

To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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



User's Manual

V850E/MA2

32-Bit Single-Chip Microcontroller

Hardware

μPD703108

Document No. U14980EJ2V1UD00 (2nd edition)
Date Published August 2005 N CP(K)

© NEC Electronics Corporation 2000
Printed in Japan

[MEMO]

NOTES FOR CMOS DEVICES

① VOLTAGE APPLICATION WAVEFORM AT INPUT PIN

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (MAX) and V_{IH} (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (MAX) and V_{IH} (MIN).

② HANDLING OF UNUSED INPUT PINS

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

③ PRECAUTION AGAINST ESD

A 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 when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up 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 benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

④ STATUS BEFORE INITIALIZATION

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

⑤ POWER ON/OFF SEQUENCE

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current.

The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

⑥ INPUT OF SIGNAL DURING POWER OFF STATE

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

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

- **The information in this document is current as of July, 2005. The information is subject to change without notice. For actual design-in, refer to the latest publications of NEC Electronics data sheets or data books, etc., for the most up-to-date specifications of NEC Electronics products. Not all products and/or types are available in every country. Please check with an NEC Electronics sales representative for availability and additional information.**

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

The "Specific" quality grade applies only to NEC Electronics products developed based on a customer-designated "quality assurance program" for a specific application. The recommended applications of an NEC Electronics product depend on its quality grade, as indicated below. Customers must check the quality grade of each NEC Electronics product before using it in a particular application.

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

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

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

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

(Note)

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

Regional Information

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

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

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

[GLOBAL SUPPORT]

<http://www.necel.com/en/support/support.html>

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

Santa Clara, California
Tel: 408-588-6000
800-366-9782

NEC Electronics (Europe) GmbH

Duesseldorf, Germany
Tel: 0211-65030

NEC Electronics Hong Kong Ltd.

Hong Kong
Tel: 2886-9318

- **Sucursal en España**

Madrid, Spain
Tel: 091-504 27 87

- **Succursale Française**

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

- **Filiale Italiana**

Milano, Italy
Tel: 02-66 75 41

- **Branch The Netherlands**

Eindhoven, The Netherlands
Tel: 040-265 40 10

- **Tyskland Filial**

Taebby, Sweden
Tel: 08-63 87 200

- **United Kingdom Branch**

Milton Keynes, UK
Tel: 01908-691-133

NEC Electronics Hong Kong Ltd.

Seoul Branch
Seoul, Korea
Tel: 02-558-3737

NEC Electronics Shanghai Ltd.

Shanghai, P.R. China
Tel: 021-5888-5400

NEC Electronics Taiwan Ltd.

Taipei, Taiwan
Tel: 02-2719-2377

NEC Electronics Singapore Pte. Ltd.

Novena Square, Singapore
Tel: 6253-8311

J05.6

Major Revisions in This Edition (1/4)

Page	Description
Throughout	<ul style="list-style-type: none"> • Deletion of BUSCLK pin • Deletion of bus cycle period control register (BCP) and port CM function control register (PFCCM)
p.27	Deletion of Note from 1.4 Ordering Information
p.37	Addition of Note to 2.2 Pin Status
p.39	Modification of description in 2.3 (3) (b) (iii) $\overline{TC0}$ (Terminal Count)
p.41	Modification of description in 2.3 (7) (b) (i) \overline{WAIT} (Wait)
p.44	Addition of description to 2.3 (10) (b) (ii) SDCLK (SDRAM Clock Output)
p.58	Modification of Caution in 3.4.3 (1) Program space
p.62	Modification of description in and addition of Caution to 3.4.5 (2) Internal RAM area
p.63	Addition of Note to 3.4.5 (3) Internal peripheral I/O area
p.66	Modification of Bit Units for Manipulation for DMA terminal count output control register in 3.4.8 Peripheral I/O registers
p.72	Modification of description in Table and addition of Remark to 3.4.10 System wait control register (VSWC)
p.74	Modification of description in 4.2.1 Pin status during internal RAM and peripheral I/O access
p.75	Addition of Note to 4.3 Memory Block Function
p.76	Addition of Caution to 4.3.1 Chip select control function
p.77	Addition of description to 4.4 (1) Bus cycle type configuration registers 0, 1 (BCT0, BCT1)
p.79	Modification of Table in 4.5.1 Number of access clocks
p.79	Addition of description to 4.5.2 (1) Bus size configuration register (BSC)
p.80	Addition of Caution to 4.5.3 (1) Endian configuration register (BEC)
p.94	Addition of Caution to 4.6.1 (2) Address setup wait control register (ASC)
p.100	Addition of description to 4.8.1 Function outline
p.109	Deletion of description from 4.10.1 Program space
p.124	Addition of Note to Figure 5-5 Page ROM Access Timing (1/4)
pp.131 and 132	Modification of description for setting of LTM2n to LTM0n bits in and addition of Caution to 5.3.4 SDRAM configuration registers 3, 4 (SCR3, SCR4)
p.147	Addition of Caution to 5.3.6 (1) SDRAM refresh control registers 3, 4 (RFS3, RFS4)
p.153	Modification of description in and addition of Note to Figure 5-14 Self Refresh Timing (SDRAM)
p.159	Addition of description to 6.3.1 (1) DMA source address registers 0H to 3H (DSA0H to DSA3H)
p.161	Addition of description to 6.3.2 (1) DMA destination address registers 0H to 3H (DDA0H to DDA3H)
p.164	Addition of Caution to 6.3.4 DMA addressing control registers 0 to 3 (DADC0 to DADC3)
p.166	Modification of description in and addition of Caution to 6.3.5 DMA channel control registers 0 to 3 (DCHC0 to DCHC3)
p.168	Addition of description to 6.3.6 DMA disable status register (DDIS)
p.168	Addition of description to 6.3.7 DMA restart register (DRST)
p.169	Modification of description in and addition of reserved word < > in device file to bits 3 to 0 to 6.3.8 DMA terminal count output control register (DTC)
p.170	Addition of Caution to 6.3.9 DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)
p.170	Addition of description and Figure to 6.5.1 Single transfer mode

Major Revisions in This Edition (2/4)

Page	Description
p.177	Addition of description to 6.5.3 Block transfer mode
p.178	Addition of Caution to 6.6 Two-Cycle Transfer
p.188	Modification of Remark in 6.7.1 Transfer type and transfer object
p.189	Addition of Caution to 6.8 DMA Channel Priorities
p.191	Addition of Figure 6-14 Terminal Count Signal ($\overline{TC0}$) Output Example
p.193	Addition of description to Remark in Figure 6-16 Example of Forcible Termination of DMA Transfer
p.194	Modification of description in Table 6-3 Number of Minimum Execution Clocks in DMA Cycle
p.195	Addition of description to and modification of description in 6.16 One-Time Transfer During Single Transfer via DMARQ0, DMARQ1 Signals
p.195	Modification of Figure 6-17 Time to Perform Single Transfer One Time
p.196	Addition of 6.17 (4) Maintenance of \overline{DMARQn} signal
p.196	Addition of 6.17 (6) DMA start factors
p.201	Modification of description in Figure 7-2 Acknowledging Non-Maskable Interrupt Request
p.211	Addition of Caution to 7.3.4 Interrupt control register (xxICn)
p.213	Addition of Caution to and deletion of reserved word < > in device file from 7.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3)
p.215	Addition of Caution to 7.3.6 In-service priority register (ISPR)
p.216	Addition of Caution to 7.3.9 (1) External interrupt mode registers 1, 2 (INTM1, INTM2)
p.218	Addition of Caution to 7.3.9 (2) Valid edge selection registers C0, C1 (SESC0, SESC1)
p.228	Modification of description in Figure 7-14 Pipeline Operation at Interrupt Request Acknowledgement (Outline)
p.229	Modification of description in 7.8 Periods in Which Interrupts Are Not Acknowledged
p.236	Addition of description to 9.3.5 Peripheral status register (PHS)
p.247	Addition of description to 9.5.4 (2) (a) Release by non-maskable interrupt request or unmasked maskable interrupt request
p.250	Addition of description to 9.5.5 (2) (a) Release by non-maskable interrupt request or unmasked maskable interrupt request
p.251	Modification of Figure in 9.6.1 (1) Securing the time using an on-chip time base counter
p.252	Modification of Figure in 9.6.1 (2) Securing the time according to the signal level width (\overline{RESET} pin input)
p.258	Addition of Caution to 10.1.4 (2) (a) Setting these registers to capture registers (CMSn0 and CMSn1 of TMCCn1 = 0)
p.260	Addition of Caution to 10.1.5 (1) Timer mode control registers C00, C10 (TMCC00, TMCC10)
pp.262 and 263	Change of bit name of bit 5 of TMCC01 register in 10.1.5 (2) Timer mode control registers C01, C11 (TMCC01, TMCC11)
p.270	Addition of Figure 10-5 Compare Operation Example (2/2)
p.278	Deletion of Caution from and addition of Note to Figure 10-12 Cycle Measurement Operation Timing Example
p.283	Modification of description in Figure 10-13 Example of Timing During TMDn Operation
p.285	Addition of Caution to 10.2.5 (1) Timer mode control registers D0 to D3 (TMCD0 to TMCD3)

Major Revisions in This Edition (3/4)

Page	Description
p.296	Modification of description for PEn bit = 0, FEn bit = 0, OVEN bit = 0 in 11.2.3 (2) Asynchronous serial interface status registers 0, 1 (ASIS0, ASIS1)
p.297	Modification of description for TXBFn bit, TXSFn bit in 11.2.3 (3) Asynchronous serial interface transmission status registers 0, 1 (ASIF0, ASIF1)
p.304	Modification of description in and addition of Figure to 11.2.5 (3) Continuous transmission operation
p.306	Modification of description in and addition of Note to Figure 11-5 Continuous Transmission Starting Procedure
p.307	Modification of description in Figure 11-6 Continuous Transmission Ending Procedure
p.309	Modification of Figure and addition of Caution to Figure 11-7 Asynchronous Serial Interface Reception Completion Interrupt Timing
p.314	Addition of Caution to 11.2.6 (2) (a) Clock select registers 0, 1 (CKSR0, CKSR1)
p.320	Addition of (2) to 11.2.7 Precautions
p.322	Addition of description to 11.3.3 (1) Clocked serial interface mode registers 0, 1 (CSIM0, CSIM1)
p.334	Addition of description to 12.2 (5) Successive approximation register (SAR)
p.338	Addition of Caution to 12.3 (2) A/D converter mode register 1 (ADM1)
p.340	Change of bit names in 12.3 (4) A/D conversion result registers (ADCR0 to ADCR3, ADCR0H to ADCR3H)
p.343	Addition of description to 12.4.2 (1) (b) Timer trigger mode
p.345	Modification of Figure 12-3 Select Mode Operation Timing: 1-Buffer Mode (ANI1)
p.346	Modification of Figure 12-4 Select Mode Operation Timing: 4-Buffer Mode (ANI2)
p.347	Modification of Figure 12-5 Scan Mode Operation Timing: 4-Channel Scan (ANI0 to ANI3)
p.359	Addition of 12.7.5 Reconversion operations in timer 1 trigger mode
p.360	Addition of 12.7.6 Supplementary information on A/D conversion time
p.362	Addition of 12.8 How to Read A/D Converter Characteristics Table
pp.368 and 369	Modification of block type in and addition of Caution to 13.2 (1) Function of each port
p.370	Modification of Register That Sets the Mode of port CM in 13.2 (2) Function when each port's pins are reset and registers that set the port/control mode
p.373	Modification of Figure 13-4 Block Diagram of Type D
p.374	Addition of Figure 13-6 Block Diagram of Type G
p.375	Modification of Figure 13-7 Block Diagram of Type H
p.378	Modification of Figure 13-10 Block Diagram of Type M
p.379	Modification of Figure 13-11 Block Diagram of Type N
p.382	Partial deletion of description from PMC0n bit = 0 in 13.3.1 (2) (b) Port 0 mode control register (PMC0)
p.387	Partial deletion of description from PMC24 bit = 0 in 13.3.3 (2) (b) Port 2 mode control register (PMC2)
p.388	Modification of block type of P40 and P43 in 13.3.4 (1) Operation in control mode
p.391	Addition of Caution to 13.3.5 (1) Operation in control mode
p.397	Addition of Caution to 13.3.8 (2) (b) Port DL mode control register (PMCDL)
p.402	Modification of block type of PCM1 in 13.3.11 (1) Operation in control mode
p.409	Addition of pin status of $\overline{\text{LBE}}$ and $\overline{\text{UBE}}$ pins to Table 14-1 Operation Status of Each Pin During Reset

Major Revisions in This Edition (4/4)

Page	Description
p.414	Addition of CHAPTER 15 ELECTRICAL SPECIFICATIONS
p.444	Addition of CHAPTER 16 PACKAGE DRAWING
p.445	Addition of CHAPTER 17 RECOMMENDED SOLDERING CONDITIONS
p.446	Addition of APPENDIX A NOTES ON TARGET SYSTEM DESIGN
p.447	Addition of APPENDIX B CAUTIONS
pp.457 to 462	Modification of description in D.2 Instruction Set (In Alphabetical Order)
Major revisions in modification version (U14980EJ2V1UD00)	
p.27	Modification of 1.4 Ordering Information
p.28	Addition of description to 1.5 Pin Configuration (Top View)
p.445	Modification of CHAPTER 17 RECOMMENDED SOLDERING CONDITIONS

The mark ★ shows major revised points.

INTRODUCTION

Readers This manual is intended for users who wish to understand the functions of the V850E/MA2 (μ PD703108) to design application systems using the V850E/MA2.

Purpose The purpose of this manual is for users to gain an understanding of the hardware functions of the VE850E/MA2.

Organization The **V850E/MA2 User's Manual** is divided into two parts: Hardware (this manual) and Architecture (**V850E1 User's Manual Architecture**). The organization of each manual is as follows:

Hardware

- Pin functions
- CPU function
- Internal peripheral functions
- Electrical specifications

Architecture

- Data type
- Register set
- Instruction format and instruction set
- Interrupt and exception
- Pipeline operation

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

- To find the details of a register where the name is known
→Refer to **APPENDIX C REGISTER INDEX**.
- To find the details of a function, etc. where the name is known
→Refer to **APPENDIX E INDEX**.
- To understand the details of an instruction function
→Refer to the **V850E1 User's Manual Architecture**.
- To know the electrical specifications of the V850E/MA2
→Refer to **CHAPTER 15 ELECTRICAL SPECIFICATIONS**.
- To understand the overall functions of the V850E/MA2
→Read this manual according to the **CONTENTS**.
- How to interpret the register format
→For the bit whose bit number is enclosed in brackets, its bit name is defined as a reserved word in the device file.

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:	Top: higher, bottom: lower
Note:	Footnote for item marked with Note in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numeric representation:	Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH
Prefix indicating power of 2 (address space, memory capacity):	K (kilo): $2^{10} = 1,024$ M (mega): $2^{20} = 1,024^2$ G (giga): $2^{30} = 1,024^3$
Data Type:	Word ... 32 bits 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.

Documents related to V850E/MA2

Document Name	Document No.
V850E1 Architecture User's Manual	U14559E
V850E/MA2 Hardware User's Manual	This manual

Documents related to development tools (user's manuals)

Document Name		Document No.
IE-V850E-MC, IE-V850E-MC-A (In-circuit emulator)		U14487E
IE-703107-MC-EM1 (In-circuit emulator option board)		U14481E
CA850 (Ver.2.30 or later) (C compiler package)	Operation	U14568E
	C Language	U14566E
	Project Manager	U14569E
	Assembly Language	U14567E
CA850 (Ver.2.40 or later) (C compiler package)	Operation	U15024E
	C Language	U15025E
	Project Manager	U15026E
	Assembly Language	U15027E
ID850 (Ver.2.40) (Integrated debugger)	Operation Windows™ Based	U15181E
SM850 (Ver.2.40) (System simulator)	Operation Windows Based	U15182E
SM850 (Ver.2.00 or later) (System simulator)	External Part User Open Interface Specification	U14873E
RX850 (Ver.3.13 or later) (Real-time OS)	Basics	U13430E
	Installation	U13410E
	Technical	U13431E
RX850 Pro (Ver.3.13) (Real-time OS)	Basics	U13773E
	Installation	U13774E
	Technical	U13772E
RD850 (Ver.3.01) (Task debugger)		U13737E
RD850 Pro (Ver.3.01) (Task debugger)		U13916E
AZ850 (Ver.3.0) (System performance analyzer)		U14410E

CONTENTS

CHAPTER 1 INTRODUCTION	24
1.1 Outline	24
1.2 Features	25
1.3 Applications	27
1.4 Ordering Information	27
1.5 Pin Configuration (Top View)	28
1.6 Function Block	30
1.6.1 Internal block diagram.....	30
1.6.2 On-chip units.....	31
CHAPTER 2 PIN FUNCTIONS	33
2.1 List of Pin Function	33
2.2 Pin Status	37
2.3 Description of Pin Functions	38
2.4 Pin I/O Circuits and Recommended Connection of Unused Pins	47
2.5 Pin I/O Circuits	49
CHAPTER 3 CPU FUNCTION	50
3.1 Features	50
3.2 CPU Register Set	51
3.2.1 Program register set	52
3.2.2 System register set	53
3.3 Operation Modes	55
3.3.1 Operation modes	55
3.3.2 Operation mode specification.....	55
3.4 Address Space	56
3.4.1 CPU address space	56
3.4.2 Image.....	57
3.4.3 Wrap-around of CPU address space	58
3.4.4 Memory map	59
3.4.5 Area	60
3.4.6 External memory expansion.....	64
3.4.7 Recommended use of address space.....	65
3.4.8 Peripheral I/O registers	66
3.4.9 Specific registers.....	72
3.4.10 System wait control register (VSWC).....	72
3.4.11 Cautions.....	72
CHAPTER 4 BUS CONTROL FUNCTION	73
4.1 Features	73
4.2 Bus Control Pins	74
4.2.1 Pin status during internal RAM and peripheral I/O access	74
4.3 Memory Block Function	75
4.3.1 Chip select control function	76
4.4 Bus Cycle Type Control Function	77

4.5	Bus Access	79
4.5.1	Number of access clocks	79
4.5.2	Bus sizing function	79
4.5.3	Endian control function.....	80
4.5.4	Big endian method usage restrictions in NEC Electronics development tools.....	81
4.5.5	Bus width.....	83
4.6	Wait Function	94
4.6.1	Programmable wait function	94
4.6.2	External wait function	97
4.6.3	Relationship between programmable wait and external wait.....	97
4.6.4	Bus cycles in which wait function is valid	98
4.7	Idle State Insertion Function	99
4.8	Bus Hold Function	100
4.8.1	Function outline	100
4.8.2	Bus hold procedure	101
4.8.3	Operation in power save mode	101
4.8.4	Bus hold timing (SRAM).....	102
4.8.5	Bus hold timing (SDRAM)	104
4.9	Bus Priority Order	108
4.10	Boundary Operation Conditions	109
4.10.1	Program space.....	109
4.10.2	Data space	109
CHAPTER 5 MEMORY ACCESS CONTROL FUNCTION		110
5.1	SRAM, External ROM, External I/O Interface	110
5.1.1	Features	110
5.1.2	SRAM connection	111
5.1.3	SRAM, external ROM, external I/O access	113
5.2	Page ROM Controller (ROMC)	119
5.2.1	Features	119
5.2.2	Page ROM connection	120
5.2.3	On-page/off-page judgment	121
5.2.4	Page ROM configuration register (PRC)	123
5.2.5	Page ROM access	124
5.3	SDRAM Controller	128
5.3.1	Features	128
5.3.2	SDRAM connection	128
5.3.3	Address multiplex function	129
5.3.4	SDRAM configuration registers 3, 4 (SCR3, SCR4).....	131
5.3.5	SDRAM access	133
5.3.6	Refresh control function	147
5.3.7	Self-refresh control function	152
5.3.8	SDRAM initialization sequence	154
CHAPTER 6 DMA FUNCTIONS (DMA CONTROLLER)		157
6.1	Features	157
6.2	Configuration	158
6.3	Control Registers	159

6.3.1	DMA source address registers 0 to 3 (DSA0 to DSA3)	159
6.3.2	DMA destination address registers 0 to 3 (DDA0 to DDA3)	161
6.3.3	DMA byte count registers 0 to 3 (DBC0 to DBC3)	163
6.3.4	DMA addressing control registers 0 to 3 (DADC0 to DADC3)	164
6.3.5	DMA channel control registers 0 to 3 (DCHC0 to DCHC3)	166
6.3.6	DMA disable status register (DDIS)	168
6.3.7	DMA restart register (DRST)	168
6.3.8	DMA terminal count output control register (DTCO)	169
6.3.9	DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)	170
6.4	DMA Bus States	172
6.4.1	Types of bus states	172
6.4.2	DMAC bus cycle state transition	173
6.5	Transfer Modes	174
6.5.1	Single transfer mode	174
6.5.2	Single-step transfer mode	176
6.5.3	Block transfer mode	177
6.6	Two-Cycle Transfer	178
6.7	Transfer Object	188
6.7.1	Transfer type and transfer object	188
6.7.2	External bus cycles during DMA transfer	188
6.8	DMA Channel Priorities	189
6.9	Next Address Setting Function	189
6.10	DMA Transfer Start Factors	190
6.11	Terminal Count Output upon DMA Transfer End	191
6.12	Forcible Interrupt	192
6.13	Forcible Termination	193
6.14	Times Related to DMA Transfer	194
6.15	Maximum Response Time for DMA Transfer Request	194
6.16	One-Time Transfer During Single Transfer via $\overline{\text{DMARQ0}}$, $\overline{\text{DMARQ1}}$ Signals	195
6.17	Cautions	196
6.17.1	Interrupt factors	196
6.18	DMA Transfer End	196
CHAPTER 7	INTERRUPT/EXCEPTION PROCESSING FUNCTION	197
7.1	Features	197
7.2	Non-Maskable Interrupt	199
7.2.1	Operation	200
7.2.2	Restore	202
7.2.3	Non-maskable interrupt status flag (NP)	203
7.2.4	Noise elimination	203
7.2.5	Edge detection function	203
7.3	Maskable Interrupts	204
7.3.1	Operation	204
7.3.2	Restore	206
7.3.3	Priorities of maskable interrupts	207
7.3.4	Interrupt control register (xxICn)	211
7.3.5	Interrupt mask registers 0 to 3 (IMR0 to IMR3)	213
7.3.6	In-service priority register (ISPR)	215

7.3.7	Maskable interrupt status flag (ID)	215
7.3.8	Noise elimination	216
7.3.9	Interrupt trigger mode selection	216
7.4	Software Exception	219
7.4.1	Operation	219
7.4.2	Restore	220
7.4.3	Exception status flag (EP)	221
7.5	Exception Trap	222
7.5.1	Illegal opcode definition	222
7.5.2	Debug trap	224
7.6	Multiple Interrupt Servicing Control	226
7.7	Interrupt Latency Time	228
7.8	Periods in Which Interrupts Are Not Acknowledged	229
CHAPTER 8 PRESCALER UNIT (PRS)		230
CHAPTER 9 CLOCK GENERATOR FUNCTION		231
9.1	Features	231
9.2	Configuration	231
9.3	Input Clock Selection	232
9.3.1	Direct mode	232
9.3.2	PLL mode	233
9.3.3	Peripheral command register (PHCMD)	233
9.3.4	Clock control register (CKC)	234
9.3.5	Peripheral status register (PHS)	236
9.4	PLL Lockup	237
9.5	Power Save Control	238
9.5.1	Overview	238
9.5.2	Control registers	240
9.5.3	HALT mode	243
9.5.4	IDLE mode	245
9.5.5	Software STOP mode	248
9.6	Securing Oscillation Stabilization Time	251
9.6.1	Oscillation stabilization time security specification	251
9.6.2	Time base counter (TBC)	253
CHAPTER 10 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)		254
10.1	Timer C	254
10.1.1	Features (timer C)	254
10.1.2	Function overview (timer C)	254
10.1.3	Basic configuration of timer C	255
10.1.4	Timer C	256
10.1.5	Timer C control registers	260
10.1.6	Timer C operation	265
10.1.7	Application examples (timer C)	272
10.1.8	Precautions (timer C)	279
10.2	Timer D	280
10.2.1	Features (timer D)	280

10.2.2	Function overview (timer D)	280
10.2.3	Basic configuration of timer D	280
10.2.4	Timer D	281
10.2.5	Timer D control registers.....	284
10.2.6	Timer D operation	286
10.2.7	Application examples (timer D)	288
10.2.8	Precautions (timer D)	288
CHAPTER 11 SERIAL INTERFACE FUNCTION.....		289
11.1	Features	289
11.1.1	Switching between UART and CSI modes.....	289
11.2	Asynchronous Serial Interfaces 0, 1 (UART0, UART1)	290
11.2.1	Features.....	290
11.2.2	Configuration.....	291
11.2.3	Control registers.....	293
11.2.4	Interrupt requests.....	300
11.2.5	Operation	301
11.2.6	Dedicated baud rate generators 0, 1 (BRG0, BRG1).....	313
11.2.7	Precautions	320
11.3	Clocked Serial Interfaces 0, 1 (CSI0, CSI1)	321
11.3.1	Features.....	321
11.3.2	Configuration.....	321
11.3.3	Control registers.....	322
11.3.4	Operation	328
11.3.5	Output pins.....	331
11.3.6	System configuration example	332
CHAPTER 12 A/D CONVERTER.....		333
12.1	Features	333
12.2	Configuration	333
12.3	Control Registers	336
12.4	A/D Converter Operation	342
12.4.1	Basic operation of A/D converter	342
12.4.2	Operation mode and trigger mode	343
12.5	Operation in A/D Trigger Mode	348
12.5.1	Select mode operation	348
12.5.2	Scan mode operations	350
12.6	Operation in Timer Trigger Mode.....	351
12.6.1	Select mode operation	352
12.6.2	Scan mode operation.....	356
12.7	Precautions in Operations	358
12.7.1	Stopping conversion operation.....	358
12.7.2	Timer trigger interval	358
12.7.3	Operation in standby mode	358
12.7.4	Compare match interrupt when in timer trigger mode	359
12.7.5	Reconversion operation in timer 1 trigger mode	359
12.7.6	Supplementary information on A/D conversion time	360
12.8	How to Read A/D Converter Characteristics Table	362

CHAPTER 13 PORT FUNCTIONS	366
13.1 Features	366
13.2 Port Configuration.....	367
13.3 Port Pin Functions	381
13.3.1 Port 0.....	381
13.3.2 Port 1.....	384
13.3.3 Port 2.....	386
13.3.4 Port 4.....	388
13.3.5 Port 7.....	391
13.3.6 Port AL.....	392
13.3.7 Port AH.....	394
13.3.8 Port DL.....	396
13.3.9 Port CS.....	398
13.3.10 Port CT.....	400
13.3.11 Port CM.....	402
13.3.12 Port CD.....	404
13.3.13 Port BD.....	407
CHAPTER 14 RESET FUNCTIONS	409
14.1 Features	409
14.2 Pin Functions.....	409
14.3 Initialization.....	411
CHAPTER 15 ELECTRICAL SPECIFICATIONS	414
CHAPTER 16 PACKAGE DRAWING.....	444
CHAPTER 17 RECOMMENDED SOLDERING CONDITIONS	445
APPENDIX A NOTES ON TARGET SYSTEM DESIGN.....	446
APPENDIX B CAUTIONS.....	447
B.1 Restriction on Page ROM Access	447
B.1.1 Description	447
B.1.2 Countermeasures.....	448
APPENDIX C REGISTER INDEX	449
APPENDIX D INSTRUCTION SET LIST	454
D.1 Conventions.....	454
D.2 Instruction Set (In Alphabetical Order).....	457
APPENDIX E INDEX	464

LIST OF FIGURES (1/4)

Figure No.	Title	Page
3-1	CPU Address Space.....	56
3-2	Images on Address Space	57
3-3	Memory Map.....	59
4-1	Big Endian Addresses Within Word.....	81
4-2	Little Endian Addresses Within Word	81
4-3	Example of Wait Insertion.....	97
5-1	Examples of Connection to SRAM	111
5-2	SRAM, External ROM, External I/O Access Timing.....	113
5-3	Examples of Connection to Page ROM	120
5-4	On-Page/Off-Page Judgment During Page ROM Connection.....	121
5-5	Page ROM Access Timing	124
5-6	Example of Connection to SDRAM.....	128
5-7	Row Address/Column Address Output	129
5-8	State Transition of SDRAM Access	133
5-9	SDRAM Single Read Cycle	135
5-10	SDRAM Single Write Cycle	139
5-11	SDRAM Access Timing	143
5-12	Auto Refresh Cycle.....	150
5-13	CBR Refresh Timing (SDRAM)	151
5-14	Self Refresh Timing (SDRAM).....	153
5-15	SDRAM Mode Register Setting Cycle	155
5-16	SDRAM Register Write Operation Timing.....	156
6-1	DMAC Bus Cycle State Transition.....	173
6-2	Single Transfer Example 1	174
6-3	Single Transfer Example 2	174
6-4	Single Transfer Example 3	175
6-5	Single-Step Transfer Example 1	176
6-6	Single-Step Transfer Example 2.....	176
6-7	Block Transfer Example	177
6-8	Timing of Access to SRAM, External ROM, and External I/O During 2-Cycle DMA Transfer	179
6-9	Timing of 2-Cycle DMA Transfer (External I/O → SRAM)	181
6-10	Timing of 2-Cycle DMA Transfer (SRAM → SDRAM)	182
6-11	Timing of 2-Cycle DMA Transfer (SDRAM → SRAM)	185
6-12	Buffer Register Configuration	189
6-13	Terminal Count Signal (TC0) Timing Example	191
6-14	Example of Terminal Count Signal (TC0) Output	191
6-15	Example of Forcible Interrupt of DMA Transfer	192
6-16	Example of Forcible Termination of DMA Transfer.....	193
6-17	Time to Perform Single Transfer One Time.....	195
7-1	Servicing Configuration of Non-Maskable Interrupt	200

LIST OF FIGURES (2/4)

Figure No.	Title	Page
7-2	Acknowledging Non-Maskable Interrupt Request	201
7-3	RETI Instruction Processing	202
7-4	Maskable Interrupt Servicing	205
7-5	RETI Instruction Processing	206
7-6	Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Processed.....	208
7-7	Example of Processing Interrupt Requests Simultaneously Generated.....	210
7-8	Software Exception Processing	219
7-9	RETI Instruction Processing	220
7-10	Exception Trap Processing	223
7-11	Restore Processing from Exception Trap	223
7-12	Debug Trap Processing	224
7-13	Restore Processing from Debug Trap	225
7-14	Pipeline Operation at Interrupt Request Acknowledgement (Outline).....	228
9-1	Power Save Mode State Transition Diagram	239
10-1	Basic Operation of Timer C.....	265
10-2	Operation After Overflow (When OSTn = 1)	266
10-3	Capture Operation Example	267
10-4	TMC1 Capture Operation Example (When Both Edges Are Specified)	268
10-5	Compare Operation Example	269
10-6	TMC0 Compare Operation Example (Set/Reset Output Mode)	271
10-7	Contents of Register Settings When Timer C Is Used as Interval Timer.....	272
10-8	Interval Timer Operation Timing Example.....	273
10-9	Contents of Register Settings When Timer C Is Used for PWM Output	274
10-10	PWM Output Timing Example.....	275
10-11	Contents of Register Settings When Timer C Is Used for Cycle Measurement	277
10-12	Cycle Measurement Operation Timing Example.....	278
10-13	Example of Timing During TMDn Operation	283
10-14	TMD0 Compare Operation Example.....	286
11-1	Asynchronous Serial Interface Block Diagram.....	292
11-2	Asynchronous Serial Interface Transmit/Receive Data Format	301
11-3	Asynchronous Serial Interface Transmission Completion Interrupt Timing.....	303
11-4	Continuous Transmission Processing Flow	305
11-5	Continuous Transmission Starting Procedure.....	306
11-6	Continuous Transmission Ending Procedure.....	307
11-7	Asynchronous Serial Interface Reception Completion Interrupt Timing	309
11-8	When Reception Error Interrupt Is Separated from INTSRn Interrupt (ISRMn Bit = 0)	310
11-9	When Reception Error Interrupt Is Included in INTSRn Interrupt (ISRMn Bit = 1).....	310
11-10	Noise Filter Circuit	312
11-11	Timing of RXDn Signal Judged as Noise	312
11-12	Baud Rate Generator Configuration	313

LIST OF FIGURES (3/4)

Figure No.	Title	Page
11-13	Allowable Baud Rate Range During Reception	318
11-14	Transfer Rate During Continuous Transmission.....	320
11-15	Clocked Serial Interface Block Diagram	322
11-16	Transfer Timing	329
11-17	Clock Timing.....	330
11-18	System Configuration Example of CSI	332
12-1	Block Diagram of A/D Converter.....	335
12-2	Relationship Between Analog Input Voltage and A/D Conversion Results.....	341
12-3	Select Mode Operation Timing: 1-Buffer Mode (ANI1)	345
12-4	Select Mode Operation Timing: 4-Buffer Mode (ANI2)	346
12-5	Scan Mode Operation Timing: 4-Channel Scan (ANI0 to ANI3)	347
12-6	Example of 1-Buffer Mode (A/D Trigger Select 1-Buffer) Operation	348
12-7	Example of 4-Buffer Mode (A/D Trigger Select 4-Buffer) Operation	349
12-8	Example of Scan Mode (A/D Trigger Scan) Operation	350
12-9	Example of 1-Trigger Mode (Timer Trigger Select 1-Buffer 1-Trigger) Operation.....	352
12-10	Example of 4-Trigger Mode (Timer Trigger Select 1-Buffer 4-Trigger) Operation.....	353
12-11	Example of 1-Trigger Mode (Timer Trigger Select 4-Buffer 1-Trigger) Operation.....	354
12-12	Example of 4-Trigger Mode (Timer Trigger Select 4-Buffer 4-Trigger) Operation.....	355
12-13	Example of 1-Trigger Mode (Timer Trigger Scan 1-Trigger) Operation	356
12-14	Example of 4-Trigger Mode (Timer Trigger Scan 4-Trigger) Operation	357
12-15	A/D Trigger Mode A/D Conversion Time: When ADM1 = 00H	360
12-16	Timer Trigger Mode A/D Conversion Time: When ADM1 = 20H or 30H.....	360
12-17	A/D Conversion Outline: One A/D Conversion, FR0 to FR2 Bits of ADM1 Register = 000 (96 Clocks).....	361
12-18	Overall Error	362
12-19	Quantization Error	363
12-20	Zero-Scale Error	363
12-21	Full-Scale Error.....	364
12-22	Differential Linearity Error.....	364
12-23	Integral Linearity Error	365
12-24	Sampling Time.....	365
13-1	Block Diagram of Type A.....	371
13-2	Block Diagram of Type B.....	372
13-3	Block Diagram of Type C.....	372
13-4	Block Diagram of Type D.....	373
13-5	Block Diagram of Type F	373
13-6	Block Diagram of Type G	374
13-7	Block Diagram of Type H.....	375
13-8	Block Diagram of Type J	376
13-9	Block Diagram of Type K.....	377
13-10	Block Diagram of Type M	378
13-11	Block Diagram of Type N.....	379
13-12	Block Diagram of Type O	380

LIST OF FIGURES (4/4)

Figure No.	Title	Page
A-1	100-Pin Plastic LQFP (Fine Pitch) (14 × 14)	446
B-1	Example of Structure of Memory Map with Error	447
B-2	Example of Structure of Memory Map Preventing Error	448

LIST OF TABLES

Table No.	Title	Page
3-1	Program Registers.....	52
3-2	System Register Numbers.....	53
3-3	Interrupt/Exception Table	61
4-1	Bus Cycles in Which Wait Function Is Valid	98
4-2	Bus Priority Order	108
5-1	Example of Interval Factor Settings.....	149
6-1	Relationship Between Transfer Type and Transfer Object	188
6-2	External Bus Cycles During DMA Transfer.....	188
6-3	Number of Minimum Execution Clocks in DMA Cycle	194
7-1	Interrupt/Exception Source List.....	198
7-2	Address and Bits of Interrupt Control Register	212
9-1	Clock Generator Operation Using Power Save Control.....	239
9-2	Operation Status in HALT Mode.....	243
9-3	Operation After HALT Mode Is Released by Interrupt Request.....	244
9-4	Operation Status in IDLE Mode	246
9-5	Operation After IDLE Mode Is Released by Interrupt Request	247
9-6	Operation Status in Software STOP Mode	249
9-7	Operation After Software STOP Mode Is Released by Interrupt Request	250
9-8	Counting Time Examples ($f_{xx} = 10 \times f_x$)	253
10-1	Timer C Configuration	255
10-2	TO00 Output Control	271
10-3	Timer D Configuration	280
11-1	Generated Interrupts and Default Priorities	300
11-2	Reception Error Causes	309
11-3	Baud Rate Generator Setting Data.....	317
11-4	Maximum and Minimum Allowable Baud Rate Error	319
14-1	Operation Status of Each Pin During Reset	409
14-2	Initial Value of CPU, Internal RAM, and On-Chip Peripheral I/O After Reset	411

CHAPTER 1 INTRODUCTION

The V850E/MA2 is a product of NEC Electronics' single-chip microcontroller "V850 Series". This chapter gives a simple outline of the V850E/MA2.

1.1 Outline

The V850E/MA2 is a 32-bit single-chip microcontroller that integrates the V850E1 CPU, which is a 32-bit RISC-type CPU core for ASIC, newly developed as the CPU core central to system LSI for the current age of system-on-chip. This device incorporates RAM, and various peripheral functions such as memory controllers, a DMA controller, real-time pulse unit, serial interfaces, and an A/D converter for realizing high-capacity data processing and sophisticated real-time control.

(1) V850E1 CPU

The V850E1 CPU is a CPU core that enhances the external bus interface performance of the V850 CPU, which is the CPU core integrated in the V850 Series, and has added instructions supporting high-level languages, such as C-language switch statement processing, table lookup branching, stack frame creation/deletion, and data conversion. This enhances the performance of both data processing and control. It is possible to use the software resources of the V850 CPU integrated system since the instruction codes of the V850E1 are upwardly compatible at the object code level with those of the V850 CPU.

(2) External memory interface function

The V850E/MA2 features various on-chip external memory interfaces including separately configured address (25 bits) and data (16 bits) buses, and SDRAM and ROM interfaces, as well as on-chip memory controllers that can be directly linked to page ROM, etc., thereby raising 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 to and from the external memory, so it is possible to process large volumes of image data or voice data, etc., and through 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/MA2 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, and 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: 83
- Minimum instruction execution time: 25 ns (at internal 40 MHz operation)
- General-purpose registers: 32 bits × 32
- Instruction set:
 - V850E1 CPU
 - Signed multiplication (16 bits × 16 bits → 32 bits or 32 bits × 32 bits → 64 bits): 1 to 2 clocks
 - Saturated operation instructions (with overflow/underflow detection function)
 - 32-bit shift instructions: 1 clock
 - Bit manipulation instructions
 - Load/store instructions with long/short format
 - Signed load instructions
- Memory space:
 - 80 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 separated)
 - 16-/8-bit bus sizing function
 - Bus hold function
 - External wait function
 - Address setup wait function
 - Endian control function
- Internal memory: Internal RAM: 4 KB
- Interrupts/exceptions:
 - External interrupts: 8 (including NMI)
 - Internal interrupts: 23 sources
 - Exceptions: 1 source
 - Eight levels of priorities can be set.
- Memory access controller:
 - SDRAM controller
 - Page-ROM controller

- DMA controller:
 - 4 channels
 - Transfer units: 8 bits/16 bits
 - Maximum transfer count: 65,536 (2^{16})
 - Transfer type: 2-cycle
 - Transfer mode: Single/Single step/Block
 - Transfer target: Memory ↔ memory, memory ↔ I/O
 - Transfer request: External request/On-chip peripheral I/O/ Software
 - DMA transfer terminate (terminal count) output signal
 - Next address setting function

- I/O lines:
 - Input ports: 5
 - I/O ports: 74

- Real-time pulse unit:
 - 16-bit timer/event counter: 2 channels
 - 16-bit timers: 2
 - 16-bit capture/compare registers: 4
 - 16-bit interval timer: 4 channels

- Serial interfaces (SIO):
 - Asynchronous serial interface (UART)
 - Clocked serial interface (CSI)
 - CSI/UART: 2 channels

- A/D converter:
 - 10-bit resolution A/D converter: 4 channels

- Clock generator:
 - A 10-multiplication function through a PLL clock synthesizer.
 - Divide-by-two function through an external clock input.

- Power save function:
 - HALT/IDLE/software STOP mode

- Package:
 - 100-pin plastic LQFP (Fine pitch) (14 × 14)

- CMOS technology:
 - All static circuits

1.3 Applications

Ink-jet printers, facsimiles, digital still cameras, DVD players, video printers, PPCs, information equipment, etc.

★

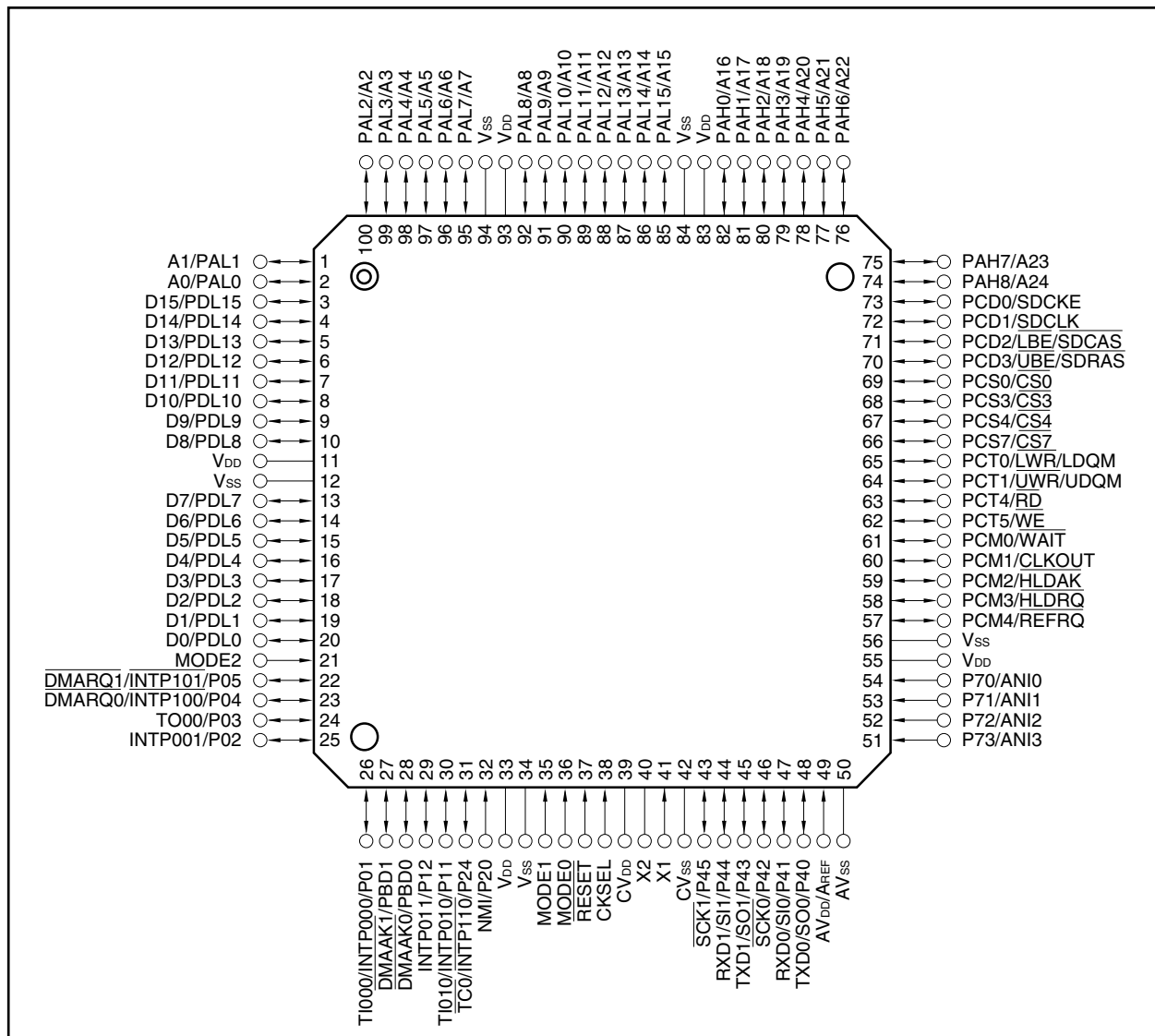
1.4 Ordering Information

Part Number	Package	Internal ROM	Internal RAM
μ PD703108GC-8EU	100-pin plastic LQFP (Fine pitch) (14 × 14)	None	4 KB
μ PD703108GC-8EU-A	100-pin plastic LQFP (Fine pitch) (14 × 14)	None	4 KB

Remark Products with -A at the end of the part number are lead-free products.

1.5 Pin Configuration (Top View)

- 100-pin plastic LQFP (Fine pitch) (14 × 14)
- μ PD703108GC-8EU
- ★ μ PD703108GC-8EU-A

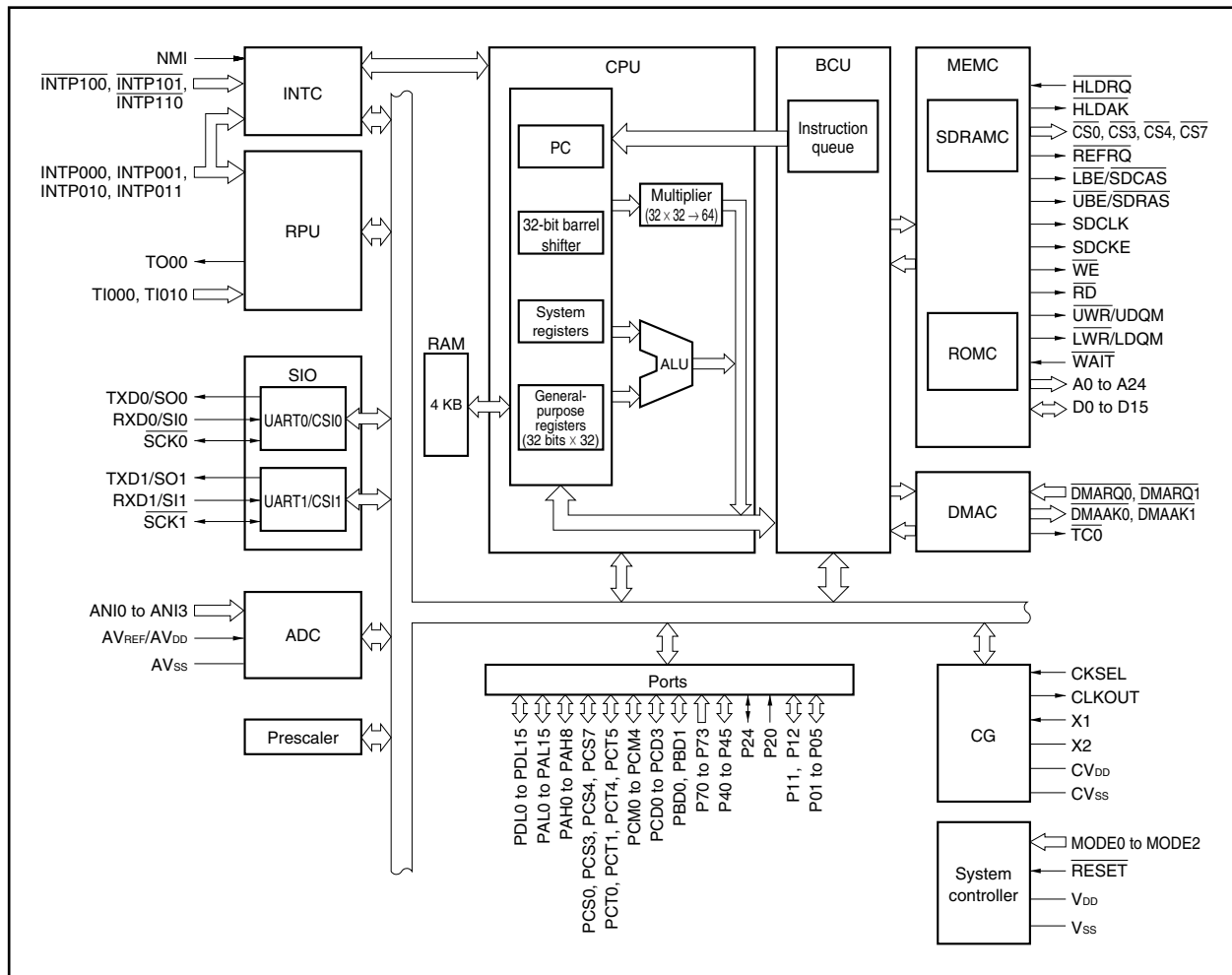


Pin Identification

A0 to A24:	Address bus	PBD0, PBD1:	Port BD
ANI0 to ANI3:	Analog input	PCD0 to PCD3:	Port CD
AV _{DD} :	Analog power supply	PCM0 to PCM4:	Port CM
AV _{REF} :	Analog reference voltage	PCS0, PCS3, :	Port CS
AV _{SS} :	Analog ground	PCS4, PCS7	
CKSEL:	Clock generator operating mode select	PCT0, PCT1, :	Port CT
CLKOUT:	Clock output	PCT4, PCT5	
$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$:	Chip select	PDL0 to PDL15:	Port DL
CV _{DD} :	Clock generator power supply	\overline{RD} :	Read strobe
CV _{SS} :	Clock generator ground	\overline{REFRQ} :	Refresh request
D0 to D15:	Data bus	\overline{RESET} :	Reset
$\overline{DMAAK0}$, $\overline{DMAAK1}$:	DMA acknowledge	RXD0, RXD1:	Receive data
$\overline{DMARQ0}$, $\overline{DMARQ1}$:	DMA request	$\overline{SCK0}$, $\overline{SCK1}$:	Serial clock
\overline{HLDK} :	Hold acknowledge	\overline{SDCAS} :	SDRAM column address strobe
\overline{HLDRQ} :	Hold request	SDCKE:	SDRAM clock enable
INTP000, INTP001, :	Interrupt request from peripherals	SDCLK:	SDRAM clock output
INTP010, INTP011,		\overline{SDRAS} :	SDRAM row address strobe
$\overline{INTP100}$, $\overline{INTP101}$,		SI0, SI1:	Serial input
$\overline{INTP110}$		SO0, SO1:	Serial output
\overline{LBE} :	Lower byte enable	$\overline{TC0}$:	Terminal count signal
LDQM:	Lower DQ mask enable	TI000, TI010:	Timer input
\overline{LWR} :	Lower write strobe	TO00:	Timer output
MODE0 to MODE2:	Mode	TXD0, TXD1:	Transmit data
NMI:	Non-maskable interrupt request	\overline{UBE} :	Upper byte enable
P01 to P05:	Port 0	UDQM:	Upper DQ mask enable
P11, P12:	Port 1	\overline{UWR} :	Upper write strobe
P20, P24:	Port 2	V _{DD} :	Power supply
P40 to P45:	Port 4	V _{SS} :	Ground
P70 to P73:	Port 7	\overline{WAIT} :	Wait
PAH0 to PAH8:	Port AH	\overline{WE} :	Write enable
PAL0 to PAL15:	Port AL	X1, X2:	Crystal

1.6 Function Block

★ 1.6.1 Internal block diagram



1.6.2 On-chip 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 the multiplier (16 bits × 16 bits → 32 bits or 32 bits × 32 bits → 64 bits) and the 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 area 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 controls an SDRAM controller (SDRAMC), page ROM controller (ROMC), and DMA controller (DMAC) and performs external memory access and DMA transfer.

(a) SDRAM controller (SDRAMC)

The DRAM controller generates the $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, $\overline{\text{UDQM}}$, and $\overline{\text{LDQM}}$ signals and performs access control for SDRAM.

CAS latency 2 and 3 are supported, and the burst length is fixed to 1.

A refresh function that supports the CBR refresh cycle and a dynamic self refresh function based on an external input are also available.

(b) Page ROM controller (ROMC)

This controller supports accessing ROM that includes the 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 128 bytes.

(c) DMA controller (DMAC)

Instead of the CPU, this controller controls data transfer between memory and I/O.

There is an address mode, 2-cycle transfer. There are three bus modes, single transfer, single step transfer, and block transfer.

(3) RAM

RAM is mapped from address FFFFC000H.

During instruction fetch or data access, data can be accessed from the CPU in 1-clock cycles.

(4) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP0n0, INTP0n1, $\overline{\text{INTP10n}}$, $\overline{\text{INTP110}}$) from on-chip peripheral I/O and external hardware (n = 0, 1). Eight levels of interrupt priorities can be specified for these interrupt requests, and multiple-interrupt servicing control can be performed for interrupt sources.

(5) Clock generator (CG)

This clock generator supplies frequencies which are 10 times the input clock (f_x) (using an on-chip PLL) or 1/2 the input clock (when an on-chip PLL is not used) as the internal system clock (f_{xx}). As the input clock, an external oscillator is connected to pins X1 and X2 (only when an on-chip PLL synthesizer is used) or an external clock is input from pin X1.

(6) Real-time pulse unit (RPU)

This unit has a 2-channel 16-bit timer/event counter and 4-channel 16-bit interval timer built in, and can measure pulse widths or frequency and output a programmable pulse.

(7) Serial interfaces (SIO)

Two channels of serial interfaces are provided for which an asynchronous serial interface (UART) and clocked serial interface (CSI) modes are selectable.

UART transfers data by using the TXD $_n$ and RXD $_n$ pins ($n = 0, 1$).

CSI transfers data by using the SOn, SIn, and $\overline{\text{SCK}}_n$ pins ($n = 0, 1$).

(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-purpose port functions and control pin functions.

Port	Port Function	Control Function
Port 0	5-bit I/O	Real-time pulse unit I/O, external interrupt input, DMA controller input
Port 1	2-bit I/O	Real-time pulse unit input, external interrupt input
Port 2	1-bit input, 1-bit I/O	NMI input, external interrupt input, DMA controller output
Port 4	6-bit I/O	Serial interface I/O
Port 7	4-bit input	A/D converter input
Port AL	16-bit I/O	External address bus
Port AH	9-bit I/O	External address bus
Port DL	16-bit I/O	External data bus
Port CS	4-bit I/O	External bus interface control signal output
Port CT	4-bit I/O	External bus interface control signal output
Port CM	5-bit I/O	Wait insertion signal input, internal system clock output, external bus interface control signal I/O
Port CD	4-bit I/O	External bus interface control signal output
Port BD	2-bit I/O	DMA controller output

CHAPTER 2 PIN FUNCTIONS

The names and functions of the pins in the V850E/MA2 are listed below. These pins can be divided into port pins and non-port pins according to their functions.

2.1 List of Pin Function

(1) Port pins

(1/2)

Pin Name	I/O	Function	Alternate Function
P01	I/O	Port 0 5-bit I/O port Input/output mode can be specified in 1-bit units.	TI000/INTP000
P02			INTP001
P03			TO00
P04			DMARQ0/INTP100
P05			DMARQ1/INTP101
P11	I/O	Port 1 2-bit I/O port Input/output mode can be specified in 1-bit units.	INTP010/TI010
P12			INTP011
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. P24 is an I/O port.	NMI
P24	I/O		$\overline{TC0}$ /INTP110
P40	I/O	Port 4 6-bit I/O port Input/output mode can be specified in 1-bit units.	TXD0/SO0
P41			RXD0/SI0
P42			$\overline{SCK0}$
P43			TXD1/SO1
P44			RXD1/SI1
P45			$\overline{SCK1}$
P70 to P73	Input	Port 7 4-bit input only port	ANI0 to ANI3
PBD0, PBD1	I/O	Port BD 2-bit I/O port Input/output mode can be specified in 1-bit units.	DMAAK0, DMAAK1
PCM0	I/O	Port CM 5-bit I/O port Input/output mode can be specified in 1-bit units.	\overline{WAIT}
PCM1			CLKOUT
PCM2			HLDK
PCM3			\overline{HLDRQ}
PCM4			\overline{REFRQ}

★

Pin Name	I/O	Function	Alternate Function
PCT0	I/O	Port CT 4-bit I/O port Input/output mode can be specified in 1-bit units.	$\overline{\text{LWR/LDQM}}$
PCT1			$\overline{\text{UWR/UDQM}}$
PCT4			$\overline{\text{RD}}$
PCT5			$\overline{\text{WE}}$
PCS0	I/O	Port CS 4-bit I/O port Input/output mode can be specified in 1-bit units.	$\overline{\text{CS0}}$
PCS3			$\overline{\text{CS3}}$
PCS4			$\overline{\text{CS4}}$
PCS7			$\overline{\text{CS7}}$
PCD0	I/O	Port CD 4-bit I/O port Input/output mode can be specified in 1-bit units.	$\overline{\text{SDCKE}}$
PCD1			$\overline{\text{SDCLK}}$
PCD2			$\overline{\text{LBE/SDCAS}}$
PCD3			$\overline{\text{UBE/SDRAS}}$
PAH0 to PAH8	I/O	Port AH 8-/9-bit I/O port Input/output mode can be specified in 1-bit units.	A16 to A24
PAL0 to PAL15	I/O	Port AL 8-/16-bit I/O port Input/output mode can be specified in 1-bit units.	A0 to A15
PDL0 to PDL15	I/O	Port DL 8-/16-bit I/O port Input/output mode can be specified in 1-bit units.	D0 to D15

(2) Non-port pins

(1/2)

Pin Name	I/O	Function	Alternate Function
TO00	Output	Pulse signal output of timer C0	P03
TI000	Input	External count clock input of timer C0, C1	P01/INTP000
TI010			P11/INTP010
INTP000	Input	External maskable interrupt request input, or timer C0 external capture trigger input	P01/TI000
INTP001			P02
INTP010		External maskable interrupt request input, or timer C1 external capture trigger input	P11/TI010
INTP011			P12
$\overline{\text{INTP100}}$	Input	External maskable interrupt request input	P04/ $\overline{\text{DMARQ0}}$
$\overline{\text{INTP101}}$			P05/ $\overline{\text{DMARQ1}}$
$\overline{\text{INTP110}}$			P24/ $\overline{\text{TC0}}$
SO0	Output	CSI0, SCI1 serial transmission data output (3-wire)	P40/TXD0
SO1			P43/TXD1
SI0	Input	CSI0, CSI1 serial reception data input (3-wire)	P41/RXD0
SI1			P44/RXD1
$\overline{\text{SCK0}}$	I/O	CSI0, CSI1 serial clock I/O (3-wire)	P42
$\overline{\text{SCK1}}$			P45
TXD0	Output	UART0, UART1 serial transmission data output	P40/SO0
TXD1			P43/SO1
RXD0	Input	UART0, UART1 serial reception data input	P41/SI0
RXD1			P44/SI1
ANI0 to ANI3	Input	Analog inputs to the A/D converter	P70 to P73
$\overline{\text{DMARQ0}}$	Input	DMA request signal input	P04/ $\overline{\text{INTP100}}$
$\overline{\text{DMARQ1}}$			P05/ $\overline{\text{INTP101}}$
$\overline{\text{DMAAK0}}$	Output	DMA acknowledge signal output	PBD0
$\overline{\text{DMAAK1}}$			PBD1
$\overline{\text{TC0}}$	Output	DMA transfer termination (terminal count) signal output	P24/ $\overline{\text{INTP110}}$
NMI	Input	Non-maskable interrupt request input	P20
MODE0 to MODE2	Input	V850E/MA2 operating mode specification	—
$\overline{\text{WAIT}}$	Input	Control signal input that inserts a wait in the bus cycle	PCM0
$\overline{\text{HLDAK}}$	Output	Bus hold acknowledge output	PCM2
$\overline{\text{HLDRQ}}$	Input	Bus hold request input	PCM3
$\overline{\text{REFRQ}}$	Output	Refresh request signal output for DRAM	PCM4
$\overline{\text{LWR}}$	Output	External data lower byte write strobe signal output	PCT0/ $\overline{\text{LDQM}}$
$\overline{\text{UWR}}$	Output	External data higher byte write strobe signal output	PCT1/ $\overline{\text{UDQM}}$
$\overline{\text{LDQM}}$	Output	Output disable/write mask signal output for SDRAM lower data	PCT0/ $\overline{\text{LWR}}$
$\overline{\text{UDQM}}$	Output	Output disable/write mask signal output for SDRAM higher data	PCT1/ $\overline{\text{UWR}}$
$\overline{\text{RD}}$	Output	External data bus read strobe signal output	PCT4
$\overline{\text{WE}}$	Output	Write enable signal output for SDRAM	PCT5

Pin Name	I/O	Function	Alternate Function
$\overline{\text{CS0}}$	Output	Chip select signal output	PCS0
$\overline{\text{CS3}}$			PCS3
$\overline{\text{CS4}}$			PCS4
$\overline{\text{CS7}}$			PCS7
SDCKE	Output	SDRAM clock enable signal output	PCD0
SDCLK	Output	SDRAM clock signal output	PCD1
$\overline{\text{SDCAS}}$	Output	Column address strobe signal output for SDRAM	PCD2/ $\overline{\text{LBE}}$
$\overline{\text{SDRAS}}$	Output	Row address strobe signal output for SDRAM	PCD3/ $\overline{\text{UBE}}$
$\overline{\text{LBE}}$	Output	External data bus lower byte enable signal output	PCD2/ $\overline{\text{SDCAS}}$
$\overline{\text{UBE}}$	Output	External data bus higher byte enable signal output	PCD3/ $\overline{\text{SDRAS}}$
D0 to D15	I/O	16-bit data bus for external memory	PDL0 to PDL15
A0 to A15	Output	25-bit address bus for external memory	PAL0 to PAL15
A16 to A24			PAH0 to PAH8
$\overline{\text{RESET}}$	Input	System reset input	–
X1	Input	Connects the crystal resonator for system clock oscillation. In the case of an external source supplying the clock, it is input to X1.	–
X2	–		–
CLKOUT	Output	System clock output	PCM1
CKSEL	Input	Input which specifies the clock generator's operating mode	–
AV _{REF}	Input	Reference voltage applied to A/D converter	AV _{DD}
AV _{DD}	–	Positive power supply for A/D converter	AV _{REF}
AV _{SS}	–	Ground potential for A/D converter	–
CV _{DD}	–	Positive power supply for the exclusive clock generator	–
CV _{SS}	–	Ground potential for the exclusive clock generator	–
V _{DD}	–	Positive power supply	–
V _{SS}	–	Ground potential	–

2.2 Pin Status

The status of each pin after reset, in power-save mode (software STOP, IDLE, HALT modes), and during DMA transfer, refresh, and bus hold (TH) is shown below.

★

Pin \ Operating Status	Reset	IDLE Mode/Software STOP Mode	HALT Mode/During DMA Transfer, Refresh	Bus Hold (TH) ^{Note}
A0 to A15 (PAL0 to PAL15)	Hi-Z	Hi-Z	Operating	Hi-Z
A16 to A24 (PAH0 to PAH8)	Hi-Z	Hi-Z	Operating	Hi-Z
D0 to D15 (PDL0 to PDL15)	Hi-Z	Hi-Z	Operating	Hi-Z
$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$ (PCS0, PCS3, PCS4, PCS7)	Hi-Z	H	Operating	Hi-Z
\overline{LWR} , \overline{UWR} (PCT0, PCT1)	Hi-Z	H	Operating	Hi-Z
LDQM, UDQM (PCT0, PCT1)	—	H	Operating	Hi-Z
\overline{RD} (PCT4)	Hi-Z	H	Operating	Hi-Z
\overline{WE} (PCT5)	Hi-Z	H	Operating	Hi-Z
\overline{WAIT} (PCM0)	Hi-Z	—	Operating	—
CLKOUT (PCM1)	Operating	L	Operating	Operating
\overline{HLDAK} (PCM2)	Hi-Z	H	Operating	L
\overline{HLDRQ} (PCM3)	Hi-Z	—	Operating	Operating
\overline{REFRQ} (PCM4)	Hi-Z	L	Operating	Operating
SDCKE (PCD0)	Hi-Z	L	Operating	Operating
SDCLK (PCD1)	Hi-Z	L	Operating	Operating
\overline{SDCAS} (PCD2)	—	SELF	Operating	Hi-Z
\overline{LBE} (PCD2)	Hi-Z	H	Operating	Hi-Z
\overline{SDRAS} (PCD3)	—	SELF	Operating	Hi-Z
\overline{UBE} (PCD3)	Hi-Z	H	Operating	Hi-Z
DMAAK0, DMAAK1 (PBD0, PBD1)	Hi-Z	H	Operating	H

★

Note Pins set to the port mode retain their previous state.

Remark Hi-Z: High-impedance
H: High-level output
L: Low-level output
—: No sampling of input
SELF: Self refresh state when pins are connected to SDRAM

2.3 Description of Pin Functions

(1) P01 to P05 (Port 0) ... 3-state I/O

P01 to P05 constitute a 5-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as an I/O port, in the control mode, these pins operate as an input/output for the real-time pulse unit (RPU), external interrupt request inputs, and DMA request inputs.

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

(a) Port mode

P01 to P05 can be set to input or output in 1-bit units by the port 0 mode register (PM0).

(b) Control mode

P01 to P05 can be set to port/control mode in 1-bit units by the PMC0 register.

(i) TI000 (Timer Input) ... input

This is the external count clock input pin for timer C0.

(ii) TO00 (Timer Output) ... output

This pin outputs the pulse signals for timer C0.

(iii) INTP000, INTP001 (Interrupt Request from Peripherals) ... input

These are external interrupt request input pins and the external capture trigger input pins for timer C0.

(iv) $\overline{\text{INTP100}}$, $\overline{\text{INTP101}}$ (Interrupt Request from Peripherals) ... input

These are external interrupt request input pins.

(v) $\overline{\text{DMARQ0}}$, $\overline{\text{DMARQ1}}$ (DMA Request) ... input

These are DMA service request signals. They correspond to DMA channels 0 and 1, respectively, and operate independently of each other. The priority order is fixed to $\overline{\text{DMARQ0}} > \overline{\text{DMARQ1}}$.

These signals are sampled at the rising edge of the CLKOUT signal. Maintain an active level until a DMA request is acknowledged.

(2) P11, P12 (Port 1) ... 3-state I/O

P11 and P12 constitute a 2-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as an I/O port, in the control mode, these pins operate as an input for the real-time pulse unit (RPU) and external interrupt request inputs.

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

(a) Port mode

P11 and P12 can be set to input or output in 1-bit units by the port 1 mode register (PM1).

(b) Control Mode

P11 and P12 can be set to port/control mode in 1-bit units by the PMC1 register.

(i) TI010 (Timer Input) ... input

This is the external count clock input pin for timer C1.

(ii) INTP010, INTP011 (Interrupt Request from Peripherals) ... input

These are external interrupt request input pins and the external capture trigger input pins for timer C1.

(3) P20, P24 (Port 2) ... 3-state I/O

P20 of port 2 is an input-only port and P24 is an I/O port.

Besides functioning as an I/O port, in the control mode, P24 operates as external interrupt request inputs and DMA transfer termination outputs (terminal count). The port/control mode is specified by the port 2 mode control register (PMC2).

(a) Port mode

P24 can be set to input or output by the port 2 mode register (PM2). P20 is an input-only port, and if a valid edge is input, it operates as an NMI input.

(b) Control mode

P24 can be set to port/control mode by the PMC2 register.

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

This is the non-maskable interrupt request input pin.

(ii) $\overline{\text{INTP110}}$ (Interrupt Request from Peripherals) ... input

These are external interrupt request input pins.

(iii) $\overline{\text{TC0}}$ (Terminal Count) ... output

These are signals from the DMA controller indicating that DMA transfer is complete. These signals become active for 1 clock at the rising edge of the CLKOUT signal.

★

(4) P40 to P45 (Port 4) ... 3-state I/O

P40 to P45 constitute a 6-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as an I/O port, in the control mode, these pins operate as input/outputs for the serial interfaces (UART0/CSI0, UART1/CSI1).

The operation mode can be set to port or control mode in 1-bit units, specified by the port 4 mode control register (PMC4).

(a) Port mode

P40 to P45 can be set to input or output in 1-bit units by the port 4 mode register (PM4).

(b) Control mode

P40 to P45 can be set to port/control mode in 1-bit units by the PMC4 register.

(i) TXD0, TXD1 (Transmit Data) ... output

These pins output UART0, UART1 serial transmit data.

(ii) RXD0, RXD1 (Receive Data) ... input

These pins input UART0, UART1 serial receive data.

(iii) SO0, SO1 (Serial Output) ... output

These pins output CSI0, CSI1 serial transmit data.

(iv) SI0, SI1 (Serial Input) ... input

These pins input CSI0, CSI1 serial receive data.

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

These are the CSI0, CSI1 serial clock I/O pins.

(5) P70 to P73 (Port 7) ... 3-state I/O

P70 to P73 constitute a 4-bit input-only port in which all pins are fixed as input pins.

Besides functioning as a port, in the control mode, these pins operate as analog inputs for the A/D converter. However, the input ports and analog input pins cannot be switched.

(a) Port mode

P70 to P73 are input-only pins.

(b) Control mode

P70 to P73 have alternate functions as 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_{SS} to prevent noise-related operation faults. Also, do not apply voltage that is outside the range for AV_{SS} and AV_{REF} to pins that are being used as inputs for the A/D converter. If it is possible for noise above the AV_{REF} range or below the AV_{SS} to enter, clamp these pins using a diode that has a small V_F value.

(6) PBD0, PBD1 (Port BD) ... 3-state I/O

PBD0 and PBD1 constitute a 2-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as an I/O port, in the control mode, these pins operate as the DMA acknowledge outputs. The operation mode can be set to port or control in 1-bit units, specified by the port BD mode control register (PMCBD).

(a) Port mode

PBD0 and PBD1 can be set to input or output in 1-bit units by the port BD mode register (PMBD).

(b) Control Mode

PBD0 and PBD1 can be set to port/control mode in 1-bit units by the PMCBD register.

(i) $\overline{\text{DMAAK0}}$, $\overline{\text{DMAAK1}}$ (DMA Acknowledge) ... output

These signals show that a DMA service request was permitted. They correspond to DMA channel 0 and 1, respectively, and operate independently of each other.

This becomes active only when external memory is being accessed. When DMA transfers are being executed between internal RAM and internal peripheral I/O, it does not become active.

This signal is activated at the rising edge of the CLKOUT signal in the T0, T1R, T1FH state of the DMA cycle, and is retained at an active level during DMA transfers.

(7) PCM0 to PCM4 (Port CM) ... 3-state I/O

PCM0 to PCM4 constitute a 5-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode, these pins operate as the wait insertion signal input, system clock output, bus hold control signal, and refresh request signal output for DRAM.

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

(a) Port mode

PCM0 to PCM4 can be set to input or output in 1-bit units by the port CM mode register (PMCM).

★

(b) Control mode

PCM0 to PCM4 can be set to port/control mode in 1-bit units by the PMCCM register.

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

★

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

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

This is the internal system clock output pin.

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

In this mode, this pin is the acknowledge signal output pin that indicates high impedance status for the address bus, data bus, and control bus when the V850E/MA2 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.

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

In this mode, this pin is the input pin by which an external device requests the V850E/MA2 to release the address bus, data bus, and control bus. This pin accepts asynchronous input for CLKOUT. When this pin is active, the address bus, data bus, and control bus are set to high impedance status. This occurs either when the V850E/MA2 completes execution of the current bus cycle or immediately if no bus cycle is being executed, then the $\overline{\text{HLDAK}}$ signal is set as active and the bus is released.

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

(v) $\overline{\text{REFRQ}}$ (Refresh Request) ... output

This is the refresh request signal for DRAM.

In cases where the address is decoded by an external circuit to increase the connected DRAM, or in cases where external SIMM's are connected, this signal is used in RAS control during the refresh cycle.

This signal becomes active during the refresh cycle. Also, during bus hold, it becomes active when a refresh request is generated and informs the external bus master that a refresh request was generated.

(8) PCT0, PCT1, PCT4, PCT5 (Port CT) ... 3-state I/O

PCT0, PCT1, PCT4, and PCT5 constitute a 4-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode, these pins operate as control signal outputs for when memory is expanded externally.

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

(a) Port mode

PCT0, PCT1, PCT4, and PCT5 can be set to input or output in 1-bit units by the port CT mode register (PMCT).

(b) Control mode

PCT0, PCT1, PCT4, and PCT5 can be set to port/control mode in 1-bit units by the PMCCT register.

(i) $\overline{\text{LWR}}$ (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, or external peripheral I/O area.

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

(ii) $\overline{\text{UWR}}$ (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, or external peripheral I/O area.

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

(iii) LDQM (Lower DQ Mask Enable) ... 3-state output

This is a control signal for the data bus to SDRAM. For the data bus, the lower byte is valid. This signal carries out SDRAM output disable control during a read operation, and SDRAM byte mask control during a write operation.

(iv) UDQM (Upper DQ Mask Enable) ... 3-state output

This is a control signal for the data bus to SDRAM. For the data bus, the higher byte is valid. This signal carries out SDRAM output disable control during a read operation, and SDRAM byte mask control during a write operation.

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

(9) PCS0, PCS3, PCS4, PCS7 (Port CS) ... 3-state I/O

PCS0, PCS3, PCS4, and PCS7 constitute a 4-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in the control mode, these pins operate as control signal outputs for when memory and peripheral I/O are expanded externally.

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

(a) Port mode

PCS0, PCS3, PCS4, and PCS7 can be set to input or output in 1-bit units by the port CS mode register (PMCS).

(b) Control mode

PCS0, PCS3, PCS4, and PCS7 can be set to port/control mode in 1-bit units by the PMCCS register.

(i) $\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$ (Chip Select) ... 3-state output

These are the chip select signals for the SRAM, external ROM, external peripheral I/O, and page ROM area.

The \overline{CSn} signal is assigned to memory block n (n = 0, 3, 4, 7).

It becomes active while the bus cycle that accesses the corresponding memory block is activated.

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

(10) PCD0 to PCD3 (Port CD) ... 3-state I/O

PCD0 to PCD3 constitute a 4-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in control mode, these pins operate as control signal outputs for when the memory and peripheral I/O are expanded externally.

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

(a) Port mode

PCD0 to PCD3 can be set to input or output in 1-bit units using the port CD mode register (PMCD).

(b) Control mode

PCD0 to PCD3 can be set to port or control mode in 1-bit units using the PMCCD register.

(i) SDCKE (SDRAM Clock Enable) ... 3-state output

This is the SDRAM clock enable output signal. It becomes inactive in self-refresh and standby mode.

(ii) SDCLK (SDRAM Clock Output) ... 3-state output

This is an SDRAM dedicated clock output signal. The same frequency as the internal system clock is output.

(iii) $\overline{\text{SDCAS}}$ (SDRAM Column Address Strobe) ... 3-state output

This is a command output signal for SDRAM.

(iv) $\overline{\text{SDRAS}}$ (SDRAM Row Address Strobe) ... 3-state output

This is a command output signal for SDRAM.

(v) $\overline{\text{LBE}}$ (Lower Byte Enable) ... 3-state output

This is the signal that enables the lower byte of the external data bus.

(vi) $\overline{\text{UBE}}$ (Upper Byte Enable) ... 3-state output

This is the signal that enables the higher byte of the external data bus.

(11) PAH0 to PAH8 (Port AH) ... 3-state I/O

PAH0 to PAH8 constitute an 8- or 9-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in control mode (external expansion mode), these pins operate as an address bus (A16 to A24) for when the memory is expanded externally.

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

(a) Port mode

PAH0 to PAH8 can be set to input or output in 1-bit units using the port AH mode register (PMAH).

(b) Control mode

PAH0 to PAH8 can function alternately as A16 to A24 by means of the PMCAH register.

(i) A16 to A24 (Address) ... 3-state output

These are the address output pins of the higher 9 bits of the address bus's 25-bit address when the external memory is accessed.

The output changes in synchronization with the fall of the CLKOUT signal in the T1 state. In the idle state (T1), the address of the bus cycle immediately before is retained.

★

(12) PAL0 to PAL15 (Port AL) ... 3-state I/O

PAL0 to PAL15 constitute an 8- or 16-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in control mode (external expansion mode), these pins operate as an address bus (A0 to A15) for when the memory is expanded externally.

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

(a) Port mode

PAL0 to PAL15 can be set to input or output in 1-bit units using the port AL mode register (PMAL).

(b) Control mode

PAL0 to PAL15 can function alternately as A0 to A15 by means of the PMCAL register.

(i) A0 to A15 (Address) ... 3-state output

These are the address output pins of the lower 16 bits of the address bus's 25-bit address when the external memory is accessed.

The output changes in synchronization with the fall of the CLKOUT signal in the T1 state. In the idle state (T1), the address of the bus cycle immediately before is retained.

(13) PDL0 to PDL15 (Port DL) ... 3-state I/O

PDL0 to PDL15 constitute an 8- or 16-bit I/O port that can be set to input or output in 1-bit units.

Besides functioning as a port, in control mode (external expansion mode), these pins operate as a data bus (D0 to D15) for when the memory is expanded externally.

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

(a) Port mode

PDL0 to PDL15 can be set to input or output in 1-bit units using the port DL mode register (PMDL).

(b) Control mode

PDL0 to PDL15 can function alternately as D0 to D15 by means of the PMCDL register.

(i) D0 to D15 (Data) ... 3-state I/O

These pins constitute a data bus for when the external memory is accessed. These are 16-bit data I/O bus pins.

The output changes in synchronization with the rise of the CLKOUT signal in the T1 state. In the idle state (T1), these pins become high impedance.

(14) CKSEL (Clock Generator Operating Mode Select) ... input

This is an input pin which specifies the clock generator's operating mode.

(15) MODE0 to MODE2 (Mode) ... input

These are input pins which specify the operating mode.

(16) $\overline{\text{RESET}}$ (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 standby mode (HALT, IDLE, or software STOP).

(17) X1, X2 (Crystal)

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

(18) CV_{DD} (Power Supply for Clock Generator)

This pin supplies positive power to the clock generator.

(19) CV_{SS} (Ground for Clock Generator)

This is the ground pin for the clock generator.

(20) V_{DD} (Power Supply)

These are the positive power supply pins for each internal unit. All the V_{DD} pins should be connected to a positive power source.

(21) V_{SS} (Ground)

These are ground pins. All the V_{SS} pins should be connected to ground.

(22) AV_{DD} (Analog Power Supply)

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

(23) AV_{SS} (Analog Ground)

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

(24) AV_{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

It is recommended that 1 to 10 kΩ resistors be used when connecting to V_{DD} or V_{SS} via resistors.

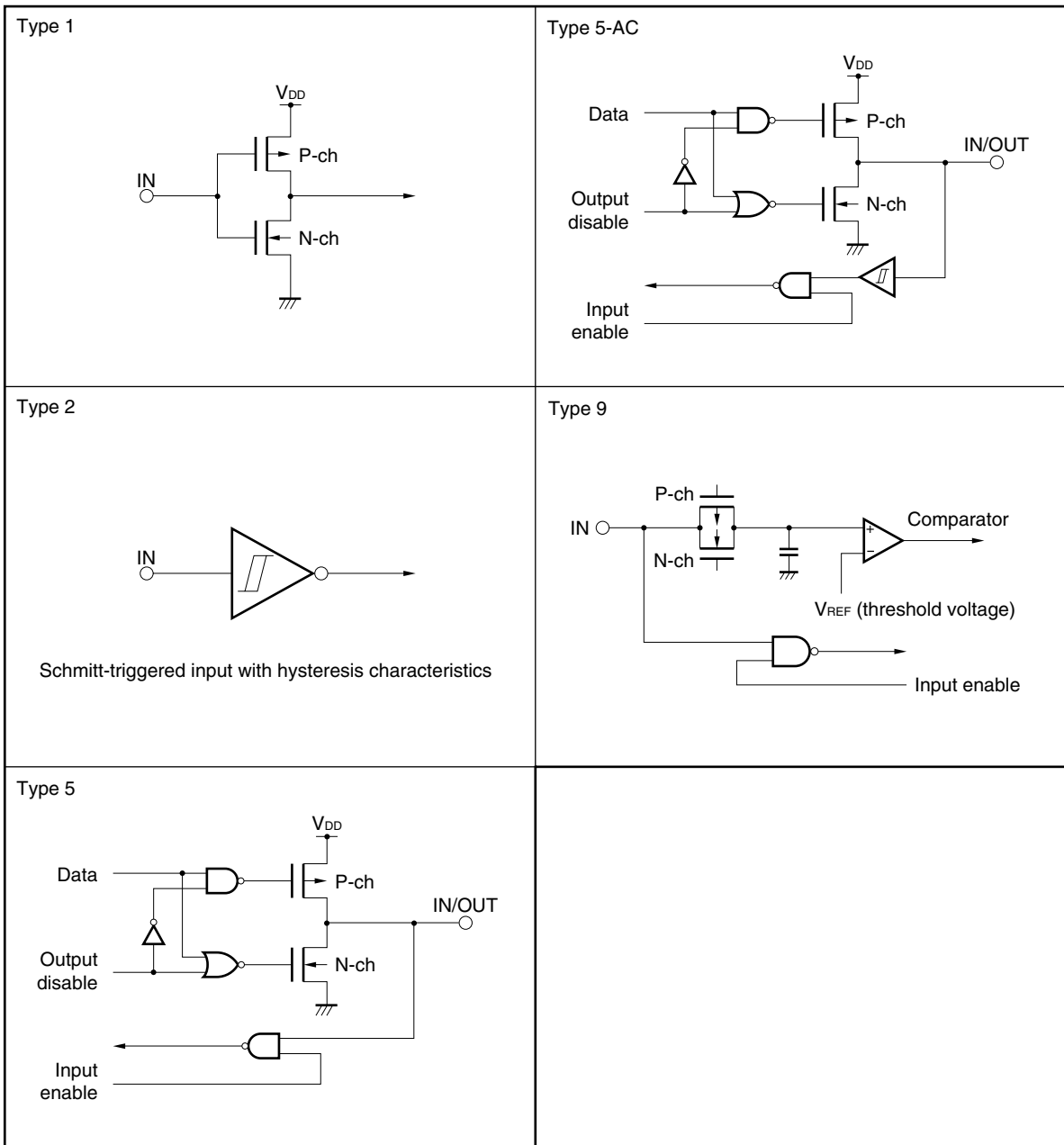
(1/2)

Pin Name	I/O Circuit Type	Recommended Connection
P01/INTP000/TI000	5-AC	Input: Independently connect to V _{DD} or V _{SS} via a resistor Output: Leave open
P02/INTP001		
P03/TO00	5	Input: Independently connect to V _{DD} or V _{SS} via a resistor Output: Leave open
P04/DMARQ0/INTP100, P05/DMARQ1/INTP101	5-AC	
P11/INTP010/TI010, P12/INTP011		
P20/NMI	2	
P24/TC0/INTP110	5-AC	Input: Independently connect to V _{DD} or V _{SS} via a resistor Output: Leave open
P40/TXD0/SO0	5	
P41/RXD0/SI0	5-AC	
P42/SCK0		
P43/TXD1/SO1	5	
P44/RXD1/SI1	5-AC	
P45/SCK1		
P70/ANI0 to P73/ANI3	9	
PBD0/DMAAK0, PBD1/DMAAK1	5	Input: Independently connect to V _{DD} or V _{SS} via a resistor Output: Leave open
PCM0/WAIT	5	Input: Independently connect to V _{DD} via a resistor
★ PCM1/CLKOUT	5	Input: Independently connect to V _{DD} or V _{SS} via a resistor Output: Leave open
PCM2/HLDAK		
PCM3/HLDRQ	5	Input: Independently connect to V _{DD} via a resistor
PCM4/REFRQ	5	Input: Independently connect to V _{DD} or V _{SS} via a resistor Output: Leave open
PCT0/LWR/LDQM		
PCT1/UWR/UDQM		
PCT4/RD		
PCT5/WE		
PCS0/CS0		
PCS3/CS3		
PCS4/CS4		
PCS7/CS7		

(2/2)

Pin Name	I/O Circuit Type	Recommended Connection
PCD0/SDCKE	5	Input: Independently connect to V_{DD} or V_{SS} via a resistor Output: Leave open
PCD1/SDCLK		
PCD2/ $\overline{LB\bar{E}}$ / \overline{SDCAS}		
PCD3/ $\overline{UB\bar{E}}$ / \overline{SDRAS}		
PAH0/A16 to PAH8/A24		
PAL0/A0 to PAL15/A15		
PDL0/D0 to PDL15/D15		
MODE0 to MODE2	2	–
\overline{RESET}		–
CKSEL	1	–
AV_{SS}	–	Connect to V_{SS} .
AV_{DD}/AV_{REF}	–	Connect to V_{DD} .

2.5 Pin I/O Circuits



CHAPTER 3 CPU FUNCTION

The CPU of the V850E/MA2 is based on RISC architecture and executes almost all the instructions in one clock cycle, using 5-stage pipeline control.

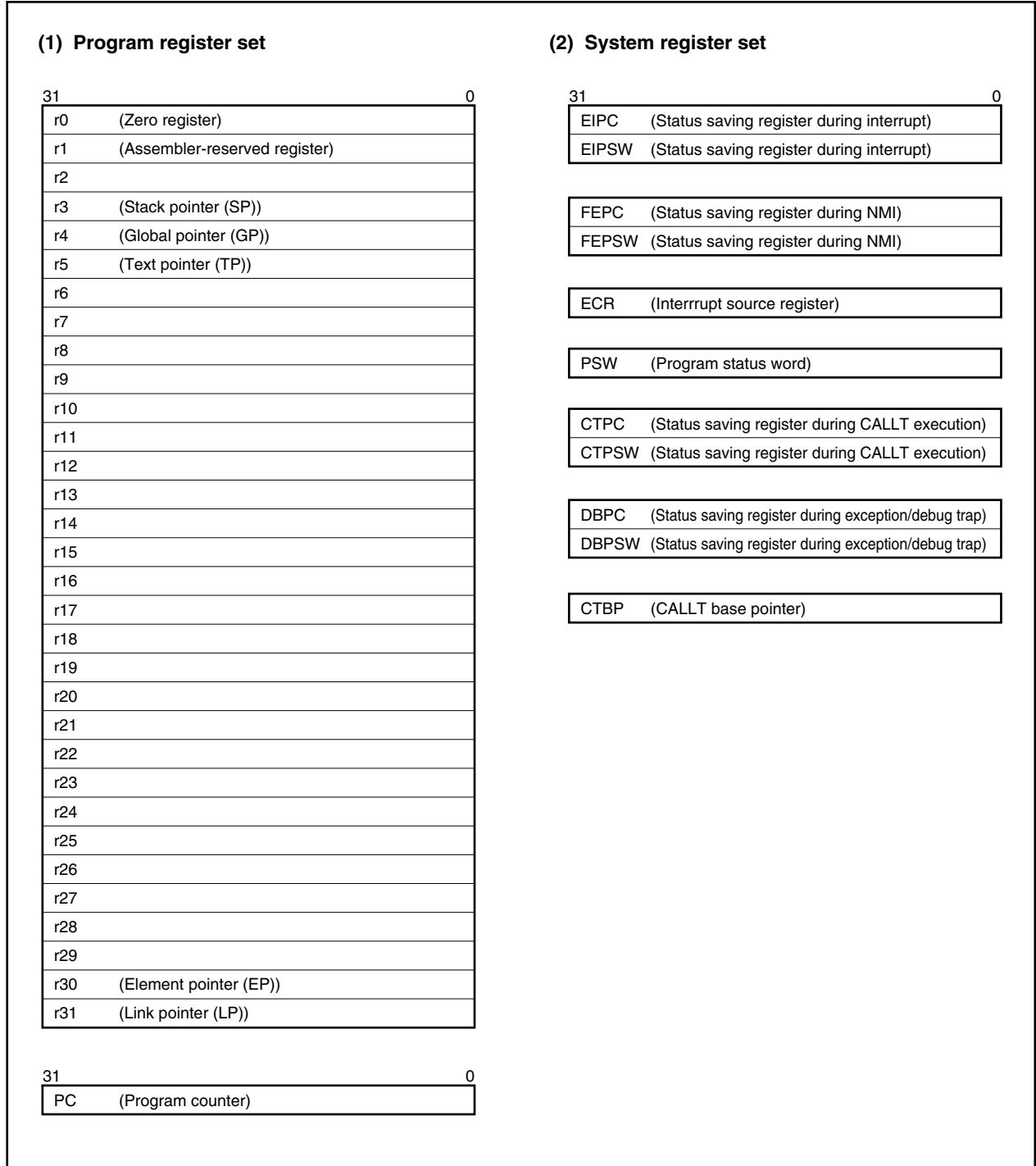
3.1 Features

- Minimum instruction cycle: 25 ns (at internal 40 MHz operation)
- Memory space Program space: 8 MB Linear
 Data space: 4 GB
- Thirty-two 32-bit general-purpose registers
- Internal 32-bit architecture
- Five-stage pipeline control
- Multiplication/division instructions
- Saturated operation instructions
- One-clock 32-bit shift instruction
- Load/store instructions with long/short format
- Four types of bit manipulation instructions
 - SET1
 - CLR1
 - NOT1
 - TST1

3.2 CPU Register Set

The registers of the V850E/MA2 can be classified into two categories: a general-purpose program register set and a dedicated system register set. All the registers are 32-bit width.

For details, refer to the **V850E1 Architecture User's Manual**.



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. r0 is a register that always holds 0, and is used for operations using 0 and offset 0 addressing. r30 is used, by means of the SLD and SST instructions, as a base pointer for when memory is accessed. Also, r1, r3 to r5, and r31 are implicitly used by the assembler and C compiler. Therefore, before using these registers, their contents must be saved so that they are not lost. The contents must be restored to the registers after the registers have been used. r2 may be used by the real-time OS. If the real-time OS does not use r2, it can be used as a variable register.

Table 3-1. Program Registers

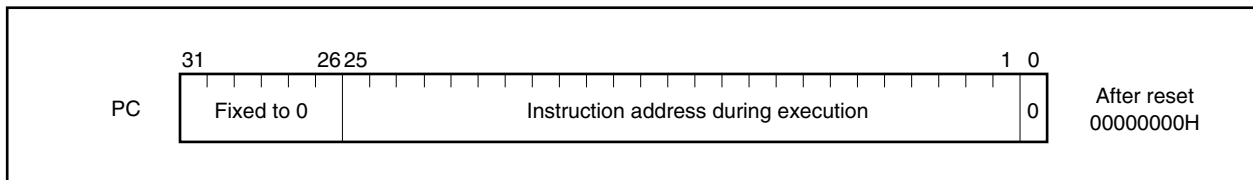
Name	Usage	Operation
r0	Zero register	Always holds 0
r1	Assembler-reserved register	Working register for generating 32-bit immediate data
r2	Address/data variable register (when r2 is not used by the real-time OS)	
r3	Stack pointer	Used to generate stack frame when function is called
r4	Global pointer	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

Remark For detailed descriptions of r1, r3 to r5, and r31, which are used by the assembler and C compiler, refer to the **CA850 (C Compiler Package) Assembly Language User's Manual**.

(2) Program counter (PC)

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.



3.2.2 System register set

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

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

Table 3-2. System Register Numbers

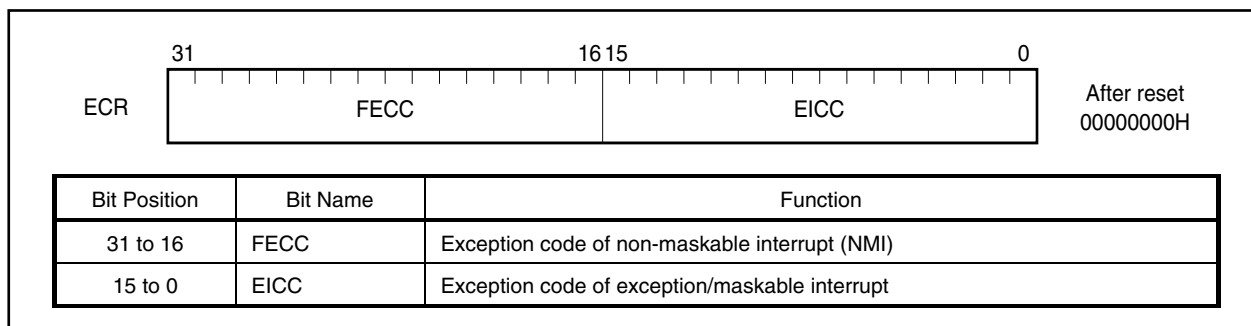
No.	System Register Name	Operand Specification	
		LDSR Instruction	STSR Instruction
0	Status saving register during interrupt (EIPC) ^{Note 1}	○	○
1	Status saving register during interrupt (EIPSW)	○	○
2	Status saving register during NMI (FEPC)	○	○
3	Status saving register during NMI (FEPSW)	○	○
4	Interrupt source register (ECR)	×	○
5	Program status word (PSW)	○	○
6 to 15	Reserved number for future function expansion (operations that access these register numbers cannot be guaranteed).	×	×
16	Status saving register during CALLT execution (CTPC)	○	○
17	Status saving register during CALLT execution (CTPSW)	○	○
18	Status saving register during exception/debug trap (DBPC)	○ ^{Note 2}	○
19	Status saving register during exception/debug trap (DBPSW)	○ ^{Note 2}	○
20	CALLT base pointer (CTBP)	○	○
21 to 31	Reserved number for future function expansion (operations that access these register numbers cannot be guaranteed).	×	×

- Notes 1.** Because this register has only one set, to enable multiple interrupts, it is necessary to save this register by program.
- 2.** Access is only possible while the DBTRAP instruction is executed.

Caution Even if bit 0 of EIPC, FEPC, or CTPC is set to 1 with the LDSR instruction, bit 0 will be ignored when the program returned by the RETI instruction after interrupt servicing (because bit 0 of the PC is fixed to 0). When setting the value of EIPC, FEPC, or CTPC, use the even value (bit 0 = 0).

Remark ○: Access allowed
 ×: Access prohibited

(1) Interrupt source register (ECR)



(2) Program status word (PSW)



Bit Position	Flag	Function
31 to 8	RFU	Reserved field (fixed to 0).
7	NP	Indicates that non-maskable interrupt (NMI) servicing is in progress. This flag is set when an NMI is acknowledged, and disables multiple interrupts. 0: NMI servicing not under execution. 1: NMI servicing under execution.
6	EP	Indicates that exception processing is in progress. This flag is set when an exception is generated. Moreover, interrupt requests can be acknowledged when this bit is set. 0: Exception processing not under execution. 1: Exception processing under execution.
5	ID	Displays whether a maskable interrupt request can be acknowledged or not. 0: Interrupt enabled. 1: Interrupt disabled.
4	SAT ^{Note}	Displays that the operation result of a saturated operation processing instruction is saturated due to overflow. Because this is a the cumulative flag, if the operation result is saturated by the saturation operation instruction, this bit is set (1), but is not cleared (0) even if the operation results of subsequent instructions are not saturated. To clear (0) this bit, load the data in PSW. Note that in a general arithmetic operation, this bit is neither set (1) nor cleared (0). 0: Not saturated. 1: Saturated.
3	CY	This flag is set if carry or borrow occurs as result of operation (if carry or borrow does not occur, it is reset). 0: Carry or borrow does not occur. 1: Carry or borrow occurs.
2	OV ^{Note}	This flag is set if overflow occurs during operation (if overflow does not occur, it is reset). 0: Overflow does not occur. 1: Overflow occurs.
1	S ^{Note}	This flag is set if the result of operation is negative (it is reset if the result is positive). 0: The operation result was positive or 0. 1: The operation result was negative.
0	Z	This flag is set if the result of operation is zero (if the result is not zero, it is reset). 0: The operation result was not 0. 1: The operation result was 0.

Note The result of a saturation-processed operation is determined by the contents of the OV and S flags in the saturation operation. Simply setting the OV flag (1) will set the SAT flag (1) in a saturation operation.

Status of Operation Result	Flag Status			Saturation-Processed Operation Result
	SAT	OV	S	
Maximum positive value exceeded	1	1	0	7FFFFFFFH
Maximum negative value exceeded	1	1	1	80000000H
Positive (not exceeding the maximum)	Retain the value before operation	0	0	Operation result itself
Negative (not exceeding the maximum)			1	

3.3 Operation Modes

3.3.1 Operation modes

The V850E/MA2 has the following operation modes. Mode specification is carried out by the MODE0 to MODE2 pins.

(1) ROMless modes 0, 1

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

In ROMless mode 0, the data bus is a 16-bit data bus and in ROMless 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 the MODE0 to MODE2 pins. 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	MODE1	MODE0	Operation Mode		Remarks
L	L	L	Normal operation mode	ROMless mode 0	16-bit data bus
L	L	H		ROMless mode 1	8-bit data bus
Other than above			Setting prohibited		

Remarks L: Low-level input
H: High-level input

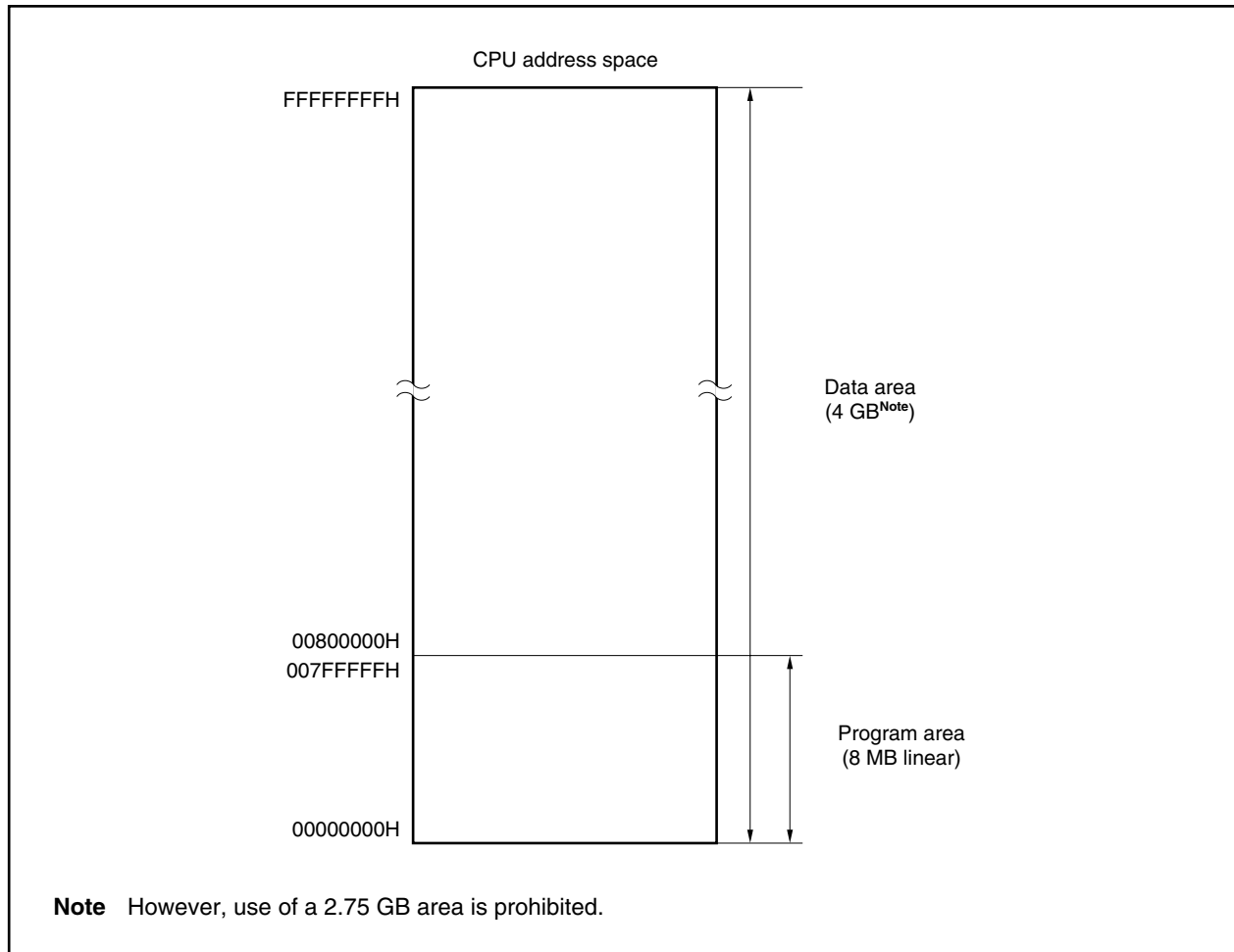
3.4 Address Space

3.4.1 CPU address space

The CPU of the V850E/MA2 is of 32-bit architecture and supports up to 4 GB of address space (data space) during operand addressing (data access). Also, in instruction address addressing, a maximum of 8 MB of linear address space (program space) is supported.

The following shows the CPU address space.

Figure 3-1. CPU Address Space



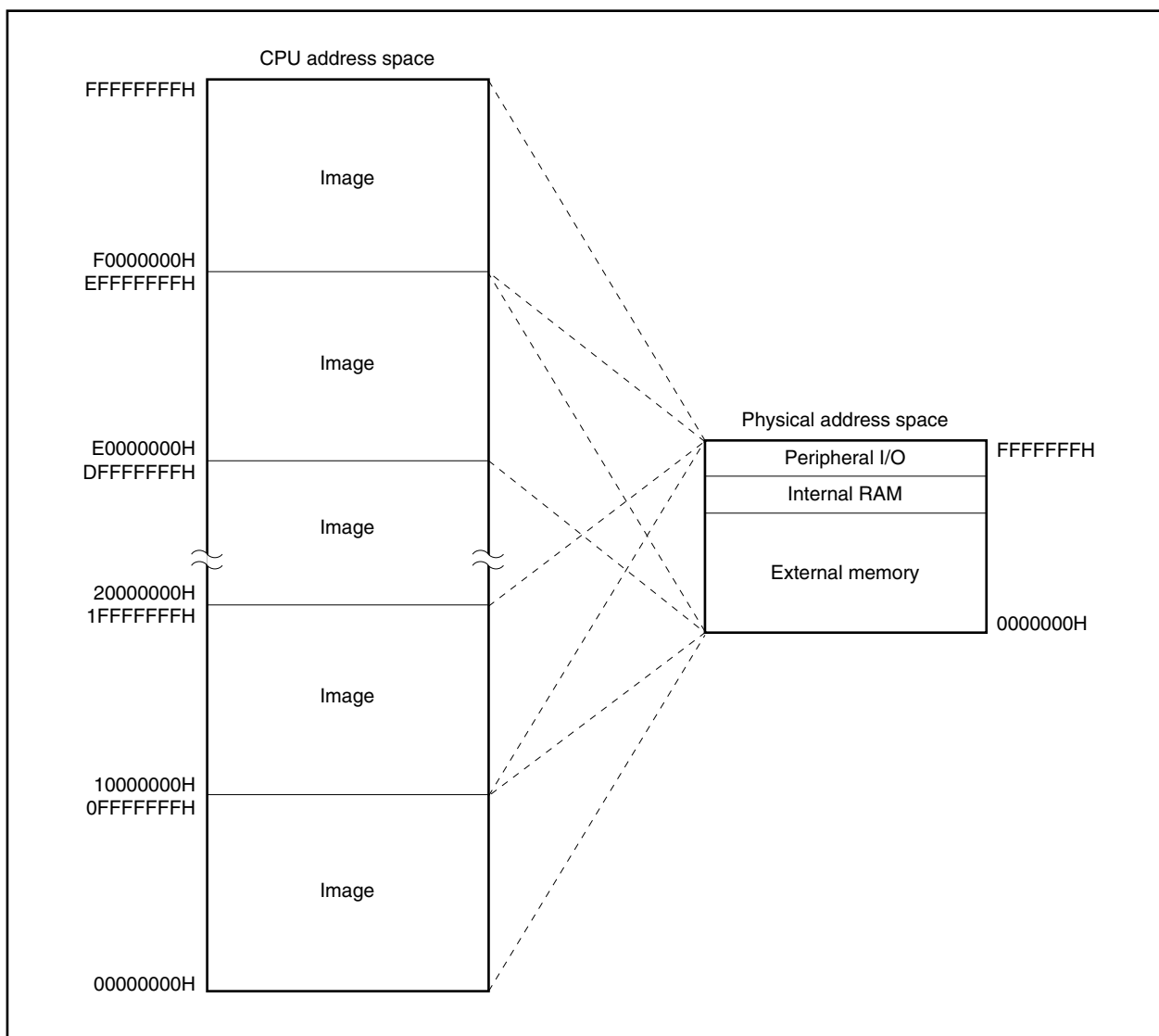
3.4.2 Image

A 256 MB physical address space is seen as 16 images in the 4 GB CPU address space. In actuality, the same 256 MB physical address space is accessed regardless of the values of bits 31 to 28 of the CPU address. Figure 3-2 shows the image of the virtual addressing space.

Physical address x0000000H can be seen as CPU address 00000000H, and in addition, can be seen as address 10000000H, address 20000000H, ... , address E0000000H, or address F0000000H.

Caution Of the 256 MB physical address space of the V850E/MA2, only an 80 MB area is usable (see 4.3 Memory Block Function). Therefore, the space that can be addressed in the operand addressing mode is not 4 GB but 1.25 GB.

Figure 3-2. Images on Address Space



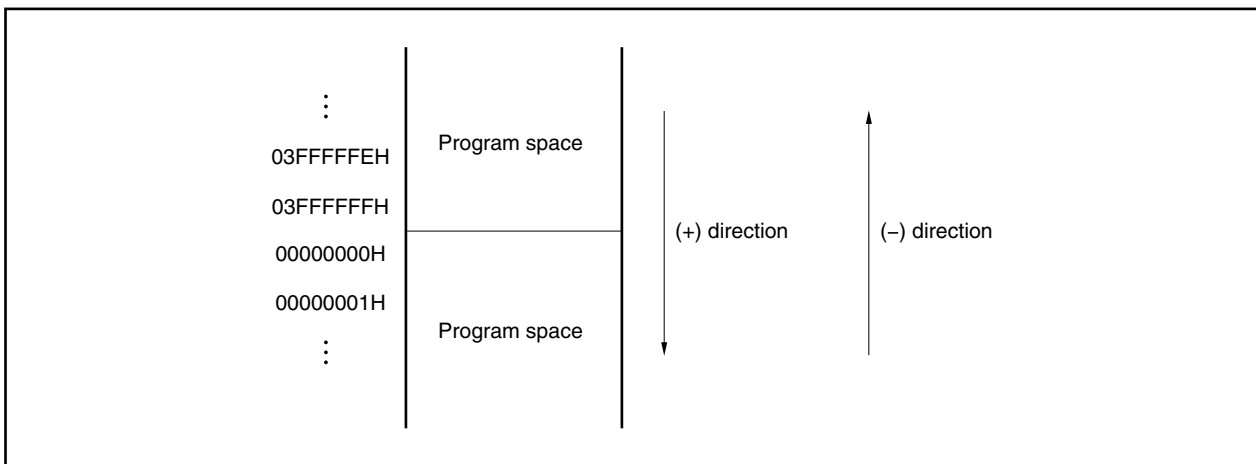
3.4.3 Wrap-around of CPU address space

(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 a situation like this whereby the lower-limit address and upper-limit address become contiguous.

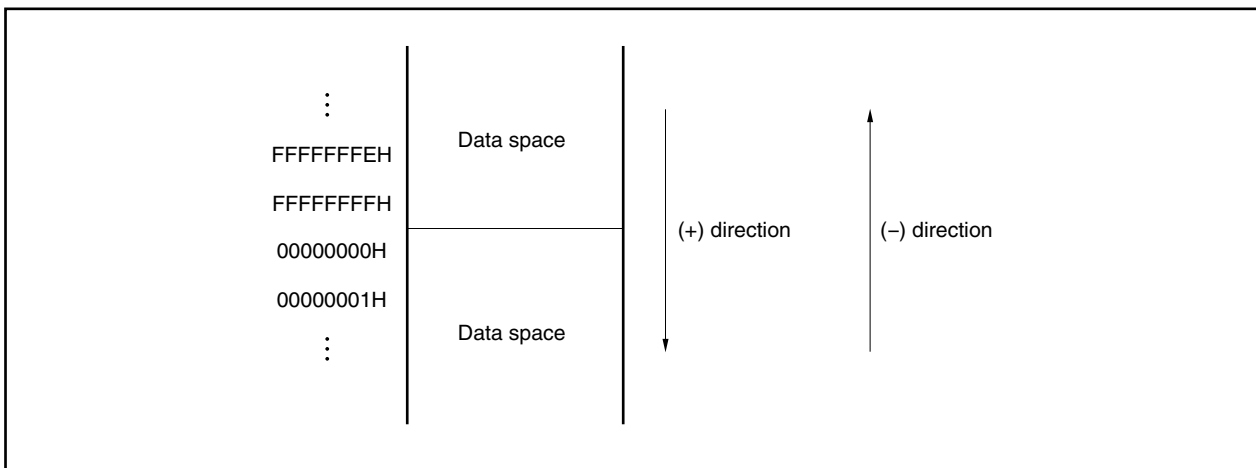
★ **Caution** The 4 KB area of 03FFF000H to 03FFFFFFH can be seen as an image of 0FFFF000H to 0FFFFFFFH. This area is access-prohibited. Therefore, do not execute any branch address calculation in which the result will reside in any part of this area.



(2) Data space

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

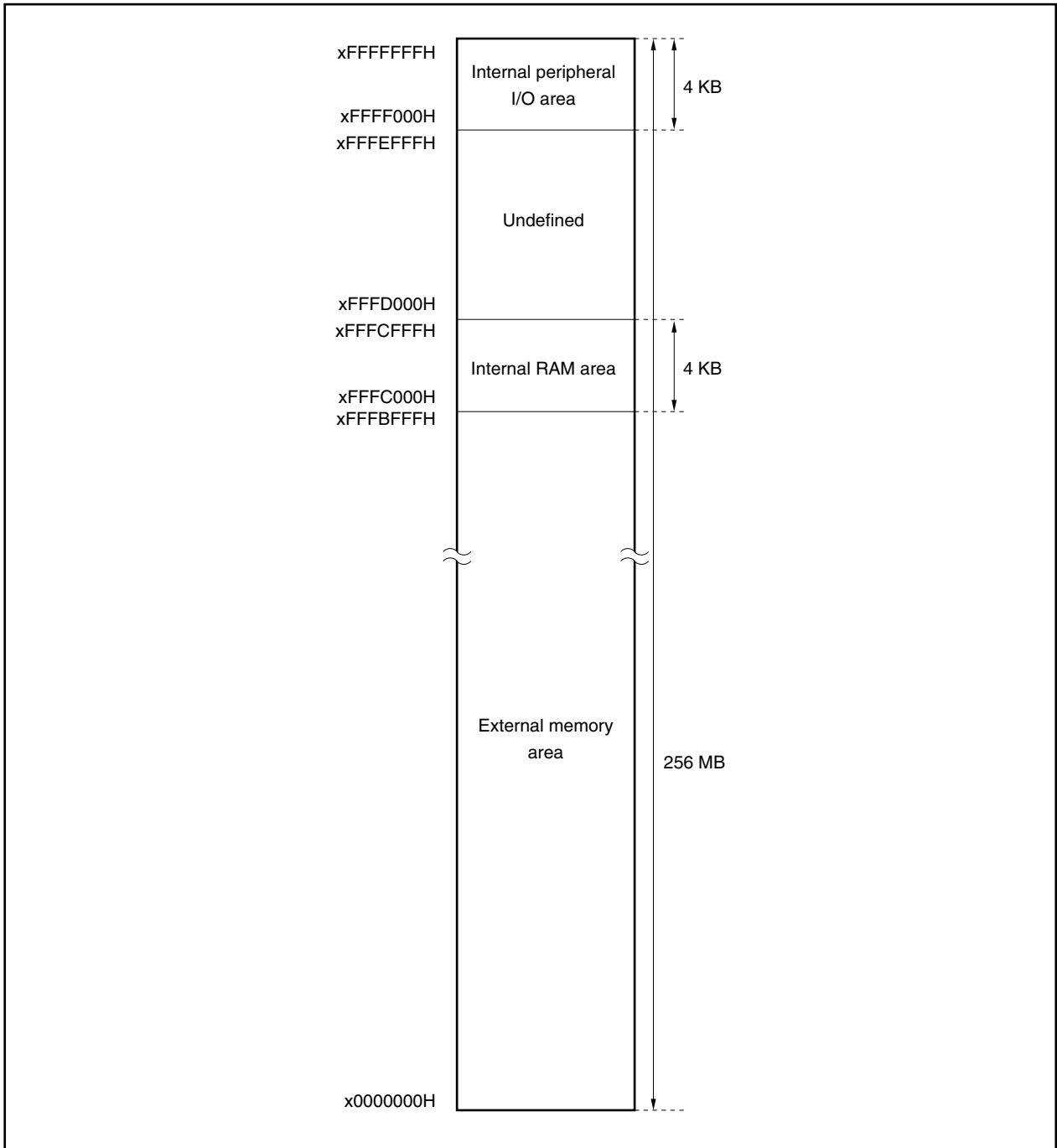
Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address FFFFFFFFH are contiguous addresses, and the data space is wrapped around at the boundary of these addresses.



3.4.4 Memory map

The V850E/MA2 reserves areas as shown below.

Figure 3-3. Memory Map



3.4.5 Area

(1) Memory area

An 80 MB external memory area is available for memory area. The lower 8 MB can be used as program/data area and the higher 72 MB as data area. The addresses of the external memory area are as follows.

x0000000H to x07FFFFFFH, x4000000H to x5FFFFFFH,
x8000000H to x9FFFFFFH, xF800000H to xFFFBFFFH

Access to the memory area uses the chip select signal assigned to each memory block (which is carried out in the CS unit set by chip area selection control registers 0 and 1 (CSC0, CSC1)).

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

(a) Interrupt/exception table

The V850E/MA2 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. When an interrupt/exception request is acknowledged, 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.

Remark In order to restore correct operation after reset, provide a handler address to the reset routine in address 0 of the external memory.

Table 3-3. Interrupt/Exception Table

Start Address of Interrupt/Exception Table	Interrupt/Exception Source
00000000H	RESET
00000010H	NMI
00000040H	TRAP0n (n = 0 to F)
00000050H	TRAP1n (n = 0 to F)
00000060H	ILGOP/DBG0
00000080H	INTOV00
00000090H	INTOV01
000000C0H	INTP000/INTM000
000000D0H	INTP001/INTM001
000000E0H	INTP010/INTM010
000000F0H	INTP011/INTM011
00000140H	INTP100
00000150H	INTP101
00000180H	INTP110
00000240H	INTCMD0
00000250H	INTCMD1
00000260H	INTCMD2
00000270H	INTCMD3
00000280H	INTDMA0
00000290H	INTDMA1
000002A0H	INTDMA2
000002B0H	INTDMA3
000002C0H	INTCSI0
000002D0H	INTSER0
000002E0H	INTSR0
000002F0H	INTST0
00000300H	INTCSI1
00000310H	INTSER1
00000320H	INTSR1
00000330H	INTST1
00000380H	INTAD

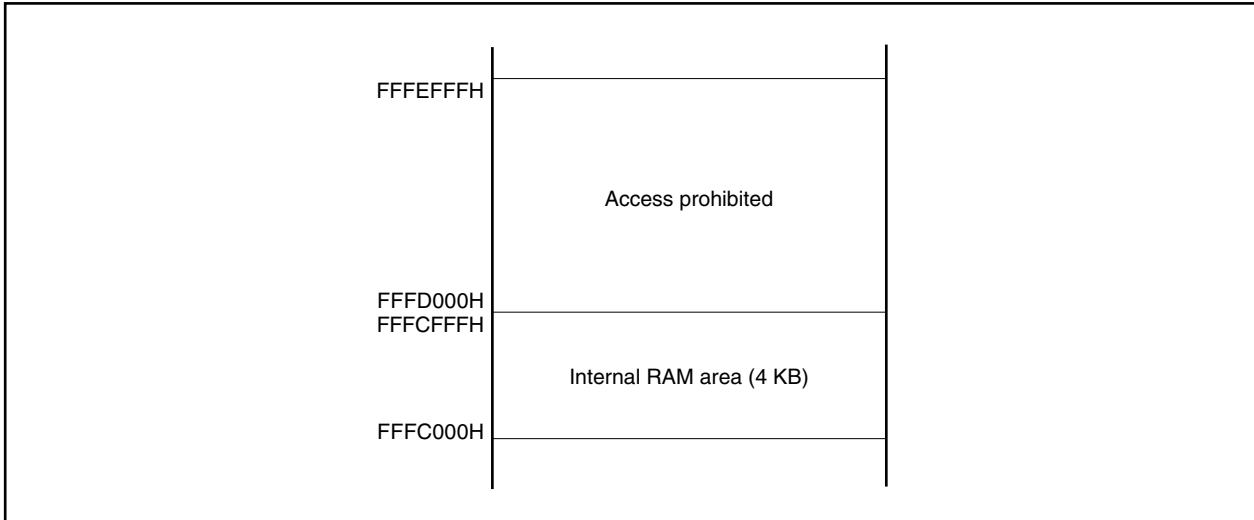
★ (2) Internal RAM area

The 12 KB of addresses FFFC000H to FFFEFFFFH are reserved for the internal RAM area.

The 12 KB area of 3FFC000H to 3FFEFFFFH can be seen as an image of FFFC000H to FFFEFFFFH.

The 4 KB of addresses FFFC000H to FFFCFFFH are provided as physical internal RAM.

Caution Addresses FFFD000H to FFFEFFFFH are access-prohibited.

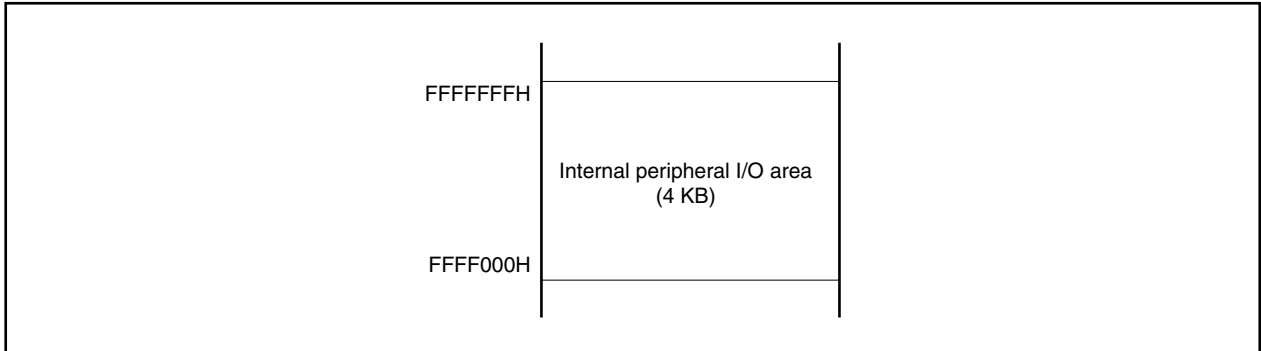


(3) Internal peripheral I/O area

4 KB of memory, addresses FFFF000H to FFFFFFFFH, are provided as an internal peripheral I/O area. 3FFF000H to 3FFFFFFFH^{Note} can be seen as an image of FFFF000H to FFFFFFFFH.

★

Note Addresses 3FFF000H to 3FFFFFFFH are access-prohibited. To access the internal peripheral I/O, specify addresses FFFF000H to FFFFFFFFH.



Peripheral I/O registers associated with the operation mode specification and the state monitoring for the internal peripherals I/O are all memory-mapped to the internal peripheral I/O area. Program fetches cannot be executed from this area.

- Cautions**
1. In the V850E/MA2, no registers exist which are capable of word access, but if a register is word accessed, halfword access is performed twice in the order of lower address, then higher address of the word area, disregarding the lower 2 bits of the address.
 2. For registers in which byte access is possible, if halfword access is executed, the higher 8 bits become undefined during the read operation, and the lower 8 bits of data are written to the register during the write operation.
 3. Addresses that are not defined as registers are reserved for future expansion. If these addresses are accessed, the operation is undefined and not guaranteed. Addresses 3FFF000H to 3FFFFFFFH cannot be specified as the source/destination address of DMA transfer. Be sure to use addresses FFFF000H to FFFFFFFFH for the source/destination address of DMA transfer.

3.4.6 External memory expansion

By setting the port n mode control register (PMCn) to control mode, an external memory device can be connected to the external memory space using each pin of ports AL, AH, DL, CS, CT, CM, and CD. Each register is set by selecting control mode for each pin of these ports using PMCn (n = AL, AH, DL, CS, CT, CM, CD).

Note that the status after reset differs as shown below in accordance with the operating mode specification set by the MODE0 to MODE2 pins (refer to **3.3 Operation Modes** concerning the operation modes).

(a) In the case of ROMless mode 0

Because each pin of ports AL, AH, DL, CS, CT, CM, and CD enters control mode following a reset, external memory can be used without making changes to the port n mode control register (PMCn) (the external data bus width is 16 bits).

(b) In the case of ROMless mode 1

Because each pin of ports AL, AH, DL, CS, CT, CM, and CD enters control mode following a reset, external memory can be used without making changes to the port n mode control register (PMCn) (the external data bus width is 8 bits).

Remark n = AL, AH, DL, CS, CT, CM, CD

3.4.7 Recommended use of address space

The architecture of the V850E/MA2 requires that a register that serves as a pointer be secured for address generation when performing operand data access in the data space. Operand data access from an instruction can be directly executed at the address in this pointer register ± 32 KB. However, because there is a limit to which general-purpose registers are used as a pointer register, 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.

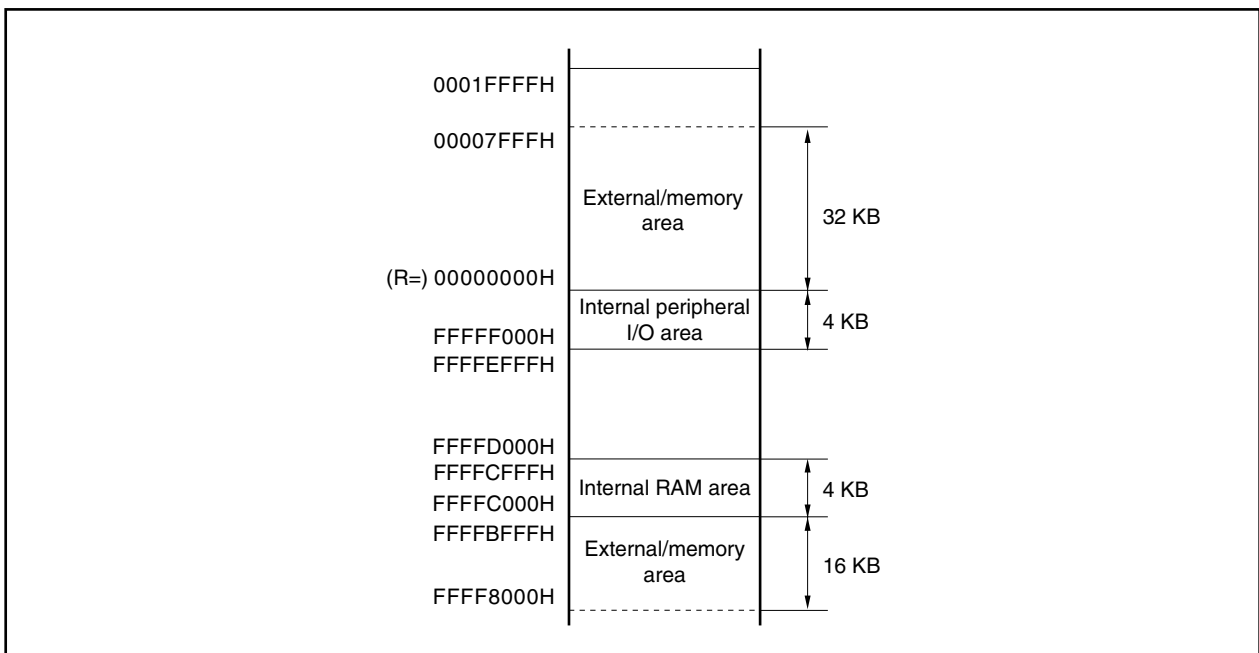
(1) Program space

Of the 32 bits of the program counter (PC), the higher 6 bits are fixed to 0, and only the lower 26 bits are valid. Of those valid bits, a contiguous 8 MB space, starting from address 00000000H, corresponds to the memory map of the program space.

(2) Data space

With the V850E/MA2, a 256 MB physical address space is seen as 16 images in the 4 GB CPU address space. The highest bit (bit 25) of this 26-bit address is assigned as an address sign-extended to 32 bits. Of the 256 MB physical address space, only an 80 MB area is usable. Therefore, the space that can be addressed in the operand addressing mode is not 4 GB but 1.25 GB.

Example Application of wrap-around



When R = r0 (zero register) is specified with the LD/ST disp16 [R] instruction, an addressing range of 00000000H ± 32 KB can be referenced with the sign-extended disp 16. By mapping the external memory area in the 16 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 the hardware, and eliminates the need for additional registers for the pointer.

★ 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	
FFFFF00H	Port AL	PAL	R/W			○	Undefined
FFFFF00H	Port ALL	PALL	R/W	○	○		Undefined
FFFFF01H	Port ALH	PALH	R/W	○	○		Undefined
FFFFF02H	Port AH	PAH	R/W			○	Undefined
FFFFF02H	Port AHL	PAHL	R/W	○	○		Undefined
FFFFF03H	Port AHH	PAHH	R/W	○	○		Undefined
FFFFF04H	Port DL	PDL	R/W			○	Undefined
FFFFF04H	Port DLL	PDLL	R/W	○	○		Undefined
FFFFF05H	Port DLH	PDLH	R/W	○	○		Undefined
FFFFF08H	Port CS	PCS	R/W	○	○		Undefined
FFFFF0AH	Port CT	PCT	R/W	○	○		Undefined
FFFFF0CH	Port CM	PCM	R/W	○	○		Undefined
FFFFF0EH	Port CD	PCD	R/W	○	○		Undefined
FFFFF012H	Port BD	PBD	R/W	○	○		Undefined
FFFFF020H	Port AL mode register	PMAL	R/W			○	FFFFH
FFFFF020H	Port AL mode register L	PMALL	R/W	○	○		FFH
FFFFF021H	Port AL mode register H	PMALH	R/W	○	○		FFH
FFFFF022H	Port AH mode register	MAH	R/W			○	FFFFH
FFFFF022H	Port AH mode register L	MAHL	R/W	○	○		FFH
FFFFF023H	Port AH mode register H	MAHH	R/W	○	○		FFH
FFFFF024H	Port DL mode register	PMDL	R/W			○	FFFFH
FFFFF024H	Port DL mode register L	PMDLL	R/W	○	○		FFH
FFFFF025H	Port DL mode register H	PMDLH	R/W	○	○		FFH
FFFFF028H	Port CS mode register	PMCS	R/W	○	○		FFH
FFFFF02AH	Port CT mode register	PMCT	R/W	○	○		FFH
FFFFF02CH	Port CM mode register	PMCM	R/W	○	○		FFH
FFFFF02EH	Port CD mode register	PMCD	R/W	○	○		FFH
FFFFF032H	Port BD mode register	PMBD	R/W	○	○		FFH
FFFFF040H	Port AL mode control register	PMCAL	R/W			○	FFFFH
FFFFF040H	Port AL mode control register L	PMCALL	R/W	○	○		FFH
FFFFF041H	Port AL mode control register H	PMCALH	R/W	○	○		FFH
FFFFF042H	Port AH mode control register	PMCAH	R/W			○	01FFH
FFFFF042H	Port AH mode control register L	PMCAHL	R/W	○	○		FFH
FFFFF043H	Port AH mode control register H	PMCAHH	R/W	○	○		01H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF044H	Port DL mode control register	PMCDL	R/W			○	FFFFH
FFFFF044H	Port DL mode control register L	PMCDLL	R/W	○	○		FFH
FFFFF045H	Port DL mode control register H	PMCDLH	R/W	○	○		FFH
FFFFF048H	Port CS mode control register	PMCCS	R/W	○	○		99H
FFFFF04AH	Port CT mode control register	PMCCT	R/W	○	○		33H
FFFFF04CH	Port CM mode control register	PMCCM	R/W	○	○		1FH
FFFFF04EH	Port CD mode control register	PMCCD	R/W	○	○		0FH
FFFFF04FH	Port CD function control register	PFCCD	R/W	○	○		00H
FFFFF052H	Port BD mode control register	PMCBD	R/W	○	○		00H
FFFFF060H	Chip area select control register 0	CSC0	R/W			○	2C11H
FFFFF062H	Chip area select control register 1	CSC1	R/W			○	2C11H
FFFFF066H	Bus size configuration register	BSC	R/W			○	0000H/5555H
FFFFF068H	Endian configuration register	BEC	R/W			○	0000H
FFFFF06EH	System wait control register	VSWC	R/W		○		77H
FFFFF080H	DMA source address register 0L	DSA0L	R/W			○	Undefined
FFFFF082H	DMA source address register 0H	DSA0H	R/W			○	Undefined
FFFFF084H	DMA destination address register 0L	DDA0L	R/W			○	Undefined
FFFFF086H	DMA destination address register 0H	DDA0H	R/W			○	Undefined
FFFFF088H	DMA source address register 1L	DSA1L	R/W			○	Undefined
FFFFF08AH	DMA source address register 1H	DSA1H	R/W			○	Undefined
FFFFF08CH	DMA destination address register 1L	DDA1L	R/W			○	Undefined
FFFFF08EH	DMA destination address register 1H	DDA1H	R/W			○	Undefined
FFFFF090H	DMA source address register 2L	DSA2L	R/W			○	Undefined
FFFFF092H	DMA source address register 2H	DSA2H	R/W			○	Undefined
FFFFF094H	DMA destination address register 2L	DDA2L	R/W			○	Undefined
FFFFF096H	DMA destination address register 2H	DDA2H	R/W			○	Undefined
FFFFF098H	DMA source address register 3L	DSA3L	R/W			○	Undefined
FFFFF09AH	DMA source address register 3H	DSA3H	R/W			○	Undefined
FFFFF09CH	DMA destination address register 3L	DDA3L	R/W			○	Undefined
FFFFF09EH	DMA destination address register 3H	DDA3H	R/W			○	Undefined
FFFFF0C0H	DMA byte count register 0	DBC0	R/W			○	Undefined
FFFFF0C2H	DMA byte count register 1	DBC1	R/W			○	Undefined
FFFFF0C4H	DMA byte count register 2	DBC2	R/W			○	Undefined
FFFFF0C6H	DMA byte count register 3	DBC3	R/W			○	Undefined
FFFFF0D0H	DMA addressing control register 0	DADC0	R/W			○	0000H
FFFFF0D2H	DMA addressing control register 1	DADC1	R/W			○	0000H
FFFFF0D4H	DMA addressing control register 2	DADC2	R/W			○	0000H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFFF0D6H	DMA addressing control register 3	DADC3	R/W			○	0000H
FFFFFF0E0H	DMA channel control register 0	DCHC0	R/W	○	○		00H
FFFFFF0E2H	DMA channel control register 1	DCHC1	R/W	○	○		00H
FFFFFF0E4H	DMA channel control register 2	DCHC2	R/W	○	○		00H
FFFFFF0E6H	DMA channel control register 3	DCHC3	R/W	○	○		00H
FFFFFF0F0H	DMA disable status register	DDIS	R		○		00H
FFFFFF0F2H	DMA restart register	DRST	R/W		○		00H
FFFFFF100H	Interrupt mask register 0	IMR0	R/W			○	FFFFH
FFFFFF100H	Interrupt mask register 0L	IMR0L	R/W	○	○		FFH
FFFFFF101H	Interrupt mask register 0H	IMR0H	R/W	○	○		FFH
FFFFFF102H	Interrupt mask register 1	IMR1	R/W			○	FFFFH
FFFFFF102H	Interrupt mask register 1L	IMR1L	R/W	○	○		FFH
FFFFFF103H	Interrupt mask register 1H	IMR1H	R/W	○	○		FFH
FFFFFF104H	Interrupt mask register 2	IMR2	R/W			○	FFFFH
FFFFFF104H	Interrupt mask register 2L	IMR2L	R/W	○	○		FFH
FFFFFF105H	Interrupt mask register 2H	IMR2H	R/W	○	○		FFH
FFFFFF106H	Interrupt mask register 3	IMR3	R/W			○	FFFFH
FFFFFF106H	Interrupt mask register 3L	IMR3L	R/W	○	○		FFH
FFFFFF107H	Interrupt mask register 3H	IMR3H	R/W	○	○		FFH
FFFFFF110H	Interrupt control register	OVIC00	R/W	○	○		47H
FFFFFF112H	Interrupt control register	OVIC01	R/W	○	○		47H
FFFFFF118H	Interrupt control register	P00IC0	R/W	○	○		47H
FFFFFF11AH	Interrupt control register	P00IC1	R/W	○	○		47H
FFFFFF11CH	Interrupt control register	P01IC0	R/W	○	○		47H
FFFFFF11EH	Interrupt control register	P01IC1	R/W	○	○		47H
FFFFFF128H	Interrupt control register	P10IC0	R/W	○	○		47H
FFFFFF12AH	Interrupt control register	P10IC1	R/W	○	○		47H
FFFFFF130H	Interrupt control register	P11IC0	R/W	○	○		47H
FFFFFF148H	Interrupt control register	CMICD0	R/W	○	○		47H
FFFFFF14AH	Interrupt control register	CMICD1	R/W	○	○		47H
FFFFFF14CH	Interrupt control register	CMICD2	R/W	○	○		47H
FFFFFF14EH	Interrupt control register	CMICD3	R/W	○	○		47H
FFFFFF150H	Interrupt control register	DMAIC0	R/W	○	○		47H
FFFFFF152H	Interrupt control register	DMAIC1	R/W	○	○		47H
FFFFFF154H	Interrupt control register	DMAIC2	R/W	○	○		47H
FFFFFF156H	Interrupt control register	DMAIC3	R/W	○	○		47H
FFFFFF158H	Interrupt control register	CSIIC0	R/W	○	○		47H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF15AH	Interrupt control register	SEIC0	R/W	○	○		47H
FFFFF15CH	Interrupt control register	SRIC0	R/W	○	○		47H
FFFFF15EH	Interrupt control register	STIC0	R/W	○	○		47H
FFFFF160H	Interrupt control register	CSIIC1	R/W	○	○		47H
FFFFF162H	Interrupt control register	SEIC1	R/W	○	○		47H
FFFFF164H	Interrupt control register	SRIC1	R/W	○	○		47H
FFFFF166H	Interrupt control register	STIC1	R/W	○	○		47H
FFFFF170H	Interrupt control register	ADIC	R/W	○	○		47H
FFFFF1FAH	In-service priority register	ISPR	R	○	○		00H
FFFFF1FCH	Command register	PRCMD	W		○		Undefined
FFFFF1FEH	Power save control register	PSC	R/W	○	○		00H
FFFFF200H	A/D converter mode register 0	ADM0	R/W	○	○		00H
FFFFF201H	A/D converter mode register 1	ADM1	R/W		○		07H
FFFFF202H	A/D converter mode register 2	ADM2	R/W	○	○		00H
FFFFF210H	A/D conversion result register 0 (10 bits)	ADCR0	R			○	0000H
FFFFF212H	A/D conversion result register 1 (10 bits)	ADCR1	R			○	0000H
FFFFF214H	A/D conversion result register 2 (10 bits)	ADCR2	R			○	0000H
FFFFF216H	A/D conversion result register 3 (10 bits)	ADCR3	R			○	0000H
FFFFF220H	A/D conversion result register 0H (8 bits)	ADCR0H	R		○		00H
FFFFF221H	A/D conversion result register 1H (8 bits)	ADCR1H	R		○		00H
FFFFF222H	A/D conversion result register 2H (8 bits)	ADCR2H	R		○		00H
FFFFF223H	A/D conversion result register 3H (8 bits)	ADCR3H	R		○		00H
FFFFF400H	Port 0	P0	R/W	○	○		Undefined
FFFFF402H	Port 1	P1	R/W	○	○		Undefined
FFFFF404H	Port 2	P2	R/W	○	○		Undefined
FFFFF408H	Port 4	P4	R/W	○	○		Undefined
FFFFF40EH	Port 7	P7	R/W	○	○		Undefined
FFFFF420H	Port 0 mode register	PM0	R/W	○	○		FFH
FFFFF422H	Port 1 mode register	PM1	R/W	○	○		FFH
FFFFF424H	Port 2 mode register	PM2	R/W	○	○		FFH
FFFFF428H	Port 4 mode register	PM4	R/W	○	○		FFH
FFFFF440H	Port 0 mode control register	PMC0	R/W	○	○		00H
FFFFF442H	Port 1 mode control register	PMC1	R/W	○	○		00H
FFFFF444H	Port 2 mode control register	PMC2	R/W	○	○		01H
FFFFF448H	Port 4 mode control register	PMC4	R/W	○	○		00H
FFFFF460H	Port 0 function control register	PFC0	R/W	○	○		00H
FFFFF464H	Port 2 function control register	PFC2	R/W	○	○		00H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF468H	Port 4 function control register	PFC4	R/W	○	○		00H
FFFFF480H	Bus cycle type configuration register 0	BCT0	R/W			○	8888H
FFFFF482H	Bus cycle type configuration register 1	BCT1	R/W			○	8888H
FFFFF484H	Data wait control register 0	DWC0	R/W			○	7777H
FFFFF486H	Data wait control register 1	DWC1	R/W			○	7777H
FFFFF488H	Bus cycle control register	BCC	R/W			○	FFFFH
FFFFF48AH	Address setup wait control register	ASC	R/W			○	FFFFH
FFFFF49AH	Page-ROM configuration register	PRC	R/W			○	7000H
FFFFF4ACH	SDRAM configuration register 3	SCR3	R/W			○	0000H
FFFFF4AEH	SDRAM refresh control register 3	RFS3	R/W			○	0000H
FFFFF4B0H	SDRAM configuration register 4	SCR4	R/W			○	0000H
FFFFF4B2H	SDRAM refresh control register 4	RFS4	R/W			○	0000H
FFFFF540H	Timer D0	TMD0	R			○	0000H
FFFFF542H	Compare register D0	CMD0	R/W			○	0000H
FFFFF544H	Timer mode control register D0	TMCD0	R/W	○	○		00H
FFFFF550H	Timer D1	TMD1	R			○	0000H
FFFFF552H	Compare register D1	CMD1	R/W			○	0000H
FFFFF554H	Timer mode control register D1	TMCD1	R/W	○	○		00H
FFFFF560H	Timer D2	TMD2	R			○	0000H
FFFFF562H	Compare register D2	CMD2	R/W			○	0000H
FFFFF564H	Timer mode control register D2	TMCD2	R/W	○	○		00H
FFFFF570H	Timer D3	TMD3	R			○	0000H
FFFFF572H	Compare register D3	CMD3	R/W			○	0000H
FFFFF574H	Timer mode control register D3	TMCD3	R/W	○	○		00H
FFFFF600H	Timer C0	TMC0	R			○	0000H
FFFFF602H	Capture/compare register C00	CCC00	R/W			○	0000H
FFFFF604H	Capture/compare register C01	CCC01	R/W			○	0000H
FFFFF606H	Timer mode control register C00	TMCC00	R/W	○	○		00H
FFFFF608H	Timer mode control register C01	TMCC01	R/W		○		20H
FFFFF609H	Valid edge select register C0	SESC0	R/W		○		00H
FFFFF610H	Timer C1	TMC1	R			○	0000H
FFFFF612H	Capture/compare register C10	CCC10	R/W			○	0000H
FFFFF614H	Capture/compare register C11	CCC11	R/W			○	0000H
FFFFF616H	Timer mode control register C10	TMCC10	R/W	○	○		00H
FFFFF618H	Timer mode control register C11	TMCC11	R/W		○		20H
FFFFF619H	Valid edge select register C1	SESC1	R/W		○		00H
FFFFF800H	Peripheral command register	PHCMD	W		○		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 bit	8 bits	16 bits	
FFFFF802H	Peripheral status register	PHS	R/W	○	○		00H
FFFFF810H	DMA trigger factor register 0	DTFR0	R/W	○	○		00H
FFFFF812H	DMA trigger factor register 1	DTFR1	R/W	○	○		00H
FFFFF814H	DMA trigger factor register 2	DTFR2	R/W	○	○		00H
FFFFF816H	DMA trigger factor register 3	DTFR3	R/W	○	○		00H
FFFFF820H	Power save mode register	PSMR	R/W	○	○		00H
FFFFF822H	Clock control register	CKC	R/W		○		00H
FFFFF824H	Lock register	LOCKR	R	○	○		0xH
FFFFF880H	External interrupt mode register 0	INTM0	R/W	○	○		00H
FFFFF882H	External interrupt mode register 1	INTM1	R/W		○		00H
FFFFF884H	External interrupt mode register 2	INTM2	R/W		○		00H
★ FFFFF8A0H	DMA terminal count output control register	DTOC	R/W	○	○		01H
FFFFF900H	Clocked serial interface mode register 0	CSIM0	R/W	○	○		00H
FFFFF901H	Clocked serial interface clock select register 0	CSIC0	R/W		○		00H
FFFFF902H	Serial I/O shift register 0	SIO0	R		○		00H
FFFFF903H	Receive-only serial I/O shift register 0	SIOE0	R		○		00H
FFFFF904H	Clocked serial interface transmit buffer register 0	SOTB0	R/W		○		00H
FFFFF910H	Clocked serial interface mode register 1	CSIM1	R/W	○	○		00H
FFFFF911H	Clocked serial interface clock select register 1	CSIC1	R/W		○		00H
FFFFF912H	Serial I/O shift register 1	SIO1	R		○		00H
FFFFF913H	Receive-only serial I/O shift register 1	SIOE1	R		○		00H
FFFFF914H	Clocked serial interface transmit buffer register 1	SOTB1	R/W		○		00H
FFFFFA00H	Asynchronous serial interface mode register 0	ASIM0	R/W	○	○		01H
FFFFFA02H	Receive buffer register 0	RXB0	R		○		FFH
FFFFFA03H	Asynchronous serial interface status register 0	ASIS0	R		○		00H
FFFFFA04H	Transmit buffer register 0	TXB0	R/W		○		FFH
FFFFFA05H	Asynchronous serial interface transmit status register 0	ASIF0	R	○	○		00H
FFFFFA06H	Clock select register 0	CKSR0	R/W		○		00H
FFFFFA07H	Baud rate generator control register 0	BRGC0	R/W		○		FFH
FFFFFA10H	Asynchronous serial interface mode register 1	ASIM1	R/W	○	○		01H
FFFFFA12H	Receive buffer register 1	RXB1	R		○		FFH
FFFFFA13H	Asynchronous serial interface status register 1	ASIS1	R		○		00H
FFFFFA14H	Transmit buffer register 1	TXB1	R/W		○		FFH
FFFFFA15H	Asynchronous serial interface transmit status register 1	ASIF1	R	○	○		00H
FFFFFA16H	Clock select register 1	CKSR1	R/W		○		00H
FFFFFA17H	Baud rate generator control register 1	BRGC1	R/W		○		FFH

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 V850E/MA2 has two specific registers, the power save control register (PSC) (refer to **9.5.2 (3) Power save control register (PSC)**) and clock control register (CKC) (refer to **9.3.4 Clock control register (CKC)**). Disable DMA transfer when writing to a specific register.

There are also two protection registers supporting write operations for specific registers to avoid an unexpected stoppage of the application system due to erroneous program execution. These two registers are the command register (PRCMD) and peripheral command register (PHCMD) (refer to **9.5.2 (2) Command register (PRCMD)** and **9.3.3 Peripheral command register (PHCMD)**).

★ 3.4.10 System wait control register (VSWC)

The system wait control register (VSWC) is a register that controls the bus access wait for the on-chip peripheral I/O registers.

Access to on-chip peripheral I/O registers is made in 3 clocks (without wait), however, in the V850E/MA2, waits may be required depending on the operating frequency. Set the values described in the table below to the VSWC register in accordance with the operating frequency used.

This register can be read/written in 8-bit units (address: FFFFF06EH, initial value: 77H).

Operating Frequency (f _{xx})	Set Value of VSWC	Number of Waits for On-Chip Peripheral I/O Register Access
4 MHz ≤ f _{xx} < 33 MHz	11H	2
33 MHz ≤ f _{xx} ≤ 40 MHz	12H	3

Remark If the timing of changing a count value conflicts with the timing of accessing a register when accessing a register with status flags that indicate the status of the internal peripheral functions (such as ASIFn) or a register that indicates the count value of a timer (such as TMCn), the register access is retried. As a result, it may take longer to access an internal peripheral I/O register.

3.4.11 Cautions

When using the V850E/MA2, the following registers must be set in the beginning.

- System wait control register (VSWC)
(Refer to **3.4.10 System wait control register (VSWC)**)
- Clock control register (CKC)
(Refer to **9.3.4 Clock control register (CKC)**)

After setting VSWC and CKC, set other registers if necessary.

To use the external bus, initialize each register in the following sequence after setting the above registers.

- <1> Set each pin to the control mode by setting each port-related register.
- <2> Select a chip select space by using chip area select control register n (CSCn) (n = 0, 1).
- <3> Specify the type of memory of each chip select space by using bus cycle type configuration register n (BCTn).

CHAPTER 4 BUS CONTROL FUNCTION

The V850E/MA2 is provided with an external bus interface function by which external I/O and memories, such as ROM and RAM, can be connected.

4.1 Features

- 16-bit/8-bit data bus sizing function
- 4-space chip select function
- Wait function
 - Programmable wait function, through which up to 7 wait states can be inserted for each memory block
 - External wait function via $\overline{\text{WAIT}}$ pin
- Idle state insertion function
- Bus mastership arbitration function
- Bus hold function
- External device connection enabled via bus control/port alternate function pins

4.2 Bus Control Pins

The following pins are used for connection to external devices.

Bus Control Pin (Function When in Control Mode)	Function When in Port Mode	Register for Port/Control Mode Switching
Data bus (D0 to D15)	PDL0 to PDL15 (Port DL)	PMCDL
Address bus (A0 to A15)	PAL0 to PAL15 (Port AL)	PMCAL
Address bus (A16 to A24)	PAH0 to PAH8 (Port AH)	PMCAH
Chip select ($\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$)	PCS0, PCS3, PCS4, PCS7 (Port CS)	PMCCS
SDRAM sync control (SDCKE, SDCLK)	PCD0, PCD1 (Port CD)	PMCCD
Byte access control/SDRAM control ($\overline{LBE}/\overline{SDCAS}$, $\overline{UBE}/\overline{SDRAS}$)	PCD2, PCD3 (Port CD)	
Read/write control ($\overline{LWR}/\overline{LDQM}$, $\overline{UWR}/\overline{UDQM}$, \overline{RD} , \overline{WE})	PCT0, PCT1, PCT4, PCT5 (Port CT)	PMCCT
External wait control (\overline{WAIT})	PCM0 (Port CM)	PMCCM
Internal system clock (CLKOUT)	PCM1 (Port CM)	
Bus hold control (\overline{HLDRQ} , \overline{HLDK})	PCM2, PCM3 (Port CM)	
DRAM refresh control (\overline{REFRQ})	PCM4 (Port CM)	

Remark When the system is reset, each bus control pin becomes unconditionally valid. (However, D8 to D15 are valid only in ROMless mode 0.)

