

NEC

Preliminary User's Manual

V850E/ME2™

32-Bit Single-Chip Microcontroller

Hardware

μPD703111

Document No. U16031EJ2V1UD00 (2nd edition)
Date Published May 2003 N CP(K)

© NEC Electronics Corporation 2002
Printed in Japan

[MEMO]

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

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

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

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

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

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

- **The information contained in this document is being issued in advance of the production cycle for the product. The parameters for the product may change before final production or NEC Electronics Corporation, at its own discretion, may withdraw the product prior to its production.**
- **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 products 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: 0211-65 03 01

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318

- **Sucursal en España**
Madrid, Spain
Tel: 091-504 27 87

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-558-3737

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

NEC Electronics Shanghai, Ltd.
Shanghai, P.R. China
Tel: 021-6841-1138

- **Filiale Italiana**
Milano, Italy
Tel: 02-66 75 41

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-2719-2377

- **Branch The Netherlands**
Eindhoven, The Netherlands
Tel: 040-244 58 45

NEC Electronics Singapore Pte. Ltd.
Novena Square, Singapore
Tel: 6253-8311

- **Tyskland Filial**
Taeby, Sweden
Tel: 08-63 80 820

- **United Kingdom Branch**
Milton Keynes, UK
Tel: 01908-691-133

J03.4

Major Revisions in This Edition (1/4)

Page	Description
Throughout	<ul style="list-style-type: none"> • Addition of the following product. μPD703111GM-15-UEU • Change of the following register name. PFCAL → PFCALL
p.29	Addition of execution time of 150 MHz products to minimum instruction execution time in 1.2 Features
p.45	Addition of Note in 2.1 (2) Non-port pins
p.48	Addition of description in 2.3 (1) (b) (vii) UCLK (USB clock)
p.50	Modification of description in 2.3 (3) (b) (ii) DMAAK0, DMAAK1 (DMA acknowledge)
p.52	Modification of description in 2.3 (5) (b) (ii) DMAAK2, DMAAK3 (DMA acknowledge)
p.68	Addition of execution time of 150 MHz products to minimum instruction execution time in 3.1 Features
p.70	Modification of default value in 3.2.1 (2) Program counter (PC)
pp.86, 98	Modification of description in 3.4.7 Peripheral I/O registers
p.111	Change of table of VSWC setting values in 3.4.9 System wait control register (VSWC)
p.115	Modification of description in 4.2.1 Pin status during internal instruction RAM, internal data RAM, and peripheral I/O access
p.121	Modification of description when BTn1 and BTn0 bits are set to 11 in 4.4.1 (1) Bus cycle type configuration registers 0, 1 (BCT0, BCT1)
p.122	Modification of table of number of access clocks in 4.5.1 Number of access clocks
p.146	Addition to Cautions and modification of description in 4.5.6 (1) Line buffer control registers 0, 1 (LBC0, LBC1)
p.147	Addition of Remark in 4.5.6 (1) (a) Speculative read function (read buffer function)
pp.148, 149	Modification of description, addition of Note , and addition to Cautions in 4.5.6 (1) (b) Write buffer function
p.155	Addition to Cautions in 4.7.1 (3) Bus cycle period control register (BCP)
p.161	Modification of Cautions in 4.9.1 (1) Cache configuration register (BHC)
p.170	Addition of (5) in 4.10.3 Cautions
p.177	Addition of timing and modification of Notes in 4.11.6 (1) SDRAM (when read, latency = 2, no idle state insertion)
p.178	Addition of timing and modification of Notes in 4.11.6 (2) SDRAM (when read, latency = 2, two idle states inserted, 32-bit bus width)
p.179	Addition of timing and modification of Notes in 4.11.6 (3) SDRAM (when written)
p.182	Addition of 4.14 Timing at Which T0 State Is Not Inserted
pp.214 to 217	Addition of timing and modification of Notes in Figure 5-9 SDRAM Single Read Cycle
pp.220 to 224	Addition of timing and modification of Notes in Figure 5-10 SDRAM Single Write Cycle
pp.227, 228, 232 to 234	Addition of timing and modification of Notes in Figure 5-11 SDRAM Access Timing
p.236	Modification of Caution in 5.3.6 (1) SDRAM refresh control registers 1, 3, 4, 6 (RFS1, RFS3, RFS4, RFS6)
p.238	Modification of description in Table 5-1 Example of Interval Factor Settings
p.247	Addition of internal instruction RAM in block diagram in 6.2 Configuration
p.248	Addition to Cautions in 6.3.1 (1) DMA source address registers 0H to 3H (DSA0H to DSA3H)
p.250	Addition to Cautions in 6.3.2 (1) DMA destination address registers 0H to 3H (DDA0H to DDA3H)

Major Revisions in This Edition (2/4)

Page	Description
p.252	Addition to Cautions in 6.3.3 DMA transfer count registers 0 to 3 (DBC0 to DBC3)
p.253	Addition to Cautions in 6.3.4 DMA addressing control registers 0 to 3 (DADC0 to DADC3)
pp.256, 257	Addition to Cautions and description in 6.3.5 DMA channel control registers 0 to 3 (DCHC0 to DCHC3)
p.259	Addition to Cautions and description in 6.3.6 DMA terminal count output control register (DIOC)
p.260	Addition to Cautions in 6.3.7 DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)
pp.264, 265	Addition of 6.3.7 (1) DMA request detection function
p.266	Addition of Caution and Note and modification of description in 6.3.8 DMA interface control register (DIFC)
p.276	Addition of timing and modification of Notes in Figure 6-12 Timing of 2-Cycle DMA Transfer (SDRAM → SRAM)
p.278	Addition of 6.5.1 (1) Timing of $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals for 2-cycle transfer
p.280	Addition of 6.5.1 (2) $\overline{\text{DMAAKn}}$ signal active width extension function
p.282	Addition of 6.5.1 (3) Outline of 2-cycle transfer timing
pp.289 to 291	Modification of timing in Figure 6-19 Timing of DMA Flyby Transfer (External I/O → SDRAM)
pp.298, 299	Modification of timing in Figure 6-23 Timing of DMA Flyby Transfer (SDRAM → External I/O)
pp.300, 301	Modification of timing in Figure 6-24 Timing of DMA Flyby Transfer (External I/O → SDRAM)
p.302	Modification of description and addition to Caution in Table 6-5 Relationship Between Transfer Type and Transfer Object
p.305	Addition of description in 6.8 Next Address Setting Function
p.307	Addition of Cautions in 6.9 DMA Transfer Start Factors
p.312	Modification of description in 6.13 Times Related to DMA Transfer
p.314	Modification of description in 6.15 (3) Bus arbitration for CPU
p.315	Addition of 6.15 (7) Read values of DSAn and DDAn registers
p.363	Modification of Note in Figure 7-14 Pipeline Operation at Interrupt Request Acknowledgment (Outline)
p.365	Modification of selection of SSCG output by PLLSEL pin and MDL-Selector Table (modulation period) in 8.1 Features
pp.367, 368	Addition to Cautions and modification of description in 8.3.1 Clock control register (CKC)
p.370	Modification of sample coding <2> for data setting sequence of clock source select register (CKS) in 8.3.2 Clock source select register (CKS)
pp.371, 372	Modification of description in 8.3.3 SSCG control register (SSCGC)
p.373	Addition of Caution in 8.3.4 USB clock control register (UCKC)
p.375	Modification of oscillation stabilization time in 8.3.6 Oscillation stabilization time select register (OSTS)
p.376	Addition to Notes in Table 8-1 Operation Status of Each Clock
p.376	Modification of description in Table 8-2 Frequency List
p.377	Addition of 8.5 Operating Clock Provisions
p.400	Modification of oscillation stabilization time in Table 8-11 Counting Time Examples
p.412	Addition to Caution in 9.1.5 (2) Timer mode control registers C01 to C51 (TMCC01 to TMCC51)
p.422	Modification of Figure 9-7 TMC1 Compare Operation Example (Set/Reset Output Mode)
p.459	Addition of noise elimination width when $f_x = 150$ MHz in Table 9-6 Relationship Between NCW1n Register Set Value and Noise Elimination Width

Major Revisions in This Edition (3/4)

Page	Description
p.475	Addition of 9.3.7 (6) Overflow interrupt signal (INTOV1n) and underflow interrupt signal (INTUD1n)
p.477	Modification of transfer rate in 10.2.1 Features
p.479	Modification of description in 10.2.2 (10) UARTBn receive data register AP (UBnRXAP), UARTBn receive data register (UBnRX) (n = 0, 1)
p.480	Addition of description in 10.2.2 (12) UARTBn transmit data register n (UBnTX) (n = 0, 1)
p.487	Modification of Caution in 10.2.3 (3) UARTBn control register 2 (UBnCTL2) (n = 0, 1)
p.488	Addition of description in 10.2.3 (4) UARTBn transmit data register (UBnTX) (n = 0, 1)
p.489	Modification of description in 10.2.3 (5) UARTBn receive data register AP (UBnRXAP), UARTBn receive data register (UBnRX) (n = 0, 1)
pp.491, 492	Addition and modification of description in 10.2.3 (6) UARTBn FIFO control register 0 (UBnFIC0) (n = 0, 1)
p.493	Modification of description in 10.2.3 (7) UARTBn FIFO control register 1 (UBnFIC1) (n = 0, 1)
p.501	Addition of description in 10.2.4 (5) (b) FIFO mode
p.503	Addition of description in 10.2.5 (2) Pending mode/pointer mode
p.503	Addition of Note in 10.2.5 (2) (a) (i) During transmission (writing to transmit FIFO)
p.504	Addition of Note in 10.2.5 (2) (a) (ii) During reception (reading from receive FIFO)
p.512	Addition of description in 10.2.6 (4) (c) (ii) Reception timeout interrupt (UBTITOn) (in FIFO mode only)
p.518	Addition of value when $f_x = 150$ MHz in Table 10-4 Baud Rate Generator Setting Data
p.522	Addition of 10.2.8 Control flow
p.535	Modification of description in Figure 10-22 Block Diagram of Clocked Serial Interfaces 30 and 31
p.536	Modification of description of Caution 2 in 10.3.3 (1) Clocked serial interface mode registers 30, 31 (CSIM30, CSIM31)
p.540	Modification of description in 10.3.3 (2) Clocked serial interface clock select registers 30, 31 (CSIC30, CSIC31)
p.541	Addition of description in 10.3.3 (3) Receive data buffer registers 30, 31 (SIRB30, SIRB31)
p.542	Addition of description in 10.3.3 (4) Transmit data CSI buffer registers 30, 31 (SFDB30, SFDB31)
p.544	Modification of description in 10.3.3 (5) CSIBUF status registers 30, 31 (SFA30, SFA31)
p.549	Modification of example in Caution 2 in 10.3.4 (2) Baud rate
p.553	Modification of description in Figure 10-25 Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register)
p.555	Modification of description in Figure 10-26 Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: LSB First (DIRn Bit = 1 in CSIM3n Register)
p.559	Deletion of description of 10.3.5 (7) Slave mode
p.559	Modification of Figure 10-30 Slave Mode (CKPn and DAPn Bits = 00 in CSIC3n Register, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits))
p.563	Modification of description in Figure 10-32 Continuous Mode
p.564	Modification of description in 10.3.5 (11) Transmission mode
p.564	Addition of description in 10.3.5 (12) Reception mode
p.567	Deletion of description in 10.3.5 (16) (a) SCKn pin
p.567	Modification of description in Table 10-8 Default Output Level of SCKn Pin
pp.568 to 573	Modification of description of (1) to (12) in 10.3.6 Usage
p.575	Addition of Caution in 11.1 Overview

Major Revisions in This Edition (4/4)

Page	Description
p.578	Addition of items in Table 11-2 Correspondence Between Requests and Decoded Values
p.589	Addition of description in 11.4.1 (3) UF0 EPNAK register (UF0EN)
p.599	Deletion of description in 11.4.1 (9) UF0 EP status 1 register (UF0EPS1)
p.601	Modification of description in 11.4.1 (11) UF0 INT status 0 register (UF0IS0)
p.629	Modification of description in 11.4.1 (34) UF0 mode status register (UF0MODS)
p.665	Modification of description in 11.4.3 (2) UF0 EP0 status register L (UF0E0SL)
p.672	Modification of description in 11.4.3 (9) UF0 address register (UF0ADRS)
p.673	Modification of description in 11.4.3 (10) UF0 configuration register (UF0CNF)
p.674	Modification of description in 11.4.3 (11) UF0 interface 0 register (UF0IF0)
p.675	Modification of description in 11.4.3 (12) UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4)
p.677	Modification of Caution 2 in 11.4.3 (14) UF0 device descriptor registers 0 to 17 (UF0DD0 to UF0DD17)
p.678	Modification of Caution 2 in 11.4.3 (15) UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255)
p.680	Modification of Caution in 11.4.4 (1) USB function 0 DMA channel select register (UF0CS)
p.686	Deletion of Caution in Table 11-8 FW-Supported Standard Requests
p.692	Modification of description in Figure 11-15 Automatically Processed Requests for Control Transfer
pp.696, 697, 699, 700, 702 to 704, 706	Modification of description in Figure 11-20 CPUDEC Request for Control Transfer
p.733	Modification of description in Figure 11-30 USB Connection Example
p.739	Addition of value when $f_x = 150$ MHz in Table 12-1 Setting of A/D Conversion Operation Time
p.763	Addition of 12.9 How to Read A/D Converter Characteristics Table
p.767	Modification of repeat frequency in 13.1 Features
p.768	Modification of description in Figure 13-1 Block Diagram of PWM Unit
p.770	Modification of description in 13.3 (1) PWM control registers 0 and 1 (PWMC0 and PWMC1)
p.775	Modification of description in 13.4.2 (1) Setting for starting PWM operation
p.779	Modification of description in Table 13-1 Repeat Cycle of PWMn
p.839	Modification of Caution in 14.3.8 Port DH
p.843	Modification of Caution and description on bit 0 in 14.3.8 (2) (c) Port DH function control register (PFCDH)
p.853	Addition to Caution in 14.3.10 (2) (c) Port CT function control register (PFCCT)
p.866	Addition of noise elimination width when $f_x = 150$ MHz in Table 14-4 Relationship Between NCW1n Register Set Value and Noise Elimination Width
p.873	Modification of value of program counter (PC) after reset in Table 15-2 Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset
p.880	Modification of description in 16.1.1 (7) Mask function
p.885	Addition of CHAPTER 17 ELECTRICAL SPECIFICATIONS (TARGET VALUES)
p.928	Addition of CHAPTER 18 PACKAGE DRAWING
p.929	Addition of CHAPTER 19 RECOMMENDED SOLDERING CONDITIONS
p.952	Addition to Note in B.2 Instruction Set (in Alphabetical Order)
p.953	Addition of APPENDIX C REVISION HISTORY

The mark ★ shows major revised points.

INTRODUCTION

Readers This manual is intended for users who wish to understand the functions of the V850E/ME2 (μ PD703111) to design application systems using the V850E/ME2.

Purpose The purpose of this manual is for users to gain an understanding of the hardware functions of the V850E/ME2.

Organization The **V850E/ME2 User's Manual** is divided into two parts: Hardware (this manual) and Architecture (**V850E1 Architecture User's Manual**). The organization of each manual is as follows:

Hardware

- Pin functions
- CPU function
- Internal peripheral functions
- Electrical specifications (target value)

Architecture

- Data type
- Register set
- Instruction format and instruction set
- Interrupts and exceptions
- 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 A REGISTER INDEX**.
- To understand the details of an instruction function
→Refer to the **V850E1 Architecture User's Manual**.

To know the electrical specifications of the V850E/ME2
→ Refer to **CHAPTER 17 ELECTRICAL SPECIFICATIONS (TARGET VALUES)**.

- To understand the overall functions of the V850E/ME2
→Read this manual according to the **CONTENTS**.
- How to interpret the register format
→For a 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:	Higher addresses on the top and lower addresses on the bottom
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.

Document related to V850E/ME2

Document Name	Document No.
V850E1 Architecture User's Manual	U14559E
V850E/ME2 Hardware User's Manual	This manual

Document related to development tools (User's Manuals)

Document Name	Document No.	
CA850 (Ver. 2.50) (C Compiler Package)	Operation	U16053E
	C Language	U16054E
	PM plus	U16055E
	Assembly Language	U16042E
RX850 (Ver. 3.13 or Later) (Real-Time OS)	Basics	U13430E
	Installation	U13410E
	Technical	U13431E
RX850 Pro (Ver. 3.15) (Real-Time OS)	Basics	U13773E
	Installation	U13774E
	Technical	U13772E
RD850 (Ver. 3.01) (Task Debugger)	U13737E	
RD850 Pro (Ver. 3.01) (Task Debugger)	U13916E	

CONTENTS

CHAPTER 1 INTRODUCTION	28
1.1 Outline	28
1.2 Features	29
1.3 Applications	31
1.4 Ordering Information	31
1.5 Pin Configuration	32
1.6 Function Blocks	34
1.6.1 Internal block diagram	34
1.6.2 On-chip units	35
CHAPTER 2 PIN FUNCTIONS	38
2.1 List of Pin Functions	38
2.2 Pin Status	46
2.3 Description of Pin Functions	48
2.4 Pin I/O Circuits and Recommended Connection of Unused Pins	64
2.5 Pin I/O Circuits	67
CHAPTER 3 CPU FUNCTION	68
3.1 Features	68
3.2 CPU Register Set	69
3.2.1 Program register set.....	70
3.2.2 System register set	71
3.3 Operating Modes	74
3.3.1 Operating modes.....	74
3.3.2 Operating mode specification	74
3.4 Address Space	75
3.4.1 CPU address space	75
3.4.2 Image	76
3.4.3 Wrap-around of CPU address space	77
3.4.4 Memory map	78
3.4.5 Area.....	79
3.4.6 Recommended use of address space	83
3.4.7 Peripheral I/O registers	85
3.4.8 Specific registers.....	111
3.4.9 System wait control register (VSWC).....	111
3.4.10 Initialization sequence	112
CHAPTER 4 BUS CONTROL FUNCTION	114
4.1 Features	114
4.2 Bus Control Pins	114
4.2.1 Pin status during internal instruction RAM, internal data RAM, and peripheral I/O access	115
4.3 Memory Block Function	116

4.3.1	Chip select control function	117
4.4	Bus Cycle Type Control Function	120
4.4.1	Bus cycle type configuration registers 0, 1 (BCT0, BCT1)	121
4.5	Bus Access	122
4.5.1	Number of access clocks	122
4.5.2	Bus sizing function	123
4.5.3	Endian control function	124
4.5.4	Big endian method usage restrictions in NEC Electronics development tools	125
4.5.5	Bus width	127
4.5.6	Data read control function	146
4.6	Bus Clock Control Function	150
4.7	Wait Function	152
4.7.1	Programmable wait function	152
4.7.2	External wait function	157
4.7.3	Relationship between programmable wait and external wait	157
4.7.4	Bus cycles in which wait function is valid	158
4.8	Idle State Insertion Function	159
4.9	Instruction Cache Function	161
4.9.1	Cache configuration register (BHC)	161
4.9.2	8 KB 2-way set associative cache	163
4.9.3	LRU algorithm	163
4.9.4	Instruction cache control function	164
4.9.5	Tag clear function	166
4.9.6	Auto fill function (way 0 only)	167
4.10	Internal Instruction RAM Control Function	168
4.10.1	Internal instruction RAM mode register (IRAMM)	168
4.10.2	Operation	169
4.10.3	Cautions	170
4.11	Bus Hold Function	171
4.11.1	Function outline	171
4.11.2	Bus hold procedure	172
4.11.3	Operation in power-save mode	172
4.11.4	Bus hold timing	173
4.11.5	Bus hold timing (SRAM)	174
4.11.6	Bus hold timing (SDRAM)	177
4.12	Bus Priority Order	181
4.13	Boundary Operation Conditions	181
4.13.1	Program space	181
4.13.2	Data space	181
★ 4.14	Timing at Which T0 State Is Not Inserted	182
CHAPTER 5 MEMORY ACCESS CONTROL FUNCTION		183
5.1	SRAM, External ROM, External I/O Interface	183
5.1.1	Features	183
5.1.2	SRAM connection	184
5.1.3	SRAM, external ROM, external I/O access	186
5.2	Page ROM Controller (ROMC)	196

5.2.1	Features	196
5.2.2	Page ROM connection	197
5.2.3	On-page/off-page judgment	198
5.2.4	Page ROM configuration register (PRC)	199
5.2.5	Page ROM access	200
5.3	DRAM Controller (SDRAM).....	206
5.3.1	Features	206
5.3.2	SDRAM connection	206
5.3.3	Address multiplex function	207
5.3.4	SDRAM configuration registers 1, 3, 4, 6 (SCR1, SCR3, SCR4, SCR6).....	209
5.3.5	SDRAM access	212
5.3.6	Refresh control function	236
5.3.7	Self-refresh control function	241
5.3.8	SDRAM initialization sequence	243

CHAPTER 6 DMA FUNCTIONS (DMA CONTROLLER).....246

6.1	Features.....	246
6.2	Configuration.....	247
6.3	Control Registers	248
6.3.1	DMA source address registers 0 to 3 (DSA0 to DSA3)	248
6.3.2	DMA destination address registers 0 to 3 (DDA0 to DDA3)	250
6.3.3	DMA transfer count registers 0 to 3 (DBC0 to DBC3)	252
6.3.4	DMA addressing control registers 0 to 3 (DADC0 to DADC3).....	253
6.3.5	DMA channel control registers 0 to 3 (DCHC0 to DCHC3)	256
6.3.6	DMA terminal count output control register (DTCO).....	259
6.3.7	DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)	260
6.3.8	DMA interface control register (DIFC)	266
6.4	Transfer Modes.....	267
6.4.1	Single transfer mode	267
6.4.2	Single-step transfer mode	269
6.4.3	Block transfer mode	270
6.5	Transfer Types.....	271
6.5.1	2-cycle transfer.....	271
6.5.2	Flyby transfer	288
6.6	Transfer Object.....	302
6.6.1	Transfer type and transfer object	302
6.6.2	External bus cycles during DMA transfer	304
6.7	DMA Channel Priorities	304
6.8	Next Address Setting Function.....	305
6.9	DMA Transfer Start Factors	307
6.10	Terminal Count Output upon DMA Transfer End.....	309
6.11	Forcible Interruption	310
6.12	Forcible Termination.....	311
6.13	Times Related to DMA Transfer	312
6.14	Maximum Response Time for DMA Transfer Request	312
6.15	Cautions	314
6.15.1	Interrupt factors	315

6.16	DMA Transfer End.....	315
CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION.....316		
7.1	Features	316
7.2	Non-Maskable Interrupts.....	320
7.2.1	Operation	321
7.2.2	Restore	323
7.2.3	Non-maskable interrupt status flag (NP)	324
7.2.4	Edge detection function	324
7.3	Maskable Interrupts	325
7.3.1	Operation	325
7.3.2	Restore	327
7.3.3	Priorities of maskable interrupts.....	328
7.3.4	Interrupt control register (xxICn)	332
7.3.5	Interrupt mask registers 0 to 5 (IMR0 to IMR5).....	335
7.3.6	NMI reset status register (NRS).....	337
7.3.7	In-service priority register (ISPR).....	338
7.3.8	Maskable interrupt status flag (ID)	338
7.3.9	Selecting interrupt trigger mode.....	339
7.4	Software Exception.....	354
7.4.1	Operation	354
7.4.2	Restore	355
7.4.3	Exception status flag (EP).....	356
7.5	Exception Trap	357
7.5.1	Illegal opcode definition	357
7.5.2	Debug trap	359
7.6	Multiple Interrupt Servicing Control.....	361
7.7	Interrupt Latency Time	363
7.8	Periods in Which Interrupts Are Not Acknowledged	364
CHAPTER 8 CLOCK GENERATION FUNCTION		
8.1	Features	365
8.2	Configuration.....	366
8.3	Control Registers	366
8.3.1	Clock control register (CKC)	366
8.3.2	Clock source select register (CKS)	369
8.3.3	SSCG control register (SSCGC).....	371
8.3.4	USB clock control register (UCKC)	373
8.3.5	Lock register (LOCKR).....	374
8.3.6	Oscillation stabilization time select register (OSTS).....	375
8.4	Operation	376
8.4.1	Operation status of each clock.....	376
8.4.2	Setting of input clock (Fx).....	376
★ 8.5	Operating Clock Provisions.....	377
8.5.1	Calculating BUSCLK frequency	378
8.5.2	Calculating operating clock frequency of each on-chip peripheral function.....	378
8.6	Power-Save Control.....	381

8.6.1	Overview	381
8.6.2	Control registers	383
8.6.3	HALT mode	386
8.6.4	IDLE mode	390
8.6.5	Software STOP mode	394
8.7	Securing Oscillation Stabilization Time.....	400
8.7.1	Oscillation stabilization time security specification	400
8.7.2	Time base counter (TBC)	400
CHAPTER 9 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT).....		401
9.1	Timer C	401
9.1.1	Features	401
9.1.2	Function overview	401
9.1.3	Basic configuration	402
9.1.4	Timer C	404
9.1.5	Control registers	408
9.1.6	Operation	416
9.1.7	Application examples	423
9.1.8	Cautions	430
9.2	Timer D	431
9.2.1	Features	431
9.2.2	Function overview	431
9.2.3	Basic configuration	431
9.2.4	Timer D	432
9.2.5	Control registers	435
9.2.6	Operation	437
9.2.7	Application examples	439
9.2.8	Cautions	439
9.3	Timer ENC1	440
9.3.1	Features	440
9.3.2	Function overview	440
9.3.3	Basic configuration	442
9.3.4	Timer ENC1	444
9.3.5	Control registers	449
9.3.6	Operation	460
9.3.7	Supplementary description of internal operation	472
CHAPTER 10 SERIAL INTERFACE FUNCTION.....		476
10.1	Features.....	476
10.1.1	Switching between UARTB0 and CSI30 modes.....	476
10.2	Asynchronous Serial Interfaces B0, B1 (UARTB0, UARTB1)	477
10.2.1	Features	477
10.2.2	Configuration	478
10.2.3	Control registers	482
10.2.4	Interrupt requests	499
10.2.5	Control method.....	502
10.2.6	Operation	505

★

10.2.7	Dedicated baud rate generators 0, 1 (BRG0, BRG1).....	516
10.2.8	Control flow	522
10.2.9	Cautions.....	532
10.3	Clocked Serial Interfaces 30, 31 (CSI30, CSI31).....	533
10.3.1	Features.....	533
10.3.2	Configuration.....	534
10.3.3	Control registers.....	536
10.3.4	Dedicated baud rate generators 0, 1 (BRG0, BRG1).....	548
10.3.5	Operation	550
10.3.6	Usage	568
10.3.7	Cautions.....	574
CHAPTER 11 USB FUNCTION CONTROLLER (USBF)		575
11.1	Overview	575
11.2	Configuration.....	576
11.3	Requests	577
11.3.1	Automatic requests	577
11.3.2	Other requests	585
11.4	Register Configuration	586
11.4.1	Control registers.....	586
11.4.2	Data hold registers.....	639
11.4.3	Request data register area	664
11.4.4	Peripheral control registers	680
11.5	STALL Handshake or No Handshake	682
11.6	Register Values in Specific Status	683
11.7	FW Processing	685
11.7.1	Initialization processing.....	687
11.7.2	Interrupt servicing	690
11.7.3	USB main processing	691
11.7.4	Suspend/Resume processing	718
11.7.5	Processing after power application	721
11.7.6	Receiving data for bulk transfer (OUT) in DMA mode.....	724
11.7.7	Transmitting data for bulk transfer (IN) in DMA mode.....	728
11.7.8	USB connection example.....	733
CHAPTER 12 A/D CONVERTER		734
12.1	Features	734
12.2	Configuration.....	735
12.3	Control Registers	737
12.4	Operation	744
12.4.1	Basic operation	744
12.4.2	Operation mode and trigger mode	745
12.5	Operation in A/D Trigger Mode.....	750
12.5.1	Select mode operation	750
12.5.2	Scan mode operations	752
12.6	Operation in Timer Trigger Mode	753
12.6.1	Select mode operation	754

12.6.2	Scan mode operation	756
12.7	Operation in External Trigger Mode	757
12.7.1	Select mode operations.....	757
12.7.2	Scan mode operation	760
12.8	Notes on Operation	761
★ 12.9	How to Read A/D Converter Characteristics Table.....	763
 CHAPTER 13 PWM UNIT		767
13.1	Features.....	767
13.2	Configuration.....	767
13.3	Control Registers	769
13.4	Operation	772
13.4.1	PWM basic operation	772
13.4.2	Starting/stopping PWM operation.....	775
13.4.3	Setting active level of PWM pulse	777
13.4.4	Specifying PWM pulse width rewrite period	778
13.4.5	Repeat cycle	779
 CHAPTER 14 PORT FUNCTIONS		780
14.1	Features.....	780
14.2	Port Configuration.....	780
14.3	Port Pin Functions	809
14.3.1	Port 1.....	809
14.3.2	Port 2.....	814
14.3.3	Port 5.....	819
14.3.4	Port 6.....	824
14.3.5	Port 7.....	829
14.3.6	Port AL	832
14.3.7	Port AH.....	837
14.3.8	Port DH	839
14.3.9	Port CS.....	848
14.3.10	Port CT.....	851
14.3.11	Port CM.....	854
14.3.12	Port CD	857
14.4	Configuration of $\overline{\text{RESET}}$, A2 to A15, and D0 to D15 Pins.....	859
14.5	Operation of Port Function	861
14.5.1	Writing to I/O port	861
14.5.2	Reading from I/O port.....	861
14.5.3	Output status of alternate function in control mode	861
14.6	Noise Eliminator	862
14.6.1	Interrupt input pin	862
14.6.2	A/D converter input pin.....	863
14.6.3	Timer C and timer ENC1 input pins.....	863
 CHAPTER 15 RESET FUNCTIONS		867
15.1	Overview	867

15.2	Configuration.....	867
15.3	Operation	868
15.4	Initialization	873
CHAPTER 16 DEBUG FUNCTION (DCU).....		879
16.1	Functional Outline	879
16.1.1	Debug function.....	879
16.1.2	Trace function	880
16.1.3	Event function	881
16.2	Connection with N-Wire Type IE.....	882
16.2.1	IE connector (on target system side)	883
16.2.2	Recommended circuit example.....	884
★	CHAPTER 17 ELECTRICAL SPECIFICATIONS (TARGET VALUES).....	885
★	CHAPTER 18 PACKAGE DRAWING	928
★	CHAPTER 19 RECOMMENDED SOLDERING CONDITIONS	929
	APPENDIX A REGISTER INDEX.....	930
	APPENDIX B INSTRUCTION SET LIST	943
	B.1 Conventions	943
	B.2 Instruction Set (in Alphabetical Order).....	946
★	APPENDIX C REVISION HISTORY	953

LIST OF FIGURES (1/6)

Figure No.	Title	Page
3-1	CPU Register Set.....	69
3-2	CPU Address Space	75
3-3	Images on Address Space.....	76
3-4	Memory Map	78
3-5	Recommended Memory Map.....	84
4-1	Example When CSC0 Register Is Set to 0803H and CSC1 Register Is Set to 0601H.....	119
4-2	Big Endian Addresses Within Word	125
4-3	Little Endian Addresses Within Word.....	125
4-4	Outline of Speculative Read Operation.....	148
4-5	BMC Register Switching Timing	151
4-6	Example of Wait Insertion.....	157
4-7	Configuration of 8 KB 2-Way Set Associative Instruction Cache	163
4-8	Memory Map of Internal Instruction RAM.....	168
5-1	Examples of Connection to SRAM.....	184
5-2	SRAM, External ROM, External I/O Access Timing.....	186
5-3	Examples of Connection to Page ROM	197
5-4	Example of Control by MA6 to MA3 Bits of PRC Register	198
5-5	Page ROM Access Timing.....	200
5-6	Example of Connection to SDRAM.....	206
5-7	Row Address/Column Address Output	207
5-8	State Transition of SDRAM Access	212
5-9	SDRAM Single Read Cycle	214
5-10	SDRAM Single Write Cycle.....	220
5-11	SDRAM Access Timing.....	227
5-12	Auto-Refresh Cycle.....	239
5-13	CBR Refresh Timing (SDRAM).....	240
5-14	Self-Refresh Timing (SDRAM).....	242
5-15	SDRAM Mode Register Setting Cycle.....	244
5-16	SDRAM Register Write Operation Timing.....	245
6-1	Single Transfer Example 1.....	267
6-2	Single Transfer Example 2.....	267
6-3	Single Transfer Example 3.....	268
6-4	Single Transfer Example 4.....	268
6-5	Single-Step Transfer Example 1	269
6-6	Single-Step Transfer Example 2	269
6-7	Block Transfer Example.....	270
6-8	Timing of 2-Cycle DMA Transfer (SRAM → External I/O).....	271
6-9	Timing of 2-Cycle DMA Transfer (SDRAM → SRAM): Single Transfer Mode (SRAM Data 1 Wait, SDRAM Latency = 2, BMC Register = 00H, Level Detection Mode).....	273
6-10	Timing of 2-Cycle DMA Transfer (SRAM → SDRAM): Single Transfer Mode (SRAM Data 1 Wait, BMC Register = 00H, Edge Detection Mode)	274

LIST OF FIGURES (2/6)

Figure No.	Title	Page
6-11	Timing of 2-Cycle DMA Transfer (SRAM → External I/O): Without Speculative Read	275
6-12	Timing of 2-Cycle DMA Transfer (SDRAM → SRAM)	276
6-13	Bus Configuration	278
6-14	Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 1 (0 SRAM Waits)	282
6-15	Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 2 (0 SRAM Waits)	283
6-16	Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 3 (0 SRAM Waits)	284
6-17	Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 4 (0 SRAM Waits)	286
6-18	Circuit Example When Flyby Transfer Is Performed Between External I/O and SRAM	288
6-19	Timing of DMA Flyby Transfer (External I/O → SDRAM)	289
6-20	Timing of DMA Flyby Transfer (SRAM → External I/O)	292
6-21	Timing of DMA Flyby Transfer (External I/O → SRAM)	294
6-22	Timing of DMA Flyby Transfer (Page ROM → External I/O)	296
6-23	Timing of DMA Flyby Transfer (SDRAM → External I/O)	298
6-24	Timing of DMA Flyby Transfer (External I/O → SDRAM)	300
6-25	Buffer Register Configuration	305
6-26	Terminal Count Signal (\overline{TCn}) Timing Example (1)	309
6-27	Terminal Count Signal (\overline{TCn}) Timing Example (2)	309
6-28	Example of Forcible Termination of DMA Transfer	311
7-1	Servicing Configuration of Non-Maskable Interrupt	321
7-2	Acknowledging Non-Maskable Interrupt Request	322
7-3	RETI Instruction Processing	323
7-4	Maskable Interrupt Servicing	326
7-5	RETI Instruction Processing	327
7-6	Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Serviced	329
7-7	Example of Servicing Interrupt Requests Simultaneously Generated	331
7-8	Software Exception Processing	354
7-9	RETI Instruction Processing	355
7-10	Exception Trap Processing	358
7-11	Restore Processing from Exception Trap	358
7-12	Debug Trap Processing	359
7-13	Restore Processing from Debug Trap	360
7-14	Pipeline Operation at Interrupt Request Acknowledgment (Outline)	363
8-1	Power-Save Mode State Transition Diagram	382
8-2	BUSCLK Operation When HALT Mode Is Released	389
8-3	BUSCLK Operation When IDLE Mode Is Released	393
8-4	BUSCLK Operation When Software STOP Mode Is Released	398
9-1	Timer C Block Diagram	403
9-2	Basic Operation of Timer C	416
9-3	Operation After Overflow (When OSTCn = 1)	417
9-4	Capture Operation Example	418

LIST OF FIGURES (3/6)

Figure No.	Title	Page
9-5	TMC1 Capture Operation Example (When Both Edges Are Specified)	419
9-6	Compare Operation Example	420
9-7	TMC1 Compare Operation Example (Set/Reset Output Mode)	422
9-8	Contents of Register Settings When Timer C Is Used as Interval Timer.....	423
9-9	Interval Timer Operation Timing Example.....	424
9-10	Contents of Register Settings When Timer C Is Used for PWM Output.....	425
9-11	PWM Output Operation Timing Example	426
9-12	Contents of Register Settings When Timer C Is Used for Cycle Measurement	428
9-13	Cycle Measurement Operation Timing Example.....	429
9-14	Timer D Block Diagram.....	431
9-15	Example of Timing During TMDn Operation	434
9-16	TMD0 Compare Operation Example.....	437
9-17	Timer ENC1 Block Diagram.....	443
9-18	TMENC1n Block Diagram (During PWM Output Operation).....	462
9-19	PWM Signal Output Example (When ALVT1n0 Bit = 0 Is Set).....	463
9-20	Mode 1 (When Rising Edge Is Specified as Valid Edge of TIUD1n Pin).....	465
9-21	Mode 1 (When Rising Edge Is Specified as Valid Edge of TIUD1n Pin): In Case of Simultaneous TIUD1n, TCUD1n Pin Edge Timing	465
9-22	Mode 2 (When Rising Edge Is Specified as Valid Edge of TIUD1n, TCUD1n Pins)	466
9-23	Mode 3 (When Rising Edge Is Specified as Valid Edge of TIUD1n pin)	467
9-24	Mode 3 (When Rising Edge Is Specified as Valid Edge of TIUD1n Pin): In Case of Simultaneous TIUD1n, TCUD1n Pin Edge Timing	467
9-25	Mode 4.....	468
9-26	Example of TMENC1n Operation When Interval Operation and Transfer Operation Are Combined	469
9-27	Example of TMENC1n Operation in UDC Mode	471
9-28	Clear Operation upon Match with CM1n0 During TMENC1n Up Count Operation	472
9-29	Clear Operation upon Match with CM1n1 During TMENC1n Down Count Operation.....	472
9-30	Count Value Clear Operation upon Compare Match.....	473
9-31	Internal Operation During Transfer Operation.....	473
9-32	Interrupt Output upon Compare Match (CM1n1 with Operation Mode Set to General-Purpose Timer Mode and Count Clock Set to $f_x/8$).....	474
9-33	UBD1n Flag Operation.....	474
10-1	Block Diagram of Asynchronous Serial Interfaces B0 and B1	481
10-2	Asynchronous Serial Interface Transmit/Receive Data Format (LSB-First Transfer).....	505
10-3	Timing of Asynchronous Serial Interface Transmission Completion Interrupt (UBTITn)	508
10-4	Timing of Asynchronous Serial Interface FIFO Transmission Completion Interrupt (UBTIFn)	508
10-5	Timing of Asynchronous Serial Interface Reception Completion Interrupt (UBTIRn).....	512
10-6	Noise Filter Circuit.....	515
10-7	Timing of RXDn Signal Judged as Noise	515
10-8	Baud Rate Generator Configuration.....	516
10-9	Allowable Baud Rate Range During Reception	519
10-10	Transfer Rate During Continuous Transmission	521
10-11	Example of Continuous Transmission Processing Flow in Single Mode (CPU Control)	522

LIST OF FIGURES (4/6)

Figure No.	Title	Page
10-12	Example of Continuous Reception Processing Flow in Single Mode (CPU Control)	523
10-13	Example of Continuous Transmission Processing Flow in Single Mode (DMA Control).....	524
10-14	Example of Continuous Reception Processing Flow in Single Mode (DMA Control).....	525
10-15	Example of Continuous Transmission Processing Flow in FIFO Mode (CPU Control).....	526
10-16	Example of Continuous Reception Processing in FIFO Mode (CPU Control).....	527
10-17	Example of Continuous Transmission (Pending Mode) Processing in FIFO Mode (DMA Control).....	528
10-18	Example of Continuous Reception (Pending Mode) Processing Flow in FIFO Mode (DMA Control)	529
10-19	Example of Reception Error Processing in Single Mode	530
10-20	Example of Reception Error Processing Flow in FIFO Mode (1)	531
10-21	Example of Reception Error Processing Flow in FIFO Mode (2)	532
10-22	Block Diagram of Clocked Serial Interfaces 30 and 31	535
10-23	Transfer Clock of CSI3n	548
10-24	Function of CSI Data Buffer Register n (CSIBUFn)	551
10-25	Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register).....	552
10-26	Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: LSB First (DIRn Bit = 1 in CSIM3n Register).....	554
10-27	Transfer Data Length: 16 Bits (CCLn3 to CCLn0 Bits = 0000 in CSIL3n Register), Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register).....	556
10-28	Clock Timing	557
10-29	Master Mode (CKPn and DAPn Bits = 00 in CSIC3n Register, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits))	558
10-30	Slave Mode (CKPn and DAPn Bits = 00 in CSIC3n Register, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits))	559
10-31	Single Mode.....	561
10-32	Continuous Mode	563
10-33	Delay Control of Transmission/Reception Completion Interrupt (INTCSI3n): CSITn Bit = 1 in CSIC3n Register, CSWEn Bit = 0, CKPn and DAPn Bits = 00, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits)	565
10-34	Enabling/Disabling Transfer Wait	566
11-1	Operation of UF0E0R Register.....	640
11-2	Operation of UF0E0ST Register.....	642
11-3	Operation of UF0E0W Register	644
11-4	Operation of UF0BO1 Register.....	646
11-5	Operation of UF0BO2 Register.....	650
11-6	Operation of UF0B11 Register	653
11-7	Operation of UF0B12 Register	657
11-8	Operation of UF0INT1 Register	661
11-9	Operation of UF0INT2 Register	663
11-10	Initializing Request Data Register Area	687
11-11	Initialization of Request Data Register Area	688
11-12	Setting of Interface and Endpoint	688
11-13	Setting of Interrupt.....	689

LIST OF FIGURES (5/6)

Figure No.	Title	Page
11-14	Interrupt Servicing.....	690
11-15	Automatically Processed Requests for Control Transfer.....	692
11-16	CLEAR_FEATURE Processing	693
11-17	SET_FEATURE Processing	694
11-18	SET_CONFIGURATION Processing	695
11-19	SET_INTERFACE Processing.....	695
11-20	CPUDEC Request for Control Transfer	696
11-21	Processing for Bulk Transfer (IN) (Endpoint1)	709
11-22	Parallel Processing by Hardware.....	710
11-23	Normal Processing for Bulk Transfer (OUT) (Endpoint2).....	712
11-24	Processing If More Data Than Expected by System Is Transmitted (Endpoint2).....	714
11-25	Processing for Interrupt Transfer (IN) (Endpoint7).....	717
11-26	Example of Suspend/Resume Processing.....	718
11-27	Example of Processing After Power Application/Power Failure	721
11-28	DMA Processing by Bulk Transfer (OUT)	725
11-29	DMA Processing by Bulk Transfer (IN)	729
11-30	USB Connection Example	733
12-1	Block Diagram of A/D Converter	736
12-2	Relationship Between Analog Input Voltage and A/D Conversion Results	743
12-3	Select Mode Operation Timing: 1-Buffer Mode (ANI1).....	747
12-4	Select Mode Operation Timing: 4-Buffer Mode (ANI2).....	748
12-5	Scan Mode Operation Timing: 4-Channel Scan (ANI0 to ANI3)	749
12-6	Example of 1-Buffer Mode Operation (A/D Trigger Select: 1 Buffer).....	750
12-7	Example of 4-Buffer Mode Operation (A/D Trigger Select: 4 Buffers).....	751
12-8	Example of Scan Mode Operation (A/D Trigger Scan)	752
12-9	Example of 1-Buffer Mode Operation (Timer Trigger Select: 1 Buffer) (ANI1)	754
12-10	Example of 4-Buffer Mode Operation (Timer Trigger Select: 4 Buffers) (ANI3)	755
12-11	Example of Scan Mode Operation (Timer Trigger Scan) (ANI0 to ANI4)	756
12-12	Example of 1-Buffer Mode Operation (External Trigger Select: 1 Buffer) (ANI1)	758
12-13	Example of 4-Buffer Mode Operation (External Trigger Select: 4 Buffers) (ANI2)	759
12-14	Example of Scan Mode Operation (External Trigger Scan) (ANI0 to ANI3).....	760
12-15	Overall Error	763
12-16	Quantization Error.....	764
12-17	Zero-Scale Error	764
12-18	Full-Scale Error.....	765
12-19	Differential Linearity Error	765
12-20	Integral Linearity Error	766
12-21	Sampling Time	766
13-1	Block Diagram of PWM Unit	768
13-2	Example of PWM Output with Main Pulse and Ancillary Pulse	773
13-3	Example of PWM Output Operation.....	774
13-4	PWM Operation Timing.....	776

LIST OF FIGURES (6/6)

Figure No.	Title	Page
13-5	Setting Active Level of PWM Output.....	777
13-6	PWM Output Timing Example 1 (PWM Pulse Width Rewrite Period: $2^{x+8}/f_{PWMC}$)	778
13-7	PWM Output Timing Example 2 (PWM Pulse Width Rewrite Period: $2^x/f_{PWMC}$)	779
14-1	Block Diagram of Type A-1.....	785
14-2	Block Diagram of Type C-1.....	786
14-3	Block Diagram of Type D-1.....	787
14-4	Block Diagram of Type D-2.....	788
14-5	Block Diagram of Type F-1.....	789
14-6	Block Diagram of Type F-2.....	790
14-7	Block Diagram of Type F-3.....	791
14-8	Block Diagram of Type F-4.....	792
14-9	Block Diagram of Type F-5.....	793
14-10	Block Diagram of Type G-1	794
14-11	Block Diagram of Type G-2	795
14-12	Block Diagram of Type G-3	796
14-13	Block Diagram of Type H-1.....	797
14-14	Block Diagram of Type J-1	798
14-15	Block Diagram of Type J-2	799
14-16	Block Diagram of Type J-3	800
14-17	Block Diagram of Type L-1	801
14-18	Block Diagram of Type L-2	802
14-19	Block Diagram of Type L-3	803
14-20	Block Diagram of Type L-4	804
14-21	Block Diagram of Type L-5	805
14-22	Block Diagram of Type M-1	806
14-23	Block Diagram of Type M-2	807
14-24	Block Diagram of Type M-3	808
14-25	Block Diagram of Type N-1.....	859
14-26	Block Diagram of Type N-2.....	859
14-27	Block Diagram of Type N-3.....	860
15-1	Reset Operation with $\overline{\text{RESET}}$ Pin Input	869
15-2	Power-on Reset Operation	871
15-3	BUSCLK Operation at Power-on Reset.....	872
16-1	Connecting N-Wire Type IE	882
16-2	Example of Recommended IE Connection Circuit.....	884

LIST OF TABLES (1/2)

Table No.	Title	Page
3-1	Program Registers	70
3-2	System Register Numbers	71
3-3	Interrupt/Exception Table	79
4-1	Bus Cycles in Which Wait Function Is Valid	158
4-2	Bus Priority Order	181
5-1	Example of Interval Factor Settings	238
6-1	Targets of 2-Cycle Transfer	278
6-2	Active Width of $\overline{\text{DMAAKn}}$ Signal for 2-Cycle Transfer	279
6-3	Correlation Between External Access (Execution to External I/O/External Memory) and Active Width of $\overline{\text{DMAAKn}}$ Signal	279
6-4	Minimum Value of Active Width of $\overline{\text{DMAAKn}}$ Signal During 2-Cycle Transfer	281
6-5	Relationship Between Transfer Type and Transfer Object	302
6-6	External Bus Cycles During DMA Transfer	304
6-7	Number of Minimum Execution Clocks in DMA Cycle	312
7-1	Interrupt/Exception Source List	317
7-2	Addresses and Bits of Interrupt Control Registers	333
8-1	Operation Status of Each Clock	376
8-2	Frequency List	376
8-3	Frequency Fluctuation of BUSCLK	378
8-4	Operation Status in HALT Mode	387
8-5	Operation After HALT Mode Is Released by Interrupt Request	388
8-6	Operation Status in IDLE Mode	391
8-7	Operation After IDLE Mode Is Released by Interrupt Request	392
8-8	Operation Status in Software STOP Mode	395
8-9	Operation After Software STOP Mode Is Released by Interrupt Request	396
8-10	Operation If Software STOP Mode Is Set in Interrupt Servicing Routine	397
8-11	Counting Time Examples	400
9-1	Timer C Configuration	402
9-2	TOCn Output Control	422
9-3	Timer D Configuration	431
9-4	Timer ENC1 Configuration	442
9-5	Timer ENC1 (TMENC1n) Clear Conditions	445
9-6	Relationship Between NCW1n Register Set Value and Noise Elimination Width	459
9-7	Capture Trigger Signal (TMENC1n) to 16-Bit Capture Register	462
9-8	List of Count Operations in UDC Mode	464
10-1	Division Value of 16-Bit Counter	487
10-2	Generated Interrupts and Default Priorities	499

LIST OF TABLES (2/2)

Table No.	Title	Page
10-3	Reception Error Causes	513
10-4	Baud Rate Generator Setting Data	518
10-5	Maximum and Minimum Allowable Baud Rate Error	520
10-6	Operation Modes	550
10-7	Conditions Under Which Data Can Be Transferred in Slave Mode	559
10-8	Default Output Level of $\overline{\text{SCKn}}$ Pin	567
10-9	Default Output Level of SOn Pin.....	567
11-1	Request Format.....	577
11-2	Correspondence Between Requests and Decoded Values.....	578
11-3	Response and Processing of Other Requests.....	585
11-4	Mapping and Data of UF0 Device Descriptor Registers	677
11-5	Mapping of UF0CIEn Register.....	678
11-6	Data of UF0CIEn Register.....	679
11-7	Register Values in Specific Status.....	683
11-8	FW-Supported Standard Requests.....	686
12-1	Setting of A/D Conversion Operation Time.....	739
12-2	Relationship Between Operation Mode and Trigger Mode	745
12-3	Correspondence Between Analog Input Pins and ADCRn Register (1-Buffer Mode (Timer Trigger Select: 1 Buffer))	754
12-4	Correspondence Between Analog Input Pins and ADCRn Register (4-Buffer Mode (Timer Trigger Select: 4 Buffers))	755
12-5	Correspondence Between Analog Input Pins and ADCRn Register (Scan Mode (Timer Trigger Scan))	756
13-1	Repeat Cycle of PWMn	779
14-1	Noise Elimination Time of Interrupt Input Pins.....	862
14-2	Noise Elimination Time of A/D Converter Input Pin	863
14-3	Noise Elimination Time of Timer C and Timer ENC1 Input Pins.....	863
14-4	Relationship Between NCW1n Register Set Value and Noise Elimination Width.....	866
15-1	Hardware Status at Reset.....	868
15-2	Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset	873
16-1	IE Connector Pin Function (on Target System Side)	883
19-1	Surface Mounting Type Soldering Conditions.....	929

CHAPTER 1 INTRODUCTION

The V850E/ME2 is a product of the NEC Electronics single-chip microcontroller “V850 Series™”. This chapter gives a simple outline of the V850E/ME2.

1.1 Outline

The V850E/ME2 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 cache, data RAM, instruction RAM, and various peripheral functions such as memory controllers, a DMA controller, real-time pulse unit, serial interfaces, USB function controller (USBF), 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/ME2 features various on-chip external memory interfaces including separately configured address (26 bits) and data (32 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 data RAM and instruction RAM, motor control, communications control and other real-time control tasks can be realized simultaneously.

(3) Internal instruction RAM

The instruction RAM can be accessed at high speed, in one clock, so that application programs can be executed in real time.

(4) A full range of middleware and development environment products

The V850E/ME2 can execute middleware such as JPEG, JBIG, MH/MR/MMR, and TCP/IP at high speed. Also, middleware that enables speech 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, and other elements is also available.

1.2 Features

- Number of instructions: 83
- ★ ○ Minimum instruction execution time: 10 ns/7.5 ns/6.7 ns (at internal 100 MHz/133 MHz/150 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:
 - 256 MB linear address space (common program/data use)
 - Chip select output function: 8 spaces
 - Memory block division function: 2, 4, 6, 8, 64 MB/block
 - Programmable wait function
 - Idle state insertion function
- External bus interface:
 - 32-bit data bus (address/data separated)
 - 32-/16-/8-bit bus sizing function
 - External bus division function: 1/1, 1/2, 1/3, 1/4 (66 MHz MAX.)
 - Bus hold function
 - External wait function
 - Address setup wait function
 - Endian control function
- Internal memory
 - Instruction RAM: 128 KB
 - Data RAM: 16 KB
- Instruction cache
 - 8 KB 2-way set associative
- Interrupts/exceptions:
 - External interrupts: 40 (including NMI)
 - Internal interrupts: 59 sources
 - Exceptions: 2 sources
 - Eight levels of priorities can be set.
- Memory access controller
 - DRAM controller (compatible with SDRAM)
 - Page ROM controller
 - Speculative read/write buffer function

- DMA controller:
 - 4 channels
 - Transfer unit: 8 bits/16 bits/32 bits
 - Maximum transfer count: 65,536 (2^{16})
 - Transfer type: Flyby (1-cycle)/2-cycle
 - Transfer mode: Single/Single step/Block
 - Transfer target: Memory \leftrightarrow memory, memory \leftrightarrow 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: 1
 - I/O ports: 77

- Real-time pulse unit:
 - 16-bit timer/event counter: 6 channels (no capture operation for 2 channels)
 - 16-bit timers: 6
 - 16-bit capture/compare registers: 12
 - 16-bit interval timer: 4 channels
 - 16-bit up/down counter/timer for 2-phase encoder input: 2 channels
 - 16-bit capture/compare registers: 4
 - 16-bit compare registers: 4

- Serial interfaces (SIO):
 - Asynchronous serial interface B (UARTB)
 - Clocked serial interface 3 (CSI3)
 - CSI3/UARTB: 1 channel
 - UARTB: 1 channel
 - CSI3: 1 channel
 - USB function controller (USBF): 1 channel
 - Full speed (12 Mbps)
 - Endpoint Control transfer: 64 bytes \times 2
 - Interrupt transfer: 8 bytes \times 2
 - Bulk transfer (IN): 64 bytes \times 2 banks \times 2
 - Bulk transfer (OUT): 64 bytes \times 2 banks \times 2

- A/D converter:
 - 10-bit resolution A/D converter: 8 channels

- PWM (Pulse Width Modulation):
 - 16-bit resolution PWM: 2 channels

- Clock generator:
 - $\times 8$ function using SSCG

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

- Package:
 - 176-pin plastic LQFP (fine pitch) (24 \times 24)

- CMOS technology:
 - All static circuits

1.3 Applications

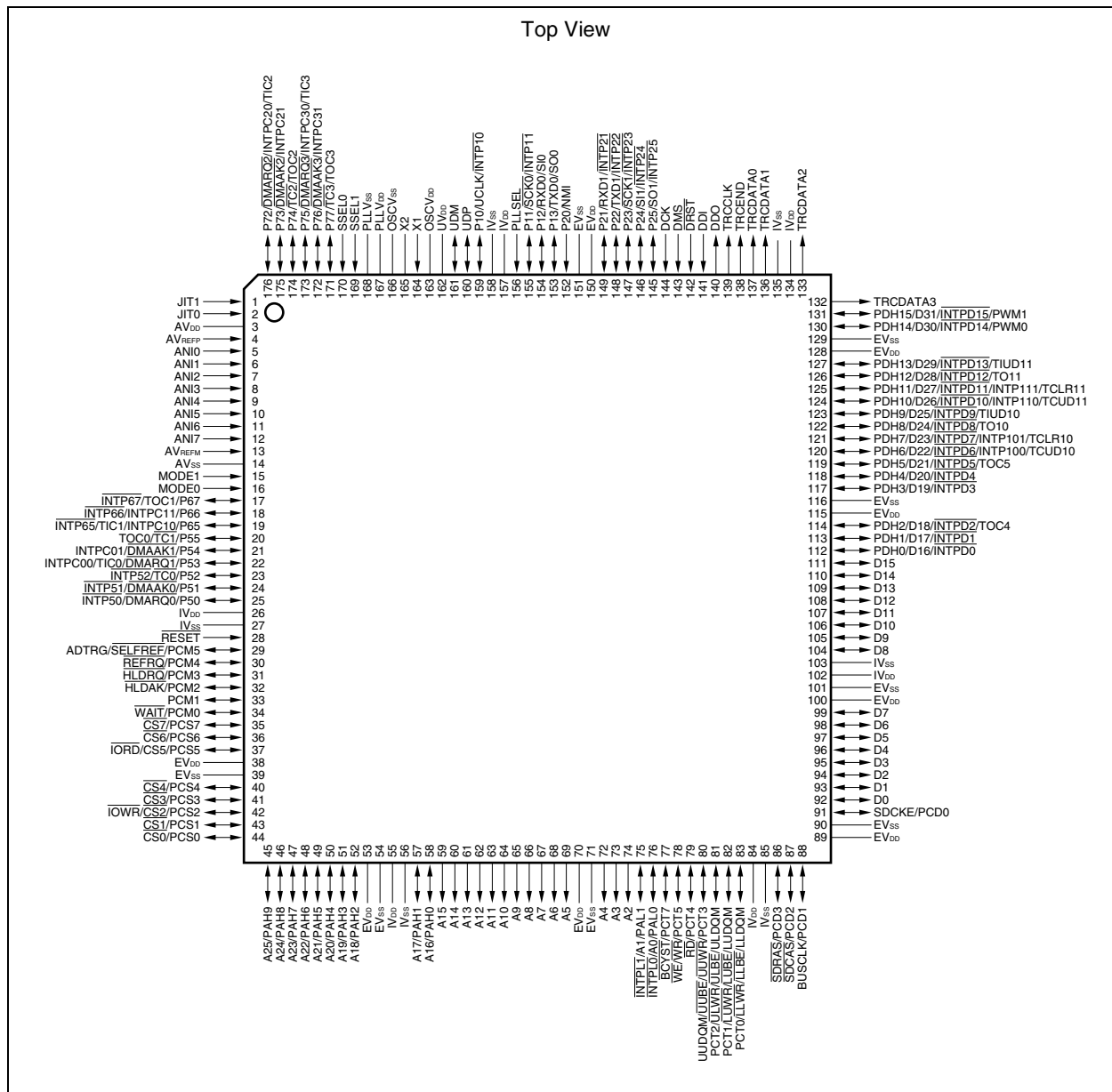
Servo control, NC machine tools, ink-jet printers, facsimiles, DVD players, video printers, PPC, information equipment, etc.

★ 1.4 Ordering Information

Part Number	Package	Maximum Operating Frequency
μ PD703111GM-10-UEU	176-pin plastic LQFP (fine pitch) (24 × 24)	100 MHz
μ PD703111GM-13-UEU	176-pin plastic LQFP (fine pitch) (24 × 24)	133 MHz
μ PD703111GM-15-UEU	176-pin plastic LQFP (fine pitch) (24 × 24)	150 MHz

1.5 Pin Configuration

- 176-pin plastic LQFP (fine pitch) (24 × 24)
- μ PD703111GM-10-UEU
- μ PD703111GM-13-UEU
- ★ μ PD703111GM-15-UEU

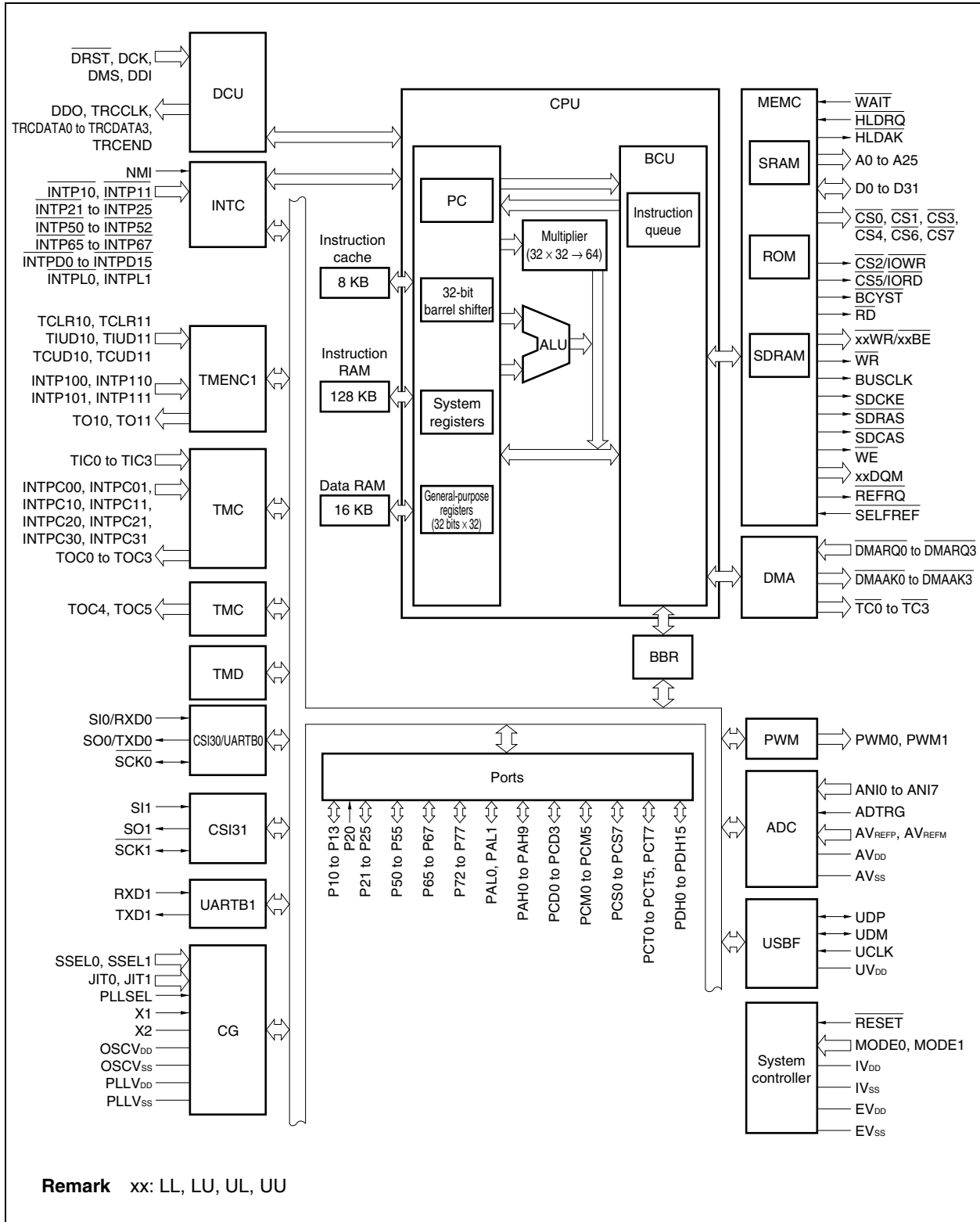


Pin Identification

A0 to A25:	Address bus	P72 to P77:	Port 7
ADTRG:	A/D trigger input	PAH0 to PAH9:	Port AH
ANI0 to ANI7:	Analog input	PAL0, PAL1:	Port AL
AV _{DD} :	Analog power supply	PCD0 to PCD3:	Port CD
AV _{REFM} :	Analog reference voltage	PCM0 to PCM5:	Port CM
AV _{REFP} :	Analog reference voltage	PCS0 to PCS7:	Port CS
AV _{SS} :	Analog ground	PCT0 to PCT5, PCT7:	Port CT
$\overline{\text{BCYST}}$:	Bus cycle start timing	PDH0 to PDH15:	Port DH
BUSCLK:	Bus clock output	PLLSEL:	PLL operating mode select
$\overline{\text{CS0}}$ to $\overline{\text{CS7}}$:	Chip select	PLL _{VDD} :	PLL power supply
D0 to D31:	Data bus	PLL _{SS} :	PLL ground
DCK:	Debug clock input	PWM0, PWM1:	Pulse width modulation
DDI:	Debug data input	$\overline{\text{RD}}$:	Read strobe
DDO:	Debug data output	$\overline{\text{REFRQ}}$:	Refresh request
$\overline{\text{DRST}}$:	Debug reset	$\overline{\text{RESET}}$:	Reset
DMS:	Debug mode	RXD0, RXD1:	Receive data
$\overline{\text{DMAAK0}}$ to $\overline{\text{DMAAK3}}$:	DMA acknowledge	$\overline{\text{SCK0}}$, $\overline{\text{SCK1}}$:	Serial clock
$\overline{\text{DMARQ0}}$ to $\overline{\text{DMARQ3}}$:	DMA request	$\overline{\text{SDCAS}}$:	SDRAM column address strobe
EV _{DD} :	Power supply for external pins	SDCKE:	SDRAM clock enable
EV _{SS} :	Ground for external pins	$\overline{\text{SDRAS}}$:	SDRAM row address strobe
$\overline{\text{HLDK}}$:	Hold acknowledge	$\overline{\text{SELFREF}}$:	Self-refresh request
$\overline{\text{HLDRQ}}$:	Hold request	SI0, SI1:	Serial input
$\overline{\text{INTP10}}$, $\overline{\text{INTP11}}$, :	Interrupt request from peripherals	SO0, SO1:	Serial output
$\overline{\text{INTP21}}$ to $\overline{\text{INTP25}}$,		SSEL0, SSEL1:	Clock generator operating mode select
$\overline{\text{INTP50}}$ to $\overline{\text{INTP52}}$,		$\overline{\text{TC0}}$ to $\overline{\text{TC3}}$:	Terminal count signal
$\overline{\text{INTP65}}$ to $\overline{\text{INTP67}}$,		TCLR10, TCLR11:	Timer clear
INTP100, INTP101,		TCUD10, TCUD11:	Timer control pulse input
INTP110, INTP111,		TIC0 to TIC3:	Timer input
INTPC00, INTPC01,		TIUD10, TIUD11:	Timer count pulse input
INTPC10, INTPC11,		TO10, TO11, :	Timer output
INTPC20, INTPC21,		TOC0 to TOC5	
INTPC30, INTPC31,		TRCCLK:	Trace clock
$\overline{\text{INTPD0}}$ to $\overline{\text{INTPD15}}$	TRCDATA0 to:	Trace data output	
$\overline{\text{INTPL0}}$, $\overline{\text{INTPL1}}$	TRCDATA3		
$\overline{\text{IRD}}$:	I/O read strobe	TRCEND:	Trace end status output
$\overline{\text{IWR}}$:	I/O write strobe	TXD0, TXD1:	Transmit data
IV _{DD} :	Power supply for internal unit	UCLK:	USB external clock input
IV _{SS} :	Ground for internal unit	UDM:	USB data input & output (-)
JIT0, JIT1:	SSCG jitter select	UDP:	USB data input & output (+)
$\overline{\text{LLBE}}$:	Lower byte enable (D0 to D7)	$\overline{\text{ULBE}}$:	Upper byte enable (D16 to D23)
LLDQM:	Lower DQ mask enable (D0 to D7)	ULDQM:	Upper DQ mask enable (D16 to D23)
$\overline{\text{LLWR}}$:	Lower write strobe (D0 to D7)	$\overline{\text{ULWR}}$:	Upper write strobe (D16 to D23)
$\overline{\text{LUBE}}$:	Lower byte enable (D8 to D15)	$\overline{\text{UUBE}}$:	Upper byte enable (D24 to D31)
LUDQM:	Lower DQ mask enable (D8 to D15)	UUDQM:	Upper DQ mask enable (D24 to D31)
$\overline{\text{LUWR}}$:	Lower write strobe (D8 to D15)	$\overline{\text{UUWR}}$:	Upper write strobe (D24 to D31)
MODE0, MODE1:	Mode	UV _{DD} :	Power supply for USB unit
NMI:	Non-maskable interrupt request	$\overline{\text{WAIT}}$:	Wait
OSCV _{DD} :	Clock generator power supply	$\overline{\text{WE}}$:	Write enable
OSCV _{SS} :	Clock generator ground	$\overline{\text{WR}}$:	Write strobe output enable
P10 to P13:	Port 1	X1, X2:	Crystal
P20 to P25:	Port 2		
P50 to P55:	Port 5		
P65 to P67:	Port 6		

1.6 Function Blocks

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 a multiplier (16 bits \times 16 bits \rightarrow 32 bits or 32 bits \times 32 bits \rightarrow 64 bits) and a barrel shifter (32 bits), help accelerate processing of complex instructions.

(2) Bus control unit (BCU)

The BCU starts the 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 a DRAM controller (DRAMC), page ROM controller (ROMC), and DMA controller (DMAC) and performs external memory access and DMA transfer.

(a) SDRAM controller

The SDRAM controller generates the $\overline{\text{SDRAS}}$, $\overline{\text{SDCAS}}$, $\overline{\text{UUDQM}}$, $\overline{\text{ULDQM}}$, $\overline{\text{LUDQM}}$, and $\overline{\text{LLDQM}}$ signals and performs access control for SDRAM.

CAS latency 1 (excluding flyby DMA transfer), 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)

This controller controls data transfer between memory and I/O instead of the CPU.

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

(3) RAM

Instruction RAM (128 KB) and data RAM (16 KB) are provided.

The instruction RAM can be accessed in one clock from the CPU when an instruction is fetched. Its write access time depends on the BUSCLK frequency to the CS0 space and the number of wait cycles. This RAM is mapped from address 00000000H.

The data RAM can be accessed in one clock from the CPU when its data is read. It is mapped from address FFFF8000H.

(4) Cache

An instruction cache (8 KB) is provided.

(5) Interrupt controller (INTC)

This controller handles hardware interrupt requests (NMI, INTP_n) from on-chip peripheral I/O and external hardware. Eight levels of interrupt priorities can be specified for these interrupt requests, and multiple-interrupt servicing control can be performed for interrupt sources (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, L1, C00, C01, C10, C11, C20, C21, C30, C31).

(6) Clock generator (CG)

This clock generator supplies frequencies which are 8 times the input clock (F_x) (using an on-chip PLL) as the internal system clock (f_{CLK}). As the input clock, an external oscillator is input from pins X1 and X2.

(7) Real-time pulse unit (RPU)

This unit incorporates a 6-channel 16-bit timer/event counter, 4-channel 16-bit interval timer, and 2-channel 16-bit up/down counter/timer for 2-phase encoder input and can measure pulse widths or frequency and output a programmable pulse.

(8) Serial interfaces (SIO)

The serial interfaces consist of 3 channels divided between an asynchronous serial interface B (UARTB) and clocked serial interface 3 (CSI3). Of these 3 channels, one is alternative with UARTB and CSI3, one is fixed to CSI3, and one is fixed to UARTB.

UARTB transfers data by using the TXD_n and RXD_n pins (n = 0, 1).

CSI3 transfers data by using the SOn, SIn, and $\overline{\text{SCKn}}$ pins (n = 0, 1).

In addition, a USB function controller (USBF) is also provided.

(9) A/D converter (ADC)

This high-speed, high-resolution 10-bit A/D converter includes 8 analog input pins. Conversion is performed using the successive approximation method.

(10) PWM

Two channels for PWM signal output of 16-bit resolution have been provided. By connecting an external low-pass filter, PWM output can be used as digital to analog conversion output. PWM is ideal for actuator control signals such as those in motors.

(11) Ports

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

Port	I/O	Control Function
Port 1	4-bit I/O	Serial interface I/O, USB clock signal input, external interrupt input
Port 2	1-bit input, 5-bit I/O	NMI input, serial interface I/O, external interrupt input
Port 5	6-bit I/O	DMA controller I/O, external interrupt input, real-time pulse unit I/O
Port 6	3-bit I/O	Real-time pulse unit I/O, external interrupt input
Port 7	6-bit I/O	DMA controller I/O, real-time pulse unit I/O, external interrupt input
Port AL	2-bit I/O	External address bus, external interrupt input
Port AH	10-bit I/O	External address bus
Port DH	16-bit I/O	External data bus, external interrupt input, PWM output, real-time pulse unit I/O
Port CS	8-bit I/O	External bus interface control signal output
Port CT	7-bit I/O	External bus interface control signal output
Port CM	6-bit I/O	Wait insertion signal input, external bus interface control signal I/O, self-refresh request signal input, A/D converter external trigger input
Port CD	4-bit I/O	External bus interface control signal output, bus clock output

CHAPTER 2 PIN FUNCTIONS

The names and functions of the pins in the V850E/ME2 are listed below. These pins can be divided into port pins and non-port pins according to their functions.

2.1 List of Pin Functions

(1) Port pins

(1/3)

Pin Name	I/O	Function	Alternate Function
P10	I/O	Port 1 4-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{INTP10/UCLK}}$
P11			$\overline{\text{INTP11/SCK0}}$
P12			SI0/RXD0
P13			SO0/TXD0
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. P21 to P25 are a 5-bit I/O port. Input/output can be specified in 1-bit units.	NMI
P21	I/O		$\overline{\text{INTP21/RXD1}}$
P22			$\overline{\text{INTP22/TXD1}}$
P23			$\overline{\text{INTP23/SCK1}}$
P24			$\overline{\text{INTP24/SI1}}$
P25			$\overline{\text{INTP25/SO1}}$
P50	I/O	Port 5 6-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{INTP50/DMARQ0}}$
P51			$\overline{\text{INTP51/DMAAK0}}$
P52			$\overline{\text{INTP52/TC0}}$
P53			$\overline{\text{INTPC00/TIC0/DMARQ1}}$
P54			$\overline{\text{INTPC01/DMAAK1}}$
P55			$\overline{\text{TOC0/TC1}}$
P65	I/O	Port 6 3-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{INTP65/INTPC10/TIC1}}$
P66			$\overline{\text{INTP66/INTPC11}}$
P67			$\overline{\text{INTP67/TOC1}}$
P72	I/O	Port 7 6-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{INTPC20/TIC2/DMARQ2}}$
P73			$\overline{\text{INTPC21/DMAAK2}}$
P74			$\overline{\text{TOC2/TC2}}$
P75			$\overline{\text{INTPC30/TIC3/DMARQ3}}$
P76			$\overline{\text{INTPC31/DMAAK3}}$
P77			$\overline{\text{TOC3/TC3}}$
PAH0 to PAH9	I/O	Port AH 8-/10-bit I/O port Input/output can be specified in 1-bit units.	A16 to A25

Pin Name	I/O	Function	Alternate Function
PAL0	I/O	Port AL 2-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{INTPL0}}/\text{A0}$
PAL1			$\overline{\text{INTPL1}}/\text{A1}$
PDH0	I/O	Port DH 8-/16-bit I/O port Input/output can be specified in 1-bit units.	$\text{D16}/\overline{\text{INTPD0}}$
PDH1			$\text{D17}/\overline{\text{INTPD1}}$
PDH2			$\text{D18}/\overline{\text{INTPD2}}/\text{TOC4}$
PDH3			$\text{D19}/\overline{\text{INTPD3}}$
PDH4			$\text{D20}/\overline{\text{INTPD4}}$
PDH5			$\text{D21}/\overline{\text{INTPD5}}/\text{TOC5}$
PDH6			$\text{D22}/\overline{\text{INTPD6}}/\overline{\text{INTP100}}/\text{TCUD10}$
PDH7			$\text{D23}/\overline{\text{INTPD7}}/\overline{\text{INTP101}}/\text{TCLR10}$
PDH8			$\text{D24}/\overline{\text{INTPD8}}/\text{TO10}$
PDH9			$\text{D25}/\overline{\text{INTPD9}}/\text{TIUD10}$
PDH10			$\text{D26}/\overline{\text{INTPD10}}/\overline{\text{INTP110}}/\text{TCUD11}$
PDH11			$\text{D27}/\overline{\text{INTPD11}}/\overline{\text{INTP111}}/\text{TCLR11}$
PDH12			$\text{D28}/\overline{\text{INTPD12}}/\text{TO11}$
PDH13			$\text{D29}/\overline{\text{INTPD13}}/\text{TIUD11}$
PDH14			$\text{D30}/\overline{\text{INTPD14}}/\text{PWM0}$
PDH15			$\text{D31}/\overline{\text{INTPD15}}/\text{PWM1}$
PCD0	I/O	Port CD 4-bit I/O port Input/output can be specified in 1-bit units.	SDCKE
PCD1			BUSCLK
PCD2			$\overline{\text{SDCAS}}$
PCD3			$\overline{\text{SDRAS}}$
PCM0	I/O	Port CM 6-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{WAIT}}$
PCM1			-
PCM2			$\overline{\text{HLDAK}}$
PCM3			$\overline{\text{HLDRQ}}$
PCM4			$\overline{\text{REFRQ}}$
PCM5			$\text{ADTRG}/\overline{\text{SELFREF}}$
PCS0	I/O	Port CS 8-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{CS0}}$
PCS1			$\overline{\text{CS1}}$
PCS2			$\overline{\text{CS2}}/\overline{\text{IOWR}}$
PCS3			$\overline{\text{CS3}}$
PCS4			$\overline{\text{CS4}}$
PCS5			$\overline{\text{CS5}}/\overline{\text{IORD}}$
PCS6			$\overline{\text{CS6}}$
PCS7			$\overline{\text{CS7}}$

Pin Name	I/O	Function	Alternate Function
PCT0	I/O	Port CT 7-bit I/O port Input/output can be specified in 1-bit units.	$\overline{\text{LLWR}}/\overline{\text{LLBE}}/\overline{\text{LLDQM}}$
PCT1			$\overline{\text{LUWR}}/\overline{\text{LUBE}}/\overline{\text{LUDQM}}$
PCT2			$\overline{\text{ULWR}}/\overline{\text{ULBE}}/\overline{\text{ULDQM}}$
PCT3			$\overline{\text{UUWR}}/\overline{\text{UUBE}}/\overline{\text{UUDQM}}$
PCT4			$\overline{\text{RD}}$
PCT5			$\overline{\text{WE}}/\overline{\text{WR}}$
PCT6			
PCT7			$\overline{\text{BCYST}}$

(2) Non-port pins

(1/5)

Pin Name	I/O	Function	Alternate Function
A0	Output	26-bit address bus for external memory	PAL0/ $\overline{\text{INTPL0}}$
A1			PAL1/ $\overline{\text{INTPL1}}$
A2 to A9			–
A10 to A15			–
A16 to A25			PAH0 to PAH9
ADTRG			Input
ANI0 to ANI7	Input	Analog inputs to A/D converter	–
AV _{DD}	–	3.3 V positive power supply for A/D converter	–
AV _{REFM}	Input	Reference voltage applied to A/D converter	–
AV _{REFP}			–
AV _{SS}	–	Ground potential for A/D converter	–
$\overline{\text{BCYST}}$	Output	Strobe signal output that shows the start of the bus cycle	PCT7
BUSCLK	Output	Clock output for SDRAM	PCD1
$\overline{\text{CS0}}$	Output	Chip select signal output	PCS0
$\overline{\text{CS1}}$			PCS1
$\overline{\text{CS2}}$			PCS2/ $\overline{\text{IOWR}}$
$\overline{\text{CS3}}$			PCS3
$\overline{\text{CS4}}$			PCS4
$\overline{\text{CS5}}$			PCS5/ $\overline{\text{IORD}}$
$\overline{\text{CS6}}$			PCS6
$\overline{\text{CS7}}$			PCS7
D0 to D15			I/O
D16	PDH0/ $\overline{\text{INTPD0}}$		
D17	PDH1/ $\overline{\text{INTPD1}}$		
D18	PDH2/ $\overline{\text{INTPD2}}$ /TOC4		
D19	PDH3/ $\overline{\text{INTPD3}}$		
D20	PDH4/ $\overline{\text{INTPD4}}$		
D21	PDH5/ $\overline{\text{INTPD5}}$ /TOC5		
D22	PDH6/ $\overline{\text{INTPD6}}$ / $\overline{\text{INTP100}}$ /TCUD10		
D23	PDH7/ $\overline{\text{INTPD7}}$ / $\overline{\text{INTP101}}$ /TCLR10		
D24	PDH8/ $\overline{\text{INTPD8}}$ /TO10		
D25	PDH9/ $\overline{\text{INTPD9}}$ /TIUD10		
D26	PDH10/ $\overline{\text{INTPD10}}$ / $\overline{\text{INTP110}}$ /TCUD11		
D27	PDH11/ $\overline{\text{INTPD11}}$ / $\overline{\text{INTP111}}$ /TCLR11		
D28	PDH12/ $\overline{\text{INTPD12}}$ /TO11		
D29	PDH13/ $\overline{\text{INTPD13}}$ /TIUD11		
D30	PDH14/ $\overline{\text{INTPD14}}$ /PWM0		
D31	PDH15/ $\overline{\text{INTPD15}}$ /PWM1		

Pin Name	I/O	Function	Alternate Function
DCK	Input	Debug clock input	–
DDI	Input	Debug data input	–
DDO	Output	Debug data output	–
$\overline{\text{DMAAK0}}$	Output	DMA acknowledge signal output	P51/ $\overline{\text{INTP51}}$
$\overline{\text{DMAAK1}}$			P54/ $\overline{\text{INTPC01}}$
$\overline{\text{DMAAK2}}$			P73/ $\overline{\text{INTPC21}}$
$\overline{\text{DMAAK3}}$			P76/ $\overline{\text{INTPC31}}$
$\overline{\text{DMARQ0}}$	Input	DMA request signal input	P50/ $\overline{\text{INTP50}}$
$\overline{\text{DMARQ1}}$			P53/ $\overline{\text{INTPC00/TIC0}}$
$\overline{\text{DMARQ2}}$			P72/ $\overline{\text{INTPC20/TIC2}}$
$\overline{\text{DMARQ3}}$			P75/ $\overline{\text{INTPC30/TIC3}}$
DMS	Input	Debug mode select	–
$\overline{\text{DRST}}$	Input	Reset input for debug	–
EV _{DD}	–	3.3 V positive power supply for external pin	–
EV _{SS}	–	Ground potential for external pin	–
$\overline{\text{HLDK}}$	Output	Bus hold acknowledge output	PCM2
$\overline{\text{HLDRQ}}$	Input	Bus hold request input	PCM3
$\overline{\text{INTP10}}$	Input	External maskable interrupt request input	P10/ $\overline{\text{UCLK}}$
$\overline{\text{INTP11}}$			P11/ $\overline{\text{SCK0}}$
$\overline{\text{INTP21}}$			P21/ $\overline{\text{RXD1}}$
$\overline{\text{INTP22}}$			P22/ $\overline{\text{TXD1}}$
$\overline{\text{INTP23}}$			P23/ $\overline{\text{SCK1}}$
$\overline{\text{INTP24}}$			P24/ $\overline{\text{SI1}}$
$\overline{\text{INTP25}}$			P25/ $\overline{\text{SO1}}$
$\overline{\text{INTP50}}$			P50/ $\overline{\text{DMARQ0}}$
$\overline{\text{INTP51}}$			P51/ $\overline{\text{DMAAK0}}$
$\overline{\text{INTP52}}$			P52/ $\overline{\text{TC0}}$
$\overline{\text{INTP65}}$			P65/ $\overline{\text{TIC1/INTPC10}}$
$\overline{\text{INTP66}}$			P66/ $\overline{\text{INTPC11}}$
$\overline{\text{INTP67}}$			P67/ $\overline{\text{TOC1}}$
$\overline{\text{INTPD0}}$			PDH0/ $\overline{\text{D16}}$
$\overline{\text{INTPD1}}$			PDH1/ $\overline{\text{D17}}$
$\overline{\text{INTPD2}}$			PDH2/ $\overline{\text{TOC4/D18}}$
$\overline{\text{INTPD3}}$			PDH3/ $\overline{\text{D19}}$
$\overline{\text{INTPD4}}$			PDH4/ $\overline{\text{D20}}$
$\overline{\text{INTPD5}}$			PDH5/ $\overline{\text{TOC5/D21}}$
$\overline{\text{INTPD6}}$			PDH6/ $\overline{\text{INTP100/TCUD10/D22}}$
$\overline{\text{INTPD7}}$			PDH7/ $\overline{\text{INTP101/TCLR10/D23}}$
$\overline{\text{INTPD8}}$	PDH8/ $\overline{\text{TO10/D24}}$		
$\overline{\text{INTPD9}}$	PDH9/ $\overline{\text{TIUD10/D25}}$		
$\overline{\text{INTPD10}}$	PDH10/ $\overline{\text{INTP110/TCUD11/D26}}$		

Pin Name	I/O	Function	Alternate Function
$\overline{\text{INTPD11}}$	Input	External maskable interrupt request input	PDH11/ $\overline{\text{INTPD11}}$ / $\overline{\text{TCLR11}}$ /D27
$\overline{\text{INTPD12}}$			PDH12/ $\overline{\text{TO11}}$ /D28
$\overline{\text{INTPD13}}$			PDH13/ $\overline{\text{TIUD11}}$ /D29
$\overline{\text{INTPD14}}$			PDH14/ $\overline{\text{PWM0}}$ /D30
$\overline{\text{INTPD15}}$			PDH15/ $\overline{\text{PWM1}}$ /D31
$\overline{\text{INTPL0}}$			PAL0/A0
$\overline{\text{INTPL1}}$			PAL1/A1
$\overline{\text{INTP100}}$	Input	Timer ENC10 external capture trigger input	PDH6/ $\overline{\text{TCUD10}}$ /D22/ $\overline{\text{INTPD6}}$
$\overline{\text{INTP101}}$			PDH7/ $\overline{\text{TCLR10}}$ /D23/ $\overline{\text{INTPD7}}$
$\overline{\text{INTP110}}$	Input	Timer ENC11 external capture trigger input	PDH10/ $\overline{\text{TCUD11}}$ /D26/ $\overline{\text{INTPD10}}$
$\overline{\text{INTP111}}$			PDH11/ $\overline{\text{TCLR11}}$ /D27/ $\overline{\text{INTPD11}}$
$\overline{\text{INTPC00}}$	Input	External maskable interrupt request input/timer C0 external capture trigger input	P53/ $\overline{\text{TIC0}}$ / $\overline{\text{DMARQ1}}$
$\overline{\text{INTPC01}}$			P54/ $\overline{\text{DMAAK1}}$
$\overline{\text{INTPC10}}$		External maskable interrupt request input/timer C1 external capture trigger input	P65/ $\overline{\text{INTP65}}$ / $\overline{\text{TIC1}}$
$\overline{\text{INTPC11}}$			P66/ $\overline{\text{INTP66}}$
$\overline{\text{INTPC20}}$		External maskable interrupt request input/timer C2 external capture trigger input	P72/ $\overline{\text{TIC2}}$ / $\overline{\text{DMARQ2}}$
$\overline{\text{INTPC21}}$			P73/ $\overline{\text{DMAAK2}}$
$\overline{\text{INTPC30}}$		External maskable interrupt request input/timer C3 external capture trigger input	P75/ $\overline{\text{TIC3}}$ / $\overline{\text{DMARQ3}}$
$\overline{\text{INTPC31}}$			P76/ $\overline{\text{DMAAK3}}$
$\overline{\text{IORD}}$	Output	DMA read strobe signal output	PCS5/ $\overline{\text{CS5}}$
$\overline{\text{IOWR}}$	Output	DMA write strobe signal output	PCS2/ $\overline{\text{CS2}}$
IV_{DD}	–	1.5 V positive power supply for internal unit	–
IV_{SS}	–	Ground potential for internal unit	–
JIT0	Input	Specifying SSSCG operating mode	–
JIT1			–
$\overline{\text{LLBE}}$	Output	External data bus byte enable signal output (lowest byte (D0 to D7))	PCT0/ $\overline{\text{LLDQM}}$ / $\overline{\text{LLWR}}$
$\overline{\text{LLDQM}}$	Output	Output disable/write mask signal output for SDRAM (lowest byte (D0 to D7))	PCT0/ $\overline{\text{LLWR}}$ / $\overline{\text{LLBE}}$
$\overline{\text{LLWR}}$	Output	External data bus write strobe signal output (lowest byte (D0 to D7))	PCT0/ $\overline{\text{LLBE}}$ / $\overline{\text{LLDQM}}$
$\overline{\text{LUBE}}$	Output	External data bus byte enable signal output (third byte (D8 to D15))	PCT1/ $\overline{\text{LUDQM}}$ / $\overline{\text{LUWR}}$
$\overline{\text{LUDQM}}$	Output	Output disable/write mask signal output for SDRAM (third byte (D8 to D15))	PCT1/ $\overline{\text{LUWR}}$ / $\overline{\text{LUBE}}$
$\overline{\text{LUWR}}$	Output	External data bus write strobe signal output (third byte (D8 to D15))	PCT1/ $\overline{\text{LUBE}}$ / $\overline{\text{LUDQM}}$
MODE0	Input	Specifying V850E/ME2 operating mode	–
MODE1			–
NMI	Input	Non-maskable interrupt request signal input	P20
OSCV_{DD}	–	3.3 V positive power supply for oscillator	–
OSCV_{SS}	–	Ground potential for oscillator	–
PLLSEL	Input	Input specifying PLL operating mode	–

Pin Name	I/O	Function	Alternate Function
PLL _{VDD}	–	1.5 V positive power supply for PLL synthesizer	–
PLL _{VSS}		Ground potential for PLL synthesizer	–
PWM0	Output	PWM pulse signal output	PDH14/D30/ $\overline{\text{INTPD14}}$
PWM1			PDH15/D31/ $\overline{\text{INTPD15}}$
$\overline{\text{RD}}$	Output	External data bus read strobe signal output	PCT4
$\overline{\text{REFRQ}}$	Output	Refresh request signal output for SDRAM	PCM4
$\overline{\text{RESET}}$	Input	System reset input	–
RXD0	Input	UARTB0 and UARTB1 serial receive data input	P12/SI0
RXD1			P21/ $\overline{\text{INTP21}}$
$\overline{\text{SCK0}}$	I/O	CSI30 and CSI31 serial clock I/O (3-wire)	P11/ $\overline{\text{INTP11}}$
$\overline{\text{SCK1}}$			P23/ $\overline{\text{INTP23}}$
$\overline{\text{SDCAS}}$	Output	Column address strobe signal output for SDRAM	PCD2
SDCKE	Output	SDRAM clock enable signal output	PCD0
$\overline{\text{SDRAS}}$	Output	Row address strobe signal output for SDRAM	PCD3
$\overline{\text{SELFREF}}$	Input	Self-refresh request input for SDRAM	PCM5/ADTRG
SI0	Input	CSI30 and CSI31 serial receive data input (3-wire)	P12/RXD0
SI1			P24/ $\overline{\text{INTP24}}$
SO0	Output	CSI30 and CSI31 serial transmit data output (3-wire)	P13/TXD0
SO1			P25/ $\overline{\text{INTP25}}$
SSEL0	Input	Specifying the clock generator's operating mode	–
SSEL1			–
$\overline{\text{TC0}}$	Output	DMA transfer end (terminal count) signal output	P52/ $\overline{\text{INTP52}}$
$\overline{\text{TC1}}$			P55/TOC0
$\overline{\text{TC2}}$			P74/TOC2
$\overline{\text{TC3}}$			P77/TOC3
TCLR10	Input	Clear signal input to timer ENC10 and ENC11	PDH7/D23/ $\overline{\text{INTPD7}}$ / $\overline{\text{INTP101}}$
TCLR11			PDH11/D27/ $\overline{\text{INTPD11}}$ / $\overline{\text{INTP111}}$
TCUD10	Input	Count operation switching signal input to timer ENC10 and ENC11	PDH6/D22/ $\overline{\text{INTPD6}}$ / $\overline{\text{INTP100}}$
TCUD11			PDH10/D26/ $\overline{\text{INTPD10}}$ / $\overline{\text{INTP110}}$
TIC0	Input	External count clock input of timer C0 to C3	P53/ $\overline{\text{DMARQ1}}$ / $\overline{\text{INTPC00}}$
TIC1			P65/ $\overline{\text{INTP65}}$ / $\overline{\text{INTPC10}}$
TIC2			P72/ $\overline{\text{INTPC20}}$ / $\overline{\text{DMARQ2}}$
TIC3			P75/ $\overline{\text{INTPC30}}$ / $\overline{\text{DMARQ3}}$
TIUD10	Input	External count clock input to timer ENC10 and ENC11	PDH9/D25/ $\overline{\text{INTPD9}}$
TIUD11			PDH13/D29/ $\overline{\text{INTPD13}}$
TO10	Output	Pulse signal output of timer ENC10 and ENC11	PDH8/D24/ $\overline{\text{INTPD8}}$
TO11			PDH12/D28/ $\overline{\text{INTPD12}}$

Pin Name	I/O	Function	Alternate Function
TOC0	Output	Pulse signal output of timer C0 to C5	P55/ $\overline{TC1}$
TOC1			P67/ $\overline{INTP67}$
TOC2			P74/ $\overline{TC2}$
TOC3			P77/ $\overline{TC3}$
TOC4			PDH2/D18/ $\overline{INTPD2}$
TOC5			PDH5/D21/ $\overline{INTPD5}$
TRCCLK	Output	Trace clock output	–
TRCDATA0	Output	Trace data output (D0 to D3)	–
TRCDATA1	Output		–
TRCDATA2	Output		–
TRCDATA3	Output		–
TRCEND	Output	Trace end status output	–
TXD0	Output	UARTB0 and UARTB1 serial transmit data output	P13/SO0
TXD1			P22/ $\overline{INTP22}$
UCLK ^{Note}	Input	USB clock signal input	P10/ $\overline{INTP10}$
UDM	I/O	USB data I/O (–)	–
UDP	I/O	USB data I/O (+)	–
\overline{ULBE}	Output	External data bus byte enable signal output (second byte (D16 to D23))	PCT2/ $\overline{ULDQM}/\overline{ULWR}$
ULDQM	Output	Output disable/write mask signal output for SDRAM (second byte (D16 to D23))	PCT2/ $\overline{ULWR}/\overline{ULBE}$
\overline{ULWR}	Output	External data bus write strobe signal output (second byte (D16 to D23))	PCT2/ $\overline{ULBE}/\overline{ULDQM}$
\overline{UUBE}	Output	External data bus byte enable signal output (highest byte (D24 to D31))	PCT3/ $\overline{UUWR}/\overline{UUDQM}$
UUDQM	Output	Output disable/write mask signal output for SDRAM (highest byte (D24 to D31))	PCT3/ $\overline{UUWR}/\overline{UUBE}$
\overline{UUWR}	Output	External data bus write strobe signal output (highest byte (D24 to D31))	PCT3/ $\overline{UUDQM}/\overline{UUBE}$
UV _{DD}	–	3.3 V positive power supply for USB	–
\overline{WAIT}	Input	Control signal input that inserts a wait in the bus cycle	PCM0
\overline{WE}	Output	Write enable signal output for SDRAM	PCT5/ \overline{WR}
\overline{WR}	Output	Write strobe signal output for SDRAM	PCT5/ \overline{WE}
X1	Input	Connects the crystal resonator for system clock oscillation.	–
X2	–	In the case of an external source supplying the clock, it is input to X1.	–

★ **Note** When using as the UCLK pin, be careful to avoid the input of a staircase waveform due to reflection, etc., or the input of noise.

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 1}
A0 to A1 (PAL0 to PAL1)	Hi-Z	Hi-Z	Operating	Hi-Z
A2 to A15	Hi-Z	Hi-Z	Operating	Hi-Z
A16 to A25 (PAH0 to PAH9)	Hi-Z	Hi-Z	Operating	Hi-Z
D0 to D15	Hi-Z	Hi-Z	Operating	Hi-Z
D16 to D31 (PDH to PDH15)	Hi-Z	Hi-Z	Operating	Hi-Z
$\overline{CS0}$ to $\overline{CS7}$ (PCS0 to PCS7)	Hi-Z	H	Operating	Hi-Z
\overline{IOWR} (PCS2)	×	H	Operating	Hi-Z
\overline{IORD} (PCS5)	×	H	Operating	Hi-Z
\overline{LLWR} , \overline{LUWR} , \overline{ULWR} , \overline{UUWR} (PCT0 to PCT3)	Hi-Z	H	Operating	Hi-Z
\overline{LLBE} , \overline{LUBE} , \overline{ULBE} , \overline{UUBE} (PCT0 to PCT3)	×	H	Operating	Hi-Z
\overline{LLDQM} , \overline{LUDQM} , \overline{ULDQM} , \overline{UUDQM} (PCT0 to PCT3)	×	H	Operating	Hi-Z
\overline{RD} (PCT4)	Hi-Z	H	Operating	Hi-Z
\overline{WR} (PCT5)	Hi-Z	H	Operating	Hi-Z
\overline{WE} (PCT5)	×	H	Operating	Hi-Z
\overline{BCYST} (PCT7)	Hi-Z	H	Operating	Hi-Z
\overline{WAIT} (PCM0)	Hi-Z	–	Operating	–
\overline{HLDK} (PCM2)	Hi-Z	H	Operating	L
\overline{HLDRQ} (PCM3)	Hi-Z	–	Operating	Operating
\overline{REFRQ} (PCM4)	Hi-Z	Operating ^{Note 2}	Operating	Operating
$\overline{SELFREF}$ (PCM5)	Hi-Z	–	Operating	Operating
\overline{SDCKE} (PCD0)	Hi-Z	L ^{Note 2}	Operating	Operating
\overline{BUSCLK} (PCD1)	Operating	L	Operating	Operating
\overline{SDCAS} (PCD2)	Hi-Z	SELF	Operating	Hi-Z
\overline{SDRAS} (PCD3)	Hi-Z	SELF	Operating	Hi-Z
$\overline{DMAAK0}$ (P51)	×	H	Operating	H
$\overline{DMAAK1}$ (P54)	×	H	Operating	H
$\overline{DMAAK2}$ (P73)	×	H	Operating	H
$\overline{DMAAK3}$ (P76)	×	H	Operating	H
Peripheral function input pin other than above	×	–	Operating	Operating
Peripheral function output pin other than above	×	Hold	Operating	Operating
Port input pin other than above	Hi-Z	–	–	Operating
Port output pin other than above	×	Hold	Hold	Operating

Remark Explanation on **Note** and **Remark** are given on the next page.

- Notes 1.** The pin set in the port mode holds the status immediately before.
2. High-level output when the SDRAM controller is not used

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

Notes on turning on/off power

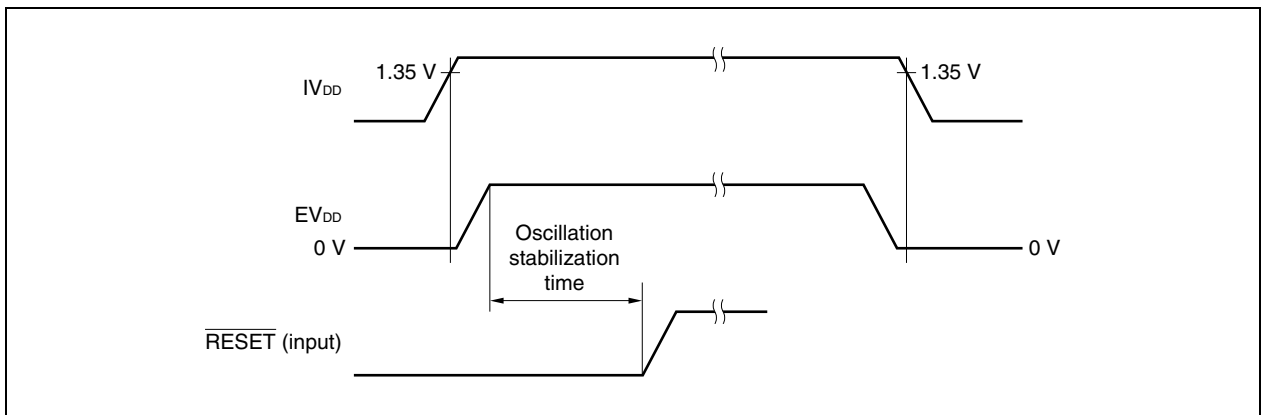
The V850E/ME2 has two power supply pins: a power supply pin for internal units (IV_{DD}) and a power supply pin for external pins (EV_{DD}). The I/O status of an alternate-function I/O pin may be undefined outside the range in which the operation is guaranteed. If this undefined I/O status affects the system, the pin can be made to go into a high-impedance state using the following measure.

• **When turning on power**

Keep the voltage on the EV_{DD} pin at 0 V until the voltage on the IV_{DD} pin reaches the operation guaranteed range (1.35 to 1.65 V).

• **When turning off power**

Keep the voltage on the IV_{DD} pin to within the operation guaranteed range (1.35 to 1.65 V) until the voltage on the EV_{DD} pin drops to 0 V.



2.3 Description of Pin Functions

(1) P10 to P13 (Port 1) ... 3-state I/O

P10 to P13 function as a 4-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 serial interface I/O (UARTB0, CSI30), USB clock signal input, and external interrupt request input.

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

P10 to P13 can be set to input or output in 1-bit units using the port 1 mode register (PM1).

(b) Control mode

P10 to P13 can be set to port/control mode in 1-bit units using the PMC1 register.

(i) $\overline{\text{INTP10}}$, $\overline{\text{INTP11}}$ (Interrupt request from peripherals) ... input

These are external interrupt request input pins.

(ii) SO0 (Serial output) ... output

This is a serial transmit data output pin of CSI30.

(iii) SI0 (Serial input) ... input

This is a serial receive data input pin of CSI30.

(iv) $\overline{\text{SCK0}}$ (Serial clock) ... 3-state I/O

This is a CSI30 serial clock I/O pin.

(v) TXD0 (Transmit data) ... output

This is a serial transmit data output pin of UARTB0.

(vi) RXD0 (Receive data) ... input

This is a serial receive data input pin of UARTB0.

(vii) UCLK (USB clock) ... input

This is a clock input pin of the USB.

★ When using as the UCLK pin, be careful to avoid the input of a staircase waveform due to reflection, etc., or the input of noise.

(2) P20 to P25 (Port 2) ... 3-state I/O

P20 is an input-only pin. P21 to P25 function as 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 serial interface I/O (UARTB1/CSI31) and external interrupt request input.

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

(a) Port mode

P21 to P25 can be set to input or output in 1-bit units using 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

P21 to P25 can be set to port/control mode in 1-bit units using the PMC2 register.

(i) NMI (Non-maskable interrupt request) ... input

This is the non-maskable interrupt request input pin.

(ii) $\overline{\text{INTP21}}$ to $\overline{\text{INTP25}}$ (Interrupt request from peripherals) ... input

These are external interrupt request input pins.

(iii) SO1 (Serial output) ... output

This is a serial transmit data output pin of CSI31.

(iv) SI1 (Serial input) ... input

This is a serial receive data input pin of CSI31.

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

This is a CSI31 serial clock I/O pin.

(vi) TXD1 (Transmit data) ... output

This is a serial transmit data output pin of UARTB1.

(vii) RXD1 (Receive data) ... input

This is a serial receive data input pin of UARTB1.

(3) P50 to P55 (Port 5) ... 3-state I/O

P50 to P55 function as 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 DMA request input, DMA acknowledge output, DMA transfer termination output (terminal count), real-time pulse unit (RPU) I/O, and external interrupt request input.

The operation mode can be set to port or control mode in 1-bit units, specified by the port 5 mode control register (PMC5).

(a) Port mode

P50 to P55 can be set to input or output in 1-bit units using the port 5 mode register (PM5).

(b) Control mode

P50 to P55 can be set to port/control mode in 1-bit units using the PMC5 register.

(i) $\overline{\text{DMARQ0}}$, $\overline{\text{DMARQ1}}$ (DMA request) ... input

These are DMA service request signal input pins. They correspond to DMA channels 0 and 1, respectively, and operate independently of each other. The priority order is fixed to $\overline{\text{DMARQ0}} > \overline{\text{DMARQ1}} > \overline{\text{DMARQ2}} > \overline{\text{DMARQ3}}$.

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

★

(ii) $\overline{\text{DMAAK0}}$, $\overline{\text{DMAAK1}}$ (DMA acknowledge) ... output

These are acknowledge signal output pins that show a DMA service request was granted. They correspond to DMA channels 0 and 1, respectively, and operate independently of each other.

In flyby transfer, these signals become active when external memory is being accessed and internal instruction RAM (in the write mode) is being accessed. When DMA transfers are being executed between internal data RAM, internal instruction RAM (in the read mode), and on-chip peripheral I/O, they do not become active.

In 2-cycle transfer, these are used as the signals to control the $\overline{\text{DMARQ0}}$ and $\overline{\text{DMARQ1}}$ signals.

(iii) $\overline{\text{TC0}}$, $\overline{\text{TC1}}$ (Terminal count) ... output

These are terminal count signal output pins that show that the DMA transfer from the DMA controller is complete. These pins correspond to DMA channels 0 and 1 respectively, and operate independently of each other. The terminal count signals of DMA channels 0 to 3 can be commonly output from the $\overline{\text{TC0}}$ pin.

These signals become active for 1 clock at the rising edge of the BUSCLK signal.

(iv) $\overline{\text{INTPC00}}$, $\overline{\text{INTPC01}}$ (Interrupt request from peripherals) ... input

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

(v) $\overline{\text{TIC0}}$ (Timer input) ... input

This is an external count clock input pin of timer C0.

(vi) $\overline{\text{TOC0}}$ (Timer output) ... output

This is a pulse signal output pin of timer C0.

(vii) $\overline{\text{INTP50}}$ to $\overline{\text{INTP52}}$ (Interrupt request from peripherals) ... input

These are external interrupt request input pins.

(4) P65 to P67 (Port 6) ... 3-state I/O

P65 to P67 function as a 3-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 real-time pulse unit (RPU) I/O and external interrupt request input.

The operation mode can be set to port or control mode in 1-bit units, specified by the port 6 mode control register (PMC6).

(a) Port mode

P65 to P67 can be set to input or output in 1-bit units using the port 6 mode register (PM6).

(b) Control mode

P65 to P67 can be set to port/control mode in 1-bit units using the PMC6 register.

(i) TIC1 (Timer input) ... input

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

(ii) TOC1 (Timer output) ... output

This is the pulse signal output pin for timer C1.

(iii) $\overline{\text{INTP65}}$ to $\overline{\text{INTP67}}$ (Interrupt request from peripherals) ... input

These are external interrupt request input pins.

(iv) INTPC10, INTPC11 (Interrupt request from peripherals) ... input

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

(5) P72 to P77 (Port 7) ... 3-state I/O

P72 to P77 function as 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 DMA request input, DMA acknowledge output, DMA transfer termination output (terminal count), real-time pulse unit (RPU) I/O, and external interrupt request input.

The operation mode can be set to port or control mode in 1-bit units, specified by the port 7 mode control register (PMC7).

(a) Port mode

P72 to P77 can be set to input or output in 1-bit units using the port 7 mode register (PM7).

(b) Control mode

P72 to P77 can be set to port/control mode in 1-bit units using the PMC7 register.

(i) $\overline{\text{DMARQ2}}$, $\overline{\text{DMARQ3}}$ (DMA request) ... input

These are DMA service request signal input pins. They correspond to DMA channels 2 and 3, respectively, and operate independently of each other. The priority order is fixed to $\overline{\text{DMARQ0}} > \overline{\text{DMARQ1}} > \overline{\text{DMARQ2}} > \overline{\text{DMARQ3}}$.

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

★

(ii) $\overline{\text{DMAAK2}}$, $\overline{\text{DMAAK3}}$ (DMA acknowledge) ... output

These are acknowledge signal output pins that show a DMA service request was granted. They correspond to DMA channels 2 and 3, respectively, and operate independently of each other.

In flyby transfer, these signals become active only when external memory is being accessed and internal instruction RAM (in the write mode) is being accessed. When DMA transfers are being executed between internal data RAM, internal instruction RAM (in the read mode), and on-chip peripheral I/O, they do not become active.

In 2-cycle transfer, these are used as the signals to control the $\overline{\text{DMARQ2}}$ and $\overline{\text{DMARQ3}}$ signals.

(iii) $\overline{\text{TC2}}$, $\overline{\text{TC3}}$ (Terminal count) ... output

These are terminal count signal output pins that show that DMA transfer from the DMA controller is complete. These pins correspond to DMA channels 2 and 3 respectively, and operate independently of each other. The terminal count signals of DMA channels 0 to 3 can be commonly output from the $\overline{\text{TC0}}$ pin.

These signals become active for 1 clock at the rising edge of the BUSCLK signal.

(iv) INTPC20 , INTPC21 , INTPC30 , INTPC31 (Interrupt request from peripherals) ... input

These are external interrupt request input pins and the external capture trigger input pins of timers C2 and C3.

(v) TIC2 , TIC3 (Timer input) ... input

These are external count clock input pins of timers C2 and C3.

(vi) TOC2 , TOC3 (Timer output) ... output

These are pulse signal output pins of timers C2 and C3.

(6) PCM0 to PCM5 (Port CM) ... 3-state I/O

PCM0 to PCM5 function as a 6-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 wait insertion signal input, bus hold control signal, refresh request signal output for SDRAM, self-refresh request signal input, and A/D converter external trigger input.

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 PCM5 can be set to input or output in 1-bit units using the port CM mode register (PMCM).

(b) Control mode

PCM0 to PCM5 can be set to port/control mode in 1-bit units using the PMCCM register.

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

This is the control signal input pin at which a data wait is inserted in the bus cycle. The $\overline{\text{WAIT}}$ signal can be input asynchronously to the BUSCLK signal. When the BUSCLK signal falls, sampling is executed. When the set/hold time is not terminated within the sampling timing, wait insertion may not be executed.

(ii) $\overline{\text{HLD\!AK}}$ (Hold acknowledge) ... output

This is the acknowledge signal output pin that indicates the high impedance status for the address bus, data bus, and control bus when the V850E/ME2 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.

(iii) $\overline{\text{HLDRQ}}$ (Hold request) ... input

This is the input pin through which an external device requests the V850E/ME2 to release the address bus, data bus, and control bus. The $\overline{\text{HLDRQ}}$ signal can be input asynchronously to the BUSCLK signal. When this pin is active, the address bus, data bus, and control bus are set to the high impedance status. This occurs either when the V850E/ME2 completes execution of the current bus cycle or immediately if no bus cycle is being executed, then the $\overline{\text{HLD\!AK}}$ 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 $\overline{\text{HLD\!AK}}$ signal is output.

(iv) $\overline{\text{REFRQ}}$ (Refresh request) ... output

This is the refresh request signal output pin for SDRAM.

In cases when the address is decoded by an external circuit to increase the connected SDRAM, or in cases when external SIMM's are connected, this signal is used for 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.

(v) $\overline{\text{SELFREF}}$ (Self refresh request) ... input

This is a self-refresh request signal input pin for SDRAM.

The internal data RAM and internal instruction RAM (in the read mode) can be accessed even in the self-refresh cycle. However, access to a peripheral I/O register or external device is held pending until the self-refresh cycle is cancelled.

Caution Input to the $\overline{\text{SELFREF}}$ pin becomes valid immediately after the reset signal has been cleared. Consequently, if a low level is input to the $\overline{\text{SELFREF}}$ pin by an external pull-down resistor, self refreshing is started. Note that, at this time, the normal instruction fetch cycle does not occur.

(vi) ADTRG (A/D trigger input) ... input

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

(7) PCT0 to PCT5, PCT7 (Port CT) ... 3-state I/O

PCT0 to PCT5 and PCT7 function as a 7-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 to PCT5 and PCT7 can be set to input or output in 1-bit units using the port CT mode register (PMCT).

(b) Control mode

PCT0 to PCT5 and PCT7 can be set to port/control mode in 1-bit units using the PMCCT register.

(i) $\overline{\text{LLWR}}$ (Lower lower byte write strobe) ... 3-state output

This is a strobe signal output pin that 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 lowest byte (D0 to D7) becomes valid. If the bus cycle is a lowest memory write, it becomes active at the falling edge of the BUSCLK signal in the T1 state and becomes inactive at the falling edge of the BUSCLK signal in the T2 state.

(ii) $\overline{\text{LUWR}}$ (Lower upper byte write strobe) ... 3-state output

This is a strobe signal output pin that 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 third byte (D8 to D15) becomes valid. If the bus cycle is a third byte memory write, it becomes active at the falling edge of the BUSCLK signal in the T1 state and becomes inactive at the falling edge of the BUSCLK signal in the T2 state.

(iii) $\overline{\text{ULWR}}$ (Upper lower byte write strobe) ... 3-state output

This is a strobe signal output pin that 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 second byte (D16 to D23) becomes valid. If the bus cycle is a second byte memory write, it becomes active at the falling edge of the BUSCLK signal in the T1 state and becomes inactive at the falling edge of the BUSCLK signal in the T2 state.

(iv) $\overline{\text{UUWR}}$ (Upper upper byte write strobe) ... 3-state output

This is a strobe signal output pin that 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 highest byte (D24 to D31) becomes valid. If the bus cycle is a highest memory write, it becomes active at the falling edge of the BUSCLK signal in the T1 state and becomes inactive at the falling edge of the BUSCLK signal in the T2 state.

(v) $\overline{\text{LLBE}}$ (Lower lower byte enable) ... 3-state output

This is a signal output pin that enables the lowest byte (D0 to D7) of the external data bus.

(vi) $\overline{\text{LUBE}}$ (Lower upper byte enable) ... 3-state output

This is a signal output pin that enables the third byte (D8 to D15) of the external data bus.

(vii) $\overline{\text{ULBE}}$ (Upper lower byte enable) ... 3-state output

This is a signal output pin that enables the second byte (D16 to D23) of the external data bus.

(viii) $\overline{\text{UUBE}}$ (Upper upper byte enable) ... 3-state output

This is a signal output pin that enables the highest byte (D24 to D31) of the external data bus.

(ix) $\overline{\text{LLDQM}}$ (Lower lower DQ mask enable) ... 3-state output

This is a control signal output pin for the data bus to SDRAM. For the data bus, the lowest byte (D0 to D7) is valid. This signal carries out SDRAM output disable control during a read operation, and SDRAM byte mask control during a write operation.

(x) $\overline{\text{LUDQM}}$ (Lower upper DQ mask enable) ... 3-state output

This is a control signal output pin for the data bus to SDRAM. For the data bus, the third byte (D8 to D15) is valid. This signal carries out SDRAM output disable control during a read operation, and SDRAM byte mask control during a write operation.

(xi) $\overline{\text{ULDQM}}$ (Upper lower DQ mask enable) ... 3-state output

This is a control signal output pin for the data bus to SDRAM. For the data bus, the second byte (D16 to D23) is valid. This signal carries out SDRAM output disable control during a read operation, and SDRAM byte mask control during a write operation.

(xii) $\overline{\text{UUDQM}}$ (Upper upper DQ mask enable) ... 3-state output

This is a control signal output pin for the data bus to SDRAM. For the data bus, the highest byte (D24 to D31) is valid. This signal carries out SDRAM output disable control during a read operation, and SDRAM byte mask control during a write operation.

(xiii) $\overline{\text{RD}}$ (Read strobe) ... 3-state output

This is a strobe signal output pin that shows 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.

(xiv) $\overline{\text{WR}}$ (Write strobe) ... 3-state output

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

It becomes active at the falling edge of the BUSCLK signal in the T1 state and becomes inactive at the falling edge of the BUSCLK signal in the T2 state.

(xv) $\overline{\text{WE}}$ (Write enable) ... 3-state output

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

(xvi) $\overline{\text{BCYST}}$ (Bus cycle start timing) ... 3-state output

This is a status signal output pin that shows the start of the bus cycle. It becomes active for 1-clock cycle from the start of each cycle. In the idle state (TI), it becomes inactive.

(8) PCS0 to PCS7 (Port CS) ... 3-state I/O

PCS0 to PCS7 function as an 8-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 to PCS7 can be set to input or output in 1-bit units using the port CS mode register (PMCS).

(b) Control mode

PCS0 to PCS7 can be set to port/control mode in 1-bit units using the PMCCS register.

(i) $\overline{CS0}$ to $\overline{CS7}$ (Chip select) ... 3-state output

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

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

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

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

(ii) \overline{IOWR} (I/O write) ... 3-state output

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

Note that if the IOEN bit of the bus cycle period control register (BCP) is set (1), this signal can be output even in the normal SRAM, external ROM, or external I/O cycle.

(iii) \overline{IORD} (I/O read) ... 3-state output

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

Note that if the IOEN bit of the BCP register is set (1), this signal can be output even in the normal SRAM, external ROM, or external I/O cycle.

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

PCD0 to PCD3 function as 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) ... output

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

(ii) BUSCLK (Clock output) ... output

This is a clock output pin for SDRAM.

(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.

(10) PAH0 to PAH9 (Port AH) ... 3-state I/O

PAH0 to PAH9 function as an 8- or 10-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 an address bus (A16 to A25) 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 PAH9 can be set to input or output in 1-bit units using the port AH mode register (PMAH).

(b) Control mode

PAH0 to PAH9 can be set to function alternately as A16 to A25 using the PMCAH register.

(i) A16 to A25 (Address) ... 3-state output

These are the address output pins of the higher 10 bits of the address bus's 26-bit address when the external memory is accessed.

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

(11) PAL0, PAL1 (Port AL) ... 3-state I/O

PAL0 and PAL1 function as a 2-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 an address bus (A0, A1) and external interrupt request input 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 and PAL1 can be set to input or output in 1-bit units using the port AL mode register (PMAL).

(b) Control mode

PAL0 and PAL1 can be set to port or control mode in 1-bit units using the PMCAL register.

(i) A0, A1 (Address) ... 3-state output

These are the address output pins of the lower 2 bits of the address bus's 26-bit address when the external memory is accessed.

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

(ii) $\overline{\text{INTPL0}}$, $\overline{\text{INTPL1}}$ (Interrupt request from peripherals) ... input

These are external interrupt request input pins.

(12) PDH0 to PDH15 (Port DH) ... 3-state I/O

PDH0 to PDH15 function as 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, these pins operate as real-time pulse unit (RPU) I/O, PWM output, data bus (D16 to D31), and external interrupt request input.

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

(a) Port mode

PDH0 to PDH15 can be set to input or output in 1-bit units using the port DH mode register (PMDH).

(b) Control mode

PDH0 to PDH15 can be set to port or control mode in 1-bit units using the PMCDH register.

(i) $\overline{\text{INTPD0}}$ to $\overline{\text{INTPD15}}$ (Interrupt request from peripherals) ... input

These are external interrupt request input pins.

(ii) TOC4, TOC5 (Timer output) ... output

These are pulse signal output pins of timers C4 and C5.

(iii) INTP100, INTP101, INTP110, INTP111 (Timer capture trigger input) ... input

These are external capture trigger input pins of timers ENC10 and ENC11.

(iv) TIUD10, TIUD11 (Timer count pulse input) ... input

These are external count clock input pins of timers ENC10 and ENC11.

(v) TCUD10, TCUD11 (Timer control pulse input) ... input

These are external count clock input pins of timers ENC10 and ENC11.

(vi) TCLR10, TCLR11 (Timer clear) ... input

These are clear signal input pins of timers ENC10 and ENC11.

(vii) TO10, TO11 (Timer output) ... output

These are pulse signal output pins of timers ENC10 and ENC11.

(viii) PWM0, PWM1 (Pulse width modulation) ... output

These pins output the PWM pulse signal.

(ix) D16 to D31 (Data) ... 3-state I/O

These pins constitute a data bus for when the external memory is accessed. These are the higher 16-bit I/O bus pins of the 32-bit data.

The output changes in synchronization with the rise of the BUSCLK signal in the T1 state.

(13) PLLSEL (PLL operating mode select) ... input

This is an input pin used to specify the PLL operating mode.

(14) PLLV_{DD} (PLL power supply)

This is an 1.5 V positive power supply pin for PLL synthesizer.

(15) PLLV_{SS} (PLL ground)

This is a ground pin for PLL synthesizer.

(16) SSEL0, SSEL1 (Clock generator operating mode select) ... input

These are input pins used to specify the clock generator's operating mode.

(17) JIT0, JIT1 (SSCG (spread spectrum frequency synthesizer phase locked loop) clock generator operating mode select) ... input

These are input pins used to specify the SSCG operating mode.

(18) DCK (Debug clock) ... input

This pin inputs a debug clock. At the rising edge of the DCK signal, the DMS and DDI signals are sampled, and data is output from the DDO pin at the falling edge of the DCK signal. Keep this pin high when the debug function is not used.

(19) DDI (Debug data input) ... input

This pin inputs debug data. This pin is sampled at the rising edge of the DCK signal when the debug serial interface is in the shift state. Data is input with the LSB first. Keep this pin high when the debug function is not used.

(20) DDO (Debug data output) ... 3-state output

This pin outputs debug data. It outputs data at the falling edge of the DCK signal when the debug serial interface is in the shift state. Data is output with the LSB first.

(21) DMS (Debug mode select) ... input

This input pin selects a debug mode. Depending on the level of the DMS signal, the state machine of the debug serial interface changes. This pin is sampled at the rising edge of the DCK signal. Keep this pin high when the debug function is not used.

(22) \overline{DRST} (Debug reset) ... input

This pin inputs a debug reset signal that is a negative-logic signal to initialize the DCU asynchronously. When this signal goes low, the DCU is reset/invalidated. Keep this pin low when the debug function is not used.

(23) MODE0, MODE1 (Mode) ... input

These are input pins used to specify the operating mode.

(24) \overline{RESET} (Reset) ... input

\overline{RESET} is a signal that is input asynchronously and 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).

(25) X1, X2 (Crystal)

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

(26) ANI0 to ANI7 (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_{DD} to pins that are being used as inputs for the A/D converter. If it is possible for noise above the AV_{DD} range or below the AV_{SS} to enter, clamp these pins using a diode that has a small V_F value.

(27) AV_{REFM}, AV_{REFP} (Analog reference voltage) ... input

These are reference voltage supply pins for the A/D converter.

(28) AV_{DD} (Analog power supply)

This is a 3.3 V positive power supply pin for the A/D converter.

(29) AV_{SS} (Analog ground)

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

(30) EV_{DD} (Port power supply)

This is a 3.3 V positive power supply pin for port.

(31) EV_{SS} (Port ground)

This is a ground pin for port.

(32) OSCV_{DD} (Power supply for clock generator)

This is a 3.3 V positive power supply pin for the clock generator.

(33) OSCV_{SS} (Ground for clock generator)

This is a ground pin for the clock generator.

(34) UV_{DD} (Ground)

This is a 3.3 V positive power supply pin for the USB.

(35) UDP (USB Data +) ... I/O

This is a data I/O pin (+) of the USB.

(36) UDM (USB Data -) ... I/O

This is a data I/O pin (-) of the USB.

(37) IV_{DD} (Power supply)

These are 1.5 V positive power supply pins for each internal unit. All the IV_{DD} pins should be connected to a positive power supply.

(38) IV_{SS} (Ground)

These are ground pins. All the IV_{SS} pins should be grounded.

(39) A2 to A15 (Address) ... output

These are the address output pins of the lower 14 bits of the address bus's 26-bit address when the external memory is accessed.

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

These pins constitute a data bus for when the external memory is accessed. These are the lower 16-bit I/O bus pins of the 32-bit data.

The output changes in synchronization with the rise of the BUSCLK signal in the T1 state.

(41) TRCCLK (Trace clock) ... output

This is a trace clock output pin.

(42) TRCDATA0 to TRCDATA3 (Trace data output) ... output

These are trace data output (D0 to D3) pins.

(43) TRCEND (Trace end status output) ... output

This is a trace end status output pin.

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.

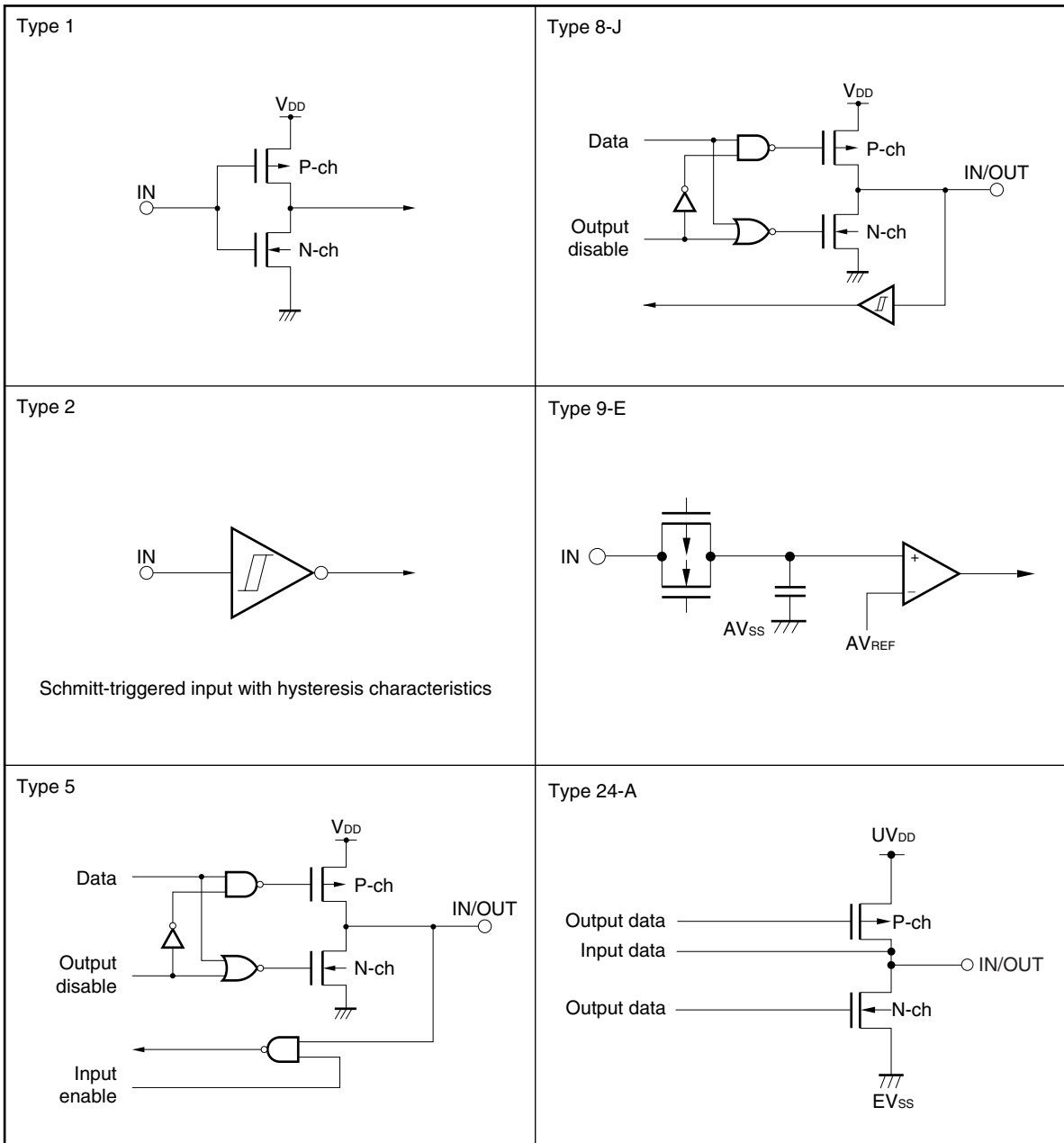
(1/3)

Pin Name	I/O Circuit Type	Recommended Connection
P10/ $\overline{\text{INTP10}}$ /UCLK	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
P11/ $\overline{\text{INTP11}}$ / $\overline{\text{SCK0}}$	8-J	
P12/SI0/RXD0		
P13/SO0/TXD0	5	
P20/NMI	2	Connect to V _{SS} directly.
P21/ $\overline{\text{INTP21}}$ /RXD1	8-J	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
P22/ $\overline{\text{INTP22}}$ /TXD1	5	
P23/ $\overline{\text{INTP23}}$ / $\overline{\text{SCK1}}$	8-J	
P24/ $\overline{\text{INTP24}}$ /SI1		
P25/ $\overline{\text{INTP25}}$ /SO1	5	
P50/ $\overline{\text{INTP50}}$ / $\overline{\text{DMARQ0}}$	5	Input: Independently connect to EV _{DD} via a resistor. Output: Leave open.
P51/ $\overline{\text{INTP51}}$ / $\overline{\text{DMAAK0}}$	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
P52/ $\overline{\text{INTP52}}$ / $\overline{\text{TC0}}$		
P53/ $\overline{\text{INTPC00}}$ / $\overline{\text{TIC0}}$ / $\overline{\text{DMARQ1}}$	5	Input: Independently connect to EV _{DD} via a resistor. Output: Leave open.
P54/ $\overline{\text{INTPC01}}$ / $\overline{\text{DMAAK1}}$	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
P55/ $\overline{\text{TOC0}}$ / $\overline{\text{TC1}}$		
P65/ $\overline{\text{INTP65}}$ / $\overline{\text{INTPC10}}$ / $\overline{\text{TIC1}}$		
P66/ $\overline{\text{INTP66}}$ / $\overline{\text{INTPC11}}$		
P67/ $\overline{\text{INTP67}}$ / $\overline{\text{TOC1}}$		
P72/ $\overline{\text{INTPC20}}$ / $\overline{\text{TIC2}}$ / $\overline{\text{DMARQ2}}$	5	Input: Independently connect to EV _{DD} via a resistor. Output: Leave open.
P73/ $\overline{\text{INTPC21}}$ / $\overline{\text{DMAAK2}}$	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
P74/ $\overline{\text{TOC2}}$ / $\overline{\text{TC2}}$		
P75/ $\overline{\text{INTPC30}}$ / $\overline{\text{TIC3}}$ / $\overline{\text{DMARQ3}}$	5	Input: Independently connect to EV _{DD} via a resistor. Output: Leave open.
P76/ $\overline{\text{INTPC31}}$ / $\overline{\text{DMAAK3}}$	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
P77/ $\overline{\text{TOC3}}$ / $\overline{\text{TC3}}$		
PAL0/ $\overline{\text{INTPL0}}$ /A0, PAL1/ $\overline{\text{INTPL1}}$ /A1		
PAH0/A16 to PAH9/A25		

Pin Name	I/O Circuit Type	Recommended Connection
PDH0/D16/ $\overline{\text{INTPD0}}$	5	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
PDH1/D17/ $\overline{\text{INTPD1}}$		
PDH2/D18/ $\overline{\text{INTPD2}}$ /TOC4		
PDH3/D19/ $\overline{\text{INTPD3}}$		
PDH4/D20/ $\overline{\text{INTPD4}}$		
PDH5/D21/ $\overline{\text{INTPD5}}$ /TOC5		
PDH6/D22/ $\overline{\text{INTPD6}}$ / $\overline{\text{INTP100}}$ /TCUD10		
PDH7/D23/ $\overline{\text{INTPD7}}$ / $\overline{\text{INTP101}}$ /TCLR10		
PDH8/D24/ $\overline{\text{INTPD8}}$ /TO10		
PDH9/D25/ $\overline{\text{INTPD9}}$ /TIUD10		
PDH10/D26/ $\overline{\text{INTPD10}}$ / $\overline{\text{INTP110}}$ /TCUD11		
PDH11/D27/ $\overline{\text{INTPD11}}$ / $\overline{\text{INTP111}}$ /TCLR11		
PDH12/D28/ $\overline{\text{INTPD12}}$ /TO11		
PDH13/D29/ $\overline{\text{INTPD13}}$ /TIUD11		
PDH14/D30/ $\overline{\text{INTPD14}}$ /PWM0		
PDH15/D31/ $\overline{\text{INTPD15}}$ /PWM1		
PCS0/ $\overline{\text{CS0}}$		
PCS1/ $\overline{\text{CS1}}$		
PCS2/ $\overline{\text{CS2}}$ / $\overline{\text{IOWR}}$		
PCS3/ $\overline{\text{CS3}}$		
PCS4/ $\overline{\text{CS4}}$		
PCS5/ $\overline{\text{CS5}}$ / $\overline{\text{IORD}}$		
PCS6/ $\overline{\text{CS6}}$		
PCS7/ $\overline{\text{CS7}}$		
PCT0/ $\overline{\text{LLWR}}$ / $\overline{\text{LLBE}}$ /LLDQM		
PCT1/ $\overline{\text{LUWR}}$ / $\overline{\text{LUBE}}$ /LUDQM		
PCT2/ $\overline{\text{ULWR}}$ / $\overline{\text{ULBE}}$ /ULDQM		
PCT3/ $\overline{\text{UUWR}}$ / $\overline{\text{UUBE}}$ /UUDQM		
PCT4/ $\overline{\text{RD}}$		
PCT5/ $\overline{\text{WE}}$ / $\overline{\text{WR}}$		
PCT7/ $\overline{\text{BCYST}}$		
PCM0/ $\overline{\text{WAIT}}$	5	Input: Independently connect to EV_{DD} via a resistor. Output: Leave open.
PCM1	8-J	Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open.
PCM2/ $\overline{\text{HLDAK}}$	5	

Pin Name	I/O Circuit Type	Recommended Connection
PCM3/ $\overline{\text{HLDRQ}}$	5	Input: Independently connect to EV _{DD} via a resistor. Output: Leave open.
PCM4/ $\overline{\text{REFRQ}}$	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
PCM5/ $\overline{\text{ADTRG}}/\overline{\text{SELFREF}}$	5	Input: Independently connect to EV _{DD} via a resistor. Output: Leave open.
PCD0/ $\overline{\text{SDCKE}}$	5	Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open.
PCD1/ $\overline{\text{BUSCLK}}$		
PCD2/ $\overline{\text{SDCAS}}$		
PCD3/ $\overline{\text{SDRAS}}$		
A2 to A15		
D0 to D15		
AV _{DD}	–	Connect to EV _{DD} .
AV _{SS}	–	Connect to EV _{SS} .
DCK	1	Connect to EV _{DD} via a resistor.
DDI		
DMS		
DDO	1	Leave open.
$\overline{\text{DRST}}$	1	Connect to EV _{SS} .
TRCCLK	1	Leave open.
TRCDATA0 to TRCDATA3	1	
TRCEND	1	
UV _{DD}	–	Connect to EV _{DD} .
UDM	24-A	Connect to EV _{SS} .
UDP		
ANI0 to ANI7	9-E	
AV _{REFM}	–	
AV _{REFP}	–	Connect to EV _{DD} .

2.5 Pin I/O Circuits



CHAPTER 3 CPU FUNCTION

The CPU of the V850E/ME2 is based on RISC architecture and executes almost all the instructions in one clock cycle using 5-stage pipeline control.

3.1 Features

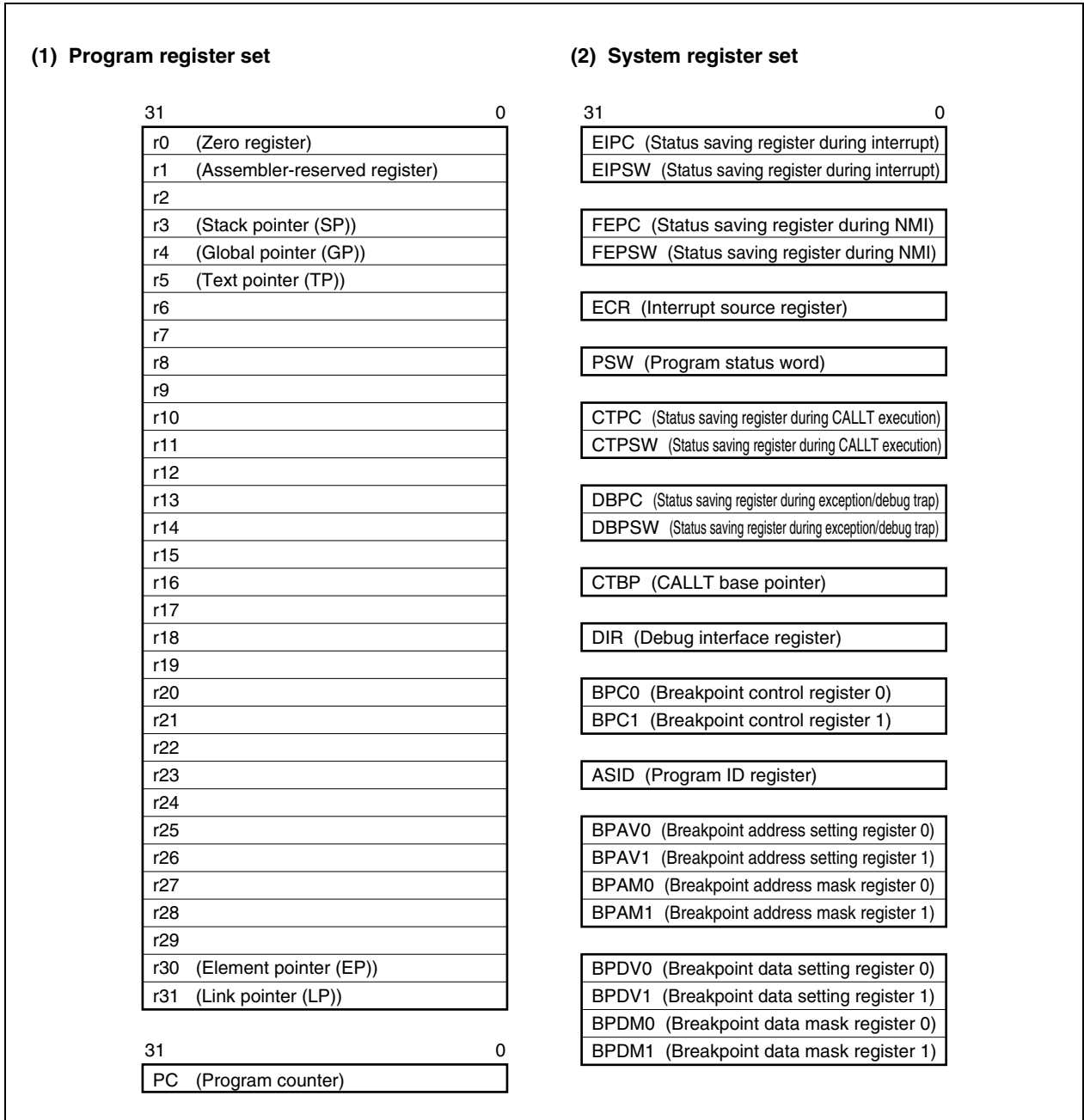
- ★
 - Minimum instruction execution time: 10 ns/7.5 ns/6.7 ns (@100 MHz/133 MHz/150 MHz internal operation)
 - Memory space Program space: 64 MB linear
 Data space: 4 GB linear
 - Thirty-two 32-bit general-purpose registers
 - Internal 32-bit architecture
 - Five-stage pipeline control
 - Multiply/divide instructions
 - Saturated operation instructions
 - One-clock 32-bit shift instruction
 - Load/store instruction with long/short format
 - Four types of bit manipulation instructions
 - SET1
 - CLR1
 - NOT1
 - TST1

3.2 CPU Register Set

The registers of the V850E/ME2 can be classified into two categories: a general-purpose program register set and a dedicated system register set. All the registers have a 32-bit width.

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

Figure 3-1. CPU Register Set



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.

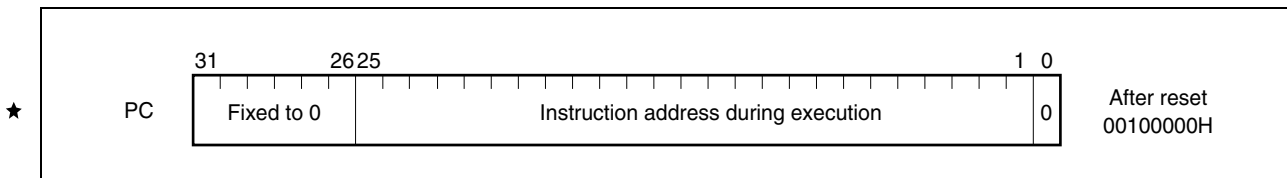
Table 3-1. Program Registers

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

(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

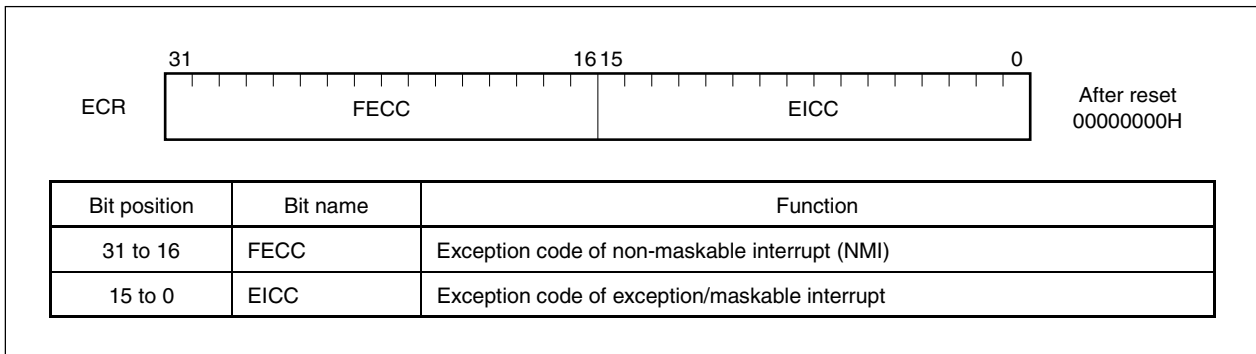
No.	System Register Name	Operand Specification	
		LDSR Instruction	STSR Instruction
0	Status saving register during interrupt (EIPC) ^{Note 1}	○	○
1	Status saving register during interrupt (EIPSW) ^{Note 1}	○	○
2	Status saving register during NMI (FEPC)	○	○
3	Status saving register during NMI (FEPSW)	○	○
4	Interrupt source register (ECR)	×	○
5	Program status word (PSW)	○	○
6 to 15	Reserved 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	Debug interface register (DIR)	○ ^{Note 3}	○
22	Breakpoint control register 0, 1 (BPC0, BPC1)	○ ^{Note 3}	○ ^{Note 3}
23	Program ID register (ASID)	○	○
24	Breakpoint address setting register 0, 1 (BPAV0, BPAV1)	○ ^{Note 3}	○ ^{Note 3}
25	Breakpoint address mask register 0, 1 (BPAM0, BPAM1)	○ ^{Note 3}	○ ^{Note 3}
26	Breakpoint data setting register 0, 1 (BPDV0, BPDV1)	○ ^{Note 3}	○ ^{Note 3}
27	Breakpoint data mask register 0, 1 (BPDM0, BPDM1)	○ ^{Note 3}	○ ^{Note 3}
28 to 31	Reserved for future function expansion (operations that access these register numbers cannot be guaranteed).	×	×

- Notes 1.** Because these registers have only one set, to enable multiple interrupts, it is necessary to save these registers by program.
- 2.** These registers can be accessed only in the debug mode. If they are accessed in the user mode, the result is undefined.
- 3.** Accessing these registers in a mode other than the debug mode is prohibited.

Caution Even if bit 0 of EIPC, FEPC, or CTPC is set to 1 by the LDSR instruction, bit 0 will be ignored when the program is 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, and CTPC, use an even value (bit 0 = 0) unless there is a special reason not to.

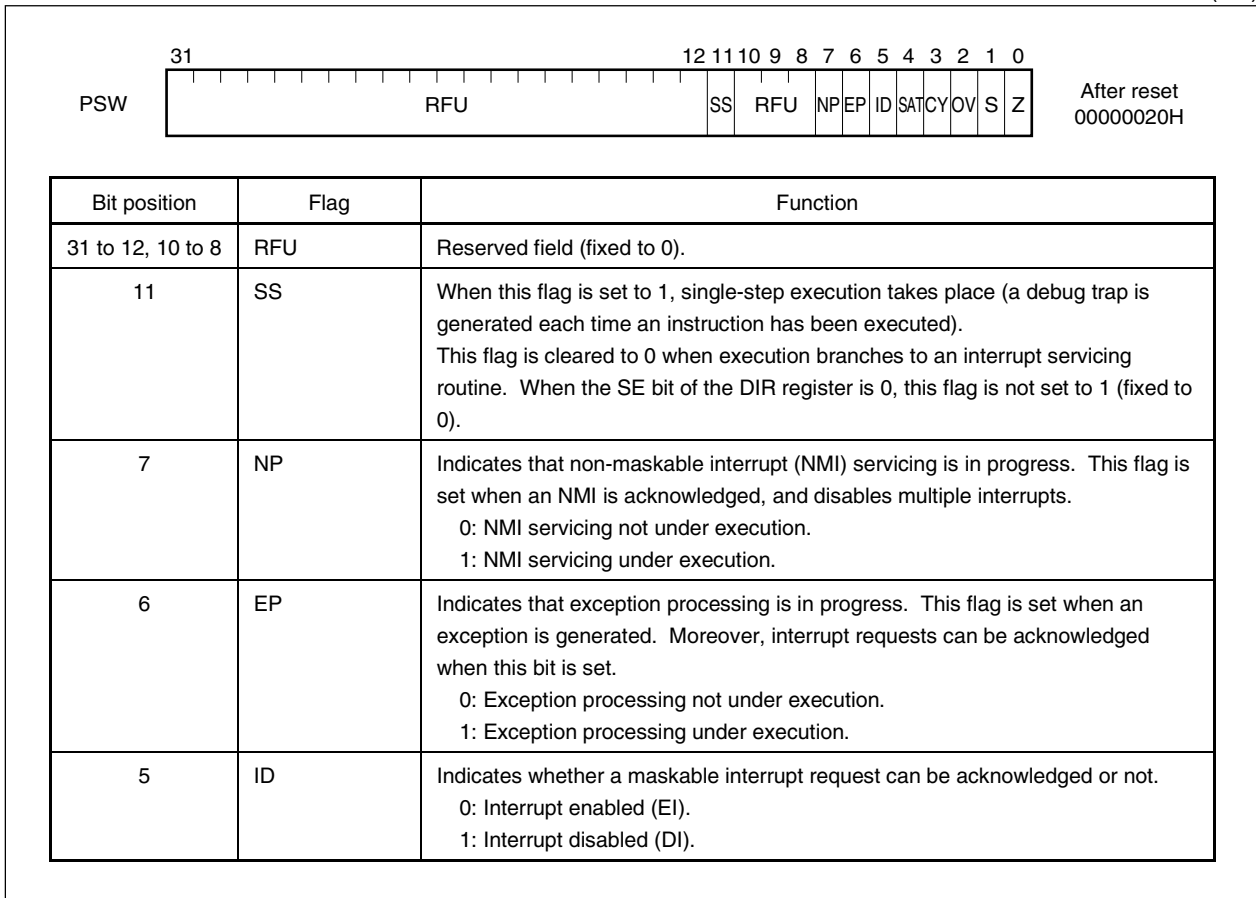
Remark ○: Access allowed
 ×: Access prohibited

(1) Interrupt source register (ECR)



(2) Program status word (PSW)

(1/2)



Bit position	Flag	Function
4	SAT ^{Note}	Indicates that the operation result of a saturated operation processing instruction is saturated due to overflow. Due to 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 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 a carry or borrow occurs as the result of an operation (if a 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 an overflow occurs during operation (if an 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 an 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 an 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)	Retains the value before operation	0	0	Operation result itself
Negative (not exceeding the maximum)			1	

3.3 Operating Modes

3.3.1 Operating modes

The V850E/ME2 has the following operating modes. Mode specification is carried out using the MODE0 and MODE1 pins.

(1) Normal operation mode

32-bit mode, 16-bit mode

After system reset is cleared, each pin related to the bus interface enters the control mode, the SRAM cycle branches to 0100000H (reset entry address) of the external device (memory), and instruction processing starts.

In the 32-bit mode, the bus interface functions as a 32-bit data bus; it functions as a 16-bit data bus in the 16-bit mode.

Caution Be sure to allocate external memory to address 0100000H for correct operation.

3.3.2 Operating mode specification

The operating mode is specified according to the status of the MODE0 and MODE1 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.

MODE1	MODE0	Operating Mode		Remarks
L	L	Normal operation mode	32-bit mode	32-bit data bus
L	H		16-bit mode	16-bit data bus
Other than above		Setting prohibited		

Remark L: Low-level input
H: High-level input
X: don't care

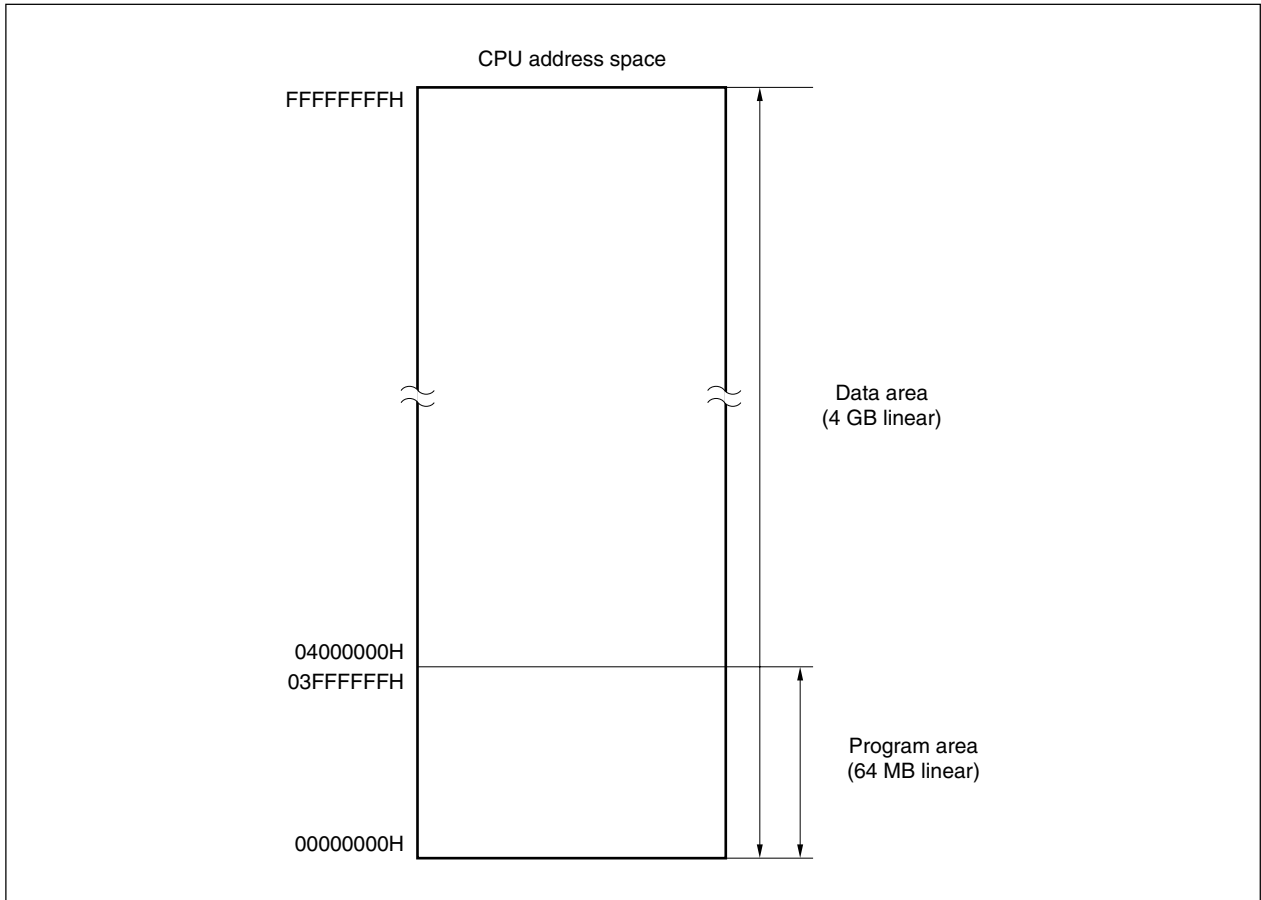
3.4 Address Space

3.4.1 CPU address space

The CPU of the V850E/ME2 is of 32-bit architecture and supports up to 4 GB of linear address space (data space) during operand addressing (data access). Also, in instruction address addressing, a maximum of 64 MB of linear address space (program space) is supported.

Figure 3-2 shows the CPU address space.

Figure 3-2. CPU Address Space

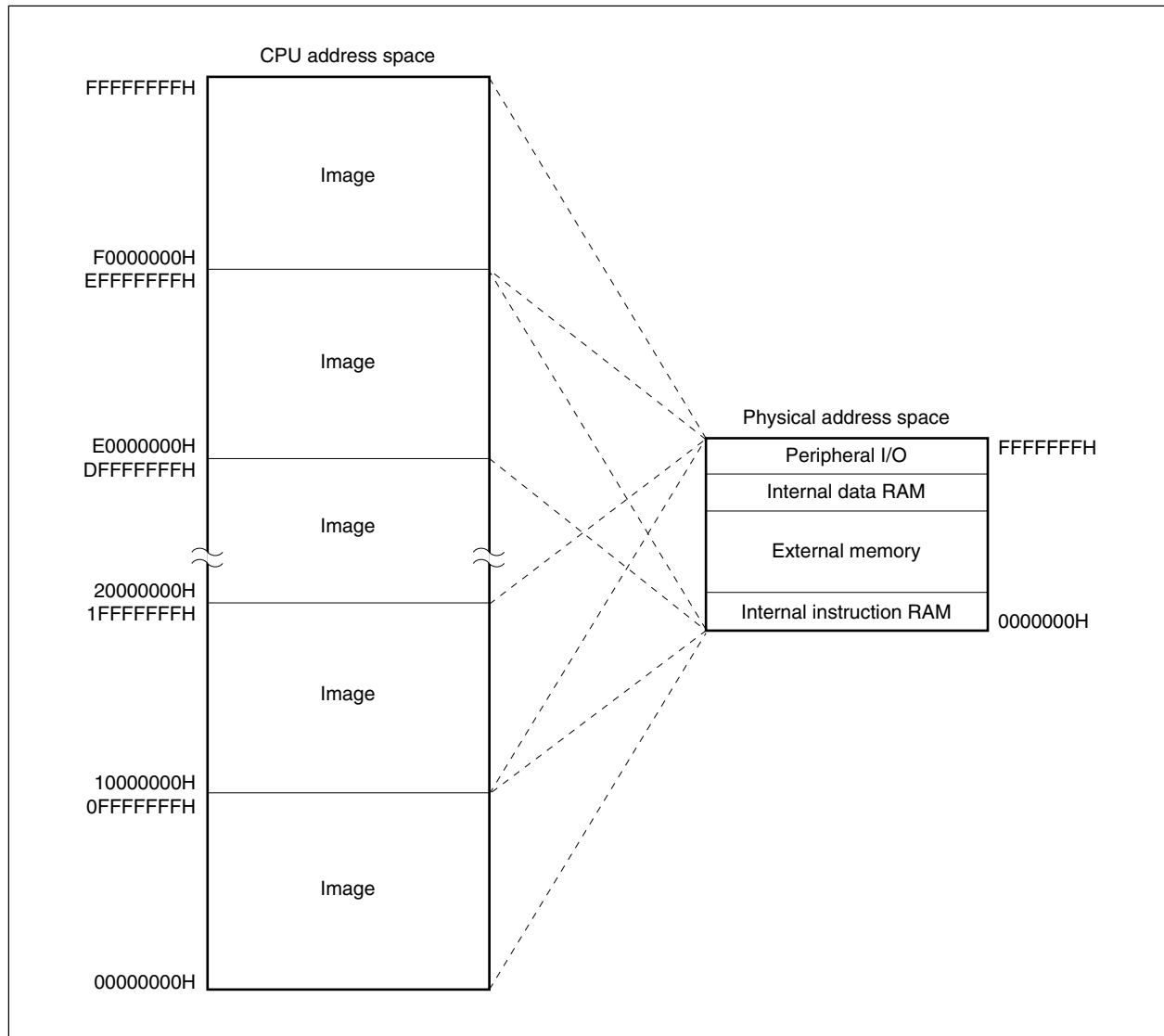


3.4.2 Image

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 3-3 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.

Figure 3-3. Images on Address Space



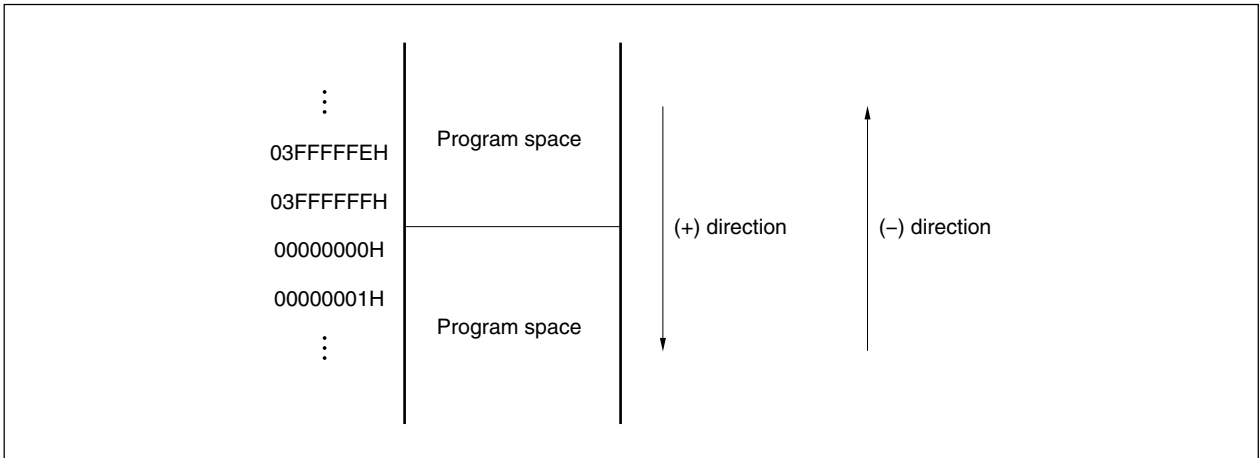
3.4.3 Wrap-around of CPU address space

(1) Program space

Of the 32 bits of the PC (program counter), the higher 6 bits are fixed 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 a 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.

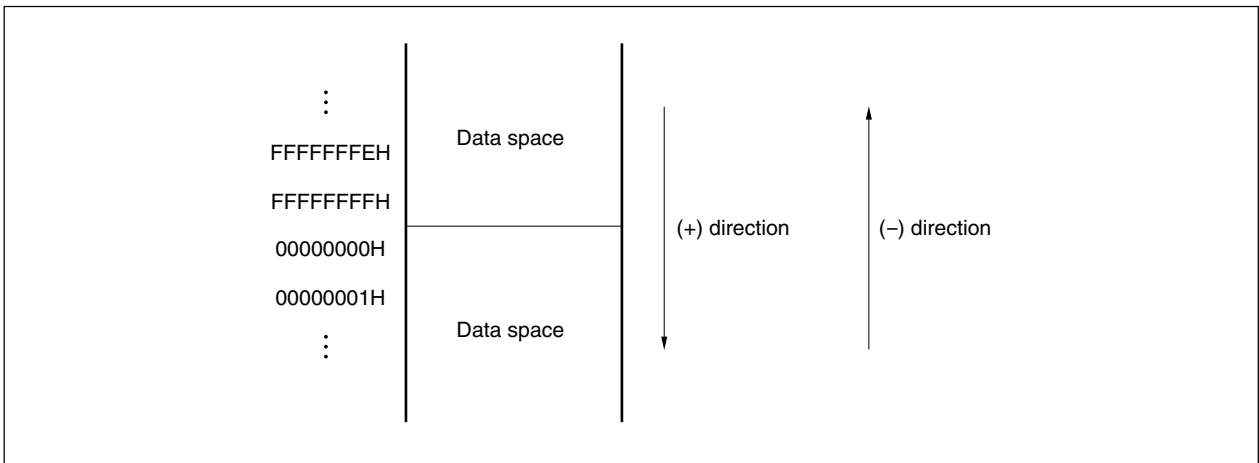
Caution The 32 KB area of 03FFF800H to 03FFFFFFH can be seen as an image of 0FFF800H to 0FFFFFFFH. Instructions cannot be fetched from this area because it is an on-chip peripheral I/O area, internal data RAM area, or access-prohibited area. Therefore, do not execute any branch address calculation in which the result will reside in any part of this area.



(2) Data space

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

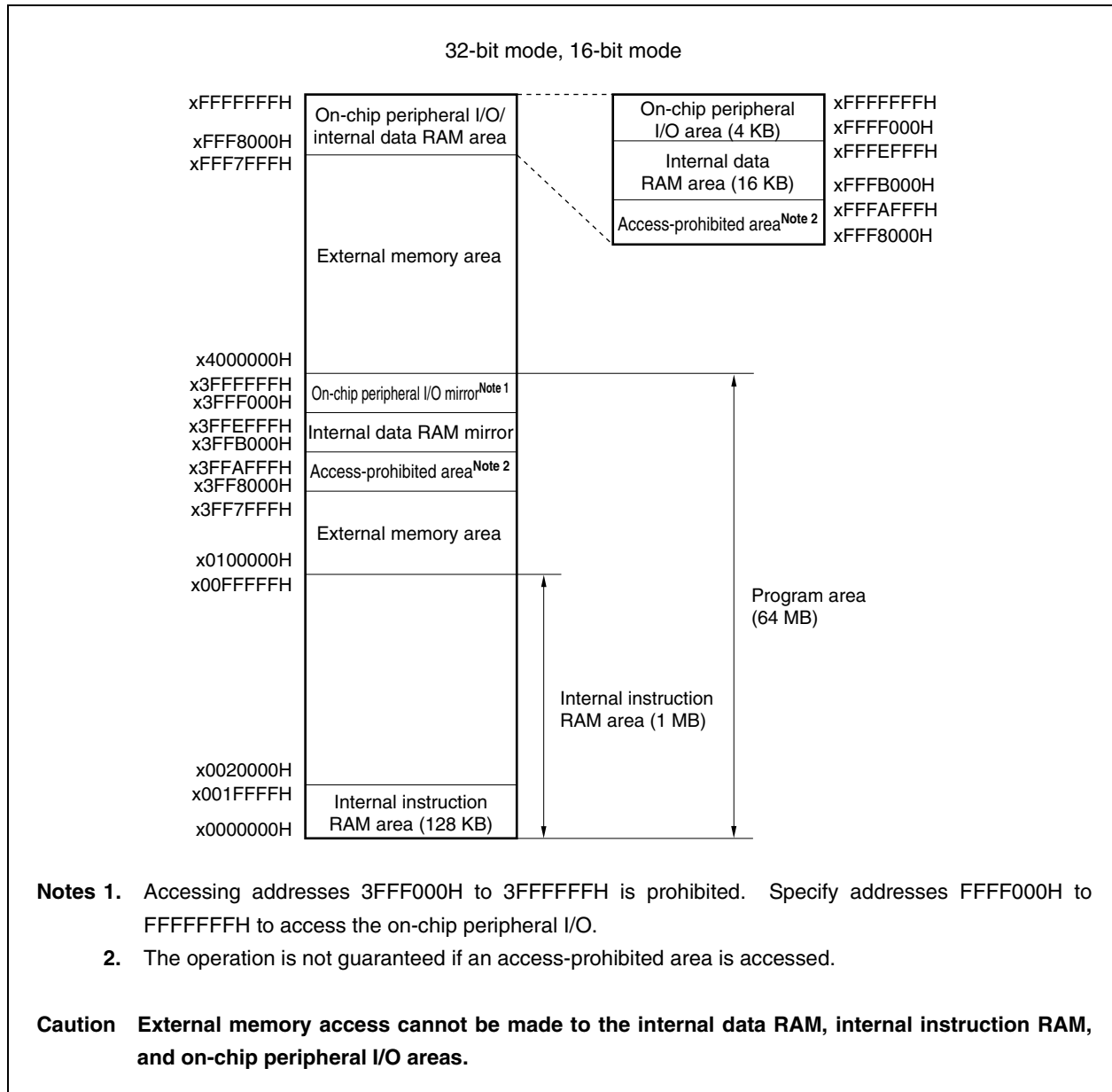
Therefore, the lower-limit address of the program space, address 00000000H, and the upper-limit address FFFFFFFFH are contiguous addresses, and the data space is wrapped around at the boundary of these addresses.



3.4.4 Memory map

Areas are reserved in the V850E/ME2 as shown in Figure 3-4.

Figure 3-4. Memory Map



3.4.5 Area

(1) Internal instruction RAM area

(a) Memory map

1 MB of internal instruction RAM area, addresses 00000H to FFFFFH, is reserved. 128 KB are provided at addresses 000000H to 01FFFFH as physical instruction RAM.

Caution External memory access cannot be made to addresses 020000H to FFFFFH. If this area is accessed, the address bus (A0 to A25) outputs a low level, the data bus (D0 to D31) goes into a high-impedance state without outputting anything, and the external bus control signal becomes inactive.

(b) Interrupt/exception table

The V850E/ME2 increases the interrupt response speed by assigning handler addresses corresponding to interrupts/exceptions.

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

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

Start Address of Interrupt/Exception Table	Interrupt/Exception Source	Start Address of Interrupt/Exception Table	Interrupt/Exception Source
00100000H	RESET	00000170H	INTPD2
00000010H	NMI0	00000180H	INTPD3
00000040H	TRAP0n (n = 0 to F)	00000190H	INTPD4
00000050H	TRAP1n (n = 0 to F)	000001A0H	INTPD5
00000060H	ILGOP/DBG0	000001B0H	INTPD6
00000080H	INTP10	000001C0H	INTPD7
00000090H	INTP11	000001D0H	INTPD8
000000A0H	INTP21	000001E0H	INTPD9
000000B0H	INTP22	000001F0H	INTPD10
000000C0H	INTP23	00000200H	INTPD11
000000D0H	INTP24	00000210H	INTPD12
000000E0H	INTP25	00000220H	INTPD13
000000F0H	INTP50	00000230H	INTPD14
00000100H	INTP51	00000240H	INTPD15
00000110H	INTP52	00000250H	INTPL0
00000120H	INTP65	00000260H	INTPL1
00000130H	INTP66	00000270H	INTOVC0
00000140H	INTP67	00000280H	INTOVC1
00000150H	INTPD0	00000290H	INTOVC2
00000160H	INTPD1	000002A0H	INTOVC3

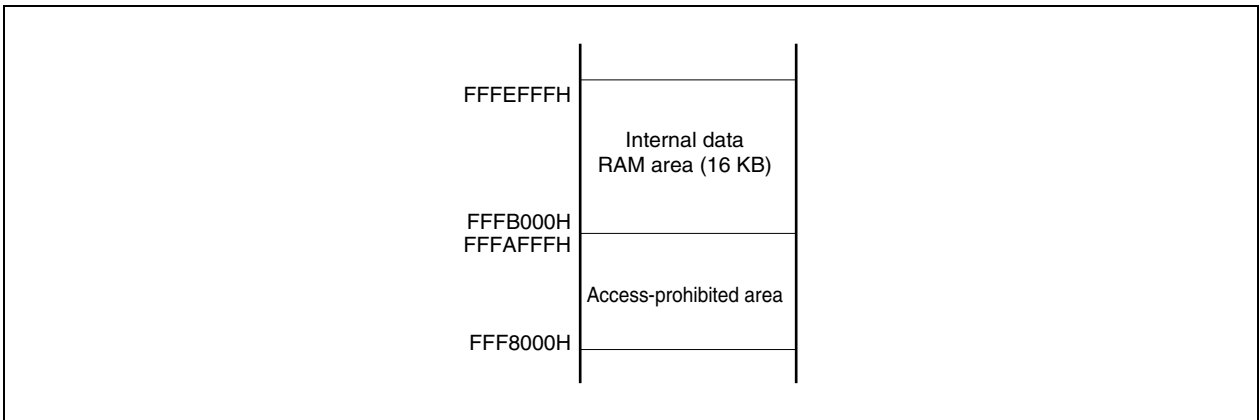
Table 3-3. Interrupt/Exception Table (2/2)

Start Address of Interrupt/Exception Table	Interrupt/Exception Source	Start Address of Interrupt/Exception Table	Interrupt/Exception Source
000002B0H	INTOVC4	00000470H	INTOV11
000002C0H	INTOVC5	00000480H	INTUD11
000002D0H	INTPC00/INTCCC00	00000490H	INTDMA0
000002E0H	INTPC01/INTCCC01	000004A0H	INTDMA1
000002F0H	INTPC10/INTCCC10	000004B0H	INTDMA2
00000300H	INTPC11/INTCCC11	000004C0H	INTDMA3
00000310H	INTPC20/INTCCC20	000004D0H	INTCSI30
00000320H	INTPC21/INTCCC21	000004E0H	INTCOVF30
00000330H	INTPC30/INTCCC30	000004F0H	INTCSI31
00000340H	INTPC31/INTCCC31	00000500H	INTCOVF31
00000350H	INTCCC40	00000510H	UBTIRE0
00000360H	INTCCC41	00000520H	UBTIR0
00000370H	INTCCC50	00000530H	UBTIT0
00000380H	INTCCC51	00000540H	UBTIF0
00000390H	INTCMD0	00000550H	UBTITO0
000003A0H	INTCMD1	00000560H	UBTIRE1
000003B0H	INTCMD2	00000570H	UBTIR1
000003C0H	INTCMD3	00000580H	UBTIT1
000003D0H	INTCC100	00000590H	UBTIF1
000003E0H	INTCC101	000005A0H	UBTITO1
000003F0H	INTCM100	000005B0H	INTAD
00000400H	INTCM101	000005C0H	INTUSB0B
00000410H	INTOV10	000005D0H	INTUSB1B
00000420H	INTUD10	000005E0H	INTUSB2B
00000430H	INTCC110	000005F0H	USBSP2B
00000440H	INTCC111	00000600H	USBSP4B
00000450H	INTCM110	00000610H	INTRSUM
00000460H	INTCM111		

(2) Internal data RAM area

The 16 KB area of addresses FFFB000H to FFFEFFFFH is provided as the internal data RAM area. The 16 KB area of 3FFB000H to 3FFEFFFFH can be seen as an image of FFFB000H to FFFEFFFFH.

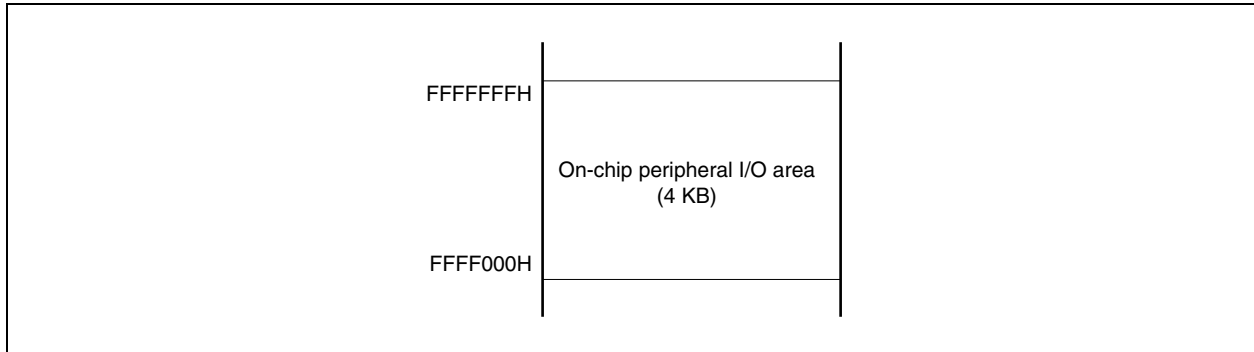
- Cautions**
1. External memory access cannot be made to addresses xFFF8000H to xFFFAFFFH and x3FF8000H to x3FFAFFFH. The operation is not guaranteed if external memory access is performed.
 2. Do not execute the program in the internal data RAM area.



(3) On-chip peripheral I/O area

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

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



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

Cautions 1. For registers in which byte access is possible, if halfword access is executed, the higher 8 bits become undefined during a read operation, and the lower 8 bits of data are written to the register during a write operation.

Do not access an 8-bit register in halfword units.

2. 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.

(4) External memory area

256 MB are available for external memory area. The lower 64 MB can be used as program/data area and the higher 192 MB as data area. The external memory area is addresses x0100000H to xFDFFFFFFH.

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

Note that the internal instruction RAM, internal data RAM, and on-chip peripheral I/O areas cannot be used as external memory areas.

3.4.6 Recommended use of address space

The architecture of the V850E/ME2 requires that a register that serves as a pointer be secured for address generation in operand data accessing of data space. Operand data access from instruction can be directly executed at the address in this pointer register ± 32 KB. However, because the general-purpose registers that can be used as a pointer register are limited, 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.

In connection with the memory map of the V850E/ME2, the following usage is recommended to enhance the efficiency of pointer operation.

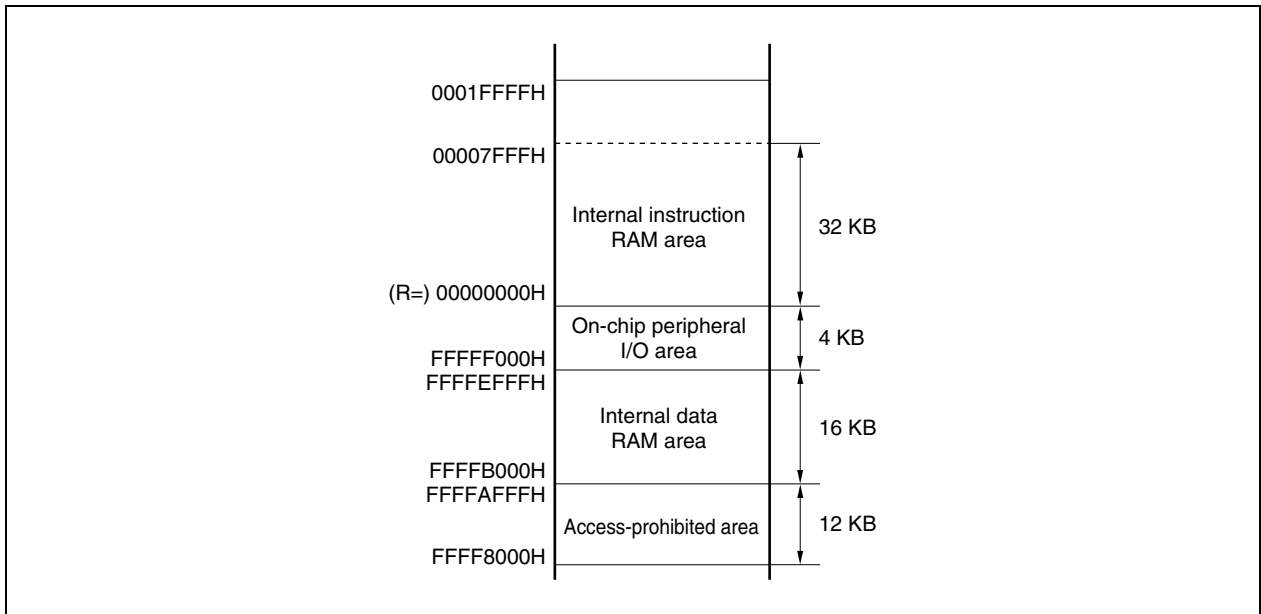
(1) Program space

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. Therefore, a contiguous 64 MB space, starting from address 00000000H, unconditionally corresponds to the memory map of the program space.

(2) Data space

To efficiently operate the resources using wrap-around of the data space, a consecutive 16 MB are used as a data space at each of the 4 GB CPU address spaces of 00000000H to 00FFFFFFFH and FF000000H to FFFFFFFFH. With the V850E/ME2, 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.

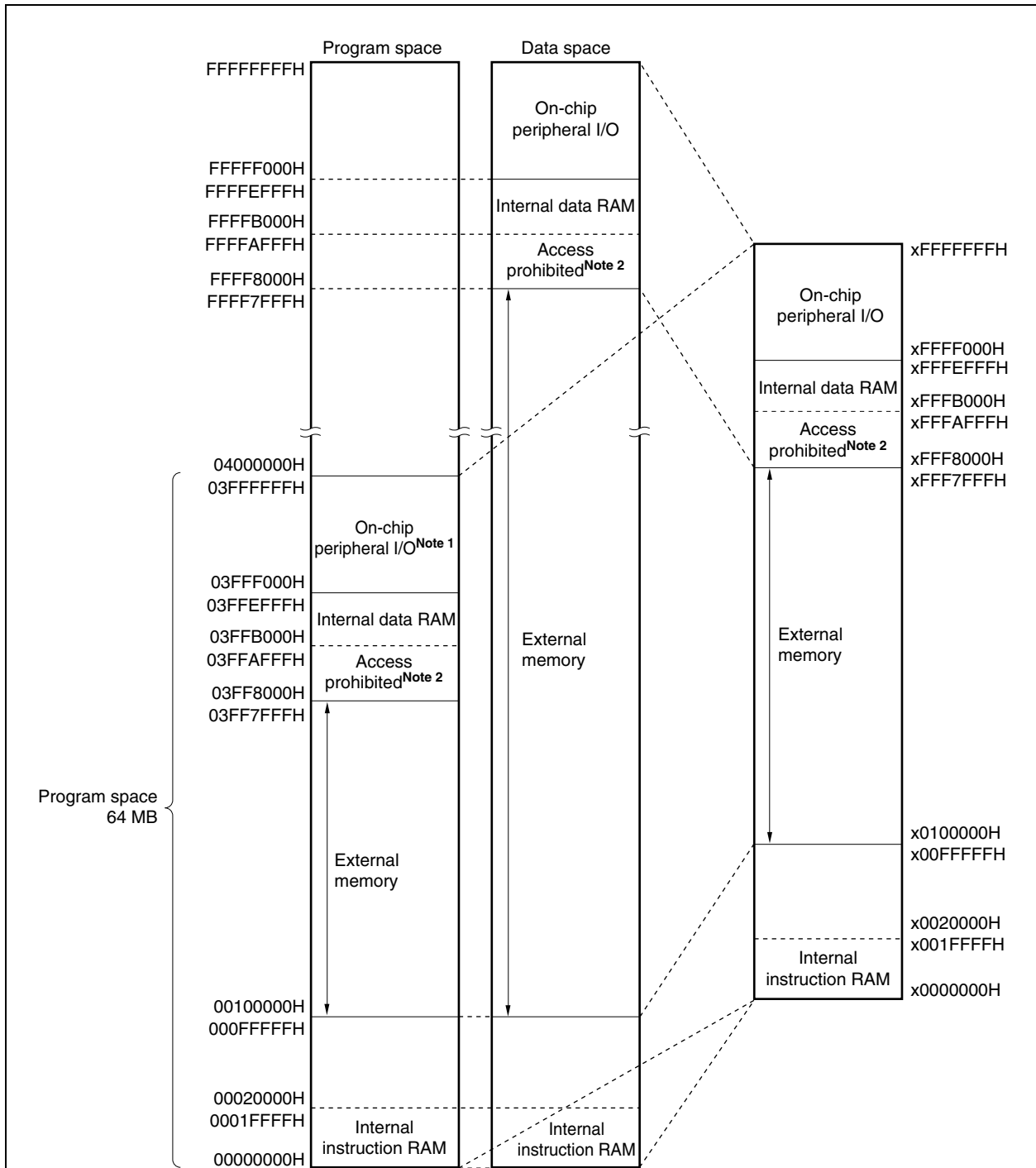
Example Application of wrap-around is as shown below.



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 disp16.

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

Figure 3-5. Recommended Memory Map



Notes 1. This area is access-prohibited. To access the on-chip peripheral I/O, specify addresses FFFF000H to FFFFFFFH.

2. The operation is not guaranteed if an access-prohibited area is accessed.

Remark The arrows indicate the recommended area.

3.4.7 Peripheral I/O registers

(1/26)

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF000H	Port AL	PAL	R/W			√	Undefined
FFFFF000H	Port ALL	PALL	R/W	√	√		Undefined
FFFFF001H	Port ALH	PALH	R	√	√		00H
FFFFF002H	Port AH	PAH	R/W			√	Undefined
FFFFF002H	Port AHL	PAHL	R/W	√	√		Undefined
FFFFF003H	Port AHH	PAHH	R/W	√	√		Undefined
FFFFF006H	Port DH	PDH	R/W			√	Undefined
FFFFF006H	Port DHL	PDHL	R/W	√	√		Undefined
FFFFF007H	Port DHH	PDHH	R/W	√	√		Undefined
FFFFF008H	Port CS	PCS	R/W	√	√		Undefined
FFFFF00AH	Port CT	PCT	R/W	√	√		Undefined
FFFFF00CH	Port CM	PCM	R/W	√	√		Undefined
FFFFF00EH	Port CD	PCD	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	√	√		FFH
FFFFF022H	Port AH mode register	PMAH	R/W			√	FFFFH
FFFFF022H	Port AH mode register L	PMAHL	R/W	√	√		FFH
FFFFF023H	Port AH mode register H	PMAHH	R/W	√	√		FFH
FFFFF026H	Port DH mode register	PMDH	R/W			√	FFFFH
FFFFF026H	Port DH mode register L	PMDHL	R/W	√	√		FFH
FFFFF027H	Port DH mode register H	PMDHH	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
FFFFF040H	Port AL mode control register	PMCAL	R/W			√	0002H
FFFFF040H	Port AL mode control register L	PMCALL	R/W	√	√		02H
FFFFF041H	Port AL mode control register H	PMCALH	R	√	√		00H
FFFFF042H	Port AH mode control register	PMCAH	R/W			√	03FFH
FFFFF042H	Port AH mode control register L	PMCAHL	R/W	√	√		FFH
FFFFF043H	Port AH mode control register H	PMCAHH	R/W	√	√		03H
FFFFF046H	Port DH mode control register	PMCDH	R/W			√	0000H
FFFFF046H	Port DH mode control register L	PMCDHL	R/W	√	√		00H
FFFFF047H	Port DH mode control register H	PMCDHH	R/W	√	√		00H
FFFFF048H	Port CS mode control register	PMCCS	R/W	√	√		FFH
FFFFF049H	Port CS function control register	PFCCS	R/W	√	√		00H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF04AH	Port CT mode control register	PMCCT	R/W	√	√		BFH
FFFFFF04BH	Port CT function control register	PFCCT	R/W	√	√		00H
FFFFFF04CH	Port CM mode control register	PMCCM	R/W	√	√		3DH
FFFFFF04DH	Port CM function control register	PFCCM	R/W	√	√		00H
FFFFFF04EH	Port CD mode control register	PMCCD	R/W	√	√		0FH
FFFFFF056H	Port DH function control register	PFCDH	R/W			√	0000H
FFFFFF056H	Port DH function control register L	PFCDHL	R/W	√	√		00H
FFFFFF057H	Port DH function control register H	PFCDHH	R/W	√	√		00H
★ FFFFFFF058H	Port AL function control register L	PFCALL	R/W	√	√		03H
FFFFFF060H	Chip area select control register 0	CSC0	R/W			√	2C11H
FFFFFF062H	Chip area select control register 1	CSC1	R/W			√	2C11H
FFFFFF068H	Endian configuration register	BEC	R/W			√	0000H
FFFFFF06AH	Cache configuration register	BHC	R/W			√	0000H
FFFFFF06EH	System wait control register	VSWC	R/W	√	√		77H
FFFFFF070H	Instruction cache control register	ICC	R/W			√	0003H ^{Note 1}
FFFFFF070H	Instruction cache control register L	ICCL	R/W	√	√		03H ^{Note 2}
FFFFFF071H	Instruction cache control register H	ICCH	R/W	√	√		00H
FFFFFF074H	Instruction cache data configuration register	ICD	R/W			√	Undefined
FFFFFF080H	DMA source address register 0L	DSA0L	R/W			√	Undefined
FFFFFF082H	DMA source address register 0H	DSA0H	R/W			√	Undefined
FFFFFF084H	DMA destination address register 0L	DDA0L	R/W			√	Undefined
FFFFFF086H	DMA destination address register 0H	DDA0H	R/W			√	Undefined
FFFFFF088H	DMA source address register 1L	DSA1L	R/W			√	Undefined
FFFFFF08AH	DMA source address register 1H	DSA1H	R/W			√	Undefined
FFFFFF08CH	DMA destination address register 1L	DDA1L	R/W			√	Undefined
FFFFFF08EH	DMA destination address register 1H	DDA1H	R/W			√	Undefined
FFFFFF090H	DMA source address register 2L	DSA2L	R/W			√	Undefined
FFFFFF092H	DMA source address register 2H	DSA2H	R/W			√	Undefined
FFFFFF094H	DMA destination address register 2L	DDA2L	R/W			√	Undefined
FFFFFF096H	DMA destination address register 2H	DDA2H	R/W			√	Undefined
FFFFFF098H	DMA source address register 3L	DSA3L	R/W			√	Undefined
FFFFFF09AH	DMA source address register 3H	DSA3H	R/W			√	Undefined
FFFFFF09CH	DMA destination address register 3L	DDA3L	R/W			√	Undefined

Notes 1. This register is set to 0003H and the tag is automatically initialized when the reset signal becomes active. When initialization of the tag is completed, the register is cleared to 0000H.

2. This register is set to 03H and the tag is automatically initialized when the reset signal becomes active. When initialization of the tag is completed, the register is cleared to 00H.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF09EH	DMA destination address register 3H	DDA3H	R/W			√	Undefined
FFFFF0C0H	DMA transfer count register 0	DBC0	R/W			√	Undefined
FFFFF0C2H	DMA transfer count register 1	DBC1	R/W			√	Undefined
FFFFF0C4H	DMA transfer count register 2	DBC2	R/W			√	Undefined
FFFFF0C6H	DMA transfer 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
FFFFF0D6H	DMA addressing control register 3	DADC3	R/W			√	0000H
FFFFF0E0H	DMA channel control register 0	DCHC0	R/W	√	√		00H
FFFFF0E2H	DMA channel control register 1	DCHC1	R/W	√	√		00H
FFFFF0E4H	DMA channel control register 2	DCHC2	R/W	√	√		00H
FFFFF0E6H	DMA channel control register 3	DCHC3	R/W	√	√		00H
FFFFF100H	Interrupt mask register 0	IMR0	R/W			√	FFFFH
FFFFF100H	Interrupt mask register 0L	IMR0L	R/W	√	√		FFH
FFFFF101H	Interrupt mask register 0H	IMR0H	R/W	√	√		FFH
FFFFF102H	Interrupt mask register 1	IMR1	R/W			√	FFFFH
FFFFF102H	Interrupt mask register 1L	IMR1L	R/W	√	√		FFH
FFFFF103H	Interrupt mask register 1H	IMR1H	R/W	√	√		FFH
FFFFF104H	Interrupt mask register 2	IMR2	R/W			√	FFFFH
FFFFF104H	Interrupt mask register 2L	IMR2L	R/W	√	√		FFH
FFFFF105H	Interrupt mask register 2H	IMR2H	R/W	√	√		FFH
FFFFF106H	Interrupt mask register 3	IMR3	R/W			√	FFFFH
FFFFF106H	Interrupt mask register 3L	IMR3L	R/W	√	√		FFH
FFFFF107H	Interrupt mask register 3H	IMR3H	R/W	√	√		FFH
FFFFF108H	Interrupt mask register 4	IMR4	R/W			√	FFFFH
FFFFF108H	Interrupt mask register 4L	IMR4L	R/W	√	√		FFH
FFFFF109H	Interrupt mask register 4H	IMR4H	R/W	√	√		FFH
FFFFF10AH	Interrupt mask register 5	IMR5	R/W			√	FFFFH
FFFFF10AH	Interrupt mask register 5L	IMR5L	R/W	√	√		FFH
FFFFF10BH	Interrupt mask register 5H	IMR5H	R/W	√	√		FFH
FFFFF110H	Interrupt control register 0	P1IC0	R/W	√	√		47H
FFFFF112H	Interrupt control register 1	P1IC1	R/W	√	√		47H
FFFFF114H	Interrupt control register 2	P2IC1	R/W	√	√		47H
FFFFF116H	Interrupt control register 3	P2IC2	R/W	√	√		47H
FFFFF118H	Interrupt control register 4	P2IC3	R/W	√	√		47H
FFFFF11AH	Interrupt control register 5	P2IC4	R/W	√	√		47H
FFFFF11CH	Interrupt control register 6	P2IC5	R/W	√	√		47H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF11EH	Interrupt control register 7	P5IC0	R/W	√	√		47H
FFFFFF120H	Interrupt control register 8	P5IC1	R/W	√	√		47H
FFFFFF122H	Interrupt control register 9	P5IC2	R/W	√	√		47H
FFFFFF124H	Interrupt control register 10	P6IC5	R/W	√	√		47H
FFFFFF126H	Interrupt control register 11	P6IC6	R/W	√	√		47H
FFFFFF128H	Interrupt control register 12	P6IC7	R/W	√	√		47H
FFFFFF12AH	Interrupt control register 13	PDIC0	R/W	√	√		47H
FFFFFF12CH	Interrupt control register 14	PDIC1	R/W	√	√		47H
FFFFFF12EH	Interrupt control register 15	PDIC2	R/W	√	√		47H
FFFFFF130H	Interrupt control register 16	PDIC3	R/W	√	√		47H
FFFFFF132H	Interrupt control register 17	PDIC4	R/W	√	√		47H
FFFFFF134H	Interrupt control register 18	PDIC5	R/W	√	√		47H
FFFFFF136H	Interrupt control register 19	PDIC6	R/W	√	√		47H
FFFFFF138H	Interrupt control register 20	PDIC7	R/W	√	√		47H
FFFFFF13AH	Interrupt control register 21	PDIC8	R/W	√	√		47H
FFFFFF13CH	Interrupt control register 22	PDIC9	R/W	√	√		47H
FFFFFF13EH	Interrupt control register 23	PDIC10	R/W	√	√		47H
FFFFFF140H	Interrupt control register 24	PDIC11	R/W	√	√		47H
FFFFFF142H	Interrupt control register 25	PDIC12	R/W	√	√		47H
FFFFFF144H	Interrupt control register 26	PDIC13	R/W	√	√		47H
FFFFFF146H	Interrupt control register 27	PDIC14	R/W	√	√		47H
FFFFFF148H	Interrupt control register 28	PDIC15	R/W	√	√		47H
FFFFFF14AH	Interrupt control register 29	PLIC0	R/W	√	√		47H
FFFFFF14CH	Interrupt control register 30	PLIC1	R/W	√	√		47H
FFFFFF14EH	Interrupt control register 31	OVCIC0	R/W	√	√		47H
FFFFFF150H	Interrupt control register 32	OVCIC1	R/W	√	√		47H
FFFFFF152H	Interrupt control register 33	OVCIC2	R/W	√	√		47H
FFFFFF154H	Interrupt control register 34	OVCIC3	R/W	√	√		47H
FFFFFF156H	Interrupt control register 35	OVCIC4	R/W	√	√		47H
FFFFFF158H	Interrupt control register 36	OVCIC5	R/W	√	√		47H
FFFFFF15AH	Interrupt control register 37	CCC0IC0	R/W	√	√		47H
FFFFFF15CH	Interrupt control register 38	CCC0IC1	R/W	√	√		47H
FFFFFF15EH	Interrupt control register 39	CCC1IC0	R/W	√	√		47H
FFFFFF160H	Interrupt control register 40	CCC1IC1	R/W	√	√		47H
FFFFFF162H	Interrupt control register 41	CCC2IC0	R/W	√	√		47H
FFFFFF164H	Interrupt control register 42	CCC2IC1	R/W	√	√		47H
FFFFFF166H	Interrupt control register 43	CCC3IC0	R/W	√	√		47H
FFFFFF168H	Interrupt control register 44	CCC3IC1	R/W	√	√		47H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF16AH	Interrupt control register 45	CCC4IC0	R/W	√	√		47H
FFFFF16CH	Interrupt control register 46	CCC4IC1	R/W	√	√		47H
FFFFF16EH	Interrupt control register 47	CCC5IC0	R/W	√	√		47H
FFFFF170H	Interrupt control register 48	CCC5IC1	R/W	√	√		47H
FFFFF172H	Interrupt control register 49	CMDIC0	R/W	√	√		47H
FFFFF174H	Interrupt control register 50	CMDIC1	R/W	√	√		47H
FFFFF176H	Interrupt control register 51	CMDIC2	R/W	√	√		47H
FFFFF178H	Interrupt control register 52	CMDIC3	R/W	√	√		47H
FFFFF17AH	Interrupt control register 53	CC10IC0	R/W	√	√		47H
FFFFF17CH	Interrupt control register 54	CC10IC1	R/W	√	√		47H
FFFFF17EH	Interrupt control register 55	CM10IC0	R/W	√	√		47H
FFFFF180H	Interrupt control register 56	CM10IC1	R/W	√	√		47H
FFFFF182H	Interrupt control register 57	OV1IC0	R/W	√	√		47H
FFFFF184H	Interrupt control register 58	UD1IC0	R/W	√	√		47H
FFFFF186H	Interrupt control register 59	CC11IC0	R/W	√	√		47H
FFFFF188H	Interrupt control register 60	CC11IC1	R/W	√	√		47H
FFFFF18AH	Interrupt control register 61	CM11IC0	R/W	√	√		47H
FFFFF18CH	Interrupt control register 62	CM11IC1	R/W	√	√		47H
FFFFF18EH	Interrupt control register 63	OV1IC1	R/W	√	√		47H
FFFFF190H	Interrupt control register 64	UD1IC1	R/W	√	√		47H
FFFFF192H	Interrupt control register 65	DMAIC0	R/W	√	√		47H
FFFFF194H	Interrupt control register 66	DMAIC1	R/W	√	√		47H
FFFFF196H	Interrupt control register 67	DMAIC2	R/W	√	√		47H
FFFFF198H	Interrupt control register 68	DMAIC3	R/W	√	√		47H
FFFFF19AH	Interrupt control register 69	CSI3IC0	R/W	√	√		47H
FFFFF19CH	Interrupt control register 70	COVF3IC0	R/W	√	√		47H
FFFFF19EH	Interrupt control register 71	CSI3IC1	R/W	√	√		47H
FFFFF1A0H	Interrupt control register 72	COVF3IC1	R/W	√	√		47H
FFFFF1A2H	Interrupt control register 73	UREIC0	R/W	√	√		47H
FFFFF1A4H	Interrupt control register 74	URIC0	R/W	√	√		47H
FFFFF1A6H	Interrupt control register 75	UTIC0	R/W	√	√		47H
FFFFF1A8H	Interrupt control register 76	UIFIC0	R/W	√	√		47H
FFFFF1AAH	Interrupt control register 77	UTOIC0	R/W	√	√		47H
FFFFF1ACH	Interrupt control register 78	UREIC1	R/W	√	√		47H
FFFFF1AEH	Interrupt control register 79	URIC1	R/W	√	√		47H
FFFFF1B0H	Interrupt control register 80	UTIC1	R/W	√	√		47H
FFFFF1B2H	Interrupt control register 81	UIFIC1	R/W	√	√		47H
FFFFF1B4H	Interrupt control register 82	UTOIC1	R/W	√	√		47H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF1B6H	Interrupt control register 83	ADIC	R/W	√	√		47H
FFFFF1B8H	Interrupt control register 84	US0BIC	R/W	√	√		47H
FFFFF1BAH	Interrupt control register 85	US1BIC	R/W	√	√		47H
FFFFF1BCH	Interrupt control register 86	US2BIC	R/W	√	√		47H
FFFFF1BEH	Interrupt control register 87	USP2IC	R/W	√	√		47H
FFFFF1C0H	Interrupt control register 88	USP4IC	R/W	√	√		47H
FFFFF1C2H	Interrupt control register 89	RSUMIC	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	√	√		00H
FFFFF202H	A/D converter mode register 2	ADM2	R/W	√	√		00H
FFFFF210H	A/D conversion result register 0	ADCR0	R			√	Undefined
FFFFF211H	A/D conversion result register 0H	ADCR0H	R		√		Undefined
FFFFF212H	A/D conversion result register 1	ADCR1	R			√	Undefined
FFFFF213H	A/D conversion result register 1H	ADCR1H	R		√		Undefined
FFFFF214H	A/D conversion result register 2	ADCR2	R			√	Undefined
FFFFF215H	A/D conversion result register 2H	ADCR2H	R		√		Undefined
FFFFF216H	A/D conversion result register 3	ADCR3	R			√	Undefined
FFFFF217H	A/D conversion result register 3H	ADCR3H	R		√		Undefined
FFFFF218H	A/D conversion result register 4	ADCR4	R			√	Undefined
FFFFF219H	A/D conversion result register 4H	ADCR4H	R		√		Undefined
FFFFF21AH	A/D conversion result register 5	ADCR5	R			√	Undefined
FFFFF21BH	A/D conversion result register 5H	ADCR5H	R		√		Undefined
FFFFF21CH	A/D conversion result register 6	ADCR6	R			√	Undefined
FFFFF21DH	A/D conversion result register 6H	ADCR6H	R		√		Undefined
FFFFF21EH	A/D conversion result register 7	ADCR7	R			√	Undefined
FFFFF21FH	A/D conversion result register 7H	ADCR7H	R		√		Undefined
FFFFF220H	ADC trigger select register	ADTS	R/W		√		00H
FFFFF402H	Port 1	P1	R/W	√	√		Undefined
FFFFF404H	Port 2	P2	R/W	√	√		Undefined
FFFFF40AH	Port 5	P5	R/W	√	√		Undefined
FFFFF40CH	Port 6	P6	R/W	√	√		Undefined
FFFFF40EH	Port 7	P7	R/W	√	√		Undefined
FFFFF422H	Port 1 mode register	PM1	R/W	√	√		FFH
FFFFF424H	Port 2 mode register	PM2	R/W	√	√		FFH
FFFFF42AH	Port 5 mode register	PM5	R/W	√	√		FFH

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF42CH	Port 6 mode register	PM6	R/W	√	√		FFH
FFFFF42EH	Port 7 mode register	PM7	R/W	√	√		FFH
FFFFF442H	Port 1 mode control register	PMC1	R/W	√	√		00H
FFFFF444H	Port 2 mode control register	PMC2	R/W	√	√		01H
FFFFF44AH	Port 5 mode control register	PMC5	R/W	√	√		00H
FFFFF44CH	Port 6 mode control register	PMC6	R/W	√	√		00H
FFFFF44EH	Port 7 mode control register	PMC7	R/W	√	√		00H
FFFFF462H	Port 1 function control register	PFC1	R/W	√	√		00H
FFFFF464H	Port 2 function control register	PFC2	R/W	√	√		00H
FFFFF46AH	Port 5 function control register	PFC5	R/W	√	√		00H
FFFFF46CH	Port 6 function control register	PFC6	R/W	√	√		00H
FFFFF46EH	Port 7 function control register	PFC7	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
FFFFF48CH	Bus cycle period control register	BCP	R/W	√	√		00H
FFFFF48EH	Local bus sizing control register	LBS	R/W			√	5555H/AAAAH ^{Note}
FFFFF490H	Line buffer control register 0	LBC0	R/W			√	0000H
FFFFF492H	Line buffer control register 1	LBC1	R/W			√	0000H
FFFFF494H	DMA flyby transfer wait control register	FWC	R/W			√	7777H
FFFFF496H	DMA flyby transfer idle control register	FIC	R/W			√	3333H
FFFFF498H	Bus mode control register	BMC	R/W		√		00H
FFFFF49AH	Page ROM configuration register	PRC	R/W			√	7000H
FFFFF4A4H	SDRAM configuration register 1	SCR1	R/W			√	30C0H
FFFFF4A6H	SDRAM refresh control register 1	RFS1	R/W			√	0000H
FFFFF4ACH	SDRAM configuration register 3	SCR3	R/W			√	30C0H
FFFFF4AEH	SDRAM refresh control register 3	RFS3	R/W			√	0000H
FFFFF4B0H	SDRAM configuration register 4	SCR4	R/W			√	30C0H
FFFFF4B2H	SDRAM refresh control register 4	RFS4	R/W			√	0000H
FFFFF4B8H	SDRAM configuration register 6	SCR6	R/W			√	30C0H
FFFFF4BAH	SDRAM refresh control register 6	RFS6	R/W			√	0000H
FFFFF540H	Timer D0	TMD0	R			√	0000H

Note 32-bit mode: AAAAH

16-bit mode: 5555H

For details of 32-bit mode and 16-bit mode, refer to **3.3.1 Operating modes**.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
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
FFFFF5A0H	Timer ENC10	TMENC10	R/W			√	0000H
FFFFF5A2H	Compare register 100	CM100	R/W			√	0000H
FFFFF5A4H	Compare register 101	CM101	R/W			√	0000H
FFFFF5A6H	Capture/compare register 100	CC100	R/W			√	0000H
FFFFF5A8H	Capture/compare register 101	CC101	R/W			√	0000H
FFFFF5AAH	Capture/compare control register 10	CCR10	R/W	√	√		00H
FFFFF5ABH	Timer unit mode register 10	TUM10	R/W	√	√		00H
FFFFF5ACH	Timer control register 10	TMC10	R/W	√	√		00H
FFFFF5ADH	Valid edge select register 10	SESA10	R/W	√	√		00H
FFFFF5AEH	Prescaler mode register 10	PRM10	R/W	√	√		07H
FFFFF5AFH	Status register 10	STATUS10	R	√	√		00H
FFFFF5C0H	Noise elimination width setting register 10	NCW10	R/W		√		02H
FFFFF5D0H	Timer ENC11	TMENC11	R/W			√	0000H
FFFFF5D2H	Compare register 110	CM110	R/W			√	0000H
FFFFF5D4H	Compare register 111	CM111	R/W			√	0000H
FFFFF5D6H	Capture/compare register 110	CC110	R/W			√	0000H
FFFFF5D8H	Capture/compare register 111	CC111	R/W			√	0000H
FFFFF5DAH	Capture/compare control register 11	CCR11	R/W	√	√		00H
FFFFF5DBH	Timer unit mode register 11	TUM11	R/W	√	√		00H
FFFFF5DCH	Timer control register 11	TMC11	R/W	√	√		00H
FFFFF5DDH	Valid edge select register 11	SESA11	R/W	√	√		00H
FFFFF5DEH	Prescaler mode register 11	PRM11	R/W	√	√		07H
FFFFF5DFH	Status register 11	STATUS11	R	√	√		00H
FFFFF5F0H	Noise elimination width setting register 11	NCW11	R/W		√		02H
FFFFF600H	Timer C0	TMC0	R			√	0000H
FFFFF602H	Capture/compare register C00	CCC00	R/W			√	0000H
FFFFF604H	Capture/compare register C01	CCC01	R/W			√	0000H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
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	Noise elimination width setting register C0	NCWC0	R/W		√		02H
FFFFF620H	Timer C1	TMC1	R			√	0000H
FFFFF622H	Capture/compare register C10	CCC10	R/W			√	0000H
FFFFF624H	Capture/compare register C11	CCC11	R/W			√	0000H
FFFFF626H	Timer mode control register C10	TMCC10	R/W	√	√		00H
FFFFF628H	Timer mode control register C11	TMCC11	R/W		√		20H
FFFFF629H	Valid edge select register C1	SESC1	R/W		√		00H
FFFFF630H	Noise elimination width setting register C1	NCWC1	R/W		√		02H
FFFFF640H	Timer C2	TMC2	R			√	0000H
FFFFF642H	Capture/compare register C20	CCC20	R/W			√	0000H
FFFFF644H	Capture/compare register C21	CCC21	R/W			√	0000H
FFFFF646H	Timer mode control register C20	TMCC20	R/W	√	√		00H
FFFFF648H	Timer mode control register C21	TMCC21	R/W		√		20H
FFFFF649H	Valid edge select register C2	SESC2	R/W		√		00H
FFFFF650H	Noise elimination width setting register C2	NCWC2	R/W		√		02H
FFFFF660H	Timer C3	TMC3	R			√	0000H
FFFFF662H	Capture/compare register C30	CCC30	R/W			√	0000H
FFFFF664H	Capture/compare register C31	CCC31	R/W			√	0000H
FFFFF666H	Timer mode control register C30	TMCC30	R/W	√	√		00H
FFFFF668H	Timer mode control register C31	TMCC31	R/W		√		20H
FFFFF669H	Valid edge select register C3	SESC3	R/W		√		00H
FFFFF670H	Noise elimination width setting register C3	NCWC3	R/W		√		02H
FFFFF680H	Timer C4	TMC4	R			√	0000H
FFFFF682H	Capture/compare register C40	CCC40	R/W			√	0000H
FFFFF684H	Capture/compare register C41	CCC41	R/W			√	0000H
FFFFF686H	Timer mode control register C40	TMCC40	R/W	√	√		00H
FFFFF688H	Timer mode control register C41	TMCC41	R/W		√		20H
FFFFF6A0H	Timer C5	TMC5	R			√	0000H
FFFFF6A2H	Capture/compare register C50	CCC50	R/W			√	0000H
FFFFF6A4H	Capture/compare register C51	CCC51	R/W			√	0000H
FFFFF6A6H	Timer mode control register C50	TMCC50	R/W	√	√		00H
FFFFF6A8H	Timer mode control register C51	TMCC51	R/W		√		20H
FFFFF6C0H	Oscillation stabilization time select register	OSTS	R/W		√		04H
FFFFF80AH	Internal instruction RAM mode register	IRAMM	R/W	√	√		03H
FFFFF80CH	NMI reset status register	NRS	R	√	√		00H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
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		√		03H
FFFFF824H	Lock register	LOCKR	R	√	√		01H
FFFFF82CH	Clock source select register	CKS	R/W	√	√		00H
FFFFF82EH	USB clock control register	UCKC	R/W	√	√		00H
FFFFF836H	SSCG control register	SSCGC	R/W		√		Note 1
FFFFF8A0H	DMA terminal count output control register	DTOC	R/W	√	√		01H
FFFFF8A8H	DMA interface control register	DIFC	R/W	√	√		00H
FFFFFA00H	UARTB0 control register 0	UB0CTL0	R/W	√	√		10H
FFFFFA02H	UARTB0 control register 2	UB0CTL2	R/W			√	FFFFH
FFFFFA04H	UARTB0 status register	UB0STR	R/W	√	√		00H
FFFFFA06H	UARTB0 receive data register AP ^{Note 2}	UB0RXAP	R			√	00FFH
FFFFFA06H	UARTB0 receive data register	UB0RX	R		√		FFH
FFFFFA08H	UARTB0 transmit data register	UB0TX	W		√		FFH
FFFFFA0AH	UARTB0 FIFO control register 0	UB0FIC0	R/W	√	√		00H
FFFFFA0BH	UARTB0 FIFO control register 1	UB0FIC1	R/W	√	√		00H
FFFFFA0CH	UARTB0 FIFO control register 2	UB0FIC2	R/W			√	0000H
FFFFFA0CH	UARTB0 FIFO control register 2L	UB0FIC2L	R/W		√		00H
FFFFFA0DH	UARTB0 FIFO control register 2H	UB0FIC2H	R/W		√		00H
FFFFFA0EH	UARTB0 FIFO status register 0	UB0FIS0	R		√		00H
FFFFFA0FH	UARTB0 FIFO status register 1	UB0FIS1	R		√		10H
FFFFFA20H	UARTB1 control register 0	UB1CTL0	R/W	√	√		10H
FFFFFA22H	UARTB1 control register 2	UB1CTL2	R/W			√	FFFFH
FFFFFA24H	UARTB1 status register	UB1STR	R/W	√	√		00H
FFFFFA26H	UARTB1 receive data register AP ^{Note 2}	UB1RXAP	R			√	00FFH
FFFFFA26H	UARTB1 receive data register	UB1RX	R		√		FFH
FFFFFA28H	UARTB1 transmit data register	UB1TX	W		√		FFH
FFFFFA2AH	UARTB1 FIFO control register 0	UB1FIC0	R/W	√	√		00H
FFFFFA2BH	UARTB1 FIFO control register 1	UB1FIC1	R/W	√	√		00H

Notes 1. For details, see **8.3.3 SSCG control register (SSCGC)**.

2. This register can be used only in FIFO mode.

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFA2CH	UARTB1 FIFO control register 2	UB1FIC2	R/W			√	0000H
FFFFFA2CH	UARTB1 FIFO control register 2L	UB1FIC2L	R/W		√		00H
FFFFFA2DH	UARTB1 FIFO control register 2H	UB1FIC2H	R/W		√		00H
FFFFFA2EH	UARTB1 FIFO status register 0	UB1FIS0	R		√		00H
FFFFFA2FH	UARTB1 FIFO status register 1	UB1FIS1	R		√		10H
FFFFFB00H	PWM control register 0	PWMC0	R/W	√	√		08H
FFFFFB02H	PWM modulo register 0	PWM0	R/W			√	0000H
FFFFFB02H	PWM modulo register L0	PWML0	R/W		√		00H
FFFFFB03H	PWM modulo register H0	PWMH0	R/W		√		00H
FFFFFB10H	PWM control register 1	PWMC1	R/W	√	√		08H
FFFFFB12H	PWM modulo register 1	PWM1	R/W			√	0000H
FFFFFB12H	PWM modulo register L1	PWML1	R/W		√		00H
FFFFFB13H	PWM modulo register H1	PWMH1	R/W		√		00H
FFFFFC02H	External interrupt falling edge specification register 1	INTF1	R/W	√	√		00H
FFFFFC04H	External interrupt falling edge specification register 2	INTF2	R/W	√	√		00H
FFFFFC0AH	External interrupt falling edge specification register 5	INTF5	R/W	√	√		00H
FFFFFC0CH	External interrupt falling edge specification register 6	INTF6	R/W	√	√		00H
FFFFFC10H	External interrupt falling edge specification register AL	INTFAL	R/W	√	√		00H
FFFFFC16H	External interrupt falling edge specification register DH	INTFDH	R/W			√	0000H
FFFFFC16H	External interrupt falling edge specification register DHL	INTFDHL	R/W	√	√		00H
FFFFFC17H	External interrupt falling edge specification register DHH	INTFDHH	R/W	√	√		00H
FFFFFC22H	External interrupt rising edge specification register 1	INTR1	R/W	√	√		03H
FFFFFC24H	External interrupt rising edge specification register 2	INTR2	R/W	√	√		3FH
FFFFFC2AH	External interrupt rising edge specification register 5	INTR5	R/W	√	√		07H
FFFFFC2CH	External interrupt rising edge specification register 6	INTR6	R/W	√	√		E0H
FFFFFC30H	External interrupt rising edge specification register AL	INTRAL	R/W	√	√		03H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFC36H	External interrupt rising edge specification register DH	INTRDH	R/W			√	FFFFH
FFFFFC36H	External interrupt rising edge specification register DHL	INTRDHL	R/W	√	√		FFH
FFFFFC37H	External interrupt rising edge specification register DHH	INTRDHH	R/W	√	√		FFH
FFFFFD00H	Clocked serial interface mode register 30	CSIM30	R/W	√	√		00H
FFFFFD01H	Clocked serial interface clock select register 30	CSIC30	R/W	√	√		07H
FFFFFD02H	Receive data buffer register 30	SIRB30	R			√	0000H
FFFFFD02H	Receive data buffer register 30L	SIRB30L	R		√		00H
FFFFFD03H	Receive data buffer register 30H	SIRB30H	R		√		00H
FFFFFD06H	Transmit data CSI buffer register 30	SFDB30	R/W			√	0000H
FFFFFD06H	Transmit data CSI buffer register 30L	SFDB30L	R/W		√		00H
FFFFFD07H	Transmit data CSI buffer register 30H	SFDB30H	R/W		√		00H
FFFFFD08H	CSIBUF status register 30	SFA30	R/W	√	√		20H
FFFFFD09H	Transfer data length select register 30	CSIL30	R/W	√	√		00H
FFFFFD0CH	Transfer data number specification register 30	SFN30	R/W	√	√		00H
FFFFFD20H	Clocked serial interface mode register 31	CSIM31	R/W	√	√		00H
FFFFFD21H	Clocked serial interface clock select register 31	CSIC31	R/W	√	√		07H
FFFFFD22H	Receive data buffer register 31	SIRB31	R			√	0000H
FFFFFD22H	Receive data buffer register 31L	SIRB31L	R		√		00H
FFFFFD23H	Receive data buffer register 31H	SIRB31H	R		√		00H
FFFFFD26H	Transmit data CSI buffer register 31	SFDB31	R/W			√	0000H
FFFFFD26H	Transmit data CSI buffer register 31L	SFDB31L	R/W		√		00H
FFFFFD27H	Transmit data CSI buffer register 31H	SFDB31H	R/W		√		00H
FFFFFD28H	CSIBUF status register 31	SFA31	R/W	√	√		20H
FFFFFD29H	Transfer data length select register 31	CSIL31	R/W	√	√		00H
FFFFFD2CH	Transfer data number specification register 31	SFN31	R/W	√	√		00H
FFFFDF0H	USB function 0 DMA channel select register	UF0CS	R/W			√	0000H
FFFFDF2H	USB function 0 buffer control register	UF0BC	R/W	√	√		00H
FFFFE00H	UF0 EP0NAK register	UF0E0N	R/W		√		00H
FFFFE01H	UF0 EP0NAKALL register	UF0E0NA	R/W		√		00H
FFFFE02H	UF0 EPNAK register	UF0EN	R/W		√		00H
FFFFE03H	UF0 EPNAK mask register	UF0ENM	R/W		√		00H
FFFFE04H	UF0 SNDSIE register	UF0SDS	R/W		√		00H
FFFFE05H	UF0 CLR request register	UF0CLR	R		√		00H
FFFFE06H	UF0 SET request register	UF0SET	R		√		00H
FFFFE07H	UF0 EP status 0 register	UF0EPS0	R		√		00H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFFE08H	UF0 EP status 1 register	UF0EPS1	R		√		00H
FFFFFFE09H	UF0 EP status 2 register	UF0EPS2	R		√		00H
FFFFFFE10H	UF0 INT status 0 register	UF0IS0	R		√		00H
FFFFFFE11H	UF0 INT status 1 register	UF0IS1	R		√		00H
FFFFFFE12H	UF0 INT status 2 register	UF0IS2	R		√		00H
FFFFFFE13H	UF0 INT status 3 register	UF0IS3	R		√		00H
FFFFFFE14H	UF0 INT status 4 register	UF0IS4	R		√		00H
FFFFFFE17H	UF0 INT mask 0 register	UF0IM0	R/W		√		00H
FFFFFFE18H	UF0 INT mask 1 register	UF0IM1	R/W		√		00H
FFFFFFE19H	UF0 INT mask 2 register	UF0IM2	R/W		√		00H
FFFFFFE1AH	UF0 INT mask 3 register	UF0IM3	R/W		√		00H
FFFFFFE1BH	UF0 INT mask 4 register	UF0IM4	R/W		√		00H
FFFFFFE1EH	UF0 INT clear 0 register	UF0IC0	W		√		FFH
FFFFFFE1FH	UF0 INT clear 1 register	UF0IC1	W		√		FFH
FFFFFFE20H	UF0 INT clear 2 register	UF0IC2	W		√		FFH
FFFFFFE21H	UF0 INT clear 3 register	UF0IC3	W		√		FFH
FFFFFFE22H	UF0 INT clear 4 register	UF0IC4	W		√		FFH
FFFFFFE26H	UF0 INT & DMARQ register	UF0IDR	R/W		√		00H
FFFFFFE27H	UF0 DMA status 0 register	UF0DMS0	R		√		00H
FFFFFFE28H	UF0 DMA status 1 register	UF0DMS1	R		√		00H
FFFFFFE30H	UF0 FIFO clear 0 register	UF0FIC0	W		√		00H
FFFFFFE31H	UF0 FIFO clear 1 register	UF0FIC1	W		√		00H
FFFFFFE35H	UF0 data end register	UF0DEND	R/W		√		00H
FFFFFFE37H	UF0 GPR register	UF0GPR	R/W		√		00H
FFFFFFE3AH	UF0 mode control register	UF0MODC	R/W		√		00H
FFFFFFE3CH	UF0 mode status register	UF0MODS	R		√		00H
FFFFFFE40H	UF0 active interface number register	UF0AIFN	R/W		√		00H
FFFFFFE41H	UF0 active alternative setting register	UF0AAS	R/W		√		00H
FFFFFFE42H	UF0 alternative setting status register	UF0ASS	R		√		00H
FFFFFFE43H	UF0 endpoint 1 interface mapping register	UF0E1IM	R/W		√		00H
FFFFFFE44H	UF0 endpoint 2 interface mapping register	UF0E2IM	R/W		√		00H
FFFFFFE45H	UF0 endpoint 3 interface mapping register	UF0E3IM	R/W		√		00H
FFFFFFE46H	UF0 endpoint 4 interface mapping register	UF0E4IM	R/W		√		00H
FFFFFFE49H	UF0 endpoint 7 interface mapping register	UF0E7IM	R/W		√		00H
FFFFFFE4AH	UF0 endpoint 8 interface mapping register	UF0E8IM	R/W		√		00H
FFFFFFE80H	UF0 EP0 read register	UF0E0R	R		√		Undefined
FFFFFFE81H	UF0 EP0 length register	UF0E0L	R		√		00H
FFFFFFE82H	UF0 EP0 setup register	UF0E0ST	R		√		00H

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFFE83H	UF0 EP0 write register	UF0E0W	W		√		Undefined
FFFFFFE84H	UF0 bulk out 1 register	UF0BO1	R		√		Undefined
FFFFFFE85H	UF0 bulk out 1 length register	UF0BO1L	R		√		00H
FFFFFFE86H	UF0 bulk out 2 register	UF0BO2	R		√		Undefined
FFFFFFE87H	UF0 bulk out 2 length register	UF0BO2L	R		√		00H
FFFFFFE88H	UF0 bulk in 1 register	UF0BI1	W		√		Undefined
FFFFFFE89H	UF0 bulk in 2 register	UF0BI2	W		√		Undefined
FFFFFFE8AH	UF0 interrupt 1 register	UF0INT1	W		√		Undefined
FFFFFFE8BH	UF0 interrupt 2 register	UF0INT2	W		√		Undefined
FFFFFFEA2H	UF0 device status register L	UF0DSTL	R/W		√		00H
FFFFFFEA6H	UF0 EP0 status register L	UF0E0SL	R/W		√		00H
FFFFFFEA8H	UF0 EP1 status register L	UF0E1SL	R/W		√		00H
FFFFFFEAAH	UF0 EP2 status register L	UF0E2SL	R/W		√		00H
FFFFFFEACH	UF0 EP3 status register L	UF0E3SL	R/W		√		00H
FFFFFFEAEH	UF0 EP4 status register L	UF0E4SL	R/W		√		00H
FFFFFFEB4H	UF0 EP7 status register L	UF0E7SL	R/W		√		00H
FFFFFFEB6H	UF0 EP8 status register L	UF0E8SL	R/W		√		00H
★ FFFFFFFEC0H	UF0 address register	UF0ADRS	R		√		00H
★ FFFFFFFEC1H	UF0 configuration register	UF0CNF	R		√		00H
★ FFFFFFFEC2H	UF0 interface 0 register	UF0IF0	R		√		00H
★ FFFFFFFEC3H	UF0 interface 1 register	UF0IF1	R		√		00H
★ FFFFFFFEC4H	UF0 interface 2 register	UF0IF2	R		√		00H
★ FFFFFFFEC5H	UF0 interface 3 register	UF0IF3	R		√		00H
★ FFFFFFFEC6H	UF0 interface 4 register	UF0IF4	R		√		00H
FFFFFFED0H	UF0 descriptor length register	UF0DSCL	R/W		√		00H
FFFFFFED1H	UF0 device descriptor register 0	UF0DD0	R/W		√		Undefined
FFFFFFED2H	UF0 device descriptor register 1	UF0DD1	R/W		√		Undefined
FFFFFFED3H	UF0 device descriptor register 2	UF0DD2	R/W		√		Undefined
FFFFFFED4H	UF0 device descriptor register 3	UF0DD3	R/W		√		Undefined
FFFFFFED5H	UF0 device descriptor register 4	UF0DD4	R/W		√		Undefined
FFFFFFED6H	UF0 device descriptor register 5	UF0DD5	R/W		√		Undefined
FFFFFFED7H	UF0 device descriptor register 6	UF0DD6	R/W		√		Undefined
FFFFFFED8H	UF0 device descriptor register 7	UF0DD7	R/W		√		Undefined
FFFFFFED9H	UF0 device descriptor register 8	UF0DD8	R/W		√		Undefined
FFFFFFEDAH	UF0 device descriptor register 9	UF0DD9	R/W		√		Undefined
FFFFFFEDBH	UF0 device descriptor register 10	UF0DD10	R/W		√		Undefined
FFFFFFEDCH	UF0 device descriptor register 11	UF0DD11	R/W		√		Undefined
FFFFFFEDDH	UF0 device descriptor register 12	UF0DD12	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFEDEH	UF0 device descriptor register 13	UF0DD13	R/W		√		Undefined
FFFFFEDFH	UF0 device descriptor register 14	UF0DD14	R/W		√		Undefined
FFFFFEE0H	UF0 device descriptor register 15	UF0DD15	R/W		√		Undefined
FFFFFEE1H	UF0 device descriptor register 16	UF0DD16	R/W		√		Undefined
FFFFFEE2H	UF0 device descriptor register 17	UF0DD17	R/W		√		Undefined
FFFFFEE3H	UF0 configuration/interface/ endpoint descriptor register 0	UF0CIE0	R/W		√		Undefined
FFFFFEE4H	UF0 configuration/interface/ endpoint descriptor register 1	UF0CIE1	R/W		√		Undefined
FFFFFEE5H	UF0 configuration/interface/ endpoint descriptor register 2	UF0CIE2	R/W		√		Undefined
FFFFFEE6H	UF0 configuration/interface/ endpoint descriptor register 3	UF0CIE3	R/W		√		Undefined
FFFFFEE7H	UF0 configuration/interface/ endpoint descriptor register 4	UF0CIE4	R/W		√		Undefined
FFFFFEE8H	UF0 configuration/interface/ endpoint descriptor register 5	UF0CIE5	R/W		√		Undefined
FFFFFEE9H	UF0 configuration/interface/ endpoint descriptor register 6	UF0CIE6	R/W		√		Undefined
FFFFFEEAH	UF0 configuration/interface/ endpoint descriptor register 7	UF0CIE7	R/W		√		Undefined
FFFFFEEBH	UF0 configuration/interface/ endpoint descriptor register 8	UF0CIE8	R/W		√		Undefined
FFFFFEECH	UF0 configuration/interface/ endpoint descriptor register 9	UF0CIE9	R/W		√		Undefined
FFFFFEE DH	UF0 configuration/interface/ endpoint descriptor register 10	UF0CIE10	R/W		√		Undefined
FFFFFEEEH	UF0 configuration/interface/ endpoint descriptor register 11	UF0CIE11	R/W		√		Undefined
FFFFFEEFH	UF0 configuration/interface/ endpoint descriptor register 12	UF0CIE12	R/W		√		Undefined
FFFFFEF0H	UF0 configuration/interface/ endpoint descriptor register 13	UF0CIE13	R/W		√		Undefined
FFFFFEF1H	UF0 configuration/interface/ endpoint descriptor register 14	UF0CIE14	R/W		√		Undefined
FFFFFEF2H	UF0 configuration/interface/ endpoint descriptor register 15	UF0CIE15	R/W		√		Undefined
FFFFFEF3H	UF0 configuration/interface/ endpoint descriptor register 16	UF0CIE16	R/W		√		Undefined
FFFFFEF4H	UF0 configuration/interface/ endpoint descriptor register 17	UF0CIE17	R/W		√		Undefined
FFFFFEF5H	UF0 configuration/interface/ endpoint descriptor register 18	UF0CIE18	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFFE6H	UF0 configuration/interface/ endpoint descriptor register 19	UF0CIE19	R/W		√		Undefined
FFFFFFE7H	UF0 configuration/interface/ endpoint descriptor register 20	UF0CIE20	R/W		√		Undefined
FFFFFFE8H	UF0 configuration/interface/ endpoint descriptor register 21	UF0CIE21	R/W		√		Undefined
FFFFFFE9H	UF0 configuration/interface/ endpoint descriptor register 22	UF0CIE22	R/W		√		Undefined
FFFFFFEAH	UF0 configuration/interface/ endpoint descriptor register 23	UF0CIE23	R/W		√		Undefined
FFFFFFEBH	UF0 configuration/interface/ endpoint descriptor register 24	UF0CIE24	R/W		√		Undefined
FFFFFFECH	UF0 configuration/interface/ endpoint descriptor register 25	UF0CIE25	R/W		√		Undefined
FFFFFFEDH	UF0 configuration/interface/ endpoint descriptor register 26	UF0CIE26	R/W		√		Undefined
FFFFFFEEH	UF0 configuration/interface/ endpoint descriptor register 27	UF0CIE27	R/W		√		Undefined
FFFFFFEFH	UF0 configuration/interface/ endpoint descriptor register 28	UF0CIE28	R/W		√		Undefined
FFFFFFF0H	UF0 configuration/interface/ endpoint descriptor register 29	UF0CIE29	R/W		√		Undefined
FFFFFFF1H	UF0 configuration/interface/ endpoint descriptor register 30	UF0CIE30	R/W		√		Undefined
FFFFFFF2H	UF0 configuration/interface/ endpoint descriptor register 31	UF0CIE31	R/W		√		Undefined
FFFFFFF3H	UF0 configuration/interface/ endpoint descriptor register 32	UF0CIE32	R/W		√		Undefined
FFFFFFF4H	UF0 configuration/interface/ endpoint descriptor register 33	UF0CIE33	R/W		√		Undefined
FFFFFFF5H	UF0 configuration/interface/ endpoint descriptor register 34	UF0CIE34	R/W		√		Undefined
FFFFFFF6H	UF0 configuration/interface/ endpoint descriptor register 35	UF0CIE35	R/W		√		Undefined
FFFFFFF7H	UF0 configuration/interface/ endpoint descriptor register 36	UF0CIE36	R/W		√		Undefined
FFFFFFF8H	UF0 configuration/interface/ endpoint descriptor register 37	UF0CIE37	R/W		√		Undefined
FFFFFFF9H	UF0 configuration/interface/ endpoint descriptor register 38	UF0CIE38	R/W		√		Undefined
FFFFFFFAH	UF0 configuration/interface/ endpoint descriptor register 39	UF0CIE39	R/W		√		Undefined
FFFFFFFBH	UF0 configuration/interface/ endpoint descriptor register 40	UF0CIE40	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF0CH	UF0 configuration/interface/ endpoint descriptor register 41	UF0CIE41	R/W		√		Undefined
FFFFFF0DH	UF0 configuration/interface/ endpoint descriptor register 42	UF0CIE42	R/W		√		Undefined
FFFFFF0EH	UF0 configuration/interface/ endpoint descriptor register 43	UF0CIE43	R/W		√		Undefined
FFFFFF0FH	UF0 configuration/interface/ endpoint descriptor register 44	UF0CIE44	R/W		√		Undefined
FFFFFF10H	UF0 configuration/interface/ endpoint descriptor register 45	UF0CIE45	R/W		√		Undefined
FFFFFF11H	UF0 configuration/interface/ endpoint descriptor register 46	UF0CIE46	R/W		√		Undefined
FFFFFF12H	UF0 configuration/interface/ endpoint descriptor register 47	UF0CIE47	R/W		√		Undefined
FFFFFF13H	UF0 configuration/interface/ endpoint descriptor register 48	UF0CIE48	R/W		√		Undefined
FFFFFF14H	UF0 configuration/interface/ endpoint descriptor register 49	UF0CIE49	R/W		√		Undefined
FFFFFF15H	UF0 configuration/interface/ endpoint descriptor register 50	UF0CIE50	R/W		√		Undefined
FFFFFF16H	UF0 configuration/interface/ endpoint descriptor register 51	UF0CIE51	R/W		√		Undefined
FFFFFF17H	UF0 configuration/interface/ endpoint descriptor register 52	UF0CIE52	R/W		√		Undefined
FFFFFF18H	UF0 configuration/interface/ endpoint descriptor register 53	UF0CIE53	R/W		√		Undefined
FFFFFF19H	UF0 configuration/interface/ endpoint descriptor register 54	UF0CIE54	R/W		√		Undefined
FFFFFF1AH	UF0 configuration/interface/ endpoint descriptor register 55	UF0CIE55	R/W		√		Undefined
FFFFFF1BH	UF0 configuration/interface/ endpoint descriptor register 56	UF0CIE56	R/W		√		Undefined
FFFFFF1CH	UF0 configuration/interface/ endpoint descriptor register 57	UF0CIE57	R/W		√		Undefined
FFFFFF1DH	UF0 configuration/interface/ endpoint descriptor register 58	UF0CIE58	R/W		√		Undefined
FFFFFF1EH	UF0 configuration/interface/ endpoint descriptor register 59	UF0CIE59	R/W		√		Undefined
FFFFFF1FH	UF0 configuration/interface/ endpoint descriptor register 60	UF0CIE60	R/W		√		Undefined
FFFFFF20H	UF0 configuration/interface/ endpoint descriptor register 61	UF0CIE61	R/W		√		Undefined
FFFFFF21H	UF0 configuration/interface/ endpoint descriptor register 62	UF0CIE62	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF22H	UF0 configuration/interface/ endpoint descriptor register 63	UF0CIE63	R/W		√		Undefined
FFFFFF23H	UF0 configuration/interface/ endpoint descriptor register 64	UF0CIE64	R/W		√		Undefined
FFFFFF24H	UF0 configuration/interface/ endpoint descriptor register 65	UF0CIE65	R/W		√		Undefined
FFFFFF25H	UF0 configuration/interface/ endpoint descriptor register 66	UF0CIE66	R/W		√		Undefined
FFFFFF26H	UF0 configuration/interface/ endpoint descriptor register 67	UF0CIE67	R/W		√		Undefined
FFFFFF27H	UF0 configuration/interface/ endpoint descriptor register 68	UF0CIE68	R/W		√		Undefined
FFFFFF28H	UF0 configuration/interface/ endpoint descriptor register 69	UF0CIE69	R/W		√		Undefined
FFFFFF29H	UF0 configuration/interface/ endpoint descriptor register 70	UF0CIE70	R/W		√		Undefined
FFFFFF2AH	UF0 configuration/interface/ endpoint descriptor register 71	UF0CIE71	R/W		√		Undefined
FFFFFF2BH	UF0 configuration/interface/ endpoint descriptor register 72	UF0CIE72	R/W		√		Undefined
FFFFFF2CH	UF0 configuration/interface/ endpoint descriptor register 73	UF0CIE73	R/W		√		Undefined
FFFFFF2DH	UF0 configuration/interface/ endpoint descriptor register 74	UF0CIE74	R/W		√		Undefined
FFFFFF2EH	UF0 configuration/interface/ endpoint descriptor register 75	UF0CIE75	R/W		√		Undefined
FFFFFF2FH	UF0 configuration/interface/ endpoint descriptor register 76	UF0CIE76	R/W		√		Undefined
FFFFFF30H	UF0 configuration/interface/ endpoint descriptor register 77	UF0CIE77	R/W		√		Undefined
FFFFFF31H	UF0 configuration/interface/ endpoint descriptor register 78	UF0CIE78	R/W		√		Undefined
FFFFFF32H	UF0 configuration/interface/ endpoint descriptor register 79	UF0CIE79	R/W		√		Undefined
FFFFFF33H	UF0 configuration/interface/ endpoint descriptor register 80	UF0CIE80	R/W		√		Undefined
FFFFFF34H	UF0 configuration/interface/ endpoint descriptor register 81	UF0CIE81	R/W		√		Undefined
FFFFFF35H	UF0 configuration/interface/ endpoint descriptor register 82	UF0CIE82	R/W		√		Undefined
FFFFFF36H	UF0 configuration/interface/ endpoint descriptor register 83	UF0CIE83	R/W		√		Undefined
FFFFFF37H	UF0 configuration/interface/ endpoint descriptor register 84	UF0CIE84	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF38H	UF0 configuration/interface/ endpoint descriptor register 85	UF0CIE85	R/W		√		Undefined
FFFFFF39H	UF0 configuration/interface/ endpoint descriptor register 86	UF0CIE86	R/W		√		Undefined
FFFFFF3AH	UF0 configuration/interface/ endpoint descriptor register 87	UF0CIE87	R/W		√		Undefined
FFFFFF3BH	UF0 configuration/interface/ endpoint descriptor register 88	UF0CIE88	R/W		√		Undefined
FFFFFF3CH	UF0 configuration/interface/ endpoint descriptor register 89	UF0CIE89	R/W		√		Undefined
FFFFFF3DH	UF0 configuration/interface/ endpoint descriptor register 90	UF0CIE90	R/W		√		Undefined
FFFFFF3EH	UF0 configuration/interface/ endpoint descriptor register 91	UF0CIE91	R/W		√		Undefined
FFFFFF3FH	UF0 configuration/interface/ endpoint descriptor register 92	UF0CIE92	R/W		√		Undefined
FFFFFF40H	UF0 configuration/interface/ endpoint descriptor register 93	UF0CIE93	R/W		√		Undefined
FFFFFF41H	UF0 configuration/interface/ endpoint descriptor register 94	UF0CIE94	R/W		√		Undefined
FFFFFF42H	UF0 configuration/interface/ endpoint descriptor register 95	UF0CIE95	R/W		√		Undefined
FFFFFF43H	UF0 configuration/interface/ endpoint descriptor register 96	UF0CIE96	R/W		√		Undefined
FFFFFF44H	UF0 configuration/interface/ endpoint descriptor register 97	UF0CIE97	R/W		√		Undefined
FFFFFF45H	UF0 configuration/interface/ endpoint descriptor register 98	UF0CIE98	R/W		√		Undefined
FFFFFF46H	UF0 configuration/interface/ endpoint descriptor register 99	UF0CIE99	R/W		√		Undefined
FFFFFF47H	UF0 configuration/interface/ endpoint descriptor register 100	UF0CIE100	R/W		√		Undefined
FFFFFF48H	UF0 configuration/interface/ endpoint descriptor register 101	UF0CIE101	R/W		√		Undefined
FFFFFF49H	UF0 configuration/interface/ endpoint descriptor register 102	UF0CIE102	R/W		√		Undefined
FFFFFF4AH	UF0 configuration/interface/ endpoint descriptor register 103	UF0CIE103	R/W		√		Undefined
FFFFFF4BH	UF0 configuration/interface/ endpoint descriptor register 104	UF0CIE104	R/W		√		Undefined
FFFFFF4CH	UF0 configuration/interface/ endpoint descriptor register 105	UF0CIE105	R/W		√		Undefined
FFFFFF4DH	UF0 configuration/interface/ endpoint descriptor register 106	UF0CIE106	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF4EH	UF0 configuration/interface/ endpoint descriptor register 107	UF0CIE107	R/W		√		Undefined
FFFFFF4FH	UF0 configuration/interface/ endpoint descriptor register 108	UF0CIE108	R/W		√		Undefined
FFFFFF50H	UF0 configuration/interface/ endpoint descriptor register 109	UF0CIE109	R/W		√		Undefined
FFFFFF51H	UF0 configuration/interface/ endpoint descriptor register 110	UF0CIE110	R/W		√		Undefined
FFFFFF52H	UF0 configuration/interface/ endpoint descriptor register 111	UF0CIE111	R/W		√		Undefined
FFFFFF53H	UF0 configuration/interface/ endpoint descriptor register 112	UF0CIE112	R/W		√		Undefined
FFFFFF54H	UF0 configuration/interface/ endpoint descriptor register 113	UF0CIE113	R/W		√		Undefined
FFFFFF55H	UF0 configuration/interface/ endpoint descriptor register 114	UF0CIE114	R/W		√		Undefined
FFFFFF56H	UF0 configuration/interface/ endpoint descriptor register 115	UF0CIE115	R/W		√		Undefined
FFFFFF57H	UF0 configuration/interface/ endpoint descriptor register 116	UF0CIE116	R/W		√		Undefined
FFFFFF58H	UF0 configuration/interface/ endpoint descriptor register 117	UF0CIE117	R/W		√		Undefined
FFFFFF59H	UF0 configuration/interface/ endpoint descriptor register 118	UF0CIE118	R/W		√		Undefined
FFFFFF5AH	UF0 configuration/interface/ endpoint descriptor register 119	UF0CIE119	R/W		√		Undefined
FFFFFF5BH	UF0 configuration/interface/ endpoint descriptor register 120	UF0CIE120	R/W		√		Undefined
FFFFFF5CH	UF0 configuration/interface/ endpoint descriptor register 121	UF0CIE121	R/W		√		Undefined
FFFFFF5DH	UF0 configuration/interface/ endpoint descriptor register 122	UF0CIE122	R/W		√		Undefined
FFFFFF5EH	UF0 configuration/interface/ endpoint descriptor register 123	UF0CIE123	R/W		√		Undefined
FFFFFF5FH	UF0 configuration/interface/ endpoint descriptor register 124	UF0CIE124	R/W		√		Undefined
FFFFFF60H	UF0 configuration/interface/ endpoint descriptor register 125	UF0CIE125	R/W		√		Undefined
FFFFFF61H	UF0 configuration/interface/ endpoint descriptor register 126	UF0CIE126	R/W		√		Undefined
FFFFFF62H	UF0 configuration/interface/ endpoint descriptor register 127	UF0CIE127	R/W		√		Undefined
FFFFFF63H	UF0 configuration/interface/ endpoint descriptor register 128	UF0CIE128	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF64H	UF0 configuration/interface/ endpoint descriptor register 129	UF0CIE129	R/W		√		Undefined
FFFFFF65H	UF0 configuration/interface/ endpoint descriptor register 130	UF0CIE130	R/W		√		Undefined
FFFFFF66H	UF0 configuration/interface/ endpoint descriptor register 131	UF0CIE131	R/W		√		Undefined
FFFFFF67H	UF0 configuration/interface/ endpoint descriptor register 132	UF0CIE132	R/W		√		Undefined
FFFFFF68H	UF0 configuration/interface/ endpoint descriptor register 133	UF0CIE133	R/W		√		Undefined
FFFFFF69H	UF0 configuration/interface/ endpoint descriptor register 134	UF0CIE134	R/W		√		Undefined
FFFFFF6AH	UF0 configuration/interface/ endpoint descriptor register 135	UF0CIE135	R/W		√		Undefined
FFFFFF6BH	UF0 configuration/interface/ endpoint descriptor register 136	UF0CIE136	R/W		√		Undefined
FFFFFF6CH	UF0 configuration/interface/ endpoint descriptor register 137	UF0CIE137	R/W		√		Undefined
FFFFFF6DH	UF0 configuration/interface/ endpoint descriptor register 138	UF0CIE138	R/W		√		Undefined
FFFFFF6EH	UF0 configuration/interface/ endpoint descriptor register 139	UF0CIE139	R/W		√		Undefined
FFFFFF6FH	UF0 configuration/interface/ endpoint descriptor register 140	UF0CIE140	R/W		√		Undefined
FFFFFF70H	UF0 configuration/interface/ endpoint descriptor register 141	UF0CIE141	R/W		√		Undefined
FFFFFF71H	UF0 configuration/interface/ endpoint descriptor register 142	UF0CIE142	R/W		√		Undefined
FFFFFF72H	UF0 configuration/interface/ endpoint descriptor register 143	UF0CIE143	R/W		√		Undefined
FFFFFF73H	UF0 configuration/interface/ endpoint descriptor register 144	UF0CIE144	R/W		√		Undefined
FFFFFF74H	UF0 configuration/interface/ endpoint descriptor register 145	UF0CIE145	R/W		√		Undefined
FFFFFF75H	UF0 configuration/interface/ endpoint descriptor register 146	UF0CIE146	R/W		√		Undefined
FFFFFF76H	UF0 configuration/interface/ endpoint descriptor register 147	UF0CIE147	R/W		√		Undefined
FFFFFF77H	UF0 configuration/interface/ endpoint descriptor register 148	UF0CIE148	R/W		√		Undefined
FFFFFF78H	UF0 configuration/interface/ endpoint descriptor register 149	UF0CIE149	R/W		√		Undefined
FFFFFF79H	UF0 configuration/interface/ endpoint descriptor register 150	UF0CIE150	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF7AH	UF0 configuration/interface/ endpoint descriptor register 151	UF0CIE151	R/W		√		Undefined
FFFFFF7BH	UF0 configuration/interface/ endpoint descriptor register 152	UF0CIE152	R/W		√		Undefined
FFFFFF7CH	UF0 configuration/interface/ endpoint descriptor register 153	UF0CIE153	R/W		√		Undefined
FFFFFF7DH	UF0 configuration/interface/ endpoint descriptor register 154	UF0CIE154	R/W		√		Undefined
FFFFFF7EH	UF0 configuration/interface/ endpoint descriptor register 155	UF0CIE155	R/W		√		Undefined
FFFFFF7FH	UF0 configuration/interface/ endpoint descriptor register 156	UF0CIE156	R/W		√		Undefined
FFFFFF80H	UF0 configuration/interface/ endpoint descriptor register 157	UF0CIE157	R/W		√		Undefined
FFFFFF81H	UF0 configuration/interface/ endpoint descriptor register 158	UF0CIE158	R/W		√		Undefined
FFFFFF82H	UF0 configuration/interface/ endpoint descriptor register 159	UF0CIE159	R/W		√		Undefined
FFFFFF83H	UF0 configuration/interface/ endpoint descriptor register 160	UF0CIE160	R/W		√		Undefined
FFFFFF84H	UF0 configuration/interface/ endpoint descriptor register 161	UF0CIE161	R/W		√		Undefined
FFFFFF85H	UF0 configuration/interface/ endpoint descriptor register 162	UF0CIE162	R/W		√		Undefined
FFFFFF86H	UF0 configuration/interface/ endpoint descriptor register 163	UF0CIE163	R/W		√		Undefined
FFFFFF87H	UF0 configuration/interface/ endpoint descriptor register 164	UF0CIE164	R/W		√		Undefined
FFFFFF88H	UF0 configuration/interface/ endpoint descriptor register 165	UF0CIE165	R/W		√		Undefined
FFFFFF89H	UF0 configuration/interface/ endpoint descriptor register 166	UF0CIE166	R/W		√		Undefined
FFFFFF8AH	UF0 configuration/interface/ endpoint descriptor register 167	UF0CIE167	R/W		√		Undefined
FFFFFF8BH	UF0 configuration/interface/ endpoint descriptor register 168	UF0CIE168	R/W		√		Undefined
FFFFFF8CH	UF0 configuration/interface/ endpoint descriptor register 169	UF0CIE169	R/W		√		Undefined
FFFFFF8DH	UF0 configuration/interface/ endpoint descriptor register 170	UF0CIE170	R/W		√		Undefined
FFFFFF8EH	UF0 configuration/interface/ endpoint descriptor register 171	UF0CIE171	R/W		√		Undefined
FFFFFF8FH	UF0 configuration/interface/ endpoint descriptor register 172	UF0CIE172	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF90H	UF0 configuration/interface/ endpoint descriptor register 173	UF0CIE173	R/W		√		Undefined
FFFFFF91H	UF0 configuration/interface/ endpoint descriptor register 174	UF0CIE174	R/W		√		Undefined
FFFFFF92H	UF0 configuration/interface/ endpoint descriptor register 175	UF0CIE175	R/W		√		Undefined
FFFFFF93H	UF0 configuration/interface/ endpoint descriptor register 176	UF0CIE176	R/W		√		Undefined
FFFFFF94H	UF0 configuration/interface/ endpoint descriptor register 177	UF0CIE177	R/W		√		Undefined
FFFFFF95H	UF0 configuration/interface/ endpoint descriptor register 178	UF0CIE178	R/W		√		Undefined
FFFFFF96H	UF0 configuration/interface/ endpoint descriptor register 179	UF0CIE179	R/W		√		Undefined
FFFFFF97H	UF0 configuration/interface/ endpoint descriptor register 180	UF0CIE180	R/W		√		Undefined
FFFFFF98H	UF0 configuration/interface/ endpoint descriptor register 181	UF0CIE181	R/W		√		Undefined
FFFFFF99H	UF0 configuration/interface/ endpoint descriptor register 182	UF0CIE182	R/W		√		Undefined
FFFFFF9AH	UF0 configuration/interface/ endpoint descriptor register 183	UF0CIE183	R/W		√		Undefined
FFFFFF9BH	UF0 configuration/interface/ endpoint descriptor register 184	UF0CIE184	R/W		√		Undefined
FFFFFF9CH	UF0 configuration/interface/ endpoint descriptor register 185	UF0CIE185	R/W		√		Undefined
FFFFFF9DH	UF0 configuration/interface/ endpoint descriptor register 186	UF0CIE186	R/W		√		Undefined
FFFFFF9EH	UF0 configuration/interface/ endpoint descriptor register 187	UF0CIE187	R/W		√		Undefined
FFFFFF9FH	UF0 configuration/interface/ endpoint descriptor register 188	UF0CIE188	R/W		√		Undefined
FFFFFFA0H	UF0 configuration/interface/ endpoint descriptor register 189	UF0CIE189	R/W		√		Undefined
FFFFFFA1H	UF0 configuration/interface/ endpoint descriptor register 190	UF0CIE190	R/W		√		Undefined
FFFFFFA2H	UF0 configuration/interface/ endpoint descriptor register 191	UF0CIE191	R/W		√		Undefined
FFFFFFA3H	UF0 configuration/interface/ endpoint descriptor register 192	UF0CIE192	R/W		√		Undefined
FFFFFFA4H	UF0 configuration/interface/ endpoint descriptor register 193	UF0CIE193	R/W		√		Undefined
FFFFFFA5H	UF0 configuration/interface/ endpoint descriptor register 194	UF0CIE194	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFFA6H	UF0 configuration/interface/ endpoint descriptor register 195	UF0CIE195	R/W		√		Undefined
FFFFFFA7H	UF0 configuration/interface/ endpoint descriptor register 196	UF0CIE196	R/W		√		Undefined
FFFFFFA8H	UF0 configuration/interface/ endpoint descriptor register 197	UF0CIE197	R/W		√		Undefined
FFFFFFA9H	UF0 configuration/interface/ endpoint descriptor register 198	UF0CIE198	R/W		√		Undefined
FFFFFFAAH	UF0 configuration/interface/ endpoint descriptor register 199	UF0CIE199	R/W		√		Undefined
FFFFFFABH	UF0 configuration/interface/ endpoint descriptor register 200	UF0CIE200	R/W		√		Undefined
FFFFFFACH	UF0 configuration/interface/ endpoint descriptor register 201	UF0CIE201	R/W		√		Undefined
FFFFFFADH	UF0 configuration/interface/ endpoint descriptor register 202	UF0CIE202	R/W		√		Undefined
FFFFFFAEH	UF0 configuration/interface/ endpoint descriptor register 203	UF0CIE203	R/W		√		Undefined
FFFFFFAFH	UF0 configuration/interface/ endpoint descriptor register 204	UF0CIE204	R/W		√		Undefined
FFFFFFB0H	UF0 configuration/interface/ endpoint descriptor register 205	UF0CIE205	R/W		√		Undefined
FFFFFFB1H	UF0 configuration/interface/ endpoint descriptor register 206	UF0CIE206	R/W		√		Undefined
FFFFFFB2H	UF0 configuration/interface/ endpoint descriptor register 207	UF0CIE207	R/W		√		Undefined
FFFFFFB3H	UF0 configuration/interface/ endpoint descriptor register 208	UF0CIE208	R/W		√		Undefined
FFFFFFB4H	UF0 configuration/interface/ endpoint descriptor register 209	UF0CIE209	R/W		√		Undefined
FFFFFFB5H	UF0 configuration/interface/ endpoint descriptor register 210	UF0CIE210	R/W		√		Undefined
FFFFFFB6H	UF0 configuration/interface/ endpoint descriptor register 211	UF0CIE211	R/W		√		Undefined
FFFFFFB7H	UF0 configuration/interface/ endpoint descriptor register 212	UF0CIE212	R/W		√		Undefined
FFFFFFB8H	UF0 configuration/interface/ endpoint descriptor register 213	UF0CIE213	R/W		√		Undefined
FFFFFFB9H	UF0 configuration/interface/ endpoint descriptor register 214	UF0CIE214	R/W		√		Undefined
FFFFFFBAH	UF0 configuration/interface/ endpoint descriptor register 215	UF0CIE215	R/W		√		Undefined
FFFFFFBBH	UF0 configuration/interface/ endpoint descriptor register 216	UF0CIE216	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFFBCH	UF0 configuration/interface/ endpoint descriptor register 217	UF0CIE217	R/W		√		Undefined
FFFFFFBDH	UF0 configuration/interface/ endpoint descriptor register 218	UF0CIE218	R/W		√		Undefined
FFFFFFBEH	UF0 configuration/interface/ endpoint descriptor register 219	UF0CIE219	R/W		√		Undefined
FFFFFFBFH	UF0 configuration/interface/ endpoint descriptor register 220	UF0CIE220	R/W		√		Undefined
FFFFFFC0H	UF0 configuration/interface/ endpoint descriptor register 221	UF0CIE221	R/W		√		Undefined
FFFFFFC1H	UF0 configuration/interface/ endpoint descriptor register 222	UF0CIE222	R/W		√		Undefined
FFFFFFC2H	UF0 configuration/interface/ endpoint descriptor register 223	UF0CIE223	R/W		√		Undefined
FFFFFFC3H	UF0 configuration/interface/ endpoint descriptor register 224	UF0CIE224	R/W		√		Undefined
FFFFFFC4H	UF0 configuration/interface/ endpoint descriptor register 225	UF0CIE225	R/W		√		Undefined
FFFFFFC5H	UF0 configuration/interface/ endpoint descriptor register 226	UF0CIE226	R/W		√		Undefined
FFFFFFC6H	UF0 configuration/interface/ endpoint descriptor register 227	UF0CIE227	R/W		√		Undefined
FFFFFFC7H	UF0 configuration/interface/ endpoint descriptor register 228	UF0CIE228	R/W		√		Undefined
FFFFFFC8H	UF0 configuration/interface/ endpoint descriptor register 229	UF0CIE229	R/W		√		Undefined
FFFFFFC9H	UF0 configuration/interface/ endpoint descriptor register 230	UF0CIE230	R/W		√		Undefined
FFFFFFCAH	UF0 configuration/interface/ endpoint descriptor register 231	UF0CIE231	R/W		√		Undefined
FFFFFFCBH	UF0 configuration/interface/ endpoint descriptor register 232	UF0CIE232	R/W		√		Undefined
FFFFFFCCH	UF0 configuration/interface/ endpoint descriptor register 233	UF0CIE233	R/W		√		Undefined
FFFFFFCDH	UF0 configuration/interface/ endpoint descriptor register 234	UF0CIE234	R/W		√		Undefined
FFFFFFCEH	UF0 configuration/interface/ endpoint descriptor register 235	UF0CIE235	R/W		√		Undefined
FFFFFFCFH	UF0 configuration/interface/ endpoint descriptor register 236	UF0CIE236	R/W		√		Undefined
FFFFFFD0H	UF0 configuration/interface/ endpoint descriptor register 237	UF0CIE237	R/W		√		Undefined
FFFFFFD1H	UF0 configuration/interface/ endpoint descriptor register 238	UF0CIE238	R/W		√		Undefined

Address	Function Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFFD2H	UF0 configuration/interface/ endpoint descriptor register 239	UF0CIE239	R/W		√		Undefined
FFFFFFD3H	UF0 configuration/interface/ endpoint descriptor register 240	UF0CIE240	R/W		√		Undefined
FFFFFFD4H	UF0 configuration/interface/ endpoint descriptor register 241	UF0CIE241	R/W		√		Undefined
FFFFFFD5H	UF0 configuration/interface/ endpoint descriptor register 242	UF0CIE242	R/W		√		Undefined
FFFFFFD6H	UF0 configuration/interface/ endpoint descriptor register 243	UF0CIE243	R/W		√		Undefined
FFFFFFD7H	UF0 configuration/interface/ endpoint descriptor register 244	UF0CIE244	R/W		√		Undefined
FFFFFFD8H	UF0 configuration/interface/ endpoint descriptor register 245	UF0CIE245	R/W		√		Undefined
FFFFFFD9H	UF0 configuration/interface/ endpoint descriptor register 246	UF0CIE246	R/W		√		Undefined
FFFFFFDAH	UF0 configuration/interface/ endpoint descriptor register 247	UF0CIE247	R/W		√		Undefined
FFFFFFDBH	UF0 configuration/interface/ endpoint descriptor register 248	UF0CIE248	R/W		√		Undefined
FFFFFFDCH	UF0 configuration/interface/ endpoint descriptor register 249	UF0CIE249	R/W		√		Undefined
FFFFFFDDH	UF0 configuration/interface/ endpoint descriptor register 250	UF0CIE250	R/W		√		Undefined
FFFFFFDEH	UF0 configuration/interface/ endpoint descriptor register 251	UF0CIE251	R/W		√		Undefined
FFFFFFDFH	UF0 configuration/interface/ endpoint descriptor register 252	UF0CIE252	R/W		√		Undefined
FFFFFFE0H	UF0 configuration/interface/ endpoint descriptor register 253	UF0CIE253	R/W		√		Undefined
FFFFFFE1H	UF0 configuration/interface/ endpoint descriptor register 254	UF0CIE254	R/W		√		Undefined
FFFFFFE2H	UF0 configuration/interface/ endpoint descriptor register 255	UF0CIE255	R/W		√		Undefined

3.4.8 Specific registers

Specific registers are registers that are protected from being written with illegal data due to erroneous program execution, etc. The V850E/ME2 has four specific registers, the power-save control register (PSC) (see **8.6.2 (3) Power-save control register (PSC)**), clock control register (CKC) (see **8.3.1 Clock control register (CKC)**), clock source select register (CKS) (see **8.3.2 Clock source select register (CKS)**), and SSCG control register (SSCGC) (see **8.3.3 SSCG control register (SSCGC)**). Disable DMA transfer when writing to a specific register.

There is also the command register (PRCMD), a protection register supporting write operations for specific registers to avoid an unexpected stoppage of the application system due to erroneous program execution (see **8.6.2 (2) Command register (PRCMD)**).

3.4.9 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/ME2 waits may be required depending on the operation frequency. Set the values described in the table below to the VSWC in accordance with the operation frequency used.

This register can be read or written in 1-bit or 8-bit units (address: FFFFF06EH, initial value: 77H).

★

Operation Frequency (fx)	VSWC Setting Value
10.00 MHz ≤ fx ≤ 25.00 MHz	00
25.00 MHz < fx ≤ 34.00 MHz	10
34.00 MHz < fx ≤ 68.00 MHz	11
68.00 MHz < fx ≤ 75.00 MHz	12
75.00 MHz < fx ≤ 103.00 MHz	22
103.00 MHz < fx ≤ 125.00 MHz	23
125.00 MHz < fx ≤ 150.00 MHz	33

Remark If the timing of changing a count value contend with the timing of accessing a register when accessing a register having status flags that indicate the status of the on-chip peripheral functions (such as UBnSTR) 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 a longer time to access an on-chip peripheral I/O register.

3.4.10 Initialization sequence

Initialize the V850E/ME2 in the following sequence.

<1> Automatically branch to address 100000H after reset is cleared

Set the following registers that affect the external bus access performance using the program located at address 100000H. Execution automatically branches to 100000H when the reset signal is input in the power-on status.

- System wait control register (VSWC)
Setting of a wait cycle for accessing the on-chip peripheral I/O
- Data wait control registers 0 and 1 (DWC0 and DWC1)
Setting of a data wait cycle of the external bus
- Address setup wait control register (ASC)
Setting of an address setup wait cycle of the external bus
- Bus cycle control register (BCC)
Setting of an idle state of the external bus

As necessary, set chip area select control registers 0 and 1 (CSC0 and CSC1), bus cycle type configuration registers 0 and 1 (BCT0 and BCT1), the local bus sizing control register (LBS), the endian configuration register (BEC), line buffer control registers 0 and 1 (LBC0 and LBC1), and the page ROM configuration register (PRC).

- Cautions**
1. **Disable all interrupts from when the reset signal is cleared until when the program code is completely transferred to the internal instruction RAM (while steps <1> to <3> of the initialization sequence are being executed). Maskable interrupts are masked by default and do not have to be disabled.**
 2. **Set SDRAM configuration registers 1, 3, 4, and 6 (SCR1, SCR3, SCR4, and SCR6) after the processing of step <2>.**

<2> Checking LOCK bit of lock register (LOCKR)

After setting the registers in <1> above, check whether the LOCK bit of the LOCKR register is cleared to 0 (PLL is locked), and set the registers as follows.

- (i) System wait control register (VSWC)
Set to x7H (x: Value set in <1>)
For example, if 11H is set in <1>, set 17H here.
- (ii) Bus mode control register (BMC)
Set the frequency division value of the external bus.
- (iii) System wait control register (VSWC)
Re-set the value set in <1>.
- (iv) Clock control register (CKC)
Set the internal system clock frequency division value.
- (v) Clock source select register (CKS)
Switch from OSC output to SSCG output (switch the clock supply to the CPU from the input frequency to the X1 and X2 pins to the frequency multiplied by 8 by the PLL).

Remark The CKC and CKS registers must be rewritten in a special sequence because they are specific registers.

<3> Transferring program code to internal instruction RAM

Transfer the program code to the internal instruction RAM by program processing or using the DMA function. When using the DMA function, check the completion of DMA transfer by polling bit 7 (DMAIFn) of the DMA interrupt control register (DMAICn), without using the DMA transfer end interrupt (INTDMAn) (n = 0 to 3). After transferring the program code, set the internal instruction RAM in the read mode using the following procedure.

- (i) Set the read mode by using (clearing to 0) the IRAMM0 bit of the internal instruction RAM mode register (IRAMM). Note that the setting of the IRAMM0 bit must not be changed before it is cleared here.
- (ii) After clearing the IRAMM0 bit of the IRAMM register to 0, read the IRAMM0 bit that has been cleared to 0 to confirm that the read mode has been set (to prevent speculative instruction execution by pipeline operation).
- (iii) Branch to the internal instruction RAM area by executing a branch instruction.

Cautions 1. After the reset signal has been cleared, the NMI input is masked by hardware. The NMI is unmasked as soon as the IRAMM0 bit of the IRAMM register is cleared in step <3> of the initialization sequence.

2. If it is necessary to confirm NMI input immediately after the reset signal has been cleared and before the internal instruction RAM is set in the read mode, read the NMIRS bit of the NMI reset status register (NRS). If this bit is set to 1, it indicates that the NMI valid edge has been input. Execute the NMI servicing routine as necessary. The NRS register is used only to check NMI input after the reset signal has been cleared and before the internal instruction RAM is set in the read mode. This register is not cleared after the reset signal has been cleared.
3. The software exception and exception trap cannot be masked. Do not execute the TRAP and DBTRAP instructions until the program code has been transferred to the internal instruction RAM.

Remarks 1. The NMIRS bit of the NRS register is also set to 1 if an NMI is input after the internal instruction RAM has been set in the read mode. In this case, execution automatically branches to the NMI servicing routine and it is not necessary to confirm the status of the NMIRS bit.

2. The NMI input mask function is valid after the reset signal has been cleared and before the internal instruction RAM is set in the read mode.
3. To write data to instruction RAM bank 0 of the internal instruction RAM in the middle of program execution, set the NP bit of the PSW to 1 to disable NMI and maskable interrupts, so that the software exception and exception trap do not occur. Clear the NP bit after the program has been rewritten, and after it has been confirmed that the IRAMM0 bit of the IRAMM register has been set to 1 and the read mode has been set.
4. NMI and maskable interrupt requests that are generated while the NP bit of the PSW is set to 1 are held pending. An NMI request is acknowledged immediately after the NP bit has been cleared to 0. A maskable interrupt is acknowledged immediately after the NP bit has been cleared to 0 if interrupts are not disabled (DI status) and the interrupt request is not cleared (by clearing the xxIFn bit of the interrupt control register (xxICn) to 0) before the NP bit is cleared to 0, and if the xxMKn bit of the interrupt control register is not set to 1. However, only one of the NMI and maskable interrupt requests is held pending for each interrupt source, and only one interrupt request is acknowledged even if the same interrupt request is generated two times or more.

CHAPTER 4 BUS CONTROL FUNCTION

The V850E/ME2 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

- 32-bit/16-bit/8-bit data bus sizing function
- 8-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)	–	–
Data bus (D16 to D31)	PDH0 to PDH15 (Port DH)	PMCDH
Address bus (A0, A1)	PAL0, PAL1 (Port AL)	PMCAL
Address bus (A16 to A25)	PAH0 to PAH9 (Port AH)	PMCAH
Chip select ($\overline{\text{CS}}_0$ to $\overline{\text{CS}}_7$, $\overline{\text{IOWR}}$, $\overline{\text{IORD}}$)	PCS0 to PCS7 (Port CS)	PMCCS
SDRAM sync control (SDCKE)	PCD0 (Port CD)	PMCCD
Bus clock (BUSCLK)	PCD1 (Port CD)	
SDRAM control ($\overline{\text{SDCAS}}$, $\overline{\text{SDRAS}}$)	PCD2, PCD3 (Port CD)	
Read/write control ($\overline{\text{LLWR}}/\overline{\text{LLBE}}/\overline{\text{LLDQM}}$, $\overline{\text{LUWR}}/\overline{\text{LUBE}}/\overline{\text{LUDQM}}$, $\overline{\text{ULWR}}/\overline{\text{ULBE}}/\overline{\text{ULDQM}}$, $\overline{\text{UUDQM}}/\overline{\text{UUBE}}/\overline{\text{UWR}}$, $\overline{\text{RD}}$, $\overline{\text{WE}}/\overline{\text{WR}}$)	PCT0 to PCT5 (Port CT)	PMCCT
Bus cycle start ($\overline{\text{BCYST}}$)	PCT7 (Port CT)	
External wait control ($\overline{\text{WAIT}}$)	PCM0 (Port CM)	PMCCM
Bus hold control ($\overline{\text{HLDRQ}}$, $\overline{\text{HLDAK}}$)	PCM2, PCM3 (Port CM)	
SDRAM refresh control ($\overline{\text{REFRQ}}$)	PCM4 (Port CM)	
Self-refresh control ($\overline{\text{SELFREF}}$)	PCM5 (Port CM)	

Remark When the system is reset, each bus control pin becomes unconditionally valid. (However, D16 to D31, $\overline{\text{ULWR}}/\overline{\text{ULBE}}/\overline{\text{ULDQM}}$, and $\overline{\text{UWR}}/\overline{\text{UUBE}}/\overline{\text{UUDQM}}$ are valid only in 32-bit mode.)

4.2.1 Pin status during internal instruction RAM, internal data RAM, and peripheral I/O access

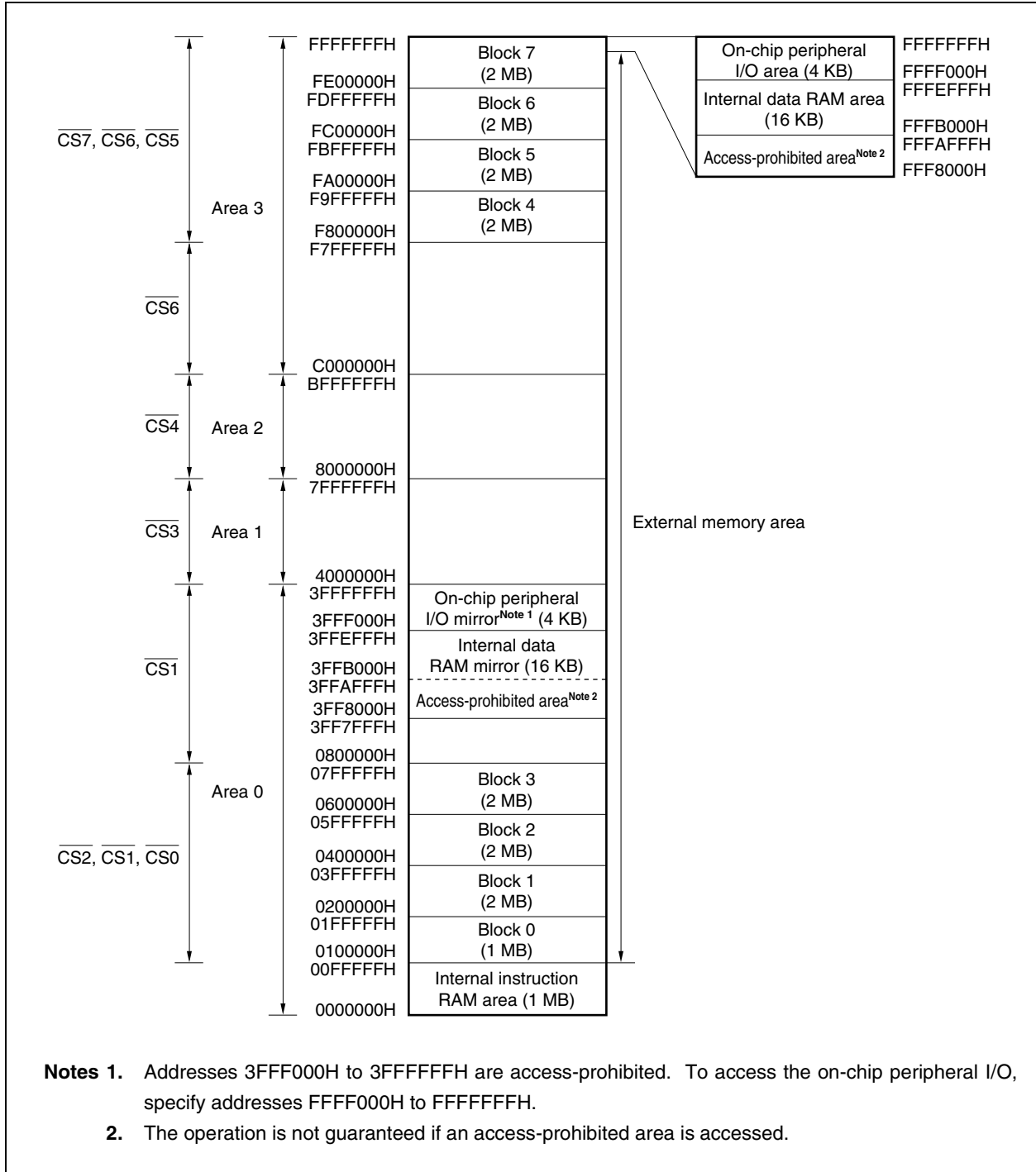
While accessing internal instruction RAM (in the read mode), internal data RAM, and peripheral I/O, the address bus outputs low level, and the data bus outputs nothing and enters the high-impedance state. The external bus control signals become inactive.

★ When the internal instruction RAM is accessed (in the write mode), the address bus and data bus output data. The external bus control signals other than \overline{UUWR} , \overline{ULWR} , \overline{LUWR} , \overline{LLWR} , and \overline{WR} become active. If output of the \overline{IOWR} signal is enabled by setting the IOEN bit of the bus cycle period control register (BCP) to 1, the \overline{IOWR} signal becomes active.

4.3 Memory Block Function

The 256 MB memory space is divided into four areas including seven 2 MB memory blocks and one 1 MB memory block.

The area that can be used as program area is the 64 MB space of addresses 0000000H to 3FFFFFFFH.



4.3.1 Chip select control function

Each memory block can be divided by chip area select 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 select control registers 0, 1 (CSC0, CSC1)

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

If different chip select signal outputs are set to the same block, the priority order is controlled as follows.

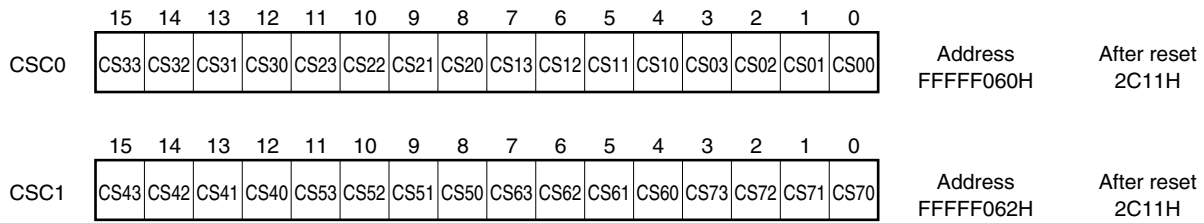
CSC0: $\overline{CS0} > \overline{CS2} > \overline{CS1}$

CSC1: $\overline{CS7} > \overline{CS5} > \overline{CS6}$

If both the CS0n and CS2n bits of the CSC0 register are set to 0, $\overline{CS1}$ is output to the corresponding block (n = 0 to 3).

Similarly, if both the CS5n and CS7n bits of the CSC1 register are set to 0, $\overline{CS6}$ is output to the corresponding block (n = 0 to 3).

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



Bit position	Bit name	Function																																										
15 to 0	CSnm (n = 0 to 7) (m = 0 to 3)	<p>Chip select is enabled by setting the CSnm bit to 1.</p> <table border="1"> <thead> <tr> <th>CSnm</th> <th>CS operation</th> </tr> </thead> <tbody> <tr> <td>CS00</td> <td>$\overline{CS0}$ output during block 0 access</td> </tr> <tr> <td>CS01</td> <td>$\overline{CS0}$ output during block 1 access.</td> </tr> <tr> <td>CS02</td> <td>$\overline{CS0}$ output during block 2 access.</td> </tr> <tr> <td>CS03</td> <td>$\overline{CS0}$ output during block 3 access.</td> </tr> <tr> <td>CS10 to CS13</td> <td>Setting has no meaning.</td> </tr> <tr> <td>CS20</td> <td>$\overline{CS2}$ output during block 0 access.</td> </tr> <tr> <td>CS21</td> <td>$\overline{CS2}$ output during block 1 access.</td> </tr> <tr> <td>CS22</td> <td>$\overline{CS2}$ output during block 2 access.</td> </tr> <tr> <td>CS23</td> <td>$\overline{CS2}$ output during block 3 access.</td> </tr> <tr> <td>CS30 to CS33</td> <td>Setting has no meaning.</td> </tr> <tr> <td>CS40 to CS43</td> <td>Setting has no meaning.</td> </tr> <tr> <td>CS50</td> <td>$\overline{CS5}$ output during block 7 access.</td> </tr> <tr> <td>CS51</td> <td>$\overline{CS5}$ output during block 6 access.</td> </tr> <tr> <td>CS52</td> <td>$\overline{CS5}$ output during block 5 access.</td> </tr> <tr> <td>CS53</td> <td>$\overline{CS5}$ output during block 4 access.</td> </tr> <tr> <td>CS60 to CS63</td> <td>Setting has no meaning.</td> </tr> <tr> <td>CS70</td> <td>$\overline{CS7}$ output during block 7 access.</td> </tr> <tr> <td>CS71</td> <td>$\overline{CS7}$ output during block 6 access.</td> </tr> <tr> <td>CS72</td> <td>$\overline{CS7}$ output during block 5 access.</td> </tr> <tr> <td>CS73</td> <td>$\overline{CS7}$ output during block 4 access.</td> </tr> </tbody> </table>	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.	CS10 to CS13	Setting has no meaning.	CS20	$\overline{CS2}$ output during block 0 access.	CS21	$\overline{CS2}$ output during block 1 access.	CS22	$\overline{CS2}$ output during block 2 access.	CS23	$\overline{CS2}$ output during block 3 access.	CS30 to CS33	Setting has no meaning.	CS40 to CS43	Setting has no meaning.	CS50	$\overline{CS5}$ output during block 7 access.	CS51	$\overline{CS5}$ output during block 6 access.	CS52	$\overline{CS5}$ output during block 5 access.	CS53	$\overline{CS5}$ output during block 4 access.	CS60 to CS63	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.
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.																																											
CS10 to CS13	Setting has no meaning.																																											
CS20	$\overline{CS2}$ output during block 0 access.																																											
CS21	$\overline{CS2}$ output during block 1 access.																																											
CS22	$\overline{CS2}$ output during block 2 access.																																											
CS23	$\overline{CS2}$ output during block 3 access.																																											
CS30 to CS33	Setting has no meaning.																																											
CS40 to CS43	Setting has no meaning.																																											
CS50	$\overline{CS5}$ output during block 7 access.																																											
CS51	$\overline{CS5}$ output during block 6 access.																																											
CS52	$\overline{CS5}$ output during block 5 access.																																											
CS53	$\overline{CS5}$ output during block 4 access.																																											
CS60 to CS63	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.																																											

The following diagram shows the \overline{CS} signal that is enabled for area 0 when the CSC0 register is set to 0F03H.

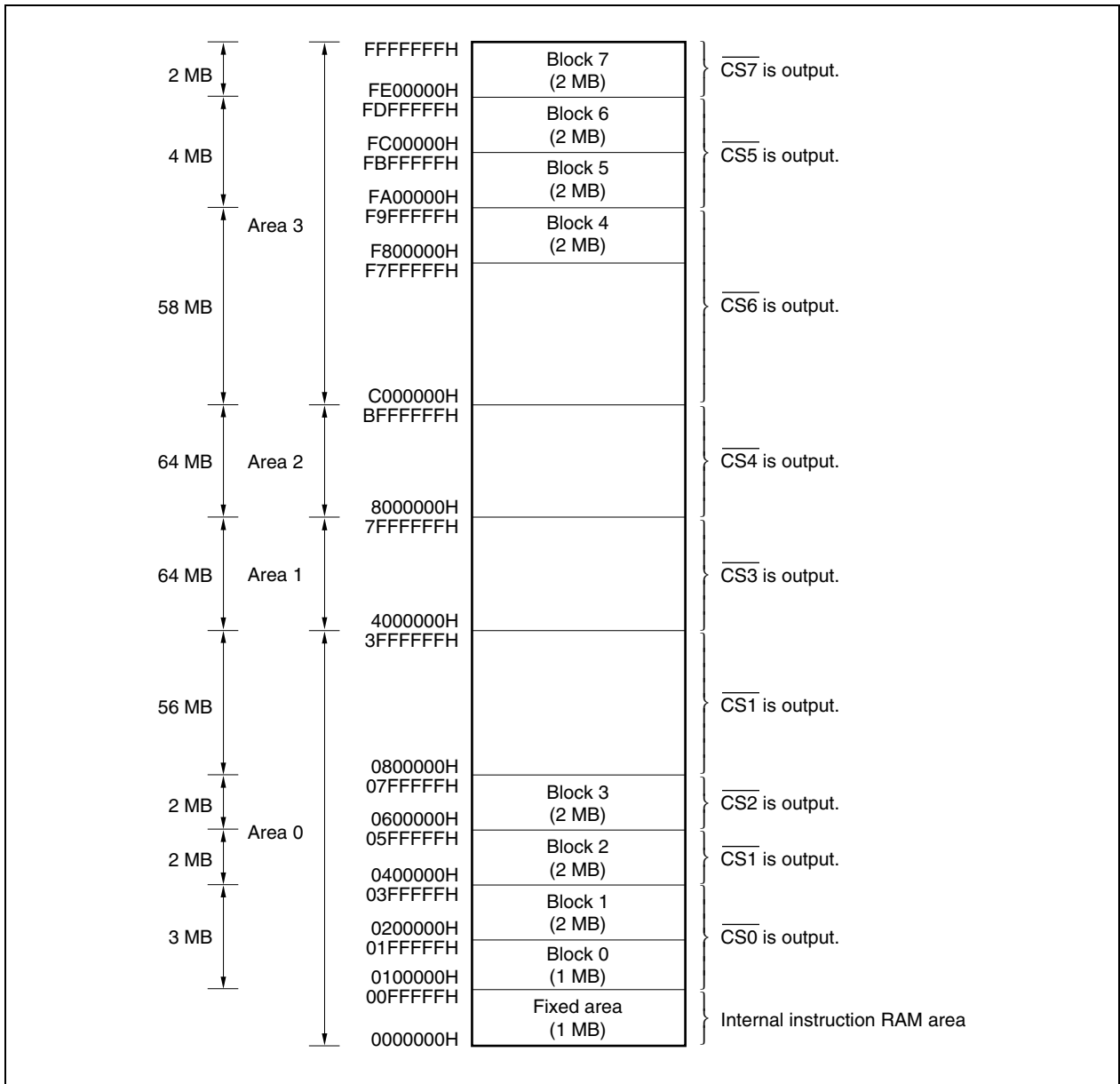
When the CSC0 register is set to 0F03H, $\overline{CS0}$ and $\overline{CS2}$ are output to block 0 and block 1, but since $\overline{CS0}$ has priority over $\overline{CS2}$, $\overline{CS0}$ is output if the addresses of block 0 and block 1 are accessed.

If the addresses of block 2 and block 3 are accessed, $\overline{CS2}$ is output.

If the addresses of area 0 other than blocks 0 to 3 are accessed, $\overline{CS1}$ is output.

The following shows an example when the CSC0 register is set to 0803H and the CSC1 register is set to 0601H.

Figure 4-1. Example When CSC0 Register Is Set to 0803H and CSC1 Register Is Set to 0601H



4.4 Bus Cycle Type Control Function

In the V850E/ME2, 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 and 1 (BCT0 and BCT1).

4.5 Bus Access

★ 4.5.1 Number of access clocks

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

Resource (Bus Width)	Bus Cycle Configuration	Instruction Fetch	Operand Data Access	
			Read	Write
Internal instruction RAM (32 bits)		1	1	2 ^{Note}
Internal data RAM (32 bits)		–	1	

Note When the internal instruction RAM is accessed (in the write mode), programmable waits, address setup waits, and idle states can be inserted in the CS0 space.

If none of the above states is set, the instruction RAM is accessed using a $\times 2$ BUSCLK frequency clock.

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 local bus sizing control register (LBS).

(1) Local bus sizing control register (LBS)

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

Cautions 1. Write to the LBS 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 LBS 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 signal shown below becomes active.

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

3. When the data bus width is specified as 16 bits, only the signals shown below become active.

LLWR, LUWR: 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	Address	After reset ^{Note}
LBS	LB71	LB70	LB61	LB60	LB51	LB50	LB41	LB40	LB31	LB30	LB21	LB20	LB11	LB10	LB01	LB00	FFFFF48EH	5555H/ AAAAH
<u>CSn</u> signal	<u>CS7</u>		<u>CS6</u>		<u>CS5</u>		<u>CS4</u>		<u>CS3</u>		<u>CS2</u>		<u>CS1</u>		<u>CS0</u>			

Note When in 32-bit mode: AAAAH
 When in 16-bit mode: 5555H
 For details of 32-bit mode and 16-bit mode, refer to **3.3.1 Operating modes**.

Bit position	Bit name	Function															
15 to 0	LBn1, LBn0	Sets the data bus width of the CSn space. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">LBn1</th> <th style="width: 10%;">LBn0</th> <th style="width: 80%;">Data bus width of CSn space</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">8 bits</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">16 bits</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">32 bits</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">32 bits</td> </tr> </tbody> </table>	LBn1	LBn0	Data bus width of CSn space	0	0	8 bits	0	1	16 bits	1	0	32 bits	1	1	32 bits
LBn1	LBn0	Data bus width of CSn space															
0	0	8 bits															
0	1	16 bits															
1	0	32 bits															
1	1	32 bits															

Remark n = 0 to 7

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 signals ($\overline{CS0}$ to $\overline{CS7}$). 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 instruction RAM area
- Internal data RAM area
- Area same as internal data RAM area of addresses 3FF8000H to 3FFBFFFH
- Program area for external memory

(1) Endian configuration register (BEC)

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

Be sure to clear bits 15, 13, 11, 9, 7, 5, 3, and 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.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
BEC	0	BE70	0	BE60	0	BE50	0	BE40	0	BE30	0	BE20	0	BE10	0	BE00	Address FFFFF068H	After reset 0000H
\overline{CSn} signal	┌───┐		┌───┐		┌───┐		┌───┐		┌───┐		┌───┐		┌───┐		┌───┐			
	$\overline{CS7}$		$\overline{CS6}$		$\overline{CS5}$		$\overline{CS4}$		$\overline{CS3}$		$\overline{CS2}$		$\overline{CS1}$		$\overline{CS0}$			

Bit position	Bit name	Function
14, 12, 10, 8, 6, 4, 2, 0	BE _n 0	Specifies the endian method. 0: Little endian method 1: Big endian method

Remark n = 0 to 7

Figure 4-2. Big Endian Addresses Within Word

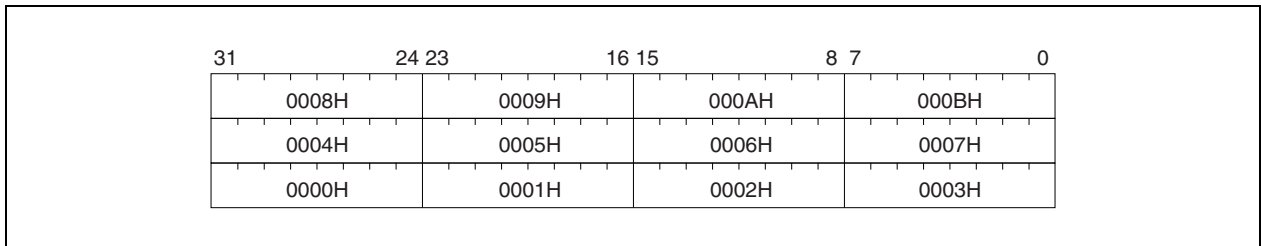
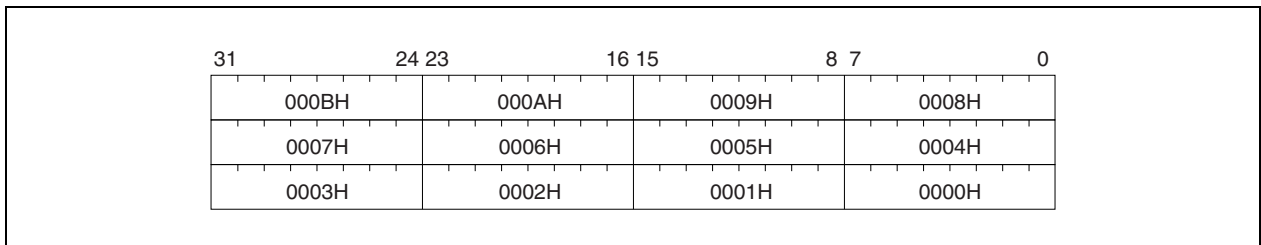


Figure 4-3. Little Endian Addresses Within Word



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 options 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

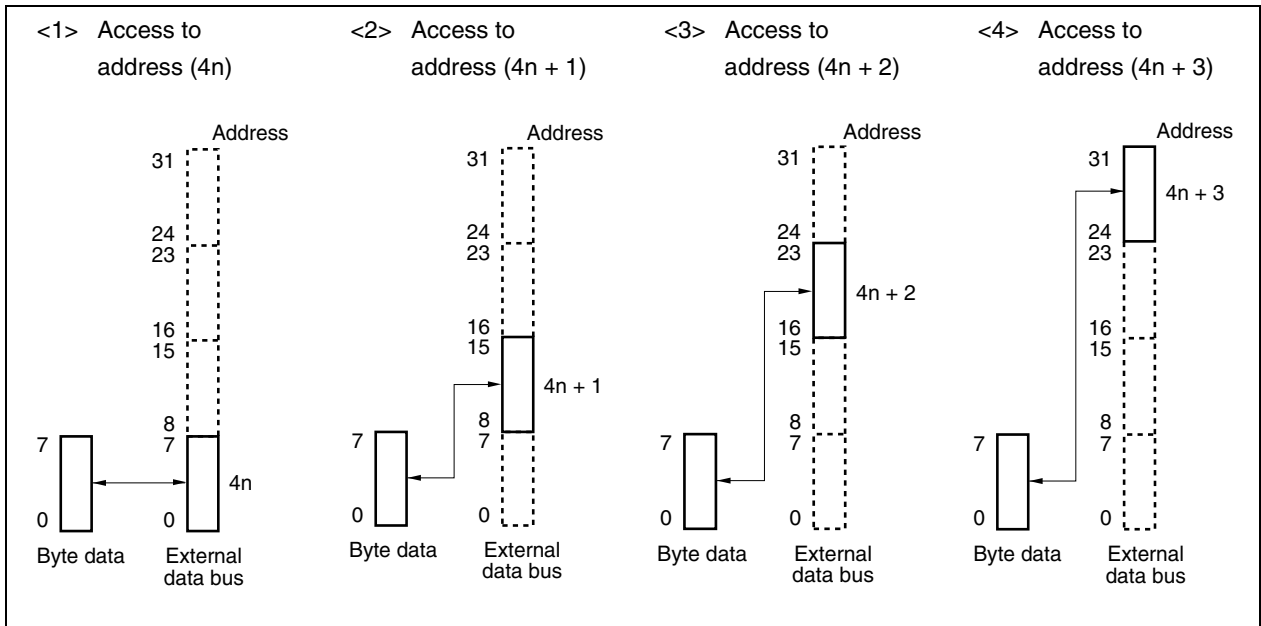
For variables located in the big endian space, a quasi directive that secures an area of other than byte size (.hword, .word, .float, .shword) cannot be used.

4.5.5 Bus width

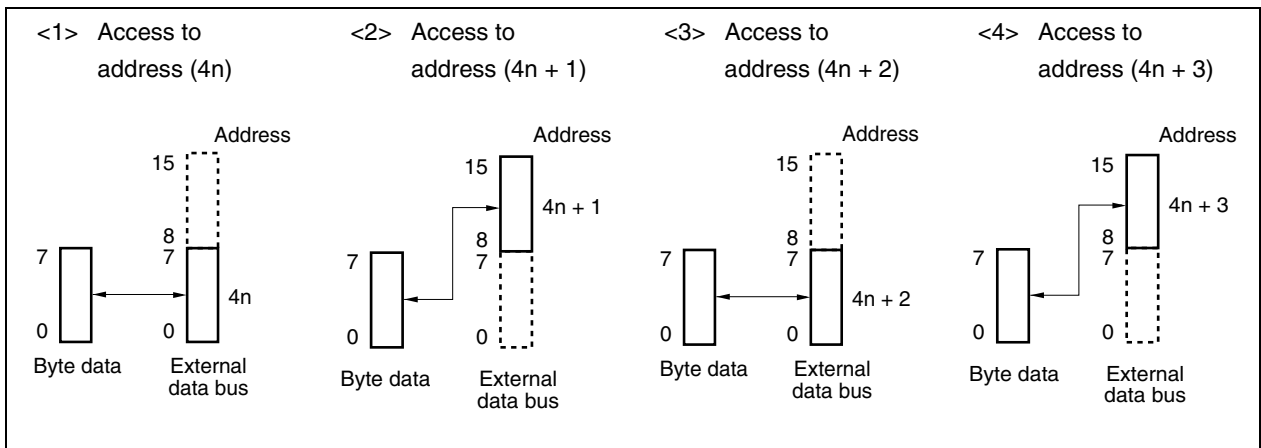
The V850E/ME2 accesses peripheral I/O and external memory in 8-bit, 16-bit, or 32-bit units. The following shows the operation for each type of access. All data is accessed in order starting from the lower order side.

