

NEC

Preliminary User's Manual

NU85E

32-Bit Microprocessor Core

Hardware

NU85E

NU85EA

Document No. A14874EJ3V0UM00 (3rd edition)

Date Published March 2002 N CP(N)

© NEC Corporation 2000
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 export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

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

M5D 98.12

Regional Information

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

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

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

NEC Electronics Inc. (U.S.)

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

NEC Electronics (Europe) GmbH

Duesseldorf, Germany
Tel: 0211-65 03 01
Fax: 0211-65 03 327

- Branch The Netherlands
Eindhoven, The Netherlands
Tel: 040-244 58 45
Fax: 040-244 45 80

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

NEC Electronics (France) S.A.

Vélizy-Villacoublay, France
Tel: 01-3067-58-00
Fax: 01-3067-58-99

NEC Electronics (France) S.A. Representación en España

Madrid, Spain
Tel: 091-504-27-87
Fax: 091-504-28-60

NEC Electronics Italiana S.R.L.

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

NEC Electronics (UK) Ltd.

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

NEC Electronics Hong Kong Ltd.

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

NEC Electronics Hong Kong Ltd.

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

NEC Electronics Singapore Pte. Ltd.

Novena Square, Singapore
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.

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

NEC do Brasil S.A.

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

J01.12

Major Revisions in This Edition

Pages	Description
p.30	Addition of description in 2.2.2 (3) VAPREQ
p.33	Addition of description in 2.2.2 (17) VMLAST, VSLAST
p.33	Addition of description in 2.2.2 (18) VMAHLD, VSAHLD
p.34	Addition of description in 2.2.2 (20) VBDC
p.34	Addition of description in 2.2.2 (21) VBDV
pp.34, 35	Addition of description in 2.2.3 (1) DCRESZ Addition of Figure 2-2
pp.46, 47	Addition of Notes in 2.3 Recommended Connection of Unused Pins
p.50	Status of DBO12 to DBO5 pins after reset modified in Table 2-10 Pin Status in Each Operating Mode
p.135	Addition of 5.5 Precautions
pp.138, 139	Addition of Remarks 4 and 5 in 6.2.1 Power save control register (PSC)
p.142	Addition of Remark in Table 6-3 Operation After Setting Software STOP Mode in Interrupt Servicing Routine
p.145	Addition of <4> and Remark in 6.6 (1) (b) When canceling software STOP mode
p.147	Addition of Remark in 6.6 (2) (b) When canceling hardware STOP mode
p.157	Addition of Caution in Figure 7-6 DMA Addressing Control Registers 0 to 3 (DADC0 to DADC3)
p.159	Addition of Caution and descriptions in Figure 7-7 DMA Channel Control Registers 0 to 3 (DCHC0 to DCHC3)
p.172	Addition of descriptions in 7.8.5 One-time transfer when executing single transfers using DMARQn signal
p.173	Addition of Figure 7-24 Example of Two-Cycle Transfer
p.174	Addition of descriptions in 7.9.2 Flyby transfer
p.174	Addition of Figure 7-25 Example of Flyby Transfer (Memory to I/O)
p.176	Addition of Figure 7-27 Example of Terminal Count Signal Output (DMTCO3 to DMTCO0)
p.179	Modification of Remark in Figure 7-29 DMA Transfer Forcible Termination Example
p.204	Modification and addition of descriptions in 7.15 (3) Intervals related to DMA transfer
p.205	Addition of descriptions in 7.15 (4) CPU access during DMA transfer
p.205	Addition of 7.15 (6) DMARQn signal retention and (7) VMLOCK signal
p.209	Modification of Caution 1 and addition of Caution 2 in 8.2 Non-Maskable Interrupts (NMI)
p.232	Modification of Figure 9-1 Peripheral Macro Connection Example
p.233	Deletion of When NPB Peripheral Is Connected and modification in 9.4 (2) Test mode pins
p.254	Addition of APPENDIX C REVISION HISTORY

The mark ★ shows major revised points.

PREFACE

Target Readers This manual is intended for users who wish to understand the hardware functions of the NU85E and NU85EA, which are the CPU cores of a cell-based IC (CBIC), to design application systems using the NU85E or NU85EA.

Purpose This manual is designed to help users understand the hardware functions of the NU85E and NU85EA outlined in Organization below.

Organization This manual describes the hardware functions of the NU85E and NU85EA. For details about the architecture and instruction functions, refer to the “V850E1 User’s Manual Architecture.” The organization of each manual is as follows:

**NU85E User’s Manual
Hardware (This manual)**

- Overview
- CPU function
- Peripheral I/O functions
- Test functions

**V850E1 User’s Manual
Architecture**

- 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 of electricity, logic circuits, and microcontrollers.

To gain a general understanding of the hardware functions of the NU85E and NU85EA
→ Read this manual according to the **CONTENTS**.

To confirm details of a function, etc. when the name is known
→ Refer to **APPENDIX B INDEX**.

To learn about the details of an instruction function
→ Refer to the **V850E1 Architecture User’s Manual (U14559E)**.

This document describes the NU85E as the representative product. When using the NU85EA, read NU85E as NU85EA.

Conventions

Data significance:	Higher digits on the left and lower digits on the right
Active low representation:	xxxZ (Z after pin or signal name)
Note:	Footnote for item marked with Note in the text
Caution:	Information requiring particular attention
Remark:	Supplementary information
Numerical representation:	Binary ... xxxx or xxxxB Decimal ... xxxx Hexadecimal ... xxxxH
Prefix indicating the power of 2 (address space, memory capacity):	K (kilo) ... $2^{10} = 1,024$ M (mega) ... $2^{20} = 1,024^2$ G (giga) ... $2^{30} = 1,024^3$
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.

- V850E1 Architecture User's Manual (U14559E)
- Memory Controller NU85E, NU85ET User's Manual (A15019E)
- Instruction Cache, Data Cache NU85E, NU85ET User's Manual (A15241E)
- CB-10 Family VX Type NU85E, NU85ET Design Manual (A15401E)
- CB-10 Family VX Type Core Library CPU Core, Peripheral Design Manual (A15133E)

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

CONTENTS

CHAPTER 1 INTRODUCTION	17
1.1 Outline	17
1.2 Application System Example	18
1.3 Features	19
1.4 Symbol Diagram	21
1.5 Function Blocks	22
1.5.1 Internal block diagram	22
1.5.2 Internal units	23
1.6 Functional Differences Between NU85E and NB85E	24
CHAPTER 2 PIN FUNCTIONS	25
2.1 List of Pin Functions	25
2.2 Explanation of Pin Functions	29
2.2.1 NPB pins	29
2.2.2 VSB pins	30
2.2.3 System control pins	34
2.2.4 DMAC pins.....	36
2.2.5 INTC pins.....	36
2.2.6 VFB pins	36
2.2.7 VDB pins.....	37
2.2.8 Instruction cache pins.....	38
2.2.9 Data cache pins	39
2.2.10 RCU pins	41
2.2.11 Peripheral evaluation chip mode pins.....	41
2.2.12 Operation mode setting pins.....	42
2.2.13 Test mode pins	44
2.3 Recommended Connection of Unused Pins	46
2.4 Pin Status	48
CHAPTER 3 CPU	51
3.1 Features	51
3.2 Registers	52
3.2.1 Program registers	53
3.2.2 System registers	55
3.3 Address Space	58
3.3.1 Program area.....	59
3.3.2 Data area	60
3.4 Areas	62
3.4.1 ROM area	62
3.4.2 RAM area.....	64
3.4.3 Peripheral I/O area	65
3.4.4 External memory area.....	67
3.5 Peripheral I/O Registers	67
3.5.1 NU85E control registers.....	68

3.5.2	Memory controller (MEMC) control registers	71
3.5.3	Instruction cache control registers	72
3.5.4	Data cache control registers	72
3.6	RCU Interface	73
3.6.1	Outline.....	73
3.6.2	On-chip debugging.....	73
CHAPTER 4	BCU.....	74
4.1	Features	74
4.2	Memory Banks	74
4.3	Programmable Chip Select Function	77
4.4	Programmable Peripheral I/O Area Selection Function	83
4.5	Bus Size Setting Function.....	86
4.6	Endian Setting Function.....	87
4.6.1	Endian configuration register (BEC).....	87
4.6.2	Usage restrictions concerning big endian format with NEC development tools.....	88
4.7	Cache Configuration.....	90
4.8	BCU-Related Register Setting Examples	91
4.9	Data Transfer Using VSB.....	94
4.9.1	Data transfer example.....	94
4.9.2	Control signals output by bus master.....	95
4.9.3	Read/write timing	98
4.9.4	VSB read/write timing example	111
4.9.5	Reset timing	113
4.9.6	Bus master transition	114
4.9.7	Misalign access timing	116
CHAPTER 5	BBR.....	118
5.1	Programmable Peripheral I/O Area.....	120
5.2	Wait Insertion Function	123
5.3	Retry Function.....	125
5.4	NPB Read/Write Timing.....	126
★ 5.5	Precautions	135
CHAPTER 6	STBC.....	136
6.1	Power Save Function.....	136
6.2	Control Registers	137
6.2.1	Power save control register (PSC).....	137
6.2.2	Command register (PRCMD)	139
6.3	HALT Mode	140
6.4	Software STOP Mode.....	141
6.5	Hardware STOP Mode.....	143
6.6	Clock Control in Software/Hardware STOP Mode.....	144
CHAPTER 7	DMAC.....	149
7.1	Features	149
7.2	Configuration	150
7.3	Transfer Objects	151

7.4	DMA Channel Priorities	151
7.5	Control Registers	152
7.5.1	DMA source address registers 0 to 3 (DSA0 to DSA3)	152
7.5.2	DMA destination address registers 0 to 3 (DDA0 to DDA3)	154
7.5.3	DMA transfer count registers 0 to 3 (DBC0 to DBC3).....	156
7.5.4	DMA addressing control registers 0 to 3 (DADC0 to DADC3)	157
7.5.5	DMA channel control registers 0 to 3 (DCHC0 to DCHC3).....	159
7.5.6	DMA disable status register (DDIS).....	160
7.5.7	DMA restart register (DRST).....	161
7.6	Next Address Setting Function	162
7.7	DMA Bus State	163
7.7.1	Bus state types	163
7.7.2	DMAC bus cycle state transitions	165
7.8	Transfer Modes	166
7.8.1	Single transfer mode.....	166
7.8.2	Single-step transfer mode.....	168
7.8.3	Line transfer mode.....	169
7.8.4	Block transfer mode.....	171
7.8.5	One-time transfer when executing single transfers using DMARQn signal	172
7.9	Transfer Types	173
7.9.1	Two-cycle transfer	173
7.9.2	Flyby transfer	174
7.10	DMA Transfer Start Factors	175
7.11	Terminal Count Output When DMA Transfer Is Complete	176
7.12	Forcible Interruption	177
7.13	Forcible Termination	178
7.14	DMA Transfer Timing Examples	180
7.15	Precautions	204
CHAPTER 8 INTC		206
8.1	Features	206
8.2	Non-Maskable Interrupts (NMI)	209
8.2.1	Operation	212
8.2.2	Restore	213
8.3	Maskable Interrupts	214
8.3.1	Operation	214
8.3.2	Restore	216
8.3.3	Maskable interrupt priorities.....	217
8.3.4	Control registers	221
8.3.5	Maskable interrupt status flag (ID).....	224
8.4	Software Exception	225
8.4.1	Operation	225
8.4.2	Restore	226
8.5	Exception Trap	227
8.5.1	Illegal opcode.....	227
8.5.2	Operation	228
8.5.3	Restore	228
8.6	Interrupt Response Time	229

8.7	Periods When Interrupts Cannot Be Acknowledged	229
CHAPTER 9 TEST FUNCTION.....		230
9.1	Test Pins	230
9.1.1	Test bus pins (TBI39 to TBI0 and TBO34 to TBO0)	230
9.1.2	BUNRI and TEST pins	230
9.1.3	BUNRIOUT pin.....	231
9.2	List of Test Interface Signals	231
9.3	Example of Connection of Peripheral Macro in Test Mode.....	232
9.4	Handling of Each Pin in Test Mode	233
CHAPTER 10 NB85E901		234
10.1	Symbol Diagram.....	234
10.2	Pin Functions	235
10.2.1	Pin function list.....	235
10.2.2	Pin functions	236
10.2.3	Recommended connection of unused pins.....	239
10.2.4	Pin status	240
10.3	Debug Function.....	242
10.4	NU85E Connection Example.....	243
10.5	N-Wire Type IE Connection.....	244
10.5.1	IE connector (target system side)	244
10.5.2	Example of recommended circuit when connecting NB85E901 and NU85E.....	246
APPENDIX A ROM/RAM ACCESS TIMING.....		247
APPENDIX B INDEX.....		249
★ APPENDIX C REVISION HISTORY.....		254

LIST OF FIGURES (1/3)

Figure No.	Title	Page
2-1	Acknowledgement of DCRESZ Signal.....	34
2-2	Stopping VBCLK Oscillation by System Reset.....	35
3-1	List of CPU Registers	52
3-2	Program Counter (PC).....	54
3-3	Interrupt Source Register (ECR)	56
3-4	Program Status Word (PSW).....	57
3-5	Address Space	58
3-6	Program Area	59
3-7	Data Area (64 MB Mode).....	60
3-8	Data Area (256 MB Mode).....	61
3-9	ROM Area.....	62
3-10	RAM Area	64
3-11	Peripheral I/O Area.....	66
3-12	Connection of NU85E and N-Wire Type In-Circuit Emulator via RCU.....	73
4-1	Chip Area Select Control Register 0 (CSC0).....	77
4-2	Chip Area Select Control Register 1 (CSC1).....	78
4-3	CSC0 and CSC1 Register Setting Example (64 MB Mode)	79
4-4	CSC0 and CSC1 Register Setting Example (256 MB Mode)	82
4-5	Peripheral I/O Area and Programmable Peripheral I/O Area.....	84
4-6	Peripheral I/O Area Select Control Register (BPC)	85
4-7	Bus Size Configuration Register (BSC).....	86
4-8	Endian Configuration Register (BEC).....	87
4-9	Word Data Little Endian Format Example	88
4-10	Word Data Big Endian Format Example.....	88
4-11	Cache Configuration Register (BHC)	90
4-12	BPC, BSC, BEC, BHC Register Setting Example	91
4-13	Example of Data Transfer Using VSB	94
4-14	Read/Write Timing of Bus Slave Connected to VSB	99
4-15	VSB Timing Example.....	111
4-16	Reset Timing	113
4-17	Bus Master Transition Timing.....	115
4-18	Misalign Access Timing	116
5-1	NPB Connection Overview	118
5-2	NU85E and Peripheral Macro Connection Example.....	119
5-3	Peripheral I/O Area and Programmable Peripheral I/O Area.....	120
5-4	Peripheral I/O Area Select Control Register (BPC)	121
5-5	BPC Register Setting Example.....	122
5-6	NPB Strobe Wait Control Register (VSWC)	123
5-7	Retry Function	125
5-8	Halfword Access Timing	126
5-9	Timing of Byte Access to Odd Address	127
5-10	Timing of Byte Access to Even Address.....	127

LIST OF FIGURES (2/3)