★ 4.2.1 Pin status during internal RAM and peripheral I/O access

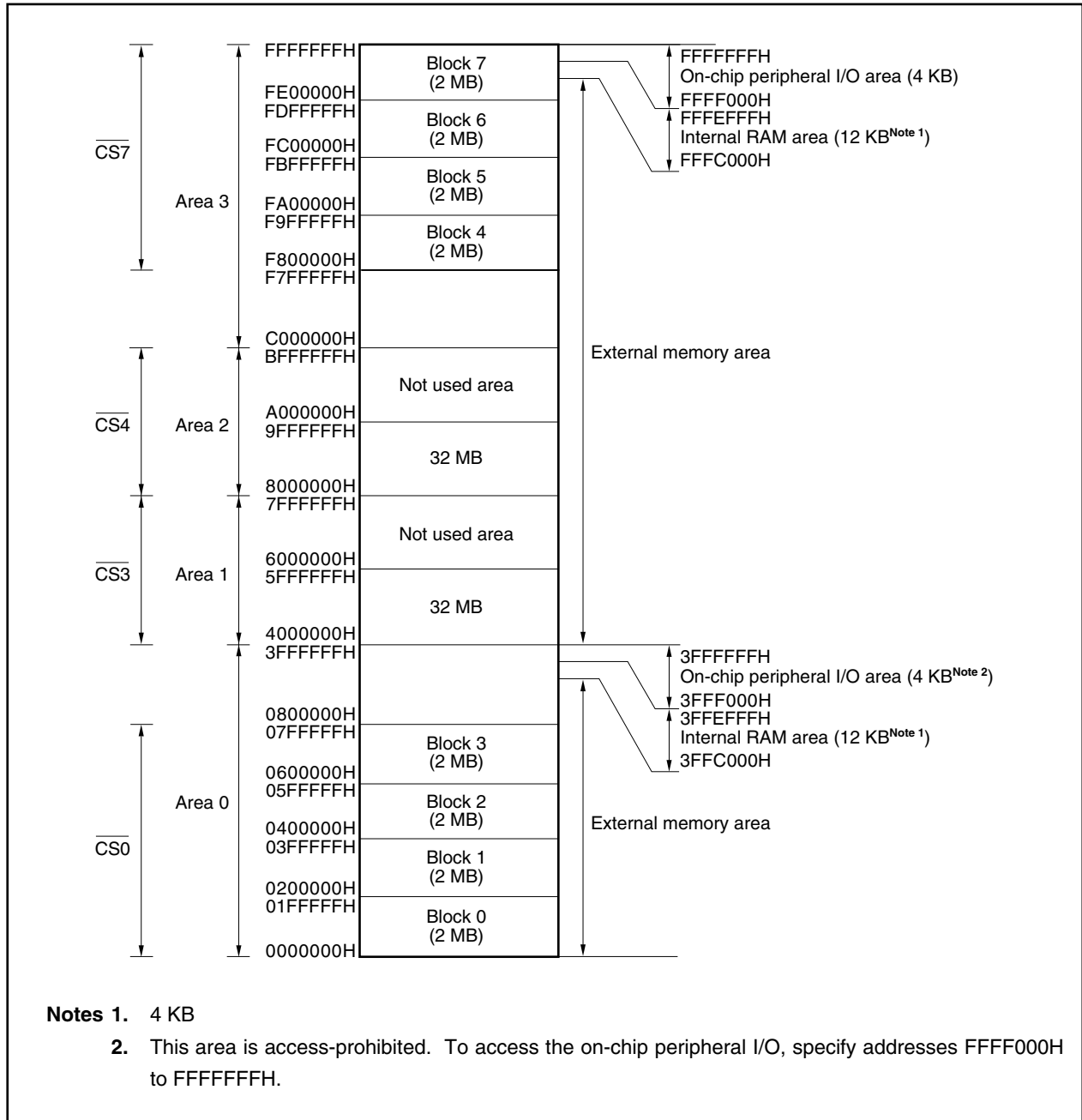
While accessing internal RAM, the address bus becomes undefined, and the data bus is not output and enters the high-impedance state. The external bus control signals become inactive.

While accessing peripheral I/O, the address bus outputs the address data of the on-chip peripheral I/O currently being accessed. The data bus enters the output state when write-accessing the peripheral I/O, and the high-impedance state when read-accessing the peripheral I/O. The external bus control signals become inactive.

4.3 Memory Block Function

The 80 MB memory space is divided into memory blocks of 2 MB and 32 MB units. The programmable wait function and bus cycle operation mode can be independently controlled for each block.

The area that can be used as program area is the space of the 8 MB of addresses 0000000H to 07FFFFFFH and internal RAM area.



Notes 1. 4 KB

2. This area is access-prohibited. To access the on-chip peripheral I/O, specify addresses FFFF000H to FFFFFFFFH.



4.3.1 Chip select control function

Of the 80 MB memory area, the lower 8 MB (0000000H to 07FFFFFFH) and the higher 8 MB (F800000H to FFFFFFFH) can be divided into 2 MB memory blocks by chip area selection control registers 0 and 1 (CSC0, CSC1) to control the chip select signal.

The memory area can be effectively used by dividing it into memory blocks using the chip select control function. The priority order is described below.

(1) Chip area selection control registers 0, 1 (CSC0, CSC1)

These registers can be read/written in 16-bit units and become valid by setting each bit to 1.

★ **Caution** Write to the CSC0 and CSC1 registers after reset, and then do not change the set value.

CSC0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address FFFFFF060H	After reset 2C11H
	CS33	CS32	CS31	CS30	1>Note	1>Note	0>Note	0>Note	0>Note	0>Note	0>Note	1>Note	CS03	CS02	CS01	CS00		
CSC1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address FFFFFF062H	After reset 2C11H
	CS43	CS42	CS41	CS40	1>Note	1>Note	0>Note	0>Note	0>Note	0>Note	0>Note	1>Note	CS73	CS72	CS71	CS70		

Note The operation of the system is not guaranteed when a value other than the initial value is set to these bits.

Bit Position	Bit Name	Function	
15 to 12, 3 to 0	CSnm (n = 0, 3, 4, 7) (m = 0 to 3)	Chip Select Chip select enabled by setting CSnm bit to 1.	
		CSnm	CS operation
		CS00	$\overline{CS0}$ output during block 0 access
		CS01	$\overline{CS0}$ output during block 1 access.
		CS02	$\overline{CS0}$ output during block 2 access.
		CS03	$\overline{CS0}$ output during block 3 access.
		CS30 to CS33	Setting has no meaning.
		CS40 to CS43	Setting has no meaning.
		CS70	$\overline{CS7}$ output during block 7 access.
		CS71	$\overline{CS7}$ output during block 6 access.
		CS72	$\overline{CS7}$ output during block 5 access.
CS73	$\overline{CS7}$ output during block 4 access.		

4.4 Bus Cycle Type Control Function

In the V850E/MA2, the following external devices can be connected directly to each memory block.

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

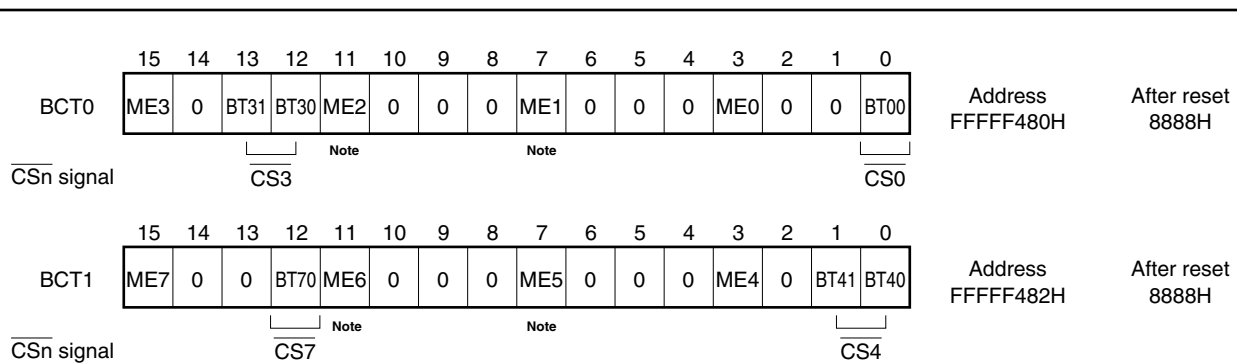
Connected external devices are specified by bus cycle type configuration registers 0, 1 (BCT0, BCT1).

(1) Bus cycle type configuration registers 0, 1 (BCT0, BCT1)

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

- ★ Be sure to set bits 14, 10 to 8, 6 to 4, 2, and 1 of the BCT0 register and bits 14, 13, 10 to 8, 6 to 4, and 2 of the BCT1 register to 0. If they are set to 1, the operation is not guaranteed.

Caution Write to the BCT0 and BCT1 registers 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 BCT0 and BCT1 registers is complete. However, it is possible to access external memory areas whose initialization settings are complete.



Note Make sure that the ME1, ME2, ME5, and ME6 bits are cleared to 0 at initialization.

Bit Position	Bit Name	Function															
15, 11, 7, 3 (BCT0), 15, 11, 7, 3 (BCT1)	ME _n (n = 0 to 7)	Memory Controller Enable Sets memory controller operation enable for each chip select. <table border="1" style="margin-left: 20px;"> <tr> <td>ME_n</td> <td>Memory controller operation enable</td> </tr> <tr> <td>0</td> <td>Operation disabled</td> </tr> <tr> <td>1</td> <td>Operation enabled</td> </tr> </table>	ME _n	Memory controller operation enable	0	Operation disabled	1	Operation enabled									
ME _n	Memory controller operation enable																
0	Operation disabled																
1	Operation enabled																
0 (BCT0), 12 (BCT1)	BT _{n0} (n = 0, 7)	Bus Cycle Type Specifies the device to be connected to the \overline{CS}_n signal. <table border="1" style="margin-left: 20px;"> <tr> <td>BT_{n0}</td> <td>External device connected directly to \overline{CS}_n signal</td> </tr> <tr> <td>0</td> <td>SRAM, external I/O</td> </tr> <tr> <td>1</td> <td>Page ROM</td> </tr> </table>	BT _{n0}	External device connected directly to \overline{CS}_n signal	0	SRAM, external I/O	1	Page ROM									
BT _{n0}	External device connected directly to \overline{CS}_n signal																
0	SRAM, external I/O																
1	Page ROM																
13, 12 (BCT0), 1, 0 (BCT1)	BT _{n1} , BT _{n0} (n = 3, 4)	Bus Cycle Type Specifies the device to be connected to the \overline{CS}_n signal. <table border="1" style="margin-left: 20px;"> <tr> <td>BT_{n1}</td> <td>BT_{n0}</td> <td>External device connected directly to \overline{CS}_n signal</td> </tr> <tr> <td>0</td> <td>0</td> <td>SRAM, external I/O</td> </tr> <tr> <td>0</td> <td>1</td> <td>Page ROM</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>SDRAM</td> </tr> </table>	BT _{n1}	BT _{n0}	External device connected directly to \overline{CS}_n signal	0	0	SRAM, external I/O	0	1	Page ROM	1	0	Setting prohibited	1	1	SDRAM
BT _{n1}	BT _{n0}	External device connected directly to \overline{CS}_n signal															
0	0	SRAM, external I/O															
0	1	Page ROM															
1	0	Setting prohibited															
1	1	SDRAM															

4.5 Bus Access

★ **4.5.1 Number of access clocks**

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

Bus Cycle Configuration Resource (Bus Width)	Instruction Fetch	Operand Data Access
Internal RAM (32 Bits)	1 ^{Note}	1

Note When bus access conflicts with a data access: 2

Remark Unit: Clock/access

4.5.2 Bus sizing function

The bus sizing function controls the data bus width for each CS space. 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.

★ Be sure to set bits 15, 13, 11, 9, 7, 5, 3, and 1 to 0. If they are set to 1, the operation is not guaranteed.

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 external memory areas whose initialization settings are complete.

2. When the data bus width is specified as 8 bits, only the signals shown below become active.

LWR: When accessing SRAM, external ROM, or external I/O (write cycle)

	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																		
BSC	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">BS70</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">0/1^{Note 1}</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">0/1^{Note 1}</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">BS40</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">BS30</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">0/1^{Note 1}</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">0/1^{Note 1}</td> <td style="width: 15px; text-align: center;">0</td> <td style="width: 15px; text-align: center;">BS00</td> </tr> </table>	0	BS70	0	0/1 ^{Note 1}	0	0/1 ^{Note 1}	0	BS40	0	BS30	0	0/1 ^{Note 1}	0	0/1 ^{Note 1}	0	BS00	Address FFFFFF066H	After reset ^{Note 2} 0000H/5555H
0	BS70	0	0/1 ^{Note 1}	0	0/1 ^{Note 1}	0	BS40	0	BS30	0	0/1 ^{Note 1}	0	0/1 ^{Note 1}	0	BS00				
CSn signal	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="width: 15px; text-align: center;">CS7</td> <td style="width: 15px; text-align: center;">CS4</td> <td style="width: 15px; text-align: center;">CS3</td> <td style="width: 15px; text-align: center;">CS0</td> </tr> </table>	CS7	CS4	CS3	CS0														
CS7	CS4	CS3	CS0																

Notes 1. The operation of the system is not guaranteed when a value other than the initial value is set to these bits.

2. When in ROM-less mode 0: 5555H
When in ROM-less mode 1: 0000H

Bit Position	Bit Name	Function
14, 8, 6, 0	BSn0 (n = 0, 3, 4, 7)	Data Bus Width Sets the data bus width of CSn space.

BSn0	Data bus width of CSn space
0	8 bits
1	16 bits

4.5.3 Endian control function

The endian control function can be used to set processing of word data in memory using either the big endian method or the little endian method for each CS space selected with the chip select signal ($\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$). Switching of the endian method is specified using the endian configuration register (BEC).

Caution In the following areas, the data processing method is fixed to little endian, so the setting of the BEC register is invalid.

- On-chip peripheral I/O area
- Internal RAM area
- On-chip peripheral I/O area and the area identical to the internal RAM area, which are located at address 3FFFFFFH or lower
- Program area for external memory

(1) Endian configuration register (BEC)

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

Be sure to set bits 15, 13 to 9, 7, and 5 to 1 to 0. If they are set to 1, the operation is not guaranteed.

★ **Caution** Write to the BEC register after reset, and then do not change the set value.

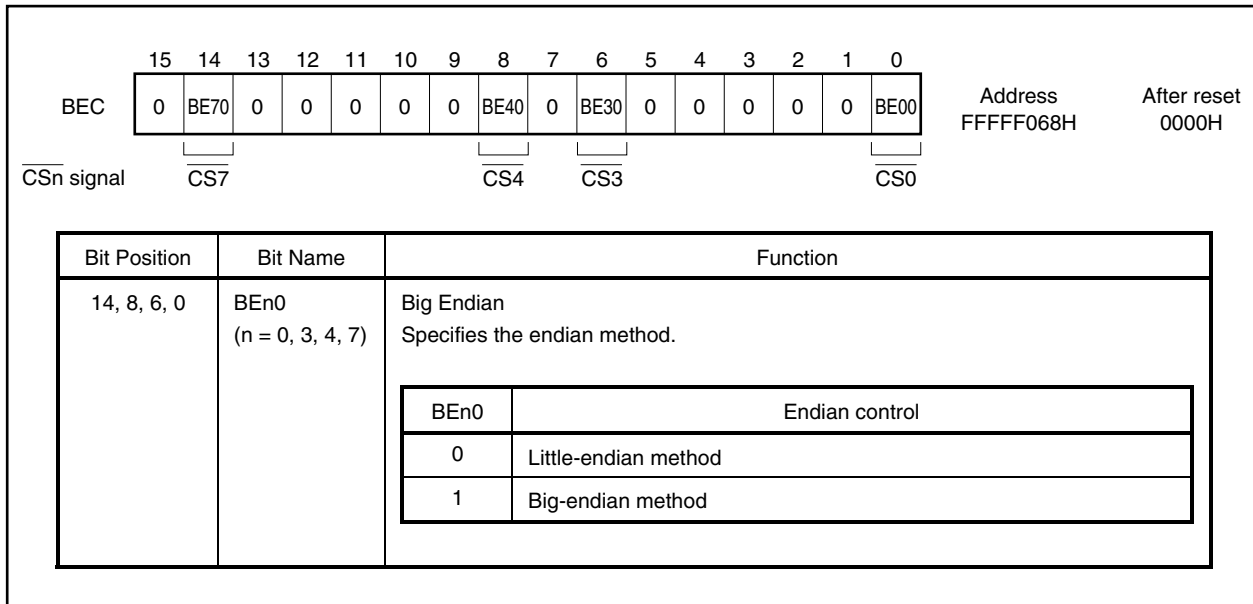


Figure 4-1. Big Endian Addresses Within Word

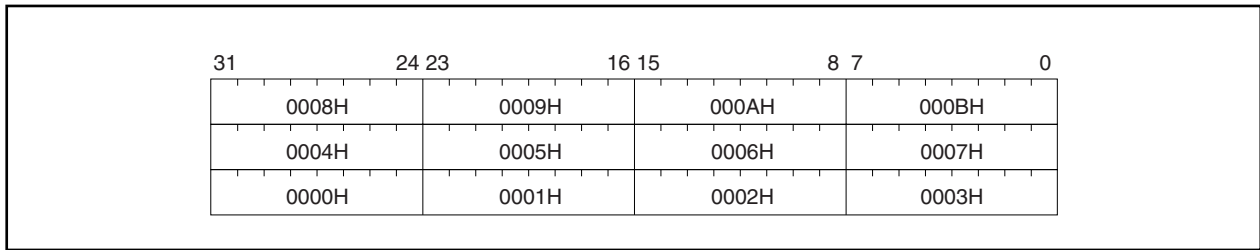
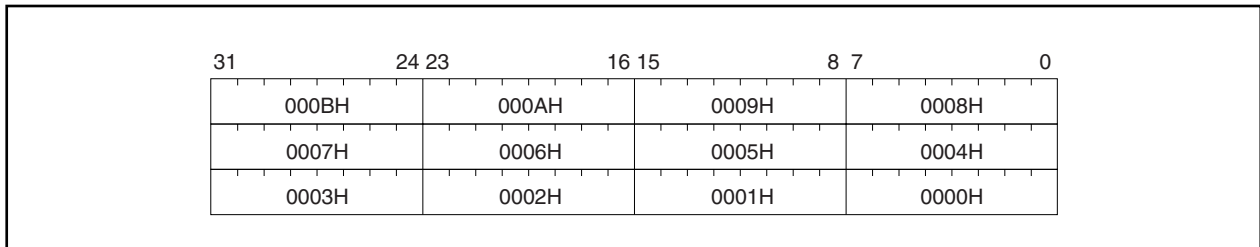


Figure 4-2. Little Endian Addresses Within Word



4.5.4 Big endian method usage restrictions in NEC Electronics development tools

(1) When using a debugger (ID850)

The big-endian method is supported only in the memory window display.

(2) When using a compiler (CA850)

(a) Restrictions in C language

- (i) There are restrictions for variables allocated to/located/in the big-endian space, as shown below.
 - union cannot be used.
 - bitfield cannot be used.
 - Access with cast (changing access size) cannot be used.
 - Variables with initial values cannot be used.
- (ii) It is necessary to specify the following optimization inhibit options because optimization may cause a change in the access size.
 - For global optimization part (opt850)... -Wo, -XTb
 - For optimization depending on model part (impr850)... -Wi, +arg_reg_opt=OFF, +std_trans_opt=OFF

The specification of the optimization inhibit option shown above is not necessary, however, if the access is not an access with cast or with masking/shifting^{Note}.

Note This is on the condition that a pattern that may cause the following optimization is not used. However, because it is very difficult for users to check the patterns completely in cases such as when several patterns are mixed (especially for optimization depending on model part), it is recommended that the optimization inhibit options shown above be specified.

[Related global optimization part]

- 1-bit set using bit or

```
int i;
i ^= 1;
```
- 1-bit clear using bit and

```
i &= ~1;
```
- 1-bit not using bit xor

```
i ^= ~1;
```
- 1-bit test using bit and

```
if(i & 1);
```

[Related optimization depending on model part]

Accessing the same variable in a different size

- Cast
- Mask
- Shift

```
Example  int i, *ip;
         char c;
         :
         c = *((char*) ip);
         :
         c = 0xff & i;
         :
         i = (i << 24) >> 24;
```

(b) Restrictions in assembly language

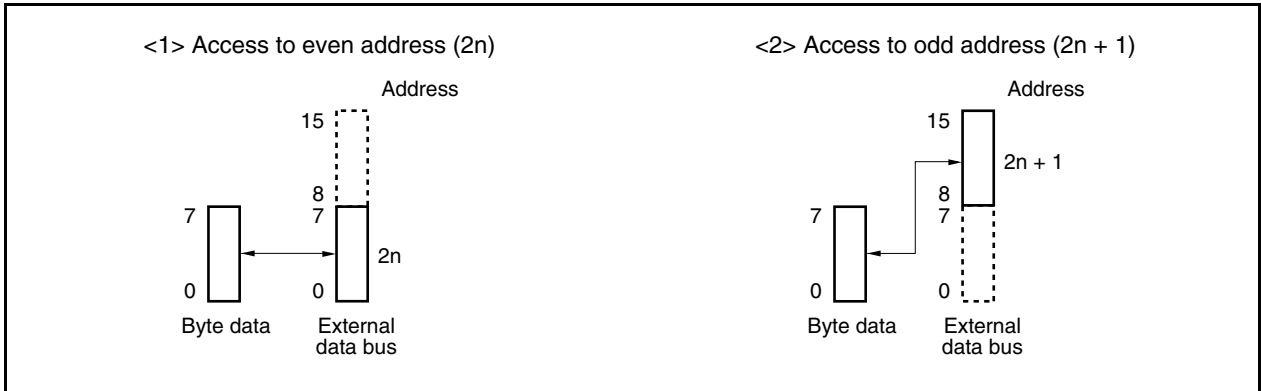
For variables located in the big endian space, a quasi directive that secures an area of other than byte size (.hword, .word, .float, .shword) is not used.

4.5.5 Bus width

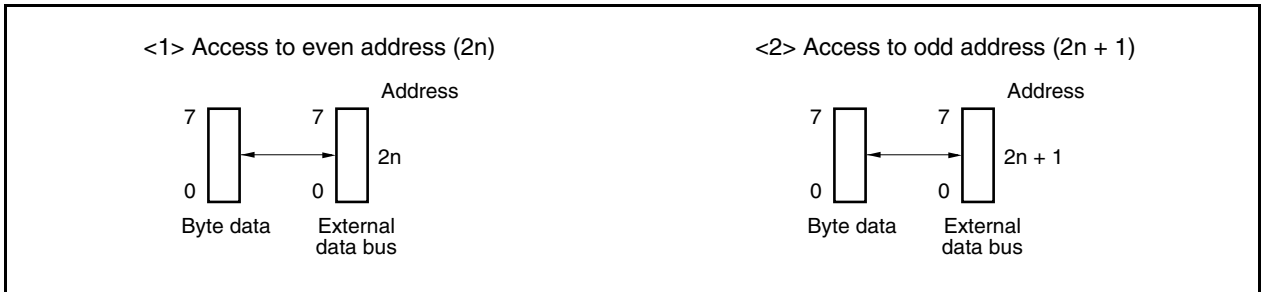
The V850E/MA2 accesses peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The following shows the operation for each type of access. Access all data in order starting from the lower order side.

(1) Byte access (8 bits)

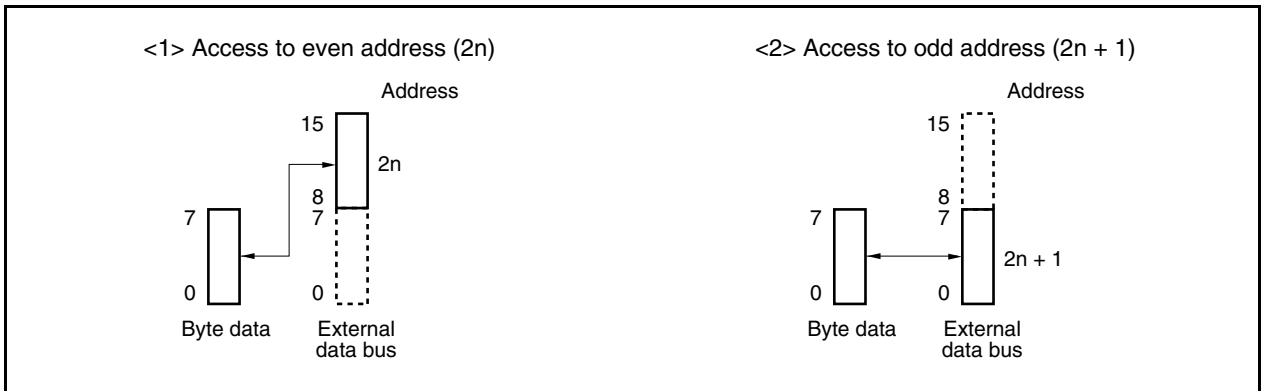
(a) When the data bus width is 16 bits (little endian)



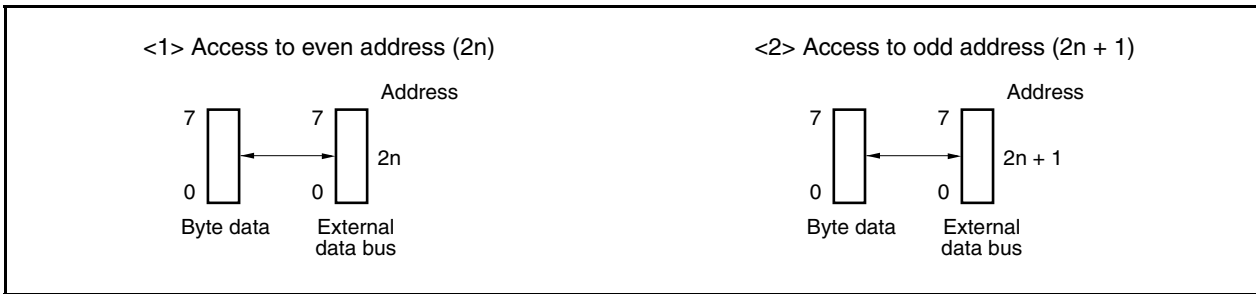
(b) When the data bus width is 8 bits (little endian)



(c) When the data bus width is 16 bits (big endian)

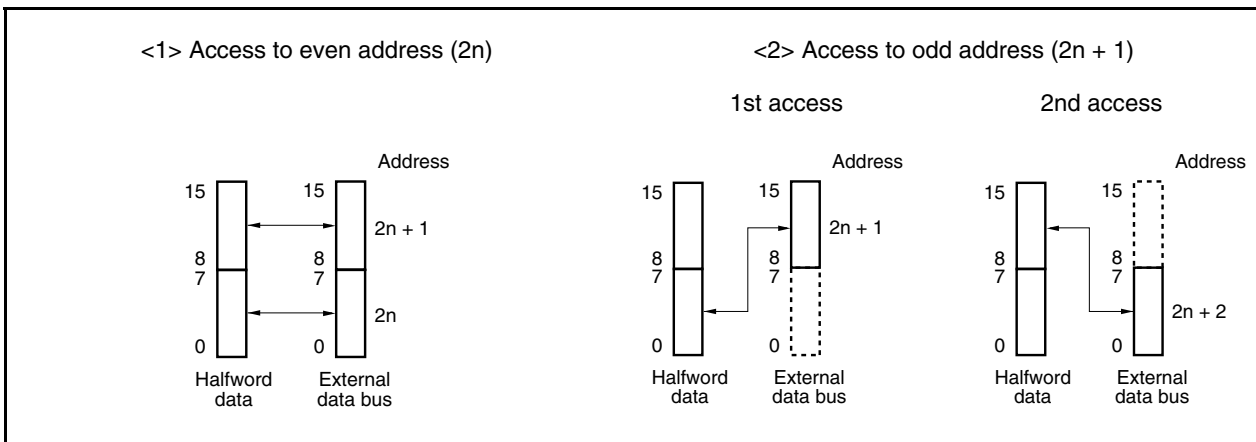


(d) When the data bus width is 8 bits (big endian)

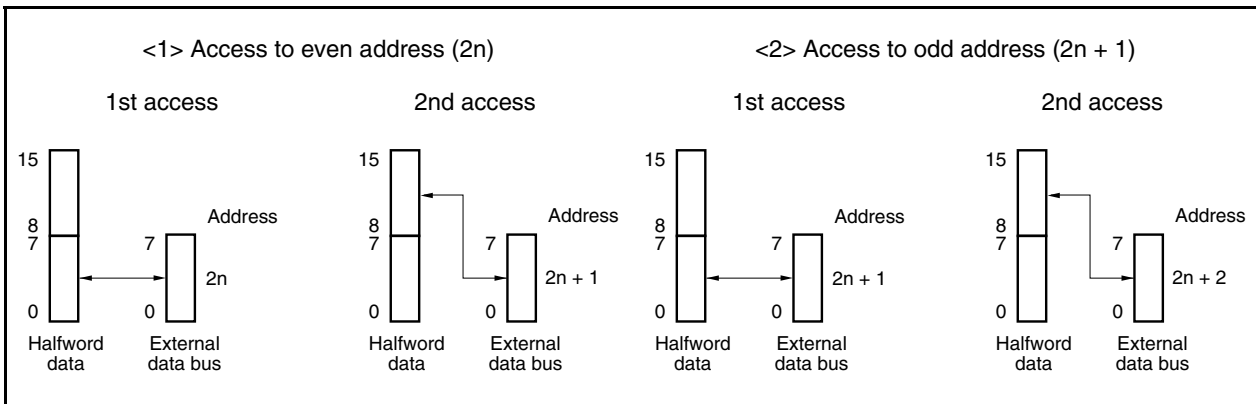


(2) Halfword access (16 bits)

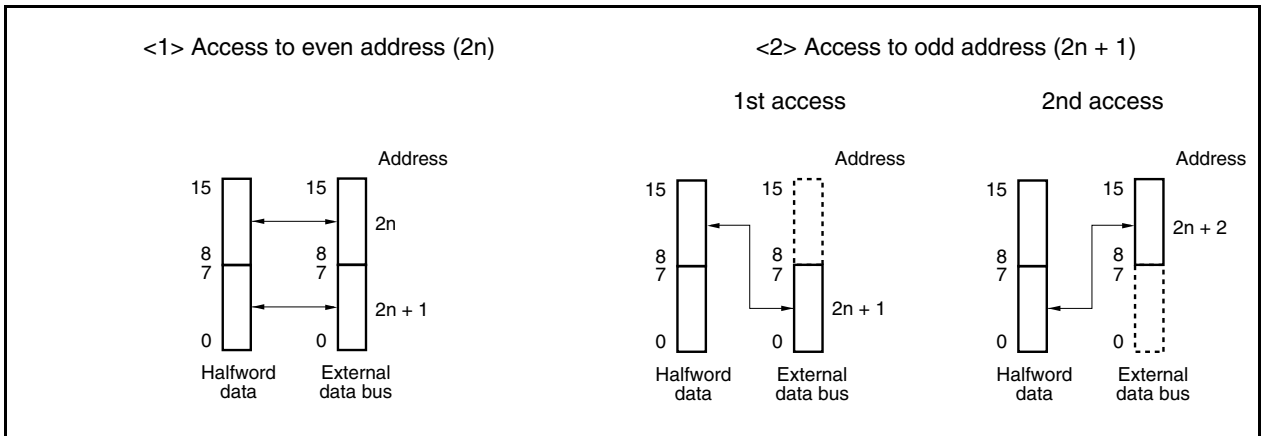
(a) When the bus width is 16 bits (little endian)



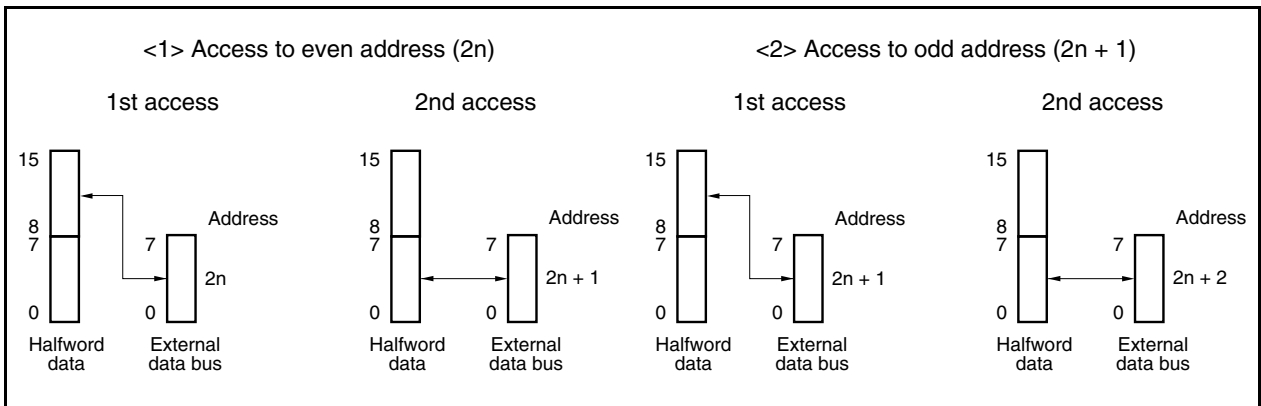
(b) When the data bus width is 8 bits (little endian)



(c) When the data bus width is 16 bits (big endian)



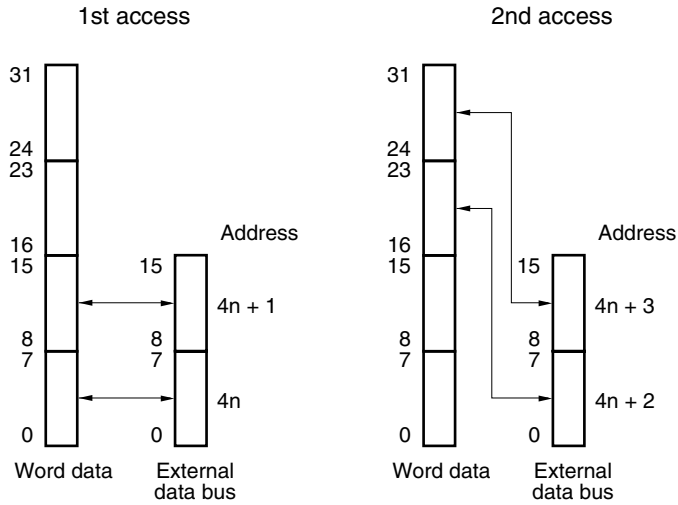
(d) When the data bus width is 8 bits (big endian)



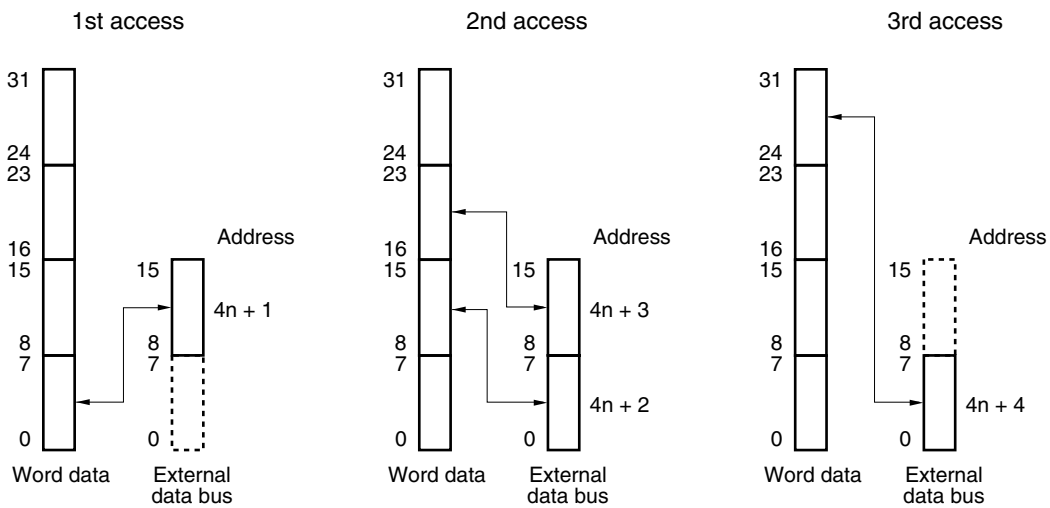
(3) Word access (32 bits)

(a) When the bus width is 16 bits (little endian) (1/2)

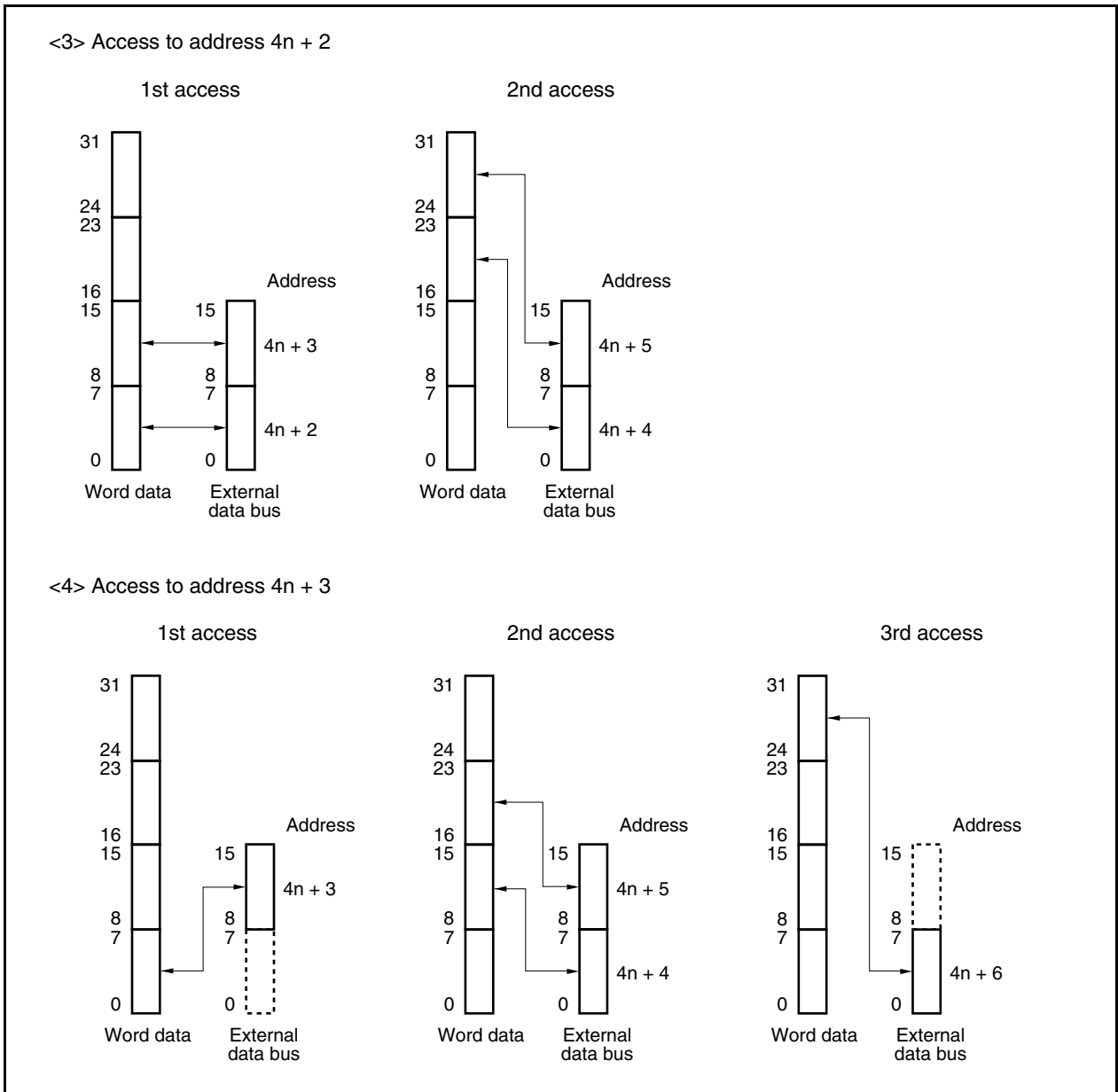
<1> Access to address $4n$



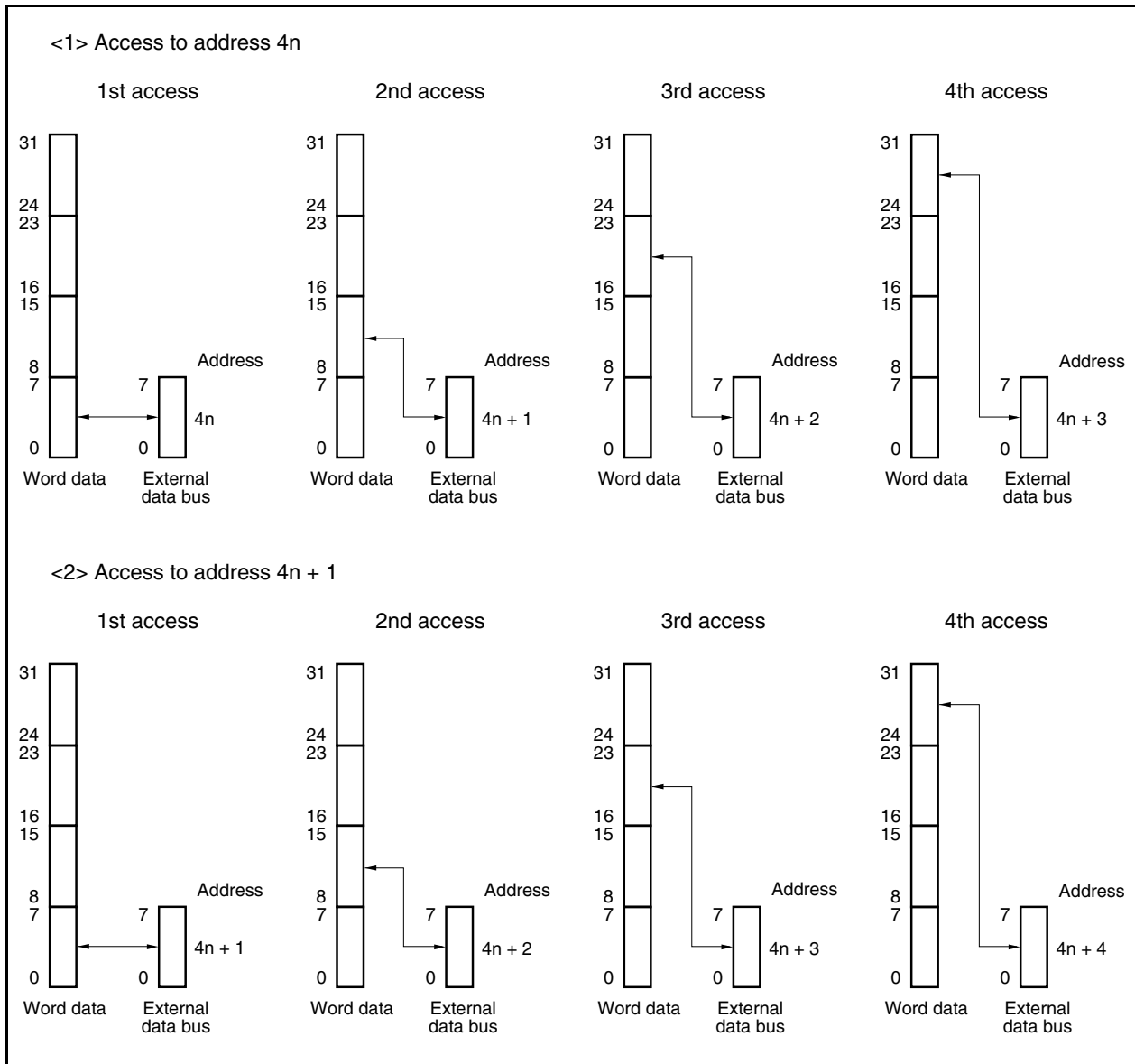
<2> Access to address $4n + 1$



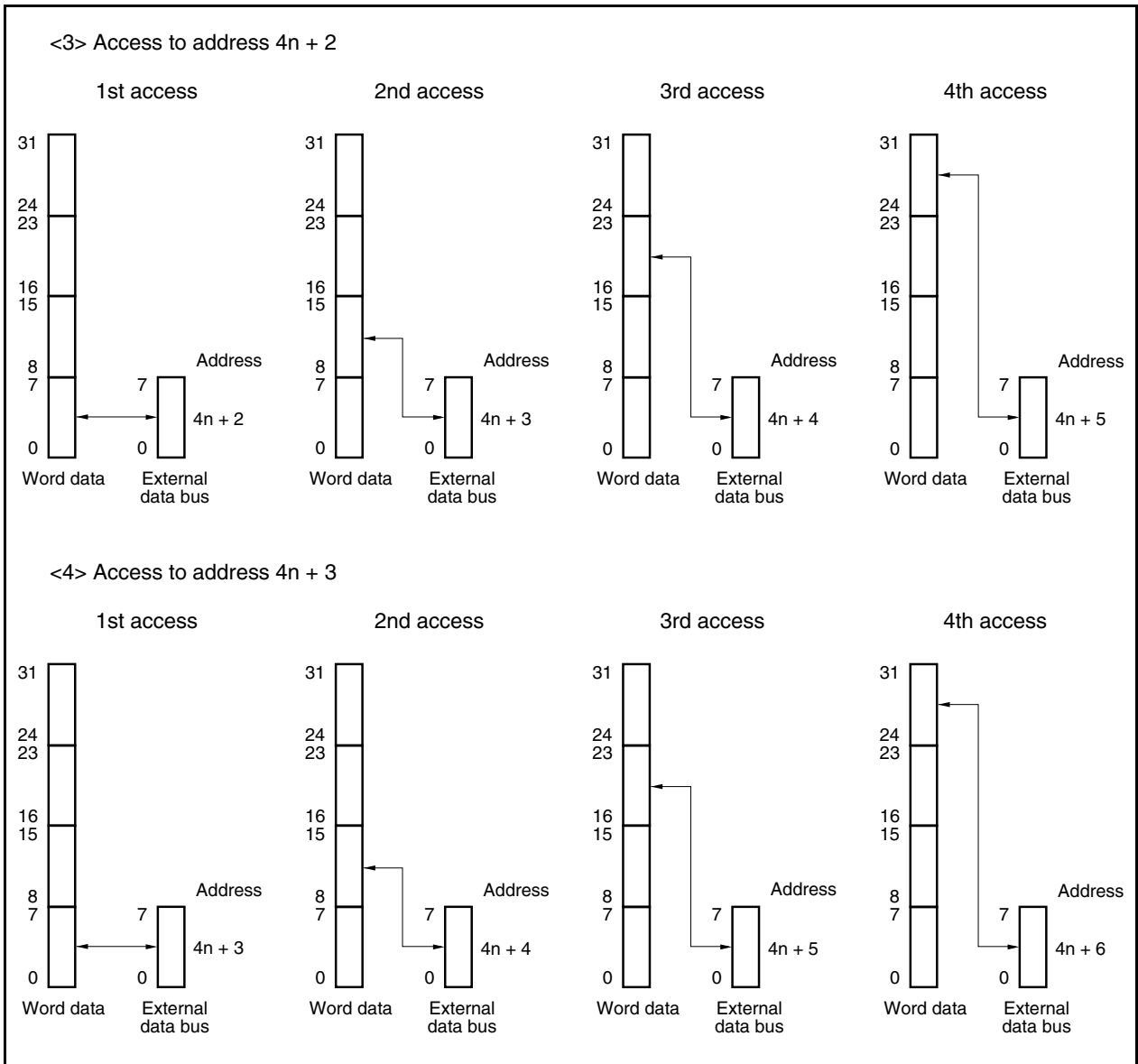
(a) When the bus width is 16 bits (little endian) (2/2)



(b) When the data bus width is 8 bits (little endian) (1/2)

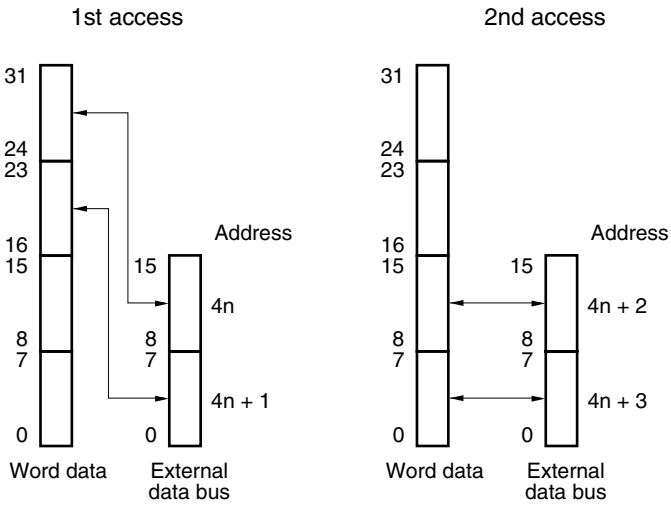


(b) When the data bus width is 8 bits (little endian) (2/2)

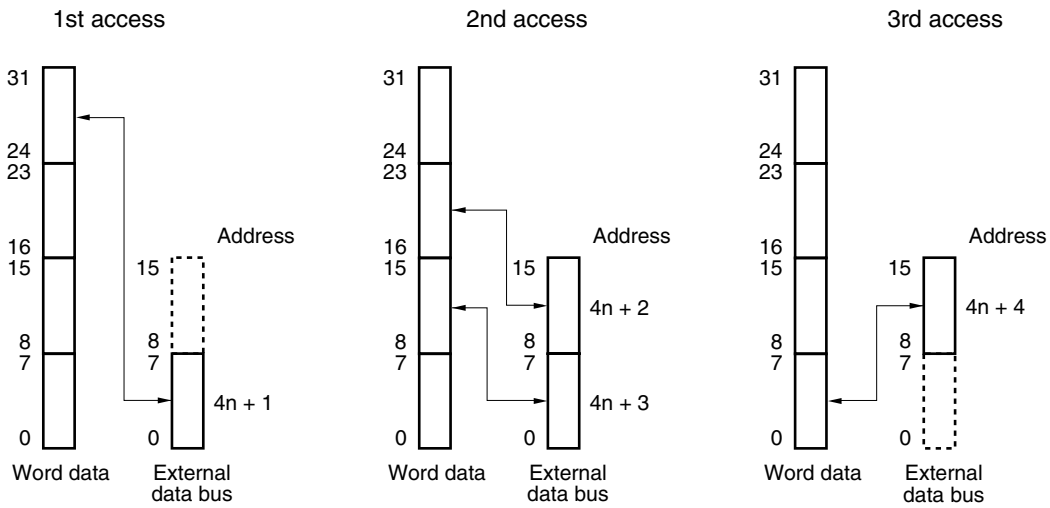


(c) When the data bus width is 16 bits (big endian) (1/2)

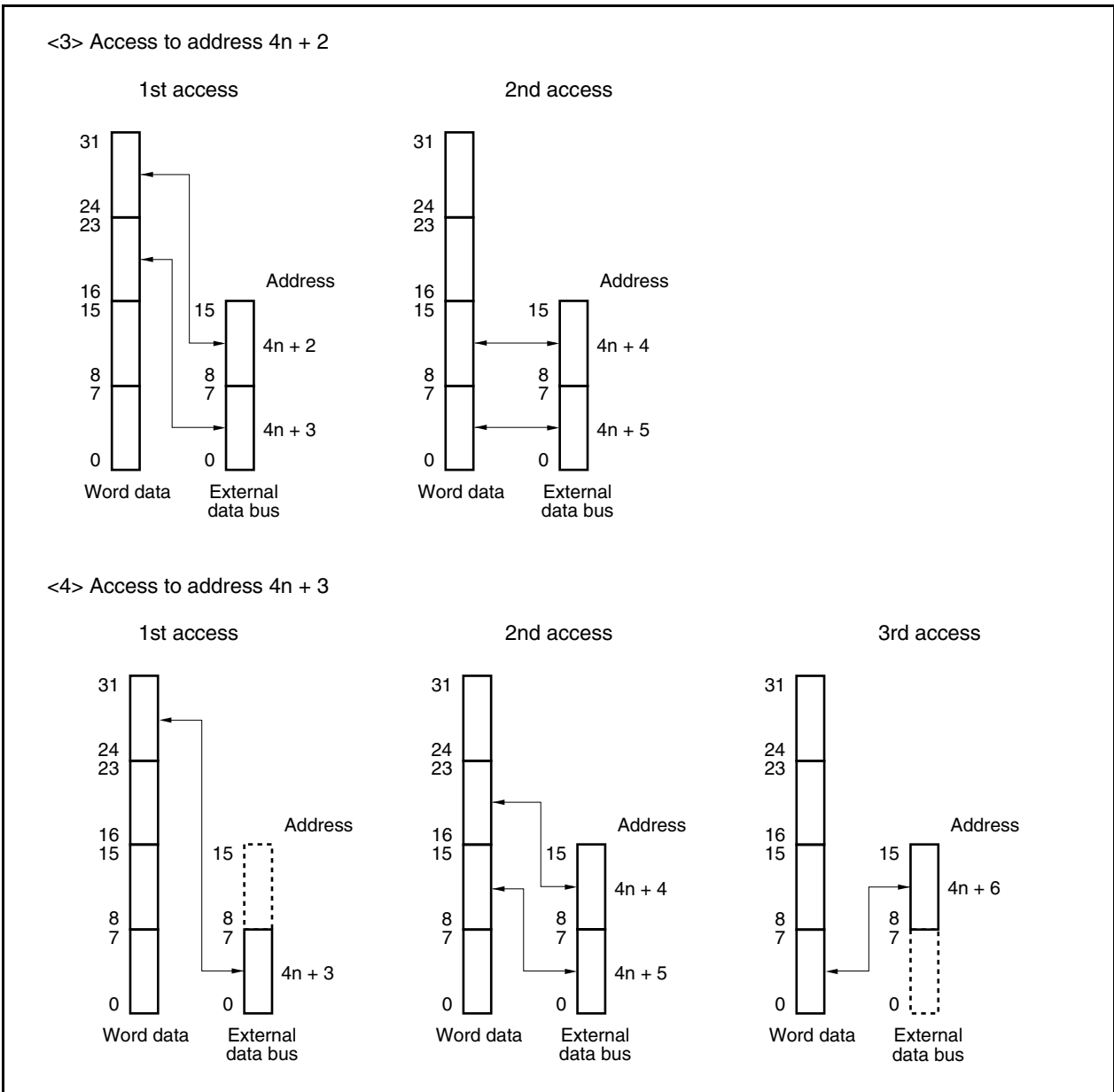
<1> Access to address $4n$



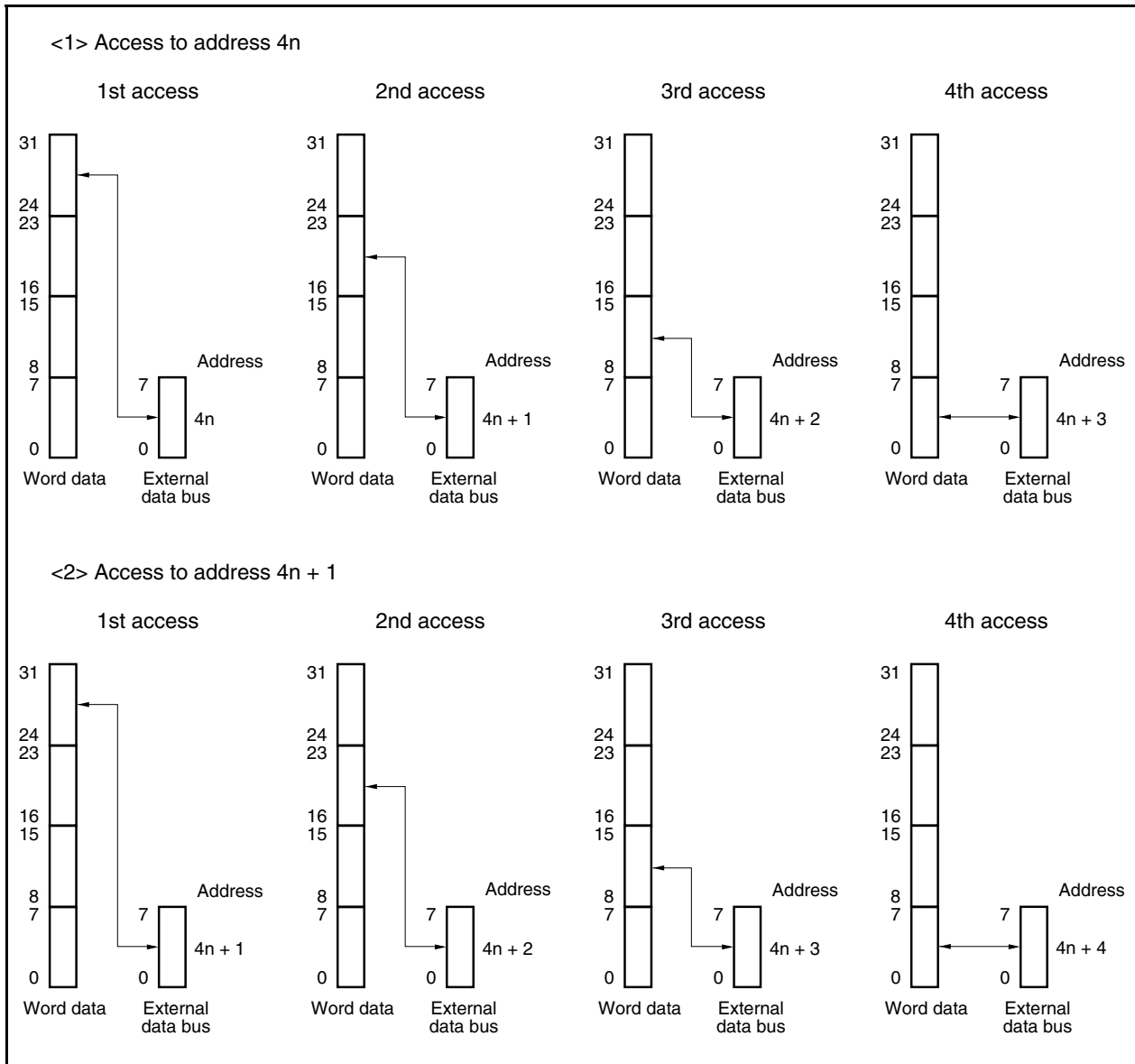
<2> Access to address $4n + 1$



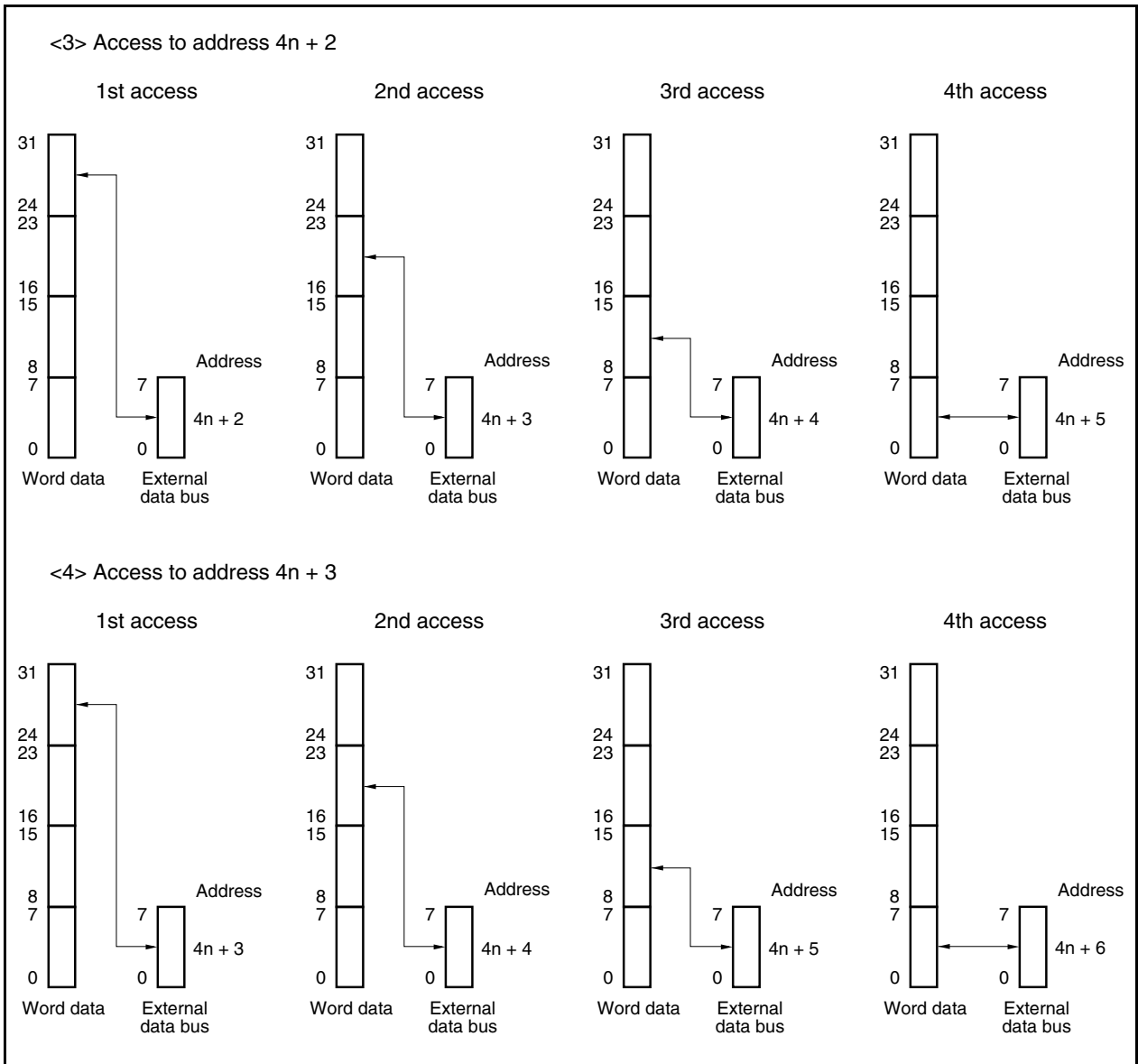
(c) When the data bus width is 16 bits (big endian) (2/2)



(d) When the data bus width is 8 bits (big endian) (1/2)



(d) When the data bus width is 8 bits (big endian) (2/2)



4.6 Wait Function

★ 4.6.1 Programmable wait function

(1) Data wait control registers 0, 1 (DWC0, DWC1)

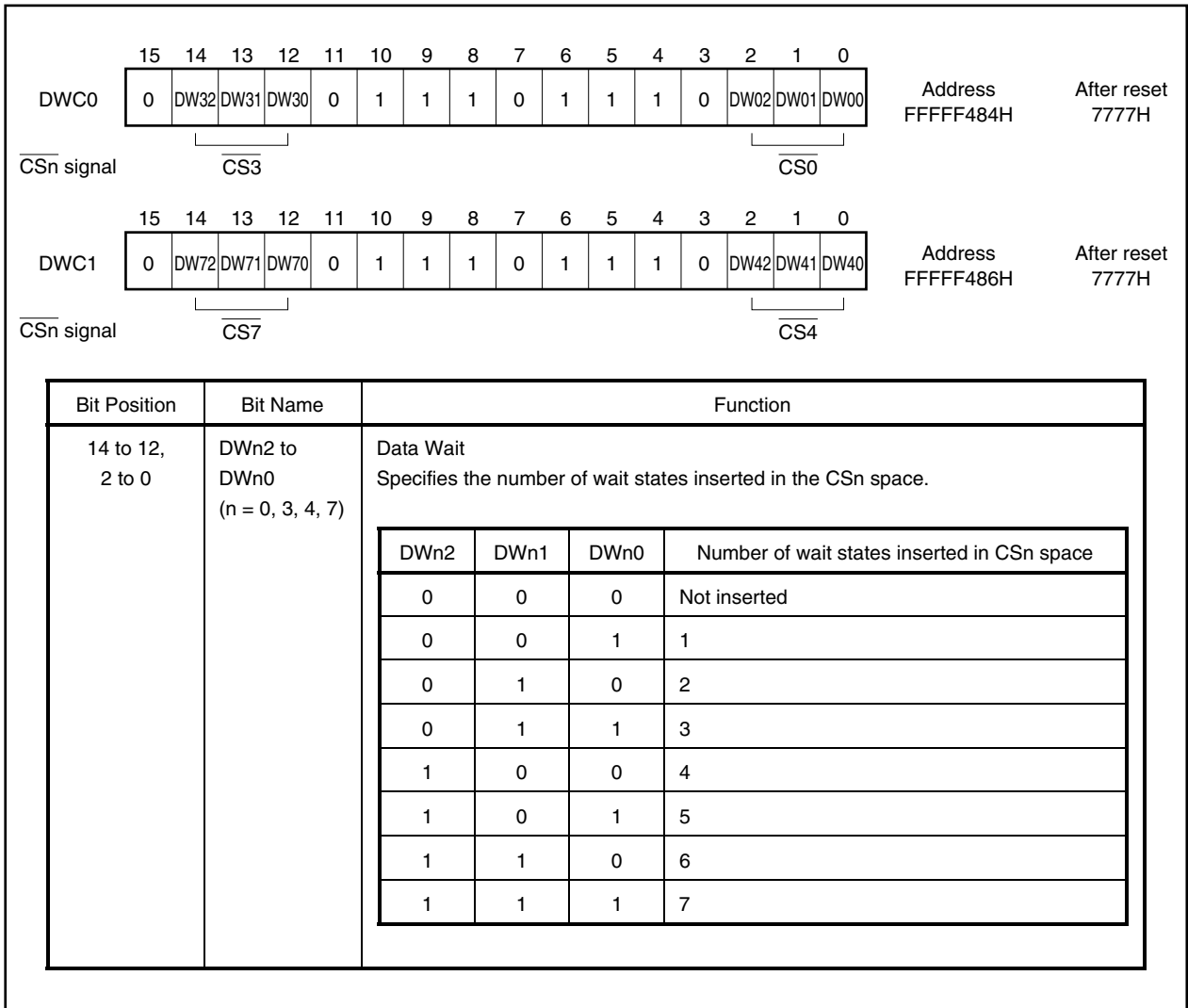
To facilitate interfacing with low-speed memory or with I/Os, it is possible to insert up to 7 data wait states in the starting bus cycle for each CS space.

The number of wait states can be specified by program using data wait control registers 0 and 1 (DWC0, DWC1). Just after system reset, all blocks have 7 data wait states inserted.

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

Be sure to set bits 10 to 8 and 6 to 4 to 1. If they are set to 0, the operation is not guaranteed.

- Cautions**
- 1. The internal RAM area is not subject to programmable waits and ordinarily no wait access is carried out. The on-chip peripheral I/O area is also not subject to programmable wait states, with wait control performed by each peripheral function only.**
 - 2. In the following cases, the settings of registers DWC0 and DWC1 are invalid (wait control is performed by each memory controller).**
 - Page ROM on-page access
 - SDRAM access
 - 3. Write to the DWC0 and DWC1 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 DWC0 and DWC1 registers is complete. However, it is possible to access external memory areas whose initialization settings are complete.**



(2) Address setup wait control register (ASC)

The V850E/MA2 allows insertion of address setup wait states before the SRAM/page ROM cycle (the setting of the ASC register in the SDRAM cycle is invalid).

The number of address setup wait states can be set with the ASC register for each CS space.

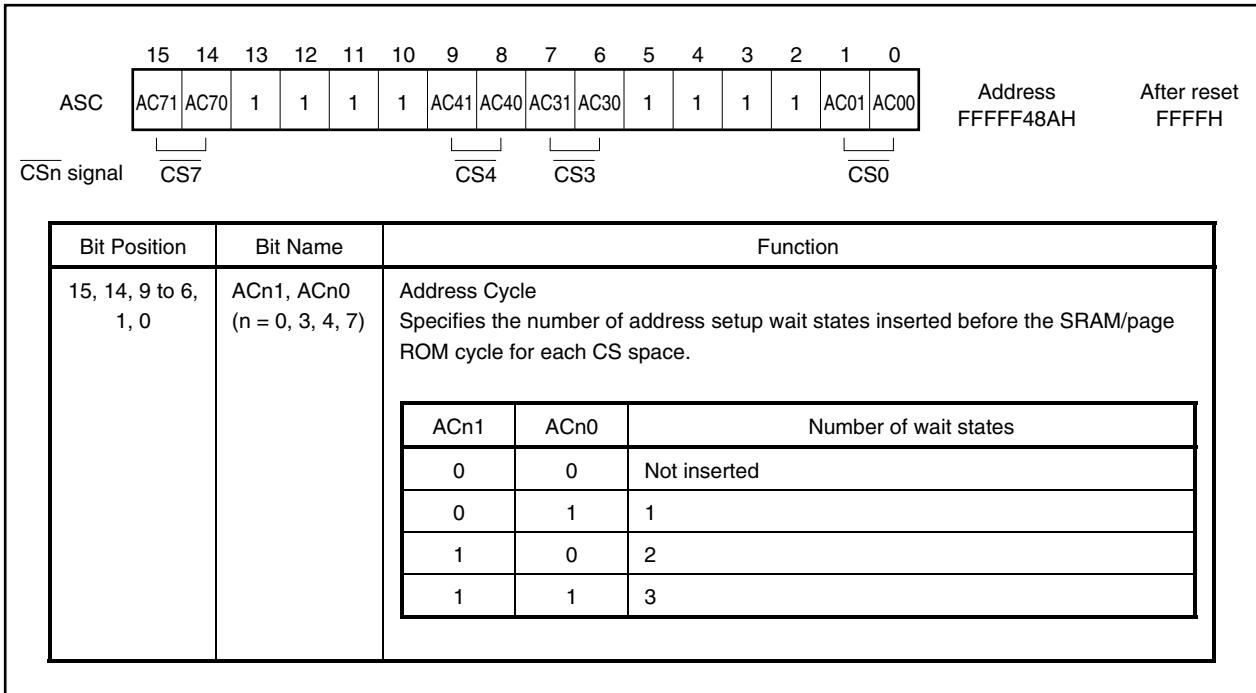
This register can be read/written in 16-bit units.

Be sure to set bits 13 to 10 and 5 to 2 to 1. If they are set to 0, the operation is not guaranteed.

Cautions 1. During address setup wait, the $\overline{\text{WAIT}}$ pin-based external wait function is disabled.

2. Write to the ASC register after reset, and then do not change the set value.

★



4.6.2 External wait function

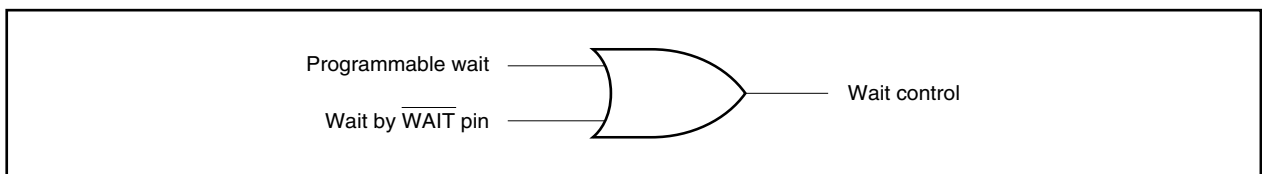
When an extremely slow device, I/O, or asynchronous system is connected, an arbitrary number of wait states can be inserted in the bus cycle by the external wait pin ($\overline{\text{WAIT}}$) for synchronization with the external device.

Just as with programmable waits, accessing internal RAM and on-chip peripheral I/O areas cannot be controlled by external waits.

The external $\overline{\text{WAIT}}$ signal can be input asynchronously to CLKOUT and is sampled at the rising 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, the wait state may or may not be inserted in the next state.

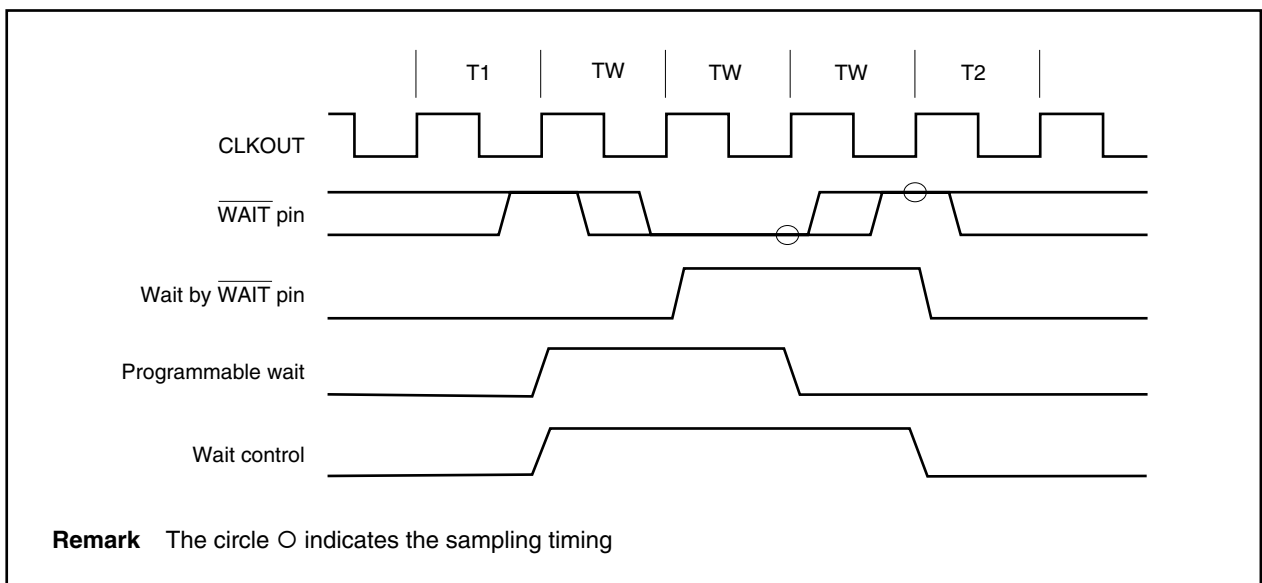
4.6.3 Relationship between programmable wait and external wait

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



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

Figure 4-3. Example of Wait Insertion



4.6.4 Bus cycles in which wait function is valid

In the V850E/MA2, the number of waits can be specified according to the memory type specified for each memory block. The following shows the bus cycles in which the wait function is valid and the registers used for wait setting.

Table 4-1. Bus Cycles in Which Wait Function Is Valid

Bus Cycle	Type of Wait	Programmable Wait Setting			Wait from $\overline{\text{WAIT}}$ Pin
		Register	Bit	Wait Count	
SRAM, external ROM, external I/O cycles	Address setup wait	ASC	ACn1, ACn0	0 to 3	– (invalid)
	Data access wait	DWC0, DWC1	DWn2 to DWn0	0 to 7	√ (valid)
Page ROM cycle	Address setup wait	ASC	ACn1, ACn0	0 to 3	– (invalid)
	Off page Data access wait	DWC0, DWC1	DWn2 to DWn0	0 to 7	√ (valid)
	On page Data access wait	PRC	PRW2 to PRW0	0 to 7	√ (valid)
SDRAM cycle	Row address precharge	SCRm	BCW1m, BCW0m	1 to 3	– (invalid)

Remark n = 0, 3, 4, 7, m = 3, 4

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 to meet the data output float delay time (t_{DF}) on memory read access for each CS space. The bus cycle following the T2 state starts after the idle state is inserted.

An idle state is inserted in the timing shown below.

- After read/write cycles for SRAM, external I/O, or external ROM
- After a read cycle for page ROM
- After a read cycle for SDRAM

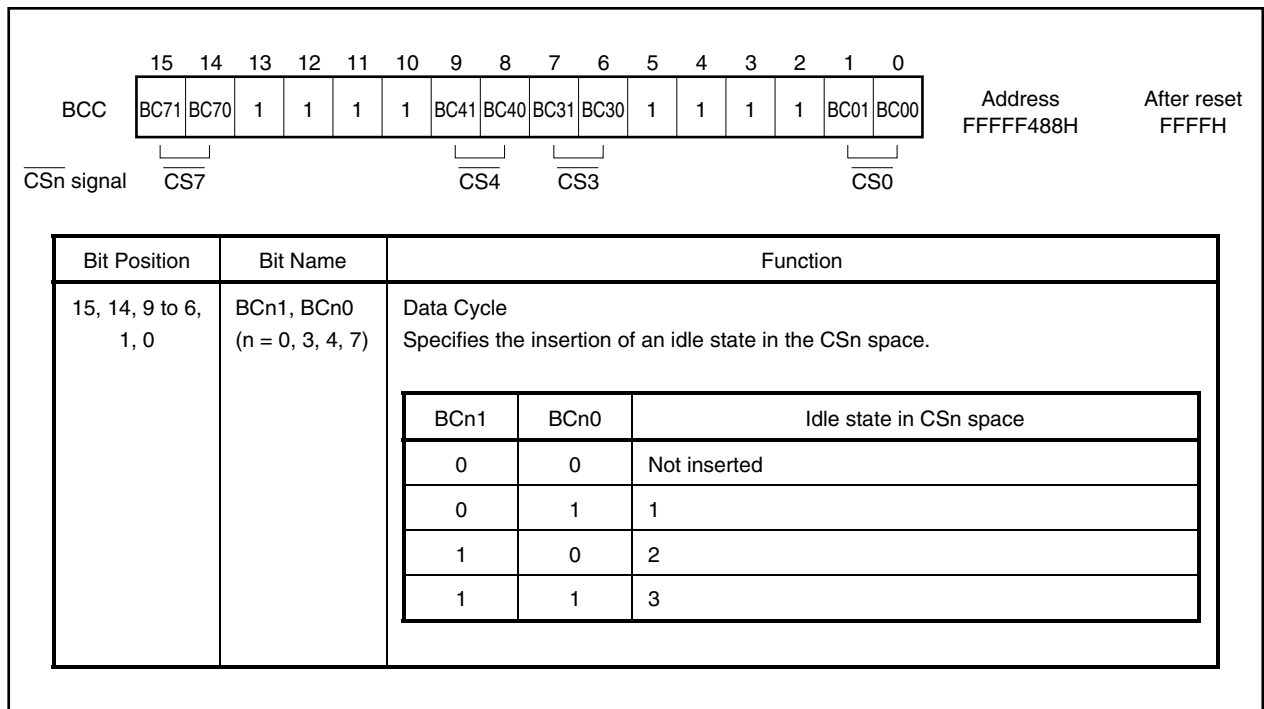
The idle state insertion setting can be specified by program using the bus cycle control register (BCC). Immediately after the system reset, idle state insertion is automatically programmed for all memory blocks. For the timing when an idle state is inserted, refer to the access timings of each memory in Chapter 5.

(1) Bus cycle control register (BCC)

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

Be sure to set bits 13 to 10 and 5 to 2 to 1. If they are set to 0, the operation is not guaranteed.

- Cautions**
1. The internal RAM area and on-chip peripheral I/O area are not subject to idle state insertion.
 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 external memory areas whose initialization settings are complete.



4.8 Bus Hold Function

4.8.1 Function outline

If pins PCM2 and PCM3 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 mastership request from another bus master, the external address/data bus and each strobe pin are shifted to high impedance and then released (bus hold state). If the $\overline{\text{HLDRQ}}$ pin becomes inactive (high level) and the bus mastership request is canceled, driving of these pins begins again.

- ★ During the bus hold period, the internal operations of the V850E/MA2 continue until the external memory or a peripheral I/O register is accessed.

The bus hold state can be known by the $\overline{\text{HLDAK}}$ pin becoming active (low level). The period from when the $\overline{\text{HLDRQ}}$ pin becomes active (low level) to when the $\overline{\text{HLDAK}}$ pin becomes active (low level) is at least 2 clocks.

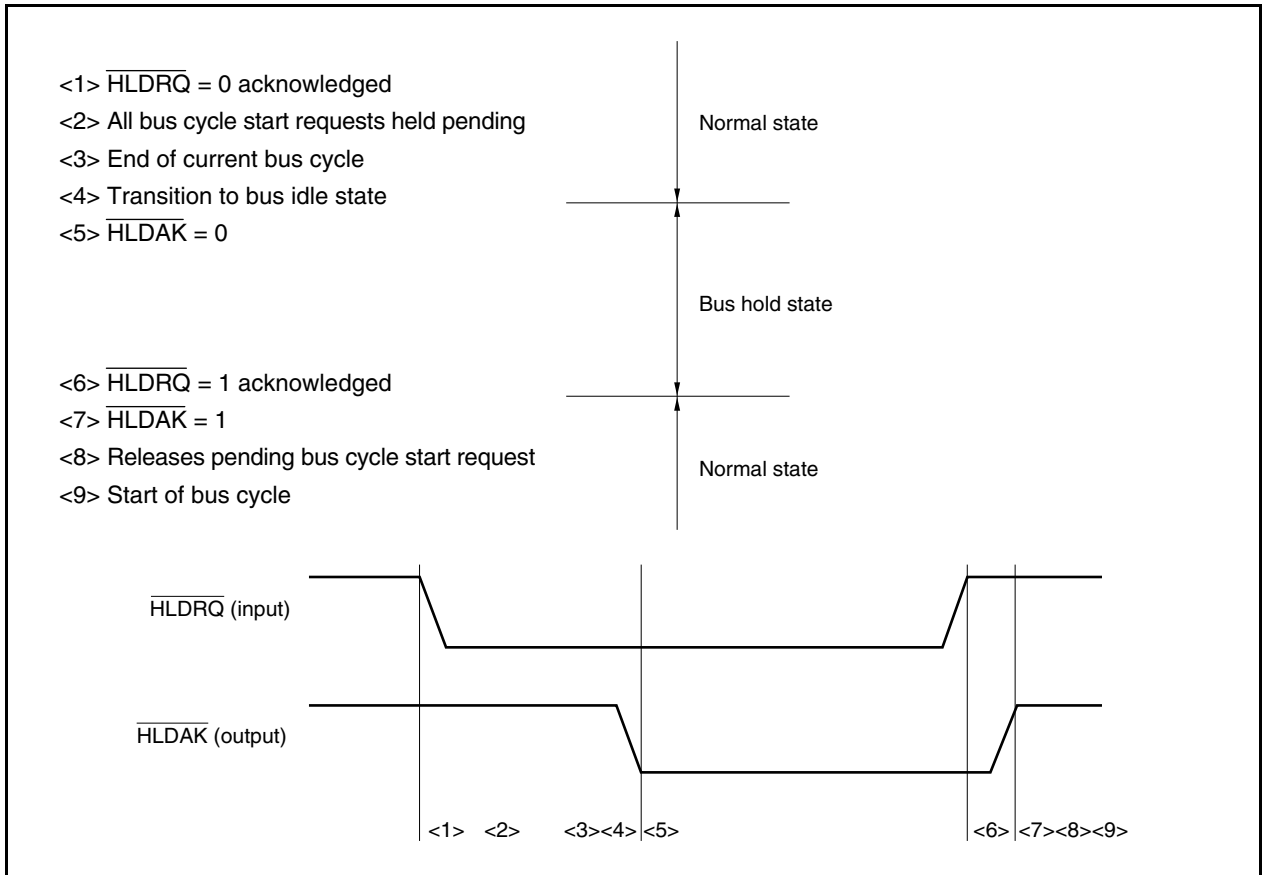
In a multiprocessor configuration, etc., a system with multiple bus masters can be configured.

State	Data Bus Width	Access Type	Timing in Which Bus Hold Request Cannot Be Acknowledged
CPU bus lock	16 bits	Word access for even address	Between first and second accesses
		Word access for odd address	Between first and second accesses
			Between second and third accesses
	Halfword access for odd address	Between first and second accesses	
	8 bits	Word access	Between first and second accesses
			Between second and third accesses
			Between third and fourth accesses
Halfword access		Between first and second accesses	
Read modify write access of bit manipulation instruction	–	–	Between read access and write access

- Cautions**
1. When an external bus master accesses SDRAM during a bus hold state, make sure that the external bus master executes the all bank precharge command.
The CPU always executes the all bank precharge command to release a bus hold state. In a bus hold state, do not allow an external bus master to change the SDRAM command register value.
 2. The $\overline{\text{HLDRQ}}$ function is invalid during a reset period. The $\overline{\text{HLDAK}}$ pin becomes active either immediately after or after the insertion of a 1-clock address cycle from when the $\overline{\text{RESET}}$ pin is set to inactive following the simultaneous activation of the $\overline{\text{RESET}}$ and $\overline{\text{HLDRQ}}$ pins.
When a bus master other than the V850E/MA2 is externally connected, use the $\overline{\text{RESET}}$ signal for bus arbitration at power-on.

4.8.2 Bus hold procedure

The procedure of the bus hold function is illustrated below.



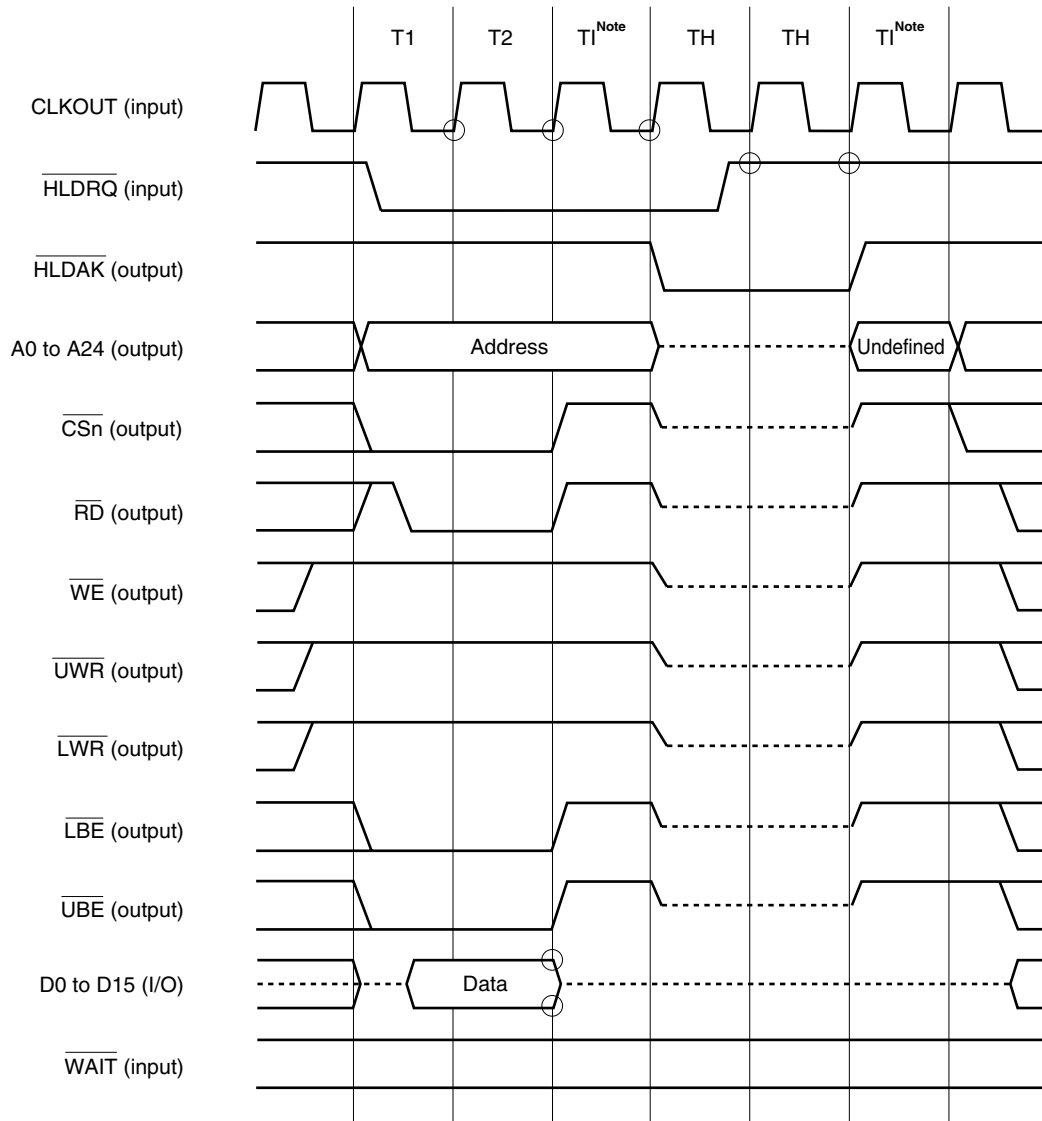
4.8.3 Operation in power save mode

In the software STOP or IDLE mode, the internal system clock is stopped. Consequently, the bus hold state is not acknowledged and set since the $\overline{\text{HLDARQ}}$ pin cannot be acknowledged even if it becomes active.

In the HALT mode, the $\overline{\text{HLDAR}}$ pin immediately becomes active when the $\overline{\text{HLDARQ}}$ pin becomes active, and the bus hold state is set. When the $\overline{\text{HLDARQ}}$ pin becomes inactive after that, the $\overline{\text{HLDAR}}$ pin also becomes inactive. As a result, the bus hold state is cleared and the HALT mode is set again.

4.8.4 Bus hold timing (SRAM)

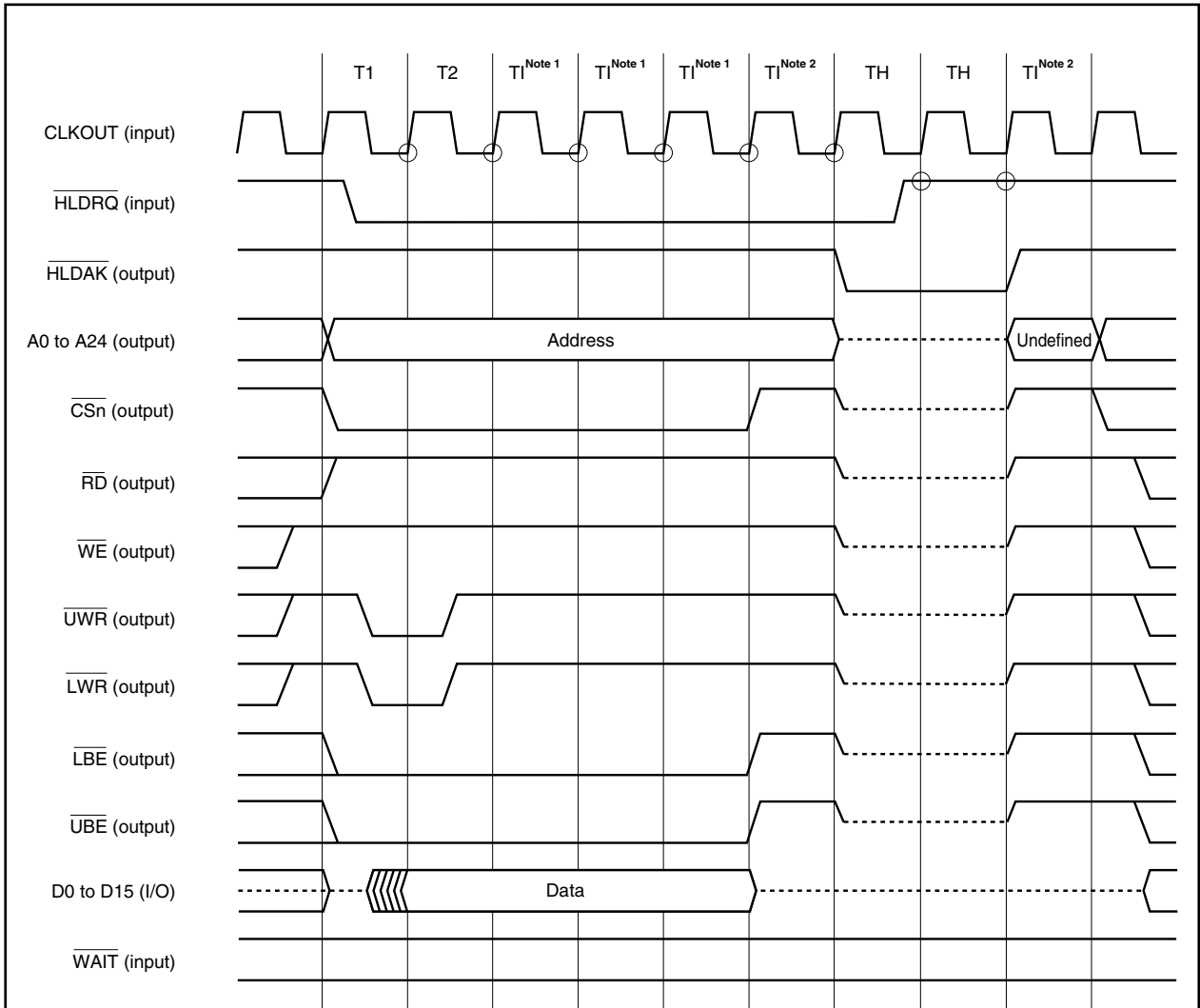
(1) SRAM (when read, no idle state inserted)



Note This idle state (TI) is independent of the BCC register setting.

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 0, 3, 4, 7

(2) SRAM (when written, three idle states inserted)

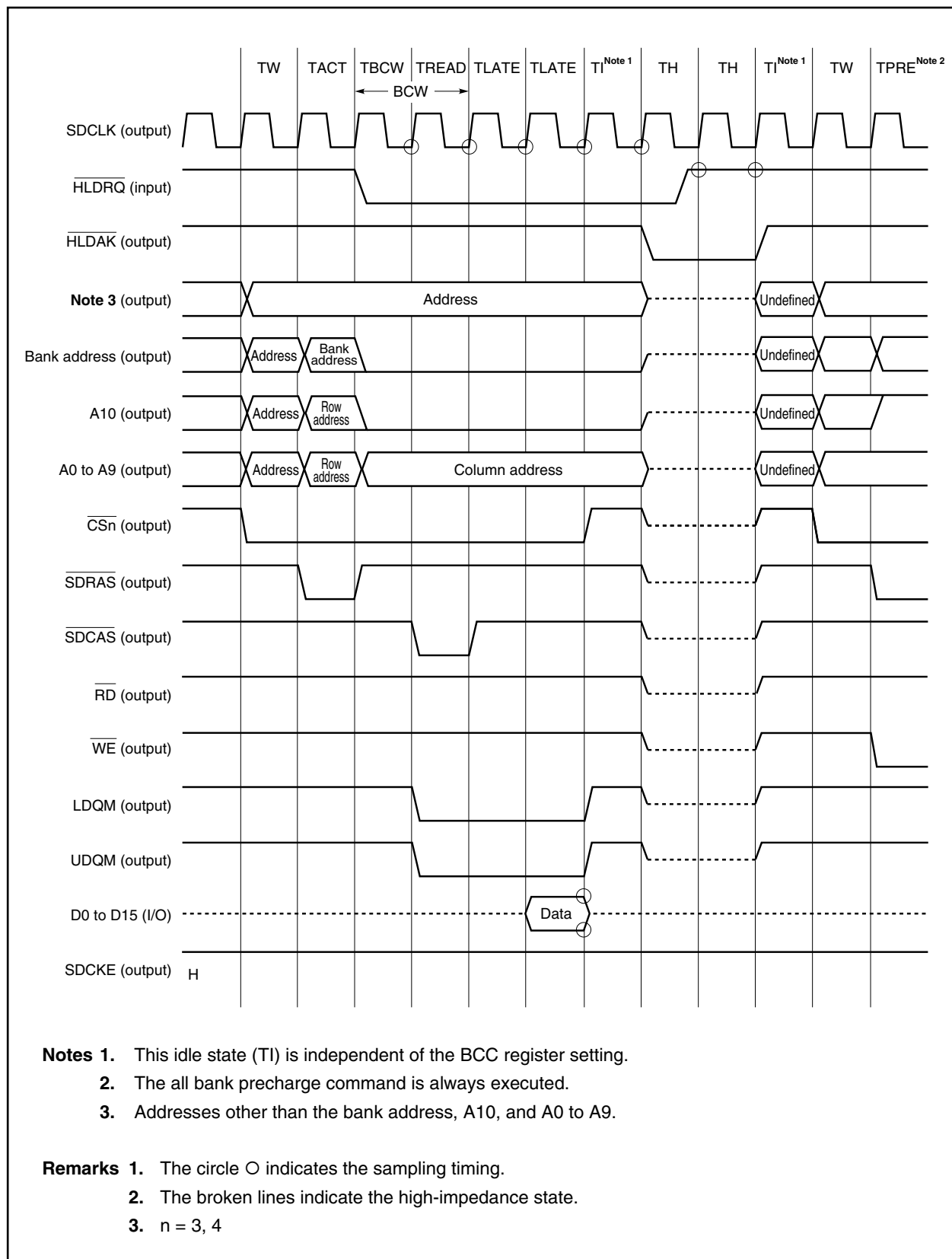


- Notes**
1. This idle state (TI) is inserted by means of a BCC register setting.
 2. This idle state (TI) is independent of the BCC register setting.

- Remarks**
1. The circle O indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 0, 3, 4, 7

4.8.5 Bus hold timing (SDRAM)

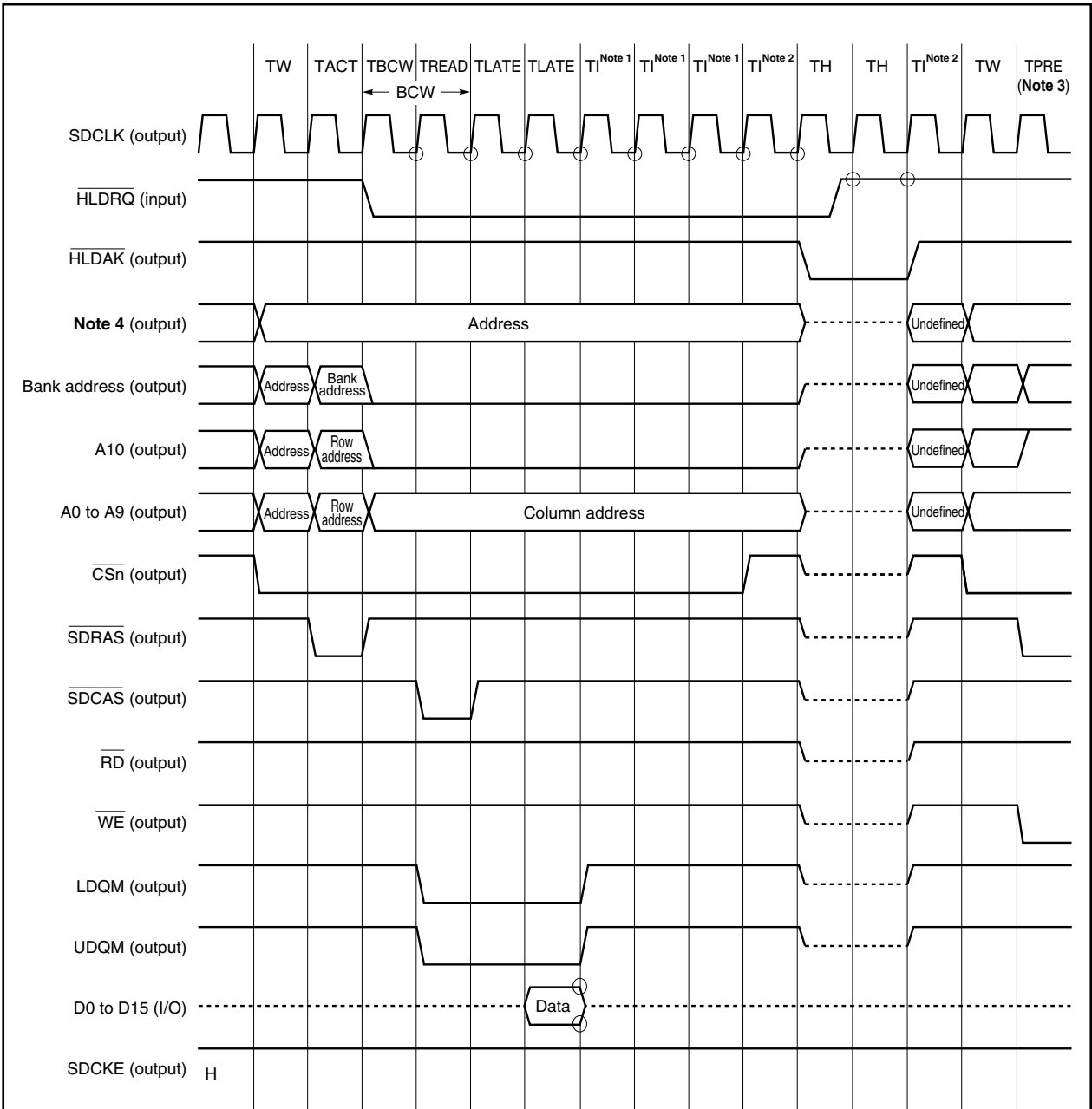
(1) SDRAM (when read, latency = 2, no idle state inserted)



- Notes**
1. This idle state (TI) is independent of the BCC register setting.
 2. The all bank precharge command is always executed.
 3. Addresses other than the bank address, A10, and A0 to A9.

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 3, 4

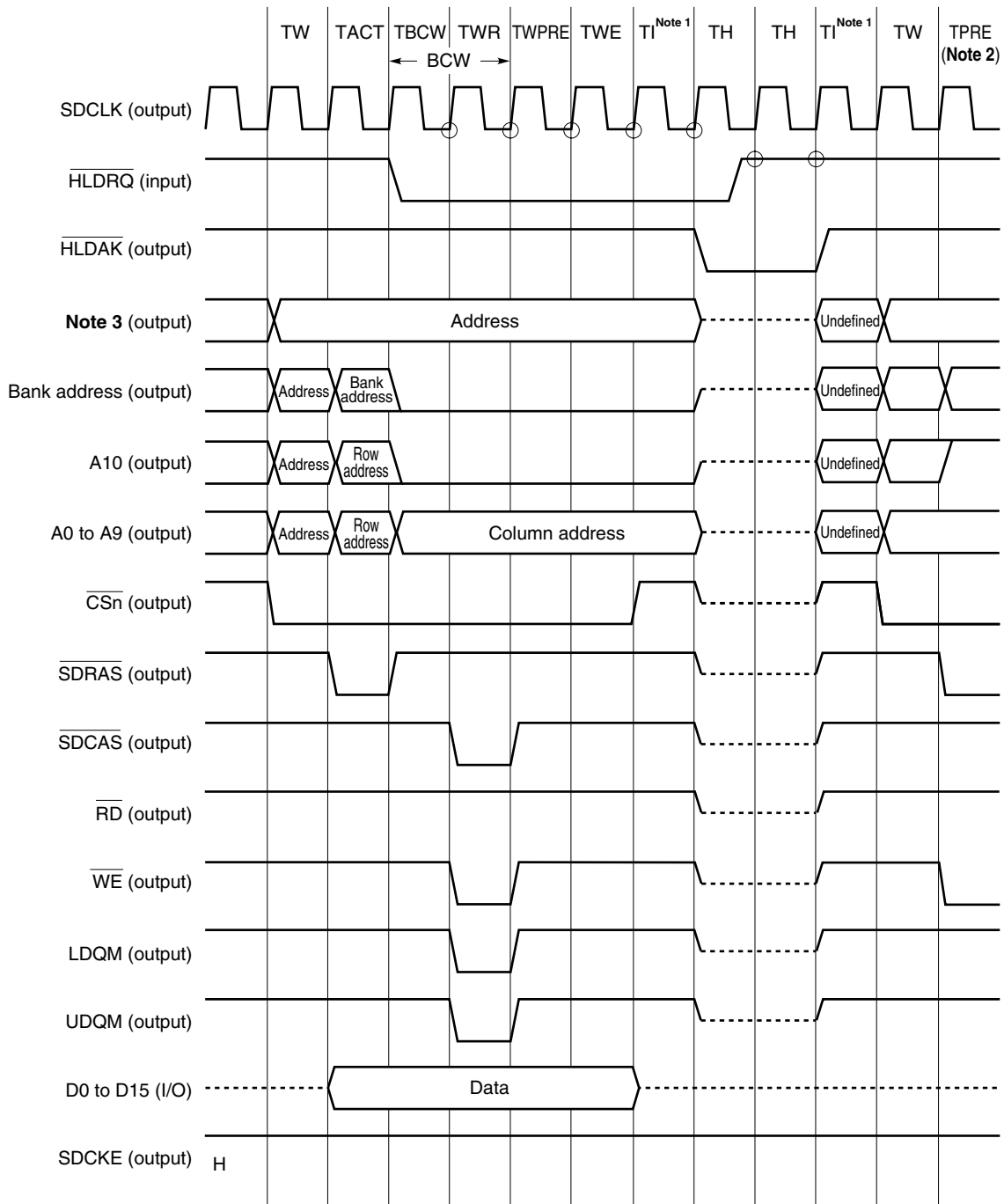
(2) SDRAM (when read, latency = 2, three idle states inserted)



- Notes**
1. This idle state (TI) is inserted by means of a BCC register setting.
 2. This idle state (TI) is independent of the BCC register setting.
 3. The all bank precharge command is always executed.
 4. Addresses other than the bank address, A10, and A0 to A9.

- Remarks**
1. The circle O indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 3, 4

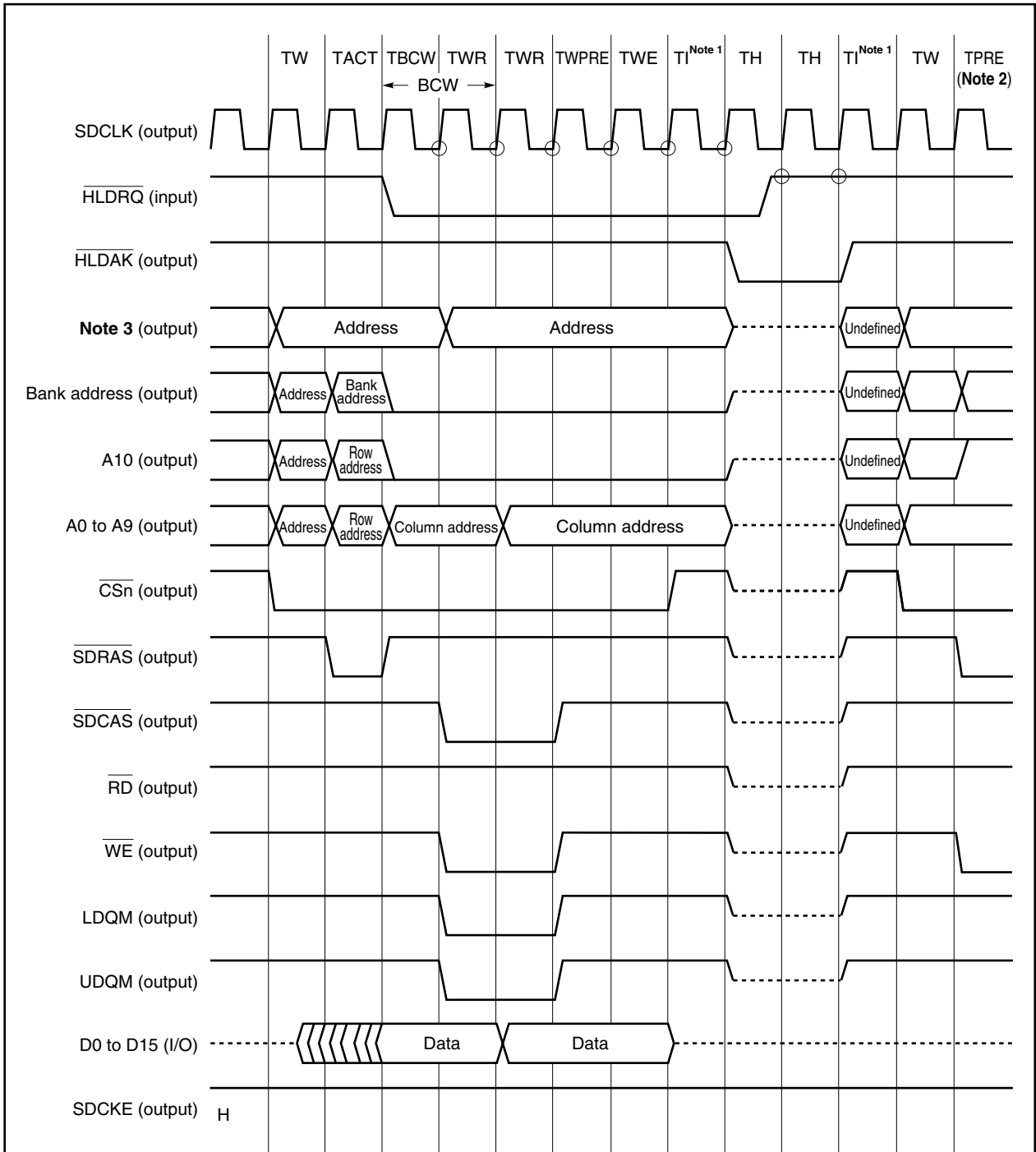
(3) SDRAM (when written)



- Notes**
1. This idle state (TI) is independent of the BCC register setting.
 2. The all bank precharge command is always executed.
 3. Addresses other than the bank address, A10, and A0 to A9.

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 3, 4

(4) SDRAM (when written, when bus hold request acknowledged during on-page access)



- Notes**
1. This idle state (TI) is independent of the BCC register setting.
 2. The all bank precharge command is always executed.
 3. Addresses other than the bank address, A10, and A0 to A9.

- Remarks**
1. The circle O indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 3, 4

4.9 Bus Priority Order


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

In order of priority, bus hold is the highest, followed by the refresh cycle, DMA cycle, operand data access, and instruction fetch, in that order.

An instruction fetch may be inserted between a read access and write access during a read modify write access.

Also, an instruction fetch may be inserted between bus accesses when a CPU bus clock is used.

Table 4-2. Bus Priority Order

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

4.10 Boundary Operation Conditions

★ 4.10.1 Program space

- (1) Branching to the on-chip peripheral I/O area or successive fetches from the internal RAM area to the on-chip peripheral I/O area are prohibited. If the above is performed (branching or successive fetch), an undefined data is fetched, 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 on-chip peripheral I/O area does not occur.

4.10.2 Data space

The V850E/MA2 is provided with an address misalign function.

Through this function, regardless of the data format (word data or halfword data), data can be allocated to all addresses. However, in the case of word data and halfword data, if the 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 LSB is 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 LSB is 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 lowest 2 bits are 10, the halfword-length bus cycle will be generated 2 times.

CHAPTER 5 MEMORY ACCESS CONTROL FUNCTION

5.1 SRAM, External ROM, External I/O Interface

5.1.1 Features

- SRAM is accessed in a minimum of 2 states.
- Up to 7 states of programmable data waits can be inserted by setting the DWC0 and DWC1 registers.
- Data wait can be controlled via $\overline{\text{WAIT}}$ pin input.
- Up to 3 idle states can be inserted after a read/write cycle by setting the BCC register.
- Up to 3 address setup wait states can be inserted by setting the ASC register.

5.1.2 SRAM connection

Examples of connection to SRAM are shown below.

Figure 5-1. Examples of Connection to SRAM (1/2)

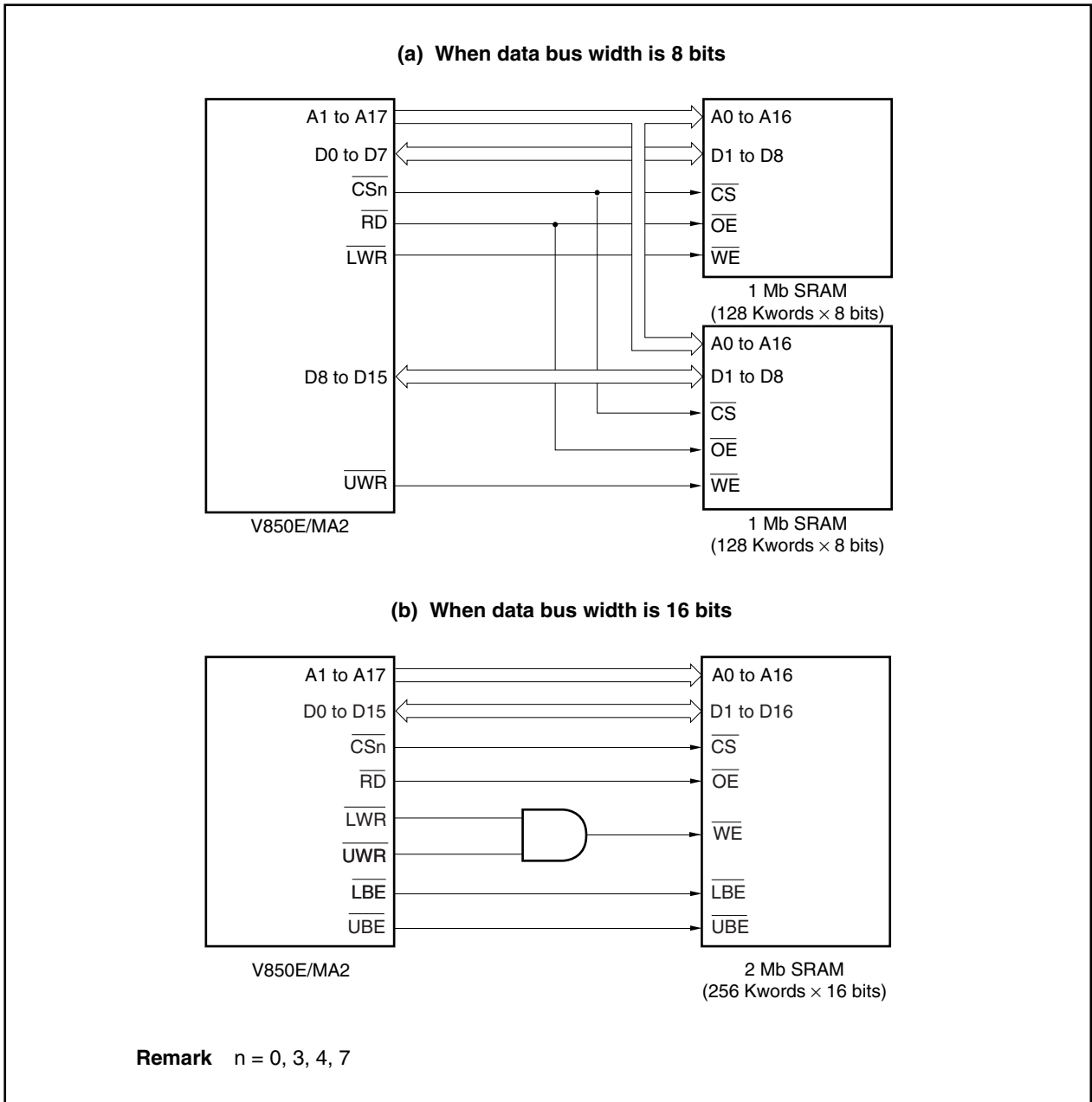
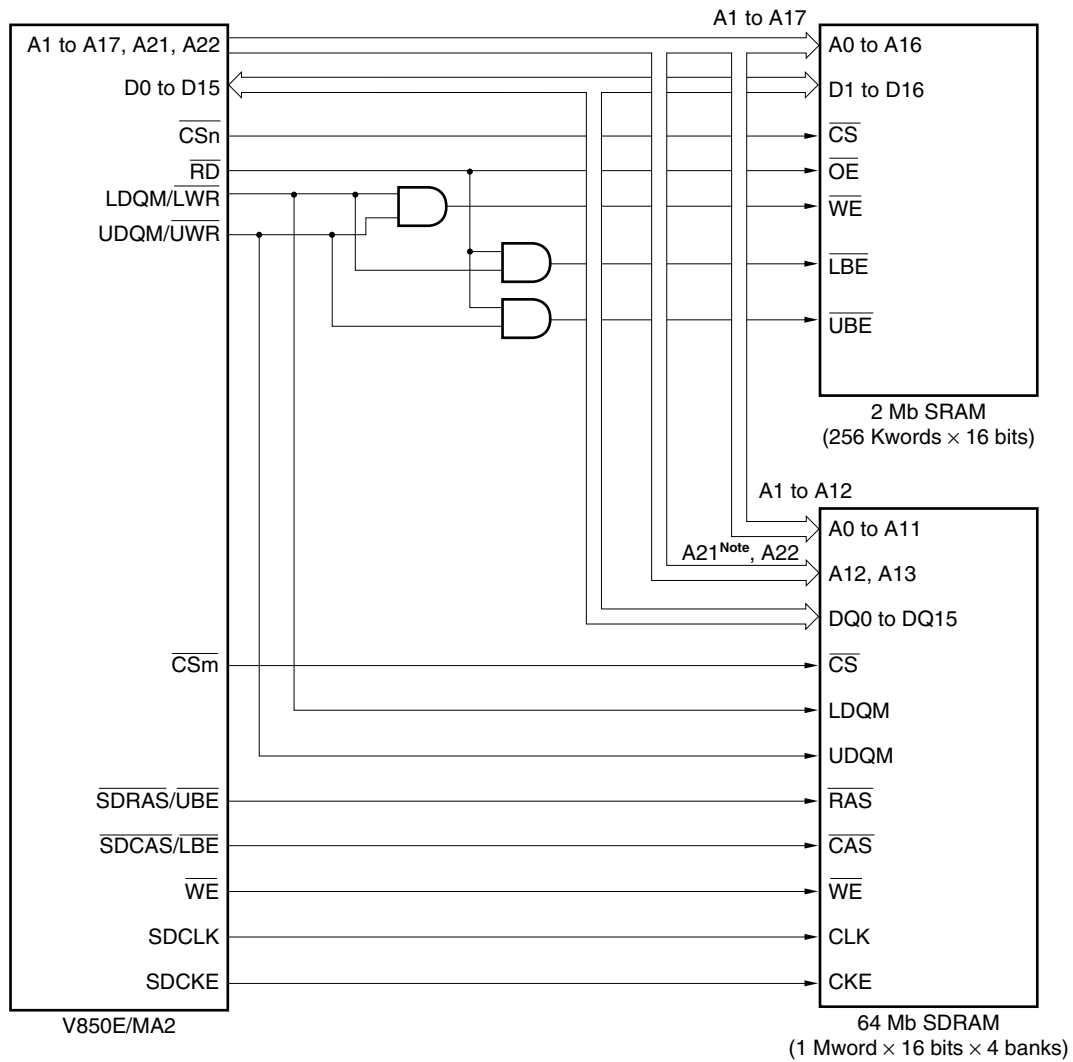


Figure 5-1. Examples of Connection to SRAM (2/2)

(c) Mixture of SRAM (256 Kwords × 16 bits) and SDRAM (1 Mword × 16 bits)



Note The address signals used depend on the SDRAM model.

Remark $n = 0, 3, 4, 7, m = 3, 4$ ($n \neq m$)

5.1.3 SRAM, external ROM, external I/O access

Figure 5-2. SRAM, External ROM, External I/O Access Timing (1/6)

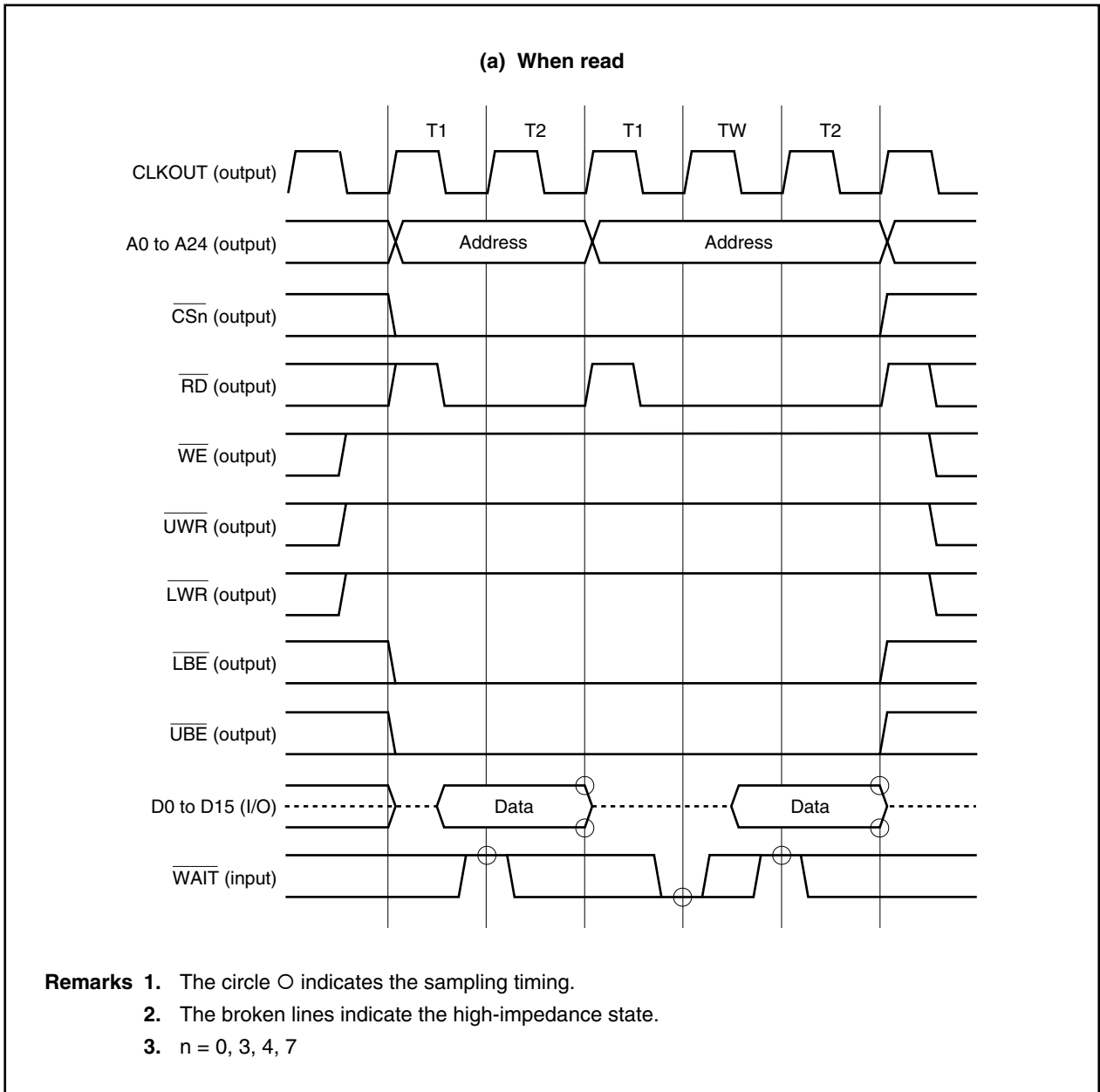
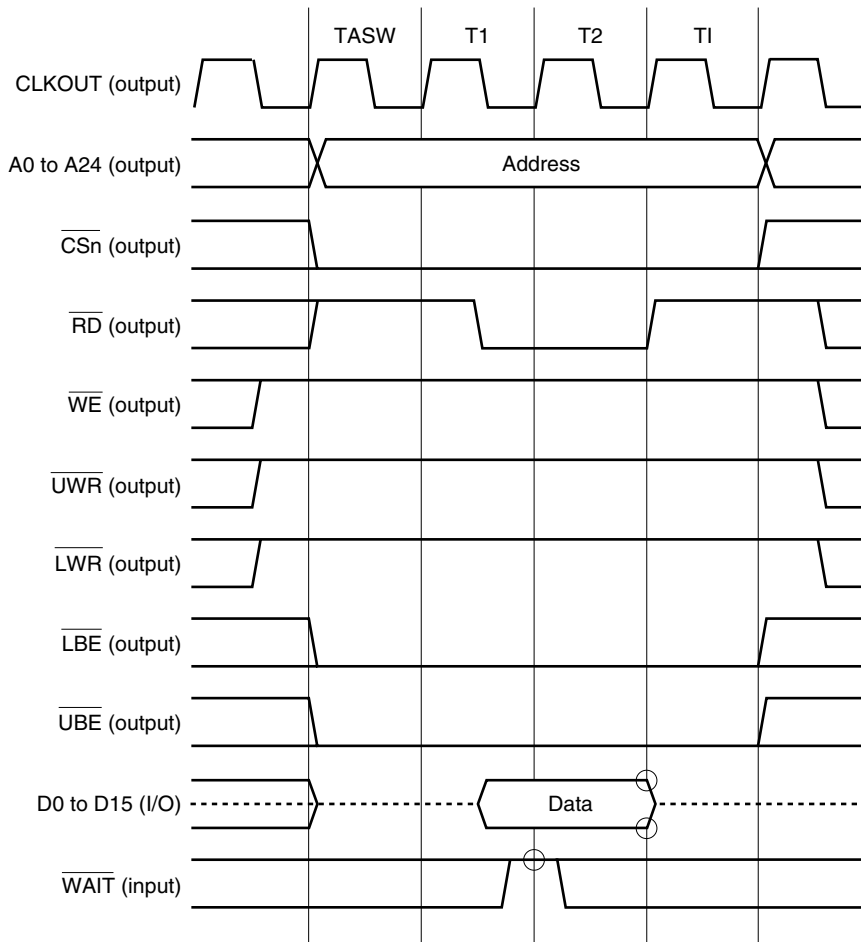


Figure 5-2. SRAM, External ROM, External I/O Access Timing (2/6)

(b) When read (address setup wait, idle state insertion)



- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. $n = 0, 3, 4, 7$

Figure 5-2. SRAM, External ROM, External I/O Access Timing (3/6)

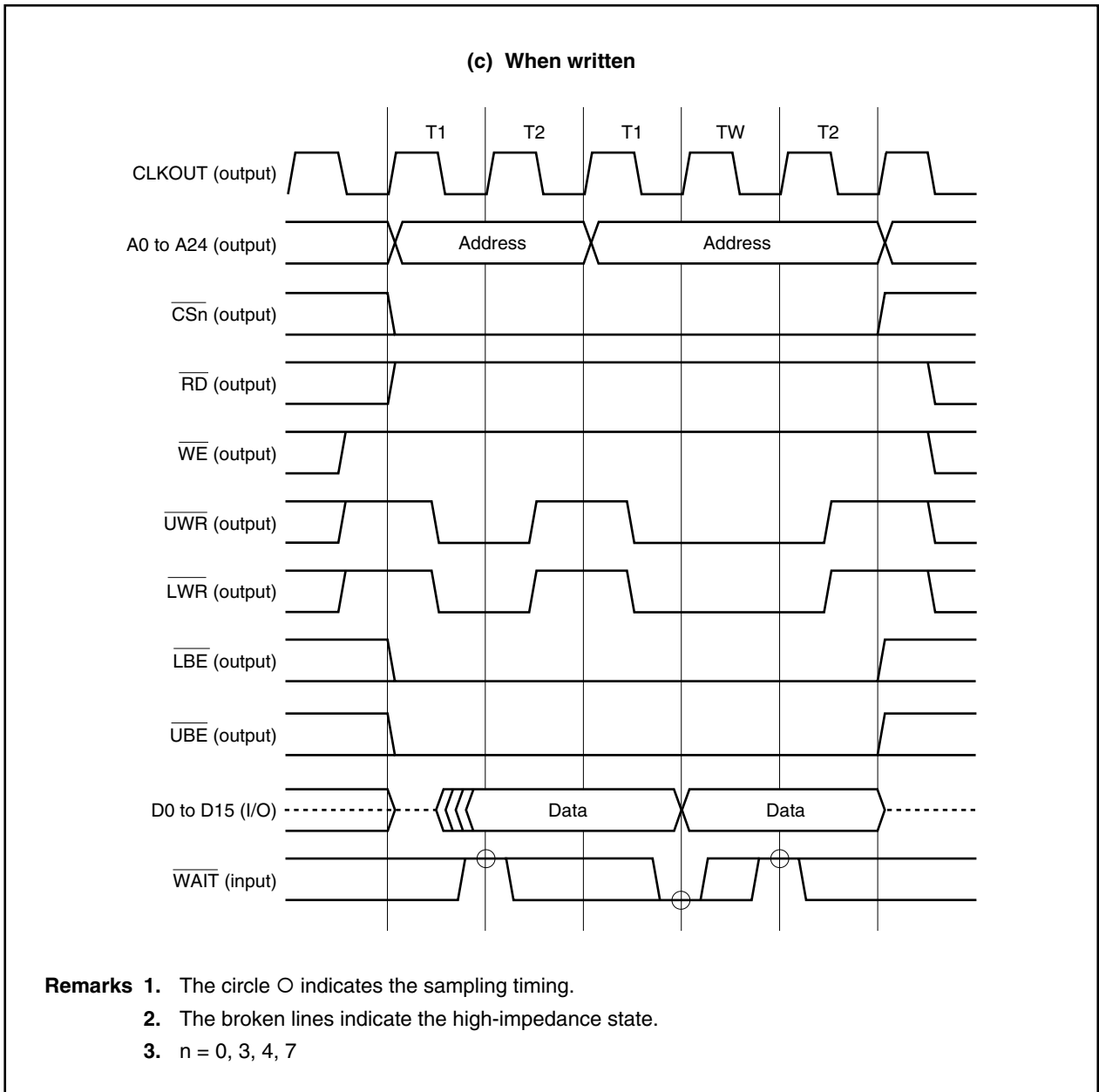
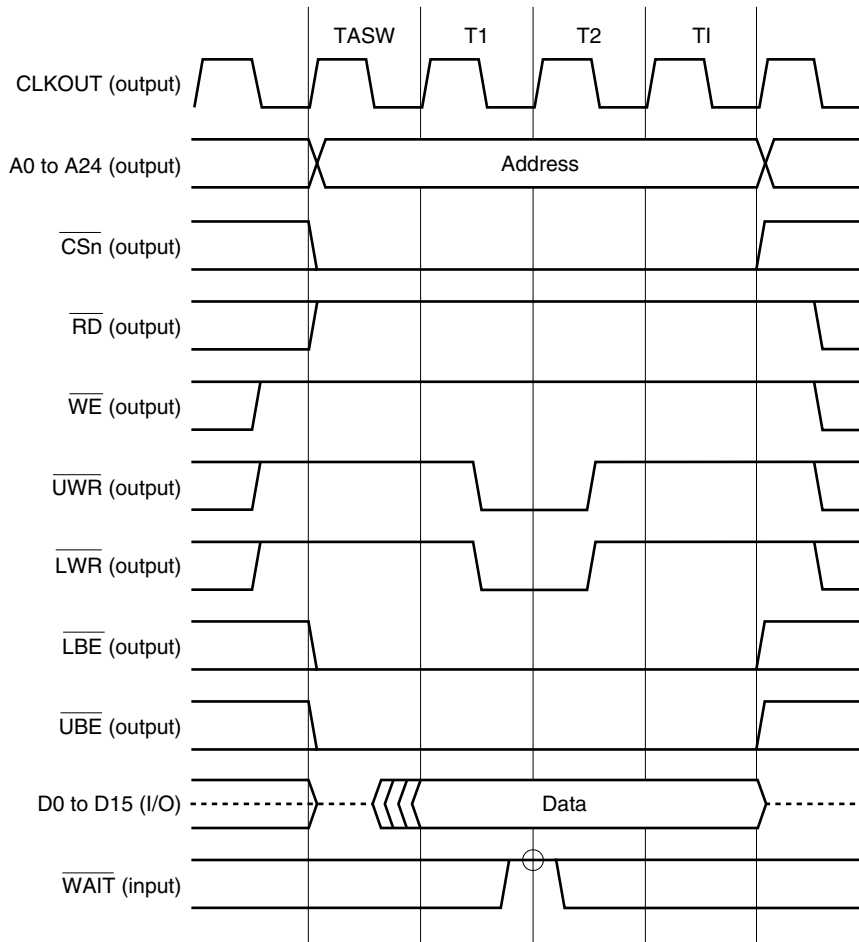


Figure 5-2. SRAM, External ROM, External I/O Access Timing (4/6)

(d) When written (address setup wait, idle state insertion)



- Remarks**
1. The circle ⊕ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 0, 3, 4, 7

Figure 5-2. SRAM, External ROM, External I/O Access Timing (5/6)

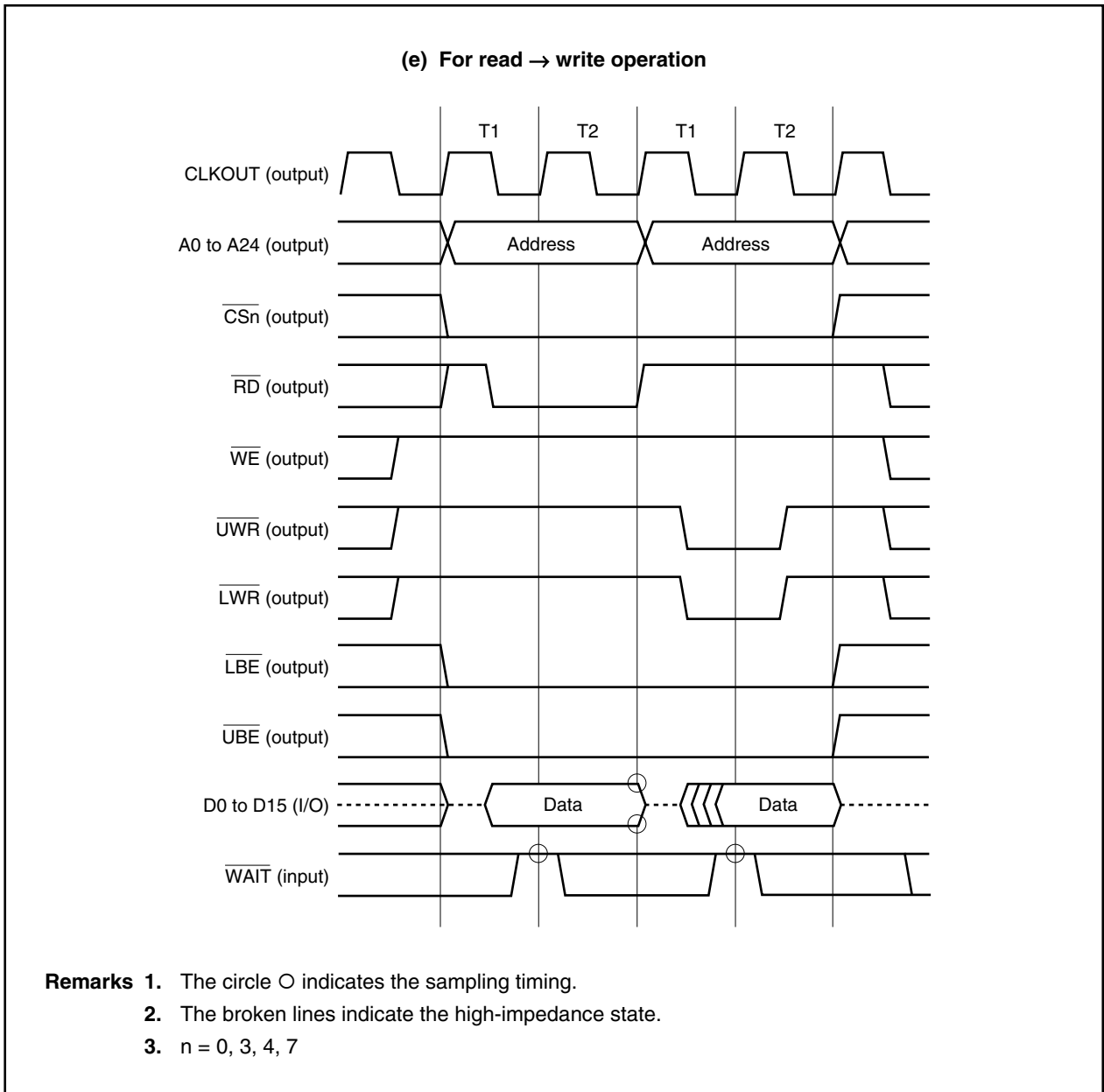
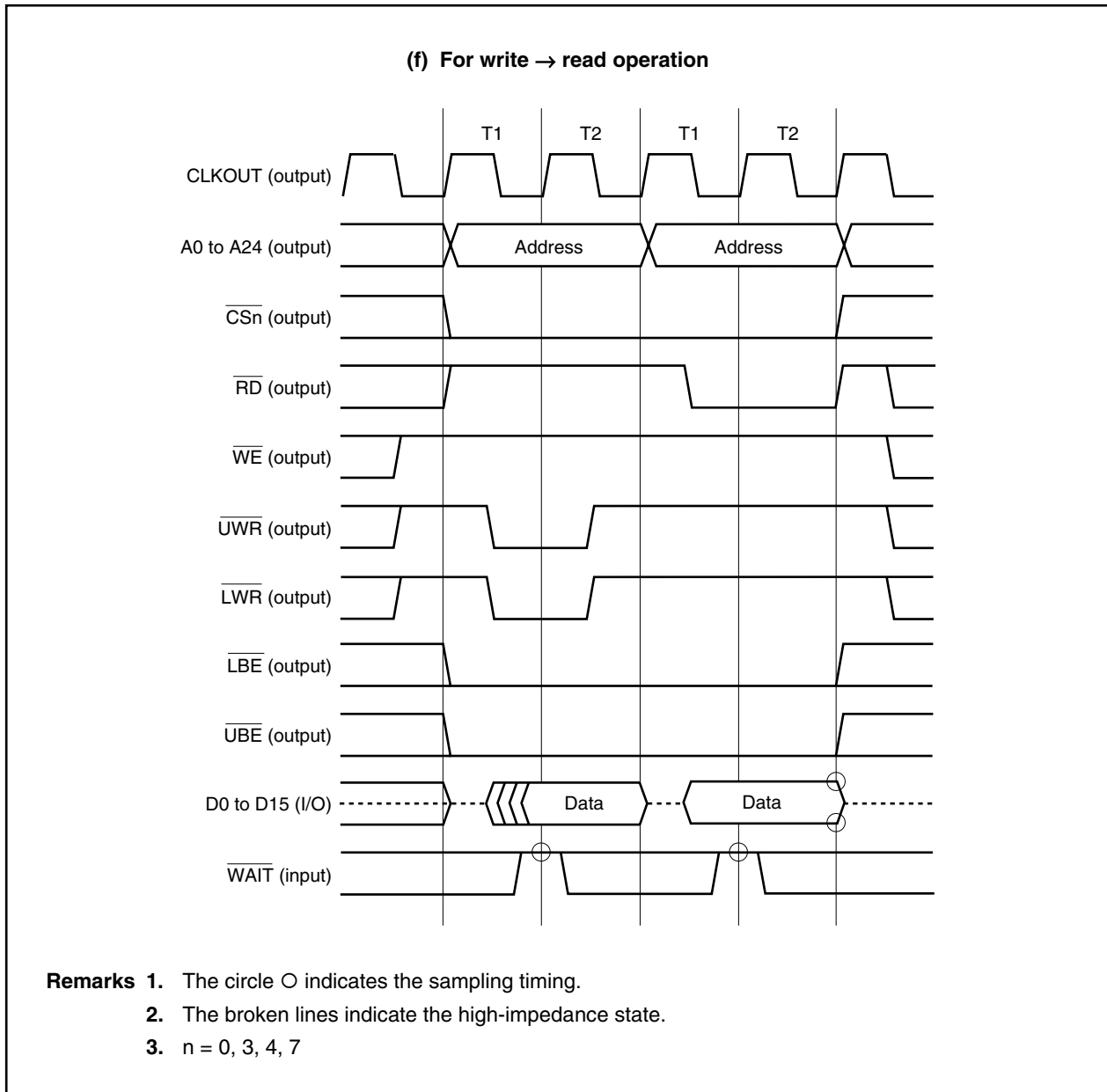


Figure 5-2. SRAM, External ROM, External I/O Access Timing (6/6)



5.2 Page ROM Controller (ROMC)

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

Addresses are compared with the immediately preceding bus cycle and wait control for normal access (off-page) and page access (on-page) is executed. This controller can handle page widths from 8 to 128 bytes.

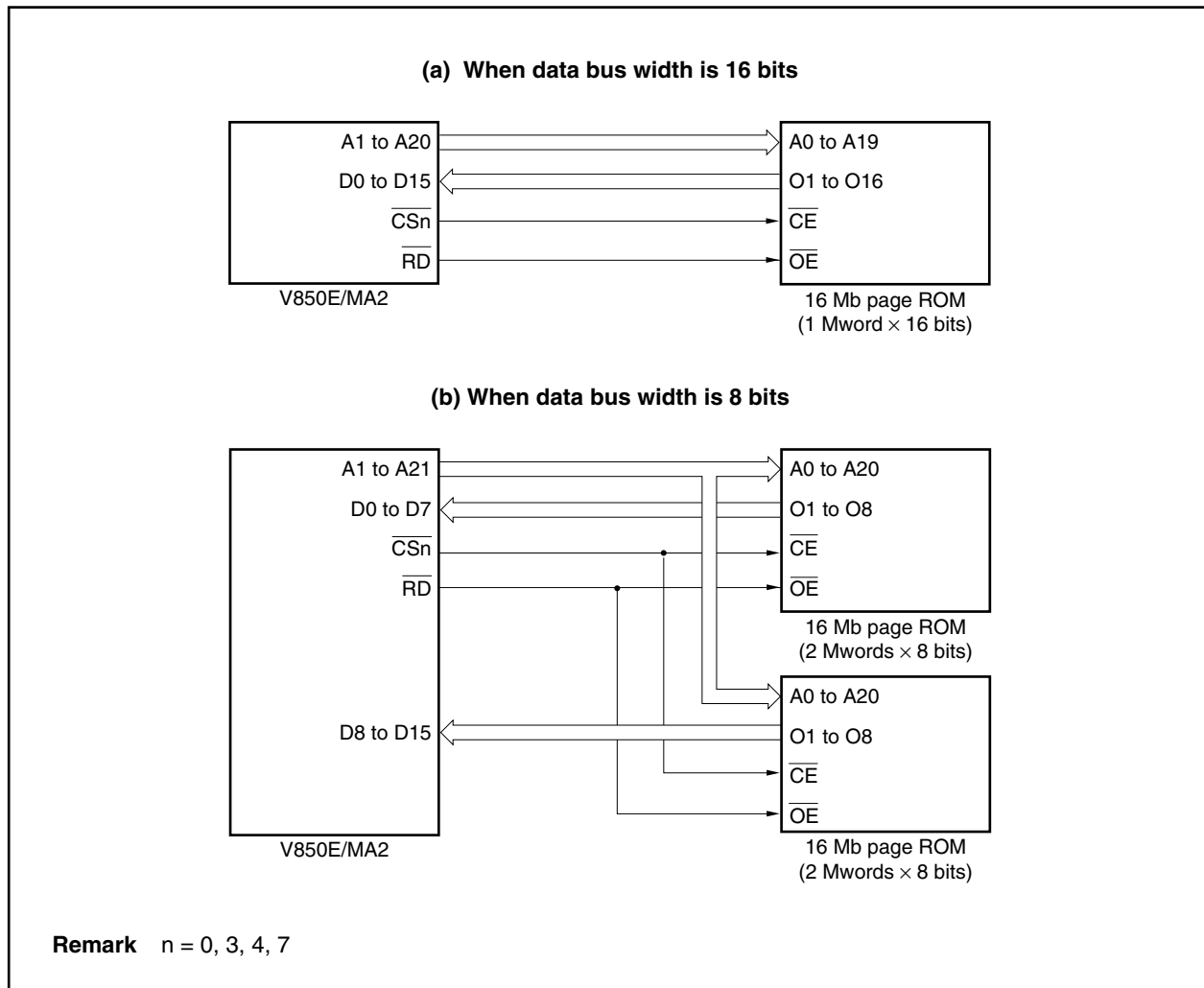
5.2.1 Features

- Direct connection to 8-bit/16-bit page ROM supported
- For 16-bit bus width: 4/8/16/32/64-word page access supported
For 8-bit bus width: 8/16/32/64/128-word page access supported
- Page ROM is accessed in a minimum of 2 states.
- On-page judgment function
- Addresses to be compared can be changed by setting the PRC register.
- Up to 7 states of programmable data waits can be inserted during an on-page cycle by setting the PRC register.
- Up to 7 states of programmable data waits can be inserted during an off-page cycle by setting the DWC0 and DWC1 registers.
- Waits can be controlled via $\overline{\text{WAIT}}$ pin input.

5.2.2 Page ROM connection

Examples of connection to page ROM are shown below.

Figure 5-3. Examples of Connection to Page ROM



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.

Through the page ROM configuration register (PRC), according to the configuration of the connected page ROM and the number of continuously readable bits, one of the addresses (A3 to A6) is set as the masking address (no comparison is made).

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

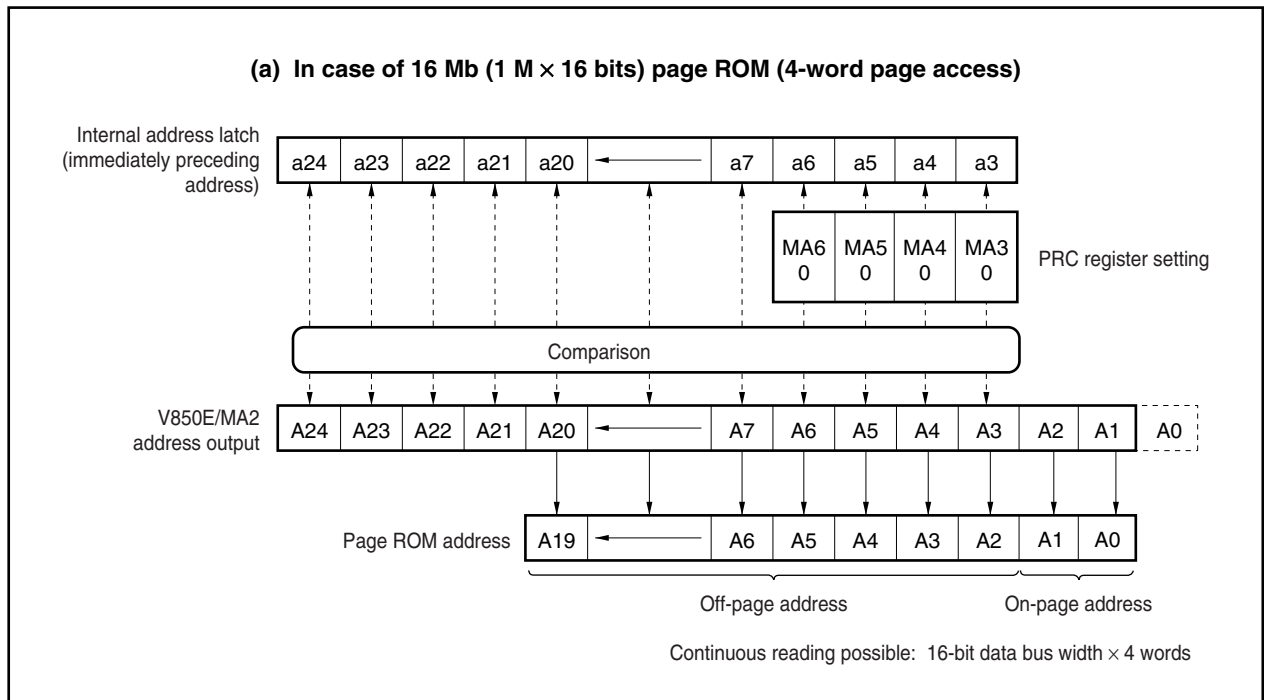
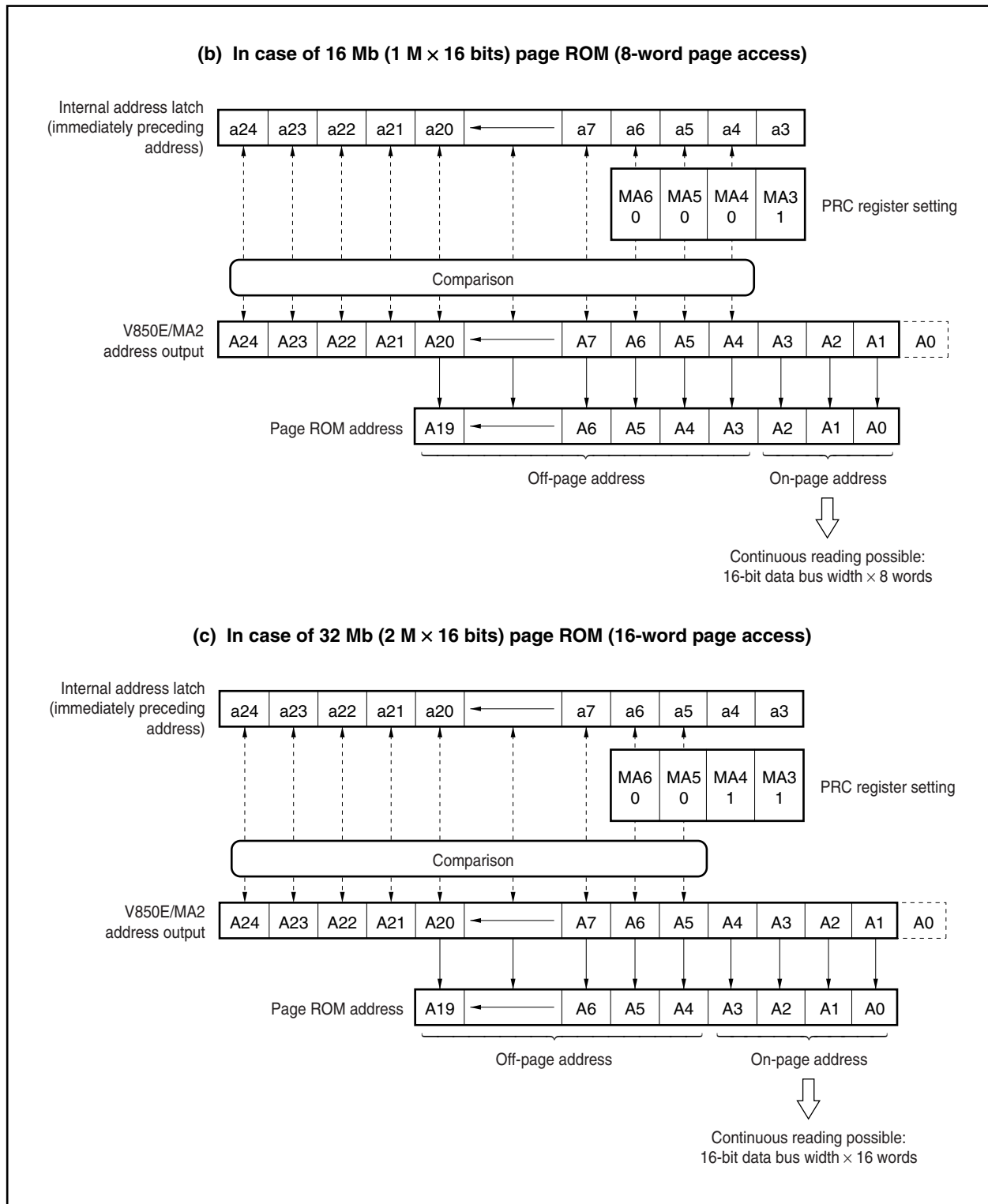


Figure 5-4. On-Page/Off-Page Judgment During Page ROM Connection (2/2)



5.2.4 Page ROM configuration register (PRC)

This register specifies whether page ROM cycle on-page access is enabled or disabled. If on-page access is enabled, the masking address (no comparison is made) out of the addresses (A3 to A6) 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 16-bit units.