(1) Byte access (8 bits)

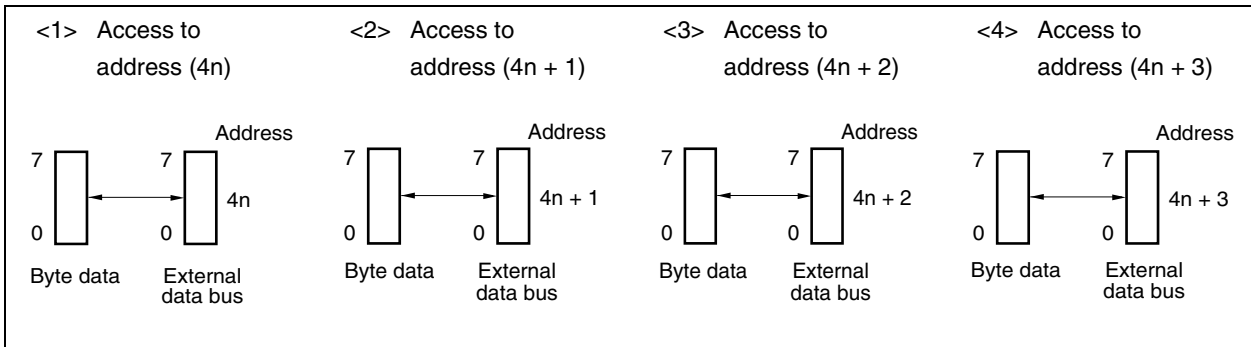
(a) When the data bus width is 32 bits (little endian)



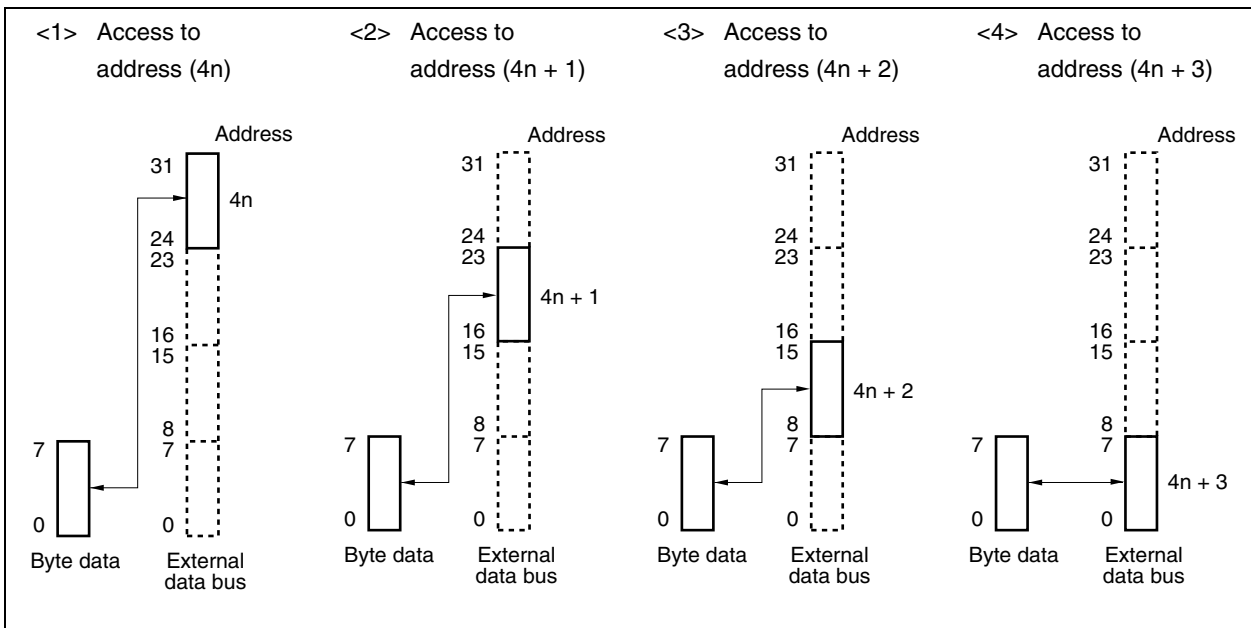
(b) When the data bus width is 16 bits (little endian)



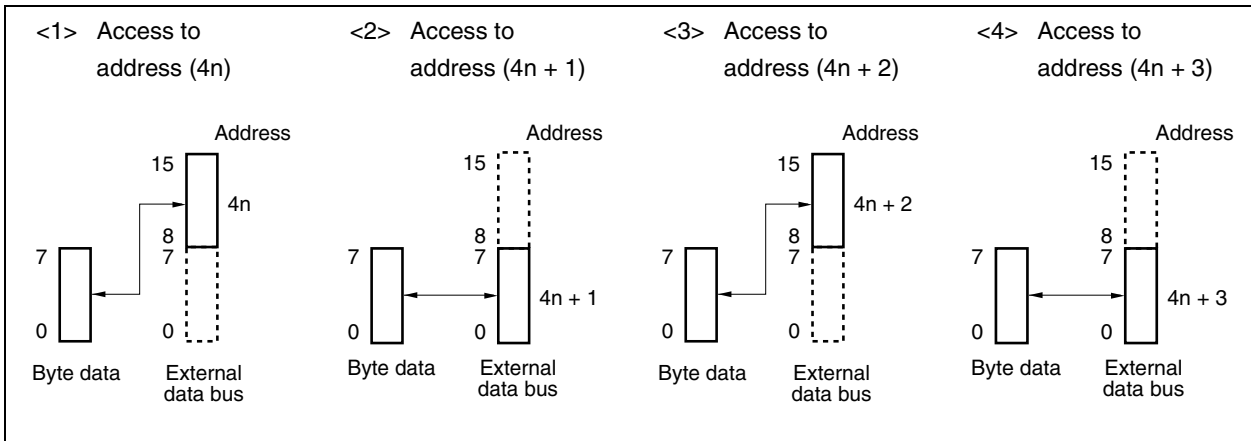
(c) When the data bus width is 8 bits (little endian)



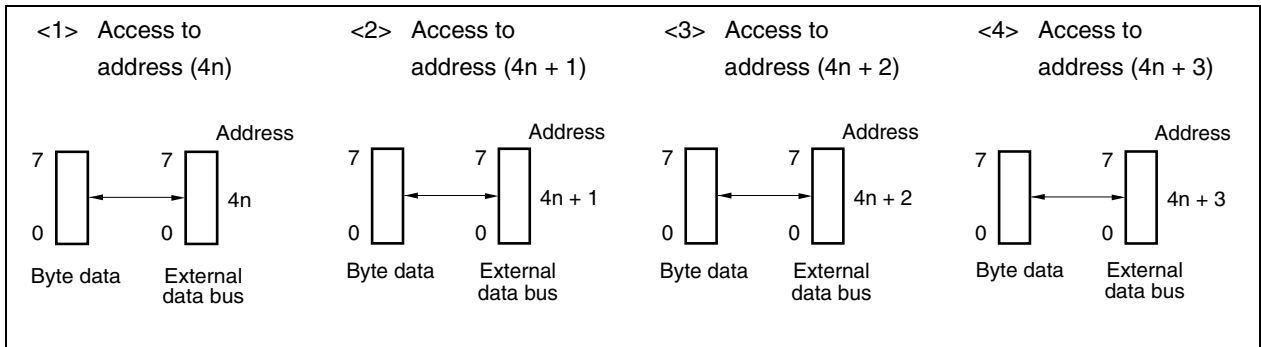
(d) When the data bus width is 32 bits (big endian)



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

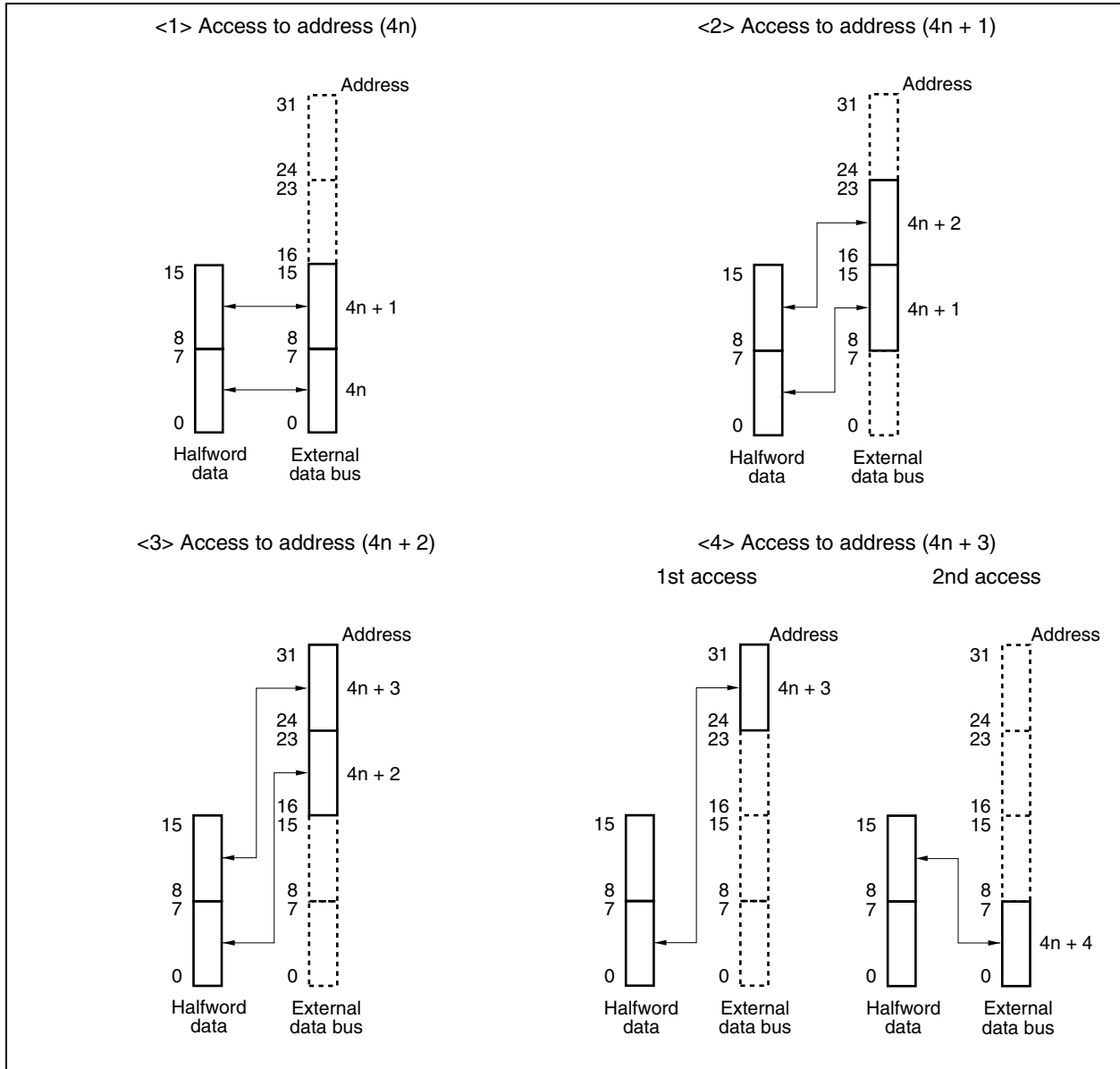


(f) When the data bus width is 8 bits (big endian)

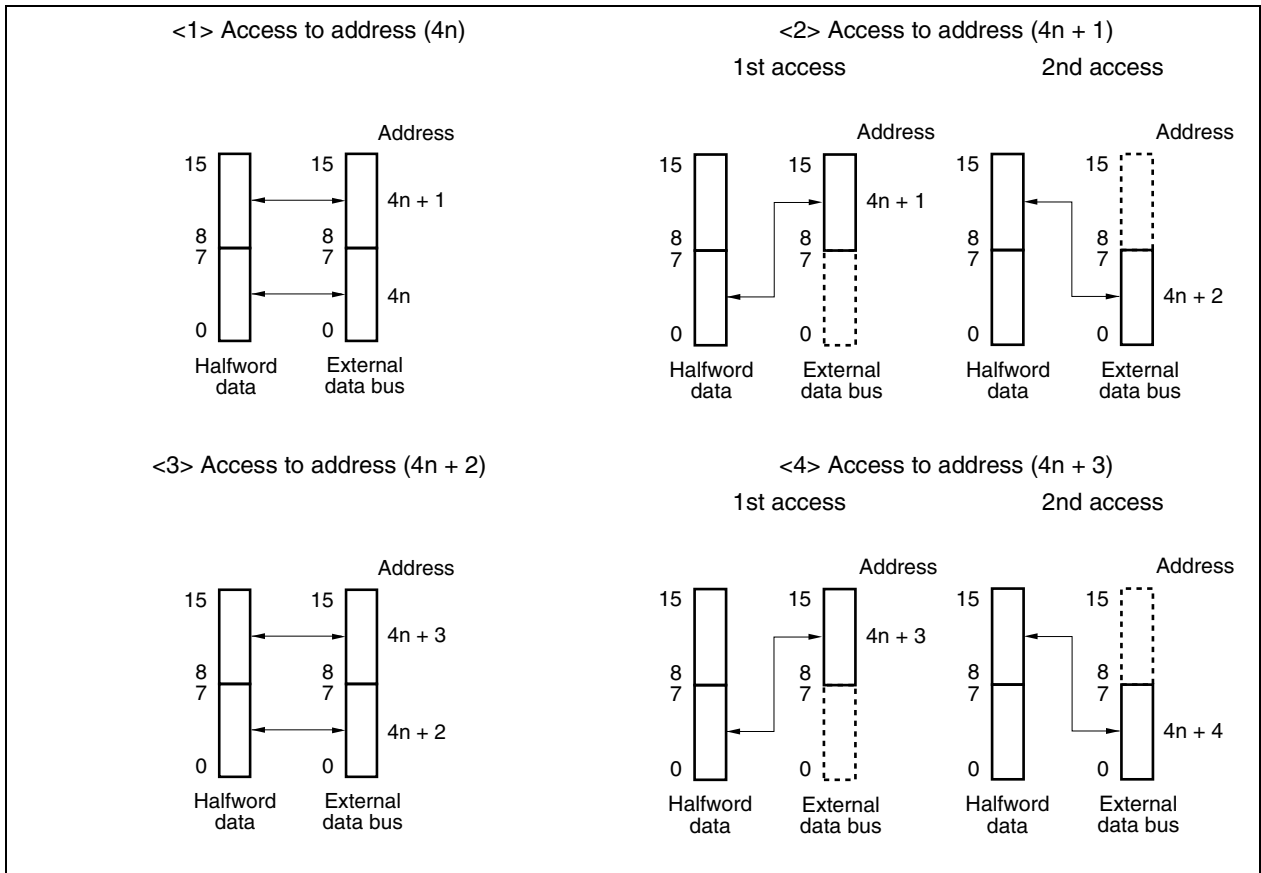


(2) Halfword access (16 bits)

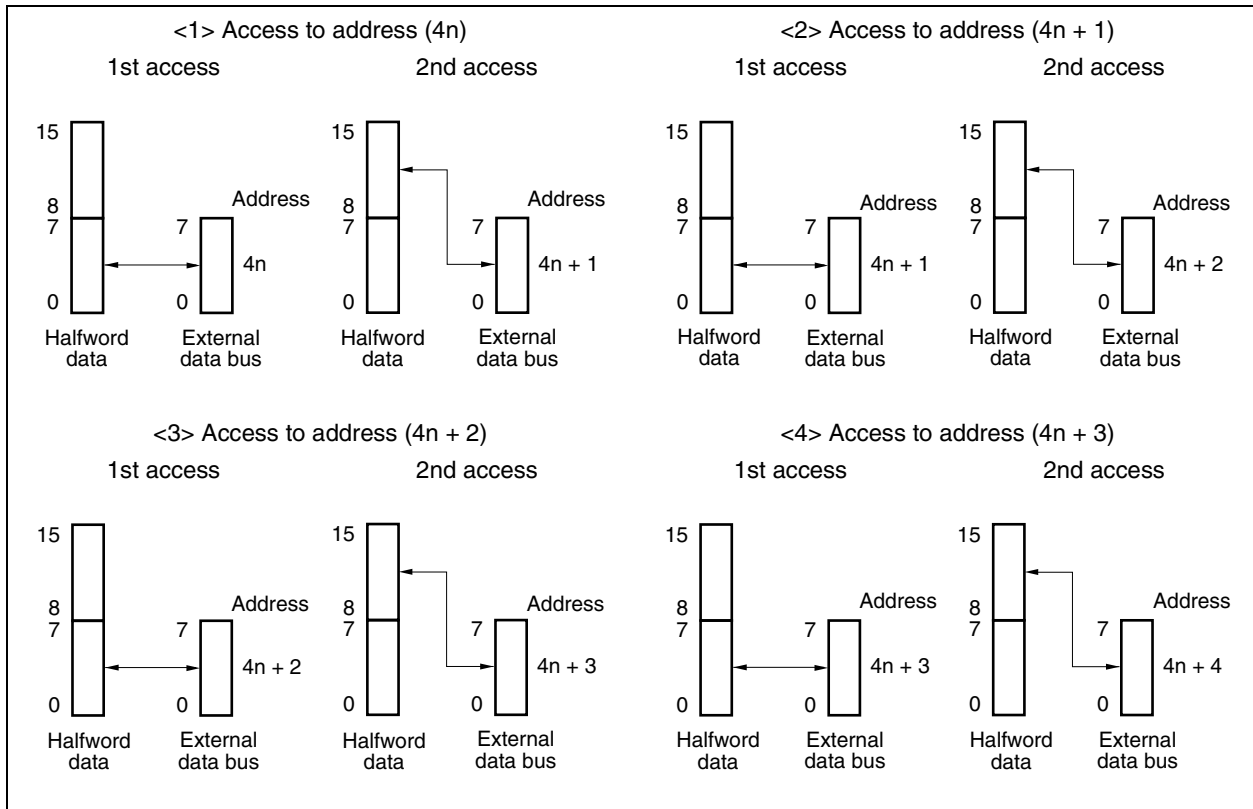
(a) When the data bus width is 32 bits (little endian)



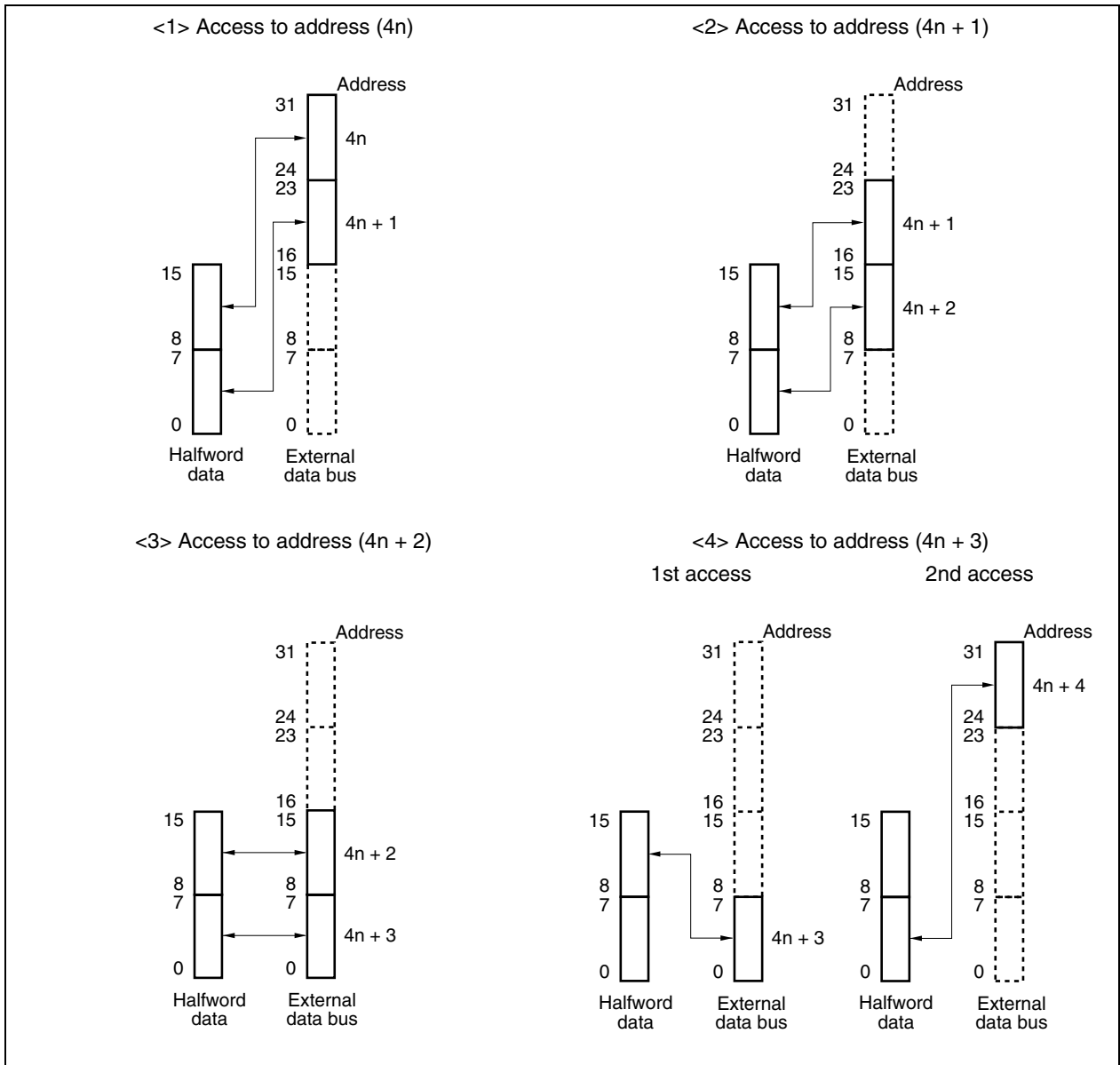
(b) When the data bus width is 16 bits (little endian)



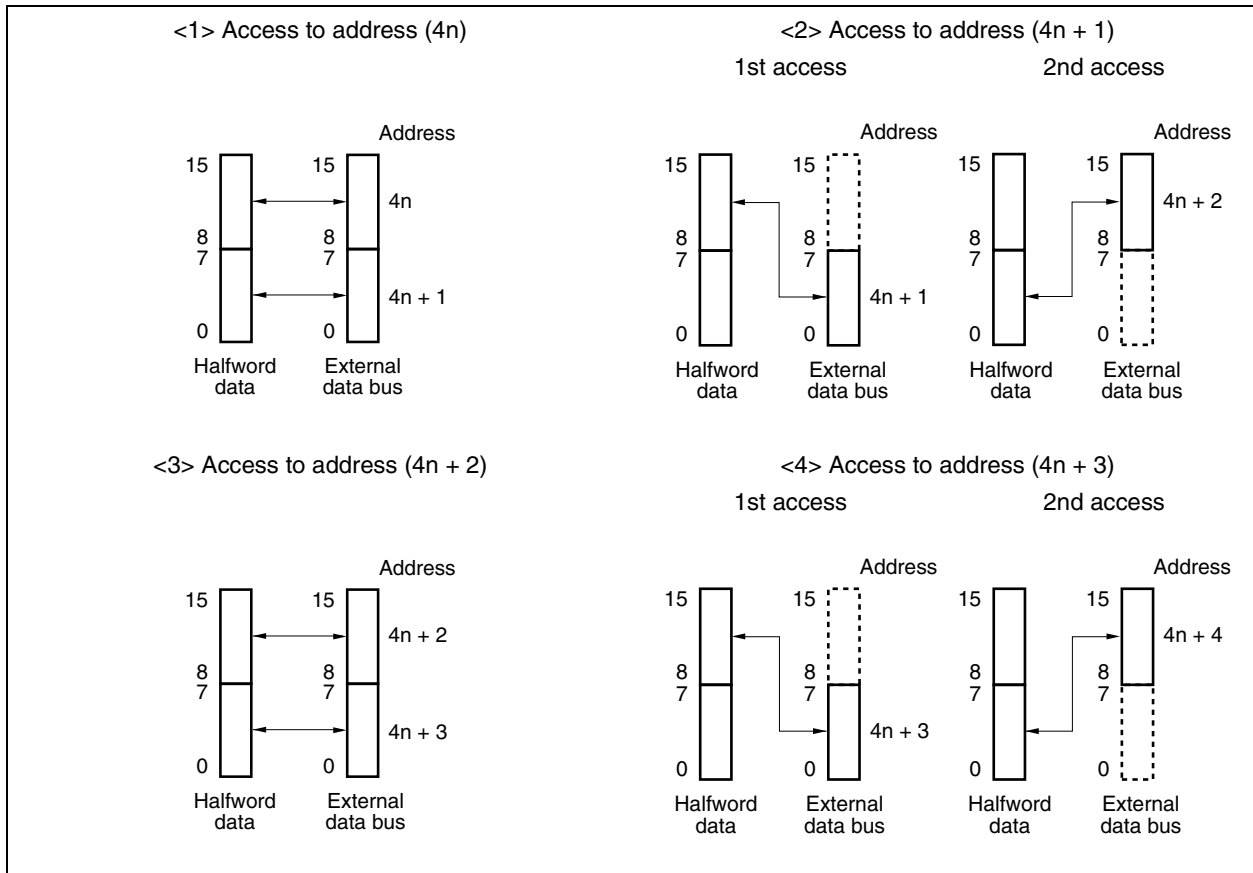
(c) When the data bus width is 8 bits (little endian)



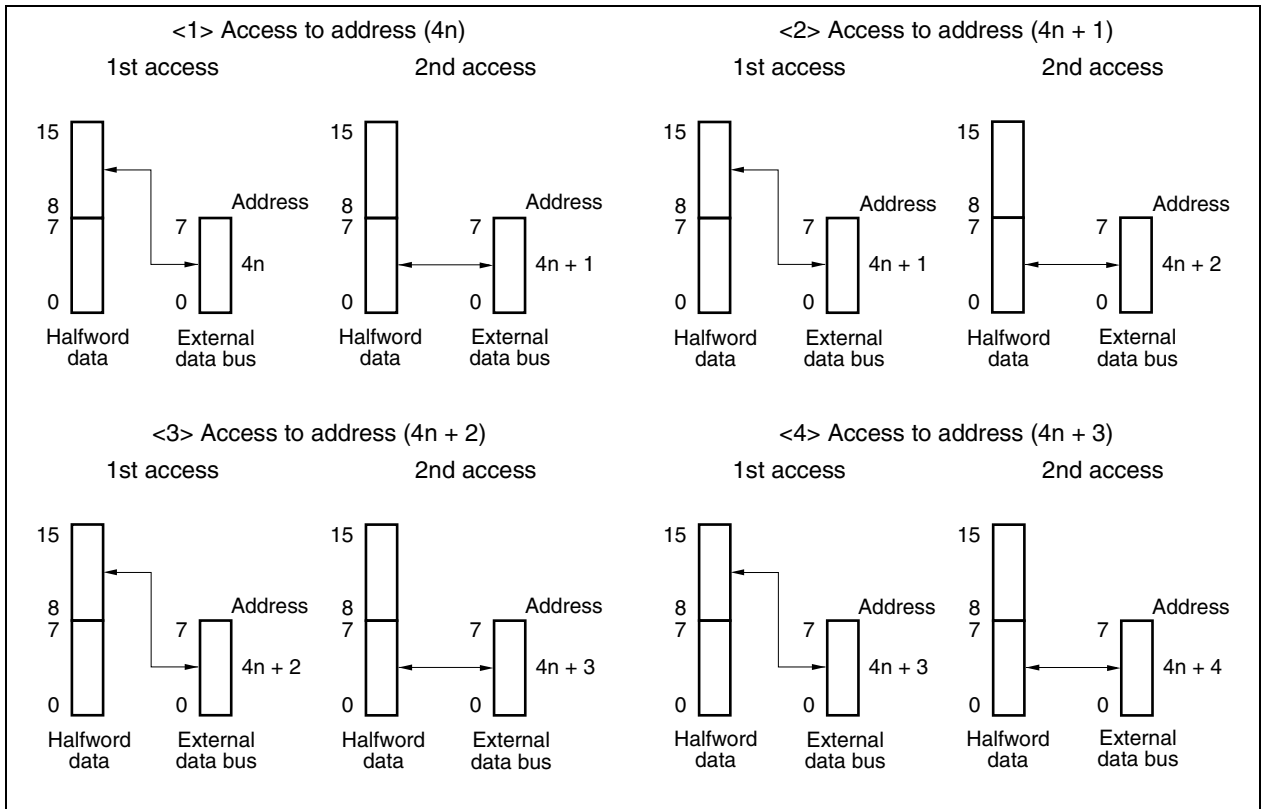
(d) When the data bus width is 32 bits (big endian)



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

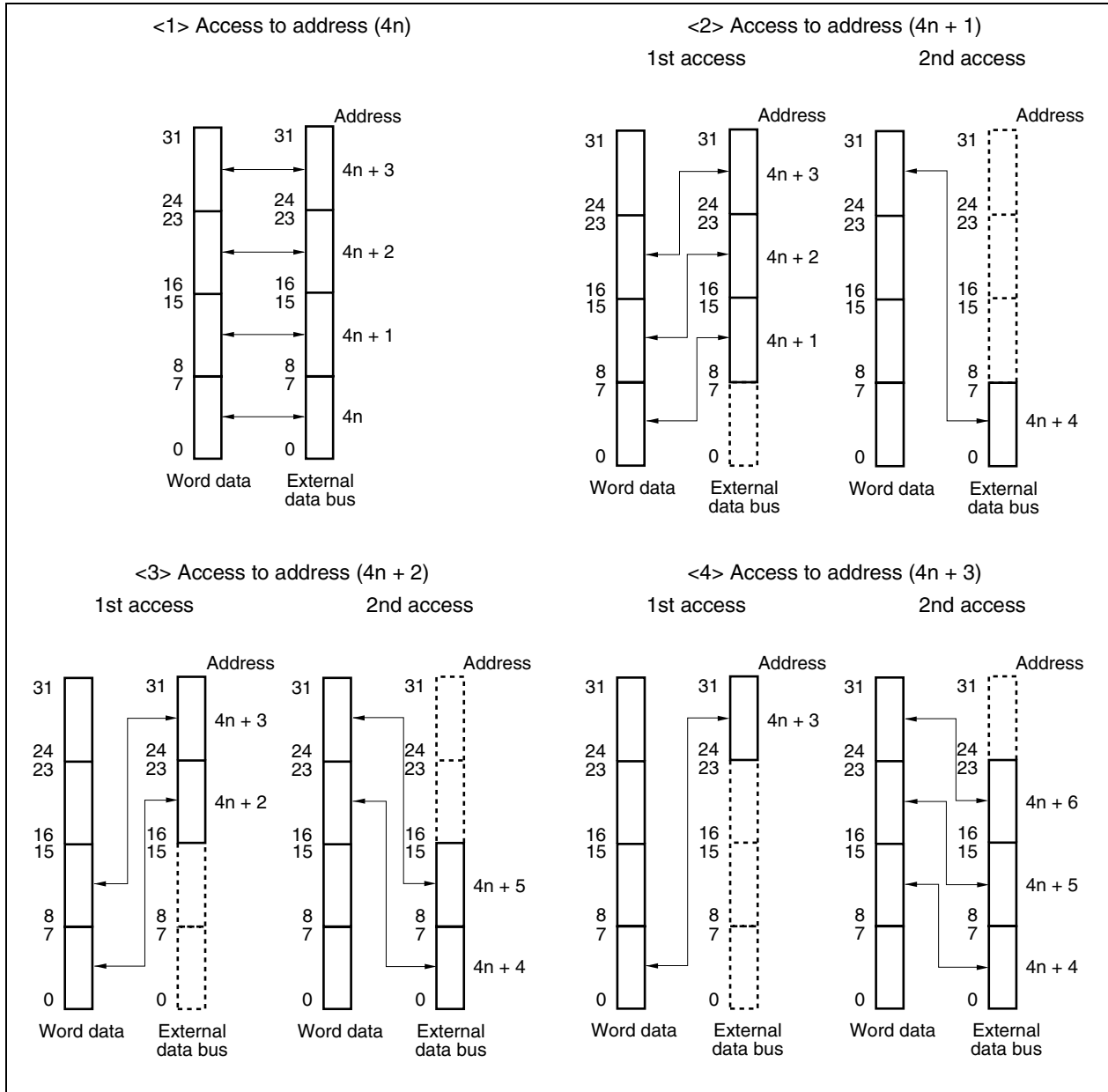


(f) When the data bus width is 8 bits (big endian)

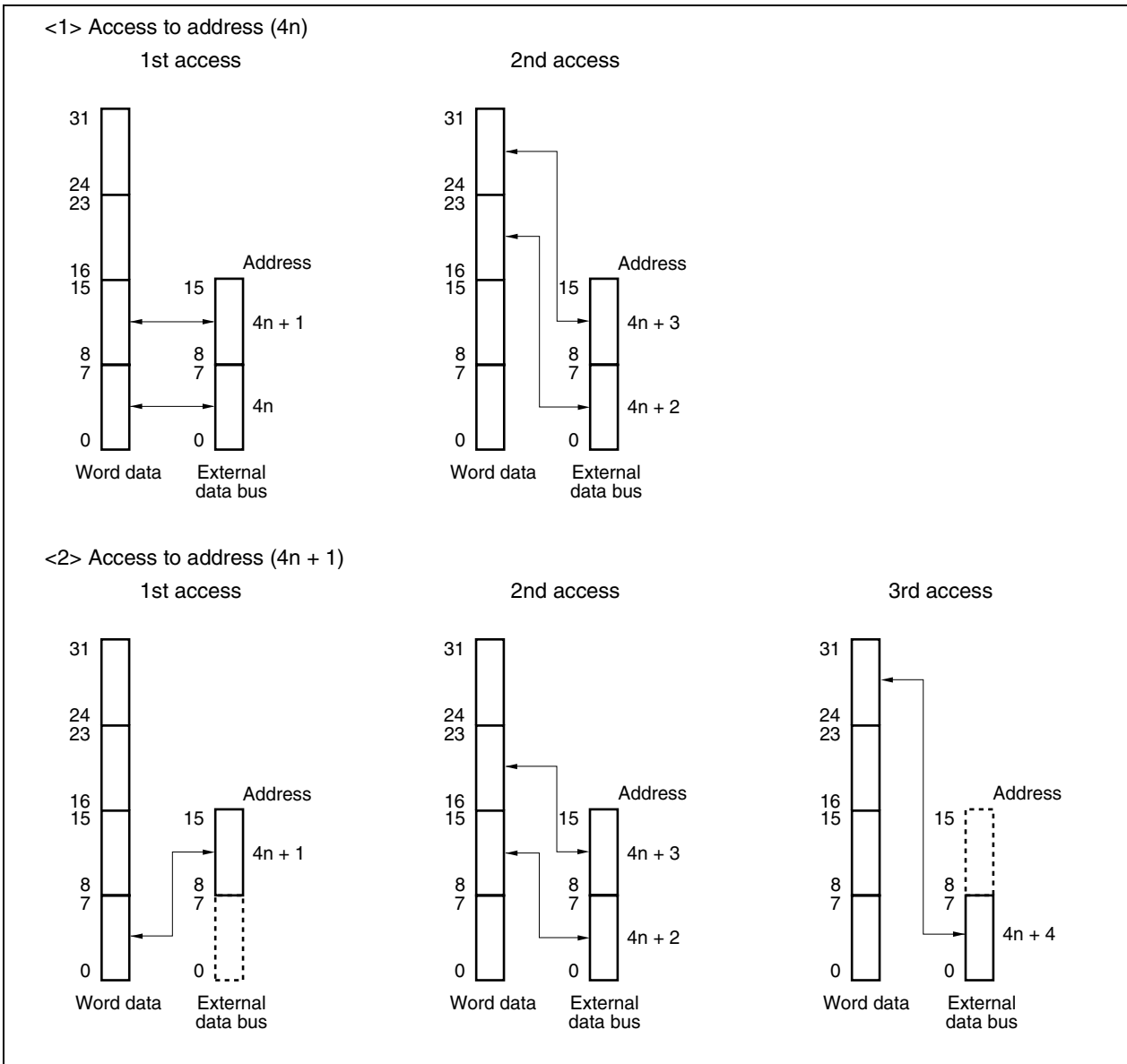


(3) Word access (32 bits)

(a) When the data bus width is 32 bits (little endian)



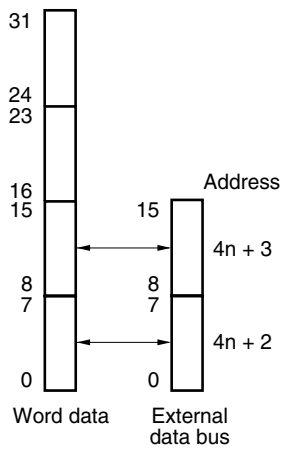
(b) When the data bus width is 16 bits (little endian) (1/2)



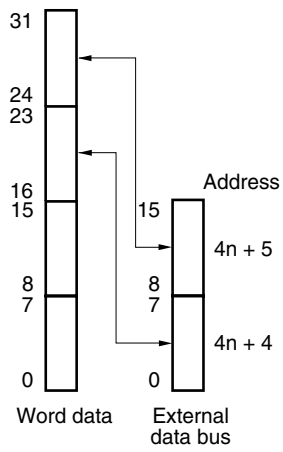
(b) When the data bus width is 16 bits (little endian) (2/2)

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

1st access

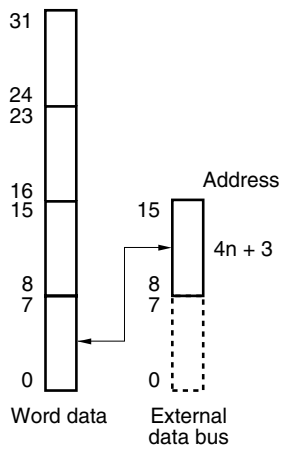


2nd access

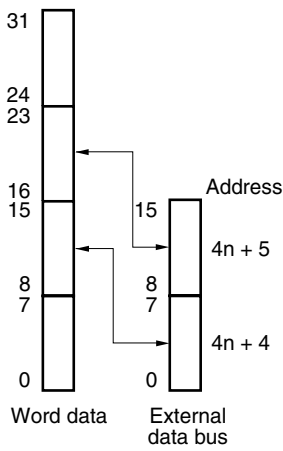


<4> Access to address $(4n + 3)$

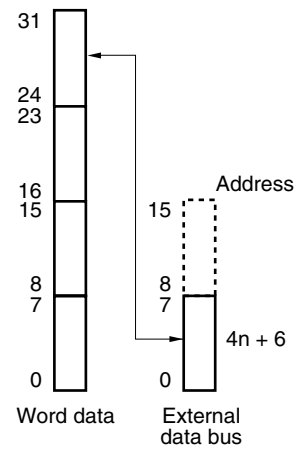
1st access



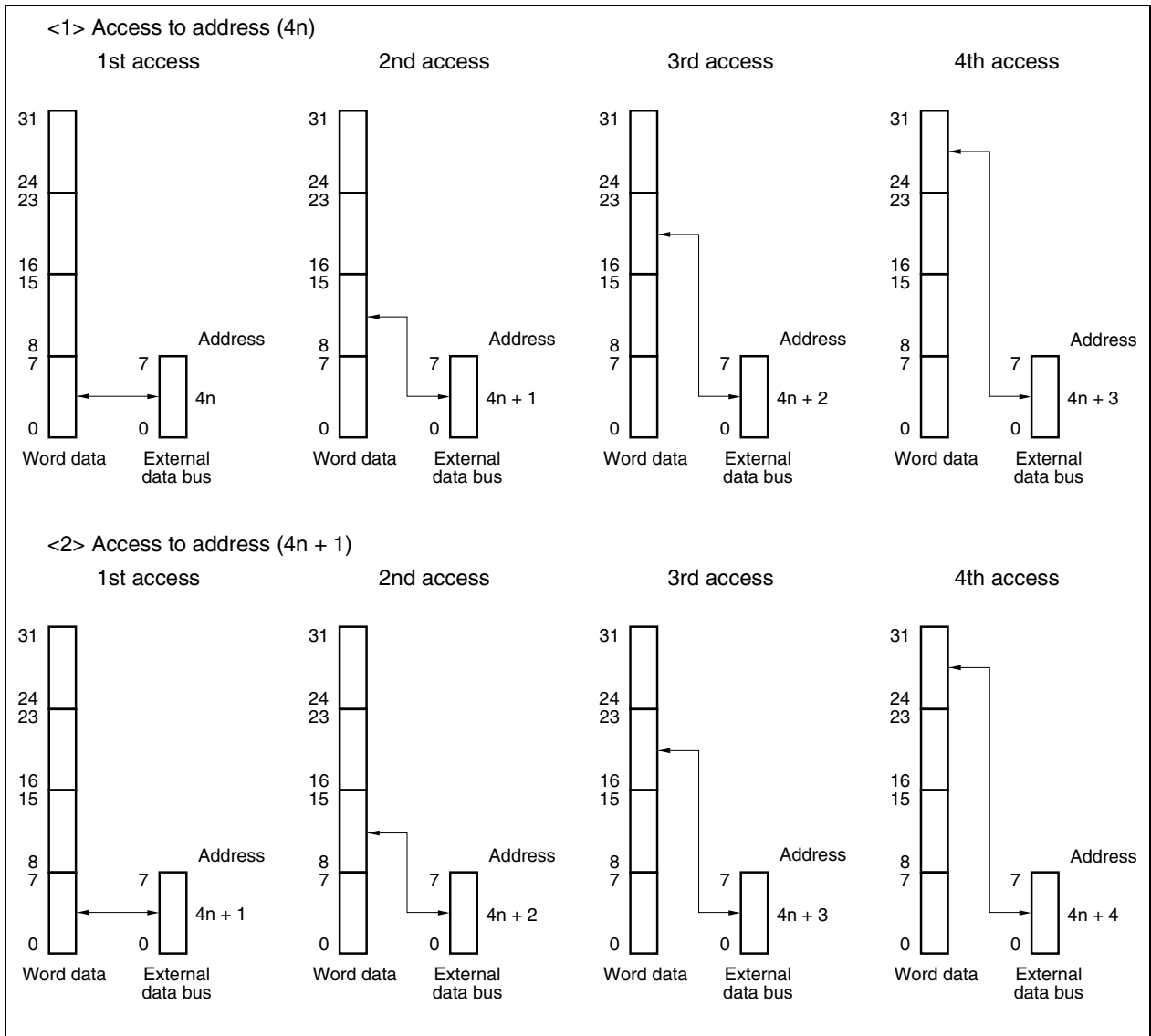
2nd access



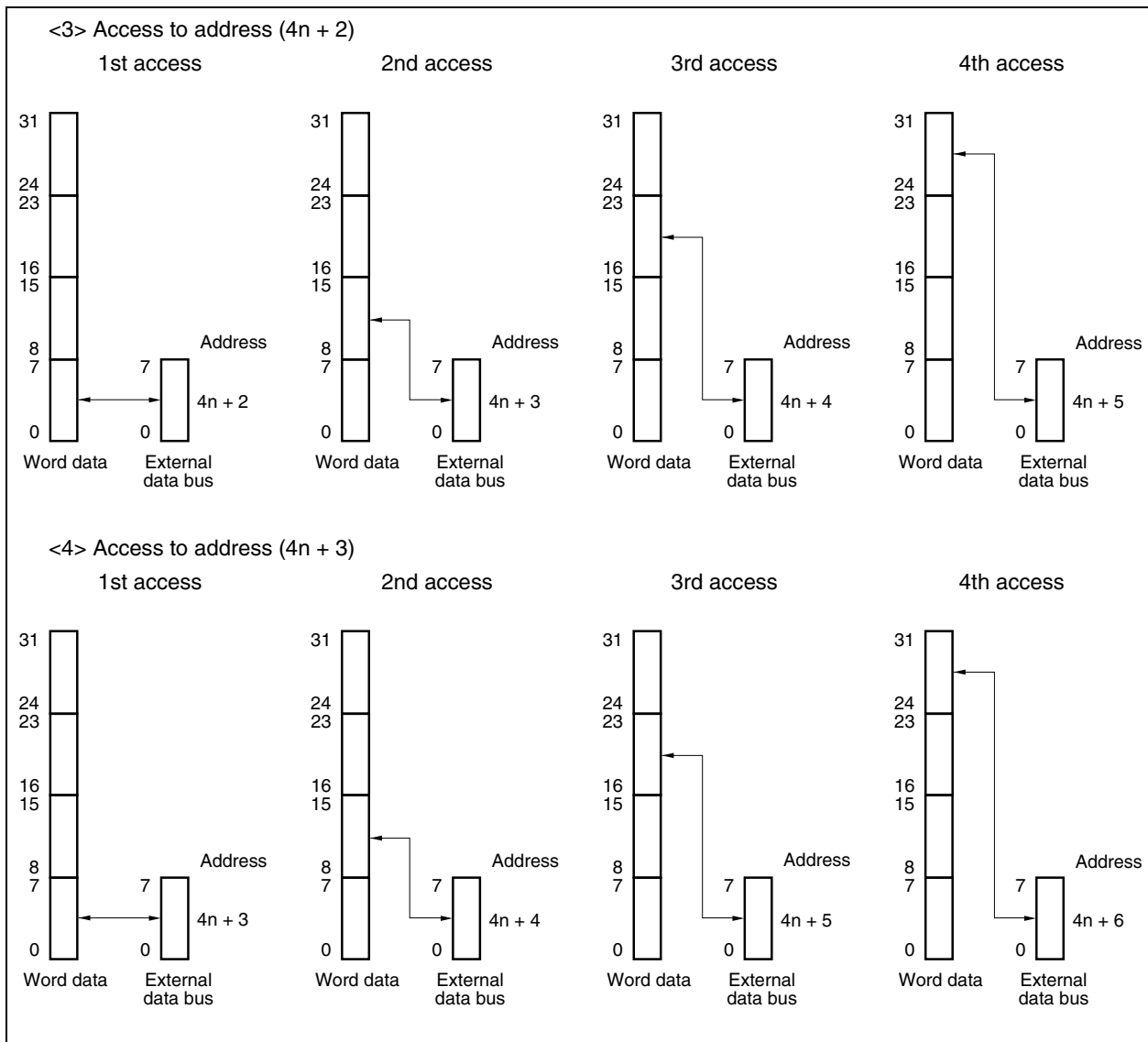
3rd access



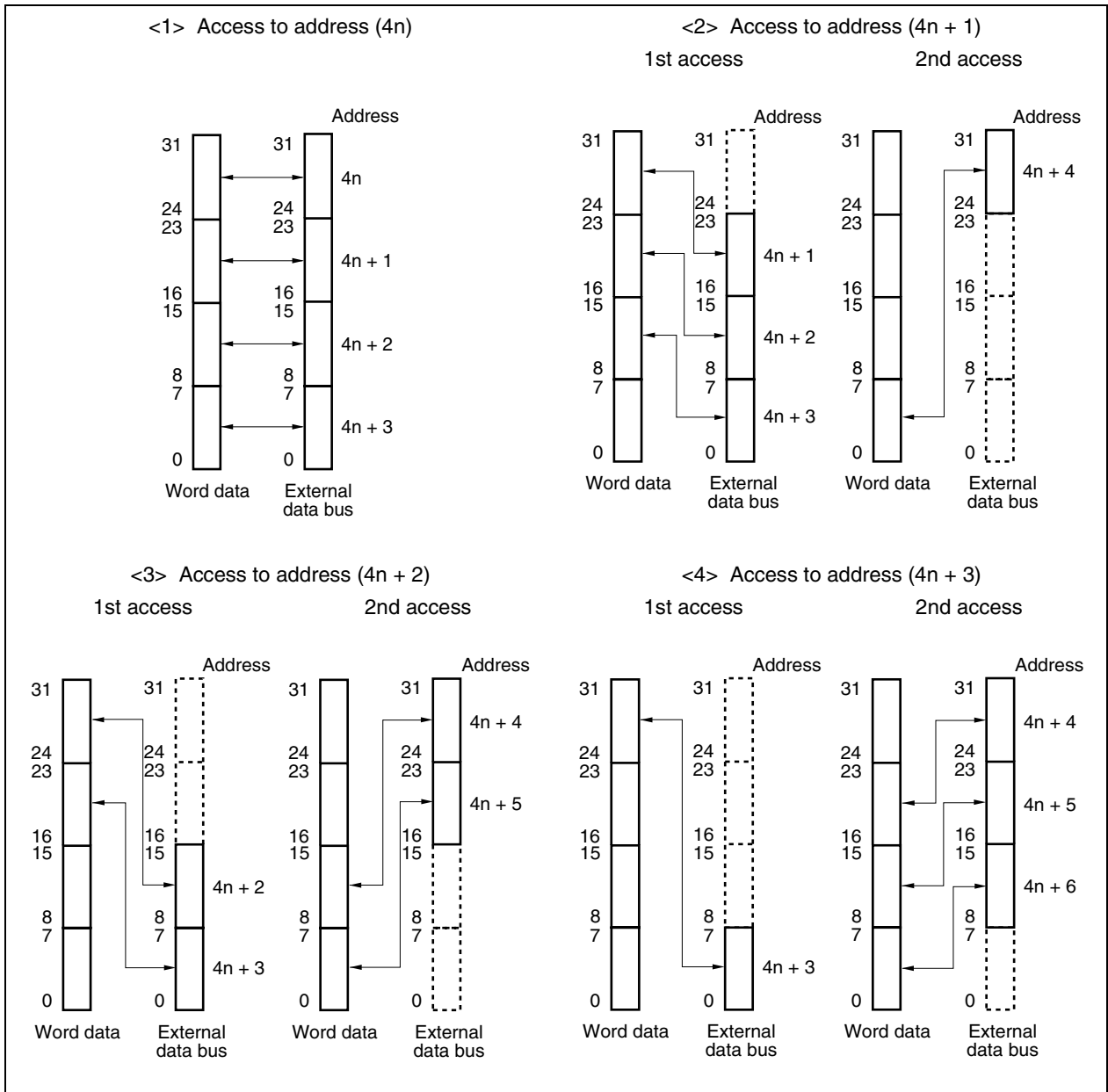
(c) When the data bus width is 8 bits (little endian) (1/2)



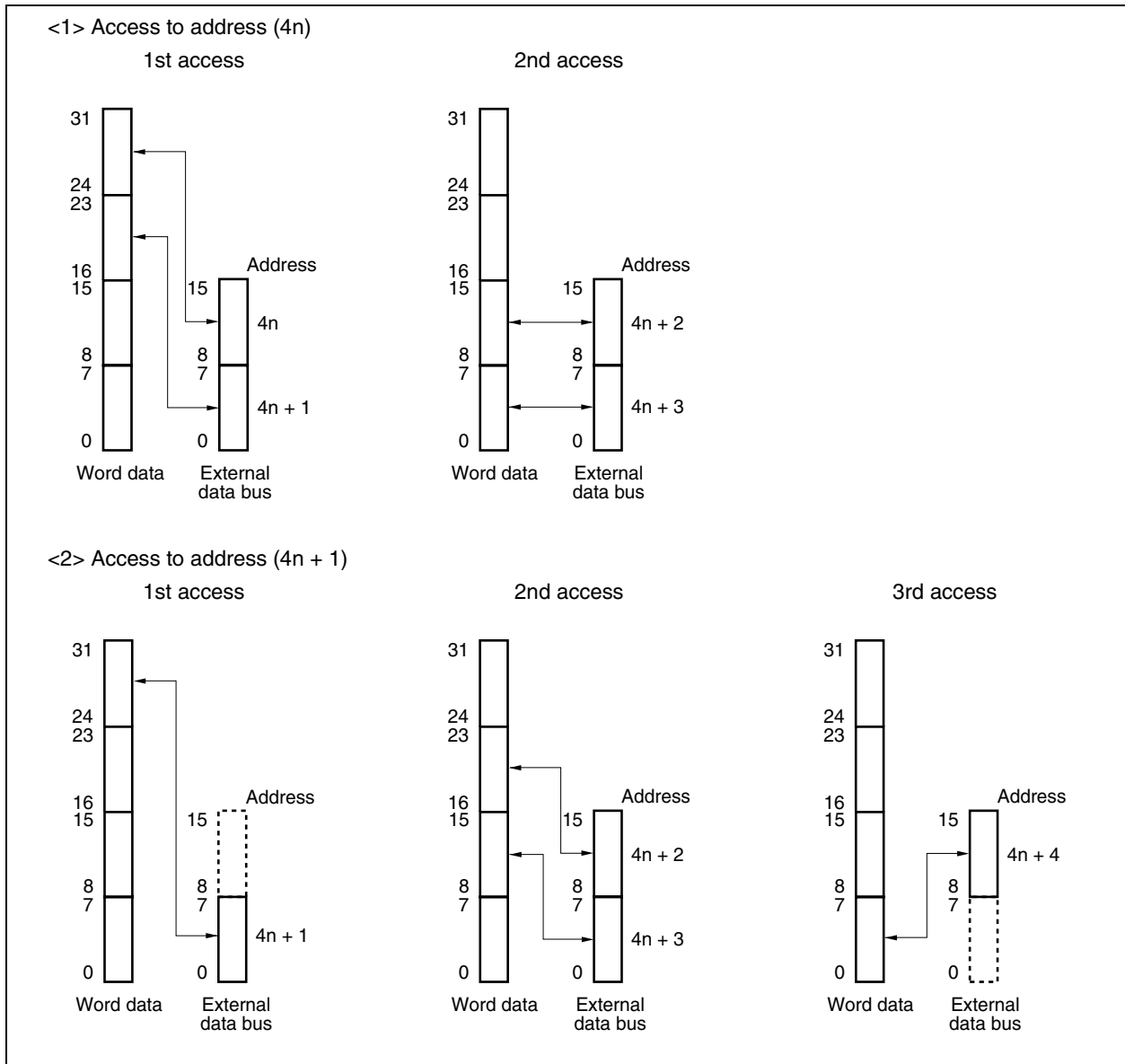
(c) When the data bus width is 8 bits (little endian) (2/2)



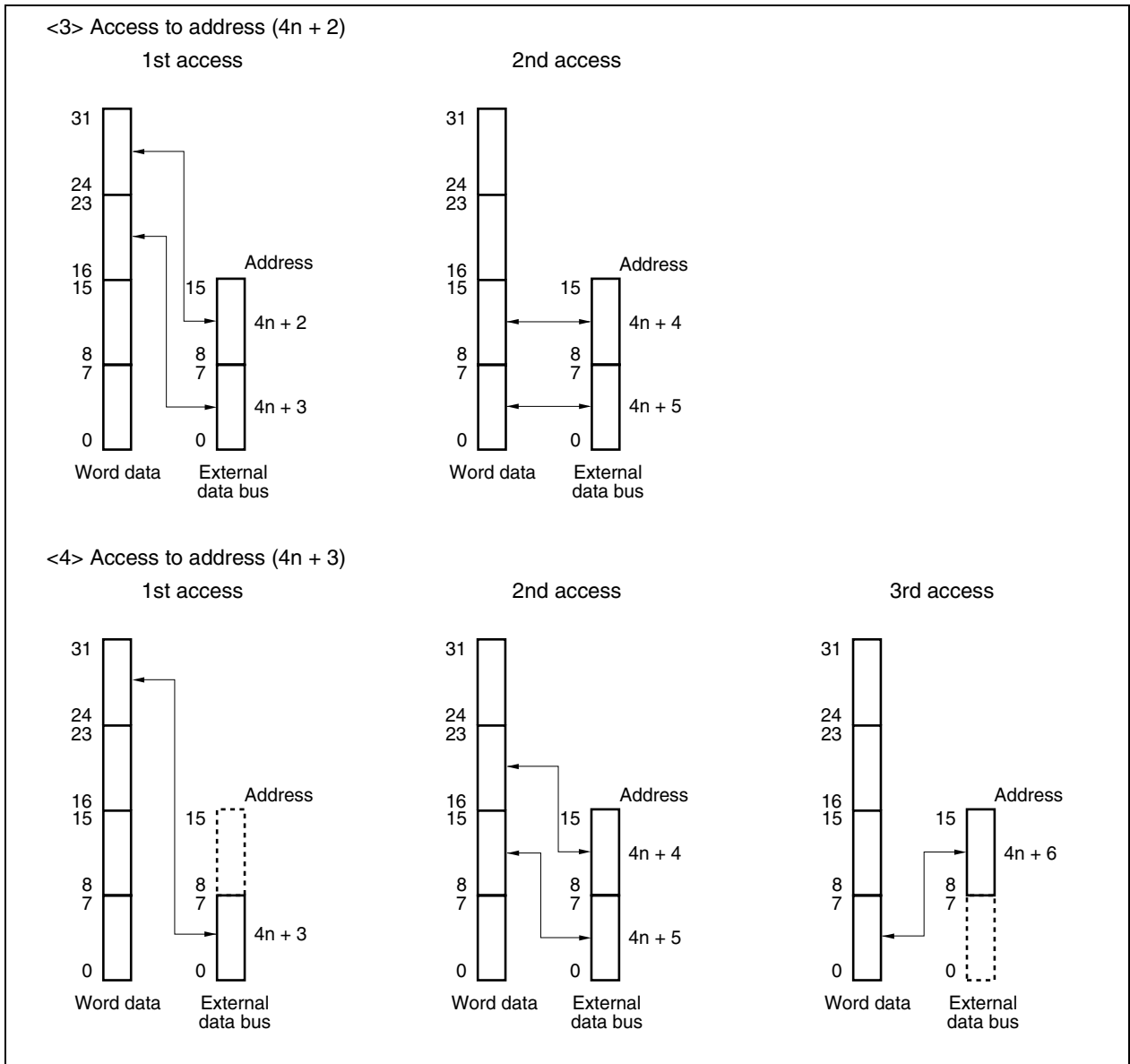
(d) When the data bus width is 32 bits (big endian)



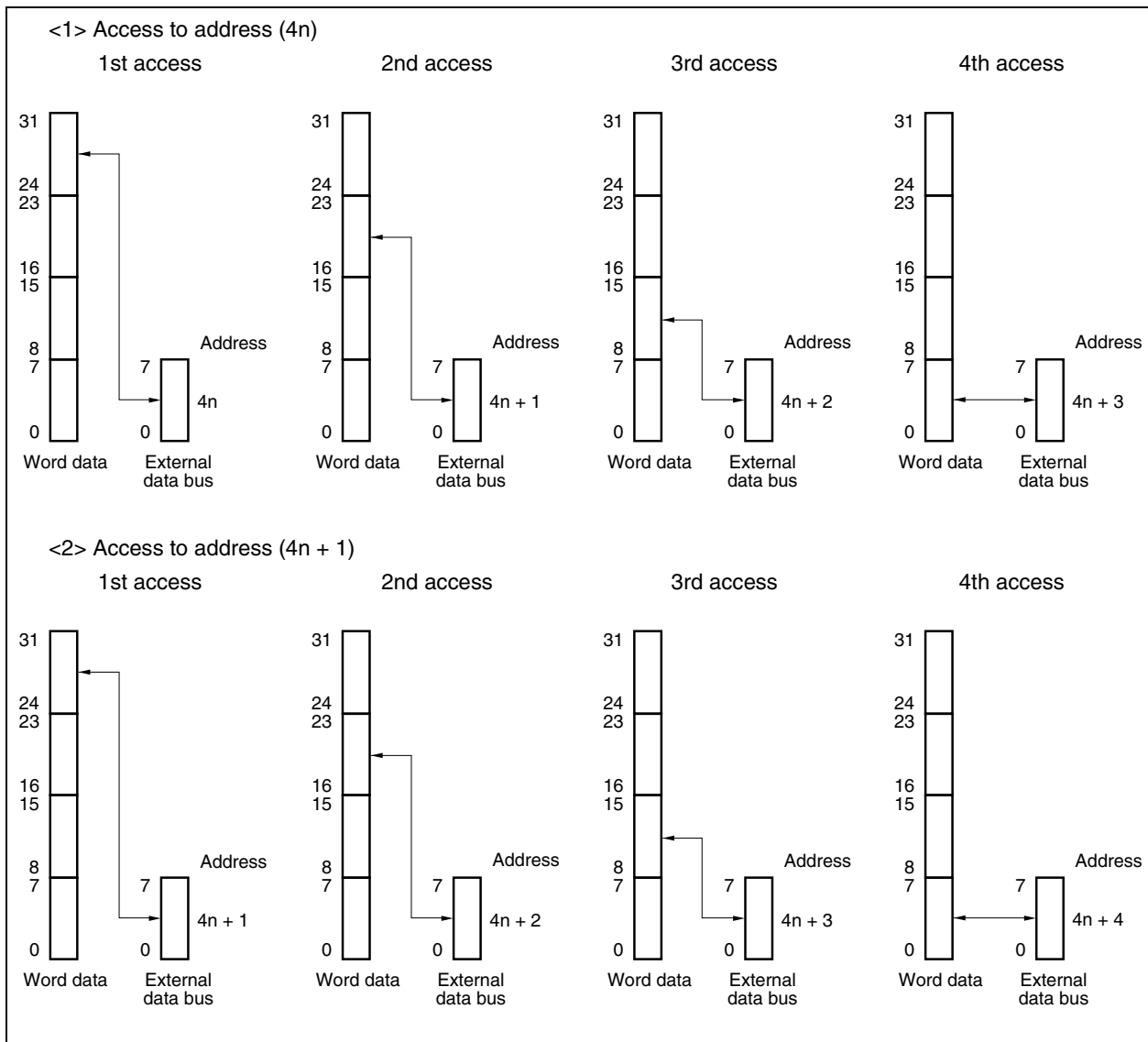
(e) When the data bus width is 16 bits (big endian) (1/2)



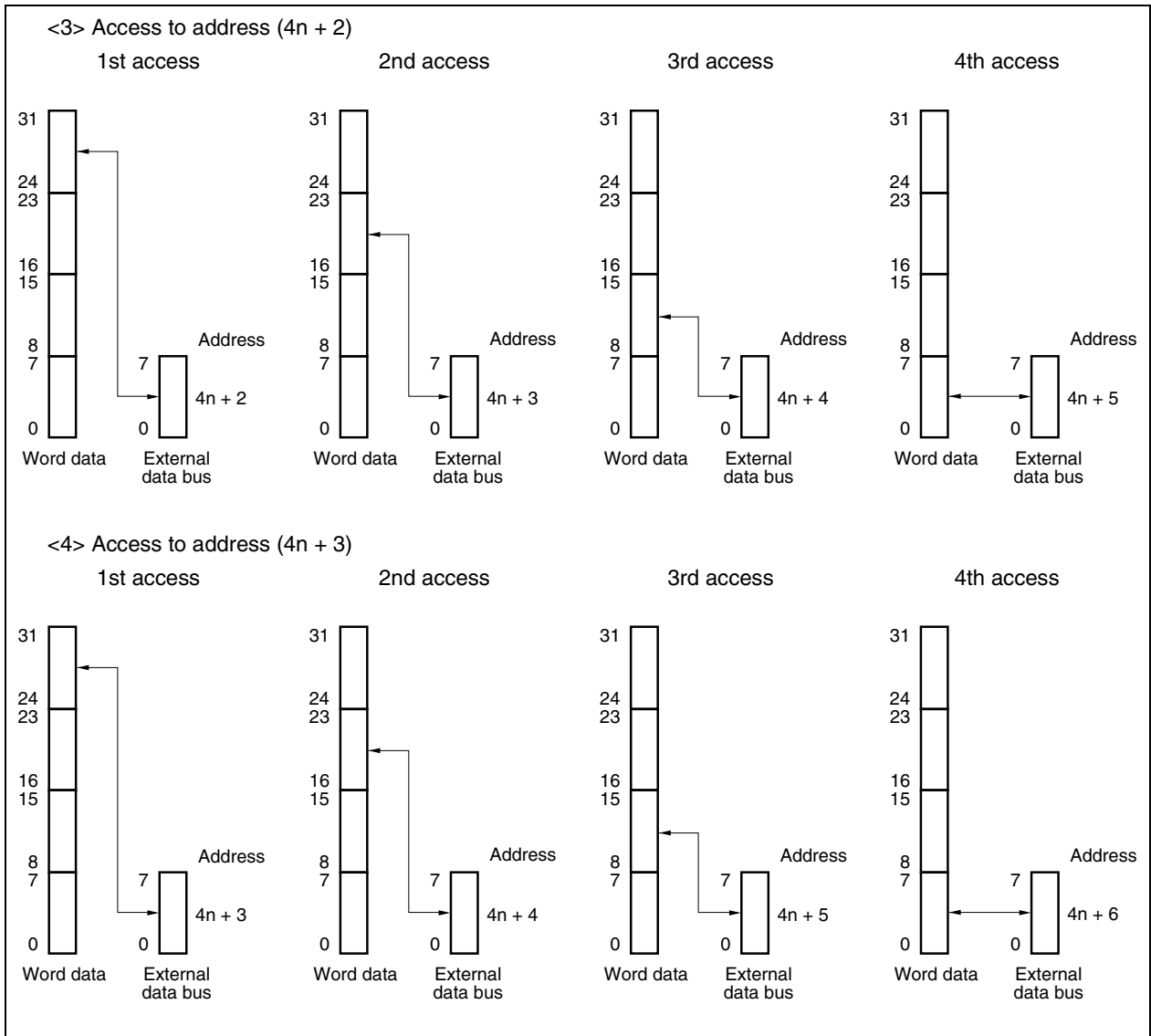
(e) When the data bus width is 16 bits (big endian) (2/2)



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



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



4.5.6 Data read control function

(1) Line buffer control registers 0, 1 (LBC0, LBC1)

The V850E/ME2 has a read buffer.

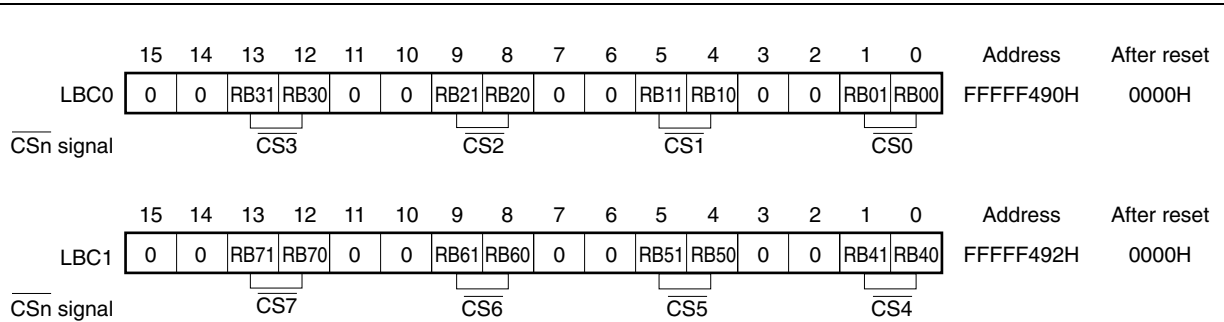
The LBC0 and LBC1 registers set the operating conditions of the read buffer in each CS space.

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

Be sure to clear bits 15, 14, 11, 10, 7, 6, 3, and 2 to 0. If they are set to 1, the operation is not guaranteed.

Cautions 1. Be sure to write data to the LBC0 and LBC1 registers after reset. After writing data to these registers, do not change their values.

★ **2.** When the speculative read function is set for each CSn space, do not insert an idle state in the CSn space for which the speculative read function is enabled by the BCC register (n = 0 to 7). If an idle state is required to be inserted in the CSn space for which the speculative read function is enabled, enable the speculative read function for all CSn spaces (set the LBC0 and LBC1 registers to 3333H) or disable the speculative read function for all CSn spaces (set the LBC0 and LBC1 registers to 0000H).



Bit position	Bit name	Function														
13, 12, 9, 8, 5, 4, 1, 0	RBn1, RBn0	<p>These bits set the operating conditions of the read buffer (timing of speculative read) in each CSn space.</p> <table border="1"> <thead> <tr> <th>RBn1</th> <th>RBn0</th> <th>Timing of speculative read</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td rowspan="2">Without speculative read (operation of read buffer is prohibited)</td> </tr> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>In all cycles (including data access)</td> </tr> </tbody> </table>	RBn1	RBn0	Timing of speculative read	0	0	Without speculative read (operation of read buffer is prohibited)	0	1	1	0	Setting prohibited	1	1	In all cycles (including data access)
RBn1	RBn0	Timing of speculative read														
0	0	Without speculative read (operation of read buffer is prohibited)														
0	1															
1	0	Setting prohibited														
1	1	In all cycles (including data access)														

★ **Remarks 1.** n = 0 to 7

2. If the setting of the RBn1 and RBn0 bits is changed, the data in the read buffer is immediately invalidated.

(a) Speculative read function (read buffer function)

The V850E/ME2 has speculative read buffers consisting of 4 words (128 bits). If the operating conditions set by the LBC0 or LBC1 register are satisfied when a read request is issued from the CPU or DMA controller to a CSn space, the requested address is read and the remaining three words of data are read (speculatively) to the read buffers, regardless of whether a request is issued from the CPU or DMA controller. Data is read to the read buffers in the sequence of critical first access.

Example of transfer: For speculative reading of 32 bits

- (i) xxxxx00H → xxxxx04H → xxxxx08H → xxxxx0CH
- (ii) xxxxx04H → xxxxx08H → xxxxx0CH → xxxxx00H
- (iii) xxxxx01H → xxxxx05H → xxxxx09H → xxxxx0DH^{Note}

Note If the lower 2 bits of an address are not used in the 32-bit mode, the operation is the same as (i) above.

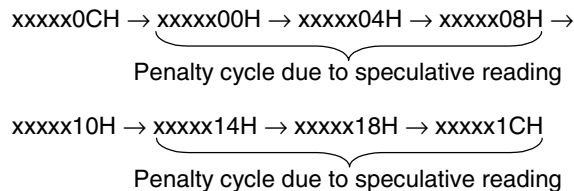
The four words of data that have been read are held in the read buffers. If the following conditions are satisfied, the data in the buffers is lost.

- Speculative read miss hit (read request to external memory area not stored in read buffer)
- Writing BCT0 or BCT1 register
- Writing LBS register
- Writing LBC0 or LBC1 register
- Generation of bus hold
- Generation of DMA flyby cycle (not dependent on CS space)
- Memory write access to speculatively read line address

If the CPU or DMA controller requests reading the data held in the read buffers, the data is transferred to the CPU or DMA controller without generating an external memory cycle.

★

Remark If the speculative read function is enabled in the 32-bit mode (with a 32-bit data bus width) and the LD.W instruction is executed at address xxxxx0DH, the following bus cycles are successively generated.

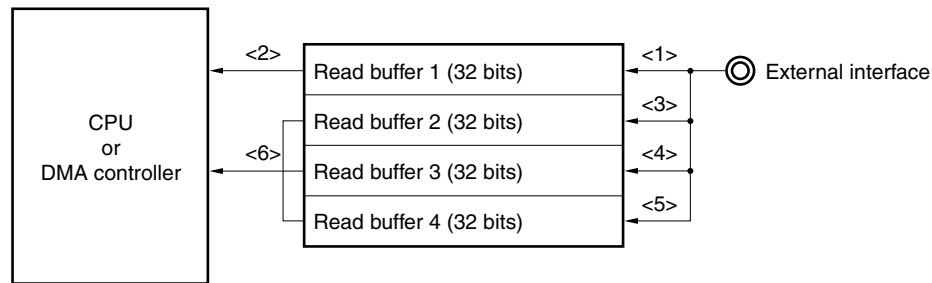


The CPU reads the following data when it stores data in the read buffer.

xxxxx0DH (8 bits) → xxxxx0EH (16 bits) → xxxxx10H (8 bits)

The speculative read operation is outlined below.

Figure 4-4. Outline of Speculative Read Operation



- (A) A read cycle to the address requested by the CPU or DMA controller is executed (<1>).
 (B) The CPU or DMA controller reads the data loaded to read buffer 1 (<2>).
 (C) Read buffers 2 to 4 are speculatively read, and a read cycle of the remaining three words is executed (<3> to <5>).
 (D) If the CPU or DMA controller generates a read request to an address that has already been speculatively read, data is read from read buffers 2 to 4 (a read access to the external memory does not occur) (<6>).

- Remarks 1.** (B) and (C) are executed in parallel ((D) is also executed in parallel if data is stored in the read buffers).
 2. If the CPU or DMA controller issues a read request to an address that is not speculatively read in (C) in the cycle of (D), the operation is kept waiting until the operation of (C) is completed. After completion of the operation of (C), all data in the read buffers is discarded, and an operation in response to the next address request is executed starting from (A) (miss hit operation of the speculative read function).

(b) Write buffer function

- ★ The V850E/ME2 has an on-chip write buffer of 4 words (128 bits). The write buffer stores data if a write cycle cannot be executed while the external bus is occupied^{Note}. The next instructions are speculatively executed until the write buffer becomes full. The write buffer is valid for all the external memory areas. If a write request is generated while the write buffer is full, the next instruction execution is postponed until there is a vacancy in the write buffer.

While data is being stored in the write buffer (when a write operation to the external memory has not been completed), DMA and bus hold requests are not acknowledged (DMA and bus hold requests are acknowledged and an enable signal is generated after all the data of the write buffers has been written to the external memory).

- ★ **Note** The external bus is occupied when there is a bus cycle currently under execution. If instruction fetch cycles are successively executed, all write cycles for the write data stored in the write buffer are always generated after the bus cycle currently under execution (instruction fetch cycle) is completed.

- Cautions**
1. Because the write buffer consists of four stages, the write buffer becomes full after 4 bytes (32 bits) when a byte write operation is executed (while a write operation to the external memory is standing by). Similarly, the conditions under which the write buffer becomes full vary due to an address miss-align access, etc.
 2. When data is written to an external device, the write operation to the external device may not be executed even when a CPU write operation has been completed by the write buffer. The CPU can access the on-chip peripheral I/O registers after the write operation has been completed, regardless of whether a write operation to the external device is executed. Therefore, if it is necessary to change the value of an on-chip peripheral I/O register after completion of execution of an external memory cycle, write the same value as the default value to the LBC0 or LBC1 register, and then write the on-chip peripheral I/O register whose value is to be changed. When writing an on-chip peripheral I/O register other than the LBC0 or LBC1 register without writing the same value as the default value to the LBC0 or LBC1 register, the register value may be changed before completion of the external memory cycle. Although rewriting the LBC0 and LBC1 registers is prohibited, the same value can be rewritten to the registers in this case. However, if a value different from the default value is written to the LBC0 or LBC1 register, the operation is not guaranteed.
 3. During 2-cycle transfer that writes data to the external device, the write operation to the external device may not be completed even if TCn bit of DCHCn register = 1 (DMA transfer completion) is read by the write buffer (n = 0 to 3). If it is necessary to change the value of an on-chip peripheral I/O register after completion of DMA transfer (completion of a write operation to the external device), perform either of the following operations.
 - Monitor the $\overline{\text{TCn}}$ signal (the $\overline{\text{TCn}}$ signal becomes active in synchronization with a write operation to the external device).
 - After detecting setting (to 1) of the TCn bit, write the same value as the default value to the LBC0 or LBC1 register, and then change the value of the on-chip peripheral I/O register. If the value of an on-chip peripheral I/O register other than the LBC0 and LBC1 registers is changed without writing the same value to LBC0 or LBC1, the value of the on-chip peripheral I/O register may be changed before completion of DMA transfer.

Although rewriting the LBC0 and LBC1 registers is prohibited, the same value can be rewritten to the registers in this case. However, if a value different from the default value is written to the LBC0 or LBC1 register, the operation is not guaranteed.

★

4.6 Bus Clock Control Function

(1) Bus mode control register (BMC)

This register sets the division rate of the bus clock (BUSCLK) with respect to the internal system clock. Writing the BMC register stops BUSCLK once at low level. BUSCLK resumes operation with the set divided clock after it has been stopped. While BUSCLK is stopped, the operation of the SDRAM refresh control register (RFSn) of SDRAM is also stopped (n = 1, 3, 4, 6).

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

Be sure to clear bits 7 to 2 to 0. If they are set to 1, the operation is not guaranteed.

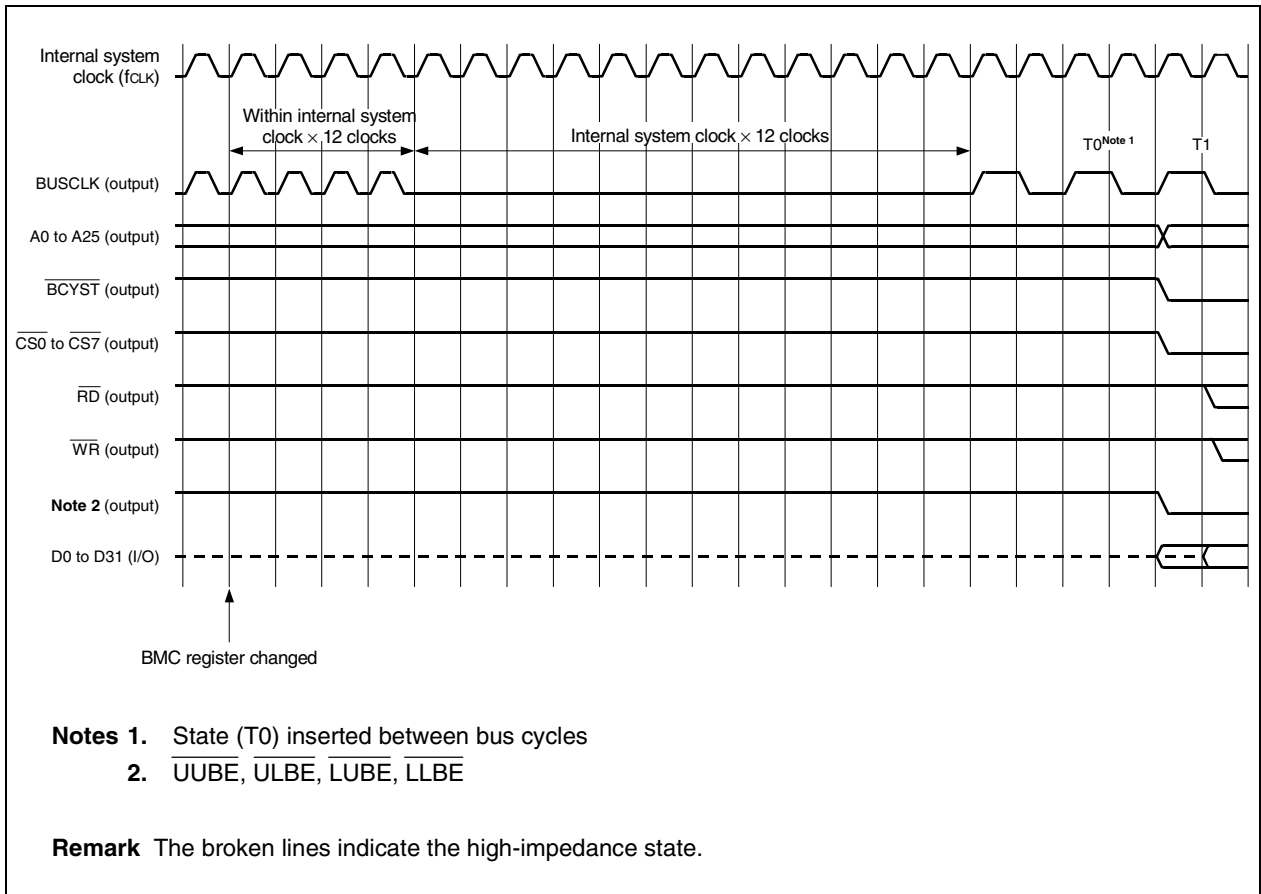
- Cautions**
1. Write the BMC register after reset, and then do not change the value that has been written.
 2. Before writing to the BMC register, be sure to set the VSWC register to x7H (x: Value before setting the BMC register). After writing the set value to the BMC register, re-set the value of the VSWC register to the value before the set value of the BMC register was written (see 3.4.10 Initialization sequence).

	7	6	5	4	3	2	1	0		Address	After reset
BMC	0	0	0	0	0	0	CKM1	CKM0	FFFFF498H	00H	

Bit position	Bit name	Function															
1, 0	CKM1, CKM0	These bits set the division ratio of the bus clock (BUSCLK) with respect to the internal system clock (f_{CLK}). <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">CKM1</th> <th style="width: 10%;">CKM0</th> <th style="width: 80%;">Division ratio of BUSCLK vs. 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;">f_{CLK}</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">$f_{CLK}/2$</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">$f_{CLK}/3$</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">$f_{CLK}/4$</td> </tr> </tbody> </table>	CKM1	CKM0	Division ratio of BUSCLK vs. f_{CLK}	0	0	f_{CLK}	0	1	$f_{CLK}/2$	1	0	$f_{CLK}/3$	1	1	$f_{CLK}/4$
CKM1	CKM0	Division ratio of BUSCLK vs. f_{CLK}															
0	0	f_{CLK}															
0	1	$f_{CLK}/2$															
1	0	$f_{CLK}/3$															
1	1	$f_{CLK}/4$															

Remark f_{CLK} : Internal system clock

Figure 4-5. BMC Register Switching Timing



4.7 Wait Function

4.7.1 Programmable wait function

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

To facilitate interfacing with low-speed memory and 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 DWC0 and DWC1 registers. Just after system reset, all blocks have 7 data wait states inserted.

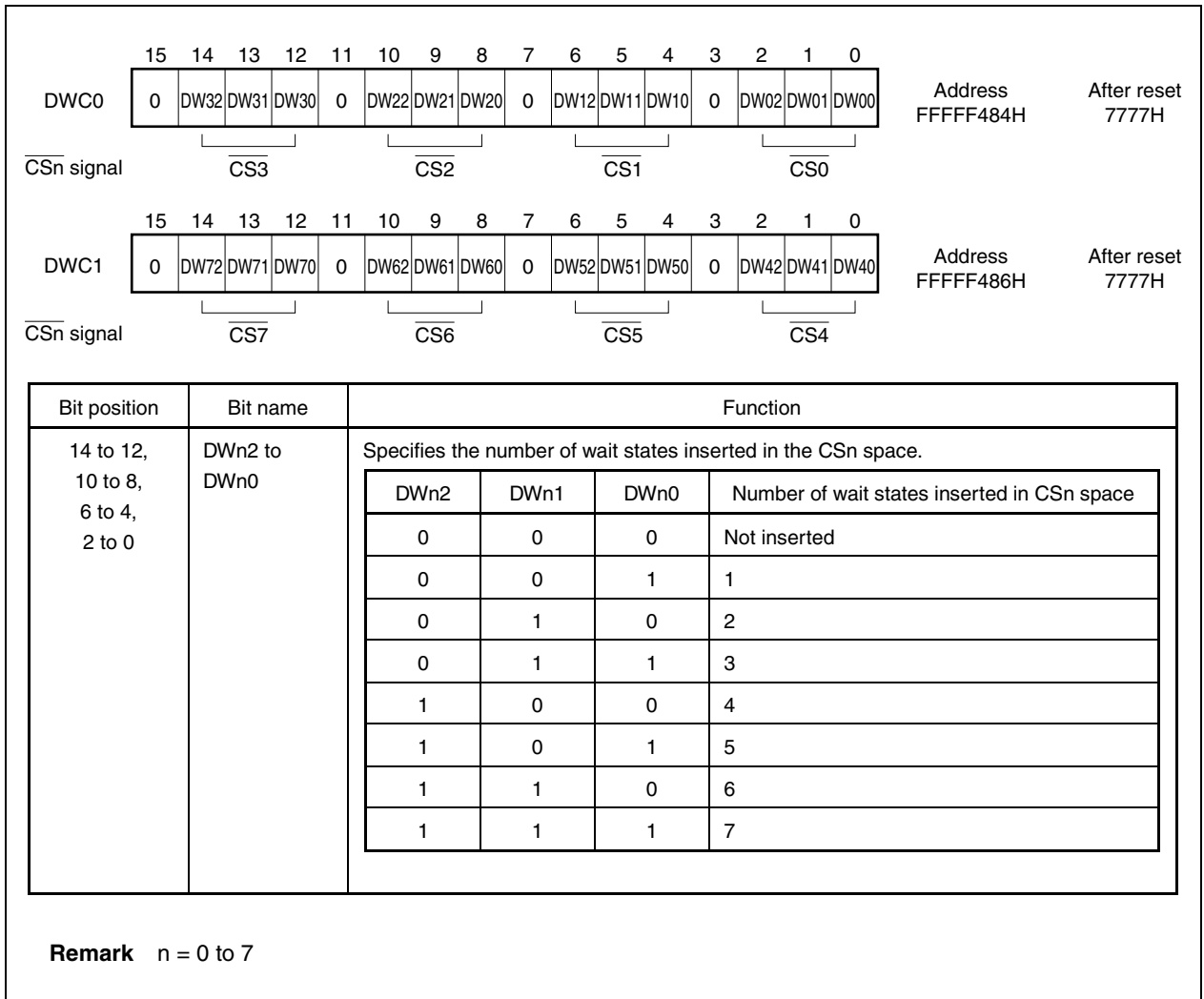
These registers can be read or written in 16-bit units.

Cautions 1. The internal instruction RAM area (in the read mode) and internal data RAM area are not subject to programmable waits and ordinarily no wait access is carried out.

When the internal instruction RAM area is accessed (in the write mode), the programmable wait value set for the CS0 space becomes valid.

The on-chip peripheral I/O area is not subject to programmable waits, with wait control performed by each peripheral function only.

2. In the following cases, the settings of the DWC0 and DWC1 registers 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/ME2 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).

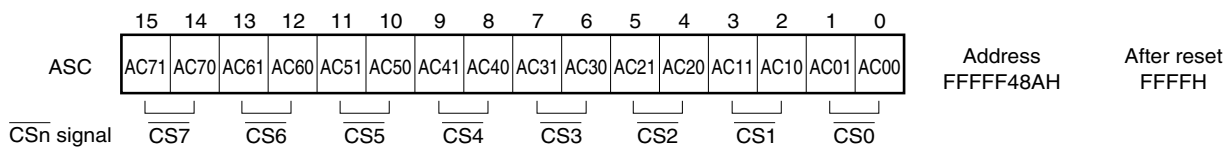
The number of address setup wait states can be set with the ASC register for each CS space.

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

Cautions 1. The internal instruction RAM area (in the read mode), internal data RAM area, and on-chip peripheral I/O area are not subject to address setup wait insertion.

When the internal instruction RAM area is accessed (in the write mode), the address setup wait value set for the CS0 space becomes valid.

2. During an address setup wait, the **WAIT** pin-based external wait function is disabled.
3. Write to the ASC register after reset, and then do not change the set value.



Bit position	Bit name	Function															
15 to 0	ACn1, ACn0	<p>Specifies the number of address setup wait states inserted before the SRAM/page ROM cycle for each CSn space.</p> <table border="1"> <thead> <tr> <th>ACn1</th> <th>ACn0</th> <th>Number of wait states</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Not inserted</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>2</td> </tr> <tr> <td>1</td> <td>1</td> <td>3</td> </tr> </tbody> </table>	ACn1	ACn0	Number of wait states	0	0	Not inserted	0	1	1	1	0	2	1	1	3
ACn1	ACn0	Number of wait states															
0	0	Not inserted															
0	1	1															
1	0	2															
1	1	3															

Remark n = 0 to 7

(3) Bus cycle period control register (BCP)

With the V850E/ME2, the operations of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ can be enabled or disabled in the SRAM, external ROM, and external I/O cycles.

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

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

★ **Cautions 1. During a flyby DMA transfer for SRAM, external ROM, or external I/O, the $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ signals are always output, irrespective of the IOEN bit setting.**

Flyby transfer from external I/O to external memory: $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{WR}}$ signals become active.

Flyby transfer from external memory to external I/O: $\overline{\text{IOW}}\overline{\text{R}}$ and $\overline{\text{RD}}$ signals become active.

In page ROM cycle, on the other hand, the IOEN bit setting has no meaning.

2. Write to the BCP register after reset, and then do not change the set values.

★ 3. If the internal instruction RAM is accessed while the IOEN bit is set (1) (write mode), the $\overline{\text{IOW}}\overline{\text{R}}$ signal becomes active.

	7	6	5	4	3	2	1	0	Address	After reset
BCP	0	0	0	0	IOEN	0	0	0	FFFFFF48CH	00H

Bit position	Bit name	Function						
3	IOEN	Specifies whether to enable/disable the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">IOEN</th> <th style="width: 90%;">Enable/disable $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ operation</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Disables the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles.</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Enables the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles.</td> </tr> </tbody> </table>	IOEN	Enable/disable $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ operation	0	Disables the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles.	1	Enables the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles.
IOEN	Enable/disable $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ operation							
0	Disables the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles.							
1	Enables the operation of $\overline{\text{IOR}}\overline{\text{D}}$ and $\overline{\text{IOW}}\overline{\text{R}}$ in SRAM, external ROM, and external I/O cycles.							

(4) DMA flyby transfer wait control register (FWC)

The FWC register sets the number of data wait cycles during DMA flyby transfer for each channel n (see the timing chart in 6.5.2 Flyby transfer) (n = 0 to 3).

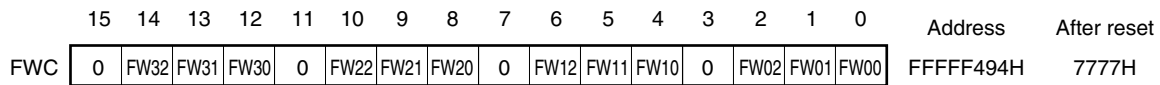
The set value of this register is valid during DMA flyby transfer, and the set values of the DWC0, DWC1, and PRC registers are invalid.

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

Cautions 1. The internal instruction RAM area and internal data RAM area are not subject to the programmable wait operation and are always accessed with no wait cycle. The on-chip peripheral I/O area is also not subject to the programmable wait operation, and is subject only to wait control by each peripheral function.

2. Write to the FWC register after reset, and then do not change the set values.

3. DMA flyby transfer that accesses SDRAM inserts the set value + 1 data wait cycles.



Bit position	Bit name	Function																																															
14 to 12, 10 to 8, 6 to 4, 2 to 0	FWn2 to FWn0	These bits set the number of data wait cycles to each channel n during DMA flyby transfer. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th rowspan="2" style="width: 10%;">FWn2</th> <th rowspan="2" style="width: 10%;">FWn1</th> <th rowspan="2" style="width: 10%;">FWn0</th> <th colspan="2" style="width: 60%;">Number of data wait cycles</th> </tr> <tr> <th style="width: 30%;">When SDRAM is not accessed</th> <th style="width: 30%;">When SDRAM is accessed</th> </tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>0</td><td>Not inserted</td><td>1</td> </tr> <tr> <td>0</td><td>0</td><td>1</td><td>1</td><td>2</td> </tr> <tr> <td>0</td><td>1</td><td>0</td><td>2</td><td>3</td> </tr> <tr> <td>0</td><td>1</td><td>1</td><td>3</td><td>4</td> </tr> <tr> <td>1</td><td>0</td><td>0</td><td>4</td><td>5</td> </tr> <tr> <td>1</td><td>0</td><td>1</td><td>5</td><td>6</td> </tr> <tr> <td>1</td><td>1</td><td>0</td><td>6</td><td>7</td> </tr> <tr> <td>1</td><td>1</td><td>1</td><td>7</td><td>8</td> </tr> </tbody> </table>	FWn2	FWn1	FWn0	Number of data wait cycles		When SDRAM is not accessed	When SDRAM is accessed	0	0	0	Not inserted	1	0	0	1	1	2	0	1	0	2	3	0	1	1	3	4	1	0	0	4	5	1	0	1	5	6	1	1	0	6	7	1	1	1	7	8
FWn2	FWn1	FWn0				Number of data wait cycles																																											
			When SDRAM is not accessed	When SDRAM is accessed																																													
0	0	0	Not inserted	1																																													
0	0	1	1	2																																													
0	1	0	2	3																																													
0	1	1	3	4																																													
1	0	0	4	5																																													
1	0	1	5	6																																													
1	1	0	6	7																																													
1	1	1	7	8																																													

Remark n = 0 to 3

4.7.2 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.

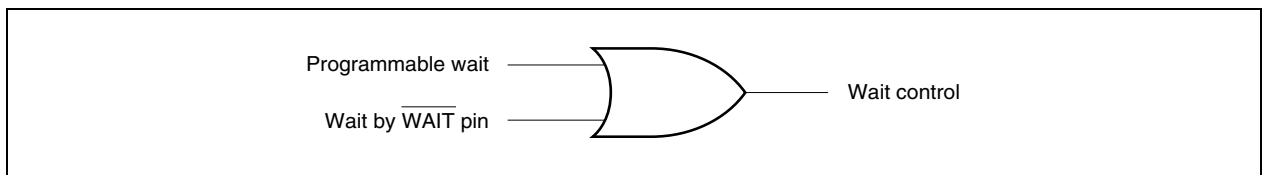
Just as with programmable waits, accessing internal instruction RAM (in the read mode), internal data RAM, and on-chip peripheral I/O areas cannot be controlled by external waits.

The external $\overline{\text{WAIT}}$ signal can be input asynchronously to BUSCLK 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.

Caution Be sure to make the $\overline{\text{WAIT}}$ pin high when the internal instruction RAM is accessed (in the write mode).

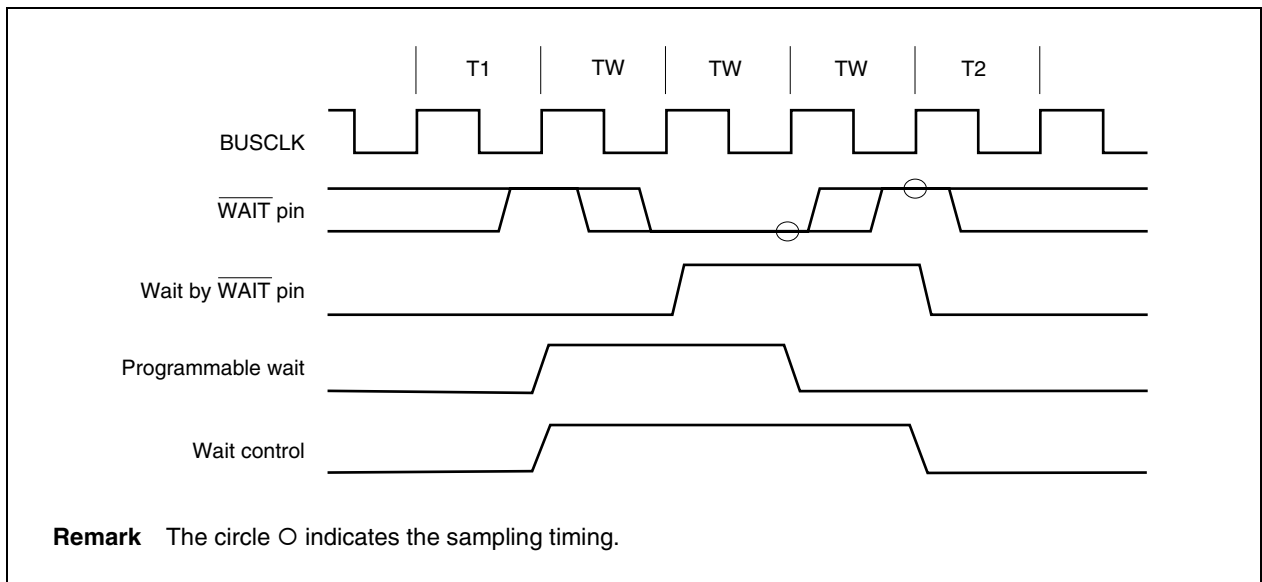
4.7.3 Relationship between programmable wait and external wait

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 4-6. Example of Wait Insertion



4.7.4 Bus cycles in which wait function is valid

In the V850E/ME2, 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 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	BCWm1, BCWm0	1 to 3	– (invalid)	
DMA flyby transfer cycle	External I/O → SRAM		Flyby transfer wait	FWC	FWa2 to FWa0	0 to 7	√ (valid)
	SDRAM → external I/O	Off-page	Row address precharge	SCRm	BCWm1, BCWm0	1 to 3	– (invalid)
			Flyby transfer wait	FWC	FWa2 to FWa0	0 to 7	√ (valid)
		On-page	Row address precharge	SCRm	BCWm1, BCWm0	1 to 3	– (invalid)
			Flyby transfer wait	FWC	FWa2 to FWa0	0 to 7	√ (valid)
	External I/O → SDRAM	Off-page	Row address precharge	SCRm	BCWm1, BCWm0	1 to 3	– (invalid)
			Flyby transfer wait	FWC	FWa2 to FWa0	0 to 7	√ (valid)
		On-page	Row address precharge	SCRm	BCWm1, BCWm0	1 to 3	– (invalid)
Flyby transfer wait			FWC	FWa2 to FWa0	0 to 7	√ (valid)	

Remark n = 0 to 7, m = 1, 3, 4, 6, a = 0 to 3

4.8 Idle State Insertion Function

(1) Bus cycle control register (BCC)

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 at 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, see the memory access timings in Chapter 5.

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

Cautions 1. The internal instruction RAM area (in the read mode), internal data RAM area, and on-chip peripheral I/O area are not subject to idle state insertion.

When the internal instruction RAM area is accessed (in the write mode), the idle state value set for the CS0 space becomes valid.

2. Write to the BCC register 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 BCC register is complete. However, it is possible to access external memory areas whose initialization settings are complete.
3. The chip select signal (\overline{CSn}) does not become active in the idle state ($n = 0$ to 7).

	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0																						
BCC	BC71 BC70 BC61 BC60 BC51 BC50 BC41 BC40 BC31 BC30 BC21 BC20 BC11 BC10 BC01 BC00	Address	After reset																				
		FFFFF488H	FFFFH																				
\overline{CSn} signal	<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> <td style="border: none; text-align: center;">┌───┐</td> </tr> <tr> <td style="border: none; text-align: center;">CS7</td> <td style="border: none; text-align: center;">CS6</td> <td style="border: none; text-align: center;">CS5</td> <td style="border: none; text-align: center;">CS4</td> <td style="border: none; text-align: center;">CS3</td> <td style="border: none; text-align: center;">CS2</td> <td style="border: none; text-align: center;">CS1</td> <td style="border: none; text-align: center;">CS0</td> <td style="border: none;"></td> <td style="border: none;"></td> </tr> </table>	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	CS7	CS6	CS5	CS4	CS3	CS2	CS1	CS0				
┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐	┌───┐														
CS7	CS6	CS5	CS4	CS3	CS2	CS1	CS0																

Bit position	Bit name	Function															
15 to 0	BCn1, BCn0	Specifies the insertion of an idle state in the CSn space. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">BCn1</th> <th style="width: 10%;">BCn0</th> <th style="width: 80%;">Idle state in CSn space</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">Not 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>	BCn1	BCn0	Idle state in CSn space	0	0	Not inserted	0	1	1	1	0	2	1	1	3
BCn1	BCn0	Idle state in CSn space															
0	0	Not inserted															
0	1	1															
1	0	2															
1	1	3															

Remark n = 0 to 7

(2) DMA flyby transfer idle control register (FIC)

The FIC register sets the number of idle states during DMA flyby transfer for each DMA channel n (see the timing chart in **6.5.2 Flyby transfer**) (n = 0 to 3). The idle state is inserted at the end of DMA flyby transfer. During DMA flyby transfer, the set value of this register is valid, and the set value of the bus cycle control register (BCC) is invalid.

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

- Cautions**
1. The internal instruction RAM area, internal data RAM area, and on-chip peripheral I/O area are not subject to idle state insertion.
 2. Write to the FIC register after reset, and then do not change the set values.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
FIC	0	0	F131	F130	0	0	F121	F120	0	0	F111	F110	0	0	F101	F100	FFFFFF496H	3333H

Bit position	Bit name	Function															
13, 12, 9, 8, 5, 4, 1, 0	F1n1, F1n0	These bits specify the number of idle states during DMA flyby transfer for each channel n. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>F1n1</th> <th>F1n0</th> <th>Idle state during DMA flyby transfer</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Not inserted</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>2</td> </tr> <tr> <td>1</td> <td>1</td> <td>3</td> </tr> </tbody> </table>	F1n1	F1n0	Idle state during DMA flyby transfer	0	0	Not inserted	0	1	1	1	0	2	1	1	3
F1n1	F1n0	Idle state during DMA flyby transfer															
0	0	Not inserted															
0	1	1															
1	0	2															
1	1	3															

Remark n = 0 to 3

4.9 Instruction Cache Function

The V850E/ME2 has an 8 KB 2-way set associative instruction cache.

This instruction cache uses an LRU (Least Recently Used) algorithm. This cache uses two ways as the cache, with the way to be replaced in case of a miss hit determined by the internal LRU bit. If the way is specified as a cacheable area, a 4-word burst sequential read cycle is issued (when the external 32-bit bus operates) (when the external 16-bit bus operates, an 8-halfword burst read cycle is issued).

Set the instruction cache using the following procedure, after system reset.

- <1> Wait until the value of the ICC register reaches "0000H" (until the tag is initialized).
- <2> Make the setting of the instruction cache cacheable by using the BHC register.

Caution Be sure to set the BHC register in an uncached area (instructions cannot be fetched correctly if it is set from a cacheable area).

4.9.1 Cache configuration register (BHC)

(1) Cache configuration register (BHC)

This register can set the configuration of the cache memory in each CS space selected by the chip select signals (CS0 to CS2). Setting a cacheable area becomes valid immediately after the BHC register has been set.

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

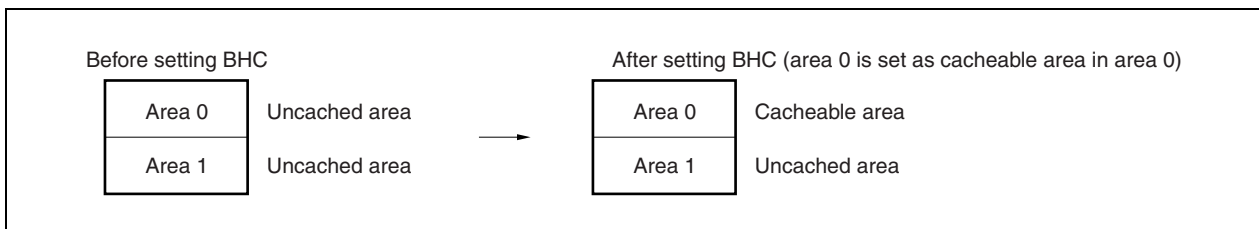
Be sure to clear bits 15 to 5 to 0. If they are set to 1, the operation is not guaranteed.

- Cautions**
1. Write the BHC register after reset, and then do not change the set values.
 2. The area that includes the instruction that sets the BHC register cannot be set from an uncached area to a cacheable area, or from a cacheable area to an uncached area. In this case, branch to area 1 and then set area 0 to a cacheable area by an instruction in area 1. If necessary, branch to area 0 again.

An uncached area and a cacheable area can be set in any CS space in the internal instruction RAM area.

(Example of prohibited setting)

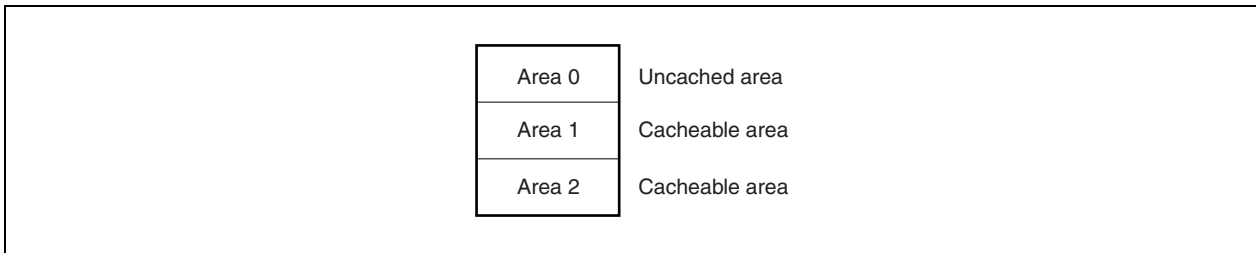
- If the instruction that sets the BHC register is in area 0



Cautions 3. If adjacent memory areas are a cacheable area and an uncached area, only a branch instruction can be used to successively access this memory boundary. The operation is not guaranteed if this memory boundary is successively accessed by an instruction other than a branch instruction.

(Operation example)

- An access from area 0 to area 1 can be made only by a branch instruction.
- Successive accesses can be made from area 1 to area 2.



	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		Address	After reset																					
BHC	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td> </tr> </table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 5px;">BH20</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">BH10</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">BH00</td> </tr> </table>	BH20	0	BH10	0	BH00	FFFFF06AH	0000H
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0										
BH20	0	BH10	0	BH00																					
CSn signal																									
			CS2	CS1	CS0																				

Bit position	Bit name	Function
4, 2, 0	BHn0	These bits set the instruction cache placed in the CSn space. 0: Uncached 1: Cacheable

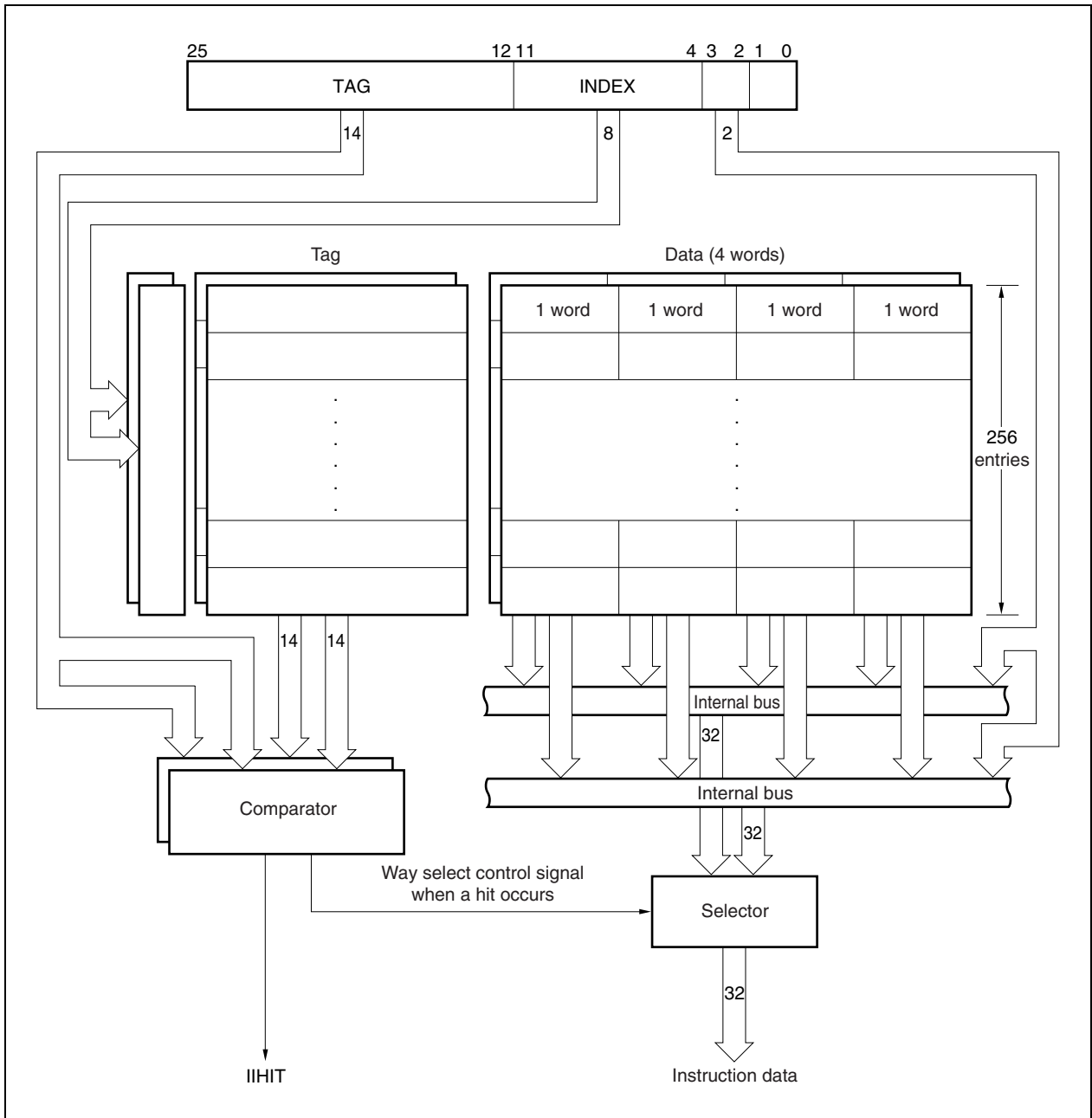
Remark n = 0 to 2

4.9.2 8 KB 2-way set associative cache

The data memory of the 8 KB 2-way set associative cache has two ways each consisting of 4-word lines and blocks of 256 entries, and has a total capacity of 8 KB. If a cache miss occurs, the data memory is refilled in 1-line units.

Only the instructions related to data access of the cacheable area can be cached in the instruction cache.

Figure 4-7. Configuration of 8 KB 2-Way Set Associative Instruction Cache



4.9.3 LRU algorithm

This algorithm is used to record accesses to blocks and replace the block that has not been used for the longest time, so that information that may be necessary is not discarded immediately.

4.9.4 Instruction cache control function

(1) Instruction cache control register (ICC)

This register sets two types of functions, tag clear and auto fill.

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

If the higher 8 bits of the ICC register are used as an ICCH register, and the lower 8 bits, as an ICCL register, these registers can be read or written in 8-bit or 1-bit units.

Bit 12 can only be read and cleared.

Be sure to clear bits 15 to 13, 11 to 5, 3, and 2 to 0. If they are set to 1, the operation is not guaranteed.

Cautions 1. Do not forcibly clear bits 0, 1, and 4 when they are set to 1.

2. Do not set bit 4 to 1 at the same time as the other bits.

3. Do not set bit 12 to 1. This bit can only be cleared to 0.

4. Be sure to set the ICC register in an uncached area (except setting of bit 4).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
ICC	0	0	0	LOCKI0	0	0	0	0	0	0	0	FILL0	0	0	TCLR1	TCLR0	FFFFFF070H	0003H ^{Note}

Note The value of this register is 0003H when the reset signal becomes active, and initialization of the tag is automatically started. When initialization of the tag has been completed, the value of this register is cleared to 0000H.

Bit position	Bit name	Function
12	LOCKI0	Indicates the cache lock status of way 0. The cache is locked when way 0 is filled, and this bit is automatically set to 1. When this bit is cleared to 0, cache lock of way 0 is cleared. 0: Way 0 is not locked. 1: Way 0 is locked.
4	FILL0	Sets auto fill of way 0. When this bit is set to 1, way 0 is automatically filled. When auto fill has been completed, this bit is automatically cleared to 0. 0: Filling way 0 is completed. 1: Way 0 is being filled.
1	TCLR1	Sets tag clear of way 1. When this bit is set to 1, the tag of way 1 is cleared (invalidated). When clearing the tag has been completed, this bit is automatically cleared to 0. 0: Clearing tag of way 1 is completed. 1: Tag of way 1 is being cleared.
0	TCLR0	Sets tag clear of way 0. When this bit is set to 1, the tag of way 0 is cleared (invalidated). When clearing the tag has been completed, this bit is automatically cleared to 0. 0: Clearing tag of way 0 is completed. 1: Tag of way 0 is being cleared.

(2) Instruction cache data configuration register (ICD)

This register sets the address of the memory area that is automatically filled by the auto fill function.

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

Be sure to clear bit 0 to 0. If it is set to 1, the operation is not guaranteed.

Cautions 1. Do not rewrite the ICD register during the auto fill operation. The operation is not guaranteed if it is rewritten.

2. Because the default value of the ICD register is undefined, be sure to set a value to the ICD register when the auto fill function is used, and then set the FILL0 bit of the ICC register to 1. If the FILL0 bit of the ICC register is set to 1 without setting a value to the ICD register, the operation is not guaranteed.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
ICD	0	ICD 14	ICD 13	ICD 12	ICD 11	ICD 10	ICD 9	ICD 8	ICD 7	ICD 6	ICD 5	ICD 4	ICD 3	ICD 2	ICD 1	0	FFFFFF074H	Undefined

Bit position	Bit name	Function
14 to 1	ICD14 to ICD1	Set the higher 14 bits of tag information (bits 25 to 12 of the first address of the memory area to be automatically filled).

4.9.5 Tag clear function

The tag of one way is cleared (invalidated).

After reset, all the ways, tags, and LRU are automatically cleared (invalidated).

The instruction cache tag is cleared using the following procedure.

- <1> The ICC register is read to confirm that both the TCLR0 and TCLR1 bits are cleared to 0.
- <2> The ICC register is read to confirm that the LOCKI0 bit is cleared to 0. Bit 13 of the ICC register is always cleared to 0.
- <3> The TCLR0 and TCLR1 bits of the ICC register are set as follows.

Cautions 1. Perform operations <1> to <3> above (tag clear) in an uncached area (the tag is not cleared from a cacheable area).

- **To clear both ways 0 and 1 at the same time**
 - (a) Set the TCLR0 and TCLR1 bits to 1.
 - (b) Read the TCLR0 and TCLR1 bits to confirm that they are cleared to 0.
- **To clear ways 0 and 1 separately^{Note}**
 - (a) Set the TCLR0 bit to 1.
 - (b) Read the TCLR0 bit to confirm that it is cleared to 0.
 - (c) Set the TCLR1 bit to 1.
 - (d) Read the TCLR1 bit to confirm that it is cleared to 0.

Note The ways can be cleared separately also in the order of (c) - (d) - (a) - (b).

- 2. The counter for clearing the tag of ways 0 and 1 is shared.**

Therefore, clear the tag (by setting the TCLR0 or TCLR1 bit of the ICC register) when the counter for clearing the tag is stopped (when TCLR0 = TCLR1 = 0). When clearing the tags of ways 0 and 1 separately, the counter stops while a tag is being cleared if the tag of one other way is cleared while the tag of the other way is being cleared (TCLR0 or TCLR1 = 1). Consequently, the tag cannot be cleared correctly because the tag of the other way is cleared with the counter indicating a midway value. Be sure to clear the tag of one other way after confirming that the tag of the other way has been cleared (TCLR0 or TCLR1 = 0).

There is no problem if both the bits are set at the same time as follows.

```

mov    0x3, r2
LOP0:
ld.h  ICC[r0], r1
cmp   r0, r1
bnz  LOP0
st.h  r2, ICC[r0]
LOP1:
ld.h  ICC[r0], r1
cmp   r0, r1
bnz  LOP1

```

- 3. Do not perform other processing in parallel with clearing the tag until it is confirmed by reading the TCLR0 and TCLR1 bits of the ICC register that the tag has been cleared to 0.**

4.9.6 Auto fill function (way 0 only)

The instruction of one way is automatically filled.

The way that has been filled automatically is automatically locked and prohibited from being written, and performs the same operation as ROM that can be accessed in one cycle. After the way is unlocked, it operates as an instruction cache again.

The instruction cache is automatically filled using the following procedure.

- <1> Clear (invalidate) the tag of way 0 (see **4.9.5 Tag clear function**).
- <2> Set information on the tag corresponding to the memory area to be automatically filled to the ICD register.
- <3> Branch to the cacheable area corresponding to the tag information set to the ICD register.
- <4> Set the FILL0 bit of the ICC register to 1.
- <5> When auto fill is completed, the LOCKI0 bit of the ICC register is automatically set to 1, and way 0 is locked. At this time, read the FILL0 bit of the ICC register at the same time to confirm that this bit is cleared to 0.

Caution Perform the above operations in the following areas.

<1>, <2>, <3> **Uncached area**

<4> **Cacheable area**

If the FILL0 bit of the ICC register is set to 1 from an uncached area, auto fill cannot be executed (is invalidated).

<5> **This can be done from both an uncached area and a cacheable area.**

Remark To unlock the way, clear the LOCKI0 bit of the ICC register to 0.

4.10 Internal Instruction RAM Control Function

The V850E/ME2 has 128 KB of instruction RAM (16 KB × 32 bits × 2 banks). While one bank is being read, the other bank can be written in parallel.

4.10.1 Internal instruction RAM mode register (IRAMMM)

(1) Internal instruction RAM mode register (IRAMMM)

This register specifies a mode in which the instruction RAM is accessed. This register can be read or written in 8-bit or 1-bit units.

Cautions 1. When the setting of a mode is changed, the next access (such as code fetching) to the internal instruction RAM may be executed first because of the pipeline operation of the CPU. When the setting of the IRAMMM register is changed, therefore, be sure to read the IRAMMM register to confirm that writing has been executed, and then execute an access to the internal instruction RAM.

2. Data can be written to the internal instruction RAM only by a word access.
3. Address miss-align access to the internal instruction RAM is prohibited.

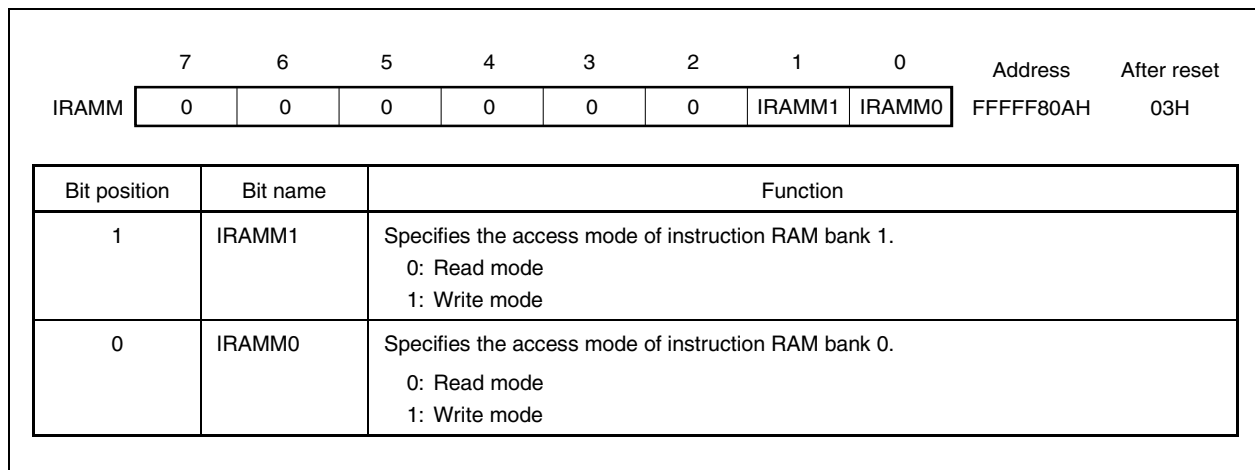
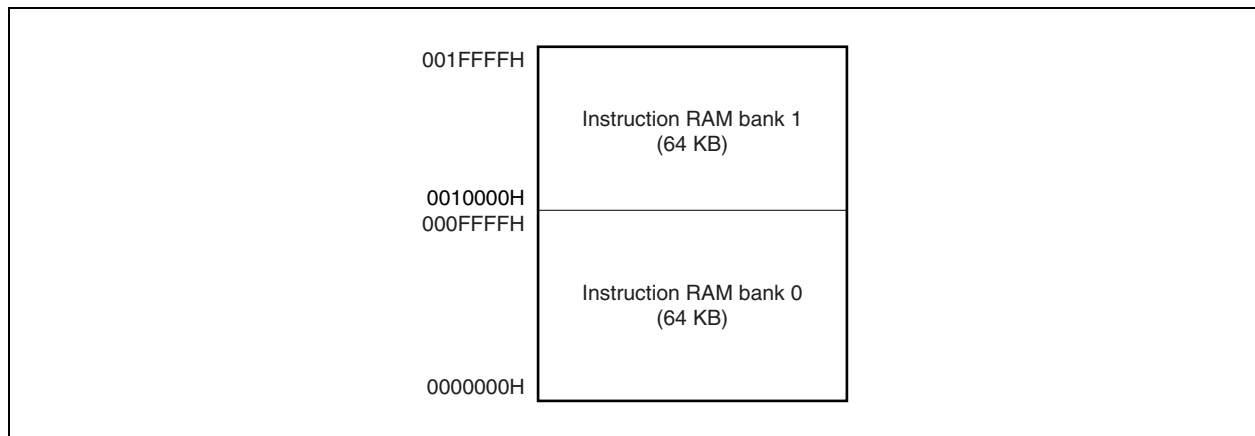


Figure 4-8. Memory Map of Internal Instruction RAM



4.10.2 Operation

(1) Read operation

The internal instruction RAM can be read by normal RAM access without having to be aware of the banks if the read mode is selected by the IRAMM register.

Cautions 1. A bank in the write mode cannot be read.

- 2. The internal instruction RAM (in the read mode) is accessed using the $\times 1$ internal system clock.**

(2) Write operation

The internal instruction RAM can be written by normal RAM access without having to be aware of the banks if the write mode is selected by the IRAMM register.

Cautions 1. A bank in the read mode cannot be written.

- 2. Internal instruction RAM bank 0 area is allocated to interrupt and exception tables. Disable interrupts until writing an instruction code to bank 0 of the internal instruction RAM is completed. Similarly, disable interrupts when bank 0 of the internal instruction RAM is set in the write mode.**

For disabling maskable interrupts, refer to interrupt mask registers 0 to 5 (IMR0 to IMR5) (7.3.5 Interrupt mask registers 0 to 5 (IMR0 to IMR5)). To disable the non-maskable interrupt, set the NP bit of the PSW to 1 to disable multiple interrupts (see 3.2.2 (2) Program status word (PSW)). For the NMI mask operation when reset is cleared, refer to the NMI reset status register (NRS) (7.3.6 NMI reset status register (NRS)).

- 3. When the internal instruction RAM is accessed (in the write mode), programmable waits, address setup waits, and idle states can be inserted in the CS0 space. If none of the above states is set, the instruction RAM is accessed using a $\times 2$ BUSCLK frequency clock.**

4.10.3 Cautions

- (1) Disable all interrupts from when reset is cleared until when program codes are completely transferred to the internal instruction RAM. It is not necessary to disable maskable interrupts because they are masked by default. Because the software exception and exception trap cannot be masked, do not execute the TRAP and DBTRAP instructions until transfer of program codes to the internal instruction RAM is completed.
- (2) After reset has been cleared, the NMI input is masked by hardware. The NMI is unmasked as soon as the IRAMM0 bit of the IRAMM register has been cleared to 0.
- (3) To write data to instruction RAM bank 0 of the internal instruction RAM, set the NP bit of the PSW to 1 to disable the NMI and maskable interrupts and suppress occurrence of the software exception and exception trap. Clear the NP bit by setting the IRAMM0 bit of the IRAMM register to 1 and confirming that the read mode is set, after the program has been rewritten.
- (4) NMI or maskable interrupt requests that have been generated while the NP bit of the PSW is set to 1 are held pending. An NMI request is acknowledged immediately after the NP bit has been cleared to 0. A maskable interrupt is acknowledged immediately after the NP bit has been cleared to 0, if the interrupt request is not cleared (by clearing the xxIFn bit of the interrupt control register (xxICn) to 0), if interrupts are not disabled (DI status), and if the xxMKn bit of the interrupt control register is not set to 1 before the NP bit is cleared to 0. However, only one interrupt request, NMI request or a maskable interrupt request, can be held pending per interrupt source. Even if the same interrupt request is generated more than once, only the first interrupt request is acknowledged (xx: Peripheral unit identification name (see **Table 7-2**), n: Peripheral unit number (see **Table 7-2**)).
- ★ (5) When the internal instruction RAM is accessed (in the write mode), the address bus and data bus output data. The external bus control signals other than \overline{UUWR} , \overline{ULWR} , \overline{LUWR} , \overline{LLWR} , and \overline{WR} become active. If output of the \overline{IOWR} signal is enabled by setting the IOEN bit of the bus cycle period control register (BCP) to 1, the \overline{IOWR} signal becomes active.

4.11 Bus Hold Function

4.11.1 Function outline

If the PCM2 and PCM3 pins are specified in the control mode, the $\overline{\text{HLD}}\text{AK}$ and $\overline{\text{HLD}}\text{RQ}$ functions become valid.

If it is determined that the $\overline{\text{HLD}}\text{RQ}$ 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 $\overline{\text{HLD}}\text{RQ}$ 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/ME2 continue until the external memory or a peripheral I/O register is accessed.

The bus hold state can be known by the $\overline{\text{HLD}}\text{AK}$ pin becoming active (low level). The period from when the $\overline{\text{HLD}}\text{RQ}$ pin becomes active (low level) to when the $\overline{\text{HLD}}\text{AK}$ 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 at Which Bus Hold Request Cannot Be Acknowledged
CPU bus lock	32 bits	Word access for 4n+1 address	Between first and second accesses
			Between second and third accesses
		Word access for 4n+2 address	Between first and second accesses
			Word access for 4n+3 address
		Halfword access for odd address	Between first and second accesses
			Between second and third accesses
	16 bits	Word access for even address	Between first and second accesses
			Word access for odd address
		Halfword access for odd address	Between second and third accesses
			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.

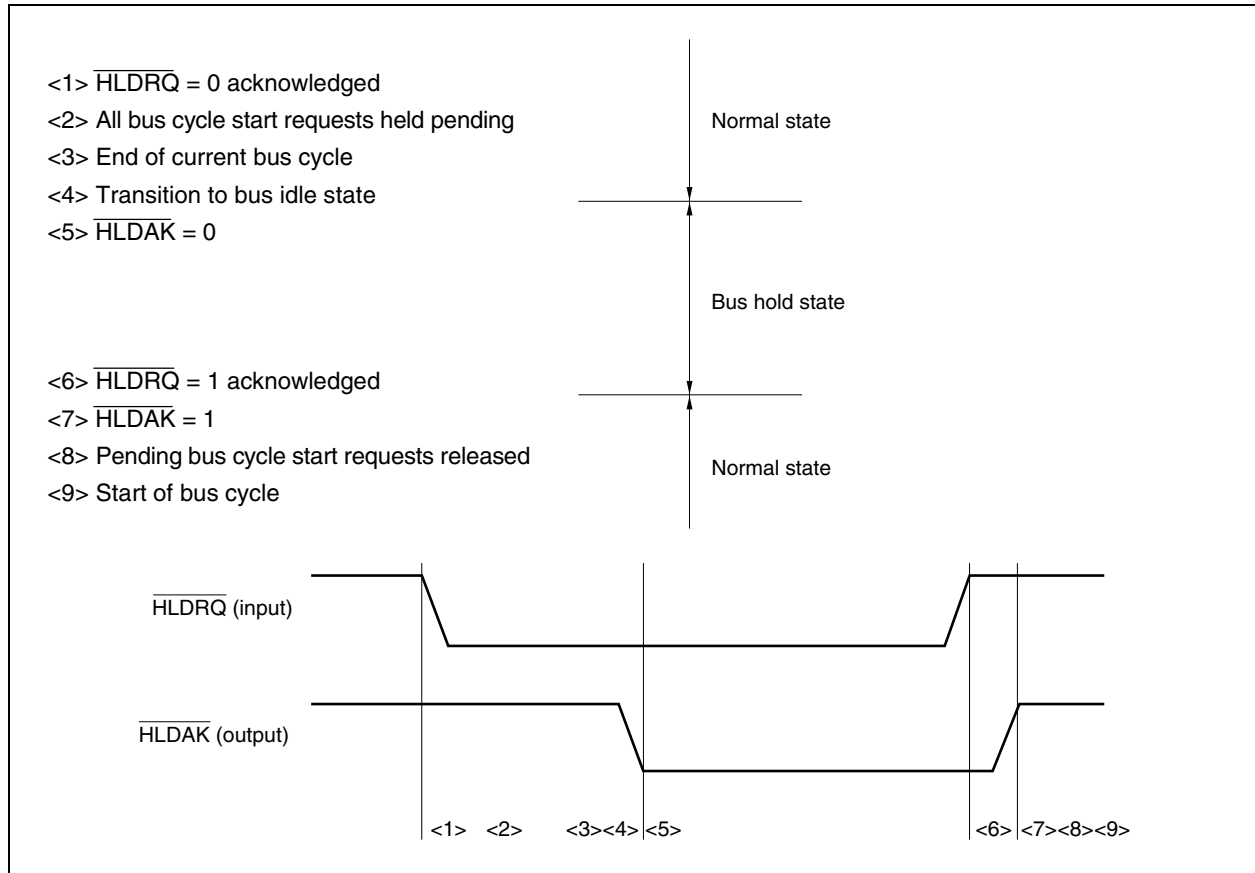
The CPU always executes the all-bank precharge command to release a bus hold state. In a bus hold state, do not allow an external bus master to change the SDRAM command register value.

2. The $\overline{\text{HLD}}\text{RQ}$ function is invalid during a reset period. The $\overline{\text{HLD}}\text{AK}$ 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 $\overline{\text{HLD}}\text{RQ}$ pins.

When a bus master other than the V850E/ME2 is externally connected, use the $\overline{\text{RESET}}$ signal for bus arbitration at power-on.

4.11.2 Bus hold procedure

The procedure of the bus hold function is illustrated below.



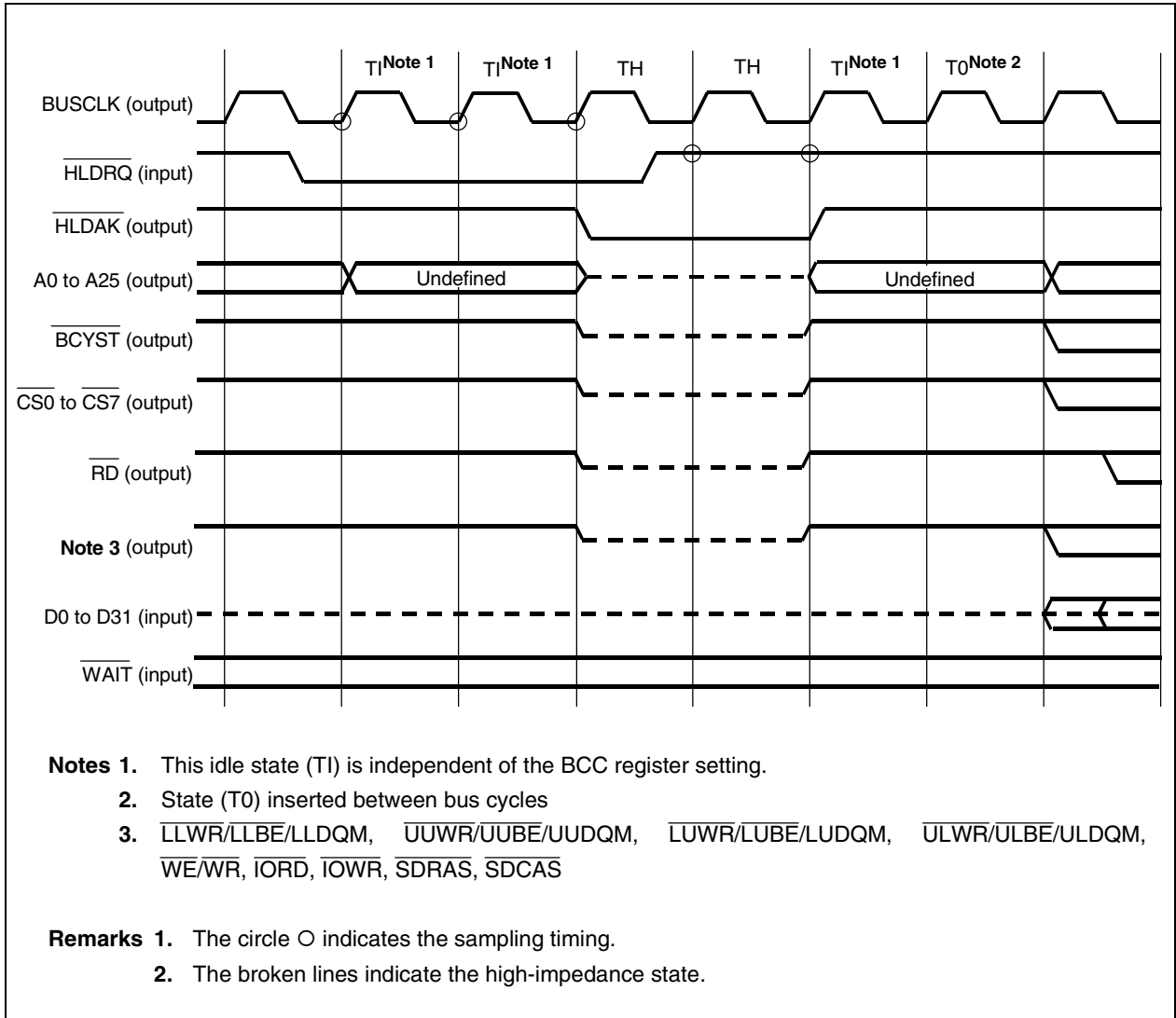
4.11.3 Operation in power-save mode

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

In the HALT mode, the $\overline{\text{HLDAK}}$ 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 $\overline{\text{HLDAK}}$ pin also becomes inactive. As a result, the bus hold state is cleared and the HALT mode is set again.

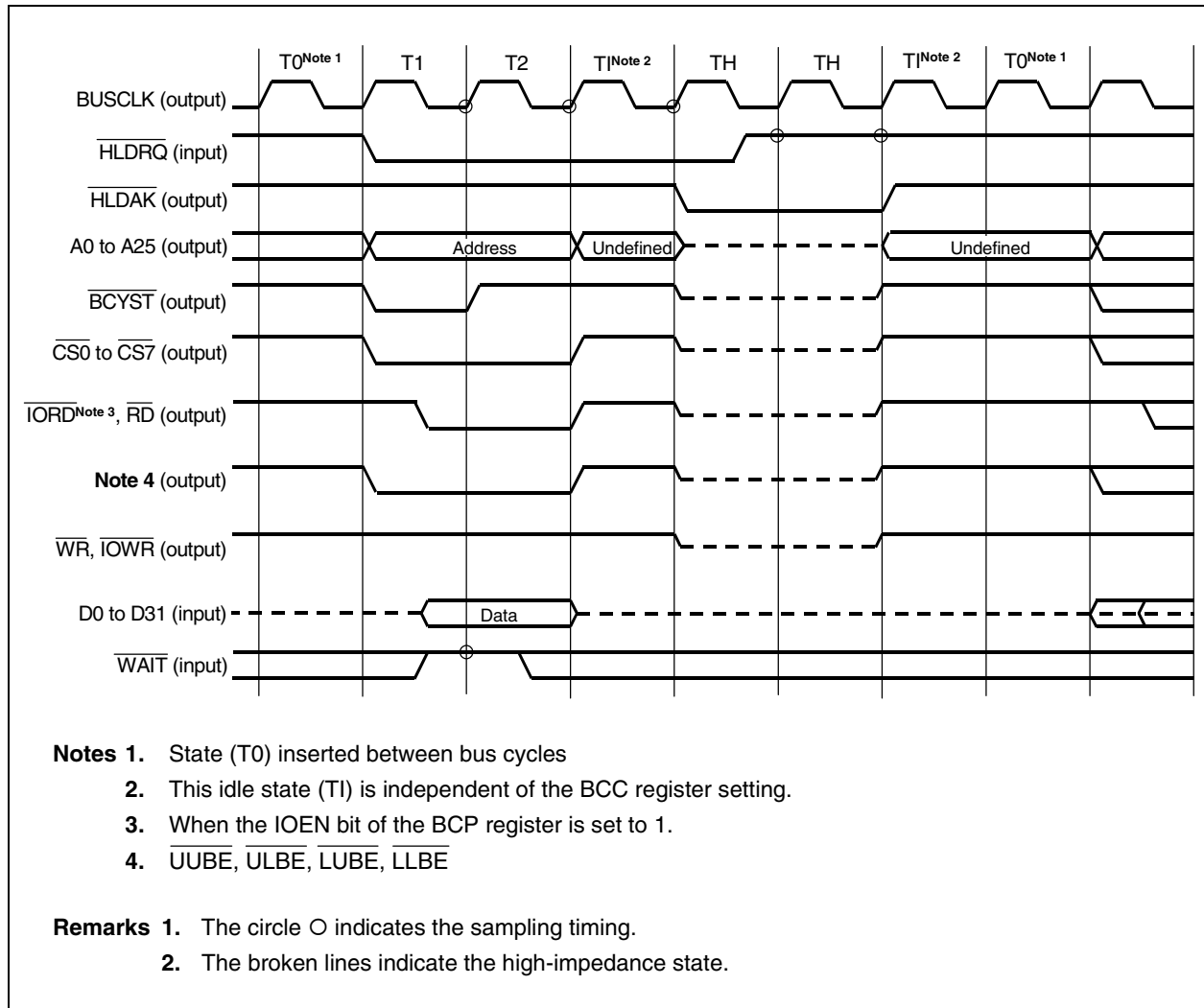
4.11.4 Bus hold timing

(1) If bus hold request is issued when bus cycle is not generated

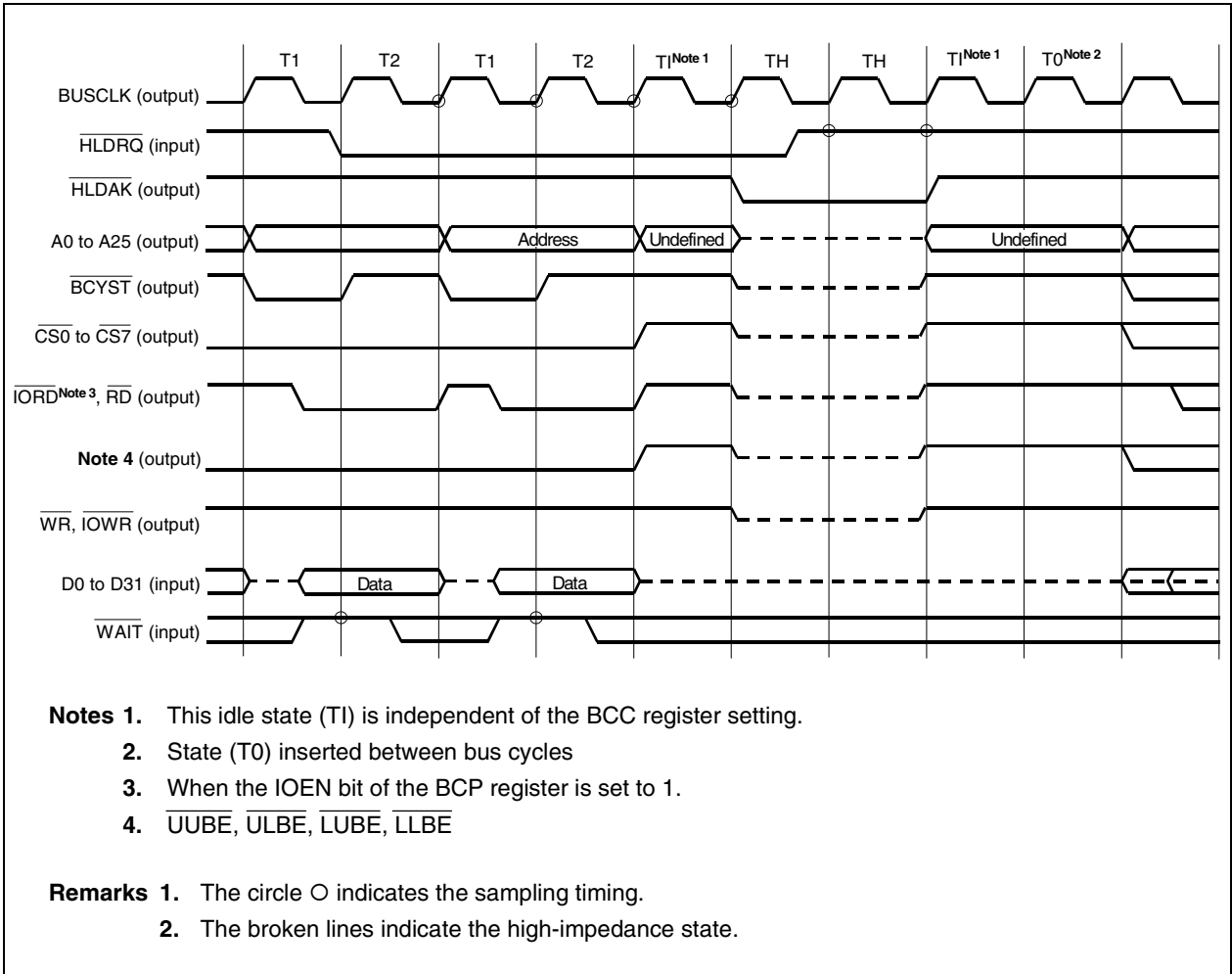


4.11.5 Bus hold timing (SRAM)

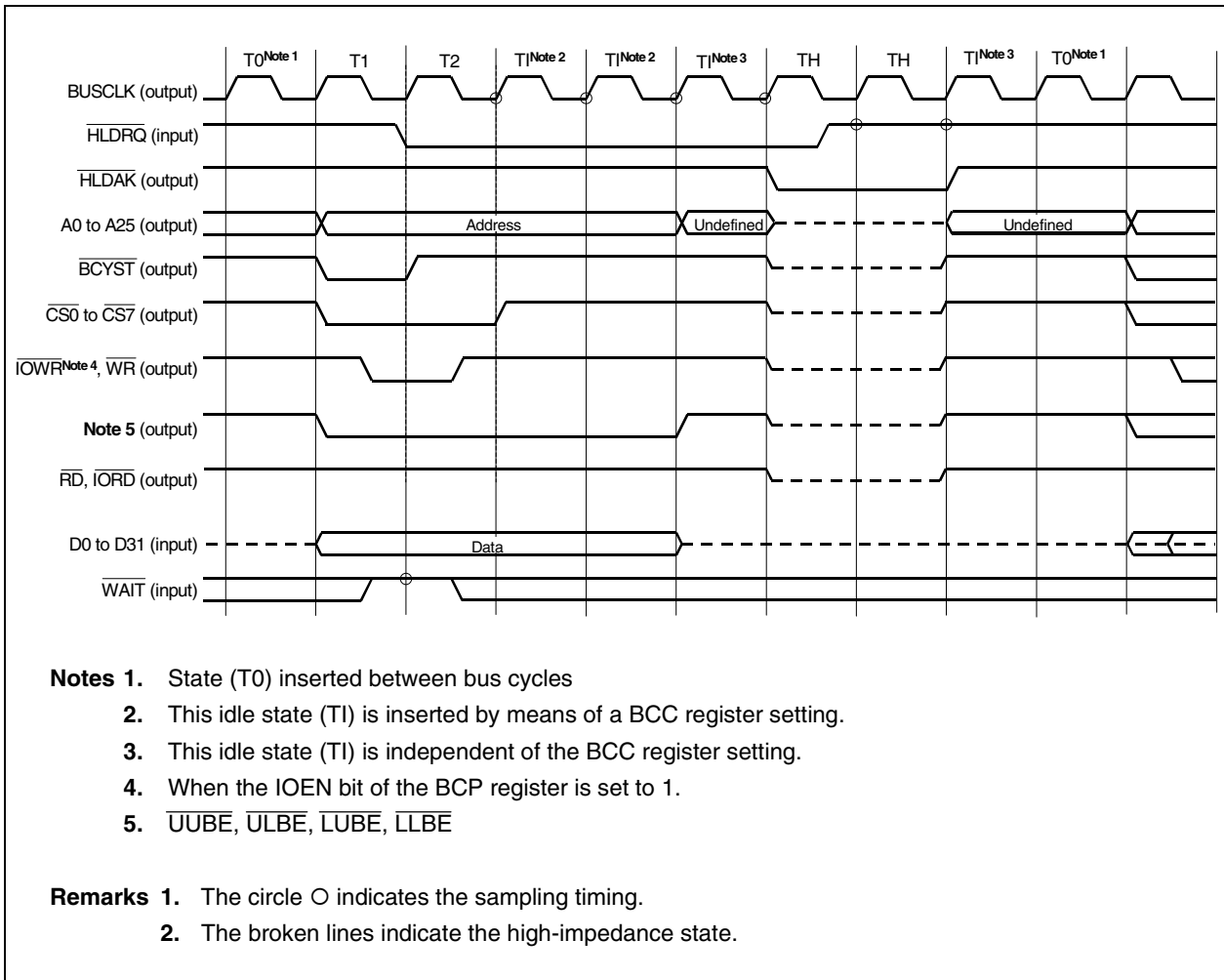
(1) SRAM (when read, without speculative read, no idle state insertion)



(2) SRAM (when read, with speculative read, no idle state insertion)

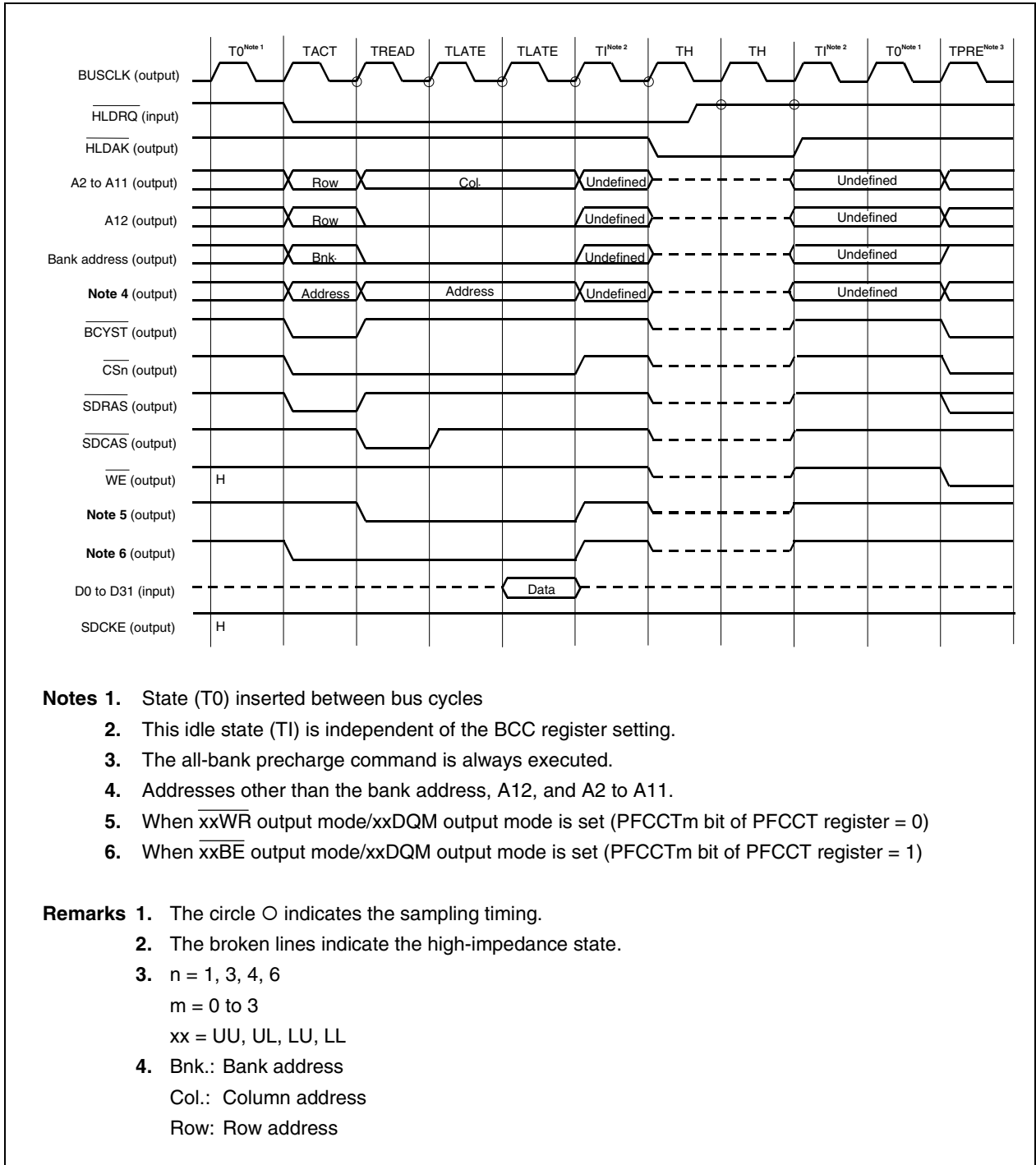


(3) SRAM (when written, two idle states inserted)

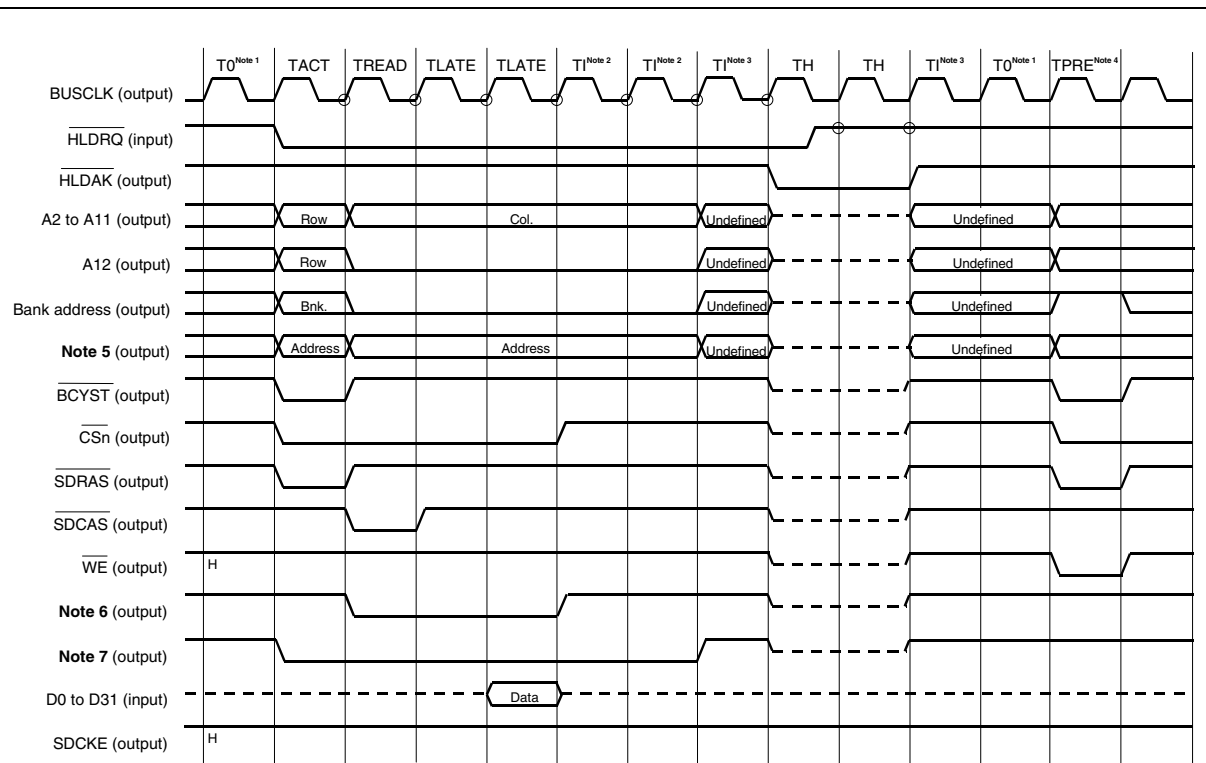


4.11.6 Bus hold timing (SDRAM)

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



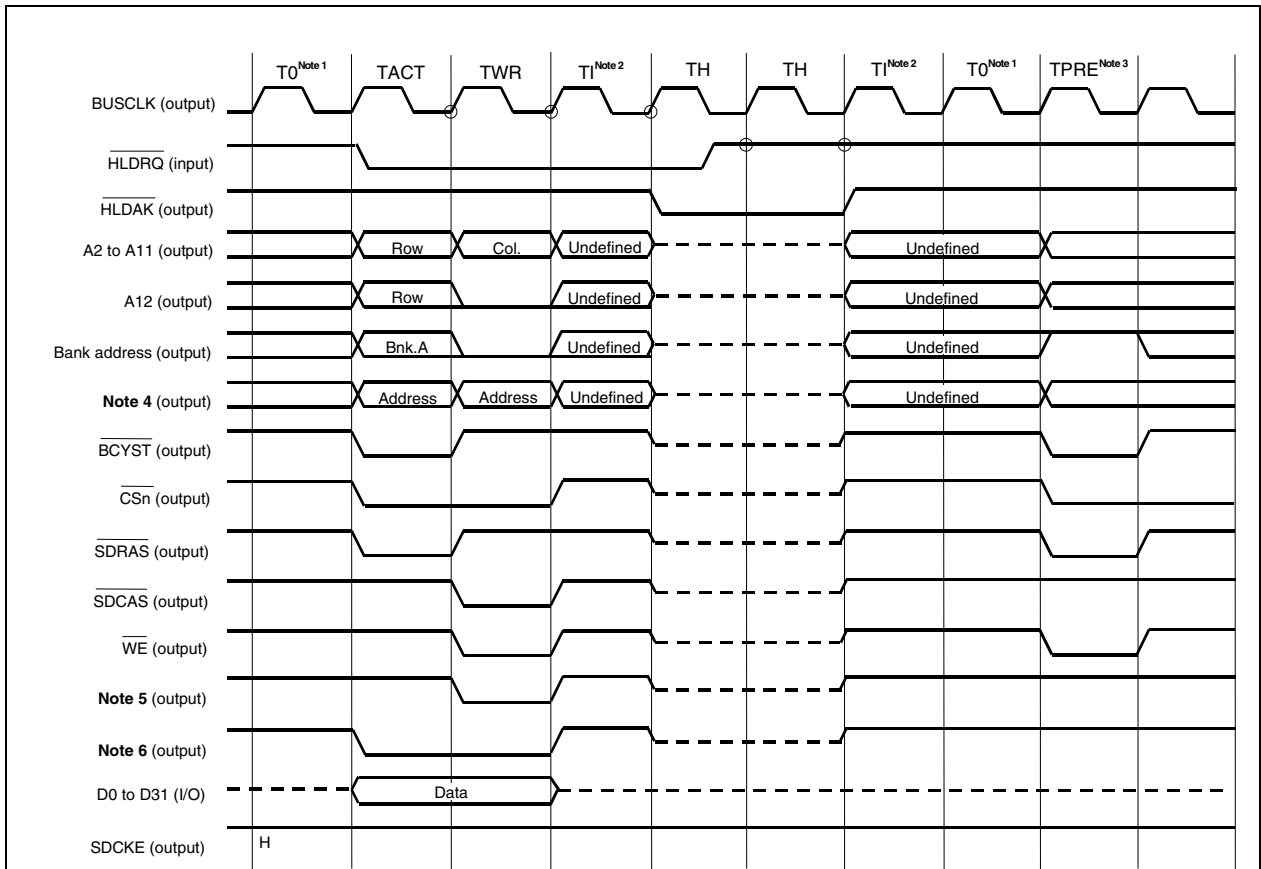
(2) SDRAM (when read, latency = 2, two idle states inserted, 32-bit bus width)



- Notes**
1. State (T0) inserted between bus cycles
 2. This idle state (TI) is inserted by means of a BCC register setting.
 3. This idle state (TI) is independent of the BCC register setting.
 4. The all-bank precharge command is always executed.
 5. Addresses other than the bank address, A12, and A2 to A11.
 6. When \overline{xxWR} output mode/ $xxDQM$ output mode is set (PFCCTm bit of PFCCT register = 0)
 7. When \overline{xxBE} output mode/ $xxDQM$ output mode is set (PFCCTm bit of PFCCT register = 1)

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 1, 3, 4, 6
m = 0 to 3
xx = UU, UL, LU, LL
 4. Bnk.: Bank address
Col.: Column address
Row: Row address

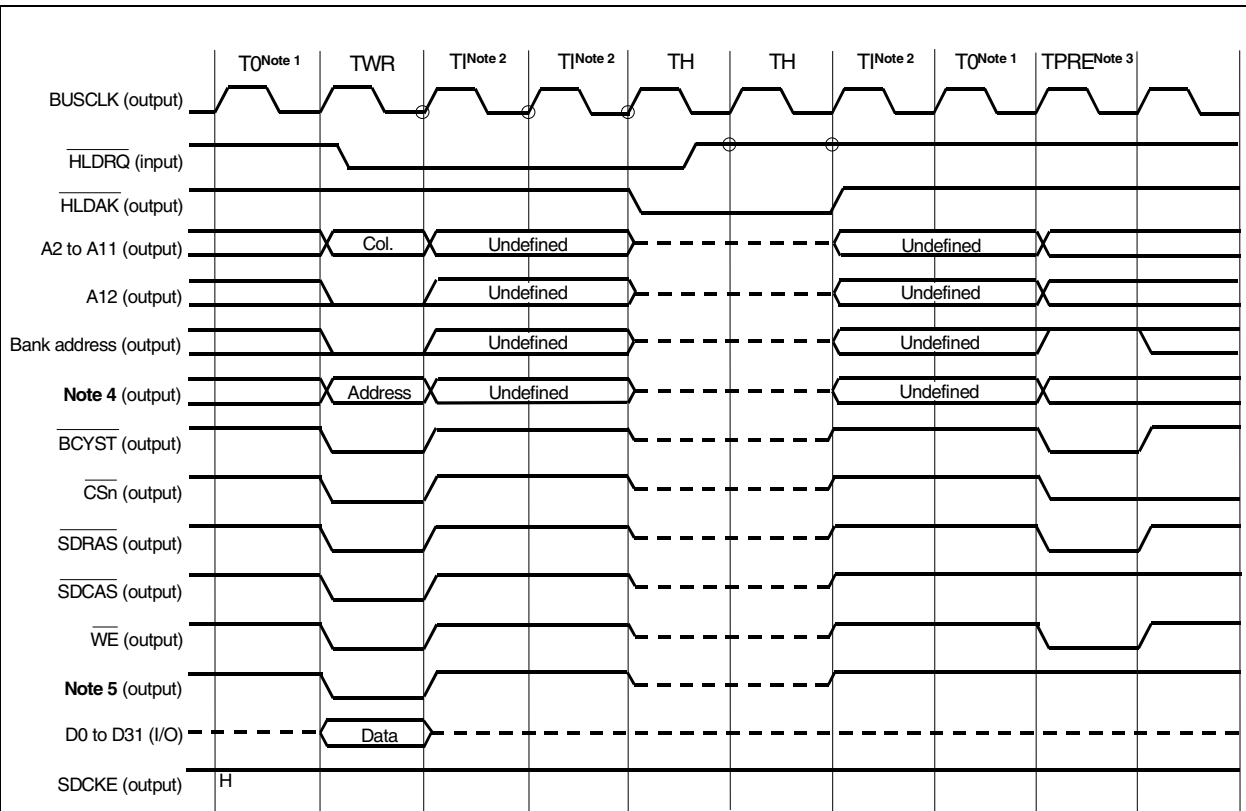
(3) SDRAM (when written)



- Notes**
1. State (T0) inserted between bus cycles
 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, A12, and A2 to A11.
 5. When \overline{xxWR} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 0)
 6. When \overline{xxBE} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 1)

- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 1, 3, 4, 6
m = 0 to 3
xx = UU, UL, LU, LL
 4. Bnk.: Bank address
Col.: Column address
Row: Row address

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



- Notes**
1. State (T0) inserted between bus cycles
 2. This idle state (T1) is independent of the BCC register setting.
 3. The all-bank precharge command is always executed.
 4. Addresses other than the bank address, A12, and A2 to A11.
 5. UUDQM, ULDQM, LUDQM, LLDQM



- Remarks**
1. The circle ○ indicates the sampling timing.
 2. The broken lines indicate the high-impedance state.
 3. n = 1, 3, 4, 6
 4. Col.: Column address

4.12 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 the 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

4.13 Boundary Operation Conditions

4.13.1 Program space

Branching to the on-chip peripheral I/O area is prohibited. If the above is performed, undefined data is fetched, and fetching from the external memory is not performed.

4.13.2 Data space

The V850E/ME2 is provided with an address misalign function.

Through this function, regardless of the data format (word or halfword), 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) External bus width: 16 bits

(a) In the case of halfword-length data access

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

(b) In the case of word-length data access

- (i) 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.
- (ii) When the address's lower 2 bits are 10, a halfword-length bus cycle will be generated 2 times.

(2) External bus width: 32 bits**(a) In the case of halfword-length data access**

When the address's lower 2 bits are 11, a byte-length bus cycle will be generated 2 times.

(b) In the case of word-length data access

When the address's lower 2 bits are 10, a halfword-length bus cycle will be generated 2 times.

★ 4.14 Timing at Which T0 State Is Not Inserted

A T0 state is not inserted at the following timing.

(1) Read

- During a refill operation when instruction cache is used (in the second and subsequent cycles)
- During speculative reading (in the second and subsequent cycles)
- If a read request (including a read request by instruction fetch or DMA) is generated before a write operation to the external device is completed when a read operation occurs immediately after a write operation
- If a read request (including a read request by instruction fetch or DMA) that is greater than the bus width of the external device is generated (in the second and subsequent cycles)

[Example]

- 32-bit read access with 16-bit external bus width
- 32-bit or 16-bit read access with 8-bit external bus width

(2) Write

- If a write request (including a write request by DMA) is generated before a write operation to the external device is completed when a write operation occurs immediately after a write operation, and data is stored in the write buffer (a cycle in which the data stored in the write buffer is written to the external device)
- If a write request (including a write request by DMA) is generated during a speculative read operation (if data is stored in the write buffer)
- If a write request (including a write request by DMA) greater than the bus width of the external bus is generated (after the second and subsequent cycles)

[Example]

- 32-bit write access with external 16-bit bus width
- 32-bit or 16-bit write access with external 8-bit bus

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 (DMA flyby transfer: FWC register).
- 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 (DMA flyby transfer: FIC register).
- Up to 3 address setup wait states can be inserted by setting the ASC register.
- DMA flyby transfer can be activated (SRAM → external I/O, external I/O → SRAM)

5.1.2 SRAM connection

Examples of connection to SRAM are shown below.

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

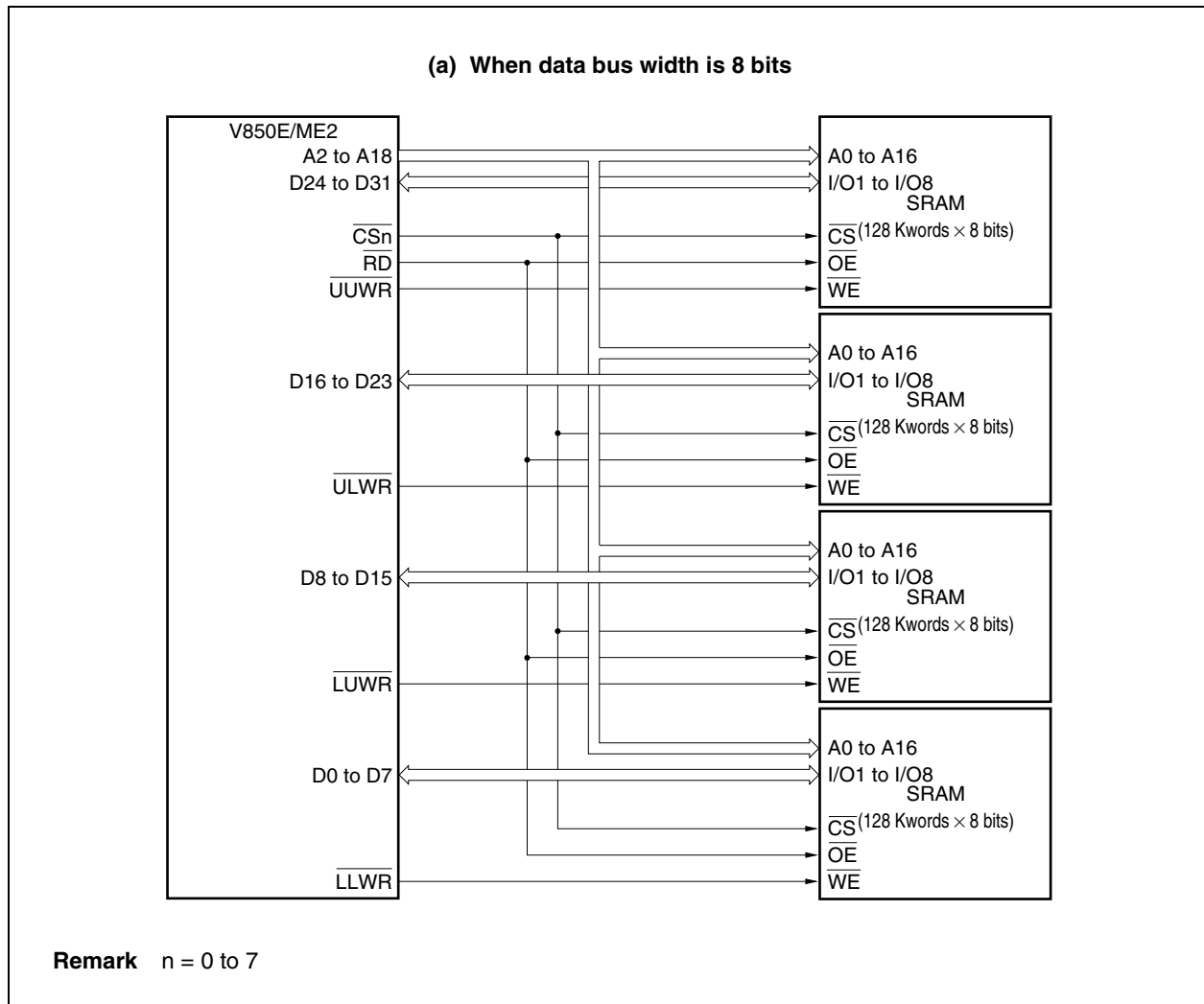
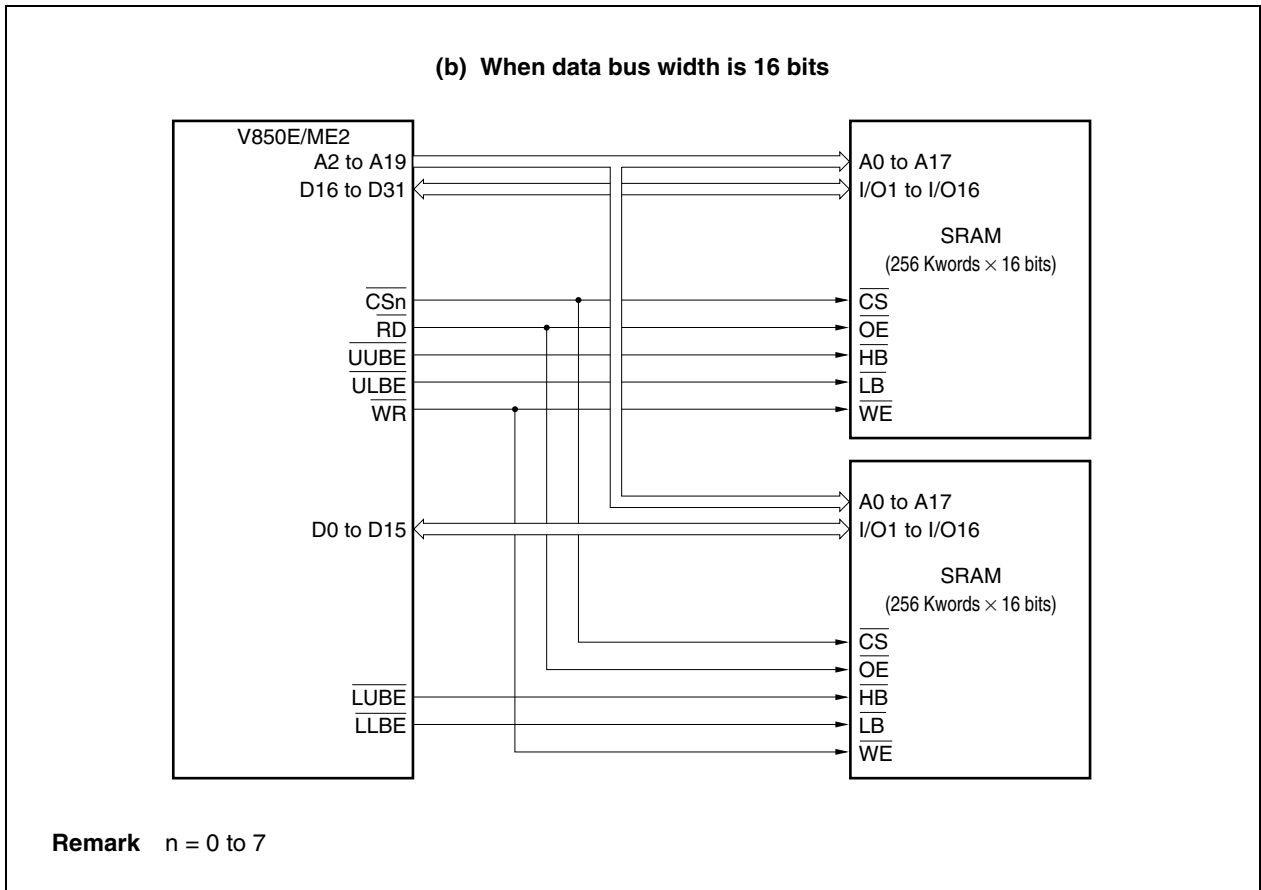


Figure 5-1. Examples of Connection to SRAM (2/2)



5.1.3 SRAM, external ROM, external I/O access

Figure 5-2. SRAM, External ROM, External I/O Access Timing (1/10)

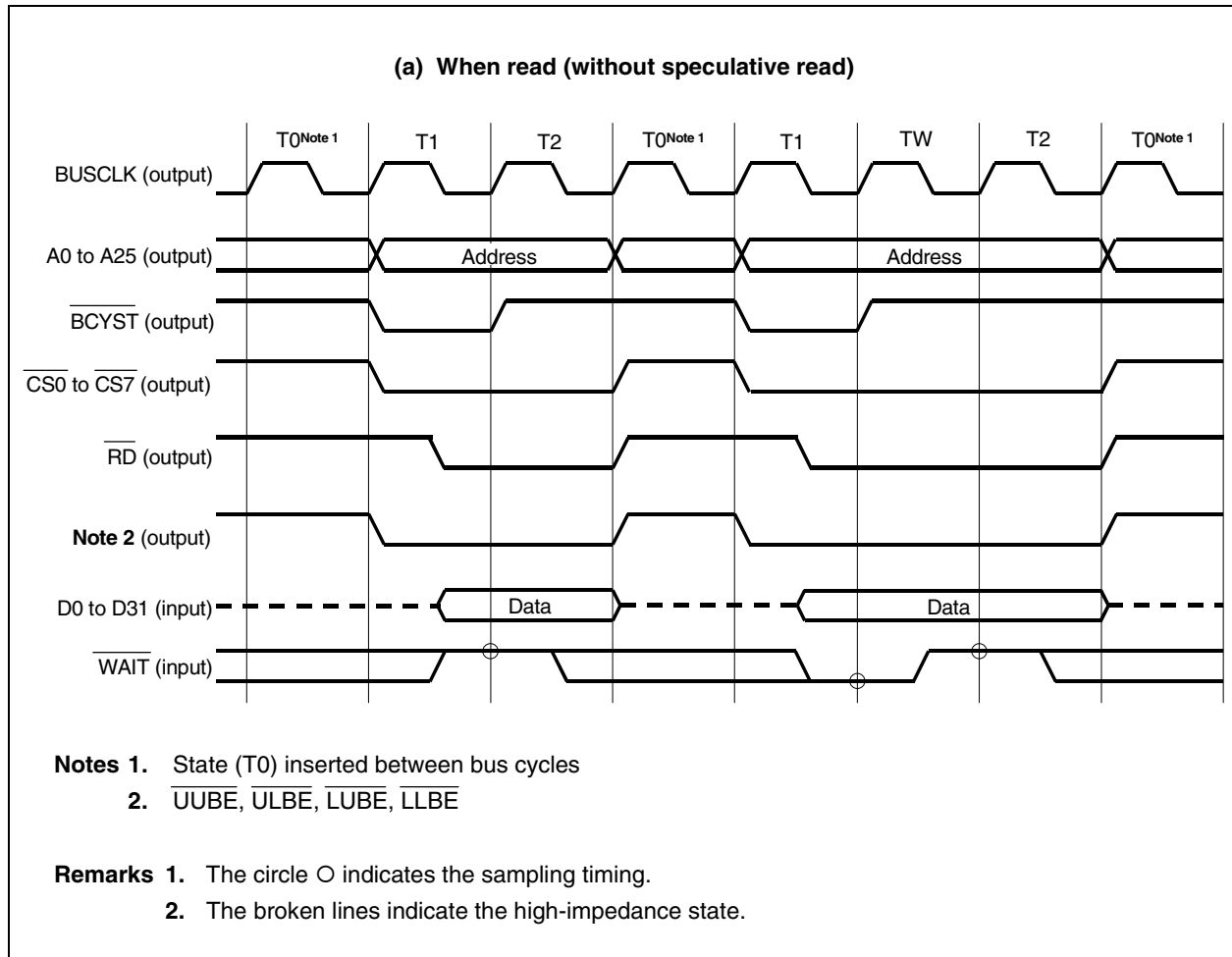


Figure 5-2. SRAM, External ROM, External I/O Access Timing (2/10)

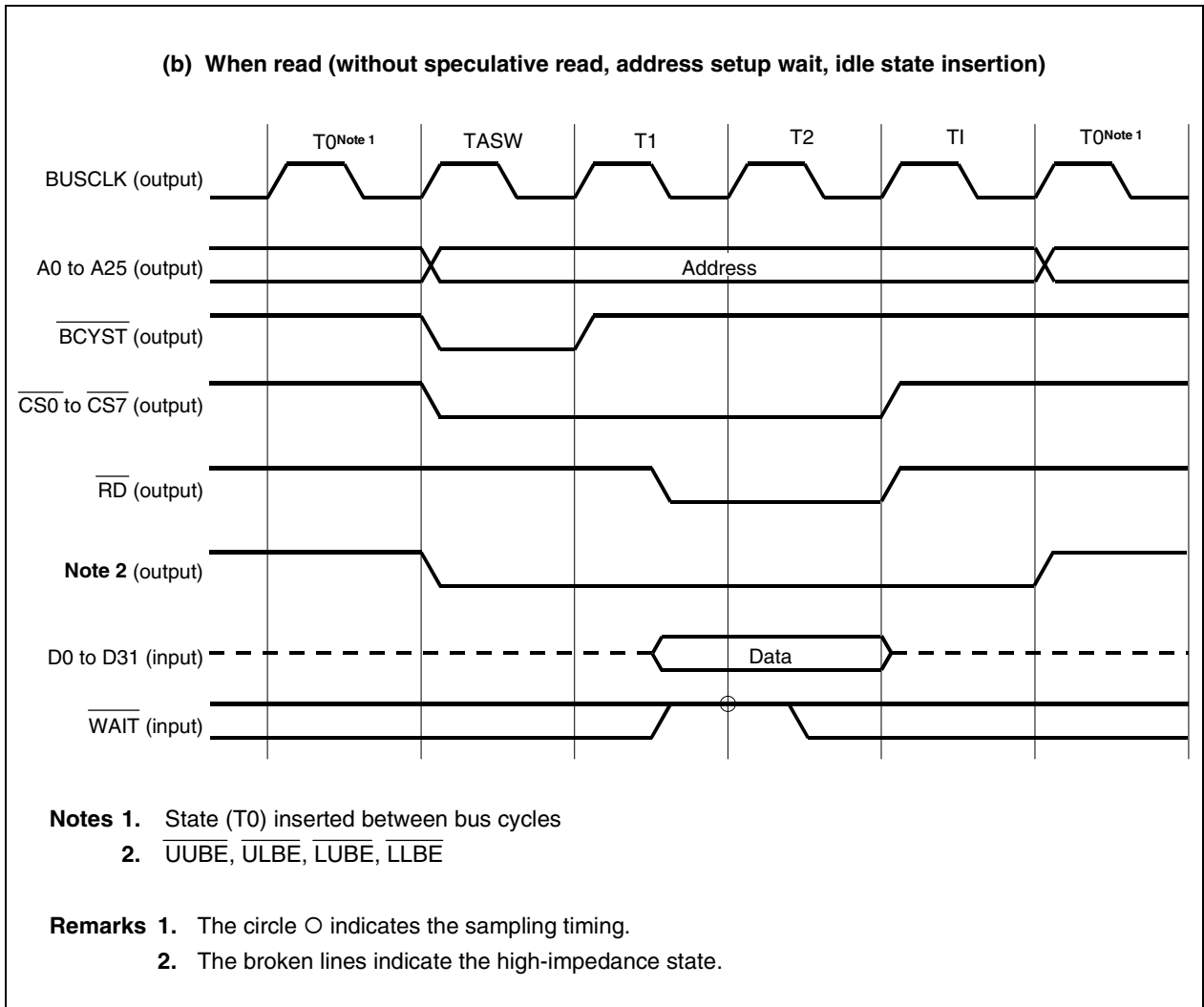


Figure 5-2. SRAM, External ROM, External I/O Access Timing (3/10)

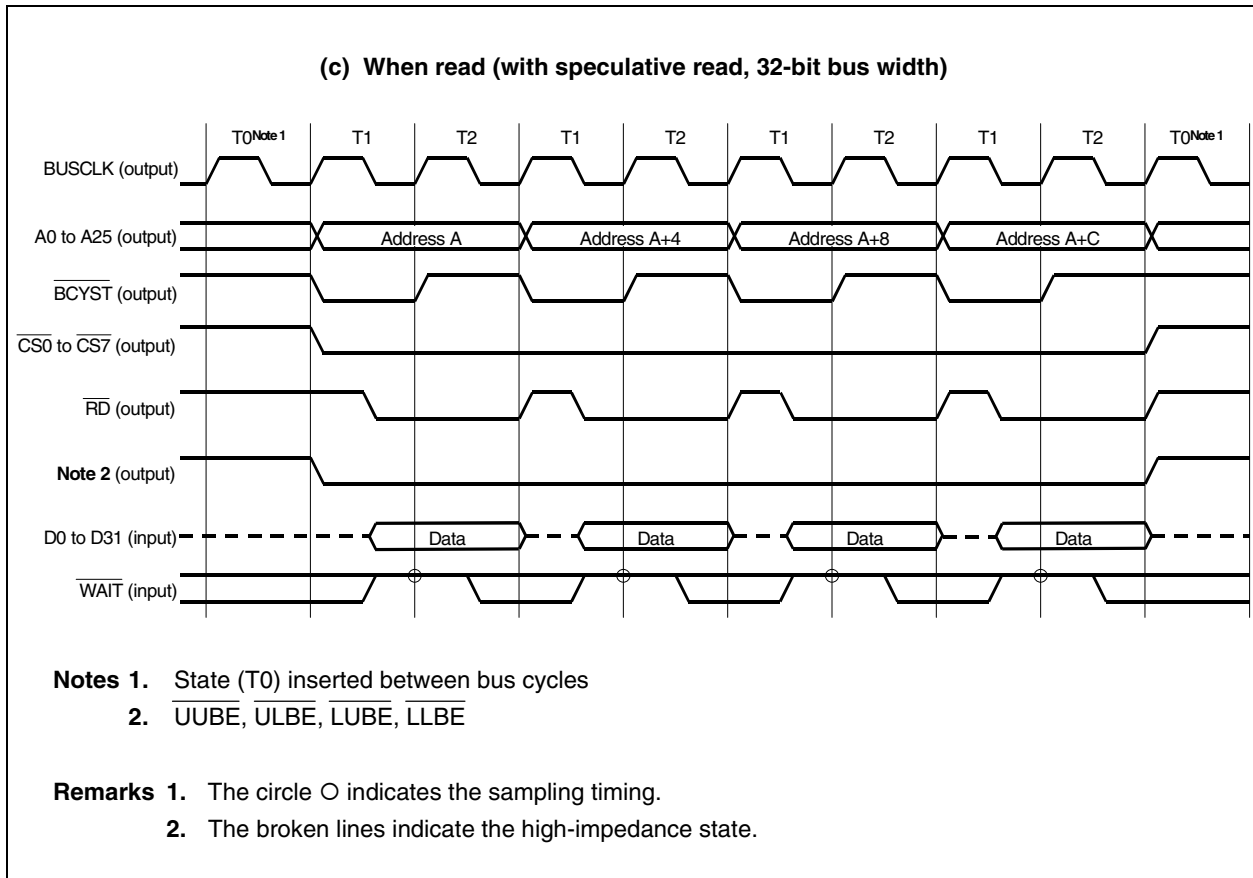


Figure 5-2. SRAM, External ROM, External I/O Access Timing (4/10)

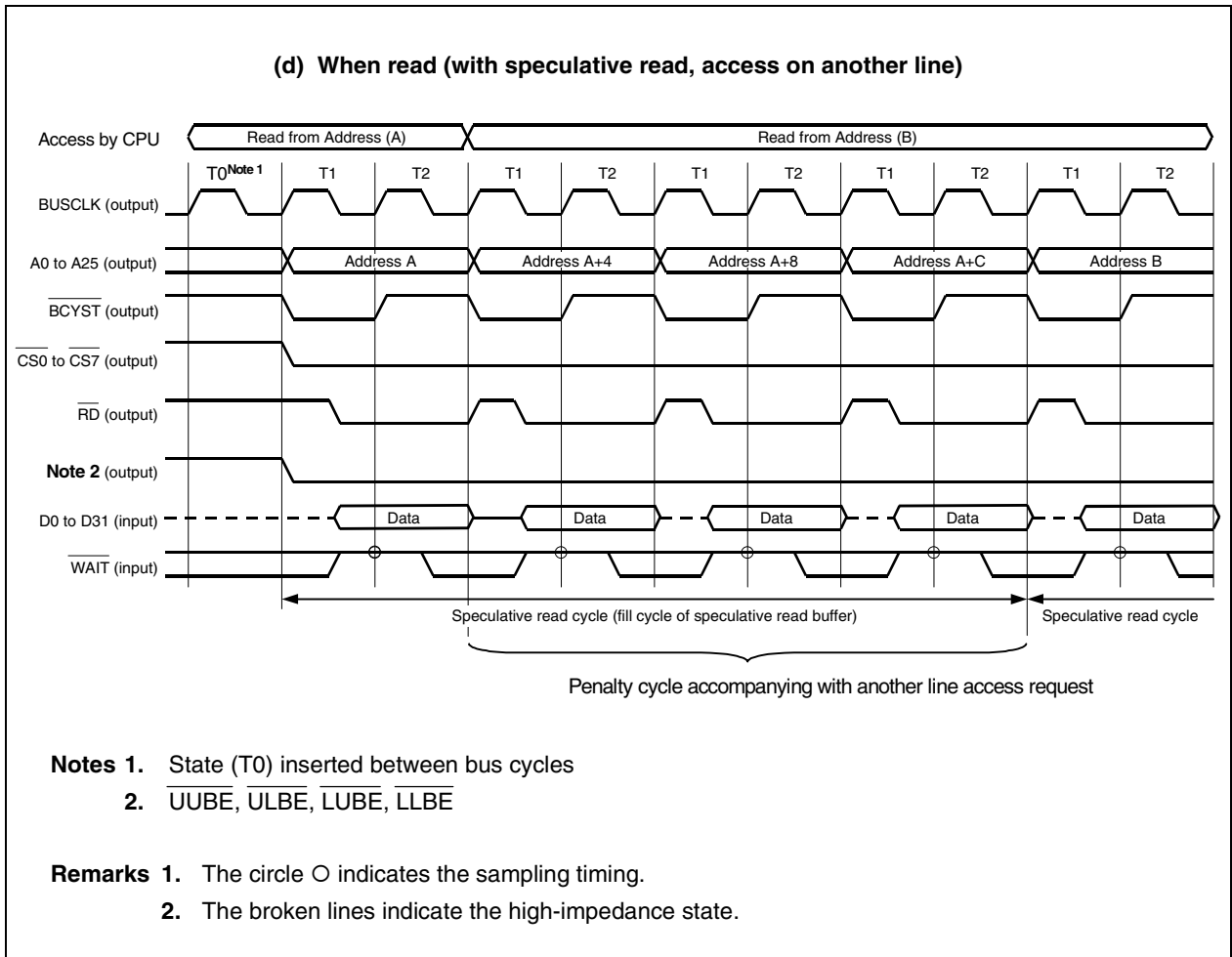


Figure 5-2. SRAM, External ROM, External I/O Access Timing (5/10)

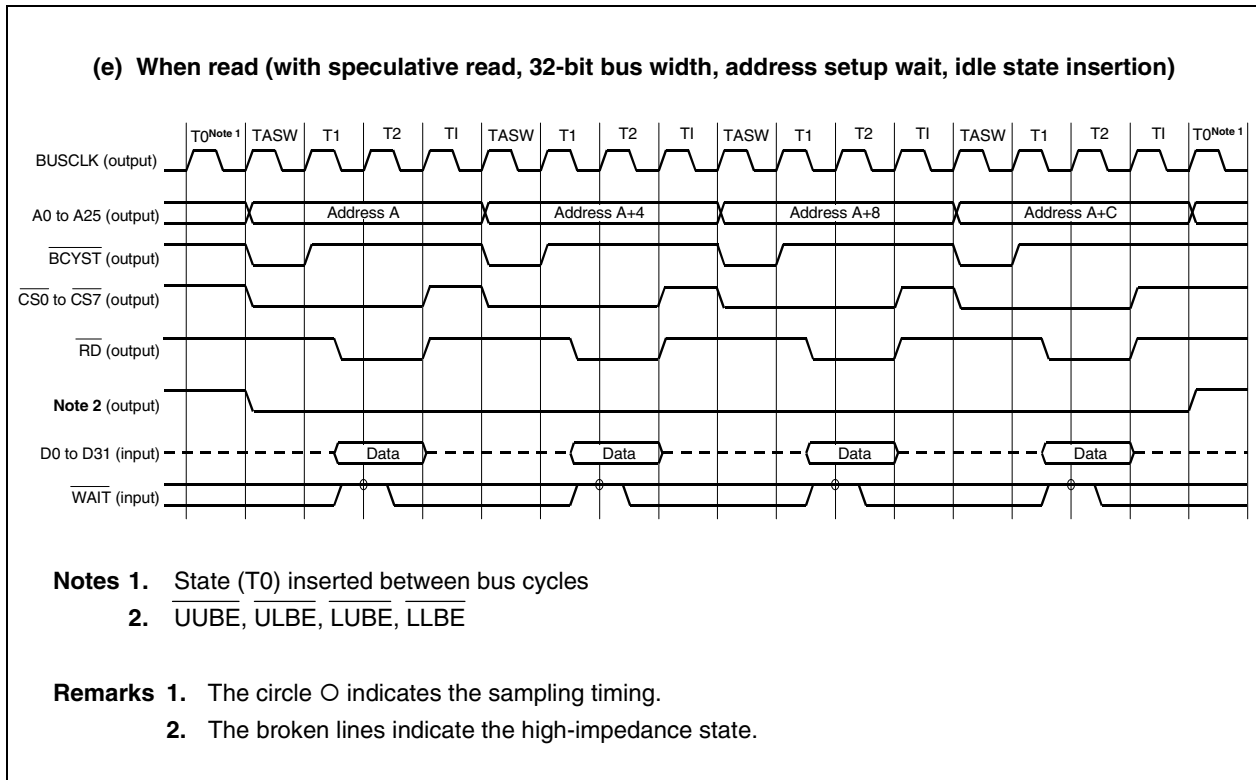


Figure 5-2. SRAM, External ROM, External I/O Access Timing (6/10)

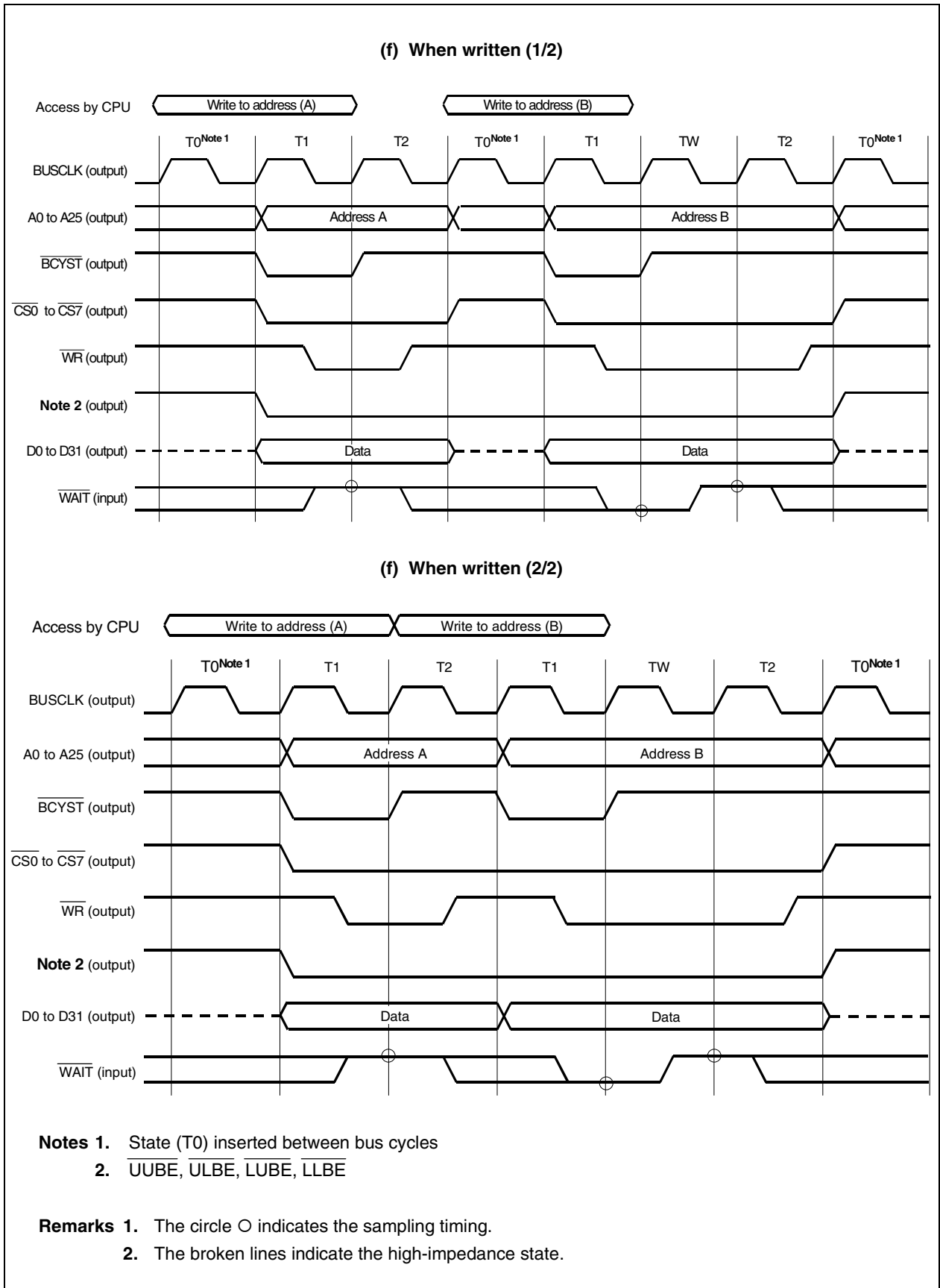


Figure 5-2. SRAM, External ROM, External I/O Access Timing (7/10)

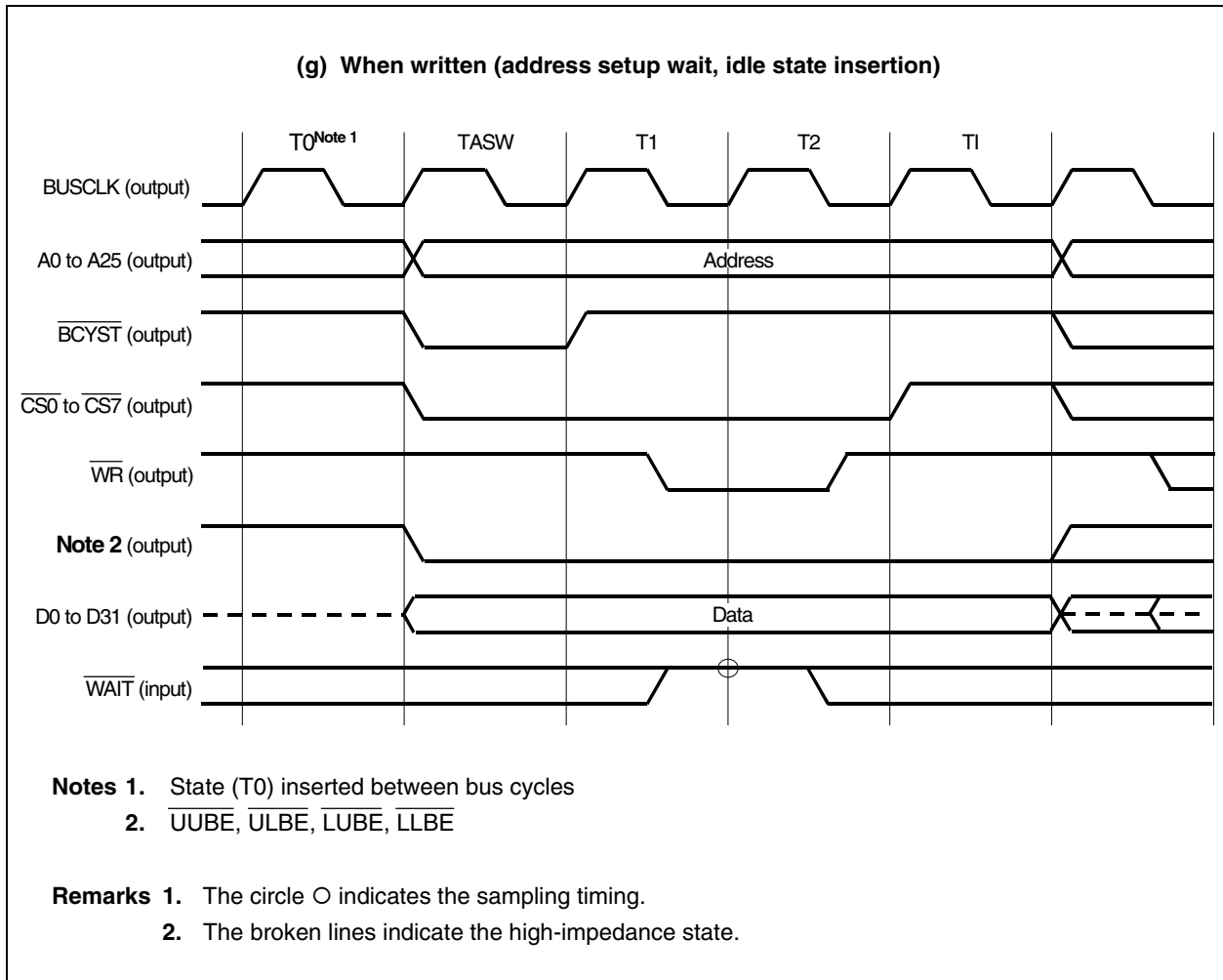


Figure 5-2. SRAM, External ROM, External I/O Access Timing (8/10)

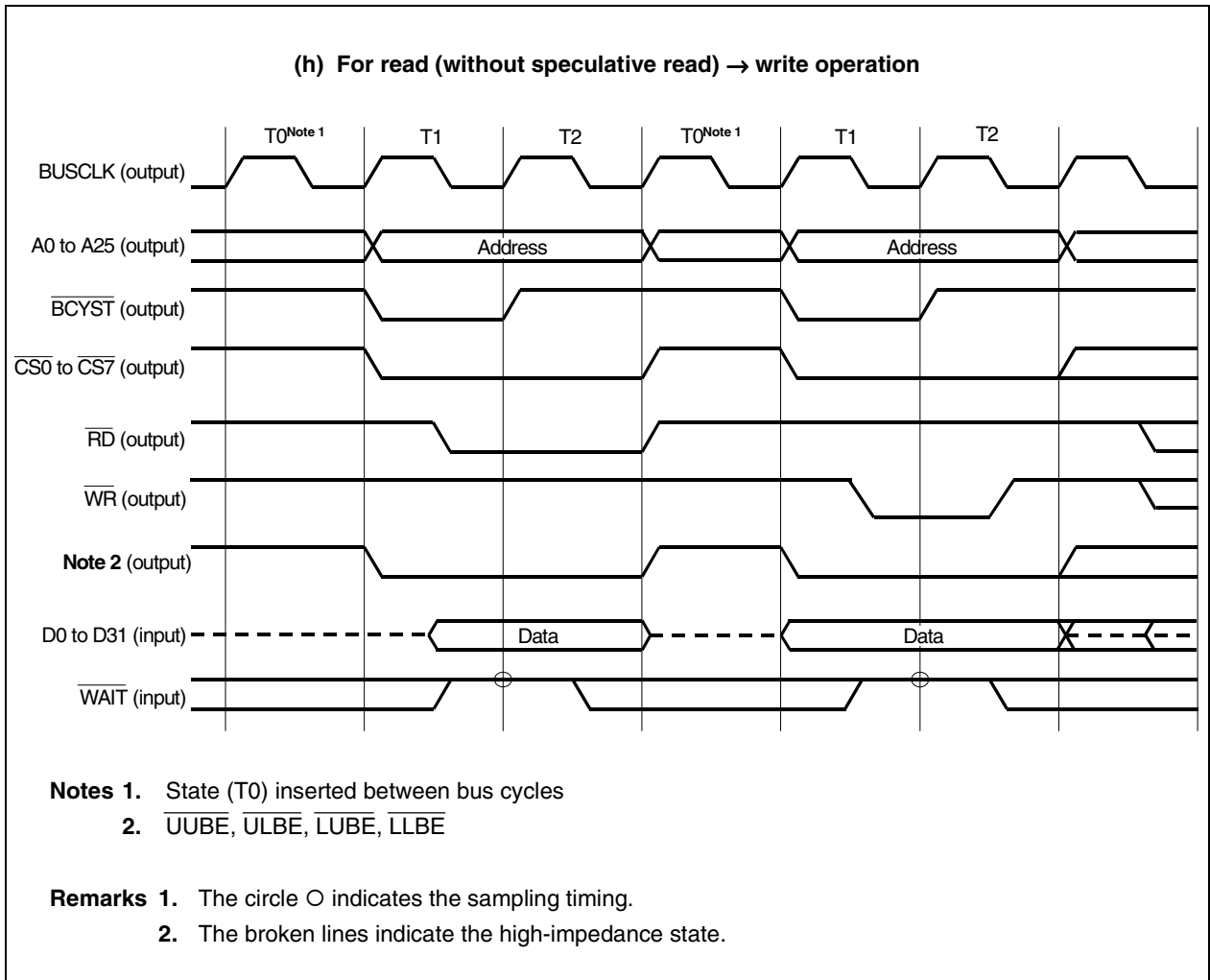


Figure 5-2. SRAM, External ROM, External I/O Access Timing (9/10)

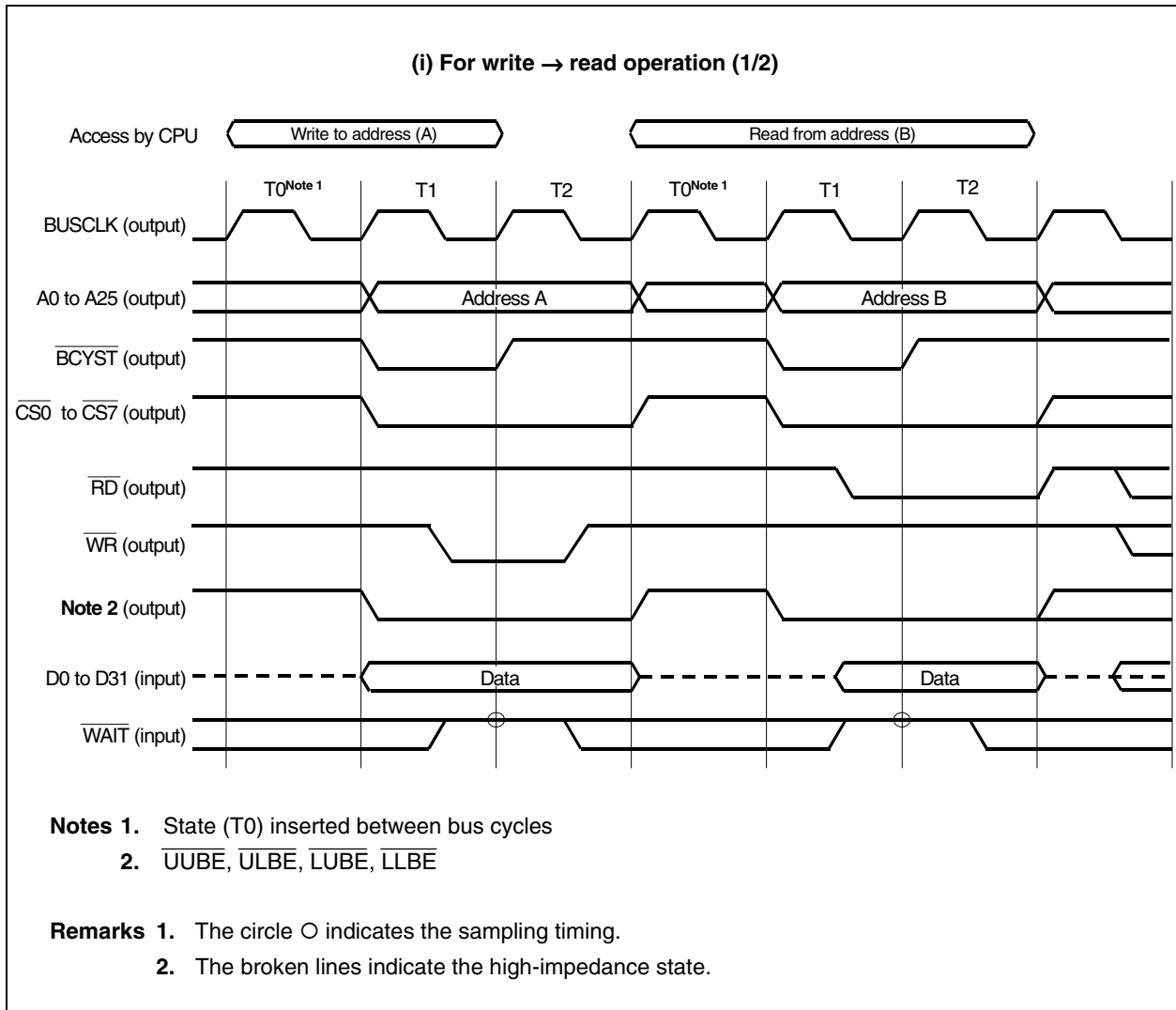
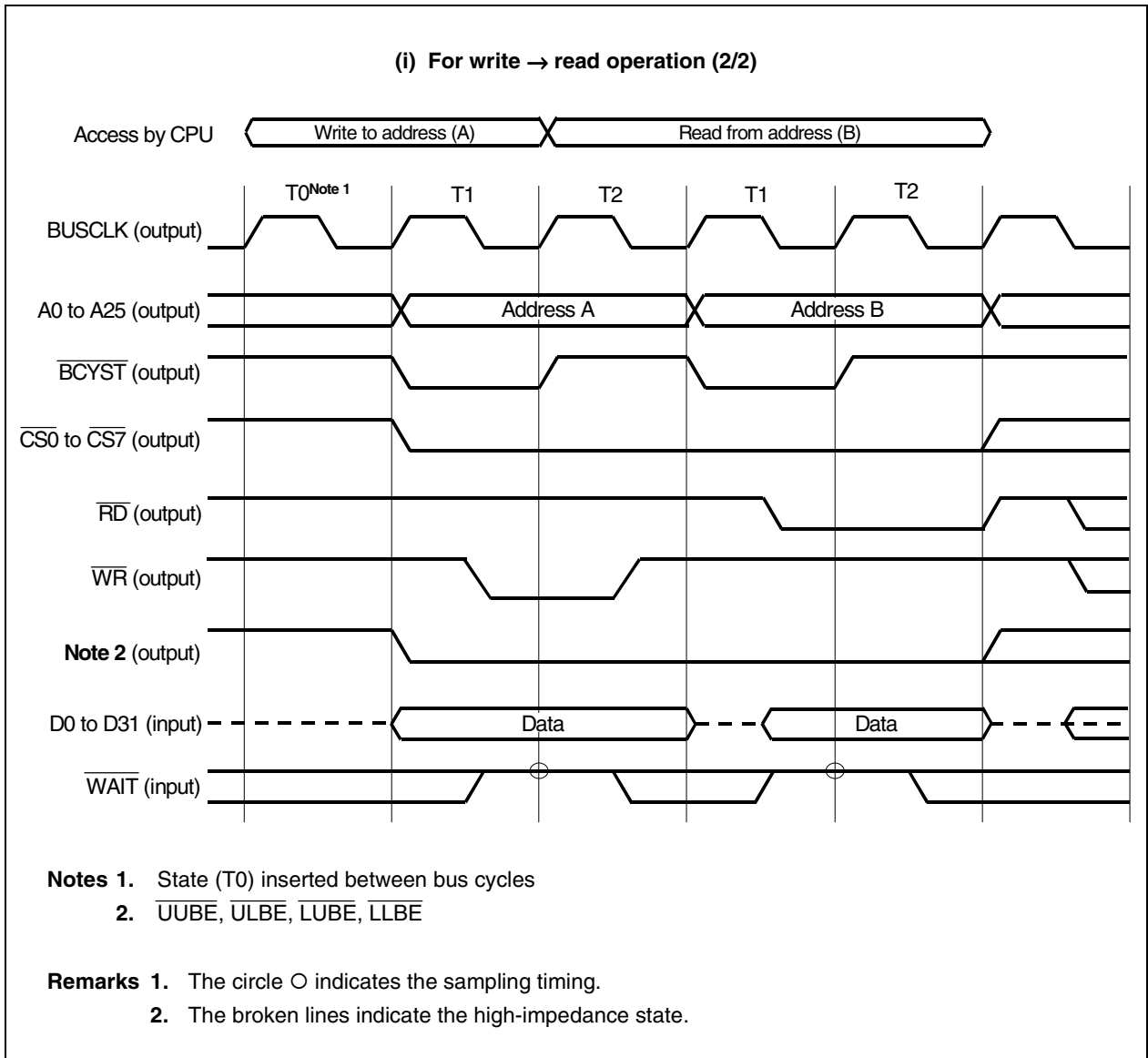


Figure 5-2. SRAM, External ROM, External I/O Access Timing (10/10)



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.

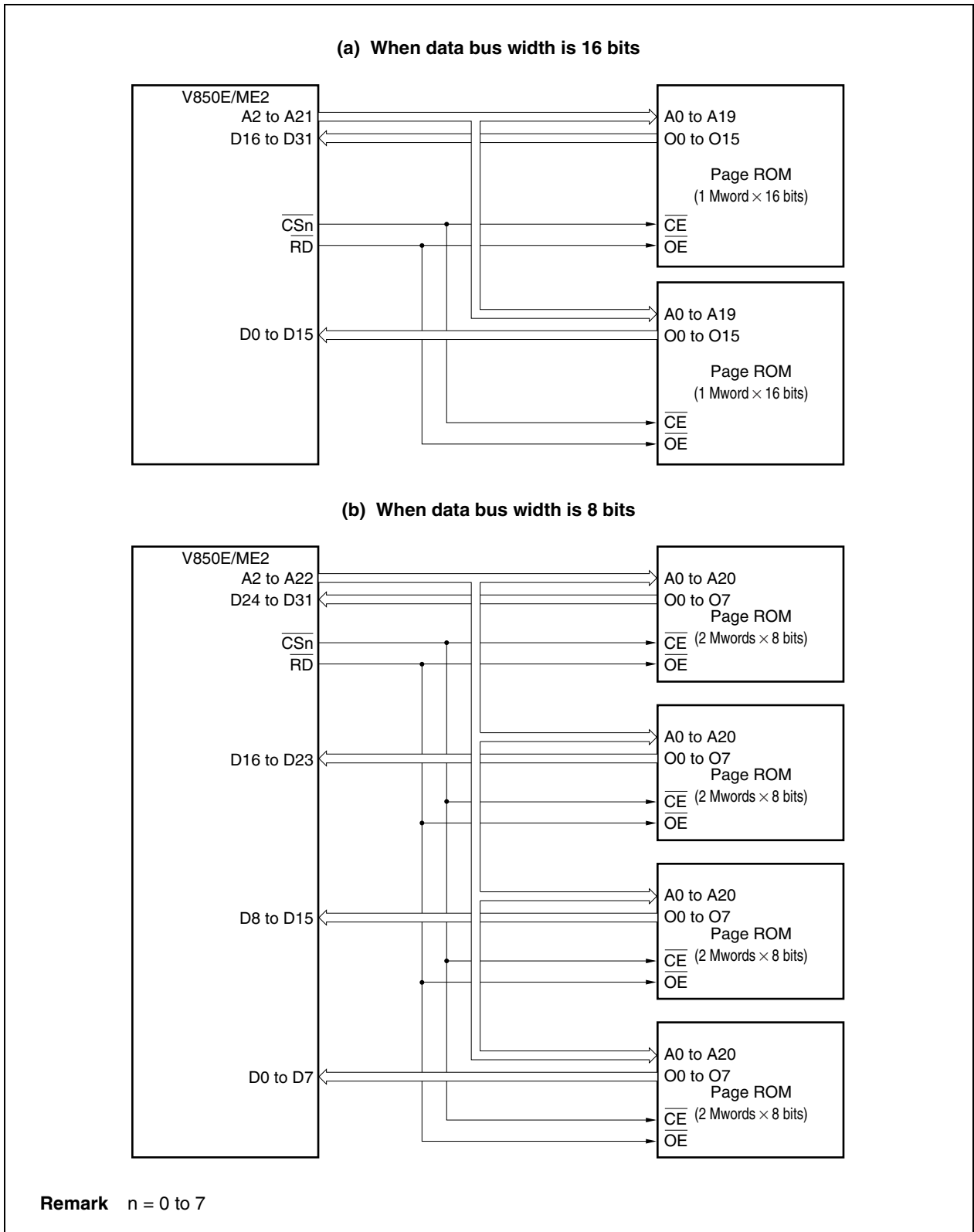
5.2.1 Features

- Direct connection to 8-bit/16-bit/32-bit page ROM supported
- For 32-bit bus width: 2/4/8/16/32-word page access 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 by setting the following registers.
 - During on-page cycle: PRC register
 - During off-page cycle: DWC0 and DWC1 registers
 - During DMA flyby cycle: FWC register
- Waits can be controlled via $\overline{\text{WAIT}}$ pin input.
- DMA flyby cycle can be activated (page ROM → external I/O)
- SRAM write cycle is started when a write cycle request is issued to a CS_n space (n = 0 to 7) where page ROM is located

5.2.2 Page ROM connection

Examples of connection to page ROM are shown below.

Figure 5-3. Examples of Connection to Page ROM



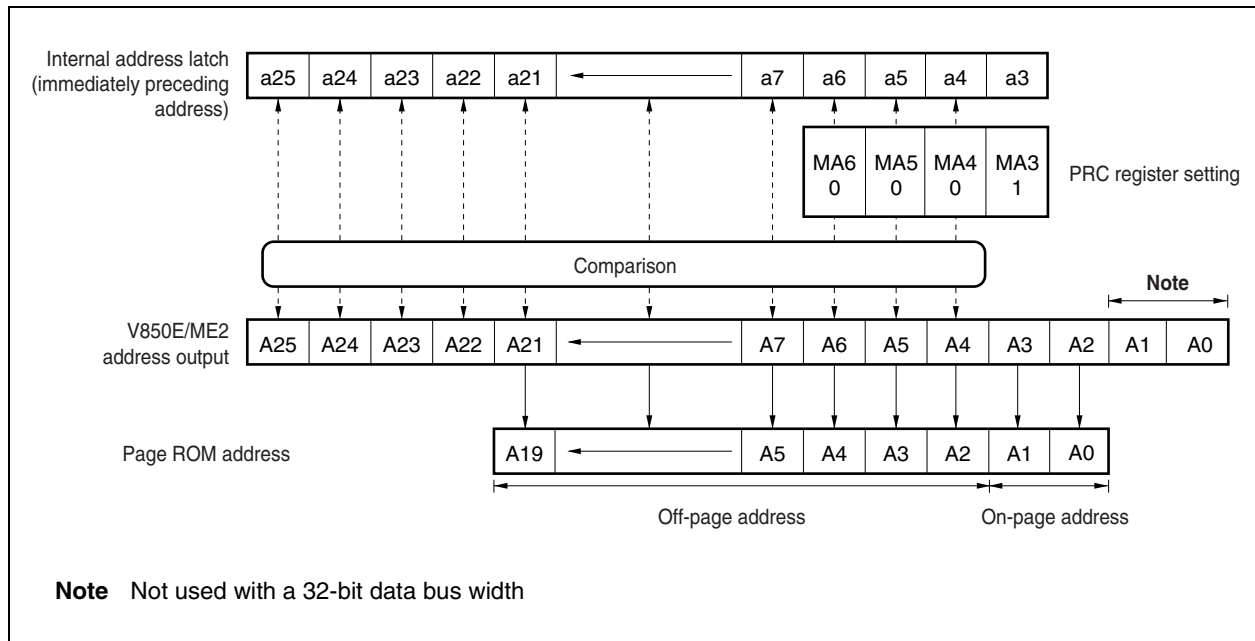
5.2.3 On-page/off-page judgment

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. If no speculative reading is specified (RBN1 and RBN0 bits of the LBCm register = 00, 01), an off-page cycle is always started (m = 0, 1, n = 0 to 7).

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

An example of controlling address masking when four page ROMs of 1 Mword × 8 bits are connected is shown below.

Figure 5-4. Example of Control by MA6 to MA3 Bits of PRC Register



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 or 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 FFFF49AH	After reset 7000H	

Bit position	Bit name	Function																																				
14 to 12	PRW2 to PRW0	<p>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.</p> <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	<p>Each respective address (A6 to A3) corresponding to MA6 to MA3 is 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.</p> <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;">2 × 32 bits, 4 × 16 bits, 8 × 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;">4 × 32 bits, 8 × 16 bits, 16 × 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;">8 × 32 bits, 16 × 16 bits, 32 × 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;">16 × 32 bits, 32 × 16 bits, 64 × 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;">32 × 32 bits, 64 × 16 bits, 128 × 8 bits</td></tr> <tr> <td colspan="4" style="text-align: center;">Other than above</td> <td style="text-align: center;">Setting prohibited (The operation cannot be guaranteed if other bits are set.)</td> </tr> </tbody> </table>	MA6	MA5	MA4	MA3	Number of continuously readable bits	0	0	0	0	2 × 32 bits, 4 × 16 bits, 8 × 8 bits	0	0	0	1	4 × 32 bits, 8 × 16 bits, 16 × 8 bits	0	0	1	1	8 × 32 bits, 16 × 16 bits, 32 × 8 bits	0	1	1	1	16 × 32 bits, 32 × 16 bits, 64 × 8 bits	1	1	1	1	32 × 32 bits, 64 × 16 bits, 128 × 8 bits	Other than above				Setting prohibited (The operation cannot be guaranteed if other bits are set.)	
MA6	MA5	MA4	MA3	Number of continuously readable bits																																		
0	0	0	0	2 × 32 bits, 4 × 16 bits, 8 × 8 bits																																		
0	0	0	1	4 × 32 bits, 8 × 16 bits, 16 × 8 bits																																		
0	0	1	1	8 × 32 bits, 16 × 16 bits, 32 × 8 bits																																		
0	1	1	1	16 × 32 bits, 32 × 16 bits, 64 × 8 bits																																		
1	1	1	1	32 × 32 bits, 64 × 16 bits, 128 × 8 bits																																		
Other than above				Setting prohibited (The operation cannot be guaranteed if other bits are set.)																																		

5.2.5 Page ROM access

Figure 5-5. Page ROM Access Timing (1/6)

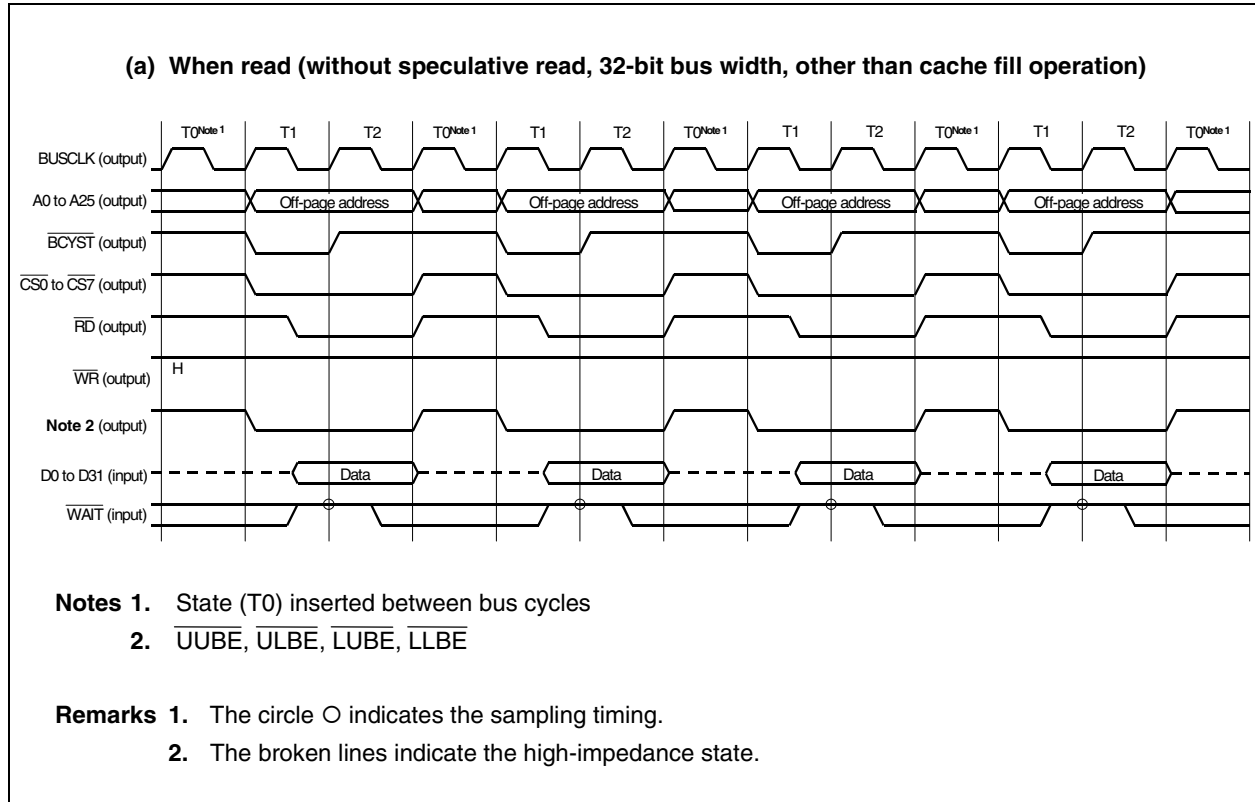


Figure 5-5. Page ROM Access Timing (2/6)

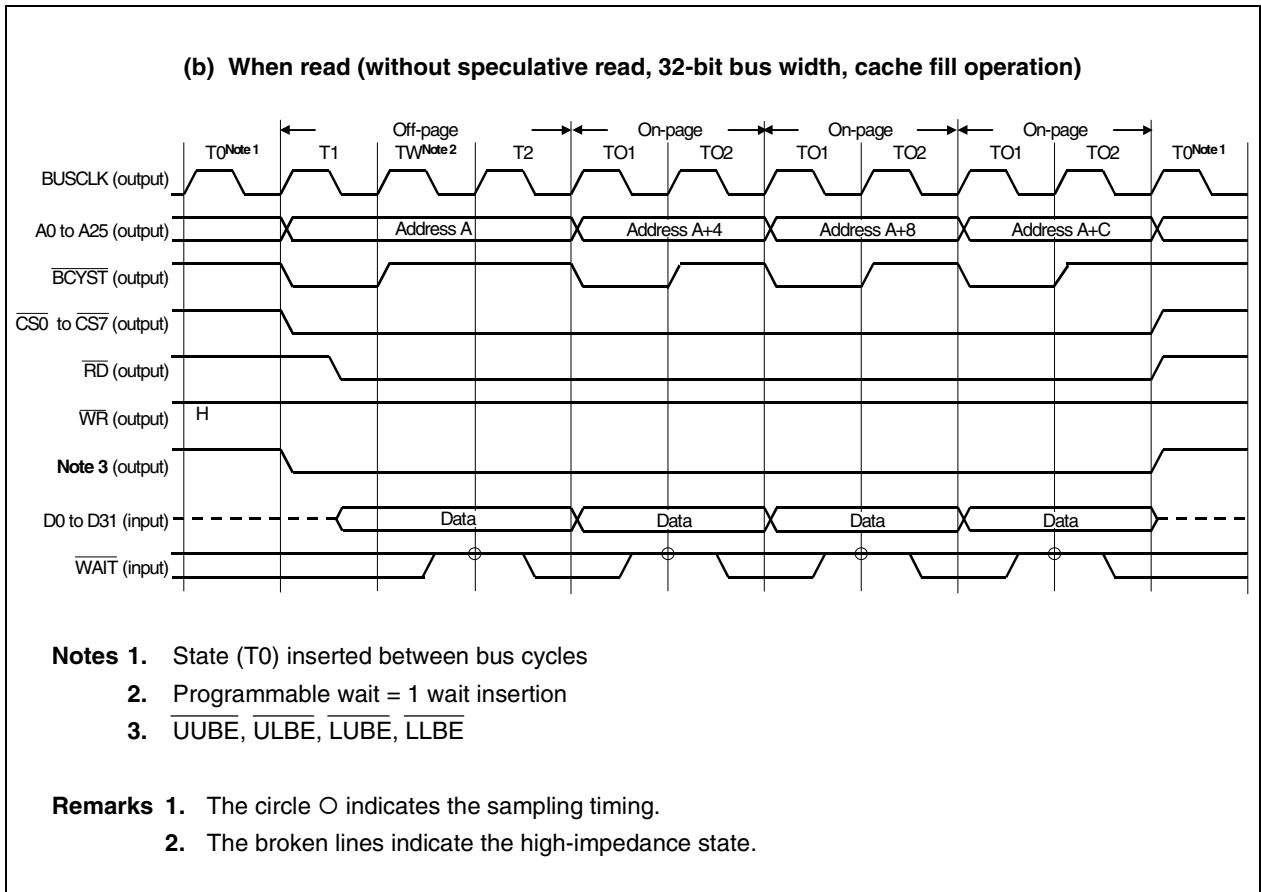


Figure 5-5. Page ROM Access Timing (3/6)

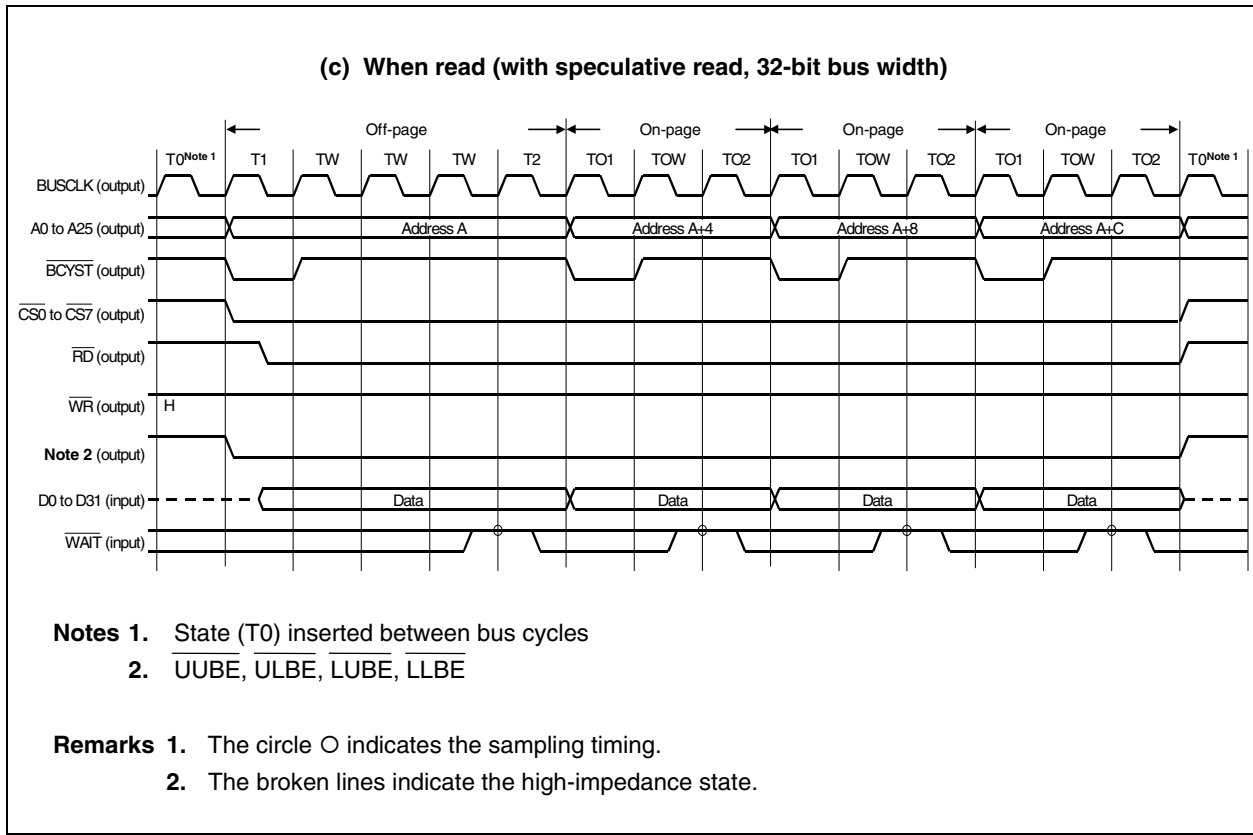


Figure 5-5. Page ROM Access Timing (4/6)

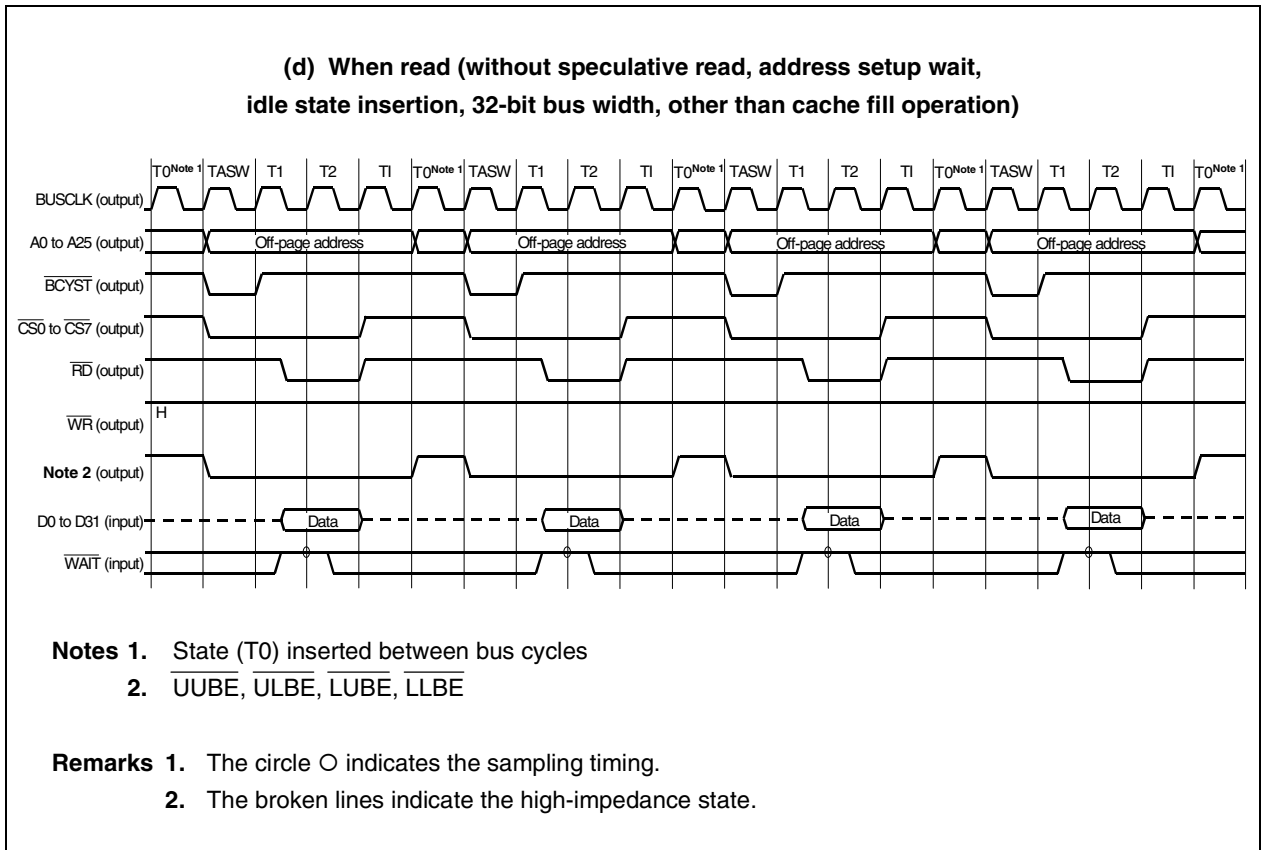


Figure 5-5. Page ROM Access Timing (5/6)

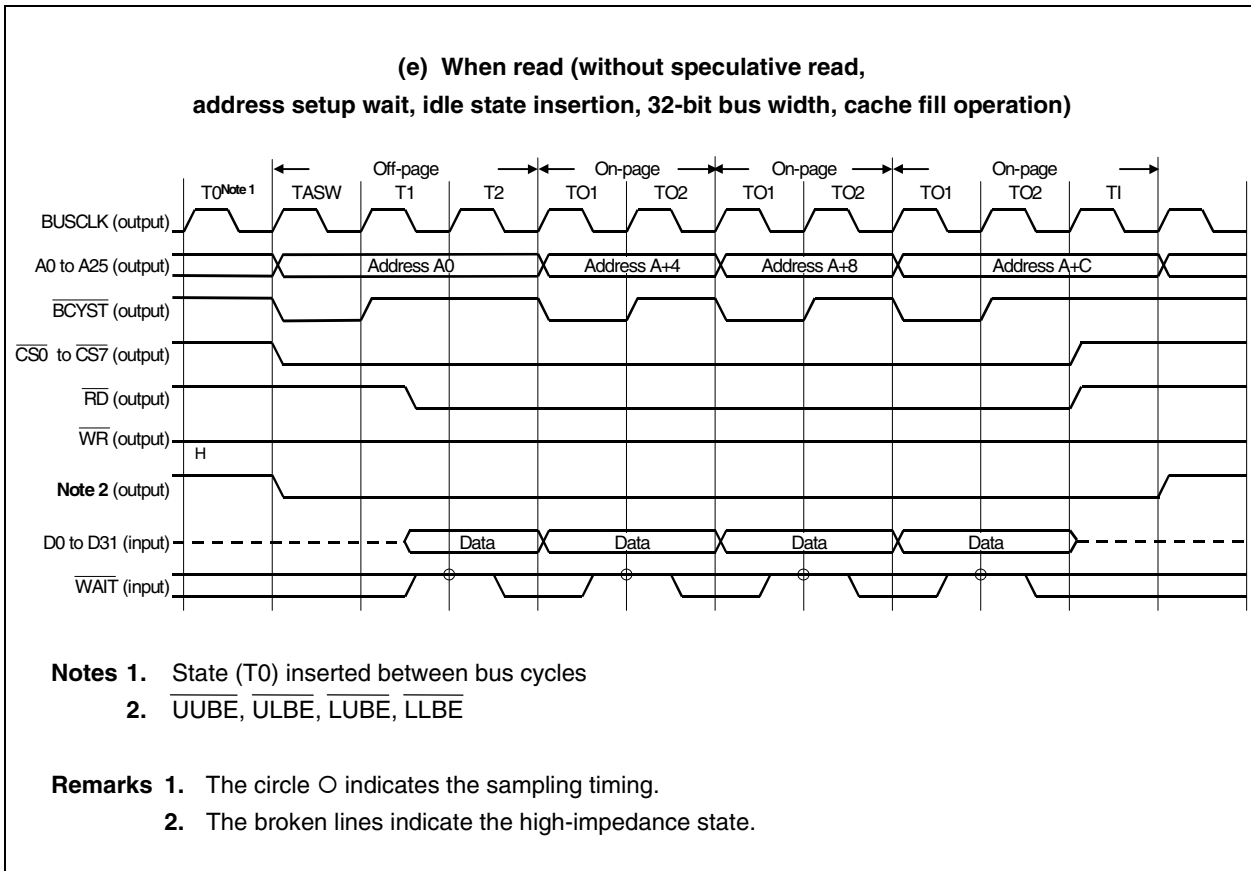
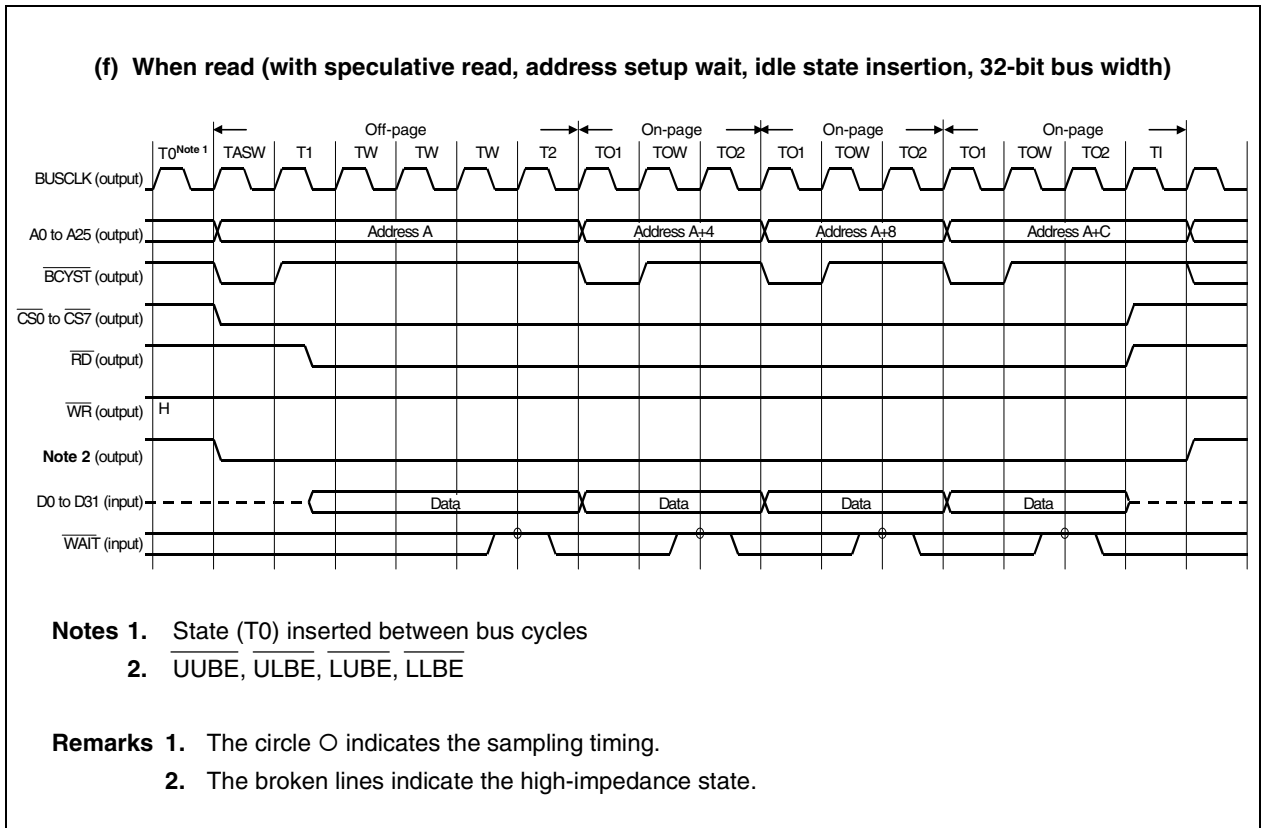


Figure 5-5. Page ROM Access Timing (6/6)



5.3 DRAM Controller (SDRAM)

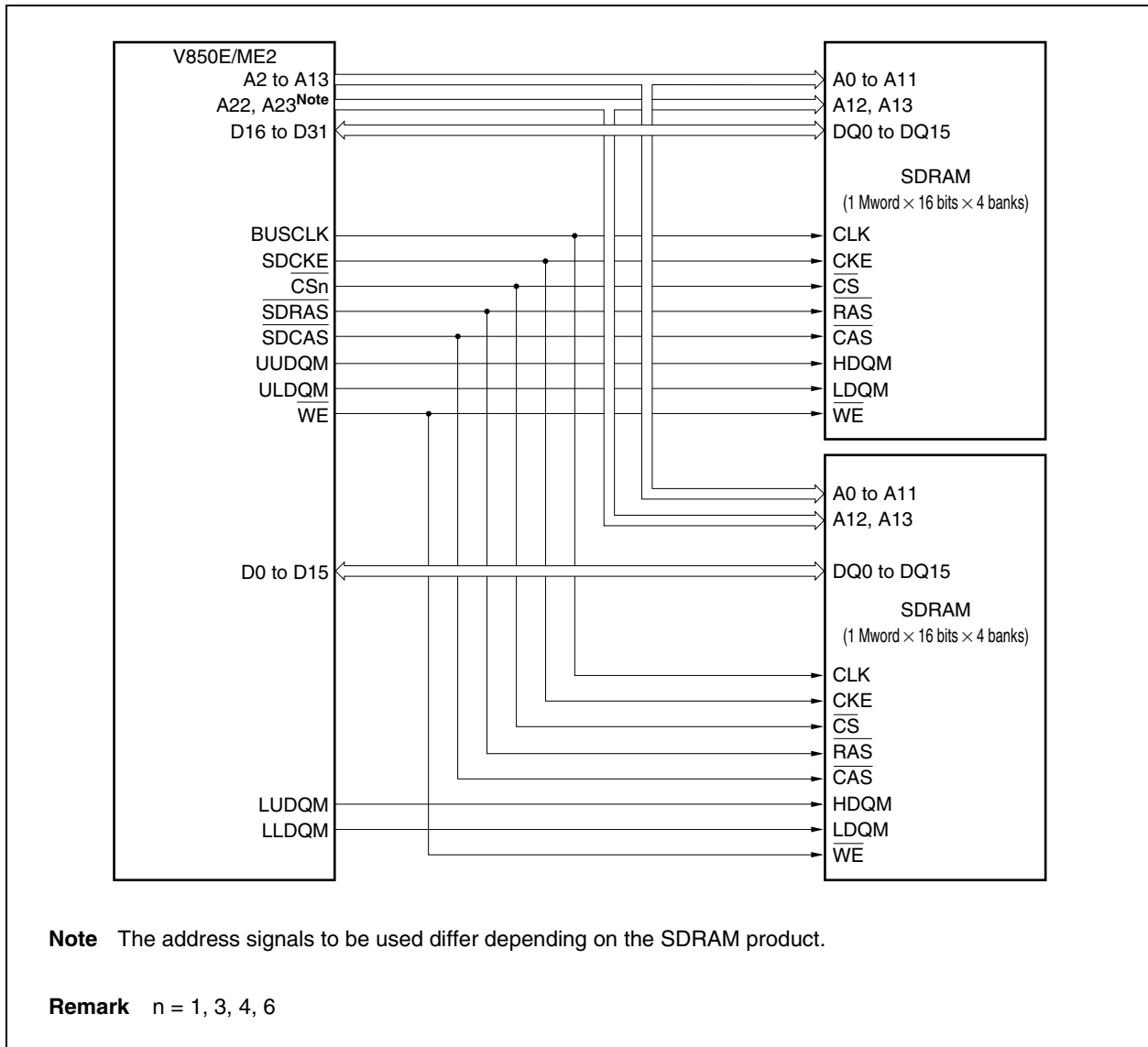
5.3.1 Features

- Burst length: 1
- Wrap type: Sequential
- CAS latency: 1, 2, and 3 supported (only 2 and 3 supported during DMA flyby transfer)
- 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 SAWn0 and SAWn1 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 = 1, 3, 4, 6). Depending on the value of the SSOOn0 and SSOOn1 bits, the column address output in the SDRAM cycle is multiplexed as shown in Figure 5-7 (b) (n = 1, 3, 4, 6). In Figures 5-7 (a) and (b), a0 to a25 indicate the addresses output from the CPU, and A0 to A25 indicate the address pins of the V850E/ME2.

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

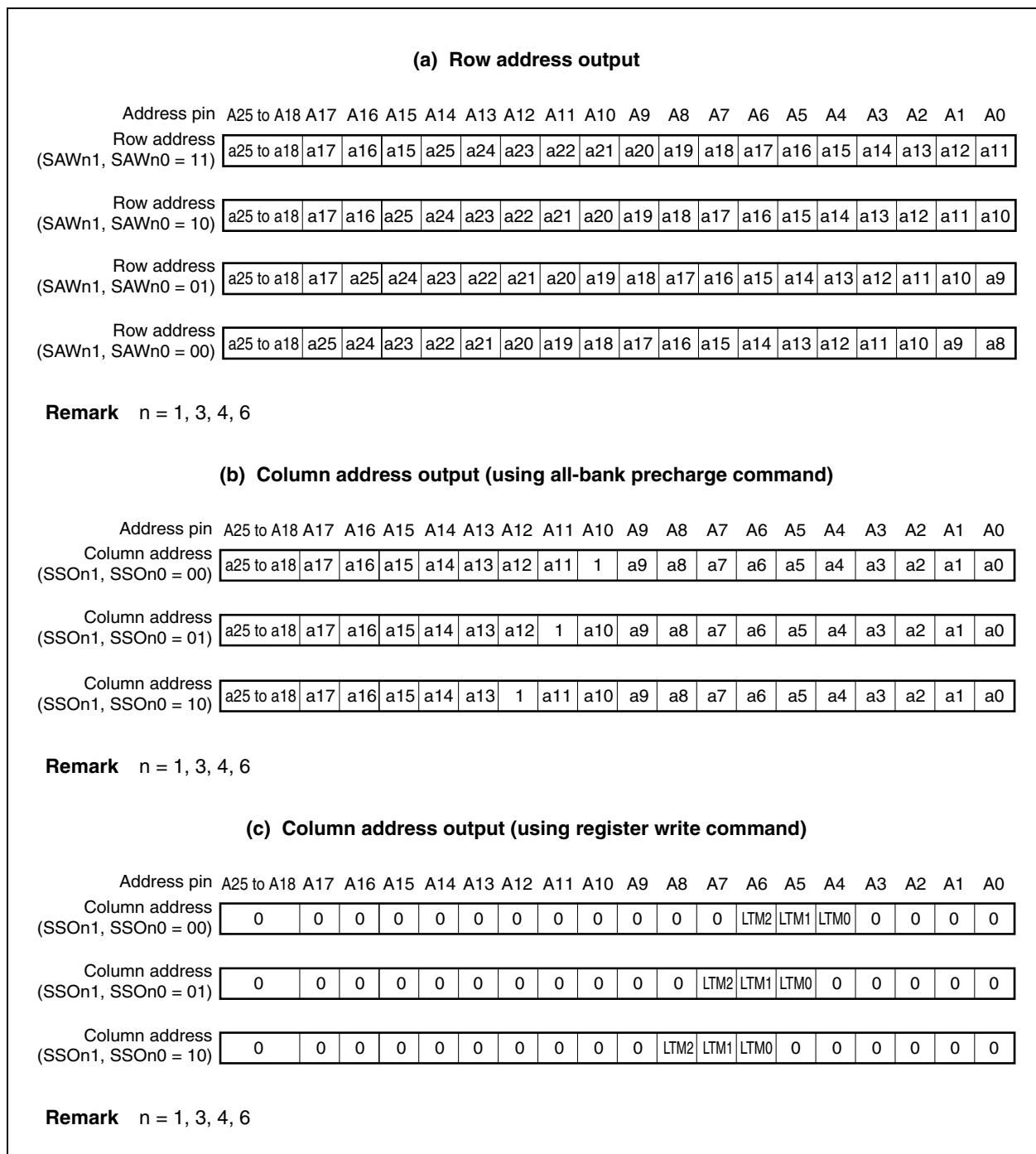


Figure 5-7. Row Address/Column Address Output (2/2)

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

Address pin	A25 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Column address (SSOn1, SSOn0 = 00)	a25 to a18	a17	a16	a15	a14	a12	a11	a10	0	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
Column address (SSOn1, SSOn0 = 01)	a25 to a18	a17	a16	a15	a14	a12	a11	0	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0
Column address (SSOn1, SSOn0 = 10)	a25 to a18	a17	a16	a15	a14	a12	0	a11	a10	a9	a8	a7	a6	a5	a4	a3	a2	a1	a0

Remark n = 1, 3, 4, 6

5.3.4 SDRAM configuration registers 1, 3, 4, 6 (SCR1, SCR3, SCR4, SCR6)

These registers specify the number of waits and the address multiplex width. SCRn corresponds to \overline{CSn} (n = 1, 3, 4, 6). For example, to connect SDRAM to $\overline{CS1}$, set SCR1. These registers can be read or written in 16-bit units.

- Cautions 1. An SDRAM read/write cycle is not generated prior to executing a register write operation. Access SDRAM after reading the value of the SCRn register and confirming that the WCFn bit is set to 1.**
- 2. To write to the SCRn register again following access to SDRAM, clear the MEn bit of the BCT0 and BCT1 registers to 0, and then set it to 1 again before performing access (n = 0 to 6).**
- 3. Do not execute continuous instructions to write to the SCRn register. Be sure to insert another instruction between commands to write to the SCRn register.**
- 4. Start accessing SDRAM after all the SCRn registers have been set.**

(1/3)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
SCR1	0	LTM12	LTM11	LTM10	0	0	0	WCF1	BCW11	BCW10	SSO11	SSO10	RAW11	RAW10	SAW11	SAW10	FFFFFF4A4H	30C0H
SCR3	0	LTM32	LTM31	LTM30	0	0	0	WCF3	BCW31	BCW30	SSO31	SSO30	RAW31	RAW30	SAW31	SAW30	FFFFFF4ACH	30C0H
SCR4	0	LTM42	LTM41	LTM40	0	0	0	WCF4	BCW41	BCW40	SSO41	SSO40	RAW41	RAW40	SAW41	SAW40	FFFFFF4B0H	30C0H
SCR6	0	LTM62	LTM61	LTM60	0	0	0	WCF6	BCW61	BCW60	SSO61	SSO60	RAW61	RAW60	SAW61	SAW60	FFFFFF4B8H	30C0H

Bit position	Bit name	Function																				
14 to 12	LTMn2 to LTMn0	Sets the CAS latency value for reading. <table border="1"> <thead> <tr> <th>LTMn2</th> <th>LTMn1</th> <th>LTMn0</th> <th>Latency</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1 (setting prohibited during DMA flyby transfer)</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>3</td> </tr> <tr> <td colspan="3">Other than above</td> <td>Setting prohibited</td> </tr> </tbody> </table>	LTMn2	LTMn1	LTMn0	Latency	0	0	1	1 (setting prohibited during DMA flyby transfer)	0	1	0	2	0	1	1	3	Other than above			Setting prohibited
LTMn2	LTMn1	LTMn0	Latency																			
0	0	1	1 (setting prohibited during DMA flyby transfer)																			
0	1	0	2																			
0	1	1	3																			
Other than above			Setting prohibited																			
8	WCFn	Indicates that execution of a register write command on SDRAM has been completed after the SCRn register was set. This bit is set to 1 when a register write command is generated. This bit can only be read. 0: Setting not completed 1: Setting completed																				

Remark n = 1, 3, 4, 6

Bit position	Bit name	Function															
7, 6	BCWn1, BCWn0	<p>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.</p> <table border="1"> <thead> <tr> <th>BCWn1</th> <th>BCWn0</th> <th>Number of wait states inserted</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>2</td> </tr> <tr> <td>1</td> <td>1</td> <td>3</td> </tr> </tbody> </table>	BCWn1	BCWn0	Number of wait states inserted	0	0	Setting prohibited	0	1	1	1	0	2	1	1	3
BCWn1	BCWn0	Number of wait states inserted															
0	0	Setting prohibited															
0	1	1															
1	0	2															
1	1	3															
5, 4	SSOn1, SSOn0	<p>Specifies the address shift width during on-page judgment. If the external data bus width is set to 16 or 32 bits, the system does not use the lower address (A0 or A1, A0). Set these bits in accordance with the contents of the LBS register corresponding to \overline{CSn}.</p> <table border="1"> <thead> <tr> <th>SSOn1</th> <th>SSOn0</th> <th>Address shift width</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0 bits (external data bus width: 8 bits)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1 bit (external data bus width: 16 bits)^{Note}</td> </tr> <tr> <td>1</td> <td>0</td> <td>2 bits (external data bus width: 32 bits)^{Note}</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	SSOn1	SSOn0	Address shift width	0	0	0 bits (external data bus width: 8 bits)	0	1	1 bit (external data bus width: 16 bits) ^{Note}	1	0	2 bits (external data bus width: 32 bits) ^{Note}	1	1	Setting prohibited
SSOn1	SSOn0	Address shift width															
0	0	0 bits (external data bus width: 8 bits)															
0	1	1 bit (external data bus width: 16 bits) ^{Note}															
1	0	2 bits (external data bus width: 32 bits) ^{Note}															
1	1	Setting prohibited															
3, 2	RAWn1, RAWn0	<p>Specifies the row address width.</p> <table border="1"> <thead> <tr> <th>RAWn1</th> <th>RAWn0</th> <th>Row address width</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>11 bits</td> </tr> <tr> <td>0</td> <td>1</td> <td>12 bits^{Note}</td> </tr> <tr> <td>1</td> <td>0</td> <td>13 bits^{Note}</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	RAWn1	RAWn0	Row address width	0	0	11 bits	0	1	12 bits ^{Note}	1	0	13 bits ^{Note}	1	1	Setting prohibited
RAWn1	RAWn0	Row address width															
0	0	11 bits															
0	1	12 bits ^{Note}															
1	0	13 bits ^{Note}															
1	1	Setting prohibited															
1, 0	SAWn1, SAWn0	<p>Specifies the address multiplex width (column address width) during SDRAM access.</p> <table border="1"> <thead> <tr> <th>SAWn1</th> <th>SAWn0</th> <th>Address multiplex width (column address width)</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>8 bits</td> </tr> <tr> <td>0</td> <td>1</td> <td>9 bits</td> </tr> <tr> <td>1</td> <td>0</td> <td>10 bits^{Note}</td> </tr> <tr> <td>1</td> <td>1</td> <td>11 bits^{Note}</td> </tr> </tbody> </table>	SAWn1	SAWn0	Address multiplex width (column address width)	0	0	8 bits	0	1	9 bits	1	0	10 bits ^{Note}	1	1	11 bits ^{Note}
SAWn1	SAWn0	Address multiplex width (column address width)															
0	0	8 bits															
0	1	9 bits															
1	0	10 bits ^{Note}															
1	1	11 bits ^{Note}															

- Remarks 1.** The **Note** is described on the next page.
2. n = 1, 3, 4, 6

Note The following setting is prohibited because the upper limit of the address is exceeded.

SSOn1	SSOn0	RAWn1	RAWn0	SAWn1	SAWn0	Setting
0	1	1	0	1	1	Data bus width: 16 bits Row address width: 13 bits Column address width: 11 bits
1	0	0	1	1	1	Data bus width: 32 bits Row address width: 12 bits Column address width: 11 bits
1	0	1	0	1	0	Data bus width: 32 bits Row address width: 13 bits Column address width: 10 bits
					1	Data bus width: 32 bits Row address width: 13 bits Column address width: 11 bits

Remark n = 1, 3, 4, 6

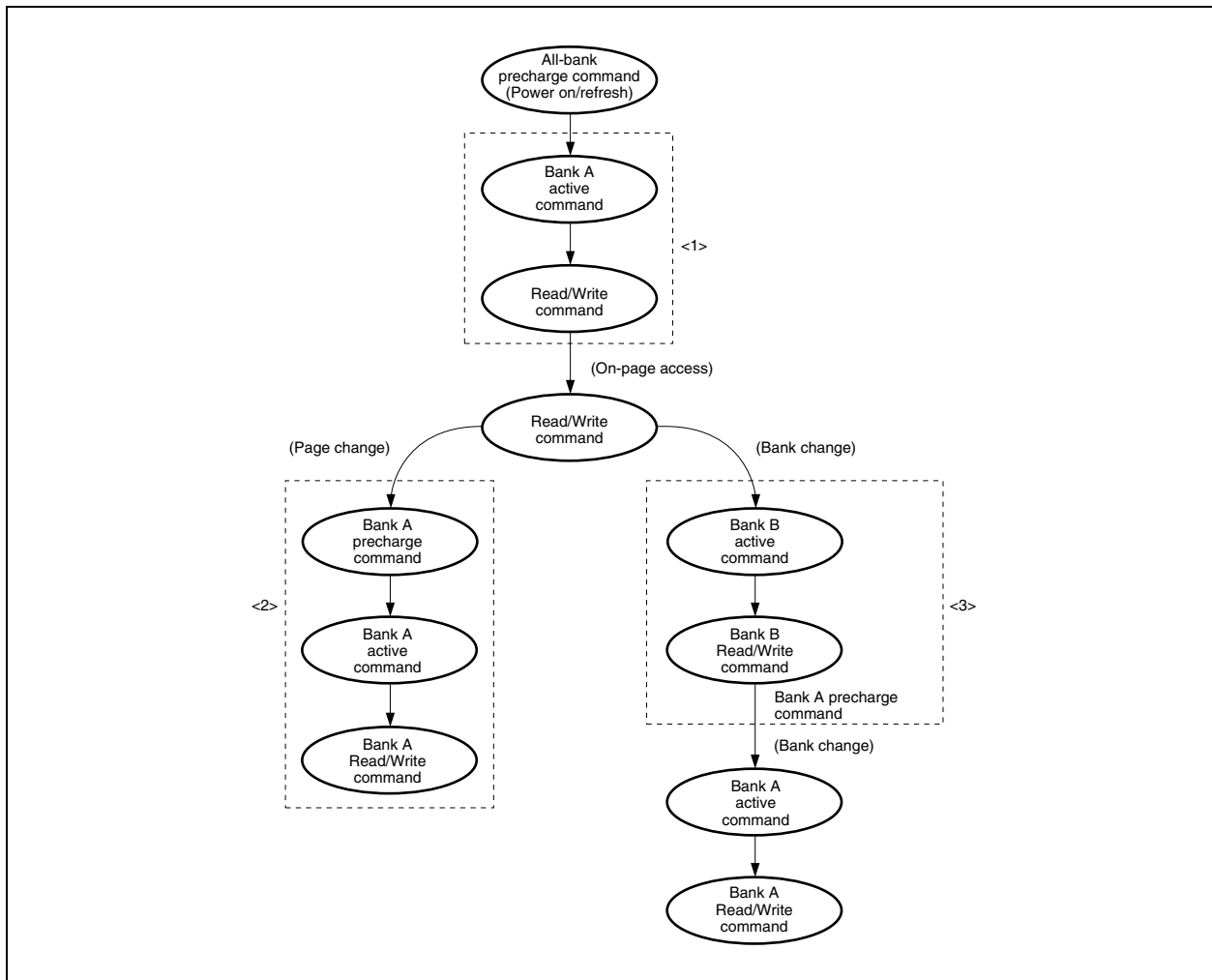
5.3.5 SDRAM access

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 to 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.

A one-state T0 cycle is always inserted immediately before all read commands activated by the CPU.

The number of idle states (TI) set by the bus cycle control register (BCC) are inserted after the read cycle (no idle states are inserted, however, if BCn1 and BCn0 are 00) (n = 1, 3, 4, 6). 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 (1/5)

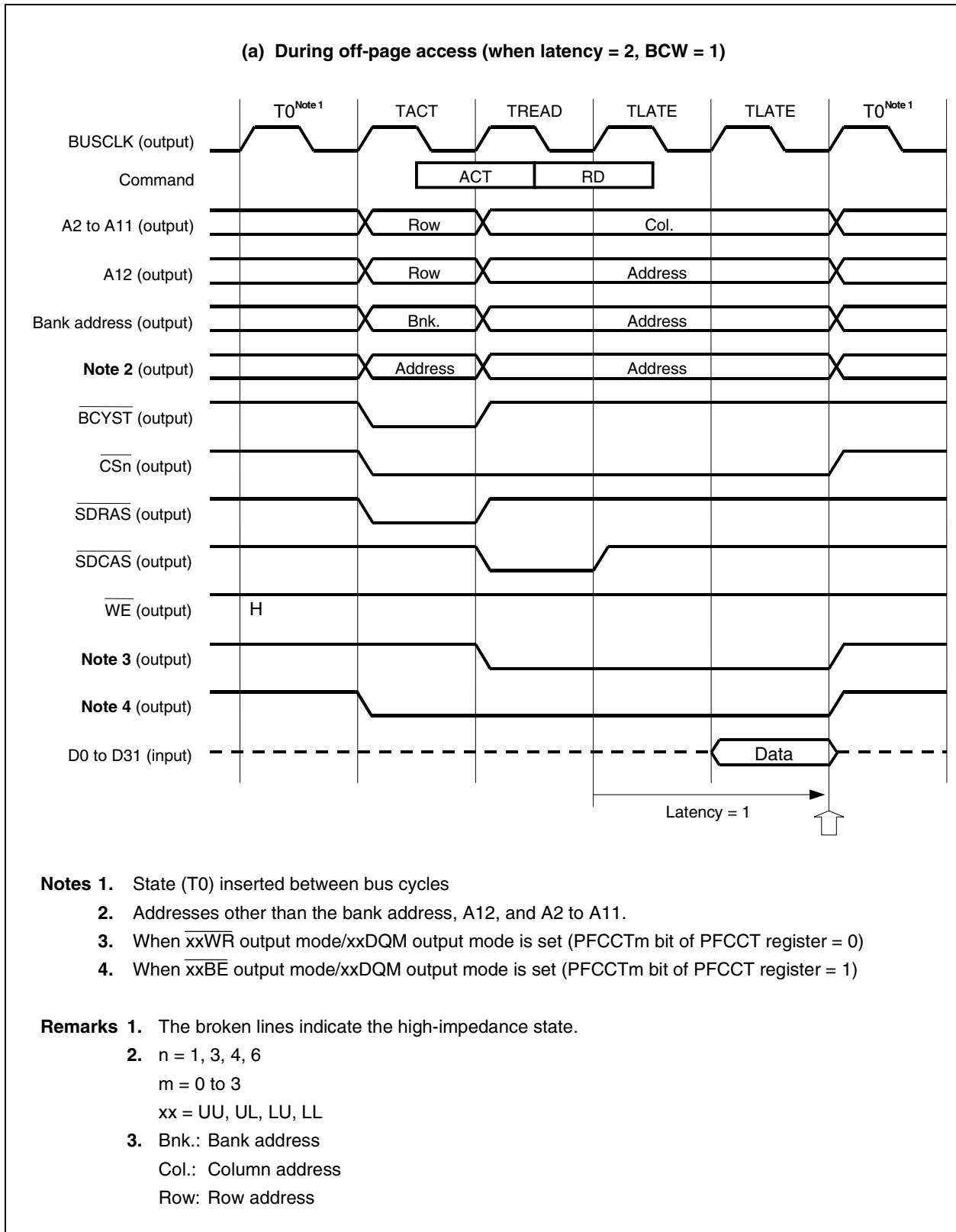


Figure 5-9. SDRAM Single Read Cycle (2/5)

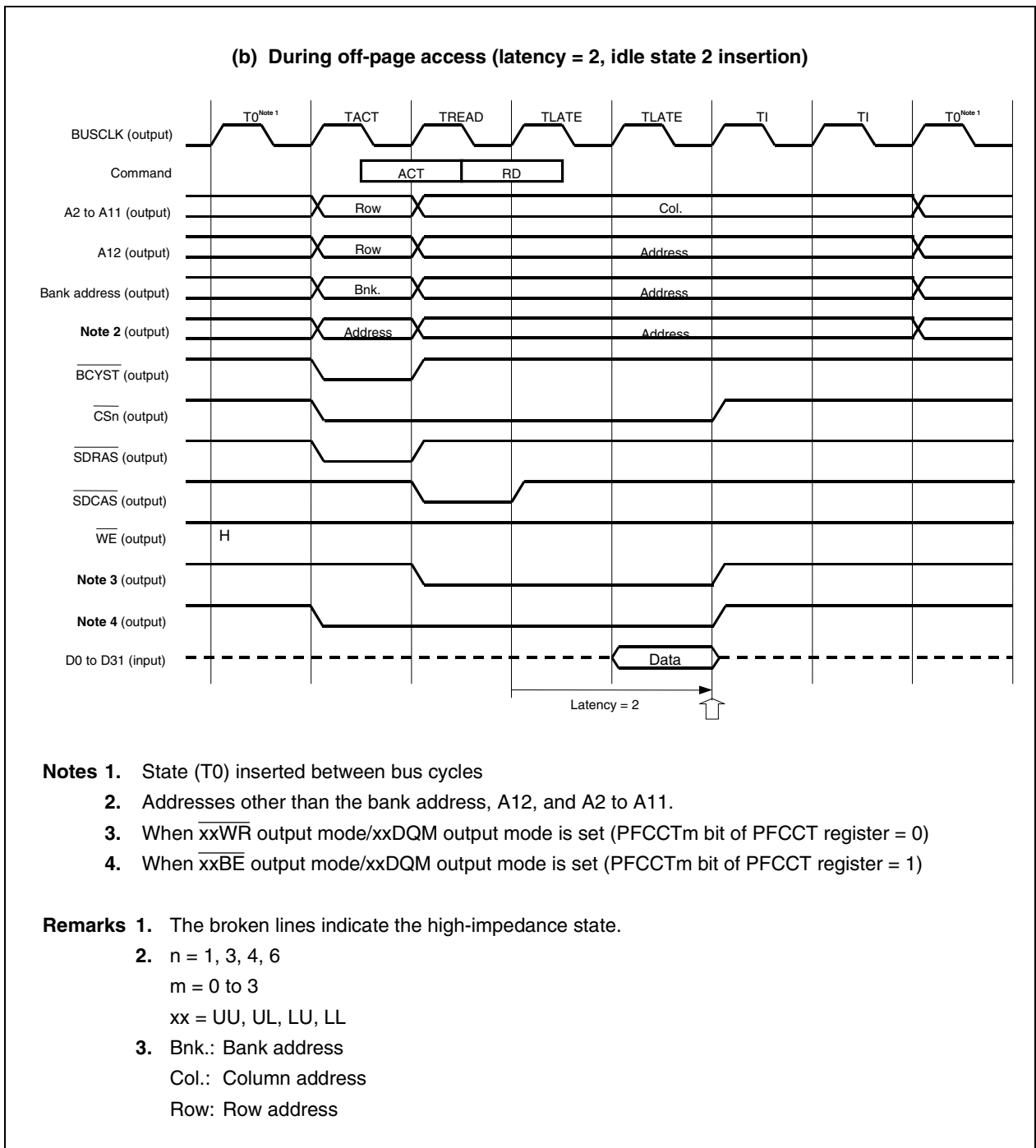
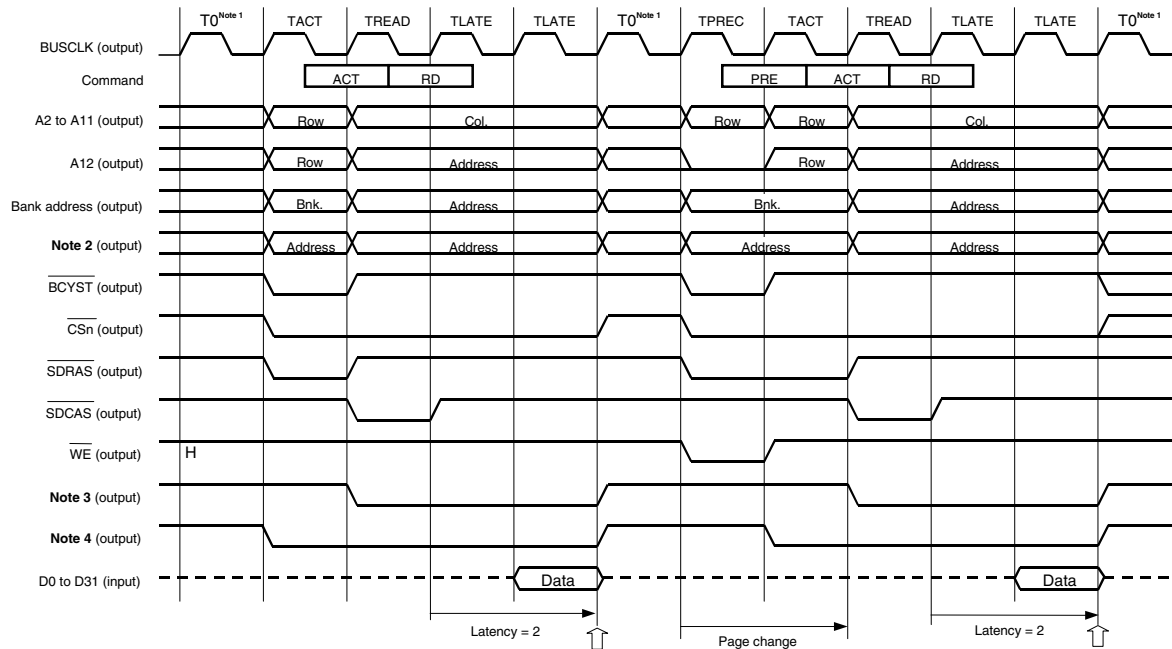


Figure 5-9. SDRAM Single Read Cycle (3/5)

(c) During off-page access (when latency = 2, page change, BCW = 1)



- Notes**
1. State (T0) inserted between bus cycles
 2. Addresses other than the bank address, A12, and A2 to A11.
 3. When \overline{xxWR} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 0)
 4. When \overline{xxBE} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 1)

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. n = 1, 3, 4, 6
m = 0 to 3
xx = UU, UL, LU, LL
 3. Bnk.: Bank address
Col.: Column address
Row: Row address

Figure 5-9. SDRAM Single Read Cycle (4/5)

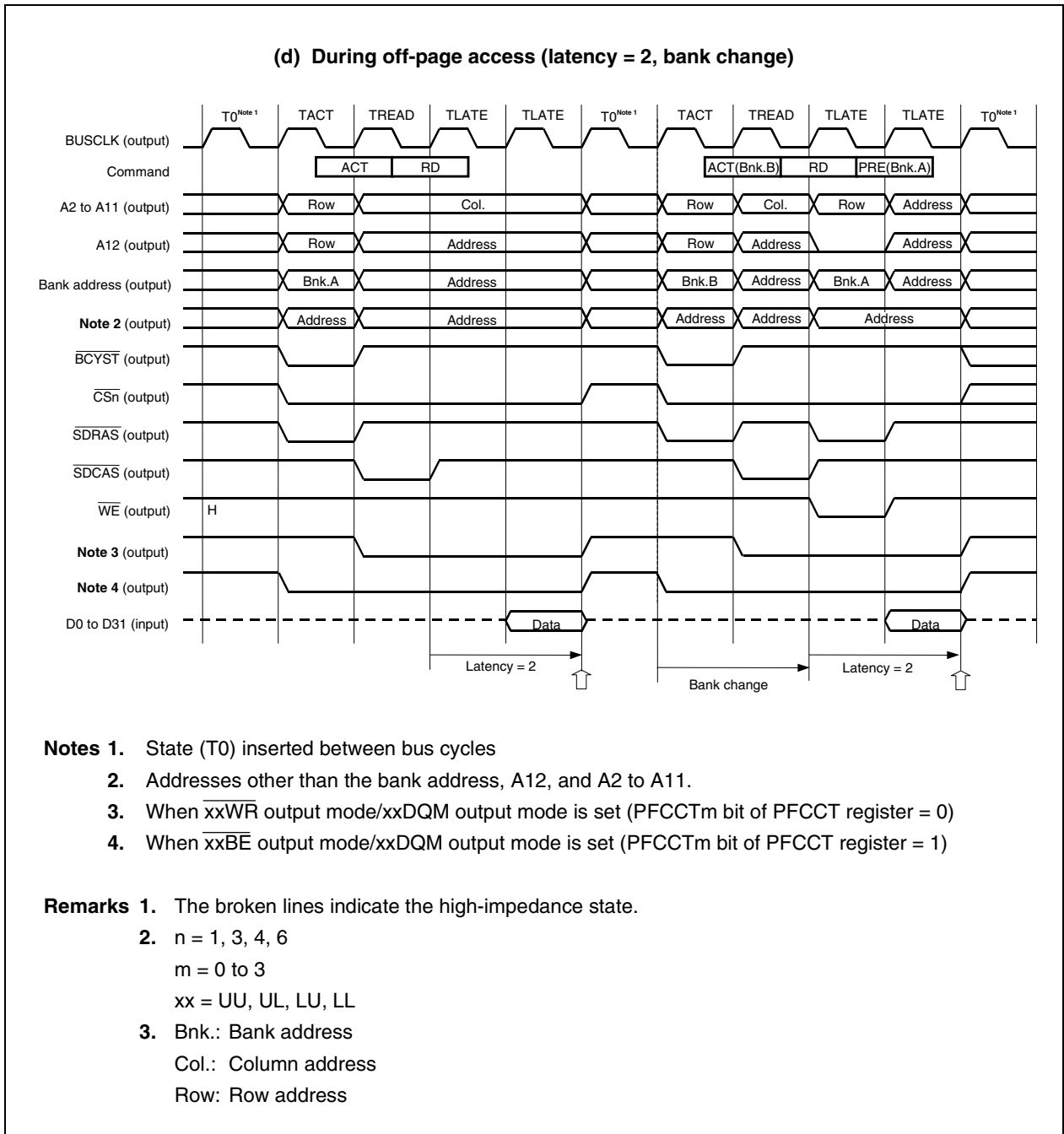
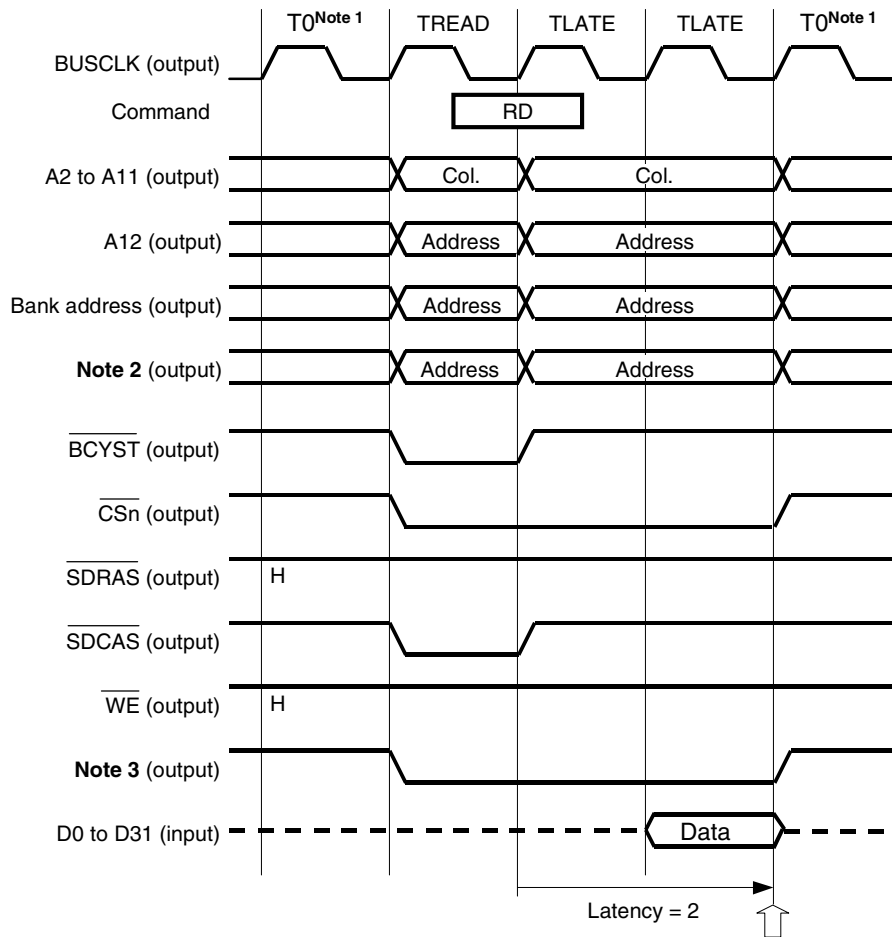


Figure 5-9. SDRAM Single Read Cycle (5/5)

(e) During on-page access (when latency = 2, 32-bit bus width)



- Notes**
1. State (T0) inserted between bus cycles
 2. Addresses other than the bank address, A12, and A2 to A11.
 3. UUDQM, ULQDM, LUDQM, LLDQM

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. n = 1, 3, 4, 6
 3. Col.: Column address

(2) SDRAM single write cycle

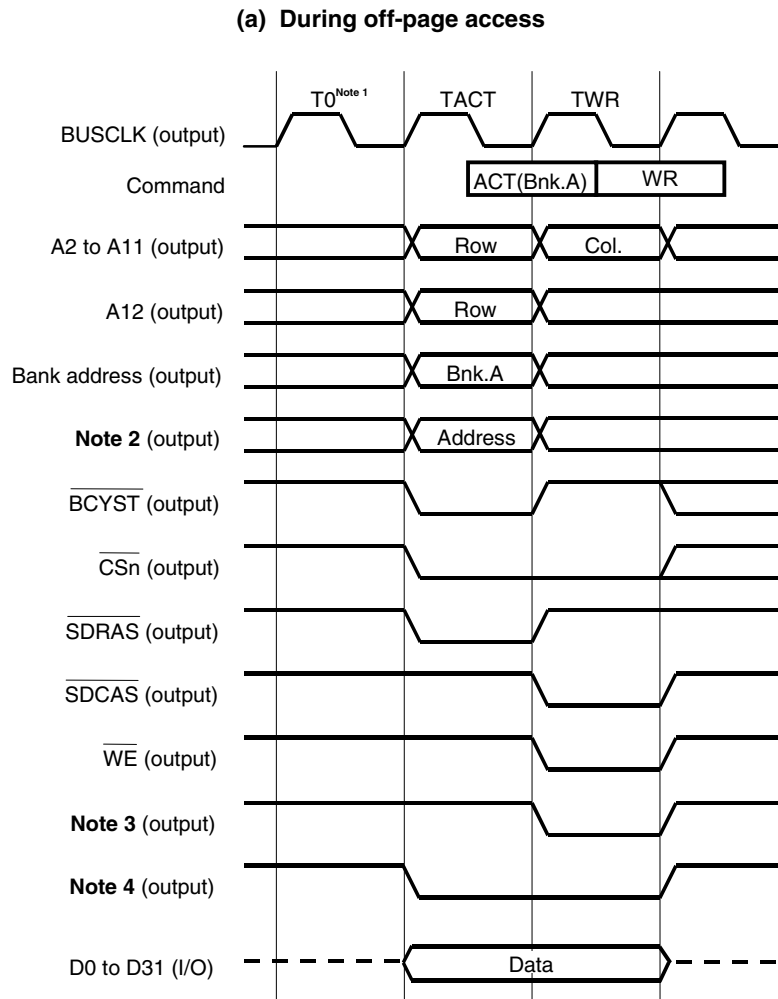
The SDRAM single write cycle is a cycle for writing to SDRAM by executing a write instruction (ST) for the SDRAM area or by 2-cycle DMA transfer.

In the SDRAM single write cycle, the active command (ACT) and write command (WR) are issued to 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 all write commands activated by the CPU.

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

Figure 5-10. SDRAM Single Write Cycle (1/6)



Notes 1. State (T0) inserted between bus cycles

2. Addresses other than the bank address, A12, and A2 to A11.

3. When \overline{xxWR} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 0)

4. When \overline{xxBE} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 1)

Remarks 1. The broken lines indicate the high-impedance state.

2. n = 1, 3, 4, 6

m = 0 to 3

xx = UU, UL, LU, LL

3. Bnk.: Bank address

Col.: Column address

Row: Row address

Figure 5-10. SDRAM Single Write Cycle (2/6)

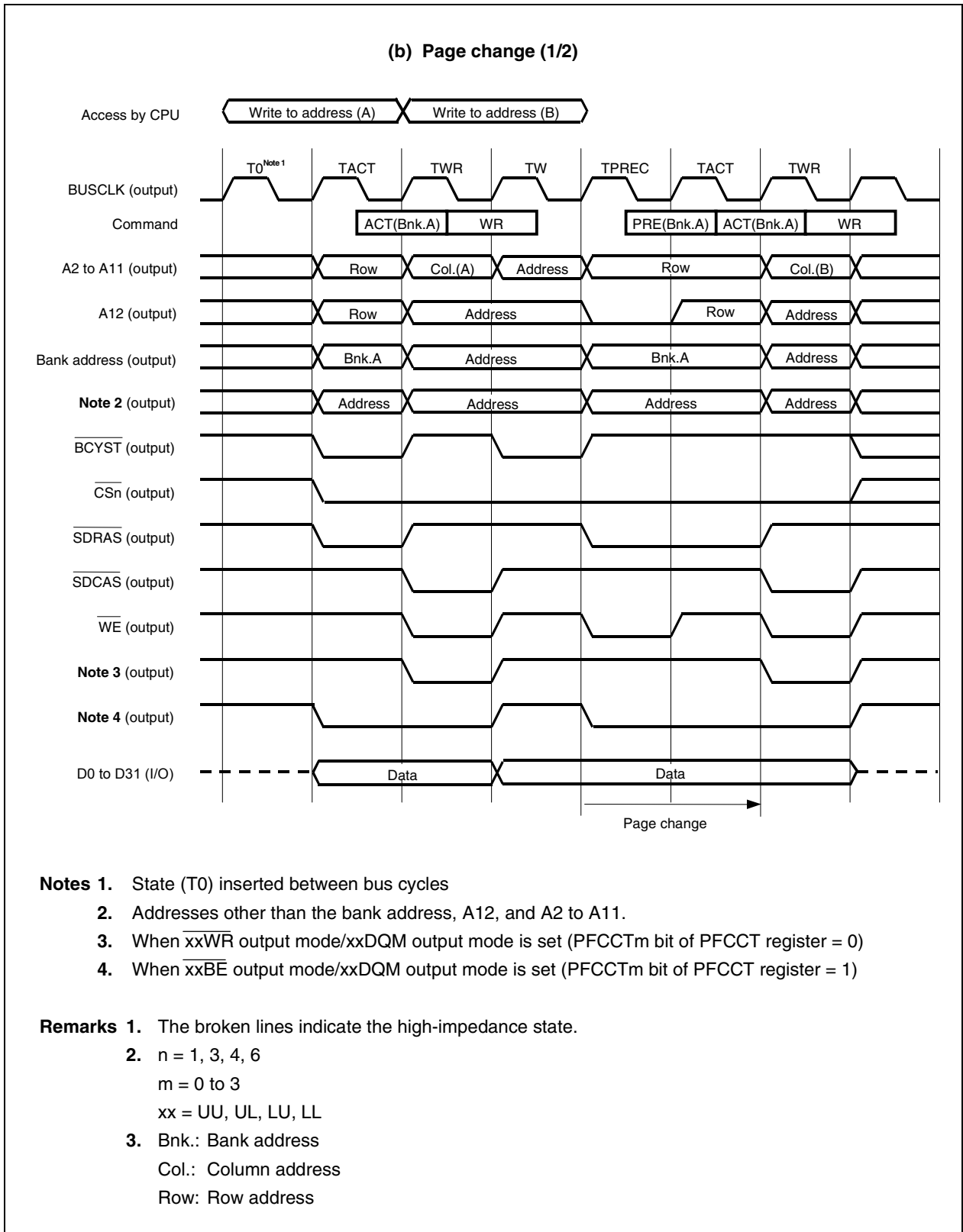


Figure 5-10. SDRAM Single Write Cycle (3/6)

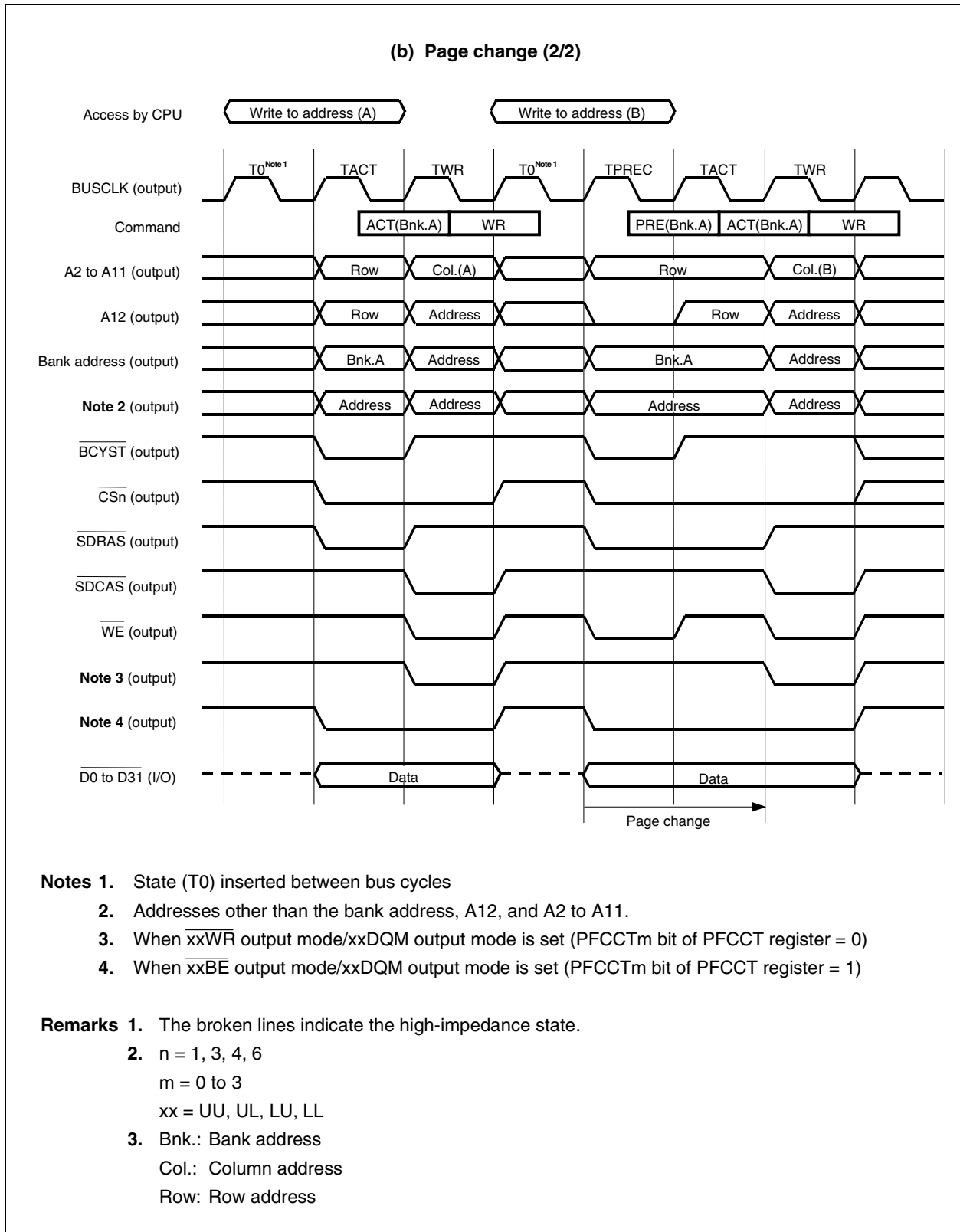


Figure 5-10. SDRAM Single Write Cycle (4/6)

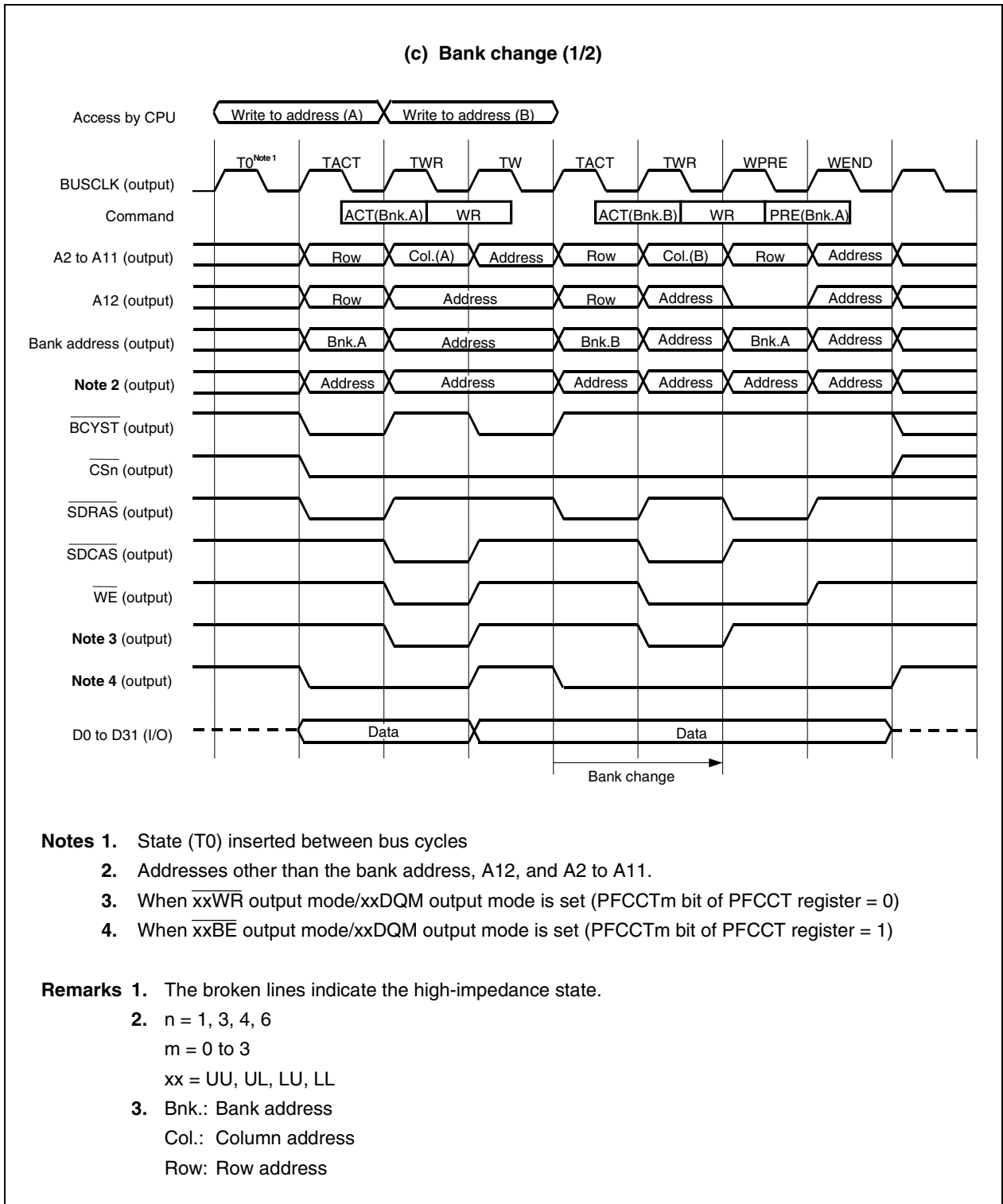
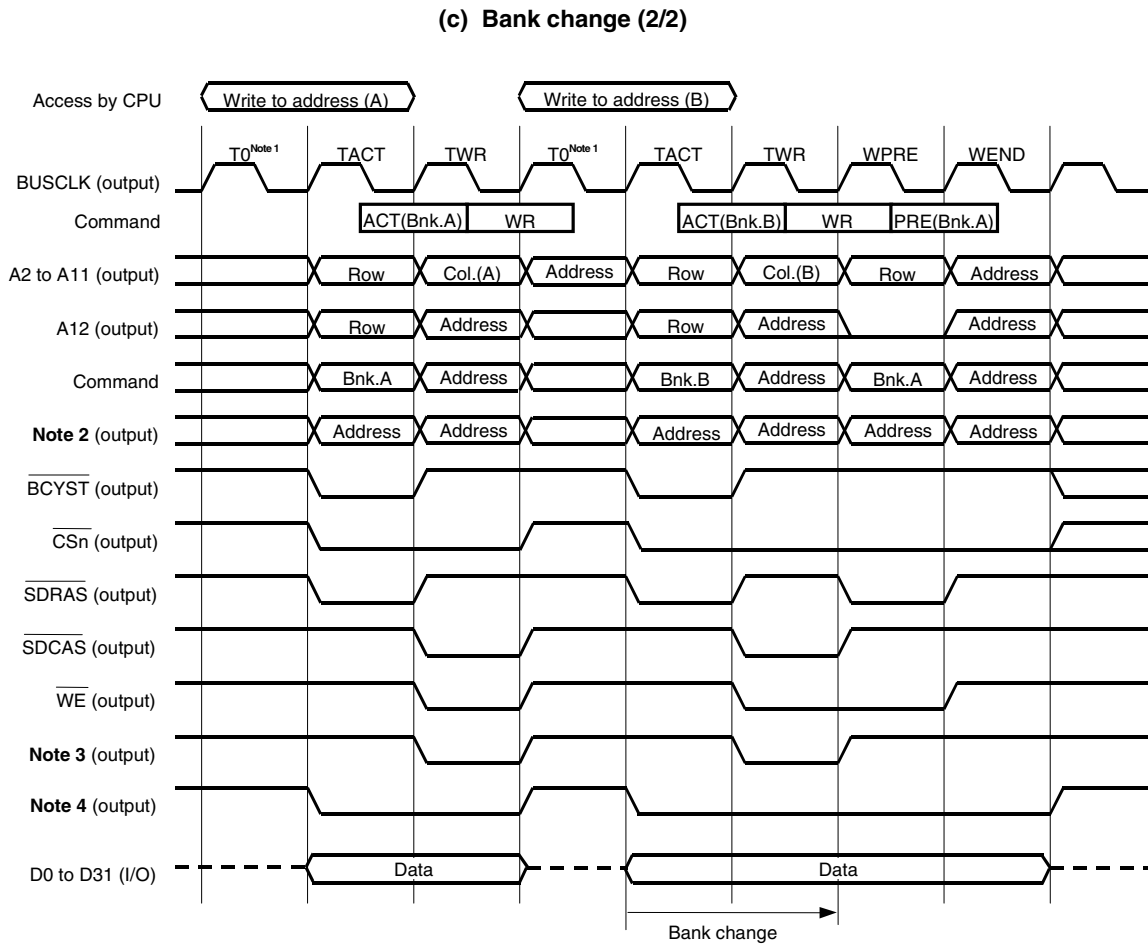


Figure 5-10. SDRAM Single Write Cycle (5/6)



Notes 1. State (T0) inserted between bus cycles

2. Addresses other than the bank address, A12, and A2 to A11.

3. When \overline{xxWR} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 0)

4. When \overline{xxBE} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 1)

Remarks 1. The broken lines indicate the high-impedance state.

2. n = 1, 3, 4, 6

m = 0 to 3

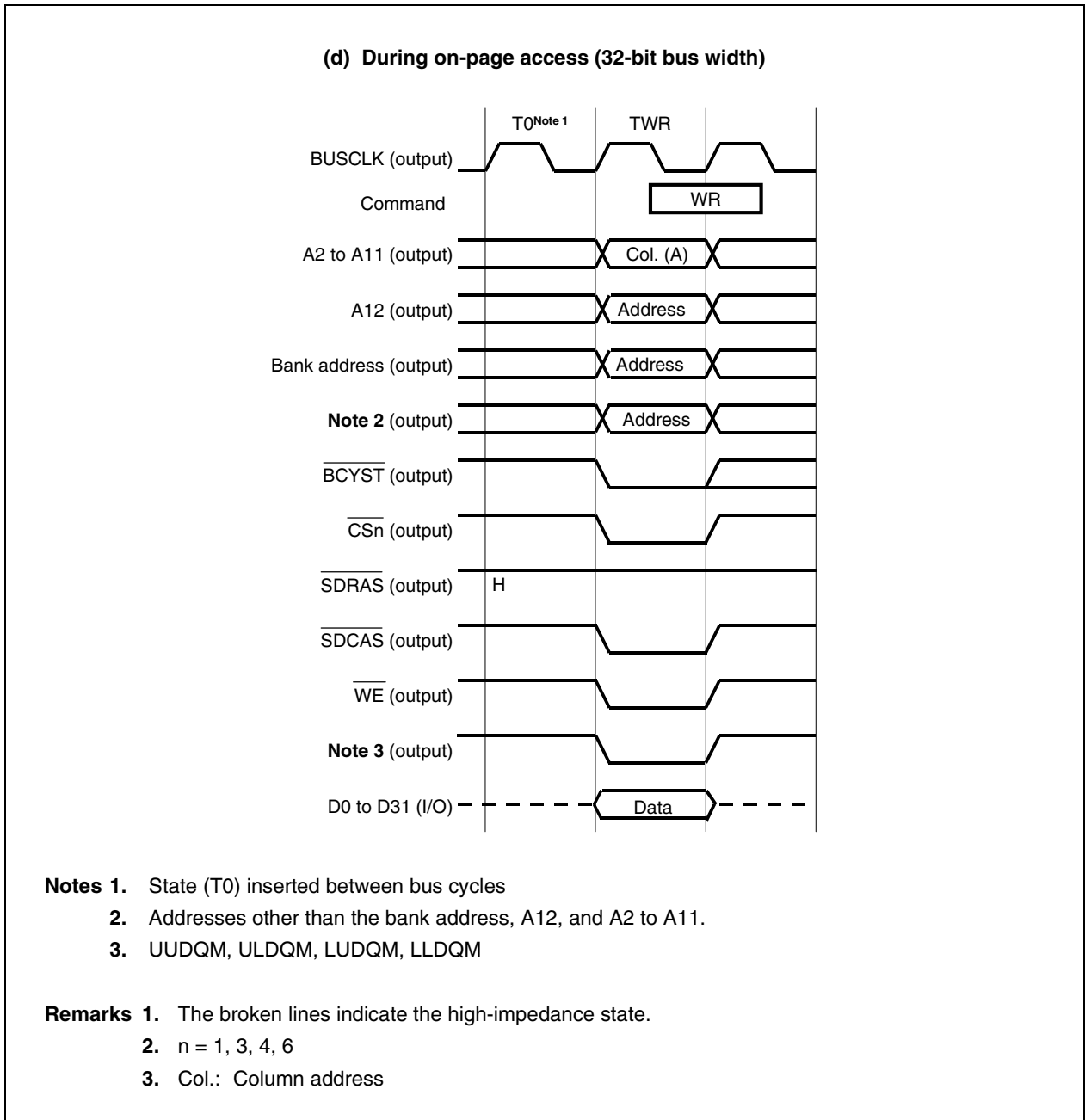
xx = UU, UL, LU, LL

3. Bnk.: Bank address

Col.: Column address

Row: Row address

Figure 5-10. SDRAM Single Write Cycle (6/6)



(3) SDRAM access timing control

The SDRAM access timing can be controlled by SDRAM configuration register n (SCRn) (n = 1, 3, 4, 6). For details, see 5.3.4 SDRAM configuration registers 1, 3, 4, 6 (SCR1, SCR3, SCR4, SCR6).

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 BCWn1 and BCWn0 bits of the SCRn register.

BCWn1, BCWn0 = 01B: 1 wait

BCWn1, BCWn0 = 10B: 2 waits

BCWn1, BCWn0 = 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 BCWn1 and BCWn0 bits of the SCRn register.

BCWn1, BCWn0 = 01B: 1 wait

BCWn1, BCWn0 = 10B: 2 waits

BCWn1, BCWn0 = 11B: 3 waits

(c) CAS latency setting when read

The CAS latency during a read operation can be set by setting the LTMn2 to LTMn0 bits of the SCRn register.

LTMn2 to LTMn0 = 001B: Latency = 1 (setting prohibited during DMA flyby transfer)

LTMn2 to LTMn0 = 010B: Latency = 2

LTMn2 to LTMn0 = 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 BCWn1 and BCWn0 bits of the SCRn register. The number of wait states becomes four times the value set by BCWn1 and BCWn0 bits.