Figure No.	Title	Page
5-11	Read Modify Write Timing	128
5-12	Retry Timing (Write).....	128
5-13	Retry Timing (Read)	129
5-14	Read/Write Timing of Bus Slave Connected to NPB	130
5-15	NPB Write Timing (Example of Timing of Data Write to CSC0 and CSC1 Registers).....	134
6-1	Power Save Function State Transition Diagram.....	136
6-2	Power Save Control Register (PSC).....	137
6-3	Command Register (PRCMD)	139
6-4	Connection of NU85E and Clock Control Circuit	144
6-5	Software STOP Mode Set/Cancel Timing Example.....	146
6-6	Hardware STOP Mode Set/Cancel Timing Example	148
7-1	DMA Source Address Registers 0H to 3H (DSA0H to DSA3H).....	152
7-2	DMA Source Address Registers 0L to 3L (DSA0L to DSA3L)	153
7-3	DMA Destination Address Registers 0H to 3H (DDA0H to DDA3H).....	154
7-4	DMA Destination Address Registers 0L to 3L (DDA0L to DDA3L)	155
7-5	DMA Transfer Count Registers 0 to 3 (DBC0 to DBC3)	156
7-6	DMA Addressing Control Registers 0 to 3 (DADC0 to DADC3).....	157
7-7	DMA Channel Control Registers 0 to 3 (DCHC0 to DCHC3).....	159
7-8	DMA Disable Status Register (DDIS)	160
7-9	DMA Restart Register (DRST).....	161
7-10	Buffer Register Configuration	162
7-11	DMAC Bus Cycle State Transition Diagram	165
7-12	Single Transfer Example 1	166
7-13	Single Transfer Example 2	166
7-14	Single Transfer Example 3	167
7-15	Single Transfer Example 4	167
7-16	Single-Step Transfer Example 1	168
7-17	Single-Step Transfer Example 2.....	168
7-18	Line Transfer Example 1.....	169
7-19	Line Transfer Example 2.....	169
7-20	Line Transfer Example 3.....	170
7-21	Line Transfer Example 4.....	170
7-22	Block Transfer Example.....	171
7-23	One-Time Transfer When Executing Single Transfers Using DMARQn Signal.....	172
7-24	Example of Two-Cycle Transfer	173
7-25	Example of Flyby Transfer (Memory to I/O).....	174
7-26	Timing Example of Terminal Count Signals (DMTCO3 to DMTCO0)	176
7-27	Example of Terminal Count Signal Output (DMTCO3 to DMTCO0)	176
7-28	DMA Transfer Forcible Interruption Example.....	177
7-29	DMA Transfer Forcible Termination Example.....	178
7-30	Example of Two-Cycle Single Transfer Timing (Between External SRAMs Connected to NT85E500).....	181
7-31	Example of Two-Cycle Single-Step Transfer Timing (Between External SRAMs Connected to NT85E500) ..	183
7-32	Example of Two-Cycle Line Transfer Timing (Between External SRAMs Connected to NT85E500).....	185

LIST OF FIGURES (3/3)

Figure No.	Title	Page
7-33	Example of Two-Cycle Block Transfer Timing (Between External SRAMs Connected to NT85E500).....	187
7-34	Example of Two-Cycle Single Transfer Timing (from RAM Connected to VDB to SDRAM Connected to NT85E502)	189
7-35	Example of Two-Cycle Single Transfer Timing (from SDRAM Connected to NT85E502 to RAM Connected to VDB)	191
7-36	Example of Flyby Single Transfer Timing (from External SRAM to External I/O Connected to NT85E500) ...	193
7-37	Example of Flyby Single-Step Transfer Timing (from External SRAM to External I/O Connected to NT85E500)	195
7-38	Example of Flyby Single-Step Transfer Timing (from External I/O to External SRAM Connected to NT85E500)	197
7-39	Example of Flyby Line Transfer Timing (from External SRAM to External I/O Connected to NT85E500).....	199
7-40	Example of Flyby Block Transfer Timing (from External SRAM to External I/O Connected to NT85E500).....	201
7-41	Example of Flyby Block Transfer Timing (from External I/O to External SRAM Connected to NT85E500).....	203
8-1	Example of Non-Maskable Interrupt Request Acknowledgement Operation.....	210
8-2	Non-Maskable Interrupt Processing Format	212
8-3	RETI Instruction Processing Format.....	213
8-4	Maskable Interrupt Processing Format.....	215
8-5	RETI Instruction Processing Format.....	216
8-6	Servicing Example in Which Another Interrupt Request Is Issued During Interrupt Servicing.....	218
8-7	Servicing Example for Simultaneously Issued Interrupt Requests	220
8-8	Interrupt Control Registers 0 to 63 (PIC0 to PIC63)	221
8-9	Interrupt Mask Registers 0 to 3 (IMR0 to IMR3)	222
8-10	In-Service Priority Register (ISPR)	223
8-11	Program Status Word (PSW).....	224
8-12	Software Exception Processing Format	225
8-13	RETI Instruction Processing Format.....	226
8-14	Illegal Opcode.....	227
8-15	Exception Trap Processing Format	228
8-16	Example of Pipeline Operation When Interrupt Request Is Acknowledged (Outline)	229
9-1	Peripheral Macro Connection Example	232
10-1	NB85E901 and NU85E Connection Example.....	243
10-2	N-Wire Type IE Connection	244
10-3	IE Connector Pin Layout Diagram (Target System Side)	244
10-4	Example of Recommended Circuit for IE Connection (NU85E + NB85E901)	246
A-1	ROM Access Timing.....	247
A-2	RAM Access Timing	248

LIST OF TABLES

Table No.	Title	Page
2-1	VMTTYP1 and VMTTYP0 Signals	30
2-2	VMBENZ3 to VMBENZ0 and VSBENZ1 Signals	31
2-3	VMSIZE1 and VMSIZE0 Signals	31
2-4	VMCTYP2 to VMCTYP0 Signals	32
2-5	VMSEQ2 to VMSEQ0 Signals	32
2-6	IRAMWR3 to IRAMWR0 Signals	37
2-7	IDDRRQ, IDDDRQ, IDSEQ4, and IDSEQ2 Signals	39
2-8	IFIRA64, IFIRA32, and IFIRA16 Signals.....	42
2-9	IFINSZ1 and IFINSZ0 Signals	43
2-10	Pin Status in Each Operating Mode.....	48
3-1	List of Program Registers	53
3-2	List of System Registers	55
3-3	Interrupt/Exception Table.....	63
3-4	RAM Area Size Settings	64
4-1	VMTTYP1 and VMTTYP0 Signals	95
4-2	VMCTYP2 to VMCTYP0 Signals	95
4-3	VMBENZ3 to VMBENZ0 Signals	96
4-4	VMSIZE1 and VMSIZE0 Signals	96
4-5	VMSEQ2 to VMSEQ0 Signals	96
4-6	VMWAIT, VMAHLD, and VMLAST Signals.....	97
4-7	VBDC and VBDV Signals	97
5-1	Setting of Setup Wait, VPSTB Wait Lengths at Each Operation Frequency	124
6-1	Operation After HALT Mode Is Canceled by Interrupt Request.....	140
6-2	Operation After Software STOP Mode Is Canceled by Interrupt Request.....	141
6-3	Operation After Setting Software STOP Mode in Interrupt Servicing Routine	142
6-4	Status After Cancellation of Hardware STOP Mode	143
7-1	Relationships Between Transfer Type and Transfer Object	151
7-2	Relationships Between Wait Function and Transfer Object	151
8-1	Interrupt/Exception List	206
9-1	List of Test Mode Settings	230
10-1	Pin Status in Each Operating Mode.....	240
10-2	IE Connector Pin Functions (Target System Side)	245

CHAPTER 1 INTRODUCTION

The NU85E Family consists of CPU cores that feature on-chip the “V850E1” 32-/16-bit RISC type CPU and peripheral I/Os, and are designed for embedding in ASICs. The V850E1 can execute almost all instructions in 1 clock through 5-stage pipeline control based on the RISC architecture. Furthermore, the NU85E Family also provides on-chip 2 types of external bus interfaces for connection to high- and low-speed peripheral I/Os, as well as functions to interface with ROM, RAM, an instruction cache, and a data cache. This product, the “NU85E”, is a CPU core that has, among other on-chip features, a DMA controller and an interrupt controller.

1.1 Outline

(1) “V850E1” CPU

The NU85E is equipped with the “V850E1”, which is a RISC type CPU that utilizes a five-stage pipeline technique. Two-byte basic instructions and instructions for high-level language support increase the efficiency of object code generated by the C compiler and reduce the program size.

In addition, to increase the speed of multiplication processing, the NU85E contains an on-chip high-speed hardware multiplier capable of executing 32-bit × 32-bit operations.

(2) Bus interfaces

The NU85E provides the following two types of bus interface for connection with peripheral macros or user logic.

- V850E system bus (VSB)
- NEC peripheral I/O bus (NPB)

The VSB, which is synchronized with the system clock, is the bus to be used for connection with high-speed peripheral macros such as a memory controller (MEMC) or macros operating as bus master (DMAC, DSP, etc.).

The NPB, which operates asynchronously with the system clock, is to be used for connection with relatively low-speed peripheral macros such as a timer or asynchronous serial interface (UART).

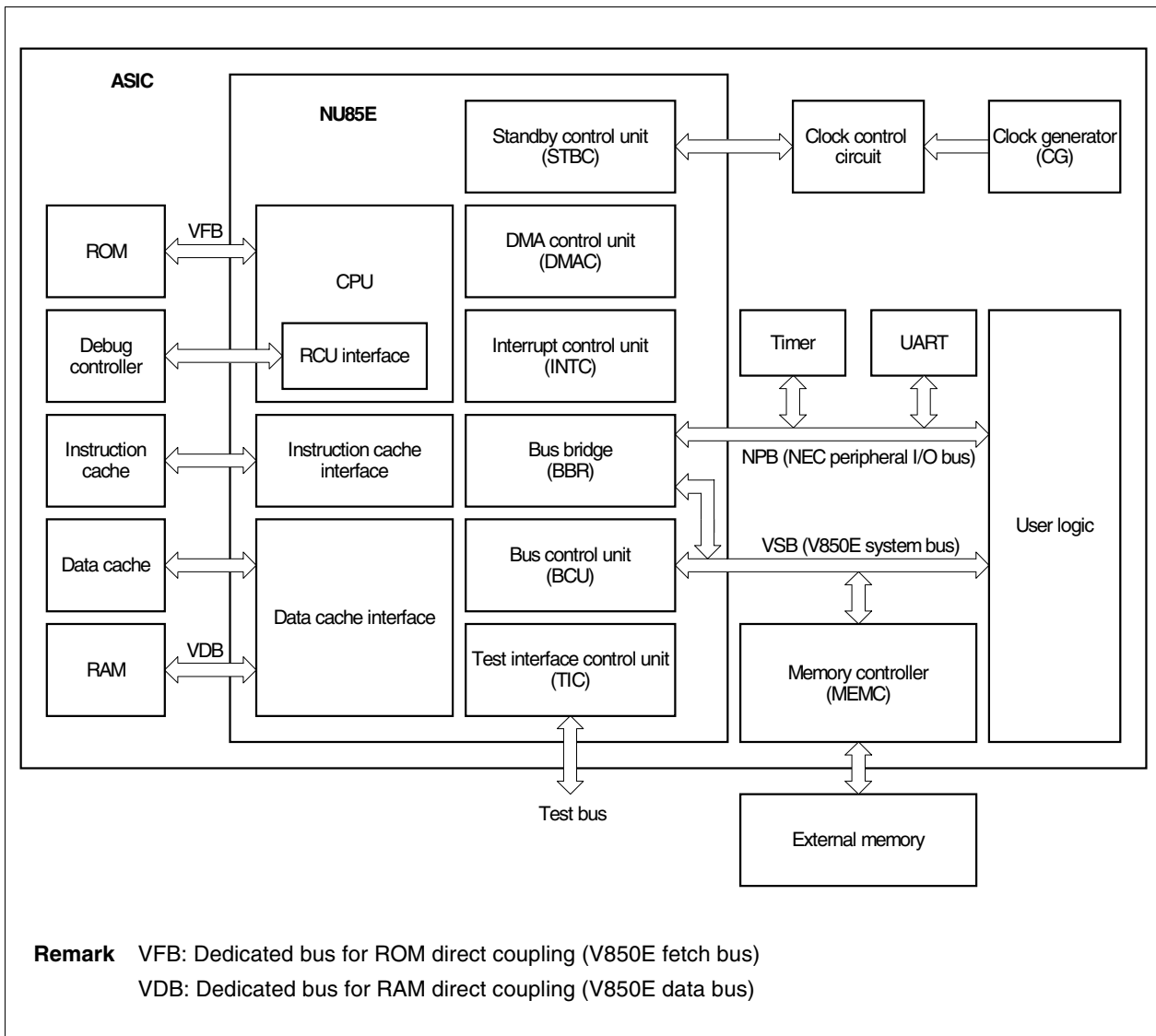
A V850E fetch bus (VFB), which can be directly coupled with ROM, and a V850E data bus (VDB), which can be directly coupled with RAM, are also provided.

In addition, since the NU85E contains on-chip special purpose interfaces for the instruction cache, data cache, and run control unit (RCU), each macro can be directly coupled.

(3) On-chip peripheral I/O

The NU85E contains an on-chip DMA control unit (DMAC) for controlling DMA transfers, an on-chip interrupt control unit (INTC) for controlling interrupt requests, and an on-chip standby control unit (STBC) for controlling the power save function.

1.2 Application System Example



Caution In this manual, representations related to the memory connected to the NU85E are unified as follows.

- **RAM: NU85E direct-coupled RAM (connected to VDB)**
- **ROM: NU85E direct-coupled ROM (connected to VFB)**
- **External memory: RAM or ROM connected via the memory controller (MEMC) (connected via VSB)**

1.3 Features

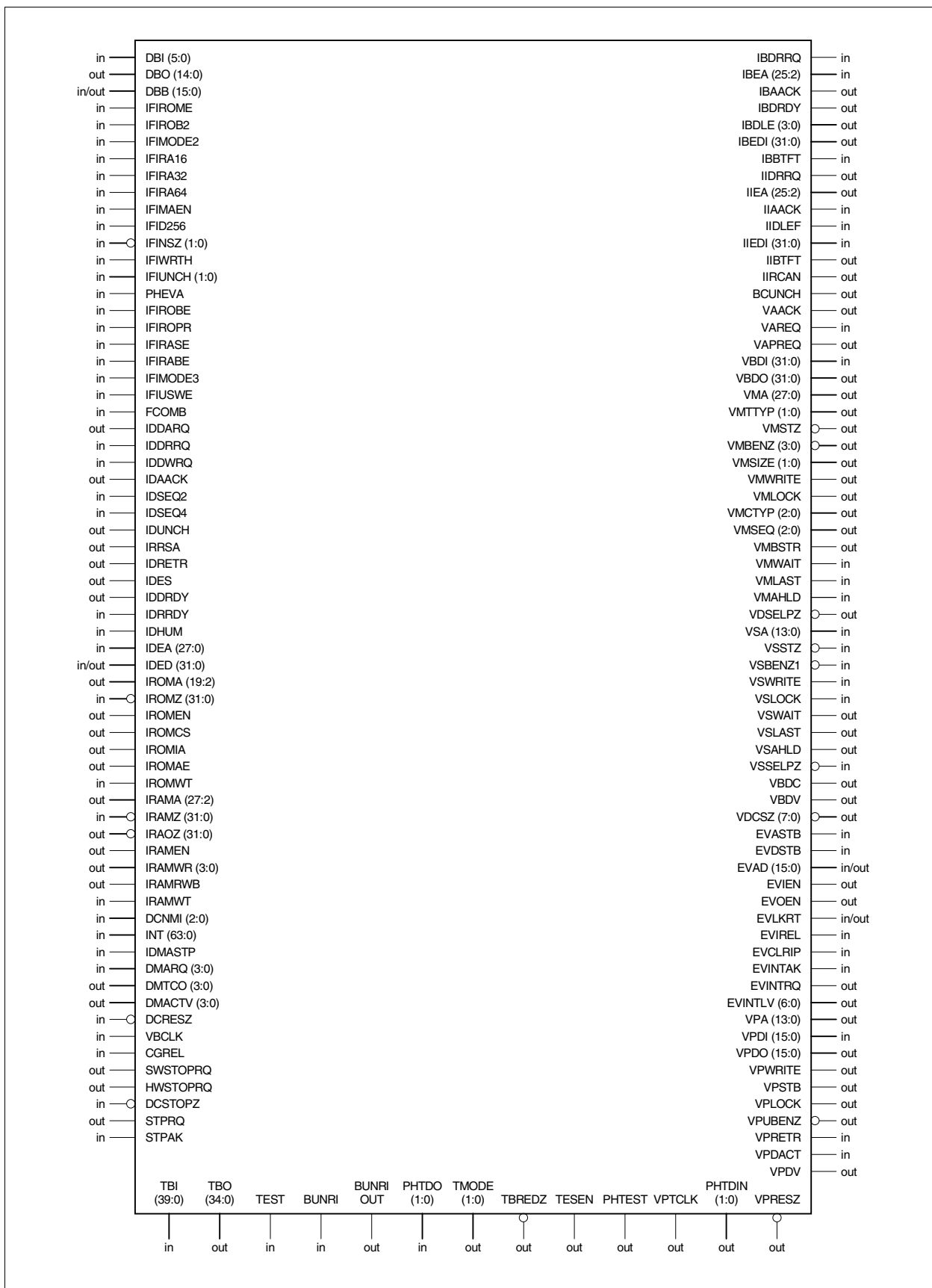
- Number of instructions 83
- General-purpose registers 32 bits × 32 registers
- Instruction set
 - Upwardly compatible with V850 CPU
 - Signed multiplication (32 bits × 32 bits → 64 bits)
 - Saturated calculation instructions (with overflow/underflow detection function)
 - 32-bit shift instructions: 1 clock
 - Bit manipulation instructions
 - Load/store instructions with long/short format
 - Signed load instructions
- Memory space
 - Program area: 64 MB linear address space
 - Data area: 4 GB linear address space
 - Memory bank division function: 2, 4, or 8 MB/bank
- External bus interface
 - VSB (V850E system bus)
 - Address/data separated bus (28-bit address^{Note}/32-bit data bus)
 - Data I/O separated bus
 - 32-/16-/8-bit bus sizing function
 - Bus hold function
 - External wait function
 - Endian switching function
 - NPB (NEC peripheral I/O bus)
 - Address/data separated bus (14-bit address/16-bit data bus)
 - Data I/O separated bus
 - Programmable wait function
 - Retry function