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 external memory areas whose initialization settings are complete.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
PRC	0	PRW2	PRW1	PRW0	0	0	0	0	0	0	0	0	MA6	MA5	MA4	MA3	Address FFFFF49AH	After reset 7000H

Bit Position	Bit Name	Function																																				
14 to 12	PRW2 to PRW0	Page-ROM On-page Wait Control Sets the number of waits corresponding to the internal system clock. The number of waits set by these bits is inserted only for on-page access. For off-page access, the waits set by registers DWC0 and DWC1 are inserted.																																				
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">PRW2</th> <th style="width: 10%;">PRW1</th> <th style="width: 10%;">PRW0</th> <th style="width: 70%;">Number of inserted wait cycles</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">2</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">3</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">4</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">5</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">6</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">7</td></tr> </tbody> </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																																			
3 to 0	MA6 to MA3	Mask Address Each respective address (A6 to A3) corresponding to MA6 to MA3 is masked (masked by 1). The masked address is not subject to comparison during on/off-page judgment, and is set according to the number of continuously readable bits.																																				
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">MA6</th> <th style="width: 10%;">MA5</th> <th style="width: 10%;">MA4</th> <th style="width: 10%;">MA3</th> <th style="width: 70%;">Number of continuously readable bits</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">4 words × 16 bits (8 words × 8 bits)</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">8 words × 16 bits (16 words × 8 bits)</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">16 words × 16 bits (32 words × 8 bits)</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">32 words × 16 bits (64 words × 8 bits)</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">64 words × 16 bits (128 words × 8 bits)</td></tr> <tr><td colspan="4" style="text-align: center;">Other than above</td><td style="text-align: center;">Setting prohibited</td></tr> </tbody> </table>	MA6	MA5	MA4	MA3	Number of continuously readable bits	0	0	0	0	4 words × 16 bits (8 words × 8 bits)	0	0	0	1	8 words × 16 bits (16 words × 8 bits)	0	0	1	1	16 words × 16 bits (32 words × 8 bits)	0	1	1	1	32 words × 16 bits (64 words × 8 bits)	1	1	1	1	64 words × 16 bits (128 words × 8 bits)	Other than above				Setting prohibited	
MA6	MA5	MA4	MA3	Number of continuously readable bits																																		
0	0	0	0	4 words × 16 bits (8 words × 8 bits)																																		
0	0	0	1	8 words × 16 bits (16 words × 8 bits)																																		
0	0	1	1	16 words × 16 bits (32 words × 8 bits)																																		
0	1	1	1	32 words × 16 bits (64 words × 8 bits)																																		
1	1	1	1	64 words × 16 bits (128 words × 8 bits)																																		
Other than above				Setting prohibited																																		

5.2.5 Page ROM access

Figure 5-5. Page ROM Access Timing (1/4)

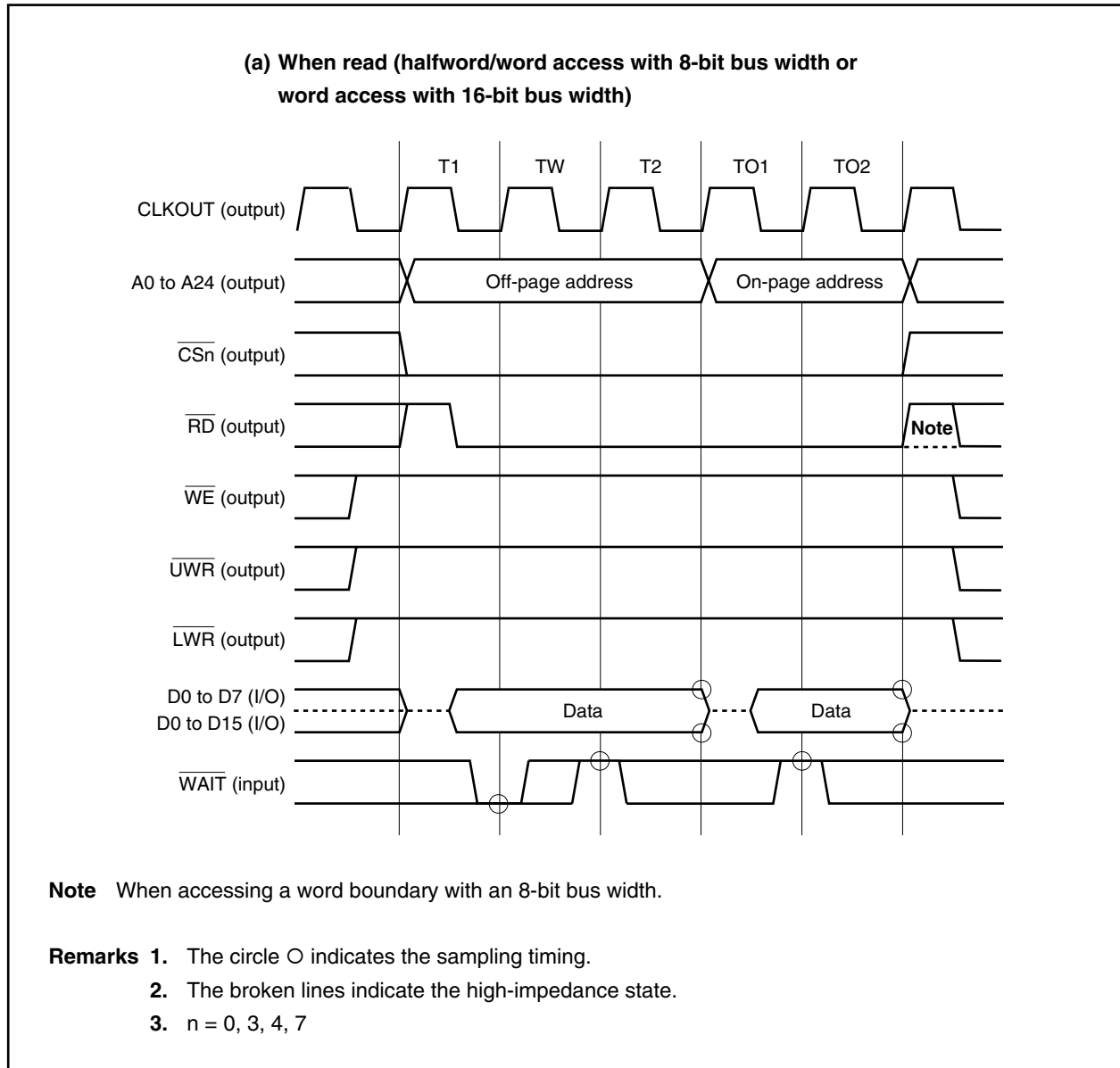


Figure 5-5. Page ROM Access Timing (2/4)

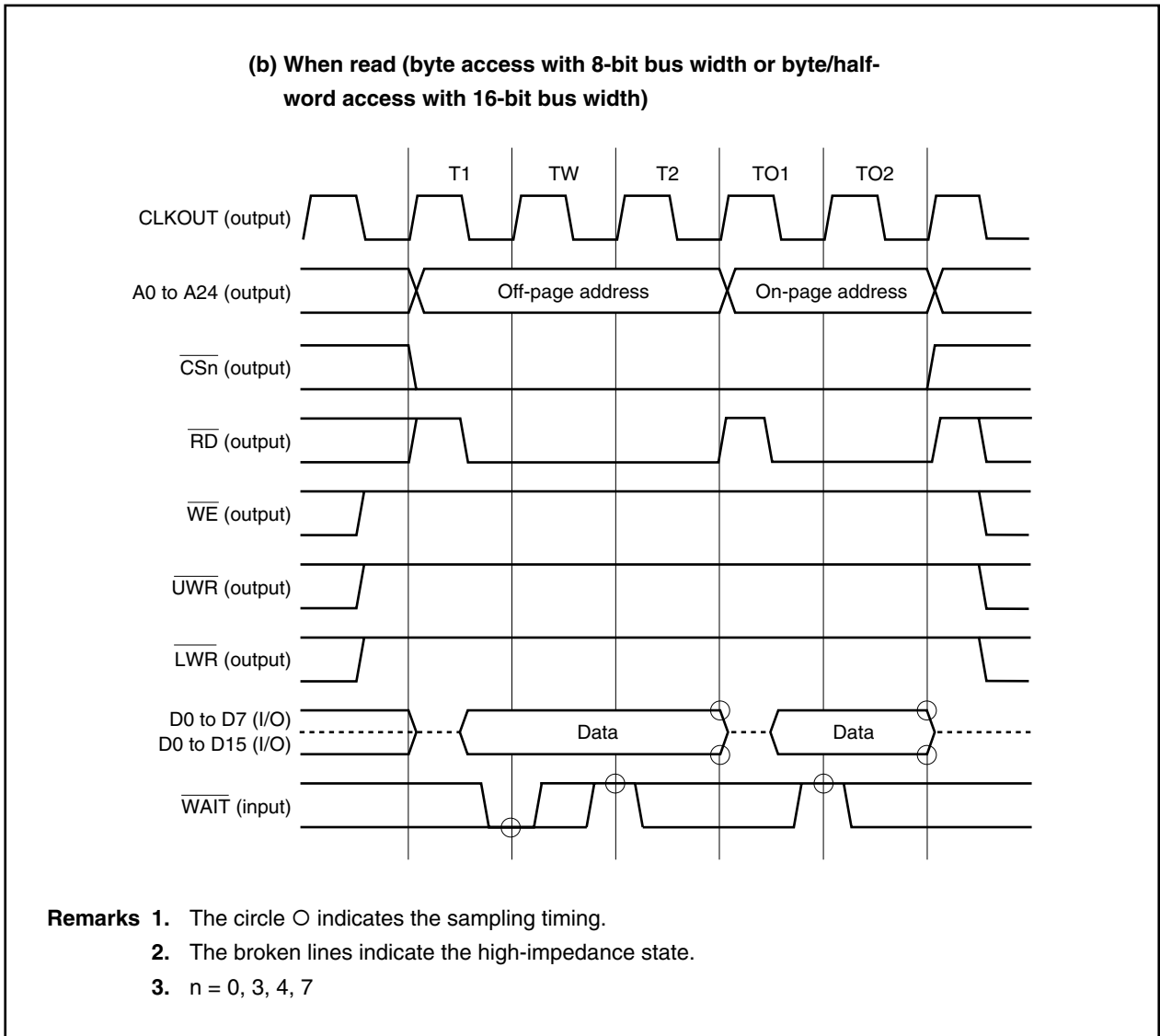


Figure 5-5. Page ROM Access Timing (3/4)

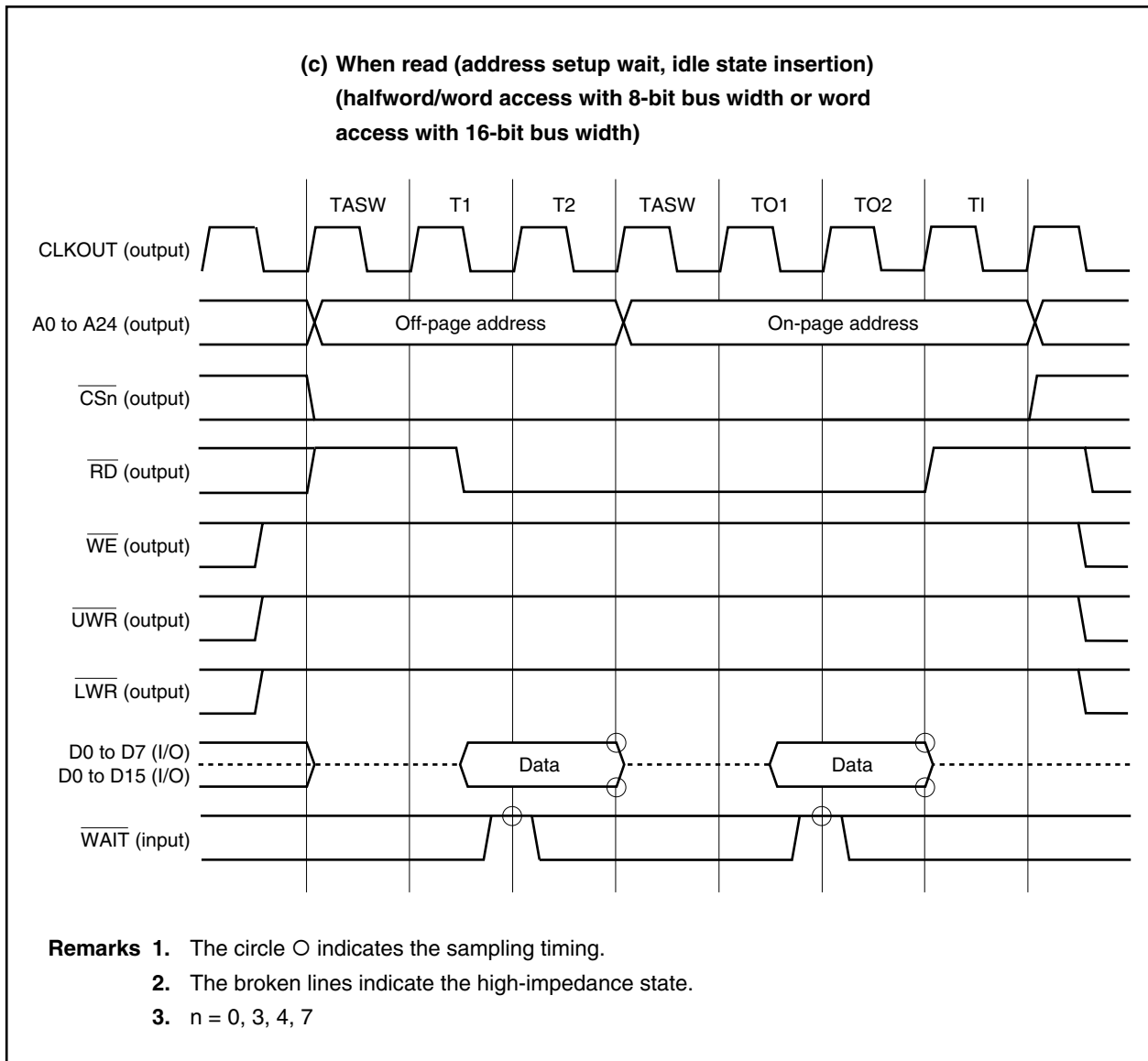
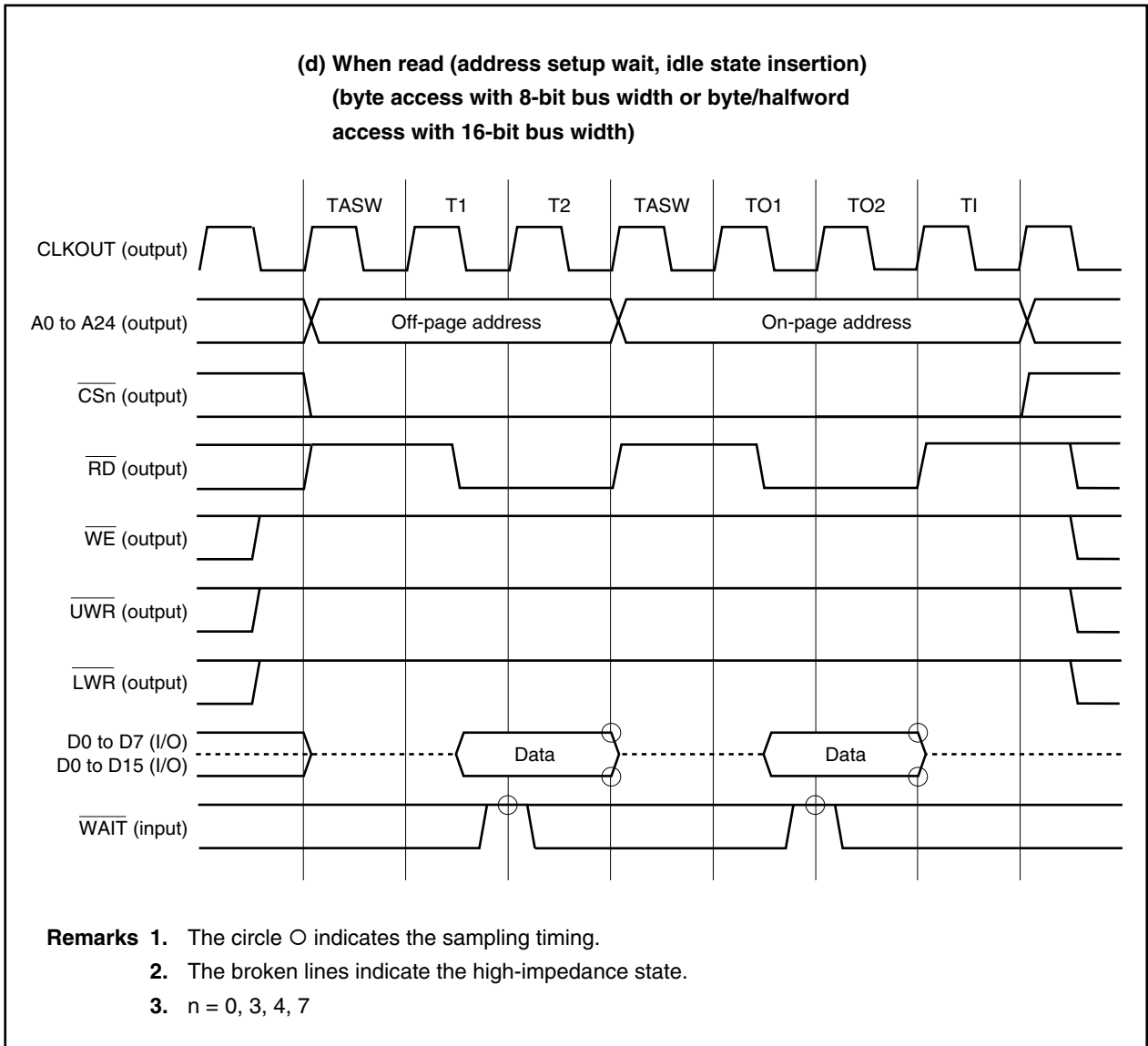


Figure 5-5. Page ROM Access Timing (4/4)



5.3 SDRAM Controller

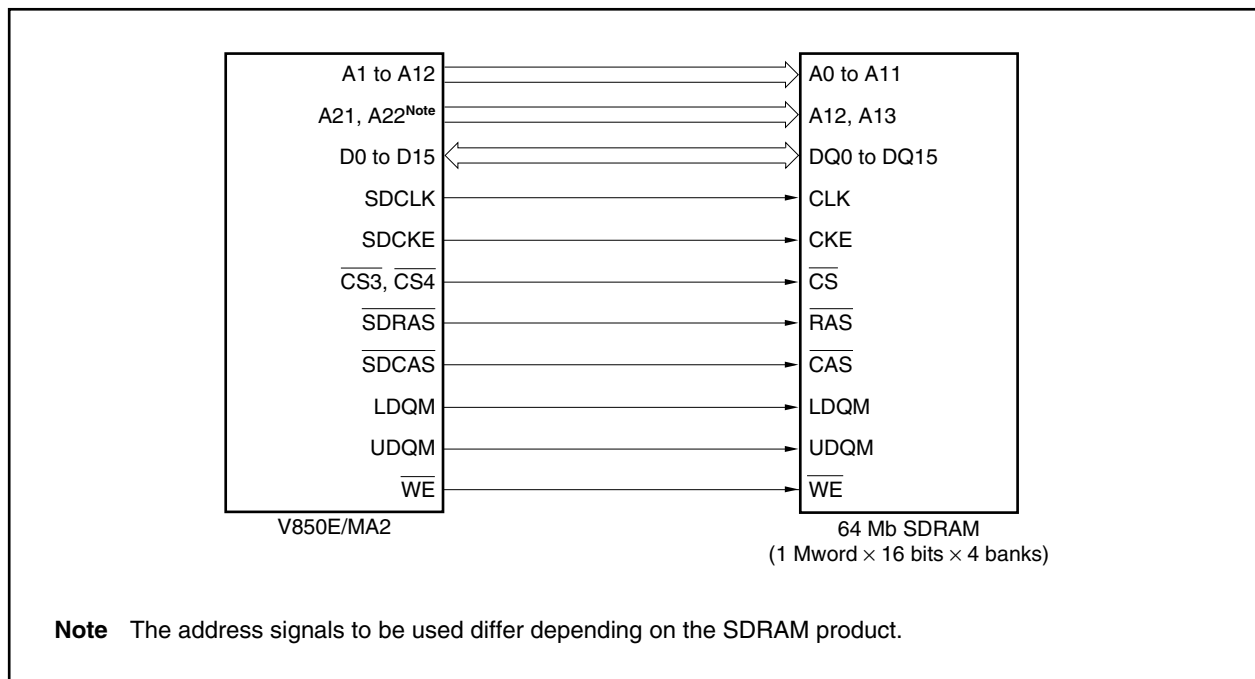
5.3.1 Features

- Burst length: 1
- Wrap type: Sequential
- CAS latency: 2 and 3 supported
- 4 types of SDRAM can be assigned to 4 memory blocks.
- Row and column address multiplex widths can be changed.
- Waits (0 to 3 waits) can be inserted between the bank active command and the read/write command.
- Supports CBR refresh and CBR self refresh.

5.3.2 SDRAM connection

An example of connection to SDRAM is shown below.

Figure 5-6. Example of Connection to SDRAM



5.3.3 Address multiplex function

Depending on the value of the SAW0n and SAW1n bits in SDRAM configuration register n (SCRn), the row address output in the SDRAM cycle is multiplexed as shown in Figure 5-7 (a) (n = 3, 4). Depending on the value of the SSO0n and SSO1n bits, the column address output in the SDRAM cycle is multiplexed as shown in Figure 5-7 (b) (n = 3, 4). In Figures 5-7 (a) and (b), a0 to a24 indicate the addresses output from the CPU, and A0 to A24 indicate the address pins of the V850E/MA2.

Figure 5-7. Row Address/Column Address Output (1/2)

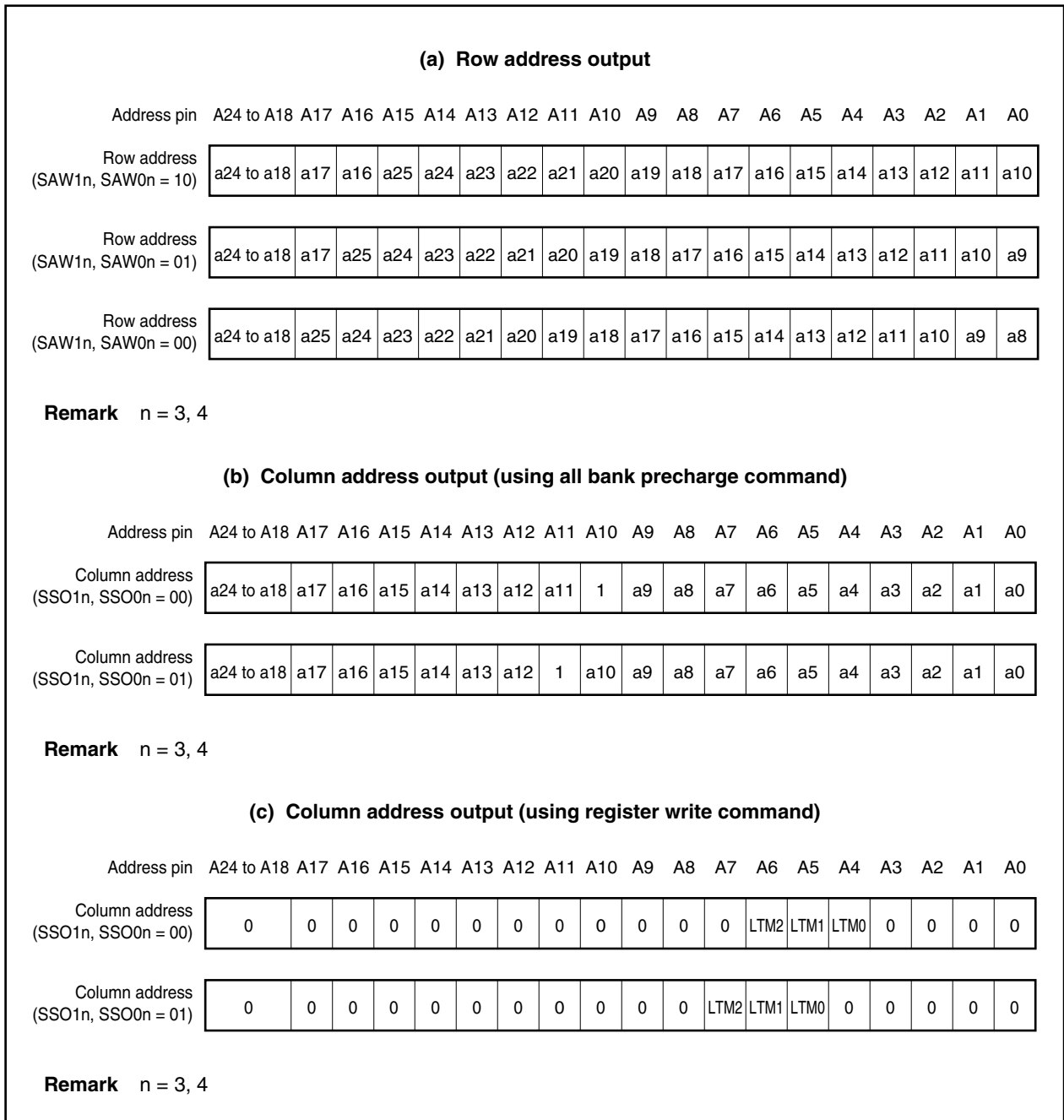


Figure 5-7. Row Address/Column Address Output (2/2)

(d) Column address output (using read/write command)

Address pin	A24 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Column address (SSO1n, SSO0n = 00)	a24 to a18	a17	a16	a15	a14	a13	a12	a11	0	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
Column address (SSO1n, SSO0n = 01)	a24 to a18	a17	a16	a15	a14	a13	a12	0	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0

Remark n = 3, 4

5.3.4 SDRAM configuration registers 3, 4 (SCR3, SCR4)

These registers specify the number of waits and the address multiplex width. SCR3 and SCR4 corresponds to $\overline{CS3}$ and $\overline{CS4}$. For example, to connect SDRAM to $\overline{CS3}$, set SCR3.

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

- Cautions 1.** The SDRAM read/write cycle is not generated prior to executing the power-on cycle. Access SDRAM after waiting 20 clocks following a program write to the SCR register. To write to the SCR register again following access to SDRAM, clear the MEn bit of the BCT0 and BCT1 registers to 0, and then set them to 1 again before performing access (n = 0 to 7).
- 2.** Do not execute continuous instructions to write to the SCR register. Be sure to insert another instruction between commands to write to the SCR register.

(1/2)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
SCR3	0	LTM23	LTM13	LTM03	0	0	0	0	BCW13	BCW03	SSO13	SSO03	RAW13	RAW03	SAW13	SAW03	Address FFFFFF4ACH	After reset 0000H
SCR4	0	LTM24	LTM14	LTM04	0	0	0	0	BCW14	BCW04	SSO14	SSO04	RAW14	RAW04	SAW14	SAW04	FFFFFF4B0H	0000H

Bit Position	Bit Name	Function																				
14 to 12	LTM2n to LTM0n (n = 3, 4)	Latency Sets the CAS latency value for reading. <table border="1" style="margin-top: 10px;"> <thead> <tr> <th>LTM2n</th> <th>LTM1n</th> <th>LTM0n</th> <th>Latency</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">×</td> <td style="text-align: center;">3</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">2</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">3</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">×</td> <td style="text-align: center;">×</td> <td style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table>	LTM2n	LTM1n	LTM0n	Latency	0	0	×	3	0	1	0	2	0	1	1	3	1	×	×	Setting prohibited
LTM2n	LTM1n	LTM0n	Latency																			
0	0	×	3																			
0	1	0	2																			
0	1	1	3																			
1	×	×	Setting prohibited																			
7, 6	BCW1n, BCW0n (n = 3, 4)	Bank Active Command Wait Control Specifies the number of wait states inserted from the bank active command to a read/write command, or from the precharge command to the bank active command. <table border="1" style="margin-top: 10px;"> <thead> <tr> <th>BCW1n</th> <th>BCW0n</th> <th>Number of wait states inserted</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1 (at least 1 wait is always inserted)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">2</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">3</td> </tr> </tbody> </table>	BCW1n	BCW0n	Number of wait states inserted	0	0	1 (at least 1 wait is always inserted)	0	1	1	1	0	2	1	1	3					
BCW1n	BCW0n	Number of wait states inserted																				
0	0	1 (at least 1 wait is always inserted)																				
0	1	1																				
1	0	2																				
1	1	3																				

Remark ×: don't care

★

Bit Position	Bit Name	Function															
5, 4	SSO1n, SSO0n (n = 3, 4)	<p>SDRAM Shift Width On-Page Control Specifies the address shift width during on-page judgment. When the external data bus width is 8 bits: Set SSO1n, SSO0n = 00B When the external data bus width is 16 bits: Set SSO1n, SSO0n = 01B</p> <table border="1"> <thead> <tr> <th>SSO1n</th> <th>SSO0n</th> <th>Address shift 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>16 bits</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	SSO1n	SSO0n	Address shift width	0	0	8 bits	0	1	16 bits	1	0	Setting prohibited	1	1	Setting prohibited
SSO1n	SSO0n	Address shift width															
0	0	8 bits															
0	1	16 bits															
1	0	Setting prohibited															
1	1	Setting prohibited															
3, 2	RAW1n, RAW0n (n = 3, 4)	<p>Row Address Width Control Specifies the row address width.</p> <table border="1"> <thead> <tr> <th>RAW1n</th> <th>RAW0n</th> <th>Row address width</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>11</td> </tr> <tr> <td>0</td> <td>1</td> <td>12</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table> <p>Caution Memories with a row address width above 13 cannot be controlled.</p>	RAW1n	RAW0n	Row address width	0	0	11	0	1	12	1	0	Setting prohibited	1	1	Setting prohibited
RAW1n	RAW0n	Row address width															
0	0	11															
0	1	12															
1	0	Setting prohibited															
1	1	Setting prohibited															
1, 0	SAW1n, SAW0n (n = 3, 4)	<p>Row Address Multiplex Width Control Specifies the address multiplex width during SDRAM access.</p> <table border="1"> <thead> <tr> <th>SAW1n</th> <th>SAW0n</th> <th>Address multiplex width</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>8</td> </tr> <tr> <td>0</td> <td>1</td> <td>9</td> </tr> <tr> <td>1</td> <td>0</td> <td>10</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	SAW1n	SAW0n	Address multiplex width	0	0	8	0	1	9	1	0	10	1	1	Setting prohibited
SAW1n	SAW0n	Address multiplex width															
0	0	8															
0	1	9															
1	0	10															
1	1	Setting prohibited															

★

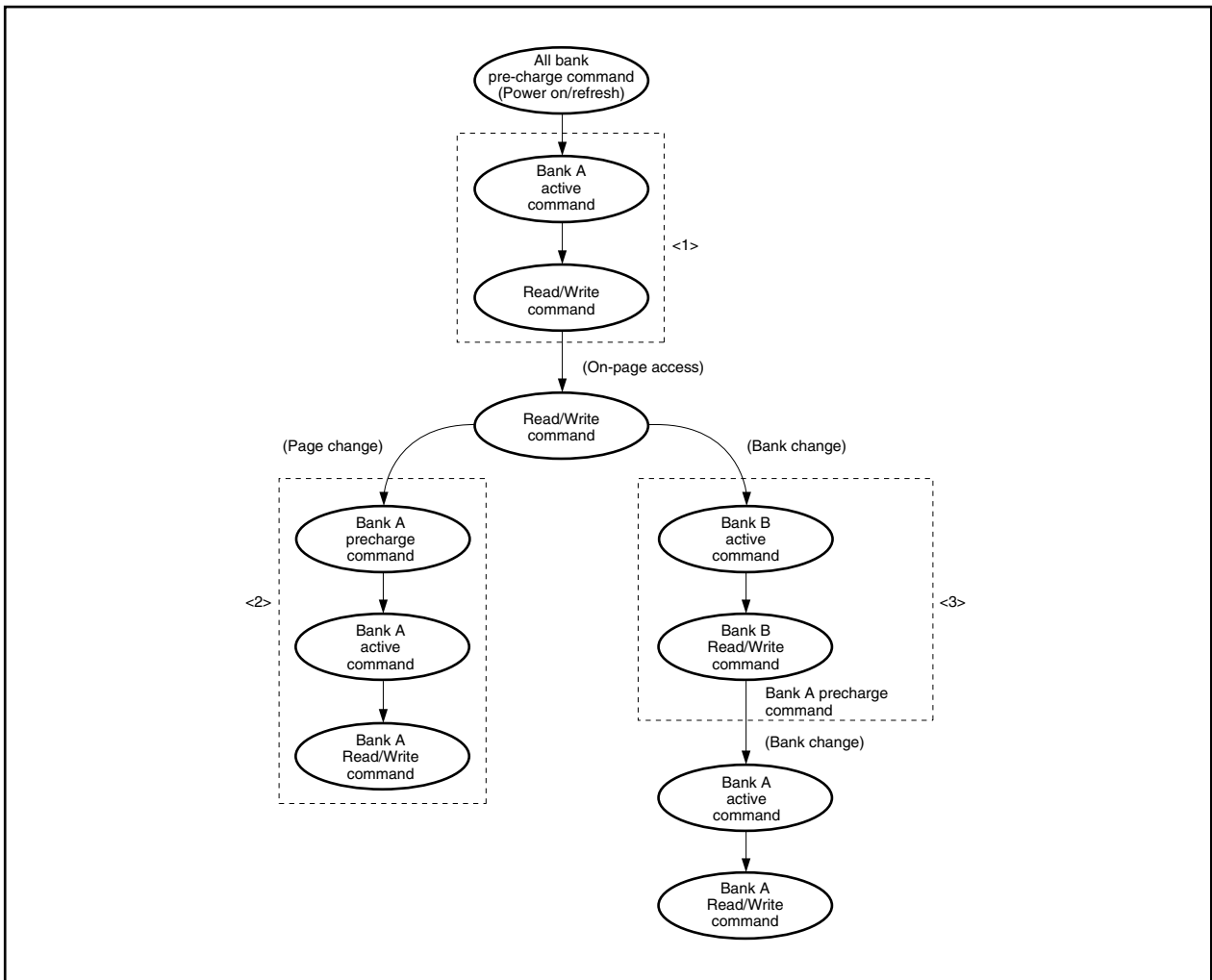
5.3.5 SDRAM access

During power-on or a refresh operation, the all bank precharge command is always issued for SDRAM. When accessing SDRAM after that, therefore, the active command and read/write command are issued in that order (see <1> in Figure 5-8).

If a page change occurs following this, the precharge command, active command, and read/write command are issued in that order (see <2> in Figure 5-8).

If a bank change occurs, the active command and read/write command for the bank to be accessed next are issued in that order. Following this read/write command, the precharge command for the bank that was accessed before the bank currently being accessed will be issued (see <3> in Figure 5-8).

Figure 5-8. State Transition of SDRAM Access



(1) SDRAM single read cycle

The SDRAM single read cycle is a cycle for reading from SDRAM by executing a load instruction (LD) for the SDRAM area, by fetching an instruction, or by 2-cycle DMA transfer.

In the SDRAM single read cycle, the active command (ACT) and read command (RD) are issued for SDRAM in that order. During on-page access, however, only the read command is issued and the precharge command and active command are not issued. When a page change occurs in the same bank, the precharge command (PRE) is issued before the active command.

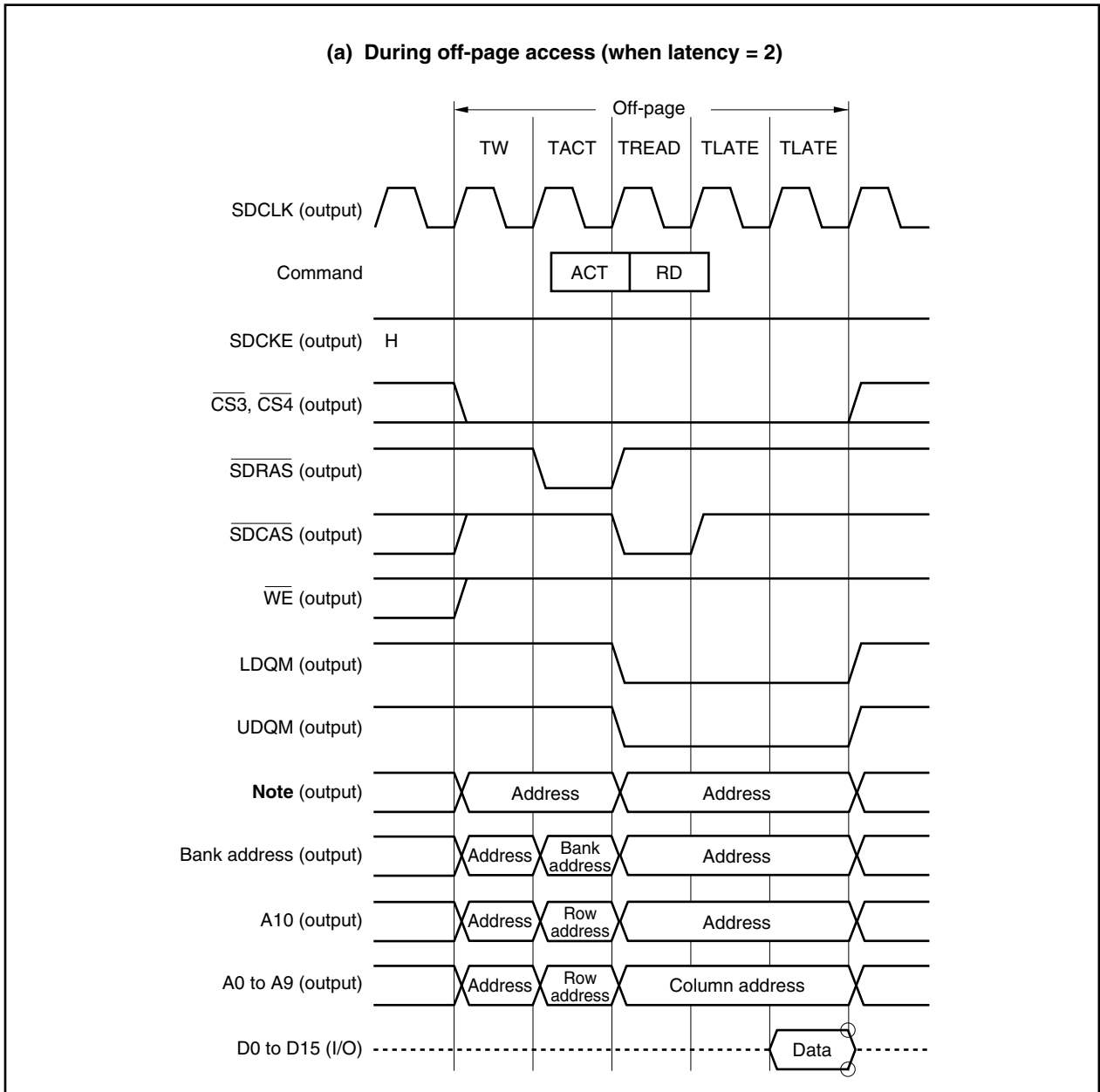
The timing to sample data is synchronized with rising of the UDQM and LDQM signals.

A one-state TW cycle is always inserted immediately before the all read command, which is activated by the CPU.

The number of idle states set by the bus cycle control register (BCC) are inserted before the read cycle (no idle state is inserted, however, if BCn1 and BCn0 are 00) (n = 3, 4). The timing charts of the SDRAM single read cycle are shown below.

Caution When executing a write access to SRAM or external I/O after read accessing SDRAM, data conflict may occur depending on the SDRAM data output float delay time. In such a case, avoid data conflict by inserting an idle state in the SDRAM space via a setting in the BCC register.

Figure 5-9. SDRAM Single Read Cycle (1/3)



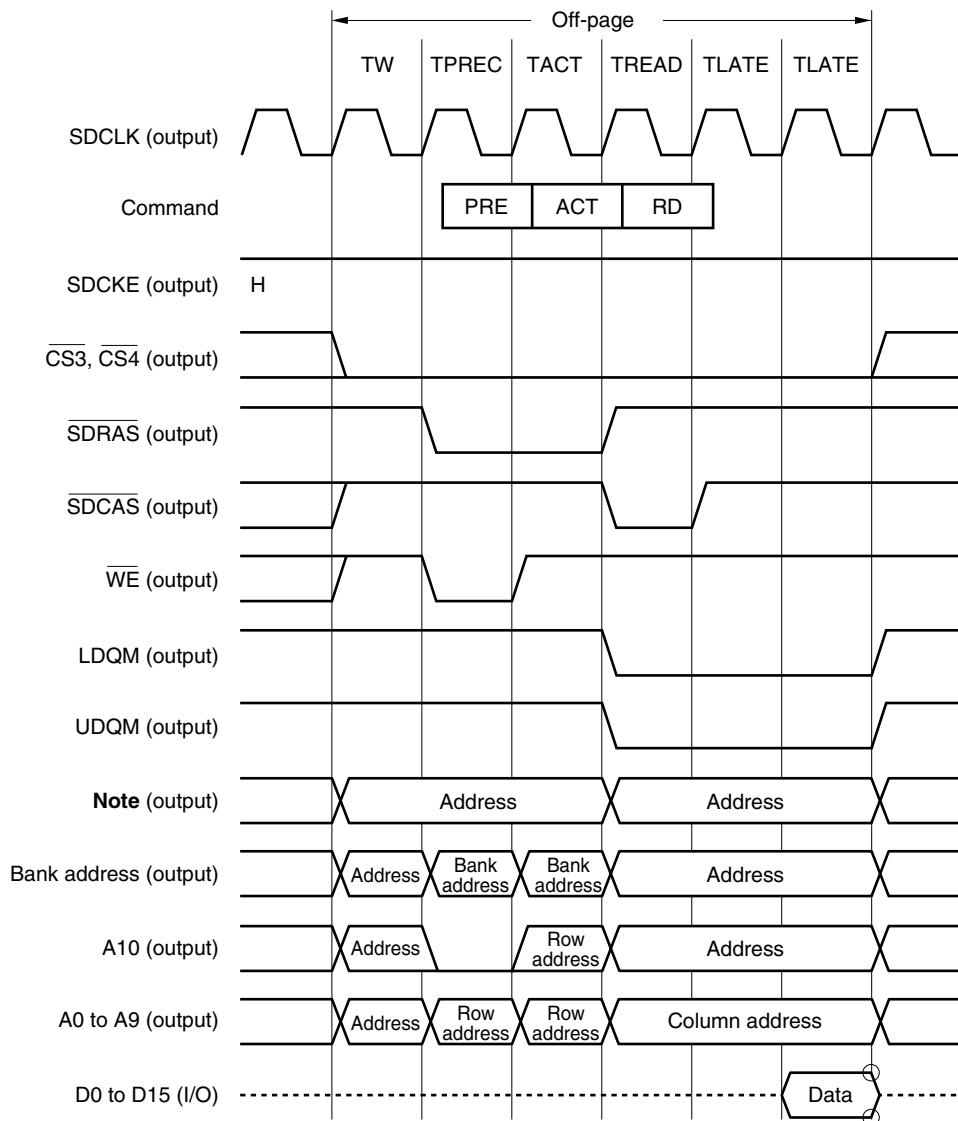
Note Addresses other than the bank address, A10, and A0 to A9.

Remarks 1. The circle O indicates the sampling timing.

2. The broken lines indicate the high-impedance state.

Figure 5-9. SDRAM Single Read Cycle (2/3)

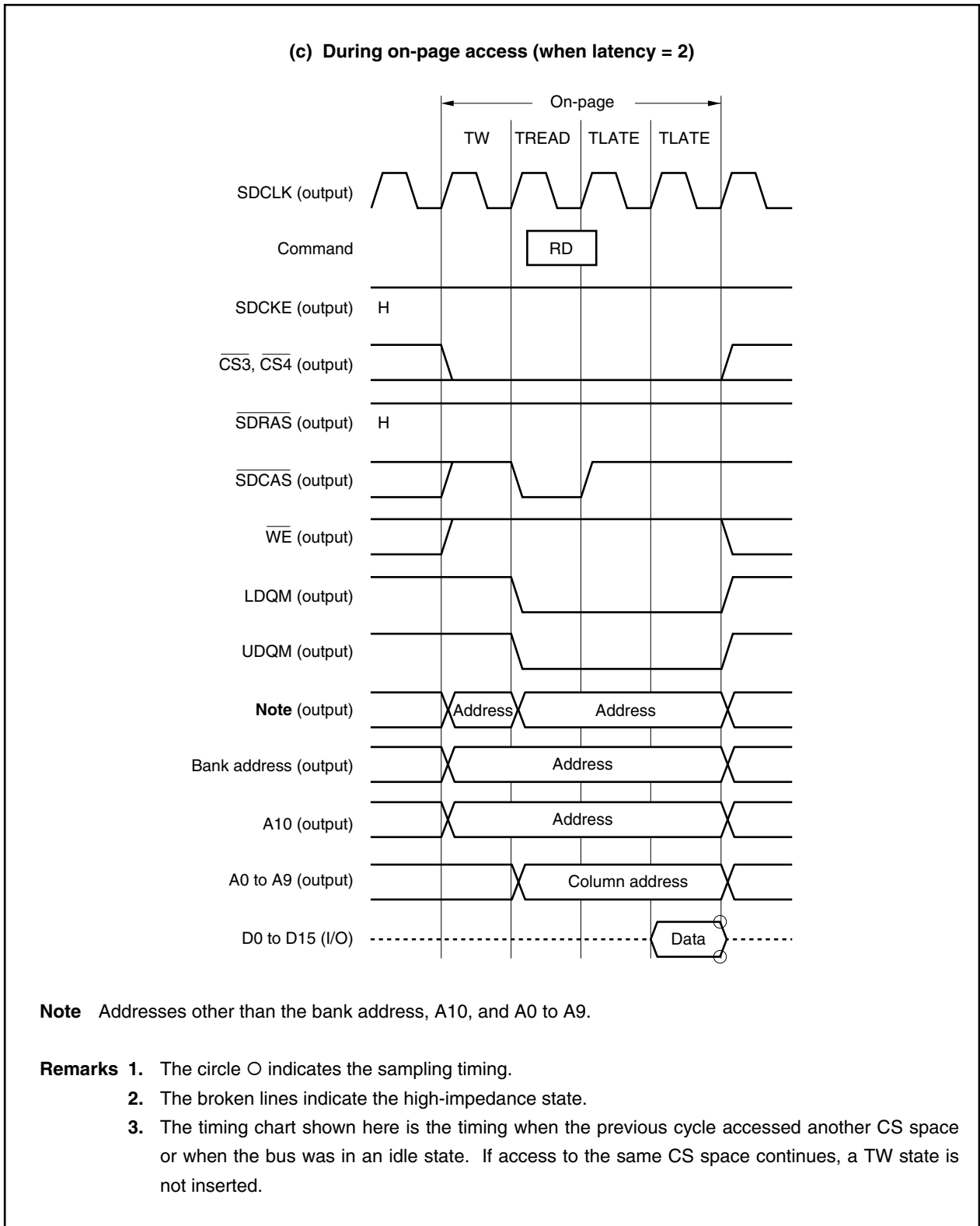
(b) During off-page access (when latency = 2, page change)



Note Addresses other than the bank address, A10, and A0 to A9.

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.

Figure 5-9. SDRAM Single Read Cycle (3/3)



(2) SDRAM single write cycle

The SDRAM single write cycle is a cycle for writing to SDRAM by executing a write instruction (ST) for the SDRAM area or by 2-cycle DMA transfer.

In the SDRAM single write cycle, the active command (ACT) and write command (WR) are issued for SDRAM in that order. During on-page access, however, only the write command is issued and the precharge command and active command are not issued. When a page change occurs in the same bank, the precharge command (PRE) is issued before the active command.

A one-state TW cycle is always inserted immediately before the all read command, which is activated by the CPU.

The timing charts of the SDRAM single write cycle are shown below.

Figure 5-10. SDRAM Single Write Cycle (1/3)

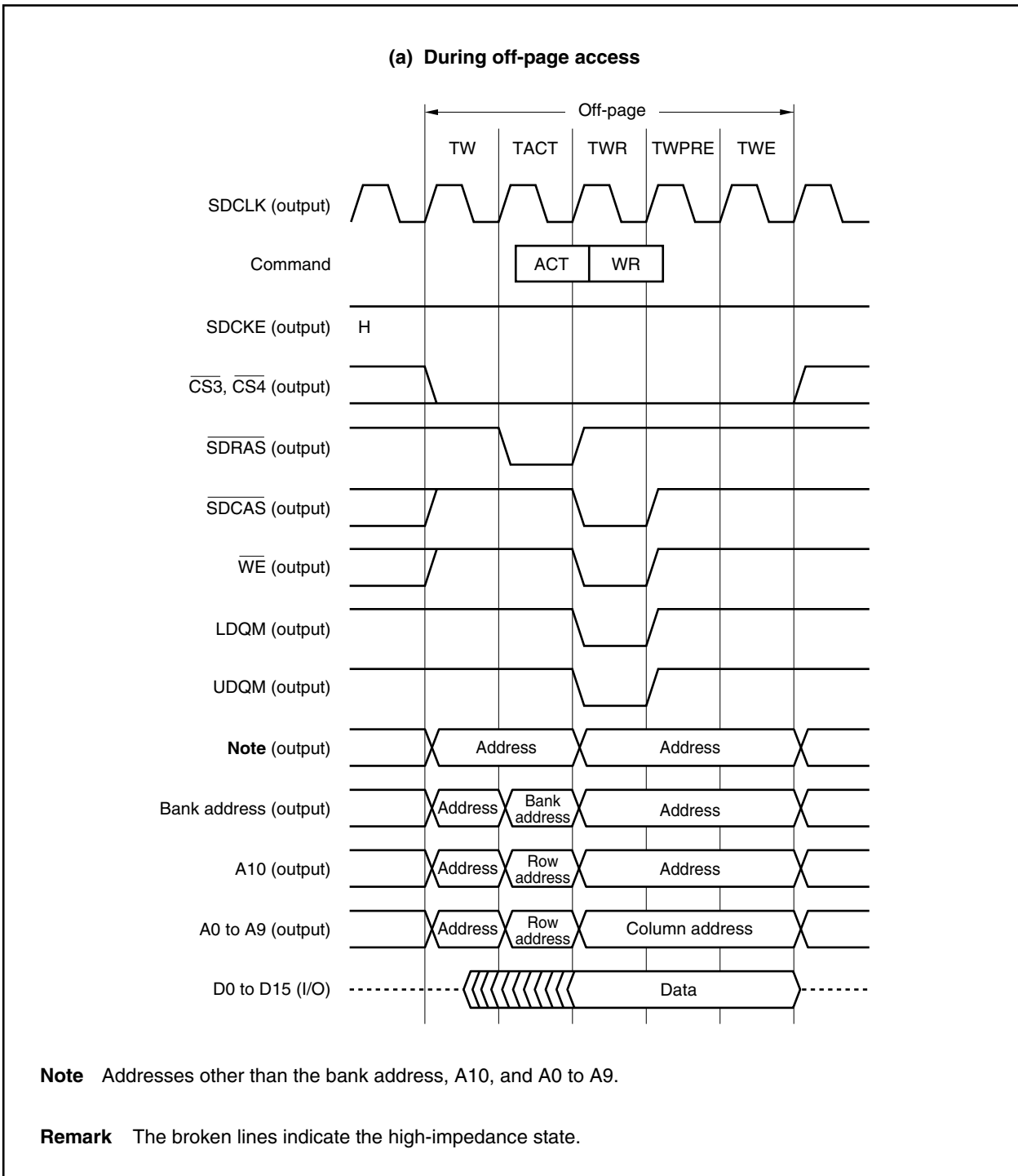


Figure 5-10. SDRAM Single Write Cycle (2/3)

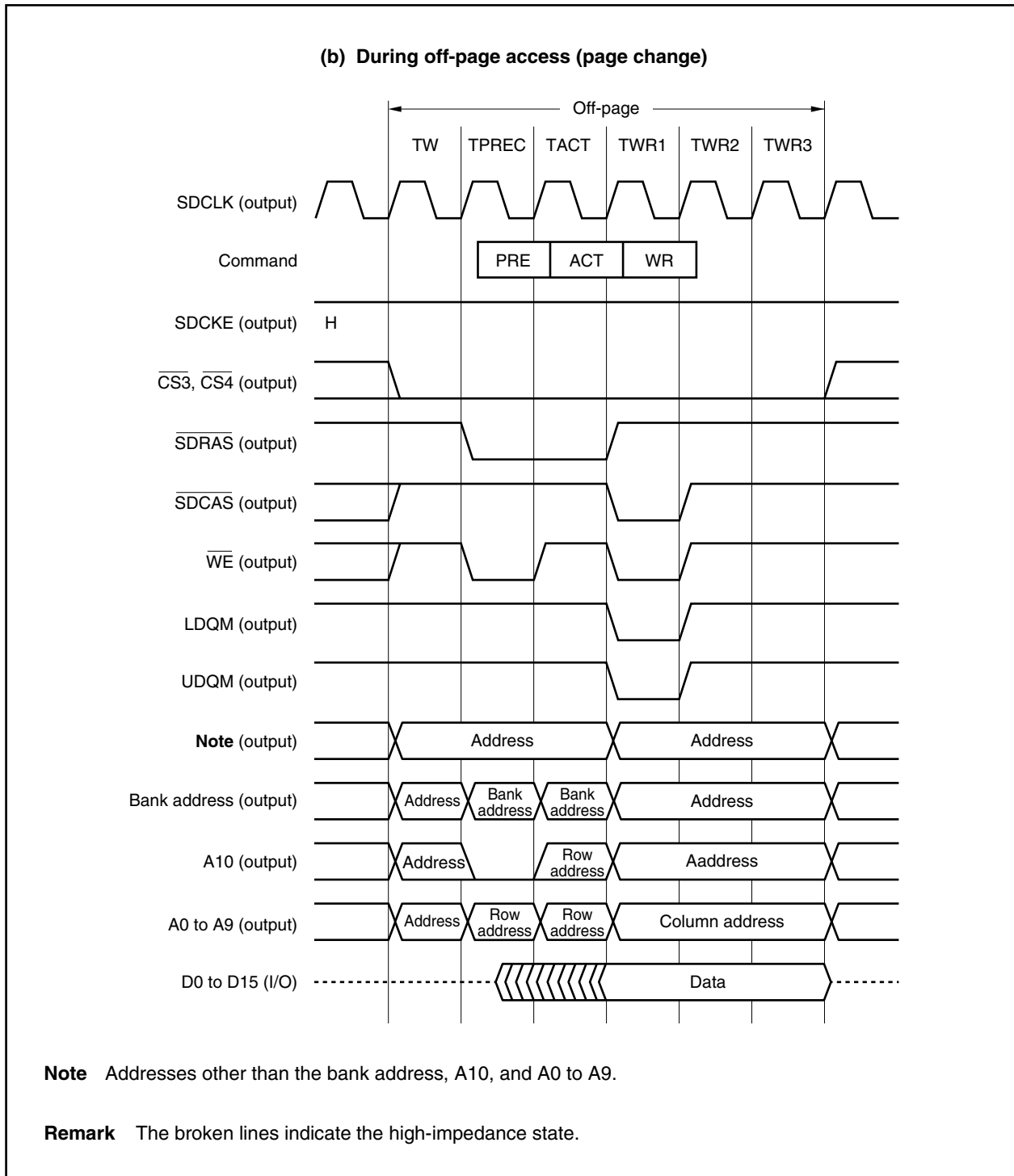
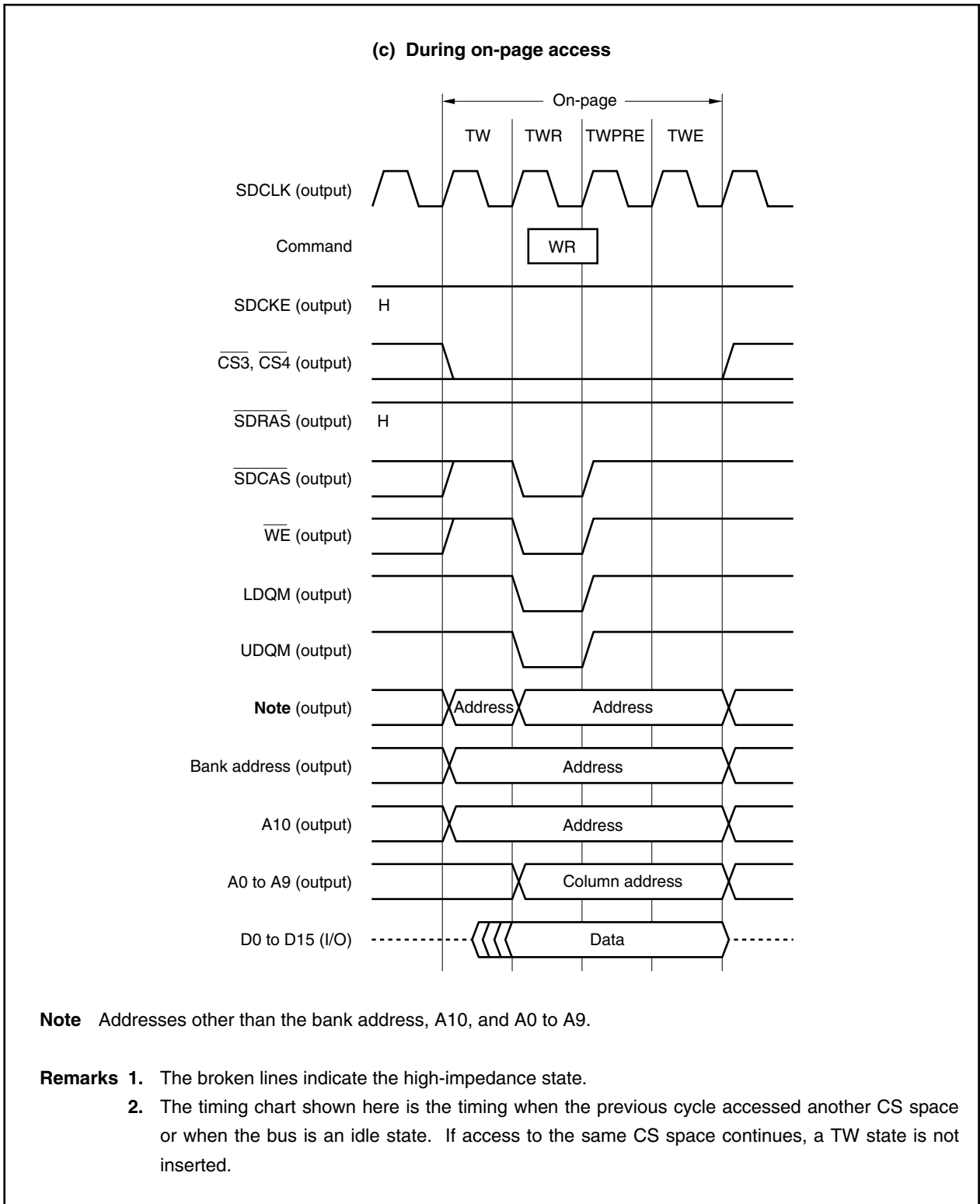


Figure 5-10. SDRAM Single Write Cycle (3/3)



(3) SDRAM access timing control

The SDRAM access timing can be controlled by SDRAM configuration registers 3 and 4 (SCR3, SCR4). For details, refer to **5.3.4 SDRAM configuration registers 3, 4 (SCR3, SCR4)**.

Caution Wait control by the $\overline{\text{WAIT}}$ pin is not available during SDRAM access.

(a) Number of waits from bank active command to read/write command

The number of wait states from bank active command issue to read/write command issue can be set by setting the BCW1n and BCW0n bits of the SCRn register (n = 3, 4).

BCW1n, BCW0n = 01B: 1 wait
 BCW1n, BCW0n = 10B: 2 waits
 BCW1n, BCW0n = 11B: 3 waits

(b) Number of waits from precharge command to bank active command

The number of wait states from precharge command issue to bank active command issue can be set by setting the BCW1n and BCW0n bits of the SCRn register (n = 3, 4).

BCW1n, BCW0n = 01B: 1 wait
 BCW1n, BCW0n = 10B: 2 waits
 BCW1n, BCW0n = 11B: 3 waits

(c) CAS latency setting when read

The CAS latency during a read operation can be set by setting the LTM2n to LTM0n bits of the SCRn register (n = 3, 4).

LTM2n to LTM0n = 010B: Latency = 2
 LTM2n to LTM0n = 011B: Latency = 3

(d) Number of waits from refresh command to next command

The number of wait states from refresh command issue to next command issue can be set by setting the BCW1n and BCW0n bits of the SCRn register. The number of wait states becomes four times the value set by the BCW1n and BCW0n (n = 3, 4).

BCW1n, BCW0n = 01B: 4 waits
 BCW1n, BCW0n = 10B: 8 waits
 BCW1n, BCW0n = 11B: 12 waits

Figure 5-11. SDRAM Access Timing (1/4)

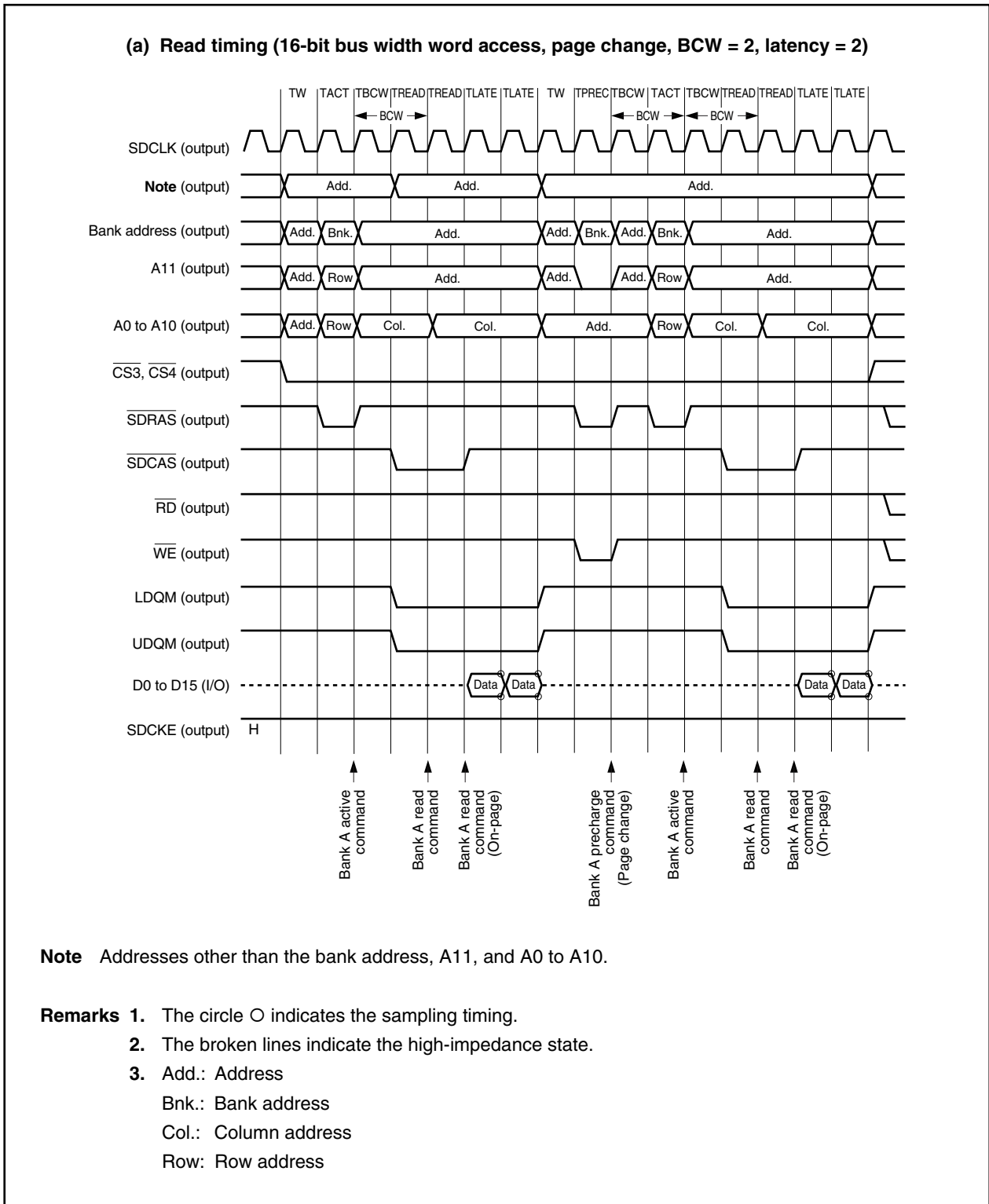
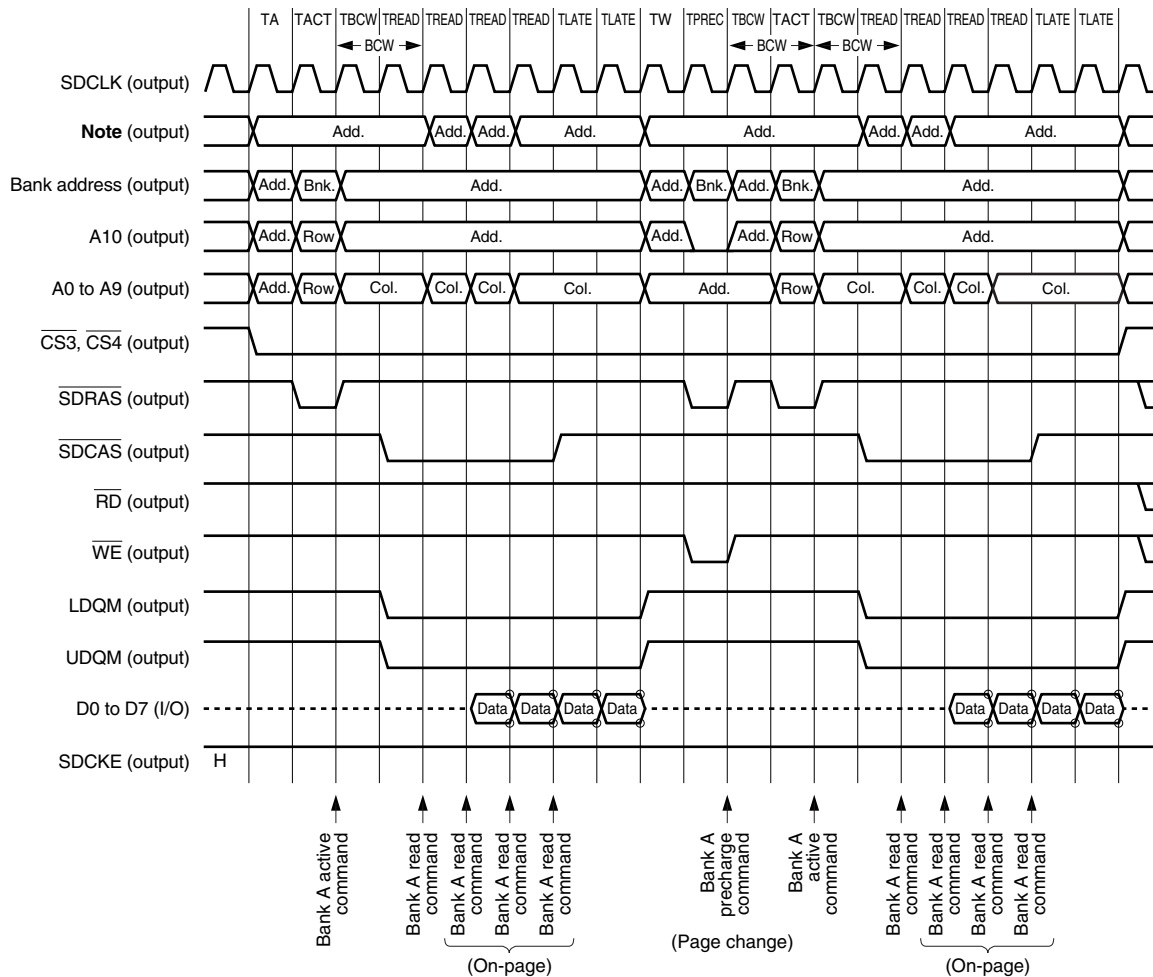


Figure 5-11. SDRAM Access Timing (2/4)

(b) Read timing (8-bit bus width word access, page change, BCW = 2, latency = 2)



Note Addresses other than the bank address, A10, and A0 to A9.

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. Add.: Address
 Bnk.: Bank address
 Col.: Column address
 Row: Row address

Figure 5-11. SDRAM Access Timing (3/4)

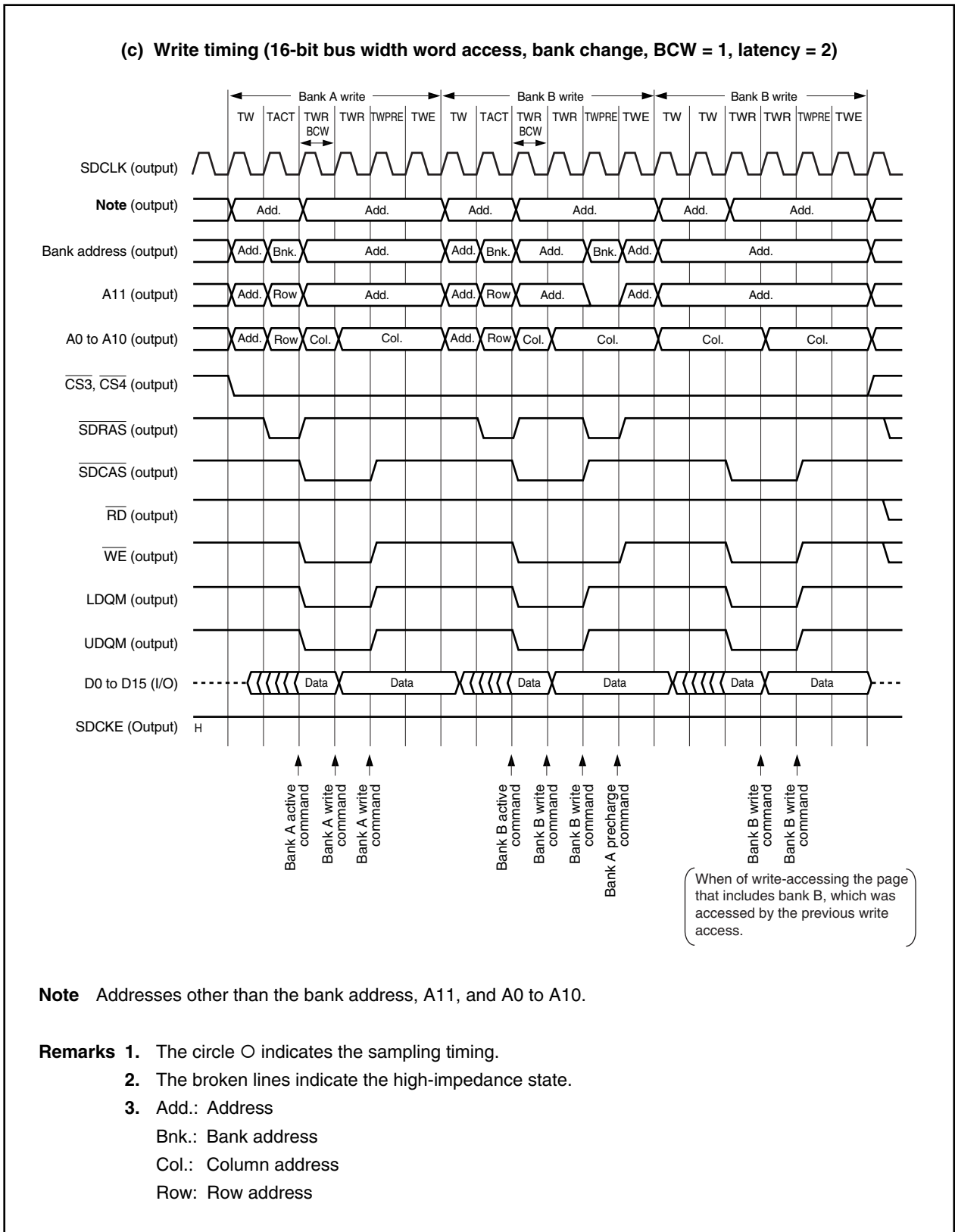
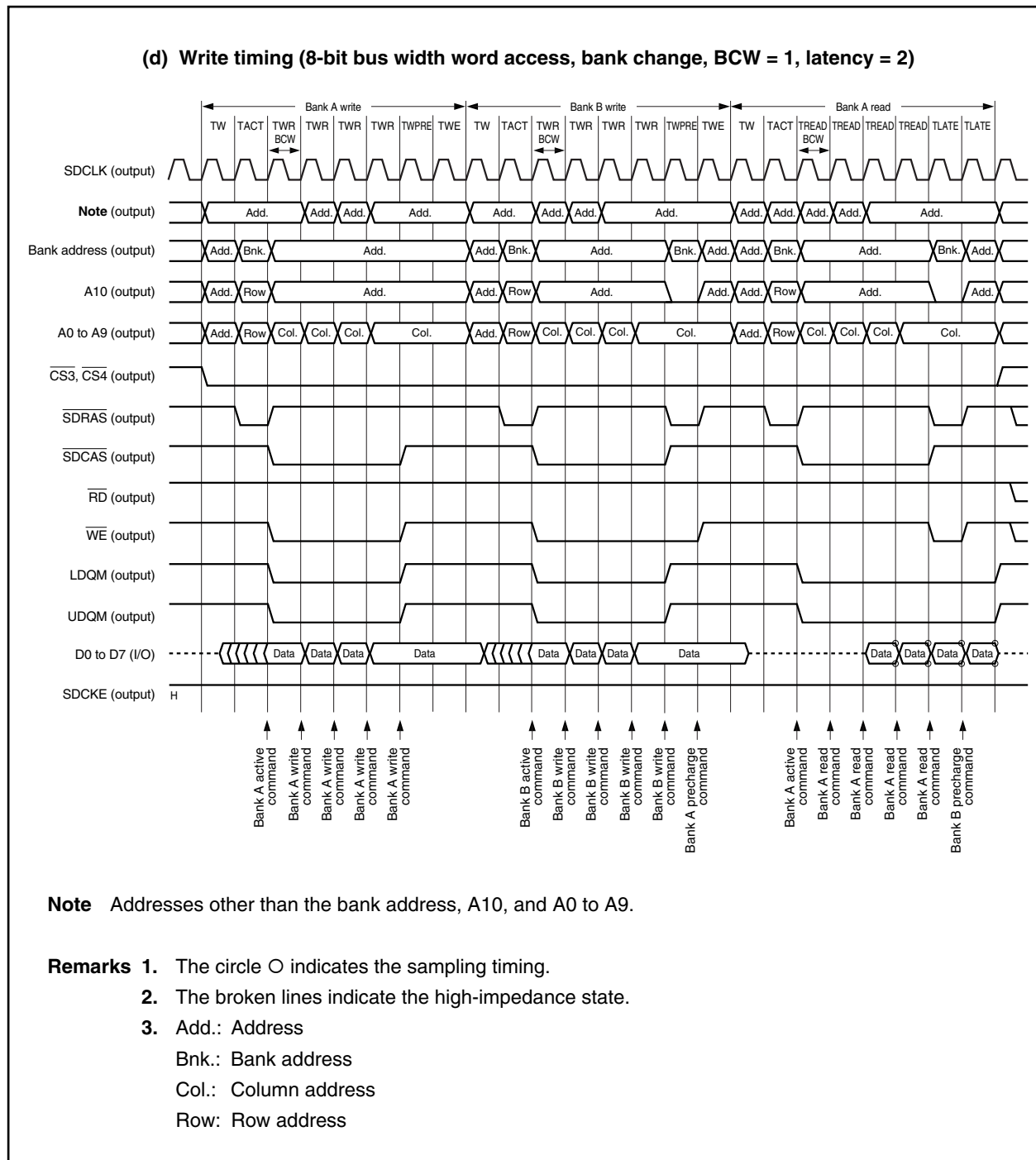


Figure 5-11. SDRAM Access Timing (4/4)



5.3.6 Refresh control function

The V850E/MA2 can generate a refresh cycle. The refresh cycle is set with SDRAM refresh control registers 3 and 4 (RFS3, RFS4). The RFS3 and RFS4 registers correspond to $\overline{CS3}$ and $\overline{CS4}$. For example, to connect SDRAM to $\overline{CS3}$, set RFS3.

When another bus master occupies the external bus, the SDRAM controller cannot occupy the external bus. In this case, the SDRAM controller issues a refresh request to the bus master by changing the \overline{REFRQ} signal to active (low level).

During a refresh operation, the address bus retains the state it was in just before the refresh cycle.

(1) SDRAM refresh control registers 3, 4 (RFS3, RFS4)

These registers are used to enable or disable a refresh and set the refresh interval. The refresh interval is determined by the following calculation formula.

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

The refresh count clock and interval factor are determined by the REN_n bit and RIN_{5n} to RIN_{0n} bits, respectively, of the RFS_n register.

Note that n corresponds to the register number (3, 4) of SDRAM configuration registers 3 and 4 (SCR_3 , SCR_4).

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

Cautions 1. Write to the RFS3 and RFS4 registers 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 RFS3 and RFS4 registers is complete. However, it is possible to access external memory areas whose initialization settings are complete.

★

2. Immediately after the REN_n bit of the RFS_n register is set (1), the refresh cycle may be executed for the SDRAM ($n = 3, 4$). This does not affect the refresh cycle occurring at this time nor the operations after the refresh cycle is executed. The refresh cycles occurring thereafter will be executed normally according to the set interval. However, set the RFS_n register as shown below for applications which will have problems with this refresh cycle.

<1> With the ME_a bit of the BCT_m register set (1), set the BT_{a1} and BT_{a0} bits to 01 (page ROM connection) ($m = 0, 1, a = 3$ when $m = 0, a = 4$ when $m = 1$).

<2> Set the REN_n bit of the RFS_n register (1) to enable refresh ($n = 3, 4$).

<3> With the ME_a bit of the BCT_m register set (1), set the BT_{a1} and BT_{a0} bits to 11 (SDRAM connection) ($m = 0, 1, a = 3$ when $m = 0, a = 4$ when $m = 1$).

<4> Set the SCR_n register to initialize the SDRAM ($n = 3, 4$).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
RFS3	REN3	0	0	0	0	0	RCC13	RCC03	0	0	RIN53	RIN43	RIN33	RIN23	RIN13	RIN03	Address FFFFF4AEH	After reset 0000H
RFS4	REN4	0	0	0	0	0	RCC14	RCC04	0	0	RIN54	RIN44	RIN34	RIN24	RIN14	RIN04	FFFFF4B2H	0000H

Bit Position	Bit Name	Function																																																	
15	REN3, REN4	Refresh Enable Specifies whether CBR refresh is enabled or disabled. 0: Refresh disabled 1: Refresh enabled																																																	
9, 8	RCC1n, RCC0n (n = 3, 4)	Refresh Count Clock Specifies the refresh count clock (T _{RCY}). <table border="1"> <thead> <tr> <th>RCC1n</th> <th>RCC0n</th> <th>Refresh count clock (T_{RCY})</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>32/f_{xx}</td> </tr> <tr> <td>0</td> <td>1</td> <td>128/f_{xx}</td> </tr> <tr> <td>1</td> <td>0</td> <td>256/f_{xx}</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	RCC1n	RCC0n	Refresh count clock (T _{RCY})	0	0	32/f _{xx}	0	1	128/f _{xx}	1	0	256/f _{xx}	1	1	Setting prohibited																																		
RCC1n	RCC0n	Refresh count clock (T _{RCY})																																																	
0	0	32/f _{xx}																																																	
0	1	128/f _{xx}																																																	
1	0	256/f _{xx}																																																	
1	1	Setting prohibited																																																	
5 to 0	RIN5n to RIN0n (n = 3, 4)	Refresh Interval Sets the interval factor of the interval timer for the generation of the refresh timing. <table border="1"> <thead> <tr> <th>RIN5n</th> <th>RIN4n</th> <th>RIN3n</th> <th>RIN2n</th> <th>RIN1n</th> <th>RIN0n</th> <th>Interval factor</th> </tr> </thead> <tbody> <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> </tbody> </table>	RIN5n	RIN4n	RIN3n	RIN2n	RIN1n	RIN0n	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
RIN5n	RIN4n	RIN3n	RIN2n	RIN1n	RIN0n	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																																													

Remark f_{xx}: Internal system clock

Table 5-1. Example of Interval Factor Settings

Specified Refresh Interval Value (μs)	Refresh Count Clock (T_{RCY})	Interval Factor Value ^{Notes 1, 2}		
		$f_{\text{xx}} = 20 \text{ MHz}$	$f_{\text{xx}} = 33 \text{ MHz}$	$f_{\text{xx}} = 40 \text{ MHz}$
15.6	$32/f_{\text{xx}}$	9 (14.4)	16 (15.5)	19 (15.2)
	$128/f_{\text{xx}}$	2 (12.8)	4 (15.5)	4 (12.8)
	$256/f_{\text{xx}}$	1 (12.8)	2 (15.5)	2 (12.8)

- Notes**
1. The interval factor is set by bits RIN0n to RIN5n of the RFSn register ($n = 3, 4$).
 2. The values in parentheses are the calculated values for the refresh interval (μs).
Refresh interval (μs) = Refresh count clock (T_{RCY}) \times Interval factor

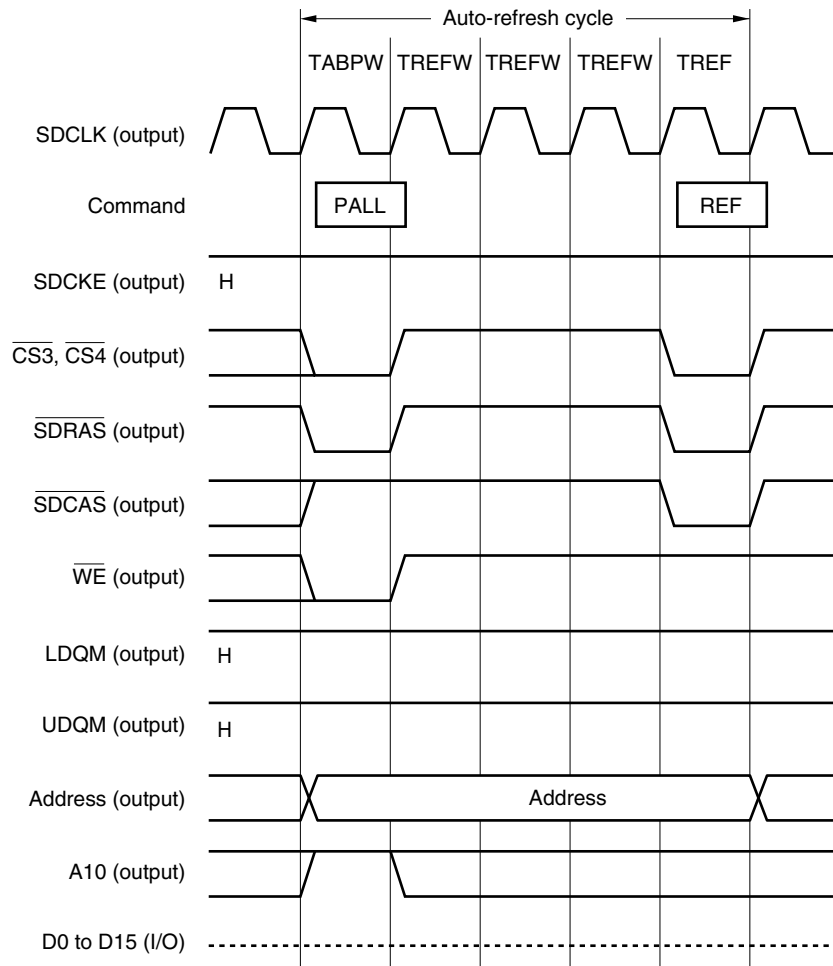
Remark f_{xx} : Internal system clock

The V850E/MA2 can automatically generate an auto refresh cycle and a self refresh cycle.

(2) Auto refresh cycle

In the auto refresh cycle, the auto refresh command (REF) is issued four clocks after the precharge command (PALL) is issued.

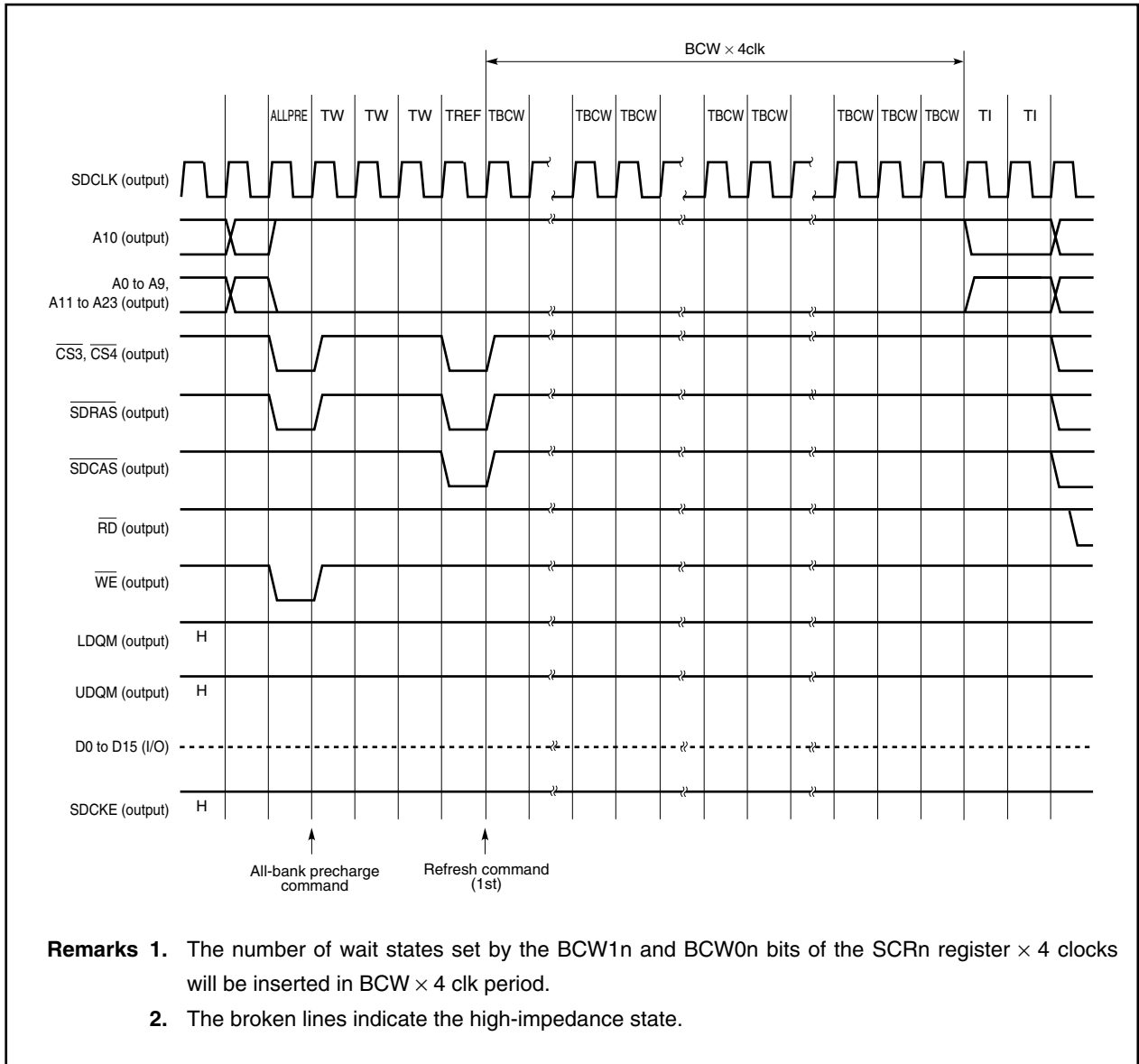
Figure 5-12. Auto Refresh Cycle



Remark The broken lines indicate the high-impedance state.

(3) Refresh timing

Figure 5-13. CBR Refresh Timing (SDRAM)



5.3.7 Self-refresh control function

In the case of transition to the IDLE mode or software STOP mode, the SDRAM controller generates the CBR self-refresh cycle.

Note that the $\overline{\text{SDRAS}}$ pulse width of SDRAM must meet the specification for SDRAM to enter the self-refresh operation.

Caution The internal ROM and internal RAM can be accessed even in the self-refresh cycle. However, access to an on-chip peripheral I/O register or external device is held pending until the self-refresh cycle is cleared.

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

(1) Release by NMI input

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

To release the self-refresh cycle, make the $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, LDQM, and UDQM signals inactive immediately.

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

To release the self-refresh cycle, make the $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, LDQM, and UDQM signals inactive after stabilizing oscillation.

(2) Release by INTP0n0 and INTP0n1 inputs (n = 0, 1)

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

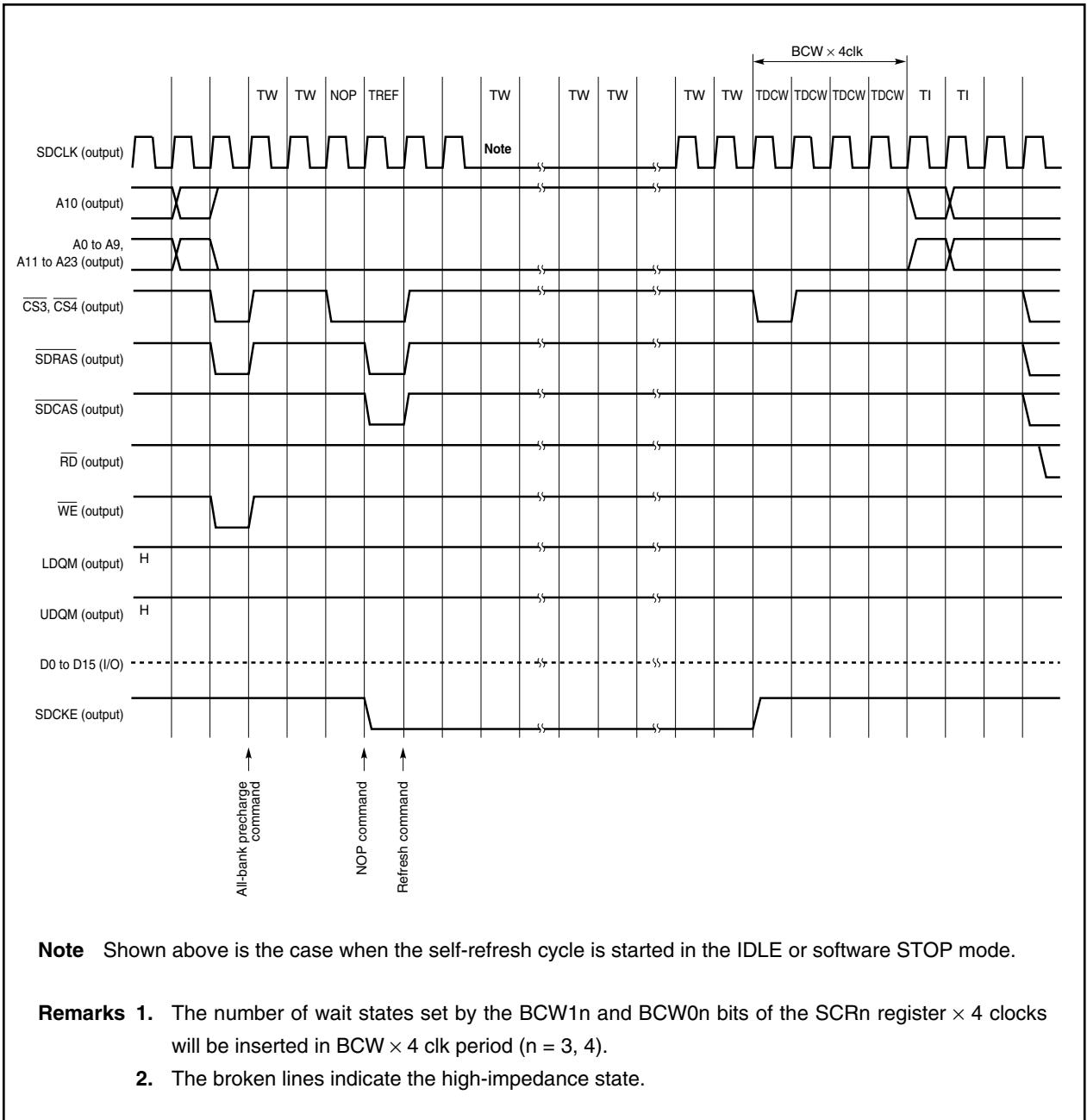
To release the self-refresh cycle, make the $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, LDQM, and UDQM signals inactive immediately.

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

To release the self-refresh cycle, make the $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, LDQM, and UDQM signals inactive after stabilizing oscillation.

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

Figure 5-14. Self Refresh Timing (SDRAM)



5.3.8 SDRAM initialization sequence

Be sure to initialize SDRAM when applying power.

- (1) Set the registers of SDRAM (other than SDRAM configuration registers 3 and 4 (SCR3, SCR4))
 - Bus cycle type configuration registers 0 and 1 (BCT0, BCT1)
 - Bus cycle control register (BCC)
 - SDRAM refresh control registers 3 and 4 (RFS3, RFS4)
- (2) Set SDRAM configuration registers 3 and 4 (SCR3, SCR4). When writing data to these registers, the following commands are issued for SDRAM in the order shown.
 - All bank precharge command
 - Refresh command (8 times)
 - Command that is used to set a mode register

Figures 5-15 and 5-16 show examples of SDRAM mode register setting timing.

Caution When using the SDCLK and SDCKE signals, it is necessary to set the SDLCK output mode and the SDCKE output mode for these signals by setting the PMCCD register. In this case, however, these settings must not be executed at the same time.

Be sure to set the SDCKE output mode after setting the SDCLK output mode (refer to 13.3.12 (2) (b) Port CD mode control register (PMCCD)).

Figure 5-15. SDRAM Mode Register Setting Cycle

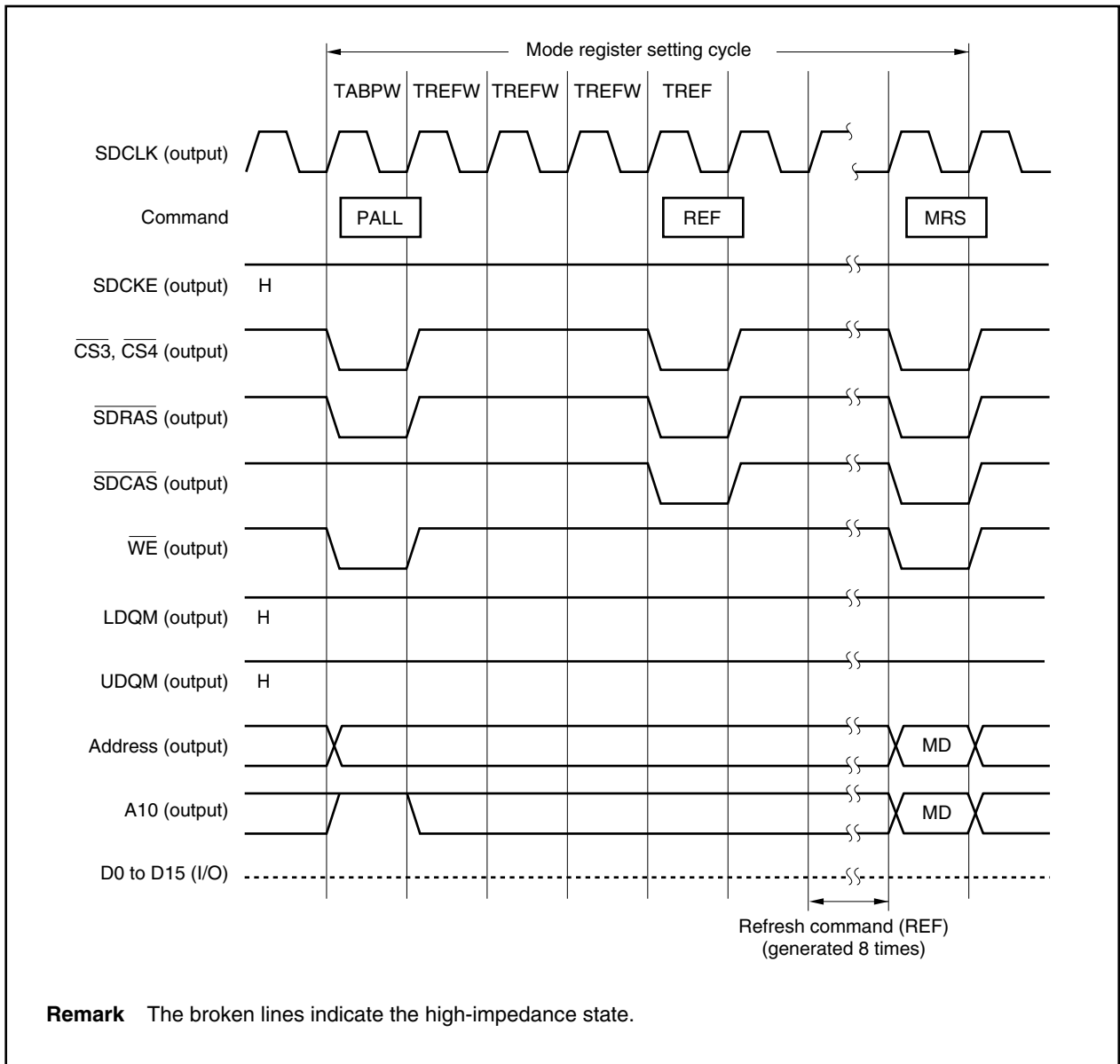
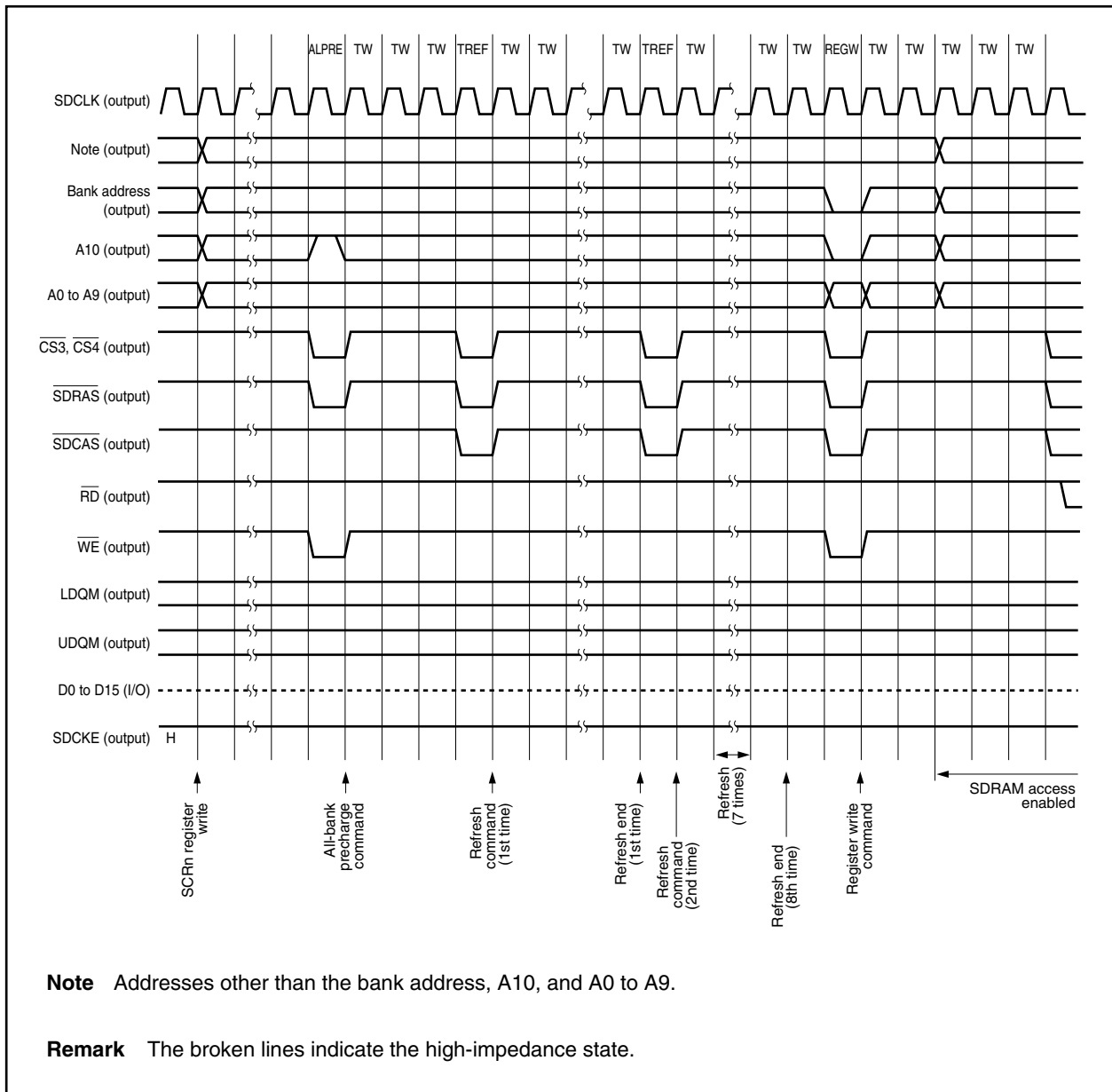


Figure 5-16. SDRAM Register Write Operation Timing



CHAPTER 6 DMA FUNCTIONS (DMA CONTROLLER)

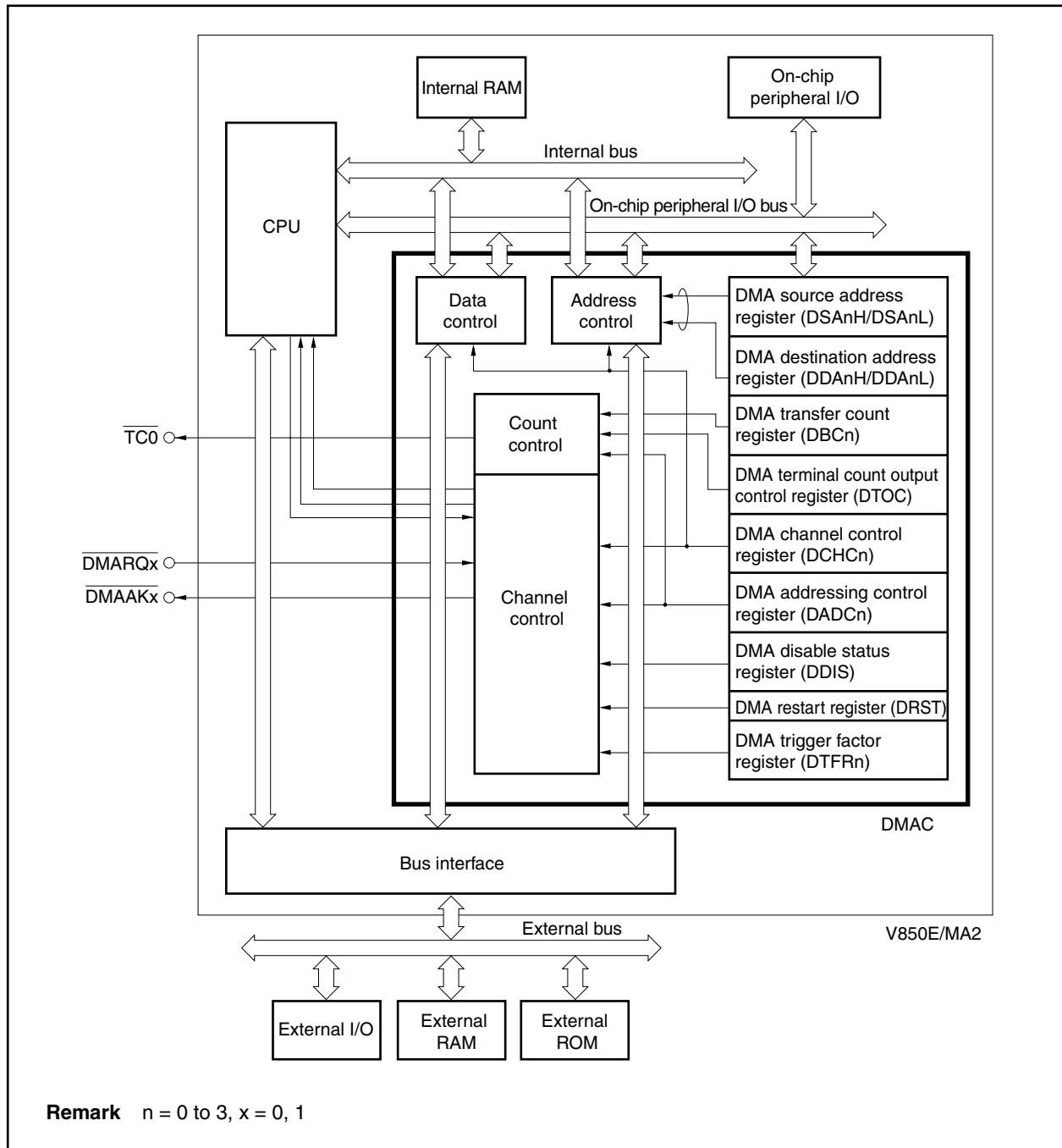
The V850E/MA2 includes a direct memory access (DMA) controller (DMAC) that executes and controls DMA transfer.

The DMAC controls data transfer between memory and I/O, or among memories, based on DMA requests issued by the on-chip peripheral I/O (serial interface, real-time pulse unit, and A/D converter), $\overline{\text{DMARQ0}}$ and $\overline{\text{DMARQ1}}$ pins, or software triggers (memory refers to internal RAM or external memory).

6.1 Features

- 4 independent DMA channels
- Transfer units: 8/16 bits
- Maximum transfer count: 65,536 (2^{16})
- Two-cycle transfer
- Three transfer modes
 - Single transfer mode
 - Single-step transfer mode
 - Block transfer mode
- Transfer requests
 - Request by interrupts from on-chip peripheral I/O (serial interface, real-time pulse unit, A/D converter)
 - Requests by $\overline{\text{DMARQ0}}$, $\overline{\text{DMARQ1}}$ pin input
 - Requests by software trigger
- Transfer objects
 - Memory \leftrightarrow I/O
 - Memory \leftrightarrow memory
- DMA transfer end output signals ($\overline{\text{TC0}}$)
- Next address setting function

6.2 Configuration



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 (28 bits each) for DMA channel n (n = 0 to 3). They are divided into two 16-bit registers, DSA_nH and DSA_nL.

Since these registers are configured as 2-stage FIFO buffer registers, a new source address for DMA transfer can be specified during DMA transfer. (Refer to **6.9 Next Address Setting Function**.)

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

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

★

Be sure to set bits 14 to 12 to 0. If they are set to 1, the operation is not guaranteed.

Caution When setting an address of a peripheral I/O register for the source address, be sure to specify an address between FFFF000H and FFFFFFFH. An address of the peripheral I/O register image (3FFF000H to 3FFFFFFH) must not be specified.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DSA0H	IR	0	0	0	SA27	SA26	SA25	SA24	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16	Address	After reset
																	FFFFF082H	Undefined
DSA1H	IR	0	0	0	SA27	SA26	SA25	SA24	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16	FFFFF08AH	Undefined
DSA2H	IR	0	0	0	SA27	SA26	SA25	SA24	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16	FFFFF092H	Undefined
DSA3H	IR	0	0	0	SA27	SA26	SA25	SA24	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16	FFFFF09AH	Undefined

Bit Position	Bit Name	Function
15	IR	Internal RAM Select Specifies the DMA source address. 0: External memory, on-chip peripheral I/O 1: Internal RAM
11 to 0	SA27 to SA16	Source Address Sets the DMA source addresses (A27 to A16). During DMA transfer, it stores the next DMA transfer source 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	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0	Address FFFFFF080H	After reset Undefined
DSA1L	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0	FFFFFF088H	Undefined
DSA2L	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0	FFFFFF090H	Undefined
DSA3L	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0	FFFFFF098H	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 transfer source address.

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

These registers are used to set the DMA destination address (28 bits each) for DMA channel n (n = 0 to 3). They are divided into two 16-bit registers, DDAnH and DDAnL.

Since these registers are configured as 2-stage FIFO buffer registers, a new destination address for DMA transfer can be specified during DMA transfer. (Refer to **6.9 Next Address Setting Function**.)

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

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

★

Be sure to set bits 14 to 12 to 0. If they are set to 1, the operation is not guaranteed.

Caution When setting an address of a peripheral I/O register for the destination address, be sure to specify an address between FFFF00H and FFFFFFFH. An address of the peripheral I/O register image (3FFF00H to 3FFFFFFH) must not be specified.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DDA0H	IR	0	0	0	DA27	DA26	DA25	DA24	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16	Address FFFFFF086H	After reset Undefined
DDA1H	IR	0	0	0	DA27	DA26	DA25	DA24	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16	FFFFFF08EH	Undefined
DDA2H	IR	0	0	0	DA27	DA26	DA25	DA24	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16	FFFFFF096H	Undefined
DDA3H	IR	0	0	0	DA27	DA26	DA25	DA24	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16	FFFFFF09EH	Undefined

Bit Position	Bit Name	Function
15	IR	Internal RAM Select Specifies the DMA destination address. 0: External memory, on-chip peripheral I/O 1: Internal RAM
11 to 0	DA27 to DA16	Destination Address Sets the DMA destination addresses (A27 to A16). During DMA transfer, it stores the next DMA transfer destination address.

(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	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	Address FFFFFF084H	After reset Undefined
DDA1L	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	FFFFFF08CH	Undefined
DDA2L	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	FFFFFF094H	Undefined
DDA3L	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	FFFFFF09CH	Undefined

Bit Position	Bit Name	Function
15 to 0	DA15 to DA0	Destination Address Sets the DMA destination address (A15 to A0). During DMA transfer, it stores the next DMA transfer destination address.

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 channels n (n = 0 to 3). They store the remaining transfer counts during DMA transfer.

Since these registers are configured as 2-stage FIFO buffer registers, a new DMA byte transfer count for DMA transfer can be specified during DMA transfer. (Refer to **6.9 Next Address Setting Function.**)

These registers are decremented by 1 for each transfer, and transfer ends when a borrow occurs.

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

Remark If a terminal count occurs without the DBCn register being rewritten during DMA transfer and then the DBCn register is read, the value set immediately before the DMA transfer is read (even after the completion of transfer, 0000H is not read).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DBC0	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0	Address FFFFF0C0H	After reset Undefined
DBC1	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0	FFFFF0C2H	Undefined
DBC2	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0	FFFFF0C4H	Undefined
DBC3	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0	FFFFF0C6H	Undefined

Bit Position	Bit Name	Function										
15 to 0	BC15 to BC0	Byte Count Sets the byte transfer count. It stores the remaining byte transfer count during DMA transfer.										
		<table border="1"> <thead> <tr> <th>DBCn (n = 0 to 3)</th> <th>States</th> </tr> </thead> <tbody> <tr> <td>0000H</td> <td>Byte transfer count 1 or remaining byte transfer count</td> </tr> <tr> <td>0001H</td> <td>Byte transfer count 2 or remaining byte transfer count</td> </tr> <tr> <td style="text-align: center;">:</td> <td style="text-align: center;">:</td> </tr> <tr> <td>FFFFH</td> <td>Byte transfer count 65,536 (2¹⁶) or remaining byte transfer count</td> </tr> </tbody> </table>	DBCn (n = 0 to 3)	States	0000H	Byte transfer count 1 or remaining byte transfer count	0001H	Byte transfer count 2 or remaining byte transfer count	:	:	FFFFH	Byte transfer count 65,536 (2 ¹⁶) or remaining byte transfer count
DBCn (n = 0 to 3)	States											
0000H	Byte transfer count 1 or remaining byte transfer count											
0001H	Byte transfer count 2 or remaining byte transfer count											
:	:											
FFFFH	Byte transfer count 65,536 (2 ¹⁶) or remaining byte transfer count											

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

These 16-bit registers are used to control the DMA transfer modes for DMA channel n (n = 0 to 3). These registers cannot be accessed during DMA operation.

They can be read/written in 16-bit units.

★ Be sure to set bits 13 to 8, 1, and 0 to 0. If they are set to 1, the operation is not guaranteed.

Caution The DS1 and DS0 bits are used to set how many bits of data are transferred.

When 8-bit data (DS1, DS0 bits = 00) is set, the lower data bus (D0 to D7) is not necessarily used.

When the transfer data size is set to 16 bits, the transfer must start from an address with bit 1 of the lower address aligned to "0". In this case, the transfer cannot start from an odd address.

(1/2)

	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		
DADC0	DS1 DS0 0 0 0 0 0 0 SAD1 SAD0 DAD1 DAD0 TM1 TM0 0 0	Address FFFFFF0D0H	After reset 0000H
DADC1	DS1 DS0 0 0 0 0 0 0 SAD1 SAD0 DAD1 DAD0 TM1 TM0 0 0	FFFFFF0D2H	0000H
DADC2	DS1 DS0 0 0 0 0 0 0 SAD1 SAD0 DAD1 DAD0 TM1 TM0 0 0	FFFFFF0D4H	0000H
DADC3	DS1 DS0 0 0 0 0 0 0 SAD1 SAD0 DAD1 DAD0 TM1 TM0 0 0	FFFFFF0D6H	0000H

Bit Position	Bit Name	Function															
15, 14	DS1, DS0	Data Size Sets the transfer data size for DMA transfer. <table border="1" style="margin-top: 10px;"> <thead> <tr> <th>DS1</th> <th>DS0</th> <th>Transfer data size</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>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	DS1	DS0	Transfer data size	0	0	8 bits	0	1	16 bits	1	0	Setting prohibited	1	1	Setting prohibited
DS1	DS0	Transfer data size															
0	0	8 bits															
0	1	16 bits															
1	0	Setting prohibited															
1	1	Setting prohibited															
7, 6	SAD1, SADO	Source Address count Direction Sets the count direction of the source address for DMA channel n (n = 0 to 3). <table border="1" style="margin-top: 10px;"> <thead> <tr> <th>SAD1</th> <th>SAD0</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>	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															

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 (n = 0 to 3). <table border="1" data-bbox="560 367 1409 567"> <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" data-bbox="560 709 1409 930"> <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>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Block transfer mode</td> </tr> </tbody> </table>	TM1	TM0	Transfer mode	0	0	Single transfer mode	0	1	Single-step transfer mode	1	0	Setting prohibited	1	1	Block transfer mode
TM1	TM0	Transfer mode															
0	0	Single transfer mode															
0	1	Single-step transfer mode															
1	0	Setting prohibited															
1	1	Block transfer mode															

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

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

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

- ★ Be sure to set bits 6 to 4 to 0. If they are set to 1, the operation is not guaranteed.
- ★ **Caution** Setting the MLEn bit to 1 is valid only for starting DMA transfer (hardware DMA) by $\overline{\text{DMARQm}}$ pin input or an interrupt from the on-chip peripheral I/O (n = 0 to 3, m = 0, 1). To start DMA transfer (software DMA) by setting the STGn bit to 1, read the TCn bit and check that it is set to 1, and then set the STGn bit to 1.

(1/2)

	<7>	6	5	4	<3>	<2>	<1>	<0>	Address	After reset
DCHC0	TC0	0	0	0	MLE0	INIT0	STG0	E00	FFFFF0E0H	00H
DCHC1	TC1	0	0	0	MLE1	INIT1	STG1	E11	FFFFF0E2H	00H
DCHC2	TC2	0	0	0	MLE2	INIT2	STG2	E22	FFFFF0E4H	00H
DCHC3	TC3	0	0	0	MLE3	INIT3	STG3	E33	FFFFF0E6H	00H

Bit Position	Bit Name	Function
7	TCn	Terminal Count This status bit indicates whether DMA transfer through DMA channel n has ended or not. This bit is read-only. It is set to 1 when DMA transfer ends and cleared (to 0) when it is read. 0: DMA transfer had not ended. 1: DMA transfer had ended.
3	MLEn	Multi Link Enable Bit When this bit is set to 1 at terminal count output, the Enn bit is not cleared to 0 and the DMA transfer enable state is retained. When the next DMA transfer request is a $\overline{\text{DMARQm}}$ pin input or an interrupt from the on-chip peripheral I/O (hardware DMA), the DMA transfer request can be acknowledged even when the TCn bit is not read. When the next DMA transfer request is the setting of the STGn bit to 1 (software DMA), the DMA transfer request can be acknowledged by reading and clearing the TCn bit to 0. When this bit is cleared to 0 at terminal count output, the Enn bit is cleared to 0 and the DMA transfer disable state is entered. At the next DMA transfer request, the setting of the Enn bit to 1 and the reading of the TCn bit are required.
2	INITn	Initialize When this bit is set to 1, DMA transfer is forcibly terminated.

★ **Remark** n = 0 to 3
n = 0, 1

Bit Position	Bit Name	Function
1	STGn	Software Trigger If this bit is set to 1 in the DMA transfer enable state (TCn bit = 0, Enn bit = 1), DMA transfer is started.
0	Enn	Enable Specifies whether DMA transfer through DMA channel n is to be enabled or disabled. This bit is cleared to 0 when DMA transfer ends. It is also cleared to 0 when DMA transfer is forcibly terminated by means of setting the INITn bit to 1 or by NMI input. 0: DMA transfer disabled 1: DMA transfer enabled

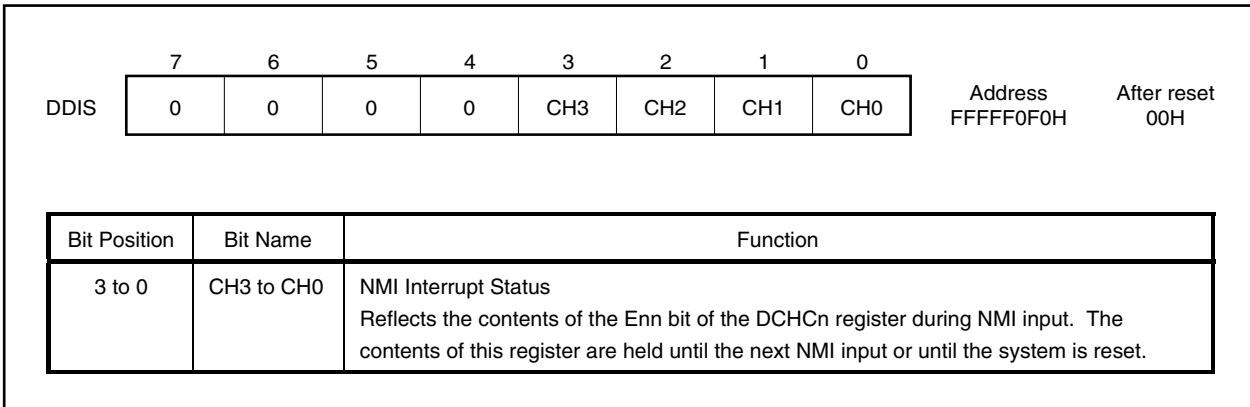
Remark n = 0 to 3
m = 0, 1

6.3.6 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).

This register is read-only in 8-bit units.

- ★ Be sure to set bits 7 to 4 to 0. If they are set to 1, the operation is not guaranteed.

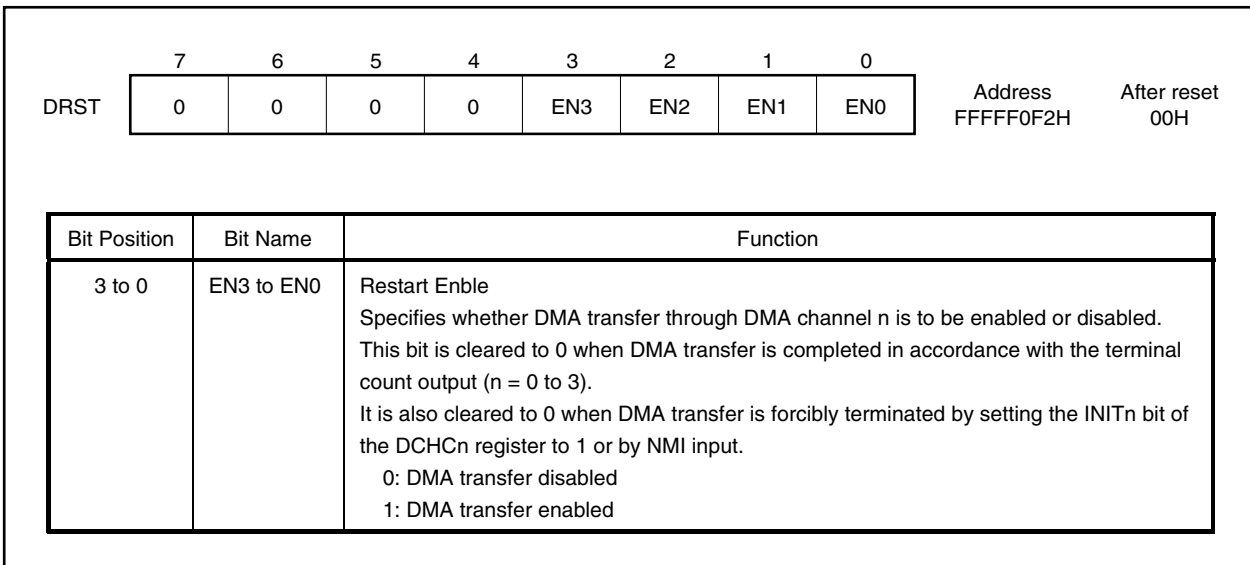


6.3.7 DMA restart register (DRST)

This register is used to restart DMA transfer that has been forcibly interrupted by NMI input. The ENn bit of this register and the Enn bit of the DCHCn register are linked to each other (n = 0 to 3). Following forcible interrupt by NMI input, the DMA channel that was interrupted is confirmed from the contents of the DDIS register, and DMA transfer is restarted by setting the ENn bit of the corresponding channel to 1.

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

- ★ Be sure to set bits 7 to 4 to 0. If they are set to 1, the operation is not guaranteed.

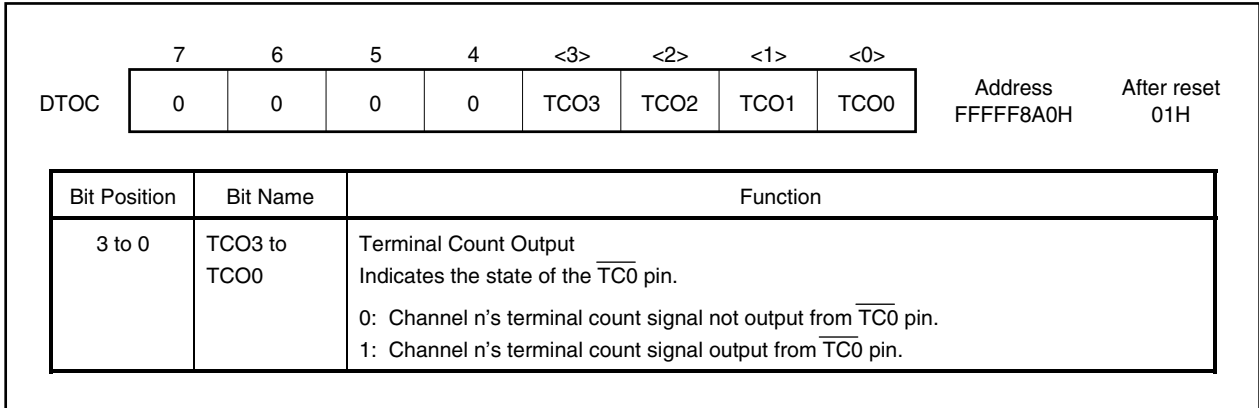


6.3.8 DMA terminal count output control register (DTCO)

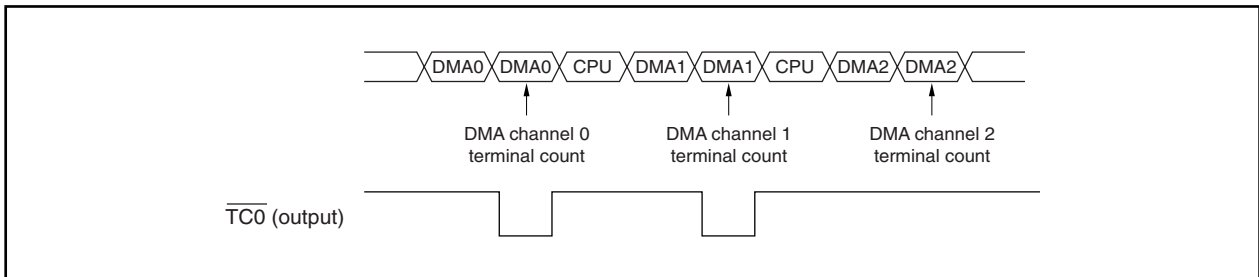
The DMA terminal count output control register (DTCO) is an 8-bit register that controls the terminal count outputs from each DMA channel. Terminal count signals from each DMA channel can be brought together and output from the $\overline{TC0}$ pin.

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

★



The following shows an example of the case when the DTCO register is set to 03H.



6.3.9 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 on-chip peripheral I/O.

The interrupt requests set with these registers serve as DMA transfer startup factors.

These registers can be read/written in 8-bit units. However, only bit 7 (DFn) can be read/written in 1-bit units.

- ★ **Cautions** 1. When changing the setting of the DTFRn register, be sure to stop the DMA operation.
- 2. An interrupt request input in the standby mode (IDLE or software STOP mode) cannot be a DMA transfer start factor.

(1/2)

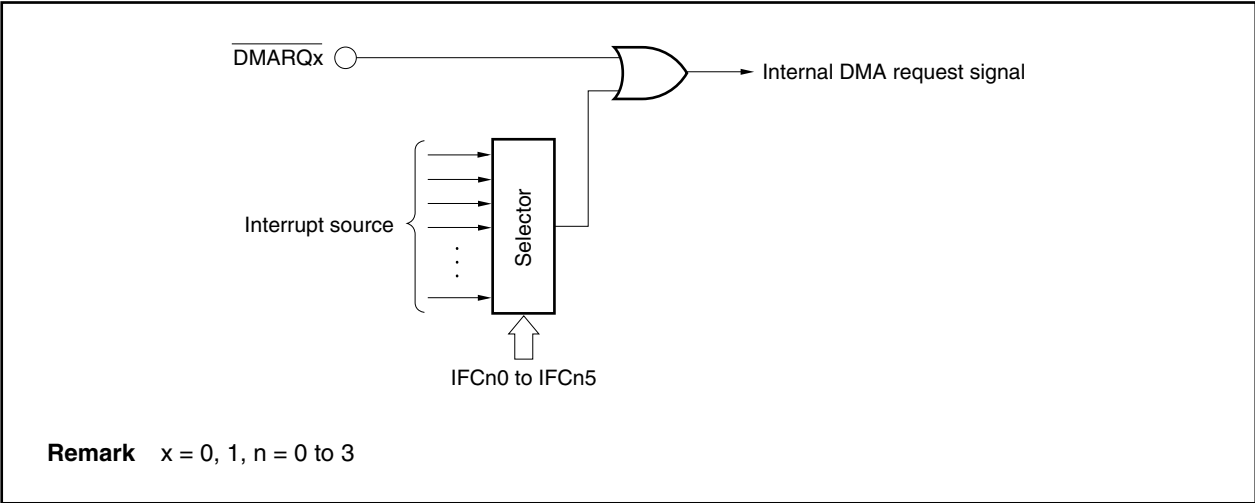
	<7>	6	5	4	3	2	1	0		
DTFR0	DF0	0	IFC05	IFC04	IFC03	IFC02	IFC01	IFC00	Address FFFFF810H	After reset 00H
DTFR1	DF1	0	IFC15	IFC14	IFC13	IFC12	IFC11	IFC10	FFFFF812H	00H
DTFR2	DF2	0	IFC25	IFC24	IFC23	IFC22	IFC21	IFC20	FFFFF814H	00H
DTFR3	DF3	0	IFC35	IFC34	IFC33	IFC32	IFC31	IFC30	FFFFF816H	00H

Bit Position	Bit Name	Function																																																								
7	DFn	<p>DMA Request Flag This is a DMA transfer request flag. Only 0 can be written to this flag. 0: DMA transfer not requested 1: DMA transfer requested</p> <p>If the interrupt specified as the DMA transfer startup factor occurs and it is necessary to clear the DMA transfer request while DMA transfer is disabled (including when it is aborted by NMI or forcibly terminated by software), stop the operation of the source causing the interrupt, and then clear the DFn bit to 0 (for example, disable reception in the case of serial reception). If it is clear that the interrupt does not occur until DMA transfer is resumed next, it is not necessary to stop the operation of the source causing the interrupt.</p>																																																								
5 to 0	IFCn5 to IFCn0	<p>Interrupt Factor Code This code is used to set the interrupt sources serving as DMA transfer startup factors.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th>IFCn5</th> <th>IFCn4</th> <th>IFCn3</th> <th>IFCn2</th> <th>IFCn1</th> <th>IFCn0</th> <th>Interrupt source</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>DMA request from on-chip peripheral I/O disabled</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>INTP000/INTM000</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>INTP001/INTM001</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>INTP010/INTM010</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>INTP011/INTM011</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>INTP100</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>INTP101</td> </tr> </tbody> </table>	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source	0	0	0	0	0	0	DMA request from on-chip peripheral I/O disabled	0	0	0	0	0	1	INTP000/INTM000	0	0	0	0	1	0	INTP001/INTM001	0	0	0	0	1	1	INTP010/INTM010	0	0	0	1	0	0	INTP011/INTM011	0	0	1	0	0	1	INTP100	0	0	1	0	1	0	INTP101
IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source																																																				
0	0	0	0	0	0	DMA request from on-chip peripheral I/O disabled																																																				
0	0	0	0	0	1	INTP000/INTM000																																																				
0	0	0	0	1	0	INTP001/INTM001																																																				
0	0	0	0	1	1	INTP010/INTM010																																																				
0	0	0	1	0	0	INTP011/INTM011																																																				
0	0	1	0	0	1	INTP100																																																				
0	0	1	0	1	0	INTP101																																																				

Bit Position	Bit Name	Function						
5 to 0	IFCn5 to IFCn0	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source
		0	0	1	1	0	1	INTP110
		0	1	1	0	0	1	INTCMD0
		0	1	1	0	1	0	INTCMD1
		0	1	1	0	1	1	INTCMD2
		0	1	1	1	0	0	INTCMD3
		0	1	1	1	0	1	INTCSI0
		0	1	1	1	1	0	INTSR0
		0	1	1	1	1	1	INTST0
		1	0	0	0	0	0	INTCSI1
		1	0	0	0	0	1	INTSR1
		1	0	0	0	1	0	INTST1
		1	0	0	1	1	0	INTAD
		Other than above						

Remark n = 0 to 3

The relationship between the $\overline{\text{DMARQn}}$ signal and the interrupt source that serves as a DMA transfer trigger is as follows.



Remark If an interrupt request is specified as the DMA transfer start factor, an interrupt request will be generated if DMA transfer starts. To prevent an interrupt from being generated, mask the interrupt by setting the interrupt request control register. DMA transfer starts even if an interrupt is masked.

6.4 DMA Bus States

6.4.1 Types of bus states

The DMAC bus states consist of the following 10 states.

(1) TI state

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

The $\overline{\text{DMARQ0}}$ and $\overline{\text{DMARQ1}}$ signals are sampled at the rising edge of the CLKOUT signal.

(2) T0 state

DMA transfer ready state (state in which a DMA transfer request has been issued and the bus mastership is acquired for the first DMA transfer).

(3) T1R state

The bus enters the T1R state at the beginning of a read operation in the two-cycle transfer mode.

Address driving starts. After entering the T1R state, the bus invariably enters the T2R state.

(4) T1RI state

The T1RI state is a state in which the bus waits for the acknowledge signal corresponding 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 the 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

State in which the bus is ready for DMA transfer to on-chip peripheral I/O or internal RAM (state in which the bus mastership is acquired for DMA transfer to on-chip 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 the two-cycle transfer mode.

Address driving starts. After entering the T1W state, the bus invariably enters the T2W state.

(8) T1WI state

State in which the bus waits for the acknowledge signal corresponding 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 the two-cycle transfer mode, or to a wait state.

In the last T2W state, the write strobe signal is made inactive.

(10) TE state

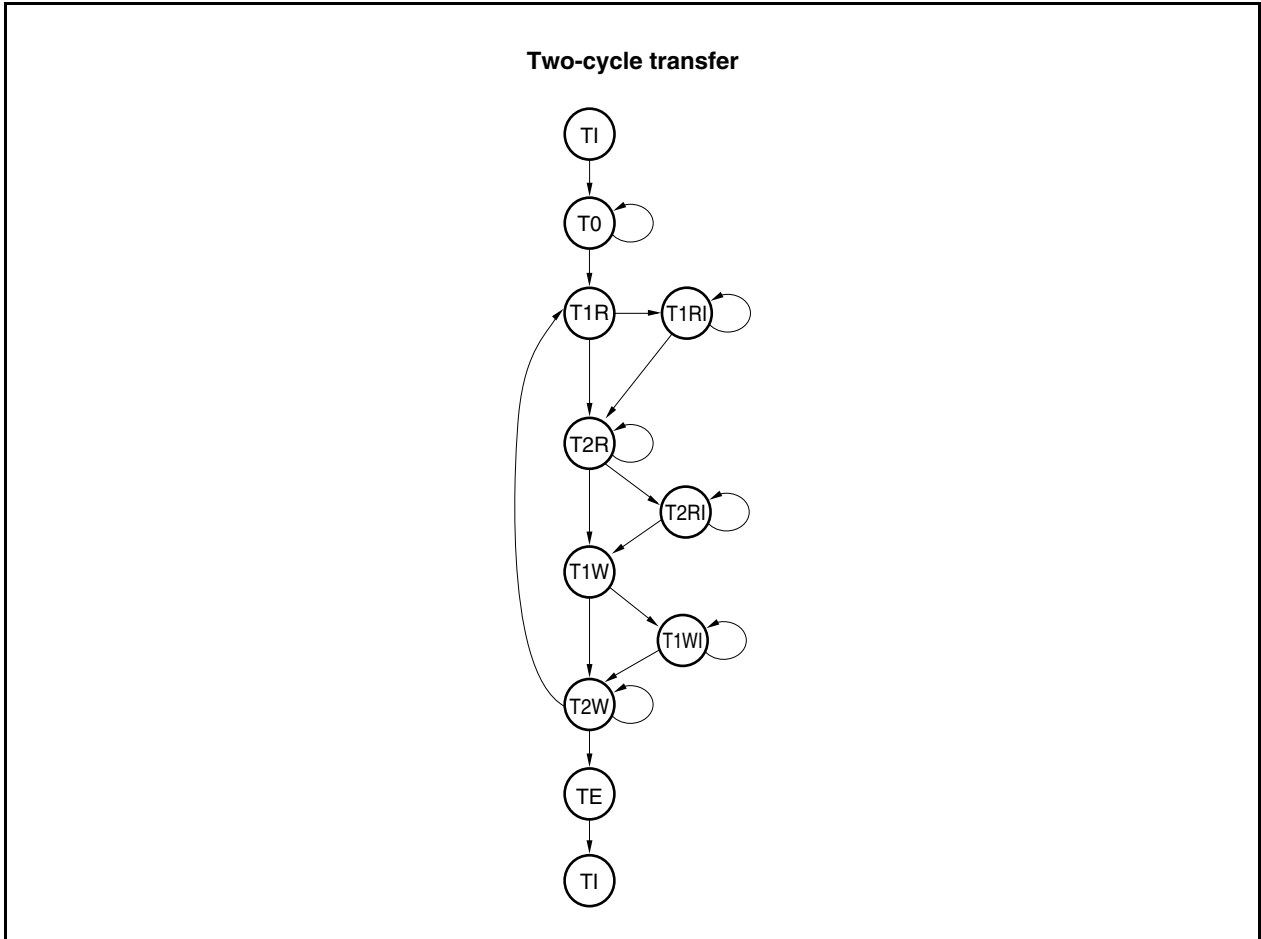
The TE state corresponds to DMA transfer completion. Various internal signals are initialized ($n = 0$ to 3).

After entering the TE state, the bus invariably enters the TI state.

6.4.2 DMAC bus cycle state transition

Except for the block transfer mode, each time the processing for a DMA transfer is completed, the bus mastership is released.

Figure 6-1. DMAC Bus Cycle State Transition



6.5 Transfer Modes

★ 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 once. 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 takes precedence. If another DMA transfer request with a lower priority occurs within one clock after single transfer has been completed, however, this request does not take precedence even if the previous DMA transfer request signal with the higher priority remains active. DMA transfer with the newly issued lower priority request is executed after the CPU bus has been released.

Figures 6-2 to 6-4 show examples of single transfer.

Figure 6-2. Single Transfer Example 1

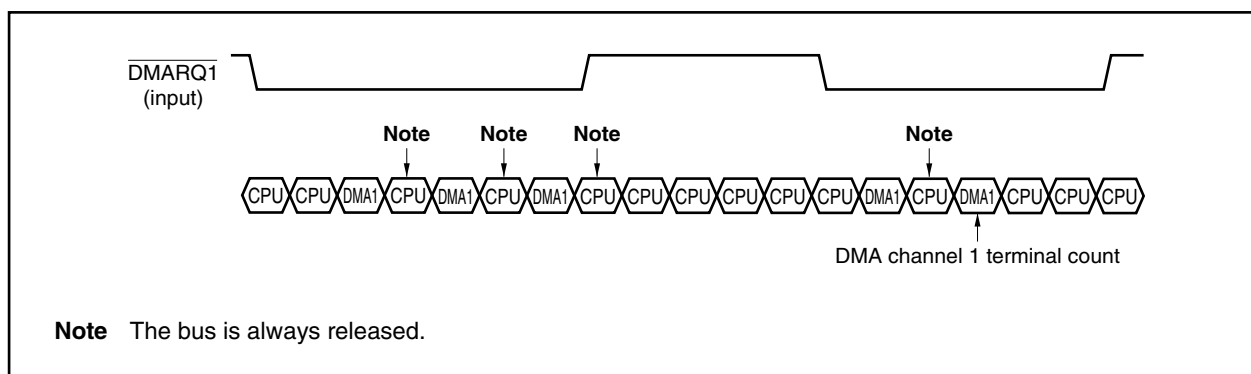


Figure 6-3 shows an example of a single transfer in which a higher priority DMA request is issued. DMA channel 0 is in the block transfer mode and channel 1 is in the single transfer mode.

Figure 6-3. Single Transfer Example 2

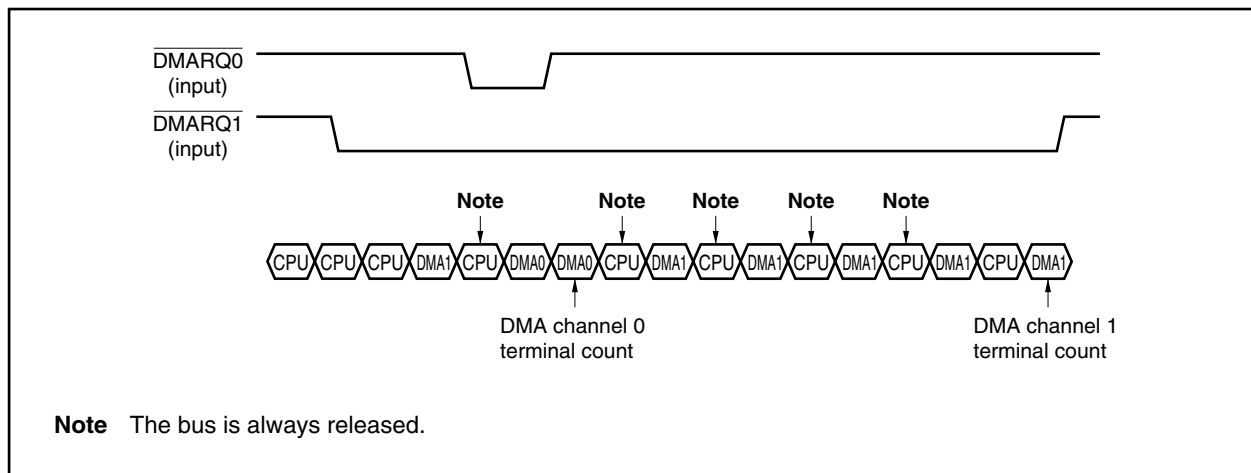
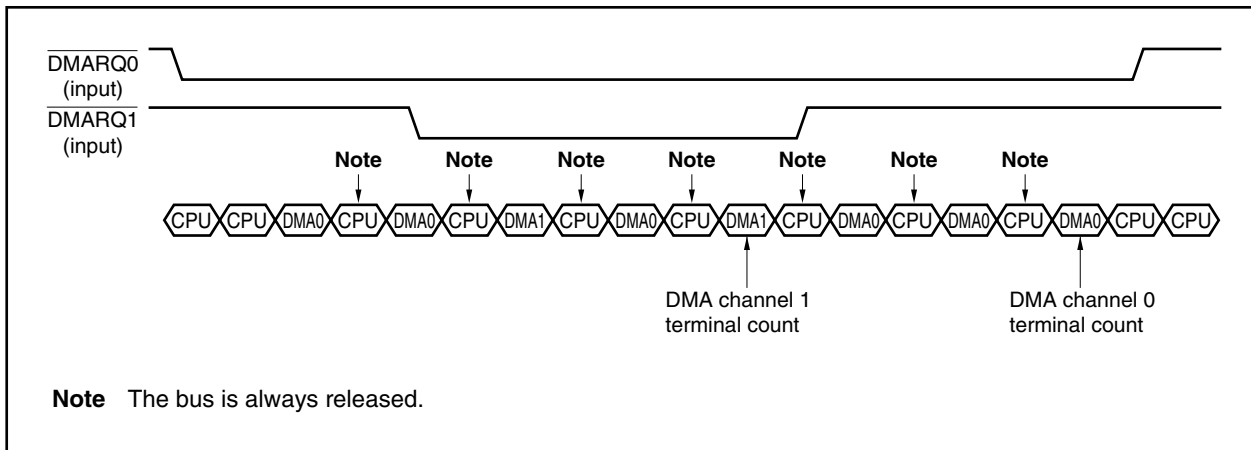


Figure 6-4 is an example of single transfer where a DMA transfer request a lower priority is issued one clock after single transfer has been completed. DMA channels 0 and 1 are used for single transfer. If two DMA transfer request signals are asserted at the same time, two DMA transfer operations are alternately executed.

Figure 6-4. Single Transfer Example 3



6.5.2 Single-step transfer mode

In single-step transfer mode, the DMAC releases the bus at each byte/halfword transfer. If there is a subsequent DMA transfer request signal ($\overline{\text{DMARQ0}}$, $\overline{\text{DMARQ1}}$), 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.

The following shows an example of a single-step transfer. Figure 6-6 shows an example of single-step transfer made in which a higher priority DMA request is issued. DMA channels 0 and 1 are in the single-step transfer mode.

Figure 6-5. Single-Step Transfer Example 1

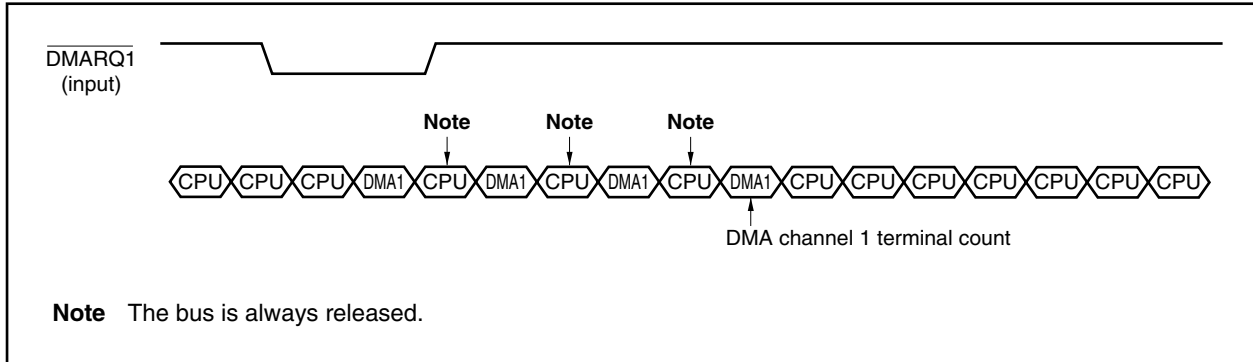
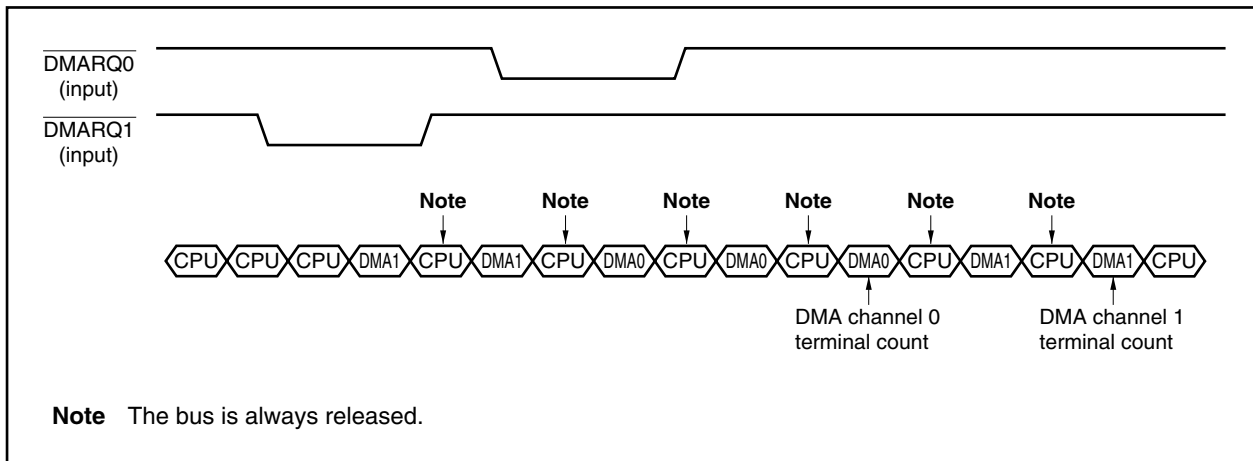


Figure 6-6. Single-Step Transfer Example 2



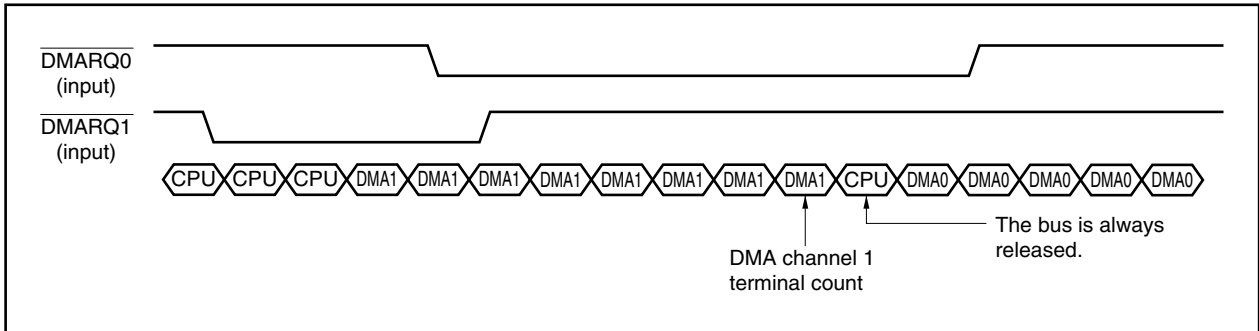
6.5.3 Block transfer mode

In the block transfer mode, once transfer starts, the DMAC continues the transfer operation without releasing the bus until a terminal count occurs. No other DMA requests are acknowledged during block transfer.