BCWn1, BCWn0 = 01B: 4 waits

BCWn1, BCWn0 = 10B: 8 waits

BCWn1, BCWn0 = 11B: 12 waits

Figure 5-11. SDRAM Access Timing (1/9)

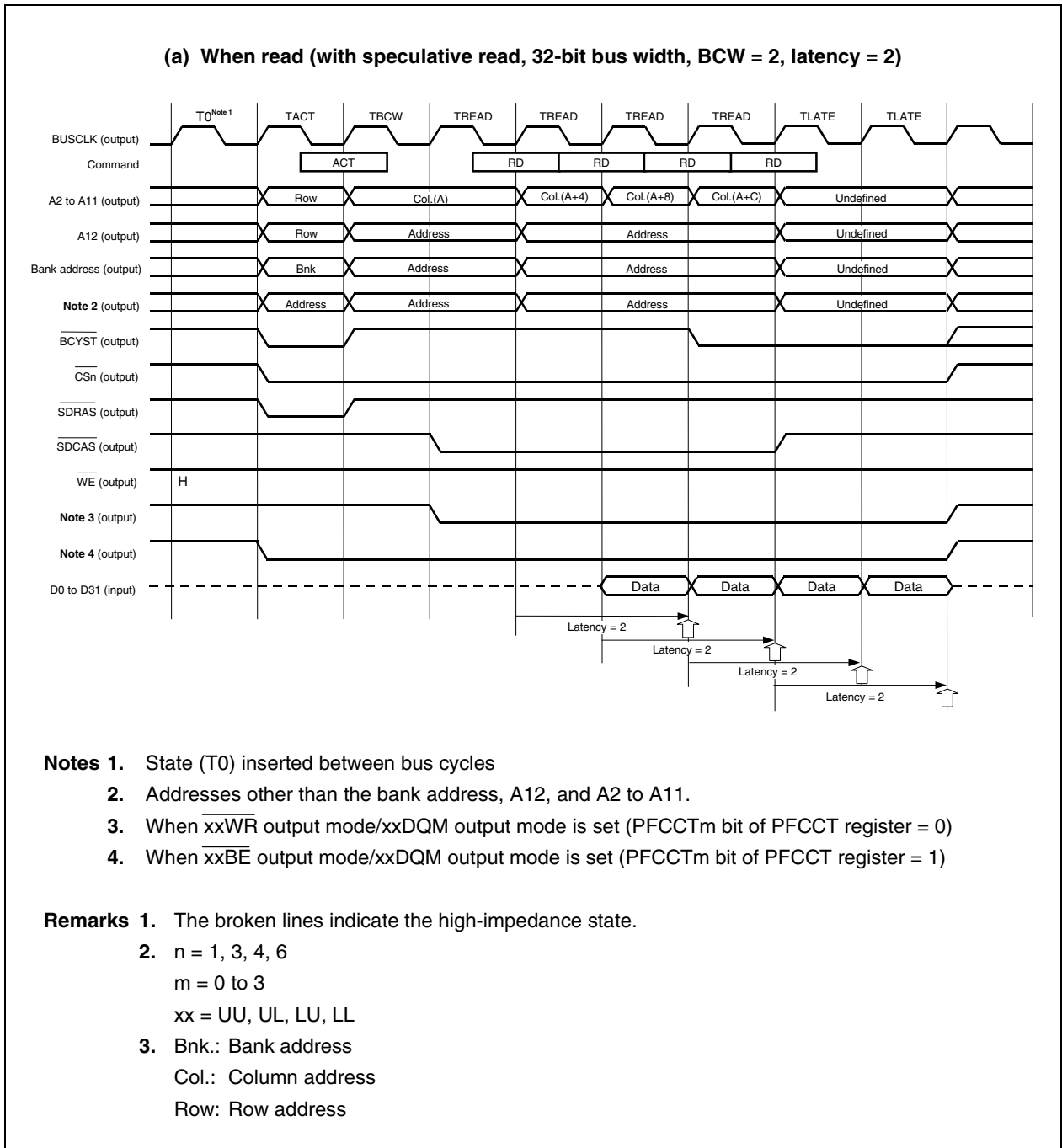
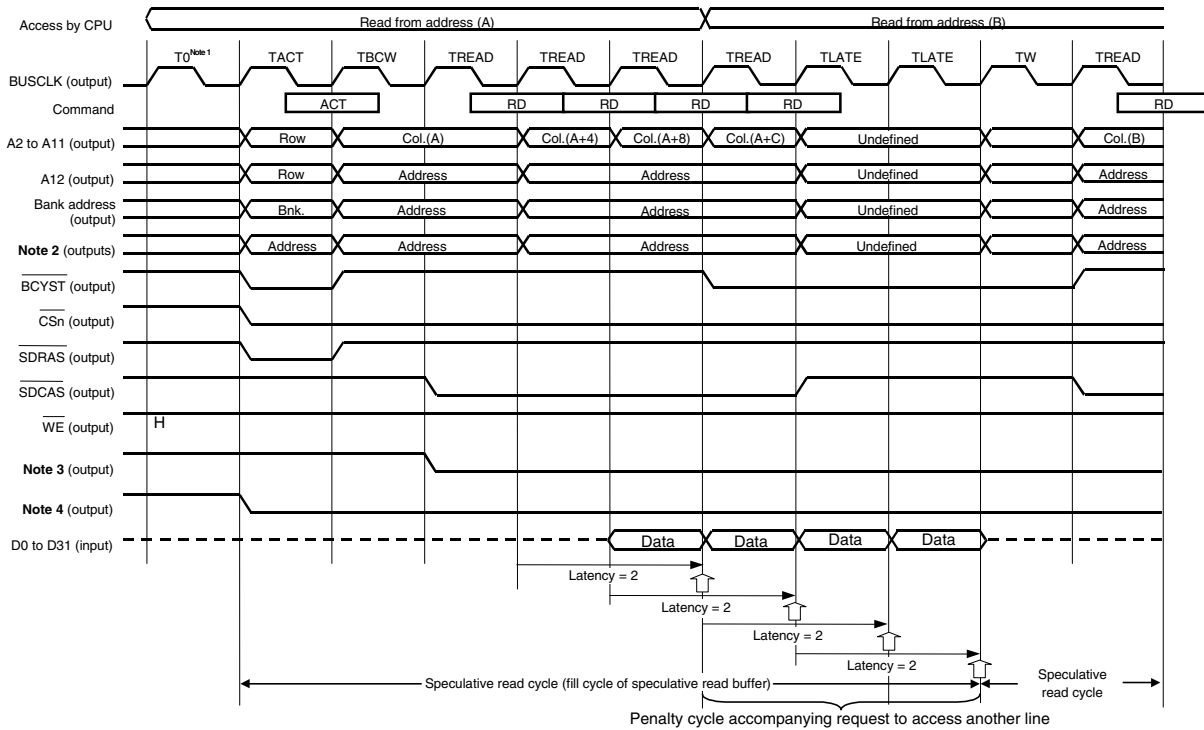


Figure 5-11. SDRAM Access Timing (2/9)

(b) When read (with speculative read, access on another line, 32-bit bus width, BCW = 2, latency = 2)



- Notes**
1. State (T0) inserted between bus cycles
 2. Addresses other than the bank address, A12, and A2 to A11.
 3. When \overline{xxWR} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 0)
 4. When \overline{xxBE} output mode/ \overline{xxDQM} output mode is set (PFCCTm bit of PFCCT register = 1)

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. $n = 1, 3, 4, 6$
 $m = 0$ to 3
 $xx = UU, UL, LU, LL$
 3. Bnk.: Bank address
 Col.: Column address
 Row: Row address

Figure 5-11. SDRAM Access Timing (3/9)

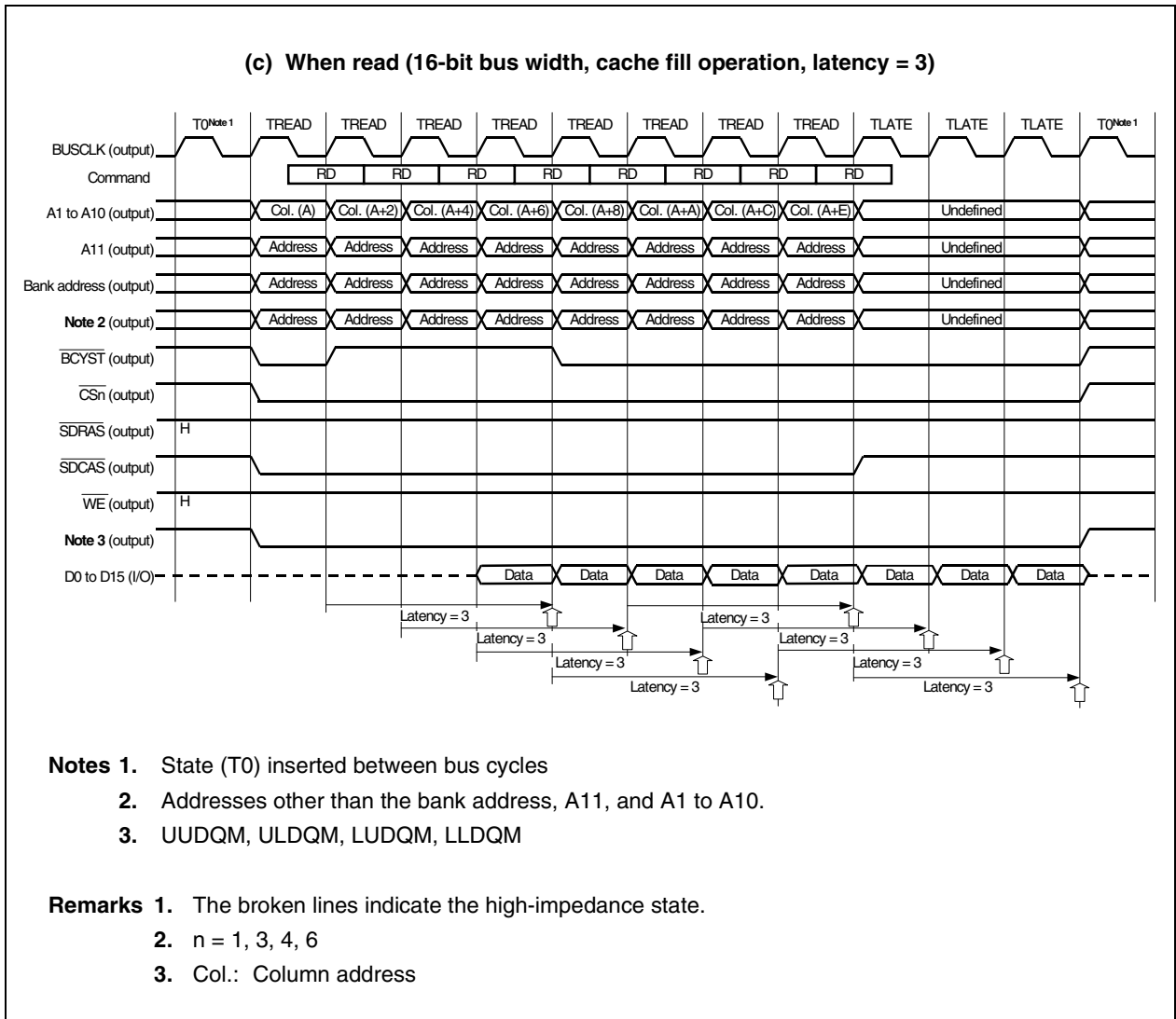
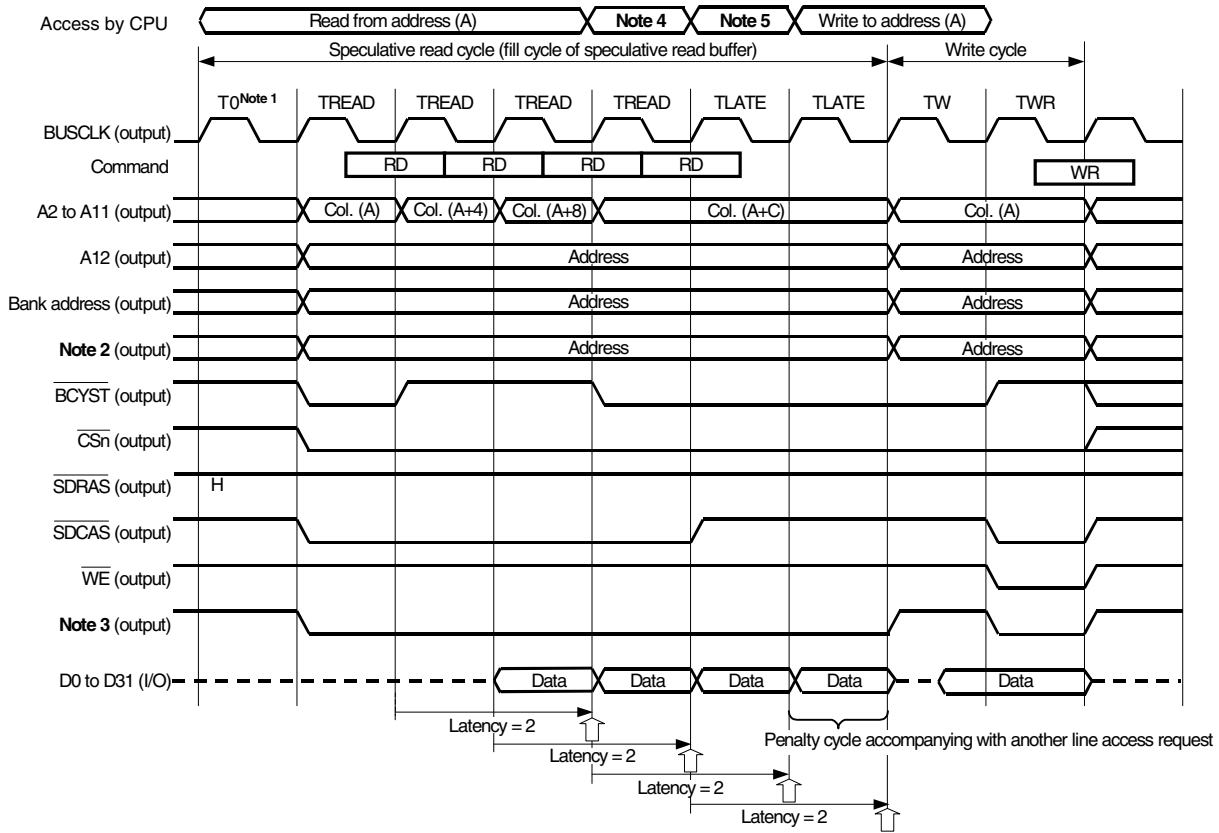


Figure 5-11. SDRAM Access Timing (4/9)

(d) For read → write operation (with speculative read, latency = 2, 32-bit bus width) (1/2)



- Notes**
1. State (T0) inserted between bus cycles
 2. Addresses other than the bank address, A12, and A2 to A11.
 3. UUDQM, ULDQM, LUDQM, LLDQM
 4. Read from address (A+4)
 5. Read from address (A+8)

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. n = 1, 3, 4, 6
 3. Col.: Column address

Figure 5-11. SDRAM Access Timing (5/9)

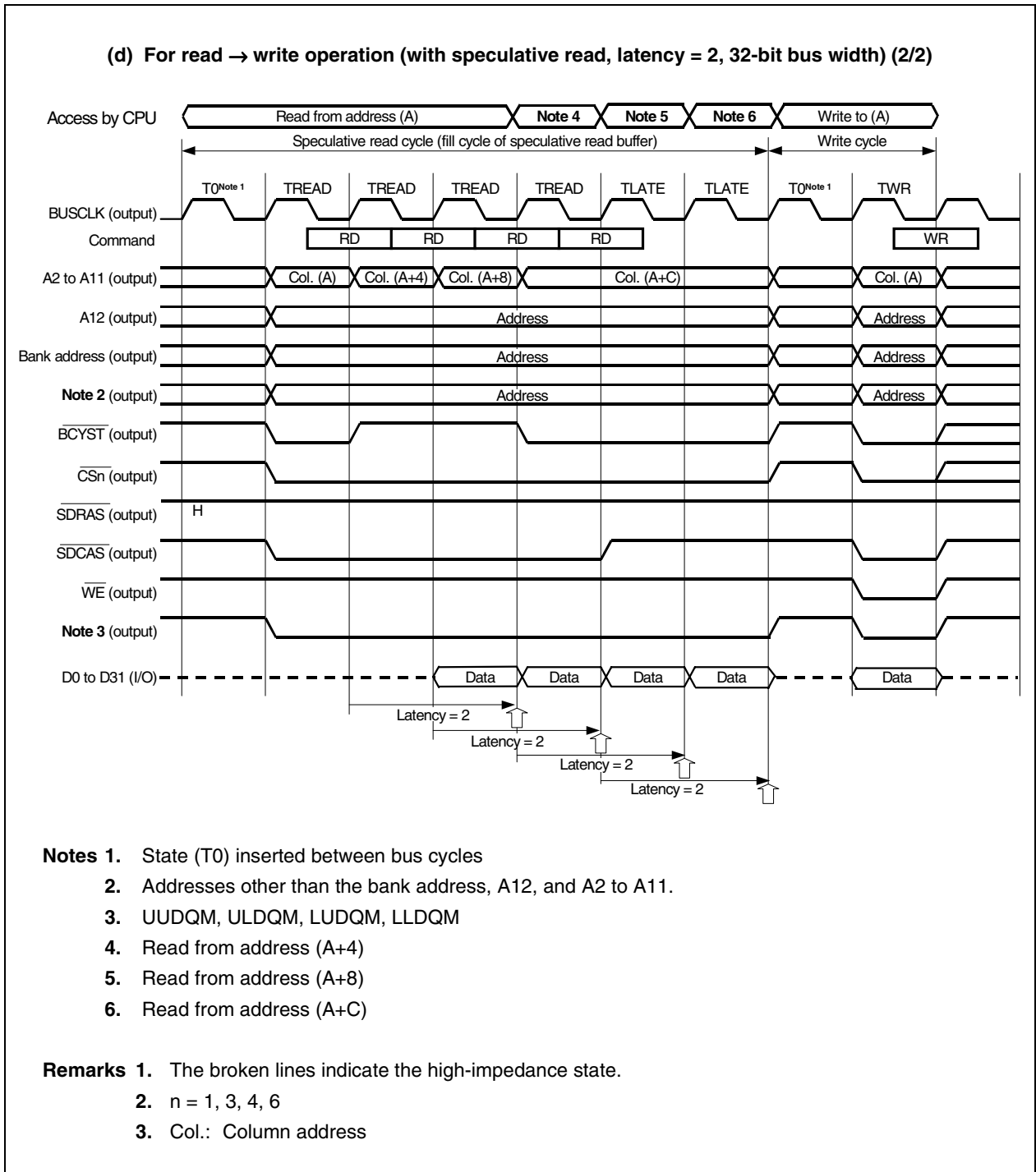
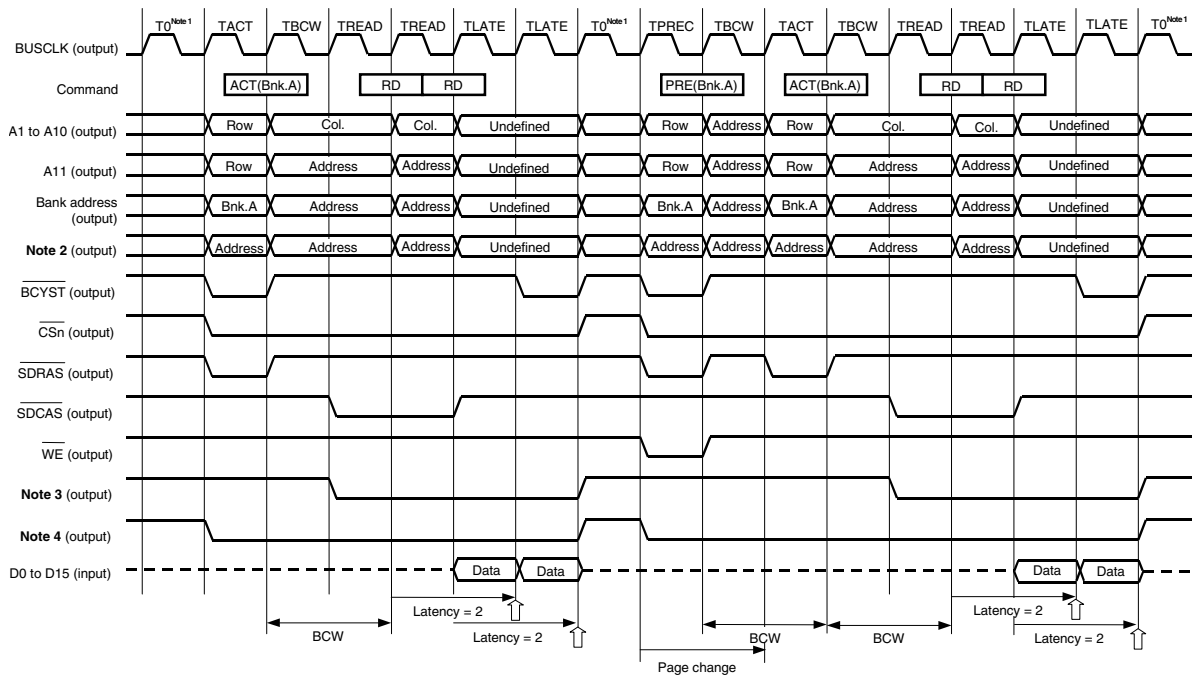


Figure 5-11. SDRAM Access Timing (6/9)

(e) When read (without speculative read, 16-bit bus width word access, page change, BCW = 2, latency = 2)

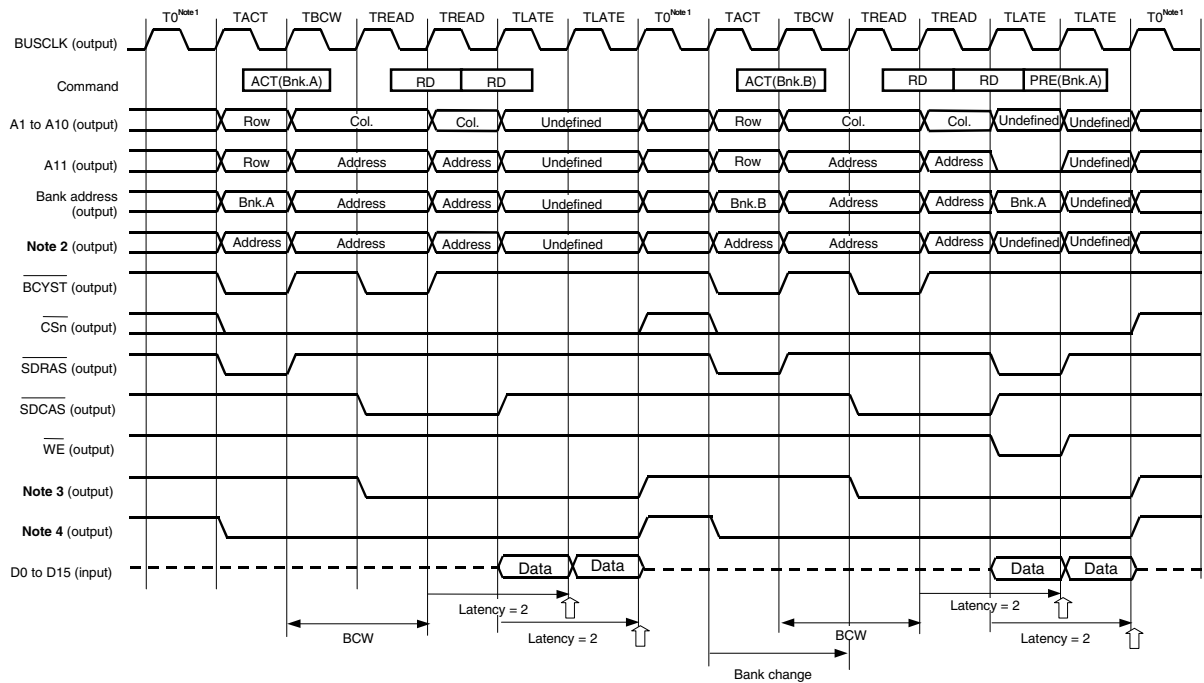


- Notes**
1. State (T0) inserted between bus cycles
 2. Addresses other than the bank address, A11, and A1 to A10
 3. When \overline{xxWR} output mode/ $xxDQM$ output mode is set (PFCCTm bit of PFCCT register = 0)
 4. When $xx\overline{BE}$ output mode/ $xxDQM$ output mode is set (PFCCTm bit of PFCCT register = 1)

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. $n = 1, 3, 4, 6$
 $m = 0$ to 3
 $xx = UU, UL, LU, LL$
 3. Bnk.: Bank address
Col.: Column address
Row: Row address

Figure 5-11. SDRAM Access Timing (7/9)

(f) When read (without speculative read, 16-bit bus width word access, bank change, BCW = 2, latency = 2)



- ★
- Notes 1. State (T0) inserted between bus cycles
 - 2. Addresses other than the bank address, A12, and A2 to A11.
 - ★
 - 3. When xxWR output mode/xxDQM output mode is set (PFCCTm bit of PFCCT register = 0)
 - ★
 - 4. When xxBE output mode/xxDQM output mode is set (PFCCTm bit of PFCCT register = 1)

- Remarks 1. The broken lines indicate the high-impedance state.
- 2. n = 1, 3, 4, 6
m = 0 to 3
xx = UU, UL, LU, LL
 - 3. Bnk.: Bank address
Col.: Column address
Row: Row address

Figure 5-11. SDRAM Access Timing (8/9)

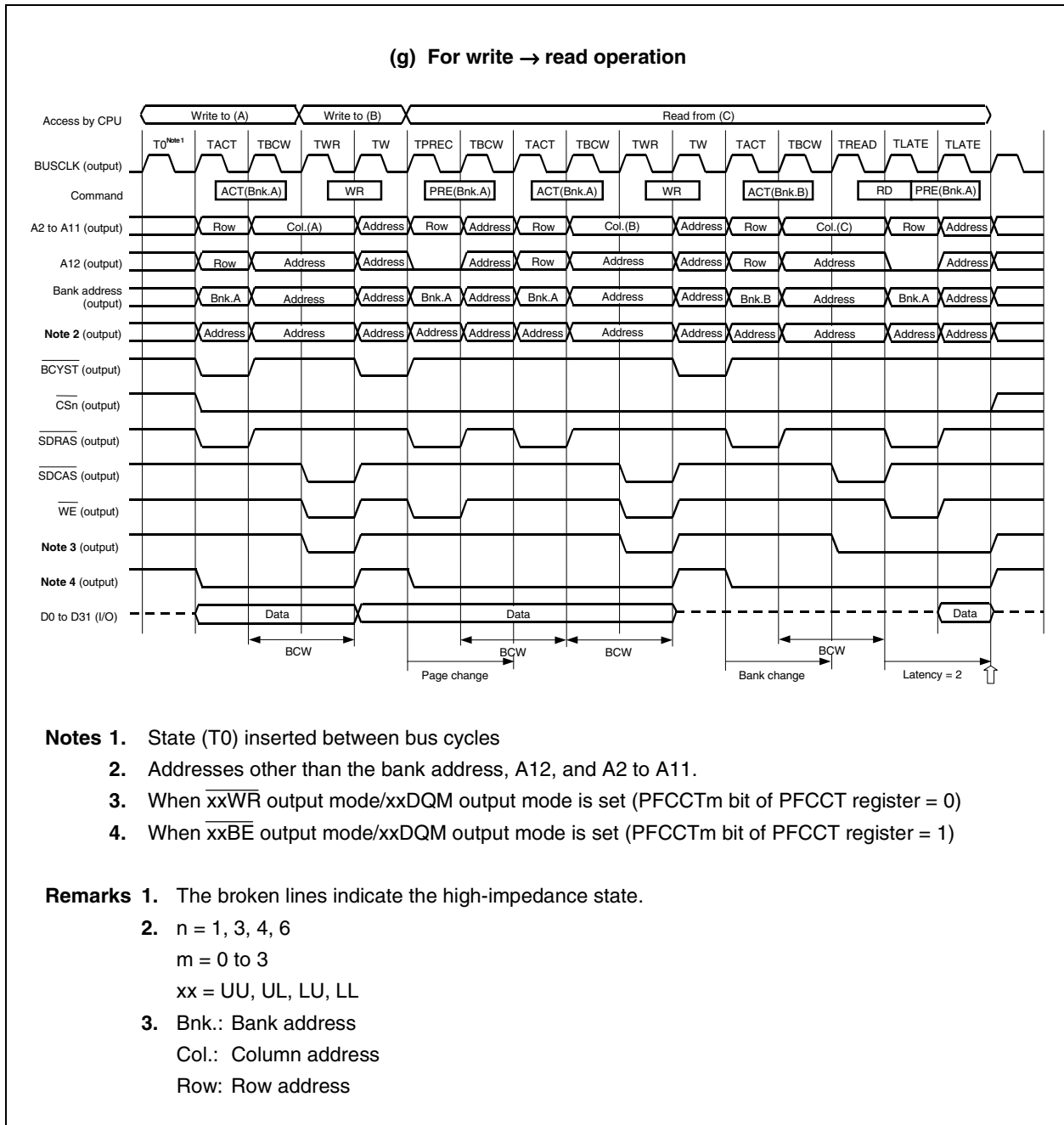
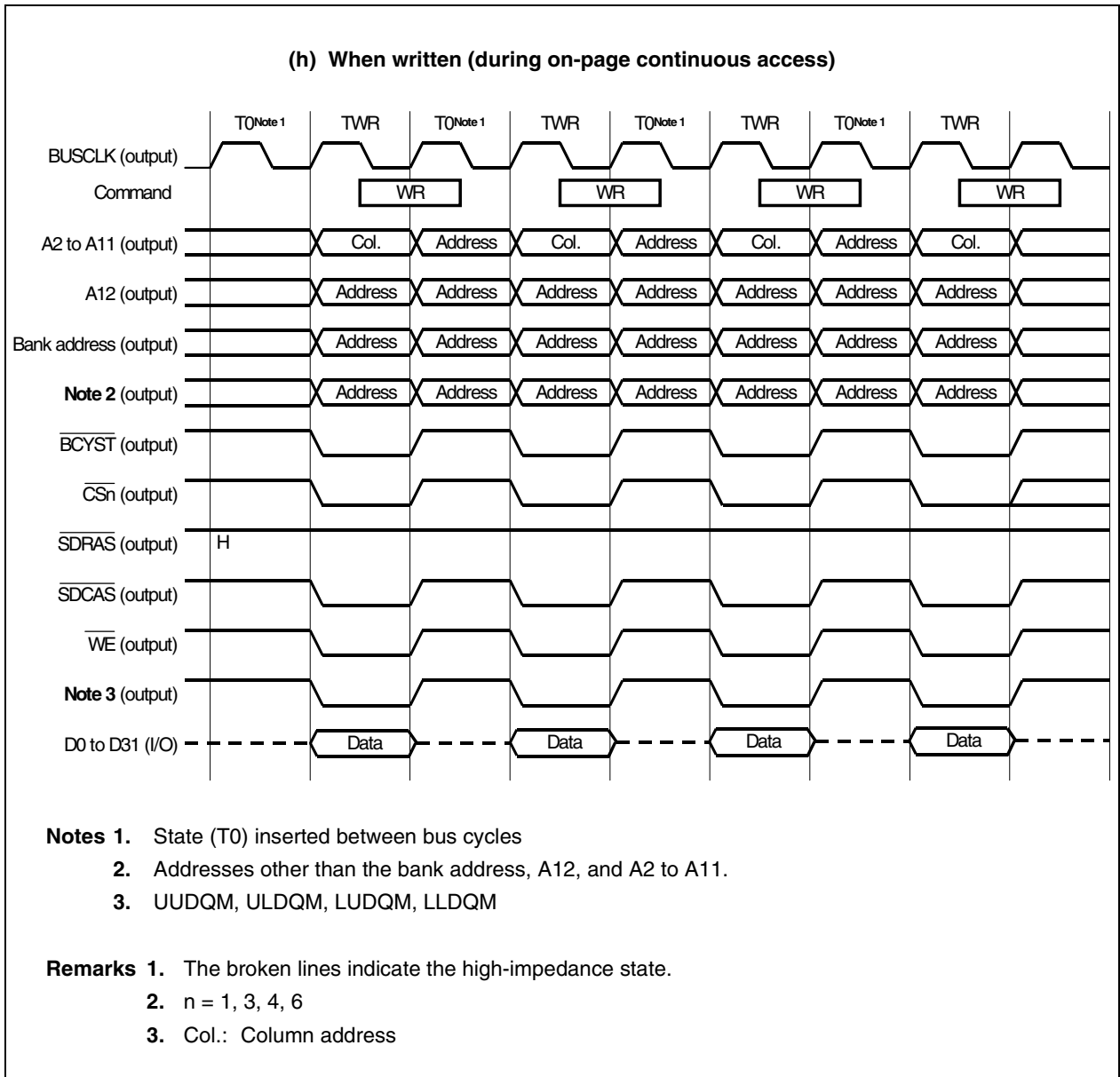


Figure 5-11. SDRAM Access Timing (9/9)



5.3.6 Refresh control function

The V850E/ME2 can generate a refresh cycle. The refresh cycle is set with SDRAM refresh control registers 1, 3, 4, and 6 (RFS1, RFS3, RFS4, RFS6). RFS n corresponds to \overline{CSn} ($n = 1, 3, 4, 6$). For example, to connect SDRAM to $\overline{CS1}$, set RFS1.

When another bus master occupies the external bus, the DRAM controller cannot occupy the external bus. In this case, the DRAM 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 1, 3, 4, 6 (RFS1, RFS3, RFS4, RFS6)

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.

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

The refresh count clock and interval factor are determined by the RCC $n1$ and RCC $n0$ bits and RIN $5n$ to RIN $0n$ bits, respectively, of the RFS n register.

Note that n corresponds to the register number (1, 3, 4, 6) of SDRAM configuration registers 1, 3, 4, 6 (SCR1, SCR3, SCR4, SCR6).

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

★ **Caution** Write to the RFS1, RFS3, RFS4, and RFS6 registers after reset, and then do not change the set values. However, when the SDRAM refresh interval needs to be changed by changing the CKC register set value (internal system clock (f_{CLK})), the set value of the RFS1, RFS3, RFS4, and RFS6 registers can be changed. For details, refer to Caution 2 in 8.3.1 Clock control register (CKC). Also, do not access an external memory area other than the one for this initialization routine until the initial settings of the RFS1, RFS3, RFS4, and RFS6 registers are 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	Address	After reset
RFS1	REN1	0	0	0	0	0	RCC11	RCC10	0	0	RIN15	RIN14	RIN13	RIN12	RIN11	RIN10	FFFFFF4A6H	0000H
RFS3	REN3	0	0	0	0	0	RCC31	RCC30	0	0	RIN35	RIN34	RIN33	RIN32	RIN31	RIN30	FFFFFF4AEH	0000H
RFS4	REN4	0	0	0	0	0	RCC41	RCC40	0	0	RIN45	RIN44	RIN43	RIN42	RIN41	RIN40	FFFFFF4B2H	0000H
RFS6	REN6	0	0	0	0	0	RCC61	RCC60	0	0	RIN65	RIN64	RIN63	RIN62	RIN61	RIN60	FFFFFF4BAH	0000H

Bit position	Bit name	Function																																																	
15	RENn	Specifies whether CBR refresh is enabled or disabled. 0: Refresh disabled 1: Refresh enabled																																																	
9, 8	RCCn1, RCCn0	Specifies the refresh count clock (T_{RCY}). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>RCCn1</th> <th>RCCn0</th> <th>Refresh count clock (T_{RCY})</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>32/BUSCLK</td> </tr> <tr> <td>0</td> <td>1</td> <td>128/BUSCLK</td> </tr> <tr> <td>1</td> <td>0</td> <td>256/BUSCLK</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	RCCn1	RCCn0	Refresh count clock (T_{RCY})	0	0	32/BUSCLK	0	1	128/BUSCLK	1	0	256/BUSCLK	1	1	Setting prohibited																																		
RCCn1	RCCn0	Refresh count clock (T_{RCY})																																																	
0	0	32/BUSCLK																																																	
0	1	128/BUSCLK																																																	
1	0	256/BUSCLK																																																	
1	1	Setting prohibited																																																	
5 to 0	RINn5 to RINn0	Sets the interval factor of the interval timer for the generation of the refresh timing. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>RINn5</th> <th>RINn4</th> <th>RINn3</th> <th>RINn2</th> <th>RINn1</th> <th>RINn0</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>	RINn5	RINn4	RINn3	RINn2	RINn1	RINn0	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
RINn5	RINn4	RINn3	RINn2	RINn1	RINn0	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 n = 1, 3, 4, 6

★

Table 5-1. Example of Interval Factor Settings

Specified Refresh Interval Value (μ s)	Refresh Count Clock (T_{RCY})	Interval Factor Value ^{Notes 1, 2}	
		BUSCLK = 66 MHz	BUSCLK = 50 MHz
15.6	32/BUSCLK	32 (15.5)	24 (15.4)
	128/BUSCLK	8 (15.5)	6 (15.4)
	256/BUSCLK	4 (15.5)	3 (15.4)

Notes 1. The interval factor is set by bits RINn0 to RINn5 of the RFSn register (n = 1, 3, 4, 6).

2. The values in parentheses are the calculated values for the refresh interval (μ s).

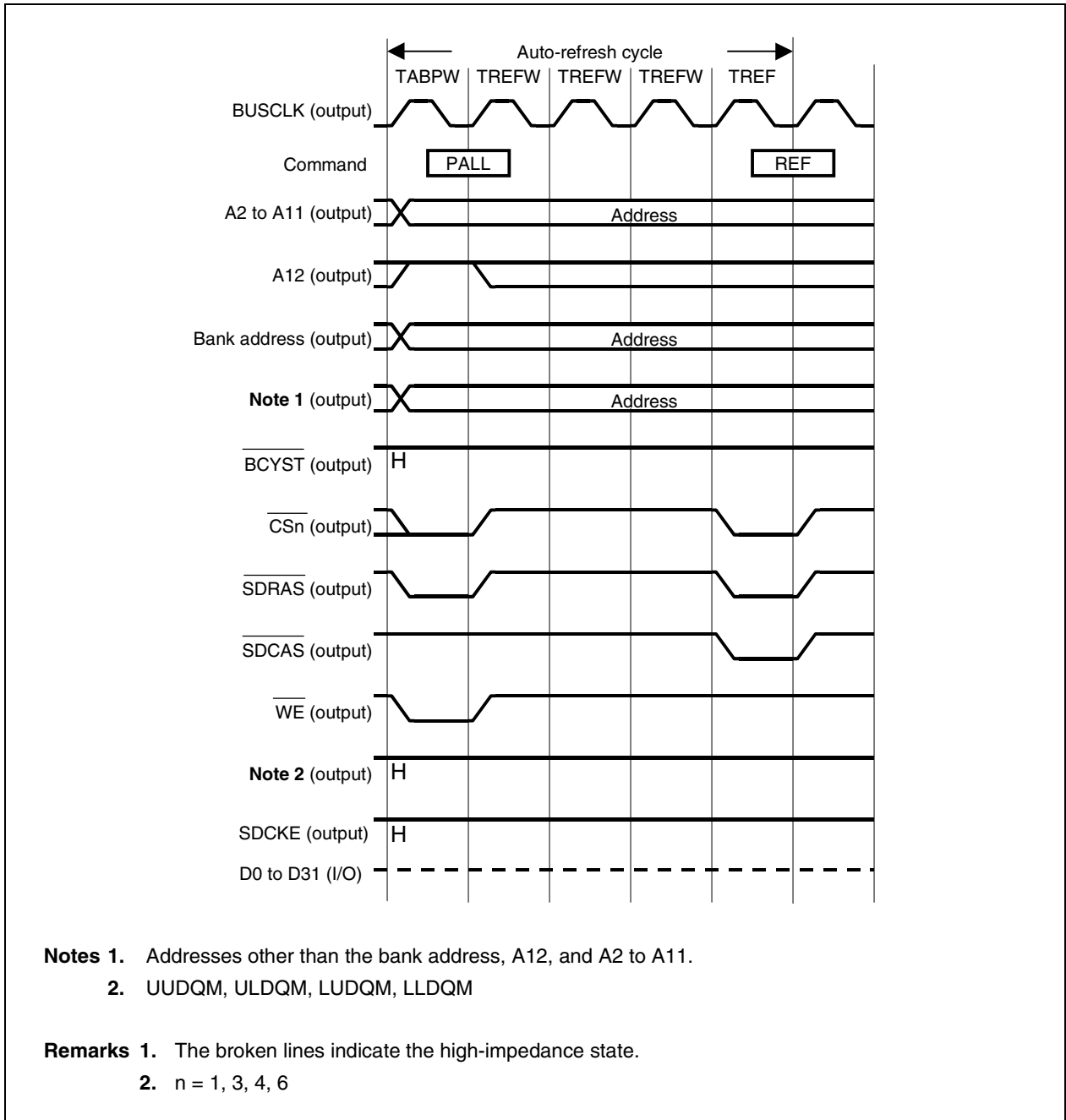
$$\text{Refresh interval } (\mu\text{s}) = \text{Refresh count clock } (T_{RCY}) \times \text{Interval factor}$$

The V850E/ME2 can automatically generate an auto-refresh cycle and a self-refresh cycle.

(2) Auto-refresh cycle

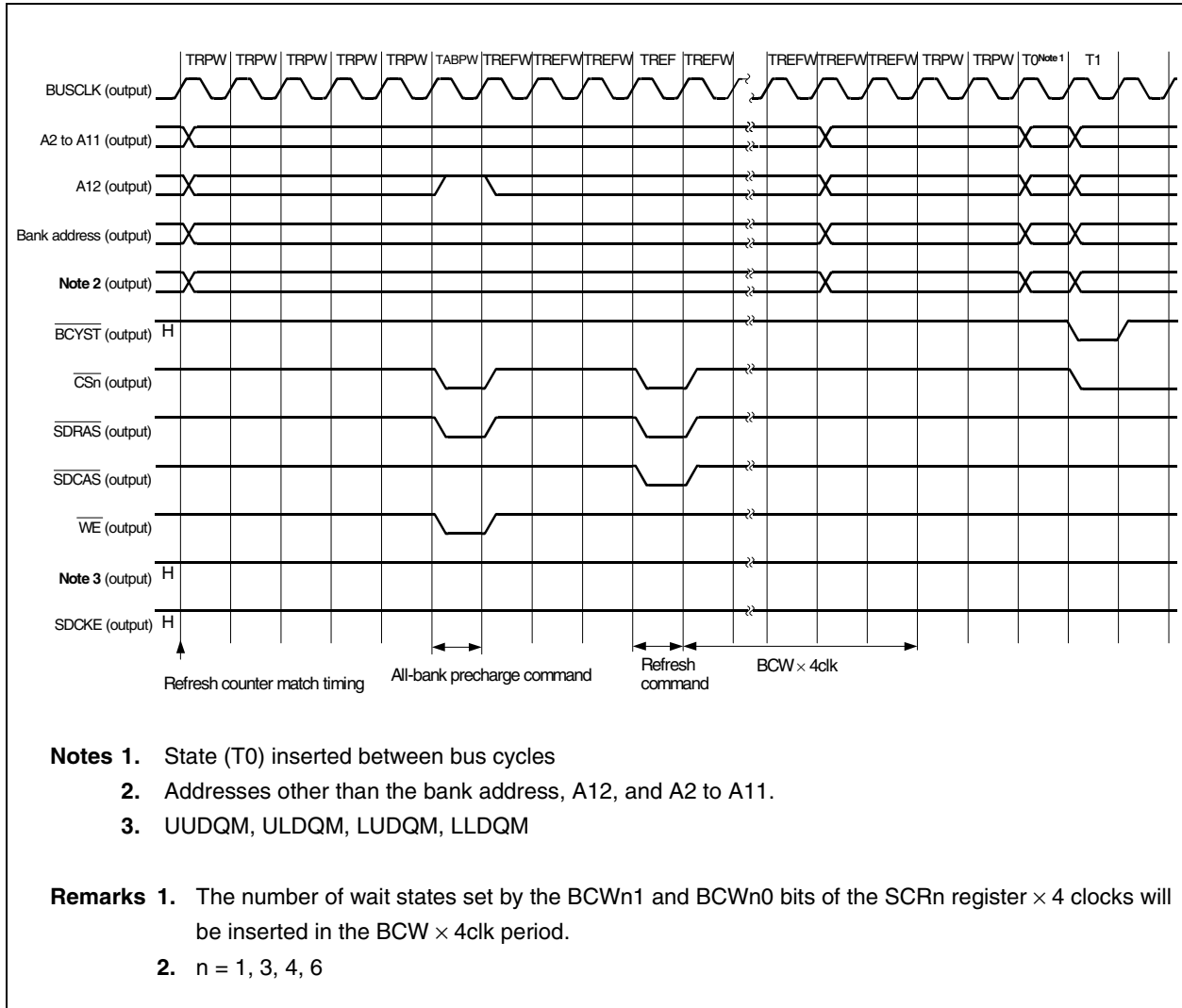
In the auto-refresh cycle, the auto-refresh command (REF) is issued four clocks after the precharge command for all banks (PALL) is issued.

Figure 5-12. Auto-Refresh Cycle



(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 or software STOP mode, or if the $\overline{\text{SELFREF}}$ signal becomes active, the DRAM controller generates the self-refresh cycle.

- Cautions**
1. When the transition to the self-refresh cycle is caused by $\overline{\text{SELFREF}}$ signal input, releasing the self-refresh cycle is only possible by inputting an inactive level to the $\overline{\text{SELFREF}}$ pin.
 2. The internal instruction RAM (only in the read mode) and internal data 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, use one 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 $\overline{\text{SDRAS}}$ and $\overline{\text{SDCAS}}$ signals inactive immediately.

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

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

(2) Release by INTP0n0 and INTP0n1 inputs (n = 0 to 3)

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

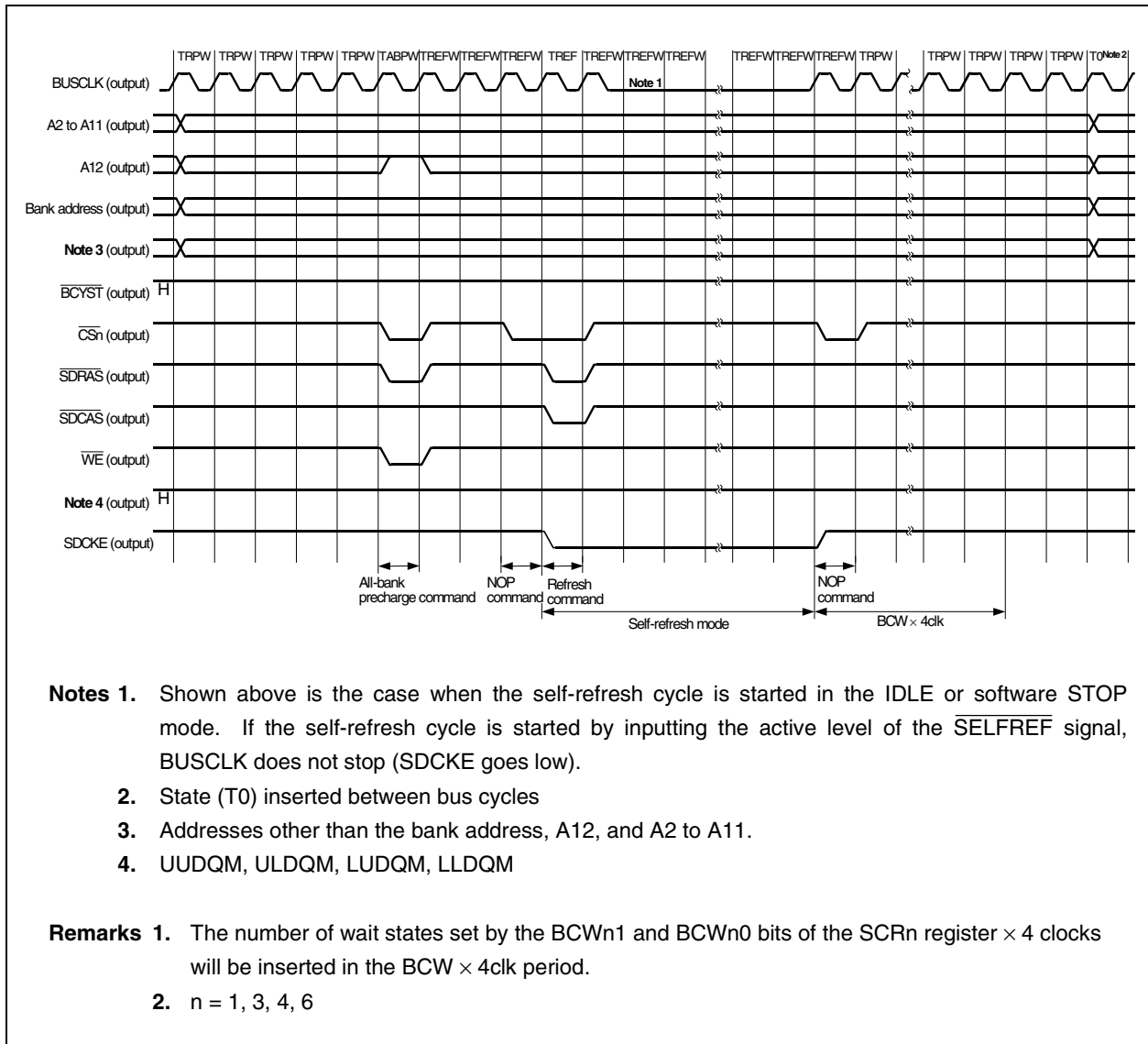
To release the self-refresh cycle, make the $\overline{\text{SDRAS}}$ and $\overline{\text{SDCAS}}$ signals inactive immediately.

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

To release the self-refresh cycle, make the $\overline{\text{SDRAS}}$ and $\overline{\text{SDCAS}}$ 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 register n (SCRn)).
 - Bus cycle type configuration registers 0 and 1 (BCT0 and BCT1)
 - Bus cycle control register (BCC)
 - SDRAM refresh control registers 1, 3, 4, 6 (RFS1, RFS3, RFS4, RFS6)
- (2) Set SDRAM configuration registers 1, 3, 4, 6 (SCR1, SCR3, SCR4, SCR6). When writing data to these registers, the following commands are issued for SDRAM in the order shown below.
 - All-bank precharge command
 - Refresh command (8 times)
 - Command that is used to set a mode register

Cautions 1. To set the SCR1, SCR3, SCR4, and SCR6 registers, confirm that the LOCK bit of the lock register (LOCKR) is set to 1, set the CKSEL bit of the clock source select register (CKS) to 1, and change clock supply to the CPU to SSCG output (see 3.4.10 Initialization sequence).

2. If it is necessary to make the input levels of the UUDQM, ULQDM, LUDQM, and LLDQM pins high until initialization of SDRAM is completed, do not change the set values of the PFCCT3 to PFCCT0 bits of the PFCCT register and do not write the external device until initialization of SDRAM is completed.

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

Figure 5-15. SDRAM Mode Register Setting Cycle

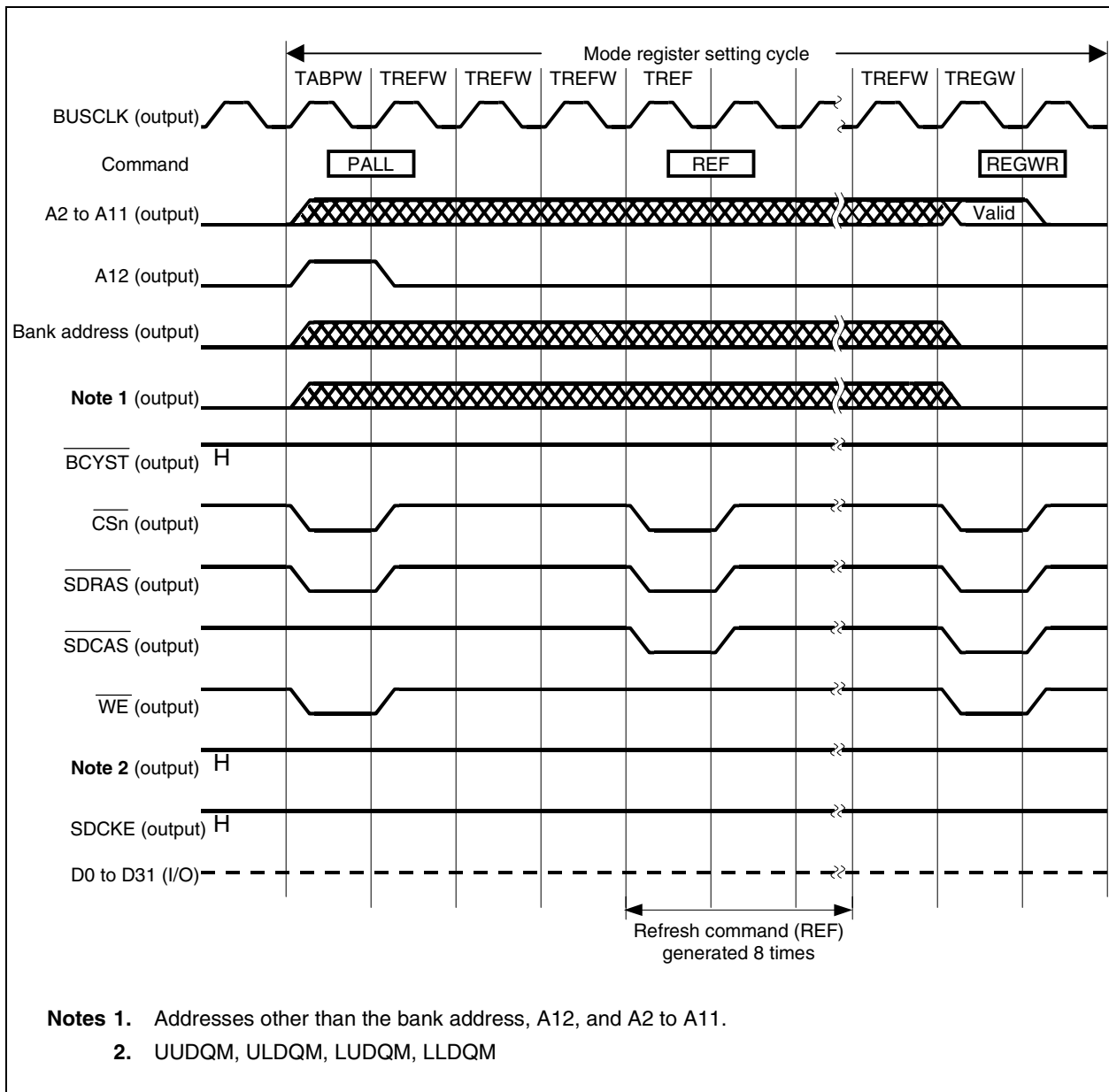
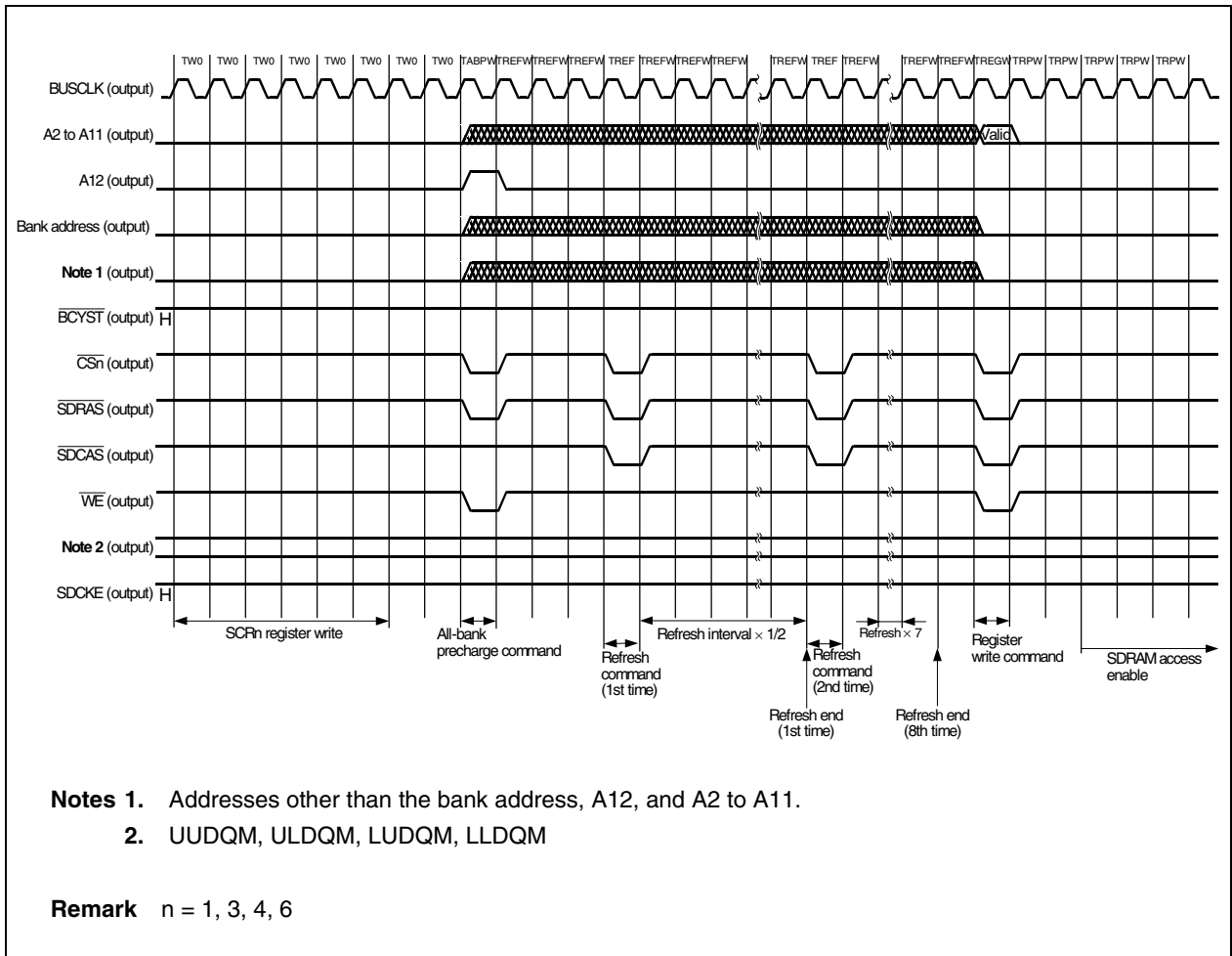


Figure 5-16. SDRAM Register Write Operation Timing



CHAPTER 6 DMA FUNCTIONS (DMA CONTROLLER)

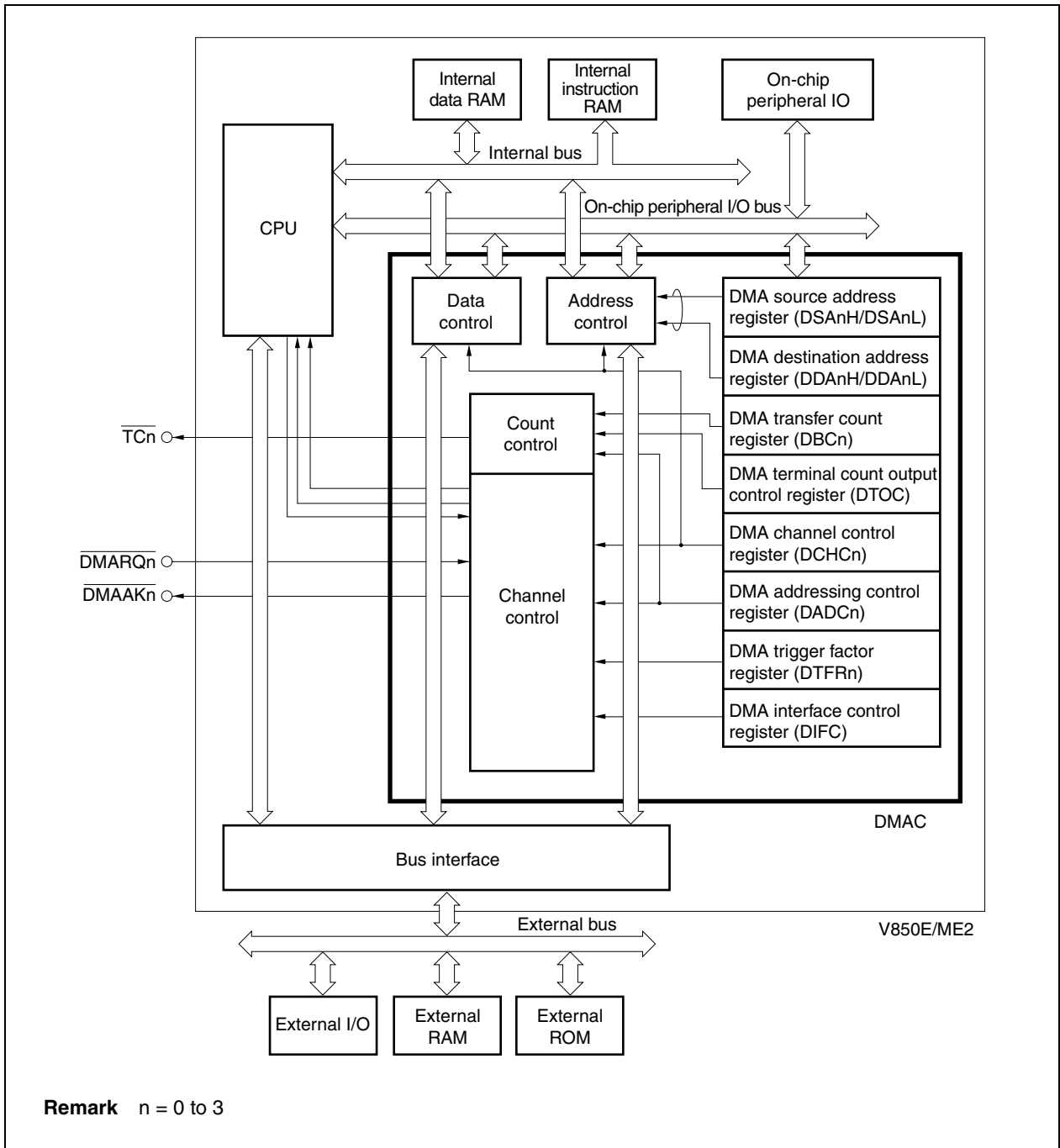
The V850E/ME2 includes a direct memory access (DMA) controller (DMAC) that executes and controls DMA transfer.

The DMAC controls data transfer between memory and I/O, or among memories, based on requests by interrupts from on-chip peripheral I/O (serial interface, real-time pulse unit, and A/D converter) or DMA requests issued by the $\overline{\text{DMARQ0}}$ to $\overline{\text{DMARQ3}}$ pins, software triggers, or USB. Memory refers to the internal instruction RAM, internal data RAM, or external memory. However, the internal instruction RAM can be used only as the transfer destination.

6.1 Features

- 4 independent DMA channels
- Transfer unit: 8/16/32 bits
- Maximum transfer count: 65,536 (2^{16})
- Two types of transfer
 - Flyby (1-cycle) transfer
 - 2-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 via $\overline{\text{DMARQ0}}$ to $\overline{\text{DMARQ3}}$ pin input
 - Requests by software trigger
 - Requests by USB (only in the single transfer mode)
- Transfer objects
 - Memory \leftrightarrow I/O
 - Memory \leftrightarrow memory
- DMA transfer end output signals ($\overline{\text{TC0}}$ to $\overline{\text{TC3}}$)
- 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 transfer source address (28 bits) 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 transfer source address for DMA transfer can be specified during DMA transfer (see **6.8 Next Address Setting Function**). In this case, the newly set value of the DSA_n register is transferred to the slave register and becomes valid only when DMA transfer has been completed normally and the TC_n bit of the DCHC_n register is set to 1, or when the INIT_n bit of the DCHC_n register is set to 1 (n = 0 to 3). However, the set value of the DSA_n register is invalid even when the Enn bit of the DCHC_n register is cleared to 0 to disable DMA transfer and then the DSA_n register is set.

When flyby transfer is specified with the TTYP_n bit of DMA addressing control register n (DADC_n), the external memory addresses are set by the DSA_n register. At this time, the setting of DMA destination address register n (DDA_n) is ignored (n = 0 to 3).

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

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

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

Cautions 1. When setting an address of a peripheral I/O register for the source 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.

★

2. Do not set the DSA_nH register while DMA transfer is suspended.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
DSA0H	IRS0	0	0	0	SA027	SA026	SA025	SA024	SA023	SA022	SA021	SA020	SA019	SA018	SA017	SA016	FFFFF082H	Undefined
DSA1H	IRS1	0	0	0	SA127	SA126	SA125	SA124	SA123	SA122	SA121	SA120	SA119	SA118	SA117	SA116	FFFFF08AH	Undefined
DSA2H	IRS2	0	0	0	SA227	SA226	SA225	SA224	SA223	SA222	SA221	SA220	SA219	SA218	SA217	SA216	FFFFF092H	Undefined
DSA3H	IRS3	0	0	0	SA327	SA326	SA325	SA324	SA323	SA322	SA321	SA320	SA319	SA318	SA317	SA316	FFFFF09AH	Undefined

Bit position	Bit name	Function
15	IRS _n	Specifies the DMA transfer source address. 0: External memory, on-chip peripheral I/O 1: Internal data RAM
11 to 0	SAn27 to SAn16	Sets the DMA transfer source address (A27 to A16). During DMA transfer, it stores the next DMA transfer source address. During flyby transfer, it stores an external memory address.

Remark n = 0 to 3

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

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

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
DSA0L	SA015	SA014	SA013	SA012	SA011	SA010	SA09	SA08	SA07	SA06	SA05	SA04	SA03	SA02	SA01	SA00	FFFFFF080H	Undefined
DSA1L	SA115	SA114	SA113	SA112	SA111	SA110	SA19	SA18	SA17	SA16	SA15	SA14	SA13	SA12	SA11	SA10	FFFFFF088H	Undefined
DSA2L	SA215	SA214	SA213	SA212	SA211	SA210	SA29	SA28	SA27	SA26	SA25	SA24	SA23	SA22	SA21	SA20	FFFFFF090H	Undefined
DSA3L	SA315	SA314	SA313	SA312	SA311	SA310	SA39	SA38	SA37	SA36	SA35	SA34	SA33	SA32	SA31	SA30	FFFFFF098H	Undefined

Bit position	Bit name	Function
15 to 0	SAn15 to SAn0	Sets the DMA transfer source address (A15 to A0). During DMA transfer, it stores the next DMA transfer source address. During flyby transfer, it stores an external memory address.

Remark n = 0 to 3

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

These registers are used to set the DMA transfer destination address (28 bits) 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 transfer destination address for DMA transfer can be specified during DMA transfer (see **6.8 Next Address Setting Function**). In this case, the newly set value of the DDAn register is transferred to the slave register and becomes valid only when DMA transfer has been completed normally and the TCn bit of the DCHCn register is set to 1, or when the INITn bit of the DCHCn register is set to 1 (n = 0 to 3). However, the set value of the DDAn register is invalid even when the Enn bit of the DCHCn register is cleared to 0 to disable DMA transfer and then the DDAn register is set.

When flyby transfer is specified with the TTYPn bit of DMA addressing control register n (DADCn), the setting of DMA destination address register n (DDAn) is ignored.

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

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

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

Cautions 1. When setting an address of a peripheral I/O register for the destination 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.

★

2. Do not set the DDAnH register while DMA transfer is suspended.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
DDA0H	IRD0	0	0	0	DA027	DA026	DA025	DA024	DA023	DA022	DA021	DA020	DA019	DA018	DA017	DA016	FFFFFF086H	Undefined
DDA1H	IRD1	0	0	0	DA127	DA126	DA125	DA124	DA123	DA122	DA121	DA120	DA119	DA118	DA117	DA116	FFFFFF08EH	Undefined
DDA2H	IRD2	0	0	0	DA227	DA226	DA225	DA224	DA223	DA222	DA221	DA220	DA219	DA218	DA217	DA216	FFFFFF096H	Undefined
DDA3H	IRD3	0	0	0	DA327	DA326	DA325	DA324	DA323	DA322	DA321	DA320	DA319	DA318	DA317	DA316	FFFFFF09EH	Undefined

Bit position	Bit name	Function
15	IRDn	Specifies the DMA transfer destination address. 0: External memory, on-chip peripheral I/O, internal instruction RAM 1: Internal data RAM
11 to 0	DAn27 to DAn16	Sets the DMA transfer destination address (A27 to A16). During DMA transfer, it stores the next DMA transfer destination address. This setting is ignored during flyby transfer.

Remark n = 0 to 3

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

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

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
DDA0L	DA015	DA014	DA013	DA012	DA011	DA010	DA09	DA08	DA07	DA06	DA05	DA04	DA03	DA02	DA01	DA00	FFFFFF084H	Undefined
DDA1L	DA115	DA114	DA113	DA112	DA111	DA110	DA19	DA18	DA17	DA16	DA15	DA14	DA13	DA12	DA11	DA10	FFFFFF08CH	Undefined
DDA2L	DA215	DA214	DA213	DA212	DA211	DA210	DA29	DA28	DA27	DA26	DA25	DA24	DA23	DA22	DA21	DA20	FFFFFF094H	Undefined
DDA3L	DA315	DA314	DA313	DA312	DA311	DA310	DA39	DA38	DA37	DA36	DA35	DA34	DA33	DA32	DA31	DA30	FFFFFF09CH	Undefined

Bit position	Bit name	Function
15 to 0	DAn15 to DAn0	Sets the DMA transfer destination address (A15 to A0). During DMA transfer, it stores the next DMA transfer destination address. This setting is ignored during flyby transfer.

Remark n = 0 to 3

6.3.3 DMA transfer count registers 0 to 3 (DBC0 to DBC3)

These 16-bit registers are used to set the byte transfer count for DMA channel n (n = 0 to 3). They store the remaining transfer count 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 (see **6.8 Next Address Setting Function**). In this case, the newly set value of the DBCn register is transferred to the slave register and becomes valid only when DMA transfer has been completed normally and the TCn bit of the DCHCn register is set to 1, or when the INITn bit of the DCHCn register is set to 1 (n = 0 to 3). However, the set value of the DBCn register is invalid even when the Enn bit of the DCHCn register is cleared to 0 to disable DMA transfer and then the DBCn register is set.

These registers are decremented by 1 for each transfer, and transfer ends when a borrow occurs. These registers can be read or written in 16-bit units.

- Cautions 1.** If the DBCn register is read during DMA transfer after a terminal count has occurred without the register being overwritten, the value set immediately before the DMA transfer will be read out (0000H will not be read, even if DMA transfer has ended).
- ★ **2.** Do not set the DBCn register while DMA transfer is suspended.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
DBC0	BC015	BC014	BC013	BC012	BC011	BC010	BC009	BC008	BC007	BC006	BC005	BC004	BC003	BC002	BC001	BC000	FFFFFF0C0H	Undefined
DBC1	BC115	BC114	BC113	BC112	BC111	BC110	BC109	BC108	BC107	BC106	BC105	BC104	BC103	BC102	BC101	BC100	FFFFFF0C2H	Undefined
DBC2	BC215	BC214	BC213	BC212	BC211	BC210	BC209	BC208	BC207	BC206	BC205	BC204	BC203	BC202	BC201	BC200	FFFFFF0C4H	Undefined
DBC3	BC315	BC314	BC313	BC312	BC311	BC310	BC309	BC308	BC307	BC306	BC305	BC304	BC303	BC302	BC301	BC300	FFFFFF0C6H	Undefined

Bit position	Bit name	Function										
15 to 0	BCn15 to BCn0	<p>Sets the byte transfer count and stores the remaining byte transfer count during DMA transfer.</p> <table border="1"> <thead> <tr> <th>DBCn</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>:</td> <td>:</td> </tr> <tr> <td>FFFFH</td> <td>Byte transfer count 65,536 (2¹⁶) or remaining byte transfer count</td> </tr> </tbody> </table>	DBCn	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	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											

Remark n = 0 to 3

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

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

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

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

Cautions 1. The DS_n1 and DS_n0 bits set how many bits of data are to be transferred.

When 8-bit data is set (DS_n1 and DS_n0 bits = 00), the lower bytes of the data bus (D0 to D7) are not always used.

If the transfer data size is set to 16 bits, transfer is always started from an address with the lowest bit of the address aligned to “0”. If the data size is set to 32 bits, transfer is started from an address with the lowest 2 bits of the address aligned to “0”. In this case, transfer cannot be started from an odd address.

- ★ 2. **Set the DADC_n register at the following timing while the target DMA channel is in operation or is not suspended (the operation is not guaranteed if the register is set at any other timing).**
 - **Period from system reset to generation of the first DMA transfer request**
 - **Period from completion of DMA transfer (after generation of terminal count) to generation of the next DMA transfer request**
 - **Period from forced termination of DMA transfer (after the INIT_n bit of the DCHC_n register is set to 1) to generation of the next DMA transfer request**

- ★ 3. **Do not set flyby transfer via the TTYP_n bit if all the following conditions are satisfied (n = 0 to 3).**
 - **BMC register = 00H**
 - **FW_n2 to FW_n0 bits of FWC register = 000**
 - **Fl_n1 to Fl_n0 bits of FIC register = 00**
 - **AC_n1 and AC_n0 bits of the CS_n space subject to flyby transfer by the ASC register = 00 (during transfer to/from SRAM)**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
DADC0	DS01	DS00	0	0	0	0	0	0	SAD01	SAD00	DAD01	DAD00	TM01	TM00	TTYP0	TDIR0	FFFFFF0D0H	0000H
DADC1	DS11	DS10	0	0	0	0	0	0	SAD11	SAD10	DAD11	DAD10	TM11	TM10	TTYP1	TDIR1	FFFFFF0D2H	0000H
DADC2	DS21	DS20	0	0	0	0	0	0	SAD21	SAD20	DAD21	DAD20	TM21	TM20	TTYP2	TDIR2	FFFFFF0D4H	0000H
DADC3	DS31	DS30	0	0	0	0	0	0	SAD31	SAD30	DAD31	DAD30	TM31	TM30	TTYP3	TDIR3	FFFFFF0D6H	0000H

Bit position	Bit name	Function															
15, 14	DSn1, DSn0	<p>Sets the transfer data size for DMA transfer.</p> <table border="1"> <thead> <tr> <th>DSn1</th> <th>DSn0</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>32 bits</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	DSn1	DSn0	Transfer data size	0	0	8 bits	0	1	16 bits	1	0	32 bits	1	1	Setting prohibited
DSn1	DSn0	Transfer data size															
0	0	8 bits															
0	1	16 bits															
1	0	32 bits															
1	1	Setting prohibited															
7, 6	SADn1, SADn0	<p>Sets the count direction of the source address for DMA channel n.</p> <table border="1"> <thead> <tr> <th>SADn1</th> <th>SADn0</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>	SADn1	SADn0	Count direction	0	0	Increment	0	1	Decrement	1	0	Fixed	1	1	Setting prohibited
SADn1	SADn0	Count direction															
0	0	Increment															
0	1	Decrement															
1	0	Fixed															
1	1	Setting prohibited															

Remark n = 0 to 3

Bit position	Bit name	Function															
5, 4	DADn1, DADn0	<p>Sets the count direction of the destination address for DMA channel n.</p> <table border="1"> <thead> <tr> <th>DADn1</th> <th>DADn0</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>	DADn1	DADn0	Count direction	0	0	Increment	0	1	Decrement	1	0	Fixed	1	1	Setting prohibited
DADn1	DADn0	Count direction															
0	0	Increment															
0	1	Decrement															
1	0	Fixed															
1	1	Setting prohibited															
3, 2	TMn1, TMn0	<p>Sets the transfer mode during DMA transfer.</p> <table border="1"> <thead> <tr> <th>TMn1</th> <th>TMn0</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>	TMn1	TMn0	Transfer mode	0	0	Single transfer mode	0	1	Single-step transfer mode	1	0	Setting prohibited	1	1	Block transfer mode
TMn1	TMn0	Transfer mode															
0	0	Single transfer mode															
0	1	Single-step transfer mode															
1	0	Setting prohibited															
1	1	Block transfer mode															
1	TTYPn	<p>Sets the DMA transfer type.</p> <p>0: 2-cycle transfer 1: Flyby transfer</p>															
0	TDIRn	<p>Sets the transfer direction during transfer between I/O and memory. The setting is valid during flyby transfer only and ignored during 2-cycle transfer.</p> <p>0: Memory → I/O (read) 1: I/O → memory (write)</p>															

Remark n = 0 to 3

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

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

These registers can be read or 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 clear bits 6 to 4 to 0. If they are set to 1, the operation is not guaranteed.

Cautions 1. During a 2-cycle transfer that performs a write operation on the external device, the write operation on the external device may not be completed even if the TCn bit of the DCHCn register is set to 1 (DMA transfer completion) by the write buffer function. Recognize completion of the write operation on the external device as follows, if necessary.

- Monitor the signal of the $\overline{\text{TCn}}$ pin (the $\overline{\text{TCn}}$ pin becomes active in synchronization with the write operation on the external device).
 - After it has been detected that the TCn bit of the DCHCn register is set to 1, write the same value as that set to the LBC0 or LBC1 register. When this dummy write is completed, completion of an access from the write buffer to the external device can be recognized. If the set value of the LBC0 or LBC1 register is rewritten when a dummy write is executed, the operation is not guaranteed.
2. Setting the MLEn bit to 1 is valid only when DMA transfer (hardware DMA) is started by $\overline{\text{DMARQn}}$ pin input or an interrupt from the on-chip peripheral I/O. To start DMA transfer by setting the STGn bit to 1 (software DMA), read the TCn bit and confirm that it is set to 1, and then set the STGn bit to 1.
 - ★ 3. Set the MLEn bit at the following timing while the target DMA channel is in operation or is not suspended (the operation is not guaranteed if the register is set at any other timing).
 - Period from system reset to generation of the first DMA transfer request
 - Period from completion of DMA transfer (after generation of terminal count) to generation of the next DMA transfer request
 - Period from forced termination of DMA transfer (after the INITn bit is set to 1) to generation of the next DMA transfer request
 - ★ 4. If DMA transfer is forcibly terminated in the last transfer cycle with the MLEn bit set to 1, the same operation as that performed when transfer is completed is performed (the TCn bit is set to 1). (The Enn bit is cleared to 0 on completion of forced termination, regardless of the value of the MLEn bit.)
In this case, the TCn bit must be read (cleared to 0) in addition to setting the Enn bit when the next DMA transfer is requested.
 - ★ 5. Do not set the Enn and STGn bits while DMA is suspended.
 - ★ 6. Each bit is updated upon completion of DMA transfer (at terminal count) with the Enn bit cleared to 0 and the TCn bit set to 1 in that order. While the statuses of the TCn and Enn bits are being polled, therefore, values indicating the status of “transfer not completed and prohibited” (TCn bit = 0 and Enn bit = 0) may be read if the DCHCn register is read while each bit is being updated.

★ **Cautions 7.** The TCn bit does not have to be read (cleared to 0) on completion of DMA transfer (on generation of the terminal count) only if the following two conditions are satisfied. If either of the conditions is not satisfied, be sure to read the TCn bit (cleared to 0) before the next DMA transfer request is generated.

- If the MLEn bit is set to 1 upon completion of DMA transfer (at terminal count)
- If the source that starts the next DMA transfer is an external pin ($\overline{\text{DMARQn}}$)

If these two conditions are not satisfied, the operation is not guaranteed if the next DMA transfer request is generated with the TCn bit set to 1.

	<7>	6	5	4	<3>	<2>	<1>	<0>	Address	After reset
DCHC0	TC0	0	0	0	MLE0	INIT0	STG0	E00	FFFFFF0E0H	00H
DCHC1	TC1	0	0	0	MLE1	INIT1	STG1	E11	FFFFFF0E2H	00H
DCHC2	TC2	0	0	0	MLE2	INIT2	STG2	E22	FFFFFF0E4H	00H
DCHC3	TC3	0	0	0	MLE3	INIT3	STG3	E33	FFFFFF0E6H	00H

Bit position	Bit name	Function
7	TCn	<p>This status bit indicates whether DMA transfer through DMA channel n is complete or not.</p> <p>This bit is read-only. It is set to 1 during the last DMA transfer and cleared to 0 when it is read.</p> <p>0: DMA transfer is not complete. 1: DMA transfer is complete.</p> <p>Caution To read the TCn bit on completion of a DMA transfer that transfers data to/from the internal data RAM, first read the status in which the TCn bit was set to 1, and then insert two dummy reads of the DCHCn register in a row.</p>
3	MLEn	<p>If this bit is set to 1 when DMA transfer is complete (at terminal count output), the Enn bit is not cleared to 0 and the DMA transfer enable state is retained.</p> <p>If the next DMA transfer startup factor is input from the $\overline{\text{DMARQn}}$ pin or is an interrupt from the on-chip peripheral I/O (hardware DMA), the DMA transfer request is acknowledged even if the TCn bit is not read.</p> <p>If the next DMA transfer startup factor is input by setting the STGn bit to 1 (software DMA), the DMA transfer request is acknowledged if the TCn bit is read and cleared to 0.</p> <p>If this bit is cleared to 0 when DMA transfer is complete (at terminal count output), the Enn bit is cleared to 0 and the DMA transfer disable state is entered. At the next DMA request, the Enn bit must be set to 1 and the TCn bit read.</p>
2	INITn	<p>If this bit is set to 1 during DMA transfer or while DMA transfer is suspended, DMA transfer is forcibly terminated.</p>
1	STGn	<p>If this bit is set to 1 in the DMA transfer enable state (TCn bit = 0, Enn bit = 1), DMA transfer is started.</p>
0	Enn	<p>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 interrupted or forcibly terminated by setting the INITn bit to 1 or by NMI input.</p> <p>0: DMA transfer disabled 1: DMA transfer enabled</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. Do not terminate a DMA transfer in progress by clearing the Enn bit to 0. 2. If the Enn bit is set to 1, do not set the Enn bit until DMA transfer is completed the number of times specified by the DBCn register or until DMA transfer is forcibly terminated by using the INITn bit.

★ Remark n = 0 to 3

6.3.6 DMA terminal count output control register (DTCO)

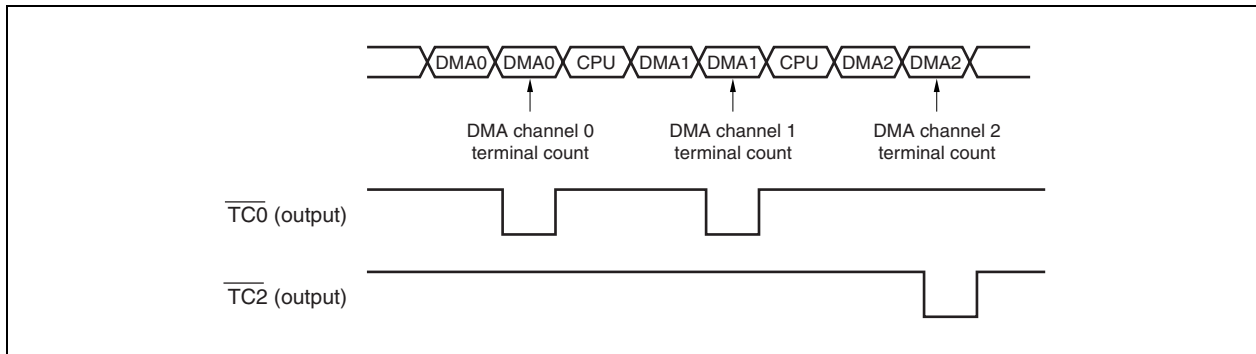
The DMA terminal count output control register (DTCO) is an 8-bit register that controls the terminal count output from each DMA channel and DMA transfer during NMI input. Terminal count signals from each DMA channel can be brought together and output from the $\overline{TC0}$ pin.

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
DTCO	DMSTPM	DAKEBC	0	0	TCO3	TCO2	TCO1	TCO0	FFFFF8A0H	01H

Bit position	Bit name	Function
7	DMSTPM	<p>Controls DMA transfer when NMI is input.</p> <p>0: Forcibly aborts DMA transfer when NMI is input. 1: Does not abort DMA transfer when NMI is input.</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. When DMSTPM bit = 0, NMI servicing can be executed immediately after completion of the DMA cycle currently under execution. Before executing the aborted DMA transfer, however, be sure to re-initialize it. 2. When DMSTPM bit = 1, NMI servicing is held pending in the block transfer mode until DMA transfer has been completed the preset number of times. In the single transfer mode and single-step transfer mode, NMI servicing is executed after the DMA cycle currently under execution is completed. As necessary, forcibly terminate DMA transfer by setting the INITn bit of the DCHCn register to 1 (n = 0 to 3). 3. Be sure to change the value of the DMSTPM bit when DMA is not used. The operation is not guaranteed if the value of the DMSTPM bit is changed after registers related to DMAC have been set or during DMA transfer.
6	DAKEBC	<p>Specifies extension of the active width of the \overline{DMAAKn} signal while the DAKEn bit of the DIFC register = 1.</p> <p>0: Active width extended by four internal system clocks. 1: Active width extended by six to seven internal system clocks.</p> <p>Remark For details of the function to extend the active width of the \overline{DMAAKn} signal, refer to 6.3.8 DMA interface control register (DIFC) or 6.5.1 (2) DMAAKn signal active width extension function.</p>
3 to 0	TCO3 to TCO0	<p>Indicates the state of the $\overline{TC0}$ pin.</p> <p>0: Channel n terminal count signal not output from $\overline{TC0}$ pin. 1: Channel n terminal count signal output from $\overline{TC0}$ pin.</p>

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



6.3.7 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 by these registers serve as DMA transfer startup factors.

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

Cautions 1. To change 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.

★ **3. If the factor of starting DMA transfer is changed by using the IFCn6 to IFCn0 bits, be sure to clear the DFn bit to 0 using the instruction immediately after.**

★ **4. When a transmission completion interrupt request signal (UBTIT0 or UBTIT1) of UARTB is used as the factor of starting DMA transfer, the factor of starting DMA transfer that is triggered by the interrupt request signal generated when the last transmission was completed is retained. In this case, clear the DFn bit to 0 to clear the DMA transfer request.**

	<7>	6	5	4	3	2	1	0	Address	After reset
DTFR0	DF0	IFC06	IFC05	IFC04	IFC03	IFC02	IFC01	IFC00	FFFFFF810H	00H
DTFR1	DF1	IFC16	IFC15	IFC14	IFC13	IFC12	IFC11	IFC10	FFFFFF812H	00H
DTFR2	DF2	IFC26	IFC25	IFC24	IFC23	IFC22	IFC21	IFC20	FFFFFF814H	00H
DTFR3	DF3	IFC36	IFC35	IFC34	IFC33	IFC32	IFC31	IFC30	FFFFFF816H	00H

Bit position	Bit name	Function																																																																																
7	DFn	<p>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 trigger 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 will not occur until DMA transfer is resumed next, it is not necessary to stop the operation of the source causing the interrupt.</p>																																																																																
6 to 0	IFCn6 to IFCn0	<p>This code is used to set the interrupt sources serving as DMA transfer startup factors.</p> <table border="1"> <thead> <tr> <th>IFCn6</th> <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>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>Level detection mode of $\overline{\text{DMARQn}}$ pin input (DMA request from on-chip peripheral I/O disabled)</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>Edge detection mode of $\overline{\text{DMARQn}}$ pin input (DMA request from on-chip peripheral I/O disabled)</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>INTP10</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>INTP11</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>INTP21</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td>INTP22</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>0</td> <td>INTP23</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>INTP24</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>INTP25</td> </tr> </tbody> </table>	IFCn6	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source	0	0	0	0	0	0	0	Level detection mode of $\overline{\text{DMARQn}}$ pin input (DMA request from on-chip peripheral I/O disabled)	0	0	0	0	0	0	1	Edge detection mode of $\overline{\text{DMARQn}}$ pin input (DMA request from on-chip peripheral I/O disabled)	0	0	0	0	0	1	0	INTP10	0	0	0	0	0	1	1	INTP11	0	0	0	0	1	0	0	INTP21	0	0	0	0	1	0	1	INTP22	0	0	0	0	1	1	0	INTP23	0	0	0	0	1	1	1	INTP24	0	0	0	1	0	0	0	INTP25
IFCn6	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source																																																																											
0	0	0	0	0	0	0	Level detection mode of $\overline{\text{DMARQn}}$ pin input (DMA request from on-chip peripheral I/O disabled)																																																																											
0	0	0	0	0	0	1	Edge detection mode of $\overline{\text{DMARQn}}$ pin input (DMA request from on-chip peripheral I/O disabled)																																																																											
0	0	0	0	0	1	0	INTP10																																																																											
0	0	0	0	0	1	1	INTP11																																																																											
0	0	0	0	1	0	0	INTP21																																																																											
0	0	0	0	1	0	1	INTP22																																																																											
0	0	0	0	1	1	0	INTP23																																																																											
0	0	0	0	1	1	1	INTP24																																																																											
0	0	0	1	0	0	0	INTP25																																																																											

Remark n = 0 to 3

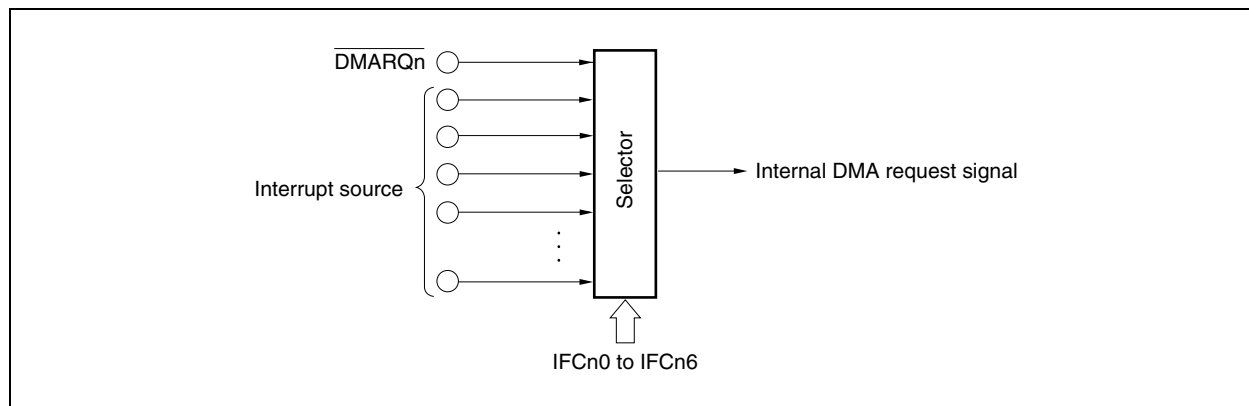
Bit position	Bit name	Function							
6 to 0	IFCn6 to IFCn0	IFCn6	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source
		0	0	0	1	0	0	1	INTP50
		0	0	0	1	0	1	0	INTP51
		0	0	0	1	0	1	1	INTP52
		0	0	0	1	1	0	0	INTP65
		0	0	0	1	1	0	1	INTP66
		0	0	0	1	1	1	0	INTP67
		0	0	0	1	1	1	1	INTPD0
		0	0	1	0	0	0	0	INTPD1
		0	0	1	0	0	0	1	INTPD2
		0	0	1	0	0	1	0	INTPD3
		0	0	1	0	0	1	1	INTPD4
		0	0	1	0	1	0	0	INTPD5
		0	0	1	0	1	0	1	INTPD6
		0	0	1	0	1	1	0	INTPD7
		0	0	1	0	1	1	1	INTPD8
		0	0	1	1	0	0	0	INTPD9
		0	0	1	1	0	0	1	INTPD10
		0	0	1	1	0	1	0	INTPD11
		0	0	1	1	0	1	1	INTPD12
		0	0	1	1	1	0	0	INTPD13
		0	0	1	1	1	0	1	INTPD14
		0	0	1	1	1	1	0	INTPD15
		0	0	1	1	1	1	1	INTPL0
		0	1	0	0	0	0	0	INTPL1
		0	1	0	0	0	0	1	INTPC00/INTCCC00
		0	1	0	0	0	1	0	INTPC01/INTCCC01
		0	1	0	0	0	1	1	INTPC10/INTCCC10
		0	1	0	0	1	0	0	INTPC11/INTCCC11
		0	1	0	0	1	0	1	INTPC20/INTCCC20
		0	1	0	0	1	1	0	INTPC21/INTCCC21
		0	1	0	0	1	1	1	INTPC30/INTCCC30
0	1	0	1	0	0	0	INTPC31/INTCCC31		

Remark n = 0 to 3

Bit position	Bit name	Function								
6 to 0	IFCn6 to IFCn0	IFCn6	IFCn5	IFCn4	IFCn3	IFCn2	IFCn1	IFCn0	Interrupt source	
		0	1	0	1	0	0	1	INTCCC40	
		0	1	0	1	0	1	0	INTCCC41	
		0	1	0	1	0	1	1	INTCCC50	
		0	1	0	1	1	0	0	INTCCC51	
		0	1	0	1	1	0	1	INTCMD0	
		0	1	0	1	1	1	0	INTCMD1	
		0	1	0	1	1	1	1	INTCMD2	
		0	1	1	0	0	0	0	INTCMD3	
		0	1	1	0	0	0	1	INTCC100	
		0	1	1	0	0	1	0	INTCC101	
		0	1	1	0	0	1	1	INTCM100	
		0	1	1	0	1	0	0	INTCM101	
		0	1	1	0	1	0	1	INTCC110	
		0	1	1	0	1	1	0	INTCC111	
		0	1	1	0	1	1	1	INTCM110	
		0	1	1	1	0	0	0	INTCM111	
		0	1	1	1	0	0	1	INTCSI30	
		0	1	1	1	0	1	0	INTCSI31	
		0	1	1	1	0	1	1	UBTIR0	
		0	1	1	1	1	0	0	UBTIT0	
		0	1	1	1	1	0	1	UBTIR1	
		0	1	1	1	1	1	0	UBTIT1	
		0	1	1	1	1	1	1	INTAD	
		1	1	1	1	1	1	1	UFDRQn	
		Other than above								Setting prohibited
		<p>Caution When using an external interrupt (when IFCn6 to IFCn0 bits = 0000010B to 0100000B) as a DMA trigger source, be sure to specify an edge (do not use level detection).</p>								

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 ($n = 0$ to 3).



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.

For the level detection mode (IFCn6 to IFCn0 bits = 0000000) and edge detection mode (IFCn6 to IFCn0 bits = 0000001) of the $\overline{\text{DMARQn}}$ pin input, see **Figures 6-8 to 6-10** ($n = 0$ to 3).

★ (1) DMA request detection function

The V850E/ME2 has a level detection mode and an edge detection mode as functions to sample the $\overline{\text{DMARQn}}$ pin ($n = 0$ to 3). The level detection mode also has a mask mode in which sampling the DMA request is masked.

(a) Level detection mode (IFCn6 to IFCn0 bits = 0000000)

In this mode, handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals can be executed at high speeds. In the single transfer mode, deassert the $\overline{\text{DMARQn}}$ signal for the duration of twice the internal system clock from the rising edge of the $\overline{\text{DMAAKn}}$ signal, in order not to start the DMA transfer cycle next to that currently under execution. If the $\overline{\text{DMARQn}}$ signal is active for the duration of twice the internal system clock, the next DMA request is recognized. In the 2-cycle transfer mode, however, the active period of the $\overline{\text{DMAAKn}}$ signal may be shortened depending on the combination of the transfer source and transfer destination (Min.: $2 \times$ internal system clock). Consequently, if the inactive timing of the $\overline{\text{DMARQn}}$ signal is generated from the falling edge of the $\overline{\text{DMAAKn}}$ signal, for example, the timing may be four times the internal system clock at the shortest.

For the active width of the $\overline{\text{DMAAKn}}$ signal for 2-cycle transfer, refer to **Table 6-4 Minimum Value of Active Width of $\overline{\text{DMAAKn}}$ Signal for 2-Cycle Transfer**.

<1> Mask mode (IFCn6 to IFCn0 bits = 0000000 and DRMKn bit of DIFC register = 1)

This mode is used to establish handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals in synchronization with BUSCLK in the level detection mode. In the single transfer mode, deassert the $\overline{\text{DMARQn}}$ signal within $3 \times \text{BUSCLK}$ after sampling the rising edge of the $\overline{\text{DMAAKn}}$ signal, in order not to start the DMA transfer cycle next to that currently under execution. When the mask mode is set, sampling the DMA request is always masked for the duration of $3 \times \text{BUSCLK}$ period from the rising edge of the $\overline{\text{DMARQn}}$ signal, regardless of the division ratio of BUSCLK (set by the BMC register).

(b) Edge detection mode (IFCn6 to IFCn0 bits = 0000001)

This mode uses the falling edge of the $\overline{\text{DMARQn}}$ signal as a DMA request. It realizes handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals at a speed much lower than that when the mask mode is used. However, note that the falling edge of the $\overline{\text{DMARQn}}$ signal is ignored, even if input, while the $\overline{\text{DMAAKn}}$ signal is active.

For the detailed timing in each mode, refer to **CHAPTER 17 ELECTRICAL SPECIFICATIONS (TARGET VALUES)**.

6.3.8 DMA interface control register (DIFC)

This 8-bit register controls the active width of the $\overline{\text{DMAAKn}}$ signal of each DMA channel and controls the mask function (DMA mask mode) of the $\overline{\text{DMARQn}}$ signal (n = 0 to 3).

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
DIFC	DAKE3	DAKE2	DAKE1	DAKE0	DRMK3	DRMK2	DRMK1	DRMK0	FFFFF8A8H	00H

Bit position	Bit name	Function
7 to 4	DAKE _n	<p>These bits specify the active width of the $\overline{\text{DMAAKn}}$ signal.</p> <p>0: Active width of $\overline{\text{DMAAKn}}$ signal output from DMAC</p> <p>1: Active width of $\overline{\text{DMAAKn}}$ signal + active extension width^{Note} output from DMAC</p> <p>Note The active width is extended by the DAKE_n bit as follows.</p> <ul style="list-style-type: none"> • Four internal system clocks when DAKEBC bit of DTOC register = 0 • Six to seven internal system clocks when DAKEBC bit of DTOC register = 1 <p>For details of the function to extend the active width of the $\overline{\text{DMAAKn}}$ signal, refer to 6.3.8 DMA interface control register (DIFC) or 6.5.1 (2) $\overline{\text{DMAAKn}}$ signal active width extension function.</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. Do not set the DAKE_n bits to 1 during flyby transfer. 2. If either of the following conditions is satisfied when the function to extend the active width of the $\overline{\text{DMAAKn}}$ signal is used, the next $\overline{\text{DMAAKn}}$ signal is asserted while the active width of the preceding $\overline{\text{DMAAKn}}$ signal is extended. As a result, the active cycles of more than one $\overline{\text{DMAAKn}}$ signal combine into one $\overline{\text{DMAAKn}}$ signal, and the number of times the $\overline{\text{DMAAKn}}$ signal is asserted is less than the actual number of DMA cycles. <ul style="list-style-type: none"> • During block transfer or single-step transfer • When the $\overline{\text{DMARQn}}$ signal is kept active (including when the next DMA request is generated while the $\overline{\text{DMAAKn}}$ signal is active or immediately after it has been deasserted in the pending mode of UARTB)
3 to 0	DRMK _n	<p>These are mask bits of the $\overline{\text{DMARQn}}$ signal. They specify the DMA mask mode.</p> <p>0: Do not mask the $\overline{\text{DMARQn}}$ signal.</p> <p>1: Mask input of the $\overline{\text{DMARQn}}$ signal for the duration of 3 bus clocks from the rising edge of the $\overline{\text{DMAAKn}}$ signal.</p> <p>Caution The period masked by the DRMK_n bit is three bus clocks (BUSCLK) from the rising edge of the $\overline{\text{DMAAKn}}$ signal. If the rising edge of the $\overline{\text{DMAAKn}}$ signal is sampled with DRMK_n bit = 1, the DMA controller does not recognize the next DMA transfer request if the $\overline{\text{DMARQn}}$ signal is deasserted within 3 bus clocks (BUSCLK) after the rising edge (high level) of the $\overline{\text{DMAAKn}}$ signal has been sampled with BUSCLK.</p>

Remark n = 0 to 3

6.4 Transfer Modes

6.4.1 Single transfer mode

In single transfer mode, the DMAC releases the bus at each byte/halfword/word 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 always takes precedence. If other DMA transfer request with the lower priority occurs 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 lower priority newly request is executed after the CPU bus has been released.

Figures 6-1 to 6-4 show examples of single transfer.

Figure 6-1. Single Transfer Example 1

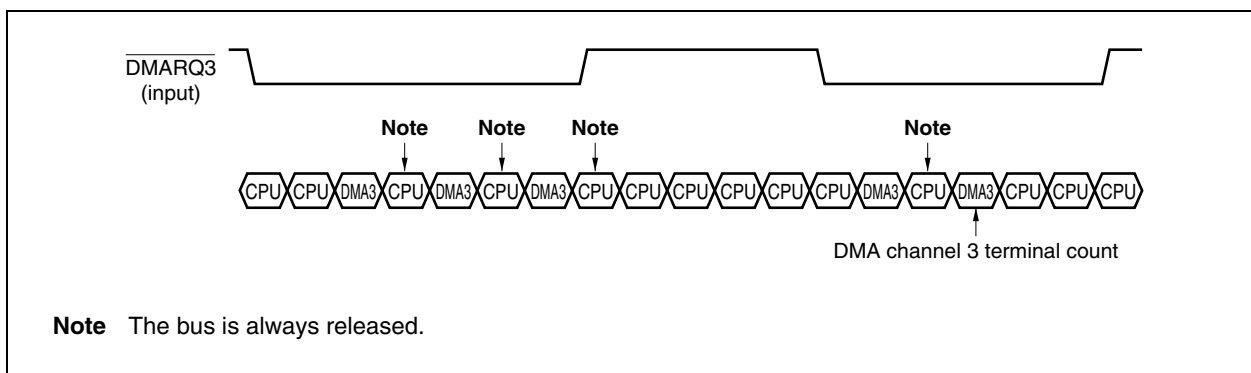


Figure 6-2 shows an example of a single transfer in which a higher priority DMA request is issued. DMA channels 0 to 2 are in the block transfer mode and channel 3 is in the single transfer mode.

Figure 6-2. Single Transfer Example 2

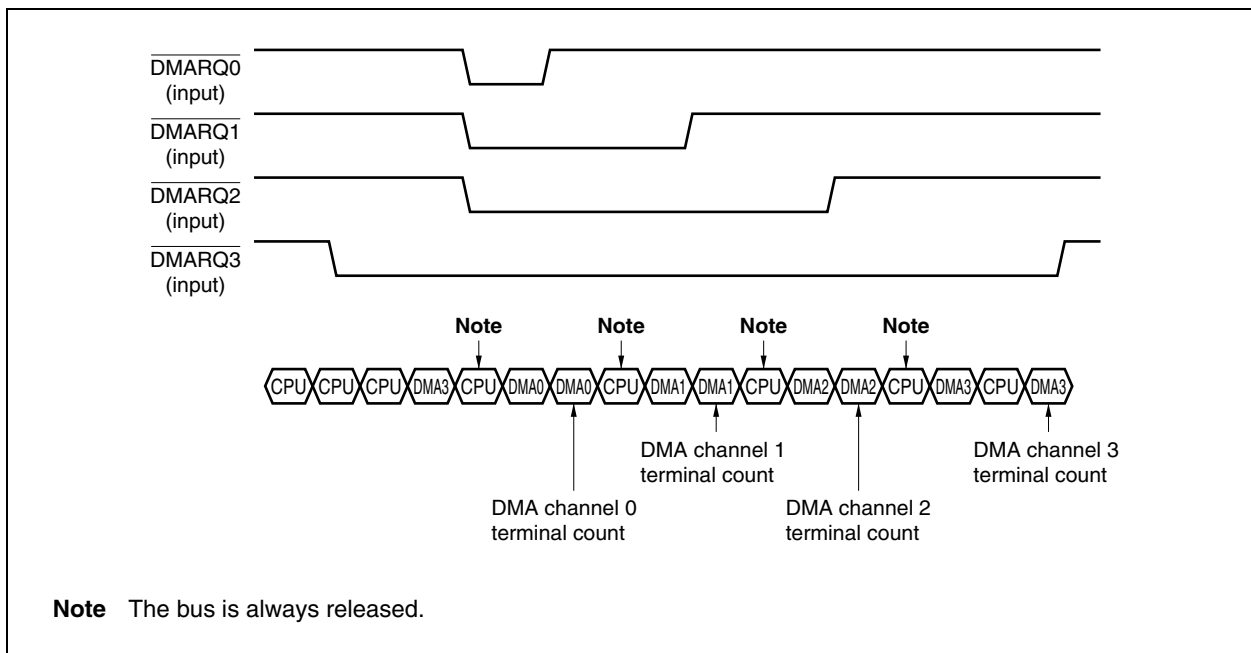


Figure 6-3 is an example of single transfer where a DMA transfer request with the lower priority is issued one clock after single transfer has been completed. DMA channels 0 and 3 are used for single transfer. If two DMA transfer request signals become active at the same time, two DMA transfer operations are alternately executed.

Figure 6-3. Single Transfer Example 3

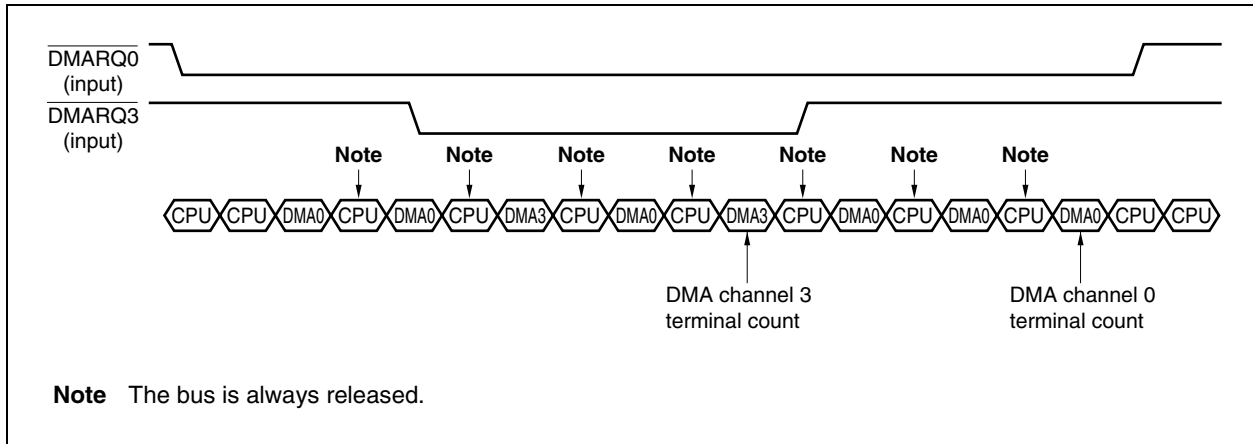
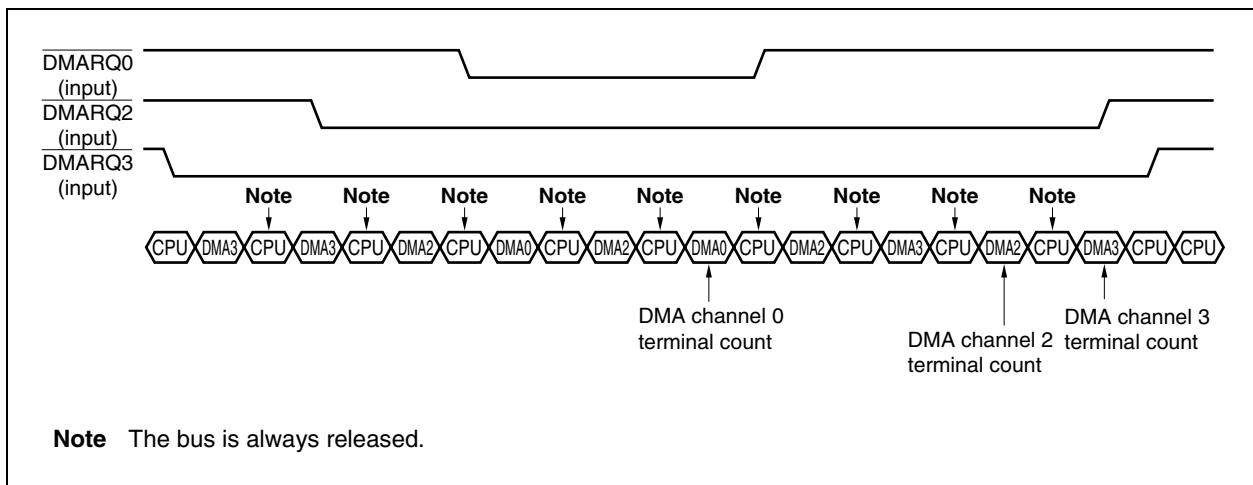


Figure 6-4 is an example of single transfer where two or more DMA transfer requests with the lower priority are issued one clock after single transfer has been completed. DMA channels 0, 2, and 3 are used for single transfer. If three or more DMA transfer request signals become active at the same time, two DMA transfer operations are alternately executed, always starting from the one with the highest priority.

Figure 6-4. Single Transfer Example 4



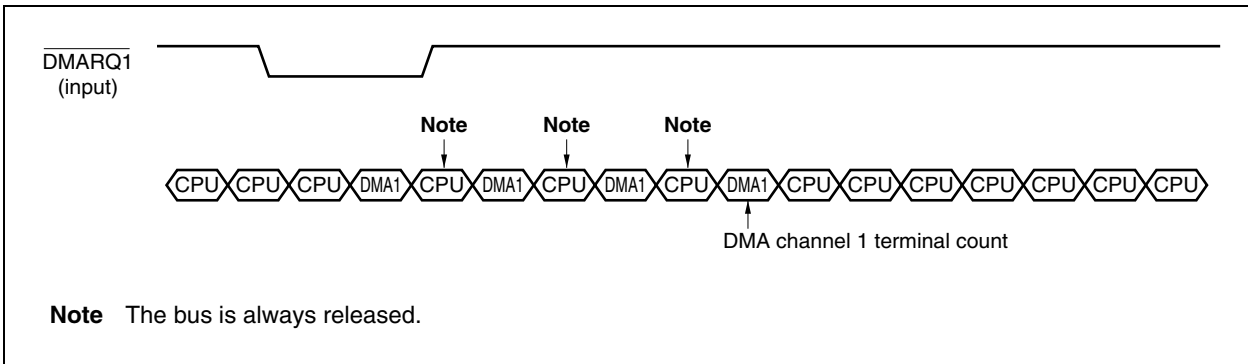
6.4.2 Single-step transfer mode

In single-step transfer mode, the DMAC releases the bus at each byte/halfword/word transfer. If there is a subsequent DMA transfer request signal ($\overline{\text{DMARQ0}}$ to $\overline{\text{DMARQ3}}$), 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.

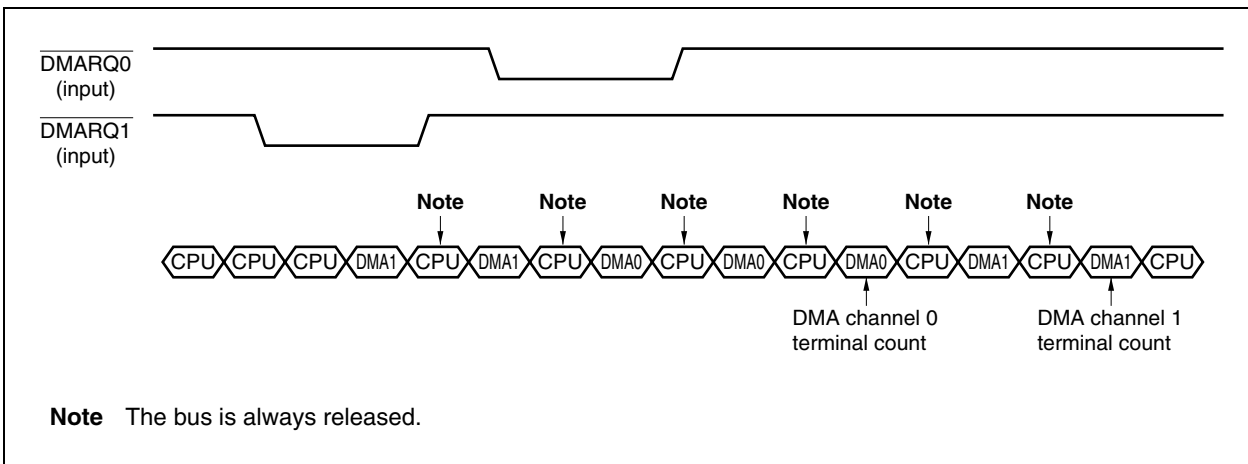
The following shows an example of a single-step transfer. Figure 6-6 shows an example of single-step transfer made in which a higher priority DMA request is issued. DMA channels 0 and 1 are in the single-step transfer mode.

Figure 6-5. Single-Step Transfer Example 1



Note The bus is always released.

Figure 6-6. Single-Step Transfer Example 2



Note The bus is always released.

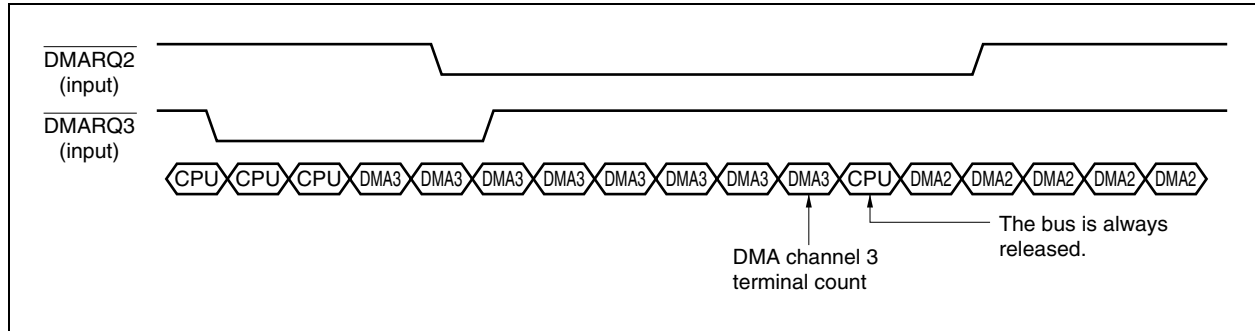
6.4.3 Block transfer mode

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 2 and 3 are in the block transfer mode.

Figure 6-7. Block Transfer Example



6.5 Transfer Types

6.5.1 2-cycle transfer

In 2-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 to 2 clocks is always inserted between a read cycle and a write cycle.

Figure 6-8. Timing of 2-Cycle DMA Transfer (SRAM → External I/O) (1/2)

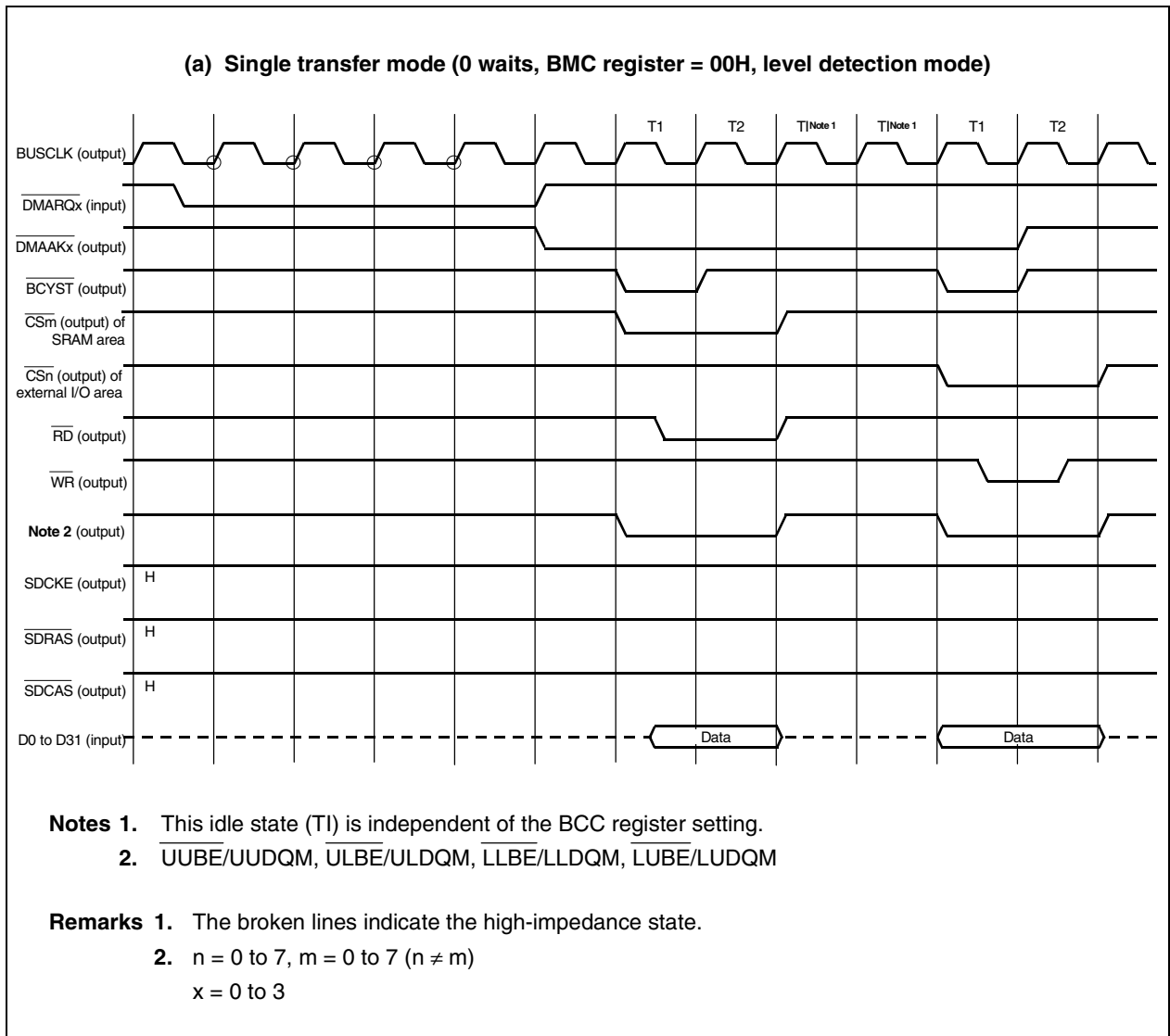
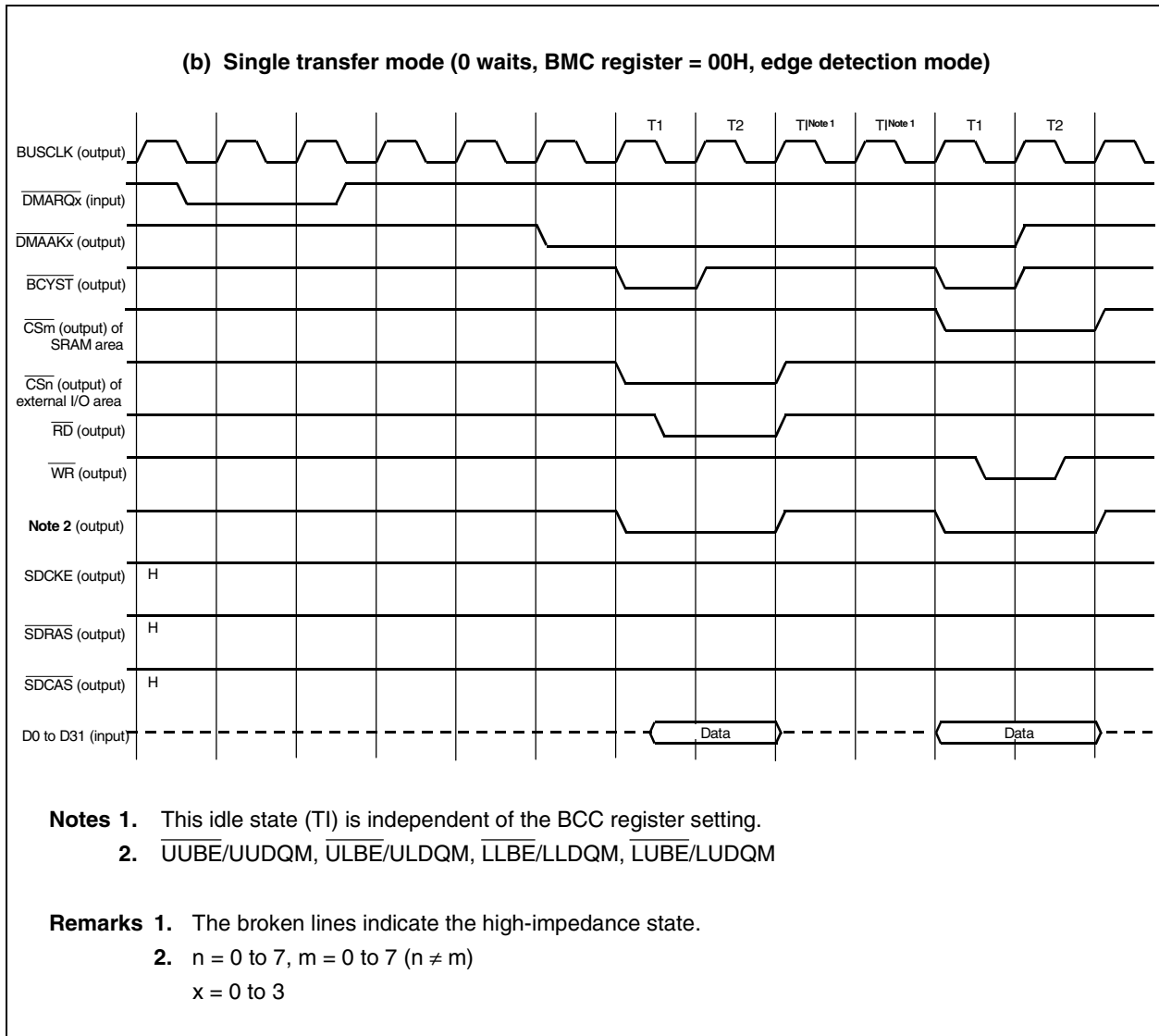
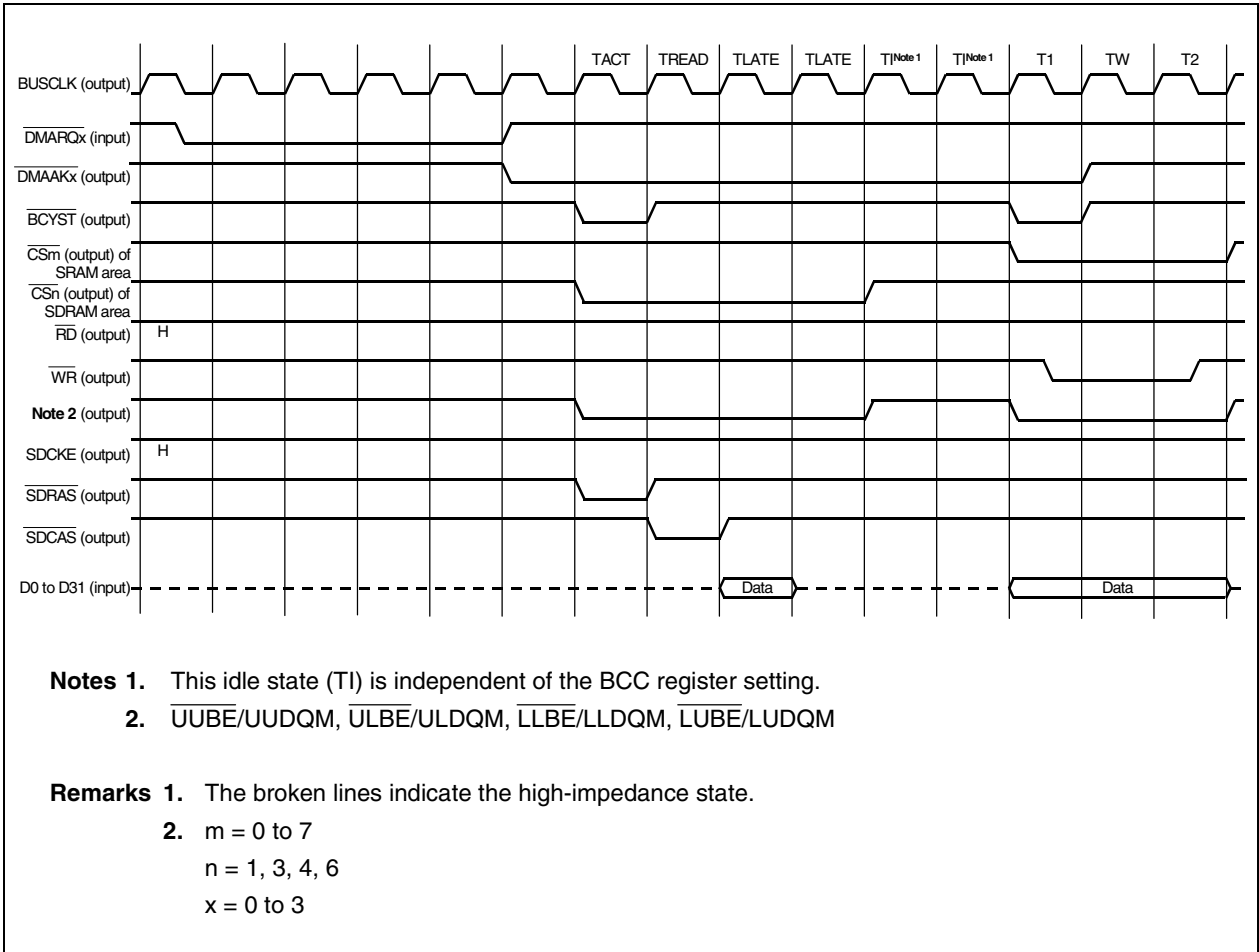


Figure 6-8. Timing of 2-Cycle DMA Transfer (SRAM → External I/O) (2/2)



**Figure 6-9. Timing of 2-Cycle DMA Transfer (SDRAM → SRAM):
Single Transfer Mode (SRAM Data 1 Wait, SDRAM Latency = 2,
BMC Register = 00H, Level Detection Mode)**



**Figure 6-10. Timing of 2-Cycle DMA Transfer (SRAM → SDRAM):
Single Transfer Mode (SRAM Data 1 Wait, BMC Register = 00H,
Edge Detection Mode)**

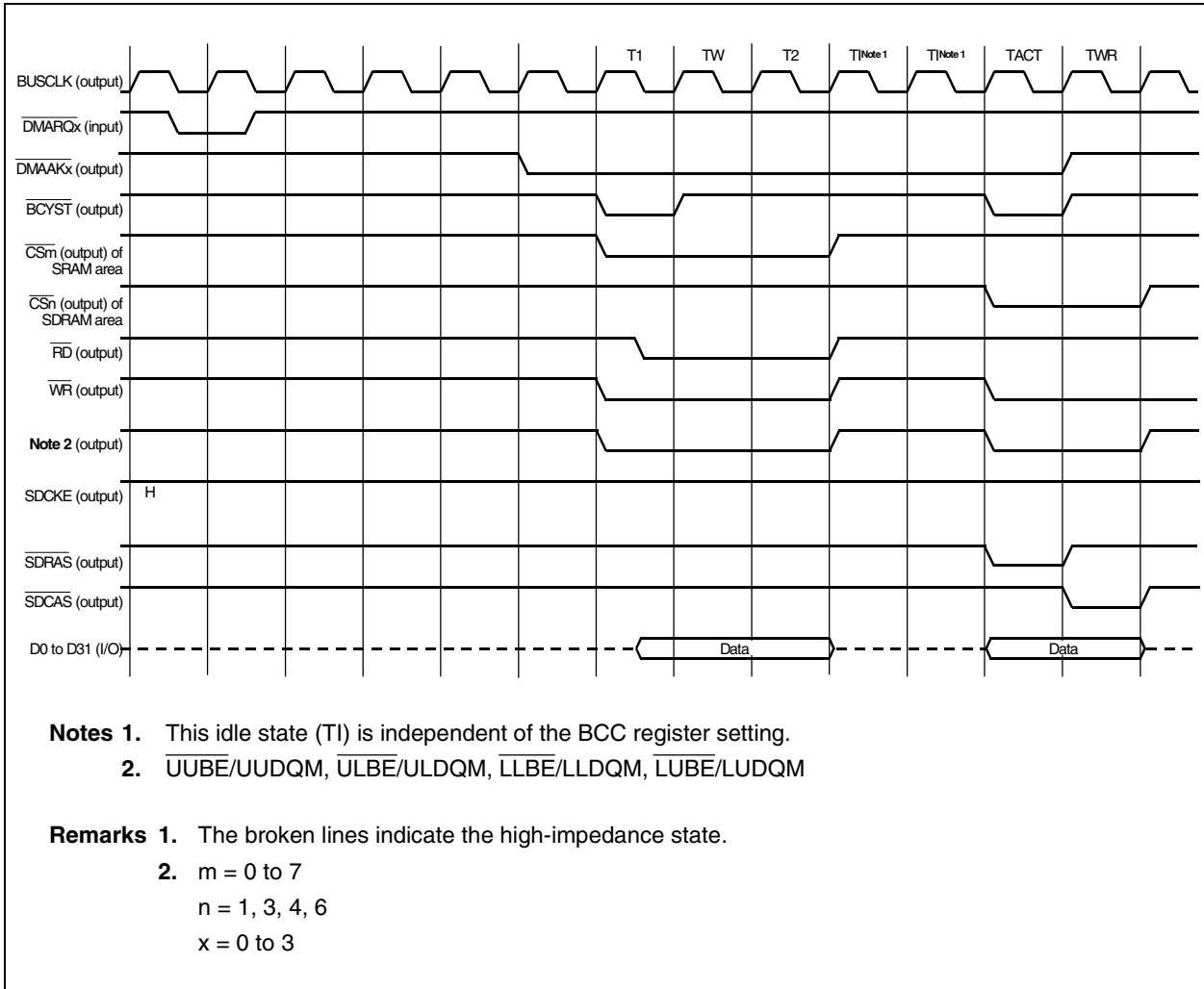


Figure 6-11. Timing of 2-Cycle DMA Transfer (SRAM → External I/O): Without Speculative Read

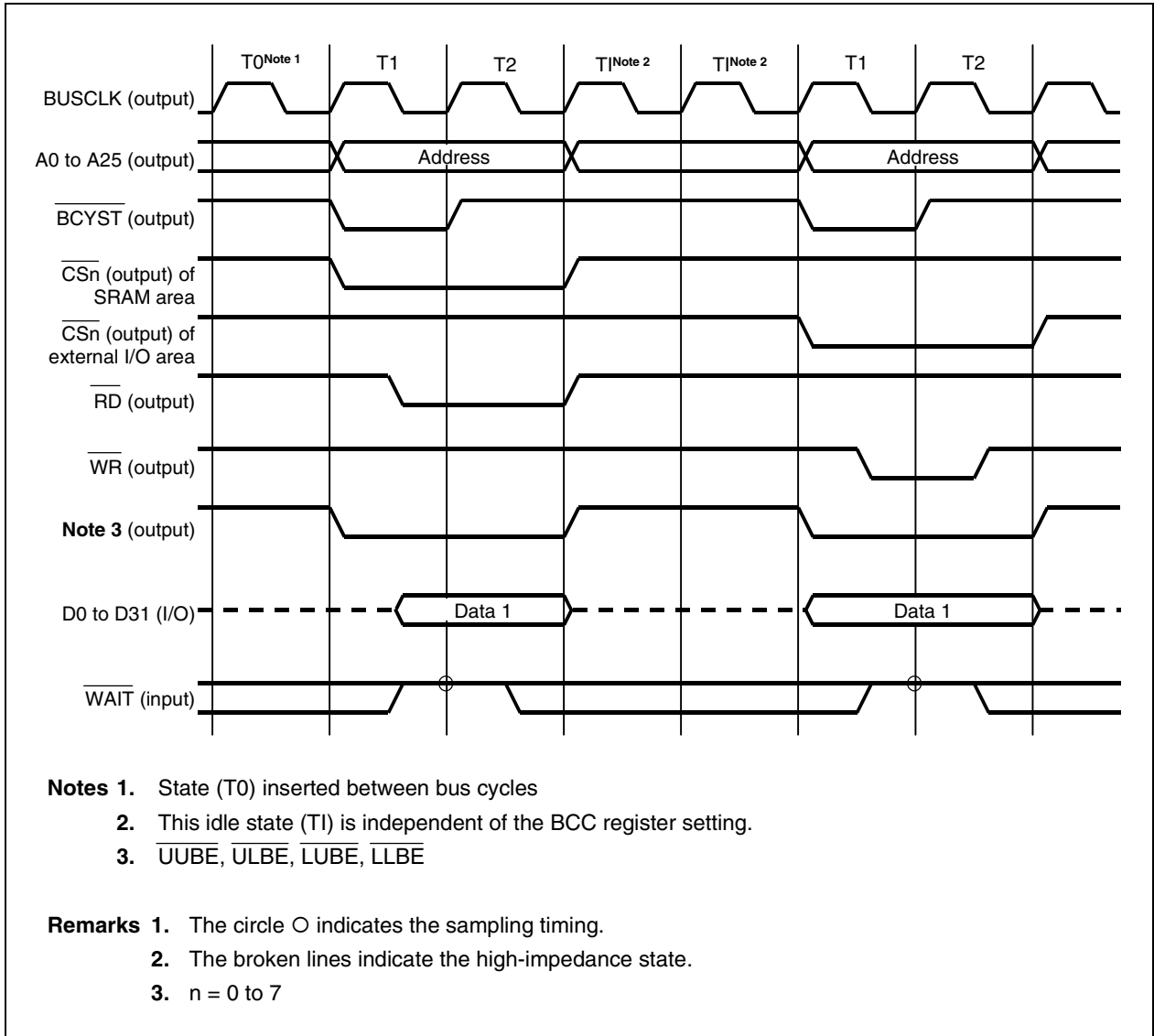


Figure 6-12. Timing of 2-Cycle DMA Transfer (SDRAM → SRAM) (1/2)

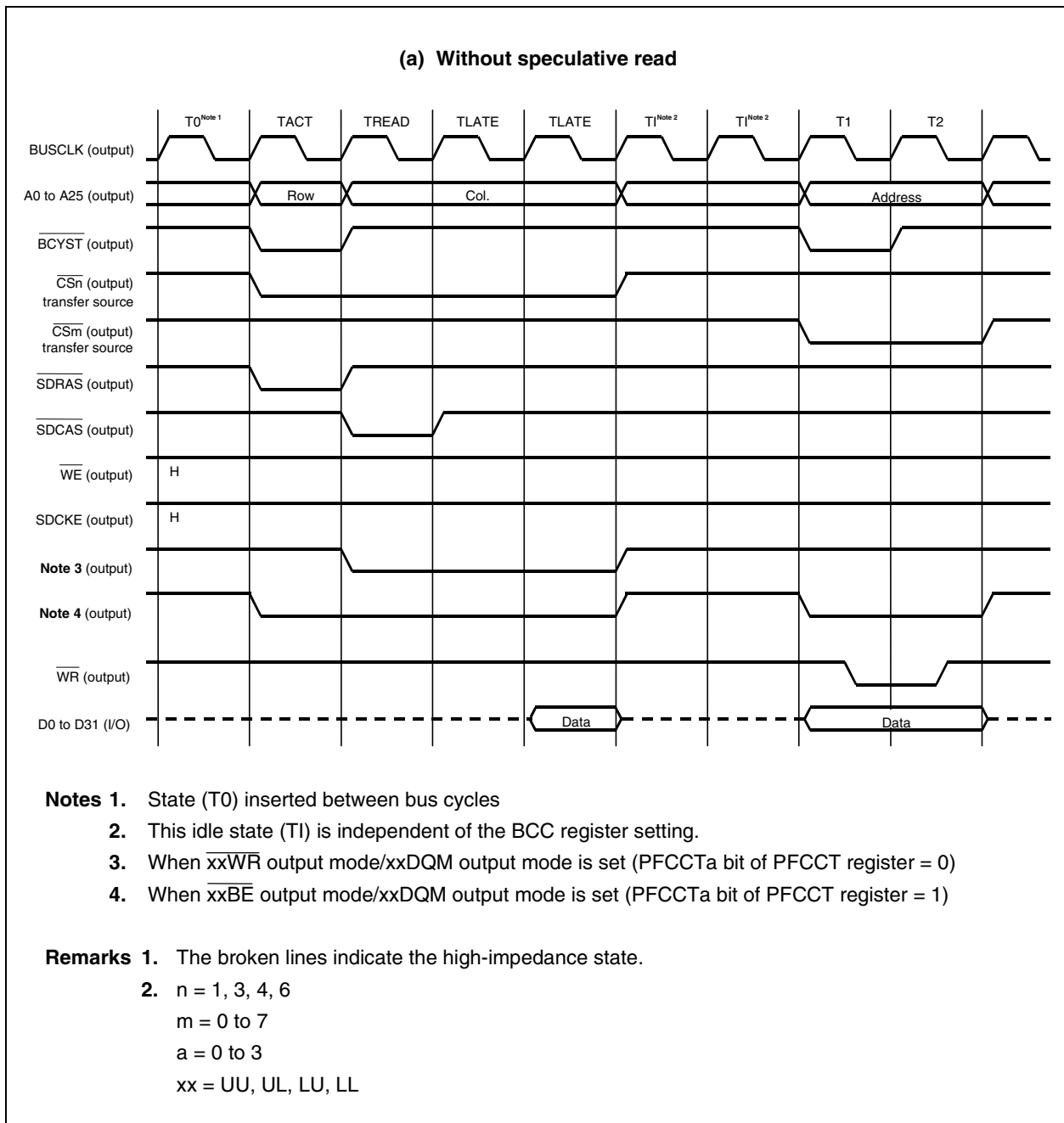
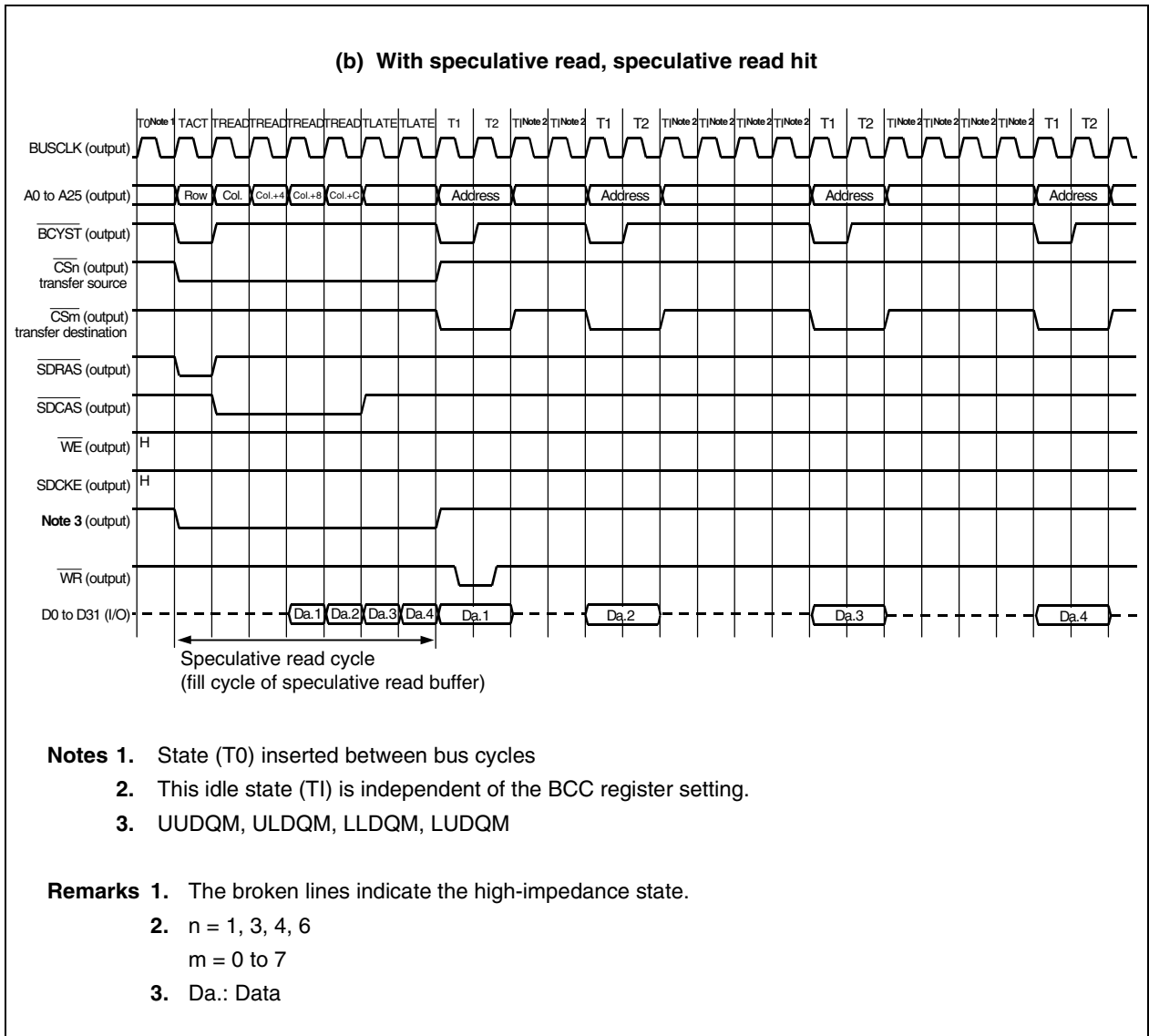


Figure 6-12. Timing of 2-Cycle DMA Transfer (SDRAM → SRAM) (2/2)



★ (1) Timing of $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals for 2-cycle transfer

The targets of 2-cycle transfer are as follows.

Table 6-1. Targets of 2-Cycle Transfer

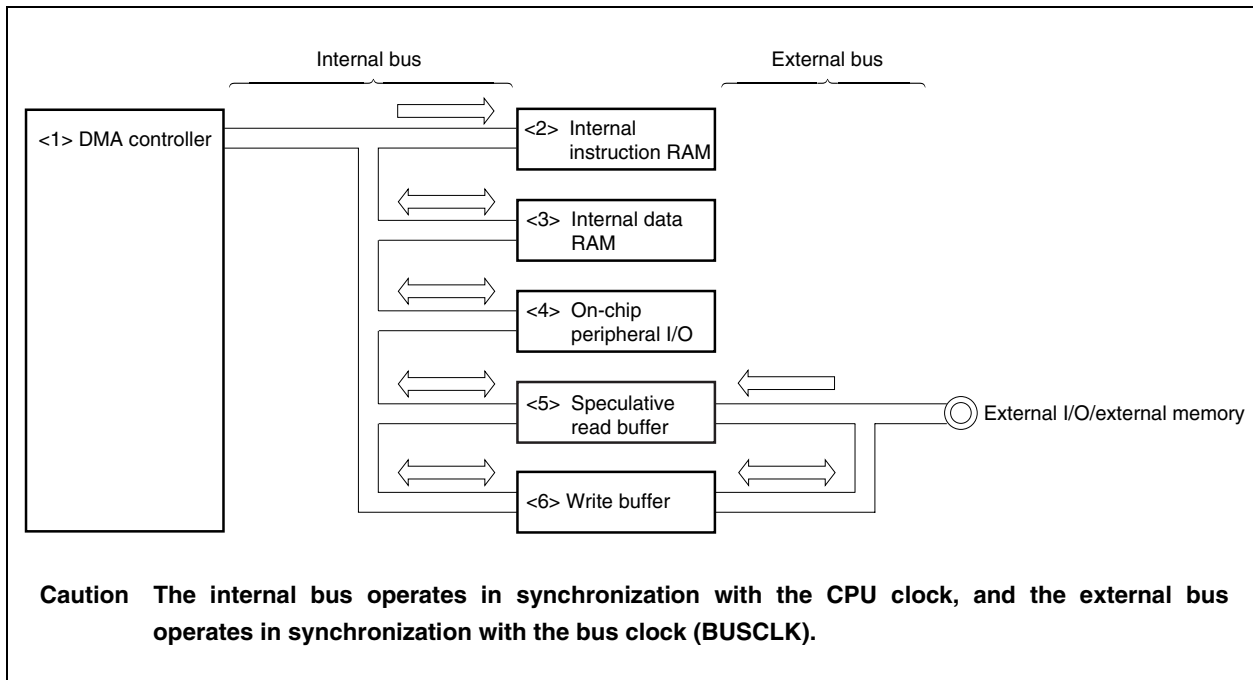
		Transfer Destination			
		External I/O/External Memory	On-Chip Peripheral I/O	Internal Data RAM	Internal Instruction RAM
Transfer source	External I/O/external memory	√	√	√	√
	On-chip peripheral I/O	√	√	√	√
	Internal data RAM	√	√	×	√

Remarks 1. √: Can be transferred, ×: Cannot be transferred

2. For details, refer to Table 6-4 Minimum Value of Active Width of $\overline{\text{DMAAKn}}$ Signal for 2-Cycle Transfer.

The bus configuration is as shown below.

Figure 6-13. Bus Configuration



The active width of the $\overline{\text{DMAAKn}}$ signal output by the DMA controller is from the beginning of a read operation by the DMA controller to the end of a write operation ($n = 0$ to 3). The $\overline{\text{DMAAKn}}$ signal is not asserted in a cycle used to write/read the internal data RAM.

The active width of the $\overline{\text{DMAAKn}}$ signal for each DMA transfer is shown below.

Table 6-2. Active Width of $\overline{\text{DMAAKn}}$ Signal for 2-Cycle Transfer

		Transfer Destination			
		External I/O/External Memory	On-Chip Peripheral I/O	Internal Data RAM	Internal Instruction RAM
Transfer source	External I/O/external memory	<5> (<7>) → <1> → <6>	<5> (<7>) → <1> → <4>	<5> (<7>) → <1>	<5> (<7>) → <1> → <2>
	On-chip peripheral I/O	<4> → <1> → <6>	<4> → <1> → <4>	<4> → <1>	<4> → <1> → <2>
	Internal data RAM	<1> → <6>	<1> → <4>	–	<1> → <2>

Remark (): Cycle without speculative reading or in case of speculative read miss-hit

Therefore, the correlation between external access during 2-cycle transfer (execution to external I/O/external memory) and the active width of the $\overline{\text{DMAAKn}}$ signal is as follows.

Table 6-3. Correlation Between External Access (Execution to External I/O/External Memory) and Active Width of $\overline{\text{DMAAKn}}$ Signal

		Speculative Read Function	Transfer Destination			
			External I/O/ External Memory	On-Chip Peripheral I/O	Internal Data RAM	Internal Instruction RAM
Transfer source	External I/O/external memory	None	√	√	√	√
	On-chip peripheral I/O		× (a)	–	–	–
	Internal data RAM		× (a)	–	–	–
	External I/O/external memory	Provided	× (b)	× (c)	× (c)	× (c)
	On-chip peripheral I/O		× (a)	–	–	–
	Internal data RAM		× (a)	–	–	–

Remarks √: The $\overline{\text{DMAAKn}}$ signal is asserted while DMA access to/from external I/O/external memory is being executed.

×: The $\overline{\text{DMAAKn}}$ signal may not be asserted while DMA access to/from external I/O/external memory is being executed.

(a): Handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals may be completed before a write operation to the external I/O or external memory is executed.

As soon as data has been transferred to the write buffer (<4> → <1> → <6> or <1> → <6>), assertion of the $\overline{\text{DMAAKn}}$ signal ends. If data of three buffers has already been stored in the write buffer separately, data transferred by DMA is stored in the fourth buffer. Subsequently, when a write operation of the stored data has taken place three times, transfer to the external I/O or external memory targeted by the last $\overline{\text{DMAAKn}}$ signal is executed (<6> → <7>).

(b): Handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals may be completed if a read operation from the external I/O or external memory (DMA access) has been completed before handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals is completed, or before a write operation to the external I/O or external memory is executed.

The active state of the $\overline{\text{DMAAKn}}$ signal ends as soon as data has been transferred to the write buffer (<5> → <1> → <6>). If speculative reading hits, data is read from the external I/O or external memory when speculative reading has been completed. Therefore, a read operation from the external I/O or external memory (<7> → <5>) is executed before the $\overline{\text{DMAAKn}}$ signal. While data is being transferred from the speculative read buffer to the write buffer (<5> → <1> → <6>) after that, the $\overline{\text{DMAAKn}}$ signal is asserted. For example, if data of three buffers has already been stored in the write buffer, data transferred by DMA is stored in the fourth buffer. After that, transfer to the external I/O or external memory (<6> → <7>) targeted by the last $\overline{\text{DMAAKn}}$ signal is executed when a write operation of the stored data has taken place three times.

(c): A read operation from the external I/O or external memory (DMA access) may be completed before handshaking of the $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals is completed.

If speculative reading hits, because the data has already been transferred from the external I/O or external memory to the speculative read buffer (<7> → <5>), execution to the external I/O or external memory (DMA access) will have already been completed while the $\overline{\text{DMAAKn}}$ signal was asserted (<5> → <1> → (<2>, <3> or <4>)).

★ (2) **$\overline{\text{DMAAKn}}$ signal active width extension function**

The $\overline{\text{DMAAKn}}$ signal is output in synchronization with the internal bus cycle during 2-cycle transfer, and is not synchronized with the external bus cycle ($n = 0$ to 3). The DMA cycle differs depending on the configuration subject to DMA transfer (refer to **6.5.1 (1) Timing of $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals for 2-cycle transfer**). Depending on the target of DMA transfer, the configuration may allow the $\overline{\text{DMAAKn}}$ signal to be asserted only for the duration of two internal system clocks. In this case, assertion of the $\overline{\text{DMAAKn}}$ signal may not be sampled with BUSCLK if the internal system clock is divided and the bus clock (BUSCLK) is used (e.g., BMC register = 02H: internal system clock divided by three).

To sample assertion of the $\overline{\text{DMAAKn}}$ signal with BUSCLK, extend the active width of the $\overline{\text{DMAAKn}}$ signal by using the DAKEBC bit of the DTOC register and DAKEn bit of the DIFC register.

The minimum value of the active width of the $\overline{\text{DMAAKn}}$ during 2-cycle transfer is shown in the table below.

Table 6-4. Minimum Value of Active Width of $\overline{\text{DMAAKn}}$ Signal During 2-Cycle Transfer

		Speculative Read Function	Transfer Destination			
			External I/O/ External Memory	On-Chip Peripheral I/O	Internal Data RAM	Internal Instruction RAM
Transfer source	External I/O/external memory	None	Read cycle + 3 internal system clocks	Read cycle + (4+i) internal system clocks	Read cycle	Read cycle + internal instruction RAM write cycle + 1 internal system clock
	On-chip peripheral I/O		(6+i) internal system clocks	(7+2i) internal system clocks	(3+i) internal system clocks	Internal instruction RAM write cycle + (4+i) internal system clocks
	Internal data RAM		2 internal system clocks	(3+i) internal system clocks	–	Internal instruction RAM write cycle
	External I/O/external memory	Provided	5 internal system clocks	(6+i) internal system clocks	2 internal system clocks	Internal instruction RAM write cycle + 3 internal system clocks
	On-chip peripheral I/O		(6+i) internal system clocks	(7+2i) internal system clocks	(3+i) internal system clocks	Internal instruction RAM write cycle + (4+i) internal system clocks
	Internal data RAM		2 internal system clocks	(3+i) internal system clocks	–	Internal instruction RAM write cycle

Caution The function to extend the active width of the $\overline{\text{DMAAKn}}$ signal can be used only during 2-cycle transfer ($n = 0$ to 3). It cannot be used during flyby transfer. The operation is not guaranteed if it is used during flyby transfer. During flyby transfer, the $\overline{\text{DMAAKn}}$ signal synchronized with the bus cycle is output.

Remark i: Number of wait cycles set by VSWC register

★ (3) Outline of 2-cycle transfer timing

The timing of 2-cycle transfer is outlined below.

Figure 6-14. Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 1 (0 SRAM Waits)

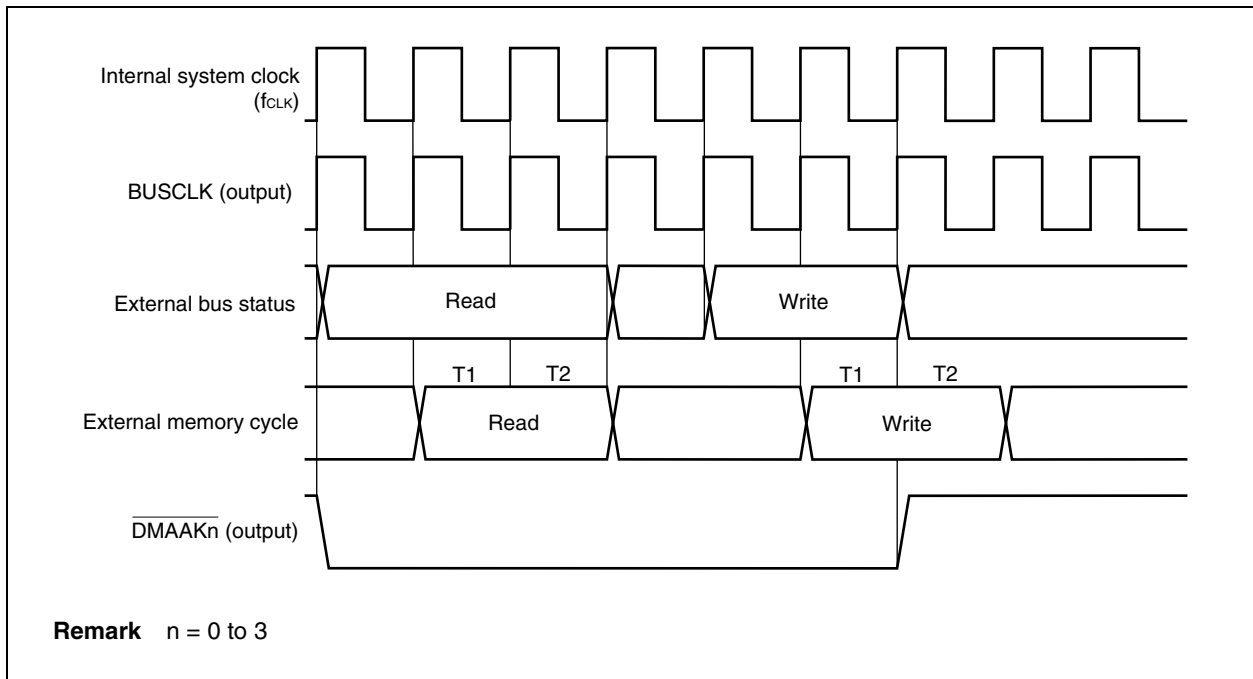


Figure 6-15. Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 2 (0 SRAM Waits)

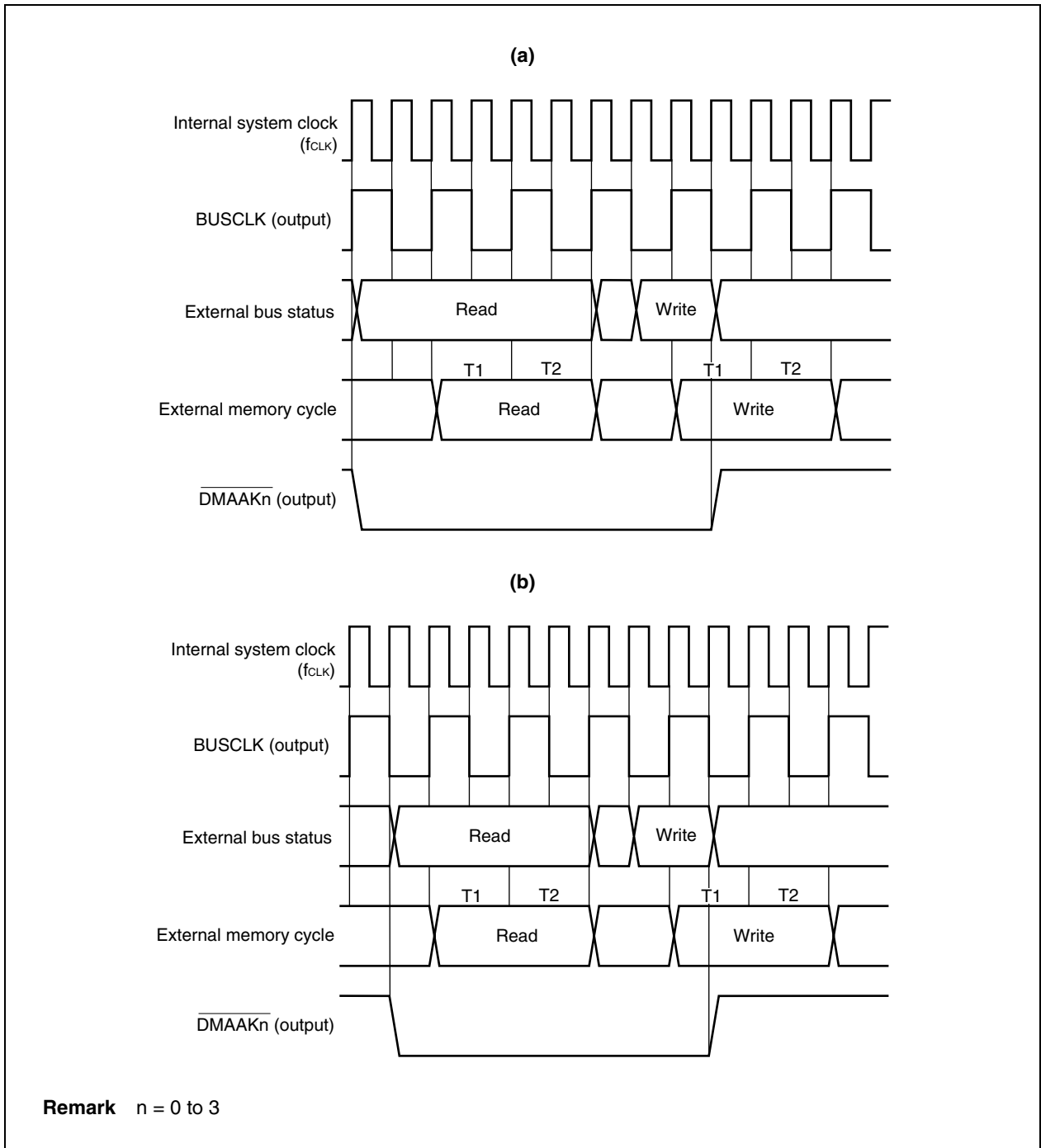


Figure 6-16. Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 3 (0 SRAM Waits) (1/2)

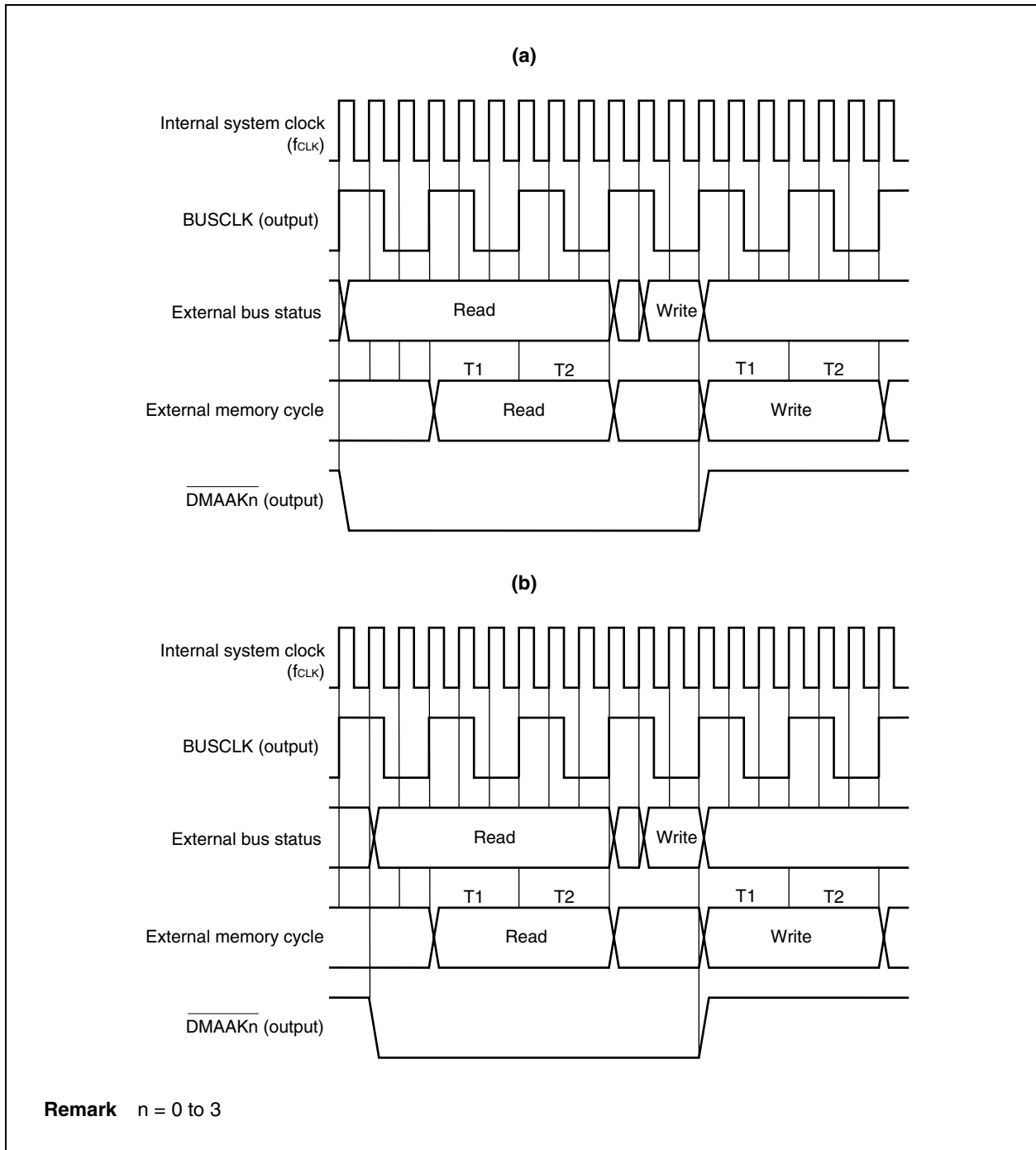


Figure 6-16. Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 3 (0 SRAM Waits) (2/2)

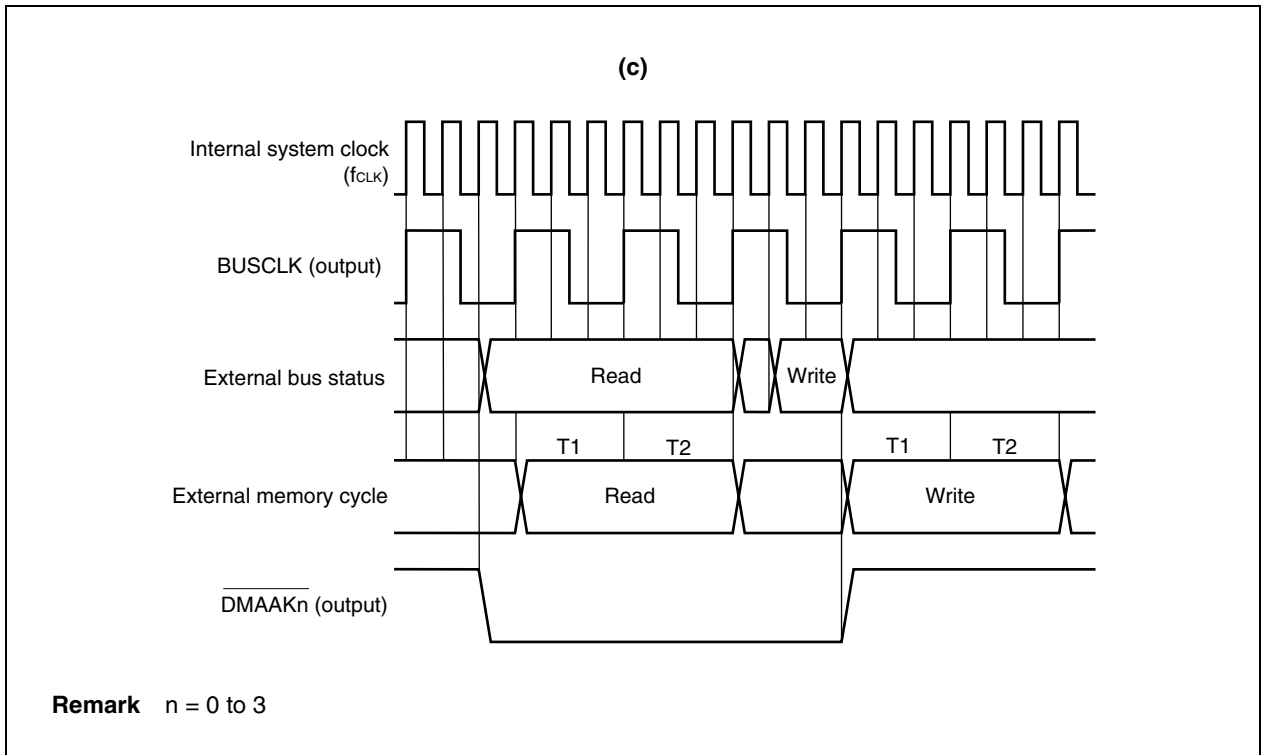
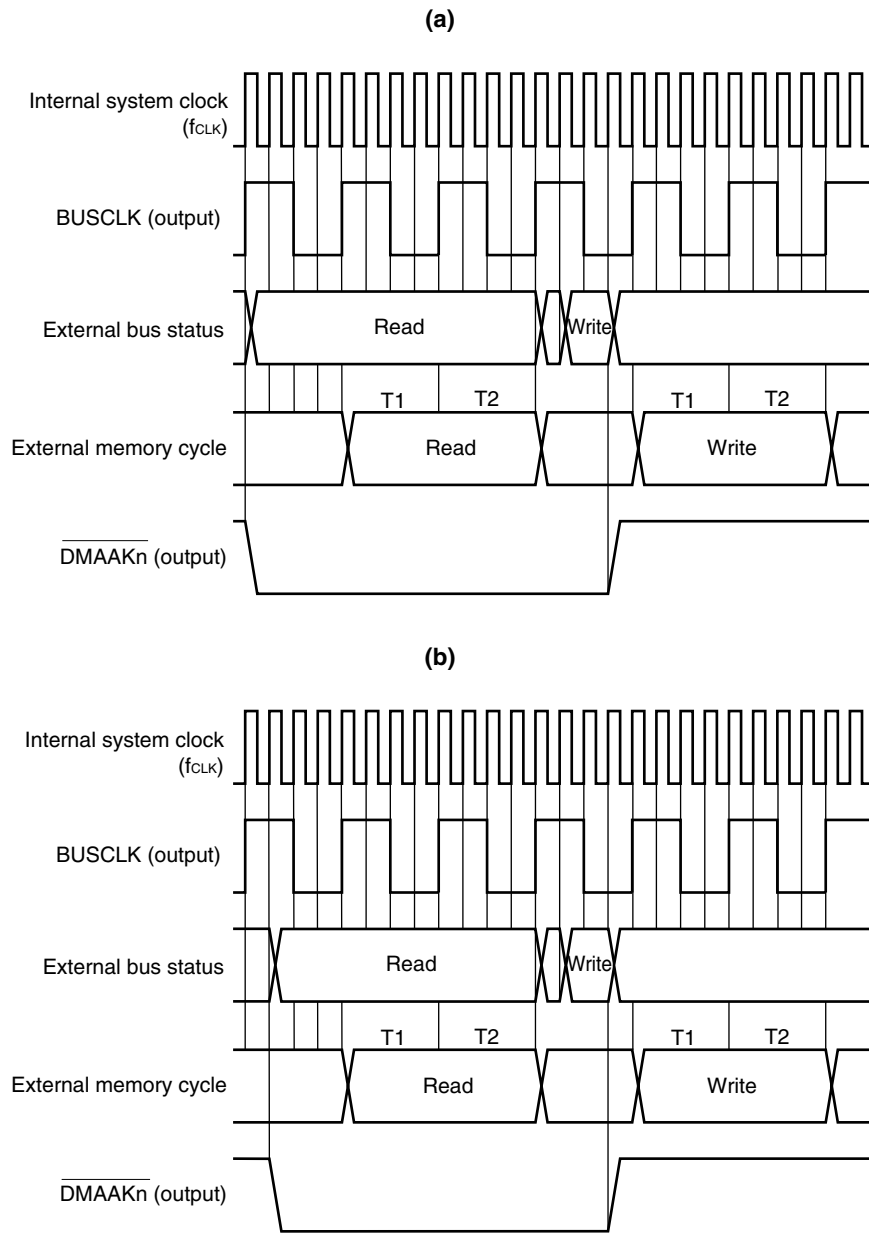
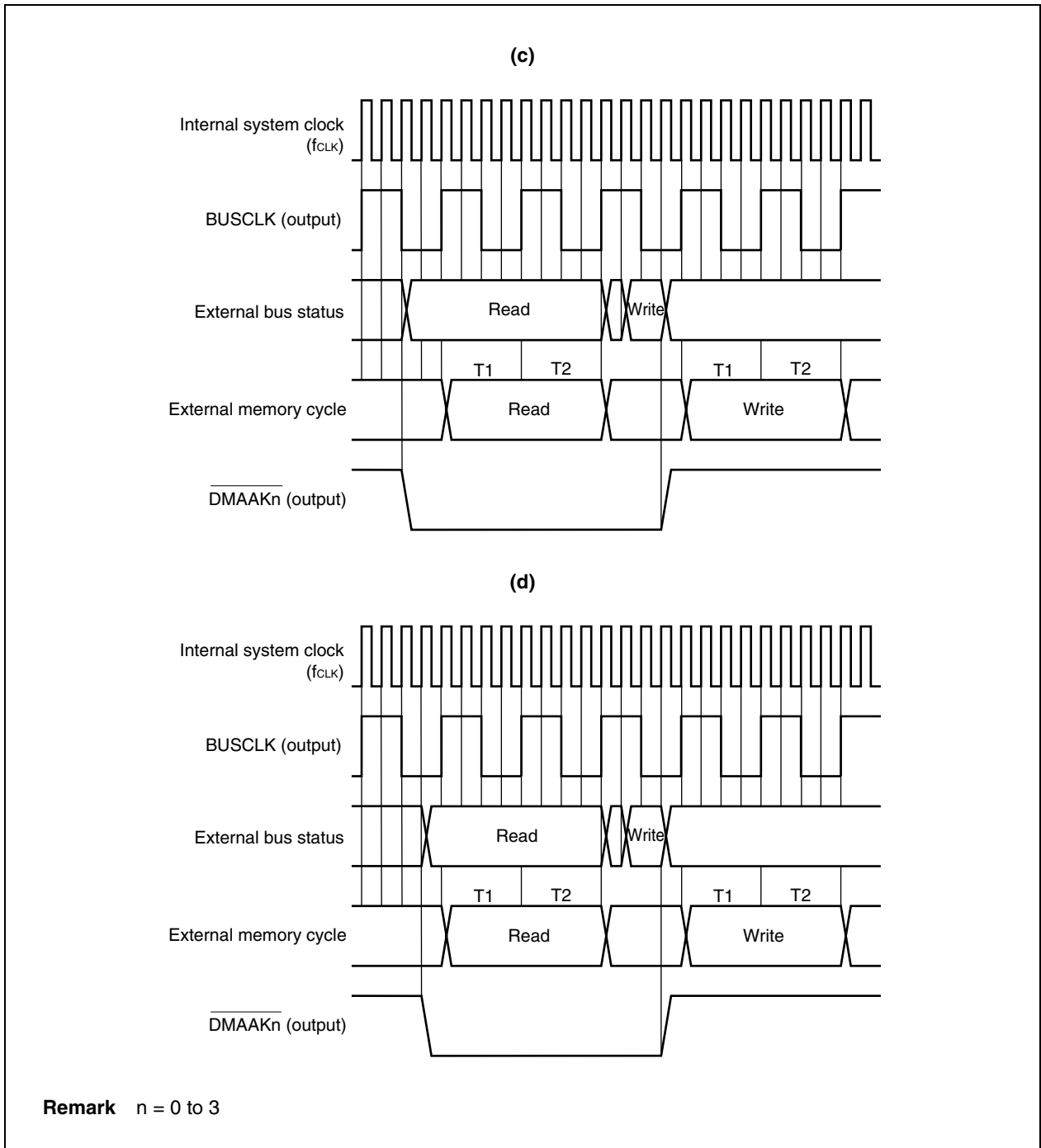


Figure 6-17. Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 4 (0 SRAM Waits) (1/2)



Remark n = 0 to 3

Figure 6-17. Outline of Timing During 2-Cycle Transfer (SRAM → SRAM): Divided by 4 (0 SRAM Waits) (2/2)



6.5.2 Flyby transfer

Since data is transferred in 1 cycle during a flyby transfer, a memory address is always output irrespective whether it is a source address or a destination address, and read/write signals of the memory and peripheral I/O become active at the same time. Therefore, the external I/O is selected by the $\overline{\text{DMAAK0}}$ to $\overline{\text{DMAAK3}}$ signals.

To perform a normal access to the external I/O by means other than DMA transfer, externally AND the $\overline{\text{CSm}}$ and $\overline{\text{DMAAKx}}$ signals ($m = 0$ to 7 , $x = 0$ to 3), and connect the resultant signal to the chip select signal of the external I/O. A circuit example of a normal access, other than DMA transfer, to external I/O is shown below.

Figure 6-18. Circuit Example When Flyby Transfer Is Performed Between External I/O and SRAM

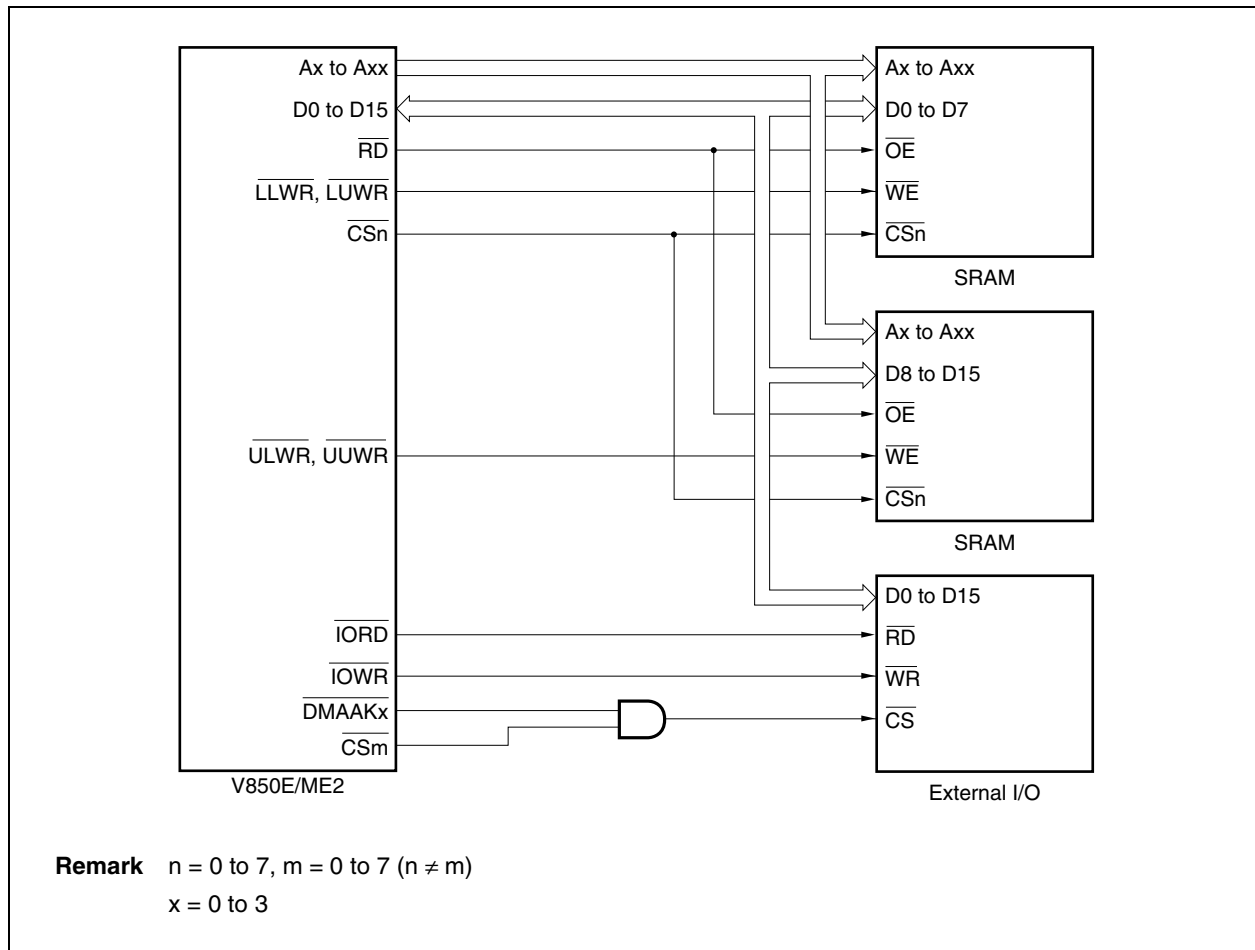


Figure 6-19. Timing of DMA Flyby Transfer (External I/O → SDRAM) (1/3)

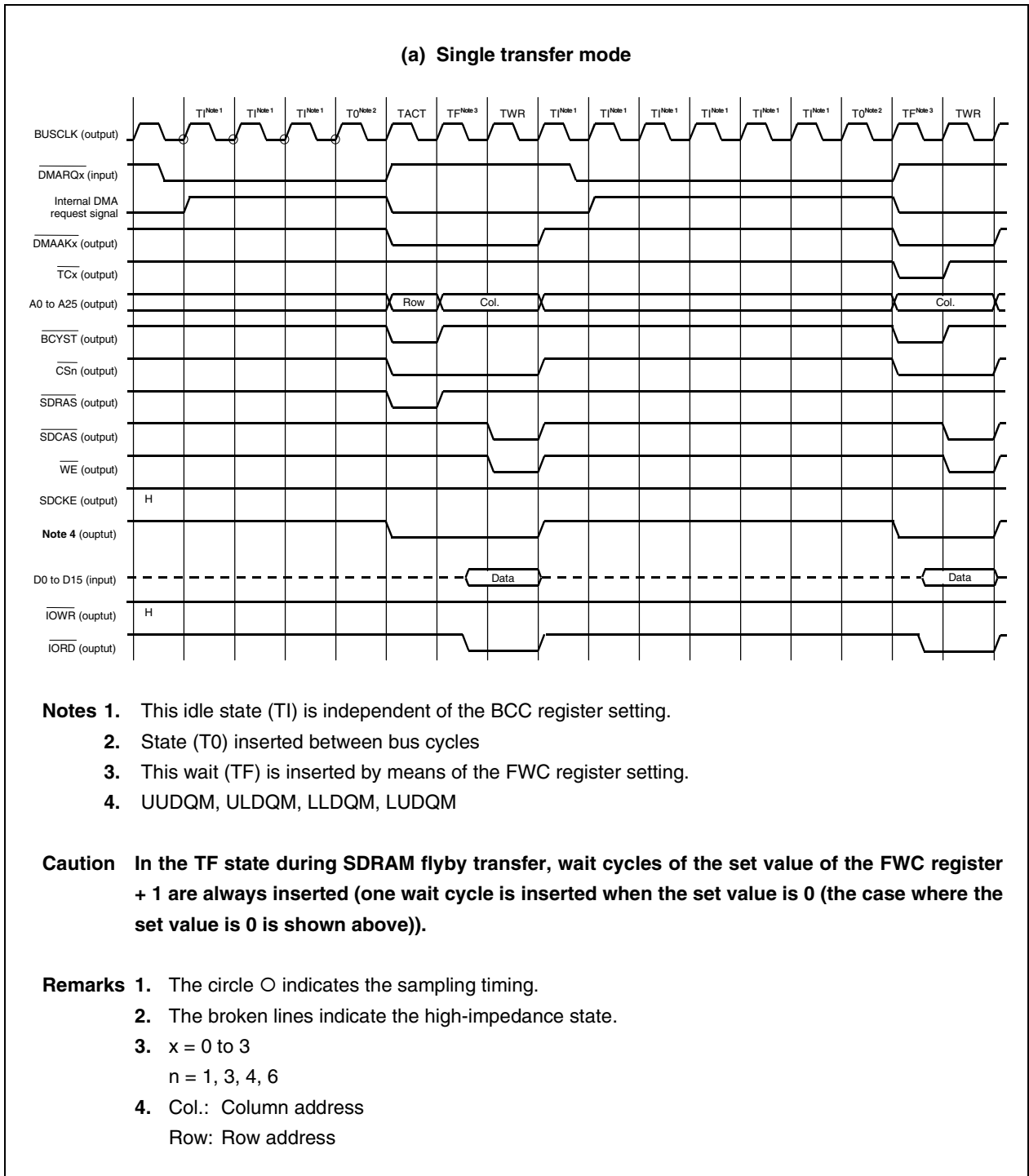


Figure 6-19. Timing of DMA Flyby Transfer (External I/O → SDRAM) (2/3)

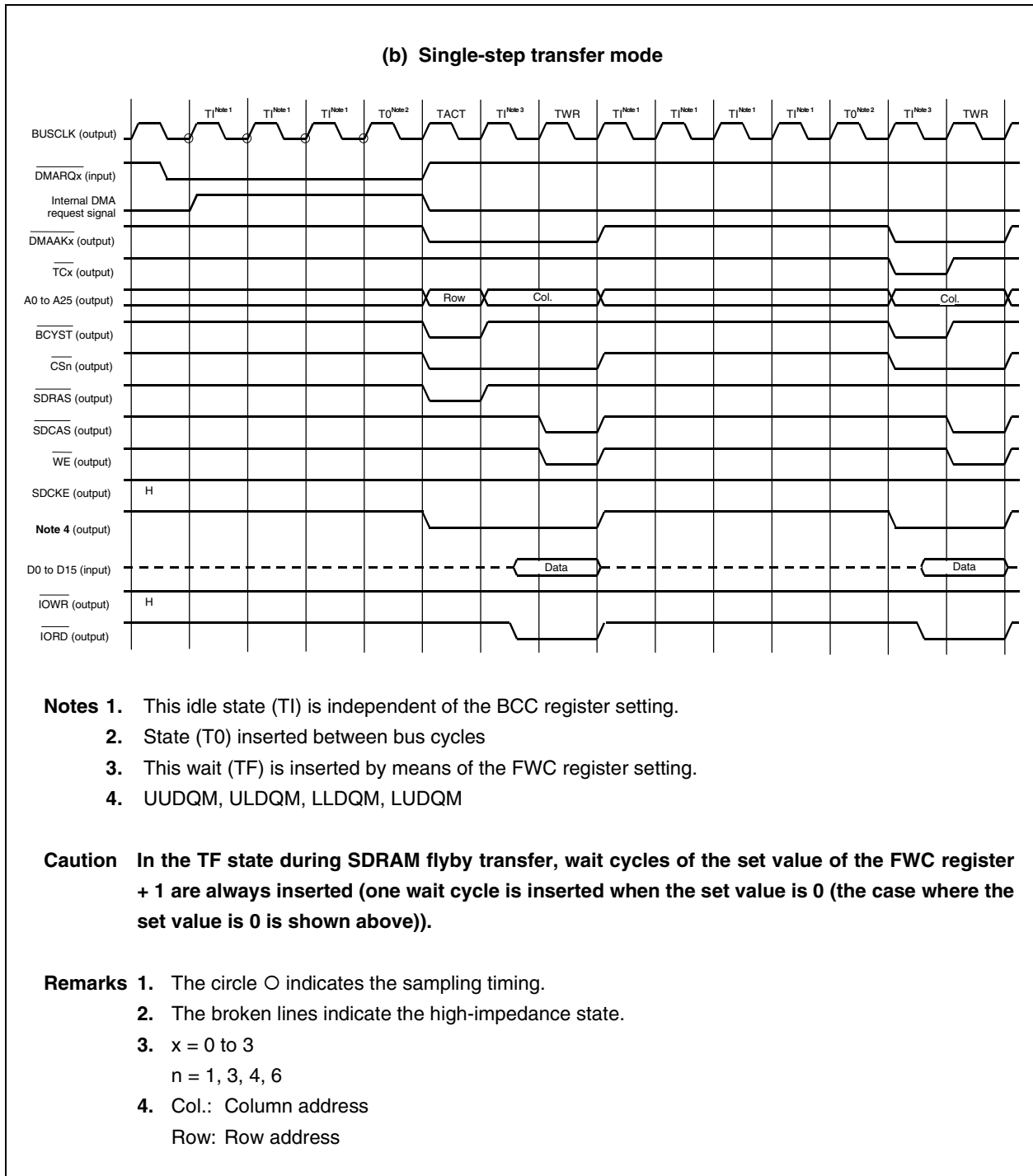


Figure 6-19. Timing of DMA Flyby Transfer (External I/O → SDRAM) (3/3)

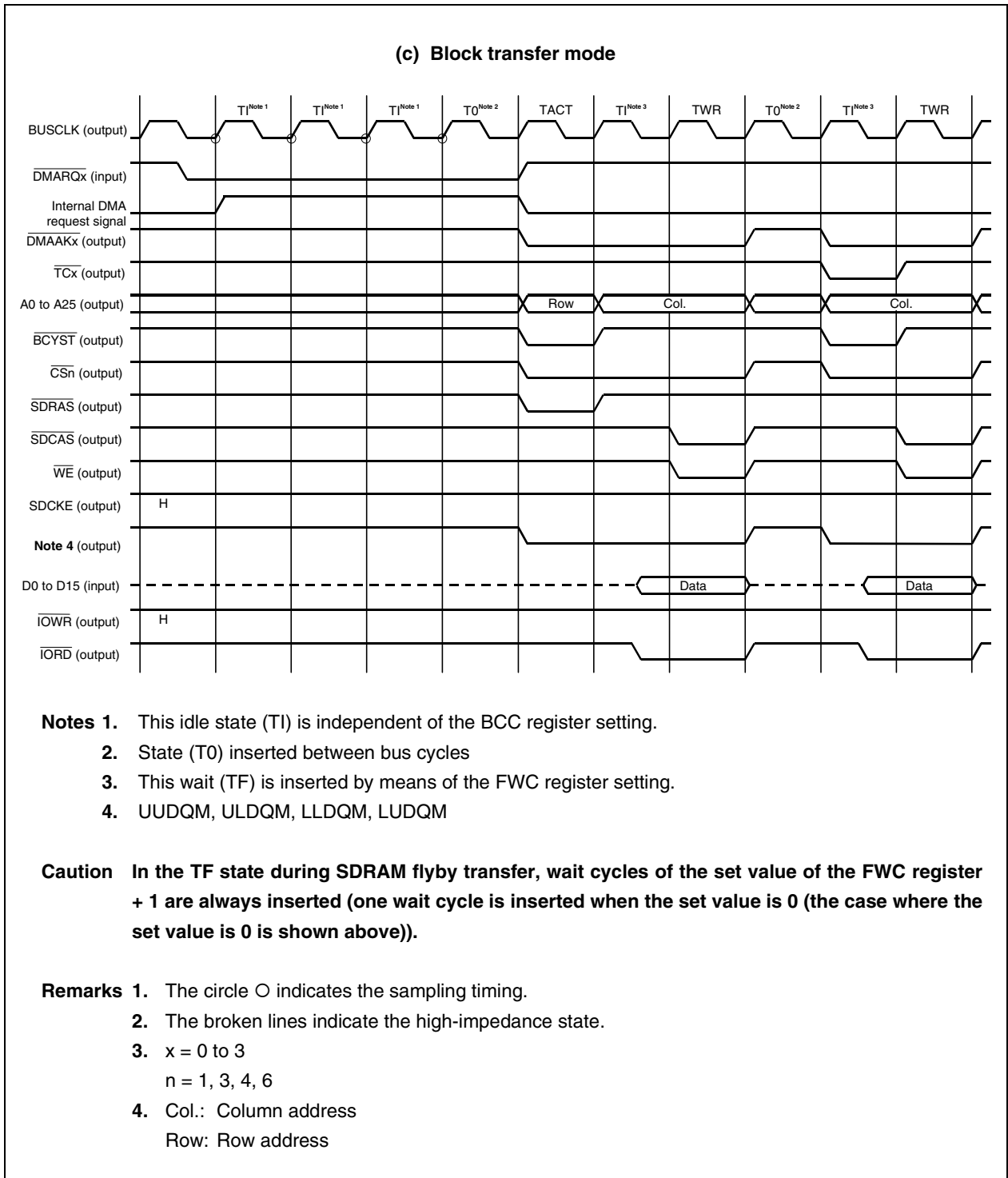


Figure 6-20. Timing of DMA Flyby Transfer (SRAM → External I/O) (1/2)

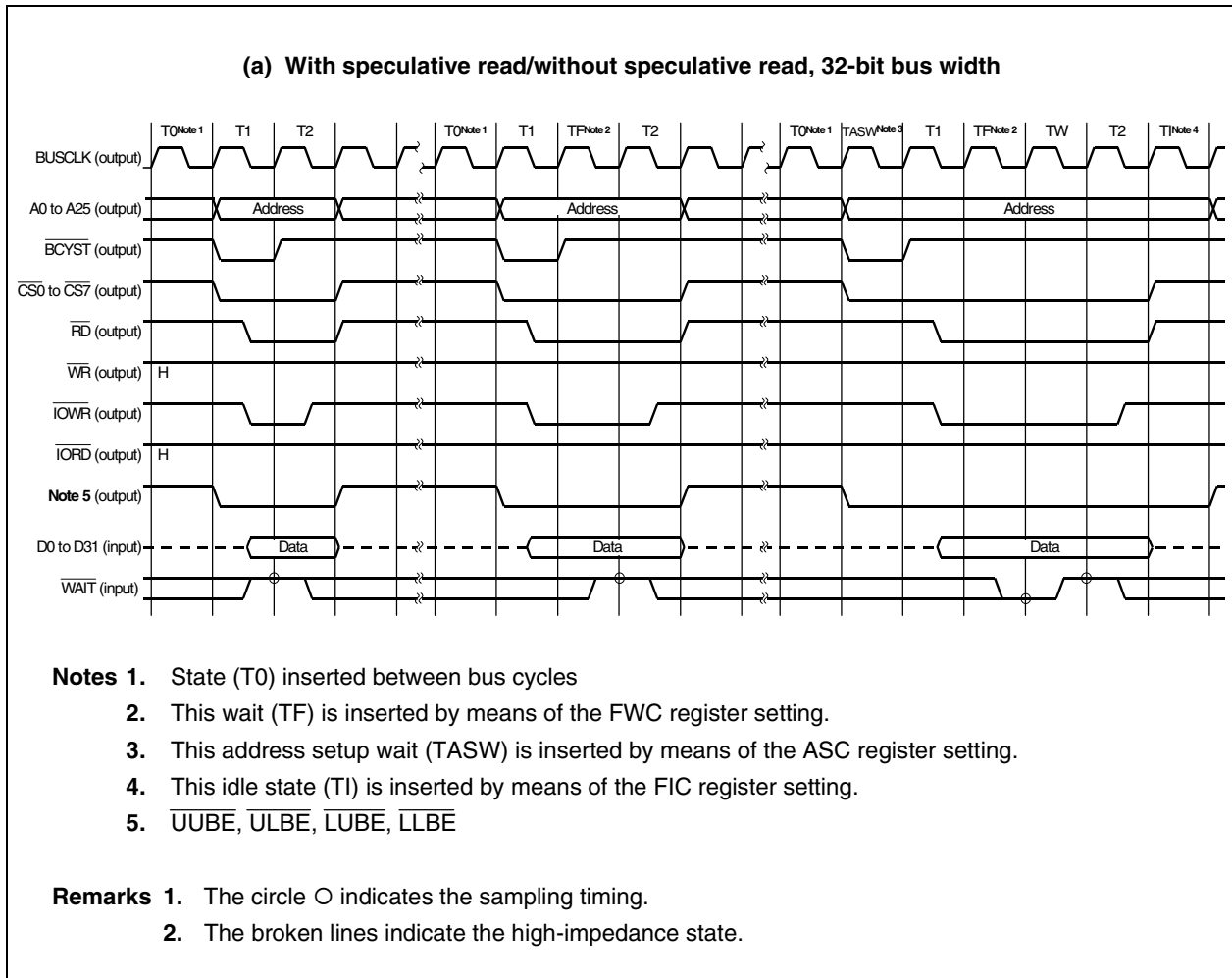


Figure 6-20. Timing of DMA Flyby Transfer (SRAM → External I/O) (2/2)

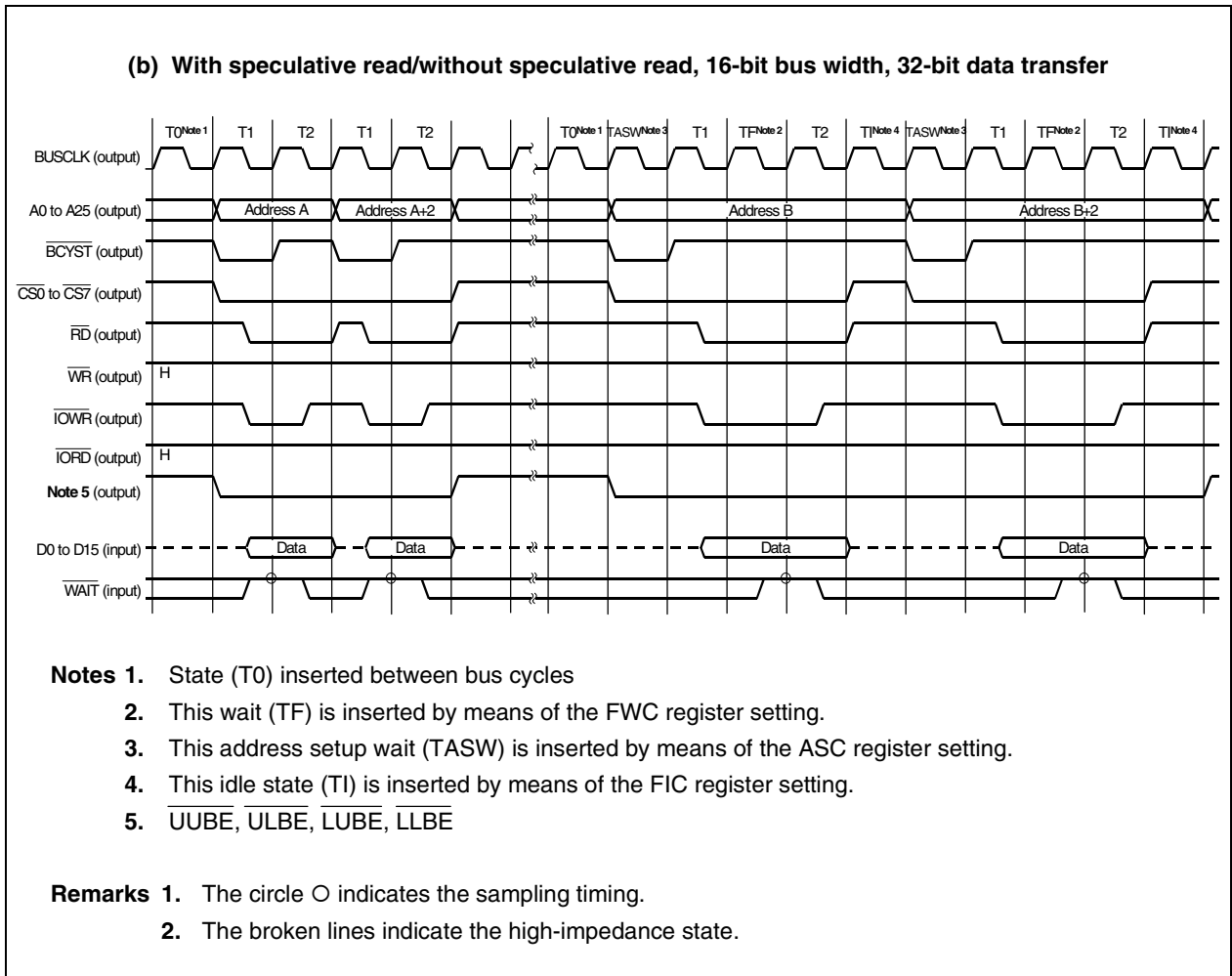


Figure 6-21. Timing of DMA Flyby Transfer (External I/O → SRAM) (1/2)

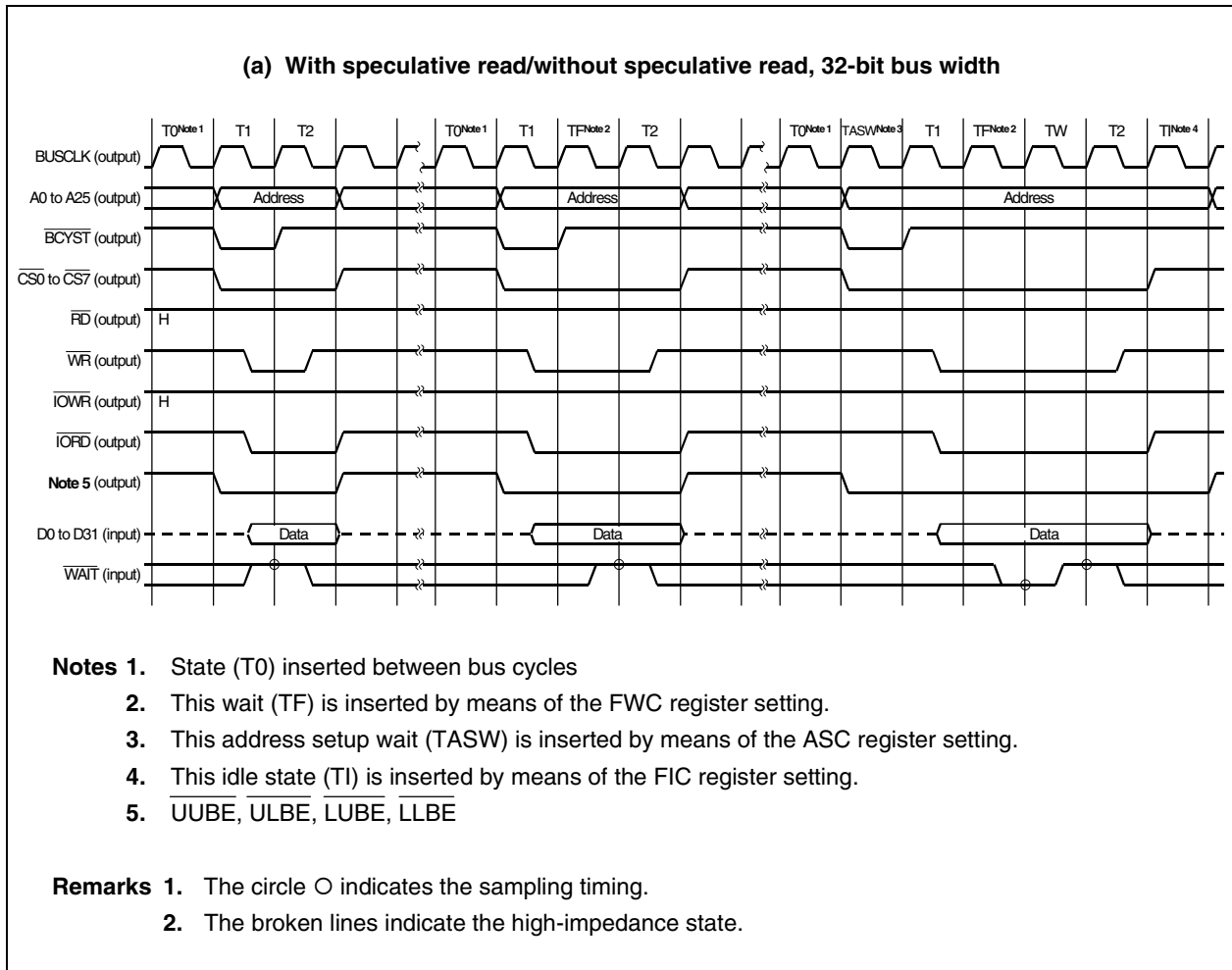


Figure 6-21. Timing of DMA Flyby Transfer (External I/O → SRAM) (2/2)

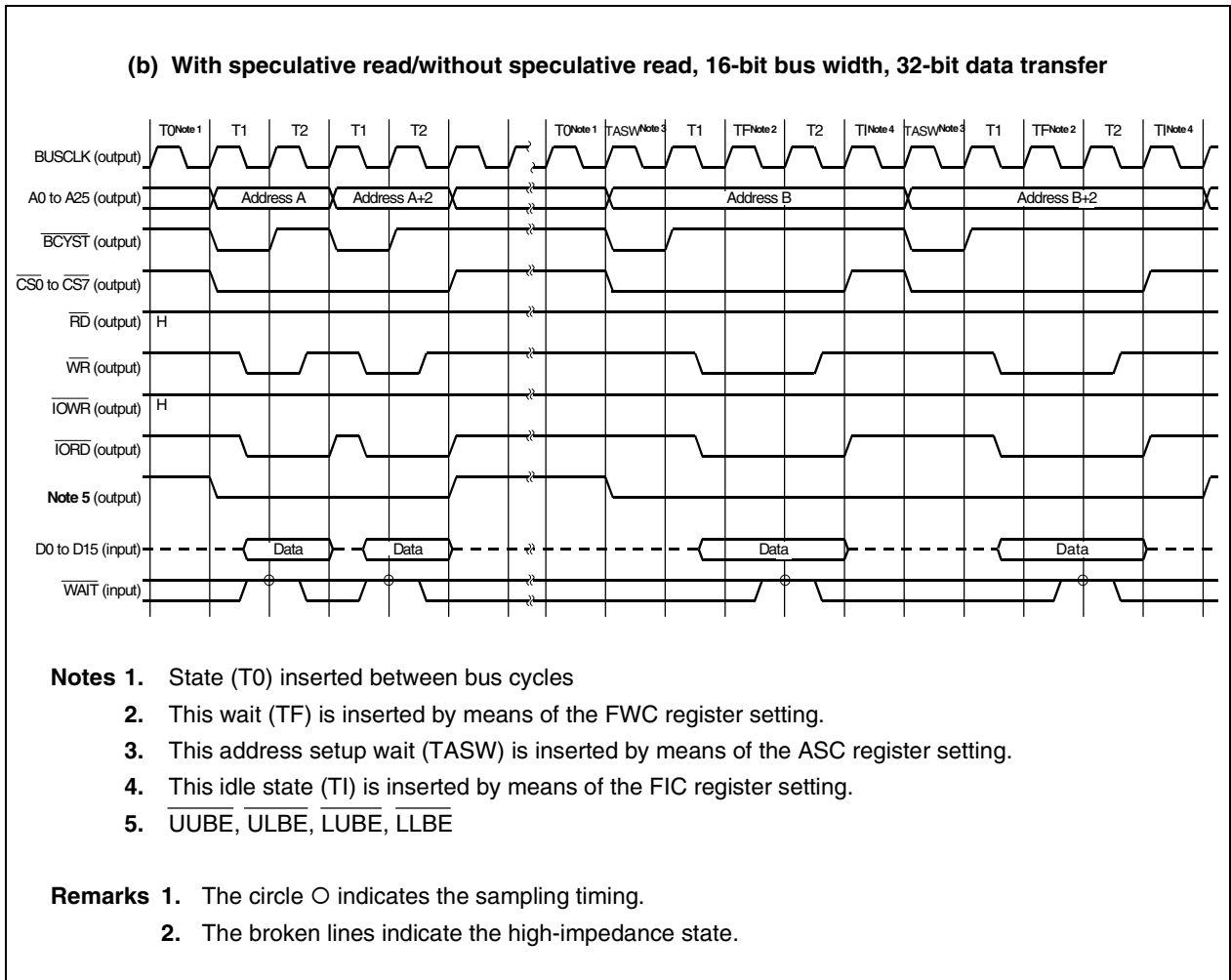


Figure 6-22. Timing of DMA Flyby Transfer (Page ROM → External I/O) (1/2)

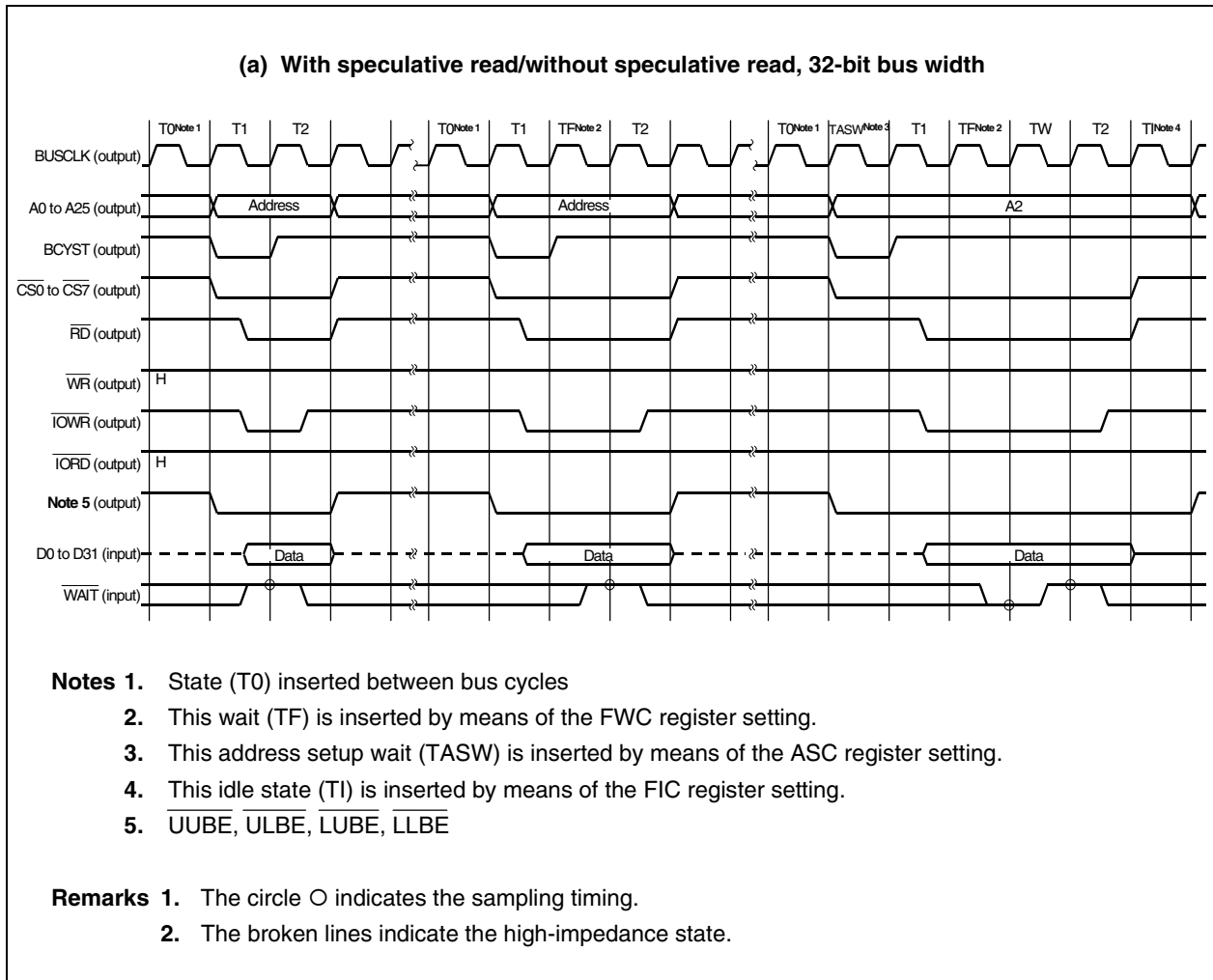


Figure 6-22. Timing of DMA Flyby Transfer (Page ROM → External I/O) (2/2)

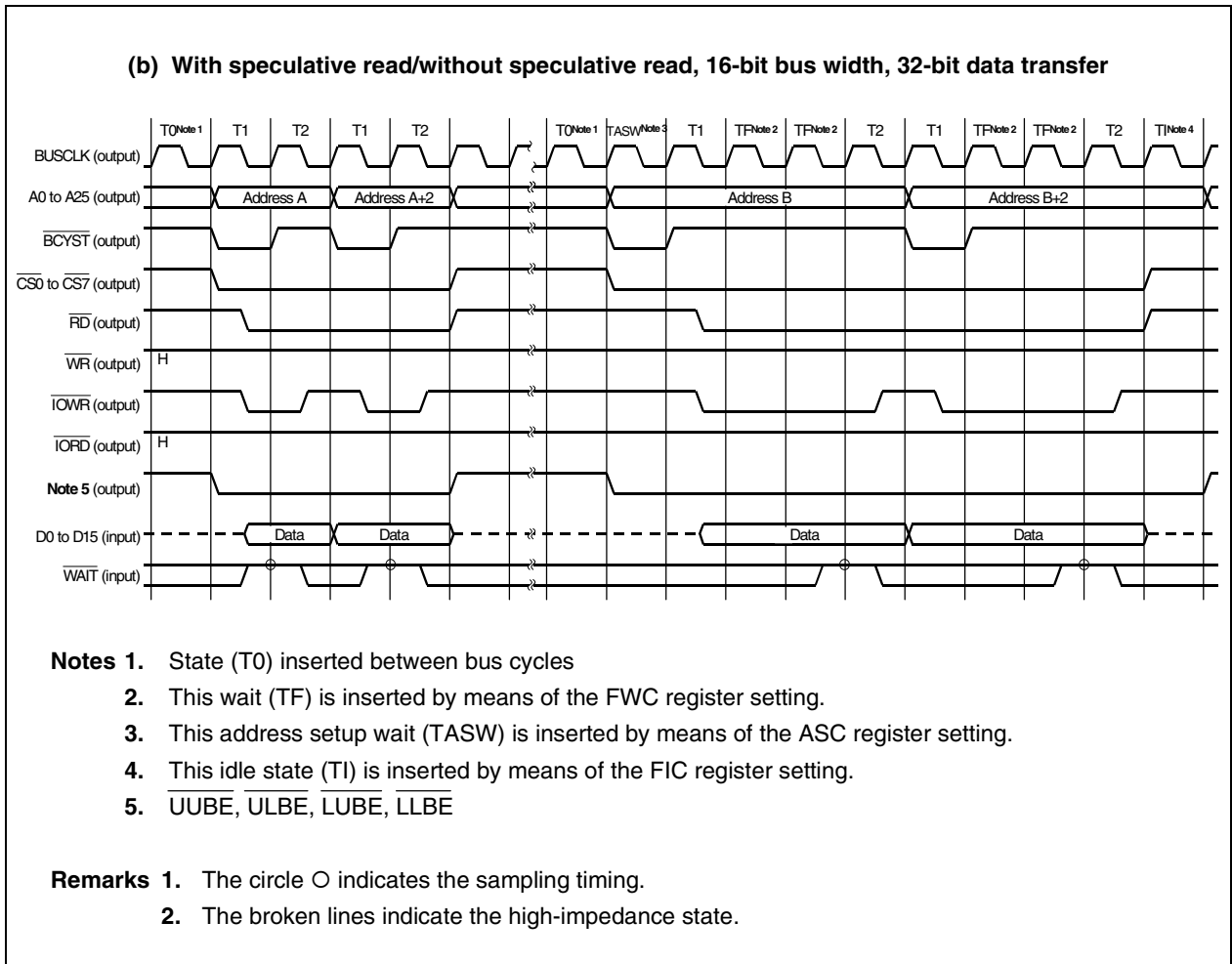
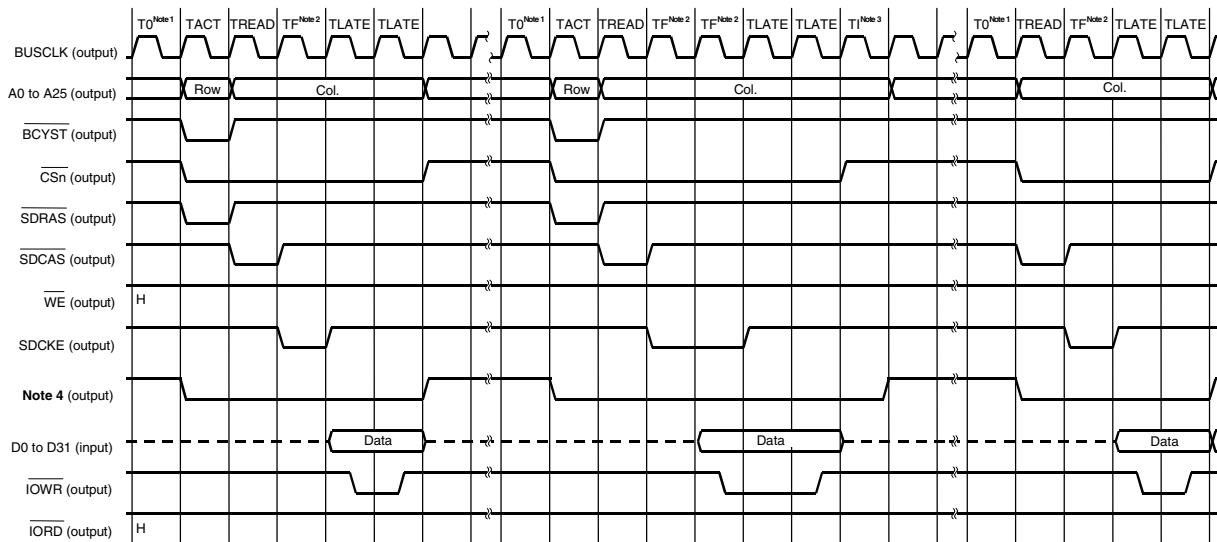


Figure 6-23. Timing of DMA Flyby Transfer (SDRAM → External I/O) (1/2)

(a) With speculative read/without speculative read, 32-bit bus width, CAS latency = 2



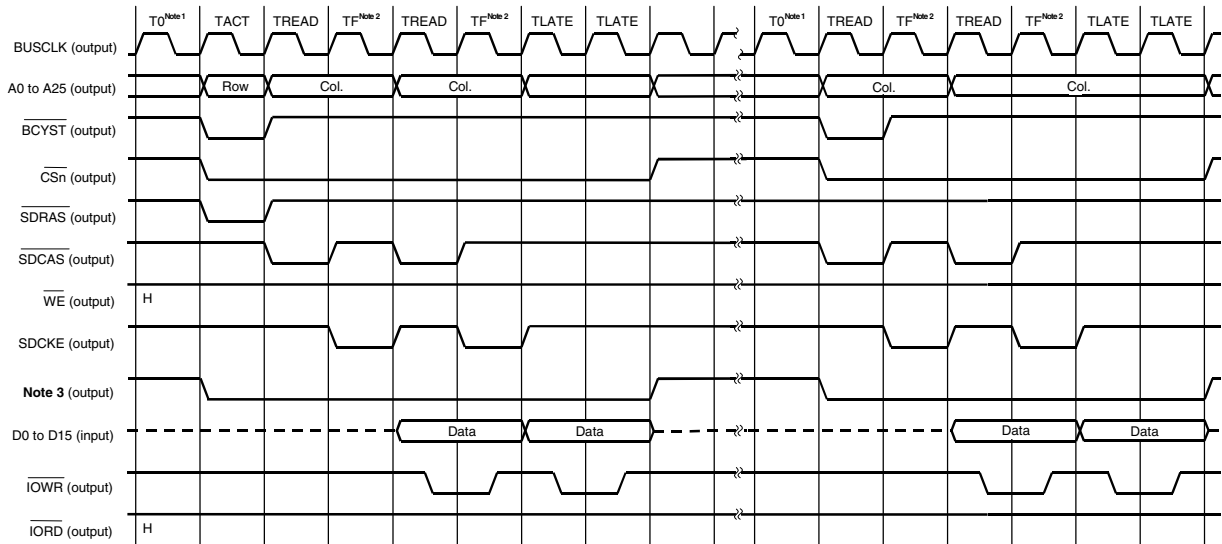
- Notes**
1. State (T0) inserted between bus cycles
 2. This wait (TF) is inserted by means of the FWC register setting.
 3. This idle state (TI) is inserted by means of the FIC register setting.
 4. UUDQM, ULDQM, LLDQM, LUDQM

- Cautions**
1. Set the latency to other than 1 (LTM2 to LTM0 bits of SCRn register = 001) during SDRAM flyby transfer.
 2. In the TF state during SDRAM flyby transfer, wait cycles of the set value of the FWC register + 1 are inserted (one wait cycle is inserted when the set value is 0).

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. n = 1, 3, 4, 6
 3. Col.: Column address
Row: Row address

Figure 6-23. Timing of DMA Flyby Transfer (SDRAM → External I/O) (2/2)

(b) With speculative read/without speculative read, 16-bit bus width, 32-bit data transfer, CAS latency = 2



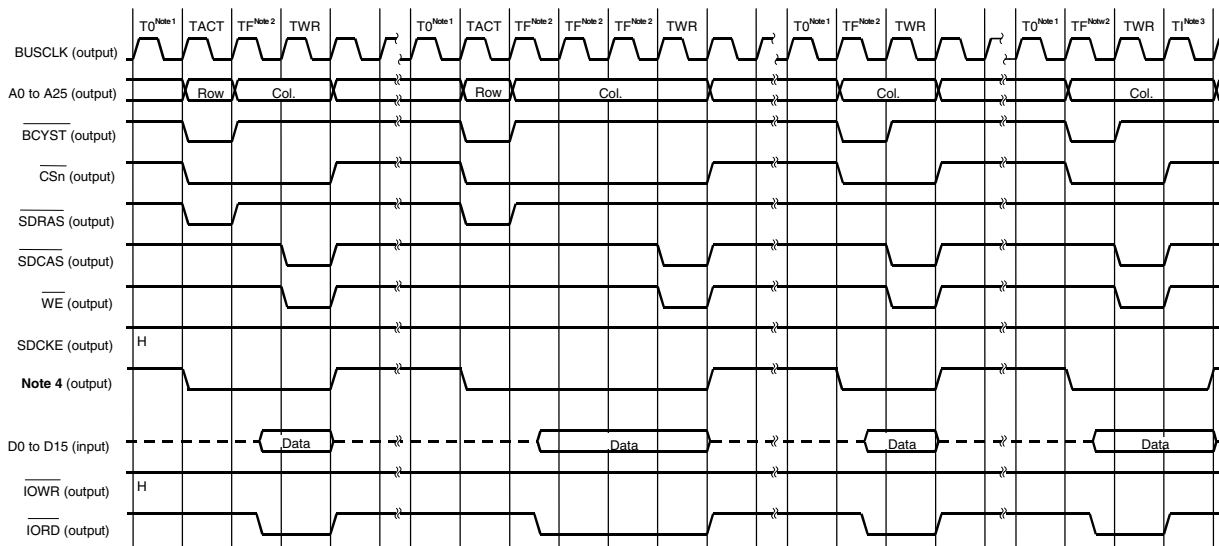
- Notes**
1. State (T0) inserted between bus cycles
 2. This wait (TF) is inserted by means of the FWC register setting.
 3. UUDQM, ULDQM, LLDQM, LUDQM

- Cautions**
1. Set the latency to other than 1 (LTM2 to LTM0 bits of SCRn register = 001) during SDRAM flyby transfer.
 2. In the TF state during SDRAM flyby transfer, wait cycles of the set value of the FWC register + 1 are inserted (one wait cycle is inserted when the set value is 0 (the case where the set value is 0 is shown above)).

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. n = 1, 3, 4, 6
 3. Col.: Column address
Row: Row address

Figure 6-24. Timing of DMA Flyby Transfer (External I/O → SDRAM) (1/2)

(a) With speculative read/without speculative read, 32-bit bus width, CAS latency = 2



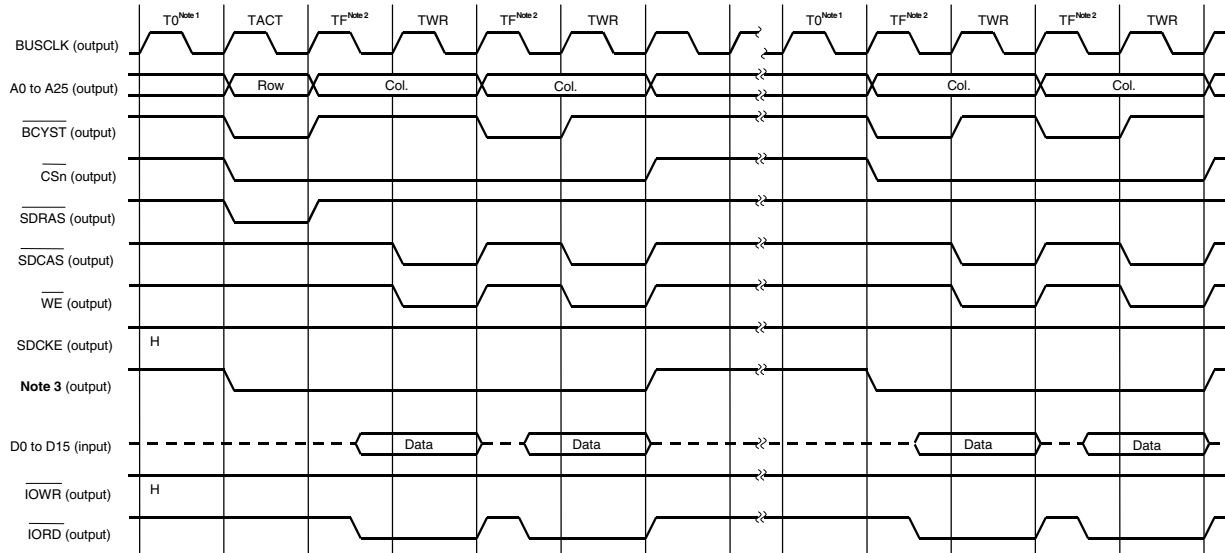
- Notes**
1. State (T0) inserted between bus cycles
 2. This wait (TF) is inserted by means of the FWC register setting.
 3. This idle state (TI) is inserted by means of the FIC register setting.
 4. UUDQM, ULDQM, LLDQM, LUDQM

Caution In the TF state during SDRAM flyby transfer, wait cycles of the set value of the FWC register + 1 are inserted (one wait cycle is inserted when the set value is 0).

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. n = 1, 3, 4, 6
 3. Col.: Column address
Row: Row address

Figure 6-24. Timing of DMA Flyby Transfer (External I/O → SDRAM) (2/2)

(b) With speculative read/without speculative read, 16-bit bus width, 32-bit data transfer, CAS latency = 2



- Notes**
1. State (T0) inserted between bus cycles
 2. This wait (TF) is inserted by means of the FWC register setting.
 3. UUDQM, ULDDQM, LLDQM, LUDQM

Caution In the TF state during SDRAM flyby transfer, wait cycles of the set value of the FWC register + 1 are inserted (one wait cycle is inserted when the set value is 0).

- Remarks**
1. The broken lines indicate the high-impedance state.
 2. $n = 1, 3, 4, 6$
 3. Col.: Column address
Row: Row address

6.6 Transfer Object

6.6.1 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 “x” means “transfer impossible”.

Table 6-5. Relationship Between Transfer Type and Transfer Object

		Transfer Destination									
		2-Cycle Transfer					Flyby Transfer				
		On-chip Peripheral I/O ^{Note 1}	External I/O	Internal Data RAM	Internal Instruction RAM	External Memory	On-Chip Peripheral I/O	External I/O	Internal Data RAM	Internal Instruction RAM	External Memory
★	Transfer Source										
	On-chip peripheral I/O ^{Note 1}	√	√	√	√ ^{Note 3}	√ ^{Note 2}	x	x	x	x	x
	External I/O	√	√	√	√ ^{Note 3}	√	x	x	x	x	√
	Internal data RAM	√	√	x	√ ^{Note 3}	√	x	x	x	x	x
	External memory	√ ^{Note 2}	√	√	√ ^{Note 3}	√ ^{Note 2}	x	√	x	x	x

- Notes**
1. If the transfer object is the on-chip peripheral I/O, only the single transfer mode can be used.
 2. Transfer can also be executed between a little-endian area and a big-endian area.
 3. When the internal instruction RAM is in the write mode (IRAMMn bit of IRAMM register = 1), it can be used only as a transfer destination.

- Cautions**
1. The operation is not guaranteed for combinations of transfer destination and source marked with “x” in Table 6-5.
 2. In the case of flyby transfer, make the data bus width the same for the source and destination.
 3. 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.
 4. Do not use DMA transfer to set on-chip peripheral I/O registers and to read the set values.

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.

If DMA transfer is executed to transfer data of an on-chip peripheral I/O register (as a transfer source or destination), be sure to specify the same transfer size as the register size. For example, to execute DMA transfer of an 8-bit register, be sure to specify byte (8-bit) transfer.

<32-bit transfer>

- Transfer from a 32-bit bus to a 16-bit bus
A read cycle (32 bits) is generated and then a write cycle (16 bits) is generated twice consecutively.
- Transfer from a 32-bit bus to an 8-bit bus
A read cycle (32 bits) is generated and then a write cycle (8 bits) is generated 4 times consecutively.

- Transfer from a 16-bit bus to an 8-bit bus
A read cycle (16 bits) is generated twice consecutively and then a write cycle (8 bits) is generated 4 times consecutively.
- Transfer from a 16-bit bus to a 32-bit bus
A read cycle (16 bits) is generated twice consecutively and then a write cycle (32 bits) is generated. For how to write data to the transfer destination, see **Remark 2** below.
- Transfer from an 8-bit bus to a 32-bit bus
A read cycle (8 bits) is generated 4 times consecutively and then a write cycle (32 bits) is generated. For how to write data to the transfer destination, see **Remark 2** below.
- Transfer from an 8-bit bus to a 16-bit bus
A read cycle (8 bits) is generated 4 times consecutively and then a write cycle (16 bits) is generated twice consecutively. For how to write data to the transfer destination, see **Remark 2** below.

<16-bit transfer>

- Transfer from a 32-bit bus to a 16-bit bus
A read cycle (the higher 16 bits are high impedance) is generated and then a write cycle (16 bits) is generated.
- Transfer from a 32-bit bus to an 8-bit bus
A read cycle (the higher 16 bits are high impedance) is generated and then a write cycle (8 bits) is generated twice consecutively.
- 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 consecutively.
- Transfer from a 16-bit bus to a 32-bit bus
A read cycle (16 bits) is generated and then a write cycle (the higher 16 bits are high impedance) is generated. For how to write data to the transfer destination, see **Remark 2** below.
- Transfer from an 8-bit bus to a 32-bit bus
A read cycle (8 bits) is generated twice consecutively and then a write cycle (the higher 16 bits are high impedance) is generated. For how to write data to the transfer destination, see **Remark 2** below.
- Transfer from an 8-bit bus to a 16-bit bus
A read cycle (8 bits) is generated twice consecutively and then a write cycle (16 bits) is generated. For how to write data to the transfer destination, see **Remark 2** below.

<8-bit transfer>

- Transfer from a 32-bit bus to a 16-bit bus
A read cycle (the higher 24 bits are high impedance) is generated and then a write cycle (the higher 8 bits are high impedance) is generated.
- Transfer from a 32-bit bus to an 8-bit bus
A read cycle (the higher 24 bits are high impedance) is generated and then a write cycle (8 bits) is generated.
- Transfer from a 16-bit bus to an 8-bit bus
A read cycle (the higher 8 bits are high impedance) is generated and then a write cycle (8 bits) is generated.
- Transfer from a 16-bit bus to a 32-bit bus
A read cycle (the higher 8 bits are high impedance) is generated and then a write cycle (the higher 24 bits are high impedance) is generated. For how to write data to the transfer destination, see **Remark 2** below.

- Transfer from an 8-bit bus to a 32-bit bus
A read cycle (8 bits) is generated and then a write cycle (the higher 24 bits are high impedance) is generated. For how to write data to the transfer destination, see **Remark 2** below.
- Transfer from an 8-bit bus to a 16-bit bus
A read cycle (8 bits) is generated and then a write cycle (the higher 8 bits are high impedance) is generated. For how to write data to the transfer destination, see **Remark 2** below.

Remarks 2. Under the following conditions, data is written to the lower byte and then the higher byte of the transfer destination in the little-endian mode, and to the higher byte and lower byte of the destination in the big-endian mode.

- Transfer from a 16-bit bus to a 32-bit bus
 - Transfer from an 8-bit bus to a 32-bit bus
 - Transfer from an 8-bit bus to a 16-bit bus
3. Transfer between the little-endian area and the big-endian area is possible.

6.6.2 External bus cycles during DMA transfer

The external bus cycles during DMA transfer are shown below.

Table 6-6. External Bus Cycles During DMA Transfer

Transfer Type	Transfer Object	External Bus Cycle	
2-cycle transfer	On-chip peripheral I/O, internal data RAM, internal instruction RAM	None	–
	External I/O	Yes	SRAM cycle
	External memory	Yes	Memory access cycle set by the BCT register
Flyby transfer	Between external memory and external I/O	Yes	DMA flyby transfer cycle accessing memory that is set as external memory by the BCT register

6.7 DMA Channel Priorities

The DMA channel priorities are fixed as follows.

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

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, the higher priority DMA transfer request is acknowledged.

Caution If DMA is started by inputting the same signal to more than one $\overline{\text{DMARQn}}$ pin (n = 0 to 3), a DMA channel with a lower priority may be acknowledged before a DMA channel with a higher priority.

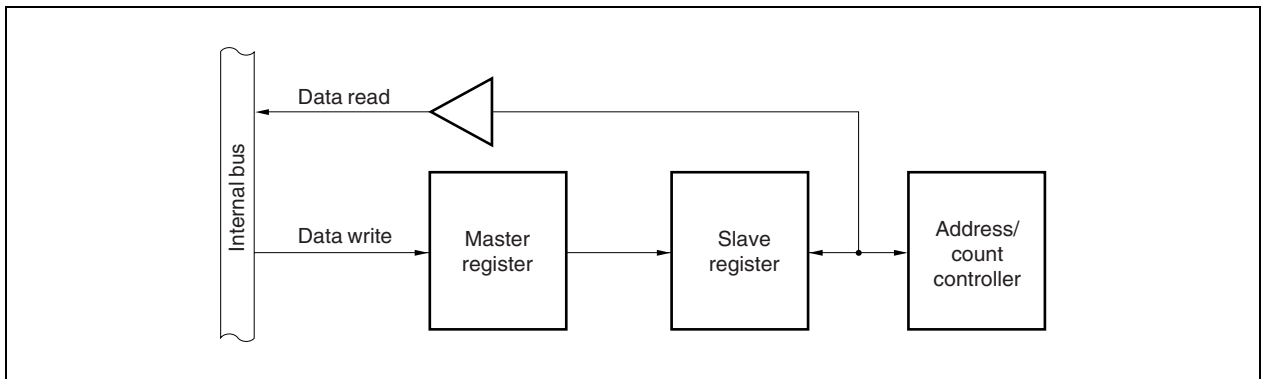
★ 6.8 Next Address Setting Function

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 6-20 shows the configuration of the buffer register.

Figure 6-25. Buffer Register Configuration



The actual DMA transfer is executed in accordance with the contents of the slave register.

The set values that are reflected in the master register and slave register differ as follows, depending on the timing (period) of setting the registers.

(1) Period from system reset to the beginning of the first DMA transfer

The set value is reflected in both the master register and slave register.

(2) During DMA transfer (period from the beginning to the end of DMA transfer)

The set value is reflected only in the master register and not in the slave register (the slave register holds the set value for the next DMA transfer).

However, the contents of the master register are automatically overwritten to the slave register after completion of DMA transfer. If the value of each register is read during this period, the value of the slave register is read.

(3) Period from the end of DMA transfer to the beginning of the next DMA transfer

The set value is reflected in both the master register and slave register.

Remark “The end of DMA transfer” means either of the following.

- Completion of DMA transfer (terminal count)
- Forced termination of DMA transfer (setting the INITn bit of the DMA channel control register (DCHCn) to 1)

If the setting of a new DMA transfer is made using the DSA_nH, DSA_nL, DDA_nL, and DBC_n registers during DMA transfer, the values of the registers are automatically updated after completion of transfer^{Note}.

Note Before setting a new DMA transfer, confirm the start of the preceding DMA transfer.
If the setting of the new DMA transfer is made before the start of the preceding DMA transfer, the new set value is overwritten to both the master register and slave register, and DMA transfer according to the preceding set value cannot be executed.

6.9 DMA Transfer Start Factors

There are 4 types of DMA transfer start factors, as shown below.

- ★ **Cautions 1. Do not use two or more start factors ((1) to (4)) in combination for the same channel (if two or more start factors are generated at the same time, only one of them is valid, but the valid start factor cannot be identified).**
The operation is not guaranteed if two or more start factors are used in combination.
- ★ **2. If DMA transfer is started via software and if the software does not correctly detect whether the expected DMA transfer operation has been completed through manipulation (setting to 1) of the STGn bit of the DCHCn register, it cannot be guaranteed whether the next (second) manipulation of the STGn bit corresponds to the start of “the next DMA transfer expected by software” (n = 0 to 3).**
For example, suppose single transfer is started by manipulating the STGn bit. Even if the STGn bit is manipulated next (the second time) without checking by software whether the single transfer has actually been executed, the next (second) DMA transfer is not always executed. This is because the STGn bit may be manipulated the second time before the first DMA transfer is started or completed because, for example, DMA transfer with a higher priority had already been started when the STGn bit was manipulated for the first time.
It is therefore necessary to manipulate the STGn bit next time (the second time) after checking whether DMA transfer started by the first manipulation of the STGn bit has been completed.
Completion of DMA transfer can be checked in the following ways.
 - Detecting the acknowledge signal ($\overline{\text{DMAAKn}}$) or terminal count signal ($\overline{\text{TCn}}$) by using a peripheral port or interrupt
 - Checking the contents of the DBCn register

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

Requests from the $\overline{\text{DMARQn}}$ pin are sampled each time the BUSCLK signal rises ($n = 0$ to 3).

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 is set, the $\overline{\text{DMARQn}}$ signal becomes valid. If the $\overline{\text{DMARQn}}$ signal set by the DTFRn register becomes active in this status, DMA transfer starts.

(2) Request from software

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

- STGn bit = 1
- Enn bit = 1
- TCn bit = 0

(3) Request from 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

(4) Request by USB (enabled only in single transfer mode)

If a request is generated from the USB set to the DTFRn register when the Enn and TCn bits of the DCHCn register are set as follows, DMA transfer starts ($n = 0$ to 3).

- Enn bit = 1
- TCn bit = 0

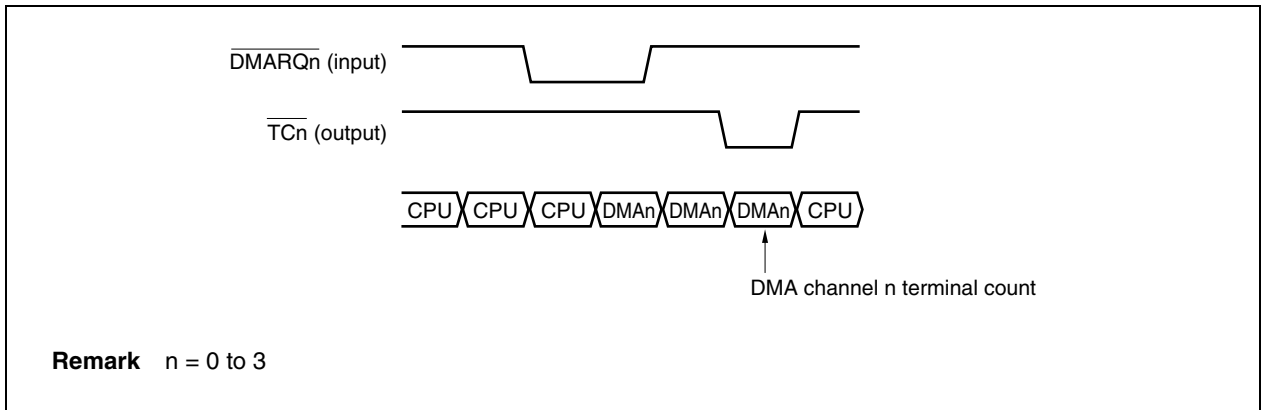
Remark Since the $\overline{\text{DMARQn}}$ signal is level-sampled and not edge-detected, to enable edge detection of a DMA request, set an external interrupt request for the DMA start trigger instead of using the $\overline{\text{DMARQn}}$ signal ($n = 0$ to 3).

6.10 Terminal Count Output upon DMA Transfer End

The terminal count signal (\overline{TCn}) becomes active for one clock of BUSCLK during the last DMA transfer cycle ($n = 3$ to 0).

The \overline{TCn} signal becomes active in the clock following the clock in which the \overline{BCYST} signal becomes active during the last DMA transfer cycle.

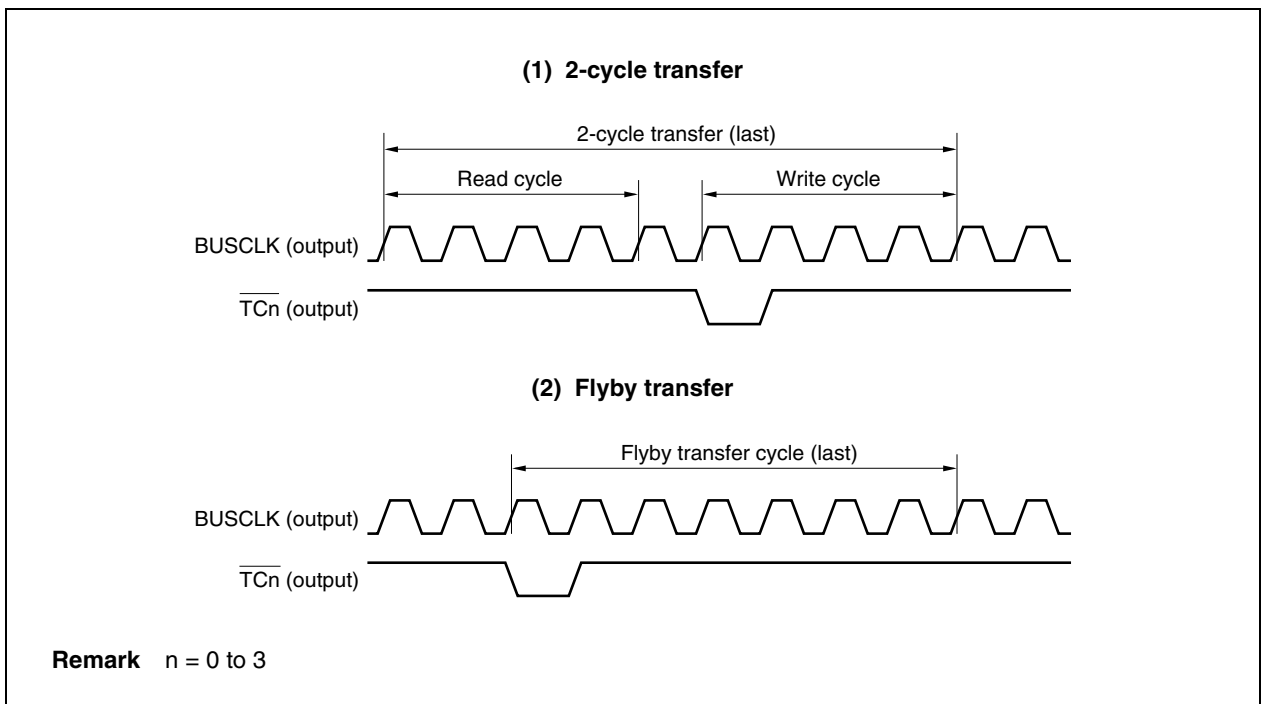
Figure 6-26. Terminal Count Signal (\overline{TCn}) Timing Example (1)



The \overline{TCn} signal becomes active for one clock at the beginning of the write cycle of the last DMA transfer when 2-cycle transfer is executed.

When flyby transfer is executed, the \overline{TCn} signal becomes active for one clock at the beginning of the last DMA transfer cycle.

Figure 6-27. Terminal Count Signal (\overline{TCn}) Timing Example (2)



6.11 Forcible Interruption

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

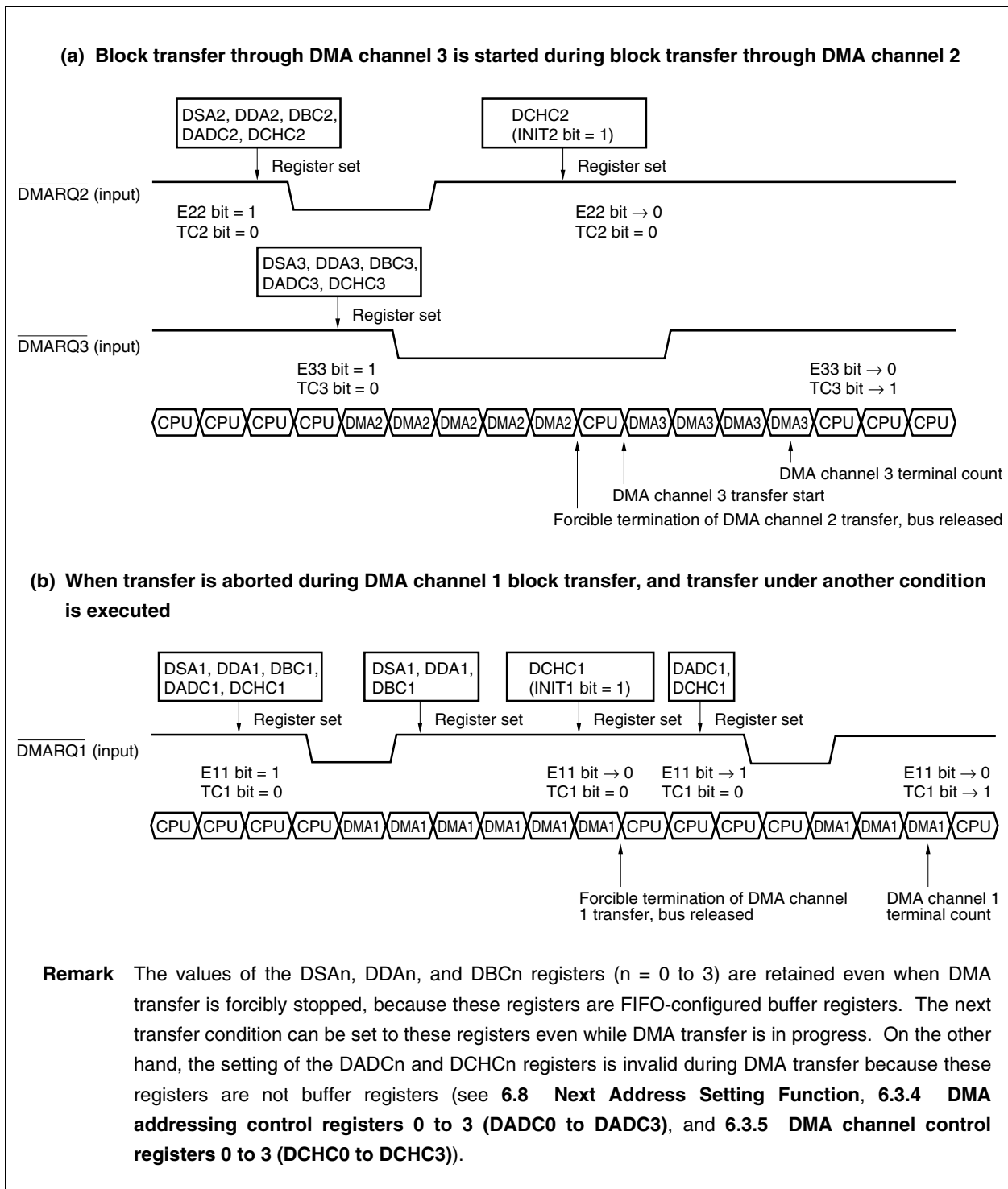
At such a time, the DMAC resets the Enn bit of the DCHCn register of all channels to 0 and the DMA transfer disabled state is entered. An NMI request can then be acknowledged after the DMA transfer that was being executed when the NMI was input is complete (n = 0 to 3).

6.12 Forcible Termination

DMA transfer can be forcibly terminated by the INITn bit of the DCHCn register, in addition to the forcible interruption 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 6-28. Example of Forcible Termination of DMA Transfer



★ 6.13 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 6-7. Number of Minimum Execution Clocks in DMA Cycle

DMA Cycle		Number of Minimum Execution Clocks
<1> Time to respond to DMA request		4 clocks ^{Note 1}
<2> Memory access	External memory access	Differs depending on the memory connected
	Internal data RAM or internal instruction RAM (read mode) access	2 clocks ^{Note 2}

- Notes**
1. If an external interrupt (INTPn) is specified as a factor of starting DMA transfer, noise elimination time is added (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, or L1).
 2. Two clocks are required for the DMA cycle.

The minimum execution clock in the DMA cycle in each transfer mode is as follows.

Single transfer: DMA response time (<1>) + Transfer source memory access (<2>) + 2^{Note} + Transfer destination memory access (<2>)

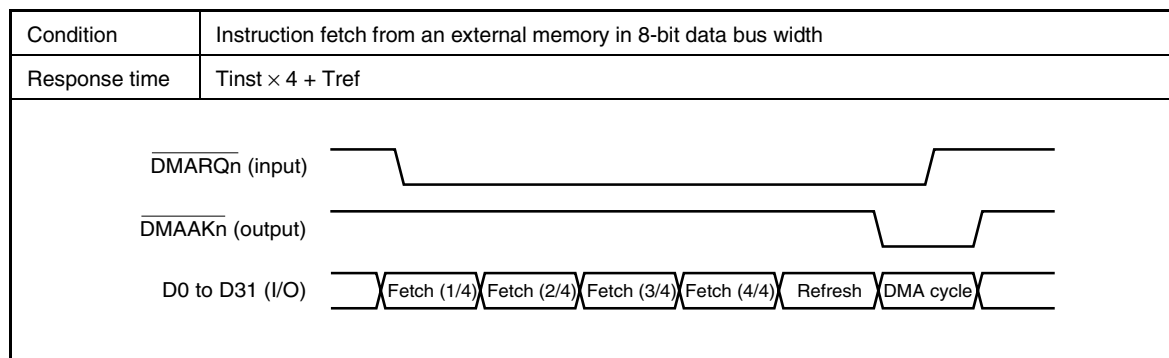
Block transfer: DMA response time (<1>) + (Transfer source memory access (<2>) + 2^{Note} + Transfer destination memory access (<2>)) × Number of transfers

Note Two clocks are always inserted between the read cycle and write cycle of DMA transfer.

6.14 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 to 3

(2) Condition 2

Condition	Word data access with an external memory in 8-bit data bus width
Response time	$T_{data} \times 4 + T_{ref}$
<p>The diagram shows three signals over time. DMARQn (input) is active low, going from high to low and back to high. DMAAKn (output) is active low, going from high to low and back to high. D0 to D31 (I/O) shows four data accesses labeled 'Data (1/4)', 'Data (2/4)', 'Data (3/4)', and 'Data (4/4)', followed by a 'Refresh' cycle and a 'DMA cycle'.</p>	

Remark n = 0 to 3

(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	$T_{inst} \times 4 + T_{data} \times 2 + T_{ref}$
<p>The diagram shows three signals over time. DMARQn (input) is active low, going from high to low and back to high. DMAAKn (output) is active low, going from high to low and back to high. D0 to D31 (I/O) shows a 'Data read', four instruction fetches labeled 'Fetch (1/4)', 'Fetch (2/4)', 'Fetch (3/4)', and 'Fetch (4/4)', followed by a 'Data write', a 'Refresh' cycle, and a 'DMA cycle'.</p>	

- 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 to 3

6.15 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 data RAM, internal instruction RAM, or peripheral I/O) during DMA transfer.

(2) Transfer of misaligned data

DMA transfer of 32-/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 DMA transfer is executed to transfer data to/from an external device, the CPU can access internal data RAM and internal instruction RAM^{Note} not being used for DMA transfer.

Because the DMA controller has a priority higher than the CPU acquiring bus mastership, a CPU access that takes place during DMA transfer is kept waiting until the preceding DMA transfer is completed and the bus is released to the CPU. If DMA transfer is executed between the external memory and on-chip peripheral I/O, however, the CPU can access the internal data RAM and internal instruction RAM^{Note}.

Note When the IRAMMn bit of the IRAMM register = 0 (n = 0, 1)

(4) Holding $\overline{\text{DMARQn}}$ signal

Be sure to keep the $\overline{\text{DMARQn}}$ signal active until the $\overline{\text{DMAAKn}}$ signal becomes active (n = 0 to 3).

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

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

If the transfer object is the on-chip peripheral I/O or internal instruction RAM, the $\overline{\text{DMAAKn}}$ signal is output even in the DMA cycle executed on the on-chip peripheral I/O or internal instruction RAM.

(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, the DMA channel with the lower priority may be accepted before the DMA channel with the higher priority.

★

(7) Read values of DSAn and DDAn registers

If the values of the DSAn and DDAn registers are read during DMA transfer, the values in the middle of being updated may be read (n = 0 to 3).

For example, if the DSAnH register and the DSAnL register are read in that order when the value of the DMA transfer source address (DSAn register) is "0000FFFFH" and the counting direction is incremental (when the SADn1 and SADn0 bits of the DADCn register = 00), the value of the DSAnL register differs as follows depending on whether DMA transfer is executed immediately after the DSAnH register has been read.

(a) If DMA transfer does not occur while the DSAn register is being read

<1> Reading DSAnH register: DSAnH = 0000H

<2> Reading DSAnL register: DSAnL = FFFFH

(b) If DMA transfer occurs while the DSAn register is being read

<1> Reading DSAnH register: DSAnH = 0000H

<2> Occurrence of DMA transfer

<3> Incrementing DSAn register : DSAn = 00010000H

<4> Reading DSAnL register: DSAnL = 0000H

6.15.1 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.

6.16 DMA Transfer End

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

CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION

The V850E/ME2 is provided with a dedicated interrupt controller (INTC) for interrupt servicing and can process a total of 91 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/ME2 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: 90 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^{Note}, edge detection, and valid edge specification for external interrupt request signals.

Note For details of the noise eliminator, see **14.6 Noise Eliminator**.

○ 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 (1/3)

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	00100000H	Undefined
Non-maskable	Interrupt	NMI0	–	NMI input	–	–	0010H	00000010H	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	INTP10	P1IC0	$\overline{\text{INTP10}}$ pin	Pin	0	0080H	00000080H	nextPC
	Interrupt	INTP11	P1IC1	$\overline{\text{INTP11}}$ pin	Pin	1	0090H	00000090H	nextPC
	Interrupt	INTP21	P2IC1	$\overline{\text{INTP21}}$ pin	Pin	2	00A0H	000000A0H	nextPC
	Interrupt	INTP22	P2IC2	$\overline{\text{INTP22}}$ pin	Pin	3	00B0H	000000B0H	nextPC
	Interrupt	INTP23	P2IC3	$\overline{\text{INTP23}}$ pin	Pin	4	00C0H	000000C0H	nextPC
	Interrupt	INTP24	P2IC4	$\overline{\text{INTP24}}$ pin	Pin	5	00D0H	000000D0H	nextPC
	Interrupt	INTP25	P2IC5	$\overline{\text{INTP25}}$ pin	Pin	6	00E0H	000000E0H	nextPC
	Interrupt	INTP50	P5IC0	$\overline{\text{INTP50}}$ pin	Pin	7	00F0H	000000F0H	nextPC
	Interrupt	INTP51	P5IC1	$\overline{\text{INTP51}}$ pin	Pin	8	0100H	00000100H	nextPC
	Interrupt	INTP52	P5IC2	$\overline{\text{INTP52}}$ pin	Pin	9	0110H	00000110H	nextPC
	Interrupt	INTP65	P6IC5	$\overline{\text{INTP65}}$ pin	Pin	10	0120H	00000120H	nextPC
	Interrupt	INTP66	P6IC6	$\overline{\text{INTP66}}$ pin	Pin	11	0130H	00000130H	nextPC
	Interrupt	INTP67	P6IC7	$\overline{\text{INTP67}}$ pin	Pin	12	0140H	00000140H	nextPC
	Interrupt	INTPD0	PDIC0	$\overline{\text{INTPD0}}$ pin	Pin	13	0150H	00000150H	nextPC
	Interrupt	INTPD1	PDIC1	$\overline{\text{INTPD1}}$ pin	Pin	14	0160H	00000160H	nextPC
	Interrupt	INTPD2	PDIC2	$\overline{\text{INTPD2}}$ pin	Pin	15	0170H	00000170H	nextPC
	Interrupt	INTPD3	PDIC3	$\overline{\text{INTPD3}}$ pin	Pin	16	0180H	00000180H	nextPC
	Interrupt	INTPD4	PDIC4	$\overline{\text{INTPD4}}$ pin	Pin	17	0190H	00000190H	nextPC
	Interrupt	INTPD5	PDIC5	$\overline{\text{INTPD5}}$ pin	Pin	18	01A0H	000001A0H	nextPC
	Interrupt	INTPD6	PDIC6	$\overline{\text{INTPD6}}$ pin	Pin	19	01B0H	000001B0H	nextPC
	Interrupt	INTPD7	PDIC7	$\overline{\text{INTPD7}}$ pin	Pin	20	01C0H	000001C0H	nextPC
	Interrupt	INTPD8	PDIC8	$\overline{\text{INTPD8}}$ pin	Pin	21	01D0H	000001D0H	nextPC
	Interrupt	INTPD9	PDIC9	$\overline{\text{INTPD9}}$ pin	Pin	22	01E0H	000001E0H	nextPC
	Interrupt	INTPD10	PDIC10	$\overline{\text{INTPD10}}$ pin	Pin	23	01F0H	000001F0H	nextPC
	Interrupt	INTPD11	PDIC11	$\overline{\text{INTPD11}}$ pin	Pin	24	0200H	00000200H	nextPC
	Interrupt	INTPD12	PDIC12	$\overline{\text{INTPD12}}$ pin	Pin	25	0210H	00000210H	nextPC
	Interrupt	INTPD13	PDIC13	$\overline{\text{INTPD13}}$ pin	Pin	26	0220H	00000220H	nextPC
	Interrupt	INTPD14	PDIC14	$\overline{\text{INTPD14}}$ pin	Pin	27	0230H	00000230H	nextPC
	Interrupt	INTPD15	PDIC15	$\overline{\text{INTPD15}}$ pin	Pin	28	0240H	00000240H	nextPC
	Interrupt	INTPL0	PLIC0	$\overline{\text{INTPL0}}$ pin	Pin	29	0250H	00000250H	nextPC
	Interrupt	INTPL1	PLIC1	$\overline{\text{INTPL1}}$ pin	Pin	30	0260H	00000260H	nextPC
	Interrupt	INTOVC0	OVCIC0	Timer C0 overflow	RPU	31	0270H	00000270H	nextPC
	Interrupt	INTOVC1	OVCIC1	Timer C1 overflow	RPU	32	0280H	00000280H	nextPC
	Interrupt	INTOVC2	OVCIC2	Timer C2 overflow	RPU	33	0290H	00000290H	nextPC
Interrupt	INTOVC3	OVCIC3	Timer C3 overflow	RPU	34	02A0H	000002A0H	nextPC	

Note n = 0 to FH

Table 7-1. Interrupt/Exception Source List (2/3)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Generating Source	Generating Unit				
Maskable	Interrupt	INTOVC4	OVCIC4	Timer C4 overflow	RPU	35	02B0H	000002B0H	nextPC
	Interrupt	INTOVC5	OVCIC5	Timer C5 overflow	RPU	36	02C0H	000002C0H	nextPC
	Interrupt	INTPC00/ INTCCC00	CCC0IC0	Match of INTPC00 pin/CCC00	Pin/RPU	37	02D0H	000002D0H	nextPC
	Interrupt	INTPC01/ INTCCC01	CCC0IC1	Match of INTPC01 pin/CCC01	Pin/RPU	38	02E0H	000002E0H	nextPC
	Interrupt	INTPC10/ INTCCC10	CCC1IC0	Match of INTPC10 pin/CCC10	Pin/RPU	39	02F0H	000002F0H	nextPC
	Interrupt	INTPC11/ INTCCC11	CCC1IC1	Match of INTPC11 pin/CCC11	Pin/RPU	40	0300H	00000300H	nextPC
	Interrupt	INTPC20/ INTCCC20	CCC2IC0	Match of INTPC20 pin/CCC20	Pin/RPU	41	0310H	00000310H	nextPC
	Interrupt	INTPC21/ INTCCC21	CCC2IC1	Match of INTPC21 pin/CCC21	Pin/RPU	42	0320H	00000320H	nextPC
	Interrupt	INTPC30/ INTCCC30	CCC3IC0	Match of INTPC30 pin/CCC30	Pin/RPU	43	0330H	00000330H	nextPC
	Interrupt	INTPC31/ INTCCC31	CCC3IC1	Match of INTPC31 pin/CCC31	Pin/RPU	44	0340H	00000340H	nextPC
	Interrupt	INTCCC40	CCC4IC0	CCC40 match	RPU	45	0350H	00000350H	nextPC
	Interrupt	INTCCC41	CCC4IC1	CCC41 match	RPU	46	0360H	00000360H	nextPC
	Interrupt	INTCCC50	CCC5IC0	CCC50 match	RPU	47	0370H	00000370H	nextPC
	Interrupt	INTCCC51	CCC5IC1	CCC51 match	RPU	48	0380H	00000380H	nextPC
	Interrupt	INTCMD0	CMDIC0	CMD0 match	RPU	49	0390H	00000390H	nextPC
	Interrupt	INTCMD1	CMDIC1	CMD1 match	RPU	50	03A0H	000003A0H	nextPC
	Interrupt	INTCMD2	CMDIC2	CMD2 match	RPU	51	03B0H	000003B0H	nextPC
	Interrupt	INTCMD3	CMDIC3	CMD3 match	RPU	52	03C0H	000003C0H	nextPC
	Interrupt	INTCC100	CC10IC0	CC100 match	RPU	53	03D0H	000003D0H	nextPC
	Interrupt	INTCC101	CC10IC1	CC101 match	RPU	54	03E0H	000003E0H	nextPC
	Interrupt	INTCM100	CM10IC0	CM100 match	RPU	55	03F0H	000003F0H	nextPC
	Interrupt	INTCM101	CM10IC1	CM101 match	RPU	56	0400H	00000400H	nextPC
	Interrupt	INTOV10	OV1IC0	Timer ENC10 overflow	RPU	57	0410H	00000410H	nextPC
	Interrupt	INTUD10	UD1IC0	Timer ENC10 underflow	RPU	58	0420H	00000420H	nextPC
	Interrupt	INTCC110	CC11IC0	CC110 match	RPU	59	0430H	00000430H	nextPC
	Interrupt	INTCC111	CC11IC1	CC111 match	RPU	60	0440H	00000440H	nextPC
	Interrupt	INTCM110	CM11IC0	CM110 match	RPU	61	0450H	00000450H	nextPC
	Interrupt	INTCM111	CM11IC1	CM111 match	RPU	62	0460H	00000460H	nextPC
	Interrupt	INTOV11	OV1IC1	Timer ENC11 overflow	RPU	63	0470H	00000470H	nextPC
	Interrupt	INTUD11	UD1IC1	Timer ENC11 underflow	RPU	64	0480H	00000480H	nextPC
	Interrupt	INTDMA0	DMAIC0	End of DMA0 transfer	DMA	65	0490H	00000490H	nextPC
	Interrupt	INTDMA1	DMAIC1	End of DMA1 transfer	DMA	66	04A0H	000004A0H	nextPC
	Interrupt	INTDMA2	DMAIC2	End of DMA2 transfer	DMA	67	04B0H	000004B0H	nextPC
	Interrupt	INTDMA3	DMAIC3	End of DMA3 transfer	DMA	68	04C0H	000004C0H	nextPC
Interrupt	INTCSI30	CSI3IC0	CSI30 transmission/ reception completion	SIO	69	04D0H	000004D0H	nextPC	
Interrupt	INTCOVF30	COVF3IC0	CSI30BUF overflow	SIO	70	04E0H	000004E0H	nextPC	

Table 7-1. Interrupt/Exception Source List (3/3)

Type	Classification	Interrupt/Exception Source				Default Priority	Exception Code	Handler Address	Restored PC
		Name	Controlling Register	Generating Source	Generating Unit				
Maskable	Interrupt	INTCSI31	CSI3IC1	CSI31 transmission/reception completion	SIO	71	04F0H	000004F0H	nextPC
	Interrupt	INTCOVF31	COVF3IC1	CSI31BUF overflow	SIO	72	0500H	00000500H	nextPC
	Interrupt	UBTIRE0	UREIC0	UARTB0 reception error	SIO	73	0510H	00000510H	nextPC
	Interrupt	UBTIR0	URIC0	UARTB0 reception completion	SIO	74	0520H	00000520H	nextPC
	Interrupt	UBTIT0	UTIC0	UARTB0 transmission completion	SIO	75	0530H	00000530H	nextPC
	Interrupt	UBTIF0	UIFIC0	UARTB0 FIFO transmission completion	SIO	76	0540H	00000540H	nextPC
	Interrupt	UBTITO0	UTOIC0	UARTB0 reception timeout	SIO	77	0550H	00000550H	nextPC
	Interrupt	UBTIRE1	UREIC1	UARTB1 reception error	SIO	78	0560H	00000560H	nextPC
	Interrupt	UBTIR1	URIC1	UARTB1 reception completion	SIO	79	0570H	00000570H	nextPC
	Interrupt	UBTIT1	UTIC1	UARTB1 transmission completion	SIO	80	0580H	00000580H	nextPC
	Interrupt	UBTIF1	UIFIC1	UARTB1 FIFO transmission completion	SIO	81	0590H	00000590H	nextPC
	Interrupt	UBTITO1	UTOIC1	UARTB1 reception timeout	SIO	82	05A0H	000005A0H	nextPC
	Interrupt	INTAD	ADIC	End of A/D conversion	ADC	83	05B0H	000005B0H	nextPC
	Interrupt	INTUSB0B	US0BIC	USB function status 0	USBF	84	05C0H	000005C0H	nextPC
	Interrupt	INTUSB1B	US1BIC	USB function status 1	USBF	85	05D0H	000005D0H	nextPC
	Interrupt	INTUSB2B	US2BIC	USB function status 2	USBF	86	05E0H	000005E0H	nextPC
	Interrupt	USBSP2B	USP2IC	Forcible end of USB function EP2 DMA	USBF	87	05F0H	000005F0H	nextPC
	Interrupt	USBSP4B	USP4IC	Forcible end of USB function EP4 DMA	USBF	88	0600H	00000600H	nextPC
	Interrupt	INTRSUM	RSUMIC	USB Resume signal detection	SIE	89	0610H	00000610H	nextPC

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 divide instruction (DIV, DIVH, DIVU, DIVHU) 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 by (Restored PC – 4).

7.2 Non-Maskable Interrupts

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 (NMIR0) of external interrupt rising edge specification register 2 (INTR2) or bit 0 (NMIF0) of external interrupt falling edge specification register 2 (INTF2) is detected at the NMI pin, the interrupt occurs.

While the service program of the non-maskable interrupt is being executed (PSW.NP = 1), the acknowledgment 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). 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

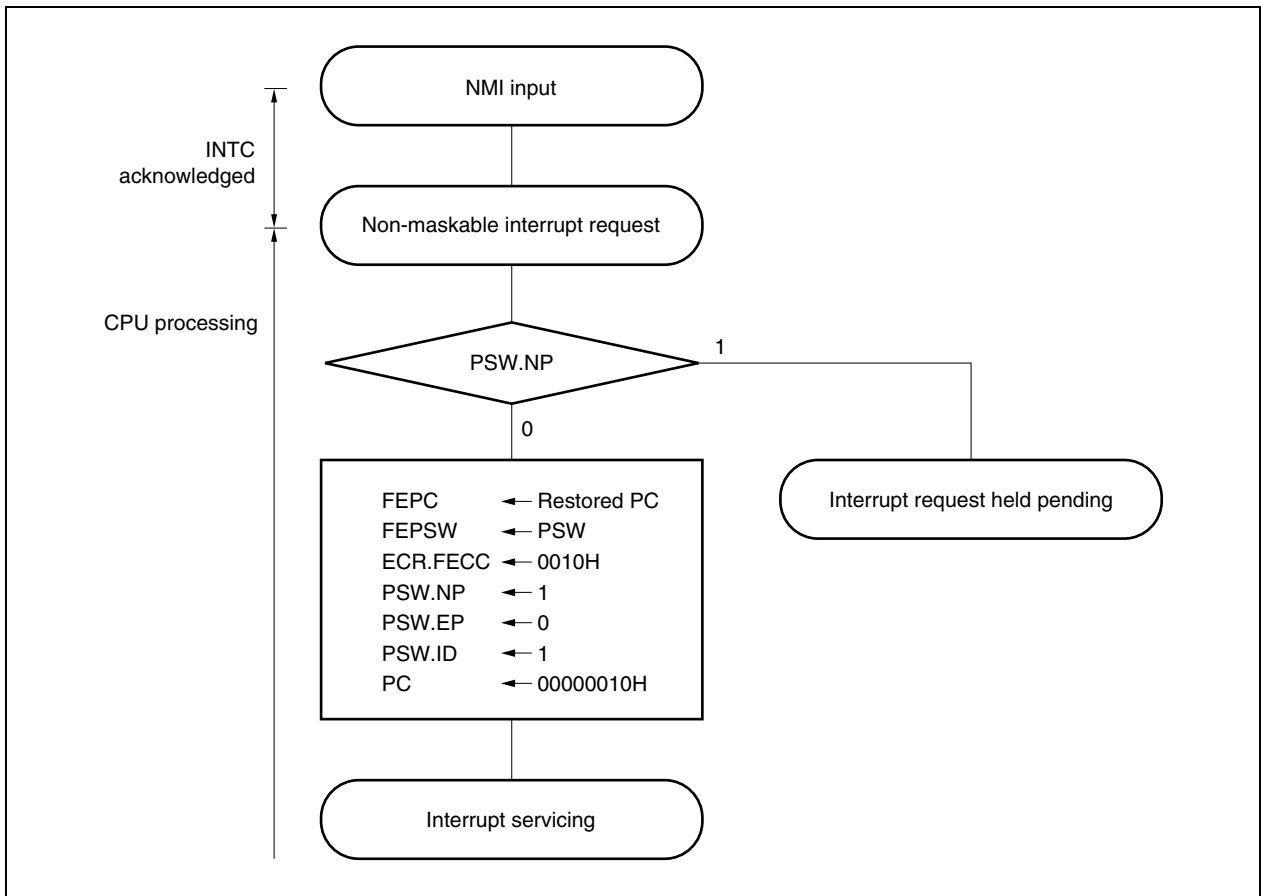
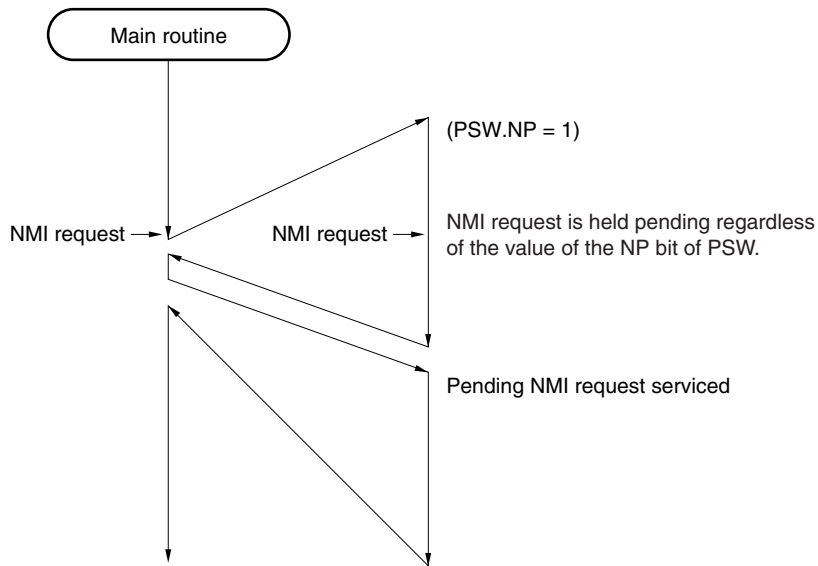
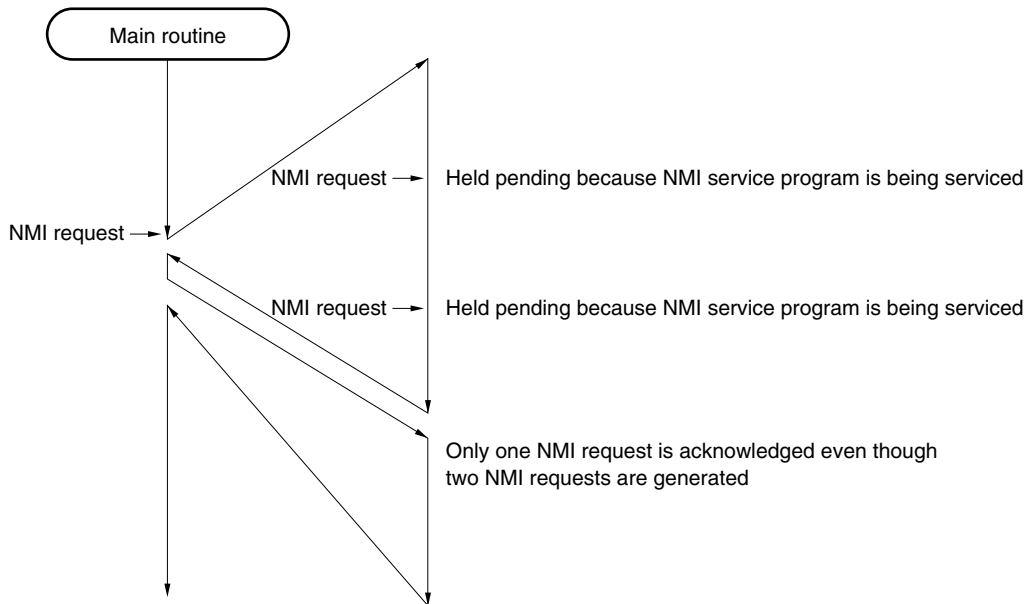


Figure 7-2. Acknowledging Non-Maskable Interrupt Request

(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

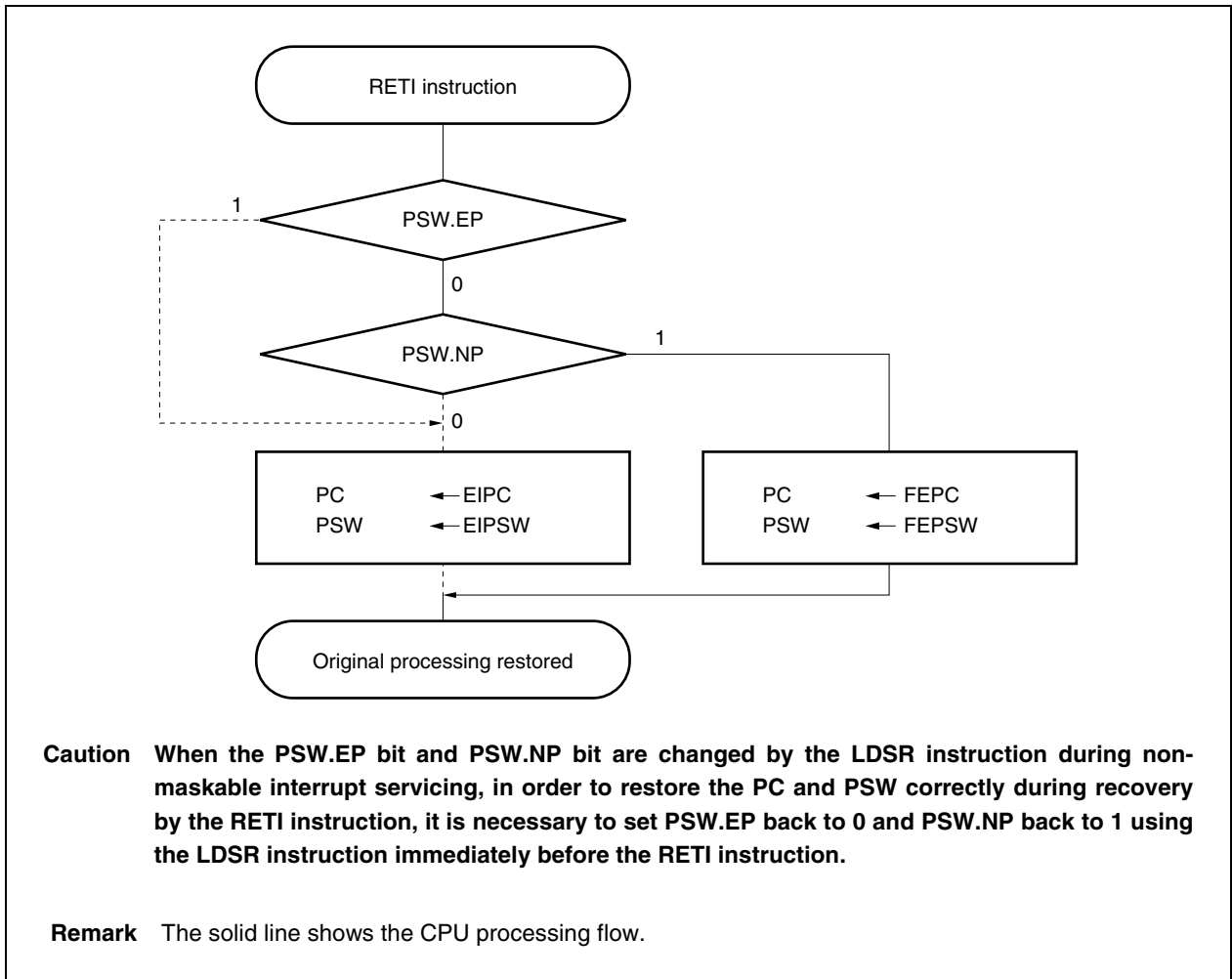
Execution is restored from the non-maskable interrupt servicing by the RETI instruction.

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



7.3 Maskable Interrupts

Maskable interrupt requests can be masked by interrupt control registers. The V850E/ME2 has 90 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 acknowledgment of other maskable interrupt requests is disabled and the interrupt disabled (DI) status is set.

When the EI instruction is executed in an interrupt service 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>.

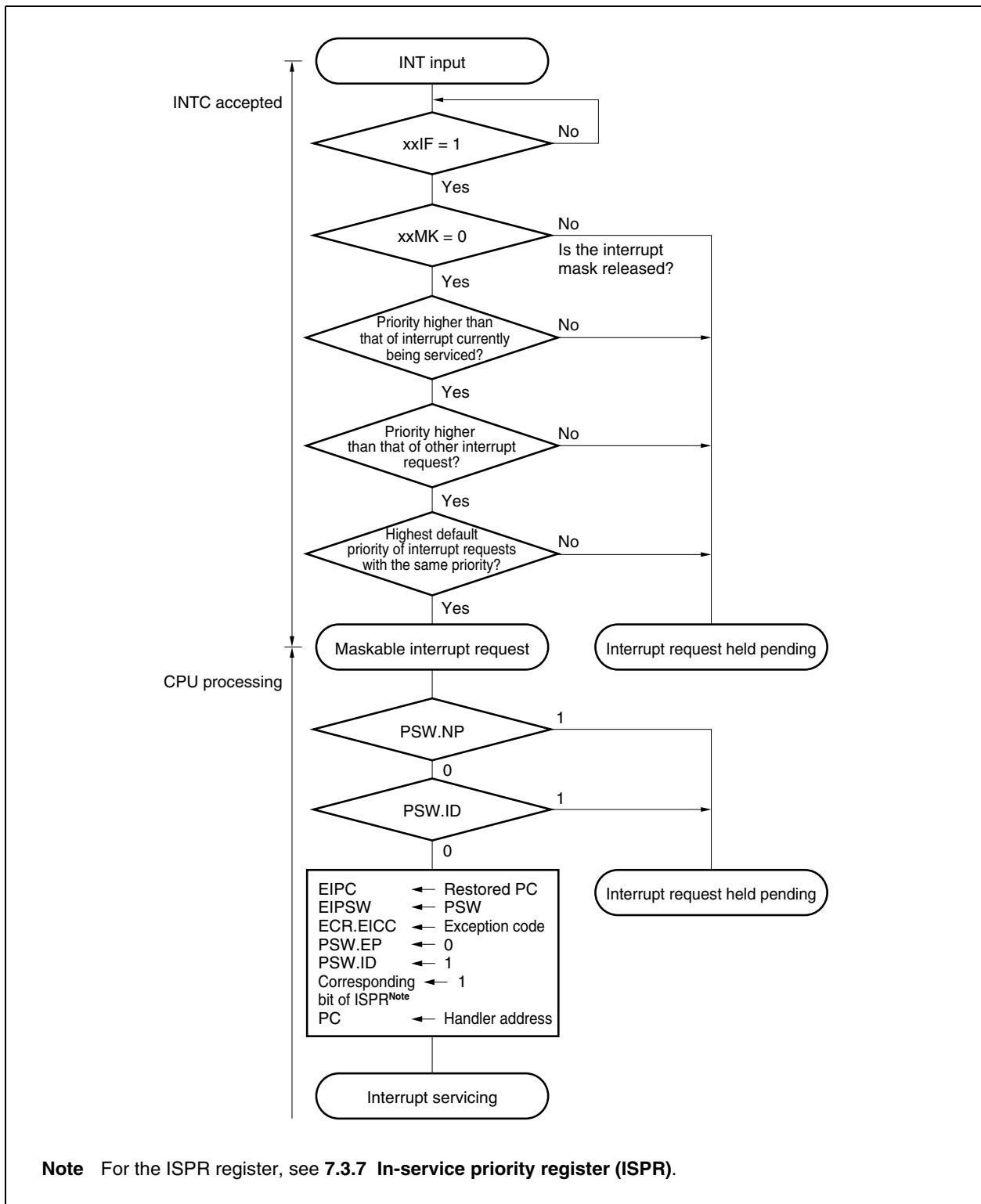
7.3.1 Operation

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 serviced (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

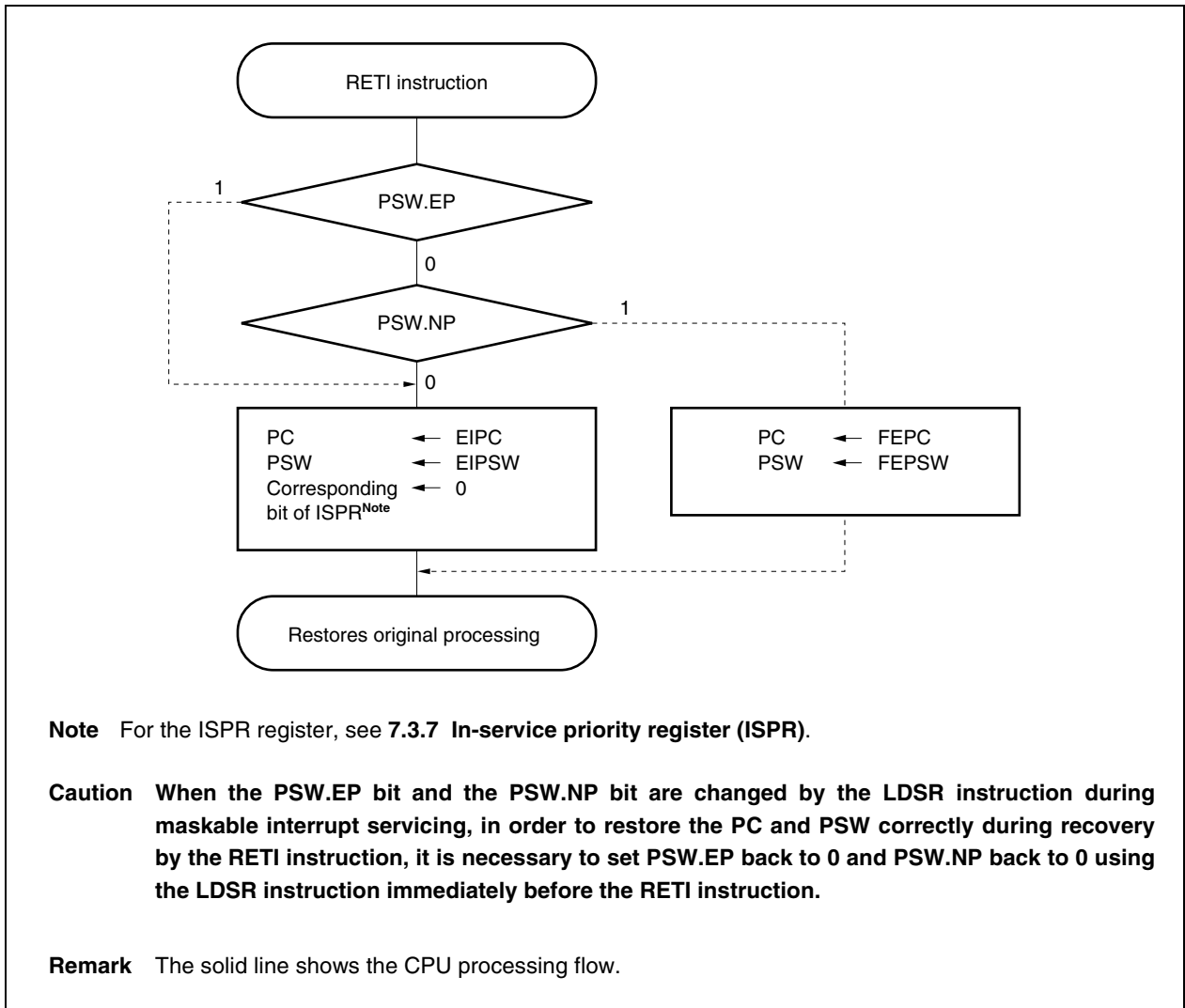
Recovery from maskable interrupt servicing is carried out by the RETI instruction.

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/ME2 provides multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced. Multiple interrupts can be controlled by priority levels.

There are two types of priority level control: control based on the default priority levels, and control based on the programmable priority levels 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 serviced in order depending on the priority level allocated to each interrupt request type (default priority level) beforehand. For more information, see **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 (see **Table 7-2**)
n: Peripheral unit number (see **Table 7-2**)

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

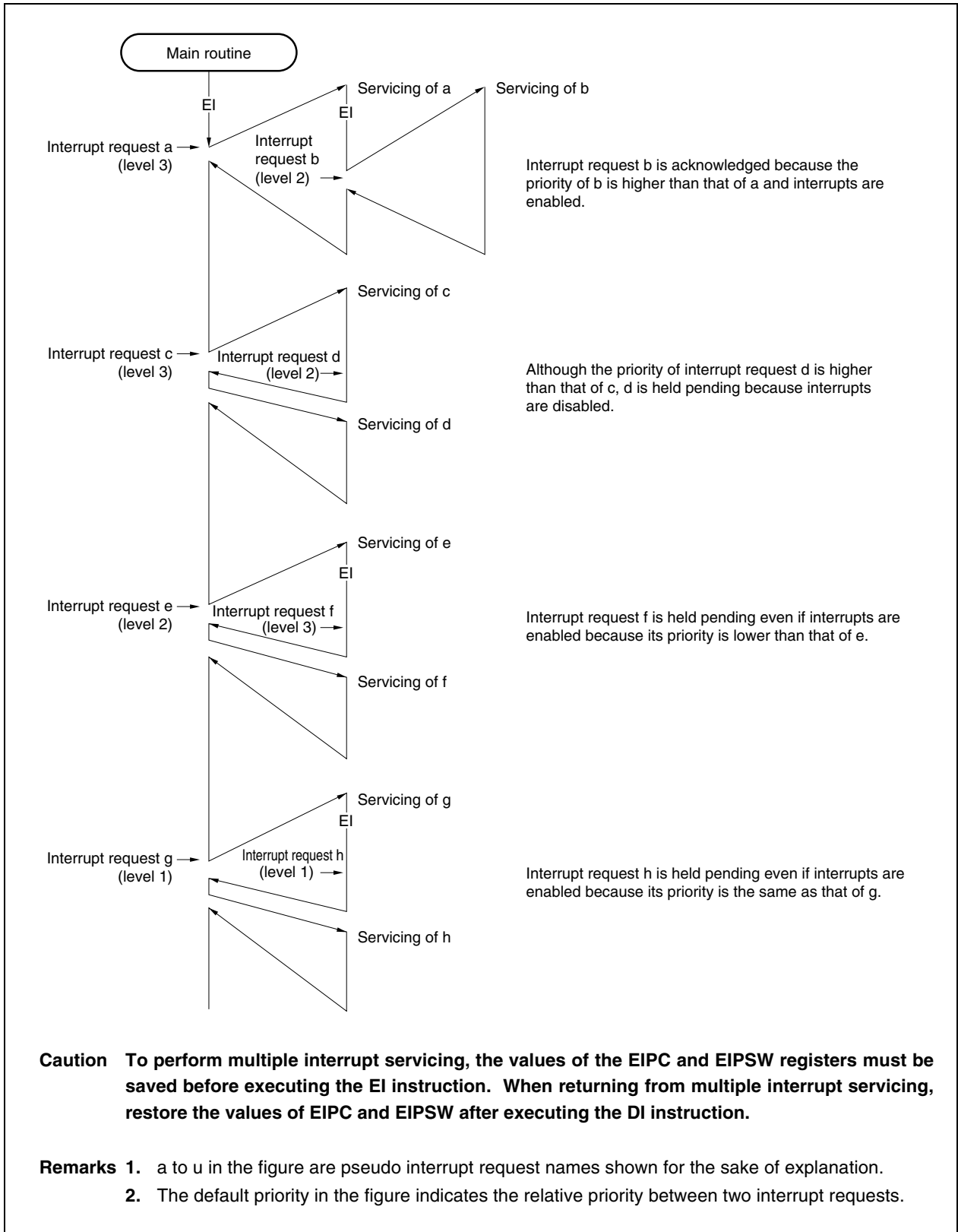
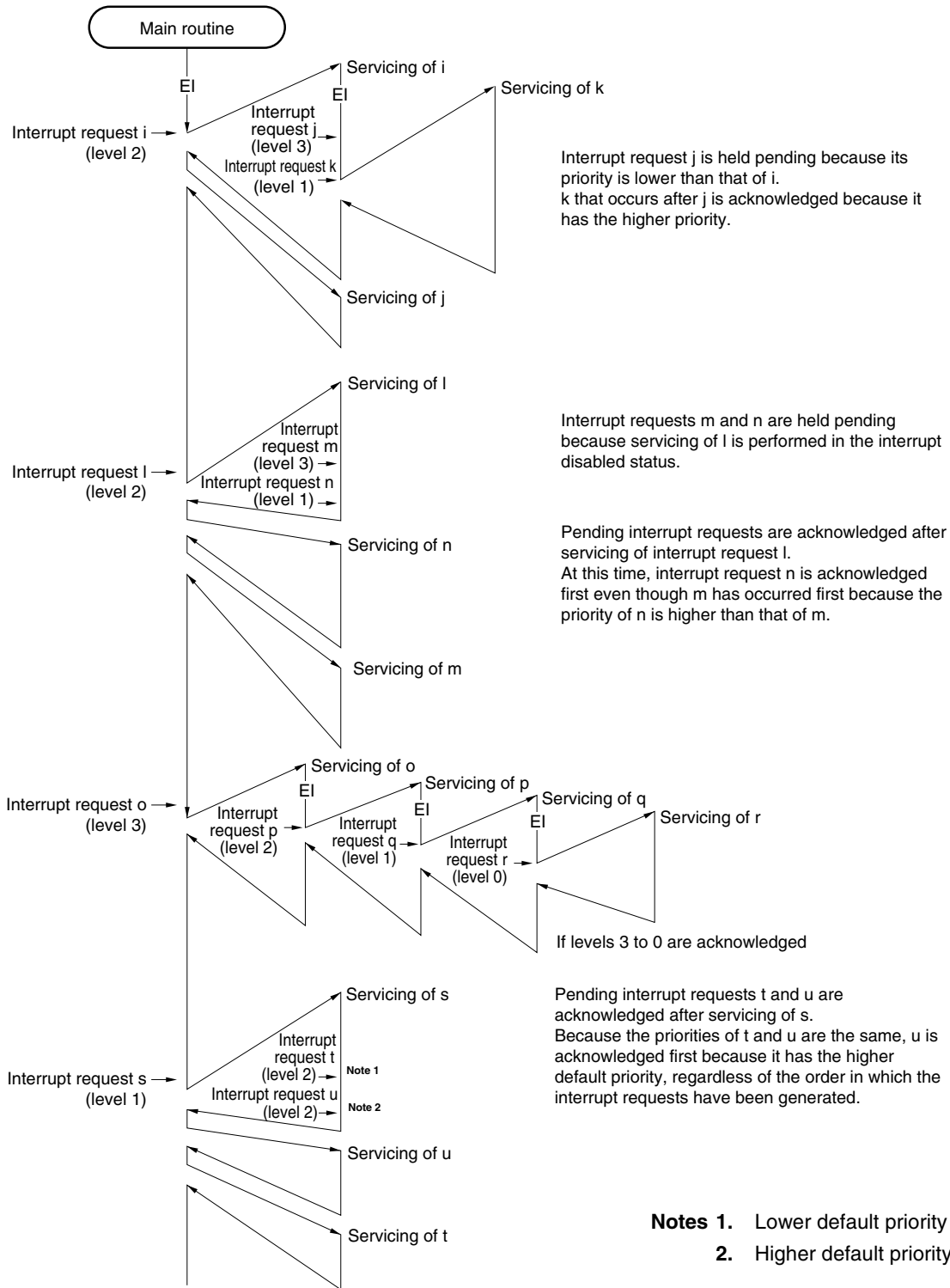
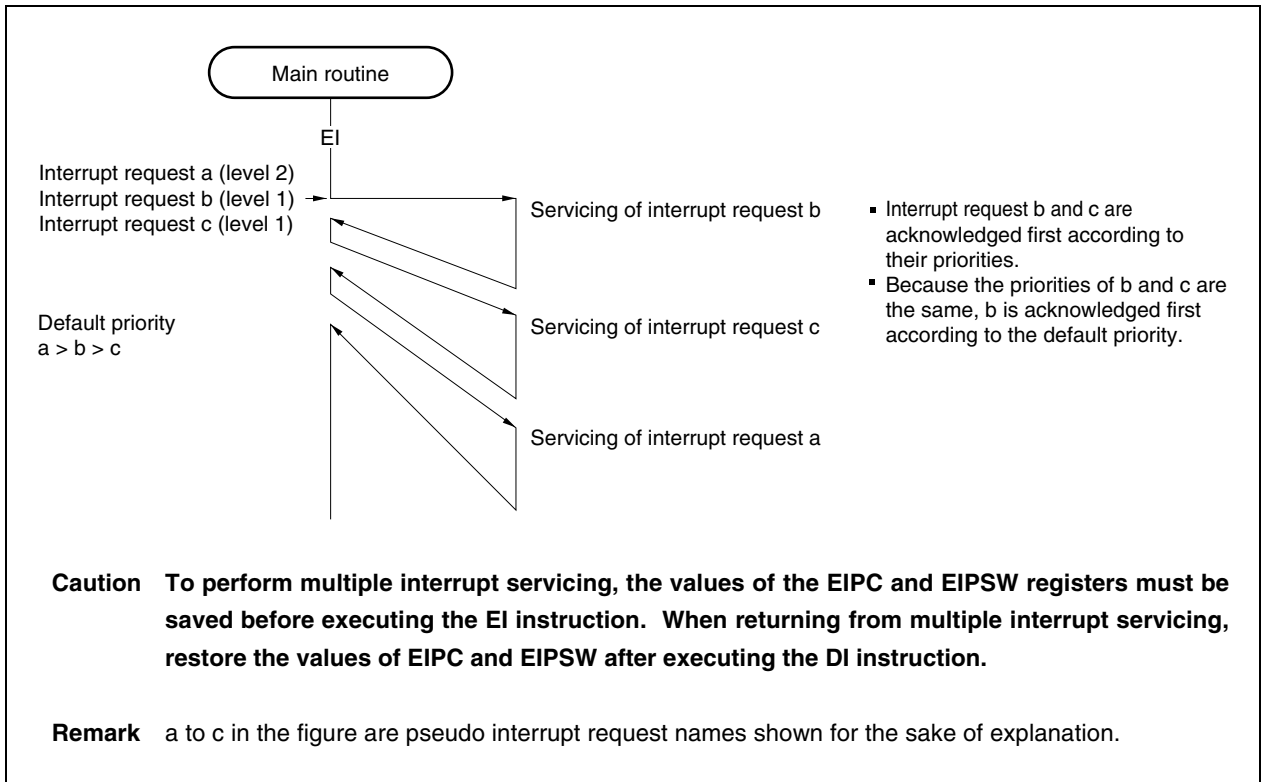


Figure 7-6. Example of Processing in Which Another Interrupt Request Is Issued While an Interrupt Is Being Serviced (2/2)



Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

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

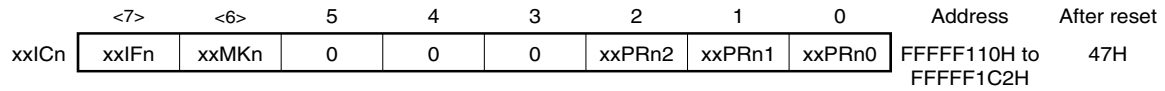


7.3.4 Interrupt control register (xxICn)

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

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

Caution Disable interrupts (DI) to read the xxIFn bit of the xxICn register. If the xxIFn bit is read while interrupts are enabled (EI), the correct value may not be read if there is a conflict between acknowledging an interrupt and reading the bit.