Note 14-bit address bus when functioning as bus slave
- Interrupt/exception control functions
 - Non-maskable interrupts: 3 sources
 - Maskable interrupts: 64 sources
 - Exceptions: 1 source
 - Eight levels of priorities can be set (maskable interrupts)
- DMA control function
 - 4-channel configuration
 - Transfer units: 8-bit, 16-bit, or 32-bit
 - Maximum transfer count: 65,536 (2¹⁶)
 - Transfer types: Flyby (1-cycle) transfer or 2-cycle transfer
 - Transfer modes: Single transfer, single-step transfer, line transfer, or block transfer
 - Terminal count output signals (DMTCO3 to DMTCO0)

- Power save function HALT mode
 Software STOP mode
 Hardware STOP mode
- RCU^{Note} interface function

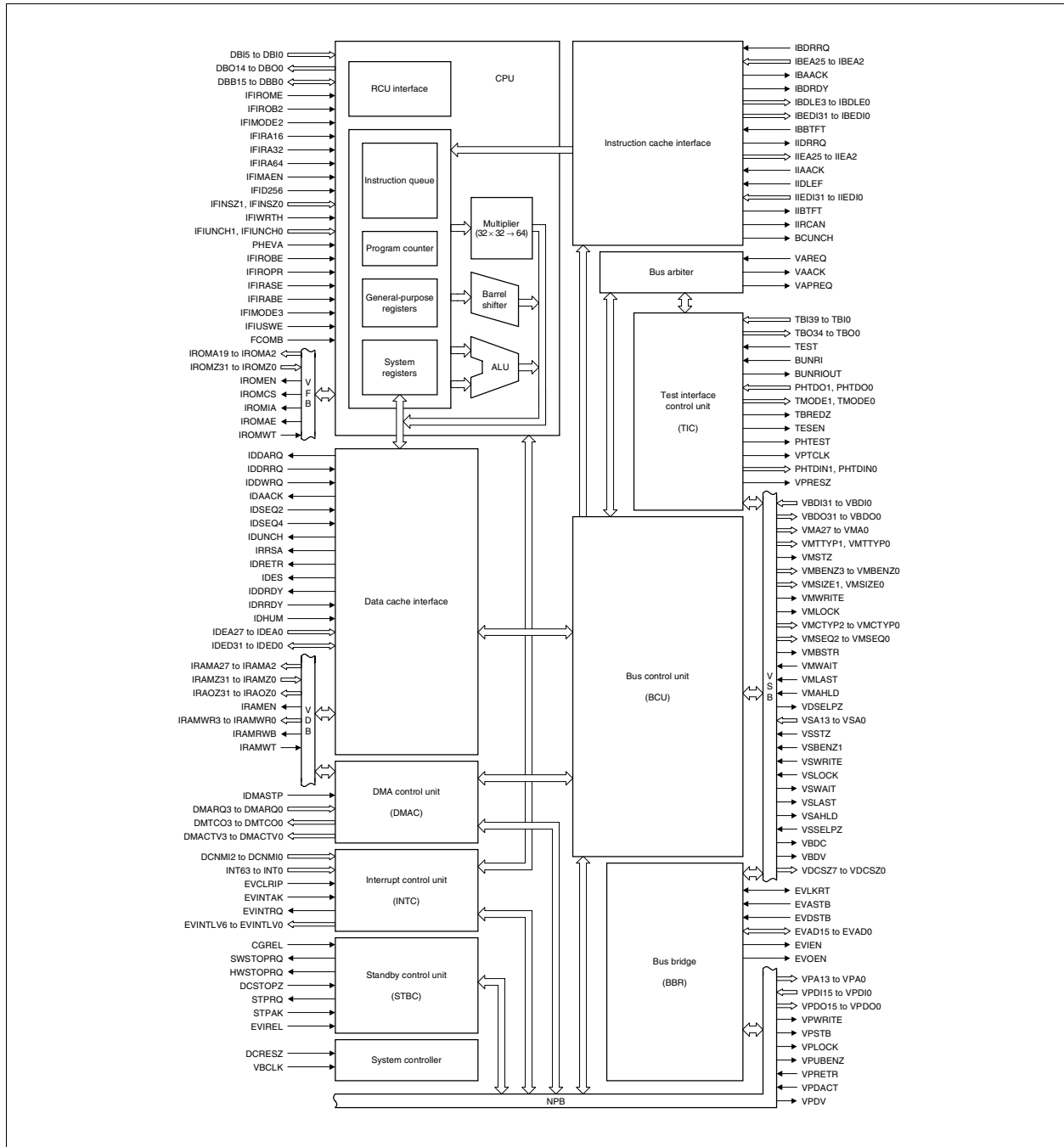
Note The Run Control Unit (RCU) communicates using JTAG and executes debug processing.

1.4 Symbol Diagram



1.5 Function Blocks

1.5.1 Internal block diagram



1.5.2 Internal units

(1) CPU

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

Other dedicated on-chip hardware, such as a hardware multiplier that enables high-speed processing of 32-bit × 32-bit multiplication and a barrel shifter, help accelerate the processing of complex instructions.

In addition, the CPU has an on-chip RCU interface for connecting to the RCU (See **CHAPTER 3 CPU**).

(2) BCU

The bus control unit (BCU), which operates as a bus master on the VSB, controls the on-chip bus bridge (BBR), test interface control unit (TIC), and peripheral macros (bus slaves) such as the memory controller (MEMC) connected to the VSB (See **CHAPTER 4 BCU**).

(3) BBR

The bus bridge (BBR) converts signals for the VSB to signals for the NPB.

The BBR sets up the wait insertion function and retry function for peripheral macros connected to the NPB (See **CHAPTER 5 BBR**).

(4) STBC

The standby control unit (STBC) controls the external clock generator (CG) when the power save function (HALT mode, software STOP mode, or hardware STOP mode) is executed (See **CHAPTER 6 STBC**).

(5) DMAC

The DMA control unit (DMAC) is a four-channel control unit that controls data transfers between memory and peripheral macros or between memory and memory based on DMA transfer requests issued by means of the DMARQ3 to DMARQ0 pins or software triggers (See **CHAPTER 7 DMAC**).

(6) INTC

The interrupt control unit (INTC) processes various types of interrupt requests (See **CHAPTER 8 INTC**).

(7) TIC

The test interface control unit (TIC) is used for test function control. When the TIC is set to test mode, test control signals become effective (See **CHAPTER 9 TEST FUNCTION**).

(8) Bus arbiter

The bus arbiter receives bus control requests from multiple bus masters and arbitrates bus access rights.

1.6 Functional Differences Between NU85E and NB85E

Item	NU85E	NB85E	
VSB data bus (n = 31 to 0)	VBDIn (input), VBDO _n (output)	VBD _n (input/output)	
VSB master/slave control pins	VMA27 to VMA0 (output)	VBA27 to VBA0 (input/output)	
	VSA13 to VSA0 (input)		
	VMTTYP1, VMTTYP0 (output)	VBTTYP1, VBTTYP0 (input/output)	
	VMSTZ (output)	VBSTZ (input/output)	
	VSSTZ (input)		
	VMBENZ3 to VMBENZ0 (output)	VBBENZ3 to VBBENZ0 (input/output)	
	VSBENZ1 (input)		
	VMSIZE1, VMSIZE0 (output)	VBSIZE1, VBSIZE0 (input/output)	
	VMWRITE (output)	VBWRITE (input/output)	
	VSWRITE (input)		
	VMLOCK (output)	VBLOCK (input/output)	
	VSLOCK (input)		
	VMCTYP2 to VMCTYP0 (output)	VBCTYP2 to VBCTYP0 (input/output)	
	VMSEQ2 to VMSEQ0 (output)	VBSEQ2 to VBSEQ0 (input/output)	
	VMBSTR (output)	VBBSTR (input/output)	
	VMWAIT (input)	VBWAIT (input/output)	
	VSWAIT (output)		
	VMLAST (input)	VBLAST (input/output)	
	VSLAST (output)		
	VMAHLD (input)	VBAHLD (input/output)	
VSAHLD (output)			
VDSELPZ (output)	VDSELPZ (input/output)		
VSSELPZ (input)			
VDCSZ7 to VDCSZ0 (output)	VDCSZ7 to VDCSZ0 (input/output)		
NPB data bus (n = 15 to 0)	VPDIn (input), VPDO _n (output)	VPD _n (input/output)	
NPB data output bus control output pin	VPDV	(None)	
VSB data output bus control output pin	VBDV	(None)	
Bus access right request output pin	VAPREQ	(None)	
Test mode status output pin	BUNRIOUT	(None)	
Input/output timing	VBD _n (n = 31 to 0)	VBDIn, VBDO _n	VBD _n
	VxTTYP _n (n = 1, 0)	VMTTYP _n	VBTTYP _n
	VxWAIT, VxLAST, VxAHLD	VMWAIT, VMLAST, VMAHLD, VSWAIT, VSLAST, VSAHLD	VBWAIT, VBLAST, VBAHLD
Pin status at reset, during idle	VxA27 to VxA0, VxSIZE1, VxSIZE0, VxSEQ2 to VxSEQ0, VBD31 to VBD0	Low-level output (VMA27 to VMA0, VMSIZE1, VMSIZE0, VMSEQ2 to VMSEQ0, VBDO31 to VBDO0)	Undefined (VBA27 to VBA0, VBSIZE1, VBSIZE0, VBSEQ2 to VBSEQ0, VBD31 to VBD0)

CHAPTER 2 PIN FUNCTIONS

2.1 List of Pin Functions

(1/4)

	Pin Name	I/O	Function
NPB pins	VPA13 to VPA0	Output	Address output for peripheral macro connected to NPB
	VPDI15 to VPDI0 ^{Note}	Input	Data input from peripheral macro connected to NPB
	VPDO15 to VPDO0	Output	Data output to peripheral macro connected to NPB
	VPWRITE	Output	Write access strobe output
	VPSTB	Output	Data strobe output
	VPLOCK	Output	Bus lock output
	VPUBENZ	Output	Upper byte enable output
	VPRETR ^{Note}	Input	Retry request input from peripheral macro connected to NPB
	VPDACT	Input	Active level input from external address decoder
	VPDV	Output	Data output (VPDO15 to VPDO0) control output
VSB pins	VAREQ	Input	Bus access right request input from external bus master
	VAACK	Output	Bus access right acknowledge output
	VAPREQ	Output	Bus access right request output from internal bus master (CPU, DMAC)
	VBDI31 to VBDI0	Input	Data input from macro connected to VSB
	VBDO31 to VBDO0	Output	Data output to macro connected to VSB
	VMA27 to VMA0	Output	Address output to macro connected to VSB
	VMTTYP1, VMTTYP0	Output	Bus transfer type output
	VMSTZ	Output	Transfer start output
	VMBENZ3 to VMBENZ0	Output	Byte enable output
	VMSIZE1, VMSIZE0	Output	Transfer size output
	VMWRITE	Output	Read/write status output
	VMLOCK	Output	Bus lock output
	VMCTYP2 to VMCTYP0	Output	Bus cycle status output
	VMSEQ2 to VMSEQ0	Output	Sequential status output
	VMBSTR	Output	Burst read status output
	VMWAIT	Input	Wait response input
	VMLAST	Input	Last response input
	VMAHLD	Input	Address hold response input
	VDSELPZ	Output	Peripheral I/O area access status output
	VSA13 to VSA0	Input	Address input from macro connected to VSB
	VSSTZ	Input	Transfer start input
	VSBENZ1	Input	Byte enable input
	VSWRITE	Input	Read/write status input

Note Connected internally to bus holder.

	Pin Name	I/O	Function
VSB pins	VSLOCK	Input	Bus lock input
	VSWAIT	Output	Wait response output
	VSLAST	Output	Last response output
	VSAHLD	Output	Address hold response output
	VSELTPZ	Input	Peripheral I/O area access status input
	VBDC	Output	Data input (VBDI31 to VBDI0) control output
	VBDV	Output	Data output (VBDO31 to VBDO0) control output
	VDCSZ7 to VDCSZ0	Output	Chip select output
System control pins	DCRESZ	Input	System reset input
	VBCLK	Input	Internal system clock input
	CGREL	Input	Clock generator release input
	SWSTOPRQ	Output	Software STOP mode request output to clock generator
	HWSTOPRQ	Output	Hardware STOP mode request output to clock generator
	DCSTOPZ	Input	Hardware STOP mode request input
	STPRQ	Output	Hardware/software STOP mode request output to MEMC
	STPAK	Input	Acknowledge input for STPRQ input of MEMC
DMAC pins	IDMASTP	Input	DMA transfer termination input
	DMARQ3 to DMARQ0	Input	DMA transfer request input
	DMTCO3 to DMTCO0	Output	Terminal count (DMA transfer completion) output
	DMACTV3 to DMACTV0	Output	DMA acknowledge output
INTC pins	DCNMI2 to DCNMI0	Input	Non-maskable interrupt request (NMI) input
	INT63 to INT0	Input	Maskable interrupt request input
VFB pins	IROMA19 to IROMA2	Output	ROM address output
	IROMZ31 to IROMZ0	Input	ROM data input
	IROMEN	Output	ROM access enable output
	IROMWT	Input	ROM wait input
	IROMCS	Output	NEC reserved pins (leave open)
	IROMIA	Output	
	IROMAE	Output	
VDB pins	IRAMA27 to IRAMA2	Output	RAM address output
	IRAMZ31 to IRAMZ0	Input	RAM data input
	IRAOZ31 to IRAOZ0	Output	RAM data output
	IRAMEN	Output	RAM access enable output
	IRAMWR3 to IRAMWR0	Output	RAM write enable output
	IRAMRWB	Output	RAM read/write status output
	IRAMWT	Input	RAM wait input
Instruction cache pins	IBDRRQ	Input	Fetch request input from instruction cache
	IBEA25 to IBEA2	Input	Fetch address input from instruction cache
	IBAACK	Output	Address acknowledge output to instruction cache

	Pin Name	I/O	Function
Instruction cache pins	IBDRDY	Output	Data ready output to instruction cache
	IBDLE3 to IBDLE0	Output	Data latch enable output to instruction cache
	IBEDI31 to IBEDI0	Output	Data output to instruction cache
	IIDRRQ	Output	Fetch request output to instruction cache
	IIEA25 to IIEA2	Output	Fetch address output to instruction cache
	IIAACK	Input	Address acknowledge input from instruction cache
	IIDLEF	Input	Data latch enable input from instruction cache
	IIEDI31 to IIEDI0	Input	Data input from instruction cache
	IIBTFT	Output	Branch target fetch status output to instruction cache
	IIRCAN	Output	Code cancel status output to instruction cache
	BCUNCH	Output	Uncache status output to instruction cache
	IBBTFT	Input	NEC reserved pin (input low level)
Data cache pins	IDDARQ	Output	Read/write access request output to data cache
	IDAACK	Output	Acknowledge output
	IDRRQ	Input	VSB read operation request input to BCU
	IDDWRQ	Input	VSB write operation request input to BCU
	IDSEQ4	Input	Read/write operation type setting input
	IDSEQ2	Input	Read/write operation type setting input
	IRRSA	Output	VDB hold status output
	IDRETR	Output	Read retry request output
	IDUNCH	Output	Uncache status output
	IDDRDY	Output	Read data ready output
	IDRRDY	Input	Read data ready input from data cache
	IDHUM	Input	Hit under miss-hit read input
	IDEA27 to IDEA0	Input	Address input
	IDED31 to IDED0 ^{Note 1}	I/O	Data input/output
IDES	Output	NEC reserved pin ^{Note 2}	
RCU pins	DBI5 to DBI0	Input	Debug control input
	DBO14 to DBO0	Output	Debug control output
	DBB15 to DBB0 ^{Note 1}	I/O	Debug control input/output
Peripheral evaluation chip mode pins	EVASTB	Input	Address strobe input
	EVDSTB	Input	Data strobe input
	EVAD15 to EVAD0 ^{Note 1}	I/O	Address/data input/output
	EVIEN	Output	EVADn input enable output (n = 15 to 0)
	EVOEN	Output	EVADn output enable output (n = 15 to 0)
	EVLKRT ^{Note 1}	I/O	Lock/retry input/output
EVIREL	Input	Standby release input	

Notes 1. Connected internally to bus holder.