★ After the block transfer ends and the DMAC releases the bus, another DMA transfer can be acknowledged. The bus cycle of the CPU is not inserted during block transfer, but bus hold and refresh cycles are inserted in between DMA transfer operations.

The following shows an example of block transfer in which a higher priority DMA request is issued. DMA channels 0 and 1 are in the block transfer mode.

Figure 6-7. Block Transfer Example



6.6 Two-Cycle Transfer

In two-cycle transfer, data transfer is performed in two cycles, a read cycle (source to DMAC) and a write cycle (DMAC to destination).

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

★ **Caution** An idle cycle of 1 clock is always inserted between the read cycle and write cycle.

Figure 6-8. Timing of Access to SRAM, External ROM, and External I/O During 2-Cycle DMA Transfer (1/2)

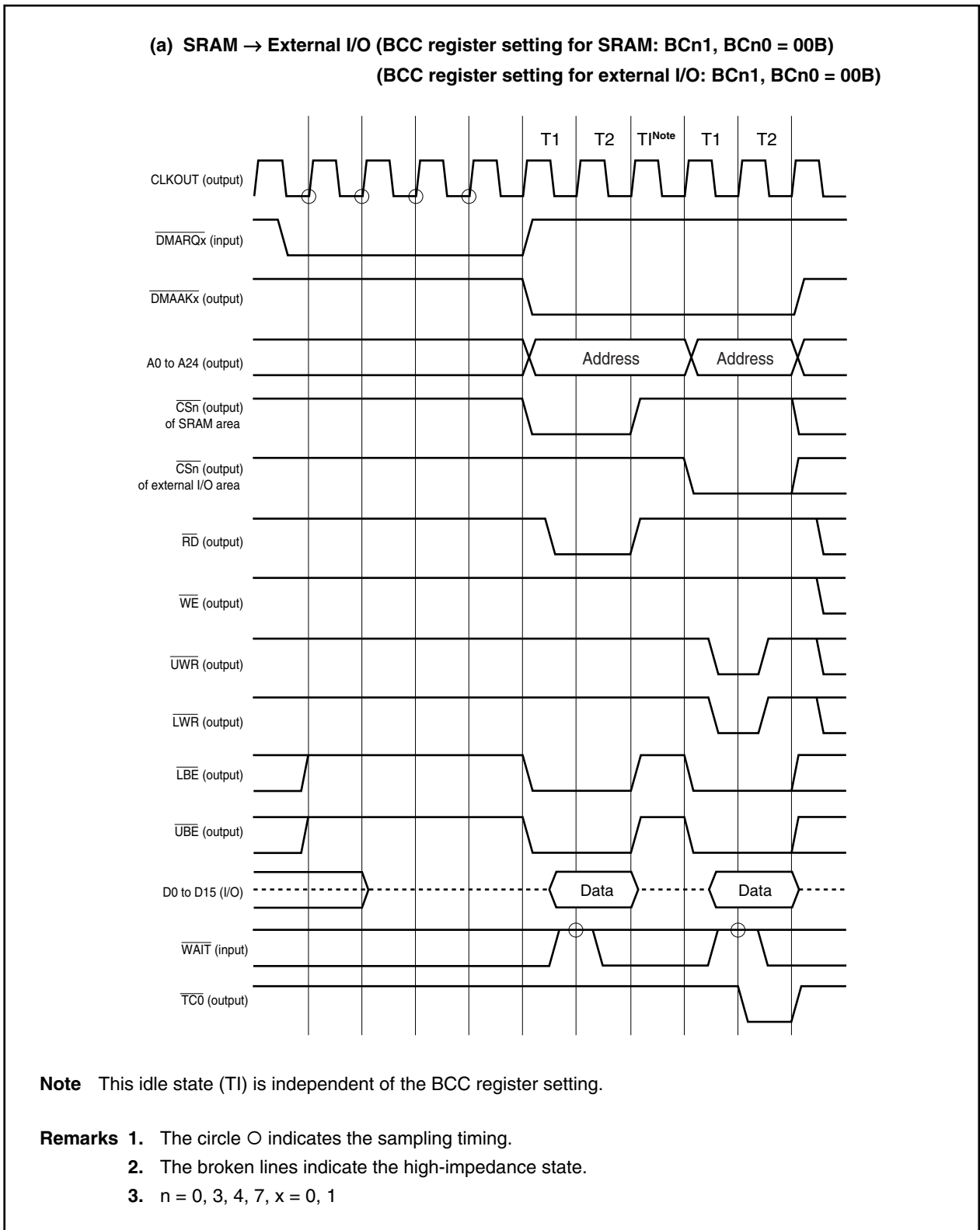
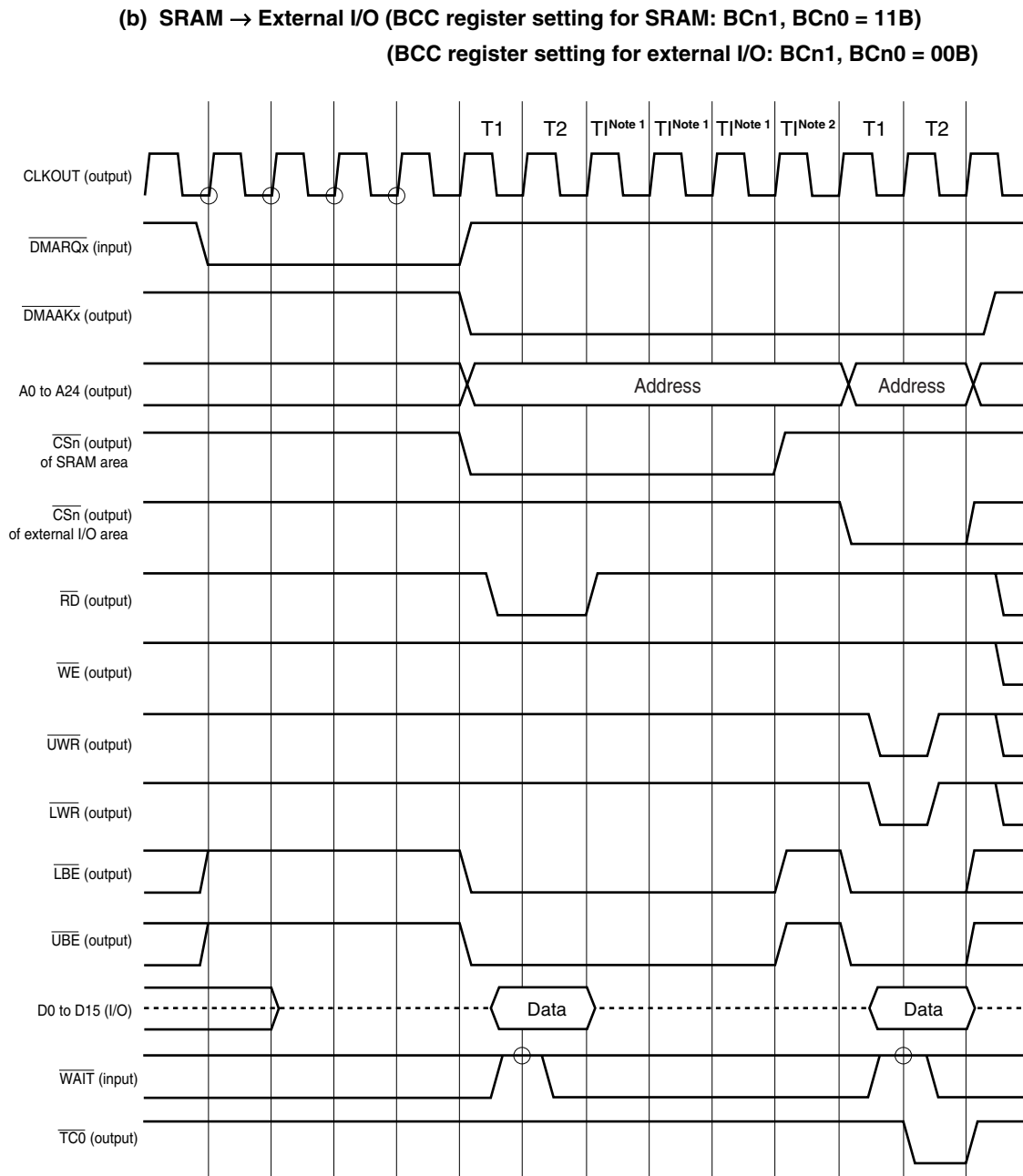


Figure 6-8. Timing of Access to SRAM, External ROM, and External I/O During 2-Cycle DMA Transfer (2/2)



- Notes 1.** This idle state (TI) is inserted by means of a BCC register setting.
2. This idle state (TI) is independent of the BCC register setting.

- Remarks 1.** The circle ○ indicates the sampling timing.
2. The broken lines indicate the high-impedance state.
3. n = 0, 3, 4, 7, x = 0, 1

Figure 6-9. Timing of 2-Cycle DMA Transfer (External I/O → SRAM)

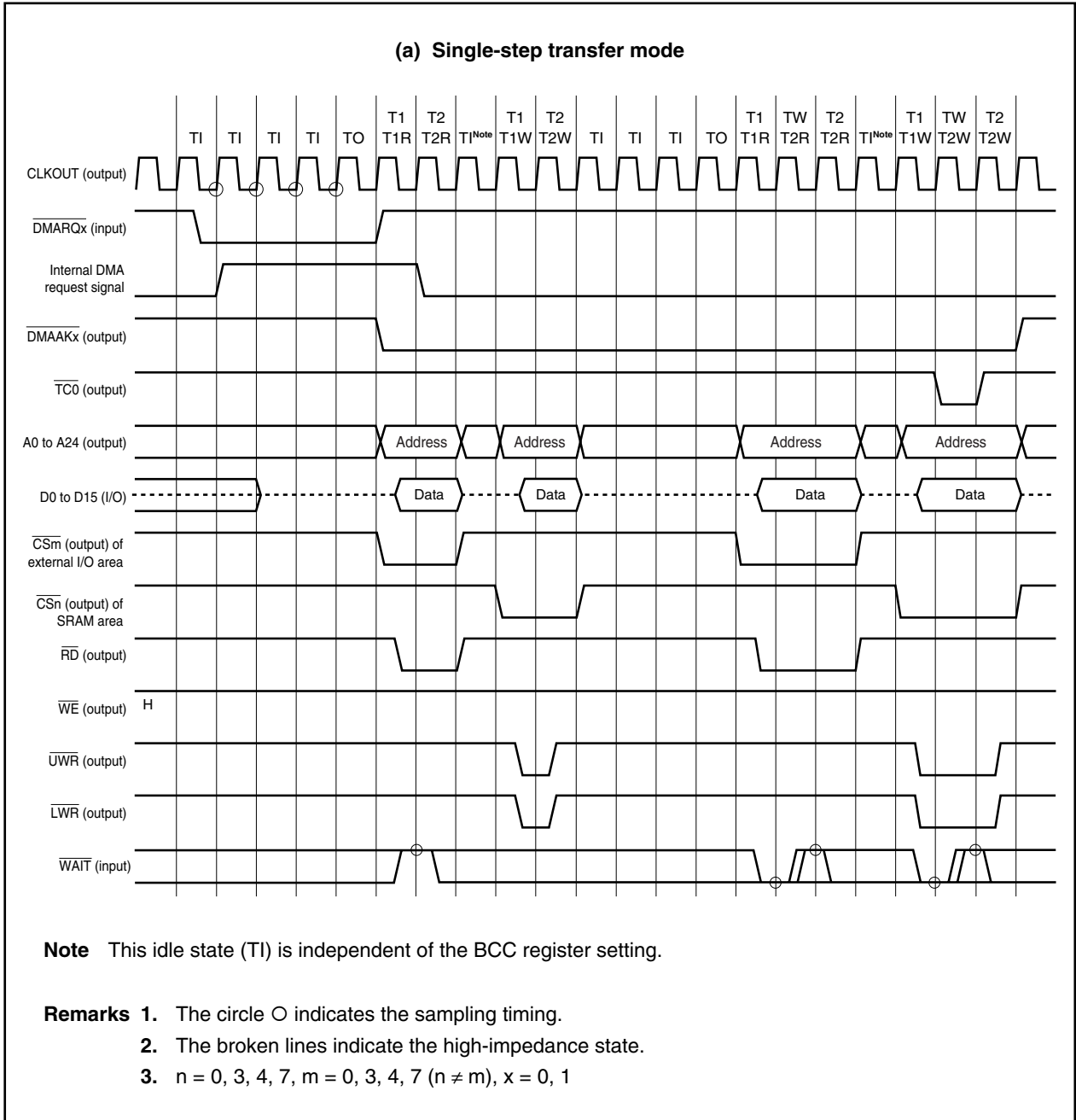


Figure 6-10. Timing of 2-Cycle DMA Transfer (SRAM → SDRAM) (1/3)

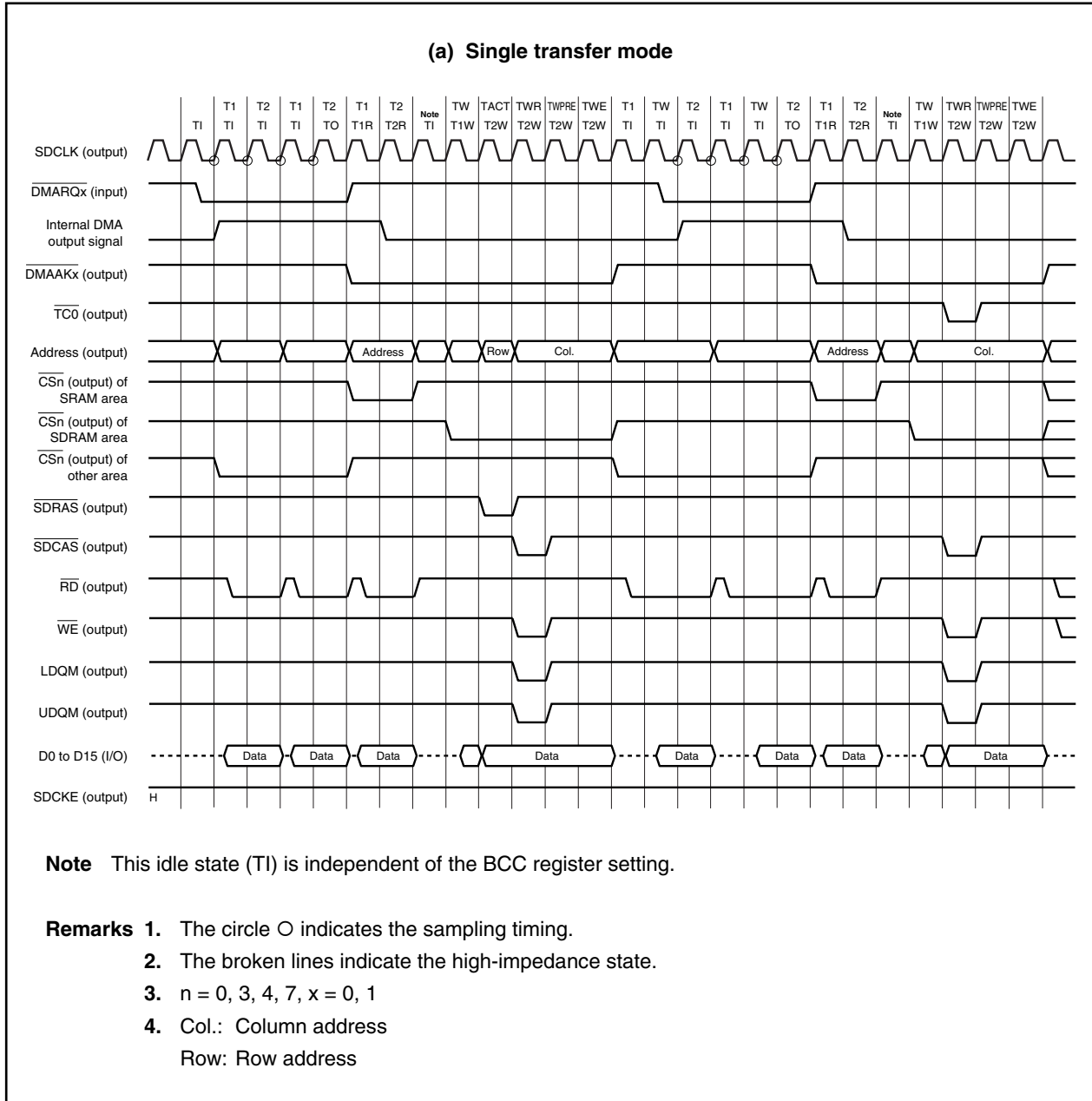


Figure 6-10. Timing of 2-Cycle DMA Transfer (SRAM → SDRAM) (2/3)

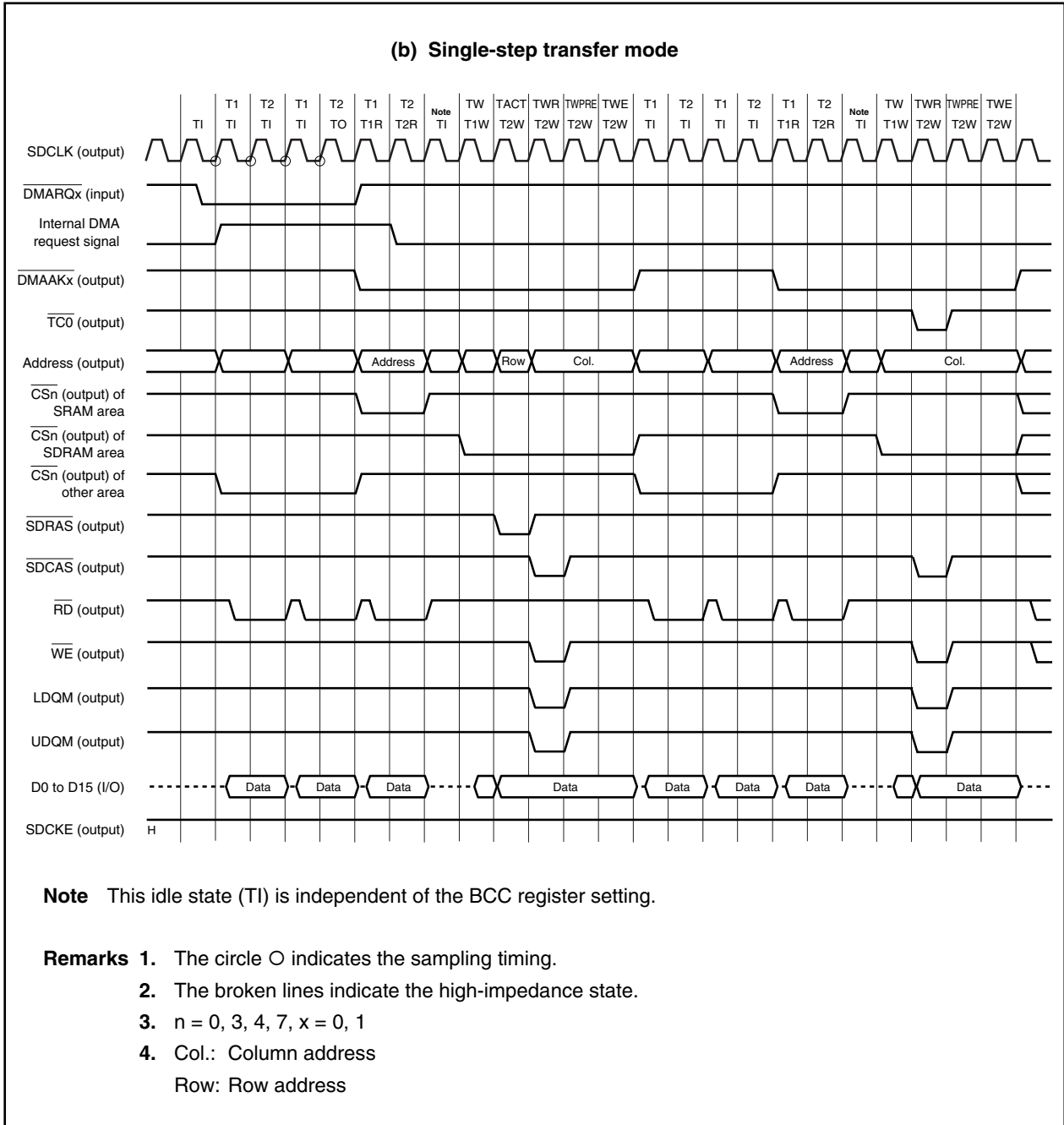


Figure 6-10. Timing of 2-Cycle DMA Transfer (SRAM → SDRAM) (3/3)

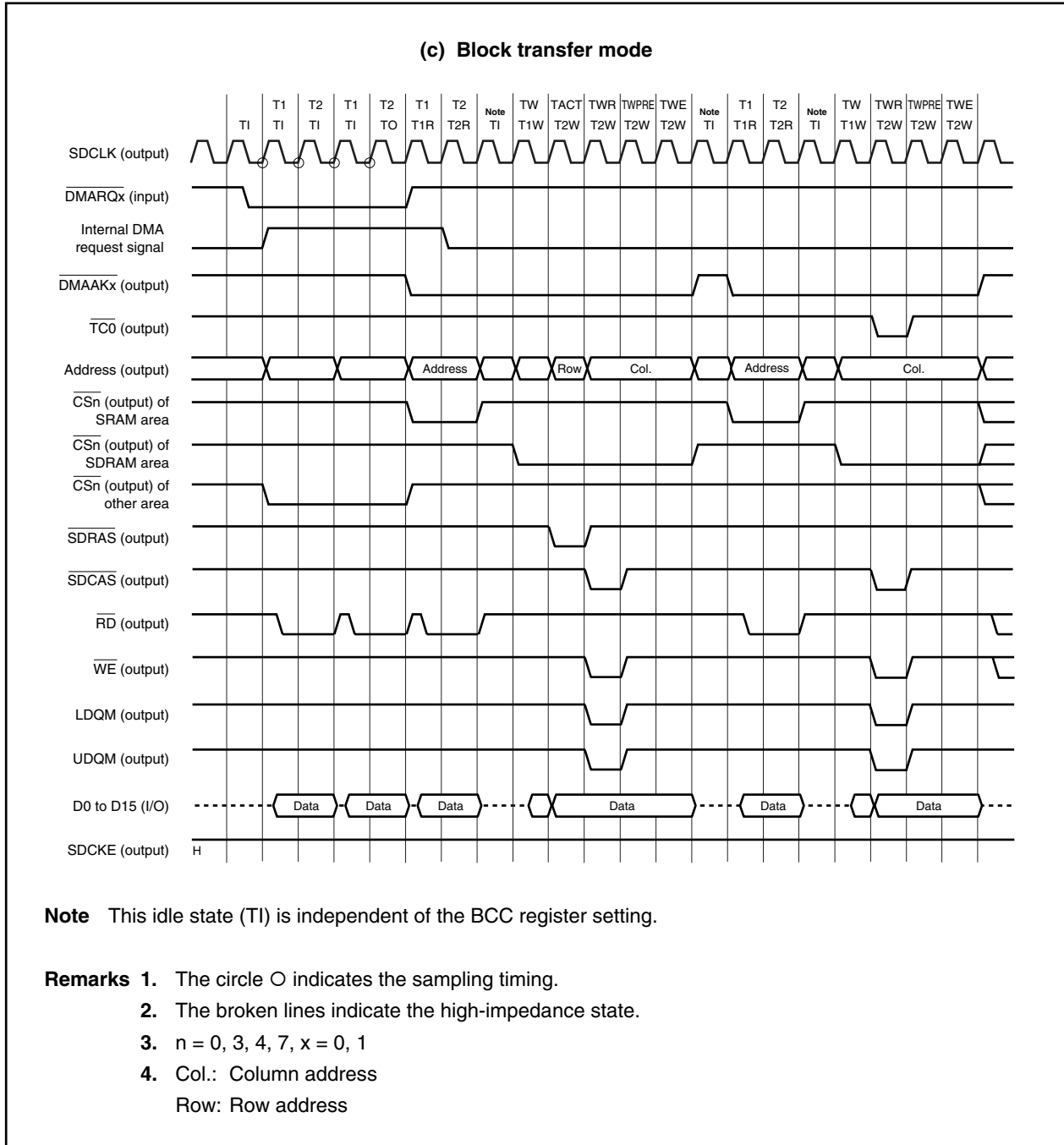


Figure 6-11. Timing of 2-Cycle DMA Transfer (SDRAM → SRAM) (1/3)

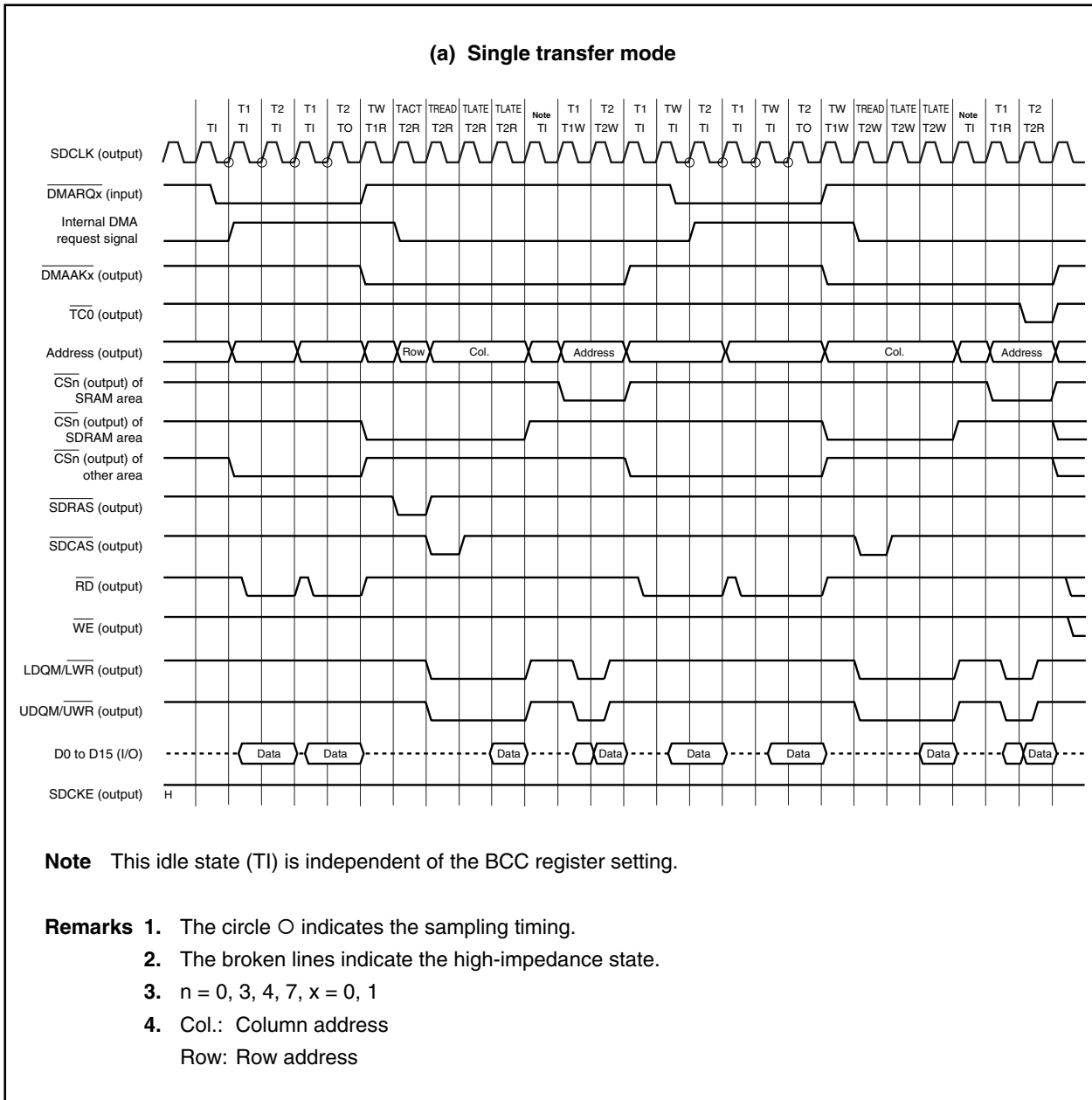


Figure 6-11. Timing of 2-Cycle DMA Transfer (SDRAM → SRAM) (2/3)

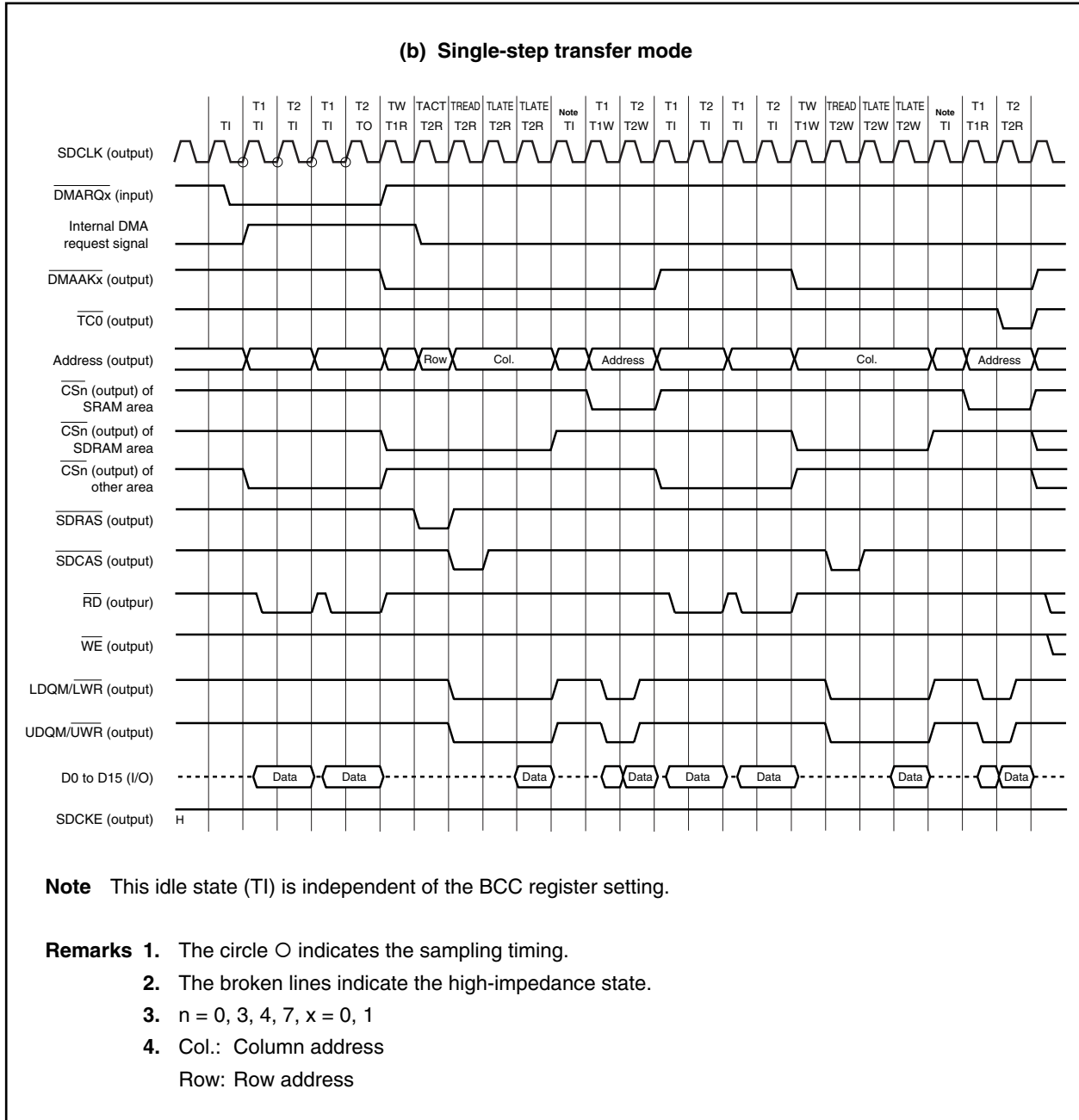
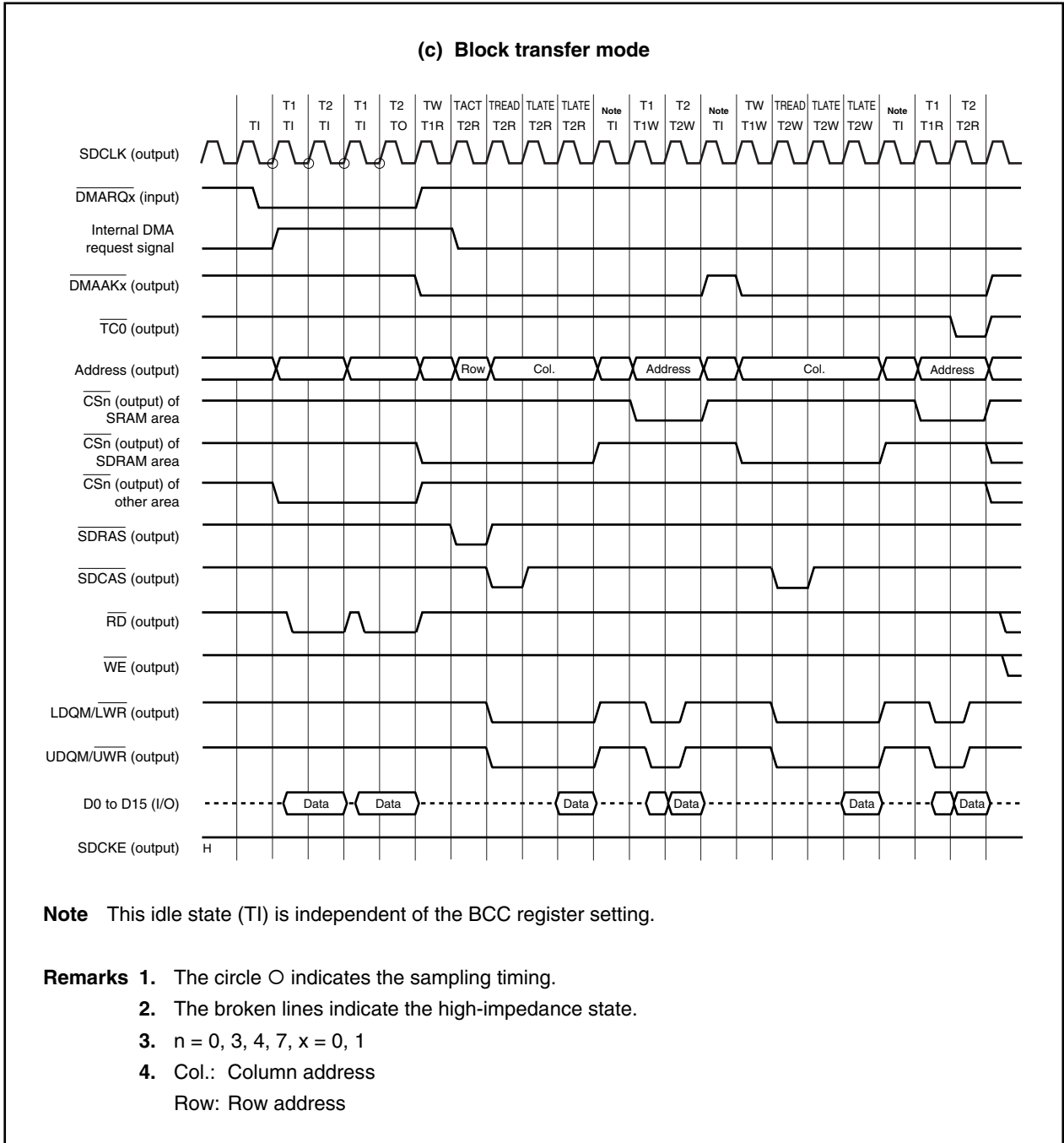


Figure 6-11. Timing of 2-Cycle DMA Transfer (SDRAM → SRAM) (3/3)



6.7 Transfer Object

6.7.1 Transfer type and transfer object

Table 6-1 lists the relationships between transfer type and transfer object. The mark “√” means “transfer possible”, and the mark “-” means “transfer impossible”.

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

		Transfer Destination (Two-Cycle Transfer)			
		On-chip peripheral I/O	External I/O	Internal RAM	External memory
Source	On-chip peripheral I/O	√	√	√	√
	External I/O	√	√	√	√
	Internal RAM	√	√	-	√
	External memory	√	√	√	√

Cautions 1. The operation is not guaranteed for combinations of transfer destination and source marked with “-” in Table 6-1.

2. Addresses between 3FFF000H and 3FFFFFFH cannot be specified for the source and destination address of DMA transfer. Be sure to specify an address between FFFF000H and FFFFFFFH.

Remarks 1. During 2-cycle DMA transfer, if the data bus width of the transfer source and that of the transfer destination are different, the operation becomes as follows.

<16-bit transfer>

- Transfer from a 16-bit bus to an 8-bit bus
A read cycle (16 bits) is generated and then a write cycle (8 bits) is generated twice successively.
- Transfer from an 8-bit bus to a 16-bit bus
A read cycle (8 bits) is generated twice successively and then a write cycle (16 bits) is generated.

<8-bit transfer>

- Transfer from 16-bit bus to 8-bit bus
A read cycle (the higher 8 bits go into a high-impedance state) is generated and then a write cycle (8 bits) is generated.
- Transfer from 8-bit bus to 16-bit bus
A read cycle (8 bits) is generated and then a write cycle is generated (the higher 8 bits go into a high-impedance state). Data is written in the order of lower bits to higher bits to the transfer destination in the case of little endian and in the reverse order in the case of big endian.

2. Transfer between the little endian area and the big endian area is also possible.

6.7.2 External bus cycles during DMA transfer

The external bus cycles during DMA transfer are shown below.

Table 6-2. External Bus Cycles During DMA Transfer

Transfer Type	Transfer Object	External Bus Cycle	
		None ^{Note}	
2-cycle transfer	On-chip peripheral I/O, internal RAM	None ^{Note}	-
	External I/O	Yes	SRAM cycle
	External memory	Yes	Memory access cycle set by the BCT register

Note Other external 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 the block transfer mode, the channel used for transfer is never switched.

In the 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 acknowledged.

★ **Caution** Do not start two or more DMA channels with the same factor. If two or more DMA channels are started with the same factor, a DMA channel with a lower priority may be acknowledged before a DMA channel with a higher priority.

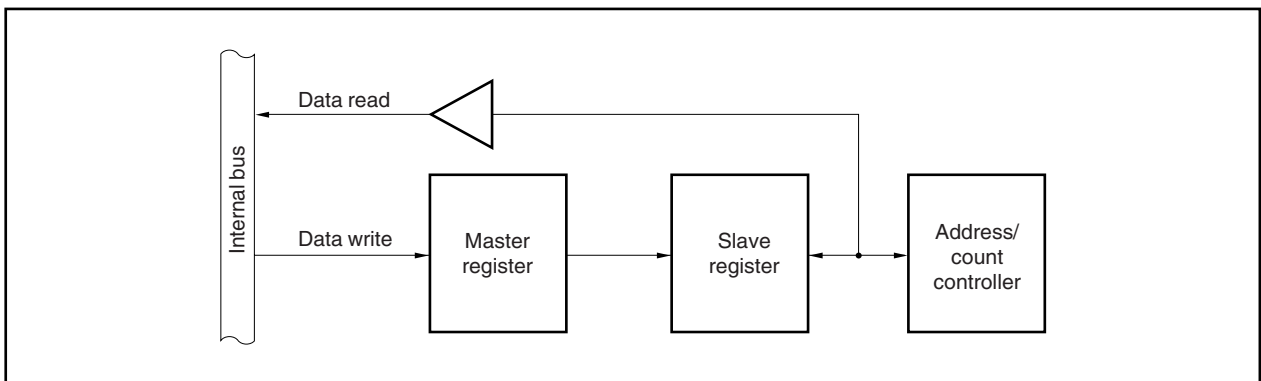
6.9 Next Address Setting Function

The DMA source address registers (DSAnH, DSAnL), DMA destination address registers (DDAnH, DDAnL), and DMA transfer 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 automatically rewritten with the value that was set immediately before.

Therefore, during DMA transfer, transfer is automatically started when a new DMA transfer setting is made for these registers and the Enn bit of the DCHCn register, and MLEn bit is set to 1 (however, the DMA transfer end interrupt may be issued even if DMA transfer is automatically started).

Figure 6-12 shows the configuration of the buffer register.

Figure 6-12. 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 ($\overline{\text{DMARQn}}$)

Requests from the $\overline{\text{DMARQn}}$ pin are sampled each time the $\overline{\text{CLKOUT}}$ signal rises ($n = 0, 1$).

Hold the request from $\overline{\text{DMARQn}}$ pin until the corresponding $\overline{\text{DMAAKn}}$ signal becomes active.

If a state whereby the Enn bit of the DCHCn register = 1 and the TCn bit = 0 ($n = 0$ to 3) is set, the $\overline{\text{DMARQn}}$ signal ($n = 0, 1$) in the T1 state becomes valid. If the $\overline{\text{DMARQn}}$ signal becomes active in the T1 state, it changes to the T0 state and DMA transfer is started.

(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 on-chip 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 on-chip peripheral I/O that is set in the DTFRn register, DMA transfer starts ($n = 0$ to 3).

- Enn bit = 1
- TCn bit = 0

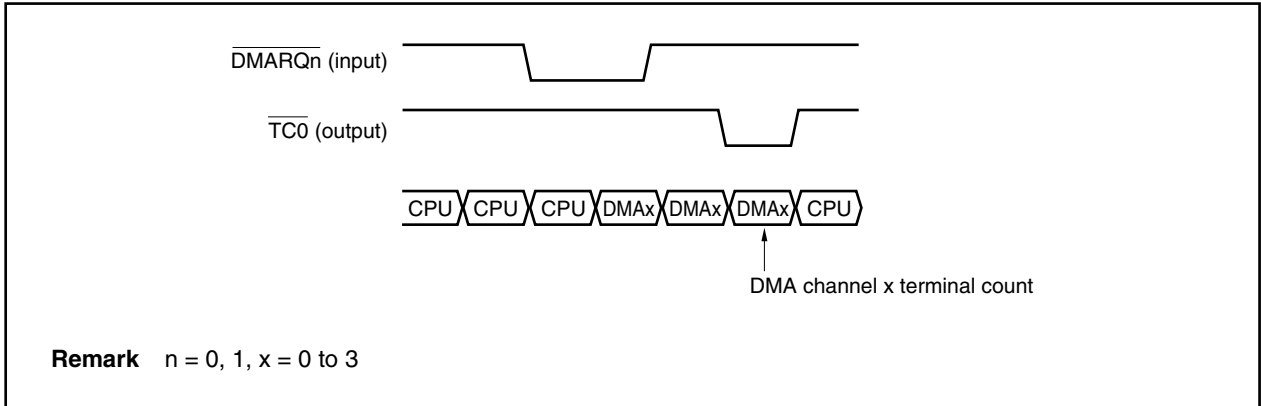
Remark When the $\overline{\text{DMARQn}}$ signal is used for the DMA start trigger, DMA transfer request is level-detected since the $\overline{\text{DMARQn}}$ signal is detected at the level ($n = 0, 1$). Edge-detection for DMA transfer request is possible, however, by not using the $\overline{\text{DMARQn}}$ signal but using an external interrupt request for the DMA start trigger.

6.11 Terminal Count Output upon DMA Transfer End

The terminal count signal ($\overline{TC0}$) becomes active for one clock during the last DMA transfer cycle.

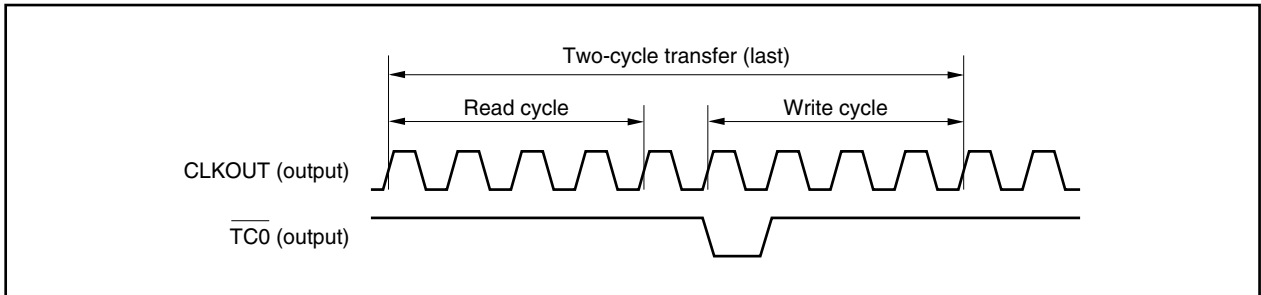
In two-cycle transfer, the $\overline{TC0}$ signal becomes active for one clock at the start of the write cycle of the last DMA transfer.

Figure 6-13. Terminal Count Signal ($\overline{TC0}$) Timing Example



★

Figure 6-14. Example of Terminal Count Signal ($\overline{TC0}$) Output



6.12 Forcible Interrupt

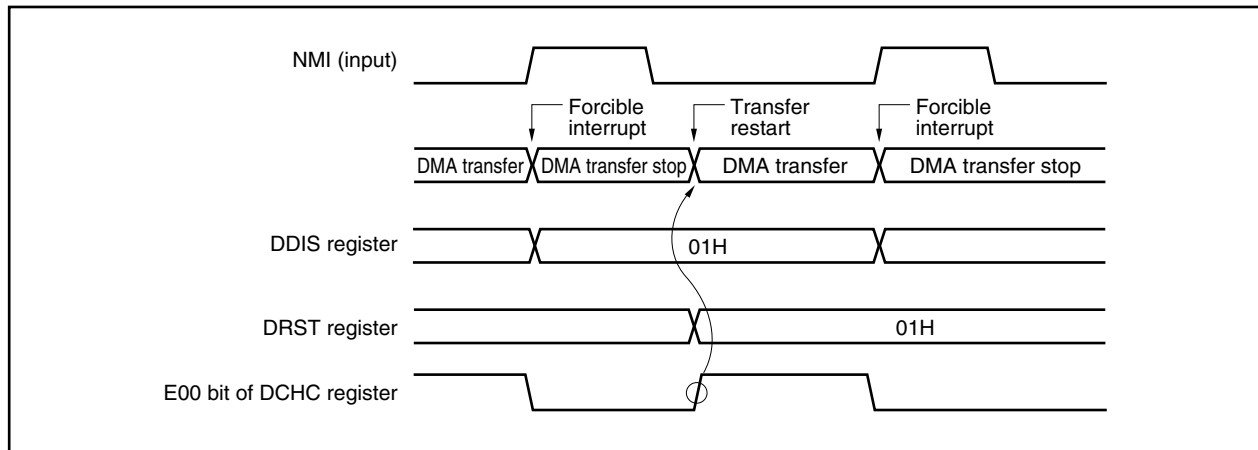
DMA transfer can be forcibly interrupted by NMI input during DMA transfer.

At such a time, the DMAC resets the Enn bit of the DCHCn register of all channels to 0 and the DMA transfer disabled state is entered. An NMI request can then be acknowledged after the DMA transfer executed during NMI input is terminated ($n = 0$ to 3).

In the single-step transfer mode or block transfer mode, the DMA transfer request is held in the DMAC. If the Enn bit is set to 1, DMA transfer restarts from the point where it was interrupted.

In the single transfer mode, if the Enn bit is set to 1, the next DMA transfer request is received and DMA transfer starts.

Figure 6-15. Example of Forcible Interrupt of DMA Transfer

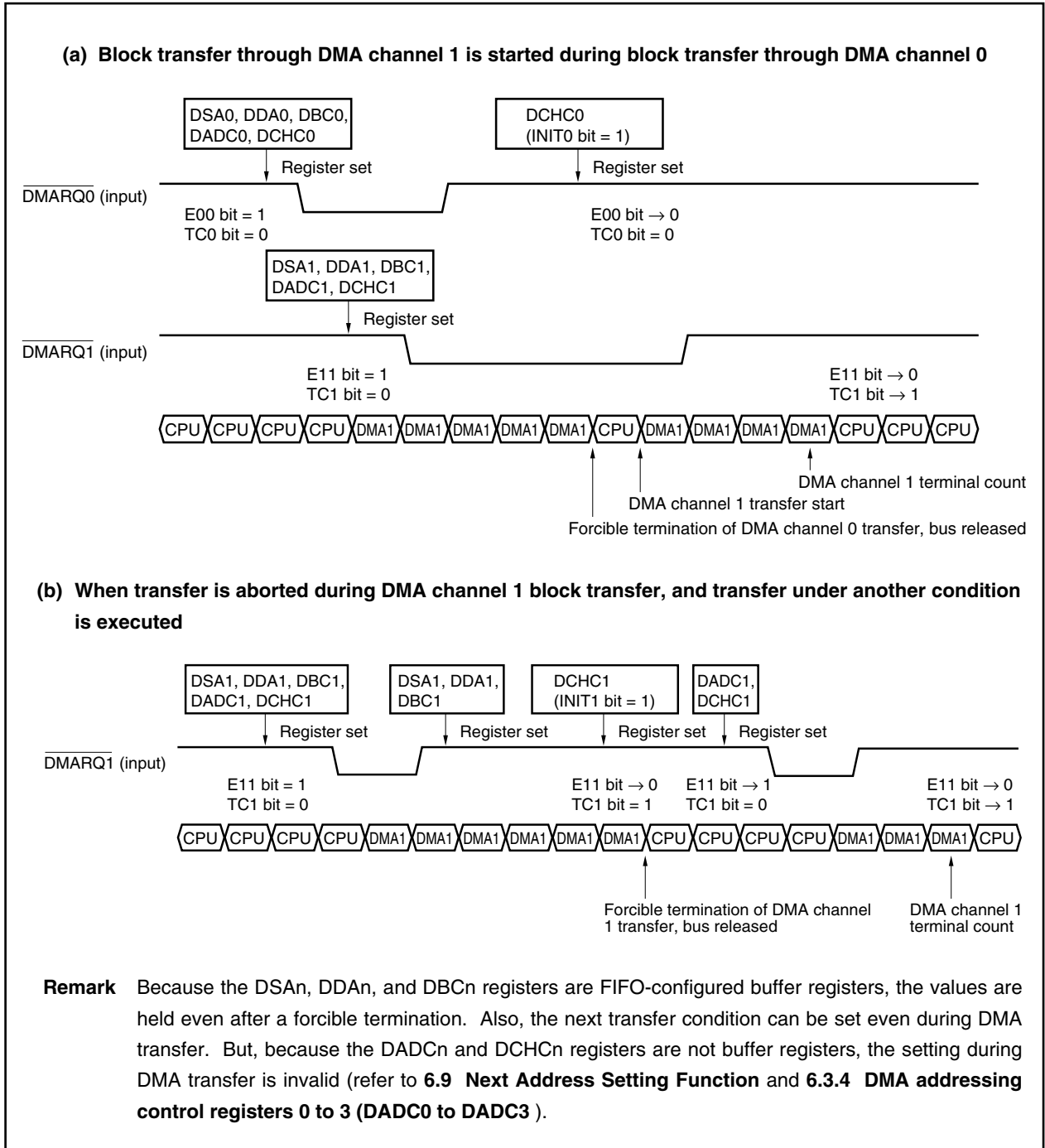


6.13 Forcible Termination

DMA transfer can be forcibly terminated by the INITn bit of the DCHCn register, in addition to the forcible interrupt operation by means of NMI input (n = 0 to 3).

An example of forcible termination by the INITn bit of the DCHCn register is illustrated below (n = 0 to 3).

Figure 6-16. Example of Forcible Termination of DMA Transfer



★

6.14 Times Related to DMA Transfer

The overhead before and after DMA transfer and minimum execution clock for DMA transfer are shown below. In the case of external memory access, the time depends on the type of external memory connected.

Table 6-3. Number of Minimum Execution Clocks in DMA Cycle

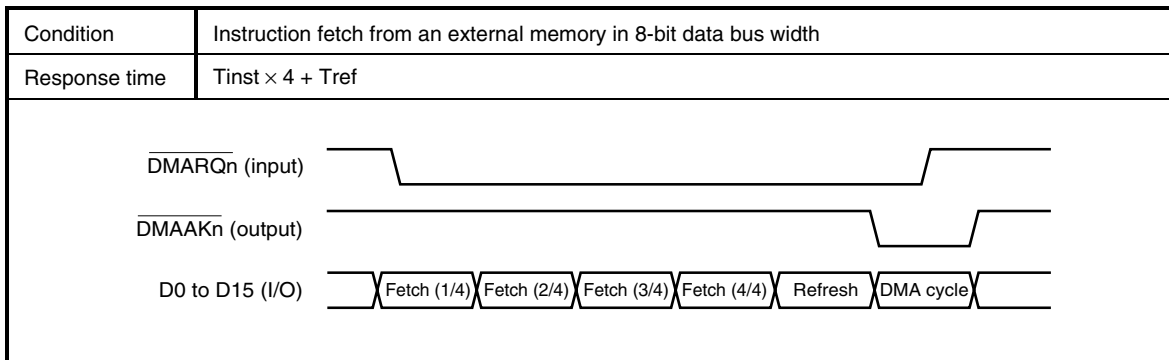
DMA Cycle	Number of Minimum Execution Clocks
From $\overline{\text{DMARQn}}$ signal acknowledgement to $\overline{\text{DMAAKn}}$ signal falling	4 clocks
External memory access	Depends on the memory connected.
Internal RAM access	1 clock

Remark n = 0, 1

6.15 Maximum Response Time for DMA Transfer Request

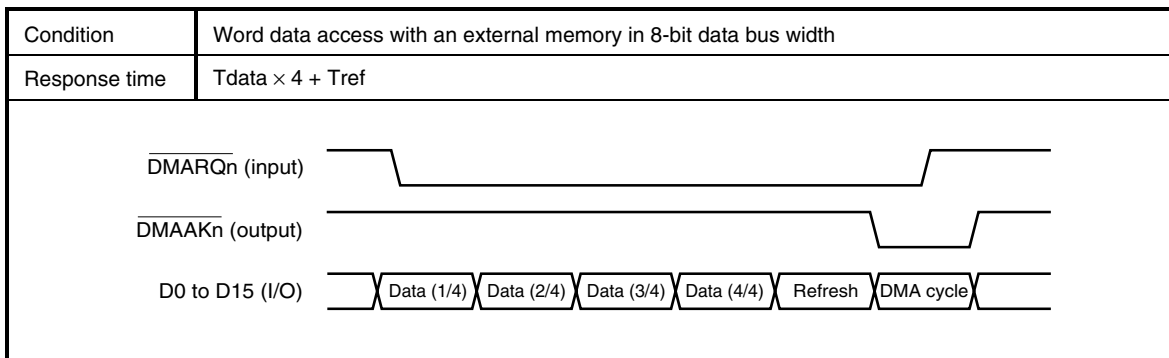
The response time for a DMA transfer request becomes the longest under the following conditions (in the SDRAM refresh cycle enabled state). However, the case when a higher priority DMA transfer is generated is excluded.

(1) Condition 1



Remark n = 0, 1

(2) Condition 2



Remark n = 0, 1

(3) Condition 3

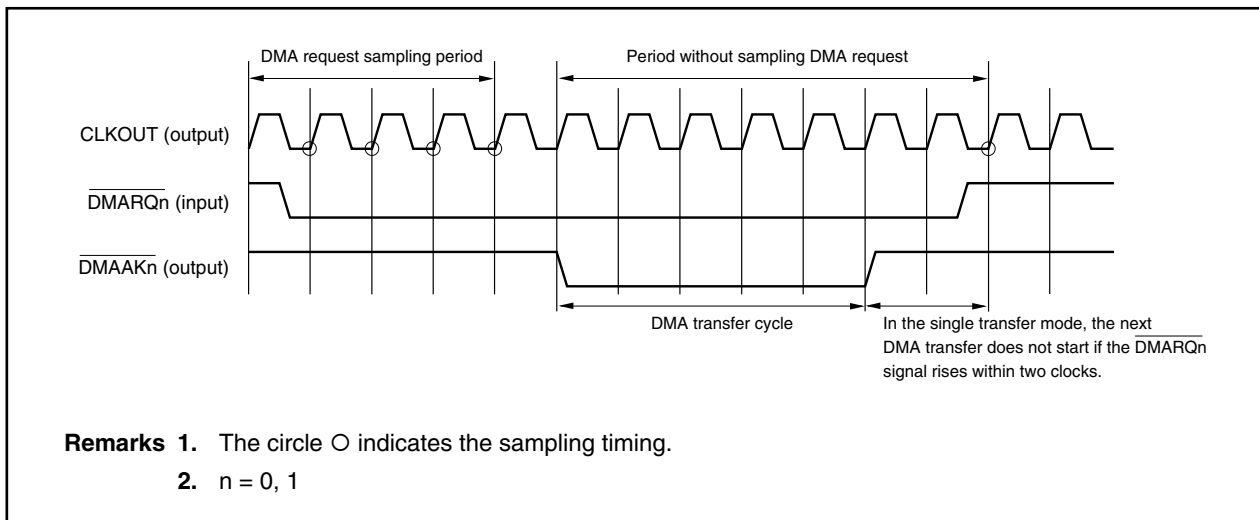
Condition	Instruction fetch from an external memory in 8-bit data bus width Execution of a bit manipulation instruction (SET1, CLR1, or NOT1)
Response time	$T_{inst} \times 4 + T_{data} \times 2 + T_{ref}$

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

★ **6.16 One-Time Transfer During Single Transfer via $\overline{DMARQ0}$, $\overline{DMARQ1}$ Signals**

The \overline{DMARQn} signal is sampled at the rising edge of the third clock after the DMA transfer cycle in the single-transfer mode has been completed. To perform transfer only one-time when single transfer is executed for an external memory via the \overline{DMARQn} signal, the \overline{DMARQn} signal must be made inactive within 2 clocks from when the \overline{DMAAKn} signal becomes inactive ($n = 0, 1$).

★ **Figure 6-17. Time to Perform Single Transfer One Time**



6.17 Cautions

(1) Memory boundary

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

(2) Transfer of Misaligned Data

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

(3) Bus arbitration for CPU

When an external device is targeted for DMA transfer, the CPU can access the internal RAM (if they are not subject to DMA transfer).

★ (4) Maintenance of $\overline{\text{DMARQn}}$ signal

Be sure to maintain the $\overline{\text{DMARQn}}$ signal at the active level until the $\overline{\text{DMAAKn}}$ signal becomes active ($n = 0, 1$).

If the $\overline{\text{DMARQn}}$ signal becomes inactive before the $\overline{\text{DMAAKn}}$ signal becomes active, DMA transfer may not be performed.

(5) $\overline{\text{DMAAKn}}$ signal output

When the transfer object is internal RAM, the $\overline{\text{DMAAKn}}$ signal is not output during a DMA cycle for internal RAM (for example, if 2-cycle transfer is performed from internal RAM to an external memory, the $\overline{\text{DMAAKn}}$ signal is output only during a DMA write cycle for the external memory) ($n = 0, 1$).

★ (6) DMA start factors

Do not start two or more DMA channels with the same factor. If two or more DMA channels are started with the same factor, a DMA channel with a lower priority may be acknowledged before a DMA channel with a higher priority.

6.17.1 Interrupt factors

DMA transfer is interrupted if the following factors are issued.

- Bus hold
- Refresh cycle

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

6.18 DMA Transfer End

When DMA transfer ends and the TCn bit of the DCHCn register is set to 1, a DMA transfer end interrupt (INTDMAn) is issued to the interrupt controller (INTC) ($n = 0$ to 3).

CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V850E/MA2 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 27 interrupt requests.

An interrupt is an event that occurs independently of program execution, and an exception is an event whose occurrence is dependent on program execution.

The V850E/MA2 can process interrupt requests from the on-chip peripheral hardware and external sources. Moreover, exception processing can be started by the TRAP instruction (software exception) or by generation of an exception event (i.e. fetching of an illegal opcode) (exception trap).

7.1 Features

- Interrupts
 - Non-maskable interrupts: 1 source
 - Maskable interrupts: 26 sources
 - 8 levels of programmable priorities (maskable interrupts)
 - Multiple interrupt control according to priority
 - Masks can be specified for each maskable interrupt request.
 - Noise elimination, edge detection, and valid edge specification for external interrupt request signals.
- Exceptions
 - Software exceptions: 32 sources
 - Exception traps: 2 sources (illegal opcode exception and debug trap)

Interrupt/exception sources are listed in Table 7-1.

Table 7-1. Interrupt/Exception Source List

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Generating Source	Generating Unit				
Reset	Interrupt	RESET	–	Reset input	–	–	0000H	00000000H	Undefined
Non-maskable	Interrupt	NMI0	–	NMI input	–	–	0010H	0000010H	nextPC
Software exception	Exception	TRAP0n ^{Note}	–	TRAP instruction	–	–	004nH ^{Note}	00000040H	nextPC
	Exception	TRAP1n ^{Note}	–	TRAP instruction	–	–	005nH ^{Note}	00000050H	nextPC
Exception trap	Exception	ILGOP/ DBG0	–	Illegal opcode/ DBTRAP instruction	–	–	0060H	00000060H	nextPC
Maskable	Interrupt	INTOV00	OVIC00	Timer 00 overflow	RPU	0	0080H	00000080H	nextPC
	Interrupt	INTOV01	OVIC01	Timer 01 overflow	RPU	1	0090H	00000090H	nextPC
	Interrupt	INTP000/ INTM000	P00IC0	Match of INTP000 pin/CCC00	Pin/RPU	4	00C0H	000000C0H	nextPC
	Interrupt	INTP001/ INTM001	P00IC1	Match of INTP001 pin/CCC01	Pin/RPU	5	00D0H	000000D0H	nextPC
	Interrupt	INTP010/ INTM010	P01IC0	Match of INTP010 pin/CCC10	Pin/RPU	6	00E0H	000000E0H	nextPC
	Interrupt	INTP011/ INTM011	P01IC1	Match of INTP011 pin/CCC11	Pin/RPU	7	00F0H	000000F0H	nextPC
	Interrupt	INTP100	P10IC0	INTP100 pin	Pin	12	0140H	00000140H	nextPC
	Interrupt	INTP101	P10IC1	INTP101 pin	Pin	13	0150H	00000150H	nextPC
	Interrupt	INTP110	P11IC0	INTP110 pin	Pin	16	0180H	00000180H	nextPC
	Interrupt	INTCMD0	CMICD0	CMD0 match signal	RPU	28	0240H	00000240H	nextPC
	Interrupt	INTCMD1	CMICD1	CMD1 match signal	RPU	29	0250H	00000250H	nextPC
	Interrupt	INTCMD2	CMICD2	CMD2 match signal	RPU	30	0260H	00000260H	nextPC
	Interrupt	INTCMD3	CMICD3	CMD3 match signal	RPU	31	0270H	00000270H	nextPC
	Interrupt	INTDMA0	DMAIC0	DMA0 transfer completion	DMA	32	0280H	00000280H	nextPC
	Interrupt	INTDMA1	DMAIC1	DMA1 transfer completion	DMA	33	0290H	00000290H	nextPC
	Interrupt	INTDMA2	DMAIC2	DMA2 transfer completion	DMA	34	02A0H	000002A0H	nextPC
	Interrupt	INTDMA3	DMAIC3	DMA3 transfer completion	DMA	35	02B0H	000002B0H	nextPC
	Interrupt	INTCSI0	CSIIC0	CSI0 transmission/ reception completion	SIO	36	02C0H	000002C0H	nextPC
	Interrupt	INTSER0	SEIC0	UART0 reception error	SIO	37	02D0H	000002D0H	nextPC
	Interrupt	INTSR0	SRIC0	UART0 reception completion	SIO	38	02E0H	000002E0H	nextPC
	Interrupt	INTST0	STIC0	UART0 transmission completion	SIO	39	02F0H	000002F0H	nextPC
	Interrupt	INTCSI1	CSIIC1	CSI1 transmission/ reception completion	SIO	40	0300H	00000300H	nextPC
	Interrupt	INTSER1	SEIC1	UART1 reception error	SIO	41	0310H	00000310H	nextPC
	Interrupt	INTSR1	SRIC1	UART1 reception completion	SIO	42	0320H	00000320H	nextPC
	Interrupt	INTST1	STIC1	UART1 transmission completion	SIO	43	0330H	00000330H	nextPC
	Interrupt	INTAD	ADIC	A/D convert completion	ADC	48	0380H	00000380H	nextPC

Note n = 0 to FH

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 saved when an interrupt is acknowledged during division (DIV, DIVH, DIVU, DIVHU) instruction execution is the value of the PC of the current instruction (DIV, DIVH, DIVU, DIVHU).

nextPC: The PC value that starts the processing following interrupt/exception processing.

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

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 the 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 request 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 is cleared 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 exception code 0010H to the higher halfword (FECC) of ECR.
- <4> Sets the NP and ID bits of the 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 servicing configuration of a non-maskable interrupt is shown in Figure 7-1.

Figure 7-1. Servicing Configuration of Non-Maskable Interrupt

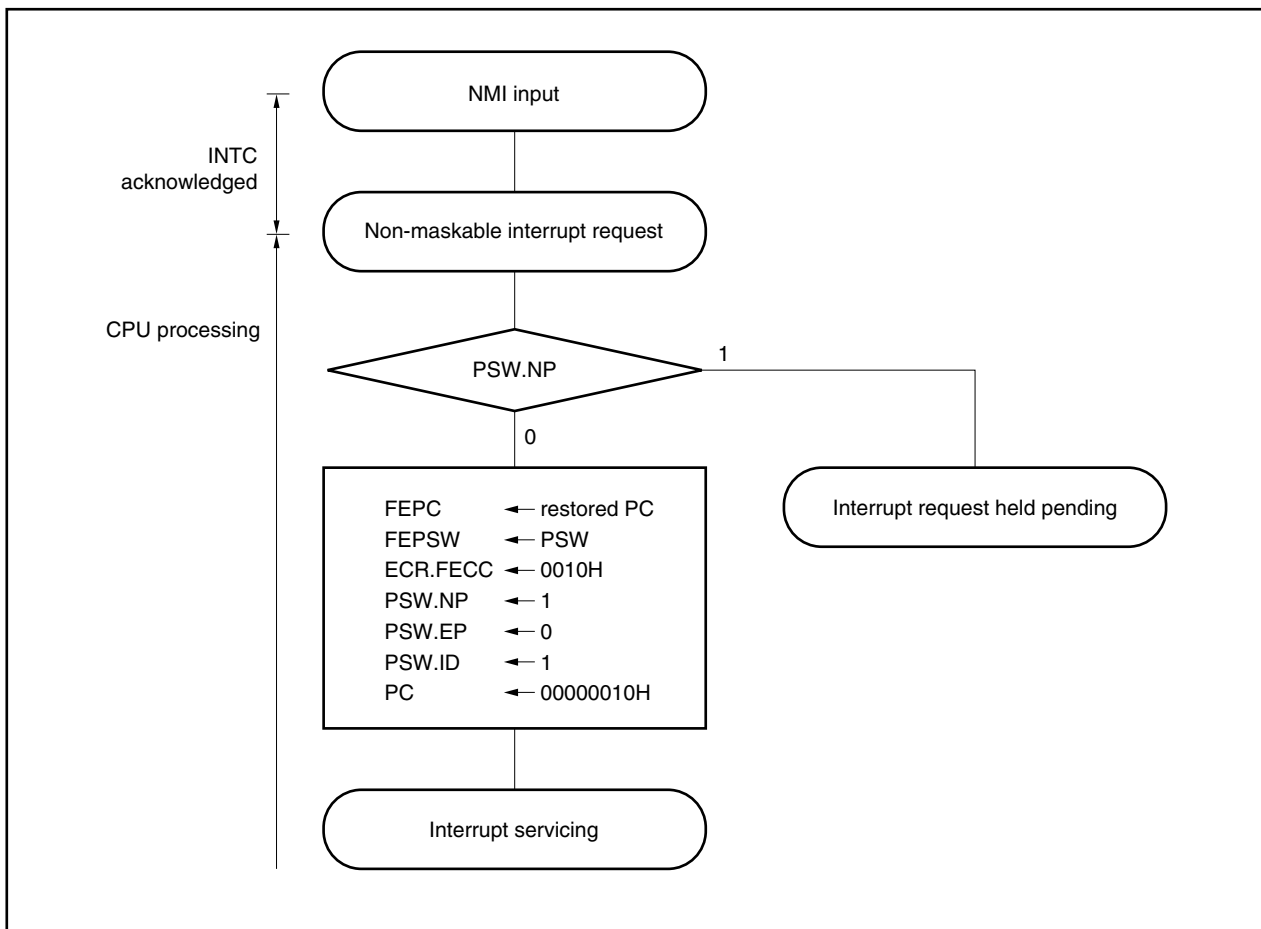
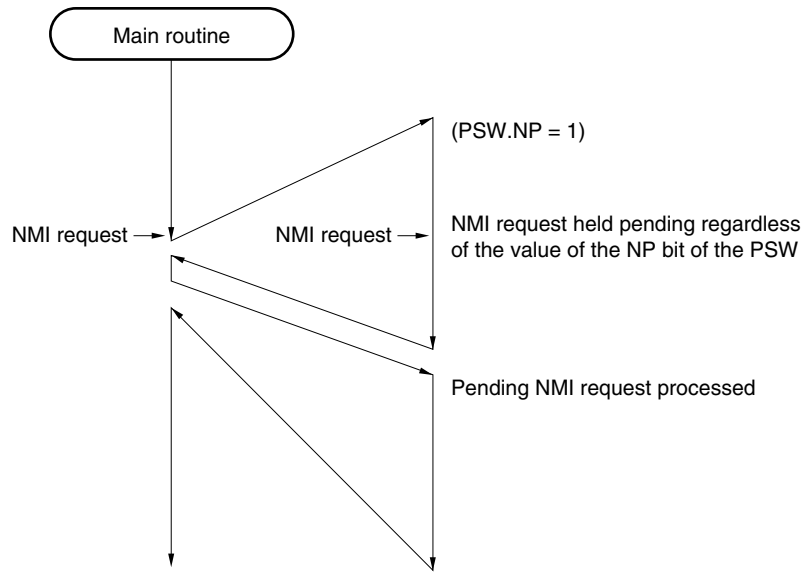
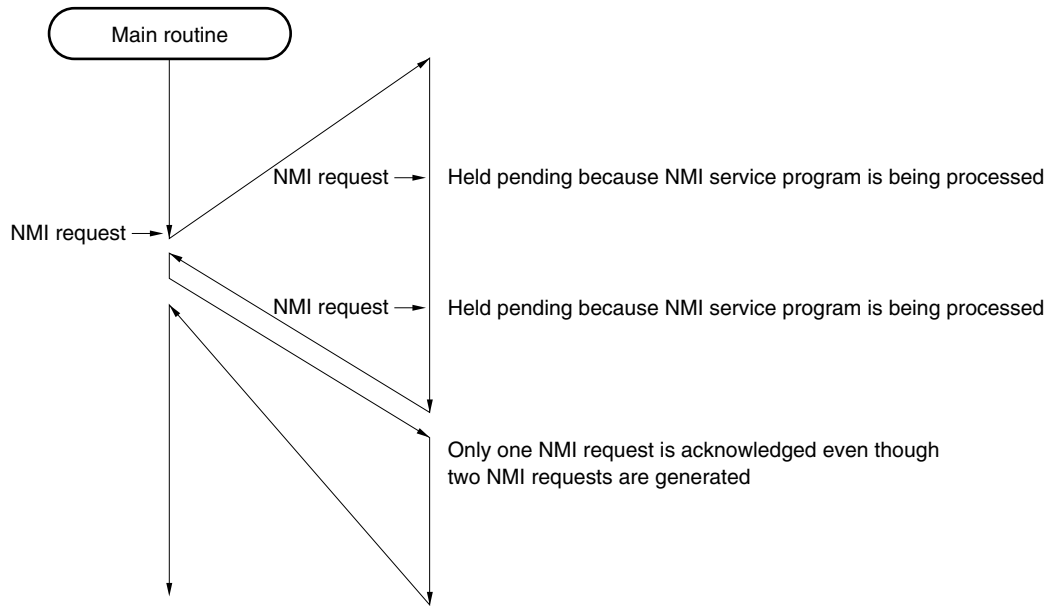


Figure 7-2. Acknowledging Non-Maskable Interrupt Request

(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

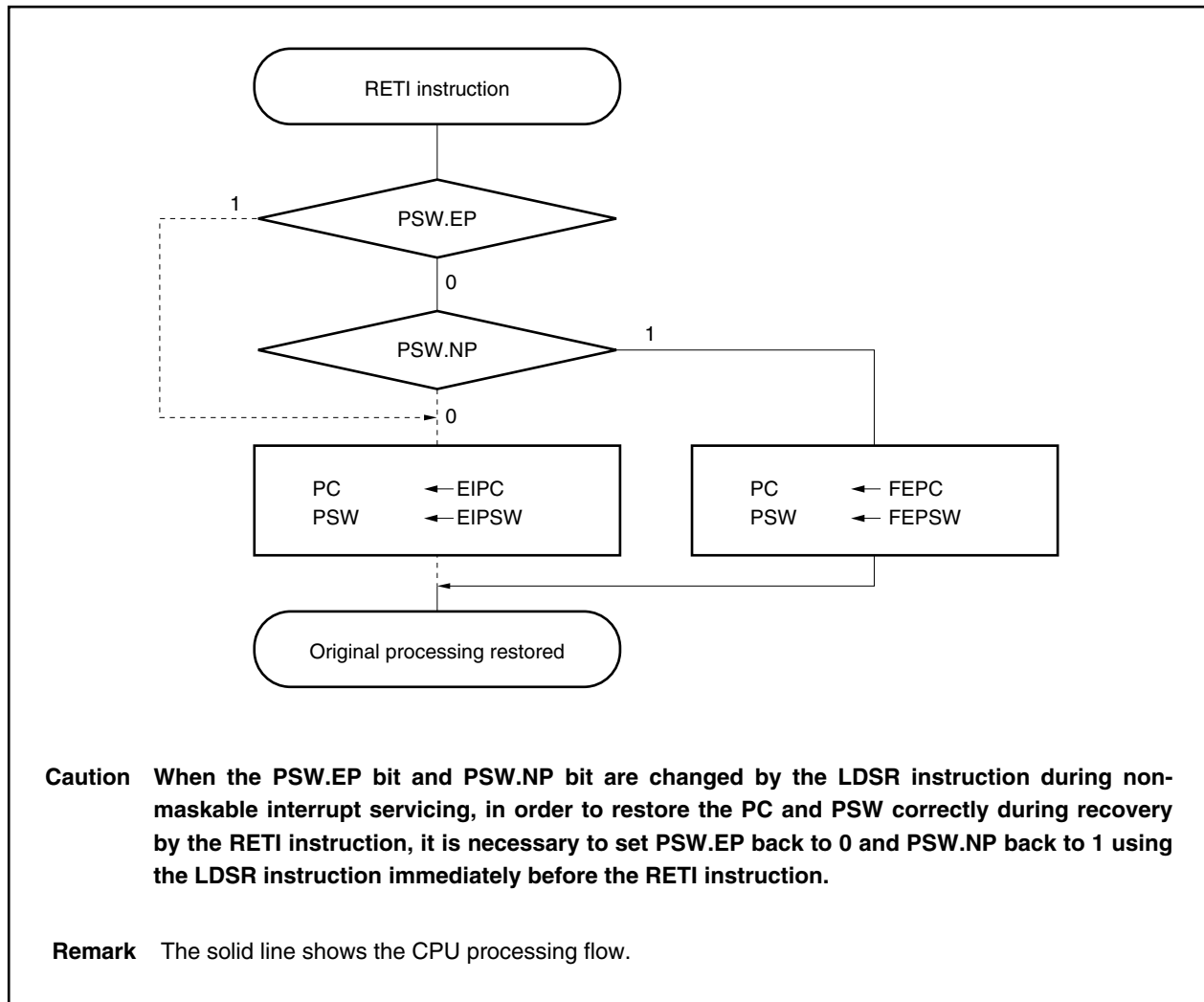
Execution is restored from the non-maskable interrupt servicing by the RETI instruction.

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 the PSW from FEPC and FEPSW, respectively, because the EP bit of the PSW is 0 and the NP bit of the PSW is 1.
- <2> Transfers control back to the address of the restored PC and PSW.

Figure 7-3 illustrates how the RETI instruction is processed.

Figure 7-3. RETI Instruction Processing



7.2.3 Non-maskable interrupt status flag (NP)

The NP flag is a status flag that indicates that non-maskable interrupt (NMI) servicing is under execution.

This flag is set when an NMI interrupt has been acknowledged, and masks all interrupt requests and exceptions to prohibit multiple interrupts from being acknowledged.

PSW	<table style="border: none; border-collapse: collapse; margin: 0 auto;"> <tr> <td style="border: none;">31</td> <td style="border: none;">0</td><td style="border: none;">1</td><td style="border: none;">2</td><td style="border: none;">3</td><td style="border: none;">4</td><td style="border: none;">5</td><td style="border: none;">6</td><td style="border: none;">7</td><td style="border: none;">8</td><td style="border: none;">9</td><td style="border: none;">10</td><td style="border: none;">11</td><td style="border: none;">12</td><td style="border: none;">13</td><td style="border: none;">14</td><td style="border: none;">15</td><td style="border: none;">16</td><td style="border: none;">17</td><td style="border: none;">18</td><td style="border: none;">19</td><td style="border: none;">20</td><td style="border: none;">21</td><td style="border: none;">22</td><td style="border: none;">23</td><td style="border: none;">24</td><td style="border: none;">25</td><td style="border: none;">26</td><td style="border: none;">27</td><td style="border: none;">28</td><td style="border: none;">29</td><td style="border: none;">30</td><td style="border: none;">31</td> </tr> <tr style="border: none;"> <td style="border: none;"></td> <td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td> </tr> </table>	31	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31		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	0	0	0	0	After reset 00000020H
31	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31																																		
	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	0	0	0	0																																				

Bit Position	Bit Name	Function
7	NP	NMI Pending Indicates whether 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 300 ns. A signal input that changes within the delay time is not internally acknowledged.

7.2.5 Edge detection function

(1) External interrupt mode register 0 (INTM0)

External interrupt mode register 0 (INTM0) is a register that specifies the valid edge of a 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-bit or 1-bit units.

INTM0	<table style="border: none; border-collapse: collapse; margin: 0 auto;"> <tr> <td style="border: none;">7</td><td style="border: none;">6</td><td style="border: none;">5</td><td style="border: none;">4</td><td style="border: none;">3</td><td style="border: none;">2</td><td style="border: none;">1</td><td style="border: none;"><0></td> </tr> <tr style="border: none;"> <td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">0</td><td style="border: 1px solid black;">ESN0</td> </tr> </table>	7	6	5	4	3	2	1	<0>	0	0	0	0	0	0	0	ESN0	Address FFFFF880H	After reset 00H
7	6	5	4	3	2	1	<0>												
0	0	0	0	0	0	0	ESN0												

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/MA2 has 26 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 servicing of interrupts having a higher priority than the interrupt request 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 multiple interrupts are executed, the following processing 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>.

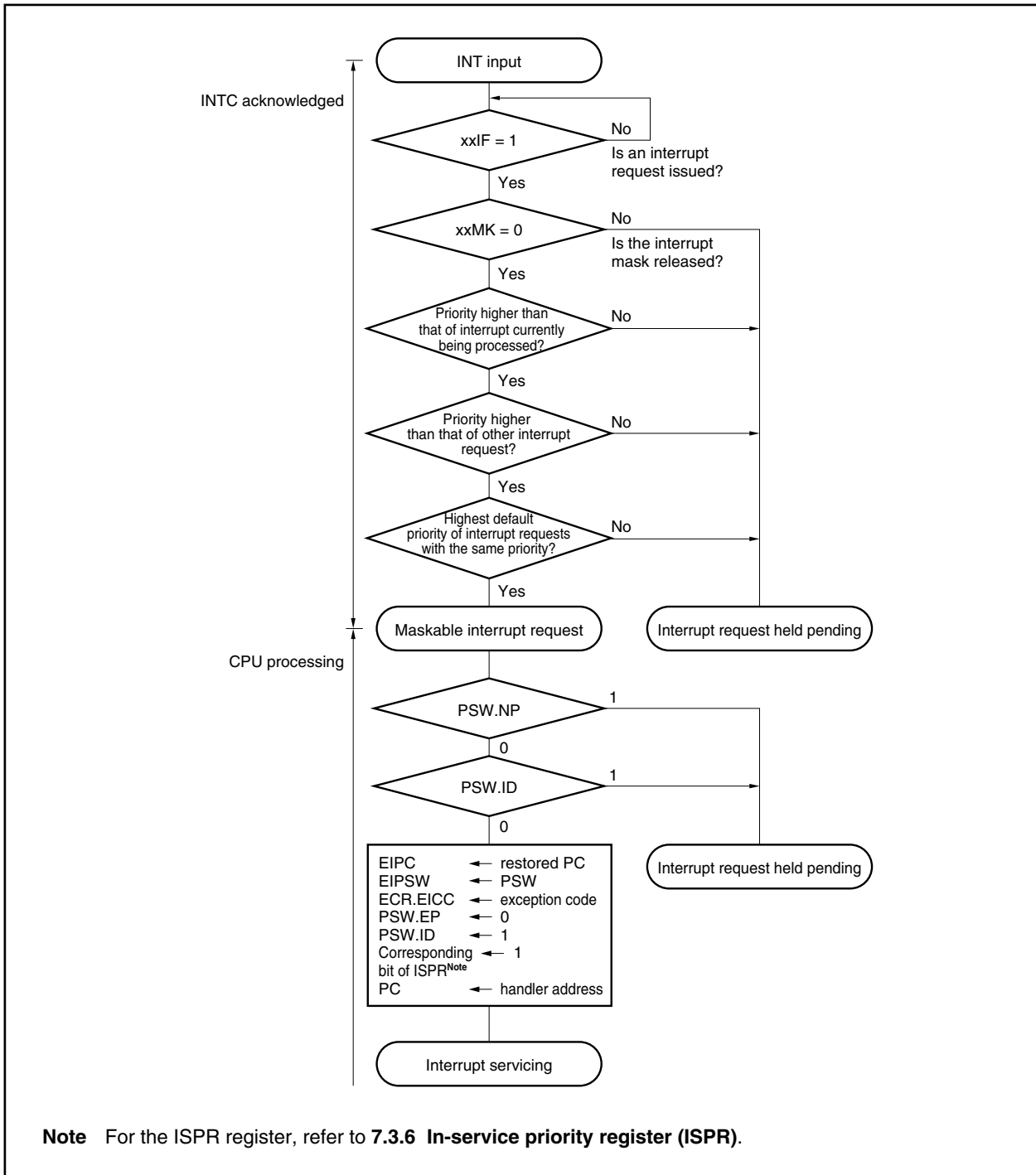
7.3.1 Operation

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

- <1> Saves the restored PC to EIPC.
- <2> Saves the current PSW to EIPSW.
- <3> Writes an exception code to the lower 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 servicing configuration of a maskable interrupt is shown in Figure 7-4.

Figure 7-4. 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. In such case, if the interrupts are unmasked, or when PSW.NP = 0 and PSW.ID = 0 as set by the RETI and LDSR instructions, input of the pending INT starts the new maskable interrupt servicing.

7.3.2 Restore

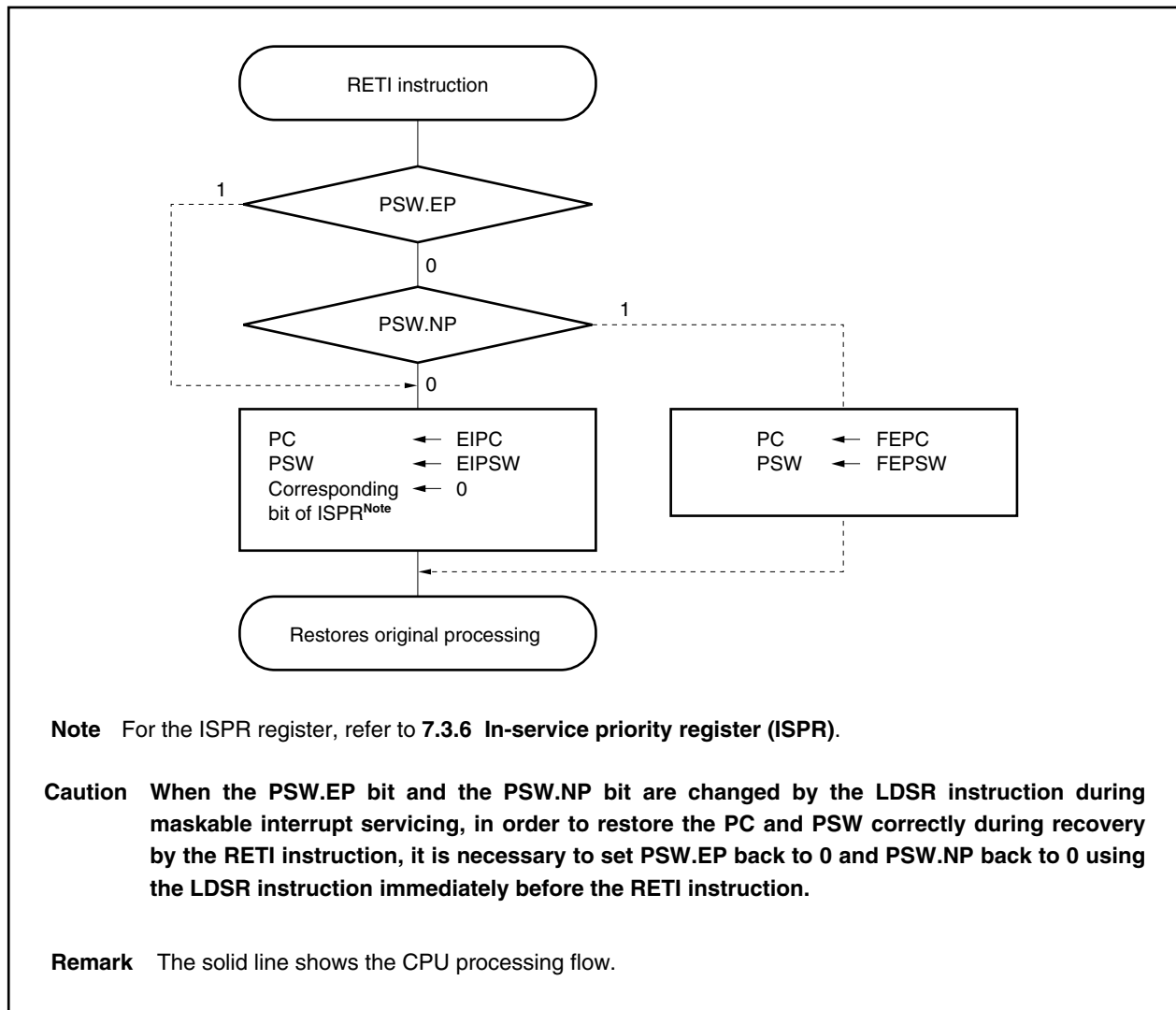
Recovery from maskable interrupt servicing is carried out by the RETI instruction.

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 the 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 7-5 illustrates the processing of the RETI instruction.

Figure 7-5. RETI Instruction Processing



7.3.3 Priorities of maskable interrupts

The V850E/MA2 provides multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being processed. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels that are specified by the interrupt priority level specification 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 processed 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/Exception Source 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 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 in the interrupt service program) to set the interrupt enable mode.

Remark xx: Identification name of each peripheral unit (refer to **Table 7-2**)
n: Peripheral unit number (refer to **Table 7-2**)

Figure 7-6. Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Processed (1/2)

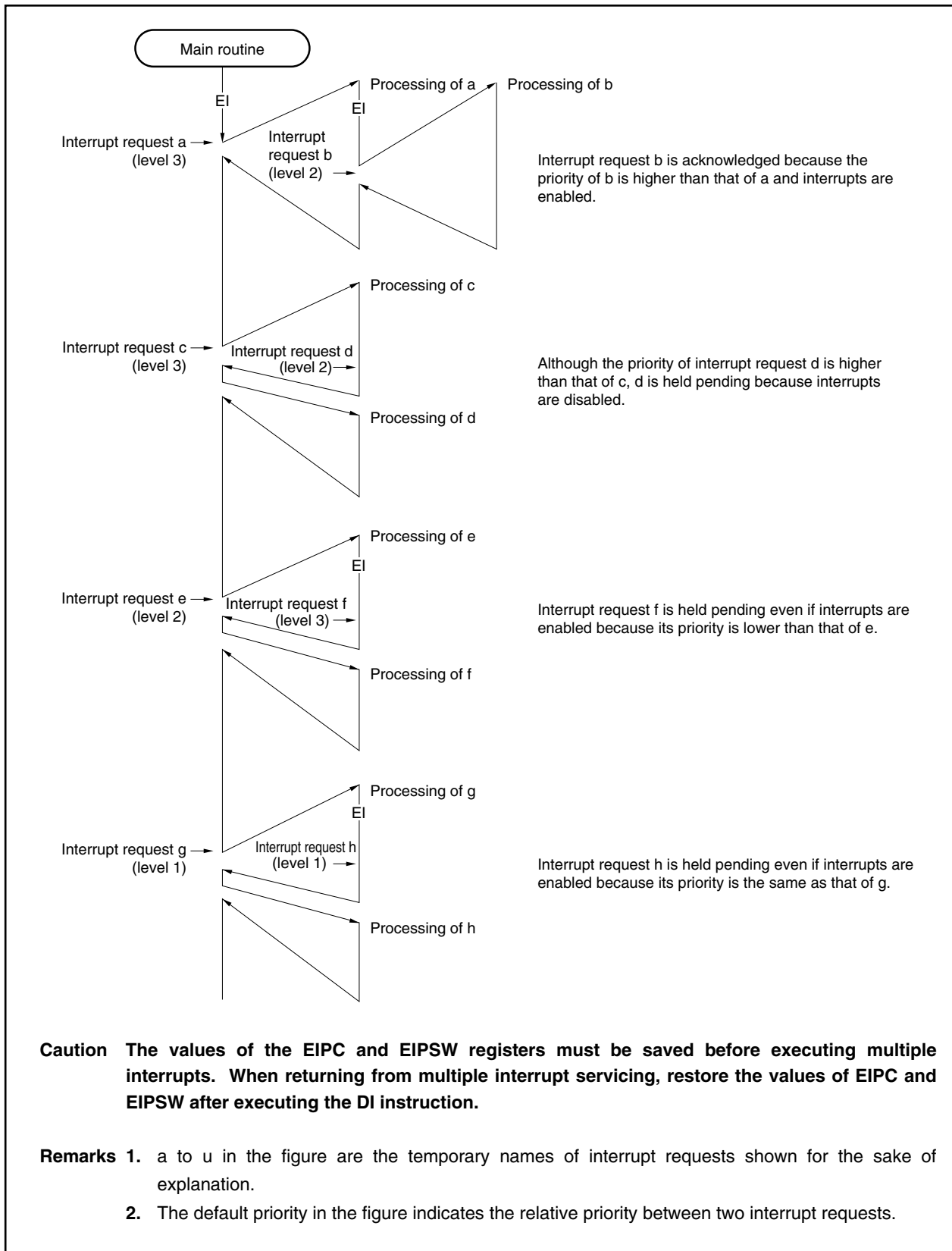


Figure 7-6. Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Processed (2/2)

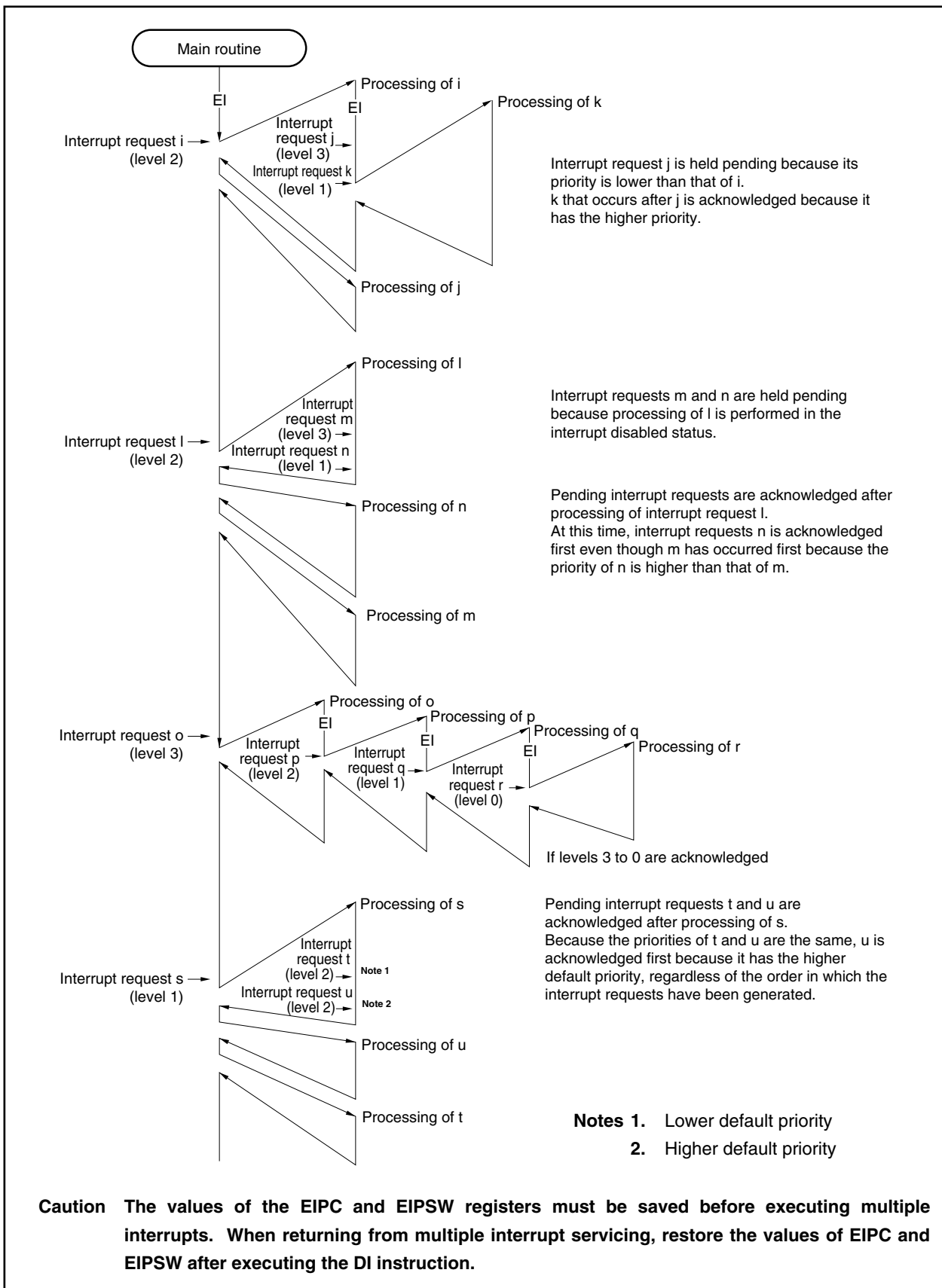
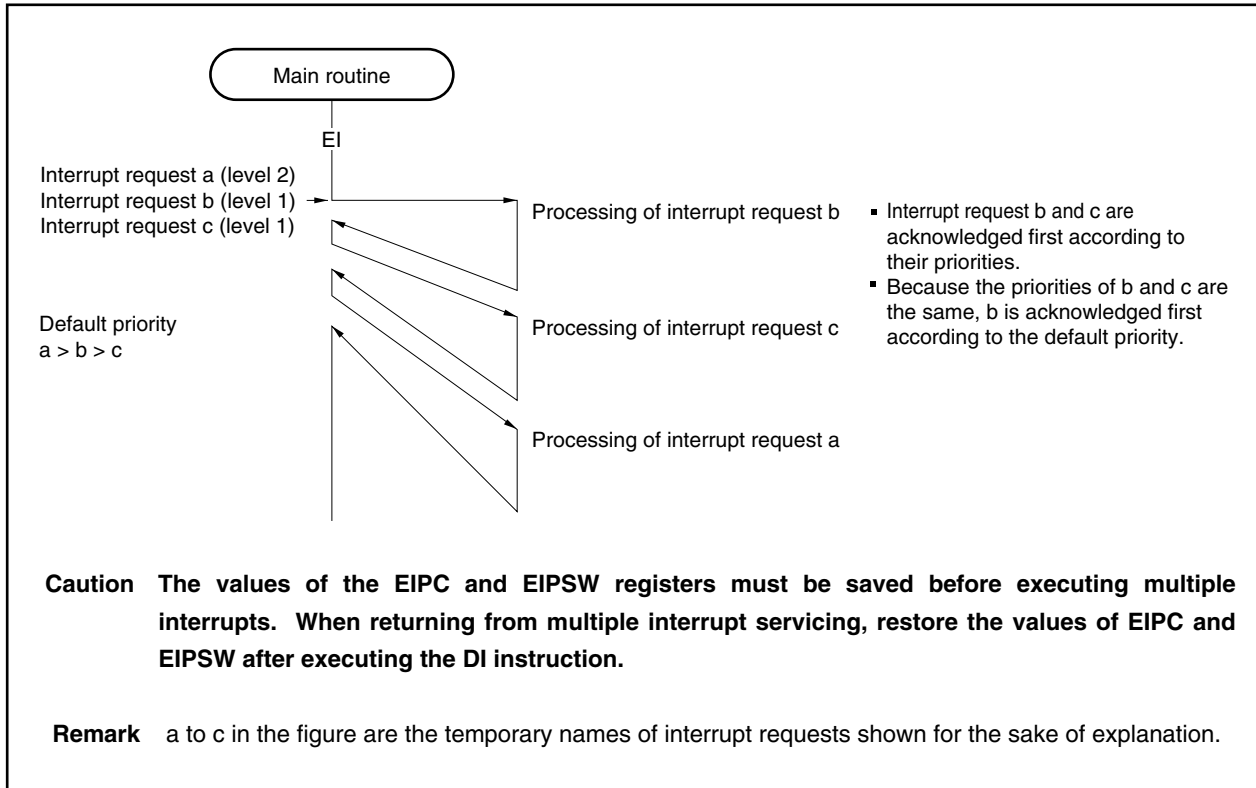


Figure 7-7. Example of Processing Interrupt Requests Simultaneously Generated



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-bit or 1-bit units.

★ **Caution** Read the xxIFn bit of the xxICn register in the interrupt disabled state. Otherwise if the interrupt acknowledgment and bit reading timing conflicts, normal values may not be read.

	<7>	<6>	5	4	3	2	1	0		
xxICn	xxIFn	xxMKn	0	0	0	xxPRn2	xxPRn1	xxPRn0	Address	After reset
									FFFFF110H to	47H
									FFFF170H	

Bit Position	Bit Name	Function
7	xxIFn	Interrupt Request Flag This is an interrupt request flag. 0: Interrupt request not issued 1: Interrupt request issued The flag xxIFn is reset automatically by the hardware if an interrupt request is acknowledged.
6	xxMKn	Mask Flag This is an interrupt mask flag. 0: Enables interrupt servicing 1: Disables interrupt servicing (pending)
2 to 0	xxPRn2 to xxPRn0	Priority 8 levels of priority order are specified for each interrupt.

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 (refer to **Table 7-2**)
 n: Peripheral unit number (refer to **Table 7-2**).

The address and bit of each interrupt control register are as follows:

Table 7-2. Address and Bits of Interrupt Control Register

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF110H	OVIC00	OVIF0	OVMK0	0	0	0	OVPR02	OVPR01	OVPR00
FFFFF112H	OVIC01	OVIF1	OVMK1	0	0	0	OVPR12	OVPR11	OVPR10
FFFFF118H	P00IC0	P00IF0	P00MK0	0	0	0	P00PR02	P00PR01	P00PR00
FFFFF11AH	P00IC1	P00IF1	P00MK1	0	0	0	P00PR12	P00PR11	P00PR10
FFFFF11CH	P01IC0	P01IF0	P01MK0	0	0	0	P01PR02	P01PR01	P01PR00
FFFFF11EH	P01IC1	P01IF1	P01MK1	0	0	0	P01PR12	P01PR11	P01PR10
FFFFF128H	P10IC0	P10IF0	P10MK0	0	0	0	P10PR02	P10PR01	P10PR00
FFFFF12AH	P10IC1	P10IF1	P10MK1	0	0	0	P10PR12	P10PR11	P10PR10
FFFFF130H	P11IC0	P11IF0	P11MK0	0	0	0	P11PR02	P11PR01	P11PR00
FFFFF148H	CMICD0	CMIF0	CMMK0	0	0	0	CMPR02	CMPR01	CMPR00
FFFFF14AH	CMICD1	CMIF1	CMMK1	0	0	0	CMPR12	CMPR11	CMPR10
FFFFF14CH	CMICD2	CMIF2	CMMK2	0	0	0	CMPR22	CMPR21	CMPR20
FFFFF14EH	CMICD3	CMIF3	CMMK3	0	0	0	CMPR32	CMPR31	CMPR30
FFFFF150H	DMAIC0	DMAIF0	DMAMK0	0	0	0	DMAPR02	DMAPR01	DMAPR00
FFFFF152H	DMAIC1	DMAIF1	DMAMK1	0	0	0	DMAPR12	DMAPR11	DMAPR10
FFFFF154H	DMAIC2	DMAIF2	DMAMK2	0	0	0	DMAPR22	DMAPR21	DMAPR20
FFFFF156H	DMAIC3	DMAIF3	DMAMK3	0	0	0	DMAPR32	DMAPR31	DMAPR30
FFFFF158H	CSIC0	CSIF0	CSIMK0	0	0	0	CSIPR02	CSIPR01	CSIPR00
FFFFF15AH	SEIC0	SEIF0	SEMK0	0	0	0	SEPR02	SEPR01	SEPR00
FFFFF15CH	SRIC0	SRIF0	SRMK0	0	0	0	SRPR02	SRPR01	SRPR00
FFFFF15EH	STIC0	STIF0	STMK0	0	0	0	STPR02	STPR01	STPR00
FFFFF160H	CSIC1	CSIF1	CSIMK1	0	0	0	CSIPR12	CSIPR11	CSIPR10
FFFFF162H	SEIC1	SEIF1	SEMK1	0	0	0	SEPR12	SEPR11	SEPR10
FFFFF164H	SRIC1	SRIF1	SRMK1	0	0	0	SRPR12	SRPR11	SRPR10
FFFFF166H	STIC1	STIF1	STMK1	0	0	0	STPR12	STPR11	STPR10
FFFFF170H	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0

7.3.5 Interrupt mask registers 0 to 3 (IMR0 to IMR3)

These registers set the interrupt mask state for the maskable interrupts. The xxMKn bit of the IMR0 to IMR3 registers is equivalent to the xxMKn bit of the xxICn register.

The IMRm register (m = 0 to 3) can be read/written in 16-bit units.

If the higher 8 bits of the IMRm register are used as an IMRmH register and the lower 8 bits as an IMRmL register, these registers can be read/written in 8- or 1-bit units.

Bits 15, 14, 11 to 8, 3, and 2 of the IMR0 register (bits 7, 6, and 3 to 0 of the IMR0H register and bits 3 and 2 of the IMR0L register), bits 11 to 1 of the IMR1 register (bits 3 to 0 of the IMR1H register and bits 7 to 1 of the IMR1L register), bits 15 to 12 of the IMR2 register (bits 7 to 4 of the IMR2H register), and bits 15 to 1 of the IMR3 register (bits 7 to 0 of the IMR3H register and bits 7 to 1 of the IMR3L register) are fixed to 1. If these bits are not 1, the operation cannot be guaranteed.

- ★ **Caution** The device file defines the xxMKn bit of the xxICn register as a reserved word. If a bit is manipulated using the name of xxMKn, the contents of the xxICn register, instead of the IMRm register, are rewritten (as a result, the contents of the IMRm register are also rewritten).

IMR0	15	14	13	12	11	10	9	8	Address FFFFFF100H	After reset FFFFH
	1	1	P10MK1	P10MK0	1	1	1	1		
	7	6	5	4	3	2	1	0		
	P01MK1	P01MK0	P00MK1	P00MK0	1	1	OVMK1	OVMK0		
IMR1	15	14	13	12	11	10	9	8	Address FFFFFF102H	After reset FFFFH
	CMMK3	CMMK2	CMMK1	CMMK0	1	1	1	1		
	7	6	5	4	3	2	1	0		
	1	1	1	1	1	1	1	P11MK0		
IMR2	15	14	13	12	11	10	9	8	Address FFFFFF104H	After reset FFFFH
	1	1	1	1	STMK1	SRMK1	SEMK1	CSIMK1		
	1	6	5	4	3	2	1	0		
	STMK0	SRMK0	SEMK0	CSIMK0	DMAMK3	DMAMK2	DMAMK1	DMAMK0		
IMR3	15	14	13	12	11	10	9	8	Address FFFFFF106H	After reset FFFFH
	1	1	1	1	1	1	1	1		
	7	6	5	4	3	2	1	0		
	1	1	1	1	1	1	1	ADMK		

Bit Position	Bit Name	Function
13, 12, 7 to 4, 1, 0 (IMR0) 15 to 12, 0 (IMR1) 11 to 0 (IMR2) 0 (IMR3)	xxMKn	Mask Flag Interrupt mask flag 0: Interrupt servicing permitted 1: Interrupt servicing prohibited (pending)

Remark xx: Identification name of each peripheral unit (refer to **Table 7-2**).
n: Peripheral unit number (refer to **Table 7-2**)

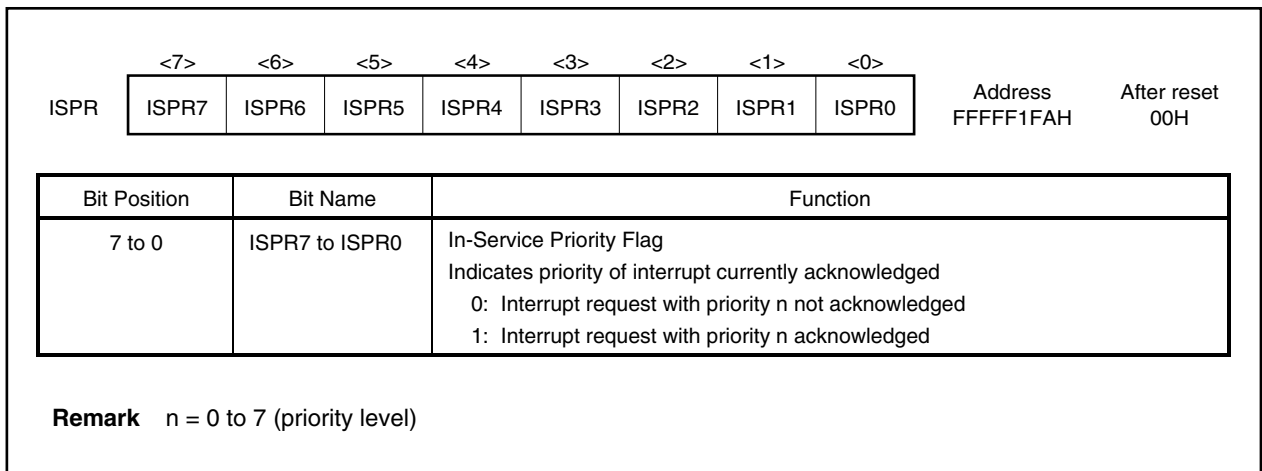
7.3.6 In-service priority register (ISPR)

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

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

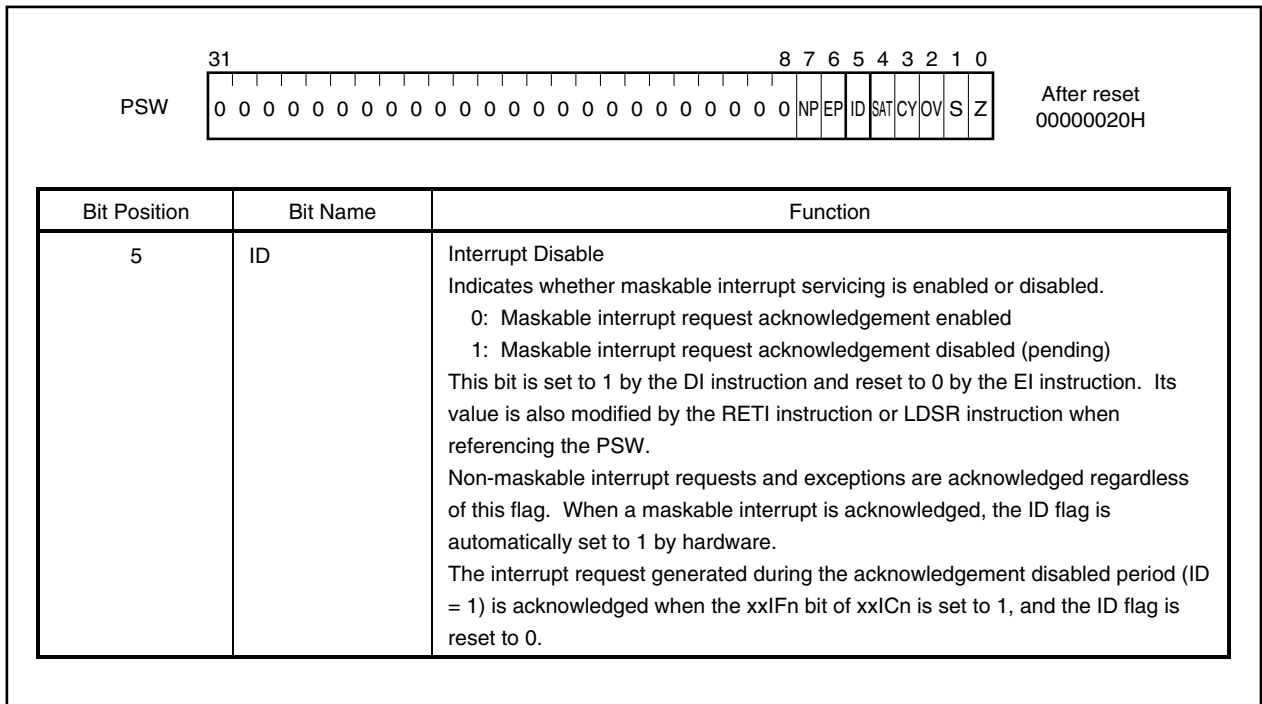
This register is read-only in 8-bit or 1-bit units.

★ **Caution** Read the ISPR register in the interrupt disabled state. Otherwise if the interrupt acknowledgment and register reading timing conflicts, normal values may not be read.



7.3.7 Maskable interrupt status flag (ID)

The ID flag is bit 5 of the PSW and this controls the maskable interrupt's operating state, and stores control information regarding enabling or disabling of interrupt requests.



7.3.8 Noise elimination

The noise of the INTP_n, $\overline{\text{INTP}}_m$, TI000, and TI010 pins is eliminated with analog delay ($n = 000, 001, 010, 011$, $m = 100, 101, 110$). The delay time is about 60 to 220 ns. A signal input that changes within the delay time is not internally acknowledged.

7.3.9 Interrupt trigger mode selection

The valid edge of pins INTP0n0, INTP0n1, $\overline{\text{INTP}}1n0$, $\overline{\text{INTP}}101$, and TI0n0 can be selected by program. Moreover, a level trigger can be selected for the $\overline{\text{INTP}}1n0$ and $\overline{\text{INTP}}101$ pins ($n = 0, 1$). The edge that can be selected as the valid edge is one of the following.

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

When the INTP0n0, INTP0n1, INTP1n0, INTP101, and TI0n0 pins are edge-detected, they become an interrupt source, a capture trigger input, and a timer external count input respectively ($n = 0, 1$).

The valid edge is specified by external interrupt mode registers 1 and 2 (INTM1, INTM2) and valid edge select registers C0 and C1 (SESC0, SESC1). The level trigger is specified by external interrupt registers 1 and 2 (INTM1, INTM2).

(1) External interrupt mode registers 1, 2 (INTM1, INTM2)

These are registers that specify the trigger mode for external interrupt requests (INTP100, INTP101, INTP110), input via external pins. The correspondence between each register and the external interrupt requests that register controls is shown below.

- INTM1: INTP100, INTP101
- INTM2: INTP110

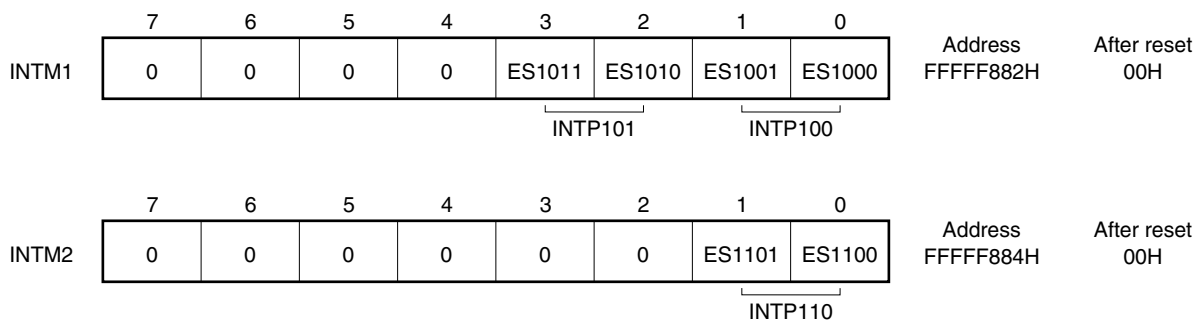
The valid edge can be specified independently for each pin (rising edge, falling edge, or both rising and falling edges).

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

Be sure to set bits 7 to 4 of the INTM1 register and bits 7 to 2 of the INTM2 register to 0. If they are set to 1, the operation is not guaranteed.

★ **Caution** Before setting the $\overline{\text{INTP}}100$, $\overline{\text{INTP}}101$, or $\overline{\text{INTP}}110$ pin in the trigger mode, set the PMC_m register.

If the PMC_m register is set after the INTM1 and INTM2 registers have been set, an illegal interrupt may occur, depending on the timing of setting the PMC_m register ($m = 0, 2$).



Bit Position	Bit Name	Function															
3 to 0 (INTM1) 1, 0 (INTM2)	ES1nm1, ES1nm0 (nm = 00, 01, 10)	Edge Select Specifies the valid edge of the INTP1nm pins. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>ES1nm1</th> <th>ES1nm0</th> <th>Operation</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>Level detection (low-level detection)^{Notes 1, 2}</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	ES1nm1	ES1nm0	Operation	0	0	Falling edge	0	1	Rising edge	1	0	Level detection (low-level detection) ^{Notes 1, 2}	1	1	Both rising and falling edges
ES1nm1	ES1nm0	Operation															
0	0	Falling edge															
0	1	Rising edge															
1	0	Level detection (low-level detection) ^{Notes 1, 2}															
1	1	Both rising and falling edges															

- Notes 1.** The level of the INTP1nm pin is sampled at the interval of the system clock divided by two, and the P1nIFm bit is latched as an interrupt request when a low level is detected. Therefore, even if the P1nIFm bit of the interrupt control register (P1nICm) is automatically cleared to 0 when the CPU acknowledges an interrupt, the P1nIFm bit is immediately set to 1, and an interrupt is generated continuously. To avoid this, forcibly clear the P1nIFm bit to 0 after making the INTP1nm pin inactive for an external device in the interrupt servicing routine (nm = 00, 01, 10).
- 2.** When a lower-priority level-detection interrupt request (INTP1nm) occurs while another interrupt is being-serviced and this newly-generated level-detection interrupt request becomes inactive before the current interrupt servicing is complete, this new interrupt request (INTP1nm) is held pending. To avoid acknowledging this INTP1nm interrupt request, clear the P1nIFm bit of the interrupt control register (nm = 00, 01, 10).

(2) Valid edge selection registers C0, C1 (SESC0, SESC1)

These are registers that specify the valid edge for external interrupt requests (INTP000, INTP001, INTP010, INTP011, TI000, TI010), input via external pins. The correspondence between each register and the external interrupt requests which that register controls is shown below.

- SESC0: TI000, INTP000, INTP001
- SESC1: TI010, INTP010, INTP011

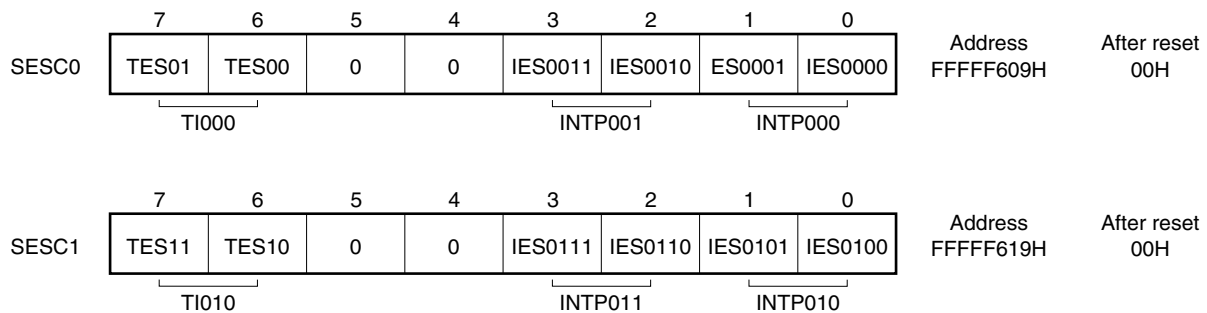
The valid edge can be specified independently for each pin (rising edge, falling edge, or both rising and falling edges).

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

Cautions 1. When using the INTP0n0/TI0n0 or INTP0n1 pin as INTP0n0, INTP0n1, be sure to preset the TMCCAEn bit of timer mode control register Cn0 (TMCCn0) to 1 (n = 0, 1).

★ **2.** Before setting the TI0n0, INTP0n1, or INTP0n0 pin in the trigger mode, set the PMCN register.

If the PMCN register is set after the SESCn register has been set, an illegal interrupt may occur, depending on the timing of setting the PMCN register (n = 0, 1).



Bit Position	Bit Name	Function															
7, 6	TESn1, TESn0 (n = 0, 1)	Edge Select Specifies the valid edge of the INTPn and TI000 and TI010 pins.															
3, 2	IESn1, IESn0 (n = 001, 011)	<table border="1"> <thead> <tr> <th>xESn1</th> <th>xESn0</th> <th>Operation</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 rising and falling edges</td> </tr> </tbody> </table>	xESn1	xESn0	Operation	0	0	Falling edge	0	1	Rising edge	1	0	RFU (reserved)	1	1	Both rising and falling edges
xESn1	xESn0	Operation															
0	0	Falling edge															
0	1	Rising edge															
1	0	RFU (reserved)															
1	1	Both rising and falling edges															
1, 0	IESn1, IESn0 (n = 000, 010)																

7.4 Software Exception

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

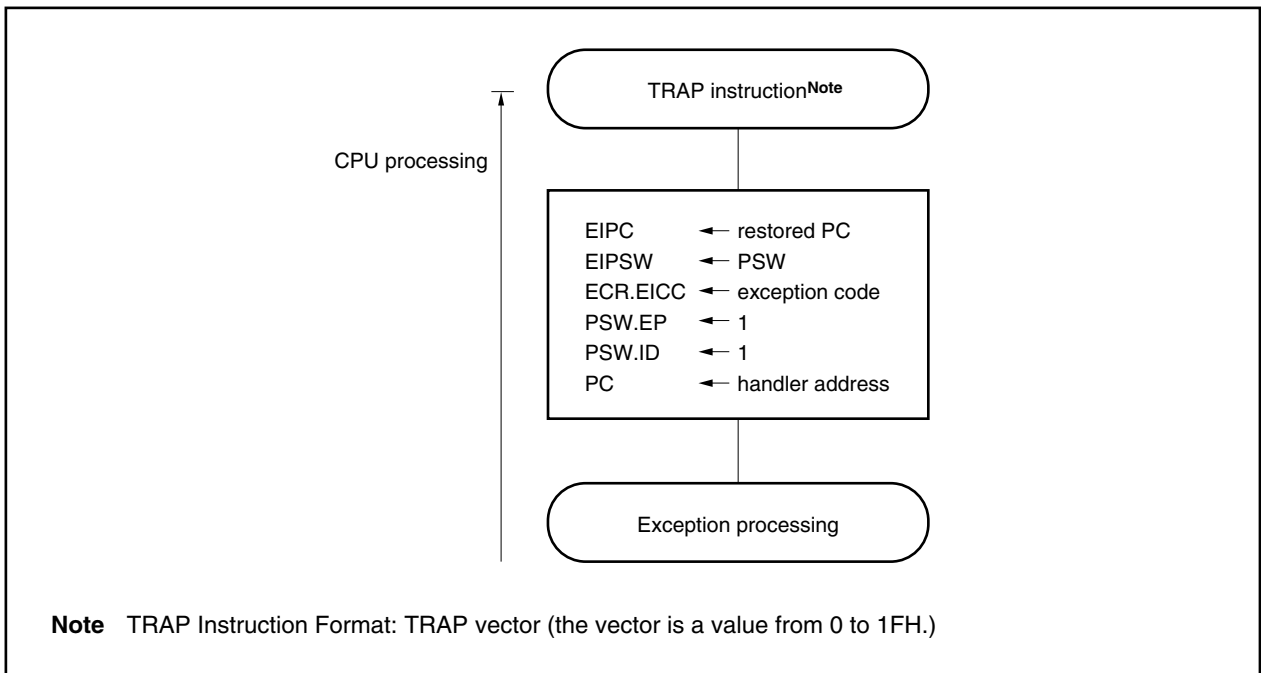
7.4.1 Operation

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 7-8 illustrates the processing of a software exception.

Figure 7-8. 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

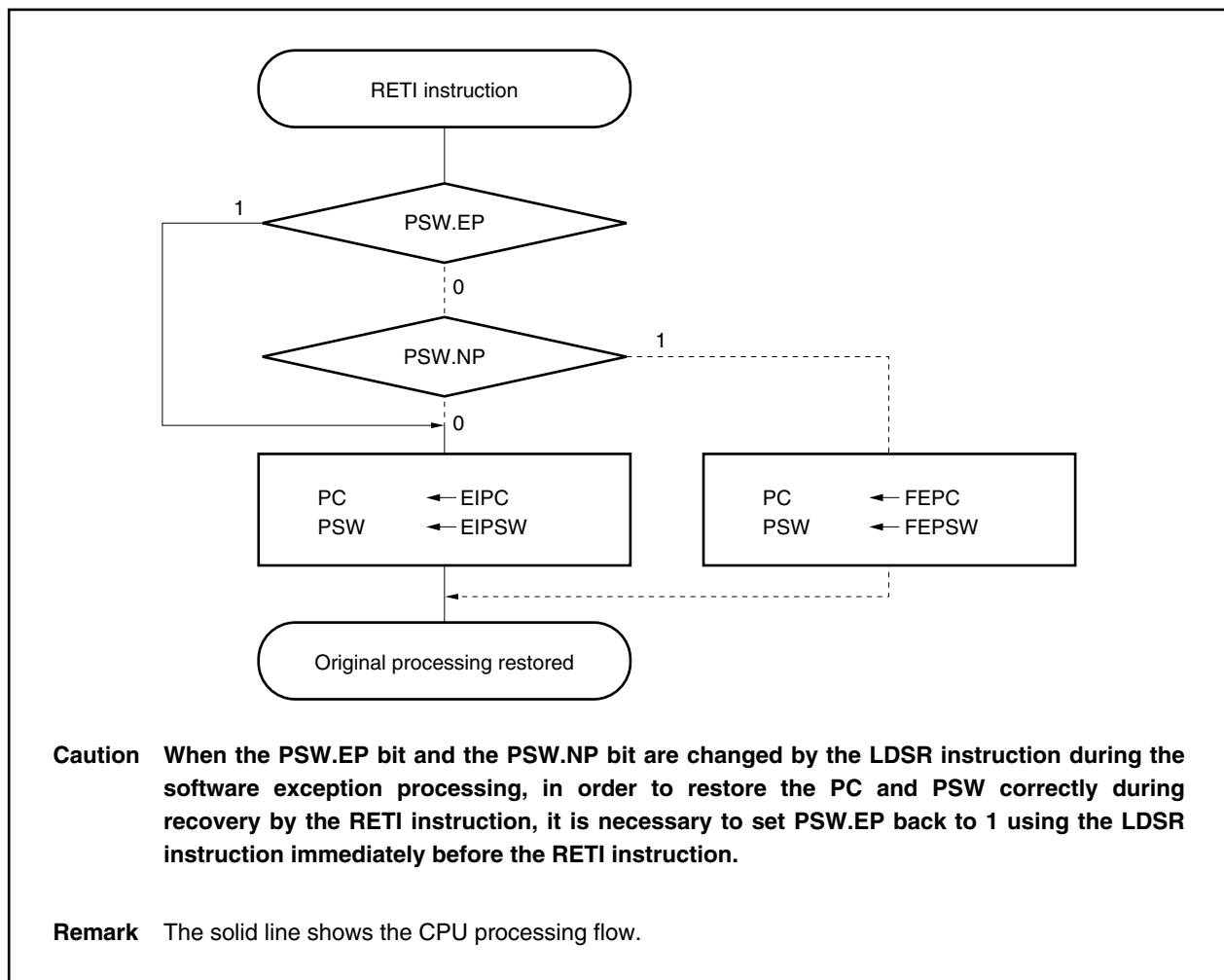
Recovery from software exception processing is carried out by the RETI instruction.

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 the PSW is 1.
- <2> Transfers control to the address of the restored PC and PSW.

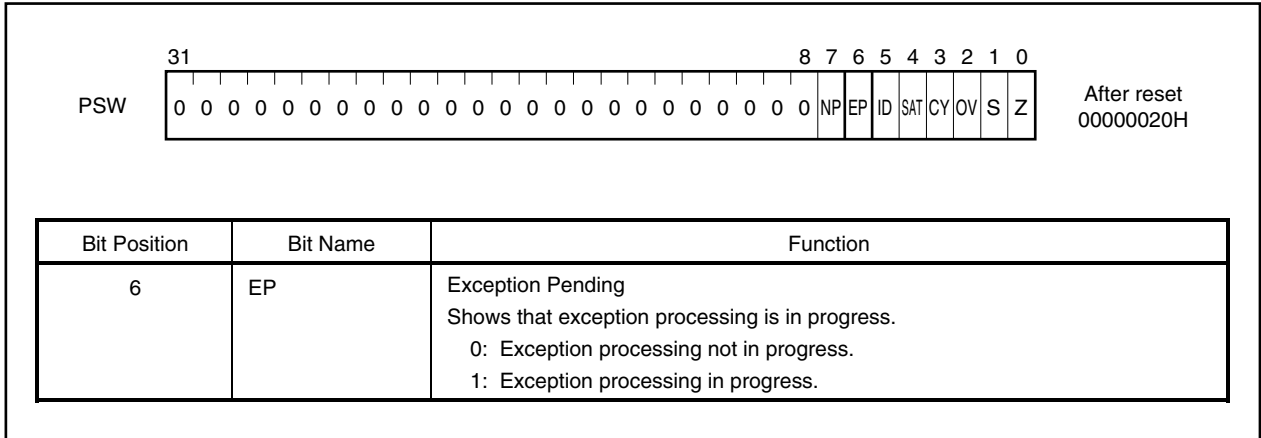
Figure 7-9 illustrates the processing of the RETI instruction.

Figure 7-9. RETI Instruction Processing



7.4.3 Exception status flag (EP)

The EP flag is bit 6 of PSW, and is a status flag used to indicate that exception processing is in progress. It is set when an exception occurs.

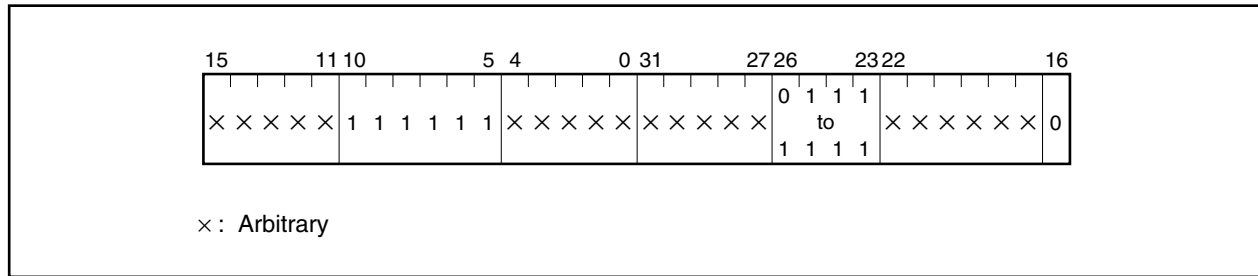


7.5 Exception Trap

An exception trap is an interrupt that is requested when an illegal execution of an instruction takes place. In the V850E/MA2, an illegal opcode exception (ILGOP: Illegal Opcode Trap) is considered as an exception trap.

7.5.1 Illegal opcode definition

The illegal instruction has an opcode (bits 10 to 5) of 111111B, a sub-opcode (bits 26 to 23) of 0111B to 1111B, and a sub-opcode (bit 16) of 0B. An exception trap is generated when an instruction applicable to this illegal instruction is executed.



Caution Since it is possible to assign this instruction to an illegal opcode in the future, it is recommended that it not be used.

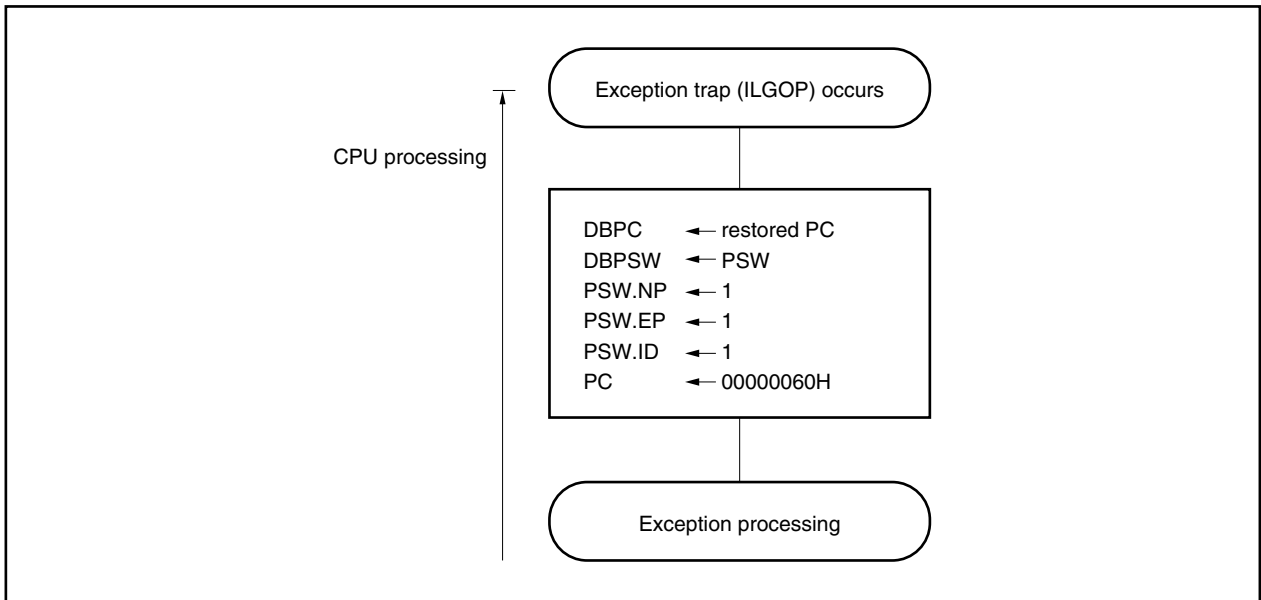
(1) Operation

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 the PSW.
- <4> Sets the handler address (00000060H) corresponding to the exception trap to the PC, and transfers control.

Figure 7-10 illustrates the processing of the exception trap.

Figure 7-10. Exception Trap Processing

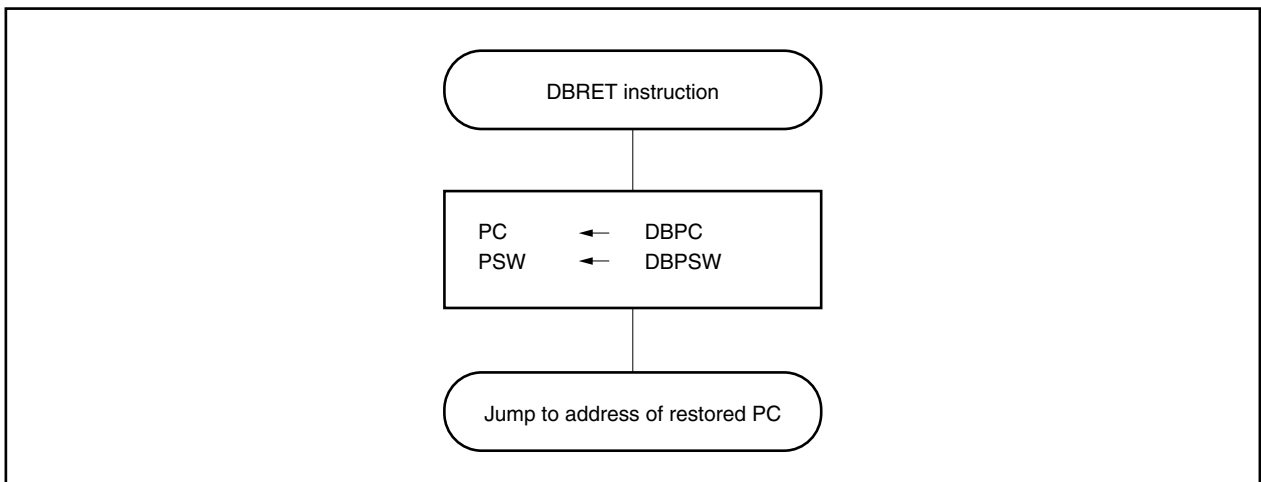
**(2) Restore**

Recovery from an exception trap is carried out by the DBRET instruction. By executing the DBRET instruction, the CPU carries out the following processing and controls the address of the restored PC.

- <1> Loads the restored PC and PSW from DBPC and DBPSW.
- <2> Transfers control to the address indicated by the restored PC and PSW.

Figure 7-11 illustrates the restore processing from an exception trap.

Figure 7-11. Restore Processing from Exception Trap



7.5.2 Debug trap

The debug trap is an exception that can be acknowledged every time and is generated by execution of the DBTRAP instruction.

When the debug trap is generated, the CPU performs the following processing.

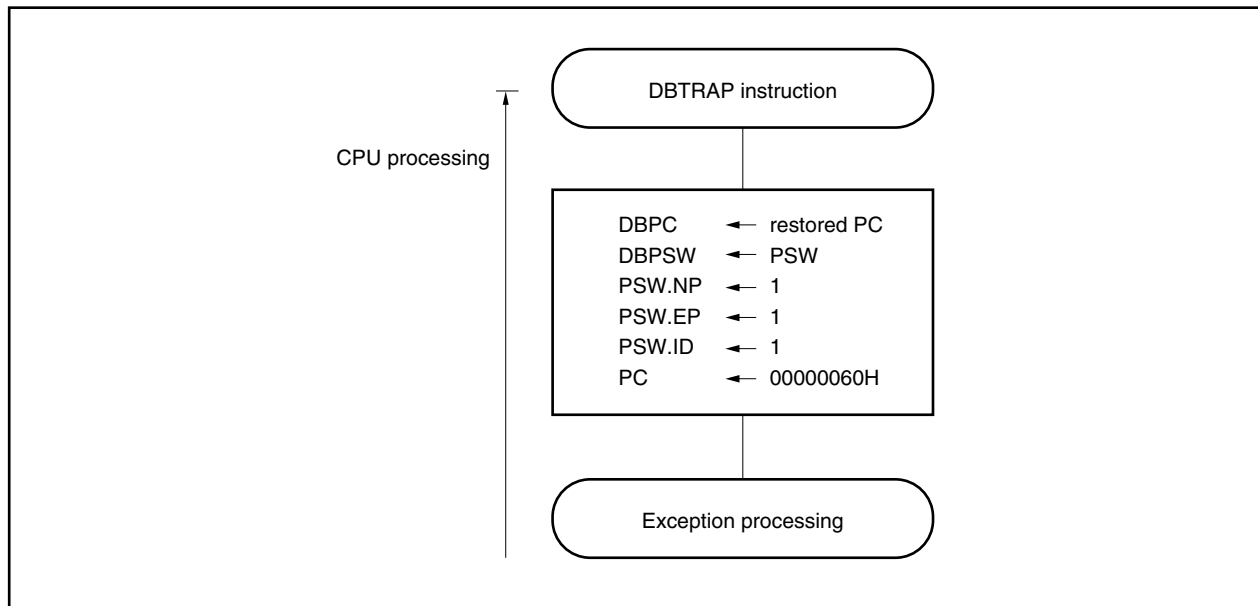
(1) Operation

When the debug trap is generated, the CPU performs the following processing, transfers control to the debug monitor routine, and shifts to debug mode.

- <1> Saves the restored PC to DBPC.
- <2> Saves the current PSW to DBPSW.
- <3> Sets the NP, EP and ID bits of the PSW.
- <4> Sets the handler address (00000060H) corresponding to the debug trap to the PC and transfers control.

Figure 7-12 illustrates the processing of the debug trap.

Figure 7-12. Debug Trap Processing



(2) Restore

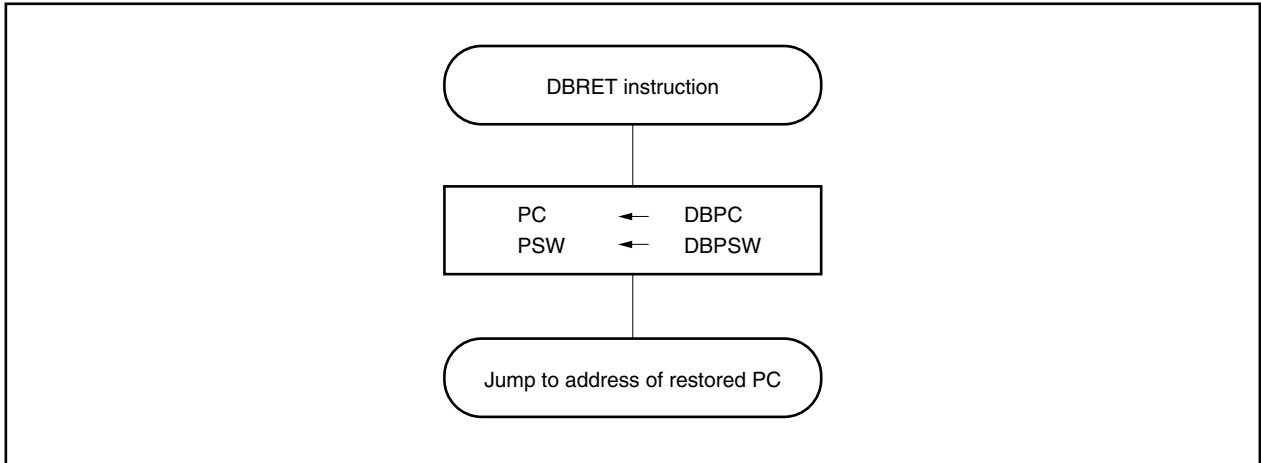
Recovery from a debug trap is carried out by the DBRET instruction. By executing the DBRET instruction, the CPU carries out the following processing and controls the address of the restored PC.

<1> Loads the restored PC and PSW from DBPC and DBPSW.

<2> Transfers control to the address indicated by the restored PC and PSW.

Figure 7-13 illustrates the restore processing from a debug trap.

Figure 7-13. Restore Processing from Debug Trap



7.6 Multiple Interrupt Servicing Control

Multiple interrupt servicing control is a process by which an interrupt request that is currently being processed can be interrupted during processing if there is an interrupt request with a higher priority level, and the higher priority interrupt request is received and processed 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.

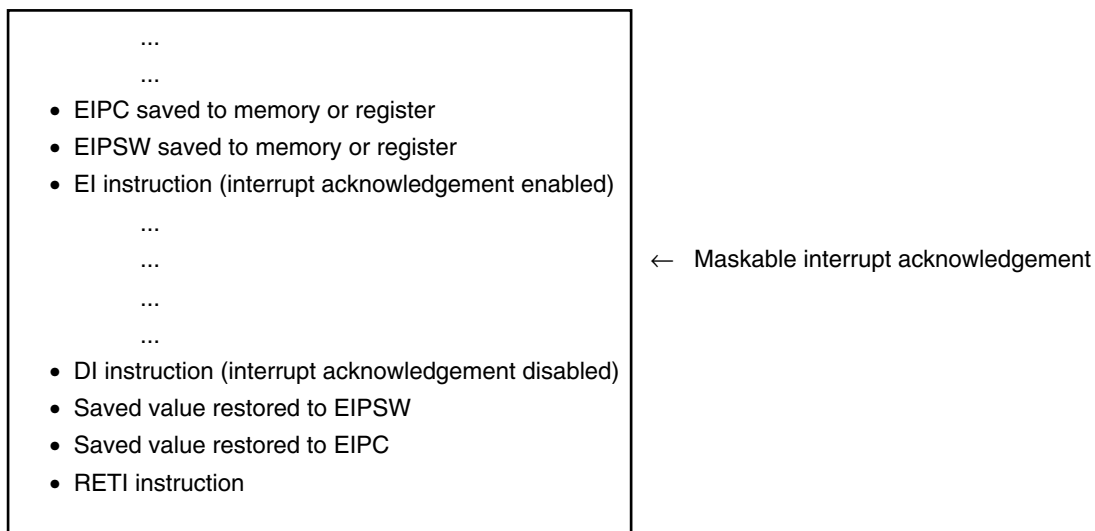
Maskable interrupt multiple servicing control is executed when interrupts are enabled (ID = 0). Thus, if multiple interrupts are executed, it is necessary to enable interrupts (ID = 0) even for an interrupt servicing routine.

If maskable interrupt enable or a software exception is generated in a maskable interrupt or software exception servicing program, it is necessary to save EIPC and EIPSW.

This is accomplished by the following procedure.

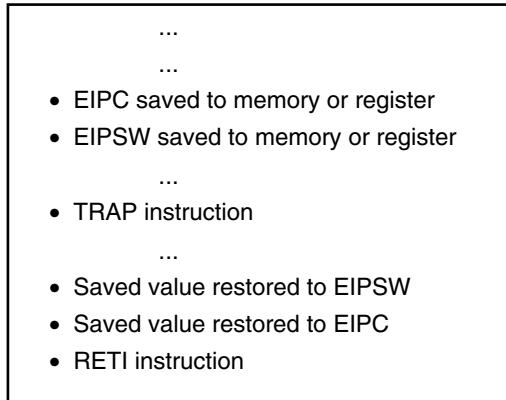
(1) Acknowledgement of maskable interrupts in servicing program

Service program of maskable interrupt or exception



(2) Generation of exception in service program

Service program of maskable interrupt or exception



← Exception such as TRAP instruction acknowledged.

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), but it can be set as desired via software. The priority order is set using the xxPRn0 to xxPRn2 bits of the interrupt control request register (xxICn), which is provided for each maskable interrupt request. After system reset, 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 servicing of the higher priority interrupt 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.

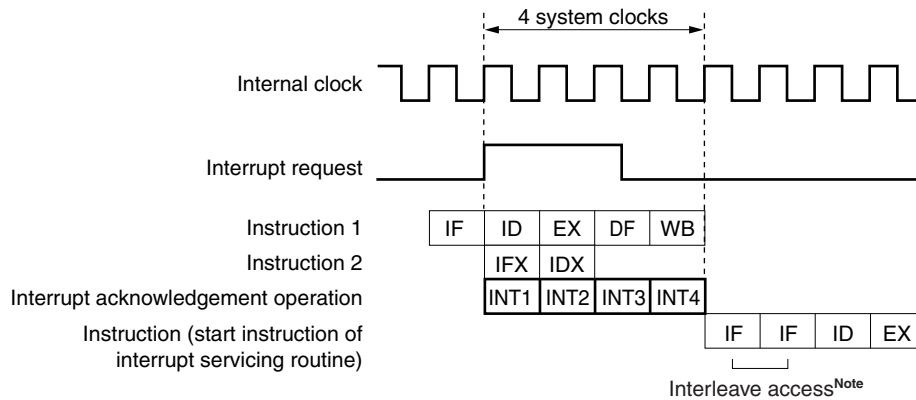
Caution In a non-maskable interrupt servicing routine (time until the RETI instruction is executed), maskable interrupts are suspended and not acknowledged.

Remark xx: Identification name of each peripheral unit (refer to **Table 7-2**)
n: Peripheral unit number (refer to **Table 7-2**)

7.7 Interrupt Latency Time

The following table describes the V850E/MA2 interrupt latency time (from interrupt generation to start of interrupt servicing).

★ **Figure 7-14. Pipeline Operation at Interrupt Request Acknowledgement (Outline)**



Note For interleave access, refer to 8.1.2 2-clock branch in the V850E1 Architecture User's Manual.

Remark INT1 to INT4: Interrupt acknowledgement servicing
 IFX: Invalid instruction fetch
 IDX: Invalid instruction decode

	Interrupt latency time (internal system clock)			Condition
	Internal interrupt	External interrupt		
		INTP0n	INTP1n	
Minimum	4	7 + Analog delay time	4 + Analog delay time	The following cases are exceptions. <ul style="list-style-type: none"> • In IDLE/software STOP mode • External bus access • Two or more interrupt request non-sample instructions are executed in succession • Access to peripheral I/O register
Maximum	7	10 + Analog delay time	7 + Analog delay time	

Remark n = 00, 01, 10, 11, m = 00, 01, 10

7.8 Periods in Which Interrupts Are 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 (interrupt is held pending).

The interrupt request non-sampling instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- Store instruction for the command register (PRCMD)
- ★ • Load, store, or bit manipulation instructions for the following registers.
 - Interrupt-related registers:
Interrupt control register (xxICn), interrupt mask registers 0 to 3 (IMR0 to IMR3), in-service priority register (ISPR)
 - CSI-related registers:
Clocking serial interface clock selection registers 0, 1 (CSIC0, CSIC1), clocked serial interface mode registers 0, 1 (CSIM0, CSIM1), serial I/O shift registers 0, 1 (SIO0, SIO1), receive-only serial I/O shift registers 0, 1 (SIOE0, SIOE1), clocked serial interface transmit buffer registers 0, 1 (SOTB0, SOTB1)

CHAPTER 8 PRESCALER UNIT (PRS)

The prescaler divides the internal system clock and supplies the divided clock to internal peripheral units. The divided clock differs depending on the unit.

For the timer units and A/D converter, a 2-division clock is used. For other units, the input clock should be selected using that unit's control register.

The CPU operates with the internal system clock.

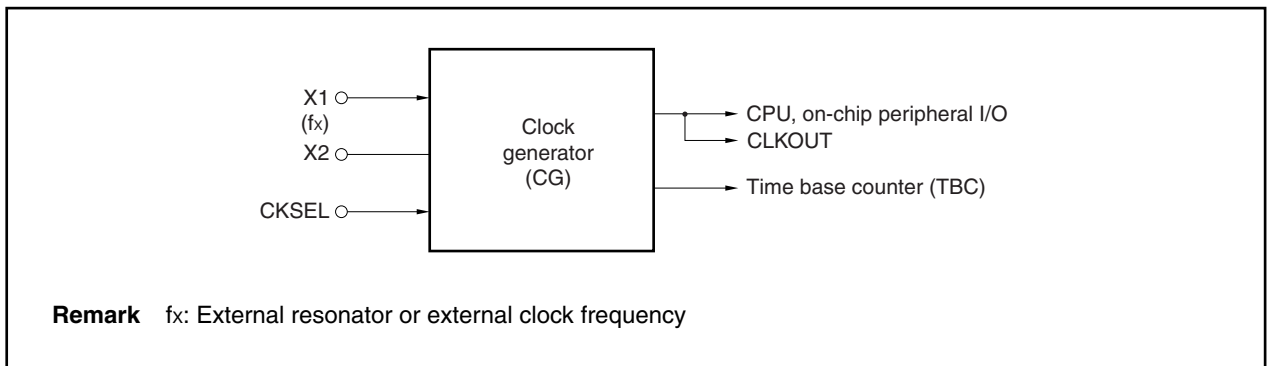
CHAPTER 9 CLOCK GENERATOR FUNCTION

The clock generator (CG) generates and controls the internal system clock (fx) that is supplied to each internal unit, such as the CPU.