Bit position	Bit name	Function																																				
7	xxIFn	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	This is an interrupt mask flag. 0: Interrupt servicing enabled 1: Interrupt servicing disabled (pending)																																				
2 to 0	xxPRn2 to xxPRn0	8 levels of priority order are specified for each interrupt. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>xxPRn2</th> <th>xxPRn1</th> <th>xxPRn0</th> <th>Interrupt priority specification bit</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Specifies level 0 (highest).</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Specifies level 1.</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Specifies level 2.</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Specifies level 3.</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Specifies level 4.</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Specifies level 5.</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Specifies level 6.</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Specifies level 7 (lowest).</td> </tr> </tbody> </table>	xxPRn2	xxPRn1	xxPRn0	Interrupt priority specification bit	0	0	0	Specifies level 0 (highest).	0	0	1	Specifies level 1.	0	1	0	Specifies level 2.	0	1	1	Specifies level 3.	1	0	0	Specifies level 4.	1	0	1	Specifies level 5.	1	1	0	Specifies level 6.	1	1	1	Specifies level 7 (lowest).
xxPRn2	xxPRn1	xxPRn0	Interrupt priority specification bit																																			
0	0	0	Specifies level 0 (highest).																																			
0	0	1	Specifies level 1.																																			
0	1	0	Specifies level 2.																																			
0	1	1	Specifies level 3.																																			
1	0	0	Specifies level 4.																																			
1	0	1	Specifies level 5.																																			
1	1	0	Specifies level 6.																																			
1	1	1	Specifies level 7 (lowest).																																			

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

The addresses and bits of the interrupt control registers are as follows:

Table 7-2. Addresses and Bits of Interrupt Control Registers (1/3)

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF110H	P1IC0	P1IF0	P1MK0	0	0	0	P1PR02	P1PR01	P1PR00
FFFFF112H	P1IC1	P1IF1	P1MK1	0	0	0	P1PR12	P1PR11	P1PR10
FFFFF114H	P2IC1	P2IF1	P2MK1	0	0	0	P2PR12	P2PR11	P2PR10
FFFFF116H	P2IC2	P2IF2	P2MK2	0	0	0	P2PR22	P2PR21	P2PR20
FFFFF118H	P2IC3	P2IF3	P2MK3	0	0	0	P2PR32	P2PR31	P2PR30
FFFFF11AH	P2IC4	P2IF4	P2MK4	0	0	0	P2PR42	P2PR41	P2PR40
FFFFF11CH	P2IC5	P2IF5	P2MK5	0	0	0	P2PR52	P2PR51	P2PR50
FFFFF11EH	P5IC0	P5IF0	P5MK0	0	0	0	P5PR02	P5PR01	P5PR00
FFFFF120H	P5IC1	P5IF1	P5MK1	0	0	0	P5PR12	P5PR11	P5PR10
FFFFF122H	P5IC2	P5IF2	P5MK2	0	0	0	P5PR22	P5PR21	P5PR20
FFFFF124H	P6IC5	P6IF5	P6MK5	0	0	0	P6PR52	P6PR51	P6PR50
FFFFF126H	P6IC6	P6IF6	P6MK6	0	0	0	P6PR62	P6PR61	P6PR60
FFFFF128H	P6IC7	P6IF7	P6MK7	0	0	0	P6PR72	P6PR71	P6PR70
FFFFF12AH	PDIC0	PDIF0	PDMK0	0	0	0	PDPR02	PDPR01	PDPR00
FFFFF12CH	PDIC1	PDIF1	PDMK1	0	0	0	PDPR12	PDPR11	PDPR10
FFFFF12EH	PDIC2	PDIF2	PDMK2	0	0	0	PDPR22	PDPR21	PDPR20
FFFFF130H	PDIC3	PDIF3	PDMK3	0	0	0	PDPR32	PDPR31	PDPR30
FFFFF132H	PDIC4	PDIF4	PDMK4	0	0	0	PDPR42	PDPR41	PDPR40
FFFFF134H	PDIC5	PDIF5	PDMK5	0	0	0	PDPR52	PDPR51	PDPR50
FFFFF136H	PDIC6	PDIF6	PDMK6	0	0	0	PDPR62	PDPR61	PDPR60
FFFFF138H	PDIC7	PDIF7	PDMK7	0	0	0	PDPR72	PDPR71	PDPR70
FFFFF13AH	PDIC8	PDIF8	PDMK8	0	0	0	PDPR82	PDPR81	PDPR80
FFFFF13CH	PDIC9	PDIF9	PDMK9	0	0	0	PDPR92	PDPR91	PDPR90
FFFFF13EH	PDIC10	PDIF10	PDMK10	0	0	0	PDPR102	PDPR101	PDPR100
FFFFF140H	PDIC11	PDIF11	PDMK11	0	0	0	PDPR112	PDPR111	PDPR110
FFFFF142H	PDIC12	PDIF12	PDMK12	0	0	0	PDPR122	PDPR121	PDPR120
FFFFF144H	PDIC13	PDIF13	PDMK13	0	0	0	PDPR132	PDPR131	PDPR130
FFFFF146H	PDIC14	PDIF14	PDMK14	0	0	0	PDPR142	PDPR141	PDPR140
FFFFF148H	PDIC15	PDIF15	PDMK15	0	0	0	PDPR152	PDPR151	PDPR150
FFFFF14AH	PLIC0	PLIF0	PLMK0	0	0	0	PLPR02	PLPR01	PLPR00
FFFFF14CH	PLIC1	PLIF1	PLMK1	0	0	0	PLPR12	PLPR11	PLPR10
FFFFF14EH	OVCIC0	OVCIF0	OVCMK0	0	0	0	OVCPR02	OVCPR01	OVCPR00
FFFFF150H	OVCIC1	OVCIF1	OVCMK1	0	0	0	OVCPR12	OVCPR11	OVCPR10
FFFFF152H	OVCIC2	OVCIF2	OVCMK2	0	0	0	OVCPR22	OVCPR21	OVCPR20
FFFFF154H	OVCIC3	OVCIF3	OVCMK3	0	0	0	OVCPR32	OVCPR31	OVCPR30
FFFFF156H	OVCIC4	OVCIF4	OVCMK4	0	0	0	OVCPR42	OVCPR41	OVCPR40
FFFFF158H	OVCIC5	OVCIF5	OVCMK5	0	0	0	OVCPR52	OVCPR51	OVCPR50
FFFFF15AH	CCC0IC0	CCC0IF0	CCC0MK0	0	0	0	CCC0PR02	CCC0PR01	CCC0PR00
FFFFF15CH	CCC0IC1	CCC0IF1	CCC0MK1	0	0	0	CCC0PR12	CCC0PR11	CCC0PR10
FFFFF15EH	CCC1IC0	CCC1IF0	CCC1MK0	0	0	0	CCC1PR02	CCC1PR01	CCC1PR00
FFFFF160H	CCC1IC1	CCC1IF1	CCC1MK1	0	0	0	CCC1PR12	CCC1PR11	CCC1PR10

Table 7-2. Addresses and Bits of Interrupt Control Registers (2/3)

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF162H	CCC2IC0	CCC2IF0	CCC2MK0	0	0	0	CCC2PR02	CCC2PR01	CCC2PR00
FFFFF164H	CCC2IC1	CCC2IF1	CCC2MK1	0	0	0	CCC2PR12	CCC2PR11	CCC2PR10
FFFFF166H	CCC3IC0	CCC3IF0	CCC3MK0	0	0	0	CCC3PR02	CCC3PR01	CCC3PR00
FFFFF168H	CCC3IC1	CCC3IF1	CCC3MK1	0	0	0	CCC3PR12	CCC3PR11	CCC3PR10
FFFFF16AH	CCC4IC0	CCC4IF0	CCC4MK0	0	0	0	CCC4PR02	CCC4PR01	CCC4PR00
FFFFF16CH	CCC4IC1	CCC4IF1	CCC4MK1	0	0	0	CCC4PR12	CCC4PR11	CCC4PR10
FFFFF16EH	CCC5IC0	CCC5IF0	CCC5MK0	0	0	0	CCC5PR02	CCC5PR01	CCC5PR00
FFFFF170H	CCC5IC1	CCC5IF1	CCC5MK1	0	0	0	CCC5PR12	CCC5PR11	CCC5PR10
FFFFF172H	CMDIC0	CMDIF0	CMDMK0	0	0	0	CMDPR02	CMDPR01	CMDPR00
FFFFF174H	CMDIC1	CMDIF1	CMDMK1	0	0	0	CMDPR12	CMDPR11	CMDPR10
FFFFF176H	CMDIC2	CMDIF2	CMDMK2	0	0	0	CMDPR22	CMDPR21	CMDPR20
FFFFF178H	CMDIC3	CMDIF3	CMDMK3	0	0	0	CMDPR32	CMDPR31	CMDPR30
FFFFF17AH	CC10IC0	CC10IF0	CC10MK0	0	0	0	CC10PR02	CC10PR01	CC10PR00
FFFFF17CH	CC10IC1	CC10IF1	CC10MK1	0	0	0	CC10PR12	CC10PR11	CC10PR10
FFFFF17EH	CM10IC0	CM10IF0	CM10MK0	0	0	0	CM10PR02	CM10PR01	CM10PR00
FFFFF180H	CM10IC1	CM10IF1	CM10MK1	0	0	0	CM10PR12	CM10PR11	CM10PR10
FFFFF182H	OV1IC0	OV1IF0	OV1MK0	0	0	0	OV1PR02	OV1PR01	OV1PR00
FFFFF184H	UD1IC0	UD1IF0	UD1MK0	0	0	0	UD1PR02	UD1PR01	UD1PR00
FFFFF186H	CC11IC0	CC11IF0	CC11MK0	0	0	0	CC11PR02	CC11PR01	CC11PR00
FFFFF188H	CC11IC1	CC11IF1	CC11MK1	0	0	0	CC11PR12	CC11PR11	CC11PR10
FFFFF18AH	CM11IC0	CM11IF0	CM11MK0	0	0	0	CM11PR02	CM11PR01	CM11PR00
FFFFF18CH	CM11IC1	CM11IF1	CM11MK1	0	0	0	CM11PR12	CM11PR11	CM11PR10
FFFFF18EH	OV1IC1	OV1IF1	OV1MK1	0	0	0	OV1PR12	OV1PR11	OV1PR10
FFFFF190H	UD1IC1	UD1IF1	UD1MK1	0	0	0	UD1PR12	UD1PR11	UD1PR10
FFFFF192H	DMAIC0	DMAIF0	DMAMK0	0	0	0	DMAPR02	DMAPR01	DMAPR00
FFFFF194H	DMAIC1	DMAIF1	DMAMK1	0	0	0	DMAPR12	DMAPR11	DMAPR10
FFFFF196H	DMAIC2	DMAIF2	DMAMK2	0	0	0	DMAPR22	DMAPR21	DMAPR20
FFFFF198H	DMAIC3	DMAIF3	DMAMK3	0	0	0	DMAPR32	DMAPR31	DMAPR30
FFFFF19AH	CSI3IC0	CSI3IF0	CSI3MK0	0	0	0	CSI3PR02	CSI3PR01	CSI3PR00
FFFFF19CH	COVF3IC0	COVF3IF0	COVF3MK0	0	0	0	COVF3PR02	COVF3PR01	COVF3PR00
FFFFF19EH	CSI3IC1	CSI3IF1	CSI3MK1	0	0	0	CSI3PR12	CSI3PR11	CSI3PR10
FFFFF1A0H	COVF3IC1	COVF3IF1	COVF3MK1	0	0	0	COVF3PR12	COVF3PR11	COVF3PR10
FFFFF1A2H	UREIC0	UREIF0	UREMK0	0	0	0	UREPR02	UREPR01	UREPR00
FFFFF1A4H	URIC0	URIF0	URMK0	0	0	0	URPR02	URPR01	URPR00
FFFFF1A6H	UTIC0	UTIF0	UTMK0	0	0	0	UTPR02	UTPR01	UTPR00
FFFFF1A8H	UIFIC0	UIFIF0	UIFMK0	0	0	0	UIFPR02	UIFPR01	UIFPR00
FFFFF1AAH	UTOIC0	UTOIF0	UTOMK0	0	0	0	UTOPR02	UTOPR01	UTOPR00
FFFFF1ACH	UREIC1	UREIF1	UREMK1	0	0	0	UREPR12	UREPR11	UREPR10
FFFFF1AEH	URIC1	URIF1	URMK1	0	0	0	URPR12	URPR11	URPR10
FFFFF1B0H	UTIC1	UTIF1	UTMK1	0	0	0	UTPR12	UTPR11	UTPR10
FFFFF1B2H	UIFIC1	UIFIF1	UIFMK1	0	0	0	UIFPR12	UIFPR11	UIFPR10

Table 7-2. Addresses and Bits of Interrupt Control Registers (3/3)

Address	Register	Bit							
		<7>	<6>	5	4	3	2	1	0
FFFFF1B4H	UTOIC1	UTOIF1	UTOMK1	0	0	0	UTOPR12	UTOPR11	UTOPR10
FFFFF1B6H	ADIC	ADIF	ADMK	0	0	0	ADPR2	ADPR1	ADPR0
FFFFF1B8H	US0BIC	US0BIF	US0BMK	0	0	0	US0BPR2	US0BPR1	US0BPR0
FFFFF1BAH	US1BIC	US1BIF	US1BMK	0	0	0	US1BPR2	US1BPR1	US1BPR0
FFFFF1BCH	US2BIC	US2BIF	US2BMK	0	0	0	US2BPR2	US2BPR1	US2BPR0
FFFFF1BEH	USP2IC	USP2IF	USP2MK	0	0	0	USP2PR2	USP2PR1	USP2PR0
FFFFF1C0H	USP4IC	USP4IF	USP4MK	0	0	0	USP4PR2	USP4PR1	USP4PR0
FFFFF1C2H	RSUMIC	RSUMIF	RSUMMK	0	0	0	RSUMPR2	RSUMPR1	RSUMPR0

7.3.5 Interrupt mask registers 0 to 5 (IMR0 to IMR5)

These registers set the interrupt mask state for the maskable interrupts. The xxMKn bit of the IMR0 to IMR5 registers is equivalent to the xxMKn bit of the xxICn register.

The IMRm register (m = 0 to 5) can be read or 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 or written in 8-bit or 1-bit units.

Bits 15 to 10 of the IMR5 register (bits 7 to 2 of the IMR5H 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	After reset
	PDMK2	PDMK1	PDMK0	P6MK7	P6MK6	P6MK5	P5MK2	P5MK1		
	7	6	5	4	3	2	1	0		
	P5MK0	P2MK5	P2MK4	P2MK3	P2MK2	P2MK1	P1MK1	P1MK0		
IMR1	15	14	13	12	11	10	9	8	Address	After reset
	OVCMK0	PLMK1	PLMK0	PDMK15	PDMK14	PDMK13	PDMK12	PDMK11		
	7	6	5	4	3	2	1	0		
	PDMK10	PDMK9	PDMK8	PDMK7	PDMK6	PDMK5	PDMK4	PDMK3		
IMR2	15	14	13	12	11	10	9	8	Address	After reset
	CCC5MK0	CCC4MK1	CCC4MK0	CCC3MK1	CCC3MK0	CCC2MK1	CCC2MK0	CCC1MK1		
	7	6	5	4	3	2	1	0		
	CCC1MK0	CCC0MK1	CCC0MK0	OVCMK5	OVCMK4	OVCMK3	OVCMK2	OVCMK1		
IMR3	15	14	13	12	11	10	9	8	Address	After reset
	OV1MK1	CM11MK1	CM11MK0	CC11MK1	CC11MK0	UD1MK0	OV1MK0	CM10MK1		
	7	6	5	4	3	2	1	0		
	CM10MK0	CC10MK1	CC10MK0	CMDMK3	CMDMK2	CMDMK1	CMDMK0	CCC5MK1		
IMR4	15	14	13	12	11	10	9	8	Address	After reset
	URMK1	UREMK1	UTOMK0	UIFMK0	UTMK0	URMK0	UREMK0	COVF3MK1		
	7	6	5	4	3	2	1	0		
	CSI3MK1	COVF3MK0	CSI3MK0	DMAMK3	DMAMK2	DMAMK1	DMAMK0	UD1MK1		
IMR5	15	14	13	12	11	10	9	8	Address	After reset
	1	1	1	1	1	1	RSUMMK	USP4MK		
	7	6	5	4	3	2	1	0		
	USP2MK	US2BMK	US1BMK	US0BMK	ADMK	UTOMK1	UIFMK1	UTMK1		

Bit position	Bit name	Function
15 to 0	xxMKn	Interrupt mask flag 0: Interrupt servicing enabled 1: Interrupt servicing disabled (pending)

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

7.3.6 NMI reset status register (NRS)

This register holds the valid edge input status of the NMI after the reset signal has been cleared.

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

- Cautions**
- 1. The NMIRS bit of the NRS register is set to 1 if an NMI occurs after the internal instruction RAM has been set in the read mode (IRAMM0 bit of IRAMM register = 0). In this case, the status of the NMIRS bit does not have to be checked because execution automatically branches to the NMI servicing routine.**
 - 2. The mask function of NMI input is valid after the reset signal has been cleared and before the internal instruction RAM is set in the read mode (IRAMM0 bit of IRAMM register = 0).**

	7	6	5	4	3	2	1	0		
NRS	0	0	0	0	0	0	0	NMIRS	Address	After reset
									FFFFF80CH	00H

Bit position	Bit name	Function
0	NMIRS	Indicates the valid edge input status of NMI. 0: No NMI input 1: NMI input

With the V850E/ME2, the NMI is masked after the reset signal has been cleared. It is unmasked if the mode of the internal instruction RAM is changed from the write mode (IRAMM0 bit of IRAMM register = 1) to the read mode (IRAMM0 bit of IRAMM register = 0) after the program has been downloaded to internal instruction RAM bank 0. (The NRS register is controlled to prohibit a jump to the interrupt vector before the program is stored in the interrupt vector table.)

If the NMI valid edge is input while the NMI is masked, the NMI input can be recognized by reading the NRS register.

7.3.9 Selecting interrupt trigger mode

The valid edge of the $\overline{\text{INTPn}}$, INTPCm , TIC0 to TIC3, INTPx , TIUD10, TIUD11, TCUD10, TCUD11, TCLR10, and TCLR11 pins can be selected by program ($n = 10, 11, 21$ to 25, 50 to 52, 65 to 67, L0, L1, or D0 to D15, $m = 00, 01, 10, 11, 20, 21, 30,$ or 31, $x = 100, 101, 110,$ or 111). A trigger level can also be selected for the $\overline{\text{INTPn}}$ pin. The following valid edges can be selected.

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

The INTPn , INTPCm , TIC0 to TIC3, INTPx , TIUD10, TIUD11, TCUD10, TCUD11, TCLR10, and TCLR11 signals detected by the edge are used as interrupt sources, to input a capture trigger, or to input an external count signal to the timers.

The valid edge of these signals is specified by external interrupt rising edge specification registers 1, 2, 5, 6, AL, and DH (INTR1, INTR2, INTR5, INTR6, INTRAL, and INTRDH), external interrupt falling edge specification registers 1, 2, 5, 6, AL, and DH (INTF1, INTF2, INTF5, INTF6, INTFAL, and INTFDH), valid edge select registers C0 to C3 (SESC0 to SESC3), and valid edge select registers 10 and 11 (SESA10 and SESA11). The trigger level is specified by external interrupt rising edge specification registers 1, 2, 5, 6, AL, and DH (INTR1, INTR2, INTR5, INTR6, INTRAL, and INTRDH) and external interrupt falling edge specification registers 1, 2, 5, 6, AL, and DH (INTF1, INTF2, INTF5, INTF6, INTFAL, and INTFDH).

(1) External interrupt rising edge specification register 1 (INTR1), external interrupt falling edge specification register 1 (INTF1)

These registers are used to specify the trigger mode of the external interrupt requests (INTP10 and INTP11) input from external pins. The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTF10 and INTR10 bits: INTP10
- INTF11 and INTR11 bits: INTP11

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

Each of these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode of the $\overline{\text{INTP10}}$ and $\overline{\text{INTP11}}$ pins, set the PMC1 register. If the PMC1 register is set after the INTR1 and INTF1 registers have been set, an illegal interrupt may occur when the PMC1 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR1	0	0	0	0	0	0	INTR11	INTR10	FFFFFFC22H	03H
INTF1	0	0	0	0	0	0	INTF11	INTF10	FFFFFFC02H	00H

Bit position	Bit name	Function															
1, 0	INTF1n, INTR1n (n = 0, 1)	Specify the trigger mode of the $\overline{\text{INTP10}}$ and $\overline{\text{INTP11}}$ pins. <table border="1"> <thead> <tr> <th>INTF1n</th> <th>INTR1n</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>	INTF1n	INTR1n	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
INTF1n	INTR1n	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 $\overline{\text{INTP1n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P1IFn bit (n = 0, 1). Consequently, even when the CPU acknowledges the interrupt and the P1IFn bit of the interrupt control register (P1ICn) is automatically cleared to 0, the P1IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP1n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P1IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP1n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP1n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP1n) is held pending (n = 0, 1). To not acknowledge the interrupt request of INTP1n, clear the P1IFn bit of the interrupt control register.

(2) External interrupt rising edge specification register 2 (INTR2), external interrupt falling edge specification register 2 (INTF2)

These registers are used to specify the trigger mode of the external interrupt requests (INTP2n) input from external pins and the non-maskable interrupt (NMI) (n = 1 to 5). For the trigger mode of the NMI, see 7.2.4

(1) External interrupt rising edge specification register 2 (INTR2), external interrupt falling edge specification register 2 (INTF2).

The correspondence between each bit and the external interrupt request and non-maskable interrupt that are controlled by that bit is as follows.

- INTF21 and INTR21 bits: INTP21
- INTF22 and INTR22 bits: INTP22
- INTF23 and INTR23 bits: INTP23
- INTF24 and INTR24 bits: INTP24
- INTF25 and INTR25 bits: INTP25

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge of the $\overline{\text{INTP2n}}$ pin.

Each of these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode of the $\overline{\text{INTP2n}}$ pin, set the PMC2 register (n = 1 to 5). If the PMC2 register is set after the INTR2 and INTF2 registers have been set, an illegal interrupt may occur when the PMC2 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR2	0	0	INTR25	INTR24	INTR23	INTR22	INTR21	NMIRO	FFFFFC24H	3FH
INTF2	0	0	INTF25	INTF24	INTF23	INTF22	INTF21	NMIF0	FFFFFC04H	00H

Bit position	Bit name	Function															
5 to 1	INTF2n, INTR2n (n = 1 to 5)	Specify the trigger mode of the $\overline{\text{INTP2n}}$ pin. <table border="1"> <thead> <tr> <th>INTF2n</th> <th>INTR2n</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>	INTF2n	INTR2n	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
INTF2n	INTR2n	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 $\overline{\text{INTP2n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P2IFn bit (n = 1 to 5). Consequently, even when the CPU acknowledges the interrupt and the P2IFn bit of the interrupt control register (P2ICn) is automatically cleared to 0, the P2IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP2n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P2IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP2n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP2n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP2n) is held pending (n = 1 to 5). To not acknowledge the interrupt request of INTP2n, clear the P2IFn bit of the interrupt control register.

Remark For the bit 0 (NMIRO) of the INTR2 register and bit 0 (NMIF0) of the INTF2 register, see **7.2.4 (1) External interrupt rising edge specification register 2 (INTR2), external interrupt falling edge specification register 2 (INTF2)**.

(3) External interrupt rising edge specification register 5 (INTR5), external interrupt falling edge specification register 5 (INTF5)

These registers are used to specify the trigger mode of the external interrupt requests (INTP5n) input from external pins (n = 0 to 2). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTF50 and INTR50 bits: INTP50
- INTF51 and INTR51 bits: INTP51
- INTF52 and INTR52 bits: INTP52

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

Each of these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMC5 register.

If the PMC5 register is set after the INTR5 and INTF5 registers have been set, an illegal interrupt may occur when the PMC5 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR5	0	0	0	0	0	INTR52	INTR51	INTR50	FFFFFFC2AH	07H
INTF5	0	0	0	0	0	INTF52	INTF51	INTF50	FFFFFFC0AH	00H

Bit position	Bit name	Function															
2 to 0	INTF5n, INTR5n (n = 0 to 2)	Specify the trigger mode of the $\overline{\text{INTP5n}}$ pin. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>INTF5n</th> <th>INTR5n</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>	INTF5n	INTR5n	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
INTF5n	INTR5n	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 $\overline{\text{INTP5n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P5IFn bit (n = 0 to 2). Consequently, even when the CPU acknowledges the interrupt and the P5IFn bit of the interrupt control register (P5ICn) is automatically cleared to 0, the P5IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP5n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P5IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP5n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP5n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP5n) is held pending (n = 0 to 2). To not acknowledge the interrupt request of INTP5n, clear the P5IFn bit of the interrupt control register.

(4) External interrupt rising edge specification register 6 (INTR6), external interrupt falling edge specification register 6 (INTF6)

These registers are used to specify the trigger mode of the external interrupt requests (INTP6n) input from external pins (n = 5 to 7). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTF65 and INTR65 bits: INTP65
- INTF66 and INTR66 bits: INTP66
- INTF67 and INTR67 bits: INTP67

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

Each of these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMC6 register.

If the PMC6 register is set after the INTR6 and INTF6 registers have been set, an illegal interrupt may occur when the PMC6 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR6	INTR67	INTR66	INTR65	0	0	0	0	0	FFFFFFC2CH	E0H
INTF6	INTF67	INTF66	INTF65	0	0	0	0	0	FFFFFFC0CH	00H

Bit position	Bit name	Function															
7 to 5	INTF6n, INTR6n (n = 7 to 5)	Specify the trigger mode of the $\overline{\text{INTP6n}}$ pin. <table border="1"> <thead> <tr> <th>INTF6n</th> <th>INTR6n</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>	INTF6n	INTR6n	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
INTF6n	INTR6n	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 $\overline{\text{INTP6n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P6IFn bit (n = 5 to 7). Consequently, even when the CPU acknowledges the interrupt and the P6IFn bit of the interrupt control register (P6ICn) is automatically cleared to 0, the P6IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP6n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P6IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP6n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP6n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP6n) is held pending (n = 5 to 7). To not acknowledge the interrupt request of INTP6n, clear the P6IFn bit of the interrupt control register.

(5) External interrupt rising edge specification register AL (INTRAL), external interrupt falling edge specification register AL (INTFAL)

These registers are used to specify the trigger mode of the external interrupt requests (INTPLn) input from external pins (n = 0, 1). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTFAL0 and INTRAL0 bits: INTPL0
- INTFAL1 and INTRAL1 bits: INTPL1

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

Each of these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMCAL register.

If the PMCAL register is set after the INTRAL and INTFAL registers have been set, an illegal interrupt may occur when the PMCAL register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTRAL	0	0	0	0	0	0	INTRAL1	INTRAL0	FFFFFC30H	03H
INTFAL	0	0	0	0	0	0	INTFAL1	INTFAL0	FFFFFC10H	00H

Bit position	Bit name	Function															
1, 0	INTFALn, INTRALn (n = 0, 1)	Specify the trigger mode of the $\overline{\text{INTPLn}}$ pin. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 15%;">INTFALn</th> <th style="width: 15%;">INTRALn</th> <th style="width: 70%;">Operation</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Falling edge</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Rising edge</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Level detection (low-level detection)^{Notes 1, 2}</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	INTFALn	INTRALn	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
INTFALn	INTRALn	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 $\overline{\text{INTPLn}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the PLIFn bit (n = 0, 1). Consequently, even when the CPU acknowledges the interrupt and the PLIFn bit of the interrupt control register (PLICn) is automatically cleared to 0, the PLIFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTPLn}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the PLIFn bit to 0.

2. If a level-detected interrupt request (INTPLn) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTPLn) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTPLn) is held pending (n = 0, 1). To not acknowledge the interrupt request of INTPLn, clear the PLIFn bit of the interrupt control register.

(6) External interrupt rising edge specification register DH (INTRDH), external interrupt falling edge specification register DH (INTFDH)

These registers are used to specify the trigger mode of the external interrupt requests (INTPD_n) input from external pins (n = 0 to 15). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTFDH0 and INTRDH0 bits: INTPD0
- INTFDH1 and INTRDH1 bits: INTPD1
- INTFDH2 and INTRDH2 bits: INTPD2
- INTFDH3 and INTRDH3 bits: INTPD3
- INTFDH4 and INTRDH4 bits: INTPD4
- INTFDH5 and INTRDH5 bits: INTPD5
- INTFDH6 and INTRDH6 bits: INTPD6
- INTFDH7 and INTRDH7 bits: INTPD7
- INTFDH8 and INTRDH8 bits: INTPD8
- INTFDH9 and INTRDH9 bits: INTPD9
- INTFDH10 and INTRDH10 bits: INTPD10
- INTFDH11 and INTRDH11 bits: INTPD11
- INTFDH12 and INTRDH12 bits: INTPD12
- INTFDH13 and INTRDH13 bits: INTPD13
- INTFDH14 and INTRDH14 bits: INTPD14
- INTFDH15 and INTRDH15 bits: INTPD15

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

The INTRDH and INTFDH registers can be read or written in 16-bit units.

If the higher 8 bits of the INTRDH and INTFDH registers are used as the INTRDHH and INTFDHH registers, and the lower 8 bits as the INTRDHL and INTFDHL registers, these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMCDH register.

If the PMCDH register is set after the INTRDH and INTFDH registers have been set, an illegal interrupt may occur when the PMCDH register is set.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
INTRDH	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	FFFFFC36H	FFFFH
	DH15	DH14	DH13	DH12	DH11	DH10	DH9	DH8	DH7	DH6	DH5	DH4	DH3	DH2	DH1	DH0		
INTFDH	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	FFFFFC16H	0000H
	DH15	DH14	DH13	DH12	DH11	DH10	DH9	DH8	DH7	DH6	DH5	DH4	DH3	DH2	DH1	DH0		

Bit position	Bit name	Function															
15 to 0	INTFDHn, INTRDHn (n = 0 to 15)	Specify the trigger mode of the $\overline{\text{INTPDn}}$ pin. <table border="1"> <thead> <tr> <th>INTFDHn</th> <th>INTRDHn</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>	INTFDHn	INTRDHn	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
INTFDHn	INTRDHn	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 $\overline{\text{INTPDn}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the PDIFn bit (n = 0 to 15). Consequently, even when the CPU acknowledges the interrupt and the PDIFn bit of the interrupt control register (PDICn) is automatically cleared to 0, the PDIFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTPDn}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the PDIFn bit to 0.
- 2.** If a level-detected interrupt request (INTPDn) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTPDn) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTPDn) is held pending (n = 0 to 15). To not acknowledge the interrupt request of INTPDn, clear the PDIFn bit of the interrupt control register.

(7) Valid edge select registers C0 to C3 (SESC0 to SESC3)

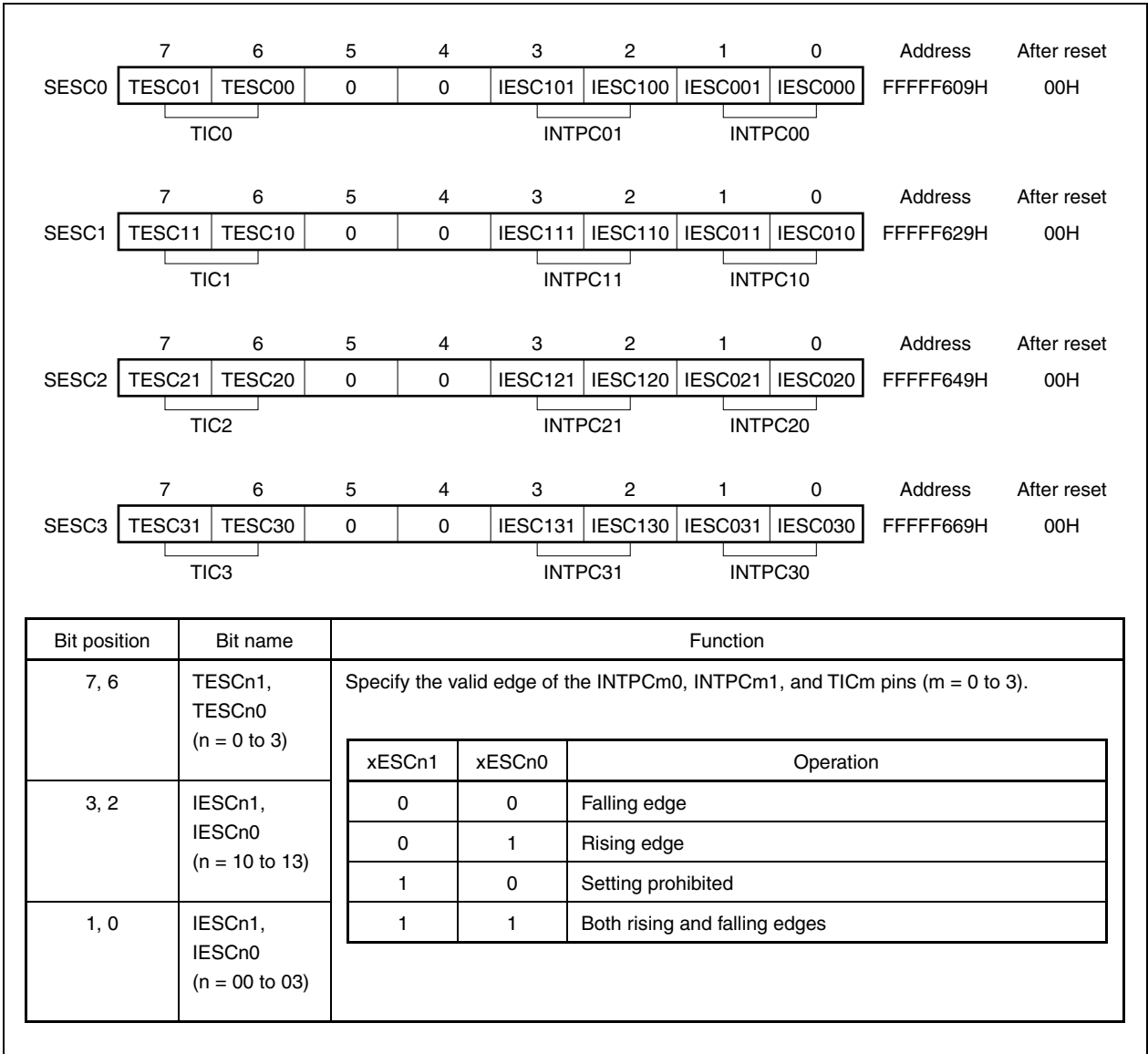
These registers are used to specify the valid edge of the external interrupt requests (INTPC00, INTPC01, INTPC10, INTPC11, INTPC20, INTPC21, INTPC30, INTPC31, or TIC0 to TIC3) input from external pins to TMCn (n = 0 to 3).

The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

Each of these registers can be read or written in 8-bit units.

Be sure to clear bits 5 and 4 to 0. If they are set to 1, the operation is not guaranteed.

- Cautions**
- 1. Do not change each bit of the SESCn register while the timer is operating (n = 0 to 3). To change a bit, clear the CECn bit of the TMCCn0 register to 0. The operation is not guaranteed if the SESCn register is rewritten during timer operation.**
 - 2. Even when the INTPC00/TIC0, INTPC10/TIC1, INTPC20/TIC2, and INTPC30/TIC3 pins are used as INTPC00, INTPC10, INTPC20, and INTPC30, respectively, without using timer C, be sure to set the CAECn and CECn bits of timer mode control registers C00 to C03 (TMCC00 to TMCC30) to 1.**
 - 3. Before setting the trigger mode of the INTPC00, INTPC01, INTPC10, INTPC11, INTPC20, INTPC21, INTPC30, INTPC31, and TIC0 to TIC3 pins, set the PMCx register (x = 5 to 7). Then set the CAECn and CECn bits of the TMCCn0 register to 1 (n = 0 to 3). If the PMCx register is set after the SESCn register has been set, an illegal interrupt, incorrect count, or incorrect clear may occur depending on the timing of setting the PMCx register (n = 0 to 3, x = 5 to 7).**



(8) Valid edge select registers 10, 11 (SESA10, SESA11)

These registers are used to specify the valid edge of the external interrupt requests (TIUD10, TIUD11, TCUD10, TCUD11, TCLR10, or TCLR11) input from external pins and the external capture trigger inputs to timer ENC1 (INTP100, INTP101, INTP110, or INTP111).

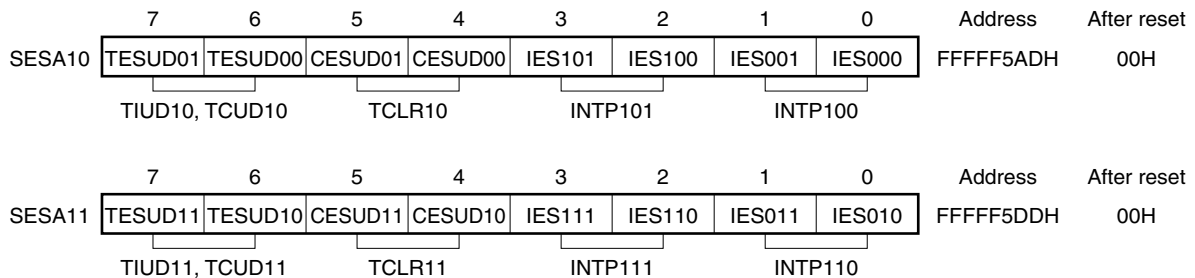
The rising edge, falling edge, or both the rising and falling edges can be independently specified as the valid edge.

Each of these registers can be read or written in 8-bit or 1-bit units.

Cautions 1. Changing each bit of the SESA1n register is prohibited while TMENC1n is operating (CE1n1 bit = 1).

2. Before setting the trigger mode of the INTP100, INTP101, INTP110, INTP111, TIUD10, TIUD11, TCUD10, TCUD11, TCLR10, and TCLR11 pins, set the PMCDH register. If the PMCDH register is set after the SESA1n register has been set, an illegal interrupt, incorrect count, or incorrect clear may occur depending on the timing of setting the PMCDH register.

(1/2)



Bit position	Bit name	Function															
7, 6	TESUDn1, TESUDn0	Specify the valid edge of the TIUD1n and TCUD1n pins. <table border="1" style="margin-top: 5px;"> <thead> <tr> <th>TESUDn1</th> <th>TESUDn0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table> <p>Cautions 1. The set values of the TESUDn1 and TESUDn0 bits are valid only in UDC mode A and UDC mode B.</p> <p>2. If mode 4 is specified for the operation of TMENC1n (specified by the PRM1n2 to PRM1n0 bits of the PRM1n register), specifying the valid edge for the TIUD1n and TCUD1n pins (TESUDn1 and TESUDn0 bits) is invalid.</p>	TESUDn1	TESUDn0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
TESUDn1	TESUDn0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															

Remark n = 0, 1

Bit position	Bit name	Function															
5, 4	CESUDn1, CESUDn0	<p>Specify the valid edge of the TCLR1n pin.</p> <table border="1"> <thead> <tr> <th>CESUDn1</th> <th>CESUDn0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Low-level</td> </tr> <tr> <td>1</td> <td>1</td> <td>High-level</td> </tr> </tbody> </table> <p>The relationship between the set values of the CESUDn1 and CESUDn0 bits and the operation of TMENC1n is as follows. 00: TMENC1n cleared after the rising edge of TCLR1n has been detected. 01: TMENC1n cleared after the falling edge of TCLR1n has been detected. 10: TMENC1n stays cleared while TCLR1n input is low. 11: TMENC1n stays cleared while TCLR1n input is high.</p> <p>Caution The set values of the CESUDn1 and CESUDn0 bits are valid only in UDC mode A.</p>	CESUDn1	CESUDn0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Low-level	1	1	High-level
CESUDn1	CESUDn0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Low-level															
1	1	High-level															
3, 2	IES1n1, IES1n0	<p>Specify the valid edge of the INTP1n1 pin.</p> <table border="1"> <thead> <tr> <th>IES1n1</th> <th>IES1n0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	IES1n1	IES1n0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
IES1n1	IES1n0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															
1, 0	IES0n1, IES0n0	<p>Specify the valid edge of the INTP1n0 pin.</p> <table border="1"> <thead> <tr> <th>IES0n1</th> <th>IES0n0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	IES0n1	IES0n0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
IES0n1	IES0n0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															

Remark n = 0, 1

7.4 Software Exception

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

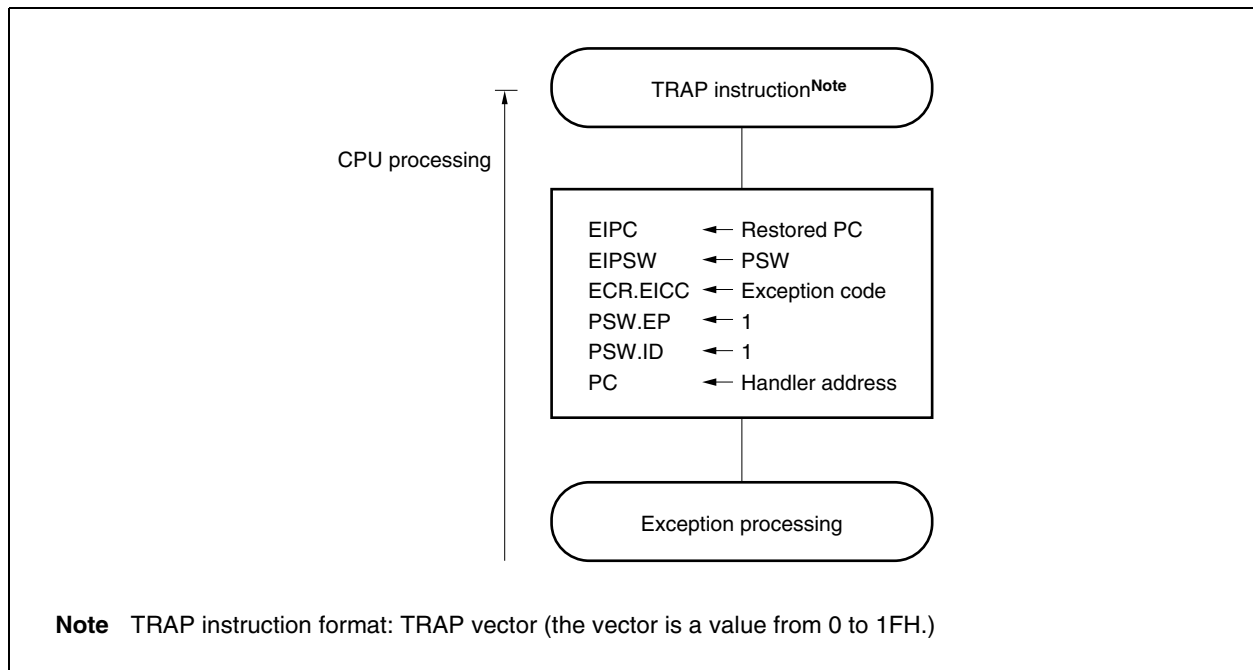
7.4.1 Operation

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

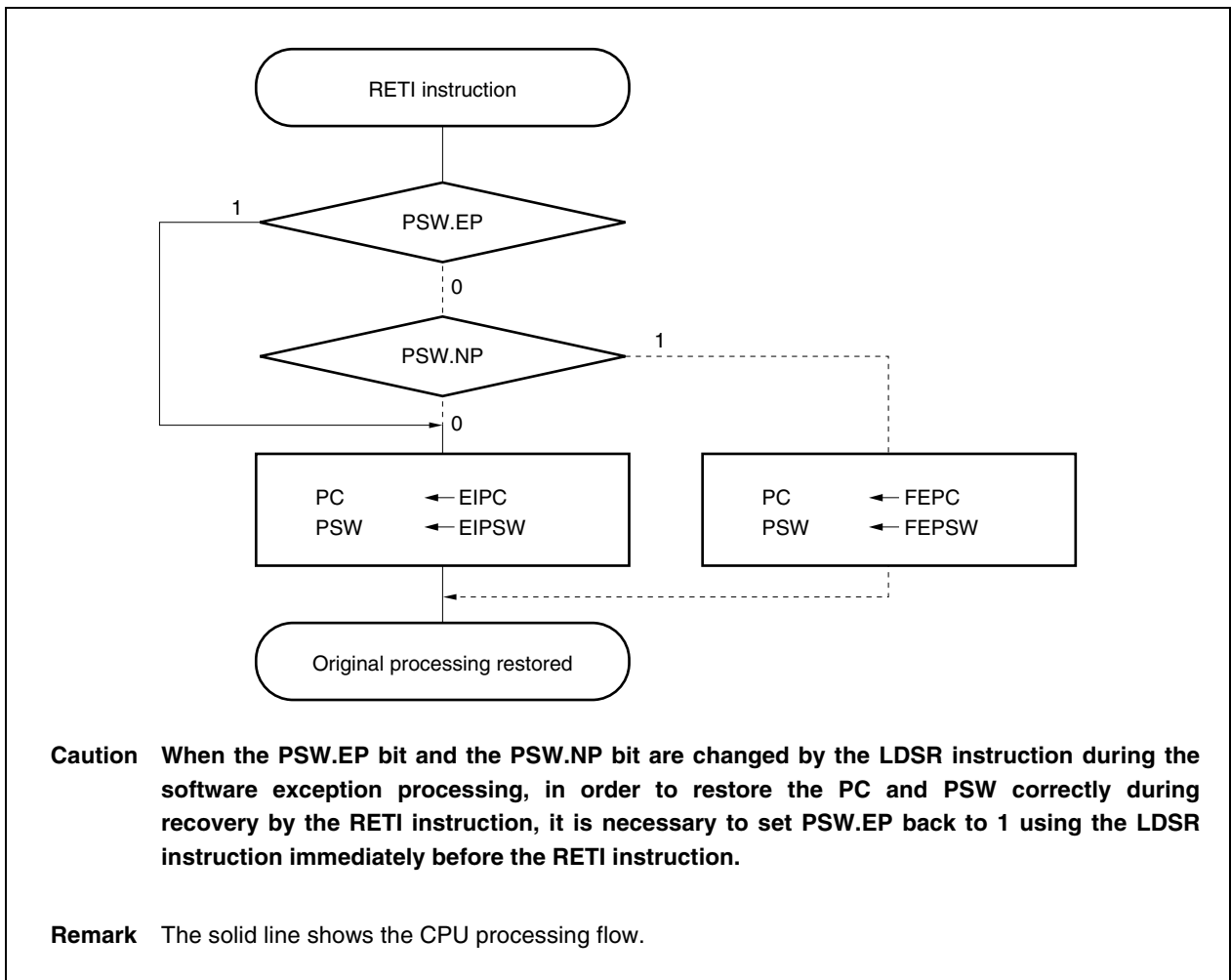
Recovery from software exception processing is carried out by the RETI instruction.

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.

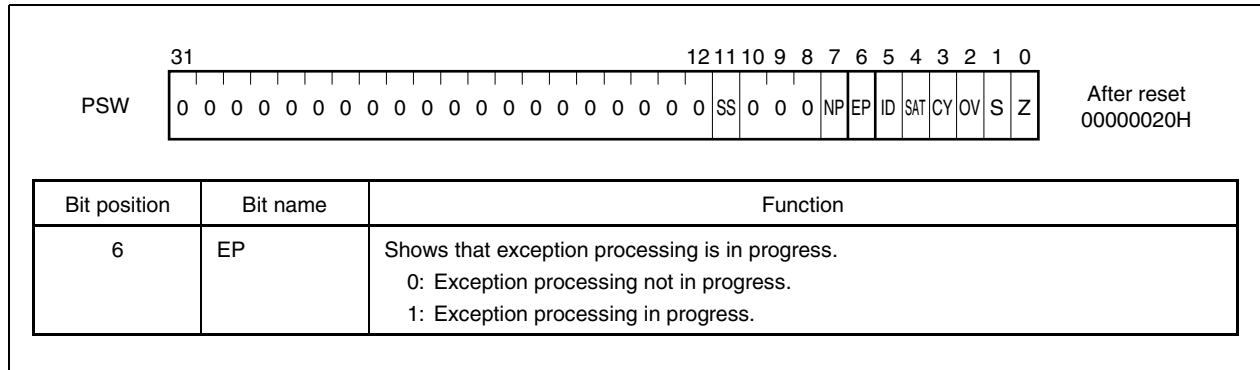
Figure 7-9 illustrates the processing of the RETI instruction.

Figure 7-9. RETI Instruction Processing



7.4.3 Exception status flag (EP)

The EP flag is bit 6 of the PSW, and is a status flag used to indicate that exception processing is in progress. It is set when an exception occurs.

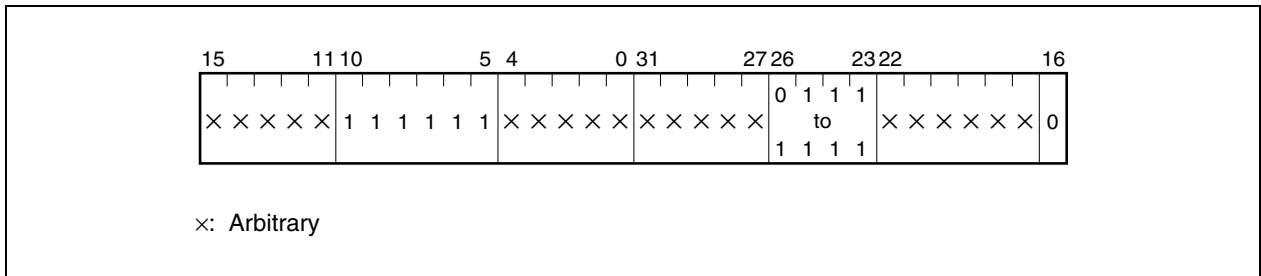


7.5 Exception Trap

An exception trap is an interrupt that is requested when the illegal execution of an instruction takes place. In the V850E/ME2, 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 11111B, 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.

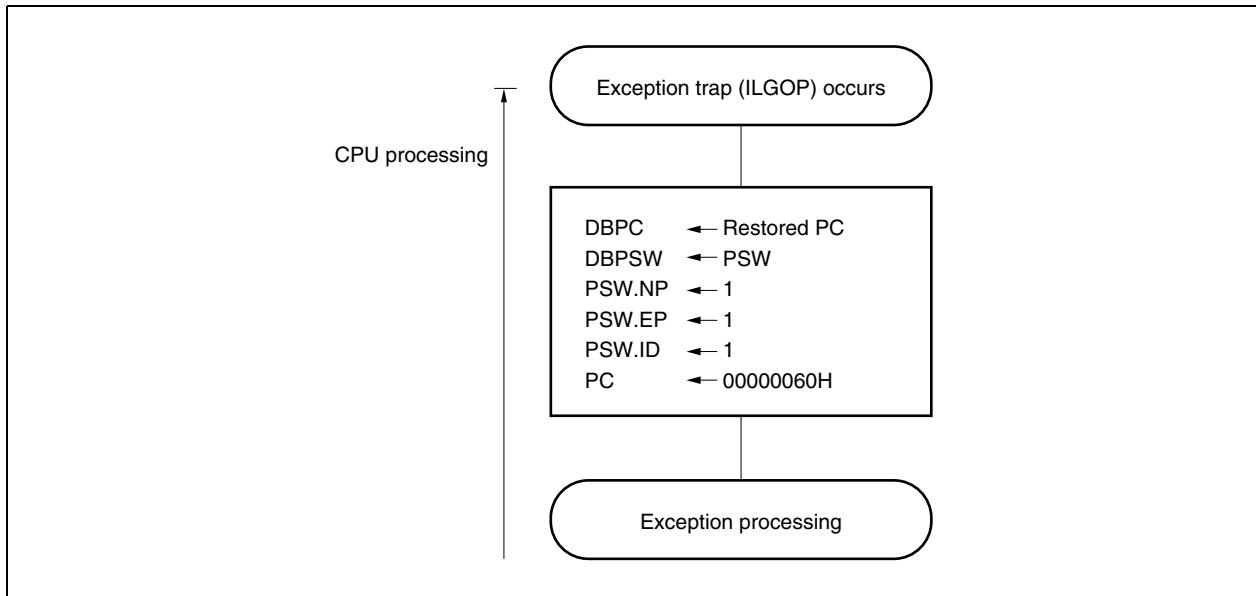
(1) Operation

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

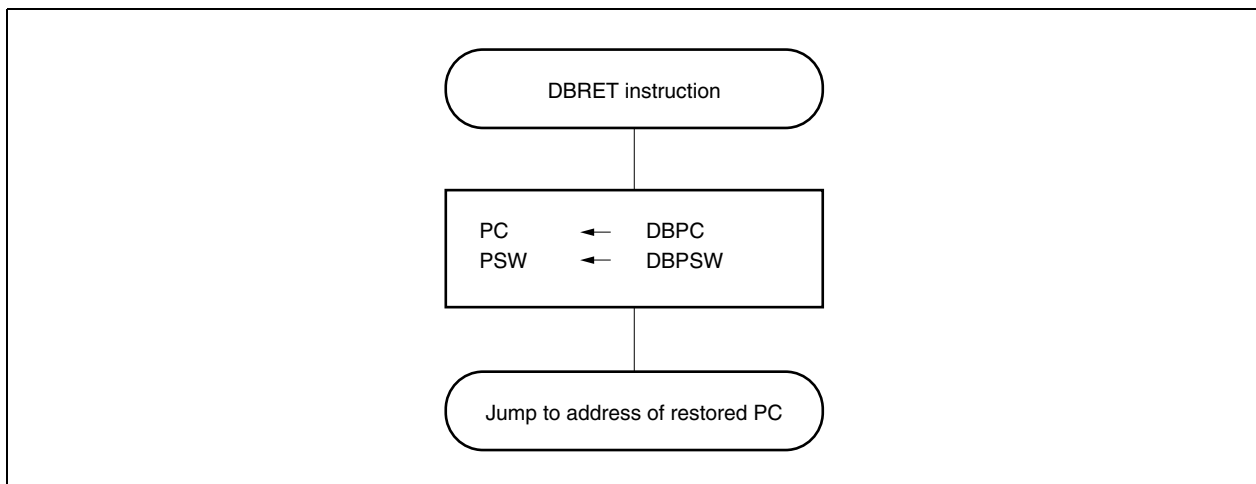
**(2) Restore**

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.

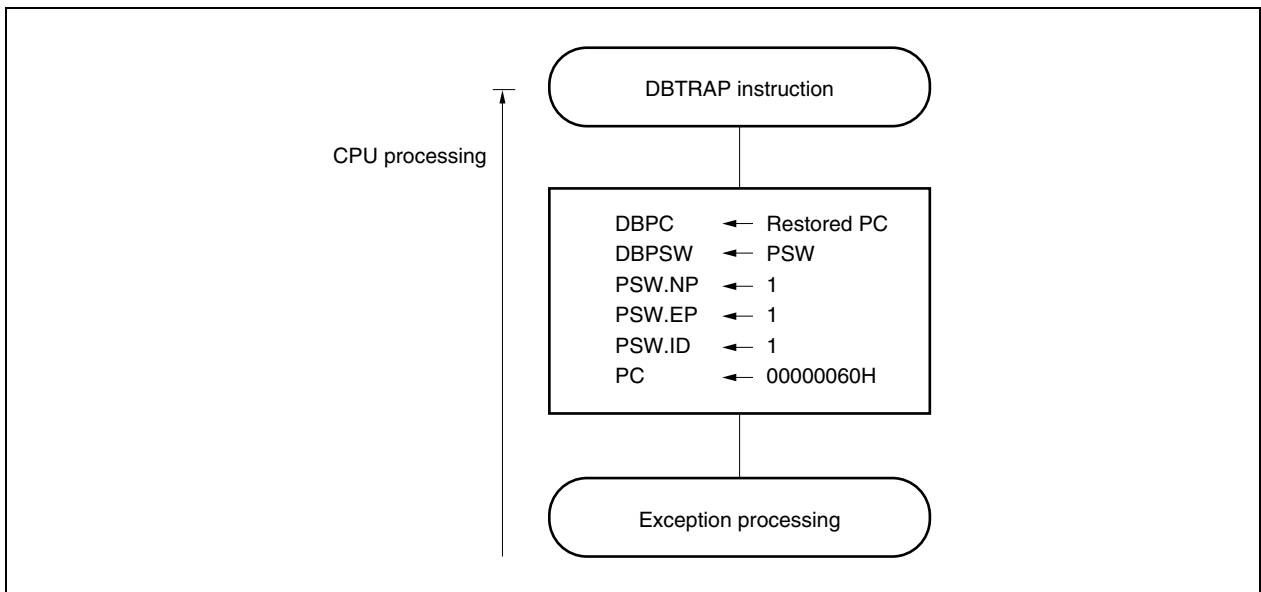
When the debug trap is generated, the CPU performs the following processing.

(1) Operation

- <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

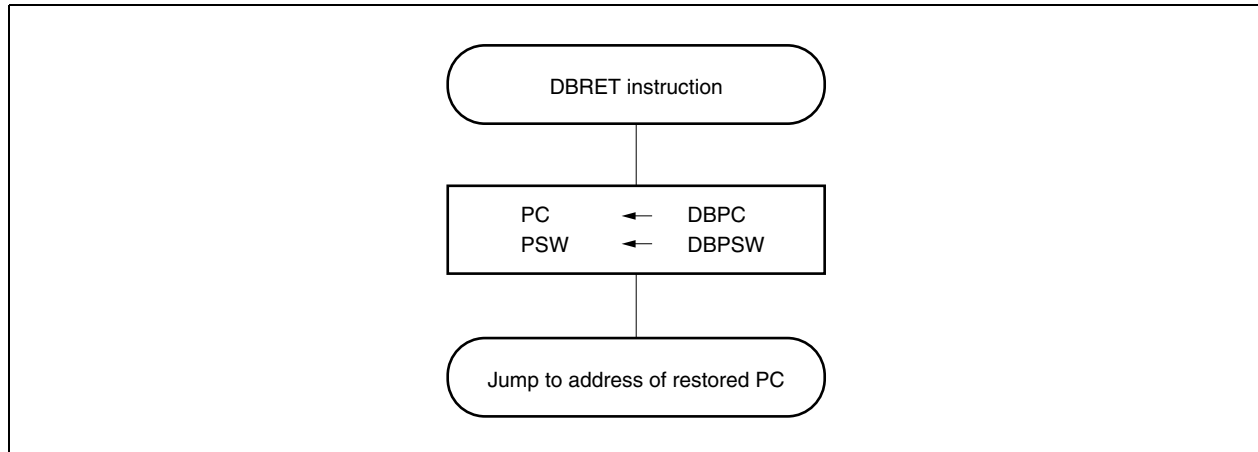
Recovery from a debug trap is carried out by the DBRET instruction. By executing the DBRET instruction, the CPU carries out the following processing and controls the address of the restored PC.

<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 serviced can be interrupted during servicing if there is an interrupt request with a higher priority level, and the higher priority interrupt request is acknowledged and serviced first.

If there is an interrupt request with a lower priority level than the interrupt request currently being serviced, that interrupt request is held pending.

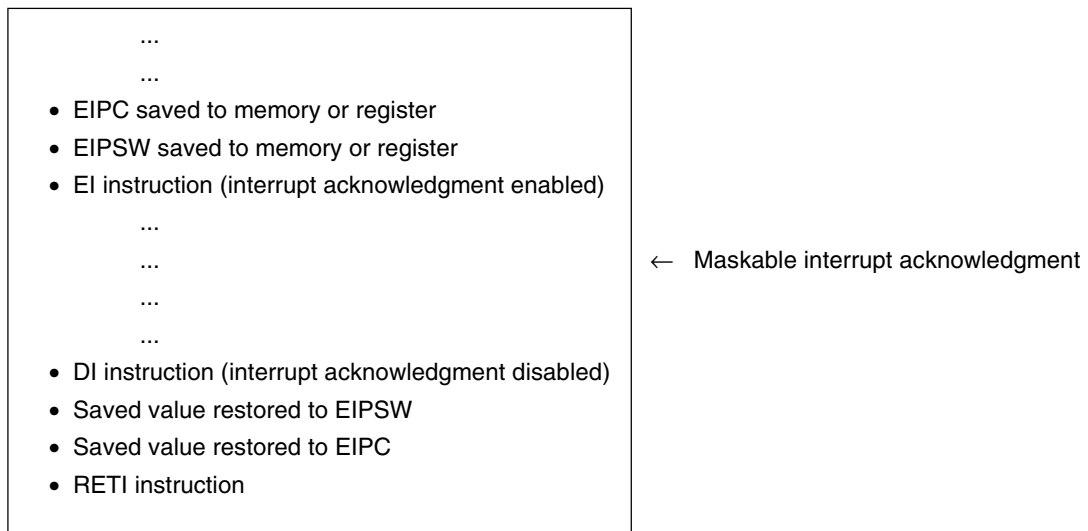
Multiple interrupt servicing control of maskable interrupts is executed when interrupts are enabled (ID = 0). Thus, to execute multiple interrupts, it is necessary to set the interrupt enabled state (ID = 0) even for an interrupt service routine.

If maskable interrupts are enabled or a software exception is generated in a maskable interrupt or software exception service program, it is necessary to save EIPC and EIPSW.

This is accomplished by the following procedure.

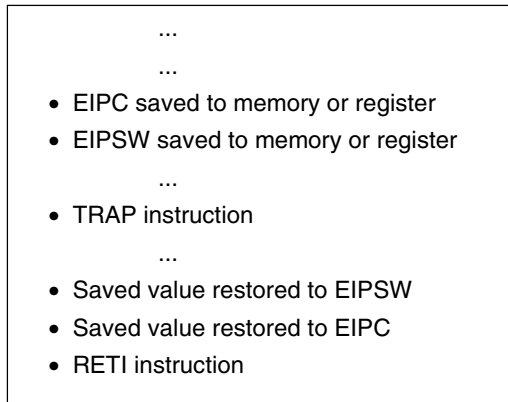
(1) Acknowledgment of maskable interrupts in service program

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), 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.

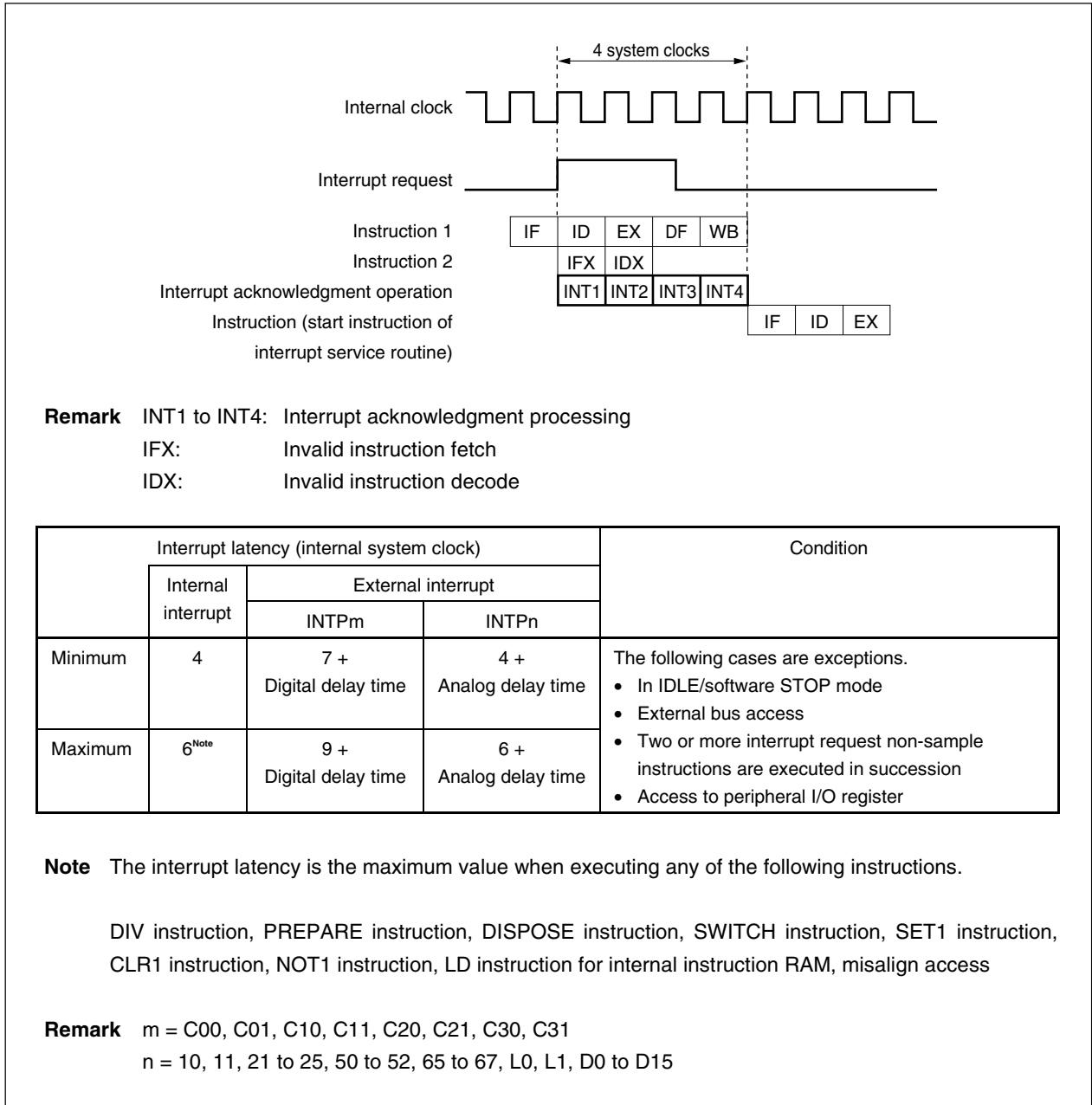
Caution In a non-maskable interrupt service routine (time until the RETI instruction is executed), maskable interrupts are suspended and not acknowledged.

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

7.7 Interrupt Latency Time

The V850E/ME2 interrupt latency time (from interrupt request generation to start of interrupt servicing) is described below.

Figure 7-14. Pipeline Operation at Interrupt Request Acknowledgment (Outline)



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 request non-sample instruction and the next instruction (interrupt is held pending).

The interrupt request non-sample instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- The store instruction for the command register (PRCMD)
- The load, store, or bit manipulation instructions for the following registers.
 - Interrupt-related registers:
Interrupt control register (xxICn), interrupt mask registers 0 to 5 (IMR0 to IMR5),
in-service priority register (ISPR), power-save control register (PSC)

CHAPTER 8 CLOCK GENERATION FUNCTION

The clock generator (CG) consists of a spread spectrum frequency synthesizer phase locked loop (SSCG) and a divider circuit, and generates and controls the clock supplied to the internal units as well as the CPU.

The SSCG is a spread spectrum clock generator used to suppress noise, and is effective in reducing the peak value of electromagnetic interference (EMI) noise.

8.1 Features

- SSCG output is used fixed to $\times 8$.
- Selection of SSCG output by PLLSEL pin
Set the PLLSEL pin as follows according to the value of f_x = Input frequency to X1 and X2 pins (F_x) $\times 8$

PLLSEL Pin	SSCG Output (f_x)
0	$96 \text{ MHz} \leq f_x \leq 150 \text{ MHz}$
1	$80 \text{ MHz} \leq f_x < 96 \text{ MHz}$

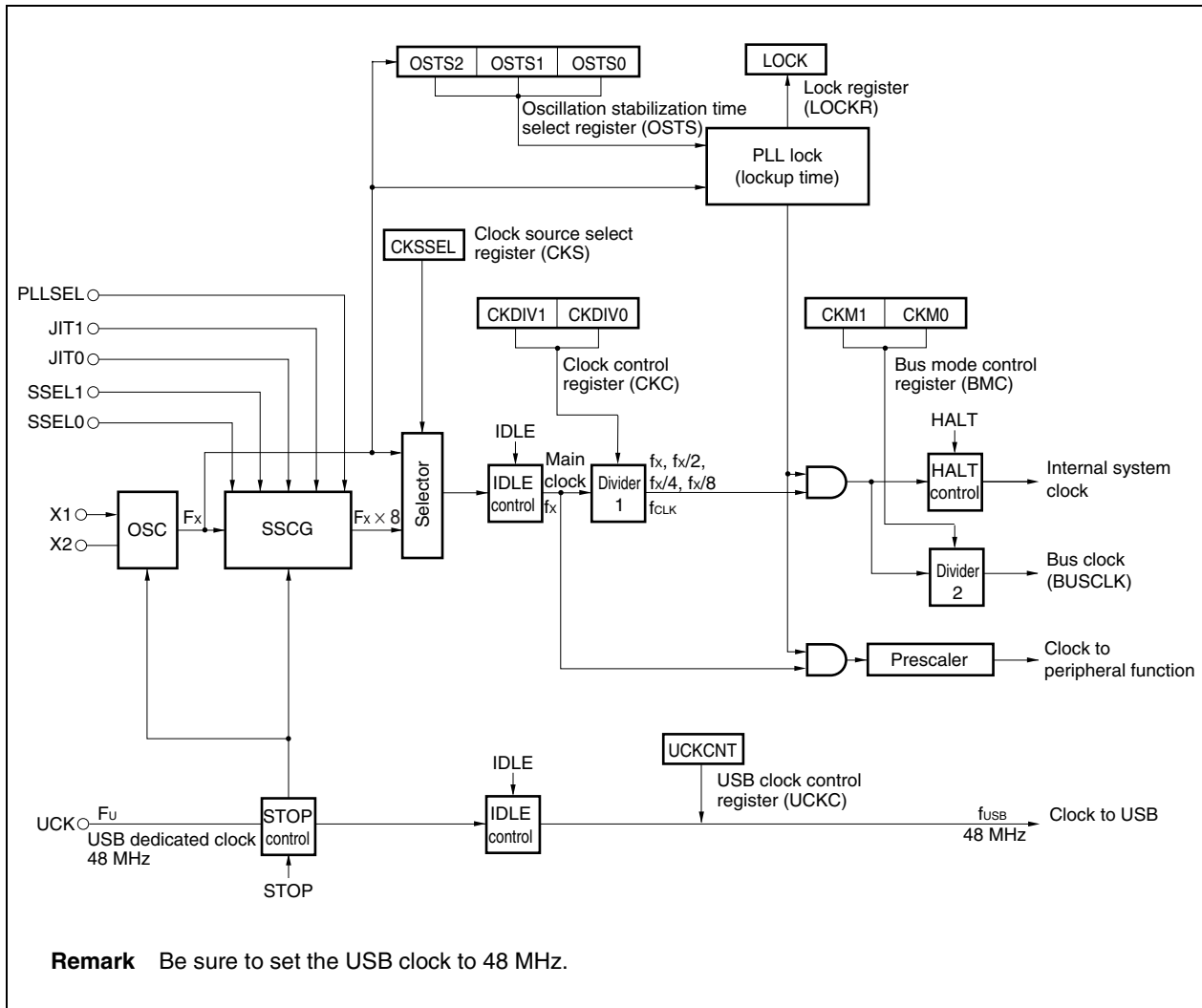
★

- Selection of frequency modulation rate (f_{DM}) of SSCG output
The following two setting methods are available.
 - Selection by J1T0 and J1T1 pins (without modulation (frequency fixed), -1% , -3% , -5%)
 - Selection by SSCGC register (without modulation (frequency fixed), -0.5% , -1% , -2% , -3% , -4% , -5%)

★

- MDL-Selector Table (modulation period)
Selection by SSCGC register (13 to 27 kHz, 23 to 37 kHz, 32 to 48 kHz)
- Division function by register setting (1/1, 1/2, 1/4, 1/8)
- Clock sources
Oscillation by connecting a resonator
- Operation mode selection by register setting
The clock to be supplied to each unit is selected from the following by register setting.
 - Clock output by SSCG
 - Clock from OSC without passing through SSCG
- USB-dedicated clock input
- Power-save control
 - HALT mode
 - IDLE mode
 - Software STOP mode

8.2 Configuration



8.3 Control Registers

8.3.1 Clock control register (CKC)

The clock control register is an 8-bit register that controls the internal system clock (f_{CLK}) in PLL mode. It can be written to only by a specific sequence so that it cannot easily be overwritten by mistake due to an inadvertent program loop.

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

Be sure to clear bits 7 to 2 to 0. If they are set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
CKC	0	0	0	0	0	0	CKDIV1	CKDIV0	FFFFFF822H	03H

Bit position	Bit name	Function															
1, 0	CKDIV1, CKDIV0	<p>Sets the internal system clock (f_{CLK}) when PLL mode is used.</p> <table border="1"> <thead> <tr> <th>CKDIV1</th> <th>CKDIV0</th> <th>Internal system clock (f_{CLK})</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>$f_x/8$</td> </tr> <tr> <td>0</td> <td>1</td> <td>$f_x/4$</td> </tr> <tr> <td>1</td> <td>0</td> <td>$f_x/2$</td> </tr> <tr> <td>1</td> <td>1</td> <td>f_x</td> </tr> </tbody> </table> <p>To change the internal system clock frequency in the middle of an operation, be sure to set it to f_x first, and then change the frequency as desired.</p>	CKDIV1	CKDIV0	Internal system clock (f_{CLK})	0	0	$f_x/8$	0	1	$f_x/4$	1	0	$f_x/2$	1	1	f_x
CKDIV1	CKDIV0	Internal system clock (f_{CLK})															
0	0	$f_x/8$															
0	1	$f_x/4$															
1	0	$f_x/2$															
1	1	f_x															

Remark f_x : Main clock

★

Cautions 1. Note that if the internal system clock (f_{CLK}) is changed, the frequency of the bus clock (BUSCLK) is also changed.

2. If it necessary to change the refresh interval of the SDRAM as a result of changing the internal system clock (f_{CLK}), follow this procedure.

<1> Mask all interrupts.

For how to disable maskable interrupts, refer to the description of interrupt mask registers 0 to 5 (IMR0 to IMR5) (7.3.5 Interrupt mask registers 0 to 5 (IMR0 to IMR5)).

For how to disable non-maskable interrupts, disable multiple interrupts by setting the NP bit of the PSW to 1 (refer to 3.2.2 (2) Program status word (PSW)).

<2> Clear the MEa bit of the BCTm register to 0 ($m = 0, 1, a = 0$ to 7).

<3> Clear the RENn bit to 0 ($n = 1, 3, 4, \text{ or } 6$).

<4> Set the MEa bit of the BCTm register to 1 ($m = 0, 1, a = 0$ to 7).

<5> Set a new value to the RCCn1, RCCn0, and RINn5 to RINn0 bits, and set the RENn bit to 1 ($n = 1, 3, 4, \text{ or } 6$).

<6> Write the same value currently set to the SCRn register to the SCRn register ($n = 1, 3, 4, \text{ or } 6$).

<7> Confirm that the WCFn bit of the SCRn register is set to 1, and access SDRAM ($n = 1, 3, 4, \text{ or } 6$).

To change the refresh interval, set a value at which refresh can be made in time even while the interval is changed. If the refresh interval is correctly secured, the processing in <1> above may be skipped. The RFSn and BCTm registers are prohibited from being rewritten, but they can be rewritten when the refreshing interval is re-set by changing the value of the CKC register.

A register write cycle of SDRAM is generated as a result of rewriting the SCRn register (processing in <6> above), but the value of SDRAM before the RFSn and SCRn registers are re-set is retained.

Set data in the clock control register (CKC) in 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 command register (PRCMD).
- <4> Set the clock control register (CKC) (with the following instruction).
 - Store instruction (ST/SST instruction)
- <5> Insert the NOP instructions (5 instructions (<5> to <9>)).
- <10> Release the interrupt disabled state (clear the NP bit of PSW to 0).

[Sample coding]

```

<1> LDSR   rX, 5
<2> MOV    0x02, r10
<3> ST.B   r10, PRCMD[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 issuance of data to the PRCMD (<3>) and writing to the specific register immediately after (<4>), the write operation to the specific register is not performed. Therefore, set the NP bit of the PSW to 1 (<1>) to disable interrupt acknowledgment. Also disable interrupt acknowledgment when selecting a bit manipulation instruction for the specific register setting.
 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. Be sure to terminate all DMA transfers prior to the execution of the above sequence.

8.3.2 Clock source select register (CKS)

This 8-bit register controls supply of the main clock (fx). It can be written to only by a specific sequence so that it cannot easily be overwritten by mistake due to an inadvertent program loop.

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

- Cautions**
1. With the V850E/ME2, it is not assumed that the CPU operates with the OSC output always supplied as the main clock (CKSSEL bit = 0). Therefore, be sure to confirm in the initialization sequence that the LOCK bit of the LOCKR register is 0, and then supply the main clock from the SSCG output (CKSSEL bit = 1). Otherwise, the operation will not be guaranteed.
 2. If the software STOP mode is released by a non-maskable interrupt request (NMI) or an unmasked maskable interrupt request, the system operates on the SSCG output clock after PLL frequency stabilization time (about 2 ms) after the count time (oscillation stabilization time set by the OSTS register) of the time base counter (TBC) has elapsed. Therefore, it is not necessary to re-set the CKS register. If the software STOP mode is released by $\overline{\text{RESET}}$ pin input, set the CKS register in accordance with the initialization sequence (see 3.4.10 Initialization sequence).

	7	6	5	4	3	2	1	<0>	Address	After reset
CKS	0	0	0	0	0	0	0	CKSSEL	FFFFF82CH	00H

Bit position	Bit name	Function
0	CKSSEL	Controls supply of the main clock (fx). 0: OSC output clock (Fx) 1: SSCG output clock (Fx × 8)

Set data in the clock source select register (CKS) in 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 command register (PRCMD).
- <4> Set the clock source select register (CKS) (with the following instruction).
 - Store instruction (ST/SST instruction)
- <5> Insert the NOP instructions (5 instructions (<5> to <9>)).
- <10> Release the interrupt disabled state (clear the NP bit of PSW to 0).

[Sample coding]

```

<1> LDSR    rX, 5
<2> MOV     0x01, r10
<3> ST.B    r10, PRCMD[r0]
<4> ST.B    r10, CKS[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 issuance of data to the PRCMD (<3>) and writing to the specific register immediately after (<4>), the write operation to the specific register is not performed. Therefore, set the NP bit of the PSW to 1 (<1>) to disable interrupt acknowledgment. Also disable interrupt acknowledgment when selecting a bit manipulation instruction for the specific register setting.
 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. Be sure to terminate all DMA transfers prior to the execution of the above sequence.

8.3.3 SSCG control register (SSCGC)

This is an 8-bit register that controls the frequency modulation rate and modulation period of SSCG output. It modulates the frequency by the frequency modulation rate (Down Spread) set by the ADJ2 to ADJ0 bits within the modulation period set by the SMDLn bit. This is effective in reducing the peak value of EMI noise. It can be written to only by a specific sequence so that it cannot easily be overwritten by mistake due to an inadvertent program loop.

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

Caution The SSCGC register can be set only when OSC output is supplied as the main clock (CKSSEL bit of CKS register = 0). If the setting of the SSCGC register is changed, the SSCG is unlocked (LOCK bit of LOCKR register = 1). Be sure to confirm that the LOCK bit = 0 before starting to supply SSCG output as the main clock (CKSSEL bit = 1). Otherwise, the operation will not be guaranteed.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
SSCGC	0	0	SMDL1	SMDL0	ADJON	ADJ2	ADJ1	ADJ0	FFFFF836H	Note

Note The default value is as follows.
 SMDL1 bit = 0, SMDL0 bit = 1
 The ADJON and ADJ2 to ADJ0 bits are set as follows by the JIT1 and JIT0 pins.

JIT1 pin	JIT0 pin	Reset value			
		ADJON bit	ADJ2 bit	ADJ1 bit	ADJ0 bit
0	0	0	0	0	0
0	1	1	0	0	1
1	0	1	0	1	1
1	1	1	1	0	1

Bit position	Bit name	Function															
5, 4	SMDL1, SMDL0	Set the modulation period of SSCG output. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th>SMDL1</th> <th>SMDL0</th> <th>Modulation period of SSCG output</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>13 to 27 kHz</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>23 to 37 kHz</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>32 to 48 kHz</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	SMDL1	SMDL0	Modulation period of SSCG output	0	0	13 to 27 kHz	0	1	23 to 37 kHz	1	0	32 to 48 kHz	1	1	Setting prohibited
SMDL1	SMDL0	Modulation period of SSCG output															
0	0	13 to 27 kHz															
0	1	23 to 37 kHz															
1	0	32 to 48 kHz															
1	1	Setting prohibited															

★

Bit position	Bit name	Function						
★ 3 to 0	ADJON, ADJ2 to ADJ0	Set the frequency modulation rate of SSCG output.						
		ADJON	ADJ2	ADJ1	ADJ0	Frequency modulation rate of SSCG output		
						Min.	Typ.	Max.
		0	×	×	×	No modulation (frequency fixed)		
		1	0	0	0	-0.31%	-0.50%	-1.28%
		1	0	0	1	-0.67%	-1.00%	-1.75%
		1	0	1	0	-1.23%	-2.00%	-3.10%
		1	0	1	1	-1.74%	-3.00%	-4.33%
		1	1	0	0	-2.48%	-4.00%	-5.83%
		1	1	0	1	-3.27%	-5.00%	-7.28%
Other than above				Setting prohibited				
<p>Cautions 1. If the frequency modulation rate of SSCG output is set to other than “No modulation (frequency fixed)” (ADJON bit = 0), the internal system clock, bus clock (BUSCLK), and clock supplied to the on-chip peripheral functions all operate at the frequency that accords with the frequency modulation rate set by the ADJ2 to ADJ0 bits (see 8.5 Operating Clock Provisions). Therefore, thoroughly evaluate and confirm the system.</p> <p>2. An overshoot/undershoot of the frequency modulation rate occurs.</p>								

Set data in the SSCG control register (SSCGC) in 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 command register (PRCMD).
- <4> Set the SSCG control register (SSCGC) (with the following instruction).
 - Store instruction (ST/SST instruction)
- <5> Insert the NOP instructions (5 instructions (<5> to <9>)).
- <10> Release the interrupt disabled state (clear the NP bit of PSW to 0).