2. When using the data cache, always connect this pin to the IDES pin of the data cache. Leave open when unused.

	Pin Name	I/O	Function
Peripheral evaluation chip mode pins	EVCLRIP	Input	ISPR clear input
	EVINTAK	Input	Interrupt acknowledge input
	EVINTRQ	Output	Interrupt request output
	EVINTLV6 to EVINTLV0	Output	Interrupt vector output
Operation mode setting pins	IFIROME	Input	ROM mapping enable input
	IFIROB2	Input	ROM area location setting input
	IFIRA64	Input	RAM area size selection input
	IFIRA32	Input	RAM area size selection input
	IFIRA16	Input	RAM area size selection input
	IFIMAEN	Input	Misalign access setting input
	IFID256	Input	Data area setting input
	IFINSZ1, IFINSZ0	Input	VSB data bus size (initial value) selection input
	IFIWRTH	Input	Data cache write-back/write-through mode selection input
	IFIUNCH1	Input	Data cache setting input
	IFIUNCH0	Input	Instruction cache setting input
	PHEVA	Input	Peripheral evaluation chip mode setting input
	IFIROBE	Input	NEC reserved pins (input low level)
	IFIROPR	Input	
	IFIRASE	Input	
	IFIRABE	Input	
	IFIMODE3	Input	
	IFIMODE2	Input	
	IFIUSWE	Input	
FCOMB	Input		
Test mode pins	TBI39 to TBI0	Input	
	TBO34 to TBO0	Output	Output test bus
	TEST	Input	Test bus control input
	BUNRI	Input	Normal/test mode selection input
	BUNRIOUT	Output	Test mode status output
	PHTDO1, PHTDO0 ^{Note}	Input	Peripheral macro test input
	TESEN	Output	Peripheral macro test enable output
	VPTCLK	Output	Peripheral macro test clock output
	PHTDIN1, PHTDIN0	Output	Peripheral macro test output
	VPRESZ	Output	Peripheral macro reset output
	PHTEST	Output	Peripheral test mode status output
	TMODE1	Output	Test mode selection output
	TMODE0	Output	NEC reserved pins (leave open)
	TBREDZ	Output	

Note Connected internally to bus holder.

2.2 Explanation of Pin Functions

2.2.1 NPB pins

(1) VPA13 to VPA0 (output)

These are pins from which addresses are output to peripheral macros connected to the NPB. They specify the lower 14 bits.

(2) VPDI15 to VPDI0 (input)

These are pins to which data is input from peripheral macros connected to the NPB.

(3) VPDO15 to VPDO0 (output)

These are pins from which data is output to peripheral macros connected to the NPB.

(4) VPWRITE (output)

This is the write access strobe output pin for the VPDO15 to VPDO0 signals.
During writing, a high level is output.

(5) VPSTB (output)

This is the data strobe output pin.

(6) VPLOCK (output)

This is the bus lock output pin. If an interrupt request occurs while a read modify write access to the interrupt control register (PICn) is executed, this pin outputs a bus lock signal to avoid loss of the interrupt request. It outputs a high level during a read modify write access.
Even when an interrupt request occurs, transfer to the PIFn flag of the PICn register is not performed while this signal outputs a high level (n = 0 to 63).

(7) VPUBENZ (output)

This is the higher byte enable output pin. It outputs a low level during a halfword data access or a byte data access to an odd address.
It outputs a high level during a byte access to an even address.

(8) VPRETR (input)

This is the pin to which retry requests are input from peripheral macros connected to the NPB. If a high level is input to this pin and to the VPDACT pin at the falling edge of the VPSTB signal, the read/write operation is performed again.

(9) VPDACT (input)

This pin, which is an input pin for input from an external address decoder, is used to enable the retry function. When a high level is input, the retry function is enabled.
When a low level is input, any retry request by VPRETR input will be ignored.

(10) VPDV (output)

This is the data output (VPDO15 to VPDO0) control signal output pin. It outputs a high level during write. To configure a bidirectional data bus, connect this pin to the 3-state buffer enable pin connected to the data bus for data output control.

2.2.2 VSB pins

(1) **VAREQ (input)**

This is the pin to which bus access right requests are input from an external bus master.

(2) **VAACK (output)**

This is an output pin for indicating that the bus access right request signal (VAREQ) from an external bus master has been acknowledged.

(3) **VAPREQ (output)**

This pin outputs bus access right requests from the internal bus master (CPU, DMAC) to the external bus master.

★ This pin is used when a bus master and a bus arbiter exist externally, to perform output to the external bus arbiter. This pin becomes active (1) when a bus access right request is generated, and it becomes inactive (0) when the bus cycle responding to the request has been generated. Its transition to active (1) during the CPU cycle indicates that there is a request from the DMA, and its transition to active (1) during the DMA cycle indicates that there is a request from the CPU.

(4) **VBDI31 to VBDI0 (input)**

These pins constitute a data input bus for macro connected to VSB.

(5) **VBDO31 to VBDO0 (output)**

These pins constitute a data output bus for macro connected to VSB.

(6) **VMA27 to VMA0 (output), VSA13 to VSA0 (input)**

These pins constitute an address bus for macro connected to VSB.

The NU85E uses the VMA27 to VMA0 pins when it has the bus access right, and the VSA13 to VSA0 pins when it operates as a bus slave.

(7) **VMTTY1, VMTTY0 (output)**

These pins output the bus transfer type when the NU85E has the bus access right.

Table 2-1. VMTTY1 and VMTTY0 Signals

VMTTY1	VMTTY0	Transfer Type
0	0	Address-only transfer (transfer without data processing)
1	0	Non-sequential transfer (single transfer or burst transfer)
1	1	Sequential transfer (transfer in which the address currently being transferred is related to the previously transferred address)
0	1	(Reserved for future function expansion)

Remark 0: low-level 1: high-level

(8) **VMSTZ (output), VSSTZ (input)**

These are low-level active pins that indicate transfer start.

The NU85E uses the VMSTZ pin when it has the bus access right, and the VSSTZ pin when it operates as a bus slave.

(9) VMBENZ3 to VMBENZO (output), VSBENZ1 (input)

These are low-level active pins that indicate the enabled byte data out of the four data bus (VBDI31 to VBDI0, VBDO31 to VBDO0) parts.

The NU85E uses the VMBENZ3 to VMBENZO pins when it has the bus access right, and the VSBENZ1 pin for the bus bridge (BBR) to generate the VPUBENZ signal when it operates as bus slave.

Table 2-2. VMBENZ3 to VMBENZO and VSBENZ1 Signals

Active (Low Level) Signal	Enabled Byte Data
VMBENZ3	VBDI31 to VBDI24, VBDO31 to VBDO24
VMBENZ2	VBDI23 to VBDI16, VBDO23 to VBDO16
VMBENZ1, VSBENZ1	VBDI15 to VBDI8, VBDO15 to VBDO8
VMBENZO	VBDI7 to VBDI0, VBDO7 to VBDO0

(10) VMSIZE1, VMSIZE0 (output)

These are pins that output the data transfer size when the NU85E has the bus access right.

Table 2-3. VMSIZE1 and VMSIZE0 Signals

VMSIZE1	VMSIZE0	Data Transfer Size
0	0	Byte (8 bits)
0	1	Halfword (16 bits)
1	0	Word (32 bits)
1	1	(Reserved for future function expansion)

Remark 0: low-level 1: high-level

(11) VMWRITE (output), VSWRITE (input)

These are pins that indicate the data transfer direction (read/write status). They become high level during write access.

The NU85E uses the VMWRITE pin when it has the bus access right, and the VSWRITE pin when it operates as a bus slave.

(12) VMLOCK (output), VSLOCK (input)

These pins are used to retain the bus access right. These pins are used to prohibit interruption through access from another bus master between the current transfer and the next transfer.

The NU85E uses the VMLOCK pin when it has the bus access right, and the VSLOCK pin when it operates as a bus slave.

(13) VMCTYP2 to VMCTYP0 (output)

These are pins that output the current bus cycle status when the NU85E has the bus access right.

Table 2-4. VMCTYP2 to VMCTYP0 Signals

VMCTYP2	VMCTYP1	VMCTYP0	Bus Cycle Status
0	0	0	Opcode fetch
0	0	1	Data access
0	1	0	Misalign access ^{Note}
0	1	1	Read modify write access
1	0	0	Opcode fetch of jump address due to branch instruction
1	1	0	DMA 2-cycle transfer
1	1	1	DMA flyby transfer
1	0	1	(Reserved for future function expansion)

Note Output only when a high level is input to the IFIMAEN pin (misalign access enabled).

Remark 0: low-level 1: high-level

(14) VMSEQ2 to VMSEQ0 (output)

These are pins that output the sequential status indicating the transfer size during burst transfer when the NU85E has bus access right.

These pins indicate “burst transfer length” at the start of burst transfer, “continuous” during burst transfer, and “single transfer” at the end of burst transfer.

In the following cases, VSB changes to burst transfer and the sequential status indicates “continuous”.

- VSB is 8 bits wide and 16-/32-bit data transfer was performed
- VSB is 16 bits wide and 32-bit data transfer was performed
- Refill from instruction/data cache
- 32-bit data transfer to peripheral macro connected to NPB (16-bit data bus width)

Table 2-5. VMSEQ2 to VMSEQ0 Signals

VMSEQ2	VMSEQ1	VMSEQ0	Sequential Status
0	0	0	Single transfer
0	0	1	Continuous (indicates that the next transfer address is related to the current transfer address) ^{Note}
0	1	0	Continuous 4 times (burst transfer length: 4)
0	1	1	Continuous 8 times (burst transfer length: 8)
1	0	0	Continuous 16 times (burst transfer length: 16)
1	0	1	Continuous 32 times (burst transfer length: 32)
1	1	0	Continuous 64 times (burst transfer length: 64)
1	1	1	Continuous 128 times (burst transfer length: 128)

Note This is output during continuous 2 times, or continuous 4, 8, 16, 32, 64, or 128 times transfer.

Remark 0: low-level 1: high-level

(15) VMBSTR (output)

This pin outputs the burst read status indicating that the current transfer is opcode fetch from external ROM when the NU85E has the bus access right and ROM connected as external memory (accessed via VSB) is used. This pin operates with the same timing as the address bus.

(16) VMWAIT (input), VSWAIT (output)

These are wait response pins.

These signals are output to the bus master to request additional bus cycles when the selected bus slave has not completed data output preparations. When these signals become high level, the bus cycle changes to the wait status.

The NU85E uses the VMWAIT pin when it has the bus access right, and the VSWAIT pin when it operates as a bus slave.

If a memory controller (MEMC) is connected to the NU85E, a high level is output to the VMWAIT pin of the NU85E from MEMC while the VSB cycle occurs because the access cycle is always 2 or more clocks.

(17) VMLAST (input), VSLAST (output)

These are last response pins. These pins are used when the bus decoder requires a decode cycle.

In the case of a system where several slave devices are connected externally and a bus decoder has been added to select slaves, decoding for bus slave selection is normally performed during non-sequential transfer. Thus even when attempts to change a slave device are made during sequential transfer such as burst transfer, the decode cycle for slave selection cannot be issued.

In such a case, the slave device outputs a last response notifying the fact that the slave selection signal has changed to the bus master. When there is a last response from the slave device, the bus master makes the next bus cycle non-sequential transfer to enable decode cycle issuance.

The NU85E uses the VMLAST pin to operate as the bus master, and the VSLAST pin when it operates as a bus slave. The VSLAST pin, however, is fixed to low-level output and does not become active.

★

(18) VMAHLD (input), VSAHLD (output)

These are address hold response pins.

These signals are output to the bus master when the selected bus slave has completed data output preparations and requests the bus cycle. When this signal and the VxWAIT signal become high level, the bus cycle goes into the address hold status.

Since, in the address hold status, addresses do not change even during the data read and write cycles, there is no need to latch addresses and the circuit can thus be kept simple.

The NU85E uses the VMAHLD pin when it has the bus access right, and the VSAHLD pin when it operates as a bus slave. The VSAHLD pin, however, is fixed to low-level output and does not become active.

★

If a memory controller (MEMC) is connected to the NU85E, a high level is output to the NU85E from the MEMC when an idle state is inserted.

(19) VDSELPZ (output), VSSELPZ (input)

These pins are used to output a low level to the bus slave when the bus master accesses a peripheral I/O area or programmable peripheral I/O area.

The NU85E uses the VDSELPZ pin when it has the bus access right, and the VSSELPZ pin when it operates as a bus slave.

(20) VBDC (output)

- ★ This is data input (VBDI31 to VBDI0) control signal output pin. This pin outputs a high level during a read cycle and during DMA flyby transfer from the external memory to the I/O. When connecting a bus slave that has an I/O separated data bus and a bidirectional data bus, this pin is connected to the enable pin of the 3-state buffer connected to the data bus for data input control.

(21) VBDV (output)

- ★ This is data output (VBDO31 to VBDO0) control signal output pin. This pin outputs a high level during a write cycle and during DMA flyby transfer from the I/O to the external memory. When configuring a bidirectional data bus, this pin is connected to the enable pin of the 3-state buffer connected to the data bus for data output control.

(22) VDCSZ7 to VDCSZ0 (output)

These are low-level active chip select output pins. For details, refer to **4.3 Programmable Chip Select Function**.