9.1 Features

- Multiplier function using a phase locked loop (PLL) synthesizer
- Clock sources
 - Oscillation by connecting a resonator
 - External clock
- Power saving modes
 - HALT mode
 - IDLE mode
 - Software STOP mode
- Internal system clock output function

9.2 Configuration



9.3 Input Clock Selection

The clock generator consists of an oscillator and a PLL synthesizer. For example, connecting a 4.0 MHz crystal resonator or ceramic resonator to pins X1 and X2 enables a 40 MHz internal system clock (f_{xx}) to be generated when the multiplier is 10.

Also, an external clock can be input directly to the oscillator. In this case, the clock signal should be input only to pin X1 (pin X2 should be left open). Two basic operation modes are provided for the clock generator. These are PLL mode and direct mode. The operation mode is selected by the CKSEL pin. The input to this pin is latched on reset.

CKSEL	Operating Mode
0	PLL mode
1	Direct mode

Caution The input level for the CKSEL pin must be fixed. If it is switched during operation, a malfunction may occur.

9.3.1 Direct mode

In direct mode, an external clock having twice the frequency of the internal system clock is input. The maximum frequency that can be input in direct mode is 40 MHz. The V850E/MA2 is mainly used in application systems which operate at relatively low frequencies.

Caution In direct mode, an external clock must be input (an external resonator should not be connected).

9.3.2 PLL mode

In PLL mode, an external resonator is connected or external clock is input and multiplied by the PLL synthesizer. The multiplied PLL output is divided by the division ratio specified by the clock control register (CKC) to generate a system clock that is 10, 5, 2.5, or 1 times the frequency (fx) of the external resonator or external clock.

After reset, an internal system clock (fxx) that is 1 time the frequency (1 × fx) of the internal clock frequency (fx) is generated.

When a frequency (10 × fx) that is 10 times the clock frequency (fx) is generated, a system with low noise and low power consumption can be realized because a frequency of up to 40 MHz is obtained based on a 4 MHz external resonator or external clock.

In PLL mode, if the clock supply from an external resonator or external clock source stops, operation of the internal system clock (fxx) based on the self-propelled frequency of the clock generator’s internal voltage controlled oscillator (VCO) continues. However, do not devise an application method expecting to use this self-propelled frequency.

Example: Clocks when PLL mode (fxx = 10 × fx) is used

System Clock Frequency (fxx)	External Resonator or External Clock Frequency (fx)
40.000 MHz	4.0000 MHz

Caution Only an fx (4 MHz) value for which 10 × fx does not exceed the system clock maximum frequency (40 MHz) can be used for the oscillation frequency or external clock frequency. However, if any of 5 × fx, 2.5 × fx, or 1 × fx is used, a frequency of 4 to 6.6 MHz can be used.

Remark Note the following when PLL mode is selected (fxx = 5 × fx, fxx = 2.5 × fx, or fxx = 1 × fx)
 If the V850E/MA2 does not need to be operated at high frequency, use fxx = 5 × fx, fxx = 2.5 × fx, or fxx = 1 × fx to reduce the power consumption by lowering the system clock frequency using software.

9.3.3 Peripheral command register (PHCMD)

This is an 8-bit register that is used to set protection for writing to registers that can significantly affect the system so that the application system is not halted unexpectedly due to erroneous program execution. This register can be written only in 8-bit units (when it is read, undefined data is read out).

Writing to the first specific register (CKC register) is only valid after first writing to the PHCMD register. Because of this, the register value can be overwritten only with the specified sequence, preventing an illegal write operation from being performed.

	7	6	5	4	3	2	1	0	Address	After reset
PHCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0	FFFFFF800H	Undefined

Bit Position	Bit Name	Function
7 to 0	REG7 to REG0	Registration Code (arbitrary 8-bit data) The specific registers targeted are as follows. • Clock control register (CKC)

The generation of an illegal store operation can be checked with the PRERR bit of the peripheral status register (PHS).

9.3.4 Clock control register (CKC)

The clock control register is an 8-bit register that controls the internal system clock (f_{xx}) in PLL mode. It can be written to only by a specific sequence combination so that it cannot easily be overwritten by mistake due to erroneous program execution.

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

Caution Do not change bits CKDIV2 to CKDIV0 in direct mode.

	7	6	5	4	3	2	1	0	Address	After reset
CKC	0	0	TBCS	CESEL	0	CKDIV2	CKDIV1	CKDIV0	FFFFFF822H	00H

Bit Position	Bit Name	Function																								
5	TBCS	Time Base Count Select Selects the time base counter clock. 0: $f_x/2^8$ 1: $f_x/2^9$ For details, see 9.6.2 Time base counter (TBC) .																								
4	CESEL	Crystal/External Select Specifies the functions of the X1 and X2 pins. 0: A resonator is connected to the X1 and X2 pins 1: An external clock is connected to the X1 pin When CESEL = 1, the oscillator feedback loop is disconnected to prevent current leak in software STOP mode.																								
2 to 0	CKDIV2 to CKDIV0	Clock Divide Sets the internal system clock (f_{xx}) when PLL mode is used. <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">CKDIV2</th> <th style="width: 10%;">CKDIV1</th> <th style="width: 10%;">CKDIV0</th> <th style="width: 70%;">Internal system clock (f_{xx})</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>f_x</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>$2.5 \times f_x$</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>$5 \times f_x$</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>$10 \times f_x$</td> </tr> <tr> <td colspan="3" style="text-align: center;">Other than above</td> <td style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table> </div> <p>To change the internal system clock frequency in the middle of an operation, be sure to set it to f_x once, and then change the frequency as desired.</p>	CKDIV2	CKDIV1	CKDIV0	Internal system clock (f_{xx})	0	0	0	f_x	0	0	1	$2.5 \times f_x$	0	1	1	$5 \times f_x$	1	1	1	$10 \times f_x$	Other than above			Setting prohibited
CKDIV2	CKDIV1	CKDIV0	Internal system clock (f_{xx})																							
0	0	0	f_x																							
0	0	1	$2.5 \times f_x$																							
0	1	1	$5 \times f_x$																							
1	1	1	$10 \times f_x$																							
Other than above			Setting prohibited																							

Example Clock generator settings

Operation Mode	CKSEL Pin	CKC Register			Input Clock (f_x)	Internal System Clock (f_{xx})
		CKDIV2	CKDIV1	CKDIV0		
Direct mode	High-level input	0	0	0	16 MHz	8 MHz
PLL mode	Low-level input	0	0	0	4 MHz	4 MHz
		0	0	1	4 MHz	10 MHz
		0	1	1	4 MHz	20 MHz
		1	1	1	4 MHz	40 MHz
Other than above					Setting prohibited	Setting prohibited

Data is set in the clock control register (CKC) according to the following sequence.

- <1> Disable interrupts (set the NP bit of PSW to 1)
- <2> Prepare data in any one of the general-purpose registers to set in the specific register.
- <3> Write data to the peripheral command register (PHCMD)
- <4> Set the clock control register (CKC) (with the following instructions).
 - Store instructions (ST/SST instruction)
- <5> Assert the NOP instructions (5 instructions (<5> to <9>))
- <10> Release the interrupt disabled state (set the NP bit of PSW to 0).

```
[Sample coding]    <1> LDSR    rX, 5
                   <2> MOV     0X07, r10
                   <3> ST.B    r10, PHCMD [r0]
                   <4> ST.B    r10, CKC [r0]
                   <5> NOP
                   <6> NOP
                   <7> NOP
                   <8> NOP
                   <9> NOP
                   <10> LDSR   rY, 5
```

Remark rX: Value written to PSW
rY: Value returned to PSW

No special sequence is required to read the specific register.

- Cautions**
1. If an interrupt is acknowledged between the issuing of data to the PHCMD <3> and writing to the specific register immediately after <4>, the write operation to the specific register is not performed and a protection error (the PRERR bit of the PHS register = 1) may occur. Therefore, set the NP bit of the PSW to 1 <1> to disable interrupt acknowledgement. Also disable interrupt acknowledgement as well when selecting a bit manipulation instruction for the specific register setting.
 2. Although the data written to the PHCMD register is dummy data however, use the same register as the general-purpose register used in specific register setting <4> for writing to the PHCMD register (<3>). The same method should be applied when using a general-purpose register for addressing.
 3. Before executing this processing, complete all DMA transfers.

9.3.5 Peripheral status register (PHS)

If a write operation is not performed in the correct sequence including access to the command register for the protection-targeted internal registers, writing is not performed and a protection error is generated, setting the status flag (PRERR) to 1. This flag is a cumulative flag. After checking the PRERR flag, it is cleared to 0 by an instruction.

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

	7	6	5	4	3	2	1	<0>	Address	After reset
PHS	0	0	0	0	0	0	0	PRERR	FFFFFF802H	00H

Bit Position	Bit Name	Function
0	PRERR	Protection Error 0: Protection error does not occur 1: Protection error occurs

The operation conditions of the PRERR flag are as follows.

- ★ Set conditions:
 - <1> If the operation of the relevant store instruction for the peripheral I/O is not a write operation for the PHCMD register, but a peripheral specific register is written to.
 - <2> If the first store instruction operation after the write operation to the PHCMD register is for memory other than the specific registers and peripheral I/O.

- Reset conditions:
- <1> If the PRERR flag of the PHS register is set to 0.
 - <2> If the system is reset

9.4 PLL Lockup

The lockup time (frequency stabilization time) is the time from when the power is turned on or the software STOP mode is released until the phase locks at the prescribed frequency. The state until this stabilization occurs is called a lockup state, and the stabilized state is called a lock state.

The lock register (LOCKR) has a LOCK flag that reflects the stabilized state of the PLL frequency.

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

Caution If the phase is locked, the LOCK flag is cleared to 0. If it is unlocked later because of a standby status, the LOCK flag is set to 1. If the phase is unlocked by a cause other than the standby status, however, the LOCK flag is not affected (LOCK = 0).

	7	6	5	4	3	2	1	<0>	Address	After reset
LOCKR	0	0	0	0	0	0	0	LOCK	FFFFF824H	0000000xB

Bit Position	Bit Name	Function
0	LOCK	Lock Status Flag This is a read-only flag that indicates the PLL lock state. This flag holds the value 0 as long as a lockup state is maintained and is not initialized by a system reset. 0: Indicates that the PLL is locked. 1: Indicates that the PLL is not locked (UNLOCK state).

If the clock stops, the power fails, or some other factor operates to cause an unlock state to occur, for control processing that depends on software execution speed, such as real-time processing, be sure to judge the LOCK flag according to software immediately after operation begins so that processing does not begin until after the clock stabilizes.

On the other hand, static processing such as the setting of internal hardware or the initialization of register data or memory data can be executed 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 frequency stabilizes) is shown below.

Oscillation stabilization time < PLL lockup time.

9.5 Power Save Control

9.5.1 Overview

The power save function has 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. Since the supply of clocks to on-chip peripheral functions other than the CPU continues, operation continues. The power consumption of the overall system can be reduced by intermittent operation that is achieved due to a combination of HALT mode and normal operation mode.

The system is switched to HALT mode by a specific instruction (the HALT instruction).

(2) IDLE mode

In this mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but the supply of internal system clocks is stopped, which causes the overall system to stop.

When the system is released from IDLE mode, it can be switched to normal operation mode quickly because the oscillator's oscillation stabilization time need not be secured.

The system is switched to IDLE mode according to the PSMR register setting.

IDLE mode is located midway between software STOP mode and HALT mode in relation to the clock stabilization time and current consumption. It is used for situations in which a low current consumption mode is to be used and the clock stabilization time is to be eliminated after the mode is released.

(3) Software STOP mode

In this mode, the overall system is stopped by stopping the clock generator (oscillator and PLL synthesizer).

The system enters an ultra-low power consumption state in which only leak current is lost.

The system is switched to software STOP mode according to a PSMR register setting.

(a) PLL mode

The system is switched to software STOP mode by setting the register according to software. The PLL synthesizer's clock output is stopped at the same time that the oscillator is stopped. After software STOP mode is released, the oscillator's oscillation stabilization time must be secured until the system clock stabilizes. Also, PLL lockup time may be required depending on the program. When a resonator or external clock is connected, following the release of the software STOP mode, execution of the program is started after the count time of the time base counter has elapsed.

(b) Direct Mode

To stop the clock, set the X1 pin to low level. After the release of software STOP mode, execution of the program is started after the count-time of the time base counter has elapsed.

Table 9-1 shows the operation of the clock generator in normal operation mode, HALT mode, IDLE mode, and software STOP mode.

An effective low power consumption system can be realized by combining these modes and switching modes according to the required use.

Figure 9-1. Power Save Mode State Transition Diagram

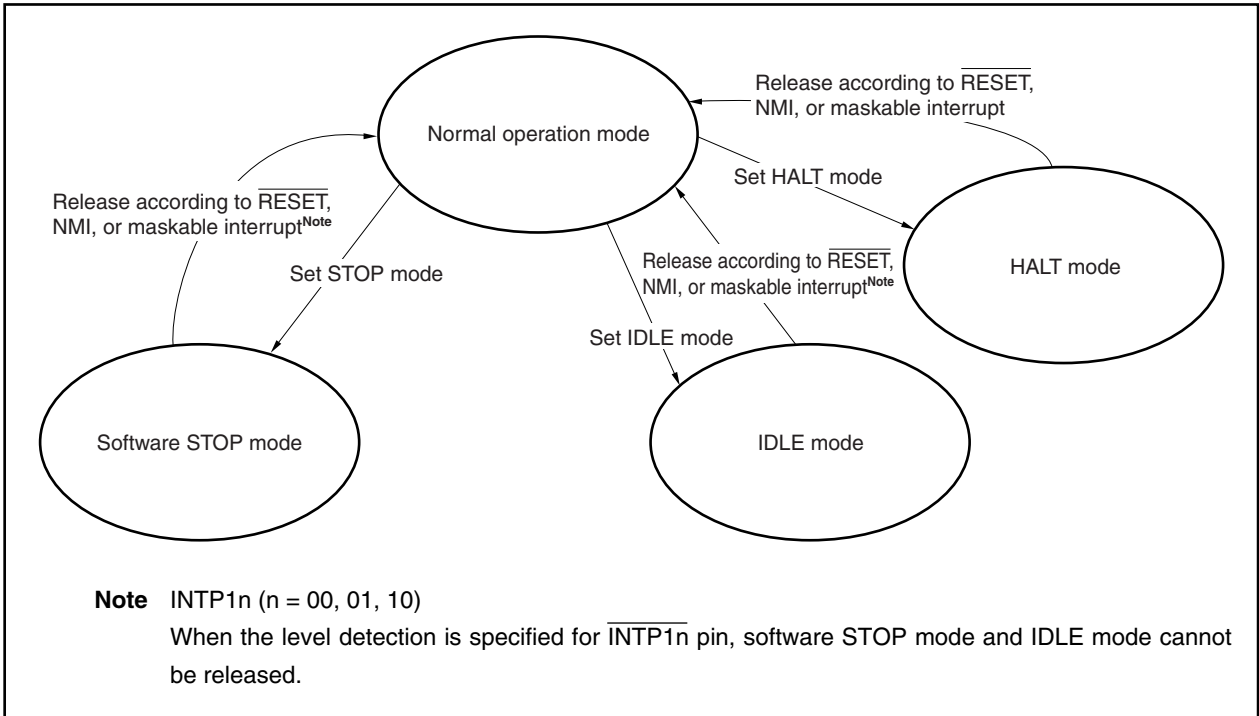


Table 9-1. Clock Generator Operation Using Power Save Control

Clock Source		Power Save Mode	Oscillator	PLL Synthesizer	Clock Supply to Peripheral I/O	Clock Supply to the CPU
PLL mode	Oscillation with resonator	Normal Operation	√	√	√	√
		HALT mode	√	√	√	–
		IDLE mode	√	√	–	–
		Software STOP mode	–	–	–	–
	External clock	Normal Operation	–	–	√	√
		HALT mode	–	–	√	–
		IDLE mode	–	–	√	–
		Software STOP mode	–	–	–	–
Direct mode	External clock	Normal Operation	–	–	√	√
		HALT mode	–	–	√	–
		IDLE mode	–	–	–	–
		Software STOP mode	–	–	–	–

Remark √: Operating
–: Stopped

9.5.2 Control registers

(1) Power save mode register (PSMR)

This is an 8-bit register that controls power save mode. It is effective only when the STB bit of the PSC register is set to 1.

Writing to the PSMR register is executed by the store instruction (ST/SST instruction) and a bit manipulation instruction (SET1/CLR1/NOT1 instruction).

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

	7	6	5	4	3	2	1	<0>	Address	After reset
PSMR	0	0	0	0	0	0	0	PSM	FFFFF820H	00H

Bit Position	Bit Name	Function
0	PSM	Power Save Mode Specifies IDLE mode or software STOP mode. 0: Switches the system to IDLE mode 1: Switches the system to software STOP mode

(2) Command register (PRCMD)

This is an 8-bit register that is used to set protection for write operations to registers that can significantly affect the system so that the application system is not halted unexpectedly due to erroneous program execution. Writing to the first specific register (power save control register (PSC)) is only valid after first writing to the PRCMD register. Because of this, the register value can be overwritten only by the specified sequence, preventing an illegal write operation from being performed.

This register can only be written in 8-bit units. The undefined data is read out if read.

	7	6	5	4	3	2	1	0	Address	After reset
PRCMD	REG7	REG6	REG5	REG4	REG3	REG2	REG1	REG0	FFFFF1FCH	Undefined

Bit Position	Bit Name	Function
7 to 0	REG7 to REG0	Registration Code (arbitrary 8-bit data) The specific register targeted is the power save control register (PSC).

(3) Power save control register (PSC)

This is an 8-bit register that controls the power save function. This register, which is one of the specific registers, is effective only when accessed by a specific sequence during a write operation.

This register can be read or written in 8-bit or 1-bit units. If bit 7 or 6 is set to 1, operation cannot be guaranteed.

Caution It is impossible to set STB bit and NMIM or INTM bit at the same time. Be sure to set STB bit after setting NMIM or INTM bit.

	7	6	<5>	<4>	3	2	<1>	0	Address	After reset
PSC	0	0	NMIM	INTM	0	0	STB	0	FFFFFF1FEH	00H

Bit Position	Bit Name	Function
5	NMIM	NMI Mode This is the enable/disable setting bit for standby mode release using valid edge input of NMI. 0: Enables NMI cancellation 1: Disables NMI cancellation
4	INTM	INT Mode This is the enable/disable setting for standby mode release using an unmasked maskable interrupt (INTP1n) (n = 00, 01, 10). 0: Enables maskable interrupt cancellation 1: Disables maskable interrupt cancellation
1	STB	Stand-by Mode Indicates the stand-by mode status. If 1 is written to this bit, the system enters IDLE or software STOP mode (using the PSM bit of the PSMR register). When stand-by mode is released, this bit is automatically reset to 0. 0: Stand-by mode is released 1: Stand-by mode is in effect

Data is set in the power save control register (PSC) according to the following sequence.

- <1> Set the power save mode register (PSMR) (with the following instructions).
 - Store instruction (ST/SST instruction)
 - Bit manipulation instruction (SET1/CLR1/NOT1 instruction)
- <2> Prepare data in any one of the general-purpose registers to set to the specific register.
- <3> Write data to the command register (PRCMD).
- <4> Set the power save control register (PSC) (with the following instructions).
 - Store instruction (ST/SST instruction)
 - Bit manipulation instruction (SET1/CLR1/NOT1 instruction)
- <5> Assert the NOP instructions (5 instructions (<5> to <9>)).

```

[Sample coding]  <1> ST.B  r11, PSMR [r0]  ; Set PSMR register
                  <2> MOV   0x02, r10
                  <3> ST.B  r10, PRCMD [r0] ; Write PRCMD register
                  <4> ST.B  r10, PSC [r0]  ; Set PSC register
                  <5> NOP                    ; Dummy instruction
                  <6> NOP                    ; Dummy instruction
                  <7> NOP                    ; Dummy instruction
                  <8> NOP                    ; Dummy instruction
                  <9> NOP                    ; Dummy instruction
                  (next instruction)        ; Execution routine after software STOP mode and
                                          IDLE mode release

```

No special sequence is required to read the specific register.

- Cautions**
1. A store instruction for the command register does not acknowledge interrupts. This coding is made on assumption that <3> and <4> above are executed by the program with consecutive store instructions. If another instruction is set between <3> and <4>, the above sequence may become in effective when the interrupt is acknowledged by that instruction, and a malfunction of the program may result.
 2. Although the data written to the PRCMD register is dummy data, use the same register as the general-purpose register used in specific register setting <4> for writing to the PRCMD register (<3>). The same method should be applied when using a general-purpose register for addressing.
 3. At least 5 NOP instructions must be inserted after executing a store instruction to the PSC register to set software STOP or IDLE mode.
 4. Before executing this processing, complete all DMA transfers.

9.5.3 HALT mode

(1) Setting and operation status

In HALT mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but the operation clock of the CPU is stopped. Since the supply of clocks to on-chip peripheral I/O units other than the CPU continues, operation continues. The power consumption of the overall system can be reduced by setting the system to HALT mode while the CPU is idle.

The system is switched to HALT mode by the HALT instruction.

Although program execution stops in HALT mode, the contents of all registers, internal RAM, and ports are maintained in the state they were in immediately before HALT mode began. Also, operation continues for all on-chip peripheral I/O units (other than ports) that do not depend on CPU instruction processing. Table 9-2 shows the status of each hardware unit in HALT mode.

Caution If the HALT instruction is executed while an interrupt is being held pending, the HALT mode is set once but it is immediately released by the pending interrupt request.

Table 9-2. Operation Status in HALT Mode

Function	Operation Status	
Clock generator	Operating	
Internal system clock	Operating	
CPU	Stopped	
Ports	Maintained	
On-chip peripheral I/O (excluding ports)	Operating	
Internal data	All internal data such as CPU registers, statuses, data, and the contents of internal RAM are maintained in the state they were in immediately before HALT mode began.	
D0 to D15	Operating	
A0 to A24		
\overline{RD} , \overline{WE}		
\overline{UWR} , \overline{LWR}		
\overline{LDQM} , \overline{UDQM}		
$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$		
\overline{SDRAS}		
\overline{SDCAS}		
\overline{REFRQ}		
\overline{HLDAK}		
\overline{HLDRQ}		
\overline{WAIT}		
SDCKE		
SDCLK		Clock output
CLKOUT		

(2) Release of HALT mode

HALT mode is released by a non-maskable interrupt request, an unmasked maskable interrupt request, or $\overline{\text{RESET}}$ pin input.

(a) Release by non-maskable interrupt request or unmasked maskable interrupt request

HALT mode is released by a non-maskable interrupt request or by an unmasked maskable interrupt request regardless of the priority. However, if the system is set to HALT mode during an interrupt servicing routine, operation will differ as follows.

- (i) If an interrupt request is generated with a lower priority than that of the interrupt request that is currently being serviced, HALT mode is released, but the newly generated interrupt request is not acknowledged. The new interrupt request is held pending.
- (ii) If an interrupt request (including non-maskable interrupt requests) is generated with a higher priority than that of the interrupt request that is currently being serviced, HALT mode is released and the newly generated interrupt request is acknowledged.

Table 9-3. Operation After HALT Mode Is Released by Interrupt Request

Release Source	Enable Interrupt (EI) Status	Disable Interrupt (DI) Status
Non-maskable interrupt request	Branch to handler address	
Maskable interrupt request	Branch to handler address or execute next instruction	Execute next instruction

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

This is the same as a normal reset operation.

9.5.4 IDLE mode

(1) Setting and operation status

In IDLE mode, the clock generator (oscillator and PLL synthesizer) continues to operate, but the supply of internal system clocks is stopped which causes the overall system to stop.

When IDLE mode is released, the system can be switched to normal operation mode quickly because the oscillator's oscillation stabilization time or the PLL lockup time need not be secured.

The system is switched to IDLE mode by setting the PSC or PS MR register using a store instruction (ST or SST instruction) or a bit manipulation instruction (SET1, CLR1, or NOT1 instruction) (refer to **9.5.2 Control registers**).

In IDLE mode, program execution is stopped, and the contents of all registers, internal RAM, and ports are maintained in the state they were in immediately before execution stopped. The operation of on-chip peripheral I/O units (excluding ports) also is stopped.

Table 9-4 shows the status of each hardware unit in IDLE mode.

Table 9-4. Operation Status in IDLE Mode

Function	Operation Status
Clock generator	Operating
Internal system clock	Stopped
CPU	Stopped
Ports	Maintained
On-chip peripheral I/O (excluding ports)	Stopped
Internal data	All internal data such as CPU registers, statuses, data, and the contents of internal RAM are maintained in the state they were in immediately before IDLE mode began.
D0 to D15	High impedance
A0 to A24	
\overline{RD} , \overline{WE}	High-level output
\overline{UWR} , \overline{LWR}	
\overline{LDQM} , \overline{UDQM}	
$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$	
\overline{SDRAS}	Operating
\overline{SDCAS}	
\overline{REFRQ}	
\overline{HLDAK}	High-level output
\overline{HLDRQ}	Input (no sampling)
\overline{WAIT}	
\overline{SDCKE}	Low-level output
\overline{SDCLK}	
\overline{CLKOUT}	

(2) Release of IDLE mode

IDLE mode is released by a non-maskable interrupt request, an unmasked maskable interrupt request (INTP1n), or $\overline{\text{RESET}}$ pin input (n = 00, 01, 10).

(a) Release by non-maskable interrupt request or unmasked maskable interrupt request

★

IDLE mode can be released by an interrupt request only when it has been set with the INTM and NMIM bits of the PSC register cleared to 0.

IDLE mode is released by a non-maskable interrupt request or by an unmasked maskable interrupt request (INTP1n) regardless of the priority. However, if the system is set to IDLE mode during a maskable interrupt servicing routine, operation will differ as follows (n = 00, 01, 10).

- (i) If an interrupt request is generated with a lower priority than that of the interrupt request that is currently being serviced, IDLE mode is released, but the newly generated interrupt request is not acknowledged. The new interrupt request is held pending.
- (ii) If an interrupt request (including non-maskable interrupt requests) is generated with a higher priority than that of the interrupt request that is currently being serviced, IDLE mode is released and the newly generated interrupt request is acknowledged.

Table 9-5. Operation After IDLE Mode Is Released by Interrupt Request

Release Source	Enable Interrupt (EI) Status	Disable Interrupt (DI) Status
Non-maskable interrupt request	Branch to handler address	
Maskable interrupt request	Branch to handler address or execute next instruction	Execute next instruction

If the system is set to IDLE mode during an NMI servicing routine, IDLE mode is released, but the interrupt is not acknowledged (interrupt is held pending).

Interrupt servicing that is started when IDLE mode is released by NMI pin input is handled in the same way as normal NMI interrupt servicing that occurs during an emergency (because the NMI interrupt handler address is unique). Therefore, when a program must be able to distinguish between these two situations, a software status must be prepared in advance and that status must be set before setting the PSMR register using a store instruction or a bit manipulation instruction. By checking for this status during NMI interrupt servicing, an ordinary NMI can be distinguished from the processing that is started when IDLE mode is released by NMI pin input.

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

This is the same as a normal reset operation.

9.5.5 Software STOP mode

(1) Setting and operation status

In software STOP mode, the clock generator (oscillator and PLL synthesizer) is stopped. The overall system is stopped, and ultra-low power consumption is achieved in which only leak current is lost.

The system is switched to software STOP mode by using a store instruction (ST or SST instruction) or bit manipulation instruction (SET1, CLR1, or NOT1 instruction) to set the PSC and PSMR registers (refer to **9.5.2 Control registers**).

When PLL mode and resonator connection mode (CESEL bit of CKC register = 0) are used, the oscillator's oscillation stabilization time must be secured after software STOP mode is released.

In both PLL and direct mode, following the release of software STOP mode, execution of the program is started after the count time of the time base counter has elapsed.

Although program execution stops in software STOP mode, the contents of all registers, internal RAM, and ports are maintained in the state they were in immediately before software STOP mode began. The operation of all on-chip peripheral I/O units (excluding ports) is also stopped.

Table 9-6 shows the status of each hardware unit in software STOP mode.

Table 9-6. Operation Status in Software STOP Mode

Function	Operation Status
Clock generator	Stopped
Internal system clock	Stopped
CPU	Stopped
Ports	Maintained ^{Note}
On-chip peripheral I/O (excluding ports)	Stopped
Internal data	All internal data such as CPU registers, statuses, data, and the contents of internal RAM are maintained in the state they were in immediately before software STOP mode began.
D0 to D15	High impedance
A0 to A24	
\overline{RD} , \overline{WE}	High-level output
\overline{UWR} , \overline{LWR}	
\overline{LDQM} , \overline{UDQM}	
$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$	
\overline{SDRAS}	Operating
\overline{SDCAS}	
\overline{REFRQ}	
\overline{HLDAK}	High-level output
\overline{HLDRQ}	Input (no sampling)
\overline{WAIT}	
\overline{SDCKE}	Low-level output
\overline{SDCLK}	
\overline{CLKOUT}	

Note When the V_{DD} value is within the operable range. However, even if it drops below the minimum operable voltage, as long as the data retention voltage V_{DDR} is maintained, the contents of only the internal RAM will be maintained.

(2) Release of software STOP mode

Software STOP mode is released by a non-maskable interrupt request, an unmasked maskable interrupt request (INTP1n), or $\overline{\text{RESET}}$ pin input. Also, to release software STOP mode when PLL mode (CKSEL pin = low level) and resonator connection mode (CESEL bit of CKC register = 0) are used, the oscillator's oscillation stabilization time must be secured (n = 00, 01, 10).

Moreover, the oscillation stabilization time must be secured even when an external clock is connected (CESEL bit = 1). See **9.4 PLL Lockup** for details.

(a) Release by non-maskable interrupt request or unmasked maskable interrupt request

★ Software STOP mode can be released by an interrupt request only when it has been set with the INTM and NMIM bits of the PSC register cleared to 0.

Software STOP mode is released by a non-maskable interrupt request or by an unmasked maskable interrupt request (INTP1n) regardless of the priority. However, if the system is set to software STOP mode during an interrupt servicing routine, operation will differ as follows (n = 00, 01, 10).

- (i) If an interrupt request is generated with a lower priority than that of the interrupt request that is currently being servicing, software STOP mode is released, but the newly generated interrupt request is not acknowledged. The new interrupt request is held pending.
- (ii) If an interrupt request (including non-maskable interrupt requests) is generated with a higher priority than that of the interrupt request that is currently being serviced, software STOP mode is released and the newly generated interrupt request is acknowledged.

Table 9-7. Operation After Software STOP Mode Is Released by Interrupt Request

Release Source	Enable Interrupt (EI) Status	Disable Interrupt (DI) Status
Non-maskable interrupt request	Branch to handler address	
Maskable interrupt request	Branch to handler address or execute next instruction	Execute next instruction

If the system is set to software STOP mode during an NMI servicing routine, software STOP mode is released, but the interrupt is not acknowledged (interrupt is held pending).

Interrupt servicing that is started when software STOP mode is released by NMI pin input is handled in the same way as normal NMI interrupt servicing that occurs during an emergency (because the NMI interrupt handler address is unique). Therefore, when a program must be able to distinguish between these two situations, a software status must be prepared in advance and that status must be set before setting the PSMR register using a store instruction or a bit manipulation instruction.

By checking for this status during NMI interrupt servicing, an ordinary NMI can be distinguished from the servicing that is started when software STOP mode is released by NMI pin input.

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

This is the same as a normal reset operation.

9.6 Securing Oscillation Stabilization Time

9.6.1 Oscillation stabilization time security specification

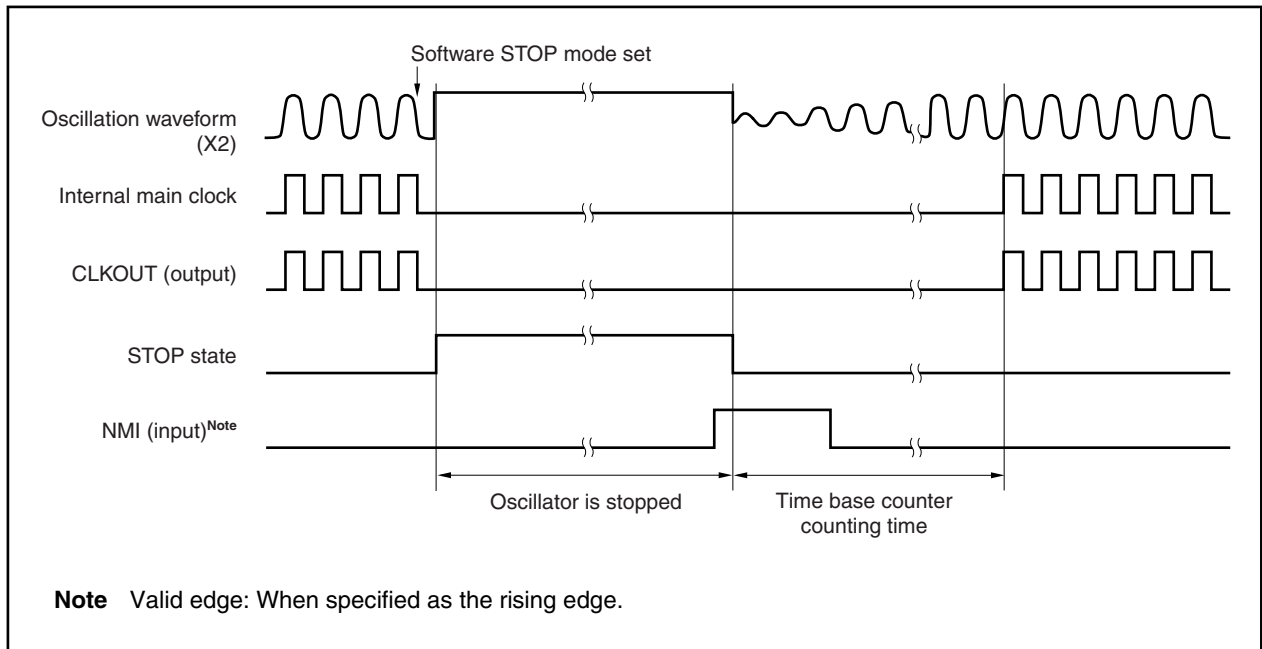
Two specification methods can be used to secure the time from when software STOP mode is released until the stopped oscillator stabilizes.

(1) Securing the time using an on-chip time base counter

Software STOP mode is released when a valid edge is input to the NMI pin or a maskable interrupt request is input (INTP1n). If oscillation is started by inputting an active edge to the pin, the time base counter (TBC) starts counting, and the time until the clock output from the oscillator stabilizes is secured during that counting time ($n = 00, 01, 10$).

Oscillation stabilization time = TBC counting time

After a fixed time, internal system clock output begins, and processing branches to the NMI interrupt or maskable interrupt (INTP1n) handler address ($n = 00, 01, 10$).



The NMI pin should usually be set to an inactive level (for example, high level when the valid edge is specified as the falling edge) in advance.

Software STOP mode is immediately released by NMI valid edge input or maskable interrupt request input (INTP1n) if software STOP mode is set before the CPU acknowledges the interrupt ($n = 00, 01, 10$).

If direct mode or external clock connection mode (CESEL bit of CKC register = 1) is used, program execution begins after the count time of the time base counter has elapsed.

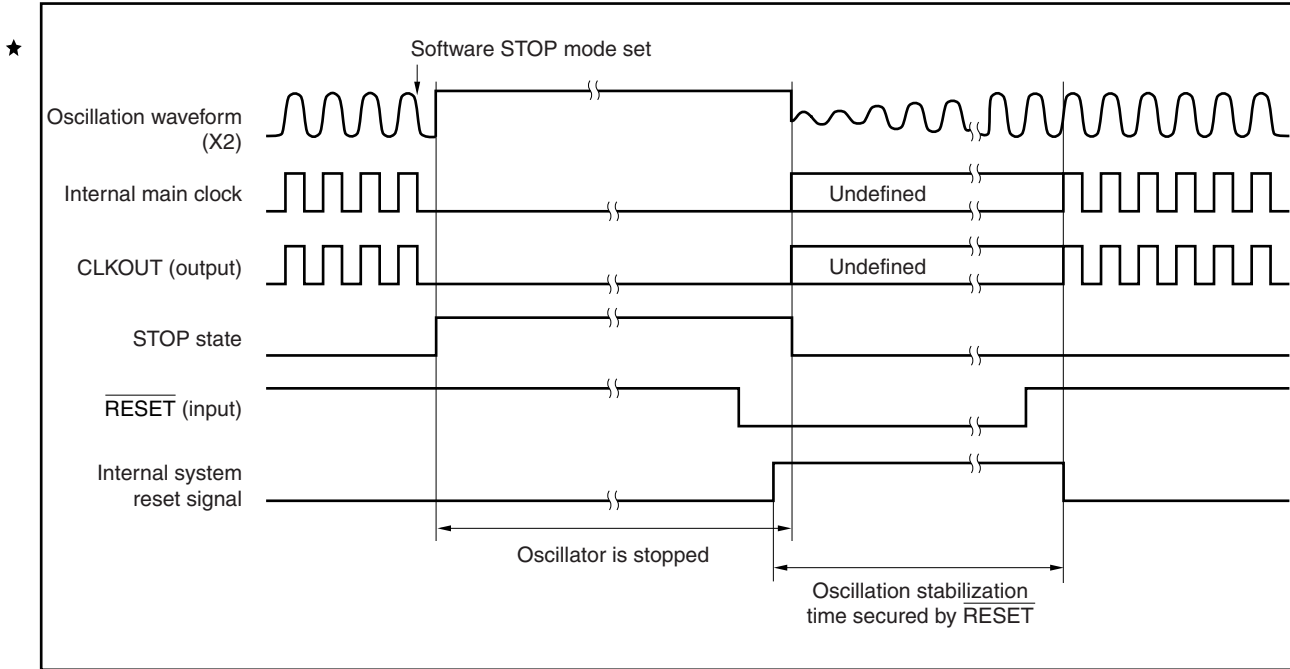
Also, even if PLL mode and resonator connection mode (CESEL bit of CKC register = 0) are used, program execution begins after the oscillation stabilization time is secured according to the time base counter, which begins counting due to NMI pin valid edge input.

(2) Securing the time according to the signal level width ($\overline{\text{RESET}}$ pin input)

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

The time until the clock output from the oscillator stabilizes is secured according to the low level width of the signal that is input to the pin.

The supply of internal system clocks begins after a rising edge is input to the $\overline{\text{RESET}}$ pin, and processing branches to the handler address used for a system reset.



9.6.2 Time base counter (TBC)

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

When an external clock is connected (CESEL bit of CKC register = 1) or a resonator is connected (PLL mode and CESEL bit of CKC register = 0), the TBC counts the oscillation stabilization time after software STOP mode is released, and program execution begins after the count is completed.

The TBC count clock is selected according to the TBCS bit of the CKC register, and the next counting time can be set (reference).

Table 9-8. Counting Time Examples ($f_{xx} = 10 \times f_x$)

TBCS Bit	Count Clock	Counting Time
		$f_x = 4.0000 \text{ MHz}$
		$f_{xx} = 40.000 \text{ MHz}$
0	$f_x/2^9$	16.3 ms
1	$f_x/2^9$	32.6 ms

f_x : External oscillation frequency

f_{xx} : Internal system clock

CHAPTER 10 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)

10.1 Timer C

10.1.1 Features (timer C)

Timer C is a 16-bit timer/counter that can perform the following operations.

- Interval timer function
- PWM output
- External signal cycle measurement

10.1.2 Function overview (timer C)

- 16-bit timer/counter
- Capture/compare common registers: 4
- Interrupt request sources
 - Capture/match interrupt requests: 4
 - Overflow interrupt requests: 2
- Timer/counter count clock sources: 2
(Selection of external pulse input or internal system clock division)
- Either free-running mode or overflow stop mode can be selected as the operation mode when the timer/counter overflows
- Timer/counter can be cleared by a match of the timer/counter and a compare register
- External pulse outputs: 1

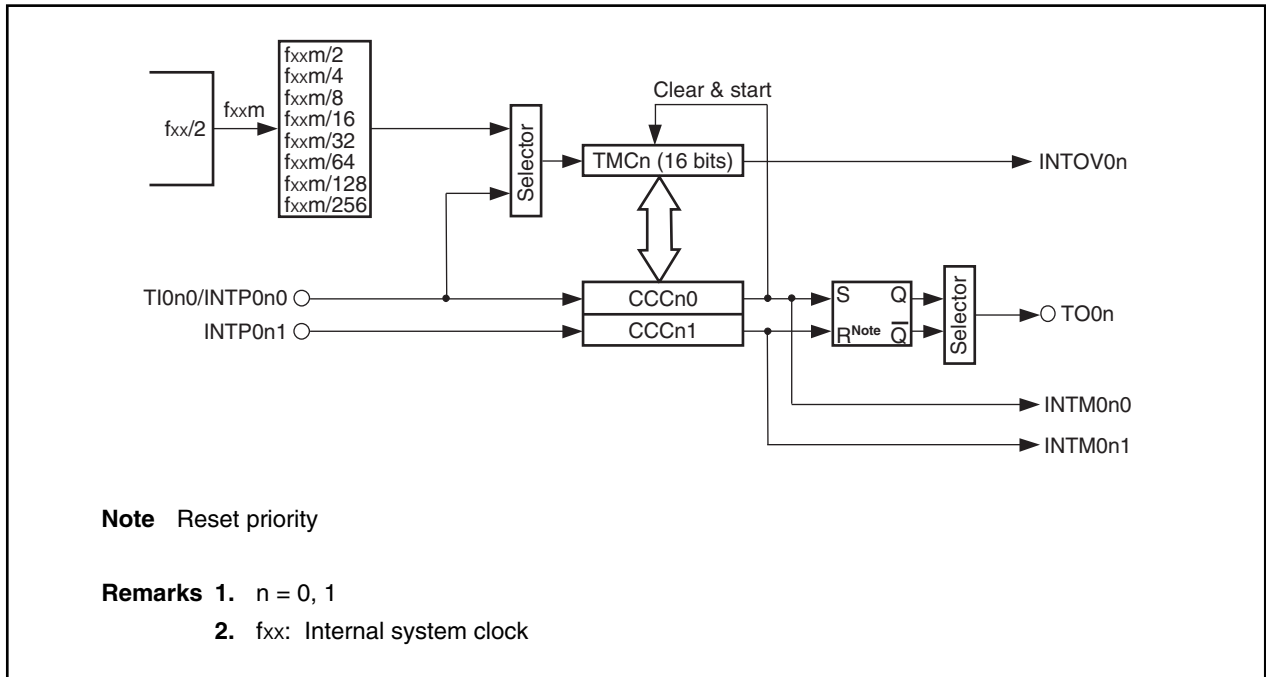
10.1.3 Basic configuration of timer C

Table 10-1. Timer C Configuration

Timer	Count Clock	Register	Read/Write	Generated Interrupt Signal	Capture Trigger	Timer Output S/R	Other Functions
Timer C	fxx/4, fxx/8, fxx/16, fxx/32, fxx/64, fxx/128, fxx/256, fxx/512,	TMC0	Read	INTOV00	-	-	-
		CCC00	Read/write	INTM000	INTP000	TO00 (S)	A/D conversion start trigger
		CCC01	Read/write	INTM001	INTP001	TO00 (R)	A/D conversion start trigger
		TMC1	Read	INTOV01	-	-	-
		CCC10	Read/write	INTM010	INTP010	-	A/D conversion start trigger
		CCC11	Read/write	INTM011	INTP011	-	A/D conversion start trigger

Remarks fxx: Internal system clock
S/R: Set/reset

(1) Timer C (16-bit timer/counter)



10.1.4 Timer C

(1) Timers C0, C1 (TMC0, TMC1)

TMCn functions as a 16-bit free-running timer or as an event counter for an external signal. Besides being mainly used for cycle measurement, TMCn can be used as pulse output (n = 0, 1).

TMCn is read-only in 16-bit units.

- Cautions**
1. The TMCn register can only be read. If writing is performed to the TMCn register, the subsequent operation is undefined.
 2. If the TMCCAEn bit of the TMCCn0 register is cleared (0), a reset is performed asynchronously.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
TMC0																	FFFFF600H	0000H
TMC1																	FFFFF610H	0000H

TMCn performs the count-up operations of an internal count clock or external count clock. Timer starting and stopping are controlled by the TMCCEn bit of timer mode control register Cn0 (TMCCn0) (n = 0, 1).

The internal or external count clock is selected by the ETIn bit of timer mode control register Cn1 (TMCCn1) (n = 0, 1).

(a) Selection of the external count clock

TMCn operates as an event counter.

When the ETI bit of timer mode control register Cn1 (TMCCn1) is set (1), TMCn counts the valid edges of the external clock input (TI0n0), synchronized with the internal count clock. The valid edge is specified by valid edge selection register Cn (SESCn) (n = 0, 1).

Caution When the INTP0n0/TI0n0 pin is used as TI0n0, disable the INTP0n0 interrupt or set CCCn0 to compare mode (n = 0, 1).

(b) Selection of the internal count clock

TMCn operates as a free-running timer.

When an internal clock is specified as a count clock by timer mode control register Cn1 (TMCCn1), TMCn is counted up for each input clock cycle specified by the CSn0 to CSn2 bits of the TMCCn0 register (n = 0, 1).

A division by the prescaler can be selected for the count clock from among fxx/4, fxx/8, fxx/16, fxx/32, fxx/64, fxx/128, fxx/256, and fxx/512 by the TMCCn0 register (fxx: Internal system clock).

An overflow interrupt can be generated if the timer overflows. Also, the timer can be stopped following an overflow by setting the OSTn bit of the TMCCn1 register to 1.

Caution The count clock cannot be changed while the timer is operating.

The conditions when the TMCn register becomes 0000H are shown below.

(a) Asynchronous reset

- TMCCAEn bit of TMCCn0 register = 0
- Reset input

(b) Synchronous reset

- TMCCEn bit of TMCCn0 register = 0
- The CCCn0 register is used as a compare register, and the TMCn and CCCn0 registers match when clearing the TMCn register is enabled (CCLRn bit of the TMCCn1 register = 1)

(2) Capture/compare registers Cn0 and Cn1 (CCn0 and CCn1) (n = 0, 1)

These capture/compare registers (Cn0 and Cn1) are 16-bit registers.

They can be used as capture registers or compare registers according to the CMSn0 and CMSn1 bit specifications of timer mode control register Cn1 (TMCCn1) (n = 0, 1).

These registers can be read or written in 16-bit units. (However, write operations can only be performed in compare mode.)

	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	Address	After reset																
CCn0n	<table border="1" style="width: 100%; height: 20px;"> <tr> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </table>																	FFFFF602H, FFFFF604H	0000H
CCn1n	<table border="1" style="width: 100%; height: 20px;"> <tr> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> </tr> </table>																	FFFFF612H, FFFFF614H	0000H
Remark	n = 0, 1																		

(a) Setting these registers to capture registers (CMSn0 and CMSn1 of TMCCn1 = 0)

When these registers are set to capture registers, the valid edges of the corresponding external interrupt signals INTP0n0 and INTP0n1 are detected as capture triggers. The timer TMCn is synchronized with the capture trigger, and the value of TMCn is latched in the CCn0 and CCn1 registers (capture operation).

The valid edge of the INTP0n0 pin is specified (rising, falling, or both edges) according to the IES0n01 and IES0n00 bits of the SESCn register, and the valid edge of the INTP0n1 pin is specified according to the IES0n11 and IES0n10 bits of the SESCn register (n = 0, 1).

The capture operation is performed asynchronously relative to the count clock. The latched value is held in the capture register until the next time the capture operation is performed.

When the TMCCAEn bit of timer mode control register Cn0 (TMCCn0) is 0, 0000H is read (n = 0, 1).

If these registers are specified as capture registers, an interrupt is generated by detecting the valid edge of signals INTP0n0 and INTP0n1 (n = 0, 1).

★ **Caution** If the capture operation conflicts with the timing of disabling the TMCn register from counting (when the TMCCEn bit of the TMCCn0 register = 0), the captured data becomes undefined. In addition, the INTM0n0 and INTM0n1 interrupts do not occur (n = 0, 1).

(b) Setting these registers to compare registers (CMSn0 and CMSn1 of TMCCn1 = 1)

When these registers are set to compare registers, the TMCn and register values are compared for each count clock, and an interrupt is generated by a match. If the CCLRn bit of timer mode control register Cn1 (TMCCn1) is set (1), the TMCn value is cleared (0) at the same time as a match with the CCCn0 register (it is not cleared (0) by a match with the CCCn1 register) (n = 0, 1).

A compare register (CCC00, CCC01) is equipped with a set/reset function. The timer output (TO00) is set or reset, synchronized with the generation of a match signal.

The interrupt selection source differs according to the function of the selected register.

- Cautions**
- 1. To write to capture/compare registers Cn0 and Cn1, always set the TMCCAEn bit to 1 first. If the TMCCAEn bit is 0, the data that is written will be invalid.**
 - 2. Perform a write operation to capture/compare registers Cn0 and Cn1 after setting them to compare registers according to the TMCCn0 and TMCCn1 registers setting. If they are set to capture registers (CMSn0 and CMSn1 bits of TMCCn1 register = 0), no data is written even if a write operation is performed to CCCn0 and CCCn1.**
 - 3. When these registers are set to compare registers, INTP0n0 and INTP0n1 cannot be used (n = 0, 1).**

10.1.5 Timer C control registers

(1) Timer mode control registers C00, C10 (TMCC00, TMCC10)

The TMCCn0 registers control the operation of TMCn (n = 0, 1).

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

Be sure to set bits 3 and 2 to 0. If they are set to 1, the operation is not guaranteed.

Cautions 1. The TMCCAEn and other bits cannot be set at the same time. The other bits and the registers of the other TMCn unit should always be set after the TMCCAEn bit has been set. Also, to use external pins related to the timer function when timer C is used, be sure to set (1) the TMCCAEn bit after setting the external pins to control mode.

★ **2. When conflict occurs between an overflow and a TMCCn0 register write, the OVFn bit value becomes the value written during the TMCCn0 register write (n = 0, 1).**

(1/2)

	<7>	6	5	4	3	2	<1>	<0>	Address	After reset
TMCC00	OVF0	CS02	CS01	CS00	0	0	TMCCE0	TMCCAEO	FFFFFF606H	00H
TMCC10	OVF1	CS12	CS11	CS10	0	0	TMCCE1	TMCCAEO	FFFFFF616H	00H

Bit Position	Bit Name	Function
7	OVFn (n = 0, 1)	<p>Overflow</p> <p>This is a flag that indicates TMCn overflow (n = 0, 1).</p> <p>0: No overflow occurs 1: Overflow occurs</p> <p>When TMCn has counted up from FFFFH to 0000H, the OVFn bit becomes 1 and an overflow interrupt request (INTOVF) is generated at the same time. However, if TMCn is cleared to 0000H after a match at FFFFH when the CCCn0 register is set to compare mode (CMSn0 bit of TMCCn1 register = 1) and clearing is enabled for a match when TMCn and CCCn0 are compared (CCLRn bit of TMCCn1 register = 1), then TMCn is considered to be cleared and the OVFn bit does not become 1. Also, no INTOVF interrupt is generated.</p> <p>The OVFn bit retains the value 1 until 0 is written directly or until an asynchronous reset is performed because the TMCCAEn bit is 0. An interrupt operation due to an overflow is independent of the OVFn bit, and the interrupt request flag (OVIFn) for INTOV0n is not affected even if the OVFn bit is manipulated. If an overflow occurs while the OVFn bit is being read, the flag value changes, and the change is reflected when the next read operation occurs.</p>

Bit Position	Bit Name	Function																																				
6 to 4	CSn2 to CSn0 (n = 0, 1)	<p>Count Enable Select Selects the TMCn internal count clock (n = 0, 1).</p> <table border="1"> <thead> <tr> <th>CSn2</th> <th>CSn1</th> <th>CSn0</th> <th>Count cycle</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>f_{xx}/4</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>f_{xx}/8</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>f_{xx}/16</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>f_{xx}/32</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>f_{xx}/64</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>f_{xx}/128</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>f_{xx}/256</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>f_{xx}/512</td> </tr> </tbody> </table> <p>Caution The CSn2 to CSn0 bits must not be changed during timer operation. If they are to be changed, they must be changed after setting the TMCCEn bit to 0. If these bits are overwritten during timer operation, operation cannot be guaranteed.</p> <p>Remark f_{xx}: Internal system clock</p>	CSn2	CSn1	CSn0	Count cycle	0	0	0	f _{xx} /4	0	0	1	f _{xx} /8	0	1	0	f _{xx} /16	0	1	1	f _{xx} /32	1	0	0	f _{xx} /64	1	0	1	f _{xx} /128	1	1	0	f _{xx} /256	1	1	1	f _{xx} /512
CSn2	CSn1	CSn0	Count cycle																																			
0	0	0	f _{xx} /4																																			
0	0	1	f _{xx} /8																																			
0	1	0	f _{xx} /16																																			
0	1	1	f _{xx} /32																																			
1	0	0	f _{xx} /64																																			
1	0	1	f _{xx} /128																																			
1	1	0	f _{xx} /256																																			
1	1	1	f _{xx} /512																																			
1	TMCCEn (n = 0, 1)	<p>Count Enable Controls the operation of TMCn (n = 0, 1). 0: Count disable (stops at 0000H and does not operate) 1: Counting operation is performed</p> <p>Caution When TMCCE0 = 0, the external pulse outputs (TO00) go to inactive level (the active level of TO00 output is set by the ACTLV0 bit of the TMCC01 register).</p>																																				
0	TMCCAEn (n = 0, 1)	<p>Clock Action Enable Controls the internal count clock (n = 0, 1). 0: The entire TMCn unit is asynchronously reset. The supply of clocks to the TMCn unit stops. 1: Clocks are supplied to the TMCn unit</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. When the TMCCAEn bit is set to 0, the TMCn unit can be asynchronously reset. 2. When TMCCAEn = 0, the TMCn unit is in a reset state. Therefore, to operate TMCn, the TMCCAEn bit must be set to 1. 3. When the TMCCAEn bit is changed from 1 to 0, all registers of the TMCn unit are initialized. When the TMCCAEn is set to 1 again, the TMCn unit registers must be set again. 																																				

(2) Timer mode control registers C01, C11 (TMCC01, TMCC11)

The TMCCn1 registers control the operation of TMCn (n = 0, 1).

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

Be sure to set bit 2 of the TMCC01 register and bits 6 and 2 of the TMCC11 register to 0, and bit 5 of the TMCC11 register to 1. If they are set to other values, the operation is not guaranteed.

- Cautions**
1. The various bits of the TMCCn1 register must not be changed during timer operation. If they are to be changed, they must be changed after setting the TMCCEn bit of the TMCCn0 register to 0. If these bits are overwritten during timer operation, operation cannot be guaranteed (n = 0, 1).
 2. If the ENT01 and ACTLV0 bits are changed at the same time, a glitch (spike-shaped noise) may be generated in the TO00 pin output. Either create a circuit configuration that will not malfunction even if a glitch is generated or make sure that the ENT01 and ACTLV0 bits do not change at the same time (n = 0, 1).
 3. TO00 output is not changed by an external interrupt signal (INTP0n0 or INTP0n1). To use the TO00 signal, specify that the capture/compare registers are compare registers (CMSn0 and CMSn1 bits of TMCCn1 register = 1) (n = 0, 1).

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
★ TMCC01	OST0	ENT01	ACTLV0	ETI0	CCLR0	0	CMS01	CMS00	FFFFFF608H	20H
TMCC11	OST1	0	1	ETI1	CCLR1	0	CMS11	CMS10	FFFFFF618H	20H

Bit Position	Bit Name	Function
7	OSTn (n = 0, 1)	Overflow Stop Sets the operation when TMCn has overflowed (n = 0, 1). 0: After the overflow, counting continues (free running mode) 1: After the overflow, the timer maintains the value 0000H, and counting stops (overflow stop mode). At this time, the TMCCEn bit of TMCCn0 remains at 1. Counting is restarted by a write operation to the TMCCEn bit.
6	ENT01	Enable To Pin External pulse output is enabled/disabled (TO00). 0: External pulse output is disabled. Output of the ACTLV0 bit inactive level to the TO00 pin is fixed. The TO00 pin level is not changed even if a match signal from the compare register (CCC00, CCC01) is generated. 1: External pulse output is enabled. A compare register match causes TO00 output to change. However, if capture mode is set, TO00 output does not change. An ACTLV0 bit inactive level is output from the time when timer output is enabled until a match signal is first generated. Caution If either CCC00 or CCC01 is specified as a capture register, the ENT01 bit must be set to 0.

★

Bit Position	Bit Name	Function
5	ACTLV0	<p>Active Level</p> <p>Specifies the active level for external pulse output (TO00).</p> <p>0: Active level is low level 1: Active level is high level</p> <p>Caution The initial value of the ACTLV0 bit is 1.</p>
4	ETIn (n = 0, 1)	<p>External Input</p> <p>Specifies a switch between the external and internal count clock.</p> <p>0: Specifies the input clock (internal). The count clock can be selected according to the CSn2 to SCn0 bits of TMCCn0 (n = 0, 1). 1: Specifies the external clock (TI0n0). The valid edge can be selected according to the TESn1 and TESn0 bit specifications of SESCn (n = 0, 1).</p>
3	CCLRn (n = 0, 1)	<p>Compare Clear Enable</p> <p>Sets whether the clearing of TMCn is enabled or disabled during a compare operation (n = 0, 1).</p> <p>0: Clearing is disabled 1: Clearing is enabled (if CCCn0 and TMCn match during a compare operation, TMCn is cleared).</p>
1	CMSn1 (n = 0, 1)	<p>Capture/Compare Mode Select</p> <p>Selects the operation mode of the capture/compare register (CCcn1) (n = 0, 1).</p> <p>0: The register operates as a capture register 1: The register operates as a compare register</p>
0	CMSn0 (n = 0, 1)	<p>Capture/Compare Mode Select</p> <p>Selects the operation mode of the capture/compare register (CCcn0) (n = 0, 1).</p> <p>0: The register operates as a capture register 1: The register operates as a compare register</p>

Remarks 1. A reset takes precedence for the flip-flop of the TO00 output.

- 2.** When the A/D converter is set to timer trigger mode, the match interrupt of the compare registers becomes a start trigger for A/D conversion, and the conversion operation begins. At this time, the compare register match interrupt also functions as a compare register match interrupt for the CPU. To prevent the generation of a compare register match interrupt for the CPU, disable an interrupt by the interrupt mask bits (P00MK0, P00MK1, P01MK0, and P01MK1) of the interrupt control registers (P00IC0, P00IC1, P01IC0, and P01IC1).

(3) Valid edge selection registers C0, C1 (SESC0, SESC1)

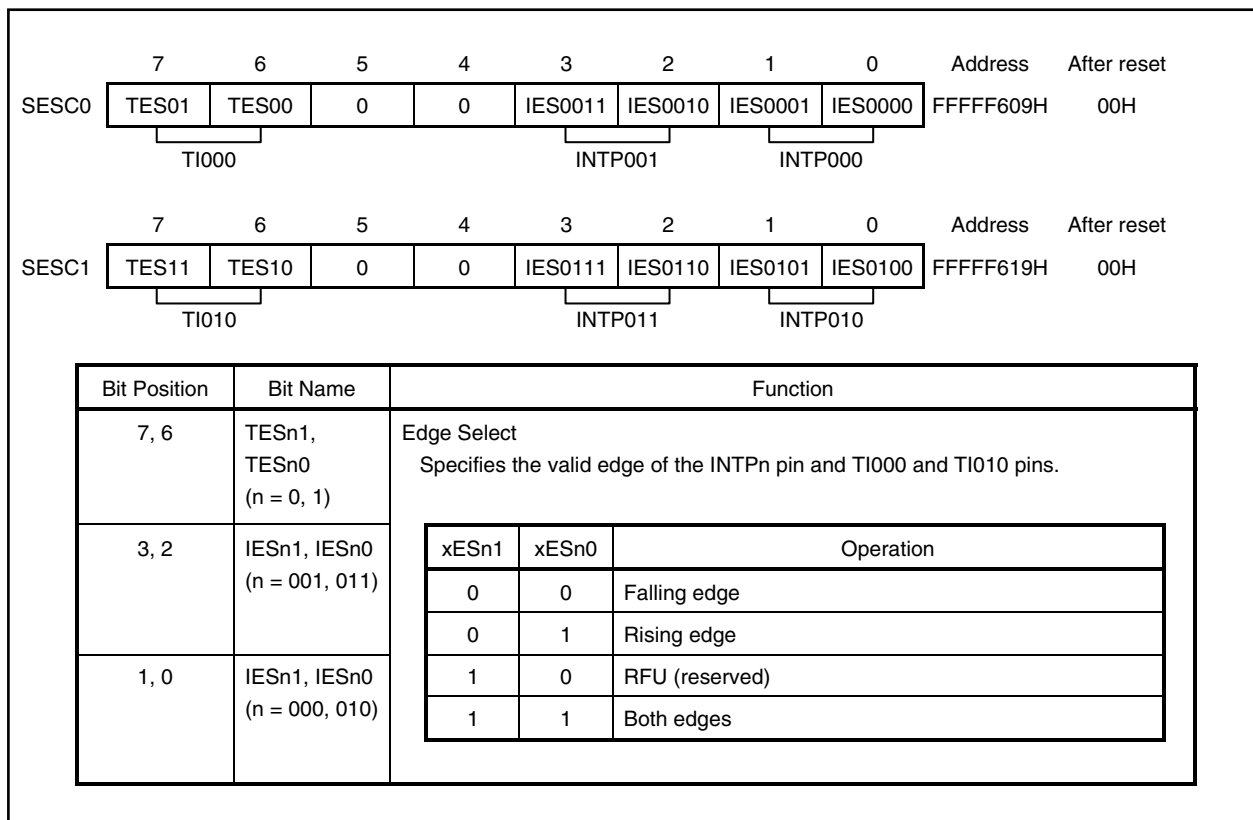
These registers specify the valid edge of an external interrupt request (INTP000, INTP001, INTP010, INTP011) from an external pin.

The rising edge, the falling edge, or both rising and falling edges can be specified as the valid edge independently for each pin.

Each of these registers can be read or written in 8-bit units.

Be sure to set bits 5 and 4 to 0. If they are set to 1, the operation is not guaranteed.

Caution The various bits of the SESCn register must not be changed during timer operation. If they are to be changed, they must be changed after setting the TMCCEn bit of the TMCCn0 register to 0. If the SESCn register is overwritten during timer operation, operation cannot be guaranteed.



10.1.6 Timer C operation

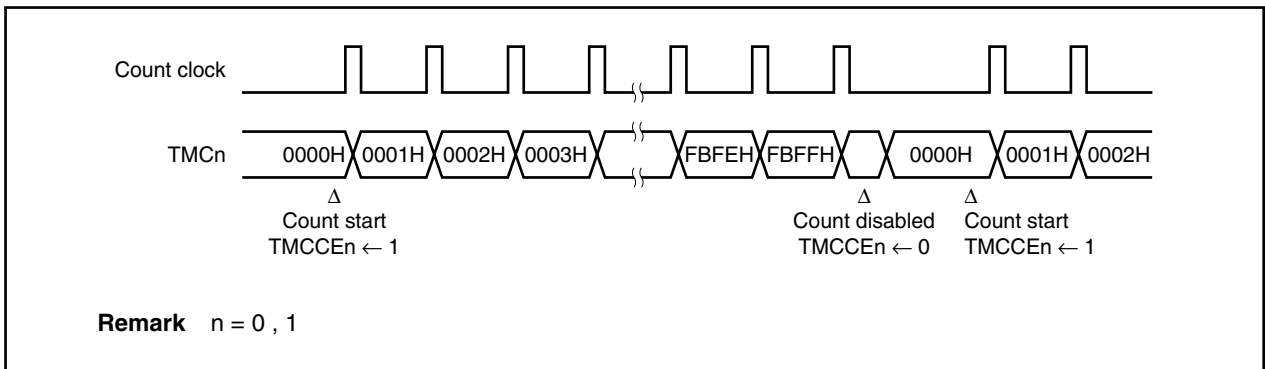
(1) Count operation

Timer C can function as a 16-bit free-running timer or as an external signal event counter. The setting for the type of operation is specified by timer mode control registers Cn0 and Cn1 (TMCCn0 and TMCCn1) (n = 0, 1).

When it operates as a free-running timer, if the CCC00 or CCC01 register and the TMC0 count value match, an interrupt signal is generated and the timer output signal (TO00) can be set or reset. Also, a capture operation that holds the TMCn count value in the CCCn0 or CCCn1 register is performed, synchronized with the valid edge that was detected from the external interrupt request input pin as an external trigger. The capture value is held until the next capture trigger is generated.

Caution When using INTP0n0/TI0n0 pin as an external clock input pin (TI0n0), be sure to disable the INTP0n0 interrupt or set the CCCn0 register to compare mode (n = 0, 1).

Figure 10-1. Basic Operation of Timer C



(2) Overflow

When the TMCn register has counted the count clock from FFFFH to 0000H, the OVFn bit of the TMCCn0 register is set (1), and an overflow interrupt (INTOV0n) is generated at the same time. However, if the CCCn0 register is set to compare mode (CMSn0 bit = 1) and to the value FFFFH when match clearing is enabled (CCLRn bit = 1), then the TMCn register is considered to be cleared and the OVFn bit is not set (1) when the TMCn register changes from FFFFH to 0000H. Also, the overflow interrupt (INTOV0n) is not generated.

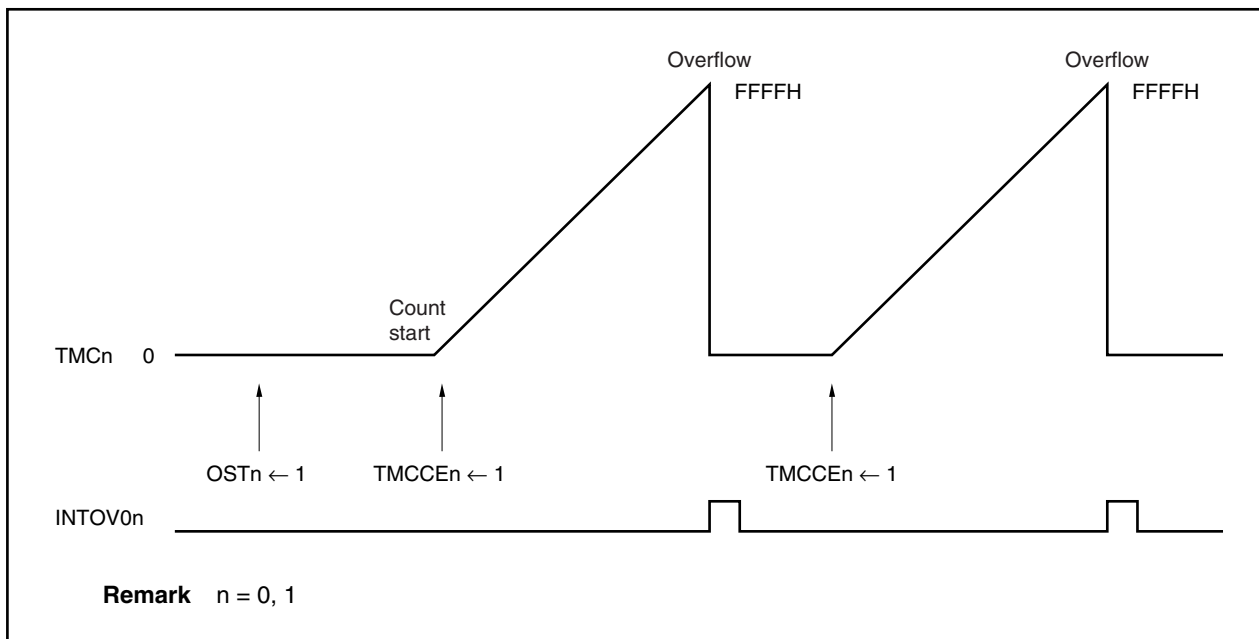
When the TMCn register is changed from FFFFH to 0000H because the TMCCEn bit changes from 1 to 0, the TMCn register is considered to be cleared, but the OVFn bit is not set (1) and no INTOV0n interrupt is generated.

Also, timer operation can be stopped after an overflow by setting the OSTn bit of the TMCCn1 register to 1. When the timer is stopped due to an overflow, the count operation is not restarted until the TMCCEn bit of the TMCCn0 register is set (1).

Operation is not affected even if the TMCCEn bit is set (1) during a count operation.

Remark n = 0, 1

Figure 10-2. Operation After Overflow (When OSTn = 1)



(3) Capture operation

The TMCn register has two capture/compare registers. These are the CCCn0 register and the CCCn1 register. A capture operation or a compare operation is performed according to the settings of both the CMSn1 and CMSn0 bits of the TMCCn1 register. If the CMSn1 and CMSn0 bits of the TMCCn1 register are set to 0, the register operates as a capture register.

A capture operation that captures and holds the TMCn count value asynchronously relative to the count clock is performed synchronized with an external trigger. The valid edge that is detected from an external interrupt request input pin (INTP0n0 or INTP0n1) is used as an external trigger (capture trigger). The TMCn count value during counting is captured and held in the capture register, synchronized with that capture trigger signal. The capture register value is held until the next capture trigger is generated.

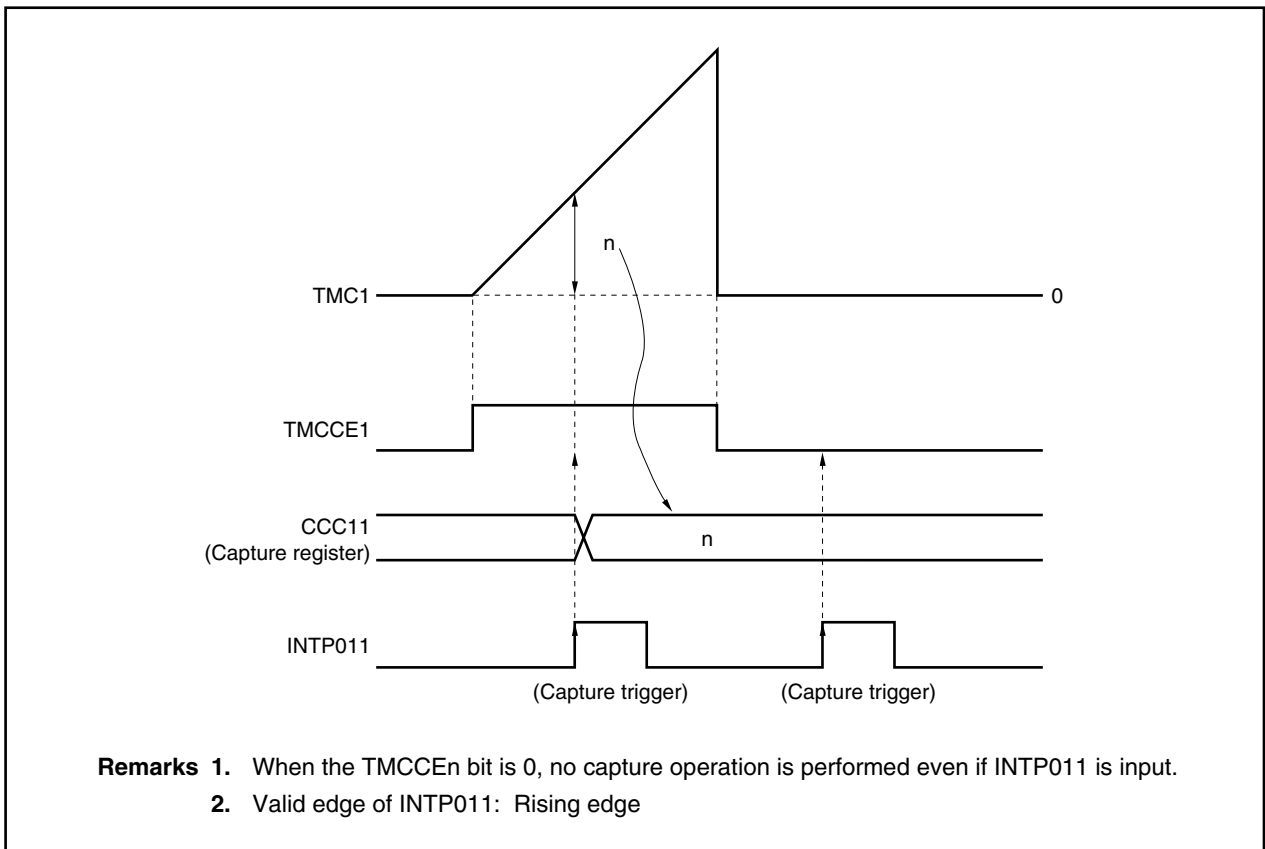
Also, an interrupt request (INTM0n0 or INTM0n1) is generated by INTP0n0 or INTP0n1 signal input.

The valid edge of the capture trigger is set by valid edge selection register Cn (SESCn).

If both the rising and falling edges are set as capture triggers, the input pulse width from an external source can be measured. Also, if only one of the edges is set as the capture trigger, the input pulse cycle can be measured.

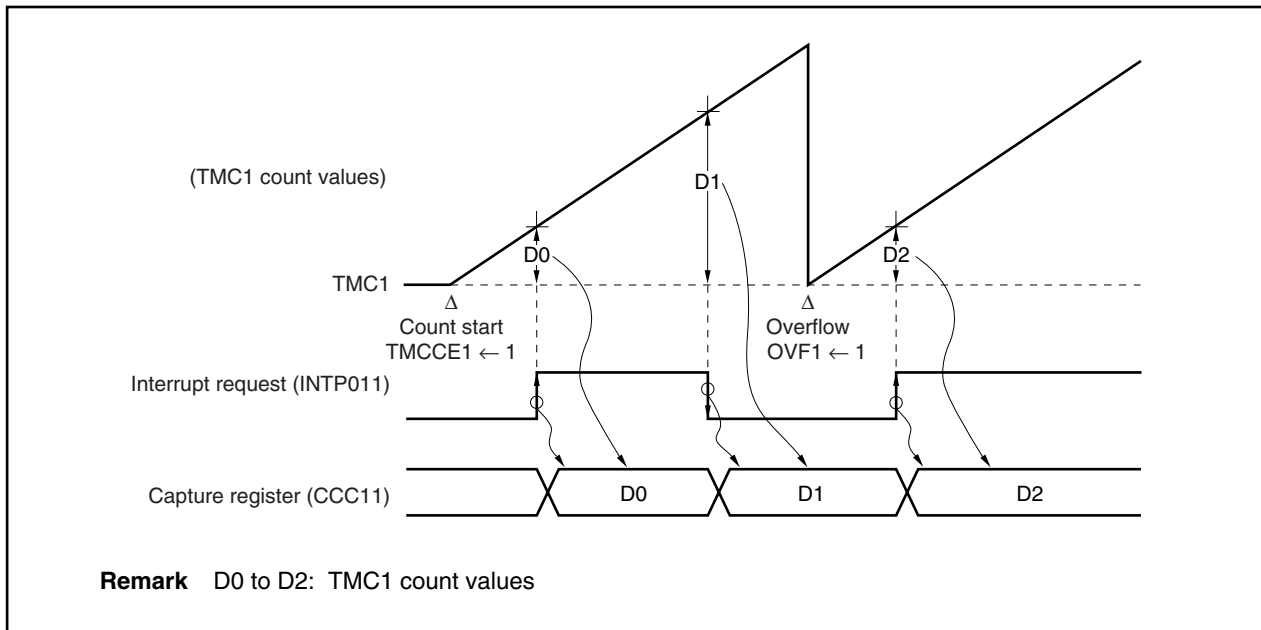
Remark $n = 0, 1$

Figure 10-3. Capture Operation Example



- Remarks**
1. When the TMCCEn bit is 0, no capture operation is performed even if INTP011 is input.
 2. Valid edge of INTP011: Rising edge

Figure 10-4. TMC1 Capture Operation Example (When Both Edges Are Specified)



(4) Compare operation

The TMCn register has two capture/compare registers. These are the CCCn0 register and the CCCn1 register. A capture operation or a compare operation is performed according to the settings of both the CMSn1 and CMSn0 bits of the TMCCn1 register. If 1 is set in the CMSn1 and CMSn0 bits of the TMCCn1 register, the register operates as a compare register.

A compare operation that compares the value that was set in the compare register and the TMCn count value is performed.

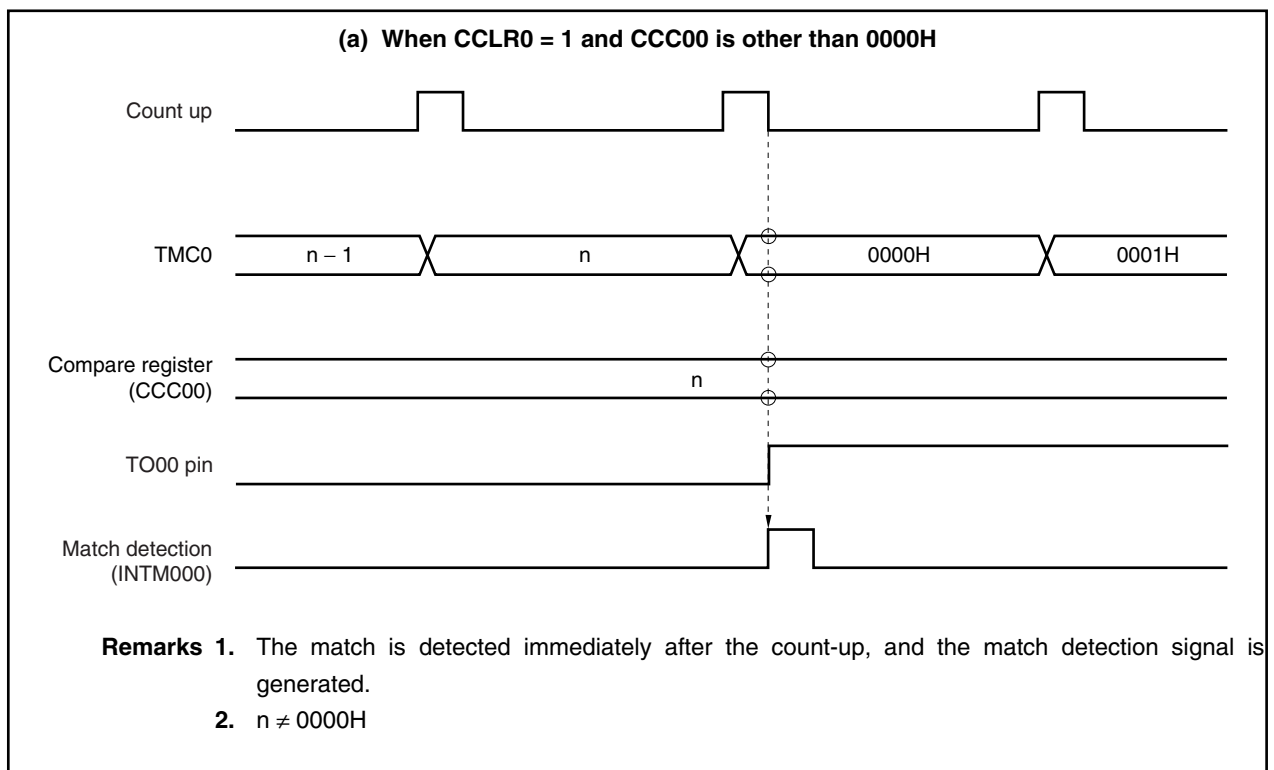
If the TMCn count value matches the value of the compare register, which had been set in advance, a match signal is sent to the output controller. The match signal causes the timer output pin (TO00) to change (timer C0 only) and an interrupt request signal (INTM0n0 or INTM0n1) to be generated at the same time.

If the CCCn0 or CCCn1 register is set to 0000H, the 0000H after the TMCn register counts up from FFFFH to 0000H is judged as a match. In this case, the value of the TMCn register is cleared (set to 0) at the next counting. However, a judgment as to whether the TMCn register value matches 0000H is not made. The 0000H when the TMCn register begins counting is not judged as a match, either.

If match clearing is enabled (CCLRn bit = 1) for the CCCn0 register, the TMCn register is cleared when a match with the TMCn register occurs during a compare operation.

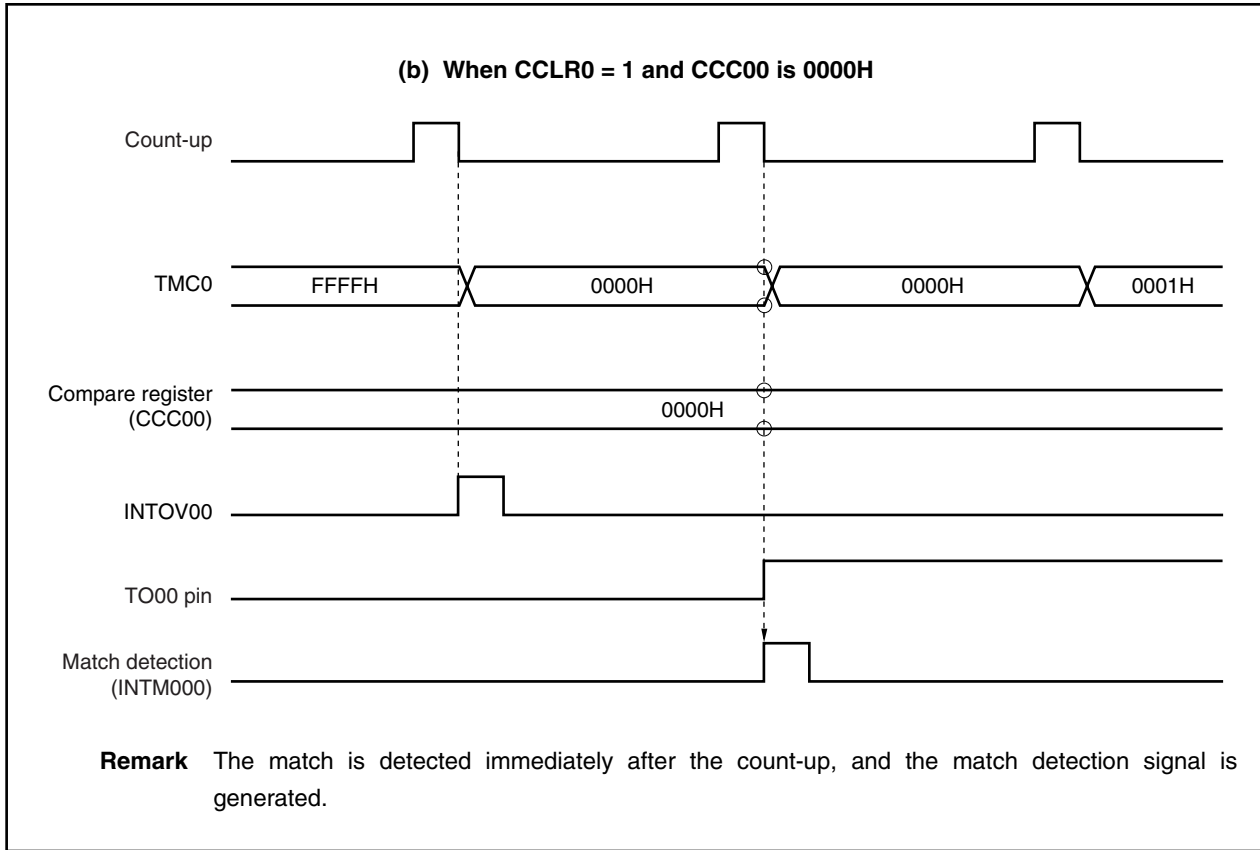
Remark n = 0, 1

Figure 10-5. Compare Operation Example (1/2)



★

Figure 10-5. Compare Operation Example (2/2)



(5) External pulse output

Timer C has one timer output pin (TO00).

An external pulse output (TO00) is generated when a match of the two compare registers (CCC00 and CCC01) and the TMC0 register is detected.

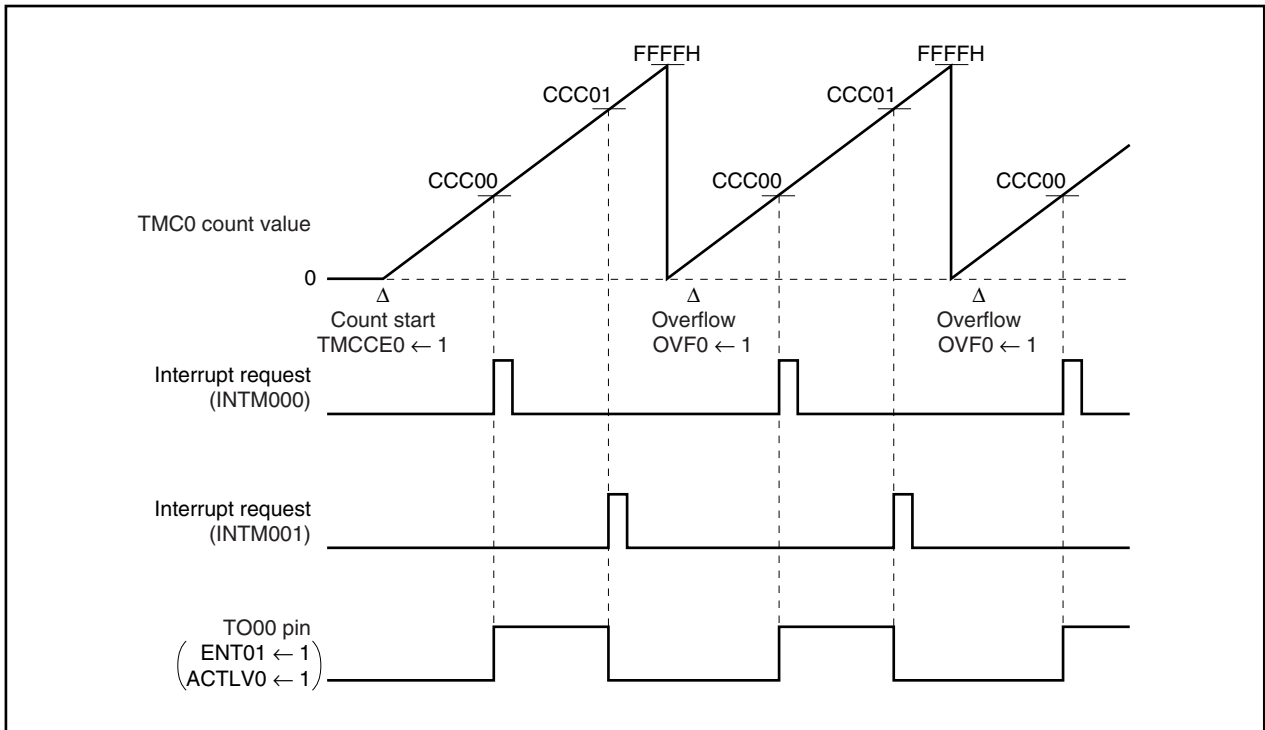
If a match is detected when the TMC0 count value and the CCC00 value are compared, the output level of the TO00 pin is set. Also, if a match is detected when the TMC0 count value and the CCC01 value are compared, the output level of the TO00 pin is reset.

The output level of the TO00 pin can be specified by the TMCC01 register.

Table 10-2. TO00 Output Control

ENT01	ACTLV0	TO00 Output	
		External Pulse Output	Output Level
0	0	Disabled	High level
0	1	Disabled	Low level
1	0	Enabled	When the CCC00 register is matched: Low level When the CCC01 register is matched: High level
1	1	Enabled	When the CCC00 register is matched: High level When the CCC01 register is matched: Low level

Figure 10-6. TMC0 Compare Operation Example (Set/Reset Output Mode)



10.1.7 Application examples (timer C)

(1) Interval timer

By setting the TMCCn0 and TMCCn1 registers as shown in Figure 10-7, timer C operates as an interval timer that repeatedly generates interrupt requests with the value that was set in advance in the CCCn0 register as the interval.

When the counter value of the TMCn register matches the setting value of the CCCn0 register, the TMCn register is cleared (0000H) and an interrupt request signal (INTM0n0) is generated at the same time that the count operation resumes.

Remark n = 0, 1

Figure 10-7. Contents of Register Settings When Timer C Is Used as Interval Timer

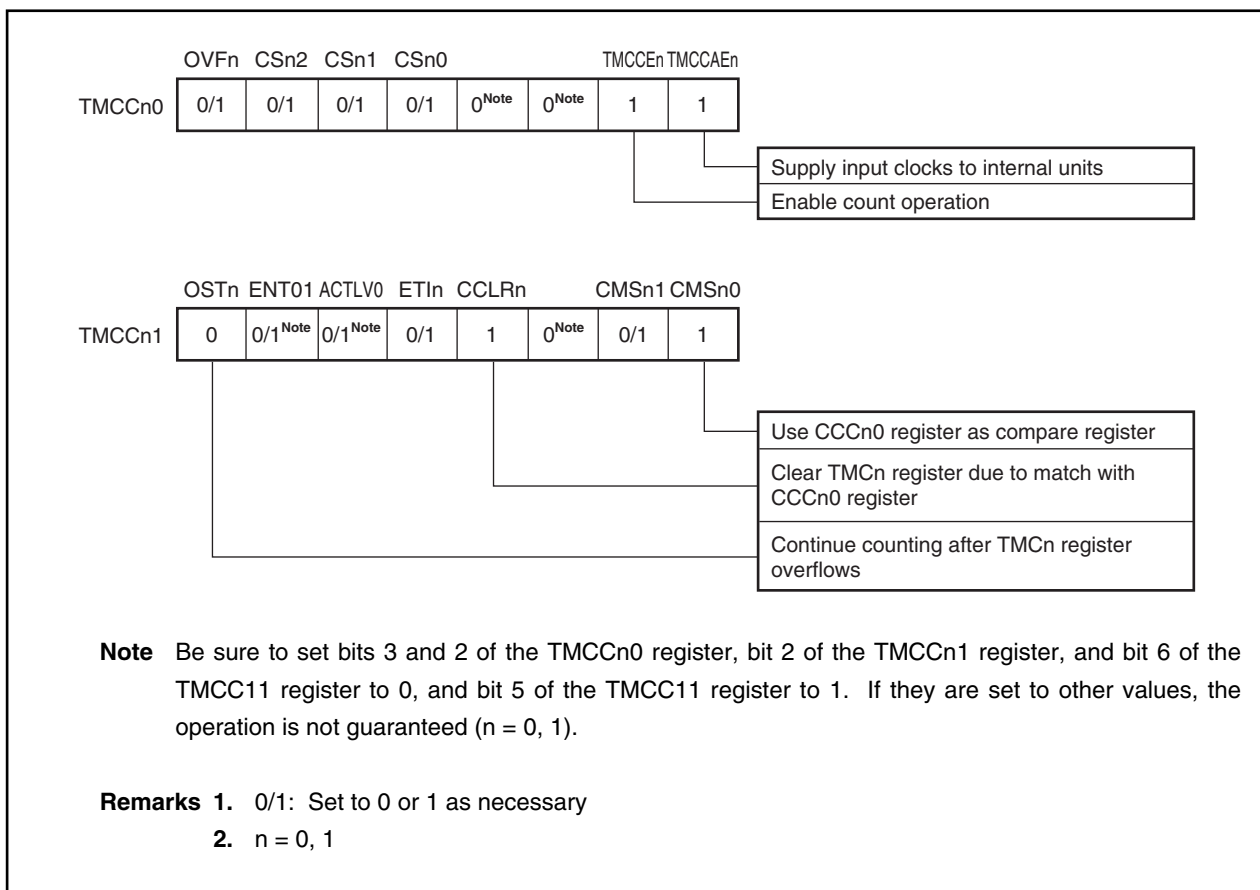
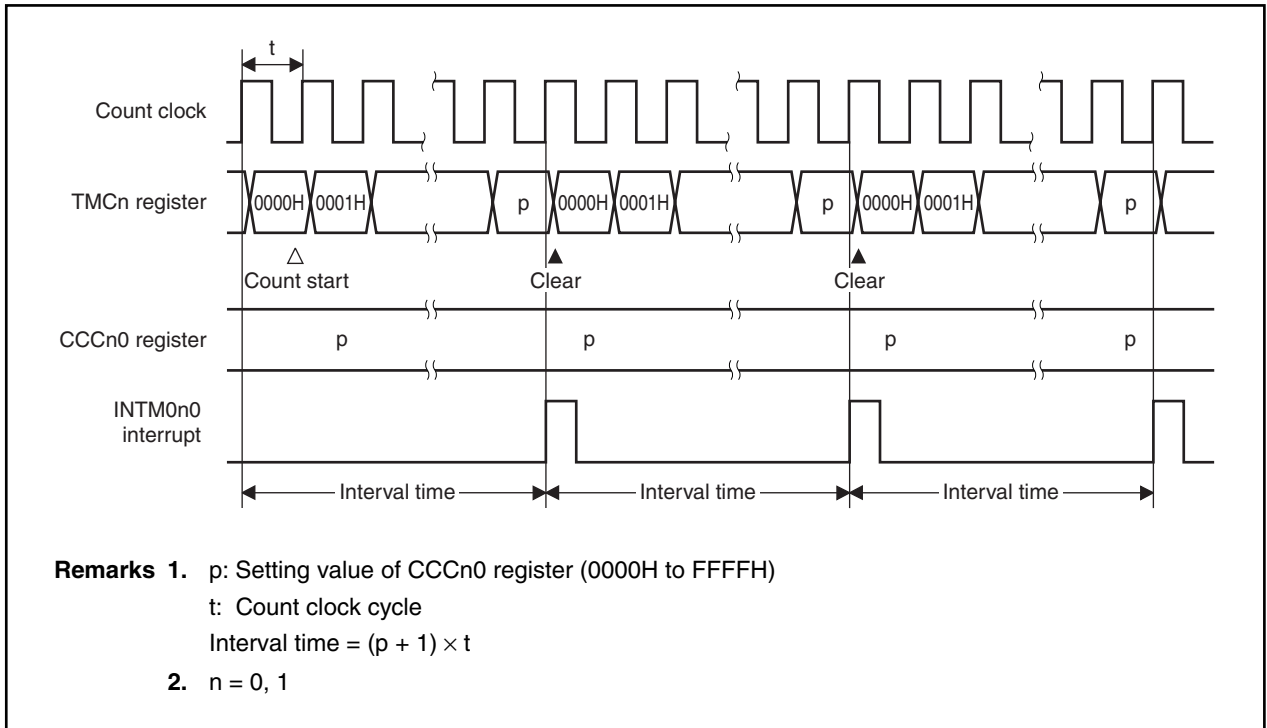


Figure 10-8. Interval Timer Operation Timing Example



(2) PWM output

By setting the TMCC00 and TMCC01 registers as shown in Figure 10-9, timer C can output a PWM signal whose frequency is determined according to the setting of the CS02 to CS00 bits of the TMCC00 register, with the values that were preset in the CCC00 and CCC01 registers as the interval.

When the counter value of the TMC0 register matches the setting value of the CCC00 register, the TO00 output becomes active. Then, when the counter value of the TMC0 register matches the setting value of the CCC01 register, the TO00 output becomes inactive. The TMC0 register continues counting. When it overflows, its count value is cleared to 0000H, and the register continues counting. In this way, a PWM signal whose frequency is determined according to the setting of the CS02 to CS00 bits of the TMCC00 register can be output. When the setting value of the CCC00 register and the setting value of the CCC01 register are the same, the TO00 output remains inactive and does not change.

The active level of TO00 output can be set by the ACTLV0 bit of the TMCC01 register.

Figure 10-9. Contents of Register Settings When Timer C Is Used for PWM Output

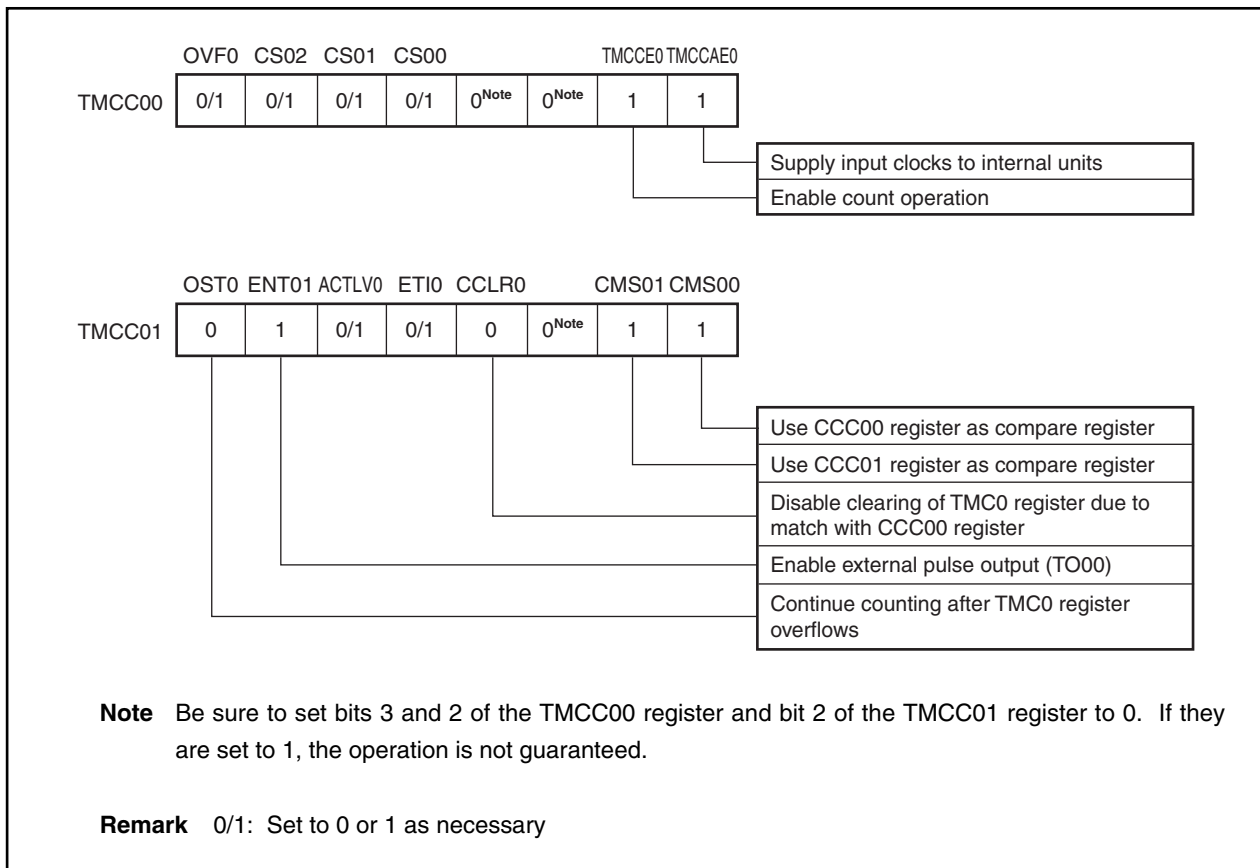
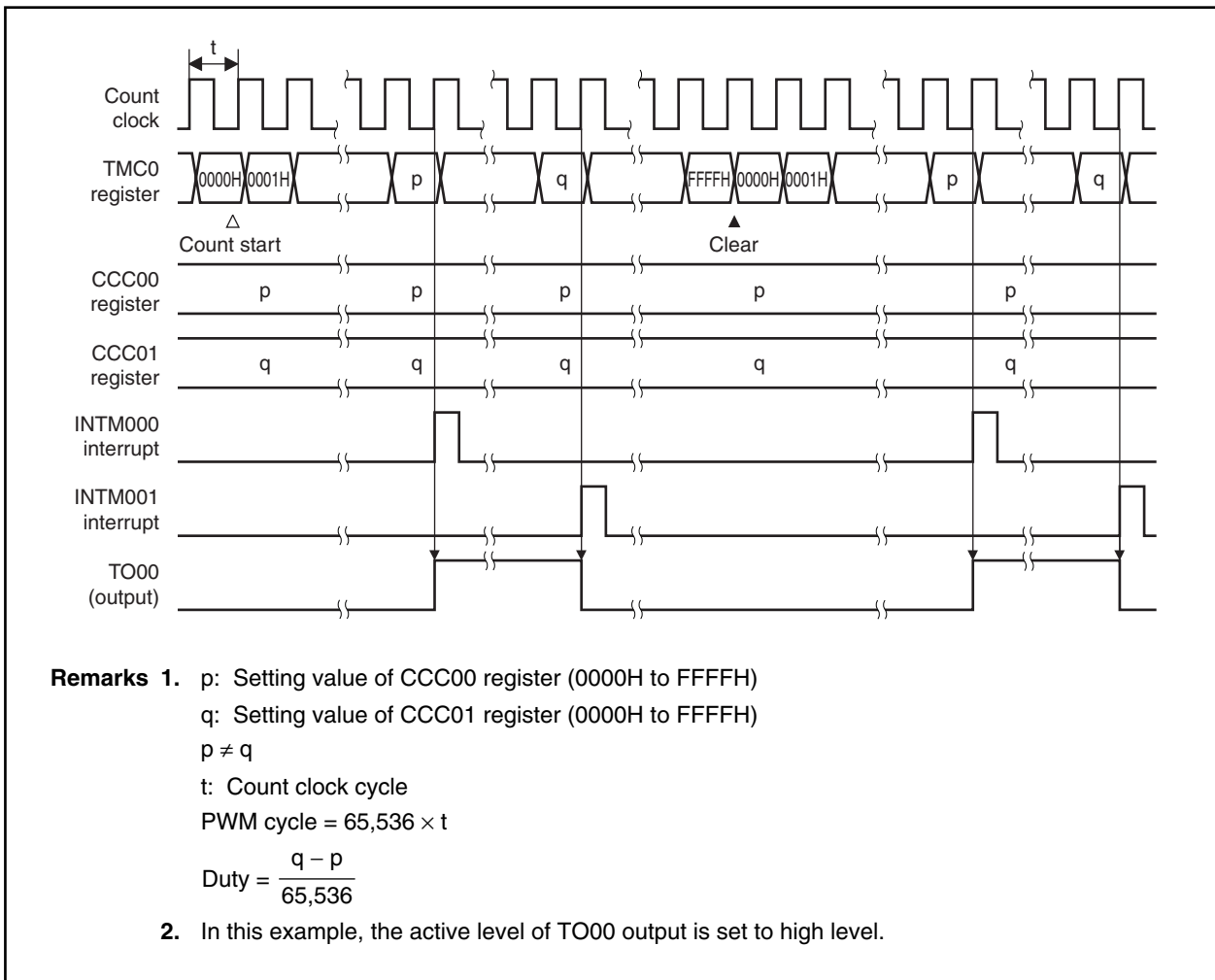


Figure 10-10. PWM Output Timing Example



(3) Cycle measurement

By setting the TMCCn0 and TMCCn1 registers as shown in Figure 10-11, timer C can measure the cycle of signals input to the INTP0n0 pin or INTP0n1 pin.

The valid edge of the INTP0n0 pin is selected according to the IES0n01 and IES0n00 bits of the SESCn register, and the valid edge of the INTP0n1 pin is selected according to the IES0n11 and IES0n10 bits of the SESCn register. Either the rising edge, the falling edge, or both edges can be selected as the valid edges of both pins.

If the CCCn0 register is set to a capture register, the valid edge input of the INTP0n0 pin is set as the trigger for capturing the TMCn register value in the CCCn0 register. When this value is captured, an INTM0n0 interrupt is generated.

Similarly, if the CCCn1 register is set to a capture register, the valid edge input of the INTP0n1 pin is set as the trigger for capturing the TMCn register value in the CCCn1 register. When this value is captured, an INTM0n1 interrupt is generated.

The cycle of signals input to the INTP0n0 pin is calculated by obtaining the difference between the TMCn register's count value (D_x) that was captured in the CCCn0 register according to the x -th valid edge input of the INTP0n0 pin and the TMCn register's count value ($D_{(x+1)}$) that was captured in the CCCn0 register according to the $(x+1)$ -th valid edge input of the INTP0n0 pin and multiplying the value of this difference by the cycle of the clock control signal.

The cycle of signals input to the INTP0n1 pin is calculated by obtaining the difference between the TMCn register's count value (D_x) that was captured in the CCCn1 register according to the x -th valid edge input of the INTP0n1 pin and the TMCn register's count value ($D_{(x+1)}$) that was captured in the CCCn1 register according to the $(x+1)$ -th valid edge input of the INTP0n1 pin and multiplying the value of this difference by the cycle of the clock control signal.

Remark $n = 0, 1$

Figure 10-11. Contents of Register Settings When Timer C Is Used for Cycle Measurement

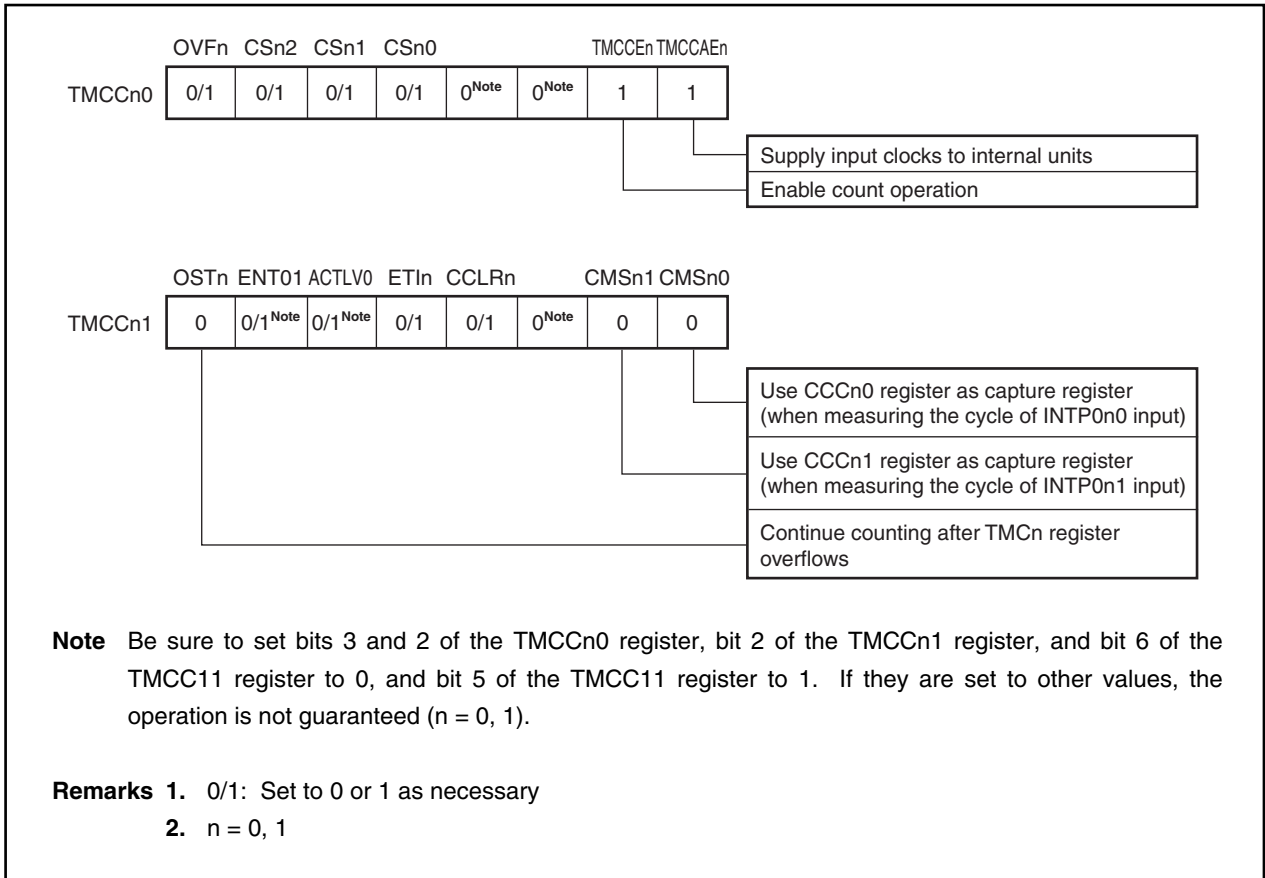
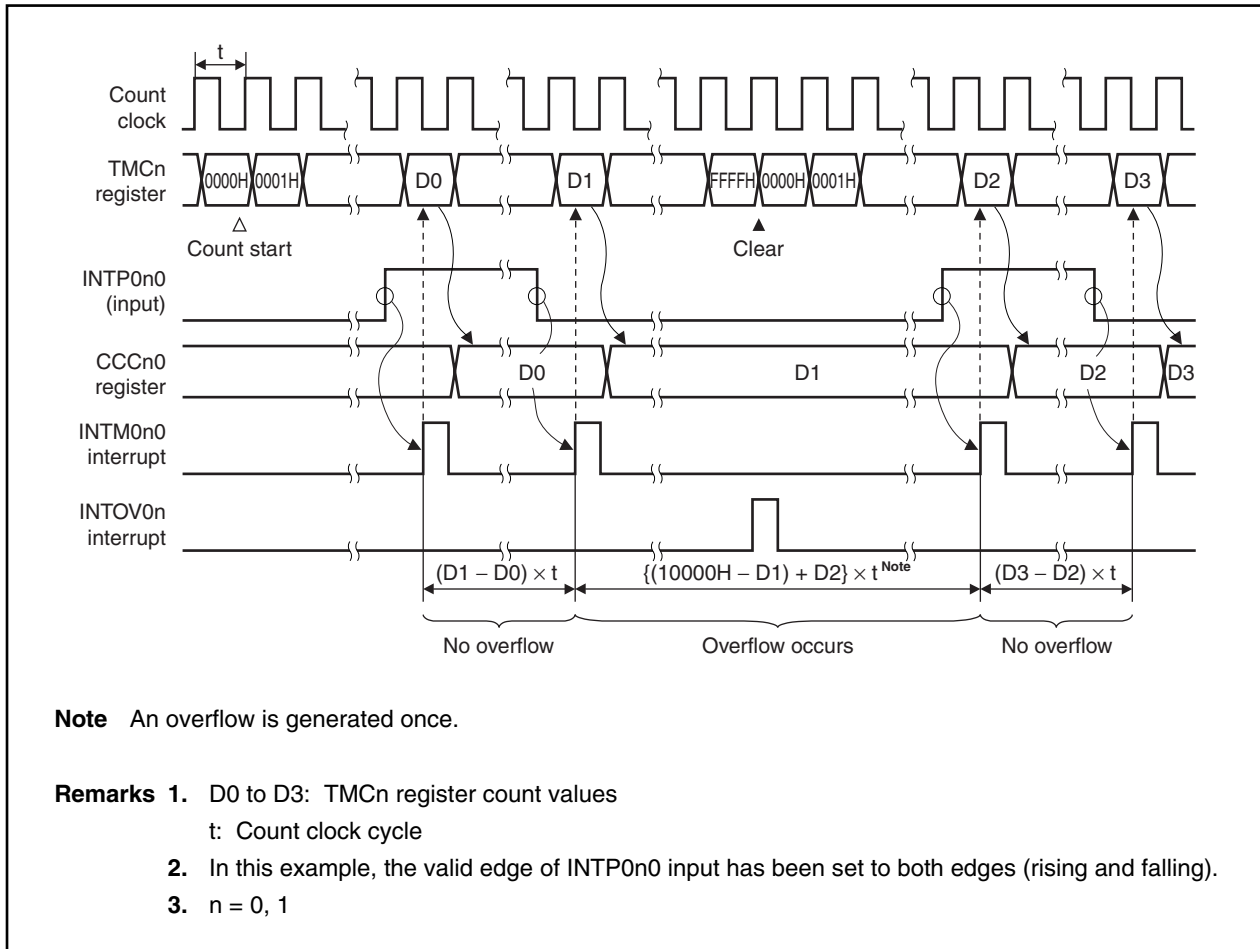


Figure 10-12. Cycle Measurement Operation Timing Example



10.1.8 Precautions (timer C)

Various precautions concerning timer C are shown below.

- (1) If a conflict occurs between the reading of the CCCn0 register and a capture operation when the CCCn0 register is used in capture mode, an external trigger (INTP0n0) valid edge is detected and an external interrupt request signal (INTM0n0) is generated however, timer value is not stored in the CCCn0 register.
- (2) If a conflict occurs between the reading of the CCCn1 register and a capture operation when the CCCn1 register is used in capture mode, an external trigger (INTP0n1) valid edge is detected and an external interrupt request signal (INTM0n1) is generated however, the timer value is not stored in the CCCn0 register.
- (3) The following registers must not be rewritten during operation (TMCCEn = 1).
 - CSn2 to CSn0 bits of TMCCn0 register
 - TMCCn1 register
 - SESCn register
- (4) The TMCCAEn bit of the TMCCn0 register is a TMCn reset signal. To use TMCn, first set (1) the TMCCAEn bit.
- (5) The analog noise elimination time + two cycles of the input clock are required to detect a valid edge of the external interrupt request signal (INTP0n0 or INTP0n1) or the external clock input (TI0n0). Therefore, edge detection will not be performed normally for changes that are less than the analog noise elimination time + two cycles of the input clock. For the analog noise elimination, refer to **7.3.8 Noise elimination**.
- (6) The operation of an external interrupt request signal (INTM0n0 or INTM0n1) is automatically determined according to the operating state of the capture/compare register. When the capture/compare register is used for a capture operation, the external interrupt request signal is used for valid edge detection. When the capture/compare register is used for a compare operation, the external interrupt request signal is used for a match interrupt indicating a match with the TMCn register.
- (7) If the ENT01 and ACTLV0 bits are changed at the same time, a glitch (spike shaped noise) may be generated in the TO00 pin output. Either create a circuit configuration that will not malfunction even if a glitch is generated or make sure that the ENT01 and ACTLV0 bits do not change at the same time.

Remark n = 0, 1

10.2 Timer D

10.2.1 Features (timer D)

Timer D functions as a 16-bit interval timer.

10.2.2 Function overview (timer D)

- 16-bit interval timer
- Compare registers: 4
- Interrupt request sources: 4 sources
- Count clock selected from divisions of internal system clock

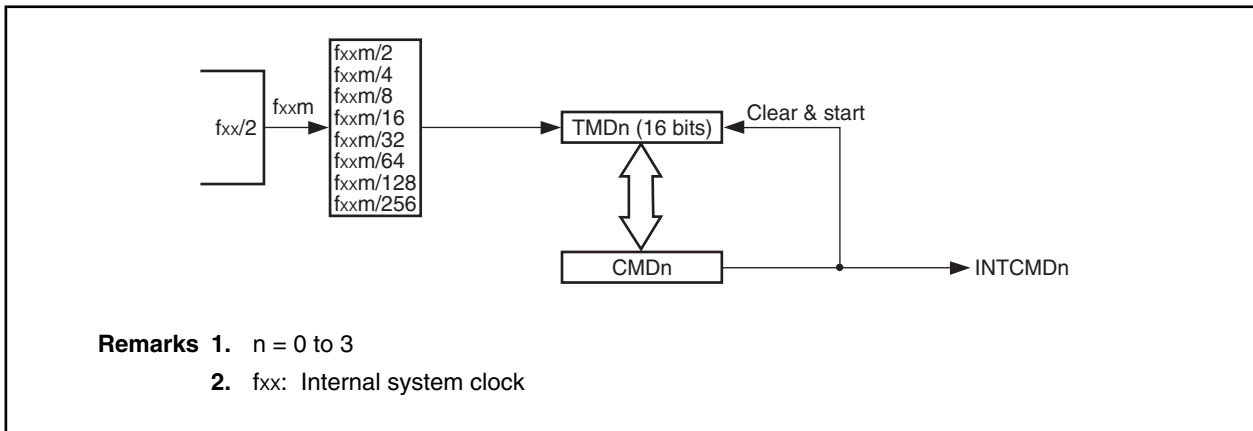
10.2.3 Basic configuration of timer D

Table 10-3. Timer D Configuration

Timer	Count Clock	Register	Read/Write	Generated Interrupt Signal	Capture Trigger	Timer Output S/R	Other Functions
Timer D	f _{xx} /4, f _{xx} /8, f _{xx} /16, f _{xx} /32, f _{xx} /64, f _{xx} /128, f _{xx} /256, f _{xx} /512	TMD0	Read	–	–	–	–
		CMD0	Read/write	INTCMD0	–	–	–
		TMD1	Read	–	–	–	–
		CMD1	Read/write	INTCMD1	–	–	–
		TMD2	Read	–	–	–	–
		CMD2	Read/write	INTCMD2	–	–	–
		TMD3	Read	–	–	–	–
		CMD3	Read/write	INTCMD3	–	–	–

Remark f_{xx}: Internal system clock
S/R: Set/reset

(1) Timer D (16-bit timer/counter)



10.2.4 Timer D

(1) Timers D0 to D3 (TMD0 to TMD3)

TMDn is a 16-bit timer. It is mainly used as an interval timer for software (n = 0 to 3).

Starting and stopping TMDn is controlled by the TMDCEn bit of the timer mode control register Dn (TMCDn) (n = 0 to 3).

A division by the prescaler can be selected for the count clock from among fxx/4, fxx/8, fxx/16, fxx/32, fxx/64, fxx/128, fxx/256, and fxx/512 by the CSn0 to CSn2 bits of the TMCDn register (fxx: Internal system clock).

TMDn is read-only in 16-bit units.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
TMD0																	FFFFF540H	0000H
TMD1																	FFFFF550H	0000H
TMD2																	FFFFF560H	0000H
TMD3																	FFFFF570H	0000H

The conditions for which the TMDn register becomes 0000H are shown below (n = 0 to 3).

- Reset input
- TMDCAEn bit = 0
- TMDCEn bit = 0
- Match of TMDn register and CMDn register
- Overflow

- Cautions 1. If the TMDCAEn bit of the TMCDn register is cleared (0), a reset is performed asynchronously.**
- 2. If the TMDCEn bit of the TMCDn register is cleared (0), a reset is performed, synchronized with the internal clock. Similarly, a synchronized reset is performed after a match with the CMDn register and after an overflow.**
- 3. The count clock must not be changed during a timer operation. If it is to be overwritten, it should be overwritten after the TMDCEn bit is cleared (0).**
- 4. Up to 4 clocks are required after a value is set in the TMDCEn bit until the set value is transferred to internal units. When a count operation begins, the count cycle from 0000H to 0001H differs from subsequent cycles.**
- 5. After a compare match is generated, the timer is cleared at the next count clock. Therefore, if the division ratio is large, the timer value may not be zero even if the timer value is read immediately after a match interrupt is generated.**

(2) Compare registers D0 to D3 (CMD0 to CMD3)

CMDn and the TMDn register count value are compared, and an interrupt request signal (INTCMDn) is generated when a match occurs. TMDn is cleared, synchronized with this match. If the TMDCAEn bit of the TMCDn register is set to 0, a reset is performed asynchronously, and the registers are initialized (n = 0 to 3). The CMDn registers are configured with a master/slave configuration. When a write operation to a CMDn register is performed, data is first written to the master register and then the master register data is transferred to the slave register. In a compare operation, the slave register value is compared with the count value of the TMDn register. When a read operation to a CMDn register is performed, data in the master side is read out.

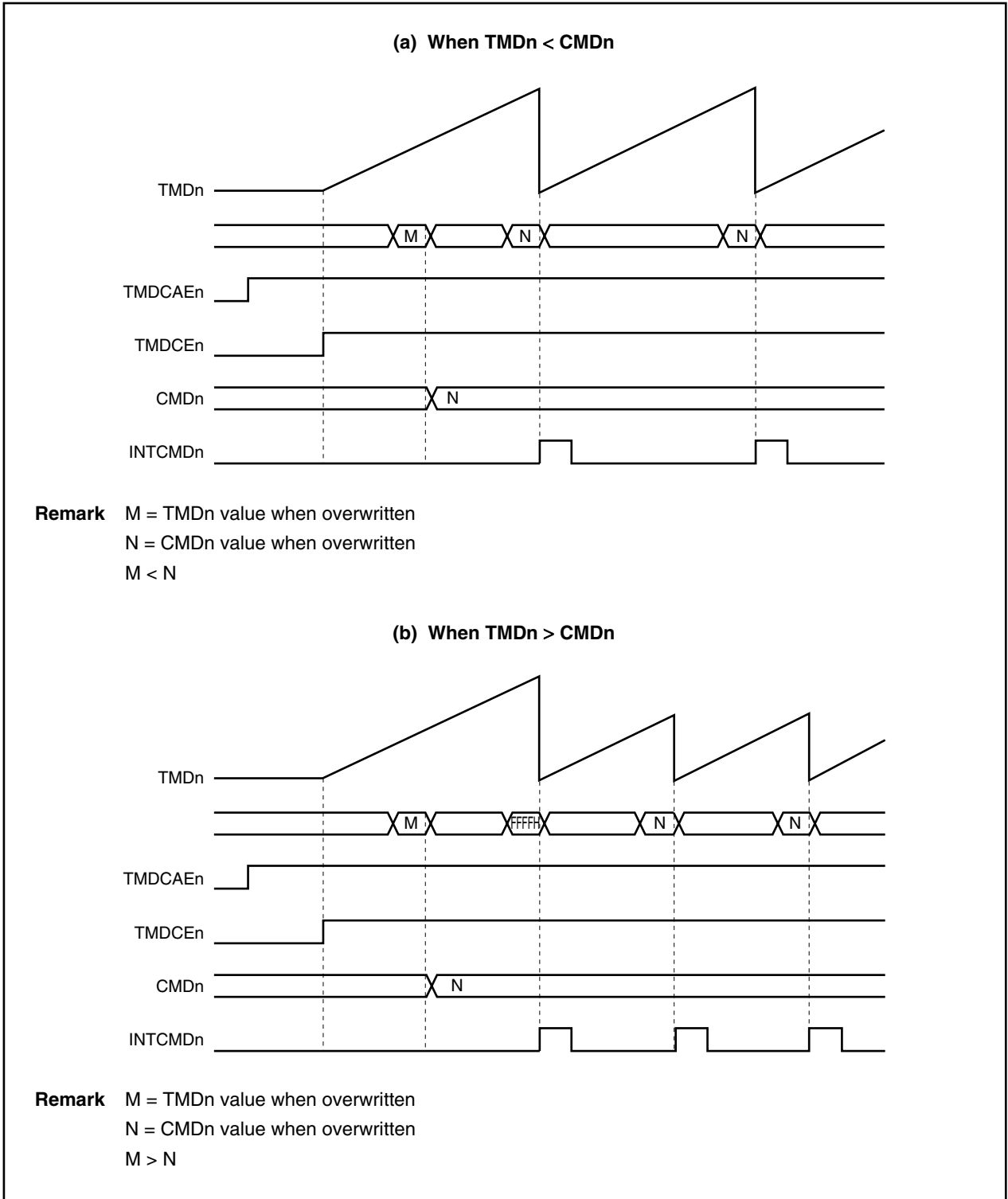
CMDn can be read or written in 16-bit units.

- Cautions**
1. A write operation to a CMDn register requires 4 clocks until the value that was set in the CMDn register is transferred to internal units. When writing continuously to the CMDn register, be sure to reserve a time interval of at least 4 clocks.
 2. The CMDn register can be overwritten only once in a single TMDn register cycle (from 0000H until an INTCMDn interrupt is generated due to a match of the TMDn register and CMDn register). If this cannot be secured by the application, make sure that the CMDn register is not overwritten during timer operation.
 3. Note that a match signal will be generated after an overflow if a value less than the counter value is written in the CMDn register during TMDn register operation (Figure 10-13).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CMD0																	FFFFF542H	0000H
CMD1																	FFFFF552H	0000H
CMD2																	FFFFF562H	0000H
CMD3																	FFFFF572H	0000H

Figure 10-13. Example of Timing During TMDn Operation

★



10.2.5 Timer D control registers

(1) Timer mode control registers D0 to D3 (TMCD0 to TMCD3)

The TMCDn registers control the operation of timer Dn (n = 0 to 3).

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

Caution The TMDCAEn and other bits cannot be set at the same time. The other bits and the registers of the other TMDn unit should always be set after the TMDCAEn bit has been set.

(1/2)

	7	6	5	4	3	2	<1>	<0>	Address	After reset
TMCD0	0	CS02	CS01	CS00	0	0	TMDCE0	TMDCAE0	FFFFF544H	00H
TMCD1	0	CS12	CS11	CS10	0	0	TMDCE1	TMDCAE1	FFFFF554H	00H
TMCD2	0	CS22	CS21	CS20	0	0	TMDCE2	TMDCAE2	FFFFF564H	00H
TMCD3	0	CS32	CS31	CS30	0	0	TMDCE3	TMDCAE3	FFFFF574H	00H

Bit Position	Bit Name	Function																																				
6 to 4	CSn2 to CSn0 (n = 0 to 3)	<p>Count Enable Select Selects the TMDn internal count clock cycle (n = 0 to 3).</p> <table border="1"> <thead> <tr> <th>CSn2</th> <th>CSn1</th> <th>CSn0</th> <th>Count cycle</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>f_{xx}/4</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>f_{xx}/8</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>f_{xx}/16</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>f_{xx}/32</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>f_{xx}/64</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>f_{xx}/128</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>f_{xx}/256</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>f_{xx}/512</td> </tr> </tbody> </table> <p>Caution The CSn2 to CSn0 bits must not be changed during timer operation. If they are to be changed, they must be changed after setting the TMDCEn bit to 0. If these bits are overwritten during timer operation, operation cannot be guaranteed.</p> <p>Remark f_{xx}: Internal system clock</p>	CSn2	CSn1	CSn0	Count cycle	0	0	0	f _{xx} /4	0	0	1	f _{xx} /8	0	1	0	f _{xx} /16	0	1	1	f _{xx} /32	1	0	0	f _{xx} /64	1	0	1	f _{xx} /128	1	1	0	f _{xx} /256	1	1	1	f _{xx} /512
CSn2	CSn1	CSn0	Count cycle																																			
0	0	0	f _{xx} /4																																			
0	0	1	f _{xx} /8																																			
0	1	0	f _{xx} /16																																			
0	1	1	f _{xx} /32																																			
1	0	0	f _{xx} /64																																			
1	0	1	f _{xx} /128																																			
1	1	0	f _{xx} /256																																			
1	1	1	f _{xx} /512																																			
1	TMDCEn (n = 0 to 3)	<p>Count Enable Controls the operation of TMDn (n = 0 to 3). 0: Count disabled (stops at 0000H and does not operate) 1: Counting operation is performed</p> <p>Caution TMDCEn bit is not cleared even if a match is detected by the compare operation. To the stop count operation, clear the TMDCEn bit.</p>																																				

Bit Position	Bit Name	Function
0	TMDCAEn (n = 0 to 3)	<p>Clock Action Enable</p> <p>Controls the internal count clock (n = 0 to 3).</p> <p>0: The entire TMDn unit is reset asynchronously. The supply of input clocks to the TMDn unit stops.</p> <p>1: Input clocks are supplied to the TMDn unit</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. When the TMDCAEn bit is set to 0, the TMDn unit can be asynchronously reset. 2. When TMDCAEn = 0, the TMDn unit is in a reset state. Therefore, to operate TMDn, the TMDCAEn bit must be set to 1. 3. If the TMDCAEn bit is cleared to 0, all the registers of the TMDn unit are initialized. If TMDCAEn is set to 1 again, be sure to subsequently set all the registers of the TMDn unit again.

★

10.2.6 Timer D operation

(1) Compare operation

TMDn can be used for a compare operation in which the value that was set in a compare register (CMDn) is compared with the TMDn count value ($n = 0$ to 3).

If a match is detected by the compare operation, an interrupt (INTCMDn) is generated. The generation of the interrupt causes TMDn to be cleared (0) at the next count timing. This function enables timer D to be used as an interval timer.

CMDn can also be set to 0. In this case, when an overflow occurs and TMDn becomes 0, a match is detected and INTCMDn is generated. Although the TMDn value is cleared (0) at the next count timing, INTCMDn is not generated by this match.

Figure 10-14. TMD0 Compare Operation Example (1/2)

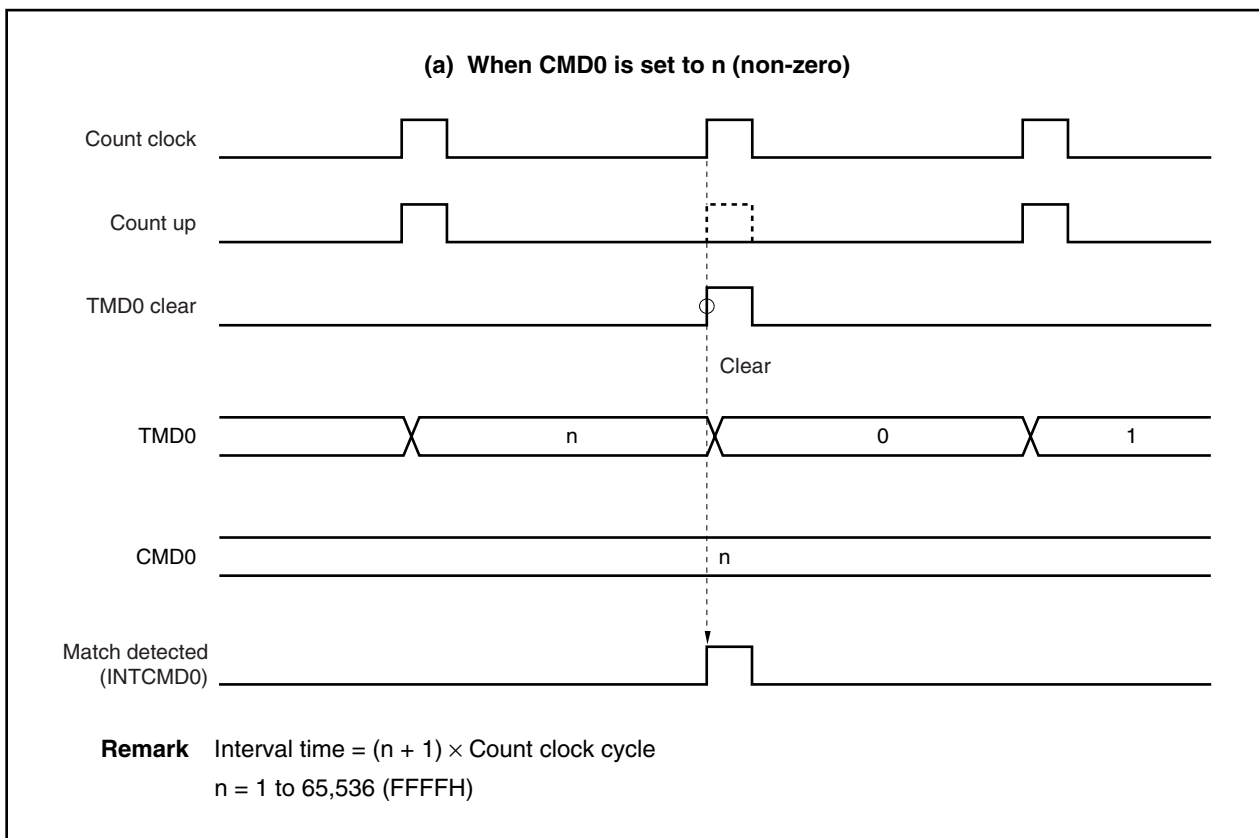
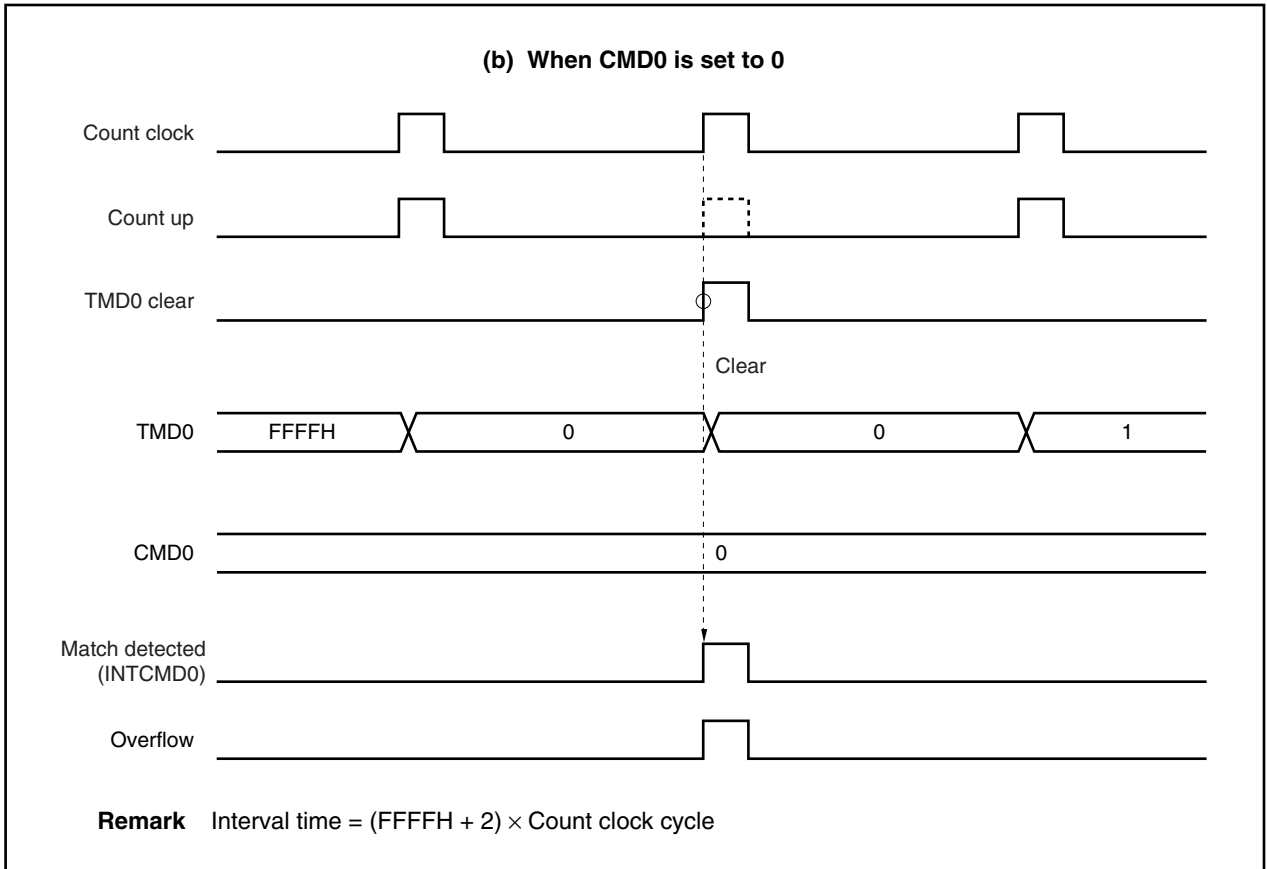


Figure 10-14. TMD0 Compare Operation Example (2/2)



10.2.7 Application examples (timer D)

(1) Interval timer

This section explains an example in which timer D is used as an interval timer with 16-bit precision. Interrupt requests (INTCMDn) are output at equal intervals (refer to **Figure 10-14 TMD0 Compare Operation Example**). The setup procedure is shown below (n = 0 to 3).

- <1> Set (1) the TMDCAEn bit.
- <2> Set each register.
 - Select the count clock using the CSn0 to CSn2 bits of the TMCDn registers.
 - Set the compare value in the CMDn register.
- <3> Start counting by setting (1) the TMDCEn bit.
- <4> If the TMDn register and CMDn register values match, an INTCMDn interrupt is generated.
- <5> INTCMDn interrupts are generated thereafter at equal intervals.

Remark n = 0 to 3

10.2.8 Precautions (timer D)

Various precautions concerning timer D are shown below.

- (1) To operate TMDn, first set (1) the TMDCAEn bit.
- (2) Up to 4 clocks are required after a value is set in the TMDCEn bit until the set value is transferred to internal units. When a count operation begins, the count cycle from 0000H to 0001H differs from subsequent cycles.
- (3) To initialize the TMDn register status and start counting again, clear (0) the TMDCEn bit and then set (1) the TMDCEn bit after an interval of 4 clocks has elapsed.
- (4) Up to 4 clocks are required until the value that was set in the CMDn register is transferred to internal units. When writing continuously to the CMDn register, be sure to secure a time interval of at least 4 clocks.
- (5) The CMDn register can be overwritten only once during a timer/counter operation (from 0000H until an INTCMDn interrupt is generated due to a match of the TMDn register and CMDn register). If this cannot be secured, make sure that the CMDn register is not overwritten during a timer/counter operation.
- (6) The count clock must not be changed during a timer operation. If it is to be overwritten, it should be overwritten after the TMDCEn bit is cleared (0). If the count clock is overwritten during a timer operation, operation cannot be guaranteed.
- (7) A match signal will be generated after an overflow if a value less than the counter value is written in the CMDn register during TMDn register operation.

Remark n = 0 to 3

CHAPTER 11 SERIAL INTERFACE FUNCTION

11.1 Features

The serial interface function provides two types of serial interfaces equipped with four transmit/receive channels of which two channels can be used simultaneously.

The following two interface formats are available.

- (1) Asynchronous serial interface (UART0, UART1): 2 channels
- (2) Clocked serial interface (CSI0, CSI1): 2 channels

UART0 and UART1, which use the method of transmitting/receiving one byte of serial data following a start bit, enable full-duplex communication to be performed.

CSI0 and CSI1 transfer data according to three types of signals (3-wire serial I/O). These signals are the serial clock ($\overline{SCK0}$, $\overline{SCK1}$), serial input (SI0, SI1), and serial output (SO0, SO1) signals.

11.1.1 Switching between UART and CSI modes

In the V850E/MA2, since UART0 and CSI0 pin and the UART1 and CSI1 pin are alternate function pins, they cannot be used at the same time. The PMC4 and PFC4 registers must be set in advance (refer to **13.3.4 Port 4**).

If the mode is switched during a transmit or receive operation in UARTn or CSIn, operation cannot be guaranteed.

11.2 Asynchronous Serial Interfaces 0, 1 (UART0, UART1)

11.2.1 Features

- Transfer rate: 300 bps to 1,250 kbps (using a dedicated baud rate generator and an internal system clock of 40 MHz)
- Full-duplex communications
 - On-chip receive buffer (RXBn)
 - On-chip transmit buffer (TXBn)
- Two-pin configuration
 - TXDn: Transmit data output pin
 - RXDn: Receive data input pin
- Reception error detection function
 - Parity error
 - Framing error
 - Overrun error
- Interrupt sources: 3 types
 - Reception error interrupt (INTSERn): Interrupt is generated according to the logical OR of the three types of reception errors
 - Reception completion interrupt (INTSRn): Interrupt is generated when receive data is transferred from the shift register to the receive buffer after serial transfer is completed during a reception enabled state
 - Transmission completion interrupt (INTSTn): Interrupt is generated when the serial transmission of transmit data (8 or 7 bits) from the shift register is completed
- The character length of transmit/receive data is specified according to the ASIM0 and ASIM1 registers
- Character length: 7 or 8 bits
- Parity functions: Odd, even, 0, or none
- Transmission stop bits: 1 or 2 bits
- On-chip dedicated baud rate generator

Remark n = 0, 1

11.2.2 Configuration

UARTn is controlled by the asynchronous serial interface mode register (ASIMn), asynchronous serial interface status register (ASISn), and asynchronous serial interface transmission status register (ASIFn) (n = 0, 1). Receive data is maintained in the receive buffer (RXBn), and transmit data is written to the transmit buffer (TXBn).

Figure 11-1 shows the configuration of the asynchronous serial interface.

(1) Asynchronous serial interface mode registers 0, 1 (ASIM0, ASIM1)

The ASIMn register is an 8-bit register for specifying the operation of the asynchronous serial interface.

(2) Asynchronous serial interface status registers 0, 1 (ASIS0, ASIS1)

The ASISn register consists of a set of flags that indicate the error contents when a reception error occurs. The various reception error flags are set (1) when a reception error occurs and are reset (0) when the ASISn register is read.

(3) Asynchronous serial interface transmission status registers 0, 1 (ASIF0, ASIF1)

The ASIFn register is an 8-bit register that indicates the status when a transmit operation is performed. This register consists of a transmit buffer data flag, which indicates the hold status of TXBn data, and the transmit shift register data flag, which indicates whether transmission is in progress.

(4) Reception control parity check

The receive operation is controlled according to the contents set in the ASIMn register. A check for parity errors is also performed during a receive operation, and if an error is detected, a value corresponding to the error contents is set in the ASISn register.

(5) Receive shift register

This is a shift register that converts the serial data that was input to the RXDn pin to parallel data. One byte of data is received, and if a stop bit is detected, the receive data is transferred to the receive buffer. This register cannot be directly manipulated.

(6) Receive buffer (RXBn)

RXBn is an 8-bit buffer register for holding receive data. When 7 characters are received, 0 is stored in the MSB.

During a reception enabled state, receive data is transferred from the receive shift register to the receive buffer, synchronized with the end of the shift-in processing of one frame.

Also, the reception completion interrupt request (INTSRn) is generated by the transfer of data to the receive buffer.

(7) Transmit shift register

This is a shift register that converts the parallel data that was transferred from the transmit buffer to serial data.

When one byte of data is transferred from the transmit buffer, the shift register data is output from the TXDn pin.

This register cannot be directly manipulated.

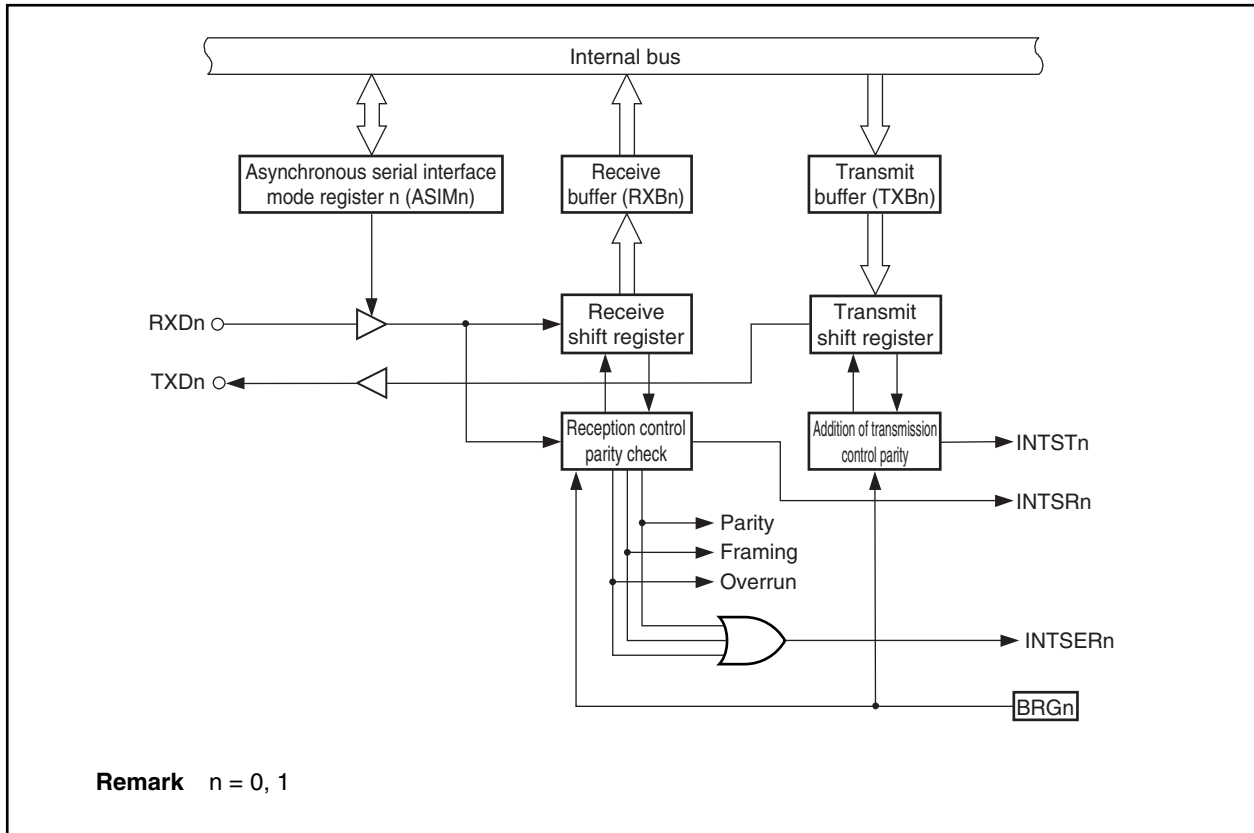
(8) Transmit buffer (TXBn)

TXBn is an 8-bit buffer for transmit data. A transmit operation is started by writing transmit data to TXBn. The transmission completion interrupt request (INTSTn) is generated synchronized with the completion of transmission of one frame.

(9) Addition of transmission control parity

A transmit operation is controlled by adding a start bit, parity bit, or stop bit to the data that is written to the TXBn register, according to the contents that were set in the ASIMn register.

Figure 11-1. Asynchronous Serial Interface Block Diagram



11.2.3 Control registers

(1) Asynchronous serial interface mode registers 0, 1 (ASIM0, ASIM1)

These are 8-bit registers for controlling the transfer operations of UART0 and UART1.

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

Caution To use UARTn, be sure to set the external pins related to the UARTn function in control mode and set clock select register n (CKSRn) and baud rate generator control register n (BRGCn). Then set the UARTCAEn bit to 1 before setting the other bits.

(1/3)

	<7>	<6>	<5>	4	3	2	1	0	Address	After reset
ASIM0	UARTCAE0	TXE0	RXE0	PS01	PS00	CL0	SL0	ISRM0	FFFFFA00H	01H
ASIM1	UARTCAE1	TXE1	RXE1	PS11	PS10	CL1	SL1	ISRM1	FFFFFA10H	01H

Bit Position	Bit Name	Function
7	UARTCAEn (n = 0, 1)	<p>Clock Enable Controls the operation clock (n = 0, 1). 0: Stop supply of clocks to UARTn unit 1: Supplies clocks to UARTn unit</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. When the UARTCAEn bit is set to 0, the UARTn unit can be asynchronously reset. 2. When UARTCAEn = 0, the UARTn unit is in a reset state. Therefore, to operate UARTn, the UARTCAEn bit must be set to 1. 3. When the UARTCAEn bit is changed from 1 to 0, all registers of the UARTn unit are initialized. When the UARTCAEn is set to 1 again, the UARTn unit registers must be set again. <p>The TXDn pin output is always high level in transmit disable state, irrespective of the setting of UARTCAEn bit.</p>
6	TXEn (n = 0, 1)	<p>Transmit Enable Specifies whether transmission is enabled or disabled. 0: Transmission is disabled 1: Transmission is enabled</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. On startup, set UARTCAEn to 1 and then set TXEn to 1. To stop transmission, clear TXEn to 0 and then UARTCAEn to 0. 2. When the transmission unit status is to be initialized, the transmission status may not be able to be initialized unless the TXEn bit is set (1) again after an interval of two cycles of the basic clock has elapsed since the TXEn bit was cleared (0) (For the basic clock, refer to 11.2.6 (1) (a) Basic clock (Clock)).