★ [Sample coding]

```

<1> LDSR    rX, 5
<2> MOV     0x08, r10
<3> ST.B    r10, PRCMD[r0]
<4> ST.B    r10, SSCGC[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 issuance of data to the PRCMD (<3>) and writing to the specific register immediately after (<4>), the write operation to the specific register is not performed. Therefore, set the NP bit of the PSW to 1 (<1>) to disable interrupt acknowledgment. Also disable interrupt acknowledgment when selecting a bit manipulation instruction for the specific register setting.
 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. Be sure to terminate all DMA transfers prior to the execution of the above sequence.

8.3.4 USB clock control register (UCKC)

This register is used to select the clock source of the USB clock. Be sure to input f_{USB} at 48 MHz.

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

Be sure to clear the bits of this register other than bit 7 to 0. If they are set to 1, the operation is not guaranteed.

★

Caution When using the USB function, be sure to set (1) the UCKCNT bit.
 If the registers related to the USB function are read while the UCKCNT bit is 0, 0 is read.

	<7>	6	5	4	3	2	1	0	Address	After reset
UCKC	UCKCNT	0	0	0	0	0	0	0	FFFFF82EH	00H

Bit position	Bit name	Function
7	UCKCNT	Controls clock supply to USB. 0: Stops clock supply to USB. 1: Supplies clock to USB.

8.3.5 Lock register (LOCKR)

The lockup time (frequency stabilization time: approx. 2 ms) is the time from when the power is turned on or software STOP mode is released until the phase locks at the prescribed frequency. The state until this stabilization occurs is called the lockup state, and the stabilized state is called the locked state.

The lock register (LOCKR) has a LOCK flag that indicates that the PLL is in the lock wait state.

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, writing to SSCGC register, or $\overline{\text{RESET}}$ input, the LOCK flag is set to 1. If the phase is unlocked by a cause other than these, 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	FFFFFF824H	01H

Bit position	Bit name	Function
0	LOCK	This is a read-only flag that indicates the PLL lock wait state. This flag holds the value 0 as long as a lockup state is maintained. 0: Indicates that the PLL is locked. 1: Indicates that the PLL is waiting to be locked (unlock state).

If 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 by software immediately after operation begins and start processing after the PLL is locked.

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 to 0.

8.3.6 Oscillation stabilization time select register (OSTS)

This is an 8-bit register that specifies the oscillation stabilization time.

The OSTS register is used to make sure that the oscillation stabilization time of the oscillator elapses when the software STOP mode is released. After the software STOP mode is released, the oscillation stabilization time is counted by the time base counter (TBC), and program execution is started about 2 ms later.

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

	7	6	5	4	3	2	1	0	Address	After reset
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFFF6C0H	04H

Bit position	Bit name	Function																																				
2 to 0	OSTS2 to OSTS0	Specify the oscillation stabilization time.																																				
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">OSTS2</th> <th style="width: 10%;">OSTS1</th> <th style="width: 10%;">OSTS0</th> <th style="width: 70%;">Oscillation stabilization time</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;">$F \times 2^{13}$</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;">$F \times 2^{15}$</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;">$F \times 2^{16}$</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;">$F \times 2^{17}$</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;">$F \times 2^{18}$</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;">$F \times 2^{19}$</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;">$F \times 2^{20}$</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;">$F \times 2^{21}$</td> </tr> </tbody> </table>	OSTS2	OSTS1	OSTS0	Oscillation stabilization time	0	0	0	$F \times 2^{13}$	0	0	1	$F \times 2^{15}$	0	1	0	$F \times 2^{16}$	0	1	1	$F \times 2^{17}$	1	0	0	$F \times 2^{18}$	1	0	1	$F \times 2^{19}$	1	1	0	$F \times 2^{20}$	1	1	1	$F \times 2^{21}$
		OSTS2	OSTS1	OSTS0	Oscillation stabilization time																																	
		0	0	0	$F \times 2^{13}$																																	
		0	0	1	$F \times 2^{15}$																																	
		0	1	0	$F \times 2^{16}$																																	
		0	1	1	$F \times 2^{17}$																																	
		1	0	0	$F \times 2^{18}$																																	
		1	0	1	$F \times 2^{19}$																																	
1	1	0	$F \times 2^{20}$																																			
1	1	1	$F \times 2^{21}$																																			

Remark Fx: OSC output clock

★

8.4 Operation

8.4.1 Operation status of each clock

The following table shows the operation status of each clock.

Table 8-1. Operation Status of Each Clock

Clock Source		OSC	PLL	Main Clock (fx)		USB Clock (f _{USB})	BUSCLK
				Clock Supply to On-Chip Peripheral Function	Internal System Clock (f _{CLK})		
PLL mode	Normal operation	√	√	√	√	√	√
	HALT mode	√	√	√	×	√	√
	IDLE mode	√	√	×	×	×	×
	Software STOP mode	×	×	×	×	×	×
	Oscillation stabilization period	√	√	×	√ ^{Note 1}	×	√ ^{Note 1}
	Reset period	√	×	×	√ ^{Note 2}	×	√ ^{Note 2}

Remark √: Operates
 ×: Stops

Notes 1. The system operates on the OSC output clock at power application. It stops when the operation is restored from the software STOP mode by an interrupt.

★ **2.** Operates on the OSC output clock.

8.4.2 Setting of input clock (F_x)

This table lists the frequencies supplied to the V850E/ME2.

Table 8-2. Frequency List

Multiple	PLLSEL Pin	SSEL1 Pin	SSEL0 Pin	Input Frequency (MHz) (Target Values)	Main Clock (fx) Frequency (MHz)
8	1	0	1	Setting prohibited	Setting prohibited
		1	0	10.00 to 10.19	80.00 to 81.59
		1	1	10.20 to 11.99	81.60 to 95.99
	0	0	0	12.00 to 14.39	96.00 to 115.19
		0	1	14.40 to 17.39	115.20 to 139.19
		1	0	17.40 to 18.75	139.20 to 150.00
		1	1	Setting prohibited	Setting prohibited

★ **Caution** The MAX. value of f_{CLK} of a 100 MHz version is 100 MHz, that of a 133 MHz version is 133 MHz, and that of a 150 MHz version is 150 MHz.
 The operation is not guaranteed if f_{CLK} (MAX.) < f_x.
 Make sure that f_x does not exceed the guaranteed operating frequency of each product.

★ 8.5 Operating Clock Provisions

SSCG consists of a multiplier that multiplies the OSC clock by eight and a modulation circuit for frequency diffusion. The multiplier that multiplies the OSC clock generates an output error of up to ± 150 ps in the output period because of its circuit characteristics.

The modulation circuit adds the modulation error with respect to the set modulation rate, which varies depending on the modulation rate.

Therefore, the output frequency (f_{CLK}) of BUSCLK, which outputs a clock via SSCG, and the operating frequency of the on-chip peripheral I/O (main clock (f_x) of timers C, D, and ENC1, PWM unit, CSI3, UARTB, and A/D converter) are affected by this error. When calculating the frequency of each signal, therefore, be sure to take this error into consideration.

The methods of calculating the output period and output frequency at each modulation rate are shown below.

(1) Minimum value of operating frequency (f_{MIN})

Modulation Rate of SSCG Output (%)	Parameter	Output Period: T_{MAX} (ns)	Output Frequency (MHz)
No modulation	1.00000	$(1/f_x) \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$
-0.50	0.99890	$\{1/(0.99110 \times f_x)\} \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$
-1.00	0.99375	$\{1/(0.98625 \times f_x)\} \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$
-2.00	0.98550	$\{1/(0.97450 \times f_x)\} \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$
-3.00	0.97665	$\{1/(0.96335 \times f_x)\} \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$
-4.00	0.96915	$\{1/(0.95085 \times f_x)\} \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$
-5.00	0.96140	$\{1/(0.93860 \times f_x)\} \times 10^3 + 0.15$	$(1/T_{MAX}) \times 10^3$

(2) Maximum value of operating frequency (f_{MAX})

Modulation Rate of SSCG Output (%)	Parameter	Output Period: T_{MAX} (ns)	Output Frequency (MHz)
No modulation	1.00000	$(1/f_x) \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$
-0.50	1.00390	$\{1/(1.00390 \times f_x)\} \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$
-1.00	1.00375	$\{1/(1.00375 \times f_x)\} \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$
-2.00	1.00550	$\{1/(1.00550 \times f_x)\} \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$
-3.00	1.00665	$\{1/(1.00665 \times f_x)\} \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$
-4.00	1.00915	$\{1/(1.00915 \times f_x)\} \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$
-5.00	1.01140	$\{1/(1.01140 \times f_x)\} \times 10^3 - 0.15$	$(1/T_{MIN}) \times 10^3$

(3) Average operating frequency (f_{AVE})

The average operating frequency can be calculated by this expression.

$$\text{Average operating frequency } (f_{AVE}) = (f_{MIN} + f_{MAX})/2$$

The average operating frequency is in one period of the modulation period set by the SMDL1 and SMDL0 bits of the SSCGC register. The average operating frequency in a period shorter than the modulation period may have a relatively larger error than the result of the above expression.

8.5.1 Calculating BUSCLK frequency

If the modulation function of SSCG is used, the internal system clock frequency (f_{CLK}) from which BUSCLK is generated fluctuates. Exercise care in calculating the frequency of BUSCLK.

The following table shows the frequency fluctuation of BUSCLK.

Table 8-3. Frequency Fluctuation of BUSCLK

BMC Register		BUSCLK Frequency (MHz)		
CKM1	CKM1	Min.	Typ.	Max.
0	0	f_{MIN}	f_{AVE}	f_{MAX}
0	1	$f_{MIN}/2$	$f_{AVE}/2$	$f_{MAX}/2$
1	0	$f_{MIN}/3$	$f_{AVE}/3$	$f_{MAX}/3$
1	1	$f_{MIN}/4$	$f_{AVE}/4$	$f_{MAX}/4$

8.5.2 Calculating operating clock frequency of each on-chip peripheral function

When the modulation function of SSCG is used, the main clock (f_x) from which the operating clock of each on-chip peripheral function is generated fluctuates. Note (1) to (8) below.

(1) Count period of timer C:

Period of count clock frequency (set by the CSCn2 to CSCn0 bits of the TMCCn0 register) when input clock is specified (ETICn bit of the TMCCn1 register = 0) ($n = 0$ to 5)

The count period can be calculated from the average operating frequency.

However, the average operating frequency is in one period of the modulation period set by the SMDL1 and SMDL0 bits of the SSCGC register. Note that the average operating frequency may have a relatively larger error than the result of calculation during a count operation in a period shorter than the modulation period.

If it is necessary to guarantee an absolute value for the count operation, it is recommended to set appropriate counts during operation at f_{MIN} if the counting operation is performed within a specific time and during operation at f_{MAX} if the operation is performed for longer than a specific time.

Do not use the modulation function of SSCG if a high-accuracy count operation is necessary.

(2) Count period of timer D:

Period of count clock frequency (set by CSDn2 to CSDn0 bits of the TMCDn register) ($n = 0$ to 3)

The count period can be calculated from the average operating frequency.

However, the average operating frequency is in one period of the modulation period set by the SMDL1 and SMDL0 bits of the SSCGC register. Note that the average operating frequency may have a relatively larger error than the result of calculation during a count operation in a period shorter than the modulation period.

If it is necessary to guarantee an absolute value for the count operation, it is recommended to set appropriate counts during operation at f_{MIN} if the counting operation is performed within a specific time and during operation at f_{MAX} if the operation is performed for longer than a specific time.

Do not use the modulation function of SSCG if a high-accuracy count operation is necessary.

(3) Count period of timer ENC1:

Period of count clock frequency (set by PRM1n2 to PRM1n0 bits of the PRM1n register) when general-purpose timer mode is specified (T1CMDn bit of the TUM1n register = 0) (n = 0, 1)

The count period can be calculated from the average operating frequency.

However, the average operating frequency is in one period of the modulation period set by the SMDL1 and SMDL0 bits of the SSCGC register. Note that the average operating frequency may have a relatively larger error than the result of calculation during a count operation in a period shorter than the modulation period.

If it is necessary to guarantee an absolute value for the count operation, it is recommended to set appropriate counts during operation at f_{MIN} if the counting operation is performed within a specific time and during operation at f_{MAX} if the operation is performed for longer than a specific time. Duty setting is guaranteed during either high level output or low level output when PWM output is performed.

Do not use the modulation function of SSCG if a high-accuracy count operation is necessary.

(4) Pulse output period of PWM unit:

Period of operating clock frequency (set by the CKSPn1 and CKSPn0 bits of the PWMCn register) (n = 0, 1)

The pulse output period can be calculated from the average operating frequency.

However, the average operating frequency is in one period of the modulation period set by the SMDL1 and SMDL0 bits of the SSCGC register. Note that the average operating frequency may have a relatively larger error than the result of calculation when a pulse with a period shorter than the modulation period is output.

If it is necessary to guarantee an absolute value for the pulse output, it is recommended to set appropriate counts during operation at f_{MIN} if the counting operation is performed within a specific time and during operation at f_{MAX} if the operation is performed for longer than a specific time. Duty setting is guaranteed during either high level output or low level output when PWM output is performed.

Do not use the modulation function of SSCG if a high-accuracy pulse output is necessary.

(5) Serial communication (transmission, reception, or transmission/reception) transfer rate of CSI3:

Period of base clock frequency (CKS3n2 to CKS3n0 of the CSIC3n register = other than 111) (n = 0, 1) with master mode specified

The transfer rate can be calculated from the average operating frequency.

However, the average operating frequency is in one period of the modulation period set by the SMDL1 and SMDL0 bits of the SSCGC register. Note that the average operating frequency may have a relatively larger error than the result of calculation at a transfer rate shorter than the modulation period.

If it is necessary to guarantee an absolute value for the transfer rate, it is recommended to set an appropriate transfer rate during operation at f_{MIN} for the minimum transfer rate and during operation at f_{MAX} for the maximum transfer rate.

- (6) Serial communication (transmission or reception) transfer rate of UARTB:

Period of serial transfer speed (set by the UBnBRS15 to UBnBRS0 bits of the UBnCTL2 register) ($n = 0, 1$)

Set the value of the transfer rate (set value (k) of the UBnCTL2 register) so that the permissible maximum and minimum baud rate errors are satisfied at both f_{MIN} and f_{MAX} .

Here is a calculation example.

<1> Determine the set value (k) of the UBnCTL2 register from the calculation result of the expression "Target baud rate $\times 2/f_{AVE}$ (Hz)".

<2> Calculate the actual baud rate value at f_{MAX} and f_{MIN} , by using the set value (k) calculated in <1> above.

<3> Check that the result of calculation in <2> above satisfies the allowable maximum/minimum baud rate error.

<4> From <1> to <3>, calculate the set value of the transfer rate (set value (k) of the UBnCTL2 register).

Caution Change the modulation rate of SSCG if the set value (set value (k) of the UBnCTL2 register) of the transfer rate that satisfies the above calculation method cannot be obtained.

(Example)

$f_x = 150$ MHz: Setting of modulation rates of 4% and 5% is prohibited.

$f_x = 133$ MHz: Setting of modulation rates of 4% and 5% is prohibited.

$f_x = 100$ MHz: Setting of modulation rate of 5% is prohibited.

- (7) A/D converter:

Conversion time

Calculate the set value (2 to 10 μ s) of the A/D conversion time (set by the FR3 to FR0 bits of the ADM1 register) by f_{AVE} (calculation by f_{MIN} and f_{MAX} is not necessary).

- (8) Digital noise elimination time

Calculate the minimum noise elimination time set by the NCWCn and NCW1m registers at f_{MAX} and the maximum noise elimination time at f_{MIN} ($n = 0$ to 3, $m = 0, 1$).

8.6 Power-Save Control

8.6.1 Overview

The power-save function has the following three modes.

(1) HALT mode

In this mode, the clock generator (oscillator (OSC, SSCG) 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 using a combination of the HALT mode and the 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 (OSC, SSCG) 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 does not need to be secured.

The system is switched to IDLE mode by a 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

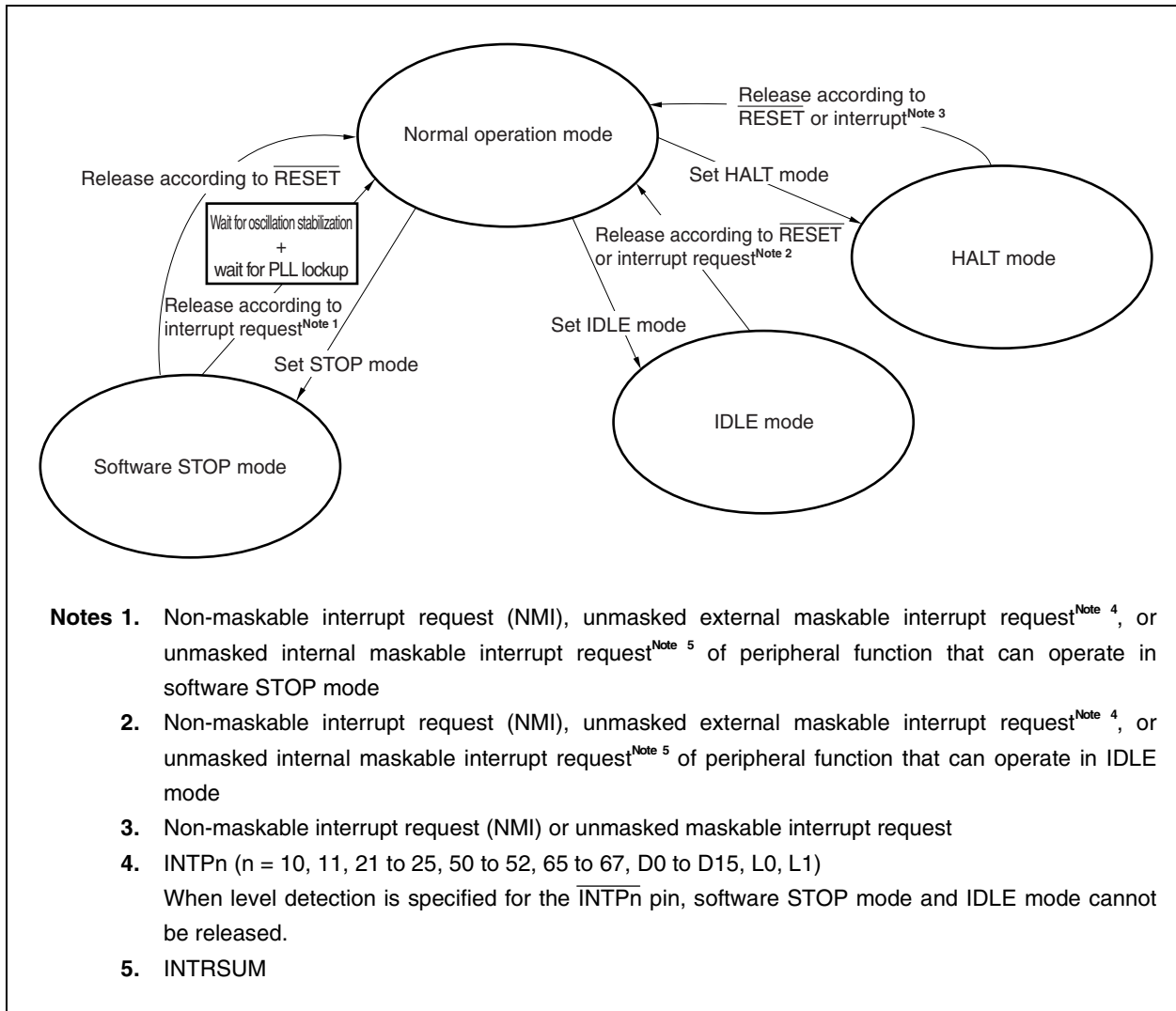
In this mode, the overall system is stopped by stopping the clock generator (oscillator (OSC, SSCG) and PLL synthesizer). The system enters an ultra-low-power-consumption state in which only leakage current is lost.

The system is switched to software STOP mode by a PSMR register setting. The PLL synthesizer's clock output is stopped at the same time the oscillator is stopped. After software STOP mode is released by $\overline{\text{RESET}}$ input, the oscillator's oscillation stabilization time must be secured until the system clock stabilizes. When the software STOP mode is released by the non-maskable interrupt request (NMI) or an unmasked maskable interrupt request, program execution is started about 2 ms after the count time of the time base counter (TBC) elapses.

Figure 8-1 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 8-1. Power-Save Mode State Transition Diagram



8.6.2 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 STP 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.

Be sure to clear bits 7, 5, and 4 to 0. If they are set to 1, the operation is not guaranteed.

7	6	5	4	3	2	1	<0>	Address	After reset
PSMR	0	0	0	0	0	0	PSM	FFFFF820H	00H

Bit position	Bit name	Function
0	PSM	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 an inadvertent program loop.

Writing to the first specific register (power-save control register (PSC), etc.) 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 is write-only in 8-bit units (when it is read, undefined data is read out).

7	6	5	4	3	2	1	0	Address	After reset	
PRCMD	RREG7	RREG6	RREG5	RREG4	RREG3	RREG2	RREG1	RREG0	FFFFF1FCH	Undefined

Bit position	Bit name	Function
7 to 0	RREG7 to RREG0	Registration code (arbitrary 8-bit data) The specific register targeted is as follows. <ul style="list-style-type: none"> • Power-save control register (PSC) • Clock control register (CKC) • Clock source select register (CKS) • SSCG control register (SSCGC)

(3) Power-save control register (PSC)

This is an 8-bit register that controls the power-save function.

If interrupts are enabled by the setting of the NMIM and INTM bits, the software STOP mode can be released by an interrupt request (except when interrupt servicing is disabled by the interrupt mask registers (IMR0 to IMR5)).

The software STOP mode is specified by the setting of the STP bit.

This register, which is one of the specific registers, is valid only when accessed in a specific sequence during a write operation.

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

Be sure to clear bits 7 and 6 to 0. If they are set to 1, the operation is not guaranteed.

- Cautions**
1. It is impossible to set the STP bit and the NMIM or INTM bit at the same time. Be sure to set the STP bit after setting the NMIM or INTM bit.
 2. The software STOP mode is not released by an interrupt request for which the NMIM and INTM bits are set to 1 because this interrupt request is invalid (it is not held pending).

	7	6	<5>	<4>	3	2	<1>	0	Address	After reset
PSC	0	0	NMIM	INTM	0	0	STP	0	FFFFFF1FEH	00H

Bit position	Bit name	Function
5	NMIM	This is the enable/disable setting bit for standby mode release using the valid edge input of NMI. 0: Release by NMI enabled 1: Release by NMI disabled
4	INTM	This is the enable/disable setting for standby mode release using an unmasked maskable interrupt (INTPn) (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, L1). 0: Release by maskable interrupt enabled 1: Release by maskable interrupt disabled
1	STP	Indicates the standby mode status. If 1 is written to this bit, the system enters IDLE or software STOP mode (set by the PSM bit of the PSMR register). When standby mode is released, this bit is automatically reset to 0. 0: Standby mode is released 1: Standby mode is in effect

Set data in the power-save control register (PSC) in 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> Insert 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 ineffective 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. **Be sure to terminate all DMA transfers prior to the execution of the above sequence.**

8.6.3 HALT mode

(1) Setting and operation status

In HALT mode, the clock generator (oscillator (OSC, SSCG) and PLL synthesizer) continues to operate, but the operation clock of the CPU is stopped. 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 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 data RAM, internal instruction RAM, and ports are maintained in the state they were in immediately before HALT mode began. Also, operation continues for all on-chip peripheral functions (other than ports) that do not depend on CPU instruction processing. The following shows the status of each hardware unit in HALT mode.

- Cautions**
- 1. 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.**
 - 2. At least 5 NOP instructions must be inserted after executing a HALT instruction.**

Table 8-4. Operation Status in HALT Mode

Function	Operation Status	
Clock generator	Operating	
Main clock (fx)	Operating	
CPU	Stopped	
Internal data	All internal data such as CPU registers, statuses, data, and the contents of internal data RAM and internal instruction RAM are maintained in the state they were in immediately before HALT mode began.	
DMAC	Operating	
INTC	Operating	
TMC0 to TMC5	Operating	
TMD0 to TMD3	Operating	
TMENC10, TMENC11	Operating	
UARTB0, UARTB1	Operating	
CSI30, CSI31	Operating	
A/D converter	Operating	
Ports	Maintained the state before HALT mode set	
USB function controller	Operating	
D0 to D31	Operating	
A0 to A25		
\overline{RD} , $\overline{WE/WR}$, \overline{BCYST}		
\overline{ULWR} , \overline{UUWR} , \overline{LLWR} , \overline{LUWR} , \overline{IORD} , \overline{IOWR}		
\overline{LLDQM} , \overline{LUDQM} , \overline{ULDQM} , \overline{UUDQM}		
\overline{LLBE} , \overline{LUBE} , \overline{ULBE} , \overline{UUBE}		
$\overline{CS0}$ to $\overline{CS7}$		
\overline{SDRAS}		
\overline{SDCAS}		
\overline{REFRQ}		
\overline{HLDK}		
\overline{HLDRQ}		
\overline{WAIT}		
$\overline{SELFREF}$		
\overline{SDCKE}		
BUSCLK		Clock output

(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 according to a non-maskable interrupt request or an 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. The operation after release is as follows.

Table 8-5. 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

However, if the system is set to HALT mode during an interrupt servicing routine, operation will differ as follows.

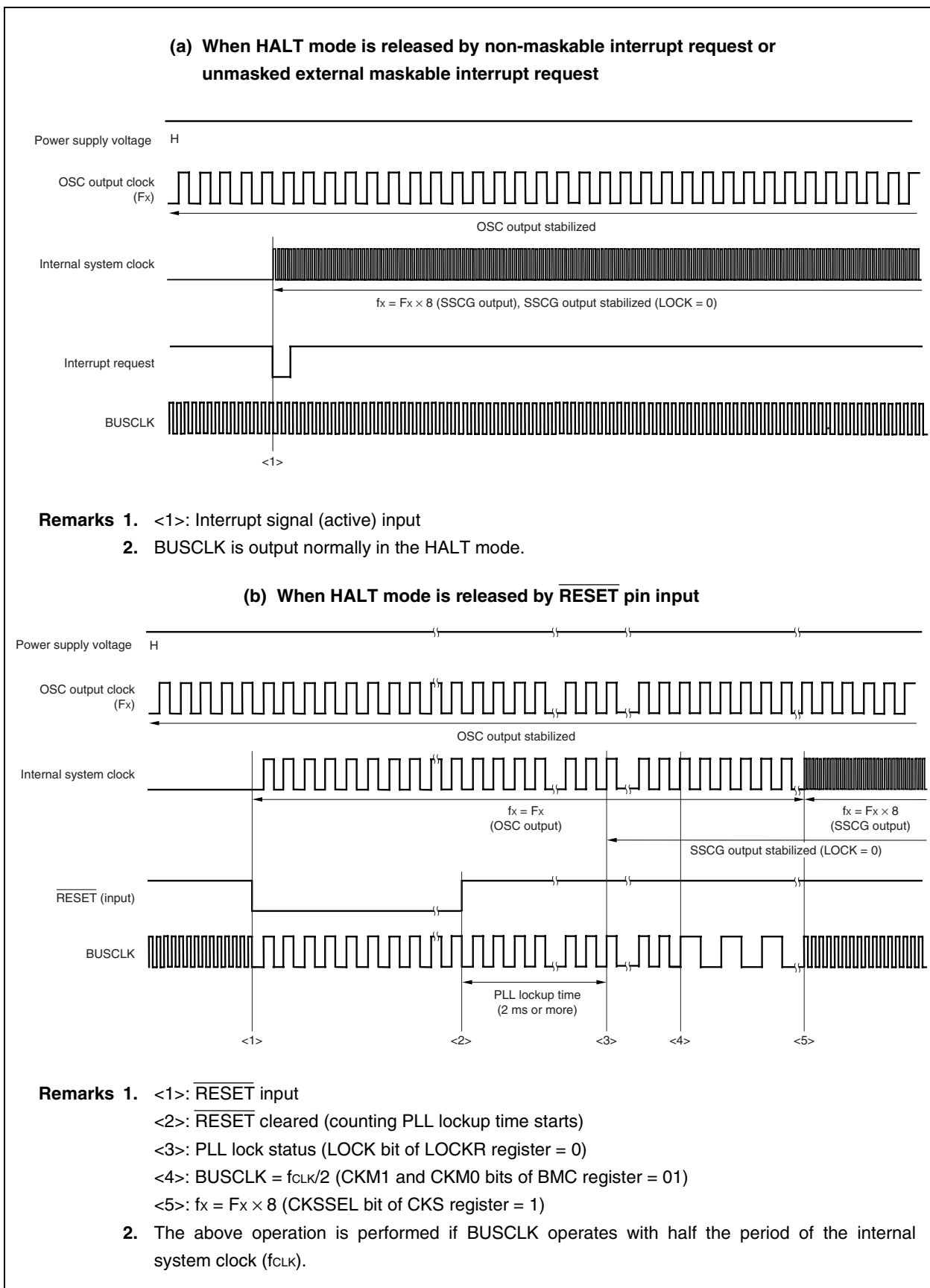
- (i) If an interrupt request is generated with a lower priority than that of the maskable interrupt request that is currently being serviced, HALT mode is released, but the newly generated interrupt request is not acknowledged. The new interrupt request is held pending.
- (ii) If an interrupt request (including non-maskable interrupt requests) is generated with a higher priority than that of the maskable interrupt request that is currently being serviced, HALT mode is released and the newly generated interrupt request is acknowledged.

(b) Release according to $\overline{\text{RESET}}$ pin input

This is the same as a normal reset operation.

The BUSCLK operation when the HALT mode is released is illustrated below.

Figure 8-2. BUSCLK Operation When HALT Mode Is Released



8.6.4 IDLE mode

(1) Setting and operation status

In this mode, the entire system is stopped with the clock generator (oscillator (OSC, SSCG) and PLL synthesizer) continuing to operate and clock supply to the CPU and other on-chip peripheral functions stopped.

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 does not need to be secured.

The system is switched to IDLE mode by setting the PSC or PSMR register using a store instruction (ST or SST instruction) or a bit manipulation instruction (SET1, CLR1, or NOT1 instruction) (see **8.6.2 Control registers**).

In IDLE mode, program execution is stopped, and the contents of all registers, internal data RAM, internal instruction RAM, and ports are maintained in the state they were in immediately before execution stopped. The operation of CPU and other on-chip peripheral functions is stopped. However, the on-chip peripheral functions that can operate on the external clock continue operating.

The following shows the status of each hardware unit in IDLE mode.

Caution Insert at least 5 NOP instructions after the instruction that stores data in the PSC register to set the IDLE mode.

Table 8-6. Operation Status in IDLE Mode

Function	Operation Status
Clock generator	Operating
Main clock (fx)	Stopped
CPU	Stopped
Internal data	All internal data such as CPU registers, statuses, data, and the contents of internal data RAM and internal instruction RAM are maintained in the state they were in immediately before IDLE mode began.
DMAC	Stopped
INTC	Stopped
TMC0 to TMC5	Stopped
TMD0 to TMD3	Stopped
TMENC10, TMENC11	Stopped
UARTB0, UARTB1	Stopped
CSI30, CSI31	Stopped
A/D converter	Stopped
Ports	Maintained the state before IDLE mode set
USB function controller	Stopped (Resume signal can only be received)
D0 to D31	High impedance
A0 to A25	
\overline{RD} , $\overline{WE}/\overline{WR}$, \overline{BCYST}	High-level output
\overline{ULWR} , \overline{UUWR} , \overline{LLWR} , \overline{LUWR} , \overline{IORD} , \overline{IOWR}	
\overline{LLDQM} , \overline{LUDQM} , \overline{ULDQM} , \overline{UUDQM}	
\overline{LLBE} , \overline{LUBE} , \overline{ULBE} , \overline{UUBE}	
$\overline{CS0}$ to $\overline{CS7}$	
\overline{SDRAS}	Self-refresh status when connected to SDRAM
\overline{SDCAS}	
\overline{REFRQ}	Operating (outputs high level when SDRAM controller is not used)
\overline{HLDK}	High-level output
\overline{HLDRQ}	Input (no sampling)
\overline{WAIT}	
$\overline{SELFREF}$	
\overline{SDCKE}	Low-level output (outputs high level when SDRAM controller is not used)
\overline{BUSCLK}	Low-level output

(2) Release of IDLE mode

The IDLE mode is released by a non-maskable interrupt request, an unmasked external maskable interrupt request (INTP_n)^{Note}, an unmasked internal maskable interrupt request of a peripheral function that can operate in the IDLE mode (INTRSUM), and $\overline{\text{RESET}}$ pin input (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, or L1).

Note When level detection is set, the IDLE mode cannot be released.

(a) Release according to a non-maskable interrupt request or an 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, an unmasked external maskable interrupt request (INTP_n), or an unmasked internal maskable interrupt request of a peripheral function that can operate in the IDLE mode (INTRSUM) regardless of the priority (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, or L1). The operation after release is as follows.

Table 8-7. 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

However, if the system is set to IDLE 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 maskable 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 maskable interrupt request that is currently being serviced, IDLE mode is released and the newly generated interrupt request is acknowledged.

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).

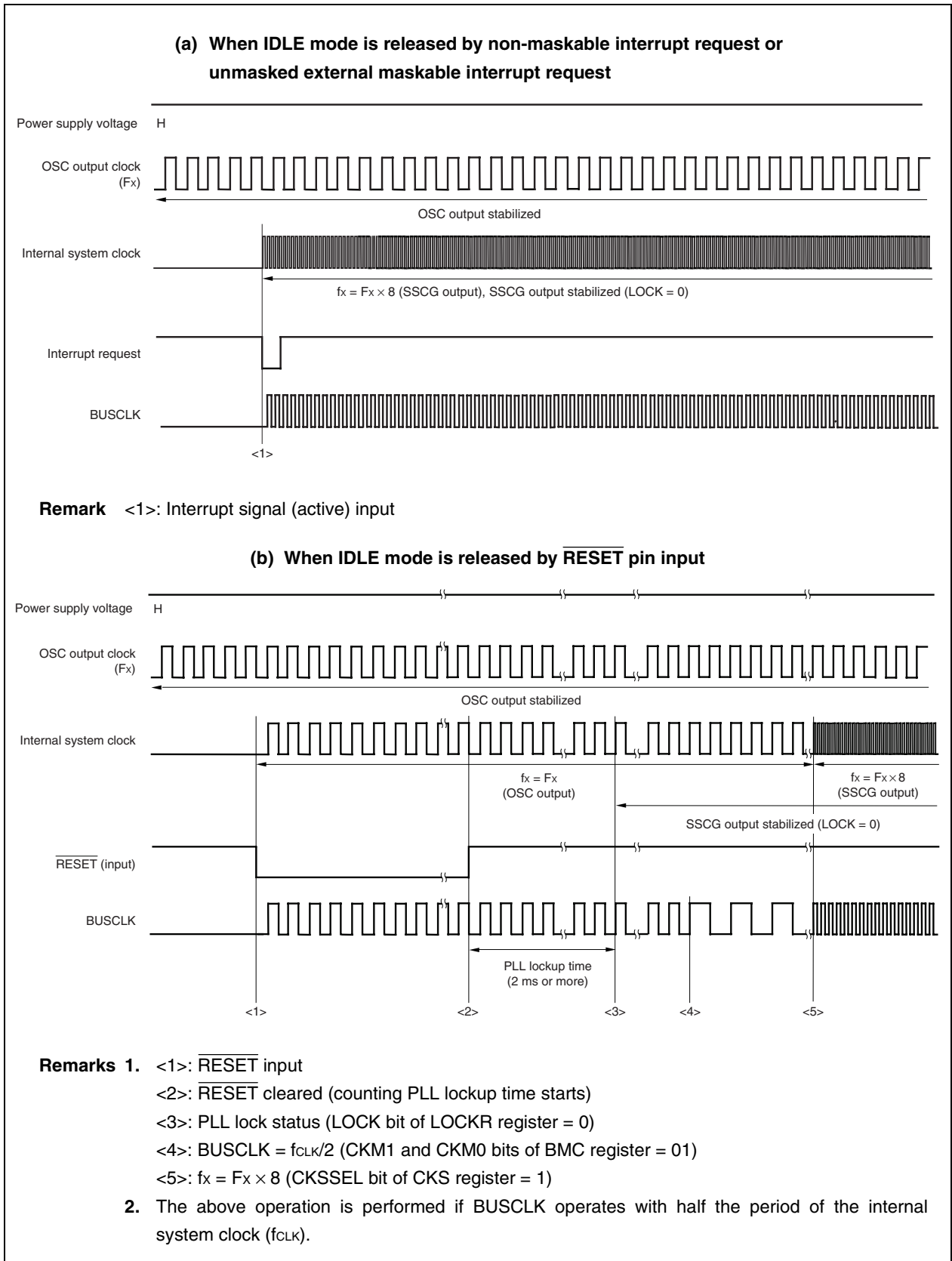
Interrupt servicing that is started when IDLE mode is released by NMI pin input is handled in the same way as normal NMI interrupt servicing that occurs during an emergency (because the NMI interrupt handler address is unique). Therefore, when a program must be able to distinguish between these two situations, 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 according to $\overline{\text{RESET}}$ pin input

This is the same as a normal reset operation.

The BUSCLK operation when the IDLE mode is released is illustrated below.

Figure 8-3. BUSCLK Operation When IDLE Mode Is Released



8.6.5 Software STOP mode

(1) Setting and operation status

In software STOP mode, the clock generator (oscillator (OSC, SSCG) and PLL synthesizer) is stopped. The overall system is stopped, and ultra-low power consumption is achieved in which only leakage 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 (see **8.6.2 Control registers**).

The oscillator's oscillation stabilization time and PLL frequency stabilization time must be secured after software STOP mode is released.

If the software STOP mode is released by an unmasked maskable interrupt request^{Note}, program execution is started after the PLL lockup time (about 2 ms) has elapsed after the count time of the time base counter (TBC) (time set by the OSTS register) elapses.

Although program execution stops in software STOP mode, the contents of all registers, internal data RAM, internal instruction RAM, and ports are maintained in the state they were in immediately before software STOP mode began. The operation of CPU and other on-chip peripheral functions is stopped.

The following shows the status of each hardware unit in software STOP mode.

Note INTP10, INTP11, INTP21 to INTP25, INTP50 to INTP52, INTP65 to INTP67, INTPD0 to INTPD15, INTPL0, INTPL1, INTRSUM

Cautions 1. Insert at least 5 NOP instructions after the instruction that stores data in the PSC register to set the software STOP mode.

- 2. If the software STOP mode is released by an unmasked maskable interrupt request, the external bus continues to operate in the same status as in the software STOP mode during the count time (time set by the OSTS register) of the time base counter (TBC) and the PLL lockup time.**

Table 8-8. Operation Status in Software STOP Mode

Function	Operation Status
Clock generator	Stopped
Main clock (fx)	Stopped
CPU	Stopped
Internal data	All internal data such as CPU registers, statuses, data, and the contents of internal data RAM and internal instruction RAM are maintained in the state they were in immediately before software STOP mode began.
DMAC	Stopped
INTC	Stopped
TMC0 to TMC5	Stopped
TMD0 to TMD3	Stopped
TMENC10, TMENC11	Stopped
UARTB0, UARTB1	Stopped
CSI30, CSI31	Stopped
A/D converter	Stopped
Ports	Maintained the state before software STOP mode set
USB function controller	Stopped (Resume signal can only be received)
D0 to D31	High impedance
A0 to A25	
\overline{RD} , $\overline{WE}/\overline{WR}$, \overline{BCYST}	High-level output
\overline{ULWR} , \overline{UUWR} , \overline{LLWR} , \overline{LUWR} , \overline{IORD} , \overline{IOWR}	
\overline{LLDQM} , \overline{LUDQM} , \overline{ULDQM} , \overline{UUDQM}	
\overline{LLBE} , \overline{LUBE} , \overline{ULBE} , \overline{UUBE}	
$\overline{CS0}$ to $\overline{CS7}$	
\overline{SDRAS}	
\overline{SDCAS}	Self-refresh status when connected to SDRAM
\overline{REFRQ}	Operating (outputs high level when SDRAM controller is not used)
\overline{HLDK}	High-level output
\overline{HLDRQ}	Input (no sampling)
\overline{WAIT}	
$\overline{SELFREF}$	
\overline{SDCKE}	
\overline{SDCKE}	Low-level output (outputs high level when SDRAM controller is not used)
\overline{BUSCLK}	Low-level output

(2) Release of software STOP mode

The software STOP mode is released by a non-maskable interrupt input, an unmasked external maskable interrupt request (INTPn)^{Note}, an unmasked internal maskable interrupt request of a peripheral function that can operate in the software STOP mode (INTRSUM), or $\overline{\text{RESET}}$ pin input (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, or L1). When the software STOP mode is released by $\overline{\text{RESET}}$ pin input, it is necessary to make sure that the oscillation stabilization time of the oscillator elapses. When the software STOP mode is released by the non-maskable interrupt or an unmasked maskable interrupt request, program execution starts after about 2 ms after the count time of the time base counter (TBC) elapses.

Note When level detection is set, the software STOP mode cannot be released.

(a) Release according to a non-maskable interrupt request or an unmasked maskable interrupt request

The software STOP mode can be released by an interrupt request only when it has been set with the INTM and NMIM bits of the PCS register cleared to 0.

Software STOP mode is released by a non-maskable interrupt request, an unmasked external maskable interrupt request (INTPn), or an unmasked internal maskable interrupt request of a peripheral function that can operate in the software STOP mode (INTRSUM) regardless of the priority (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, or L1). The operation after release is as follows.

Table 8-9. 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

The operation is performed as shown below if the software STOP mode is set in an interrupt servicing routine.

Table 8-10. Operation If Software STOP Mode Is Set in Interrupt Servicing Routine

Type of Interrupt Servicing Routine When Software STOP Mode Is Set	Releasing Cause		Operation
	Releasing Source	Priority ^{Note 1}	
Maskable interrupt	Maskable interrupt request	Low	Only software STOP mode is released and interrupt request is not acknowledged (is held pending).
		Same	
		High (ID = 1) ^{Note 2}	
	Non-maskable interrupt request	High (ID = 0) ^{Note 3}	Interrupt request is acknowledged when software STOP mode is released.
Non-maskable interrupt	Maskable interrupt request	–	Only software STOP mode is released and interrupt request is not acknowledged (is held pending).
	Non-maskable interrupt request	Low	
		Same	Interrupt request is acknowledged when software STOP mode is released.
	High		

- Notes**
1. Priority of interrupt (being serviced) when software STOP mode is set
 2. When the ID bit of the PSW is 1 (disabling acknowledging interrupt)
 3. When the ID bit of the PSW is 0 (enabling acknowledging interrupt)

Remark The software STOP mode is released by NMI regardless of the value of the NP bit of the PSW.

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 according to $\overline{\text{RESET}}$ pin input

This is the same as a normal reset operation.

The BUSCLK operation when the software STOP mode is released is illustrated below.

Figure 8-4. BUSCLK Operation When Software STOP Mode Is Released (1/2)

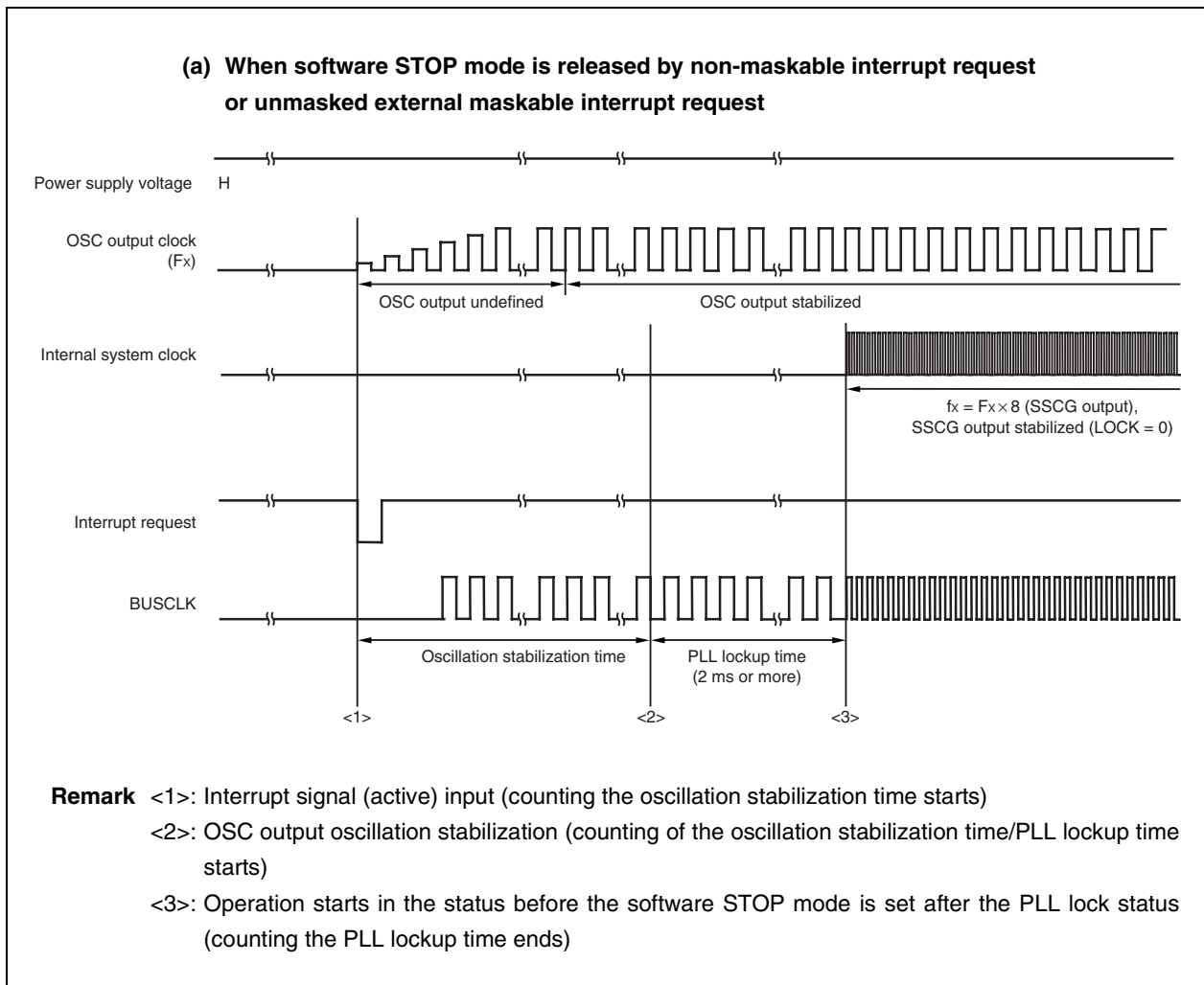
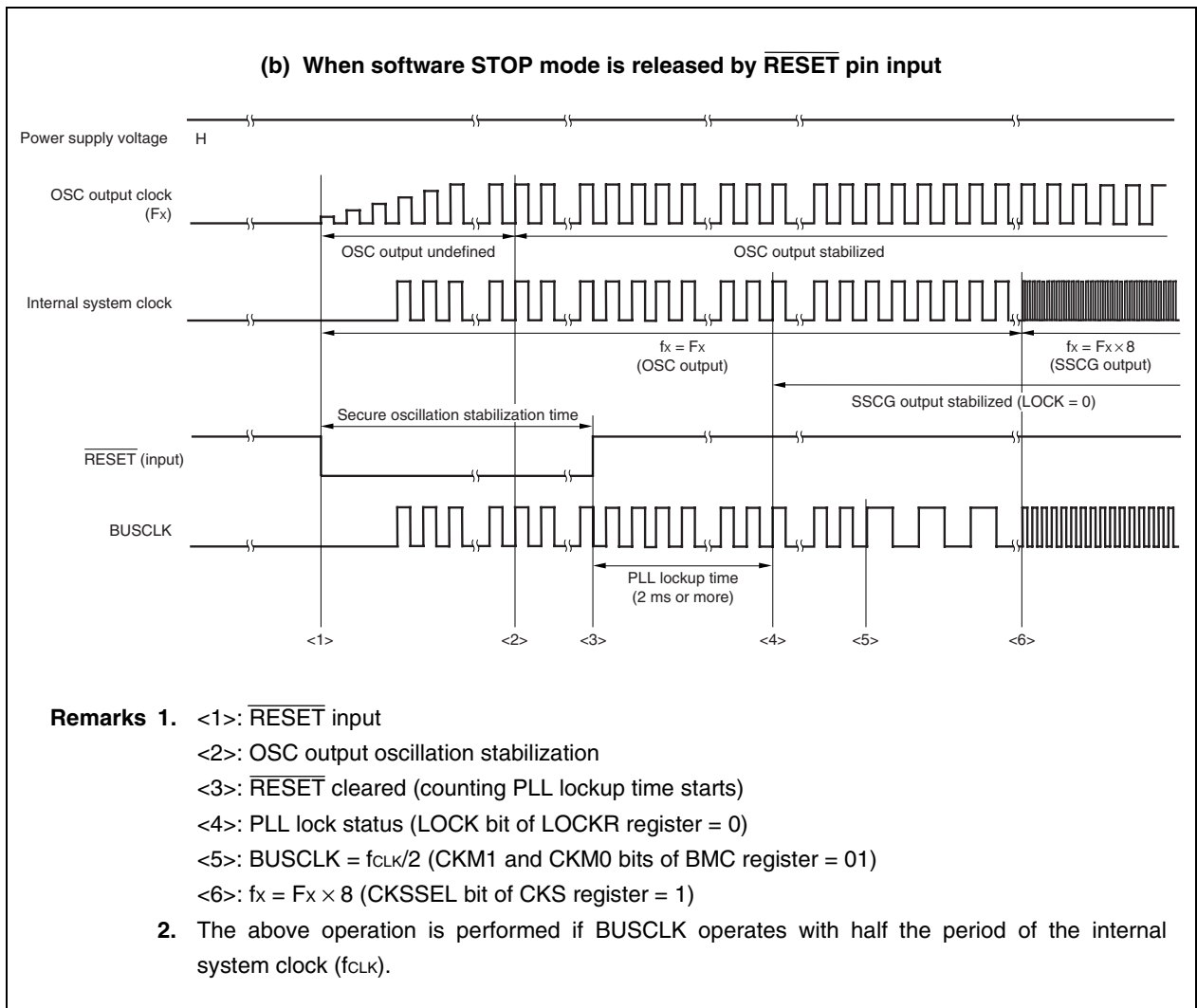


Figure 8-4. BUSCLK Operation When Software STOP Mode Is Released (2/2)



8.7 Securing Oscillation Stabilization Time

8.7.1 Oscillation stabilization time security specification

When the software STOP mode is released by a non-maskable interrupt request, an unmasked external maskable interrupt request (INTPn)^{Note} (n = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, or L1), or an unmasked internal maskable interrupt request of a peripheral function that can operate in the software STOP mode (INTRSUM), the oscillation stabilization time set by the OSTS register is secured (see **Figure 8-4 (a)**). About 2 ms after that, program execution is started. When the software STOP mode is released by $\overline{\text{RESET}}$ pin input, the oscillation stabilization time does not elapse. Therefore, secure the oscillation stabilization time by the low-active width of $\overline{\text{RESET}}$ (see **Figure 8-4 (b)**).

The internal time base counter (TBC) is used as a timer that counts and secures the oscillation stabilization time.

Oscillation stabilization time = Count time of TBC

Note When level detection is set, the software STOP mode cannot be released.

8.7.2 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.

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 OSTS2 to OSTS0 bits of the OSTS register, and the next counting time can be set.

★

Table 8-11. Counting Time Examples

OSTS2 to OSTS0 Bits			Oscillation Stabilization Time	Counting Time	
				$F_x = 16.6 \text{ MHz}$	$f_x = 12.5 \text{ MHz}$
0	0	0	$F_x/2^{13}$	0.49 ms	0.66 ms
0	0	1	$F_x/2^{15}$	1.97 ms	2.62 ms
0	1	0	$F_x/2^{16}$	3.95 ms	5.24 ms
0	1	1	$F_x/2^{17}$	7.90 ms	10.49 ms
1	0	0	$F_x/2^{18}$	15.79 ms	20.97 ms
1	0	1	$F_x/2^{19}$	31.58 ms	41.94 ms
1	1	0	$F_x/2^{20}$	63.17 ms	83.89 ms
1	1	1	$F_x/2^{21}$	126.33 ms	167.77 ms

Remark F_x : OSC output clock

CHAPTER 9 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)

9.1 Timer C

9.1.1 Features

Timer C is a 16-bit timer/counter that can perform the following operations.

- Interval timer function
- PWM output
- External signal cycle measurement

9.1.2 Function overview

- 16-bit timer/counter: 6 channels (no capture operation for 2 channels)
- Capture/compare common registers: 12 sources
- Interrupt request sources
 - Capture/match interrupt requests: 12 sources
 - Capture register: Generates INTCCm0 or INTCCm1 at INTPCm0 or INTPCm1 input
 - Compare register: Generates INTCCn0 or INTCCn1 at match signal of CCCn0 or CCCn1
 - Overflow interrupt requests: 6 sources
- Timer/counter count clock sources: 2 types
(Selection of external pulse input or main 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: 6

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

9.1.3 Basic configuration

Table 9-1. Timer C Configuration

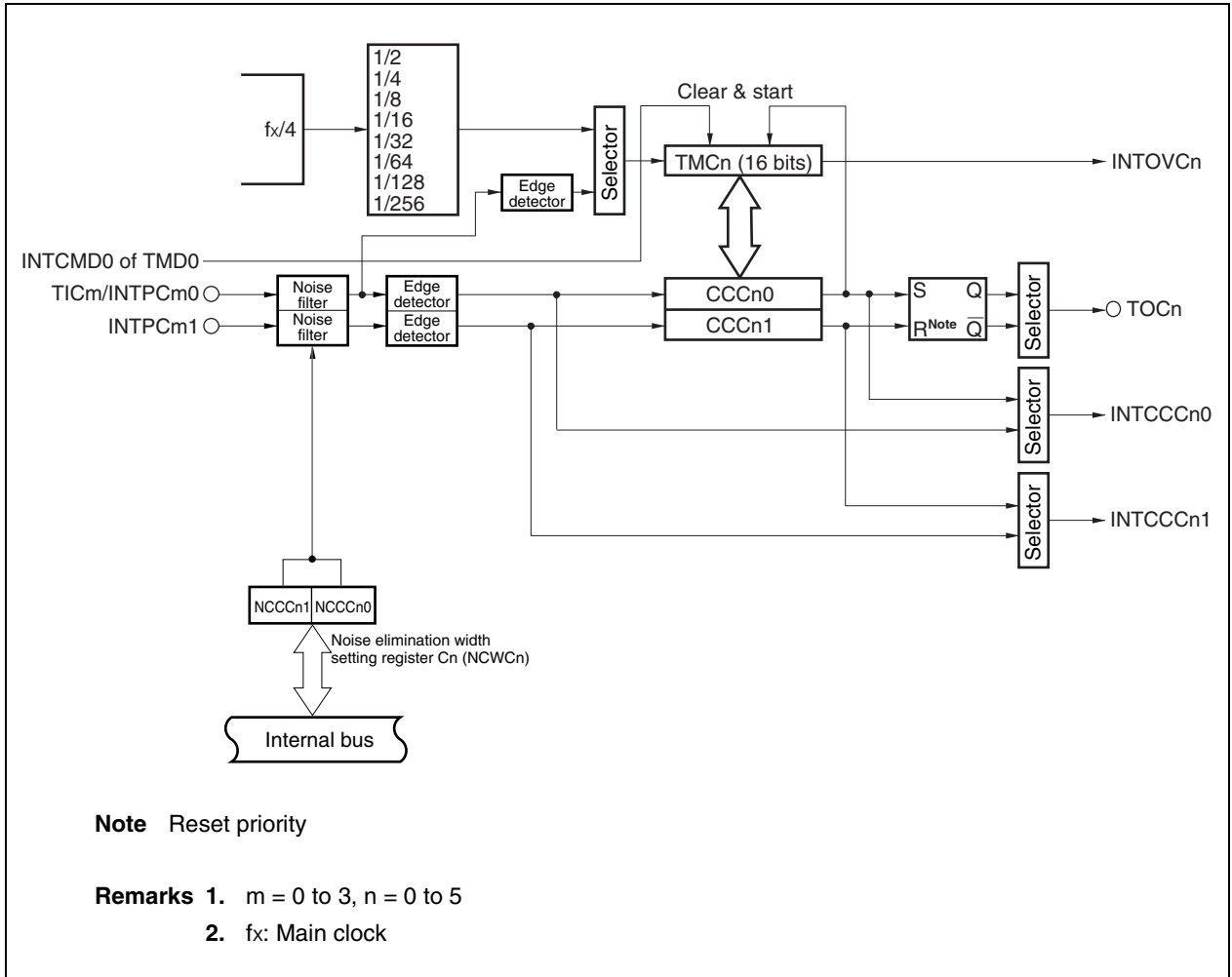
Timer	Count Clock	Register	Read/Write	Generated Interrupt Signal	Capture Trigger	Timer Output S/R	Other Functions
Timer C	fx/8, fx/16, fx/32, fx/64, fx/128, fx/256, fx/512, fx/1,024	TMC0	Read	INTOVC0	–	–	
		CCC00	Read/write	INTCCC00	INTPC00	TOC0 (S)	
		CCC01	Read/write	INTCCC01	INTPC01	TOC0 (R)	
		TMC1	Read	INTOVC1	–	–	
		CCC10	Read/write	INTCCC10	INTPC10	TOC1 (S)	
		CCC11	Read/write	INTCCC11	INTPC11	TOC1 (R)	
		TMC2	Read	INTOVC2	–	–	
		CCC20	Read/write	INTCCC20	INTPC20	TOC2 (S)	
		CCC21	Read/write	INTCCC21	INTPC21	TOC2 (R)	
		TMC3	Read	INTOVC3	–	–	
		CCC30	Read/write	INTCCC30	INTPC30	TOC3 (S)	
		CCC31	Read/write	INTCCC31	INTPC31	TOC3 (R)	
		TMC4	Read	INTOVC4	–	–	
		CCC40	Read/write	INTCCC40	–	TOC4 (S)	A/D conversion start trigger
		CCC41	Read/write	INTCCC41	–	TOC4 (R)	A/D conversion start trigger
		TMC5	Read	INTOVC5	–	–	
		CCC50	Read/write	INTCCC50	–	TOC5 (S)	A/D conversion start trigger
		CCC51	Read/write	INTCCC51	–	TOC5 (R)	A/D conversion start trigger

Remark fx: Main clock

S/R: Set/reset

(1) Timer C (16-bit timer/counter)

Figure 9-1. Timer C Block Diagram



9.1.4 Timer C

(1) Timers C0 to C5 (TMC0 to TMC5)

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 to 5).

TMCn is read-only in 16-bit units.

- Cautions**
1. The TMCn register can only be read. If the TMCn register is written, the subsequent operation is undefined.
 2. If the CAECn bit of the TMCCn0 register is cleared to 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																	FFFFF620H	0000H
TMC2																	FFFFF640H	0000H
TMC3																	FFFFF660H	0000H
TMC4																	FFFFF680H	0000H
TMC5																	FFFFF6A0H	0000H

TMCn performs the count-up operations of an internal count clock or external count clock. Timer start and stop are controlled by the CECn bit of timer mode control register Cn0 (TMCCn0) (n = 0 to 5).

The internal or external count clock is selected by the ETICn bit of timer mode control register Cn1 (TMCCn1) (n = 0 to 5).

(a) Selection of the external count clock

TMCn operates as an event counter.

When the ETICn bit of timer mode control register Cn1 (TMCCn1) is set to 1, TMCm counts the valid edges of the external clock input (TICm), synchronized with the internal count clock. The valid edge is specified by valid edge select register Cm (SESCm) (m = 0 to 3, n = 0 to 5).

Caution When the INTPCm0/TICm pin is used as TICm (external clock input pin), disable the INTPCm0 interrupt or set CCCm0 to compare mode (m = 0 to 3, n = 0 to 5).

(b) Selection of the internal count clock

TMCn operates as a free-running timer.

When an internal clock is specified as the count clock by timer mode control register Cn1 (TMCCn1), TMCn is counted up for each input clock cycle specified by the CSCn0 to CSCn2 bits of the TMCCn0 register (n = 0 to 5).

Division by the prescaler can be selected for the count clock from among $fx/8$, $fx/16$, $fx/32$, $fx/64$, $fx/128$, $fx/256$, $fx/512$, and $fx/1,024$ by the TMCCn0 register (fx: main clock).

An overflow interrupt can be generated if the timer overflows. Also, the timer can be stopped following an overflow by setting the OSTCn 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

- CAECn bit of TMCCn0 register = 0
- Reset input

(b) Synchronous reset

- CECn 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 (CCLRCn bit of the TMCCn1 register = 1)

(2) Capture/compare registers Cn0 and Cn1 (CCCn0 and CCCn1) (n = 0 to 5)

These capture/compare registers (Cn0 and Cn1) are 16-bit registers.

They can be used as capture registers or compare registers according to the CMSCn0 and CMSCn1 bit specifications of timer mode control register Cn1 (TMCCn1) (n = 0 to 5).

These registers can be read or written in 16-bit units. (However, write operations can only be performed in compare mode.)

Caution The CCC40, CCC41, CCC50, and CCC51 registers can only be used as compare registers. They cannot be used as capture registers.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CCC0n																	FFFFF602H, FFFFF604H	0000H
CCC1n																	FFFFF622H, FFFFF624H	0000H
CCC2n																	FFFFF642H, FFFFF644H	0000H
CCC3n																	FFFFF662H, FFFFF664H	0000H
CCC4n																	FFFFF682H, FFFFF684H	0000H
CCC5n																	FFFFF6A2H, FFFFF6A4H	0000H

Remark n = 0, 1

(a) Setting these registers as capture registers (CMSCn0 and CMSCn1 of TMCCn1 = 0)

When these registers are set as capture registers, the valid edges of the corresponding external interrupt signals INTPCm0 and INTPCm1 are detected as capture triggers. The timer TMCm is synchronized with the capture trigger, and the value of TMCm is latched in the CCCm0 and CCCm1 registers (capture operation) ($m = 0$ to 3, $n = 0$ to 5).

The valid edge of the INTPCm0 pin is specified (rising edge, falling edge, or both rising and falling edges) according to the IESC0m1 and IESC0m0 bits of the SESCm register, and the valid edge of the INTPCm1 pin is specified according to the IESC1m1 and IESC1m0 bits of the SESCm register.

The capture operation is performed asynchronously to the count clock. The latched value is held in the capture register until another capture operation is performed.

When the CAECn bit of timer mode control register Cn0 (TMCCn0) is 0, 0000H is read.

If these registers are specified as capture registers, an interrupt is generated by detecting the valid edge of signals INTPCm0 and INTPCm1.

Caution If the capture operation contends with the timing of disabling the TMCm register from counting (when the CECn bit of the TMCCn0 register = 0), the captured data becomes undefined. In addition, the INTCCm0 and INTCCm1 interrupts do not occur ($m = 0$ to 3, $n = 0$ to 5).

(b) Setting these registers as compare registers (CMSCn0 and CMSCn1 of TMCCn1 = 1)

When these registers are set as compare registers, the TMCn and register values are compared for each count clock, and an interrupt is generated by a match. If the CCLRCn bit of timer mode control register Cn1 (TMCCn1) is set to 1, the TMCn value is cleared to 0 at the same time as a match with the CCCn0 register (it is not cleared to 0 by a match with the CCCn1 register) ($n = 0$ to 5).

A compare register is equipped with a set/reset function. The corresponding timer output (TOCn) is set or reset, in synchronization 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 CAECn bit to 1 first. If the CAECn bit is 0, the data that is written will be invalid.
 2. Write to capture/compare registers Cn0 and Cn1 after setting them as compare registers via TMCCn0 and TMCCn1 register settings. If they are set as capture registers (CMSCn0 and CMSCn1 bits of TMCCn1 register = 0), no data is written even if a write operation is performed to CCCm0 and CCCm1 ($m = 0$ to 3, $n = 0$ to 5).
 3. When these registers are set as compare registers, INTPCn0 and INTPCn1 cannot be used ($n = 0$ to 5).

When using these registers as an external interrupt input pin, use the $\overline{\text{INTP65}}$ and $\overline{\text{INTP66}}$ pins.

9.1.5 Control registers

(1) Timer mode control registers C00 to C50 (TMCC00 to TMCC50)

The TMCCn0 registers control the operation of TMCn (n = 0 to 5).

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

Be sure to clear bits 3 and 2 to 0. If they are set to 1, the operation is not guaranteed.

Cautions 1. The CAECn 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 CAECn bit has been set. Also, to use external pins related to the timer function when timer C is used, be sure to set the CAECn bit to 1 after setting the external pins to control mode.

2. When conflict occurs between an overflow and a TMCCn0 register write, the OVFCn bit value becomes the value written during the TMCCn0 register write (n = 0 to 5).

(1/2)

	<7>	6	5	4	3	2	<1>	<0>	Address	After reset
TMCC00	OVFC0	CSC02	CSC01	CSC00	0	0	CEC0	CAEC0	FFFFFF606H	00H
TMCC10	OVFC1	CSC12	CSC11	CSC10	0	0	CEC1	CAEC1	FFFFFF626H	00H
TMCC20	OVFC2	CSC22	CSC21	CSC20	0	0	CEC2	CAEC2	FFFFFF646H	00H
TMCC30	OVFC3	CSC32	CSC31	CSC30	0	0	CEC3	CAEC3	FFFFFF666H	00H
TMCC40	OVFC4	CSC42	CSC41	CSC40	0	0	CEC4	CAEC4	FFFFFF686H	00H
TMCC50	OVFC5	CSC52	CSC51	CSC50	0	0	CEC5	CAEC5	FFFFFF6A6H	00H

Bit position	Bit name	Function
7	OVFCn	<p>This is a flag that indicates TMCn overflow.</p> <p>0: No overflow occurs 1: Overflow occurs</p> <p>When TMCn has counted up from FFFFH to 0000H, the OVFCn bit becomes 1 and an overflow interrupt request (INTOVFn) 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 (CMSCn0 bit of TMCCn1 register = 1) and clearing is enabled for a match when TMCn and CCCn0 are compared (CCLRCn bit of TMCCn1 register = 1), then TMCn is considered to be cleared and the OVFCn bit does not become 1. Also, no INTOVFn interrupt is generated.</p> <p>The OVFCn bit retains the value 1 until 0 is written directly or until an asynchronous reset is performed because the CAECn bit is 0. An interrupt operation due to an overflow is independent of the OVFCn bit, and the interrupt request flag (OVCIFn) for INTOVFn is not affected even if the OVFCn bit is manipulated. If an overflow occurs while the OVFCn bit is being read, the flag value changes, and the change is reflected when the next read operation occurs.</p>

Remark n = 0 to 5

Bit position	Bit name	Function																																				
6 to 4	CSCn2 to CSCn0	<p>Selects the TMCn internal count clock.</p> <table border="1"> <thead> <tr> <th>CSCn2</th> <th>CSCn1</th> <th>CSCn0</th> <th>Count cycle</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>fx/8</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>fx/16</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>fx/32</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>fx/64</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>fx/128</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>fx/256</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>fx/512</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>fx/1,024</td> </tr> </tbody> </table> <p>Caution The CSCn2 to CSCn0 bits must not be changed during timer operation. If they are to be changed, they must be changed after setting the CECn bit to 0. If these bits are overwritten during timer operation, operation cannot be guaranteed.</p> <p>Remark fx: Main clock</p>	CSCn2	CSCn1	CSCn0	Count cycle	0	0	0	fx/8	0	0	1	fx/16	0	1	0	fx/32	0	1	1	fx/64	1	0	0	fx/128	1	0	1	fx/256	1	1	0	fx/512	1	1	1	fx/1,024
CSCn2	CSCn1	CSCn0	Count cycle																																			
0	0	0	fx/8																																			
0	0	1	fx/16																																			
0	1	0	fx/32																																			
0	1	1	fx/64																																			
1	0	0	fx/128																																			
1	0	1	fx/256																																			
1	1	0	fx/512																																			
1	1	1	fx/1,024																																			
1	CECn	<p>Controls the operation of TMCn.</p> <p>0: Count disabled (stops at 0000H and does not operate)</p> <p>1: Counting operation is performed</p> <p>Caution When CECn = 0, the external pulse output (TOCn) becomes inactive (the active level of TOCn output is set by the ALVCn bit of the TMCCn1 register).</p>																																				
0	CAECn	<p>Controls the internal count clock.</p> <p>0: The entire TMCn unit is asynchronously reset. The supply of clocks to the TMCn unit stops.</p> <p>1: Clocks are supplied to the TMCn unit</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. When the CAECn bit is cleared to 0, the TMCn unit can be asynchronously reset. 2. When CAECn = 0, the TMCn unit is in a reset state. Therefore, to operate TMCn, the CAECn bit must be set to 1. 3. When the CAECn bit is changed from 1 to 0, all registers of the TMCn unit are initialized. When CAECn is set to 1 again, the TMCn unit registers must be set again. 																																				

Remark n = 0 to 5

(2) Timer mode control registers C01 to C51 (TMCC01 to TMCC51)

The TMCCn1 registers control the operation of TMCn (n = 0 to 5).

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

- 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 CECn bit of the TMCCn0 register to 0. If these bits are overwritten during timer operation, operation cannot be guaranteed (n = 0 to 5).
 2. If the ENTOCn and ALVCn bits are changed at the same time, a glitch (spike shaped noise) may be generated in the TOCn pin output. Either create a circuit configuration that will not malfunction even if a glitch is generated or make sure that the ENTOCn and ALVCn bits do not change at the same time (n = 0 to 5).
 3. TOCn output is not changed by an external interrupt signal (INTPCm0 or INTPCm1). To use the TOCn signal, specify that the capture/compare registers are compare registers (CMSCn0 and CMSCn1 bits of TMCCn1 register = 1) (m = 0 to 3, n = 0 to 5).
 4. The TMCC41 and TMCC51 registers can be used only when the CCC40, CCC41, CCC50, and CCC51 registers are used as compare registers. They cannot be used when the CCC40, CCC41, CCC50, and CCC51 registers are used as capture registers.
 5. To clear the external interrupt signal by setting the ECLRCn bit, be sure to make the following setting.
 Count clock period of TMD0 > Count clock period of TMCn
 If this setting is not made, the falling edge of the INTCMD0 interrupt signal may not be detected.

Correct example (if count clock of TMD0 is slow)

CSD02 to CSD00 bits of TMCD0 register = 011: fx/64

CSCn2 to CSCn0 bits of TMCC0 register = 001: fx/16

Remark Resetting the flip-flop of the TOCn output takes precedence (n = 0 to 5).

	7	6	5	4	3	2	1	0	Address	After reset
TMCC01	OSTC0	ENTOC0	ALVC0	ETIC0	CCLRC0	ECLRC0	CMSC01	CMSC00	FFFFFF608H	20H
TMCC11	OSTC1	ENTOC1	ALVC1	ETIC1	CCLRC1	ECLRC1	CMSC11	CMSC10	FFFFFF628H	20H
TMCC21	OSTC2	ENTOC2	ALVC2	ETIC2	CCLRC2	ECLRC2	CMSC21	CMSC20	FFFFFF648H	20H
TMCC31	OSTC3	ENTOC3	ALVC3	ETIC3	CCLRC3	ECLRC3	CMSC31	CMSC30	FFFFFF668H	20H
TMCC41	OSTC4	ENTOC4	ALVC4	ETIC4	CCLRC4	ECLRC4	CMSC41	CMSC40	FFFFFF688H	20H
TMCC51	OSTC5	ENTOC5	ALVC5	ETIC5	CCLRC5	ECLRC5	CMSC51	CMSC50	FFFFFF6A8H	20H

Bit position	Bit name	Function
7	OSTCn	<p>Sets the operation when TMCn has overflowed.</p> <p>0: After the overflow, counting continues (free-running mode)</p> <p>1: After the overflow, the timer maintains the value 0000H, and counting stops (overflow stop mode). Counting is resumed by the following operation.</p> <p>When ECLRCn bit = 0: Writing 1 to CECn bit</p> <p>When ECLRCn bit = 1: Inputting valid edge to INTCMD0</p>
6	ENTOCn	<p>External pulse output is enabled/disabled (TOCn).</p> <p>0: External pulse output is disabled. Output of the ALVCn bit inactive level to the TOCn pin is fixed. The TOCn pin level is not changed even if a match signal from the corresponding compare register is generated.</p> <p>1: External pulse output is enabled. A compare register match causes TOCn output to change. However, if capture mode is set, TOCn output does not change. The ALVCn bit inactive level is output from the time when timer output is enabled until a match signal is first generated.</p> <p>Caution If either CCCn0 or CCCn1 is specified as a capture register, the ENTOCn bit must be cleared to 0.</p>
5	ALVCn	<p>Specifies the active level for external pulse output (TOCn).</p> <p>0: Active level is low level</p> <p>1: Active level is high level</p> <p>Caution The initial value of the ALVCn bit is 1.</p>
4	ETICn	<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 CSCn2 to CSCn0 bits of TMCCn0.</p> <p>1: Specifies the external clock (TICm). The valid edge can be selected according to the TESCM1 and TESCM0 bit specifications of SESCm.</p>

Remark m = 0 to 3
n = 0 to 5

★

Bit position	Bit name	Function
3	CCLRCn	Sets whether the clearing of TMCn is enabled or disabled during a compare operation. 0: Clearing is disabled 1: Clearing is enabled (if CCCn0 and TMCn match during a compare operation, TMCn is cleared)
2	ECLRCn	Enables/disables clearing TMCn by INTCMD0 (TMD0). 0: Disables clearing by INTCMD0. 1: Enables clearing by INTCMD0. After TMCn has been cleared, it resumes counting. Caution When ECLRCn = 1, unless a compare match interrupt (INTCMD0) is generated by TMD0, timer counting does not start (even if the CECn bit of the TMCCn0 register is set (1)).
1	CMSCn1	Selects the operation mode of the capture/compare register (CCcn1). 0: The register operates as a capture register (CCcm1) 1: The register operates as a compare register (CCcn1)
0	CMSCn0	Selects the operation mode of the capture/compare register (CCcn0). 0: The register operates as a capture register (CCcm0) 1: The register operates as a compare register (CCcn0)

Remark m = 0 to 3
n = 0 to 5

(3) Valid edge select registers C0 to C3 (SESC0 to SESC3)

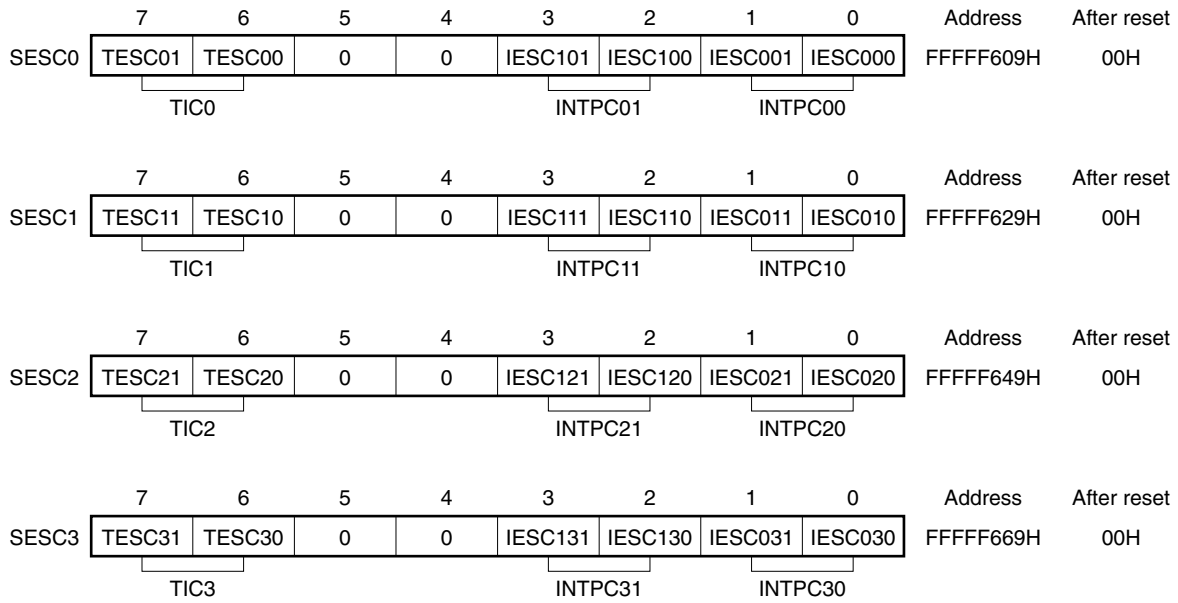
These registers specify the valid edge of an external interrupt request (INTPC00, INTPC01, INTPC10, INTPC11, INTPC20, INTPC21, INTPC30, INTPC31, and TIC0 to TIC3) from an external pin of TMCn (n = 0 to 3).

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 clear bits 5 and 4 to 0. If they are set to 1, the operation is not guaranteed.

- Cautions**
- 1. The various bits of the SESCn register must not be changed during timer operation (n = 0 to 3). If they are to be changed, they must be changed after setting the CECn bit of the TMCCn0 register to 0. If the SESCn register is overwritten during timer operation, operation cannot be guaranteed.**
 - 2. Even when using the INTPC00/TIC0, INTPC10/TIC1, INTPC20/TIC2, and INTPC30/TIC3 pins as INTPC00, INTPC10, INTPC20, and INTPC30, respectively, without using timer C, be sure to set the CAECn and CECn bits of timer mode control registers C00 to C03 (TMCC00 to TMCC30) to 1.**
 - 3. Before setting the trigger mode of the INPTC00, INTPC01, INTPC10, INTPC11, INTPC20, INTPC21, INTPC30, INTPC31, and TIC0 to TIC3 pins, set the PMCx register (x = 5 to 7). Then set the CAECn and CECn bits of the TMCCn0 register to 1 (n = 0 to 3). If the PMCx register is set after the SESCn register has been set, an illegal interrupt, incorrect counting, and incorrect clearing may occur depending on the timing of setting the PMCx register (n = 0 to 3, x = 5 to 7).**



Bit position	Bit name	Function															
7, 6	TESCn1, TESCn0 (n = 0 to 3)	Specifies the valid edge of the INTPCm0, INTPCm1, and TICm pins (m = 0 to 3). <table border="1" style="margin: 10px auto;"> <thead> <tr> <th>xESCn1</th> <th>xESCn0</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>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	xESCn1	xESCn0	Operation	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
xESCn1	xESCn0		Operation														
0	0		Falling edge														
0	1		Rising edge														
1	0	Setting prohibited															
1	1	Both rising and falling edges															
3, 2	IESCn1, IESCn0 (n = 10 to 13)																
1, 0	IESCn1, IESCn0 (n = 00 to 03)																

(4) Noise elimination width setting registers C0 to C3 (NCWC0 to NCWC3)

The NCWCn registers are used to set the noise elimination width of the digital noise filter of the timer C input pin (n = 0 to 3).

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

Be sure to clear bits 7 to 2 to 0. If they are set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
NCWC0	0	0	0	0	0	0	NCCC01	NCCC00	FFFFF610H	02H
NCWC1	0	0	0	0	0	0	NCCC11	NCCC10	FFFFF630H	02H
NCWC2	0	0	0	0	0	0	NCCC21	NCCC20	FFFFF650H	02H
NCWC3	0	0	0	0	0	0	NCCC31	NCCC30	FFFFF670H	02H

Bit position	Bit name	Function															
1, 0	NCCcn1, NCCcn0	Specifies the number of clocks by which noise is to be eliminated. <table border="1"> <thead> <tr> <th>NCCcn1</th> <th>NCCcn0</th> <th>Number of clocks by which noise is to be eliminated</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0 (through input)</td> </tr> <tr> <td>0</td> <td>1</td> <td>2</td> </tr> <tr> <td>1</td> <td>0</td> <td>3</td> </tr> <tr> <td>1</td> <td>1</td> <td>5</td> </tr> </tbody> </table>	NCCcn1	NCCcn0	Number of clocks by which noise is to be eliminated	0	0	0 (through input)	0	1	2	1	0	3	1	1	5
NCCcn1	NCCcn0	Number of clocks by which noise is to be eliminated															
0	0	0 (through input)															
0	1	2															
1	0	3															
1	1	5															
Remark 1 clock = $f_x/4$ f_x : Main clock																	

Remark n = 0 to 3

9.1.6 Operation

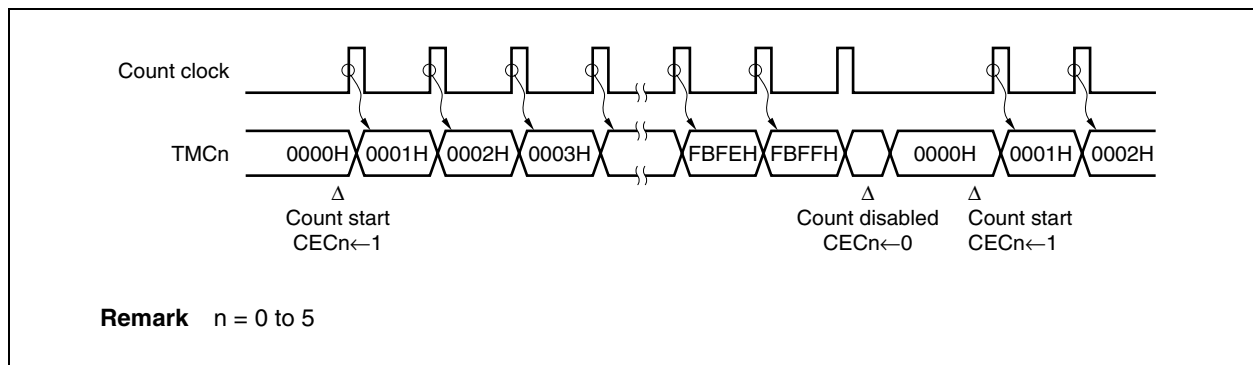
(1) Count operation

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) ($n = 0$ to 5).

When it operates as a free-running timer, if the CCCn0 or CCCn1 register and the TMCn count value match, an interrupt signal is generated and the timer output signal (TOCn) can be set or reset. Also, a capture operation that holds the TMCm count value in the CCCm0 or CCCm1 register is performed, in synchronization with the valid edge that was detected from the external interrupt request input pin as an external trigger ($m = 0$ to 3). The capture value is held until the next capture trigger is generated.

Caution When using the INTPCm0/TICm0 pin as TICm0 (external clock input pin), be sure to disable the INTPCm0 interrupt or set the CCCm0 register to compare mode ($m = 0$ to 3).

Figure 9-2. Basic Operation of Timer C



(2) Overflow

When the TMCn register has counted the count clock from FFFFH to 0000H, the OVFCn bit of the TMCCn0 register is set to 1, and an overflow interrupt (INTOVCn) is generated at the same time. However, if the CCCn0 register is set to compare mode (CMSCn0 bit = 1) and to the value FFFFH when match clearing is enabled (CCLRCn bit = 1), then the TMCn register is considered to be cleared and the OVFCn bit is not set to 1 when the TMCn register changes from FFFFH to 0000H. Also, the overflow interrupt (INTOVCn) is not generated.

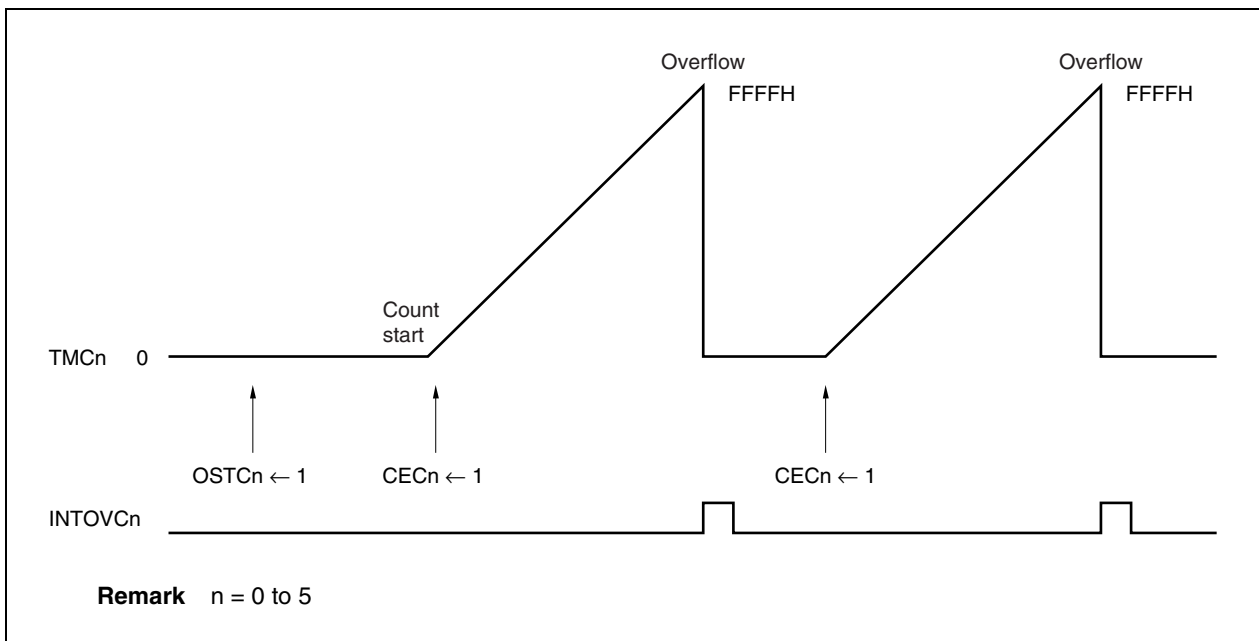
When the TMCn register is changed from FFFFH to 0000H because the CECn bit changes from 1 to 0, the TMCn register is considered to be cleared, but the OVFCn bit is not set to 1 and no INTOVCn interrupt is generated.

Also, timer operation can be stopped after an overflow by setting the OSTCn bit of the TMCCn1 register to 1. When the timer is stopped due to an overflow, the count operation is not restarted until the CECn bit of the TMCCn0 register is set to 1.

Operation is not affected even if the CECn bit is set to 1 during a count operation.

Remark n = 0 to 5

Figure 9-3. Operation After Overflow (When OSTCn = 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 CMSCn1 and CMSCn0 bits of the TMCCn1 register. If the CMSCn1 and CMSCn0 bits of the TMCCn1 register are cleared to 0, the register operates as a capture register.

A capture operation that captures and holds the TMCm count value asynchronously relative to the count clock is performed in synchronization with an external trigger. The valid edge that is detected from an external interrupt request input pin (INTPCm0 or INTPCm1) is used as an external trigger (capture trigger). The TMCm count value during counting is captured and held in the capture register, in synchronization with that capture trigger signal. The capture register value is held until the next capture trigger is generated.

Also, an interrupt request (INTCCm0 or INTCCm1) is generated by INTPCm0 or INTPCm1 signal input.

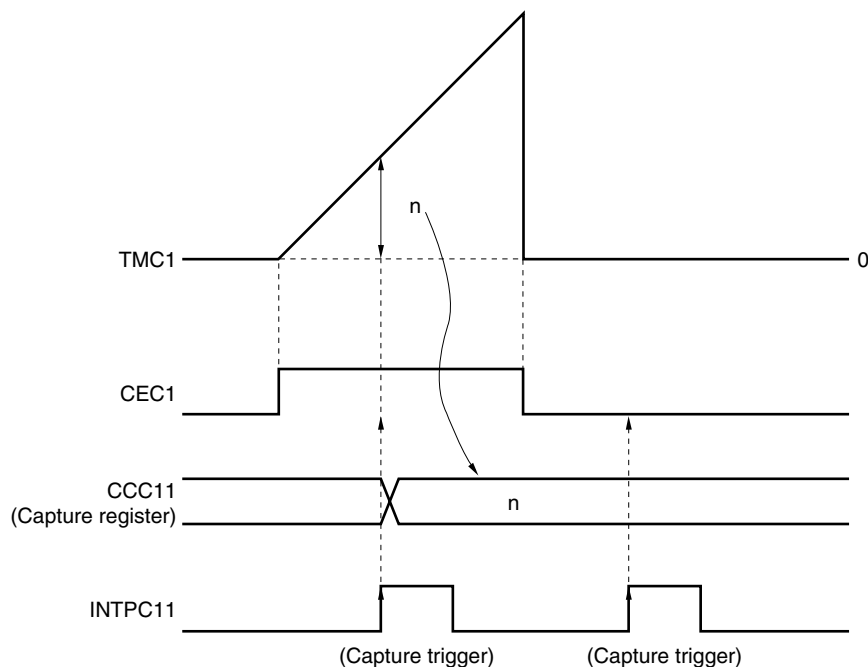
The valid edge of the capture trigger is set by valid edge select register Cm (SESCm).

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 m = 0 to 3

n = 0 to 5

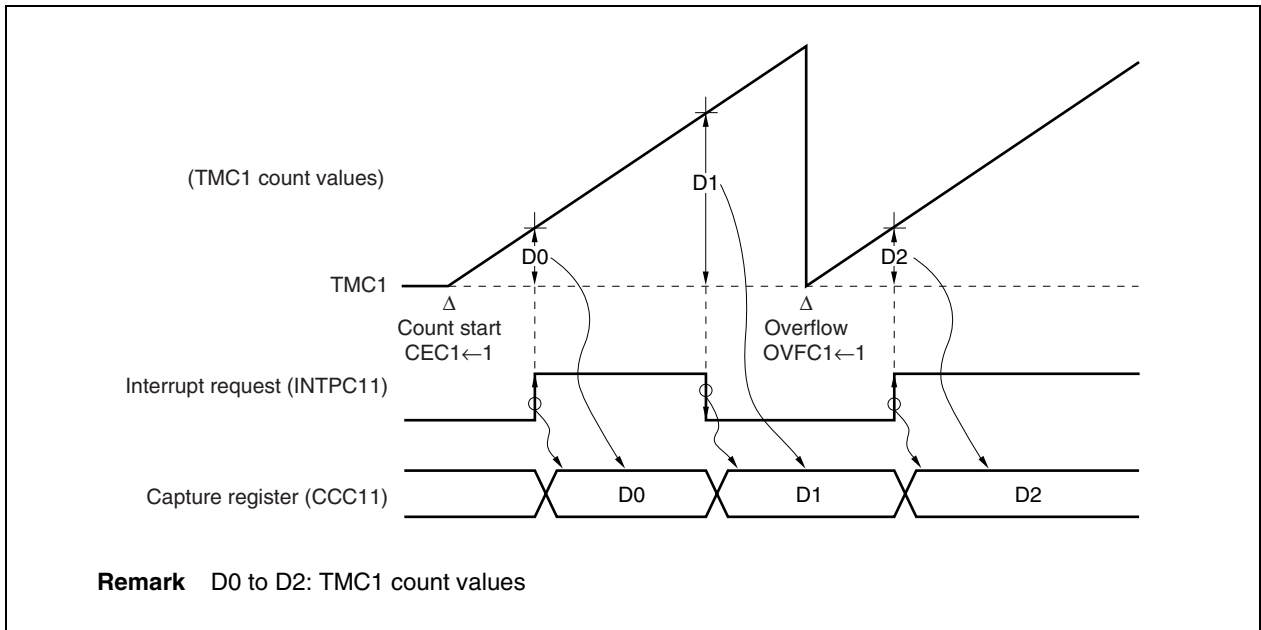
Figure 9-4. Capture Operation Example



Remarks 1. When the CEC1 bit is 0, no capture operation is performed even if INTPC11 is input.

2. Valid edge of INTPC11: Rising edge

Figure 9-5. 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 CMSCn1 and CMSCn0 bits of the TMCCn1 register. If the CMSCn1 and CMSCn0 bits of the TMCCn1 register are set to 1, 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 (TOCn) to change and an interrupt request signal (INTCCn0 or INTCCn1) to be generated at the same time.

If the CCCn0 or CCCn1 registers are cleared to 0000H, the 0000H after the TMCn register counts up from FFFFH to 0000H is judged as a match. In this case, the TMCn register value is cleared to 0 at the next count timing, however, this 0000H is not judged as a match. Also, the 0000H when the TMCn register begins counting is not judged as a match.

If match clearing is enabled (CCLRCn 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 to 5

Figure 9-6. Compare Operation Example (1/2)

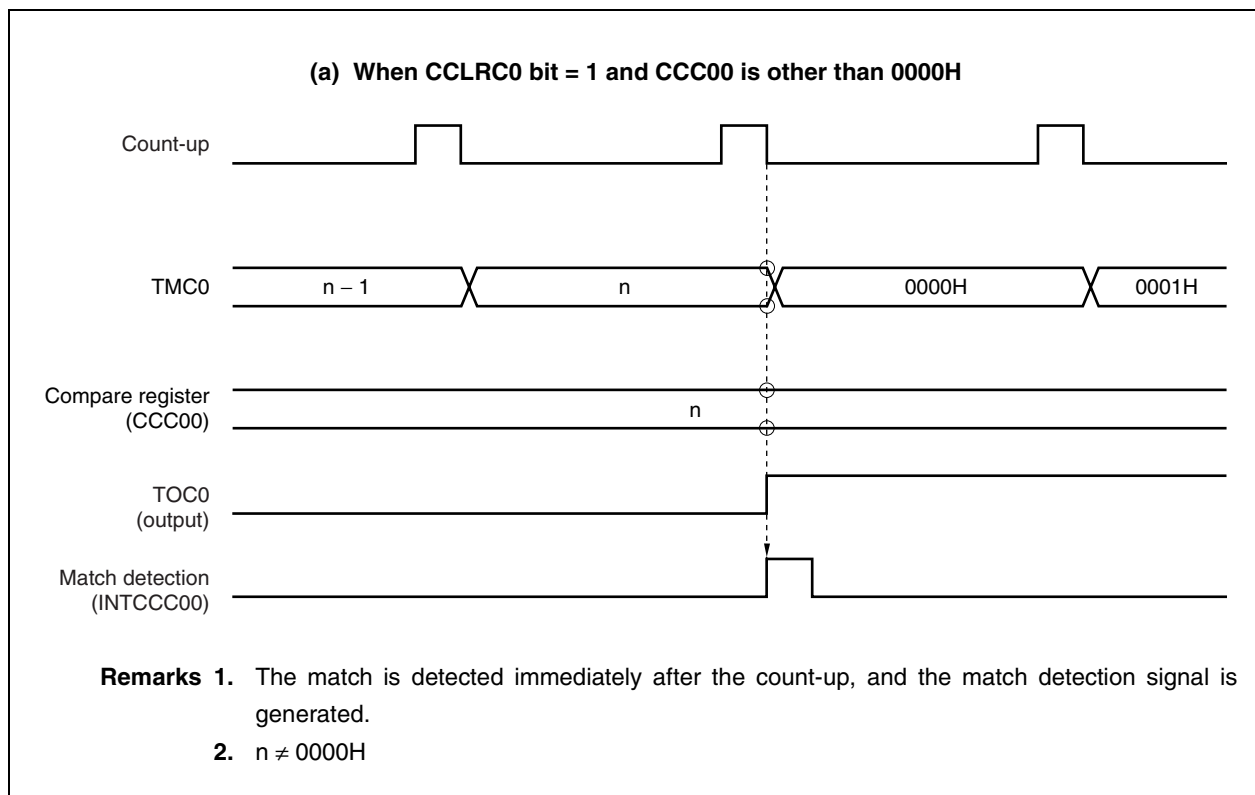
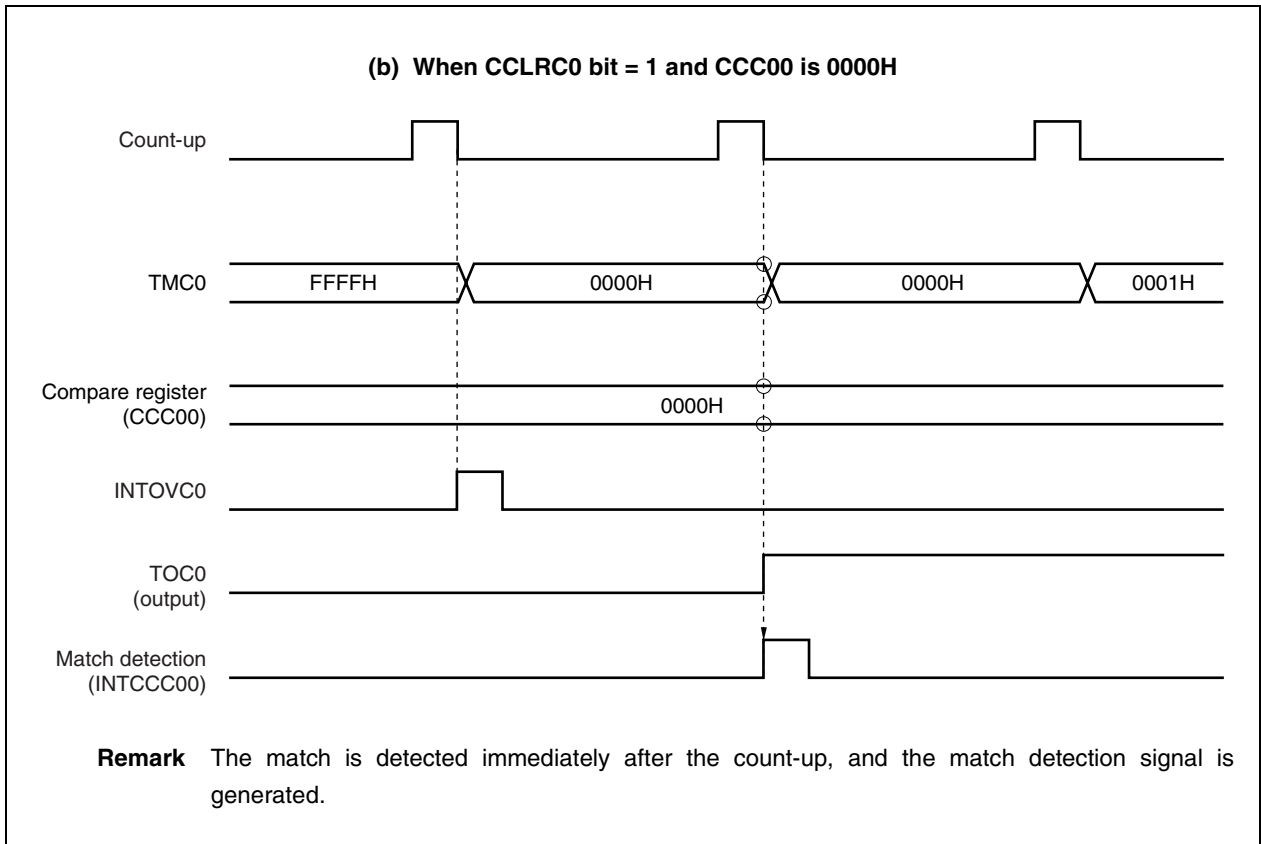


Figure 9-6. Compare Operation Example (2/2)



(5) External pulse output

Timer C has six timer output pins (TOCn).

An external pulse output (TOCn) is generated when a match of the two compare registers (CCCn0 and CCCn1) and the TMCn register is detected.

If a match is detected when the TMCn count value and the CCCn0 value are compared, the output level of the TOCn pin is set. Also, if a match is detected when the TMCn count value and the CCCn1 value are compared, the output level of the TOCn pin is reset.

The output level of the TOCn pin can be specified by the TMCCn1 register.

Remark n = 0 to 5

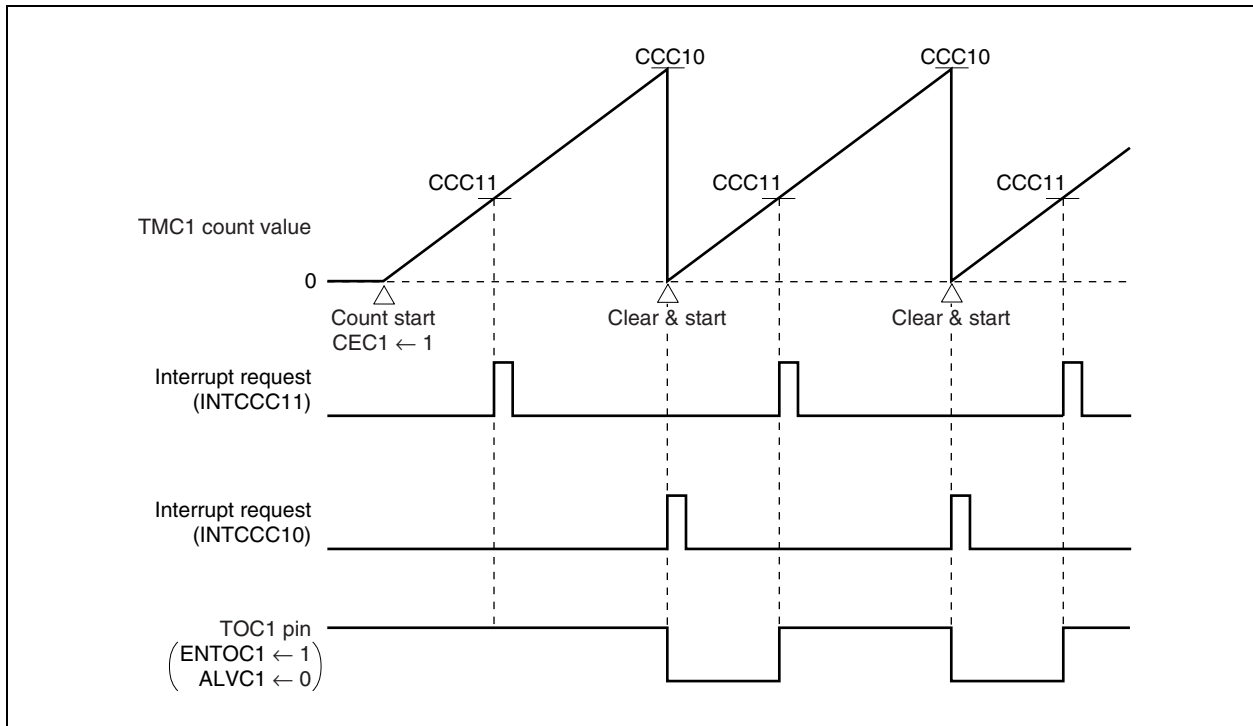
Table 9-2. TOCn Output Control

ENTOCn	ALVCn	TOCn Output	
		External Pulse Output	Output Level
0	0	Disable	High level
0	1	Disable	Low level
1	0	Enable	When the CCCn0 register is matched: low level When the CCCn1 register is matched: high level
1	1	Enable	When the CCCn0 register is matched: high level When the CCCn1 register is matched: low level

Remark n = 0 to 5

★

Figure 9-7. TMC1 Compare Operation Example (Set/Reset Output Mode)



9.1.7 Application examples

(1) Interval timer

By setting the TMCCn0 and TMCCn1 registers as shown in Figure 9-8, timer C operates as an interval timer that repeatedly generates interrupt requests with the value that was preset 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 to 0000H and an interrupt request signal (INTCCn0) is generated at the same time that the count operation resumes.

Remark n = 0 to 5

Figure 9-8. Contents of Register Settings When Timer C Is Used as Interval Timer

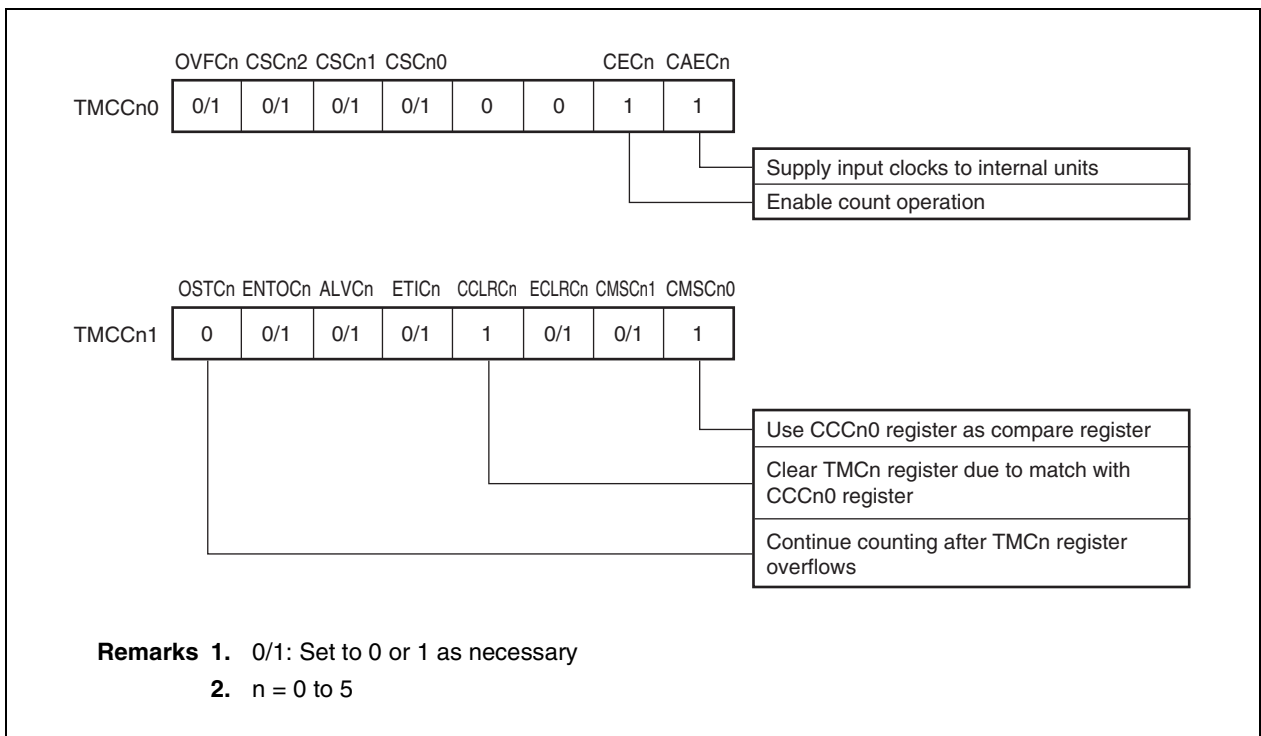
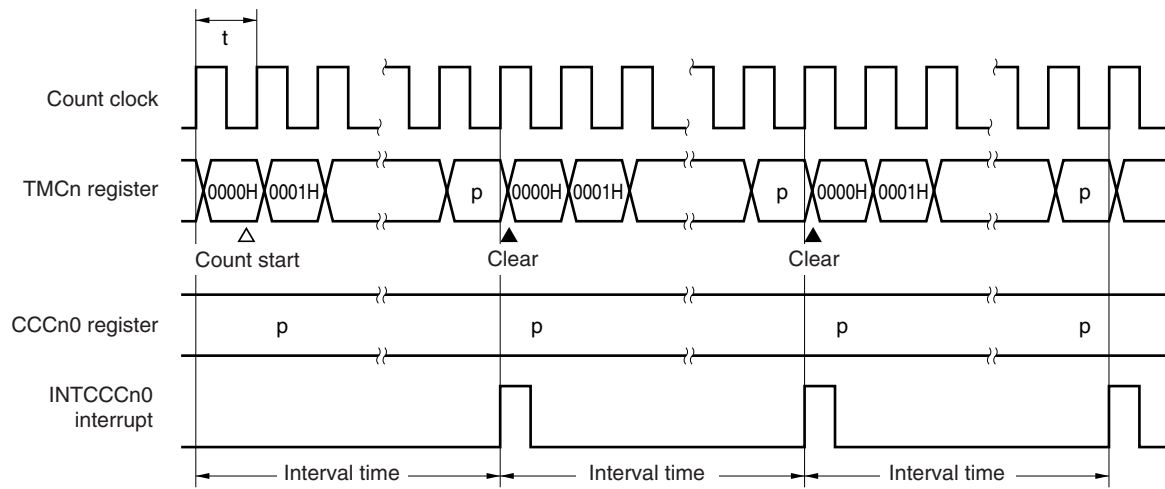


Figure 9-9. Interval Timer Operation Timing Example



- Remarks 1.** p: Setting value of CCCn0 register (0000H to FFFFH)
t: Count clock cycle
Interval time = $(p + 1) \times t$
- 2.** n = 0 to 5

(2) PWM output

By setting the TMCCn0 and TMCCn1 registers as shown in Figure 9-10, timer C can output a PWM signal, whose frequency is determined according to the setting of the CSCn2 to CSCn0 bits of the TMCCn0 register, with the values that were preset in the CCCn0 and CCCn1 registers determining the intervals.

When the counter value of the TMCn register matches the setting value of the CCCn0 register, the TOCn output becomes active. Then, when the counter value of the TMCn register matches the setting value of the CCCn1 register, the TOCn output becomes inactive. The TMCn 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 CSCn2 to CSCn0 bits of the TMCCn0 register can be output. When the setting value of the CCCn0 register and the setting value of the CCCn1 register are the same, the TOCn output remains inactive and does not change.

The active level of the TOCn output can be set by the ALVCn bit of the TMCCn1 register.

Remark n = 0 to 5

Figure 9-10. Contents of Register Settings When Timer C Is Used for PWM Output

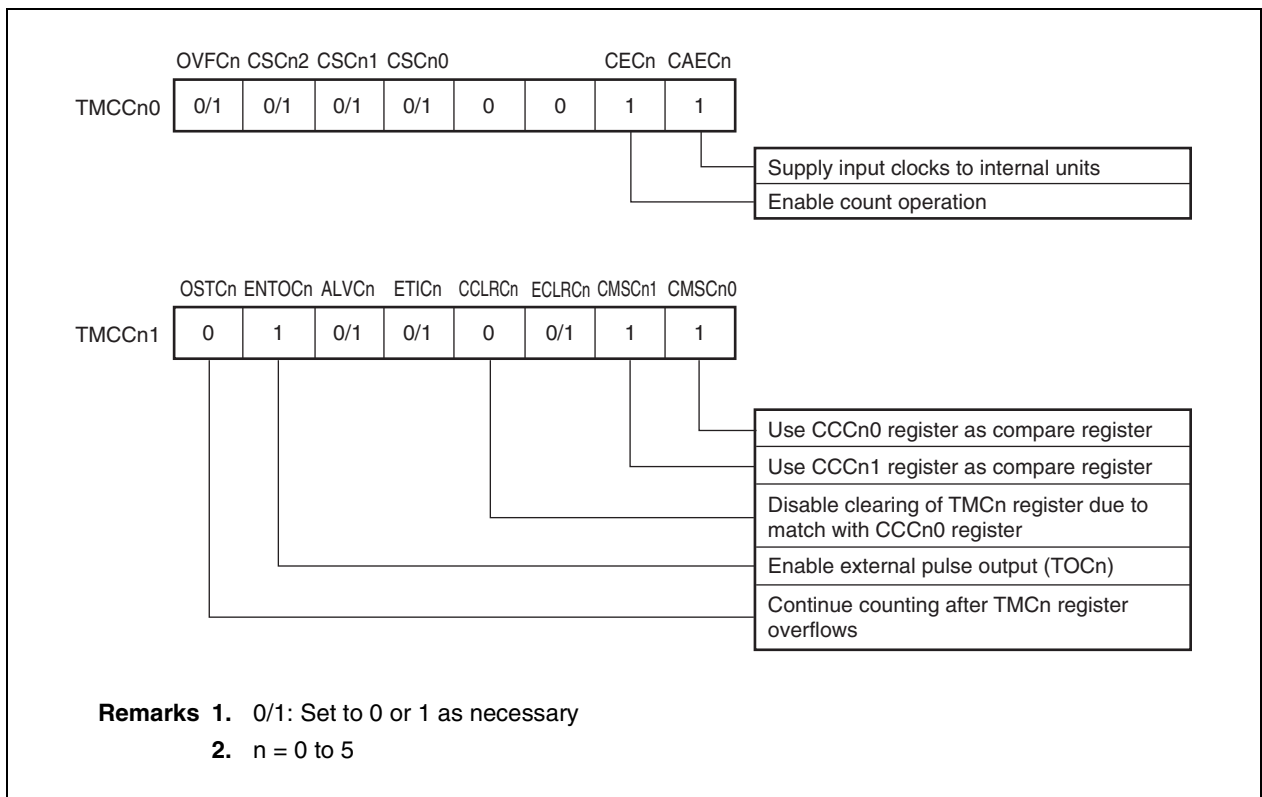
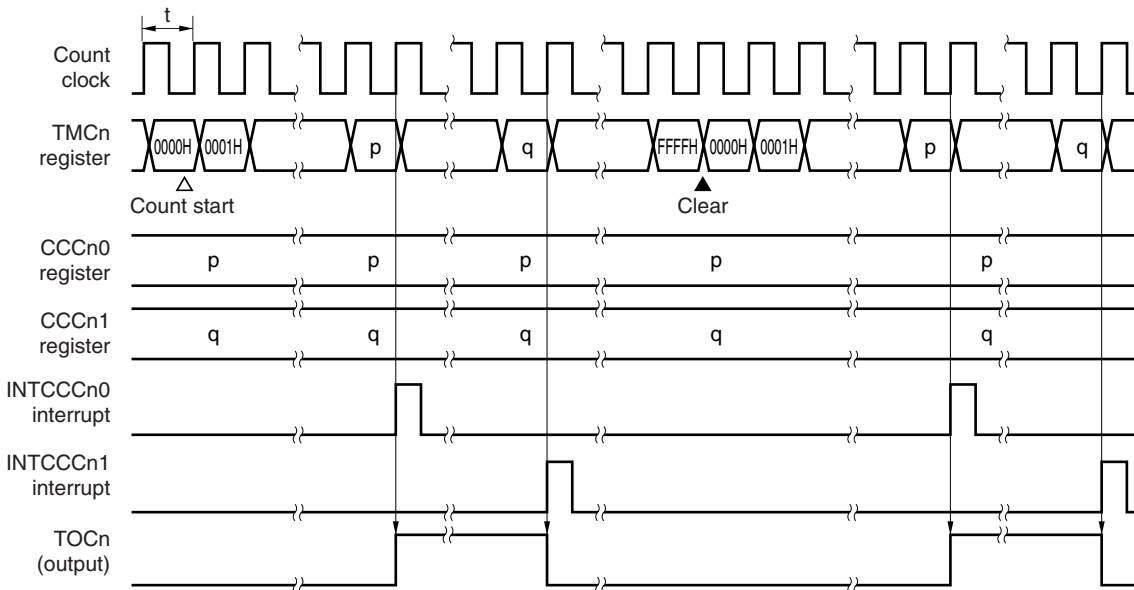


Figure 9-11. PWM Output Operation Timing Example



Remarks 1. p : Setting value of CCCn0 register (0000H to FFFFH)

q : Setting value of CCCn1 register (0000H to FFFFH)

$p \neq q$

t : Count clock cycle

PWM cycle = $65,536 \times t$

$$\text{Duty} = \frac{q - p}{65,536}$$

2. In this example, the active level of the TOCn output is set to the high level.

3. $n = 0$ to 5

(3) Cycle measurement

By setting the TMCCn0 and TMCCn1 registers as shown in Figure 9-12, timer C can measure the cycle of signals input to the INTPCm0 or INTPCm1 pin.

The valid edge of the INTPCm0 pin is selected according to the IESC0m1 and IESC0m0 bits of the SESCm register, and the valid edge of the INTPCm1 pin is selected according to the IESC1m1 and IESC1m0 bits of the SESCm register. Either the rising edge, the falling edge, or both edges can be selected as the valid edges of both pins.

If the CCCm0 register is set as a capture register, the valid edge input of the INTPCm0 pin is set as the trigger for capturing the TMCm register value in the CCCm0 register. When this value is captured, an INTCCm0 interrupt is generated.

Similarly, if the CCCm1 register is set as a capture register, the valid edge input of the INTPCm1 pin is set as the trigger for capturing the TMCm register value in the CCCm1 register. When this value is captured, an INTCCm1 interrupt is generated.

The cycle of signals input to the INTPCm0 pin is calculated by obtaining the difference between the TMCm register's count value (D_x) that was captured in the CCCm0 register according to the x -th valid edge input of the INTPCm0 pin and the TMCm register's count value ($D_{(x+1)}$) that was captured in the CCCm0 register according to the $(x+1)$ -th valid edge input of the INTPCm0 pin and multiplying the value of this difference by the cycle of the clock control signal.

The cycle of signals input to the INTPCm1 pin is calculated by obtaining the difference between the TMCm register's count value (D_x) that was captured in the CCCm1 register according to the x -th valid edge input of the INTPCm1 pin and the TMCm register's count value ($D_{(x+1)}$) that was captured in the CCCm1 register according to the $(x+1)$ -th valid edge input of the INTPCm1 pin and multiplying the value of this difference by the cycle of the clock control signal.

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

Figure 9-12. Contents of Register Settings When Timer C Is Used for Cycle Measurement

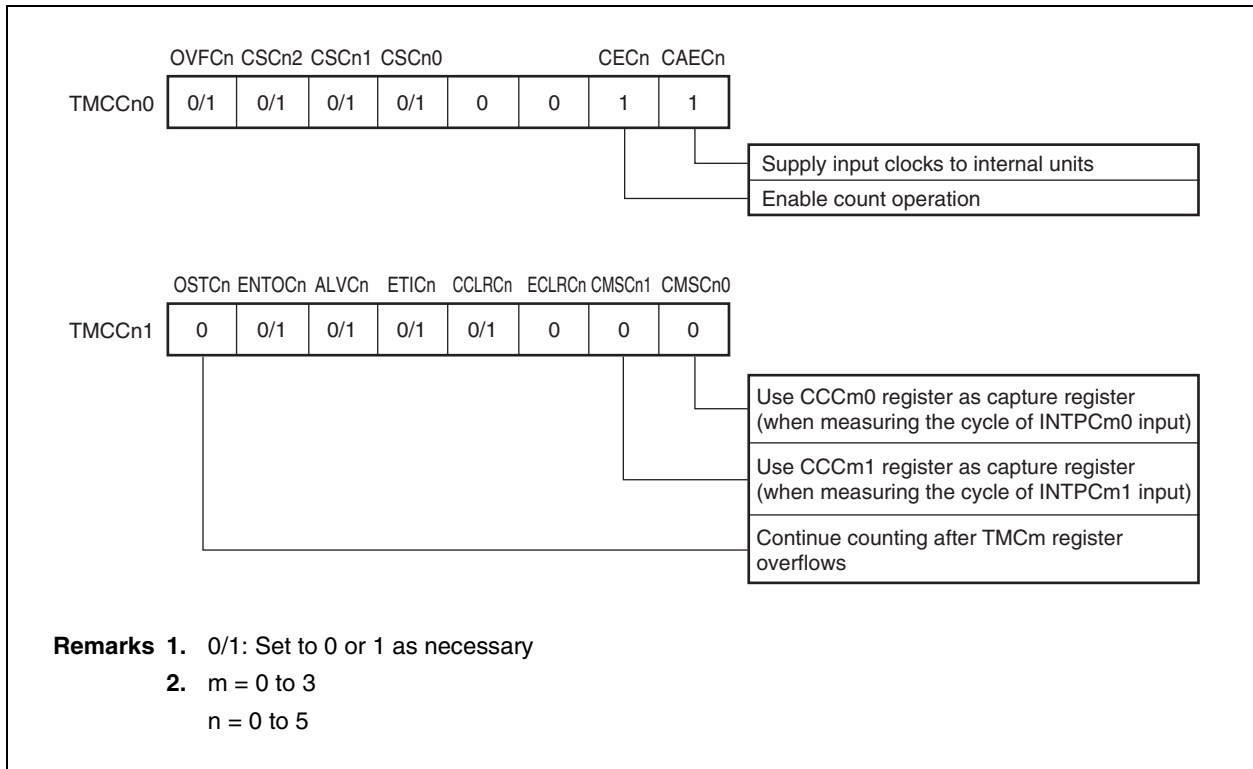
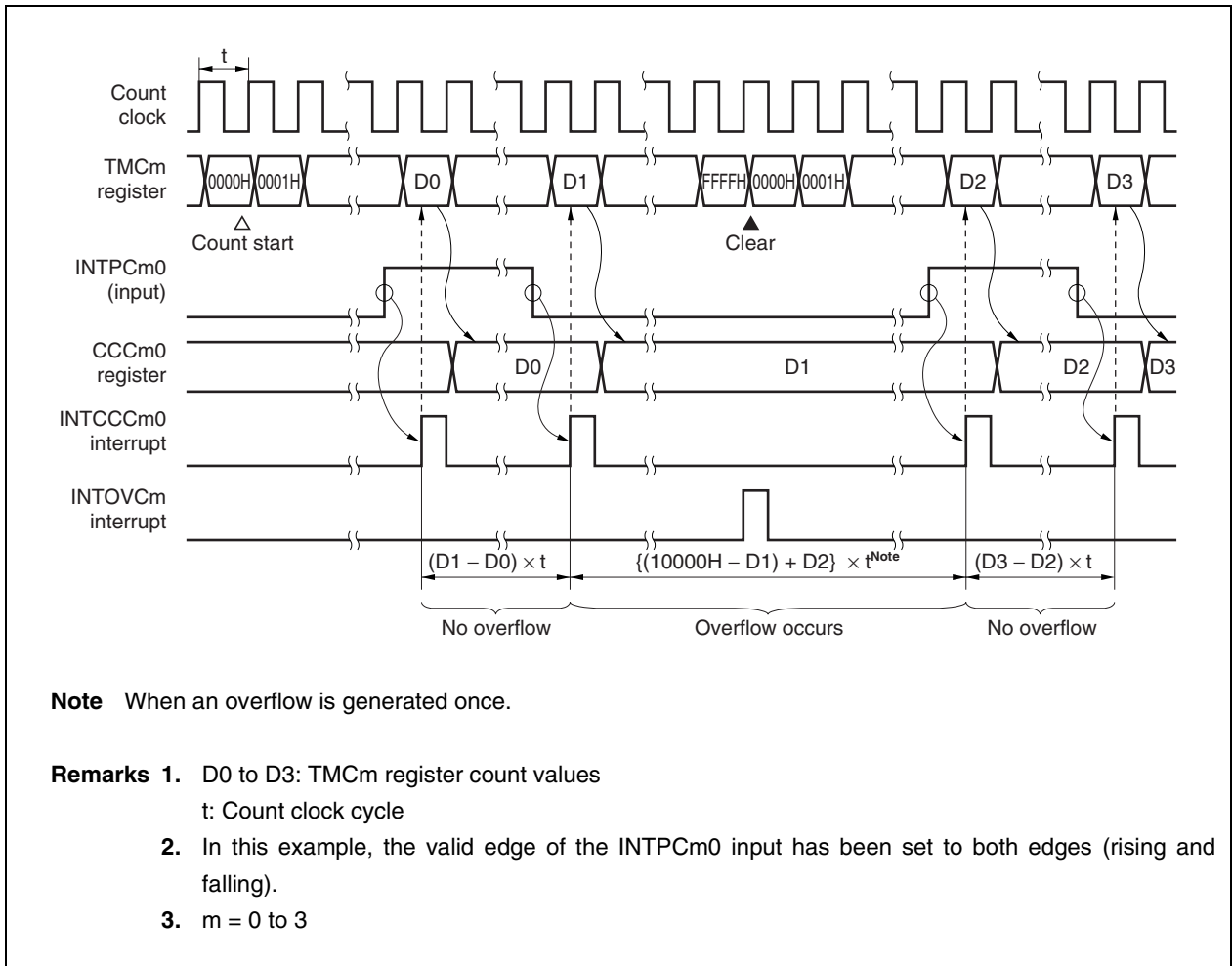


Figure 9-13. Cycle Measurement Operation Timing Example



9.1.8 Cautions

Various cautions concerning timer C are shown below.

- (1) If a conflict occurs between the reading of the CCCm0 register and a capture operation when the CCCm0 register is used in capture mode, an external trigger (INTPCm0) valid edge is detected and an external interrupt request (INTCCm0) is generated, however, the timer value is not stored in the CCCm0 register.
- (2) If a conflict occurs between the reading of the CCCm1 register and a capture operation when the CCCm1 register is used in capture mode, an external trigger (INTPCm1) valid edge is detected and an external interrupt request (INTCCm1) is generated, however, the timer value is not stored in the CCCm1 register.
- (3) The following bits and registers must not be rewritten during operation (CECn = 1).
 - CSCn2 to CSCn0 bits of TMCCn0 register
 - TMCCn1 register
 - SESCm register
- (4) The CAECn bit of the TMCCn0 register is a TMCn reset signal. To use TMCn, first set the CAECn bit to 1.
- (5) The digital noise elimination time $(0 \text{ to } 5) \times f_x/4$ are required to detect the valid edge of the external interrupt request signal (INTPCm0 or INTPCm1) or the external clock input (TICm). Therefore, edge detection will not be performed normally for changes that are less than the digital noise elimination time. For details of digital noise elimination, see **14.6.3 Timer C and timer ENC1 input pins**.
- (6) The operation of an external interrupt request signal (INTCCn0 or INTCCn1) 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 an interrupt indicating a match with the TMCn register.
- (7) If the ENTOCn and ALVCn bits are changed at the same time, a glitch (spike shaped noise) may be generated in the TOCn pin output. Either create a circuit configuration that will not malfunction even if a glitch is generated or make sure that the ENTOCn and ALVCn bits are not changed at the same time.
- (8) Even when using the INTPC00/TIC0, INTPC10/TIC1, INTPC20/TIC2, and INTPC30/TIC3 pins as INTPC00, INTPC10, INTPC20, and INTPC30, respectively, without using timer C, be sure to set the CAECn and CECn bits of timer mode control registers C00 to C03 (TMCC00 to TMCC30) to 1.

Remark m = 0 to 3
n = 0 to 5

9.2 Timer D

9.2.1 Features

Timer D functions as a 16-bit interval timer.

9.2.2 Function overview

- 16-bit interval timer: 4 channels
- Compare registers: 4
- Interrupt request sources: 4 sources
- Count clock selected from divisions of main clock

9.2.3 Basic configuration

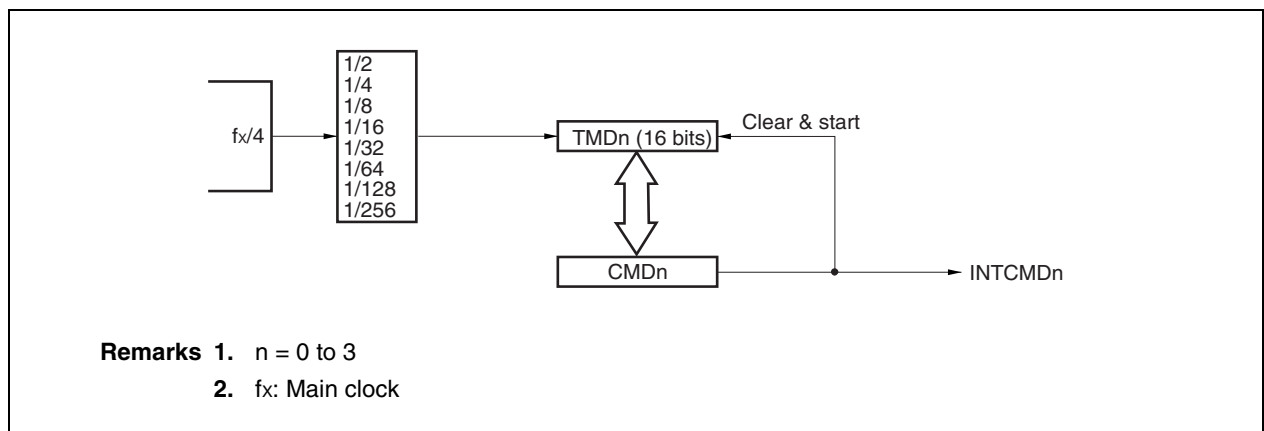
Table 9-3. Timer D Configuration

Timer	Count Clock	Register	Read/Write	Generated Interrupt Signal	Capture Trigger	Timer Output S/R	Other Functions
Timer D	fx/8, fx/16, fx/32, fx/64, fx/128, fx/256, fx/512, fx/1,024	TMD0	Read	–	–	–	–
		CMD0	Read/write	INTCMD0	–	–	Clear & start of timer C
		TMD1	Read	–	–	–	–
		CMD1	Read/write	INTCMD1	–	–	–
		TMD2	Read	–	–	–	–
		CMD2	Read/write	INTCMD2	–	–	–
		TMD3	Read	–	–	–	–
		CMD3	Read/write	INTCMD3	–	–	–

Remark fx: Main clock
S/R: Set/reset

(1) Timer D (16-bit timer/counter)

Figure 9-14. Timer D Block Diagram



9.2.4 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 CEDn bit of the timer mode control register Dn (TMCDn) (n = 0 to 3).

Division by the prescaler can be selected for the count clock from among $fx/8$, $fx/16$, $fx/32$, $fx/64$, $fx/128$, $fx/256$, $fx/512$, and $fx/1,024$ by the CSDn0 to CSDn2 bits of the TMCDn register (fx: main 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
- CAEDn bit = 0
- CEDn bit = 0
- Match of TMDn register and CMDn register
- Overflow

Cautions 1. If the CAEDn bit of the TMCDn register is cleared to 0, a reset is performed asynchronously.

2. If the CEDn bit of the TMCDn register is cleared to 0, a reset is performed, in synchronization 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 CEDn bit is cleared to 0.

4. Up to 8 clocks (1 cycle = $1/fx$ (fx: main clock)) are required after a value is set in the CEDn 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.

6. To initialize the TMDn register and start counting again, set the CEDn bit of the TMCDn register to 1 after 8 clocks (1 cycle = $1/fx$ (fx: main clock)).

(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, in synchronization with this match. If the CAEDn bit of the TMCn register is cleared 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 CMDn register is written, 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 CMDn register is read, 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 8 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 8 clocks (1 cycle = 1/fx (fx: main clock)).
 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 an INTCMDn interrupt 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 9-15).

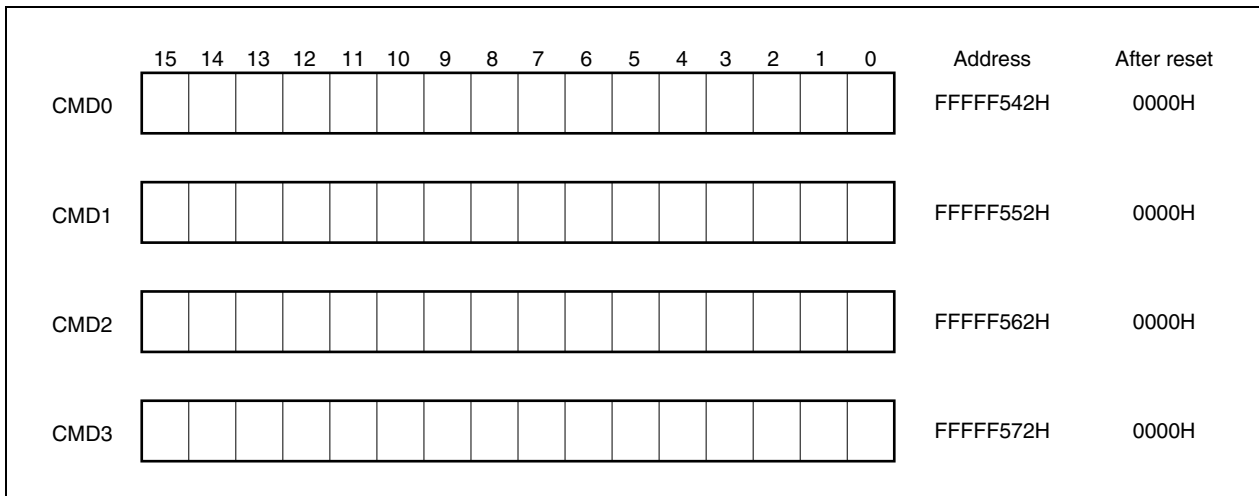
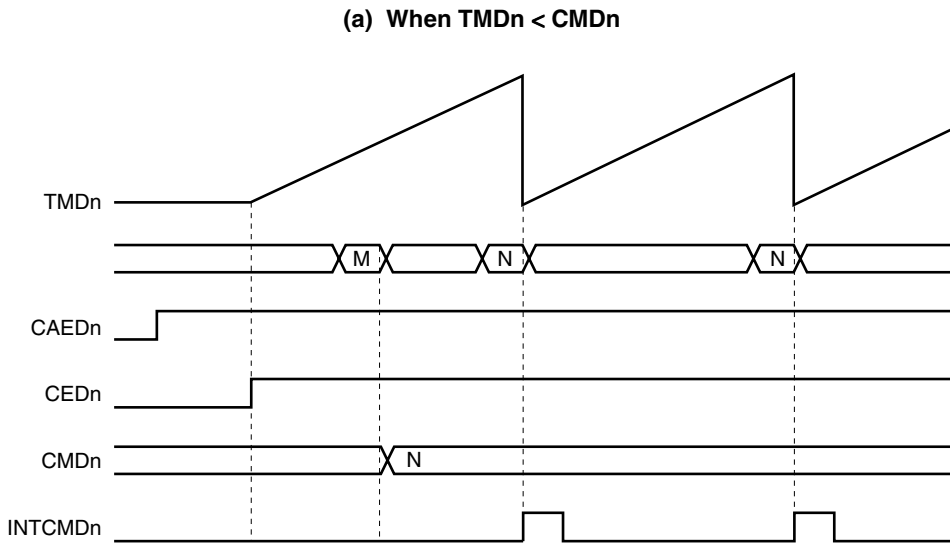
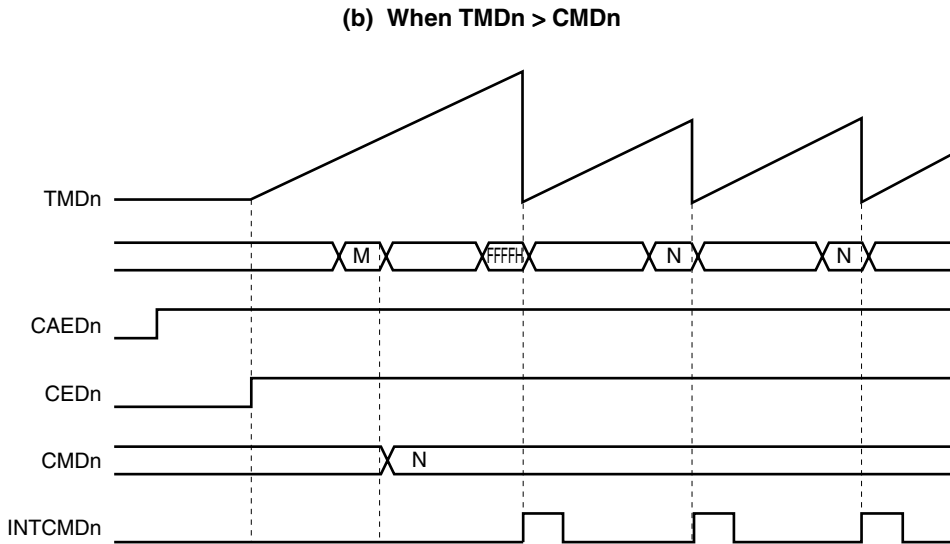


Figure 9-15. Example of Timing During TMDn Operation



Remark M = TMDn value when overwritten
 N = CMDn value when overwritten
 $M < N$



Remark M = TMDn value when overwritten
 N = CMDn value when overwritten
 $M > N$

9.2.5 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 CAEDn and other bits cannot be set at the same time. The other bits and the registers of the other TMDn units should always be set after the CAEDn bit has been set.

(1/2)

	7	6	5	4	3	2	<1>	<0>	Address	After reset
TMCD0	0	CSD02	CSD01	CSD00	0	0	CED0	CAED0	FFFFFF544H	00H
TMCD1	0	CSD12	CSD11	CSD10	0	0	CED1	CAED1	FFFFFF554H	00H
TMCD2	0	CSD22	CSD21	CSD20	0	0	CED2	CAED2	FFFFFF564H	00H
TMCD3	0	CSD32	CSD31	CSD30	0	0	CED3	CAED3	FFFFFF574H	00H

Bit position	Bit name	Function																																				
6 to 4	CSDn2 to CSDn0	<p>Selects the TMDn internal count clock cycle.</p> <table border="1"> <thead> <tr> <th>CSDn2</th> <th>CSDn1</th> <th>CSDn0</th> <th>Count cycle</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>fx/8</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>fx/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>fx/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>fx/64</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>fx/128</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>fx/256</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>fx/512</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>fx/1,024</td> </tr> </tbody> </table> <p>Caution The CSDn2 to CSDn0 bits must not be changed during timer operation. If they are to be changed, they must be changed after setting the CEDn bit to 0. If these bits are overwritten during timer operation, operation cannot be guaranteed.</p> <p>Remark fx: Main clock</p>	CSDn2	CSDn1	CSDn0	Count cycle	0	0	0	fx/8	0	0	1	fx/16	0	1	0	fx/32	0	1	1	fx/64	1	0	0	fx/128	1	0	1	fx/256	1	1	0	fx/512	1	1	1	fx/1,024
CSDn2	CSDn1	CSDn0	Count cycle																																			
0	0	0	fx/8																																			
0	0	1	fx/16																																			
0	1	0	fx/32																																			
0	1	1	fx/64																																			
1	0	0	fx/128																																			
1	0	1	fx/256																																			
1	1	0	fx/512																																			
1	1	1	fx/1,024																																			
1	CEDn	<p>Controls the operation of TMDn.</p> <p>0: Count disabled (stops at 0000H and does not operate)</p> <p>1: Counting operation is performed</p> <p>Caution The CEDn bit is not cleared even if a match is detected by the compare operation. To stop the count operation, clear the CEDn bit.</p>																																				

Remark n = 0 to 3

Bit position	Bit name	Function
0	CAEDn	<p>Controls the internal count clock.</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 CAEDn bit is cleared to 0, the TMDn unit can be asynchronously reset. 2. When CAEDn = 0, the TMDn unit is in a reset state. Therefore, to operate TMDn, the CAEDn bit must be set to 1. 3. If the CAEDn bit is cleared to 0, all the registers of the TMDn unit are initialized. If CAEDn is set to 1 again, be sure all the registers of the TMDn unit have been set again.

Remark n = 0 to 3

9.2.6 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 to 0 at the next count timing. This function enables timer D to be used as an interval timer.

CMDn can also be cleared 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 to 0 at the next count timing, INTCMDn is not generated by this match.

Figure 9-16. TMD0 Compare Operation Example (1/2)

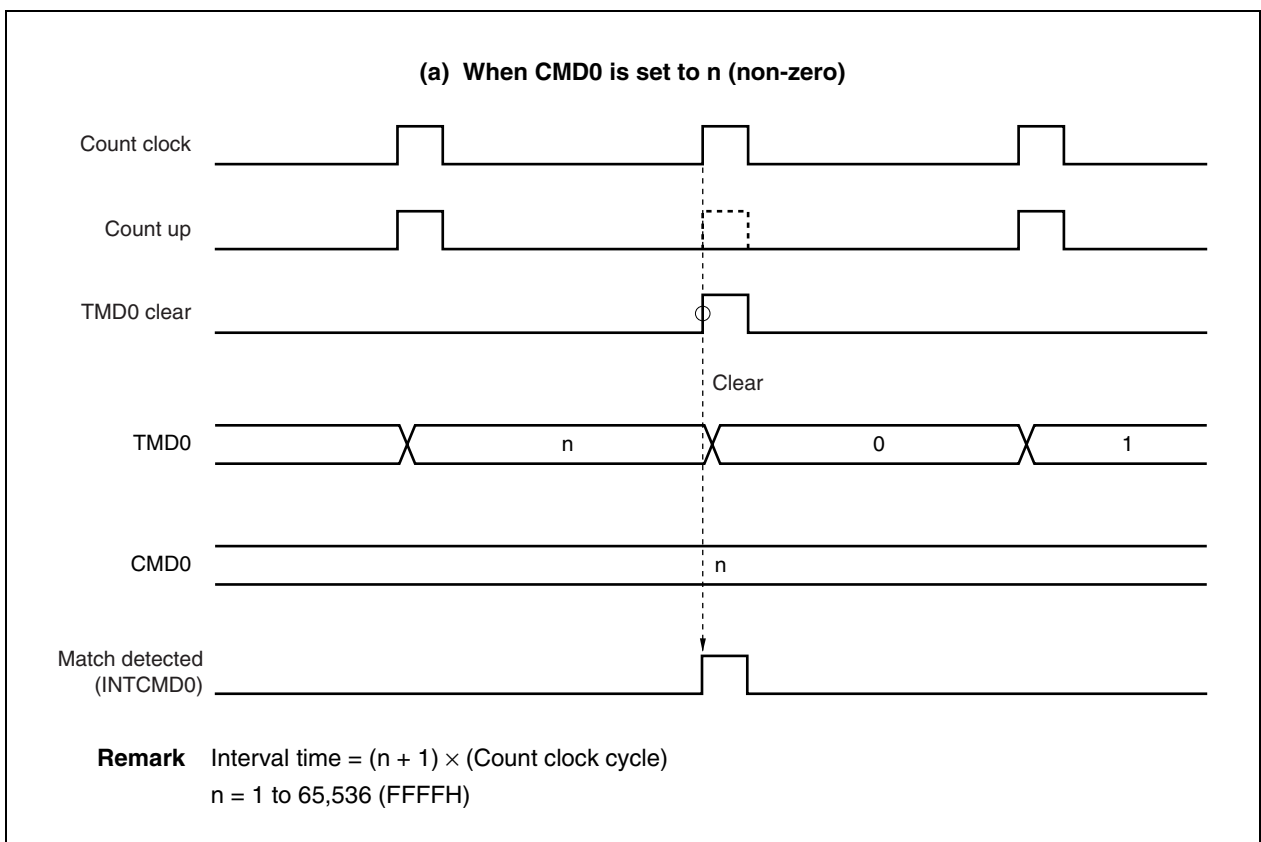
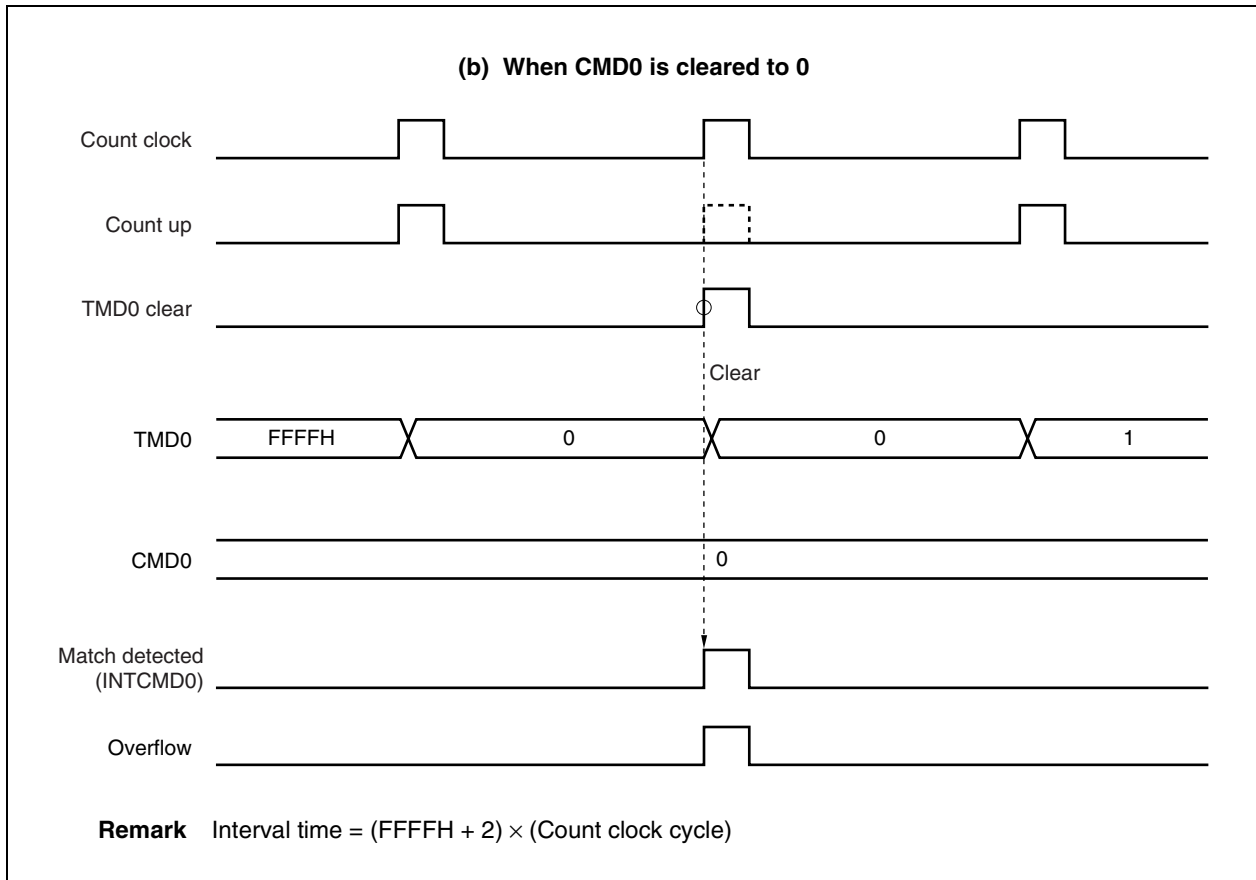


Figure 9-16. TMD0 Compare Operation Example (2/2)



9.2.7 Application examples

(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 (see **Figure 9-16 TMD0 Compare Operation Example**). The setup procedure is shown below (n = 0 to 3).

- <1> Set the CAEDn bit to 1.
- <2> Set each register.
 - Select the count clock using the CSDn0 to CSDn2 bits of the TMCDn register.
 - Set the compare value in the CMDn register.
- <3> Start counting by setting the CEDn bit to 1.
- <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

9.2.8 Cautions

Various cautions concerning timer D are shown below.

- (1) To operate TMDn, first set the CAEDn bit to 1.
- (2) Up to 8 clocks are required after a value is set in the CEDn 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 (1 cycle = 1/fx (fx: main clock)).
- (3) To initialize the TMDn register status and start counting again, clear the CEDn bit to 0 and then set the CEDn bit to 1 after an interval of 8 clocks has elapsed.
- (4) Up to 8 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 8 clocks (1 cycle = 1/fx (fx: main clock)).
- (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 CEDn bit is cleared to 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

9.3 Timer ENC1

9.3.1 Features

Timers ENC10 and ENC11 (TMENC10, TMENC11) are 16-bit up/down counters that perform the following operations.

- General-purpose timer mode
 - Free-running timer
 - PWM output
- Up/down counter mode
 - UDC mode A
 - UDC mode B

9.3.2 Function overview

- 16-bit 2-phase encoder input up/down counter & general-purpose timer (TMENC10, TMENC11): 2 channels
- Compare registers: 4
- Capture/compare registers: 4
- Interrupt request sources
 - Capture/compare match interrupt: 4 sources
 - Compare match interrupt request: 4 sources
- Capture request signal: 4 sources
 - The TMENC1n value can be latched using the valid edge of the INTP1n0 and INTP1n1 pins corresponding to the capture/compare register as the capture trigger.
- Count clock selectable through division by prescaler
- Timer/counter count clock sources: 2 types
(selection of external pulse input or main clock division)
- 2-phase encoder input

The 2-phase external encoder signal is used as the count clock of the timer/counter via the external clock input pins (TIUD1n, TCUD1n). The counter mode can be selected from among the four following modes.

 - Mode 1: Counts the input pulses of the count pulse input pin.
Up/down is specified by the level of the other input pin.
 - Mode 2: Counts up/down using the respective input pulses of the up count pulse input pin and down count pulse input pin.
 - Mode 3: Counts up/down using the phase relationship of the pulses input to the 2 pins.
 - Mode 4: Counts up/down using the phase relationship of the pulses input to the 2 pins. Counting is done using the respective rising edges and the falling edges of the pulses.
- PWM output function

In the general-purpose timer mode, 16-bit resolution PWM can be output from the TO1n pin.

- Timer clear

The following timer clear operations are performed according to the mode that is used.

- (a) General-purpose timer mode: Timer clear operation is possible upon occurrence of match with CM1n0 set value.
- (b) Up/down counter mode: The timer clear operation can be selected from among the following four conditions.
 - (i) Timer clear performed upon occurrence of match with CM1n0 set value during TMENC1n up count operation, and timer clear performed upon occurrence of match with CM1n1 set value during TMENC1n down count operation.
 - (ii) Timer clear performed only by external input.
 - (iii) Timer clear performed upon occurrence of match between TMENC1n count value and CM1n0 set value.
 - (iv) Timer clear performed upon occurrence of external input and match between TMENC1n count value and CM1n0 set value.

- External pulse output (TO1n): 2

9.3.3 Basic configuration

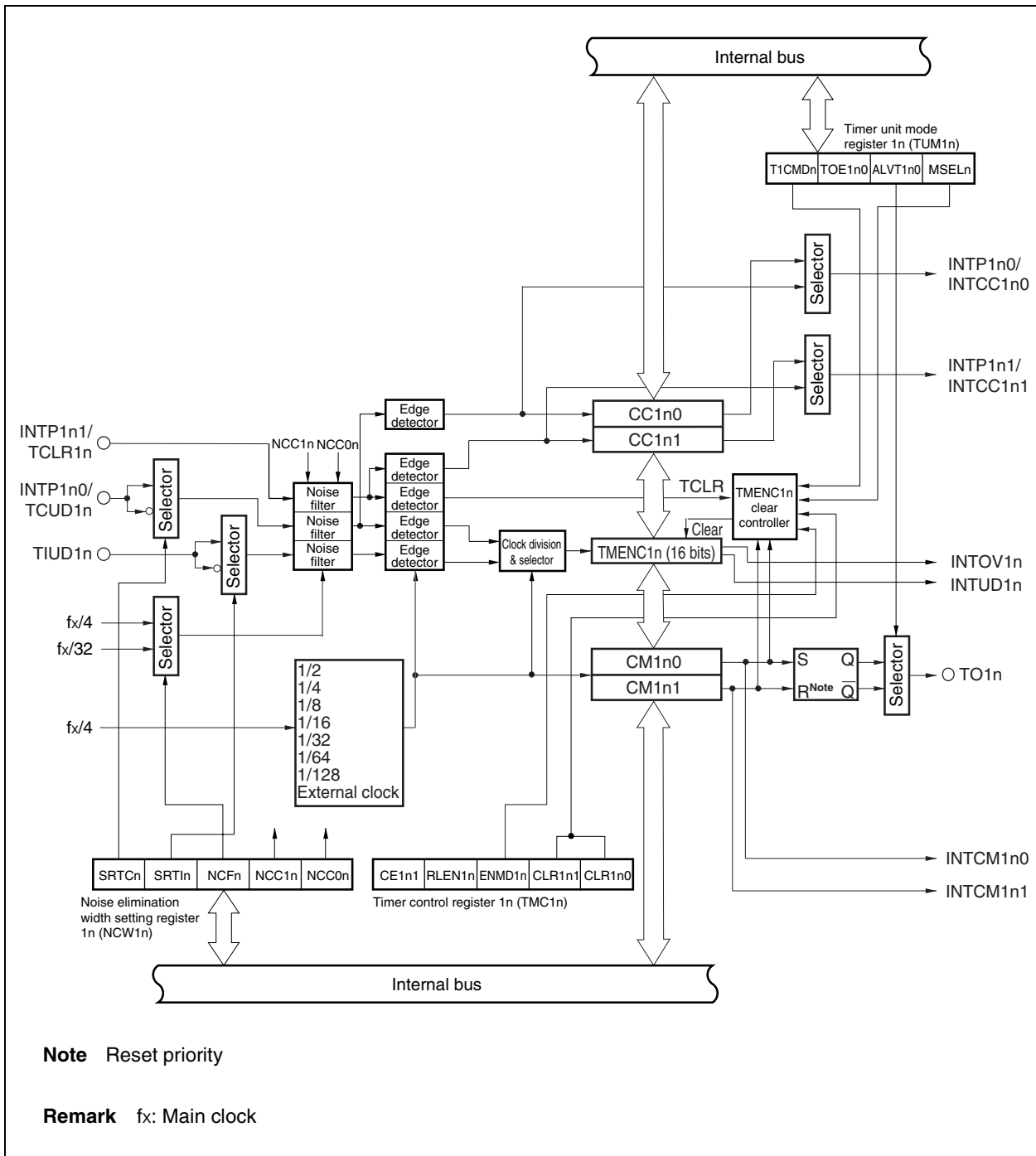
Table 9-4. Timer ENC1 Configuration

Timer	Count Clock	Register	Read/Write	Generated Interrupt Signal	Capture Trigger
Timer ENC1	fx/8, fx/16, fx/32, fx/64, fx/128, fx/256, fx/512	TMENC10	Read/write	–	–
		CM100	Read/write	INTCM100	–
		CM101	Read/write	INTCM101	–
		CC100	Read/write	INTCC100	INTP100
		CC101	Read/write	INTCC101	INTP101
		TMENC11	Read/write	–	–
		CM110	Read/write	INTCM110	–
		CM111	Read/write	INTCM111	–
		CC110	Read/write	INTCC110	INTP110
		CC111	Read/write	INTCC111	INTP111

Remark fx: Main clock

(1) Timer ENC1 (16-bit up/down counter)

Figure 9-17. Timer ENC1 Block Diagram



9.3.4 Timer ENC1

(1) Timers ENC10, ENC11 (TMENC10, TMENC11)

TMENC1n is a 2-phase encoder input up/down counter and general-purpose timer.
It can be read or written in 16-bit units.

- Cautions**
1. Writing to TMENC1n is enabled only when the CE1n1 bit of the TMC1n register is 0 (count operation disabled).
 2. It is prohibited to set the T1CMDn bit (general-purpose timer mode) and the MSELn bit (UDC mode B) of the TUM1n register to 0 and 1, respectively.
 3. Continuous reading of TMENC1n is prohibited. If TMENC1n is continuously read, the second read value may differ from the actual value. If TMENC1n must be read twice, be sure to read another register between the first and the second read operation.
 4. Writing the same value to the TMENC1n, CC1n0, and CC1n1 registers, and the STATUS1n register is prohibited. Writing the same value to the CCR1n, TUM1n, TMC1n, SESA1n, and PRM1n registers, and CM1n0 and CM1n1 registers is permitted (writing the same value is guaranteed even during a count operation).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
TMENC10																	FFFF5A0H	0000H
TMENC11																	FFFF5D0H	0000H

TMENC1n start and stop is controlled by the CE1n1 bit of timer control register 1n (TMC1n).
The TMENC1n operation consists of the following two modes.

(a) General-purpose timer mode

In the general-purpose timer mode, TMENC1n operates as a 16-bit interval timer, free-running timer, or PWM output.

Counting is performed based on the clock selected by software.

Division by the prescaler can be selected for the count clock from among $f_x/8$, $f_x/16$, $f_x/32$, $f_x/64$, $f_x/128$, $f_x/256$, or $f_x/512$ using the PRM1n2 to PRM1n0 bits of prescaler mode register 1n (PRM1n) (f_x : main clock).

(b) Up/down counter mode (UDC mode)

In the UDC mode, TMENC1n functions as a 16-bit up/down counter that performs counting based on the TCUD1n and TIUD1n input signals.

Two operation modes can be set by the MSELn bit of the TUM1n register for this mode.

(i) UDC mode A (when T1CMDn bit = 1, MSELn bit = 0)

TMENC1n can be cleared by setting the CLR1n1 and CLR1n0 bits of the TMC1n register.

(ii) UDC mode B (when T1CMDn bit = 1, MSELn bit = 1)

TMENC1n is cleared upon a match with CM1n0 during a TMENC1n up count operation.

TMENC1n is cleared upon a match with CM1n1 during a TMENC1n down count operation.

When the CE1n1 bit of the TMC1n register is 1, TMENC1n counts up when the operation mode is the general-purpose mode, and counts up/down when the operation mode is the UDC mode.

- Cautions 1.** The TCUD1n and INTP1n0 pins function alternately. Therefore, because the TCUD1n pin is used in the UDC mode, the external capture function of the INTP1n0 pin cannot be used.
- 2.** The TCLR1n and INTP1n1 pins function alternately. Therefore, when the TCLR1n input is used in UDC mode A, the external capture function of the INTP1n1 pin cannot be used.

The conditions for clearing TMENC1n are as follows, depending on the operation mode.

Table 9-5. Timer ENC1 (TMENC1n) Clear Conditions

Operation Mode	TUM1n Register		TMC1n Register			TMENC1n Clear
	T1CMDn Bit	MSELn Bit	ENMD1n Bit	CLR1n1 Bit	CLR1n0 Bit	
General-purpose timer mode	0	0	0	×	×	Clearing not performed
			1	×	×	Cleared upon match with CM1n0 set value
UDC mode A	1	0	×	0	0	Cleared only by TCLR1n input
			×	0	1	Cleared upon match with CM1n0 set value during up count operation
			×	1	0	Cleared by TCLR1n input or upon match with CM1n0 set value during up count operation
			×	1	1	Clearing not performed
UDC mode B	1	1	×	×	×	Cleared upon match with CM1n0 set value during up count operation or upon match with CM1n1 set value during down count operation
Other than the above						Setting prohibited

Remark ×: Indicates that the set value of that bit is ignored.

(2) Compare registers 100, 110 (CM100, CM110)

CM1n0 is a 16-bit register that always compares its value with the value of TMENC1n. When the value of a compare register matches the value of TMENC1n, an interrupt signal is generated. The interrupt generation timing in the various modes is described below.

- In the general-purpose timer mode (T1CMDn bit of TUM1n register = 0) and UDC mode A (MSELn bit of TUM1n register = 0), an interrupt signal (INTCM1n0) is always generated upon occurrence of a match.
- In UDC mode B (MSELn bit of TUM1n register = 1), an interrupt signal (INTCM1n0) is generated only upon occurrence of a match during an up count operation.

CM1n0 can be read or written in 16-bit units.

Caution When the CE1n1 bit of the TMC1n register is 1, it is prohibited to overwrite the value of the CM1n0 register.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CM100																	FFFFF5A2H	0000H
CM110																	FFFFF5D2H	0000H

(3) Compare registers 101, 111 (CM101, CM111)

CM1n1 is a 16-bit register that always compares its value with the value of TMENC1n. When the value of the compare register matches the value of TMENC1n, an interrupt signal is generated. The interrupt generation timing in the various modes is described below.

- In the general-purpose timer mode (T1CMDn bit of TUM1n register = 0) and UDC mode A (MSELn bit of TUM1n register = 0), an interrupt signal (INTCM1n1) is always generated upon occurrence of a match.
- In UDC mode B (MSELn bit of TUM1n register = 1), an interrupt signal (INTCM1n1) is generated only upon occurrence of a match during a down count operation.

CM1n1 can be read or written in 16-bit units.

Caution When the CE1n1 bit of the TMC1n register is 1, it is prohibited to overwrite the value of the CM1n1 register.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CM101																	FFFFF5A4H	0000H
CM111																	FFFFF5D4H	0000H

(4) Capture/compare registers 100, 110 (CC100, CC110)

CC1n0 is a 16-bit register. It can be specified as a capture register or as a compare register using capture/compare control register 1n (CCR1n). CC1n0 can be read or written in 16-bit units.

- Cautions**
1. When used as a capture register (CMSn0 bit of CCR1n register = 0), write access from the CPU is prohibited.
 2. When used as a compare register (CMSn0 bit of CCR1n register = 1) and the CE1n1 bit of the TMC1n register is 1, overwriting the CC1n0 register values is prohibited.
 3. When the CE1n1 bit of the TMC1n register is 0, the capture trigger is disabled.
 4. When the operation mode is changed from capture register to compare register, set a new compare value.
 5. Continuous reading of CC1n0 is prohibited. If CC1n0 is continuously read, the second read value may differ from the actual value. If CC1n0 must be read twice, be sure to read another register between the first and the second read operation.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CC100																	FFFFF5A6H	0000H
CC110																	FFFFF5D6H	0000H

(a) When set as a capture register

When CC1n0 is set as a capture register, the valid edge of the corresponding external interrupt INTP1n0 signal is detected as the capture trigger. TMENC1n latches the count value in synchronization with the capture trigger (capture operation). The latched value is held in the capture register until the next capture operation.

The valid edge of external interrupts (rising edge, falling edge, both edges) is selected by valid edge select register 1n (SESA1n).

When the CC1n0 register is specified as a capture register, interrupts are generated upon detection of the valid edge of the INTP1n0 signal.

Caution The TCUD1n and INTP1n0 pins function alternately. Therefore, because the TCUD1n pin is used in the UDC mode, the external capture function of the INTP1n0 pin cannot be used.

(b) When set as a compare register

When CC1n0 is set as a compare register, it always compares its own value with the value of TMENC1n. If the value of CC1n0 matches the value of the TMENC1n, CC1n0 generates an interrupt signal (INTCC1n0).

(5) Capture/compare registers 101, 111 (CC101, CC111)

CC1n1 is a 16-bit register. It can be specified as a capture register or as a compare register using capture/compare control register 1n (CCR1n). CC1n1 can be read or written in 16-bit units.

- Cautions**
1. When used as a capture register (CMSn1 bit of CCR1n register = 0), write access from the CPU is prohibited.
 2. When used as a compare register (CMSn1 bit of CCR1n register = 1) and the CE1n1 bit of the TMC1n register is 1, overwriting the CC1n1 register values is prohibited.
 3. When the CE1n1 bit of the TMC1n register is 0, the capture trigger is disabled.
 4. When the operation mode is changed from capture register to compare register, newly set a compare value.
 5. Continuous reading of CC1n1 is prohibited. If CC1n1 is continuously read, the second read value may differ from the actual value. If CC1n1 must be read twice, be sure to read another register between the first and the second read operation.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CC101																	FFFFF5A8H	0000H
CC111																	FFFFF5D8H	0000H

(a) When set as a capture register

When CC1n1 is set as a capture register, the valid edge of the corresponding external interrupt INTP1n1 signal is detected as the capture trigger. TMENC1n latches the count value in synchronization with the capture trigger (capture operation). The latched value is held in the capture register until the next capture operation.

The valid edge of external interrupts (rising edge, falling edge, both edges) is selected by valid edge select register 1n (SESA1n).

When the CC1n1 register is specified as a capture register, interrupts are generated upon detection of the valid edge of the INTP1n1 signal.

Caution The TCLR1n and INTP1n1 pins function alternately. Therefore, when the TCLR1n input is used in UDC mode A, the external capture function of the INTP1n1 pin cannot be used.

(b) When set as a compare register

When CC1n1 is set as a compare register, it always compares its own value with the value of TMENC1n. If the value of CC1n1 matches the value of the TMENC1n, CC1n1 generates an interrupt signal (INTCC1n1).

9.3.5 Control registers

(1) Timer unit mode registers 10, 11 (TUM10, TUM11)

The TUM1n register is an 8-bit register used to specify the TMENC1n operation mode or to control the operation of the PWM output pin.

TUM1n can be read or written in 8-bit or 1-bit units.

- Cautions**
1. Changing the value of the TUM1n register during TMENC1n operation (CE1n1 bit of TMC1n register = 1) is prohibited.
 2. When the T1CMDn bit = 0 (general-purpose timer mode), setting MSELn bit = 1 (UDC mode B) is prohibited.

	7	6	5	4	3	2	1	0	Address	After reset
TUM10	T1CMD0	0	0	0	TOE100	ALVT100	0	MSEL0	FFFFF5ABH	00H
TUM11	T1CMD1	0	0	0	TOE110	ALVT110	0	MSEL1	FFFFF5DBH	00H

Bit position	Bit name	Function
7	T1CMDn	Specifies TMENC1n operation mode. 0: General-purpose timer mode (up count) 1: UDC mode (up/down count)
3	TOE1n0	Specifies timer output (TO1n) enable. 0: Timer output disabled 1: Timer output enabled Caution When T1CMDn bit = 1 (UDC mode), timer output is not performed regardless of the setting of the TOE1n0 bit. At this time, timer output consists of the negative phase level of the level set by the ALVT1n0 bit.
2	ALVT1n0	Specifies active level of timer output (TO1n). 0: Active level is high level 1: Active level is low level Caution When T1CMDn bit = 1 (UDC mode), timer output is not performed regardless of the setting of the TOE1n0 bit. At this time, timer output consists of the negative phase level of the level set by the ALVT1n0 bit.
0	MSELn	Specifies operation in UDC mode (up/down count) 0: UDC mode A TMENC1n can be cleared by setting the CLR1n1 and CLR1n0 bits of the TMC1n register. 1: UDC mode B TMENC1n is cleared in the following cases. • Upon match with CM1n0 during TMENC1n up count operation • Upon match with CM1n1 during TMENC1n down count operation When UDC mode B is set, the ENMD1n, CLR1n1, and CLR1n0 bits of the TMC1n register become invalid.

Remark n = 0, 1

(2) Timer control registers 10, 11 (TMC10, TMC11)

The TMC1n register is used to enable/disable TMENC1n operation and to set transfer and timer clear operations.

TMC1n can be read or written in 8-bit or 1-bit units.

Caution Changing the values of the TMC1n register bits other than the CE1n1 bit during TMENC1n operation (CE1n1 bit = 1) is prohibited.

(1/2)

	7	<6>	5	4	3	2	1	0	Address	After reset
TMC10	0	CE101	0	0	RLEN10	ENMD10	CLR101	CLR100	FFFFFF5ACH	00H
TMC11	0	CE111	0	0	RLEN11	ENMD11	CLR111	CLR110	FFFFFF5DCH	00H

Bit position	Bit name	Function
6	CE1n1	Enables/disables TMENC1n operation. 0: TMENC1n count operation disabled 1: TMENC1n count operation enabled
3	RLEN1n	Enables/disables transfer from CM1n0 to TMENC1n. 0: Transfer disabled 1: Transfer enabled Cautions 1. When RLEN1n = 1, the value set to CM1n0 is transferred to TMENC1n upon occurrence of a TMENC1n underflow. 2. When the T1CMDn bit of the TUM1n register = 0 (general-purpose timer mode), the RLEN1n bit setting becomes invalid. 3. The RLEN1n bit is valid only in UDC mode A (TUM1n register's T1CMDn bit = 1, MSELn bit = 0). In the general-purpose timer mode (T1CMDn bit = 0) and in UDC mode B (T1CMDn bit = 1, MSELn bit = 1), a transfer operation is not performed even the RLEN1n bit is set to 1.
2	ENMD1n	Enables/disables clearing of TMENC1n in general-purpose timer mode (T1CMDn bit of TUM1n register = 0). 0: Clear disabled (free-running mode) Clearing is not performed even when TMENC1n and CM1n0 values match. 1: Clear enabled Clearing is performed when TMENC1n and CM1n0 values match. Caution When the T1CMDn bit of the TUM1n register = 1 (UDC mode), the ENMD1n bit setting becomes invalid.

Remark n = 0, 1

Bit position	Bit name	Function															
1, 0	CLR1n1, CLR1n0	Controls TMENC1n clear operation in UDC mode A. <table border="1" data-bbox="597 317 1430 604"> <thead> <tr> <th>CLR1n1</th> <th>CLR1n0</th> <th>Specifies TMENC1n clear source</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Cleared only by external input (TCLR1n)</td> </tr> <tr> <td>0</td> <td>1</td> <td>Cleared upon match of TMENC1n count value and CM1n0 set value</td> </tr> <tr> <td>1</td> <td>0</td> <td>Cleared by TCLR1n input or upon match of TMENC1n count value and CM1n0 set value</td> </tr> <tr> <td>1</td> <td>1</td> <td>Not cleared</td> </tr> </tbody> </table> <p>Cautions</p> <ol style="list-style-type: none"> 1. Clearing by match of the TMENC1n count value and CM1n0 set value is valid only during a TMENC1n up count operation (TMENC1n is not cleared during a TMENC1n down count operation). 2. When the T1CMDn bit of the TUM1n register = 0 (general-purpose timer mode), the CLR1n1 and CLR1n0 bit settings are invalid. 3. When the MSELn bit of the TUM1n register = 1 (UDC mode B), the CLR1n1 and CLR1n0 bit settings are invalid. 4. When clearing by TCLR1n has been enabled by bits CLR1n1 and CLR1n0, clearing is performed whether the value of the CE1n1 bit is 1 or 0. 	CLR1n1	CLR1n0	Specifies TMENC1n clear source	0	0	Cleared only by external input (TCLR1n)	0	1	Cleared upon match of TMENC1n count value and CM1n0 set value	1	0	Cleared by TCLR1n input or upon match of TMENC1n count value and CM1n0 set value	1	1	Not cleared
CLR1n1	CLR1n0	Specifies TMENC1n clear source															
0	0	Cleared only by external input (TCLR1n)															
0	1	Cleared upon match of TMENC1n count value and CM1n0 set value															
1	0	Cleared by TCLR1n input or upon match of TMENC1n count value and CM1n0 set value															
1	1	Not cleared															
Remark n = 0, 1																	

(3) Capture/compare control registers 10, 11 (CCR10, CCR11)

The CCR1n register specifies the operation mode of the capture/compare registers (CC1n0, CC1n1). CCR1n can be read or written in 8-bit or 1-bit units.

- Cautions**
1. **Overwriting the CCR1n register during TMENC1n operation (CE1n1 bit = 1) is prohibited.**
 2. **The TCUD1n and INTP1n0 pins function alternately. Therefore, because the TCUD1n pin is used in the UDC mode, the external capture function of the INTP1n0 pin cannot be used.**
 3. **The TCLR1n and INTP1n1 pins function alternately. Therefore, when the TCLR1n input is used in UDC mode A, the external capture function of the INTP1n1 pin cannot be used.**

	7	6	5	4	3	2	1	0	Address	After reset
CCR10	0	0	0	0	0	0	CMS01	CMS00	FFFFFF5AAH	00H
CCR11	0	0	0	0	0	0	CMS11	CMS10	FFFFFF5DAH	00H

Bit position	Bit name	Function
1	CMSn1	Specifies operation mode of CC1n1. 0: Capture register 1: Compare register
0	CMSn0	Specifies operation mode of CC1n0. 0: Capture register 1: Compare register

Remark n = 0, 1

(4) Valid edge select registers 10, 11 (SESA10, SESA11)

The SESA1n register is used to specify the valid edge of external interrupt requests from external pins (INTP100, INTP101, INTP110, INTP111, TIUD10, TIUD11, TCUD10, TCUD11, TCLR10, TCLR11).

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

SESA1n can be read or written in 8-bit or 1-bit units.

- Cautions 1.** Changing the values of the SESA1n register bits during TMENC1n operation (CE1n1 bit = 1) is prohibited.
- 2.** Before setting the trigger mode of the INTP100, INTP101, INTP110, INTP111, TIUD10, TIUD11, TCUD10, TCUD11, TCLR10, and TCLR11 pins, set the PMCDH register. If the PMCDH register is set after the SESA1n register has been set, an illegal interrupt, incorrect counting, and incorrect clearing may occur depending on the timing of setting the PMCDH register.

(1/2)

SESA10	7 6 5 4 3 2 1 0	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 12.5%; text-align: center;">TESUD01</td> <td style="width: 12.5%; text-align: center;">TESUD00</td> <td style="width: 12.5%; text-align: center;">CESUD01</td> <td style="width: 12.5%; text-align: center;">CESUD00</td> <td style="width: 12.5%; text-align: center;">IES101</td> <td style="width: 12.5%; text-align: center;">IES100</td> <td style="width: 12.5%; text-align: center;">IES001</td> <td style="width: 12.5%; text-align: center;">IES000</td> </tr> <tr> <td colspan="2" style="text-align: center;">TIUD10, TCUD10</td> <td colspan="2" style="text-align: center;">TCLR10</td> <td colspan="2" style="text-align: center;">INTP101</td> <td colspan="2" style="text-align: center;">INTP100</td> </tr> </table>	TESUD01	TESUD00	CESUD01	CESUD00	IES101	IES100	IES001	IES000	TIUD10, TCUD10		TCLR10		INTP101		INTP100		Address	After reset
TESUD01	TESUD00	CESUD01	CESUD00	IES101	IES100	IES001	IES000													
TIUD10, TCUD10		TCLR10		INTP101		INTP100														
								FFFFFF5ADH	00H											
SESA11	7 6 5 4 3 2 1 0	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 12.5%; text-align: center;">TESUD11</td> <td style="width: 12.5%; text-align: center;">TESUD10</td> <td style="width: 12.5%; text-align: center;">CESUD11</td> <td style="width: 12.5%; text-align: center;">CESUD10</td> <td style="width: 12.5%; text-align: center;">IES111</td> <td style="width: 12.5%; text-align: center;">IES110</td> <td style="width: 12.5%; text-align: center;">IES011</td> <td style="width: 12.5%; text-align: center;">IES010</td> </tr> <tr> <td colspan="2" style="text-align: center;">TIUD11, TCUD11</td> <td colspan="2" style="text-align: center;">TCLR11</td> <td colspan="2" style="text-align: center;">INTP111</td> <td colspan="2" style="text-align: center;">INTP110</td> </tr> </table>	TESUD11	TESUD10	CESUD11	CESUD10	IES111	IES110	IES011	IES010	TIUD11, TCUD11		TCLR11		INTP111		INTP110		Address	After reset
TESUD11	TESUD10	CESUD11	CESUD10	IES111	IES110	IES011	IES010													
TIUD11, TCUD11		TCLR11		INTP111		INTP110														
								FFFFFF5DDH	00H											

Bit position	Bit name	Function															
7, 6	TESUDn1, TESUDn0	Specifies valid edge of pins TIUD1n, TCUD1n. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 12.5%;">TESUDn1</th> <th style="width: 12.5%;">TESUDn0</th> <th style="width: 75%;">Valid edge</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Falling edge</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Rising edge</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Setting prohibited</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table> <p>Cautions 1. The set values of the TESUDn1 and TESUDn0 bits are only valid in UDC mode A and UDC mode B.</p> <p>2. If mode 4 is specified as the operation mode of TMENC1n (specified by the PRM1n2 to PRM1n0 bits of the PRM1n register), the valid edge specifications for the TIUD1n and TCUD1n pins (bits TESUDn1 and TESUDn0) are not valid.</p>	TESUDn1	TESUDn0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
TESUDn1	TESUDn0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															

Remark n = 0, 1

Bit position	Bit name	Function															
5, 4	CESUDn1, CESUDn0	<p>Specifies valid edge of TCLR1n pin.</p> <table border="1"> <thead> <tr> <th>CESUDn1</th> <th>CESUDn0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Low level</td> </tr> <tr> <td>1</td> <td>1</td> <td>High level</td> </tr> </tbody> </table> <p>The set values of bits CESUDn1 and CESUDn0 and the TMENC1n operation are related as follows.</p> <p>00: TMENC1n cleared after detection of rising edge of TCLR1n 01: TMENC1n cleared after detection of falling edge of TCLR1n 10: TMENC1n cleared status held while TCLR1n input is low level 11: TMENC1n cleared status held while TCLR1n input is high level</p> <p>Caution The set values of the CESUDn1 and CESUDn0 bits are valid only in UDC mode A.</p>	CESUDn1	CESUDn0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Low level	1	1	High level
CESUDn1	CESUDn0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Low level															
1	1	High level															
3, 2	IES1n1, IES1n0	<p>Specifies valid edge of INTP1n1 pin.</p> <table border="1"> <thead> <tr> <th>IES1n1</th> <th>IES1n0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	IES1n1	IES1n0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
IES1n1	IES1n0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															
1, 0	IES0n1, IES0n0	<p>Specifies valid edge of INTP1n0 pin.</p> <table border="1"> <thead> <tr> <th>IES0n1</th> <th>IES0n0</th> <th>Valid edge</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Falling edge</td> </tr> <tr> <td>0</td> <td>1</td> <td>Rising edge</td> </tr> <tr> <td>1</td> <td>0</td> <td>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	IES0n1	IES0n0	Valid edge	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
IES0n1	IES0n0	Valid edge															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															

Remark n = 0, 1

(5) Prescaler mode registers 10, 11 (PRM10, PRM11)

The PRM1n register is used to perform the following selections.

- Selection of count clock in general-purpose timer mode (T1CMDn bit of TUM1n register = 0)
- Selection of count operation mode in UDC mode (T1CMDn bit = 1)

PRM1n can be read or written in 8-bit or 1-bit units.

- Cautions**
1. **Overwriting the PRM1n register during TMENC1n operation (CE1n1 bit = 1) is prohibited.**
 2. **When the T1CMDn bit of the TUM1n register = 1 (UDC mode), setting the values of the PRM1n2 to PRM1n0 to 000, 001, 010, and 011 bits is prohibited.**
 3. **When TMENC1n is in mode 4, specification of the valid edge for the TIUD1n and TCUD1n pins is invalid.**

	7	6	5	4	3	2	1	0		
PRM10	0	0	0	0	0	PRM102	PRM101	PRM100	FFFFF5AEH	07H
PRM11	0	0	0	0	0	PRM112	PRM111	PRM110	FFFFF5DEH	07H

Bit position	Bit name	Function			
2 to 0	PRM1n2 to PRM1n0	Specifies the up/down count operation mode during input of the clock rate when the internal clock of the TMENC1n is used, or during external clock (TIUD1n) input.			
		PRM1n2	PRM1n1	PRM1n0	T1CMDn = 0
					T1CMDn = 1
					Count clock
					Count clock
					Up/down count
		0	0	0	Setting prohibited
		0	0	1	fx/8
		0	1	0	fx/16
		0	1	1	fx/32
1	0	0	fx/64		
1	0	1	fx/128		
1	1	0	fx/256		
1	1	1	fx/512		
				TIUD1n	
				Mode 1	
				Mode 2	
				Mode 3	
				Mode 4	
Remark fx: Main clock					

Remark n = 0, 1

(a) In general-purpose timer mode (T1CMDn bit of TUM1n register = 0)

The count clock is fixed to the internal clock. The clock rate of TMENC1n is specified by bits PRM1n2 to PRM1n0.

(b) UDC mode (T1CMDn bit of TUM1n register = 1)

The TMENC1n count triggers in the UDC mode are as follows.

Operation Mode	TMENC1n Operation
Mode 1	Down count when TCUD1n = high level Up count when TCUD1n = low level
Mode 2	Up count upon detection of valid edge of TIUD1n input Down count upon detection of valid edge of TCUD1n input
Mode 3	Automatic judgment with TCUD1n input level upon detection of valid edge of TIUD1n input
Mode 4	Automatic judgment upon detection of both edges of TIUD1n input and both edges of TCUD1n input

(6) Status registers 10, 11 (STATUS10, STATUS11)

The STATUS1n register indicates the operating status of TMENC1n.
STATUS1n is read-only in 8-bit or 1-bit units.

Caution Overwriting the STATUS1n register during TMENC1n operation (CE1n1 bit = 1) is prohibited.

	7	6	5	4	3	<2>	<1>	<0>	Address	After reset
STATUS10	0	0	0	0	0	UDF10	OVF10	UBD10	FFFFFF5AFH	00H
STATUS11	0	0	0	0	0	UDF11	OVF11	UBD11	FFFFFF5DFH	00H

Bit position	Bit name	Function
2	UDF1n	TMENC1n underflow flag 0: No TMENC1n count underflow 1: TMENC1n count underflow Caution The UDF1n bit is cleared to 0 upon completion of a read access to the STATUS1n register from the CPU.
1	OVF1n	TMENC1n overflow flag 0: No TMENC1n count overflow 1: TMENC1n count overflow Caution The OVF1n bit is cleared to 0 upon completion of a read access to the STATUS1n register from the CPU.
0	UBD1n	Indicates the operating status of TMENC1n up/down count. 0: TMENC1n up count in progress 1: TMENC1n down count in progress Caution The state of the UBD1n bit differs according to the mode as follows. <ul style="list-style-type: none"> • The UBD1n bit is fixed to 0 by hardware when the T1CMDn bit of the TUM1n register = 0 (general-purpose timer mode). • The UBD1n bit indicates the TMENC1n up/down count status when the T1CMDn bit of the TUM1n register = 1 (UDC mode).

Remark n = 0, 1

(7) Noise elimination width setting registers 10, 11 (NCW10, NCW11)

The NCW1n register is used to set the noise elimination width of the digital noise filter of the timer ENC1 input pin.

NCW1n can be read or written in 8-bit units.

Cautions 1. Whether the signal is input through or inverted can be specified for each of the INTP1n0/TCUD1n and TIUD1n pins. The setting of the noise elimination width by the NCFn, NCC1n, and NCC0n bits is for each timer and cannot be changed for each pin.

2. The setting of the SRTCn bit is valid even when the INTP1n0/TCUD1n pin is used as a capture trigger (INTP1n0).

	7	6	5	4	3	2	1	0	Address	After reset
NCW10	0	0	SRTC0	SRTI0	0	NCF0	NCC10	NCC00	FFFFFF5C0H	02H
NCW11	0	0	SRTC1	SRTI1	0	NCF1	NCC11	NCC01	FFFFFF5F0H	02H

Bit position	Bit name	Function															
5	SRTCn	Sets the input mode of the INTP1n0/TCUD1n pin. 0: Through input 1: Inverted This bit specifies whether the signal input from the INTP1n0/TCUD1n pin is input through to TMENC1n or inverted.															
4	SRTIn	Sets the input mode of the TIUD1n pin. 0: Through input 1: Inverted This bit specifies whether the signal input from the TIUD1n pin is input through to TMENC1n or inverted.															
2	NCFn	Specifies the clock frequency for noise elimination. 0: $f_x/4$ 1: $f_x/32$ This bit selects the clock source of the noise filter.															
1, 0	NCC1n, NCC0n	Specify the number of clocks by which noise is to be eliminated. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>NCC1n</th> <th>NCC0n</th> <th>Number of clocks by which noise is to be eliminated</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0 (through input)^{Note}</td> </tr> <tr> <td>0</td> <td>1</td> <td>2</td> </tr> <tr> <td>1</td> <td>0</td> <td>3</td> </tr> <tr> <td>1</td> <td>1</td> <td>5</td> </tr> </tbody> </table> <p>Note Clear the NCFn bit to 0 for through input.</p> <p>These bits select the number of clocks by which noise is to be eliminated.</p>	NCC1n	NCC0n	Number of clocks by which noise is to be eliminated	0	0	0 (through input) ^{Note}	0	1	2	1	0	3	1	1	5
NCC1n	NCC0n	Number of clocks by which noise is to be eliminated															
0	0	0 (through input) ^{Note}															
0	1	2															
1	0	3															
1	1	5															

Remark n = 0, 1

(a) Relationship between NCW1n register set value and noise elimination width

★

Table 9-6. Relationship Between NCW1n Register Set Value and Noise Elimination Width

NCW1n Register			Noise Elimination Width (ns)			Remark
NCFn Bit	NCC1n Bit	NCC0n Bit	f _x = 150 MHz	f _x = 133 MHz	f _x = 100 MHz	
0	0	0	0	0	0	Through
0	0	1	53.3 to 80.0	60.2 to 90.2	80 to 120	$(1/(f_x/4)) \times 2$
0	1	0	80.0 to 106.7	90.2 to 120.3	120 to 160	$(1/(f_x/4)) \times 3$
0	1	1	133.3 to 160.0	150.4 to 180.5	200 to 240	$(1/(f_x/4)) \times 5$
1	0	0	0	0	0	Through
1	0	1	426.7 to 640.0	481.2 to 721.8	640 to 960	$(1/(f_x/32)) \times 2$
1	1	0	640.0 to 853.3	721.8 to 962.9	960 to 1,280	$(1/(f_x/32)) \times 3$
1	1	1	1,066.7 to 1,280.0	1,203.0 to 1,443.6	1,600 to 1,920	$(1/(f_x/32)) \times 5$

Remarks 1. n = 0, 1

2. f_x: Main clock

9.3.6 Operation

(1) Basic operation

The following two operation modes can be selected for TMENC1n.

(a) General-purpose timer mode (T1CMDn bit of TUM1n register = 0)

In the general-purpose timer mode, TMENC1n operates either as a 16-bit interval timer or as a PWM output timer (count operation is up count only).

The count clock to TMENC1n is selected by prescaler mode register 1n (PRM1n) (n = 0, 1).

(b) Up/down counter mode (UDC mode) (T1CMDn bit of TUM1n register = 1)

In the UDC mode, TMENC1n operates as a 16-bit up/down counter.

The external clock input (TIUD1n, TCUD1n pins) by PRM1n register setting is used as the TMENC1n count clock.

The UDC mode is further divided into two modes according to the TMENC1n clear conditions.

- **UDC mode A (TUM1n register's T1CMDn bit = 1, MSELn bit = 0)**

The TMENC1n clear source can be selected as only external clear input (TCLR1n), a match signal between the TMENC1n count value and the CM1n0 set value during up count operation, or the logical sum (OR) of the two signals, using bits CLR1n1 and CLR1n0 of the TMC1n register. TMENC1n can transfer the value of CM1n0 upon occurrence of a TMENC1n underflow.

- **UDC mode B (TUM1n register's T1CMDn bit = 1, MSELn bit = 1)**

The status of TMENC1n after a match of the TMENC1n count value and CM1n0 set value is as follows.

<1> In the case of an up count operation, TMENC1n is cleared to 0000H, and the INTCM1n0 interrupt is generated.

<2> In the case of a down count operation, the TMENC1n count value is decremented (-1).

The status of TMENC1n after a match of the TMENC1n count value and CM1n1 set value is as follows.

<1> In the case of an up count operation, the TMENC1n count value is incremented (+1).

<2> In the case of a down count operation, TMENC1n is cleared to 0000H, and the INTCM1n1 interrupt is generated.

(2) Operation in general-purpose timer mode

TMENC1n can perform the following operations in the general-purpose timer mode.

(a) Interval operation

TMENC1n and CM1n0 always compare their values and the INTCM1n0 interrupt is generated upon occurrence of a match.

TMENC1n is cleared to 0000H at the count clock following the match.

Furthermore, when one more count clock is input, TMENC1n counts up to 0001H. The interval time can be calculated with the following formula.

$$\text{Interval time} = (\text{CM1n0 value} + 1) \times \text{TMENC1n count clock rate}$$

Caution Interval operation can be achieved by setting the ENMD1n bit of the TMC1n register to 1.

(b) Free-running operation

TMENC1n performs a full count operation from 0000H to FFFFH, and after the OVF1n bit of the STATUS1n register is set to 1, TMENC1n is cleared and resumes counting. The free-running cycle can be calculated by the following formula.

$$\text{Free-running cycle} = 65,536 \times \text{TMENC1n count clock rate}$$

Caution The free-running operation can be achieved by setting the ENMD1n bit of the TMC1n register to 0.

(c) Compare function

TMENC1n connects two compare register (CM1n0, CM1n1) channels and two capture/compare register (CC1n0, CC1n1) channels.

When the TMENC1n count value and the set value of one of the compare registers match, a match interrupt (INTCM1n0, INTCM1n1, INTCC1n0^{Note}, INTCC1n1^{Note}) is output.

Particularly in the case of interval operation, TMENC1n is cleared upon generation of the INTCM1n0 interrupt.

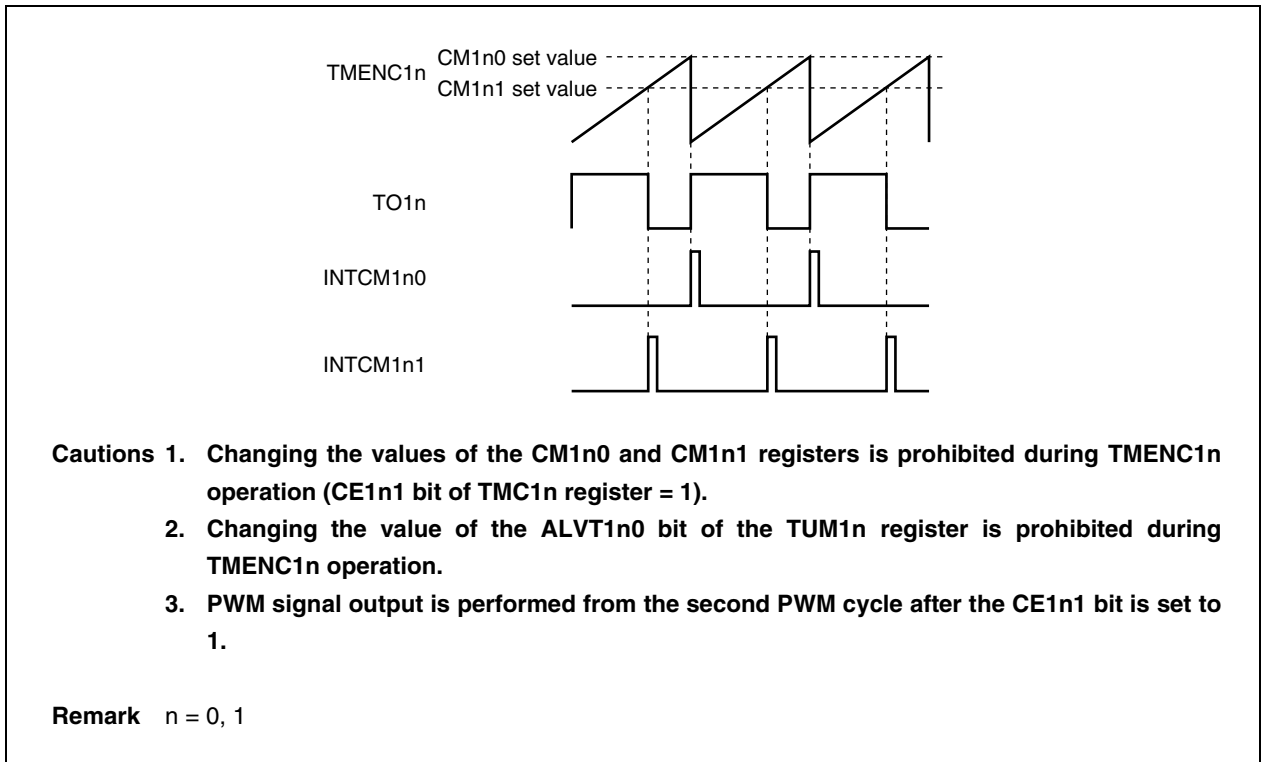
Note This match interrupt is generated when CC1n0 and CC1n1 are set to the compare register mode.

(i) Description of operation

The CM1n0 register is a compare register used to set the PWM output cycle. When the value of this register matches the value of TMENC1n, the INTCM1n0 interrupt is generated. The compare match is saved by hardware, and TMENC1n is cleared at the next count clock after the match.

The CM1n1 register is a compare register used to set the PWM output duty. Set the duty required for the PWM cycle.

Figure 9-19. PWM Signal Output Example (When ALVT1n0 Bit = 0 Is Set)



(3) Operation in UDC mode**(a) Overview of operation in UDC mode**

The count clock input to TMENC1n in the UDC mode (T1CMDn bit of TUM1n register = 1) can only be externally input from the TIUD1n and TCUD1n pins. Up/down count judgment in the UDC mode is determined based on the phase difference of the TIUD1n and TCUD1n pin inputs according to the PRM1n register setting (there is a total of four choices).

Table 9-8. List of Count Operations in UDC Mode

PRM1n Register			Operation Mode	TMENC1n Operation
PRM1n2	PRM1n1	PRM1n0		
1	0	0	Mode 1	Down count when TCUD1n = high level Up count when TCUD1n = low level
1	0	1	Mode 2	Up count upon detection of valid edge of TIUD1n input Down count upon detection of valid edge of TCUD1n input
1	1	0	Mode 3	Automatic judgment in TCUD1n input level upon detection of valid edge of TIUD1n input
1	1	1	Mode 4	Automatic judgment upon detection of both edges of TIUD1n input and both edges of TCUD1n input

Remark n = 0, 1

The UDC mode is further divided into two modes according to the TMENC1n clear conditions (a count operation is performed only with TIUD1n and TCUD1n input in both modes).

- **UDC mode A (TUM1n register's T1CMDn bit = 1, MSELn bit = 0)**

The TMENC1n clear source can be selected as only external clear input (TCLR1n), a match signal between the TMENC1n count value and the CM1n0 set value during up count operation, or the logical sum (OR) of the two signals, using bits CLR1n1 and CLR1n0 of the TMC1n register. TMENC1n can transfer the value of CM1n0 upon occurrence of a TMENC1n underflow.

- **UDC mode B (TUM1n register's T1CMDn bit = 1, MSELn bit = 1)**

The status of TMENC1n after a match of the TMENC1n count value and CM1n0 set value is as follows.

<1> In the case of an up count operation, TMENC1n is cleared to 0000H, and the INTCM1n0 interrupt is generated.

<2> In the case of a down count operation, the TMENC1n count value is decremented (-1).

The status of TMENC1n after a match of the TMENC1n count value and CM1n1 set value is as follows.

<1> In the case of an up count operation, the TMENC1n count value is incremented (+1).

<2> In the case of a down count operation, TMENC1n is cleared to 0000H, and the INTCM1n1 interrupt is generated.

(b) Up/down count operation in UDC mode

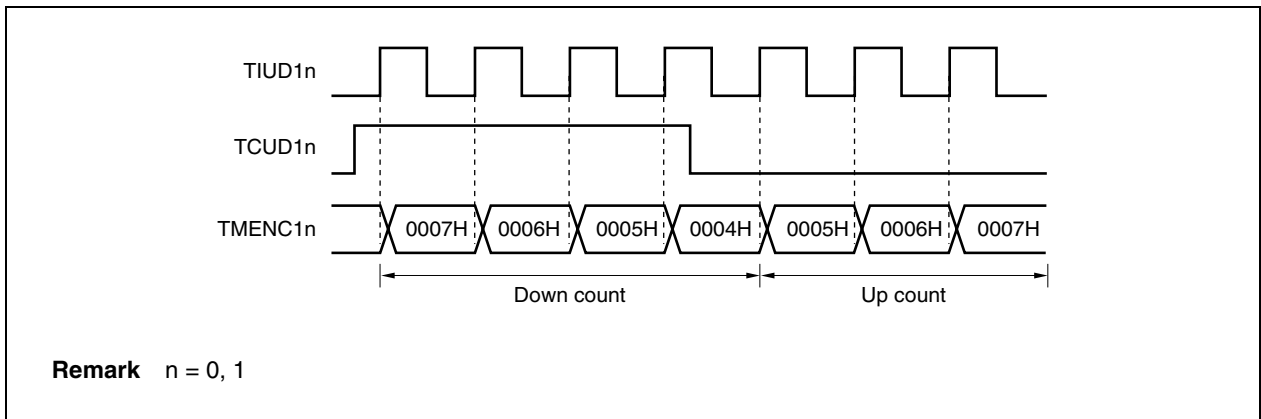
TMENC1n up/down count judgment in the UDC mode is determined based on the phase difference of the TIUD1n and TCUD1n pin inputs according to the PRM1n register setting.

(i) Mode 1 (PRM1n2 bit = 1, PRM1n1 bit = 0, PRM1n0 bit = 0)

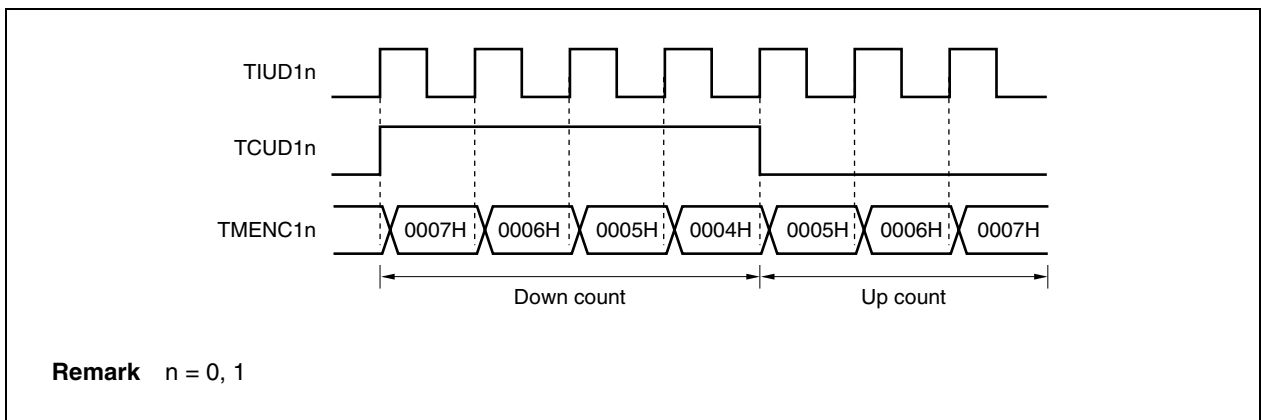
In mode 1, the following count operations are performed based on the level of the TCUD1n pin upon detection of the valid edge of the TIUD1n pin.

- TMENC1n down count operation when TCUD1n pin = high level
- TMENC1n up count operation when TCUD1n pin = low level

Figure 9-20. Mode 1 (When Rising Edge Is Specified as Valid Edge of TIUD1n Pin)



**Figure 9-21. Mode 1 (When Rising Edge Is Specified as Valid Edge of TIUD1n Pin):
In Case of Simultaneous TIUD1n, TCUD1n Pin Edge Timing**



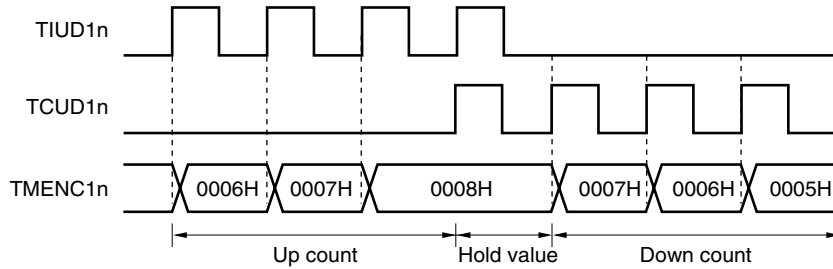
(ii) Mode 2 (PRM1n2 bit = 1, PRM1n1 bit = 0, PRM1n0 bit = 1)

The count conditions in mode 2 are as follows.

- TMENC1n up count upon detection of valid edge of TIUD1n pin
- TMENC1n down count upon detection of valid edge of TCUD1n pin

Caution If the count clock is simultaneously input to the TIUD1n pin and the TCUD1n pin, count operation is not performed and the immediately preceding value is held.

Figure 9-22. Mode 2 (When Rising Edge Is Specified as Valid Edge of TIUD1n, TCUD1n Pins)



Remark n = 0, 1

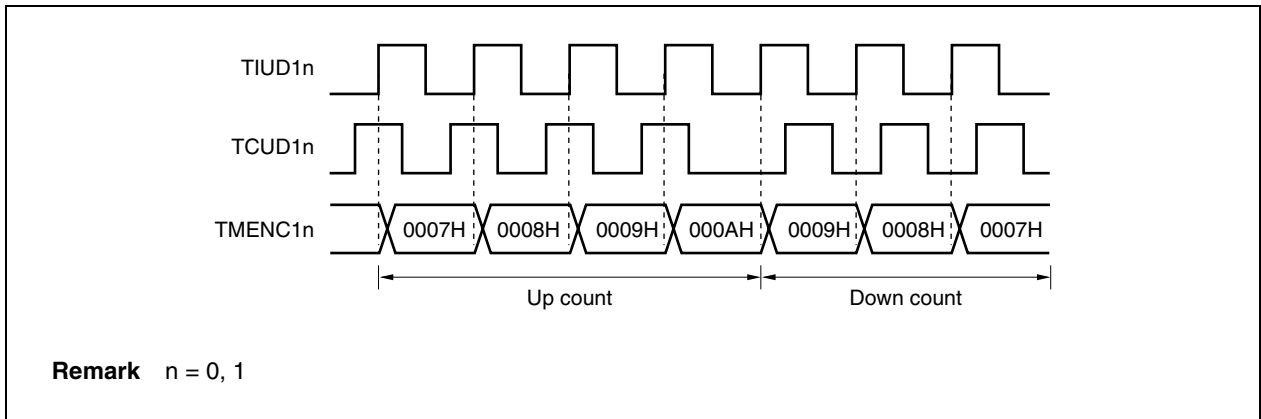
(iii) Mode 3 (PRM1n2 = 1, PRM1n1 = 1, PRM1n0 = 0)

In mode 3, when two signals 90 degrees out of phase are input to the TIUD1n and TCUD1n pins, the level of the TCUD1n pin is sampled at the input of the valid edge of the TIUD1n pin (see **Figure 9-23**).

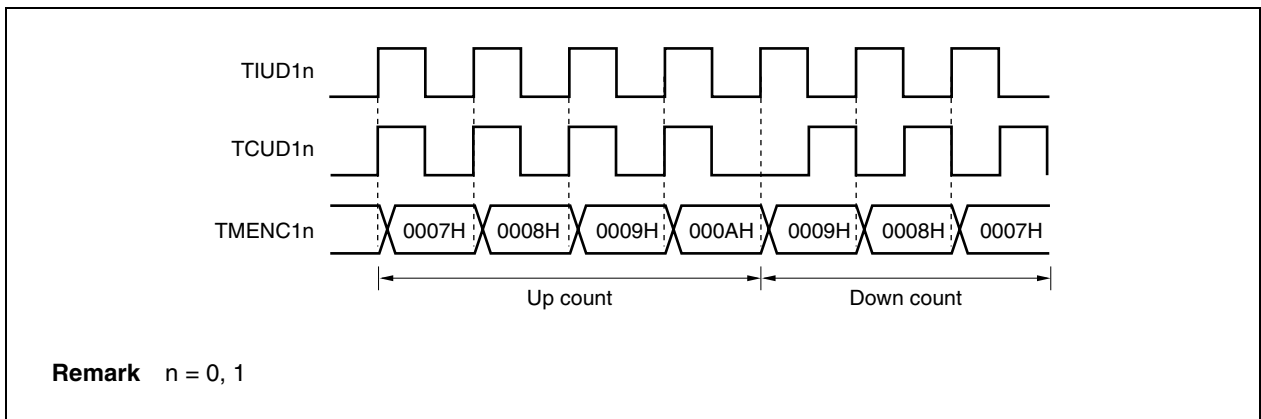
If the TCUD1n pin level sampled at the valid edge input to the TIUD1n pin is low, TMENC1n counts down when the valid edge is input to the TIUD1n pin.

If the TCUD1n pin level sampled at the valid edge input to the TIUD1n pin is high, TMENC1n counts up when the valid edge is input to the TIUD1n pin.

Figure 9-23. Mode 3 (When Rising Edge Is Specified as Valid Edge of TIUD1n pin)



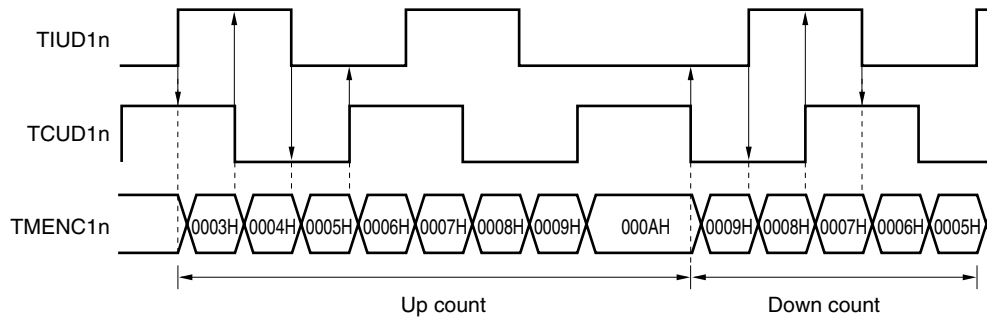
**Figure 9-24. Mode 3 (When Rising Edge Is Specified as Valid Edge of TIUD1n Pin):
In Case of Simultaneous TIUD1n, TCUD1n Pin Edge Timing**



(iv) Mode 4 (PRM1n2 = 1, PRM1n1 = 1, PRM1n0 = 1)

In mode 4, when two signals out of phase are input to the TIUD1n and TCUD1n pins, up/down operation is automatically judged and counting is performed according to the timing shown in **Figure 9-25**.

In mode 4, counting is executed at both the rising and falling edges of the two signals input to the TIUD1n and TCUD1n pins. Therefore, TMENC1n counts four times per cycle of an input signal ($\times 4$ count).

Figure 9-25. Mode 4

- Cautions**
1. When mode 4 is specified as the operation mode of TMENC1n, the valid edge specifications for the TIUD1n and TCUD1n pins are not valid.
 2. If the TIUD1n pin edge and TCUD1n pin edge are input simultaneously in mode 4, TMENC1n continues the same count operation (up or down) it was performing immediately before the input.

Remark n = 0, 1

(c) Operation in UDC mode A**(i) Interval operation**

The operations at the count clock following a match of the TMENC1n count value and the CM1n0 set value are as follows.

- In case of up count operation: TMENC1n is cleared to 0000H and the INTCM1n0 interrupt is generated.
- In case of down count operation: The TMENC1n count value is decremented (-1) and the INTCM1n0 interrupt is generated.

Remark The interval operation can be combined with the transfer operation.

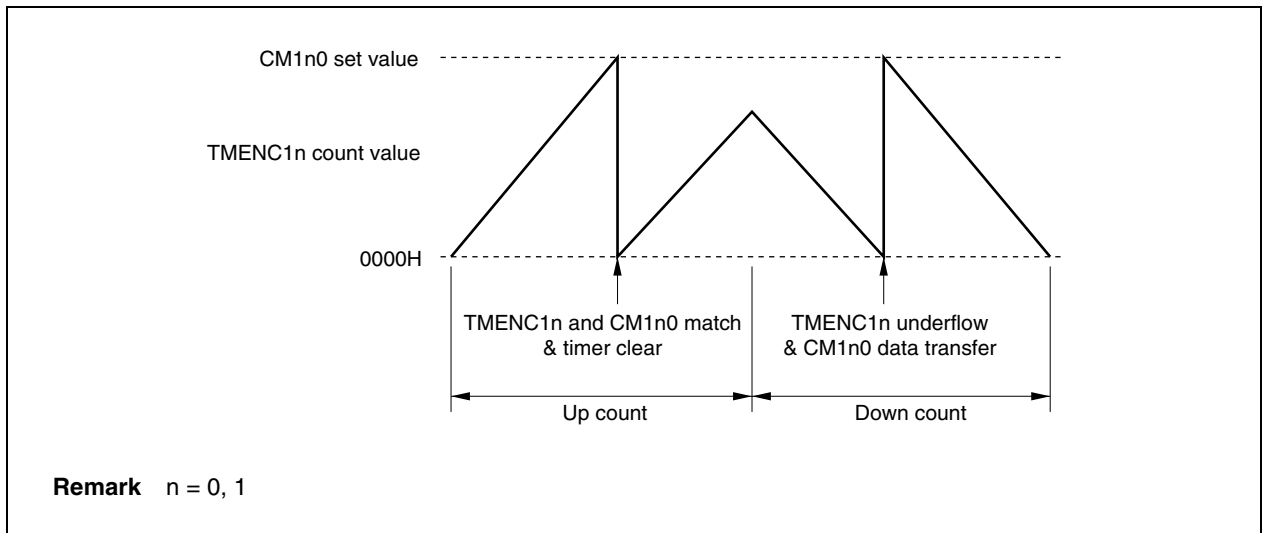
(ii) Transfer operation

The operations at the next count clock after the count value of TMENC1n becomes 0000H during a TMENC1n count down operation are as follows.

- In case of down count operation: The data held in CM1n0 is transferred.
- In case of up count operation: The TMENC1n count value is incremented (+1).

- Remarks**
1. Transfer enable/disable can be set using the RLEN1n bit of the TMC1n register.
 2. The transfer operation can be combined with the interval operation.

Figure 9-26. Example of TMENC1n Operation When Interval Operation and Transfer Operation Are Combined



(iii) Compare function

TMENC1n connects two compare register (CM1n0, CM1n1) channels and two capture/compare register (CC1n0, CC1n1) channels.

When the TMENC1n count value and the set value of one of the compare registers match, a match interrupt (INTCM1n0, INTCM1n1, INTCC1n0^{Note}, INTCC1n1^{Note}) is output.

Note This match interrupt is generated when CC1n0 and CC1n1 are set to the compare register mode.

(iv) Capture function

TMENC1n connects two capture/compare register (CC1n0, CC1n1) channels.

When CC1n0 and CC1n1 are set to the capture register mode, the value of TMENC1n is captured in synchronization with the corresponding capture trigger signal.

When CC1n0 and CC1n1 are set to the capture register mode, a capture interrupt (INTCC1n0, INTCC1n1) is generated upon detection of the valid edge.

(d) Operation in UDC mode B**(i) Basic operation**

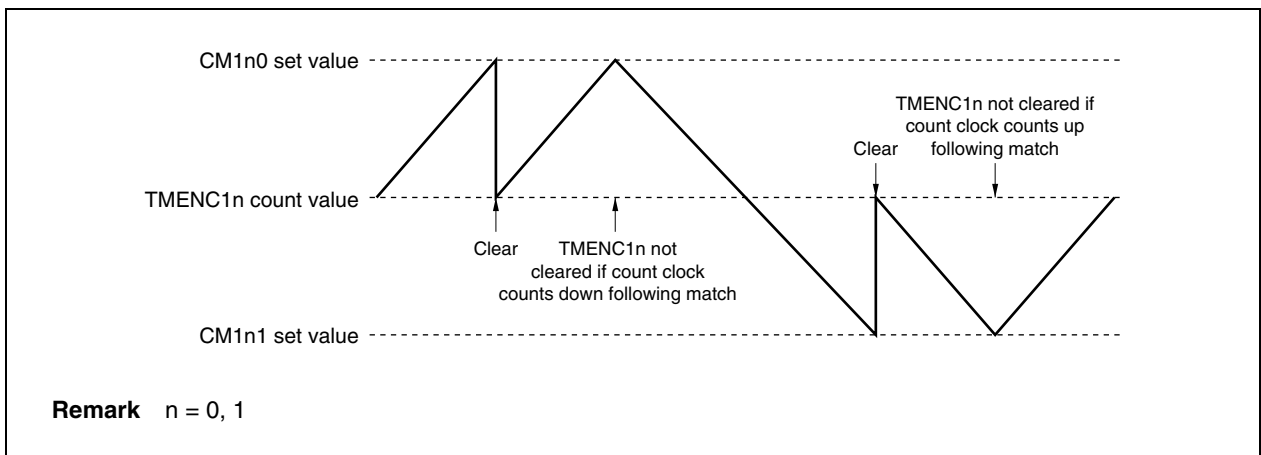
The operations at the next count clock after the count value of TMENC1n and the CM1n0 set value match when TMENC1n is in UDC mode B are as follows.

- In case of up count operation: TMENC1n is cleared to 0000H and the INTCM1n0 interrupt is generated.
- In case of down count operation: The TMENC1n count value is decremented (-1).

The operations at the next count clock after the count value of TMENC1n and the CM1n1 set value match when TMENC1n is in UDC mode B are as follows.

- In case of up count operation: The TMENC1n count value is incremented (+1).
- In case of down count operation: TMENC1n is cleared to 0000H and the INTCM1n1 interrupt is generated.

Figure 9-27. Example of TMENC1n Operation in UDC Mode

**(ii) Compare function**

TMENC1n connects two compare register (CM1n0, CM1n1) channels and two capture/compare register (CC1n0, CC1n1) channels.

When the TMENC1n count value and the set value of one of the compare registers match, a match interrupt (INTCM1n0 (only during up count operation), INTCM1n1 (only during down count operation), INTCC1n0^{Note}, INTCC1n1^{Note}) is output.

Note This match interrupt is generated when CC1n0 and CC1n1 are set to the compare register mode.

(iii) Capture function

TMENC1n connects two capture/compare register (CC1n0, CC1n1) channels.

When CC1n0 and CC1n1 are set to the capture register mode, the value of TMENC1n is captured in synchronization with the corresponding capture trigger signal.

When CC1n0 and CC1n1 are set to the capture register mode, a capture interrupt (INTCC1n0, INTCC1n1) is generated upon detection of the valid edge.

9.3.7 Supplementary description of internal operation

(1) Clearing of count value in UDC mode B

When TMENC1n is in UDC mode B, the count value clear operation is as follows.

- In case of TMENC1n up count operation: TMENC1n is cleared upon match with CM1n0
- In case of TMENC1n down count operation: TMENC1n is cleared upon match with CM1n1

Figure 9-28. Clear Operation upon Match with CM1n0 During TMENC1n Up Count Operation

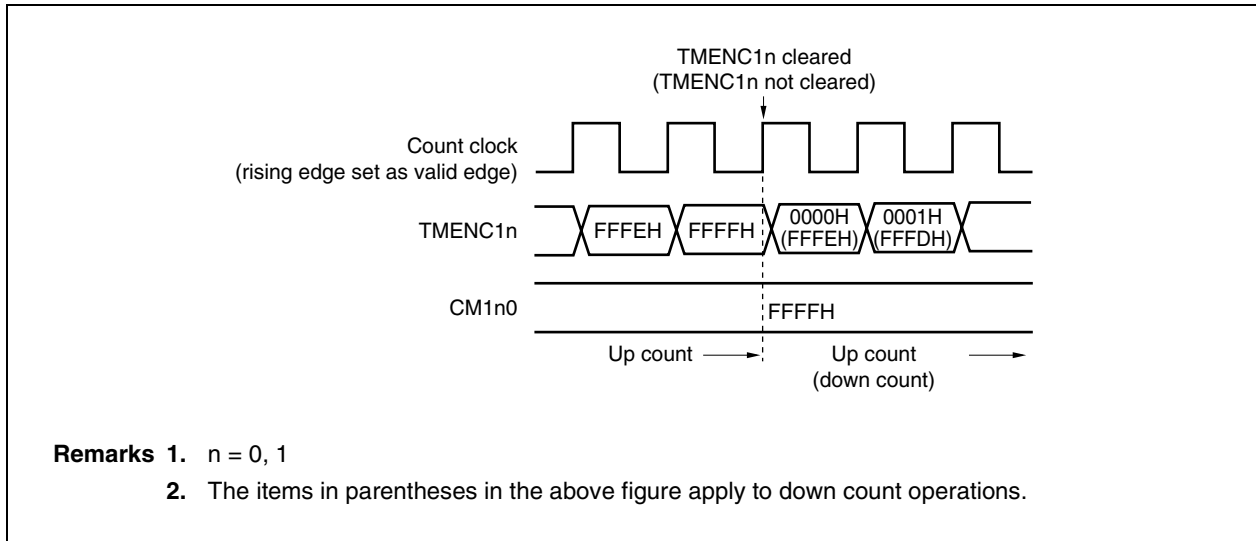
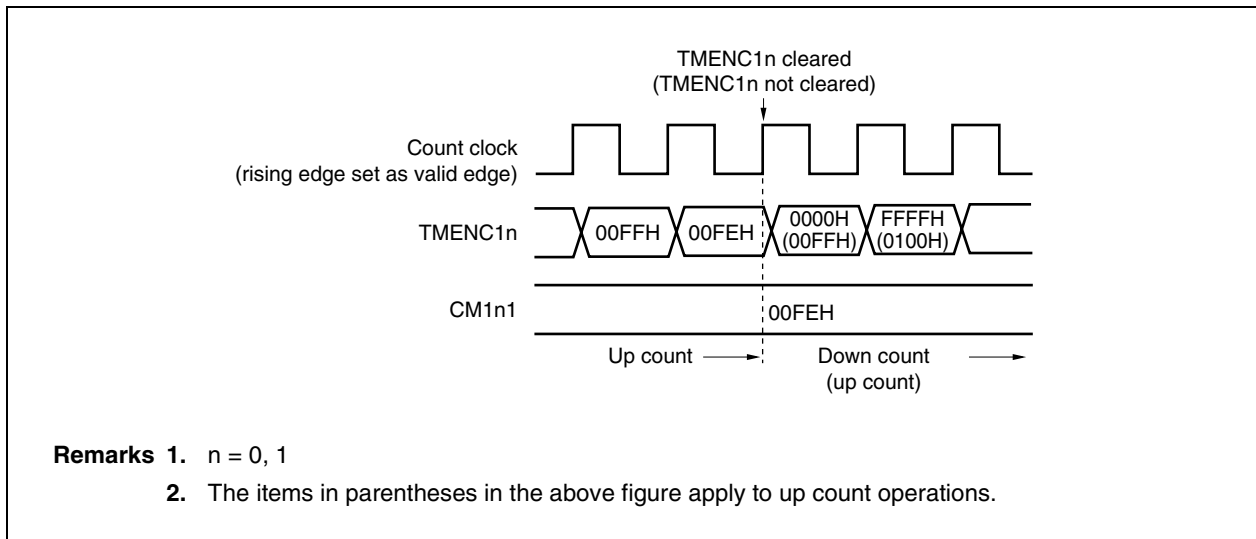


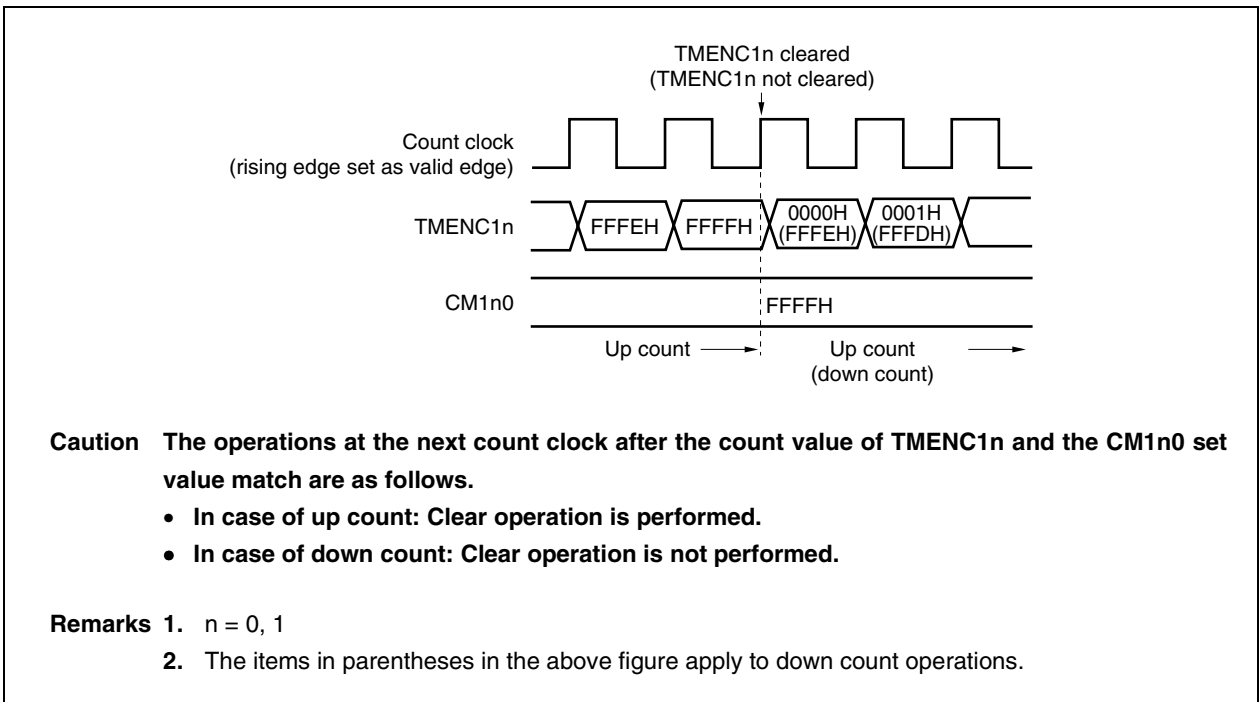
Figure 9-29. Clear Operation upon Match with CM1n1 During TMENC1n Down Count Operation



(2) Clearing of count value upon occurrence of compare match

The internal operation during a TMENC1n clear operation upon occurrence of a compare match is as follows.

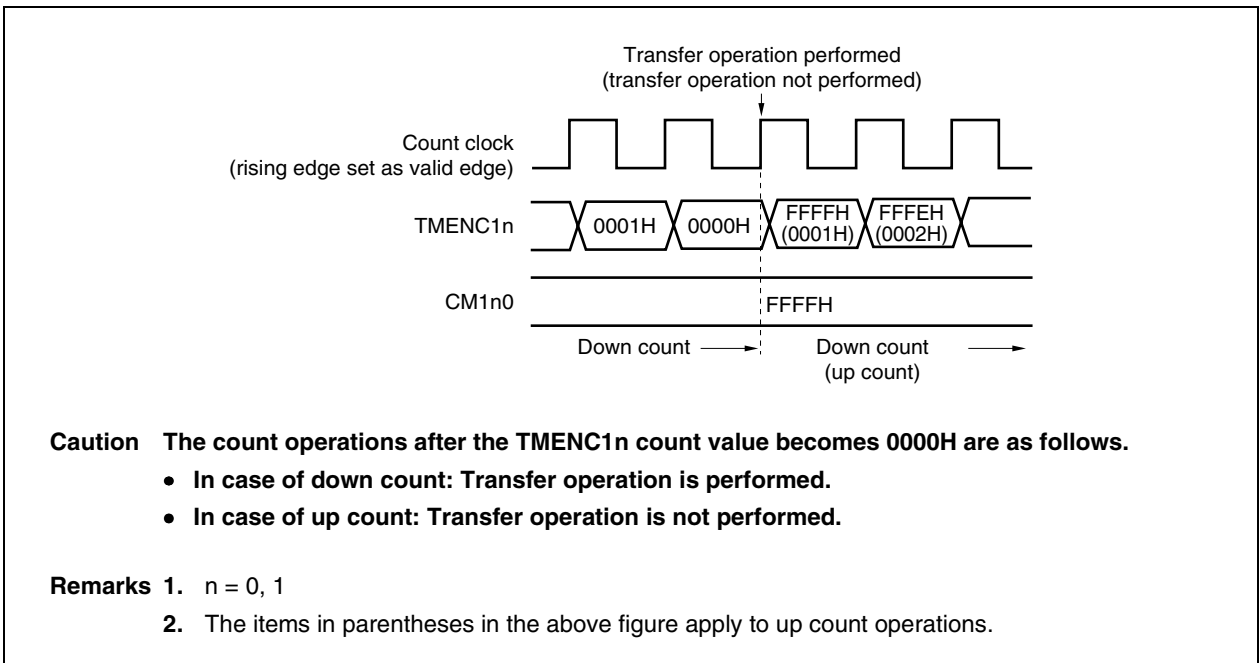
Figure 9-30. Count Value Clear Operation upon Compare Match



(3) Transfer operation

The internal operation during TMENC1n transfer operation is as follows.

Figure 9-31. Internal Operation During Transfer Operation

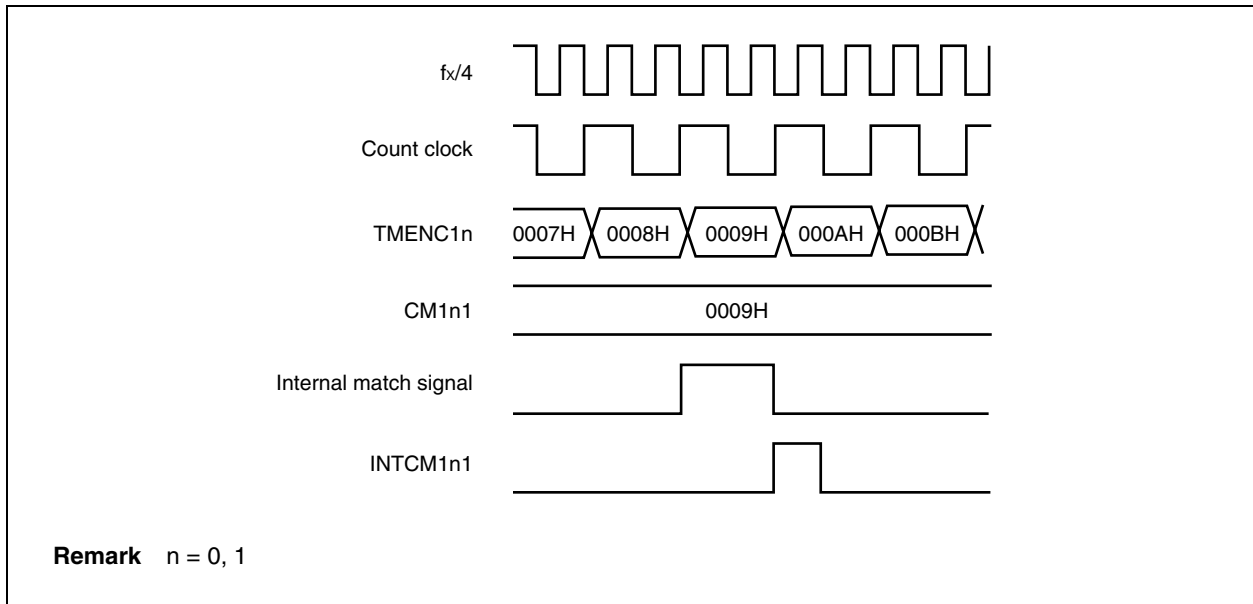


(4) Interrupt signal output upon compare match

An interrupt signal is output when the count value of TMENC1n matches the set value of the CM1n0, CM1n1, CC1n0^{Note}, or CC1n1^{Note} register. The interrupt generation timing is as follows.

Note When CC1n0 and CC1n1 are set to the compare register mode.

**Figure 9-32. Interrupt Output upon Compare Match
(CM1n1 with Operation Mode Set to General-Purpose Timer Mode and Count Clock Set to $f_x/8$)**

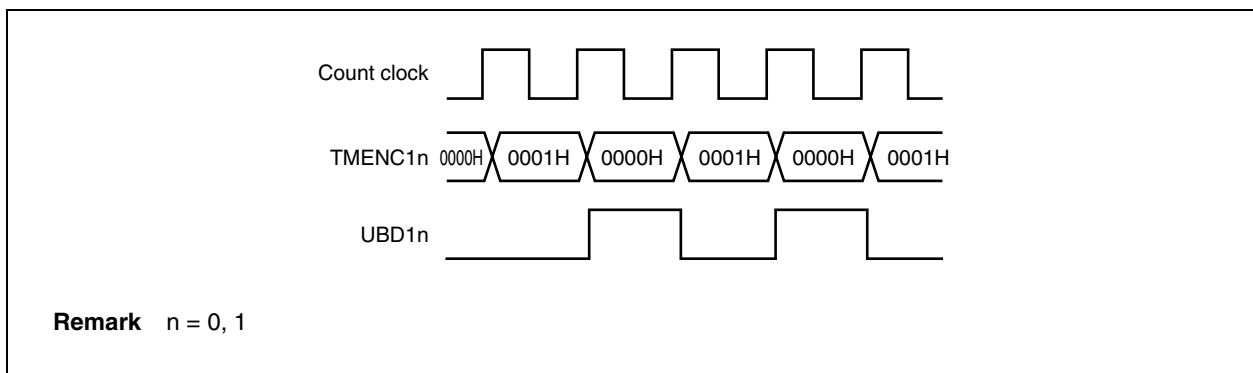


An interrupt signal such as the one illustrated in Figure 9-32 is output at the next count following a match of the TMENC1n count value and the set value of the corresponding compare register.

(5) UBD1n flag (bit 0 of STATUS1n register) operation

In the UDC mode (T1CMDn bit of TUM1n register = 1), the UBD1n flag changes as follows during TMENC1n up/down count operation at every internal operation clock.

Figure 9-33. UBD1n Flag Operation



★

(6) Overflow interrupt signal (INTOV1n) and underflow interrupt signal (INTUD1n)

- (a) The overflow interrupt signal (INTOV1n) is generated when the count value of TMENC1n has reached FFFFH and the next count operation is an up-count.
- (b) The underflow interrupt signal (INTUD1n) is generated when the count value of TMENC1n has reached 0000H and the next count operation is a down-count.
- (c) TMENC1n continues counting even after occurrence of an overflow or underflow, if a count edge is detected. If the condition of the underflow (0000H → FFFFH) is satisfied when the RLEN1n bit of the TMC1n register is 1 (enabling transfer), however, the set value of CM1n0 is transferred.
- (d) If the condition of the underflow (0000H → FFFFH) is satisfied with CM1n0 = FFFFH when the RLEN1n bit of the TMC1n register is cleared to 0 (disabling transfer), the INTCM1n0 interrupt and INTUD1n interrupt occur simultaneously.

CHAPTER 10 SERIAL INTERFACE FUNCTION

10.1 Features

The serial interface function provides three types of serial interfaces equipped with five transmit/receive channels of which four channels can be used simultaneously.

The following three interface formats are available.

- (1) Asynchronous serial interfaces B0 and B1 (UARTB0 and UARTB1): 2 channels
- (2) Clocked serial interfaces 30 and 31 (CSI30 and CSI31): 2 channels
- (3) USB function controller (USBF): 1 channel

Remark For details of the USB function, see **CHAPTER 11 USB FUNCTION CONTROLLER (USBF)**.

UARTB0 and UARTB1, which use the method of transmitting/receiving one byte of serial data following a start bit, enable full-duplex communication to be performed.

CSI30 and CSI31 transfer data according to three types of signals (3-wire serial I/O). These signals are the serial clock ($\overline{SCK0}$ and $\overline{SCK1}$), serial input (SI0 and SI1), and serial output (SO0 and SO1) signals.

The USB supports full-speed transfer of 12 Mbps and consists of seven endpoints.

10.1.1 Switching between UARTB0 and CSI30 modes

In the V850E/ME2, since UARTB0 and CSI30 are alternate function pins, they cannot be used at the same time. The registers must be set in advance.

10.2 Asynchronous Serial Interfaces B0, B1 (UARTB0, UARTB1)

10.2.1 Features

- ★
 - Transfer rate: Maximum 1.5 Mbps (using a dedicated baud rate generator)
 - Full-duplex communications
 - Single mode and FIFO mode selectable
 - Single mode: 8-bit × 1-stage data register (UBnTX register or UBnRX register) is used for each of transmission and reception.
 - FIFO mode
 - Transmit FIFO: UBnTX register (8 bits × 16 stages).
 - Receive FIFO: UBnRXAP register (16 bits × 16 stages)
 - 2 bits of the higher 8 bits of the UBnRXAP register are for an error flag.
 - Two-pin configuration
 - TXDn: Transmit data output pin
 - RXDn: Receive data input pin
 - Reception error detection function
 - Overflow error (FIFO mode only)
 - Parity error
 - Framing error
 - Overrun error (single mode only)
 - Interrupt sources: 5 types
 - Reception error interrupt (UBTIREn)
 - Reception completion interrupt (UBTIRn)
 - Transmission completion interrupt (UBTITn)
 - FIFO transmission completion interrupt (UBTIFn) (FIFO mode only)
 - Reception timeout interrupt (UBTITOn) (FIFO mode only)
 - The character length of transmit/receive data is specified according to the UBnCTL0 register
 - Character length: 7 or 8 bits
 - Parity functions: Odd, even, 0, or none
 - Transmission stop bits: 1 or 2 bits
 - MSB first/LSB first selectable for transfer data
 - On-chip dedicated baud rate generator

Remark n = 0, 1

10.2.2 Configuration

UARTBn is controlled by UARTBn control register 0 (UBnCTL0), the UARTBn status register (UBnSTR), UARTBn control register 2 (UBnCTL2), UARTBn FIFO control register 0 (UBnFIC0), UARTBn FIFO control register 1 (UBnFIC1), UARTBn FIFO control register 2 (UBnFIC2), UARTBn FIFO status register 0 (UBnFIS0), and UARTBn FIFO status register 1 (UBnFIS1) ($n = 0, 1$). Receive data is stored in a receive data register (the UBnRX register in the single mode or receive FIFO in the FIFO mode (the UBnRXAP register)) and transmit data is written to a transmit data register (the UBnTX register in the single mode or transmit FIFO in the FIFO mode). If a reception error (such as a parity error or a framing error) occurs in the FIFO mode, the error data can be identified by reading UARTBn receive data register AP (UBnRXAP) in 16-bit (halfword) units.

Figure 10-1 shows the configuration of the asynchronous serial interface.

(1) UARTBn control register 0 (UBnCTL0) ($n = 0, 1$)

This register controls the transfer operation of UARTBn.

(2) UARTBn status register (UBnSTR) ($n = 0, 1$)

This register indicates the transfer status during transmission and the contents of a reception error. The status flag of this register, which indicates the transfer status during transmission, indicates the data retention status of transmit shift register n and transmit data register n (the UBnTX register in the single mode or transmit FIFO in the FIFO mode). Each reception error flag is set to 1 when a reception error occurs, and cleared to 0 when 0 is written to the UBnSTR register.

(3) UARTBn control register 2 (UBnCTL2) ($n = 0, 1$)

This register is used to specify the division rate by which to control the baud rate (serial transfer speed) of UARTBn.

(4) UARTBn FIFO control register 0 (UBnFIC0) ($n = 0, 1$)

This register is used to select the operation mode of UARTBn, clear the transmit FIFO/receive FIFO that becomes valid in the FIFO mode, and specify the timing mode in which the transmission completion interrupt (UBTITn)/reception completion interrupt (UBTIRn) occurs.

(5) UARTBn FIFO control register 1 (UBnFIC1) ($n = 0, 1$)

This register is valid in the FIFO mode. It generates a reception timeout interrupt request (UBTITOn) if data is stored in the receive FIFO when the next data does not come (start bit is not detected) even after the reception wait time of the next data has elapsed.

(6) UARTBn FIFO control register 2 (UBnFIC2) ($n = 0, 1$)

This register is valid in the FIFO mode. It is used to set the timing to generate the transmission completion interrupt (UBTITn)/reception completion interrupt (UBTIRn), using the number of data transmitted or received as a trigger.

(7) UARTBn FIFO status register 0 (UBnFIS0) ($n = 0, 1$)

This register is valid in the FIFO mode. The number of bytes of data stored in the receive FIFO can be read from this register.

(8) UARTBn FIFO status register 1 (UBnFIS1) ($n = 0, 1$)

This register is valid in the FIFO mode. The number of vacant bytes of the transmit FIFO can be read from this register.

(9) Receive shift register n (n = 0, 1)

This is a shift register that converts the serial data that was input to the RXDn pin into parallel data. One byte of data is received, and if a stop bit is detected, the received data is transferred to receive data register n.

This register cannot be directly manipulated.

★ **(10) UARTBn receive data register AP (UBnRXAP), UARTBn receive data register (UBnRX) (n = 0, 1)**

Receive data register n holds receive data. In the single mode, the 8-bit × 1-stage UBnRX register is used. The 16-bit × 16-stage receive FIFO n (UBnRXAP register) is used in the FIFO mode.

The receive data is stored in the lower 8 bits of the receive FIFO n (UBnRXAP register) and the error information of the received data is stored in the higher 8 bits (bit 8 and bit 9). If a reception error (such as a parity error or a framing error) occurs in the FIFO mode, the error data can be identified by reading the UBnRXAP register in 16-bit (halfword) units (error information is appended as UBnPEF bit = 1 or UBnFEF bit = 1). When the lower 8 bits of the UBnRXAP register are read in 8-bit (byte) units, the higher 8 bits are discarded. Therefore, if no error has occurred, the receive data of the UBnRXAP register can be read consecutively by being read in 8-bit (byte) units in the same way as the UBnRX register.

When 7-bit length data is received with the LSB first, the received data is transferred to bits 6 to 0 of receive data register n from the LSB (bit 0), with the MSB (bit 7) always being 0. When data is received with the MSB first, the received data is transferred to bits 7 to 1 of receive data register n from the MSB (bit 7), with the LSB (bit 0) always being 0. If an overrun error occurs, the receive data at that time is not transferred to receive data register n.

While reception is enabled, the received data is transferred from receive shift register n to receive data register n, in synchronization with the shift-in processing of one frame.

A reception completion interrupt request (UBTIRn) is generated by transferring the data to the UBnRX register in the single mode, or transferring the number of receive data set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register to receive FIFO n in the FIFO mode. If data is stored in receive FIFO n when the next data does not come (start bit is not detected) after the next data reception wait time specified by the UBnTC4 to UBnTC0 bits of the UBnFIC1 register has elapsed in the FIFO mode, a reception timeout interrupt request (UBTITOn) is generated.

(11) Transmit shift register n (n = 0, 1)

This is a shift register that converts the parallel data that was transferred from transmit data register n into serial data.

When one byte of data is transferred from transmit data register n, transmit shift register n data is output from the TXDn pin.

This register cannot be directly manipulated.

(12)UARTBn transmit data register n (UBnTX) (n = 0, 1)

Transmit data register n is a buffer for transmit data. The 8-bit × 1-stage UBnTX register is used as this buffer in the single mode. In the FIFO mode, the 8-bit × 16-stage transmit FIFO is used.

- ★ When 7-bit length data is transmitted with the LSB first, bits 6 to 0 of transmit data register n are transmitted as the transmit data from the LSB (bit 0) with the MSB (bit 7) always being 0. When data is transmitted with the MSB first, bits 7 to 1 of transmit data register n are transmitted as the transmit data from the MSB (bit 7) with the LSB (bit 0) always being 0.

In the single mode, transmission is started by writing transmit data to the UBnTX register while transmission is enabled (UBnTXE bit = 1 in the UBnCTL0 register). When writing the transmit data to the UBnTX register is enabled (when 1-byte data is transferred from the UBnTX register to transmit shift register n), a transmission completion interrupt request (UBTITn) is generated.

In the FIFO mode, transmission is started by writing at least the number of transmit data set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register and 16 bytes or less to transmit FIFO and then enabling transmission (UBnTXE bit = 1). When the number of transmit data set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register have been transferred from transmit FIFO to transmit shift register n (transmit data of the number set as the trigger can be written), a transmission completion interrupt request (UBTITn) is generated. In the FIFO mode, a FIFO transmission completion interrupt request (UBTIFn) is generated when there is no more data in transmit FIFO and transmit shift register n (when FIFO and the register become empty).

(13)Timeout counter

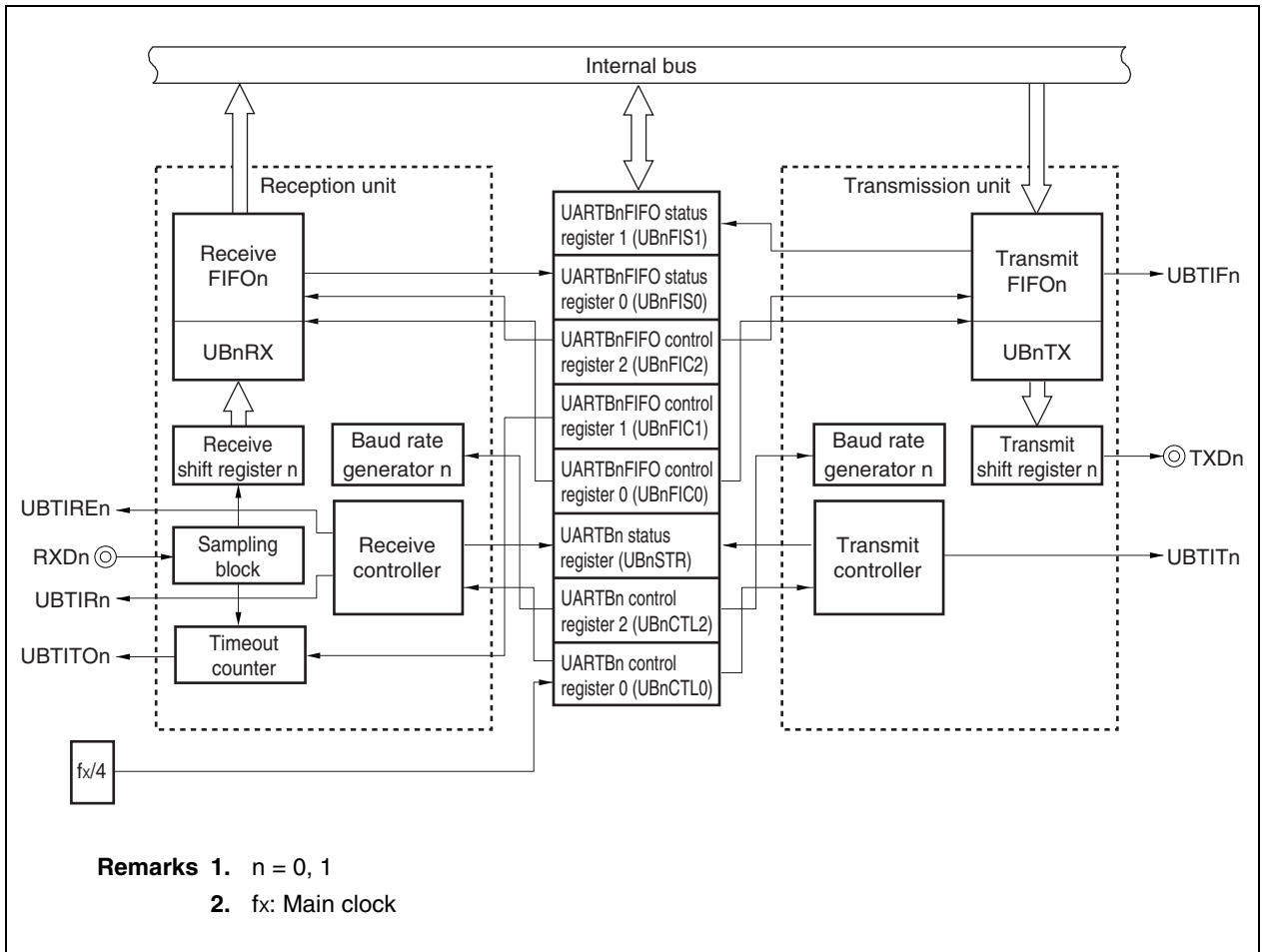
This counter is used to recognize that data exists (remains) in receive FIFO when the number of received data does not reach the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register, and is valid only in the FIFO mode.

If data is stored in receive FIFO when the next data does not come (start bit is not detected) after the next data reception wait time specified by the UBnTC4 to UBnTC0 bits of the UBnFIC1 register has elapsed after the stop bit has been received, a reception timeout interrupt request (UBTITOn) is generated.

(14) Sampling block

This block samples the RXDn signal at the rising edge of the input clock ($f_x/4$) (f_x : main clock). If the same sampling value is detected two times, output of the match detector changes, and the value is sampled as input data. Data of less than one clock width is judged as noise and is not transmitted to the internal circuitry.

Figure 10-1. Block Diagram of Asynchronous Serial Interfaces B0 and B1



10.2.3 Control registers

(1) UARTBn control register 0 (UBnCTL0) (n = 0, 1)

The UBnCTL0 register controls the transfer operations of UARTBn.
This register can be read or written in 8-bit or 1-bit units.

- Cautions**
1. When using UARTBn, set the external pins related to the UARTBn function in the control mode, set UARTBn control register 2 (UBnCTL2). Then set the UBnPWR bit to 1 before setting the other bits.
 2. Be sure to input a high level to the RXDn pin when setting the external pins related to the UARTBn function in the control mode. If a low level is input, it is judged that a falling edge is input after the UBnRXE bit has been set to 1, and reception may be started.

(1/3)

	7	6	5	4	3	2	1	0	Address	After reset
UB0CTL0	UB0PWR	UB0TXE	UB0RXE	UB0DIR	UB0PS1	UB0PS0	UB0CL	UB0SL	FFFFFA00H	10H
UB1CTL0	UB1PWR	UB1TXE	UB1RXE	UB1DIR	UB1PS1	UB1PS0	UB1CL	UB1SL	FFFFFA20H	10H

Bit position	Bit name	Function
7	UBnPWR	<p>Controls the operation clock.</p> <p>0: Stops supply of clocks to UARTBn 1: Supplies clocks to UARTBn</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. When the UBnPWR bit is cleared to 0, the UARTBn can be asynchronously reset. 2. When UBnPWR = 0, the UARTBn is in a reset state. Therefore, to operate UARTBn, the UBnPWR bit must be set to 1. 3. When the UBnPWR bit is changed from 1 to 0, all registers of the UARTBn are initialized. When the UBnPWR is set to 1 again, the UARTBn registers must be set again. <p>The TXDn pin output is high level when the UBnPWR bit is cleared to 0.</p>
6	UBnTXE	<p>Specifies whether transmission is enabled or disabled.</p> <p>0: Transmission is disabled 1: Transmission is enabled</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. On startup, set UBnPWR to 1 and then set UBnTXE to 1. To stop transmission, clear UBnTXE to 0 and then UBnPWR to 0. 2. When the transmission unit status is to be initialized, the transmission status may not be able to be initialized unless the UBnTXE bit is set to 1 again after an interval of two cycles of $f_x/4$ (f_x: main clock) has elapsed since the UBnTXE bit was cleared to 0.

Remark n = 0, 1

Bit position	Bit name	Function																				
5	UBnRXE	<p>Specifies whether reception is enabled or disabled.</p> <p>0: Reception is disabled 1: Reception is enabled</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. On startup, set UBnPWR to 1 and then set UBnRXE to 1. To stop reception, clear UBnRXE to 0 and then UBnPWR to 0. 2. When the reception unit status is to be initialized, the reception status may not be able to be initialized unless the UBnRXE bit is set to 1 again after an interval of two cycles of $f_x/4$ (f_x: main clock) has elapsed since the UBnRXE bit was cleared to 0. 																				
4	UBnDIR	<p>Specifies the transfer direction mode (MSB/LSB).</p> <p>0: The first bit of transfer data is the MSB. 1: The first bit of transfer data is the LSB.</p> <p>Caution Before changing the setting of the UBnDIR bit, clear the UBnPWR bit or UBnTXE and UBnRXE bits to 0.</p>																				
3, 2	UBnPS1, UBnPS0	<p>Controls the parity bit.</p> <table border="1"> <thead> <tr> <th>UBnPS1</th> <th>UBnPS0</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 UBnPS1 and UBnPS0 bits, first clear the UBnTXE and UBnRXE bits to 0. 2. If “0 parity” is selected for reception, no parity judgement is made. Therefore, no error interrupt is generated because the UBnPE bit of the UBnSTR register is not set to 1. <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 to 1. If it contains an even number of bits with the value “1”, the parity bit is cleared to 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. 	UBnPS1	UBnPS0	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
UBnPS1	UBnPS0	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																			

Remark n = 0, 1

Bit position	Bit name	Function
3, 2	UBnPS1, UBnPS0	<ul style="list-style-type: none"> • 0 parity During transmission, the parity bit is cleared to 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.
1	UBnCL	<p>Specifies the character length of the transmit/receive data.</p> <p>0: 7 bits 1: 8 bits</p> <p>Caution To overwrite the UBnCL bit, first clear the UBnTXE and UBnRXE bits to 0.</p>
0	UBnSL	<p>Specifies the stop bit length of the transmit data.</p> <p>0: 1 bit 1: 2 bits</p> <p>Cautions 1. To overwrite the UBnSL bit, first clear the UBnTXE bit to 0. 2. Since reception always operates by using a single stop bit length, the UBnSL bit setting does not affect receive operations.</p>

Remarks 1. When reception is disabled, receive shift register n does not detect a start bit. No shift-in processing or transfer processing to receive data register n is performed, and the contents of receive data register n are retained.

When reception is enabled, the receive shift operation starts, in synchronization with the detection of the start bit, and when the reception of one frame is completed, the contents of receive shift register n are transferred to receive data register n. A reception completion interrupt (UBTIRn) is also generated, in synchronization with the transfer to receive data register n (in FIFO mode, transfer triggered by reaching set number of receive data).

If data is stored in receive FIFO n when the next data does not come (start bit is not detected) after the next data reception wait time specified by the UBnTC4 to UBnTC0 bits of the UBnFIC1 register has elapsed in the FIFO mode, a reception timeout interrupt request (UBTITOn) is generated.

2. n = 0, 1

(2) UARTBn status register (UBnSTR) (n = 0, 1)

The UBnSTR register indicates the transfer status and reception error contents while UARTBn is transmitting data.

The status flag that indicates the transfer status during transmission indicates the data retention status of transmit shift register n and transmit data register n (the UBnTX register in the single mode or transmit FIFO n in the FIFO mode). The status flag that indicates a reception error holds its status until it is cleared to 0.

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

Caution When the UBnPWR bit or UBnRXE bit of the UBnCTL0 register is cleared to 0, or when 0 is written to the UBnSTR register, the UBnOVF, UBnPE, UBnFE, and UBnOVE bits of the UBnSTR register are cleared to 0.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UB0STR	UB0TSF	0	0	0	UB0OVF	UB0PE	UB0FE	UB0OVE	FFFFFA04H	00H
UB1STR	UB1TSF	0	0	0	UB1OVF	UB1PE	UB1FE	UB1OVE	FFFFFA24H	00H

Bit position	Bit name	Function
7	UBnTSF	<p>This is a status flag indicating the transfer status.</p> <ul style="list-style-type: none"> • In single mode (UBnMOD bit = 0 in the UBnFIC0 register) <ul style="list-style-type: none"> 0: Data to be transferred to transmit shift register n and UBnTX register does not exist (cleared (0) when UBnPWR bit = 0 or UBnTXE bit = 0 in the UBnCTL0 register). 1: Data to be transferred to transmit shift register n or UBnTX register exists (transmission in progress). • In FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register) <ul style="list-style-type: none"> 0: Data to be transferred to transmit shift register n and transmit FIFO n does not exist (cleared (0) when UBnPWR bit = 0 or UBnTXE bit = 0 in the UBnCTL0 register). 1: Data to be transferred to transmit shift register n and transmit FIFO n exists (transmission in progress). <p>Caution The value of the UBnTSF bit is reflected after two periods of $f_x/4$ (f_x: main clock) have elapsed, after the transmit data is written to the UBnTX register. Therefore, exercise care when referencing the UBnTSF bit after transmit data has been written to the UBnTX register.</p>
3	UBnOVF	<p>This is a status flag indicating an overflow. The setting of this flag is valid only in the FIFO mode (when UBnMOD bit = 1 in the UBnFIC0 register), and invalid in the single mode (when UBnMOD bit = 0 in the UBnFIC0 register).</p> <ul style="list-style-type: none"> 0: Overflow did not occur. 1: Overflow occurred (during reception). <p>Caution If an overflow occurs, the received data is not written to receive FIFO n but discarded.</p>

Remark n = 0, 1

Bit position	Bit name	Function
2	UBnPE	<p>This is a status flag that indicates a parity error. The setting of this flag is valid only in the single mode (when UBnMOD bit = 0 in the UBnFIC0 register), and invalid in the FIFO mode (when UBnMOD bit = 1 in the UBnFIC0 register).</p> <p>0: Parity error did not occur. 1: Parity error occurred (during reception).</p> <p>Caution The operation of the UBnPE bit differs according to the settings of the UBnPS1 and UBnPS0 bits of the UBnCTL0 register.</p>
1	UBnFE	<p>This is a status flag that indicates a framing error. The setting of this flag is valid only in the single mode (when UBnMOD bit = 0 in the UBnFIC0 register), and invalid in the FIFO mode (when UBnMOD bit = 1 in the UBnFIC0 register).</p> <p>0: Framing error did not occur. 1: Framing error occurred (during reception).</p> <p>Caution For receive data stop bits, only the first bit is checked regardless of the stop bit length.</p>
0	UBnOVE	<p>This is a status flag that indicates an overrun error. The setting of this flag is valid only in the single mode (when UBnMOD bit = 0 in the UBnFIC0 register), and invalid in the FIFO mode (when UBnMOD bit = 1 in the UBnFIC0 register).</p> <p>0: Overrun error did not occur. 1: Overrun error occurred (during reception).</p> <p>Caution When an overrun error occurs, the next receive data value is not written to the UBnRX register and the data is discarded.</p>

Remark n = 0, 1

(3) UARTBn control register 2 (UBnCTL2) (n = 0, 1)

The UBnCTL2 register is used to specify the division ratio by which to control the baud rate (serial transfer speed) of UARTBn.

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



Caution When rewriting the UBnBRS15 to UBnBRS0 bits of this register, clear the UBnTXE and UBnRXE bits of the UBnCTL0 register to 0 or clear the UBnPWR bit (n = 0, 1).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
UB0CTL2	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	UB0	FFFFFA02H	FFFFH
	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
UB1CTL2	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	UB1	FFFFFA22H	FFFFH
	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS	BRS		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		

Bit position	Bit name	Function
15 to 0	UBnBRS15 to UBnBRS0	Specify the division value of the 16-bit counter (see Table 10-1).

Remark n = 0, 1

Table 10-1. Division Value of 16-Bit Counter

UBnB RS15	UBnB RS14	UBnB RS13	UBnB RS12	UBnB RS11	UBnB RS10	UBnB RS9	UBnB RS8	UBnB RS7	UBnB RS6	UBnB RS5	UBnB RS4	UBnB RS3	UBnB RS2	UBnB RS1	UBnB RS0	k	Output Clock Selected
0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	4	$fx/(4 \times k)$
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	4	$fx/(4 \times k)$
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	5	$fx/(4 \times k)$
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	6	$fx/(4 \times k)$
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	65,532	$fx/(4 \times k)$
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	65,533	$fx/(4 \times k)$
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	65,534	$fx/(4 \times k)$
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	65,535	$fx/(4 \times k)$

- Remarks**
1. fx: Main clock
 2. k: Value set by UBnBRS15 to UBnBRS0 bits of UBnCTL2 register (k = 4, 5, 6, ..., 65,535)
 3. x: Don't care

(4) UARTBn transmit data register (UBnTX) (n = 0, 1)

The UBnTX register is used to set transmit data. It functions as the 8-bit × 1-stage UBnTX register, in the single mode (UBnMOD bit = 0 in the UBnFIC0 register), and as the 8-bit × 16-stage transmit FIFO in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register).

In the single mode, transmission is started by writing transmit data to the UBnTX register when transmission is enabled (the UBnTXE bit = 1 in the UBnCTL0 register). When data can be written to the UBnTX register (when 1 byte of data is transferred from the UBnTX register to transmit shift register n), a transmission completion interrupt request (UBTITn) is generated.

In the FIFO mode, transmission is started by enabling transmission (UBnTXE bit = 1) after writing at least the number of transmit data set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register and 16 bytes or less to transmit FIFO. When the number of transmit data set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register have been transferred from transmit FIFO to transmit shift register n (transmit data of the number set as the trigger can be written to transmit FIFO), a transmission completion interrupt request (UBTITn) is generated. In the FIFO mode, a FIFO transmission completion interrupt request (UBTIFn) is generated when there is no more data in transmit FIFO and transmit shift register n (when the FIFO and register become empty).

For the generation timing of the interrupt, see **10.2.4 Interrupt requests**.

★ When 7-bit length data is transmitted with the LSB first, bits 6 to 0 of transmit data register n are transmitted as the transmit data from the LSB (bit 0) with the MSB (bit 7) always being 0. When data is transmitted with the MSB first, bits 7 to 1 of transmit data register n are transmitted as the transmit data from the MSB (bit 7) with the LSB (bit 0) always being 0.

This register is write-only, in 8-bit units. Data is written to transmit data register n.

	7	6	5	4	3	2	1	0	Address	After reset
UB0TX	UB0TD7	UB0TD6	UB0TD5	UB0TD4	UB0TD3	UB0TD2	UB0TD1	UB0TD0	FFFFFA08H	FFH
UB1TX	UB1TD7	UB1TD6	UB1TD5	UB1TD4	UB1TD3	UB1TD2	UB1TD1	UB1TD0	FFFFFA28H	FFH

Bit position	Bit name	Function
7 to 0	UBnTD7 to UBnTD0	Write transmit data.

Remark n = 0, 1

★ (5) **UARTBn receive data register AP (UBnRXAP), UARTBn receive data register (UBnRX) (n = 0, 1)**

These registers store parallel data converted by receive shift register n. They function as the 8-bit × 1-stage UBnRX register, in the single mode (UBnMOD bit = 0 in the UBnFIC0 register), and as the 16-bit × 16-stage receive FIFO (UBnRXAP register) in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register).

The receive data is stored in the lower 8 bits of the receive FIFO (UBnRXAP register) and the error information of the received data is stored in the higher 8 bits (bit 8 and bit 9). If a reception error (such as a parity error or a framing error) occurs in the FIFO mode, the UBnRXAP register is read in 16-bit (halfword) units. In this way, the flag of the data stored in receive FIFO can be checked (error information is appended as UBnPEF bit = 1 or UBnFEF bit = 1), so that the error data can be recognized (when the lower 8 bits of the UBnRXAP register are read in 8-bit (byte) units, the higher 8 bits are discarded. Therefore, if no error has occurred, the receive data of the UBnRXAP register can be read consecutively by being read in 8-bit (byte) units in the same way as the UBnRX register).

If reception is enabled (UBnRXE bit = 1 in the UBnCTL0 register), the receive data is transferred from receive shift register n to receive data register n, in synchronization with the completion of the shift-in processing of one frame.

By transferring the receive data to the UBnRX register in the single mode or by transferring the number of receive data set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register to the receive FIFO in the FIFO mode, a reception completion interrupt request (UBnTIRn) is generated. If data is stored in receive FIFO when the next data does not come (start bit is not detected) even after the next data reception wait time specified by the UBnTC4 to UBnTC0 bits of the UBnFIC1 register has elapsed in the FIFO mode, a reception timeout interrupt request (UBnTITOn) is generated.

For information about the timing for generating these interrupt requests, see **10.2.4 Interrupt requests**.

If data is received with the LSB first when the data length is specified as 7 bits, the received data is transferred to bits 6 to 0 of receive data register n from the LSB (bit 0), with the MSB (bit 7) always being 0. If data is received with the MSB first, it is transferred to bits 7 to 1 of receive data register n from the MSB (bit 7) with the LSB (bit 0) always being 0. However, if an overrun error occurs, the receive data at that time is not transferred to receive data register n.

The UBnRXAP register is read-only, in 16-bit units. However, the lower 8 bits of the UBnRXAP register are read-only, in 8-bit units.

The UBnRX register is read-only, in 8-bit units.

In addition to reset input, the value of these registers can be set to FFH in the single mode or to 00FFH in the FIFO mode, by clearing the UBnPWR bit of the UBnCTL0 register to 0.

Cautions 1. Because these registers serve as 8-bit registers in the single mode, the UBnPEF and UBnFEF bits cannot be read.

- 2. When no reception error has occurred in the FIFO mode, the receive data of the UBnRXAP register can be read consecutively by reading the lower 8 bits of the UBnRXAP register in 8-bit (byte) units. An 8-bit access to the higher 8 bits is prohibited. If they are accessed, the operation is not guaranteed.**

[UARTBn receive data register AP]

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
UB0RXAP	0	0	0	0	0	0	UB0 PEF	UB0 FEF	UB0 RD7	UB0 RD6	UB0 RD5	UB0 RD4	UB0 RD3	UB0 RD2	UB0 RD1	UB0 RD0	FFFFFFA06H	00FFH
UB1RXAP	0	0	0	0	0	0	UB1 PEF	UB1 FEF	UB1 RD7	UB1 RD6	UB1 RD5	UB1 RD4	UB1 RD3	UB1 RD2	UB1 RD1	UB1 RD0	FFFFFFA26H	00FFH

[UARTBn receive data register]

	7	6	5	4	3	2	1	0	Address	After reset
UB0RX	UB0 RD7	UB0 RD6	UB0 RD5	UB0 RD4	UB0 RD3	UB0 RD2	UB0 RD1	UB0 RD0	FFFFFFA06H	FFH
UB1RX	UB1 RD7	UB1 RD6	UB1 RD5	UB1 RD4	UB1 RD3	UB1 RD2	UB1 RD1	UB1 RD0	FFFFFFA26H	FFH

Bit position	Bit name	Function
9 (UBnRXAP)	UBnPEF	This is a status flag that indicates a parity error. Status is valid only in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register), and is invalid in the single mode (UBnMOD bit = 0 in the UBnFIC0 register). 0: No parity error 1: Parity error occurs (during reception). Caution The operation of the UBnPEF bit differs depending on the set values of the UBnPS1 and UBnPS0 bits of the UBnCTL0 register.
8 (UBnRXAP)	UBnFEF	This is a status flag indicating a framing error. Status is valid only in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register), and is invalid in the single mode (UBnMOD bit = 0 in the UBnFIC0 register). 0: No framing error 1: Framing error occurs (during reception). Caution Only the first bit of the stop bits of the receive data is checked, regardless of the stop bit length.
7 to 0	UBnRD7 to UBnRD0	Stores receive data.

Remark n = 0, 1

(6) UARTBn FIFO control register 0 (UBnFIC0) (n = 0, 1)

The UBnFIC0 register is used to select the operation mode of UARTBn and the functions that become valid in the FIFO mode (UBnMOD bit = 1). In the FIFO mode, it clears transmit FIFO/receive FIFO and specifies the timing mode in which the transmission completion interrupt (UBTITn)/reception completion interrupt (UBTIRn) is generated.

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

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UB0FIC0	UB0MOD	0	0	0	UB0TFC	UB0RFC	UB0ITM	UB0IRM	FFFFFA0AH	00H
UB1FIC0	UB1MOD	0	0	0	UB1TFC	UB1RFC	UB1ITM	UB1IRM	FFFFFA2AH	00H

Bit position	Bit name	Function
7	UBnMOD	Specifies an operation mode of UARTBn. 0: Single mode 1: FIFO mode
3	UBnTFC	This is a transmit FIFO clear trigger bit. The setting of this bit is valid only in the FIFO mode (UBnMOD bit = 1) and invalid in the single mode (UBnMOD bit = 0). 0: Normal status 1: Clear (This bit automatically returns to 0 after transmit FIFO is cleared.) When 1 is written to the UBnTFC bit, the pointer to transmit FIFO is cleared to 0. In the pending mode (UBnITM bit = 0), the interrupt request (UBTITn) held pending is cleared ^{Note} . However, bit 7 (UTIFn) of the interrupt control register (UTICn) is not cleared to 0. Clear this bit to 0 as necessary. When 0 is written to the UBnTFC bit, the status is retained. No operation, such as clearing or setting, is executed. Note After transmit FIFO is cleared (UBnTFC bit = 1), accessing the registers related to UARTB is prohibited for the duration of four cycles of fx/4 (fx: main clock) or until clearing the UBnTFC bit (automatic recovery) is confirmed by reading the UBnFIC0 register. If these registers are accessed, the operation is not guaranteed. Caution When writing 1 to the UBnTFC bit, be sure to clear the UBnTXE bit of the UBnCTL0 register to 0 (disabling transmission). If 1 is written to the UBnTFC bit when the UBnTXE bit is 1 (transmission enabled), the operation is not guaranteed.

Remark n = 0, 1

★

★

★

Bit position	Bit name	Function
2	UBnRFC	<p>This is a receive FIFO (UBnRXAP) clear trigger bit. The setting of this bit is valid only in the FIFO mode (UBnMOD bit = 1) and invalid in the single mode (UBnMOD bit = 0).</p> <p>0: Normal status 1: Clear (This bit automatically returns to 0 after receive FIFO is cleared.)</p> <p>When 1 is written to the UBnRFC bit, the pointer to receive FIFO is cleared to 0. In the pending mode (UBnIRM bit = 0), the interrupt request (UBTIRn) held pending is cleared^{Note}. However, bit 7 (URIFn) of the interrupt control register (URICn) is not cleared to 0. Clear this bit to 0 as necessary.</p> <p>When 0 is written to the UBnRFC bit, the status is retained. No operation, such as clearing or setting, is executed.</p> <p>Note After receive FIFO (UBnRXAP) is cleared (UBnRFC bit = 1), accessing the registers related to UARTB is prohibited for the duration of four cycles of $f_x/4$ (f_x: main clock) or until clearing the UBnRFC bit (automatic recovery) is confirmed by reading the UBnFIC0 register. If these registers are accessed, the operation is not guaranteed.</p> <p>Caution When writing 1 to the UBnRFC bit, be sure to clear the UBnRXE bit of the UBnCTL0 register to 0 (disabling reception). If 1 is written to the UBnRFC bit when the UBnRXE bit is 1 (reception enabled), the operation is not guaranteed.</p>
1	UBnITM	<p>This bit specifies the timing mode in which the UBTITn interrupt is generated in FIFO mode.</p> <p>0: Pending mode 1: Pointer mode</p> <p>In the FIFO mode, the UBTITn interrupt is generated as soon as transmit data of the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register have been transferred from transmit FIFO to transmit shift register n. After the UBTITn interrupt request has been generated, specify the timing of actually generating the UBTITn interrupt as the pending mode or pointer mode. For details, refer to 10.2.5 (2) Pending mode/pointer mode.</p>
0	UBnIRM	<p>This bit specifies the timing mode in which the UBTIRn interrupt is generated in FIFO mode.</p> <p>0: Pending mode 1: Pointer mode</p> <p>In the FIFO mode, the UBTIRn interrupt is generated as soon as receive data of the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register have been transferred from receive shift register n to receive FIFO. After the UBTIRn interrupt request has been generated, specify the timing of actually generating the UBTIRn interrupt as the pending mode or pointer mode. For details, refer to 10.2.5 (2) Pending mode/pointer mode.</p>

Remark n = 0, 1

(7) UARTBn FIFO control register 1 (UBnFIC1) (n = 0, 1)

The UBnFIC1 register is valid in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register). It generates a reception timeout interrupt request (UBTITOn) if data is stored in receive FIFO when the next data does not come (start bit is not detected) after the lapse of the time set by the UBnTC4 to UBnTC0 bits (next data reception wait time), after the stop bit has been received.

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
UB0FIC1	UB0TCE	0	0	UB0TC4	UB0TC3	UB0TC2	UB0TC1	UB0TC0	FFFFFA0BH	00H
UB1FIC1	UB1TCE	0	0	UB1TC4	UB1TC3	UB1TC2	UB1TC1	UB1TC0	FFFFFA2BH	00H

Bit position	Bit name	Function																																																																								
7	UBnTCE	Specifies the timeout counter function. 0: Disable use of timeout counter function. 1: Enable use of timeout counter function.																																																																								
4 to 0	UBnTC4 to UBnTC0	Specify the time to wait for the next data reception. <table border="1"> <thead> <tr> <th>UBn TC4</th> <th>UBn TC3</th> <th>UBn TC2</th> <th>UBn TC1</th> <th>UBn TC0</th> <th>Next data reception wait time</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>32 bytes (32 × 8 / baud rate)</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>31 bytes (31 × 8 / baud rate)</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>30 bytes (30 × 8 / baud rate)</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>29 bytes (29 × 8 / baud rate)</td> </tr> <tr> <td>•</td> <td>•</td> <td>•</td> <td>•</td> <td>•</td> <td>•</td> </tr> <tr> <td>•</td> <td>•</td> <td>•</td> <td>•</td> <td>•</td> <td>•</td> </tr> <tr> <td>•</td> <td>•</td> <td>•</td> <td>•</td> <td>•</td> <td>•</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>4 bytes (4 × 8 / baud rate)</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>3 bytes (3 × 8 / baud rate)</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>2 bytes (2 × 8 / baud rate)</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1 byte (1 × 8 / baud rate)</td> </tr> </tbody> </table> <p>Caution When the count set by the UBnTC4 to UBnTC0 bits is up, the count value of the timeout counter is cleared to 0, regardless of the status of the data stored in receive FIFO. When the next start bit is later detected, counting is started again from the stop bit of that data.</p>	UBn TC4	UBn TC3	UBn TC2	UBn TC1	UBn TC0	Next data reception wait time	0	0	0	0	0	32 bytes (32 × 8 / baud rate)	0	0	0	0	1	31 bytes (31 × 8 / baud rate)	0	0	0	1	0	30 bytes (30 × 8 / baud rate)	0	0	0	1	1	29 bytes (29 × 8 / baud rate)	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	1	1	1	0	0	4 bytes (4 × 8 / baud rate)	1	1	1	0	1	3 bytes (3 × 8 / baud rate)	1	1	1	1	0	2 bytes (2 × 8 / baud rate)	1	1	1	1	1	1 byte (1 × 8 / baud rate)
UBn TC4	UBn TC3	UBn TC2	UBn TC1	UBn TC0	Next data reception wait time																																																																					
0	0	0	0	0	32 bytes (32 × 8 / baud rate)																																																																					
0	0	0	0	1	31 bytes (31 × 8 / baud rate)																																																																					
0	0	0	1	0	30 bytes (30 × 8 / baud rate)																																																																					
0	0	0	1	1	29 bytes (29 × 8 / baud rate)																																																																					
•	•	•	•	•	•																																																																					
•	•	•	•	•	•																																																																					
•	•	•	•	•	•																																																																					
1	1	1	0	0	4 bytes (4 × 8 / baud rate)																																																																					
1	1	1	0	1	3 bytes (3 × 8 / baud rate)																																																																					
1	1	1	1	0	2 bytes (2 × 8 / baud rate)																																																																					
1	1	1	1	1	1 byte (1 × 8 / baud rate)																																																																					

Remark n = 0, 1

(8) UART_n FIFO control register 2 (UBnFIC2) (n = 0, 1)

The UBnFIC2 register is valid in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register). It sets the timing of generating an interrupt, using the number of transmit/receive data as a trigger. When data is transmitted, the number of data transferred from transmit FIFO_n is specified as the condition of generating the interrupt. When data is received, the number of data stored in receive FIFO_n is specified as the interrupt generation condition.

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

When the higher 8 bits of the UBnFIC2 register can be used as the UBnFIC2H register and the lower 8 bits, as the UBnFIC2L register, these registers can be read or written in 8-bit units.

Caution Before writing data to the UBnFIC2 register, be sure to clear the UBnTXE bit (to disable transmission) and UBnRXE bit (to disable reception) of the UBnCTL0 register to 0. If data is written to the UBnFIC2 register with the UBnTXE or UBnRXE bit set to 1, the operation is not guaranteed.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
UB0FIC2	0	0	0	0	UB0 TT3	UB0 TT2	UB0 TT1	UB0 TT0	0	0	0	0	UB0 RT3	UB0 RT2	UB0 RT1	UB0 RT0	FFFFFFA0CH	0000H
UB1FIC2	0	0	0	0	UB1 TT3	UB1 TT2	UB1 TT1	UB1 TT0	0	0	0	0	UB1 RT3	UB1 RT2	UB1 RT1	UB1 RT0	FFFFFFA2CH	0000H

Bit position	Bit name	Function																																																																																										
11 to 8	UBnTT3 to UBnTT0	<p>Set the number of transmit FIFO on transmit data to be the trigger. Each time data of the specified number has shifted out from transmit FIFO on to transmit shift register n, the UBTITn interrupt is generated.</p> <p>In the pending mode (UBnITM bit = 0 in the UBnFIC0 register), the UBTITn interrupt is generated under the conditions of the pending mode.</p> <table border="1"> <thead> <tr> <th>UBn TT3</th> <th>UBn TT2</th> <th>UBn TT1</th> <th>UBn TT0</th> <th>Number of data of transmit FIFO set as trigger</th> <th>Pointer mode</th> <th>Pending mode</th> </tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>0</td><td>0</td><td>1 byte</td><td>Settable</td><td rowspan="16">Settable</td> </tr> <tr> <td>0</td><td>0</td><td>0</td><td>1</td><td>2 bytes</td><td rowspan="15">Setting prohibited</td> </tr> <tr> <td>0</td><td>0</td><td>1</td><td>0</td><td>3 bytes</td> </tr> <tr> <td>0</td><td>0</td><td>1</td><td>1</td><td>4 bytes</td> </tr> <tr> <td>0</td><td>1</td><td>0</td><td>0</td><td>5 bytes</td> </tr> <tr> <td>0</td><td>1</td><td>0</td><td>1</td><td>6 bytes</td> </tr> <tr> <td>0</td><td>1</td><td>1</td><td>0</td><td>7 bytes</td> </tr> <tr> <td>0</td><td>1</td><td>1</td><td>1</td><td>8 bytes</td> </tr> <tr> <td>1</td><td>0</td><td>0</td><td>0</td><td>9 bytes</td> </tr> <tr> <td>1</td><td>0</td><td>0</td><td>1</td><td>10 bytes</td> </tr> <tr> <td>1</td><td>0</td><td>1</td><td>0</td><td>11 bytes</td> </tr> <tr> <td>1</td><td>0</td><td>1</td><td>1</td><td>12 bytes</td> </tr> <tr> <td>1</td><td>1</td><td>0</td><td>0</td><td>13 bytes</td> </tr> <tr> <td>1</td><td>1</td><td>0</td><td>1</td><td>14 bytes</td> </tr> <tr> <td>1</td><td>1</td><td>1</td><td>0</td><td>15 bytes</td> </tr> <tr> <td>1</td><td>1</td><td>1</td><td>1</td><td>16 bytes</td> </tr> </tbody> </table> <p>Caution In the pointer mode (UBnITM bit = 1 in the UBnFIC0 register), the number of transmit data set as the trigger can be only 1 byte (UBnTT3 to UBnTT0 bits = 0000), and other settings are prohibited. If a setting of other than 1 byte is made, the operation is not guaranteed.</p>	UBn TT3	UBn TT2	UBn TT1	UBn TT0	Number of data of transmit FIFO set as trigger	Pointer mode	Pending mode	0	0	0	0	1 byte	Settable	Settable	0	0	0	1	2 bytes	Setting prohibited	0	0	1	0	3 bytes	0	0	1	1	4 bytes	0	1	0	0	5 bytes	0	1	0	1	6 bytes	0	1	1	0	7 bytes	0	1	1	1	8 bytes	1	0	0	0	9 bytes	1	0	0	1	10 bytes	1	0	1	0	11 bytes	1	0	1	1	12 bytes	1	1	0	0	13 bytes	1	1	0	1	14 bytes	1	1	1	0	15 bytes	1	1	1	1	16 bytes
UBn TT3	UBn TT2	UBn TT1	UBn TT0	Number of data of transmit FIFO set as trigger	Pointer mode	Pending mode																																																																																						
0	0	0	0	1 byte	Settable	Settable																																																																																						
0	0	0	1	2 bytes	Setting prohibited																																																																																							
0	0	1	0	3 bytes																																																																																								
0	0	1	1	4 bytes																																																																																								
0	1	0	0	5 bytes																																																																																								
0	1	0	1	6 bytes																																																																																								
0	1	1	0	7 bytes																																																																																								
0	1	1	1	8 bytes																																																																																								
1	0	0	0	9 bytes																																																																																								
1	0	0	1	10 bytes																																																																																								
1	0	1	0	11 bytes																																																																																								
1	0	1	1	12 bytes																																																																																								
1	1	0	0	13 bytes																																																																																								
1	1	0	1	14 bytes																																																																																								
1	1	1	0	15 bytes																																																																																								
1	1	1	1	16 bytes																																																																																								

Remark n = 0, 1

Bit position	Bit name	Function																																																																																										
3 to 0	UBnRT3 to UBnRT0	<p>Set the number of receive FIFO_n receive data to be the trigger. Each time data of the specified number has been stored from receive shift register n to receive FIFO_n, the UBTIR interrupt is generated. In the pending mode (UBnITM bit = 0 in the UBnFIC0 register), the UBTIR_n interrupt is generated under the conditions of the pending mode.</p> <table border="1"> <thead> <tr> <th>UBn RT3</th> <th>UBn RT2</th> <th>UBn RT1</th> <th>UBn RT0</th> <th>Number of data of receive FIFO_n set as trigger</th> <th>Pointer mode</th> <th>Pending mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1 byte</td> <td>Settable</td> <td rowspan="16">Settable</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>2 bytes</td> <td rowspan="15">Setting prohibited</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>3 bytes</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>4 bytes</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>5 bytes</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td>6 bytes</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>0</td> <td>7 bytes</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>8 bytes</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>9 bytes</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>1</td> <td>10 bytes</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>11 bytes</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1</td> <td>12 bytes</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>13 bytes</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>14 bytes</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>15 bytes</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>16 bytes</td> </tr> </tbody> </table> <p>Caution In the pointer mode (UBnIRM bit = 1 in the UBnFIC0 register), the number of receive data set as the trigger can be only 1 byte (UBnRT3 to UBnRT0 bits = 0000), and other settings are prohibited. If a setting of other than 1 byte is made, the operation is not guaranteed.</p>	UBn RT3	UBn RT2	UBn RT1	UBn RT0	Number of data of receive FIFO _n set as trigger	Pointer mode	Pending mode	0	0	0	0	1 byte	Settable	Settable	0	0	0	1	2 bytes	Setting prohibited	0	0	1	0	3 bytes	0	0	1	1	4 bytes	0	1	0	0	5 bytes	0	1	0	1	6 bytes	0	1	1	0	7 bytes	0	1	1	1	8 bytes	1	0	0	0	9 bytes	1	0	0	1	10 bytes	1	0	1	0	11 bytes	1	0	1	1	12 bytes	1	1	0	0	13 bytes	1	1	0	1	14 bytes	1	1	1	0	15 bytes	1	1	1	1	16 bytes
UBn RT3	UBn RT2	UBn RT1	UBn RT0	Number of data of receive FIFO _n set as trigger	Pointer mode	Pending mode																																																																																						
0	0	0	0	1 byte	Settable	Settable																																																																																						
0	0	0	1	2 bytes	Setting prohibited																																																																																							
0	0	1	0	3 bytes																																																																																								
0	0	1	1	4 bytes																																																																																								
0	1	0	0	5 bytes																																																																																								
0	1	0	1	6 bytes																																																																																								
0	1	1	0	7 bytes																																																																																								
0	1	1	1	8 bytes																																																																																								
1	0	0	0	9 bytes																																																																																								
1	0	0	1	10 bytes																																																																																								
1	0	1	0	11 bytes																																																																																								
1	0	1	1	12 bytes																																																																																								
1	1	0	0	13 bytes																																																																																								
1	1	0	1	14 bytes																																																																																								
1	1	1	0	15 bytes																																																																																								
1	1	1	1	16 bytes																																																																																								

Remark n = 0, 1

(9) UARTBn FIFO status register 0 (UBnFIS0) (n = 0, 1)

The UBnFIS0 register is valid in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register). It is used to read the number of bytes of the data stored in receive FIFO_n.

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

	7	6	5	4	3	2	1	0	Address	After reset
UB0FIS0	0	0	0	UB0RB4	UB0RB3	UB0RB2	UB0RB1	UB0RB0	FFFFFA0EH	00H
UB1FIS0	0	0	0	UB1RB4	UB1RB3	UB1RB2	UB1RB1	UB1RB0	FFFFFA2EH	00H

Bit position	Bit name	Function																																																																																																																		
4 to 0	UBnRB4 to UBnRB0	Indicates the number of bytes (readable bytes) of the data stored in receive FIFO _n as a receive FIFO _n pointer.																																																																																																																		
		<table border="1"> <thead> <tr> <th>UBn RB4</th> <th>UBn RB3</th> <th>UBn RB2</th> <th>UBn RB1</th> <th>UBn RB0</th> <th>Receive FIFO_n pointer</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0 bytes</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1 byte</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>2 bytes</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>3 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>4 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>5 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>6 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>7 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>8 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>9 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>10 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>11 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>12 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>13 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>14 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>15 bytes</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>16 bytes</td></tr> <tr> <td colspan="5">Other than above</td> <td>Invalid</td> </tr> </tbody> </table>	UBn RB4	UBn RB3	UBn RB2	UBn RB1	UBn RB0	Receive FIFO _n pointer	0	0	0	0	0	0 bytes	0	0	0	0	1	1 byte	0	0	0	1	0	2 bytes	0	0	0	1	1	3 bytes	0	0	1	0	0	4 bytes	0	0	1	0	1	5 bytes	0	0	1	1	0	6 bytes	0	0	1	1	1	7 bytes	0	1	0	0	0	8 bytes	0	1	0	0	1	9 bytes	0	1	0	1	0	10 bytes	0	1	0	1	1	11 bytes	0	1	1	0	0	12 bytes	0	1	1	0	1	13 bytes	0	1	1	1	0	14 bytes	0	1	1	1	1	15 bytes	1	0	0	0	0	16 bytes	Other than above					Invalid
UBn RB4	UBn RB3	UBn RB2	UBn RB1	UBn RB0	Receive FIFO _n pointer																																																																																																															
0	0	0	0	0	0 bytes																																																																																																															
0	0	0	0	1	1 byte																																																																																																															
0	0	0	1	0	2 bytes																																																																																																															
0	0	0	1	1	3 bytes																																																																																																															
0	0	1	0	0	4 bytes																																																																																																															
0	0	1	0	1	5 bytes																																																																																																															
0	0	1	1	0	6 bytes																																																																																																															
0	0	1	1	1	7 bytes																																																																																																															
0	1	0	0	0	8 bytes																																																																																																															
0	1	0	0	1	9 bytes																																																																																																															
0	1	0	1	0	10 bytes																																																																																																															
0	1	0	1	1	11 bytes																																																																																																															
0	1	1	0	0	12 bytes																																																																																																															
0	1	1	0	1	13 bytes																																																																																																															
0	1	1	1	0	14 bytes																																																																																																															
0	1	1	1	1	15 bytes																																																																																																															
1	0	0	0	0	16 bytes																																																																																																															
Other than above					Invalid																																																																																																															

Remark n = 0, 1

(10)UARTBn FIFO status register 1 (UBnFIS1) (n = 0, 1)

The UBnFIS1 register is valid in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register). This register can be used to read the number of vacant bytes of transmit FIFO.

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