2.2.3 System control pins

(1) DCRESZ (input)

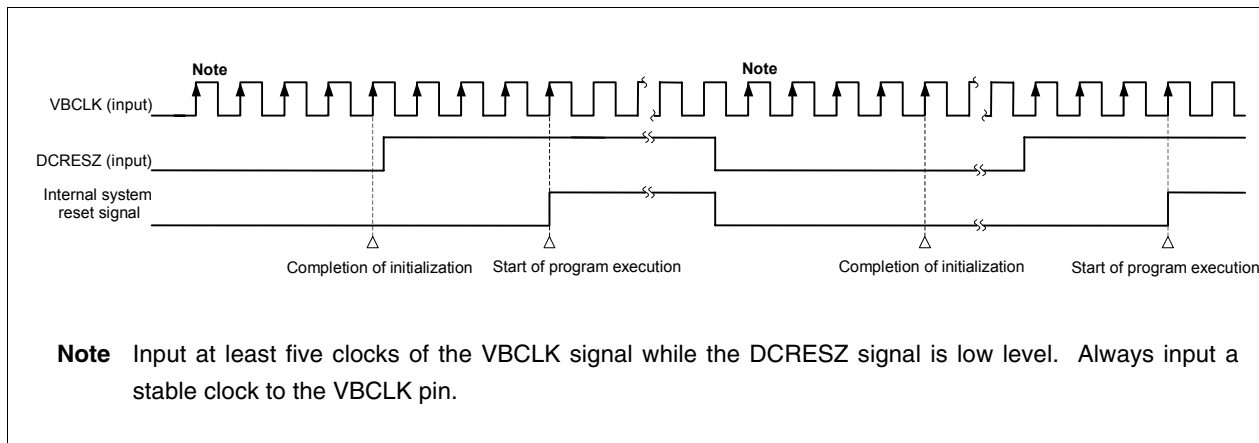
This is the clock-synchronized system reset input pin.

When the stable input clock rising edge is detected five times after a low level was input to this pin, the pin statuses and internal signals are completely initialized (The time required until the statuses of the internal signals and each pin are stabilized is 5 clocks or less depending on the pin. Noise elimination is not performed.). Also, when the input clock rising edge is detected four times after this signal has risen from low level to high level, the pipeline is cleared and program execution starts from memory address 0.

- ★ In addition to normal initialization and start operations, this pin is used to cancel the power save function.

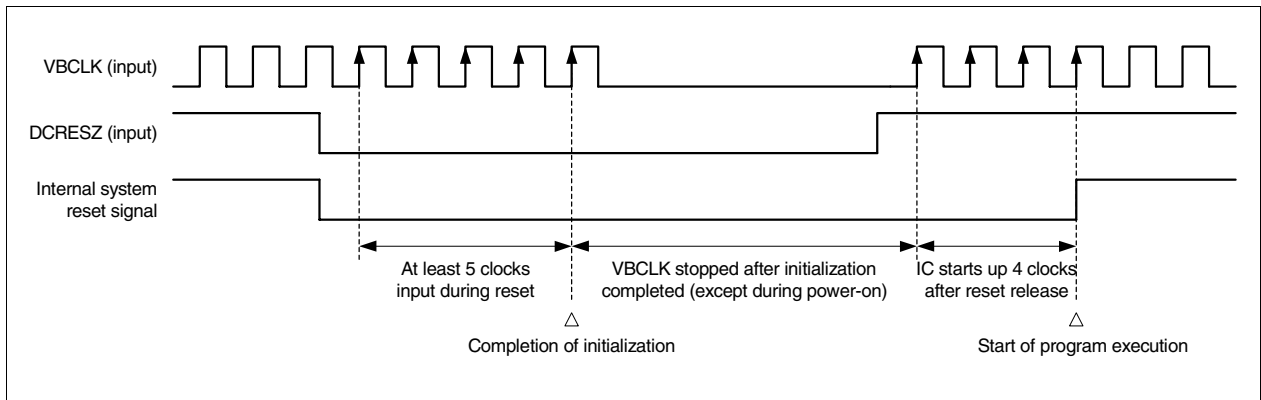
Caution Be sure to input the DCRESZ signal so that the setup and hold times referenced to the VBCLK signal are satisfied.

Figure 2-1. Acknowledgement of DCRESZ Signal



- ★ To stop VBCLK oscillation by system reset when the NB85E901 is connected, input at least five VBCLK clocks after the system reset has become low level to completely initialize the status of the pins related to the CPU and the internal signals, then stop VBCLK oscillation. Unless this restriction is followed, the debugger may not start successfully.

★ **Figure 2-2. Stopping VBCLK Oscillation by System Reset**



(2) VBCLK (input)

This is the external clock input pin for the internal system clock. A 50% duty stable clock is input from an external clock control circuit.

(3) CGREL (input)

This is the release input pin for the external clock generator (CG). An active level (high level) is input upon the start of VBCLK input at least one clock after STOP mode is canceled and the oscillation stabilization time has been ensured (it is not necessary to set CGREL input at the same time as VBCLK input).

(4) SWSTOPRQ (output)

This is the pin from which software STOP mode requests are output to the external clock generator (CG). When software STOP mode is set, this pin outputs a high-level signal.

VBCLK input from the CG is stopped by using this signal. When software STOP mode is canceled, this pin outputs a low-level signal.

(5) HWSTOPRQ (output)

This is the pin from which hardware STOP mode requests are output to the external clock generator (CG). When hardware STOP mode is set by DCSTOPZ input, this pin outputs a high-level signal.

VBCLK input from the CG is stopped by using this signal. When hardware STOP mode is canceled, this pin outputs a low-level signal.

(6) DCSTOPZ (input)

This is a hardware STOP mode request input pin. When a low-level signal is input, the NU85E is set to hardware STOP mode.

(7) STPRQ (output)

This is the pin from which hardware/software STOP mode requests are output to the memory controller (MEMC).

(8) STPAK (input)

This is the pin to which acknowledge signals are input from the memory controller (MEMC) acknowledging the STPRQ signal.

2.2.4 DMAC pins

(1) IDMASTP (input)

This is the DMA transfer forcible interrupt input pin. Input an active level (high level) of two clocks in synchronization with the rising edge of the VBCLK signal.

To restart transfer, set (1) the EN bit of the DRST register after inputting a low level to this pin.

(2) DMARQ3 to DMARQ0 (input)

These are the DMA transfer request input pins.

Input an active level (high level) in synchronization with the rising edge of the VBCLK signal, and continue inputting until the corresponding DMACTV_n signal becomes high level (n = 3 to 0).

(3) DMTCO3 to DMTCO0 (output)

These are the terminal count (DMA transfer completion) output pins. A one-clock high level is output from these pins when the final DMA transfer is performed.

The high level is output in synchronization with the rising edge of the VBCLK signal.

(4) DMACTV3 to DMACTV0 (output)

These are the DMA acknowledge output pins. These pins become active (high-level output) during a 2-cycle transfer VSB read or VSB write cycle, or during a flyby transfer.

2.2.5 INTC pins

(1) DCNMI2 to DCNMI0 (input)

These are the non-maskable interrupt request (NMI) input pins. When a rising edge is input, a non-maskable interrupt is generated.

(2) INT63 to INT0 (input)

These are the maskable interrupt request input pins. When a rising edge is input, a maskable interrupt is generated.

2.2.6 VFB pins

(1) IROMA19 to IROMA2 (output)

These pins constitute a bus from which addresses are output to ROM.

(2) IROMZ31 to IROMZ0 (input)

These pins constitute a bus to which data is input from ROM.

(3) IROMEN (output)

This is the pin from which access enable signals are output to ROM. It changes in synchronization with the falling edge of the VBCLK signal.

(4) IROMWT (input)

This is the pin to which wait signals are input from ROM. A high level is input during the wait period.

(5) IROMCS, IROMIA, IROMAE (output)

These are NEC reserved pins. Leave them open.

2.2.7 VDB pins

(1) IRAMA27 to IRAMA2 (output)

These pins constitute a bus from which addresses are output to RAM. The IRAMA27 to IRAMA16 signals are output for the data cache. Therefore, they do not have to be decoded when RAM is connected.

(2) IRAMZ31 to IRAMZ0 (input)

These pins constitute a bus to which data is input from RAM.

(3) IRAOZ31 to IRAOZ0 (output)

These pins constitute a bus from which data is output to RAM.

(4) IRAMEN (output)

This is the pin from which access enable signals are output to RAM. It changes in synchronization with the falling edge of the VBCLK signal.

(5) IRAMWR3 to IRAMWR0 (output)

These are the pins from which write enable signals are output to RAM. They are high-level active pins that indicate the enabled byte data among the output data bus pins (IRAOZ31 to IRAOZ0).

Table 2-6. IRAMWR3 to IRAMWR0 Signals

Active (High-Level Output) Signal	Enabled Byte Data
IRAMWR0	IRAOZ7 to IRAOZ0
IRAMWR1	IRAOZ15 to IRAOZ8
IRAMWR2	IRAOZ23 to IRAOZ16
IRAMWR3	IRAOZ31 to IRAOZ24

(6) IRAMRWB (output)

This is the pin from which the read/write status is output to RAM. During reading, a high-level signal is output. During writing, a low-level signal is output.

(7) IRAMWT (input)

This is the pin to which wait signals are input from the data cache. A high level is input during the wait period.

2.2.8 Instruction cache pins

(1) IBDRRQ (input)

This is the pin to which fetch requests are input from the instruction cache.
A request signal is input which fetches data from the external memory to the NU85E.

(2) IBEA25 to IBEA2 (input)

These pins constitute a bus to which fetch addresses are input from the instruction cache.
Upon a miss-hit, the address to be read is input from the instruction cache.

(3) IBAACK (output)

This is the pin from which address acknowledgements are output to the instruction cache.
This signal is output when the NU85E recognizes the IBEA25 to IBEA2 signals input from the instruction cache.

(4) IBDRDY (output)

This is the pin from which data ready signals are output to the instruction cache.
Upon an instruction cache miss-hit, when the NU85E has finished fetching the data to be read from the external memory, this signal is output to indicate that a refill for the instruction cache is ready.

(5) IBDLE3 to IBDLE0 (output)

These are the pins from which data latch enable signals are output to the instruction cache.

(6) IBEDI31 to IBEDI0 (output)

These pins constitute a bus from which data is output to the instruction cache.
Upon an instruction cache miss-hit, the data to be refilled is output to the instruction cache.

(7) IIDRRQ (output)

This is the pin from which fetch requests are output to the instruction cache.

(8) IIEA25 to IIEA2 (output)

These pins constitute a bus from which fetch addresses are output to the instruction cache.
The address to be fetched is output from the external memory simultaneous with the fetch request (IIDRRQ).

(9) IIAACK (input)

This is the pin to which address acknowledgements are input from the instruction cache.
This signal is input to the NU85E when the instruction cache recognizes the fetch address signals (IIEA25 to IIEA2) input from the NU85E.

(10) IIDLEF (input)

This is the pin to which data latch enable signals are input from the instruction cache.

(11) IIEDI31 to IIEDI0 (input)

These pins constitute a bus to which data is input from the instruction cache.
The data to be read is input from the instruction cache.

(12) IIBTFT (output)

This is the pin from which the branch target fetch status is output to the instruction cache.
A high level is output when a jump destination address is fetched due to a branch instruction.

(13) IIRCAN (output)

This is the pin from which the code cancel status is output to the instruction cache.

This signal cancels previous requests when data becomes unwanted due to a branch or interrupt after the NU85E outputs a fetch request to the instruction cache.

(14) BCUNCH (output)

This is the pin from which the uncache status is output to the instruction cache.

A low level is output when the area in which the instruction cache setting has been set to cache-enable using the cache configuration register (BHC) is accessed.

(15) IBBTFT (input)

This is an NEC reserved pin. Always input a low level.

Note that the IBBTFT pin of the connected instruction cache should be left open when using the instruction cache.

2.2.9 Data cache pins

(1) IDDARQ (output)

This is the pin from which read/write access requests are output to the data cache.

(2) IDAACK (output)

This is the pin from which acknowledgements are output to the data cache.

This signal is output when the NU85E recognizes the IDEA27 to IDEA0 signals input from the data cache.

(3) IDRRQ, IDDWRQ, IDSEQ4, IDSEQ2 (input)

These are the pins to which the operation type settings are input from the data cache.

Table 2-7. IDRRQ, IDDWRQ, IDSEQ4, and IDSEQ2 Signals

IDRRQ	IDDWRQ	IDSEQ4	IDSEQ2	Operation Type
1	0	1	0	4-word sequential read
1	0	0	1	2-word sequential read
1	0	0	0	1-word read
0	1	1	0	4-word sequential write
0	1	0	1	2-word sequential write
0	1	0	0	1-word write
1	1	1	1	1-word write
1	1	1	0	1-halfword write
1	1	0	0	1-byte write
Other than above				Setting prohibited

Remark 0: low-level input 1: high-level input

(a) IDRRQ (input)

This is the pin to which VSB read operation requests are input from the data cache.

(b) IDWRQ (input)

This is the pin to which VSB write operation requests are input from the data cache.

(c) IDSEQ4, IDSEQ2 (input)

These are the pins to which the read/write operation type settings are input from the data cache.

(4) IRRSA (output)

This is the pin from which the VDB hold status is output to the data cache.

An active level (high level) is output when the VDB is accessing RAM or is in the hold state.

(5) IDRETR (output)

This is the pin from which read retry requests are output to the data cache.

(6) IDUNCH (output)

This is the pin from which the uncache status is output to the data cache.

A low level is output when the area in which the data cache setting has been set to cache-enable using the cache configuration register (BHC) is accessed.

(7) IDRDY (output)

This is the pin from which read data ready signals are output to the data cache.

Upon a data cache miss-hit, when the NU85E has finished fetching the data to be read from the external memory, this signal is output to indicate that a refill for the data cache is ready.

(8) IDRRDY (input)

This is the pin to which read data ready signals are input from the data cache.

(9) IDHUM (input)

This is the pin to which hit-under-miss-hit read signals are input from the data cache.

A high level is input in cases when a subsequent access is made to the data cache while the external memory is being accessed due to the generation of a miss-hit during a read operation, and the data that scored a hit on this subsequent access is input to the NU85E ahead of the data from the external memory (hit-under-miss-hit).

(10) IDEA27 to IDEA0 (input)

These pins constitute a bus to which addresses are input from the data cache.

The address to be accessed is input to the NU85E upon a data cache miss-hit.

(11) IDED31 to IDED0 (input/output)

These pins constitute a data bus through which data is input/output from/to the data cache.

Data for refilling the data cache and data written to the external memory in write back mode is exchanged.

(12) IDES (output)

This is an NEC reserved pin. When using the data cache, be sure to connect this pin to the IDES pin of the connected data cache. When not using the data cache, leave this pin open.

2.2.10 RCU pins

(1) DBI5 to DBI0 (input)

These are debug control input pins. They are connected to the DBI5 to DBI0 pins of the RCU.

(2) DBO14 to DBO0 (output)

These are debug control output pins. They are connected to the DBO14 to DBO0 pins of the RCU.

(3) DBB15 to DBB0 (input/output)

These are debug control I/O pins. They are connected to the DBB15 to DBB0 pins of the RCU.

2.2.11 Peripheral evaluation chip mode pins

If a high-level signal is input to the PHEVA pin, the NU85E is set to peripheral evaluation chip mode.

In peripheral evaluation chip mode, the ASIC in which the NU85E is incorporated is used as a peripheral emulation chip when the in-circuit emulator is used to perform debugging.

The peripheral evaluation chip mode pins constitute an interface with the evaluation chip within the in-circuit emulator, and various evaluation chip signals are converted to NPB signals via these pins.

(1) EVASTB (input)

This is the address strobe input pin. It is connected to the EPHASTB pin of the evaluation chip.

(2) EVDSTB (input)

This is the data strobe input pin. It is connected to the EPHDSTB pin of the evaluation chip.

(3) EVAD15 to EVAD0 (input/output)

These pins constitute an address/data bus. They are connected to the EPHADn pins of the evaluation chip (n = 15 to 0).

(4) EVIEN (output)

This pin outputs an input enable signal for controlling the direction of the I/O buffer on the EVADn bus (n = 15 to 0).

(5) EVOEN (output)

This pin outputs an output enable signal for controlling the direction of the I/O buffer on the EVADn bus (n = 15 to 0).

(6) EVLKRT (input/output)

This is the lock/retry input/output pin. It is connected to the EPHLKRT pin of the evaluation chip.

(7) EVIREL (input)

This is the standby release input pin.

(8) EVCLRIP (input)

This is the ISPR clear input pin. It is connected to the ECLRIP pin of the evaluation chip.

(9) EVINTAK (input)

This is the interrupt acknowledge input pin. It is connected to the EINTAK pin of the evaluation chip.

(10) EVINTRQ (output)

This is the interrupt request output pin. It is connected to the EINTRQ pin of the evaluation chip.