Bit Position	Bit Name	Function																				
5	RXEn (n = 0, 1)	<p>Receive Enable Specifies whether reception is enabled or disabled. 0: Reception is disabled 1: Reception is enabled</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. On startup, set UARTCAEn to 1 and then set RXEn to 1. To stop transmission, clear RXEn to 0 and then UARTCAEn to 0. 2. When the reception unit status is to be initialized, the reception status may not be able to be initialized unless the RXEn bit is set (1) again after an interval of two cycles of the basic clock has elapsed since the RXEn bit was cleared (0) (For the basic clock, refer to 11.2.6 (1) (a) Basic clock (Clock)) . 																				
4, 3	PSn1, PSn0 (n = 0, 1)	<p>Parity Select Controls the parity bit.</p> <table border="1"> <thead> <tr> <th>PSn1</th> <th>PSn0</th> <th>Transmit operation</th> <th>Receive operation</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Do not output a parity bit</td> <td>Receive with no parity</td> </tr> <tr> <td>0</td> <td>1</td> <td>Output 0 parity</td> <td>Receive as 0 parity</td> </tr> <tr> <td>1</td> <td>0</td> <td>Output odd parity</td> <td>Judge as odd parity</td> </tr> <tr> <td>1</td> <td>1</td> <td>Output even parity</td> <td>Judge as even parity</td> </tr> </tbody> </table> <p>Cautions</p> <ol style="list-style-type: none"> 1. To overwrite the PSn1 and PSn0 bits, first clear (0) the TXEn and RXEn bits. 2. If “0 parity” is selected for reception, no parity judgement is made. Therefore, no error interrupt is generated because the PEn bit of the ASISn register is not set. <ul style="list-style-type: none"> • Even parity If the transmit data contains an odd number of bits with the value “1”, the parity bit is set (1). If it contains an even number of bits with the value “1”, the parity bit is cleared (0). This controls the number of bits with the value “1” contained in the transmit data and the parity bit so that it is an even number. During reception, the number of bits with the value “1” contained in the receive data and the parity bit is counted, and if the number is odd, a parity error is generated. • Odd parity In contrast to even parity, odd parity controls the number of bits with the value “1” contained in the transmit data and the parity bit so that it is an odd number. During reception, the number of bits with the value “1” contained in the receive data and the parity bit is counted, and if the number is even, a parity error is generated. • 0 parity During transmission, the parity bit is cleared (0) regardless of the transmit data. During reception, no parity error is generated because no parity bit is checked. • No parity No parity bit is added to transmit data. During reception, the receive data is considered to have no parity bit. No parity error is generated because there is no parity bit. 	PSn1	PSn0	Transmit operation	Receive operation	0	0	Do not output a parity bit	Receive with no parity	0	1	Output 0 parity	Receive as 0 parity	1	0	Output odd parity	Judge as odd parity	1	1	Output even parity	Judge as even parity
PSn1	PSn0	Transmit operation	Receive operation																			
0	0	Do not output a parity bit	Receive with no parity																			
0	1	Output 0 parity	Receive as 0 parity																			
1	0	Output odd parity	Judge as odd parity																			
1	1	Output even parity	Judge as even parity																			

Bit Position	Bit Name	Function
2	CLn (n = 0, 1)	Character Length Specifies the character length of the transmit/receive data. 0: 7 bits 1: 8 bits Caution To overwrite the CLn bit, first clear (0) the TXEn and RXEn bits.
1	SLn (n = 0, 1)	Stop Bit Length Specifies the stop bit length of the transmit data. 0: 1 bit 1: 2 bits Cautions 1. When overwriting the SLn bit, first clear (0) the TXEn bit. 2. Since reception always operates by using a 1-bit stop bit, the SLn bit setting does not affect receive operations.
0	ISRMn (n = 0, 1)	Interrupt Serial Receive Mode Specifies whether the generation of reception completion interrupt requests when an error occurs is enabled or disabled. 0: A reception error interrupt request (INTSERn) is generated when an error occurs. In this case, no reception completion interrupt request (INTSRn) is generated. 1: A reception completion interrupt request (INTSRn) is generated when an error occurs. In this case, no reception error interrupt request (INTSERn) is generated. Caution When overwriting the ISRMn bit, first clear (0) the RXEn bit.

Remark When reception is disabled, the receive shift register does not detect a start bit. No shift-in processing or transfer processing to the receive buffer is performed, and the contents of the receive buffer are retained.

When reception is enabled, the receive shift operation starts, synchronized with the detection of the start bit, and when the reception of one frame is completed, the contents of the receive shift register are transferred to the receive buffer. A reception completion interrupt (INTSRn) is also generated, synchronized with the transfer to the receive buffer.

(2) Asynchronous serial interface status registers 0, 1 (ASIS0, ASIS1)

These registers, which consist of 3-bit error flags (PE_n, FE_n, and OVE_n), indicate the error status when UART_n reception is completed (n = 0, 1).

The status flag, which indicates a reception error, always indicates the status of the error that occurred most recently. That is, if the same error occurred several times before the receive data was read, this flag would hold only the status of the error that occurred last.

The ASIS_n register is cleared to 00H by a read operation. When a reception error occurs, the receive buffer (RXB_n) should be read after the ASIS_n register is read.

These registers are read-only in 8-bit units.

Caution When the UARTCAEn bit or RXEn bit of the ASIMn register is set to 0, or when the ASISn register is read, the PEn, FE_n, and OVE_n bits of the ASISn register are cleared (0).

	7	6	5	4	3	2	1	0	Address	After reset
ASIS0	0	0	0	0	0	PE0	FE0	OVE0	FFFFFA03H	00H
ASIS1	0	0	0	0	0	PE1	FE1	OVE1	FFFFFA13H	00H

Bit Position	Bit Name	Function
2	PE _n (n = 0, 1)	Parity Error This is a status flag that indicates a parity error. 0: When the UARTCAEn and RXEn bits of the ASIMn register are cleared to 0 or when the ASISn register is read 1: When reception was completed, the transmit data parity did not match the parity bit Caution The operation of the PEn bit differs according to the settings of the PSn1 and PSn0 bits of the ASIMn register.
1	FE _n (n = 0, 1)	Framing Error This is a status flag that indicates a framing error. 0: When the UARTCAEn and RXEn bits of the ASIMn register are cleared to 0 or when the ASISn register is read 1: When reception was completed, no stop bit was detected Caution For receive data stop bits, only the first bit is checked regardless of the stop bit length.
0	OVE _n (n = 0, 1)	Overrun Error This is a status flag that indicates an overrun error. 0: When the UARTCAEn and RXEn bits of the ASIMn register are cleared to 0 or when the ASISn register is read 1: UARTn completed the next receive operation before reading the RXBn receive data. Caution When an overrun error occurs, the next receive data value is not written to the RXBn register and the data is discarded.

(3) Asynchronous serial interface transmission status registers 0, 1 (ASIF0, ASIF1)

These registers, which consist of 2-bit status flags, indicate the status during transmission.

By writing the next data to the TXBn register after data is transferred from the TXBn register to the TXSn register, transmit operations can be performed continuously without suspension even during an interrupt interval. When transmission is performed continuously, data should be written after referencing the TXBFn bit of the ASIFn register to prevent writing to the TXBn register by mistake.

These registers are read-only in 8-bit or 1-bit units.

Remark n = 0, 1

	7	6	5	4	3	2	<1>	<0>	Address	After reset
ASIF0	0	0	0	0	0	0	TXBF0	TXSF0	FFFFFA05H	00H
ASIF1	0	0	0	0	0	0	TXBF1	TXSF1	FFFFFA15H	00H

Bit Position	Bit Name	Function
1	TXBFn (n = 0, 1)	Transmit Buffer Flag This is a transmit buffer data flag. 0: No data to be transferred next exists in the TXBn register (when the UARTCAEn or TXEn bit of the ASIMn register is cleared to 0 or when data has been transferred to the transmit shift register) 1: Data to be transferred next exists in the TXBn register (when data has been written to the TXBn register). Caution To successively transmit data, make sure that this flag is 0, and then write data to the TXBn register. If data is written to the TXBn register while this flag is 1, the transmit data cannot be guaranteed.
0	TXSFn (n = 0, 1)	Transmit Shift Flag This is a transmit shift register data flag. It indicates the transmission status of UARTn. 0: Initial status or waiting for transmission (when the UARTCAEn or TXEn bit of the ASIMn register is cleared to 0 or if no next data is transferred from the TXBn register after completion of transfer). 1: Under transmission (if data is transferred from the TXBn register) Caution Before initializing the transmit unit, make sure that this flag is 0 after occurrence of the transmission completion interrupt. If initialization is executed while this flag is 1, the transmit data is not guaranteed.

(4) Receive buffer registers 0, 1 (RXB0, RXB1)

These are 8-bit buffer registers for storing parallel data that had been converted by the receive shift register. When reception is enabled (RXEn = 1 in the ASIMn register), receive data is transferred from the receive shift register to the receive buffer, synchronized with the completion of the shift-in processing of one frame. Also, a reception completion interrupt request (INTSRn) is generated by the transfer to the receive buffer. For information about the timing for generating these interrupt requests, refer to **11.2.5 (4) Receive operation**. If reception is disabled (RXEn = 0 in the ASIMn register), the contents of the receive buffer are retained, and no processing is performed for transferring data to the receive buffer even when the shift-in processing of one frame is completed. Also, no reception completion interrupt is generated. When 7 bits is specified for the data length, bits 6 to 0 of the RXBn register are transferred for the receive data and the MSB (bit 7) is always 0. However, if an overrun error occurs, the receive data at that time is not transferred to the RXBn register. Except when a reset is input, the RXBn register becomes FFH even when UARTCAEn = 0 in the ASIMn register. These registers are read-only in 8-bit units.

Remark n = 0, 1

	7	6	5	4	3	2	1	0	Address	After reset
RXB0	RXB07	RXB06	RXB05	RXB04	RXB03	RXB02	RXB01	RXB00	FFFFFA02H	FFH
RXB1	RXB17	RXB16	RXB15	RXB14	RXB13	RXB12	RXB11	RXB10	FFFFFA12H	FFH

Bit Position	Bit Name	Function
7 to 0	RXBn7 to RXBn0 (n = 0, 1)	Receive Buffer Stores receive data. 0 can be read for RXBn7 when 7-bit or character data is received.

(5) Transmit buffer registers 0, 1 (TXB0, TXB1)

These are 8-bit buffer registers for setting transmit data.

When transmission is enabled (TXEn = 1 in the ASIMn register), the transmit operation is started by writing data to TXBn.

When transmission is disabled (TXEn = 0 in the ASIMn register), even if data is written to TXBn, the value is ignored.

The TXBn data is transferred to the transmit shift register, and a transmission completion interrupt request (INTSTn) is generated, synchronized with the completion of the transmission of one frame from the transmit shift register. For information about the timing for generating these interrupt requests, refer to 11.2.5 (2)

Transmit operation.

When TXBFn = 1 in the ASIFn register, writing must not be performed to TXBn.

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

Remark n = 0, 1

	7	6	5	4	3	2	1	0	Address	After reset
TXB0	TXB07	TXB06	TXB05	TXB04	TXB03	TXB02	TXB01	TXB00	FFFFFFA04H	FFH
TXB1	TXB17	TXB16	TXB15	TXB14	TXB13	TXB12	TXB11	TXB10	FFFFFFA14H	FFH

Bit Position	Bit Name	Function
7 to 0	TXBn7 to TXBn0 (n = 0, 1)	Transmit Buffer Writes transmit data.

11.2.4 Interrupt requests

The following three types of interrupt requests are generated from UARTn (n = 0, 1).

- Reception error interrupt (INTSERn)
- Reception completion interrupt (INTSRn)
- Transmission completion interrupt (INTSTn)

The default priorities among these three types of interrupt requests is, from high to low, reception error interrupt, reception completion interrupt, and transmission completion interrupt.

Table 11-1. Generated Interrupts and Default Priorities

Interrupt	Priority
Reception error	1
Reception completion	2
Transmission completion	3

(1) Reception error interrupt (INTSERn)

When reception is enabled, a reception error interrupt is generated according to the logical OR of the three types of reception errors explained for the ASISn register. Whether a reception error interrupt (INTSERn) or a reception completion interrupt (INTSRn) is generated when an error occurs can be specified according to the ISRMn bit of the ASIMn register.

When reception is disabled, no reception error interrupt is generated.

(2) Reception completion interrupt (INTSRn)

When reception is enabled, a reception completion interrupt is generated when data is shifted in to the receive shift register and transferred to the receive buffer.

A reception completion interrupt request can be generated in place of a reception error interrupt according to the ISRMn bit of the ASIMn register even when a reception error has occurred.

When reception is disabled, no reception completion interrupt is generated.

(3) Transmission completion interrupt (INTSTn)

A transmission completion interrupt is generated when one frame of transmit data containing 7-bit or 8-bit characters is shifted out from the transmit shift register.

11.2.5 Operation

(1) Data format

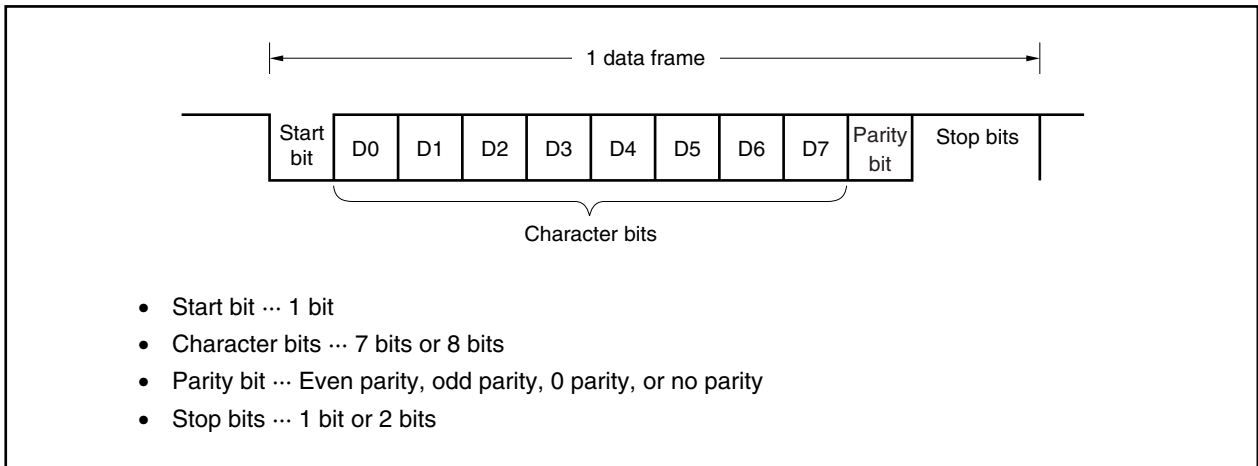
Full-duplex serial data transmission and reception can be performed.

The transmit/receive data format consists of one data frame containing a start bit, character bits, a parity bit, and stop bits as shown in Figure 11-2.

The character bit length within one data frame, the type of parity, and the stop bit length are specified according to the asynchronous serial interface mode register n (ASIMn) ($n = 0, 1$).

Also, data is transferred with the least significant bit (LSB) first.

Figure 11-2. Asynchronous Serial Interface Transmit/Receive Data Format



(2) Transmit operation

When UARTCAEn is set to 1 in the ASIMn register, a high level is output to the TXDn pin.

Then, when TXEn is set to 1 in the ASIMn register, transmission is enabled, and the transmit operation is started by writing transmit data to transmit buffer register n (TXBn) (n = 0, 1).

(a) Transmission enabled state

This state is set by the TXEn bit in the ASIMn register (n = 0, 1).

- TXEn = 1: Transmission enabled state
- TXEn = 0: Transmission disabled state

However, when the transmission enabled state is set, to use UART0 and UART1, which share pins with clocked serial interfaces 0 and 1 (CSI0 and CSI1), the CSICAEn bit of clocked serial interface mode registers 0 and 1 (CSIM0 and CSIM1) should be set to 0.

Since UARTn does not have a CTS (transmission enabled signal) input pin, a port should be used to confirm whether the destination is in a reception enabled state.

(b) Starting a transmit operation

In transmission enabled state, a transmit operation is started by writing transmit data to transmit buffer register n (TXBn). When a transmit operation is started, the data in TXBn is transferred to transmit shift register n (TXSn). Then, the TXSn register outputs data to the TXDn pin sequentially beginning with the LSB (the transmit data is transferred sequentially starting with the start bit). The start bit, parity bit, and stop bits are added automatically (n = 0, 1).

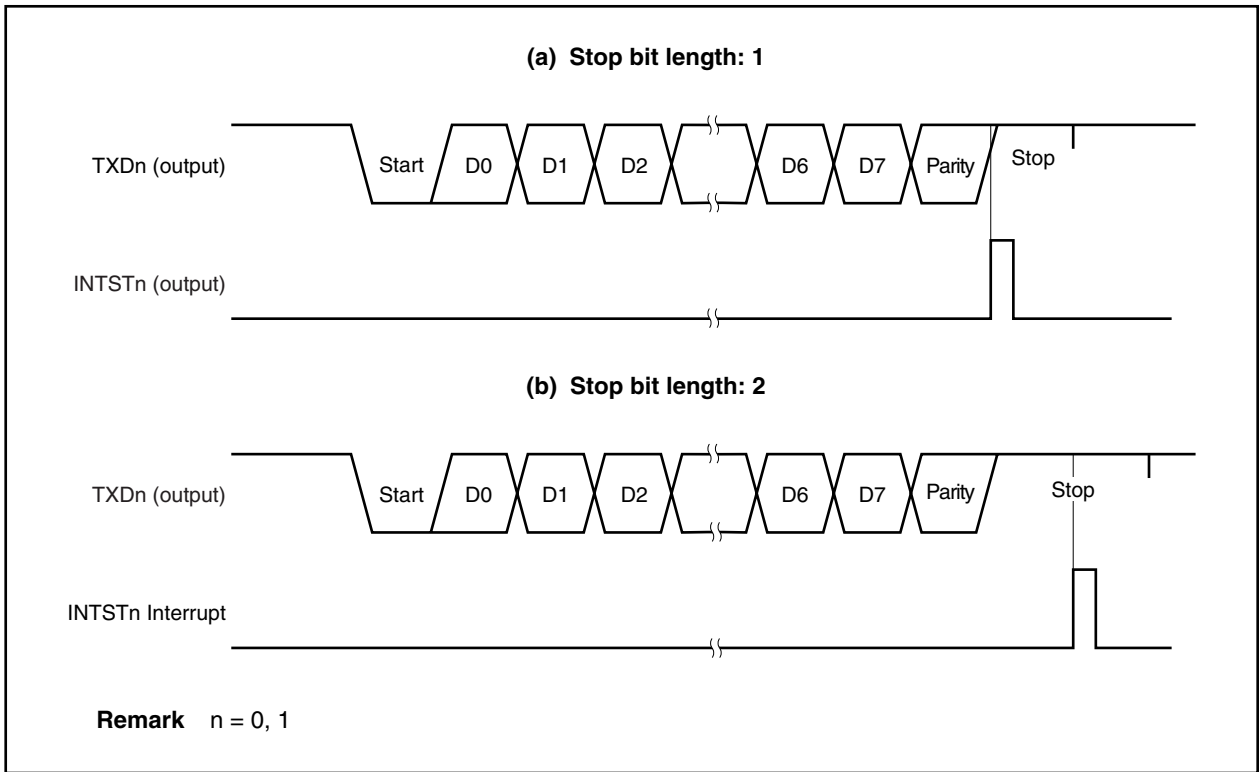
(c) Transmission interrupt request

When the transmit shift register (TXSn) becomes empty, a transmission completion interrupt request (INTSTn) is generated. The timing for generating the INTSTn interrupt differs according to the specification of the stop bit length. The INTSTn interrupt is generated at the same time that the last stop bit is output (n = 0, 1).

If the data to be transmitted next has not been written to the TXBn register, the transmit operation is suspended.

Caution Normally, when transmit shift register n (TXSn) becomes empty, a transmission completion interrupt (INTSTn) is generated. However, no transmission completion interrupt (INTSTn) is generated if transmit shift register n (TXSn) becomes empty due to the input of **RESET**.

Figure 11-3. Asynchronous Serial Interface Transmission Completion Interrupt Timing



★ (3) Continuous transmission operation

UARTn can write the next data to the TXBn register at the time that the TXSn register starts the shift operation. This enables an efficient transmission rate to be realized by continuously transmitting data even during interrupt servicing after the transmission of one data frame (n = 0, 1). By reading the TXSFn bit of the ASIFn register after the transmission completion interrupt has occurred, data can be efficiently written to the TXBn register two times (2 bytes) without having to wait for the transmission time of 1 data frame.

When continuous transmission is performed, data should be written after referencing the ASIFn register to confirm the transmission status and whether or not data can be written to the TXBn register (n = 0, 1).

TXBFn	Enables/Disables Writing to the TXBn Register
0	Enables writing.
1	Disables writing.

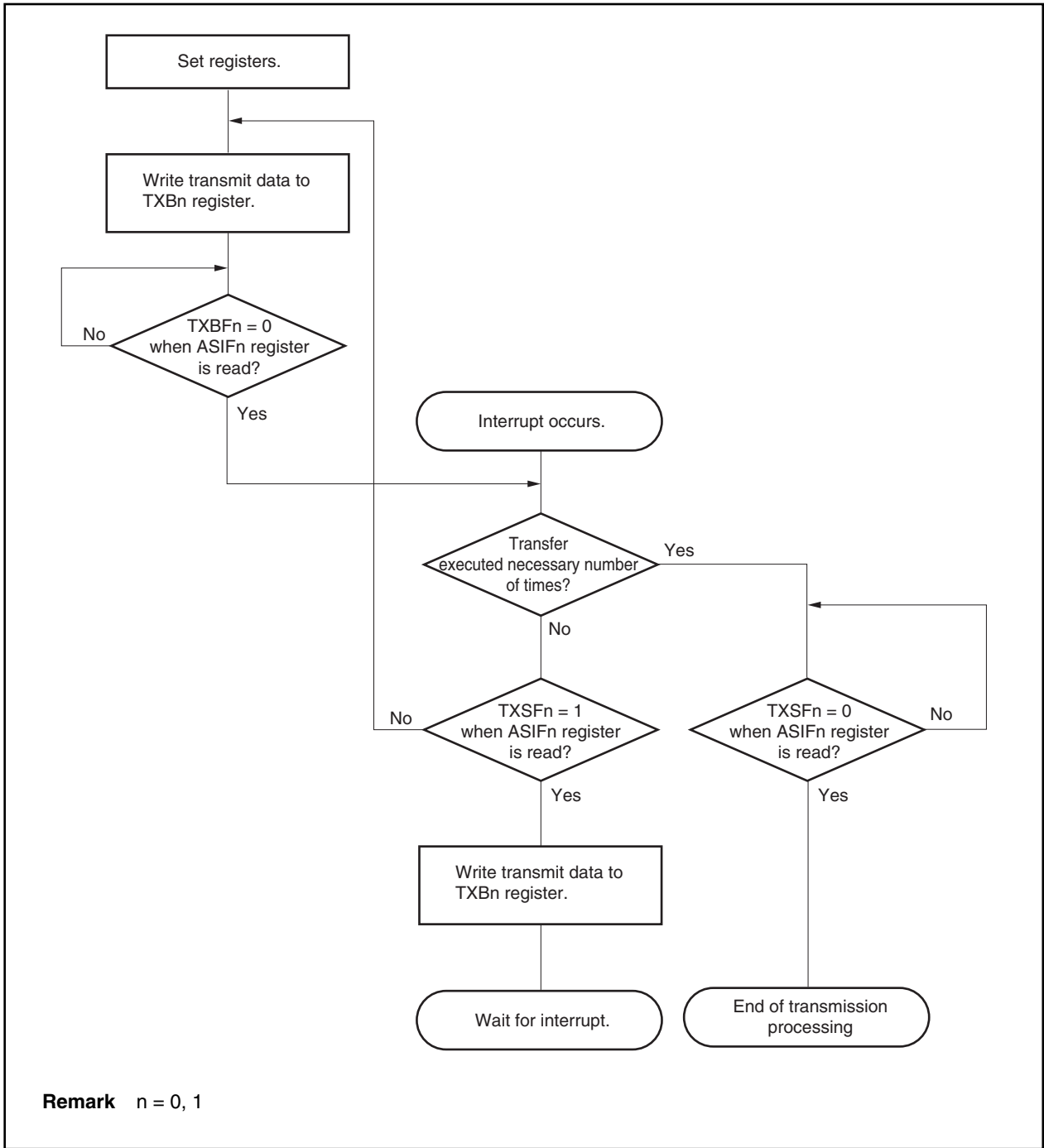
Caution To successively transmit data, make sure that the TXBFn bit is 0 after the first transmit data (first byte) has been written to the TXBn register, before writing the next transmit data (second byte) to the TXBn register. If data is written to the TXBn register while the TXBFn bit is 1, the transmit data is not guaranteed.

While successive transmission is under execution, whether data has been written to the TXBn register can be checked by checking the TXSFn bit after occurrence of the transmission completion interrupt.

TXSFn	Transmission Status
0	Transmission has been completed. However, note Caution of the TXBFn bit. Transmit data can be written two times (2 bytes).
1	Transmission is under execution. Transmit data can be written once (1 byte).

- Cautions**
1. Before initializing the transmit unit after completion of successive transmission, make sure that the TXSFn bit is 0 after the transmission completion interrupt has occurred. If initialization is executed while the TXSFn bit is 1, the transmit data cannot be guaranteed.
 2. While data is being successively transmitted, an overrun error may occur because the next transmission may be completed before the INTSTn interrupt servicing is executed after transmission of 1 data frame. The overrun error can be detected by incorporating a program that can count the number of transmit data and by referencing the TXSFn bit.

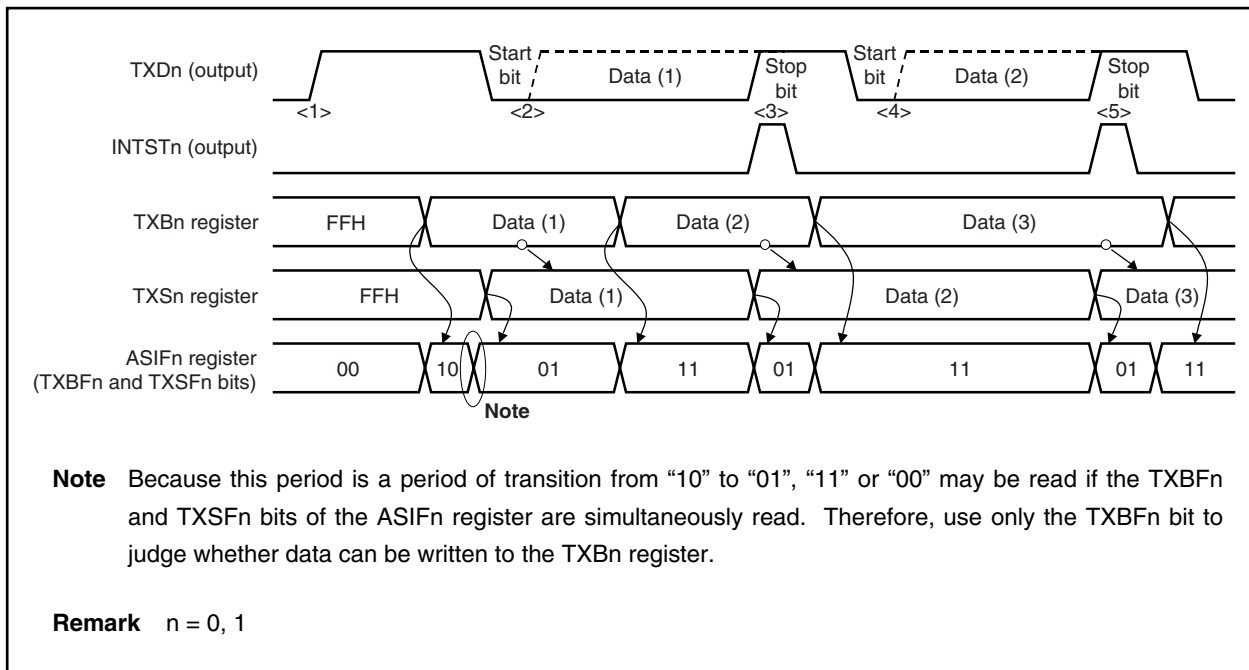
Figure 11-4. Continuous Transmission Processing Flow



(a) Starting procedure

The procedure for starting continuous transmission is shown below.

Figure 11-5. Continuous Transmission Starting Procedure



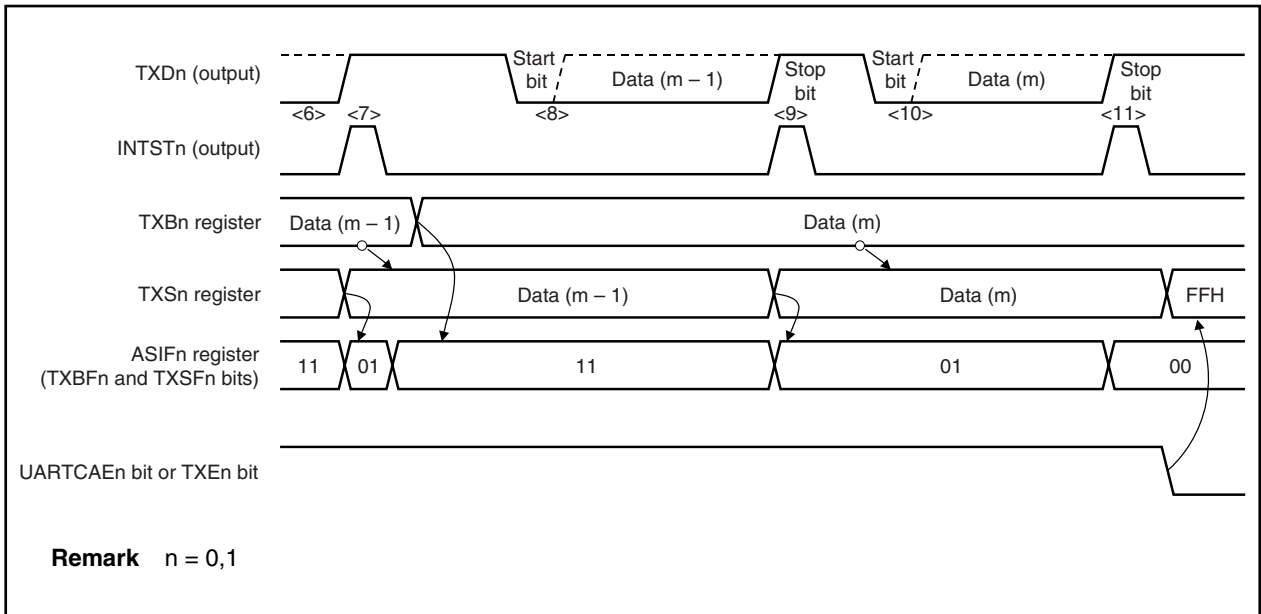
Transmission starting procedure	Internal operation	ASIFn register	
		TXBFn	TXSFn
• Set transmission mode	<1> Start transmission unit	0	0
• Write data (1)	<2> Generate start bit	1	1/0 ^{Note}
	Start data (1) transmission	0	1
• Read ASIFn register (confirm that TXBFn bit = 0)		<u>0</u>	1
• Write data (2)	<<Transmission in progress>>	1	1
	<3> Generate INTSTn interrupt	0	1
• Read ASIFn register (confirm that TXBFn bit = 0)		<u>0</u>	1
• Write data (3)	<4> Generate start bit	1	1
	Start data (2) transmission		
	<<Transmission in progress>>		
	<5> Generate INTSTn interrupt	0	1
• Read ASIFn register (confirm that TXBFn bit = 0)		<u>0</u>	1
• Write data (4)		1	1

Note Transition period

(b) Ending procedure

The procedure for ending continuous transmission is shown below.

Figure 11-6. Continuous Transmission Ending Procedure



★

Transmission ending procedure	Internal operation	ASIFn register	
		TXBFn	TXSFn
<ul style="list-style-type: none"> • Read ASIFn register (confirm that the TXBFn bit = 0) • Write data (n) 	<6> Transmission of data (m - 2) is in progress	1	1
	<7> Generate INTST interrupt	0	1
<ul style="list-style-type: none"> • Read ASIFn register (confirm that the TXSFn bit = 1) There is no write data 	<8> Generate start bit Start data (m - 1) transmission <<Transmission in progress>>	1	1
	<9> Generate INTSTn interrupt	0	1
<ul style="list-style-type: none"> • Read ASIFn register (confirm that the TXSFn bit = 0) • Clear (0) the UARTCAEn bit or TXEn bit 	<10> Generate start bit Start data (m) transmission <<Transmission in progress>>	0	0
	<11> Generate INTSTn interrupt	0	0
	Initialize internal circuits	0	0

(4) Receive operation

An awaiting reception state is set by setting UARTCAEn to 1 in the ASIMn register and then setting RXEn to 1 in the ASIMn register. RXDn pin sampling begins and a start bit is detected. When the start bit is detected, the receive operation begins, and data is stored sequentially in the receive shift register according to the baud rate that was set. A reception completion interrupt (INTSRn) is generated each time the reception of one frame of data is completed. Normally, the receive data is transferred from the receive buffer (RXBn) to memory by this interrupt servicing (n = 0, 1).

(a) Reception enabled state

The receive operation is set to reception enabled state by setting the RXEn bit in the ASIMn register to 1 (n = 0, 1).

- RXEn = 1: Reception enabled state
- RXEn = 0: Reception disabled state

However, when the reception enabled state is set, to use UART0 and UART1, which share pins with clocked serial interfaces 0 and 1 (CSI0 and CSI1), the operation of CSIn must be disabled by setting the CSICAEn bit of clocked serial interface mode registers 0 and 1 (CSIM0 and CSIM1) to 0 (n = 0, 1).

In reception disabled state, the reception hardware stands by in the initial state. At this time, the contents of the receive buffer are retained, and no reception completion interrupt or reception error interrupt is generated.

(b) Starting a receive operation

A receive operation is started by the detection of a start bit.

The RXDn pin is sampled according to the serial clock from the baud rate generator (BRGn) (n = 0, 1).

(c) Reception completion interrupt

When RXEn = 1 in the ASIMn register and the reception of one frame of data is completed (the stop bit is detected), a reception completion interrupt (INTSRn) is generated and the receive data within the receive shift register is transferred to RXBn at the same time (n = 0, 1).

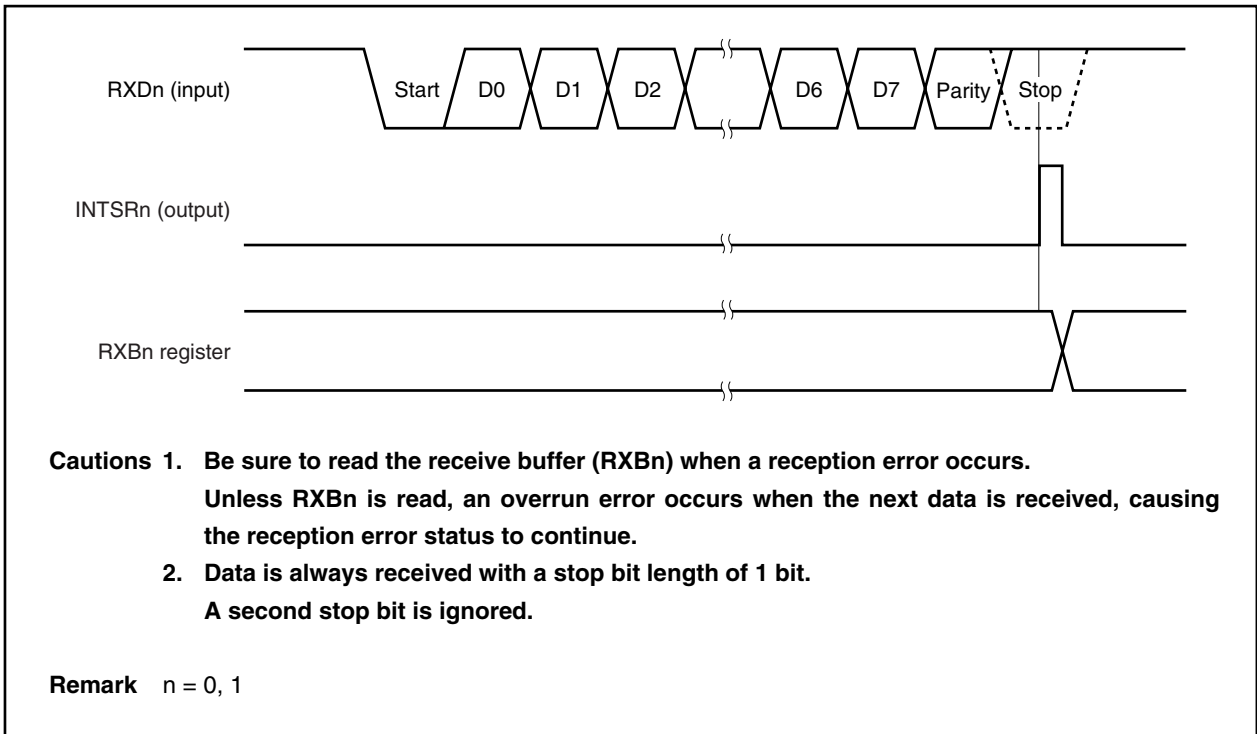
Also, if an overrun error occurs, the receive data at that time is not transferred to the receive buffer (RXBn), and either a reception completion interrupt (INTSRn) or a reception error interrupt (INTSERn) is generated according to the ISRMn bit setting in the ASIMn register.

Even if a parity error or framing error occurs during a receive operation, the receive operation continues until a stop bit is received, and after reception is completed, either a reception completion interrupt (INTSRn) or a reception error interrupt (INTSERn) is generated (the receive data within the receive shift register is transferred to RXBn) according to the ISRMn bit setting in the ASIMn register.

If the RXEn bit is reset (0) during a receive operation, the receive operation is immediately stopped. The contents of the receive buffer (RXBn) and of the asynchronous serial interface status register (ASISn) at this time do not change, and no reception completion interrupt (INTSRn) or reception error interrupt (INTSERn) is generated.

No reception completion interrupt is generated when RXEn = 0 (reception is disabled).

★ **Figure 11-7. Asynchronous Serial Interface Reception Completion Interrupt Timing**



(5) Reception error

The three types of error that can occur during a receive operation are a parity error, framing error, or overrun error. The data reception result is that the various flags of the ASISn register are set (1), and a reception error interrupt (INTSERn) or a reception completion interrupt (INTSRn) is generated at the same time. The ISRMn bit of the ASIMn register specifies whether INTSERn or INTSRn is generated.

The type of error that occurred during reception can be detected by reading the contents of the ASISn register during the INTSERn or INTSRn interrupt servicing.

The contents of the ASISn register are reset (0) by reading the ASISn register.

Table 11-2. Reception Error Causes

Error Flag	Reception Error	Cause
PEn	Parity error	The parity specification during transmission did not match the parity of the reception data
FEn	Framing error	No stop bit was detected
OVEEn	Overrun error	The reception of the next data was completed before data was read from the receive buffer

Remark n = 0, 1

(a) Separation of reception error interrupt

A reception error interrupt can be separated from the INTSRn interrupt and generated as an INTSERn interrupt by clearing the ISRMn bit of the ASIMn register (n = 0, 1) to 0.

Figure 11-8. When Reception Error Interrupt Is Separated from INTSRn Interrupt (ISRMn Bit = 0)

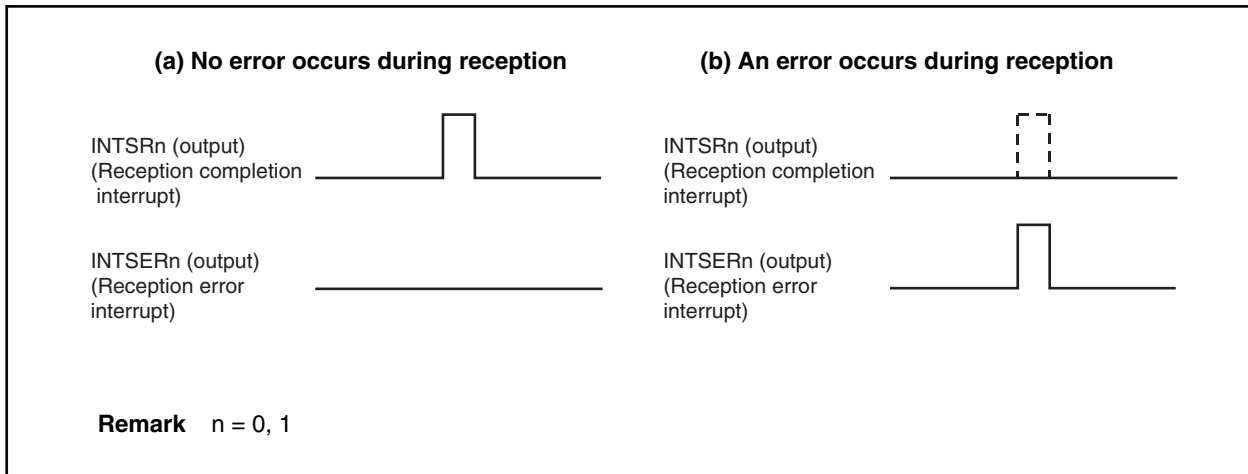
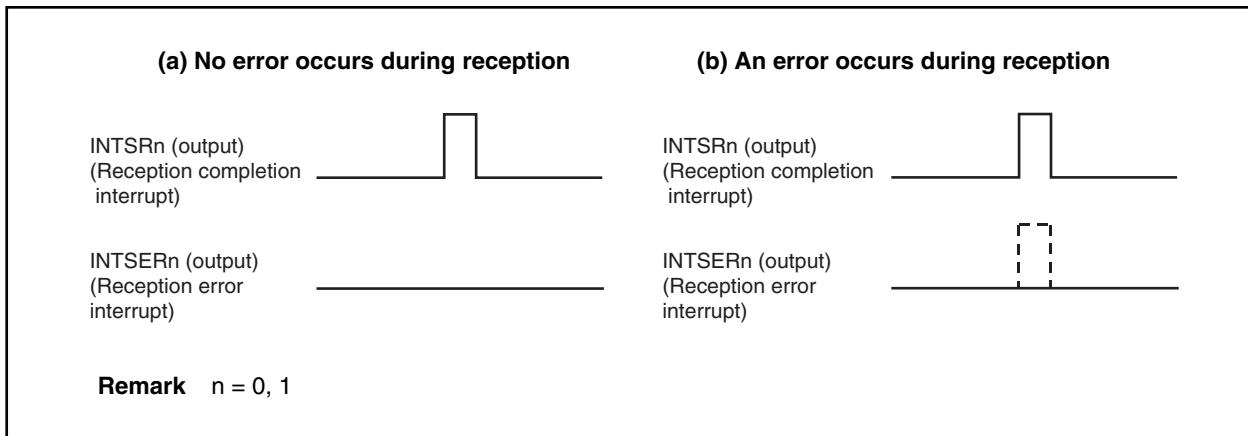


Figure 11-9. When Reception Error Interrupt Is Included in INTSRn Interrupt (ISRMn Bit = 1)



(6) Parity types and corresponding operation

A parity bit is used to detect a bit error in communication data. Normally, the same type of parity bit is used at the transmission and reception sides.

(a) Even parity**(i) During transmission**

The parity bit is controlled so that the number of bits with the value “1” within the transmit data including the parity bit is even. The parity bit value is as follows.

- If the number of bits with the value “1” within the transmit data is odd: 1
- If the number of bits with the value “1” within the transmit data is even: 0

(ii) During reception

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

(b) Odd parity**(i) During transmission**

In contrast to even parity, the parity bit is controlled so that the number of bits with the value “1” within the transmit data including the parity bit is odd. The parity bit value is as follows.

- If the number of bits with the value “1” within the transmit data is odd: 0
- If the number of bits with the value “1” within the transmit data is even: 1

(ii) During reception

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

(c) 0 parity

During transmission the parity bit is set to “0” regardless of the transmit data.

During reception, no parity bit check is performed. Therefore, no parity error is generated regardless of whether the parity bit is “0” or “1”.

(d) No parity

No parity bit is added to the transmit data.

During reception, the receive operation is performed as if there were no parity bit. Since there is no parity bit, no parity error is generated.

(7) Receive data noise filter

The RXDn signal is sampled at the rising edge of the prescaler output clock. If the same sampling value is obtained twice, the match detector output changes, and this output is sampled as input data. Therefore, data not exceeding one clock width is judged to be noise and is not delivered to the internal circuit (refer to **Figure 11-11**). Refer to **11.2.6 (1) (a) Basic clock (Clock)** regarding the basic clock.

Also, since the circuit is configured as shown in Figure 11-10, internal processing during a receive operation is delayed by up to 2 clocks according to the external signal status.

Figure 11-10. Noise Filter Circuit

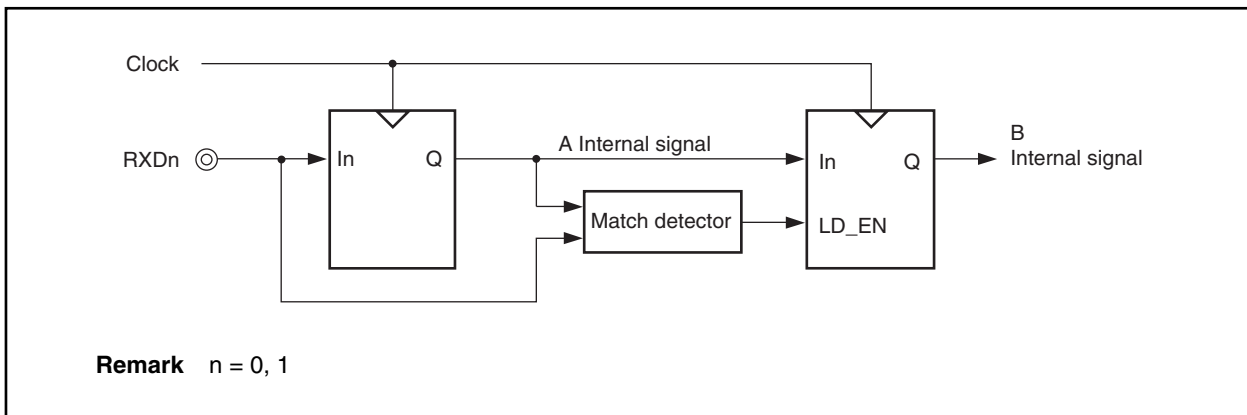
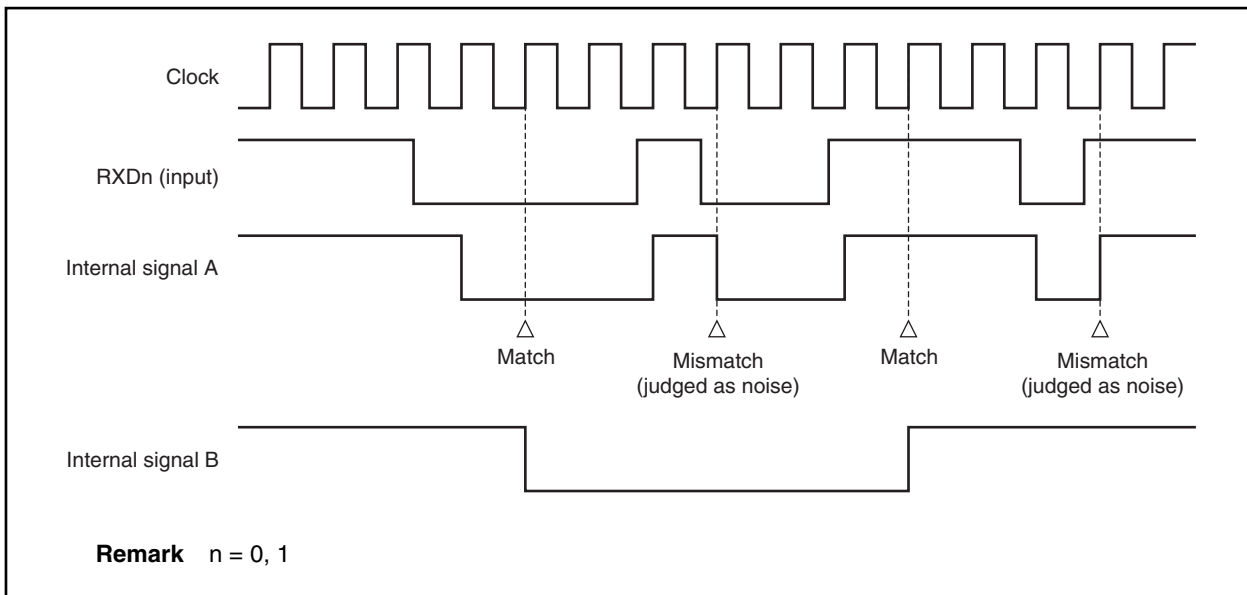


Figure 11-11. Timing of RXDn Signal Judged as Noise



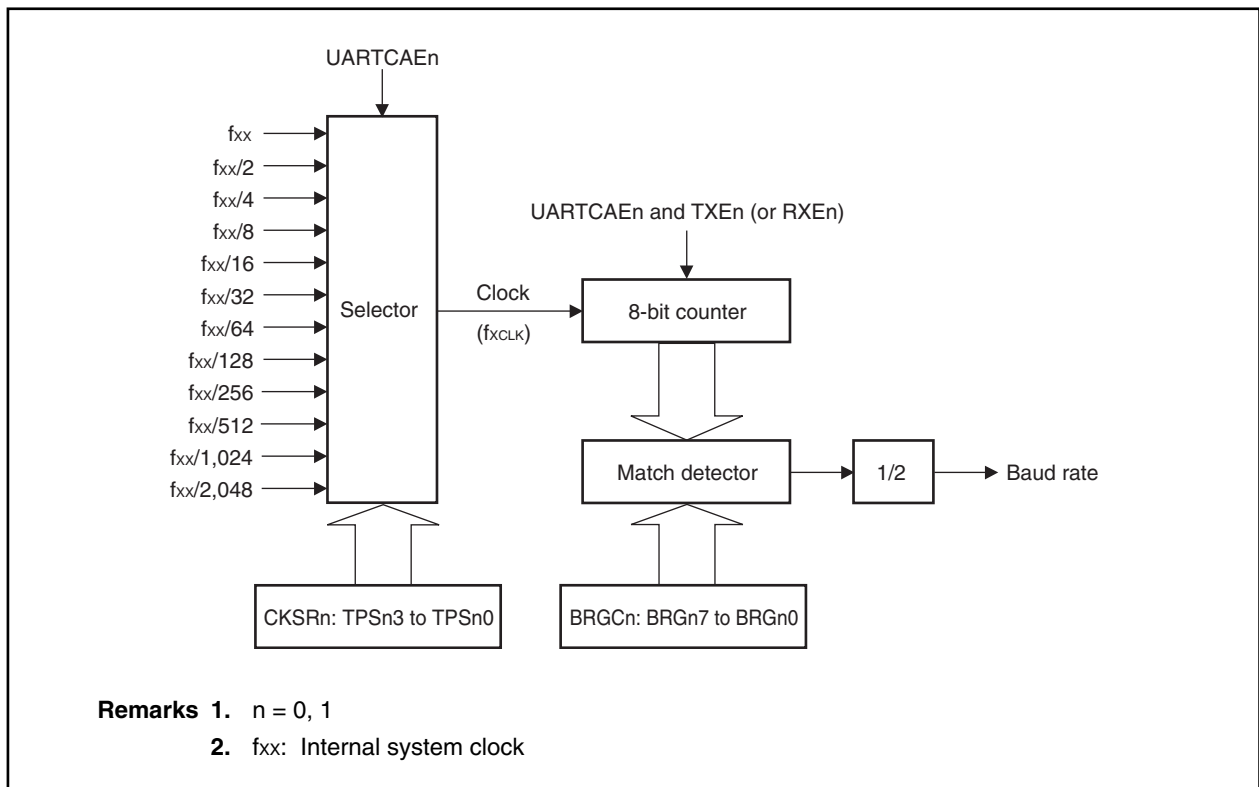
11.2.6 Dedicated baud rate generators 0, 1 (BRG0, BRG1)

A dedicated baud rate generator, which consists of a source clock selector and an 8-bit programmable counter, generates serial clocks during transmission/reception in UARTn. The dedicated baud rate generator output can be selected as the serial clock for each channel.

Separate 8-bit counters exist for transmission and for reception.

(1) Baud rate generator configuration

Figure 11-12. Baud Rate Generator Configuration



(a) Basic clock (Clock)

When $UARTCAEn = 1$ in the $ASIMn$ register, the clock selected according to the $TPSn3$ to $TPSn0$ bits of the $CKSRn$ register is supplied to the transmission/reception unit. This clock is called the basic clock, and its frequency is referred to as f_{CLK} . When $UARTCAEn = 0$, the clock signal is fixed at low level.

(2) Serial clock generation

A serial clock can be generated according to the settings of the CKSRn and BRGCn registers (n = 0, 1). The input clock to the 8-bit counter is selected according to the TPSn3 to TPSn0 bits of the CKSRn register. The 8-bit counter divisor value can be selected according to the BRGn7 to BRGn0 bits of the BRGCn register.

(a) Clock select registers 0, 1 (CKSR0, CKSR1)

The CKSRn register is an 8-bit register for selecting the input block according to the TPSn3 to TPSn0 bits. The clock selected by the TPSn3 to TPSn0 bits becomes the basic clock of the transmission/reception module. Its frequency is referred to as f_{CLK} . These registers can be read or written in 8-bit units.

Cautions 1. The maximum allowable frequency of the basic clock (f_{CLK}) is 20 MHz. Therefore, when the system clock's frequency is 40 MHz, bits TPSn3 to TPSn0 cannot be set to 0000B (n = 0, 1).

If the system clock frequency is 40 MHz, set the TPSn3 to TPSn0 bits to a value other than 0000B and set the UARTCAEn bit of the ASIMn register to 1.

★ 2. If the TPSn3 to TPSn0 bits are to be overwritten, the UARTCAEn bit of the ASIMn register should be set to 0 first.

	7	6	5	4	3	2	1	0	Address	After reset
CKSR0	0	0	0	0	TPS03	TPS02	TPS01	TPS00	FFFFFFA06H	00H
CKSR1	0	0	0	0	TPS13	TPS12	TPS11	TPS10	FFFFFFA16H	00H

Bit Position	Bit Name	Function																																																																						
3 to 0	TPSn3 to TPSn0 (n = 0, 1)	Specifies the basic clock <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">TPSn3</th> <th style="width: 10%;">TPSn2</th> <th style="width: 10%;">TPSn1</th> <th style="width: 10%;">TPSn0</th> <th style="width: 60%;">Basic Clock (f_{CLK})</th> </tr> </thead> <tbody> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td>f_{xx}</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td>$f_{xx}/2$</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td>$f_{xx}/4$</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td>$f_{xx}/8$</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td>$f_{xx}/16$</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td>$f_{xx}/32$</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td>$f_{xx}/64$</td></tr> <tr><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td>$f_{xx}/128$</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td>$f_{xx}/256$</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td>$f_{xx}/512$</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td>$f_{xx}/1,024$</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">0</td><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td>$f_{xx}/2,048$</td></tr> <tr><td style="text-align: center;">1</td><td style="text-align: center;">1</td><td style="text-align: center;">Arbitrary</td><td style="text-align: center;">Arbitrary</td><td>Setting prohibited</td></tr> </tbody> </table>	TPSn3	TPSn2	TPSn1	TPSn0	Basic Clock (f_{CLK})	0	0	0	0	f_{xx}	0	0	0	1	$f_{xx}/2$	0	0	1	0	$f_{xx}/4$	0	0	1	1	$f_{xx}/8$	0	1	0	0	$f_{xx}/16$	0	1	0	1	$f_{xx}/32$	0	1	1	0	$f_{xx}/64$	0	1	1	1	$f_{xx}/128$	1	0	0	0	$f_{xx}/256$	1	0	0	1	$f_{xx}/512$	1	0	1	0	$f_{xx}/1,024$	1	0	1	1	$f_{xx}/2,048$	1	1	Arbitrary	Arbitrary	Setting prohibited
TPSn3	TPSn2	TPSn1	TPSn0	Basic Clock (f_{CLK})																																																																				
0	0	0	0	f_{xx}																																																																				
0	0	0	1	$f_{xx}/2$																																																																				
0	0	1	0	$f_{xx}/4$																																																																				
0	0	1	1	$f_{xx}/8$																																																																				
0	1	0	0	$f_{xx}/16$																																																																				
0	1	0	1	$f_{xx}/32$																																																																				
0	1	1	0	$f_{xx}/64$																																																																				
0	1	1	1	$f_{xx}/128$																																																																				
1	0	0	0	$f_{xx}/256$																																																																				
1	0	0	1	$f_{xx}/512$																																																																				
1	0	1	0	$f_{xx}/1,024$																																																																				
1	0	1	1	$f_{xx}/2,048$																																																																				
1	1	Arbitrary	Arbitrary	Setting prohibited																																																																				

Remark f_{xx} : Internal system clock

(b) Baud rate generator control registers 0, 1 (BRGC0, BRGC1)

The BRGCn register is an 8-bit register that controls the baud rate (serial transfer speed) of UARTn. These registers can be read or written in 8-bit units.

Caution If the BRGn7 to BRGn0 bits are to be overwritten, TXEn and RXEn should be set to 0 in the ASIMn register first (n = 0, 1).

	7	6	5	4	3	2	1	0	Address	After reset
BRGC0	MDL07	MDL06	MDL05	MDL04	MDL03	MDL02	MDL01	MDL00	FFFFFFA07H	FFH
BRGC1	MDL17	MDL16	MDL15	MDL14	MDL13	MDL12	MDL11	MDL10	FFFFFFA17H	FFH

Bit Position	Bit Name	Function																																																																																																														
7 to 0	BRGn7 to BRGn0 (n = 0, 1)	Specifies the 8-bit counter's divisor value. <table border="1" style="margin: 10px auto; border-collapse: collapse; text-align: center;"> <thead> <tr> <th>BRGn7</th> <th>BRGn6</th> <th>BRGn5</th> <th>BRGn4</th> <th>BRGn3</th> <th>BRGn2</th> <th>BRGn1</th> <th>BRGn0</th> <th>Divisor Value (k)</th> <th>Serial Clock</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>×</td> <td>×</td> <td>×</td> <td>–</td> <td>Setting prohibited</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>8</td> <td>f_{XCLK}/8</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>1</td> <td>9</td> <td>f_{XCLK}/9</td> </tr> <tr> <td>0</td> <td></td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>10</td> <td>f_{XCLK}/10</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>250</td> <td>f_{XCLK}/250</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>1</td> <td>251</td> <td>f_{XCLK}/251</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>252</td> <td>f_{XCLK}/252</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>253</td> <td>f_{XCLK}/253</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>254</td> <td>f_{XCLK}/254</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>255</td> <td>f_{XCLK}/255</td> </tr> </tbody> </table>	BRGn7	BRGn6	BRGn5	BRGn4	BRGn3	BRGn2	BRGn1	BRGn0	Divisor Value (k)	Serial Clock	0	0	0	0	0	×	×	×	–	Setting prohibited	0	0	0	0	1	0	0	0	8	f _{XCLK} /8	0	0	0	0	1	0	0	1	9	f _{XCLK} /9	0		0	0	1	0	1	0	10	f _{XCLK} /10	1	1	1	1	1	0	1	0	250	f _{XCLK} /250	1	1	1	1	1	0	1	1	251	f _{XCLK} /251	1	1	1	1	1	1	0	0	252	f _{XCLK} /252	1	1	1	1	1	1	0	1	253	f _{XCLK} /253	1	1	1	1	1	1	1	0	254	f _{XCLK} /254	1	1	1	1	1	1	1	1	255	f _{XCLK} /255
BRGn7	BRGn6	BRGn5	BRGn4	BRGn3	BRGn2	BRGn1	BRGn0	Divisor Value (k)	Serial Clock																																																																																																							
0	0	0	0	0	×	×	×	–	Setting prohibited																																																																																																							
0	0	0	0	1	0	0	0	8	f _{XCLK} /8																																																																																																							
0	0	0	0	1	0	0	1	9	f _{XCLK} /9																																																																																																							
0		0	0	1	0	1	0	10	f _{XCLK} /10																																																																																																							
1	1	1	1	1	0	1	0	250	f _{XCLK} /250																																																																																																							
1	1	1	1	1	0	1	1	251	f _{XCLK} /251																																																																																																							
1	1	1	1	1	1	0	0	252	f _{XCLK} /252																																																																																																							
1	1	1	1	1	1	0	1	253	f _{XCLK} /253																																																																																																							
1	1	1	1	1	1	1	0	254	f _{XCLK} /254																																																																																																							
1	1	1	1	1	1	1	1	255	f _{XCLK} /255																																																																																																							

Remarks

1. f_{XCLK}: Frequency of clock selected according to TPSn3 to TPSn0 bits of CKSRn register.
2. k: Value set according to BRGn7 to BRGn0 bits (k = 8, 9, 10, ..., 255)
3. ×: don't care

(c) Baud rate error

The baud rate is the value obtained according to the following formula.

$$\text{Baud rate} = \frac{f_{\text{CLK}}}{2 \times k} \text{ [bps]}$$

f_{CLK} = Frequency of basic clock selected according to TPSn3 to TPSn0 bits of CKSRn register.

k = Value set according to BRGn7 to BRGn0 bits of BRGCn register ($k = 8, 9, 10, \dots, 255$)

The baud rate error is obtained according to the following formula.

$$\text{Error (\%)} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (normal baud rate)}} - 1 \right) \times 100 \text{ [\%]}$$

- Cautions**
1. Make sure that the baud rate error during transmission does not exceed the allowable error of the reception destination.
 2. Make sure that the baud rate error during reception is within the allowable baud rate range during reception, which is described in paragraph (4).

Example: Basic clock frequency = 20 MHz = 20,000,000 Hz
 Settings of BRGn7 to BRGn0 bits in BRGCn register = 01000001B ($k = 65$)
 Target baud rate = 153,600 bps

$$\begin{aligned} \text{Baud rate} &= 20 \text{ M}/(2 \times 65) \\ &= 20,000,000/(2 \times 65) = 153,846 \text{ [bps]} \end{aligned}$$

$$\begin{aligned} \text{Error} &= (153,846/153,600 - 1) \times 100 \\ &= 0.160 \text{ [\%]} \end{aligned}$$

(3) Baud rate setting example

Table 11-3. Baud Rate Generator Setting Data

Baud Rate (bps)	f _{xx} = 40 MHz			f _{xx} = 33 MHz			f _{xx} = 10 MHz		
	f _{xCLK}	k	ERR	f _{xCLK}	k	ERR	f _{xCLK}	k	ERR
300	f _{xx} /2 ¹⁰	65	0.16	f _{xx} /2 ⁸	215	-0.07	f _{xx} /2 ⁷	130	0.16
600	f _{xx} /2 ⁹	65	0.16	f _{xx} /2 ⁷	215	-0.07	f _{xx} /2 ⁶	130	0.16
1,200	f _{xx} /2 ⁸	65	0.16	f _{xx} /2 ⁶	215	-0.07	f _{xx} /2 ⁵	130	0.16
2,400	f _{xx} /2 ⁷	65	0.16	f _{xx} /2 ⁵	215	-0.07	f _{xx} /2 ⁴	130	0.16
4,800	f _{xx} /2 ⁶	65	0.16	f _{xx} /2 ⁴	215	-0.07	f _{xx} /2 ³	130	0.16
9,600	f _{xx} /2 ⁵	65	0.16	f _{xx} /2 ³	215	-0.07	f _{xx} /2 ²	130	0.16
19,200	f _{xx} /2 ⁴	80	0.16	f _{xx} /2 ²	215	-0.07	f _{xx} /2 ¹	130	0.16
31,250	f _{xx} /2 ³	65	0	f _{xx} /2 ²	132	0	f _{xx} /2 ¹	80	0
38,400	f _{xx} /2 ³	65	0.16	f _{xx} /2 ¹	215	-0.07	f _{xx} /2 ⁰	130	0.16
76,800	f _{xx} /2 ²	65	0.16	f _{xx} /2 ¹	107	0.39	f _{xx} /2 ⁰	65	0.16
153,600	f _{xx} /2 ¹	65	0.16	f _{xx} /2 ¹	54	-0.54	f _{xx} /2 ⁰	33	-1.36
312,500	f _{xx} /2 ¹	32	0	f _{xx} /2 ¹	26	1.54	f _{xx} /2 ⁰	16	0

Caution The maximum allowable frequency of the basic clock (f_{xCLK}) is 20 MHz.

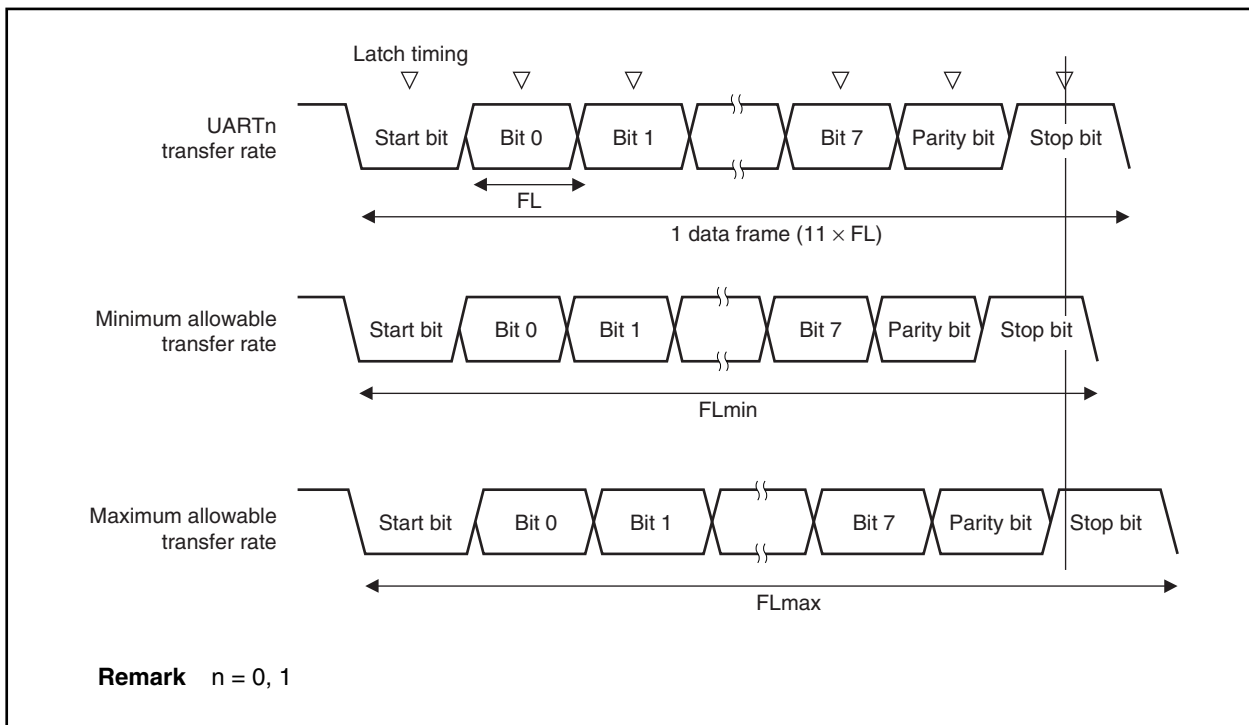
Remarks f_{xx}: Internal system clock
 f_{xCLK}: Basic clock
 k: Settings of BRGn7 to BRGn0 bits in BRGCn register (n = 0, 1)
 ERR: Baud rate error [%]

(4) Allowable baud rate range during reception

The degree to which a discrepancy from the transmission destination's baud rate is allowed during reception is shown below.

Caution The equations described below should be used to set the baud rate error during reception so that it always is within the allowable error range.

Figure 11-13. Allowable Baud Rate Range During Reception



As shown in Figure 11-13, after the start bit is detected, the receive data latch timing is determined according to the counter that was set by the BRGCn register. If all data up to the final data (stop bit) is in time for this latch timing, the data can be received normally.

Applying this to 11-bit reception is, theoretically, as follows.

$$FL = (\text{Brate})^{-1}$$

- Brate: UARTn baud rate (n = 0, 1)
- k: BRGCn setting value (n = 0, 1)
- FL: 1-bit data length
- Latch timing margin: 2 clocks

$$\text{Minimum allowable transfer rate: } FL_{\min} = 11 \times FL - \frac{k - 2}{2k} \times FL = \frac{21k + 2}{2k} FL$$

Therefore, the transfer destination's maximum baud rate that can be received is as follows.

$$BR_{max} = (FL_{min}/11)^{-1} = \frac{22k}{21k + 2} \text{ Brate}$$

Similarly, the maximum allowable transfer rate can be obtained as follows.

$$\frac{10}{11} \times FL_{max} = 11 \times FL - \frac{k + 2}{2 \times k} \times FL = \frac{21k - 2}{2 \times k} FL$$

$$FL_{max} = \frac{21k - 2}{20k} FL \times 11$$

Therefore, the transfer destination's minimum baud rate that can be received is as follows.

$$BR_{min} = (FL_{max}/11)^{-1} = \frac{20k}{21k - 2} \text{ Brate}$$

The allowable baud rate error of UARTn and the transfer destination can be obtained as follows from the expressions described above for computing the minimum and maximum baud rate values.

Table 11-4. Maximum and Minimum Allowable Baud Rate Error

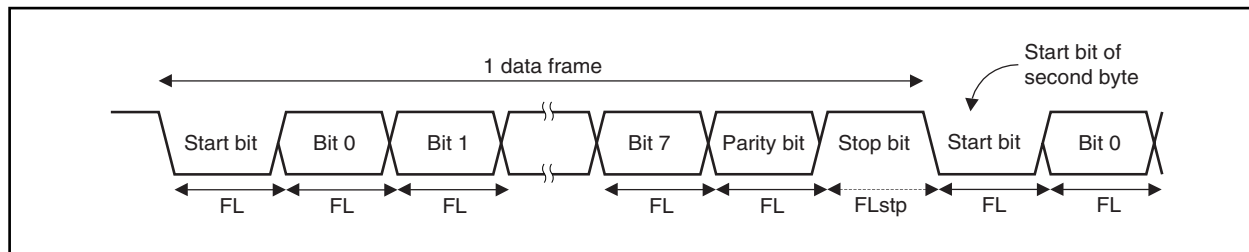
Division Ratio (k)	Maximum Allowable Baud Rate Error	Minimum Allowable Baud Rate Error
8	+3.53%	-3.61%
20	+4.26%	-4.31%
50	+4.56%	-4.58%
100	+4.66%	-4.67%
255	+4.72%	-4.73%

- Remarks 1.** The reception precision depends on the number of bits in one frame, the input clock frequency, and the division ratio (k). The higher the input clock frequency and the larger the division ratio (k), the higher the precision.
- 2.** k: BRGCn setting value (n = 0, 1)

(5) Transfer rate during continuous transmission

During continuous transmission, the transfer rate from a stop bit to the next start bit is extended two clocks longer than normal. However, on the reception side, the transfer result is not affected since the timing is initialized by the detection of the start bit.

Figure 11-14. Transfer Rate During Continuous Transmission



Representing the 1-bit data length by FL, the stop bit length by FLstp, and the basic clock frequency by f_{CLK} yields the following equation.

$$FL_{stp} = FL + 2/f_{CLK}$$

Therefore, the transfer rate during continuous transmission is as follows.

$$\text{Transfer rate} = 11 \times FL + 2/f_{CLK}$$

11.2.7 Precautions

The points to be noted when using UARTn are described below (n = 0, 1).

- (1) When the supply of clocks to UARTn is stopped (for example, IDLE or software STOP mode), operation stops with each register retaining the value it had immediately before the supply of clocks was stopped. The TXDn pin output also holds and outputs the value it had immediately before the supply of clocks was stopped. However, operation is not guaranteed after the supply of clocks is restarted. Therefore, after the supply of clocks is restarted, the circuits should be initialized by setting $UARTCAEn = 0$, $RXEn = 0$, and $TXEn = 0$.
- ★ (2) UARTn has a two-buffer configuration, consisting of transmit buffers (TXBn) and transmit shift registers, and has status flags (TXBFn and TXSFn bits of the ASIFn register) that indicate the status of the respective buffers. If the TXBFn and TXSFn bits are read at the same time during successive transmission, these bits change from "10" to "01". Because this change timing is the data transition period from TXBn to the transmit shift register, however, "11" or "00" may be read depending on the timing. To successively transmit data, therefore, read only the TXBFn bit.

11.3 Clocked Serial Interfaces 0, 1 (CSI0, CSI1)

11.3.1 Features

- Transfer rate: Master mode: Maximum 2.5 Mbps (when internal system clock operates at 40 MHz)
Slave mode: Maximum 4 Mbps
- Half-duplex communications
- Master mode and slave mode can be selected
- Transmission data length: 8 bits
- Transfer data direction can be switched between MSB first and LSB first
- Eight clock signals can be selected (7 master clocks and 1 slave clock)
- 3-wire method
 - SO_n: Serial data output
 - SIn: Serial data input
 - $\overline{\text{SCKn}}$: Serial clock I/O
- Interrupt sources: 1 type
 - Transmission/reception completion interrupt (INTCSIn)
- Transmission/reception mode or reception-only mode can be specified
- On-chip transmit buffer (SOTB_n)

Remark n = 0, 1

11.3.2 Configuration

CSIn is controlled by the clocked serial interface mode register (CSIM_n) (n = 0, 1). Transmit/receive data can be written to or read from the SIO_n register.

(1) Clocked serial interface mode registers 0, 1 (CSIM0, CSIM1)

The CSIM_n register is an 8-bit register for specifying the operation of CSIn.

(2) Clocked serial interface clock selection registers 0, 1 (CSIC0, CSIC1)

The CSIC_n register is an 8-bit register for controlling the transmit operation of CSIn.

(3) Serial I/O shift registers 0, 1 (SIO0, SIO1)

The SIO_n register is an 8-bit register for converting between serial data and parallel data. SIO_n is used for both transmission and reception.

Data is shifted in (reception) or shifted out (transmission) beginning at either the MSB side or the LSB side.

Actual transmit/receive operations are controlled by reading or writing operations for SIO_n.

(4) Clocked serial interface transmit buffer registers 0, 1 (SOTB0, SOTB1)

The SOTB_n register is an 8-bit buffer register for storing transmit data.

(5) Selector

The selector selects the serial clock to be used.

(6) Serial clock controller

The serial clock controller controls the supply of serial clocks to the shift register. When an internal clock is used, it also controls the clocks that are output to the $\overline{\text{SCKn}}$ pin.

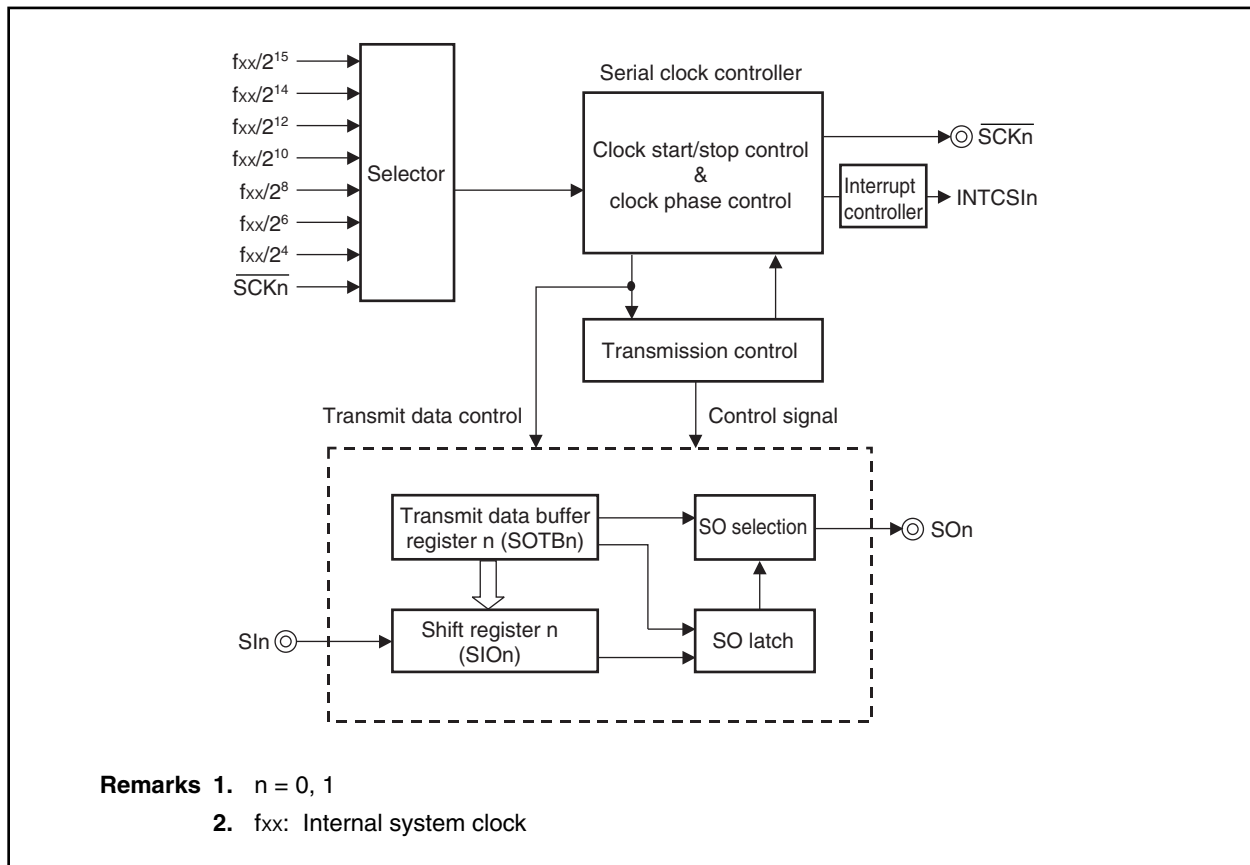
(7) Serial clock counter

The serial clock counter counts serial clocks that are output or input during transmit and receive operations and checks that 8-bit data has been transmitted or received.

(8) Interrupt controller

The interrupt controller controls whether or not an interrupt request is generated when the serial clock counter has counted eight serial clocks.

Figure 11-15. Clocked Serial Interface Block Diagram



11.3.3 Control registers

(1) Clocked serial interface mode registers 0, 1 (CSIM0, CSIM1)

The CSIMn register controls the operation of CSIn ($n = 0, 1$). These registers can be read or written in 8-bit or 1-bit units.

★ Be sure to set bits 5 and 3 to 1 to 0. If they are set to 1, the operation is not guaranteed.

Caution To use CSIn, be sure to set the external pins related to the CSIn function to control mode and set the CSICn register. Then set the CSICAEn bit to 1 before setting the other bits.

	<7>	<6>	5	<4>	3	2	1	<0>	Address	After reset
CSIM0	CSICAE0	TRMD0	0	DIR0	0	0	0	CSOT0	FFFFFF900H	00H
CSIM1	CSICAE1	TRMD1	0	DIR1	0	0	0	CSOT1	FFFFFF910H	00H

Bit Position	Bit Name	Function
7	CSICAE _n (n = 0, 1)	<p>CSI Operation Permission/Prohibition</p> <p>Specifies whether CSIn operation is enabled or disabled (n = 0, 1).</p> <p>0: CSIn operation is disabled (SOn = low level, \overline{SCKn} = high level)</p> <p>1: CSIn operation is enabled</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. If CSICAE_n is set to 0, the CSIn unit can be reset asynchronously. 2. If CSICAE_n = 0, the CSIn unit is in a reset state. Therefore, to operate CSIn, CSICAE_n must be set to 1. 3. If the CSICAE_n bit is changed from 1 to 0, all registers of the CSIn unit are initialized. To set CSICAE_n to 1 again, the registers of the CSIn unit must be set again.
6	TRMD _n (n = 0, 1)	<p>Transmission/Reception Mode Control</p> <p>Specifies the transmission/reception mode.</p> <p>0: Reception-only mode</p> <p>1: Transmission/reception mode</p> <p>If TRMD_n = 0, reception-only transfers are performed. In addition, the SOn pin output is fixed at low level. Data reception is started by reading the SIO_n register.</p> <p>If TRMD_n = 1, transmission/reception is started by writing data to the SOTB_n register.</p> <p>Caution The TRMD_n bit can be overwritten only when CSOT_n = 0.</p>
4	DIR _n (n = 0, 1)	<p>Transmit Direction Mode Control</p> <p>Specifies the transfer direction mode (MSB or LSB).</p> <p>0: The transfer data's start bit is MSB</p> <p>1: The transfer data's start bit is LSB</p> <p>Caution The DIR_n bit can be overwritten only when CSOT_n = 0.</p>
0	CSOT _n (n = 0, 1)	<p>CSI Status of Transmission</p> <p>This is a transfer status display flag.</p> <p>0: Idle status</p> <p>1: Transfer execution status</p> <p>This flag is used to judge whether writing to the shift register (SIO_n) is enabled or not when starting serial data transmission in transmission/reception mode (TRMD_n = 1)</p> <p>Caution The CSOT_n bit is reset when the CSICAE_n bit is cleared (0).</p>

(2) Clocked serial interface clock selection registers 0, 1 (CSIC0, CSIC1)

The CSICn register is an 8-bit register that controls the transmit operation of CSIn.

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

Caution The CSICn register can be overwritten when CSICAE_n = 0 in the CSIMn register.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
CSIC0	0	0	0	CKP0	DAP0	CKS02	CKS01	CKS00	FFFFF901H	00H
CSIC1	0	0	0	CKP1	DAP1	CKS12	CKS11	CKS10	FFFFF911H	00H

Bit Position	Bit Name	Function															
4, 3	CKPn, DAPn (n = 0, 1)	<p>Clock Phase Selection Bit, Data Phase Selection Bit</p> <p>Specifies the data transmission/reception timing for SCKn.</p> <table border="1"> <thead> <tr> <th>CKPn</th> <th>DAPn</th> <th>Operation mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td> </td> </tr> <tr> <td>0</td> <td>1</td> <td> </td> </tr> <tr> <td>1</td> <td>0</td> <td> </td> </tr> <tr> <td>1</td> <td>1</td> <td> </td> </tr> </tbody> </table>	CKPn	DAPn	Operation mode	0	0		0	1		1	0		1	1	
CKPn	DAPn	Operation mode															
0	0																
0	1																
1	0																
1	1																

Bit Position	Bit Name	Function																																													
2 to 0	CKSn2 to CKSn0 (n = 0, 1)	Input Clock Selection Specifies the input clock. <table border="1"> <thead> <tr> <th>CKSn2</th> <th>CKSn1</th> <th>CKSn0</th> <th>Input Clock</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>$f_{xx}/2^{15}$</td> <td>Master mode</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>$f_{xx}/2^{14}$</td> <td>Master mode</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>$f_{xx}/2^{12}$</td> <td>Master mode</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>$f_{xx}/2^{10}$</td> <td>Master mode</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>$f_{xx}/2^8$</td> <td>Master mode</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>$f_{xx}/2^6$</td> <td>Master mode</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>$f_{xx}/2^4$</td> <td>Master mode</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>External clock (SCKn)</td> <td>Slave mode</td> </tr> </tbody> </table> Remark f_{xx} : Internal system clock frequency	CKSn2	CKSn1	CKSn0	Input Clock	Mode	0	0	0	$f_{xx}/2^{15}$	Master mode	0	0	1	$f_{xx}/2^{14}$	Master mode	0	1	0	$f_{xx}/2^{12}$	Master mode	0	1	1	$f_{xx}/2^{10}$	Master mode	1	0	0	$f_{xx}/2^8$	Master mode	1	0	1	$f_{xx}/2^6$	Master mode	1	1	0	$f_{xx}/2^4$	Master mode	1	1	1	External clock (SCKn)	Slave mode
CKSn2	CKSn1	CKSn0	Input Clock	Mode																																											
0	0	0	$f_{xx}/2^{15}$	Master mode																																											
0	0	1	$f_{xx}/2^{14}$	Master mode																																											
0	1	0	$f_{xx}/2^{12}$	Master mode																																											
0	1	1	$f_{xx}/2^{10}$	Master mode																																											
1	0	0	$f_{xx}/2^8$	Master mode																																											
1	0	1	$f_{xx}/2^6$	Master mode																																											
1	1	0	$f_{xx}/2^4$	Master mode																																											
1	1	1	External clock (SCKn)	Slave mode																																											

(a) Baud rate

CKSn2	CKSn1	CKSn0	Baud Rate (bps)			
			40 MHz Operation	33 MHz Operation	25 MHz Operation	20 MHz Operation
0	0	0	1,221	1,007	763	610
0	0	1	2,441	2,014	1,526	1,221
0	1	0	9,766	8,057	6,104	4,883
0	1	1	39,063	32,227	24,414	19,531
1	0	0	156,250	128,906	97,656	78,125
1	0	1	625,000	515,625	390,625	312,500
1	1	0	2,500,000	2,062,500	1,562,500	1,250,000

(3) Serial I/O shift registers 0, 1 (SIO0, SIO1)

The SIO_n register is an 8-bit shift register that converts parallel data to serial data. If TRMD_n = 0 in the CSIM_n register, the transfer is started by reading SIO_n.

Except when a reset is input, the SIO_n register becomes 00H even when the CSICA_n bit of the CSIM_n register is cleared (0).

These registers are read-only in 8-bit units.

Caution SIO_n can be accessed only when the system is set to an idle state (CSOT_n = 0 in the CSIM_n register).

	7	6	5	4	3	2	1	0	Address	After reset
SIO0	SIO07	SIO06	SIO05	SIO04	SIO03	SIO02	SIO01	SIO00	FFFFF902H	00H
SIO1	SIO17	SIO16	SIO15	SIO14	SIO13	SIO12	SIO11	SIO10	FFFFF912H	00H

Bit Position	Bit Name	Function
7 to 0	SIO _n 7 to SIO _n 0 (n = 0, 1)	Serial I/O Shifts data in (reception) or shifts data out (transmission) beginning at the MSB or the LSB side.

(4) Receive-only serial I/O shift registers 0, 1 (SIOE0, SIOE1)

The SIOE_n register is an 8-bit shift register that converts parallel data into serial data. A receive operation does not start even if the SIOE_n register is read while the TRMD bit of the CSIM_n register is 0. Therefore this register is used to read the value of the SIO_n register (receive data) without starting a receive operation.

Except when a reset is input, the SIOE_n register becomes 00H even when the CSICA_n bit of the CSIM_n register is cleared (0).

These registers are read-only in 8-bit units.

Caution SIOE_n can be accessed only when the system is set to an idle state (CSOT_n = 0 in the CSIM_n register).

	7	6	5	4	3	2	1	0	Address	After reset
SIOE0	SIOE07	SIOE06	SIOE05	SIOE04	SIOE03	SIOE02	SIOE01	SIOE00	FFFFF903H	00H
SIOE1	SIOE17	SIOE16	SIOE15	SIOE14	SIOE13	SIOE12	SIOE11	SIOE10	FFFFF913H	00H

Bit Position	Bit Name	Function
7 to 0	SIOE _n 7 to SIOE _n 0 (n = 0, 1)	Serial I/O Shifts data in (reception) beginning at the MSB or the LSB side.

(5) Clocked serial interface transmit buffer registers 0, 1 (SOTB0, SOTB1)

The SOTBn register is an 8-bit buffer register for storing transmit data.

If transmission/reception mode is set (TRMDn = 1 in the CSIMn register), a transmit operation is started by writing data to the SOTBn register.

When a reset is input, the SOTBn register becomes 00H.

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

Caution SOTBn can be accessed only when the system is set to an idle state (CSOTn = 0 in the CSIMn register).

	7	6	5	4	3	2	1	0	Address	After reset
SOTB0	SOTB07	SOTB06	SOTB05	SOTB04	SOTB03	SOTB02	SOTB01	SOTB00	FFFFFF904H	00H
SOTB1	SOTB17	SOTB16	SOTB15	SOTB14	SOTB13	SOTB12	SOTB11	SOTB10	FFFFFF914H	00H

Bit Position	Bit Name	Function
7 to 0	SOTBn7 to SOTBn0 (n = 0, 1)	Serial I/O Writes transmit data.

11.3.4 Operation

(1) Transfer mode

CSIn transmits and receives data in three lines: 1 clock line and 2 data lines.

In reception-only mode (TRMDn = 0 in the CSIMn register), the transfer is started by reading the SIO_n register (n = 0, 1). The SIOEn register must be read when the SIO_n register is read without reception being started.

In transmission/reception mode (TRMDn = 1 in the CSIMn register), the transfer is started by writing data to the SOTBn register.

Once transfer has started, when an 8-bit transfer of CSIn ends, the CSOTn bit of the CSIMn register becomes 0, and transfer stops automatically. Also, when the transfer ends, a transmission/reception completion interrupt (INTCSIn) is generated.

Cautions 1. When CSOTn = 1 in the CSIMn register, the control registers and data registers should not be accessed.

2. If transmit data is written to the SOTBn register and the TRMDn bit of the CSIMn register is changed from 0 to 1, serial transfer is not performed.

(2) Serial clock

(a) When internal clock is selected as the serial clock

If reception or transmission is started, a serial clock is output from the $\overline{\text{SCKn}}$ pin, and the data of the SIn pin is taken into the SIO_n register sequentially or data is output to the SOn pin sequentially from the SIO_n register at the timing when the data has synchronized with the serial clock in accordance with the setting of the CKPn and DAPn bits of the CSICn register (n = 0, 1).

(b) When external clock is selected as the serial clock

If reception or transmission is started, the data of the SIn pin is taken into the SIO_n register sequentially or output to the SOn pin sequentially in synchronization with the serial clock that has been input to the $\overline{\text{SCKn}}$ pin following transmission/reception startup in accordance with the setting of the CKPn and DAPn bits of the CSICn register (n = 0, 1).

If serial clock is input to the $\overline{\text{SCKn}}$ pin when neither reception nor transmission is started, shift operation will not be executed.

Figure 11-16. Transfer Timing

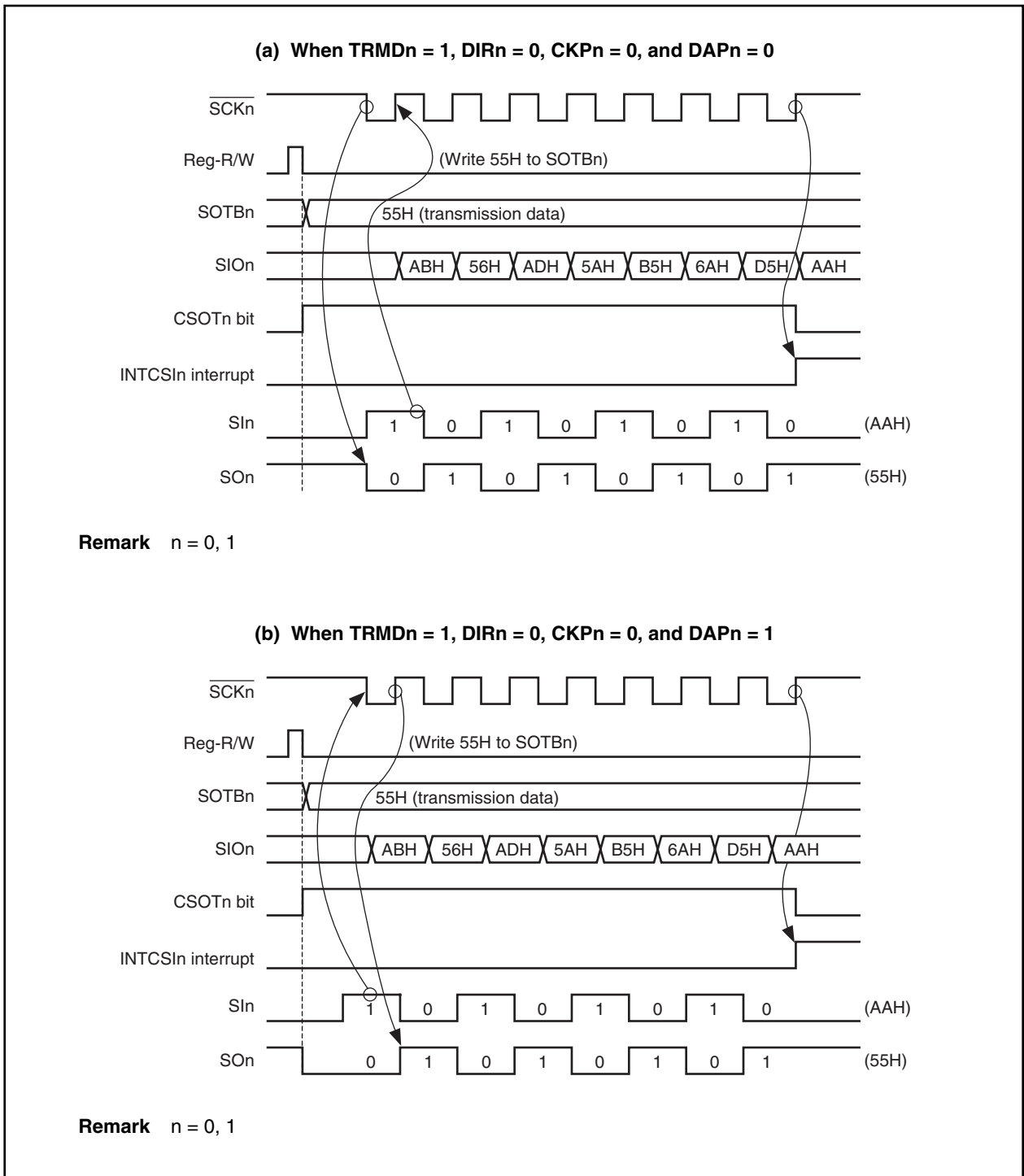
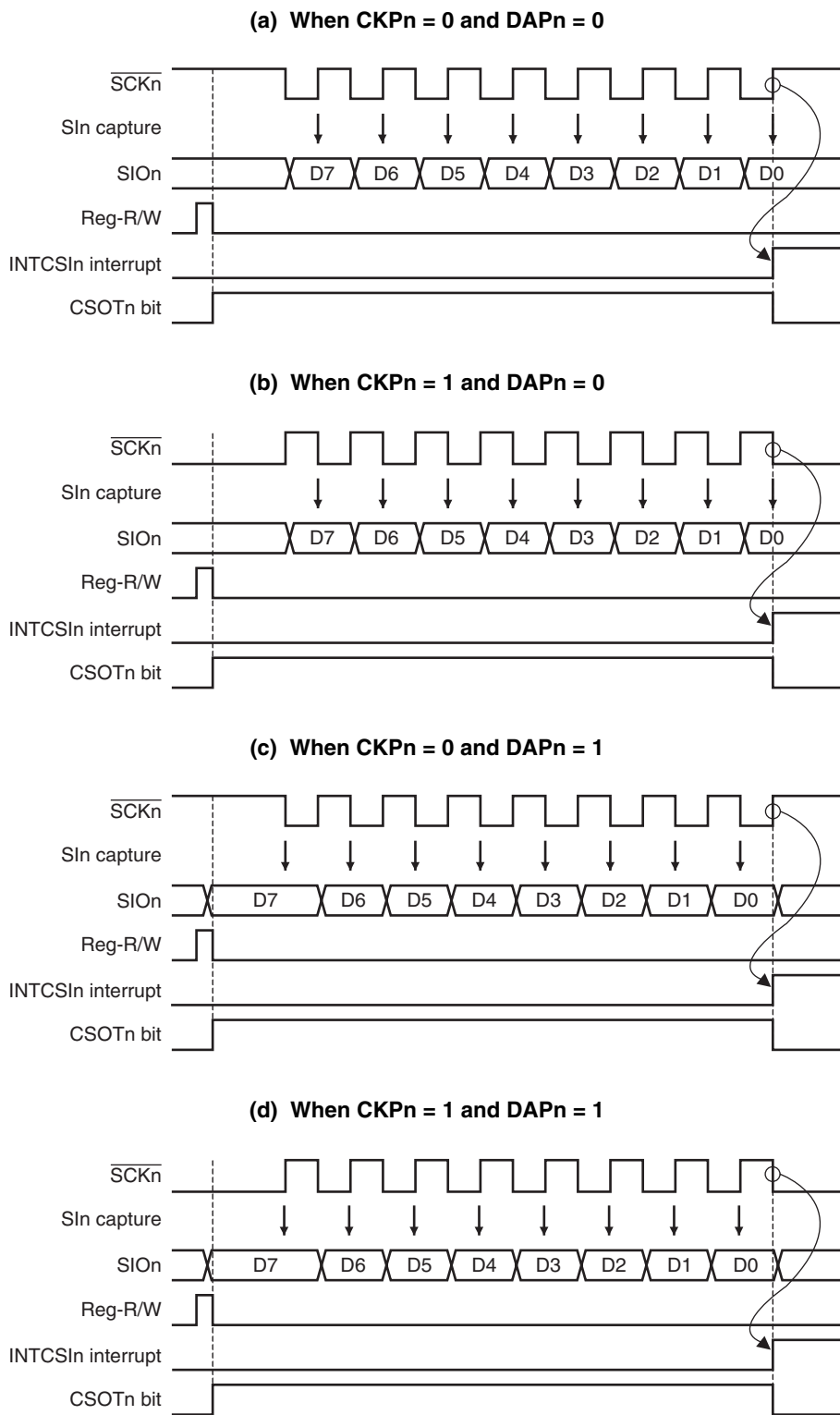


Figure 11-17. Clock Timing



Remark n = 0, 1

11.3.5 Output pins

(1) $\overline{\text{SCKn}}$ pin

When CSIn operation is disabled (CSICAEn = 0), the $\overline{\text{SCKn}}$ pin output state is as follows.

CKPn	$\overline{\text{SCKn}}$ Pin Output
0	Fixed at high level
1	Fixed at low level

- Remarks**
1. When the CKPn bit is overwritten, the $\overline{\text{SCKn}}$ pin output changes.
 2. n = 0, 1

(2) SON pin

When CSIn operation is disabled (CSICAEn = 0), the SON pin output state is as follows.

TRMDn	DAPn	DIRn	SON Pin Output
0	×	×	Fixed at low level
1	0	×	SON latch value (low level)
	1	0	SOTBn7 value
		1	SOTBn0 value

- Remarks**
1. If any of the TRMDn, DAPn, and DIRn bits is overwritten, the SON pin output changes.
 2. n = 0, 1
 3. ×: don't care

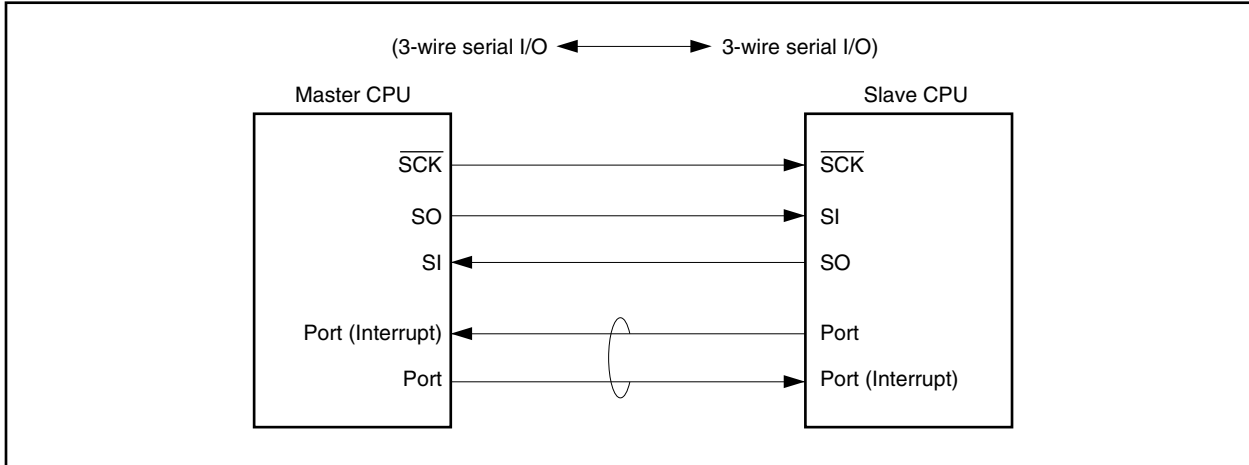
11.3.6 System configuration example

CSIn performs 8-bit length data transfer using three types of signal lines: a serial clock ($\overline{\text{SCKn}}$), serial input (SI_n), and serial output (SO_n). This is effective when connecting peripheral I/O that incorporates a conventional clocked serial interface, or a display controller to the V850E/MA2 ($n = 0, 1$).

When connecting the V850E/MA2 to several devices, lines for handshake are required.

Since the first communication bit can be selected as an MSB or LSB, communication with various devices can be achieved.

Figure 11-18. System Configuration Example of CSI



CHAPTER 12 A/D CONVERTER

12.1 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

12.2 Configuration

The A/D converter of the V850E/MA2 adopts the successive approximation method, and uses A/D converter mode registers 0, 1, 2 (ADM0, ADM1, ADM2), and the A/D conversion result register (ADCR0 to ADCR3) to perform A/D conversion operations.

(1) Input circuit

The input circuit selects the analog input (ANI0 to ANI3) according to the mode set by 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 them to the voltage comparator. This circuit also holds the sampled analog input signal during A/D conversion.

(3) Voltage comparator

The voltage comparator compares the analog input signal with the output voltage of the series resistor string voltage tap.

(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 that sets series resistor string voltage tap data, whose values match analog input voltage values, 1 bit at a time starting from the most significant bit (MSB).

★ If data are 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 result register (ADCRn). When all the specified A/D conversion operations have been completed, an A/D conversion end interrupt (INTAD) occurs.

(6) A/D conversion result register (ADCRn: A/D Conversion Result Register n)

The ADCR is a 10-bit register which 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 sets this register to 0000H.

(7) Controller

The controller selects the analog input, generates the sample and hold circuit operation timing, and controls the conversion trigger according to the mode set by the ADM0 and ADM1 registers.

(8) ANI0 to ANI3 pins

These are 4-channel analog input pins for the A/D converter. They input the analog signals to be A/D converted.

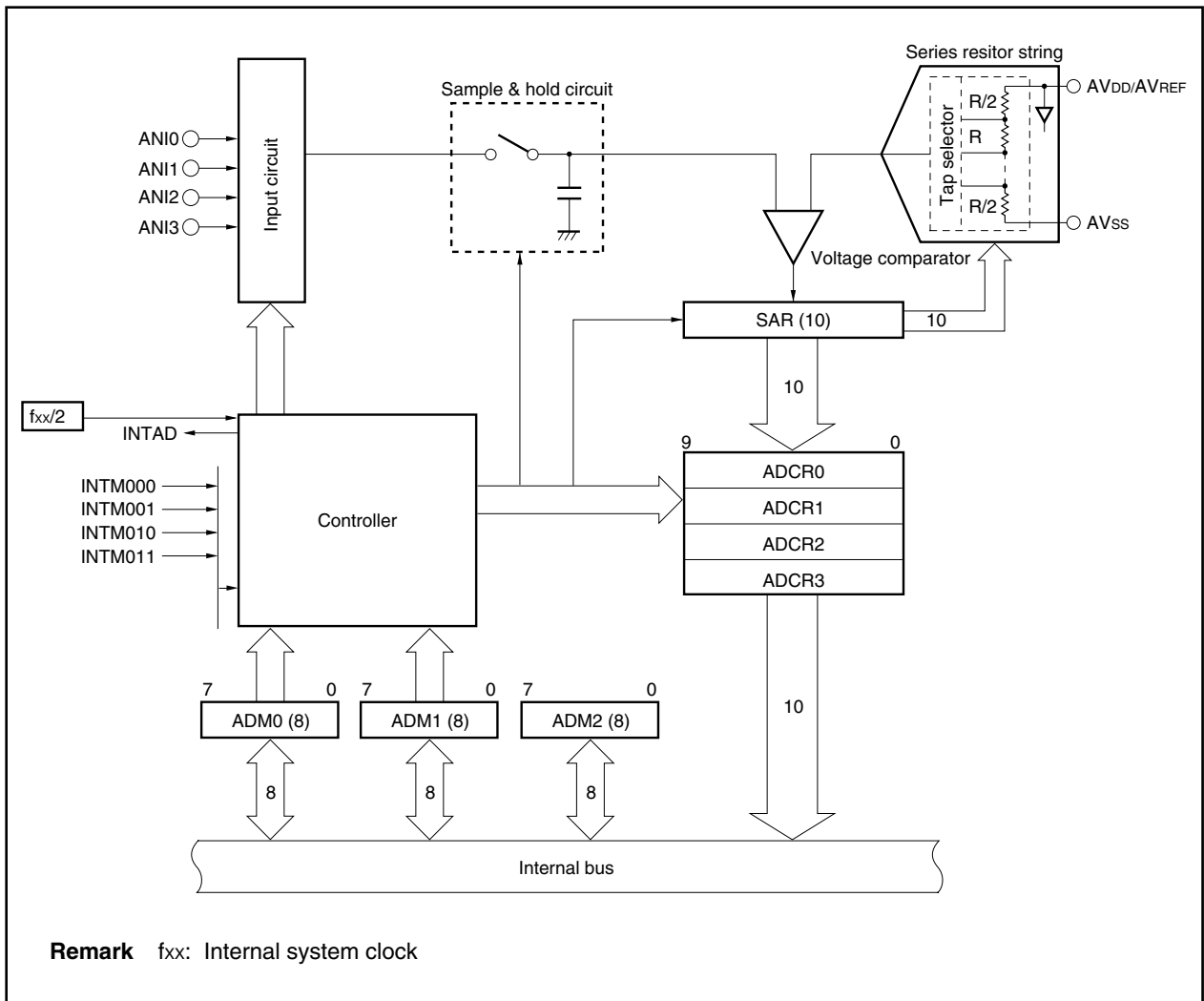
Caution Make sure that the voltages input to ANI0 to ANI3 do not exceed the rated values. If a voltage higher than AV_{DD} or lower than AV_{SS} (even within the range of the absolute maximum ratings) is input to a channel, the conversion value of the channel is undefined, and the conversion values of the other channels may also be affected.

(9) AV_{REF} pin

This is the pin for inputting the reference voltage of the A/D converter. It converts signals input to the ANIn pin to digital signals based on the voltage applied between AV_{REF} and AV_{SS} .

In the V850E/MA2, the AV_{REF} pin functions alternately as the AV_{DD} pin. It is therefore impossible to set voltage separately for the AV_{REF} pin and the AV_{DD} pin.

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



Cautions 1. If there is noise at the analog input pins (ANI0 to ANI3) or at the reference voltage input pin (AV_{REF}), that noise may generate an illegal conversion result.

Software processing will be needed to avoid a negative effect on the system from this illegal conversion result.

An example of this software processing is shown below.

- Take the average result of a number of A/D conversions and use that as the A/D conversion result.
- Execute a number of A/D conversions consecutively and use those results, omitting any exceptional results that may have been obtained.
- If an A/D conversion result that is judged to have generated a system malfunction is obtained, be sure to recheck the system malfunction before performing malfunction processing.

2. Do not apply a voltage outside the AV_{SS} to AV_{REF} range to the pins that are used as A/D converter input pins.

12.3 Control Registers

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

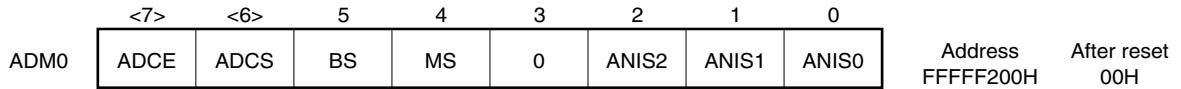
The ADM0 register is an 8-bit register which executes the selection of the analog input pin, specification of operation mode, and conversion operations.

This register can be read/written in 8- or 1-bit units. However, when the data is written to the ADM0 register during an A/D conversion operation, the conversion operation is initialized and conversion is executed from the beginning. Bit 6 cannot be written to and writing executed is ignored.

Cautions 1. When the ADCE bit is 1 in the timer trigger mode, the trigger signal standby state is set. To clear the ADCE bit, write “0” or reset.

In the A/D trigger mode, the conversion trigger is set by writing 1 to the ADCE bit. After the operation, when the mode is changed to the timer trigger mode without clearing the ADCE bit, the trigger input standby state is set immediately after the register is changed.

- 2. There are 10 clocks between the beginning of conversion and when the ADCS bit becomes 1.**



Bit Position	Bit Name	Function																																														
7	ADCE	Convert Enable Enables or disables A/D conversion operation. 0: Disabled 1: Enabled																																														
6	ADCS	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																																														
2 to 0	ANIS2 to ANIS0	Analog Input Select Specifies analog input pin to A/D convert. <table border="1" style="margin: 10px auto; border-collapse: collapse; width: 80%;"> <thead> <tr> <th rowspan="2" style="width: 10%;">ANIS2</th> <th rowspan="2" style="width: 10%;">ANIS1</th> <th rowspan="2" style="width: 10%;">ANIS0</th> <th colspan="2" style="width: 30%;">Select mode</th> <th colspan="2" style="width: 30%;">Scan mode</th> </tr> <tr> <th style="width: 15%;">A/D trigger mode</th> <th style="width: 15%;">Timer trigger mode</th> <th style="width: 15%;">A/D trigger mode</th> <th style="width: 15%;">Timer trigger mode^{Note}</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">ANI0</td> <td style="text-align: center;">ANI0</td> <td style="text-align: center;">ANI0</td> <td style="text-align: center;">1</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">ANI1</td> <td style="text-align: center;">ANI1</td> <td style="text-align: center;">ANI0, ANI1</td> <td style="text-align: center;">2</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">ANI2</td> <td style="text-align: center;">ANI2</td> <td style="text-align: center;">ANI0 to ANI2</td> <td style="text-align: center;">3</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">ANI3</td> <td style="text-align: center;">ANI3</td> <td style="text-align: center;">ANI0 to ANI3</td> <td style="text-align: center;">4</td> </tr> <tr> <td colspan="3" style="text-align: center;">Other than above</td> <td colspan="2" style="text-align: center;">Setting prohibited</td> <td colspan="2" style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table>	ANIS2	ANIS1	ANIS0	Select mode		Scan mode		A/D trigger mode	Timer trigger mode	A/D trigger mode	Timer trigger mode ^{Note}	0	0	0	ANI0	ANI0	ANI0	1	0	0	1	ANI1	ANI1	ANI0, ANI1	2	0	1	0	ANI2	ANI2	ANI0 to ANI2	3	0	1	1	ANI3	ANI3	ANI0 to ANI3	4	Other than above			Setting prohibited		Setting prohibited	
ANIS2	ANIS1	ANIS0				Select mode		Scan mode																																								
			A/D trigger mode	Timer trigger mode	A/D trigger mode	Timer trigger mode ^{Note}																																										
0	0	0	ANI0	ANI0	ANI0	1																																										
0	0	1	ANI1	ANI1	ANI0, ANI1	2																																										
0	1	0	ANI2	ANI2	ANI0 to ANI2	3																																										
0	1	1	ANI3	ANI3	ANI0 to ANI3	4																																										
Other than above			Setting prohibited		Setting prohibited																																											

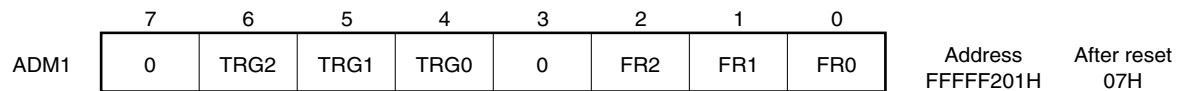
Note In the timer trigger mode (4-trigger mode) during the scan mode, because the scanning sequence of the ANI0 to ANI3 pins is specified by the sequence in which the match signals are generated from the compare register, the number of trigger inputs should be specified instead of specifying a certain analog input pin.

(2) A/D converter mode register 1 (ADM1)

The ADM1 register is an 8-bit register which specifies the conversion operation time and trigger mode.

This register can be read/written in 8-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.

- ★ **Cautions 1. It takes the following number of clocks from trigger input to completion of A/D conversion, in addition to the clocks specified using the FR2 to FR0 bits. (Refer to 12.7.6 Supplementary information on A/D conversion time.)**
In A/D trigger mode: 11 to 13 clocks (9 to 11 clocks + 2 clocks)
In timer trigger mode: 7 to 9 clocks (5 to 7 clocks + 2 clocks)
- 2. In the timer trigger mode, be sure to input the trigger at an interval longer than the number of clocks specified using the FR2 to FR0 bits. (Refer to 12.7.2 Timer trigger interval.)**



Bit Position	Bit Name	Function																																																								
6 to 4	TRG2 to TRG0	Trigger Mode Specifies trigger mode. <table border="1" style="border-collapse: collapse; margin-top: 10px; width: 100%;"> <thead> <tr> <th style="width: 10%;">TRG2</th> <th style="width: 10%;">TRG1</th> <th style="width: 10%;">TRG0</th> <th style="width: 70%;">Trigger mode</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0/1</td> <td>A/D trigger mode</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Timer trigger mode (1-trigger mode)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Timer trigger mode (4-trigger mode)</td> </tr> <tr> <td colspan="3" style="text-align: center;">Other than above</td> <td style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table>	TRG2	TRG1	TRG0	Trigger mode	0	0	0/1	A/D trigger mode	0	1	0	Timer trigger mode (1-trigger mode)	0	1	1	Timer trigger mode (4-trigger mode)	Other than above			Setting prohibited																																				
TRG2	TRG1	TRG0	Trigger mode																																																							
0	0	0/1	A/D trigger mode																																																							
0	1	0	Timer trigger mode (1-trigger mode)																																																							
0	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 conversion time to be same value irrespective of oscillation frequency. <table border="1" style="border-collapse: collapse; margin-top: 10px; width: 100%;"> <thead> <tr> <th rowspan="2" style="width: 8%;">FR2</th> <th rowspan="2" style="width: 8%;">FR1</th> <th rowspan="2" style="width: 8%;">FR0</th> <th rowspan="2" style="width: 15%;">Number of conversion clock</th> <th colspan="2" style="width: 41%;">Conversion operation time^{Note}</th> </tr> <tr> <th style="width: 15%;">f_{xx} = 40 MHz</th> <th style="width: 15%;">f_{xx} = 33 MHz</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">96</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">144</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">192</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">5.82 μs</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">240</td> <td style="text-align: center;">6.00 μs</td> <td style="text-align: center;">7.27 μs</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">336</td> <td style="text-align: center;">8.40 μs</td> <td style="text-align: center;">10.18 μs</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">384</td> <td style="text-align: center;">9.60 μs</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">480</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">Setting prohibited</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">672</td> <td style="text-align: center;">Setting prohibited</td> <td style="text-align: center;">Setting prohibited</td> </tr> </tbody> </table> <p>Note Figures in the conversion operation time are target values. Set the conversion operation time to 5 to 10 μs.</p> <p>Remark f_{xx} = internal system clock</p>	FR2	FR1	FR0	Number of conversion clock	Conversion operation time ^{Note}		f _{xx} = 40 MHz	f _{xx} = 33 MHz	0	0	0	96	Setting prohibited	Setting prohibited	0	0	1	144	Setting prohibited	Setting prohibited	0	1	0	192	Setting prohibited	5.82 μs	0	1	1	240	6.00 μs	7.27 μs	1	0	0	336	8.40 μs	10.18 μs	1	0	1	384	9.60 μs	Setting prohibited	1	1	0	480	Setting prohibited	Setting prohibited	1	1	1	672	Setting prohibited	Setting prohibited
FR2	FR1	FR0					Number of conversion clock	Conversion operation time ^{Note}																																																		
			f _{xx} = 40 MHz	f _{xx} = 33 MHz																																																						
0	0	0	96	Setting prohibited	Setting prohibited																																																					
0	0	1	144	Setting prohibited	Setting prohibited																																																					
0	1	0	192	Setting prohibited	5.82 μs																																																					
0	1	1	240	6.00 μs	7.27 μs																																																					
1	0	0	336	8.40 μs	10.18 μs																																																					
1	0	1	384	9.60 μs	Setting prohibited																																																					
1	1	0	480	Setting prohibited	Setting prohibited																																																					
1	1	1	672	Setting prohibited	Setting prohibited																																																					

(3) A/D converter mode register 2 (ADM2)

The ADM2 register is an 8-bit register that controls the reset and clock of the A/D converter.

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

Caution Because ADCAE = 0 after reset release, the A/D converter enters the reset state. When operating the A/D converter, be sure to write to the ADM0 and ADM1 registers after setting the ADCAE bit of the ADM2 register to 1 (it is impossible to write to the ADM0 and ADM1 registers when ADCAE = 0). Moreover, when the ADCAE bit is set to 0, all registers related to the A/D converter are initialized.

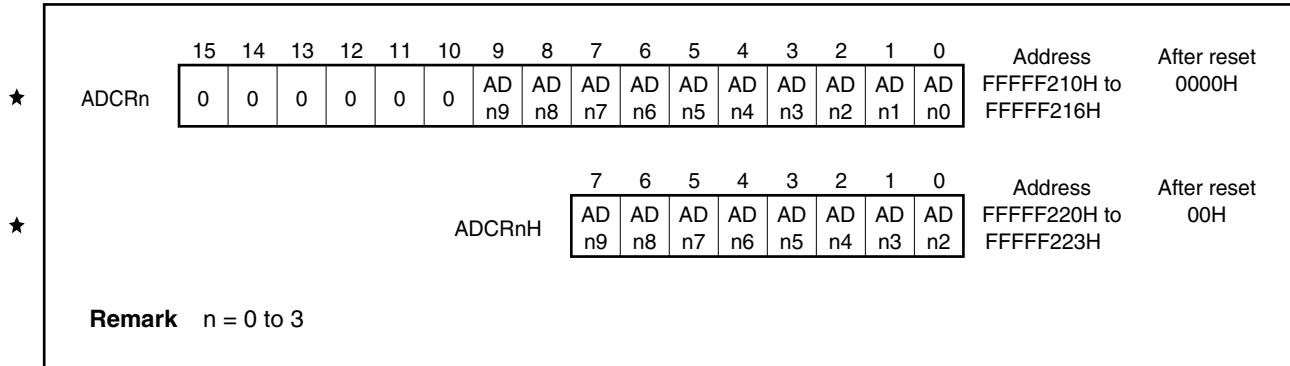
	7	6	5	4	3	2	1	<0>		
ADM2	0	0	0	0	0	0	0	ADCAE	Address FFFFFF202H	After reset 00H

Bit Position	Bit Name	Function
0	ADCAE	Clock Action Enable Controls A/D converter operation. 0: Clock supply to the A/D converter is stopped, the A/D converter is in the reset state 1: The clock is supplied to the A/D converter, A/D converter operation is enabled

(4) 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. There are four 10-bit registers. These registers are read-only in 16- or 8-bit units. During the 16-bit access, the ADCRn register is specified, and during higher 8-bit access, the ADCRnH register is specified (n = 0 to 3).

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

The relationship between the analog voltage input to the analog input pins (ANI0 to ANI3) and the A/D conversion result (of the A/D conversion result register (ADCRn)) is as follows:

$$ADCR = \text{INT} \left(\frac{V_{IN}}{AV_{REF}} \times 1,024 + 0.5 \right)$$

Or,

$$(ADCR - 0.5) \times \frac{AV_{REF}}{1,024} \leq V_{IN} < (ADCR + 0.5) \times \frac{AV_{REF}}{1,024}$$

INT(): Function that returns the integer of the value in ()

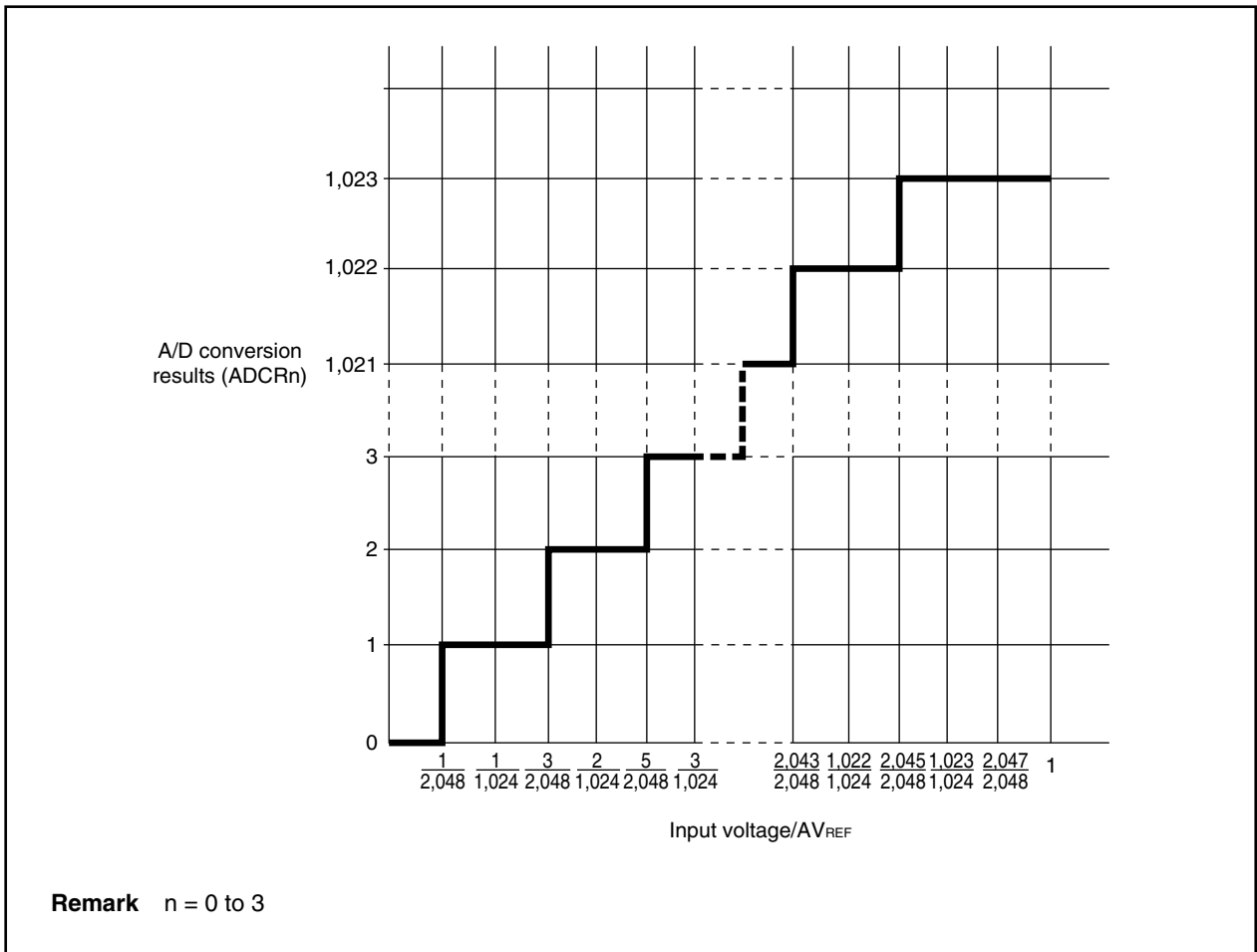
V_{IN}: Analog input voltage

AV_{REF}: AV_{REF} pin voltage

ADCR: Value of A/D conversion result register (ADCRn)

Figure 12-2 shows the relationship between the analog input voltage and the A/D conversion results.

Figure 12-2. Relationship Between Analog Input Voltage and A/D Conversion Results



12.4 A/D Converter Operation

12.4.1 Basic operation of A/D converter

A/D conversion is executed in the following order.

- (1) The ADCAE bit of the ADM2 register is set (1).
- (2) The selection of the analog input and specification of the operation mode, trigger mode, etc. should be specified using the ADM0 and ADM1 registers^{Note 1}.
When the ADCE bit of the ADM0 register is set (1), A/D conversion starts in A/D trigger mode. In the timer trigger mode, the trigger standby state^{Note 2} is set.
- (3) The voltage generated from the voltage tap of the series resistor string and analog input are compared by the comparator.
- (4) When the comparison of the 10 bits ends, the conversion results are stored in the ADCRn register. When A/D conversion has been 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 to ADM2 registers are changed during the 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. During the timer trigger mode, if the ADCE 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.

12.4.2 Operation mode and trigger mode

Various conversion operations can be specified for the A/D converter 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	ADM1	
A/D trigger		Select	1 buffer	xx010xxxB	000x0xxxB	ANI0 to ANI3
			4 buffers	xx110xxxB	000x0xxxB	
		Scan		xxx00xxxB	000x0xxxB	
Timer trigger	1 trigger	Select	1 buffer	xx010xxxB	00100xxxB	
			4 buffers	xx110xxxB	00100xxxB	
		Scan		xxx00xxxB	00100xxxB	
	4 trigger	Select	1 buffer	xx010xxxB	00110xxxB	
			4 buffers	xx110xxxB	00110xxxB	
		Scan		xxx00xxxB	00110xxxB	

(1) Trigger mode

There are two types of trigger modes which serve as the start timing of 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

This mode starts the conversion timing of the analog input set to pins ANI0 to ANI3, and by setting the ADCE bit of the ADM0 register to 1, starts A/D conversion.

(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 timer C compare register.

This register creates the analog input conversion timing by generating the match interrupts (INTM000, INTM001, INTM010, INTM011) of the four capture/compare registers (CCC00, CCC01, CCC10, CCC11) connected to the 16-bit timer C (TMC0, TMC1). Moreover, because the match interrupts (INTM000, INTM001, INTM010, INTM011) are also used as external pin interrupts (INTP000, INTP001, INTP010, INTP011), the analog input conversion timing is generated even when external pin interrupts are input.

There are two types of sub-trigger modes: 1-trigger mode and 4-trigger mode.

- **1-trigger mode**

A mode which uses one match interrupt from timer C as the A/D conversion start timing.

- **4-trigger mode**

A mode which uses four match interrupts from timer C as the A/D conversion start timing.

★

(2) Operation mode

There are two types of operation modes which set the ANI0 to ANI3 pins: select mode and scan mode. The select mode has a sub-mode that consists of 1-buffer mode and 4-buffer mode. These modes are set by the ADM0 register.

(a) Select mode

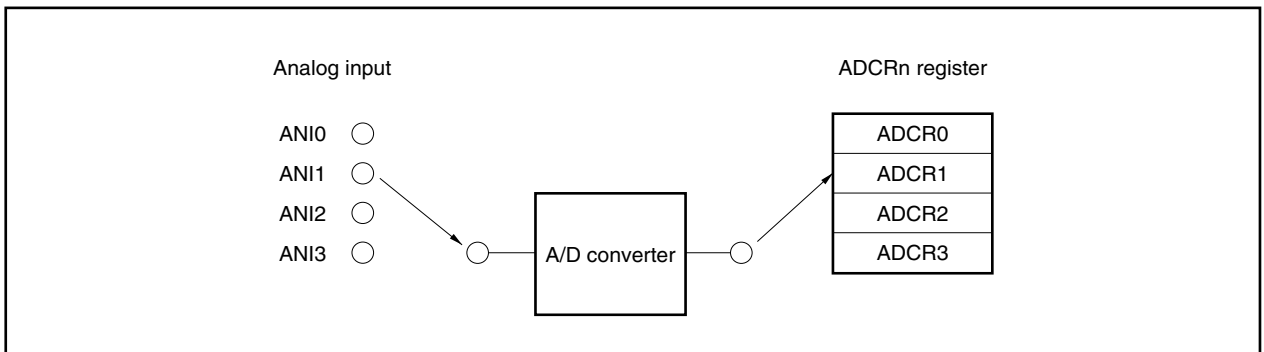
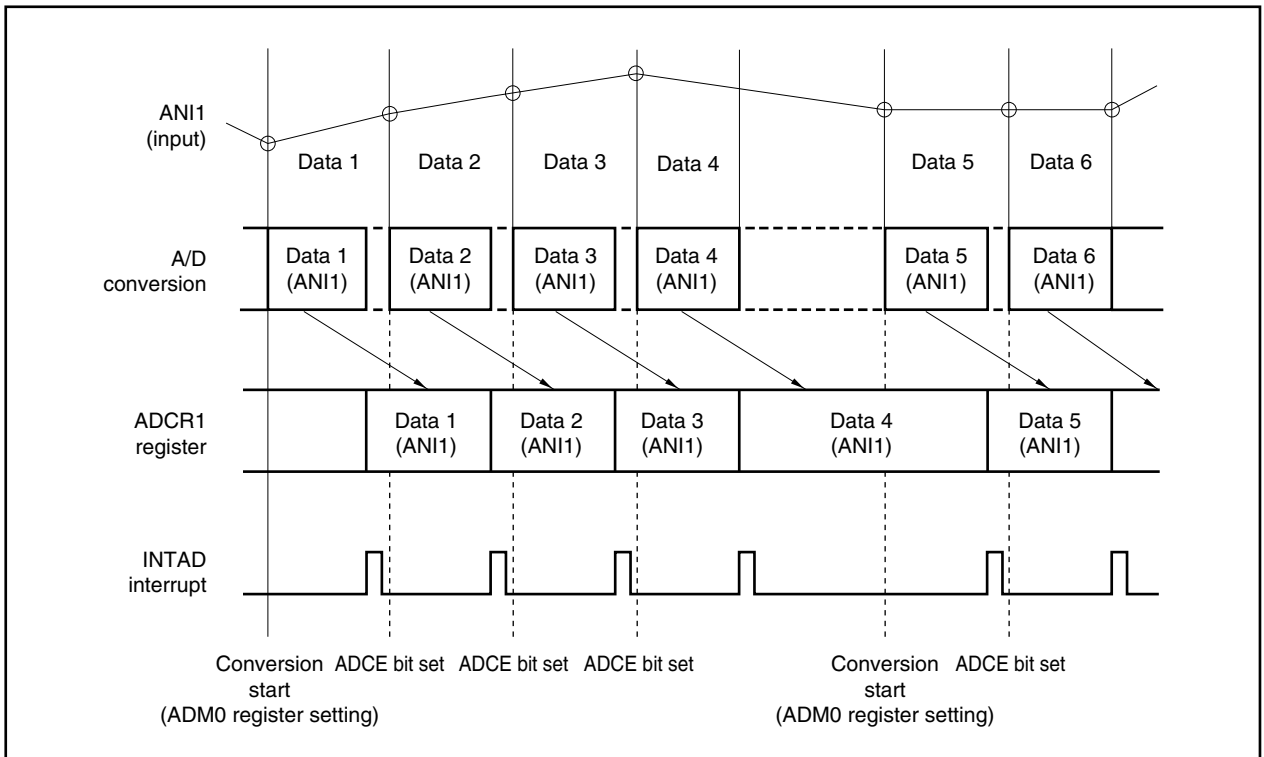
In this mode, one analog input specified by the ADM0 register is A/D converted. The conversion results are stored in the ADCRn register corresponding to the analog input (ANIn). For this mode, the 1-buffer mode and 4-buffer mode are provided for storing the A/D conversion results (n = 0 to 3).

- **1-buffer mode**

In this 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 register correspond one to one, and an A/D conversion end interrupt (INTAD) is generated each time one A/D conversion ends.

★

Figure 12-3. Select Mode Operation Timing: 1-Buffer Mode (ANI1)

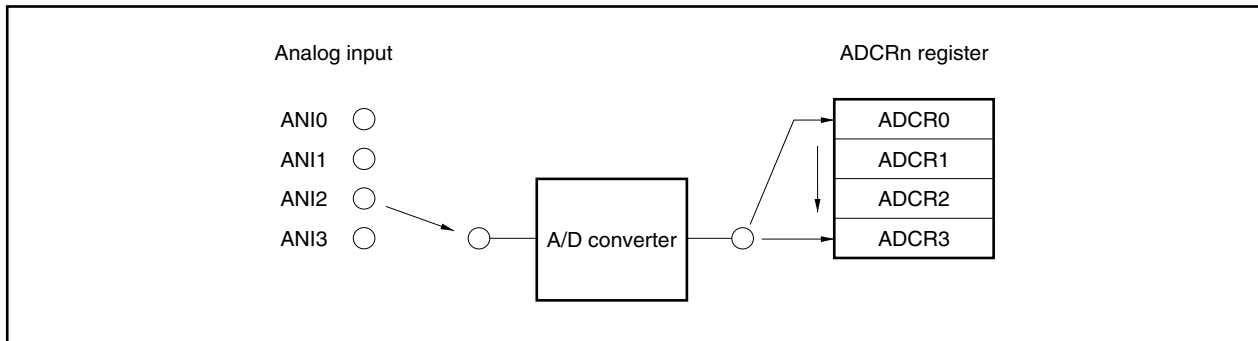
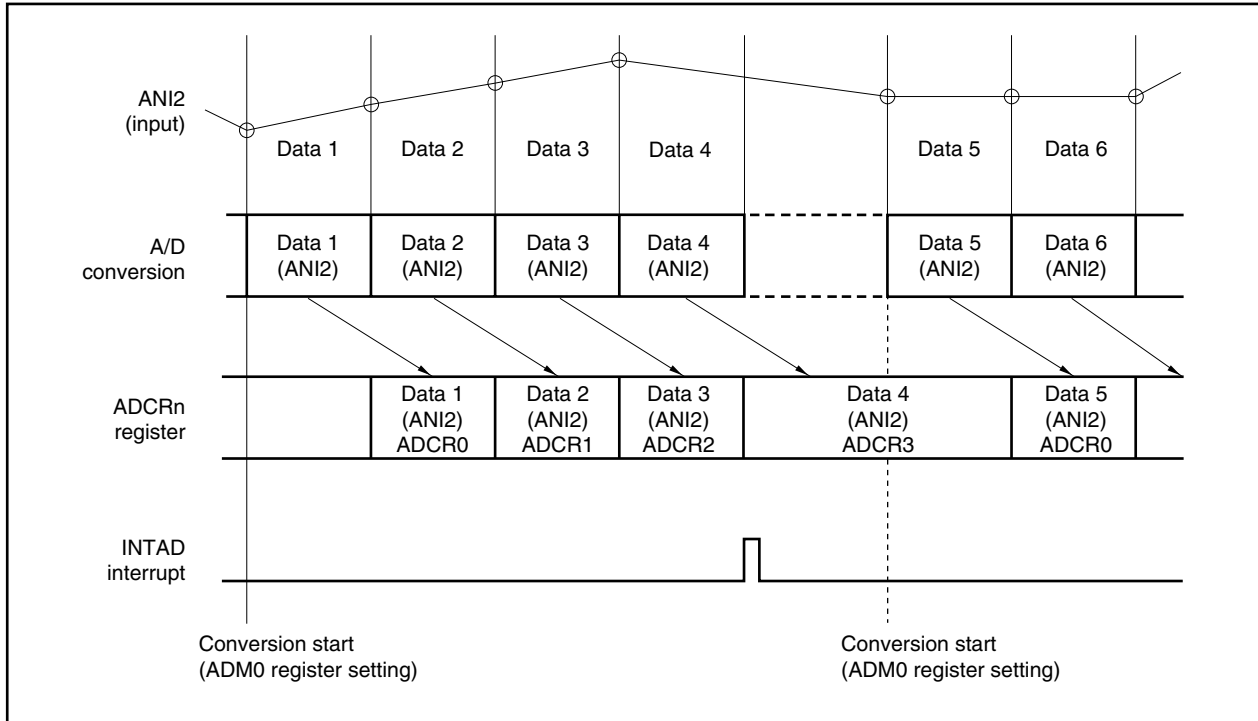


- **4-buffer mode**

In this 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 12-4. Select Mode Operation Timing: 4-Buffer Mode (ANI2)

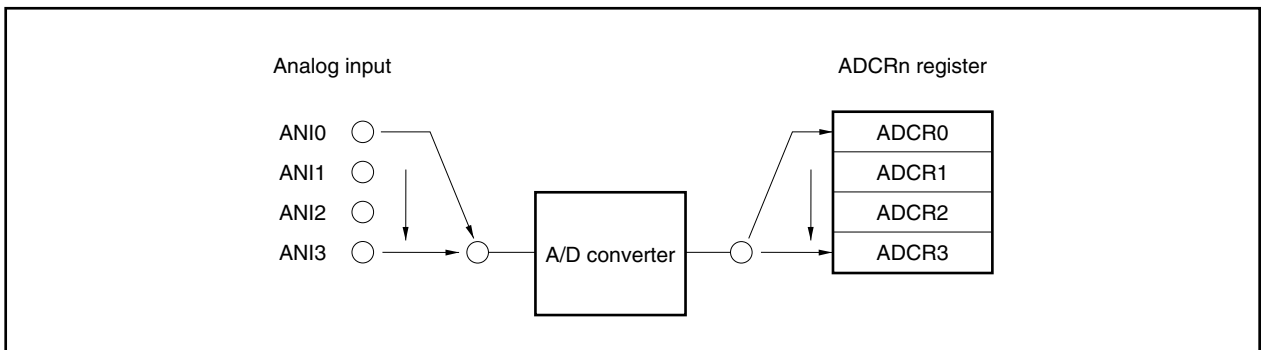
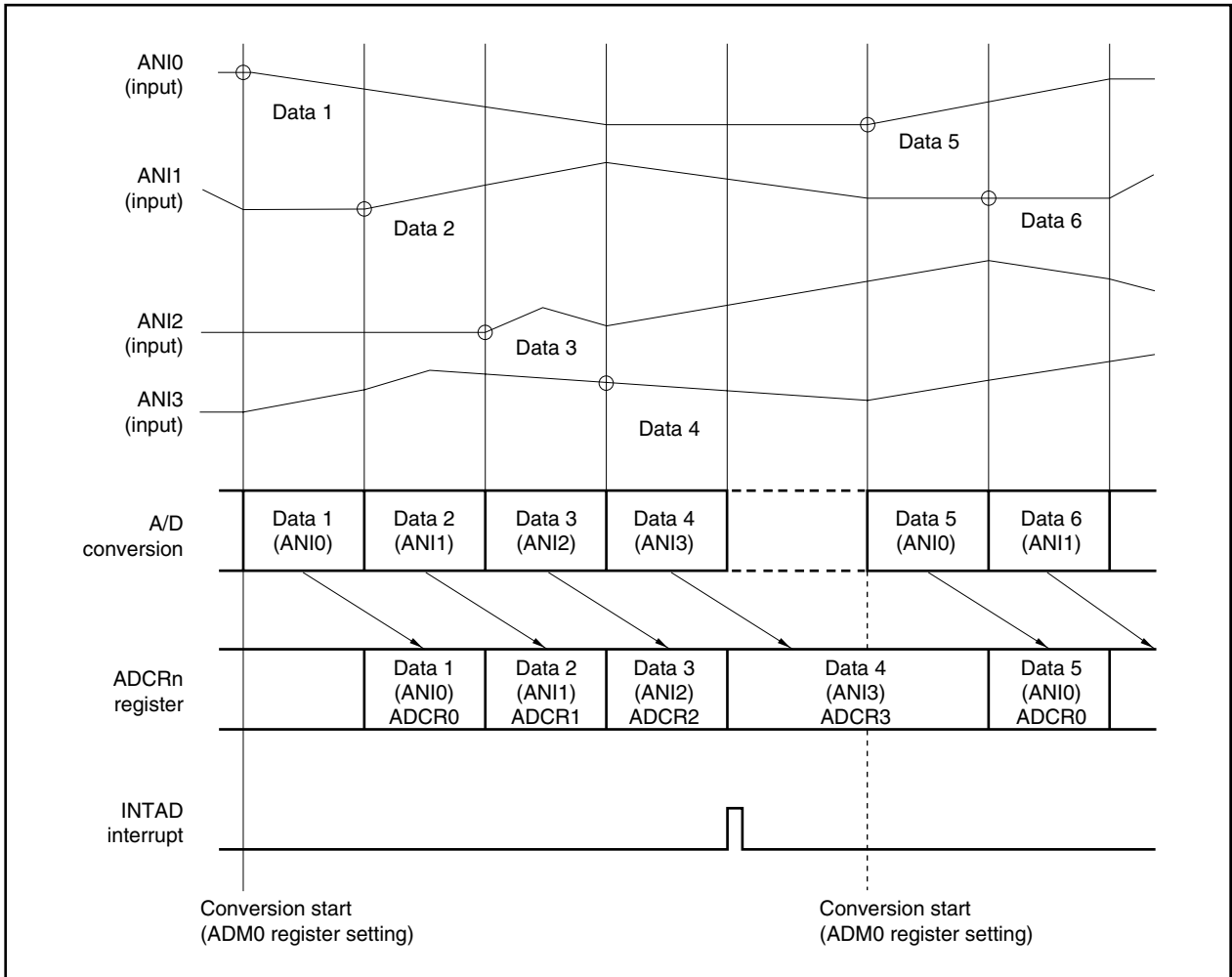


(b) Scan mode

In this mode, 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 the specified analog input ends, the A/D conversion end interrupt (INTAD) is generated.

★

Figure 12-5. Scan Mode Operation Timing: 4-Channel Scan (ANI0 to ANI3)



12.5 Operation in A/D Trigger Mode

When the ADCE bit of the ADM0 register is set to 1, A/D conversion is started.

12.5.1 Select mode operation

In this mode, 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. In 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)

In this mode, 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.

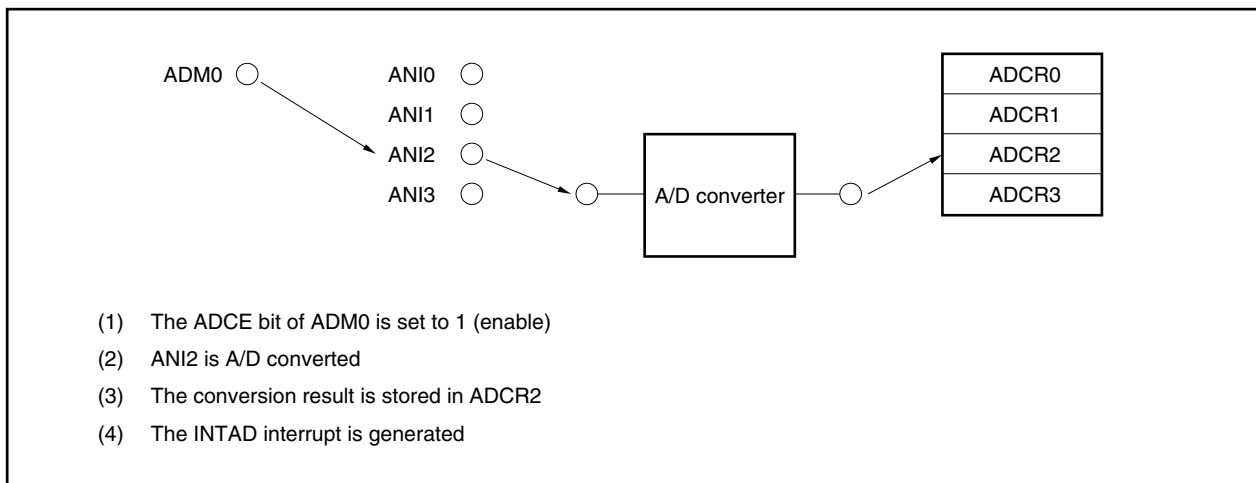
Each time an A/D conversion is executed, an A/D conversion end interrupt (INTAD) is generated and the AD conversion completes.

Analog Input	A/D Conversion Result Register
ANIn	ADCRn

If 1 is written in the ADCE bit of the ADM0 register, A/D conversion can be restarted.

This mode is most appropriate for applications in which the results of each first time A/D conversion are read.

Figure 12-6. Example of 1-Buffer Mode (A/D Trigger Select 1-Buffer) Operation



(2) 4-buffer mode (A/D trigger select: 4-buffer)

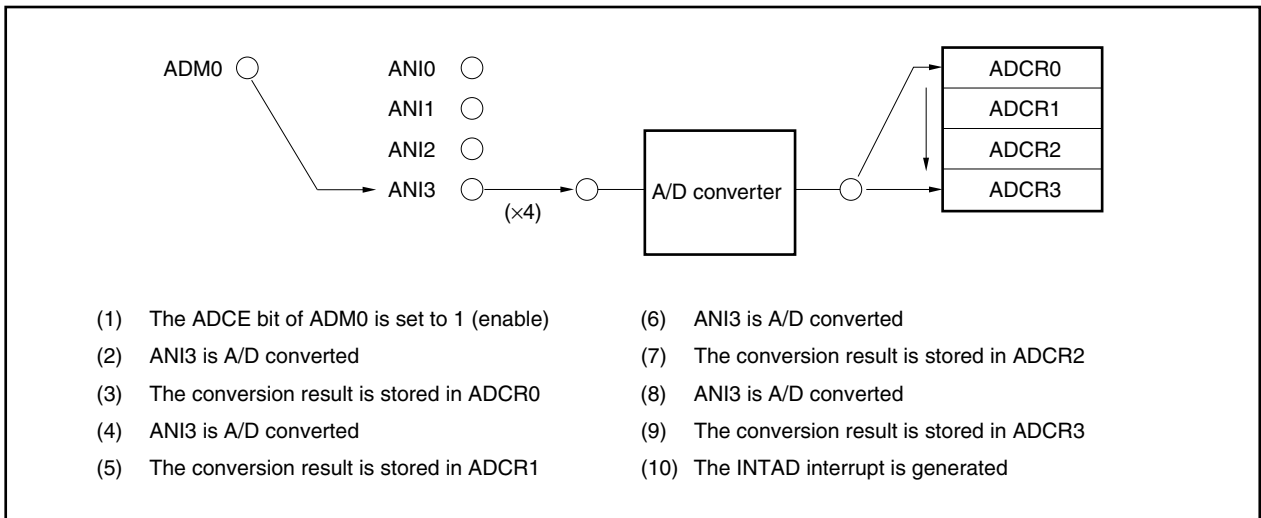
In this mode, one analog input is A/D converted four times and the results are stored in the ADCR0 to ADCR3 registers. When the 4th A/D conversion ends, an A/D conversion end interrupt (INTAD) is generated and the A/D conversion is stopped.

Analog Input	A/D Conversion Result Register
ANIn	ADCR0
ANIn	ADCR1
ANIn	ADCR2
ANIn	ADCR3

If 1 is written in the ADCE bit of the ADM0 register, A/D conversion can be restarted.

This mode is suitable for applications in which the average of A/D conversion result is calculated.

Figure 12-7. Example of 4-Buffer Mode (A/D Trigger Select 4-Buffer) Operation



12.5.2 Scan mode operations

In this mode, 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 A/D conversion end interrupt (INTAD) is generated, and A/D conversion is stopped.

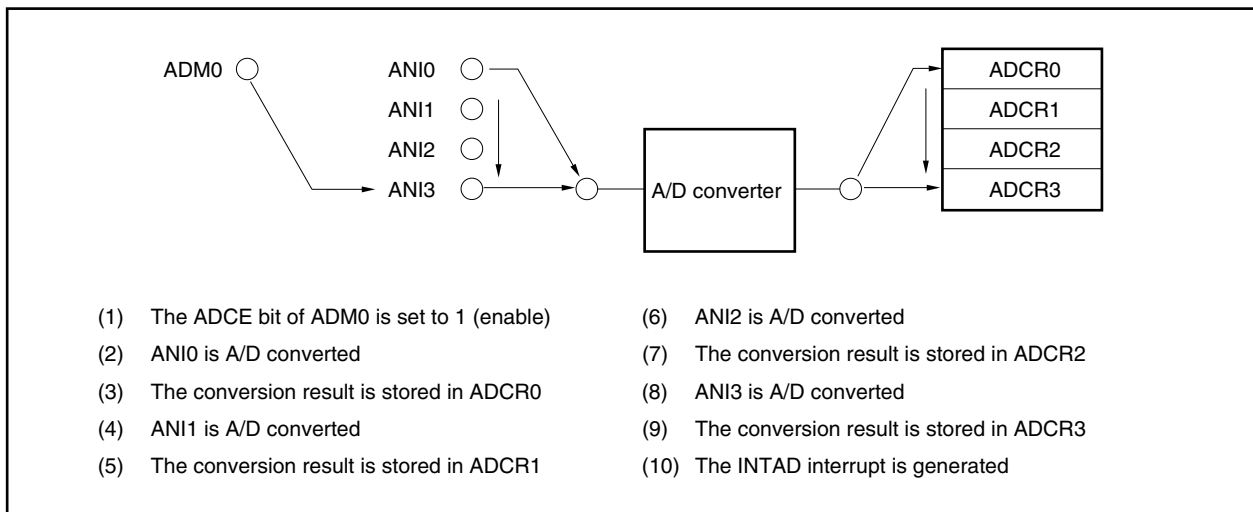
Analog Input	A/D Conversion Result Register
ANI0	ADCR0
⋮	⋮
ANI ⁿ _{Note}	ADCR _n

Note Set by the ANI0 to ANI2 bits of the ADM0 register.

If 1 is written in the ADCE bit of the ADM0 register, A/D conversion can be restarted.

This mode is most appropriate for applications in which multiple analog inputs are constantly monitored.

Figure 12-8. Example of Scan Mode (A/D Trigger Scan) Operation



12.6 Operation in Timer Trigger Mode

Conversion timings for up to four-channel analog inputs (ANI0 to ANI3) can be set for the A/D converter using the interrupt signal output from the TMC compare register.

Two 16-bit timers (TMC0, TMC1) and four capture/compare registers (CCC00, CCC01, CCC10, CC11) are used for the timer to specify the analog conversion trigger.

The following two modes are provided according to the value set in the TMCC01 or TMCC11 register.

(1) 1-shot mode

To use the 1-shot mode, set the OSTn bit of the TMCCn1 register (overflow stop mode) to 1 (n = 0, 1).