Caution The values of the UBnTB4 to UBnTB0 bits are reflected after transmit data has been written to the UBnTX register and then time of two cycles of $f_x/4$ (f_x : main clock) has passed. Therefore, care must be exercised when referencing the UBnFIS1 register after transmit data has been written to the UBnTX register.

	7	6	5	4	3	2	1	0	Address	After reset
UB0FIS1	0	0	0	UB0TB4	UB0TB3	UB0TB2	UB0TB1	UB0TB0	FFFFFA0FH	10H
UB1FIS1	0	0	0	UB1TB4	UB1TB3	UB1TB2	UB1TB1	UB1TB0	FFFFFA2FH	10H

Bit position	Bit name	Function																																																																																																																		
4 to 0	UBnTB4 to UBnTB0	Indicates the number of vacant bytes of transmit FIFO (bytes that can be written) as a transmit FIFO pointer. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>UBn TB4</th> <th>UBn TB3</th> <th>UBn TB2</th> <th>UBn TB1</th> <th>UBn TB0</th> <th>Transmit FIFO pointer</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0 bytes</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1 byte</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>2 bytes</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>3 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>4 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>5 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>6 bytes</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>7 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>8 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>9 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>10 bytes</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>11 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>12 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>13 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>14 bytes</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>15 bytes</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>16 bytes</td></tr> <tr> <td colspan="5" style="text-align: center;">Other than above</td> <td>Invalid</td> </tr> </tbody> </table>	UBn TB4	UBn TB3	UBn TB2	UBn TB1	UBn TB0	Transmit FIFO pointer	0	0	0	0	0	0 bytes	0	0	0	0	1	1 byte	0	0	0	1	0	2 bytes	0	0	0	1	1	3 bytes	0	0	1	0	0	4 bytes	0	0	1	0	1	5 bytes	0	0	1	1	0	6 bytes	0	0	1	1	1	7 bytes	0	1	0	0	0	8 bytes	0	1	0	0	1	9 bytes	0	1	0	1	0	10 bytes	0	1	0	1	1	11 bytes	0	1	1	0	0	12 bytes	0	1	1	0	1	13 bytes	0	1	1	1	0	14 bytes	0	1	1	1	1	15 bytes	1	0	0	0	0	16 bytes	Other than above					Invalid
UBn TB4	UBn TB3	UBn TB2	UBn TB1	UBn TB0	Transmit FIFO pointer																																																																																																															
0	0	0	0	0	0 bytes																																																																																																															
0	0	0	0	1	1 byte																																																																																																															
0	0	0	1	0	2 bytes																																																																																																															
0	0	0	1	1	3 bytes																																																																																																															
0	0	1	0	0	4 bytes																																																																																																															
0	0	1	0	1	5 bytes																																																																																																															
0	0	1	1	0	6 bytes																																																																																																															
0	0	1	1	1	7 bytes																																																																																																															
0	1	0	0	0	8 bytes																																																																																																															
0	1	0	0	1	9 bytes																																																																																																															
0	1	0	1	0	10 bytes																																																																																																															
0	1	0	1	1	11 bytes																																																																																																															
0	1	1	0	0	12 bytes																																																																																																															
0	1	1	0	1	13 bytes																																																																																																															
0	1	1	1	0	14 bytes																																																																																																															
0	1	1	1	1	15 bytes																																																																																																															
1	0	0	0	0	16 bytes																																																																																																															
Other than above					Invalid																																																																																																															

Remark n = 0, 1

10.2.4 Interrupt requests

The following five types of interrupt requests are generated from UARTBn (n = 0, 1).

- Reception error interrupt (UBTIREn)
- Reception completion interrupt (UBTIRn)
- Transmission completion interrupt (UBTITn)
- FIFO transmission completion interrupt (UBTIFn)
- Reception timeout interrupt (UBTITOn)

The default priorities among these five types of interrupt requests is, from high to low, reception error interrupt, reception completion interrupt, transmission completion interrupt, FIFO transmission completion interrupt, and reception timeout interrupt.

Table 10-2. Generated Interrupts and Default Priorities

Interrupt	Priority
Reception error	1
Reception completion	2
Transmission completion	3
FIFO transmission completion	4
Reception timeout	5

(1) Reception error interrupt (UBTIREn)

(a) Single mode

When reception is enabled, a reception error interrupt is generated according to the logical OR of the three types of reception errors (parity error, framing error, overrun error) explained for the UBnSTR register.

When reception is disabled, no reception error interrupt is generated.

(b) FIFO mode

When reception is enabled, a reception error interrupt is generated according to the logical OR of the three types of reception errors (parity error, framing error, overflow error) explained for the UBnSTR register.

When reception is disabled, no reception error interrupt is generated.

(2) Reception completion interrupt (UBTIRn)**(a) Single mode**

When reception is enabled, a reception completion interrupt is generated if data is shifted into receive shift register n and stored in the UBnRX register (if the receive data can be read).

When reception is disabled, no reception completion interrupt is generated.

(b) FIFO mode

When reception is enabled, a reception completion interrupt is generated if data is shifted into receive shift register n and receive data of the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register is transferred to receive FIFO n (if receive data of the specified number can be read).

When reception is disabled, no reception completion interrupt is generated.

(3) Transmission completion interrupt (UBTITn)**(a) Single mode**

The transmission completion interrupt is generated if transmit data of one frame, including 7 or 8 bits of characters, is shifted out from transmit shift register n and the UBnTX register becomes vacant (if transmit data can be written).

(b) FIFO mode

The transmission completion interrupt is generated if transmit data of the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register is transferred to transmit shift register n from transmit FIFO n (if transmit data of the specified number can be written).

(4) FIFO transmission completion interrupt (UBTIFn)**(a) Single mode**

Cannot be used.

(b) FIFO mode

The FIFO transmission completion interrupt is generated when no more data is in transmit FIFO n and transmit shift register n (when the FIFO and register become empty). After the FIFO transmission completion interrupt has occurred, clear the interrupt held pending (UBTITn) in the pending mode (UBnITM bit = 0 in the UBnFIC0 register) by clearing the FIFO (UBnTFC bit = 1 in the UBnFIC0 register).

Caution If the FIFO transmission completion interrupt is generated (all transmit data are not transmitted) because writing the next transmit data to transmit FIFO n is delayed, do not clear the FIFO.

(5) Reception timeout interrupt (UBTITOn)**(a) Single mode**

Cannot be used.

(b) FIFO mode

The reception timeout interrupt is generated if data is stored in receive FIFO when the next data does not come (start bit is not detected) even after the next data reception wait time specified by the UBnTC4 to UBnTC0 bits of the UBnFIC1 register has elapsed, when the timeout counter function is used (UBnTCE bit = 1 in the UBnFIC1 register).

The reception timeout interrupt is not generated while reception is disabled.

If receive data of the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register is not received, the timing of reading the number of receive data less than the specified number can be set by the reception timeout interrupt.

★

Since the timeout counter starts counting at start bit detection, a receive timeout interrupt does not occur if data of 1 character has not been received.

10.2.5 Control method

(1) Single mode/FIFO mode

The single mode or FIFO mode can be selected by using the UBnMOD bit of the UBnFIC0 register.

(a) Single mode

- Each of the UBnRX and UBnTX registers consists of 8 bits \times 1 stage.
- When 1 byte of data is received, the UBTIRn interrupt is generated.
- If the next reception operation of UARTBn is completed before the receive data of the UBnRX register is read after the UBTIRn interrupt has been generated, the UBTIREn interrupt is generated and an overrun error occurs.

(b) FIFO mode

- Receive FIFO (UBnRXAP register) consists of 16 bits \times 16 stages and transmit FIFO consists of 8 bits \times 16 stages.
- Receive FIFO can recognize error data by reading the 16-bit UBnRXAP register only when a reception error (parity error or framing error) occurs.
- Transmission is started when transmission is enabled (UBnTXE bit = 1 in the UBnCTL0 register) after transmit data of at least the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register and 16 bytes or less are written to transmit FIFO.
- The pending mode or pointer mode can be selected for the generation timing of the UBTITn and UBTIRn interrupts.

(2) Pending mode/pointer mode

The pending mode or pointer mode can be selected by using the UBnITM and UBnIRM bits of the UBnFIC0 register in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register).

★

If transmission is started by writing data of more than double the amount set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register to transmit FIFO, the transmission completion interrupt (UBTITn) may occur more than once. The reception completion interrupt (UBTIRn) may also occur more than once if the number of receive data set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register is 8 bytes or less in receive FIFO. In the pending or pointer mode, it can be specified how an interrupt is handled after it has been held pending.

(a) Pending mode**(i) During transmission (writing to transmit FIFO)**

- If the data of the first transmission completion interrupt (UBTITn) is not written to transmit FIFO after the interrupt has occurred, the second UBTITn interrupt does not occur (is held pending) even if the generation condition of the second UBTITn interrupt is satisfied (when transmit data of the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register is transferred from transmit FIFO to the transmit shift register).

When data for the first UBTITn interrupt is later written to transmit FIFO, the pending UBTITn interrupt is generated^{Note}.

★

Note The number of pending interrupts is as follows.

When trigger is set to 1 byte (UBnTT3 to UBnTT0 bits of UBnFIC2 register = 0000): 15 times max.

When trigger is set to 2 bytes (UBnTT3 to UBnTT0 bits of UBnFIC2 register = 0001): 7 times max.

:

When trigger is set to 6 bytes (UBnTT3 to UBnTT0 bits of UBnFIC2 register = 0101): 1 time max.

When trigger is set to 7 bytes (UBnTT3 to UBnTT0 bits of UBnFIC2 register = 0110): 1 time max.

When trigger is set to 8 bytes (UBnTT3 to UBnTT0 bits of UBnFIC2 register = 0111): 1 time max.

- In the pending mode, transmit data of the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register is always written to transmit FIFO when the transmission completion interrupt (UBTITn) occurs. Writing data to transmit FIFO is prohibited if the data is more or less than the specified number. If data more or less than the specified number is written, the operation is not guaranteed.
- To write transmit data to transmit FIFO by DMA, fix the UBnTT3 to UBnTT0 bits of the UBnFIC2 register to 0000 (set number of transmit data: 1 byte). If any other setting is made, the operation is not guaranteed.

(ii) During reception (reading from receive FIFO)

- If data for the first reception completion interrupt (UBTIRn) is not read from receive FIFO, the second UBTIRn interrupt does not occur (is held pending) even if the generation condition of the second UBTIRn is satisfied (if receive data of the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register can be read from receive FIFO). When data for the first UBTIRn interrupt is later read from the receive FIFO, the pending UBTIRn interrupt is generated^{Note}.

★

Note The number of pending interrupts is as follows.

When trigger is set to 1 byte (UBnRT3 to UBnRT0 bits of UBnFIC2 register = 0000): 15 times max.

When trigger is set to 2 bytes (UBnRT3 to UBnRT0 bits of UBnFIC2 register = 0001): 7 times max.

:

When trigger is set to 6 bytes (UBnRT3 to UBnRT0 bits of UBnFIC2 register = 0101): 1 time max.

When trigger is set to 7 bytes (UBnRT3 to UBnRT0 bits of UBnFIC2 register = 0110): 1 time max.

When trigger is set to 8 bytes (UBnRT3 to UBnRT0 bits of UBnFIC2 register = 0111): 1 time max.

- In the pending mode, receive data of the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register is always read from receive FIFO when the reception completion interrupt (UBTIRn) occurs. Reading data from receive FIFO is prohibited if the data is more or less than the specified number. If data more or less than the specified number is read, the operation is not guaranteed.
- To read receive data from receive FIFO by DMA, fix the UBnRT3 to UBnRT0 bits of the UBnFIC2 register to 0000 (set number of receive data: 1 byte). If any other setting is made, the operation is not guaranteed.

(b) Pointer mode**(i) During transmission (writing to transmit FIFO)**

- Each time the data of 1 byte is transferred to transmit shift register n from transmit FIFO, a transmission completion interrupt (UBTITn) occurs.
- In the pointer mode, be sure to fix the UBnTT3 to UBnTT0 bits of the UBnFIC2 register to 0000 (set number of transmit data: 1 byte) as the number of transmit data set as the trigger for transmit FIFO when the transmission completion interrupt (UBTITn) occurs. If any other setting is made, the operation is not guaranteed.
- Writing transmit data to transmit FIFO by DMA is prohibited. The operation is not guaranteed if DMA control is used.
- After the transmission completion interrupt (UBTITn) has been acknowledged, data of the number of vacant bytes of transmit FIFO can be written to transmit FIFO by referencing the UBnFIS1 register.

(ii) During reception (reading from receive FIFO)

- Each time the data of 1 byte is transferred to receive FIFO from receive shift register n, a reception completion interrupt (UBTIRn) occurs.
- In the pointer mode, be sure to fix the UBnRT3 to UBnRT0 bits of the UBnFIC2 register to 0000 (set number of receive data: 1 byte) as the number of receive data set as the trigger for receive FIFO when the reception completion interrupt (UBTIRn) occurs. If any other setting is made, the operation is not guaranteed.
- Reading receive data from receive FIFO by DMA is prohibited. The operation is not guaranteed if DMA control is used.
- After the reception completion interrupt (UBTIRn) has been acknowledged, data of the number of bytes stored in receive FIFO can be read from receive FIFO by referencing the UBnFIS0 register. In some cases, however, data is not stored in receive FIFO even though the UBTIRn interrupt is generated (UBnRB4 to UBnRB0 bits = 00000 in the UBnFIS0 register). In these cases, do not read data from receive FIFO. Always read data from receive FIFO when the number of bytes stored in receive FIFO is 1 byte or more (UBnRB4 to UBnRB0 bits = other than 00000).

10.2.6 Operation**(1) Data format**

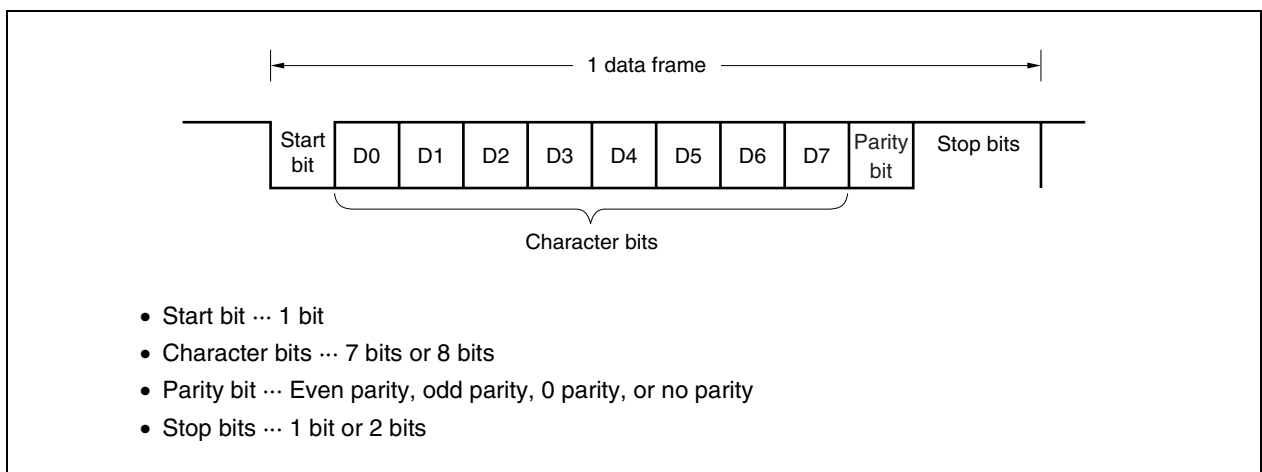
Full-duplex serial data transmission and reception can be performed.

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 10-2.

The character bit length within one data frame, the type of parity, and the stop bit length are specified by the UARTBn control register 0 (UBnCTL0) (n = 0, 1).

Also, data is transferred with LSB first/MSB first.

Figure 10-2. Asynchronous Serial Interface Transmit/Receive Data Format (LSB-First Transfer)



(2) Transmit operation

In the single mode (UBnMOD bit = 0 in the UBnFIC0 register), transmission is enabled when the UBnTXE bit of the UBnCTL0 register is set to 1, and transmission is started when transmit data is written to the UBnTX register (n = 0, 1).

In the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register), transmission is started when transmit data of at least the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register and 16 bytes or less is written to transmit FIFO_n and then the UBnTXE bit is set to 1.

Caution Setting the UBnTXE bit of the UBnCTL0 register to 1 before writing transmit data to transmit FIFO_n in the FIFO mode is prohibited. The operation is not guaranteed if this setting is made.

(a) Transmission enabled state

This state is set by the UBnTXE bit in the UBnCTL0 register (n = 0, 1).

- UBnTXE = 1: Transmission enabled state
- UBnTXE = 0: Transmission disabled state

However, when the transmission enabled state is set, to use UARTB0, which shares pins with clocked serial interface 30 (CSI30), the CSICAE0 bit of clocked serial interface mode register 30 (CSIM30) should be cleared to 0.

Since UARTB_n does not have a CTS (transmission enabled signal) input pin, a port should be used to confirm whether the destination is in the reception enabled state.

(b) Starting a transmit operation

- **In single mode (UBnMOD bit = 0 in UBnFIC0 register)**

In the single mode, transmission is started when transmit data is written to the UBnTX register while transmission is enabled (n = 0, 1).

- **In FIFO mode (UBnMOD bit = 1 in UBnFIC0 register)**

In the FIFO mode, transmission is started when transmit data of at least the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register and 16 bytes or less is written to transmit FIFO_n and then transmission is enabled (UBnTXE bit = 1).

Data in transmit data register n (UBnTX register in single mode or transmit FIFO_n in the FIFO mode) is transferred to transmit shift register n when transmission is started. Then, transmit shift register n outputs data to the TXD_n 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**(i) Transmission completion interrupt (UBTITn)**

- **In single mode (UBnMOD bit = 0 in UBnFIC0 register)**

In the single mode, the transmission completion interrupt (UBTITn) occurs when transmit data can be written to the UBnTX register (when 1 byte of data is transferred from the UBnTX register to transmit shift register n) (n = 0, 1).

- **In FIFO mode (UBnMOD bit = 1 in UBnFIC0 register)**

In the FIFO mode, the UBTITn interrupt occurs when transmit data of the number set as the trigger specified by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register is transferred from transmit FIFO n to transmit shift register n (if transmit data of the number set as the trigger can be written) (n = 0, 1).

- **If pending mode is specified (UBnITM bit = 0 in UBnFIC0 register) in FIFO mode**

If the pending mode is specified in the FIFO mode, the second UBTITn interrupt is held pending after the first UBTITn interrupt has occurred, until as many transmit data as the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register are written to transmit FIFO n, even if the generation condition of the second UBTITn interrupt is satisfied. When as many transmit data as the number set as the trigger are written to transmit FIFO n in response to the first UBTITn interrupt, the second pending UBTITn interrupt is generated (n = 0, 1).

- **If pointer mode is specified (UBnITM bit = 1 in UBnFIC0 register) in FIFO mode**

If the pointer mode is specified in the FIFO mode, the second UBTITn interrupt occurs when the generation condition of the second UBTITn interrupt is satisfied even if as many transmit data as the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register are not written to transmit FIFO n when the first UBTITn interrupt occurs (n = 0, 1).

(ii) FIFO transmission completion interrupt (UBTIFn)

The FIFO transmission completion interrupt (UBTIFn) occurs when no more data is in transmit FIFO n and transmit shift register n in the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register) (n = 0, 1). After the UBTIFn interrupt has occurred, clear the pending UBTITn interrupt in the pending mode (UBnITM bit = 0 in the UBnFIC0 register) by clearing the FIFO (UBnTFC bit = 1 in the UBnFIC0 register). If the UBTIFn interrupt occurs because writing the next transmit data to transmit FIFO n is delayed (if all transmit data have not been transmitted), do not clear the FIFO.

If the data to be transmitted next has not been written to transmit data register n, the transmit operation is suspended.

Caution In the single mode, the transmission completion interrupt (UBTITn) occurs when the UBnTX register becomes empty (when 1 byte of data is transferred from the UBnTX register to transmit shift register n). In the FIFO mode, the FIFO transmission completion interrupt (UBTIFn) occurs when data is no longer in transmit FIFO n and transmit shift register n (when the FIFO and register are vacant). However, the UBTITn interrupt or UBTIFn interrupt is not generated if transmit data register n becomes empty due to $\overline{\text{RESET}}$ input.

Figure 10-3. Timing of Asynchronous Serial Interface Transmission Completion Interrupt (UBTITn)

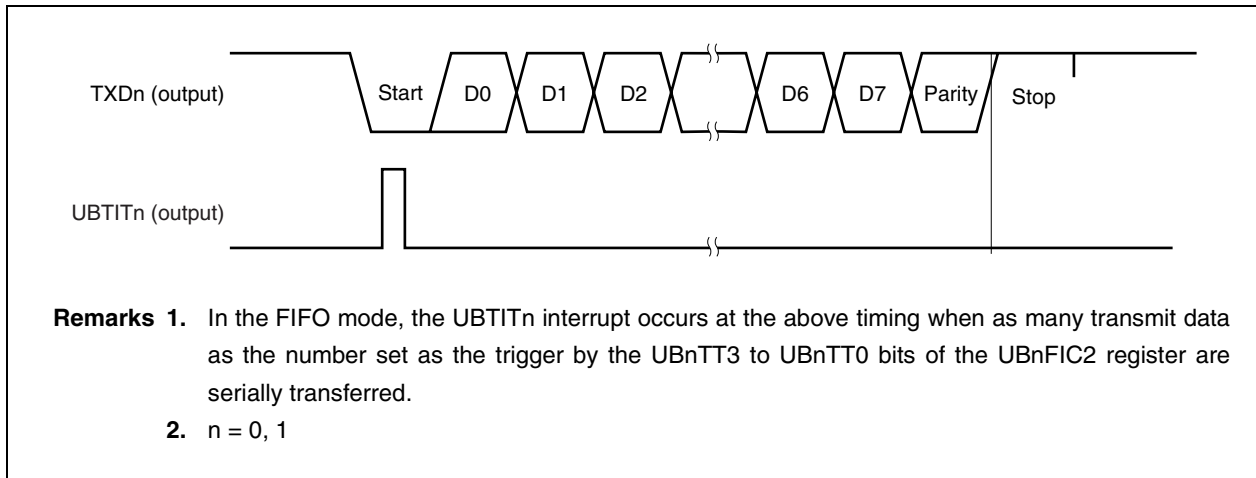
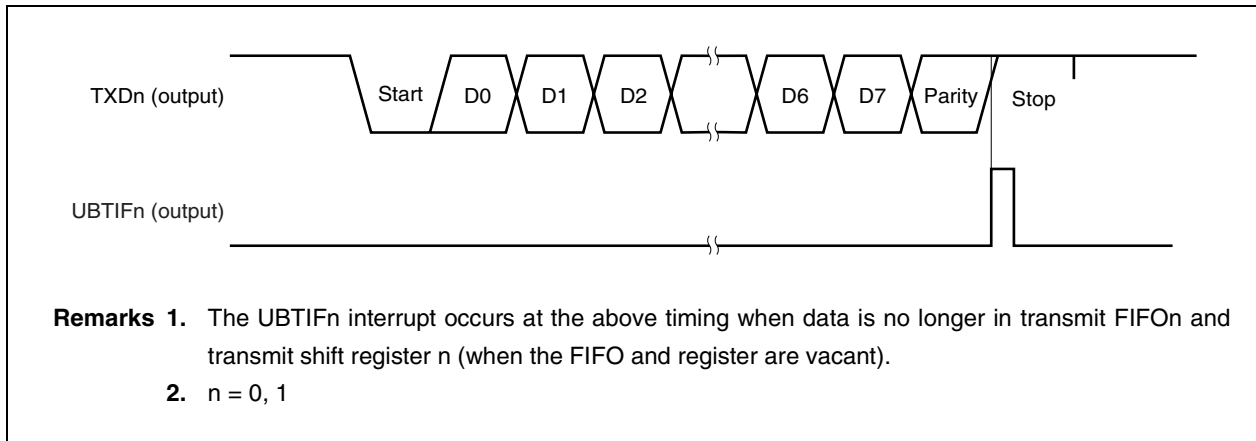


Figure 10-4. Timing of Asynchronous Serial Interface FIFO Transmission Completion Interrupt (UBTIFn)



(3) Continuous transmission operation

- **In single mode (UBnMOD bit = 0 in UBnFIC0 register)**

In the single mode, the next data can be written to the UBnTX register as soon as transmit shift register n has started a shift operation (n = 0, 1). The timing of transfer can be identified by the transmission completion interrupt (UBTITn). By writing the next transmit data to the UBnTX register via the UBTITn interrupt within one data frame transmission period, data can be transmitted without an interval and an efficient communication rate can be realized.

Caution Before executing initialization during transmission processing, confirm that the UBnTSF bit of the UBnSTR register is 0. If initialization is executed while the UBnTSF bit is 1, the transmit data is not guaranteed.

- **If pending mode is specified (UBnITM bit = 0 in UBnFIC0 register) in FIFO mode**

If transmit data of at least the number set as the transmit trigger by UBnTT3 to UBnTT0 bits of the UBnFIC2 register and 16 bytes or less is written to transmit FIFO, transmission starts.

If the pending mode is specified in the FIFO mode, as many of the next transmit data as the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register can be written to transmit FIFO as soon as transmit shift register n has started shifting the last data of the specified number of data. The timing of transfer can be identified by the UBTITn interrupt. By writing as many of the next transmit data as the number set as the trigger to transmit FIFO or writing the data to the FIFO within the transmission period of the data in transmit FIFO via the UBTITn interrupt, data can be transmitted without an interval and an efficient communication rate can be realized.

Caution Before executing initialization during transmission processing, confirm that the UBnTSF bit of the UBnSTR register is 0 (this can also be done by the FIFO transmission completion interrupt (UBTIFn)). If initialization is executed while the UBnTSF bit is 1, the transmit data is not guaranteed. To write transmit data to transmit FIFO by DMA, set the number of transmit data specified as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register to 1 byte. Otherwise, the operation will not be guaranteed.

- **If pointer mode is specified (UBnITM bit = 1 in UBnFIC0 register) in FIFO mode**

If the pointer mode is specified in the FIFO mode, a UBTITn interrupt occurs and the next data can be written to transmit FIFO as soon as transmit shift register n has started shifting the number of transmit data set as the trigger. At this time, as many data as the number of vacant bytes of transmit FIFO can be written by referencing the UBnFIS1 register. The timing of transfer can be identified by the UBTITn interrupt. By writing as many of the next transmit data as the number specified as the trigger to transmit FIFO (= 1 byte) or writing the data to the FIFO within the transmission period of the data in transmit FIFO via the UBTITn interrupt, data can be transmitted without an interval and an efficient communication rate can be realized.

Caution Before executing initialization during transmission processing, confirm that the UBnTSF bit of the UBnSTR register is 0 (this can also be done by the FIFO transmission completion interrupt (UBTIFn)). If initialization is executed while the UBnTSF bit is 1, the transmit data is not guaranteed.

(4) Receive operation

The awaiting reception state is set by setting the UBnPWR bit to 1 in the UBnCTL0 register and then setting UBnRXE to 1 in the UBnCTL0 register (n = 0, 1). 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 receive shift register n according to the baud rate that was set.

In the single mode (UBnMOD bit = 0 in the UBnFIC0 register), a reception completion interrupt (UBTIRn) is generated each time the reception of one frame of data is completed. Normally, the receive data is transferred from the UBnRX register to memory by this interrupt servicing.

In the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register), the UBTIRn interrupt occurs when as many receive data as the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register are transferred to receive FIFO n.

If the pending mode is specified (UBnIRM bit = 0 in the UBnFIC0 register) in the FIFO mode, as many receive data as the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register can be read from receive FIFO n.

If the pointer mode is specified (UBnIRM bit = 1 in the UBnFIC0 register) in the FIFO mode, as many data as the number of bytes stored in receive FIFO n (0 bytes or more) can be read from receive FIFO n by referencing the number of receive data specified as the trigger by the UBnRT3 to UBnRT0 bits (1 byte) or the UBnFIS0 register.

Caution If the pointer mode is specified in the FIFO mode and if as many data as the number of bytes stored in receive FIFO n are read by referencing the UBnFIS0 register, no data may be stored in receive FIFO n (UBnRB4 to UBnRB0 bits = 00000 in the UBnFIS0 register) even though the reception completion interrupt (UBTIRn) has occurred. In this case, do not read data from receive FIFO n. Be sure to read data from receive FIFO n after confirming that the number of bytes stored in receive FIFO n = 1 byte or more (UBnRB4 to UBnRB0 bits = other than 00000).

(a) Reception enabled state

This state is set by the UBnRXE bit in the UBnCTL0 register (n = 0, 1).

- UBnRXE = 1: Reception enabled state
- UBnRXE = 0: Reception disabled state

However, when the reception enabled state is set, to use UARTB0, which shares pins with clocked serial interface 30 (CSI30), the operation of CSI30 must be disabled by clearing the CSICAE0 bit of clocked serial interface mode register 30 (CSIM30) to 0 (n = 0, 1).

In the reception disabled state, the reception hardware stands by in the initial state. At this time, the reception completion interrupt or reception error interrupt does not occur, and the contents of receive data register n (UBnRX register in the single mode or receive FIFO n in the FIFO mode (UBnRXAP register)) are retained.

(b) Starting a receive operation

A receive operation is started by the detection of a start bit.

The RXDn pin is sampled using the serial clock from UARTBn control register 2 (UBnCTL2) (n = 0, 1).

(c) Reception completion interrupt**(i) Reception completion interrupt (UBTIRn)****• In single mode (UBnMOD bit = 0 in UBnFIC0 register)**

When UBnRXE bit = 1 in the UBnCTL0 register and the reception of one frame of data is completed (the stop bit is detected) in the single mode, a reception completion interrupt request (UBTIRn) is generated and the receive data in receive shift register n is transferred to the UBnRX register at the same time (n = 0, 1).

Also, if an overrun error occurs, the receive data at that time is not transferred to the UBnRX register, and a reception error interrupt (UBTIREn) is generated.

If a parity error or framing error occurs during the reception operation, the reception operation continues up to the position at which the stop bit is received. After completion of reception, an UBTIREn interrupt occurs (the receive data in receive shift register n is transferred to the UBnRX register).

If the UBnRXE bit is reset (0) during a receive operation, the receive operation is immediately stopped. At this time, the contents of the UBnRX register remain unchanged, the contents of the UARTBn status register (UBnSTR) are cleared, and the UBTIRn and UBTIREn interrupts do not occur.

No UBTIRn interrupt is generated when UBnRXE = 0 (reception is disabled).

• In FIFO mode (UBnMOD bit = 1 in UBnFIC0 register)

In the FIFO mode, the reception completion interrupt (UBTIRn) occurs when data of one frame has been received (stop bit is detected) and when as many receive data as the number specified as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register are transferred from receive shift register n to receive FIFO n (n = 0, 1). If an overflow error occurs, the receive data is not transferred to receive FIFO n and the reception error interrupt (UBTIREn) occurs.

If a parity error or framing error occurs during reception, reception continues up to the reception position of the stop bit. After reception has been completed, the UBTIREn interrupt occurs and the receive data in receive shift register n is transferred to receive FIFO n. At this time, error information is appended as the UBnPEF or UBnFEF bit = 1 in the UBnRXAP register. If the UBTIREn interrupt occurs, the error data can be recognized by reading receive FIFO n as a 16-bit register, UBnRXAP.

(ii) Reception timeout interrupt (UBTITOn) (in FIFO mode only)

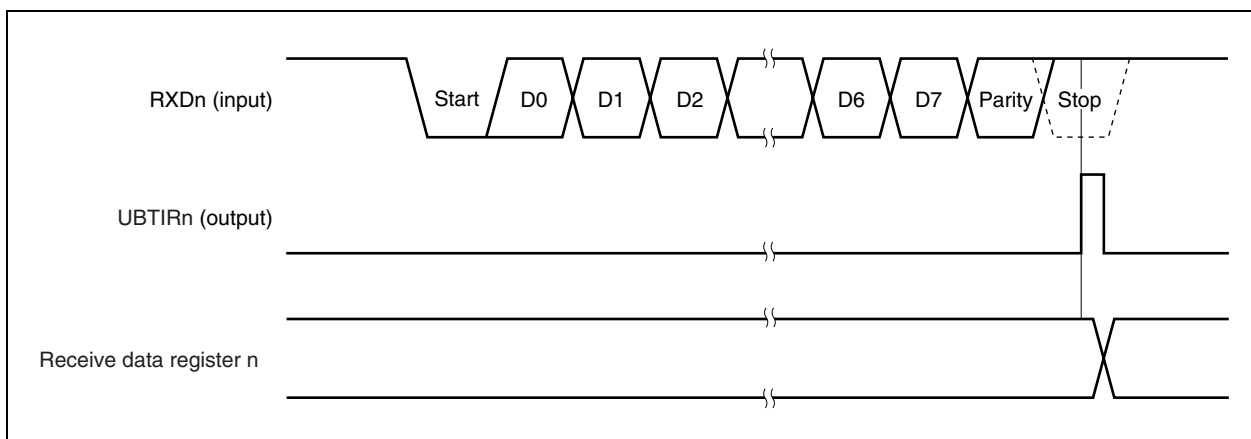
When the timeout counter function (UBnTCE bit = 1 in the UBnFIC1 register) is used in the FIFO mode, the reception timeout interrupt (UBTITOn) occurs if the next data does not come even after the next data reception wait time specified by the UBnTC4 to UBnTC0 bits of the UBnFIC1 register has elapsed and if data is stored in receive FIFO_n ($n = 0, 1$).

The UBTITOn interrupt does not occur while reception is disabled.

If as many receive data as the number set as the trigger by the UBnRT3 to UBnRT0 bits of the UBnFIC2 register are not received, the timing of reading less receive data than the specified number can be set by the UBTITOn interrupt.

- ★ Since the timeout counter starts counting at start bit detection, a receive timeout interrupt does not occur if data of 1 character has not been received.

Figure 10-5. Timing of Asynchronous Serial Interface Reception Completion Interrupt (UBTIRn)



Cautions 1. Be sure to read all the data (the number of data indicated by the UBnRB4 to UBnRB0 bits of the UBnFIS0 register) stored in receive data register n (UBnRX register in the single mode or receive FIFO_n in the FIFO mode (UBnRXAP register)) even when a reception error occurs.

Unless receive data register n is read, an overrun error occurs when the next data is received, causing the reception error status to persist.

If the pending mode is specified in the FIFO mode, however, be sure to clear the FIFO (UBnRFC bit = 1 in the UBnFIC0 register) after reading the data stored in receive FIFO_n.

In the FIFO mode, the FIFO can be cleared even without reading the data stored in receive FIFO_n.

If a parity error or framing error occurs in the FIFO mode, the UBnRXAP register can be read in 16-bit (halfword) units.

- 2.** Data is always received with one stop bit.
A second stop bit is ignored.

Remark $n = 0, 1$

(5) Reception error

In the single mode (UBnMOD bit = 0 in the UBnFIC0 register), the three types of errors that can occur during a receive operation are a parity error, framing error, and overrun error. In the FIFO mode (UBnMOD bit = 1 in the UBnFIC0 register), the three types of errors that can occur during a receive operation are a parity error, framing error, and overflow error.

As a result of data reception, the UBnPE, UBnFE, or UBnOVE bit of the UBnSTR register is set to 1 if a parity error, framing error, or overrun error occurs in the single mode. The UBnOVF bit of the UBnSTR register is set to 1 if an overflow error occurs in the FIFO mode. The UBnPEF or UBnFEF bit of the UBnRXAP register is set to 1 if a parity error or framing error occurs in the FIFO mode. At the same time, a reception error interrupt (UBTIREn) occurs. The contents of the error can be detected by reading the contents of the UBnSTR or UBnRXAP register.

The contents of the UBnSTR register are reset when 0 is written to the UBnOVF, UBnPE, UBnFE, or UBnOVE bit, or the UBnPWR or UBnRXE bit of the UBnCTL0 register. The contents of the UBnRXAP register are reset when 0 is written to the UBnPWR bit of the UBnCTL0 register.

Table 10-3. Reception Error Causes

Error Flag	Valid Operation Mode	Error Flag	Reception Error	Cause
UBnPE	Single mode	UBnPE	Parity error	The parity specification during transmission does not match the parity of the receive data
UBnFE		UBnFE	Framing error	No stop bit detected
UBnOVE		UBnOVE	Overrun error	The reception of the next data is completed before data is read from the UBnRX register
UBnOVF	FIFO mode	UBnOVF	Overflow error	The reception of the next data is completed while receive FIFO is full and before data is read.
UBnPEF		UBnPEF	Parity error	The parity specification during transmission does not match the parity of the data to be received.
UBnFEF		UBnFEF	Framing error	The stop bit is not detected when the target data is loaded.

Remark n = 0, 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 input clock ($f_x/4$) (f_x : main 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 (see **Figure 10-7**).

Also, since the circuit is configured as shown in Figure 10-6, internal processing during a receive operation is delayed by up to 2 clocks according to the external signal status.

Figure 10-6. Noise Filter Circuit

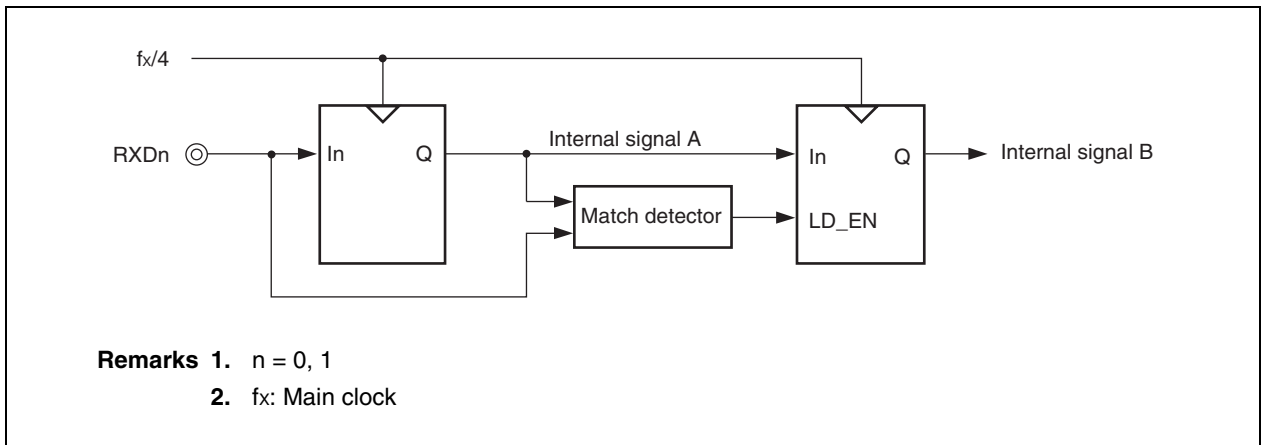
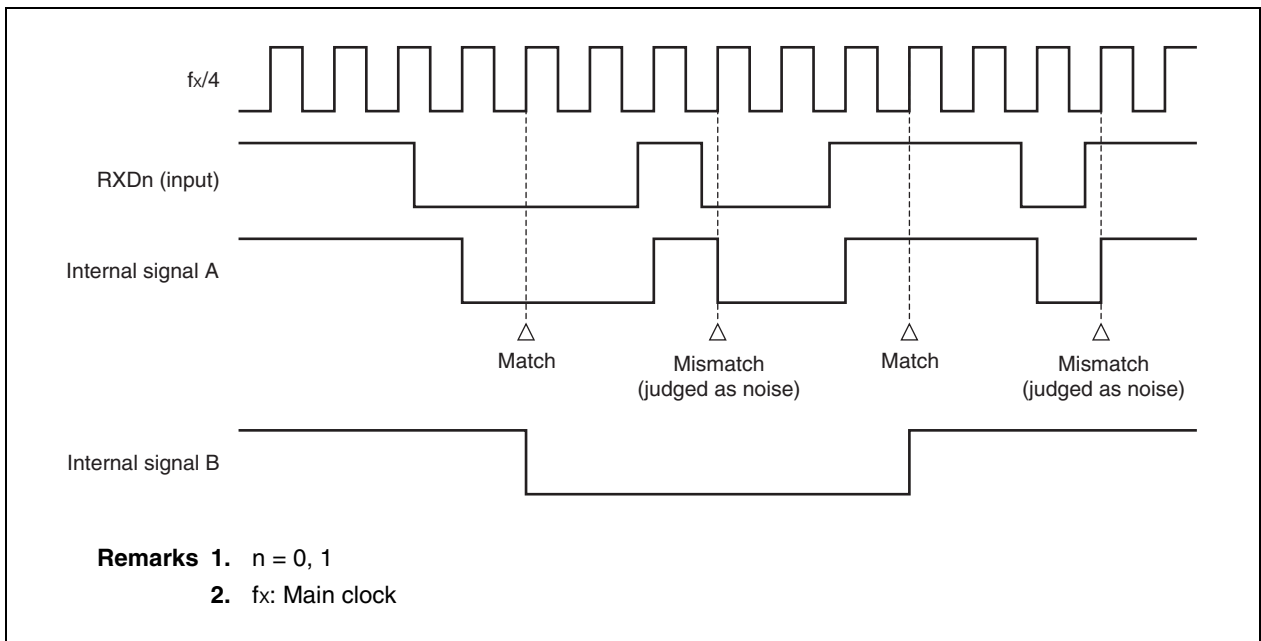


Figure 10-7. Timing of RXDn Signal Judged as Noise



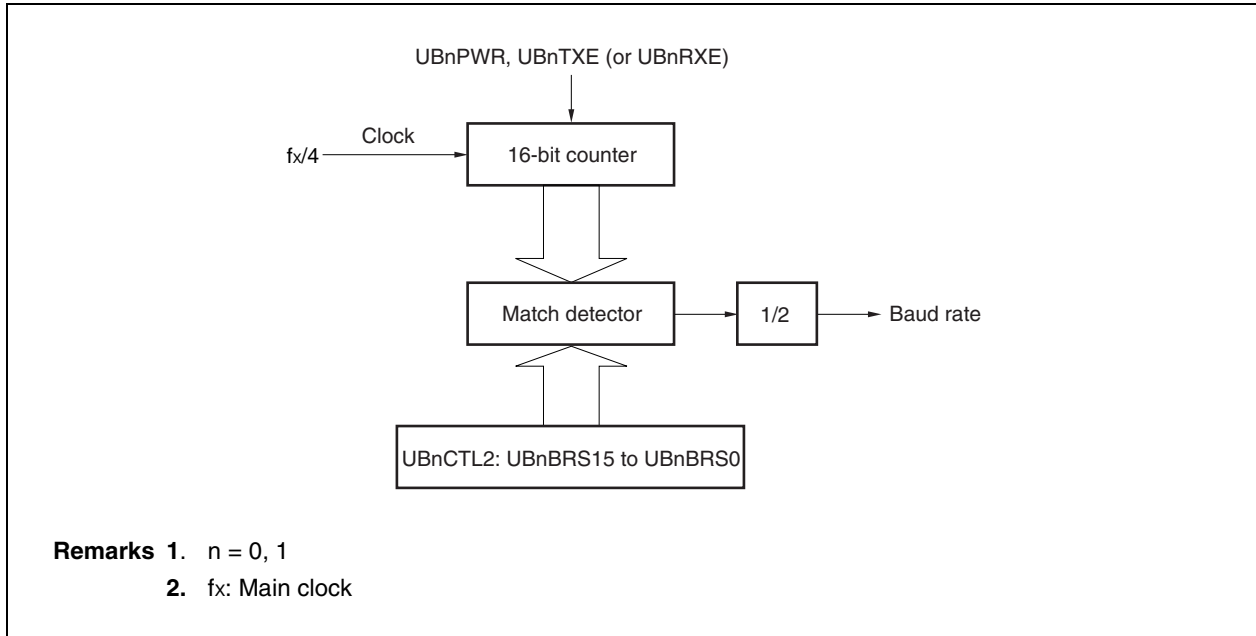
10.2.7 Dedicated baud rate generators 0, 1 (BRG0, BRG1)

A dedicated baud rate generator, which consists of a 16-bit programmable counter, generates serial clocks during transmission/reception in UARTBn. The dedicated baud rate generator output can be selected as the serial clock for each channel.

Separate 16-bit counters exist for transmission and for reception. The baud rate for transmission/reception is the same at the same channel.

(1) Baud rate generator configuration

Figure 10-8. Baud Rate Generator Configuration



(a) Base clock (Clock)

When UBnPWR bit = 1 in the UBnCTL0 register, the input clock ($fx/4$) is supplied to the transmission/reception unit (fx : main clock). This clock is called the base clock. When UBnPWR = 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 UBnCTL2 register (n = 0, 1).

The 16-bit counter divisor value can be selected according to the UBnBRS15 to UBnBRS0 bits of the UBnCTL2 register.

(a) Baud rate

The baud rate is the value obtained according to the following formula.

$$\text{Baud rate} = \frac{\text{Base clock frequency}}{2 \times k} \text{ [bps]}$$

Base clock frequency = $f_x/4$ (f_x : main clock)

k = Value set according to UBnBRS15 to UBnBRS0 bits of UBnCTL2 register (k = 4, 5, 6, ..., 65,535)

(b) Baud rate error

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: Main clock (f_x) = 150 MHz = 150,000,000 Hz

Settings of UBnBRS15 to UBnBRS0 bits in UBnCTL2 register = 000000001111010B

(k = 122)

Target baud rate = 153,600 bps

Baud rate = $150 \text{ M}/4/(2 \times 122)$
 $= 150,000,000/4/(2 \times 122) = 153,689 \text{ [bps]}$

Error = $(153,689/153,600 - 1) \times 100$
 $= 0.058 \text{ [\%]}$

(3) Baud rate setting example

★

Table 10-4. Baud Rate Generator Setting Data

Baud Rate (bps)	fx = 150 MHz		fx = 133 MHz		fx = 100 MHz		fx = 80 MHz	
	k	ERR	k	ERR	k	ERR	k	ERR
300	62,500	0.000	55,417	-0.001	41,667	-0.001	33,333	0.001
600	31,250	0.000	27,708	0.001	20,833	0.002	16,667	-0.002
1,200	15,625	0.000	13,854	0.001	10,417	-0.003	8,333	0.004
2,400	7,813	-0.006	6,927	0.001	5,208	0.006	4,167	-0.008
4,800	3,906	0.006	3,464	-0.013	2,604	0.006	2,083	0.016
9,600	1,953	0.006	1,732	-0.013	1,302	0.006	1,042	-0.032
19,200	977	-0.045	866	-0.013	651	0.006	521	-0.032
31,250	600	0.000	532	0.000	400	0.000	320	0.000
38,400	488	0.058	433	-0.013	326	-0.147	260	0.160
76,800	244	0.058	216	0.218	163	-0.147	130	0.160
153,600	122	0.058	108	0.218	81	0.469	65	0.160
312,500	60	0.000	53	0.377	40	0.000	32	0.000

Caution The maximum allowable frequency of the main clock (fx) is 150 MHz.

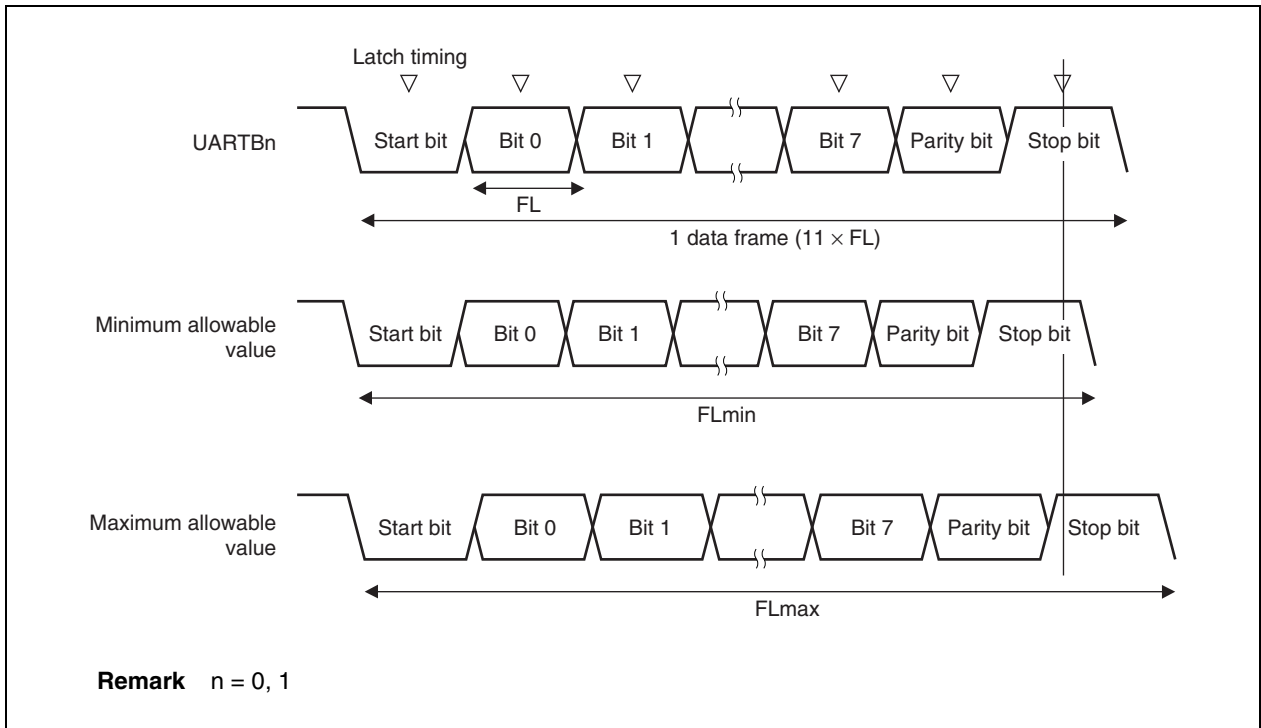
Remark fx: Main clock
k: Settings of UBnBRS15 to UBnBRS0 bits in UBnCTL2 register (n = 0, 1)
ERR: Baud rate error [%]

(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 10-9. Allowable Baud Rate Range During Reception



As shown in Figure 10-9, after the start bit is detected, the receive data latch timing is determined according to the counter that was set by the UBnCTL2 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: UARTBn baud rate ($n = 0, 1$)
- k: UBnCTL2 setting value ($n = 0, 1$)
- FL: 1-bit data length
- Latch timing margin: 2 clocks

$$\text{Minimum allowable value: } FL_{\min} = 11 \times FL - \frac{k - 2}{2k} \times FL = \frac{21k + 2}{2k} FL$$

Therefore, the maximum baud rate that can be received at the transfer destination is as follows.

$$BR_{max} = (FL_{min}/11)^{-1} = \frac{22k}{21k + 2} \text{ Brate}$$

Similarly, the maximum allowable value 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 minimum baud rate that can be received at the transfer destination is as follows.

$$BR_{min} = (FL_{max}/11)^{-1} = \frac{20k}{21k - 2} \text{ Brate}$$

The allowable baud rate error of UAR_TB_n and the transfer destination can be obtained as follows from the expressions described above for computing the minimum and maximum baud rate values.

Table 10-5. Maximum and Minimum Allowable Baud Rate Error

Division Ratio (k)	Maximum Allowable Baud Rate Error	Minimum Allowable Baud Rate Error
4	+2.33 %	-2.44
8	+3.53 %	-3.61
16	+4.14 %	-4.19
32	+4.45 %	-4.48
64	+4.61 %	-4.62
128	+4.68 %	-4.69
256	+4.72 %	-4.73
512	+4.74 %	-4.74
1,024	+4.75 %	-4.75
2,048	+4.76 %	-4.76
4,096	+4.76 %	-4.76
8,192	+4.76 %	-4.76
16,384	+4.76 %	-4.76
32,768	+4.76 %	-4.76
65,535	+4.76 %	-4.76

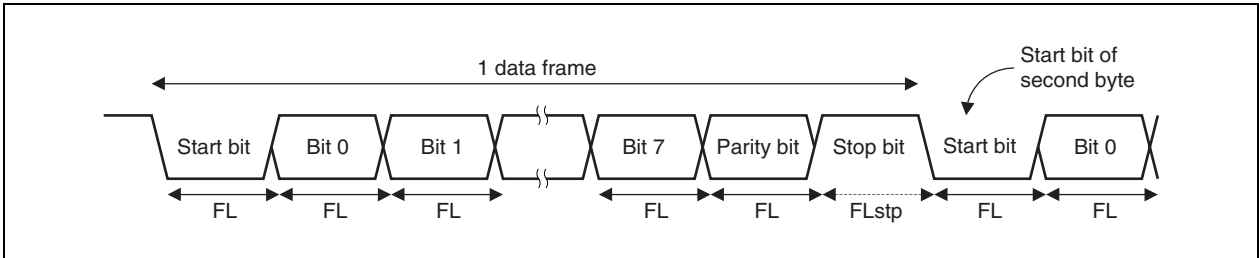
Remarks 1. The reception precision depends on the number of bits in one frame, the base clock frequency, and the division ratio (k). The higher the base clock frequency and the larger the division ratio (k), the higher the precision.

2. k: UB_nCTL2 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 10-10. Transfer Rate During Continuous Transmission



Representing the 1-bit data length by FL, the stop bit length by FLstp, and the base clock frequency by $f_x/4$ (f_x : main clock) yields the following equation.

$$FL_{stp} = FL + 2/(f_x/4)$$

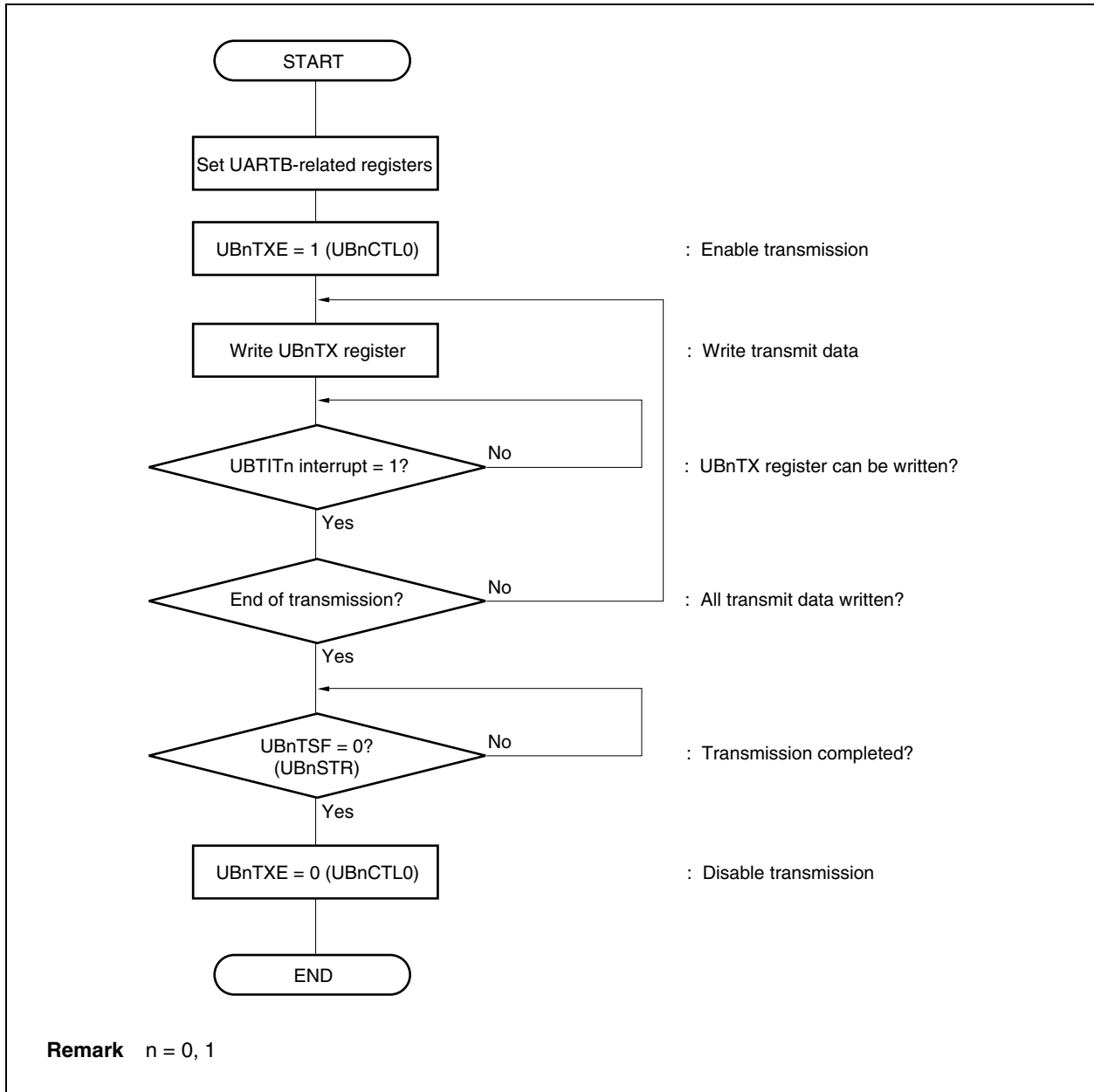
Therefore, the transfer rate during continuous transmission is as follows.

$$\text{Transfer rate} = 11 \times FL + 2/(f_x/4)$$

★ 10.2.8 Control flow

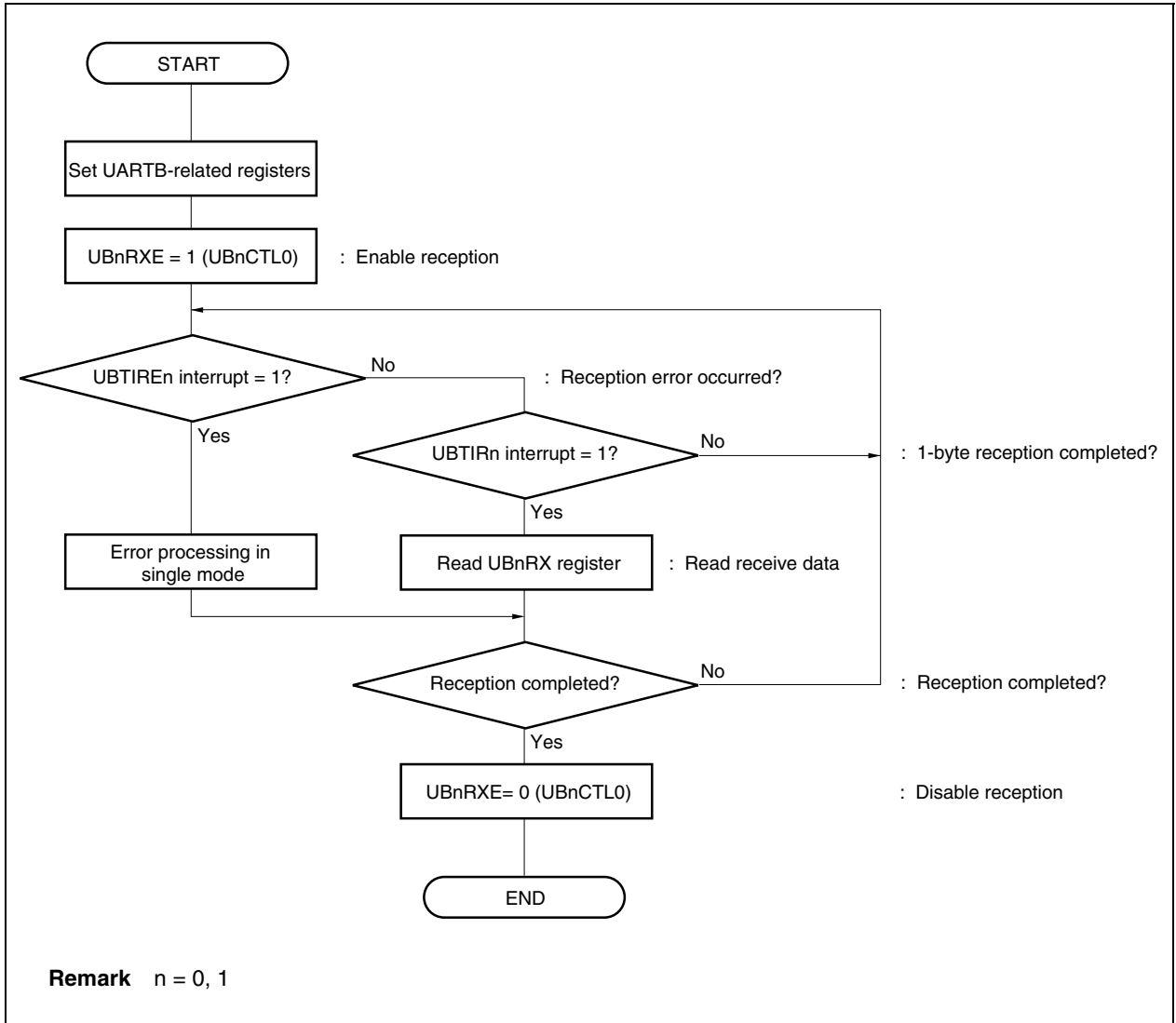
(1) Example of continuous transmission processing flow in single mode (CPU control)

Figure 10-11. Example of Continuous Transmission Processing Flow in Single Mode (CPU Control)



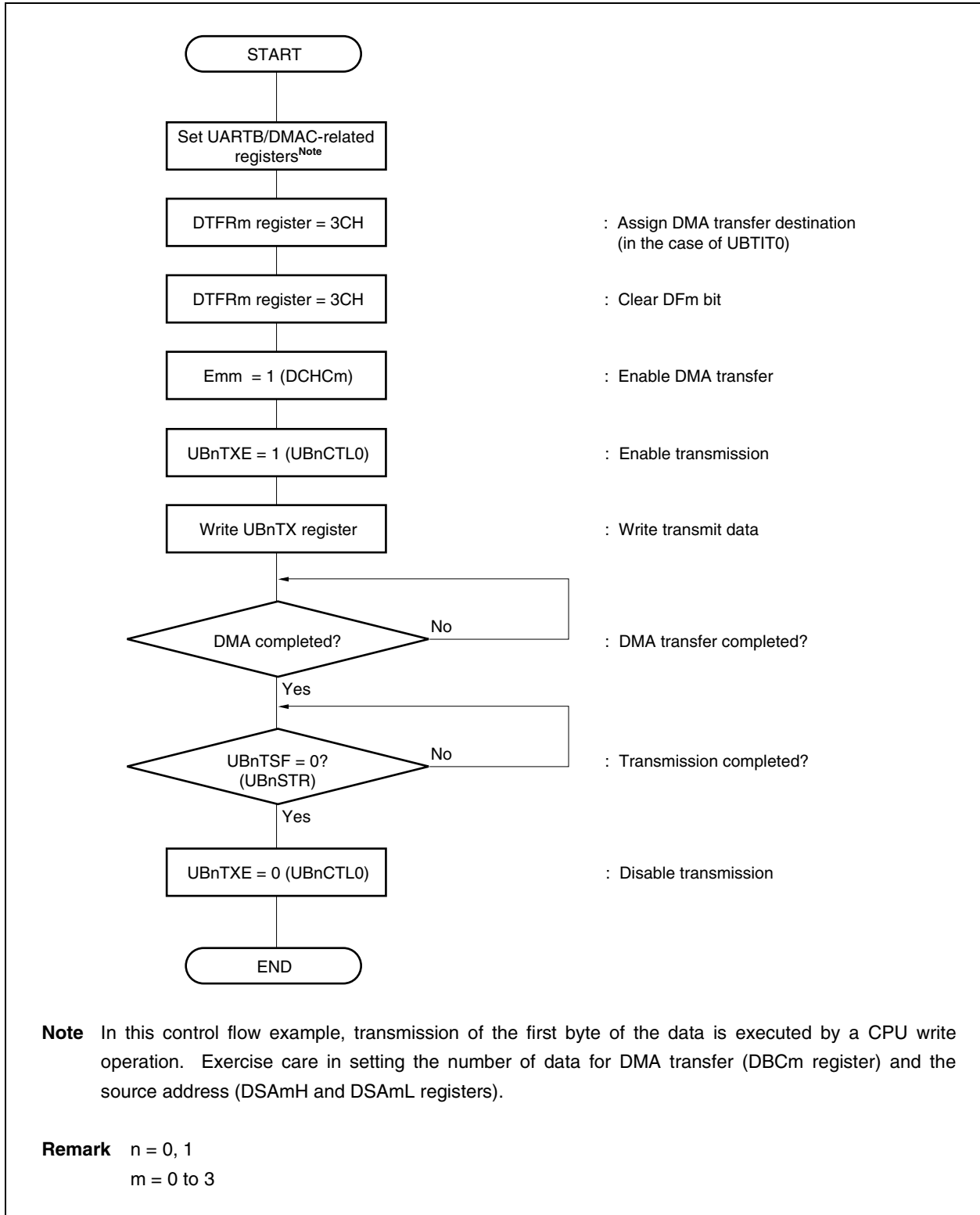
(2) Example of continuous reception processing flow in single mode (CPU control)

Figure 10-12. Example of Continuous Reception Processing Flow in Single Mode (CPU Control)



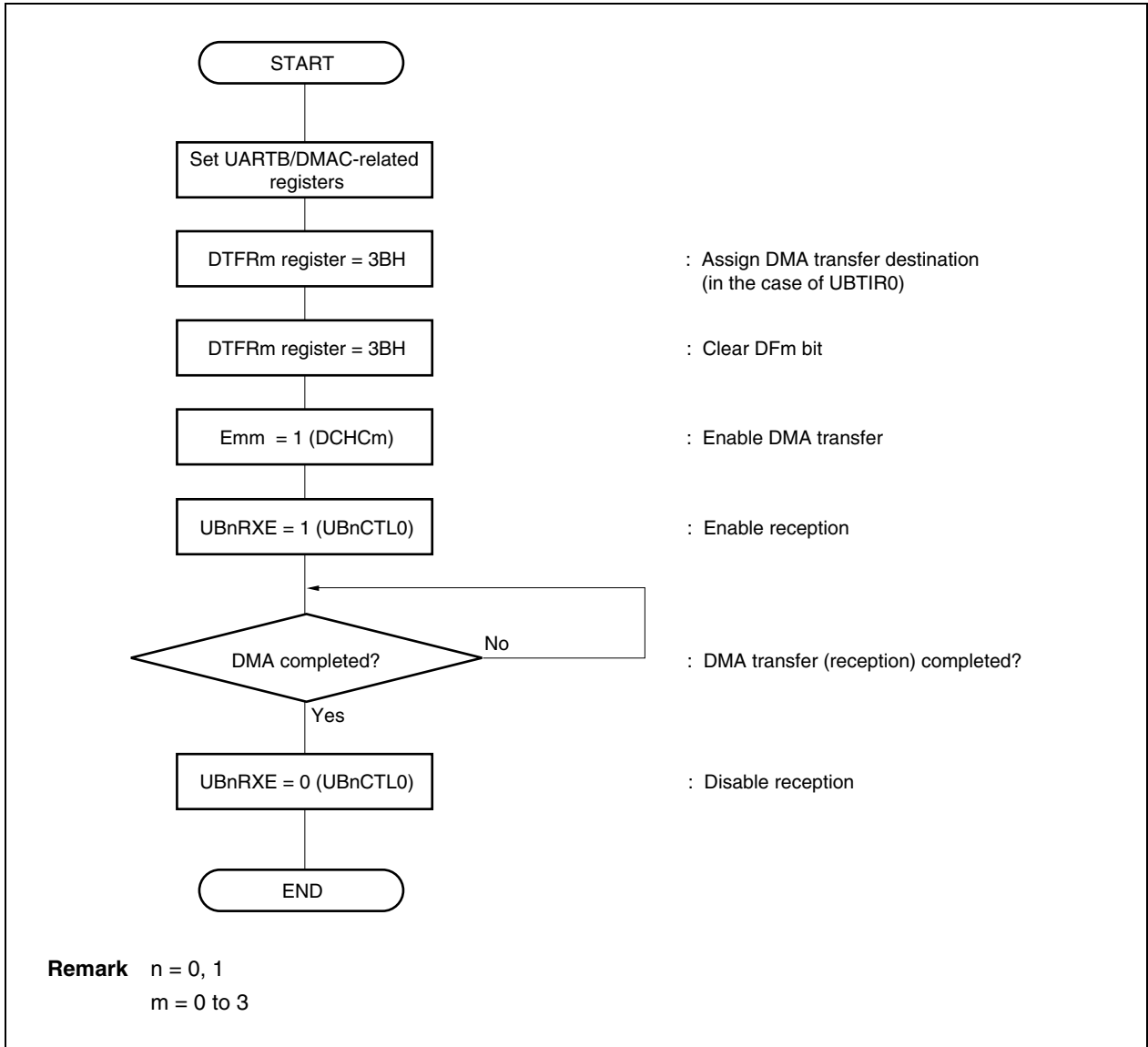
(3) Example of continuous transmission processing flow in single mode (DMA control)

Figure 10-13. Example of Continuous Transmission Processing Flow in Single Mode (DMA Control)



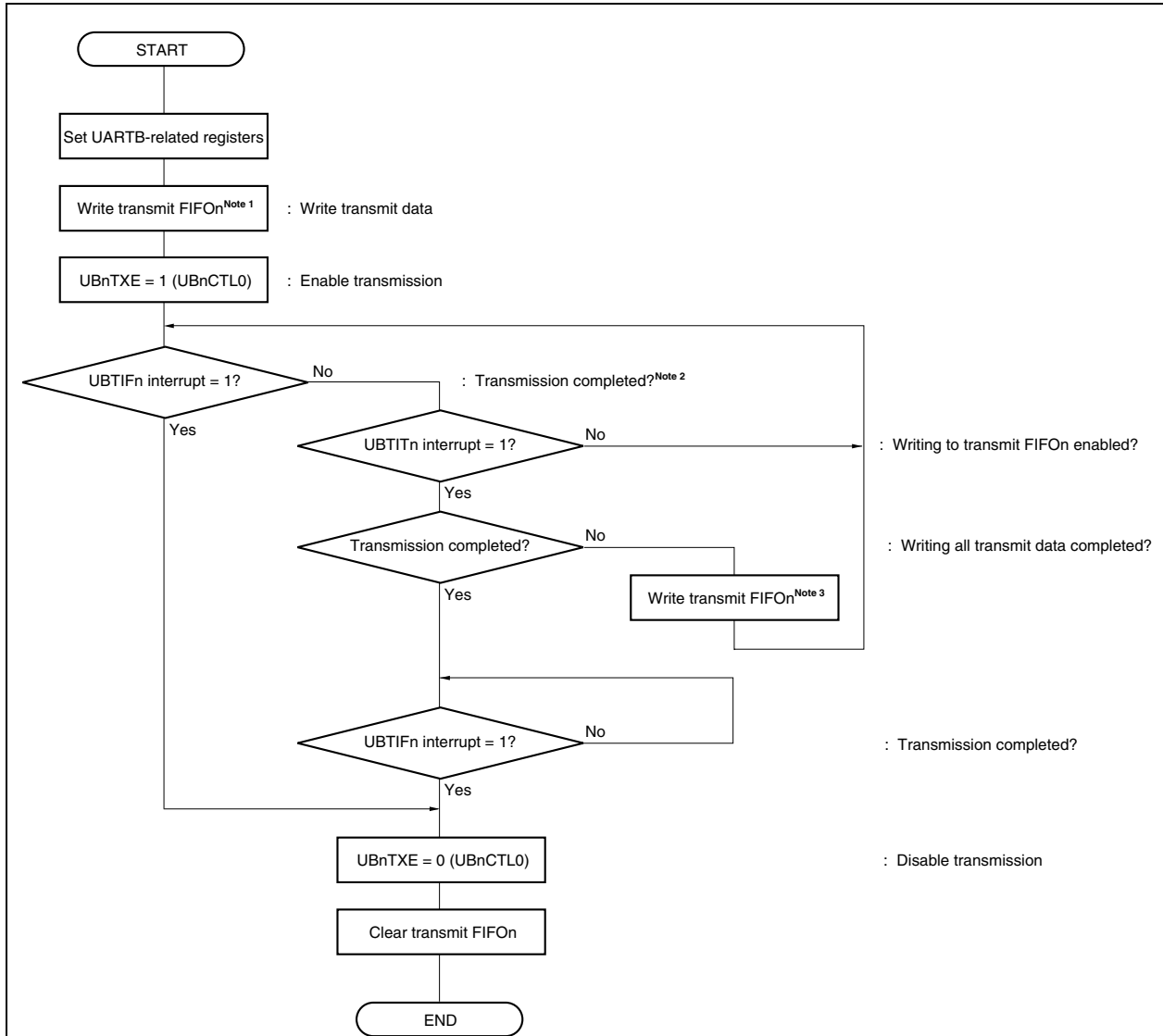
(4) Example of continuous reception processing flow in single mode (DMA control)

Figure 10-14. Example of Continuous Reception Processing Flow in Single Mode (DMA Control)



(5) Example of continuous transmission processing flow in FIFO mode (CPU control)

Figure 10-15. Example of Continuous Transmission Processing Flow in FIFO Mode (CPU Control)



Notes 1. Write more transmit data than the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register to transmit FIFO_n.

2. This is the case where transmission is completed (transmit FIFO_n and transmit shift register become empty) before the next transmit data is written. To continue data transmission, clear the UBTIF_n and UBTIT_n interrupts and write the next data to transmit FIFO_n.

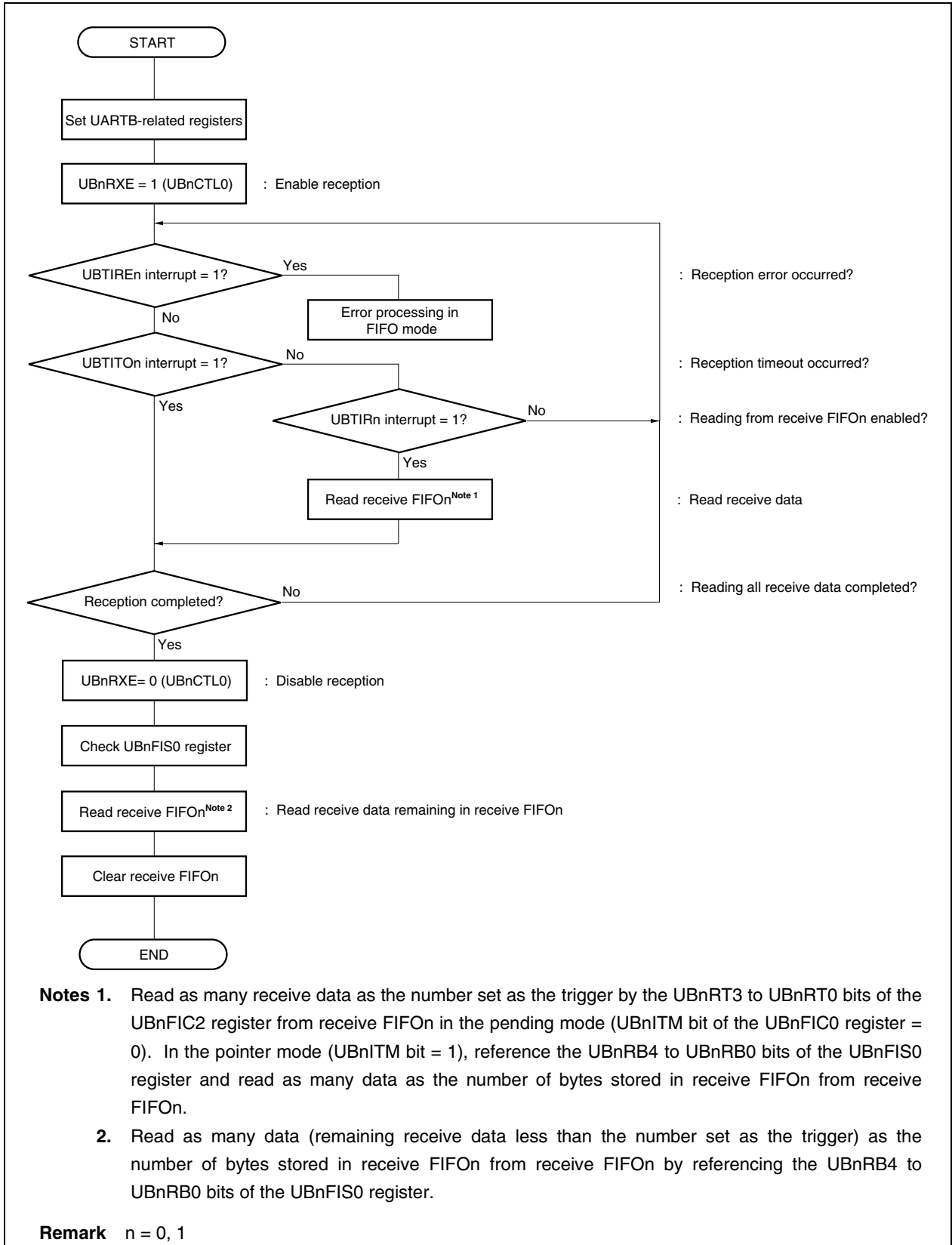
3. In the pending mode (UBnITM bit of the UBnFIC0 register = 0), write as many transmit data as the number set as the trigger by the UBnTT3 to UBnTT0 bits of the UBnFIC2 register to transmit FIFO_n. In the pointer mode (UBnITM bit = 1), reference the UBnTB4 to UBnTB0 bits of the UBnFIS1 register and write as many data as the number of vacant bytes in transmit FIFO_n to transmit FIFO_n.

To fully use the 8-bit × 16-stage FIFO function, write 16-byte data.

Remark n = 0, 1

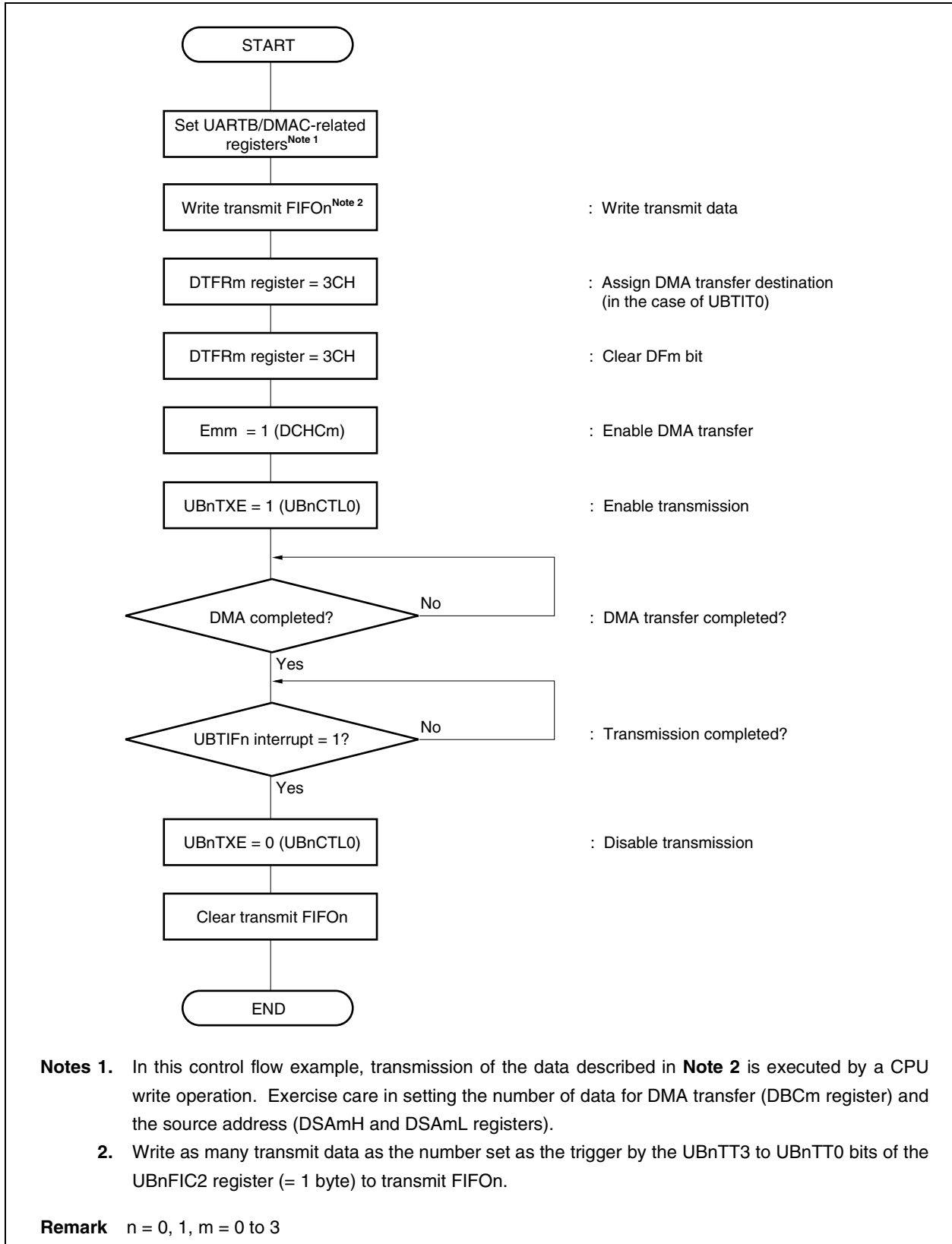
(6) Example of continuous reception processing in FIFO mode (CPU control)

Figure 10-16. Example of Continuous Reception Processing in FIFO Mode (CPU Control)



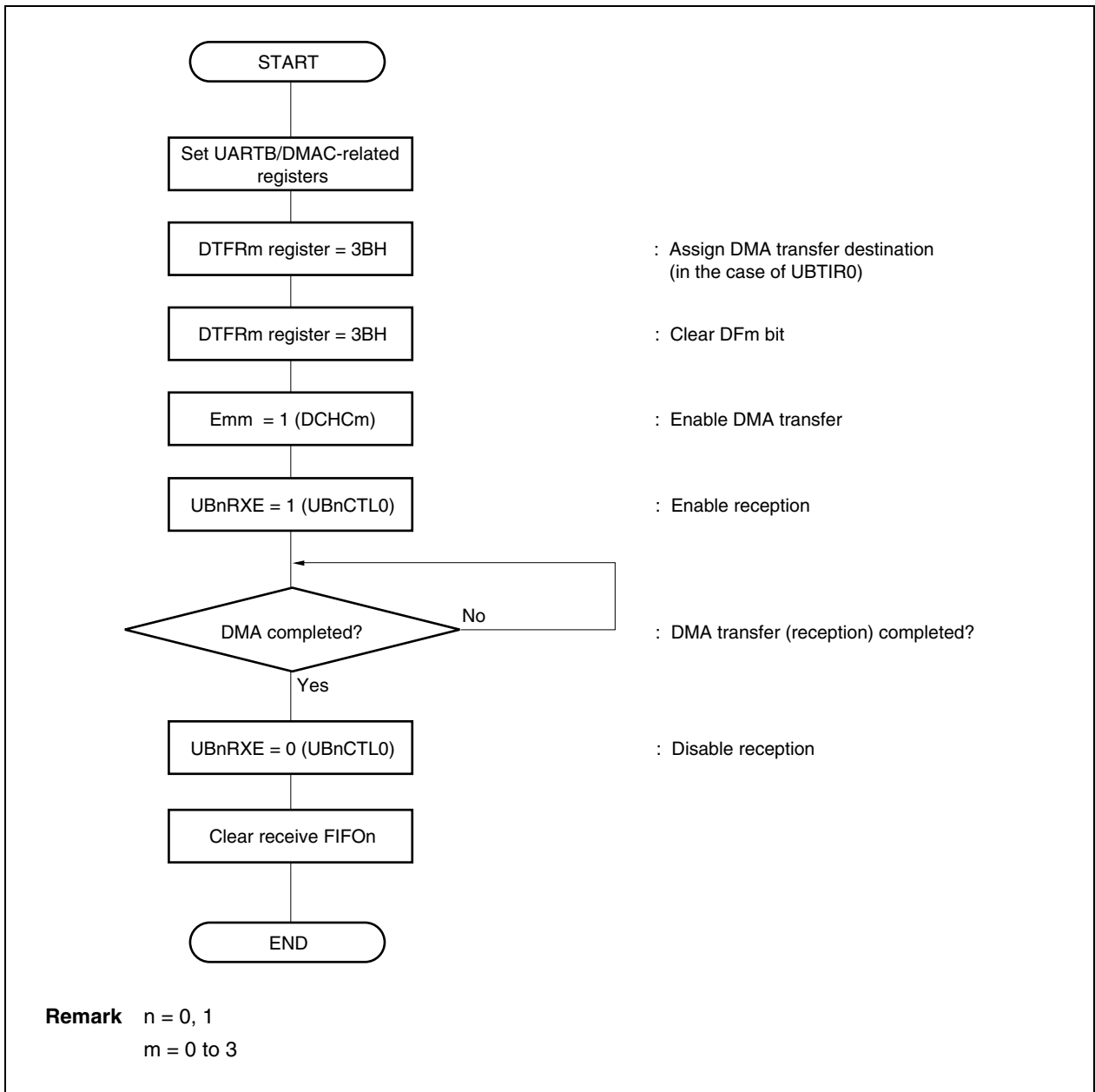
(7) Example of continuous transmission (pending mode) processing in FIFO mode (DMA control)

Figure 10-17. Example of Continuous Transmission (Pending Mode) Processing in FIFO Mode (DMA Control)



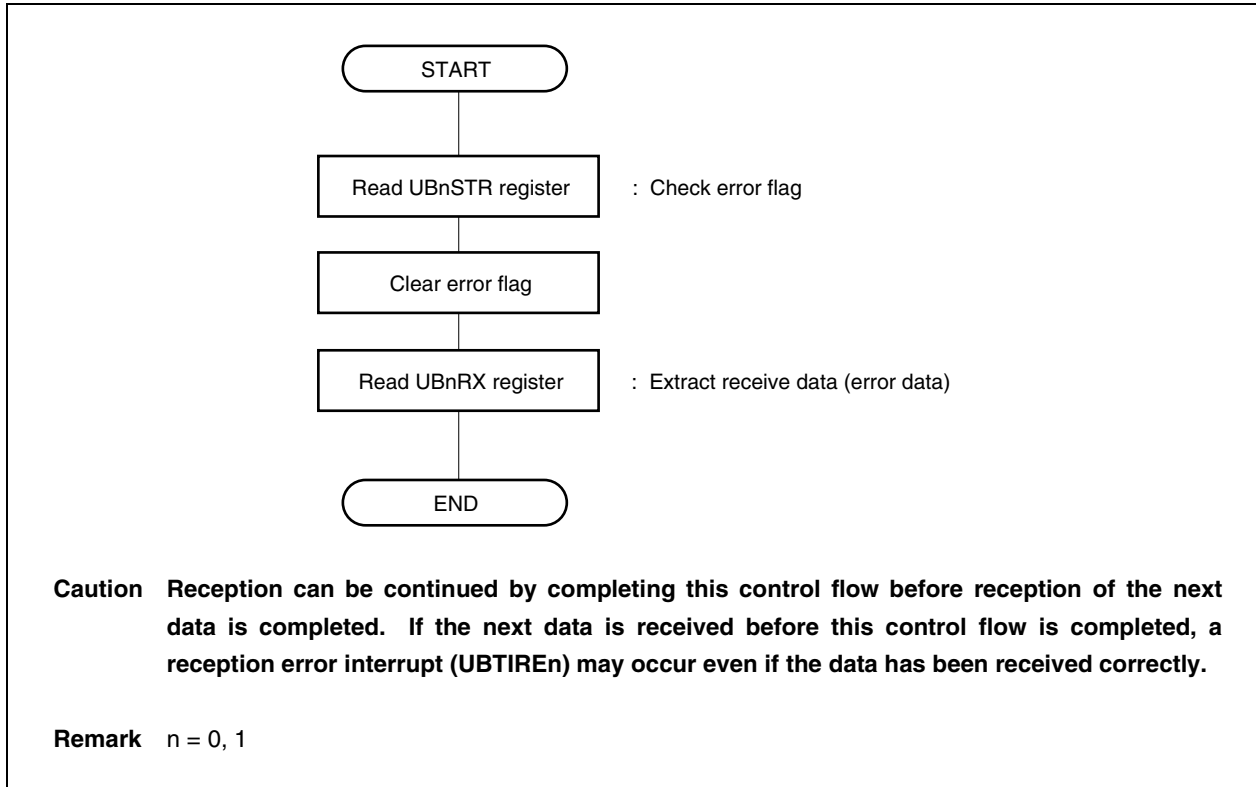
(8) Example of continuous reception (pending mode) processing flow in FIFO mode (DMA control)

Figure 10-18. Example of Continuous Reception (Pending Mode) Processing Flow in FIFO Mode (DMA Control)



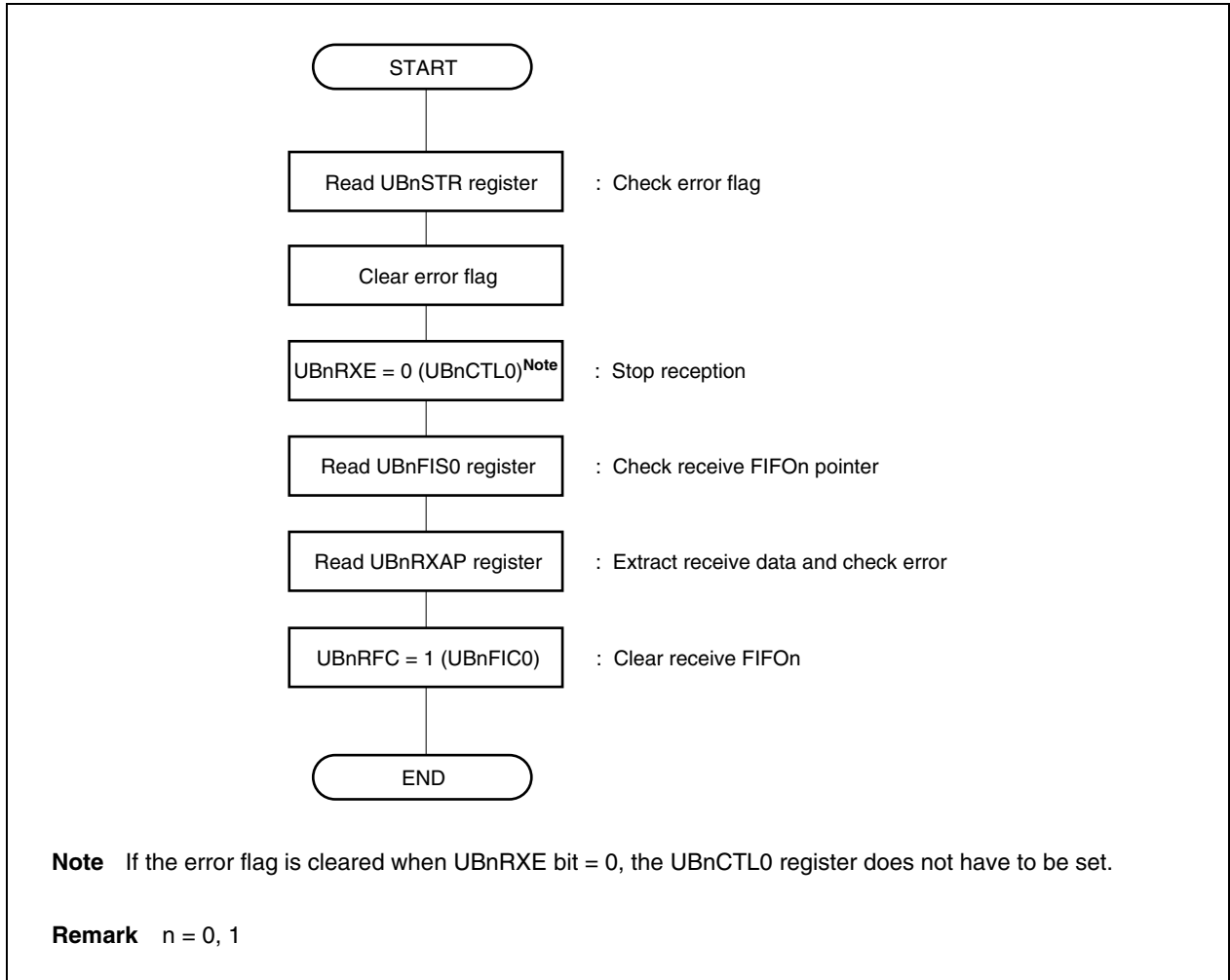
(9) Example of reception error processing in single mode

Figure 10-19. Example of Reception Error Processing in Single Mode



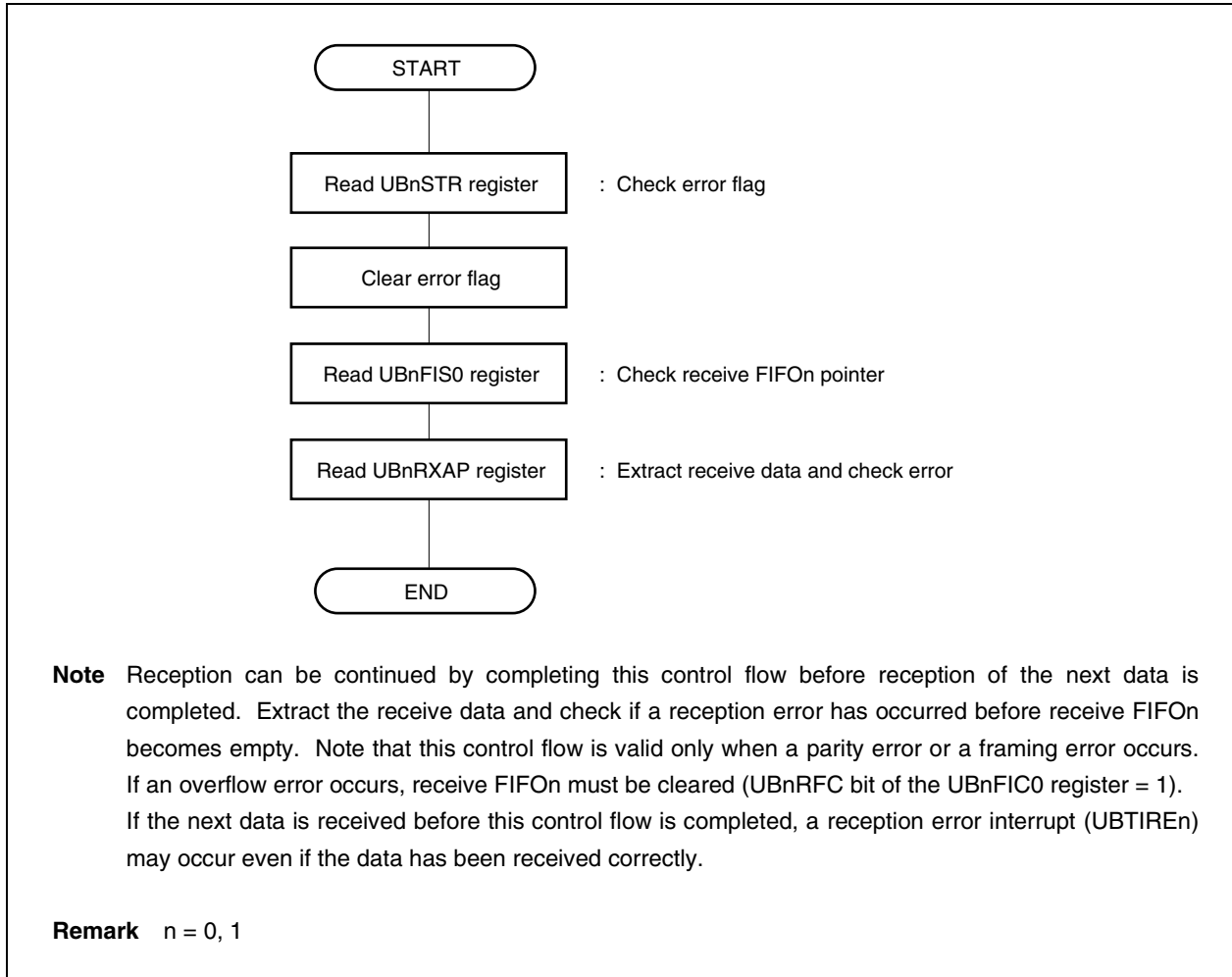
(10) Example of reception error processing flow in FIFO mode (1)

Figure 10-20. Example of Reception Error Processing Flow in FIFO Mode (1)



(11) Example of reception error processing flow in FIFO mode (2)

Figure 10-21. Example of Reception Error Processing Flow in FIFO Mode (2)

**10.2.9 Cautions**

When the supply of clocks to UARTBn is stopped (for example, IDLE or 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 UBnPWR = 0, UBnRXE = 0, and UBnTXE = 0.

10.3 Clocked Serial Interfaces 30, 31 (CSI30, CSI31)

10.3.1 Features

- Transfer rate: Master mode/slave mode: Maximum 5.5 Mbps
- Half-duplex communications
- Master mode and slave mode can be selected
- Transmission data length: 8 to 16 bits (selectable in 1-bit units)
- Transfer data direction can be switched between MSB first and LSB first
- 3-wire mode
 - SO_n: Serial data output
 - SIn: Serial data input
 - SCK_n: Serial clock I/O
- Bit rate
 - In master mode: BRG output (selected by the CKS3_n2 to CKS3_n0 bits and MDL_n2 to MDL_n0 bits of clocked serial interface clock select register n (CSIC3_n))
 - In slave mode: Clock input from master (when CKS3_n2 to CKS3_n0 bits = 111 in the CSIC3_n register)
- Interrupt sources: 2 types
 - Transmission/reception completion interrupt (INTCSI3_n)
 - CSIBUF_n overflow interrupt (INTCOVF3_n)
- Transmission mode, reception mode, or transmission/reception mode can be selected
 - Transmission mode: Transmission is started by writing transmit data to transmit data CSI buffer register 3_n (SFDB3_n) while transmission is enabled (refer to **10.3.5 (11) Transmission mode**).
 - Reception mode: Reception is started by using processing that writes dummy data to transmit data CSI buffer register 3_n (SFDB3_n) as a trigger while reception is enabled (refer to **10.3.5 (12) Reception mode**).
 - Transmission/reception mode: Transmission/reception is started by using processing that writes transmit data to transmit data CSI buffer register 3_n (SFDB3_n) while transmission/reception is enabled (refer to **10.3.5 (13) Transmission/reception mode**).
- Sixteen on-chip 16-bit transmit/receive buffers (CSIBUF_n)
- On-chip dedicated baud rate generator

Remark n = 0, 1

10.3.2 Configuration

CSI3n is controlled by the clocked serial interface mode register 3n (CSIM3n) (n = 0, 1).

(1) Clocked serial interface mode registers 30, 31 (CSIM30, CSIM31)

The CSIM3n register is an 8-bit register for specifying the operation of CSI3n.

(2) Clocked serial interface clock select registers 30, 31 (CSIC30, CSIC31)

The CSIC3n register is an 8-bit register for controlling the operation clock and operating mode of CSI3n.

(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.

(4) Receive data buffer registers 30, 31 (SIRB30, SIRB31)

The SIRB3n register is a 16-bit buffer register that stores receive data. This register is also divided into two registers: the higher 8 bits (SIRB3nH) and lower 8 bits (SIRB3nL).

(5) Transmit data CSI buffer registers 30, 31 (SFDB30, SFDB31)

The SFDB3n register is a 16-bit buffer register that stores transmit data. This register is also divided into two registers: the higher 8 bits (SFDB3nH) and lower 8 bits (SFDB3nL).

(6) CSIBUF status registers 30, 31 (SFA30, SFA31)

The SFA3n register is an 8-bit register that indicates the status of CSI data buffer register n (CSIBUF_n) or the transfer status.

(7) Transfer data length select registers 30, 31 (CSIL30, CSIL31)

The CSIL3n register is an 8-bit register that selects the CSI3n transfer data length.

(8) Transfer data number specification registers 30, 31 (SFN30, SFN31)

The SFN3n register is an 8-bit register that sets the number of CSI3n transfer data in continuous mode.

(9) CSI data buffer registers 0, 1 (CSIBUF0, CSIBUF1)

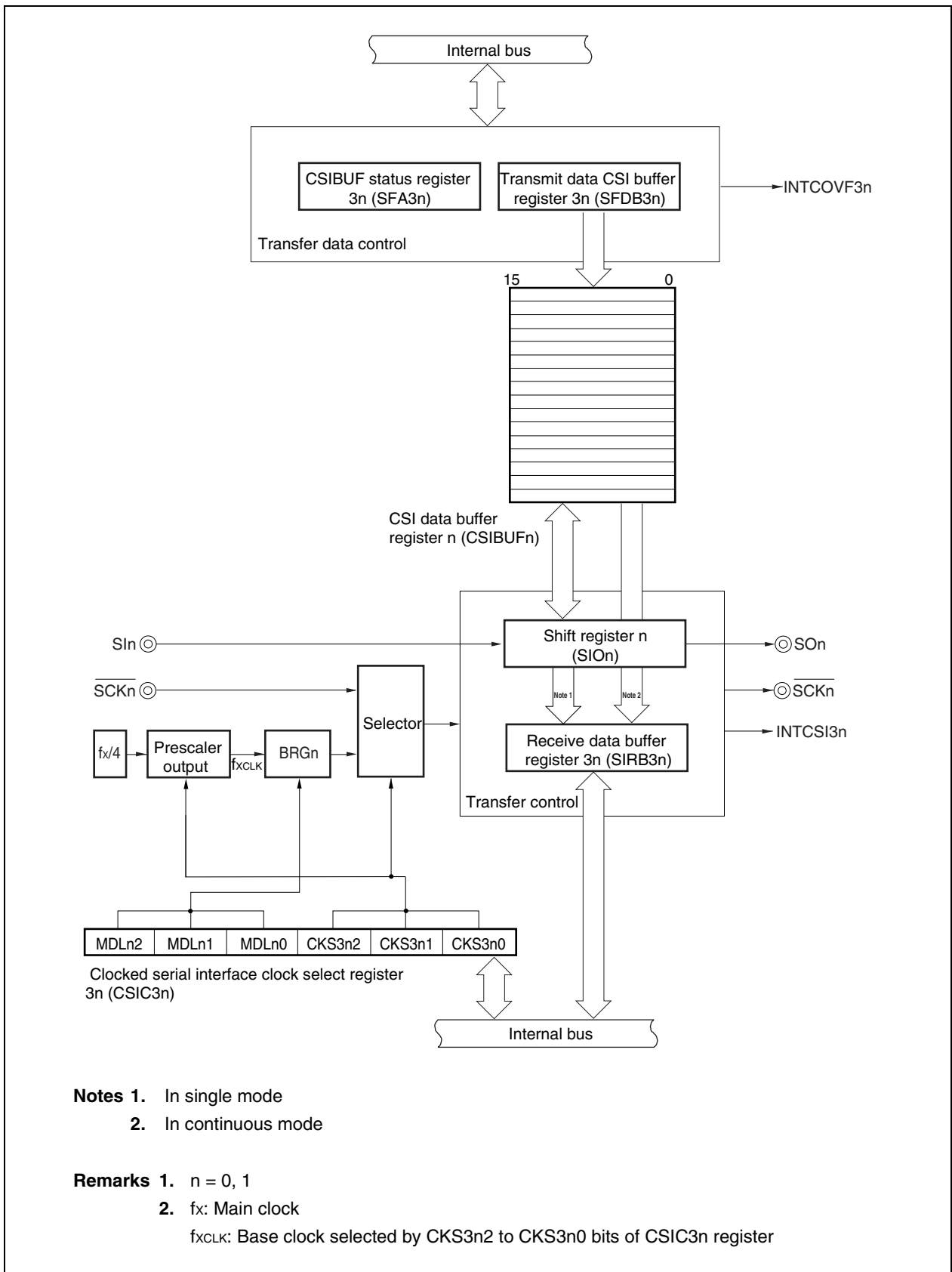
By consecutively writing transmit data to the SFDB3n register from where it is transferred, the data can be stored in the CSIBUF_n register while the CSIBUF_n pointer for writing is automatically incremented (CSIBUF_n).

The CSIBUF_n is a 16-bit buffer register.

In the continuous mode, the data received in the CSIBUF_n register can be sequentially read while the read CSIBUF_n pointer is automatically incremented, by continuously reading the receive data from the SIRB3n register.

★

Figure 10-22. Block Diagram of Clocked Serial Interfaces 30 and 31



10.3.3 Control registers

Because CSI30 shares pins with UARTB0, the CSI30 mode must be set in advance by using the PMC1 and PFC1 registers (see 10.1.1 Switching between UARTB0 and CSI30 modes).

(1) Clocked serial interface mode registers 30, 31 (CSIM30, CSIM31)

The CSIM3n register controls the operation of CSI3n (n = 0, 1).

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

Be sure to clear bit 0 to 0. If it is set to 1, the operation is not guaranteed.

Cautions 1. Writing the TRMDn, DIRn, CSITn, and CSWEn bits is enabled only when CTXEn bit = 0 and CRXEn bit = 0.

★ **2 To use CSI3n, be sure to set the external pins related to the CSI3n function to control the mode. Then set the CSICAE n bit to 1 before setting the other bits.**

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
CSIM30	CSICAE0	CTXE0	CRXE0	TRMD0	DIR0	CSIT0	CSWE0	0	FFFFFD00H	00H
CSIM31	CSICAE1	CTXE1	CRXE1	TRMD1	DIR1	CSIT1	CSWE1	0	FFFFFD20H	00H

Bit position	Bit name	Function
7	CSICAE n	<p>Controls the operating clock.</p> <p>0: Stop clock supply to CSI3n. 1: Supply clock to CSI3n.</p> <p>Cautions 1. The CSI3n unit is reset when the CSICAE n bit = 0, and CSI3n is stopped. To operate CSI3n, first set the CSICAE n bit to 1.</p> <p>2. When rewriting the CSICAE n bit from 0 to 1 or from 1 to 0, simultaneously rewriting the bits other than the CSICAE n bit of the CSIM3n register is prohibited.</p> <p>When the CSICAE n bit = 0, rewriting the bits other than the CSICAE n bit of the CSIM3n register, and the SFDB3n, SFDB3nL, and SFA3n registers is prohibited.</p>
6	CTXE n	<p>Enables or disables transmission.</p> <p>1: Disables transmission. 0: Enables transmission.</p> <p>Caution The CTXE n bit is reset when the CSICAE n bit is cleared to 0.</p>

Remark n = 0, 1

Bit position	Bit name	Function
5	CRXEn	<p>Enables or disables reception.</p> <p>1: Disables reception. 0: Enables reception.</p> <p>Caution The CRXEn bit is reset when the CSICAEn bit is cleared to 0.</p>
4	TRMDn	<p>Specifies the transfer mode.</p> <p>0: Single mode 1: Continuous mode</p>
3	DIRn	<p>Specifies the transfer direction when data is written from the SFDB3n register to the CSIBUFn register or read from the SIRB3n and CSIBUFn registers.</p> <p>0: The first bit of transfer data is the MSB. 1: The first bit of transfer data is the LSB.</p>
2	CSITn	<p>Controls delay of the transmission completion interrupt signal (INTCSI3n) (see 10.3.5 (14) Delay control of transmission/reception completion interrupt (INTCSI3n)).</p> <p>0: No delay 1: Delay mode (In the continuous mode (TRMDn = 1), the next data transfer is delayed half a cycle because the interrupt request signal is delayed half a cycle.)</p> <p>Cautions</p> <ol style="list-style-type: none"> The delay mode (CSIT bit = 1) is valid only in the master mode (when the CKS3n2 to CKS3n0 bits of the CSIC3n register are other than 111). In the slave mode (when the CKS3n2 to CKS3n0 bits are 111), do not set the delay mode. Even if the delay mode is set, INTCSI3n is not affected by the CSITn bit. If the CSITn bit is set to 1 in the continuous mode (TRMDn bit = 1), the INTCSI3n interrupt is not output except when the last data set by the SFNn3 to SFNn0 bits of the SFN3n register is transferred, but a delay of half a clock can be inserted between each data transfer.
1	CSWEn	<p>Enables or disables transfer wait.</p> <p>0: Disables transfer wait (1 wait cycle not inserted on starting transfer). 1: Enables transfer wait (1 wait cycle inserted on starting transfer).</p> <p>Caution Inserting a transfer wait cycle (CSWEn bit = 1) is valid only in the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register). In the slave mode (CKS3n2 to CKS3n0 bits = 111), do not insert a transfer wait cycle. Even if set, a transfer wait cycle is not inserted.</p>

Remark n = 0, 1

(2) Clocked serial interface clock select registers 30, 31 (CSIC30, CSIC31)

The CSIC3n register is an 8-bit register that controls the operation clock and operating mode of CSI3n. These registers can be read or written in 8-bit or 1-bit units.

Caution Data can be written to the CSIC3n register only when the CTXEn bit = 0 and CRXEn bit = 0 in the CSIM3n register.

(1/3)

	7	6	5	4	3	2	1	0	Address	After reset
CSIC30	MDL02	MDL01	MDL00	CKP0	DAP0	CKS302	CKS301	CKS300	FFFFFFD01H	07H
CSIC31	MDL12	MDL11	MDL10	CKP1	DAP1	CKS312	CKS311	CKS310	FFFFFFD21H	07H

Bit position	Bit name	Function																																													
7 to 5	MDLn2 to MDLn0	Specify the transfer clock (BRGn output signal). <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>MDLn2</th> <th>MDLn1</th> <th>MDLn0</th> <th>Set value (N)</th> <th>Transfer clock (BRGn output signal)</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>–</td> <td>BRGn stop mode (power save)</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>f_{XCLK}/2</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>2</td> <td>f_{XCLK}/4</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>3</td> <td>f_{XCLK}/6</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>4</td> <td>f_{XCLK}/8</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>5</td> <td>f_{XCLK}/10</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>6</td> <td>f_{XCLK}/12</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>7</td> <td>f_{XCLK}/14</td> </tr> </tbody> </table> <p>Caution In the slave mode (CKS3n2 to CKS3n0 bits = 111), it is recommended to clear the MDLn2 to MDLn0 bits to 000 (BRGn stop mode).</p> <p>Remark f_{XCLK}: Base clock selected by CKS3n2 to CKS3n0 bits</p>	MDLn2	MDLn1	MDLn0	Set value (N)	Transfer clock (BRGn output signal)	0	0	0	–	BRGn stop mode (power save)	0	0	1	1	f _{XCLK} /2	0	1	0	2	f _{XCLK} /4	0	1	1	3	f _{XCLK} /6	1	0	0	4	f _{XCLK} /8	1	0	1	5	f _{XCLK} /10	1	1	0	6	f _{XCLK} /12	1	1	1	7	f _{XCLK} /14
MDLn2	MDLn1	MDLn0	Set value (N)	Transfer clock (BRGn output signal)																																											
0	0	0	–	BRGn stop mode (power save)																																											
0	0	1	1	f _{XCLK} /2																																											
0	1	0	2	f _{XCLK} /4																																											
0	1	1	3	f _{XCLK} /6																																											
1	0	0	4	f _{XCLK} /8																																											
1	0	1	5	f _{XCLK} /10																																											
1	1	0	6	f _{XCLK} /12																																											
1	1	1	7	f _{XCLK} /14																																											

Remark n = 0, 1

Bit position	Bit name	Function															
4, 3	CKPn, DAPn	Specify the data transmission/reception timing for \overline{SCKn} . <table border="1" style="margin: 5px auto; width: 80%;"> <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^{Note}</td> <td>0</td> <td> </td> </tr> <tr> <td>1^{Note}</td> <td>1</td> <td> </td> </tr> </tbody> </table>	CKPn	DAPn	Operation mode	0	0		0	1		1 ^{Note}	0		1 ^{Note}	1	
CKPn	DAPn	Operation mode															
0	0																
0	1																
1 ^{Note}	0																
1 ^{Note}	1																

Note If the CKPn bit is set to 1 in the master mode (CKS3n2 to CKS3n0 bits are other than 111), the \overline{SCKn} pin outputs a low level when it is inactive. If the CTXEn bit of the CSIM3n register is cleared to 0 (disabling transmission) and CRXEn bit is cleared to 0 (disabling reception), the \overline{SCKn} pin outputs a high level.

Therefore, take the following measures to fix the \overline{SCKn} pin to low level when CSI3n is not used.

[$\overline{SCK0}$ pin ($\overline{SCK1}$ pin)]

- <1> Clearing the P11 bit of the P1 register to 0 (clearing the P23 bit of the P2 register to 0):
The port output level is set to low.
- <2> Clearing the PM11 bit of the PM1 register to 0 (clearing the PM23 bit of the PM2 register to 0):
The port is set in the output mode.
- <3> Clearing the PMC11 bit of the PMC1 register to 0 (clearing the PMC23 bit of the PMC2 register to 0):
The pin is set in the port mode (fixed to low-level output).
- <4> Clearing the CTXE0 and CRXE0 bits of the CSIM30 register to 0 (clearing the CTXE1 and CRXE1 bits of the CSIM31 register to 0):
Transmission and reception are disabled.
- <5> Setting the CTXE0 or CRXE0 bit of the CSIM30 register to 1 (setting the CTXE1 or CRXE1 bit of the CSIM31 register to 1):
Transmission or reception is enabled (both transmission and reception can also be enabled).
- <6> Setting the PMC11 bit of the PMC1 register to 1 (setting the PMC23 bit of the PMC2 register to 1):
The pin is set in the control mode ($\overline{SCK0}$ and $\overline{SCK1}$ pin output).

Because the register set values <1> and <2> are retained, control can be performed only by <3> to <6> once they have been set.

Remark n = 0, 1

★

Bit position	Bit name	Function					
2 to 0	CKS3n2 to CKS3n0	Specify the base clock (prescaler output).					
		CKS3n2	CKS3n1	CKS3n0	Set value (k)	Base clock (f_{XCLK})	Mode
		0	0	0	0	$f_x/4$	Master mode
		0	0	1	1	$f_x/8$	Master mode
		0	1	0	2	$f_x/16$	Master mode
		0	1	1	3	$f_x/32$	Master mode
		1	0	0	4	$f_x/64$	Master mode
		1	0	1	5	$f_x/128$	Master mode
		1	1	0	6	$f_x/256$	Master mode
1	1	1	7	External clock (\overline{SCKn})	Slave mode		
Remark f_x : Main clock							

Remark n = 0, 1

(3) Receive data buffer registers 30, 31 (SIRB30, SIRB31)

The SIRB3n register is a 16-bit buffer register that stores receive data.

By consecutively reading this register in the continuous mode (TRMDn bit = 1 in the CSIM3n register), the received data in the CSIBUFn register can be sequentially read while the CSIBUFn pointer for reading is incremented.

In the single mode (TRMDn bit = 0 in the CSIM3n register), received data is read by reading the SIRB3n register and it is judged that the SIRB3n register has become empty.

The SIRB3n register is read-only, in 16-bit units.

When the higher 8 bits of the SIRB3n register are used as the SIRB3nH register and the lower 8 bits as the SIRB3nL register, these registers are read-only, in 8-bit units. When reading in 8-bit units, be sure to read the SIRB3nH register and SIRB3nL register in that order.

★

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset ^{Note}
SIRB30	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	FFFFFD02H	0000H
	015	014	013	012	011	010	009	008	007	006	005	004	003	002	001	000		
SIRB31	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	SIRB	FFFFFD22H	0000H
	115	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100		

Note In continuous mode (TRMDn bit = 1 in the CSIM3n register): Undefined

Bit position	Bit name	Function
15 to 0	SIRBn15 to SIRBn0	Store receive data.

Remarks

1. n = 0, 1
2. The SIRB3n register is cleared to 0000H when the CSICAEn bit of the CSIM3n register is cleared to 0.

(4) Transmit data CSI buffer registers 30, 31 (SFDB30, SFDB31)

The SFDB3n register is a 16-bit buffer register that stores transmit data.

When transmit data is written to this register, the data is sequentially stored in the CSIBUFn register while the CSIBUFn pointer for writing is incremented.

When the data of this register is read, the value of the transmit data written last is read.

The SFDB3n register can be read or written in 16-bit units.

When the higher 8 bits of the SFDB3n register are used as the SFDB3nH register, and the lower 8 bits as the SFDB3nL register, these registers can be read or written in 8-bit units. When reading in 8-bit units, be sure to read the SFDB3nH register and SFDB3nL register in that order.

★

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
SFDB30	SFDB 015	SFDB 014	SFDB 013	SFDB 012	SFDB 011	SFDB 010	SFDB 09	SFDB 08	SFDB 07	SFDB 06	SFDB 05	SFDB 04	SFDB 03	SFDB 02	SFDB 01	SFDB 00	FFFFFFD06H	0000H
SFDB31	SFDB 115	SFDB 114	SFDB 113	SFDB 112	SFDB 111	SFDB 110	SFDB 19	SFDB 18	SFDB 17	SFDB 16	SFDB 15	SFDB 14	SFDB 13	SFDB 12	SFDB 11	SFDB 10	FFFFFFD26H	0000H

Bit position	Bit name	Function
15 to 0	SFDBn15 to SFDBn0	Store transmit data.

Remark n = 0, 1

(5) CSIBUF status registers 30, 31 (SFA30, SFA31)

These registers indicate the status of the CSIBUFn register or the transfer status.

These registers can be read or written in 8-bit or 1-bit units (however, bits 6 to 0 can only be read. They do not change even if they are written).

- Cautions**
1. Reading the SFA3n register is prohibited when the CSICAE_n bit of the CSIM3n register is 1 and when the main clock (fx) is stopped.
 2. Because the values of the SFFUL_n, SFEMP_n, CSOT_n, and SFPn3 to SFPn0 bits may change at any time during transfer, their values during transfer may differ from the actual values. Especially, use the CSOT_n bit independently (do not use this bit in relation with the other bits). To detect the end of transfer by the SFA3n register, check to see if the SFEMP_n bit is 1 after the data to be transferred has been written to the CSIBUFn register.
 3. If the SFA3n register is read immediately after data has been written to the SFDB3n and SFDB3nL registers when the main clock (fx) is 84 MHz or lower, the values of the SFFUL_n, SFEMP_n, and SFPn3 to SFPn0 bits do not change in time.
 4. If the SFA3n register is read before the SFFUL_n bit is set to 1 and the 17th data is written, the CSIBUFn overflow interrupt (INTCOVF3n) is generated.

(1/3)

	7	6	5	4	3	2	1	0	Address	After reset
SFA30	FPCLR0	SFFUL0	SFEMP0	CSOT0	SFP03	SFP02	SFP01	SFP00	FFFFFFD08H	20H
SFA31	FPCLR1	SFFUL1	SFEMP1	CSOT1	SFP13	SFP12	SFP11	SFP10	FFFFFFD28H	20H

Bit position	Bit name	Function
7	FPCLRn	Specifies clearing of the CSIBUFn pointer. 0: No operation 1: Clear all CSIBUFn pointers to 0. Cautions 1. This bit is always 0 when it is read. 2. If 1 is written to the FPCLRn bit in the middle of transfer, transfer is aborted. Because all the CSIBUFn pointers are cleared to 0, the remaining data in the CSIBUFn register is ignored. If 1 is written to the FPCLRn bit, be sure to read the SFA3n register to check to see if all the CSIBUFn pointers have been correctly cleared to 0 (SFFULn bit = 0, SFEMPn bit = 1, SFPn3 to SFPn0 bits = 0000). Nothing happens even if 0 is written to the FPCLRn bit.

Remarks

1. n = 0, 1
2. The SFA3n register is set to 20H when the CSICAE_n bit of the CSIM3n register is cleared to 0.

Bit position	Bit name	Function
6	SFFULn	<p>This flag indicates the full status of the CSIBUFn register.</p> <p>0: CSIBUFn register has a vacancy. 1: CSIBUFn register is full.</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. This bit is cleared to 0 when the CSICAEn bit of the CSIM3n register is cleared to 0 and the FPCLR bit is set to 1. 2. If transfer of 16 data is specified in the continuous mode (TRMDn bit = 1 in the CSIM3n register) (SFNn3 to SFNn0 bits = 0000 in the SFN3n register), the SFFULn bit is set to 1 in the same way as in the single mode (TRMDn bit = 0 in the CSIM3n register) when 16 data are in the CSIBUFn register. If even one of the data has been completely transferred, the SFFULn bit is cleared to 0. However, this does not mean that the CSIBUFn register has a vacancy.
5	SFEMPn	<p>This flag indicates the empty status of the CSIBUFn register.</p> <p>0: Data is in CSIBUFn register. 1: CSIBUFn register is empty.</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. This flag is set to 1 when the CSICAEn bit of the CSIM3n register is cleared to 0 and the FPCLR bit is set to 1. 2. If the data written to the CSIBUFn register has been transferred, the SFEMPn bit is set to 1 (even if receive data is stored in the CSIBUFn register).
4	CSOTn	<p>This flag indicates transfer status.</p> <p>0: Idle status 1: Transfer or transfer start processing in progress</p> <p>Cautions</p> <ol style="list-style-type: none"> 1. This flag is cleared to 0 when the CSICAEn bit of the CSIM3n register is cleared to 0 and the FPCLRn bit is set to 1, or when the CTXEn and CRXEn bits of the CSIM3n register are cleared to 0. 2. This flag is "1" from when transfer is started until there is no more transfer data in the CSIBUFn register in the single mode (TRMDn bit = 0 in the CSIM3n register) or until the specified number of data has been transferred in the continuous mode (TRMDn bit = 1 in the CSIM3n register).

★ Remark n = 0, 1

Bit position	Bit name	Function
3 to 0	SFPn3 to SFPn0	<ul style="list-style-type: none"> • In the single mode (TRMDn bit = 0 in the CSIM3n register), the “number of transfer data remaining in CSIBUFn register (CSIBUFn pointer value for writing – CSIBUFn pointer value for SIO n loading)” can be read. • In the continuous mode (TRMDn bit = 1 in the CSIM3n register), the “number of data completely transferred (value of CSIBUFn pointer for SIO n loading/storing)” can be read. If the SFPn3 to SFPn0 bits are 0H, however, the number of transferred data is as follows, depending on the setting of the SFEMPn bit. <ul style="list-style-type: none"> When SFEMPn bit = 0: Number of transferred data = 0 When SFEMPn bit = 1: Number of transferred data = 16 or status before starting transfer (before writing transfer data) <p>Caution These bits are cleared to 0 in synchronization with the operating clock when the FPCLRn bit = 1. However, the values of these bits are held until the CSICAEn bit of the CSIM3n register is cleared to 0 or the FPCLRn bit is set to 1.</p>

Remark n = 0, 1

(6) Transfer data length select registers 30, 31 (CSIL30, CSIL31)

The CSIL3n register is used to select the transfer data length of CSI3n.

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

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

Caution The CSIL3n register may be transferring data when the CTXEn or CRXEn bit of the CSIM3n register is 1. Before writing data to the CSIL3n register, be sure to clear the CTXEn and CRXEn bits to 0.

	7	6	5	4	3	2	1	0	Address	After reset
CSIL30	0	0	0	0	CCL03	CCL02	CCL01	CCL00	FFFFFD09H	00H
CSIL31	0	0	0	0	CCL13	CCL12	CCL11	CCL10	FFFFFD29H	00H

Bit position	Bit name	Function																																																							
3 to 0	CCLn3 to CCLn0	<p>Specifies a transfer data length.</p> <table border="1"> <thead> <tr> <th>CCLn3</th> <th>CCLn2</th> <th>CCLn1</th> <th>CCLn0</th> <th>Transfer data length</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>16 bits</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>8 bits</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>1</td> <td>9 bits</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>10 bits</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1</td> <td>11 bits</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>12 bits</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>13 bits</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>14 bits</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>15 bits</td> </tr> <tr> <td colspan="4">Other than above</td> <td>Setting prohibited</td> </tr> </tbody> </table> <p>Caution If a transfer data length other than 16 bits is specified (CCLn3 to CCLn0 bits = 0000), an undefined value is read to the higher excess bits of the SIRB3n and CSIBUFn registers (see 10.3.5 (3) Data transfer direction specification function).</p>	CCLn3	CCLn2	CCLn1	CCLn0	Transfer data length	0	0	0	0	16 bits	1	0	0	0	8 bits	1	0	0	1	9 bits	1	0	1	0	10 bits	1	0	1	1	11 bits	1	1	0	0	12 bits	1	1	0	1	13 bits	1	1	1	0	14 bits	1	1	1	1	15 bits	Other than above				Setting prohibited
CCLn3	CCLn2	CCLn1	CCLn0	Transfer data length																																																					
0	0	0	0	16 bits																																																					
1	0	0	0	8 bits																																																					
1	0	0	1	9 bits																																																					
1	0	1	0	10 bits																																																					
1	0	1	1	11 bits																																																					
1	1	0	0	12 bits																																																					
1	1	0	1	13 bits																																																					
1	1	1	0	14 bits																																																					
1	1	1	1	15 bits																																																					
Other than above				Setting prohibited																																																					

Remark n = 0, 1

(7) Transfer data number specification registers 30, 31 (SFN30, SFN31)

The SFN3n register is used to set the number of transfer data of CSI3n in the continuous mode (TRMDn bit = 1 in the CSIM3n register).

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

	7	6	5	4	3	2	1	0	Address	After reset
SFN30	-	-	-	-	SFN03	SFN02	SFN01	SFN00	FFFFFFD0CH	00H
SFN31	-	-	-	-	SFN13	SFN12	SFN11	SFN10	FFFFFFD2CH	00H

Bit position	Bit name	Function																																																																																					
3 to 0	SFNn3 to SFNn0	Specify the number of transfer data.																																																																																					
		<table border="1"> <thead> <tr> <th>SFNn3</th> <th>SFNn2</th> <th>SFNn1</th> <th>SFNn0</th> <th>Number of transfer data</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>16</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>2</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>3</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>4</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td><td>5</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>6</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td><td>7</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td><td>8</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>9</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>0</td><td>10</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td><td>11</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td><td>12</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>1</td><td>13</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>0</td><td>14</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td><td>15</td></tr> </tbody> </table>	SFNn3	SFNn2	SFNn1	SFNn0	Number of transfer data	0	0	0	0	16	0	0	0	1	1	0	0	1	0	2	0	0	1	1	3	0	1	0	0	4	0	1	0	1	5	0	1	1	0	6	0	1	1	1	7	1	0	0	0	8	1	0	0	1	9	1	0	1	0	10	1	0	1	1	11	1	1	0	0	12	1	1	0	1	13	1	1	1	0	14	1	1	1	1	15
		SFNn3	SFNn2	SFNn1	SFNn0	Number of transfer data																																																																																	
		0	0	0	0	16																																																																																	
		0	0	0	1	1																																																																																	
		0	0	1	0	2																																																																																	
		0	0	1	1	3																																																																																	
		0	1	0	0	4																																																																																	
		0	1	0	1	5																																																																																	
		0	1	1	0	6																																																																																	
		0	1	1	1	7																																																																																	
		1	0	0	0	8																																																																																	
		1	0	0	1	9																																																																																	
		1	0	1	0	10																																																																																	
		1	0	1	1	11																																																																																	
		1	1	0	0	12																																																																																	
		1	1	0	1	13																																																																																	
1	1	1	0	14																																																																																			
1	1	1	1	15																																																																																			
<p>Caution Writing data exceeding the value set by the SFNn3 to SFNn0 bits (number of CSI3n transfer data) to the CSIBUFn register is prohibited (data is ignored even if written).</p>																																																																																							

Remark n = 0, 1

10.3.4 Dedicated baud rate generators 0, 1 (BRG0, BRG1)

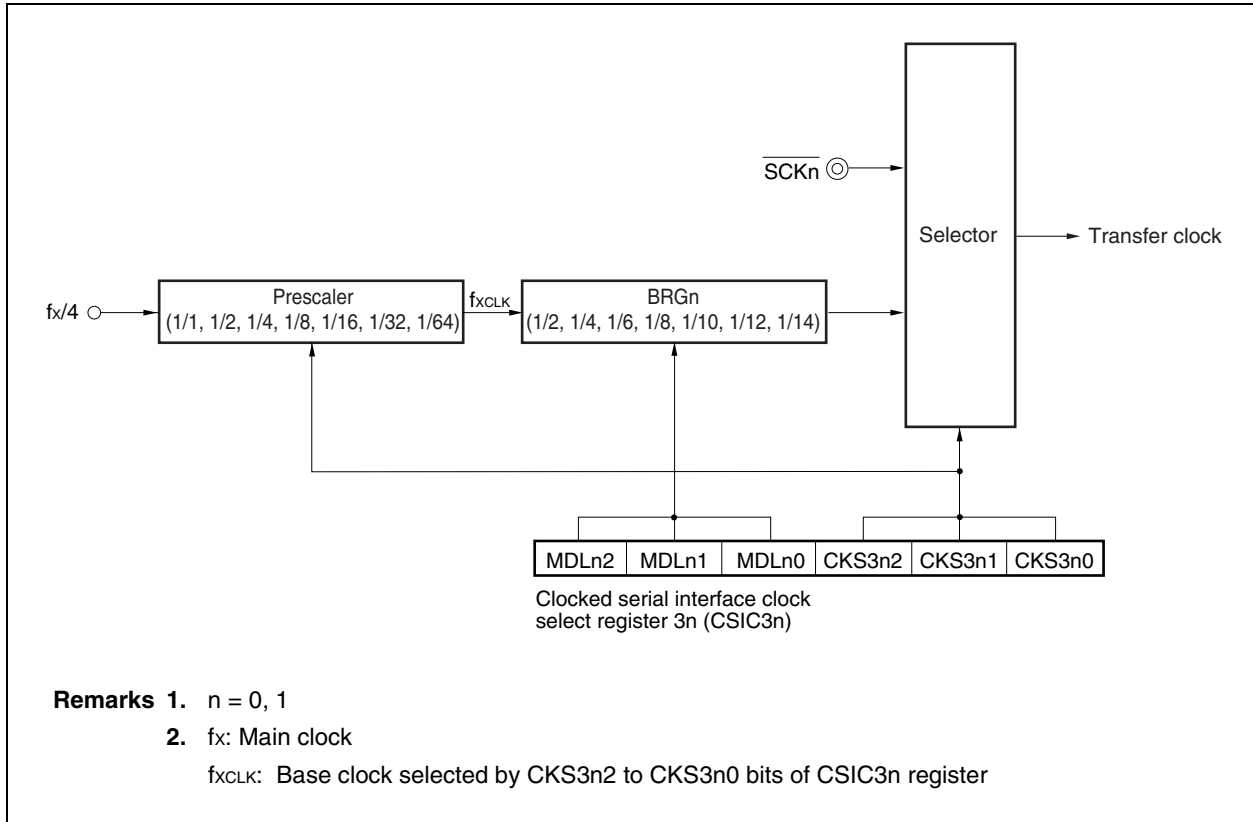
The transfer clock of CSI3n can be selected from the output of a dedicated baud rate generator or external clock (n = 0, 1).

The serial clock source is specified by the CSIC3n register.

In the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register), BRGn is selected as the clock source.

(1) Transfer clock

Figure 10-23. Transfer Clock of CSI3n



(2) Baud rate

The baud rate is calculated by the following expression.

$$\text{Baud rate} = \frac{F}{N \times 2^{(K+1)}} \text{ [bps]}$$

F = fx/4 (fx: main clock)

K = Value set by CKS3n2 to CKS3n0 bits of CSIC3n register (K = 0, 1, 2, ..., 6)

N = Value set by MDLn2 to MDLn0 bits of CSIC3n register (N = 1, 2, 3, ..., 7)

Cautions 1. If the CKS3n2 to CKS3n0 bits of the CSIC3n register are cleared to 000, setting the MDLn2 to MDLn0 bits of the CSIC3n register to 001 is prohibited.

2. Because the maximum transfer rate in the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register) is 5.5 Mbps, do not exceed this value.

Example: When CSI3n operates at 133 MHz, the maximum transfer rate is set when the CKS3n2 to CKS3n0 bits = 000 and the MDLn2 to MDLn0 bits = 011, and at 150 MHz, when the CKS3n2 to CKS3n0 bits = 000 and MDLn2 to MDLn0 bits = 100, in the CSIC3n register.

★

10.3.5 Operation

(1) Operation modes

Table 10-6. Operation Modes

TRMDn Bit	CKS3n2 to CKS3n0 Bits	CTXEn and CRXEn Bits	DIRn Bit	CSITn Bit	CSWEn Bit
Single mode	Master mode	Transmission/reception/ transmission and reception	MSB/LSB first	Enables/disables INTCSI3n delay mode	Disables transfer wait
					Enables transfer wait
Slave mode	–			–	
Consecutive mode	Master mode			Enables/disables INTCSI3n delay mode	Disables transfer wait
	Slave mode	–	–		

- Remarks 1.** CTXEn bit: Bit 6 of CSIM3n register
 CRXEn bit: Bit 5 of CSIM3n register
 TRMDn bit: Bit 4 of CSIM3n register
 DIRn bit: Bit 3 of CSIM3n register
 CSITn bit: Bit 2 of CSIM3n register
 CSWEn bit: Bit 1 of CSIM3n register
 CKS3n2 to CKS3n0 bits: Bits 2 to 0 of CSIC3n register
- 2.** n = 0, 1

(2) Function of CSI data buffer registers 0, 1 (CSIBUF0, CSIBUF1)

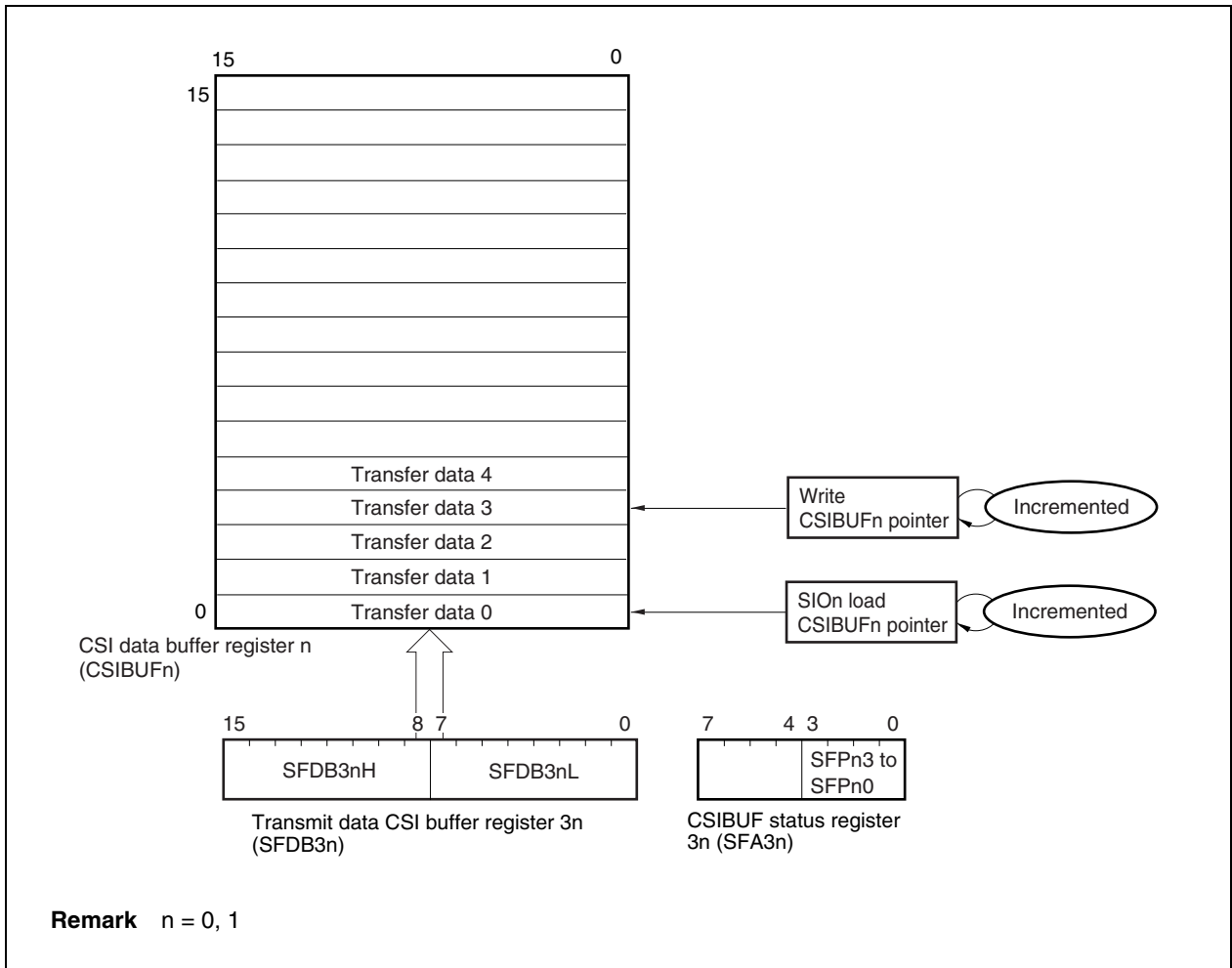
By consecutively writing the transmit data to the SFDB3n register from where it is transferred, the data can be stored in the CSIBUFn register while the CSIBUFn pointer for writing is automatically incremented (the CSIBUFn register size is 16 bits × 16) (n = 0, 1).

The condition under which transfer is to be started (SFEMPn bit = 0 in the SFA3n register) is satisfied when data is written to the lower 8 bits (SFDB3nL register) of the SFDB3n register. If a transfer data length of 9 bits or more is specified (CCLn3 to CCLn0 bits = 0000 or 1001 to 1111 in the CSIL3n register), data must be written to the SFDB3n register in 16-bit units or to the SFDB3nH and SFDB3nL registers, in that order, in 8-bit units. If the transfer data length is set to 8 bits (CCLn3 to CCLn0 bits = 1000 in the CSIL3n register), data must be written to the SFDB3nL register in 8-bit units or to the SFDB3n register in 16-bit units. (If data is written to the SFDB3nL register in 16-bit units, however, the higher 8 bits of the data (of the SFDB3nH register) are ignored and not transferred).

The SFFULn bit of the SFA3n register is set to 1 when 16 data exist in the CSIBUFn register and outputs a CSIBUFn overflow interrupt (INTCOVF3n) when the SFFULn bit = 1 and when the 17th transfer data is written.

Sixteen data exist in the CSIBUFn register in the single mode (TRMDn bit = 0 in the CSIM3n register) when “CSIBUFn pointer value for writing = CSIBUFn pointer value for SIO n loading, and SFFULn bit = 1 in the SFA3n register”. When the CSIBUFn pointer for SIO n loading is incremented after completion of transfer, the CSIBUFn register has a vacancy of one data (in the continuous mode (TRMDn bit = 1 in the CSIM3n register), the CSIBUFn register does not have a vacancy even if one data has been transferred).

Figure 10-24. Function of CSI Data Buffer Register n (CSIBUFn)



(3) Data transfer direction specification function

The data transfer direction can be changed by using the DIRn bit of the CSIM3n register (n = 0, 1).

(a) MSB first (DIRn bit = 0)

Figure 10-25. Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register) (1/2)

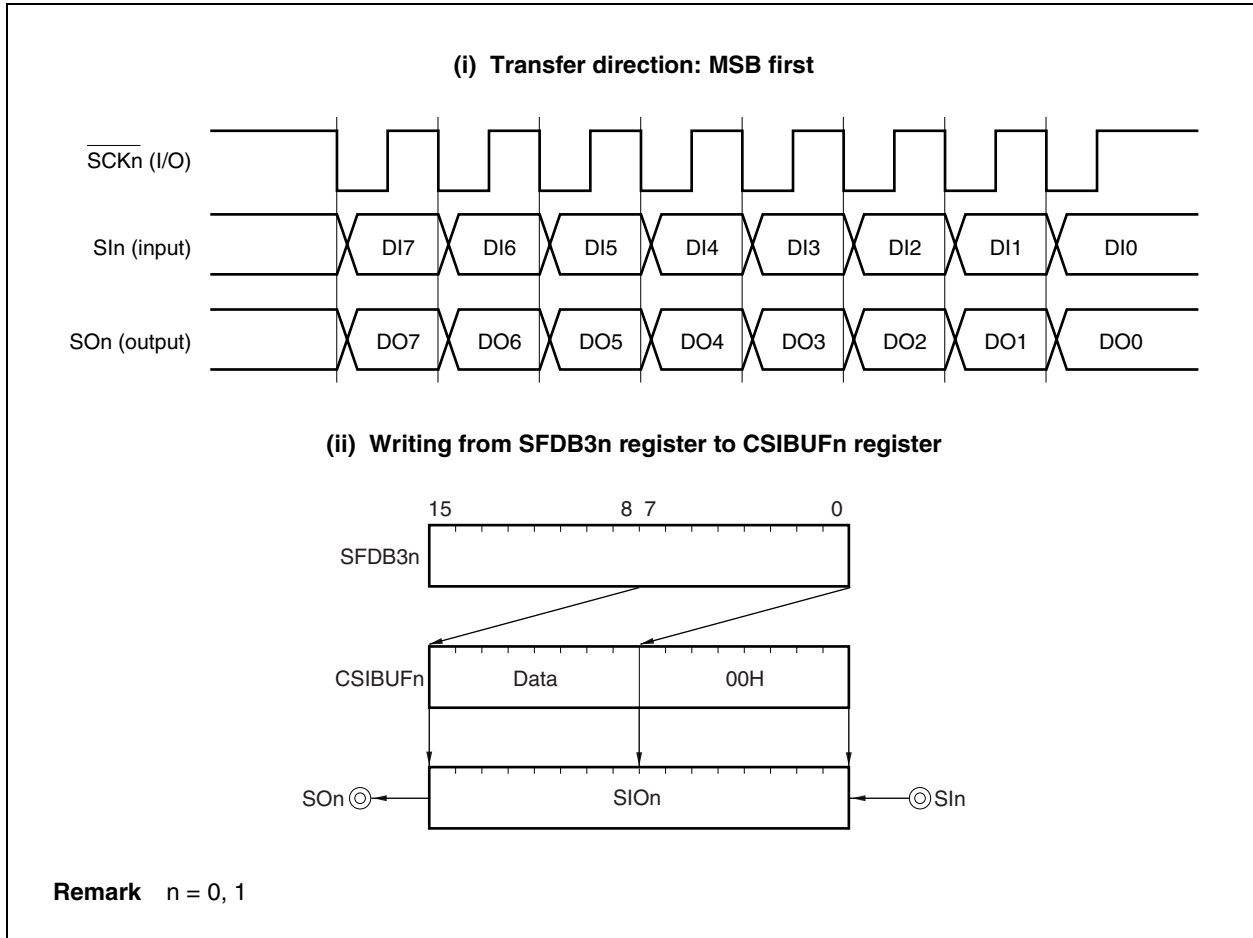
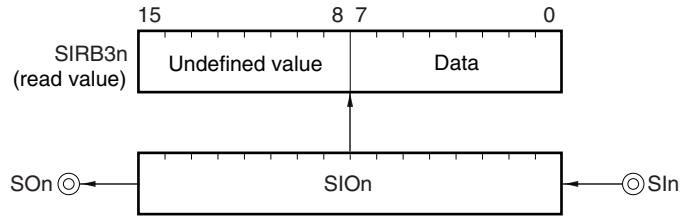


Figure 10-25. Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register),
Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register) (2/2)

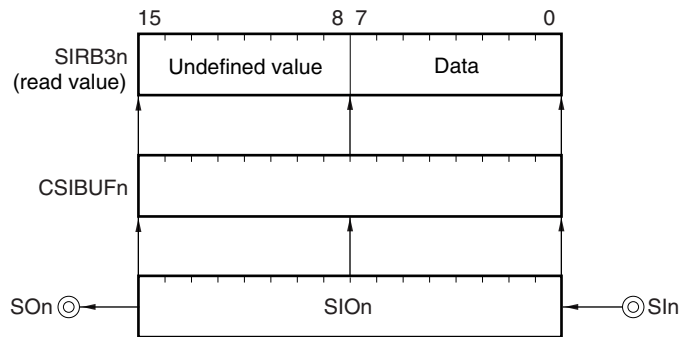
★

(iii) Reading from SIRB3n register (in single mode (TRMDn bit of CSIM3n register = 0))



★

(iv) Reading from SIRB3n register (in continuous mode (TRMDn bit of CSIM3n register = 1))



Remark n = 0, 1

(b) LSB first (DIRn bit = 1)

Figure 10-26. Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register),
Transfer Direction: LSB First (DIRn Bit = 1 in CSIM3n Register) (1/2)

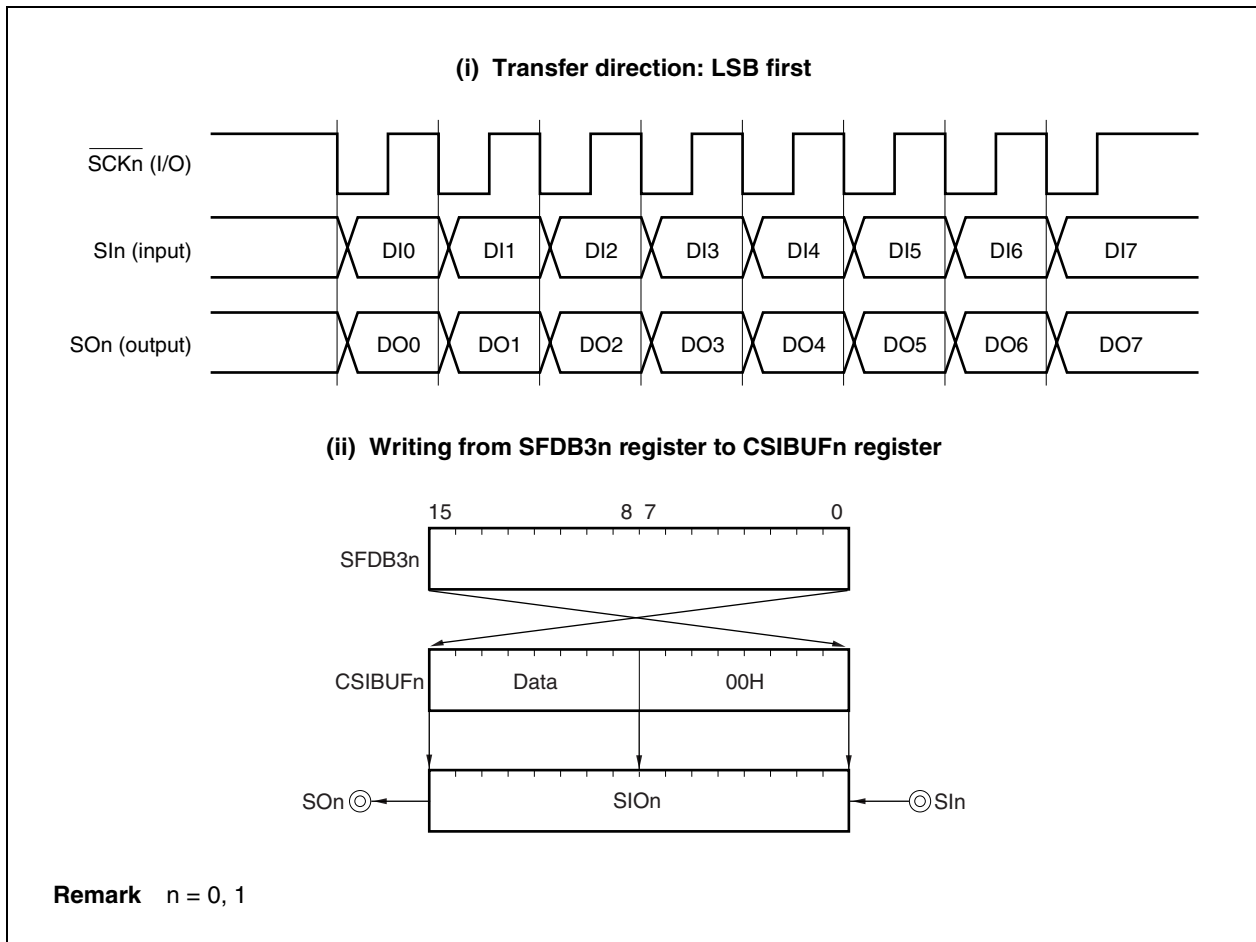
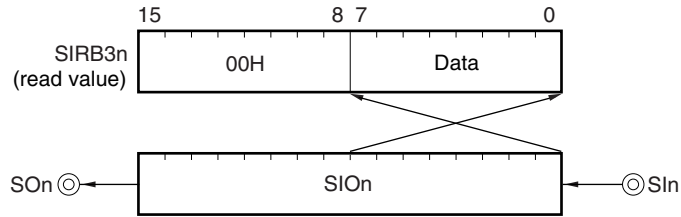


Figure 10-26. Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register),
Transfer Direction: LSB First (DIRn Bit = 1 in CSIM3n Register) (2/2)

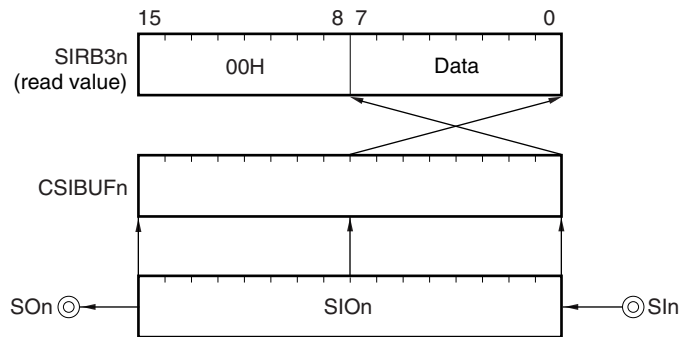
★

(iii) Reading from SIRB3n register (in single mode (TRMDn bit of CSIM3n register = 0))



★

(iv) Reading from SIRB3n register (in continuous mode (TRMDn bit of CSIM3n register = 1))

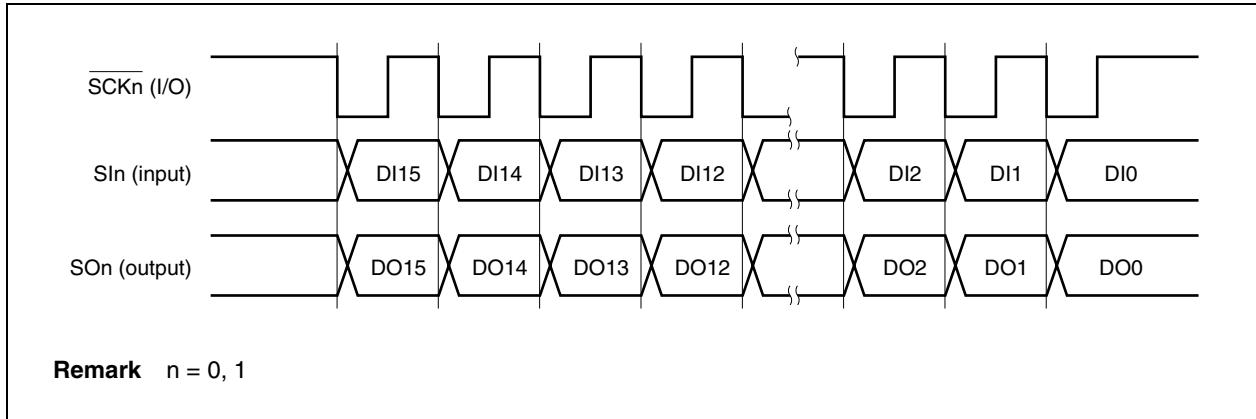


Remark n = 0, 1

(4) Transfer data length changing function

The transfer data length can be set from 8 to 16 bits in 1-bit units, by using the CCLn3 to CCLn0 bits of the CSIL3n register (n = 1, 0).

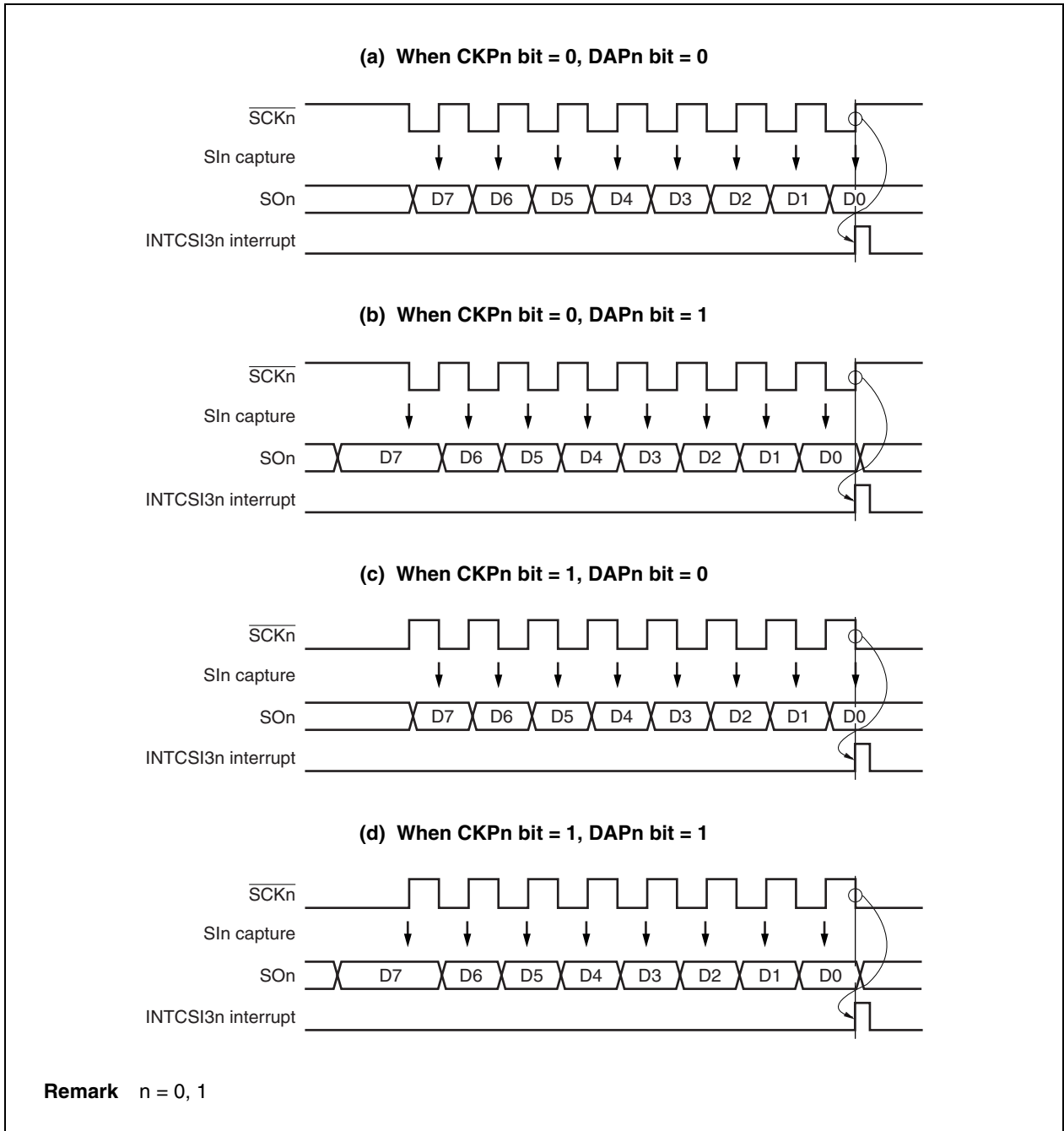
**Figure 10-27. Transfer Data Length: 16 Bits (CCLn3 to CCLn0 Bits = 0000 in CSIL3n Register),
Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register)**



(5) Function to select serial clock and data phase

The serial clock and data phase can be changed by using the CKPn and DAPn bits of the CSIC3n register (n = 0, 1).

Figure 10-28. Clock Timing

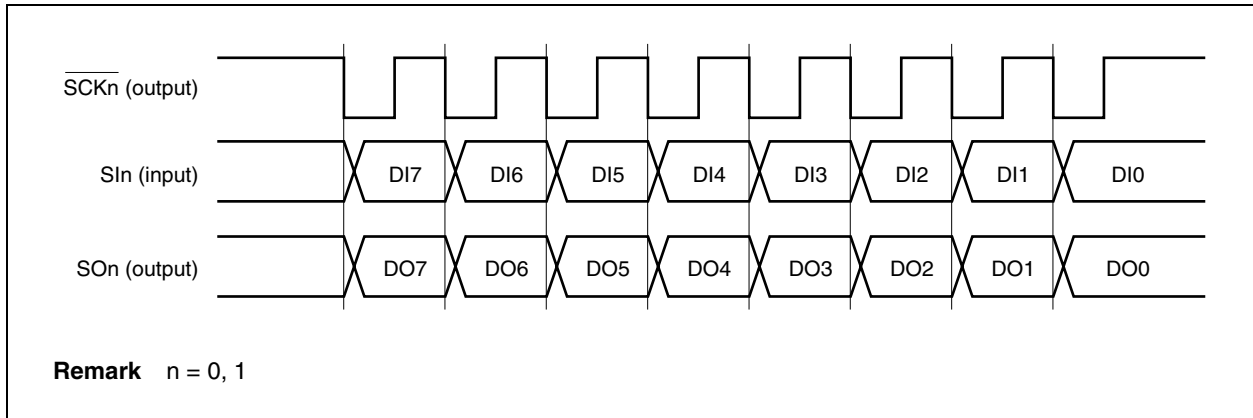


(6) Master mode

The master mode is set and data is transferred with the transfer clock output to the \overline{SCKn} pin when the \overline{SCKn} pin input is invalid) (n = 0, 1).

The default output level of the \overline{SCKn} pin is high when the CKPn bit of the CSIC3n register is 0, and low when the CKPn bit is 1.

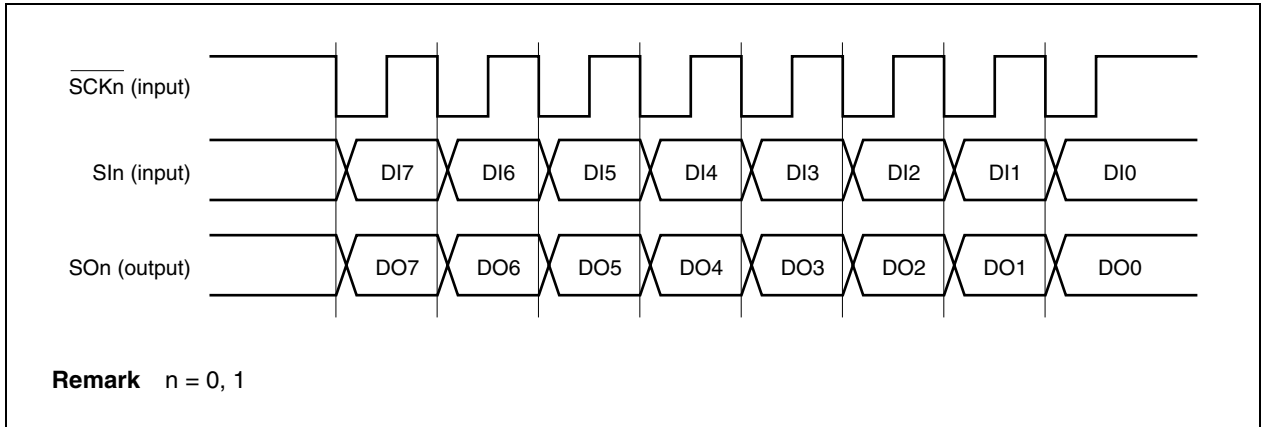
Figure 10-29. Master Mode (CKPn and DAPn Bits = 00 in CSIC3n Register, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits))



★ (7) Slave mode

The slave mode is set when the CKS3n2 to CKS3n0 bits of the CSIC3n register are set to 111, and data is transferred with the transfer clock input to the \overline{SCKn} pin (in the slave mode, it is recommended to set the MDLn2 to MDLn0 bits of the CSIC3n register to 000 and set the BRGn stop mode) (n = 0, 1).

★ **Figure 10-30. Slave Mode (CKPn and DAPn Bits = 00 in CSIC3n Register, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits))**



The conditions under which data can be transferred in the slave mode are listed in the table below.

Table 10-7. Conditions Under Which Data Can Be Transferred in Slave Mode

Transfer Mode		CTXEn Bit	CRXEn Bit	CSIBUFn Register	SIRB3n Register and SIOn Register
Single mode	Transmission mode	1	0	Data is in CSIBUFn register (SFEMPn bit = 0).	–
	Reception mode	0	1	Dummy data is in CSIBUFn register (SFEMPn bit = 0).	SIRB3n register or SIOn register is empty.
	Transmission/reception mode	1	1	Data is in CSIBUFn register (SFEMPn bit = 0).	
Consecutive mode	Transmission mode	1	0	Data is in CSIBUFn register (SFEMPn bit = 0).	–
	Reception mode	0	1	Dummy data is in CSIBUFn register (SFEMPn bit = 0).	–
	Transmission/reception mode	1	1	Data is in CSIBUFn register (SFEMPn bit = 0).	–

- Remarks**
- CTXEn bit: Bit 6 of CSIM3n register
 CRXEn bit: Bit 5 of CSIM3n register
 SFEMPn bit: Bit 5 of SFA3n register
 - n = 0, 1

(8) Transfer clock selection function

In the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register), the bit transfer rate can be selected by setting the CKS3n2 to CKS3n0 and MDLn2 to MDLn0 bits of the CSIC3n register (see **10.3.3 (2) Clocked serial interface clock select registers 30, 31 (CSIC30, CSIC31)**).

(9) Single mode

The single mode is set when the TRMDn bit of the CSIM3n register is 0 (n = 0, 1).

In this mode, transfer is started when the CTXEn bit or CRXEn bit is set to 1 and when data is in the CSIBUFn register (SFEMPn bit = 0 in the SFA3n register).

If no data is in the CSIBUFn register (SFEMPn bit = 1), transfer is kept waiting until transmit data or dummy data is written to the SFDB3n register.

When data is transferred to the CSIBUFn register while transmission or reception is enabled (CTXEn or CRXEn bit is 1), the CSOTn bit of the SFA3n register (transfer status flag) is set to 1. If transfer is not in the wait status, the transfer data indicated by the SIOOn load CSIBUFn pointer is loaded from the CSIBUFn register to the SIOOn register, and transfer processing is started.

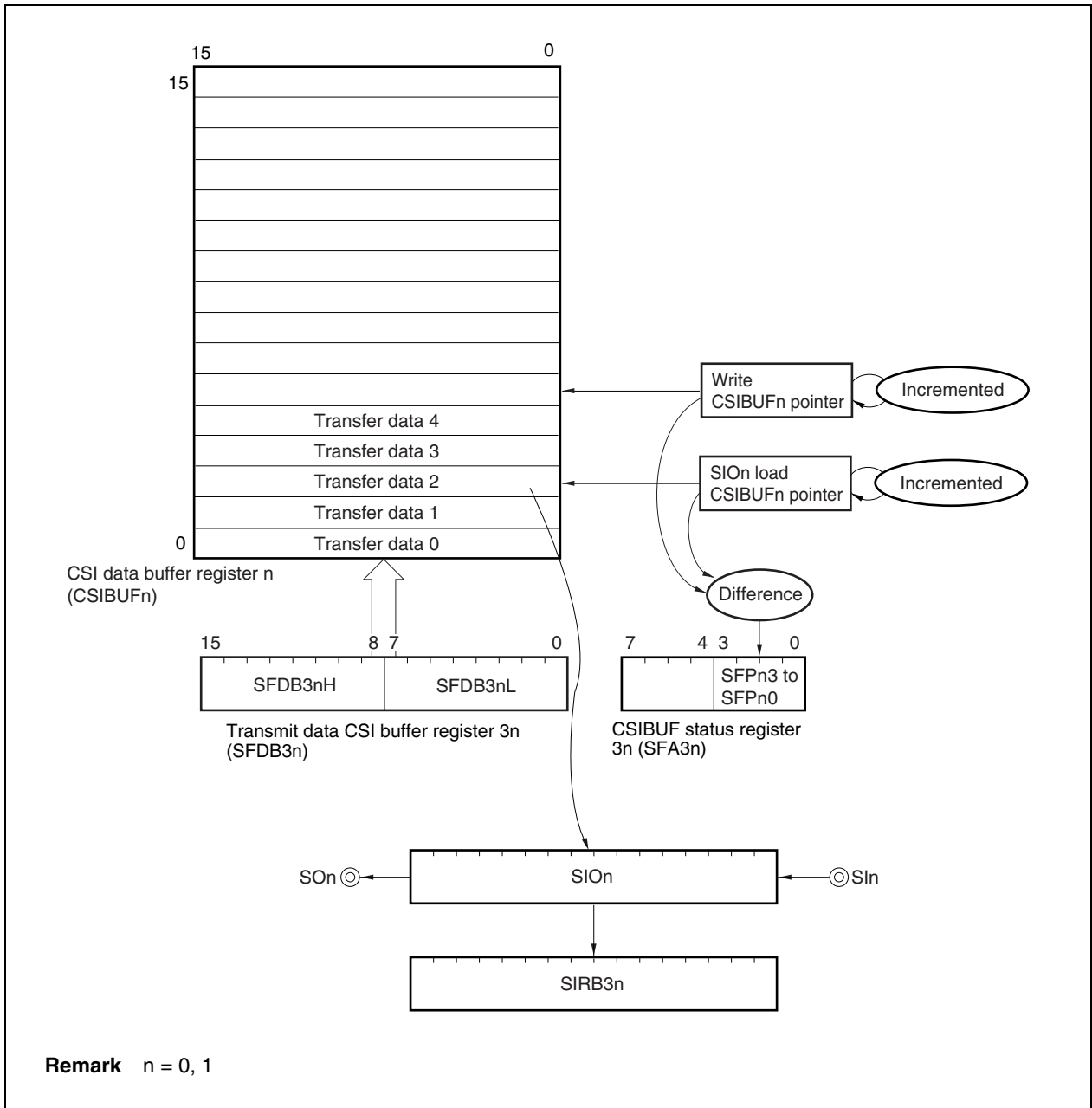
If the SIRB3n register is empty when one data has been transferred in the reception mode or transmission/reception mode, the received data is stored from the SIOOn register to the SIRB3n register, the transmission/reception completion interrupt (INTCSI3n) is output, and the SIOOn load CSIBUFn pointer is incremented. If transmit data or dummy data is stored in the CSIBUFn register, the next transfer processing is started. However, storing the receive data in the SIRB3n register, outputting the INTCSI3n interrupt, and incrementing the SIOOn load CSIBUFn pointer are held pending, until the previously received data is read from the SIRB3n register and the SIRB3n register becomes empty.

In the transmission mode, the INTCSI3n interrupt is output and the SIOOn load pointer is incremented when transfer processing of one data has been completed (the SIRB3n register is always empty because no data is stored from the SIOOn register to the SIRB3n register).

In all modes (transmission, reception, and transmission/reception modes), if the CSIBUFn register is empty (write CSIBUFn pointer value = SIOOn load CSIBUFn pointer value) when transfer processing of one data has been completed, the CSOTn bit is cleared to 0. The value of the “number of remaining data in the CSIBUFn register (write CSIBUFn pointer – SIOOn load pointer)” can always be read from the SFPn3 to SFPn0 bits of the SFA3n register.

Caution When writing data to the SFDB3n register, be sure to confirm that the SFFULn bit of the SFA3n register is 0. Even if data is written to this register when SFFULn bit is 1, the CSIBUFn overflow interrupt (INTCOVF3n) is output, and the written data is ignored.

Figure 10-31. Single Mode



(10) Continuous mode

The continuous mode is set when the TRMDn bit of the CSIM3n register is 1 (n = 0, 1).

In this mode, transfer is started when the CTXEn bit or CRXEn bit is 1 and when data is in the CSIBUFn register (SFEMPn bit = 0 in the SFA3n register). At this time, set the number of transfer data in advance by using the SFNn3 to SFNn0 bits of the SFN3n register. If 17 or more transfer data are written to the CSIBUFn register, the excess data are ignored and not transferred. Do not write data exceeding the number of transfer data specified by the SFNn3 to SFNn0 bits of the SFN3n register to the CSIBUFn register.

If no data is in the CSIBUFn register (SFEMPn bit = 1), transfer is kept waiting until transmit data or dummy data is written to the SFDB3n register.

If data is transferred to the CSIBUFn register when transmission or reception is enabled (CTXEn or CRXEn bit is 1), the CSOTn bit (transfer status flag) of the SFA3n register is set to 1 and the transfer data indicated by the SIOn load/store CSIBUFn pointer is loaded from the CSIBUFn register to SIOn register. Then transfer processing is started.

When transfer processing of one data is completed in the reception mode or transmission/reception mode, the received data is overwritten from the SIOn register to the transfer data in the CSIBUFn register indicated by the SIOn load/store CSIBUFn pointer, and then the pointer is incremented. By consecutively reading the transfer data from the SIRB3n register after all data in the CSIBUFn register have been transferred (when the INTCSI3n interrupt has occurred), the receive data can be sequentially read while the read CSIBUFn pointer is incremented.

In the transmission mode, the SIOn load/store CSIBUFn pointer is incremented when transfer processing of one data has been completed.

In all modes (transmission, reception, and transmission/reception modes), when data has been transferred by the value set by the SFNn3 to SFNn0 bits of the SFN3n register, the CSOTn bit is cleared to 0 and the transmission/reception completion interrupt (INTCSI3n) is output.

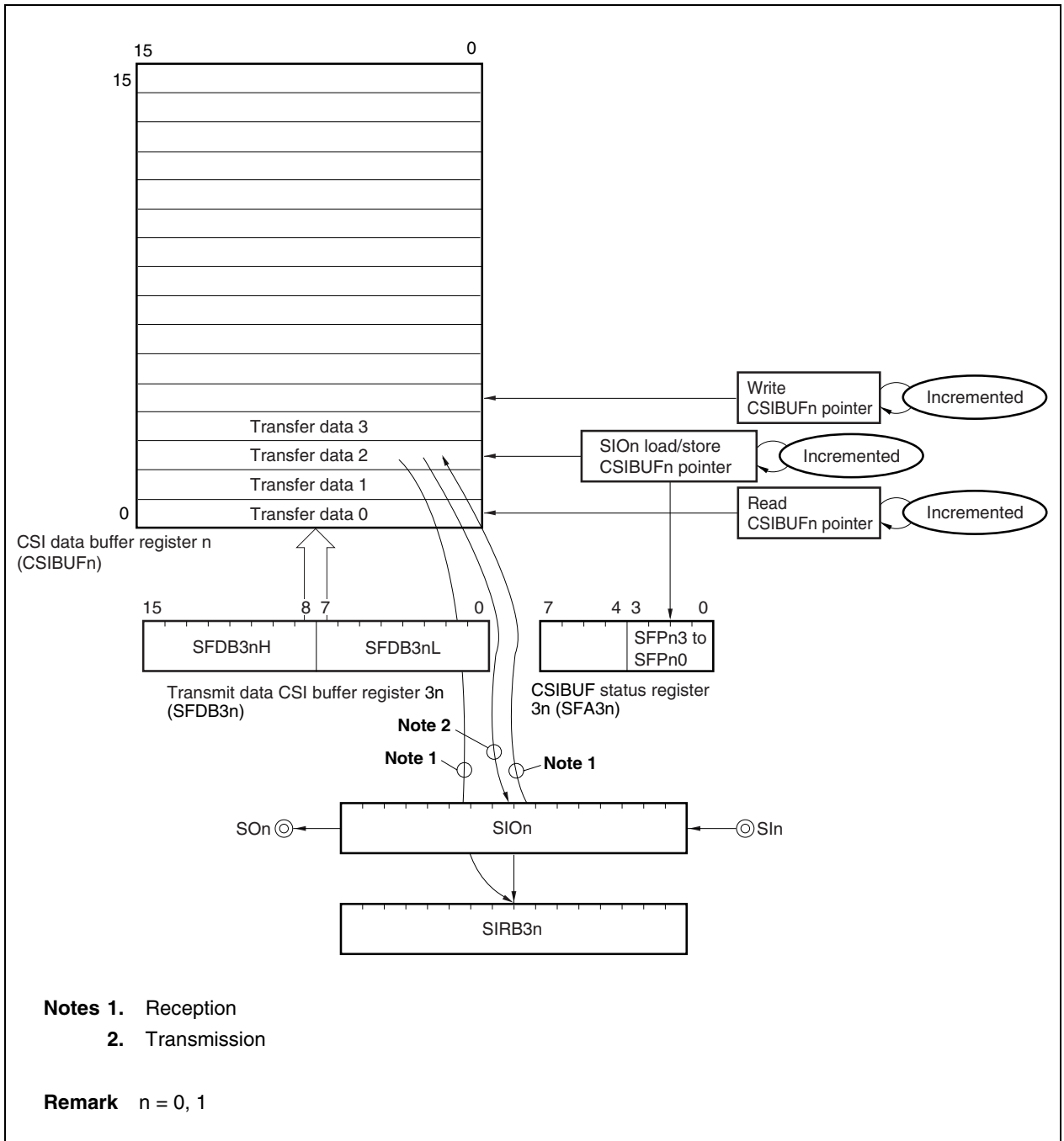
To transfer the next data, be sure to write 1 to the FPCLRn bit of the SFA3n register and clear all the CSIBUFn pointers to 0.

The “number of transferred data (SIOn load/store CSIBUFn pointer value)” can always be read from the SFPn3 to SFPn0 bits of the SFA3n register.

Caution The SFA3n register is in the same status when transfer data is written (before start of transfer) after the CSIBUFn pointer is cleared (FPCLRn bit = 1 in the SFA3n register) and when 16 data have been transferred (SFFULn bit = 0, SFEMPn bit = 1, SFPn3 to SFPn0 bits = 0000 in the SFA3n register).

★

Figure 10-32. Continuous Mode



(11) Transmission mode

The transmission mode is set when the CTXEn bit of the CSIM3n register is set to 1 and the CRXEn bit is cleared to 0. In this mode, transmission is started by a trigger that writes transmit data to the SFDB3n register or sets the CTXEn bit to 1 when transmit data is in the CSIBUFn register (n = 0, 1). Even in the single mode (TRMDn bit = 0 in the CSIM3n register), whether the SIRB3n or SIO n register is empty has nothing to do with starting transmission. The value input to the SIn pin during transmission is latched in the shift register (SIO n) but is not transferred to the SIRB3n and CSIBUFn registers at the end of transmission.

The transmission/reception completion interrupt (INTCSI3n) occurs immediately after data is sent out from the SIO n register.

(12) Reception mode

The reception mode is set when the CTXEn bit of the CSIM3n register is cleared to 0 and CRXEn bit is set to 1. In this mode, reception is started by using the processing of writing dummy data to the SFDB3n register as a trigger (n = 0, 1). In the single mode (TRMDn bit = 0 in the CSIM3n register), however, the condition of starting reception includes that the SIRB3n or SIO n register is empty. (If reception to the SIO n register is completed when the previously received data is held in the SIRB3n register without being read, the previously received data is read from the SIRB3n register and the wait status continues until the SIRB3n register becomes empty.) In the continuous mode, reception starts by writing dummy data of the number of receive data to the SFDB3n register with the first dummy data write processing taken as a trigger.

The SOn pin outputs a low level.

The transmission/reception completion interrupt (INTCSI3n) occurs immediately after receive data is transferred from the SIO n register to the SIRB3n register.

(13) Transmission/reception mode

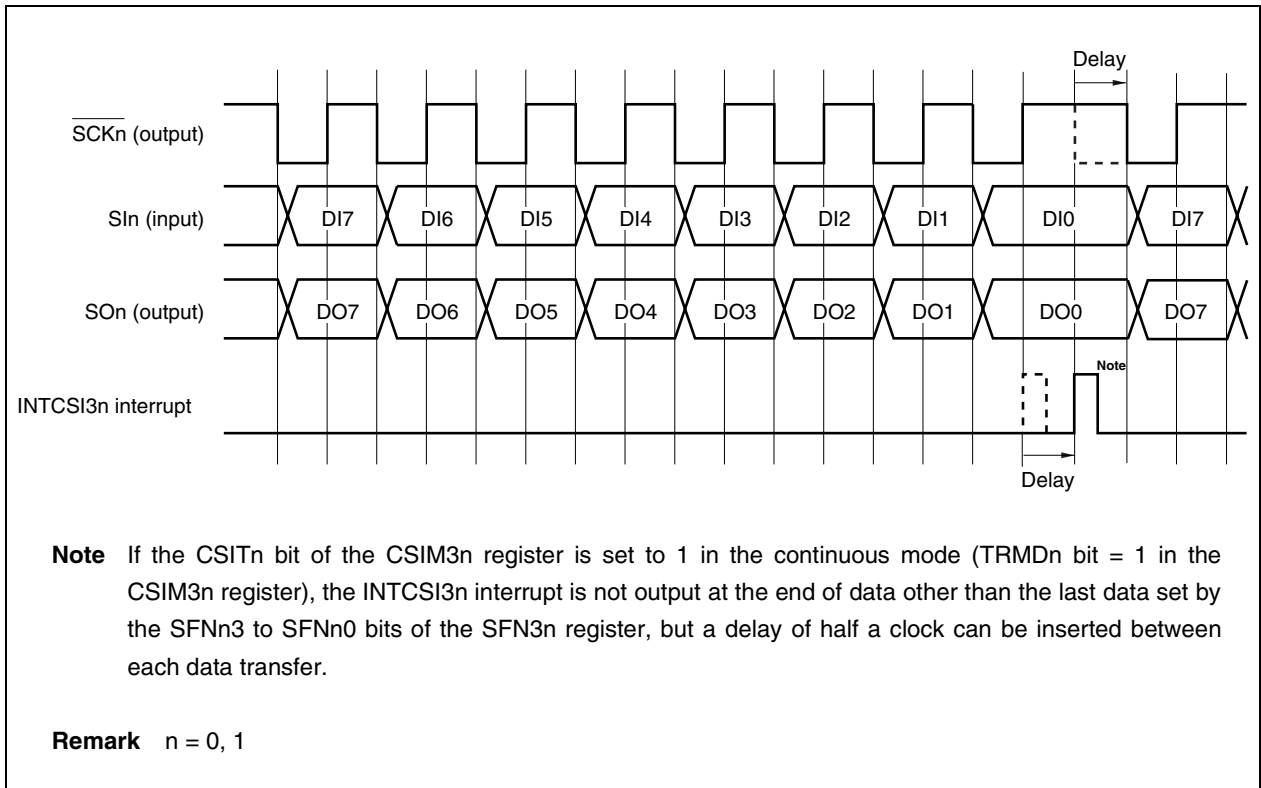
The transmission/reception mode is set when both the CTXEn and CRXEn bits of the CSIM3n register are set to 1. In this mode, transmission/reception is started by using the processing to write transmit data to the SFDB3n register as a trigger (n = 0, 1). In the single mode (TRMDn bit = 0 in the CSIM3n register), however, the condition of starting transmission/reception includes that the SIRB3n or SIO n register is empty. (If reception to the SIO n register is completed when the previously received data is held in the SIRB3n register without being read, the previously received data is read from the SIRB3n register and the wait status continues until the SIRB3n register becomes empty.)

(14) Delay control of transmission/reception completion interrupt (INTCSI3n)

In the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register), occurrence of the transmission/reception completion interrupt (INTCSI3n) can be delayed by half a clock (1/2 serial clock), depending on the setting (1) of the CSITn bit of the CSIM3n register. The CSITn bit is valid only in the master mode. In the slave mode (CKS3n2 to CKS3n0 bits = 111 in the CSIC3n register), setting the CSITn bit to 1 is prohibited (even if set, the INTCSI3n interrupt is not affected).

Caution If the CSITn bit of the CSIM3n register is set to 1 in the continuous mode (TRMDn bit = 1 in the CSIM3n register), the INTCSI3n interrupt is not output at the end of data other than the last data set by the SFNn3 to SFNn0 bits of the SFN3n register, but a delay of half a clock can be inserted between each data transfer.

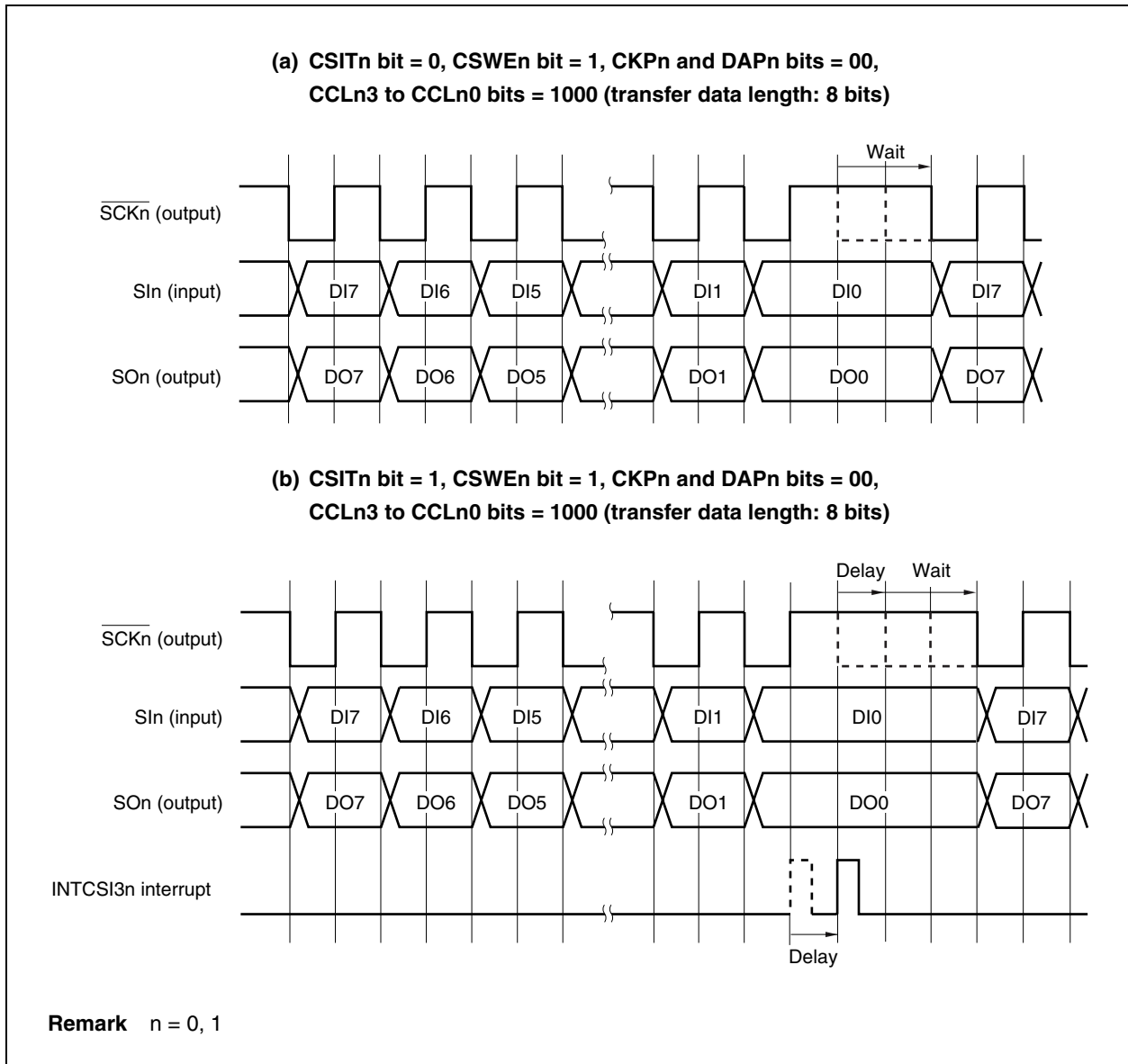
Figure 10-33. Delay Control of Transmission/Reception Completion Interrupt (INTCSI3n): CSITn Bit = 1 in CSIC3n Register, CSWEn Bit = 0, CKPn and DAPn Bits = 00, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits)



(15) Enabling/disabling transfer wait

In the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register), starting transfer can be delayed by one clock, depending on the setting (1) of the CSWEn bit of the CSIM3n register (CSWEn bit = 1 (transfer wait inserted)). The CSWEn bit is valid only in the master mode. In the slave mode (CKS3n2 to CKS3n0 bits = 111 in the CSIC3n register), setting the CSWEn bit to 1 is prohibited (even if set, transfer wait is not inserted).

Figure 10-34. Enabling/Disabling Transfer Wait



(16) Output pins

★

(a) \overline{SCKn} pin

The \overline{SCKn} pin outputs a high level when both the CTXEn and CRXEn bits of the CSIM3n register are 0 (n = 0, 1).

In the master mode (CKS3n2 to CKS3n0 bits = other than 111 in the CSIC3n register), this pin outputs the default level when the FPCLRn bit of the SFA3n register is set to 1.

Table 10-8. Default Output Level of \overline{SCKn} Pin

CKPn Bit	CKS3n2 to CKS3n0 Bits	Default Output Level of \overline{SCKn} Pin
0	111 (slave mode)	High level ^{Note}
	Other than 111 (master mode)	High level
1	111 (slave mode)	– (input)
	Other than 111 (master mode)	Low level

★

Note Default value after reset or value when CSICAEn bit = 0 in the CSIM3n register

Remarks 1. The output of the \overline{SCKn} pin changes if the CKPn bit is rewritten in the master mode.
 2. n = 0, 1

(b) SOn pin

The SOn pin outputs a low level when both the CTXEn and CRXEn bits of the CSIM3n register are 0 (n = 0, 1).

This pin outputs a low level when the FPCLRn bit of the SFA3n register is 1 (the previous value is retained only in the slave mode (CKS3n2 to CKS3n0 bits = 111 in the CSIC3n register) and when the DAPn bit = 0 in the CSIC3n register).

Table 10-9. Default Output Level of SOn Pin

Default Output Level of SOn Pin
Low level ^{Note}

Note Default value after reset or value when CSICAEn bit = 0 in the CSIM3n register

Remark n = 0, 1

(17) CSIBUFn overflow interrupt signal (INTCOVF3n)

The INTCOVF3n interrupt is output when 16 data exist in the CSIBUFn register and when the 17th data is written (to the SFDB3n or SFDB3nL register) (the 17th data is not written but ignored).

In the single mode (TRMDn bit = 0 in the CSIM3n register), 16 data exist in the CSIBUFn register when “write CSIBUFn pointer value = SIO n load CSIBUFn pointer value and SFFULn bit = 1 in the SFA3n register”. When transfer is completed and the SIO n load CSIBUFn pointer is incremented, the CSIBUFn register has one vacancy (the CSIBUFn register has no vacancy even when transfer of one data has been completed in the continuous mode (TRMDn bit = 1 in the CSIM3n register)).

10.3.6 Usage

(1) Single mode (in master mode and transmission mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
- <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
- <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable transmission by setting the CTXEn bit to 1.
- <7> Confirm that the SFFULn bit of the SFA3n register is 0, and then write transfer data to the SFDB3n register. If it is clearly known that the SFFULn bit is 0 because transfer data is written to that bit by the interrupt servicing routine of INTCSI3n, it is not always necessary to confirm that the SFFULn bit is 0.
- <8> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit of the SFA3n register is 1, and disable transmission by clearing the CTXEn bit of the CSIM3n register to 0 (end of transmission).

Caution To execute further transfer, repeat <7> before <8>.

(2) Single mode (in master mode and reception mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
- <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
- <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable reception by setting the CRXEn bit to 1.
- <7> Confirm that the SFFULn bit of the SFA3n register is 0, and then write dummy transfer data to the SFDB3n register (reception start trigger). If it is clearly known that the SFFULn bit is 0 because dummy transfer data is written to that bit by the interrupt servicing routine of INTCSI3n, it is not always necessary to confirm that the SFFULn bit is 0.
- <8> Confirm that the INTCSI3n interrupt has occurred, and then read the SIRB3n register.
- <9> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1, and disable reception by clearing the CRXEn bit of the CSIM3n register to 0 (end of reception).

- ★ **Cautions** 1. To execute further transfer, repeat <7> and <8> before <9>.
2. The SOn pin outputs a low level but this is invalid.

(3) Single mode (in master mode and transmission/reception mode)

- ★
 - <1> Set the external pins related to the CSI3n function to control mode.
 - <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
 - <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
 - <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
 - <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
 - <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable transmission/reception by setting the CTXEn and CRXEn bits to 1.
 - <7> Confirm that the SFFULn bit of the SFA3n register is 0, and then write transfer data to the SFDB3n register. If it is clearly known that the SFFULn bit is 0 because transfer data is written to that bit by the interrupt servicing routine of INTCSI3n, it is not always necessary to confirm that the SFFULn bit is 0.
 - <8> Confirm that the INTCSI3n interrupt has occurred, and then read the SIRB3n register.
 - <9> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1, and disable transmission/reception by clearing the CTXEn and CRXEn bits of the CSIM3n register to 0 (end of transmission/reception).

Caution To execute further transfer, repeat <7> and <8> before <9>.

(4) Single mode (in slave mode and transmission mode)

- ★
 - <1> Set the external pins related to the CSI3n function to control mode.
 - <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
 - <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
 - <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
 - <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
 - <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable transmission by setting the CTXEn bit to 1.
 - <7> Confirm that the SFFULn bit of the SFA3n register is 0, and then write transfer data to the SFDB3n register. If it is clearly known that the SFFULn bit is 0 because transfer data is written to that bit by the interrupt servicing routine of INTCSI3n, it is not always necessary to confirm that the SFFULn bit is 0.
 - <8> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1, and disable transmission by clearing the CTXEn bit of the CSIM3n register to 0 (end of transmission).

Caution To execute further transfer, repeat <7> before <8>.

(5) Single mode (in slave mode and reception mode)

- ★
 - <1> Set the external pins related to the CSI3n function to control mode.
 - <2> When the CSICAE_n bit of the CSIM3_n register is set to 1, supplying the operating clock is enabled.
 - <3> Specify the transfer mode by setting the CSIC3_n and CSIL3_n registers.
 - <4> Write 1 to the FPCLR_n bit of the SFA3_n register to clear all the CSIBUF_n pointers to 0.
 - <5> Confirm that the SFFUL_n bit = 0, SFEMP_n bit = 1, and SFP_n3 to SFP_n0 bits = 0000 in the SFA3_n register.
 - <6> Specify the transfer mode by using the TRMD_n, DIR_n, CSIT_n, and CSWEN bits of the CSIM3_n register and, at the same time, enable reception by setting the CRXEN bit to 1.
 - <7> Confirm that the SFFUL_n bit of the SFA3_n register is 0, and then write dummy transfer data to the SFDB3_n register (reception start trigger). If it is clearly known that the SFFUL_n bit is 0 because dummy transfer data is written to that bit by the interrupt servicing routine of INTCSI3_n, it is not always necessary to confirm that the SFFUL_n bit is 0.
 - <8> Confirm that the INTCSI3_n interrupt has occurred, and then read the SIRB3_n register.
 - <9> Confirm that the INTCSI3_n interrupt has occurred and the SFEMP_n bit is 1, and disable reception by clearing the CRXEN bit of the CSIM3_n register to 0 (end of reception).

- ★
 - Cautions 1. To execute further transfer, repeat <7> and <8> before <9>.**
 - 2. The SOn pin outputs a low level but this is invalid.**

(6) Single mode (in slave mode and transmission/reception mode)

- ★
 - <1> Set the external pins related to the CSI3n function to control mode.
 - <2> When the CSICAE_n bit of the CSIM3_n register is set to 1, supplying the operating clock is enabled.
 - <3> Specify the transfer mode by setting the CSIC3_n and CSIL3_n registers.
 - <4> Write 1 to the FPCLR_n bit of the SFA3_n register to clear all the CSIBUF_n pointers to 0.
 - <5> Confirm that the SFFUL_n bit = 0, SFEMP_n bit = 1, and SFP_n3 to SFP_n0 bits = 0000 in the SFA3_n register.
 - <6> Specify the transfer mode by using the TRMD_n, DIR_n, CSIT_n, and CSWEN bits of the CSIM3_n register and, at the same time, enable transmission/reception by setting the CTXEN and CRXEN bits to 1.
 - <7> Confirm that the SFFUL_n bit of the SFA3_n register is 0, and then write transfer data to the SFDB3_n register. If it is clearly known that the SFFUL_n bit is 0 because transfer data is written to that bit by the interrupt servicing routine of INTCSI3_n, it is not always necessary to confirm that the SFFUL_n bit is 0.
 - <8> Confirm that the INTCSI3_n interrupt has occurred, and then read the SIRB3_n register.
 - <9> Confirm that the INTCSI3_n interrupt has occurred and the SFEMP_n bit is 1, and disable transmission/reception by clearing the CTXEN and CRXEN bits of the CSIM3_n register to 0 (end of transmission/reception).

Caution To execute further transfer, repeat <7> and <8> before <9>.

(7) Continuous mode (in master mode and transmission mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
- <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
- <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable transmission by setting the CTXEn bit to 1.
- <7> Set the number of data to be transmitted by using the SFNn3 to SFNn0 bits of the SFN3n register.
- <8> Write transfer data to the SFDB3n register. Writing data exceeding the set value of the SFN3n register is prohibited.
- <9> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1. Then write 1 to the FPCLRn bit of the SFA3n register, and clear all the CSIBUFn pointers to 0 in preparation for the next transfer.
- <10> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <11> Disable transmission by clearing the CTXEn bit of the CSIM3n register to 0 (end of transmission).

Caution To execute further transfer, repeat <7> to <10> before <11>.

(8) Continuous mode (in master mode and reception mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
- <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
- <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable reception by setting the CRXEn bit to 1.
- <7> Set the number of data to be received by using the SFNn3 to SFNn0 bits of the SFN3n register.
- ★ <8> Write dummy transfer data of the number of receive data to the SFDB3n register. The first dummy transfer data write is the trigger to start reception. Writing dummy data exceeding the set value of the SFN3n register is prohibited.
- <9> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1. Then read the SIRB3n register (sequentially read the receive data stored in the CSIBUFn register).
- <10> Write 1 to the FPCLRn bit of the SFA3n register, and clear all the CSIBUFn pointers to 0 in preparation for the next transfer.
- <11> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <12> Disable reception by clearing the CRXEn bit of the CSIM3n register to 0 (end of reception).

- ★ **Cautions** 1. To execute further transfer, repeat <7> to <11> before <12>.
2. The SOn pin outputs a low level.

(9) Continuous mode (in master mode and transmission/reception mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAE_n bit of the CSIM3_n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3_n and CSIL3_n registers.
- <4> Write 1 to the FPCLR_n bit of the SFA3_n register to clear all the CSIBUF_n pointers to 0.
- <5> Confirm that the SFFUL_n bit = 0, SFEMP_n bit = 1, and SFP_n3 to SFP_n0 bits = 0000 in the SFA3_n register.
- <6> Specify the transfer mode by using the TRMD_n, DIR_n, CSIT_n, and CSWEN bits of the CSIM3_n register and, at the same time, enable transmission/reception by setting both the CTXEN and CRXEN bits to 1.
- <7> Set the number of data to be transmitted/received by using the SFN_n3 to SFN_n0 bits of the SFN3_n register.
- <8> Write transfer data to the SFDB3_n register. Writing data exceeding the set value of the SFN3_n register is prohibited.
- <9> Confirm that the INTCSI3_n interrupt has occurred and the SFEMP_n bit is 1. Then read the SIRB3_n register (sequentially read the receive data stored in the CSIBUF_n register).
- <10> Write 1 to the FPCLR_n bit of the SFA3_n register, and clear all the CSIBUF_n pointers to 0 in preparation for the next transfer.
- <11> Confirm that the SFFUL_n bit = 0, SFEMP_n bit = 1, and SFP_n3 to SFP_n0 bits = 0000 in the SFA3_n register.
- <12> Disable transmission/reception by clearing the CTXEN and CRXEN bits of the CSIM3_n register to 0 (end of transmission/reception).

Caution To execute further transfer, repeat <7> to <11> before <12>.

(10) Continuous mode (in slave mode and transmission mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAE_n bit of the CSIM3_n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3_n and CSIL3_n registers.
- <4> Write 1 to the FPCLR_n bit of the SFA3_n register to clear all the CSIBUF_n pointers to 0.
- <5> Confirm that the SFFUL_n bit = 0, SFEMP_n bit = 1, and SFP_n3 to SFP_n0 bits = 0000 in the SFA3_n register.
- <6> Specify the transfer mode by using the TRMD_n, DIR_n, CSIT_n, and CSWEN bits of the CSIM3_n register and, at the same time, enable transmission by setting the CTXEN bit to 1.
- <7> Set the number of data to be transmitted by using the SFN_n3 to SFN_n0 bits of the SFN3_n register.
- <8> Write transfer data to the SFDB3_n register. Writing data exceeding the set value of the SFN3_n register is prohibited.
- <9> Confirm that the INTCSI3_n interrupt has occurred and the SFEMP_n bit is 1. Then write 1 to the FPCLR_n bit of the SFA3_n register, and clear all the CSIBUF_n pointers to 0 in preparation for the next transfer.
- <10> Confirm that the SFFUL_n bit = 0, SFEMP_n bit = 1, and SFP_n3 to SFP_n0 bits = 0000 in the SFA3_n register.
- <11> Disable transmission by clearing the CTXEN bit of the CSIM3_n register to 0 (end of transmission).

Caution To execute further transfer, repeat <7> to <10> before <11>.

(11) Continuous mode (in slave mode and reception mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
- <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
- <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable reception by setting the CRXEn bit to 1.
- <7> Set the number of data to be received by using the SFNn3 to SFNn0 bits of the SFN3n register.
- ★ <8> Write dummy transfer data of the number of receive data to the SFDB3n register. The first dummy transfer data write is the trigger to start reception. Writing dummy data exceeding the set value of the SFN3n register is prohibited.
- <9> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1. Then read the SIRB3n register (sequentially read the receive data stored in the CSIBUFn register).
- <10> Write 1 to the FPCLRn bit of the SFA3n register, and clear all the CSIBUFn pointers to 0 in preparation for the next transfer.
- <11> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <12> Disable reception by clearing the CRXEn bit of the CSIM3n register to 0 (end of reception).

- ★ **Cautions 1. To execute further transfer, repeat <7> to <11> before <12>.**
2. The SOn pin outputs a low level.

(12) Continuous mode (in slave mode and transmission/reception mode)

- ★ <1> Set the external pins related to the CSI3n function to control mode.
- <2> When the CSICAEn bit of the CSIM3n register is set to 1, supplying the operating clock is enabled.
- <3> Specify the transfer mode by setting the CSIC3n and CSIL3n registers.
- <4> Write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0.
- <5> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <6> Specify the transfer mode by using the TRMDn, DIRn, CSITn, and CSWEn bits of the CSIM3n register and, at the same time, enable transmission/reception by setting both the CTXEn and CRXEn bits to 1.
- <7> Set the number of data to be transmitted/received by using the SFNn3 to SFNn0 bits of the SFN3n register.
- <8> Write transfer data to the SFDB3n register. Writing data exceeding the set value of the SFN3n register is prohibited.
- <9> Confirm that the INTCSI3n interrupt has occurred and the SFEMPn bit is 1. Then read the SIRB3n register (sequentially read the receive data stored in the CSIBUFn register).
- <10> Write 1 to the FPCLRn bit of the SFA3n register, and clear all the CSIBUFn pointers to 0 in preparation for the next transfer.
- <11> Confirm that the SFFULn bit = 0, SFEMPn bit = 1, and SFPn3 to SFPn0 bits = 0000 in the SFA3n register.
- <12> Disable transmission/reception by clearing the CTXEn and CRXEn bits of the CSIM3n register to 0 (end of transmission/reception).

Caution To execute further transfer, repeat <7> to <11> before <12>.

10.3.7 Cautions

The following points must be noted when using CSI3n (n = 0, 1).

- (1) The CSI3n unit is reset and CSI3n is stopped when the CSICAEn bit of the CSIM3n register is cleared to 0. To operate CSI3n, first set the CSICAEn bit to 1. Usually, before clearing the CSICAEn bit to 0, clear both the CTXEn and CRXEn bits to 0 (after the end of transfer).
- (2) Be sure to write 1 to the FPCLRn bit of the SFA3n register to clear all the CSIBUFn pointers to 0 before enabling transfer by setting the CTXEn or CRXEn bit of the CSIM3n register to 1. If the CTXEn or CRXEn bit is set to 1 without clearing the pointers, and if the previously transferred data remains in the CSIBUFn register, transferring that data is immediately started. If transfer data is set to the CSIBUFn register before transfer is enabled, transfer is started as soon as the CTXEn or CRXEn bit is set to 1.
- (3) If the SFA3n register is read immediately after data has been written to the SFDB3n and SFDB3nL registers when the main clock (fx) is used at 84 MHz or lower, the SFFULn, SFEMPn, and SFPn3 to SFPn0 bits of the SFA3n register may not change their values in time. If the SFA3n register is read before the SFFULn bit is set to 1 and a 17th data is written, the CSIBUFn overflow interrupt (INTCOVF3n) occurs.

CHAPTER 11 USB FUNCTION CONTROLLER (USBF)

The V850E/ME2 has an internal USB function controller (USBF) conforming to the Universal Serial Bus Specification.

11.1 Overview

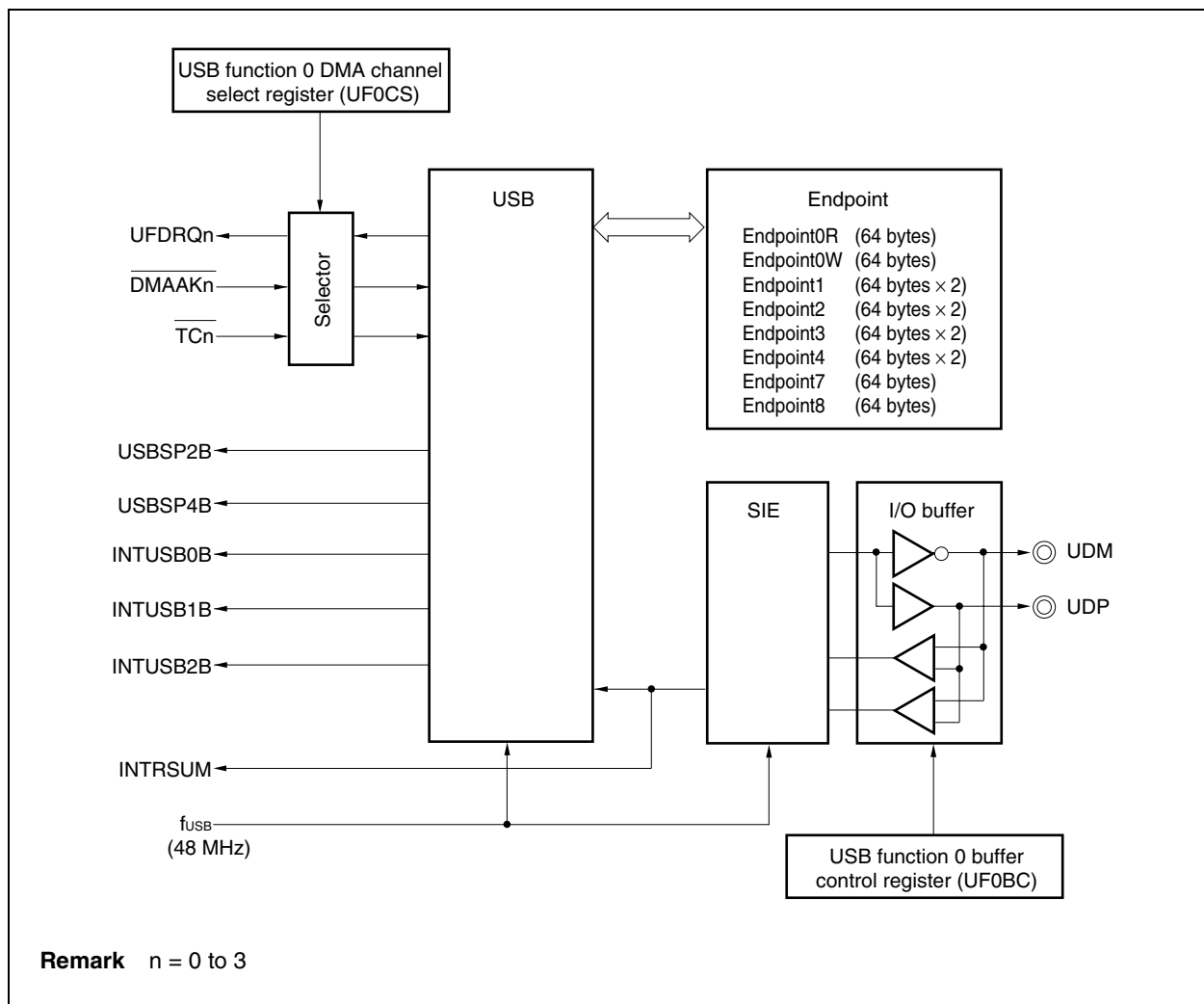
- Conforms to the Universal Serial Bus Specification.
- Supports 12 Mbps (full-speed) transfer
- Endpoint for transfer incorporated

Endpoint Name	FIFO Size (Bytes)	Transfer Type	Remark
Endpoint0 Read	64	Control transfer	–
Endpoint0 Write	64	Control transfer	–
Endpoint1	64 × 2	Bulk 1 transfer (IN)	2-buffer configuration
Endpoint2	64 × 2	Bulk 1 transfer (OUT)	2-buffer configuration
Endpoint3	64 × 2	Bulk 2 transfer (IN)	2-buffer configuration
Endpoint4	64 × 2	Bulk 2 transfer (OUT)	2-buffer configuration
Endpoint7	8	Interrupt 1 transfer	–
Endpoint8	8	Interrupt 2 transfer	–

- Clock: Clock input from UCK pin ($f_{USB} = 48 \text{ MHz}$)

★ **Caution** When using the USB function, be sure to set (1) the UCKCNT bit of the UCKC register.
If the registers related to the USB function while the UCKCNT bit is 0, 0 is read.

11.2 Configuration



11.3 Requests

11.3.1 Automatic requests

(1) Decode

The following tables show the request formats and correspondence between requests and decoded values.

Table 11-1. Request Format

Offset	Field Name	
0	bmRequestType	
1	bRequest	
2	wValue	Lower side
3		Higher side
4	wIndex	Lower side
5		Higher side
6	wLength	Lower side
7		Higher side

★

Table 11-2. Correspondence Between Requests and Decoded Values

Request	Offset	Decoded Value							Response			Data Stage	
		bmRequestType	bRequest	wValue		wIndex		wLength		Df	Ad		Cf
		0	1	3	2	5	4	7	6				
GET_INTERFACE	81H	0AH	00H	00H	00H	0nH	00H	01H	STALL	STALL	ACK NAK	√	
GET_CONFIGURATION	80H	08H	00H	00H	00H	00H	00H	01H	ACK NAK	ACK NAK	ACK NAK	√	
GET_DESCRIPTOR Device	80H	06H	01H	00H	00H	00H	XXH	XXH ^{Note 1}	ACK NAK	ACK NAK	ACK NAK	√	
GET_DESCRIPTOR Configuration	80H	06H	02H	00H	00H	00H	XXH	XXH ^{Note 1}	ACK NAK	ACK NAK	ACK NAK	√	
GET_STATUS Device	80H	00H	00H	00H	00H	00H	00H	02H	ACK NAK	ACK NAK	ACK NAK	√	
GET_STATUS Endpoint 0	82H	00H	00H	00H	00H	00H	80H	02H	ACK NAK	ACK NAK	ACK NAK	√	
GET_STATUS Endpoint X	82H	00H	00H	00H	00H	\$\$H	00H	02H	STALL	STALL	ACK NAK	√	
CLEAR_FEATURE Device ^{Note 2}	00H	01H	00H	01H	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×	
CLEAR_FEATURE Endpoint 0 ^{Note 2}	02H	01H	00H	00H	00H	00H	80H	00H	ACK NAK	ACK NAK	ACK NAK	×	
CLEAR_FEATURE Endpoint X ^{Note 2}	02H	01H	00H	00H	00H	\$\$H	00H	00H	STALL	STALL	ACK NAK	×	
SET_FEATURE Device ^{Note 3}	00H	03H	00H	01H	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×	
SET_FEATURE Endpoint 0 ^{Note 3}	02H	03H	00H	00H	00H	00H	80H	00H	ACK NAK	ACK NAK	ACK NAK	×	
SET_FEATURE Endpoint X ^{Note 3}	02H	03H	00H	00H	00H	\$\$H	00H	00H	STALL	STALL	ACK NAK	×	
SET_INTERFACE	01H	0BH	00H	0#H	00H	0?H	00H	00H	STALL	STALL	ACK NAK	×	
SET_CONFIGURATION ^{Note 4}	00H	09H	00H	00H 01H	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×	
SET_ADDRESS	00H	05H	XXH	XXH	00H	00H	00H	00H	ACK NAK	ACK NAK	ACK NAK	×	

Notes 1. If the wLength value is less than the prepared value, the wLength value is returned; if the wLength value is greater than the prepared value, the prepared value is returned.

2. The CLEAR_FEATURE request clears UF0 device status register L (UF0DSTL) and UF0 EPn status register L (UF0EnSL) (n = 0 to 4, 7, 8) when ACK is received in the status stage.

Remark √: Data stage is provided
 ×: Data stage is not provided

Notes 3. The SET_FEATURE request sets the UF0 device status register L (UF0DSTL) and UF0 EPn status register L (UF0EnSL) (n = 0 to 4, 7, 8) when ACK is received in the status stage. If the E0HALT bit of the UF0E0SL register is set, a STALL response is made in the status stage or data stage of control transfer for a request other than the GET_STATUS Endpoint0 request, SET_FEATURE Endpoint0 request, and a request generated by the CPUDEC interrupt request, until the CLEAR_FEATURE Endpoint0 request is received. A STALL response to an unsupported request does not set the E0HALT bit of the UF0E0SL register to 1, and the STALL response is cleared as soon as the next SETUP token has been received.

4. If the wValue is not the default value, an automatic STALL response is made.

Cautions 1. The sequence of control transfer defined by the Universal Serial Bus Specification is not satisfied under the following conditions. The operation is not guaranteed under these conditions.

- If an IN/OUT token is suddenly received without a SETUP stage
 - If DATA PID1 is sent in the data phase of the SETUP stage
 - If a token of 128 addresses or more is received
 - If the request data transmitted in the SETUP stage is of less than 8 bytes
2. An ACK response is made even when the host transmits data other than a Null packet in the status stage.
3. If the wLength value is 00H during control transfer (read) of FW processing, a Null packet is automatically transmitted for control transfer (without data). The FW request does not automatically transmit a Null packet.

Remarks 1. Df: Default state, Ad: Addressed state, Cf: Configured state

2. n = 0 to 4

It is determined by the setting of the UF0 active interface number register (UF0AIFN) whether a request with Interface number 1 to 4 is correctly responded to, depending on whether the Interface number of the target is valid or not.

3. \$\$: Valid Endpoint number including transfer direction

The valid Endpoint is determined by the currently set Alternate Setting number (see 11.4 (36) UF0 active alternative setting register (UF0AAS), (38) UF0 endpoint 1 interface mapping register (UF0E1IM) to (43) UF0 endpoint 8 interface mapping register (UF0E8IM)).

4. ? and #: Value transmitted from host (information on Interface numbers 0 to 4)

It is determined by the UF0 active interface number register (UF0AIFN) and UF0 active alternative setting register (UF0AAS) whether an Alternate Setting request corresponding to each Interface number is correctly responded to or not, depending on whether the Interface number and Alternate Setting of the target are valid or not.

(2) Processing

The processing of an automatic request in the Default state, Addressed state, and Configured state is described below.

Remark Default state: State in which an operation is performed with the Default address
 Addressed state: State after an address has been allocated
 Configured state: State after SET_CONFIGURATION wValue = 1 has been correctly received

(a) CLEAR_FEATURE() request

A STALL response is made in the status stage if the CLEAR_FEATURE() request cannot be cleared, if FEATURE does not exist, or if the target is an interface or an endpoint that does not exist. A STALL response is also made if the wLength value is other than 0.

- Default state: The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for Endpoint0. Otherwise, a STALL response is made in the status stage.
- Addressed state: The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for Endpoint0. Otherwise, a STALL response is made in the status stage.
- Configured state: The correct response is made when the CLEAR_FEATURE() request has been received only if the target is a device or a request for an endpoint that exists. Otherwise, a STALL response is made in the status stage.

When the CLEAR_FEATURE() request has been correctly processed, the corresponding bit of the UF0 CLR request register (UF0CLR) is set to 1, the EnHALT bit of the UF0 EPn status register L (UF0EnSL) is cleared to 0, and an interrupt is issued (n = 0 to 4, 7, 8). If the CLEAR_FEATURE() request is received when the subject is an endpoint, the toggle bit (that controls switching between DATA0 and DATA1) of the corresponding endpoint is always re-set to DATA0.

(b) GET_CONFIGURATION() request

A STALL response is made in the data stage if any of wValue, wIndex, or wLength is other than the values shown in Table 11-2.

- Default state: The value stored in the UF0 configuration register (UF0CNF) is returned when the GET_CONFIGURATION() request has been received.
- Addressed state: The value stored in the UF0CNF register is returned when the GET_CONFIGURATION() request has been received.
- Configured state: The value stored in the UF0CNF register is returned when the GET_CONFIGURATION() request has been received.

(c) GET_DESCRIPTOR() request

If the subject descriptor has a length that is a multiple of `wMaxPacketSize`, a Null packet is returned to indicate the end of the data stage. If the length of the descriptor at this time is less than the `wLength` value, the entire descriptor is returned; if the length of the descriptor is greater than the `wLength` value, the descriptor up to the `wLength` value is returned.

- **Default state:** The value stored in UF0 device descriptor register `n` (`UF0DDn`) and UF0 configuration/interface/endpoint descriptor register `m` (`UF0CIEm`) is returned (`n = 0` to `17`, `m = 0` to `255`) when the `GET_DESCRIPTOR()` request has been received.
- **Addressed state:** The value stored in the `UF0DDn` register and `UF0CIEm` register is returned when the `GET_DESCRIPTOR()` request has been received.
- **Configured state:** The value stored in the `UF0DDn` register and `UF0CIEm` register is returned when the `GET_DESCRIPTOR()` request has been received.

A descriptor of up to 256 bytes can be stored in the `UF0CIEm` register. To return a descriptor of more than 256 bytes, set the `CDCGDST` bit of the `UF0MODC` register to 1 and process the `GET_DESCRIPTOR()` request by FW.

Store the value of the total number of bytes of the descriptor set by the `UF0CIEm` register – 1 in the UF0 descriptor length register (`UF0DSCL`). The transfer data is controlled by the value of this data + 1 and `wLength`.

(d) GET_INTERFACE() request

If either of `wValue` and `wLength` is other than that shown in Table 11-2, or if `wIndex` is other than that set by the UF0 active interface number register (`UF0AIFN`), a STALL response is made in the data stage.

- **Default state:** A STALL response is made in the data stage when the `GET_INTERFACE()` request has been received.
- **Addressed state:** A STALL response is made in the data stage when the `GET_INTERFACE()` request has been received.
- **Configured state:** The value stored in the UF0 interface `n` register (`UF0IFn`) corresponding to the `wIndex` value is returned (`n = 0` to `4`) when the `GET_INTERFACE()` request has been received.

(e) GET_STATUS() request

A STALL response is made in the data stage if any of wValue, wIndex, or wLength is other than the values shown in Table 11-2. A STALL response is also made in the data stage if the target is an interface or an endpoint that does not exist.

- **Default state:** The value stored in the target status register^{Note} is returned only when the GET_STATUS() request has been received and when the request is for a device or Endpoint0. Otherwise, a STALL response is made in the data stage.
- **Addressed state:** The value stored in the target status register^{Note} is returned only when the GET_STATUS() request has been received and when the request is for a device or Endpoint0. Otherwise, a STALL response is made in the data stage.
- **Configured state:** The value stored in the target status register^{Note} is returned only when the GET_STATUS() request has been received and when the request is for a device or an endpoint that exists. Otherwise, a STALL response is made in the data stage.

Note The target status register is as follows.

- If the target is a device: UF0 device status register L (UF0DSTL)
- If the target is Endpoint0: UF0 EP0 status register L (UF0E0SL)
- If the target is Endpoint n: UF0 EPn status register L (UF0EnSL) (n = 1 to 4, 7, 8)

(f) SET_ADDRESS() request

A STALL response is made in the status stage if either of wIndex or wLength is other than the values shown in Table 11-2. A STALL response is also made if the specified device address is greater than 127.

- **Default state:** The device enters the Addressed state and changes the USB Address value to be input to SIE into a specified address value if the specified address is other than 0 when the SET_ADDRESS() request has been received. If the specified address is 0, the device remains in the Default state.
- **Addressed state:** The device enters the Default state and returns the USB Address value to be input to SIE to the default address if the specified address is 0 when the SET_ADDRESS() request has been received. If the specified address is other than 0, the device remains in the Addressed state, and changes the USB Address value to be input to SIE into a specified new address value.
- **Configured state:** The device remains in the Configured state and returns the USB Address value to be input to SIE to the default address if the specified address is 0 when the SET_ADDRESS() request has been received. In this case, the endpoints other than Endpoint0 remain valid, and control transfer (IN), control transfer (OUT), bulk transfer and interrupt transfer for an endpoint other than Endpoint0 are also acknowledged. If the specified address is other than 0, the device remains in the Configured state and changes the USB Address value to be input to SIE into a specified new address value.

(g) SET_CONFIGURATION() request

If any of wValue, wIndex, or wLength is other than the values shown in Table 11-2, a STALL response is made in the status stage.

- **Default state:** The CONF bit of the UF0 mode status register (UF0MODS) and the UF0 configuration register (UF0CNF) are set to 1 if the specified configuration value is 1 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 0, the CONF bit of the UF0MODS register and UF0CNF register are cleared to 0. In other words, the device skips the Addressed state and moves to the Configured state in which it responds to the Default address.
- **Addressed state:** The CONF bit of the UF0MODS register and UF0CNF register are set to 1 and the device enters the Configured state if the specified configuration value is 1 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 0, the device remains in the Addressed state.
- **Configured state:** The CONF bit of the UF0MODS register and UF0CNF register are set to 1 and the device returns to the Addressed state if the specified configuration value is 0 when the SET_CONFIGURATION() request has been received. If the specified configuration value is 1, the device remains in the Configured state.

If the SET_CONFIGURATION() request has been correctly processed, the target bit of the UF0 SET request register (UF0SET) is set to 1, and an interrupt is issued. All Halt Features are cleared after the SET_CONFIGURATION() request has been completed even if the specified configuration value is the same as the current configuration value. If the SET_CONFIGURATION() request has been correctly processed, the data toggle of all endpoints is always initialized to DATA0 again (it is defined that the default status, Alternative Setting 0, is set from when the SET_CONFIGURATION request is received to when the SET_INTERFACE request is received).

(h) SET_FEATURE() request

A STALL response is made in the status stage if the SET_FEATURE() request is for a Feature that cannot be set or does not exist, or if the target is an interface or an endpoint that does not exist. A STALL response is also made if the wLength value is other than 0.

- **Default state:** The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or Endpoint0. Otherwise, a STALL response is made in the status stage.
- **Addressed state:** The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or Endpoint0. Otherwise, a STALL response is made in the status stage.
- **Configured state:** The correct response is made when the SET_FEATURE() request has been received, only if the request is for a device or an endpoint that exists. Otherwise, a STALL response is made in the status stage.

When the SET_FEATURE() request has been correctly processed, the target bit of the UF0 SET request register (UF0SET) and the EnHALT bit of the UF0 EPn status register L (UF0EnSL) are set to 1, and an interrupt is issued (n = 0 to 4, 7, 8).

(i) SET_INTERFACE() request

If wLength is other than the values shown in Table 11-2, if wIndex is other than the value set to the UF0 active interface number register (UF0AIFN), or if wValue is other than the value set to the UF0 active alternative setting register (UF0AAS), a STALL response is made in the status stage.

- Default state: A STALL response is made in the status stage when the SET_INTERFACE() request has been received.
- Addressed state: A STALL response is made in the status stage when the SET_INTERFACE() request has been received.
- Configured state: Null packet is transmitted in the status stage when the SET_INTERFACE() request has been received.

When the SET_INTERFACE() request has been correctly processed, an interrupt is issued. All the Halt Features of the endpoint linked to the target Interface are cleared after the SET_INTERFACE() request has been cleared. The data toggle of all the endpoints related to the target Interface number is always initialized again to DATA0. When the currently selected Alternative Setting is to be changed by correctly processing the SET_INTERFACE() request, the FIFO of the endpoint that is affected is completely cleared, and all the related interrupt sources are also initialized.

When the SET_INTERFACE() request has been completed, the FIFO of all the endpoints linked to the target Interface are cleared. At the same time, Halt Feature and Data PID are initialized, and the related UF0 INT status n register (UF0ISn) is cleared to 0 (n = 0 to 4). (Only Halt Feature and Data PID are cleared when the SET_CONFIGURATION request has been completed.)

11.3.2 Other requests

(1) Response and processing

The following table shows how other requests are responded to and processed.

Table 11-3. Response and Processing of Other Requests

Request	Response and Processing
GET_DESCRIPTOR String	Generation of CPUDEC interrupt request
GET_STATUS Interface	Automatic STALL response
CLEAR_FEATURE Interface	Automatic STALL response
SET_FEATURE Interface	Automatic STALL response
all SET_DESCRIPTOR	Generation of CPUDEC interrupt request
All other requests	Generation of CPUDEC interrupt request

11.4 Register Configuration

11.4.1 Control registers

(1) UF0 EP0NAK register (UF0E0N)

This register controls NAK of Endpoint0 (except an automatically executed request).

This register can be read or written in 8-bit units (however, bit 0 can only be read).

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate a write signal that accesses the UF0FIC0 and UF0FIC1 registers from a read signal that accesses the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

While NAK is being transmitted to Endpoint0 Read, Endpoint2, and Endpoint4, a write access to the EP0NKR bit is ignored.

7	6	5	4	3	2	1	0	Address	After reset	
UF0E0N	0	0	0	0	0	0	EP0NKR	EP0NKW	FFFFFFE00H	00H

Bit position	Bit name	Function
1	EP0NKR	<p>This bit controls NAK to the OUT token to Endpoint0 (except an automatically executed request). It is automatically set to 1 by hardware when Endpoint0 has correctly received data. It is also cleared to 0 by hardware when the data of the UF0E0R register has been read by FW (counter value = 0).</p> <p>1: Transmit NAK. 0: Do not transmit NAK (default value).</p> <p>Set this bit to 1 by FW when data should not be received from the USB bus for some reason even when USBF is ready for receiving data. In this case, USBF continues transmitting NAK until this bit is cleared to 0 by FW. This bit is also cleared to 0 as soon as the UF0E0R register has been cleared.</p>
0	EP0NKW	<p>This bit indicates how NAK to the IN token to Endpoint0 is controlled (except an automatically executed request). This bit is automatically cleared to 0 by hardware when the data of Endpoint0 is transmitted and the host correctly receives the transmitted data. The data of the UF0E0W register is retained until this bit is cleared. Therefore, it is not necessary to rewrite this bit even in the case of a retransmission request that is made if the host could not receive data correctly. To send a short packet, be sure to set the E0DED bit of the UF0DEND register to 1. This bit is automatically set to 1 when the FIFO is full. As soon as the E0DED bit of the UF0DEND register is set to 1, the EP0NKW bit is automatically set to 1 at the same time.</p> <p>1: Do not transmit NAK. 0: Transmit NAK (default value).</p> <p>If control transfer enters the status stage while ACK cannot be correctly received in the data stage, this bit is cleared to 0 as soon as the UF0E0W register is cleared. This bit is also cleared to 0 when UF0E0W is cleared by FW.</p>

Next, the procedure of a SETUP transaction that uses IN/OUT tokens is explained below.

(a) When IN token is used (except a request automatically executed by hardware)

FW should be used to clear the PROT bit of the UF0IS1 register to 0 after receiving the CPUDEC interrupt and before reading data from the UF0E0ST register. Next, perform processing in accordance with the request and, if it is necessary to return data by an IN token, write data to the UF0E0W register. Confirm that the PROT bit of the UF0IS1 register is 0 after writing has been completed, and set the E0DED bit of the UF0DEND register to 1. The hardware sends out data at the first IN token after the EP0NKW bit has been set to 1. If the PROT bit of the UF0IS1 register is 1, it indicates that a SETUP transaction has occurred again before completion of control transfer. In this case, clear the PROTC bit of the UF0IC1 register to 0 and clear the PROT bit of the UF0IS1 register to 0, and then read data from the UF0E0ST register again. A request received later can be read.

(b) When OUT token is used (except a request automatically executed by hardware)

FW should be used to clear the PROT bit of the UF0IS1 register after receiving the CPUDEC interrupt and before reading data from the UF0E0ST register. Confirm that the PROT bit of the UF0IS1 register is 0 before reading data from the UF0E0R register. If the PROT bit is 1, it means that invalid data is retained. Clear the FIFO by FW (the EP0NKR bit is automatically cleared to 0). If the PROT bit of the UF0IS1 register is 0, read the data of the UF0E0L register and read as many data from the UF0E0R register as set. When reading data from the UF0E0R register has been completed (when the counter of the UF0E0R register has been cleared to 0), the hardware automatically clears the EP0NKR bit to 0.

(2) UF0 EP0NAKALL register (UF0E0NA)

This register controls NAK to all the requests of Endpoint0. It is also valid for automatically executed requests.

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

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0NA	0	0	0	0	0	0	0	EP0NKA	FFFFFFE01H	00H

Bit position	Bit name	Function
0	EP0NKA	<p>This bit controls NAK to a transaction other than a SETUP transaction to Endpoint0 (including an automatically executed request). This bit is manipulated by FW.</p> <p>1: Transmit NAK. 0: Do not transmit NAK (default value).</p> <p>This register is used to prevent a conflict between a write access by FW and a read access from SIE when the data used for an automatically executed request is to be changed. It postpones reflecting a write access on this bit from FW while an access from SIE is being made. Before rewriting the request data register from FW, confirm that this bit has been correctly set to 1.</p> <p>Setting this bit to 1 is reflected only in the following cases.</p> <ul style="list-style-type: none"> • Immediately after USBF has been reset and a SETUP token has never been received • Immediately after reception of Bus Reset and a SETUP token has never been received • PID of a SETUP token has been detected • The stage has been changed to the status stage <p>Clearing this bit to 0 is reflected immediately, except while an IN token is being received and a NAK response is being made.</p> <p>Setting the EP0NKA bit to 1 is reflected in the above four cases during Endpoint0 transfer, but it is reflected immediately after data has been written to the bit while Endpoint0 is transferring no data.</p>

(3) UF0 EPNK register (UF0EN)

This register controls NAK of endpoints other than Endpoint0.

This register can be read or written in 8-bit units (however, bits 6 to 4, 1, and 0 can only be read).

★

The BKO2NK bit can be written only when the BKO2NKM bit of the UF0ENM register is 1 and the BKO1NK bit can be written only when the BKO1NKM bit of the UF0ENM register is 1.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate a write signal that accesses the UF0FIC0 and UF0FIC1 registers from a read signal that accesses the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

While NAK is being transmitted to Endpoint0 Read, Endpoint2, and Endpoint4, a write access to the BKO1NK and BKO2NK bits is ignored.

Be sure to clear bit 7 to 0. If it is set to 1, the operation is not guaranteed.

(1/4)

7	6	5	4	3	2	1	0	Address	After reset
0	0	IT2NK	IT1NK	BKO2NK	BKO1NK	BKI2NK	BK11NK	FFFFFFE02H	00H

Bit position	Bit name	Function
5	IT2NK	This bit controls NAK to Endpoint8 (interrupt 2 transfer). It is automatically set to 1 and transmission is started when the UF0INT2 register has become full as a result of writing data to it. To send a short packet that does not make the FIFO full, set the IT2DEND bit of the UF0DEND register to 1. As soon as the IT2DEND bit has been set to 1, this bit is automatically set to 1. 1: Do not transmit NAK. 0: Transmit NAK (default value). This bit is also cleared to 0 when the UF0INT2 register has been cleared.
4	IT1NK	This bit controls NAK to Endpoint7 (interrupt 1 transfer). It is automatically set to 1 and transmission is started when the UF0INT1 register has become full as a result of writing data to it. To send a short packet that does not make the FIFO full, set the IT1DEND bit of the UF0DEND register to 1. As soon as the IT1DEND bit has been set to 1, this bit is automatically set to 1. 1: Do not transmit NAK. 0: Transmit NAK (default value). This bit is also cleared to 0 when the UF0INT1 register has been cleared.

Bit position	Bit name	Function
3	BKO2NK	<p>This bit controls NAK to Endpoint4 (bulk 2 transfer (OUT)).</p> <p>1: Transmit NAK. 0: Do not transmit NAK (default value).</p> <p>This bit is set to 1 only when the FIFO connected to the SIE side of the UF0BO2 register (64-byte FIFO of bank configuration) cannot receive data. It is cleared to 0 when a toggle operation is performed. The bank is changed (toggle operation) when the following conditions are satisfied.</p> <ul style="list-style-type: none"> • Data correctly received is stored in the FIFO connected to the SIE side. • The value of the FIFO counter connected to the CPU side is 0 (completion of reading). <p>FW should be used to read data of the UF0BO2L register when it has received the BLKO2DT interrupt request and read as many data from the UF0BO2 register as the value of that data. To not receive data from the USB bus for some reason even if USBF is ready to receive data, set this bit to 1 by FW. In this case, USBF keeps transmitting NAK until the FW clears this bit to 0. This bit is also cleared to 0 as soon as the UF0BO2 register has been cleared.</p>
2	BKO1NK	<p>This bit controls NAK to Endpoint2 (bulk 1 transfer (OUT)).</p> <p>1: Transmit NAK. 0: Do not transmit NAK (default value).</p> <p>This bit is set to 1 only when the FIFO connected to the SIE side of the UF0BO1 register (64-byte FIFO of bank configuration) cannot receive data. It is cleared to 0 when a toggle operation is performed. The bank is changed (toggle operation) when the following conditions are satisfied.</p> <ul style="list-style-type: none"> • Data correctly received is stored in the FIFO connected to the SIE side. • The value of the FIFO counter connected to the CPU side is 0 (completion of reading). <p>FW should be used to read data of the UF0BO1L register when it has received the BLKO1DT interrupt request and read as many data from the UF0BO1 register as the value of that data. To not receive data from the USB bus for some reason even if USBF is ready to receive data, set this bit to 1 by FW. In this case, USBF keeps transmitting NAK until the FW clears this bit to 0. This bit is also cleared to 0 as soon as the UF0BO1 register has been cleared.</p>

- Cautions**
1. If DMA is enabled while data is being read from the UF0BO2 register in the PIO mode, a DMA request is immediately issued.
 2. If the last data of the FIFO on the CPU side is read in the DMA transfer mode, the DMA request signal becomes inactive.
 3. If the TC signal is received in the DMA transfer mode, the DMA request signal becomes inactive.

Bit position	Bit name	Function
1	BKI2NK	<p>This bit controls NAK to Endpoint3 (bulk 2 transfer (IN)).</p> <p>1: Do not transmit NAK. 0: Transmit NAK (default value).</p> <p>This bit is cleared to 0 only when the FIFO connected to the SIE side of the UF0BI2 register (64-byte FIFO of bank configuration) cannot receive data. It is set to 1 when a toggle operation is performed (the data of the UF0BI2 register is retained until transmission has been correctly completed). The bank is changed (toggle operation) when the following conditions are satisfied.</p> <ul style="list-style-type: none"> • Data is correctly written to the FIFO connected to the CPU bus side (writing has been completed and the FIFO is full or the UF0DEND register is set). • The value of the FIFO counter connected to the SIE side is 0. <p>This bit is automatically set to 1 and data transmission is started when the FIFO on the CPU side becomes full and a FIFO toggle operation is performed as a result of writing data to the FIFO. However, if the FIFO on the CPU side becomes full as a result of writing data to it by DMA while the BKI2T bit of the UF0DEND register is cleared to 0, the toggle operation is not performed because the condition of the toggle operation is not satisfied until the BKI2DED bit of the UF0DEND register is set to 1. To send a short packet that does not make the FIFO on the CPU side full, set the BKI2DED bit to 1 after completing writing data. When the BKI2DED bit is set to 1, a toggle operation is performed and at the same time, this bit is automatically set to 1. This bit is also cleared to 0 as soon as the UF0BI2 register has been cleared.</p>

- Cautions 1.** If DMA is enabled while data is being written to the UF0BI2 register in the PIO mode, a DMA request is immediately issued.
2. If 64-byte data is written in the DMA transfer mode, the DMA request signal becomes inactive. If the BKI2NK bit is then set to 1, data is transmitted in synchronization with an IN token. The DMA request signal becomes active again as long as the DMA request is not masked as soon as the FIFO is toggled. If the BKI2NK bit is not set, data is not transmitted even if an IN token has been received. In this case, set the BKI2DED bit of the UF0DEND register to 1.
 3. If the TC signal is received in the DMA transfer mode, the DMA request signal becomes inactive. At the same time, the DMA request is masked. If the BKI2NK bit is not set to 1, data is not transmitted even if an IN token is received. When the BKI2DED bit of the UF0DEND register is set to 1 by FW, data is transmitted in synchronization with the IN token. To execute DMA transfer again, unmask the DMA request.

Bit position	Bit name	Function
0	BK11NK	<p>This bit controls NAK to Endpoint1 (bulk 1 transfer (IN)).</p> <p>1: Do not transmit NAK. 0: Transmit NAK (default value).</p> <p>This bit is cleared to 0 only when the FIFO connected to the SIE side of the UF0B11 register (64-byte FIFO of bank configuration) cannot receive data. It is set to 1 when a toggle operation is performed (the data of the UF0B11 register is retained until transmission has been correctly completed). The bank is changed (toggle operation) when the following conditions are satisfied.</p> <ul style="list-style-type: none"> • Data is correctly written to the FIFO connected to the CPU bus side (writing has been completed and the FIFO is full or the UF0DEND register is set). • The value of the FIFO counter connected to the SIE side is 0. <p>This bit is automatically set to 1 and data transmission is started when the FIFO on the CPU side becomes full and a FIFO toggle operation is performed as a result of writing data to the FIFO. However, if the FIFO on the CPU side becomes full as a result of writing data to it by DMA while the BK11T bit of the UF0DEND register is cleared to 0, the toggle operation is not performed because the condition of the toggle operation is not satisfied until the BK11DED bit of the UF0DEND register is set to 1. To send a short packet that does not make the FIFO on the CPU side full, set the BK11DED bit to 1 after completing writing data. When the BK11DED bit is set to 1, a toggle operation is performed and at the same time, this bit is automatically set to 1. This bit is also cleared to 0 as soon as the UF0B11 register has been cleared.</p>

- Cautions**
1. If DMA is enabled while data is being written to the UF0B11 register in the PIO mode, a DMA request is immediately issued.
 2. If 64-byte data is written in the DMA transfer mode, the DMA request signal becomes inactive. If the BK11NK bit is then set to 1, data is transmitted in synchronization with an IN token. The DMA request signal becomes active again as long as the DMA request is not masked as soon as the FIFO is toggled. If the BK11NK bit is not set, data is not transmitted even if an IN token has been received. In this case, set the BK11DED bit of the UF0DEND register to 1.
 3. If the TC signal is received in the DMA transfer mode, the DMA request signal becomes inactive. At the same time, the DMA request is masked. If the BK11NK bit is not set to 1, data is not transmitted even if an IN token is received. When the BK11DED bit of the UF0DEND register is set to 1 by FW, data is transmitted in synchronization with the IN token. To execute DMA transfer again, unmask the DMA request.

(4) UF0 EPNAK mask register (UF0ENM)

This register controls masking a write access to the UF0EN register.

This register can be read or written in 8-bit units (however, bits 6 to 4, 1, and 0 can only be read).

Be sure to clear bit 7 to 0. If it is set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
UF0ENM	0	0	0	0	BKO2NKM	BKO1NKM	0	0	FFFFFFE03H	00H

Bit position	Bit name	Function
3	BKO2NKM	This bit specifies whether a write access to bit 3 (BKO2NK) of the UF0EN register is masked or not. 1: Do not mask. 0: Mask (default value).
2	BKO1NKM	This bit specifies whether a write access to bit 2 (BKO1NK) of the UF0EN register is masked or not. 1: Do not mask. 0: Mask (default value).

(5) UF0 SNDSIE register (UF0SDS)

This register performs manipulation such as no handshake. It can directly manipulate the pins of SIE.

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

Be sure to clear bit 2 to 0. If it is set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
UF0SDS	0	0	0	0	SNDSTL	0	0	RSUMIN	FFFFFFE04H	00H

Bit position	Bit name	Function
3	SNDSTL	<p>This bit makes Endpoint0 issue a STALL handshake. Setting this bit to 1 if a request for CPUDEC processing is not supported by the system results in a STALL handshake response. If an unsupported wValue is sent by the SET_CONFIGURATION or SET_INTERFACE request, the hardware sets this bit to 1. If a problem occurs in Endpoint0 due to overrun of an automatically executed request, this bit is also set to 1. However, the E0HALT bit of the UF0E0SL register is not set to 1.</p> <p>1: Respond with STALL handshake. 0: Do not respond with STALL handshake (default value).</p> <p>This bit is cleared to 0 and the handshake response to the bus is other than STALL when the next SETUP token is received. To set the SNDSTL bit to 1 by FW, do not write data to the UF0E0W register. Depending on the timing of setting this bit, the STALL response is not made in time, and it may be made to the next transfer after a NAK response has been made.</p> <p>Setting this bit is valid only while an FW-executed request is under execution when this bit is set to 1. It is automatically cleared to 0 when the next SETUP token is received.</p> <p>Remark The SNDSTL bit is valid only for an FW-executed request.</p>
0	RSUMIN	<p>This bit outputs the Resume signal onto the USB bus. Writing this bit is invalid unless the RMWK bit of the UF0DSTL register is set to 1.</p> <p>1: Generate the Resume signal. 0: Do not generate the Resume signal (default value).</p> <p>While this bit is set to 1, the Resume signal continues to be generated. Clear this bit to 0 by FW after a specific time has elapsed. Because the signal is internally sampled at the clock, the operation is guaranteed only while CLK is supplied. Care must be exercised when CLK of the system is stopped.</p>

(6) UF0 CLR request register (UF0CLR)

This register indicates the target of the received CLEAR_FEATURE request.

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

This register is meaningful only when an interrupt request is generated. Each bit is set to 1 after completion of the status stage, and automatically cleared to 0 when this register is read.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0CLR	CLREP8	CLREP7	CLREP4	CLREP3	CLREP2	CLREP1	CLREP0	CLRDEV	FFFFFFE05H	00H

Bit position	Bit name	Function
7 to 1	CLREPN	These bits indicate that a CLEAR_FEATURE Endpoint n request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)
0	CLRDEV	This bit indicates that a CLEAR_FEATURE Device request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)

Remark n = 0 to 4, 7, 8

(7) UF0 SET request register (UF0SET)

This register indicates the target of the automatically processed SET_XXXX (except SET_INTERFACE) request.

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

This register is meaningful only when an interrupt request is generated. Each bit is set to 1 after completion of the status stage, and automatically cleared to 0 when this register is read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0SET	SETCON	0	0	0	0	SETEP	0	SETDEV	FFFFFFE06H	00H

Bit position	Bit name	Function
7	SETCON	This bit indicates that a SET_CONFIGURATION request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)
2	SETEP	This bit indicates that a SET_FEATURE Endpoint n request (n = 0 to 4, 7, 8) is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)
0	SETDEV	This bit indicates that a SET_FEATURE Device request is received and automatically processed. 1: Automatically processed 0: Not automatically processed (default value)

(8) UF0 EP status 0 register (UF0EPS0)

This register indicates the USB bus status and the presence or absence of register data.

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

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

It takes five USB clocks to reflect the status on this register after the UF0FIC0 and UF0FIC1 registers have been set. If it is necessary to read the status correctly, therefore, separate writing to the UF0FIC0 and UF0FIC1 registers from reading from the UF0EPS0, UF0EPS1, UF0EPS2, UF0E0N, and UF0EN registers by at least four USB clocks.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0EPS0	IT2	IT1	BKOUT2	BKOUT1	BKIN2	BKIN1	EP0W	EP0R	FFFFFFE07H	00H

Bit position	Bit name	Function
7, 6	ITn	These bits indicate that data is in the UF0INTn register (FIFO). By setting the ITnDED bit of the UF0DEND register to 1, the status in which data is in the UF0INTn register can be created even if data is not written to the register (Null data transmission). As soon as the ITnDED bit of the UF0DEND register is set to 1 even when the counter of the UF0INTn register is 0, this bit is set to 1 by hardware. It is cleared to 0 after correct transmission. 1: Data is in the register. 0: No data is in the register (default value).
5, 4	BKOUTn	These bits indicate that data is in the UF0BOn register (FIFO) connected to the CPU side. When the FIFO configuring the UF0BOn register is toggled, this bit is automatically set to 1 by hardware. It is automatically cleared to 0 by hardware when reading the UF0BOn register (FIFO) connected to the CPU side has been completed (counter value = 0). It is not set to 1 when Null data is received (toggling the FIFO does not take place either). 1: Data is in the register. 0: No data is in the register (default value).
3, 2	BKINn	These bits indicate that data is in the UF0BIn register (FIFO) connected to the CPU side. By setting the BKInDED bit of the UF0DEND register to 1, the status in which data is in the UF0BIn register can be created even if data is not written to the register (Null data transmission). As soon as the BKInDED bit of the UF0DEND register has been set to 1 while the counter of the UF0BIn register is 0, this bit is set to 1 by hardware. It is cleared to 0 when a toggle operation is performed. 1: Data is in the register. 0: No data is in the register (default value).

Remark n = 1, 2

Bit position	Bit name	Function
1	EP0W	<p>This bit indicates that data is in the UF0E0W register (FIFO). By setting the E0DED bit of the UF0DEND register to 1, the status in which data is in the UF0E0W register can be created even if data is not written to the register (Null data transmission). As soon as the E0DED bit of the UF0DEND register is set to 1 even when the counter of the UF0E0W register is 0, this bit is set to 1 by hardware. It is cleared to 0 after correct transmission.</p> <p>1: Data is in the register. 0: No data is in the register (default value).</p>
0	EP0R	<p>This bit indicates that data is in the UF0E0R register (FIFO). It is automatically cleared to 0 by hardware when reading the UF0E0R register (FIFO) has been completed (counter value = 0). It is not set to 1 if Null data is received.</p> <p>1: Data is in the register. 0: No data is in the register (default value).</p>

★ (9) UF0 EP status 1 register (UF0EPS1)

This register indicates the USB bus status and the presence or absence of register data.

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

	7	6	5	4	3	2	1	0	Address	After reset
UF0EPS1	RSUM	0	0	0	0	0	0	0	FFFFFFE08H	00H

Bit position	Bit name	Function
7	RSUM	<p>This bit indicates that the USB bus is in the Resume status. This bit is meaningful only when an interrupt request is generated.</p> <p>1: Suspend status 0: Resume status (default value)</p> <p>Because sampling is internally performed with the clock, the operation is guaranteed only when CLK is supplied. Care must be exercised when CLK of the system is stopped. The INTRSUM signal of SIE operates even when CLK is stopped. It can therefore be supported by making the interrupt control register (RSUMIC) valid or lowering the frequency of CLK to the USBF.</p> <p>This bit is automatically cleared to 0 when it is read.</p>

(10) UF0 EP status 2 register (UF0EPS2)

This register indicates the USB bus status and the presence or absence of register data.

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

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0EPS2	0	HALT8	HALT7	HALT4	HALT3	HALT2	HALT1	HALT0	FFFFFFE09H	00H

Bit position	Bit name	Function
6 to 0	HALTn	<p>These bits indicate that Endpoint n is currently stalled. They are set to 1 when a stall condition, such as occurrence of an overrun and reception of an undefined request, is satisfied. These bits are automatically set to 1 by hardware.</p> <p>1: Endpoint is stalled. 0: Endpoint is not stalled (default value).</p> <p>The SNDSTL bit is set to 1 as soon as the HALT0 bit has been set to 1 as a result of occurrence of an overrun or reception of an undefined request. If the next SETUP token is received in this status, the SNDSTL bit is cleared to 0 and, therefore, the HALT0 bit is also cleared to 0. If Endpoint0 is stalled by the SET_FEATURE Endpoint0 request, this bit is not cleared to 0 until the CLEAR_FEATURE Endpoint0 request is received or Halt Feature is cleared by FW. If the GET_STATUS Endpoint0, CLEAR_FEATURE Endpoint0, or SET_FEATURE Endpoint0 request is received, or if a request to be processed by FW is received due to the CPUDEC interrupt request, the HALT0 bit is masked and cleared to 0, until the next SETUP token is received.</p> <p>The HALTn bit is not cleared to 0 until Endpoint n receives the CLEAR_FEATURE Endpoint request, Halt Feature is cleared by the SET_INTERFACE or SET_CONFIGURATION request to the interface to which the endpoint is linked, or Halt Feature is cleared by FW. When the SET_INTERFACE or SET_CONFIGURATION request is correctly processed, the Halt Feature of all the target endpoints, except Endpoint0, is cleared after the request has been processed, even if the wValue is the same as the currently set value, and these bits are also cleared to 0. Halt Feature of Endpoint0 cannot be cleared if it is set because the STALL response is made in response to the SET_INTERFACE and SET_CONFIGURATION requests.</p>

Remark n = 0 to 4, 7, 8

(11) UF0 INT status 0 register (UF0IS0)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB0B signal becomes active.

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

If an interrupt request (INTUSB0B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC0 register.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS0	BUSRST	RSUSPD	0	SHORT	DMAED	SETRQ	CLRRQ	EPHALT	FFFFFFE10H	00H

Bit position	Bit name	Function
7	BUSRST	This bit indicates that Bus Reset has occurred. 1: Bus Reset has occurred (interrupt request is generated). 0: Not Bus Reset status (default value)
6	RSUSPD	This bit indicates that the Resume or Suspend status has occurred. Reference bit 7 of the UF0EPS1 register by FW. 1: Resume or Suspend status has occurred (interrupt request is generated). 0: Resume or Suspend status has not occurred (default value).
4	SHORT	This bit indicates that data is read from the FIFO of either the UF0BO1 or UF0BO2 register and that the USBSPnB signal (n = 2, 4) is active. It is valid only when the FIFO is full in the DMA mode. 1: USBSPnB signal is active (interrupt request is generated). 0: USBSPnB signal is not active (default value). Identify on which endpoint the operation is performed, by using the UF0DMS1 register. This bit is not automatically cleared to 0 even when the UF0DMS1 register is read by FW.

★

Bit position	Bit name	Function
3	DMAED	<p>This bit indicates that the DMA end (TC) signal for Endpoint n (n = 1 to 4, 7, 8) is active.</p> <p>1: DMA end signal for Endpoint n has been input (interrupt request is generated).</p> <p>0: DMA end signal for Endpoint n has not been input (default value).</p> <p>When this bit is set to 1, the DMA request signal for Endpoint n becomes inactive. The DMA request signal for Endpoint n does not become active unless FW enables DMA transfer.</p> <p>Use the UF0DMS0 register to confirm on which endpoint the operation is actually performed. However, this bit is not automatically cleared to 0 even if the UF0DMS0 register is read by FW.</p>
2	SETRQ	<p>This bit indicates that the SET_XXXX request to be automatically processed has been received and automatically processed (XXXX = ADDRESS, CONFIGURATION, FEATURE).</p> <p>1: SET_XXXX request to be automatically processed has been received (interrupt request is generated).</p> <p>0: SET_XXXX request to be automatically processed has not been received (default value).</p> <p>This bit is set to 1 after completion of the status stage. Reference the UF0SET register to identify what is the target of the request. This bit is not automatically cleared to 0 even if the UF0SET register is read by FW.</p> <p>The EPHALT bit is also set to 1 when the SET_FEATURE Endpoint request has been received.</p>
1	CLRRQ	<p>This bit indicates that the CLEAR_FEATURE request has been received and automatically processed.</p> <p>1: CLEAR_FEATURE request has been received (interrupt request is generated).</p> <p>0: CLEAR_FEATURE request has not been received (default value).</p> <p>This bit is set to 1 after completion of the status stage. Reference the UF0CLR register to identify what is the target of the request. This bit is not automatically cleared to 0 even if the UF0CLR register is read by FW.</p>
0	EPHALT	<p>This bit indicates that an endpoint has stalled.</p> <p>1: Endpoint has stalled (interrupt request is generated).</p> <p>0: Endpoint has not stalled (default value).</p> <p>This bit is also set to 1 when an endpoint has stalled by setting FW.</p> <p>Identify the endpoint that has stalled, by referencing the UF0EPS2 register. This bit is not automatically cleared to 0 even when the CLEAR_FEATURE Endpoint, SET_INTERFACE, or SET_CONFIGURATION request is received. It is not automatically cleared to 0, either, if the next SETUP token is received in case of overrun of Endpoint0.</p> <p>Caution Even if Halt Feature of Endpoint0 is set and this interrupt request is generated, bit 0 of the UF0EPS2 register is masked and cleared to 0 between when a SET_FEATURE Endpoint0, CLEAR_FEATURE Endpoint0, or GET_STATUS Endpoint0 request, or FW-processed request is received and when a SETUP token other than the above is received.</p>

(12) UF0 INT status 1 register (UF0IS1)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB0B signal becomes active.

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

If an interrupt request (INTUSB0B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC1 register. However, the SUCES and STG bits of the UF0IS1 register are automatically cleared to 0 when the next SETUP token has been received.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS1	0	E0IN	E0INDT	E0ODT	SUCES	STG	PROT	CPU DEC	FFFFFE11H	00H

Bit position	Bit name	Function
6	E0IN	This bit indicates that an IN token for Endpoint0 has been received and that the hardware has automatically transmitted NAK. 1: IN token is received and NAK is transmitted (interrupt request is generated). 0: IN token is not received (default value).
5	E0INDT	This bit indicates that data has been correctly transmitted from the UF0E0W register. 1: Transmission from UF0E0W register is completed (interrupt request is generated). 0: Transmission from UF0E0W register is not completed (default value). Data is transmitted in synchronization with the IN token next to the one that set the EPONKW bit of the UF0E0N register to 1. This bit is automatically set to 1 by hardware when the host correctly receives that data. It is also set to 1 even if the data is a Null packet. This bit is automatically cleared to 0 by hardware when the first write access is made to the UF0E0W register.
4	E0ODT	This bit indicates that data has been correctly received in the UF0E0R register. 1: Data is in UF0E0R register (interrupt request is generated). 0: Data is not in UF0E0R register (default value). This bit is automatically set to 1 by hardware when data has been correctly received. At the same time, bit 0 of the UF0EPS0 register is also set to 1. If a Null packet has been received, this bit is not set to 1. It is automatically cleared to 0 by hardware when the FW reads the UF0E0R register and the value of the UF0E0L register becomes 0.
3	SUCES	This bit indicates that either an FW-processed or hardware-processed request has been received and that the status stage has been correctly completed. 1: Control transfer has been correctly processed (interrupt request is generated). 0: Control transfer has not been processed correctly (default value). This bit is set to 1 upon completion of the status stage. It is automatically cleared to 0 by hardware when the next SETUP token is received. This bit is also set to 1 when data with Data PID of 0 (Null data) is received in the status stage of control transfer.

Bit position	Bit name	Function
2	STG	<p>This bit is set to 1 when the stage of control transfer has changed to the status stage. It is valid for both FW-processed and hardware-processed requests. This bit is also set to 1 when the stage of control transfer (without data) has changed to the status stage.</p> <p>1: Status stage (interrupt request is generated) 0: Not status stage (default value)</p> <p>This bit is automatically cleared to 0 by hardware when the next SETUP token is received.</p> <p>It is also set to 1 when the stage of control transfer has changed to the status stage while ACK cannot be correctly received in the data stage. In this case, the EPONKW bit of the UF0E0N register is also cleared to 0 as soon as the UF0E0W register has been cleared, if the FW is processing control transfer (read).</p>
1	PROT	<p>This bit indicates that a SETUP token has been received. It is valid for both FW-processed and hardware-processed requests.</p> <p>1: SETUP token is correctly received (interrupt request is generated). 0: SETUP token is not received (default value).</p> <p>This bit is set to 1 when data has been correctly received in the UF0E0ST register. Clear this bit to 0 by FW when the first read access is made to the UF0E0ST register. If it is not cleared to 0 by FW, reception of the next SETUP token cannot be correctly recognized.</p> <p>This bit is used to accurately recognize that a SETUP transaction has been executed again during control transfer. If the SETUP transaction is re-executed during control transfer and if a second request is executed by hardware, the CPUDEC bit is not set to 1, but the PROT bit can be used for recognition of the re-execution.</p>
0	CPUDEC	<p>This bit indicates that the UF0E0ST register has a request that is to be decoded by FW.</p> <p>1: Data is in UF0E0ST register (interrupt request is generated). 0: Data is not in UF0E0ST register (default value).</p> <p>This bit is automatically cleared to 0 by hardware when all the data of the UF0E0ST register is read.</p>

(13) UF0 INT status 2 register (UF0IS2)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB1B signal becomes active.

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

If an interrupt request (INTUSB1B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC2 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS2	BKl2IN	BKl2DT	BKl1IN	BKl1DT	0	0	IT2DT	IT1DT	FFFFFE12H	00H

Bit position	Bit name	Function
7, 5	BKInIN	These bits indicate that an IN token has been received in the UF0BIn register (Endpoint m) and that NAK has been returned. 1: IN token is received and NAK is transmitted (interrupt request is generated). 0: IN token is not received (default value).
6, 4	BKInDT	These bits indicate that the FIFO of the UF0BIn register (Endpoint m) has been toggled. This means that data can be written to Endpoint m. 1: FIFO has been toggled (interrupt request is generated). 0: FIFO has not been toggled (default value). The data written to Endpoint m is transmitted in synchronization with the IN token next to the one that set the BKInNK bit of the UF0EN register to 1. When the FIFO has been toggled and then data can be written from the CPU, this bit is automatically set to 1 by hardware. It is also set to 1 when the FIFO has been toggled, even if the data is a Null packet. This bit is automatically cleared to 0 by hardware when the first write access is made to the UF0BIn register.
1, 0	ITnDT	These bits indicate that data has been correctly received from the UF0INTn register (Endpoint x). 1: Transmission is completed (interrupt request is generated). 0: Transmission is not completed (default value). Data is transmitted in synchronization with the IN token next to the one that set the ITnNK bit of the UF0EN register to 1. This bit is automatically set to 1 by hardware when the host has correctly received that data. It is automatically cleared to 0 by hardware when the first write access is made to the UF0INTn register. This bit is also set to 1 even when the data is a Null packet.

Remark n = 1, 2
 m = 1 and x = 7 where n = 1
 m = 3 and x = 8 where n = 2

(14) UF0 INT status 3 register (UF0IS3)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB1B signal becomes active.

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

If an interrupt request (INTUSB1B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC3 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS3	BKO2FL	BKO2NL	BKO2 NAK	BKO2DT	BKO1FL	BKO1NL	BKO1 NAK	BKO1DT	FFFFFFE13H	00H

Bit position	Bit name	Function
7, 3	BKOnFL	<p>These bits indicate that data has been correctly received in the UF0BOn register (Endpoint m) and that both the FIFOs of the CPU and SIE hold the data.</p> <p>1: Received data is in both the FIFOs of the UF0BOn register (interrupt request is generated).</p> <p>0: Received data is not in the FIFO on the SIE side of the UF0BOn register (default value).</p> <p>If data is held in both the FIFOs of the CPU and SIE, these bits are automatically set to 1 by hardware. They are automatically cleared to 0 by hardware when the FIFO is toggled.</p>
6, 2	BKOnNL	<p>These bits indicate that a Null packet (packet with a length of 0) has been received in the UF0BOn register (Endpoint m).</p> <p>1: Null packet is received (interrupt request is generated).</p> <p>0: Null packet is not received (default value).</p> <p>These bits are set to 1 immediately after reception of a Null packet when the FIFO is empty. They are set to 1 when the FIFO on the CPU side has been completely read if data is in that FIFO.</p>
5, 1	BKOnNAK	<p>These bits indicate that an OUT token has been received to the UF0BOn register (Endpoint m) and that NAK has been returned.</p> <p>1: OUT token is received and NAK is transmitted (interrupt request is generated).</p> <p>0: OUT token is not received (default value).</p>

Remark n = 1, 2
 m = 2 where n = 1
 m = 4 where n = 2

Bit position	Bit name	Function
4, 0	BKOnDT	<p>These bits indicate that data has been correctly received in the UF0BOn register (Endpoint m).</p> <p>1: Reception has been completed correctly (interrupt request is generated).</p> <p>0: Reception has not been completed (default value).</p> <p>These bits are automatically set to 1 by hardware when data has been correctly received and the FIFO has been toggled. At the same time, the corresponding bits of the UF0EPS0 register are also set to 1. They are not set to 1 when the data is a Null packet.</p> <p>These bits are automatically cleared to 0 by hardware when the value of the UF0BOnL register becomes 0 as a result of reading the UF0BOn register by FW.</p> <p>These bits are automatically cleared to 0 when all the contents of the FIFO on the CPU side have been read. However, the interrupt request is not cleared if data is in the FIFO on the SIE side at this time, and the INTUSB1B signal does not become inactive. The signal is kept active if data is successively received.</p>

Remark n = 1, 2
m = 2 where n = 1
m = 4 where n = 2

(15) UF0 INT status 4 register (UF0IS4)

This register indicates the interrupt source. If the contents of this register are changed, the INTUSB2B signal becomes active.

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

If an interrupt request (INTUSB2B) is generated from USBF, the FW must read this register to identify the interrupt source.

Each bit of this register is forcibly cleared to 0 when 0 is written to the corresponding bit of the UF0IC4 register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IS4	0	0	SETINT	0	0	0	0	0	FFFFFFE14H	00H

Bit position	Bit name	Function
5	SETINT	<p>This bit indicates that the SET_INTERFACE request has been received and automatically processed.</p> <p>1: The request has been automatically processed (interrupt request is generated).</p> <p>0: The request has not been automatically processed (default value).</p> <p>The current setting of this bit can be identified by reading the UF0ASS or UF0IFn register (n = 0 to 4).</p>

(16) UF0 INT mask 0 register (UF0IM0)

This register controls masking of the interrupt sources indicated by the UF0IS0 register.

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

FW can mask occurrence of an interrupt request from USBF (INTUSB0B) by writing 1 to the corresponding bit of this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM0	BUS RSTM	RSU SPDM	0	SHORTM	DMA EDM	SET RQM	CLR RQM	EP HALTM	FFFFFE17H	00H

Bit position	Bit name	Function
7	BUSRSTM	This bit masks the Bus Reset interrupt. 1: Mask 0: Do not mask (default value)
6	RSUSPDM	This bit masks the Resume/Suspend interrupt. 1: Mask 0: Do not mask (default value)
4	SHORTM	This bit masks the Short interrupt. 1: Mask 0: Do not mask (default value)
3	DMAEDM	This bit masks the DMA_END interrupt. 1: Mask 0: Do not mask (default value)
2	SETRQM	This bit masks the SET_RQ interrupt. 1: Mask 0: Do not mask (default value)
1	CLRRQM	This bit masks the CLR_RQ interrupt. 1: Mask 0: Do not mask (default value)
0	EPHALTM	This bit masks the EP_Halt interrupt. 1: Mask 0: Do not mask (default value)

(17) UF0 INT mask 1 register (UF0IM1)

This register controls masking of the interrupt sources indicated by the UF0IS1 register.

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

FW can mask occurrence of an interrupt request from USBF (INTUSB0B) by writing 1 to the corresponding bit of this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM1	0	E0INM	E0 INDTM	E0 ODTM	SUCESM	STGM	PROTM	CPU DECM	FFFFFFE18H	00H

Bit position	Bit name	Function
6	E0INM	This bit masks the EP0IN interrupt. 1: Mask 0: Do not mask (default value)
5	E0INDTM	This bit masks the EP0INDT interrupt. 1: Mask 0: Do not mask (default value)
4	E0ODTM	This bit masks the EP0OUTDT interrupt. 1: Mask 0: Do not mask (default value)
3	SUCESM	This bit masks the Success interrupt. 1: Mask 0: Do not mask (default value)
2	STGM	This bit masks the Stg interrupt. 1: Mask 0: Do not mask (default value)
1	PROTM	This bit masks the Protect interrupt. 1: Mask 0: Do not mask (default value)
0	CPUDECM	This bit masks the CPUDEC interrupt. 1: Mask 0: Do not mask (default value)

(18) UF0 INT mask 2 register (UF0IM2)

This register controls masking of the interrupt sources indicated by the UF0IS2 register.

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

FW can mask occurrence of an interrupt request from USBF (INTUSB1B) by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM2	BKl2INM	BKl2 DTM	BKl1INM	BKl1 DTM	0	0	IT2DTM	IT1DTM	FFFFFE19H	00H

Bit position	Bit name	Function
7, 5	BKlInNM	These bits mask the BLKlInIN interrupt. 1: Mask 0: Do not mask (default value)
6, 4	BKlInDTM	These bits mask the BLKlInDT interrupt. 1: Mask 0: Do not mask (default value)
1, 0	ITnDTM	These bits mask the INTnDT interrupt. 1: Mask 0: Do not mask (default value)

Remark n = 1, 2

(19) UF0 INT mask 3 register (UF0IM3)

This register controls masking of the interrupt sources indicated by the UF0IS3 register.

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

FW can mask occurrence of an interrupt request from USBF (INTUSB1B) by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM3	BKO2	BKO2	BKO2	BKO2	BKO1	BKO1	BKO1	BKO1	FFFFFFE1AH	00H
	FLM	NLM	NAKM	DTM	FLM	NLM	NAKM	DTM		

Bit position	Bit name	Function
7, 3	BKOnFLM	These bits mask the BLKOnFL interrupt. 1: Mask 0: Do not mask (default value)
6, 2	BKOnNLM	These bits mask the BLKOnNL interrupt. 1: Mask 0: Do not mask (default value)
5, 1	BKOnNAKM	These bits mask the BLKOnNK interrupt. 1: Mask 0: Do not mask (default value)
4, 0	BKOnDTM	These bits mask the BLKOnDT interrupt. 1: Mask 0: Do not mask (default value)

Remark n = 1, 2

(20) UF0 INT mask 4 register (UF0IM4)

This register controls masking of the interrupt sources indicated by the UF0IS4 register.

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

FW can mask occurrence of an interrupt request from USBF (INTUSB2B) by writing 1 to the corresponding bit of this register.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IM4	0	0	SETINTM	0	0	0	0	0	FFFFFE1BH	00H

Bit position	Bit name	Function
5	SETINTM	This bit masks the SET_INT interrupt. 1: Mask 0: Do not mask (default value)

(21) UF0 INT clear 0 register (UF0IC0)

This register controls clearing the interrupt sources indicated by the UF0IS0 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC0	BUS RSTC	RSU SPDC	1	SHORTC	DMA EDC	SET RQC	CLR RQC	EP HALTC	FFFFFFE1EH	FFH

Bit position	Bit name	Function
7	BUSRSTC	This bit clears the Bus Reset interrupt. 0: Clear
6	RSUSPDC	This bit clears the Resume/Suspend interrupt. 0: Clear
4	SHORTC	This bit clears the Short interrupt. 0: Clear
3	DMAEDC	This bit clears the DMA_END interrupt. 0: Clear
2	SETRQC	This bit clears the SET_RQ interrupt. 0: Clear
1	CLRRQC	This bit clears the CLR_RQ interrupt. 0: Clear
0	EPHALTC	This bit clears the EP_Halt interrupt. 0: Clear

(22) UF0 INT clear 1 register (UF0IC1)

This register controls clearing the interrupt sources indicated by the UF0IS1 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC1	1	E0INC	E0 INDTC	E0ODTC	SUCESC	STGC	PROTC	CPU DECC	FFFFFE1FH	FFH

Bit position	Bit name	Function
6	E0INC	This bit clears the EP0IN interrupt. 0: Clear
5	E0INDTC	This bit clears the EP0INDT interrupt. 0: Clear
4	E0ODTC	This bit clears the EP0OUTDT interrupt. 0: Clear
3	SUCESC	This bit clears the Success interrupt. 0: Clear
2	STGC	This bit clears the Stg interrupt. 0: Clear
1	PROTC	This bit clears the Protect interrupt. 0: Clear
0	CPUDECC	This bit clears the CPUDEC interrupt. 0: Clear

(23) UF0 INT clear 2 register (UF0IC2)

This register controls clearing the interrupt sources indicated by the UF0IS2 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC2	BKI2INC	BKI2 DTC	BKI1INC	BKI1 DTC	1	1	IT2DTC	IT1DTC	FFFFFFE20H	FFH

Bit position	Bit name	Function
7, 5	BKInINC	These bits clear the BLKInIN interrupt. 0: Clear
6, 4	BKInDTC	These bits clear the BLKInDT interrupt. 0: Clear
1, 0	ITnDTC	These bits clear the INTnDT interrupt. 0: Clear

Remark n = 1, 2

(24) UF0 INT clear 3 register (UF0IC3)

This register controls clearing the interrupt sources indicated by the UF0IS3 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC3	BKO2	BKO2	BKO2	BKO2	BKO1	BKO1	BKO1	BKO1	FFFFFFE21H	FFH
	FLC	NLC	NAKC	DTC	FLC	NLC	NAKC	DTC		

Bit position	Bit name	Function
7, 3	BKOnFLC	These bits clear the BLKOnFL interrupt. 0: Clear
6, 2	BKOnNLC	These bits clear the BLKOnNL interrupt. 0: Clear
5, 1	BKOnNAKC	These bits clear the BLKOnNK interrupt. 0: Clear
4, 0	BKOnDTC	These bits clear the BLKOnDT interrupt. 0: Clear

Remark n = 1, 2

(25) UF0 INT clear 4 register (UF0IC4)

This register controls clearing the interrupt sources indicated by the UF0IS4 register.

This register is write-only, in 8-bit units. If this register is read, the value FFH is read.

FW can clear an interrupt source by writing 0 to the corresponding bit of this register. Even a bit that is automatically cleared to 0 by hardware can be cleared by FW before it is cleared by hardware. Writing 0 to a bit of this register automatically sets the bit to 1. Writing 1 is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IC4	1	1	SETINTC	1	1	1	1	1	FFFFFFE22H	FFH

Bit position	Bit name	Function
5	SETINTC	This bit clears the SET_INT interrupt. 0: Clear

(26) UF0 INT & DMARQ register (UF0IDR)

This register selects reporting via an interrupt request or starting DMA.

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

If data exists in either the UF0BO1 or UF0BO1 register, or if data can be written to the UF0BI1 or UF0BI2 register, this register selects whether it is reported to the FW by an interrupt request, or whether starting DMA is requested. If starting DMA is requested, the DMA transfer mode can be selected according to the setting of bits 0 and 1.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4) and the current setting of the interface.

Be sure to clear bits 3 and 2 to 0. If they are set to 1, the operation is not guaranteed.

Caution If the target endpoint is not supported by the SET_INTERFACE request under DMA transfer, the DMA request signal becomes inactive immediately, and the corresponding bit is automatically cleared to 0 by hardware.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0IDR	DQBI2 MS	DQBI1 MS	DQBO2 MS	DQBO1 MS	0	0	MODE1	MODE0	FFFFFFE26H	00H

Bit position	Bit name	Function
7, 6	DQBI _n MS	These bits enable (mask) a write DMA transfer request (DMA request signal for Endpoint m) to the UF0BI _n register. When these bits are set to 1, the DMA request signal for Endpoint m becomes active while writing data can be acknowledged. If the DMA end signal for Endpoint m is input (if the DMA controller issues TC), these bits are automatically cleared to 0 by hardware. To continue DMA transfer, re-set these bits to 1 by FW. 1: Enables active DMA request signal for Endpoint m (masks BKInDT interrupt). 0: Disables active DMA request signal for Endpoint m (default value).
5, 4	DQBO _n MS	These bits enable (mask) a read DMA transfer request (DMA request signal for Endpoint x) to the UF0BO _n register. When these bits are set to 1, the DMA request signal for Endpoint x becomes active if the data to be read is prepared in the UF0BO _n register. If the DMA end signal for Endpoint x is input (if the DMA controller issues TC), these bits are automatically cleared to 0 by hardware. They are also cleared to 0 when the USBSPxB signal is active. To continue DMA transfer, re-set these bits to 1 by FW. 1: Enables active DMA request signal for Endpoint x (masks BKOnDT interrupt). 0: Disables active DMA request signal for Endpoint x (default value).

Remark n = 1, 2
 m = 1 and x = 2 where n = 1
 m = 3 and x = 4 where n = 2

Bit position	Bit name	Function			
1, 0	MODE1, MODE0	These bits select the DMA transfer mode.			
		MODE1	MODE2	Mode	Remark
		1	1	Setting prohibited	Operation cannot be guaranteed.
		1	0	Demand mode	DMA request signal becomes active as long as there is data. It becomes inactive if there is no more data.
		0	X	Single mode	DMA request signal becomes inactive each time DMA transfer has been executed.
Remark X: Don't care					

(27) UF0 DMA status 0 register (UF0DMS0)

This register indicates the DMA status of Endpoint1 to Endpoint4.

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

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DMS0	0	0	DQE4	DQE3	DQE2	DQE1	0	0	FFFFFFE27H	00H

Bit position	Bit name	Function
5	DQE4	This bit indicates that a DMA read request is being issued from Endpoint4 to memory. 1: DMA read request from Endpoint4 is being issued. 0: DMA read request from Endpoint4 is not being issued (default value).
4	DQE3	This bit indicates that a DMA write request is being issued from memory to Endpoint3. Note that, even if data is in Endpoint3 (when the FIFO is not full and after the BK12DED bit has been set to 1), the DMA request signal becomes active immediately and DMA transfer is started when the DQB12MS bit of the UF0IDR register is set to 1. 1: DMA write request for Endpoint3 is being issued. 0: DMA write request for Endpoint3 is not being issued (default value).
3	DQE2	This bit indicates that a DMA read request is being issued from Endpoint2 to memory. 1: DMA read request from Endpoint2 is being issued. 0: DMA read request from Endpoint2 is not being issued (default value).
2	DQE1	This bit indicates that a DMA write request is being issued from memory to Endpoint1. Note that, even if data is in Endpoint1 (when the FIFO is not full and after the BK11DED bit has been set to 1), the DMA request signal becomes active immediately and DMA transfer is started when the DQB11MS bit of the UF0IDR register is set to 1. 1: DMA write request for Endpoint1 is being issued. 0: DMA write request for Endpoint1 is not being issued (default value).

(28) UF0 DMA status 1 register (UF0DMS1)

This register indicates the DMA status of Endpoint1 to Endpoint4.

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

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1 to 4) and the current setting of the interface.

Each bit is automatically cleared to 0 when this register is read. Even when this register is read, however, bits 4 and 3 of the UF0IS0 register are not cleared to 0. If the target endpoint is no longer supported by the SET_INTERFACE request, each bit is automatically cleared to 0 by hardware (however, the DMA_END interrupt request and Short interrupt request are not cleared).

	7	6	5	4	3	2	1	0	Address	After reset
UF0DMS1	DEDE4	DSPE4	DEDE3	DEDE2	DSPE2	DEDE1	0	0	FFFFFFE28H	00H

Bit position	Bit name	Function
7, 5, 4, 2	DEDEn	These bits indicate that the DMA end (TC) signal for Endpoint n becomes active and DMA is stopped while a DMA read request is being issued from Endpoint n to memory. 1: DMA end signal for Endpoint n is active. 0: DMA end signal for Endpoint n is inactive (default value).
6, 3	DSPEm	These bits indicate that, although a DMA read request was being issued from Endpoint m to memory, DMA has been stopped because the received data is a short packet and there is no more data to be transferred. 1: USBSPmB signal is active. 0: USBSPmB signal is inactive (default value).

Remark n = 1 to 4
m = 2, 4

(29) UF0 FIFO clear 0 register (UF0FIC0)

This register clears each FIFO.

This register is write-only, in 8-bit units. If this register is read, 00H is read.

FW can clear the target FIFO by writing 1 to the corresponding bit of this register. The bit to which 1 has been written is automatically cleared to 0. Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7, 8) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0FIC0	BK12SC	BK12CC	BK11SC	BK11CC	ITR2C	ITR1C	EP0WC	EP0RC	FFFFFFE30H	00H