(11) EVINTLV6 to EVINTLV0 (output)

These are the interrupt vector output pins. They are connected to the EINTLV6 to EINTLV0 pins of the evaluation chip.

2.2.12 Operation mode setting pins

The following pins are used to specify the operation mode of the NU85E.

The input level to these pins should remain fixed during NU85E operation. Do not change the input level to these pins during operation.

(1) IFIROME (input)

This is the ROM area setting input pin. The setting is made according to the level input to the pin, as shown below.

- Low level: ROM connected as external memory (via VSB) is used
- High level: ROM connected to VFB is used

When a low level is input to this pin, instruction processing begins after branching to the reset entry address of the external memory, following the release of system reset. Instruction fetches and data access to the ROM connected to VFB cannot be performed.

If a high level is input to this pin and a low level is input to the IFIROB2 pin, instruction processing begins after branching to the reset entry address of the ROM connected to VFB, following the release of system reset. If a high level is input to the IFIROB2 pin, instruction processing begins after branching to the reset entry address of the external memory, following the release of system reset, however it is possible to access the area of ROM connected to VFB that is allocated to addresses 100000H and higher.

(2) IFIROB2 (input)

This is the ROM area relocation setting input pin. It specifies the range for locating the ROM area. The ROM area range is set as follows according to the input level to this pin.

- Low level: Addresses 000000H to 0FFFFFFH
- High level: Addresses 100000H to 1FFFFFFH

For details, see **3.4.1 (1) ROM relocation function**.

(3) IFIRA64, IFIRA32, IFIRA16 (input)

These are RAM area size selection input pins.

The RAM area size is set as follows according to the input level to these pins.

For details, see **3.4.2 RAM area**.

Table 2-8. IFIRA64, IFIRA32, and IFIRA16 Signals

IFIRA64	IFIRA32	IFIRA16	RAM Area Size
0	0	0	4 KB
0	0	1	12 KB
0	1	Arbitrary	28 KB
1	Arbitrary	Arbitrary	60 KB

Remark 0: low-level input 1: high-level input

(4) IFIMAEN (input)

This is the misalign access setting input pin.

Misalign access is enabled or disabled as follows according to the input level to this pin.

- Low level: Misalign access disabled
- High level: Misalign access enabled

(5) IFID256 (input)

This is the data area setting input pin. It is used to set the data area size.

Each mode is set as follows according to the input level to this pin.

For details, see **3.3.2 Data area**.

- Low level: 64 MB mode
- High level: 256 MB mode

(6) IFINSZ1, IFINSZ0 (input)

These are the VSB data bus size (initial value) selection input pins.

The VSB data bus size is set as follows according to the input level to these pins.

Table 2-9. IFINSZ1 and IFINSZ0 Signals

IFINSZ1	IFINSZ0	VSB Data Bus Size
0	0	32 bits
0	1	16 bits
1	0	8 bits
1	1	Setting prohibited

Remark 0: low-level input 1: high-level input

If the VSB data bus size is changed after reset through the bus size configuration register (BSC), the setting of the BSC register is valid regardless of the input level to these pins.

(7) IFIWRTH (input)

This is the data cache write-back or write-through mode selection input pin.

When using the data cache, connect to the IFIWRTH pin of the data cache.

Each mode is set as follows according to the input level to this pin.

- Low level: Write-back mode
- High level: Write-through mode

(8) IFIUNCH1 (input)

This is the data cache setting input pin.

When using the data cache, connect to the IFIUNCH1 pin of the data cache.

The data cache is enabled or disabled as follows according to the input level to this pin.

- Low level: Data cache is enabled
- High level: Data cache is disabled

(9) IFIUNCHO (input)

This is the instruction cache setting input pin.

The instruction cache is enabled or disabled as follows according to the input level to this pin.

- Low level: Instruction cache is enabled
- High level: Instruction cache is disabled

(10) PHEVA (input)

This is the peripheral evaluation chip mode setting input pin. A high level is input when the ASIC in which the NU85E has been incorporated is used as a peripheral evaluation chip.

(11) IFIROBE, IFIROPR, IFIRASE, IFIRABE, IFIMODE3, IFIMODE2, IFIUSWE, FCOMB (input)

These are NEC reserved pins. Always input low-level signals.

2.2.13 Test mode pins**(1) TBI39 to TBI0 (input)**

These pins constitute an input test bus.

(2) TBO34 to TBO0 (output)

These pins constitute an output test bus.

(3) TEST (input)

This is the test bus control input pin.

(4) BUNRI (input)

This is the input pin for selecting normal or test mode.

(5) BUNRIOUT (output)

This is the status output pin that indicates the test mode status.

The level of the BUNRI pin (input) is output as is.

(6) PHTDO1, PHTDO0 (input)

These pins are the peripheral macro test input pins.

(7) TESEN (output)

This is the enable output pin for setting peripheral macros to test mode.

(8) VPTCLK (output)

This is the clock output pin for peripheral macro tests.

(9) PHTDIN1, PHTDIN0 (output)

These are the peripheral macro test output pins.

(10) VPRESZ (output)

This is the pin from which reset signals are output to the peripheral macros.

Caution The VPRESZ signal is the reset signal for the peripheral macros in normal operation mode as well as test mode.

(11) PHTEST (output)

This is the pin from which signals indicating the peripheral test mode status are output.

(12) TMODE1 (output)

This is the test mode selection output pin. When using the RCU, connect this pin to the TMODE1 and TEST pins of the RCU.

(13) TMODE0, TBREDZ (output)

These are NEC reserved pins. Leave them open.

2.3 Recommended Connection of Unused Pins

(1/2)

	Pin Name	I/O	Recommended Connection Method
NPB pins	VPA13 to VPA0, VPDO15 to VPDO0, VPWRITE, VPSTB, VPLOCK, VPUBENZ, VPDV	Output	Leave open.
	VPDI15 to VPDI0 ^{Note} , VPRETR ^{Note}	Input	Input low level.
	VPDACT	Input	Input high level.
VSB pins	VAREQ	Input	Input low level.
	VAACK, VAPREQ, VBDO31 to VBDO0, VMA27 to VMA0, VMTTYP1, VMTTYP0, VMSTZ, VMBENZ3 to VMBENZ0, VMSIZE1, VMSIZE0, VMWRITE, VMLOCK, VMCTYP2 to VMCTYP0, VMSEQ2 to VMSEQ0, VMBSTR, VDSELPZ, VSWAIT, VSLAST, VSAHLD, VBDC, VBDV, VDACS7 to VDACS0	Output	Leave open.
	VBDI31 to VBDI0, VMWAIT, VMLAST, VMAHLD, VSA13 to VSA0, VSWRITE, VSLOCK	Input	Input low level.
	VSSTZ, VSBENZ1, VSSELPZ	Input	Input high level.
System control pins	DCRESZ, VBCLK	Input	–
	CGREL	Input	Input low level.
	SWSTOPRQ, HWSTOPRQ, STPRQ	Output	Leave open.
	DCSTOPZ, STPAK	Input	Input high level.
DMAC pins	IDMASTP, DMARQ3 to DMARQ0	Input	Input low level.
	DMTCO3 to DMTCO0, DMACTV3 to DMACTV0	Output	Leave open.
INTC pins	DCNMI2 to DCNMI0, INT63 to INT0	Input	Input low level.
VFB pins	IROMA19 to IROMA2, IROMEN, IROMCS, IROMIA, IROMAE	Output	Leave open.
	IROMZ31 to IROMZ0	Input	Input high level.
	IROMWT	Input	Input low level.
VDB pins	IRAMA27 to IRAMA2, IRAOZ31 to IRAOZ0, IRAMEN, IRAMWR3 to IRAMWR0, IRAMRWB	Output	Leave open.
	IRAMZ31 to IRAMZ0	Input	Input high level.
	IRAMWT	Input	Input low level.
Instruction cache pins	IBDRRQ, IBEA25 to IBEA2, IIAACK, IIDLEF, IIEDI31 to IIEDI0, IBBTFT	Input	Input low level.
	IIBAACK, IBDRDY, IBdle3 to IBdle0, IBEDI31 to IBEDI0, IIDRRQ, IIEA25 to IIEA2, IIBTFT, IIRCAN, BCUNCH	Output	Leave open.
Data cache pins	IDDARQ, IDAACK, IRRSA, IDRETR, IDUNCH, IDDRDY, IDES	Output	Leave open.
	IDDRRQ, IDDWRQ, IDSEQ4, IDSEQ2, IDRRDY, IDHUM, IDEA27 to IDEA0	Input	Input low level.
	IDED31 to IDED0	I/O	Leave open.

★ **Note** Clamp to low level via a buffer.

	Pin Name	I/O	Recommended Connection Method
RCU pins	DBI5, DBI1	Input	Input high level.
	DBI4 to DBI2, DBI0	Input	Input low level.
	DBO14 to DBO0	Output	Leave open.
	DBB15 to DBB0	I/O	Leave open.
Peripheral evaluation chip mode pins	EVASTB, EVDSTB, EVIREL, EVCLRIP, EVINTAK	Input	Input low level.
	EVAD15 to EVAD0, EVLKRT	I/O	Leave open.
	EVIEN, EVOEN, EVINTRQ, EVINTLV6 to EVINTLV0	Output	Leave open.
Operation mode setting pins	IFIROME, IFIRA64, IFIRA32, IFIRA16, IFIMAEN, IFID256, IFINSZ1, IFINSZ0	Input	–
	IFIROB2, IFIWRTH, IFIUNCH0	Input	Input low level or high level.
	PHEVA, IFIROBE, IFIROPR, IFIRASE, IFIRABE, IFIMODE3, IFIMODE2, IFIUSWE, FCOMB	Input	Input low level.
	IFIUNCH1	Input	Input high level.
Test mode pins	TBI39 to TBI0	Input	Refer to the design editions of the various cell-based IC family user's manuals.
	TBO34 to TBO0	Output	
	TEST, BUNRI	Input	–
	PHTDO1 ^{Note} , PHTDO0 ^{Note}	Input	Input low level.
	BUNRIOUT, TESEN, VPTCLK, PHTDIN1, PHTDIN0, VPRESZ, PHTTEST, TMODE1, TMODE0, TBREDZ	Output	Leave open.

★ **Note** Clamp to low level via a buffer.

2.4 Pin Status

The following table shows the status in each operating mode of the pins that have output functions.

Table 2-10. Pin Status in Each Operating Mode (1/3)

Pin Name		Pin Status					
		Reset ^{Note}	Software STOP Mode	Hardware STOP Mode	HALT Mode	Standby Test Mode	Unit Test Mode
NPB pins	VPA13 to VPA0	Undefined	Retained	Retained	Operates	Undefined	Operates
	VPDO15 to VPDO0	Undefined	Retained	Retained	Operates	Undefined	Operates
	VPWRITE	Undefined	Retained	Retained	Operates	Undefined	Operates
	VPSTB	L	L	L	Operates	Undefined	Operates
	VPLOCK	Undefined	Retained	Retained	Operates	Undefined	Operates
	VPUBENZ	Undefined	Retained	Retained	Operates	Undefined	Operates
	VPDV	Undefined	Retained	Retained	Operates	Undefined	Operates
VSB pins	VAACK	L	Retained	Retained	Operates	Undefined	Operates
	VAPREQ	L	Retained	Retained	Operates	Undefined	Operates
	VBDO31 to VBDO0	L	Retained	Retained	Operates	Undefined	Operates
	VMA27 to VMA0	L	Retained	Retained	Operates	Undefined	Operates
	VMTTYP1, VMTTYP0	L	Retained	Retained	Operates	Undefined	Operates
	VMSTZ	H	Retained	Retained	Operates	Undefined	Operates
	VMBENZ3 to VMBENZ0	H	Retained	Retained	Operates	Undefined	Operates
	VMSIZE1, VMSIZE0	L	Retained	Retained	Operates	Undefined	Operates
	VMWRITE	L	Retained	Retained	Operates	Undefined	Operates
	VMLOCK	L	Retained	Retained	Operates	Undefined	Operates
	VMCTYP2 to VMCTYP0	Undefined	Retained	Retained	Operates	Undefined	Operates
	VMSEQ2 to VMSEQ0	L	Retained	Retained	Operates	Undefined	Operates
	VMBSTR	L	Retained	Retained	Operates	Undefined	Operates
	VDELPZ	H	Retained	Retained	Operates	Undefined	Operates
	VSWAIT	L	Retained	Retained	Operates	Undefined	Operates
VSLAST	L	Retained	Retained	Operates	Undefined	Operates	
VSAHLD	L	Retained	Retained	Operates	Undefined	Operates	

Note When a low level is input to the DCRESZ pin and an external clock is input to the VBCLK pin.

Remark L: Low-level output
H: High-level output
Retained: Retains status immediately before

Table 2-10. Pin Status in Each Operating Mode (2/3)

Pin Name		Pin Status					
		Reset ^{Note}	Software STOP Mode	Hardware STOP Mode	HALT Mode	Standby Test Mode	Unit Test Mode
VSB pins	VBDC	L	L	L	Operates	Undefined	Operates
	VBDV	L	L	L	Operates	Undefined	Operates
	VDCSZ7 to VDCSZ0	H	Retained	Retained	Operates	Undefined	Operates
System control pins	SWSTOPRQ	L	H	L	L	Undefined	Undefined
	HWSTOPRQ	L	L	H	L	Undefined	Undefined
	STPRQ	L	H	H	L	Undefined	Undefined
DMAC pins	DMTCO3 to DMTCO0	L	L	L	Operates	Undefined	Undefined
	DMACTV3 to DMACTV0	L	L	L	Operates	Undefined	Undefined
VFB pins	IROMA19 to IROMA2	Undefined	Retained	Retained	Retained	Undefined	Operates
	IROMEN	L	L	L	L	Undefined	Operates
	IROMCS	L	L	L	L	Undefined	Operates
	IROMIA	H	Undefined	Undefined	Undefined	Undefined	Operates
	IROMAE	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
VDB pins	IRAMA27 to IRAMA2	Undefined	Undefined	Undefined	Operates	Undefined	Operates
	IRAOZ31 to IRAOZ0	Undefined	Undefined	Undefined	Operates	Undefined	Operates
	IRAMEN	L	L	L	Operates	Undefined	Operates
	IRAMWR3 to IRAMWR0	L	L	L	Operates	Undefined	Operates
	IRAMRWB	Undefined	Undefined	Undefined	Operates	Undefined	Operates
Instruction cache pins	IBAACK	L	L	L	L	Undefined	Operates
	IBDRDY	L	L	L	L	Undefined	Operates
	IBDLE3 to IBDLE0	L	L	L	L	Undefined	Operates
	IBEDI31 to IBEDI0	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
	IIDRRQ	L	L	L	L	Undefined	Operates
	IIEA25 to IIEA2	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
	IIBTFT	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
	IIRCAN	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
BCUNCH	L	L	L	L	Undefined	Operates	

Note When a low level is input to the DCRESZ pin and an external clock is input to the VBCLK pin.