When TMC overflows, 0000H is held, and counter operation stops. Thereafter, TMCn does not output the match interrupt signal (A/D conversion trigger) of the compare register, and the A/D converter enters the A/D conversion standby state. The TMCn count operation restarts when the TMCCEn bit of the TMCCn0 register is set to 1. The 1-shot mode is used when the A/D conversion cycle is longer than the TMC cycle. (n = 0, 1).

(2) Loop mode

To use the loop mode, set the OST bit of the TMCCn1 register to 0 (free running mode) (n = 0, 1).

When TMCn overflows, the TMCn 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.

12.6.1 Select mode operation

In this mode, an 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. In 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 operation (timer trigger select: 1-buffer)

In this mode, one analog input is A/D converted once and the conversion results are stored in one ADCRn register.

There are two modes in the 1-buffer mode: 1-trigger mode and 4-trigger mode, according to the number of triggers.

(a) 1-trigger mode (timer trigger select: 1 buffer, 1 trigger)

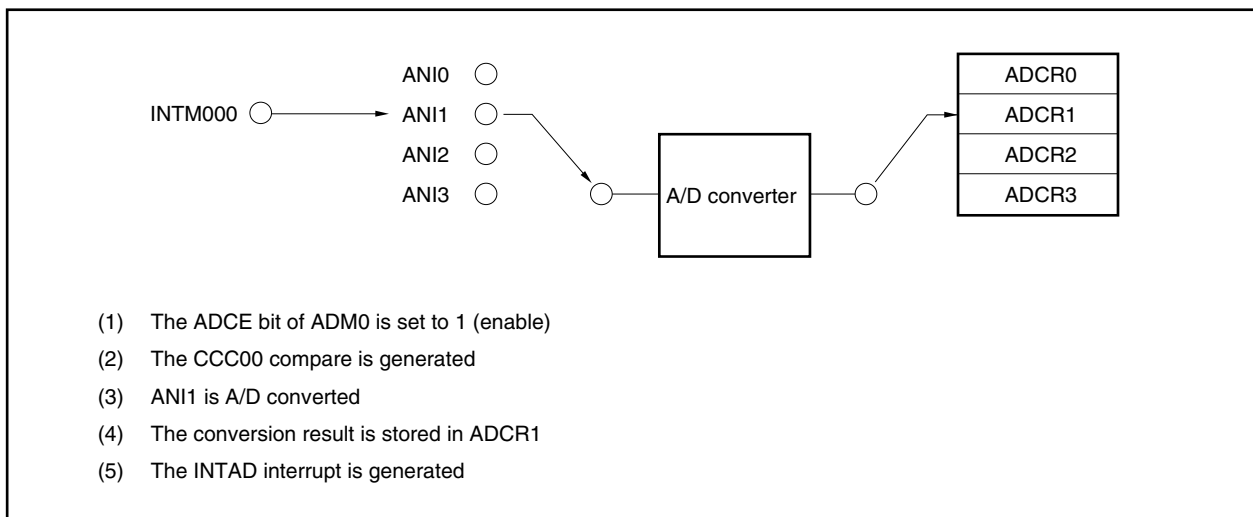
In this mode, one analog input is A/D converted once using the trigger of the match interrupt signal (INTM000) and the results are stored in one ADCRn register. An A/D conversion end interrupt (INTAD) is generated for each A/D conversion and A/D conversion (n = 0 to 3) is stopped.

Trigger	Analog Input	A/D Conversion Result Register
INTM000 interrupt	ANIn	ADCRn

In 1-shot mode, A/D conversion stops after one conversion. To restart A/D conversion, set the TMCCE0 bit of the TMCC00 register to 1.

When set to the loop mode, unless the ADCE bit of the ADM0 register is set to 0, A/D conversion is repeated each time a match interrupt is generated.

Figure 12-9. Example of 1-Trigger Mode (Timer Trigger Select 1-Buffer 1-Trigger) Operation



(b) 4-trigger mode (timer trigger select: 1-buffer, 4-trigger)

In this mode, one analog input is A/D converted using four match interrupt signals (INTM000, INTM001, INTM010, INTM011) as triggers and the results are stored in one ADCRn register. The A/D conversion end interrupt (INTAD) is generated with each A/D conversion, and the ADCE bit of the ADM0 register is reset (0). The results of one A/D conversion are held in 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 (n = 0 to 3).

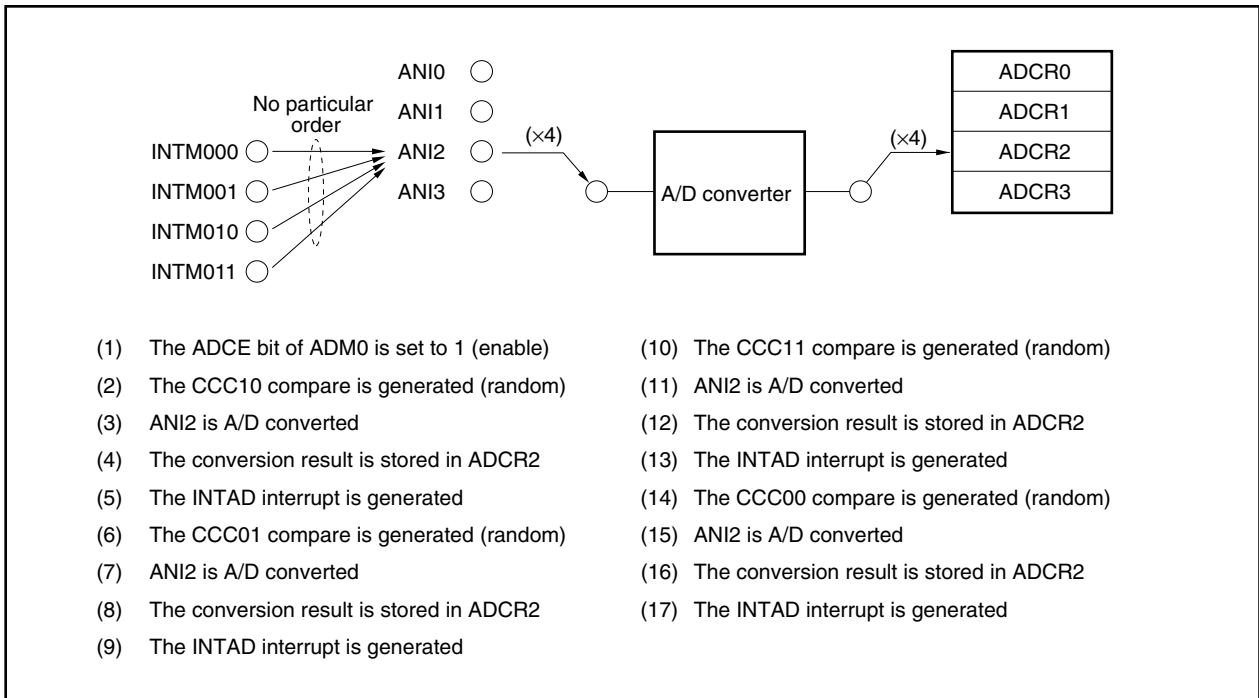
Trigger	Analog Input	A/D Conversion Result Register
INTM000 interrupt	ANIn	ADCRn
INTM001 interrupt	ANIn	ADCRn
INTM010 interrupt	ANIn	ADCRn
INTM011 interrupt	ANIn	ADCRn

In 1-shot mode, A/D conversion stops after four conversions. To restart A/D conversion, set the TMCCEn bit of the TMCCn0 register to 1 to restart the TMCn. When the first match interrupt after TMCn is restarted is generated, the ADCE bit is set (1) and A/D conversion is started (n = 0, 1).

When set to the loop mode, unless the ADCE bit of the ADM0 register is set to 0, A/D conversion is repeated each time a match interrupt is generated.

The match interrupts (INTM000, INTM001, INTM010, INTM011) can be generated in any order. Also, even in cases where the same trigger is input continuously, it is received as a trigger.

Figure 12-10. Example of 4-Trigger Mode (Timer Trigger Select 1-Buffer 4-Trigger) Operation



(2) 4-buffer mode operation (Timer trigger select: 4-buffer)

In this mode, A/D conversion of one analog input is executed 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 in which the average of the A/D conversion result is calculated.

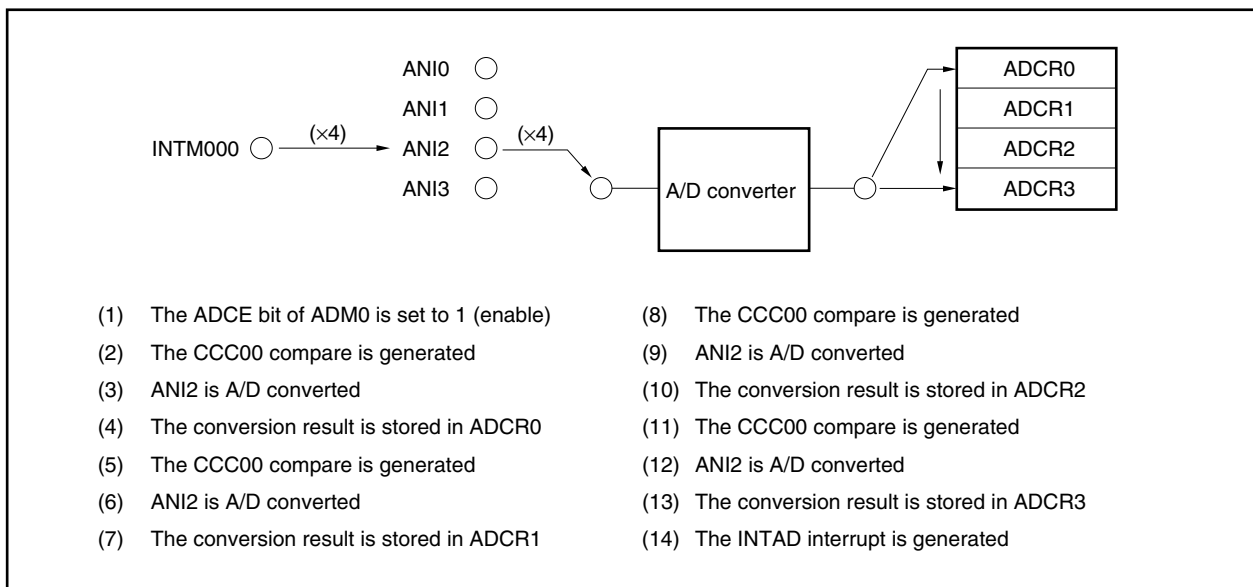
(a) 1-trigger mode

In this mode, one analog input is A/D converted four times using the match interrupt signal (INTM000) as a trigger, and the results are stored in ADCR0 to ADCR3 registers. The A/D conversion end interrupt (INTAD) is generated when the four A/D conversions end and A/D conversion is stopped.

Trigger	Analog Input	A/D Conversion Result Register
INTM000 interrupt	ANIn	ADCR0
INTM000 interrupt	ANIn	ADCR1
INTM000 interrupt	ANIn	ADCR2
INTM000 interrupt	ANIn	ADCR3

If the one-shot mode is set and the TMCCE0 bit of the TMCC00 register is set to 1, and if the match interrupt occurs less than four times, the INTAD interrupt does not occur but is in the standby status.

Figure 12-11. Example of 1-Trigger Mode (Timer Trigger Select 4-Buffer 1-Trigger) Operation



(b) 4-trigger mode

In this mode, one analog input is A/D converted using four match interrupt signals (INTM000, INTM001, INTM010, INTM011) as triggers and the results are stored in four ADCRn registers. The A/D conversion end interrupt (INTAD) is generated when A/D conversion ends, the ADCS bit is reset (0), and A/D conversion is stopped.

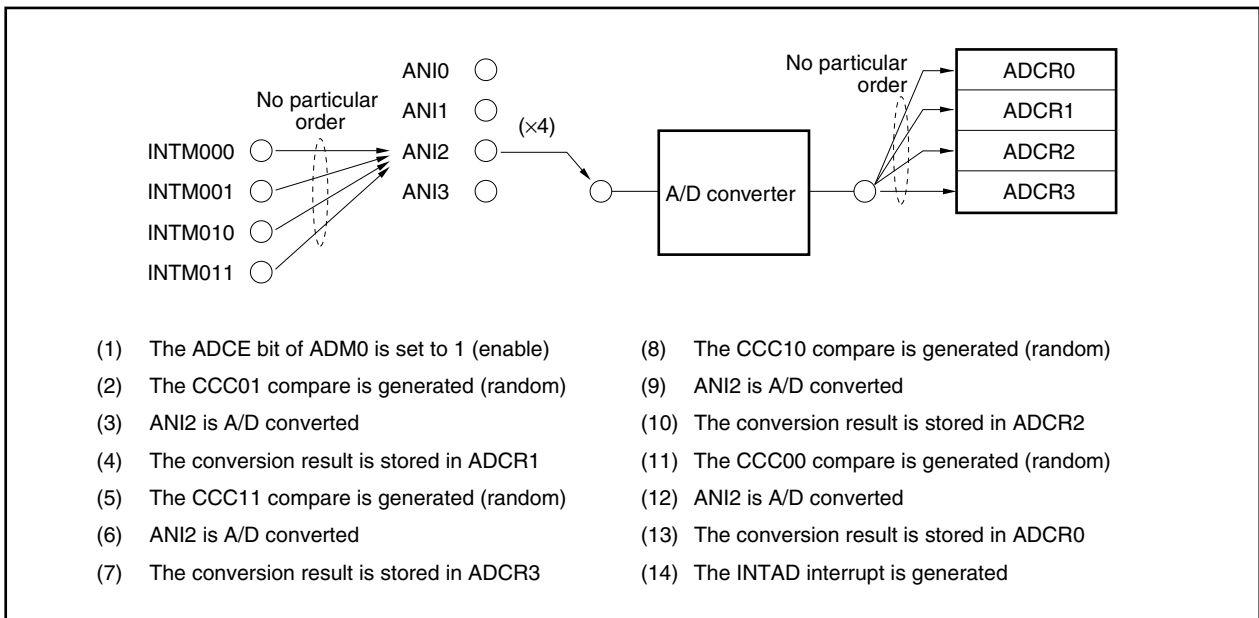
Trigger	Analog Input	A/D Conversion Result Register
INTM000 interrupt	ANIn	ADCR0
INTM001 interrupt	ANIn	ADCR1
INTM010 interrupt	ANIn	ADCR2
INTM011 interrupt	ANIn	ADCR3

In 1-shot mode, A/D conversion stops after four conversions. To restart the A/D conversion, set the TMCCEn bit of the TMCn0 register to 1 to restart the TMCn. When the first match interrupt after TMCn is restarted is generated, the ADCS bit is set (1) and A/D conversion is started (n = 0, 1).

When set to the loop mode, unless the ADCE bit of the ADM0 register is set to 0, A/D conversion is repeated each time a match interrupt is generated.

The match interrupts (INTM000, INTM001, INTM010, INTM011) can be generated in any order, 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 12-12. Example of 4-Trigger Mode (Timer Trigger Select 4-Buffer 4-Trigger) Operation



12.6.2 Scan mode operation

In this mode, the analog inputs specified by the ADM0 register are selected sequentially from the ANI0 pin and are A/D converted for the specified number of times using the match interrupt signal as a trigger.

In the conversion operation, the analog input channels (ANI0 to ANI3) are A/D converted for the specified number of times. When the set number of A/D conversions ends, the A/D conversion end interrupt (INTAD) is generated and A/D conversion is stopped.

There are two scan modes: 1-trigger mode and 4-trigger mode, according to the number of triggers.

This mode is most appropriate for applications in which multiple analog inputs are constantly monitored.

(1) 1-trigger mode (timer triggers scan: 1-trigger)

In this mode, analog inputs are A/D converted for the specified number of times using the match interrupt signal (INTM000) as a trigger. The analog input and ADCRn register correspond one to one.

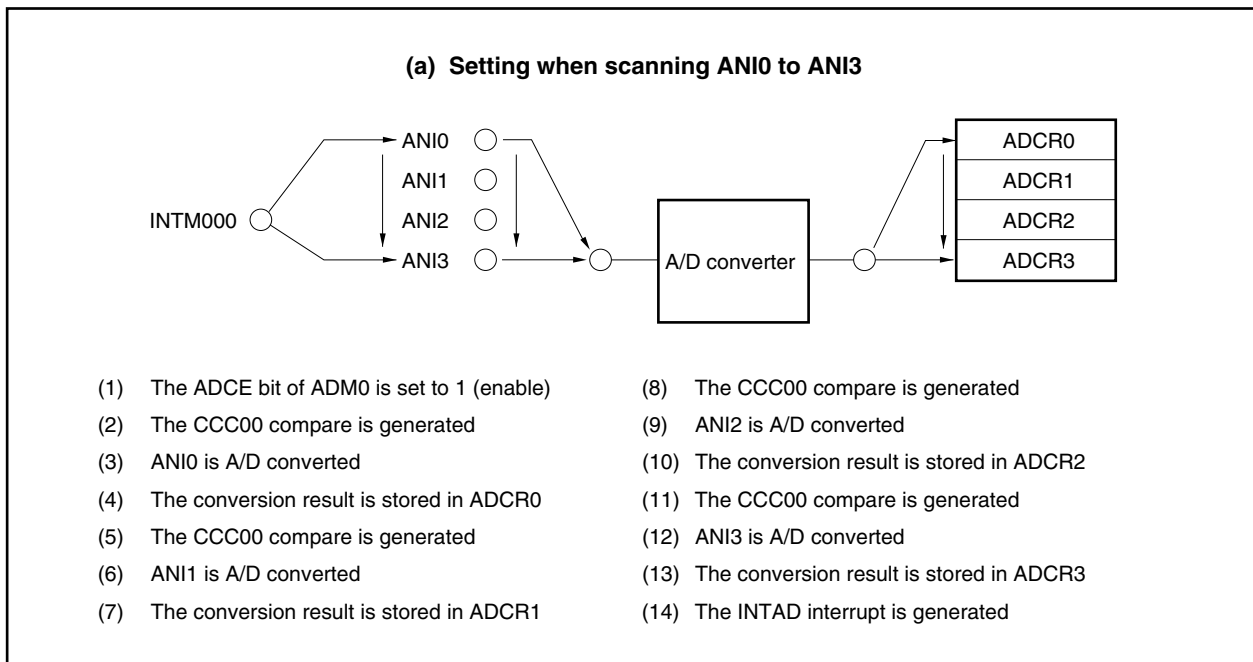
When all the A/D conversions specified have been ended, the A/D conversion end interrupt (INTAD) is generated and A/D conversion is stopped.

Trigger	Analog Input	A/D Conversion Result Register
INTM000 interrupt	ANI0	ADCR0
INTM000 interrupt	ANI1	ADCR1
INTM000 interrupt	ANI2	ADCR2
INTM000 interrupt	ANI3	ADCR3

When the match interrupt is generated after all the specified A/D conversions have ended, A/D conversion is restarted.

In 1-shot mode, and when less than a specified number of match interrupts are generated, the INTAD interrupt is not generated and the standby state is set.

Figure 12-13. Example of 1-Trigger Mode (Timer Trigger Scan 1-Trigger) Operation



(2) 4-trigger mode

In this mode, analog inputs are A/D converted for the number of times specified using the match interrupt signal (INTM000, INTM001, INTM010, INTM011) as a trigger.

The analog input and ADCRn register correspond one to one.

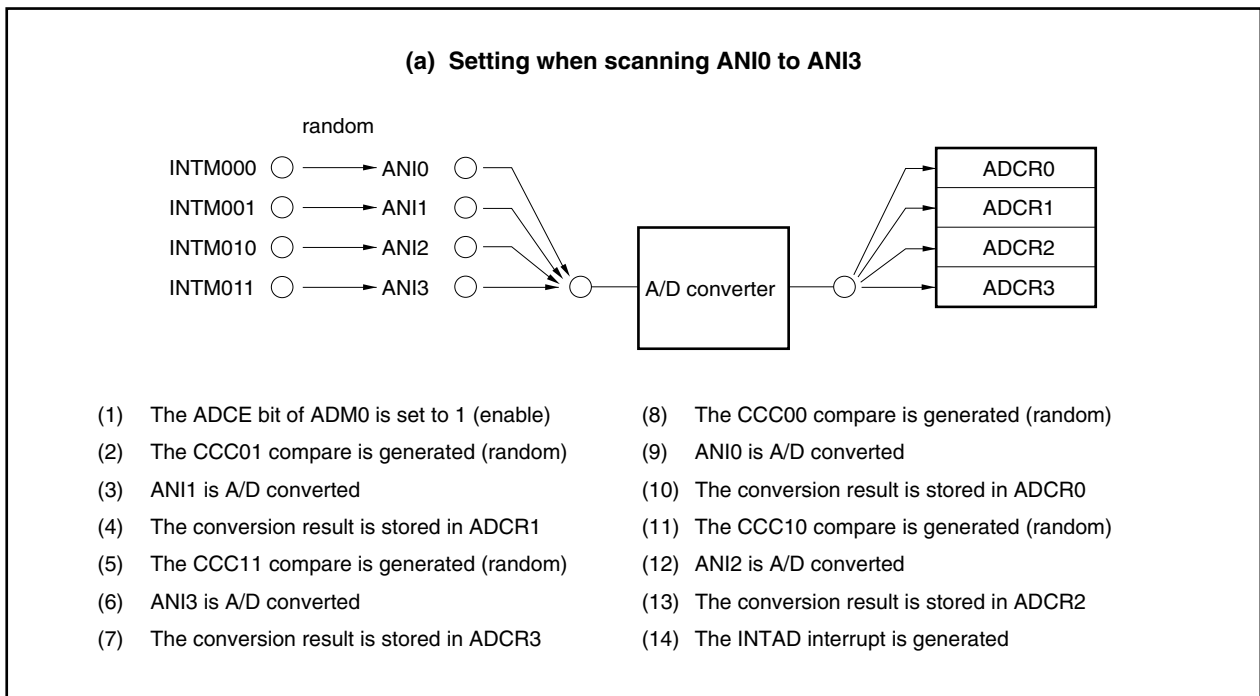
When all the A/D conversions specified have ended, the A/D conversion end interrupt (INTAD) is generated and A/D conversion is stopped.

Trigger	Analog Input	A/D Conversion Result Register
INTM000 interrupt	ANI0	ADCR0
INTM001 interrupt	ANI1	ADCR1
INTM010 interrupt	ANI2	ADCR2
INTM011 interrupt	ANI3	ADCR3

To restart A/D conversion in 1-shot mode, restart TMCn. If set to the loop mode and the ADCE bit of the ADM0 register is 1, A/D conversion is restarted when a match interrupt is generated after conversion has ended.

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 12-14. Example of 4-Trigger Mode (Timer Trigger Scan 4-Trigger) Operation



12.7 Precautions in Operations

12.7.1 Stopping conversion operation

When the ADCE bit of the ADM0 register is set to 0 during a conversion operation, the conversion operation stops and the conversion results are not stored in the ADCRn register (n = 0 to 3).

12.7.2 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 input is not counted. Take note, therefore, that the saving of the result to the ADCRn register upon the generation of an interrupt is an abnormality (n = 0 to 3).

(2) When $0 < \text{interval} < \text{conversion operation time}$

When the timer trigger is input during a conversion operation, the conversion operation is aborted and the conversion starts according to the last timer trigger input.

When conversion operations are aborted, the conversion results are not stored in the ADCRn register, and the number of trigger input are not counted. Take note, therefore, that the saving of the result to the ADCRn register upon the generation of an interrupt is an abnormality (n = 0 to 3).

(3) When interval = conversion operation time

When a trigger is input concurrently with the end of conversion (the conversion complete signal and the trigger are in contention), although the number of triggers input are counted, an interrupt is generated, and the value at the end of conversion is correctly saved in the ADCRn register, design should be performed so that the interval is greater than the conversion operation time.

12.7.3 Operation in standby mode

(1) HALT mode

In this mode, A/D conversion continues. When this mode is released using the NMI input, the ADM0 and ADM1 registers and ADCRn register hold the value (n = 0 to 3).

(2) IDLE mode, software 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 or maskable interrupt input (INTP1x), the ADM0 and ADM1 registers and the ADCRn register hold the value. However, when the IDLE or software STOP mode is set during a conversion operation, the conversion operation is stopped. At this time, if the mode released using the NMI input or maskable interrupt input (INTP1x), the conversion operation resumes, but the conversion result written to the ADCRn register will become undefined (x = 00, 01, 10, n = 0 to 3).

12.7.4 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 also functions as a compare register match interrupt for the CPU. In order to prevent match interrupts from the compare register for the CPU, disable interrupts by the mask bits (P00MK0, P00MK1, P01MK0, P01MK1) of the interrupt control register (P00IC0, P00IC1, P01IC0, P01IC1).

★ 12.7.5 Reconversion operation in timer 1 trigger mode

In the timer 1 trigger mode, A/D conversion is started with the match interrupt signal (INTM000) as the trigger. However, when interrupt sources which are non-triggers (INTM001, INTM010, INTM011, INTP001^{Note}, INTP010^{Note}, INTP011^{Note}) are generated during A/D conversion, after this A/D conversion ends normally, the same A/D conversion may start again (reconversion operation). However, the reconversion operation will not be performed unless non-trigger interrupt sources are generated under these conditions.

Note External interrupt signals also used as external capture trigger inputs of timer C (TMC0, TMC1) also trigger reconversion.

(1) Reconversion operation in the timer trigger select 1 buffer 1 trigger mode

When non-trigger interrupt sources are generated during A/D conversion, the first A/D conversion ends normally, and the A/D conversion end interrupt (INTAD) is generated. The A/D conversion results are stored in the ADCRn register. A restarted A/D conversion is carried out normally, and the A/D conversion results are overwritten in the ADCRn register. During reconversion, the ADCRn register can be read. After A/D conversion ends, the INTAD interrupt is generated, and A/D conversion stops.

(2) Reconversion operation in timer trigger select 4 buffer 1 trigger mode, timer trigger scan 1 trigger mode

A/D conversion is performed smoothly until non-trigger interrupt sources are generated during conversion. When non-trigger interrupt sources are generated during A/D conversion, the current A/D conversion ends normally, and the A/D conversion results are stored in the ADCRn register. After this, the same A/D conversion is performed, and the A/D conversion results are overwritten in the ADCRn register. During reconversion, the ADCRn register can be read. After this, the remaining A/D conversion operations are performed normally, the A/D conversion end interrupt (INTAD) is generated, and A/D conversion stops.

Caution When non-trigger interrupt sources are generated during the last A/D conversion, the last A/D conversion ends normally, and the A/D conversion end interrupt (INTAD) is generated. After this, the same conversion as the last A/D conversion is performed, the INTAD interrupt is generated, and A/D conversion stops.

When reconversion operations occur, as conversion results are normal values, the effect on conversion will be minimized when using a methods in which the latest conversion values are acquired. However, if reconversion operations become abnormal, be sure to use the A/D trigger mode and start A/D conversion by setting the ADCE bit of the ADM0 register in the interrupt servicing routine of the compare match interrupt of the timer.

★ 12.7.6 Supplementary information on A/D conversion time

The time taken from trigger input to the end of A/D conversion (t) is as follows.

In A/D trigger mode (refer to **Figures 12-15** and **12-17**):

$$t = 9 \text{ to } 11 \text{ clocks} + \text{Number of clocks specified by the FR2 to FR0 bits of ADM1} + 2 \text{ clocks}$$

In timer trigger mode (refer to **Figures 12-16** and **12-17**):

$$t = 5 \text{ to } 7 \text{ clocks} + \text{Number of clocks specified by the FR2 to FR0 bits of ADM1} + 2 \text{ clocks}$$

Figure 12-15. A/D Trigger Mode A/D Conversion Time: When ADM1 = 00H

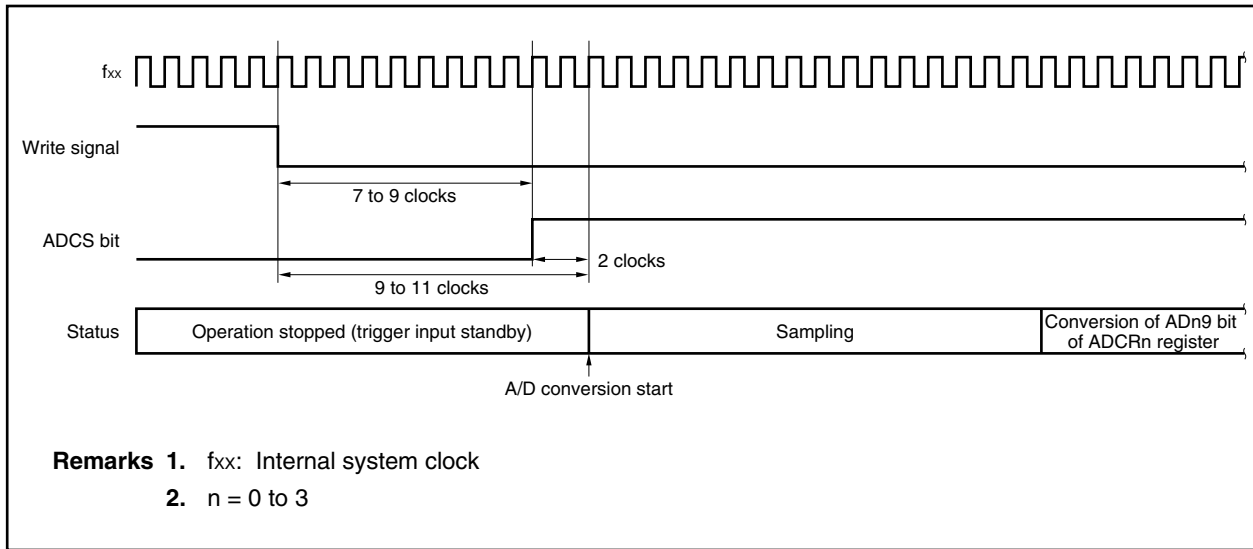
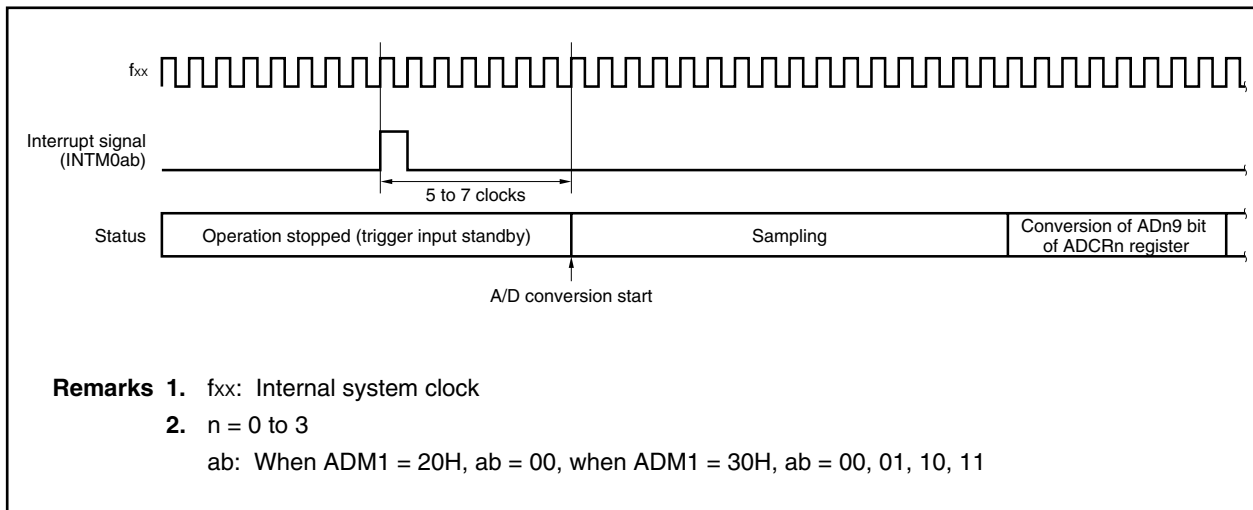
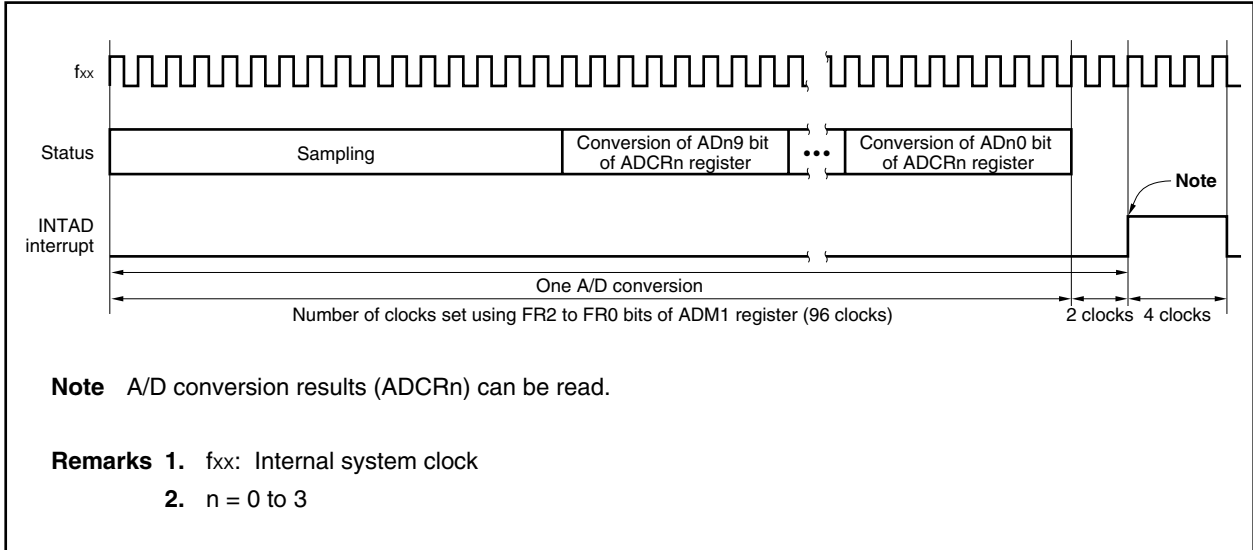


Figure 12-16. Timer Trigger Mode A/D Conversion Time: When ADM1 = 20H or 30H



**Figure 12-17. A/D Conversion Outline: One A/D Conversion, FR0 to FR2
Bits of ADM1 Register = 000 (96 Clocks)**



★ 12.8 How to Read A/D Converter Characteristics Table

This section describes the terms related to the A/D converter.

(1) Resolution

The minimum analog input voltage that can be recognized, i.e., the ratio of an analog input voltage to 1 bit of digital output is called 1 LSB (least significant bit). The ratio of 1 LSB to the full scale is expressed as %FSR (full-scale range). %FSR is the ratio of a range of convertible analog input voltages expressed as a percentage, and can be expressed as follows, independently of the resolution.

$$\begin{aligned}
 1\%FSR &= (\text{Maximum value of convertible analog input voltage} - \text{Minimum value of convertible analog input voltage})/100 \\
 &= (AV_{REF} - 0)/100 \\
 &= AV_{REF}/100
 \end{aligned}$$

Where the resolution is 10 bits, 1 LSB is as follows:

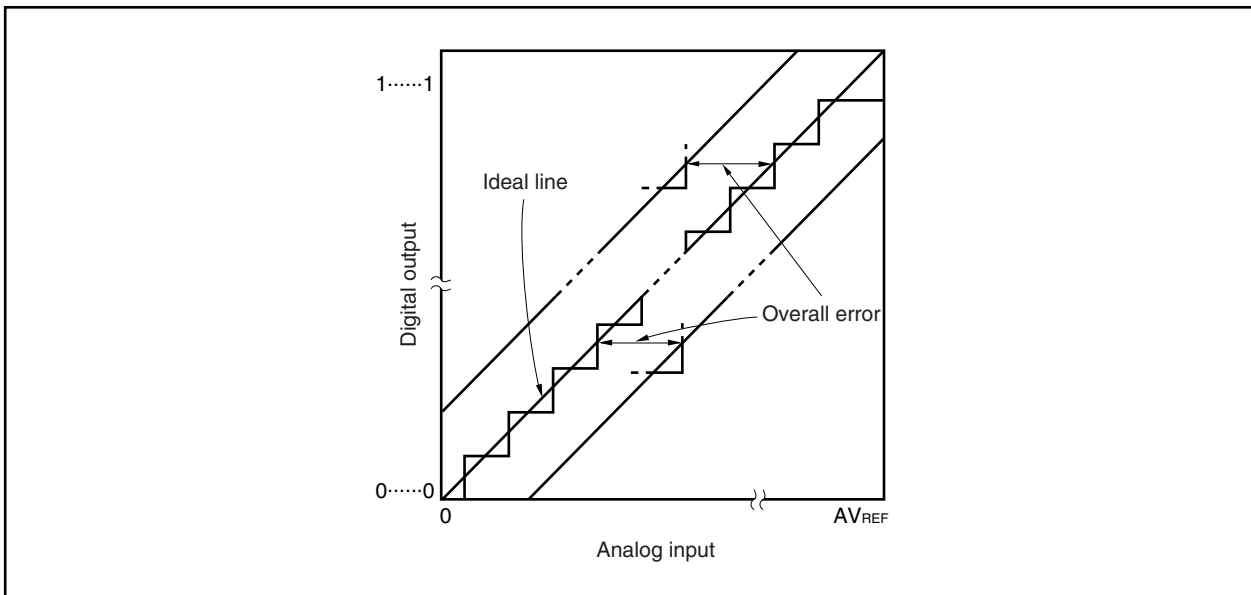
$$\begin{aligned}
 1 \text{ LSB} &= 1/2^{10} = 1/1,024 \\
 &= 0.098\%FSR
 \end{aligned}$$

The accuracy is determined by the overall error, independently of the resolution.

(2) Overall error

This is the maximum value of the difference between an actually measured value and a theoretical value. It is a total of zero-scale error, full-scale error, linearity error, and a combination of these errors. The overall error in the characteristics table does not include the quantization error.

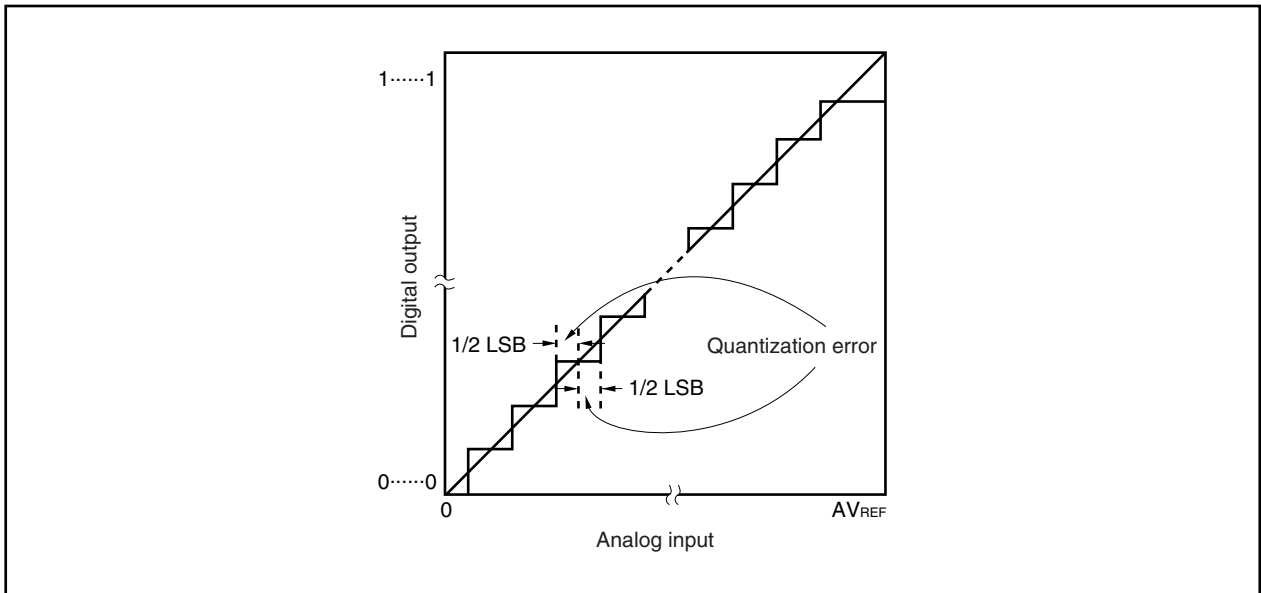
Figure 12-18. Overall Error



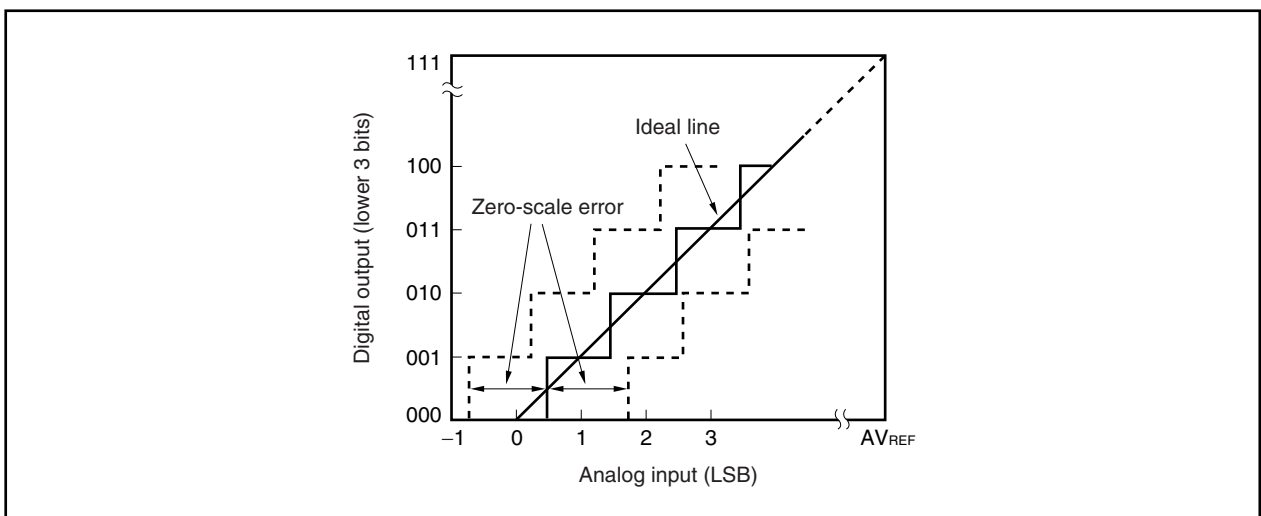
(3) Quantization error

This is an error of $\pm 1/2$ LSB that inevitably occurs when an analog value is converted into a digital value. Because the A/D converter converts analog input voltages in a range of $\pm 1/2$ LSB into the same digital codes, a quantization error is unavoidable.

This error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, or differential linearity error in the characteristics table.

Figure 12-19. Quantization Error**(4) Zero-scale error**

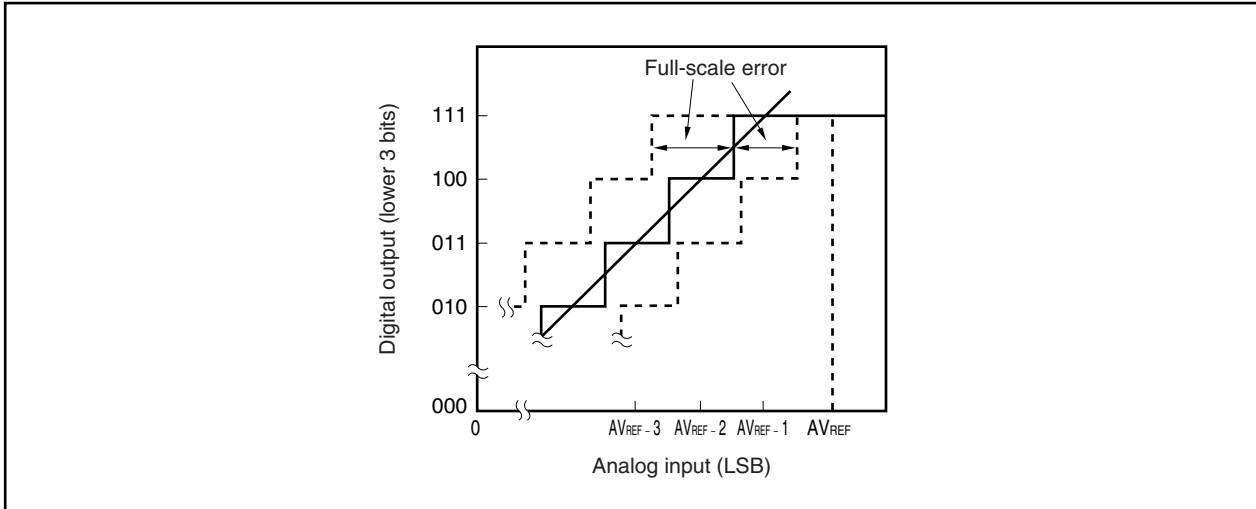
This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 0...000 to 0...001 (1/2 LSB).

Figure 12-20. Zero-Scale Error

(5) Full-scale error

This is the difference between the actually measured analog input voltage and its theoretical value when the digital output changes from 1...110 to 0...111 (full scale - 3/2 LSB).

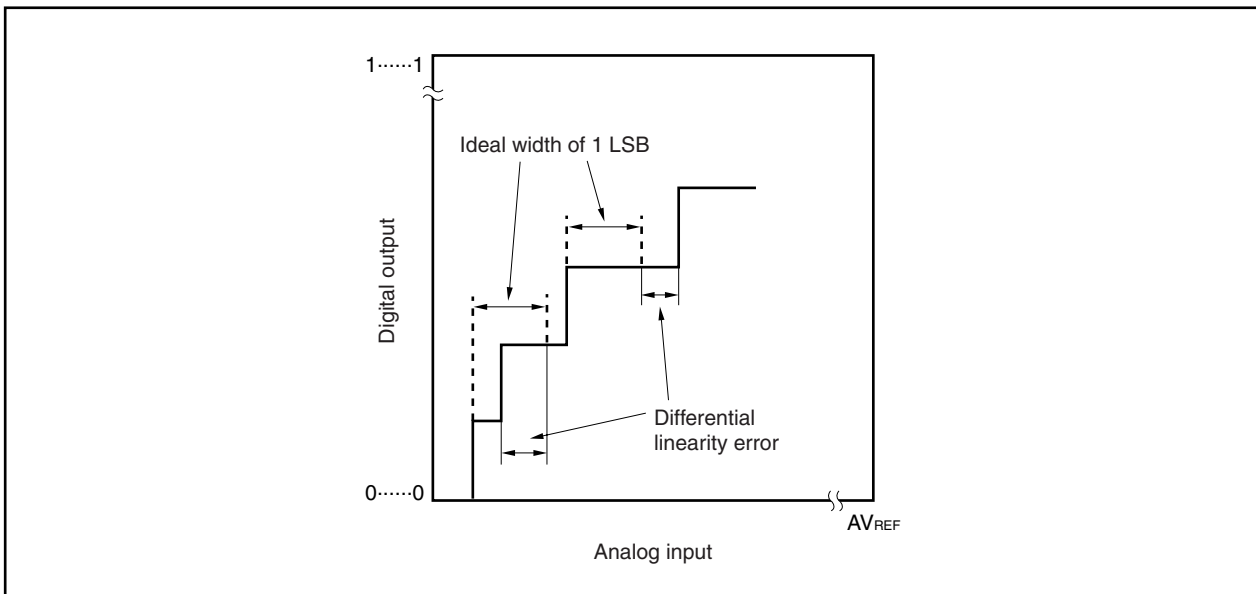
Figure 12-21. Full-Scale Error



(6) Differential linearity error

Ideally, the width to output a specific code is 1 LSB. This error indicates the difference between the actually measured value and its theoretical value when a specific code is output.

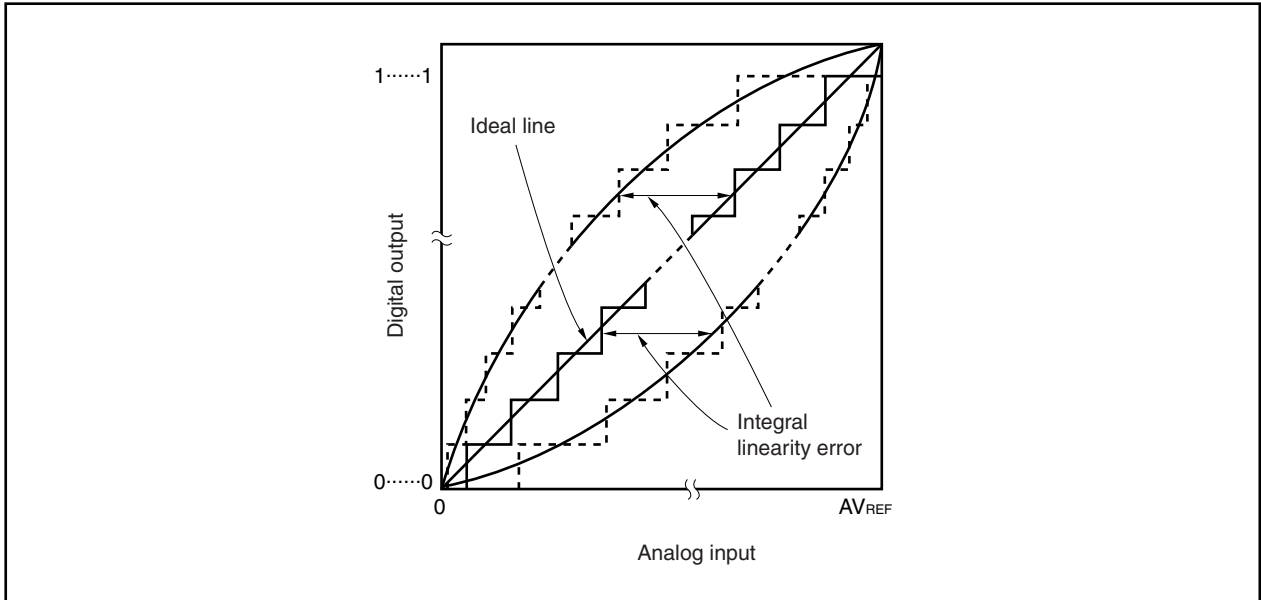
Figure 12-22. Differential Linearity Error



(7) Integral linearity error

This error indicates the extent to which the conversion characteristics differ from the ideal linear relations. It indicates the maximum value of the difference between the actually measured value and its theoretical value where the zero-scale error and full-scale error are 0.

Figure 12-23. Integral Linearity Error



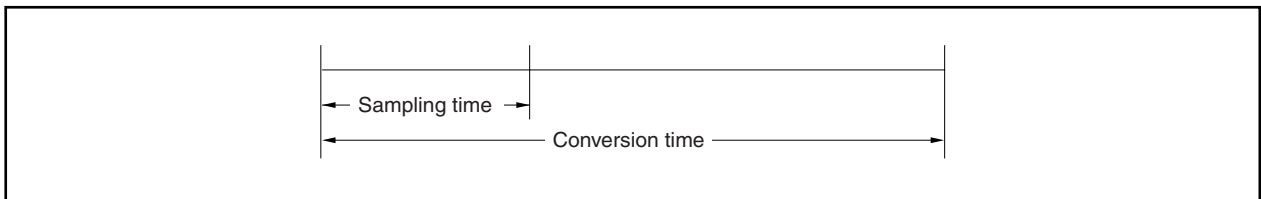
(8) Conversion time

This is the time required to obtain a digital output after an analog input voltage has been assigned. The conversion time in the characteristics table includes the sampling time.

(9) Sampling time

This is the time for which the analog switch is ON to load an analog voltage to the sample & hold circuit.

Figure 12-24. Sampling Time



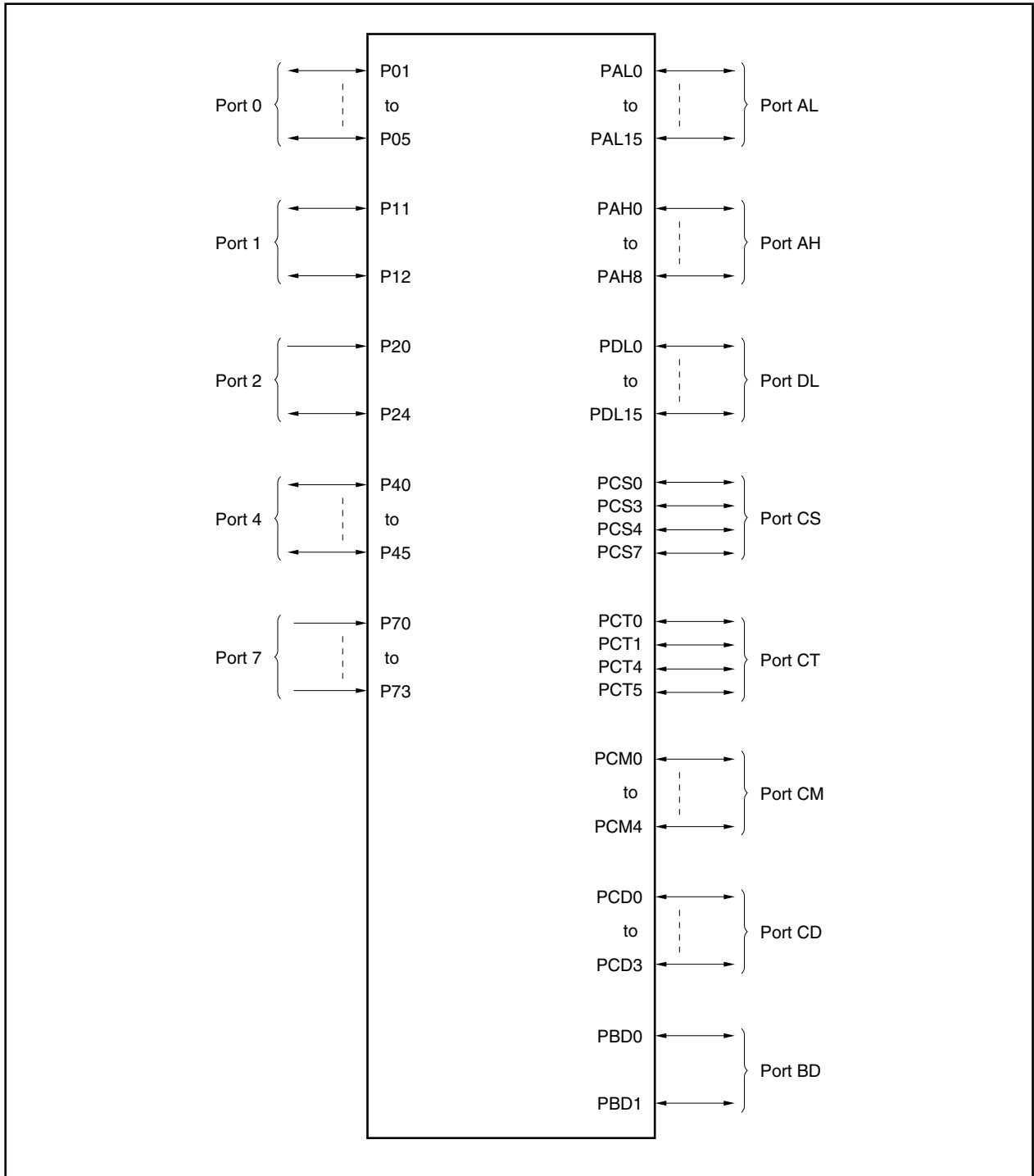
CHAPTER 13 PORT FUNCTIONS

13.1 Features

- Input-only ports: 5
I/O ports: 74
- Alternately function as other peripheral I/O pins.
- Input and output can be specified in 1-bit units.

13.2 Port Configuration

The V850E/MA2 incorporates a total of 79 input/output ports (including 5 input-only ports) labeled ports 0 to 2, 4, 7, AL, AH, DL, CS, CT, CM, CD, and BD. The port configuration is shown below.



(1) Function of each port

The port functions of this product are shown below. 8-bit and 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 pins in the control mode. For the block types of each port, refer to **(3) Block diagram of port.**

Port Name	Pin Name	Port Function	Function in Control Mode	Block Type
Port 0	P01 to P05	5-bit I/O	Real-time pulse unit (RPU) I/O External interrupt input DMA controller input	A, B, H
Port 1	P11, P12	2-bit I/O	Real-time pulse unit (RPU) input External interrupt input	B
Port 2	P20, P24	1-bit input, 1-bit I/O	NMI input External interrupt input DMA controller output	F, N
★ Port 4	P40 to P45	6-bit I/O	Serial interface I/O (UART0/CSI0, UART1/CSI1)	G, H, M
Port 7	P70 to P73	4-bit input	A/D converter input	C
Port AL	PAL0 to PAL15	16-bit I/O	External address bus (A0 to A15)	J
Port AH	PAH0 to PAH8	9-bit I/O	External address bus (A16 to A24)	J
Port DL	PDL0 to PDL15	16-bit I/O	External data bus (D0 to D15)	O
Port CS	PCS0, PCS3, PCS4, PCS7	4-bit I/O	External bus interface control signal output	J
Port CT	PCT0,PCT1, PCT4, PCT5	4-bit I/O	External bus interface control signal output	J
★ Port CM	PCM0 to PCM4	5-bit I/O	Wait insertion signal input Internal system clock output External bus interface control signal I/O	D, J
Port CD	PCD0 to PCD3	4-bit I/O	External bus interface control signal output	J, K
Port BD	PBD0, PBD1	2-bit I/O	DMA controller output	J

Remark The **Cautions** are explained on the following page.

Cautions 1. When switching the mode of a port that functions as an output or I/O pin to control mode, be sure to follow the procedure below.

<1> Set the inactive level of the signals output in control mode to the appropriate bits in port n (n = 0 to 2, 4, AL, AH, DL, CS, CT, CM, CD, and BD).

<2> The mode is switched to control mode by the port n mode control register (PM_{Cn}).

If <1> above is not performed, the contents of port n may be output for a moment when the mode is switched from port mode to control mode.

★ 2. When port manipulation is performed by a bit manipulation instruction (SET1, CLR1, or NOT1), perform a byte-data read of the port, process the data of only the bits to be manipulated, and write the byte data after conversion back to the port.

For example, in ports in which input and output are mixed, because the contents of the output latch are overwritten to bits other than the bits for manipulation, the output latch of the input pin becomes undefined (in the input mode, however, the pin status does not change because the output buffer is off).

Therefore, when switching the port from input to output, set the expected output value to the corresponding bit, and then switch to the output port. This is the same as when the control mode and output ports are mixed.

★ 3. The state of a port pin can be read by setting the port n mode register (PM_n) to the input mode regardless of the settings of the PM_{Cn} register. When the PM_n register is set to the output mode, the value of the port n register (P_n) can be read in the port mode while the output state of the alternate function can be read in the control mode.

(2) Function when each port's pins are reset and registers that set the port/control mode

Port Name	Pin Name	Pin Function After Reset		Register That Sets the Mode
Port 0	P01/INTP000/TI000	P01 (input mode)		PMC0
	P02/INTP001	P02 (input mode)		
	P03/TO00	P03 (input mode)		
	P04/ $\overline{\text{DMARQ0}}$ / $\overline{\text{INTP100}}$	P04 (input mode)		PMC0, PFC0
	P05/ $\overline{\text{DMARQ1}}$ / $\overline{\text{INTP101}}$	P05 (input mode)		
Port 1	P11/INTP010/TI010	P11 (input mode)		PMC1
	P12/INTP011	P12 (input mode)		
Port 2	P20/NMI	NMI		-
	P24/ $\overline{\text{TC0}}$ / $\overline{\text{INTP110}}$	P24 (input mode)		PMC2, PFC2
Port 4	P40/TXD0/SO0	P40 (input mode)		PMC4, PFC4
	P41/RXD0/SI0	P41 (input mode)		
	P42/ $\overline{\text{SCK0}}$	P42 (input mode)		PMC4
	P43/TXD1/SO1	P43 (input mode)		PMC4, PFC4
	P44/RXD1/SI1	P44 (input mode)		
	P45/ $\overline{\text{SCK1}}$	P45 (input mode)		PMC4
Port 7	P70/ANI0 to P73/ANI3	P70 to P73 (input mode)		-
Port BD	PBD0/ $\overline{\text{DMAAK0}}$, PBD1/ $\overline{\text{DMAAK1}}$	PBD0, PBD1 (input mode)		PM CBD
★ Port CM	PCM0/ $\overline{\text{WAIT}}$	PCM0 (input mode)	$\overline{\text{WAIT}}$	PMCCM
	PCM1/CLKOUT	PCM1 (input mode)	CLKOUT	
	PCM2/ $\overline{\text{HLDAK}}$	PCM2 (input mode)	$\overline{\text{HLDAK}}$	
	PCM3/ $\overline{\text{HLDRQ}}$	PCM3 (input mode)	$\overline{\text{HLDRQ}}$	
	PCM4/ $\overline{\text{REFRQ}}$	PCM4 (input mode)	$\overline{\text{REFRQ}}$	
Port CT	PCT0/ $\overline{\text{LWR}}$ / $\overline{\text{LDQM}}$	PCT0 (input mode)	$\overline{\text{LWR}}$ / $\overline{\text{LDQM}}$	PM CCT
	PCT1/ $\overline{\text{UWR}}$ / $\overline{\text{UDQM}}$	PCT1 (input mode)	$\overline{\text{UWR}}$ / $\overline{\text{UDQM}}$	
	PCT4/ $\overline{\text{RD}}$	PCT4 (input mode)	$\overline{\text{RD}}$	
	PCT5/ $\overline{\text{WE}}$	PCT5 (input mode)	$\overline{\text{WE}}$	
Port CS	PCS0/ $\overline{\text{CS0}}$	PCS0 (input mode)	$\overline{\text{CS0}}$	PM CCS
	PCS3/ $\overline{\text{CS3}}$	PCS3 (input mode)	$\overline{\text{CS3}}$	
	PCS4/ $\overline{\text{CS4}}$	PCS4 (input mode)	$\overline{\text{CS4}}$	
	PCS7/ $\overline{\text{CS7}}$	PCS7 (input mode)	$\overline{\text{CS7}}$	
Port CD	PCD0/SDCKE	PCD0 (input mode)	SDCKE	PM CCD
	PCD1/SDCLK	PCD1 (input mode)	SDCLK	
	PCD2/ $\overline{\text{LBE}}$ / $\overline{\text{SDCAS}}$	PCD2 (input mode)	$\overline{\text{LBE}}$ / $\overline{\text{SDCAS}}$	PM CCD, PFCCD
	PCD3/ $\overline{\text{UBE}}$ / $\overline{\text{SDRAS}}$	PCD3 (input mode)	$\overline{\text{UBE}}$ / $\overline{\text{SDRAS}}$	
Port AH	PAH0/A16 to PAH8/A24	PAH0 to PAH8 (input mode)	A16 to A24	PM CAH
Port AL	PAL0/A0 to PAL15/A15	PAH0 to PAH15 (input mode)	A0 to A15	PM CAL
Port DL	PDL0/D0 to PDL15/D15	PDL0 to PDL15 (input mode)	D0 to D15	PM CDL

(3) Block diagram of port

Figure 13-1. Block Diagram of Type A

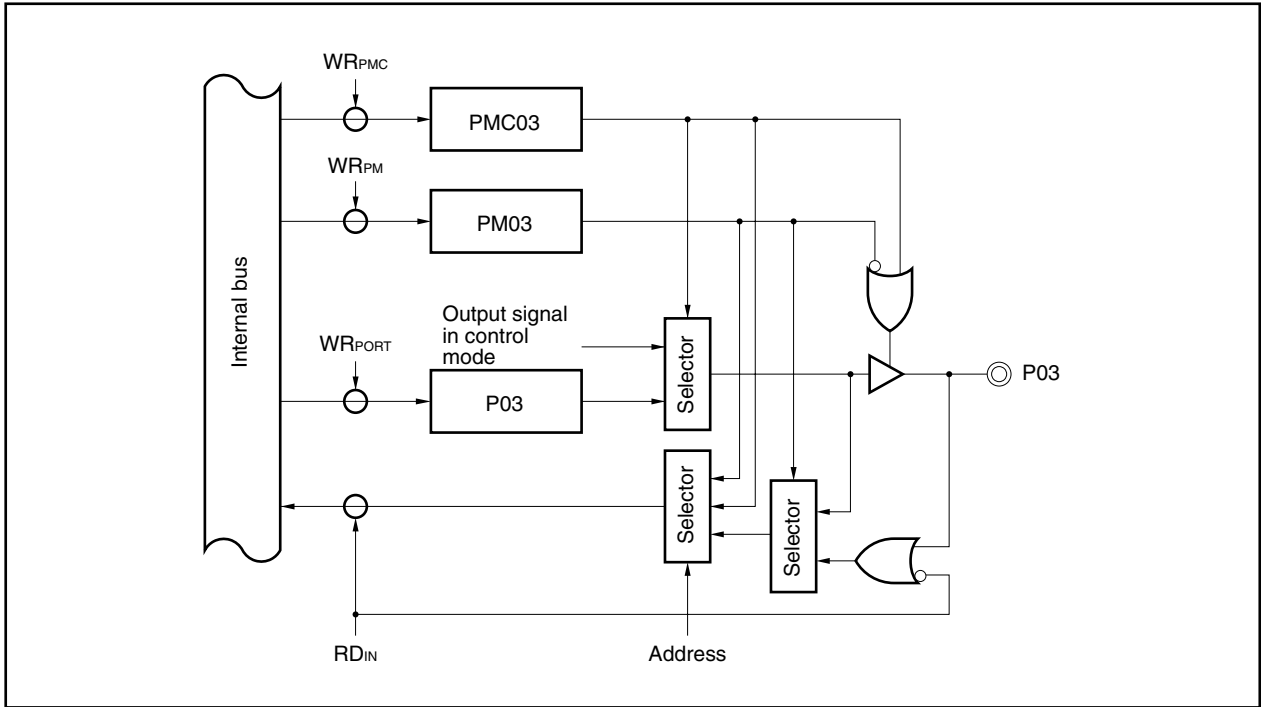


Figure 13-2. Block Diagram of Type B

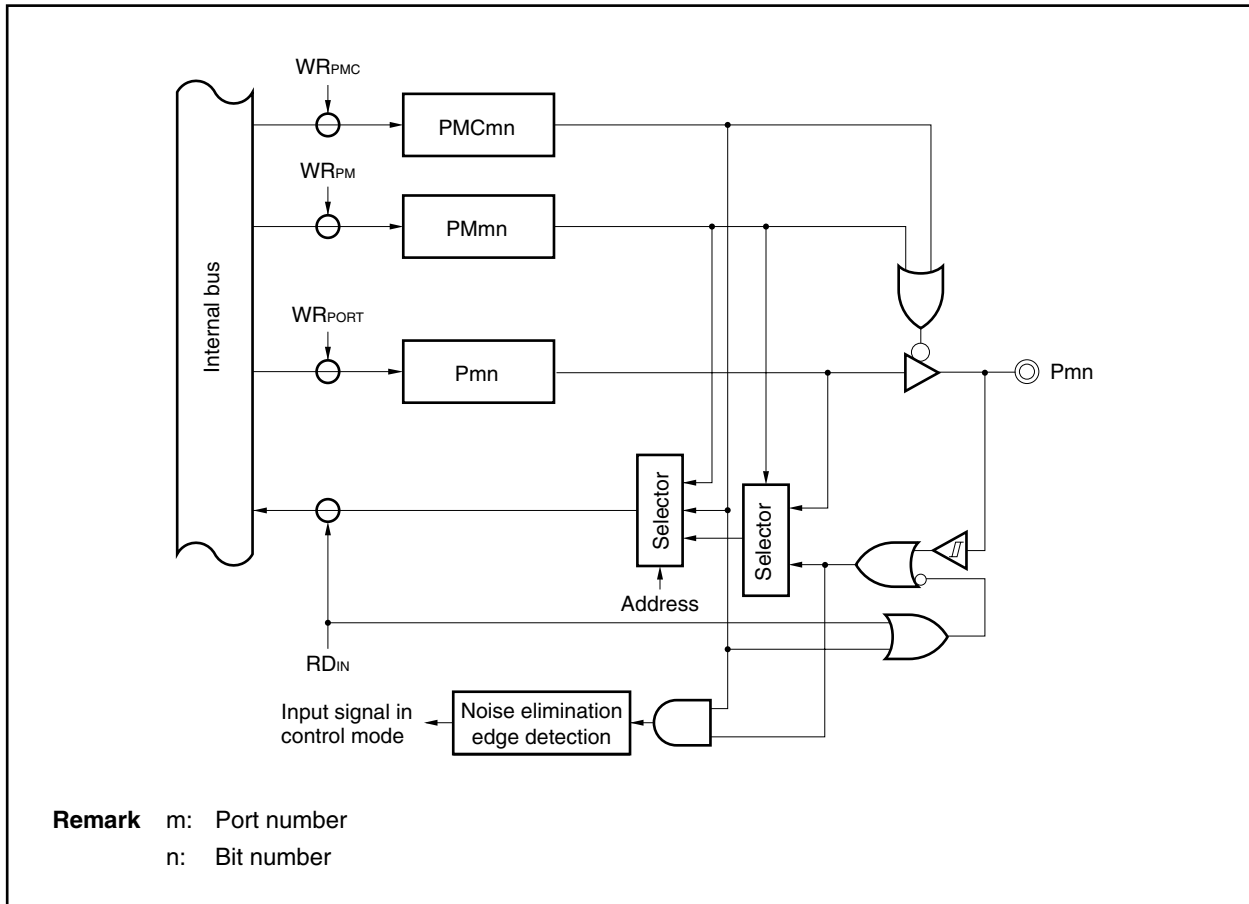
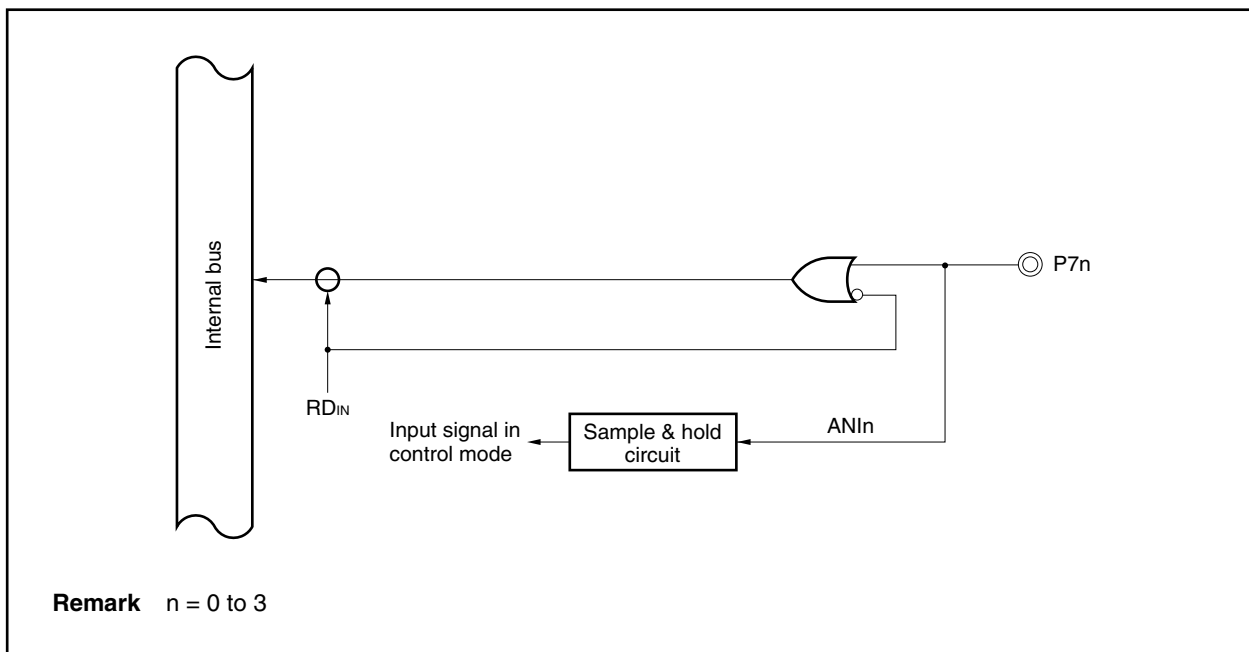


Figure 13-3. Block Diagram of Type C



★

Figure 13-4. Block Diagram of Type D

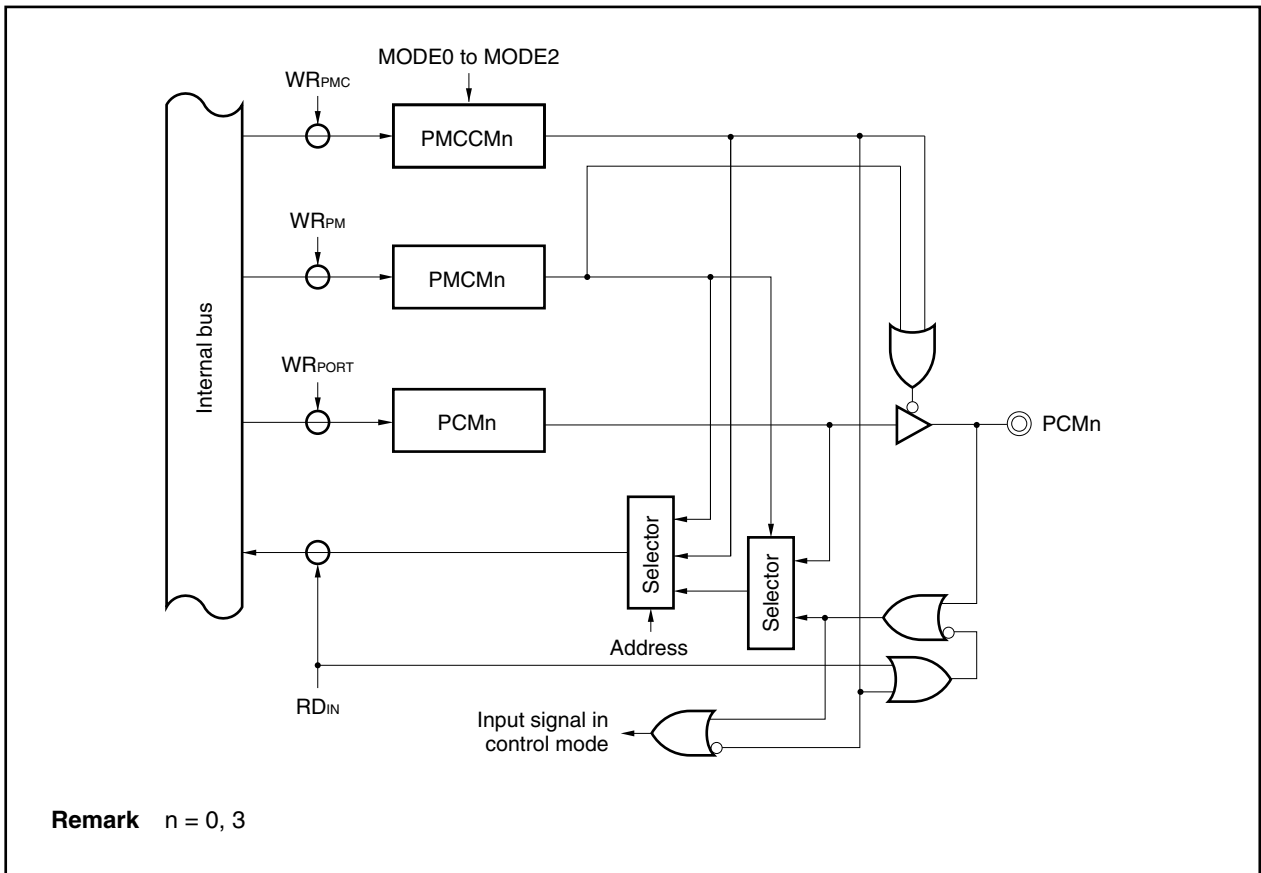
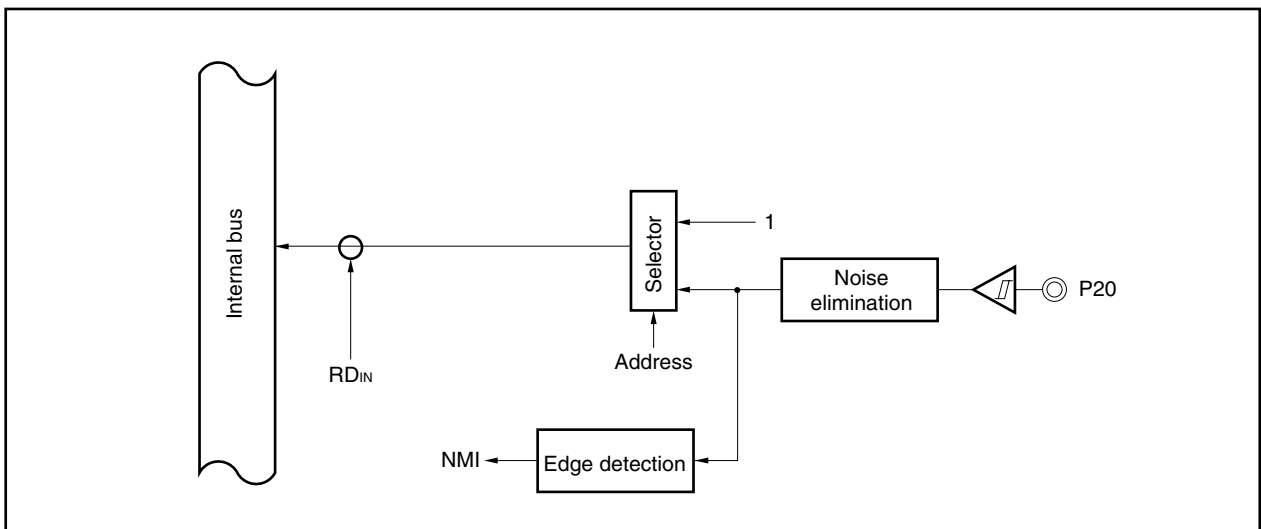
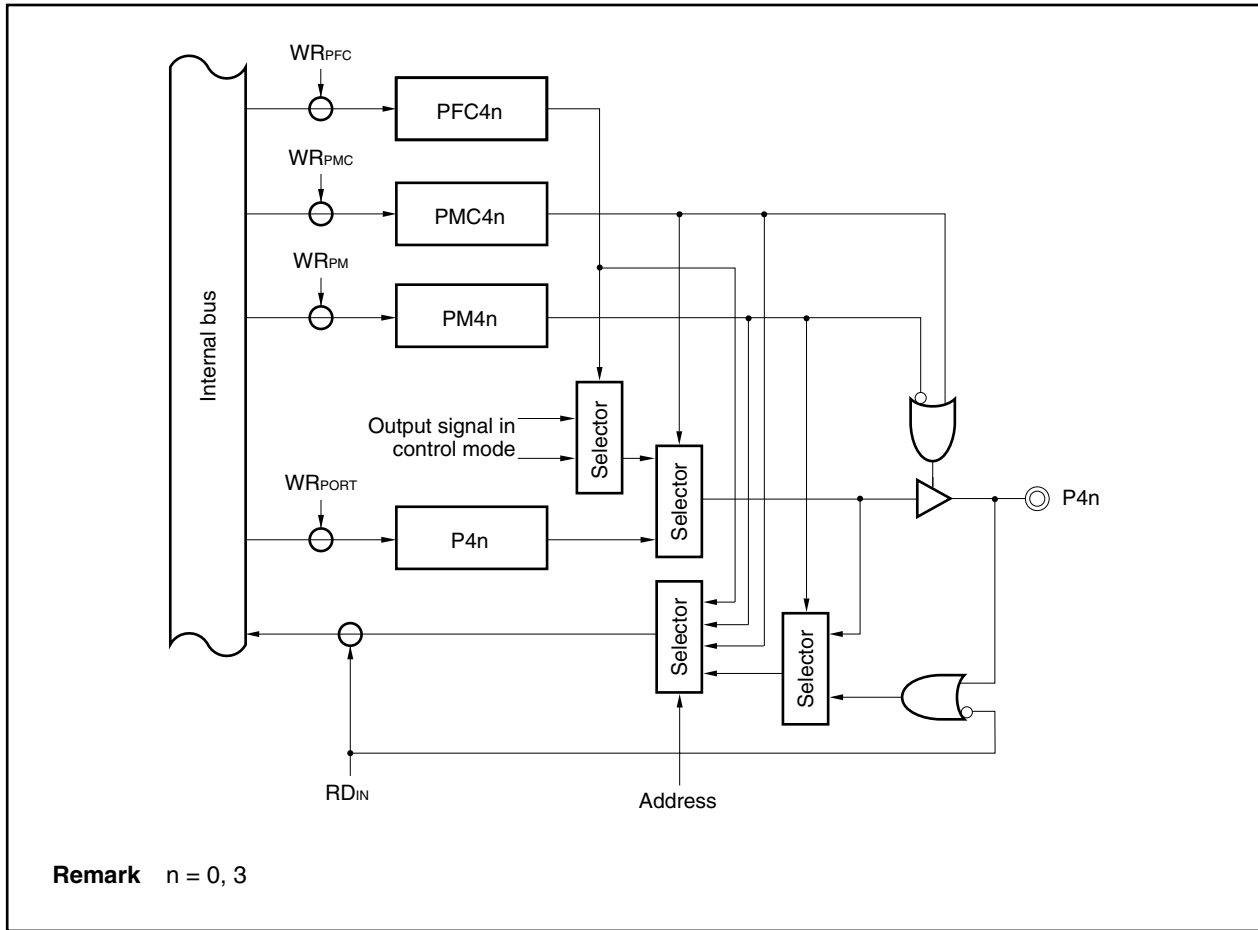


Figure 13-5. Block Diagram of Type F



★

Figure 13-6. Block Diagram of Type G



★

Figure 13-7. Block Diagram of Type H

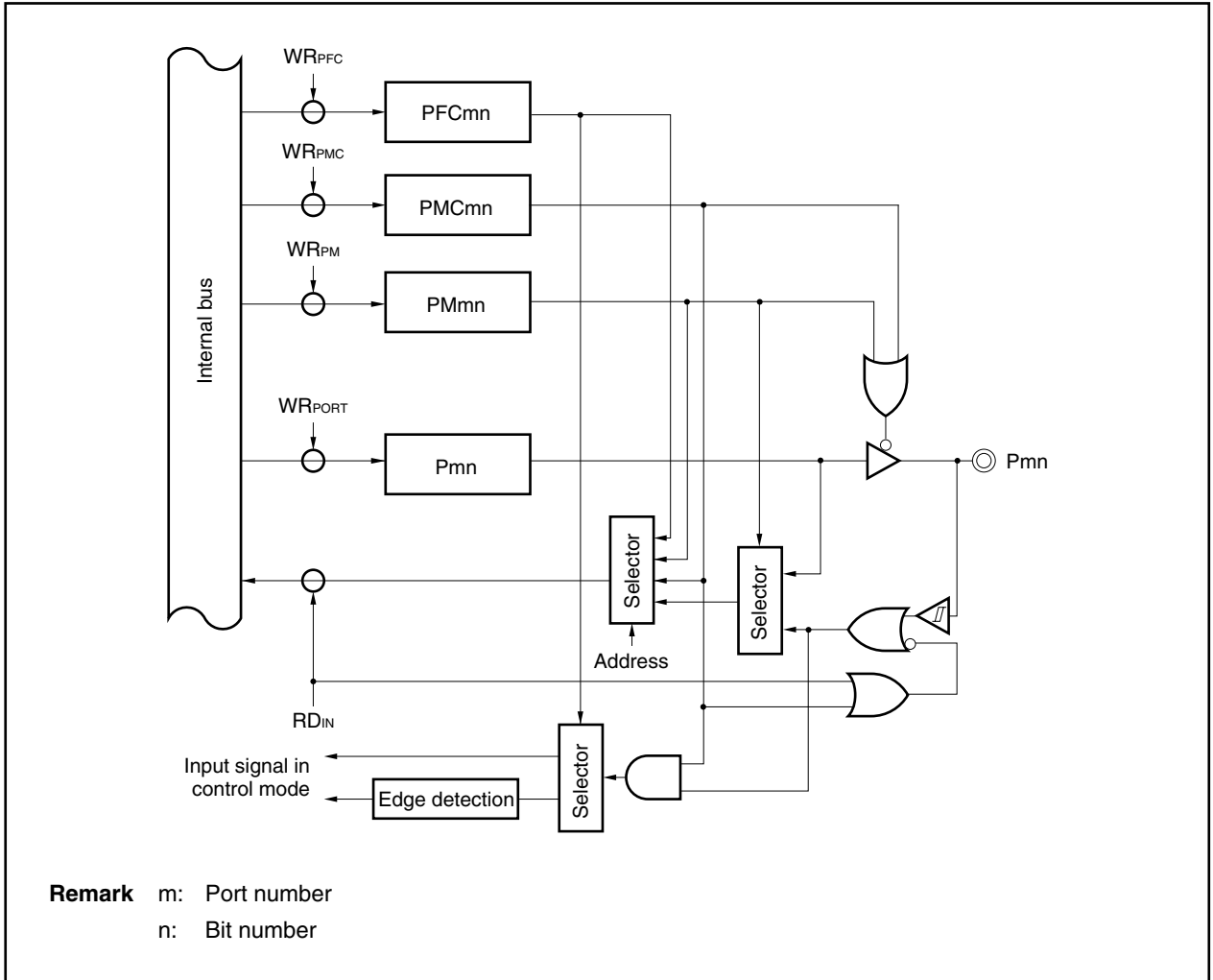


Figure 13-8. Block Diagram of Type J

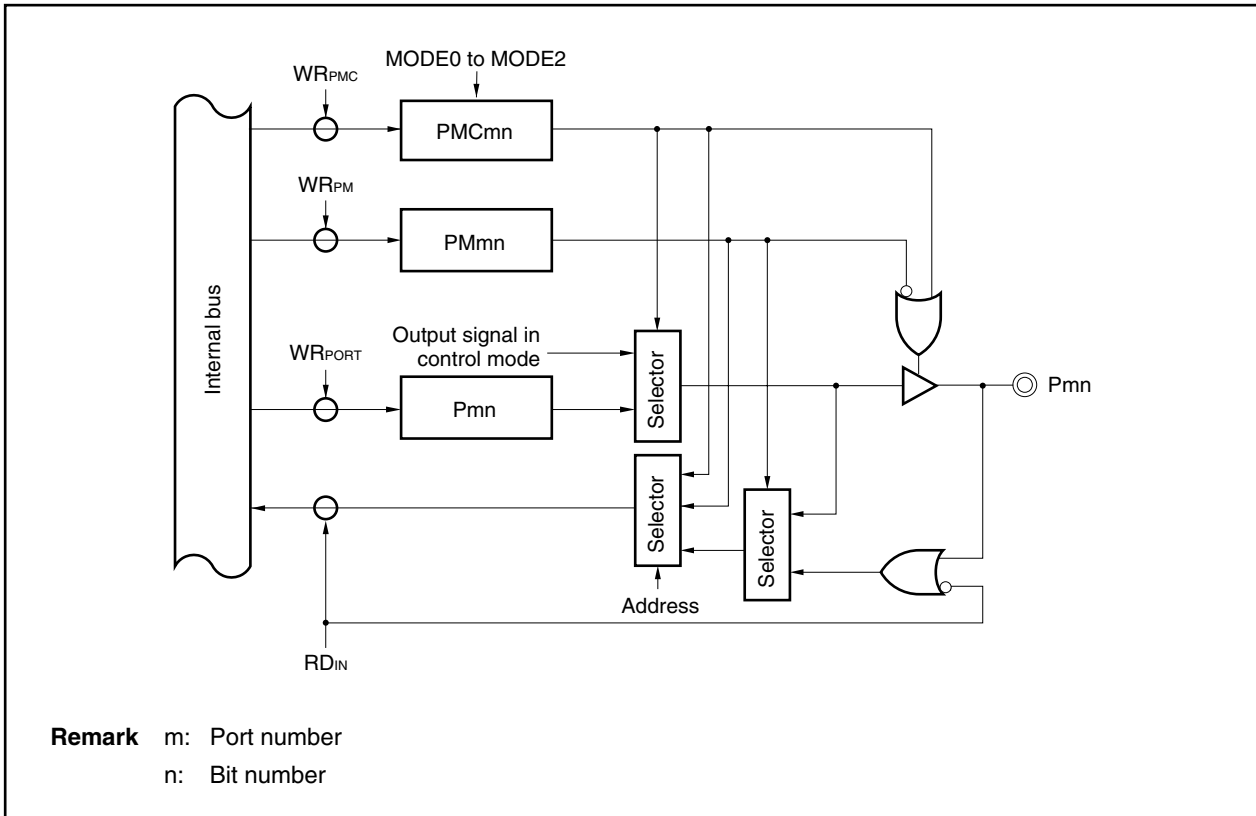
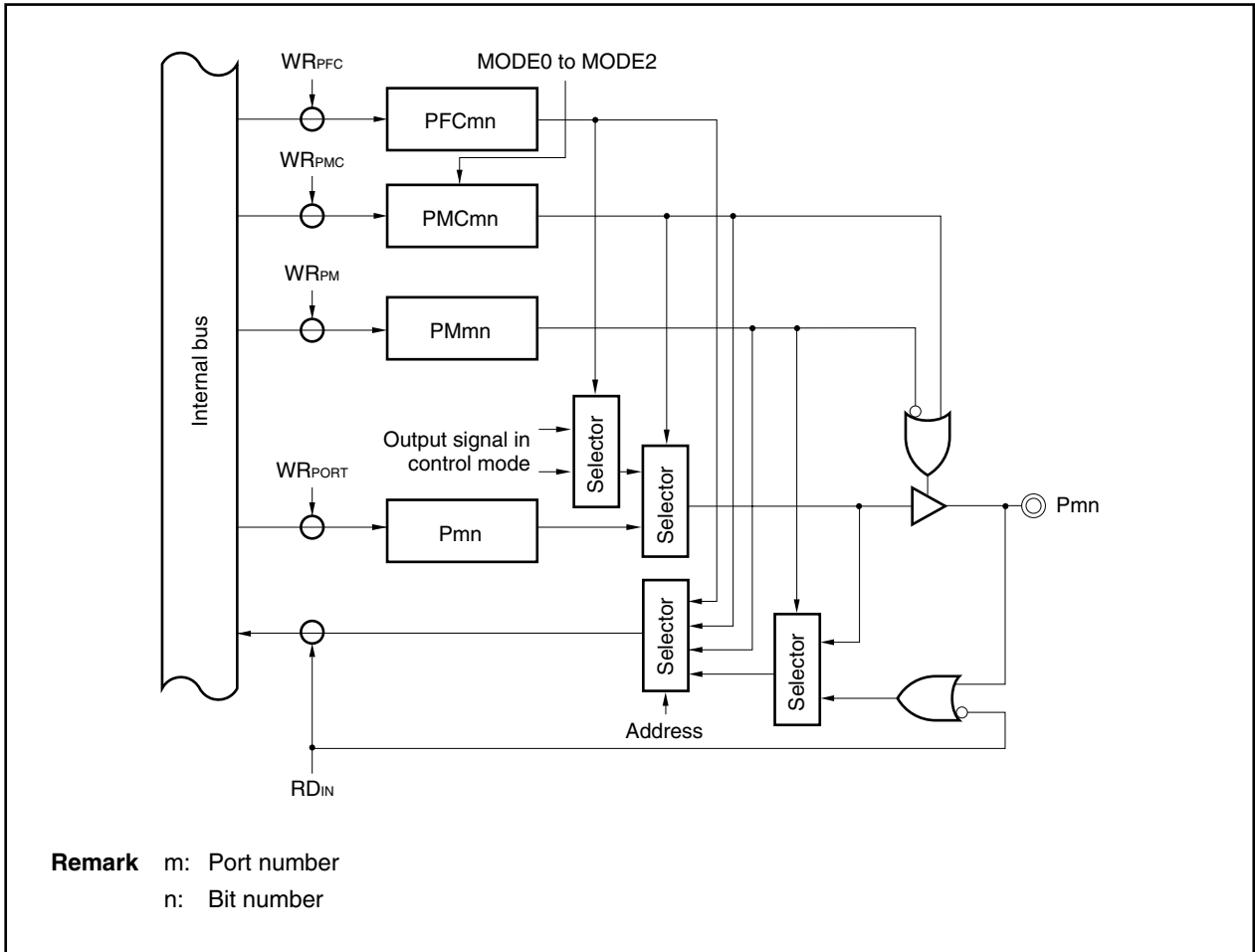
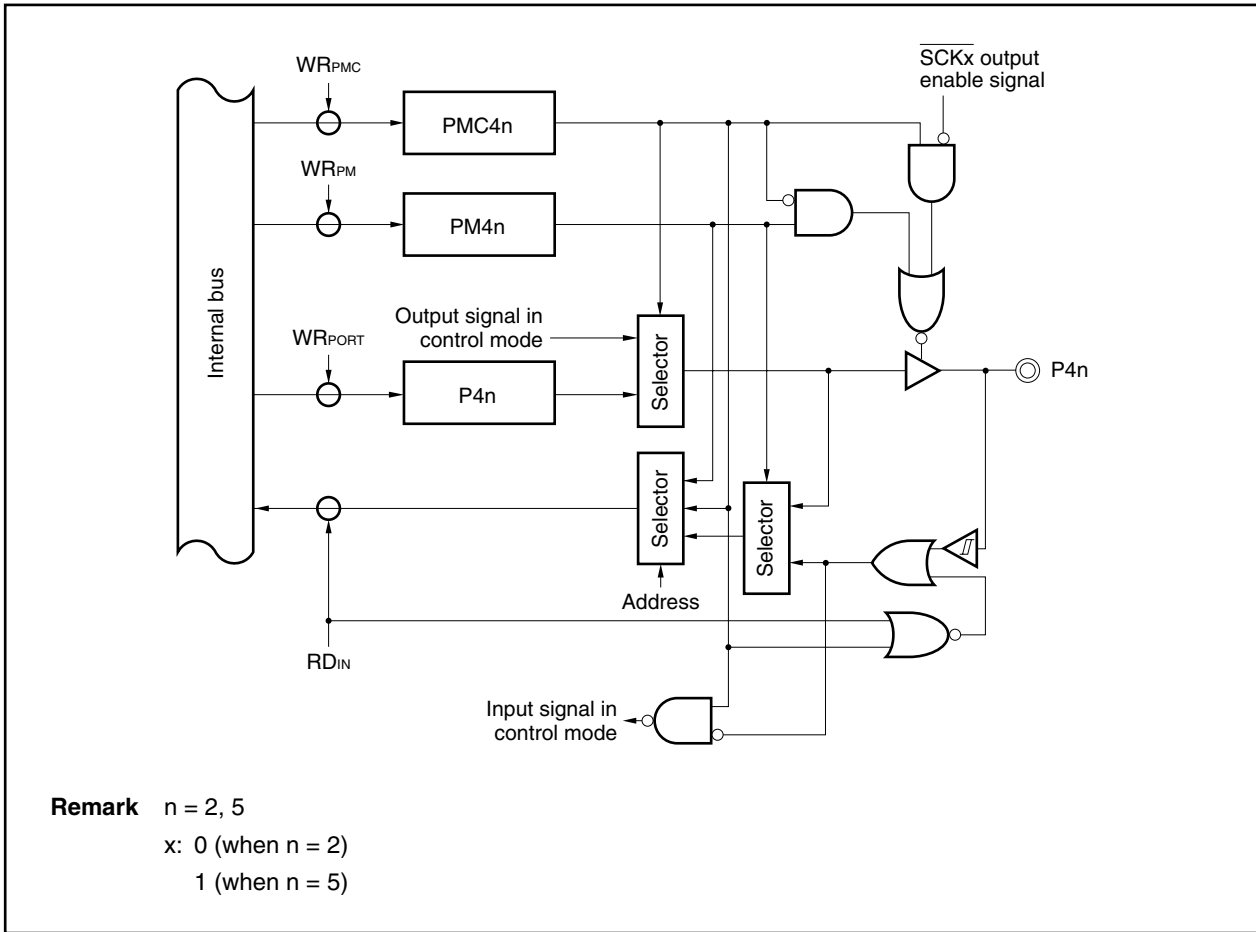


Figure 13-9. Block Diagram of Type K



★

Figure 13-10. Block Diagram of Type M



★

Figure 13-11. Block Diagram of Type N

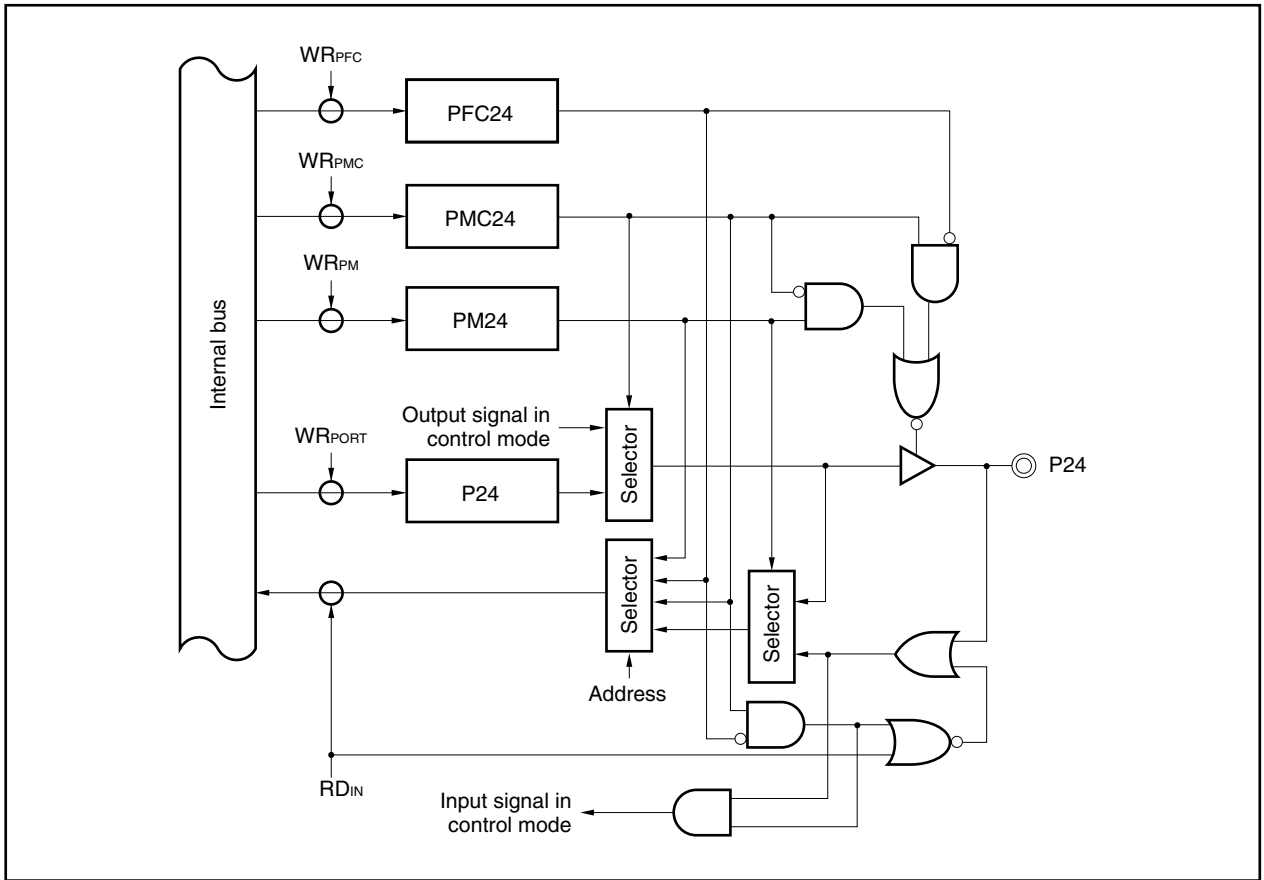
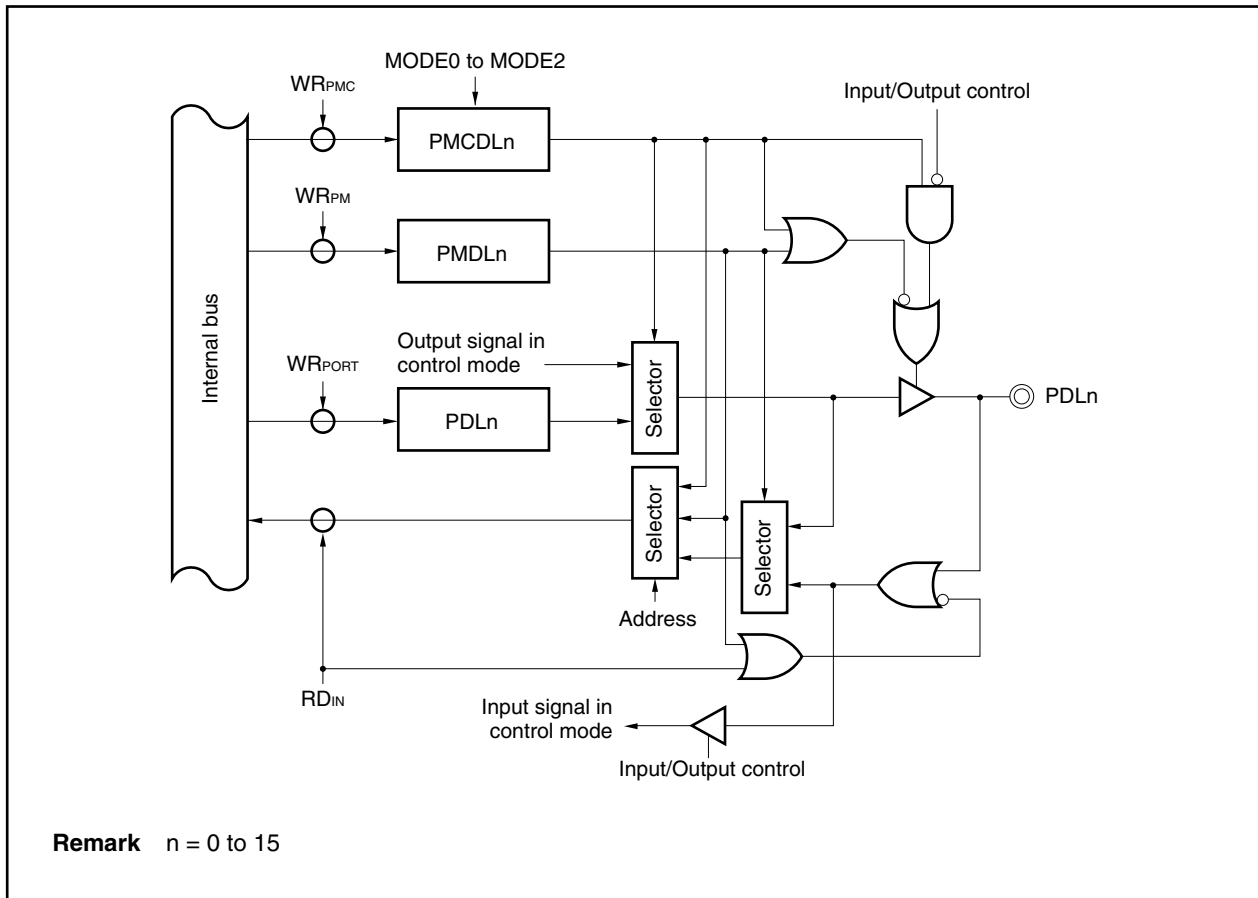


Figure 13-12. Block Diagram of Type O



13.3 Port Pin Functions

13.3.1 Port 0

Port 0 is a 5-bit I/O port that can be set to the input or output mode in 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
P0	-	-	P05	P04	P03	P02	P01	-	FFFF400H	Undefined

Bit Position	Bit Name	Function
5 to 1	P0n (n = 5 to 1)	Port 0 I/O 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	Alternate Function	Remark	Block Type	
Port 0	P01	INTP000/TI000	B	
	P02	INTP001		External interrupt request input
	P03	TO00	Real-time pulse unit (RPU) output	A
	P04, P05	$\overline{\text{DMARQ0}}/\text{INTP100},$ $\overline{\text{DMARQ1}}/\text{INTP101}$	DMA request input/ external interrupt request input	H

(2) I/O mode/control mode setting

The port 0 I/O mode setting is performed by means of the port 0 mode register (PM0), and the control mode setting is performed by means of the port 0 mode control register (PMC0) and the port 0 function control register (PFC0).

(a) Port 0 mode register (PM0)

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

	7	6	5	4	3	2	1	0	Address	After reset
PM0	1	1	PM05	PM04	PM03	PM02	PM01	1	FFFF420H	FFH

Bit Position	Bit Name	Function
5 to 1	PM0n (n = 5 to 1)	Port Mode Specifies input/output mode for P0n pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(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	Address	After reset
PMC0	0	0	PMC05	PMC04	PMC03	PMC02	PMC01	0	FFFF440H	00H

Bit Position	Bit Name	Function
5, 4	PMC0n (n = 5, 4)	Port Mode Control Specifies operation mode of P0n pin in combination with the PFC0 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP101}}$, $\overline{\text{INTP100}}$) input mode/DMA request ($\overline{\text{DMARQ1}}$, $\overline{\text{DMARQ0}}$) input mode
3	PMC03	Port Mode Control Specifies operation mode of P03 pin. 0: I/O port mode 1: TO00 output mode
2	PMC02	Port Mode Control Specifies operation mode of P02 pin. 0: I/O port mode 1: External interrupt request (INTP001) input mode
1	PMC01	Port Mode Control Specifies operation mode of P01 pin. 0: I/O port mode 1: External interrupt request (INTP000) input mode/TI000 input mode There is no register that switches between the external interrupt request (INTP000) input mode and TI000 input mode <ul style="list-style-type: none"> • When TI000 input mode is selected: Mask the external interrupt request (INTP000) or specify the CCC00 register as compare register. • When external interrupt request (INTP000) input mode (including timer capture input) is selected: Set the ETI0 bit of the TMCC01 register to 0.

★

(c) Port 0 function control register (PFC0)

This register can be read/written in 8- or 1-bit units. Bits 7, 6, and 3 to 0, however, are fixed to 0, so writing 1 to these bits is ignored. However, only bits 3 to 0 are fixed to 0 on the in-circuit emulator, and the values written to bits 7 and 6 are reflected.

Caution When the port mode is specified by the port 0 mode control register (PMC0), the PFC0 setting becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC0	0	0	PFC05	PFC04	0	0	0	0	FFFFF460H	00H

Bit Position	Bit Name	Function
5, 4	PFC0n (n = 5, 4)	Port Function Control Specifies operation mode of P0n pin in control mode. 0: External interrupt request (INTP101, INTP100) input mode 1: DMA (DMARQ1, DMARQ0) request input mode

13.3.2 Port 1

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

	7	6	5	4	3	2	1	0	Address	After reset
P1	-	-	-	-	-	P12	P11	-	FFFFF402H	Undefined

Bit Position	Bit Name	Function
2, 1	P1n (n = 2, 1)	Port 1 I/O port

In addition to their function as port pins, the port 1 pins can also operate as real-time pulse unit (RPU) inputs and external interrupt request inputs in the control mode.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type
Port 1	P11	TI010/INTP010	B
	P12	INTP011	

(2) I/O mode/control mode setting

The port 1 I/O mode setting is performed by means of the port 1 mode register (PM1), and the control mode setting is performed by means of the port 1 mode control register (PMC1).

(a) Port 1 mode register (PM1)

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

	7	6	5	4	3	2	1	0	Address	After reset
PM1	1	1	1	1	1	PM12	PM11	1	FFFFF422H	FFH

Bit Position	Bit Name	Function
2, 1	PM1n (n = 2, 1)	Port Mode Specifies input/output mode for P1n pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(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	Address	After reset
PMC1	0	0	0	0	0	PMC12	PMC11	0	FFFFF442H	00H

Bit Position	Bit Name	Function
2	PMC12	Port Mode Control Specifies operation mode of P12 pin. 0: I/O port mode 1: External interrupt request (INTP011) input mode
1	PMC11	Port Mode Control Specifies operation mode of P11 pin. 0: I/O port mode 1: External interrupt request (INTP010) input mode/TI010 input mode There is no register that switches between the external interrupt request (INTP010) input mode and TI010 input mode. <ul style="list-style-type: none"> • When the TI010 input mode is selected: Mask the external interrupt (INTP010) or specify the CCC10 register as compare register. • When external interrupt request (INTP010) input mode (including timer capture input) is selected: Set the ETI1 bit of the TMCC11 register to 0.

13.3.3 Port 2

P20 of port 2 is an input-only port and P24 is an I/O port.

Caution P20 is fixed to NMI input. The level of the NMI input can be read regardless of the PM2 and PMC2 registers' value.

	7	6	5	4	3	2	1	0	Address	After reset
P2	-	-	-	P24	-	-	-	P20	FFFFFF404H	Undefined

Bit Position	Bit Name	Function
4, 0	P2n (n = 4, 0)	Port 2 I/O port

In addition to their function as port pins, the port 2 pins can also operate as the external interrupt request inputs and the DMA end (terminal count) signal outputs in the control mode.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type
Port 2	P20	NMI	Non-maskable interrupt request input F
	P24	$\overline{TC0}/\overline{INTP110}$	DMA end signal outputs/External interrupt request inputs N

(2) I/O mode/control mode setting

The port 2 I/O mode setting is performed by means of the port 2 mode register (PM2), and the control mode setting is performed by means of the port 2 mode control register (PMC2) and the port 2 function control register (PFC2).

(a) Port 2 mode register (PM2)

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

	7	6	5	4	3	2	1	0	Address	After reset
PM2	1	1	1	PM24	1	1	1	1	FFFFFF424H	FFH

Bit Position	Bit Name	Function
4	PM24	Port Mode Specifies input/output mode for P24 pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port 2 mode control register (PMC2)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMC2	0	0	0	PMC24	0	0	0	1	FFFFF444H	01H

Bit Position	Bit Name	Function
4	PMC24	Port Mode Control Specifies operation mode of P24 pin in combination with the PFC2 register. 0: I/O port mode 1: External input request ($\overline{\text{INTP110}}$) input mode/ DMA end signal ($\overline{\text{TC0}}$) output mode

★

(c) Port 2 function control register (PFC2)

This register can be read/written in 8- or 1-bit units. Bits 7 to 5 and 3 to 0, however, are fixed to 0 by hardware, so writing 1 to this bit is ignored. However, only bits 3 to 0 are fixed to 0 on the in-circuit emulator, and the values written to bits 7 to 5 are reflected.

Caution When the port mode is specified by the port 2 mode control register (PMC2), the PFC2 setting becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC2	0	0	0	PFC24	0	0	0	0	FFFFF464H	00H

Bit Position	Bit Name	Function
4	PFC24	Port Function Control Specifies operation mode of P24 pin in control mode. 0: External interrupt request ($\overline{\text{INTP110}}$) input mode 1: DMA end signal ($\overline{\text{TC0}}$) output mode

13.3.4 Port 4

Port 4 is a 6-bit I/O port that can be set to the input or output mode in 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
P4	-	-	P45	P44	P43	P42	P41	P40	FFFFF408H	Undefined

Bit Position	Bit Name	Function
5 to 0	P4n (n = 5 to 0)	Port 4 I/O port

In addition to their function as port pins, the port 4 pins can also operate as the serial interface (UART0/CSI0, UART1/CSI1) I/O in the control mode.

(1) Operation in control mode

	Port	Alternate Function	Remark	Block Type	
★	Port 4	P40	TXD0/SO0	Serial interface (UART0/CSI0) I/O	G
		P41	RXD0/SI0		H
		P42	SCK0		M
★	Port 4	P43	TXD1/SO1	Serial interface (UART1/CSI1) I/O	G
		P44	RXD1/SI1		H
		P45	SCK1		M

(2) I/O mode/control mode setting

The port 4 I/O mode setting is performed by means of the port 4 mode register (PM4), and the control mode setting is performed by means of the port 4 mode control register (PMC4) and the port 4 function control register (PFC4).

(a) Port 4 mode register (PM4)

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

	7	6	5	4	3	2	1	0	Address	After reset
PM4	1	1	PM45	PM44	PM43	PM42	PM41	PM40	FFFFF428H	FFH

Bit Position	Bit Name	Function
5 to 0	PM4n (n = 5 to 0)	Port Mode Specifies input/output mode for P4n pin 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port 4 mode control register (PMC4)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMC4	0	0	PMC45	PMC44	PMC43	PMC42	PMC41	PMC40	FFFFF448H	00H

Bit Position	Bit Name	Function
5	PMC45	Port Mode Control Specifies operation mode of P45 pin. 0: I/O port mode 1: SCK1 input/output mode
4	PMC44	Port Mode Control Specifies operation mode of P44 pin. 0: I/O port mode 1: RXD1/SI1 input mode
3	PMC43	Port Mode Control Specifies operation mode of P43 pin. 0: I/O port mode 1: TXD1/SO1 output mode
2	PMC42	Port Mode Control Specifies operation mode of P42 pin. 0: I/O port mode 1: SCK0 input/output mode
1	PMC41	Port Mode Control Specifies operation mode of P41 pin. 0: I/O port mode 1: RXD0/SI0 input mode
0	PMC40	Port Mode Control Specifies operation mode of P40 pin. 0: I/O port mode 1: TXD0/SO0 output mode

(c) Port 4 function control register (PFC4)

This register can be read/written in 8- or 1-bit units. Bits 7 to 5 and 2, however, are fixed to 0, so writing 1 to these bits is ignored.

Caution When the port mode is specified by the port 4 mode control register (PMC4), the PFC4 register setting becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC4	0	0	0	PFC44	PFC43	0	PFC41	PFC40	FFFFF468H	00H

Bit Position	Bit Name	Function
4	PFC44	Port Function Control Specifies operation mode of P44 pin in control mode. 0: SI1 input mode 1: RXD1 input mode
3	PFC43	Port Function Control Specifies operation mode of P43 pin in control mode. 0: SO1 output mode 1: TXD1 output mode
1	PFC41	Port Function Control Specifies operation mode of P41 pin in control mode. 0: SI0 input mode 1: RXD0 input mode
0	PFC40	Port Function Control Specifies operation mode of P40 pin in control mode. 0: SO0 output mode 1: TXD0 output mode

13.3.5 Port 7

Port 7 is a 4-bit input-only port whose pins are fixed to input.

	7	6	5	4	3	2	1	0	Address	After reset
P7	-	-	-	-	P73	P72	P71	P70	FFFFF40EH	Undefined

Bit Position	Bit Name	Function
3 to 0	P7n (n = 3 to 0)	Port 7 Input-only port

In addition to their function as port pins, the port 7 pins can also operate as the analog inputs to the A/D converter in the control mode.

(1) Operation in control mode

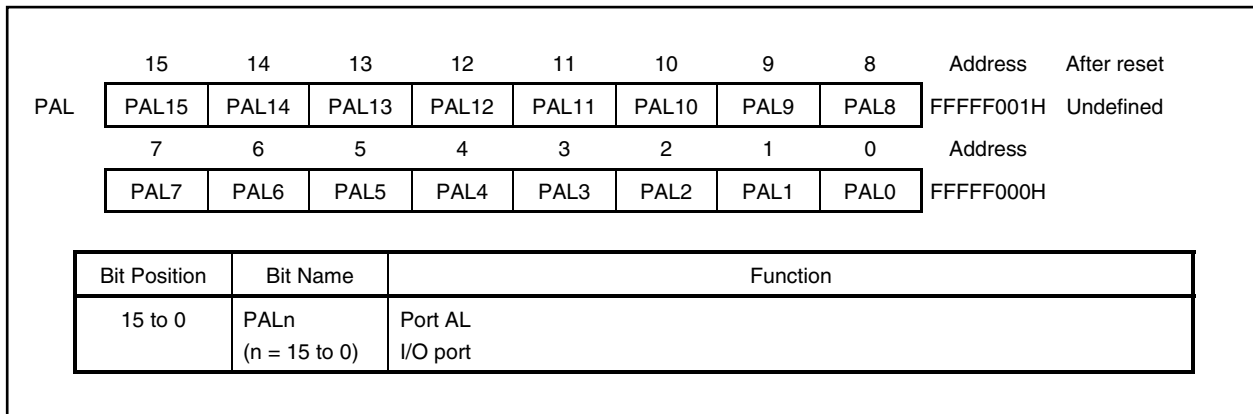
Port	Alternate Function	Remark	Block Type
Port 7	P73 to P70	ANI3 to ANI0	Analog input to A/D converter C

★ **Caution** When performing A/D conversion by selecting a pin from ANI0 to ANI3, the resolution of the A/D conversion may drop when port 7 (P7) is read during A/D conversion (ADCS bit of ADM0 register = 1).
 If a digital pulse is applied to the pin adjacent to the pin executing A/D conversion, the A/D conversion value may not be obtained as expected due to coupling noise. Do not apply a digital pulse to the pin adjacent to the pin executing A/D conversion.

13.3.6 Port AL

Port AL (PAL) is a 16-bit I/O port that can be set to the input or output mode in 1-bit units.

If the higher 8 bits of port AL are used as port ALH (PALH) and the lower 8 bits as port ALL (PALL), these 8-bit ports can be set in the input or output mode in 1-bit units.



In addition to their functions as port pins, in the control mode, the port AL pins operate as an address bus for when the memory is externally expanded.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type
Port AL	PAL15 to PAL0	A15 to A0 Address bus when memory expanded	J

(2) I/O mode/control mode setting

The port AL I/O mode setting is performed by means of the port AL mode register (PMAL), and control mode setting is performed by means of the port AL mode control register (PMCAL).

(a) Port AL mode register (PMAL)

The port AL mode register (PMAL) can be read/written in 16-bit units.

If the higher 8 bits of PMAL are used as a port AL mode register H (PMALH), and the lower 8 bits as a port AL mode register L (PMALL), these two 8-bit port mode registers can be read/written in 8- or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMAL	PMAL15	PMAL14	PMAL13	PMAL12	PMAL11	PMAL10	PMAL9	PMAL8	FFFFF021H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	PMAL7	PMAL6	PMAL5	PMAL4	PMAL3	PMAL2	PMAL1	PMAL0	FFFFF020H	

Bit Position	Bit Name	Function
15 to 0	PMALn (n = 15 to 0)	Port Mode Specifies input/output mode for PALn pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port AL mode control register (PMCAL)

The port AL mode control register (PMCAL) can be read/written in 16-bit units.

If the higher 8 bits of PMCAL are used as a port AL mode control register H (PMCALH), and the lower 8 bits as a port AL mode control register L (PMCALL), these two 8-bit port mode registers can be read/written in 8- or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMCAL	PMCAL15	PMCAL14	PMCAL13	PMCAL12	PMCAL11	PMCAL10	PMCAL9	PMCAL8	FFFFF041H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	PMCAL7	PMCAL6	PMCAL5	PMCAL4	PMCAL3	PMCAL2	PMCAL1	PMCAL0	FFFFF040H	

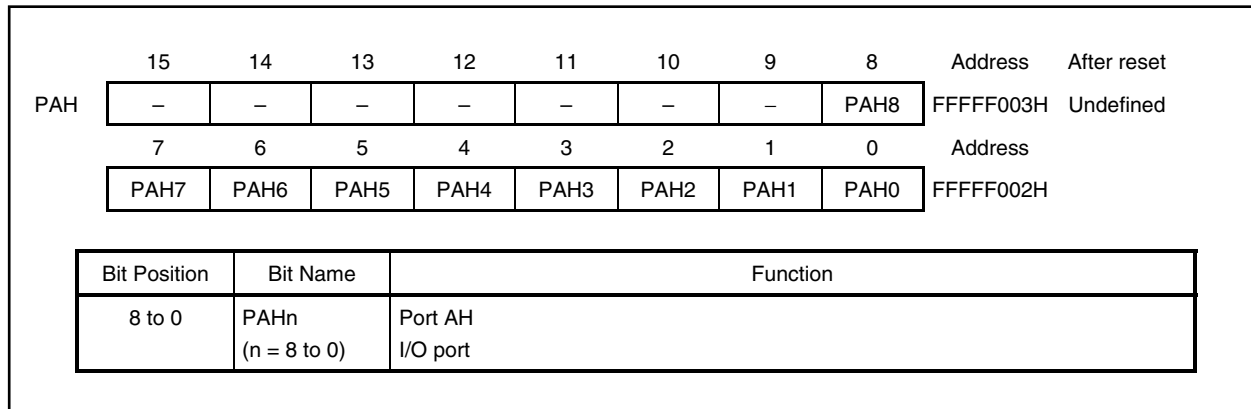
Bit Position	Bit Name	Function
15 to 0	PMCALn (n = 15 to 0)	Port Mode Control Specifies operation mode of PALn pin. 0: I/O port mode 1: A15 to A0 output mode

13.3.7 Port AH

Port AH (PAH) is a 16-bit I/O port that can be set in the input or output mode in 1-bit units.

If the higher 8 bits of port AH are used as port AHH (PAHH) and the lower 8 bits as port AHL (PAHL), these 8-bit ports can be set in the input or output mode in 1-bit units.

Bits 15 to 9 of port AH (bits 7 to 1 of port AHH) are undefined.



In addition to their functions as port pins, in the control mode, the port AH pins operate as an address bus for when the memory is externally expanded.

(1) Operation in control mode

Port	Alternate Function Pin Name	Remark	Block Type
Port AH	PAL8 to PAL0	A24 to A16 Address bus when memory expanded	J

(2) I/O mode/control mode setting

The port AH I/O mode setting is performed by means of the port AH mode register (PMAH), and the control mode setting is performed by means of the port AH mode control register (PMCAH).

(a) Port AH mode register (PMAH)

The port AH mode register (PMAH) can be read/written in 16-bit units.

If the higher 8 bits of PMAH are used as a port AH mode register H (PMAHH), and the lower 8 bits as a port AH mode register L (PMAHL), these two 8-bit port mode registers can be read/written in 8- or 1-bit units.

Bits 15 to 9 of PMAH (bits 7 to 1 of PMAHH) are fixed to 1.

	15	14	13	12	11	10	9	8	Address	After reset
PMAH	1	1	1	1	1	1	1	PMAH8	FFFFF023H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	PMAH7	PMAH6	PMAH5	PMAH4	PMAH3	PMAH2	PMAH1	PMAH0	FFFFF022H	

Bit Position	Bit Name	Function
8 to 0	PMAHn (n = 8 to 0)	Port Mode Specifies input/output mode for PAHn pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port AH mode control register (PMCAH)

The port AH mode control register (PMCAH) can be read/written in 16-bit units.

If the higher 8 bits of PMCAH are used as a port AH mode control register H (PMCAHH), and the lower 8 bits as a port AH mode control register L (PMCAHL), these two 8-bit port mode registers can be read/written in 8- or 1-bit units.

Bits 15 to 9 of PMCAH (bits 7 to 1 of PMCAHH) are fixed to 0.

	15	14	13	12	11	10	9	8	Address	After reset
PMCAH	0	0	0	0	0	0	0	PMDAH8	FFFFF043H	01FFH
	7	6	5	4	3	2	1	0	Address	
	PMDAH7	PMDAH6	PMDAH5	PMDAH4	PMDAH3	PMDAH2	PMDAH1	PMDAH0	FFFFF042H	

Bit Position	Bit Name	Function
8 to 0	PMDAHn (n = 8 to 0)	Port Mode Control Specifies operation mode of PAHn pin. 0: I/O port mode 1: A24 to A16 output mode

13.3.8 Port DL

Port DL (PDL) is a 16-bit I/O port that can be set in the input or output mode in 1-bit units.

If the higher 8 bits of port DL are used as port DLH (PDLH), and the lower 8 bits as port DLL (PDLL), these 8-bit ports can be set in the input or output mode in 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PDL	PDL15	PDL14	PDL13	PDL12	PDL11	PDL10	PDL9	PDL8	FFFFF005H	Undefined
	7	6	5	4	3	2	1	0	Address	
	PDL7	PDL6	PDL5	PDL4	PDL3	PDL2	PDL1	PDL0	FFFFF004H	

Bit Position	Bit Name	Function
15 to 0	PDLn (n = 15 to 0)	Port DL I/O port

In addition to their functions as port pins, in the control mode, the port DL pins operate as a data bus for when the memory is externally expanded.

(1) Operation in control mode

Port	Alternate Function Pin Name	Remark	Block Type
Port DL	PDL15 to PDL0	D15 to D0	Data bus when memory expanded O

(2) I/O mode/control mode setting

The port DL I/O mode setting is performed by means of the port DL mode register (PMDL), and the control mode setting is performed by means of the port DL mode control register (PMCDL).

(a) Port DL mode register (PMDL)

The port DL mode register (PMDL) can be read/written in 16-bit units.

If the higher 8 bits of PMDL are used as a port DL mode register H (PMDLH), and the lower 8 bits as a port DL mode register L (PMDLL), these two 8-bit port mode registers can be read/written in 8- or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMDL	PMDL15	PMDL14	PMDL13	PMDL12	PMDL11	PMDL10	PMDL9	PMDL8	FFFFF025H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	PMDL7	PMDL6	PMDL5	PMDL4	PMDL3	PMDL2	PMDL1	PMDL0	FFFFF024H	

Bit Position	Bit Name	Function
15 to 0	PMDLn (n = 15 to 0)	Port Mode Specifies input/output mode for PDLn pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port DL mode control register (PMCDL)

The port DL mode control register (PMCDL) can be read/written in 16-bit units.

If the higher 8 bits of PMCDL are used as a port DL mode control register H (PMCDLH), and the lower 8 bits as a port DL mode control register L (PMCDLL), these two 8-bit port mode registers can be read/written in 8- or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMCDL	PMCDL15	PMCDL14	PMCDL13	PMCDL12	PMCDL11	PMCDL10	PMCDL9	PMCDL8	FFFFF045H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	PMCDL7	PMCDL6	PMCDL5	PMCDL4	PMCDL3	PMCDL2	PMCDL1	PMCDL0	FFFFF044H	

Bit Position	Bit Name	Function
15 to 0	PMCDLn (n = 15 to 0)	Port Mode Control Specifies operation mode of PDLn pin. 0: I/O port mode 1: D15 to D0 output mode

★ **Caution** The D8 to D15 pins are in the input status in ROMless mode 1.

13.3.9 Port CS

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

	7	6	5	4	3	2	1	0	Address	After reset
PCS	PCS7	-	-	PCS4	PCS3	-	-	PCS0	FFFFF008H	Undefined

Bit Position	Bit Name	Function
7, 4, 3, 0	PCS _n (n = 7, 4, 3, 0)	Port CS I/O port

In addition to their function as port pins, in the control mode, the port pins can also operate as the chip select signal outputs when externally expanding memory.

(1) Operation in control mode

Port	Alternate Function Pin Name	Remark	Block Type
Port CS PCS0, PCS3, PCS4, PCS7	$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$	Chip select signal output	J

(2) I/O mode/control mode setting

The port CS I/O mode setting is performed by means of the port CS mode register (PMCS), and the control mode setting is performed by means of the port CS mode control register (PMCCS).

(a) Port CS mode register (PMCS)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMCS	PMCS7	1	1	PMCS4	PMCS3	1	1	PMCS0	FFFFF028H	FFH

Bit Position	Bit Name	Function
7, 4, 3, 0	PMCS _n (n = 7, 4, 3, 0)	Port Mode Specifies input/output mode for PCS _n pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port CS mode control register (PMCCS)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMCCS	PMCCS7	0	0	PMCCS4	PMCCS3	0	0	PMCCS0	FFFFF048H	99H

Bit Position	Bit Name	Function
7, 4, 3, 0	PMCCSn (n = 7, 4, 3, 0)	Port Mode Control Specifies operation mode of PCSn pin. 0: I/O port mode 1: CSn output mode

13.3.10 Port CT

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

	7	6	5	4	3	2	1	0	Address	After reset
PCT	-	-	PCT5	PCT4	-	-	PCT1	PCT0	FFFFF00AH	Undefined

Bit Position	Bit Name	Function
5, 4, 1, 0	PCTn (n = 5, 4, 1, 0)	Port CT I/O port

In addition to their function as port pins, in the control mode, the port CT pins operate as control signal outputs for when the memory is externally expanded.

(1) Operation in control mode

Port	Alternate Function Pin Name	Remark	Block Type
Port CT	PCT0	$\overline{LWR}/LDQM$	J
	PCT1	$\overline{UWR}/UDQM$	
	PCT4	\overline{RD}	
	PCT5	\overline{WE}	

(2) I/O mode/control mode setting

The port CT I/O mode setting is performed by means of the port CT mode register (PMCT), and the control mode setting is performed by means of the port CT mode control register (PMCCT).

(a) Port CT mode register (PMCT)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMCT	1	1	PMCT5	PMCT4	1	1	PMCT1	PMCT0	FFFFF02AH	FFH

Bit Position	Bit Name	Function
5, 4, 1, 0	PMCTn (n = 5, 4, 1, 0)	Port Mode Specifies input/output mode for PCTn pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port CT mode control register (PMCCT)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMCCT	0	0	PMCCT5	PMCCT4	0	0	PMCCT1	PMCCT0	FFFFF04AH	33H

Bit Position	Bit Name	Function
5	PMCCT5	Port Mode Control Specifies operation mode of PCT5 pin. 0: I/O port mode 1: WE output mode
4	PMCCT4	Port Mode Control Specifies operation mode of PCT4 pin. 0: I/O port mode 1: RD output mode
1	PMCCT1	Port Mode Control Specifies operation mode of PCT1 pin. 0: I/O port mode 1: UWR/UDQM output mode (UWR/UDQM signal automatically switched by accessing the targeted memory of each signal.)
0	PMCCT0	Port Mode Control Specifies operation mode of PCT0 pin. 0: I/O port mode 1: LWR/LDQM output mode (LWR/LDQM signal automatically switched by accessing the targeted memory of each signal.)

13.3.11 Port CM

Port CM is a 5-bit I/O port that can be set to the input or output mode in 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
PCM	-	-	-	PCM4	PCM3	PCM2	PCM1	PCM0	FFFFFF0CH	Undefined

Bit Position	Bit Name	Function
4 to 0	PCMn (n = 4 to 0)	Port CM I/O port

In addition to their function as port pins, in the control mode, the port CM pins operate as the wait insertion signal input, internal system clock output, bus hold control signal output, and refresh request signal output from DRAM.

(1) Operation in control mode

	Port	Alternate Function Pin Name	Remark	Block Type	
★	Port CM	PCM0	$\overline{\text{WAIT}}$	Wait insertion signal input	D
		PCM1	CLKOUT	Internal system clock output	J
		PCM2	$\overline{\text{HLDAK}}$	Bus hold acknowledge signal output	J
		PCM3	$\overline{\text{HLDRQ}}$	Bus hold request signal input	D
		PCM4	$\overline{\text{REFRQ}}$	Refresh request signal output	J

(2) I/O mode/control mode setting

The port CM I/O mode setting is performed by means of the port CM mode register (PMCM), and the control mode setting is performed by means of the port CM mode control register (PMCCM).

(a) Port CM mode register (PMCM)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMCM	1	1	1	PMCM4	PMCM3	PMCM2	PMCM1	PMCM0	FFFFFF02CH	FFH

Bit Position	Bit Name	Function
4 to 0	PMCMn (n = 4 to 0)	Port Mode Specifies input/output mode for PCMn pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port CM mode control register (PMCCM)

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

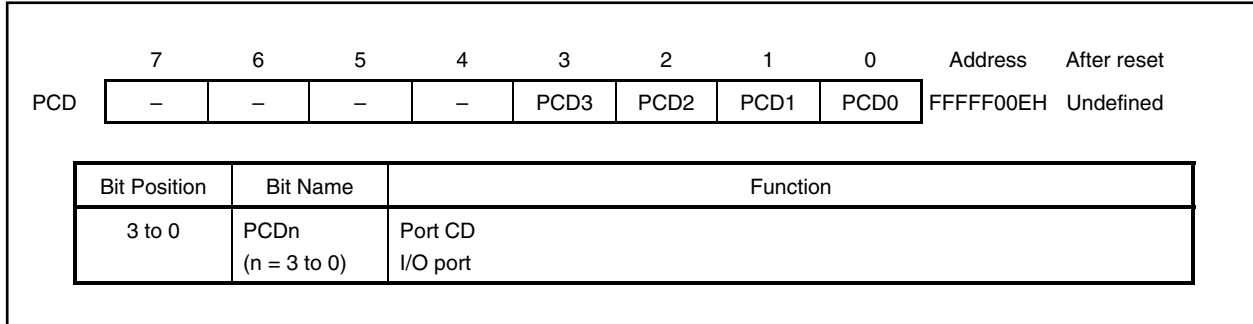
Caution If the mode of the PCM1/CLKOUT pin is changed from the I/O port mode to the CLKOUT mode, a glitch may be generated on the CLKOUT output immediately after the change. Therefore, pull up the CLKOUT pin when using it. In the PLL mode (CKSEL = 0), change the mode to the CLKOUT output mode at a multiple of 1 (CKDIV2 to CKDIV0 bits of CKC register = 000B).

	7	6	5	4	3	2	1	0	Address	After reset
PMCCM	0	0	0	PMCCM4	PMCCM3	PMCCM2	PMCCM1	PMCCM0	FFFFFF04CH	1FH

Bit Position	Bit Name	Function
4	PMCCM4	Port Mode Control Specifies operation mode of PCM4 pin. 0: I/O port mode 1: REFRQ output mode
3	PMCCM3	Port Mode Control Specifies operation mode of PCM3 pin. 0: I/O port mode 1: HLDRQ input mode
2	PMCCM2	Port Mode Control Specifies operation mode of PCM2 pin. 0: I/O port mode 1: HLDAK output mode
1	PMCCM1	Port Mode Control Specifies operation mode of PCM1 pin. 0: I/O port mode 1: CLKOUT output mode
0	PMCCM0	Port Mode Control Specifies operation mode of PCM0 pin. 0: I/O port mode 1: WAIT input mode

13.3.12 Port CD

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



In addition to their function as port pins, the port CD pins operate as the clock enable signal output to SDRAM, synchronous clock output, column address strobe signal output, row address strobe signal output, and byte enable signal output to SDRAM upon byte access, in the control mode.

(1) Operation in control mode

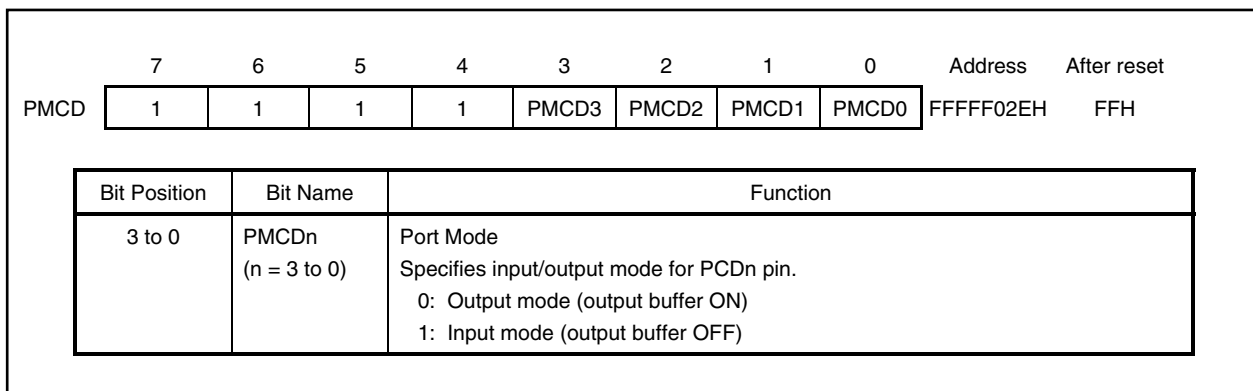
Port	Alternate Function Pin Name	Remark	Block Type
Port CD	PCD0	SDCKE	J
	PCD1	SDCLK	
	PCD2	$\overline{\text{LBE/SDCAS}}$	K
	PCD3	$\overline{\text{UBE/SDRAS}}$	

(2) I/O mode/control mode setting

The port CD I/O mode setting is performed by means of the port CD mode register (PMCD), and the control mode setting is performed by means of the port CD mode control register (PMCCD) and the port CD function control register (PFCCD).

(a) Port CD mode register (PMCD)

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



(b) Port CD mode control register (PMCCD)

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

- Cautions**
1. Do not perform the SDCLK and SDCKE output mode setting simultaneously. Be sure to perform the SDCLK output mode setting before the SDCKE output mode setting.
 2. Bits 1 and 0 of the PMCCD register become SDCLK output mode and SDCKE output mode after the reset is released, however, bits 3 and 2 become \overline{UBE} output mode and \overline{LBE} output mode. When using SDRAM be sure to set the \overline{SDRAS} output mode and \overline{SDCAS} output mode using the PFCCD register.

	7	6	5	4	3	2	1	0	Address	After reset
PMCCD	0	0	0	0	PMCCD3	PMCCD2	PMCCD1	PMCCD0	FFFFF04EH	0FH

Bit Position	Bit Name	Function
3	PMCCD3	Port Mode Control Specifies operation mode of PCD3 pin. 0: $\overline{I/O}$ port mode 1: $\overline{UBE/SDRAS}$ output mode
2	PMCCD2	Port Mode Control Specifies operation mode of PCD2 pin. 0: $\overline{I/O}$ port mode 1: $\overline{LBE/SDCAS}$ output mode
1	PMCCD1	Port Mode Control Specifies operation mode of PCD1 pin. 0: $\overline{I/O}$ port mode 1: SDCLK output mode
0	PMCCD0	Port Mode Control Specifies operation mode of PCD0 pin. 0: $\overline{I/O}$ port mode 1: SDCKE output mode

(c) Port CD function control register (PFCCD)

This register can be read/written in 8- or 1-bit units. Bits 7 to 4, 1, and 0, however, are fixed to 0, so writing 1 to these bits is ignored.

Caution When the port mode is specified by the port CD mode control register (PMCCD), the PFCCD setting becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFCCD	0	0	0	0	PFCCD3	PFCCD2	0	0	FFFFFF04FH	00H

Bit Position	Bit Name	Function
3	PFCCD3	Port Function Control Specifies operation mode of PCD3 pin in control mode. 0: \overline{UBE} output mode 1: \overline{SDRAS} output mode
2	PFCCD2	Port Function Control Specifies operation mode of PCD2 pin in control mode. 0: \overline{LBE} output mode 1: \overline{SDCAS} output mode

13.3.13 Port BD

Port BD is a 2-bit I/O port that can be set to the input or output mode in 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
PBD	-	-	-	-	-	-	PBD1	PBD0	FFFFFF012H	Undefined

Bit Position	Bit Name	Function
1, 0	PBDn (n = 1, 0)	Port BD I/O port

In addition to their function as port pins, the port BD pins operate as the DMA acknowledge signal outputs in the control mode.

(1) Operation in control mode

Port	Alternate Function Pin Name	Remark	Block Type
Port BD PBD0, PBD1	DMAAK0, DMAAK1	DMA acknowledge signal output	J

(2) I/O mode/control mode setting

The port BD I/O mode setting is performed by means of the port BD mode register (PMBD), and the control mode setting is performed by means of the port BD mode control register (PM CBD).

(a) Port BD mode register (PMBD)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMBD	1	1	1	1	1	1	PMBD1	PMBD0	FFFFFF032H	FFH

Bit Position	Bit Name	Function
1, 0	PMBDn (n = 1, 0)	Port Mode Specifies input/output mode for PBDn pin. 0: Output mode (output buffer ON) 1: Input mode (output buffer OFF)

(b) Port BD mode control register (PMCBD)

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

	7	6	5	4	3	2	1	0	Address	After reset
PMCBD	0	0	0	0	0	0	PMCBD1	PMCBD0	FFFFF052H	00H

Bit Position	Bit Name	Function
1, 0	PMCBDn (n = 1, 0)	Port Mode Control Specifies operation mode of PBDn pin. 0: I/O port mode 1: DMAAKn output mode

CHAPTER 14 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 CPU starts program execution. Register contents must be initialized as required in the program.

14.1 Features

The reset pin ($\overline{\text{RESET}}$) incorporates a noise eliminator that uses analog delay (≈ 60 ns) to prevent malfunction due to noise.

14.2 Pin Functions

During a system reset, most pins (all but the CLKOUT, $\overline{\text{RESET}}$, X2, V_{DD}, V_{SS}, CV_{DD}, CV_{SS}, AV_{DD}, AV_{REF}, and AV_{SS} 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 AL, AH, DL, CS, CT, CM, CD, and BD. If no resistor is connected there, external memory may be destroyed when these pins enter the high impedance state.

For the same reason, the output pins of the internal peripheral I/O functions and other output ports should be handled in the same manner.

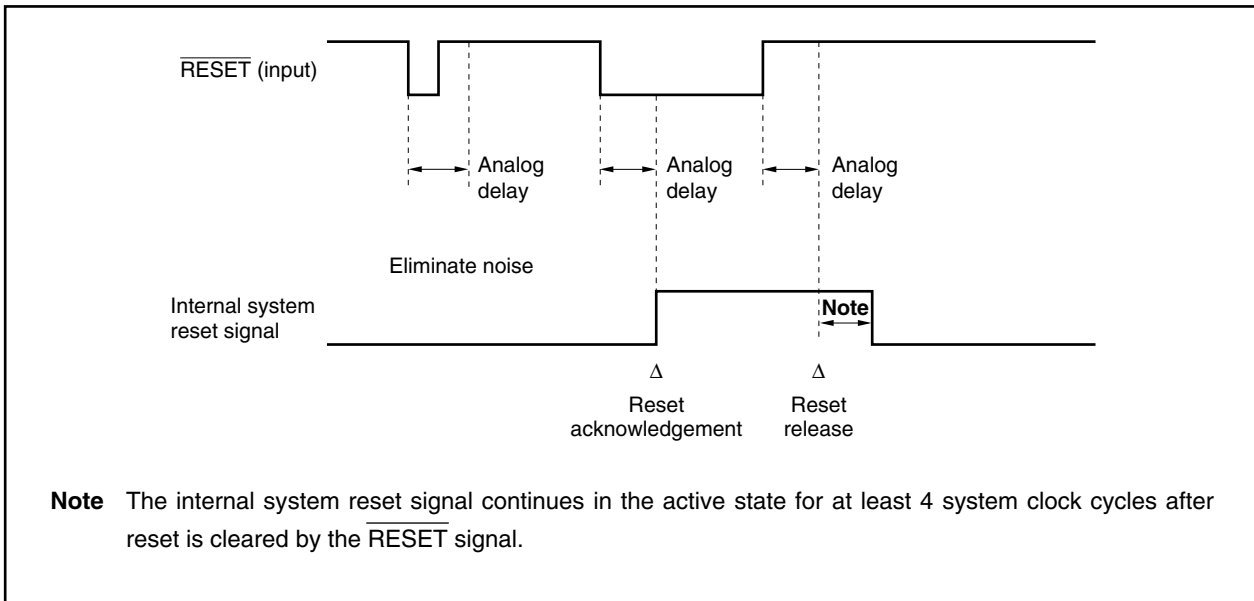
The operation status of each pin during reset is shown below (Table 14-1).

Table 14-1. Operation Status of Each Pin During Reset

Pin Name		Pin State
A0 to A15, A16 to A24, D0 to D15, $\overline{\text{CS0}}$, $\overline{\text{CS3}}$, $\overline{\text{CS4}}$, $\overline{\text{CS7}}$, $\overline{\text{LWR}}$, $\overline{\text{UWR}}$, $\overline{\text{LDQM}}$, $\overline{\text{UDQM}}$, $\overline{\text{RD}}$, $\overline{\text{WE}}$, $\overline{\text{WAIT}}$, $\overline{\text{HLDAK}}$, $\overline{\text{HLDRQ}}$, $\overline{\text{REFREQ}}$, $\overline{\text{SDCKE}}$, $\overline{\text{SDCLK}}$, $\overline{\text{LBE}}$, $\overline{\text{UBE}}$, $\overline{\text{SDCAS}}$, $\overline{\text{SDRAS}}$		High impedance
CLKOUT		Operating
Port pin	Ports 0 to 2, 4, 7, BD	(Input)
	Ports AL, AH, DL, CM, CT, CS, CD	(Control mode)

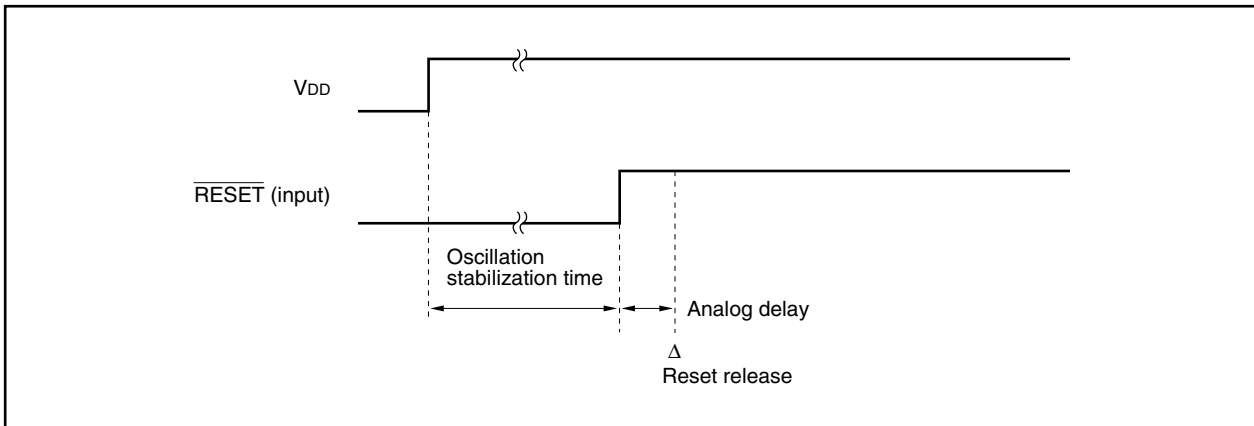
★

(1) Receiving the reset signal



(2) Reset when turning on the power

In a reset operation when the power is turned on, because of the low-level width of the $\overline{\text{RESET}}$ signal, it is necessary to secure the oscillation stabilization time between when the power is turned on and when the reset is acknowledged.



14.3 Initialization

Initialize the contents of each register as necessary while programming.

The initial values of the CPU, internal RAM, and on-chip peripheral I/O after a reset are shown in Table 14-2.