Bit position	Bit name	Function
7, 5	BKInSC	These bits clear only the FIFO on the SIE side of the UF0BIn register (reset the counter). 1: Clear Writing these bits is invalid while an IN token for Endpoint m is being processed with the BKInNK bit set to 1. The BKInNK bit is automatically cleared to 0 by clearing the FIFO. Make sure that the FIFO on the CPU side is empty when these bits are used.
6, 4	BKInCC	These bits clear only the FIFO on the CPU side of the UF0BIn register (reset the counter). 1: Clear
3, 2	ITRnC	These bits clear the UF0INTn register (reset the counter). 1: Clear Writing these bits is invalid while an IN token for Endpoint x is being processed with the ITnNK bit set to 1. The ITnNK bit is automatically cleared to 0 by clearing the FIFO.
1	EP0WC	This bit clears the UF0E0W register (resets the counter). 1: Clear Writing this bit is invalid while an IN token for Endpoint0 is being processed with the EP0NKW bit set to 1. The EP0NKW bit is automatically cleared to 0 by clearing the FIFO.
0	EP0RC	This bit clears the UF0E0R register (resets the counter). 1: Clear When the EP0NKR bit is set to 1 (except when it has been set by FW), the EP0NKR bit is automatically cleared to 0 by clearing the FIFO.

Remark n = 1, 2
m = 1 and x = 7 where n = 1
m = 3 and x = 8 where n = 2

(30) UF0 FIFO clear 1 register (UF0FIC1)

This register clears each FIFO.

This register is write-only, in 8-bit units. If this register is read, 00H is read.

FW can clear the target FIFO by writing 1 to the corresponding bit of this register. The bit to which 1 has been written is automatically cleared to 0. Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 2, 4) and the current setting of the interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0FIC1	0	0	0	0	BKO2C	BKO2CC	BKO1C	BKO1CC	FFFFE31H	00H

Bit position	Bit name	Function
3, 1	BKOnC	These bits clear the FIFOs on both the SIE and CPU sides of the UF0BOn register (reset the counter). 1: Clear When the BKOnNK bit is set to 1 (except when it has been set by FW), the BKOnNK bit is automatically cleared to 0 by clearing the FIFO.
2, 0	BKOnCC	These bits clear only the FIFO on the CPU side of the UF0BOn register (reset the counter). 1: Clear When the BKOnNK bit is set to 1 (except when it has been set by FW), the BKOnNK bit is automatically cleared to 0 by clearing the FIFO.

Remark n = 1, 2

(31) UF0 data end register (UF0DEND)

This register reports the end of writing to the transmission system.

This register is write-only, in 8-bit units (however, bits 7 and 6 can be read and written). If this register is read, 00H is read.

FW can start data transfer of the target endpoint by writing 1 to the corresponding bit of this register. The bit to which 1 has been written is automatically cleared to 0. Writing 0 to the bit is invalid.

The related bits are invalid if each endpoint is not supported by the setting of the UF0EnIM register (n = 1, 3, 7, 8) and the current setting of the interface.

(1/2)

	7	6	5	4	3	2	1	0	Address	After reset
UF0DEND	BKI2T	BKI1T	0	IT2DEND	IT1DEND	BKI2DED	BKI1DED	E0DED	FFFFFFE35H	00H

Bit position	Bit name	Function
7, 6	BKInT	These bits specify whether toggling the FIFO is automatically executed if the FIFO on the CPU side of the UF0BIn register becomes full as a result of DMA. 1: Automatically execute a toggle operation of the FIFO as soon as the FIFO has become full. 0: Do not automatically execute a toggle operation of the FIFO even if the FIFO becomes full (default value).
4, 3	ITnDEND	Set these bits to 1 to transmit the data of the UF0INTn register. When these bits are set to 1, the ITnNK bit is set to 1 and data transfer is executed. 1: Transmit a short packet. 0: Do not transmit a short packet (default value). If the ITRnC bit of the UF0FIC0 register is set to 1 and then these bits are set to 1 (counter of UF0INTn register = 0 and the corresponding bit of the UF0EPS0 register = 1), a Null packet (with a data length of 0) is transmitted. If data exists in the UF0INTn register and if these bits are set to 1 (counter of UF0INTn register ≠ 0 and the corresponding bit of the UF0EPS0 register = 1), a short packet is transmitted. These bits are automatically controlled by hardware when the FIFO is full.

Remark n = 1, 2

Bit position	Bit name	Function
2, 1	BKInDED	<p>Set these bits to 1 when writing transmit data to the UF0BIn register has been completed. When these bits are set to 1, the FIFO is toggled as soon as possible, the BKInNK bit is set to 1, and data is transferred.</p> <p>1: Transmit a short packet. 0: Do not transmit a short packet (default value).</p> <p>These bits control the FIFO on the CPU side.</p> <p>If the BKInCC bit of the UF0FIC0 register is set to 1 and then these bits are set to 1 (counter of UF0BIn register = 0), a Null packet (with a data length of 0) is transmitted.</p> <p>If data exists in the UF0BIn register and if these bits are set to 1 (counter of UF0BIn register ≠ 0), and if the FIFO is not full, a short packet is transmitted.</p> <p>If the FIFO on the CPU side of the UF0BIn register becomes full as a result of DMA, with the PIO or BKInT bit set to 1, the hardware starts data transmission even if these bits are not set to 1.</p> <p>If the FIFO on the CPU side of the UF0BIn register becomes full as a result of DMA, with the BKInT bit cleared to 0, be sure to set these bits to 1 (see 11.4.1 (3) UF0 EPNACK register (UF0EN)).</p>
0	E0DED	<p>Set this bit to 1 to transmit data of the UF0E0W register. When this bit is set to 1, the EPONKW bit is set to 1 and data is transferred.</p> <p>1: Transmit a short packet. 0: Do not transmit a short packet (default value).</p> <p>If the EP0WC bit of the UF0FIC0 register is set to 1 and if this bit is set to 1 (counter of UF0E0W register = 0 and bit 1 of UF0EPS0 register = 1), a Null packet (with a data length of 0) is transmitted.</p> <p>If data exists in the UF0E0W register and if this bit is set to 1 (counter of UF0E0W register ≠ 0 and bit 1 of the UF0EPS0 register = 1), and if the FIFO is not full, a short packet is transmitted.</p>

Remark n = 1, 2

(32) UF0 GPR register (UF0GPR)

This register controls USBF and the USB interface.

This register can be read or written in 8-bit units (however, bit 0 can only be written). If this register is read, 00H is read.

FW can reset the USBF by writing 1 to bit 0 of this register. This bit is automatically cleared to 0 after 1 has been written to it. Writing 0 to this bit is invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0GPR	0	0	0	0	0	0	0	MRST	FFFFFFE37H	00H

Bit position	Bit name	Function
0	MRST	Set this bit to 1 to reset USBF. 1: Reset Actually, USBF is reset two USB clocks after this bit has been set to 1 by FW and the write signal has become inactive. While the system clock is operating, USBF is reset by this bit in the same manner as by the $\overline{\text{RESET}}$ pin (hardware reset).

(33) UF0 mode control register (UF0MODC)

This register controls CPUDEC processing.

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

By setting each bit of this register, the setting of the UF0MODS register can be changed. The bit of this register is automatically cleared to 0 only at hardware reset and when the MRST bit of the UF0GRP register has been set to 1.

Even if the bit of this register has automatically been set to 1 by hardware, the setting by FW takes precedence.

Be sure to clear bits 7 and 5 to 2 to 0. If they are set to 1, the operation is not guaranteed.

Caution This register is provided for debugging purposes. Usually, do not set this register except for verifying the operation or when a special mode is used.

	7	6	5	4	3	2	1	0	Address	After reset
UF0MODC	0	CDC GDST	0	0	0	0	0	0	FFFFFE3AH	00H

Bit position	Bit name	Function
6	CDCGDST	Set this bit to 1 to switch the GET_DESCRIPTOR Configuration request to CPUDEC processing. By setting this bit to 1, the CDCGD bit of the UF0MODS register can be forcibly set to 1. <ul style="list-style-type: none"> 1: Forcibly change the GET_DESCRIPTOR Configuration request to CPUDEC processing (sets the CDCGD bit of the UF0MODS register to 1). 0: Automatically process the GET_DESCRIPTOR Configuration request (default value).

(34) UF0 mode status register (UF0MODS)

This register indicates the configuration status.

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

★

	7	6	5	4	3	2	1	0	Address	After reset
UF0MODS	0	CDCGD	0	MPACK	DFLT	CONF	0	0	FFFFFE3CH	00H

Bit position	Bit name	Function
6	CDCGD	This bit specifies whether CPUDEC processing is performed for the GET_DESCRIPTOR Configuration request. 1: Forcibly change the GET_DESCRIPTOR Configuration request to CPUDEC processing. 0: Automatically process the GET_DESCRIPTOR Configuration request (default value).
4	MPACK	This bit indicates the transmit packet size of Endpoint0. 1: Transmit a packet of other than 8 bytes. 0: Transmit a packet of 8 bytes (default value). This bit is automatically set to 1 by hardware after the GET_DESCRIPTOR Device request has been processed (on normal completion of the status stage). It is not cleared to 0 until the USBF has been reset (it is not cleared to 0 by Bus Reset). If this bit is not set to 1, the hardware transfers only the automatically-executed request in 8-byte units. Therefore, even if data of more than 8 bytes is sent by the OUT token to be processed by FW before completion of the GET_DESCRIPTOR Device request, the data is correctly received. This bit is ignored if the size of Endpoint0 is 8 bytes.
3	DFLT	This bit indicates the default status (DFLT bit = 1). 1: Enables response. 0: Disables response (always no response) (default value). This bit is automatically set to 1 by Bus Reset. The transaction for all the endpoints is not responded to until this bit is set to 1.
2	CONF	This bit indicates whether the SET_CONFIGURATION request has been completed. 1: SET_CONFIGURATION request has been completed. 0: SET_CONFIGURATION request has not been completed (default value). This bit is set to 1 when Configuration value = 1 is received by the SET_CONFIGURATION request. Unless this bit is set to 1, access to an endpoint other than Endpoint0 is ignored. This bit is cleared to 0 when Configuration value = 0 is received by the SET_CONFIGURATION request. It is also cleared to 0 when Bus Reset is detected.

(35) UF0 active interface number register (UF0AIFN)

This register sets the valid Interface number that correctly responds to the GET/SET_INTERFACE request. Because Interface 0 is always valid, Interfaces 1 to 4 can be selected.

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

	7	6	5	4	3	2	1	0	Address	After reset
UF0AIFN	ADDIF	0	0	0	0	0	IFNO1	IFNO0	FFFFFFE40H	00H

Bit position	Bit name	Function															
7	ADDIF	This bit allows use of Interfaces numbered other than 0. 1: Support up to the Interface number specified by the IFNO1 and IFNO0 bits. 0: Support only Interface 0 (default value). Setting bits 1 and 0 of this register is invalid when this bit is not set to 1.															
1, 0	IFNO1, IFNO0	These bits specify the range of Interface numbers to be supported. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">IFNO1</th> <th style="width: 10%;">IFNO0</th> <th style="width: 80%;">Valid Interface No.</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0, 1, 2, 3, 4</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0, 1, 2, 3</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0, 1, 2</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0, 1</td> </tr> </tbody> </table>	IFNO1	IFNO0	Valid Interface No.	1	1	0, 1, 2, 3, 4	1	0	0, 1, 2, 3	0	1	0, 1, 2	0	0	0, 1
IFNO1	IFNO0	Valid Interface No.															
1	1	0, 1, 2, 3, 4															
1	0	0, 1, 2, 3															
0	1	0, 1, 2															
0	0	0, 1															

(36) UF0 active alternative setting register (UF0AAS)

This register specifies a link between the Interface number and Alternative Setting.

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

USBF of the V850E/ME2 can set a five-series Alternative Setting (Alternate Setting 0, 1, 2, 3, and 4 can be defined) and a two-series Alternative Setting (Alternative Setting 0 and 1 can be defined) for one Interface.

	7	6	5	4	3	2	1	0	Address	After reset
UF0AAS	ALT2	IFAL21	IFAL20	ALT2EN	ALT5	IFAL51	IFAL50	ALT5EN	FFFFFE41H	00H

Bit position	Bit name	Function															
7, 3	ALTn	These bits specify whether an n-series Alternative Setting is linked with Interface 0. When these bits are set to 1, the setting of the IFALn1 and IFALn0 bits is invalid. <ul style="list-style-type: none"> 1: Link n-series Alternative Setting with Interface 0. 0: Do not link n-series Alternative Setting with Interface 0 (default value). 															
6, 5, 2, 1	IFALn1, IFALn0	These bits specify the Interface number to be linked with the n-series Alternative Setting. If the linked Interface number is outside the range specified by the UF0AIFN register, the n-series Alternative Setting is invalid (ALTnEN bit = 0). <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th style="width: 10%;">IFALn1</th> <th style="width: 10%;">IFALn0</th> <th style="width: 80%;">Interface number to be linked</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Links Interface 4.</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Links Interface 3.</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Links Interface 2.</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Links Interface 1.</td> </tr> </tbody> </table> Do not link a five-series Alternative Setting and a two-series Alternative Setting with the same Interface number.	IFALn1	IFALn0	Interface number to be linked	1	1	Links Interface 4.	1	0	Links Interface 3.	0	1	Links Interface 2.	0	0	Links Interface 1.
IFALn1	IFALn0	Interface number to be linked															
1	1	Links Interface 4.															
1	0	Links Interface 3.															
0	1	Links Interface 2.															
0	0	Links Interface 1.															
4, 0	ALTnEN	These bits validate the n-series Alternative Setting. Unless these bits are set to 1, the setting of the ALTn, IFALn1, and IFALn0 bits is invalid. <ul style="list-style-type: none"> 1: Validate the n-series Alternative Setting. 0: Do not validate the n-series Alternative Setting (default value). 															

Remark n = 2, 5

For example, when the UF0AIFN register is set to 82H and the UF0AAS register is set to 15H, Interfaces 0, 1, 2, and 3 are valid. Interfaces 0 and 2 support only Alternative Setting 0. Interface 1 supports Alternative Setting 0 and 1, and Interface 3 supports Alternative Setting 0, 1, 2, 3, and 4. With this setting, requests GET_INTERFACE wIndex = 0/1/2/3, SET_INTERFACE wValue = 0 & wIndex = 0/2, SET_INTERFACE wValue = 0/1 & wIndex = 1, and SET_INTERFACE wValue = 0/1/2/3/4 & wIndex = 3 are automatically responded to, and a STALL response is made to the other GET/SET_INTERFACE requests.

(37) UF0 alternative setting status register (UF0ASS)

This register indicates the current status of the Alternative Setting.

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

Check this register when the SET_INT interrupt request has been issued. The value received by the SET_INTERFACE request is reflected on the UF0IFn register (n = 0 to 4) as well as on this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0ASS	0	0	0	0	AL5ST3	AL5ST2	AL5ST1	AL2ST	FFFFFFE42H	00H

Bit position	Bit name	Function																								
3 to 1	AL5ST3 to AL5ST1	These bits indicate the current status of the five-series Alternative Setting. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">AL5ST3</th> <th style="width: 10%;">AL5ST2</th> <th style="width: 10%;">AL5ST1</th> <th style="width: 70%;">Selected Alternative Setting number</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Alternative Setting 4</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>Alternative Setting 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>Alternative Setting 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>Alternative Setting 1</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>Alternative Setting 0</td> </tr> </tbody> </table>	AL5ST3	AL5ST2	AL5ST1	Selected Alternative Setting number	1	0	0	Alternative Setting 4	0	1	1	Alternative Setting 3	0	1	0	Alternative Setting 2	0	0	1	Alternative Setting 1	0	0	0	Alternative Setting 0
AL5ST3	AL5ST2	AL5ST1	Selected Alternative Setting number																							
1	0	0	Alternative Setting 4																							
0	1	1	Alternative Setting 3																							
0	1	0	Alternative Setting 2																							
0	0	1	Alternative Setting 1																							
0	0	0	Alternative Setting 0																							
0	AL2ST	This bit indicates the current status of the two-series Alternative Setting (selected Alternative Setting number). 1: Alternative Setting 1 0: Alternative Setting 0																								

(38) UF0 endpoint 1 interface mapping register (UF0E1IM)

This register specifies for which Interface and Alternative Setting Endpoint1 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint1 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint1 request and the IN transaction to Endpoint1 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E1IM	E1EN2	E1EN1	E1EN0	E12AL1	E15AL4	E15AL3	E15AL2	E15AL1	FFFFFFE43H	00H

Bit position	Bit name	Function																																			
7 to 5	E1EN2 to E1EN0	<p>These bits set a link between the Interface of Endpoint1 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin: 10px 0;"> <thead> <tr> <th style="width: 10%;">E1EN2</th> <th style="width: 10%;">E1EN1</th> <th style="width: 10%;">E1EN0</th> <th style="width: 70%;">Link status</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td rowspan="2" style="text-align: center;">Not linked with Interface</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</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;">Linked with Interface 4 and Alternative Setting 0</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;">Linked with Interface 3 and Alternative Setting 0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Linked with Interface 2 and Alternative Setting 0</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;">Linked with Interface 1 and Alternative Setting 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;">Linked with Interface 0 and Alternative Setting 0</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;">Not linked with Interface (default value)</td> </tr> </tbody> </table> <p>When these bits are set to 110 or 111, they are invalid even if the E12AL1 bit is cleared to 0. If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint1 is valid.</p>	E1EN2	E1EN1	E1EN0	Link status	1	1	1	Not linked with Interface	1	1	0	1	0	1	Linked with Interface 4 and Alternative Setting 0	1	0	0	Linked with Interface 3 and Alternative Setting 0	0	1	1	Linked with Interface 2 and Alternative Setting 0	0	1	0	Linked with Interface 1 and Alternative Setting 0	0	0	1	Linked with Interface 0 and Alternative Setting 0	0	0	0	Not linked with Interface (default value)
E1EN2	E1EN1	E1EN0	Link status																																		
1	1	1	Not linked with Interface																																		
1	1	0																																			
1	0	1	Linked with Interface 4 and Alternative Setting 0																																		
1	0	0	Linked with Interface 3 and Alternative Setting 0																																		
0	1	1	Linked with Interface 2 and Alternative Setting 0																																		
0	1	0	Linked with Interface 1 and Alternative Setting 0																																		
0	0	1	Linked with Interface 0 and Alternative Setting 0																																		
0	0	0	Not linked with Interface (default value)																																		
4	E12AL1	<p>This bit validates Endpoint1 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.</p> <p>1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).</p> <p>This bit is valid when the E15AL4 to E15AL1 bits are 0000.</p>																																			
3 to 0	E15ALn	<p>These bits validate Endpoint1 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.</p> <p>1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1. 0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).</p>																																			

Remark n = 1 to 4

(39) UF0 endpoint 2 interface mapping register (UF0E2IM)

This register specifies for which Interface and Alternative Setting Endpoint2 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint2 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint2 request and the OUT transaction to Endpoint2 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E2IM	E2EN2	E2EN1	E2EN0	E22AL1	E25AL4	E25AL3	E25AL2	E25AL1	FFFFFE44H	00H

Bit position	Bit name	Function																																			
7 to 5	E2EN2 to E2EN0	<p>These bits set a link between the Interface of Endpoint2 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.</p> <table border="1"> <thead> <tr> <th>E2EN2</th> <th>E2EN1</th> <th>E2EN0</th> <th>Link status</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>1</td> <td rowspan="2">Not linked with Interface</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>Linked with Interface 4 and Alternative Setting 0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>Linked with Interface 3 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Linked with Interface 2 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Linked with Interface 1 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Linked with Interface 0 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Not linked with Interface (default value)</td> </tr> </tbody> </table> <p>When these bits are set to 110 or 111, they are invalid even if the E22AL1 bit is cleared to 0.</p> <p>If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint2 is valid.</p>	E2EN2	E2EN1	E2EN0	Link status	1	1	1	Not linked with Interface	1	1	0	1	0	1	Linked with Interface 4 and Alternative Setting 0	1	0	0	Linked with Interface 3 and Alternative Setting 0	0	1	1	Linked with Interface 2 and Alternative Setting 0	0	1	0	Linked with Interface 1 and Alternative Setting 0	0	0	1	Linked with Interface 0 and Alternative Setting 0	0	0	0	Not linked with Interface (default value)
E2EN2	E2EN1	E2EN0	Link status																																		
1	1	1	Not linked with Interface																																		
1	1	0																																			
1	0	1	Linked with Interface 4 and Alternative Setting 0																																		
1	0	0	Linked with Interface 3 and Alternative Setting 0																																		
0	1	1	Linked with Interface 2 and Alternative Setting 0																																		
0	1	0	Linked with Interface 1 and Alternative Setting 0																																		
0	0	1	Linked with Interface 0 and Alternative Setting 0																																		
0	0	0	Not linked with Interface (default value)																																		
4	E22AL1	<p>This bit validates Endpoint2 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.</p> <p>1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).</p> <p>This bit is valid when the E25AL4 to E25AL1 bits are 0000.</p>																																			
3 to 0	E25ALn	<p>These bits validate Endpoint2 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.</p> <p>1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).</p>																																			

Remark n = 1 to 4

(40) UF0 endpoint 3 interface mapping register (UF0E3IM)

This register specifies for which Interface and Alternative Setting Endpoint3 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint3 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint3 request and the IN transaction to Endpoint3 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E3IM	E3EN2	E3EN1	E3EN0	E32AL1	E35AL4	E35AL3	E35AL2	E35AL1	FFFFFFE45H	00H

Bit position	Bit name	Function																																			
7 to 5	E3EN2 to E3EN0	<p>These bits set a link between the Interface of Endpoint3 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin: 10px 0;"> <thead> <tr> <th style="width: 10%;">E3EN2</th> <th style="width: 10%;">E3EN1</th> <th style="width: 10%;">E3EN0</th> <th style="width: 70%;">Link status</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td rowspan="2" style="text-align: center;">Not linked with Interface</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</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;">Linked with Interface 4 and Alternative Setting 0</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;">Linked with Interface 3 and Alternative Setting 0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Linked with Interface 2 and Alternative Setting 0</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;">Linked with Interface 1 and Alternative Setting 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;">Linked with Interface 0 and Alternative Setting 0</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;">Not linked with Interface (default value)</td> </tr> </tbody> </table> <p>When these bits are set to 110 or 111, they are invalid even if the E32AL1 bit is cleared to 0.</p> <p>If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint3 is valid.</p>	E3EN2	E3EN1	E3EN0	Link status	1	1	1	Not linked with Interface	1	1	0	1	0	1	Linked with Interface 4 and Alternative Setting 0	1	0	0	Linked with Interface 3 and Alternative Setting 0	0	1	1	Linked with Interface 2 and Alternative Setting 0	0	1	0	Linked with Interface 1 and Alternative Setting 0	0	0	1	Linked with Interface 0 and Alternative Setting 0	0	0	0	Not linked with Interface (default value)
E3EN2	E3EN1	E3EN0	Link status																																		
1	1	1	Not linked with Interface																																		
1	1	0																																			
1	0	1	Linked with Interface 4 and Alternative Setting 0																																		
1	0	0	Linked with Interface 3 and Alternative Setting 0																																		
0	1	1	Linked with Interface 2 and Alternative Setting 0																																		
0	1	0	Linked with Interface 1 and Alternative Setting 0																																		
0	0	1	Linked with Interface 0 and Alternative Setting 0																																		
0	0	0	Not linked with Interface (default value)																																		
4	E32AL1	<p>This bit validates Endpoint3 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.</p> <p>1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).</p> <p>This bit is valid when the E35AL4 to E35AL1 bits are 0000.</p>																																			
3 to 0	E35ALn	<p>These bits validate Endpoint3 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.</p> <p>1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).</p>																																			

Remark n = 1 to 4

(41) UF0 endpoint 4 interface mapping register (UF0E4IM)

This register specifies for which Interface and Alternative Setting Endpoint4 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint4 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint4 request and the OUT transaction to Endpoint4 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E4IM	E4EN2	E4EN1	E4EN0	E42AL1	E45AL4	E45AL3	E45AL2	E45AL1	FFFFFE46H	00H

Bit position	Bit name	Function																																			
7 to 5	E4EN2 to E4EN0	<p>These bits set a link between the Interface of Endpoint4 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">E4EN2</th> <th style="width: 10%;">E4EN1</th> <th style="width: 10%;">E4EN0</th> <th style="width: 70%;">Link status</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td rowspan="2">Not linked with Interface</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</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>Linked with Interface 4 and Alternative Setting 0</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>Linked with Interface 3 and Alternative Setting 0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Linked with Interface 2 and Alternative Setting 0</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>Linked with Interface 1 and Alternative Setting 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>Linked with Interface 0 and Alternative Setting 0</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>Not linked with Interface (default value)</td> </tr> </tbody> </table> <p>When these bits are set to 110 or 111, they are invalid even if the E42AL1 bit is cleared to 0.</p> <p>If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint4 is valid.</p>	E4EN2	E4EN1	E4EN0	Link status	1	1	1	Not linked with Interface	1	1	0	1	0	1	Linked with Interface 4 and Alternative Setting 0	1	0	0	Linked with Interface 3 and Alternative Setting 0	0	1	1	Linked with Interface 2 and Alternative Setting 0	0	1	0	Linked with Interface 1 and Alternative Setting 0	0	0	1	Linked with Interface 0 and Alternative Setting 0	0	0	0	Not linked with Interface (default value)
E4EN2	E4EN1	E4EN0	Link status																																		
1	1	1	Not linked with Interface																																		
1	1	0																																			
1	0	1	Linked with Interface 4 and Alternative Setting 0																																		
1	0	0	Linked with Interface 3 and Alternative Setting 0																																		
0	1	1	Linked with Interface 2 and Alternative Setting 0																																		
0	1	0	Linked with Interface 1 and Alternative Setting 0																																		
0	0	1	Linked with Interface 0 and Alternative Setting 0																																		
0	0	0	Not linked with Interface (default value)																																		
4	E42AL1	<p>This bit validates Endpoint4 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.</p> <p>1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).</p> <p>This bit is valid when the E45AL4 to E45AL1 bits are 0000.</p>																																			
3 to 0	E45ALn	<p>These bits validate Endpoint4 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.</p> <p>1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).</p>																																			

Remark n = 1 to 4

(42) UF0 endpoint 7 interface mapping register (UF0E7IM)

This register specifies for which Interface and Alternative Setting Endpoint7 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint7 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint7 request and the IN transaction to Endpoint7 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E7IM	E7EN2	E7EN1	E7EN0	E72AL1	E75AL4	E75AL3	E75AL2	E75AL1	FFFFFFE49H	00H

Bit position	Bit name	Function																																			
7 to 5	E7EN2 to E7EN0	<p>These bits set a link between the Interface of Endpoint7 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin: 10px 0;"> <thead> <tr> <th style="width: 10%;">E7EN2</th> <th style="width: 10%;">E7EN1</th> <th style="width: 10%;">E7EN0</th> <th style="width: 70%;">Link status</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td rowspan="2" style="text-align: center;">Not linked with Interface</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</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;">Linked with Interface 4 and Alternative Setting 0</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;">Linked with Interface 3 and Alternative Setting 0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">Linked with Interface 2 and Alternative Setting 0</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;">Linked with Interface 1 and Alternative Setting 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;">Linked with Interface 0 and Alternative Setting 0</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;">Not linked with Interface (default value)</td> </tr> </tbody> </table> <p>When these bits are set to 110 or 111, they are invalid even if the E72AL1 bit is cleared to 0.</p> <p>If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint7 is valid.</p>	E7EN2	E7EN1	E7EN0	Link status	1	1	1	Not linked with Interface	1	1	0	1	0	1	Linked with Interface 4 and Alternative Setting 0	1	0	0	Linked with Interface 3 and Alternative Setting 0	0	1	1	Linked with Interface 2 and Alternative Setting 0	0	1	0	Linked with Interface 1 and Alternative Setting 0	0	0	1	Linked with Interface 0 and Alternative Setting 0	0	0	0	Not linked with Interface (default value)
E7EN2	E7EN1	E7EN0	Link status																																		
1	1	1	Not linked with Interface																																		
1	1	0																																			
1	0	1	Linked with Interface 4 and Alternative Setting 0																																		
1	0	0	Linked with Interface 3 and Alternative Setting 0																																		
0	1	1	Linked with Interface 2 and Alternative Setting 0																																		
0	1	0	Linked with Interface 1 and Alternative Setting 0																																		
0	0	1	Linked with Interface 0 and Alternative Setting 0																																		
0	0	0	Not linked with Interface (default value)																																		
4	E72AL1	<p>This bit validates Endpoint7 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.</p> <p>1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).</p> <p>This bit is valid when the E75AL4 to E75AL1 bits are 0000.</p>																																			
3 to 0	E75ALn	<p>These bits validate Endpoint7 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.</p> <p>1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).</p>																																			

Remark n = 1 to 4

(43) UF0 endpoint 8 interface mapping register (UF0E8IM)

This register specifies for which Interface and Alternative Setting Endpoint8 is valid.

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

The setting of this register and the Alternative Setting selected by the SET_INTERFACE request indicate whether Endpoint8 is currently valid, and the hardware determines how the GET_STATUS/CLEAR_FEATURE/SET_FEATURE Endpoint8 request and the IN transaction to Endpoint8 are responded to, and whether the related bits are valid or invalid.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E8IM	E8EN2	E8EN1	E8EN0	E82AL1	E85AL4	E85AL3	E85AL2	E85AL1	FFFFFE4AH	00H

Bit position	Bit name	Function																																			
7 to 5	E8EN2 to E8EN0	<p>These bits set a link between the Interface of Endpoint8 and the two-/five-series Alternative Setting. The endpoint is linked with Alternative Setting 0. The endpoint linked with Alternative Setting 0 cannot be excluded from Alternative Setting 1 to 4.</p> <table border="1"> <thead> <tr> <th>E8EN2</th> <th>E8EN1</th> <th>E8EN0</th> <th>Link status</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>1</td> <td rowspan="2">Not linked with Interface</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>Linked with Interface 4 and Alternative Setting 0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>Linked with Interface 3 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Linked with Interface 2 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Linked with Interface 1 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Linked with Interface 0 and Alternative Setting 0</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Not linked with Interface (default value)</td> </tr> </tbody> </table> <p>When these bits are set to 110 or 111, they are invalid even if the E82AL1 bit is cleared to 0.</p> <p>If the endpoint is linked, setting of the CONF bit of the UF0MODS register to 1 indicates that Endpoint8 is valid.</p>	E8EN2	E8EN1	E8EN0	Link status	1	1	1	Not linked with Interface	1	1	0	1	0	1	Linked with Interface 4 and Alternative Setting 0	1	0	0	Linked with Interface 3 and Alternative Setting 0	0	1	1	Linked with Interface 2 and Alternative Setting 0	0	1	0	Linked with Interface 1 and Alternative Setting 0	0	0	1	Linked with Interface 0 and Alternative Setting 0	0	0	0	Not linked with Interface (default value)
E8EN2	E8EN1	E8EN0	Link status																																		
1	1	1	Not linked with Interface																																		
1	1	0																																			
1	0	1	Linked with Interface 4 and Alternative Setting 0																																		
1	0	0	Linked with Interface 3 and Alternative Setting 0																																		
0	1	1	Linked with Interface 2 and Alternative Setting 0																																		
0	1	0	Linked with Interface 1 and Alternative Setting 0																																		
0	0	1	Linked with Interface 0 and Alternative Setting 0																																		
0	0	0	Not linked with Interface (default value)																																		
4	E82AL1	<p>This bit validates Endpoint8 when the two-series Alternative Setting and the Alternative Setting of the linked Interface are set to 1.</p> <p>1: Validate the endpoint when Alternative Setting 1 is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting 1 is set with CONF bit = 1 (default value).</p> <p>This bit is valid when the E85AL4 to E85AL1 bits are 0000.</p>																																			
3 to 0	E85ALn	<p>These bits validate Endpoint8 when the five-series Alternative Setting and the Alternative Setting of the linked Interface are set to n.</p> <p>1: Validate the endpoint when Alternative Setting n is set with CONF bit = 1.</p> <p>0: Do not validate the endpoint even when Alternative Setting n is set with CONF bit = 1 (default value).</p>																																			

Remark n = 1 to 4

11.4.2 Data hold registers

(1) UF0 EP0 read register (UF0E0R)

The UF0E0R register is a 64-byte FIFO that stores the OUT data sent from the host in the data stage of control transfer to/from Endpoint0.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The hardware automatically transfers data to the UF0E0R register when it has received the data from the host. When the data has been correctly received, the E0ODT bit of the UF0IS1 register is set to 1. The UF0E0L register holds the quantity of the received data, and an interrupt request (INTUSB0B) is issued. The UF0E0L register always updates the length of the received data while it is receiving data. If the final transfer is correct reception, the interrupt request is generated. If the reception is abnormal, the UF0E0L register is cleared to 0 and the interrupt request is not generated.

The data held by the UF0E0R register must be read by FW up to the value of the amount of data read by the UF0E0L register. Check that all data has been read by using the EP0R bit of the UF0EPS0 register (EP0R = 0 when all data has been read). If the value of the UF0E0L register is 0, the EP0NKR bit of the UF0E0N register is cleared to 0, and the UF0E0R register is ready for reception. The UF0E0R register is cleared when the next SETUP token has been received.

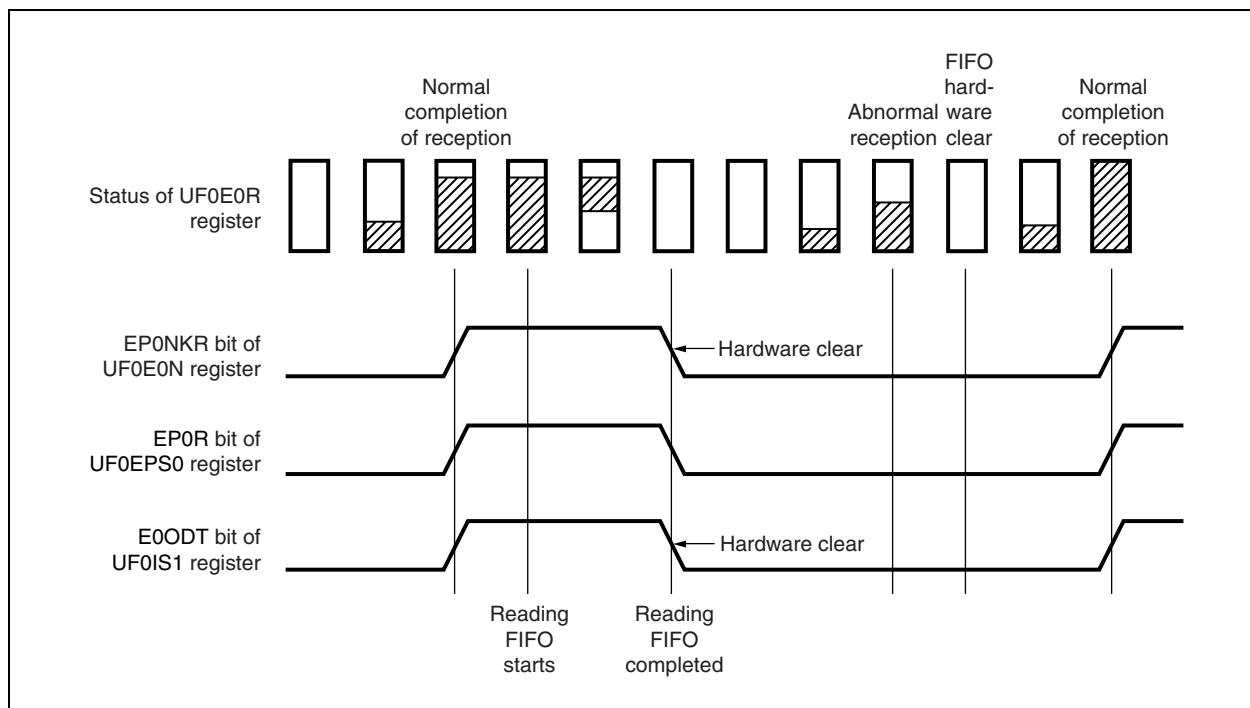
Caution Read all the data stored. To discard some data, clear the FIFO.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0R	E0R7	E0R6	E0R5	E0R4	E0R3	E0R2	E0R1	E0R0	FFFFFE80H	Undefined

Bit position	Bit name	Function
7 to 0	E0R7 to E0R0	These bits store the OUT data sent from the host in the data stage of control transfer to/from Endpoint0.

The operation of the UF0E0R register is illustrated below.

Figure 11-1. Operation of UF0E0R Register



(2) UF0 EP0 length register (UF0E0L)

The UF0E0L register stores the data length held by the UF0E0R register.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0E0L register always updates the length of the received data while it is receiving data. If the final transfer is abnormal reception, the UF0E0L register is cleared to 0 and the interrupt request is not generated.

The interrupt request is generated only when the reception is normal, and the FW can read as many data from the UF0E0R register as the value read from the UF0E0L register. The value of the UF0E0L register is decremented each time the UF0E0R register has been read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0L	E0L7	E0L6	E0L5	E0L4	E0L3	E0L2	E0L1	E0L0	FFFFFFE81H	00H
Bit position	Bit name		Function							
7 to 0	E0L7 to E0L0		These bits store the data length held by the UF0E0R register.							

(3) UF0 EP0 setup register (UF0E0ST)

The UF0E0ST register holds the SETUP data sent from the host.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0E0ST register always writes data when a SETUP transaction has been received. The hardware sets the PROT bit of the UF0IS1 register when it has correctly received the SETUP transaction. It sets the CPUDEC bit of the UF0IS1 register in the case of an FW-processed request. Then an interrupt request (INTUSB0B) is issued. In the case of an FW-processed request, be sure to read the request in 8-byte units. If it is not read in 8-byte units, the subsequent requests cannot be correctly decoded. The read counter of the UF0E0ST register is not cleared even when Bus Reset is received. Always read this counter in 8-byte units regardless of whether Bus Reset is received or not.

Because the UF0E0ST register always enables writing, the hardware overwrites data to this register even if a SETUP transaction is received while the data of the register is being read. Even if the SETUP transaction cannot be correctly received, the CPUDEC interrupt request and Protect interrupt request are not generated, but the previous data is discarded. If a SETUP token of less than 8 bytes is received, however, the received SETUP token is discarded, and the previously received SETUP data is retained. If the SETUP token is received more than once when control transfer is executed once, be sure to check the PROT bit of the UF0IS1 register under the conditions below. If PROT bit = 1, read the UF0E0ST register again because the SETUP transaction has been received more than once.

<1> If a request is decoded by FW and the UF0E0R register is read or the UF0E0W register is written

<2> When preparing for a STALL response for the request to which the decode result does not correspond

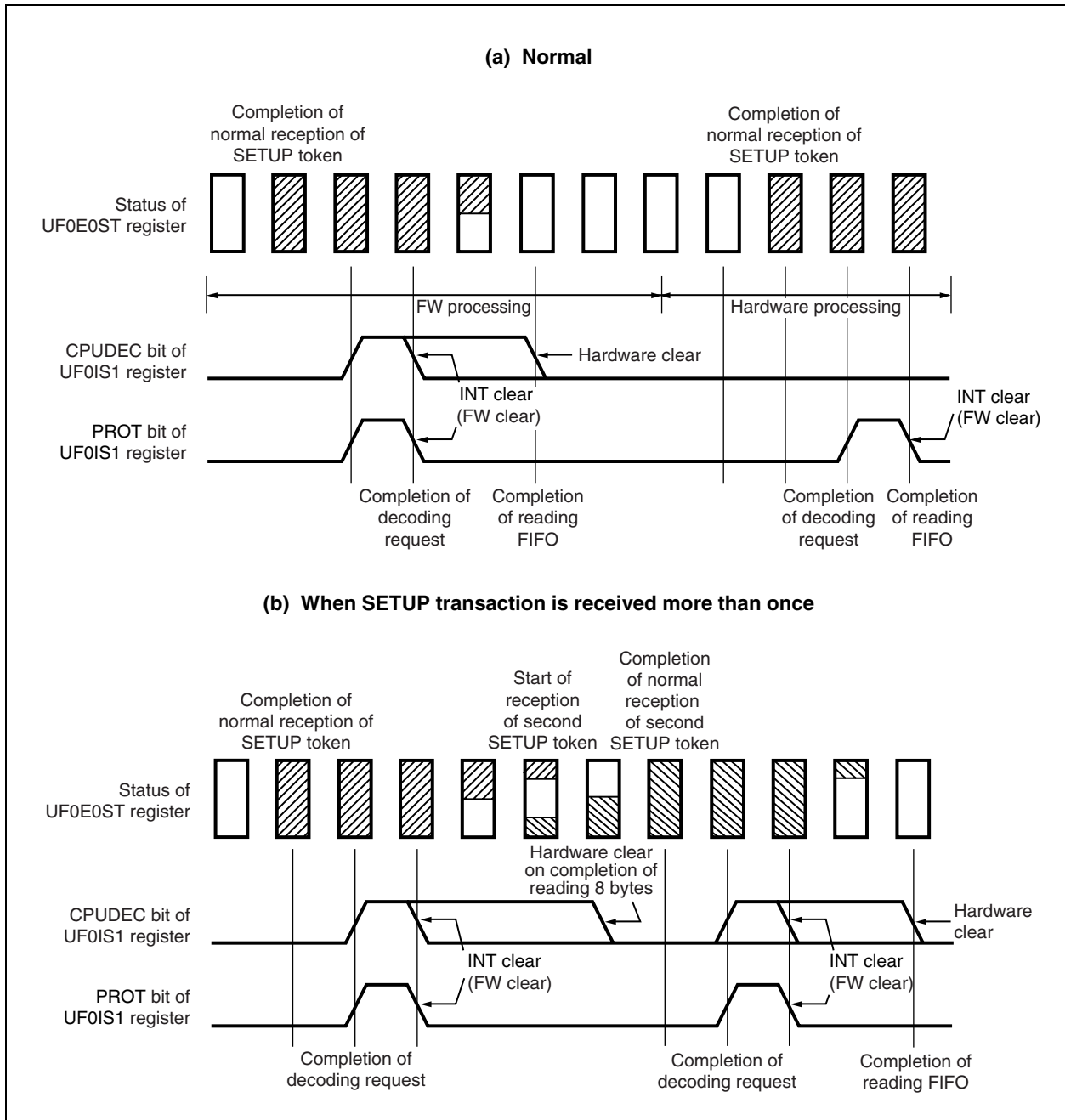
Caution Be sure to read all the stored data. The UF0E0ST register is always updated by the request in the SETUP transaction.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0ST	E0S7	E0S6	E0S5	E0S4	E0S3	E0S2	E0S1	E0S0	FFFFFFE82H	00H

Bit position	Bit name	Function
7 to 0	E0S7 to E0S0	These bits hold the SETUP data sent from the host.

The operation of the UF0E0ST register is illustrated below.

Figure 11-2. Operation of UF0E0ST Register



(4) UF0 EP0 write register (UF0E0W)

The UF0E0W register is a 64-byte FIFO that stores the IN data (passes it to SIE) sent from the host to Endpoint0 in the data stage.

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with an IN token only when the EP0NKW bit of the UF0E0N register is set to 1 (when NAK is not transmitted). When data is transmitted and when the host correctly receives the data, the EP0NKW bit of the UF0E0N register is automatically cleared to 0 by hardware. A short packet is transmitted when data is written to the UF0E0W register and the E0DED bit of the UF0DEND register is set to 1 (EP0W bit of the UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0E0W register is cleared and the E0DED bit of the UF0DEND register is set to 1 (EP0W bit of the UF0EPS0 register = 1 (data exists)).

The UF0E0W register is cleared to 0 when the next SETUP token is received while transmission has not been completed yet. If the stage of control transfer (read) changes to the status stage while ACK has not been correctly received in the data stage, the UF0E0W register is automatically cleared to 0. At the same time, it is also cleared to 0 if the EP0NKW bit of the UF0E0N register is 1.

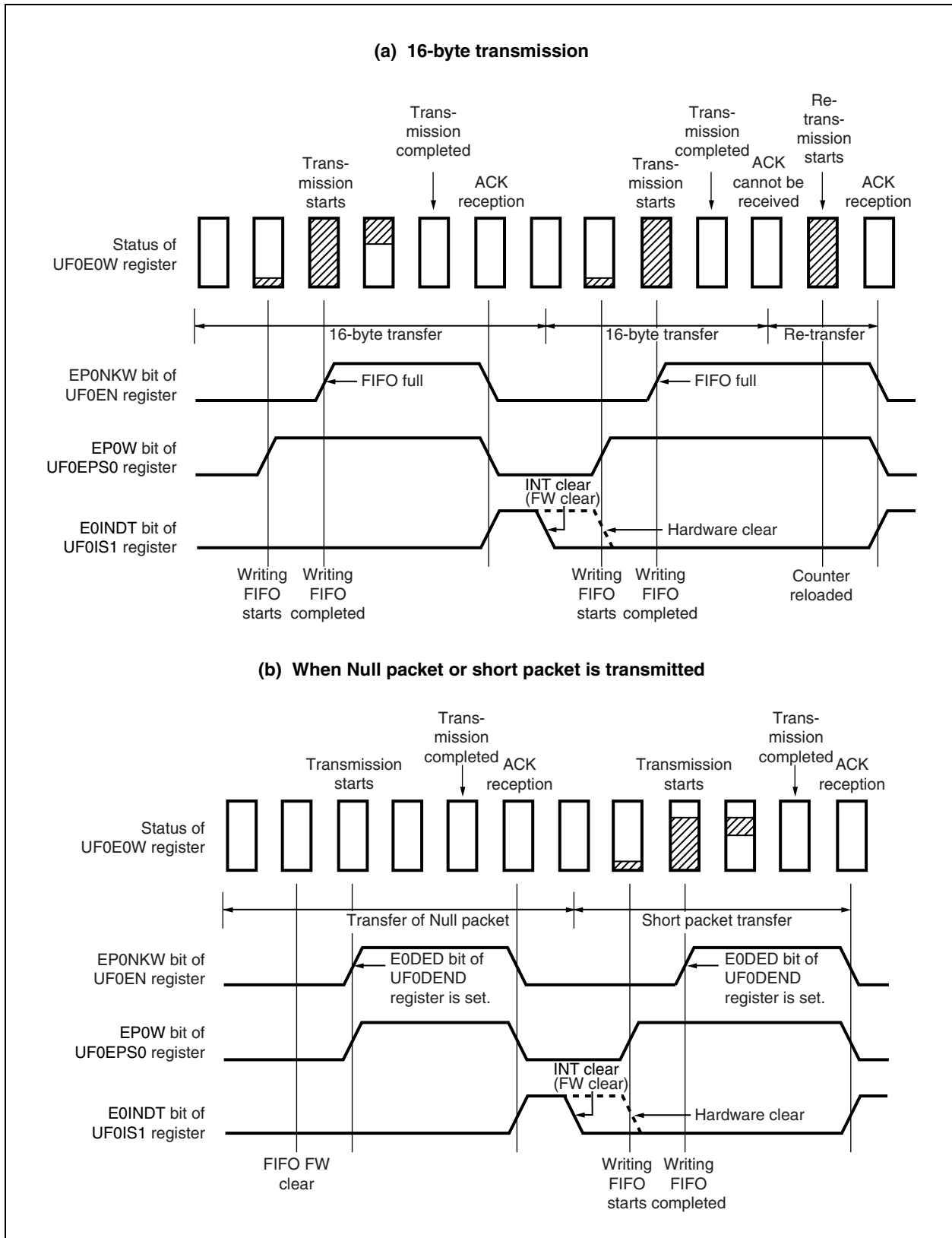
If the UF0E0W register is read while no data is in it, 00H is read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0W	E0W7	E0W6	E0W5	E0W4	E0W3	E0W2	E0W1	E0W0	FFFFFE83H	Undefined

Bit position	Bit name	Function
7 to 0	E0W7 to E0W0	These bits store the IN data sent from the host to Endpoint0 in the data stage.

The operation of the UF0E0W register is illustrated below.

Figure 11-3. Operation of UF0E0W Register



(5) UF0 bulk out 1 register (UF0BO1)

The UF0BO1 register is a 64-byte × 2 FIFO that stores data for Endpoint2. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when data is in the FIFO on the SIE side and when no data is in the FIFO on the CPU side (counter value = 0).

This register is read-only, in 8-bit units. A write access to this register is ignored.

When the hardware receives data for Endpoint2 from the host, it automatically transfers the data to the UF0BO1 register. When the register correctly receives the data, a FIFO toggle operation occurs. As a result, the BKO1DT bit of the UF0IS3 register is set to 1, the quantity of the received data is held by the UF0BO1L register, and an interrupt request or DMA request is issued to the CPU. Whether the interrupt request or DMA request is issued can be selected by using the DQBO1MS bit of the UF0IDR register.

Read the data held by the UF0BO1 register by FW, up to the value of the amount of data read by the UF0BO1L register. When the correct received data is held by the FIFO connected to the SIE side and the value of the UF0BO1L register reaches 0, the toggle operation of the FIFO occurs, and the BKO1NK bit of the UF0EN register is automatically cleared to 0. If data greater than the value of the UF0BO1L register is read and if the FIFO toggle condition is satisfied, the toggle operation of the FIFO occurs. As a result, the next packet may be read by mistake. Note that, if the toggle condition is not satisfied, the first data is repeatedly read.

If overrun data is received while data is held by the FIFO connected to the CPU side, Endpoint2 stalls, and the FIFO on the CPU side is cleared.

When the UF0BO1 register is read while no data is in it, an undefined value is read.

Caution Be sure to read all the data stored in this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BO1	BKO17	BKO16	BKO15	BKO14	BKO13	BKO12	BKO11	BKO10	FFFFE84H	Undefined

Bit position	Bit name	Function
7 to 0	BKO17 to BKO10	These bits store data for Endpoint2.

The operation of the UF0BO1 register is illustrated below.

Figure 11-4. Operation of UF0BO1 Register (1/2)

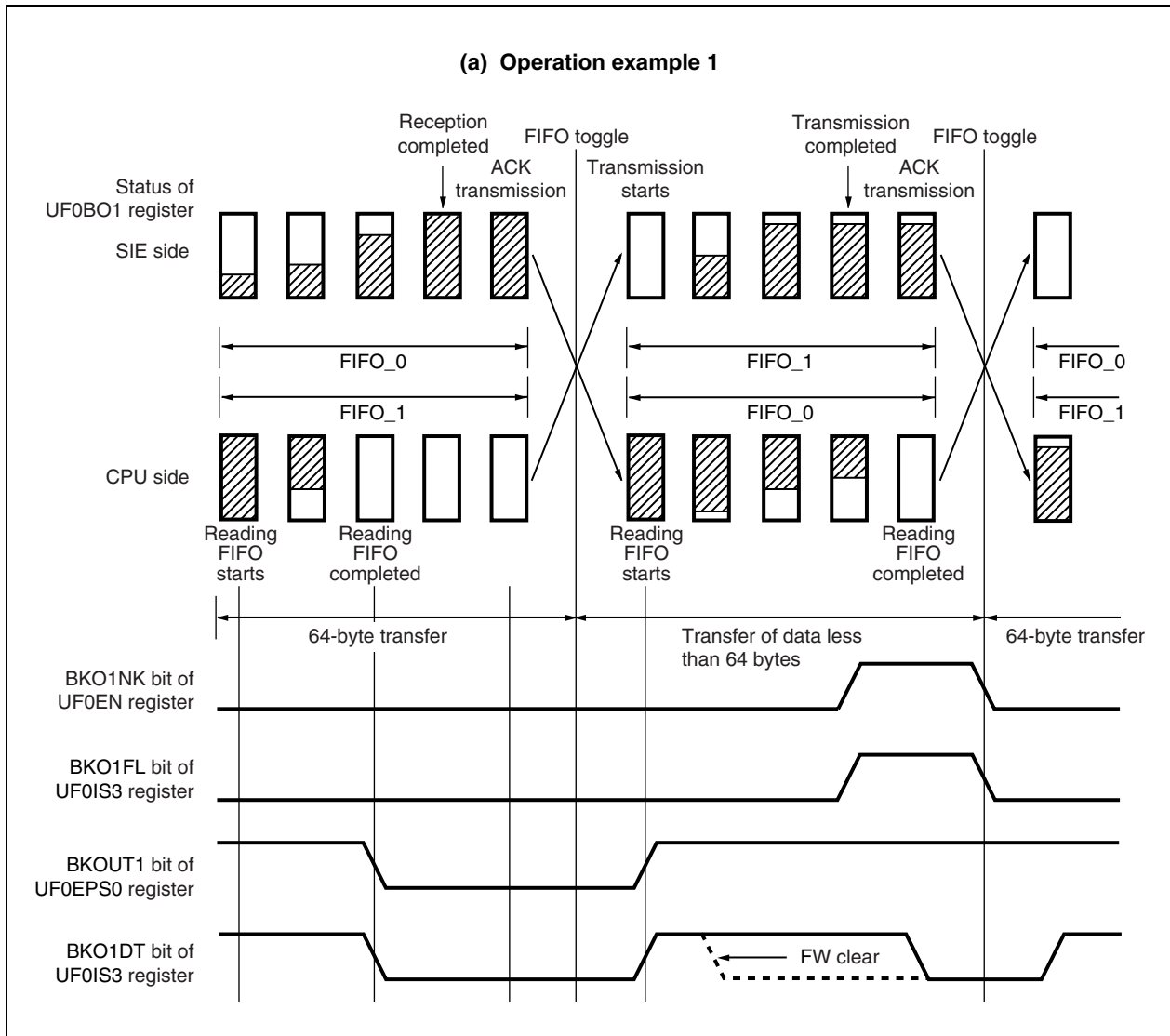
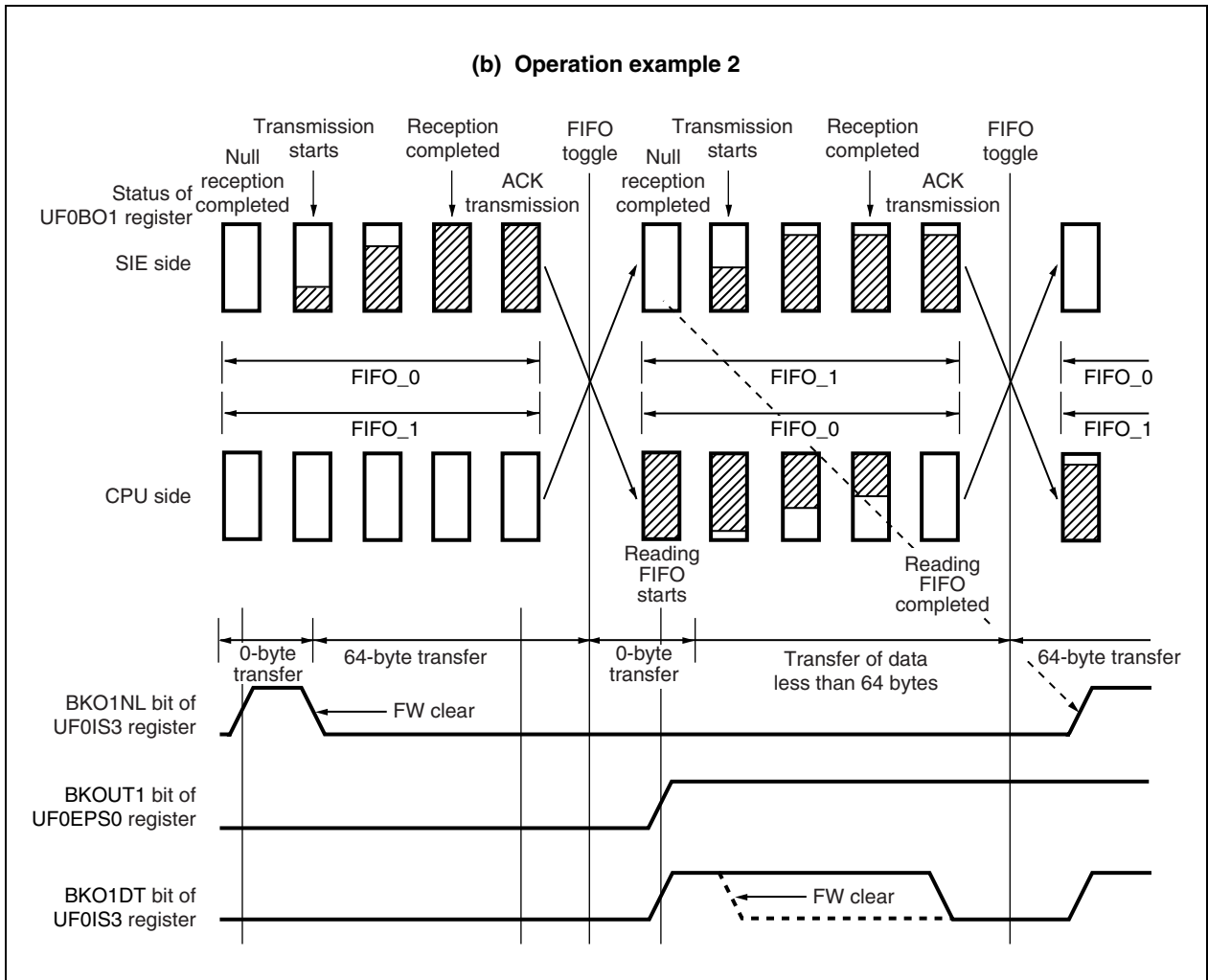


Figure 11-4. Operation of UF0BO1 Register (2/2)



(6) UF0 bulk out 1 length register (UF0BO1L)

The UF0BO1L register stores the length of the data held by the UF0BO1 register.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0BO1L register always updates the received data length while it is receiving data. If the final transfer is abnormal reception, the UF0BO1L register is cleared to 00H, and an interrupt request is not generated. Only if the reception is normal, the interrupt request is generated, and FW can read as much data from the UF0BO1 register as the value read from the UF0BO1L register. The value of the UF0BO1L register is decremented each time the UF0BO1 register has been read.

		7	6	5	4	3	2	1	0	Address	After reset
UF0BO1L		BKO1L7	BKO1L6	BKO1L5	BKO1L4	BKO1L3	BKO1L2	BKO1L1	BKO1L0	FFFFFE85H	00H

Bit position	Bit name	Function
7 to 0	BKO1L7 to BKO1L0	These bits store the length of the data held by the UF0BO1 register.

(7) UF0 bulk out 2 register (UF0BO2)

The UF0BO2 register is a 64-byte × 2 FIFO that stores data for Endpoint4. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when data is in the FIFO on the SIE side and when no data is in the FIFO on the CPU side (counter value = 0).

This register is read-only, in 8-bit units. A write access to this register is ignored.

When the hardware receives data for Endpoint4 from the host, it automatically transfers the data to the UF0BO2 register. When the register correctly receives the data, a FIFO toggle operation occurs. As a result, the BKO2DT bit of the UF0IS3 register is set to 1, the quantity of the received data is held by the UF0BO2L register, and an interrupt request or DMA request is issued to the CPU. Whether the interrupt request or DMA request is issued can be selected by using the DQBO2MS bit of the UF0IDR register.

Read the data held by the UF0BO2 register by FW, up to the value of the amount of data read by the UF0BO2L register. When the correct received data is held by the FIFO connected to the SIE side and the value of the UF0BO2L register reaches 0, the toggle operation of the FIFO occurs, and the BKO2NK bit of the UF0EN register is automatically cleared to 0. If data greater than the value of the UF0BO2L register is read and if the FIFO toggle condition is satisfied, the toggle operation of the FIFO occurs. As a result, the next packet may be read by mistake. Note that, if the toggle condition is not satisfied, the first data is repeatedly read.

If overrun data is received while data is held by the FIFO connected to the CPU side, Endpoint4 stalls, and the FIFO on the CPU side is cleared.

When the UF0BO2 register is read while no data is in it, an undefined value is read.

Caution Be sure to read all the data stored in this register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BO2	BKO27	BKO26	BKO25	BKO24	BKO23	BKO22	BKO21	BKO20	FFFFE86H	Undefined

Bit position	Bit name	Function
7 to 0	BKO27 to BKO20	These bits store data for Endpoint4.

The operation of the UF0BO2 register is illustrated below.

Figure 11-5. Operation of UF0BO2 Register (1/2)

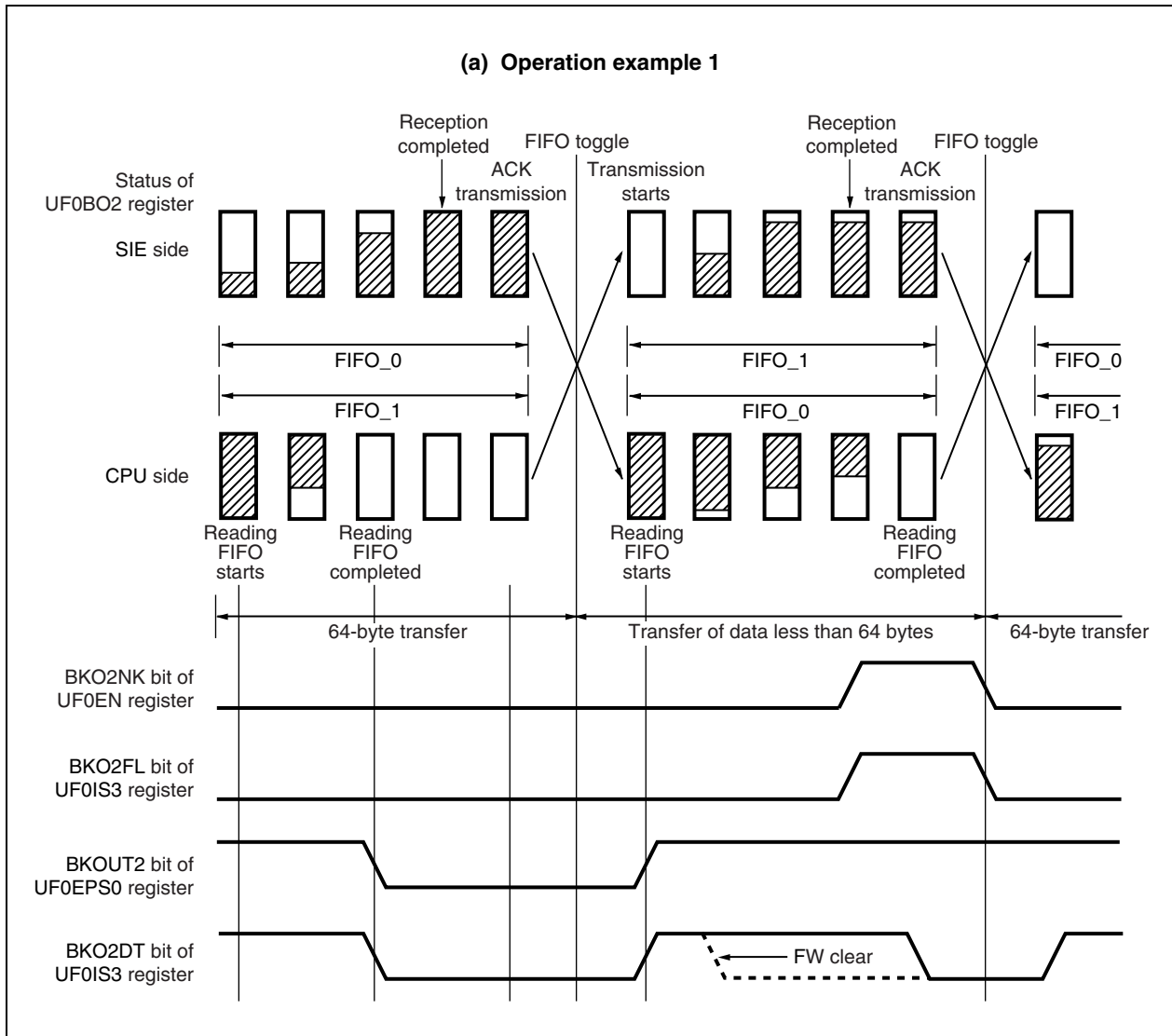
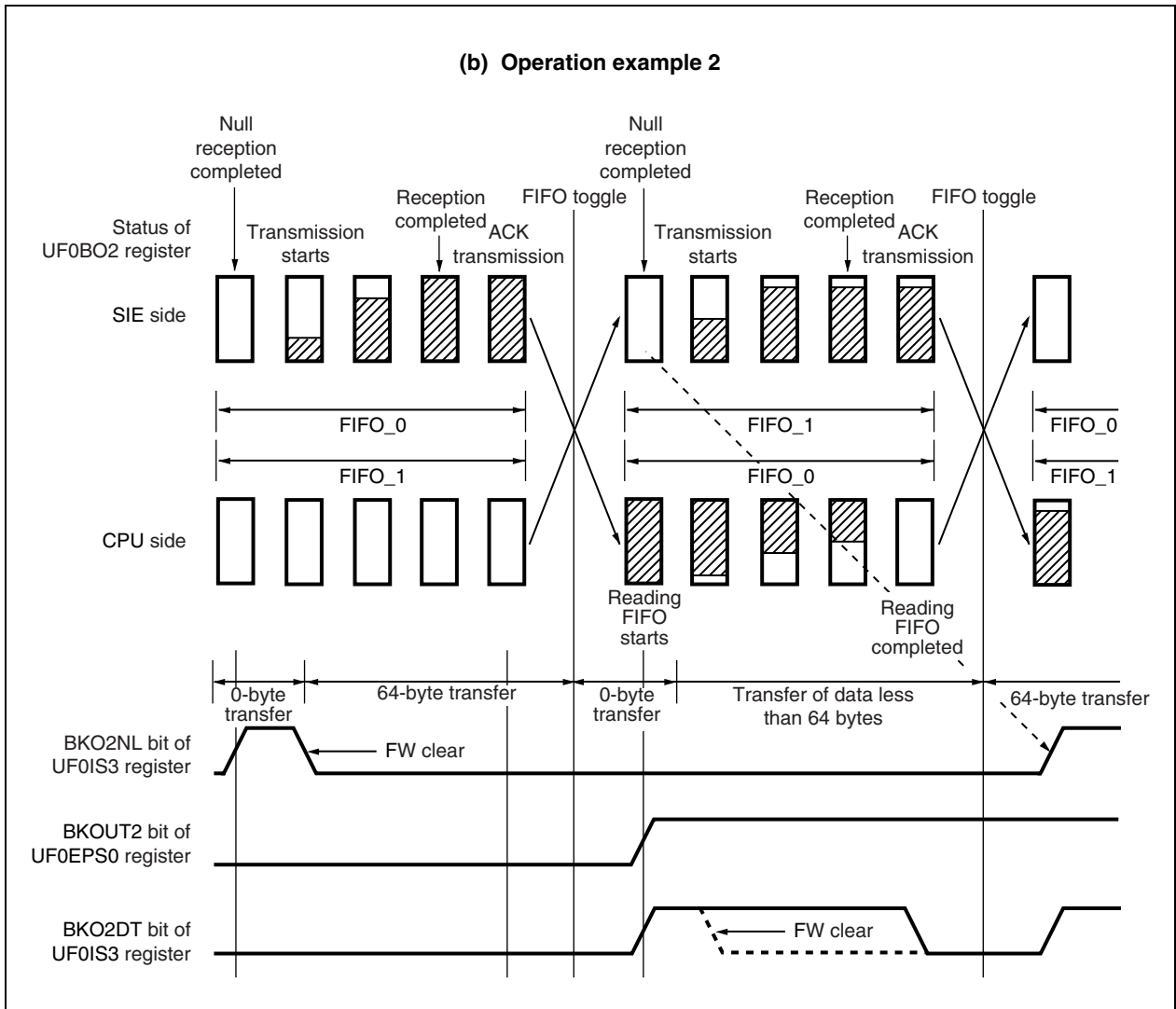


Figure 11-5. Operation of UF0BO2 Register (2/2)



(8) UF0 bulk out 2 length register (UF0BO2L)

The UF0BO2L register stores the length of the data held by the UF0BO2 register.

This register is read-only, in 8-bit units. A write access to this register is ignored.

The UF0BO2L register always updates the received data length while it is receiving data. If the final transfer is abnormal reception, the UF0BO2L register is cleared to 00H, and an interrupt request is not generated. Only if the reception is normal, the interrupt request is generated, and FW can read as much data from the UF0BO2 register as the value read from the UF0BO2L register. The value of the UF0BO2L register is decremented each time the UF0BO2 register has been read.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BO2L	BKO2L7	BKO2L6	BKO2L5	BKO2L4	BKO2L3	BKO2L2	BKO2L1	BKO2L0	FFFFFE87H	00H

Bit position	Bit name	Function
7 to 0	BKO2L7 to BKO2L0	These bits store the length of the data held by the UF0BO2 register.

(9) UF0 bulk in 1 register (UF0BI1)

The UF0BI1 register is a 64-byte × 2 FIFO that stores data for Endpoint1. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when no data is in the FIFO on the SIE side (counter value = 0) and when the FIFO on the CPU side is correctly written (FIFO full or BKI1DED bit = 1).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint1 only when the BKI1NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). The address at which data is to be written or read is managed by the hardware. Therefore, FW can transmit data to the host only by writing the data to the UF0BI1 register sequentially. A short packet is transmitted when data is written to the UF0BI1 register and the BKI1DED bit of the UF0DEND register is set to 1 (BKIN1 bit of UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0BI1 register is cleared and the BKI1DED bit of the UF0DEND register is set to 1 (BKIN1 bit of the UF0EPS0 register = 1 (data exists)). An interrupt request or DMA request can be selected by using the DQBI1MS bit of the UF0IDR register.

	7	6	5	4	3	2	1	0	Address	After reset
UF0BI1	BKI17	BKI16	BKI15	BKI14	BKI13	BKI12	BKI11	BKI10	FFFFFE88H	Undefined

Bit position	Bit name	Function
7 to 0	BKI17 to BKI10	These bits store data for Endpoint1.

The operation of the UF0BI1 register is illustrated below.

Figure 11-6. Operation of UF0B11 Register (1/3)

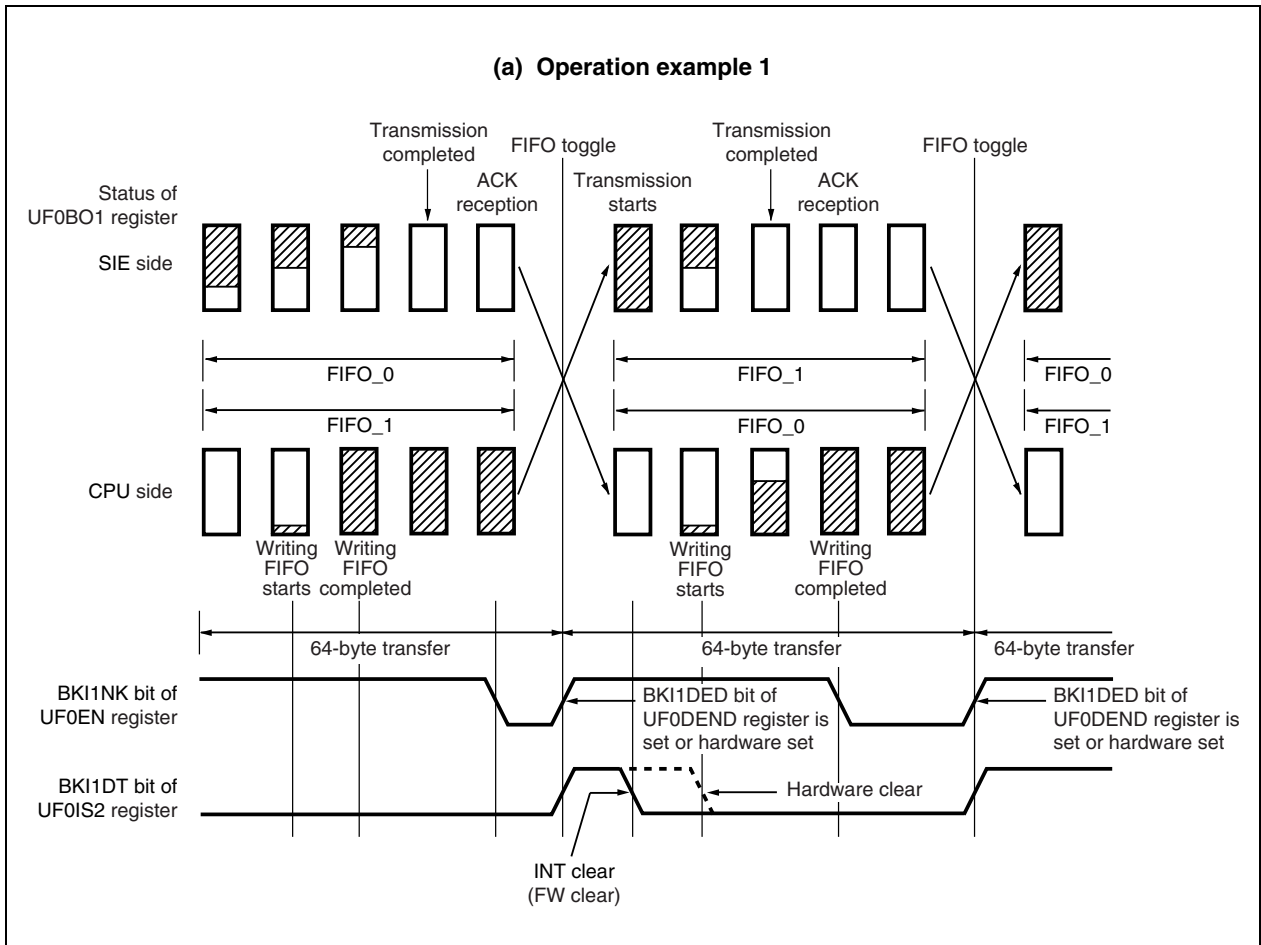


Figure 11-6. Operation of UF0BI1 Register (2/3)

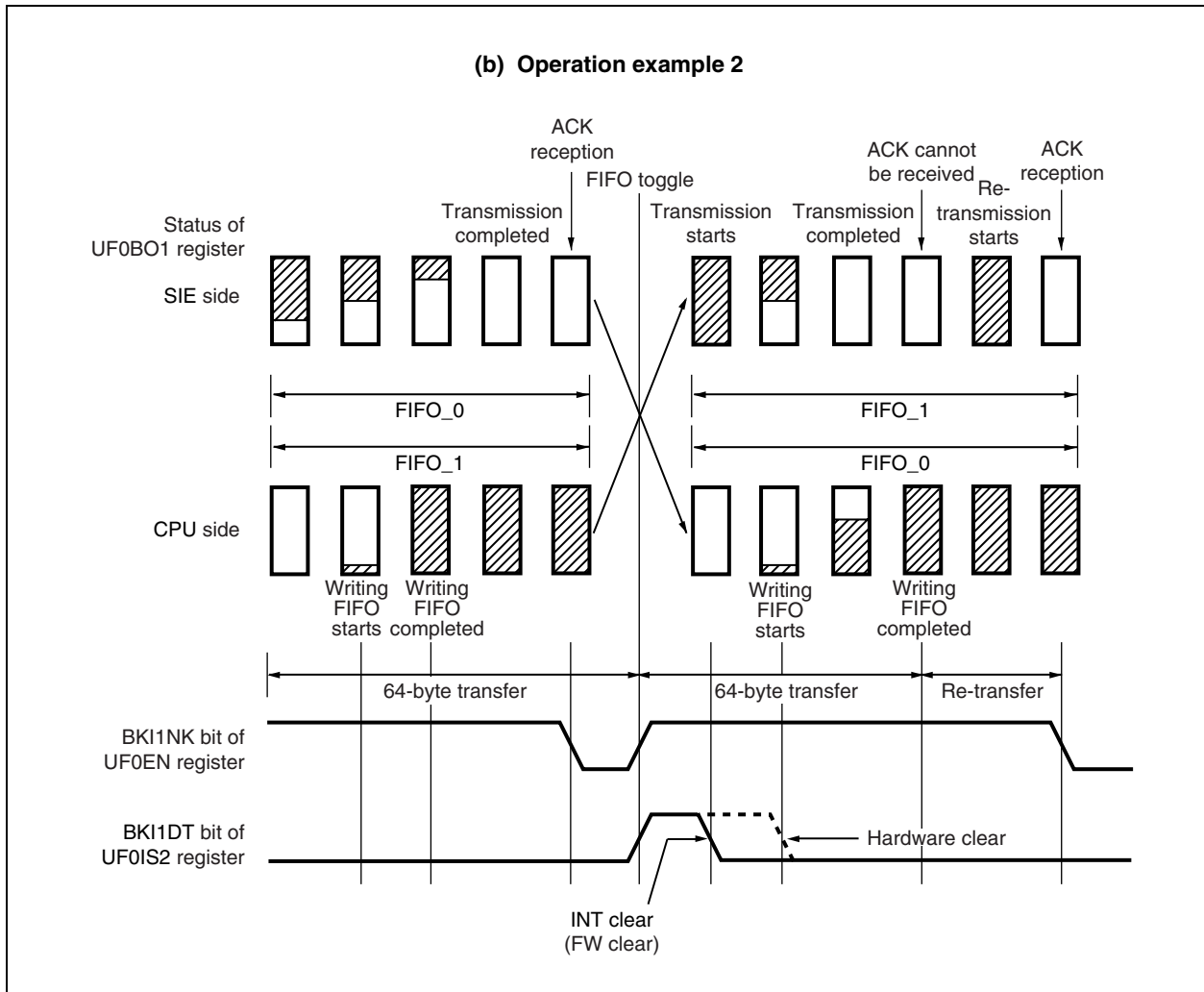
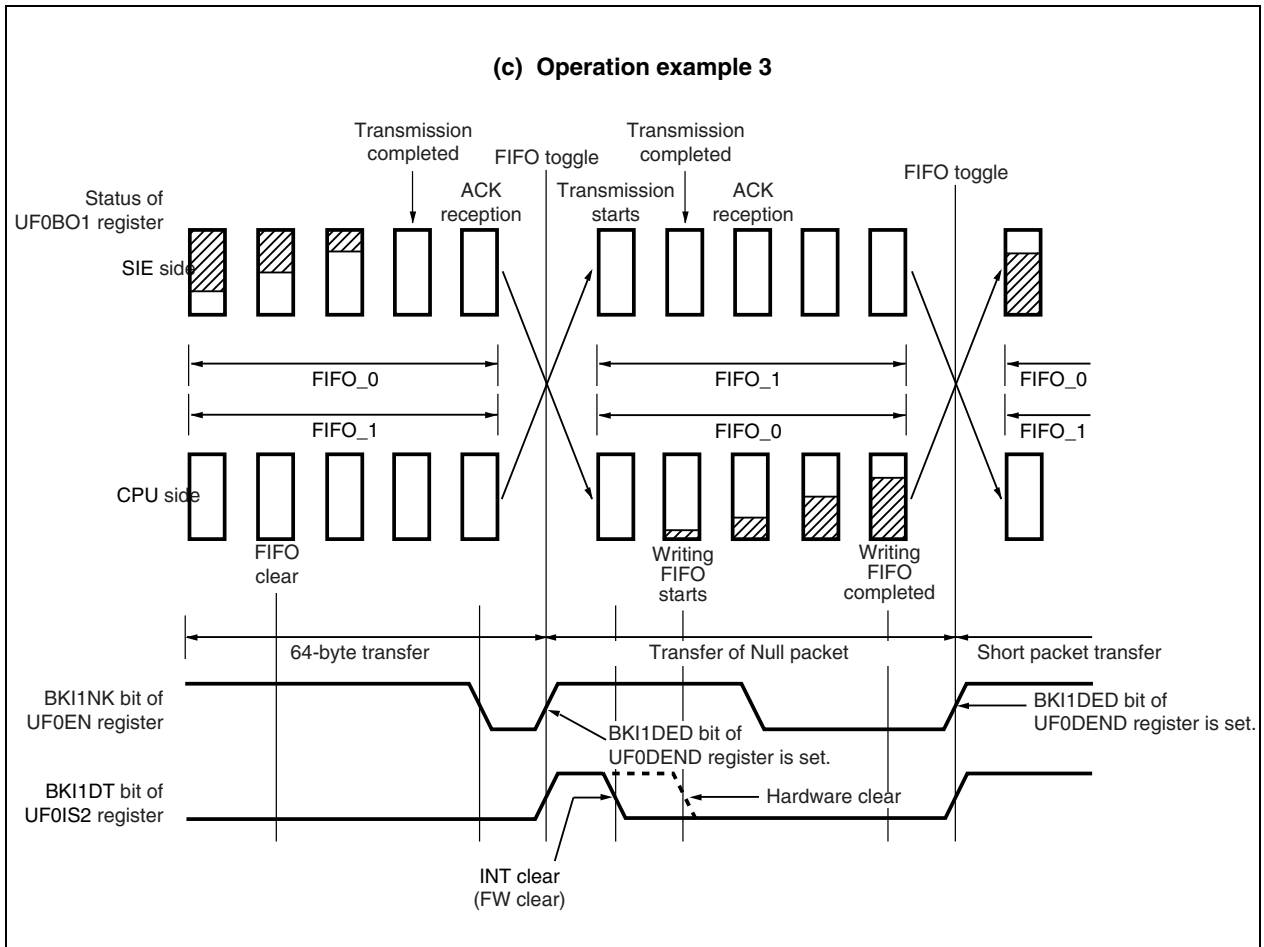


Figure 11-6. Operation of UF0B1 Register (3/3)



(10) UF0 bulk in 2 register (UF0BI2)

The UF0BI2 register is a 64-byte × 2 FIFO that stores data for Endpoint3. This register consists of two banks of 64-byte FIFOs each of which performs a toggle operation and repeatedly connects the buses on the SIE and CPU sides. The toggle operation takes place when no data is in the FIFO on the SIE side (counter value = 0) and when the FIFO on the CPU side is correctly written (FIFO full or BKI2DED bit = 1).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint3 only when the BKI2NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). The address at which data is to be written or read is managed by the hardware. Therefore, FW can transmit data to the host only by writing the data to the UF0BI2 register sequentially. A short packet is transmitted when data is written to the UF0BI2 register and the BKI2DED bit of the UF0DEND register is set to 1 (BKIN2 bit of UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0BI2 register is cleared and the BKI2DED bit of the UF0DEND register is set to 1 (BKIN2 bit of the UF0EPS0 register = 1 (data exists)). An interrupt request or DMA request can be selected by using the DQBI2MS bit of the UF0IDR register.

		7	6	5	4	3	2	1	0	Address	After reset
UF0BI2		BKI27	BKI26	BKI25	BKI24	BKI23	BKI22	BKI21	BKI20	FFFFFFE89H	Undefined

Bit position	Bit name	Function
7 to 0	BKI27 to BKI20	These bits store data for Endpoint3.

The operation of the UF0BI2 register is illustrated below.

Figure 11-7. Operation of UF0BI2 Register (1/3)

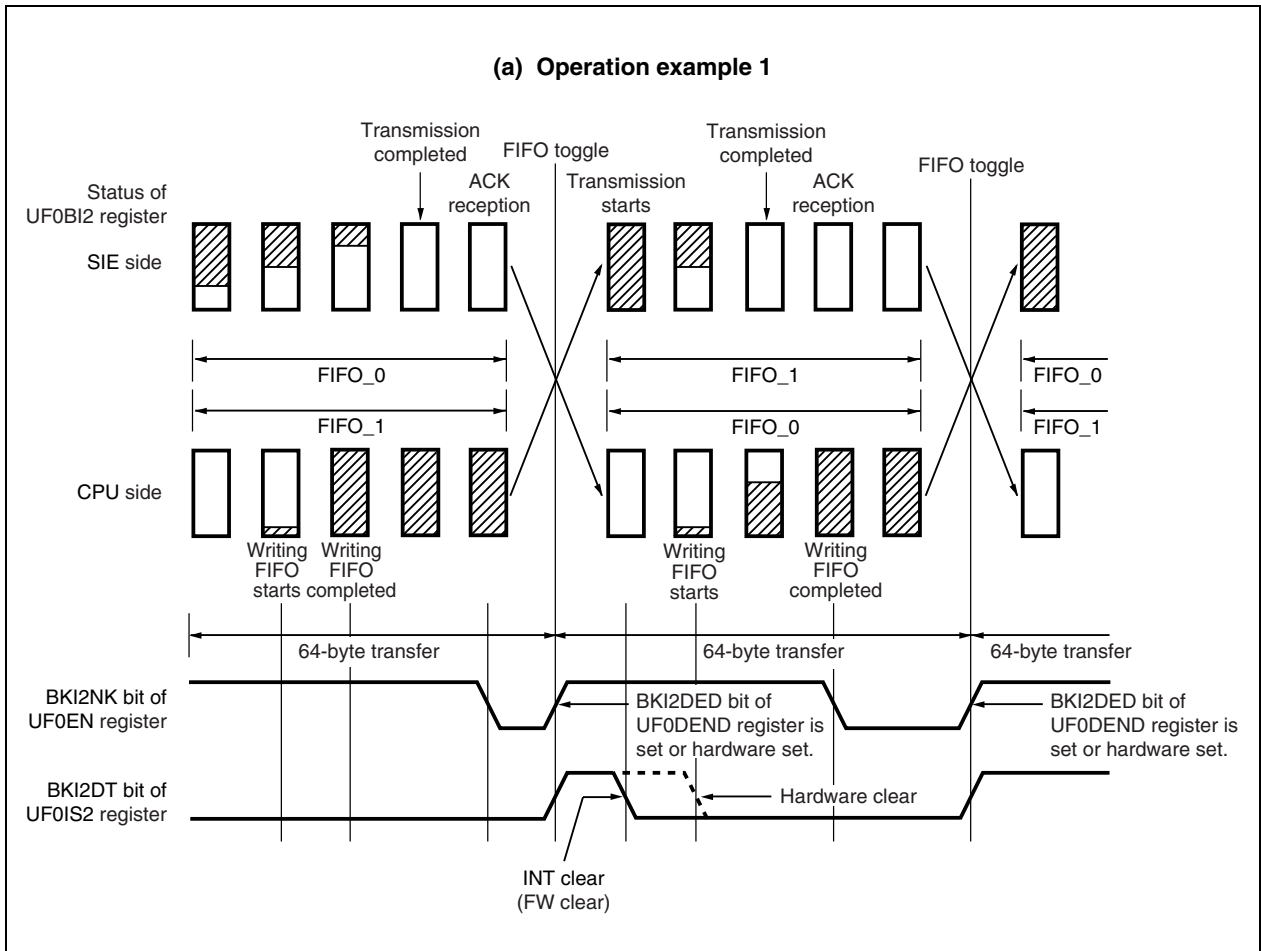


Figure 11-7. Operation of UF0BI2 Register (2/3)

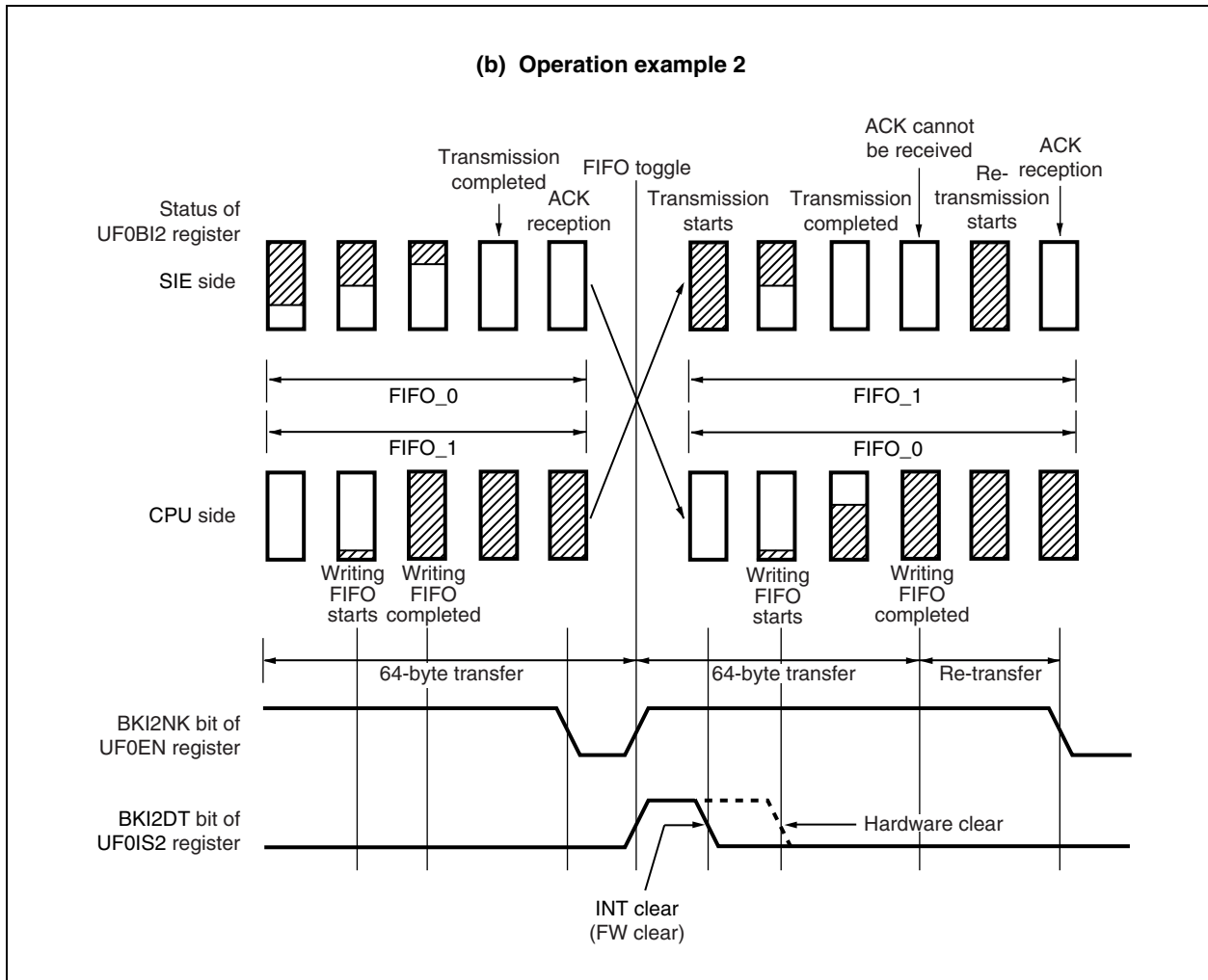
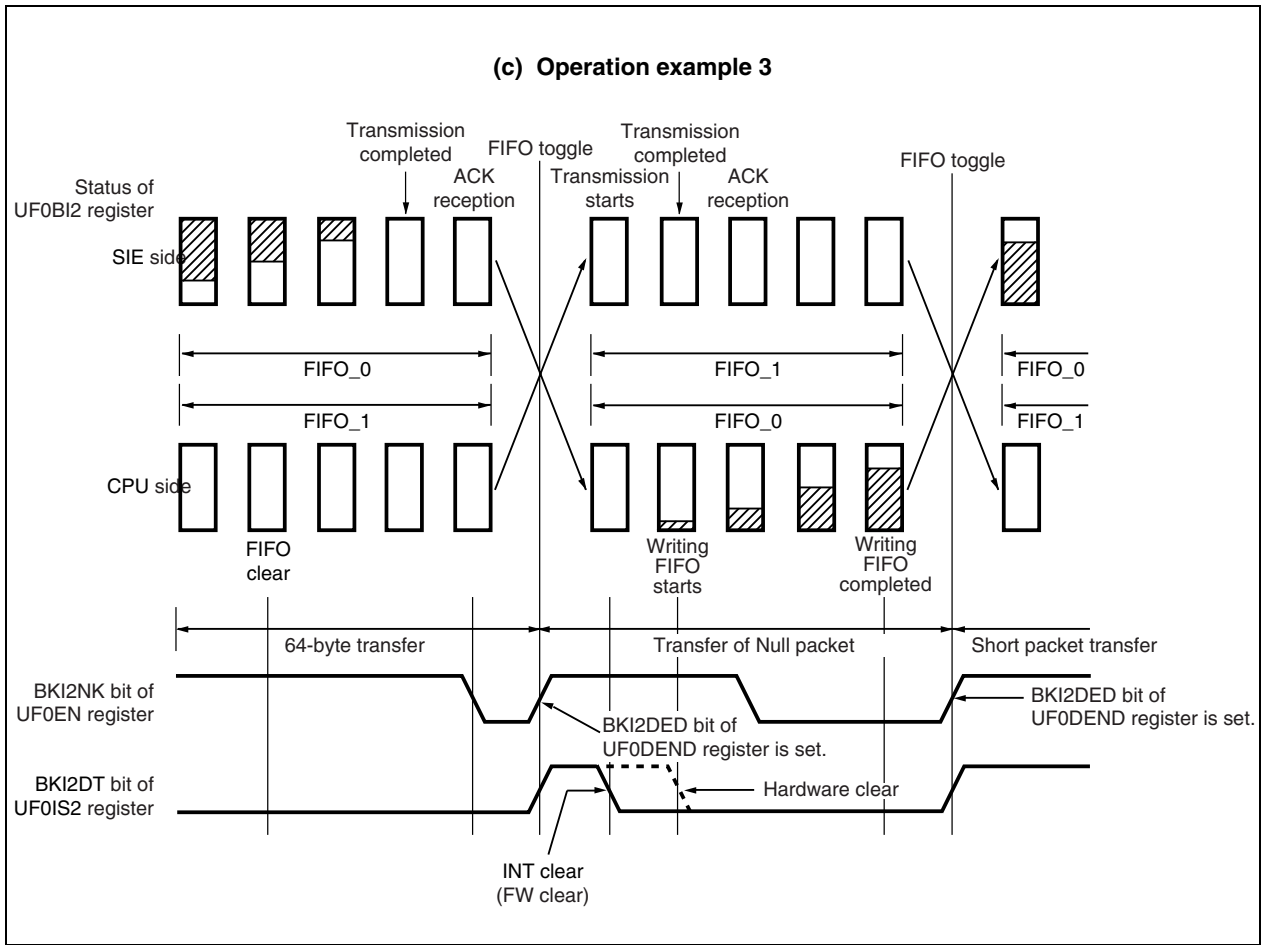


Figure 11-7. Operation of UF0BI2 Register (3/3)



(11) UF0 interrupt 1 register (UF0INT1)

The UF0INT1 register is an 8-byte FIFO that stores data for Endpoint7 (to be passed to SIE).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

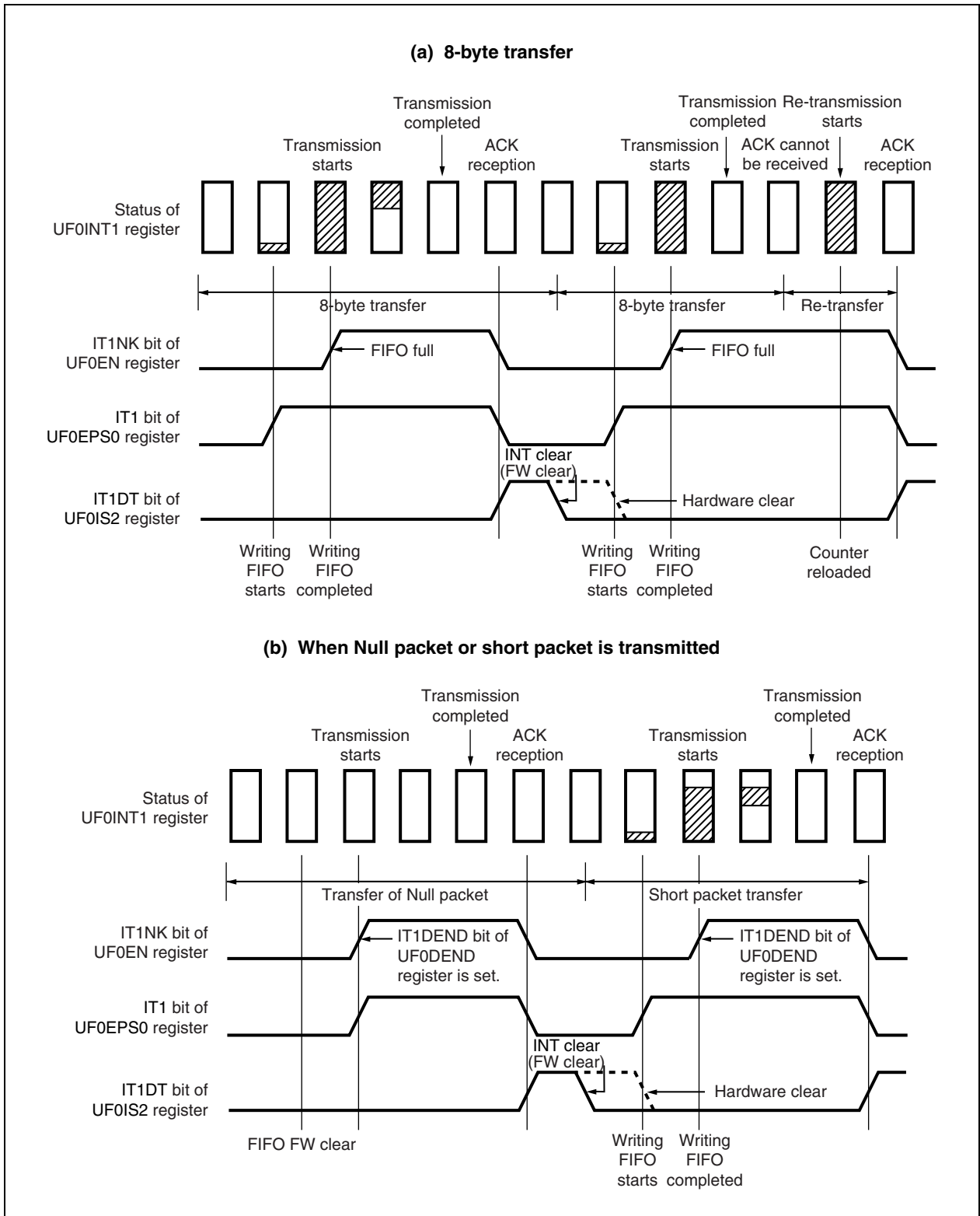
The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint7 only when the IT1NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). When the data is transmitted and the host correctly receives it, the IT1NK bit of the UF0EN register is automatically cleared to 0 by hardware. A short packet is transmitted when data is written to the UF0INT1 register and the IT1DEND bit of the UF0DEND register is set to 1 (IT1 bit of the UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0INT1 register is cleared and the IT1DEND bit of the UF0DEND register is set to 1 (IT1 bit of the UF0EPS0 register = 1 (data exists)).

	7	6	5	4	3	2	1	0	Address	After reset
UF0INT1	IT17	IT16	IT15	IT14	IT13	IT12	IT11	IT10	FFFFFE8AH	Undefined

Bit position	Bit name	Function
7 to 0	IT17 to IT10	These bits store data for Endpoint7.

The operation of the UF0INT1 register is illustrated below.

Figure 11-8. Operation of UF0INT1 Register



(12) UF0 interrupt 2 register (UF0INT2)

The UF0INT2 register is an 8-byte FIFO that stores data for Endpoint8 (to be passed to SIE).

This register is write-only, in 8-bit units. When this register is read, 00H is read.

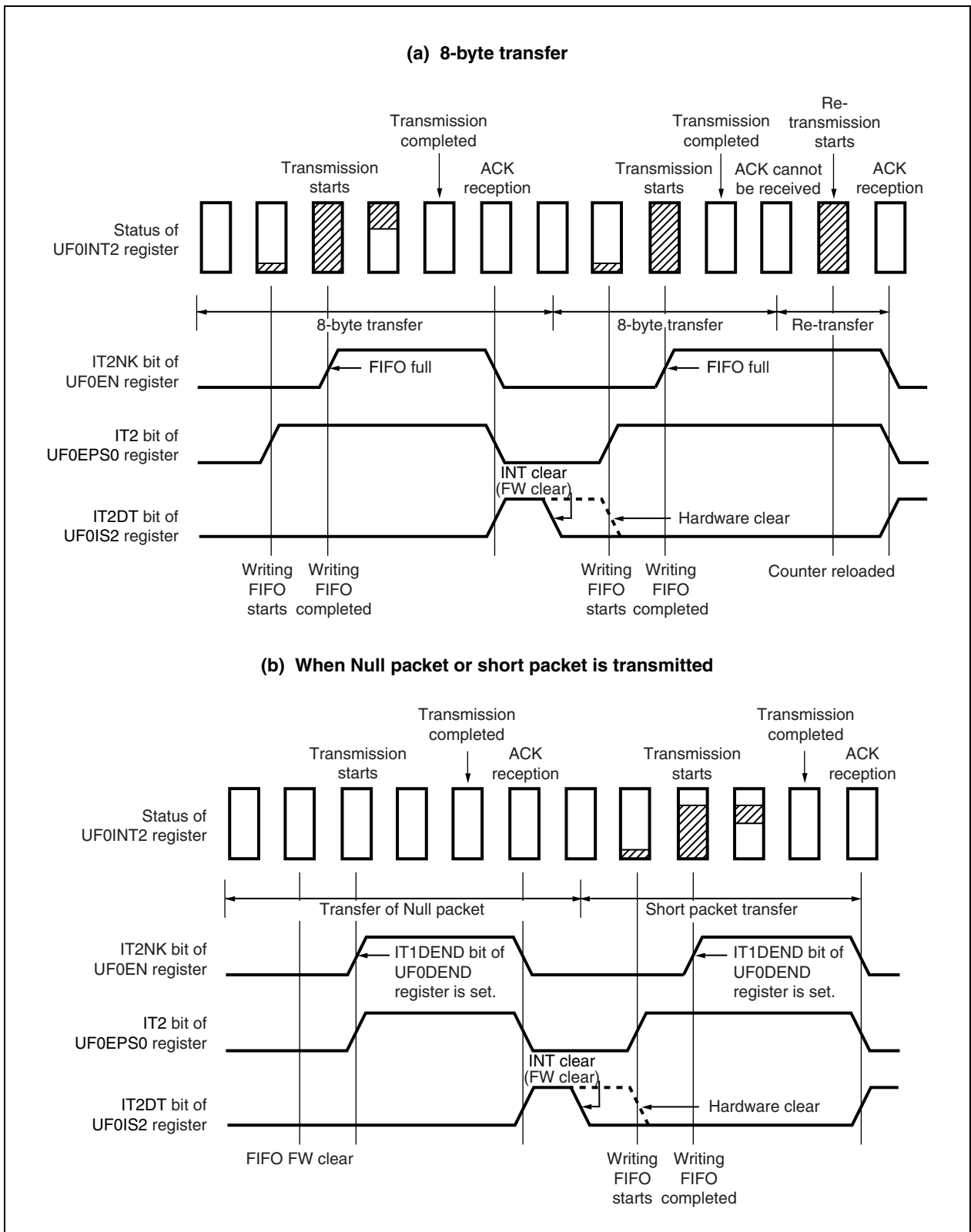
The hardware transmits data to the USB bus in synchronization with the IN token for Endpoint8 only when the IT2NK bit of the UF0EN register is set to 1 (when NAK is not transmitted). When the data is transmitted and the host correctly receives it, the IT2NK bit of the UF0EN register is automatically cleared to 0 by hardware. A short packet is transmitted when data is written to the UF0INT2 register and the IT2DEND bit of the UF0DEND register is set to 1 (IT2 bit of the UF0EPS0 register = 1 (data exists)). A Null packet is transmitted when the UF0INT2 register is cleared and the IT2DEND bit of the UF0DEND register is set to 1 (IT2 bit of the UF0EPS0 register = 1 (data exists)).

	7	6	5	4	3	2	1	0	Address	After reset
UF0INT2	IT27	IT26	IT25	IT24	IT23	IT22	IT21	IT20	FFFFFFE8BH	Undefined

Bit position	Bit name	Function
7 to 0	IT27 to IT20	These bits store data for Endpoint8.

The operation of the UF0INT2 register is illustrated below.

Figure 11-9. Operation of UF0INT2 Register



11.4.3 Request data register area

(1) UF0 device status register L (UF0DSTL)

This register stores the value that is to be returned in response to the GET_STATUS Device request.

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

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Device request.

Caution To rewrite this register, set the EPONKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DSTL	0	0	0	0	0	0	RMWK	SFPW	FFFFEA2H	00H

Bit position	Bit name	Function
1	RMWK	This bit specifies whether the remote wakeup function of the device is used. 1: Enabled 0: Disabled If the device supports a remote wakeup function, this bit is set to 1 by hardware when the SET_FEATURE Device request has been received, and is cleared to 0 by hardware when the CLEAR_FEATURE Device request has been received. If the device does not support a remote wakeup function, make sure that the SET_FEATURE Device request is not issued from the host.
0	SFPW	This bit indicates whether the device is self-powered or bus-powered. 1: Self-powered 0: Bus-powered

(2) UF0 EP0 status register L (UF0E0SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint0 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in USBF, the E0HALT bit is set to 1 by FW. A write access to this register is ignored while a USB-side access to Endpoint0 is being received.

When the E0HALT bit is set to 1 by FW, it is not reflected until the next SETUP token is received if the control transfer immediately before is for the SET_FEATURE Endpoint0, CLEAR_FEATURE Endpoint0, GET_STATUS Endpoint0 request, or an FW-processed request.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint0 request. If Endpoint0 has stalled, the UF0E0W and UF0E0R registers are cleared, and the EP0NKW and EP0NKR bits of the UF0E0N register are cleared to 0.

★

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E0SL	0	0	0	0	0	0	0	E0HALT	FFFFEA6H	00H

Bit position	Bit name	Function
0	E0HALT	This bit indicates the status of Endpoint0. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint0 request has been received, and cleared to 0 by hardware when the CLEAR_FEATURE Endpoint0 request has been received. DATA PID is initialized to DATA0.

(3) UF0 EP1 status register L (UF0E1SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint1 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint1, the E1HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint1 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint1 request. If Endpoint1 has stalled, the UF0B11 register is cleared and the BK11NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint1, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E1SL	0	0	0	0	0	0	0	E1HALT	FFFFFEA8H	00H

Bit position	Bit name	Function
0	E1HALT	This bit indicates the status of Endpoint1. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint1 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint1 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint1 is linked has correctly been received. DATA PID is initialized to DATA0.

(4) UF0 EP2 status register L (UF0E2SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint2 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint2, the E2HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint2 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint2 request. If Endpoint2 has stalled, the UF0BO1 register is cleared and the BKO1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint2, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E2SL	0	0	0	0	0	0	0	E2HALT	FFFFFFEAH	00H

Bit position	Bit name	Function
0	E2HALT	This bit indicates the status of Endpoint2. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint2 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint2 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint2 is linked has correctly been received. DATA PID is initialized to DATA0.

(5) UF0 EP3 status register L (UF0E3SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint3 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint3, the E3HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint3 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint3 request. If Endpoint3 has stalled, the UF0BI2 register is cleared and the BKI2NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint3, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E3SL	0	0	0	0	0	0	0	E3HALT	FFFFFEACH	00H

Bit position	Bit name	Function
0	E3HALT	This bit indicates the status of Endpoint3. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint3 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint3 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint3 is linked has correctly been received. DATA PID is initialized to DATA0.

(6) UF0 EP4 status register L (UF0E4SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint4 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint4, the E4HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint4 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint4 request. If Endpoint4 has stalled, the UF0BO2 register is cleared and the BKO2NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint4, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E4SL	0	0	0	0	0	0	0	E4HALT	FFFFFFEAH	00H

Bit position	Bit name	Function
0	E4HALT	This bit indicates the status of Endpoint4. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint4 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint4 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint4 is linked has correctly been received. DATA PID is initialized to DATA0.

(7) UF0 EP7 status register L (UF0E7SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint7 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint7, the E7HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint7 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint7 request. If Endpoint7 has stalled, the UF0INT1 register is cleared and the IT1NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint7, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E7SL	0	0	0	0	0	0	0	E7HALT	FFFFFFEB4H	00H

Bit position	Bit name	Function
0	E7HALT	This bit indicates the status of Endpoint7. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint7 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint7 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint7 is linked has correctly been received. DATA PID is initialized to DATA0.

(8) UF0 EP8 status register L (UF0E8SL)

This register stores the value that is to be returned in response to the GET_STATUS Endpoint8 request.

This register can be read or written in 8-bit units. Note, however, that data can be written to this register only when the EP0NKA bit is set to 1.

If an error occurs in Endpoint8, the E8HALT bit is set to 1. A write access to this register is ignored while a USB-side access to Endpoint8 is being received.

The hardware automatically transmits the contents of this register to the host when it has received the GET_STATUS Endpoint8 request. If Endpoint8 has stalled, the UF0INT2 register is cleared and the IT2NK bit is cleared to 0.

Because writing this register is always masked when transfer to Endpoint8, rather than control transfer, is executed, be sure to check this register to see if data has been correctly written to it.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0E8SL	0	0	0	0	0	0	0	E8HALT	FFFFFFEB6H	00H

Bit position	Bit name	Function
0	E8HALT	This bit indicates the status of Endpoint8. 1: Stalled 0: Not stalled This bit is set to 1 by hardware when the SET_FEATURE Endpoint8 request has been received. It is cleared to 0 by hardware when the CLEAR_FEATURE Endpoint8 request, SET_CONFIGURATION request, or the SET_INTERFACE request for the Interface to which Endpoint8 is linked has correctly been received. DATA PID is initialized to DATA0.

(9) UF0 address register (UF0ADRS)

This register stores the device address.

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

The device address sent by the SET_ADDRESS request is analyzed and the resultant value is automatically written to this register. If the SET_ADDRESS request is processed by FW, the value of this register is reflected as the device address when the SUCCESS signal is received in the status stage.

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0ADRS	0	ADRS6	ADRS5	ADRS4	ADRS3	ADRS2	ADRS1	ADRS0	FFFFFFE0H	00H

Bit position	Bit name	Function
6 to 0	ADRS6 to ADRS0	These bits hold the device address of SIE.

(10) UF0 configuration register (UF0CNF)

This register stores the value that is to be returned in response to the GET_CONFIGURATION request.

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

When the SET_CONFIGURATION request is received, its wValue is automatically written to this register.

To change the value of this register by FW after a value other than 00H has been written to the register, write 00H once and then write the desired value. When a change of the value of this register from 00H to other than 00H is detected, the CONF bits are set to 1. If the SET_CONFIGURATION request is processed by FW, the status of this register is immediately reflected on the UF0MODS register as soon as data has been written to this register (CONF bits = 1 before completion of the status stage).

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0CNF	0	0	0	0	0	0	CONF1	CONF0	FFFFFEC1H	00H

Bit position	Bit name	Function
1, 0	CONF1, CONF0	These bits hold the data to be returned in response to the GET_CONFIGURATION request.

(11) UF0 interface 0 register (UF0IF0)

This register stores the value that is to be returned in response to the GET_INTERFACE wIndex = 0 request.

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

When the SET_INTERFACE request is received, its wValue is automatically written to this register.

If the SET_INTERFACE request is processed by FW, wIndex and wValue are decoded, and the setting of endpoint is automatically changed. At this time, the status bit of the target endpoint and DPID are automatically cleared to 0, depending on the setting. The FIFO is not cleared automatically.

Caution To rewrite this register, set the EPONKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IF0	0	0	0	0	0	IF02	IF01	IF00	FFFFFFEC2H	00H

Bit position	Bit name	Function
2 to 0	IF02 to IF00	These bits hold the data to be returned in response to GET_INTERFACE wIndex = 0 request.

(12) UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4)

These registers store the value that is to be returned in response to the GET_INTERFACE wIndex = n request (n = 1 to 4).

★

These registers are read-only, in 8-bit units.

When the SET_INTERFACE request is received, its wValue is automatically written to these registers.

These registers are invalidated according to the setting of the UF0AIFN and UF0AAS registers.

If the SET_INTERFACE request is processed by FW, wIndex and wValue are decoded, and the setting of endpoint is automatically changed. At this time, the status bit of the target endpoint and DPID are automatically cleared to 0, depending on the setting. The FIFO is not cleared automatically.

Caution To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0IF1	0	0	0	0	0	IF12	IF11	IF10	FFFFFEC3H	00H
UF0IF2	0	0	0	0	0	IF22	IF21	IF20	FFFFFEC4H	00H
UF0IF3	0	0	0	0	0	IF32	IF31	IF30	FFFFFEC5H	00H
UF0IF4	0	0	0	0	0	IF42	IF41	IF40	FFFFFEC6H	00H

Bit position	Bit name	Function
2 to 0	IFn2 to IFn0	These bits hold the data to be returned in response to GET_INTERFACE wIndex = n request.

Remark n = 1 to 4

(13) UF0 descriptor length register (UF0DSCL)

This register stores the length of the value that is to be returned in response to the GET_DESCRIPTOR Configuration request. The value of this register is the number of bytes of all the descriptors set by the UF0CIEn register minus 1 (n = 0 to 255). The total descriptor length that is to be returned in response to the GET_DESCRIPTOR Configuration request is determined according to the value of this register.

This register can be read or written in 8-bit units. However, data can be written to this register only when the EP0NKA bit is set to 1.

Processing of wLength is automatically controlled. If this register is set to 00H, it means that the descriptor to be returned is 1 byte long. If the register is set to FFH, a descriptor length of 256 bytes is returned. When a descriptor exceeding 256 bytes in length is used, set the CDCGDST bit of the UF0MODC register to 1 and process the GET_DESCRIPTOR request by FW (at this time, the CDCGD bit of the UF0MODS register is also set to 1).

Caution To rewrite this register, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

	7	6	5	4	3	2	1	0	Address	After reset
UF0DSCL	DPL7	DPL6	DPL5	DPL4	DPL3	DPL2	DPL1	DPL0	FFFFFFD0H	00H

Bit position	Bit name	Function
7 to 0	DPL7 to DPL0	These bits set the value of the number of bytes of all the descriptors to be returned in response to the GET_DESCRIPTOR Configuration request minus 1.

(14) UF0 device descriptor registers 0 to 17 (UF0DD0 to UF0DD17)

These registers store the value to be returned in response to the GET_DESCRIPTOR Device request.

These registers can be read or written in 8-bit units. However, data can be written to these registers only when the EP0NKA bit is set to 1.

Cautions 1. To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.

★

2. Use the value defined by USB Specification Ver. 2.0 and the latest Class Specification as the set value.

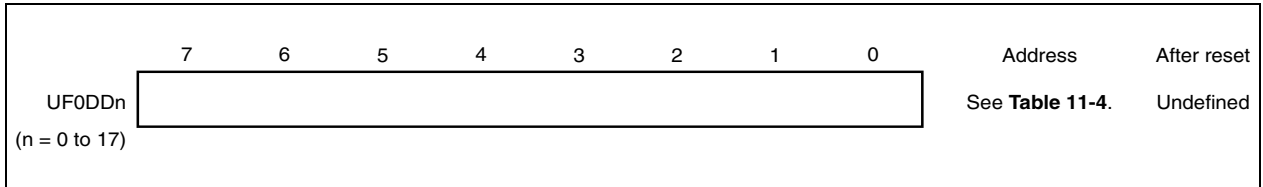


Table 11-4. Mapping and Data of UF0 Device Descriptor Registers

Symbol	Address	Field Name	Contents
UF0DD0	FFFFFFED1H	bLength	Size of this descriptor
UF0DD1	FFFFFFED2H	bDescriptorType	Device descriptor type
UF0DD2	FFFFFFED3H	bcdUSB	Value below decimal point of Rev. number of USB specification
UF0DD3	FFFFFFED4H		Value above decimal point of Rev. number of USB specification
UF0DD4	FFFFFFED5H	bDeviceClass	Class code
UF0DD5	FFFFFFED6H	bDeviceSubClass	Subclass code
UF0DD6	FFFFFFED7H	bDeviceProtocol	Protocol code
UF0DD7	FFFFFFED8H	bMaxPacketSize0	Maximum packet size of Endpoint0
UF0DD8	FFFFFFED9H	idVendor	Lower value of vendor ID
UF0DD9	FFFFFFEDA H		Higher value of vendor ID
UF0DD10	FFFFFFEDBH	idProduct	Lower value of product ID
UF0DD11	FFFFFFEDCH		Higher value of product ID
UF0DD12	FFFFFFEDDH	bcdDevice	Lower value of device release number
UF0DD13	FFFFFFEDEH		Higher value of device release number
UF0DD14	FFFFFFEDFH	iManufacturer	Index of string descriptor describing manufacturer
UF0DD15	FFFFFFEE0H	iProduct	Index of string descriptor describing product
UF0DD16	FFFFFFEE1H	ISerialNumber	Index of string descriptor describing device serial number
UF0DD17	FFFFFFEE2H	BNumConfigurations	Number of settable configurations

(15) UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255)

These registers store the value to be returned in response to the GET_DESCRIPTOR Configuration request. These registers can be read or written in 8-bit units. However, data can be written to these registers only when the EP0NKA bit is set to 1.

Descriptor information of up to 256 bytes can be stored in these registers. Store each descriptor in the order of Configuration, Interface, and Endpoint (see **Table 11-5**). If there are two or more Interfaces, repeatedly store the data following the Interface descriptor.

Table 11-5. Mapping of UF0CIEn Register

Address	Descriptor Stored
FFFFFFE3H	Configuration descriptor (9 bytes)
FFFFFFE4H	Interface descriptor (9 bytes)
FFFFFFE5H	Endpoint1 descriptor (7 bytes)
FFFFFFE6H	Endpoint2 descriptor (7 bytes)
FFFFFFE7H	Endpoint3 descriptor (7 bytes)
:	:
FFFFFFxxH	Interface descriptor (9 bytes)
FFFFFFxxH+9	Endpoint1 descriptor (7 bytes)
FFFFFFxxH+16	Endpoint2 descriptor (7 bytes)
FFFFFFxxH+23	Endpoint3 descriptor (7 bytes)
:	:

The range of the valid data that can be set to these registers varies according to the setting of the UF0DSCL register. In addition to the descriptors listed in Table 11-6, descriptors peculiar to classes and vendors can also be stored.

If all the values are fixed, they can be stored in ROM.

- Cautions**
- 1. To rewrite these registers, set the EP0NKA bit to 1 before reading the register contents, and rewrite the register contents after confirming that the bit has been set, in order to prevent conflict between a read access and a write access.**
 - 2. Use the value defined by USB Specification Ver. 2.0 and the latest Class Specification as the set value.**

★

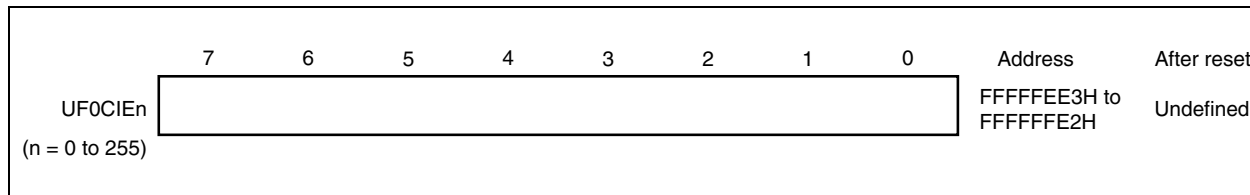


Table 11-6. Data of UF0CIEn Register**(a) Configuration descriptor (9 bytes)**

Offset	Field Name	Contents
0	bLength	Size of this descriptor
1	bDescriptorType	Descriptor type
2	wTotalLength	Lower value of the total number of bytes of Configuration, all Interface, and all Endpoint descriptors
3		Higher value of the total number of bytes of Configuration, all Interface, and all Endpoint descriptors
4	bNumInterface	Number of Interfaces
5	bConfigurationValue	Value to select this Configuration
6	iConfiguration	Index of string descriptor describing this Configuration
7	bmAttributes	Features of this Configuration (self-powered, without remote wakeup)
8	MaxPower	Maximum power consumption of this Configuration (unit: mA)

(b) Interface descriptor (9 bytes)

Offset	Field Name	Contents
0	bLength	Size of this descriptor
1	bDescriptorType	Descriptor type
2	bInterfaceNumber	Value of this Interface
3	bAlternateSetting	Value to select alternative setting of Interface
4	bNumEndpoints	Number of usable Endpoints
5	bInterfaceClass	Class code
6	bInterfaceSubClass	Subclass code
7	bInterfaceProtocol	Protocol code
8	Interface	Index of string descriptor describing this Interface

(c) Endpoint descriptor (7 bytes)

Offset	Field Name	Contents
0	bLength	Size of this descriptor
1	bDescriptorType	Descriptor type
2	bEndpointAddress	Address/transfer direction of this Endpoint
3	bmAttributes	Transfer type
4	wMaxPaketSize	Lower value of maximum number of transfer data
5		Higher value of maximum number of transfer data
6	bInterval	Transfer interval

11.4.4 Peripheral control registers

(1) USB function 0 DMA channel select register (UF0CS)

This register allocates each DMA service of the USB function to a DMA channel.

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

To allocate the service of a USB function to a DMA channel by using this register, set this register in advance, set the DTFRn register (n = 0 to 3) of the DMA controller to 7FH, and enable USB_DMA in advance.

★ **Caution** Setting the same DMA service to different DMA channels and setting different DMA services to the same DMA channel are prohibited.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
UF0CS	0	UFD	UFD	UFD	0	UFD	UFD	UFD	0	UFD	UFD	UFD	0	UFD	UFD	UFD	FFFFFFDF0H,	0000H
		C32	C31	C30		C22	C21	C20		C12	C11	C10		C02	C01	C00	FFFFFFDF1H	

Bit position	Bit name	Function																								
14 to 12, 10 to 8, 6 to 4, 2 to 0	UFDCn2 to UFDCn0	These bits set the DMA service of a USB function to be allocated to DMA channel n. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 10%;">UFDCn2</th> <th style="width: 10%;">UFDCn1</th> <th style="width: 10%;">UFDCn0</th> <th style="width: 70%;">Service to be allocated</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>EP4_DMA</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>EP3_DMA</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>EP2_DMA</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>EP1_DMA</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">X</td> <td style="text-align: center;">X</td> <td>No allocation (DMA not used)</td> </tr> </tbody> </table>	UFDCn2	UFDCn1	UFDCn0	Service to be allocated	1	1	1	EP4_DMA	1	1	0	EP3_DMA	1	0	1	EP2_DMA	1	0	0	EP1_DMA	0	X	X	No allocation (DMA not used)
UFDCn2	UFDCn1	UFDCn0	Service to be allocated																							
1	1	1	EP4_DMA																							
1	1	0	EP3_DMA																							
1	0	1	EP2_DMA																							
1	0	0	EP1_DMA																							
0	X	X	No allocation (DMA not used)																							
Remark X: Don't care																										

Remark n = 0 to 3

(2) USB function 0 buffer control register (UF0BC)

This register performs enable control and floating control on the input buffer of the USB function.

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
UF0BC	0	0	0	0	0	0	UBFIEN	UBFIOR	FFFFFFEDF2H	00H

Bit position	Bit name	Function
1	UBFIEN	This bit controls use of the USB buffer. 1: Buffer valid 0: Buffer invalid Caution Clear this bit to 0 when the USB is not used. If this bit is set to 1, a current of 3 mA (TYP.) constantly flows, regardless of whether the USB is used or not.
0	UBFIOR	This bit controls use of floating measures of the USB buffer. 1: Disables floating measures 0: Enables floating measures This bit prevents erroneous recognition of Bus Reset, Suspend, and Resume due to an undefined value when a cable is not connected (when data input is floated). When this bit is set to 1, control the processing for floating by the VBUS signal (which recognizes cable connection).

11.5 STALL Handshake or No Handshake

Errors of USBF are defined to be handled as follows.

Transfer Type	Transaction	Target Packet	Error Type	Function Response	Processing
Control transfer/ bulk transfer/ interrupt transfer	IN/OUT/SETUP	Token	Endpoint not supported	No response	None
			Endpoint transfer direction mismatch	No response	None
			CRC error	No response	None
			Bit stuffing error	No response	None
Control transfer/ bulk transfer	OUT/SETUP	Data	Timeout	No response	None
			PID check error	No response	None
			Unsupported PID (other than Data PID)	No response	None
			CRC error	No response	Discard received data
			Bit stuffing error	No response	Discard received data
	OUT	Data	Data PID mismatch	ACK	Discard received data
Control transfer (SETUP stage)	SETUP	Data	Overrun	No response	Discard received data
Control transfer (data stage)	OUT	Data	Overrun	No response ^{Note 1}	Set SNDSTL bit of UF0SDS register to 1 and discard received data
Control transfer (status stage)	OUT	Data	Overrun	ACK or no response ^{Note 2}	Set SNDSTL bit of UF0SDS register to 1 and discard received data
Bulk transfer	OUT	Data	Overrun	No response ^{Note 1}	Set EnHALT bit of UF0EnSL register (n = 0 to 4, 7, 8) to 1
Control transfer/ bulk transfer/ interrupt transfer	IN	Handshake	PID check error	–	Hold transferred data and re-transfer data ^{Note 3}
			Unsupported PID (other than ACK PID)	–	Hold transferred data and re-transfer data ^{Note 3}
			Timeout	–	Hold transferred data and re-transfer data ^{Note 3}

Notes 1. A STALL response is made to re-transfer by the host.

2. An ACK response is made if the transfer data is of less than MaxPacketSize and the data received in the status stage is discarded. If MaxPacketSize is exceeded, no response is made, the SNDSTL bit of the UF0SDS register is set to 1, and the received data is discarded.

3. If an OUT transaction indicating a change from the data stage to the status stage is received during control transfer, an error is not handled and it is assumed that reception has been correctly completed.

Cautions 1. It is judged by the Alternative Setting number currently set whether the target Endpoint is valid or invalid.

2. For the response to the request included in control transfer to/from Endpoint0, see 11.3 Requests.

11.6 Register Values in Specific Status

Table 11-7. Register Values in Specific Status (1/2)

Register Name	After CPU Reset ($\overline{\text{RESET}}$)	After Bus Reset
UF0E0N register	00H	Value is held.
UF0E0NA register	00H	Value is held.
UF0EN register	00H	Value is held.
UF0ENM register	00H	Value is held.
UF0SDS register	00H	Value is held.
UF0CLR register	00H	Value is held.
UF0SET register	00H	Value is held.
UF0EPS0 register	00H	Value is held.
UF0EPS1 register	00H	Value is held.
UF0EPS2 register	00H	Value is held.
UF0IS0 register	00H	Value is held.
UF0IS1 register	00H	Value is held.
UF0IS2 register	00H	Value is held.
UF0IS3 register	00H	Value is held.
UF0IS4 register	00H	Value is held.
UF0IM0 register	00H	Value is held.
UF0IM1 register	00H	Value is held.
UF0IM2 register	00H	Value is held.
UF0IM3 register	00H	Value is held.
UF0IM4 register	00H	Value is held.
UF0IC0 register	FFH	Value is held.
UF0IC1 register	FFH	Value is held.
UF0IC2 register	FFH	Value is held.
UF0IC3 register	FFH	Value is held.
UF0IC4 register	FFH	Value is held.
UF0IDR register	00H	Value is held.
UF0DMS0 register	00H	Value is held.
UF0DMS1 register	00H	Value is held.
UF0FIC0 register	00H	Value is held.
UF0FIC1 register	00H	Value is held.
UF0DEND register	00H	Value is held.
UF0GPR register	00H	Value is held.
UF0MODC register	00H	Value is held.
UF0MODS register	00H	Bit 2 (CONF): Cleared (0), Other bits: Value is held.
UF0AIFN register	00H	Value is held.
UF0AAS register	00H	Value is held.
UF0ASS register	00H	00H
UF0E1IM register	00H	Value is held.
UF0E2IM register	00H	Value is held.

Table 11-7. Register Values in Specific Status (2/2)

Register Name	After CPU Reset ($\overline{\text{RESET}}$)	After Bus Reset
UF0E3IM register	00H	Value is held.
UF0E4IM register	00H	Value is held.
UF0E7IM register	00H	Value is held.
UF0E8IM register	00H	Value is held.
UF0E0R register	Undefined ^{Note 1}	Value is held.
UF0E0L register	00H	Value is held.
UF0E0ST register	00H	00H
UF0E0W register	Undefined ^{Note 1}	Value is held.
UF0BO1 register	Undefined ^{Note 1}	Value is held.
UF0BO1L register	00H	Value is held.
UF0BO2 register	Undefined ^{Note 1}	Value is held.
UF0BO2L register	00H	Value is held.
UF0BI1 register	Undefined ^{Note 1}	Value is held.
UF0BI2 register	Undefined ^{Note 1}	Value is held.
UF0INT1 register	Undefined	Value is held.
UF0INT2 register	Undefined	Value is held.
UF0DSTL register	00H	00H
UF0E0SL register	00H	00H
UF0E1SL register	00H	00H
UF0E2SL register	00H	00H
UF0E3SL register	00H	00H
UF0E4SL register	00H	00H
UF0E7SL register	00H	00H
UF0E8SL register	00H	00H
UF0ADRS register	00H	00H
UF0CNF register	00H	00H
UF0IF0 register	00H	00H
UF0IF1 register	00H	00H
UF0IF2 register	00H	00H
UF0IF3 register	00H	00H
UF0IF4 register	00H	00H
UF0DSCL register	00H	Value is held.
UF0DDn register (n = 0 to 17)	Note 2	Note 2
UF0CIEn register (n = 0 to 255)	Note 2	Note 2

- Notes 1.** This register can be cleared to 0 by the $\overline{\text{RESET}}$ signal because its write pointer, counter, and read pointer are cleared to 0 when the $\overline{\text{RESET}}$ signal becomes active, in the same manner as clearing by the UF0FICn register, as the register is controlled by FIFO.
- 2.** This register cannot be cleared to 0. Because data can be written to it by FW, however, any value can be written to the register (before doing so, however, be sure to set the EPONKA bit of the UF0E0NA register to 1).

11.7 FW Processing

The following FW processing is performed.

- Setting processing on device side for the SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, and CLEAR_FEATURE requests during enumeration processing
- Analysis and processing of XXXXStandard, XXXXClass, and XXXXVendor requests not subject to automatic processing
- Reading data following bulk-transferred OUT token from receive buffer
- Writing data to be returned in response to bulk-transferred IN token
- Writing data to be returned in response to interrupt-transferred token

The following table lists the requests supported by FW.

★

Table 11-8. FW-Supported Standard Requests

Request	Reception Side	Processing/Frequency	Explanation
CLEAR_FEATURE	Interface	Automatic STALL response	It is considered that this request does not come to Interface because there is no function selector value, though it is reserved for bmRequestType. When this request is received, the hardware makes an automatic STALL response.
SET_FEATURE	Interface	Automatic STALL response	It is considered that this request does not come to Interface because there is no function selector value, though it is reserved for bmRequestType. When this request is received, the hardware makes an automatic STALL response.
GET_DESCRIPTOR	String	FW	Returns the string descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and writes the data to be returned to the host, to the UF0E0W register.
SET_DESCRIPTOR	Device	FW	Rewrites the device descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and the writes the data for the next control transfer (OUT) to the UF0DDn register (n = 0 to 17).
SET_DESCRIPTOR	Configuration	FW	Rewrites the configuration descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and the writes the data for the next control transfer (OUT) to the UF0CIEn register (n = 0 to 255).
SET_DESCRIPTOR	String	FW	Rewrites the string descriptor. When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and loads the data for the next control transfer (OUT).
Other	NA	FW	When this request is received by the SETUP token, the hardware generates the CPUDEC interrupt request for FW. FW decodes the contents of the request from the CPUDEC interrupt request, and performs the necessary processing.

11.7.1 Initialization processing

Initialization processing is executed in the following two ways.

- Initialization of request data register area
- Setting of interrupt

When the request data register area is initialized, data for the GET_XXXX request to which a value is to be automatically returned is written and an endpoint is allocated to an interface. In the interrupt settings, the interrupt sources that do not have to be checked can be masked by using the UF0IMn register (n = 0 to 4).

The following flowcharts illustrate the above processing.

Figure 11-10. Initializing Request Data Register Area

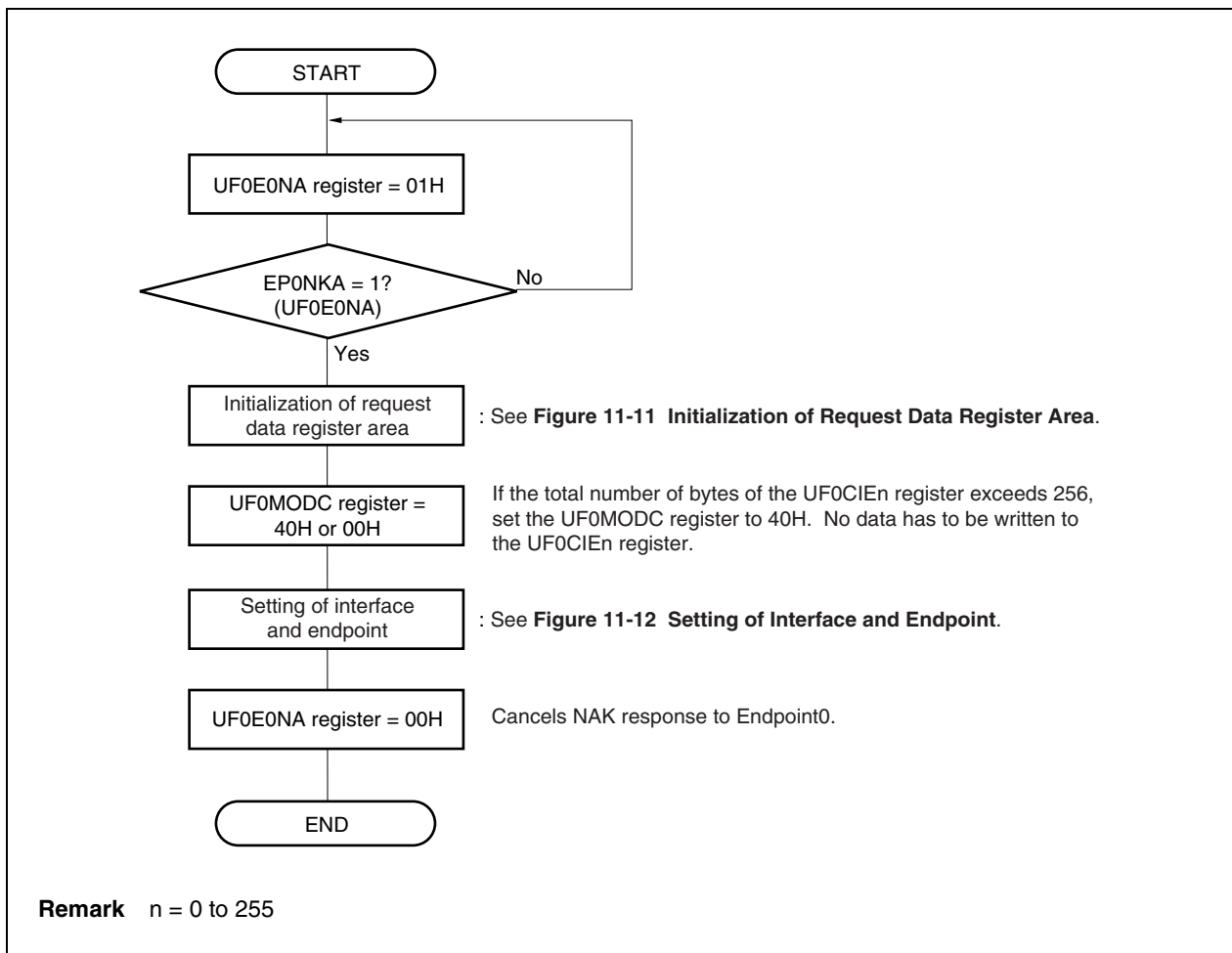


Figure 11-11. Initialization of Request Data Register Area

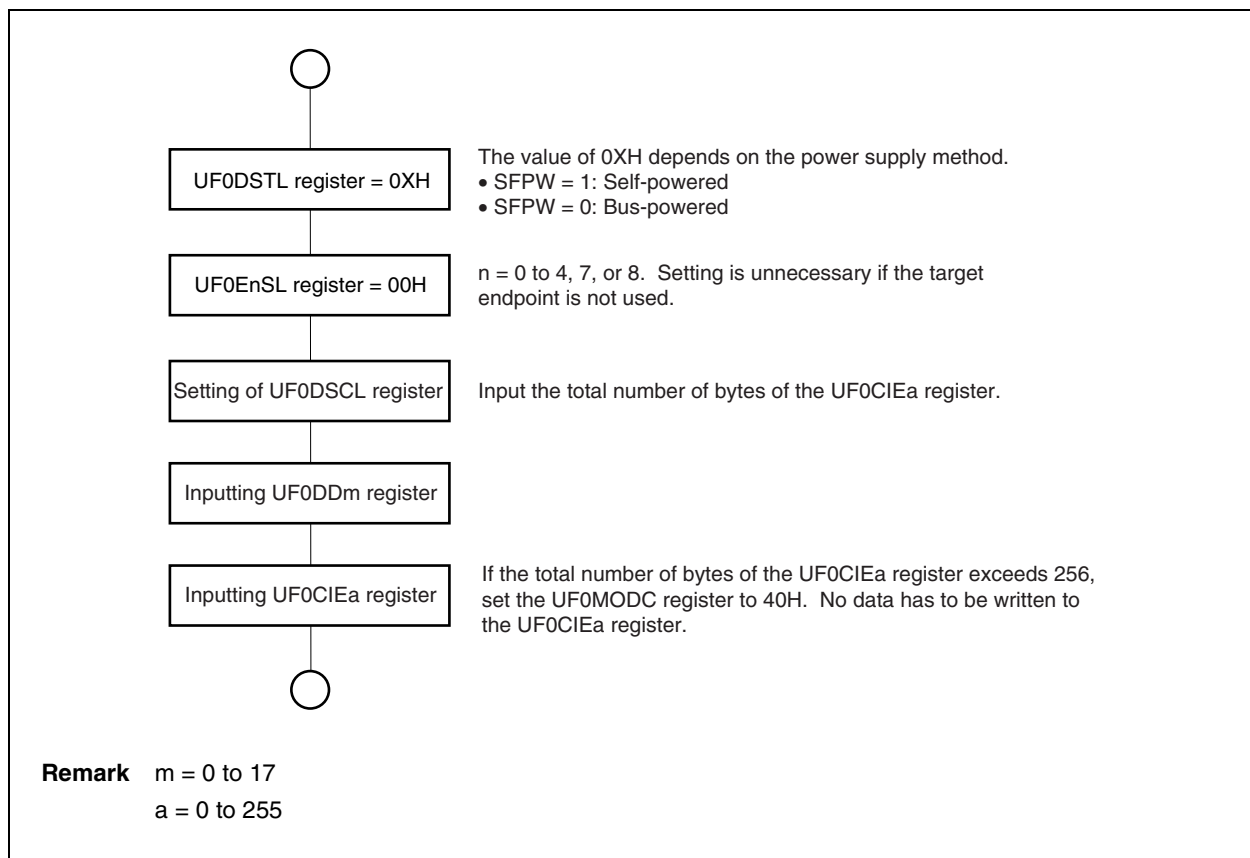


Figure 11-12. Setting of Interface and Endpoint

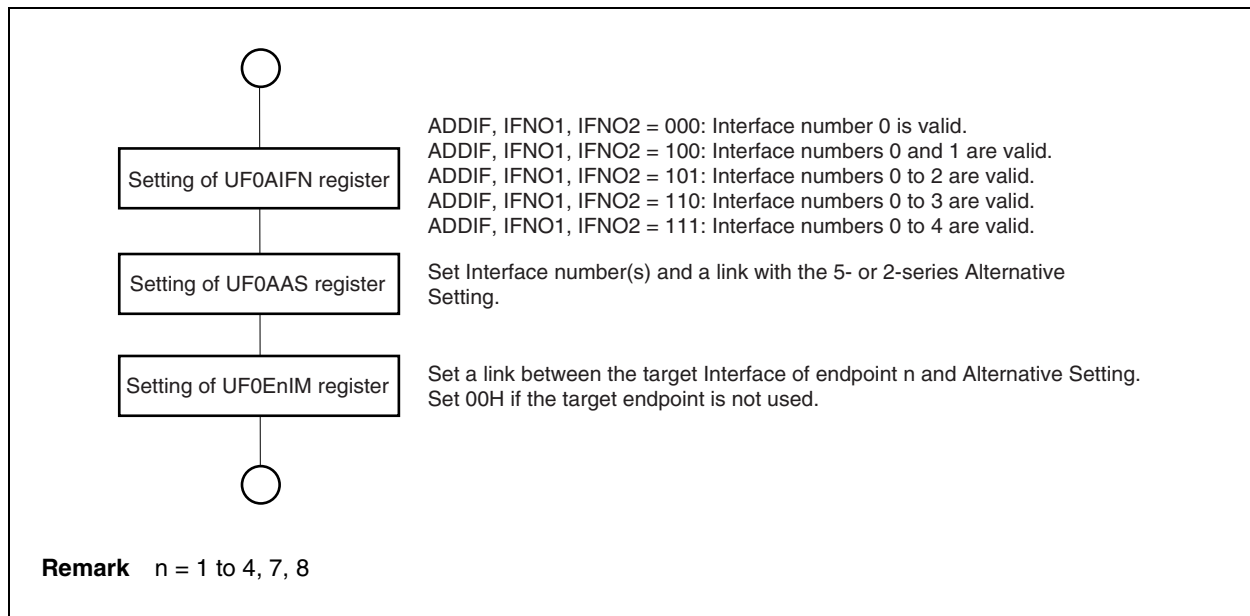
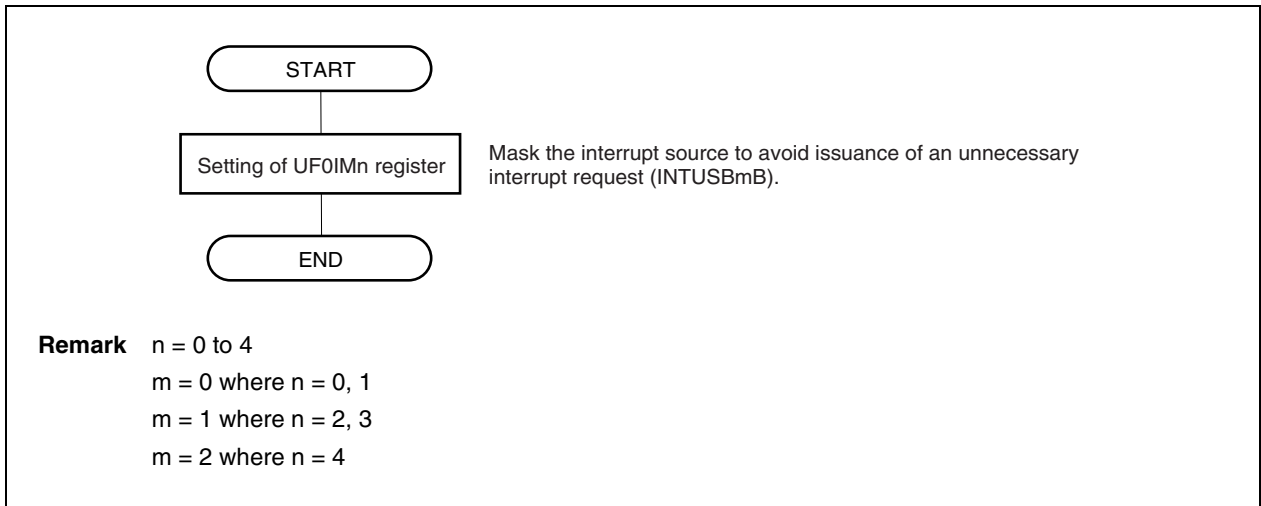


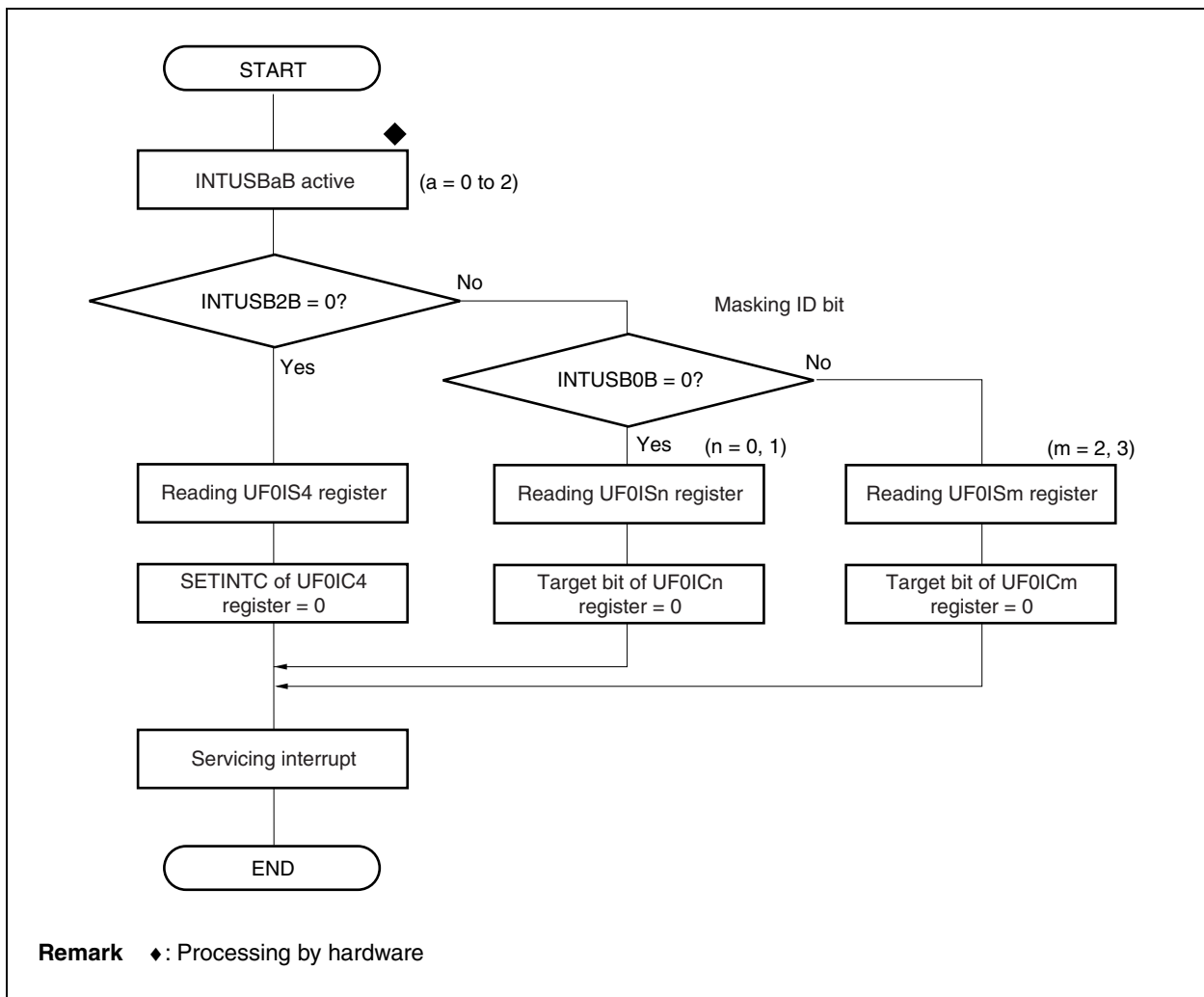
Figure 11-13. Setting of Interrupt



11.7.2 Interrupt servicing

The following flowchart illustrates how an interrupt is serviced.

Figure 11-14. Interrupt Servicing



The following bits of the UF0ISn register are automatically cleared by hardware when a given condition is satisfied (n = 1 to 4).

- E0INDT, E0ODT, SUCES, STG, and CPUDEC bits of UF0IS1 register
- BKI2DT, BKI1DT, IT2DT, and IT1DT bits of UF0IS2 register
- BKO2FL, BKO2DT, BKO1FL, and BKO1DT bits of UF0IS3 register

Because clearing an interrupt source by the UF0ICn register is given a lower priority than setting an interrupt source by hardware, the interrupt source may not be cleared depending on the timing (n = 0 to 4).

11.7.3 USB main processing

USB main processing involves processing USB transactions. The types of transactions to be processed are as follows.

- Fully automatically processed request for control transfer
- Automatically processed requests for control transfer
(SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, CLEAR_FEATURE)
- CPUDEC request for control transfer
- Processing for bulk transfer (IN)
- Processing for bulk transfer (OUT)
- Processing for interrupt transfer (IN)

Processing for endpoint n involves writing or reading for data transfer. The flowchart shown below is for PIO.

(1) Fully automatically processed request for control transfer

Because the fully automatically processed request for control transfer is executed by hardware, it cannot be referenced by FW. Therefore, FW does not have to perform any special processing for this request.

(2) Automatically processed requests for control transfer

(SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, CLEAR_FEATURE)

Processing to write a register for automatically processed requests for control transfer, such as SET_CONFIGURATION, SET_INTERFACE, SET_FEATURE, and CLEAR_FEATURE requests, is automatically executed by hardware, but an interrupt request is issued for recognition on the device side. This processing may be ignored if there is no special processing to be executed.

The flowcharts are shown below.

Figure 11-15. Automatically Processed Requests for Control Transfer

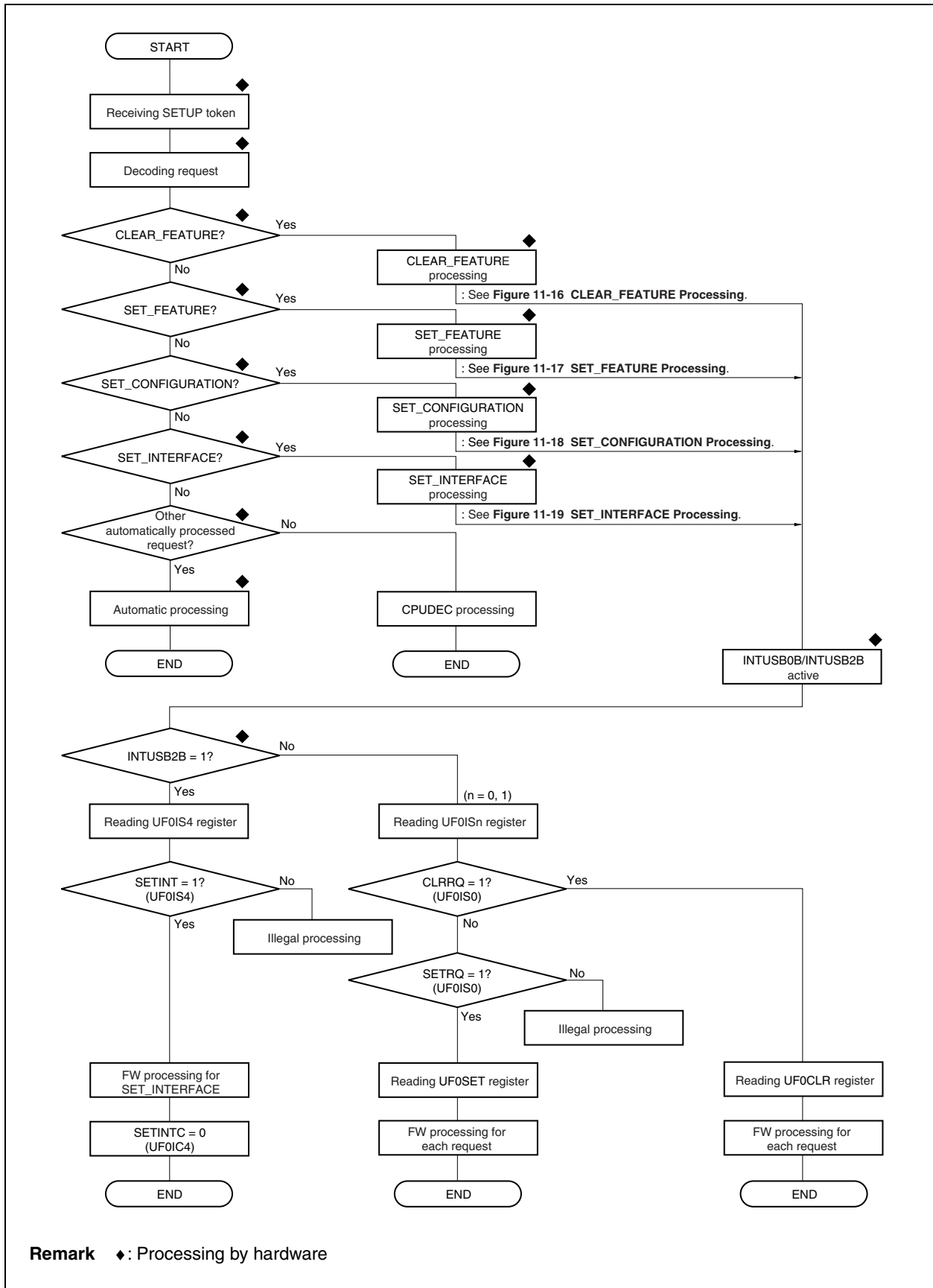


Figure 11-16. CLEAR_FEATURE Processing

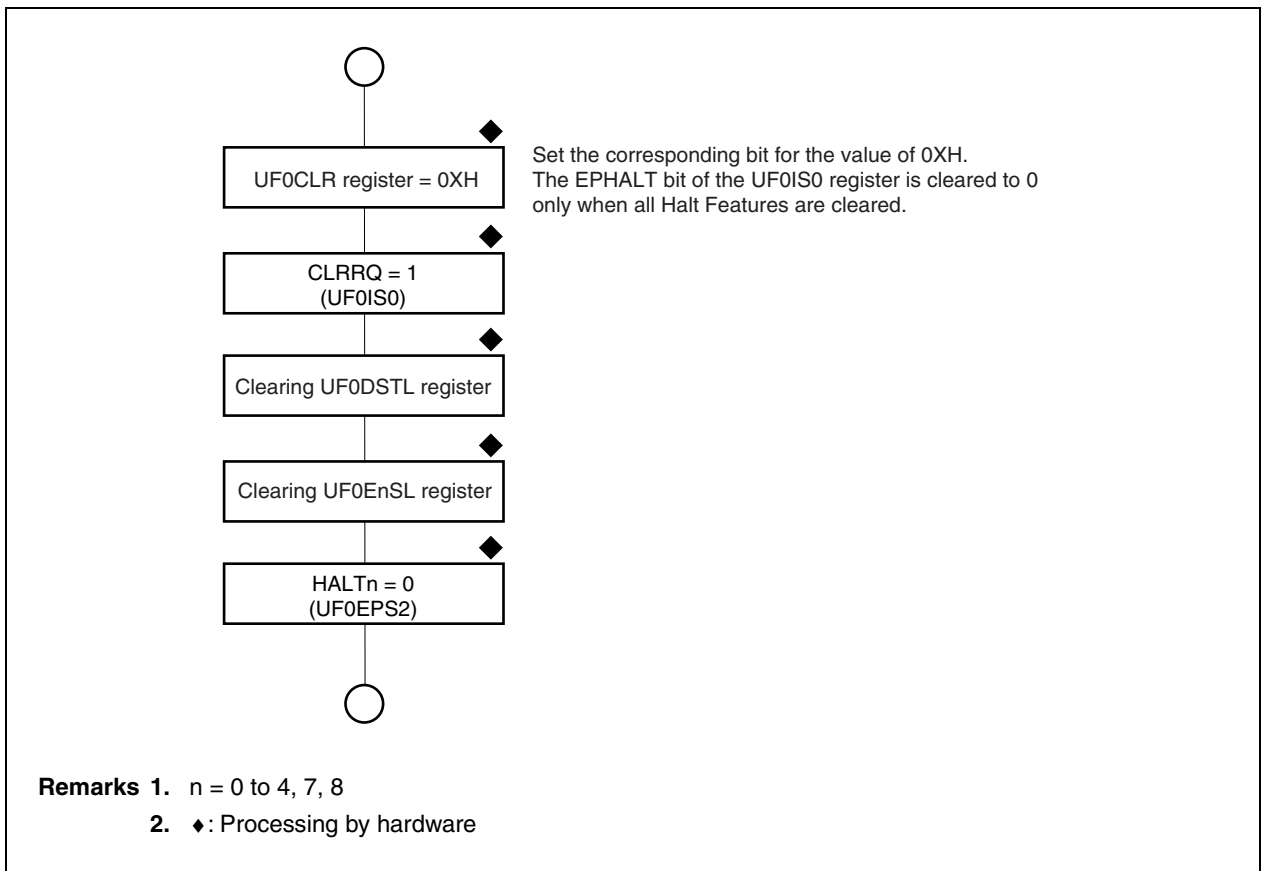


Figure 11-17. SET_FEATURE Processing

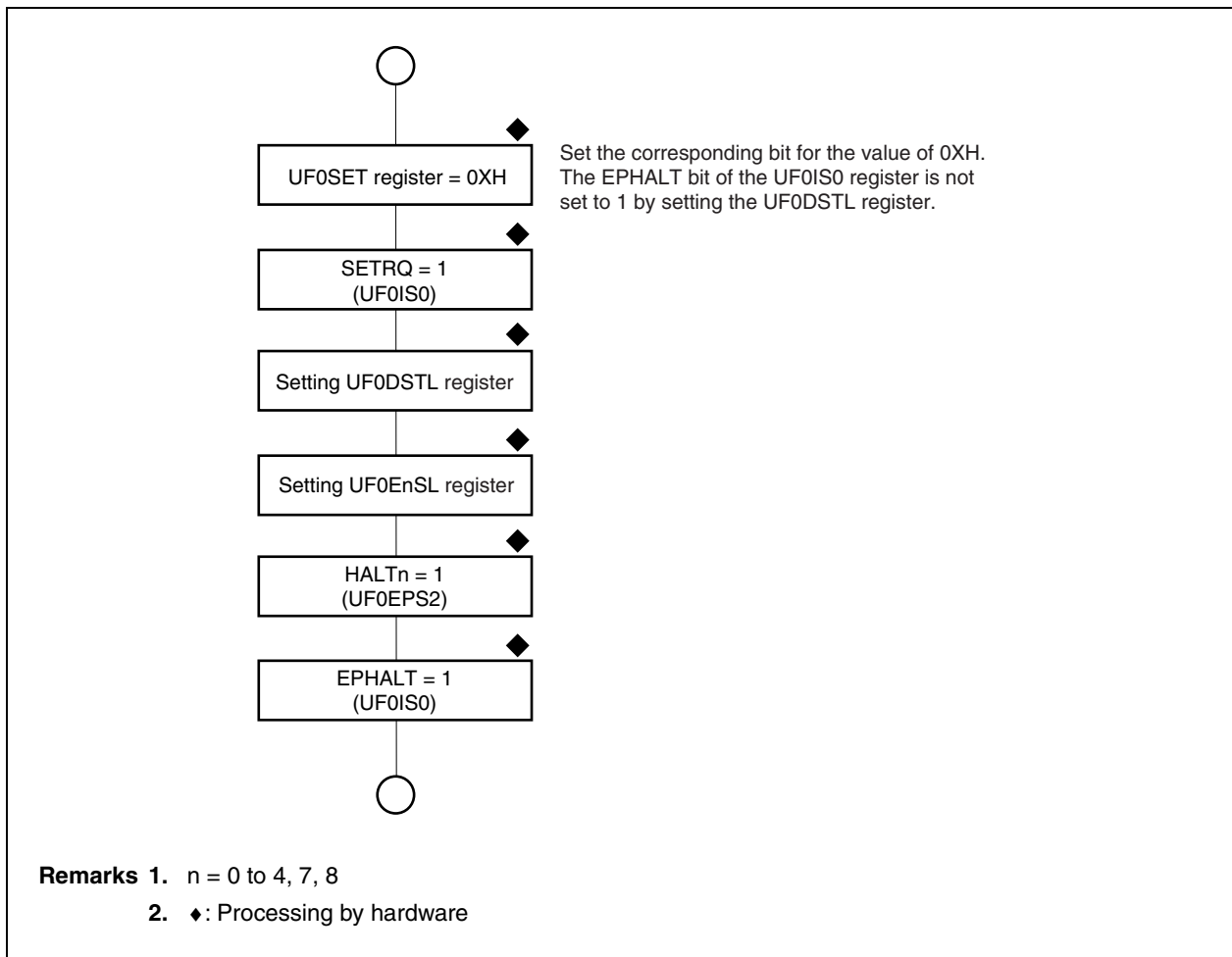


Figure 11-18. SET_CONFIGURATION Processing

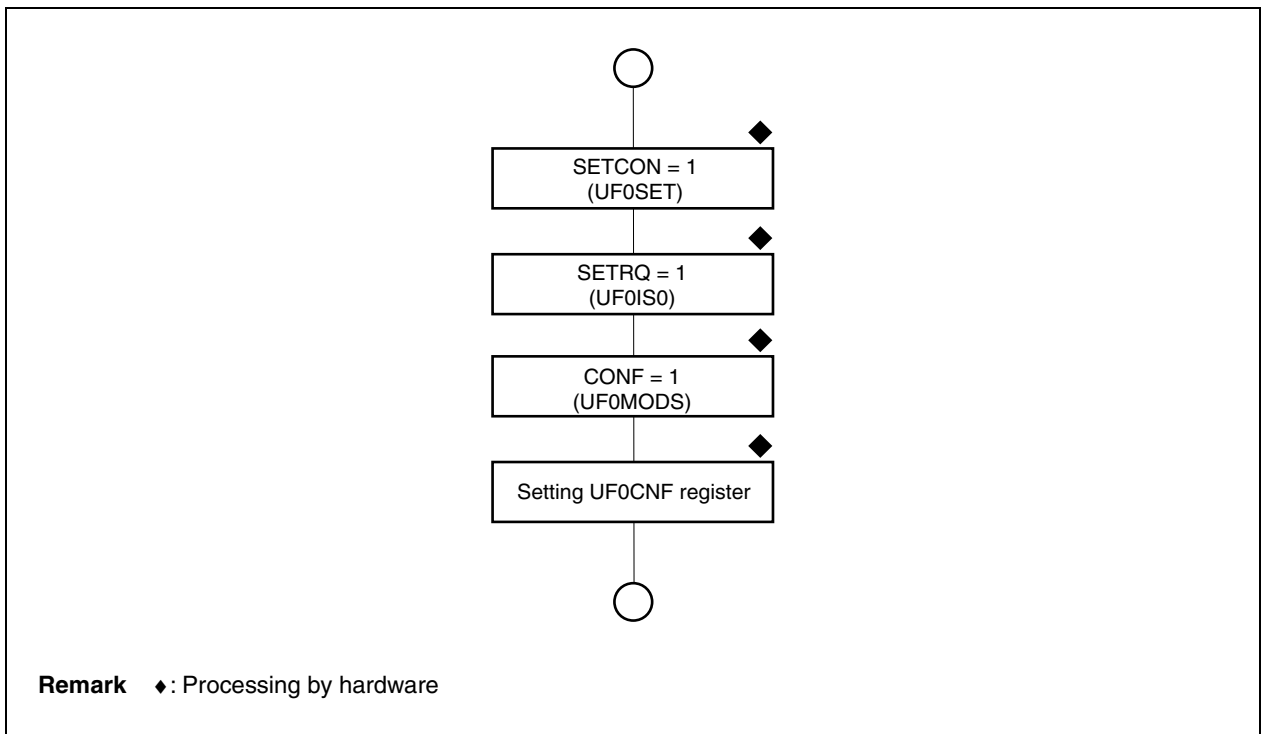
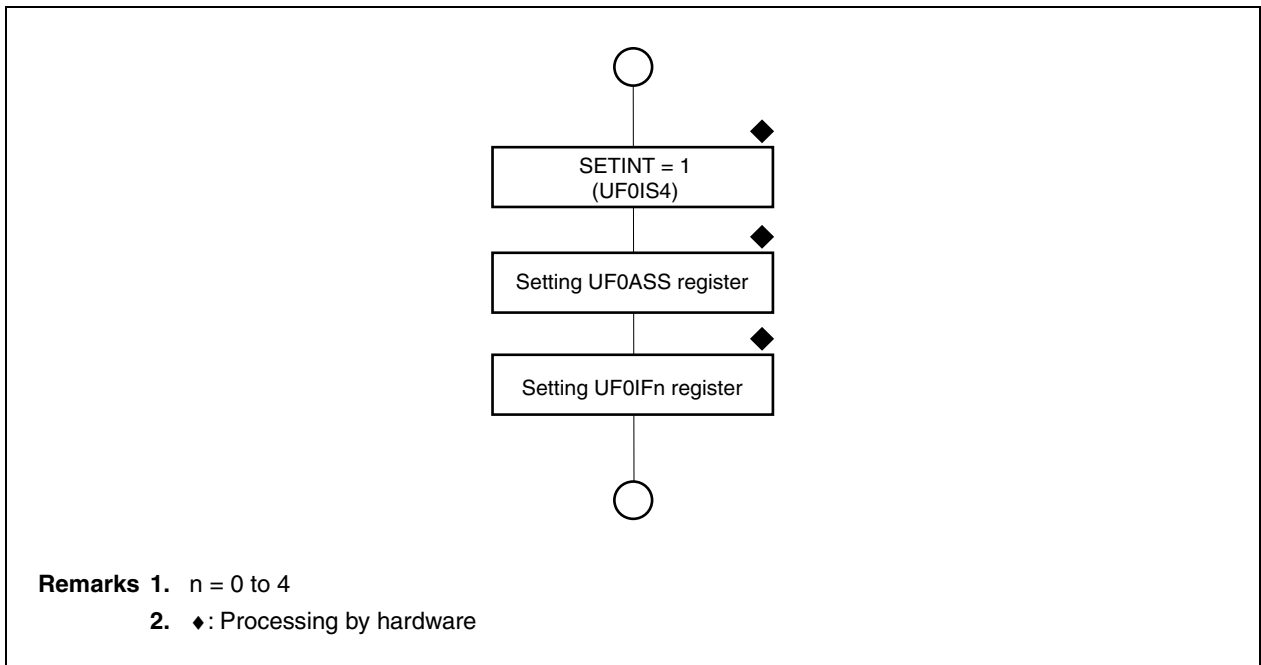


Figure 11-19. SET_INTERFACE Processing



(3) CPUDEC request for control transfer

The CPUDEC request can be classified into three types of processing: control transfer (write), control transfer (read), and control transfer (without data). Control transfer (write) indicates a request that uses the OUT transaction in the data stage (e.g., SET_DESCRIPTOR), and control transfer (read) indicates a request that uses the IN transaction in the data stage (e.g., GET_DESCRIPTOR). Control transfer (without data) indicates a request that has no data stage (e.g., SET_CONFIGURATION).

The flowcharts are shown below.

Figure 11-20. CPUDEC Request for Control Transfer (1/12)

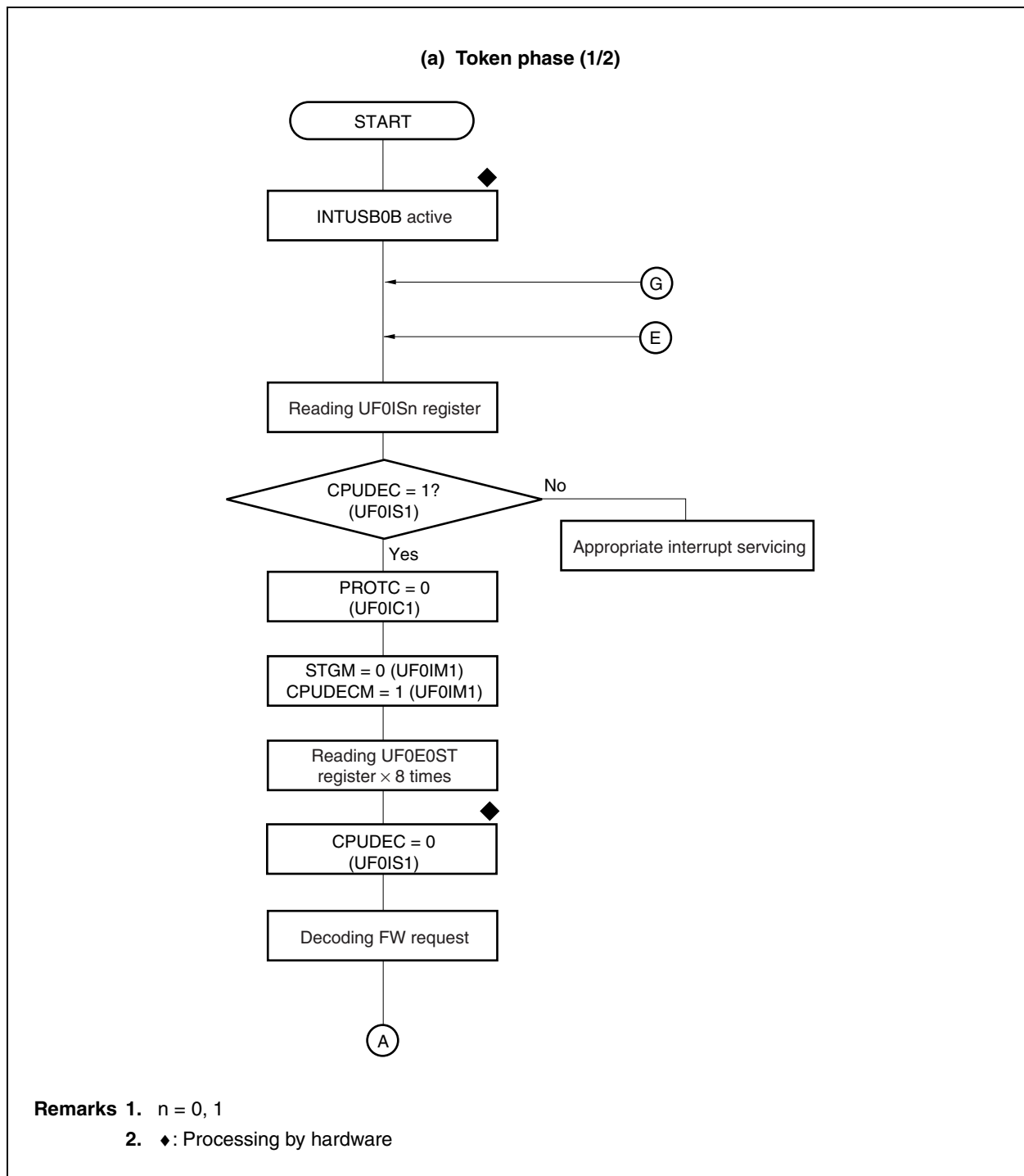


Figure 11-20. CPUDEC Request for Control Transfer (2/12)

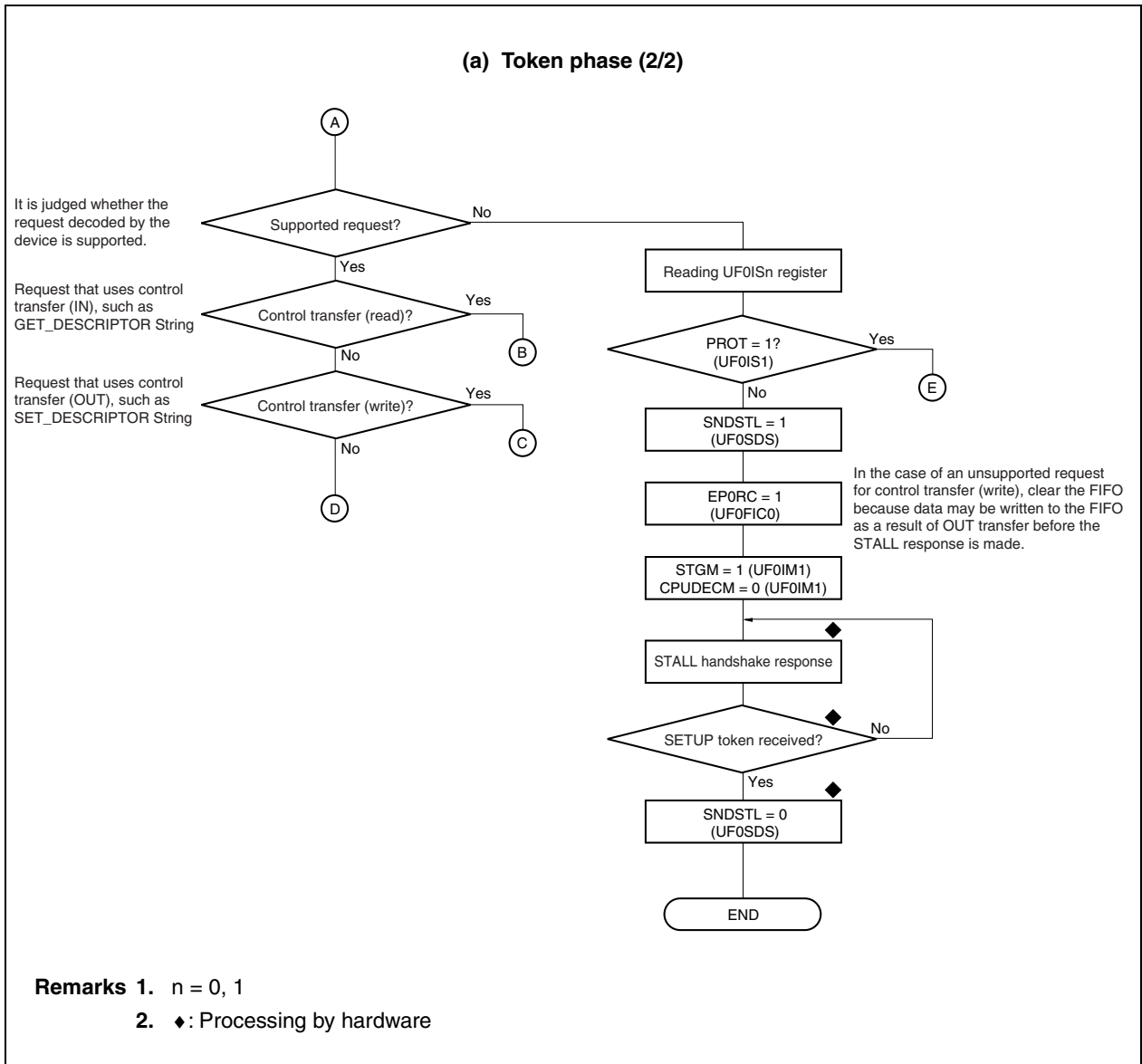


Figure 11-20. CPUDEC Request for Control Transfer (3/12)

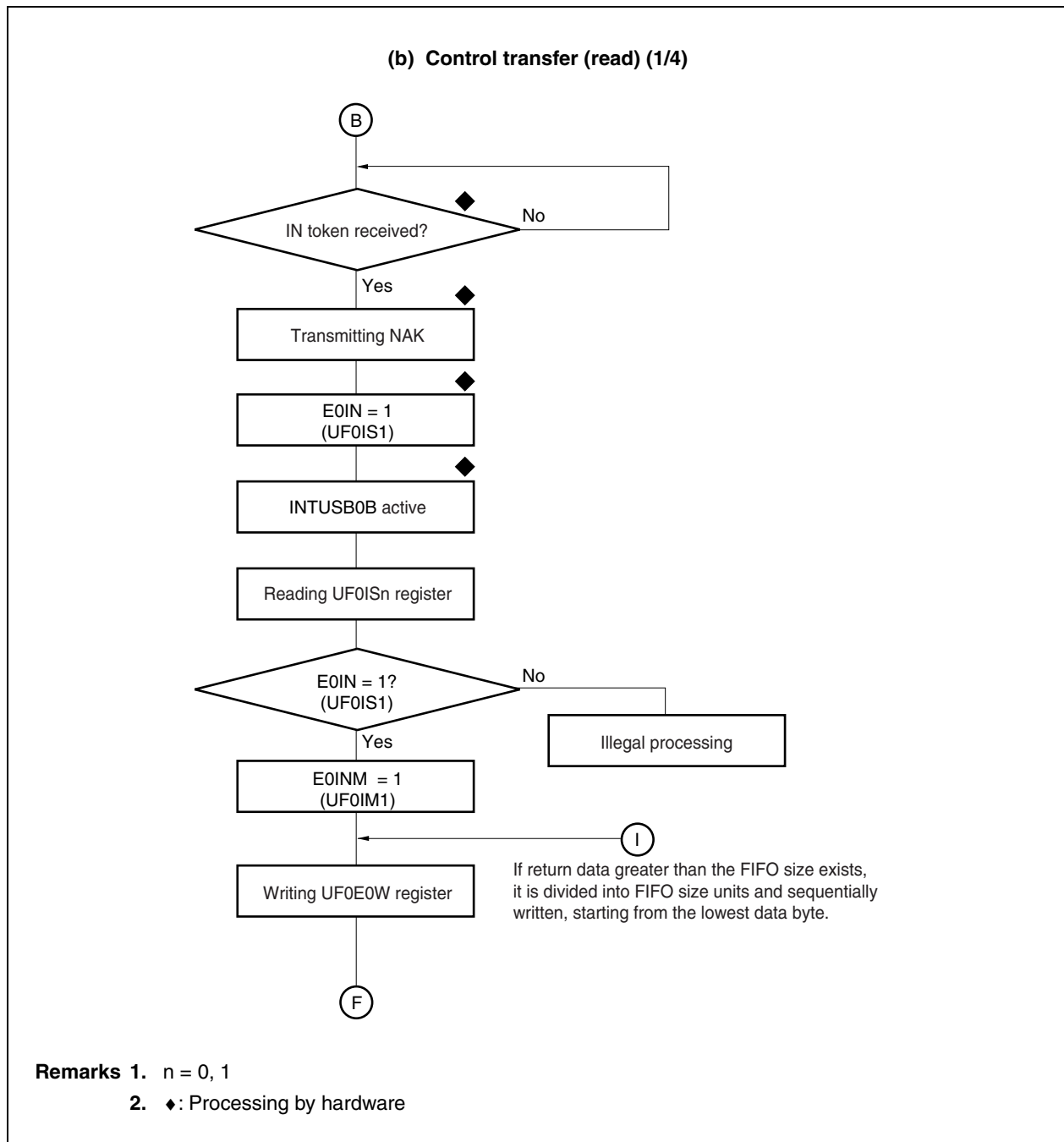


Figure 11-20. CPUDEC Request for Control Transfer (4/12)

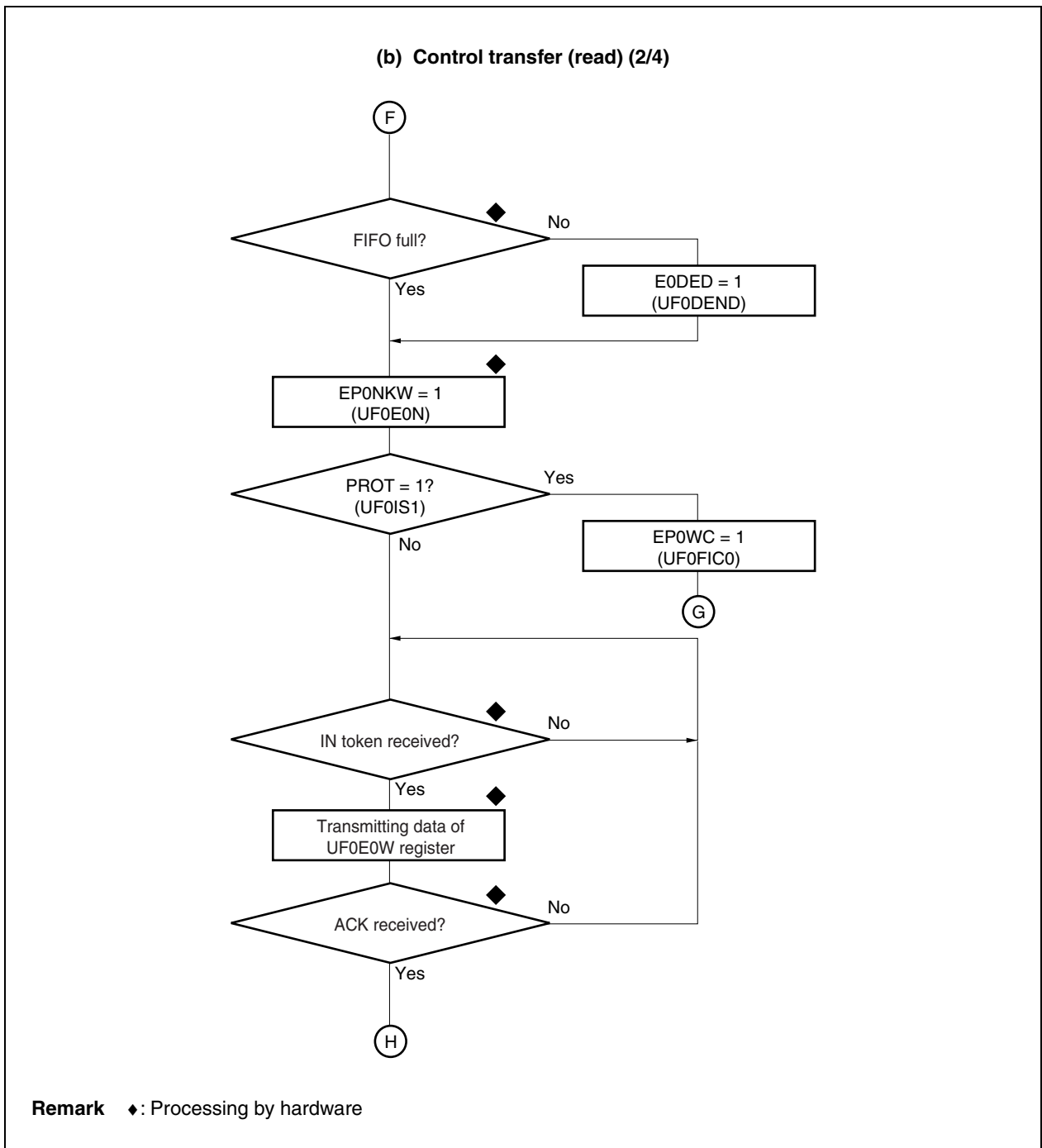


Figure 11-20. CPUDEC Request for Control Transfer (5/12)

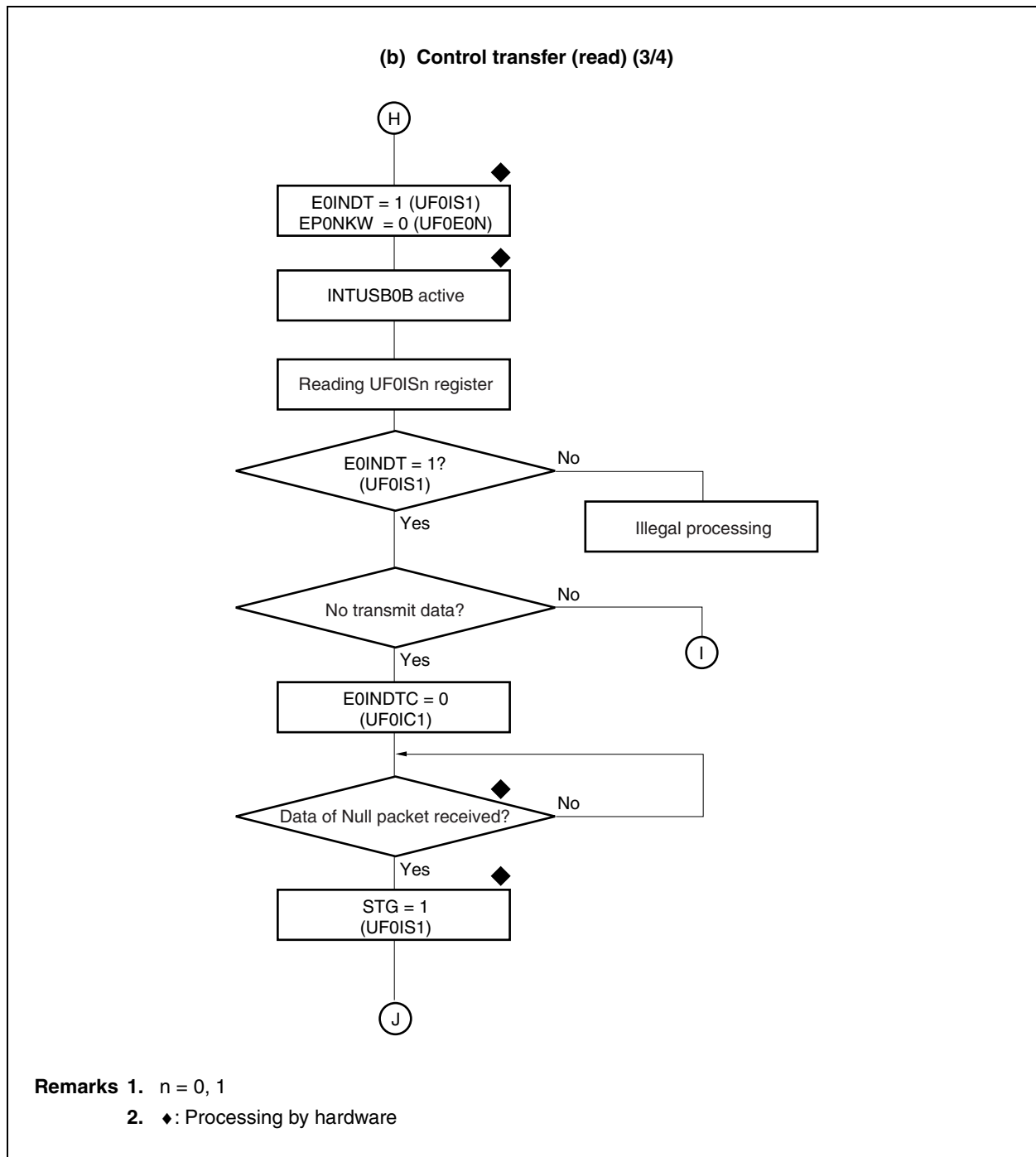


Figure 11-20. CPUDEC Request for Control Transfer (6/12)

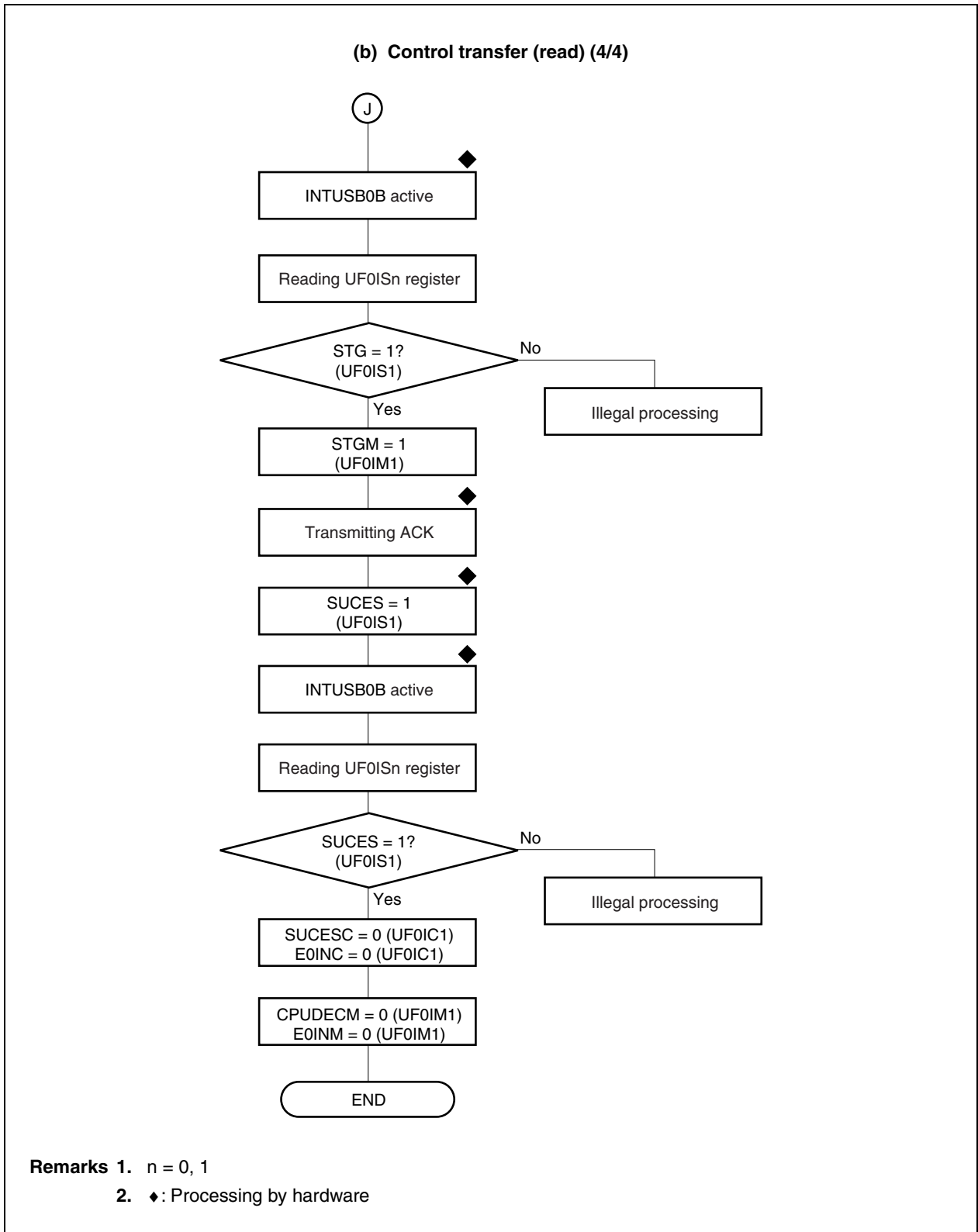


Figure 11-20. CPUDEC Request for Control Transfer (7/12)

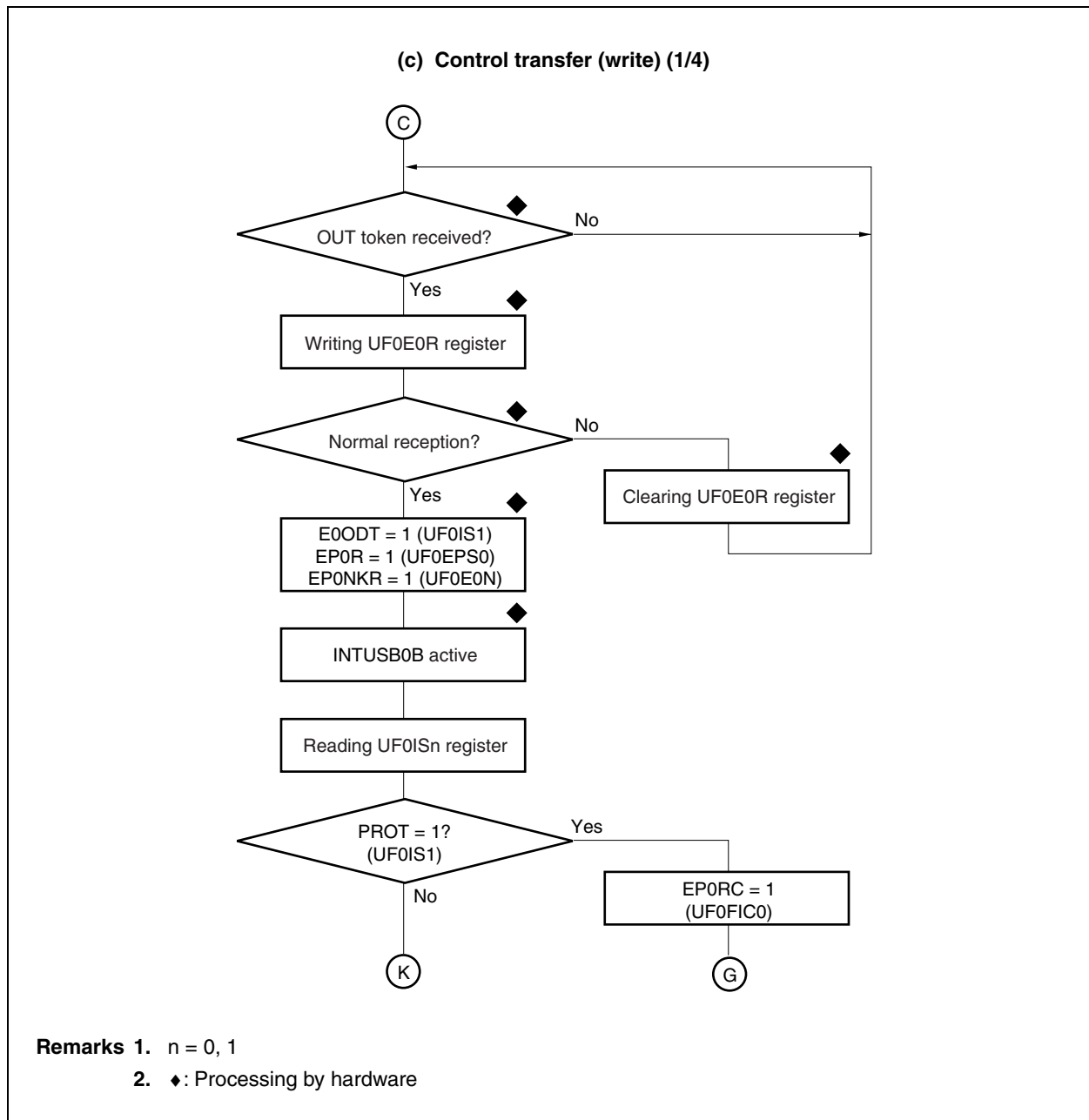


Figure 11-20. CPUDEC Request for Control Transfer (8/12)

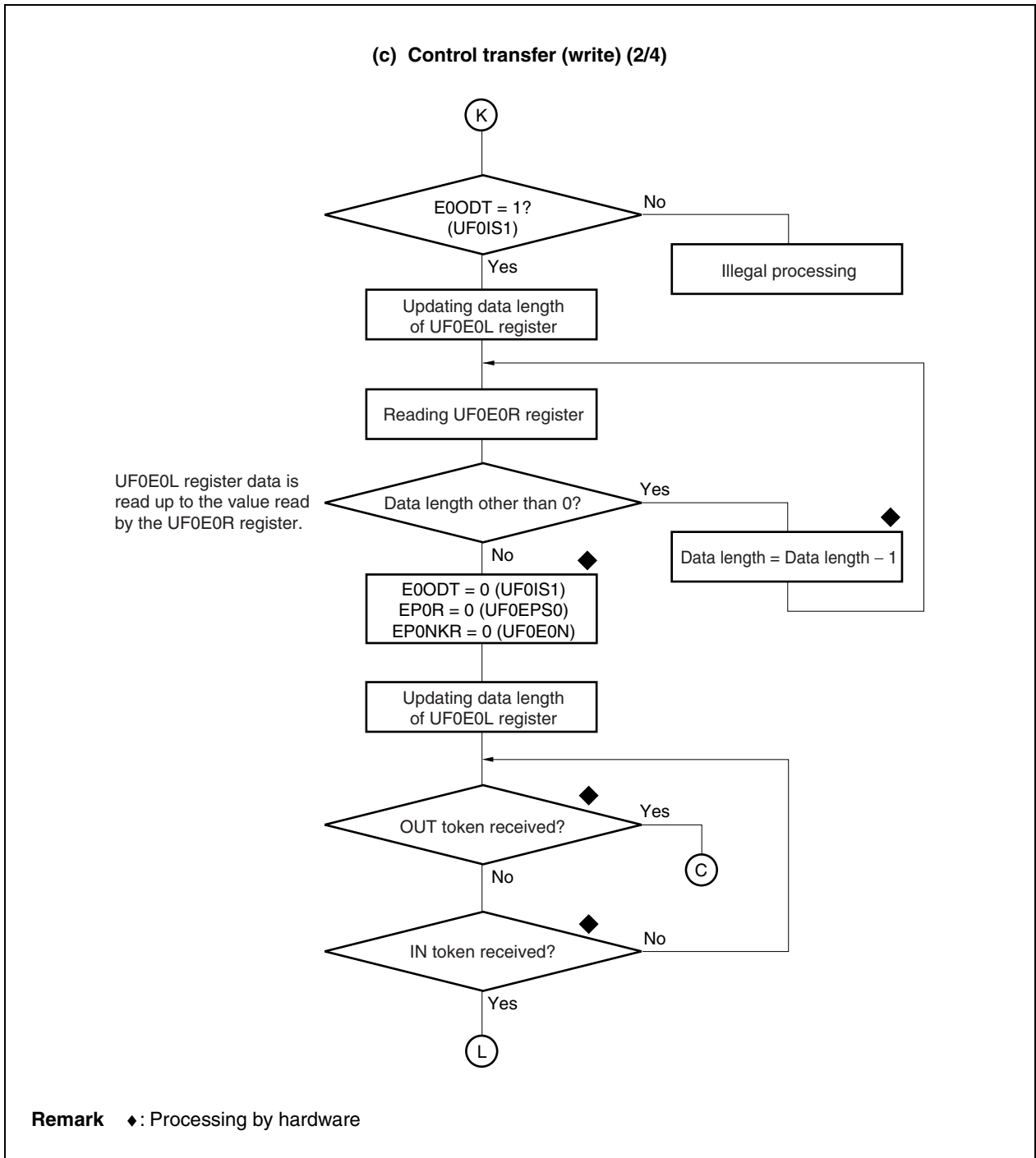


Figure 11-20. CPUDEC Request for Control Transfer (9/12)

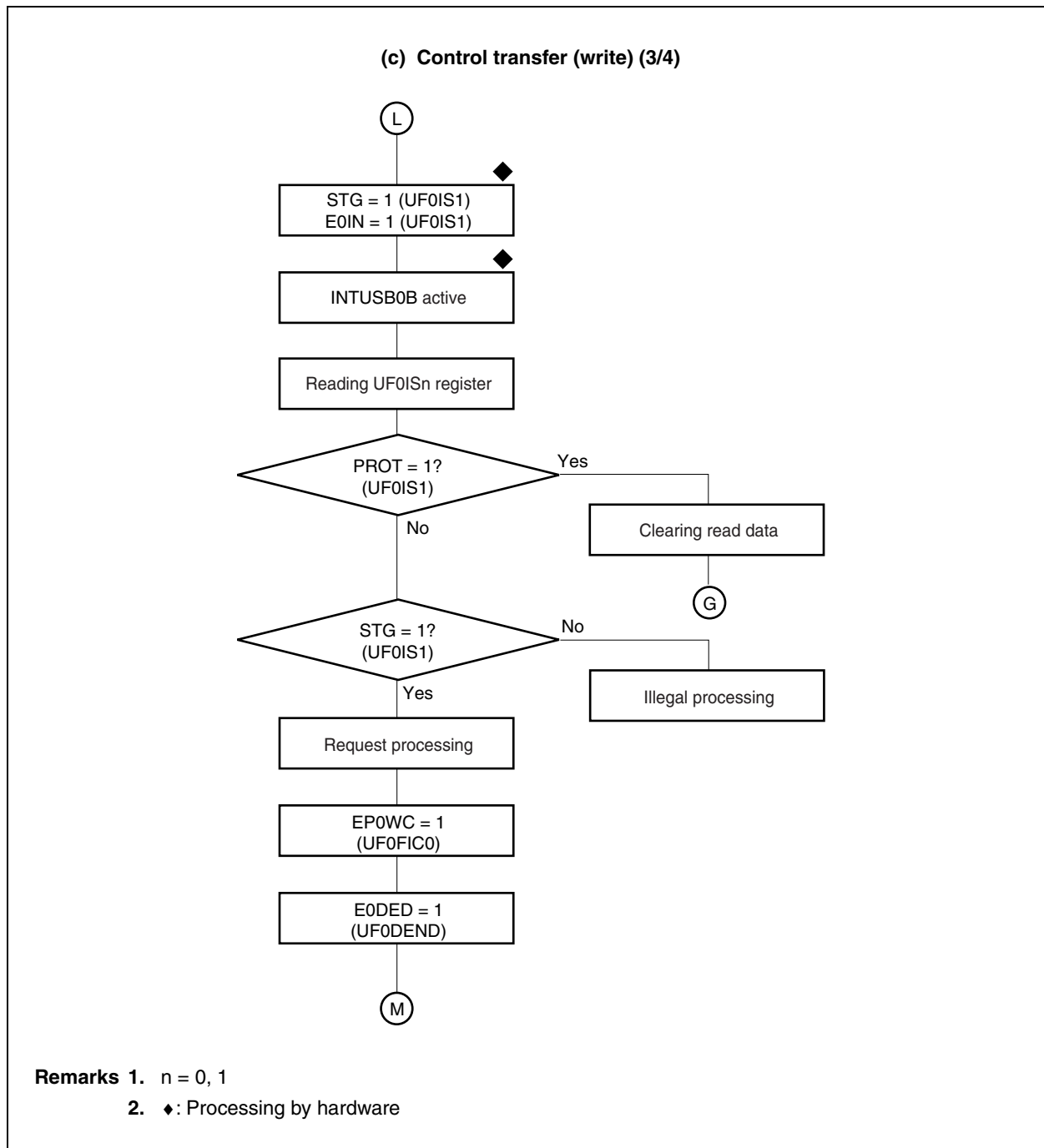


Figure 11-20. CPUDEC Request for Control Transfer (10/12)

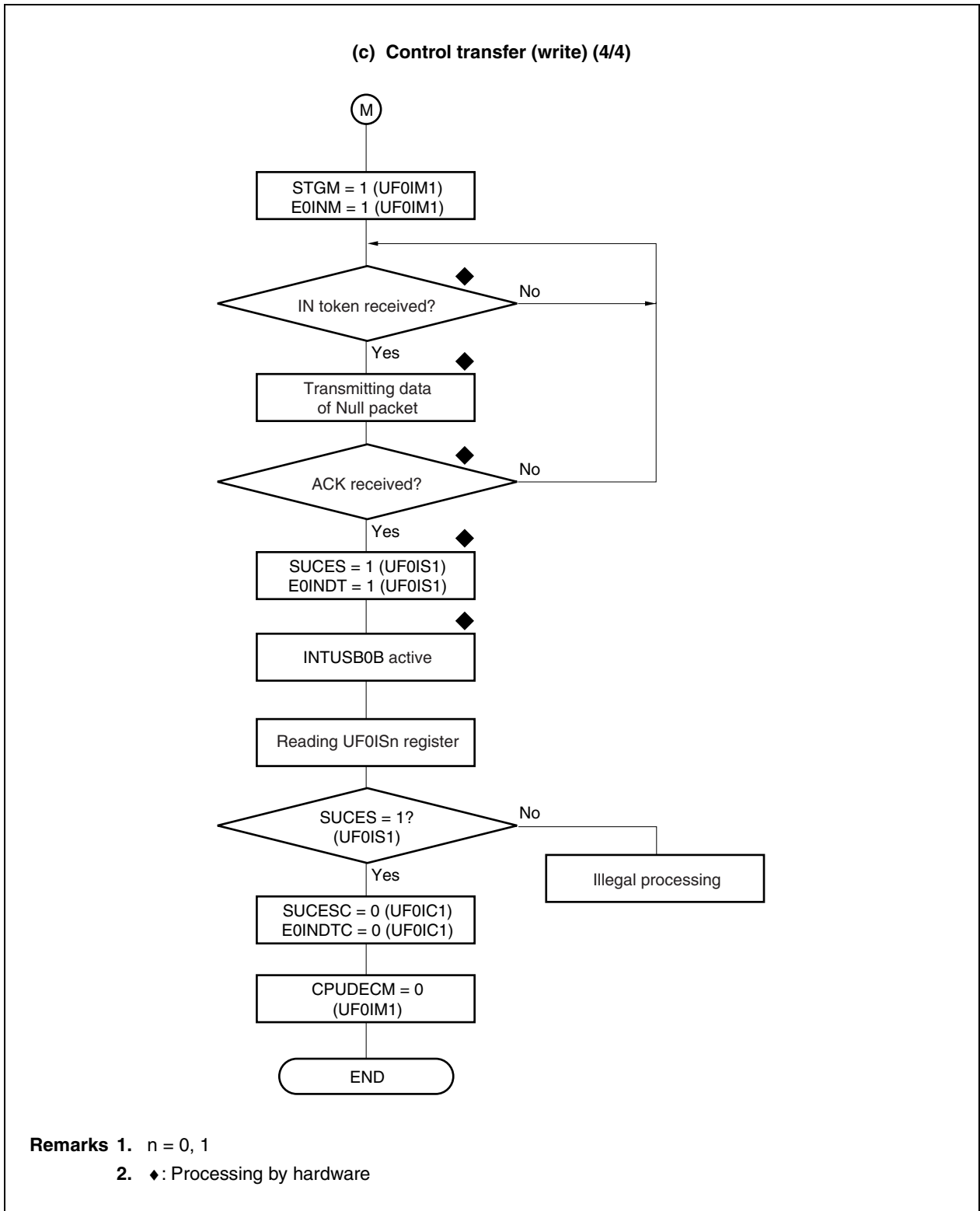


Figure 11-20. CPUDEC Request for Control Transfer (11/12)

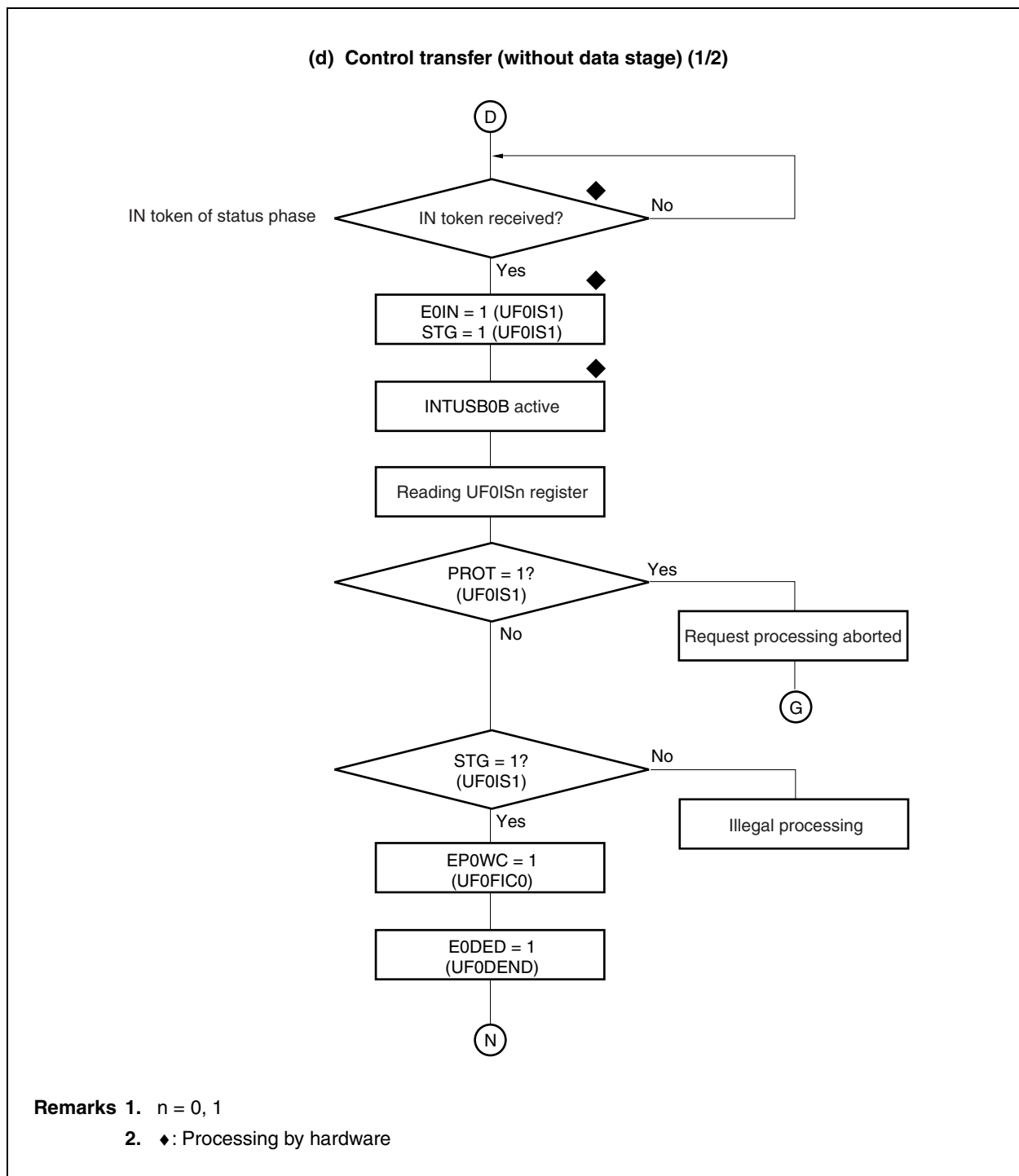
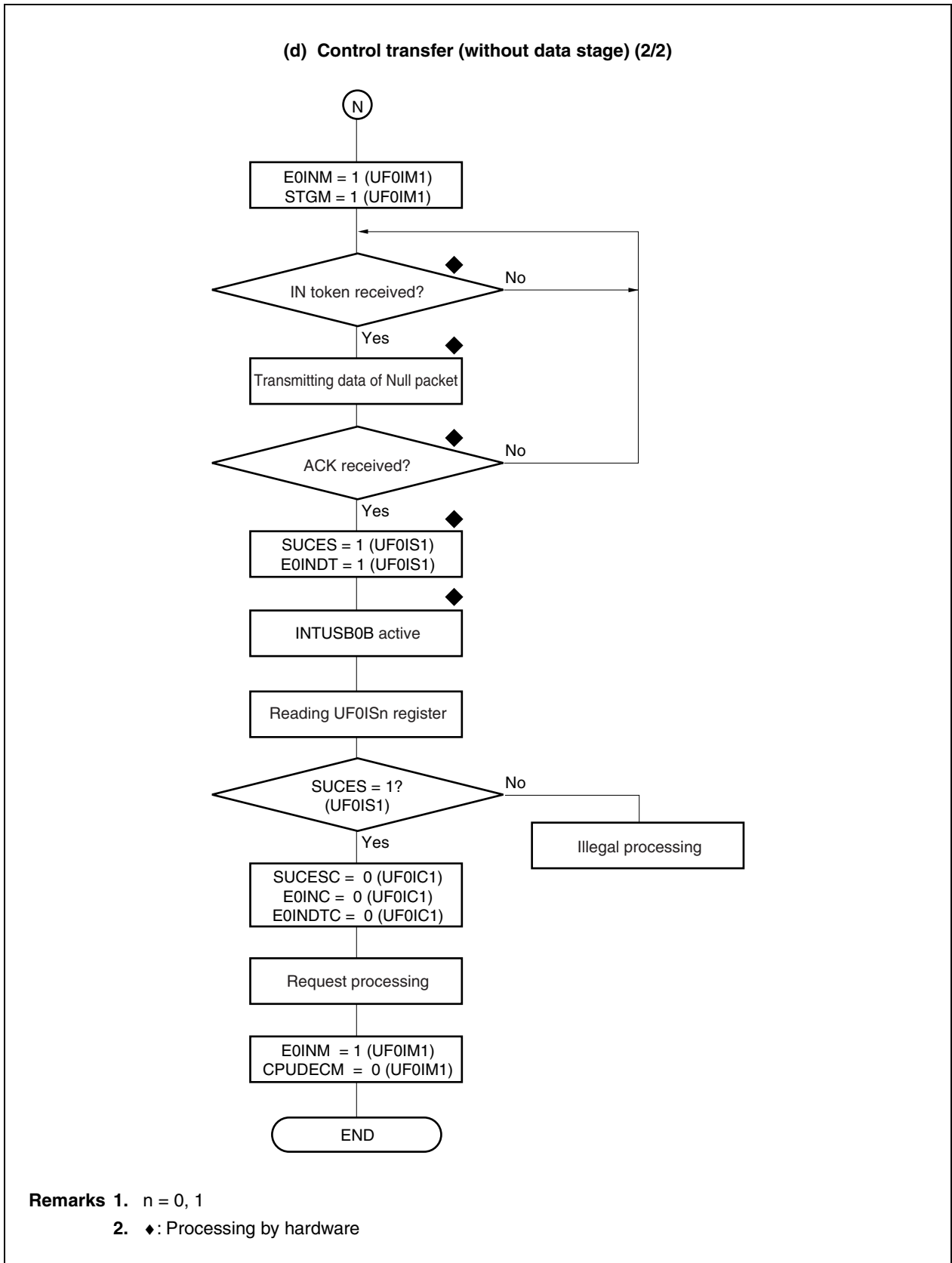


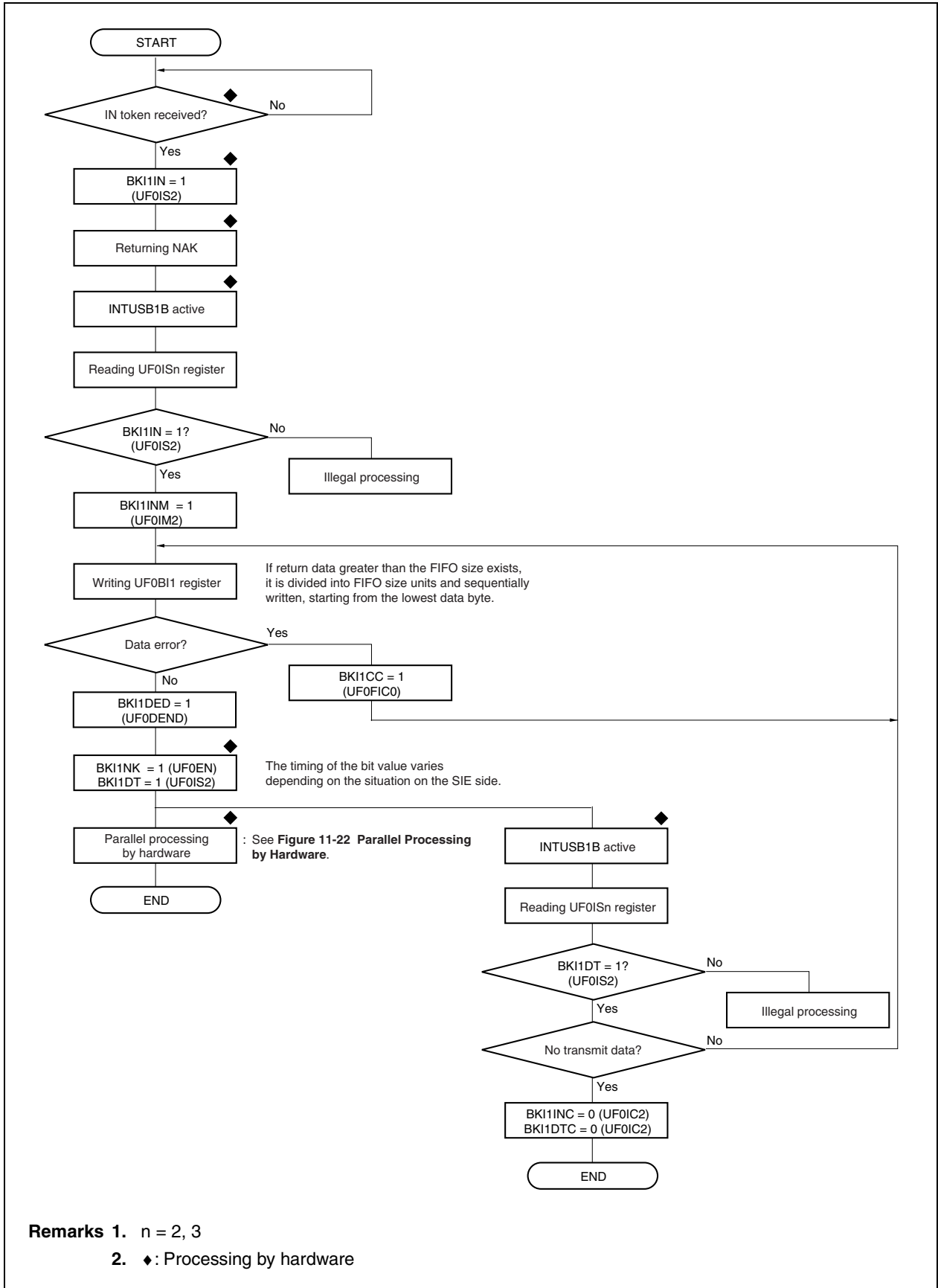
Figure 11-20. CPUDEC Request for Control Transfer (12/12)



(4) Processing for bulk transfer (IN)

Bulk transfer (IN) is allocated to Endpoint1 and Endpoint3. The flowchart shown below illustrates how Endpoint1 is controlled. Endpoint3 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint3, therefore, read the bit names of Endpoint1 in the flowchart as those of Endpoint3.

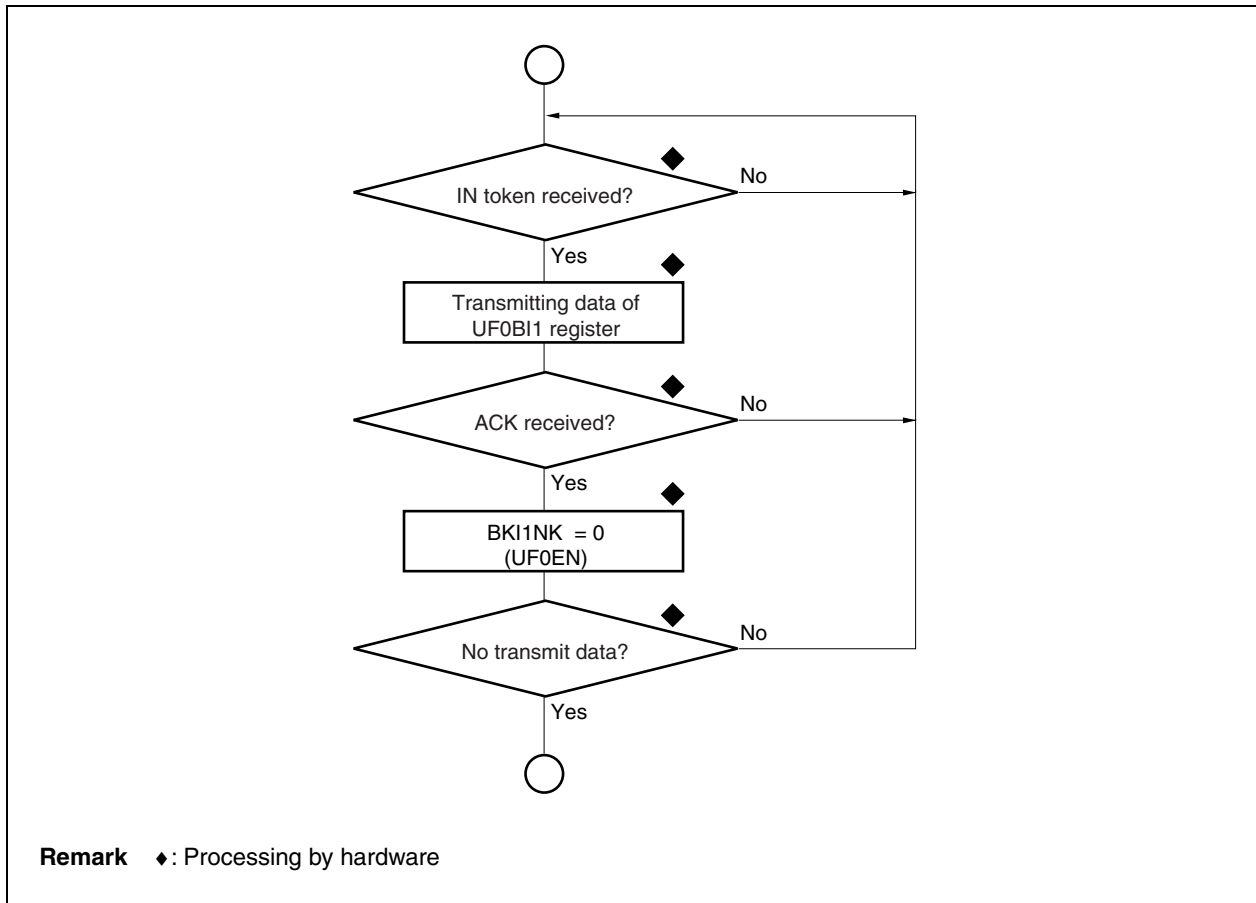
Figure 11-21. Processing for Bulk Transfer (IN) (Endpoint1)



Remarks 1. n = 2, 3

2. ♦: Processing by hardware

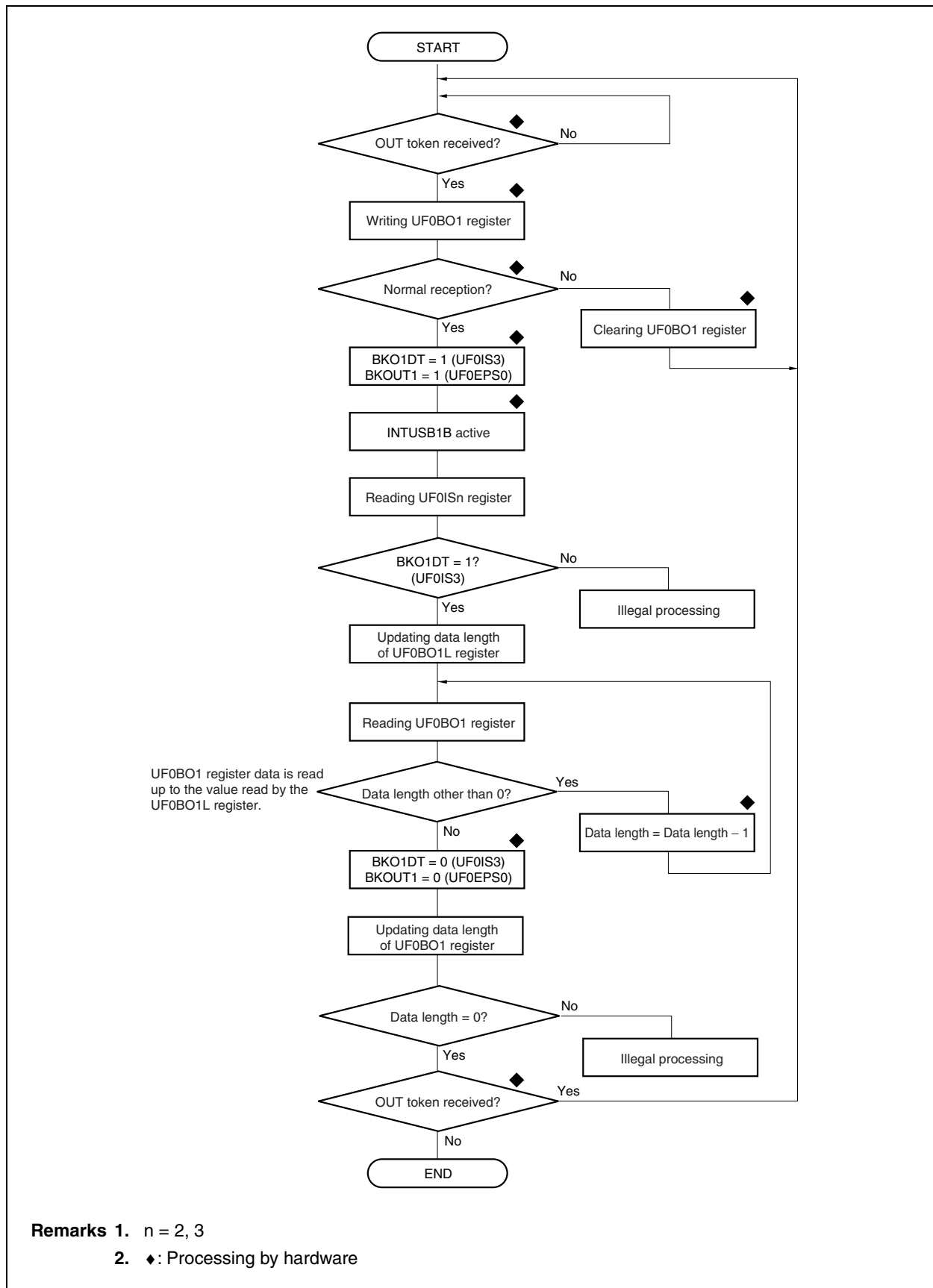
Figure 11-22. Parallel Processing by Hardware



(5) Processing for bulk transfer (OUT)

Bulk transfer (OUT) is allocated to Endpoint2 and Endpoint4. The flowchart shown below illustrates how Endpoint2 is controlled. Endpoint4 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint4, therefore, read the bit names of Endpoint2 in the flowchart as those of Endpoint4.

Figure 11-23. Normal Processing for Bulk Transfer (OUT) (Endpoint2)



During bulk transfer (OUT), more data may be transmitted from the host than expected by the system. Endpoint2 and Endpoint4 for bulk transfer (OUT) of the V850E/ME2 consist of two 64-byte buffers so that NAK responses are suppressed as much as possible and data can be read from the CPU side even while the bus side is being accessed as the transfer rate of the USB bus increases. Consequently, if the host sends more data than expected by the system, up to 128 bytes of extra data may be automatically received in the worst case. In this case, change the control flow from that of the normal processing of Endpoint2 and Endpoint4 to the flow illustrated below when the quantity of data expected by the system has decreased to two packets. This flowchart illustrates how Endpoint2 is controlled. Endpoint4 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint4, therefore, read the bit names of Endpoint2 in the flowchart as those of Endpoint4.

Figure 11-24. Processing If More Data Than Expected by System Is Transmitted (Endpoint2) (1/2)

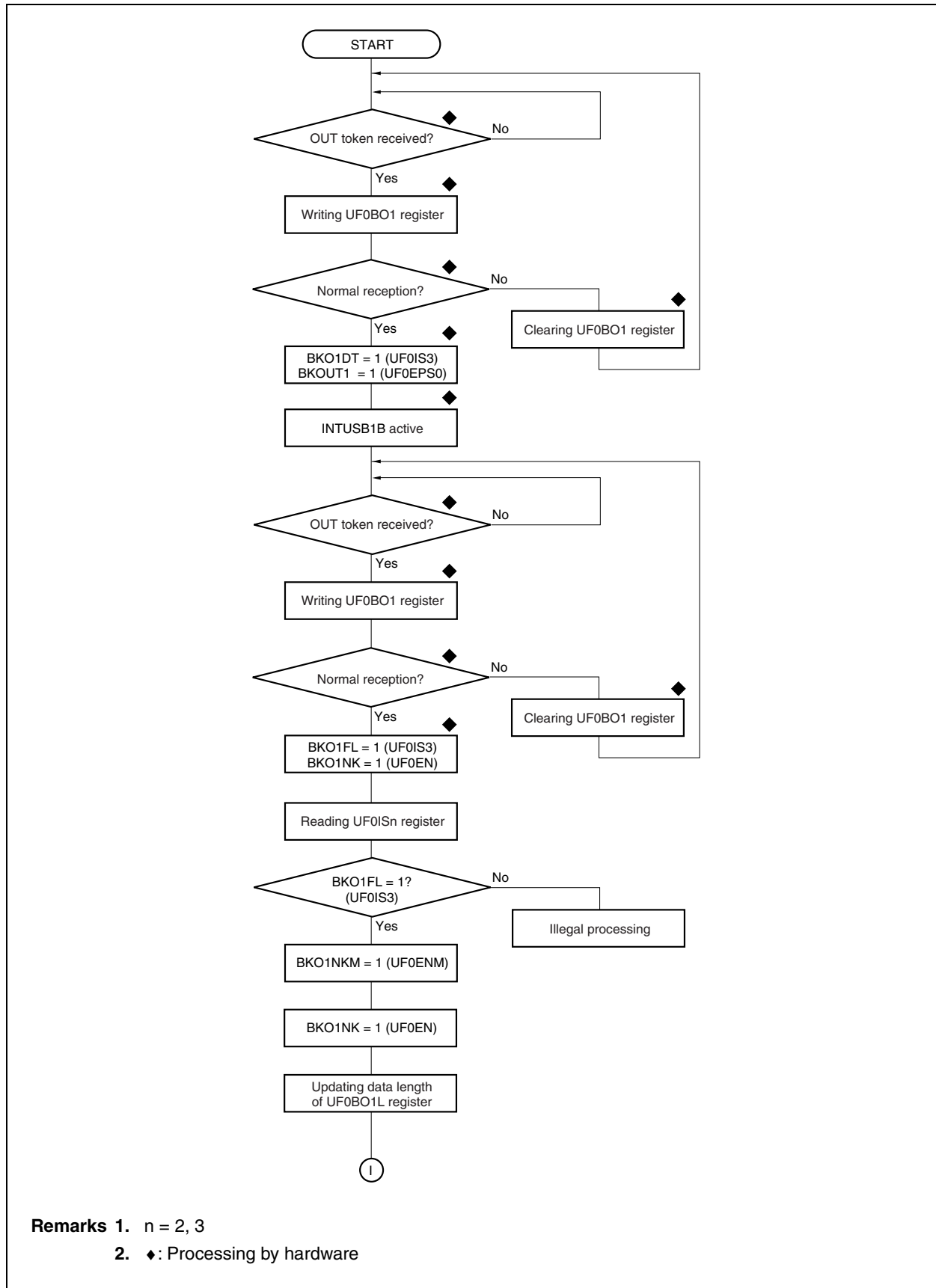
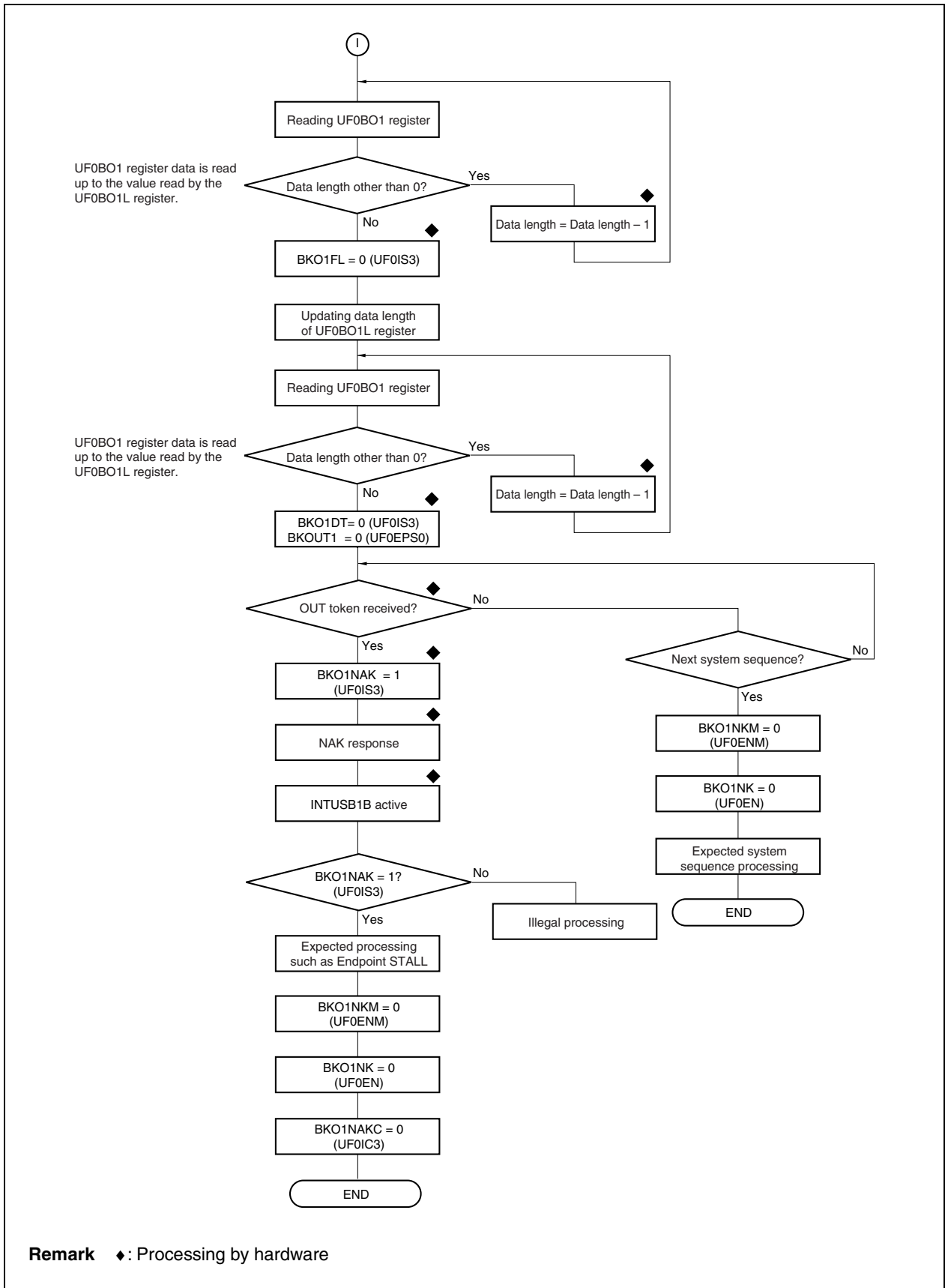


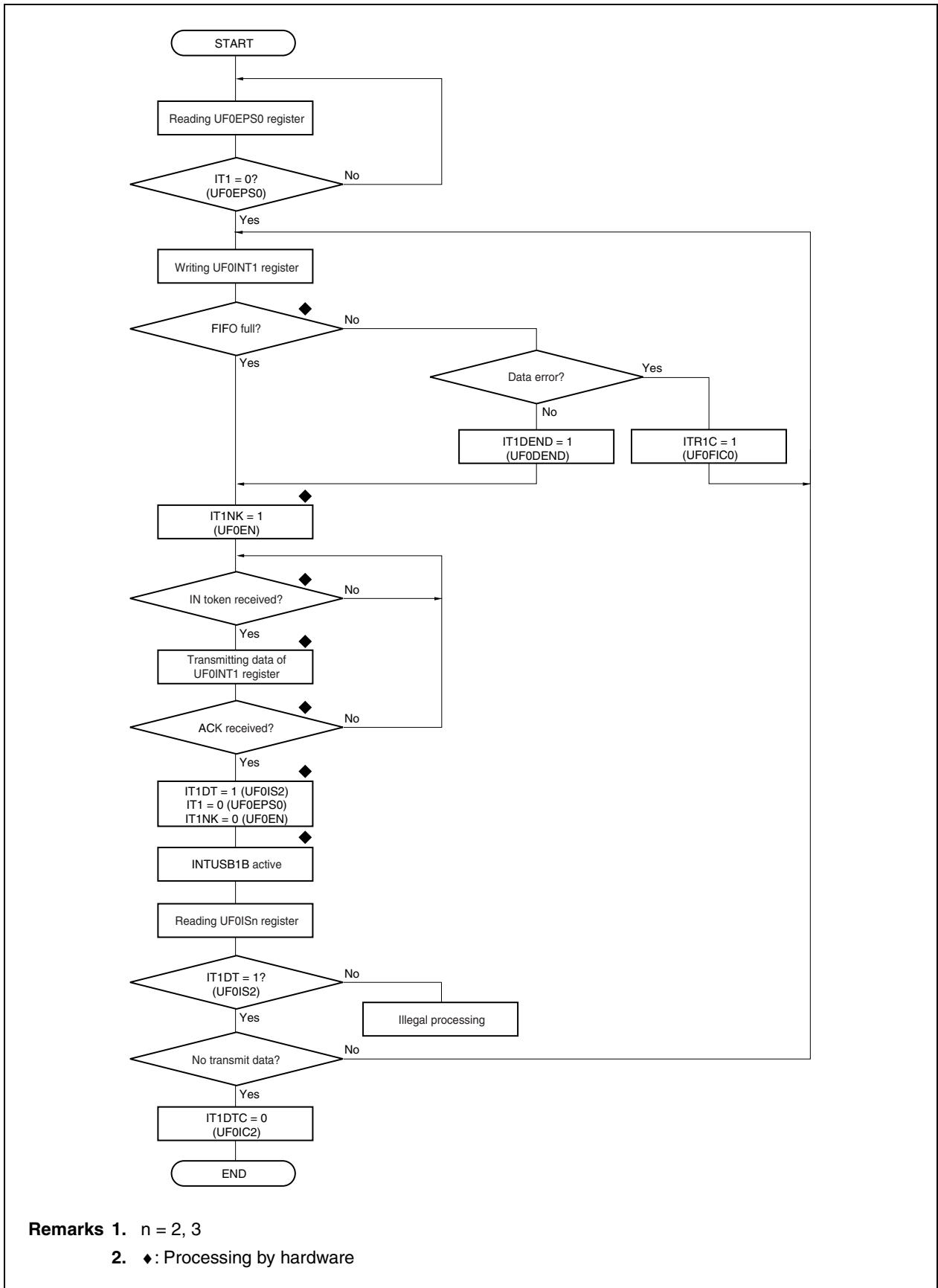
Figure 11-24. Processing If More Data Than Expected by System Is Transmitted (Endpoint2) (2/2)



(6) Processing for interrupt transfer (IN)

Interrupt transfer (IN) is allocated to Endpoint7 and Endpoint8. The flowchart shown below illustrates how Endpoint7 is controlled. Endpoint8 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint8, therefore, read the bit names of Endpoint7 in the flowchart as those of Endpoint8.

Figure 11-25. Processing for Interrupt Transfer (IN) (Endpoint7)



- Remarks 1. n = 2, 3
 2. ♦: Processing by hardware

11.7.4 Suspend/Resume processing

How Suspend/Resume processing is performed differs depending on the configuration of the system. One example is given below.

Figure 11-26. Example of Suspend/Resume Processing (1/3)

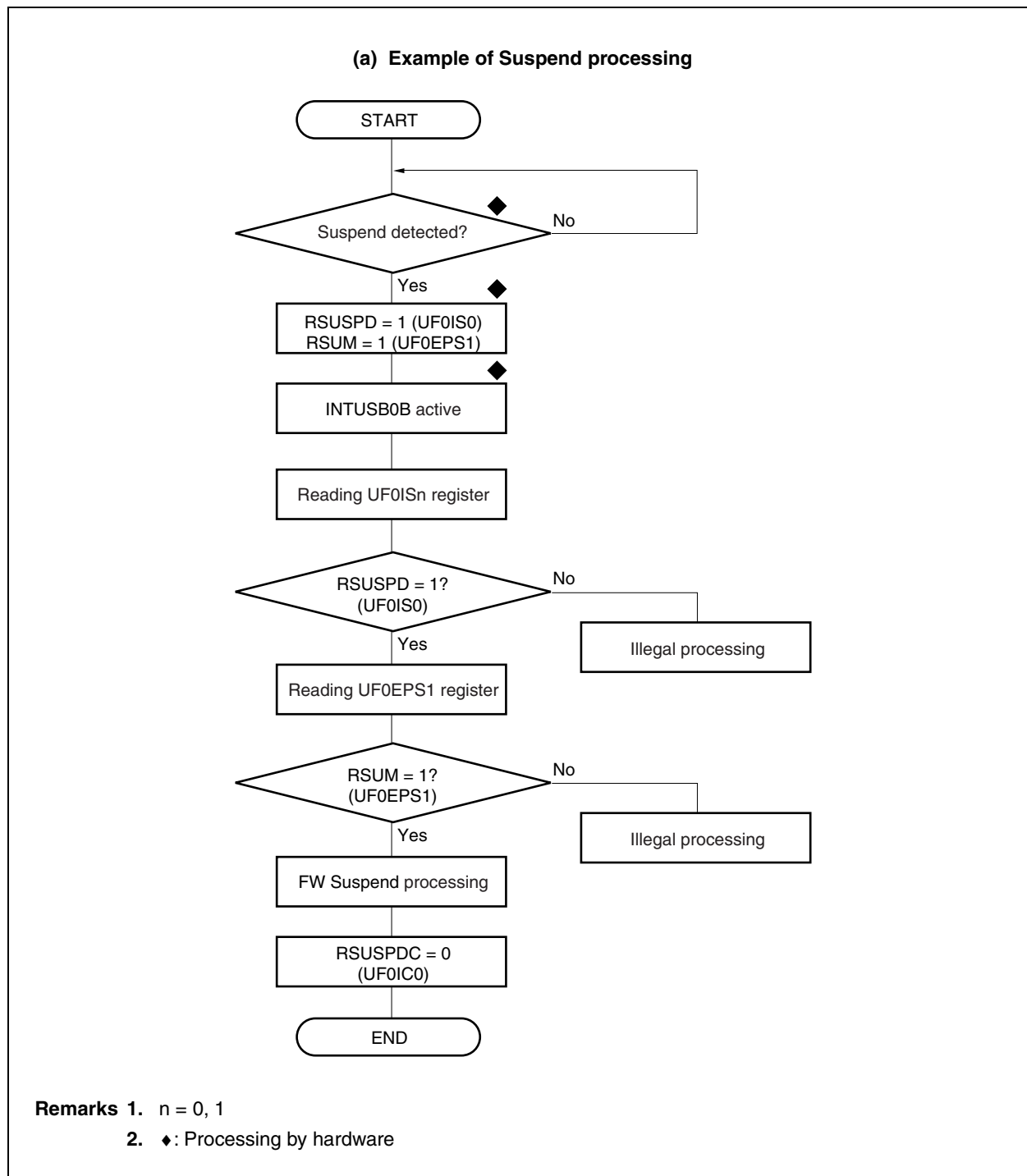


Figure 11-26. Example of Suspend/Resume Processing (2/3)

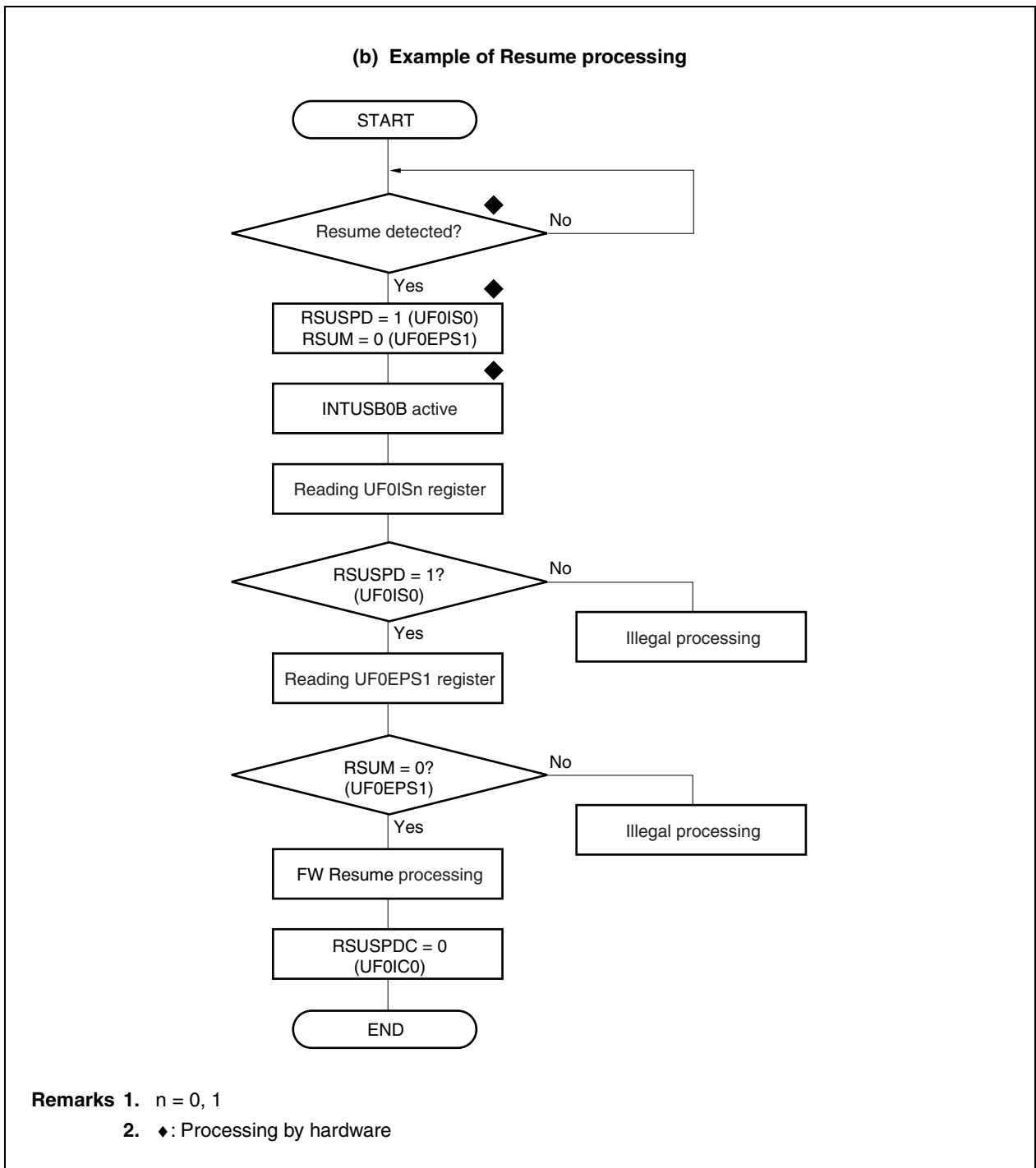
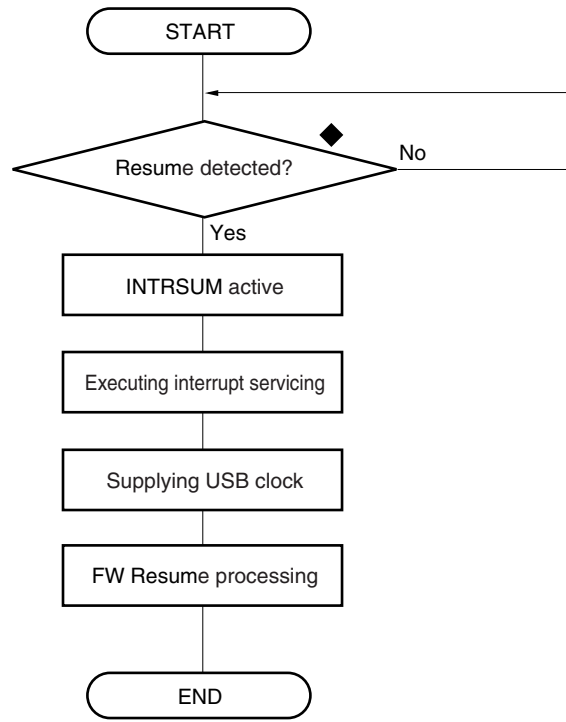


Figure 11-26. Example of Suspend/Resume Processing (3/3)

(c) Example of Resume processing (when supply of USB clock to USBF is stopped)



Remark ◆: Processing by hardware

11.7.5 Processing after power application

The processing to be performed after power application differs depending on the configuration of the system. One example is given below.

Figure 11-27. Example of Processing After Power Application/Power Failure (1/3)

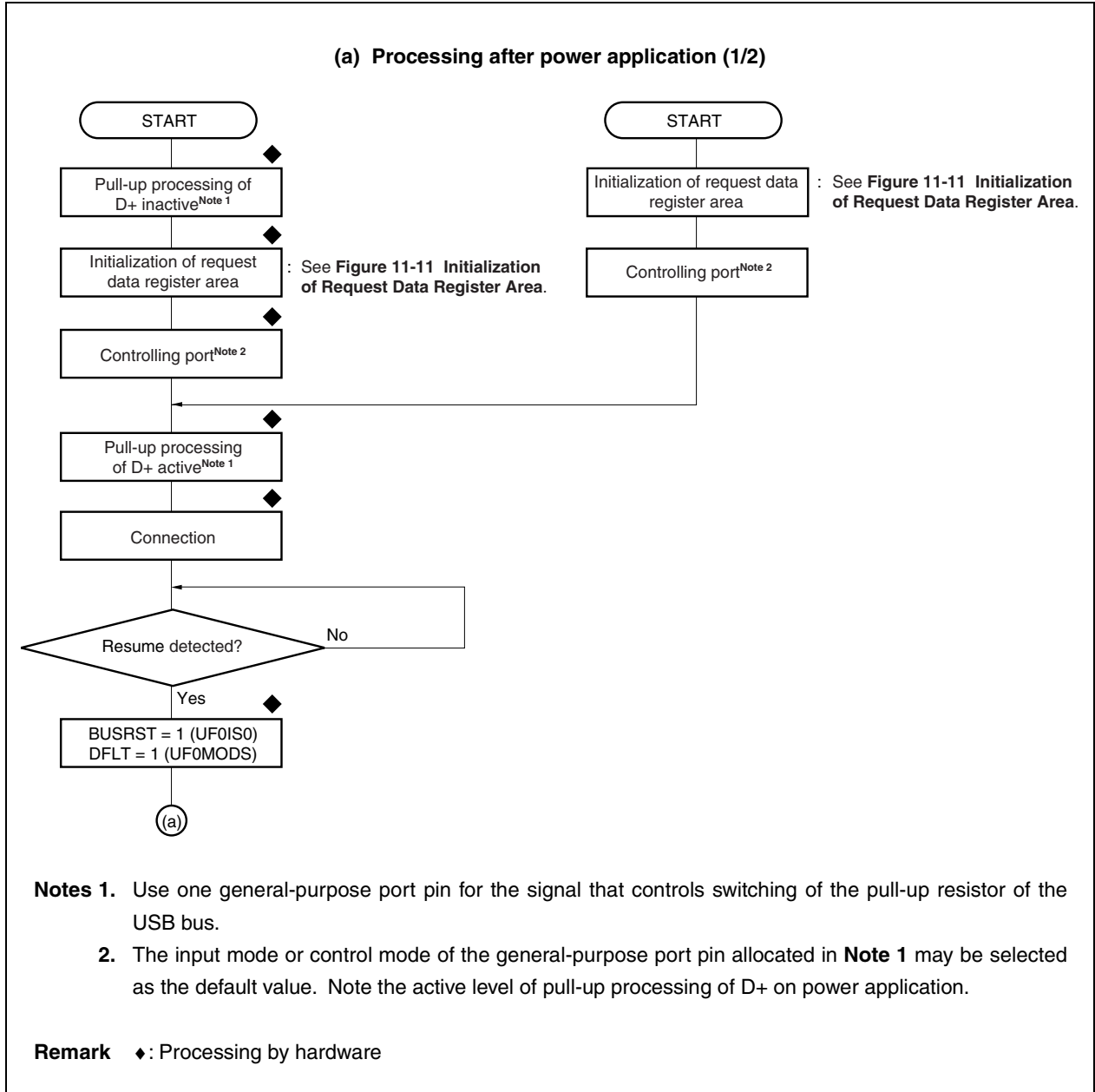


Figure 11-27. Example of Processing After Power Application/Power Failure (2/3)

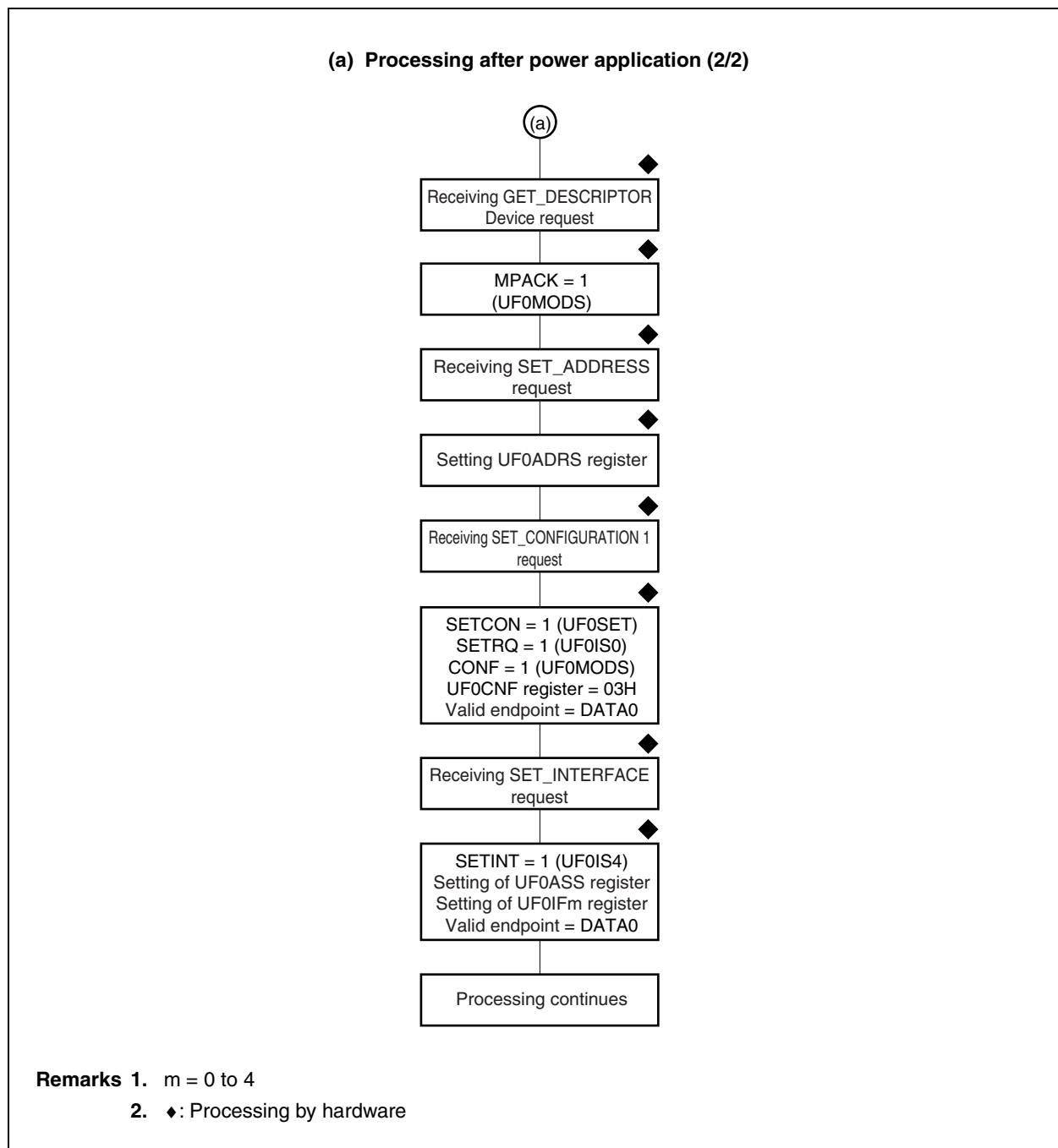
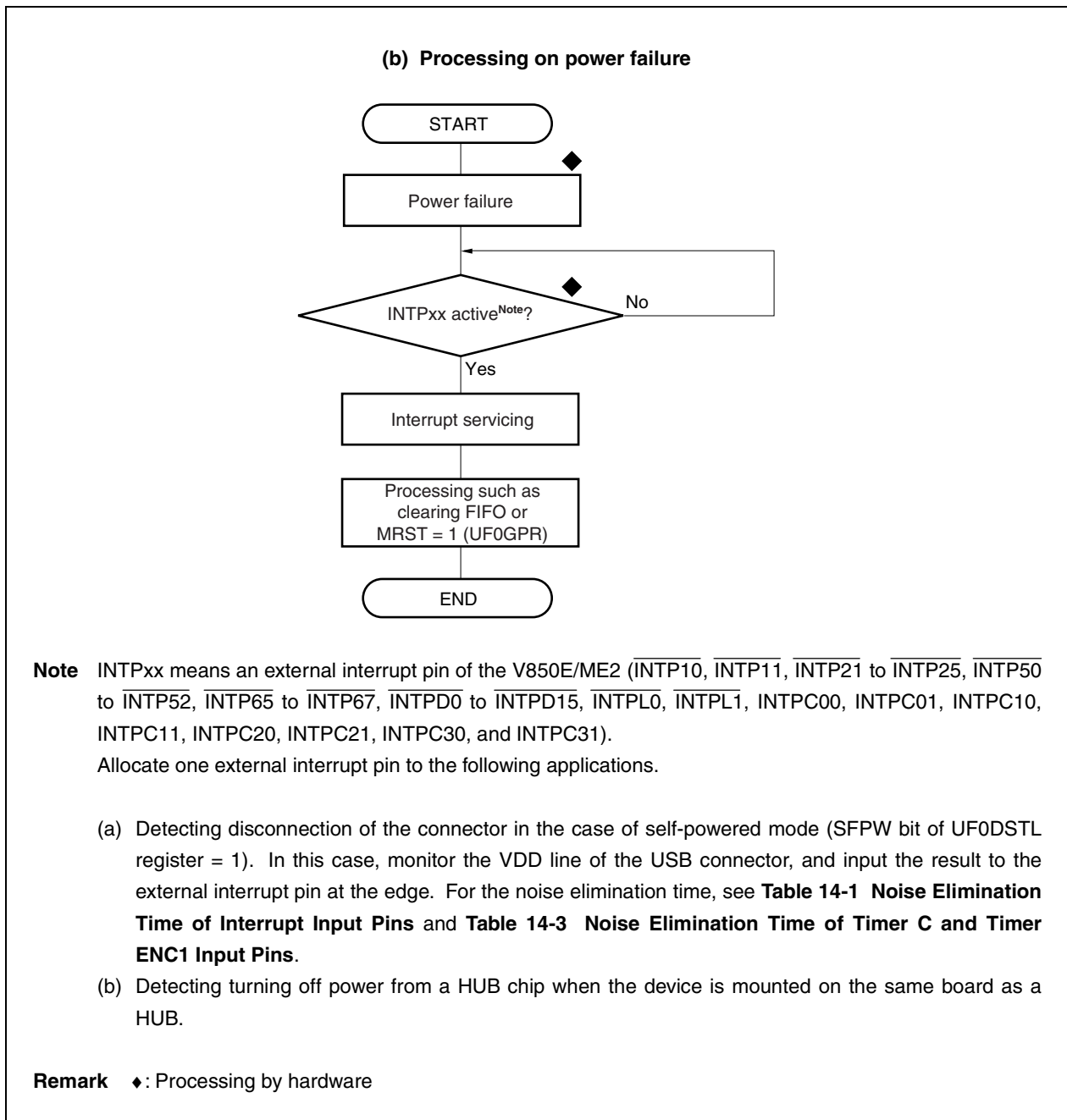


Figure 11-27. Example of Processing After Power Application/Power Failure (3/3)



11.7.6 Receiving data for bulk transfer (OUT) in DMA mode

Bulk transfer (OUT) is allocated to Endpoint2 and Endpoint4. The flowchart shown below illustrates how Endpoint2 is controlled when DMA is used. Endpoint4 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint4, therefore, read the bit names of Endpoint2 in the flowchart as those of Endpoint4. The control flowchart shown below illustrates how remaining data is read by the CPU.

If data for bulk transfer (OUT) has been correctly received by setting the DQBO1MS bit of the UF0IDR register to 1, the DMA request signal for Endpoint2, instead of an interrupt request (INTUSB1B), becomes active. This DMA request signal for Endpoint2 operates according to the setting of the MODEn bit of the UF0IDR register (n = 0, 1). If all the data stored in the UF0BO1 register has been read by DMA, the DMA request signal for Endpoint2 becomes inactive. In this status, if data for the next bulk transfer (OUT) has been correctly received, the DMA request signal for Endpoint2 becomes active again. If the data for bulk transfer (OUT) that has been received is equal to or less than the FIFO size, a Short interrupt request is issued and the USBSP2B signal becomes active, as soon as reading the data by DMA is completed. As a result, the DQBO1MS bit of the UF0IDR register is cleared to 0, and the DMA request signal for Endpoint2 becomes inactive. To read data by DMA again, set the DQBO1MS bit to 1 again. If DMA is completed by the DMA end signal for Endpoint2, the DQBO1MS bit of the UF0IDR register is cleared to 0, and the DMA request signal for Endpoint2 becomes inactive. At the same time, the DMA_END interrupt request is issued. If data remains in the UF0BO1 register at this time, DMA can be started again by setting the DQBO1MS bit of the UF0IDR register again. However, the data for bulk transfer (OUT) is always equal to or less than the FIFO size. Consequently, a Short interrupt request is issued, the USBSP2B signal becomes active, the DQBO1MS bit is cleared, and the DMA request signal for Endpoint2 becomes inactive, as soon as the data is read by DMA.

- Cautions**
- 1. The DMA request signal for Endpoint n (n = 2, 4) becomes active in the demand mode (MODE1 and MODE0 bits of the UF0IDR register = 10), as long as there is data to be transferred.**
 - 2. The DMA request signal for Endpoint n (n = 2, 4) becomes active in the single mode (MODE1 and MODE0 bits of the UF0IDR register = 0X (X: don't care)) if there is data to be transferred, but this signal becomes inactive each time one byte has been transferred. This operation is repeated until there is no more data to be transferred.**

Figure 11-28. DMA Processing by Bulk Transfer (OUT) (1/3)

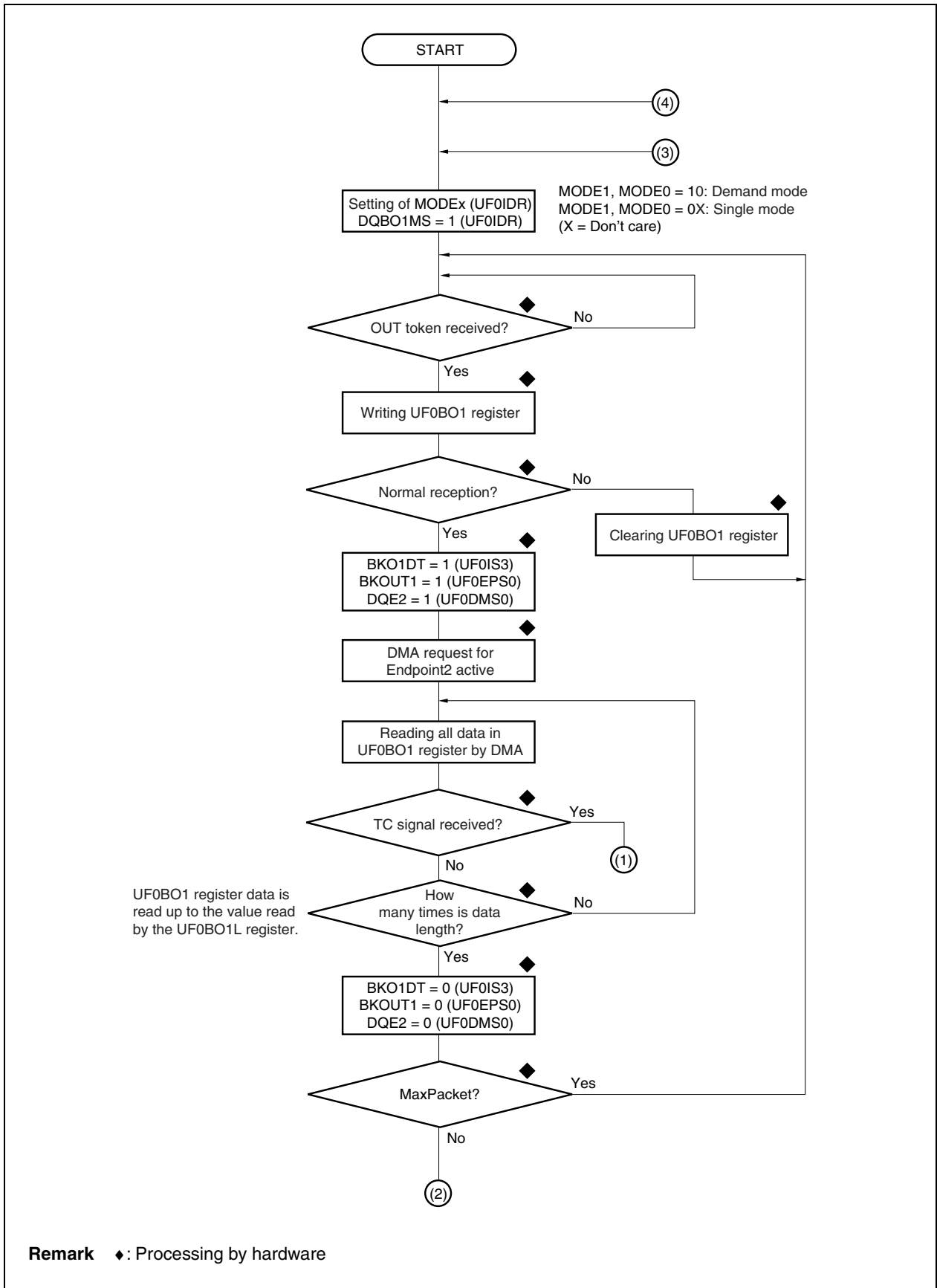
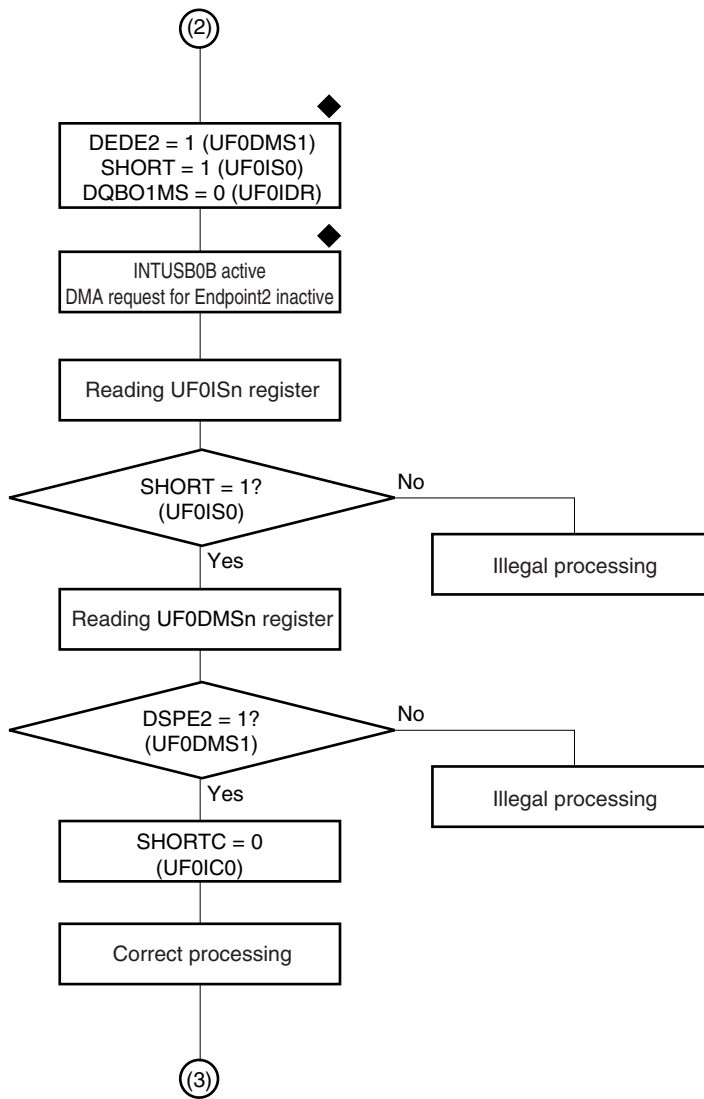


Figure 11-28. DMA Processing by Bulk Transfer (OUT) (2/3)



Remarks 1. n = 0, 1

2. ♦: Processing by hardware

11.7.7 Transmitting data for bulk transfer (IN) in DMA mode

Bulk transfer (IN) is allocated to Endpoint1 and Endpoint3. The flowchart shown below illustrates how Endpoint1 is controlled when DMA is used. Endpoint3 can also be controlled in the same sequence. To use this flowchart as the control flow of Endpoint3, therefore, read the bit names of Endpoint1 in the flowchart as those of Endpoint3.

If data for bulk transfer (IN) can be written by setting the DQBI1MS bit of the UF0IDR register to 1, the DMA request signal for Endpoint1, instead of an interrupt request (INTUSB1B), becomes active. This DMA request signal for Endpoint1 operates according to the setting of the MODEn bit of the UF0IDR register (n = 0, 1). If all the data that can be written to the UF0BI1 register has been written by DMA, the DMA request signal for Endpoint1 becomes inactive. In this status, the toggle operation of the FIFO takes place and, if data for bulk transfer (IN) can be written, the DMA request signal for Endpoint1 becomes active again. The automatic toggle operation of the FIFO is not executed even if the FIFO has become full as a result of DMA transfer, unless the BK11T bit of the UF0DEND register is set to 1. Therefore, be sure to set the BK11DED bit of the UF0DEND register to 1 to transfer data. If DMA is completed by the DMA end signal for Endpoint1, the DQBI1MS bit of the UF0IDR register is cleared to 0, and the DMA_END interrupt request is issued. To transmit a short packet at this time when the FIFO is not full, set the BK11DED bit of the UF0DEND register to 1.

- Cautions**
- 1. The DMA request signal for Endpoint n (n = 1, 3) becomes active in the demand mode (MODE1 and MODE0 bits of the UF0IDR register = 10), as long as data can be transferred.**
 - 2. The DMA request signal for Endpoint n (n = 1, 3) becomes active in the single mode (MODE1 and MODE0 bits of the UF0IDR register = 0X (X: don't care)) if data can be transferred, but this signal becomes inactive each time one byte has been transferred. This operation is repeated until there is no more data to be transferred.**

Figure 11-29. DMA Processing by Bulk Transfer (IN) (1/4)

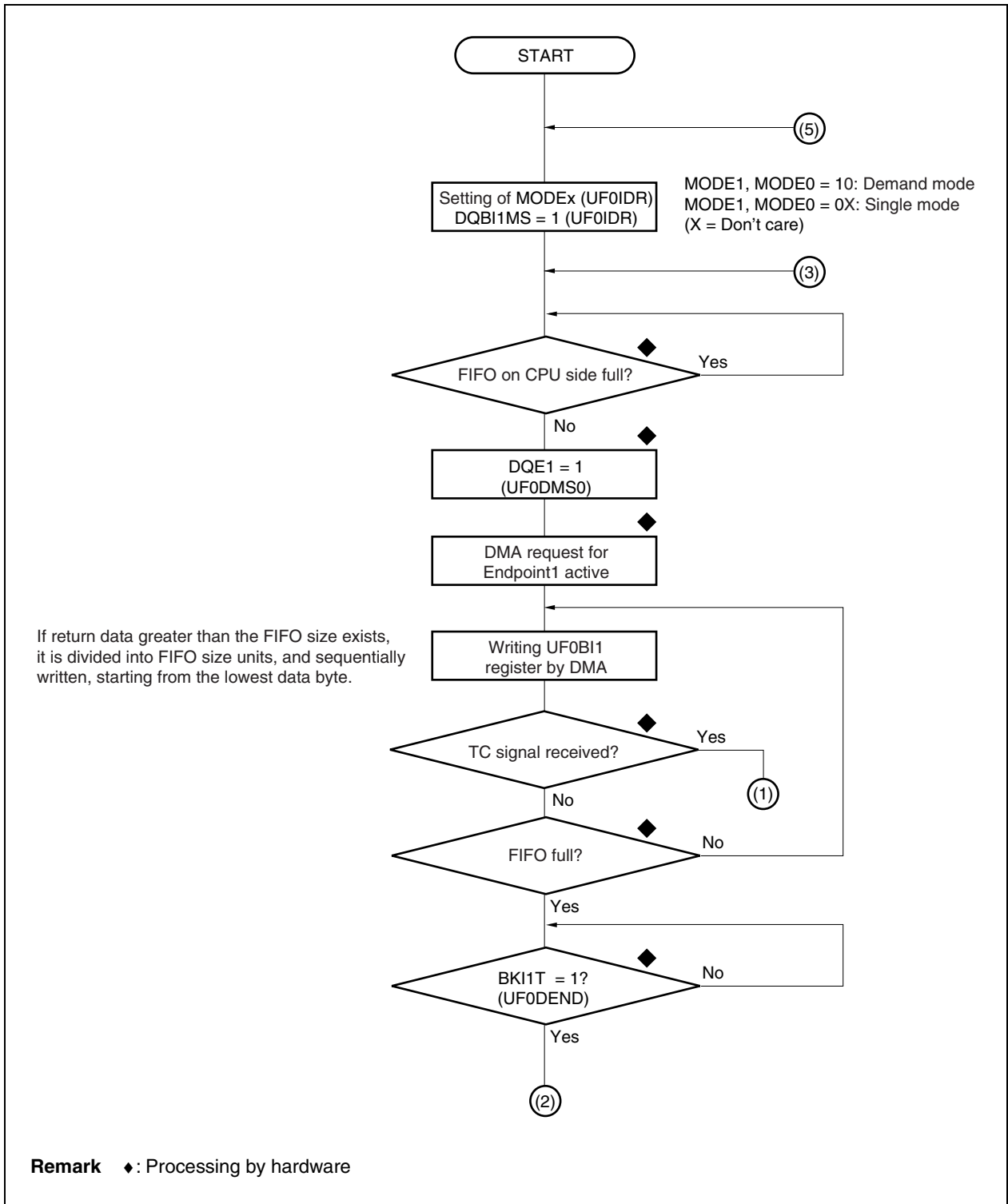


Figure 11-29. DMA Processing by Bulk Transfer (IN) (2/4)

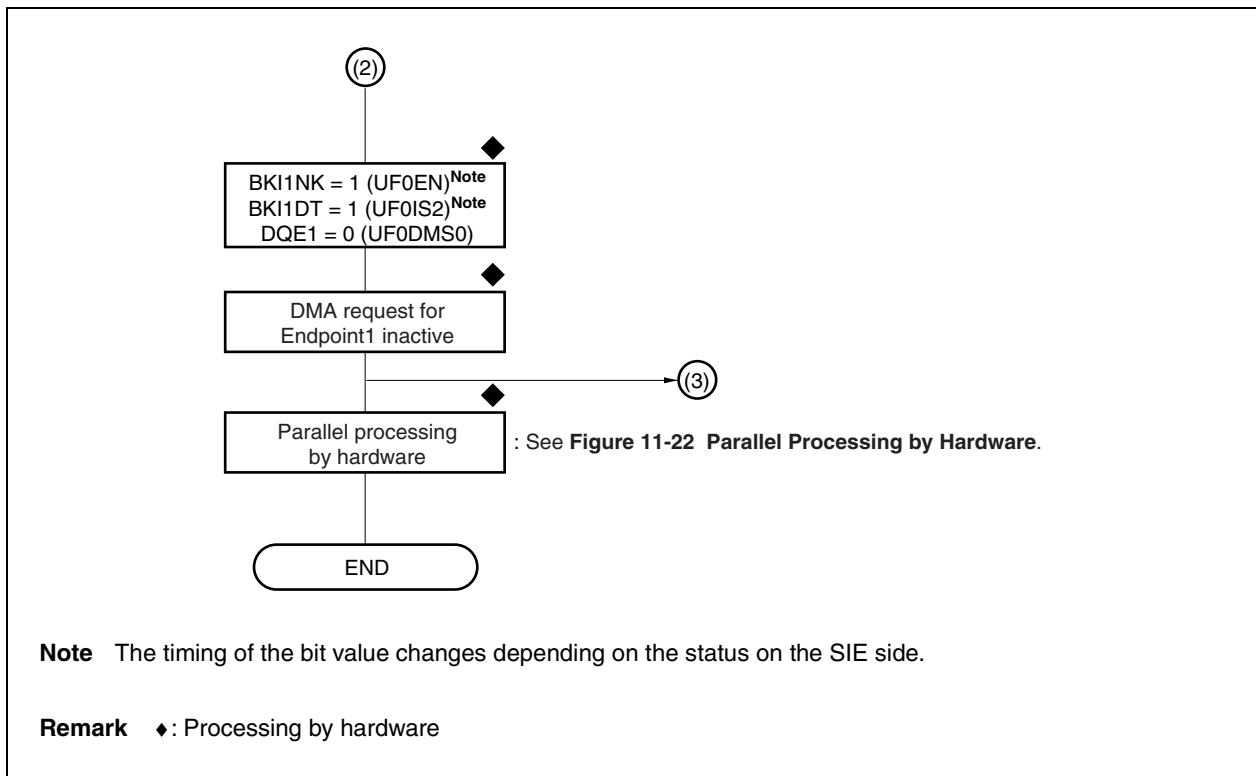


Figure 11-29. DMA Processing by Bulk Transfer (IN) (3/4)

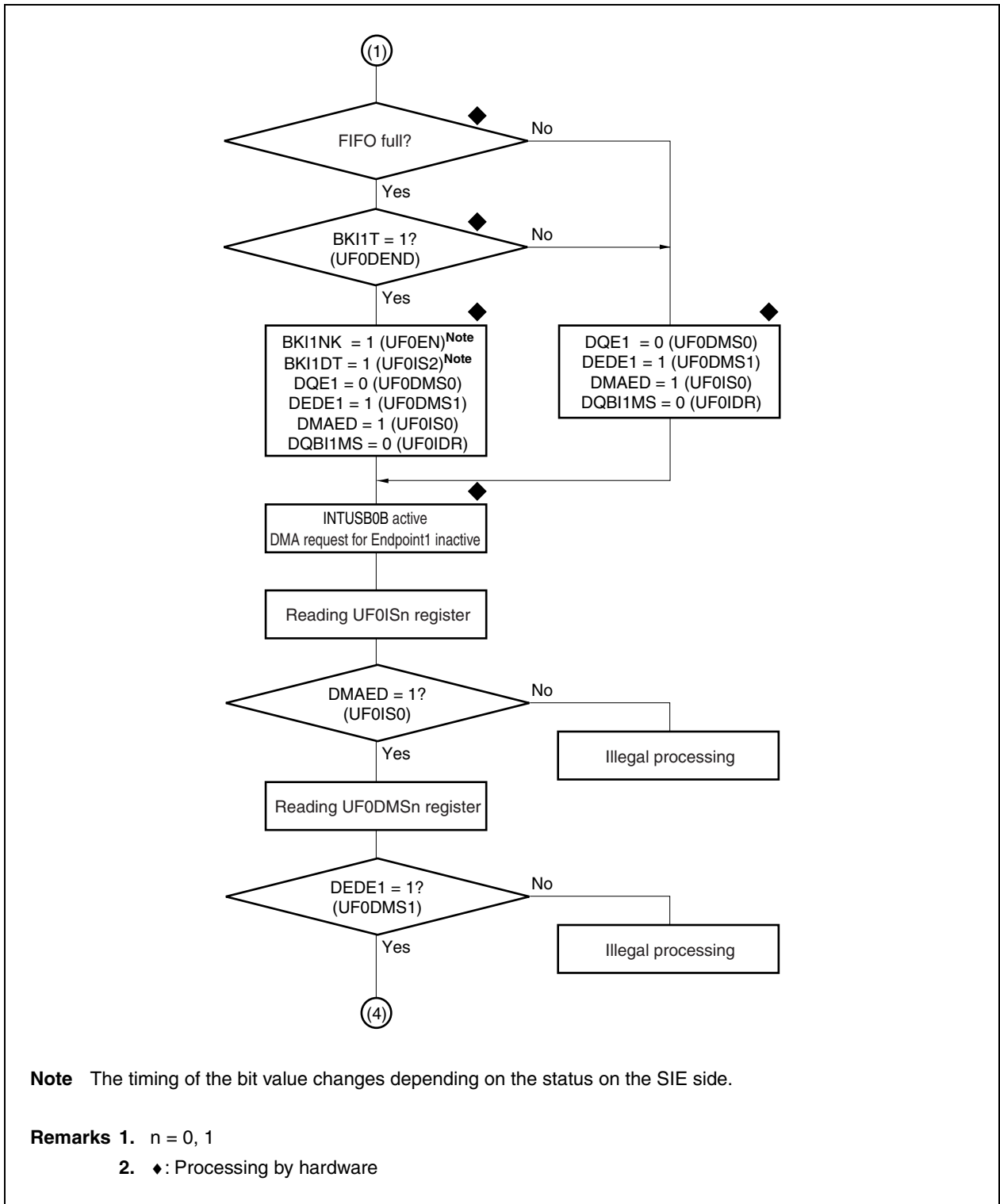
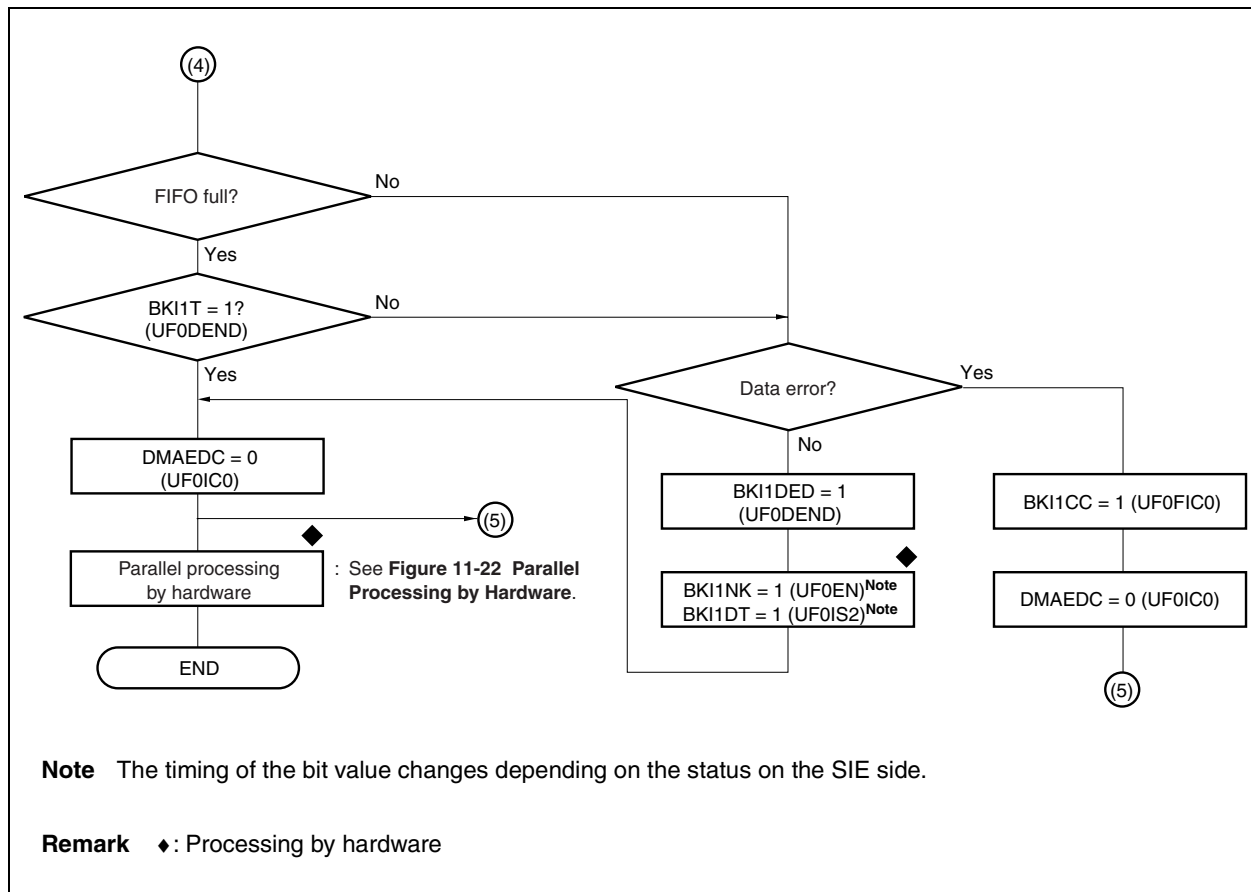


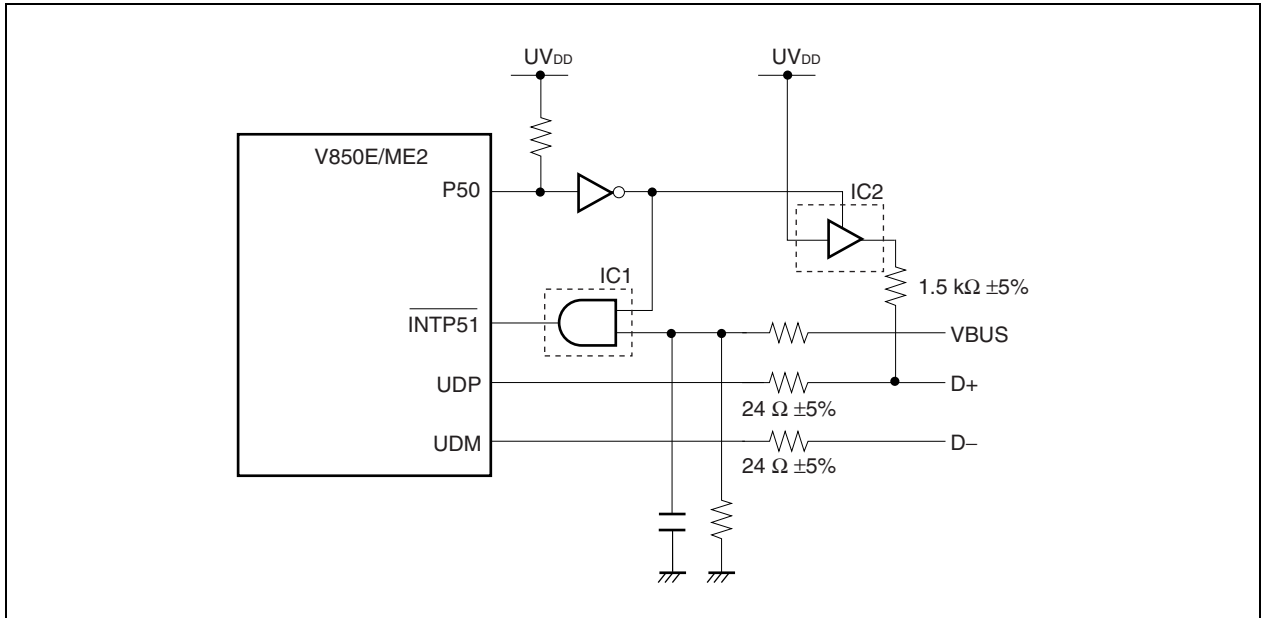
Figure 11-29. DMA Processing by Bulk Transfer (IN) (4/4)



11.7.8 USB connection example

★

Figure 11-30. USB Connection Example

**(1) Pull-up control of D+**

To prohibit connection report (D+ pull-up) to the USB host/HUB (such as while higher priority processing or initialization processing is under execution), the system must control pulling up of D+ via a general-purpose port. As shown in the circuit example in Figure 11-30, control the pull-up control signal of D+ and VBUS input signal by using a general-purpose port and USB cable VBUS (AND circuit). In Figure 11-30, pulling up D+ is prohibited when the general-purpose port is high level (secure the high level of the general-purpose port pin by pulling it up because the port pin is in the input mode by default). Use an IC to which power can be applied when the system power supply is off as IC2 shown in Figure 11-30.

(2) Detecting USB cable connection/disconnection

The USB function controller (USBF) requires a VBUS input signal that recognizes connection and disconnection, because the state of USBF is managed by hardware. Voltage (5 V) is applied from the USB host/HUB as the VBUS input signal if the USB cable VBUS is connected to the USB host/HUB when power to the USBF is turned off. Therefore, use an IC to which power can be applied when the system power supply is off as IC1 shown in Figure 11-30.

CHAPTER 12 A/D CONVERTER

12.1 Features

- Analog input: 8 channels
- 10-bit A/D converter
- On-chip A/D conversion result register (ADCR0 to ADCR7)
10 bits × 8
- A/D conversion trigger mode
 - A/D trigger mode
 - Timer trigger mode
 - External trigger mode
- Successive approximation method

12.2 Configuration

The A/D converter of the V850E/ME2 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 ADCR7) to perform A/D conversion operations.

(1) Input circuit

The input circuit selects the analog input (ANI0 to ANI7) according to the mode set by the ADM0, ADM1, and ADM2 registers.

(2) C-Array

Holds the charge of the differential voltage between the voltage input from the analog input pins (ANI0 to ANI7) and the reference voltage ($1/2 AV_{DD}$), and redistributes the sampled charges.

(3) C-Dummy

This block holds the reference voltage ($1/2 AV_{DD}$) and assigns the reference of the comparator input.

(4) Voltage comparator

The voltage comparator compares the C-Array comparison potential with the C-Dummy reference potential.

(5) A/D conversion result register n (ADCRn), A/D conversion result register nH (ADCRnH)

ADCRn is a 10-bit register that holds A/D conversion results. Each time A/D conversion is completed, the conversion results are loaded from the successive approximation register (SAR).

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

(6) ANI0 to ANI7 pins

These are 8-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 ANI7 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.

(7) AV_{REFM} and AV_{REFP} pins

This is the pin for inputting the reference voltage of the A/D converter. It converts signals input to the ANI0 to ANI7 pins to digital signals based on the voltage applied between AV_{REFM} and AV_{REFP} .

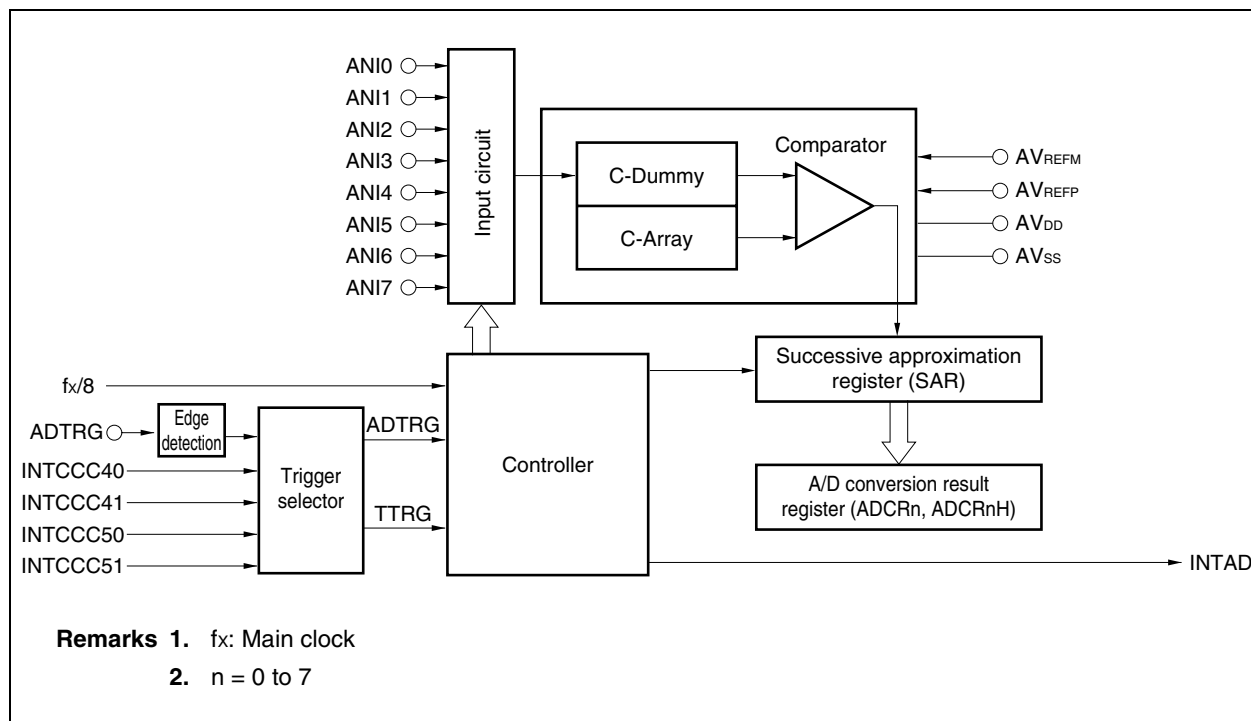
(8) AV_{SS} pin

This is the ground pin of the A/D converter. Always use this pin at the same potential as that of the EV_{SS} pin even when the A/D converter is not used.

(9) AV_{DD} pin

This is the analog power supply pin of the A/D converter. Always use this pin at the same potential as that of the EV_{DD} pin even when the A/D converter is not used.

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



Cautions

1. If there is noise at the analog input pins (ANI0 to ANI7) or at the reference voltage input pin (AV_{REFP}, AV_{REFM}), 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.
2. Do not apply a voltage outside the AV_{REFM} to AV_{REFP} 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 that specifies the operation mode, and executes conversion operations. This register can be read or written in 8-bit or 1-bit units. However, bit 6 can only be read. Writing this bit is ignored.

- Cautions**
1. When the ADCE bit is 1 in the timer trigger mode and external 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 or external trigger mode without clearing the ADCE bit, the trigger input standby state is set immediately after changing the register.
 2. Changing the setting of the BS and MS bits is prohibited while A/D conversion is enabled (ADCE bit = 1).
 3. When 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.

	<7>	<6>	5	4	3	2	1	0	Address	After reset
ADM0	ADCE	ADCS	BS	MS	0	0	0	0	FFFFF200H	00H

Bit position	Bit name	Function
7	ADCE	Enables or disables A/D conversion operation. 0: Disabled 1: Enabled
6	ADCS	Indicates the status of A/D converter. This bit is read only. 0: Stopped 1: Operating
5	BS	Specifies buffer mode in the select mode. 0: 1-buffer mode 1: 4-buffer mode
4	MS	Specifies operation mode of A/D converter. 0: Scan mode 1: Select mode

(2) A/D converter mode register 1 (ADM1)

The ADM1 register is an 8-bit register that specifies the conversion operation time and trigger mode. This register can be read or written in 8-bit or 1-bit units.

- Cautions**
1. Changing the setting of the EGA1, EGA0, and FR3 to FR0 bits is prohibited while A/D conversion is enabled (ADCE bit = 1 in the ADM0 register).
 2. When 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.

	7	6	5	4	3	2	1	0	Address	After reset
ADM1	EGA1	EGA0	TRG1	TRG0	FR3	FR2	FR1	FR0	FFFFFF201H	00H

Bit position	Bit name	Function															
7, 6	EGA1, EGA0	Specify valid edge of ADTRG.															
		<table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">EGA1</th> <th style="width: 15%;">EGA0</th> <th>ADTRG valid edge specification</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>No edge detected (does not operate as external trigger)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Falling edge detected</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Rising edge detected</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Both edges detected</td> </tr> </tbody> </table>	EGA1	EGA0	ADTRG valid edge specification	0	0	No edge detected (does not operate as external trigger)	0	1	Falling edge detected	1	0	Rising edge detected	1	1	Both edges detected
		EGA1	EGA0	ADTRG valid edge specification													
		0	0	No edge detected (does not operate as external trigger)													
		0	1	Falling edge detected													
1	0	Rising edge detected															
1	1	Both edges detected															
5, 4	TRG1, TRG0	Specify the trigger mode.															
		<table border="1" style="width: 100%;"> <thead> <tr> <th style="width: 15%;">TRG1</th> <th style="width: 15%;">TRG0</th> <th>Trigger mode</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>A/D trigger mode</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Timer trigger mode</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>External trigger mode</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	TRG1	TRG0	Trigger mode	0	0	A/D trigger mode	0	1	Timer trigger mode	1	0	External trigger mode	1	1	Setting prohibited
		TRG1	TRG0	Trigger mode													
		0	0	A/D trigger mode													
		0	1	Timer trigger mode													
1	0	External trigger mode															
1	1	Setting prohibited															

Remark Sets A/D conversion operation time by using bits FR3 to FR0. For details, see **Table 12-1**.



Table 12-1. Setting of A/D Conversion Operation Time

FR3	FR2	FR1	FR0	Number of Conversion Clocks	Conversion Operation Time ^{Note 1}				A/D Stabilization Time ^{Note 2}
					f _x = 150 MHz	f _x = 133 MHz	f _x = 100 MHz	f _x = 80 MHz	
0	0	0	0	128	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	64/f _x
0	0	0	1	256	Setting prohibited	Setting prohibited	2.56 μs	3.20 μs	128/f _x
0	0	1	0	384	2.56 μs	2.89 μs	3.84 μs	4.80 μs	160/f _x
0	0	1	1	512	3.42 μs	3.85 μs	5.12 μs	6.40 μs	160/f _x
0	1	0	0	640	4.27 μs	4.82 μs	6.40 μs	8.00 μs	160/f _x
0	1	0	1	768	5.12 μs	5.78 μs	7.68 μs	9.60 μs	160/f _x
0	1	1	0	896	5.98 μs	6.74 μs	8.96 μs	Setting prohibited	160/f _x
0	1	1	1	1,024	6.83 μs	7.70 μs	Setting prohibited	Setting prohibited	160/f _x
1	0	0	0	1,152	7.68 μs	8.67 μs	Setting prohibited	Setting prohibited	160/f _x
1	0	0	1	1,280	8.54 μs	9.63 μs	Setting prohibited	Setting prohibited	160/f _x
1	0	1	0	1,408	9.39 μs	Setting prohibited	Setting prohibited	Setting prohibited	160/f _x
1	0	1	1	1,536	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	–
1	1	0	0	1,664	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	–
1	1	0	1	1,792	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	–
1	1	1	0	1,920	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	–
1	1	1	1	2,048	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	–

- Notes**
1. Set the conversion operation time in the range of 2 to 10 μs (target value).
 2. After the ADCE bit is set to “1” from “0” to secure the stabilization time of the A/D converter, conversion is started after the A/D stabilization time has elapsed only before the first A/D conversion is executed.

- Cautions**
1. Do not change the set value of the A/D conversion time (FR3 to FR0 bits) during an A/D conversion operation (ADCE bit = 1). To change the value, clear the ADCE bit to 0.
 2. When the trigger mode (TRG1 and TGR0 bits) is changed midway, A/D conversion can be started immediately without having to secure the A/D stabilization time by re-setting the ADCE bit to “1”.

Remark f_x: Main clock

(3) A/D converter mode register 2 (ADM2)

The ADM2 register is an 8-bit register that specifies the analog input pin of the A/D converter. This register can be read or written in 8-bit or 1-bit units.

- Cautions**

 1. If a channel for which no analog input pin exists is specified, the result of A/D conversion is undefined.
 2. Changing the setting of the ANIS2 to ANIS0 bits is prohibited while A/D conversion is enabled (ADCE bit = 1 in the ADM0 register).
 3. When data is written to the ADM2 register during an A/D conversion operation, the conversion operation is initialized and conversion is executed from the beginning.

	7	6	5	4	3	2	1	0	Address	After reset
ADM2	0	0	0	0	0	ANIS2	ANIS1	ANIS0	FFFFFF202H	00H

Bit position	Bit name	Function																																															
2 to 0	ANIS2 to ANIS0	Specify the analog input pin for A/D conversion.																																															
		<table border="1"> <thead> <tr> <th rowspan="2">ANIS2</th> <th rowspan="2">ANIS1</th> <th rowspan="2">ANIS0</th> <th colspan="2">Specification of pin for A/D conversion</th> </tr> <tr> <th>Select mode</th> <th>Scan mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>ANI0</td> <td>ANI0</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>ANI1</td> <td>ANI0, ANI1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>ANI2</td> <td>ANI0 to ANI2</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>ANI3</td> <td>ANI0 to ANI3</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>ANI4</td> <td>ANI0 to ANI4</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>ANI5</td> <td>ANI0 to ANI5</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>ANI6</td> <td>ANI0 to ANI6</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>ANI7</td> <td>ANI0 to ANI7</td> </tr> </tbody> </table>	ANIS2	ANIS1	ANIS0	Specification of pin for A/D conversion		Select mode	Scan mode	0	0	0	ANI0	ANI0	0	0	1	ANI1	ANI0, ANI1	0	1	0	ANI2	ANI0 to ANI2	0	1	1	ANI3	ANI0 to ANI3	1	0	0	ANI4	ANI0 to ANI4	1	0	1	ANI5	ANI0 to ANI5	1	1	0	ANI6	ANI0 to ANI6	1	1	1	ANI7	ANI0 to ANI7
ANIS2	ANIS1	ANIS0				Specification of pin for A/D conversion																																											
			Select mode	Scan mode																																													
0	0	0	ANI0	ANI0																																													
0	0	1	ANI1	ANI0, ANI1																																													
0	1	0	ANI2	ANI0 to ANI2																																													
0	1	1	ANI3	ANI0 to ANI3																																													
1	0	0	ANI4	ANI0 to ANI4																																													
1	0	1	ANI5	ANI0 to ANI5																																													
1	1	0	ANI6	ANI0 to ANI6																																													
1	1	1	ANI7	ANI0 to ANI7																																													

(4) ADC trigger select register (ADTS)

This is an 8-bit register that specifies the timer trigger signal in the timer trigger mode.

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

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

Caution Before changing the setting of the ADTS register, stop the A/D conversion operation (by clearing the ADCE bit of the ADM0 register to 0). The operation is not guaranteed if the setting of the ADTS register is changed while A/D conversion is enabled (ADCE bit = 1).

	7	6	5	4	<3>	<2>	<1>	<0>	Address	After reset
ADTS	0	0	0	0	TMS3	TMS2	TMS1	TMS0	FFFFFF220H	00H

Bit position	Bit name	Function
3	TMS3	Controls connection of a timer trigger signal (INTCCC51). 0: Timer trigger of ADC is invalid. 1: Timer trigger of ADC is valid.
2	TMS2	Controls connection of a timer trigger signal (INTCCC50). 0: Timer trigger of ADC is invalid. 1: Timer trigger of ADC is valid.
1	TMS1	Controls connection of a timer trigger signal (INTCCC41). 0: Timer trigger of ADC is invalid. 1: Timer trigger of ADC is valid.
0	TMS0	Controls connection of a timer trigger signal (INTCCC40). 0: Timer trigger of ADC is invalid. 1: Timer trigger of ADC is valid.

(5) A/D conversion result registers 0 to 7, 0H to 7H (ADCR0 to ADCR7, ADCR0H to ADCR7H)

The ADCRn register is a 10-bit register holding the A/D conversion results. There are eight 10-bit registers. These registers are read-only in 16-bit or 8-bit units. During 16-bit access, the ADCRn register is specified, and during higher 8-bit access, the ADCRnH register is specified (n = 0 to 7).

When reading the 10-bit data of the A/D conversion results from the ADCRn register, only the higher 10 bits are valid and the lower 6 bits are always read as 0.

ADCRn	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
	ADn9	ADn8	ADn7	ADn6	ADn5	ADn4	ADn3	ADn2	ADn1	ADn0	0	0	0	0	0	0	FFFFF210H to FFFFF21EH	Undefined
ADCRnH	7	6	5	4	3	2	1	0	Address	After reset								
	ADn9	ADn8	ADn7	ADn6	ADn5	ADn4	ADn3	ADn2	FFFFF211H to FFFFF21FH	Undefined								

Remark n = 0 to 7

The correspondence between each analog input pin and the ADCRn register is shown below.

Analog Input Pin	ADCRn Register	
	Select 1 Buffer Mode/ Scan Mode	Select 4 Buffer Mode
ANI0	ADCR0, ADCR0H	ADCR0 to ADCR3, ADCR0H to ADCR3H
ANI1	ADCR1, ADCR1H	
ANI2	ADCR2, ADCR2H	
ANI3	ADCR3, ADCR3H	
ANI4	ADCR4, ADCR4H	ADCR4 to ADCR7, ADCR4H to ADCR7H
ANI5	ADCR5, ADCR5H	
ANI6	ADCR6, ADCR6H	
ANI7	ADCR7, ADCR7H	

The relationship between the analog voltage input to the analog input pins (ANI0 to ANI7) 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 ()

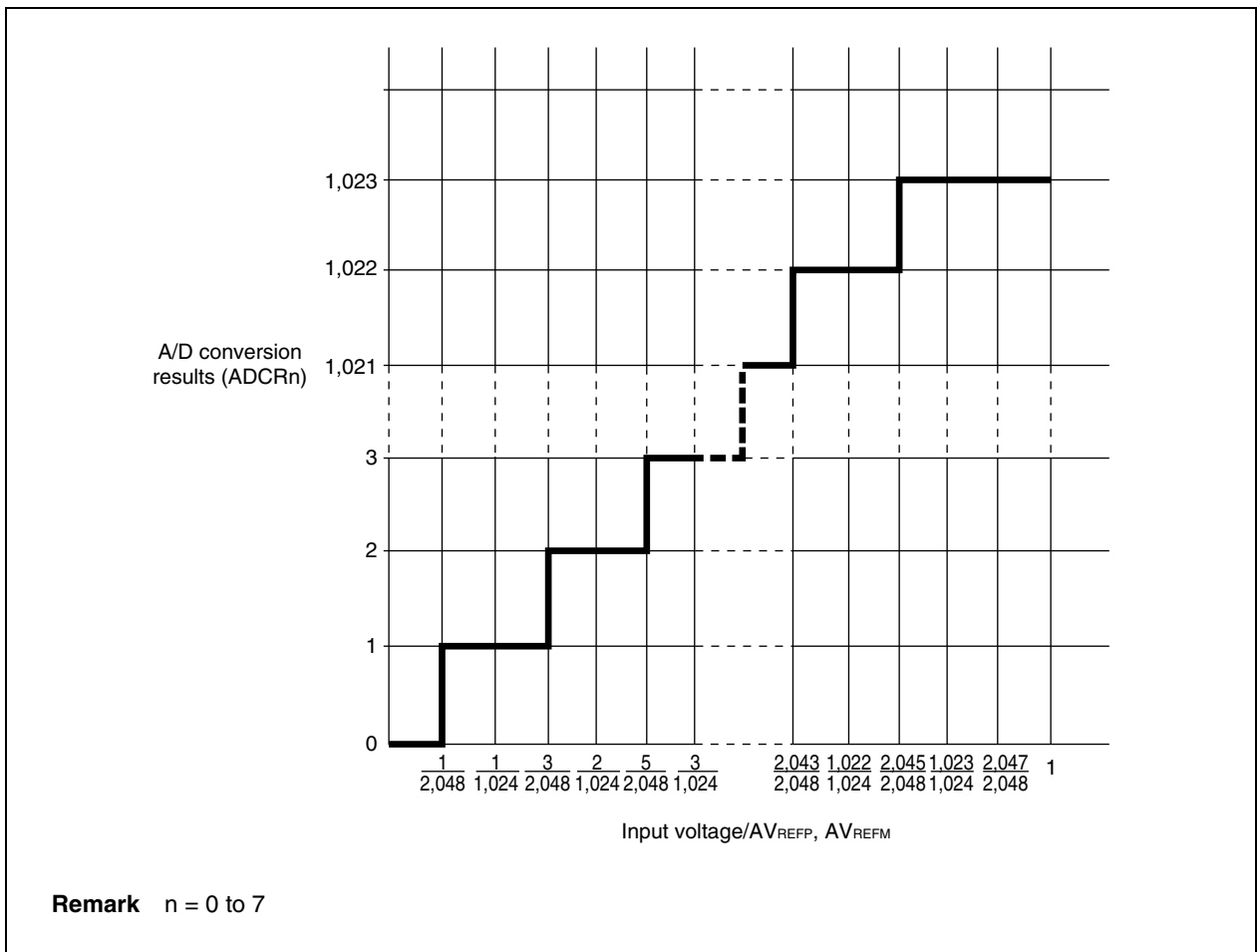
V_{IN}: Analog input voltage

AV_{REF}: AV_{REFM}, AV_{REFP} 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 Operation

12.4.1 Basic operation

A/D conversion is executed by the following procedure.

- (1) The selection of the analog input and specification of the operation mode, trigger mode, etc. should be specified using the ADMn register^{Note 1} (n = 0 to 2).
When the ADCE bit of the ADM0 register is set to 1, A/D conversion starts in the A/D trigger mode. In the timer trigger mode and external trigger mode, the trigger standby state^{Note 2} is set.
- (2) When A/D conversion is started, the C array voltage on the analog input side and the C array voltage on the reference side are compared by the comparator.
- (3) When the comparison of the 10 bits ends, the conversion results are stored in the ADCRn register. When A/D conversion has been performed the specified number of times, the A/D conversion end interrupt (INTAD) is generated (n = 0 to 7).

Notes 1. If the setting of the ADMn register (n = 0 to 2) is changed during A/D conversion, the operation immediately before is stopped, and the result of the conversion is not stored in the ADCRn register (n = 0 to 7). The A/D conversion operation is then initialized, and conversion is executed from the beginning again.

2. During the timer trigger mode and external 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 (ADCS bit = 1 in the ADM0 register), and the trigger standby state (ADCS bit = 0) 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 to ADM2 registers.

The following shows the relationship between the operation mode and trigger mode.

Table 12-2. Relationship Between Operation Mode and Trigger Mode

Trigger Mode	Operation Mode		Set Value			Analog Input
			ADM0	ADM1	ADM2	
A/D trigger	Select	1 buffer	xx010000B	00000xxxB	00000xxxB	ANI0 to ANI7
		4 buffers	xx110000B		00000xxxB	ANI0 to ANI7
	Scan		xxx00000B		00000xxxB	ANI0 to ANI7
Timer trigger	Select	1 buffer	xx010000B	00001xxxB	00000xxxB	ANI0 to ANI7
		4 buffers	xx110000B		00000xxxB	ANI0 to ANI7
	Scan		xxx00000B		00000xxxB	ANI0 to ANI7
External trigger	Select	1 buffer	xx010000B	xx10xxxxB	00000xxxB	ANI0 to ANI7
		4 buffers	xx110000B		00000xxxB	ANI0 to ANI7
	Scan		xxx00000B		00000xxxB	ANI0 to ANI7

(1) Trigger mode

There are three types of trigger modes that serve as the start timing of A/D conversion processing: A/D trigger mode, timer trigger mode, and external trigger mode. These trigger modes are set by the TRG1 and TRG0 bits of the ADM1 register.

(a) A/D trigger mode

This mode starts the conversion timing of the analog input set to the ANI0 to ANI7 pins, and by setting the ADCE bit of the ADM0 register to 1, starts A/D conversion. Unless the ADCE bit is cleared to 0 after conversion, the next conversion operation is repeated. If data is written to the ADM0 to ADM2 registers during conversion, conversion is stopped and then executed from the beginning again.

(b) Timer trigger mode

This mode specifies the conversion timing of the analog input set for the ANI0 to ANI7 pins using the values set to the timer C capture/compare register.

This register creates the analog input conversion timing by generating the compare match interrupts of the four capture/compare registers (CCC40, CCC41, CCC50, CCC51) connected to the 16-bit timer C (TMC4, TMC5).

If the ADCE bit of the ADM0 register is set to 1, the A/D converter waits for an interrupt (INTCCC40, INTCCC41, INTCCC50, or INTCCC51), and starts conversion when INTCCC40, INTCCC41, INTCCC50, or INTCCC51 occurs (ADCS bit = 1 in the ADM0 register). When conversion has finished, the converter waits for an interrupt again (ADCS = 0). If data is written to the ADM0 to ADM2 registers during conversion, conversion is stopped and then executed from the beginning again.

(c) External trigger mode

This mode specifies the conversion timing of the analog input to the ANI0 to ANI7 pins using the ADTRG pin.

The EGA1 and EGA0 bits of the ADM1 register are used to specify the valid edge to be input to the ADTRG pin.

When the ADCE bit of the ADM0 register is set to 1, the A/D converter waits for an external trigger (ADTRG), and starts conversion when the valid edge of ADTRG is detected (ADCS bit = 1 in the ADM0 register). When the converter has completed its conversion operation, it waits for an external trigger again (ADCS bit = 0).

If the valid edge is detected at the ADTRG pin during conversion, conversion is executed from the beginning again.

If data is written to the ADM0 to ADM2 registers during conversion, conversion is stopped and then executed from the beginning again.

(2) Operation mode

There are two operation modes that set the ANI0 to ANI7 pins: select mode and scan mode. The select mode has sub-modes that consist of 1-buffer mode and 4-buffer mode. These modes are set by the BS and MS bits of the ADM0 register.

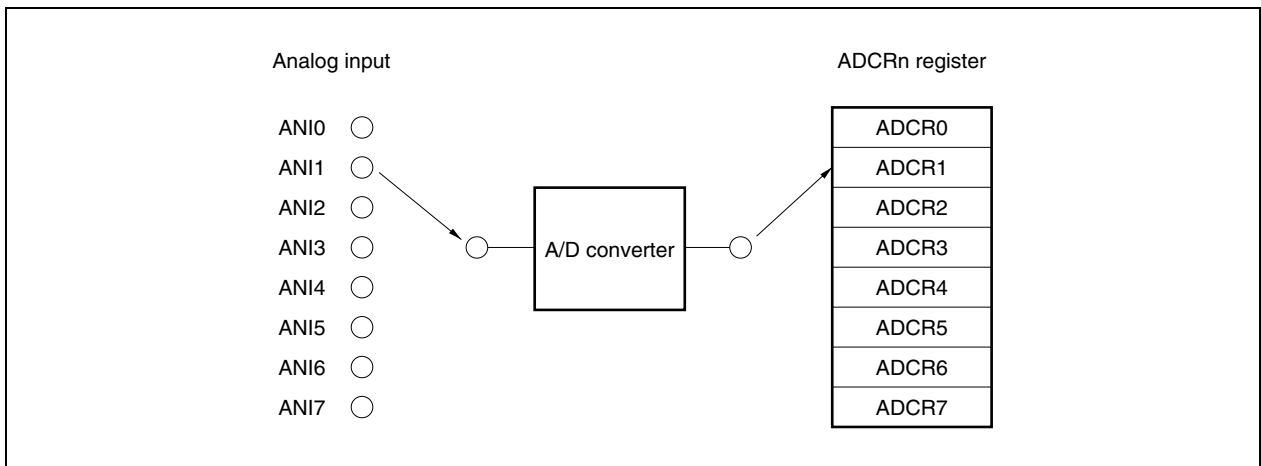
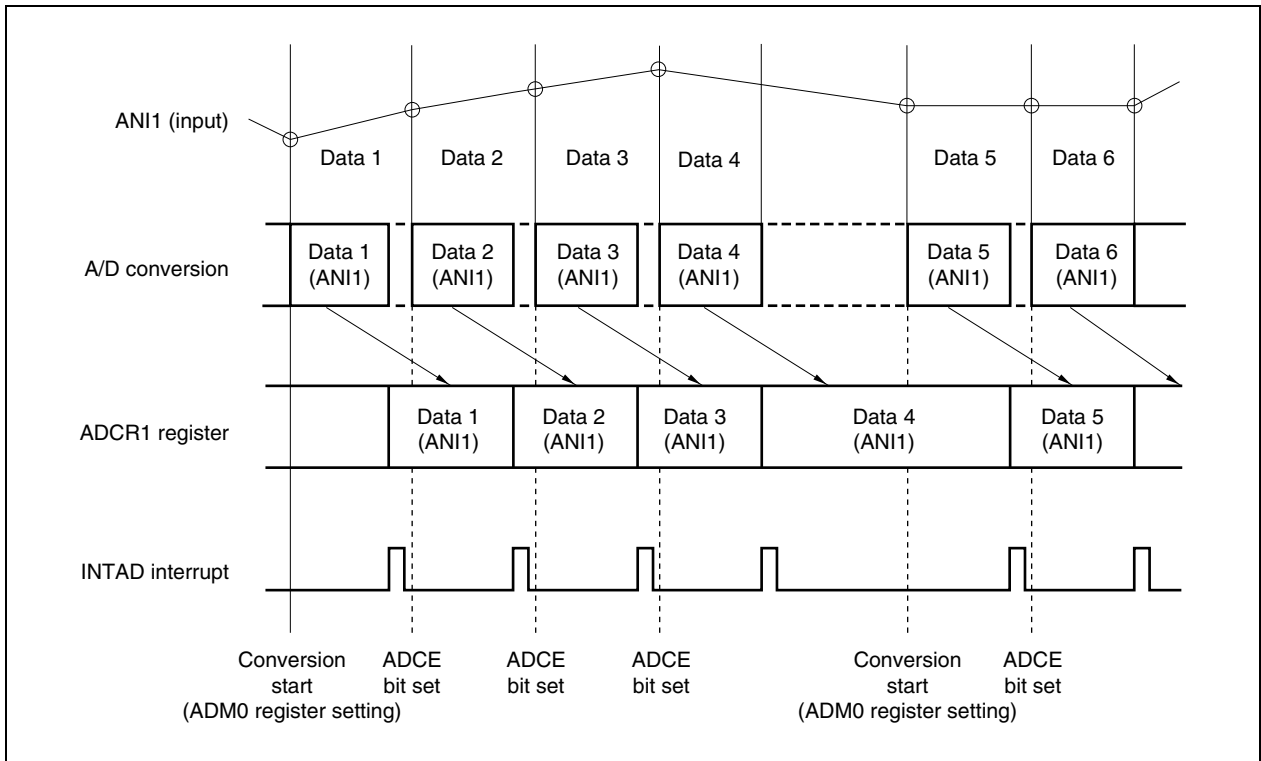
(a) Select mode

In this mode, one analog input specified by the ADM2 register is A/D converted. The conversion results are stored in the ADCR_n register corresponding to the analog input (ANI_n). For this mode, the 1-buffer mode and 4-buffer mode are provided for storing the A/D conversion results (n = 0 to 7).

- **1-buffer mode**

In this mode, one analog input specified by the ADM2 register is A/D converted. The conversion results are stored in the ADCR_n register corresponding to the analog input (ANI_n) (n = 0 to 7). The ANI_n and ADCR_n register correspond one to one, and an A/D conversion end interrupt (INTAD) is generated each time one A/D conversion ends. After conversion has finished, the next conversion operation is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

Figure 12-3. Select Mode Operation Timing: 1-Buffer Mode (ANI1)

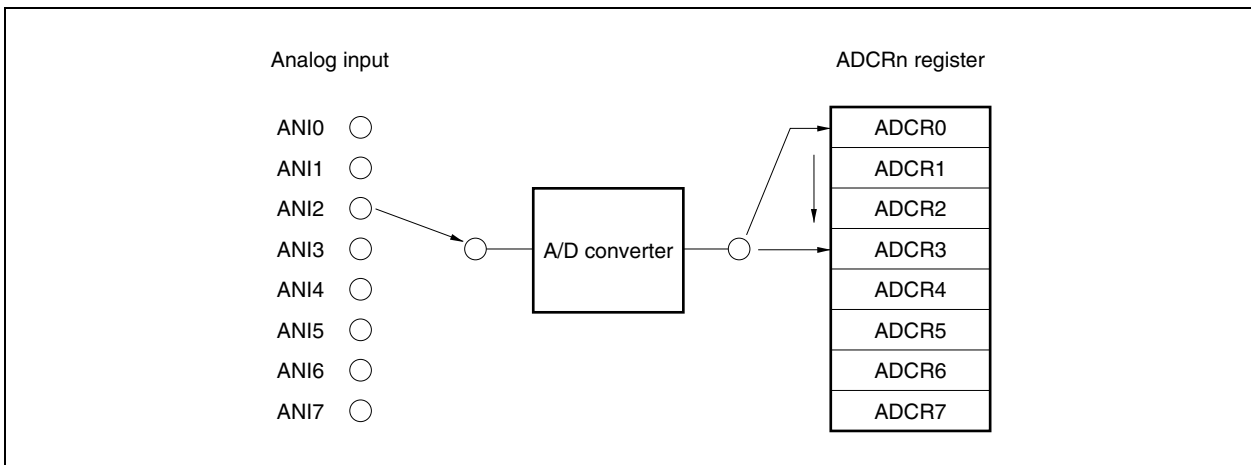
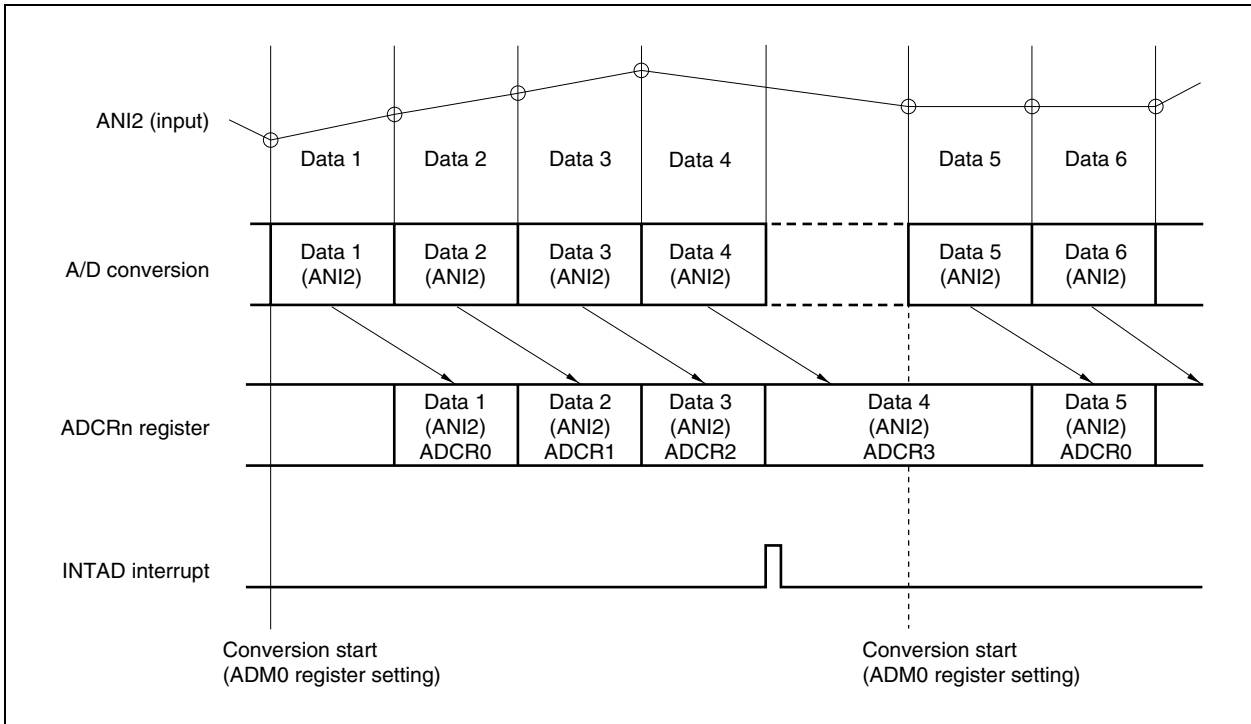


• **4-buffer mode**

In this mode, one analog input is A/D converted four times and the results are stored in the ADCRn register. The A/D conversion end interrupt (INTAD) is generated when the four A/D conversions end (n = 0 to 3 when the lower analog input channel is specified and n = 4 to 7 when the higher analog input channel is specified).

After conversion has finished, the next conversion operation is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

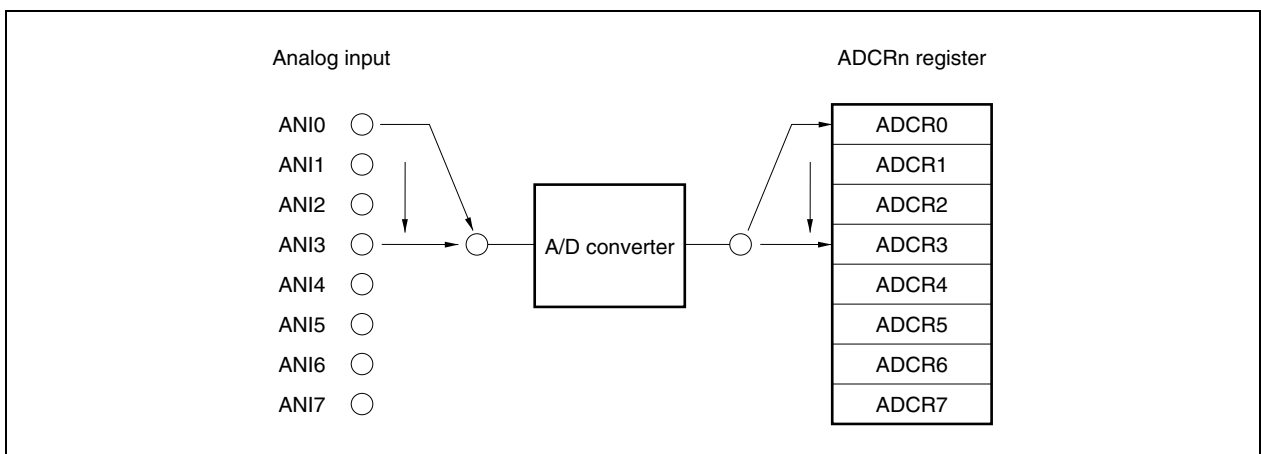
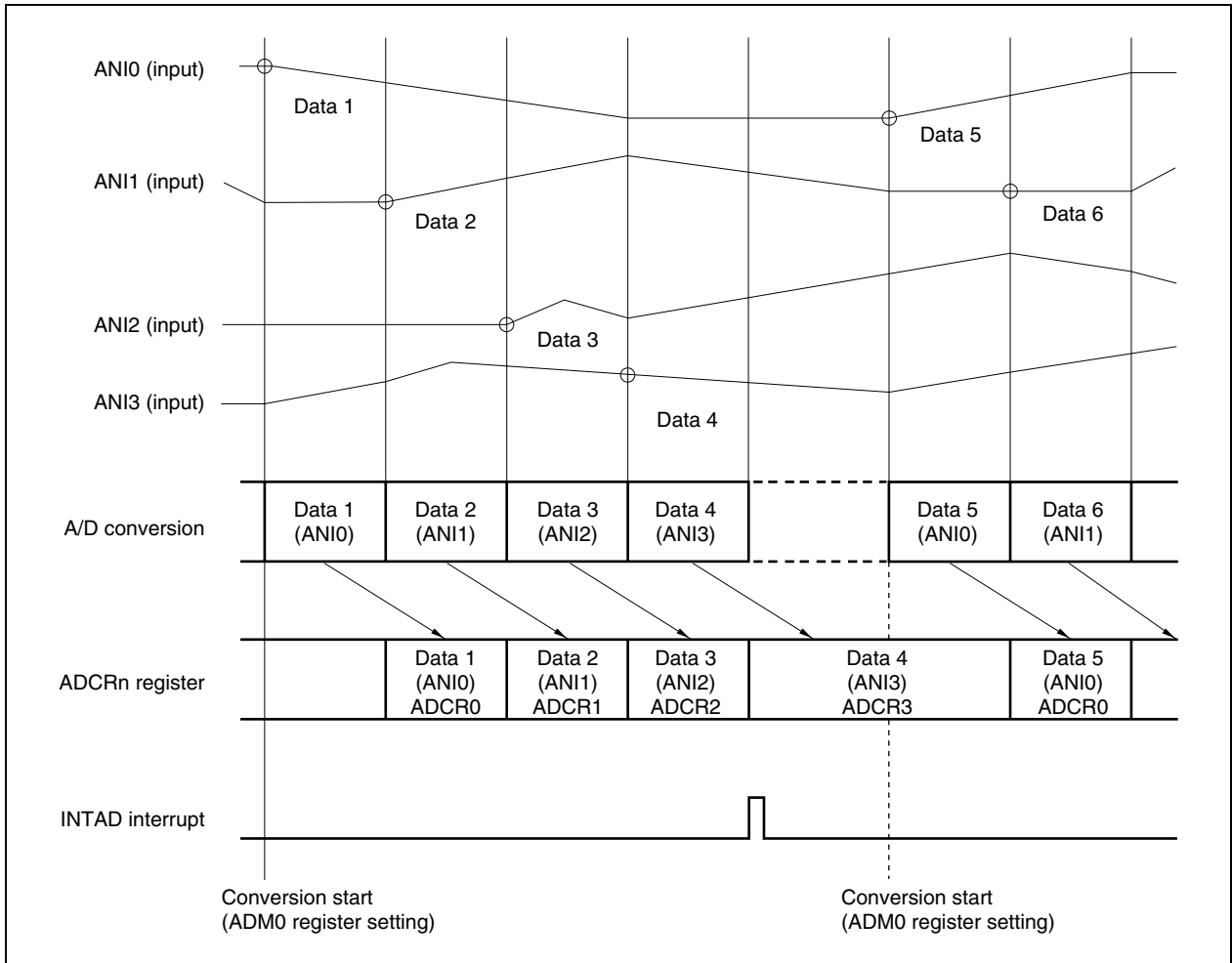
Figure 12-4. Select Mode Operation Timing: 4-Buffer Mode (ANI2)



(b) Scan mode

In this mode, the analog inputs specified by the ADM2 register are selected sequentially from the ANIO 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 7). When the conversion of the specified analog input ends, the A/D conversion end interrupt (INTAD) is generated. After conversion has finished, the next conversion operation is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

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 ADM2 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 7).

(1) 1-buffer mode (A/D trigger select: 1 buffer)

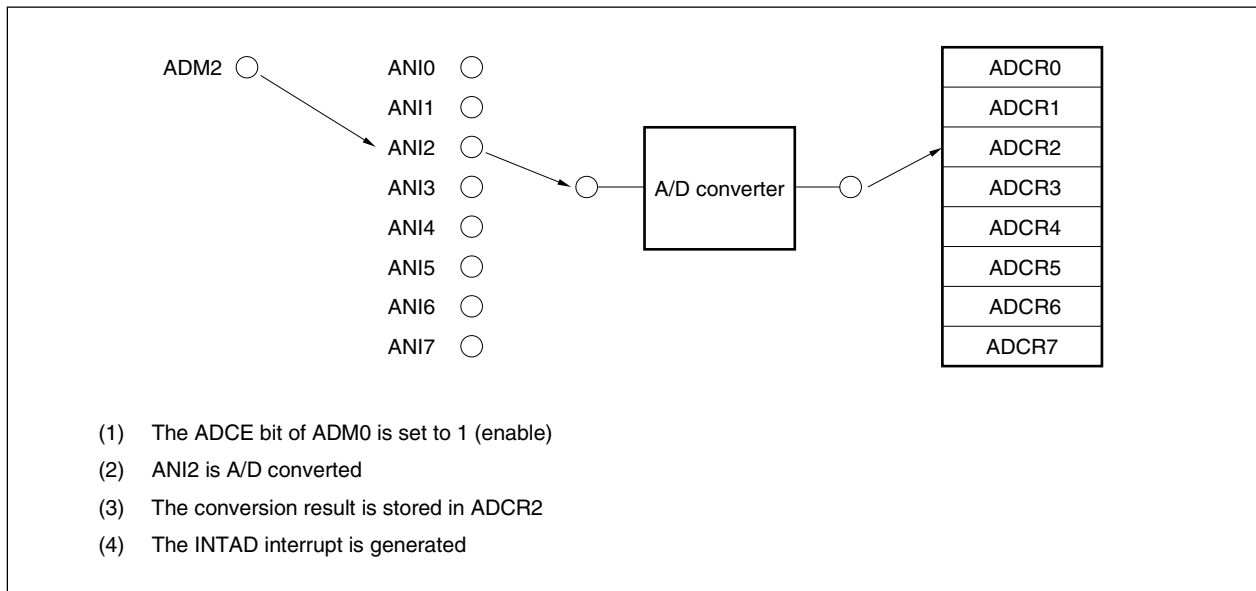
In this mode, one analog input is A/D converted once. The conversion results are stored in one ADCRn register. The analog input and ADCRn register correspond one to one.

Each time an A/D conversion is executed, an A/D conversion end interrupt (INTAD) is generated and A/D conversion ends. The next conversion operation is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

Analog Input	A/D Conversion Result Register
ANIn	ADCRn

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 Operation (A/D Trigger Select: 1 Buffer)



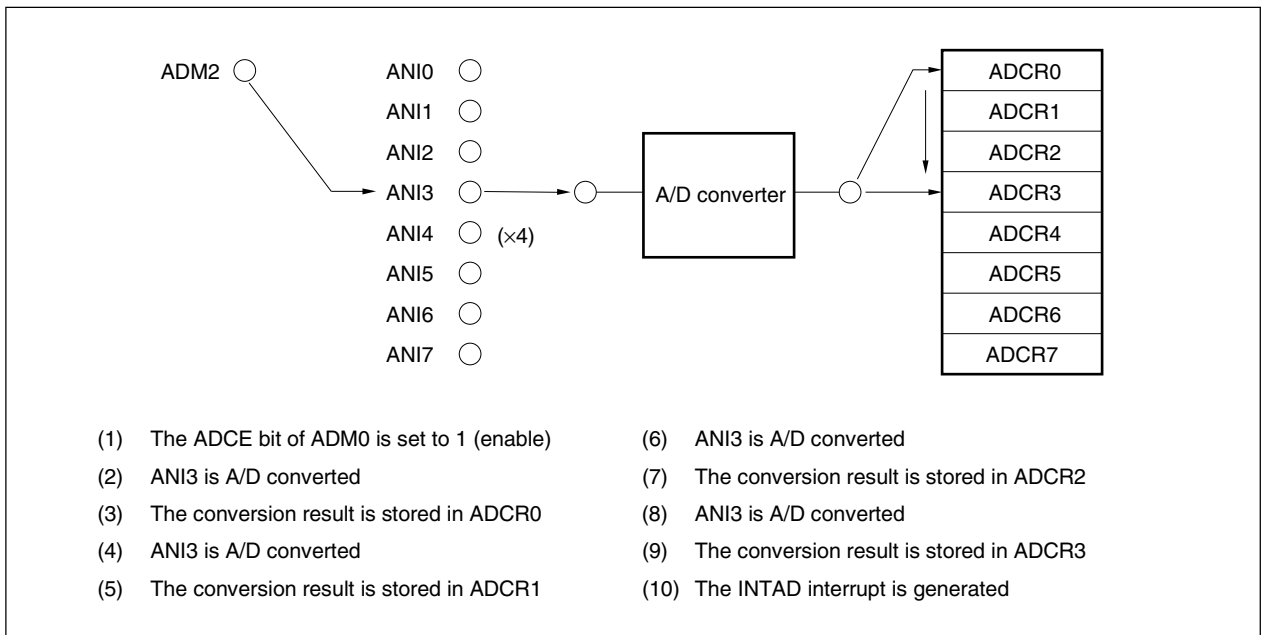
(2) 4-buffer mode (A/D trigger select: 4 buffers)

In this mode, one analog input is A/D converted four times and the results are stored in the ADCRn register. When the 4th A/D conversion ends, an A/D conversion end interrupt (INTAD) is generated and the A/D conversion is stopped. The next conversion operation is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

Analog Input	A/D Conversion Result Register
ANI0 to ANI3	ADCR0 (1st time)
	ADCR1 (2nd time)
	ADCR2 (3rd time)
	ADCR3 (4th time)
ANI4 to ANI7	ADCR4 (1st time)
	ADCR5 (2nd time)
	ADCR6 (3rd time)
	ADCR7 (4th time)

This mode is suitable for applications in which the average of the A/D conversion results is calculated.

Figure 12-7. Example of 4-Buffer Mode Operation (A/D Trigger Select: 4 Buffers)



12.5.2 Scan mode operations

In this mode, the analog inputs specified by the ADM2 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 7).

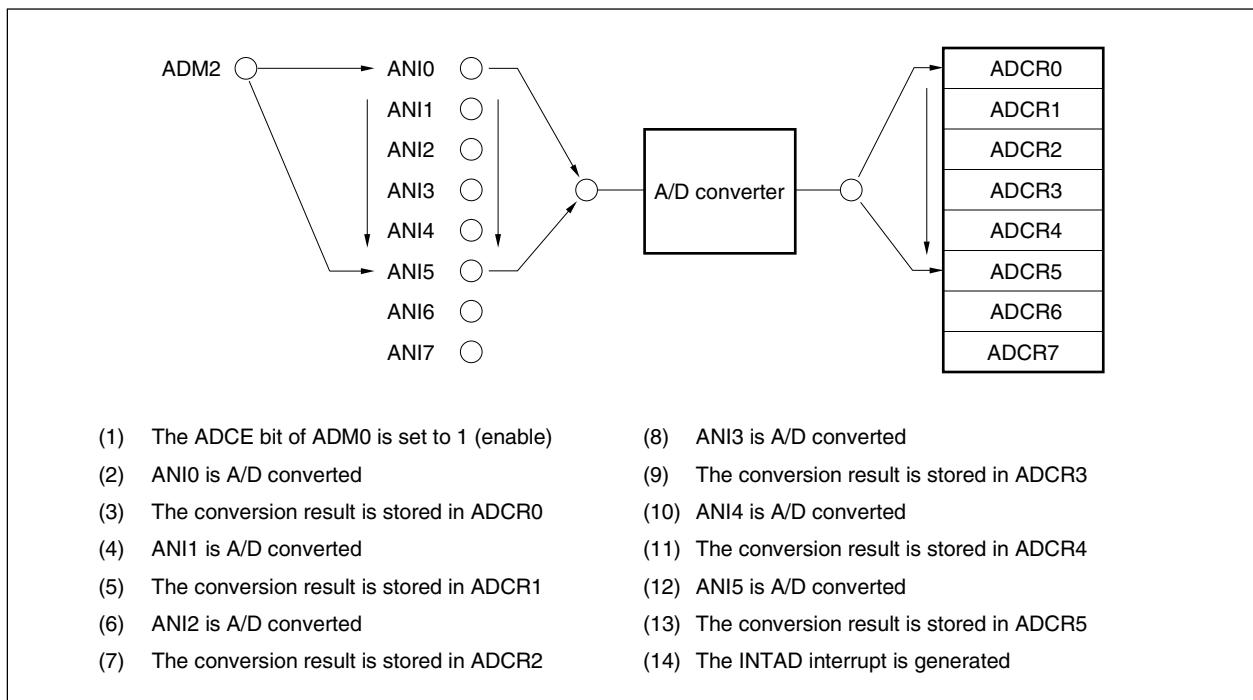
When conversion of all the specified analog input ends, the A/D conversion end interrupt (INTAD) is generated, and A/D conversion is stopped. The next conversion operation is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

Analog Input	A/D Conversion Result Register
ANI0	ADCR0
⋮	⋮
ANIn ^{Note}	ADCRn

Note Set by the ANIS2 to ANIS0 bits of the ADM2 register.

This mode is most appropriate for applications in which multiple analog inputs are constantly monitored.

Figure 12-8. Example of Scan Mode Operation (A/D Trigger Scan)



12.6 Operation in Timer Trigger Mode

In this mode, the conversion timing of the analog input signal set by the ANI0 to ANI7 pins is defined by the value set to the capture/compare registers of timers C4 and C5.

The analog input conversion timing is generated when a compare match interrupt (INTCCC40, INTCCC41, INTCCC50, or INTCCC51) is generated by the capture/compare register connected to 16-bit timer C4 or C5 (TMC4 or TMC5).

When the ADCE bit of the ADM0 register is set to 1, the A/D converter waits for the interrupt (INTCCC40, INTCCC41, INTCCC50, or INTCCC51), and starts conversion when INTCCC40, INTCCC41, INTCCC50, or INTCCC51 occurs (ADCS bit = 1 in the ADM0 register). When conversion is finished, the converter waits for an interrupt again (ADCS bit = 0).

If INTCCC40, INTCCC41, INTCCC50, or INTCCC51 occurs during conversion, the conversion operation is executed from the beginning again.

If data is written to the ADM0 to ADM2 registers during conversion, the conversion operation is stopped and executed from the beginning again.

12.6.1 Select mode operation

In this mode, an analog input (ANI0 to ANI7) specified by the ADM2 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.

(1) 1-buffer mode (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.

One analog input is A/D converted once using the trigger of the match interrupt signals (INTCCC40, INTCCC41, INTCCC50, INTCCC51) and the results are stored in one ADCRn register. An A/D conversion end interrupt (INTAD) is generated for each A/D conversion.

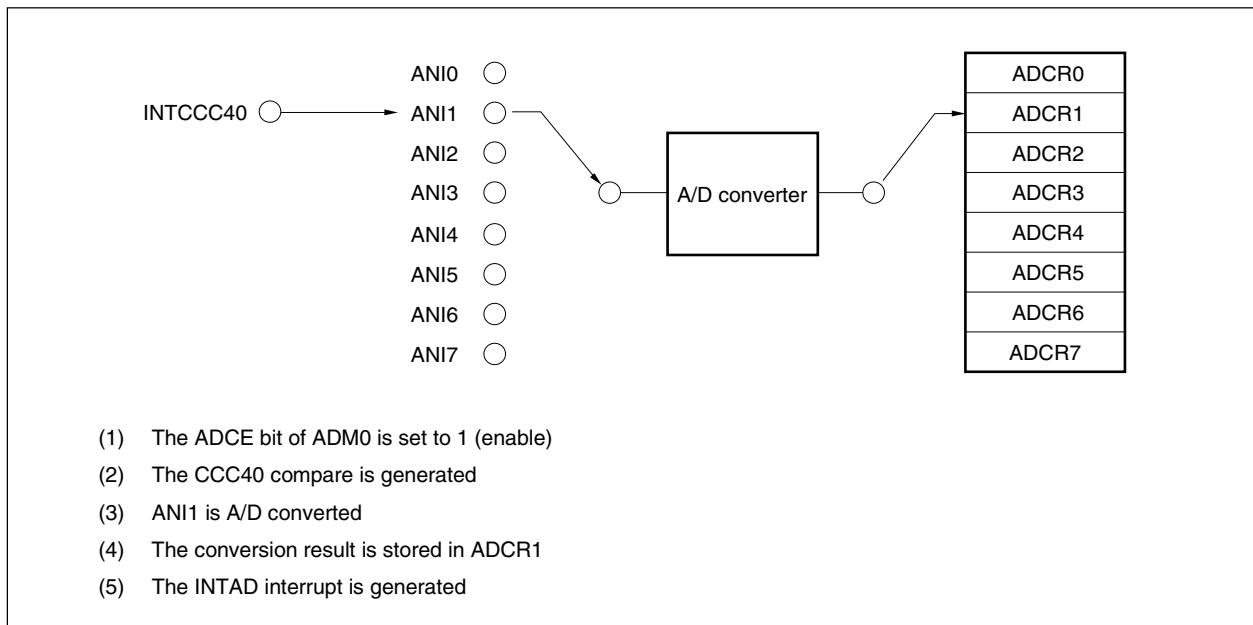
Unless the ADCE bit of the ADM0 register is cleared to 0, A/D conversion is repeated each time a timer match interrupt is generated.

**Table 12-3. Correspondence Between Analog Input Pins and ADCRn Register
(1-Buffer Mode (Timer Trigger Select: 1 Buffer))**

Trigger	Analog Input	A/D Conversion Result Register
INTCCCn interrupt	ANI0	ADCR0
INTCCCn interrupt	ANI1	ADCR1
INTCCCn interrupt	ANI2	ADCR2
INTCCCn interrupt	ANI3	ADCR3
INTCCCn interrupt	ANI4	ADCR4
INTCCCn interrupt	ANI5	ADCR5
INTCCCn interrupt	ANI6	ADCR6
INTCCCn interrupt	ANI7	ADCR7

Remark n = 40, 41, 50, 51

Figure 12-9. Example of 1-Buffer Mode Operation (Timer Trigger Select: 1 Buffer) (ANI1)



(2) 4-buffer mode (timer trigger select: 4 buffers)

In this mode, A/D conversion of one analog input is executed four times, and the results are stored in the ADCRn register.

One analog input is A/D converted four times using the match interrupt signals (INTCCC40, INTCCC41, INTCCC50, INTCCC51) as a trigger, and the results are stored in four ADCRn registers. The A/D conversion end interrupt (INTAD) is generated when the four A/D conversions end.

After conversion has finished, the next conversion is repeated, unless the ADCE bit of the ADM0 register is cleared to 0.

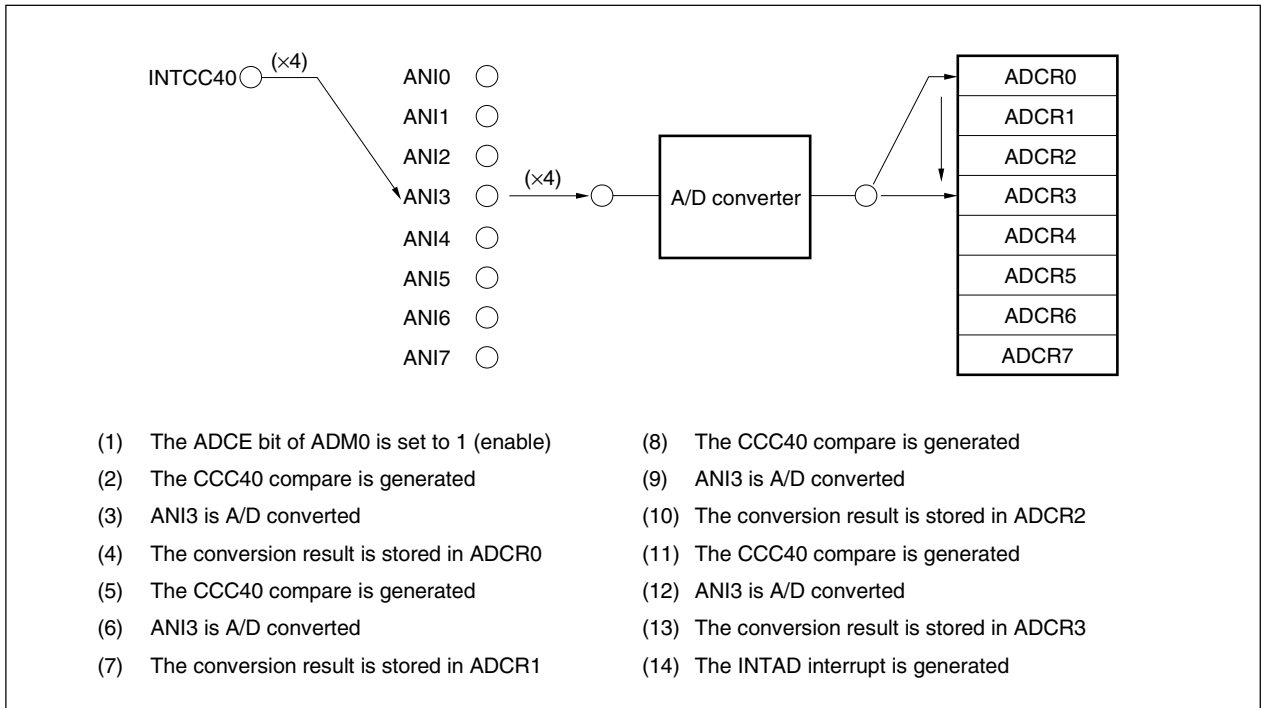
This mode is suitable for applications in which the average of the A/D conversion results is calculated.

**Table 12-4. Correspondence Between Analog Input Pins and ADCRn Register
(4-Buffer Mode (Timer Trigger Select: 4 Buffers))**

Trigger	Analog Input	A/D Conversion Result Register
INTCCn interrupt	ANI0 to ANI3	ADCR0 (1st time) ADCR1 (2nd time) ADCR2 (3rd time) ADCR3 (4th time)
	ANI4 to ANI7	ADCR4 (1st time) ADCR5 (2nd time) ADCR6 (3rd time) ADCR7 (4th time)

Remark n = 40, 41, 50, 51

Figure 12-10. Example of 4-Buffer Mode Operation (Timer Trigger Select: 4 Buffers) (ANI3)



12.6.2 Scan mode operation

In this mode, the analog inputs specified by the ADM2 register are selected sequentially from the ANI0 pin and are A/D converted the specified number of times using the timer match interrupt signal as a trigger.

The result of conversion is stored in the ADCRn register corresponding to the analog input. When all the specified analog input signals have been converted, an A/D conversion end interrupt (INTAD) occurs.

After conversion has finished, the A/D converter waits for a trigger unless the ADCE bit of the ADM0 register is cleared to 0. When a timer match interrupt occurs again, the converter starts A/D conversion again, starting from the ANI0 input.

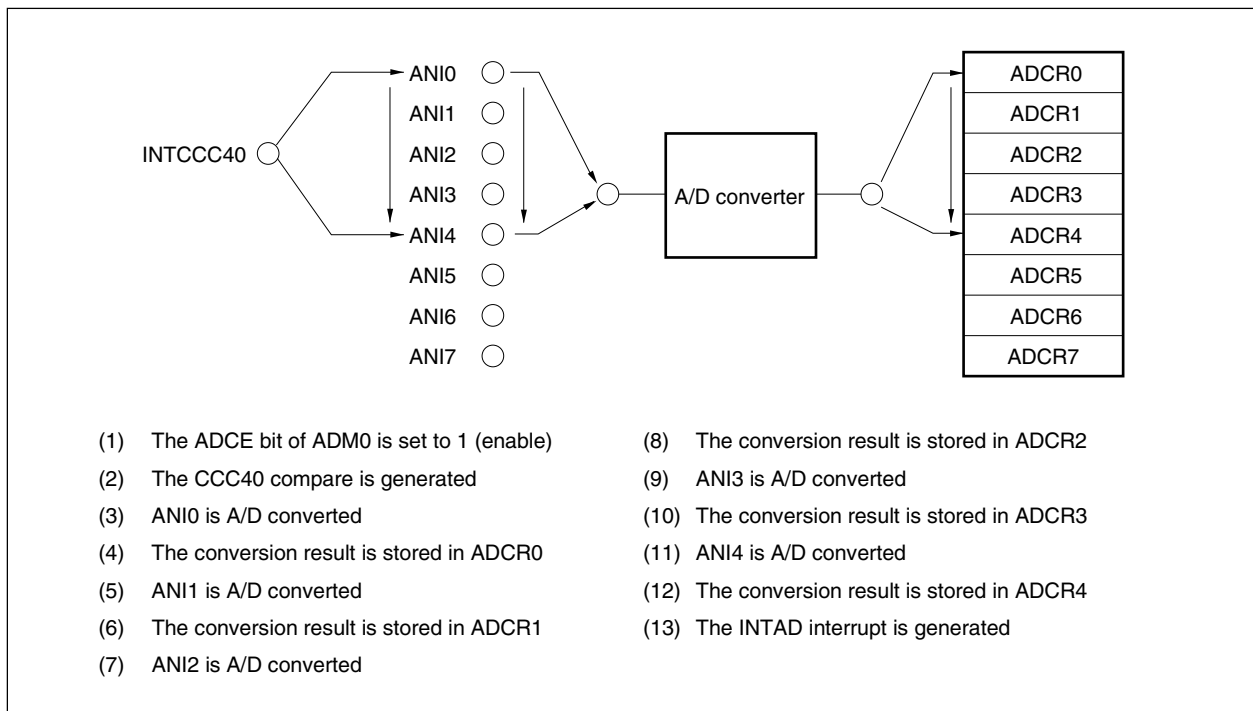
This mode is most appropriate for applications in which multiple analog inputs are constantly monitored.

Table 12-5. Correspondence Between Analog Input Pins and ADCRn Register (Scan Mode (Timer Trigger Scan))

Trigger	Analog Input	A/D Conversion Result Register
INTCCn interrupt	ANI0	ADCR0
	ANI1	ADCR1
	ANI2	ADCR2
	ANI3	ADCR3
	ANI4	ADCR4
	ANI5	ADCR5
	ANI6	ADCR6
	ANI7	ADCR7

Remark n = 40, 41, 50, 51

Figure 12-11. Example of Scan Mode Operation (Timer Trigger Scan) (ANI0 to ANI4)



12.7 Operation in External Trigger Mode

In this mode, the conversion timing of the analog signals input to the ANI0 to ANI7 pins is specified by the ADTRG pin.

Detection of the valid edge at the ADTRG input pin is specified by using the EGA1 and EGA0 bits of the ADM1 register.

When the ADCE bit of the ADM0 register is set to 1, the A/D converter waits for an external trigger (ADTRG), and starts conversion when the valid edge of ADTRG is detected (ADCS bit = 1 in the ADM0 register). When the converter has ended conversion, it waits for the external trigger again (ADCS bit = 0).

If the valid edge is detected at the ADTRG pin during conversion, conversion is executed from the beginning again.

If data is written to the ADM0 to ADM2 registers during conversion, conversion is stopped and executed from the beginning again.

12.7.1 Select mode operations

In this mode, one analog input (ANI0 to ANI7) specified by the ADM2 register is A/D converted. The conversion results are stored in the ADCRn register corresponding to the analog input. In the select mode, there are two select modes: 1-buffer mode and 4-buffer mode, according to the storing method of the conversion results.

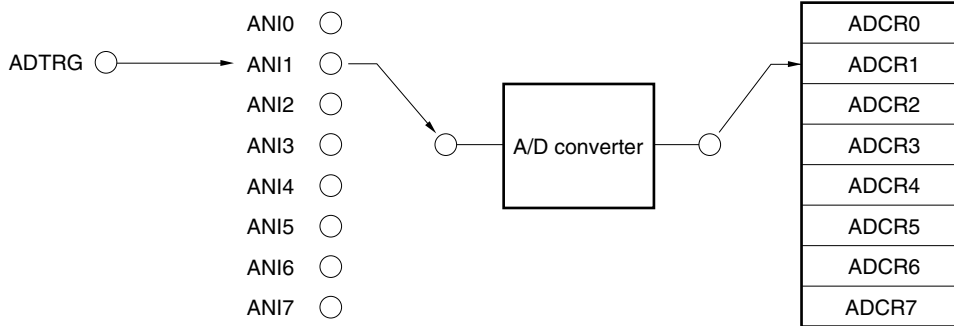
(1) 1-buffer mode (external trigger select: 1-buffer)

In this mode, one analog input is A/D converted using the ADTRG signal as a trigger. The conversion results are stored in one ADCRn register. The analog input and the A/D conversion results register correspond one to one. The A/D conversion end interrupt (INTAD) is generated for each A/D conversion, and A/D conversion is stopped.

Trigger	Analog Input	A/D Conversion Result Register
ADTRG signal	ANIn	ADCRn

While the ADCE bit of the ADM0 register is 1, A/D conversion is repeated every time a trigger is input from the ADTRG pin.

This mode is most appropriate for applications in which the results are read after each A/D conversion.

Figure 12-12. Example of 1-Buffer Mode Operation (External Trigger Select: 1 Buffer) (ANI1)

- (1) The ADCE bit of ADM0 is set to 1 (enable)
- (2) The external trigger is generated
- (3) ANI1 is A/D converted
- (4) The conversion result is stored in ADCR1
- (5) The INTAD interrupt is generated

(2) 4-buffer mode (external trigger select: 4 buffers)

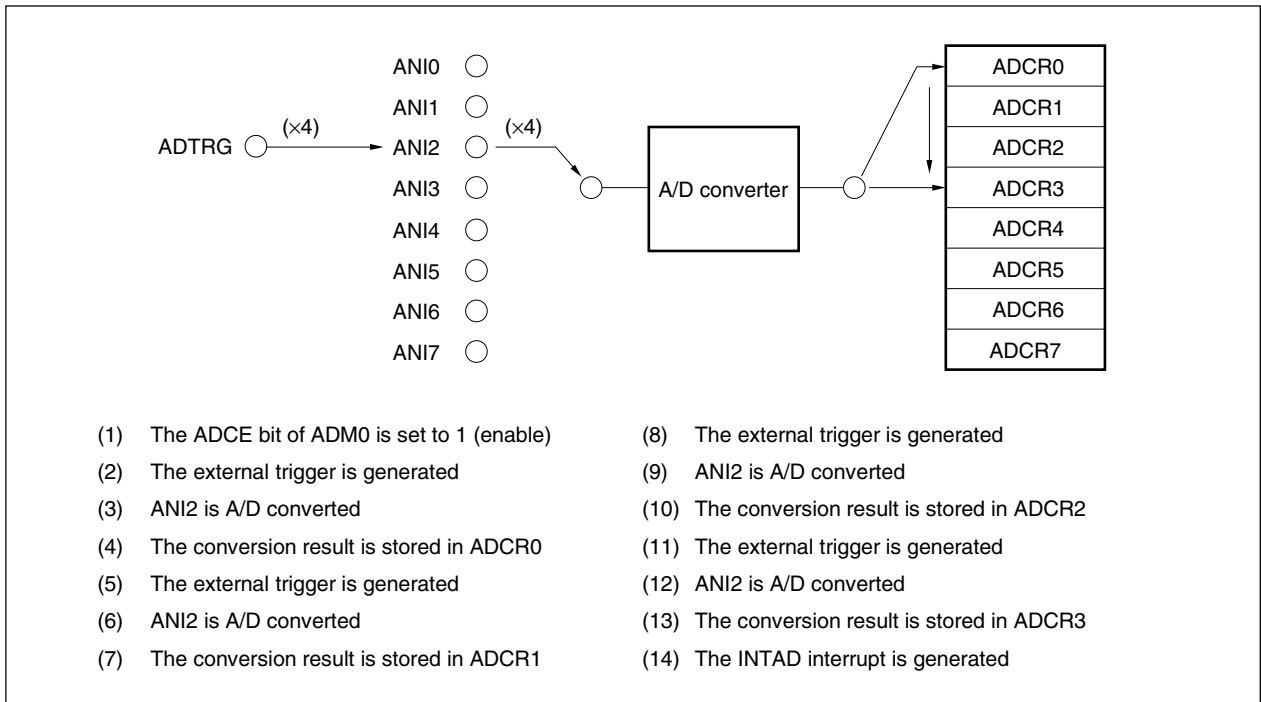
In this mode, one analog input is A/D converted four times using the ADTRG signal as a trigger and the results are stored in the ADCRn register. The A/D conversion end interrupt (INTAD) is generated and A/D conversion is stopped after the 4th A/D conversion.

Trigger	Analog Input	A/D Conversion Result Register
ADTRG signal	ANI0 to ANI3	ADCR0 (1st time)
		ADCR1 (2nd time)
		ADCR2 (3rd time)
		ADCR3 (4th time)
	ANI4 to ANI7	ADCR4 (1st time)
		ADCR5 (2nd time)
		ADCR6 (3rd time)
		ADCR7 (4th time)

While the ADCE bit of the ADM0 register is 1, A/D conversion is repeated every time a trigger is input from the ADTRG pin.

This mode is suitable for applications in which the average of the A/D conversion results is calculated.

Figure 12-13. Example of 4-Buffer Mode Operation (External Trigger Select: 4 Buffers) (ANI2)



12.7.2 Scan mode operation

In this mode, the analog inputs specified by the ADM2 register are selected sequentially from the ANI0 pin using the ADTRG signal as a trigger, and A/D converted. The A/D conversion results are stored in the ADCRn register corresponding to the analog input (n = 0 to 7).

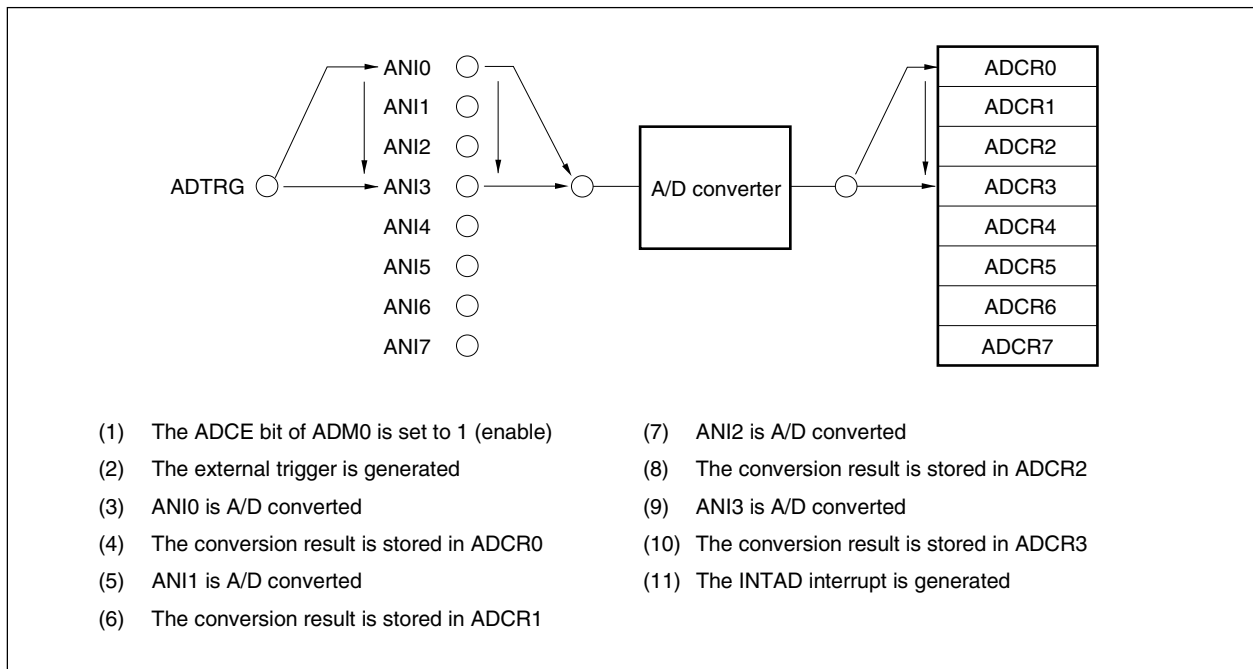
When conversion of all the specified analog inputs has ended, the A/D conversion end interrupt (INTAD) is generated. Unless the ADCE bit of the ADM0 register is cleared to 0 after end of conversion, the A/D converter waits for a trigger. The converter starts A/D conversion from the ANI0 input when a trigger is input to the ADTRG pin again.

Trigger	Analog Input	A/D Conversion Result Register
ADTRG signal	ANI0	ADCR0
	ANI1	ADCR1
	ANI2	ADCR2
	ANI3	ADCR3
	ANI4	ADCR4
	ANI5	ADCR5
	ANI6	ADCR6
	ANI7	ADCR7

When a trigger is input to the ADTRG pin while the ADCE bit of the ADM0 register is 1, A/D conversion is started again.

This is most appropriate for applications in which multiple analog inputs are constantly monitored.

Figure 12-14. Example of Scan Mode Operation (External Trigger Scan) (ANI0 to ANI3)



12.8 Notes on Operation

(1) Stopping conversion operation

When the ADCE bit of the ADM0 register is cleared to 0 during a conversion operation, the conversion operation stops and the conversion results are not stored in the ADCRn register ($n = 0$ to 7).

(2) External/timer trigger interval

Set the interval (input time interval) of the trigger in the external or timer trigger mode longer than the conversion time specified by the FR3 to FR0 bits of the ADM1 register.

When $0 < \text{interval} \leq \text{conversion operation time}$

When the following external trigger or timer trigger is input during a conversion operation, the conversion operation is aborted and the conversion starts according to the last external trigger input or timer trigger input.

When conversion operations are aborted, the conversion results are not stored in the ADCRn register ($n = 0$ to 7). However, the number of times the trigger has been input is counted. When an interrupt occurs, the values that have been converted are stored in the ADCRn register.

(3) Operation in standby mode

<1> HALT mode

In this mode, A/D conversion continues. When this mode is released by NMI input or unmasked maskable interrupt input (see **8.6.3 (2) Release of HALT mode**), the ADM0, ADM1, and ADM2 registers and ADCRn register hold the value ($n = 0$ to 7).

<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 by NMI input, the ADM0, ADM1, and ADM2 registers and the ADCRn register hold the value ($n = 0$ to 7). 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 by NMI input or unmasked maskable interrupt input (see **8.6.4 (2) Release of IDLE mode**, **8.6.5 (2) Release of software STOP mode**), the conversion operation resumes, but the conversion result written to the ADCRn register will become undefined.

(4) Compare match interrupt 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 using the interrupt mask bits (CCC4MK0, CCC4MK1, CCC5MK0, CCC5MK1) of the interrupt control register (CCC4IC0, CCC4IC1, CCC5IC0, CCC5IC1).

(5) Input range of ANI0 to ANI7

Use the input voltage at ANI0 to ANI7 within the specified range. If a voltage outside the range of AV_{REFP} and AV_{REFM} is input to any of these pins (even within the absolute maximum rating range), the converted value of the channel is undefined. In addition, the converted value of the other channels may also be affected.

(6) Conflict**<1> Conflict between writing A/D conversion result registers (ADCRn, ADCRnH) at end of conversion and reading ADCRn and ADCRnH registers by instruction**

Reading the ADCRn and ADCRnH registers takes precedence. After these registers have been read, the new conversion result is written to the ADCRn and ADCRnH registers.

<2> Conflict between writing ADCRn and ADCRnH at end of conversion and input of external trigger signal

The external trigger signal is not accepted during A/D conversion. Therefore, it is not accepted while ADCRn and ADCRnH are being written.

<3> Conflict between writing ADCRn and ADCRnH at end of conversion and writing A/D converter mode register 1 (ADM1) or A/D converter mode register 2 (ADM2)

If ADM1 or ADM2 is written immediately after ADCRn and ADCRnH have been written on completion of A/D conversion, the conversion result is written to the ADCRn and ADCRnH registers, but the A/D conversion end interrupt (INTAD) may not occur depending on the timing.

★ 12.9 How to Read A/D Converter Characteristics Table

Here, special terms unique to the A/D converter are explained.

(1) Resolution

This is the minimum analog input voltage that can be identified. That is, the percentage of the analog input voltage per bit of digital output is called 1LSB (Least Significant Bit). The percentage of 1LSB with respect to the full scale is expressed by %FSR (Full Scale Range). %FSR indicates the ratio of analog input voltage that can be converted as a percentage, and is always represented by the following formula regardless of the resolution.

$$\begin{aligned} 1\%FSR &= (\text{Max. value of analog input voltage that can be converted} - \text{Min. value of analog input voltage that} \\ &\quad \text{can be converted})/100 \\ &= (AV_{REFP} - AV_{REFM})/100 \end{aligned}$$

1LSB is as follows when the resolution is 10 bits.

$$\begin{aligned} 1LSB &= 1/2^{10} = 1/1024 \\ &= 0.098\%FSR \end{aligned}$$

Accuracy has no relation to resolution, but is determined by overall error.

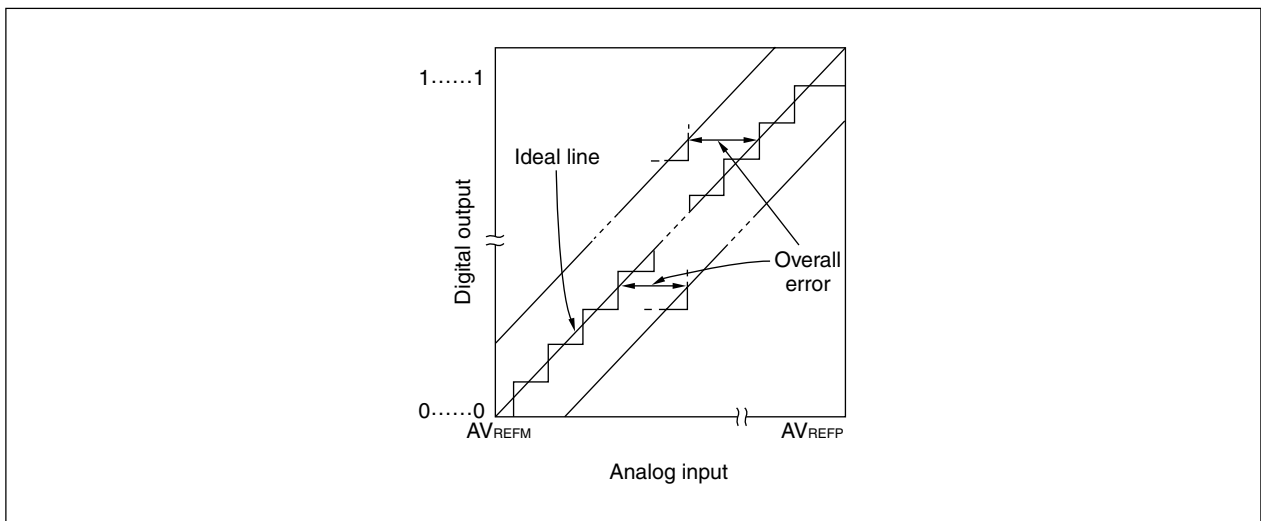
(2) Overall error

This shows the maximum error value between the actual measured value and the theoretical value.

Zero-scale error, full-scale error, linearity error and errors that are combinations of these express the overall error.

Note that the quantization error is not included in the overall error in the characteristics table.

Figure 12-15. Overall Error

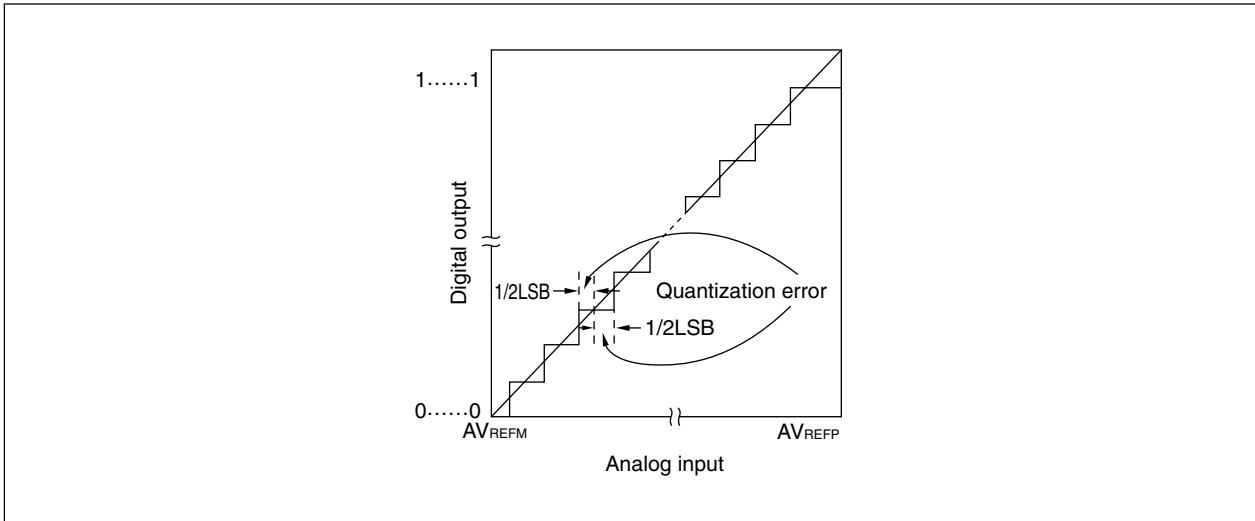


(3) Quantization error

When analog values are converted to digital values, a $\pm 1/2\text{LSB}$ error naturally occurs. In an A/D converter, an analog input voltage in a range of $\pm 1/2\text{LSB}$ is converted to the same digital code, so a quantization error cannot be avoided.

Note that the quantization error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

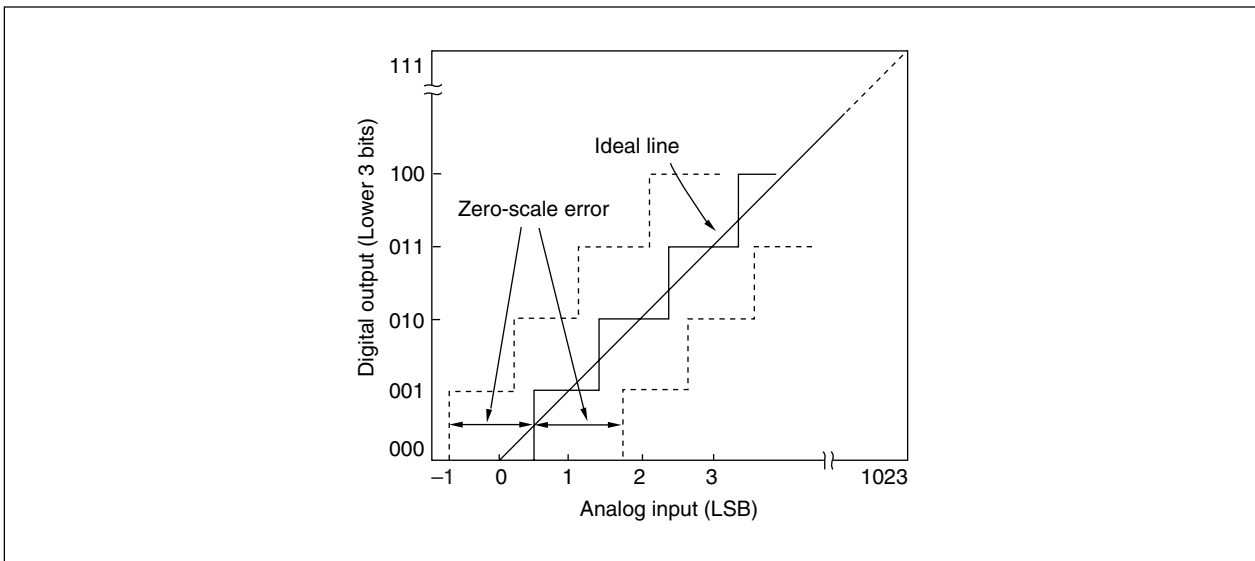
Figure 12-16. Quantization Error



(4) Zero-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value ($1/2\text{LSB}$) when the digital output changes from 0.....000 to 0.....001.

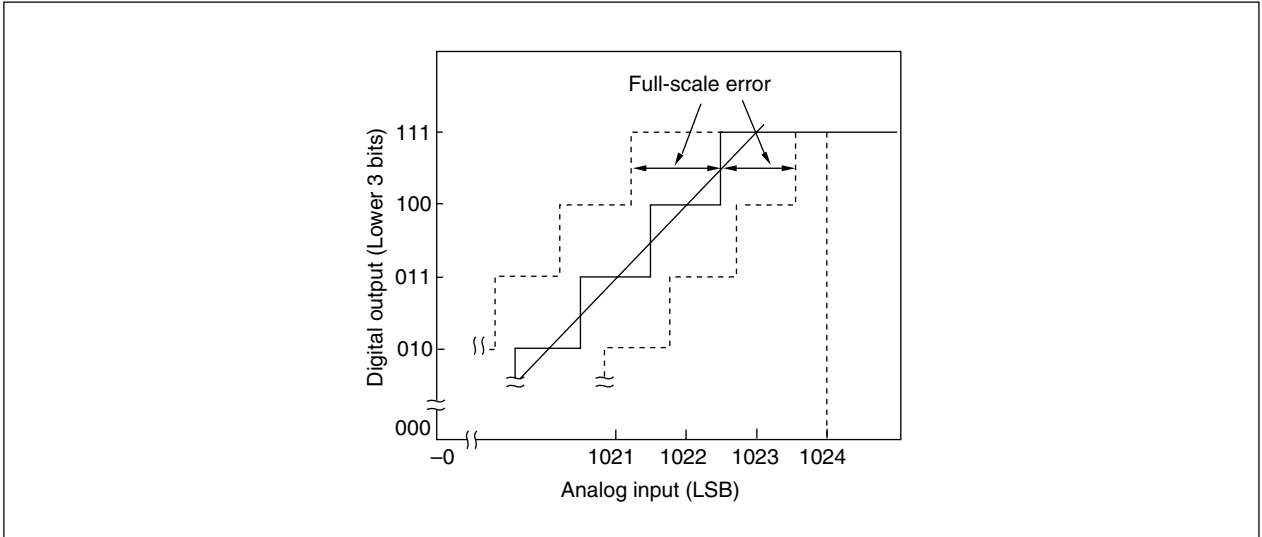
Figure 12-17. Zero-Scale Error



(5) Full-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value ($3/2\text{LSB}$) when the digital output changes from 1.....110 to 1.....111.

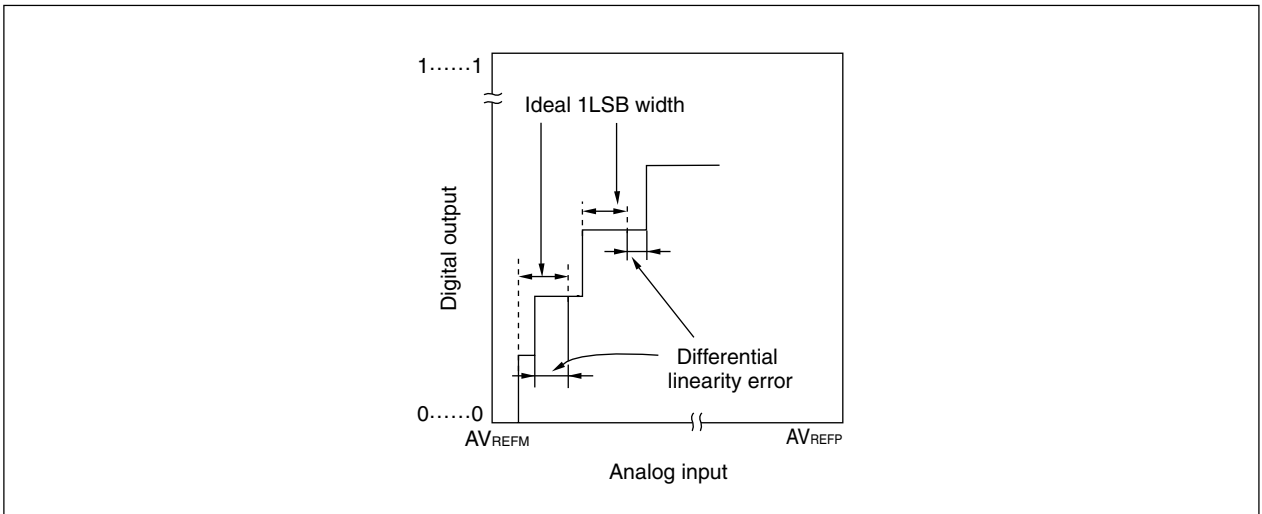
Figure 12-18. Full-Scale Error



(6) Differential linearity error

While the ideal width of code output is 1LSB, this indicates the difference between the actual measurement value and the ideal value.

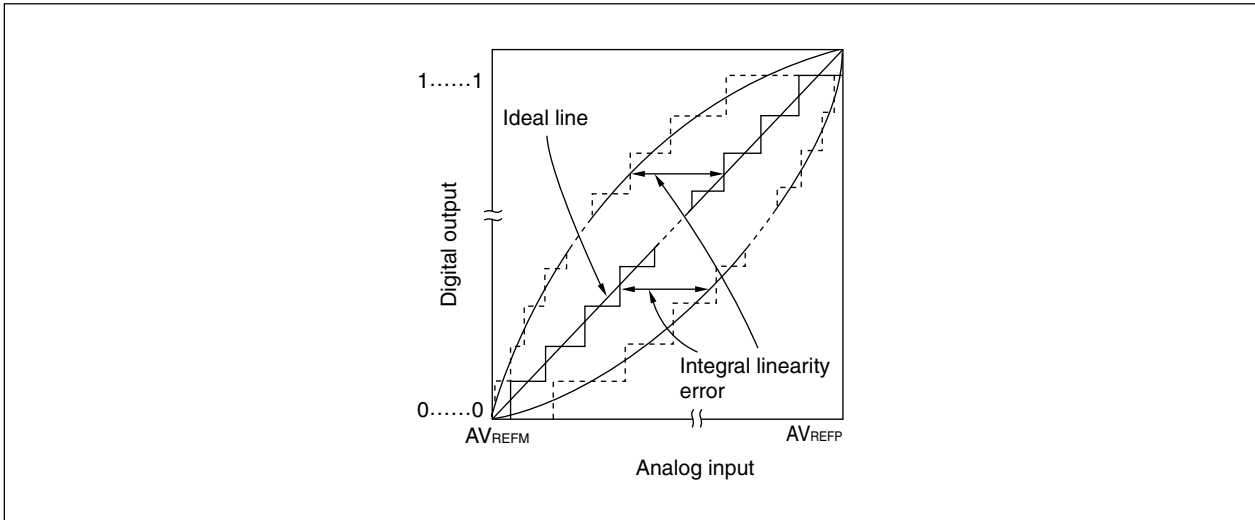
Figure 12-19. Differential Linearity Error



(7) Integral linearity error

This shows the degree to which the conversion characteristics deviate from the ideal linear relationship. It expresses the maximum value of the difference between the actual measurement value and the ideal straight line when the zero-scale error and full-scale error are 0.

Figure 12-20. Integral Linearity Error



(8) Conversion time

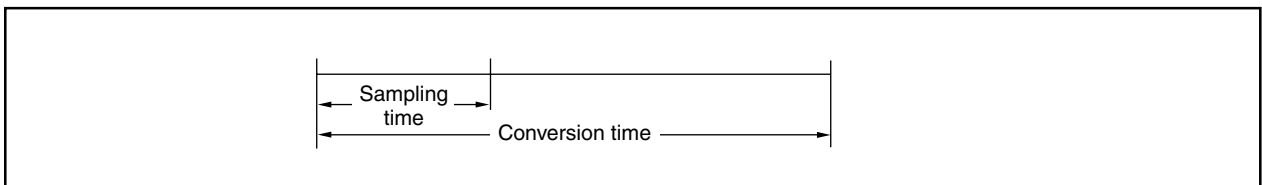
This expresses the time from when the analog input voltage was applied to the time when the digital output was obtained.

The sampling time is included in the conversion time in the characteristics table.

(9) Sampling time

This is the time the analog switch is turned on for the analog voltage to be sampled by the sample & hold circuit.

Figure 12-21. Sampling Time



CHAPTER 13 PWM UNIT

13.1 Features

- PWMn: 2 channels
- 12- to 16-bit accuracy selectable
- Minimization of low-pass filter size due to main pulse + ancillary pulse configuration
 - Main pulse 4/5/6/7/8 bits
 - Ancillary pulse 8 bits
- ★ Repeat frequency: 129 kHz to 2 MHz ($f_{\text{PWMC}} = 33 \text{ MHz}$)
- Selecting pulse width rewrite period: Every 1 pulse/256 pulses
- Active level of PWM output pulse selectable
- PWM operating clock (f_{PWMC}): $f_x/4$, $f_x/8$, $f_x/16$, or $f_x/32$ selectable

- Remarks**
1. $n = 0, 1$
 2. f_{PWMC} : PWM operating clock
 f_x : Main clock

13.2 Configuration

PWMn has a PWM output configuration of 256 main pulses and sets the active-level width using modulo H register n.

(1) Prescaler

The prescaler divides f_x to generate the PWM operating clock (f_{PWMC}). The output frequency of the prescaler can be selected from $f_x/4$, $f_x/8$, $f_x/16$, and $f_x/32$, by using the CKSPn1 and CKSPn0 bits of the PWMCn register ($n = 0, 1$).

(2) Reload controller

The reload controller controls reloading of the modulo register value.

The reload timing (PWM pulse width rewrite period) can be selected from $2^x/f_{\text{PWMC}}$ or $2^{x+8}/f_{\text{PWMC}}$ by using the SYNn bit of the PWMCn register ($n = 0, 1$, $x = 4$ to 8 (main pulse bit length)).

(3) Main pulse generator/output controller

This circuit controls the output timing of the main pulse.

It generates the main pulse from the reload signal generated by the reload controller, according to the value of modulo H register n ($n = 0, 1$).

(4) Ancillary pulse generator/output controller

This circuit controls the output timing of the ancillary pulse.

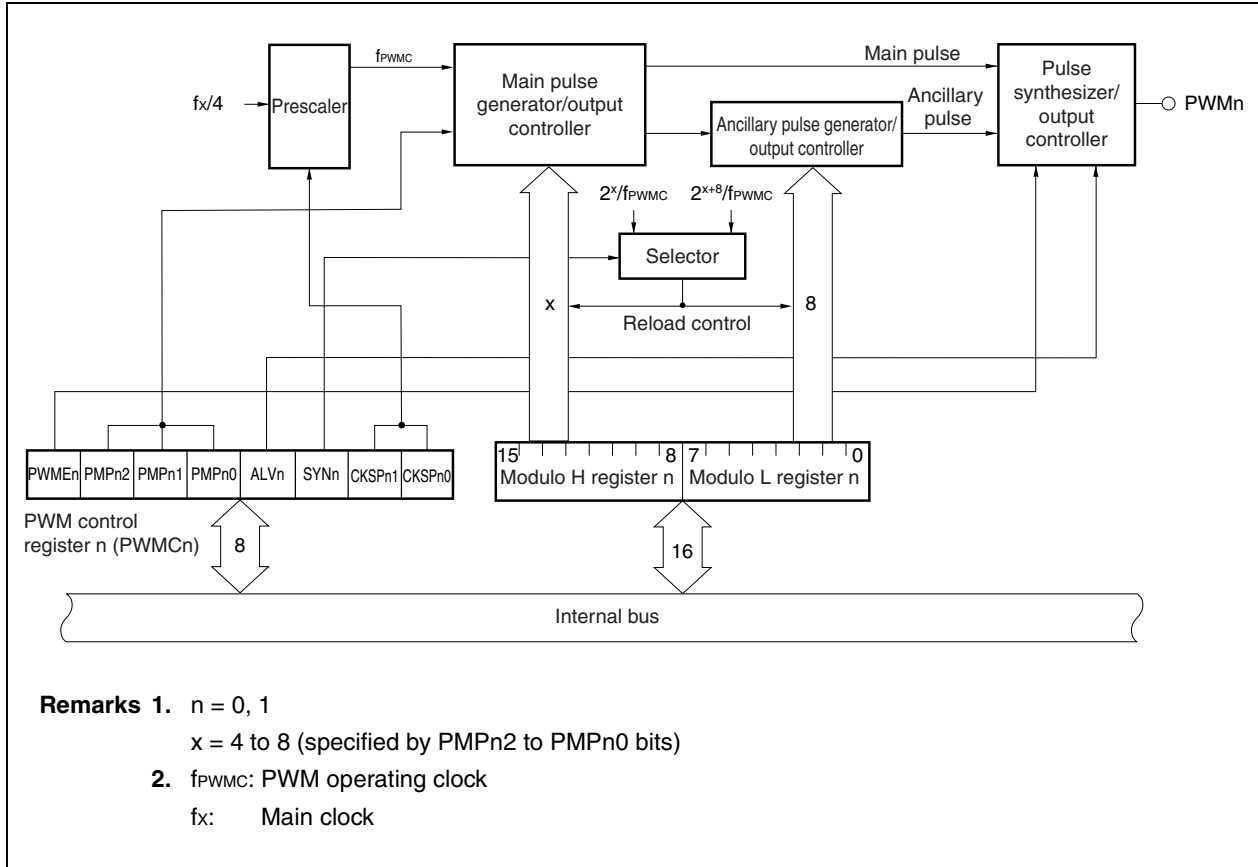
It generates an ancillary pulse from the reload signal generated by the reload controller, according to the value of modulo L register n ($n = 0, 1$).

(5) Pulse synthesizer/output controller

This circuit controls the timing of the PWM pulse signal output by synthesizing the main pulse and ancillary pulse.

★

Figure 13-1. Block Diagram of PWM Unit



13.3 Control Registers

(1) PWM control registers 0 and 1 (PWMC0 and PWMC1)

PWMCn is a register that controls PWMn (n = 0, 1).

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

- Cautions**
1. Do not change the setting of the PMPn2 to PMPn0, SYNn, CKSPn1, and CKSPn0 bits during PWM operation (when PWME bit = 1). Otherwise, the operation cannot be guaranteed.
 2. Note that, if the setting of the ALVn bit is changed during PWM operation (when PWME bit = 1), noise may be generated.

	7	6	5	4	3	2	1	0	Address	After reset
PWMC0	PWME0	PMP02	PMP01	PMP00	ALV0	SYN0	CKSP01	CKSP00	FFFFFB00H	08H
PWMC1	PWME1	PMP12	PMP11	PMP10	ALV1	SYN1	CKSP11	CKSP10	FFFFFB10H	08H

Bit position	Bit name	Function																																			
7	PWME _n	This bit enables or disables the operation of PWM _n . 0: Disables PWM operation. Outputs the inactive level as PWM output (PWM _n). 1: Enables PWM operation.																																			
6 to 4	PMP _{n2} to PMP _{n0}	<p>These bits specify the number of main pulse length specification bits and the main pulse length.</p> <table border="1"> <thead> <tr> <th>PMP_{n2}</th> <th>PMP_{n1}</th> <th>PMP_{n0}</th> <th>Number of main pulse length specification bits</th> <th>Main pulse bit length</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>8 bits</td> <td>2⁸ (256 bits)</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>7 bits</td> <td>2⁷ (128 bits)</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>6 bits</td> <td>2⁶ (64 bits)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>5 bits</td> <td>2⁵ (32 bits)</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>4 bits</td> <td>2⁴ (16 bits)</td> </tr> <tr> <td colspan="3">Other than above</td> <td colspan="2">Setting prohibited</td> </tr> </tbody> </table>	PMP _{n2}	PMP _{n1}	PMP _{n0}	Number of main pulse length specification bits	Main pulse bit length	0	0	0	8 bits	2 ⁸ (256 bits)	0	0	1	7 bits	2 ⁷ (128 bits)	0	1	0	6 bits	2 ⁶ (64 bits)	0	1	1	5 bits	2 ⁵ (32 bits)	1	0	0	4 bits	2 ⁴ (16 bits)	Other than above			Setting prohibited	
PMP _{n2}	PMP _{n1}	PMP _{n0}	Number of main pulse length specification bits	Main pulse bit length																																	
0	0	0	8 bits	2 ⁸ (256 bits)																																	
0	0	1	7 bits	2 ⁷ (128 bits)																																	
0	1	0	6 bits	2 ⁶ (64 bits)																																	
0	1	1	5 bits	2 ⁵ (32 bits)																																	
1	0	0	4 bits	2 ⁴ (16 bits)																																	
Other than above			Setting prohibited																																		
3	ALV _n	This bit specifies the active level of PWM _n . 0: Active low 1: Active high At reset, the inactive level of the ALV _n bit (low level) is output as PWM output.																																			
2	SYN _n	This bit specifies the period during which the pulse width of PWM _n is to be rewritten. 0: Long period (every 256 PWM cycles (2 ^{x+8} /f _{PWMC})) 1: Short period (every 1 PWM cycle (2 ^x /f _{PWMC}))																																			
1, 0	CKSP _{n1} , CKSP _{n0}	<p>These bits specify the operating clock of PWM_n (f_{PWMC}).</p> <table border="1"> <thead> <tr> <th>CKSP_{n1}</th> <th>CKSP_{n0}</th> <th>Operating clock (f_{PWMC})</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>f_x/4</td> </tr> <tr> <td>0</td> <td>1</td> <td>f_x/8</td> </tr> <tr> <td>1</td> <td>0</td> <td>f_x/16</td> </tr> <tr> <td>1</td> <td>1</td> <td>f_x/32</td> </tr> </tbody> </table>	CKSP _{n1}	CKSP _{n0}	Operating clock (f _{PWMC})	0	0	f _x /4	0	1	f _x /8	1	0	f _x /16	1	1	f _x /32																				
CKSP _{n1}	CKSP _{n0}	Operating clock (f _{PWMC})																																			
0	0	f _x /4																																			
0	1	f _x /8																																			
1	0	f _x /16																																			
1	1	f _x /32																																			

- ★
- Remarks 1.** n = 0, 1
x = 4 to 8 (specified by PMP_{n2} to PMP_{n0} bits)
- 2.** f_x: Main clock

(2) PWM modulo registers 0 and 1 (PWM0 and PWM1)

These 16-bit registers determine the pulse width of the PWM pulse.

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

If the higher 8 bits of the PWMn register are used as a PWMHn register and the lower 8 bits are used as a PWMLn register, these registers can be read or written in 8-bit units.

<1> Modulo H register n (PWMHn): Bits 15 to 8

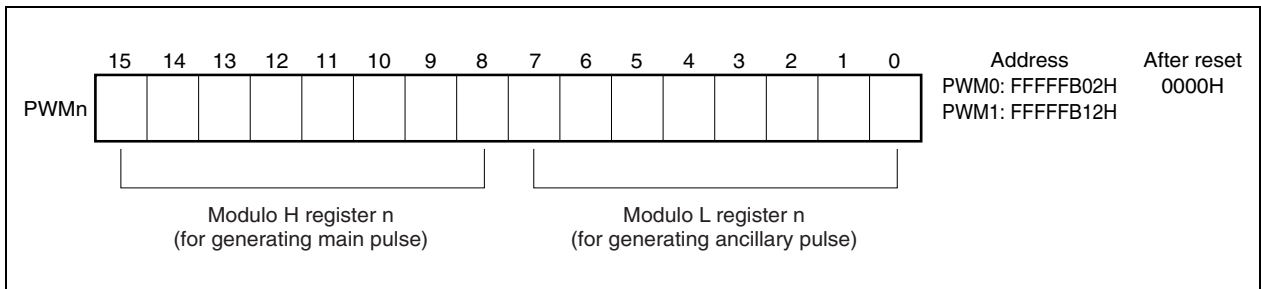
This register specifies the active level of the main pulse.

The data of the active level width of only the bit length specified by the PMPn2 to PMPn0 bits of the PWMCn register is valid. If the counter length is set to 4 to 7 bits by the PMPn2 to PMPn0 bits, data input to the higher bits is invalid.

<2> Modulo L register n (PWMLn): Bits 7 to 0

This register adjusts the timing of the ancillary pulse for fine tuning (see **Figure 13-2**).

A value of 0000H to FFFFH can be set to the PWMn register, and the PWM output changed linearly. If a value of 0000H is set, the inactive level is held. If the value is FFFFH, the PWM output becomes inactive for one ancillary pulse ($1/f_{PWM}$) in one rewrite cycle ($2^{16}/f_{PWM}$) (see **Figure 13-3**).



13.4 Operation

13.4.1 PWM basic operation

PWMn is divided into 256 parts, each of which is called a main pulse. Each main pulse has 4- to 8-bit accuracy. PWMn realizes a signal with an accuracy of 12 to 16 bits by synthesizing the required number of main pulses and an ancillary pulse with a pulse width of one clock.

The main pulse is set by the PMPn2 to PMPn0 bits of PWM control register n (PWMCn), and the pulse width is determined by the value of modulo H register n (valid number of bits). The repeat cycle of the PWM pulse output is the operating clock cycle (f_{PWM}) of PWMn specified by the CKSPn1 and CKSPn0 bits of the PWMCn register divided by 2^x ($f_{PWM}/2^x$).

Of the 256 main pulses, the ancillary pulses are generated only for the main pulses of the set number specified by modulo L register n. The pulse width is $1/f_{PWM}$.

The logical sum of the main pulse and the ancillary pulse is output as the PWM pulse signal. Therefore, the average value of 256 PWM pulse signals is the PWM pulse signal with a resolution of 12 to 16 bits.

The duty factor of the output PWM pulse is determined as follows, by the value set to modulo H register n of PWM modulo register n (PWM0 or PWM1).

(1) If ancillary pulse is not generated

$$\text{Duty of output PWM pulse} = \frac{(\text{Value of modulo H register n})}{2^x}$$

(2) If ancillary pulse is generated

$$\text{Duty of output PWM pulse} = \frac{(\text{Value of modulo H register n}) + 1}{2^x}$$

Remark $x = 4$ to 8 (bit length of main pulse)

Figure 13-2. Example of PWM Output with Main Pulse and Ancillary Pulse

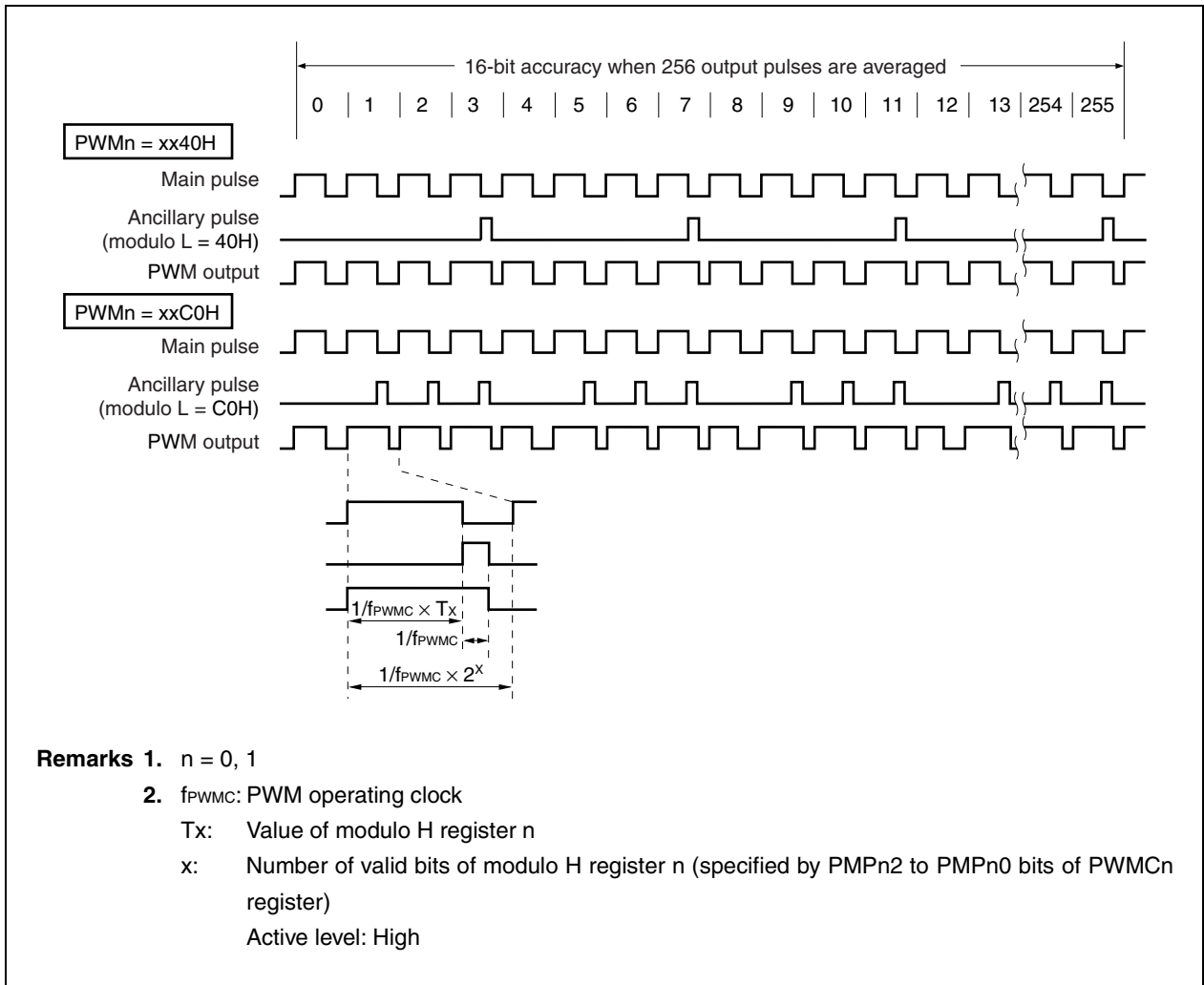
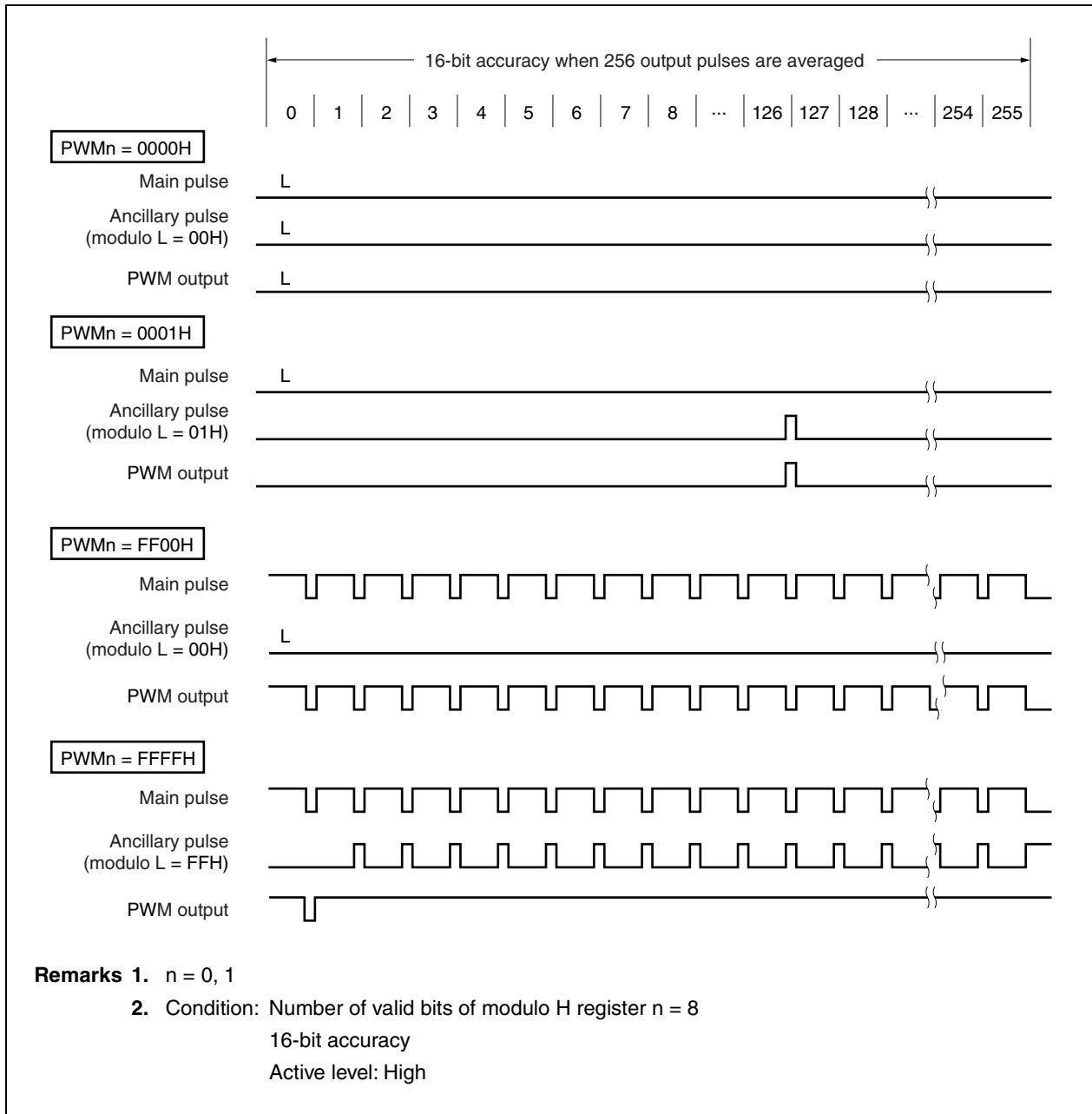


Figure 13-3. Example of PWM Output Operation



13.4.2 Starting/stopping PWM operation

To output a PWM pulse, set data to PWM modulo register n (PWMn) and then set the PWME_n bit of PWM control register n (PWMC_n) to 1 (n = 0, 1).

As a result, the PWM output pin outputs a PWM pulse of the active level specified by the ALV_n bit of the PWMC_n register.

When the PWME_n bit of the PWMC_n register is cleared to 0, the PWM output unit immediately stops the PWM output operation, and the PWM output pin becomes inactive.

(1) Setting for starting PWM operation

Before starting the operation of PWM_n (when PWME_n bit of PWMC_n register = 0), be sure to initialize the following registers.

- PMCDH, PFCDH registers: Set the control mode.
- PWM_n register: Set the pulse width.
- PWMC_n register:
 - CKSP_n1 and CKSP_n0 bits: Specify the operating clock (f_{PWM}) ($f_x/4$, $f_x/8$, $f_x/16$, $f_x/32$)
 - PMP_n2 to PMP_n0 bits: Specify the bit length (x) of the main pulse.
 - ALV_n bit: Specify the active level of the PWM pulse.
 - SYN_n bit: Specify the PWM pulse width rewrite period.

Cautions 1. Do not change the setting of the PMP_n2 to PMP_n0, SYN_n, CKSP_n1, and CKSP_n0 bits when PWME_n bit of the PWMC_n register = 1. Otherwise, the operation cannot be guaranteed.

2. Note that, if the setting of the ALV_n bit is changed during PWM operation (when PWME_n bit = 1), noise may be generated.

Remark n = 0, 1

When the PWME_n bit of the PWMC_n register is set, PWM_n starts operating. Immediately after PWM_n has started operating, however, the PWM pin maintains the status of the port mode (inactive level) until the reload signal of the PWM_n register is generated. After the operation has been started, the PWM output becomes active when the reload signal is generated, regardless of the setting of the SYN_n bit (PWM_n = other than 00xxH). If the timing of rewriting the pulse width is set to 2^{x+8} (long period: SYN_n bit = 0), the operation is started up to $2^{x+8}/f_{PWM}$ after the PWME_n bit has been set. The PWM_n register can be rewritten even during PWM output.

★

13.4.3 Setting active level of PWM pulse

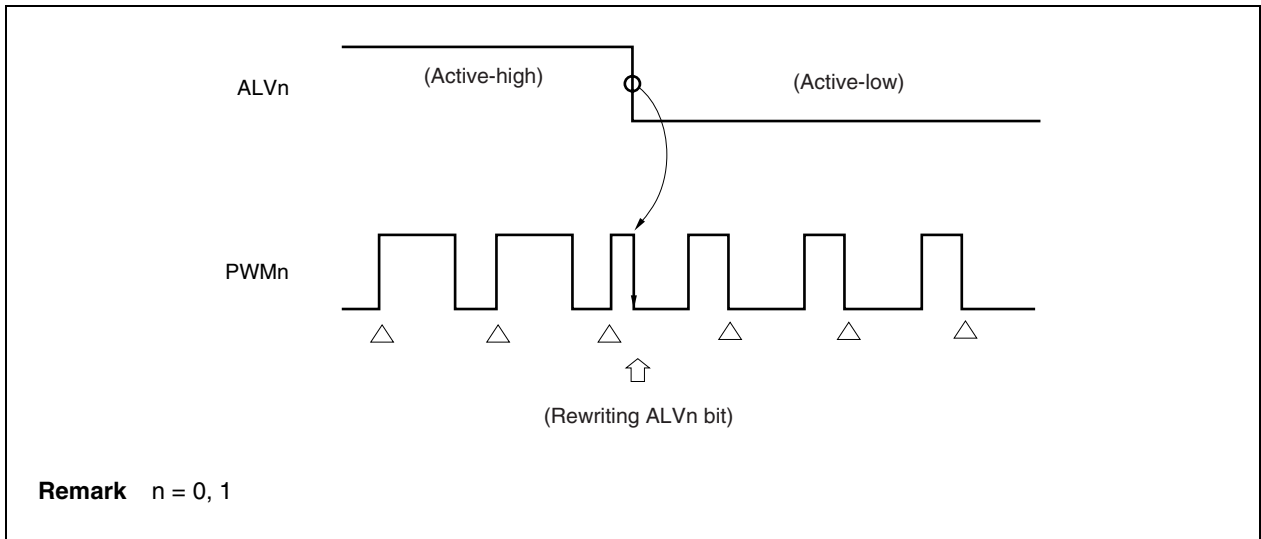
The ALVn bit of PWM control register n (PWMCn) specifies the active level of the PWM pulse output from the PWM output pin (n = 0, 1).

If the ALVn bit is set to 1, an active-high pulse is output; if it is cleared to 0, an active-low pulse is output.

If the ALVn bit is rewritten, the active level of PWM output is immediately changed. The figure below shows the setting of the active level of PWM output and pin status.

Regardless of the setting of the PWME n bit (which enables or disables PWM), the active level of PWM output can be changed by manipulating the ALVn bit.

Figure 13-5. Setting Active Level of PWM Output



13.4.4 Specifying PWM pulse width rewrite period

PWM output is started and the pulse width is changed every 256 cycles of the PWM pulse ($2^{x+8}/f_{PWM}$) or every one PWM cycle ($2^x/f_{PWM}$). This PWM pulse width rewrite period is specified by the SYNn bit of the PWMn register (n = 0, 1).

When the SYNn bit is cleared to 0, the pulse width is changed every 256 PWM pulse cycles ($2^{x+8}/f_{PWM}$) (long period). Therefore, it takes up to 2^{x+8} clocks to output a pulse of the width corresponding to the data written to the PWMn register.

An example of the PWM output timing at this time is shown in Figure 13-6.

When the SYNn bit is set to 1, the pulse width is changed every one PWM pulse cycle ($2^x/f_{PWM}$) (short period). In this case, it takes up to 2^x clocks to output a pulse of the width corresponding to the data written to the PWMn register.

If the PWM pulse rewrite period is specified to be every $2^x/f_{PWM}$ (if the SYNn bit is set to 1), the accuracy of the PWM pulse is x bits or more and (x+8) bits or less, or is lower than the accuracy when the rewrite cycle is set to $2^{x+8}/f_{PWM}$. However, the response speed improves because the repeat cycle increases.

Figure 13-7 shows an example of the PWM output timing where the rewrite timing is $2^x/f_{PWM}$.

Figure 13-6. PWM Output Timing Example 1 (PWM Pulse Width Rewrite Period: $2^{x+8}/f_{PWM}$)

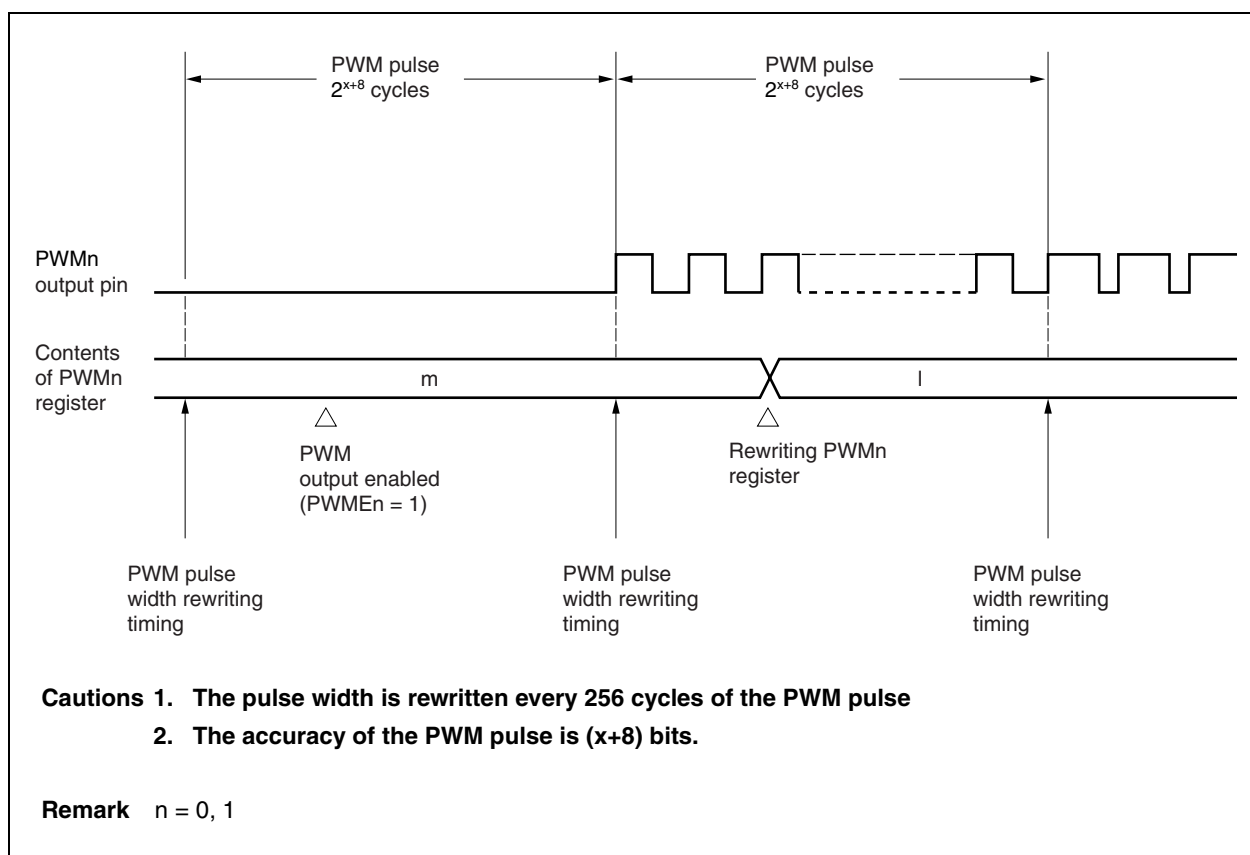
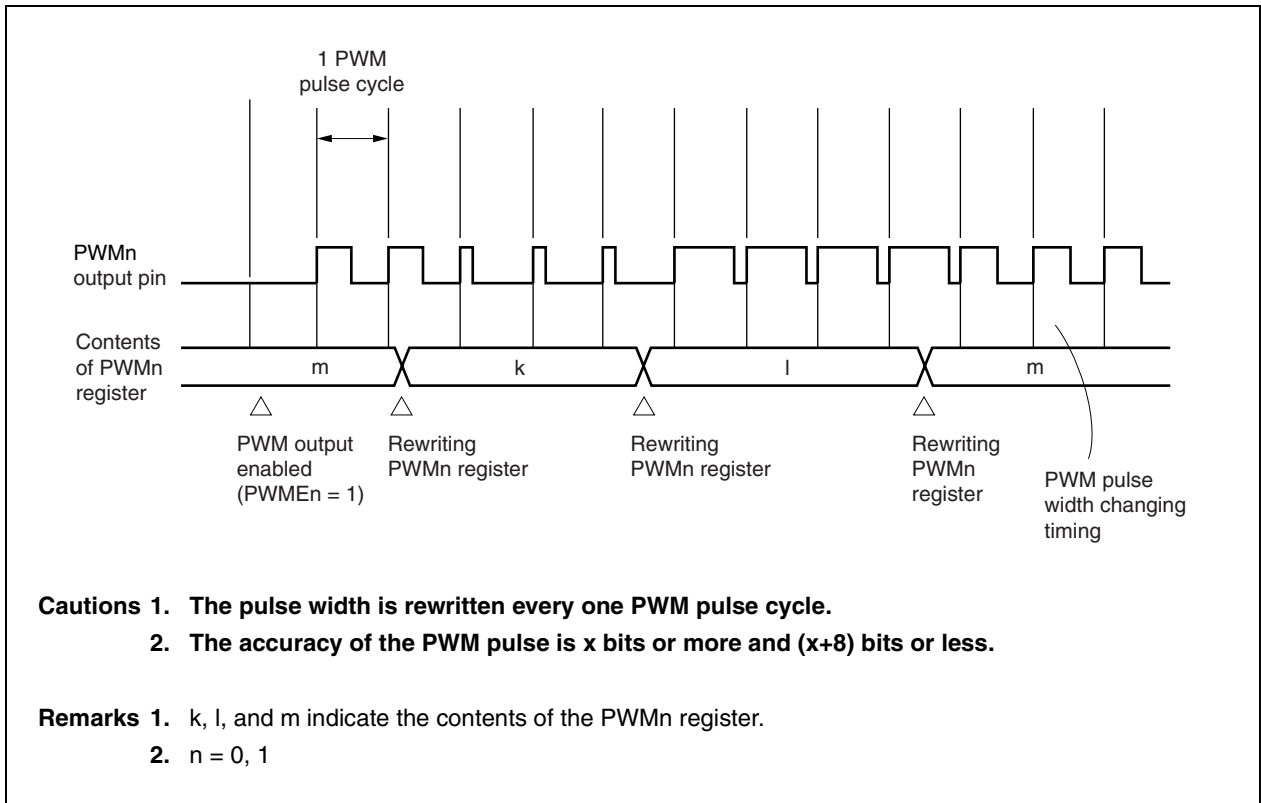


Figure 13-7. PWM Output Timing Example 2 (PWM Pulse Width Rewrite Period: $2^x/f_{PWMC}$)



13.4.5 Repeat cycle

The following table shows the repeat cycle of PWMn ($n = 0, 1$).

★

Table 13-1. Repeat Cycle of PWMn

Main Pulse Accuracy	Ancillary Pulse Accuracy	Repeat Cycle	Pulse Width Rewrite Period	
			Long Period (SYNn Bit = 0)	Short Period (SYNn Bit = 1)
4 bits	8 bits	$16/f_{PWMC}$	$2^{12}/f_{PWMC}$	$2^4/f_{PWMC}$
5 bits	8 bits	$32/f_{PWMC}$	$2^{13}/f_{PWMC}$	$2^5/f_{PWMC}$
6 bits	8 bits	$64/f_{PWMC}$	$2^{14}/f_{PWMC}$	$2^6/f_{PWMC}$
7 bits	8 bits	$128/f_{PWMC}$	$2^{15}/f_{PWMC}$	$2^7/f_{PWMC}$
8 bits	8 bits	$256/f_{PWMC}$	$2^{16}/f_{PWMC}$	$2^8/f_{PWMC}$

- Remarks**
1. $n = 0, 1$
 2. f_{PWMC} : PWM operating clock

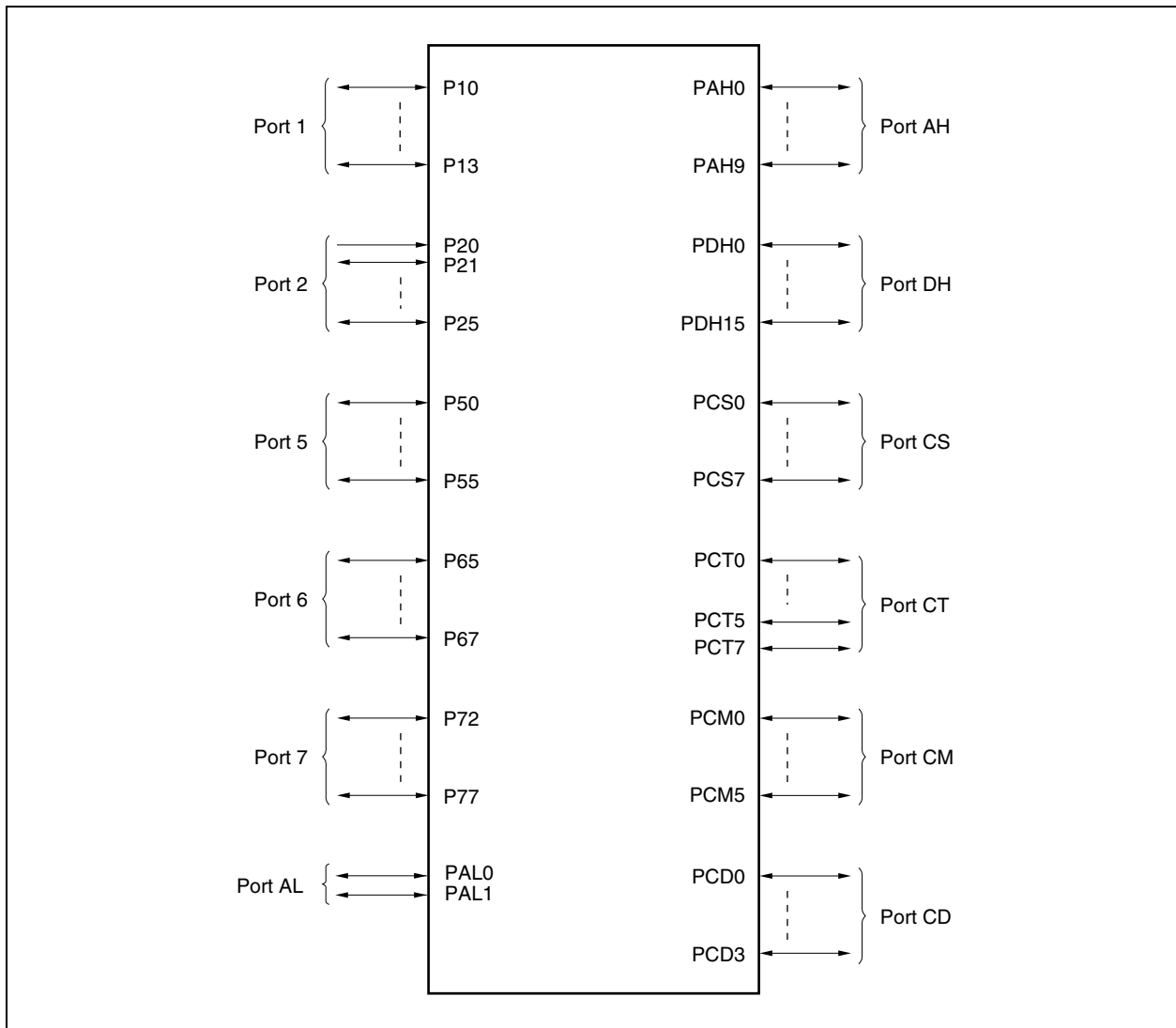
CHAPTER 14 PORT FUNCTIONS

14.1 Features

- Input-only ports: 1
I/O ports: 77
- Function alternately as other peripheral I/O pins.
- It is possible to specify input and output in 1-bit units.

14.2 Port Configuration

The V850E/ME2 incorporates a total of 78 input/output ports (including 1 input-only port) labeled ports 1, 2, 5 to 7, AL, AH, DH, CS, CT, CM, and CD. 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 on-chip peripheral I/O input/output pins in the control mode. For the block types of each port, see **(3) Block diagram of port.**

Port Name	Pin Name	Port Function	Function in Control Mode	Block Type
Port 1	P10 to P13	4-bit I/O	Serial interface I/O (CSI30, UARTB0) External interrupt input USB clock signal input	F-1, F-3, H-1, J-1
Port 2	P20 to P25	1-bit input, 5-bit I/O	NMI input Serial interface I/O (CSI31, UARTB1) External interrupt input	A-1, F-4, G-2, H-1
Port 5	P50 to P55	6-bit I/O	DMA controller I/O External interrupt input Real-time pulse unit (RPU) I/O	F-2, F-3, G-1, G-2, J-1
Port 6	P65 to P67	3-bit I/O	Real-time pulse unit (RPU) I/O External interrupt input	L-3, L-5
Port 7	P72 to P77	6-bit I/O	DMA controller I/O Real-time pulse unit (RPU) I/O External interrupt input	L-1, L-2, L-4
Port AL	PAL0, PAL1	2-bit I/O	External address bus (A0, A1) External interrupt input	G-3
Port AH	PAH0 to PAH9	10-bit input	External address bus (A16 to A25)	D-2
Port DH	PDH0 to PDH15	16-bit I/O	External data bus (D16 to D31) External interrupt input PWM output Real-time pulse unit (RPU) I/O	M-1, M-2, M-3
Port CS	PCS0 to PCS7	8-bit I/O	External bus interface control signal output	D-2, J-2
Port CT	PCT0 to PCT5, PCT7	7-bit I/O	External bus interface control signal output	D-2, J-3
Port CM	PCM0 to PCM5	6-bit I/O	Wait insertion signal input External bus interface control signal I/O Self-refresh request signal input A/D converter external trigger input	C-1, D-1, D-2, F-5
Port CD	PCD0 to PCD3	4-bit I/O	External bus interface control signal output Bus clock output	D-1, D-2

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 = 1, 2, 5 to 7, AL, AH, DH, CS, CT, CM, and CD).**
- <2> The mode is switched to control mode by the port n mode control register (PMCN).**

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. To manipulate a port by using a bit manipulation instruction (SET1, CLR1, or NOT1), read the port in byte units, process the data of only the bit to be manipulated, and then write back the byte data to the port after conversion. In the case of a port having a mixture of input and output pins, the contents of the output latch are written over bits other than the bit to be manipulated. Consequently, the output latch of the input pin is undefined (in the input mode, however, the pin status does not change because the output buffer is turned off).**

To change the port mode from the input to the output mode, therefore, set an expected output value to the corresponding bits, and then change the mode to the output mode. The same applies to a port that has both a control mode and an output mode.

(2) Function when each port's pins are reset and registers that set the port/control mode

(1/2)

Port Name	Pin Name	Pin Function After Reset		Register That Sets the Mode
Port 1	P10/ $\overline{\text{INTP10}}$ / $\overline{\text{UCLK}}$	P10 (input mode)		PMC1, PFC1
	P11/ $\overline{\text{INTP11}}$ / $\overline{\text{SCK0}}$	P11 (input mode)		
	P12/SI0/RXD0	P12 (input mode)		
	P13/SO0/TXD0	P13 (input mode)		
Port 2	P20/NMI	NMI		PMC2, PFC2
	P21/ $\overline{\text{INTP21}}$ /RXD1	P21 (input mode)		
	P22/ $\overline{\text{INTP22}}$ /TXD1	P22 (input mode)		
	P23/ $\overline{\text{INTP23}}$ / $\overline{\text{SCK1}}$	P23 (input mode)		
	P24/ $\overline{\text{INTP24}}$ /SI1	P24 (input mode)		
	P25/ $\overline{\text{INTP25}}$ /SO1	P25 (input mode)		
Port 5	P50/ $\overline{\text{INTP50}}$ / $\overline{\text{DMARQ0}}$	P50 (input mode)		PMC5, PFC5
	P51/ $\overline{\text{INTP51}}$ / $\overline{\text{DMAAK0}}$	P51 (input mode)		
	P52/ $\overline{\text{INTP52}}$ /TC0	P52 (input mode)		
	P53/ $\overline{\text{INTPC00}}$ / $\overline{\text{TIC0}}$ / $\overline{\text{DMARQ1}}$	P53 (input mode)		
	P54/ $\overline{\text{INTPC01}}$ / $\overline{\text{DMAAK1}}$	P54 (input mode)		
	P55/ $\overline{\text{TOC0}}$ / $\overline{\text{TC1}}$	P55 (input mode)		
Port 6	P65/ $\overline{\text{INTP65}}$ / $\overline{\text{INTPC10}}$ /TIC1	P65 (input mode)		PMC6, PFC6
	P66/ $\overline{\text{INTP66}}$ / $\overline{\text{INTPC11}}$	P66 (input mode)		
	P67/ $\overline{\text{INTP67}}$ /TOC1	P67 (input mode)		
Port 7	P72/ $\overline{\text{INTPC20}}$ / $\overline{\text{TIC2}}$ / $\overline{\text{DMARQ2}}$	P72 (input mode)		PMC7, PFC7
	P73/ $\overline{\text{INTPC21}}$ / $\overline{\text{DMAAK2}}$	P73 (input mode)		
	P74/ $\overline{\text{TOC2}}$ / $\overline{\text{TC2}}$	P74 (input mode)		
	P75/ $\overline{\text{INTPC30}}$ / $\overline{\text{TIC3}}$ / $\overline{\text{DMARQ3}}$	P75 (input mode)		
	P76/ $\overline{\text{INTPC31}}$ / $\overline{\text{DMAAK3}}$	P76 (input mode)		
	P77/ $\overline{\text{TOC3}}$ / $\overline{\text{TC3}}$	P77 (input mode)		
★ Port AL	PAL0/ $\overline{\text{INTPL0}}$ /A0	PAL0 (input mode)	$\overline{\text{INTPL0}}$ /A0	PMCAL, PFCALL
	PAL1/ $\overline{\text{INTPL1}}$ /A1	PAL1 (input mode)	$\overline{\text{INTPL1}}$ /A1	
Port AH	PAH0/A16 to PAH9/A25	PAH0 to PAH9 (input mode)	A16 to A25	PMCAH
Port CS	PCS0/ $\overline{\text{CS0}}$	PCS0 (input mode)	$\overline{\text{CS0}}$	PMCCS
	PCS1/ $\overline{\text{CS1}}$	PCS1 (input mode)	$\overline{\text{CS1}}$	
	PCS2/ $\overline{\text{CS2}}$ / $\overline{\text{IOWR}}$	PCS2 (input mode)	$\overline{\text{CS2}}$ / $\overline{\text{IOWR}}$	PMCCS, PFCCS
	PCS3/ $\overline{\text{CS3}}$	PCS3 (input mode)	$\overline{\text{CS3}}$	PMCCS
	PCS4/ $\overline{\text{CS4}}$	PCS4 (input mode)	$\overline{\text{CS4}}$	
	PCS5/ $\overline{\text{CS5}}$ / $\overline{\text{IORD}}$	PCS5 (input mode)	$\overline{\text{CS5}}$ / $\overline{\text{IORD}}$	PMCCS, PFCCS
	PCS6/ $\overline{\text{CS6}}$	PCS6 (input mode)	$\overline{\text{CS6}}$	PMCCS
	PCS7/ $\overline{\text{CS7}}$	PCS7 (input mode)	$\overline{\text{CS7}}$	

Port Name	Pin Name	Pin Function After Reset		Register That Sets the Mode
Port CT	PCT0/LLWR/LLBE/LLDQM	PCT0 (input mode)	LLWR/LLBE/LLDQM	PMCCCT, PFCCT
	PCT1/LUWR/LUBE/LUDQM	PCT1 (input mode)	LUWR/LUBE/LUDQM	
	PCT2/ULWR/ULBE/ULDQM	PCT2 (input mode)	ULWR/ULBE/ULDQM	
	PCT3/UUWR/UUBE/UUDQM	PCT3 (input mode)	UUWR/UUBE/UUDQM	
	PCT4/RD	PCT4 (input mode)	R \bar{D}	PMCCCT
	PCT5/WE/WR	PCT5 (input mode)	WE/WR	
	PCT7/BCYST	PCT7 (input mode)	BCYST	
Port CM	PCM0/WAIT	PCM0 (input mode)	WAIT	PMCCM
	PCM1	PCM1 (input mode)	–	–
	PCM2/HLDAK	PCM2 (input mode)	H \bar{LDAK}	PMCCM
	PCM3/HLDRQ	PCM3 (input mode)	H \bar{LDRQ}	
	PCM4/REFRQ	PCM4 (input mode)	R $\bar{E}FRQ$	
	PCM5/SELFREF/ADTRG	PCM5 (input mode)	S $\bar{E}LFREF/ADTRG$	PMCCM, PFCM
Port CD	PCD0/SDCKE	PCD0 (input mode)	SDCKE	PMCCD
	PCD1/BUSCLK	PCD1 (input mode)	BUSCLK	
	PCD2/SDCAS	PCD2 (input mode)	S $\bar{D}CAS$	
	PCD3/SDRAS	PCD3 (input mode)	S $\bar{D}RAS$	
Port DH	PDH0/D16/INTPD0	PDH0 (input mode)	D16/INTPD0	PMCDH
	PDH1/D17/INTPD1	PDH1 (input mode)	D17/INTPD1	
	PDH2/D18/INTPD2/TOC4	PDH2 (input mode)	D18/INTPD2/TOC4	PMCDH, PFCDH
	PDH3/D19/INTPD3	PDH3 (input mode)	D19/INTPD3	PMCDH
	PDH4/D20/INTPD4	PDH4 (input mode)	D20/INTPD4	
	PDH5/D21/INTPD5/TOC5	PDH5 (input mode)	D21/INTPD5/TOC5	PMCDH, PFCDH
	PDH6/D22/INTPD6/INTP100/TCUD10	PDH6 (input mode)	D22/INTPD6/INTP100/TCUD10	
	PDH7/D23/INTPD7/INTP101/TCLR10	PDH7 (input mode)	D23/INTPD7/INTP101/TCLR10	
	PDH8/D24/INTPD8/TO10	PDH8 (input mode)	D24/INTPD8/TO10	
	PDH9/D25/INTPD9/TIUD10	PDH9 (input mode)	D25/INTPD9/TIUD10	
	PDH10/D26/INTPD10/INTP110/TCUD11	PDH10 (input mode)	D26/INTPD10/INTP110/TCUD11	
	PDH11/D27/INTPD11/INTP111/TCLR11	PDH11 (input mode)	D27/INTPD11/INTP111/TCLR11	
	PDH12/D28/INTPD12/TO11	PDH12 (input mode)	D28/INTPD12/TO11	
	PDH13/D29/INTPD13/TIUD11	PDH13 (input mode)	D29/INTPD13/TIUD11	
	PDH14/D30/INTPD14/PWM0	PDH14 (input mode)	D30/INTPD14/PWM0	
PDH15/D31/INTPD15/PWM1	PDH15 (input mode)	D31/INTPD15/PWM1		

(3) Block diagram of port

Figure 14-1. Block Diagram of Type A-1

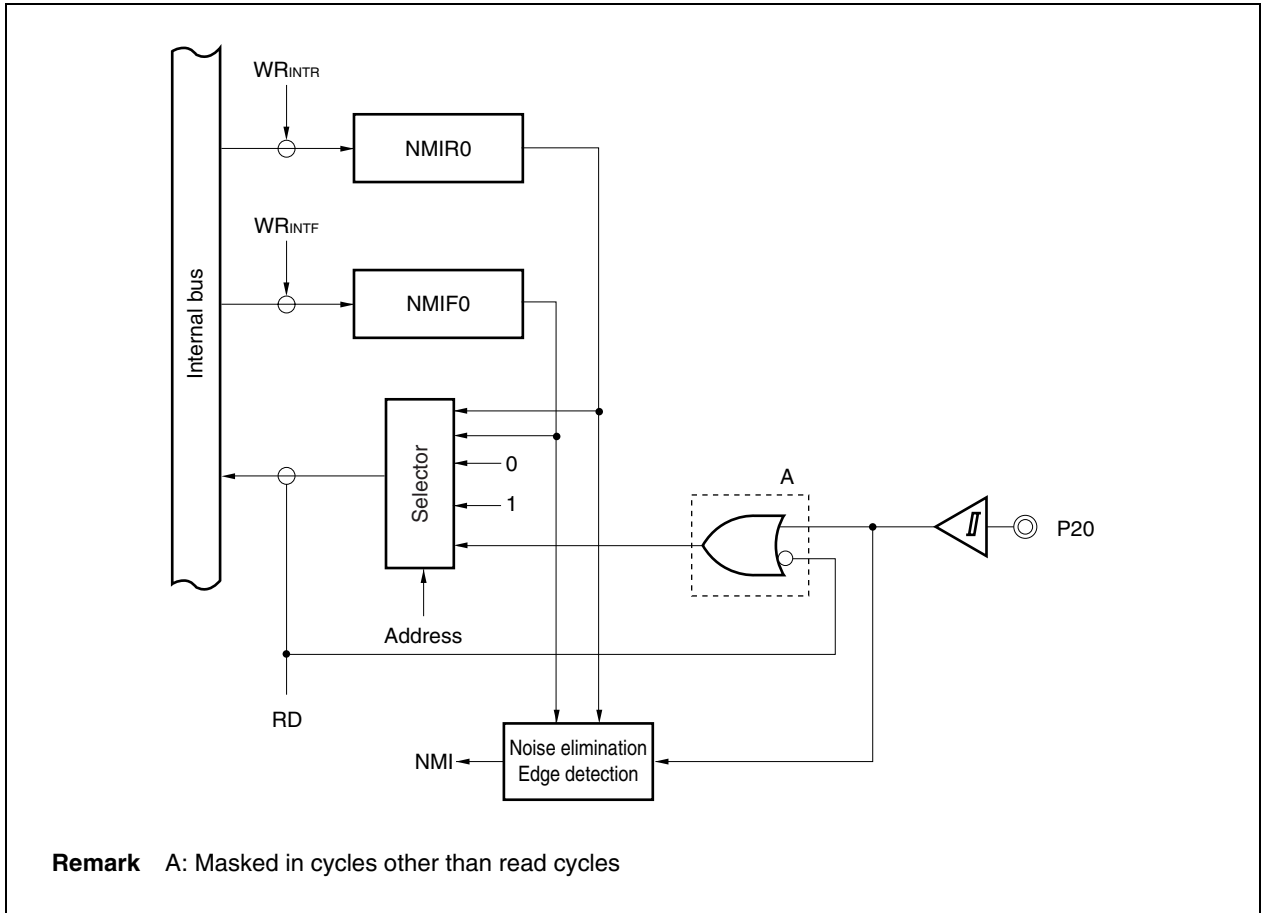


Figure 14-3. Block Diagram of Type D-1

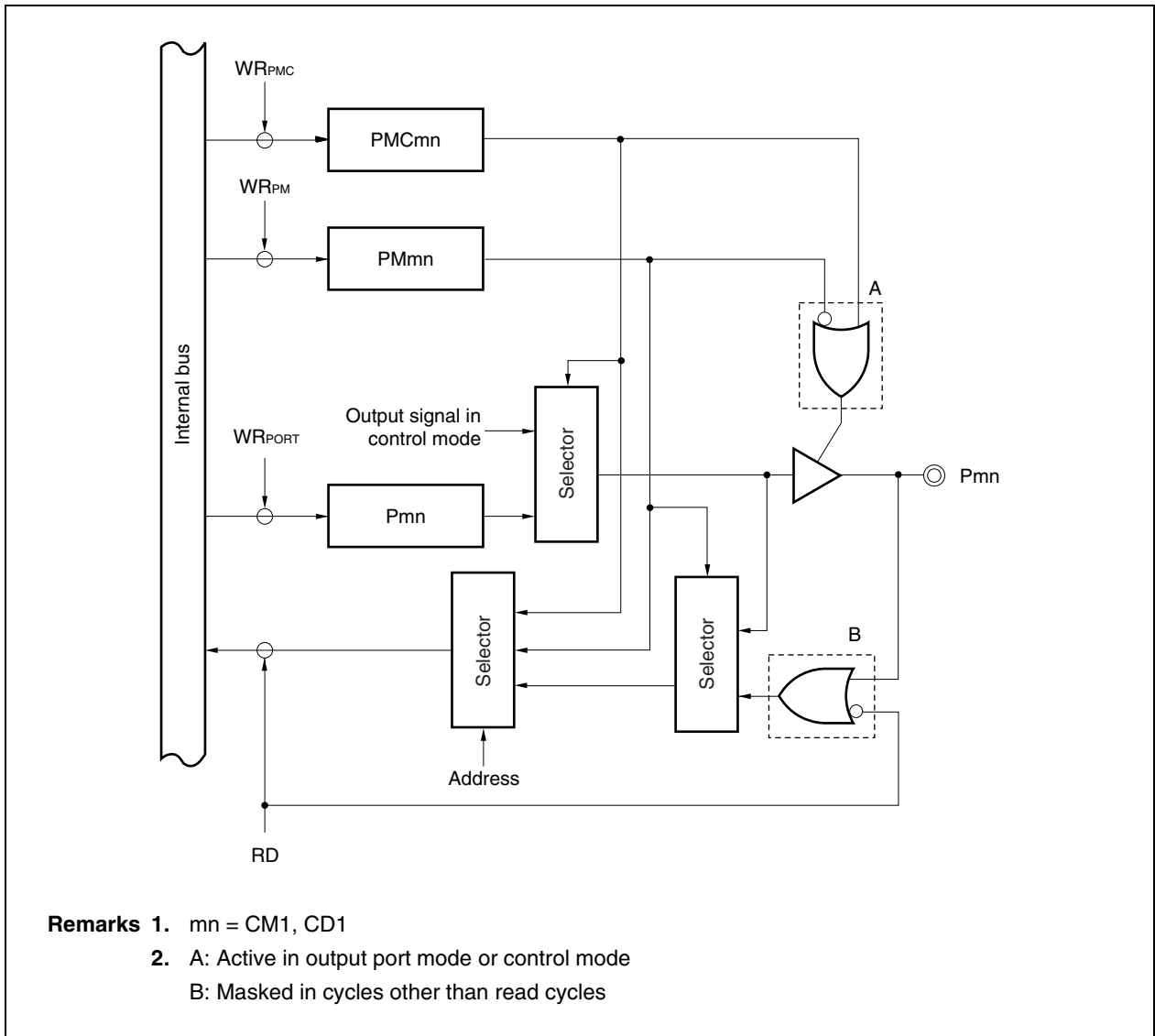
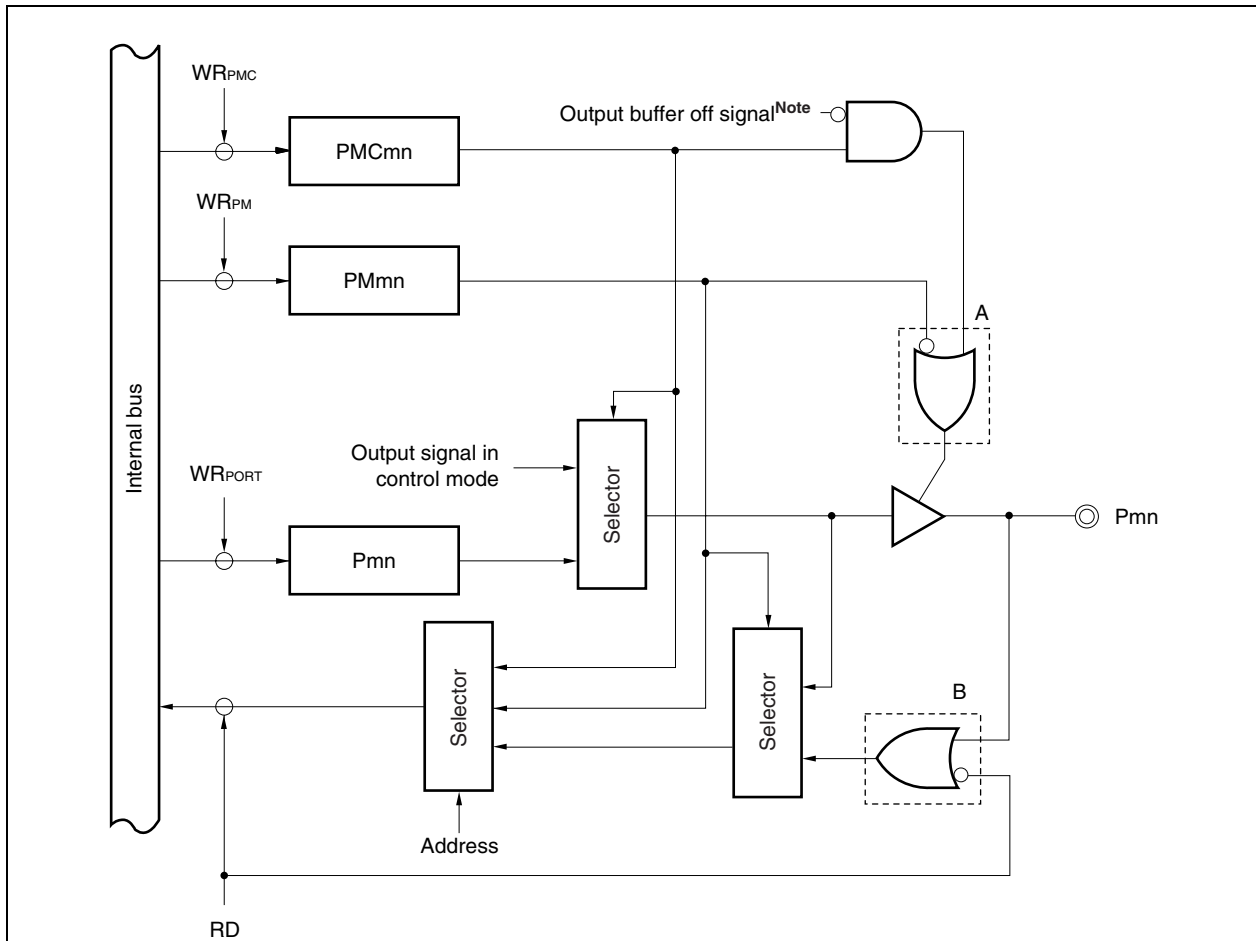


Figure 14-4. Block Diagram of Type D-2

**Note** PAH0 to PAH9:

These signals become active in the IDLE mode and software STOP mode, and by bus hold and reset.

PCS0, PCS1, PCS3, PCS4, PCS6, PCS7, PCD2, PCD3, PCT4, PCT5, PCT7:

These signals become active by bus hold and reset.

PCM2, PCM4, PCD0:

These signals become active at reset.

Remarks 1. mn = AH0 to AH9, CS0, CS1, CS3, CS4, CS6, CS7, CT4, CT5, CT7, CM2, CM4, CD0, CD2, CD3

2. A: Active in output port mode or control mode

B: Masked in cycles other than read cycles

Figure 14-5. Block Diagram of Type F-1

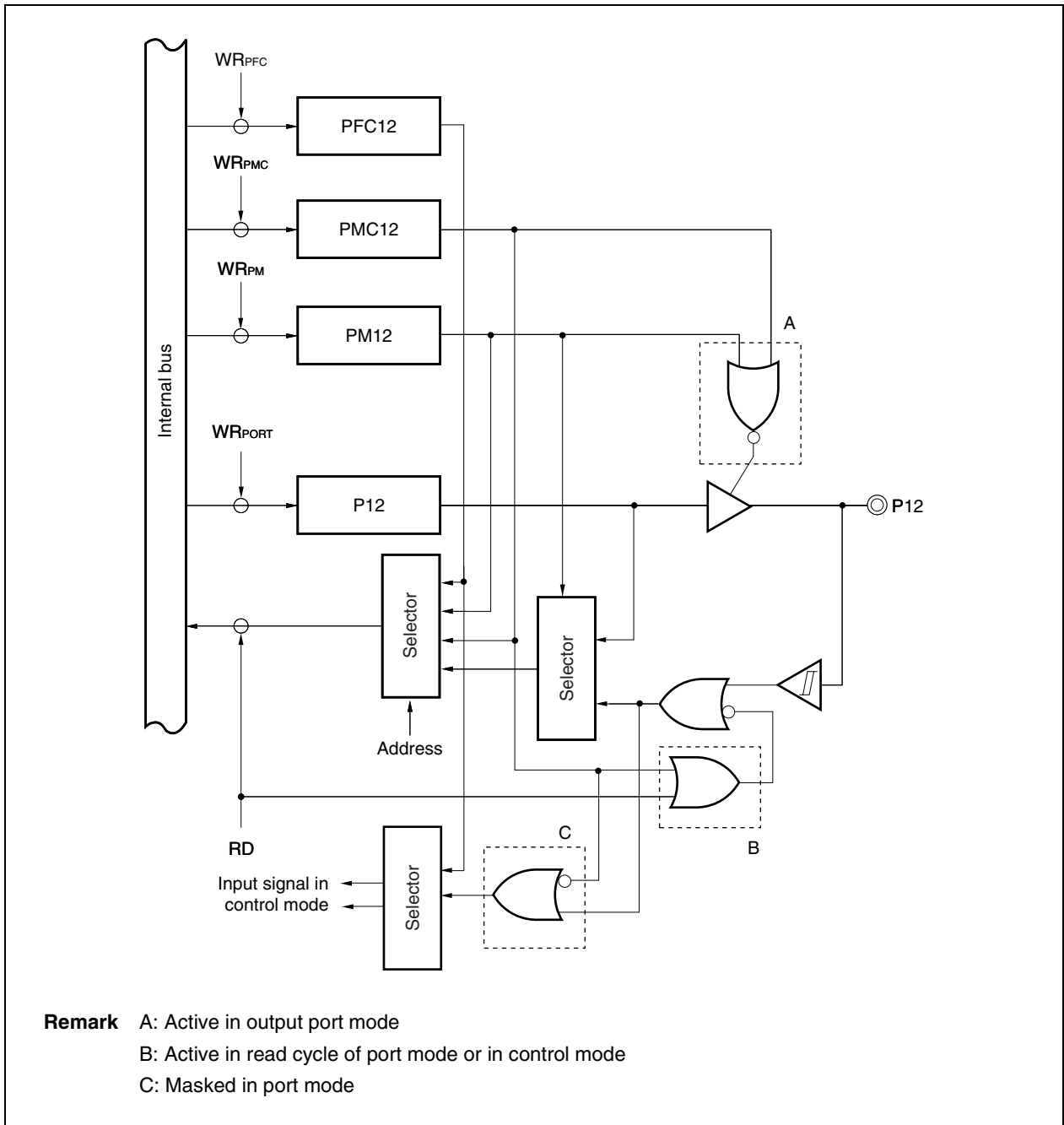


Figure 14-6. Block Diagram of Type F-2

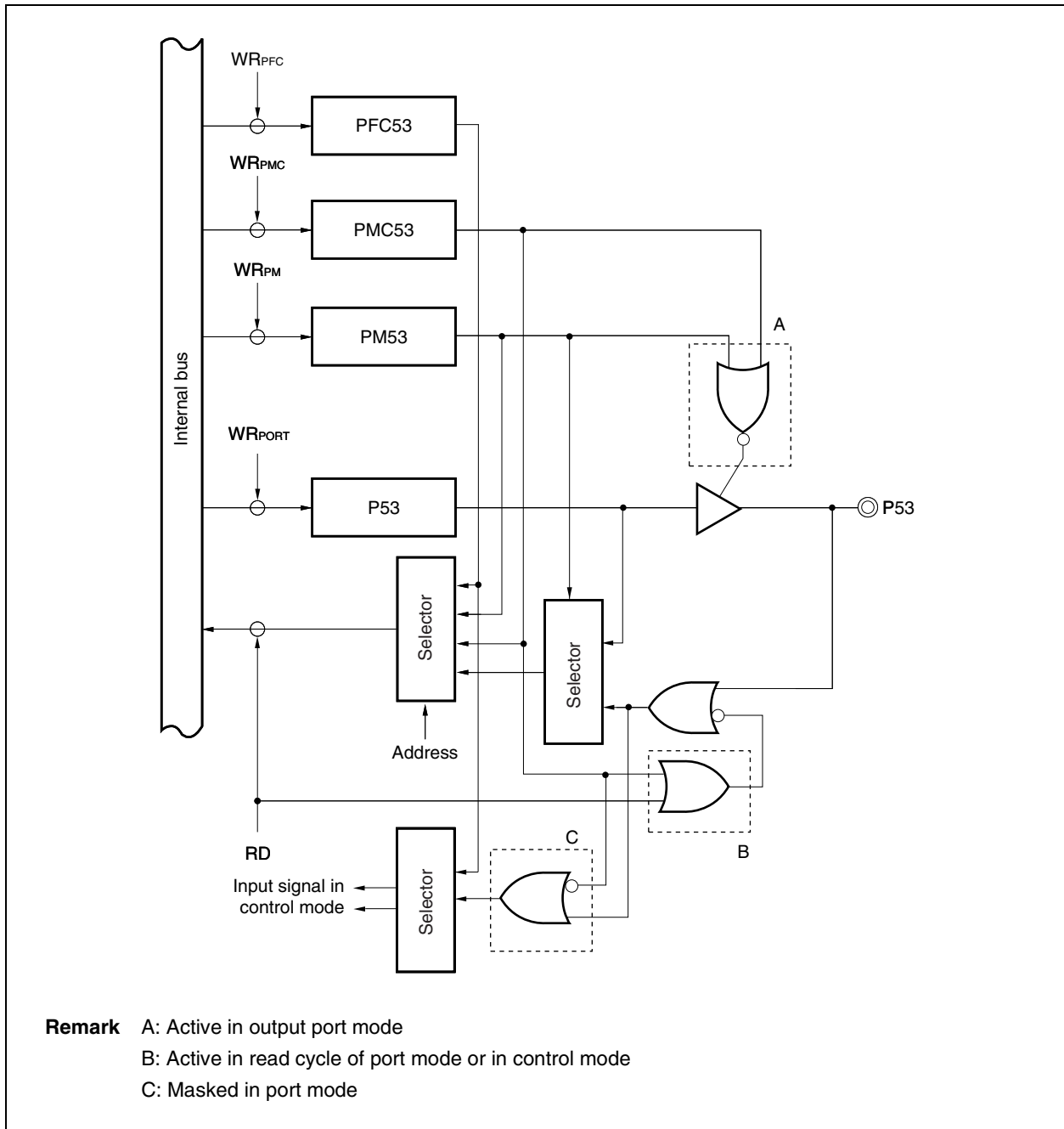


Figure 14-7. Block Diagram of Type F-3

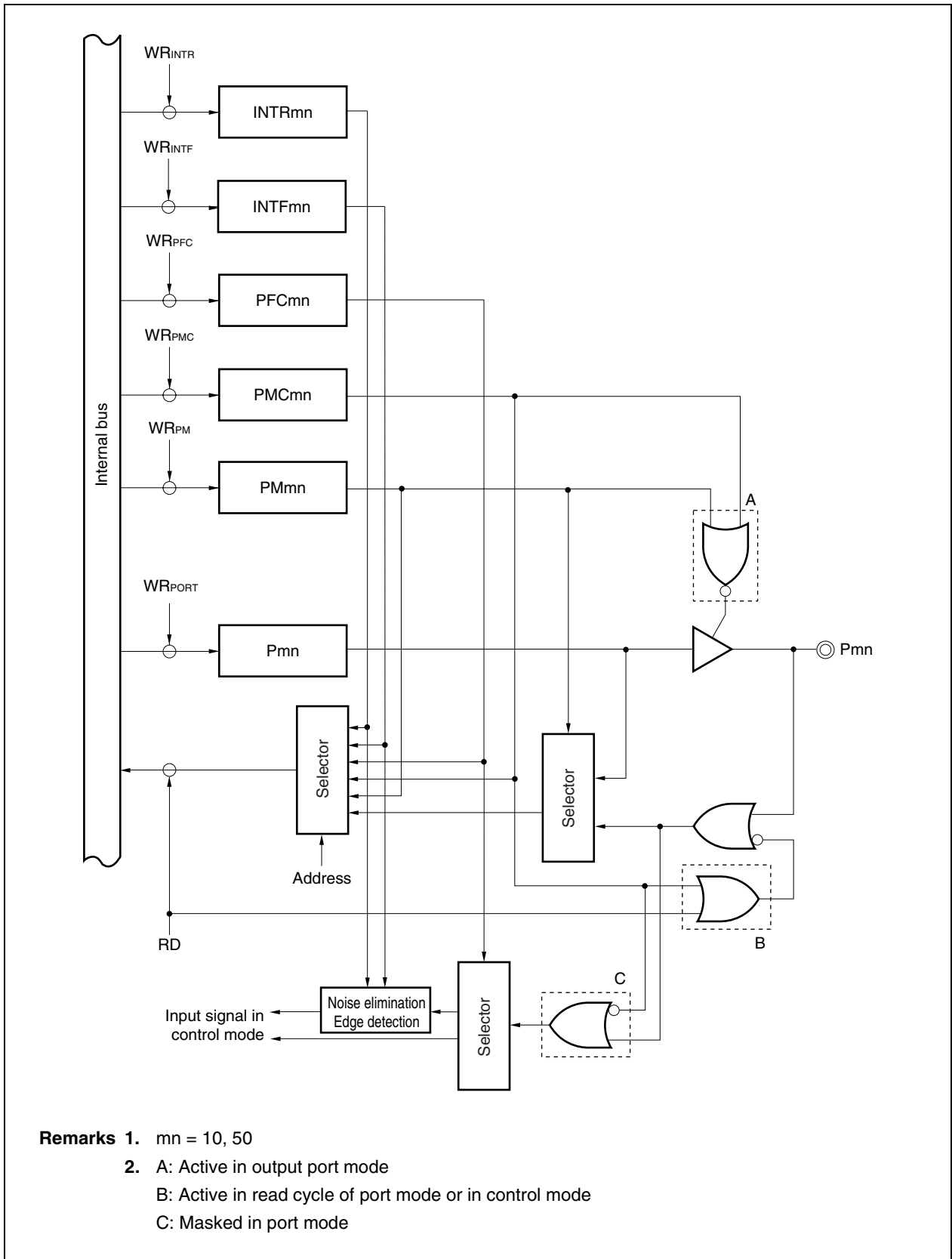


Figure 14-8. Block Diagram of Type F-4

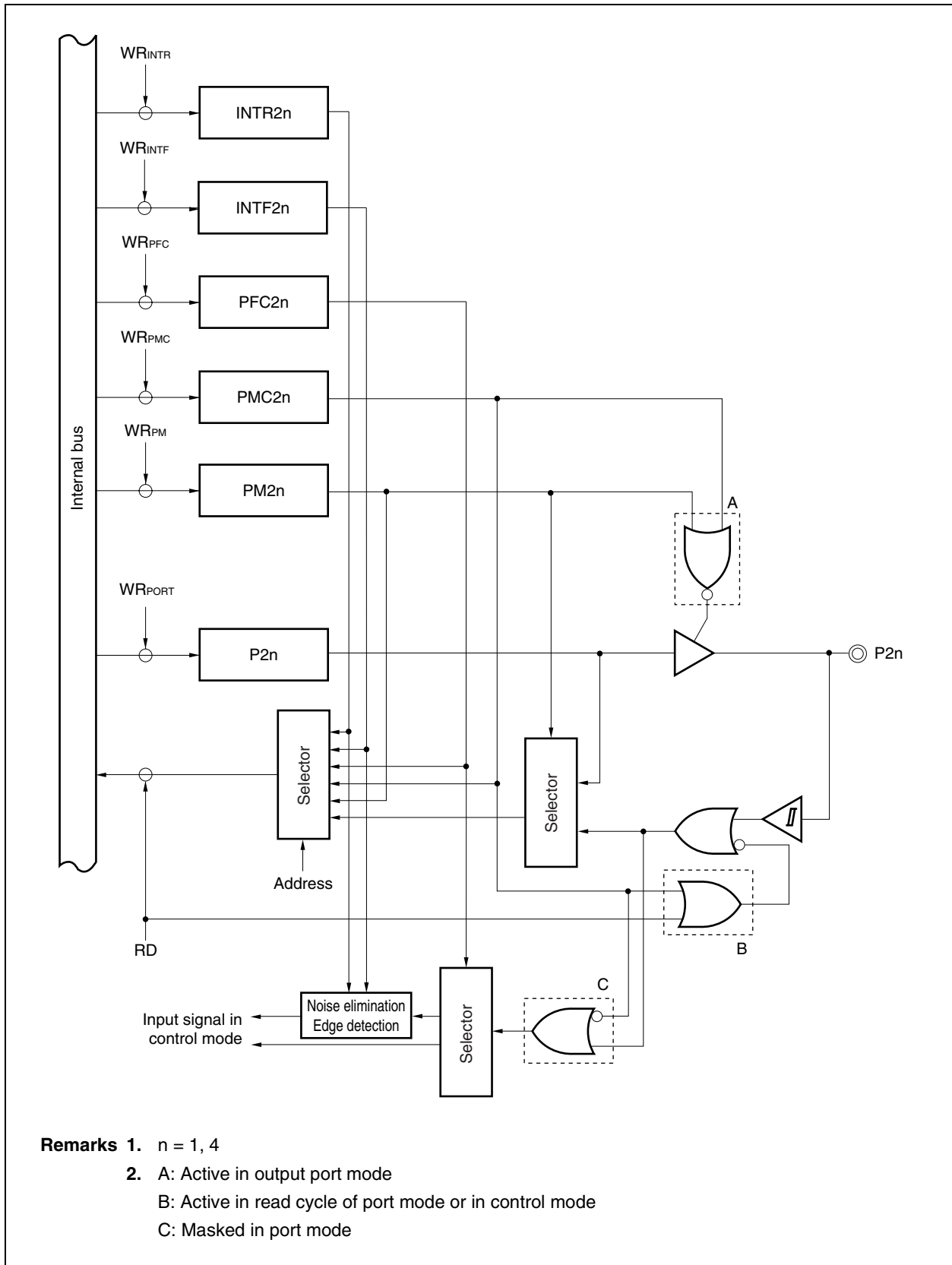


Figure 14-9. Block Diagram of Type F-5

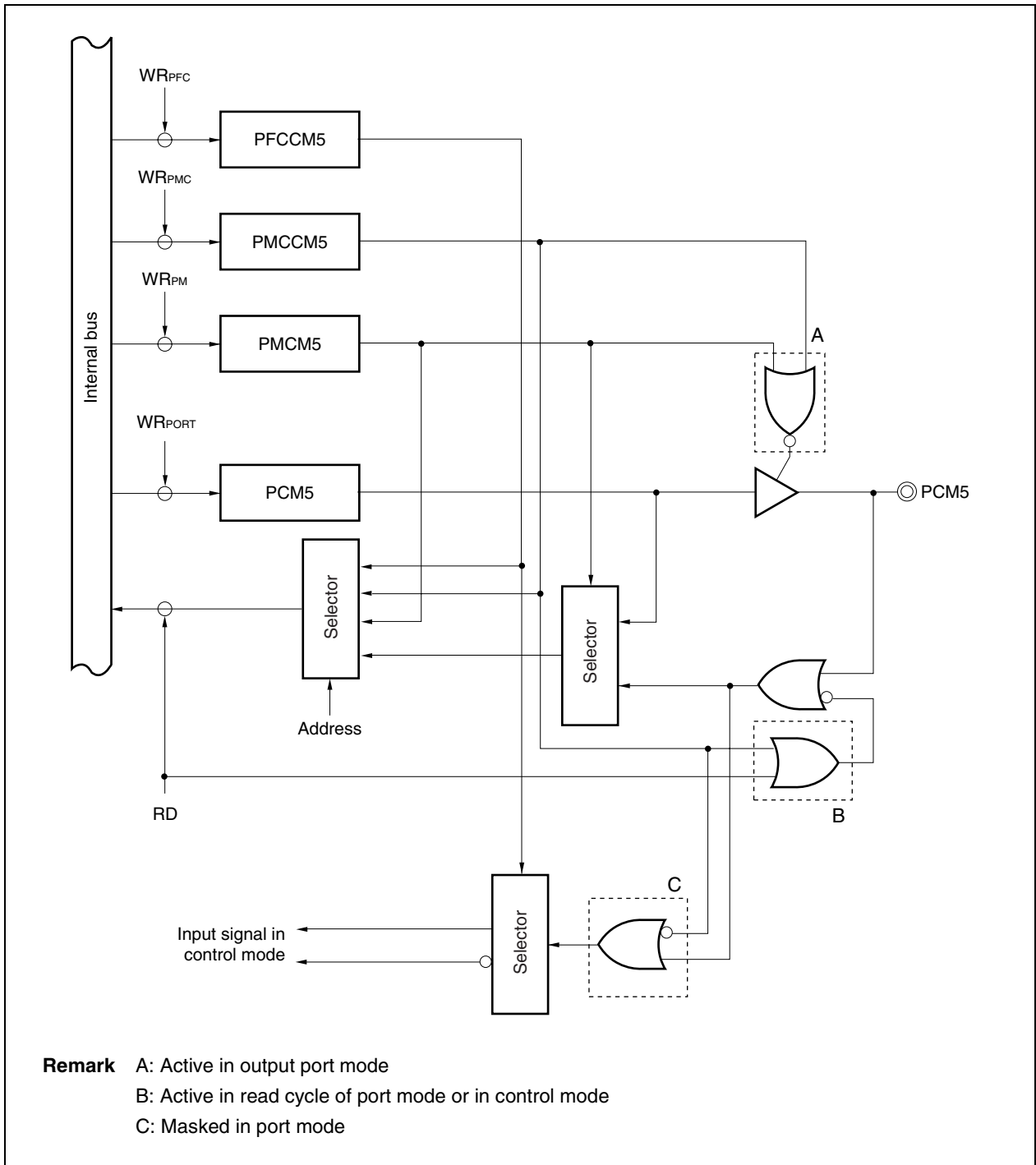


Figure 14-11. Block Diagram of Type G-2

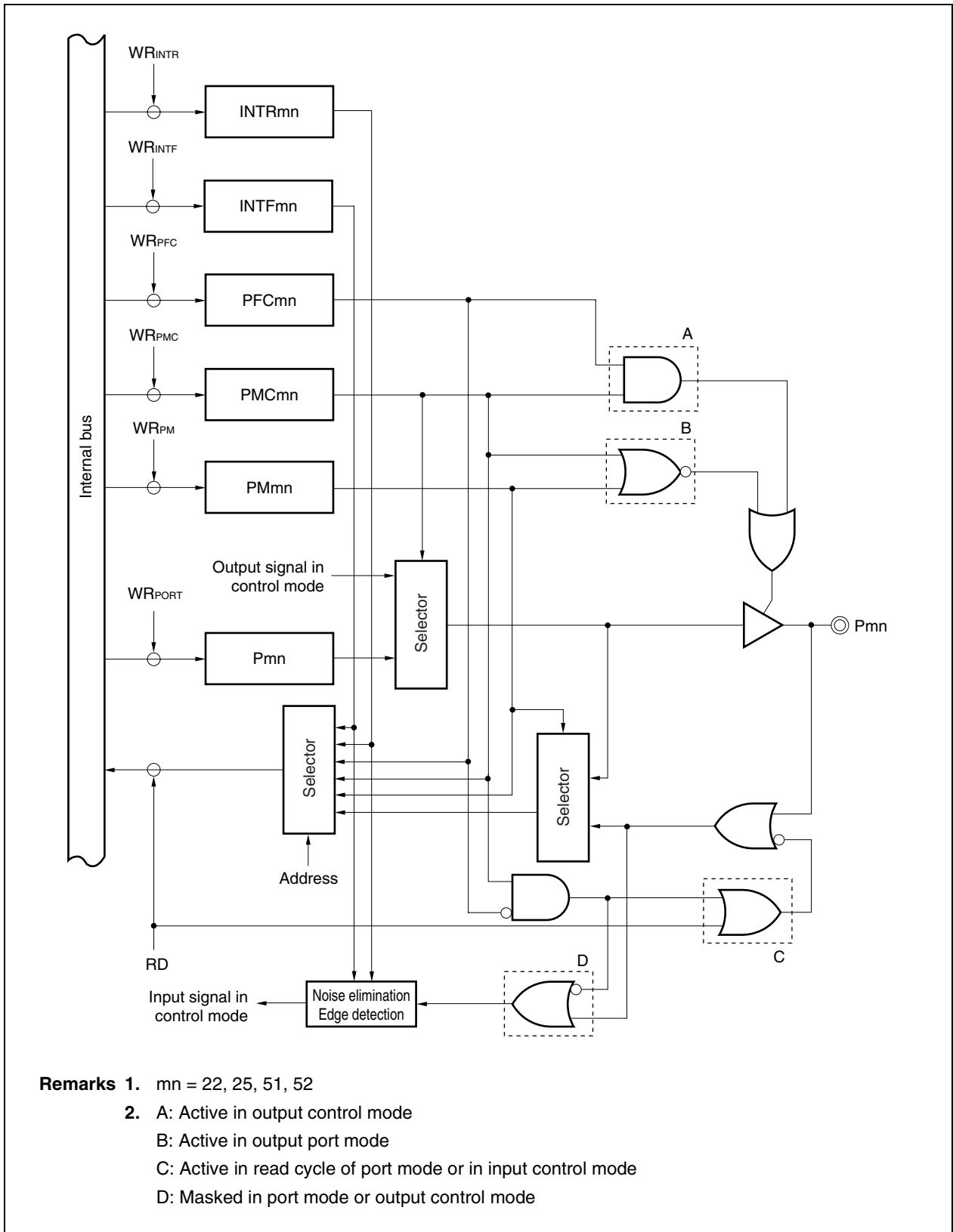
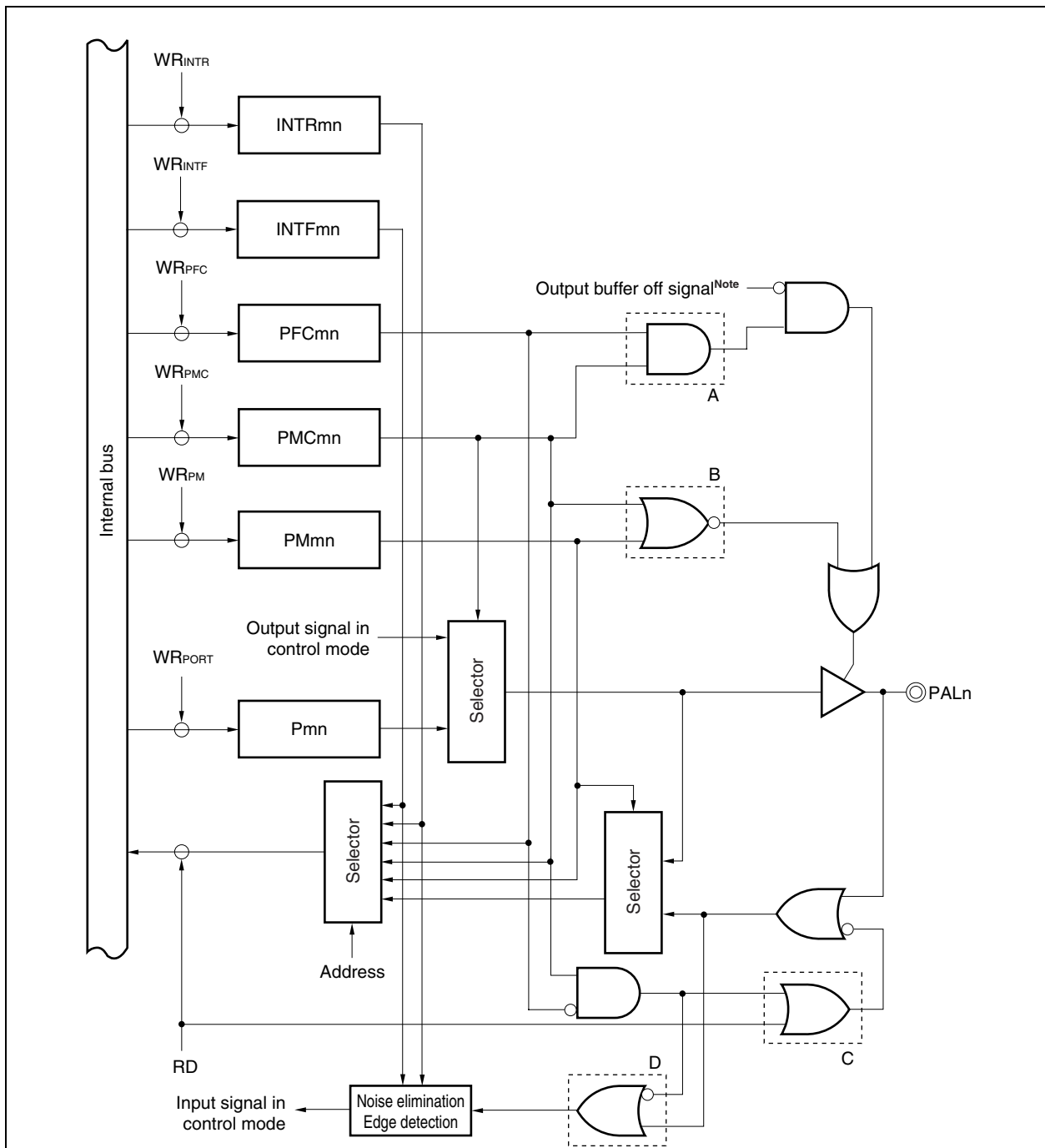


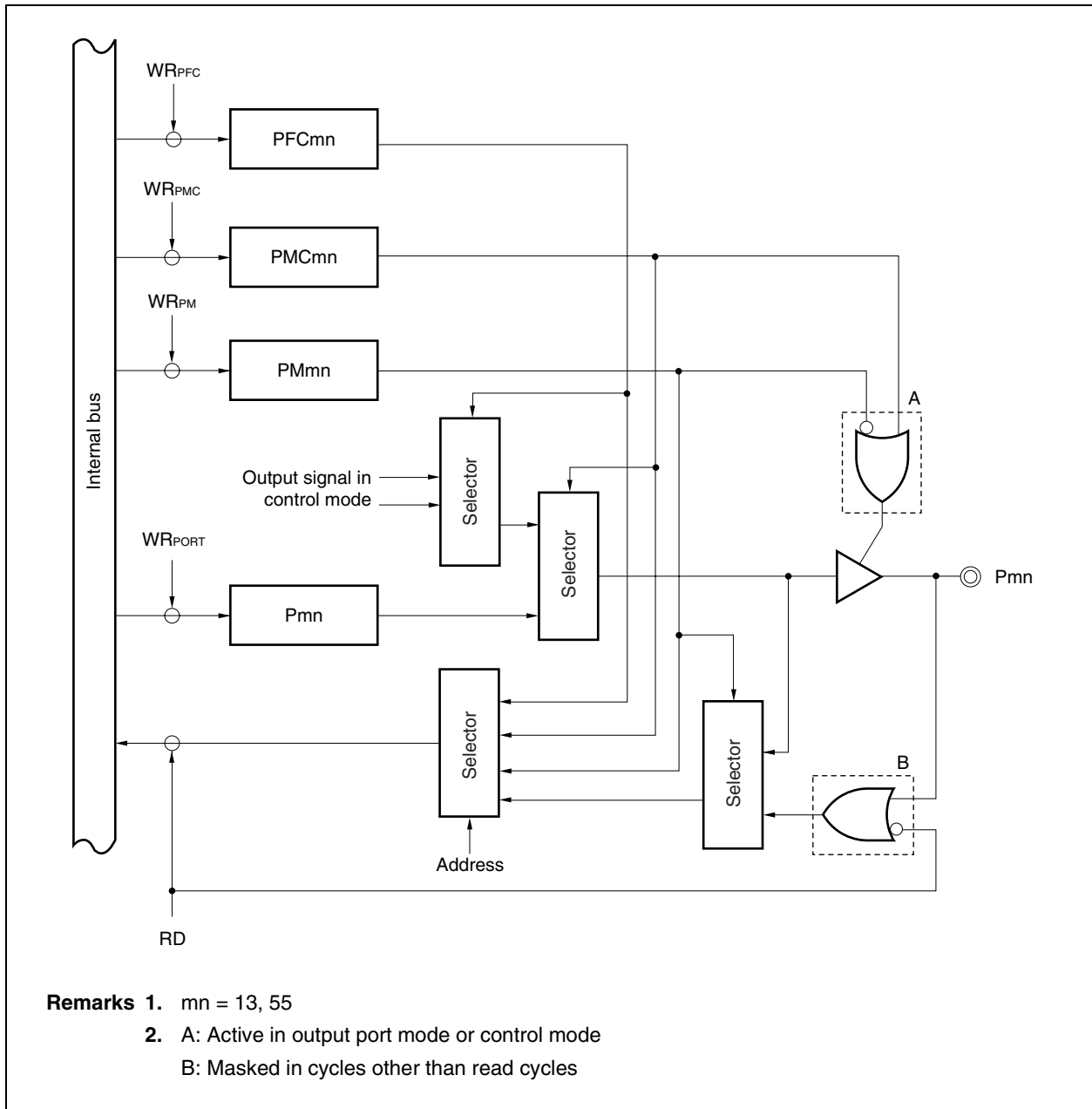
Figure 14-12. Block Diagram of Type G-3



Note Signal that becomes active in IDLE mode and software STOP mode, and by bus hold and reset

- Remarks**
1. $n = 0, 1$
 2. A: Active in output control mode
 - B: Active in output port mode
 - C: Active in read cycle of port mode or in input control mode
 - D: Masked in port mode or output control mode

Figure 14-14. Block Diagram of Type J-1



- Remarks**
1. $mn = 13, 55$
 2. A: Active in output port mode or control mode
B: Masked in cycles other than read cycles

Figure 14-15. Block Diagram of Type J-2

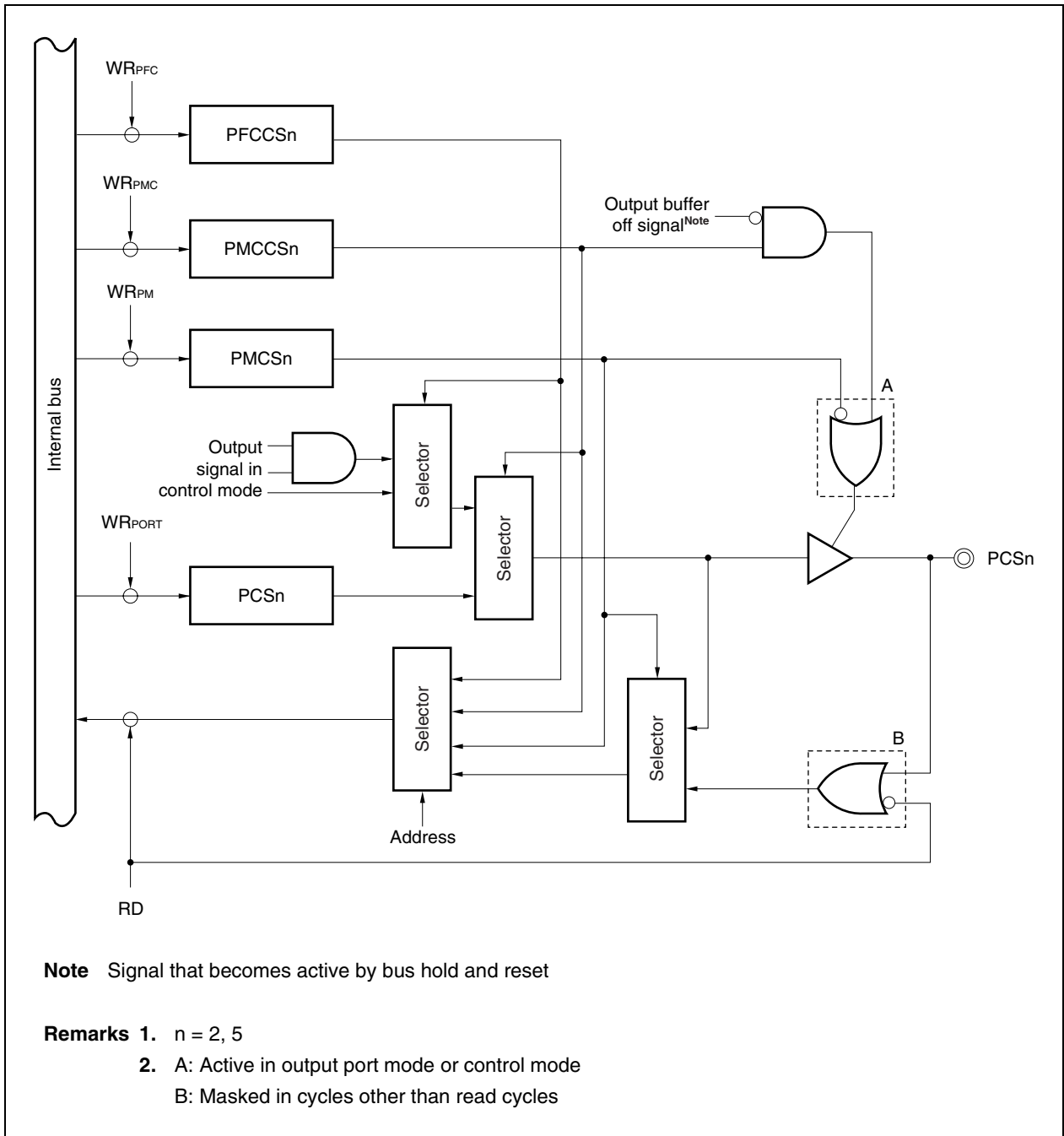


Figure 14-16. Block Diagram of Type J-3

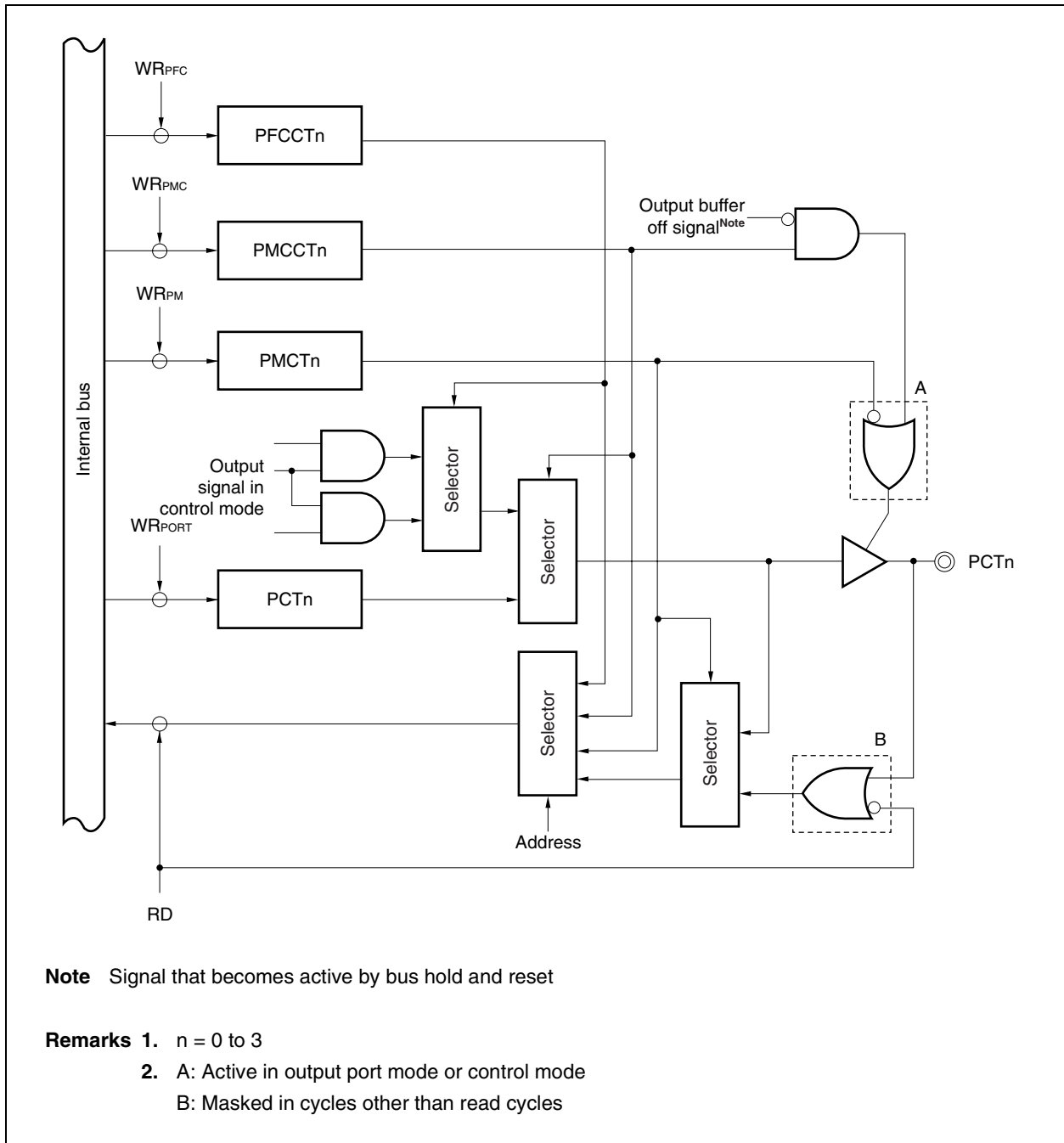


Figure 14-17. Block Diagram of Type L-1

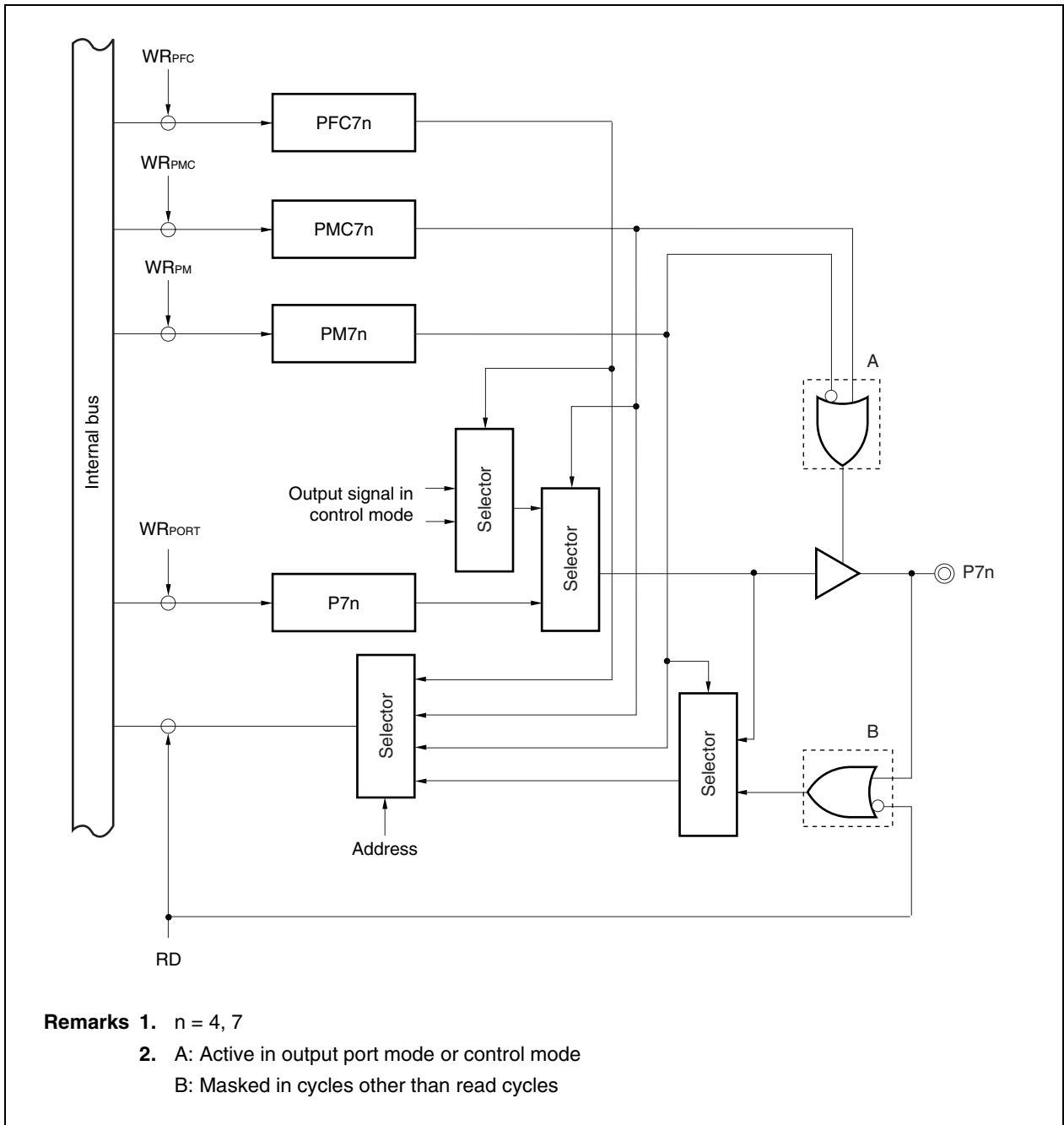


Figure 14-18. Block Diagram of Type L-2

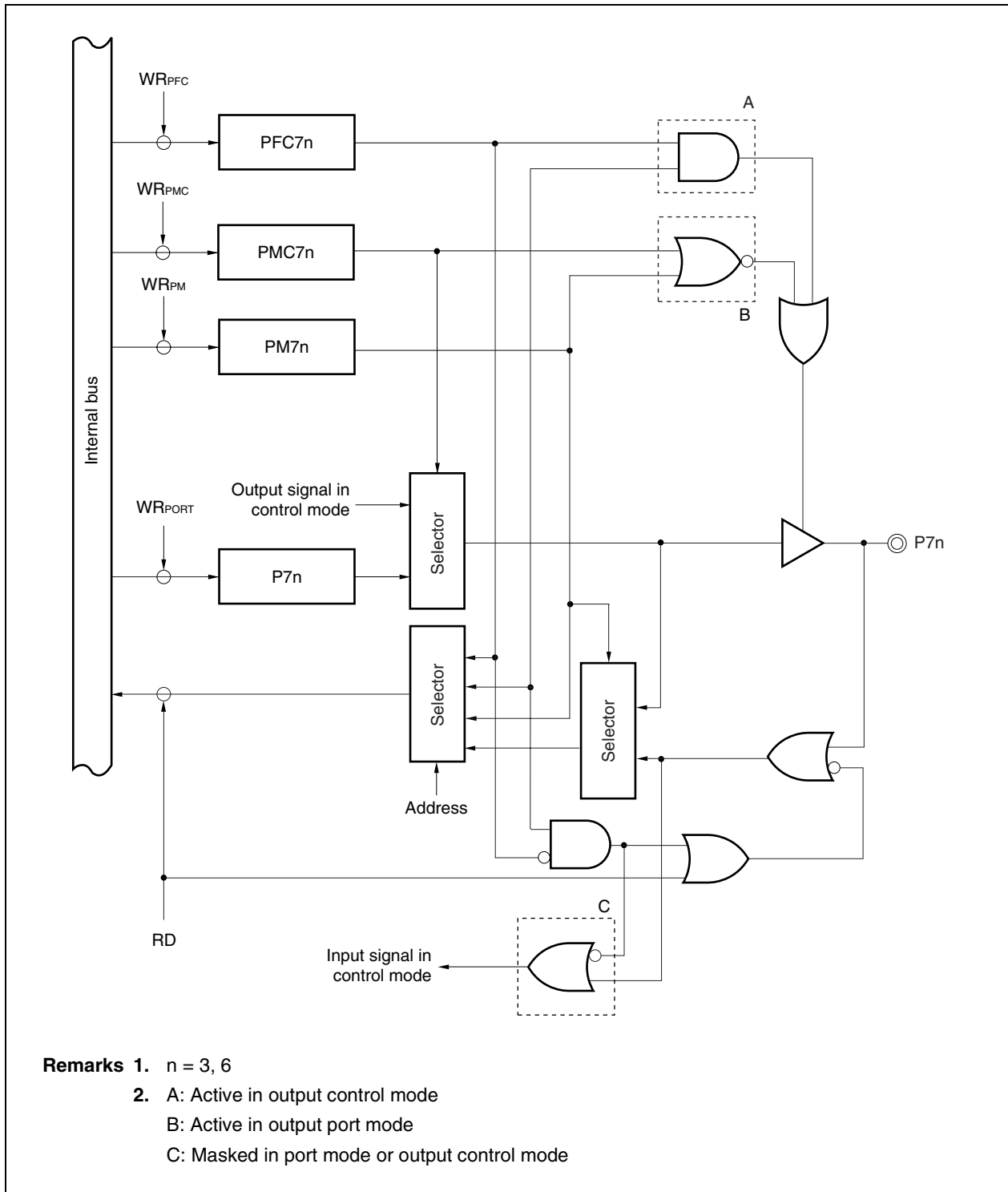


Figure 14-19. Block Diagram of Type L-3

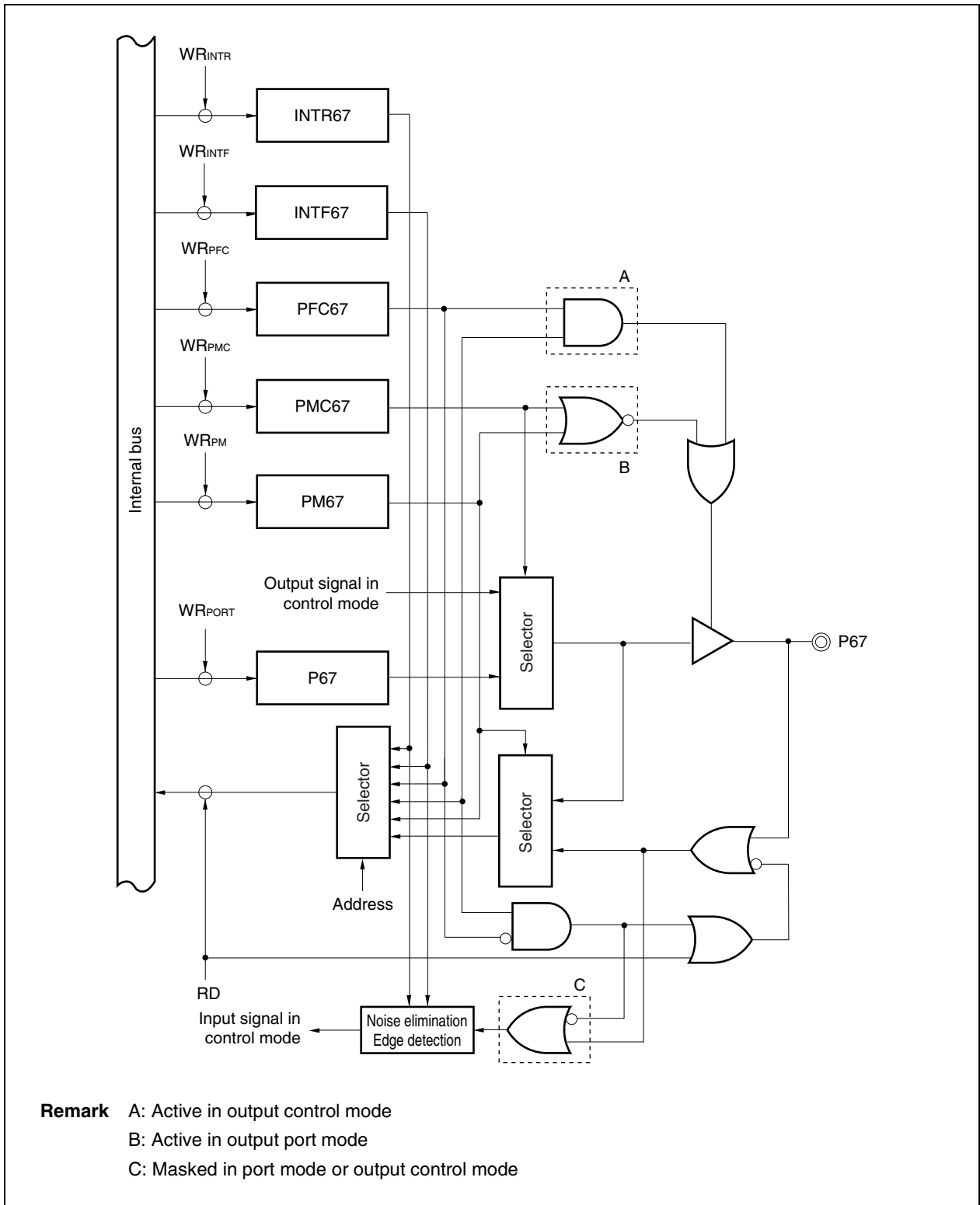


Figure 14-20. Block Diagram of Type L-4

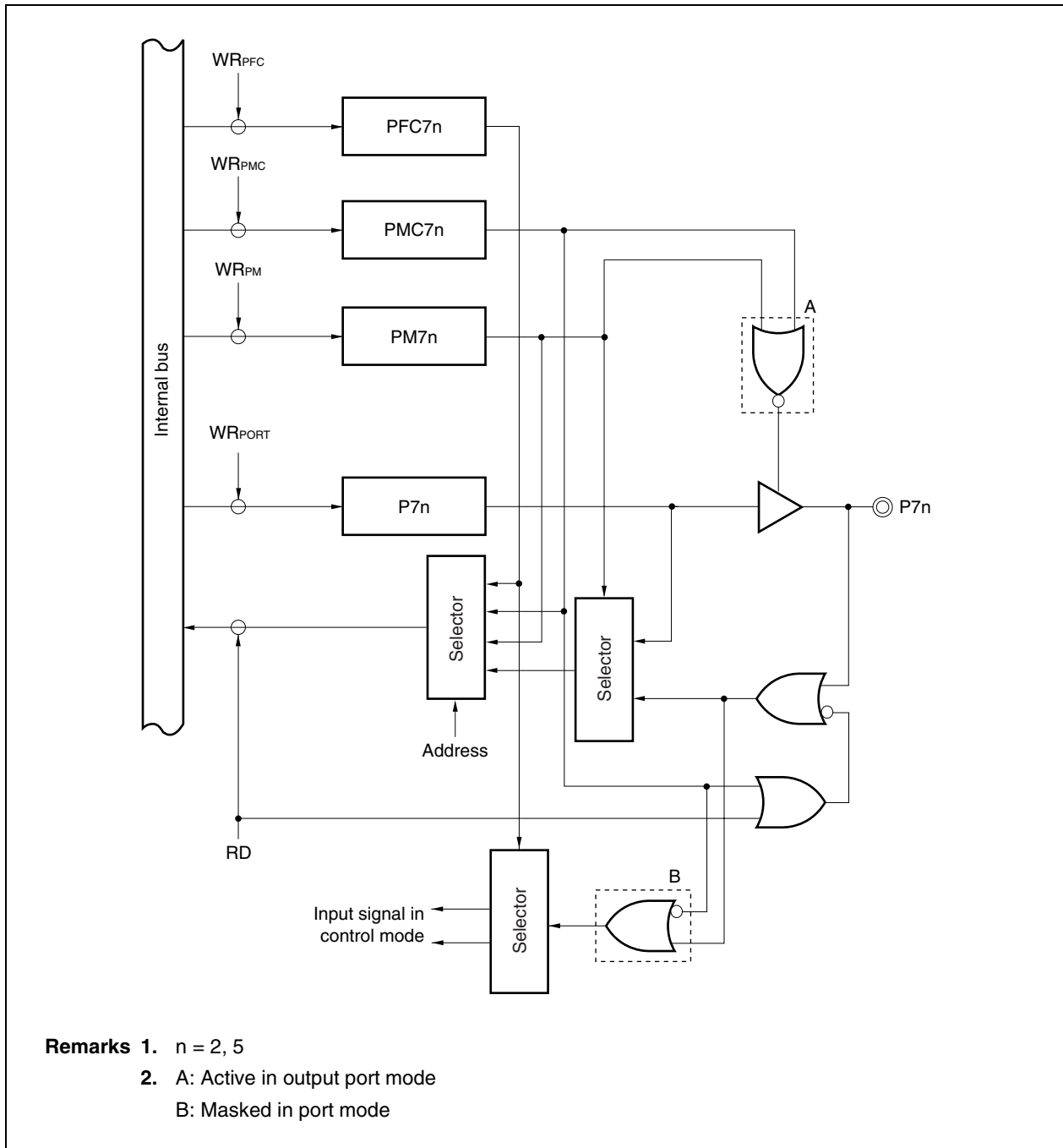


Figure 14-21. Block Diagram of Type L-5

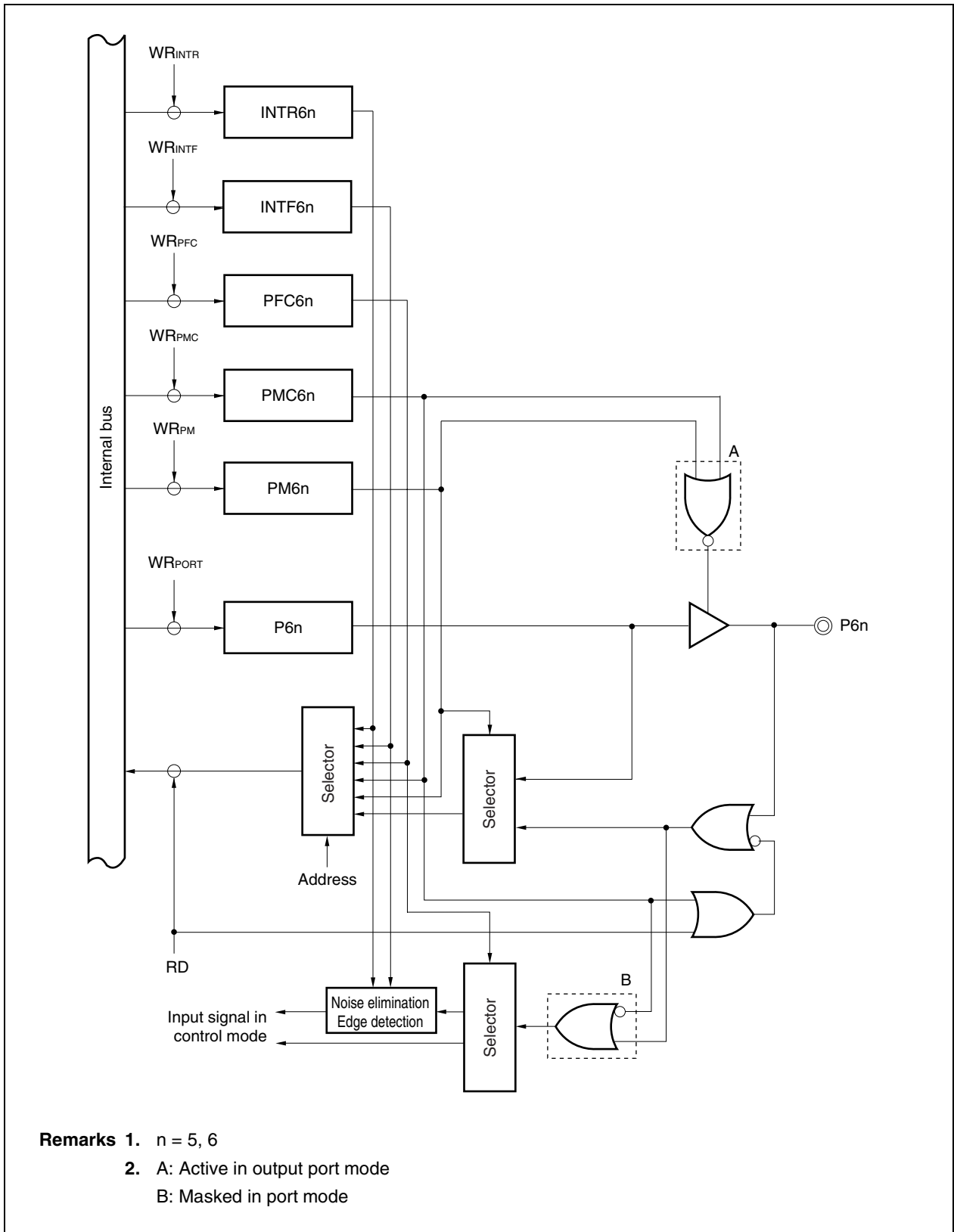


Figure 14-22. Block Diagram of Type M-1

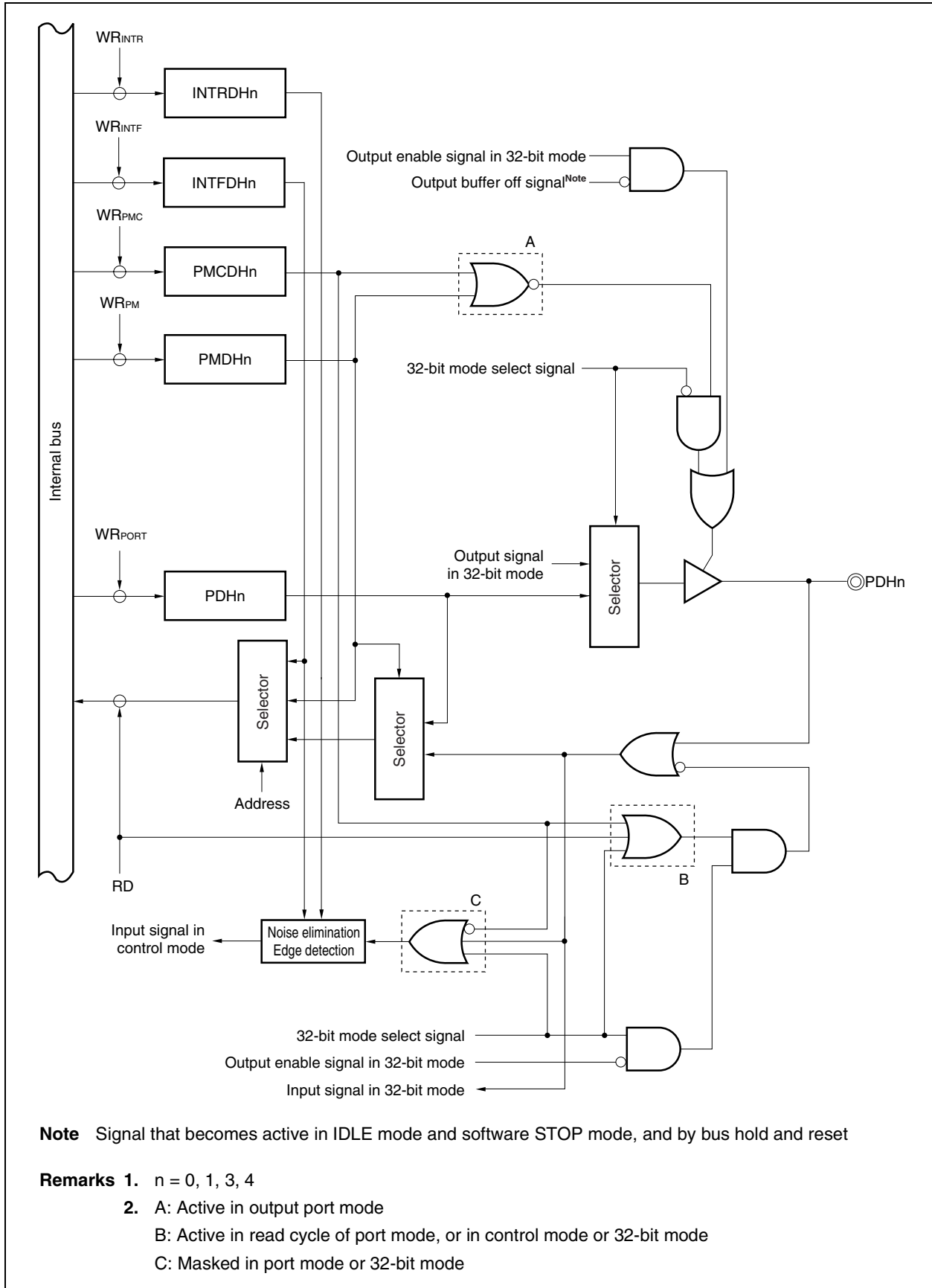
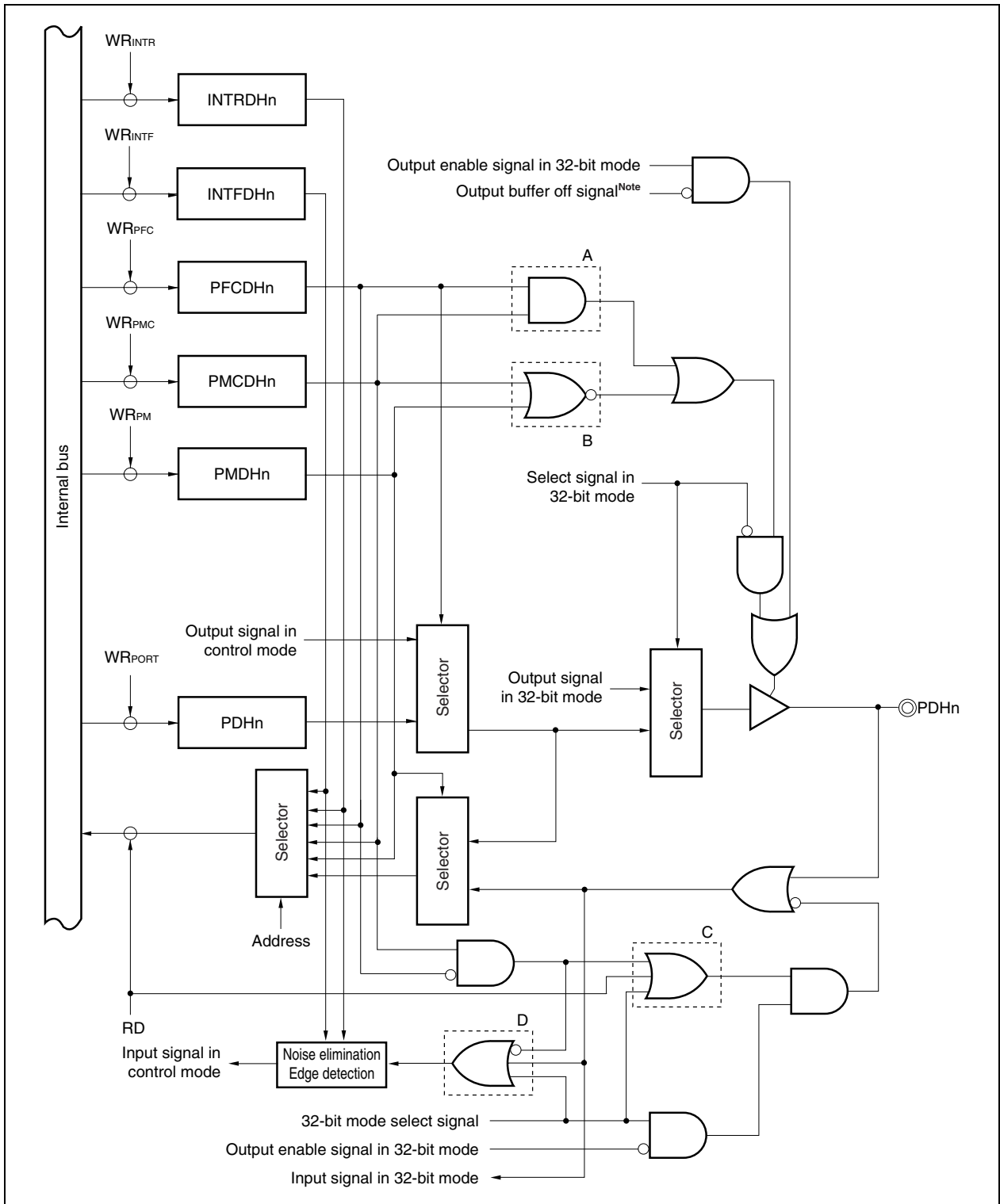


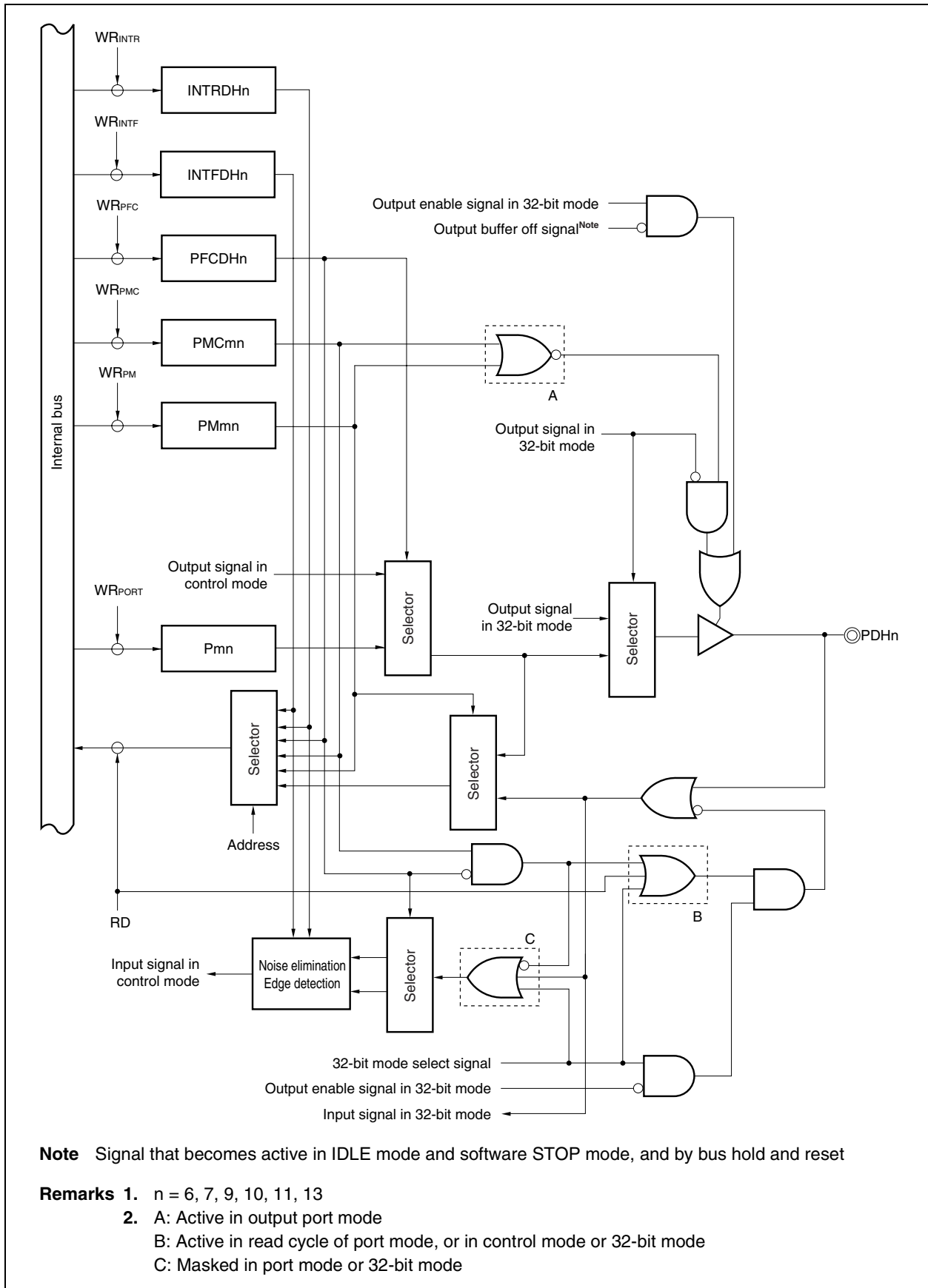
Figure 14-23. Block Diagram of Type M-2



Note Signal that becomes active in IDLE mode and software STOP mode, and by bus hold and reset

- Remarks**
1. $n = 2, 5, 8, 12, 14, 15$
 2. A: Active in output control mode
 B: Active in output port mode
 C: Active in read cycle of port mode, or in input control mode or 32-bit mode
 D: Masked in port mode, input control mode, or 32-bit mode

Figure 14-24. Block Diagram of Type M-3



14.3 Port Pin Functions

14.3.1 Port 1

Port 1 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	
P1	0	0	0	0	P13	P12	P11	P10	FFFFFF402H	Undefined

Bit position	Bit name	Function
3 to 0	P1n (n = 3 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, the port 1 pins can also operate as serial interface I/O (CSI30, UARTB0), USB clock signal input, and external interrupt request input in the control mode.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type	
Port 1	P10	$\overline{\text{INTP10}}/\text{UCLK}$	External interrupt input/USB clock signal input	F-3
	P11	$\overline{\text{INTP11}}/\overline{\text{SCK0}}$	External interrupt request input/ serial interface (CSI30) I/O	H-1
	P12	SI0/RXD0	Serial interface (CSI30) I/O/ serial interface (UARTB0) I/O	F-1
	P13	SO0/TXD0		J-1

(2) I/O mode/control mode setting

The port 1 I/O mode setting is performed by the port 1 mode register (PM1), and the control mode setting is performed by the port 1 mode control register (PMC1) and port 1 function control register (PFC1).

(a) Port 1 mode register (PM1)

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
PM1	1	1	1	1	PM13	PM12	PM11	PM10		FFFFFF422H	FFH

Bit position	Bit name	Function
3 to 0	PM1n (n = 3 to 0)	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 or written in 8-bit or 1-bit units.

	7	6	5	4	3	2	1	0		Address	After reset
PMC1	0	0	0	0	PMC13	PMC12	PMC11	PMC10		FFFFFF442H	00H

Bit position	Bit name	Function
3	PMC13	Specifies operation mode of P13 pin in combination with the PFC1 register. 0: I/O port mode 1: SO0 output mode/TXD0 output mode
2	PMC12	Specifies operation mode of P12 pin in combination with the PFC1 register. 0: I/O port mode 1: SI0 input mode/RXD0 input mode
1	PMC11	Specifies operation mode of P11 pin in combination with the PFC1 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP11}}$) input mode/ $\overline{\text{SCK0}}$ I/O mode
0	PMC10	Specifies operation mode of P10 pin in combination with the PFC1 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP10}}$) input mode/UCLK input mode

(c) Port 1 function control register (PFC1)

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

Caution When the port mode is specified by the port 1 mode control register (PMC1), the setting of this register becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC1	0	0	0	0	PFC13	PFC12	PFC11	PFC10	FFFFFF462H	00H

Bit position	Bit name	Function
3	PFC13	Specifies operation mode of P13 pin in control mode. 0: SO0 output mode 1: TXD0 output mode
2	PFC12	Specifies operation mode of P12 pin in control mode. 0: SI0 input mode 1: RXD0 input mode
1	PFC11	Specifies operation mode of P11 pin in control mode. 0: External interrupt request ($\overline{\text{INPT11}}$) input mode 1: SCK0 I/O mode
0	PFC10	Specifies operation mode of P10 pin in control mode. 0: External interrupt request ($\overline{\text{INPT10}}$) input mode 1: UCLK input mode

(3) Selecting interrupt trigger mode

The valid edges of the $\overline{\text{INTP10}}$ and $\overline{\text{INTP11}}$ pins can be selected by program. The levels to be detected can also be selected.

External interrupt rising edge specification register 1 (INTR1) and external interrupt falling edge specification register 1 (INTF1) are used to specify the valid edge and level detection.

(a) External interrupt rising edge specification register 1 (INTR1) and external interrupt falling edge specification register 1 (INTF1)

These registers are used to specify the trigger mode of the external interrupt requests (INTP10 and INTP11) from external pins.

The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTF10 and INTR10 bits: INTP10
- INTF11 and INTR11 bits: INTP11

The valid edge can be independently selected from the rising edge, falling edge, and both rising and falling edges.

Both the registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode of the $\overline{\text{INTP10}}$ and $\overline{\text{INTP11}}$ pins, set the PMC1 register. If the PMC1 register is set after the INTR1 and INTF1 registers have been set, an illegal interrupt may occur when the PMC1 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR1	0	0	0	0	0	0	INTR11	INTR10	FFFFFFC22H	03H
INTF1	0	0	0	0	0	0	INTF11	INTF10	FFFFFFC02H	00H

Bit position	Bit name	Function															
1, 0	INTF1n, INTR1n (n = 0, 1)	Specify trigger mode of $\overline{\text{INTP10}}$ and $\overline{\text{INTP11}}$ pins. <table border="1"> <thead> <tr> <th>INTF1n</th> <th>INTR1n</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>	INTF1n	INTR1n	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
INTF1n	INTR1n	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 $\overline{\text{INTP1n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P1IFn bit (n = 0, 1). Consequently, even when the CPU acknowledges the interrupt and the P1IFn bit of the interrupt control register (P1ICn) is automatically cleared to 0, the P1IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP1n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P1IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP1n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP1n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP1n) is held pending (n = 0, 1). To not acknowledge the interrupt request of INTP1n, clear the P1IFn bit of the interrupt control register.

14.3.2 Port 2

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

	7	6	5	4	3	2	1	0	Address	After reset
P2	0	0	P25	P24	P23	P22	P21	P20	FFFFF404H	Undefined

Bit position	Bit name	Function
5 to 0	P2n (n = 5 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, the port 2 pins can also operate as the serial interface (CSI31, UARTB1) I/O and external interrupt request input in the control mode.

(1) Operation in control mode

	Port	Alternate Function	Remark	Block Type
Port 2	P20	NMI	Non-maskable interrupt request input	A-1
	P21	$\overline{\text{INTP21}}/\text{RXD1}$	External interrupt request input/ serial interface (UARTB1) I/O	F-4
	P22	$\overline{\text{INTP22}}/\text{TXD1}$		G-2
	P23	$\overline{\text{INTP23}}/\text{SCK1}$	External interrupt request input/ serial interface (CSI31) I/O	H-1
	P24	$\overline{\text{INTP24}}/\text{SI1}$		F-4
	P25	$\overline{\text{INTP25}}/\text{SO1}$		G-2

(2) I/O mode/control mode setting

The port 2 I/O mode setting is performed by the port 2 mode register (PM2), and the control mode setting is performed by 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 or written in 8-bit or 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
PM2	1	1	PM25	PM24	PM23	PM22	PM21	1	FFFFF424H	FFH

Bit position	Bit name	Function
5 to 1	PM2n (n = 5 to 1)	Specifies input/output mode for P2n 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 or written in 8-bit or 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
PMC2	0	0	PMC25	PMC24	PMC23	PMC22	PMC21	1	FFFFF444H	01H

Bit position	Bit name	Function
5	PMC25	Specifies operation mode of P25 pin in combination with the PFC2 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP25}}$) input mode/SO1 output mode
4	PMC24	Specifies operation mode of P24 pin in combination with the PFC2 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP24}}$) input mode/SI1 input mode
3	PMC23	Specifies operation mode of P23 pin in combination with the PFC2 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP23}}$) input mode/ $\overline{\text{SCK1}}$ I/O mode
2	PMC22	Specifies operation mode of P22 pin in combination with the PFC2 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP22}}$) input mode/TXD1 output mode
1	PMC21	Specifies operation mode of P21 pin in combination with the PFC2 register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP21}}$) input mode/RXD1 input mode

(c) Port 2 function control register (PFC2)

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

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	PFC25	PFC24	PFC23	PFC22	PFC21	0	FFFFF464H	00H

Bit position	Bit name	Function
5	PFC25	Specifies operation mode of P25 pin in control mode. 0: External interrupt request ($\overline{\text{INTP25}}$) input mode 1: SO1 output mode
4	PFC24	Specifies operation mode of P24 pin in control mode. 0: External interrupt request ($\overline{\text{INTP24}}$) input mode 1: SI1 input mode
3	PFC23	Specifies operation mode of P23 pin in control mode. 0: External interrupt request ($\overline{\text{INTP23}}$) input mode 1: $\text{SCK}\bar{1}$ I/O mode
2	PFC22	Specifies operation mode of P22 pin in control mode. 0: External interrupt request ($\overline{\text{INTP22}}$) input mode 1: TXD1 output mode
1	PFC21	Specifies operation mode of P21 pin in control mode. 0: External interrupt request ($\overline{\text{INTP21}}$) input mode 1: RXD1 input mode

(3) Selecting interrupt trigger mode

The valid edges of the $\overline{\text{INTP2n}}$ and NMI pins can be selected by program ($n = 1$ to 5). The level detection of the $\overline{\text{INTP2n}}$ pin can also be selected.

External interrupt rising edge specification register 2 (INTR2) and external interrupt falling edge specification register 2 (INTF2) are used to specify the valid edge and level detection.

(a) External interrupt rising edge specification register 2 (INTR2) and external interrupt falling edge specification register 2 (INTF2)

These registers are used to specify the trigger mode of an external interrupt request (INTP2n) from an external pin and the non-maskable interrupt (NMI) ($n = 1$ to 5). The correspondence between each bit of this register and the external interrupt request and non-maskable interrupt controlled by that bit is as follows.

- NMIF0 and NMIR0 bits: NMI
- INTF21 and INTR21 bits: INTP21
- INTF22 and INTR22 bits: INTP22
- INTF23 and INTR23 bits: INTP23
- INTF24 and INTR24 bits: INTP24
- INTF25 and INTR25 bits: INTP25

The rising edge, falling edge, or both the rising and falling edges can be specified as the valid edge of the $\overline{\text{INTP2n}}$ pin and the rising or falling edge can be specified as the valid edge of the NMI pin, independently for each pin.

Both the registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode of the $\overline{\text{INTP2n}}$ pin, set the PMC2 register ($n = 1$ to 5). If the PMC2 register is set after the INTR2 and INTF2 registers have been set, an illegal interrupt may occur when the PMC2 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR2	0	0	INTR25	INTR24	INTR23	INTR22	INTR21	NMIR0	FFFFFFC24H	3FH
INTF2	0	0	INTF25	INTF24	INTF23	INTF22	INTF21	NMIF0	FFFFFFC04H	00H

Bit position	Bit name	Function															
5 to 1	INTF2n, INTR2n (n = 1 to 5)	Specify trigger mode of $\overline{\text{INTP2n}}$ pin. <table border="1"> <thead> <tr> <th>INTF2n</th> <th>INTR2n</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>	INTF2n	INTR2n	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
INTF2n	INTR2n	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															
0	NMIF0, NMIR0	Specify trigger mode of NMI pin. <table border="1"> <thead> <tr> <th>NMIF0</th> <th>NMIR0</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>Setting prohibited</td> </tr> <tr> <td>1</td> <td>1</td> <td>Both rising and falling edges</td> </tr> </tbody> </table>	NMIF0	NMIR0	Operation	0	0	Falling edge	0	1	Rising edge	1	0	Setting prohibited	1	1	Both rising and falling edges
NMIF0	NMIR0	Operation															
0	0	Falling edge															
0	1	Rising edge															
1	0	Setting prohibited															
1	1	Both rising and falling edges															

- Notes 1.** The level of the $\overline{\text{INTP2n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P2IFn bit (n = 1 to 5). Consequently, even when the CPU acknowledges the interrupt and the P2IFn bit of the interrupt control register (P2ICn) is automatically cleared to 0, the P2IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP2n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P2IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP2n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP2n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP2n) is held pending (n = 1 to 5). To not acknowledge the interrupt request of INTP2n, clear the P2IFn bit of the interrupt control register.

14.3.3 Port 5

Port 5 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	
P5	0	0	P55	P54	P53	P52	P51	P50	FFFFF40AH	Undefined

Bit position	Bit name	Function
5 to 0	P5n (n = 5 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, the port 5 pins can also operate as the DMA controller I/O, real-time pulse unit (RPU) I/O, and external interrupt request input in the control mode.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type	
Port 5	P50	$\overline{\text{INTP50}}/\overline{\text{DMARQ0}}$	External interrupt request input/DMA request input	F-3
	P51	$\overline{\text{INTP51}}/\overline{\text{DMAAK0}}$	External interrupt request input/ DMA acknowledge signal output	G-2
	P52	$\overline{\text{INTP52}}/\overline{\text{TC0}}$	External interrupt request input/DMA end signal output	G-2
	P53	$\overline{\text{INTPC00}}/\overline{\text{TIC0}}/\overline{\text{DMARQ1}}$	External interrupt request and timer C0 external capture trigger input/real-time pulse unit (RPU) input/ DMA request input	F-2
	P54	$\overline{\text{INTPC01}}/\overline{\text{DMAAK1}}$	External interrupt request and timer C0 external capture trigger input/DMA acknowledge signal output	G-1
	P55	$\overline{\text{TOC0}}/\overline{\text{TC1}}$	Real-time pulse unit (RPU) output/ DMA end signal output	J-1

(2) I/O mode/control mode setting

The port 5 I/O mode setting is performed by the port 5 mode register (PM5), and the control mode setting is performed by the port 5 mode control register (PMC5) and port 5 function control register (PFC5).

(a) Port 5 mode register (PM5)

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
PM5	1	1	PM55	PM54	PM53	PM52	PM51	PM50	FFFFFF42AH	FFH

Bit position	Bit name	Function
5 to 0	PM5n (n = 5 to 0)	Specifies input/output mode for P5n pin. 0: Output mode (output buffer on) 1: Input mode (output buffer off)

(b) Port 5 mode control register (PMC5)

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
PMC5	0	0	PMC55	PMC54	PMC53	PMC52	PMC51	PMC50	FFFFFF44AH	00H

Bit position	Bit name	Function
5	PMC55	Specifies operation mode of P55 pin in combination with the PFC5 register. 0: I/O port mode 1: TOC0 output mode/DMA end signal ($\overline{TC1}$) output mode
4	PMC54	Specifies operation mode of P54 pin in combination with the PFC5 register. 0: I/O port mode 1: External interrupt request and timer C0 external capture trigger (INTPC01) input mode/DMA acknowledge signal ($\overline{DMAAK1}$) output mode
3	PMC53	Specifies operation mode of P53 pin in combination with the PFC5 register. 0: I/O port mode 1: External interrupt request and timer C0 external capture trigger (INTPC00) input mode/TIC0 input mode/DMA request ($\overline{DMARQ1}$) input mode
2	PMC52	Specifies operation mode of P52 pin in combination with the PFC5 register. 0: I/O port mode 1: External interrupt request ($\overline{INTP52}$) input mode/DMA end signal ($\overline{TC0}$) output mode
1	PMC51	Specifies operation mode of P51 pin in combination with the PFC5 register. 0: I/O port mode 1: External interrupt request ($\overline{INTP51}$) input mode/DMA acknowledge signal ($\overline{DMAAK0}$) output mode
0	PMC50	Specifies operation mode of P50 pin in combination with the PFC5 register. 0: I/O port mode 1: External interrupt request ($\overline{INTP50}$) input mode/DMA request ($\overline{DMARQ0}$) input mode

(c) Port 5 function control register (PFC5)

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

Caution When the port mode is specified by the port 5 mode control register (PMC5), the setting of this register becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC5	0	0	PFC55	PFC54	PFC53	PFC52	PFC51	PFC50	FFFFFF46AH	00H

Bit position	Bit name	Function
5	PFC55	Specifies operation mode of P55 pin in control mode. 0: TOC0 output mode 1: DMA end signal ($\overline{TC1}$) output mode
4	PFC54	Specifies operation mode of P54 pin in control mode. 0: External interrupt request and timer C0 external capture trigger (INTPC01) input mode 1: DMA acknowledge signal ($\overline{DMAAK1}$) output
3	PFC53	Specifies operation mode of P53 pin in control mode. 0: External interrupt request and timer C0 external capture trigger (INTPC01) input mode/TIC0 input mode 1: DMA request ($\overline{DMARQ1}$) input mode There is no register that selects an external interrupt request and timer C0 external capture trigger (INTPC00) input mode or TIC0 input mode. <ul style="list-style-type: none"> • To use TIC0 input mode: Mask the external interrupt request and external capture trigger (INTPC00) of timer C0, or use the CCC00 register as a compare register. • To use external interrupt request and external capture trigger (INTPC00) of timer C0: Clear the ETIC0 bit of the TMCC01 register to 0.