Remark L: Low-level output
H: High-level output
Retained: Retains status immediately before

Table 2-10. Pin Status in Each Operating Mode (3/3)

Pin Name		Pin Status					
		Reset ^{Note}	Software STOP Mode	Hardware STOP Mode	HALT Mode	Standby Test Mode	Unit Test Mode
Data cache pins	IDDARQ	L	L	L	L	Undefined	Operates
	IDAACK	L	L	L	L	Undefined	Operates
	IRRSA	L	Undefined	Undefined	Undefined	Undefined	Operates
	IDRETR	L	L	L	L	Undefined	Operates
	IDUNCH	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
	IDDRDY	L	L	L	L	Undefined	Operates
	IDED31 to IDED0	Undefined	Undefined	Undefined	Undefined	Undefined	Operates
IDES	Undefined	Undefined	Undefined	Undefined	Undefined	Operates	
★ RCU pins	DBO14	H	L	L	L	Undefined	Undefined
	DBO13	L	L	L	L	Undefined	Undefined
	DBO12 to DBO5	Undefined	Retained	Retained	Retained	Undefined	Undefined
	DBO4	L	L	L	H	Undefined	Undefined
	DBO3 to DBO1	L	L	L	L	Undefined	Undefined
	DBO0	L	Retained	Retained	Retained	Undefined	Undefined
	DBB15 to DBB0	Undefined	Retained	Retained	Retained	Undefined	Undefined
Peripheral evaluation chip mode pins	EVIEN	Undefined	L	L	L	Undefined	Undefined
	EVOEN	Undefined	L	L	L	Undefined	Undefined
	EVINTRQ	L	Retained	Retained	Operates	Undefined	Undefined
	EVINTLV6 to EVINTLV0	Undefined	Retained	Retained	Operates	Undefined	Undefined
	EVAD15 to EVAD0	Undefined	Retained	Retained	Undefined	Undefined	Undefined
	EVLKRT	Undefined	Retained	Retained	Undefined	Undefined	Undefined
Test mode pins	TBO34 to TBO0	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Operates
	BUNRIOUT	L	L	L	L	H	H
	TESEN	L	L	L	L	L	Operates
	VPTCLK	L	L	L	L	L	Operates
	PHTDIN1, PHTDIN0	L	L	L	L	L	Operates
	VPRESZ	L	H	H	H	Undefined	Undefined
	PHTEST	L	L	L	L	L	Operates
	TMODE1, TMODE0	L	L	L	L	L	Operates
TBREDZ	H	H	H	H	H	Operates	

Note When a low level is input to the DCRESZ pin and an external clock is input to the VBCLK pin.

Remark L: Low-level output
H: High-level output
Retained: Retains status immediately before
Hi-Z: High-impedance

CHAPTER 3 CPU

The CPU of the NU85E, which is based on a RISC architecture, executes almost all instructions in one clock cycle due to its five-stage pipeline control.

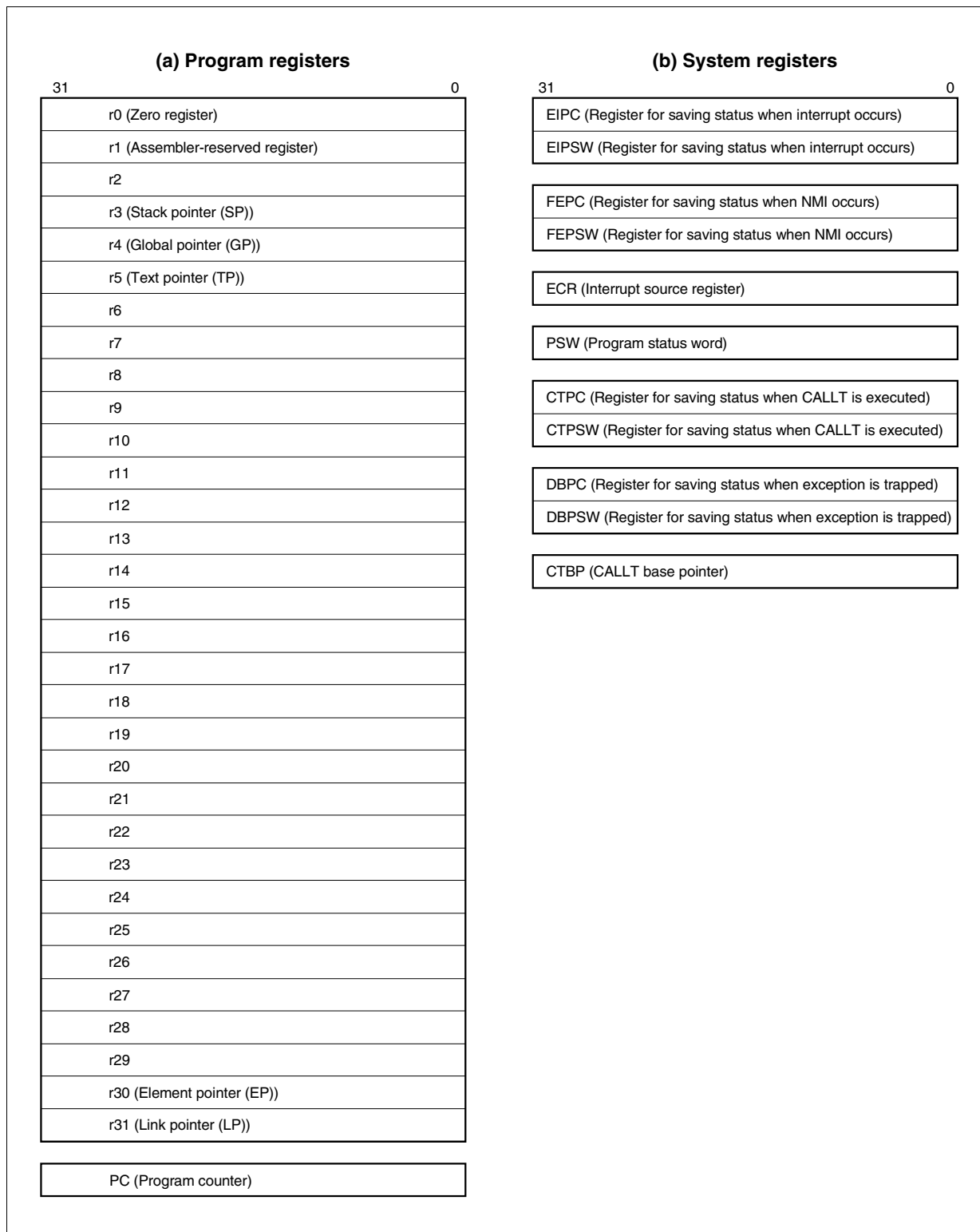
3.1 Features

- Advanced 32-bit architecture for embedded control
 - Number of instructions: 83
 - Number of 32-bit general-purpose registers: 32
 - Load/store instructions having long/short format
 - Three-operand instructions
 - Five-stage pipeline structure with one-clock pitch
 - Register/flag hazard interlock supported by hardware
 - Memory space
 - Program area: 64 MB linear address space
 - Data area: 4 GB linear address space
- Instruction set suited to various application fields
 - Saturated calculation instructions
 - Bit manipulation instructions (set, clear, not, test)
 - Multiplication can be performed in 1 or 2 clocks due to on-chip hardware multiplier
 - 16 bits \times 16 bits \rightarrow 32 bits
 - 32 bits \times 32 bits \rightarrow 32 bits or 64 bits
- RCU interface function

3.2 Registers

The CPU registers can be classified into program registers, which are used by programs, and system registers, which are used to control the execution environment. All registers are 32-bit registers.

Figure 3-1. List of CPU Registers



3.2.1 Program registers

The program registers include the general-purpose registers (r0 to r31) and the program counter (PC).

Table 3-1. List of Program Registers

Program Register	Name	Function
General-purpose register	r0	Zero register (always holds zero)
	r1	Assembler-reserved register (used as a working register for address generation)
	r2	Address/data variable register (when this register is not used by the real-time OS)
	r3	Stack pointer (used to generate a stack frame when a function is called)
	r4	Global pointer (used to access a global variable of the data area)
	r5	Text pointer (used as the register indicating the beginning of the text area (area for locating program code))
	r6 to r29	Registers for address/data variables
	r30	Element pointer (used as the base pointer for address generation when accessing memory)
	r31	Link pointer (used when the compiler calls a function)
Program counter	PC	Holds instruction address during program execution

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

(1) General-purpose registers

The 32 registers r0 to r31 are provided as general-purpose registers. All of these registers can be used for data variables or address variables.

However, take note of the following points when using the r0 to r5, r30, and r31 registers.

(a) r0, r30

These registers are implicitly used by instructions.

r0, which is a register that always holds 0, is used by operations that use 0, or in 0-offset addressing.

r30 is used as a base pointer when accessing memory by the SLD and SST instructions.

(b) r1, r3 to r5, r31

These registers are implicitly used by the assembler and C compiler.

The contents of these registers must be saved before they are used so that the contents are not destroyed, and the original contents must be returned after use.

(c) r2

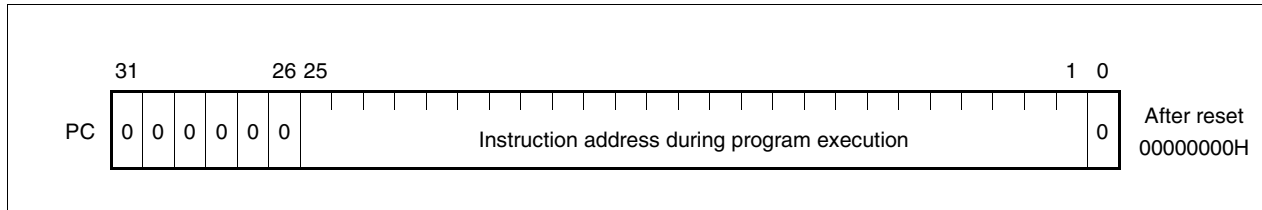
This register may be used by the real-time OS.

When not being used by the real-time OS, r2 can be used as an address variable or data variable register.

(2) Program counter

This register holds the instruction address during program execution. The lower 26 bits are valid, and bits 31 to 26 are reserved for future function expansion (fixed at 0). If a carry from bit 25 to bit 26 occurs, it is ignored. Also, bit 0 is fixed at 0, and no branching to an odd address can be performed.

Figure 3-2. Program Counter (PC)



3.2.2 System registers

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

To read from or write to these system registers, specify the system register number (see **Table 3-2**) indicated by the system register load or store instruction (LDSR or STSR instruction).

Table 3-2. List of System Registers

Register No.	Name		Operation	Whether or Not Operand Can be Specified	
				LDSR Instruction	STSR Instruction
0	EIPC	Register for saving status when interrupt occurs ^{Note}	This register saves the value of the PC when a software exception or interrupt occurs.	Yes	Yes
1	EIPSW		This register saves the value of the PSW when a software exception or interrupt occurs.	Yes	Yes
2	FEPC	Register for saving status when NMI occurs	This register saves the value of the PC when a non-maskable interrupt (NMI) occurs.	Yes	Yes
3	FEPSW		This register saves the value of the PSW when a non-maskable interrupt (NMI) occurs.	Yes	Yes
4	ECR	Interrupt source register	This register holds information about the source when an exception or interrupt occurs. The exception code of a non-maskable interrupt (NMI) is set in the higher 16 bits of this register (FECC). The exception code of an exception or maskable interrupt is set in the lower 16 bits (EICC) (See Figure 3-3).	No	Yes
5	PSW	Program status word	This is a collection of flags indicating the program status (instruction execution result) or CPU status (See Figure 3-4).	Yes	Yes
16	CTPC	Register for saving status when CALLT is executed	This register saves the value of the PC when a CALLT instruction is executed.	Yes	Yes
17	CTPSW		This register saves the value of the PSW when a CALLT instruction is executed.	Yes	Yes
18	DBPC	Register for saving status when exception is trapped	This register saves the value of the PC when an exception trap is generated due to the detection of an illegal instruction code.	No	Yes
19	DBPSW		This register saves the value of the PSW when an exception trap is generated due to the detection of an illegal instruction code.	No	Yes
20	CTBP	CALLT base pointer	This is used to specify the table address or generate the target address.	Yes	Yes
6 to 15, 21 to 31	Reserved numbers for future function expansion (if these are accessed, operation is not guaranteed).			No	No

Remark Yes: Access enabled No: Access disabled

Note Since there is only one set of these registers, their contents must be saved by the program when multiple interrupts are permitted.

Caution When interrupt servicing is performed and control is returned by the RETI instruction after bit 0 of the EIPC, FEPC, or CTPC had been set (1) by the LDSR instruction, bit 0 is ignored (because bit 0 of the PC is fixed at 0). When setting a value in EIPC, FEPC, or CTPC, set an even value (bit 0 = 0) as long as there is no specific reason not to.

Figure 3-3. Interrupt Source Register (ECR)

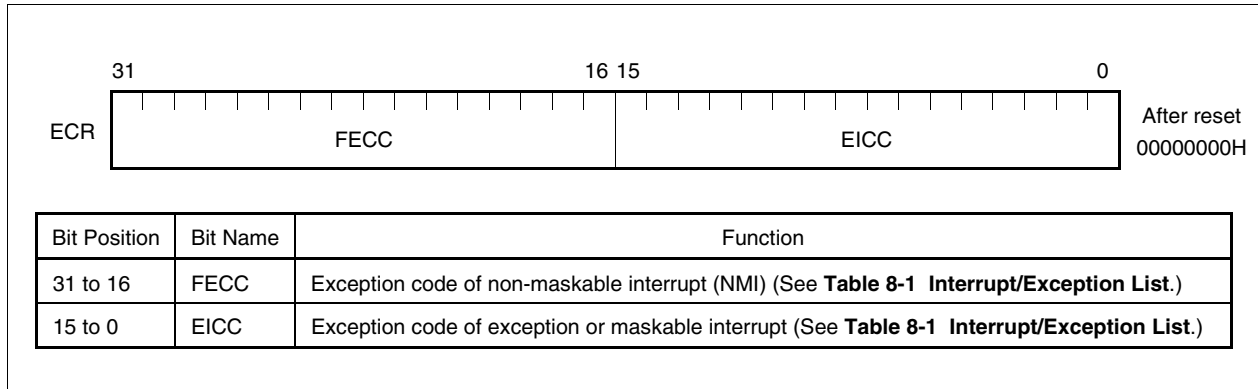
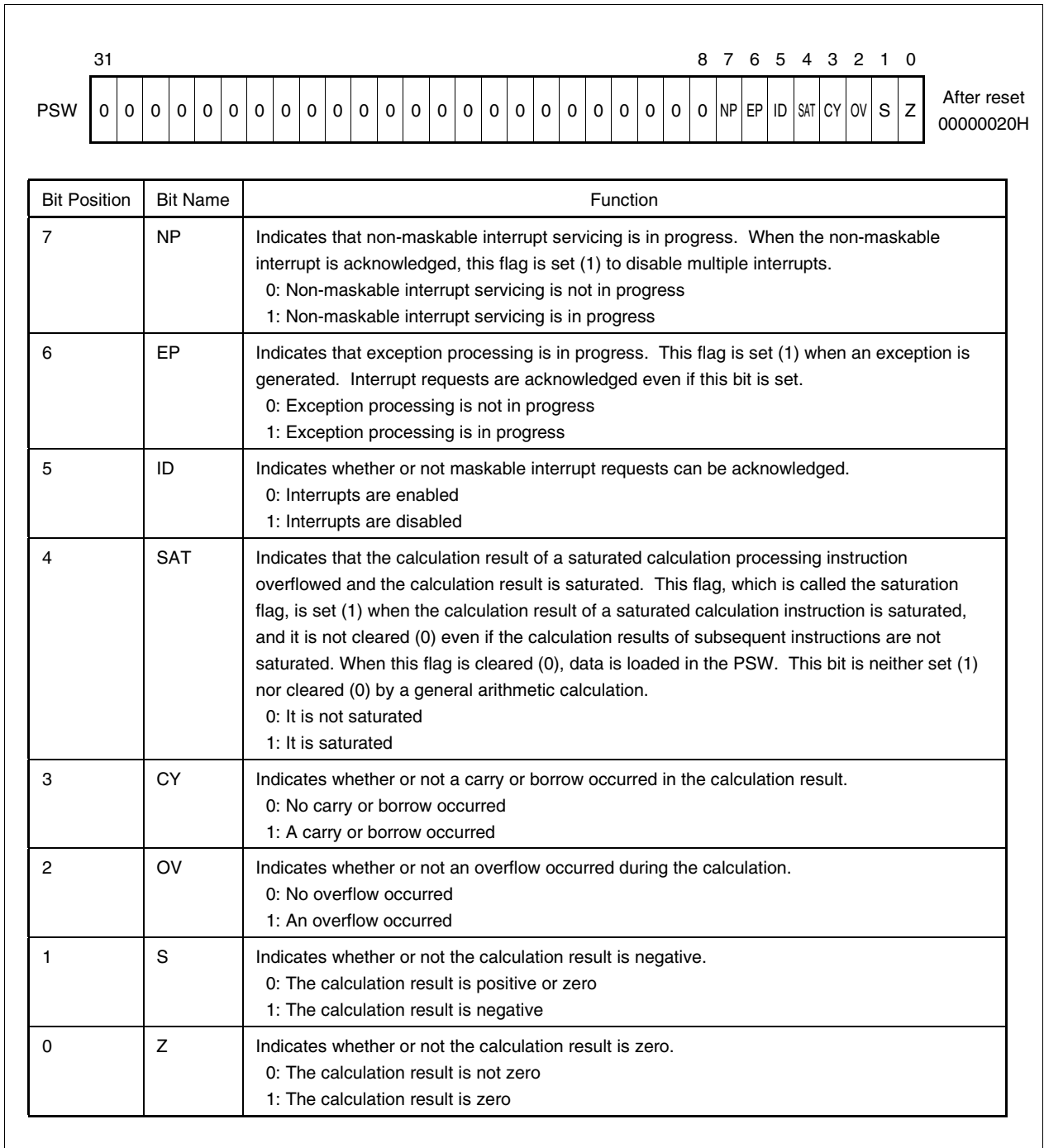