Table 14-2. Initial Value of CPU, Internal RAM, and On-Chip Peripheral I/O After Reset (1/3)

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 registers during interrupt (EIPC, EIPSW)	Undefined
		Status saving registers during NMI (FEPC, FEPSW)	Undefined
		Interrupt source register (ECR)	00000000H
		Program status word (PSW)	00000020H
		Status saving registers during CALLT execution (CTPC, CTPSW)	Undefined
		Status saving registers during exception/debug trap (DBPC, DBPSW)	Undefined
CALLT base pointer (CTBP)		Undefined	
Internal RAM		—	Undefined
On-chip peripheral I/O	Port functions	Ports (P0 to P2, P4, P7, PAL, PAH, PDL, PCS, PCT, PCM, PCD, PBD)	Undefined
		Mode registers (PM0 to PM2, PM4, PMCS, PMCT, PMCM, PMCD, PMBD)	FFH
		Mode registers (PMAL, PMAH, PMDL)	FFFFH
		Mode control registers (PMC0, PMC1, PMC4, PMCBD)	00H
		Mode control register (PMC2)	01H
		Mode control registers (PMCAL, PMCDL)	FFFFH
		Mode control register (PMCAH)	01FFH
		Mode control register (PMCCS)	99H
		Mode control register (PMCCT)	33H
		Mode control register (PMCCM)	1FH
		Mode control register (PMCCD)	0FH
		Function control registers (PFC0, PFC2, PFC4, PFCCD)	00H
		Timer/counter functions	Timer Cn (TMCn) (n = 0, 1)
	Capture/compare registers Cn0 and Cn1 (CCn0, CCn1) (n = 0, 1)		0000H
	Timer mode control register Cn0 (TMCCn0) (n = 0, 1)		00H
	Timer mode control register Cn1 (TMCCn1) (n = 0, 1)		20H
	Timer Dn (TMDn) (n = 0 to 3)		0000H
	Compare register (CMDn) (n = 0 to 3)		0000H
	Timer mode control register Dn (n = 0 to 3)		00H

★

Table 14-2. Initial Value of CPU, Internal RAM, and On-Chip Peripheral I/O After Reset (2/3)

Internal Hardware		Register Name	Initial Value After Reset
On-chip peripheral I/O	Serial interface functions	Clocked serial interface mode register n (CSIMn) (n = 0, 1)	00H
		Clocked serial interface clock select register n (CSICn) (n = 0, 1)	00H
		Clocked serial interface transmit buffer register n (SOTBn) (n = 0, 1)	00H
		Serial I/O shift register n (SIO _n) (n = 0, 1)	00H
		Receive-only serial I/O shift register n (SIOEn) (n = 0, 1)	00H
		Receive buffer register n (RXBn) (n = 0, 1)	FFH
		Transmit buffer register n (TXBn) (n = 0, 1)	FFH
		Asynchronous serial interface mode register n (ASIMn) (n = 0, 1)	01H
		Asynchronous serial interface status register n (ASISn) (n = 0, 1)	00H
		Asynchronous serial interface transmit status register n (ASIFn) (n = 0, 1)	00H
		Clock selection register n (CKSRn) (n = 0, 1)	00H
		Baud rate generator control register n (BRGCn) (n = 0, 1)	FFH
		A/D converter	A/D converter mode registers 0 and 2 (ADM0, ADM2)
	A/D converter mode register 1 (ADM1)		07H
	A/D conversion result register n (ADCRn) (10 bits) (n = 0 to 3)		0000H
	A/D conversion result register nH (ADCRnH) (8 bits) (n = 0 to 3)		00H
	Interrupt/exception control functions	In-service priority register (ISPR)	00H
		External interrupt mode register n (INTMn) (n = 0 to 2)	00H
		Interrupt mask register n (IMRn) (n = 0 to 3)	FFFFH
		Valid edge selection register Cn (SESCn) (n = 0, 1)	00H
		Interrupt control registers (OVIC00, OVIC01, P00IC0, P00IC1, P01IC0, P01IC1, P10IC0, P10IC1, P11IC0, CMICD0 to CMICD3, DMAIC0 to DMAIC3, CSIIC0, CSIIC1, SEIC0, SEIC1, SRIC0, SRIC1, STIC0, STIC1, ADIC)	47H
	Memory control functions	Page ROM configuration register (PRC)	7000H
		SDRAM configuration register n (SCRn) (n = 3, 4)	0000H
		SDRAM refresh control register n (RFSn) (n = 3, 4)	0000H
	DMA functions	DMA addressing control register n (DADCn) (n = 0 to 3)	0000H
		DMA byte count register n (DBCn) (n = 0 to 3)	Undefined
		DMA channel control register n (DCHCn) (n = 0 to 3)	00H
		DMA destination address register nH (DDAnH) (n = 0 to 3)	Undefined
		DMA destination address register nL (DDAnL) (n = 0 to 3)	Undefined
		DMA disable status register (DDIS)	00H
		DMA restart register (DRST)	00H
		DMA source address register nH (DSAnH) (n = 0 to 3)	Undefined
		DMA source address register nL (DSAnL) (n = 0 to 3)	Undefined
		DMA terminal count output control register (DTCO)	01H
		DMA trigger source register n (DTFRn) (n = 0 to 3)	00H

Table 14-2. Initial Value of CPU, Internal RAM, and On-Chip Peripheral I/O After Reset (3/3)

Internal Hardware		Register Name	Initial Value After Reset
★ On-chip peripheral I/O	Bus control functions	Address setup wait control register (ASC)	FFFFH
		Bus cycle control register (BCC)	FFFFH
		Bus cycle type configuration register n (BCTn) (n = 0, 1)	8888H
		Endian configuration register (BEC)	0000H
		Bus size configuration register (BSC)	0000H/5555H
		Chip area selection control register n (CSCn) (n = 0, 1)	2C11H
		Data wait control register n (DWCn) (n = 0, 1)	7777H
	Power save control functions	Command register (PRCMD)	Undefined
		Power save control register (PSC)	00H
		Clock control register (CKC)	00H
		Power save mode register (PSMR)	00H
	System control	Peripheral command register (PHCMD)	Undefined
		Peripheral status register (PHS)	00H
		System wait control register (VSWC)	77H
		Lock register (LOCKR)	0xH

Caution “Undefined” in the above table is undefined after 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 $\overline{\text{RESET}} \downarrow$ signals, data is held in the same state it was in before the $\overline{\text{RESET}}$ operation.

Absolute Maximum Ratings (T_A = 25°C)

Parameter	Symbol	Conditions	Ratings	Unit
Power supply voltage	V _{DD}	V _{DD} pin	-0.5 to +4.6	V
	CV _{DD}	CV _{DD} pin	-0.5 to +4.6	V
	CV _{SS}	CV _{SS} pin	-0.5 to +0.5	V
	AV _{DD}	AV _{DD} pin	-0.5 to +4.6	V
	AV _{SS}	AV _{SS} pin	-0.5 to +0.5	V
Input voltage	V _I	Except X1 pin, V _I < V _{DD} + 3.0 V	-0.5 to V _{DD} + 0.5	V
Clock input voltage	V _K	X1, V _{DD} = 3.3 V ±0.3 V	-0.5 to V _{DD} + 1.0	V
Output current, low	I _{OL}	Per pin	4.0	mA
		Total of all pins	100	mA
Output current, high	I _{OH}	Per pin	-4.0	mA
		Total of all pins	-100	mA
Output voltage	V _O	V _{DD} = 3.3 V ±0.3 V	-0.5 to V _{DD} + 0.5	V
Analog input voltage	V _{WASN}	ANI0 to ANI3, V _{DD} = 3.3 V ±0.3 V, AV _{DD} < V _{DD} + 0.5 V	-0.3 to AV _{DD} + 0.3	V
Operating ambient temperature	T _A		-40 to +85	°C
Storage temperature	T _{stg}		-60 to +150	°C

Cautions 1. Avoid direct connections among the IC device output (or I/O) pins and between V_{DD} or V_{CC} and GND. However, direct connections among open-drain and open-collector pins are possible, as are direct connections to external circuits that have timing designed to prevent output conflict with pins that become high-impedance.

2. Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded. The ratings and conditions shown below for DC characteristics and AC characteristics are within the range for normal operation and quality assurance.

Capacitance (T_A = 25°C, V_{DD} = V_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input capacitance	C _I	f _C = 1 MHz			15	pF
I/O capacitance	C _{IO}	Unmeasured pins returned to 0 V.			15	pF
Output capacitance	C _O				15	pF

Operating Conditions

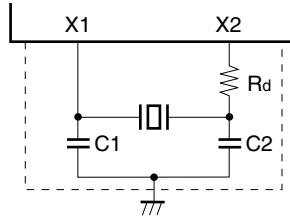
Operation Mode	Internal Operation Clock Frequency (f _{xx})	Operating Ambient Temperature (T _A)	Power Supply Voltage (V _{DD})
Direct mode	4 to 20 MHz	-40 to +85°C	V _{DD} = 3.3 V ±0.3 V
PLL mode ^{Note}	4 to 40 MHz	-40 to +85°C	V _{DD} = 3.3 V ±0.3 V

★ **Note** Set the input clock frequency (f_x) used in the PLL mode to 4.0 to 6.6 MHz. However, when inputting a frequency higher than 4.0 MHz, be sure to set the CKDIV2 to CKDIV0 bits of the clock control register (CKC) to other than 111 (10 × f_x).

Recommended Oscillator

(a) Ceramic resonator

(i) Kyocera Corporation ($T_A = -20$ to $+80^\circ\text{C}$)

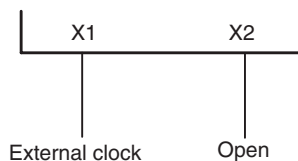


Type	Product	Oscillation Frequency f_x (MHz)	Recommended Circuit Constant			Oscillation Voltage Range		Oscillation Stabilization Time (MAX.) T_{OST} (ms)
			C1 (pF)	C2 (pF)	R_d (k Ω)	MIN. (V)	MAX. (V)	
Surface mount	PBRC4.00HR	4.0	On-chip	On-chip	0	3.0	3.6	0.34
	PBRC5.00HR ^{Note}	5.0	On-chip	On-chip	0	3.0	3.6	0.27
	PBRC6.00HR ^{Note}	6.0	On-chip	On-chip	0	3.0	3.6	0.33
Lead	KBR-4.0MKC	4.0	On-chip	On-chip	0	3.0	3.6	0.34
	KBR-5.0MKC ^{Note}	5.0	On-chip	On-chip	0	3.0	3.6	0.27
	KBR-6.0MKC ^{Note}	6.0	On-chip	On-chip	0	3.0	3.6	0.33

Note Use with a setting other than $\times 10$ multiplication.

- Cautions**
1. Connect the oscillator as close to the X1 and X2 pins as possible.
 2. Do not wire any other signal lines in the area indicated by the broken lines.
 3. Thoroughly evaluate the matching between the $\mu\text{PD703108}$ and the resonator.
 4. The oscillator constant is a reference value based on evaluation in specific environments by the resonator manufacturer. If the oscillator characteristics need to be optimized in the actual application, request the resonator manufacturer for evaluation on the implementation circuit. Note that the oscillation voltage and oscillation frequency merely indicate the characteristics of the oscillator. Use the internal operation conditions of the $\mu\text{PD703108}$ within the specifications of the DC and AC characteristics.

(b) External clock input ($T_A = -40$ to $+85^\circ\text{C}$)



DC Characteristics (T_A = -40 to +85°C, V_{DD} = 3.3 V ±0.3 V, V_{SS} = 0 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Input voltage, high	V _{IH}	Except for Note 1		2.0		V _{DD} + 0.3	V
		Note 1		0.75V _{DD}		V _{DD} + 0.3	V
Input voltage, low	V _{IL}	Except for Note 1		-0.5		0.8	V
		Note 1		-0.5		0.2V _{DD}	V
Clock input voltage, high	V _{XH}	X1 pin	Direct mode	0.8V _{DD}		V _{DD} + 0.3	V
			PLL mode	0.8V _{DD}		V _{DD} + 0.3	V
Clock input voltage, low	V _{XL}	X1 pin	Direct mode	-0.5		0.15V _{DD}	V
			PLL mode	-0.5		0.15V _{DD}	V
Schmitt-triggered input threshold voltage	V _T ⁺	Note 1 , rising edge			2.0		V
	V _T ⁻	Note 1 , falling edge			1.0		V
Schmitt-triggered input hysteresis width	V _T ⁺ - V _T ⁻	Note 1		0.3			V
Output voltage, high	V _{OH}	I _{OH} = -2.5 mA		0.8V _{DD}			V
		I _{OH} = -100 μA		V _{DD} - 0.4			V
Output voltage, low	V _{OL}	I _{OL} = 2.5 mA				0.45	V
Input leakage current, high	I _{LIH}	V _I = V _{DD} , except for Note 2				10	μA
Input leakage current, low	I _{LIL}	V _I = 0 V, except for Note 2				-10	μA
Output leakage current, high	I _{LOH}	V _O = V _{DD}				10	μA
Output leakage current, low	I _{LOL}	V _O = 0 V				-10	μA
Analog pin input leakage current	I _{LWASN}	Note 2				±10	μA
Power supply current	During normal operation	I _{DD1}	Direct mode		2.4 × f _{xx} + 30	3.6 × f _{xx} + 45	mA
			PLL mode		2.4 × f _{xx} + 30	3.6 × f _{xx} + 45	mA
	In HALT mode	I _{DD2}	Direct mode		1.2 × f _{xx} + 20	1.8 × f _{xx} + 30	mA
			PLL mode		1.2 × f _{xx} + 20	1.8 × f _{xx} + 30	mA
	In IDLE mode	I _{DD3}	Direct mode		10	30	mA
			PLL mode		10	30	mA
In STOP mode	I _{DD4}			10	200	μA	

Notes 1. P01/TI000/INTP000, P02/INTP001, P04/DMARQ0/INTP100, P05/DMARQ1/INTP101, P11/TI010/INTP010, P12/INTP011, P24/TC0/INTP110, P41/RXD0/SI0, P42/SCK0, P44/RXD1/SI1, P45/SCK1, MODE0 to MODE2, RESET

2. P70/ANI0 to P73/ANI3

Remarks 1. TYP. values are reference values for when T_A = 25°C, V_{DD} = 3.3 V. The current does not include the current flowing through pull-up resistors.

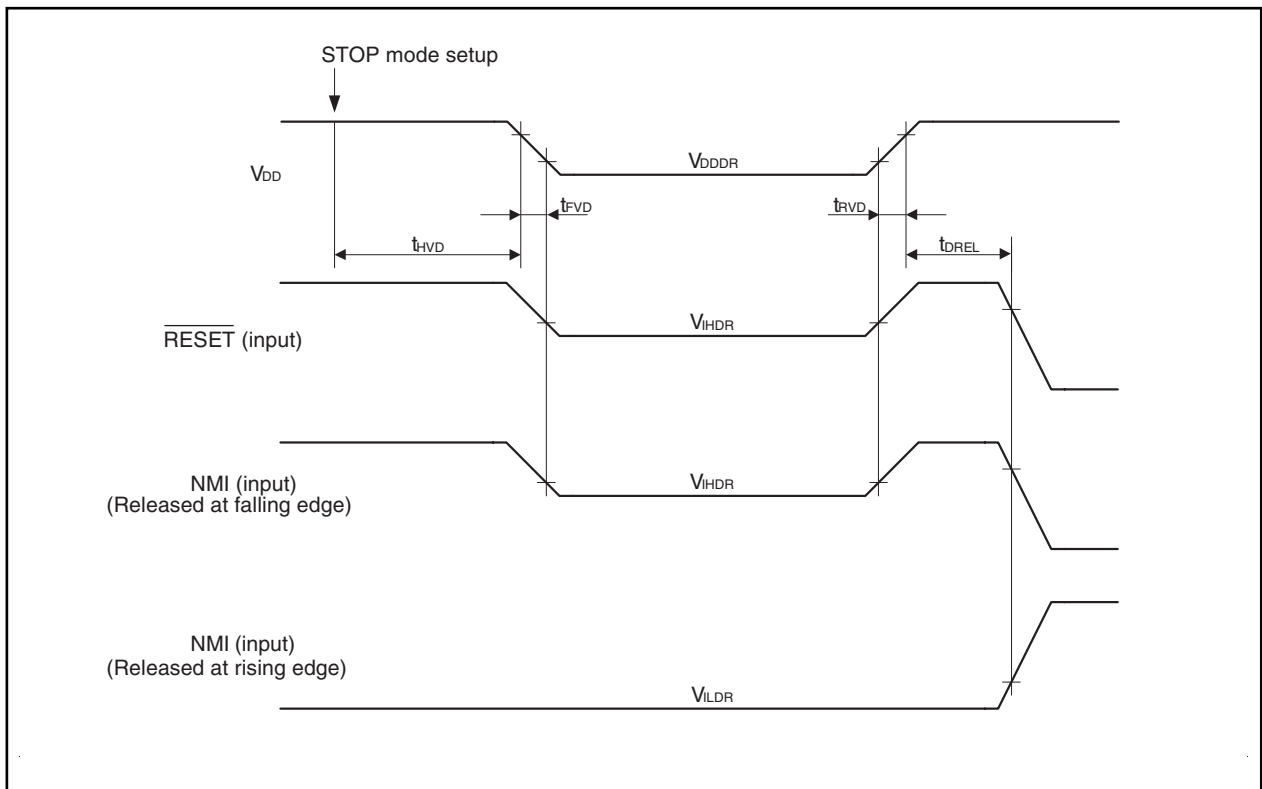
2. f_{xx}: CPU operation frequency

Data Retention Characteristics (T_A = -40 to +85°C)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention voltage	V _{DDDR}	STOP mode and V _{DD} = V _{DDDR}	1.5		3.6	V
Data retention current	I _{DDDR}	V _{DD} = V _{DDDR}		10	200	μA
Power supply voltage rise time	t _{RVD}		200			μs
Power supply voltage fall time	t _{FVD}		200			μs
Power supply voltage hold time (from STOP mode setting)	t _{HVD}		0			ms
STOP release signal input time	t _{DREL}		0			ns
Data retention high-level input voltage	V _{IHDR}	Note	0.8V _{DDDR}		V _{DDDR}	V
Data retention low-level input voltage	V _{ILDR}	Note	-0.5		0.2V _{DDDR}	V

Note P01/TI000/INTP000, P02/INTP001, P04/DMARQ0/INTP100, P05/DMARQ1/INTP101, P11/TI010/INTP010, P12/INTP011, P24/TC0/INTP110, P41/RXD0/SI0, P42/SCK0, P44/RXD1/SI1, P45/SCK1, MODE0 to MODE2, RESET

Remark TYP. values are reference values for when T_A = 25°C.

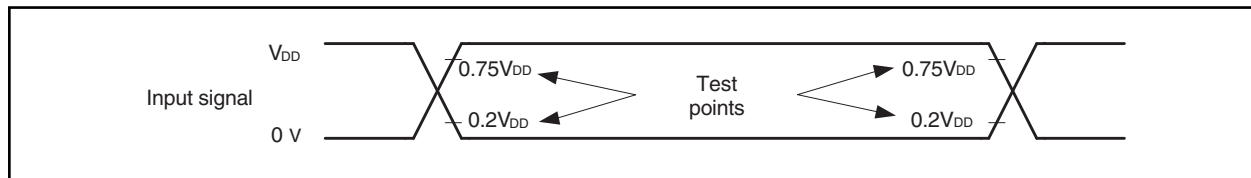


AC Characteristics

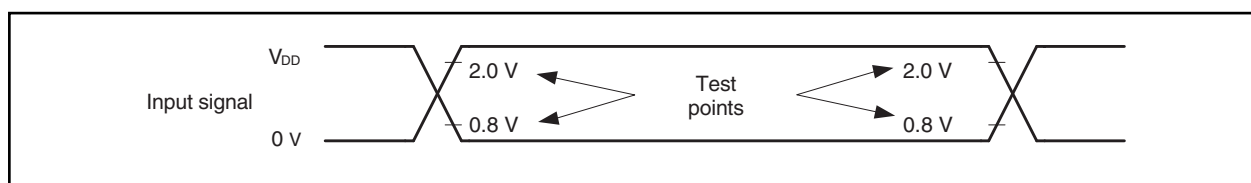
($T_A = -40$ to $+85^\circ\text{C}$, $V_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$, $V_{SS} = 0\text{ V}$, output pin load capacitance: $C_L = 50\text{ pF}$)

AC test input points

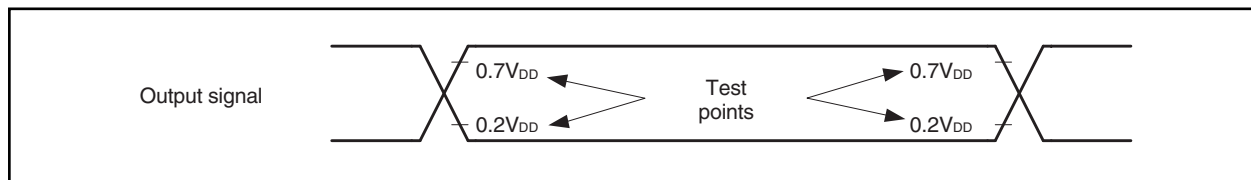
- (a) P01/TI000/INTP000, P02/INTP001, P04/ $\overline{\text{DMARQ0}}$ /INTP100, P05/ $\overline{\text{DMARQ1}}$ /INTP101, P11/TI010/INTP010, P12/INTP011, P24/ $\overline{\text{TC0}}$ /INTP110, P41/RXD0/SI0, P42/ $\overline{\text{SCK0}}$, P44/RXD1/SI1, P45/ $\overline{\text{SCK1}}$, MODE0 to MODE2, $\overline{\text{RESET}}$



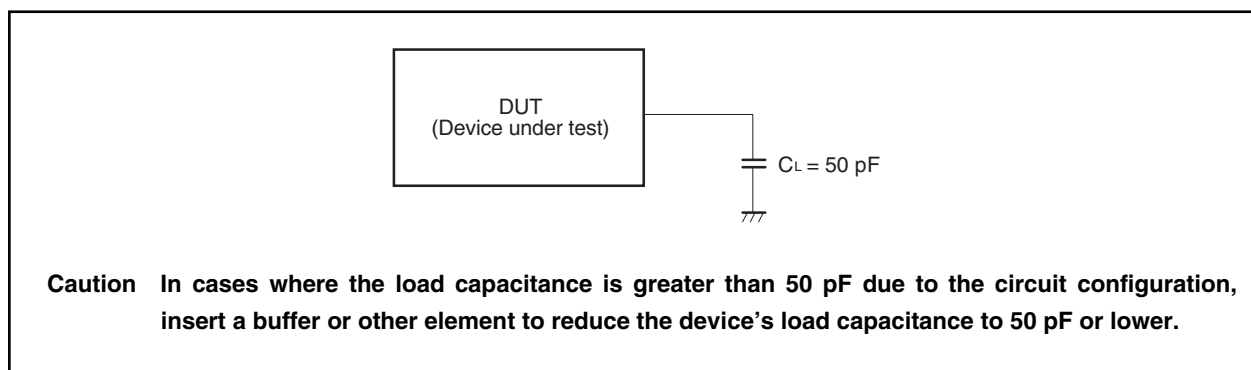
- (b) Other than (a) above



AC test output test points



Load condition



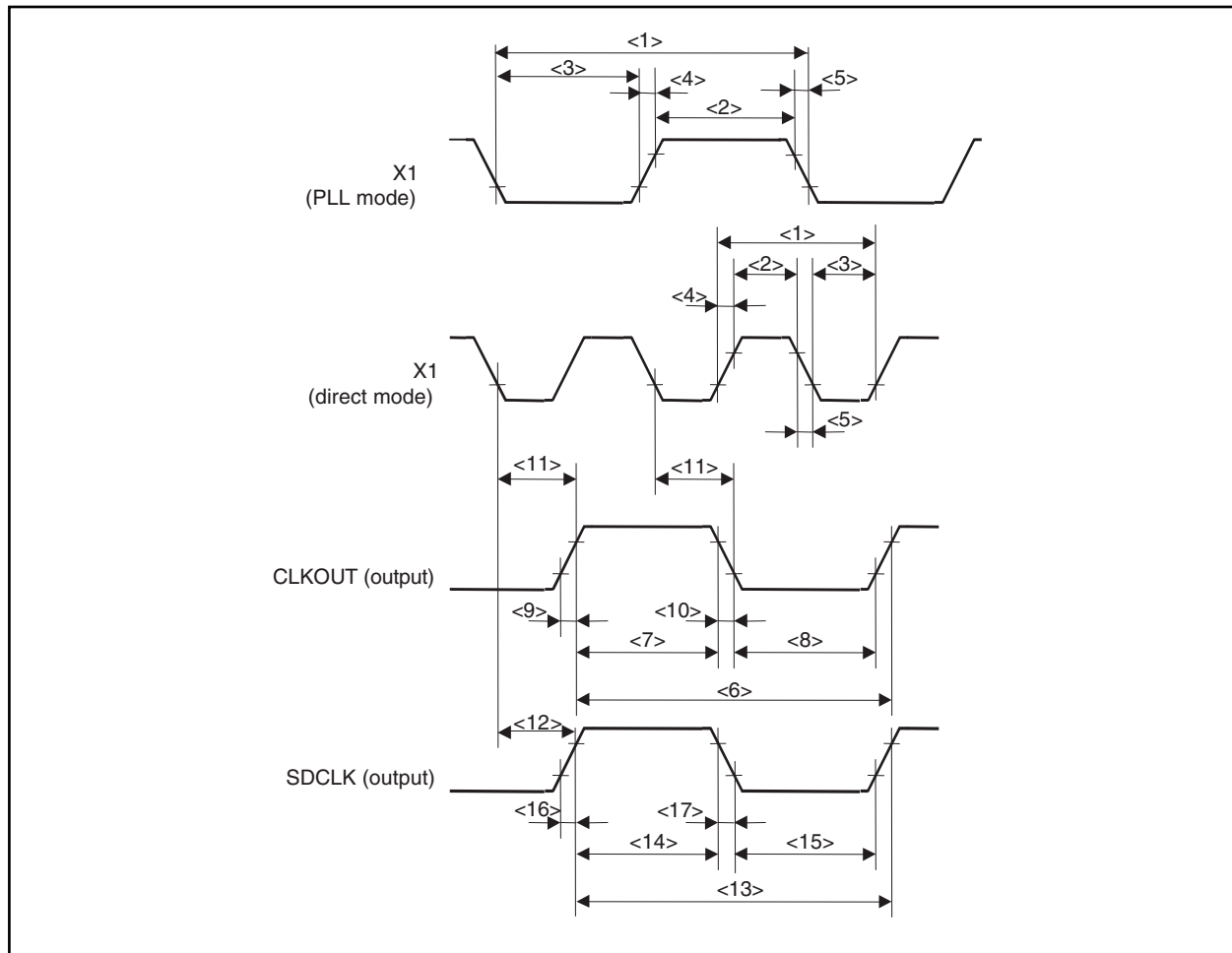
(1) Clock timing (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit		
X1 input cycle	<1>	t _{CYX}	Direct mode	25	125	ns	
			PLL mode	×10	250	250	ns
				Other than ×10	150	250	ns
X1 input high-level width	<2>	t _{WXH}	Direct mode	5		ns	
			PLL mode	50		ns	
X1 input low-level width	<3>	t _{WXL}	Direct mode	5		ns	
			PLL mode	50		ns	
X1 input rise time	<4>	t _{XR}	Direct mode		4	ns	
			PLL mode		10	ns	
X1 input fall time	<5>	t _{XF}	Direct mode		4	ns	
			PLL mode		10	ns	
CLKOUT output cycle	<6>	t _{CYK1}	25	250	ns		
CLKOUT high-level width	<7>	t _{WKH1}	0.5T – 5		ns		
CLKOUT low-level width	<8>	t _{WKL1}	0.5T – 6		ns		
CLKOUT rise time	<9>	t _{KR1}		5	ns		
CLKOUT fall time	<10>	t _{KF1}		4	ns		
Delay time from X1↓ to CLKOUT	<11>	t _{DKX}		40	ns		
Delay time from X1↓ to SDCLK	<12>	t _{DSX}		40	ns		
SDCLK output cycle	<13>	t _{CYK2}	25	250	ns		
SDCLK high-level width	<14>	t _{WKH2}	0.5T – 5		ns		
SDCLK low-level width	<15>	t _{WKL2}	0.5T – 6		ns		
SDCLK rise time	<16>	t _{KR2}		5	ns		
SDCLK fall time	<17>	t _{KF2}		4	ns		

Remarks 1. T = t_{CYK1}

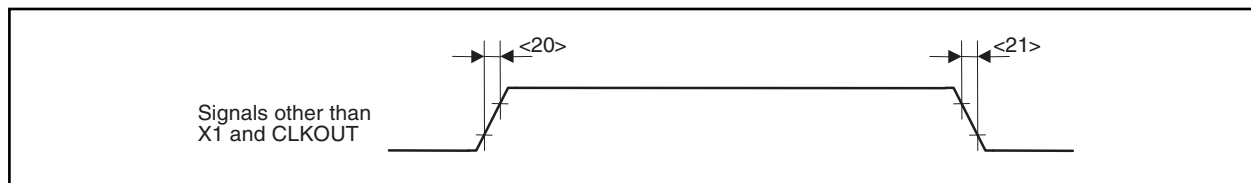
2. The phase difference between CLKOUT and SDCLK cannot be defined.

(1) Clock timing (2/2)



(2) Output waveform (other than X1 and CLKOUT)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Output rise time	<20> t_{OR}			5	ns
Output fall time	<21> t_{OF}			4	ns

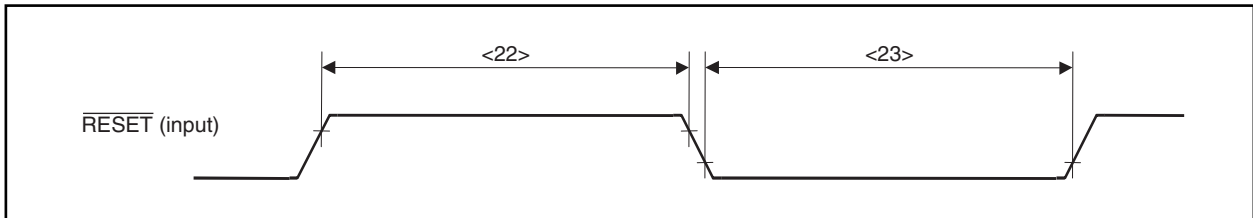


(3) Reset timing

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
$\overline{\text{RESET}}$ pin high-level width	<22>	t_{WRSH}		500		ns
$\overline{\text{RESET}}$ pin low-level width	<23>	t_{WRSL}	At power-on and at STOP mode release	500 + T_{os}		ns
			Except at power-on and at STOP mode release	500		ns

Remark T_{os} : Oscillation stabilization time

Caution Thoroughly evaluate the oscillation stabilization time.



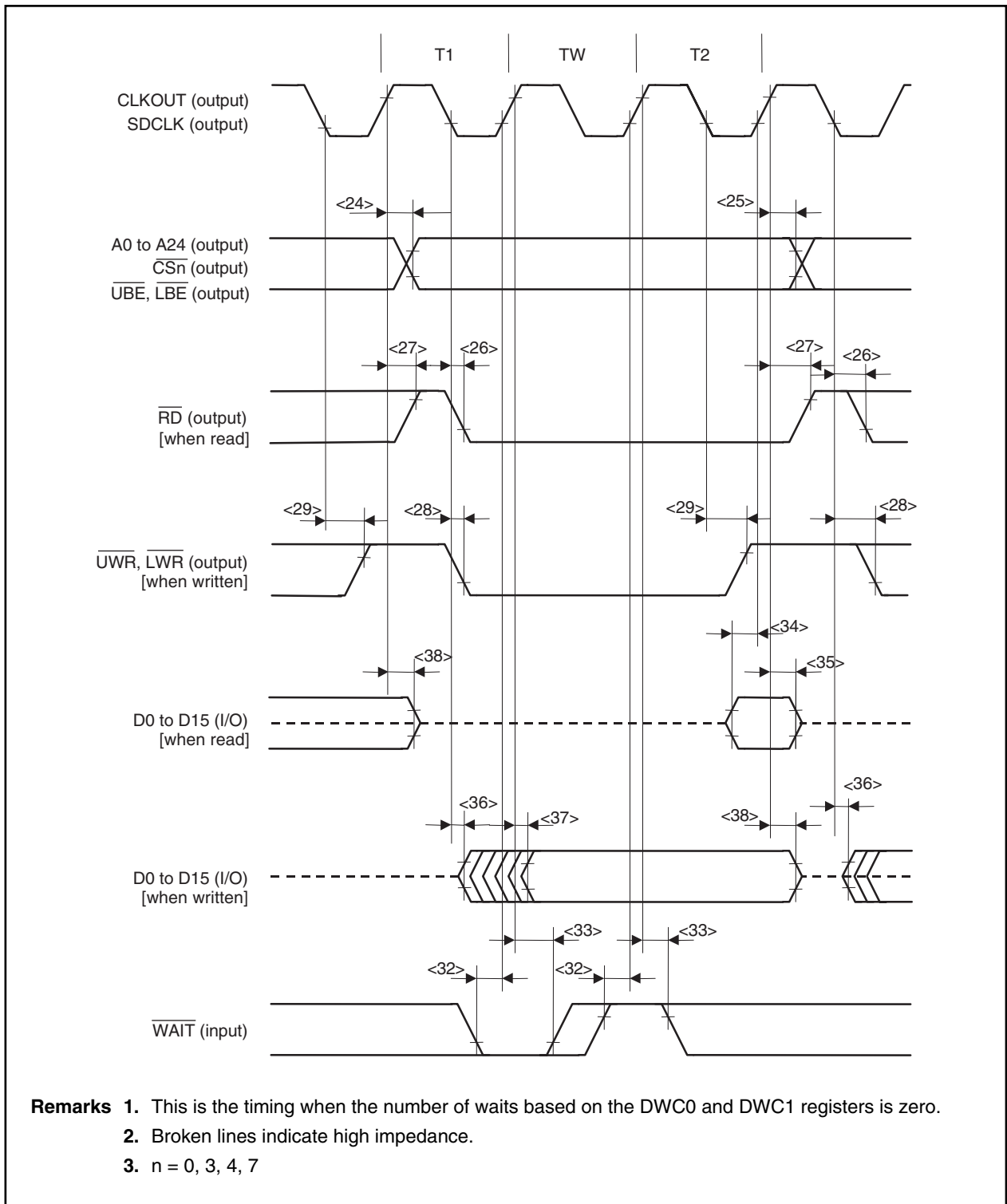
(4) SRAM, external ROM, and external I/O access timing**(a) Access timing (SRAM, external ROM, external I/O) (1/2)**

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Address, \overline{CS}_n output delay time (from CLKOUT \uparrow)	<24>	t_{DKA1}	2	15	ns
Address, \overline{CS}_n output delay time (from SDCLK \uparrow)			0	15	ns
Address, \overline{CS}_n output hold time (from CLKOUT \uparrow)	<25>	t_{HKA}	2	13	ns
Address, \overline{CS}_n output hold time (from SDCLK \uparrow)			0	13	ns
\overline{RD} , \overline{IORD} \downarrow delay time (from CLKOUT \downarrow)	<26>	t_{DKRDL}	2	13	ns
\overline{RD} , \overline{IORD} \downarrow delay time (from SDCLK \downarrow)			0	13	ns
\overline{RD} , \overline{IORD} \uparrow delay time (from CLKOUT \uparrow)	<27>	t_{HKRDH}	2	13	ns
\overline{RD} , \overline{IORD} \uparrow delay time (from SDCLK \uparrow)			0	13	ns
\overline{UWR} , \overline{LWR} , \overline{IOWR} \downarrow delay time (from CLKOUT \downarrow)	<28>	t_{DKWRL}	2	13	ns
\overline{UWR} , \overline{LWR} , \overline{IOWR} \downarrow delay time (from SDCLK \downarrow)			0	13	ns
\overline{UWR} , \overline{LWR} , \overline{IOWR} \uparrow delay time (from CLKOUT \downarrow)	<29>	t_{HKWRH}	2	13	ns
\overline{UWR} , \overline{LWR} , \overline{IOWR} \uparrow delay time (from SDCLK \downarrow)			0	13	ns
\overline{WAIT} setup time (to CLKOUT \uparrow)	<32>	t_{SWK}	8		ns
\overline{WAIT} setup time (to SDCLK \uparrow)			10		ns
\overline{WAIT} hold time (from CLKOUT \uparrow)	<33>	t_{HKW}	2		ns
\overline{WAIT} hold time (from SDCLK \uparrow)			2		ns
Data input setup time (to CLKOUT \uparrow)	<34>	t_{SKID}	8		ns
Data input setup time (to SDCLK \uparrow)			10		ns
Data input hold time (from CLKOUT \uparrow)	<35>	t_{HKID}	2		ns
Data input hold time (from SDCLK \uparrow)			2		ns
Data output delay time (from CLKOUT \downarrow)	<36>	t_{DKOD1}	2	13	ns
Data output delay time (from SDCLK \downarrow)			0	13	ns
Data output delay time (from CLKOUT \uparrow)	<37>	t_{DKOD2}	2	15	ns
Data output delay time (from SDCLK \uparrow)			0	15	ns
Data float delay time (from CLKOUT \uparrow)	<38>	t_{HKOD}	2	13	ns
Data float delay time (from SDCLK \uparrow)			0	13	ns

Remarks 1. Maintain at least one of the data input hold times, t_{HRDID} or t_{HKID} .

2. $n = 0, 3, 4, 7$

(a) Access timing (SRAM, external ROM, external I/O) (2/2)



- Remarks**
1. This is the timing when the number of waits based on the DWC0 and DWC1 registers is zero.
 2. Broken lines indicate high impedance.
 3. n = 0, 3, 4, 7

(b) Read timing (SRAM, external ROM, external I/O) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Data input setup time (to address)	<39>	t_{SAID}		$(2 + w + w_D + w_{AS})T - 19$	ns
Data input setup time (to \overline{RD})	<40>	t_{SRDID}		$(1.5 + w + w_D)T - 19$	ns
\overline{RD} low-level width	<41>	t_{WRDL}	$(1.5 + w + w_D)T - 10$		ns
\overline{RD} high-level width	<42>	t_{WRDH}	$(0.5 + w_{AS} + i)T - 10$		ns
Delay time from address, \overline{CS}_n to $\overline{RD}\downarrow$	<43>	t_{DARD}	$(0.5 + w_{AS})T - 10$		ns
Delay time from $\overline{RD}\uparrow$ to address	<44>	t_{DRDA}	iT		ns
Data input hold time (from $\overline{RD}\uparrow$)	<45>	t_{HRDID}	0		ns
Delay time from $\overline{RD}\uparrow$ to data output	<46>	t_{DRDOD}	$(0.5 + i)T - 10$		ns
\overline{WAIT} setup time (to address)	<47>	t_{SAW}	Note	$(1 + w_{AS})T - 21$	ns
\overline{WAIT} high-level width	<50>	t_{WWH}	$T - 10$		ns
Data output hold time (from \overline{UWR} , $\overline{LWR}\uparrow$)	<57>	t_{HWROD}	$(0.5 + i)T - 8$		ns

Note For the first \overline{WAIT} sampling when the wait count based on the DWC0 and DWC1 registers is zero.

Remarks 1. $T = t_{CYK1}$

2. w : Wait count based on \overline{WAIT}

3. w_D : Wait count based on the DWC0 and DWC1 registers

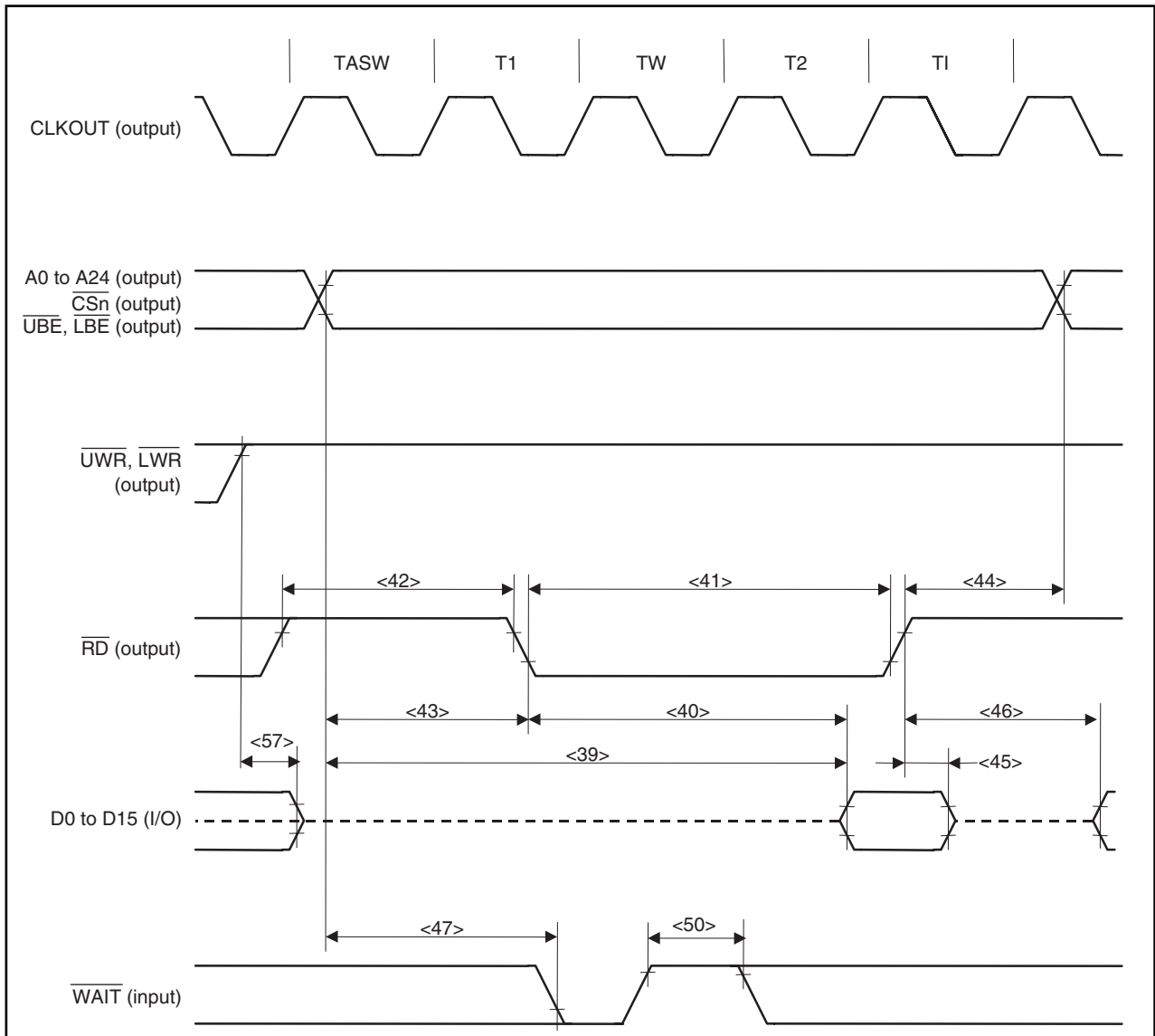
4. Maintain at least one of the data input hold times t_{HRDID} or t_{HKID}

5. $n = 0, 3, 4, 7$

6. i : Idle state count

7. w_{AS} : Address setup wait count based on the ASC register

(b) Read timing (SRAM, external ROM, external I/O) (2/2)



- Remarks**
1. This is the timing when the wait count based on the DWC0 and DWC1 registers is zero, the idle state count based on the BCC register is 1, and the wait count based on the ASC register is 1.
 2. Broken lines indicate high impedance.
 3. n = 0, 3, 4, 7

(c) Write timing (SRAM, external ROM, external I/O) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to address)	<47> t_{SAW}	Note		$(1 + w_{\text{AS}})T - 21$	ns
$\overline{\text{WAIT}}$ high-level width	<50> t_{WWH}		$T - 10$		ns
Delay time from address, $\overline{\text{CS}}_n$ to $\overline{\text{UWR}}$, $\overline{\text{LWR}}\downarrow$	<51> t_{DAWR}		$(0.5 + w_{\text{AS}})T - 10$		ns
Address setup time (to $\overline{\text{UWR}}$, $\overline{\text{LWR}}\uparrow$)	<52> t_{SAWR}		$(1.5 + w + w_{\text{D}} + w_{\text{AS}})T - 10$		ns
Delay time from $\overline{\text{UWR}}$, $\overline{\text{LWR}}\uparrow$ to address	<53> t_{DWRA}		$(0.5 + i)T - 10$		ns
$\overline{\text{UWR}}$, $\overline{\text{LWR}}$ high-level width	<54> t_{WWRH}		$(0.5 + i + w_{\text{AS}})T - 10$		ns
$\overline{\text{UWR}}$, $\overline{\text{LWR}}$ low-level width	<55> $t_{\text{WWR L}}$		$(1 + w + w_{\text{D}})T - 10$		ns
Data output setup time (to $\overline{\text{UWR}}$, $\overline{\text{LWR}}\uparrow$)	<56> t_{SODWR}		$(0.5 + w + w_{\text{D}})T - 10$		ns
Data output hold time (from $\overline{\text{UWR}}$, $\overline{\text{LWR}}\uparrow$)	<57> t_{HWROD}		$(0.5 + i)T - 8$		ns

Note For the first $\overline{\text{WAIT}}$ sampling when the wait count based on the DWC0 and DWC1 registers is zero.

Remarks 1. $T = t_{\text{CYK1}}$

2. w : Wait count based on $\overline{\text{WAIT}}$

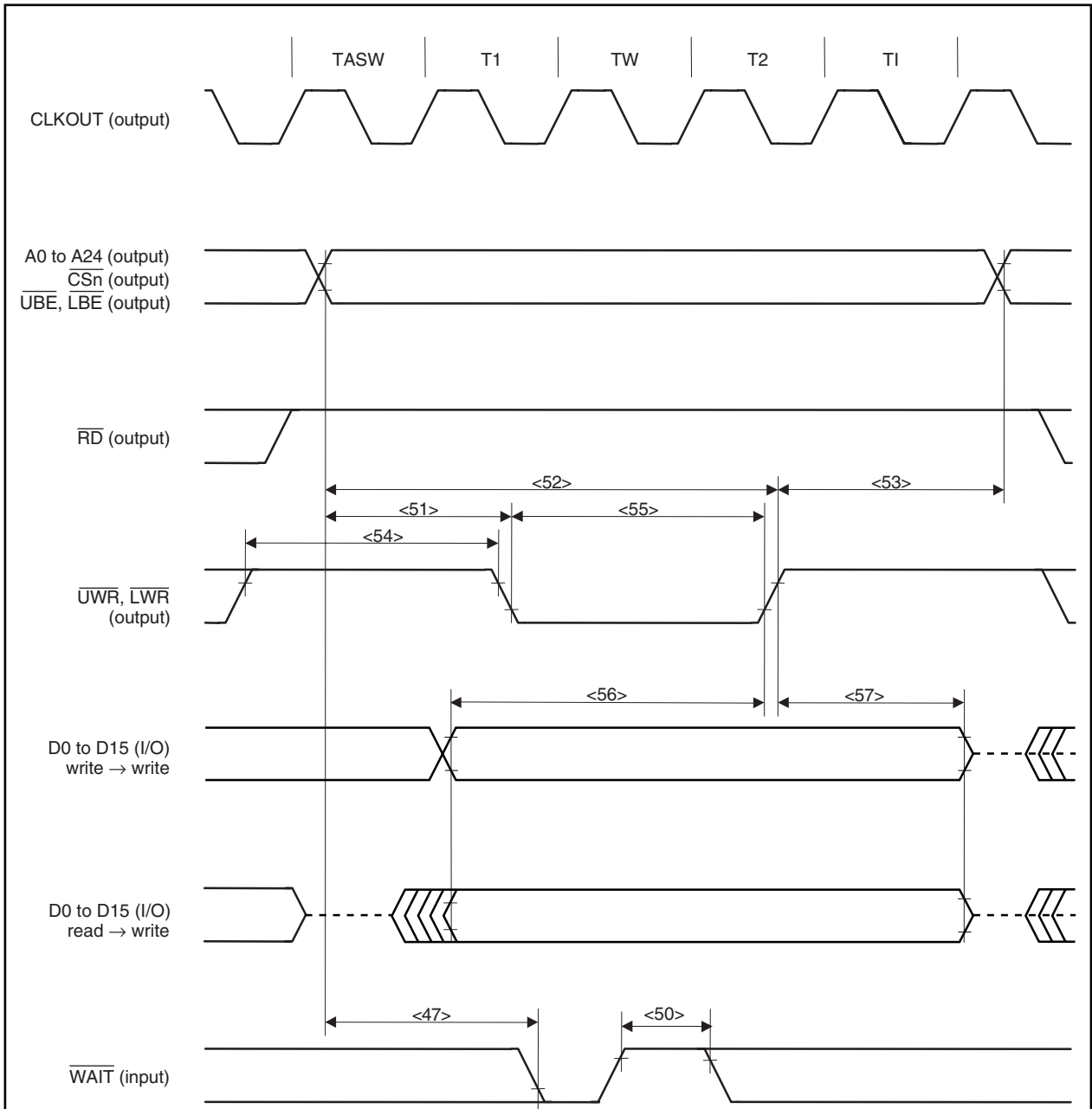
3. w_{D} : Wait count based on the DWC0 and DWC1 registers

4. $n = 0, 3, 4, 7$

5. i : Idle state count

6. w_{AS} : Address setup wait count based on the ASC register

(c) Write timing (SRAM, external ROM, external I/O) (2/2)



- Remarks**
1. This is the timing when the wait count based on the DWC0 and DWC1 registers is zero, the idle state count based on the BCC register is 1, and the wait count based on the ASC register is 1.
 2. Broken lines indicate high impedance.
 3. n = 0, 3, 4, 7

(5) Page ROM access timing

(a) 8-bit bus width (halfword/word access), 16-bit bus width (word access) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to CLKOUT \uparrow)	<32> t_{SWK}		8		ns
$\overline{\text{WAIT}}$ hold time (from CLKOUT \uparrow)	<33> t_{HKW}		0		ns
Data input setup time (to CLKOUT \uparrow)	<34> t_{SKID}		8		ns
Data input hold time (from CLKOUT \uparrow)	<35> t_{HKID}		0		ns
Off-page data input setup time (to address)	<39> t_{SAID}			$(2 + w + w_D + w_{AS})T - 21$	ns
Off-page data input setup time (to $\overline{\text{RD}}$)	<40> t_{SRDID}			$(1.5 + w + w_D)T - 21$	ns
Data input hold time (from $\overline{\text{RD}}\uparrow$)	<45> t_{HRDID}		0		ns
Delay time from $\overline{\text{RD}}\uparrow$ to data output	<46> t_{DRDOD}		$(0.5 + i)T - 10$		ns
On-page data input setup time (to address)	<64> t_{SOAID}			$(2 + w + w_{PR} + w_{AS})T - 21$	ns

Remarks 1. $T = t_{\text{CYK1}}$

2. w : Wait count based on $\overline{\text{WAIT}}$

3. w_D : Wait count based on the DWC0 and DWC1 registers

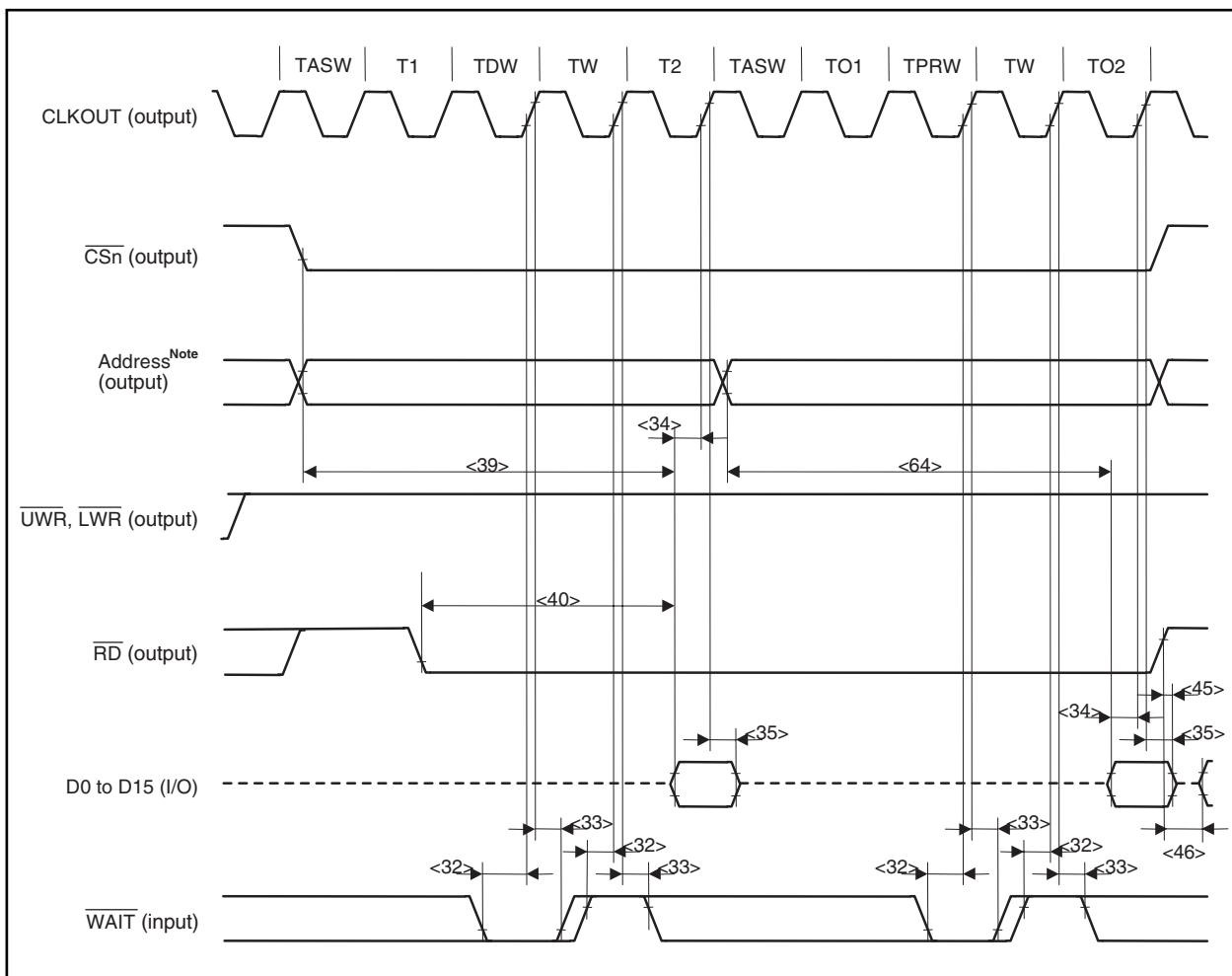
4. w_{PR} : Wait count based on the PRC register

5. i : Count of idle states inserted when a write cycle follows a read cycle

6. w_{AS} : Address setup wait count based on the ASC register

7. Maintain at least one of the data input hold times t_{HKID} or t_{HRDID}

(a) 8-bit bus width (halfword/word access), 16-bit bus width (word access) (2/2)



Note On-page and off-page addresses are as follows.

PRC Register				On-Page Address	Off-Page Address
MA6	MA5	MA4	MA3		
0	0	0	0	A0 to A2	A3 to A24
0	0	0	1	A0 to A3	A4 to A24
0	0	1	1	A0 to A4	A5 to A24
0	1	1	1	A0 to A5	A6 to A24
1	1	1	1	A0 to A6	A7 to A24

Remarks 1. This is the timing for the following case.

Wait count based on the DWC0 and DWC1 registers (TDW): 1

Wait count based on the PRC register (TPRW): 1

Wait count based on the ASC register (TASW): 1

2. Broken lines indicate high impedance.

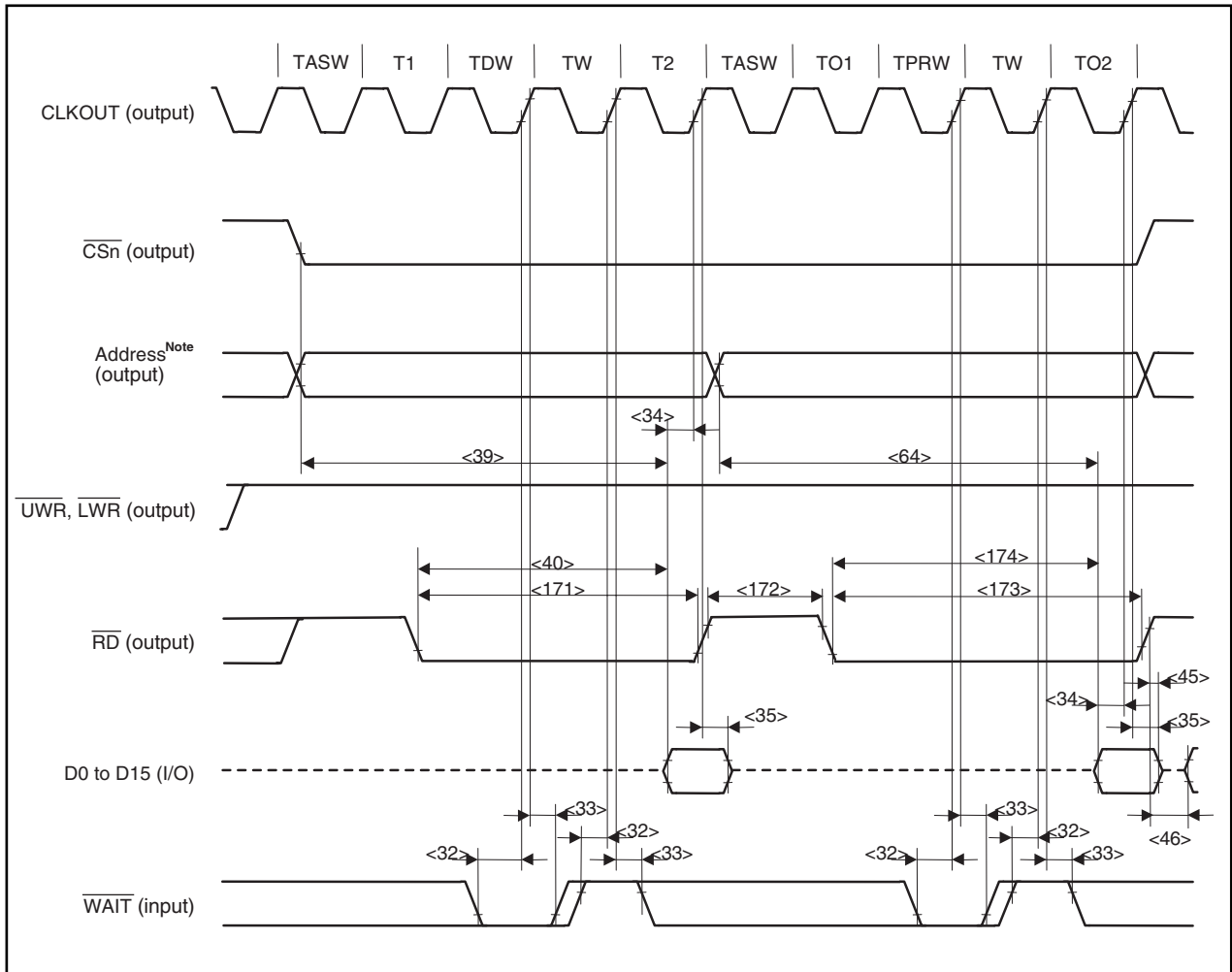
3. n = 0, 3, 4, 7

(b) 8-bit bus width (byte access), 16-bit bus width (byte/halfword access) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to CLKOUT \uparrow)	<32> t_{SWK}		8		ns
$\overline{\text{WAIT}}$ hold time (from CLKOUT \uparrow)	<33> t_{HKW}		0		ns
Data input setup time (to CLKOUT \uparrow)	<34> t_{SKID}		8		ns
Data input hold time (from CLKOUT \uparrow)	<35> t_{HKID}		0		ns
Off-page data input setup time (to address)	<39> t_{SAID}			$(2 + w + w_D + w_{AS})T - 21$	ns
Off-page data input setup time (to $\overline{\text{RD}}$)	<40> t_{SRDID}			$(1.5 + w + w_D)T - 21$	ns
Off-page $\overline{\text{RD}}$ low-level width	<171> t_{WRDL}		$(1.5 + w + w_D)T - 10$		ns
$\overline{\text{RD}}$ high-level width	<172> t_{WRDH}		$(0.5 + w_{AS})T - 10$		ns
Data input hold time (from $\overline{\text{RD}}\uparrow$)	<45> t_{HRDID}		0		ns
Delay time from $\overline{\text{RD}}\uparrow$ to data output	<46> t_{DRDOD}		$(0.5 + i)T - 10$		ns
On-page $\overline{\text{RD}}$ low-level width	<173> t_{WORDL}		$(1.5 + w + w_{PR})T - 10$		ns
On-page data input setup time (to address)	<64> t_{SOAID}			$(2 + w + w_{PR} + w_{AS})T - 21$	ns
On-page data input setup time (to $\overline{\text{RD}}$)	<174> t_{SORDID}			$(1.5 + w + w_{PR})T - 21$	ns

- Remarks**
1. $T = t_{\text{CYK1}}$
 2. w : Wait count based on $\overline{\text{WAIT}}$
 3. w_D : Wait count based on the DWC0 and DWC1 registers
 4. w_{PR} : Wait count based on the PRC register
 5. i : Count of idle states inserted when a write cycle follows a read cycle
 6. w_{AS} : Address setup wait count based on the ASC register
 7. Maintain at least one of the data input hold times t_{HKID} or t_{HRDID}

(b) 8-bit bus width (byte access), 16-bit bus width (byte/halfword access) (2/2)



Note On-page and off-page addresses are as follows.

PRC Register				On-Page Address	Off-Page Address
MA6	MA5	MA4	MA3		
0	0	0	0	A0 to A2	A3 to A24
0	0	0	1	A0 to A3	A4 to A24
0	0	1	1	A0 to A4	A5 to A24
0	1	1	1	A0 to A5	A6 to A24
1	1	1	1	A0 to A6	A7 to A24

Remarks 1. This is the timing for the following case.

Wait count based on the DWC0 and DWC1 registers (TDW): 1

Wait count based on the PRC register (TPRW): 1

Wait count based on the ASC register (TASW): 1

2. Broken lines indicate high impedance.

3. n = 0, 3, 4, 7

(6) SDRAM access timing**(a) Read timing (SDRAM access) (1/2)**

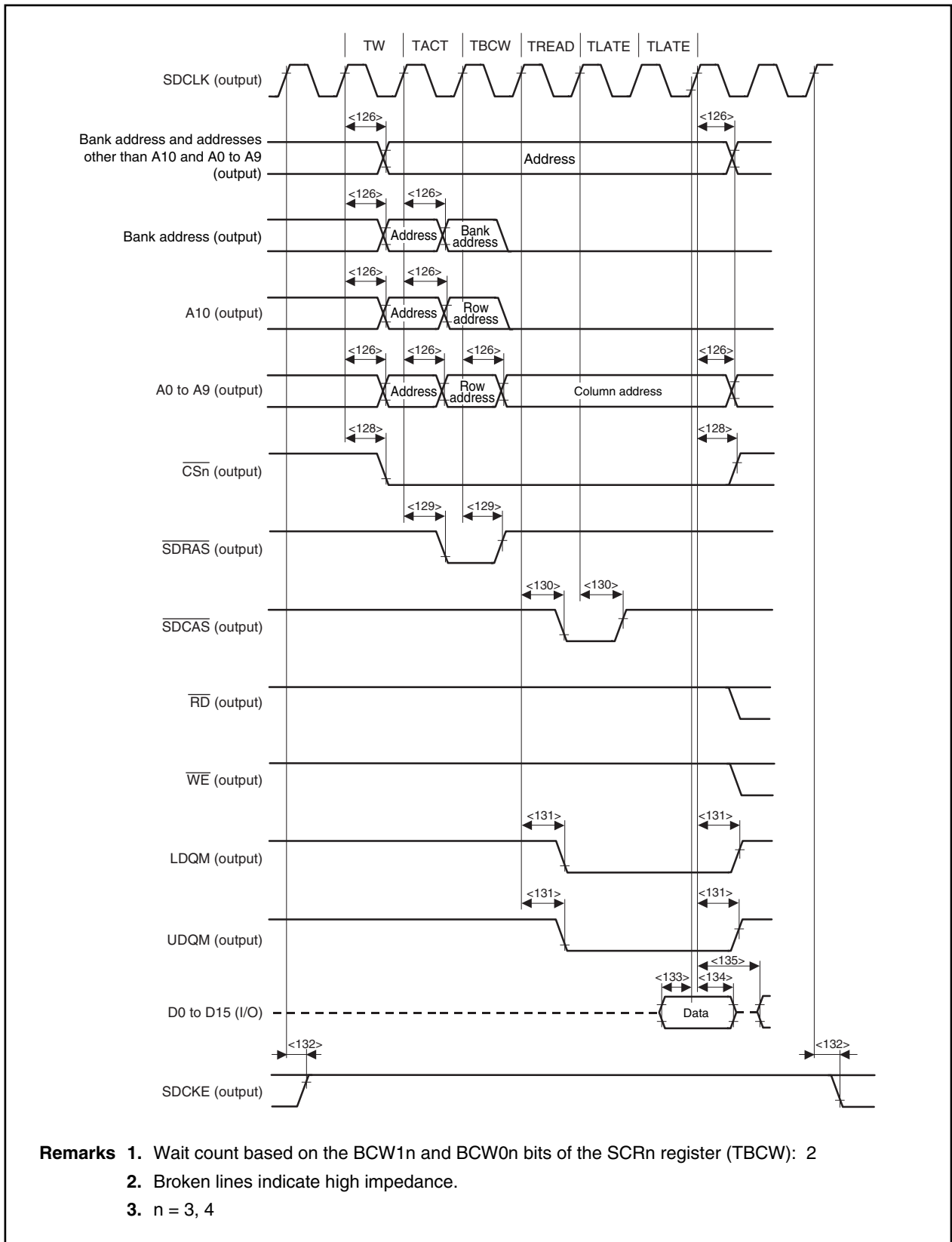
Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Address delay time (from SDCLK↑)	<126> t_{DKA2}		2	13	ns
\overline{CS}_n delay time (from SDCLK↑)	<128> t_{DKCS}		2	13	ns
\overline{SDRAS} delay time (from SDCLK↑)	<129> t_{DKRAS}		2	13	ns
\overline{SDCAS} delay time (from SDCLK↑)	<130> t_{DKCAS}		2	13	ns
UDQM, LDQM delay time (from SDCLK↑)	<131> t_{DKDQM}		2	13	ns
SDCKE delay time (from SDCLK↑)	<132> t_{DKCKE}		2	13	ns
Data input setup time (at SDRAM read, to SDCLK↑)	<133> t_{SDRMK}		8		ns
Data input hold time (at SDRAM read, from SDCLK↑)	<134> t_{HKDRM}		0		ns
Delay time from SDCLK↑ to data output	<135> t_{SDOD}			$(1 + i)T - 5$	ns

Remarks 1. $T = t_{CYK2}$

2. i : Idle state count

3. $n = 3, 4$

(a) Read timing (SDRAM access) (2/2)

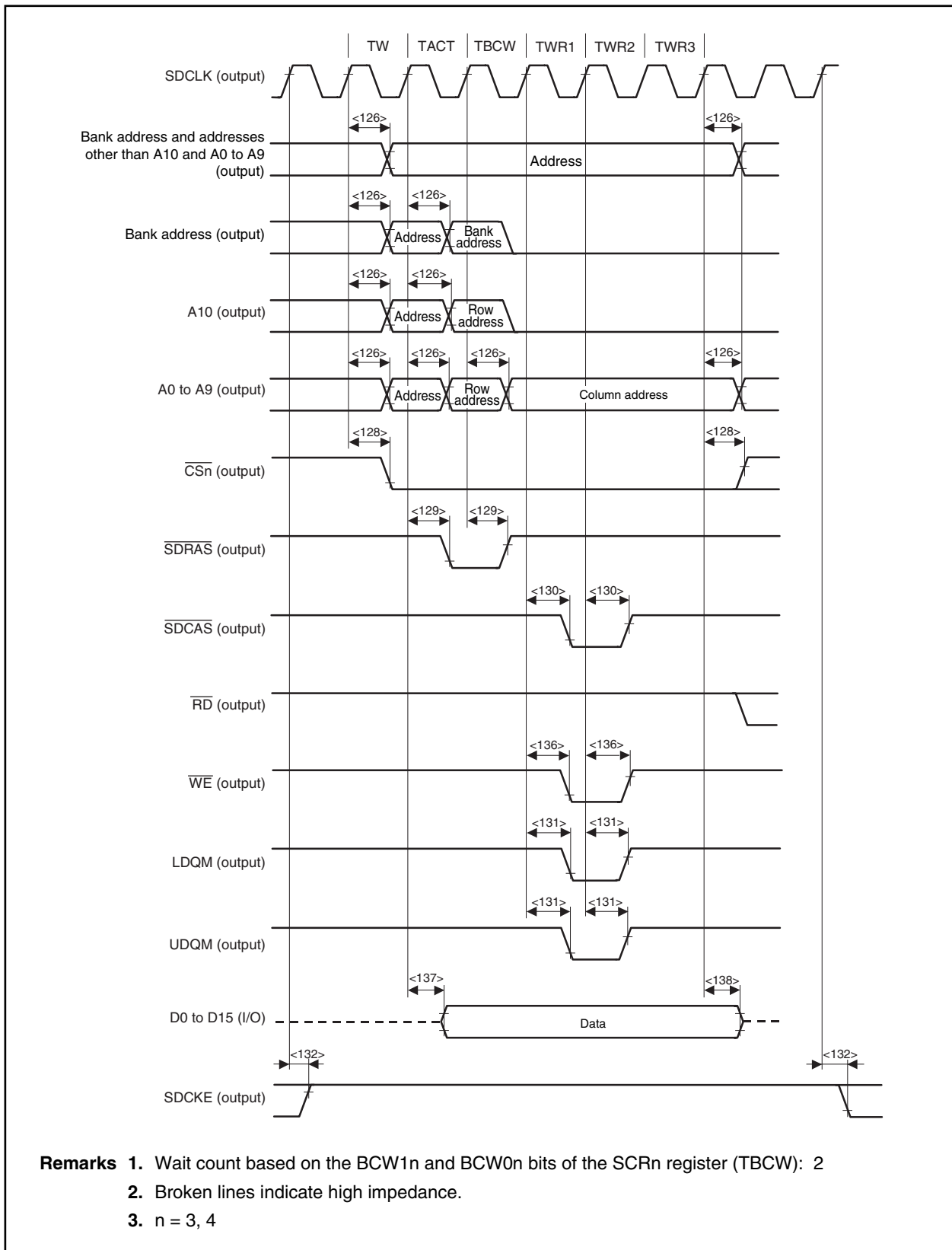


(b) Write timing (SDRAM access) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Address delay time (from SDCLK↑)	<126> t_{DKA2}		2	13	ns
\overline{CS}_n delay time (from SDCLK↑)	<128> t_{DKCS}		2	13	ns
\overline{SDRAS} delay time (from SDCLK↑)	<129> t_{DKRAS}		2	13	ns
\overline{SDCAS} delay time (from SDCLK↑)	<130> t_{DKCAS}		2	13	ns
UDQM, LDQM delay time (from SDCLK↑)	<131> t_{DKDQM}		2	13	ns
SDCKE delay time (from SDCLK↑)	<132> t_{DKCKE}		2	13	ns
\overline{WE} delay time (from SDCLK↑)	<136> t_{DKWE}		2	13	ns
Data output delay time (from SDCLK↑)	<137> t_{DKDT}		2	13	ns
Data float delay time (from SDCLK↑)	<138> t_{HZKDT}		2	13	ns

Remark n = 3, 4

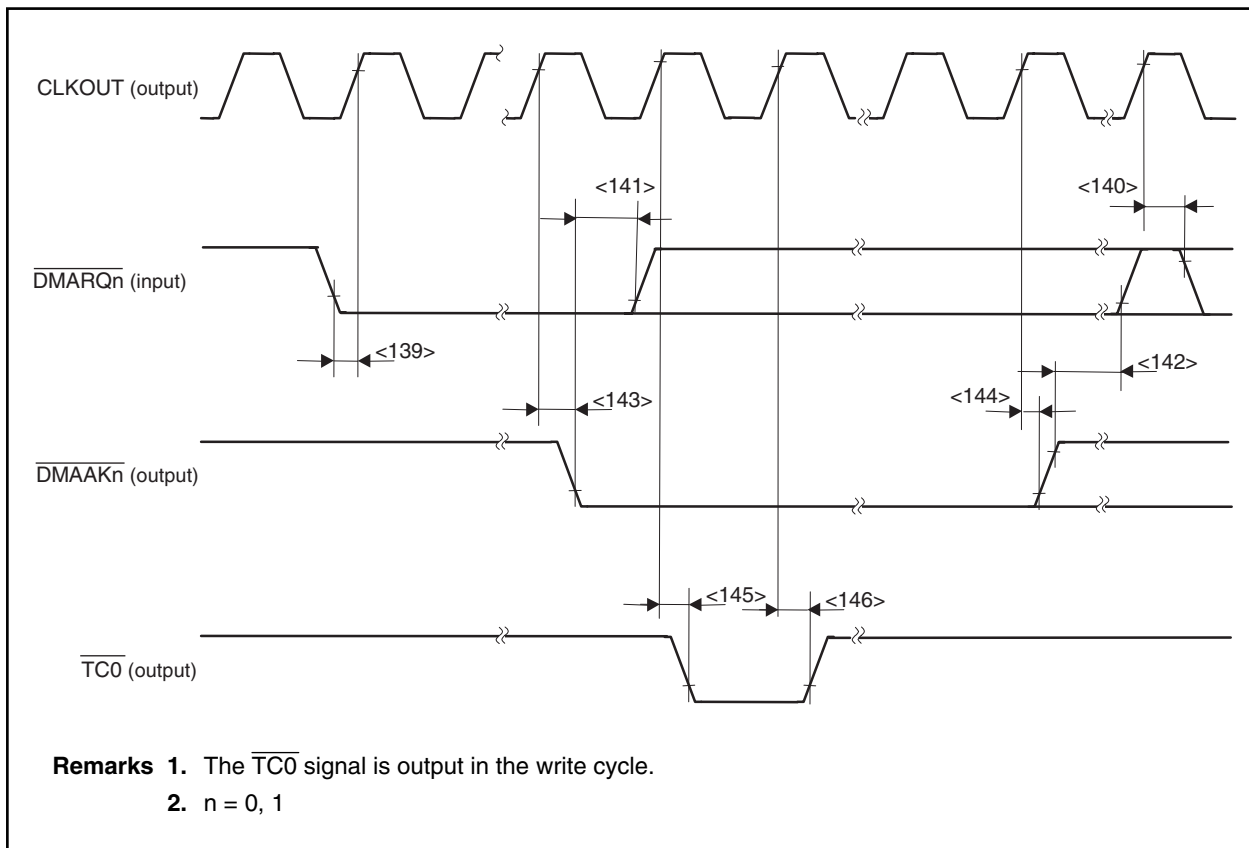
(b) Write timing (SDRAM access) (2/2)



(7) DMAC timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{DMARQn}}$ setup time (to CLKOUT↑)	<139> t_{SDRK}		8		ns
$\overline{\text{DMARQn}}$ hold time	<140> t_{HKDR1}	After inactive (from CLKOUT↑)	3		ns
	<141> t_{HKDR2}			Until $\overline{\text{DMAAKn}}\downarrow$	ns
Second DMA request disable timing in single transfer	<142> t_{AKDR}			3T	ns
$\overline{\text{DMAAKn}}$ output delay time (from CLKOUT↑)	<143> t_{DKDA}		2	13	ns
$\overline{\text{DMAAKn}}$ output hold time (from CLKOUT↑)	<144> t_{HKDA}		2	13	ns
$\overline{\text{TC0}}$ output delay time (from CLKOUT↑)	<145> t_{HKTC}		2	13	ns
$\overline{\text{TC0}}$ output hold time (from CLKOUT↑)	<146> t_{HKTC}		2	13	ns

- Remarks**
1. $T = t_{\text{CYK1}}$
 2. $n = 0, 1$

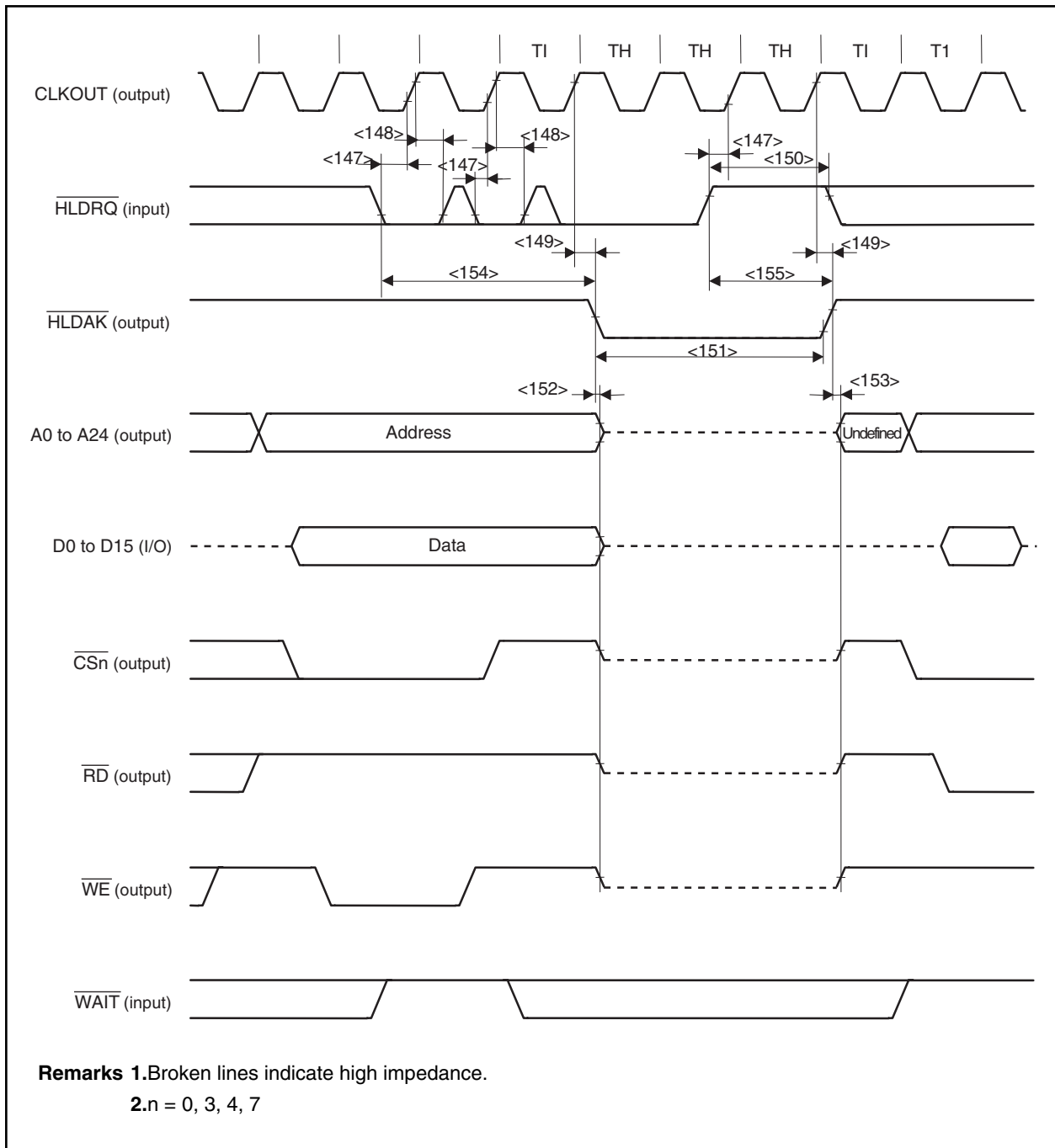


(8) Bus hold timing (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{HLDRQ}}$ setup time (to $\text{CLKOUT}\uparrow$)	<147> t_{SHRK}		8		ns
$\overline{\text{HLDRQ}}$ hold time (from $\text{CLKOUT}\uparrow$)	<148> t_{HKHR}		3		ns
Delay time from $\text{CLKOUT}\uparrow$ to $\overline{\text{HLDAK}}$	<149> t_{DKHA}		2	13	ns
$\overline{\text{HLDRQ}}$ high-level width	<150> t_{WHQH}		$T + 3$		ns
$\overline{\text{HLDAK}}$ low-level width	<151> t_{WHAL}		$T - 11$		ns
Delay time from $\overline{\text{HLDAK}}\downarrow$ to bus float	<152> t_{DKCF}		0		ns
Delay time from $\overline{\text{HLDAK}}\uparrow$ to bus output	<153> t_{DHAC}		0	13	ns
Delay time from $\overline{\text{HLDRQ}}\downarrow$ to $\overline{\text{HLDAK}}\downarrow$	<154> t_{DHQHA1}		$2T$		ns
Delay time from $\overline{\text{HLDRQ}}\uparrow$ to $\overline{\text{HLDAK}}\uparrow$	<155> t_{DHQHA2}		T	$2T + 10$	ns

Remark $T = t_{\text{CYK1}}$

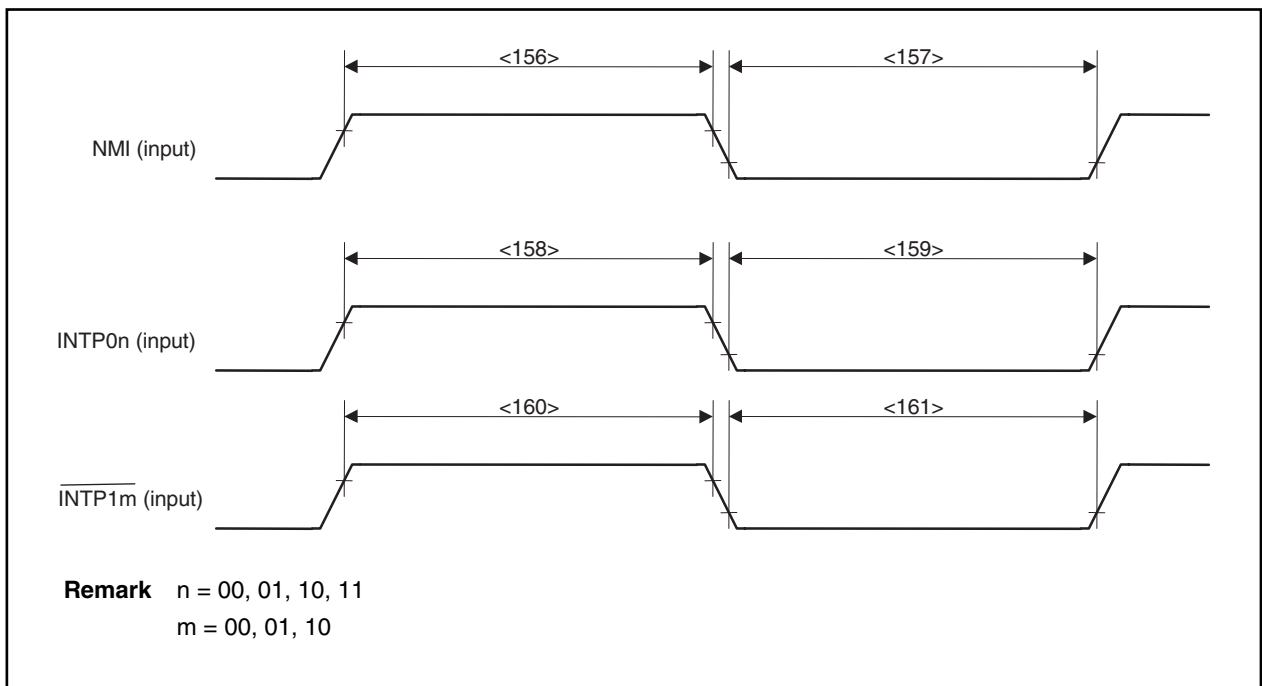
(8) Bus hold timing (2/2)



(9) Interrupt timing

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
NMI high-level width	<156>	t_{WNIH}		500		ns
NMI low-level width	<157>	t_{WNIL}		500		ns
INTP0n high-level width	<158>	t_{WIT0H}		$3T + 500$		ns
INTP0n low-level width	<159>	t_{WIT0L}		$3T + 500$		ns
$\overline{\text{INTP1m}}$ high-level width	<160>	t_{WIT1H}		500		ns
$\overline{\text{INTP1m}}$ low-level width	<161>	t_{WIT1L}		500		ns

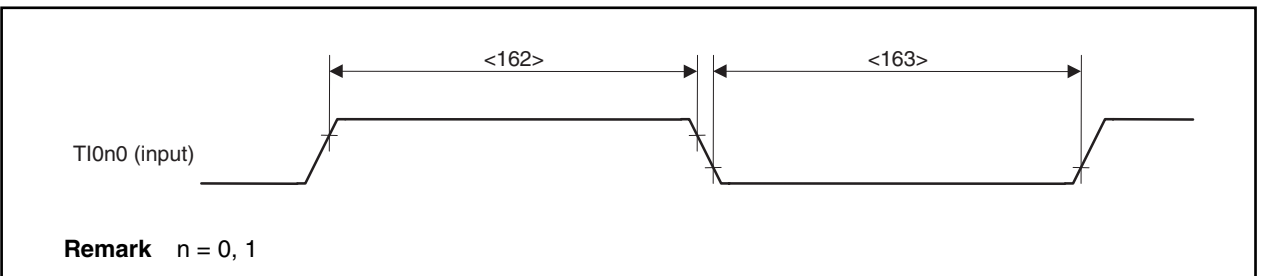
- Remarks** 1. $n = 00, 01, 10, 11$
 $m = 00, 01, 10$
 2. $T = t_{CYK1}$



(10) RPU timing

Parameter	Symbol		Conditions	MIN.	MAX.	Unit
TI0n0 high-level width	<162>	t_{WTIH}		$3T + 500$		ns
TI0n0 low-level width	<163>	t_{WTIL}		$3T + 500$		ns

- Remarks** 1. $n = 0, 1$
 2. $T = t_{CYK1}$



(11) CSI0, CSI1 timing (1/3)

(a) Master mode

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCKn}}$ cycle	<164> t_{CYSK1}	Output	320		ns
$\overline{\text{SCKn}}$ high-level width	<165> t_{WSK1H}	Output	$0.5t_{\text{CYSK1}} - 20$		ns
$\overline{\text{SCKn}}$ low-level width	<166> t_{WSK1L}	Output	$0.5t_{\text{CYSK1}} - 20$		ns
SIn setup time (to $\overline{\text{SCKn}}\uparrow$)	<167> t_{SSISK}		30		ns
SIn setup time (to $\overline{\text{SCKn}}\downarrow$)			30		ns
SIn hold time (from $\overline{\text{SCKn}}\uparrow$)	<168> t_{HSKSI}		30		ns
SIn hold time (from $\overline{\text{SCKn}}\downarrow$)			30		ns
SOn output delay time (from $\overline{\text{SCKn}}\downarrow$)	<169> t_{DSKSO}			30	ns
SOn output delay time (from $\overline{\text{SCKn}}\uparrow$)				30	ns
SOn output hold time (from $\overline{\text{SCKn}}\uparrow$)	<170> t_{HSKSO}		$0.5t_{\text{CYSK1}} - 5$		ns
SOn output hold time (from $\overline{\text{SCKn}}\downarrow$)			$0.5t_{\text{CYSK1}} - 5$		ns

Remark n = 0, 1

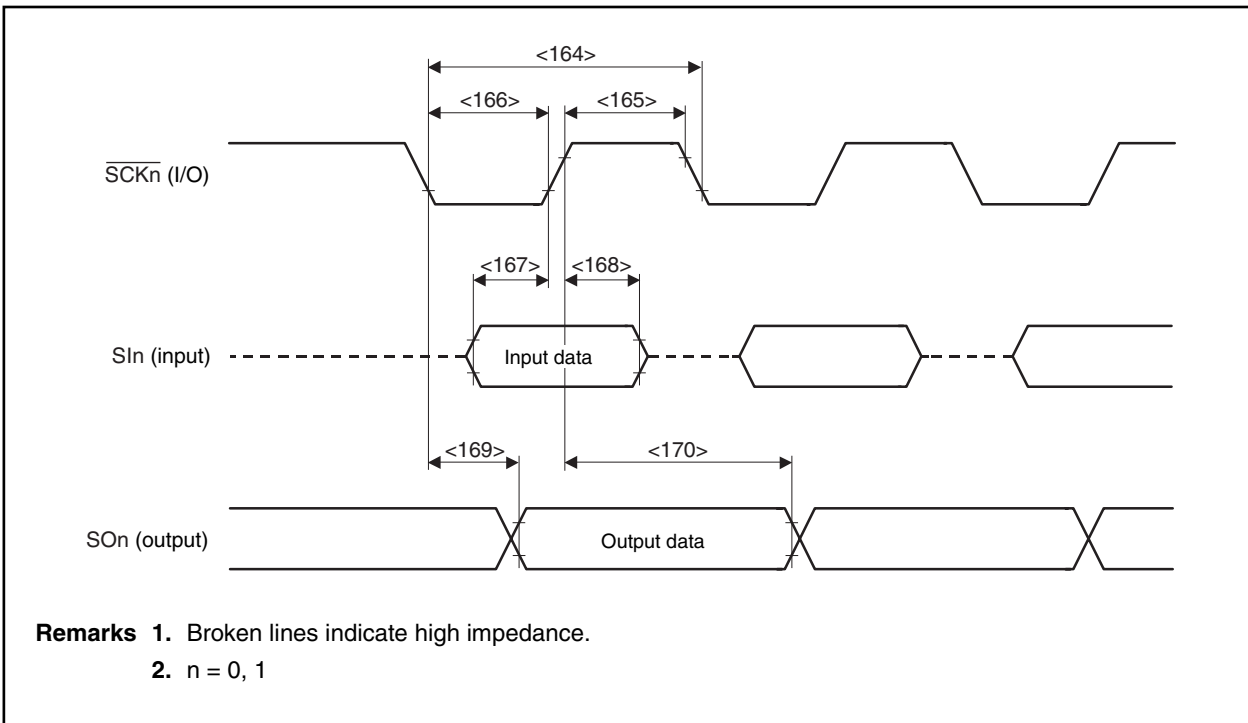
(b) Slave mode

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCKn}}$ cycle	<164> t_{CYSK1}	Input	200		ns
$\overline{\text{SCKn}}$ high-level width	<165> t_{WSK1H}	Input	90		ns
$\overline{\text{SCKn}}$ low-level width	<166> t_{WSK1L}	Input	90		ns
SIn setup time (to $\overline{\text{SCKn}}\uparrow$)	<167> t_{SSISK}		50		ns
SIn setup time (to $\overline{\text{SCKn}}\downarrow$)			50		ns
SIn hold time (from $\overline{\text{SCKn}}\uparrow$)	<168> t_{HSKSI}		50		ns
SIn hold time (from $\overline{\text{SCKn}}\downarrow$)			50		ns
SOn output delay time (from $\overline{\text{SCKn}}\downarrow$)	<169> t_{DSKSO}			50	ns
SOn output delay time (from $\overline{\text{SCKn}}\uparrow$)				50	ns
SOn output hold time (from $\overline{\text{SCKn}}\uparrow$)	<170> t_{HSKSO}		t_{WSK1H}		ns
SOn output hold time (from $\overline{\text{SCKn}}\downarrow$)			t_{WSK1H}		ns

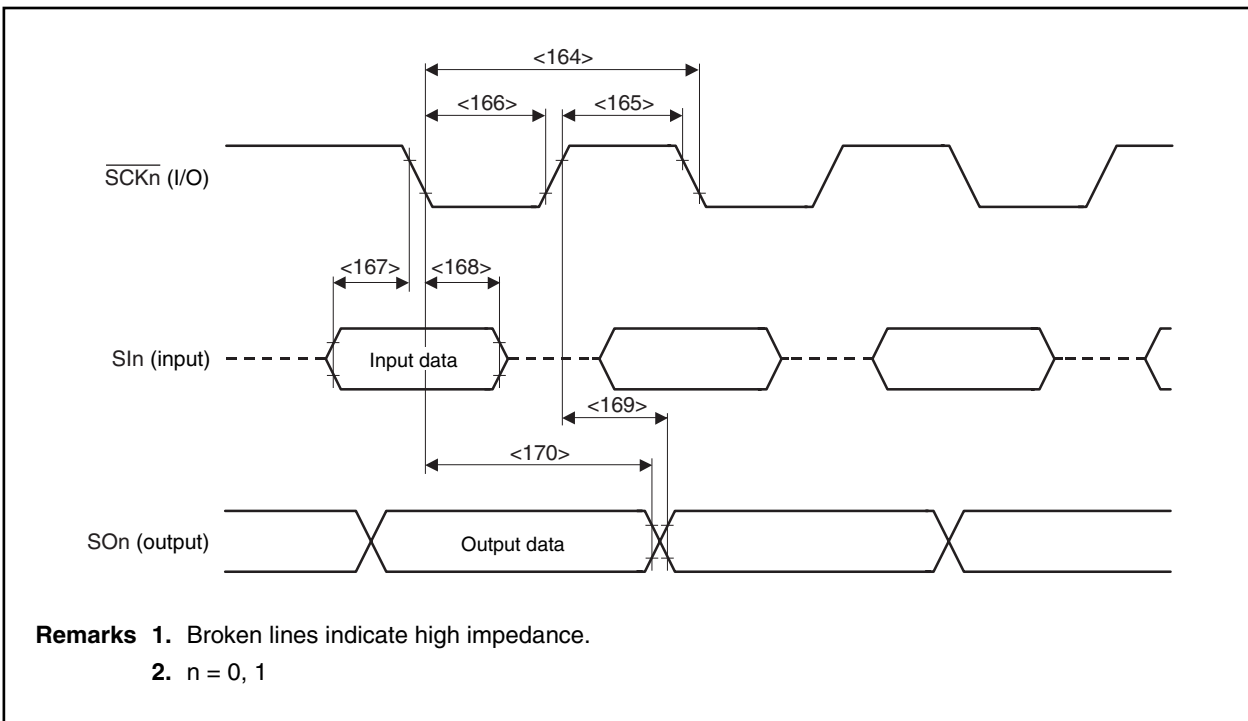
Remark n = 0, 1

(11) CSI0 to CSI1 timing (2/3)

(c) Timing when CKPn, DAPn bits of CSICn register = 00

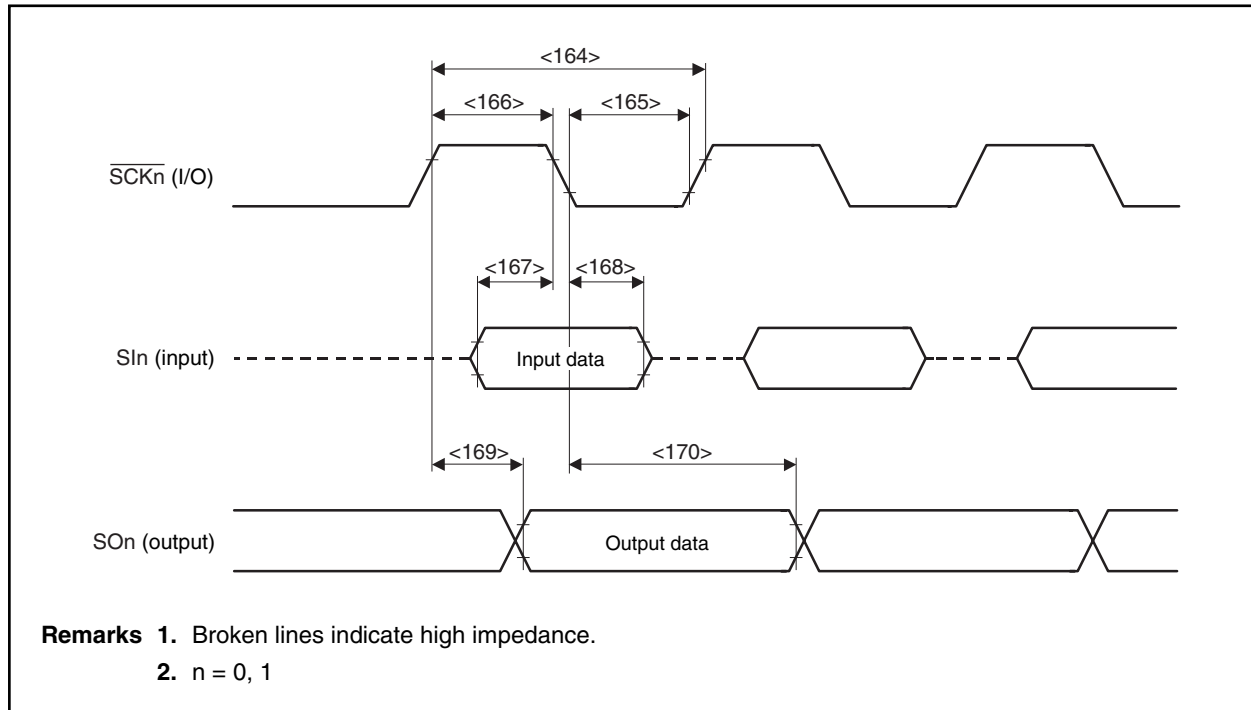


(d) Timing when CKPn, DAPn bits of CSICn register = 01

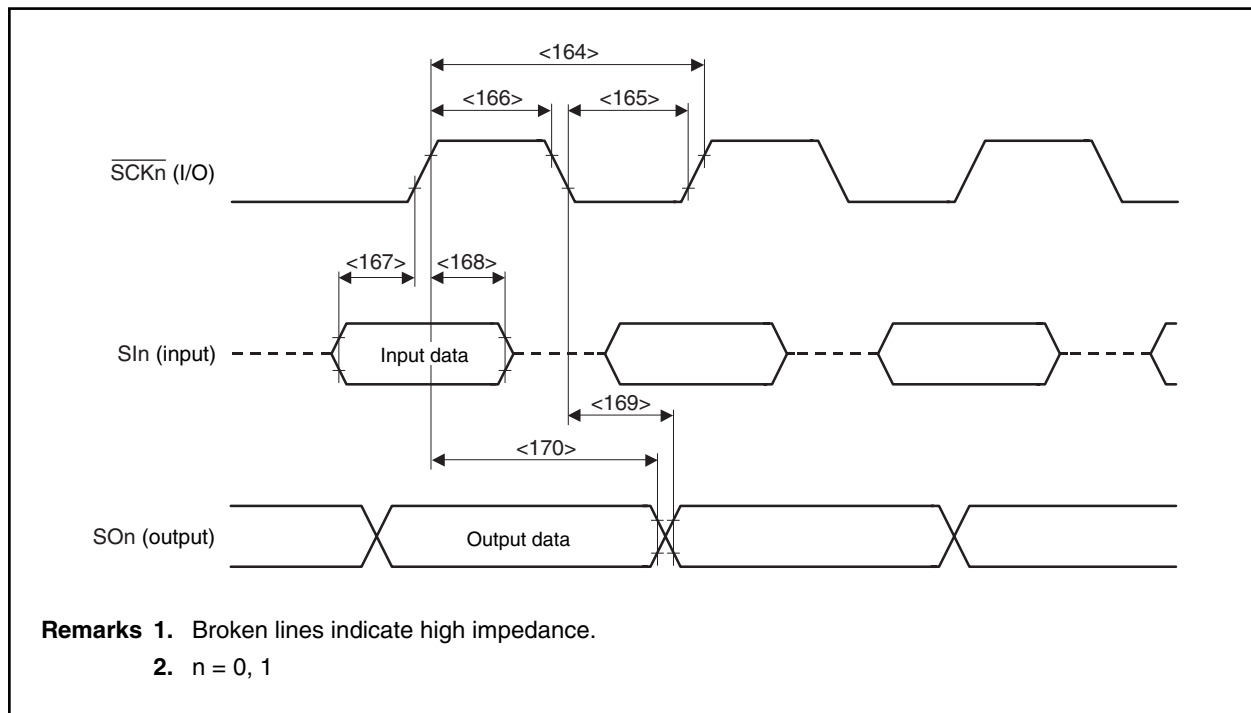


(11) CSI0, CSI1 timing (3/3)

(e) Timing when CKPn, DAPn bits of CSICn register = 10



(f) Timing when CKPn, DAPn bits of CSICn register = 11



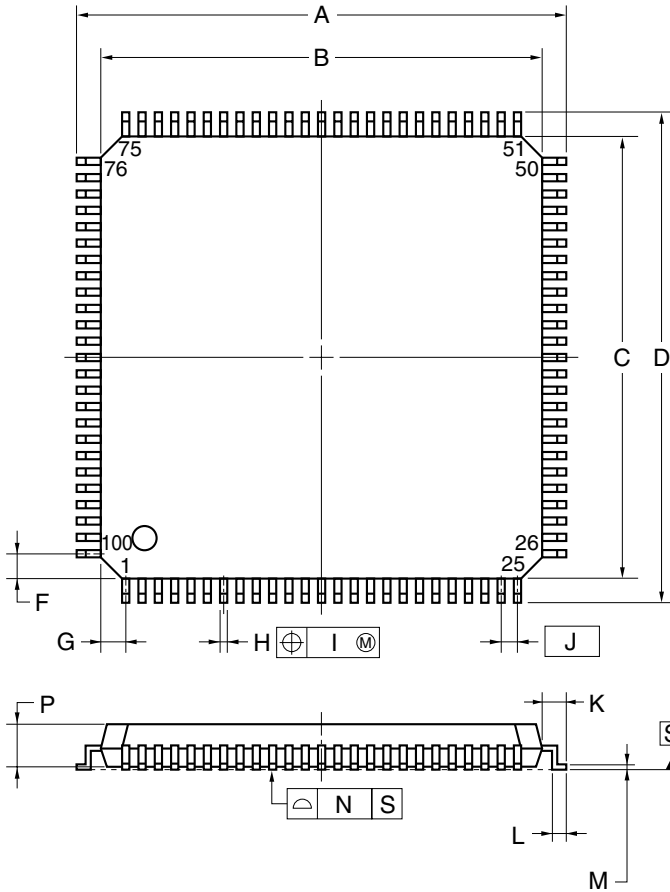
A/D Converter Characteristics ($T_A = -40$ to $+85^\circ\text{C}$, $V_{DD} = AV_{DD} = 3.0$ to 3.6 V, $V_{SS} = AV_{SS} = 0$ V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	–		10			bit
Overall error ^{Note 1}	–				± 0.49	%FSR
Quantization error	–				$\pm 1/2$	LSB
Conversion time	t_{CONV}		5		10	μs
Sampling time	t_{SAMP}		Conversion clock ^{Note 2} /6			ns
Zero-scale error ^{Note 1}	–				± 0.49	%FSR
Full-scale error ^{Note 1}	–				± 0.49	%FSR
Integral linearity error ^{Note 3}	–				± 4	LSB
Differential linearity error ^{Note 3}	–				± 4	LSB
Analog input voltage	V_{WASN}		-0.3		$AV_{REF} + 0.3$	V
AV_{REF} input voltage	AV_{REF}	$AV_{REF} = AV_{DD}$	3.0		3.6	V
AV_{DD} supply current	AI_{DD}				10	mA

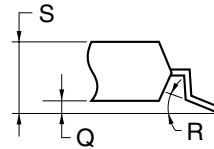
- Notes**
1. Excluding quantization error (± 0.05 %FSR)
 2. Conversion clock is the number of clocks set by the ADM1 register.
 3. Excluding quantization error (± 0.5 LSB)

Remark LSB: Least Significant Bit
FSR: Full Scale Range
%FSR is the ratio to the full-scale value.

100-PIN PLASTIC LQFP (FINE PITCH) (14x14)



detail of lead end



NOTE

Each lead centerline is located within 0.08 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	16.00±0.20
B	14.00±0.20
C	14.00±0.20
D	16.00±0.20
F	1.00
G	1.00
H	0.22 ^{+0.05} _{-0.04}
I	0.08
J	0.50 (T.P.)
K	1.00±0.20
L	0.50±0.20
M	0.17 ^{+0.03} _{-0.07}
N	0.08
P	1.40±0.05
Q	0.10±0.05
R	3° ^{+7°} _{-3°}
S	1.60 MAX.

S100GC-50-8EU, 8EA-2

CHAPTER 17 RECOMMENDED SOLDERING CONDITIONS

The V850E/MA2 should be soldered and mounted under the following recommended conditions.
For technical information, see the following website.

Semiconductor Device Mount Manual (<http://www.necel.com/pkg/en/mount/index.html>)

Table 17-1. Surface Mounting Type Soldering Conditions

μPD703108GC-8EU-A : 100-pin plastic LQFP (fine pitch) (14 × 14)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 260°C, Time: 60 seconds max. (at 220°C or higher), Count: Three times or less, Exposure limit: 7 days ^{Note} (after that, prebake at 125°C for 20 to 72 hours)	IR60-207-3
Wave soldering	For details, consult an NEC Electronics sales representative.	–
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

Note After opening the dry pack, store it at 25°C or less and 65% RH or less for the allowable storage period.

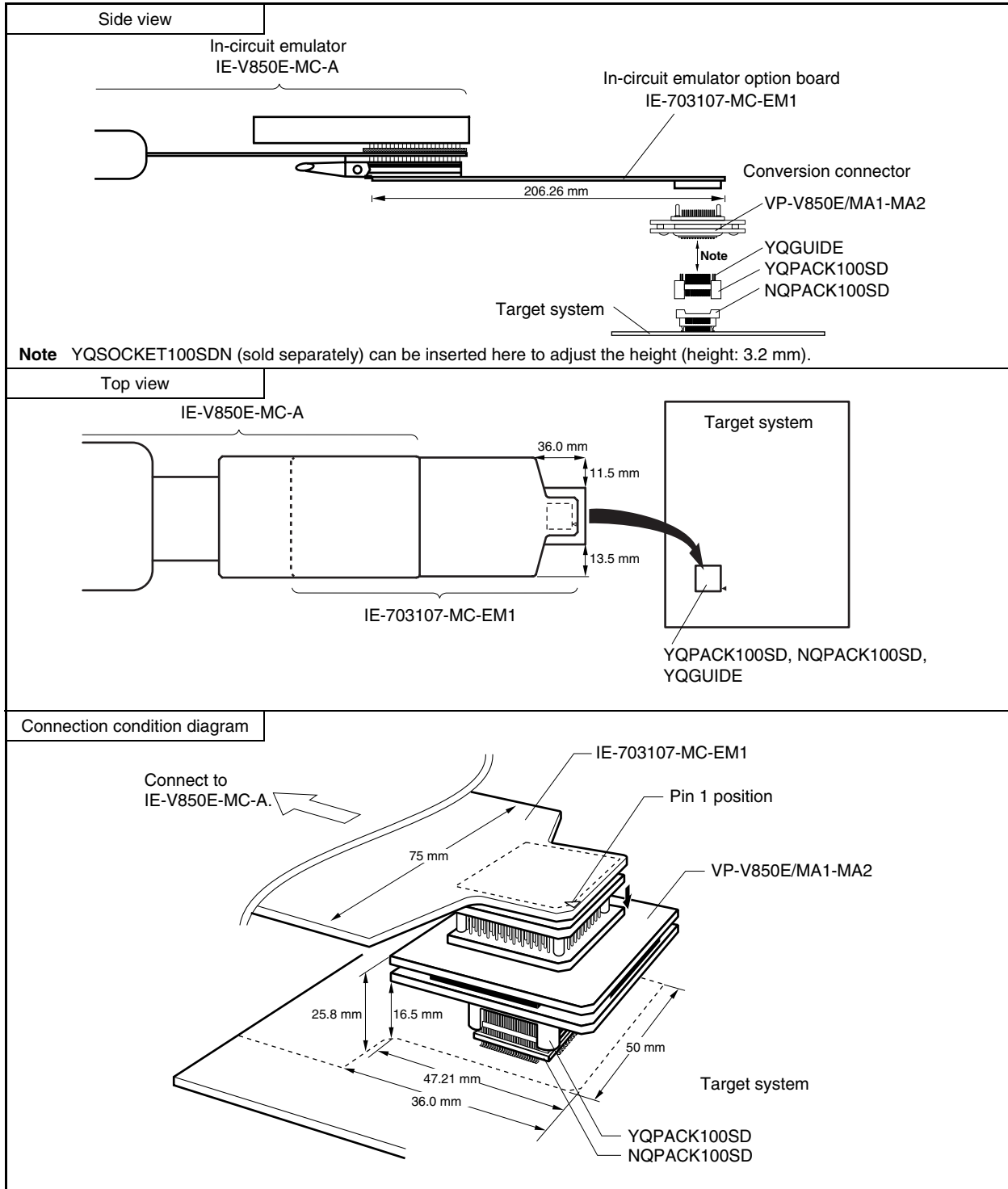
Caution Do not use different soldering methods together (except for partial heating).

- Remarks**
1. Products with -A at the end of the part number are lead-free products.
 2. For soldering methods and conditions other than those recommended above, consult an NEC Electronics sales representative.
 3. For soldering conditions for the μPD703108GC-8EU have not been determined.

APPENDIX A NOTES ON TARGET SYSTEM DESIGN

The following shows a diagram of the connection conditions between the in-circuit emulator option board and conversion connector. Design your system making allowances for conditions such as the form of parts mounted on the target system as shown below.

Figure A-1. 100-Pin Plastic LQFP (Fine Pitch) (14 × 14)



B.1 Restriction on Page ROM Access

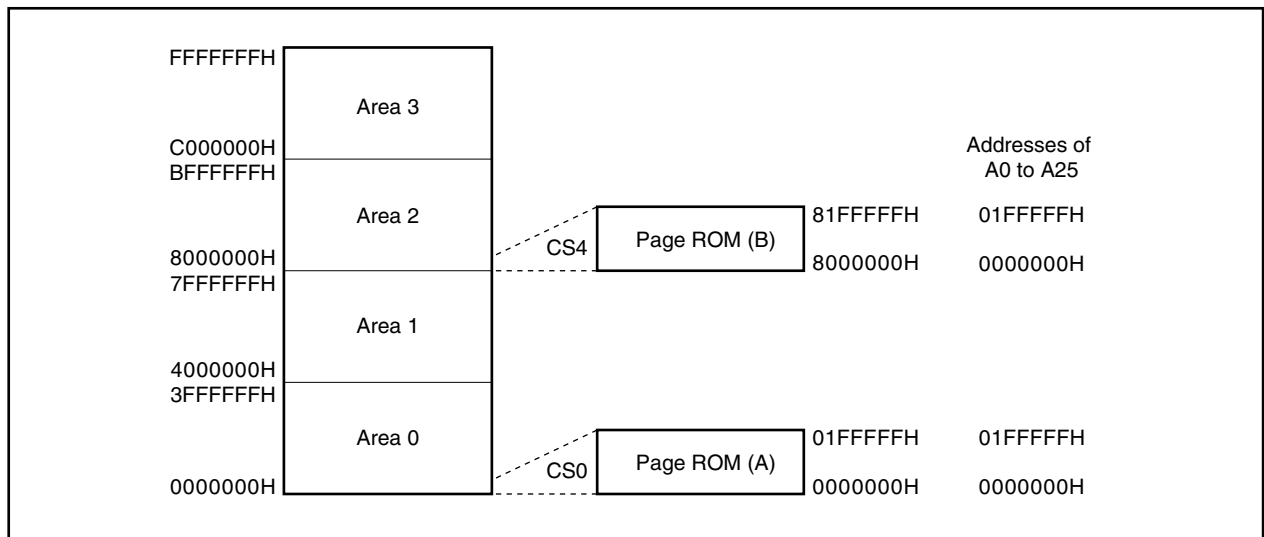
B.1.1 Description

In systems connecting multiple page ROMs to multiple different CSn spaces, when the page ROM of a different CSn space is continuously accessed immediately after a page ROM is accessed, if the value of the former address and that of the latter address are on the same page of the page ROM, even if the two CSn spaces are different, it is taken as access of the same page of the page ROM, and the on-page cycle is issued for the latter access (n = 0, 3, 4, 7). As a result, the data access time of the latter access is insufficient, making it impossible to perform normal reading.

Caution The page ROM has a page access function and includes memory that allows high-speed continuous access on the page (refer to Figure B-1).

For example, if the 8xxxxx2H address of the CS4 space is accessed immediately after the 0xxxxx0H address of the CS0 space is accessed, the on-page cycle is executed for 8xxxxx2H. (Refer to **Figure B-1**.)

Figure B-1. Example of Structure of Memory Map with Error



Examples of conditions under which an error does not occur are shown below.

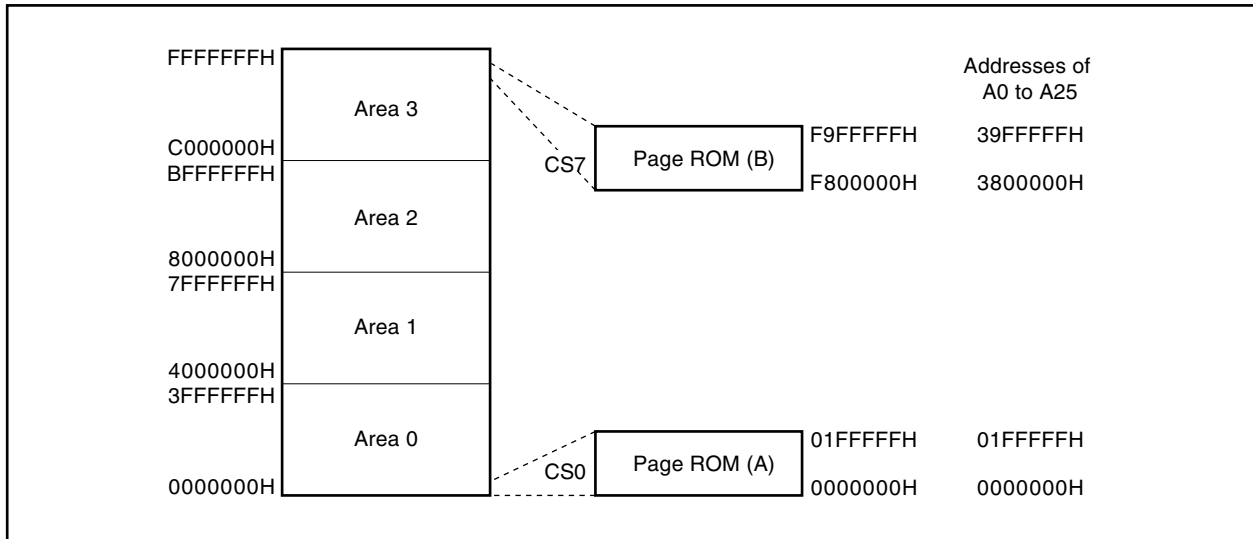
- ROM with page mode is not used.
- Only one ROM with page mode is used.
- The addresses of A0 to A25 do not overlap in all the ROMs with page mode used.

B.1.2 Countermeasures

When using several page ROMs, arrange the page ROMs so that the addresses of A0 to A25 do not overlap.

For example, when arranging two 2 MB page ROMs in different CSn spaces, set one page ROM to 0000000H to 01FFFFFFH, and the other page ROM to F800000H to F9FFFFFFH. (Refer to **Figure B-2.**)

Figure B-2. Example of Structure of Memory Map Preventing Error



APPENDIX C REGISTER INDEX

(1/5)

Register Symbol	Register Name	Unit	Page
ADCR0	A/D conversion result register 0 (10 bits)	ADC	340
ADCR0H	A/D conversion result register 0H (8 bits)	ADC	340
ADCR1	A/D conversion result register 1 (10 bits)	ADC	340
ADCR1H	A/D conversion result register 1H (8 bits)	ADC	340
ADCR2	A/D conversion result register 2 (10 bits)	ADC	340
ADCR2H	A/D conversion result register 2H (8 bits)	ADC	340
ADCR3	A/D conversion result register 3 (10 bits)	ADC	340
ADCR3H	A/D conversion result register 3H (8 bits)	ADC	340
ADIC	Interrupt control register	INTC	212
ADM0	A/D converter mode register 0	ADC	336
ADM1	A/D converter mode register 1	ADC	338
ADM2	A/D converter mode register 2	ADC	339
ASC	Address setup wait control register	BCU	96
ASIF0	Asynchronous serial interface transmission status register 0	UART0	297
ASIF1	Asynchronous serial interface transmission status register 1	UART1	297
ASIM0	Asynchronous serial interface mode register 0	UART0	293
ASIM1	Asynchronous serial interface mode register 1	UART1	293
ASIS0	Asynchronous serial interface status register 0	UART0	296
ASIS1	Asynchronous serial interface status register 1	UART1	296
BCC	Bus cycle control register	BCU	99
BCT0	Bus cycle type configuration register 0	BCU	77
BCT1	Bus cycle type configuration register 1	BCU	77
BEC	Endian configuration register	BCU	80
BRGC0	Baud rate generator control register 0	BRG0	315
BRGC1	Baud rate generator control register 1	BRG1	315
BSC	Bus size configuration register	BCU	79
CCC00	Capture/compare register C00	RPU	258
CCC01	Capture/compare register C01	RPU	258
CCC10	Capture/compare register C10	RPU	258
CCC11	Capture/compare register C11	RPU	258
CKC	Clock control register	CG	234
CKSR0	Clock select register 0	UART0	314
CKSR1	Clock select register 1	UART1	314
CMD0	Compare register D0	RPU	282
CMD1	Compare register D1	RPU	282
CMD2	Compare register D2	RPU	282
CMD3	Compare register D3	RPU	282
CMICD0	Interrupt control register	INTC	212

Register Symbol	Register Name	Unit	Page
CMICD1	Interrupt control register	INTC	212
CMICD2	Interrupt control register	INTC	212
CMICD3	Interrupt control register	INTC	212
CSC0	Chip area selection control register 0	BCU	76
CSC1	Chip area selection control register 1	BCU	76
CSIC0	Clocked serial interface clock selection register 0	CSI0	324
CSIC1	Clocked serial interface clock selection register 1	CSI1	324
CSIIC0	Interrupt control register	INTC	212
CSIIC1	Interrupt control register	INTC	212
CSIM0	Clocked serial interface mode register 0	CSI0	322
CSIM1	Clocked serial interface mode register 1	CSI1	322
DADC0	DMA addressing control register 0	DMAC	164
DADC1	DMA addressing control register 1	DMAC	164
DADC2	DMA addressing control register 2	DMAC	164
DADC3	DMA addressing control register 3	DMAC	164
DBC0	DMA byte count register 0	DMAC	163
DBC1	DMA byte count register 1	DMAC	163
DBC2	DMA byte count register 2	DMAC	163
DBC3	DMA byte count register 3	DMAC	163
DCHC0	DMA channel control register 0	DMAC	166
DCHC1	DMA channel control register 1	DMAC	166
DCHC2	DMA channel control register 2	DMAC	166
DCHC3	DMA channel control register 3	DMAC	166
DDA0H	DMA destination address register 0H	DMAC	161
DDA0L	DMA destination address register 0L	DMAC	162
DDA1H	DMA destination address register 1H	DMAC	161
DDA1L	DMA destination address register 1L	DMAC	162
DDA2H	DMA destination address register 2H	DMAC	161
DDA2L	DMA destination address register 2L	DMAC	162
DDA3H	DMA destination address register 3H	DMAC	161
DDA3L	DMA destination address register 3L	DMAC	162
DDIS	DMA disable status register	DMAC	168
DMAIC0	Interrupt control register	INTC	212
DMAIC1	Interrupt control register	INTC	212
DMAIC2	Interrupt control register	INTC	212
DMAIC3	Interrupt control register	INTC	212
DRST	DMA restart register	DMAC	168
DSA0H	DMA source address register 0H	DMAC	159
DSA0L	DMA source address register 0L	DMAC	160
DSA1H	DMA source address register 1H	DMAC	159

Register Symbol	Register Name	Unit	Page
DSA1L	DMA source address register 1L	DMAC	160
DSA2H	DMA source address register 2H	DMAC	159
DSA2L	DMA source address register 2L	DMAC	160
DSA3H	DMA source address register 3H	DMAC	159
DSA3L	DMA source address register 3L	DMAC	160
DTFR0	DMA trigger factor register 0	DMAC	170
DTFR1	DMA trigger factor register 1	DMAC	170
DTFR2	DMA trigger factor register 2	DMAC	170
DTFR3	DMA trigger factor register 3	DMAC	170
DTOC	DMA terminal count output control register	DMAC	169
DWC0	Data wait control register 0	BCU	94
DWC1	Data wait control register 1	BCU	94
IMR0	Interrupt mask register 0	INTC	213
IMR1	Interrupt mask register 1	INTC	213
IMR2	Interrupt mask register 2	INTC	213
IMR3	Interrupt mask register 3	INTC	213
INTM0	External interrupt mode register 0	INTC	203
INTM1	External interrupt mode register 1	INTC	216
INTM2	External interrupt mode register 2	INTC	216
ISPR	In-service priority register	INTC	215
LOCKR	Lock register	CPU	237
OVIC00	Interrupt control register	INTC	212
OVIC01	Interrupt control register	INTC	212
P0	Port 0	Port	381
P00IC0	Interrupt control register	INTC	212
P00IC1	Interrupt control register	INTC	212
P01IC0	Interrupt control register	INTC	212
P01IC1	Interrupt control register	INTC	212
P1	Port 1	Port	384
P10IC0	Interrupt control register	INTC	212
P10IC1	Interrupt control register	INTC	212
P11IC0	Interrupt control register	INTC	212
P2	Port 2	Port	386
P4	Port 4	Port	388
P7	Port 7	Port	391
PAH	Port AH	Port	394
PAL	Port AL	Port	391
PBD	Port BD	Port	407
PCD	Port CD	Port	404
PCM	Port CM	Port	402

Register Symbol	Register Name	Unit	Page
PCS	Port CS	Port	398
PCT	Port CT	Port	400
PDL	Port DL	Port	396
PFC0	Port 0 function control register	Port	383
PFC2	Port 2 function control register	Port	387
PFC4	Port 4 function control register	Port	390
PFCCD	Port CD function control register	Port	406
PHCMD	Peripheral command register	CPU	233
PHS	Peripheral status register	CPU	236
PM0	Port 0 mode register	Port	381
PM1	Port 1 mode register	Port	384
PM2	Port 2 mode register	Port	386
PM4	Port 4 mode register	Port	388
PMAH	Port AH mode register	Port	395
PMAL	Port AL mode register	Port	393
PMBD	Port BD mode register	Port	407
PMC0	Port 0 mode control register	Port	382
PMC1	Port 1 mode control register	Port	385
PMC2	Port 2 mode control register	Port	387
PMC4	Port 4 mode control register	Port	389
PMCAH	Port AH mode control register	Port	395
PMCAL	Port AL mode control register	Port	393
PMCBD	Port BD mode control register	Port	408
PMCCD	Port CD mode control register	Port	405
PMCCM	Port CM mode control register	Port	403
PMCCS	Port CS mode control register	Port	399
PMCCT	Port CT mode control register	Port	401
PMCD	Port CD mode register	Port	404
PMCDL	Port DL mode control register	Port	397
PMCM	Port CM mode register	Port	402
PMCS	Port CS mode register	Port	398
PMCT	Port CT mode register	Port	400
PMDL	Port DL mode register	Port	396
PRC	Page ROM configuration register	MEMC	123
PRCMD	Command register	CPU	240
PSC	Power save control register	CPU	241
PSMR	Power save mode register	CPU	240
RFS3	SDRAM refresh control register 3	MEMC	147
RFS4	SDRAM refresh control register 4	MEMC	147
RXB0	Receive buffer register 0	UART0	298

Register Symbol	Register Name	Unit	Page
RXB1	Receive buffer register 1	UART1	298
SCR3	SDRAM configuration register 3	MEMC	131
SCR4	SDRAM configuration register 4	MEMC	131
SEIC0	Interrupt control register	INTC	212
SEIC1	Interrupt control register	INTC	212
SESC0	Valid edge selection register C0	INTC	218, 264
SESC1	Valid edge selection register C1	INTC	218, 264
SIO0	Serial I/O shift register 0	CSI0	326
SIO1	Serial I/O shift register 1	CSI1	326
SIOE0	Receive-only serial I/O shift register 0	CSI0	326
SIOE1	Receive-only serial I/O shift register 1	CSI1	326
SOTB0	Clocked serial interface transmit buffer register 0	CSI0	327
SOTB1	Clocked serial interface transmit buffer register 1	CSI1	327
SRIC0	Interrupt control register	INTC	212
SRIC1	Interrupt control register	INTC	212
STIC0	Interrupt control register	INTC	212
STIC1	Interrupt control register	INTC	212
TMC0	Timer C0	RPU	256
TMC1	Timer C1	RPU	256
TMCC00	Timer mode control register C00	RPU	260
TMCC01	Timer mode control register C01	RPU	262
TMCC10	Timer mode control register C10	RPU	260
TMCC11	Timer mode control register C11	RPU	262
TMCD0	Timer mode control register D0	RPU	284
TMCD1	Timer mode control register D1	RPU	284
TMCD2	Timer mode control register D2	RPU	284
TMCD3	Timer mode control register D3	RPU	284
TMD0	Timer D0	RPU	281
TMD1	Timer D1	RPU	281
TMD2	Timer D2	RPU	281
TMD3	Timer D3	RPU	281
TXB0	Transmit buffer register 0	UART0	299
TXB1	Transmit buffer register 1	UART1	299
VSWC	System wait control register	BCU	72

APPENDIX D INSTRUCTION SET LIST

D.1 Conventions

(1) Register symbols used to describe operands

Register Symbol	Explanation
reg1	General-purpose registers: Used as source registers.
reg2	General-purpose registers: Used mainly as destination registers. Also used as source register in some instructions.
reg3	General-purpose registers: Used mainly to store the remainders of division results and the higher order 32 bits of multiplication results.
bit#3	3-bit data for specifying the bit number
immX	X bit immediate data
dispX	X bit displacement data
regID	System register number
vector	5-bit data that specifies the trap vector (00H to 1FH)
cccc	4-bit data that shows the conditions code
sp	Stack pointer (SP)
ep	Element pointer (r30)
listX	X item register list

(2) Register symbols used to describe opcodes

Register Symbol	Explanation
R	1-bit data of a code that specifies reg1 or regID
r	1-bit data of the code that specifies reg2
w	1-bit data of the code that specifies reg3
d	1-bit displacement data
l	1-bit immediate data (indicates the higher bits of immediate data)
i	1-bit immediate data
cccc	4-bit data that shows the condition codes
CCCC	4-bit data that shows the condition codes of Bcond instruction
bbb	3-bit data for specifying the bit number
L	1-bit data that specifies a program register in the register list
S	1-bit data that specifies a system register in the register list

(3) Register symbols used in operations

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 size b data from address a.
store-memory (a, b, c)	Write data b into address a in size c.
load-memory-bit (a, b)	Read bit b of address a.
store-memory-bit (a, b, c)	Write c to bit b of address a.
saturated (n)	Execute saturated processing of n (n is a 2's complement). If, as a result of calculations, n ≥ 7FFFFFFFH, let it be 7FFFFFFFH. n ≤ 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 execution clock

Register Symbol	Explanation
i	If executing another instruction immediately after executing the first instruction (issue).
r	If repeating execution of the same instruction immediately after executing the first instruction (repeat).
l	If using the results of instruction execution in the instruction immediately after the execution (latency).

(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

D.2 Instruction Set (In Alphabetical Order)

(1/6)

Mnemonic	Operand	Opcode	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	×	×	×	×		
	imm5,reg2	rrrrr010010iiii	GR[reg2]←GR[reg2]+sign-extend(imm5)	1	1	1	×	×	×	×		
ADDI	imm16,reg1,reg2	rrrrr110000RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1	×	×	×	×		
AND	reg1,reg2	rrrrr001010RRRRR	GR[reg2]←GR[reg2]AND GR[reg1]	1	1	1		0	×	×		
ANDI	imm16,reg1,reg2	rrrrr110110RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]AND zero-extend(imm16)	1	1	1		0	0	×		
★ Bcond	disp9	dddd1011dddcccc Note 1	if conditions are satisfied then PC←PC+sign-extend(disp9)	When conditions are satisfied	3	3	3					
				When conditions are not satisfied	Note 2	Note 2	Note 2					
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	×	0	×	×		
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	×	0	×	×		
★ 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(adr,Half-word))	5	5	5						
CLR1	bit#3, disp16[reg1]	10bbb111110RRRRR ddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,0)	3	3	3				×		
	reg2,[reg1]	rrrrr11111RRRRR 000000011100100	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,0)	Note 3	Note 3	Note 3				×		
CMOV	cccc,imm5,reg2,reg3	rrrrr11111iiii wwwww01100cccc0	if conditions are satisfied then GR[reg3]←sign-extended(imm5) else GR[reg3]←GR[reg2]	1	1	1						
	cccc,reg1,reg2,reg3	rrrrr11111RRRRR 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	×	×	×	×		
	imm5,reg2	rrrrr010011iiii	result←GR[reg2]-sign-extend(imm5)	1	1	1	×	×	×	×		
★ CTRET		000001111100000 0000000101000100	PC←CTPC PSW←CTPSW	4	4	4	R	R	R	R	R	
★ DBRET		000001111100000 0000000101000110	PC←DBPC PSW←DBPSW	4	4	4	R	R	R	R	R	

APPENDIX D INSTRUCTION SET LIST

(2/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags					
				i	r	l	CY	OV	S	Z	SAT	
★ DBTRAP		1111100001000000	DBPC←PC+2 (returned PC) DBPSW←PSW PSW.NP←1 PSW.EP←1 PSW.ID←1 PC←00000060H	4	4	4						
DI		000001111100000 0000000101100000	PSW.ID←1	1	1	1						
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 Note 5	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] ^{Note 6}	35	35	35		×	×	×		
	reg1,reg2,reg3	rrrrr11111RRRRR wwwww01010000000	GR[reg2]←GR[reg2]÷GR[reg1] ^{Note 6} GR[reg3]←GR[reg2]%GR[reg1]	35	35	35		×	×	×		
DIVHU	reg1,reg2,reg3	rrrrr11111RRRRR wwwww01010000010	GR[reg2]←GR[reg2]÷GR[reg1] ^{Note 6} 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	rrrrr11110dddd dddddddddddddd0 Note 7	GR[reg2]←PC+4 PC←PC+sign-extend(disp22)	3	3	3						
★ JMP	[reg1]	0000000011RRRRR	PC←GR[reg1]	4	4	4						
★ JR	disp22	0000011110dddd dddddddddddddd0 Note 7	PC←PC+sign-extend(disp22)	3	3	3						
LD.B	disp16[reg1],reg2	rrrrr11100RRRRR ddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 11						
LD.BU	disp16[reg1],reg2	rrrrr11110bRRRRR ddddddddddddddd1 Notes 8, 10	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	Note 11						

APPENDIX D INSTRUCTION SET LIST

(3/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags					
				i	r	l	CY	OV	S	Z	SAT	
LD.H	disp16[reg1],reg2	rrrrr111001RRRRR dddddddddddddd0 Note 8	adr←GR[reg1]+sign-extend(dispatch16) GR[reg2]←sign-extend(Load-memory(adr,Half-word))	1	1	Note 11						
LDSR	reg2,regID	rrrrr111111RRRRR 0000000000100000 Note 12	SR[regID]←GR[reg2]	Other than regID = PSW	1	1	1					
				regID = PSW	1	1	1	×	×	×	×	×
LD.HU	disp16[reg1],reg2	rrrrr111111RRRRR ddddddddddddddd1 Note 8	adr←GR[reg1]+sign-exend(dispatch16) GR[reg2]←zero-extend(Load-memory(adr,Half-word))	1	1	Note 11						
LD.W	disp16[reg1],reg2	rrrrr111001RRRRR ddddddddddddddd1 Note 8	adr←GR[reg1]+sign-exend(dispatch16) GR[reg2]←Load-memory(adr,Word)	1	1	Note 9						
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 0 ¹⁶)	1	1	1						
MUL	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100000	GR[reg3] GR[reg2]←GR[reg2]xGR[reg1] reg1 ≠ reg2 ≠ reg3, reg3 ≠ r0	1	2	2						
	imm9,reg2,reg3	rrrrr111111iiii wwwww01001111100	GR[reg3] GR[reg2]←GR[reg2]xsign-extend(imm9)	1	2	2						
MULH	reg1,reg2	rrrrr000111RRRRR	GR[reg2]←GR[reg2] ^{Note 5} xGR[reg1] ^{Note 6}	1	1	2						
	imm5,reg2	rrrrr010111iiii	GR[reg2]←GR[reg2] ^{Note 6} xsign-extend(imm5)	1	1	2						
MULHI	imm16,reg1,reg2	rrrrr110111RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] ^{Note 6} ximm16	1	1	2						
MULU	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100010	GR[reg3] GR[reg2]←GR[reg2]xGR[reg1] reg1 ≠ reg2 ≠ reg3, reg3 ≠ r0	1	2	2						
	imm9,reg2,reg3	rrrrr111111iiii wwwww01001111110	GR[reg3] GR[reg2]←GR[reg2]xzero-extend(imm9)	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	×	×		
NOT1	bit#3,disp16[reg1]	01bbb111110RRRRR ddddddddddddddd	adr←GR[reg1]+sign-extend(dispatch16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,Z flag)	3	3	3					×	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100010	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,Z flag)	3	3	3					×	

★

★

APPENDIX D INSTRUCTION SET LIST

(4/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
OR	reg1,reg2	rrrrr001000RRRRR	GR[reg2]←GR[reg2]OR GR[reg1]	1	1	1		0	×	×	
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 ^{Note 15}	0000011110iiiiL LLLLLLLLLLLLff011 imm16/imm32 Note 16	Store-memory(sp-4,GR[reg in list12],Word) GR[reg in list 12]←Load-memory(sp,Word) sp←sp+4 repeat 2 step above until all regs in list12 is loaded PC←GR[reg1]	n+2 Note 4	n+2 Note 4	n+2 Note 4					
★ RETI		000001111100000 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	4	4	4	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					

APPENDIX D INSTRUCTION SET LIST

(5/6)

Mnemonic	Operand	Opcode	Operation	Execution Clock			Flags				
				i	r	l	CY	OV	S	Z	SAT
SET1	bit#3,disp16[reg1]	00bbb11110RRRRR 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				×	
	reg2,[reg1]	rrrrr11111RRRRR 000000011100000	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				×	
SHL	reg1,reg2	rrrrr11111RRRRR 000000011000000	GR[reg2]←GR[reg2] logically shift left by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010110iiii	GR[reg2]←GR[reg2] logically shift left by zero-extend(imm5)	1	1	1	×	0	×	×	
SHR	reg1,reg2	rrrrr11111RRRRR 000000010000000	GR[reg2]←GR[reg2] logically shift right by GR[reg1]	1	1	1	×	0	×	×	
	imm5,reg2	rrrrr010100iiii	GR[reg2]←GR[reg2] logically shift right by zero-extend(imm5)	1	1	1	×	0	×	×	
SLD.B	disp7[ep],reg2	rrrrr0110dddddd	adr←ep+zero-extend(disp7) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 9					
SLD.BU	disp4[ep],reg2	rrrrr0000110dddd Note 18	adr←ep+zero-extend(disp4) GR[reg2]←zero-extend(Load-memory(adr,Byte))	1	1	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	Note 9					
SLD.HU	disp5[ep],reg2	rrrrr0000111dddd Notes 18, 20	adr←ep+zero-extend(disp5) GR[reg2]←zero-extend(Load-memory(adr,Half- word))	1	1	Note 9					
SLD.W	disp8[ep],reg2	rrrrr1010dddddd0 Note 21	adr←ep+zero-extend(disp8) GR[reg2]←Load-memory(adr,Word)	1	1	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	rrrrr11111RRRRR 000000001000000	GR[reg2]←SR[regID]	1	1	1					

Mnemonic	Operand	Opcode	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	0000000101RRRRR	GR[reg1]←sign-extend (GR[reg1] (7 : 0))	1	1	1					
SXH	reg1	0000000111RRRRR	GR[reg1]←sign-extend (GR[reg1] (15 : 0))	1	1	1					
★ TRAP	vector	0 0 0 0 0 1 1 1 1 1 1 i i i i 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0	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)	4	4	4					
TST	reg1,reg2	rrrrr001011RRRRR	result←GR[reg2] AND GR[reg1]	1	1	1		0	×	×	
TST1	bit#3,disp16[reg1]	11bbb111110RRRRR ddddddddddddddd	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 i i i i i i i i i i i i i i i i	GR[reg2]←GR[reg1] XOR zero-extend (imm16)	1	1	1		0	×	×	
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. 4 if there is an instruction that rewrites the contents of the PSW immediately before.
- 3. If there is no wait state (3 + the number of read access wait states).
- ★ 4. n is the total number of list12 load registers. (According to the number of wait states. Also, if there are no wait states, n is the number of list12 registers. If n = 0, same operation as when n = 1)
- 5. RRRRRR: other than 00000.
- 6. The lower half word data only are valid.
- 7. dddddddddddddddddddd: 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 opcode. Therefore, the meaning of register specification in the mnemonic description and in the opcode differs from other instructions.

rrrrr = regID specification

RRRRR = reg2 specification

13. iiii: Lower 5 bits of imm9.

IIII: Lower 4 bits of imm9.

14. In the case of reg2 = reg3 (the lower 32 bits of the results are not written in the register) or reg3 = r0 (the higher 32 bits of the results are not written in the register), shortened by 1 clock.

15. sp/imm: specified by bits 19 and 20 of the sub-opcode.

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 clocks.

18. rrrrr : Other than 00000.

19. ddddddd: Higher 7 bits of disp8.

20. dddd: Higher 4 bits of disp5.

21. ddddddd: Higher 6 bits of disp8.

APPENDIX E INDEX

[A]

A/D conversion result registers 0 to 3	340
A/D conversion result registers 0H to 3H	340
A/D converter characteristics	362
A/D converter mode register 0	336
A/D converter mode register 1	338
A/D converter mode register 2	339
A/D converter operation	342
A0 to A15	45
A16 to A24	44
AC characteristics	418
ADCR0 to ADCR3	340
ADCR0H to ADCR3H	340
Address multiplex function	129
Address setup wait control register	96
Address space	56
ADIC	212
ADM0	336
ADM1	338
ADM2	339
ANI0 to ANI3	40
Applications	27
Area	60
ASC	96
ASIF0, ASIF1	297
ASIM0, ASIM1	293
ASIS0, ASIS1	296
Asynchronous serial interface mode registers 0, 1	293
Asynchronous serial interface status registers 0, 1	296
Asynchronous serial interface transmission status registers 0, 1	297
Asynchronous serial interfaces 0, 1	290
AV _{DD}	46
AV _{REF}	46
AV _{SS}	46

[B]

Basic configuration of timer C	255
Basic configuration of timer D	280
Baud rate generator control registers 0, 1	315
BCC	99
BCT0, BCT1	77
BEC	80
Block transfer mode	177

Boundary operation conditions	109
BRG0, BRG1	313
BRGC0, BRGC1	315
BSC	79
Bus access	79
Bus control pins	74
Bus cycle control register	99
Bus cycle type configuration registers 0, 1	77
Bus cycle type control function	77
Bus cycles in which wait function is valid	98
Bus hold function	100
Bus hold procedure	101
Bus hold timing (SDRAM)	104
Bus hold timing (SRAM)	102
Bus priority order	108
Bus size configuration register	79
Bus sizing function	79
Bus width	83

[C]

Capture/compare registers C00, 01, 10, 11	258
Cautions (Page ROM access)	447
CCC00, 01, 10, 11	258
Chip area selection control registers 0, 1	76
Chip select control function	76
CKC	234
CKSEL	45
CKSR0, CKSR1	314
CLKOUT	41
Clock control register	234
Clock select registers 0, 1	314
Clocked serial interface clock selection registers 0, 1	324
Clocked serial interface mode registers 0, 1	322
Clocked serial interface transmit buffer registers 0, 1	327
Clocked serial interfaces 0, 1	321
CMD0 to CMD3	282
CMICD0 to CMICD3	212
Command register	240
Compare match interrupt when in timer trigger mode	359
Compare registers D0 to D3	282
Conversion time	365
CPU address space	56
CPU register set	51

$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$, $\overline{CS7}$	43	DTFR0 to DTFR3	170
CSC0, CSC1	76	DTOC	169
CSI0, CSI1	321	DWC0, DWC1	94
CSIC0, CSIC1	324	[E]	
CSIIC0, CSIIC1	212	Edge detection function	
CSIM0, CSIM1	322	(non-maskable interrupt)	203
CV _{DD}	46	Electrical specifications	414
CV _{SS}	46	Endian configuration register	80
[D]		Endian control function	80
D0 to D15	45	EP	221
DADC0 to DADC3	164	Exception status flag	221
Data retention characteristics	417	Exception trap	222
Data space	109	External bus cycles during DMA transfer	188
Data wait control registers 0, 1	94	External interrupt mode register 0	203
DBC0 to DBC3	163	External interrupt mode registers 1, 2	216
DC characteristics	416	External memory expansion	64
DCHC0 to DCHC3	166	External wait function	97
DDA0H to DDA3H	161	[F]	
DDA0L to DDA3L	162	Forcible interrupt	192
DDIS	168	Forcible termination	193
Debug trap	224	Full-scale error	364
Dedicated baud rate generators 0, 1	313	[H]	
Description of pin functions	38	HALT mode	243
Differential linearity error	364	\overline{HLDAK}	41
Direct mode	232	\overline{HLDRQ}	42
DMA addressing control registers 0 to 3	164	[I]	
DMA bus states	172	ID	215
DMA byte count registers 0 to 3	163	IDLE mode	245
DMA channel control registers 0 to 3	166	Idle state insertion function	99
DMA channel priorities	189	Illegal opcode definition	222
DMA destination address registers 0H to 3H	161	Image	57
DMA destination address registers 0L to 3L	162	IMR0 to IMR3	213
DMA disable status register	168	Input clock selection	232
DMA restart register	168	In-service priority register	215
DMA source address registers 0H to 3H	159	Integral linearity error	365
DMA source address registers 0L to 3L	160	Internal block diagram	30
DMA terminal count output control register	169	Interrupt control register	211
DMA transfer end	196	Interrupt latency time	228
DMA transfer start factors	190	Interrupt mask registers 0 to 3	213
DMA trigger factor registers 0 to 3	170	Interrupt trigger mode selection	216
$\overline{DMAAK0}$, $\overline{DMAAK1}$	41	INTM0	203
DMAC bus cycle state transition	173	INTM1, INTM2	216
DMAIC0 to DMAIC3	212	INTP000, INTP001	38
$\overline{DMARQ0}$, $\overline{DMARQ1}$	38	INTP010, INTP011	39
DRST	168	INTP100, INTP101	38
DSA0H to DSA3H	159		
DSA0L to DSA3L	160		

INTP110.....	39	[P]	
ISPR.....	215	P0.....	381
[L]		P00IC0, P00IC1.....	212
<u>LBE</u>	44	P01 to P05.....	38
LDQM.....	43	P01IC0, P01IC1.....	212
List of pin function.....	33	P1.....	384
Lock register.....	237	P10IC0, P10IC1.....	212
LOCKR.....	237	P11, P12.....	39
<u>LWR</u>	42	P11IC0.....	212
[M]		P2.....	386
Maskable interrupt status flag.....	215	P20, P24.....	39
Maskable interrupts.....	204	P4.....	388
Maximum response time for DMA transfer request.....	194	P40 to P45.....	40
Memory block function.....	75	P7.....	391
Memory map.....	59	P70 to P73.....	40
MODE0 to MODE2.....	45	Package drawing.....	444
Multiple interrupt servicing control.....	226	Page ROM access.....	124
[N]		Page ROM configuration register.....	123
Next address setting function.....	189	Page ROM connection.....	120
NMI.....	39	Page ROM controller.....	119
Noise elimination (maskable interrupt).....	216	PAH.....	394
Noise elimination (non-maskable interrupt).....	203	PAH0 to PAH8.....	44
Non-maskable interrupt.....	199	PAL.....	391
Non-maskable interrupt status flag.....	203	PAL0 to PAL15.....	45
Notes on target system design.....	446	PBD.....	407
NP.....	203	PBD0, PBD1.....	41
Number of access clocks.....	79	PCD.....	404
[O]		PCD0 to PCD3.....	43
On-chip units.....	30	PCM.....	402
One-time transfer during single transfer via <u>DMARQ0</u> , <u>DMARQ1</u> signals.....	195	PCM0 to PCM4.....	41
On-page/off-page judgment.....	121	PCS.....	398
Operation in A/D trigger mode.....	348	PCS0, PCS3, PCS4, PCS7.....	43
Operation in power save mode.....	101	PCT.....	400
Operation in standby mode.....	358	PCT0, PCT1, PCT4, PCT5.....	42
Operation in timer trigger mode.....	351	PDL.....	396
Operation mode and trigger mode.....	343	PDL0 to PDL15.....	45
Operation mode specification.....	55	Periods in which interrupts are not acknowledged.....	229
Operation modes.....	55	Peripheral command register.....	233
Ordering information.....	27	Peripheral I/O registers.....	66
Overall error.....	362	Peripheral status register.....	236
OVIC00, OVIC01.....	212	PFC0.....	383
		PFC2.....	387
		PFC4.....	390
		PFCCD.....	406
		PHCMD.....	233
		PHS.....	236
		Pin configuration.....	28

Pin I/O circuits.....	49	Port AL	391
Pin I/O circuits and recommended connection of unused pins	47	Port AL mode control register.....	393
Pin status	37	Port AL mode register.....	393
PLL lockup	237	Port BD.....	407
PLL mode	233	Port BD mode control register	408
PM0	381	Port BD mode register	407
PM1	384	Port CD.....	404
PM2	386	Port CD function control register.....	406
PM4	388	Port CD mode control register	405
PMAH	395	Port CD mode register	404
PMAL	393	Port CM	402
PMBD	407	Port CM mode control register.....	403
PMC0.....	382	Port CM mode register	404
PMC1	385	Port configuration	367
PMC2.....	387	Port CS.....	398
PMC4.....	389	Port CS mode control register	399
PMCAH.....	395	Port CS mode register	398
PMCAL	393	Port CT	400
PMCBD.....	408	Port CT mode control register	401
PMCCD.....	405	Port CT mode register	400
PMCCM	403	Port DL	396
PMCCS.....	399	Port DL mode control register.....	397
PMCCCT	401	Port DL mode register	396
PMCD	404	Power save control	238
PMCDL	397	Power save control register	241
PMCM.....	402	Power save mode register	240
PMCS	398	PRC.....	123
PMCT.....	400	PRCMD	240
PMDL.....	396	Precautions (A/D converter)	358
Port 0	381	Precautions (DMA)	196
Port 0 function control register	383	Precautions (timer C).....	279
Port 0 mode control register.....	382	Precautions (timer D).....	288
Port 0 mode register	381	Precautions (UART)	320
Port 1	384	Prescaler unit	230
Port 1 mode control register.....	385	Priorities of maskable interrupts	207
Port 1 mode register	384	Program register set	52
Port 2	386	Program space	109
Port 2 function control register	387	Programmable wait function	94
Port 2 mode control register.....	387	PRS	230
Port 2 mode register	386	PSC.....	241
Port 4	388	PSMR.....	240
Port 4 function control register	390		
Port 4 mode control register.....	389	[Q]	
Port 4 mode register	388	Quantization error.....	363
Port AH	394		
Port AH mode control register.....	395	[R]	
Port AH mode register	395	\overline{RD}	43
		Receive buffer registers 0, 1.....	298

Receive-only serial I/O shift registers 0, 1	326	SRIC0, SRIC1	212
Recommended oscillator	415	STIC0, STIC1	212
Recommended soldering conditions	445	Stopping conversion operation	358
Recommended use of address space	65	Switching between UART and CSI modes	289
Refresh control function	147	System configuration example (CSI0, SCI1).....	332
$\overline{\text{REFRQ}}$	42	System register set	53
Relationship between programmable wait and external wait	97	System wait control register.....	72
$\overline{\text{RESET}}$	46	[T]	
Reset functions	409	TBC	253
Resolution	362	$\overline{\text{TC0}}$	39
RFS3, RFS4.....	147	Terminal count output upon DMA transfer end	191
ROMC	119	TI000	38
RXB0, RXB1	298	TI010	39
RXD0, RXD1	40	Time base counter	253
[S]		Timer C.....	254
Sampling time	365	Timer C operation	265
$\overline{\text{SCK0}}$, $\overline{\text{SCK1}}$	40	Timer D.....	280
SCR3, SCR4	131	Timer D operation	286
$\overline{\text{SDCAS}}$	44	Timer mode control registers C00, C10	260
SDCKE.....	44	Timer mode control registers C01, C11	262
SDCLK	44	Timer mode control registers D0 to D3	284
SDRAM access	133	Timer trigger interval.....	358
SDRAM configuration registers 3, 4	131	Timers C0, C1.....	256
SDRAM connection.....	128	Timers D0 to D3.....	281
SDRAM controller	128	Times related to DMA transfer	194
SDRAM initialization sequence	154	TMC0, TMC1	256
SDRAM refresh control registers 3, 4.....	147	TMCC00, TMCC10	260
$\overline{\text{SDRAS}}$	44	TMCC01, TMCC11	262
Securing oscillation stabilization time	251	TMCD0 to TMCD3	284
SEIC0, SEIC1	212	TMD0 to TMD3	281
Self-refresh control function	152	TO00.....	38
Serial I/O shift registers 0, 1	326	Transfer modes.....	174
SESC0, SESC1.....	218, 264	Transfer object.....	188
SI0, SI1	40	Transfer type and transfer object	188
Single transfer mode	174	Transmit buffer registers 0, 1	299
Single-step transfer mode	176	Two-cycle transfer	78
SIO0, SIO1.....	326	TXB0, TXB1.....	299
SIOE0, SIOE1	326	TXD0, TXD1	40
SO0, SO1	40	Types of bus states.....	172
Software exception.....	219	[U]	
Software STOP mode	248	UART0, UART1	290
SOTB0, SOTB1.....	327	$\overline{\text{UBE}}$	44
Specific registers.....	72	UDQM.....	43
SRAM connection	111	$\overline{\text{UWR}}$	42
SRAM, external ROM, external I/O access	113		
SRAM, external ROM, external I/O interface.....	110		

[V]

Valid edge selection registers C0, C1 218, 264
V_{DD} 46
V_{SS} 46
VSWC 72

[W]

$\overline{\text{WAIT}}$ 41

Wait function 94
 $\overline{\text{WE}}$ 43
Wrap-around of CPU address space 58

[X]

X1, X2 46

[Z]

Zero-scale error 363