2	PFC52	Specifies operation mode of P52 pin. 0: External interrupt request ($\overline{INTP52}$) input mode 1: DMA end signal ($\overline{TC0}$) output mode
1	PFC51	Specifies operation mode of P51 pin. 0: External interrupt request ($\overline{INTP51}$) input mode 1: DMA acknowledge signal ($\overline{DMAAK0}$) output
0	PFC50	Specifies operation mode of P50 pin in control mode. 0: External interrupt request ($\overline{INTP50}$) input mode 1: DMA request ($\overline{DMARQ0}$) input mode

(3) Selecting interrupt trigger mode

The valid edges of the $\overline{\text{INTP5n}}$ pin can be selected by program ($n = 0$ to 2). The level detection of the $\overline{\text{INTP5n}}$ pin can also be selected.

External interrupt rising edge specification register 5 (INTR5) and external interrupt falling edge specification register 5 (INTF5) are used to specify the valid edge and level detection.

(a) External interrupt rising edge specification register 5 (INTR5) and external interrupt falling edge specification register 5 (INTF5)

These registers are used to specify the trigger mode of an external interrupt request ($\overline{\text{INTP5n}}$) from an external pin ($n = 0$ to 2). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTF50 and INTR50 bits: INTP50
- INTF51 and INTR51 bits: INTP51
- INTF52 and INTR52 bits: INTP52

The rising edge, falling edge, or both the rising and falling edges can be specified as the valid edge of the $\overline{\text{INTP5n}}$ pin, independently for each pin.

Both the registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMC5 register.

If the PMC5 register is set after the INTR5 and INTF5 registers have been set, an illegal interrupt may occur when the PMC5 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR5	0	0	0	0	0	INTR52	INTR51	INTR50	FFFFFFC2AH	07H
INTF5	0	0	0	0	0	INTF52	INTF51	INTF50	FFFFFFC0AH	00H

Bit position	Bit name	Function															
2 to 0	INTF5n, INTR5n (n = 0 to 2)	Specify trigger mode of $\overline{\text{INTP5n}}$ pin. <table border="1"> <thead> <tr> <th>INTF5n</th> <th>INTR5n</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>	INTF5n	INTR5n	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
INTF5n	INTR5n	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 $\overline{\text{INTP5n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P5IFn bit (n = 0 to 2). Consequently, even when the CPU acknowledges the interrupt and the P5IFn bit of the interrupt control register (P5ICn) is automatically cleared to 0, the P5IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP5n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P5IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP5n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP5n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP5n) is held pending (n = 0 to 2). To not acknowledge the interrupt request of INTP5n, clear the P5IFn bit of the interrupt control register.

14.3.4 Port 6

Port 6 is a 3-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
P6	P67	P66	P65	0	0	0	0	0	FFFFF40CH	Undefined

Bit position	Bit name	Function
7 to 5	P6n (n = 7 to 5)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, the port 6 pins can also operate as the real-time pulse unit (RPU) I/O and external interrupt request input in the control mode.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type
Port 6	P65	$\overline{\text{INTP65}}/\text{INTPC10}/\text{TIC1}$	External interrupt request and timer C1 external capture trigger input/real-time pulse unit (RPU) input
	P66	$\overline{\text{INTP66}}/\text{INTPC11}$	External interrupt request and timer C1 external capture trigger input
P67	$\overline{\text{INTP67}}/\text{TOC1}$	External interrupt request input/real-time pulse unit (PRU) output	L-3

(2) I/O mode/control mode setting

The port 6 I/O mode setting is performed by the port 6 mode register (PM6), and the control mode setting is performed by the port 6 mode control register (PMC6) and port 6 function control register (PFC6).

(a) Port 6 mode register (PM6)

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
PM6	PM67	PM66	PM65	1	1	1	1	1		FFFFFF42CH	FFH

Bit position	Bit name	Function
7 to 5	PM6n (n = 7 to 5)	Specifies input/output mode for P6n pin. 0: Output mode (output buffer on) 1: Input mode (output buffer off)

(b) Port 6 mode control register (PMC6)

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
PMC6	PMC67	PMC66	PMC65	0	0	0	0	0		FFFFFF44CH	00H

Bit position	Bit name	Function
7	PMC67	Specifies operation mode of P67 pin. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP67}}$) input mode/TOC1 output mode
6	PMC66	Specifies operation mode of P66 pin. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP66}}$) input mode/external interrupt request and timer C1 external capture trigger (INTPC11) input mode
5	PMC65	Specifies operation mode of P65 pin. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTP65}}$) input mode/external interrupt request and timer C1 external capture trigger (INTPC10) input mode/TIC1 input mode

(c) Port 6 function control register (PFC6)

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

Caution When the port mode is specified by the port 6 mode control register (PMC6), the setting of this register becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC6	PFC67	PFC66	PFC65	0	0	0	0	0	FFFFFF46CH	00H

Bit position	Bit name	Function
7	PFC67	Specifies operation mode of P67 pin in control mode. 0: External interrupt request (INTP67) input mode 1: TOC1 output mode
6	PFC66	Specifies operation mode of P66 pin in control mode. 0: External interrupt request (INTP66) input mode 1: External interrupt request and timer C1 external capture trigger (INTPC11) input mode
5	PFC65	Specifies operation mode of P65 pin in control mode. 0: External interrupt request (INTP65) input mode 1: External interrupt request and timer C1 external capture trigger (INTPC10) input mode/TIC1 input mode There is no register that selects an external interrupt request and timer C1 external capture trigger (INTPC10) input mode or TIC1 input mode. <ul style="list-style-type: none"> To use TIC1 input mode: Mask the external interrupt request and external capture trigger (INTPC10) of timer C1, or use the CCC10 register as a compare register. To use external interrupt request and external capture trigger (INTPC10) of timer C1: Clear the ETIC1 bit of the TMCC11 register to 0.

(3) Selecting interrupt trigger mode

The valid edges of the $\overline{\text{INTP6n}}$ pin can be selected by program ($n = 5$ to 7). The level detection of the $\overline{\text{INTP6n}}$ pin can also be selected.

External interrupt rising edge specification register 6 (INTR6) and external interrupt falling edge specification register 6 (INTF6) are used to specify the valid edge and level detection.

(a) External interrupt rising edge specification register 6 (INTR6) and external interrupt falling edge specification register 6 (INTF6)

These registers are used to specify the trigger mode of an external interrupt request (INTP6n) from an external pin ($n = 5$ to 7). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTF65 and INTR65 bits: INTP65
- INTF66 and INTR66 bits: INTP66
- INTF67 and INTR67 bits: INTP67

The rising edge, falling edge, or both the rising and falling edges can be specified as the valid edge of the $\overline{\text{INTP6n}}$ pin, independently for each pin.

Both the registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMC6 register.

If the PMC6 register is set after the INTR6 and INTF6 registers have been set, an illegal interrupt may occur when the PMC6 register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTR6	INTR67	INTR66	INTR65	0	0	0	0	0	FFFFFFC2CH	E0H
INTF6	INTF67	INTF66	INTF65	0	0	0	0	0	FFFFFFC0CH	00H

Bit position	Bit name	Function															
7 to 5	INTF6n, INTR6n (n = 7 to 5)	Specify trigger mode of $\overline{\text{INTP6n}}$ pin. <table border="1"> <thead> <tr> <th>INTF6n</th> <th>INTR6n</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>	INTF6n	INTR6n	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
INTF6n	INTR6n	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 $\overline{\text{INTP6n}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the P6IFn bit (n = 5 to 7). Consequently, even when the CPU acknowledges the interrupt and the P6IFn bit of the interrupt control register (P6ICn) is automatically cleared to 0, the P6IFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTP6n}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the P6IFn bit to 0.
- 2.** If a level-detected interrupt request (INTP6n) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTP6n) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTP6n) is held pending (n = 5 to 7). To not acknowledge the interrupt request of INTP6n, clear the P6IFn bit of the interrupt control register.

14.3.5 Port 7

Port 7 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
P7	P77	P76	P75	P74	P73	P72	0	0	FFFFF40EH	Undefined

Bit position	Bit name	Function
7 to 2	P7n (n = 7 to 2)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, the port 7 pins can also operate as the DMA controller I/O, real-time pulse unit (RPU) I/O, and external interrupt request input in the control mode.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type	
Port 7	P72	INTPC20/TIC2/ $\overline{\text{DMARQ2}}$	External interrupt request and timer C2 external capture trigger input/real-time pulse unit (RPU) input/ DMA request input	L-4
	P73	INTPC21/ $\overline{\text{DMAAK2}}$	External interrupt request and timer C2 external capture trigger input/DMA acknowledge signal output	L-2
	P74	$\overline{\text{TOC2/TC2}}$	Real-time pulse unit (RPU) output/ DMA end signal output	L-1
	P75	INTPC30/TIC3/ $\overline{\text{DMARQ3}}$	External interrupt request and timer C3 external capture trigger input/real-time pulse unit (RPU) input/ DMA request input	L-4
	P76	INTPC31/ $\overline{\text{DMAAK3}}$	External interrupt request and timer C3 external capture trigger input/DMA acknowledge signal output	L-2
	P77	$\overline{\text{TOC3/TC3}}$	Real-time pulse unit (RPU) output/ DMA end signal output	L-1

(2) I/O mode/control mode setting

The port 7 I/O mode setting is performed by the port 7 mode register (PM7), and the control mode setting is performed by the port 7 mode control register (PMC7) and port 7 function control register (PFC7).

(a) Port 7 mode register (PM7)

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
PM7	PM77	PM76	PM75	PM74	PM73	PM72	1	1	FFFFFF42EH	FFH

Bit position	Bit name	Function
7 to 2	PM7n (n = 7 to 2)	Specifies input/output mode for P7n pin. 0: Output mode (output buffer on) 1: Input mode (output buffer off)

(b) Port 7 mode control register (PMC7)

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
PMC7	PMC77	PMC76	PMC75	PMC74	PMC73	PMC72	0	0	FFFFFF44EH	00H

Bit position	Bit name	Function
7	PMC77	Specifies operation mode of P77 pin in combination with the PFC7 register. 0: I/O port mode 1: TOC3 output mode/DMA end signal ($\overline{TC3}$) output mode
6	PMC76	Specifies operation mode of P76 pin in combination with the PFC7 register. 0: I/O port mode 1: External interrupt request and time C3 external capture trigger (INTPC31) input mode/DMA acknowledge signal ($\overline{DMAAK3}$) output mode
5	PMC75	Specifies operation mode of P75 pin in combination with the PFC7 register. 0: I/O port mode 1: External interrupt request and timer C3 external capture trigger (INTPC30) input mode/TIC3 input mode/DMA request ($\overline{DMARQ3}$) input mode
4	PMC74	Specifies operation mode of P74 pin in combination with the PFC7 register. 0: I/O port mode 1: TOC2 output mode/DMA end signal ($\overline{TC2}$) output mode
3	PMC73	Specifies operation mode of P73 pin in combination with the PFC7 register. 0: I/O port mode 1: External interrupt request and time C2 external capture trigger (INTPC21) input mode/DMA acknowledge signal ($\overline{DMAAK2}$) output mode
2	PMC72	Specifies operation mode of P72 pin in combination with the PFC7 register. 0: I/O port mode 1: External interrupt request and timer C2 external capture trigger (INTPC20) input mode/TIC2 input mode/DMA request ($\overline{DMARQ2}$) input mode

(c) Port 7 function control register (PFC7)

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

Caution When the port mode is specified by the port 7 mode control register (PMC7), the setting of this register becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFC7	PFC77	PFC76	PFC75	PFC74	PFC73	PFC72	0	0	FFFFF46EH	00H

Bit position	Bit name	Function
7	PFC77	Specifies operation mode of P77 pin in control mode. 0: TOC3 output mode 1: DMA end signal ($\overline{TC3}$) output mode
6	PFC76	Specifies operation mode of P76 pin in control mode. 0: External interrupt request and timer C3 external capture trigger (INTPC31) input mode 1: DMA acknowledge signal ($\overline{DMAAK3}$) output
5	PFC75	Specifies operation mode of P75 pin in control mode. 0: External interrupt request and timer C3 external capture trigger (INTPC30) input mode/TIC3 input mode 1: DMA request ($\overline{DMARQ3}$) input mode There is no register that selects an external interrupt request and timer C3 external capture trigger (INTPC30) input mode or TIC3 input mode. <ul style="list-style-type: none"> • To use TIC3 input mode: Mask the external interrupt request and external capture trigger (INTPC30) of timer C3, or use the CCC30 register as a compare register. • To use external interrupt request and external capture trigger (INTPC30) of timer C3: Clear the ETIC3 bit of the TMCC31 register to 0.
4	PFC74	Specifies operation mode of P74 pin. 0: TOC2 output mode 1: DMA end signal ($\overline{TC2}$) output mode
3	PFC73	Specifies operation mode of P73 pin. 0: External interrupt request and timer C2 external capture trigger (INTPC21) input mode 1: DMA acknowledge signal ($\overline{DMAAK2}$) output
2	PFC72	Specifies operation mode of P72 pin in control mode. 0: External interrupt request and timer C2 external capture trigger (INTPC20) input mode/TIC2 input mode 1: DMA request ($\overline{DMARQ2}$) input mode There is no register that selects an external interrupt request and timer C2 external capture trigger (INTPC20) input mode or TIC2 input mode. <ul style="list-style-type: none"> • To use TIC2 input mode: Mask the external interrupt request and external capture trigger (INTPC20) of timer C2, or use the CCC20 register as a compare register. • To use external interrupt request and external capture trigger (INTPC20) of timer C2: Clear the ETIC2 bit of the TMCC21 register to 0.

14.3.6 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.

When the higher 8 bits of port AL are used as port ALH (PALH) and the lower 8 bits as port ALL (PALL), port AL becomes two 8-bit ports that can be set in the input or output mode in 1-bit units.

	15	14	13	12	11	10	9	8		Address	After reset
PAL	0	0	0	0	0	0	0	0		FFFFFF01H	Undefined
	7	6	5	4	3	2	1	0		Address	
	0	0	0	0	0	0	PAL1	PAL0		FFFFFF00H	

Bit position	Bit name	Function
1, 0	PALn (n = 1, 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

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 and external interrupt request input.

(1) Operation in control mode

	Port	Alternate Function	Remark	Block Type
Port AL	PAL0, PAL1	INTPL0/A0, INTPL1/A1	Address bus when memory expanded/ external interrupt request input	G-3

(2) I/O mode/control mode setting

The port AL I/O mode setting is performed by the port AL mode register (PMAL), and control mode setting is performed by port AL mode control register L (PMCALL) and the port AL function control register (PFCAL).

★

(a) Port AL mode register (PMAL)

The port AL mode register (PMAL) can be read or written in 16-bit units.

When using the higher 8 bits of PMAL as port AL mode register H (PMALH) and the lower 8 bits as port AL mode register L (PMALL), the PMALH register is read-only in 8-bit or 1-bit units, and the PMALL register can be read or written in 8-bit or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMAL	1	1	1	1	1	1	1	1	FFFFFF021H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	1	1	1	1	1	1	PMAL1	PMAL0	FFFFFF020H	

Bit position	Bit name	Function
1, 0	PMALn (n = 1, 0)	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 or written in 16-bit units.

When using the higher 8 bits of PMCAL as port AL mode control register H (PMCALH) and the lower 8 bits as port AL mode control register L (PMCALL), the PMCALH register is read-only in 8-bit or 1-bit units, and the PMCALL register can be read or written in 8-bit or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMCAL	0	0	0	0	0	0	0	0	FFFFFF041H	0002H
	7	6	5	4	3	2	1	0	Address	
	0	0	0	0	0	0	PMCAL1	PMCAL0	FFFFFF040H	

Bit position	Bit name	Function
1	PMCAL1	Specifies operation mode of PAL1 pin. 0: I/O port mode 1: INTPL1 input mode/A1 output mode
0	PMCAL0	Specifies operation mode of PAL0 pin. 0: I/O port mode 1: INTPL0 input mode/A0 output mode

★ (c) Port AL function control register L (PFCALL)

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

Caution When the port mode is specified by the port AL mode control register (PMCAL), the setting of this register becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFCALL	0	0	0	0	0	0	PFCAL1	PFCAL0	FFFFFF058H	03H

Bit position	Bit name	Function
1	PFCAL1	Specifies operation mode of PAL1 pin in control mode. 0: External interrupt request ($\overline{\text{INTPL1}}$) input mode 1: A1 output mode
0	PFCAL0	Specifies operation mode of PAL0 pin in control mode. 0: External interrupt request ($\overline{\text{INTPL0}}$) input mode 1: A0 output mode

(3) Selecting interrupt trigger mode

The valid edges of the $\overline{\text{INTPLn}}$ pin can be selected by program ($n = 0, 1$). The level detection of the $\overline{\text{INTPLn}}$ pin can also be selected.

External interrupt rising edge specification register AL (INTRAL) and external interrupt falling edge specification register AL (INTFAL) are used to specify the valid edge and level detection.

(a) External interrupt rising edge specification register AL (INTRAL) and external interrupt falling edge specification register AL (INTFAL)

These registers are used to specify the trigger mode of an external interrupt request ($\overline{\text{INTPLn}}$) from an external pin ($n = 0, 1$). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTFAL0 and INTRAL0 bits: INTPAL0
- INTFAL1 and INTRAL1 bits: INTPAL1

The rising edge, falling edge, or both the rising and falling edges can be specified as the valid edge of the $\overline{\text{INTPLn}}$ pin, independently for each pin.

Both the registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMCAL register.

If the PMCAL register is set after the INTRAL and INTFAL registers have been set, an illegal interrupt may occur when the PMCAL register is set.

	7	6	5	4	3	2	1	0	Address	After reset
INTRAL	0	0	0	0	0	0	INTRAL1	INTRAL0	FFFFFFC30H	03H
INTFAL	0	0	0	0	0	0	INTFAL1	INTFAL0	FFFFFFC10H	00H

Bit position	Bit name	Function															
1, 0	INTFALn, INTRALn (n = 0, 1)	Specify trigger mode of $\overline{\text{INTPLn}}$ pin. <table border="1"> <thead> <tr> <th>INTFALn</th> <th>INTRALn</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>	INTFALn	INTRALn	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
INTFALn	INTRALn	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 $\overline{\text{INTPLn}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the PLIFn bit (n = 0, 1). Consequently, even when the CPU acknowledges the interrupt and the PLIFn bit of the interrupt control register (PLICn) is automatically cleared to 0, the PLIFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTPLn}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the PLIFn bit to 0.
- 2.** If a level-detected interrupt request (INTPLn) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTPLn) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTPLn) is held pending (n = 0, 1). To not acknowledge the interrupt request of INTPLn, clear the PLIFn bit of the interrupt control register.

14.3.7 Port AH

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

When the higher 8 bits of port AH are used as port AHH (PAHH) and the lower 8 bits as port AHL (PAHL), port AH becomes two 8-bit ports that can be set in the input or output mode in 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PAH	0	0	0	0	0	0	PAH9	PAH8	FFFFFF03H	Undefined
	7	6	5	4	3	2	1	0	Address	
	PAH7	PAH6	PAH5	PAH4	PAH3	PAH2	PAH1	PAH0	FFFFFF02H	

Bit position	Bit name	Function
9 to 0	PAHn (n = 9 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

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	Remark	Block Type
Port AH	PAL9 to PAL0	A25 to A16	Address bus when memory expanded	D-2

(2) I/O mode/control mode setting

The port AH I/O mode setting is performed by the port AH mode register (PMAH), and the control mode setting is performed by the port AH mode control register (PMCAH).

(a) Port AH mode register (PMAH)

The port AH mode register (PMAH) can be read or written in 16-bit units.

If the higher 8 bits of PMAH are used as port AH mode register H (PMAHH), and the lower 8 bits as port AH mode register L (PMAHL), these two 8-bit port mode registers can be read or written in 8-bit or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMAH	1	1	1	1	1	1	PMAH9	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
9 to 0	PMAHn (n = 9 to 0)	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 or written in 16-bit units.

If the higher 8 bits of PMCAH are used as port AH mode control register H (PMCAHH), and the lower 8 bits as port AH mode control register L (PMCAHL), these two 8-bit port mode registers can be read or written in 8-bit or 1-bit units.

	15	14	13	12	11	10	9	8	Address	After reset
PMCAH	0	0	0	0	0	0	PMCAH9	PMCAH8	FFFFF043H	03FFH
	7	6	5	4	3	2	1	0	Address	
	PMCAH7	PMCAH6	PMCAH5	PMCAH4	PMCAH3	PMCAH2	PMCAH1	PMCAH0	FFFFF042H	

Bit position	Bit name	Function
9 to 0	PMCAHn (n = 9 to 0)	Specifies operation mode of PAHn pin. 0: I/O port mode 1: A25 to A16 output mode

14.3.8 Port DH

Port DH (PDH) is a 16-bit I/O port that can be set in the input or output mode in 1-bit units.

When the higher 8 bits of port DH are used as port DHH (PDHH), and the lower 8 bits as port DHL (PDHL), port DH becomes two 8-bit ports that can be set in the input or output mode in 1-bit units.

★ **Caution** In the 32-bit mode (MODE1 and MODE0 pins = 00) and when the BMODCN bit of the PFCDH register is set, all the register functions of port DH (see 14.3.8) become invalid.

	15	14	13	12	11	10	9	8	Address	After reset
PDH	PDH15	PDH14	PDH13	PDH12	PDH11	PDH10	PDH9	PDH8	FFFFFF07H	Undefined
	7	6	5	4	3	2	1	0	Address	
	PDH7	PDH6	PDH5	PDH4	PDH3	PDH2	PDH1	PDH0	FFFFFF06H	

Bit position	Bit name	Function
15 to 0	PDHn (n = 15 to 0)	I/O port

Remark For reading/writing of the I/O port, see 14.5 Operation of Port Function.

In addition to their functions as port pins, in the control mode, the port DH pins operate as a data bus for when the memory is externally expanded, real-time pulse unit (RPU) I/O, PWM output, and external interrupt request input.

(1) Operation in control mode

(1/2)

Port	Alternate Function	Remark	Block Type
Port DH	PDH0	D16/ $\overline{\text{INTPD0}}$	Data bus when memory expanded/ external interrupt request input
	PDH1	D17/ $\overline{\text{INTPD1}}$	Data bus when memory expanded/ external interrupt request input
	PDH2	D18/ $\overline{\text{INTPD2}}$ /TOC4	Data bus when memory expanded/ external interrupt request input/ real-time pulse unit (RPU) output
	PDH3	D19/ $\overline{\text{INTPD3}}$	Data bus when memory expanded/ external interrupt request input
	PDH4	D20/ $\overline{\text{INTPD4}}$	Data bus when memory expanded/ external interrupt request input
	PDH5	D21/ $\overline{\text{INTPD5}}$ /TOC5	Data bus when memory expanded/ external interrupt request input/ real-time pulse unit (RPU) output
	PDH6	D22/ $\overline{\text{INTPD6}}$ / INTP100/TCUD10	Data bus when memory expanded/ external interrupt request input/ timer ENC10 external capture trigger input/ real-time pulse unit (RPU) input

Port		Alternate Function	Remark	Block Type
Port DH	PDH7	D23/ $\overline{\text{INTPD7}}$ / INTP101/TCLR10	Data bus when memory expanded/ external interrupt request input/ timer ENC10 external capture trigger input/ real-time pulse unit (RPU) input	M-3
	PDH8	D24/ $\overline{\text{INTPD8}}$ /TO10	Data bus when memory expanded/ external interrupt request input/ real-time pulse unit (RPU) output	M-2
	PDH9	D25/ $\overline{\text{INTPD9}}$ / TIUD10	Data bus when memory expanded/ external interrupt request input/ real-time pulse unit (RPU) input	M-3
	PDH10	D26/ $\overline{\text{INTPD10}}$ / INTP110/TCUD11	Data bus when memory expanded/ external interrupt request input/ timer ENC11 external capture trigger input/ real-time pulse unit (RPU) input	
	PDH11	D27/ $\overline{\text{INTPD11}}$ / INTP111/TCLR11	Data bus when memory expanded/ external interrupt request input/ timer ENC11 external capture trigger input/ real-time pulse unit (RPU) input	
	PDH12	D28/ $\overline{\text{INTPD12}}$ /TO11	Data bus when memory expanded/ external interrupt request input/ real-time pulse unit (RPU) output	M-2
	PDH13	D29/ $\overline{\text{INTPD13}}$ / TIUD11	Data bus when memory expanded/ external interrupt request input/ real-time pulse unit (RPU) input	M-3
	PDH14	D30/ $\overline{\text{INTPD14}}$ / PWM0	Data bus when memory expanded/ external interrupt request input/PWM output	M-2
	PDH15	D31/ $\overline{\text{INTPD15}}$ / PWM1	Data bus when memory expanded/ external interrupt request input/PWM output	

(2) I/O mode/control mode setting

The port DH I/O mode setting is performed by the port DH mode register (PMDH), and the control mode setting is performed by the port DH mode control register (PMCDH) an port DH function control register (PFCDH).

(a) Port DH mode register (PMDH)

The port DH mode register (PMDH) can be read or written in 16-bit units.

If the higher 8 bits of PMDH are used as port DH mode register H (PMDHH), and the lower 8 bits as port DH mode register L (PMDHL), these two 8-bit port mode registers can be read or written in 8-bit or 1-bit units.

	15	14	13	12	11	10	9	8		
PMDH	PMDH15	PMDH14	PMDH13	PMDH12	PMDH11	PMDH10	PMDH9	PMDH8	Address	After reset
									FFFFF027H	FFFFH
	7	6	5	4	3	2	1	0	Address	
	PMDH7	PMDH6	PMDH5	PMDH4	PMDH3	PMDH2	PMDH1	PMDH0	FFFFF026H	

Bit position	Bit name	Function
15 to 0	PMDHn (n = 15 to 0)	Specifies input/output mode for PDHn pin. 0: Output mode (output buffer on) 1: Input mode (output buffer off)

(b) Port DH mode control register (PMCDH)

The port DH mode control register (PMCDH) can be read or written in 16-bit units.

If the higher 8 bits of PMCDH are used as port DH mode control register H (PMCDHH), and the lower 8 bits as port DH mode control register L (PMCDHL), these two 8-bit port mode registers can be read or written in 8-bit or 1-bit units.

(1/2)

	15	14	13	12	11	10	9	8		
PMCDH	PMCDH15	PMCDH14	PMCDH13	PMCDH12	PMCDH11	PMCDH10	PMCDH9	PMCDH8	Address	After reset
									FFFFF047H	0000H
	7	6	5	4	3	2	1	0	Address	
	PMCDH7	PMCDH6	PMCDH5	PMCDH4	PMCDH3	PMCDH2	PMCDH1	PMCDH0	FFFFF046H	

Bit position	Bit name	Function
15	PMCDH15	Specifies operation mode of PDH15 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD15}}$) input mode/PWM1 output mode
14	PMCDH14	Specifies operation mode of PDH14 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD14}}$) input mode/PWM0 output mode
13	PMCDH13	Specifies operation mode of PDH13 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD13}}$) input mode/TIUD11 input mode

Bit position	Bit name	Function
12	PMCDH12	Specifies operation mode of PDH12 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD12}}$) input mode/TO11 output mode
11	PMCDH11	Specifies operation mode of PDH11 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD11}}$) input mode/timer ENC11 external capture trigger (INTP111) input mode/TCLR11 input mode
10	PMCDH10	Specifies operation mode of PDH10 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD10}}$) input mode/timer ENC11 external capture trigger (INTP110) input mode/TCUD11 input mode
9	PMCDH9	Specifies operation mode of PDH9 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD9}}$) input mode/TIUD10 input mode
8	PMCDH8	Specifies operation mode of PDH8 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD8}}$) input mode/TO10 output mode
7	PMCDH7	Specifies operation mode of PDH7 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD7}}$) input mode/timer ENC10 external capture trigger (INTP101) input mode/TCLR10 input mode
6	PMCDH6	Specifies operation mode of PDH6 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD6}}$) input mode/timer ENC10 external capture trigger (INTP100) input mode/TCUD10 input mode
5	PMCDH5	Specifies operation mode of PDH5 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD5}}$) input mode/TOC5 output mode
4	PMCDH4	Specifies operation mode of PDH4 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD4}}$) input mode
3	PMCDH3	Specifies operation mode of PDH3 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD3}}$) input mode
2	PMCDH2	Specifies operation mode of PDH2 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD2}}$) input mode/TOC4 output mode
1	PMCDH1	Specifies operation mode of PDH1 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD1}}$) input mode
0	PMCDH0	Specifies operation mode of PDH0 pin in combination with the PFCDH register. 0: I/O port mode 1: External interrupt request ($\overline{\text{INTPD0}}$) input mode

(c) Port DH function control register (PFCDH)

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

If the higher 8 bits of the PFCDH are used as port DH function control register H (PFCDHH), and the lower 8 bits as port DH function control register L (PFCDHL), these registers can be read or written in 8-bit or 1-bit units.

★ **Caution** When the port mode is specified by the port DH mode control register (PMCDH), the setting of this register becomes invalid. However, bit 0 is independent of the setting of the PMCDH0 bit of the PMCDH register.

(1/3)

	15	14	13	12	11	10	9	8	Address	After reset
PFCDH	PFCDH15	PFCDH14	PFCDH13	PFCDH12	PFCDH11	PFCDH10	PFCDH9	PFCDH8	FFFFF057H	0000H
	7	6	5	4	3	2	1	0	Address	
	PFCDH7	PFCDH6	PFCDH5	0	0	PFCDH2	0	BMODCN	FFFFF056H	

Bit position	Bit name	Function
15	PFCDH15	Specifies operation mode of PDH15 pin in control mode. 0: External interrupt request ($\overline{\text{INTPD15}}$) input mode 1: PWM1 output mode
14	PFCDH14	Specifies operation mode of PDH14 pin in control mode. 0: External interrupt request ($\overline{\text{INTPD14}}$) input mode 1: PWM0 output mode
13	PFCDH13	Specifies operation mode of PDH13 pin in control mode. 0: External interrupt request ($\overline{\text{INTPD13}}$) input mode 1: TIUD11 input mode
12	PFCDH12	Specifies operation mode of PDH12 pin in control mode. 0: External interrupt request ($\overline{\text{INTPD12}}$) input mode 1: TO11 output mode
11	PFCDH11	Specifies operation mode of PDH11 pin in control mode. 0: External interrupt request ($\overline{\text{INTPD11}}$) input mode 1: Timer ENC11 external capture trigger (INTP111) input mode/TCLR11 input mode There is no register that selects a timer ENC11 external capture trigger (INTP111) input mode or TCLR11 input mode. <ul style="list-style-type: none"> To use TCLR11 input mode: Mask the external capture trigger (INTP111) of timer ENC11, or use the CC111 register as a compare register. To use external capture trigger (INTP111) of timer ENC11: Set the CLR111 and CLR110 bits of the TMC11 register to other than 00.

Bit position	Bit name	Function
10	PFCDH10	<p>Specifies operation mode of PDH10 pin in control mode.</p> <p>0: External interrupt request (INTPD10) input mode 1: Timer ENC11 external capture trigger (INTP110) input mode/TCUD10 input mode</p> <p>There is no register that selects a timer ENC11 external capture trigger (INTP110) input mode or TCUD10 input mode.</p> <ul style="list-style-type: none"> To use TUCD10 input mode: Mask the external capture trigger (INTP110) of timer ENC11, or use the CC110 register as a compare register. To use external capture trigger (INTP110) of timer ENC11: Set the T1CMD1 bit of the TUM11 register to 0.
9	PFCDH9	<p>Specifies operation mode of PDH9 pin in control mode.</p> <p>0: External interrupt request (INTPD9) input mode 1: TIUD10 input mode</p>
8	PFCDH8	<p>Specifies operation mode of PDH8 pin in control mode.</p> <p>0: External interrupt request (INTPD8) input mode 1: TO10 output mode</p>
7	PFCDH7	<p>Specifies operation mode of PDH7 pin in control mode.</p> <p>0: External interrupt request (INTPD7) input mode 1: Timer ENC10 external capture trigger (INTP101) input mode/TCLR10 input mode</p> <p>There is no register that selects a timer ENC10 external capture trigger (INTP101) input mode or TCLR10 input mode.</p> <ul style="list-style-type: none"> To use TCLR10 input mode: Mask the external capture trigger (INTP101) of timer ENC10, or use the CC101 register as a compare register. To use external capture trigger (INTP101) of timer ENC10: Set the CLR101 and CLR100 bits of the TMC10 register to other than 00.
6	PFCDH6	<p>Specifies operation mode of PDH6 pin in control mode.</p> <p>0: External interrupt request (INTPD6) input mode 1: Timer ENC10 external capture trigger (INTP100) input mode/TCUD10 input mode</p> <p>There is no register that selects a timer ENC10 external capture trigger (INTP100) input mode or TCUD10 input mode.</p> <ul style="list-style-type: none"> To use TUCD10 input mode: Mask the external capture trigger (INTP100) of timer ENC10, or use the CC100 register as a compare register. To use external capture trigger (INTP100) of timer ENC10: Set the T1CMD0 bit of the TUM10 register to 0.
5	PFCDH5	<p>Specifies operation mode of PDH5 pin in control mode.</p> <p>0: External interrupt request (INTPD5) input mode 1: TOC5 output mode</p>
2	PFCDH2	<p>Specifies operation mode of PDH2 pin in control mode.</p> <p>0: External interrupt request (INTPD2) input mode 1: TOC4 output mode</p>

★

Bit position	Bit name	Function
0	BMODCN	<p>Specifies the operation mode of the D16 to D31 pins in the 16-bit mode (16-bit data bus). However, changing the value of the BMODCN bit from 0 to 1 is not reflected in the LBS register.</p> <p>0: The D16 to D31 pins are not used for starting in the 16-bit mode (data bus width: 16/8 bits).</p> <p>1: The D16 to D31 pins are not used for starting in the 16-bit mode (data bus width: 32/16/8 bits).</p> <p>Caution The BMODCN bit is valid only when the 16-bit mode is specified in accordance with the status of the MODE0 and MODE1 pins. This bit is invalid if the 32-bit mode is specified.</p> <p>The BMODCN bit can be rewritten only once. If it is rewritten twice or more, the operation is not guaranteed.</p> <p>When the BMODCN bit = 1, the operation is the same as when the 32-bit mode is specified in accordance with the status of the MODE0 and MODE1 pins.</p>

(3) Selecting interrupt trigger mode

The valid edges of the $\overline{\text{INTPDn}}$ pin can be selected by program ($n = 0$ to 15). The level detection of the $\overline{\text{INTPDn}}$ pin can also be selected.

External interrupt rising edge specification register DH (INTRDH) and external interrupt falling edge specification register DH (INTFDH) are used to specify the valid edge and level detection.

(a) External interrupt rising edge specification register DH (INTRDH) and external interrupt falling edge specification register DH (INTFDH)

These registers are used to specify the trigger mode of an external interrupt request ($\overline{\text{INTPDn}}$) from an external pin ($n = 0$ to 15). The correspondence between each bit of this register and the external interrupt request controlled by that bit is as follows.

- INTFDH0 and INTRDH0 bits: INTPD0
- INTFDH1 and INTRDH1 bits: INTPD1
- INTFDH2 and INTRDH2 bits: INTPD2
- INTFDH3 and INTRDH3 bits: INTPD3
- INTFDH4 and INTRDH4 bits: INTPD4
- INTFDH5 and INTRDH5 bits: INTPD5
- INTFDH6 and INTRDH6 bits: INTPD6
- INTFDH7 and INTRDH7 bits: INTPD7
- INTFDH8 and INTRDH8 bits: INTPD8
- INTFDH9 and INTRDH9 bits: INTPD9
- INTFDH10 and INTRDH10 bits: INTPD10
- INTFDH11 and INTRDH11 bits: INTPD11
- INTFDH12 and INTRDH12 bits: INTPD12
- INTFDH13 and INTRDH13 bits: INTPD13
- INTFDH14 and INTRDH14 bits: INTPD14
- INTFDH15 and INTRDH15 bits: INTPD15

The rising edge, falling edge, or both the rising and falling edges can be specified as the valid edge of the $\overline{\text{INTPDn}}$ pin, independently for each pin.

INTRDH and INTFDH registers can be read or written in 16-bit units.

When the higher 8 bits of the INTRDH and INTFDH registers are used as INTRDHH and INTFDHH registers, and the lower 8 bits as INTRDHL and INTFDHL registers, these registers can be read or written in 8-bit or 1-bit units.

Caution Before setting the trigger mode, set the PMCDH register.

If the PMCDH register is set after the INTRDH and INTFDH registers have been set, an illegal interrupt may occur when the PMCDH register is set.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
INTRDH	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	INTR	FFFFFC36H	FFFFH
	DH15	DH14	DH13	DH12	DH11	DH10	DH9	DH8	DH7	DH6	DH5	DH4	DH3	DH2	DH1	DH0		
INTFDH	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	INTF	FFFFFC16H	0000H
	DH15	DH14	DH13	DH12	DH11	DH10	DH9	DH8	DH7	DH6	DH5	DH4	DH3	DH2	DH1	DH0		

Bit position	Bit name	Function															
15 to 0	INTFDHn, INTRDHn (n = 0 to 15)	Specify trigger mode of $\overline{\text{INTPDn}}$ pin. <table border="1"> <thead> <tr> <th>INTFDHn</th> <th>INTRDHn</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>	INTFDHn	INTRDHn	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
INTFDHn	INTRDHn	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 $\overline{\text{INTPDn}}$ pin is sampled each time the main clock (fx) is divided by four. When the low level of this pin is detected, an interrupt request is latched as the PDIFn bit (n = 0 to 15). Consequently, even when the CPU acknowledges the interrupt and the PDIFn bit of the interrupt control register (PDICn) is automatically cleared to 0, the PDIFn bit is immediately set to 1 and interrupts occur consecutively. To avoid this status, make the $\overline{\text{INTPDn}}$ pin of the external device inactive in the interrupt servicing routine, and forcibly clear the PDIFn bit to 0.
- 2.** If a level-detected interrupt request (INTPDn) with a lower priority occurs while an interrupt is being serviced and if this level-detected interrupt request (INTPDn) that has newly occurred becomes inactive before the current interrupt has been serviced, the interrupt request of the new interrupt (INTPDn) is held pending (n = 0 to 15). To not acknowledge the interrupt request of INTPDn, clear the PDIFn bit of the interrupt control register.

14.3.9 Port CS

Port CS (PCS) is an 8-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
PCS7	PCS6	PCS5	PCS4	PCS3	PCS2	PCS1	PCS0	FFFFF008H	Undefined

Bit position	Bit name	Function
7 to 0	PCS _n (n = 7 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

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 memory is externally expanded and the read/write strobe signal output to an external I/O.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type	
Port CS	PCS0	$\overline{CS0}$	Chip select signal output	D-2
	PCS1	$\overline{CS1}$	Chip select signal output	
	PCS2	$\overline{CS2}/\overline{IOWR}$	Chip select signal output/write strobe signal output	J-2
	PCS3	$\overline{CS3}$	Chip select signal output	D-2
	PCS4	$\overline{CS4}$	Chip select signal output	
	PCS5	$\overline{CS5}/\overline{IORD}$	Chip select signal output/read strobe signal output	J-2
	PCS6	$\overline{CS6}$	Chip select signal output	D-2
	PCS7	$\overline{CS7}$	Chip select signal output	

(2) I/O mode/control mode setting

The port CS I/O mode setting is performed by the port CS mode register (PMCS), and the control mode setting is performed by the port CS mode control register (PMCCS) and the port CS function control register (PFCCS).

(a) Port CS mode register (PMCS)

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
PMCS	PMCS7	PMCS6	PMCS5	PMCS4	PMCS3	PMCS2	PMCS1	PMCS0	FFFFF028H	FFH

Bit position	Bit name	Function
7 to 0	PMCSn (n = 7 to 0)	Specifies input/output mode for PCSn 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 or written in 8-bit or 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
PMCCS	PMCCS7	PMCCS6	PMCCS5	PMCCS4	PMCCS3	PMCCS2	PMCCS1	PMCCS0	FFFFF048H	FFH

Bit position	Bit name	Function
7	PMCCS7	Specifies operation mode of PCS7 pin. 0: I/O port mode 1: $\overline{CS7}$ output mode
6	PMCCS6	Specifies operation mode of PCS6 pin. 0: I/O port mode 1: $\overline{CS6}$ output mode
5	PMCCS5	Specifies operation mode of PCS5 pin. 0: I/O port mode 1: $\overline{CS5}$ output mode/ \overline{IORD} output mode
4	PMCCS4	Specifies operation mode of PCS4 pin. 0: I/O port mode 1: $\overline{CS4}$ output mode
3	PMCCS3	Specifies operation mode of PCS3 pin. 0: I/O port mode 1: $\overline{CS3}$ output mode
2	PMCCS2	Specifies operation mode of PCS2 pin. 0: I/O port mode 1: $\overline{CS2}$ output mode/ \overline{IOWR} output mode
1	PMCCS1	Specifies operation mode of PCS1 pin. 0: I/O port mode 1: $\overline{CS1}$ output mode
0	PMCCS0	Specifies operation mode of PCS0 pin. 0: I/O port mode 1: $\overline{CS0}$ output mode

(c) Port CS function control register (PFCCS)

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

Caution When the port mode is specified by the port CS mode control register (PMCCS), the PFCCS setting becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFCCS	0	0	PFCCS5	0	0	PFCCS2	0	0	FFFFFF049H	00H

Bit position	Bit name	Function
5	PFCCS5	Specifies operation mode of PCS5 pin in control mode. 0: $\overline{CS5}$ output mode 1: \overline{IORD} output mode ^{Note}
2	PFCCS2	Specifies operation mode of PCS2 pin in control mode. 0: $\overline{CS2}$ output mode 1: \overline{IOWR} output mode ^{Note}

Note To output the \overline{IORD} and \overline{IOWR} signals during access to the external I/O other than by a DMA flyby transfer, the IOEN bit of the BCP register must be set.

14.3.10 Port CT

Port CT (PCT) is a 6-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	PCT7	0	PCT5	PCT4	PCT3	PCT2	PCT1	PCT0	FFFFFF0AH	Undefined

Bit position	Bit name	Function
7, 5 to 0	PCTn (n = 7, 5 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to the port function, this port outputs, in the control mode, control signals to externally expand the memory and byte enable signals when SDRAM is accessed in byte units.

(1) Operation in control mode

Port	Alternate Function	Remark	Block Type
Port CT	PCT0	$\overline{LLWR}/\overline{LLBE}/\overline{LLDQM}$ Write strobe signal output/ byte enable signal output/ output disable/write mask signal	J-3
	PCT1	$\overline{LUWR}/\overline{LUBE}/\overline{LUDQM}$ Write strobe signal output/ byte enable signal output/ output disable/write mask signal	
	PCT2	$\overline{ULWR}/\overline{ULBE}/\overline{ULDQM}$ Write strobe signal output/ byte enable signal output/ output disable/write mask signal	
	PCT3	$\overline{UUWR}/\overline{UUBE}/\overline{UUDQM}$ Write strobe signal output/ byte enable signal output/ output disable/write mask signal	
	PCT4	\overline{RD} Read strobe signal output	D-2
	PCT5	$\overline{WE}/\overline{WR}$ Write enable signal output/ write strobe signal output	
	PCT7	\overline{BCYST} Bus cycle status signal output	

(2) I/O mode/control mode setting

The port CT I/O mode setting is performed by the port CT mode register (PMCT), and the control mode setting is performed by the port CT mode control register (PMCCT) and port CT function control register (PFCCT).

(a) Port CT mode register (PMCT)

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
PMCT	PMCT7	1	PMCT5	PMCT4	PMCT3	PMCT2	PMCT1	PMCT0	FFFFF02AH	FFH

Bit position	Bit name	Function
7, 5 to 0	PMCTn (n = 7, 5 to 0)	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 or written in 8-bit or 1-bit units.

	7	6	5	4	3	2	1	0	Address	After reset
PMCCT	PMCCT7	0	PMCCT5	PMCCT4	PMCCT3	PMCCT2	PMCCT1	PMCCT0	FFFFF04AH	BFH

Bit position	Bit name	Function
7	PMCCT7	Specifies operation mode of PCT7 pin. 0: I/O port mode 1: $\overline{\text{BCYST}}$ output mode
5	PMCCT5	Specifies operation mode of PCT5 pin. 0: I/O port mode 1: $\overline{\text{WE}}/\overline{\text{WR}}$ output mode The $\overline{\text{WE}}$ output mode and $\overline{\text{WR}}$ output mode are automatically selected by accessing the memory for which each mode is targeted.
4	PMCCT4	Specifies operation mode of PCT4 pin. 0: I/O port mode 1: $\overline{\text{RD}}$ output mode
3	PMCCT3	Specifies operation mode of PCT3 pin. 0: I/O port mode 1: $\overline{\text{UUWR}}$ output mode/ $\overline{\text{UUBE}}$ output mode/ $\overline{\text{UUDQM}}$ output mode
2	PMCCT2	Specifies operation mode of PCT2 pin. 0: I/O port mode 1: $\overline{\text{ULWR}}$ output mode/ $\overline{\text{ULBE}}$ output mode/ $\overline{\text{ULDQM}}$ output mode
1	PMCCT1	Specifies operation mode of PCT1 pin. 0: I/O port mode 1: $\overline{\text{LUWR}}$ output mode/ $\overline{\text{LUBE}}$ output mode/ $\overline{\text{LUDQM}}$ output mode
0	PMCCT0	Specifies operation mode of PCT0 pin. 0: I/O port mode 1: $\overline{\text{LLWR}}$ output mode/ $\overline{\text{LLBE}}$ output mode/ $\overline{\text{LLDQM}}$ output mode

(c) Port CT function control register (PFCCT)

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

Cautions 1. When the port mode is specified by the port CT mode control register (PMCCT), the setting of this register becomes invalid.

★

2. The timing of the \overline{xxDQM} signal differs between when the \overline{xxWR} output mode/ \overline{xxDQM} output mode is selected and when the \overline{xxBE} output mode/ \overline{xxDQM} output mode is selected. However, if either mode is selected, no problems occur when connecting to SDRAM. For the output timing of the \overline{xxDQM} signal, refer to the timing diagrams (Figures 5-9 to 5-11) in 5.3.5 SDRAM access (xx = UU, UL, LU, LL).

	7	6	5	4	3	2	1	0	Address	After reset
PFCCT	0	0	0	0	PFCCT3	PFCCT2	PFCCT1	PFCCT0	FFFFF04BH	00H

Bit position	Bit name	Function
3	PFCCT3	Specifies operation mode of PCT3 pin in control mode. 0: \overline{UUWR} output mode/ \overline{UUDQM} output mode 1: \overline{UUBE} output mode/ \overline{UUDQM} output mode The \overline{UUWR} output mode and \overline{UUDQM} output mode, and the \overline{UUBE} output mode and \overline{UUDQM} output mode are automatically selected by accessing the memory for which each mode is targeted.
2	PFCCT2	Specifies operation mode of PCT2 pin in control mode. 0: \overline{ULWR} output mode/ \overline{ULDQM} output mode 1: \overline{ULBE} output mode/ \overline{ULDQM} output mode The \overline{ULWR} output mode and \overline{ULDQM} output mode, and the \overline{ULBE} output mode and \overline{ULDQM} output mode are automatically selected by accessing the memory for which each mode is targeted.
1	PFCCT1	Specifies operation mode of PCT1 pin in control mode. 0: \overline{LUWR} output mode/ \overline{LUDQM} output mode 1: \overline{LUBE} output mode/ \overline{LUDQM} output mode The \overline{LUWR} output mode and \overline{LUDQM} output mode, and the \overline{LUBE} output mode and \overline{LUDQM} output mode are automatically selected by accessing the memory for which each mode is targeted.
0	PFCCT0	Specifies operation mode of PCT0 pin in control mode. 0: \overline{LLWR} output mode/ \overline{LLDQM} output mode 1: \overline{LLBE} output mode/ \overline{LUDQM} output mode The \overline{LLWR} output mode and \overline{LLDQM} output mode, and the \overline{LLBE} output mode and \overline{LUDQM} output mode are automatically selected by accessing the memory for which each mode is targeted.

14.3.11 Port CM

Port CM (PCM) 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
PCM	0	0	PCM5	PCM4	PCM3	PCM2	PCM1	PCM0	FFFFFF00CH	Undefined

Bit position	Bit name	Function
5 to 0	PCMn (n = 5 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, in the control mode, the port CM pins operate as the wait insertion signal input, bus hold control signal output, refresh request signal output from SDRAM, and A/D converter external trigger input.

(1) Operation in control mode

	Port	Alternate Function	Remark	Block Type
Port CM	PCM0	$\overline{\text{WAIT}}$	Wait insertion signal input	C-1
	PCM1	-	-	D-1
	PCM2	$\overline{\text{HLD}}\text{AK}$	Bus hold acknowledge signal output	D-2
	PCM3	$\overline{\text{HLD}}\text{RQ}$	Bus hold request signal input	C-1
	PCM4	$\overline{\text{REF}}\text{RQ}$	Refresh request signal output	D-2
	PCM5	$\overline{\text{SELF}}\text{REF}/\text{ADTRG}$	Self-refresh request signal input	F-5

(2) I/O mode/control mode setting

The port CM I/O mode setting is performed by the port CM mode register (PMCM), and the control mode setting is performed by the port CM mode control register (PMCCM) and the port CM function control register (PFCCM).

(a) Port CM mode register (PMCM)

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
PMCM	1	1	PMCM5	PMCM4	PMCM3	PMCM2	PMCM1	PMCM0	FFFFF02CH	FFH

Bit position	Bit name	Function
5 to 0	PMCMn (n = 5 to 0)	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 or written in 8-bit or 1-bit units.

Be sure to clear bit 1 to 0. If it is set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
PMCCM	0	0	PMCCM5	PMCCM4	PMCCM3	PMCCM2	0	PMCCM0	FFFFF04CH	3DH

Bit position	Bit name	Function
5	PMCCM5	Specifies operation mode of PCM5 pin. 0: I/O port mode 1: SELFR \overline{E} F input mode/A/D converter external trigger (ADTRG) input mode
4	PMCCM4	Specifies operation mode of PCM4 pin. 0: I/O port mode 1: REFR \overline{Q} output mode
3	PMCCM3	Specifies operation mode of PCM3 pin. 0: I/O port mode 1: HLDR \overline{Q} input mode
2	PMCCM2	Specifies operation mode of PCM2 pin. 0: I/O port mode 1: HLDA \overline{K} output mode
0	PMCCM0	Specifies operation mode of PCM0 pin. 0: I/O port mode 1: WAIT input mode

(c) Port CM function control register (PFCCM)

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

Caution When the port mode is specified by the port CM mode control register (PMCCM), the PFCCM setting becomes invalid.

	7	6	5	4	3	2	1	0	Address	After reset
PFCCM	0	0	PFCCM5	0	0	0	0	0	FFFFFF04DH	00H

Bit position	Bit name	Function
5	PFCCM5	Specifies operation mode of PCM5 pin in control mode. 0: SELFREF input mode 1: A/D converter external trigger (ADTRG) input mode

14.3.12 Port CD

Port CD (PCD) 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
PCD	0	0	0	0	PCD3	PCD2	PCD1	PCD0	FFFFF00EH	Undefined

Bit position	Bit name	Function
3 to 0	PCDn (n = 3 to 0)	I/O port

Remark For reading/writing of the I/O port, see **14.5 Operation of Port Function**.

In addition to their function as port pins, the port CD pins operate as the clock enable signal output, bus clock output, column address strobe signal output, and row address strobe signal output in the control mode.

(1) Operation in control mode

	Port	Alternate Function	Remark	Block Type
Port CD	PCD0	SDCKE	Clock enable signal output	D-2
	PCD1	BUSCLK	Bus clock output	D-1
	PCD2	$\overline{\text{SDCAS}}$	Column address strobe signal output	D-2
	PCD3	$\overline{\text{SDRAS}}$	Row address strobe signal output	

(2) I/O mode/control mode setting

The port CD I/O mode setting is performed by the port CD mode register (PMCD), and the control mode setting is performed by the port CD mode control register (PMCCD).

(a) Port CD mode register (PMCD)

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
PMCD	1	1	1	1	PMCD3	PMCD2	PMCD1	PMCD0	FFFFFF02EH	FFH

Bit position	Bit name	Function
3 to 0	PMCDn (n = 3 to 0)	Specifies input/output mode for PCDn pin. 0: Output mode (output buffer on) 1: Input mode (output buffer off)

(b) Port CD mode control register (PMCCD)

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
PMCCD	0	0	0	0	PMCCD3	PMCCD2	PMCCD1	PMCCD0	FFFFFF04EH	0FH

Bit position	Bit name	Function
3	PMCCD3	Specifies operation mode of PCD3 pin. 0: I/O port mode 1: $\overline{\text{SDRAS}}$ output mode
2	PMCCD2	Specifies operation mode of PCD2 pin. 0: I/O port mode 1: $\overline{\text{SDCAS}}$ output mode
1	PMCCD1	Specifies operation mode of PCD1 pin. 0: I/O port mode 1: $\overline{\text{BUSCLK}}$ output mode
0	PMCCD0	Specifies operation mode of PCD0 pin. 0: I/O port mode 1: $\overline{\text{SDCKE}}$ output mode

14.4 Configuration of $\overline{\text{RESET}}$, A2 to A15, and D0 to D15 Pins

The $\overline{\text{RESET}}$, A2 to A15, and D0 to D15 pins are not alternative function pins. Their configuration is as follows.

(1) Configuration of $\overline{\text{RESET}}$, A2 to A15, and D0 to D15 pins

Pin Function	Remark	Block Type
$\overline{\text{RESET}}$	Schmitt buffer	N-1
A2 to A15	Output buffer off control	N-2
D0 to D15	Output buffer off control and DIR control	N-3

(2) Block diagram of $\overline{\text{RESET}}$, A2 to A15, and D0 to D15 pins

Figure 14-25. Block Diagram of Type N-1

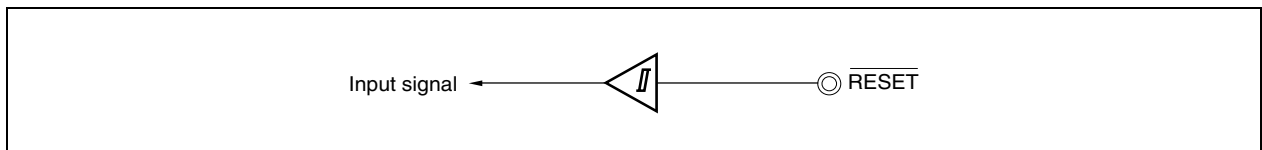


Figure 14-26. Block Diagram of Type N-2

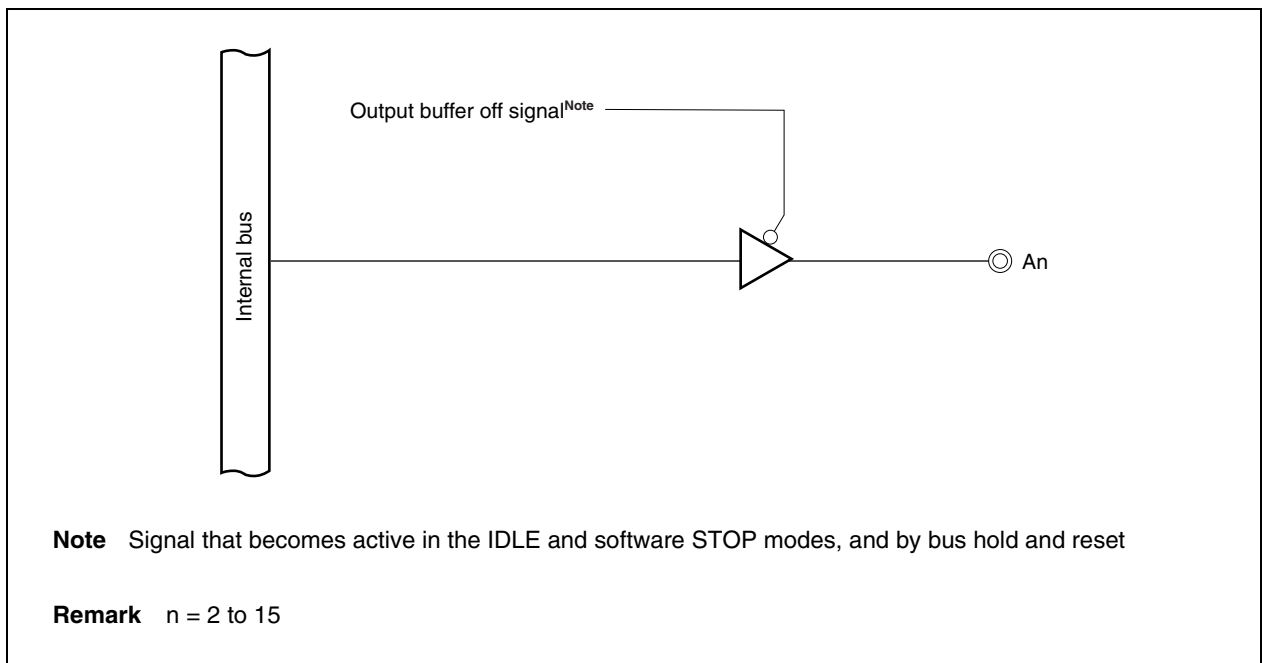
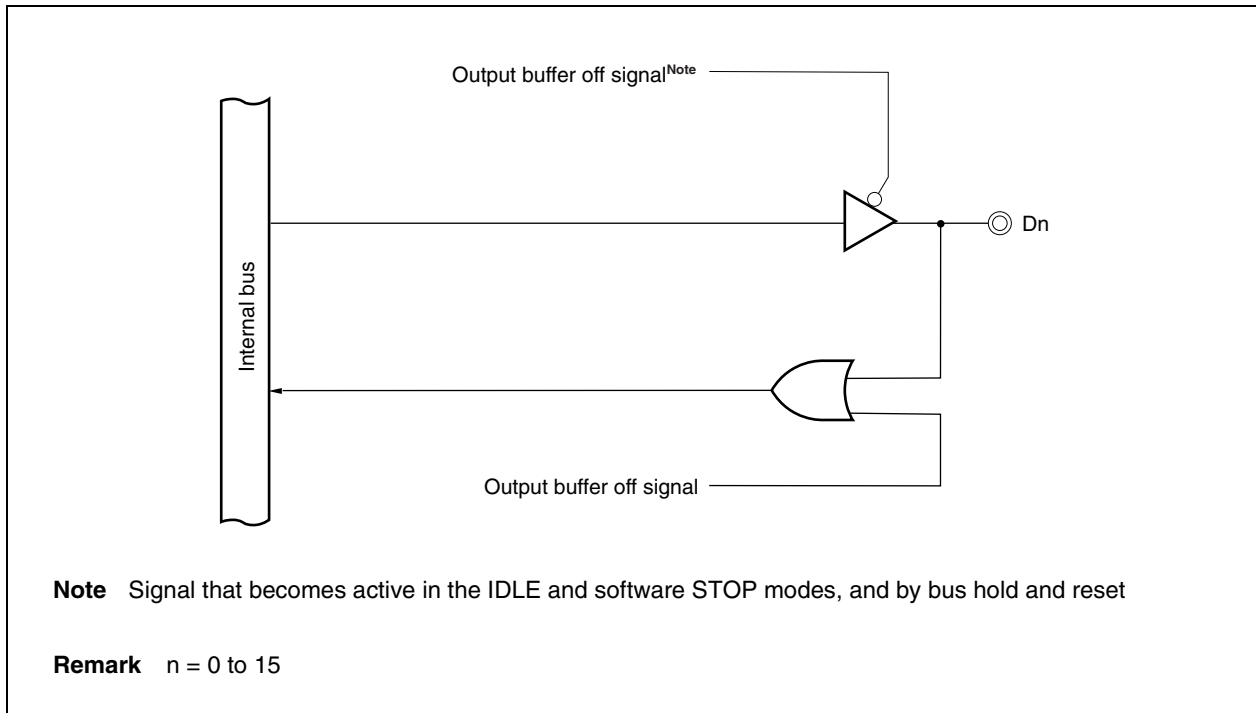


Figure 14-27. Block Diagram of Type N-3



14.5 Operation of Port Function

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

14.5.1 Writing to I/O port

(1) In output mode

A value can be written to the output latch (Pn) by writing it to the port n register (Pn). The contents of the output latch are output from the pin.

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

(2) In input mode

A value can be written to the output latch (Pn) by writing it to the port n register (Pn). However, the status of the pin does not change because the output buffer is off.

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

Caution A bit manipulation instruction (CLR1, SET1, NOT1) manipulates 1 bit but accesses a port in 8-bit units. If this instruction is executed to manipulate a port with a mixture of input and output bits, the contents of the output latch of a pin set in the input mode, in addition to the bit to be manipulated, are overwritten to the current input pin status and become undefined.

14.5.2 Reading from I/O port

(1) In output mode

The contents of the output latch (Pn) can be read by reading the port n register (Pn). The contents of the output latch do not change.

(2) In input mode

The status of the pin can be read by reading the port n register (Pn). The contents of the output latch (Pn) do not change.

14.5.3 Output status of alternate function in control mode

The status of a port pin is not dependent upon the setting of the PMcN register and can be read by setting the port n mode register (PMn) to the input mode. If the PMn register is set to the output mode, the value of the port n register (Pn) can be read in the port mode, and the output status of the alternate function can be read in the control mode.

14.6 Noise Eliminator

14.6.1 Interrupt input pin

The following timing controller used to secure the noise elimination time is provided for the NMI and port pins that operate in the control mode when the valid edge is input. Input signals that change within the noise elimination time are not internally acknowledged.

Table 14-1. Noise Elimination Time of Interrupt Input Pins

Pin	Noise Elimination Time
NMI	Analog delay (80 ns typ.)
$\overline{\text{INTP10}}$	
$\overline{\text{INTP11}}$	
$\overline{\text{INTP21}}$	
$\overline{\text{INTP22}}$	
$\overline{\text{INTP23}}$	
$\overline{\text{INTP24}}$	
$\overline{\text{INTP25}}$	
$\overline{\text{INTP50}}$	
$\overline{\text{INTP51}}$	
$\overline{\text{INTP52}}$	
$\overline{\text{INTP65}}$	
$\overline{\text{INTP66}}$	
$\overline{\text{INTP67}}$	
$\overline{\text{INTPL0}}$	
$\overline{\text{INTPL1}}$	
$\overline{\text{INTPD0}}$	
$\overline{\text{INTPD1}}$	
$\overline{\text{INTPD2}}$	
$\overline{\text{INTPD3}}$	
$\overline{\text{INTPD4}}$	
$\overline{\text{INTPD5}}$	
$\overline{\text{INTPD6}}$	
$\overline{\text{INTPD7}}$	
$\overline{\text{INTPD8}}$	
$\overline{\text{INTPD9}}$	
$\overline{\text{INTPD10}}$	
$\overline{\text{INTPD11}}$	
$\overline{\text{INTPD12}}$	
$\overline{\text{INTPD13}}$	
$\overline{\text{INTPD14}}$	
$\overline{\text{INTPD15}}$	

- Cautions**
1. The above non-maskable and maskable interrupt pins are used to release the standby mode. A timing circuit that controls the clock is not employed because the internal system clock is stopped in the standby mode.
 2. The noise eliminator is valid only in the control mode.

14.6.2 A/D converter input pin

The following timing controller used to secure the noise elimination time is provided for the ADTRG pin. An input signal that changes within the noise elimination time is not internally acknowledged.

Table 14-2. Noise Elimination Time of A/D Converter Input Pin

Pin	Noise Elimination Time
PCM5/ADTRG/SELFREF	Analog delay (80 ns typ.)

Caution The noise eliminator is valid only in the control mode.

14.6.3 Timer C and timer ENC1 input pins

The following noise filter that operates via clock sampling is provided for the pins of timers C and ENC1 that operate when the valid edge is input. Input signals that change within the noise elimination time are not internally acknowledged.

Table 14-3. Noise Elimination Time of Timer C and Timer ENC1 Input Pins

Pin	Noise Elimination Time
INTPC00/TIC0 INTPC01 INTPC10/TIC1 INTPC11 INTPC20/TIC2 INTPC21 INTPC30/TIC3 INTPC31	Selected from 0, 2, 3, or 5 clocks
INTP100/TCUD10 INTP101/TCLR10 TIUD10 INTP110/TCUD11 INTP111/TCLR11 TIUD11	

- Cautions**
1. The noise filter of the above pins cannot acknowledge an input signal when the CPU clock is stopped because it uses clock sampling.
 2. The noise eliminator is valid only in the control mode.

(1) Noise elimination width setting registers C0 to C3 (NCWC0 to NCWC3)

These registers are used to set the noise elimination width of the digital noise filter of the timer C input pins. These registers can be read or written in 8-bit units.

Be sure to clear bits 7 to 2 to 0. If they are set to 1, the operation is not guaranteed.

	7	6	5	4	3	2	1	0	Address	After reset
NCWC0	0	0	0	0	0	0	NCCC01	NCCC00	FFFFFF610H	02H
NCWC1	0	0	0	0	0	0	NCCC11	NCCC10	FFFF630H	02H
NCWC2	0	0	0	0	0	0	NCCC21	NCCC20	FFFFFF650H	02H
NCWC3	0	0	0	0	0	0	NCCC31	NCCC30	FFFFFF670H	02H

Bit position	Bit name	Function															
1, 0	NCCCN1, NCCCN0	Specify number of clocks from which noise is to be eliminated. <table border="1"> <thead> <tr> <th>NCCCN1</th> <th>NCCCN0</th> <th>Number of clocks from which noise is to be eliminated</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0 (through input)</td> </tr> <tr> <td>0</td> <td>1</td> <td>2</td> </tr> <tr> <td>1</td> <td>0</td> <td>3</td> </tr> <tr> <td>1</td> <td>1</td> <td>5</td> </tr> </tbody> </table>	NCCCN1	NCCCN0	Number of clocks from which noise is to be eliminated	0	0	0 (through input)	0	1	2	1	0	3	1	1	5
NCCCN1	NCCCN0	Number of clocks from which noise is to be eliminated															
0	0	0 (through input)															
0	1	2															
1	0	3															
1	1	5															
Remark 1 clock = $f_x/4$ f_x : Main clock																	

Remark n = 0 to 3

(2) Noise elimination width setting registers 10, 11 (NCW10, NCW11)

These registers are used to set the noise elimination width of the digital noise filter of the timer ENC1 input pins.

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

Cautions 1. Whether the signal is input through or inverted can be specified for each of the INTP1n0/TCUD1n and TIUD1n pins. The noise elimination width set by the NCFn, NCC1n, and NCC0n bits is for each timer and cannot be changed for each pin.

2. The setting of the SRTCn bit is valid even when the INTP1n0/TCUD1n pin is used as a capture trigger (INTP1n0).

	7	6	5	4	3	2	1	0	Address	After reset
NCW10	0	0	SRTC0	SRTI0	0	NCF0	NCC10	NCC00	FFFFF5C0H	02H
NCW11	0	0	SRTC1	SRTI1	0	NCF1	NCC11	NCC01	FFFFF5F0H	02H

Bit position	Bit name	Function															
5	SRTCn	Specifies the input mode of the INTP1n0/TCUD1n pin. 0: Through input 1: Inverted This bit specifies whether the signal input from the INTP1n0/TCUD1n pin is input through to TMENC1n, or inverted.															
4	SRTIn	Specifies the input mode of the TIUD1n pin. 0: Through input 1: Inverted This bit specifies whether the signal input from the TIUD1n pin is input through to TMENC1n, or inverted.															
2	NCFn	Specifies the clock frequency for noise elimination. 0: $f_x/4$ 1: $f_x/32$ This bit selects a clock source for the noise filter.															
1, 0	NCC1n, NCC0n	Specify the number of clocks from which noise is to be eliminated. <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 15%;">NCC1n</th> <th style="width: 15%;">NCC0n</th> <th style="width: 70%;">Number of clocks from which noise is to be eliminated</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>0 (through input)^{Note}</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>2</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>3</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>5</td> </tr> </tbody> </table> <p>Note Clear the NCFn bit to 0 to input the signal through.</p> <p>These bits are used to select the number of clocks from which noise is to be eliminated.</p>	NCC1n	NCC0n	Number of clocks from which noise is to be eliminated	0	0	0 (through input) ^{Note}	0	1	2	1	0	3	1	1	5
NCC1n	NCC0n	Number of clocks from which noise is to be eliminated															
0	0	0 (through input) ^{Note}															
0	1	2															
1	0	3															
1	1	5															

Remark n = 0, 1

(a) Relationship between NCW1n register set value and noise elimination width

★ Table 14-4. Relationship Between NCW1n Register Set Value and Noise Elimination Width

NCW1n Register			Noise Elimination Width (ns)			Remark
NCFn Bit	NCC1n Bit	NCC0n Bit	$f_x = 150$ MHz	$f_x = 133$ MHz	$f_x = 100$ MHz	
0	0	0	0	0	0	Through
0	0	1	53.3 to 80.0	60.2 to 90.2	80 to 120	$(1/(f_x/4)) \times 2$
0	1	0	80.0 to 106.7	90.2 to 120.3	120 to 160	$(1/(f_x/4)) \times 3$
0	1	1	133.3 to 160.0	150.4 to 180.5	200 to 240	$(1/(f_x/4)) \times 5$
1	0	0	0	0	0	Through
1	0	1	426.7 to 640.0	481.2 to 721.8	640 to 960	$(1/(f_x/32)) \times 2$
1	1	0	640.0 to 853.3	721.8 to 962.9	960 to 1,280	$(1/(f_x/32)) \times 3$
1	1	1	1,066.7 to 1,280.0	1,203.0 to 1,443.6	1,600 to 1,920	$(1/(f_x/32)) \times 5$

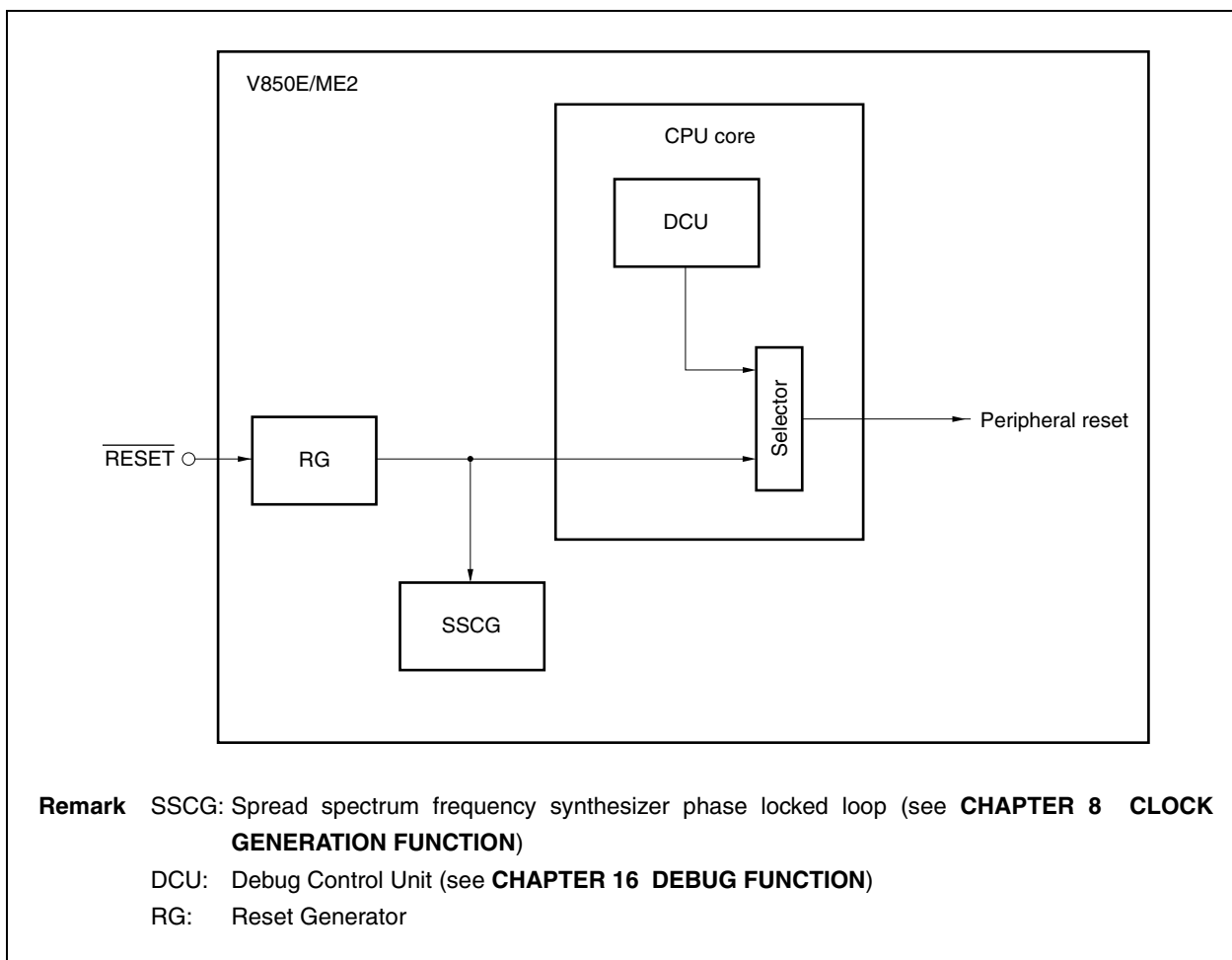
- Remarks**
1. $n = 0, 1$
 2. f_x : Main clock

CHAPTER 15 RESET FUNCTIONS

15.1 Overview

- Reset function by $\overline{\text{RESET}}$ input
- Forced reset function by DCU (See **CHAPTER 16 DEBUG FUNCTION (DCU)**.)
- Low level is input to the $\overline{\text{RESET}}$ pin for 100 μs min. If an oscillation stabilization time of 100 μs or longer is necessary, secure the low level for as long as the necessary oscillation stabilization time.
- Reset generator (RG) eliminates noise from the $\overline{\text{RESET}}$ pin.

15.2 Configuration



During a system reset, most pins (all excluding the BUSCLK, $\overline{\text{RESET}}$, X2, EV_{DD}, EV_{SS}, IV_{DD}, IV_{SS}, PLLV_{DD}, PLLV_{SS}, OSCV_{DD}, OSCV_{SS}, UV_{DD}, AV_{DD}, AV_{REFP}, AV_{REFM}, and AV_{SS} pins) enter the high-impedance state.

Therefore, a pull-up (or pull-down) resistor must be connected to each pin of the address bus, data bus, and external bus control signals. If no resistor is connected, external memory may be destroyed when these pins enter the high-impedance state.

For the same reason, the output signals of the on-chip peripheral I/O functions and other output ports should be handled in the same manner.

15.3 Operation

When a low-level signal is input to the $\overline{\text{RESET}}$ pin, a system reset is effected and each on-chip hardware is initialized.

When the $\overline{\text{RESET}}$ pin level changes from low to high, the reset state is released and the CPU starts program execution using OSC output clock (F_x). Register contents must be initialized as required in the program.

When the reset signal is cleared, the oscillation stabilization time is not inserted. When the $\overline{\text{RESET}}$ signal is to be input (reset input on power application or reset input when the software STOP mode is released) while the clock oscillator is stopped, therefore, a low-level width longer than the oscillation stabilization time (100 μs min.) must be secured at the $\overline{\text{RESET}}$ pin. When the $\overline{\text{RESET}}$ signal is to be input (reset input when the IDLE mode is released) while the clock oscillator is not stopped, a low level of 100 μs min. must be secured at the $\overline{\text{RESET}}$ pin.

Table 15-1. Hardware Status at Reset

Hardware	During Reset Period	After Reset Is Cleared
OSC	Oscillation/supply continues	
Clock generator	<ul style="list-style-type: none"> • Outputs OSC clock (F_x) • Output is not guaranteed before oscillation stabilization time. 	<ul style="list-style-type: none"> • Outputs OSC clock (F_x) • Operable at $f_x = F_x \times 8$ by controlling the CKS register
CPU	Stops operating	Starts operating
Internal instruction RAM, internal data RAM	Undefined	
Debug function	Stops operating	Operable
On-chip peripheral I/O registers	Initialized to specific status	
On-chip peripheral functions other than above	Stops operating	Can start operating
Pin function	See 2.2 Pin Status .	

The reset operation when the $\overline{\text{RESET}}$ pin is input is illustrated below.

Figure 15-1. Reset Operation with $\overline{\text{RESET}}$ Pin Input (1/2)

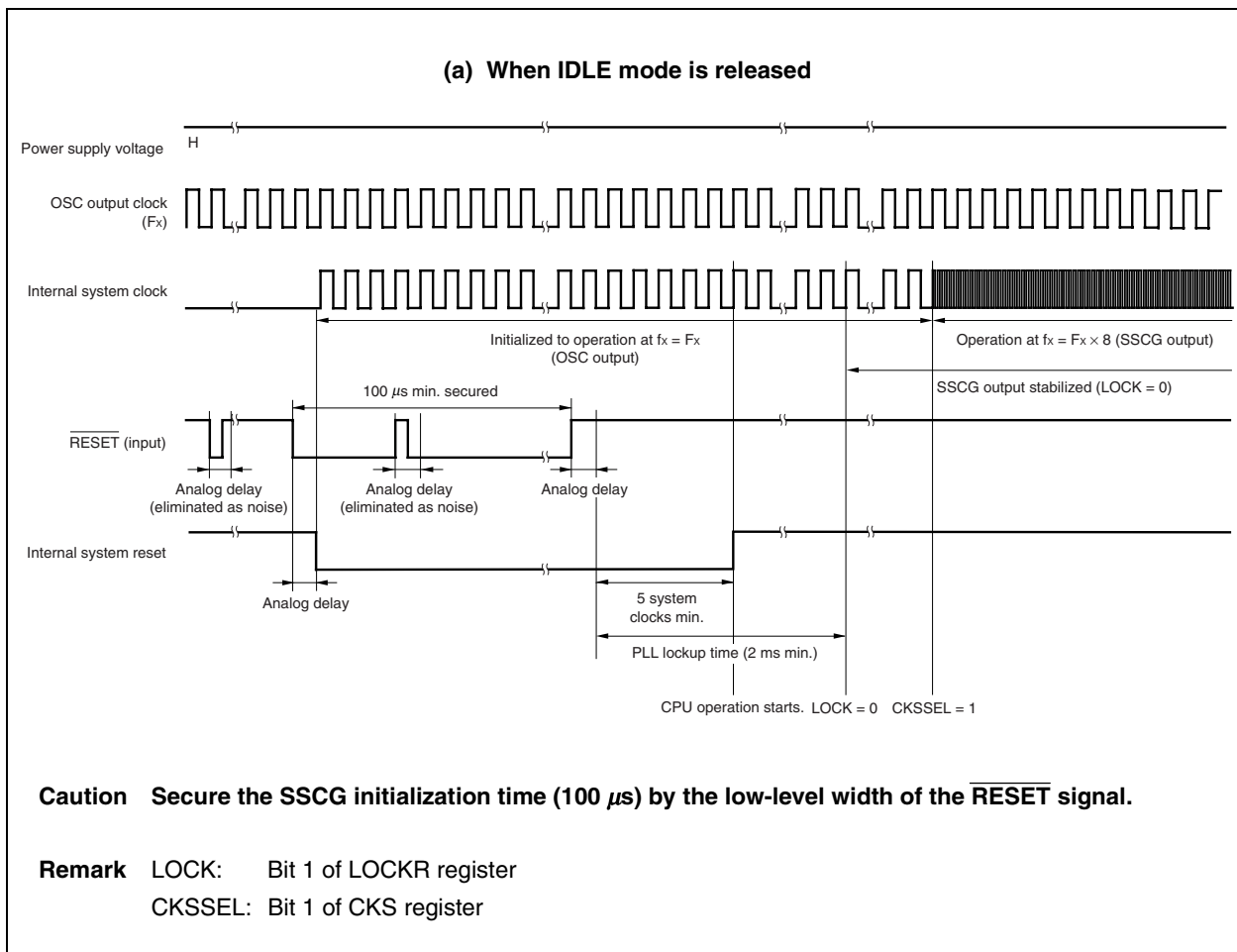
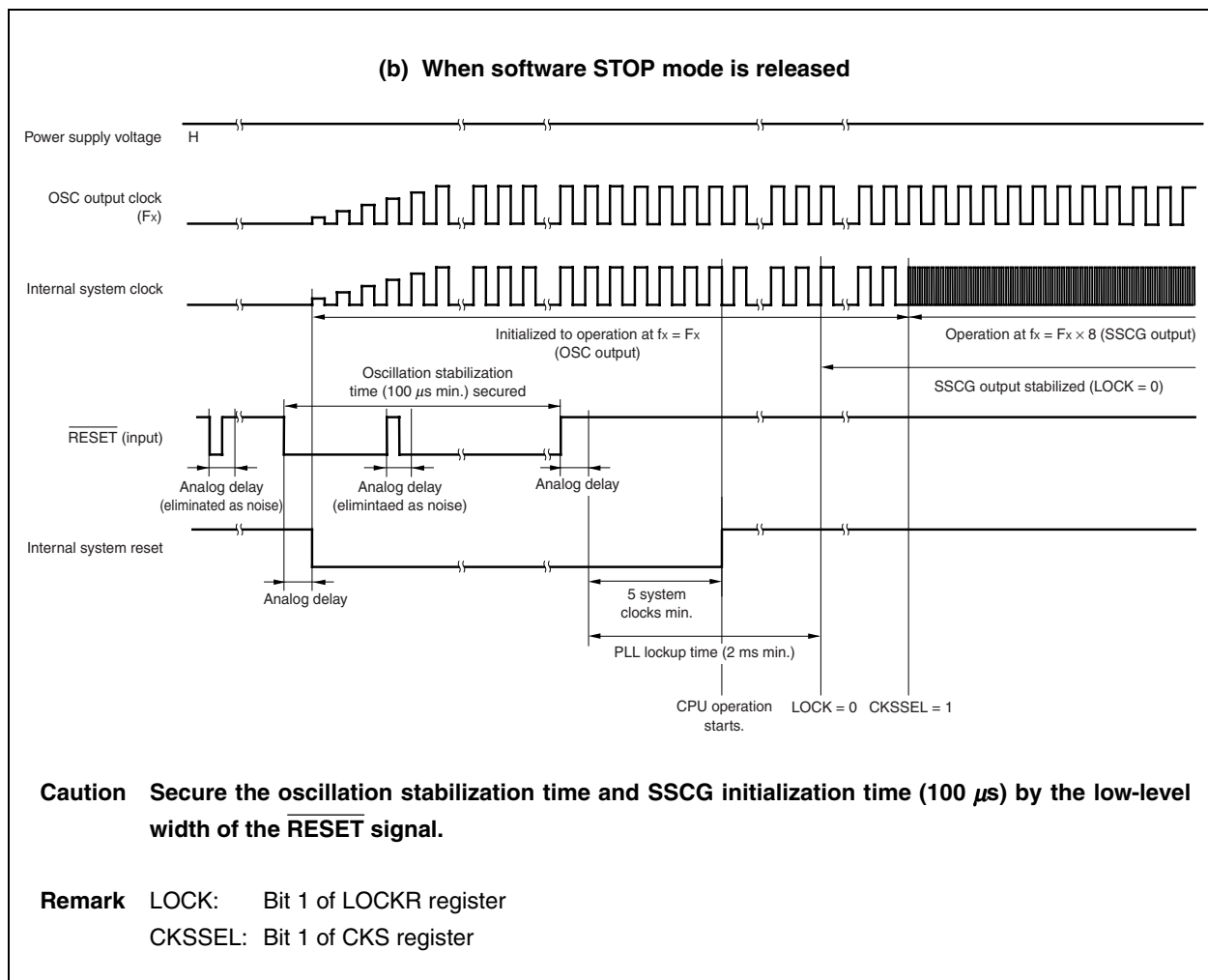
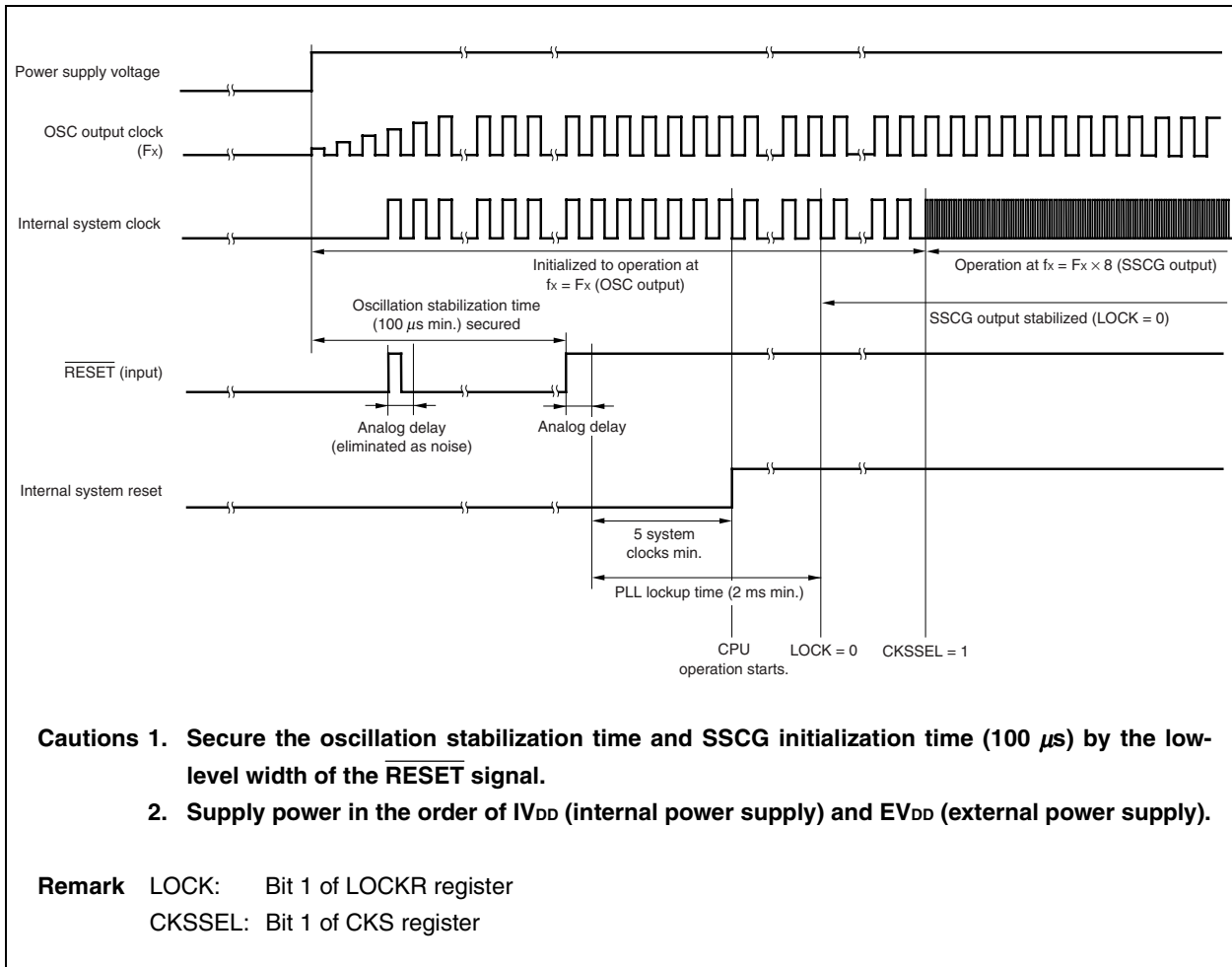


Figure 15-1. Reset Operation with $\overline{\text{RESET}}$ Pin Input (2/2)



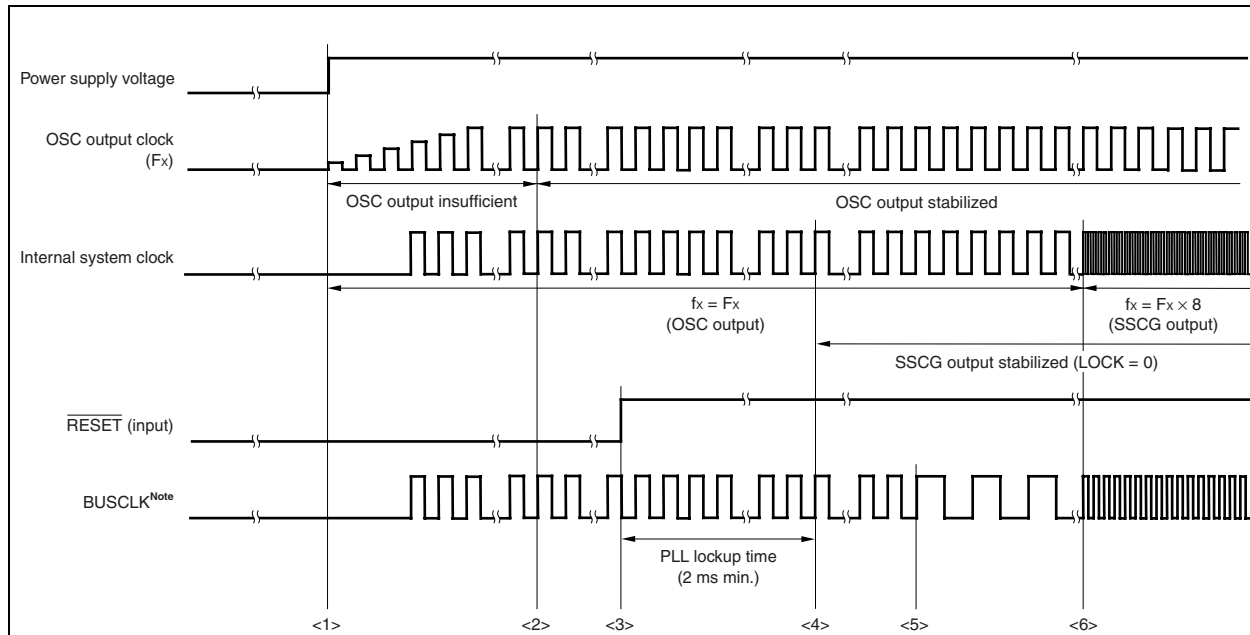
The power-on reset operation is illustrated below.

Figure 15-2. Power-on Reset Operation



The BUSCLK operation at power-on reset is illustrated below.

Figure 15-3. BUSCLK Operation at Power-on Reset



- Remarks 1.**
- <1>: Power on
 - <2>: OSC oscillation stabilization
 - <3>: RESET cleared (counting of PLL lockup time starts)
 - <4>: PLL lock status (LOCK bit of LOCKR register = 0)
 - <5>: BUSCLK = f_{CLK}/2 (CKM1 and CKM0 bits of BMC register = 01)
 - <6>: Set to f_x = F_X × 8 (CKSSEL bit of CKS register = 1)
- 2.** The above operation is when BUSCLK operates on 1/2 the cycle of the internal system clock (f_{CLK}).

15.4 Initialization

Initialize the contents of each register as necessary while programming.

The initial values of the CPU, internal data RAM, internal instruction RAM, and on-chip peripheral I/O after a reset are shown below.

Table 15-2. Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset (1/6)

Internal Hardware		Register Name	Initial Value After Reset
★ CPU	Program registers	General-purpose register (r0)	00000000H
		General-purpose register (r1 to r31)	Undefined
		Program counter (PC)	00100000H
	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
		Debug interface register (DIR)	00000040H
		Breakpoint control registers 0, 1 (BPC0, BPC1)	Undefined
		Program ID register (ASID)	Undefined
		Breakpoint address setting registers 0, 1 (BPAV0, BPAV1)	Undefined
		Breakpoint address mask registers 0, 1 (BPAM0, BPAM1)	Undefined
		Breakpoint data setting registers 0, 1 (BPDV0, BPDV1)	Undefined
Breakpoint data mask registers 0, 1 (BPD0, BPD1)	Undefined		
Internal instruction RAM, internal data RAM		–	Undefined
On-chip peripheral I/O	Bus control functions	Chip area select control register n (CSCn) (n = 0, 1)	2C11H
		Bus cycle type configuration register n (BCTn) (n = 0, 1)	8888H
		Local bus sizing control register (LBS)	5555H/AAAAH ^{Note 1}
		Endian configuration register (BEC)	0000H
		Line buffer control register n (LBCn) (n = 0, 1)	0000H
		Bus mode control register (BMC)	00H
		Data wait control register n (DWCn) (n = 0, 1)	7777H
		Address setup wait control register (ASC)	FFFFH
		Bus cycle period control register (BCP)	00H
		DMA flyby transfer wait control register (FWC)	7777H
		Bus cycle control register (BCC)	FFFFH
		DMA flyby transfer idle control register (FIC)	3333H
		Cache configuration register (BHC)	0000H
		Instruction cache control register (ICC)	0003H ^{Note 2}
		Instruction cache control register L (ICCL)	03H ^{Note 2}
		Instruction cache control register H (ICCH)	00H

Notes 1. For details, see 4.5.2 (1) Local bus sizing control register (LBS).

2. For details, see 4.9.4 (1) Instruction cache control register (ICC).

Table 15-2. Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset (2/6)

Internal Hardware		Register Name	Initial Value After Reset
On-chip peripheral I/O	Bus control functions	Instruction cache data configuration register (ICD)	Undefined
		Internal instruction RAM mode register (IRAMM)	03H
		System wait control register (VSWC)	77H
	Memory control functions	Page ROM configuration register (PRC)	7000H
		SDRAM configuration register n (SCRn) (n = 1, 3, 4, 6)	30C0H
		SDRAM refresh control register n (RFSn) (n = 1, 3, 4, 6)	0000H
	DMA functions	DMA source address register nH (DSAnH) (n = 0 to 3)	Undefined
		DMA source address register nL (DSAnL) (n = 0 to 3)	Undefined
		DMA destination address register nH (DDAnH) (n = 0 to 3)	Undefined
		DMA destination address register nL (DDAnL) (n = 0 to 3)	Undefined
		DMA transfer count register n (DBCn) (n = 0 to 3)	Undefined
		DMA addressing control register n (DADCn) (n = 0 to 3)	0000H
		DMA channel control register n (DCHCn) (n = 0 to 3)	00H
		DMA terminal count output control register (DTC)	01H
		DMA trigger factor register n (DTFRn) (n = 0 to 3)	00H
		DMA interface control register (DIFC)	00H
	Interrupt/exception control functions	Interrupt control registers (P1IC0, P1IC1, P2IC1 to P2IC5, P5IC0 to P5IC2, P6IC5 to P6IC7, PDIC0 to PDIC15, PLIC0, PLIC1, OVCIC0 to OVCIC5, CCC0IC0, CCC0IC1, CCC1IC0, CCC1IC1, CCC2IC0, CCC2IC1, CCC3IC0, CCC3IC1, CCC4IC0, CCC4IC1, CCC5IC0, CCC5IC1, CMDIC0 to CMDIC3, CC10IC0, CC10IC1, CC11IC0, CC11IC1, CM10IC0, CM10IC1, CM11IC0, CM11IC1, OV1IC0, OV1IC1, UD1IC0, UD1IC1, DMAIC0 to DMAIC3, CSI3IC0, CSI3IC1, COVF3IC0, COVF3IC1, UREIC0, UREIC1, URIC0, URIC1, UTIC0, UTIC1, UIFIC0, UIFIC1, UTOIC0, UTOIC1, ADIC, US0BIC to US2BIC, USP2IC, USP4IC, RSUMIC)	47H
		Interrupt mask register n (IMRn) (n = 0 to 5)	FFFFH
		Interrupt mask register nL (IMRnL) (n = 0 to 5)	FFH
		Interrupt mask register nH (IMRnH) (n = 0 to 5)	FFH
		NMI reset status register (NRS)	00H
		In-service priority register (ISPR)	00H
		External interrupt falling edge specification register n (INTFn) (n = 1, 2, 5, 6, AL)	00H
		External interrupt falling edge specification register DH (INTFDH)	0000H
		External interrupt falling edge specification register DHL (INTFDHL)	00H
		External interrupt falling edge specification register DHH (INTFDHH)	00H
		External interrupt rising edge specification register n (INTRn) (n = 1, AL)	03H
		External interrupt rising edge specification register 2 (INTR2)	3FH
		External interrupt rising edge specification register 5 (INTR5)	07H
		External interrupt rising edge specification register 6 (INTR6)	E0H
		External interrupt rising edge specification register DH (INTRDH)	FFFFH
		External interrupt rising edge specification register DHL (INTRDHL)	FFH
		External interrupt rising edge specification register DHH (INTRDHH)	FFH
Valid edge select register Cn (SESCn) (n = 0 to 3)		00H	
Valid edge select register 1n (SESA1n) (n = 0, 1)		00H	

Table 15-2. Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset (3/6)

Internal Hardware		Register Name	Initial Value After Reset
On-chip peripheral I/O	Clock generation functions	Clock control register (CKC)	03H
		Clock source select register (CKS)	00H
		SSCG control register (SSCGC)	See 8.3.3
		USB clock control register (UCKC)	00H
		Lock register (LOCKR)	01H
		Oscillation stabilization time select register (OSTS)	04H
		Power-save mode register (PSMR)	00H
		Power-save control register (PSC)	00H
	System control	Command register (PRCMD)	Undefined
	Timer/counter functions (timer C)	Timer Cn (TMCn) (n = 0 to 5)	0000H
		Capture/compare registers Cn0, Cn1 (CCCN0, CCCN1) (n = 0 to 5)	0000H
		Timer mode control register Cn0 (TMCCn0) (n = 0 to 5)	00H
		Timer mode control register Cn1 (TMCCn1) (n = 0 to 5)	20H
		Valid edge select register Cn (SESCn) (n = 0 to 3)	00H
		Noise elimination width setting register Cn (NCWCn) (n = 0 to 3)	02H
	Timer/counter functions (timer D)	Timer Dn (TMDn) (n = 0 to 3)	0000H
		Compare register Dn (CMDn) (n = 0 to 3)	0000H
		Timer mode control register Dn (n = 0 to 3)	00H
	Timer/counter functions (timer ENC1)	Timer ENC1n (TMENC1n) (n = 0, 1)	0000H
		Compare register 1n (CM1n) (n = 00, 01, 10, 11)	0000H
		Capture/compare register 1n (CC1n) (n = 00, 01, 10, 11)	0000H
		Timer unit mode register 1n (TUM1n) (n = 0, 1)	00H
		Timer control register 1n (TMC1n) (n = 0, 1)	00H
		Capture/compare control register 1n (CCR1n) (n = 0, 1)	00H
		Valid edge select register 1n (SESA1n) (n = 0, 1)	00H
		Prescaler mode register 1n (PRM1n) (n = 0, 1)	07H
		Status register 1n (STATUS1n) (n = 0, 1)	00H
		Noise elimination width setting register 1n (NCW1n) (n = 0, 1)	02H
	Serial interface functions (USBF)	UF0 EP0NAK register (UF0E0N)	00H
		UF0 EP0NAKALL register (UF0E0NA)	00H
		UF0 EPNAK register (UF0EN)	00H
		UF0 EPNAK mask register (UF0ENM)	00H
		UF0 SNDSIE register (UF0SDS)	00H
		UF0 CLR request register (UF0CLR)	00H
		UF0 SET request register (UF0SET)	00H
		UF0 EP status n register (UF0EPSn) (n = 0 to 2)	00H
		UF0 INT status n register (UF0ISn) (n = 0 to 4)	00H
		UF0 INT mask n register (UF0IMn) (n = 0 to 4)	00H
		UF0 INT clear n register (UF0ICn) (n = 0 to 4)	FFH
		UF0 INT & DMARQ register (UF0IDR)	00H
		UF0 DMA status n register (UF0DMSn) (n = 0, 1)	00H
		UF0 FIFO clear n register (UF0FICn) (n = 0, 1)	00H

Table 15-2. Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset (4/6)

Internal Hardware		Register Name	Initial Value After Reset
On-chip peripheral I/O	Serial interface functions (USBF)	UF0 data end register (UF0DEND)	00H
		UF0 GPR register (UF0GPR)	00H
		UF0 mode control register (UF0MODC)	00H
		UF0 mode status register (UF0MODS)	00H
		UF0 active interface number register (UF0AIFN)	00H
		UF0 active alternative setting register (UF0AAS)	00H
		UF0 alternative setting status register (UF0ASS)	00H
		UF0 endpoint n interface mapping register (UF0EnIM) (n = 1 to 4, 7, 8)	00H
		UF0 EP0 read register (UF0E0R)	Undefined
		UF0 EP0 length register (UF0E0L)	00H
		UF0 EP0 setup register (UF0E0ST)	00H
		UF0 EP0 write register (UF0E0W)	Undefined
		UF0 bulk out n register (UF0BO _n) (n = 1, 2)	Undefined
		UF0 bulk out n length register (UF0BO1L) (n = 1, 2)	00H
		UF0 bulk in n register (UF0BI _n) (n = 1, 2)	Undefined
		UF0 interrupt n register (UF0INT _n) (n = 1, 2)	Undefined
		UF0 device status register L (UF0DSTL)	00H
		UF0 EP _n status register L (UF0EnSL) (n = 0 to 4, 7, 8)	00H
		UF0 address register (UF0ADRS)	00H
		UF0 configuration register (UF0CNF)	00H
		UF0 interface n register (UF0IF _n) (n = 0 to 4)	00H
		UF0 descriptor length register (UF0DSC _L)	00H
		UF0 device descriptor register n (UF0DD _n) (n = 0 to 17)	00H
		UF0 configuration/interface/endpoint descriptor register n (UF0CI _{En}) (n = 0 to 255)	Undefined
		USB function 0 DMA channel select register (UF0CS)	0000H
		USB function 0 buffer control register (UF0BC)	00H
		Serial interface functions (UARTB)	UARTB _n control register 0 (UBnCTL0) (n = 0, 1)
	UARTB _n control register 2 (UBnCTL2) (n = 0, 1)		FFFFH
	UARTB _n status register (UBnSTR) (n = 0, 1)		00H
	UARTB _n transmit data register (UBnTX) (n = 0, 1)		FFH
	UARTB _n receive data register AP (UBnRXAP) (n = 0, 1)		00FFH
	UARTB _n receive data register (UBnRX) (n = 0, 1)		FFH
	UARTB _n FIFO control register 0 (UBnFIC0) (n = 0, 1)		00H
	UARTB _n FIFO control register 1 (UBnFIC1) (n = 0, 1)		00H
UARTB _n FIFO control register 2 (UBnFIC2) (n = 0, 1)	0000H		
UARTB _n FIFO control register 2L (UBnFIC2L) (n = 0, 1)	00H		
UARTB _n FIFO control register 2H (UBnFIC2H) (n = 0, 1)	00H		
UARTB _n FIFO status register 0 (UBnFIS0) (n = 0, 1)	00H		
UARTB _n FIFO status register 1 (UBnFIS1) (n = 0, 1)	10H		

Table 15-2. Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset (5/6)

Internal Hardware		Register Name	Initial Value After Reset	
On-chip peripheral I/O	Serial interface functions (CSI3)	Clocked serial interface mode register 3n (CSIM3n) (n = 0, 1)	00H	
		Clocked serial interface clock select register 3n (CSIC3n) (n = 0, 1)	07H	
		Receive data buffer register 3n (SIRB3n) (n = 0, 1)	0000H	
		Receive data buffer register 3nL (SIRB3nL) (n = 0, 1)	00H	
		Receive data buffer register 3nH (SIRB3nH) (n = 0, 1)	00H	
		Transmit data CSI buffer register 3n (SFDB3n)	0000H	
		Transmit data CSI buffer register 3nL (SFDB3nL)	00H	
		Transmit data CSI buffer register 3nH (SFDB3nH)	00H	
		CSIBUF status register 3n (SFA3n) (n = 0, 1)	20H	
		Transfer data length select register 3n (CSIL3n) (n = 0, 1)	00H	
		Transfer data number specification register 3n (SFN3n) (n = 0, 1)	00H	
		A/D converter	A/D converter mode register n (ADMn) (n = 0 to 2)	00H
			ADC trigger select register (ADTS)	00H
	A/D conversion result register n (ADCRn) (10 bits) (n = 0 to 7)		Undefined	
	A/D conversion result register nH (ADCRnH) (8 bits) (n = 0 to 7)		Undefined	
	PWM	PWM control register n (PWMCn) (n = 0, 1)	08H	
		PWM modulo register n (PWMn) (n = 0, 1)	0000H	
		PWM modulo register Ln (PWMLn) (n = 0, 1)	00H	
		PWM modulo register Hn (PWMHn) (n = 0, 1)	00H	
	Port functions	Ports (P1, P2, P5 to P7, PCS, PCT, PCM, PCD)	Undefined	
		Port (PAL)	Undefined	
		Port (PALL)	Undefined	
		Port (PALH)	Undefined	
		Port (PAH)	Undefined	
		Port (PAHL)	Undefined	
		Port (PAHH)	Undefined	
		Port (PDH)	Undefined	
		Port (PDHL)	Undefined	
		Port (PDHH)	Undefined	
		Mode register (PM1, PM2, PM5 to PM7, PMCS, PMCT, PMCM, PMCD)	FFH	
		Mode register (PMAL)	FFFFH	
		Mode register (PMALL)	FFH	
		Mode register (PMALH)	FFH	
		Mode register (PMAH)	FFFFH	
		Mode register (PMAHL)	FFH	
		Mode register (PMAHH)	FFH	
		Mode register (PMDH)	FFFFH	
		Mode register (PMDHL)	FFH	
		Mode register (PMDHH)	FFH	
		Mode control register (PMC1, PMC5 to PMC7)	00H	
		Mode control register (PMC2)	01H	
		Mode control register (PMCCS)	FFH	

Table 15-2. Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset (6/6)

Internal Hardware		Register Name	Initial Value After Reset
On-chip peripheral I/O	Port functions	Mode control register (PMCCCT)	BFH
		Mode control register (PMCCM)	3DH
		Mode control register (PMCCD)	0FH
		Mode control register (PMCAL)	0002H
		Mode control register (PMCALL)	02H
		Mode control register (PMCALH)	00H
		Mode control register (PMCAH)	03FFH
		Mode control register (PMCAHL)	FFH
		Mode control register (PMCAHH)	03H
		Mode control register (PMCDH)	0000H
		Mode control register (PMCDHL)	00H
		Mode control register (PMCDHH)	00H
		Function control registers (PFC1, PFC2, PFC5 to PFC7, PFCCS, PFCCT, PFCCM)	00H
		Function control register L (PFCALL)	03H
		Function control register (PFCDH)	0000H
		Function control register (PFCDHL)	00H
		Function control register (PFCDHH)	00H
		External interrupt falling edge specification register n (INTFn) (n = 1, 2, 5, 6, AL)	00H
		External interrupt falling edge specification register DH (INTFDH)	0000H
		External interrupt falling edge specification register DHL (INTFDHL)	00H
		External interrupt falling edge specification register DHH (INTFDHH)	00H
		External interrupt rising edge specification register n (INTRn) (n = 1, AL)	03H
		External interrupt rising edge specification register 2 (INTR2)	3FH
		External interrupt rising edge specification register 5 (INTR5)	07H
		External interrupt rising edge specification register 6 (INTR6)	E0H
		External interrupt rising edge specification register DH (INTRDH)	FFFFH
		External interrupt rising edge specification register DHL (INTRDHL)	FFH
		External interrupt rising edge specification register DHH (INTRDHH)	FFH
		Noise elimination width setting register Cn (NCWCn) (n = 0 to 3)	02H
		Noise elimination width setting register 1n (NCW1n) (n = 0, 1)	02H

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 write 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.

CHAPTER 16 DEBUG FUNCTION (DCU)

The debug control unit (DCU) consists of three function units: an execution control unit (RCU) that realizes communication with JTAG and execution of debug processing, a trace control unit (TRCU) that implements trace functions, and a trigger event unit (TEU) that implements event detection functions. On-chip debugging of the V850E/ME2 can be executed by connecting an N-Wire type IE.

Caution The debug function is supported by the V850E/ME2, but whether this function can be used or not depends on the debugger used.

16.1 Functional Outline

16.1.1 Debug function

(1) Debug interface

This interface establishes communication with the host machine by using the $\overline{\text{DRST}}$, DCK, DMS, DDI, and DDO signals, via an N-Wire type IE. The communication specifications of JTAG are used for this interface. It does not support a boundary scan function.

(2) On-chip debug

On-chip debugging can be performed if wiring and connectors for debugging are provided on the target system.

Connect an N-Wire type IE to the connector for debugging.

(3) Forced reset function

The V850E/ME2 can be forcibly reset.

(4) Break reset function

The CPU can be started in the debug mode immediately after resetting the CPU has been cleared.

(5) Forced break function

Execution of the user program can be forcibly stopped (however, the handler of the illegal instruction code exception (first address: 00000060H) cannot be used).

(6) Debug monitor function

During debugging, a memory space for debugging, different from the user memory space, is used (background monitor format). The user program can be executed starting from any address.

While execution of the user program is stopped, the user resources (such as memory and I/O) can be read/written, and the user program can be downloaded.

★ (7) Mask function

- (a) NMI and all maskable interrupt request signals can be masked.
- (b) When the debugger is connected, the $\overline{\text{RESET}}$ pin input on the target board is masked by default (the $\overline{\text{RESET}}$ pin input is masked when the debugger is started after power application to the V850E/ME2). The $\overline{\text{RESET}}$ pin input can be unmasked from the debugger. If a signal is input to the $\overline{\text{RESET}}$ pin during debugging (during RUN execution), however, the following problems may occur.
 - The break function may malfunction. If this happens, restart.
 - Trace data may be illegal before and after $\overline{\text{RESET}}$ pin input. Recovery will occur after the $\overline{\text{RESET}}$ signal has been released.

16.1.2 Trace function**(1) PC trace (branch trace) function**

All branches (transition of processing) that occur during user program execution can be traced. The trace sources can be selected from 12 types of branch sources that are classified by function, and PC trace can be started from execution of an instruction at any address, and the trace source can be changed. Two trace start triggers are available.

(2) Data trace function

A data access issued by the CPU to any address can be traced in a range of 1 KB to 4 bytes. Read or written data can be traced, and two data trace points are available. However, a data access issued by the DMAC cannot be traced.

(3) Real-time trace mode

Branch and data access can be traced during real-time execution of the user program. The trace packet of the trace source detected is stored in a trace buffer, and output from trace interface pins (TRCCLK, TRCDATA0 to TRCDATA3, and TRCEND) (some trace packets may not be traced if no more trace packets can be stored in the trace buffer).

(4) Full trace mode (non-real-time trace mode)

All branches and data accesses of the user program can be traced. In the full trace mode, the pipeline of the CPU is temporarily held and instruction execution is stopped to secure the time of trace data output from trace interface pins, so that all trace packets can be correctly traced.

16.1.3 Event function

(1) Instruction event detection function

Event detection (10 events) via size comparison by the execution PC and range event detection (up to four pairs with each pair consisting of two events) of the execution PC can be executed.

If an instruction event source is used as a break source, two breakpoints before execution of the instruction at which an event is detected and eight breakpoints after instruction execution can be detected.

(2) Access event detection function

Events can be detected as follows.

- Comparison of access addresses (4 addresses)
- Range of access address (up to two pairs with each pair consisting of two addresses)
- Match or mismatch of access data
- Data of specific bit by masking data
- Access size

An access event source is detected after access. If an access event source is used as a break source, a break occurs after several instructions have been executed after the instruction that issued the access that caused event detection.

(3) Sequential event detection function

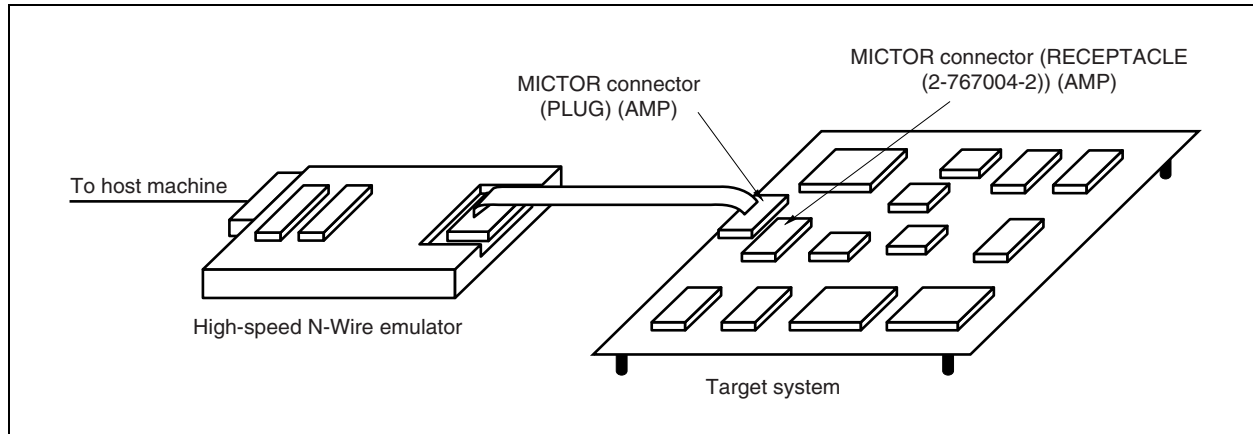
An event can be detected when up to four stages of events have successively occurred or an event that clears successive occurrence of events can be detected.

Sequential events can be counted by using a 12-bit pass counter.

16.2 Connection with N-Wire Type IE

A connector for the IE and a connection circuit must be provided on the target system.

Figure 16-1. Connecting N-Wire Type IE



16.2.1 IE connector (on target system side)

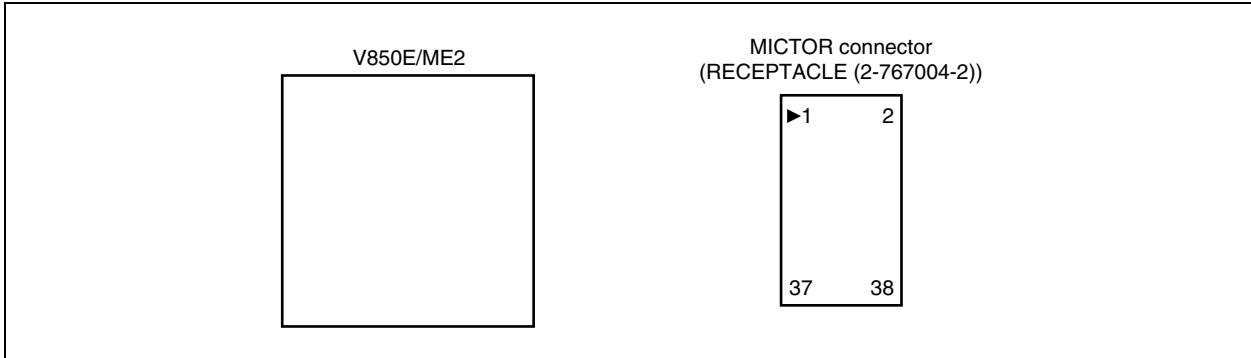
The following table shows the pin functions of the IE connector (on the target system side).

Table 16-1. IE Connector Pin Function (on Target System Side)

Pin No.	Pin Name	I/O	Pin Function
1	GND	–	–
2	GND	–	–
3	DCK	Output	Clock output for debug serial interface
4	V _{DD}	–	+3.3 V input (for monitoring power to target)
5	DMS	Output	Transfer mode select output for debug serial interface
6	$\overline{\text{DRST}}$	Output	DCU reset output
7	DDI	Output	Data output for debug serial interface
8	PORT0_OUT	Output	General-purpose control signal 0 output
9	DDO	Input	Data input for debug serial interface
10	PORT1_OUT	Output	General-purpose control signal 1 output
11	(Reserved 1)	–	(Leave this pin open)
12	PORT2_OUT	Output	General-purpose control signal 2 output
13	(Reserved 2)	–	(Leave this pin open)
14	PORT0_IN	Input	General-purpose control signal 0 input
15	(Reserved 3)	–	(Leave this pin open)
16	PORT1_IN	Input	General-purpose control signal 1 input
17	TRCCLK	Input	Trace clock input
18	PORT2_IN	Input	General-purpose control signal 2 input
19	TRCEND	Input	Trace data end input
20	TRCCE	Input	Trace packet compression enable signal input
21	TRCDATA0	Input	Trace data 0 input
22	TRCDATA8	Input	Trace data 8 input
23	TRCDATA1	Input	Trace data 1 input
24	TRCDATA9	Input	Trace data 9 input
25	TRCDATA2	Input	Trace data 2 input
26	TRCDATA10	Input	Trace data 10 input
27	TRCDATA3	Input	Trace data 3 input
28	TRCDATA11	Input	Trace data 11 input
29	TRCDATA4	Input	Trace data 4 input
30	TRCDATA12	Input	Trace data 12 input
31	TRCDATA5	Input	Trace data 5 input
32	TRCDATA13	Input	Trace data 13 input
33	TRCDATA6	Input	Trace data 6 input
34	TRCDATA14	Input	Trace data 14 input
35	TRCDATA7	Input	Trace data 7 input
36	TRCDATA15	Input	Trace data 15 input
37	GND	–	–
38	GND	–	–

Remark Cautions are given on the next page.

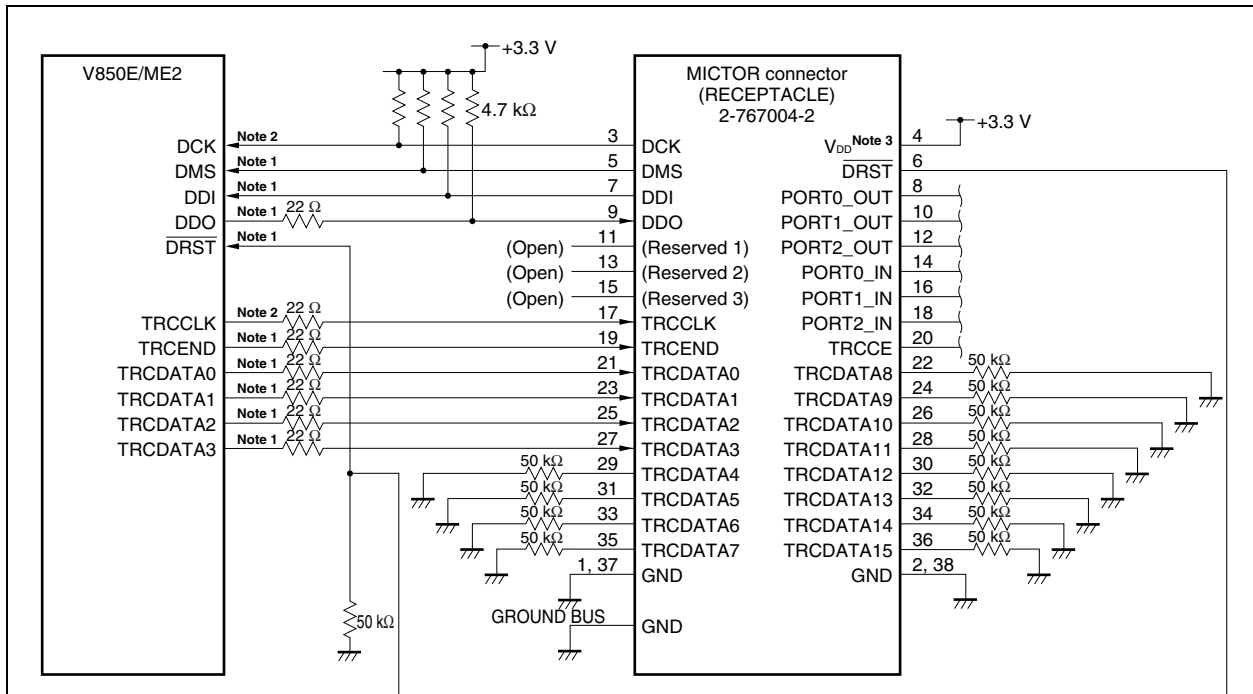
- Cautions**
1. The connection of pins not supported by the V850E/ME2 depends on the ICE used.
 2. The pattern on the target board must satisfy the following conditions to support high-speed interfacing.
 - Lay out the pattern with the odd pins facing the device (V850E/ME2).
 - Keep the pattern length to within 1.97 inches (50 mm).
 - Shield the clock signal with GND.



16.2.2 Recommended circuit example

The following figure shows an example of the recommended circuit of the IE connector (on the target system side).

Figure 16-2. Example of Recommended IE Connection Circuit



- Notes**
1. Keep the pattern length to within 1.97 inches (50 mm).
 2. Shield the DCK and TRCCLK signals by GND.
 3. For detecting power to the target board

Caution The recommended circuit example shown above assumes that a 3.3 V interface is used.

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

Parameter	Symbol	Conditions	Ratings	Unit
Supply voltage	IV_{DD}	IV_{DD} pin	-0.5 to +2.0	V
	IV_{SS}	IV_{SS} pin	-0.5 to +0.5	V
	EV_{DD}	EV_{DD} pin, $EV_{DD} \geq IV_{DD}$	-0.5 to +4.6	V
	EV_{SS}	EV_{SS} pin	-0.5 to +0.5	V
	$OSCV_{DD}$	$OSCV_{DD}$ pin	-0.5 to +4.6	V
	$OSCV_{SS}$	$OSCV_{SS}$ pin	-0.5 to +0.5	V
	$PLLV_{DD}$	$PLLV_{DD}$ pin	-0.5 to +2.0	V
	$PLLV_{SS}$	$PLLV_{SS}$ pin	-0.5 to +0.5	V
	UV_{DD}	UV_{DD} pin	-0.5 to +4.6	V
	AV_{DD}	AV_{DD} pin, $AV_{DD} < EV_{DD} \pm 0.5\text{ V}$	-0.5 to +4.6	V
	AV_{SS}	AV_{SS} pin	-0.5 to +0.5	V
Input voltage	V_I	Except for X1 pin, $V_I < EV_{DD} + 0.3\text{ V}$	-0.5 to +4.6	V
Clock input voltage	V_K	X1 pin	-0.5 to $OSCV_{DD} + 0.5^{\text{Note}}$	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	$EV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$	-0.5 to $EV_{DD} + 0.5^{\text{Note}}$	V
Analog input voltage	V_{WASN}	ANI0 to ANI7 pins, $AV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$	-0.3 to $AV_{DD} + 0.3^{\text{Note}}$	V
A/D converter reference input voltage	AV_{REFP}		-0.3 to $AV_{DD} + 0.3^{\text{Note}}$	V
	AV_{REFM}		-0.3 to +0.3	V
Operating ambient temperature	T_A	30 pF < output pin load capacitance \leq 50 pF	-40 to +70	$^\circ\text{C}$
		Output pin load capacitance \leq 30 pF	-40 to +85	$^\circ\text{C}$
Storage temperature	T_{stg}		-60 to +150	$^\circ\text{C}$

Note Be sure not to exceed the absolute maximum ratings (MAX. value) of each supply voltage.

- Cautions 1.** Do not directly connect the output (or I/O) pins of IC products to each other, or to IV_{DD} , EV_{DD} , and GND. Direct connection of the output pins between an IC product and an external circuit is possible, if the output pins can be set to the high-impedance state and the output timing of the external circuit is designed to avoid output conflict.
- 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 indicated for DC Characteristics and AC Characteristics represent the quality assurance range during normal operation.

Capacitance ($T_A = 25^\circ\text{C}$, $IV_{DD} = IV_{SS} = EV_{DD} = EV_{SS} = OSCV_{DD} = OSCV_{SS} = PLLV_{DD} = PLLV_{SS} = UV_{DD} = AV_{DD} = AV_{SS} = 0\text{ V}$)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Input capacitance	C_i	$f_c = 1\text{ MHz}$			15	pF
I/O capacitance	C_{io}	Unmeasured pins returned to 0 V			15	pF
Output capacitance	C_o				15	pF

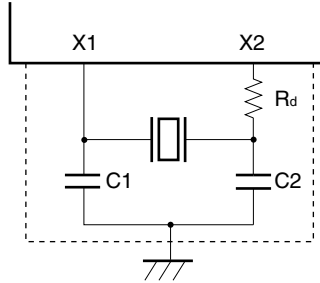
Operating Conditions

Part Number	Internal Operation Clock Frequency (fx)	Operating Ambient Temperature (T_A)	Supply Voltage (V_{DD})
$\mu\text{PD7030111GM-10-UEU}$	10 to 100 MHz	When external pin load capacitance $C_L = 50\text{ pF}$, -40 to +70°C	$IV_{DD} = 1.5\text{ V} \pm 0.15\text{ V}$ $PLLV_{DD} = 1.5\text{ V} \pm 0.15\text{ V}$
		When external pin load capacitance $C_L = 30\text{ pF}$, -40 to +85°C	$EV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$ $OSCV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$ $UV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$ $AV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$
$\mu\text{PD7030111GM-13-UEU}$	10 to 133 MHz	When external pin load capacitance $C_L = 50\text{ pF}$, -40 to +70°C	$IV_{DD} = 1.4\text{ to }1.65\text{ V}$ $PLLV_{DD} = 1.4\text{ to }1.65\text{ V}$ $EV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$ $OSCV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$ $UV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$ $AV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$
		When external pin load capacitance $C_L = 30\text{ pF}$, -40 to +85°C	
$\mu\text{PD7030111GM-15-UEU}$	10 to 150 MHz	When external pin load capacitance $C_L = 50\text{ pF}$, -40 to +70°C	

Recommended Oscillator

(a) Ceramic resonator

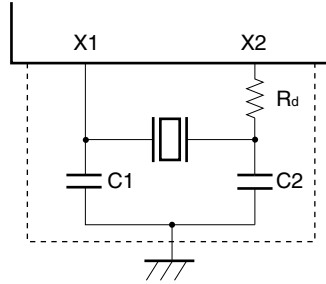
(i) Murata Mfg. Co., Ltd. ($T_A = -40$ to $+85^\circ\text{C}$)



Type	Part Number	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 mounting	CSTCE10M0G55-R0	10.000	On-chip	On-chip	0	3.0	3.6	0.07
	CSTCE12M5G55-R0	12.500	On-chip	On-chip	0	3.0	3.6	0.08
	CSTCE16M6V53-R0	16.625	On-chip	On-chip	0	3.0	3.6	0.04
	CSTCE18M0V53-R0	18.000	On-chip	On-chip	0	3.0	3.6	0.03
	CSTCE18M7V53-R0	18.750	On-chip	On-chip	0	3.0	3.6	0.03

- Cautions**
1. Connect the oscillator as close as possible to the X1 and X2 pins.
 2. Do not route the wiring near broken lines.
 3. Sufficiently evaluate the matching between the $\mu\text{PD703111}$ and the resonator.

(ii) TDK ($T_A = -40$ to $+85^\circ\text{C}$)



Type	Part Number	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)	
Lead	FCR10.0MC5	10.0	On-chip	On-chip	0	3.0	3.6	0.076
Surface mounting	CCR18.0MXC7	18.0	On-chip	On-chip	0	3.0	3.6	0.9

- Cautions**
1. Connect the oscillator as close as possible to the X1 and X2 pins.
 2. Do not route the wiring near broken lines.
 3. Sufficiently evaluate the matching between the $\mu\text{PD703111}$ and the resonator.

DC Characteristics (T_A = –40 to +85°C, EV_{DD} = 3.3 V ±0.3 V, EV_{SS} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Input voltage, high	V _{IH}	Except for Notes 1, 2	2.0		EV _{DD} + 0.3	V	
		Note 1	0.75EV _{DD}		EV _{DD} + 0.3	V	
		Note 2	2.0		UV _{DD} + 0.3	V	
Input voltage, low	V _{IL}	Except for Notes 1, 2	–0.5		0.8	V	
		Note 1	–0.5		0.2EV _{DD}	V	
		Note 2	–0.5		0.8	V	
Schmitt-triggered input threshold voltage	V _T ⁺	Note 1 , rising edge		1.9		V	
	V _T [–]	Note 1 , falling edge		1.3		V	
Schmitt-triggered input hysteresis width	V _T ⁺ –V _T [–]	Note 1	0.3	0.6		V	
Output voltage, high	V _{OH}	Except for Note 2 I _{OH} = –2.5 mA	0.8EV _{DD}			V	
		I _{OH} = –100 μA	EV _{DD} – 0.4			V	
		Note 2 , R _L = 15 kΩ (EV _{SS} connection)	2.8			V	
Output voltage, low	V _{OL}	Except for Note 2 , I _{OL} = 2.5 mA			0.45	V	
		Note 2 , R _L = 15 kΩ (UV _{DD} connection)			0.3	V	
Input leakage current, high	I _{LIH}	V _I = EV _{DD} , except for Note 3			10	μA	
Input leakage current, low	I _{LIL}	V _I = 0 V, except for Note 3			–10	μA	
Output leakage current, high	I _{LOH}	V _O = EV _{DD}			10	μA	
Output leakage current, low	I _{LOL}	V _O = 0 V			–10	μA	
Analog pin input leakage current	I _{LWASN}	Note 3			±10	μA	
Supply current	Normal	I _{DD1}	IV _{DD} + PLLV _{DD} pins		0.82 × f _x + 10	0.90 × f _x + 117	mA
			EV _{DD} pin ^{Note 4}		0.33 × F _{BUS} + 3.2	0.66 × F _{BUS} + 6.4 ^{Note 5}	mA
	HALT	I _{DD2}	IV _{DD} + PLLV _{DD} pins		0.82 × f _x + 4	0.85 × f _x + 114	mA
			EV _{DD} pin ^{Note 4}		0.33 × F _{BUS} + 3.2	0.66 × F _{BUS} + 6.4 ^{Note 5}	mA
	IDLE	I _{DD3}	IV _{DD} + PLLV _{DD} pins		0.015 × f _x + 10		mA
			EV _{DD} pin ^{Note 4}		500		μA
	STOP	I _{DD4}	IV _{DD} + PLLV _{DD} pins		4		mA
			EV _{DD} pin ^{Note 4}		500		μA

- Notes**
- P11/SCK0/INTP11, P12/RXD0/SI0, P20/NMI, P21/RXD1/INTP21, P23/SCK1/INTP23, P24/SI1/INTP24, PCM1, RESET
 - UDM, UDP
 - ANIO to ANI7
 - Current value with no load
 - Calculate the current value with a load using the following expression.

$$\text{Current value per pin } (\mu\text{A}) = 3.63 \times C_L \times F$$

C_L: Load capacitance (pF) F: Pin average operating frequency

- Remarks**
- TYP. values are reference values for when T_A = 25°C, EV_{DD} = 3.3 V. The current that flows through pull-up resistors is not included.
 - f_x: Main clock frequency (MHz)
F_{BUS}: USCLK frequency (MHz)

AC Characteristics

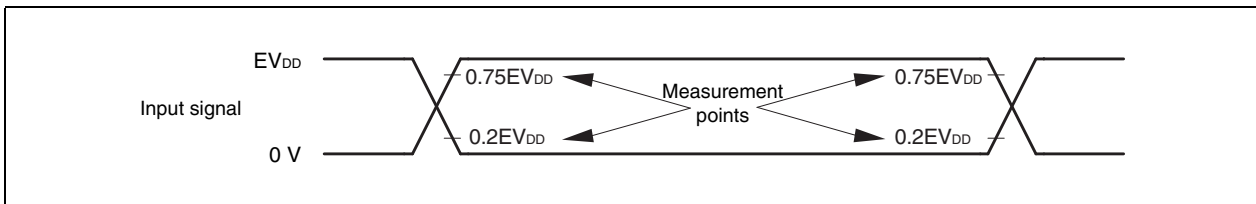
($T_A = -40$ to $+85^\circ\text{C}$, $I_{VDD} = PLLV_{DD} = 1.5\text{ V} \pm 0.15\text{ V}^{\text{Note}}$, $E_{VDD} = OSCV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$, $I_{VSS} = EV_{SS} = PLLV_{SS} = OSCV_{SS} = 0\text{ V}$, output pin load capacitance: $C_L = 30\text{ pF}$)

($T_A = -40$ to $+70^\circ\text{C}$, $I_{VDD} = PLLV_{DD} = 1.5\text{ V} \pm 0.15\text{ V}^{\text{Note}}$, $E_{VDD} = OSCV_{DD} = 3.3\text{ V} \pm 0.3\text{ V}$, $I_{VSS} = EV_{SS} = PLLV_{SS} = OSCV_{SS} = 0\text{ V}$, output pin load capacitance: $C_L = 50\text{ pF}$)

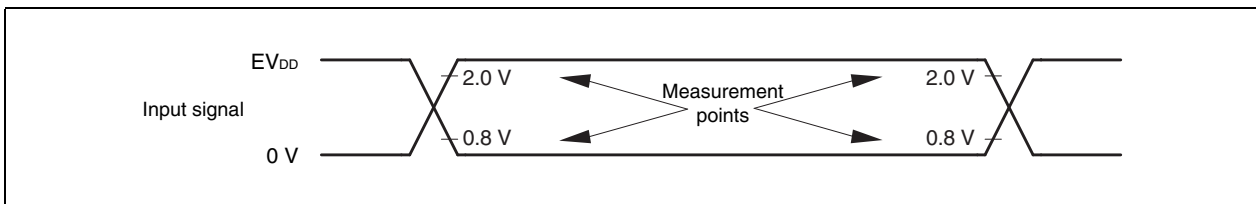
Note Consider the operation conditions when operating at 133.34 to 150 MHz.

AC test input measurement points

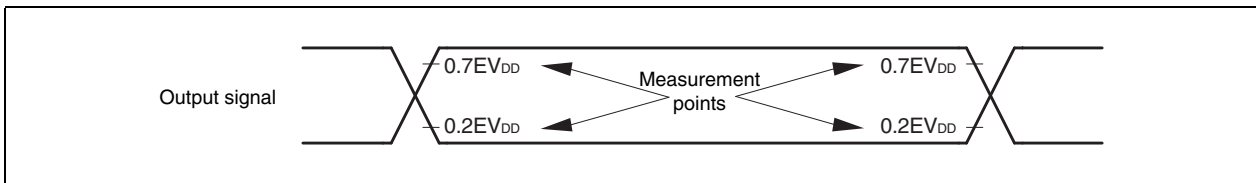
- (a) P11/SCK0/INTP11, P12/RXD0/SI0, P20/NMI, P21/RXD1/INTP21, P23/SCK1/INTP23, P24/SI1/INTP24, PCM1, RESET

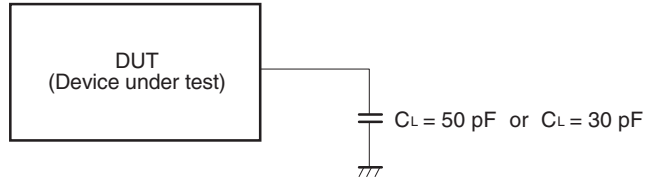


- (b) Other than (a)



AC test output measurement points



Load conditions**Caution At -40 to +70°C:**

If the load capacitance exceeds 50 pF due to the circuit configuration, make the load capacitance of this device 50 pF or lower by inserting a buffer, etc.

At -40 to +85°C:

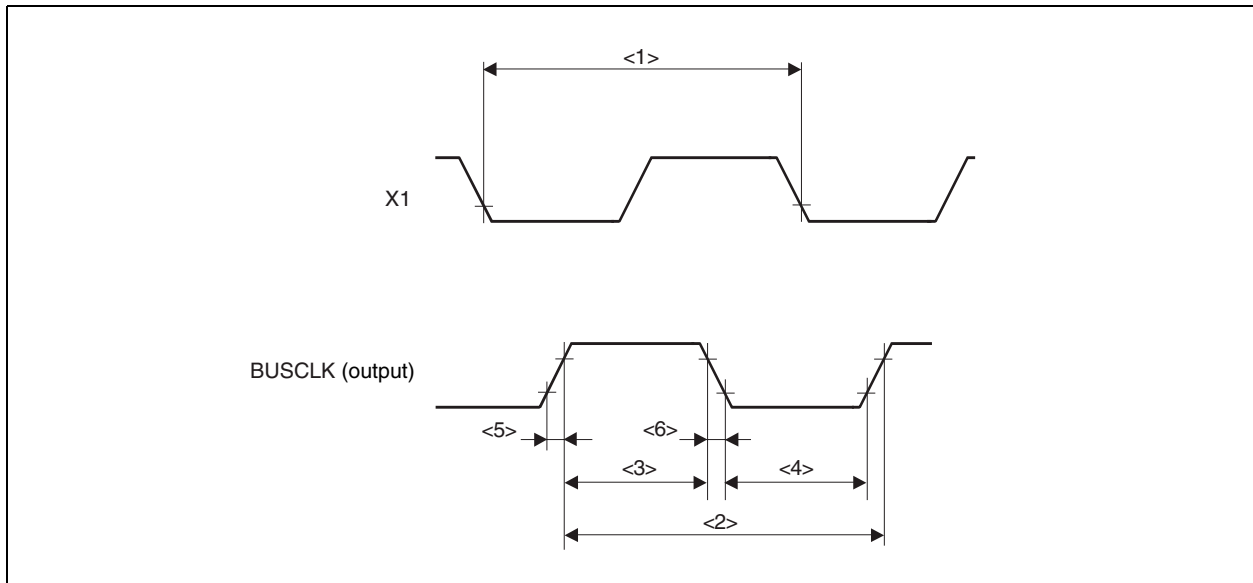
If the load capacitance exceeds 30 pF due to the circuit configuration, make the load capacitance of this device 30 pF or lower by inserting a buffer, etc.

(1) Clock timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
X1 input cycle	<1> t_{CYX}	Including resonator error	52.5	100	ns
BUSCLK output cycle	<2> t_{CYK}	Output load capacitance ≤ 30 pF, $T_A = -40$ to $+85^\circ\text{C}$	14.3	400 ^{Note}	ns
		30 pF < output load capacitance ≤ 50 pF, $T_A = -40$ to $+70^\circ\text{C}$	19.0	400 ^{Note}	ns
BUSCLK high-level width	<3> t_{WKH1}		$0.5T - 2$	$0.5T + 2$	ns
BUSCLK low-level width	<4> t_{WKL1}		$0.5T - 2$	$0.5T + 2$	ns
BUSCLK rise time	<5> t_{KR1}			3	ns
BUSCLK fall time	<6> t_{KF1}			3	ns

Note The output cycle after SSCG output is selected by the CKS register.

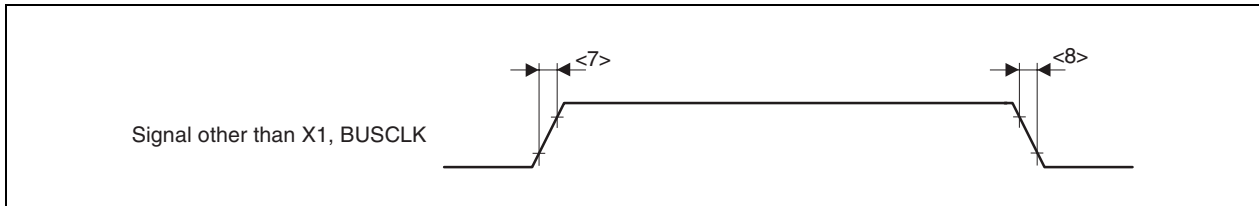
Remark $T = t_{CYK}$



(2) Output waveform (other than X1, BUSCLK)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Output rise time	<7>	toR	Other than Note	6	ns
			Note	8	ns
Output fall time	<8>	toF	Other than Note	6	ns
			Note	8	ns

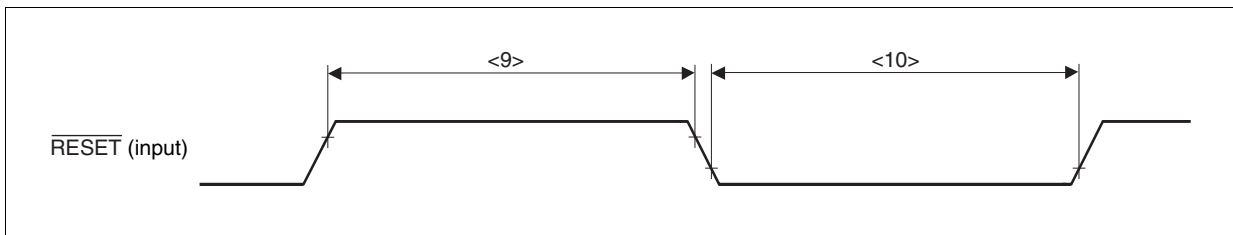
Note P10/UCLK/INTP10, P12/RXD0/SIO, P20/NMI, P21/RXD1/INTP21, P22/TXD1/INTP22, P24/SI1/INTP24, P50/DMARQ0/INTP50, P51/DMAAK0/INTP51, P52/TC0/INTP52, P53/DMARQ1/TIC0/INTPC00, P54/DMAAK1/INTPC01, P55/TC1/TOC0, P65/INTPC10/TIC1/INTP65, P66/INTPC11/INTP66, P67/TOC1/INTP67, P72/DMARQ2/INTPC20/TIC2, P73/DMAAK2/INTPC21, P74/TC2/TOC2, P75/DMARQ3/INTPC30/TIC3, P76/DMAAK3/INTPC31, P77/TC3/TOC3



(3) Reset timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
RESET pin high-level width	<9>	t _{WRSH}	500		ns
RESET pin low-level width	<10>	t _{WRSL}	Including oscillation stabilization time at power on and STOP mode release. However, when the oscillation stabilization time exceeds 100 μs, secure the necessary oscillation stabilization time.	100	μs
			Excluding at power on and STOP mode release.	100	μs

Caution Sufficiently evaluate the oscillation stabilization time.



(4) SRAM, external ROM, external I/O access timing

(a) Access timing (SRAM, external ROM, external I/O) (1/2)

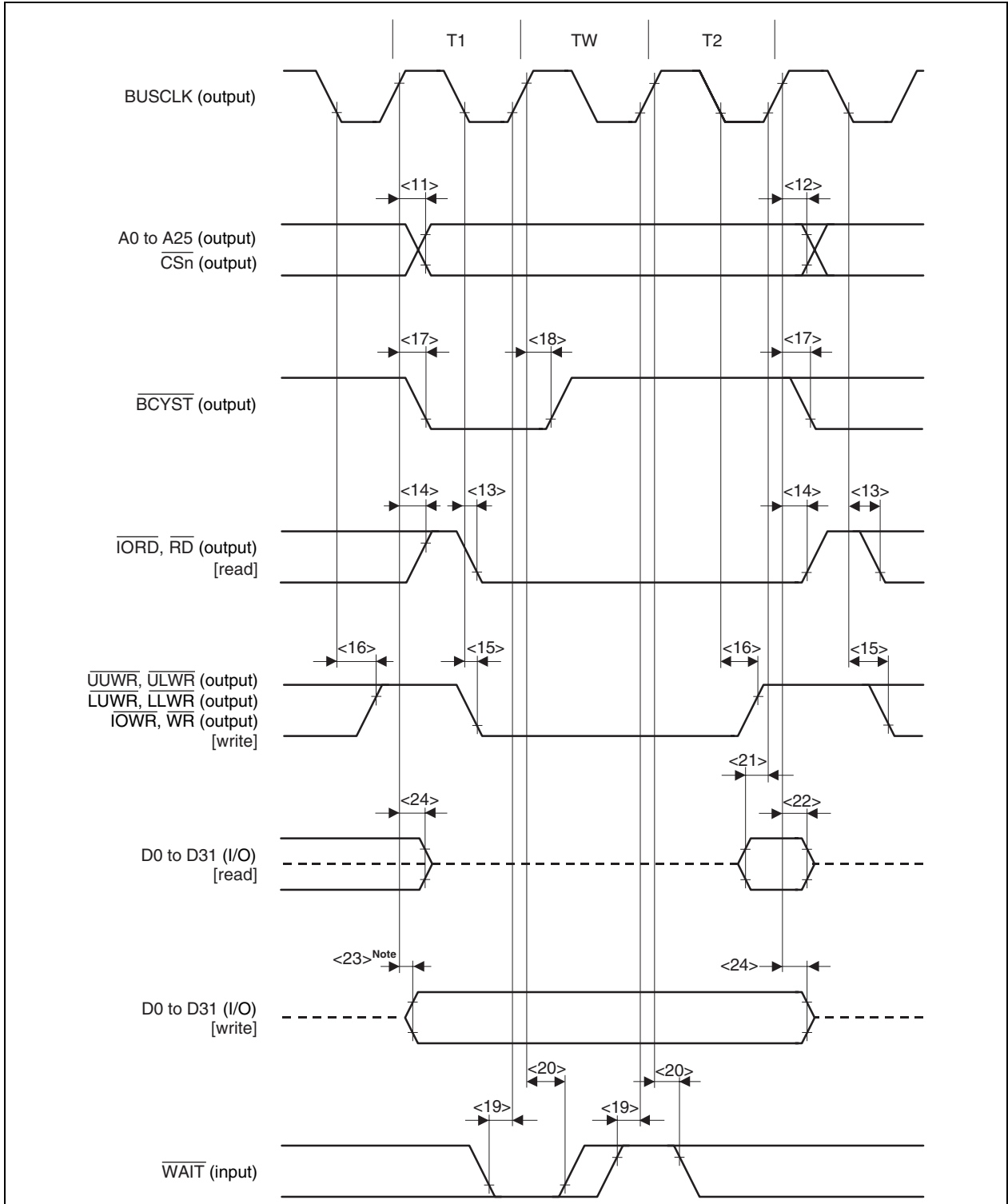
Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Address, \overline{CSn} output delay time (from BUSCLK \uparrow)	<11> t_{DKA}		2	11	ns
Address, \overline{CSn} output hold time (from BUSCLK \uparrow)	<12> t_{HKA}		2	11	ns
\overline{RD} , \overline{IORD} delay time (from BUSCLK \downarrow)	<13> t_{DKRDL}		1	11	ns
\overline{RD} , \overline{IORD} delay time (from BUSCLK \uparrow)	<14> t_{HKRDH}		2	11	ns
\overline{xxWR} , \overline{IOWR} , \overline{WR} delay time (from BUSCLK \downarrow)	<15> t_{DKWRL}		2	11	ns
\overline{xxWR} , \overline{IOWR} , \overline{WR} delay time (from BUSCLK \downarrow)	<16> t_{HKWRH}		1	11	ns
\overline{BCYST} delay time (from BUSCLK \uparrow)	<17> t_{DKBSL}		2	11	ns
\overline{BCYST} delay time (from BUSCLK \uparrow)	<18> t_{HKBSH}		2	11	ns
\overline{WAIT} setup time (to BUSCLK \uparrow)	<19> t_{SWK}		6		ns
\overline{WAIT} hold time (from BUSCLK \uparrow)	<20> t_{HKW}		2		ns
Data input setup time (to BUSCLK \uparrow)	<21> t_{SKID}		6		ns
Data input hold time (from BUSCLK \uparrow)	<22> t_{HKID}		2		ns
Data output delay time (from BUSCLK \downarrow)	<23> t_{DKOD1}		2	11	ns
Data float delay time (from BUSCLK \uparrow)	<24> t_{HKOD}		2	11	ns

Remarks 1. Observe at least one of the data input hold times t_{HRDID} and t_{HKID} .

2. $n = 0$ to 7

$xx = UU, UL, LU, LL$

(a) Access timing (SRAM, external ROM, external I/O) (2/2)



Note When a write cycle is executed without inserting a T0 state immediately after a read cycle, the data output timing is delayed by a half clock (in synchronization with the falling edge of BUSCLK).

- Remarks**
1. Timing when the number of waits set by the DWC0 and DWC1 registers is 0.
 2. Broken lines indicate high impedance.
 3. n = 0 to 7

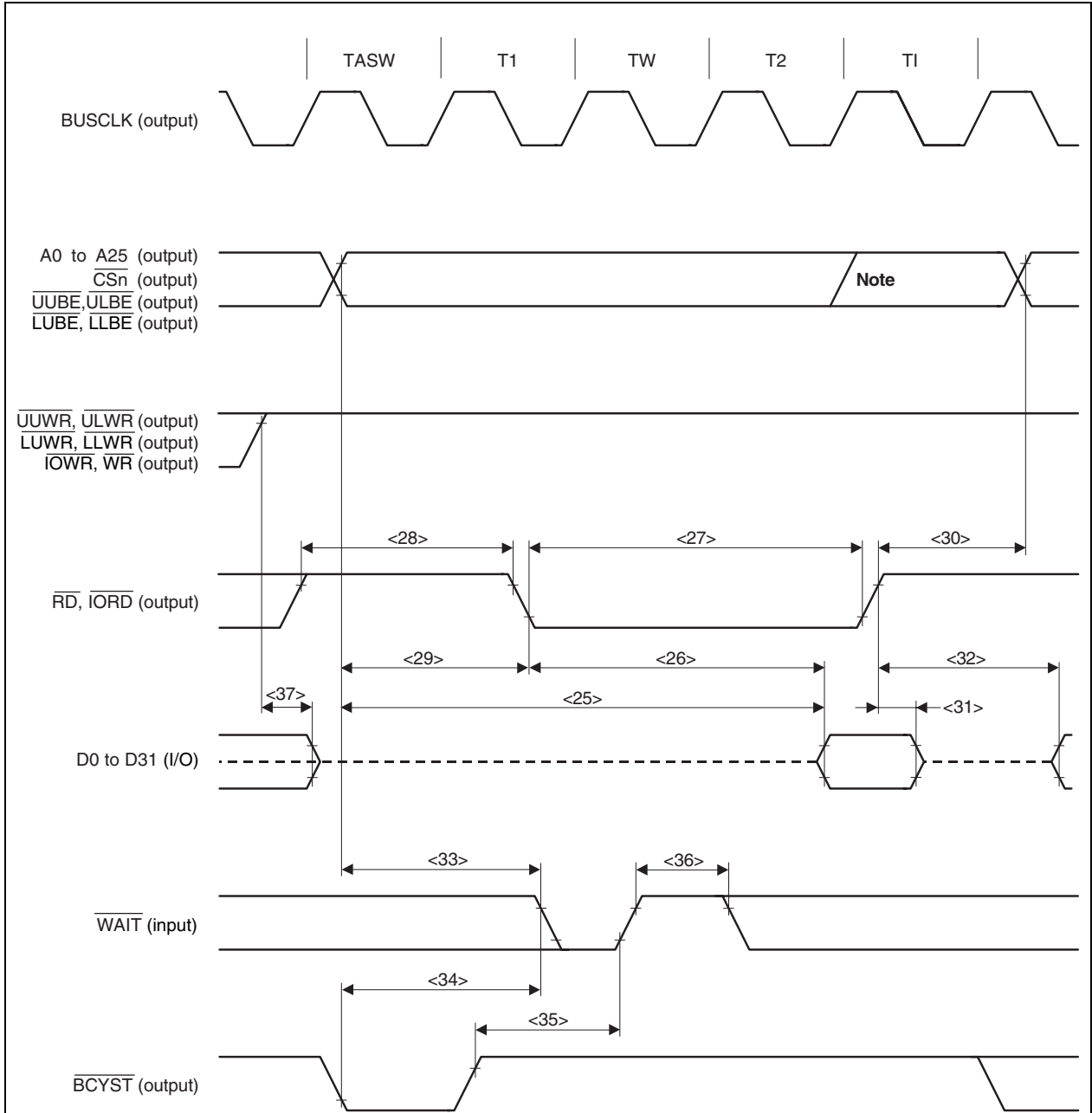
(b) Read timing (SRAM, external ROM, external I/O) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Data input setup time (to address)	<25> t_{SAID}			$(2 + w + w_D + w_{AS}) T - 17$	ns
Data input setup time (to \overline{RD})	<26> t_{SRDID}			$(1.5 + w + w_D) T - 17$	ns
\overline{RD} , \overline{IORD} low-level width	<27> t_{WRDL}		$(1.5 + w + w_D) T - 6$		ns
\overline{RD} , \overline{IORD} high-level width	<28> t_{WRDH}		$(0.5 + w_{AS} + i) T - 6$		ns
Delay time from address, \overline{CS}_n to \overline{RD} , $\overline{IORD}\downarrow$	<29> t_{DARD}		$(0.5 + w_{AS}) T - 7.5$		ns
Delay time from \overline{RD} , $\overline{IORD}\uparrow$ to address	<30> t_{DRDA}		$iT - 2$		ns
Data input hold time (from \overline{RD} , $\overline{IORD}\uparrow$)	<31> t_{HRDID}		0		ns
Delay time from \overline{RD} , $\overline{IORD}\uparrow$ to data output	<32> t_{DRDOD}		$(0.5 + i) T - 6$		ns
\overline{WAIT} setup time (to address)	<33> t_{SAW}	Note 1		$(1 + w_{AS}) T - 17$	ns
\overline{WAIT} setup time (to $\overline{BCYST}\downarrow$)	<34> t_{SBSW}	Note 1		$(1 + w_{AS}) T - 17$	ns
\overline{WAIT} hold time (from $\overline{BCYST}\uparrow$)	<35> t_{HBSW}	Note 1	$(w_{AS} + w_D) T + 2$		ns
\overline{WAIT} high-level width	<36> t_{WWH}	Note 2	$T + 2$		ns
Data output hold time (from \overline{xxWR} , \overline{IOWR} , $\overline{WR}\uparrow$)	<37> t_{HWROD}		$(0.5 + i) T - 5.5$		ns

- Notes**
1. At the first \overline{WAIT} sampling
 2. Time necessary for releasing the wait state

- Remarks**
1. $T = t_{CYK}$
 2. w : Number of waits inserted due to \overline{WAIT}
 3. w_D : Number of waits inserted by $DWC0$ and $DWC1$ registers
 4. Observe at least one of the data input hold times t_{HRDID} and t_{HKID} .
 5. $n = 0$ to 7
 $xx = UU, UL, LU, LL$
 6. i : Number of idle states
 7. w_{AS} : Number of address setup waits inserted by ASC register

(b) Read timing (SRAM, external ROM, external I/O) (2/2)



Note In the case of the \overline{CSn} signal

- Remarks**
1. Timing when the number of waits inserted by the DWC0 or DWC1 register is 0, the number of idle states inserted by the BCC register is 1, and the number of waits inserted by the ASC register is 1.
 2. Broken lines indicate high impedance.
 3. n = 0 to 7

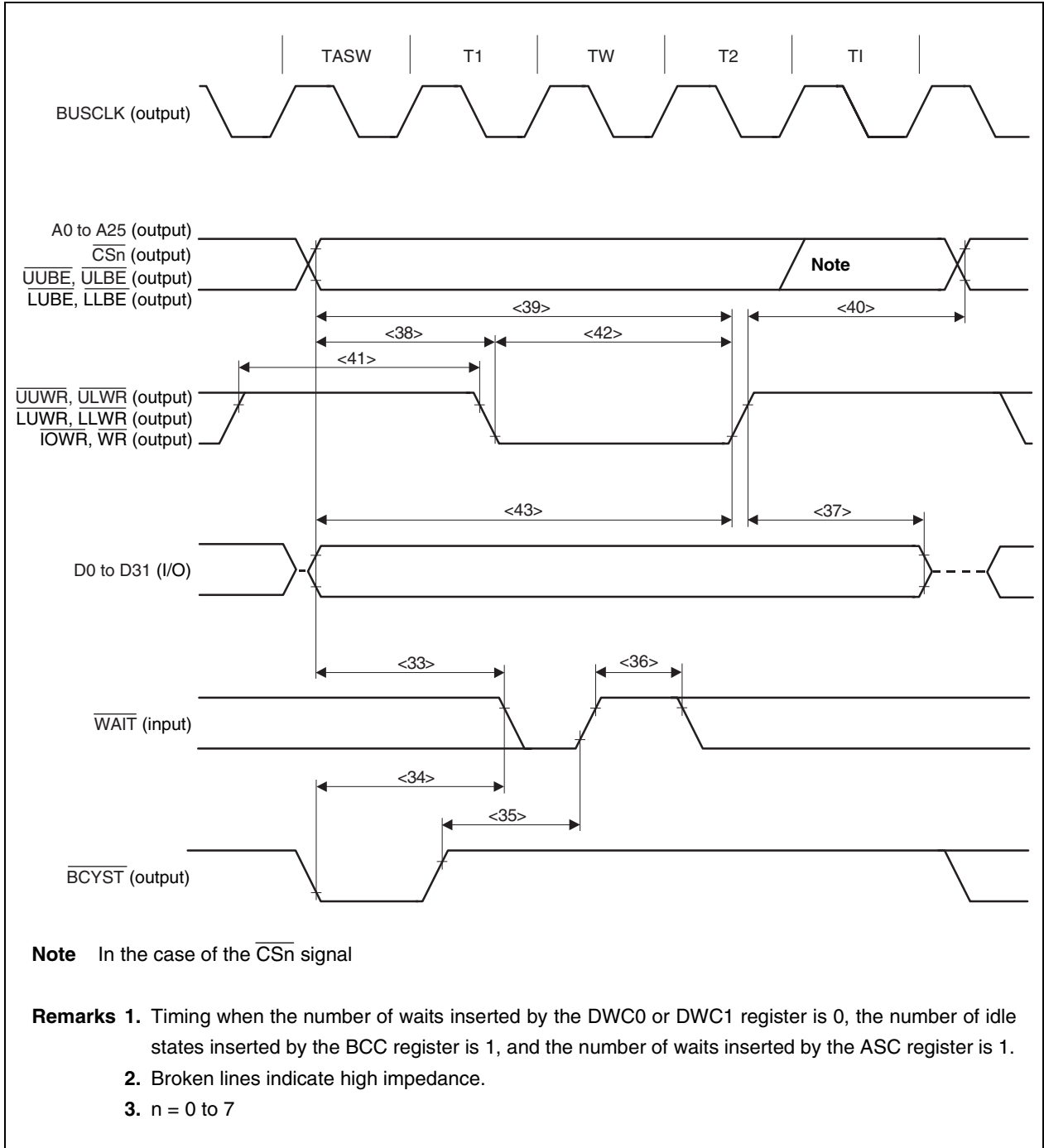
(c) Write timing (SRAM, external ROM, external I/O) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to address)	<33> t_{SAW}	Note 1		$(1 + \text{WAS}) T - 17$	ns
$\overline{\text{WAIT}}$ setup time (to $\overline{\text{BCYST}}\downarrow$)	<34> t_{SBSW}	Note 1		$(1 + \text{WAS}) T - 17$	ns
$\overline{\text{WAIT}}$ hold time (from $\overline{\text{BCYST}}\uparrow$)	<35> t_{HBSW}	Note 1	$(\text{WAS} + \text{WD}) T + 2$		ns
$\overline{\text{WAIT}}$ high-level width	<36> t_{WWH}	Note 2	$T + 2$		ns
Delay time from address, $\overline{\text{CS}}_n$ to $\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}\downarrow$	<38> t_{DAWR}		$(0.5 + \text{WAS}) T - 7$		ns
Address setup time (to $\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}\uparrow$)	<39> t_{SAWR}		$(1.5 + \text{W} + \text{WD} + \text{WAS}) T - 10$		ns
Delay time from $\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}\uparrow$ to address	<40> t_{DWRA}		$(0.5 + i) T - 5$		ns
$\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}$ high-level width	<41> t_{WWRH}		$(0.5 + i + \text{WAS}) T - 5$		ns
$\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}$ low-level width	<42> t_{WWRL}		$(1 + \text{W} + \text{WD}) T - 5$		ns
Data output setup time (to $\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}\uparrow$)	<43> t_{SODWR}		$(1.5 + \text{WAS} + \text{W} + \text{WD}) T - 5$		ns
Data output hold time (from $\overline{\text{xxWR}}$, $\overline{\text{IOWR}}$, $\overline{\text{WR}}\uparrow$)	<37> t_{HWROD}		$(0.5 + i) T - 5.5$		ns

- Notes**
1. At the first $\overline{\text{WAIT}}$ sampling
 2. Time necessary for releasing the wait state

- Remarks**
1. $T = t_{\text{CYK}}$
 2. w : Number of waits inserted due to $\overline{\text{WAIT}}$
 3. wd : Number of waits inserted by $\overline{\text{DWC0}}$ and $\overline{\text{DWC1}}$ registers
 4. $n = 0$ to 7
 $\text{xx} = \text{UU}, \text{UL}, \text{LU}, \text{LL}$
 5. i : Number of idle states
 6. was : Number of address setup waits inserted by $\overline{\text{ASC}}$ register

(c) Write timing (SRAM, external ROM, external I/O) (2/2)



(d) DMA flyby transfer timing (transfer from SRAM to external I/O) (1/2)

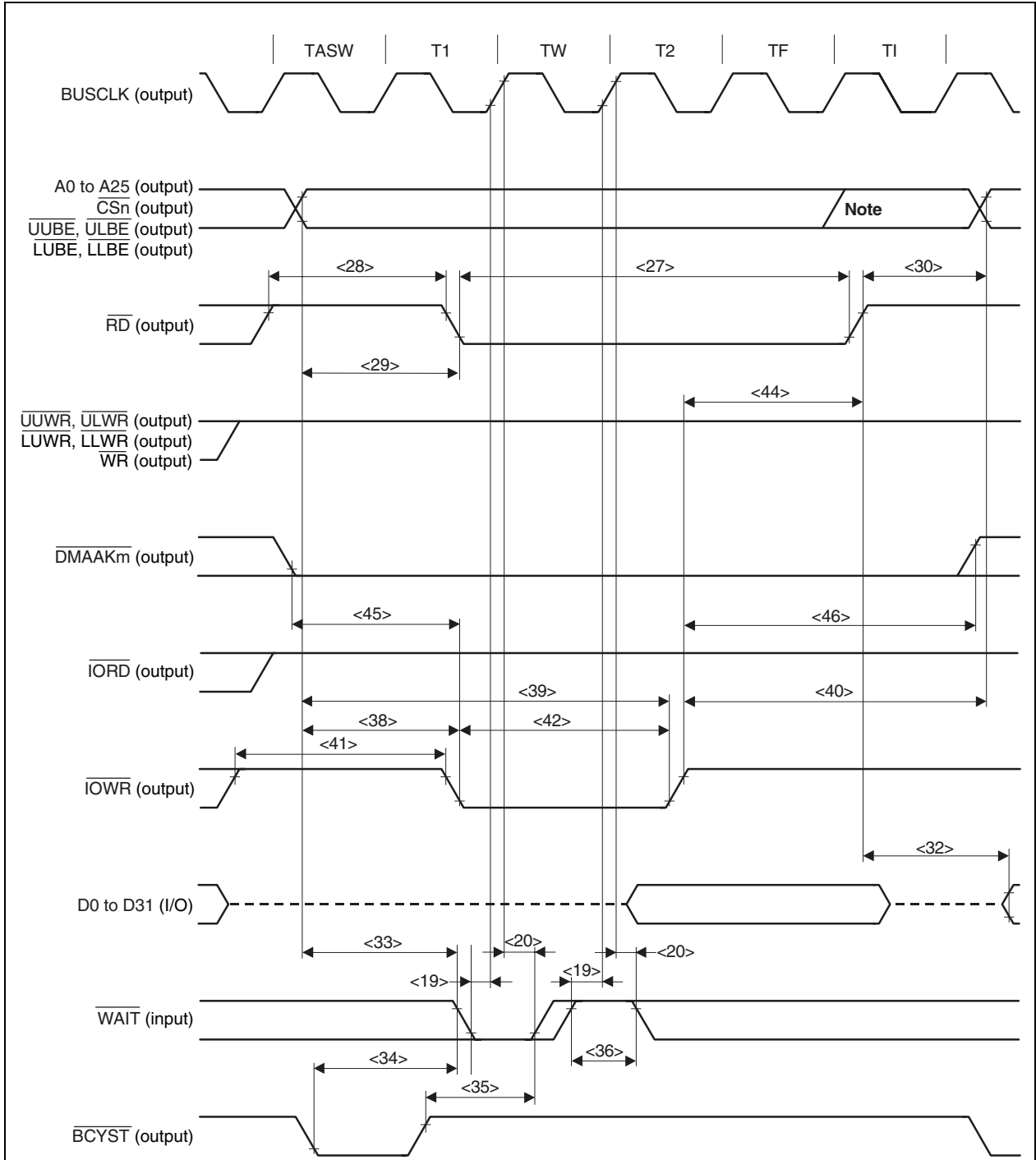
Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to $\text{BUSCLK}\uparrow$)	<19>	t _{SWK}	6		ns
$\overline{\text{WAIT}}$ hold time (from $\text{BUSCLK}\uparrow$)	<20>	t _{HKW}	2		ns
$\overline{\text{RD}}$ low-level width	<27>	t _{WRDL}	$(1.5 + w + w_{FW}) T - 6$		ns
$\overline{\text{RD}}$ high-level width	<28>	t _{WRDH}	$(0.5 + w_{AS} + i) T - 6$		ns
Delay time from address, $\overline{\text{CS}}_n$ to $\overline{\text{RD}}\downarrow$	<29>	t _{DARD}	$(0.5 + w_{AS}) T - 7.5$		ns
Delay time from $\overline{\text{RD}}\uparrow$ to address	<30>	t _{DRDA}	$iT - 2$		ns
Delay time from $\overline{\text{RD}}\uparrow$ to data output	<32>	t _{DRDOD}	$(0.5 + i) T - 6$		ns
$\overline{\text{WAIT}}$ setup time (to address)	<33>	t _{SAW}	Note	$(1 + w_{AS}) T - 17$	ns
$\overline{\text{WAIT}}$ setup time (to $\overline{\text{BCYST}}\downarrow$)	<34>	t _{SBSW}	Note	$(1 + w_{AS}) T - 17$	ns
$\overline{\text{WAIT}}$ hold time (from $\overline{\text{BCYST}}\uparrow$)	<35>	t _{HBSW}	Note	$(w_{AS} + w_{FW}) T + 2$	ns
$\overline{\text{WAIT}}$ high-level width	<36>	t _{WWH}	$T + 2$		ns
Delay time from address to $\overline{\text{IOWR}}\downarrow$	<38>	t _{DAWR}	$(0.5 + w_{AS}) T - 7$		ns
Address setup time (to $\overline{\text{IOWR}}\uparrow$)	<39>	t _{SAWR}	$(1.5 + w + w_{FW} + w_{AS}) T - 10$		ns
Delay time from $\overline{\text{IOWR}}\uparrow$ to address	<40>	t _{DWRA}	$(1.5 + i) T - 5$		ns
$\overline{\text{IOWR}}$ high-level width	<41>	t _{WWRH}	$(0.5 + i + w_{AS}) T - 5$		ns
$\overline{\text{IOWR}}$ low-level width	<42>	t _{WWRL}	$(1 + w + w_{FW}) T - 5$		ns
Delay time from $\overline{\text{IOWR}}\uparrow$ to $\overline{\text{RD}}\uparrow$	<44>	t _{DIWRRD}	$1.5T - 10$		ns
Delay time from $\overline{\text{DMAAK}}_m\downarrow$ to $\overline{\text{IOWR}}\downarrow$	<45>	t _{DDAWR}	$(0.5 + w_{AS}) T - 7.5$		ns
Delay time from $\overline{\text{IOWR}}\uparrow$ to $\overline{\text{DMAAK}}_m\uparrow$	<46>	t _{DWRDA}	$(1.5 + i) T - 10$		ns

Note At the first $\overline{\text{WAIT}}$ sampling

Remarks 1. $T = t_{CYK}$

2. w: Number of waits inserted due to $\overline{\text{WAIT}}$
3. w_{FW}: Number of waits inserted by FWC register
4. n = 0 to 7, m = 0 to 3
5. i: Number of idle states
6. w_{AS}: Number of address setup waits inserted by ASC register

(d) DMA flyby transfer timing (transfer from SRAM to external I/O) (2/2)



Note In the case of the CSn signal

- Remarks**
1. Timing when the number of waits inserted by the FWC register is 0, the number of idle states inserted by the FIC register is 1, and the number of waits inserted by the ASC register is 1.
 2. Broken lines indicate high impedance.
 3. n = 0 to 7, m = 0 to 3

(e) DMA flyby transfer timing (transfer from external I/O to SRAM) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to $\text{BUSCLK}\uparrow$)	<19> t_{SWK}		6		ns
$\overline{\text{WAIT}}$ hold time (from $\text{BUSCLK}\uparrow$)	<20> t_{HKW}		2		ns
$\overline{\text{IORD}}$ low-level width	<27> t_{WRDL}		$(2 + w + w_{\text{FW}}) T - 6$		ns
$\overline{\text{IORD}}$ high-level width	<28> t_{WRDH}		$(1 + w_{\text{AS}} + i) T - 6$		ns
Delay time from address, $\overline{\text{CSn}}$ to $\overline{\text{IORD}}\downarrow$	<29> t_{DARD}		$(0.5 + w_{\text{AS}}) T - 7.5$		ns
Delay time from $\overline{\text{IORD}}\uparrow$ to address	<30> t_{DRDA}		$iT - 2$		ns
Delay time from $\overline{\text{IORD}}\uparrow$ to data output	<32> t_{DRDOD}		$(0.5 + i) T - 6$		ns
$\overline{\text{WAIT}}$ setup time (to address)	<33> t_{SAW}	Note		$(1 + w_{\text{AS}}) T - 17$	ns
$\overline{\text{WAIT}}$ setup time (to $\overline{\text{BCYST}}\downarrow$)	<34> t_{SBSW}	Note		$(1 + w_{\text{AS}}) T - 17$	ns
$\overline{\text{WAIT}}$ hold time (from $\overline{\text{BCYST}}\uparrow$)	<35> t_{HBSW}	Note	$(w_{\text{AS}} + w_{\text{FW}}) T + 2$		ns
$\overline{\text{WAIT}}$ high-level width	<36> t_{WWH}		$T + 2$		ns
Delay time from address to $\overline{\text{xxWR}}$, $\overline{\text{WR}}\downarrow$	<38> t_{DAWR}		$(0.5 + w_{\text{AS}}) T - 7$		ns
Address setup time (to $\overline{\text{xxWR}}$, $\overline{\text{WR}}\uparrow$)	<39> t_{SAWR}		$(1.5 + w + w_{\text{FW}} + w_{\text{AS}}) T - 10$		ns
Delay time from $\overline{\text{xxWR}}$, $\overline{\text{WR}}\uparrow$ to address	<40> t_{DWRA}		$(0.5 + i) T - 5$		ns
$\overline{\text{xxWR}}$, $\overline{\text{WR}}$ high-level width	<41> t_{WWRH}		$(0.5 + i + w_{\text{AS}}) T - 5$		ns
$\overline{\text{xxWR}}$, $\overline{\text{WR}}$ low-level width	<42> t_{WWRL}		$(1 + w + w_{\text{FW}}) T - 5$		ns
Delay time from $\overline{\text{xxWR}}$, $\overline{\text{WR}}\uparrow$ to $\overline{\text{IORD}}\uparrow$	<47> t_{DWRIRD}		$T - 10$		ns
Delay time from $\overline{\text{DMAAKm}}\downarrow$ to $\overline{\text{IORD}}\downarrow$	<48> t_{DDARD}		$(0.5 + w_{\text{AS}}) T - 7.5$		ns
Delay time from $\overline{\text{IORD}}\uparrow$ to $\overline{\text{DMAAKm}}\uparrow$	<49> t_{DRDDA}		$(0.5 + i) T - 7.5$		ns

Note At the first $\overline{\text{WAIT}}$ sampling

Remarks 1. $T = t_{\text{CYK}}$

2. w : Number of waits inserted due to $\overline{\text{WAIT}}$

3. w_{FW} : Number of waits inserted by FWC register

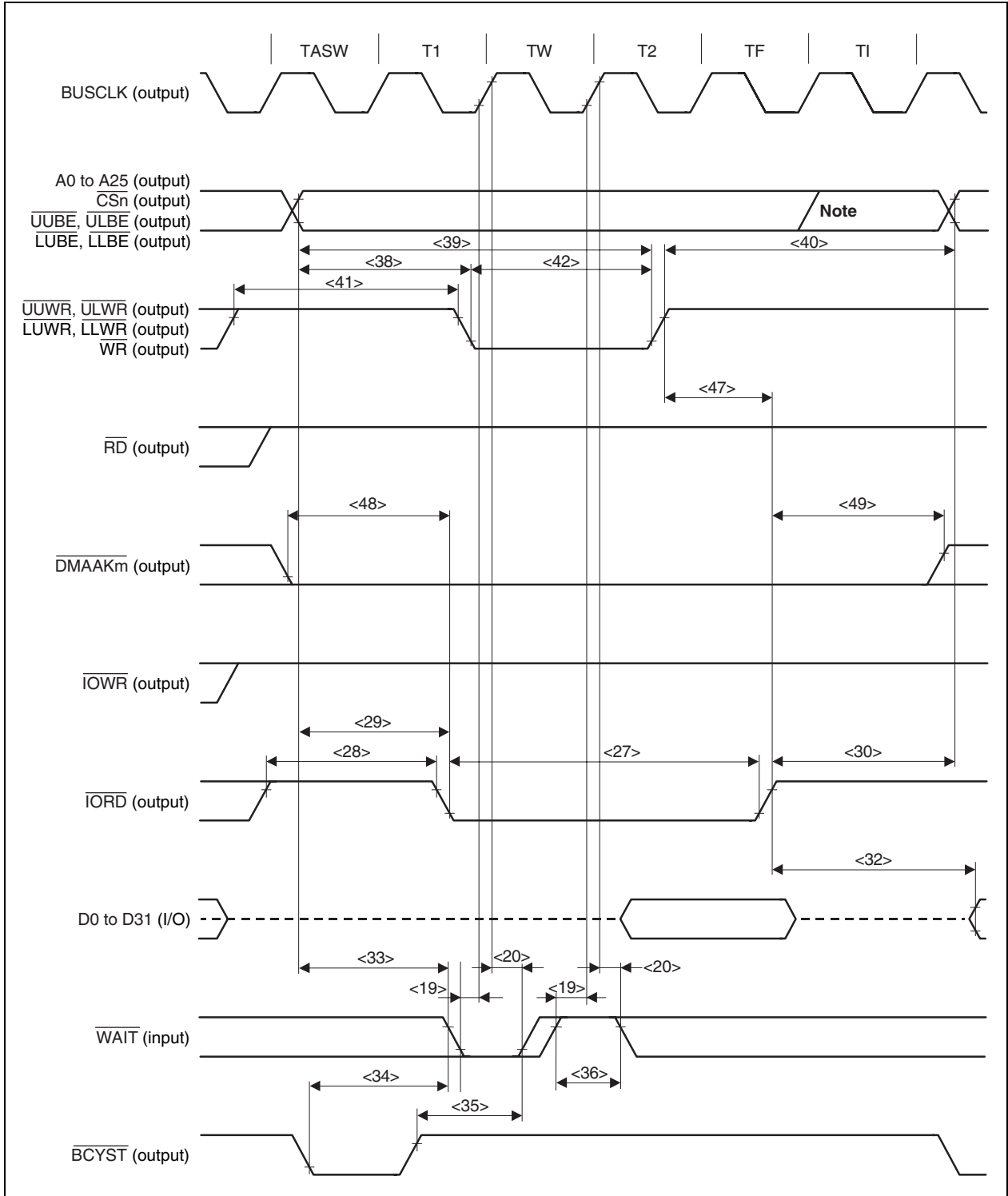
4. $n = 0$ to 7, $m = 0$ to 3

$\text{xx} = \text{UU}, \text{UL}, \text{LU}, \text{LL}$

5. i : Number of idle states inserted when a write cycle is inserted after a read cycle

6. w_{AS} : Number of address setup waits inserted by ASC register

(e) DMA flyby transfer timing (transfer from external I/O to SRAM) (2/2)



Note In the case of the \overline{CSn} signal

Remarks 1. Timing when the number of waits inserted by the FWC register is 0, the number of idle states inserted by the FIC register is 1, and the number of waits inserted by the ASC register is 1.

2. Broken lines indicate high impedance.

3. n = 0 to 7, m = 0 to 3

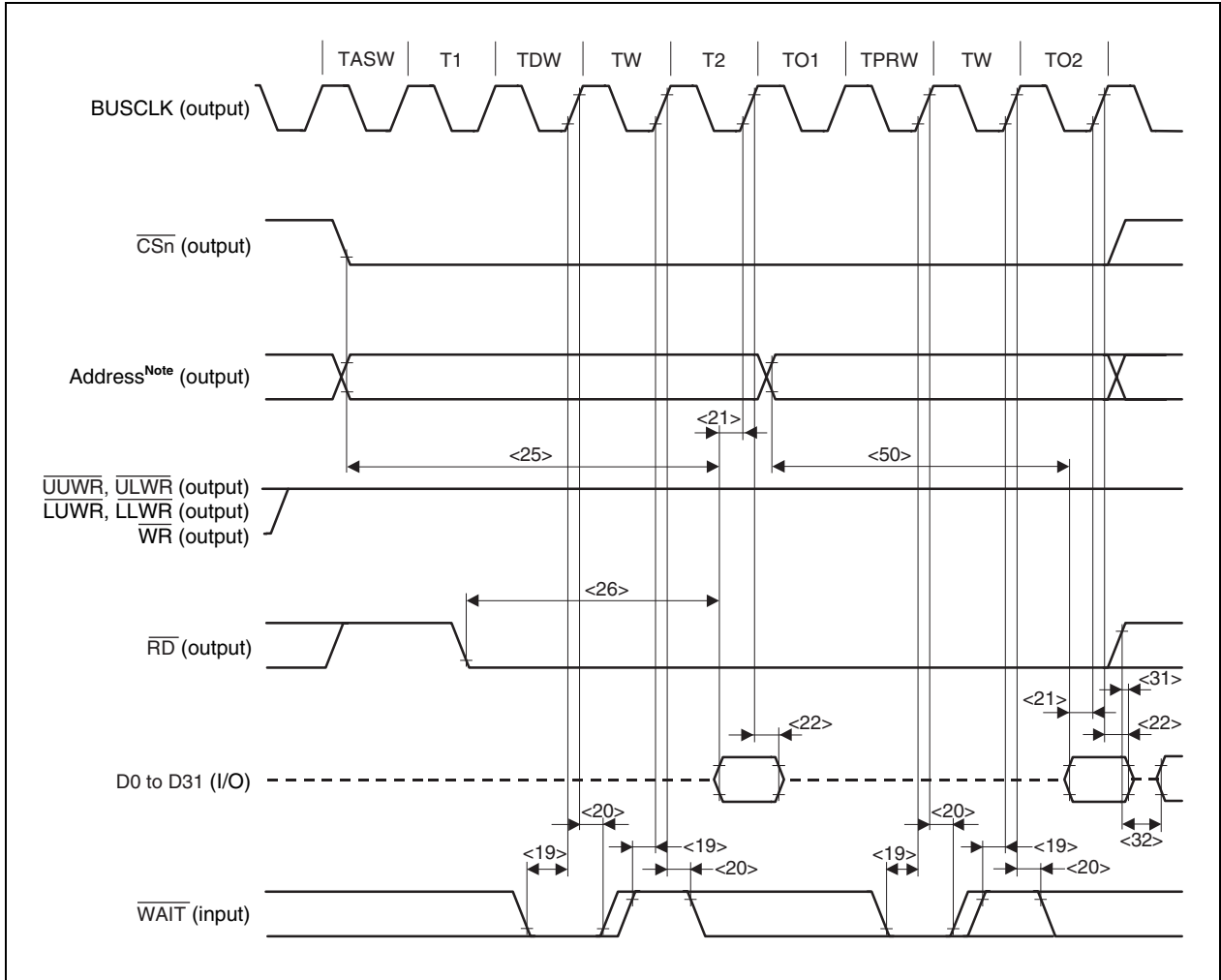
(5) Page ROM access timing (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{WAIT}}$ setup time (to $\text{BUSCLK}\uparrow$)	<19> t_{SWK}		6		ns
$\overline{\text{WAIT}}$ hold time (from $\text{BUSCLK}\uparrow$)	<20> t_{HKW}		2		ns
Data input setup time (to $\text{BUSCLK}\uparrow$)	<21> t_{SKID}		6		ns
Data input hold time (from $\text{BUSCLK}\uparrow$)	<22> t_{HKID}		2		ns
Off-page data input setup time (to address)	<25> t_{SAID}			$(2 + w + w_D + w_{AS}) T - 17$	ns
Off-page data input setup time (to $\overline{\text{RD}}$)	<26> t_{SRDID}			$(1.5 + w + w_D) T - 17$	ns
Data input hold time (from $\overline{\text{RD}}\uparrow$)	<31> t_{HRDID}		2		ns
Delay time from $\overline{\text{RD}}\uparrow$ to data output	<32> t_{DRDOD}		$(0.5 + i) T - 6$		ns
On-page data input setup time (to address)	<50> t_{SOAID}			$(2 + w + w_{PR}) T - 17$	ns

Remarks 1. $T = t_{\text{CYK}}$

2. w : Number of waits inserted due to $\overline{\text{WAIT}}$
3. w_D : Number of waits inserted by DWC0 and DWC1 registers
4. w_{PR} : Number of waits inserted by PRC register
5. i : Number of idle states inserted when a write cycle is inserted after a read cycle
6. w_{AS} : Number of address setup waits inserted by ASC register
7. Observe at least one of the data input hold times t_{HRDID} and t_{HKID} .

(5) Page ROM access timing (2/2)



Note On-page addresses and off-page addresses are shown below.

PRC register				On-page address	Off-page address
MA6	MA5	MA4	MA3		
0	0	0	0	A0 to A2	A3 to A25
0	0	0	1	A0 to A3	A4 to A25
0	0	1	1	A0 to A4	A5 to A25
0	1	1	1	A0 to A5	A6 to A25
1	1	1	1	A0 to A6	A7 to A25

Remarks 1. Timing in the following case.

Number of waits inserted by DWC0 or DWC1 register (TDW): 1

Number of waits inserted by PRC register (TPRW): 1

Number of waits inserted by ASC register (TASW): 1

2. Broken lines indicate high impedance.

3. n = 0 to 7

(6) SDRAM access timing

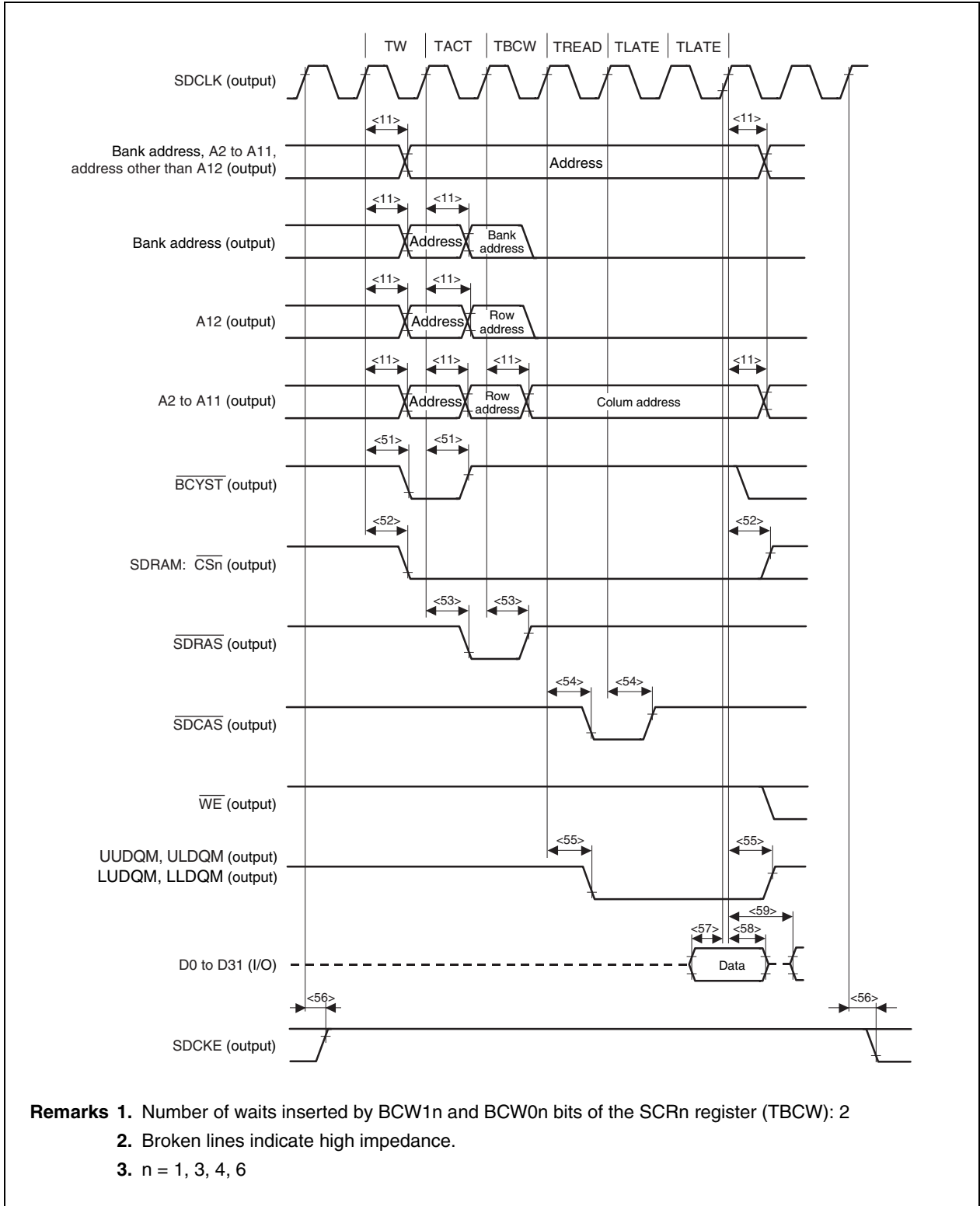
(a) Read timing (SDRAM access) (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Address delay time (from BUSCLK↑)	<11> t_{DKA}		2	11	ns
\overline{BCYST} delay time (from BUSCLK↑)	<51> t_{DKBC}		2	11	ns
\overline{CSn} delay time (from BUSCLK↑)	<52> t_{DKCS}		2	11	ns
\overline{SDRAS} delay time (from BUSCLK↑)	<53> t_{DKRAS}		2	11	ns
\overline{SDCAS} delay time (from BUSCLK↑)	<54> t_{DKCAS}		2	11	ns
xxDQM delay time (from BUSCLK↑)	<55> t_{DKDQM}		2	11	ns
SDCKE delay time (from BUSCLK↑)	<56> t_{DKCKE}		2	11	ns
Data input setup time (SDRAM read, from BUSCLK↑)	<57> t_{SDRMK}		6		ns
Data input hold time (SDRAM read, from BUSCLK↑)	<58> t_{HKDRM}		2		ns
Delay time from BUSCLK↑ to data output	<59> t_{DSDOD}		(0.5 + i) T		ns

Caution If an SRAM (external I/O) cycle that uses the \overline{xxWR} signal is generated immediately after a read cycle to SDRAM, an SRAM (external I/O) writing error may occur. In this case, set the BCC register to insert an idle state in the SDRAM space or execute a countermeasure using external circuits. However, no writing error occurs in a synchronization design in which the \overline{xxWR} signal is sampled using BUSCLK.

- Remarks**
1. $T = t_{CYK}$
 2. i: Number of idle states
 3. $n = 1, 3, 4, 6$
xx = UU, UL, LU, LL

(a) Read timing (SDRAM access) (2/2)



(b) Write timing (SDRAM access) (1/2)

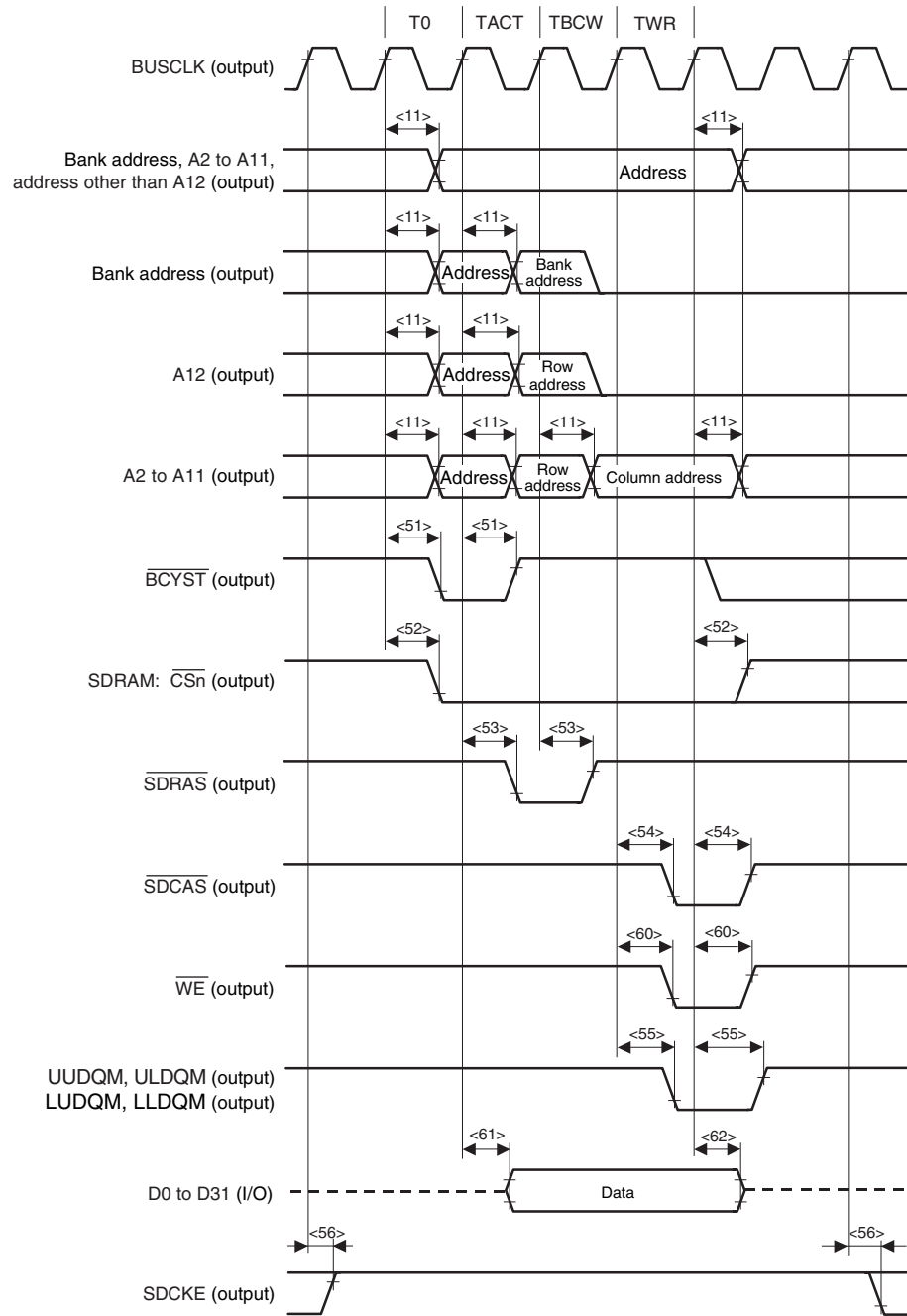
Parameter	Symbol	Conditions	MIN.	MAX.	Unit
Address delay time (from BUSCLK↑)	<11> t _{DKA}		2	11	ns
$\overline{\text{BCYST}}$ delay time (from BUSCLK↑)	<51> t _{DKBC}		2	11	ns
$\overline{\text{CSn}}$ delay time (from BUSCLK↑)	<52> t _{DKCS}		2	11	ns
$\overline{\text{SDRAS}}$ delay time (from BUSCLK↑)	<53> t _{DKRAS}		2	11	ns
$\overline{\text{SDCAS}}$ delay time (from BUSCLK↑)	<54> t _{DKCAS}		2	11	ns
xxDQM delay time (from BUSCLK↑)	<55> t _{DKDQM}		2	11	ns
SDCKE delay time (from BUSCLK↑)	<56> t _{DKCKE}		2	11	ns
$\overline{\text{WE}}$ delay time (from BUSCLK↑)	<60> t _{DKWE}		2	11	ns
Data output delay time (from BUSCLK↑)	<61> t _{DKDT}		2	11	ns
Data float delay time (from BUSCLK↑)	<62> t _{HZKDT}		2	11	ns

Caution If an SRAM (external I/O) cycle that uses the $\overline{\text{xxWR}}$ signal is generated immediately after a read cycle to SDRAM, an SRAM (external I/O) writing error may occur. In this case, execute a countermeasure using external circuits.

However, no writing error occurs in a synchronization design in which the $\overline{\text{xxWR}}$ signal is sampled using BUSCLK.

Remark n = 1, 3, 4, 6
xx = UU, UL, LU, LL

(b) Write timing (SDRAM access) (2/2)



- Remarks**
1. Number of waits inserted by BCW1n and BCW0n bits of SCRn register (TBCW): 2
 2. Broken lines indicate high impedance.
 3. n = 1, 3, 4, 6

(7) DMAC timing

(a) Level mode (1/3)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{DMARQn}}$ setup time (to $\text{BUSCLK}\uparrow$)	<63> t_{SDRK}	2-cycle/flyby transfer	8		ns
$\overline{\text{DMARQn}}$ hold time (from $\overline{\text{DMAAKn}}\downarrow$)	<64> t_{HKDR}	2-cycle transfer	0	$4T_{\text{CPU}} - 20$ ^{Note 1}	ns
		Flyby transfer	0	$6T_{\text{CPU}} - 20$ ^{Note 2}	ns
			0	$2T_{\text{BUS}} + 2T_{\text{CPU}} - 20$	ns
$\overline{\text{DMAAKn}}$ output delay time (from $\text{BUSCLK}\uparrow$)	<65> t_{DKDA}	2-cycle transfer	0	$nT_{\text{CPU}} + 13$ ^{Note 3}	ns
		Flyby transfer	0	13	ns
$\overline{\text{DMAAKn}}$ output hold time (from $\text{BUSCLK}\uparrow$)	<66> t_{HKDA}	2-cycle transfer	0	$nT_{\text{CPU}} + 13$ ^{Note 3}	ns
		Flyby transfer	0	13	ns

Notes 1. Second DMA transfer request disable timing in single transfer.

The accesses is as follows.

Transfer Source	Transfer Destination	Speculative Read Function
Internal data RAM	External memory/ internal instruction RAM	Provided/none
External memory	Internal data RAM	Provided/none However, when the speculative read function is not provided, the following conditions apply. <ul style="list-style-type: none"> • BUSCLK = Internal system clock (f_{CLK}) • External memory access wait setting = 0

2. Second DMA transfer request disable timing in single transfer.

The access is other than that shown in the table in **Note 1**.

3. n is as follows.

CKM1	CKM0	BUSCLK Division Ratio with Respect to Internal System Clock (f_{CLK})	n
0	0	$f_{\text{CLK}}/1$	1
0	1	$f_{\text{CLK}}/2$	2
1	0	$f_{\text{CLK}}/3$	3
1	1	$f_{\text{CLK}}/4$	4

Remarks 1. $n = 0$ to 3

2. $T_{\text{BUS}} = 1 \times \text{BUSCLK cycle}$

3. $T_{\text{CPU}} = 1 \times \text{internal system clock cycle}$

(a) Level mode (2/3)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit	
$\overline{\text{DMAAKn}}$ low-level width	<67>	2-cycle transfer	Note 1	$2T_{\text{CPU}} - 8^{\text{Note 2}}$		ns
			Note 3	$6T_{\text{CPU}} - 8^{\text{Note 2}}$		ns
		Flyby transfer		$2T_{\text{BUS}} + W_{\text{AS}} + W_{\text{FW}} + W_{\text{IC}} - 8$		ns
$\overline{\text{TCn}}$ output delay time (from $\text{BUSCLK}\uparrow$)	<68>	t_{DKTC}	2-cycle/flyby transfer	2	13	ns
$\overline{\text{TCn}}$ output hold time (from $\text{BUSCLK}\uparrow$)	<69>	t_{HKTC}	2-cycle/flyby transfer	2	13	ns

- Notes**
1. Normal operation ($\overline{\text{DMAAKn}}$ output width = memory controller output)
 2. The access is as follows.

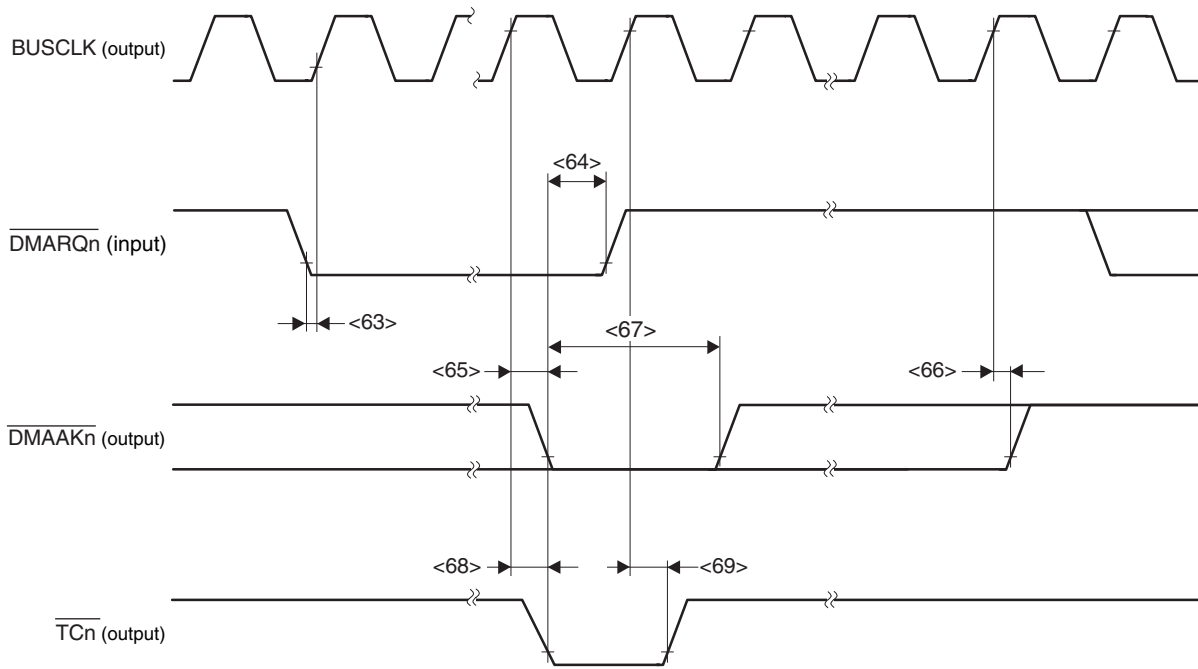
Transfer Source	Transfer Destination	Speculative Read Function
Internal data RAM	External memory/ internal instruction RAM	Provided/none
External memory	Internal data RAM	Provided/none However, when the speculative read function is not provided, the following conditions apply. <ul style="list-style-type: none"> • BUSCLK = Internal system clock (f_{CLK}) • External memory access wait setting = 0

3. When selecting a mode in which $\overline{\text{DMAAKn}}$ output width = memory controller output + $4T_{\text{CPU}}$

Remarks 1. $n = 0$ to 3

2. $T_{\text{BUS}} = 1 \times \text{BUSCLK cycle}$
3. $T_{\text{CPU}} = 1 \times \text{internal system clock cycle}$
4. W_{AS} = Number of address setup waits inserted by ASC register
 W_{FW} = Number of data waits inserted by FWC register
 W_{IC} = Number of idle states inserted by FIC register

(a) Level mode (3/3)



Remarks 1. The minimum $\overline{\text{DMAAKn}}$ inactive time is as follows.

Transfer mode	Transfer source	Transfer destination	Inactive time
Single transfer, single-step transfer	External memory/ external I/O on-chip peripheral I/O	External memory/ external I/O on-chip peripheral I/O/ internal instruction RAM	$4T_{\text{CPU}}$
	Internal data RAM	Internal data RAM	$5T_{\text{CPU}}$
	Internal data RAM	External memory/ external I/O on-chip peripheral I/O/ internal instruction RAM	$9T_{\text{CPU}}$
	External memory/ external I/O on-chip peripheral I/O	Internal data RAM	$9T_{\text{CPU}}$
Block transfer	—	—	T_{CPU}

2. $n = 0$ to 3

(b) Mask mode (1/3)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{DMARQn}}$ setup time (to $\text{BUSCLK}\uparrow$)	<63> t_{SDRK}	2-cycle/flyby transfer	8		ns
$\overline{\text{DMARQn}}$ hold time 1 (from $\overline{\text{DMAAKn}}\downarrow$)	<70> t_{HKDR1}	2-cycle/flyby transfer	To $\overline{\text{DMAAKn}}\downarrow$		ns
$\overline{\text{DMARQn}}$ hold time 2 (from $\text{BUSCLK}\downarrow$) ^{Note 2} ($\overline{\text{DMAAKn}}$ = "H" sample)	<71> t_{HKDR2}	2-cycle transfer	0	$3T_{\text{BUS}} - 8$ ^{Note 1}	ns
		Flyby transfer	0	$4T_{\text{BUS}} - 8$ ^{Note 3}	ns
$\overline{\text{DMAAKn}}$ output delay time (from $\text{BUSCLK}\uparrow$)	<65> t_{DKDA}	2-cycle transfer	0	$nT_{\text{CPU}} + 13$ ^{Note 4}	ns
		Flyby transfer	0	13	ns
$\overline{\text{DMAAKn}}$ output hold time (from $\text{BUSCLK}\uparrow$)	<66> t_{HKDA}	2-cycle transfer	0	$nT_{\text{CPU}} + 13$ ^{Note 4}	ns
		Flyby transfer	0	13	ns

Notes 1. The second transfer request disable timing in single transfer.

To $\text{BUSCLK}\uparrow$: Since the $\overline{\text{DMAAKn}}$ is output asynchronously with BUSCLK , in accordance with the $\overline{\text{DMAAKn}}$ rising "H" sample prescription, if $\overline{\text{DMAAKn}}$ = "H" is output at the timing in which it cannot be sampled at the BUSCLK (setup time < 8 ns), $+1T_{\text{BUS}}$ is added.

2. Time to $\overline{\text{DMAAKn}}$ high level from $\text{BUSCLK}\uparrow$ after $\overline{\text{DMAAKn}}$ rises
3. The second DMA transfer request disable timing in single transfer.
4. n is as follows.

CKM1	CKM0	BUSCLK Division Ratio to Internal System Clock (f_{CLK})	n
0	0	$f_{\text{CLK}}/1$	1
0	1	$f_{\text{CLK}}/2$	2
1	0	$f_{\text{CLK}}/3$	3
1	1	$f_{\text{CLK}}/4$	4

Remarks 1. n = 0 to 3

2. $T_{\text{BUS}} = 1 \times \text{BUSCLK cycle}$
3. $T_{\text{CPU}} = 1 \times \text{internal system clock cycle}$

(b) Mask mode (2/3)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit	
$\overline{\text{DMAAKn}}$ low-level width	<67>	2-cycle transfer	Note 1	$2T_{\text{CPU}} - 8$ ^{Note 2}		ns
			Note 3	$6T_{\text{CPU}} - 8$ ^{Note 2}		ns
		Flyby transfer		$2T_{\text{BUS}} + W_{\text{AS}} + W_{\text{FW}} + W_{\text{IC}} - 8$		ns
$\overline{\text{TCn}}$ output delay time (from $\text{BUSCLK}\uparrow$)	<68>	t_{DKTC}	2-cycle/flyby transfer	2	13	ns
$\overline{\text{TCn}}$ output hold time (from $\text{BUSCLK}\uparrow$)	<69>	t_{HKTC}	2-cycle/flyby transfer	2	13	ns

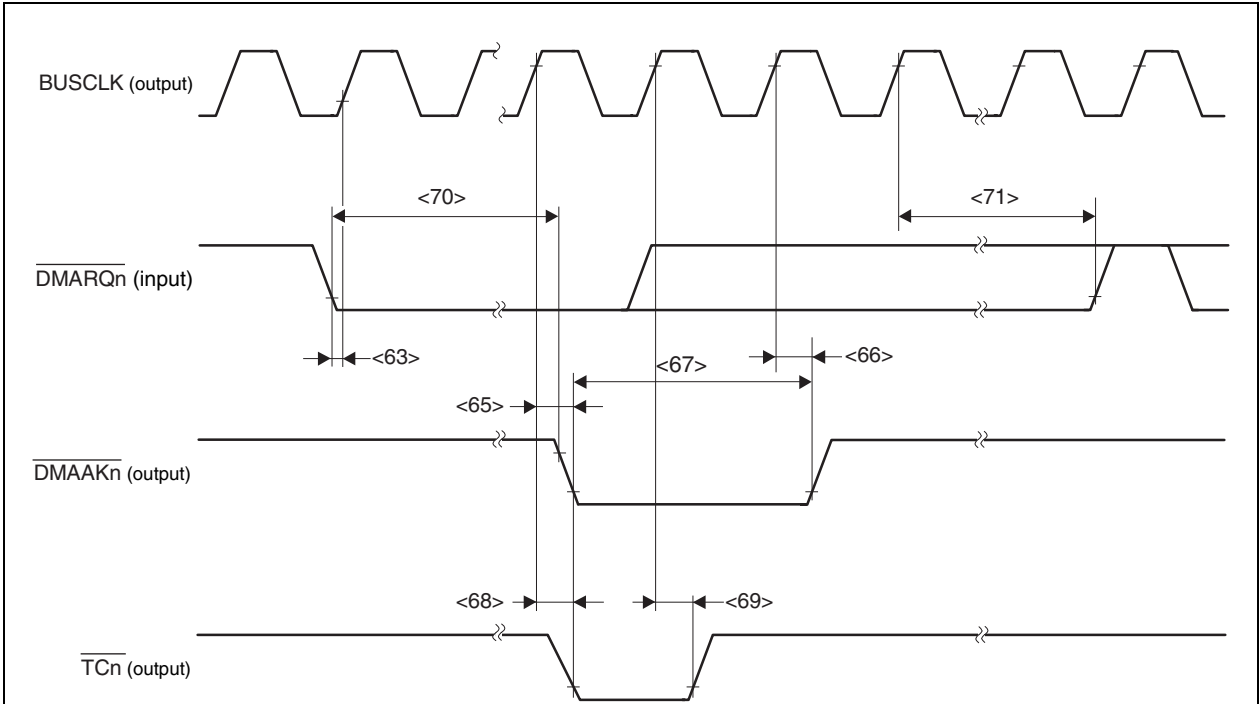
- Notes**
1. Normal operation ($\overline{\text{DMAAKn}}$ output width = memory controller output)
 2. The access is as follows.

Transfer Source	Transfer Destination	Speculative Read Function
Internal data RAM	External memory/ internal instruction RAM	Provided/none
External memory	Internal data RAM	Provided/none However, when the speculative read function is not provided, the following conditions apply. <ul style="list-style-type: none"> • BUSCLK = Internal system clock (f_{CLK}) • External memory access wait setting = 0

3. When selecting a mode in which $\overline{\text{DMAAKn}}$ output width = memory controller output + $4T_{\text{CPU}}$

- Remarks**
1. $n = 0$ to 3
 2. $T_{\text{BUS}} = 1 \times \text{BUSCLK cycle}$
 3. $T_{\text{CPU}} = 1 \times \text{internal system clock cycle}$
 4. w_{AS} = Number of address setup waits inserted by ASC register
 w_{FW} = Number of data waits inserted by FWC register
 w_{IC} = Number of idle states inserted by FIC register

(b) Masks mode (3/3)



Remarks 1. The minimum $\overline{\text{DMAAKn}}$ inactive time is as follows.

Transfer mode	Transfer source	Transfer destination	Inactive time
Single transfer, Single-step transfer	External memory/ external I/O on-chip peripheral I/O	External memory/ external I/O on-chip peripheral I/O/ internal instruction RAM	$4T_{\text{CPU}}$
	Internal data RAM	Internal data RAM	$5T_{\text{CPU}}$
	Internal data RAM	External memory/ external I/O on-chip peripheral I/O/ internal instruction RAM	$9T_{\text{CPU}}$
	External memory/ external I/O on-chip peripheral I/O	Internal data RAM	$9T_{\text{CPU}}$
Block transfer	—	—	T_{CPU}

2. $n = 0$ to 3

(c) Edge mode (1/3)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{DMARQn}}$ hold time	<64> t_{HKDR}	2-cycle/flyby transfer	$2T_{\text{CPU}}$		ns
$\overline{\text{DMARQn}}$ high-level time 1 (from $\overline{\text{DMARQn}}\uparrow$)	<72> t_{WDRH1}	2-cycle/flyby transfer	$2T_{\text{CPU}}$		ns
$\overline{\text{DMARQn}}$ high-level time 2 (from $\overline{\text{DMAAKn}}\uparrow$)	<73> t_{WDRH2}	2-cycle/flyby transfer	0		ns
$\overline{\text{DMAAKn}}$ output delay time (from $\overline{\text{BUSCLK}}\uparrow$)	<65> t_{DKDA}	2-cycle transfer	0	$nT_{\text{CPU}} + 13^{\text{Note}}$	ns
		Flyby transfer	0	13	ns
$\overline{\text{DMAAKn}}$ output hold time (from $\overline{\text{BUSCLK}}\uparrow$)	<66> t_{HKDA}	2-cycle transfer	0	$nT_{\text{CPU}} + 13^{\text{Note}}$	ns
		Flyby transfer	0	13	ns

Note n is as follows.

CKM1	CKM0	BUSCLK Ratio to Internal System Clock (f_{CLK})	n
0	0	$f_{\text{CLK}}/1$	1
0	1	$f_{\text{CLK}}/2$	2
1	0	$f_{\text{CLK}}/3$	3
1	1	$f_{\text{CLK}}/4$	4

Remarks 1. n = 0 to 3

2. $T_{\text{CPU}} = 1 \times$ internal system clock cycle

(c) Edge mode (2/3)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit	
$\overline{\text{DMAAKn}}$ low-level width	<67>	2-cycle transfer	Note 1	$2T_{\text{CPU}} - 8^{\text{Note 2}}$		ns
			Note 3	$6T_{\text{CPU}} - 8^{\text{Note 2}}$		ns
		Flyby transfer		$2T_{\text{BUS}} + W_{\text{AS}} + W_{\text{FW}} + W_{\text{IC}} - 8$		ns
$\overline{\text{TCn}}$ output delay time (from $\text{BUSCLK}\uparrow$)	<68>	t_{DKTC}	2-cycle/flyby transfer	2	13	ns
$\overline{\text{TCn}}$ output hold time (from $\text{BUSCLK}\uparrow$)	<69>	t_{HKTC}	2-cycle/flyby transfer	2	13	ns

- Notes**
1. Time to $\overline{\text{DMAAKn}}$ high level from $\text{BUSCLK}\uparrow$ after $\overline{\text{DMAAKn}}$ rises
 2. The access is as follows.

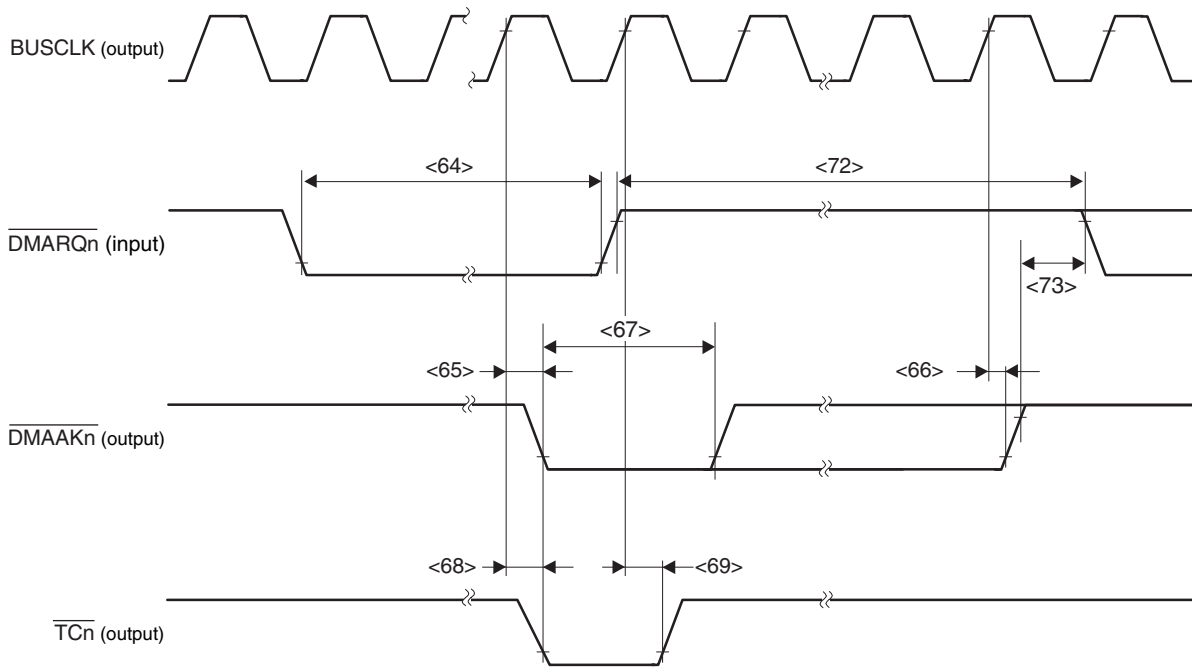
Transfer Source	Transfer Destination	Speculative Read Function
Internal data RAM	External memory/ internal instruction RAM	Provided/none
External memory	Internal data RAM	Provided/none However, when the speculative read function is not provided, the following conditions apply. <ul style="list-style-type: none"> • BUSCLK = Internal system clock (f_{CLK}) • External memory access wait setting = 0

3. Normal operation ($\overline{\text{DMAAKn}}$ output width = memory controller output)

Remarks 1. $n = 0$ to 3

2. $T_{\text{BUS}} = 1 \times \text{BUSCLK cycle}$
3. $T_{\text{CPU}} = 1 \times \text{internal system clock cycle}$
4. W_{AS} = Number of address setup waits inserted by ASC register
 W_{FW} = Number of data waits inserted by FWC register
 W_{IC} = Number of idle states inserted by FIC register

(c) Edge mode (3/3)



Remarks 1. The minimum $\overline{\text{DMAAKn}}$ inactive time is as follows.

Transfer mode	Transfer source	Transfer destination	Inactive time
Single transfer, Single-step transfer	External memory/ external I/O on-chip peripheral I/O	External memory/ external I/O on-chip peripheral I/O/ internal instruction RAM	$4T_{\text{CPU}}$
	Internal data RAM	Internal data RAM	$5T_{\text{CPU}}$
	Internal data RAM	External memory/ external I/O on-chip peripheral I/O/ internal instruction RAM	$9T_{\text{CPU}}$
	External memory/ external I/O on-chip peripheral I/O	Internal data RAM	$9T_{\text{CPU}}$
Block transfer	—	—	T_{CPU}

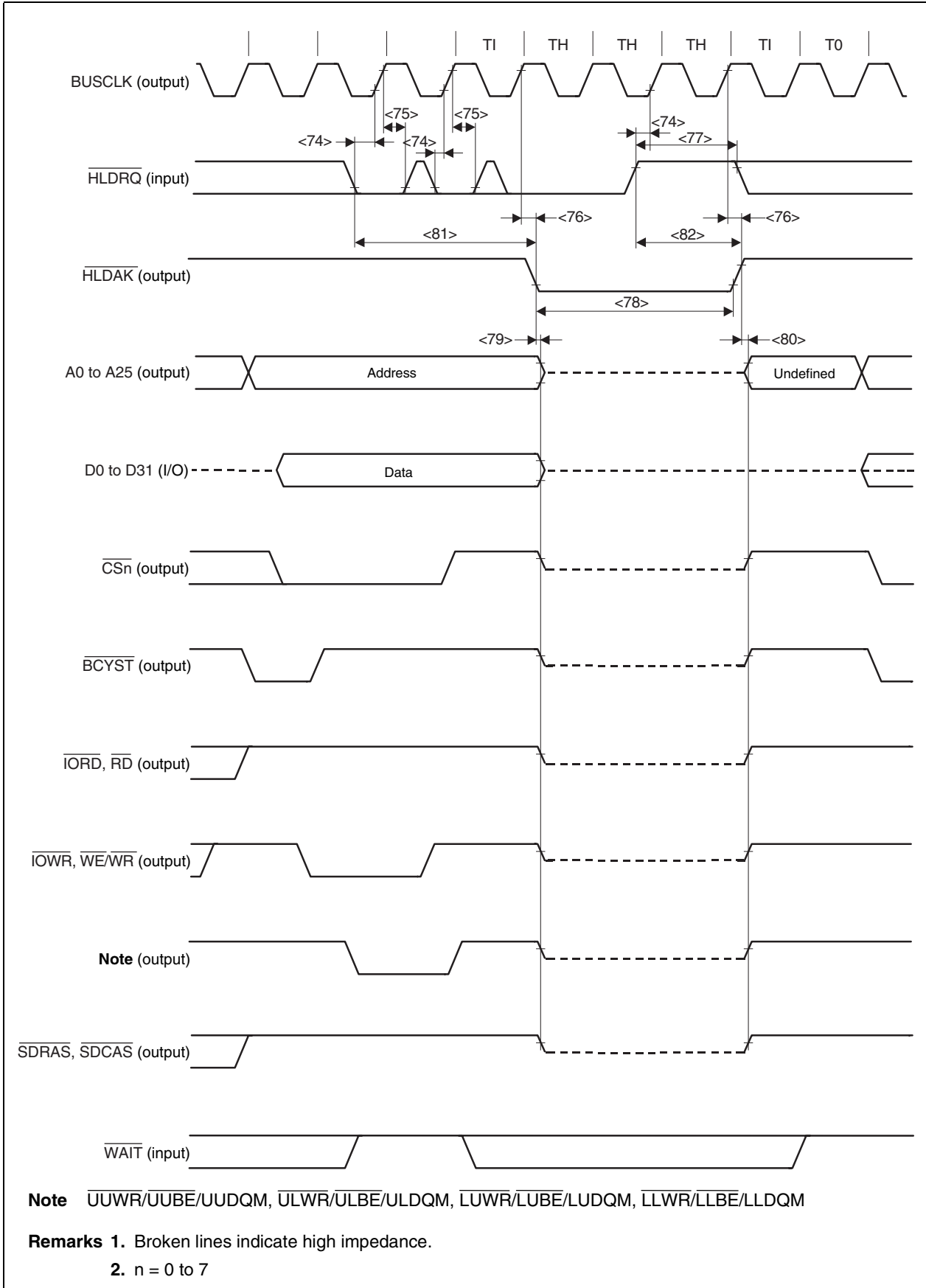
2. $n = 0$ to 3

(8) Bus hold timing (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{HLDRQ}}$ setup time (to $\text{BUSCLK}\uparrow$)	<74> t_{SHRK}		10		ns
$\overline{\text{HLDRQ}}$ hold time (from $\text{BUSCLK}\uparrow$)	<75> t_{HKHR}		2		ns
Delay time from $\text{BUSCLK}\uparrow$ to $\text{HLD}\overline{\text{AK}}$	<76> t_{DKHA}		2	11	ns
$\overline{\text{HLDRQ}}$ high-level width	<77> t_{WHQH}		$T + 12$		ns
$\text{HLD}\overline{\text{AK}}$ low-level width	<78> t_{WHAL}		$T - 11$		ns
Delay time from $\overline{\text{HLD}\overline{\text{AK}}}\downarrow$ to bus float	<79> t_{DKCF}		0		ns
Delay time from $\overline{\text{HLD}\overline{\text{AK}}}\uparrow$ to bus output	<80> t_{DHAC}		0	11	ns
Delay time from $\overline{\text{HLDRQ}}\downarrow$ to $\overline{\text{HLD}\overline{\text{AK}}}\downarrow$	<81> t_{DHQHA1}		$2T$		ns
Delay time from $\overline{\text{HLDRQ}}\uparrow$ to $\overline{\text{HLD}\overline{\text{AK}}}\uparrow$	<82> t_{DHQHA2}		T	$2T + 10$	ns

Remark $T = t_{\text{CYK}}$

(8) Bus hold timing (2/2)



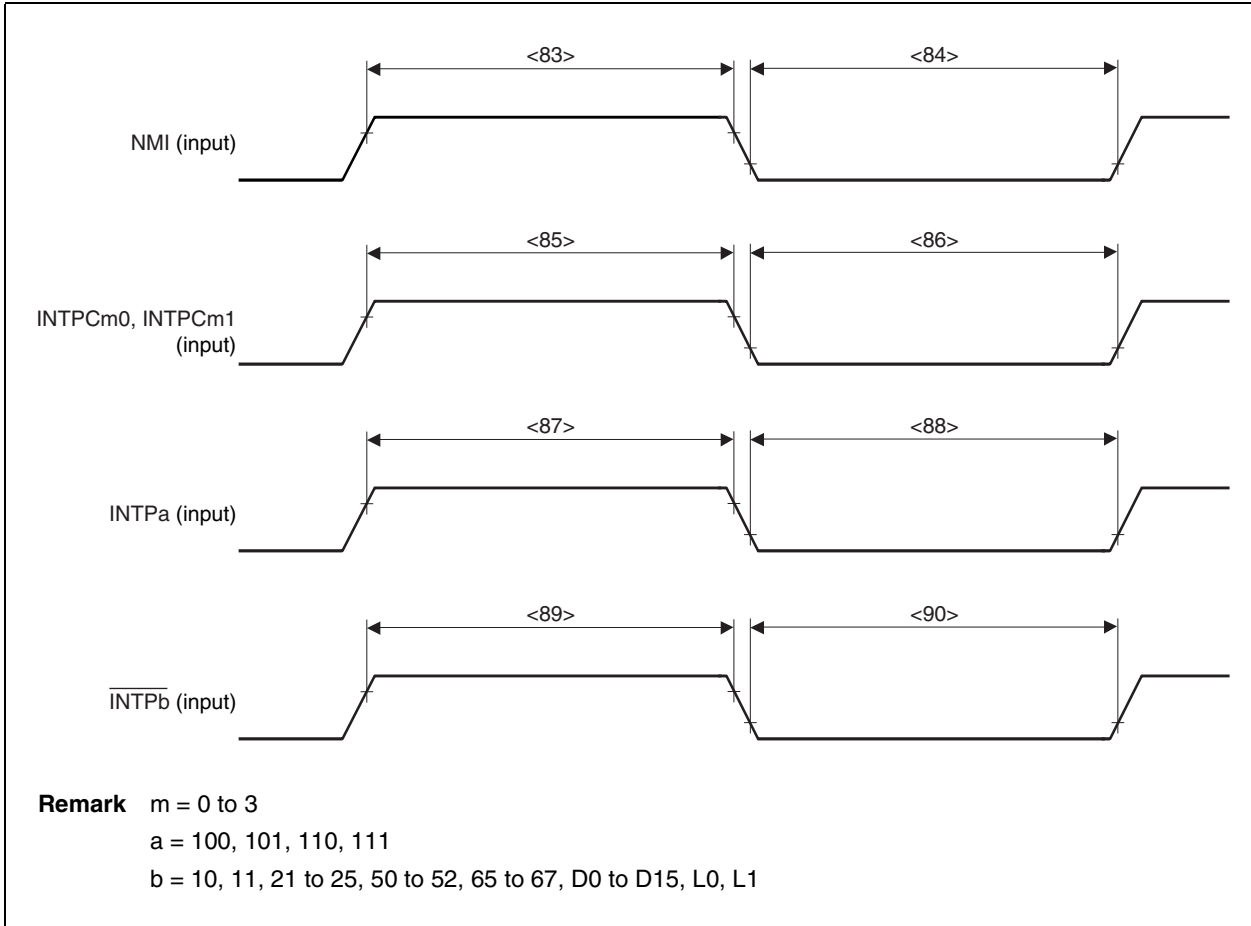
(9) Interrupt timing (1/2)

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
NMI high-level width	<83> t_{WNIH}		500		ns
NMI low-level width	<84> t_{WNIL}		500		ns
INTPCm0, INTPCm1 pin (m = 0 to 3) high-level width	<85> t_{WITCH}	When noise elimination has been set	(Number of set elimination clocks + 1)/(fx/4) + 10		ns
		When clock-through has been set	$1/(fx/4) \times 2 + 10$		ns
INTPCm0, INTPCm1 pin (m = 0 to 3) low-level width	<86> t_{WITCL}	When noise elimination has been set	(Number of set elimination clocks + 1)/(fx/4) + 10		ns
		When clock-through has been set	$1/(fx/4) \times 2 + 10$		ns
INTPa pin high-level width	<87> t_{WIT1H}	When noise elimination has been set	Noise elimination clock = fx/4 selected	(Number of set elimination clocks + 1)/(fx/4) + 10	ns
			Noise elimination clock = fx/32 selected	(Number of set elimination clocks + 1)/(fx/32) + 10	ns
		When clock-through has been set ^{Note}	Noise elimination clock = fx/4 selected	$1/(fx/4) \times 2 + 10$	ns
INTPa pin low-level width	<88> t_{WIT1L}	When noise elimination has been set	Noise elimination clock = fx/4 selected	(Number of set elimination clocks + 1)/(fx/4) + 10	ns
			Noise elimination clock = fx/32 selected	(Number of set elimination clocks + 1)/(fx/32) + 10	ns
		When clock-through has been set ^{Note}	Noise elimination clock = fx/4 selected	$1/(fx/4) \times 2 + 10$	ns
INTPb high-level width	<89> t_{WITPH}	Both edge and level detection	$500 + 1/(fx/4) + 10$		ns
INTPb low-level width	<90> t_{WITPL}	Both edge and level detection	$500 + 1/(fx/4) + 10$		ns

Note When clock-through has been set, do not select a noise elimination clock of fx/32.

- Remarks 1.** The noise elimination clock and clock-through are set values of the NCWC0 to NCWC3, NCW10, and NCW11 registers.
2. fx: Main clock
 3. a = 100, 101, 110, 111
b = 10, 11, 21 to 25, 50 to 52, 65 to 67, D0 to D15, L0, L1

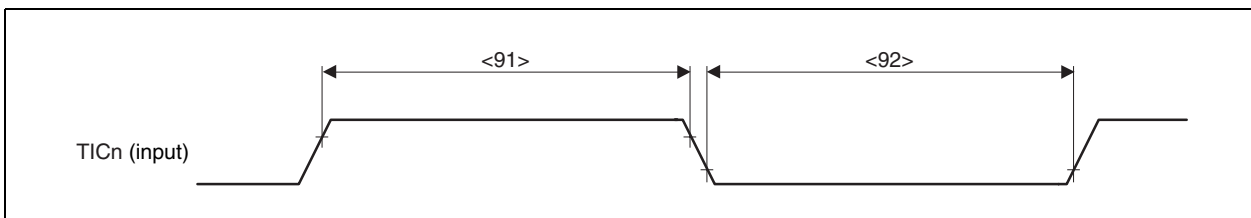
(9) Interrupt timing (2/2)



(10) Timer C timing

Parameter	Symbol	Conditions	MIN.	MAX.	Unit	
TICn high-level width	<91>	t _{WTCH}	When noise elimination has been set	(Number of set elimination clocks + 1) / (f _x /4) + 10		ns
			When clock-through has been set	1 / (f _x /4) × 2 + 10		ns
TICn low-level width	<92>	t _{WTCL}	When noise elimination has been set	(Number of set elimination clocks + 1) / (f _x /4) + 10		ns
			When clock-through has been set	1 / (f _x /4) × 2 + 10		ns

- Remarks** 1. n = 0 to 3
 2. The noise elimination clock and clock-through are set values of the NCWC0 to NCWC3 registers.



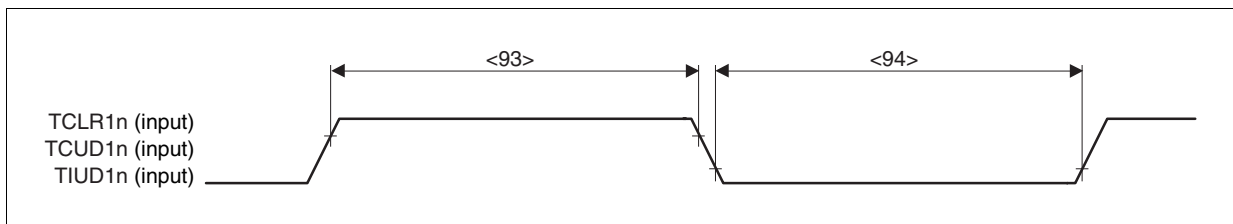
(11) Timer ENC1 timing

Parameter	Symbol	Conditions		MIN.	MAX.	Unit
TCLR1n, TCUD1n, TIUD1n pin high-level width	<93>	tWTENCH	When noise elimination has been set	Noise elimination clock = $f_x/4$ selected	(Number of set elimination clocks + 1)/($f_x/4$) + 10	ns
				Noise elimination clock = $f_x/32$ selected	(Number of set elimination clocks + 1)/($f_x/32$) + 10	ns
			When clock-through has been set ^{Note}	Noise elimination clock = $f_x/4$ selected	$1/(f_x/4) \times 2 + 10$	ns
TCLR1n, TCUD1n, TIUD1n pin low-level width	<94>	tWTENCL	When noise elimination has been set	Noise elimination clock = $f_x/4$ selected	(Number of set elimination clocks + 1)/($f_x/4$) + 10	ns
				Noise elimination clock = $f_x/32$ selected	(Number of set elimination clocks + 1)/($f_x/32$) + 10	ns
			When clock-through has been set ^{Note}	Noise elimination clock = $f_x/4$ selected	$1/(f_x/4) \times 2 + 10$	ns

Note When clock-through has been set, do not select a noise elimination clock of $f_x/32$.

Remarks 1. n = 0, 1

2. The noise elimination clock and clock-through are the set values of the NCW10 and NCW11 registers.



(12) CSI30, CSI31 timing (1/3)

(a) Master mode

Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCKn}}$ cycle	<95> t_{CYSK1}	output	182		ns
$\overline{\text{SCKn}}$ high-level width	<96> t_{WSK1H}	output	$0.5t_{\text{CYSK1}} - 12$		ns
$\overline{\text{SCKn}}$ low-level width	<97> t_{WSK1L}	output	$0.5t_{\text{CYSK1}} - 12$		ns
SIn setup time (to $\overline{\text{SCKn}}\uparrow$)	<98>	t_{SSISK}	12		ns
SIn setup time (to $\overline{\text{SCKn}}\downarrow$)			12		ns
SIn hold time (from $\overline{\text{SCKn}}\uparrow$)	<99>	t_{HSKSI}	5		ns
SIn hold time (from $\overline{\text{SCKn}}\downarrow$)			5		ns
SOn output delay time (from $\overline{\text{SCKn}}\downarrow$)	<100>	t_{DSKSO}		7	ns
SOn output delay time (from $\overline{\text{SCKn}}\uparrow$)				7	ns
SOn output hold time (from $\overline{\text{SCKn}}\uparrow$)	<101>	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

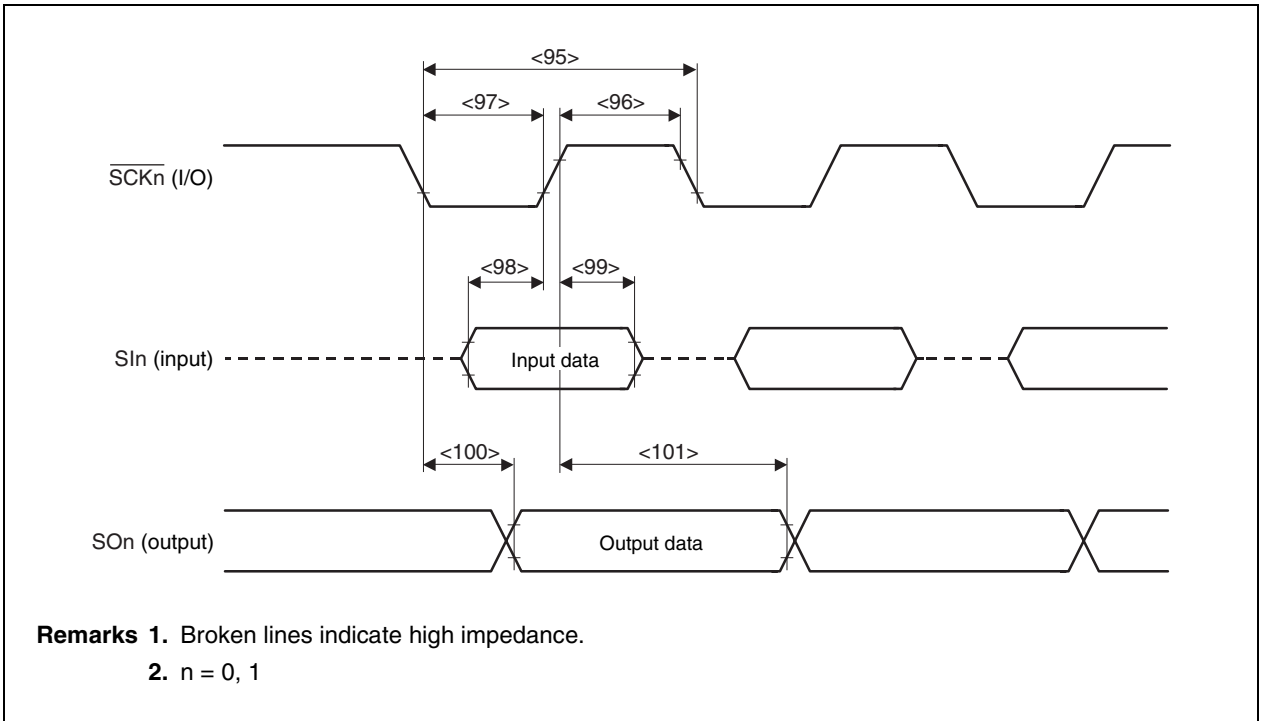
Parameter	Symbol	Conditions	MIN.	MAX.	Unit
$\overline{\text{SCKn}}$ cycle	<95> t_{CYSK1}	Input	182		ns
$\overline{\text{SCKn}}$ high-level width	<96> t_{WSK1H}	Input	$0.5t_{\text{CYSK1}} - 20$		ns
$\overline{\text{SCKn}}$ low-level width	<97> t_{WSK1L}	Input	$0.5t_{\text{CYSK1}} - 20$		ns
SIn setup time (to $\overline{\text{SCKn}}\uparrow$)	<98>	t_{SSISK}	30		ns
SIn setup time (to $\overline{\text{SCKn}}\downarrow$)			30		ns
SIn hold time (from $\overline{\text{SCKn}}\uparrow$)	<99>	t_{HSKSI}	$1.5T + 10$		ns
SIn hold time (from $\overline{\text{SCKn}}\downarrow$)			$1.5T + 10$		ns
SOn output delay time (from $\overline{\text{SCKn}}\downarrow$)	<100>	t_{DSKSO}		12	ns
SOn output delay time (from $\overline{\text{SCKn}}\uparrow$)				12	ns
SOn output hold time (from $\overline{\text{SCKn}}\uparrow$)	<101>	t_{HSKSO}	t_{WSK1H}		ns
SOn output hold time (from $\overline{\text{SCKn}}\downarrow$)			t_{WSK1H}		ns

Remarks 1. $n = 0, 1$

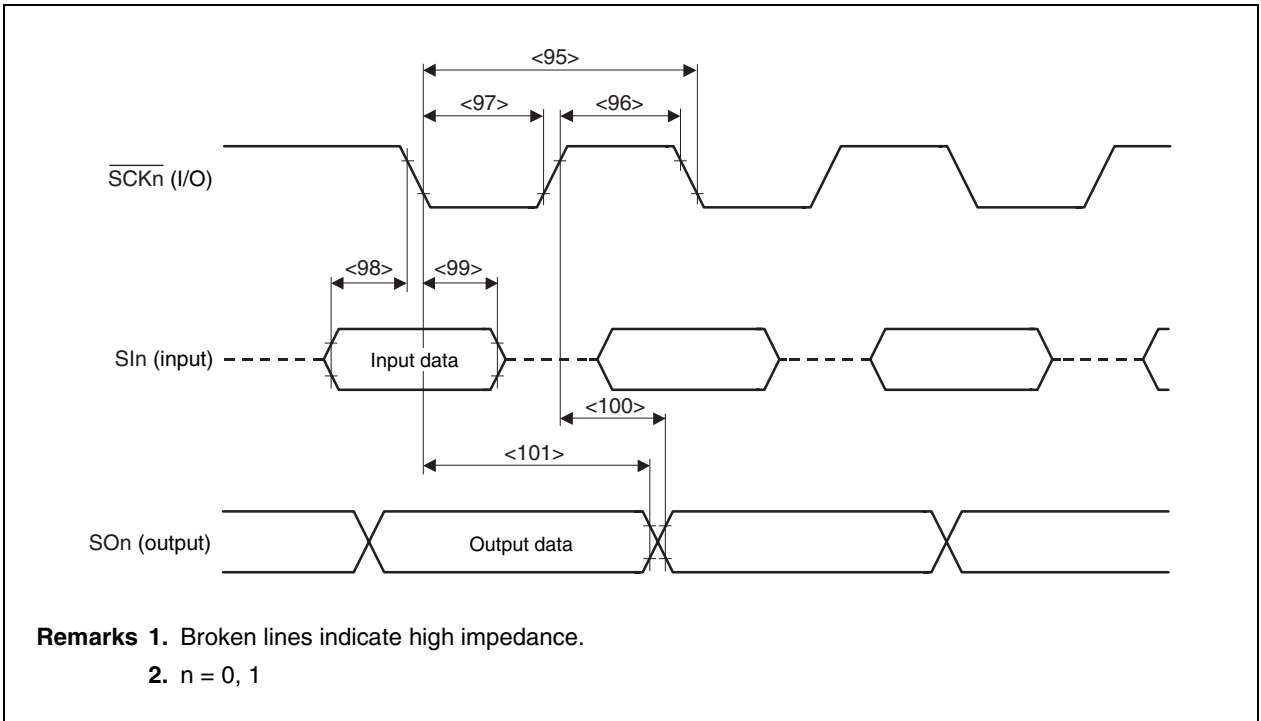
2. $T = f_x/4$

(12) CSI30, CSI31 timing (2/3)

(c) Timing when CKPn and DAPn bits of CSIC3n register = 00

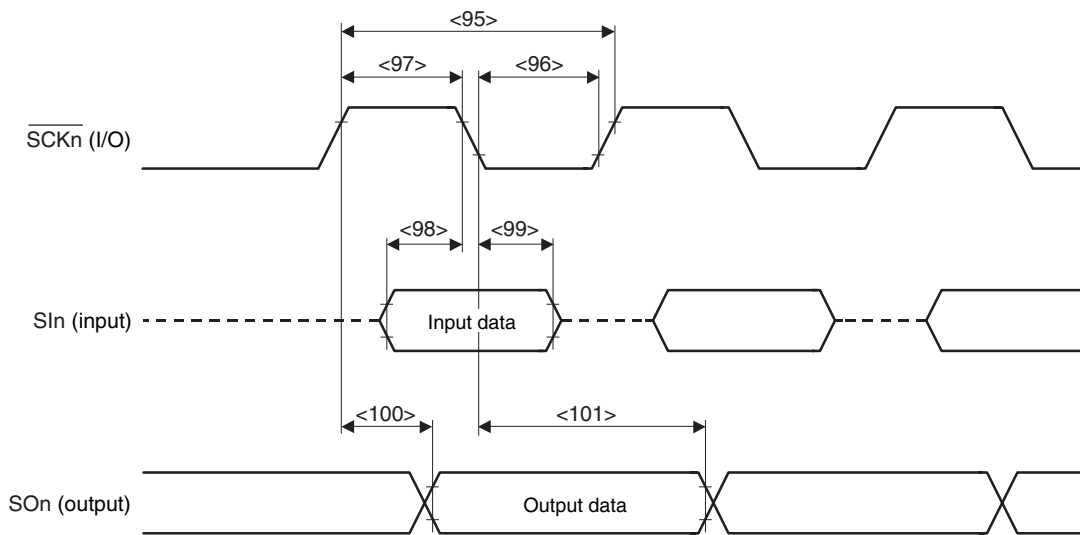


(d) Timing when CKPn and DAPn bits of CSIC3n register = 01



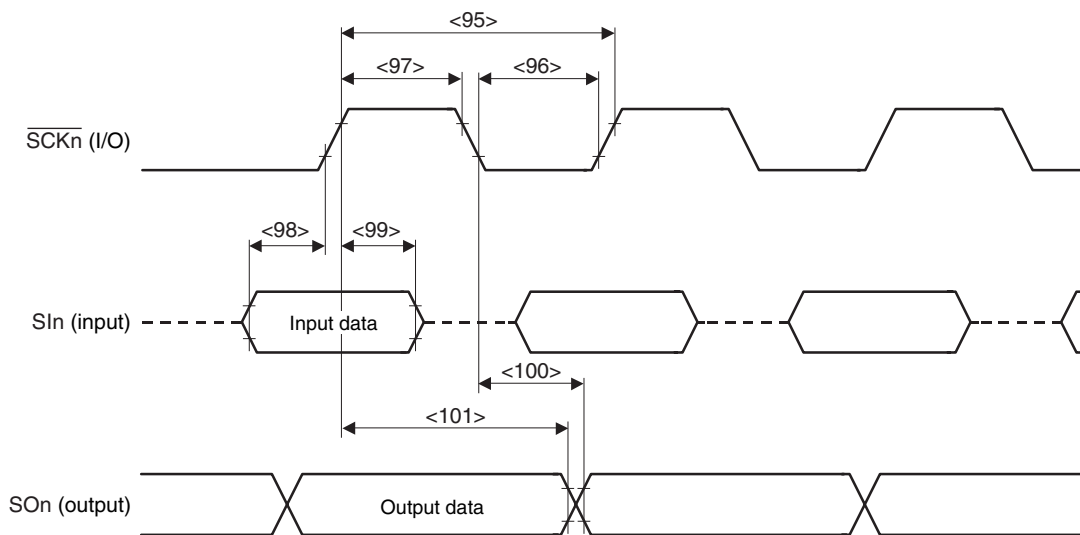
(12) CSI30, CSI31 timing (3/3)

(e) Timing when CKPn and DAPn bits of CSIC3n register = 10



- Remarks**
1. Broken lines indicate high impedance.
 2. $n = 0, 1$

(e) Timing when CKPn and DAPn bits of CSIC3n register = 11



- Remarks**
1. Broken lines indicate high impedance.
 2. $n = 0, 1$

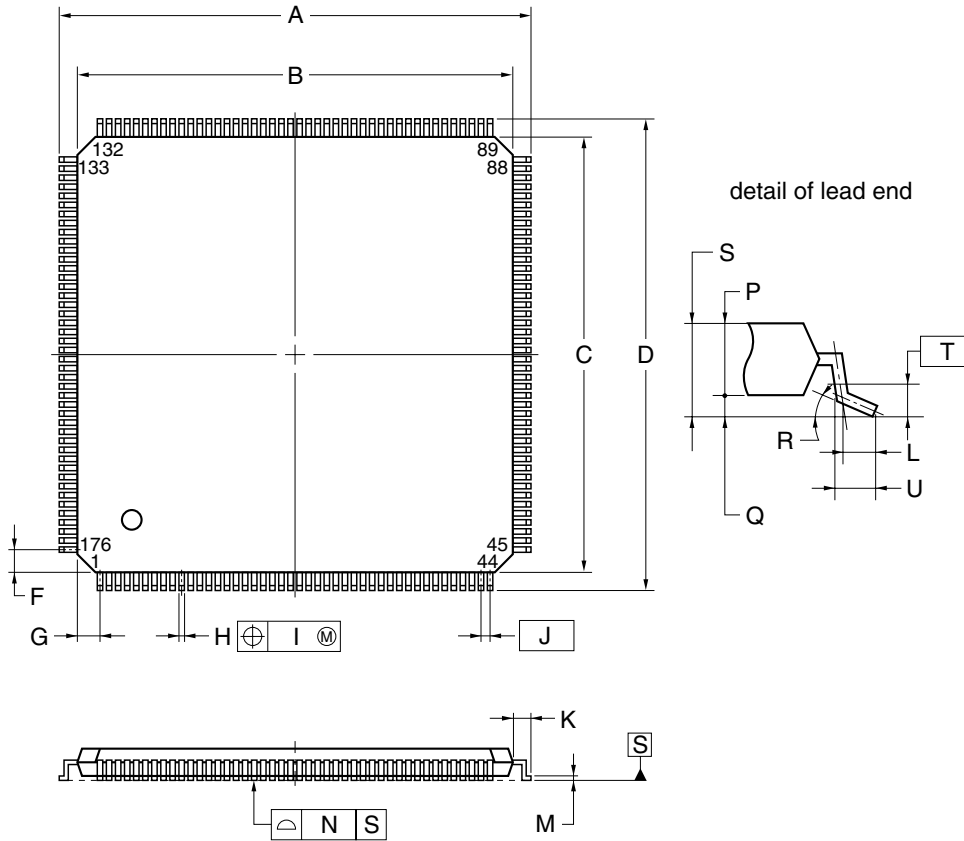
A/D Converter Characteristics (EV_{DD} = AV_{DD} = AV_{REFP} = 3.0 to 3.6 V, EV_{SS} = AV_{SS} = AV_{REFP} = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	–		10			bit
Overall error ^{Note 1}	–				±0.49	%FSR
Quantization error	–				±1/2	LSB
Conversion time	t _{CONV}		2.00		10	μs
Sampling time	t _{SAMP}		3 × conversion clock ^{Note 2} /16			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}		AV _{REFM}		AV _{REFP}	V
AV _{DD} power supply current	I _{DD}				10	mA

- Notes**
1. Excluding quantization error (±0.05%FSR).
 2. The conversion clock indicates the number of clocks set by the ADM1 register.
 3. Excluding quantization error (±0.5LSB).

Remark LSB: Least Significant Bit
FSR: Full Scale Range
%FSR indicates the ratio to the full-scale value.

176-PIN PLASTIC LQFP (FINE PITCH) (24x24)

**NOTE**

Each lead centerline is located within 0.10 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	26.0±0.2
B	24.0±0.2
C	24.0±0.2
D	26.0±0.2
F	1.25
G	1.25
H	0.22±0.05
I	0.08
J	0.5 (T.P.)
K	1.0±0.2
L	0.5
M	0.17 ^{+0.03} _{-0.07}
N	0.08
P	1.4
Q	0.1±0.05
R	3° ^{+4°} _{-3°}
S	1.5±0.1
T	0.25
U	0.60±0.15

S176GM-50-UEU-1

CHAPTER 19 RECOMMENDED SOLDERING CONDITIONS

The V850E/ME2 should be soldered and mounted under the following recommended conditions.

For soldering methods and conditions other than those recommended below, contact an NEC Electronics sales representative.

For technical information, see the following website.

Semiconductor Device Mount Manual (<http://www.necel.com/pkg/en/mount/index.html>)

Table 19-1. Surface Mounting Type Soldering Conditions

μPD703111GM-10-UEU: 176-pin plastic LQFP (fine pitch) (24 × 24)

μPD703111GM-13-UEU: 176-pin plastic LQFP (fine pitch) (24 × 24)

μPD703111GM-15-UEU: 176-pin plastic LQFP (fine pitch) (24 × 24)

Soldering	Soldering Conditions	Recommended Method Condition Symbol
Infrared reflow	Package peak temperature: 250°C, Time: 30 seconds max. (at 210°C or higher), Count: Three times or less Exposure limit: 3days ^{Note} (after that, prebake at 125°C for 20 hours)	IR50-203-3
Partial heating	Pin temperature: 350°C max., Time: 3 seconds max. (per pin row)	–

Caution Do not use different soldering methods together (except for partial heating).

APPENDIX A REGISTER INDEX

(1/13)

Symbol	Name	Unit	Page
ADCR0 to ADCR7	A/D conversion result registers 0 to 7	ADC	742
ADCR0H to ADCR7H	A/D conversion result registers 0H to 7H	ADC	742
ADIC	Interrupt control register 83	ADC	332
ADM0	A/D converter mode register 0	ADC	737
ADM1	A/D converter mode register 1	ADC	738
ADM2	A/D converter mode register 2	ADC	740
ADTS	ADC trigger select register	ADC	741
ASC	Address setup wait control register	BCU	154
BCC	Bus cycle control register	BCU	159
BCP	Bus cycle period control register	BCU	155
BCT0	Bus cycle type configuration register 0	BCU	121
BCT1	Bus cycle type configuration register 1	BCU	121
BEC	Endian configuration register	BCU	124
BHC	Cache configuration register	BCU	161
BMC	Bus mode control register	BCU	150
CC100	Capture/compare register 100	RPU	447
CC101	Capture/compare register 101	RPU	448
CC10IC0	Interrupt control register 53	INTC	332
CC10IC1	Interrupt control register 54	INTC	332
CC110	Capture/compare register 110	RPU	447
CC111	Capture/compare register 111	RPU	448
CC11IC0	Interrupt control register 59	INTC	332
CC11IC1	Interrupt control register 60	INTC	332
CCC00	Capture/compare register C00	RPU	406
CCC01	Capture/compare register C01	RPU	406
CCC0IC0	Interrupt control register 37	INTC	332
CCC0IC1	Interrupt control register 38	INTC	332
CCC10	Capture/compare register C10	RPU	406
CCC11	Capture/compare register C11	RPU	406
CCC1IC0	Interrupt control register 39	INTC	332
CCC1IC1	Interrupt control register 40	INTC	332
CCC20	Capture/compare register C20	RPU	406
CCC21	Capture/compare register C21	RPU	406
CCC2IC0	Interrupt control register 41	INTC	332
CCC2IC1	Interrupt control register 42	INTC	332

Symbol	Name	Unit	Page
CCC30	Capture/compare register C30	RPU	406
CCC31	Capture/compare register C31	RPU	406
CCC3IC0	Interrupt control register 43	INTC	332
CCC3IC1	Interrupt control register 44	INTC	332
CCC40	Capture/compare register C40	RPU	406
CCC41	Capture/compare register C41	RPU	406
CCC4IC0	Interrupt control register 45	INTC	332
CCC4IC1	Interrupt control register 46	INTC	332
CCC50	Capture/compare register C50	RPU	406
CCC51	Capture/compare register C51	RPU	406
CCC5IC0	Interrupt control register 47	INTC	332
CCC5IC1	Interrupt control register 48	INTC	332
CCR10	Capture/compare control register 10	RPU	452
CCR11	Capture/compare control register 11	RPU	452
CKC	Clock control register	CG	366
CKS	Clock source select register	CG	369
CM100	Compare register 100	RPU	446
CM101	Compare register 101	RPU	446
CM10IC0	Interrupt control register 55	INTC	332
CM10IC1	Interrupt control register 56	INTC	332
CM110	Compare register 110	RPU	446
CM111	Compare register 111	RPU	446
CM11IC0	Interrupt control register 61	INTC	332
CM11IC1	Interrupt control register 62	INTC	332
CMD0	Compare register D0	RPU	433
CMD1	Compare register D1	RPU	433
CMD2	Compare register D2	RPU	433
CMD3	Compare register D3	RPU	433
CMDIC0	Interrupt control register 49	INTC	332
CMDIC1	Interrupt control register 50	INTC	332
CMDIC2	Interrupt control register 51	INTC	332
CMDIC3	Interrupt control register 52	INTC	332
COVF3IC0	Interrupt control register 70	INTC	332
COVF3IC1	Interrupt control register 72	INTC	332
CSC0	Chip area select control register 0	BCU	117
CSC1	Chip area select control register 1	BCU	117
CSI3IC0	Interrupt control register 69	INTC	332
CSI3IC1	Interrupt control register 71	INTC	332
CSIC30	Clocked serial interface clock select register 30	CSI30	538

Symbol	Name	Unit	Page
CSIC31	Clocked serial interface clock select register 31	CSI31	538
CSIL30	Transfer data length select register 30	CSI30	546
CSIL31	Transfer data length select register 31	CSI31	546
CSIM30	Clocked serial interface mode register 30	CSI30	536
CSIM31	Clocked serial interface mode register 31	CSI31	536
DADC0	DMA addressing control register 0	DMAC	253
DADC1	DMA addressing control register 1	DMAC	253
DADC2	DMA addressing control register 2	DMAC	253
DADC3	DMA addressing control register 3	DMAC	253
DBC0	DMA transfer count register 0	DMAC	252
DBC1	DMA transfer count register 1	DMAC	252
DBC2	DMA transfer count register 2	DMAC	252
DBC3	DMA transfer count register 3	DMAC	252
DCHC0	DMA channel control register 0	DMAC	256
DCHC1	DMA channel control register 1	DMAC	256
DCHC2	DMA channel control register 2	DMAC	256
DCHC3	DMA channel control register 3	DMAC	256
DDA0H	DMA destination address register 0H	DMAC	250
DDA0L	DMA destination address register 0L	DMAC	251
DDA1H	DMA destination address register 1H	DMAC	250
DDA1L	DMA destination address register 1L	DMAC	251
DDA2H	DMA destination address register 2H	DMAC	250
DDA2L	DMA destination address register 2L	DMAC	251
DDA3H	DMA destination address register 3H	DMAC	250
DDA3L	DMA destination address register 3L	DMAC	251
DIFC	DMA interface control register	DMAC	266
DMAIC0	Interrupt control register 65	INTC	332
DMAIC1	Interrupt control register 66	INTC	332
DMAIC2	Interrupt control register 67	INTC	332
DMAIC3	Interrupt control register 68	INTC	332
DSA0H	DMA source address register 0H	DMAC	248
DSA0L	DMA source address register 0L	DMAC	249
DSA1H	DMA source address register 1H	DMAC	248
DSA1L	DMA source address register 1L	DMAC	249
DSA2H	DMA source address register 2H	DMAC	248
DSA2L	DMA source address register 2L	DMAC	249
DSA3H	DMA source address register 3H	DMAC	248
DSA3L	DMA source address register 3L	DMAC	249
DTFR0	DMA trigger factor register 0	DMAC	260

Symbol	Name	Unit	Page
DTFR1	DMA trigger factor register 1	DMAC	260
DTFR2	DMA trigger factor register 2	DMAC	260
DTFR3	DMA trigger factor register 3	DMAC	260
DTOC	DMA terminal count output control register	DMAC	259
DWC0	Data wait control register 0	BCU	152
DWC1	Data wait control register 1	BCU	152
FIC	DMA flyby transfer idle control register	BCU	160
FWC	DMA flyby transfer wait control register	BCU	156
ICC	Instruction cache control register	BCU	164
ICCH	Instruction cache control register H	BCU	164
ICCL	Instruction cache control register L	BCU	164
ICD	Instruction cache data configuration register	BCU	165
IMR0	Interrupt mask register 0	INTC	335
IMR0H	Interrupt mask register 0H	INTC	335
IMR0L	Interrupt mask register 0L	INTC	335
IMR1	Interrupt mask register 1	INTC	335
IMR1H	Interrupt mask register 1H	INTC	335
IMR1L	Interrupt mask register 1L	INTC	335
IMR2	Interrupt mask register 2	INTC	335
IMR2H	Interrupt mask register 2H	INTC	335
IMR2L	Interrupt mask register 2L	INTC	335
IMR3	Interrupt mask register 3	INTC	335
IMR3H	Interrupt mask register 3H	INTC	335
IMR3L	Interrupt mask register 3L	INTC	335
IMR4	Interrupt mask register 4	INTC	335
IMR4H	Interrupt mask register 4H	INTC	335
IMR4L	Interrupt mask register 4L	INTC	335
IMR5	Interrupt mask register 5	INTC	335
IMR5H	Interrupt mask register 5H	INTC	335
IMR5L	Interrupt mask register 5L	INTC	335
INTF1	External interrupt falling edge specification register 1	INTC	339, 812
INTF2	External interrupt falling edge specification register 2	INTC	324, 341, 817
INTF5	External interrupt falling edge specification register 5	INTC	343, 822
INTF6	External interrupt falling edge specification register 6	INTC	345, 827
INTFAL	External interrupt falling edge specification register AL	INTC	347, 835
INTFDH	External interrupt falling edge specification register DH	INTC	348, 846
INTFDHH	External interrupt falling edge specification register DHH	INTC	348, 846
INTFDHL	External interrupt falling edge specification register DHL	INTC	348, 846

Symbol	Name	Unit	Page
INTR1	External interrupt rising edge specification register 1	INTC	339, 812
INTR2	External interrupt rising edge specification register 2	INTC	324, 341, 817
INTR5	External interrupt rising edge specification register 5	INTC	343, 822
INTR6	External interrupt rising edge specification register 6	INTC	345, 827
INTRAL	External interrupt rising edge specification register AL	INTC	347, 835
INTRDH	External interrupt rising edge specification register DH	INTC	348, 846
INTRDHH	External interrupt rising edge specification register DHH	INTC	348, 846
INTRDHL	External interrupt rising edge specification register DHL	INTC	348, 846
IRAMM	Internal instruction RAM mode register	BCU	168
ISPR	In-service priority register	INTC	338
LBC0	Line buffer control register 0	BCU	146
LBC1	Line buffer control register 1	BCU	146
LBS	Local bus sizing control register	BCU	123
LOCKR	Lock register	CPU	374
NCW10	Noise elimination width setting register 10	RPU	458, 865
NCW11	Noise elimination width setting register 11	RPU	458, 865
NCWC0	Noise elimination width setting register C0	RPU	415, 864
NCWC1	Noise elimination width setting register C1	RPU	415, 864
NCWC2	Noise elimination width setting register C2	RPU	415, 864
NCWC3	Noise elimination width setting register C3	RPU	415, 864
NRS	NMI reset status register	INTC	337
OSTS	Oscillation stabilization time select register	CG	375
OV1IC0	Interrupt control register 57	INTC	332
OV1IC1	Interrupt control register 63	INTC	332
OVCIC0	Interrupt control register 31	INTC	332
OVCIC1	Interrupt control register 32	INTC	332
OVCIC2	Interrupt control register 33	INTC	332
OVCIC3	Interrupt control register 34	INTC	332
OVCIC4	Interrupt control register 35	INTC	332
OVCIC5	Interrupt control register 36	INTC	332
P1	Port 1	Port	809
P1IC0	Interrupt control register 0	INTC	332
P1IC1	Interrupt control register 1	INTC	332
P2	Port 2	Port	814
P2IC1	Interrupt control register 2	INTC	332
P2IC2	Interrupt control register 3	INTC	332
P2IC3	Interrupt control register 4	INTC	332
P2IC4	Interrupt control register 5	INTC	332

Symbol	Name	Unit	Page
P2IC5	Interrupt control register 6	INTC	332
P5	Port 5	Port	819
P5IC0	Interrupt control register 7	INTC	332
P5IC1	Interrupt control register 8	INTC	332
P5IC2	Interrupt control register 9	INTC	332
P6	Port 6	Port	824
P6IC5	Interrupt control register 10	INTC	332
P6IC6	Interrupt control register 11	INTC	332
P6IC7	Interrupt control register 12	INTC	332
P7	Port 7	Port	829
PAH	Port AH	Port	837
PAHH	Port AHH	Port	837
PAHL	Port AHL	Port	837
PAL	Port AL	Port	832
PALH	Port ALH	Port	832
PALL	Port ALL	Port	832
PCD	Port CD	Port	857
PCM	Port CM	Port	854
PCS	Port CS	Port	848
PCT	Port CT	Port	851
PDH	Port DH	Port	839
PDHH	Port DHH	Port	839
PDHL	Port DHL	Port	839
PDIC0	Interrupt control register 13	INTC	332
PDIC1	Interrupt control register 14	INTC	332
PDIC2	Interrupt control register 15	INTC	332
PDIC3	Interrupt control register 16	INTC	332
PDIC4	Interrupt control register 17	INTC	332
PDIC5	Interrupt control register 18	INTC	332
PDIC6	Interrupt control register 19	INTC	332
PDIC7	Interrupt control register 20	INTC	332
PDIC8	Interrupt control register 21	INTC	332
PDIC9	Interrupt control register 22	INTC	332
PDIC10	Interrupt control register 23	INTC	332
PDIC11	Interrupt control register 24	INTC	332
PDIC12	Interrupt control register 25	INTC	332
PDIC13	Interrupt control register 26	INTC	332
PDIC14	Interrupt control register 27	INTC	332
PDIC15	Interrupt control register 28	INTC	332

Symbol	Name	Unit	Page
PFC1	Port 1 function control register	Port	811
PFC2	Port 2 function control register	Port	816
PFC5	Port 5 function control register	Port	821
PFC6	Port 6 function control register	Port	826
PFC7	Port 7 function control register	Port	831
PFCALL	Port AL function control register L	Port	834
PFCCM	Port CM function control register	Port	856
PFCCS	Port CS function control register	Port	850
PFCCT	Port CT function control register	Port	853
PFCDH	Port DH function control register	Port	843
PFCDHH	Port DH function control register H	Port	843
PFCDHL	Port DH function control register L	Port	843
PLIC0	Interrupt control register 29	INTC	332
PLIC1	Interrupt control register 30	INTC	332
PM1	Port 1 mode register	Port	810
PM2	Port 2 mode register	Port	815
PM5	Port 5 mode register	Port	820
PM6	Port 6 mode register	Port	825
PM7	Port 7 mode register	Port	830
PMAH	Port AH mode register	Port	838
PMAHH	Port AH mode register H	Port	838
PMAHL	Port AH mode register L	Port	838
PMAL	Port AL mode register	Port	833
PMALH	Port AL mode register H	Port	833
PMALL	Port AL mode register L	Port	833
PMC1	Port 1 mode control register	Port	810
PMC2	Port 2 mode control register	Port	815
PMC5	Port 5 mode control register	Port	820
PMC6	Port 6 mode control register	Port	825
PMC7	Port 7 mode control register	Port	830
PMCAH	Port AH mode control register	Port	838
PMCAHH	Port AH mode control register H	Port	838
PMCAHL	Port AH mode control register L	Port	838
PMCAL	Port AL mode control register	Port	833
PMCALH	Port AL mode control register H	Port	833
PMCALL	Port AL mode control register L	Port	833
PMCCD	Port CD mode control register	Port	858
PMCCM	Port CM mode control register	Port	855
PMCCS	Port CS mode control register	Port	849

Symbol	Name	Unit	Page
PMCCT	Port CT mode control register	Port	852
PMCD	Port CD mode register	Port	858
PMCDH	Port DH mode control register	Port	841
PMCDHH	Port DH mode control register H	Port	841
PMCDHL	Port DH mode control register L	Port	841
PMCM	Port CM mode register	Port	855
PMCS	Port CS mode register	Port	849
PMCT	Port CT mode register	Port	852
PMDH	Port DH mode register	Port	841
PMDHH	Port DH mode register H	Port	841
PMDHL	Port DH mode register L	Port	841
PRC	Page ROM configuration register	MEMC	199
PRCMD	Command register	CPU	383
PRM10	Prescaler mode register 10	RPU	455
PRM11	Prescaler mode register 11	RPU	455
PSC	Power-save control register	CPU	384
PSMR	Power-save mode register	CPU	383
PWM0	PWM modulo register 0	PWM	771
PWM1	PWM modulo register 1	PWM	771
PWMC0	PWM control register 0	PWM	769
PWMC1	PWM control register 1	PWM	769
PWMH0	PWM modulo register H0	PWM	771
PWMH1	PWM modulo register H1	PWM	771
PWML0	PWM modulo register L0	PWM	771
PWML1	PWM modulo register L1	PWM	771
RFS1	SDRAM refresh control register 1	MEMC	236
RFS3	SDRAM refresh control register 3	MEMC	236
RFS4	SDRAM refresh control register 4	MEMC	236
RFS6	SDRAM refresh control register 6	MEMC	236
RSUMIC	Interrupt control register 89	INTC	332
SCR1	SDRAM configuration register 1	MEMC	209
SCR3	SDRAM configuration register 3	MEMC	209
SCR4	SDRAM configuration register 4	MEMC	209
SCR6	SDRAM configuration register 6	MEMC	209
SESA10	Valid edge select register 10	INTC	352, 453
SESA11	Valid edge select register 11	INTC	352, 453
SESC0	Valid edge select register C0	INTC	350, 413
SESC1	Valid edge select register C1	INTC	350, 413
SESC2	Valid edge select register C2	INTC	350, 413

Symbol	Name	Unit	Page
SESC3	Valid edge select register C3	INTC	350, 413
SFA30	CSIBUF status register 30	CSI30	543
SFA31	CSIBUF status register 31	CSI31	543
SFDB30	Transmit data CSI buffer register 30	CSI30	542
SFDB30H	Transmit data CSI buffer register 30H	CSI30	542
SFDB30L	Transmit data CSI buffer register 30L	CSI30	542
SFDB31	Transmit data CSI buffer register 31	CSI31	542
SFDB31H	Transmit data CSI buffer register 31H	CSI31	542
SFDB31L	Transmit data CSI buffer register 31L	CSI31	542
SFN30	Transfer data number specification register 30	CSI30	547
SFN31	Transfer data number specification register 31	CSI31	547
SIRB30	Receive data buffer register 30	CSI30	541
SIRB30H	Receive data buffer register 30H	CSI31	541
SIRB30L	Receive data buffer register 30L	CSI30	541
SIRB31	Receive data buffer register 31	CSI31	541
SIRB31H	Receive data buffer register 31H	CSI31	541
SIRB31L	Receive data buffer register 31L	CSI31	541
SSCGC	SSCG control register	CG	371
STATUS10	Status register 10	RPU	457
STATUS11	Status register 11	RPU	457
TMC0	Timer C0	RPU	404
TMC1	Timer C1	RPU	404
TMC10	Timer control register 10	RPU	450
TMC11	Timer control register 11	RPU	450
TMC2	Timer C2	RPU	404
TMC3	Timer C3	RPU	404
TMC4	Timer C4	RPU	404
TMC5	Timer C5	RPU	404
TMCC00	Timer mode control register C00	RPU	408
TMCC01	Timer mode control register C01	RPU	410
TMCC10	Timer mode control register C10	RPU	408
TMCC11	Timer mode control register C11	RPU	410
TMCC20	Timer mode control register C20	RPU	408
TMCC21	Timer mode control register C21	RPU	410
TMCC30	Timer mode control register C30	RPU	408
TMCC31	Timer mode control register C31	RPU	410
TMCC40	Timer mode control register C40	RPU	408
TMCC41	Timer mode control register C41	RPU	410
TMCC50	Timer mode control register C50	RPU	408

Symbol	Name	Unit	Page
TMCC51	Timer mode control register C51	RPU	410
TMCD0	Timer mode control register D0	RPU	435
TMCD1	Timer mode control register D1	RPU	435
TMCD2	Timer mode control register D2	RPU	435
TMCD3	Timer mode control register D3	RPU	435
TMD0	Timer D0	RPU	432
TMD1	Timer D1	RPU	432
TMD2	Timer D2	RPU	432
TMD3	Timer D3	RPU	432
TMENC10	Timer ENC10	RPU	444
TMENC11	Timer ENC11	RPU	444
TUM10	Timer unit mode register 10	RPU	449
TUM11	Timer unit mode register 11	RPU	449
UB0CTL0	UARTB0 control register 0	UARTB	482
UB0CTL2	UARTB0 control register 2	UARTB	487
UB0FIC0	UARTB0 FIFO control register 0	UARTB	491
UB0FIC1	UARTB0 FIFO control register 1	UARTB	493
UB0FIC2	UARTB0 FIFO control register 2	UARTB	494
UB0FIC2H	UARTB0 FIFO control register 2H	UARTB	494
UB0FIC2L	UARTB0 FIFO control register 2L	UARTB	494
UB0FIS0	UARTB0 FIFO status register 0	UARTB	497
UB0FIS1	UARTB0 FIFO status register 1	UARTB	498
UB0RX	UARTB0 receive data register	UARTB	489
UB0RXAP	UARTB0 receive data register AP	UARTB	489
UB0STR	UARTB0 status register	UARTB	485
UB0TX	UARTB0 transmit data register	UARTB	488
UB1CTL0	UARTB1 control register 0	UARTB	482
UB1CTL2	UARTB1 control register 2	UARTB	487
UB1FIC0	UARTB1 FIFO control register 0	UARTB	491
UB1FIC1	UARTB1 FIFO control register 1	UARTB	493
UB1FIC2	UARTB1 FIFO control register 2	UARTB	494
UB1FIC2H	UARTB1 FIFO control register 2H	UARTB	494
UB1FIC2L	UARTB1 FIFO control register 2L	UARTB	494
UB1FIS0	UARTB1 FIFO status register 0	UARTB	497
UB1FIS1	UARTB1 FIFO status register 1	UARTB	498
UB1RX	UARTB1 receive data register	UARTB	489
UB1RXAP	UARTB1 receive data register AP	UARTB	489
UB1STR	UARTB1 status register	UARTB	485
UB1TX	UARTB1 transmit data register	UARTB	488

Symbol	Name	Unit	Page
UCKC	USB clock control register	CG	373
UD1IC0	Interrupt control register 58	INTC	332
UD1IC1	Interrupt control register 64	INTC	332
UF0AAS	UF0 active alternative setting register	USBF	631
UF0ADRS	UF0 address register	USBF	672
UF0AIFN	UF0 active interface number register	USBF	630
UF0ASS	UF0 alternative setting status register	USBF	632
UF0BC	USB function 0 buffer control register	USBF	681
UF0BI1	UF0 bulk in 1 register	USBF	652
UF0BI2	UF0 bulk in 2 register	USBF	656
UF0BO1	UF0 bulk out 1 register	USBF	645
UF0BO1L	UF0 bulk out 1 length register	USBF	648
UF0BO2	UF0 bulk out 2 register	USBF	649
UF0BO2L	UF0 bulk out 2 length register	USBF	652
UF0CIE0 to UF0CIE255	UF0 configuration/interface/endpoint descriptor registers 0 to 255	USBF	678
UF0CLR	UF0 CLR request register	USBF	595
UF0CNF	UF0 configuration register	USBF	673
UF0CS	USB function 0 DMA channel select register	USBF	680
UF0DD0 to UF0DD17	UF0 device descriptor registers 0 to 17	USBF	677
UF0DEND	UF0 data end register	USBF	625
UF0DMS0	UF0 DMA status 0 register	USBF	621
UF0DMS1	UF0 DMA status 1 register	USBF	622
UF0DSDL	UF0 descriptor length register	USBF	676
UF0DSTL	UF0 device status register L	USBF	664
UF0E0L	UF0 EP0 length register	USBF	640
UF0E0N	UF0 EP0NAK register	USBF	586
UF0E0NA	UF0 EP0NAKALL register	USBF	588
UF0E0R	UF0 EP0 read register	USBF	639
UF0E0SL	UF0 EP0 status register L	USBF	665
UF0E0ST	UF0 EP0 setup register	USBF	641
UF0E0W	UF0 EP0 write register	USBF	643
UF0E1IM	UF0 endpoint 1 interface mapping register	USBF	633
UF0E1SL	UF0 EP1 status register L	USBF	666
UF0E2IM	UF0 endpoint 2 interface mapping register	USBF	634
UF0E2SL	UF0 EP2 status register L	USBF	667
UF0E3IM	UF0 endpoint 3 interface mapping register	USBF	635
UF0E3SL	UF0 EP3 status register L	USBF	668

Symbol	Name	Unit	Page
UF0E4IM	UF0 endpoint 4 interface mapping register	USBF	636
UF0E4SL	UF0 EP4 status register L	USBF	669
UF0E7IM	UF0 endpoint 7 interface mapping register	USBF	637
UF0E7SL	UF0 EP7 status register L	USBF	670
UF0E8IM	UF0 endpoint 8 interface mapping register	USBF	638
UF0E8SL	UF0 EP8 status register L	USBF	671
UF0EN	UF0 EPNACK register	USBF	589
UF0ENM	UF0 EPNACK mask register	USBF	593
UF0EPS0	UF0 EP status 0 register	USBF	597
UF0EPS1	UF0 EP status 1 register	USBF	599
UF0EPS2	UF0 EP status 2 register	USBF	600
UF0FIC0	UF0 FIFO clear 0 register	USBF	623
UF0FIC1	UF0 FIFO clear 1 register	USBF	624
UF0GPR	UF0 GPR register	USBF	627
UF0IC0	UF0 INT clear 0 register	USBF	614
UF0IC1	UF0 INT clear 1 register	USBF	615
UF0IC2	UF0 INT clear 2 register	USBF	616
UF0IC3	UF0 INT clear 3 register	USBF	617
UF0IC4	UF0 INT clear 4 register	USBF	618
UF0IDR	UF0 INT & DMARQ register	USBF	619
UF0IF0	UF0 interface 0 register	USBF	674
UF0IF1	UF0 interface 1 register	USBF	675
UF0IF2	UF0 interface 2 register	USBF	675
UF0IF3	UF0 interface 3 register	USBF	675
UF0IF4	UF0 interface 4 register	USBF	675
UF0IM0	UF0 INT mask 0 register	USBF	609
UF0IM1	UF0 INT mask 1 register	USBF	610
UF0IM2	UF0 INT mask 2 register	USBF	611
UF0IM3	UF0 INT mask 3 register	USBF	612
UF0IM4	UF0 INT mask 4 register	USBF	613
UF0INT1	UF0 interrupt 1 register	USBF	660
UF0INT2	UF0 interrupt 2 register	USBF	662
UF0IS0	UF0 INT status 0 register	USBF	601
UF0IS1	UF0 INT status 1 register	USBF	603
UF0IS2	UF0 INT status 2 register	USBF	605
UF0IS3	UF0 INT status 3 register	USBF	606
UF0IS4	UF0 INT status 4 register	USBF	608
UF0MODC	UF0 mode control register	USBF	628
UF0MODS	UF0 mode status register	USBF	629

Symbol	Name	Unit	Page
UF0SDS	UF0 SNDSIE register	USBF	594
UF0SET	UF0 SET request register	USBF	596
UIFIC0	Interrupt control register 76	INTC	332
UIFIC1	Interrupt control register 81	INTC	332
UREIC0	Interrupt control register 73	INTC	332
UREIC1	Interrupt control register 78	INTC	332
URIC0	Interrupt control register 74	INTC	332
URIC1	Interrupt control register 79	INTC	332
US0BIC	Interrupt control register 84	INTC	332
US1BIC	Interrupt control register 85	INTC	332
US2BIC	Interrupt control register 86	INTC	332
USP2IC	Interrupt control register 87	INTC	332
USP4IC	Interrupt control register 88	INTC	332
UTIC0	Interrupt control register 75	INTC	332
UTIC1	Interrupt control register 80	INTC	332
UTOIC0	Interrupt control register 77	INTC	332
UTOIC1	Interrupt control register 82	INTC	332
VSWC	System wait control register	BCU	111

APPENDIX B INSTRUCTION SET LIST

B.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)
Halfword	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

B.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	x	x	x	x		
	imm5,reg2	rrrrr010010iiii	GR[reg2]←GR[reg2]+sign-extend(imm5)	1	1	1	x	x	x	x		
ADDI	imm16,reg1,reg2	rrrrr110000RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1	x	x	x	x		
AND	reg1,reg2	rrrrr001010RRRRR	GR[reg2]←GR[reg2]AND GR[reg1]	1	1	1		0	x	x		
ANDI	imm16,reg1,reg2	rrrrr110110RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]AND zero-extend(imm16)	1	1	1		0	0	x		
Bcond	disp9	dddd1011ddcccc Note 1	if conditions are satisfied then PC←PC+sign-extend(disp9)	When conditions are satisfied	3	3	3					
				When conditions are not satisfied	1	1	1					
BSH	reg2,reg3	rrrrr11111100000 www01101000010	GR[reg3]←GR[reg2] (23 : 16) GR[reg2] (31 : 24) GR[reg2] (7 : 0) GR[reg2] (15 : 8)	1	1	1	x	0	x	x		
BSW	reg2,reg3	rrrrr11111100000 www01101000000	GR[reg3]←GR[reg2] (7 : 0) GR[reg2] (15 : 8) GR [reg2] (23 : 16) GR[reg2] (31 : 24)	1	1	1	x	0	x	x		
CALLT	imm6	0000001000iiii	CTPC←PC+2(return PC) CTPSW←PSW adr←CTBP+zero-extend(imm6 logically shift left by 1) PC←CTBP+zero-extend(Load-memory(adr,Halfword))	5	5	5						
CLR1	bit#3, disp16[reg1]	10bbb111110RRRRR dddddddddddd	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				x		
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100100	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,0)	3	3	3				x		
CMOV	cccc,imm5,reg2,reg3	rrrrr111111iiii www011000cccc0	if conditions are satisfied then GR[reg3]←sign-extended(imm5) else GR[reg3]←GR[reg2]	1	1	1						
	cccc,reg1,reg2,reg3	rrrrr111111RRRRR www011001cccc0	if conditions are satisfied then GR[reg3]←GR[reg1] else GR[reg3]←GR[reg2]	1	1	1						
CMP	reg1,reg2	rrrrr001111RRRRR	result←GR[reg2]-GR[reg1]	1	1	1	x	x	x	x		
	imm5,reg2	rrrrr010011iiii	result←GR[reg2]-sign-extend(imm5)	1	1	1	x	x	x	x		
CTRET		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 B 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←0000060H	4	4	4						
DI		0000011111100000 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 wwwww0101000000	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 wwwww0101000000	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		1000011111100000 0000000101100000	PSW.ID←0	1	1	1						
HALT		0000011111100000 0000000100100000	Stop	1	1	1						
HSW	reg2,reg3	rrrrr11111100000 wwwww01101000100	GR[reg3]←GR[reg2](15 : 0) GR[reg2] (31 : 16)	1	1	1	×	0	×	×		
JARL	disp22,reg2	rrrrr11110dddddd dddddddddddddd0 Note 7	GR[reg2]←PC+4 PC←PC+sign-extend(disp22)	3	3	3						
JMP	[reg1]	00000000011RRRRR	PC←GR[reg1]	4	4	4						
JR	disp22	0000011110dddddd dddddddddddddd0 Note 7	PC←PC+sign-extend(disp22)	3	3	3						
LD.B	disp16[reg1],reg2	rrrrr111000RRRRR 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 B 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	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←sign-extend(Load-memory(adr,Halfword))	1	1	Note 11						
LDSR	reg2,regID	rrrrr111111RRRRR 000000000100000	SR[regID]←GR[reg2]	Other than regID = PSW	1	1	1					
				regID = PSW	1	1	1	x	x	x	x	x
LD.HU	disp16[reg1],reg2	rrrrr111111RRRRR dddddddddddddd1	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←zero-extend(Load-memory(adr,Halfword))	1	1	Note 11						
LD.W	disp16[reg1],reg2	rrrrr111001RRRRR dddddddddddddd1	adr←GR[reg1]+sign-extend(disp16) GR[reg2]←Load-memory(adr,Word)	1	1	Note 11						
MOV	reg1,reg2	rrrrr00000RRRRR	GR[reg2]←GR[reg1]	1	1	1						
	imm5,reg2	rrrrr010000iiii	GR[reg2]←sign-extend(imm5)	1	1	1						
	imm32,reg1	00000110001RRRRR iiiiiiiiiiiiiiii iiiiiiiiiiiiiiii	GR[reg1]←imm32	2	2	2						
MOVEA	imm16,reg1,reg2	rrrrr110001RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+sign-extend(imm16)	1	1	1						
MOVHI	imm16,reg1,reg2	rrrrr110010RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1]+(imm16 ll 0 ¹⁶)	1	1	1						
★ MUL ^{Note 22}	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100000	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]	1	2	2						
	imm9,reg2,reg3	rrrrr111111iiii wwwww01001111100	GR[reg3] ll GR[reg2]←GR[reg2]xsign-extend(imm9)	1	2	2						
MULH	reg1,reg2	rrrrr000111RRRRR	GR[reg2]←GR[reg2] ^{Note 6} 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 ^{Note 22}	reg1,reg2,reg3	rrrrr111111RRRRR wwwww01000100010	GR[reg3] ll GR[reg2]←GR[reg2]xGR[reg1]	1	2	2						
	imm9,reg2,reg3	rrrrr111111iiii wwwww0100111110	GR[reg3] ll GR[reg2]←GR[reg2]xzero-extend(imm9)	1	2	2						
NOP		000000000000000	Pass at least one clock cycle doing nothing.	1	1	1						
NOT	reg1,reg2	rrrrr000001RRRRR	GR[reg2]←NOT(GR[reg1])	1	1	1		0	x	x		
NOT1	bit#3,disp16[reg1]	01bbb111110RRRRR dddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not(Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,Z flag)	3	3	3					x	
	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					x	

APPENDIX B 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 Note4	n+1 Note4	n+1 Note4					
	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 Note4 Note17	n+2 Note4 Note17	n+2 Note4 Note17					
RETI		000001111100000 000000010100000	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	rrrrr11111RRRRR 000000010100000	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	rrrrr111110cccc 000000100000000	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	rrrrr111110cccc 000000000000000	If conditions are satisfied then GR[reg2]←00000001H else GR[reg2]←00000000H	1	1	1					

APPENDIX B 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]	00bbb111110RRRRR ddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Z flag←Not (Load-memory-bit(adr,bit#3)) Store-memory-bit(adr,bit#3,1)	3 Note 3	3 Note 3	3 Note 3					x	
	reg2,[reg1]	rrrrr111111RRRRR 0000000011100000	adr←GR[reg1] Z flag←Not(Load-memory-bit(adr,reg2)) Store-memory-bit(adr,reg2,1)	3 Note 3	3 Note 3	3 Note 3					x	
SHL	reg1,reg2	rrrrr111111RRRRR 0000000011000000	GR[reg2]←GR[reg2] logically shift left by GR[reg1]	1	1	1	x	0	x	x		
	imm5,reg2	rrrrr010110iiii	GR[reg2]←GR[reg2] logically shift left by zero-extend(imm5)	1	1	1	x	0	x	x		
SHR	reg1,reg2	rrrrr111111RRRRR 0000000010000000	GR[reg2]←GR[reg2] logically shift right by GR[reg1]	1	1	1	x	0	x	x		
	imm5,reg2	rrrrr010100iiii	GR[reg2]←GR[reg2] logically shift right by zero-extend(imm5)	1	1	1	x	0	x	x		
SLD.B	disp7[ep],reg2	rrrrr0110dddddd	adr←ep+zero-extend(disp7) GR[reg2]←sign-extend(Load-memory(adr,Byte))	1	1	Note 9						
SLD.BU	disp4[ep],reg2	rrrrr0000110ddd 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,Halfword))	1	1	Note 9						
SLD.HU	disp5[ep],reg2	rrrrr0000111ddd Notes 18, 20	adr←ep+zero-extend(disp5) GR[reg2]←zero-extend(Load-memory(adr,Halfword))	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]	rrrrr01111dddddd	adr←ep+zero-extend(disp7) Store-memory(adr,GR[reg2],Byte)	1	1	1						
SST.H	reg2,disp8[ep]	rrrrr10011dddddd Note 19	adr←ep+zero-extend(disp8) Store-memory(adr,GR[reg2],Halfword)	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 ddddddddddddddd	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr,GR[reg2],Byte)	1	1	1						
ST.H	reg2,disp16[reg1]	rrrrr111011RRRRR ddddddddddddddd0 Note 8	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr,GR[reg2],Halfword)	1	1	1						
ST.W	reg2,disp16[reg1]	rrrrr111011RRRRR ddddddddddddddd1 Note 8	adr←GR[reg1]+sign-extend(disp16) Store-memory(adr,GR[reg2],Word)	1	1	1						
STSR	regID,reg2	rrrrr111111RRRRR 0000000010000000	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,Halfword)) logically shift left by 1	5	5	5					
SXB	reg1	00000000101RRRRR	GR[reg1]←sign-extend (GR[reg1] (7 : 0))	1	1	1					
SXH	reg1	00000000111RRRRR	GR[reg1]←sign-extend (GR[reg1] (15 : 0))	1	1	1					
TRAP	vector	000001111111iiii 0000000100000000	EIPC ←PC+4 (Return PC) EIPSW ←PSW ECR.EICC ←Exception code (40H to 4FH, 50H to 5FH) PSW.EP ←1 PSW.ID ←1 PC ←00000040H (when vector is 00H to 0FH (exception code: 40H to 4FH)) 00000050H (when vector is 10H to 1FH (exception code: 50H to 5FH))	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	Note3	Note3	Note3		×
	reg2, [reg1]	rrrrr111111RRRRR 0000000011100110	adr←GR[reg1] Z flag←Not (Load-memory-bit (adr,reg2))	3	3	3	Note3	Note3	Note3		×
XOR	reg1,reg2	rrrrr001001RRRRR	GR[reg2]←GR[reg2] XOR GR[reg1]	1	1	1		0	×	×	
XORI	imm16,reg1,reg2	rrrrr110101RRRRR iiiiiiiiiiiiiiii	GR[reg2]←GR[reg1] XOR zero-extend (imm16)	1	1	1		0	×	×	
ZXB	reg1	00000000100RRRRR	GR[reg1]←zero-extend (GR[reg1] (7 : 0))	1	1	1					
ZXH	reg1	00000000110RRRRR	GR[reg1]←zero-extend (GR[reg1] (15 : 0))	1	1	1					

- Notes**
1. ddddddd: 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 total number of list12 registers. If n = 0, same operation as when n = 1)
 5. RRRRR: other than 00000.
 6. The lower halfword data only are valid.
 7. ddddddddddddddddddd: The higher 21 bits of disp22.
 8. ddddddddddddddd: 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: Higher 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. ddddd: Higher 6 bits of disp8.

★ **22.** Do not make a combination that satisfies all the following conditions when using the “MUL reg1, reg2, reg3” instruction and “MULU reg1, reg2, reg3” instruction. Operation is not guaranteed when an instruction that satisfies the following conditions is executed.

- Reg1 = reg3
- Reg1 ≠ reg2
- Reg1 ≠ r0
- Reg3 ≠ r0

APPENDIX C REVISION HISTORY

The following table shows the revision history up to this edition. The “Applied to:” column indicates the chapters of each edition in which the revision was applied.

(1/5)

Edition	Major Revision from Previous Edition	Applied to:
2nd	<ul style="list-style-type: none"> • Addition of the following product. μPD703111GM-15-UEU • Change of the following register name. PFCAL → PFCALL 	Throughout
	Addition of execution time of 150 MHz products to minimum instruction execution time in 1.2 Features	CHAPTER 1 INTRODUCTION
	Addition of Note in 2.1 (2) Non-port pins	CHAPTER 2 PIN FUNCTIONS
	Addition of description in 2.3 (1) (b) (vii) UCLK (USB clock)	
	Modification of description in 2.3 (3) (b) (ii) DMAAK0, DMAAK1 (DMA acknowledge)	
	Modification of description in 2.3 (5) (b) (ii) DMAAK2, DMAAK3 (DMA acknowledge)	
	Addition of execution time of 150 MHz products to minimum instruction execution time in 3.1 Features	CHAPTER 3 CPU FUNCTION
	Modification of default value in 3.2.1 (2) Program counter (PC)	
	Modification of description in 3.4.7 Peripheral I/O registers	
	Change of table of VSWC setting values in 3.4.9 System wait control register (VSWC)	
	Modification of description in 4.2.1 Pin status during internal instruction RAM, internal data RAM, and peripheral I/O access	CHAPTER 4 BUS CONTROL FUNCTION
	Modification of description when BTn1 and BTn0 bits are set to 11 in 4.4.1 (1) Bus cycle type configuration registers 0, 1 (BCT0, BCT1)	
	Modification of table of number of access clocks in 4.5.1 Number of access clocks	
	Addition to Cautions and modification of description in 4.5.6 (1) Line buffer control registers 0, 1 (LBC0, LBC1)	
	Addition of Remark in 4.5.6 (1) (a) Speculative read function (read buffer function)	
	Modification of description, addition of Note , and addition to Cautions in 4.5.6 (1) (b) Write buffer function	
	Addition to Cautions in 4.7.1 (3) Bus cycle period control register (BCP)	
	Modification of Cautions in 4.9.1 (1) Cache configuration register (BHC)	
	Addition of (5) in 4.10.3 Cautions	
	Addition of timing and modification of Notes in 4.11.6 (1) SDRAM (when read, latency = 2, no idle state insertion)	
	Addition of timing and modification of Notes in 4.11.6 (2) SDRAM (when read, latency = 2, two idle states inserted, 32-bit bus width)	
	Addition of timing and modification of Notes in 4.11.6 (3) SDRAM (when written)	
	Addition of 4.14 Timing at Which T0 State Is Not Inserted	
	Addition of timing and modification of Notes in Figure 5-9 SDRAM Single Read Cycle	CHAPTER 5 MEMORY ACCESS CONTROL FUNCTION
	Addition of timing and modification of Notes in Figure 5-10 SDRAM Single Write Cycle	

Edition	Major Revision from Previous Edition	Applied to:
2nd	Addition of timing and modification of Notes in Figure 5-11 SDRAM Access Timing	CHAPTER 5 MEMORY ACCESS CONTROL FUNCTION
	Modification of Caution in 5.3.6 (1) SDRAM refresh control registers 1, 3, 4, 6 (RFS1, RFS3, RFS4, RFS6)	
	Modification of description in Table 5-1 Example of Interval Factor Settings	
	Addition of internal instruction RAM in block diagram in 6.2 Configuration	CHAPTER 6 DMA FUNCTIONS (DMA CONTROLLER)
	Addition to Cautions in 6.3.1 (1) DMA source address registers 0H to 3H (DSA0H to DSA3H)	
	Addition to Cautions in 6.3.2 (1) DMA destination address registers 0H to 3H (DDA0H to DDA3H)	
	Addition to Cautions in 6.3.3 DMA transfer count registers 0 to 3 (DBC0 to DBC3)	
	Addition to Cautions in 6.3.4 DMA addressing control registers 0 to 3 (DADC0 to DADC3)	
	Addition to Cautions and description in 6.3.5 DMA channel control registers 0 to 3 (DCHC0 to DCHC3)	
	Addition to Cautions and description in 6.3.6 DMA terminal count output control register (DTC)	
	Addition to Cautions in 6.3.7 DMA trigger factor registers 0 to 3 (DTFR0 to DTFR3)	
	Addition of 6.3.7 (1) DMA request detection function	
	Addition of Caution and Note and modification of description in 6.3.8 DMA interface control register (DIFC)	
	Addition of timing and modification of Notes in Figure 6-12 Timing of 2-Cycle DMA Transfer (SDRAM → SRAM)	
	Addition of 6.5.1 (1) Timing of $\overline{\text{DMARQn}}$ and $\overline{\text{DMAAKn}}$ signals for 2-cycle transfer	
	Addition of 6.5.1 (2) $\overline{\text{DMAAKn}}$ signal active width extension function	
	Addition of 6.5.1 (3) Outline of 2-cycle transfer timing	
	Modification of timing in Figure 6-19 Timing of DMA Flyby Transfer (External I/O → SDRAM)	
	Modification of timing in Figure 6-23 Timing of DMA Flyby Transfer (SDRAM → External I/O)	
	Modification of timing in Figure 6-24 Timing of DMA Flyby Transfer (External I/O → SDRAM)	
Modification of description and addition to Caution in Table 6-5 Relationship Between Transfer Type and Transfer Object		
Addition of description in 6.8 Next Address Setting Function		
Addition of Cautions in 6.9 DMA Transfer Start Factors		
Modification of description in 6.13 Times Related to DMA Transfer		
Modification of description in 6.15 (3) Bus arbitration for CPU		
Addition of 6.15 (7) Read values of DSA_n and DDA_n registers		
Modification of Note in Figure 7-14 Pipeline Operation at Interrupt Request Acknowledgment (Outline)	CHAPTER 7 INTERRUPT/EXCEPTION PROCESSING FUNCTION	

Edition	Major Revision from Previous Edition	Applied to:		
2nd	Modification of selection of SSCG output by PLLSEL pin and MDL-Selector Table (modulation period) in 8.1 Features	CHAPTER 8 CLOCK GENERATION FUNCTION		
	Addition to Cautions and modification of description in 8.3.1 Clock control register (CKC)			
	Modification of sample coding <2> for data setting sequence of clock source select register (CKS) in 8.3.2 Clock source select register (CKS)			
	Modification of description in 8.3.3 SSCG control register (SSCGC)			
	Addition of Caution in 8.3.4 USB clock control register (UCKC)			
	Modification of oscillation stabilization time in 8.3.6 Oscillation stabilization time select register (OSTS)			
	Addition to Notes in Table 8-1 Operation Status of Each Clock			
	Modification of description in Table 8-2 Frequency List			
	Addition of 8.5 Operating Clock Provisions			
	Modification of oscillation stabilization time in Table 8-11 Counting Time Examples			
2nd	Addition to Caution in 9.1.5 (2) Timer mode control registers C01 to C51 (TMCC01 to TMCC51)	CHAPTER 9 TIMER/COUNTER FUNCTION (REAL-TIME PULSE UNIT)		
	Modification of Figure 9-7 TMC1 Compare Operation Example (Set/Reset Output Mode)			
	Addition of noise elimination width when $f_x = 150$ MHz in Table 9-6 Relationship Between NCW1n Register Set Value and Noise Elimination Width			
	Addition of 9.3.7 (6) Overflow interrupt signal (INTOV1n) and underflow interrupt signal (INTUD1n)			
2nd	Modification of transfer rate in 10.2.1 Features	CHAPTER 10 SERIAL INTERFACE FUNCTION		
	Modification of description in 10.2.2 (10) UARTBn receive data register AP (UBnRXAP), UARTBn receive data register (UBnRX) (n = 0, 1)			
	Addition of description in 10.2.2 (12) UARTBn transmit data register n (UBnTX) (n = 0, 1)			
	Modification of Caution in 10.2.3 (3) UARTBn control register 2 (UBnCTL2) (n = 0, 1)			
	Addition of description in 10.2.3 (4) UARTBn transmit data register (UBnTX) (n = 0, 1)			
	Modification of description in 10.2.3 (5) UARTBn receive data register AP (UBnRXAP), UARTBn receive data register (UBnRX) (n = 0, 1)			
	Addition and modification of description in 10.2.3 (6) UARTBn FIFO control register 0 (UBnFIC0) (n = 0, 1)			
	Modification of description in 10.2.3 (7) UARTBn FIFO control register 1 (UBnFIC1) (n = 0, 1)			
	Addition of description in 10.2.4 (5) (b) FIFO mode			
	Addition of description in 10.2.5 (2) Pending mode/pointer mode			
	Addition of Note in 10.2.5 (2) (a) (i) During transmission (writing to transmit FIFO)			
	Addition of Note in 10.2.5 (2) (a) (ii) During reception (reading from receive FIFO)			
	2nd		Addition of description in 10.2.6 (4) (c) (ii) Reception timeout interrupt (UBTITOn) (in FIFO mode only)	CHAPTER 10 SERIAL INTERFACE FUNCTION
			Addition of value when $f_x = 150$ MHz in Table 10-4 Baud Rate Generator Setting Data	

Edition	Major Revision from Previous Edition	Applied to:
2nd	Addition of 10.2.8 Control flow	CHAPTER 10 SERIAL INTERFACE FUNCTION
	Modification of description in Figure 10-22 Block Diagram of Clocked Serial Interfaces 30 and 31	
	Modification of description of Caution 2 in 10.3.3 (1) Clocked serial interface mode registers 30, 31 (CSIM30, CSIM31)	
	Modification of description in 10.3.3 (2) Clocked serial interface clock select registers 30, 31 (CSIC30, CSIC31)	
	Addition of description in 10.3.3 (3) Receive data buffer registers 30, 31 (SIRB30, SIRB31)	
	Addition of description in 10.3.3 (4) Transmit data CSI buffer registers 30, 31 (SFDB30, SFDB31)	
	Modification of description in 10.3.3 (5) CSIBUF status registers 30, 31 (SFA30, SFA31)	
	Modification of example in Caution 2 in 10.3.4 (2) Baud rate	
	Modification of description in Figure 10-25 Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: MSB First (DIRn Bit = 0 in CSIM3n Register)	
	Modification of description in Figure 10-26 Transfer Data Length: 8 Bits (CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register), Transfer Direction: LSB First (DIRn Bit = 1 in CSIM3n Register)	
	Deletion of description of 10.3.5 (7) Slave mode	
	Modification of Figure 10-30 Slave Mode (CKPn and DAPn Bits = 00 in CSIC3n Register, CCLn3 to CCLn0 Bits = 1000 in CSIL3n Register (Transfer Data Length: 8 Bits))	
	Modification of description in Figure 10-32 Continuous Mode	
	Modification of description in 10.3.5 (11) Transmission mode	
	Addition of description in 10.3.5 (12) Reception mode	
	Deletion of description in 10.3.5 (16) (a) SCKn pin	
	Modification of description in Table 10-8 Default Output Level of SCKn Pin	
	Modification of description of (1) to (12) in 10.3.6 Usage	
	Addition of Caution in 11.1 Overview	CHAPTER 11 USB FUNCTION CONTROLLER (USBF)
	Addition of items in Table 11-2 Correspondence Between Requests and Decoded Values	
	Addition of description in 11.4.1 (3) UF0 EPNACK register (UF0EN)	
	Deletion of description in 11.4.1 (9) UF0 EP status 1 register (UF0EPS1)	
	Modification of description in 11.4.1 (11) UF0 INT status 0 register (UF0IS0)	
	Modification of description in 11.4.1 (34) UF0 mode status register (UF0MODS)	
	Modification of description in 11.4.3 (2) UF0 EP0 status register L (UF0E0SL)	
	Modification of description in 11.4.3 (9) UF0 address register (UF0ADRS)	
	Modification of description in 11.4.3 (10) UF0 configuration register (UF0CNF)	
	Modification of description in 11.4.3 (11) UF0 interface 0 register (UF0IF0)	
	Modification of description in 11.4.3 (12) UF0 interface 1 to 4 registers (UF0IF1 to UF0IF4)	

Edition	Major Revision from Previous Edition	Applied to:
2nd	Modification of Caution 2 in 11.4.3 (14) UF0 device descriptor registers 0 to 17 (UF0DD0 to UF0DD17)	CHAPTER 11 USB FUNCTION CONTROLLER (USBF)
	Modification of Caution 2 in 11.4.3 (15) UF0 configuration/interface/endpoint descriptor registers 0 to 255 (UF0CIE0 to UF0CIE255)	
	Modification of Caution in 11.4.4 (1) USB function 0 DMA channel select register (UF0CS)	
	Deletion of Caution in Table 11-8 FW-Supported Standard Requests	
	Modification of description in Figure 11-15 Automatically Processed Requests for Control Transfer	
	Modification of description in Figure 11-20 CPUDEC Request for Control Transfer	
	Modification of description in Figure 11-30 USB Connection Example	
	Addition of value when $f_x = 150$ MHz in Table 12-1 Setting of A/D Conversion Operation Time	CHAPTER 12 A/D CONVERTER
	Addition of 12.9 How to Read A/D Converter Characteristics Table	
	Modification of repeat frequency in 13.1 Features	CHAPTER 13 PWM UNIT
	Modification of description in Figure 13-1 Block Diagram of PWM Unit	
	Modification of description in 13.3 (1) PWM control registers 0 and 1 (PWMC0 and PWMC1)	
	Modification of description in 13.4.2 (1) Setting for starting PWM operation	
	Modification of description in Table 13-1 Repeat Cycle of PWMn	
	Modification of Caution in 14.3.8 Port DH	CHAPTER 14 PORT FUNCTIONS
	Modification of Caution and description on bit 0 in 14.3.8 (2) (c) Port DH function control register (PFCDH)	
	Addition to Caution in 14.3.10 (2) (c) Port CT function control register (PFCCT)	
	Addition of noise elimination width when $f_x = 150$ MHz in Table 14-4 Relationship Between NCW1n Register Set Value and Noise Elimination Width	
	Modification of value of program counter (PC) after reset in Table 15-2 Initial Value of CPU, Internal Data RAM, Internal Instruction RAM, and On-Chip Peripheral I/O After Reset	CHAPTER 15 RESET FUNCTIONS
	Modification of description in 16.1.1 (7) Mask function	CHAPTER 16 DEBUG FUNCTION (DCU)
	Addition of CHAPTER 17 ELECTRICAL SPECIFICATIONS (TARGET VALUES)	CHAPTER 17 ELECTRICAL SPECIFICATIONS
	Addition of CHAPTER 18 PACKAGE DRAWING	CHAPTER 18 PACKAGE DRAWING
	Addition of CHAPTER 19 RECOMMENDED SOLDERING CONDITIONS	CHAPTER 19 RECOMMENDED SOLDERING CONDITIONS
	Addition to Note in B.2 Instruction Set (in Alphabetical Order)	APPENDIX B INSTRUCTION SET LIST
Addition of APPENDIX C REVISION HISTORY	APPENDIX C REVISION HISTORY	