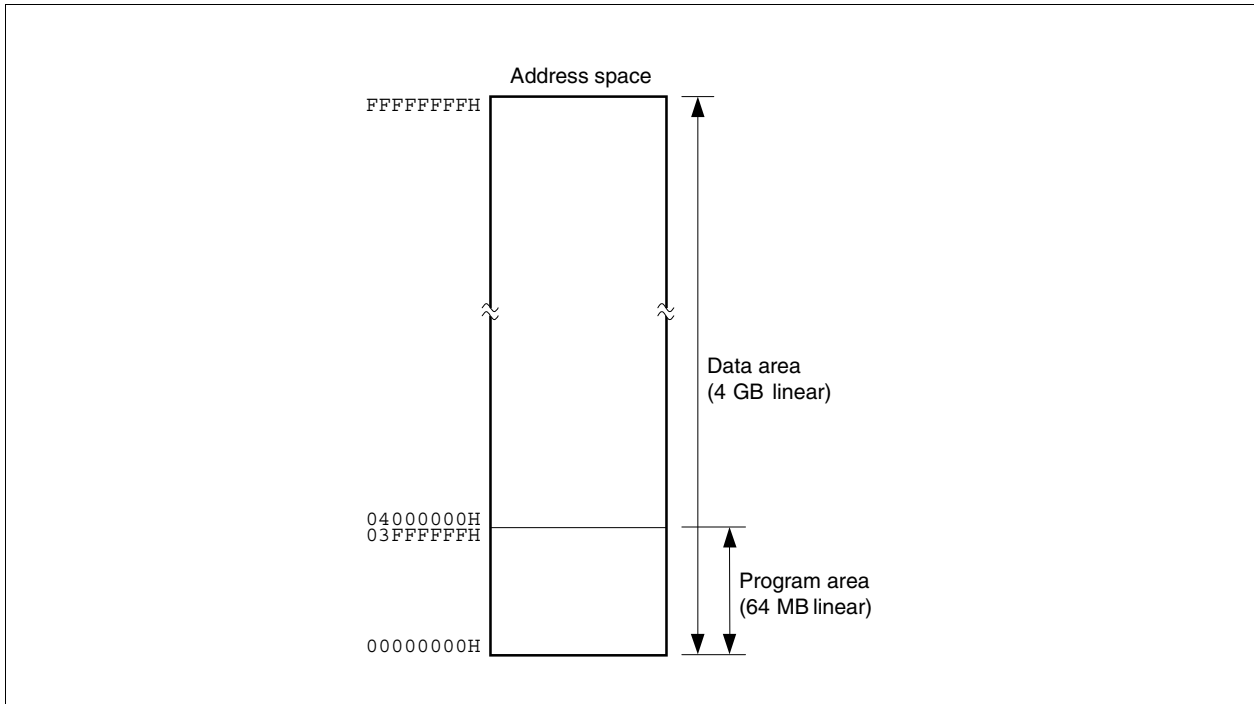
Figure 3-4. Program Status Word (PSW)



3.3 Address Space

The CPU of the NU85E supports a linear address space with a maximum size of 4 GB. Memory and I/O are located in this address space (memory mapped I/O method).

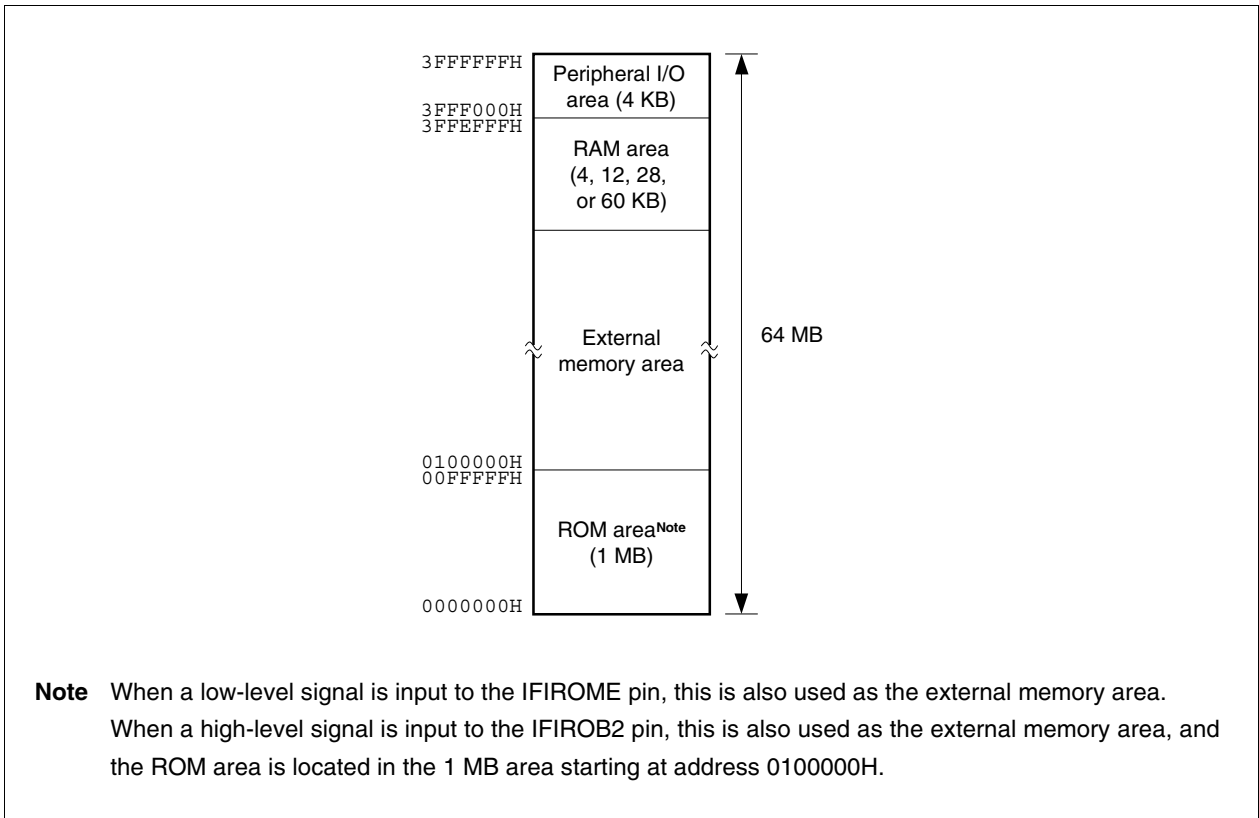
Figure 3-5. Address Space



3.3.1 Program area

For instruction addressing, the CPU of the NU85E supports a linear address space (program area) with a maximum size of 64 MB.

Figure 3-6. Program Area



3.3.2 Data area

For operand addressing (data access), the CPU of the NU85E supports a linear address space (data area) with a maximum size of 4 GB.

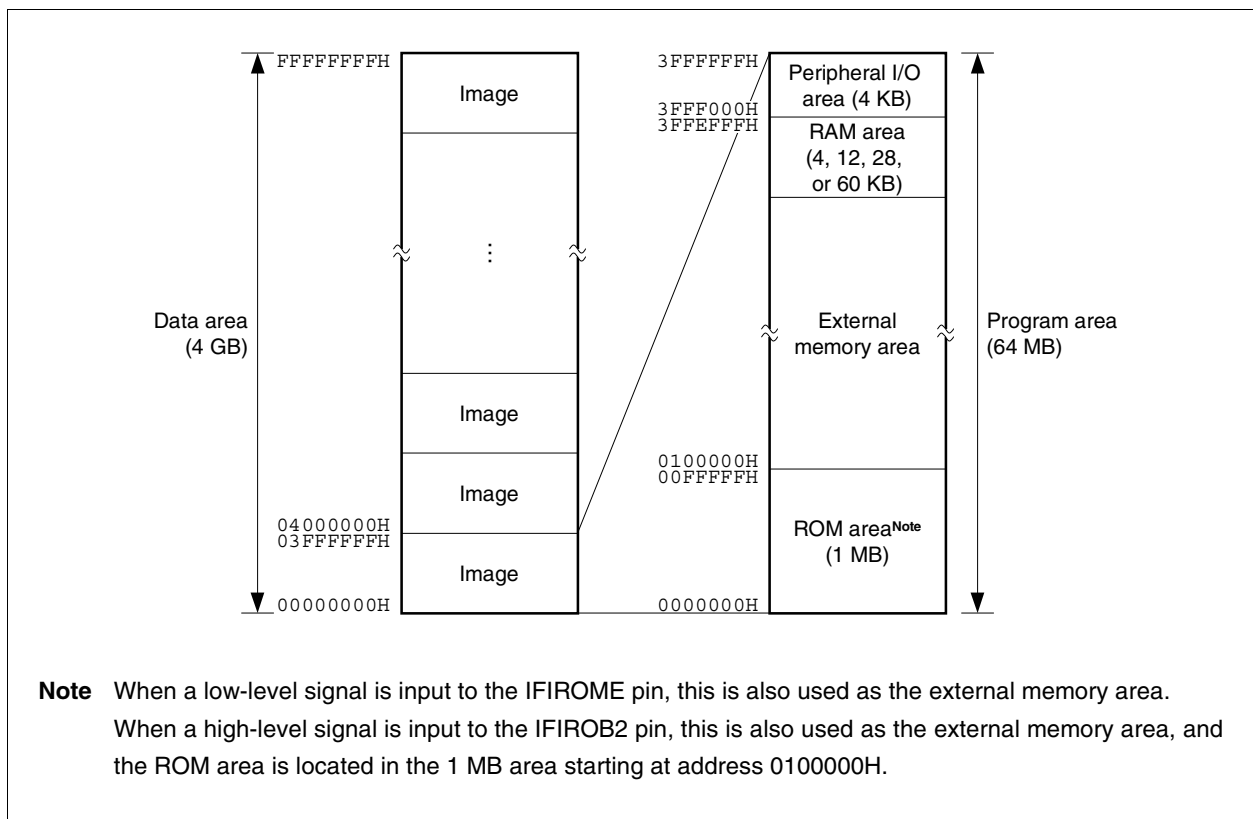
The ROM, RAM, and peripheral I/O areas are each located in 64 MB or 256 MB address spaces. The size setting is selected according to the input level to the IFID256 pin.

(1) 64 MB mode

When a low-level signal is input to the IFID256 pin, the data area is set to 64 MB mode.

In this mode, the 64 MB physical address space can be viewed as 64 images in the 4 GB address space. That is, the same 64 MB physical address space is accessed regardless of the values of bits 31 to 26 of the CPU address.

Figure 3-7. Data Area (64 MB Mode)



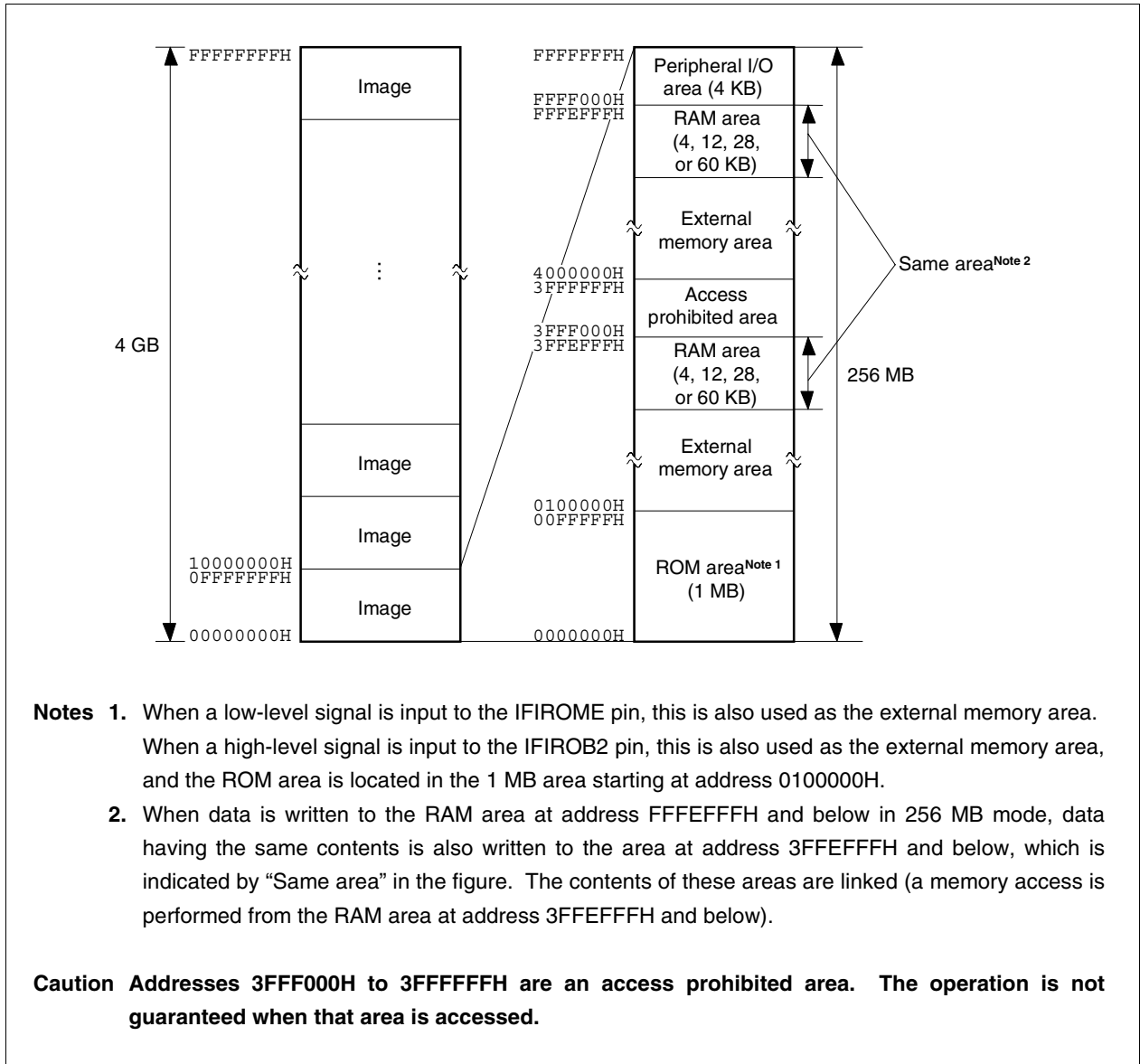
Note When a low-level signal is input to the IFIROME pin, this is also used as the external memory area. When a high-level signal is input to the IFIROB2 pin, this is also used as the external memory area, and the ROM area is located in the 1 MB area starting at address 0100000H.

(2) 256 MB mode

When a high-level signal is input to the IFID256 pin, the data area is set to 256 MB mode.

In this mode, the 256 MB physical address space can be viewed as 16 images in the 4 GB address space. That is, the same 256 MB physical address space is accessed regardless of the values of bits 31 to 28 of the CPU address.

Figure 3-8. Data Area (256 MB Mode)



- Notes**
1. When a low-level signal is input to the IFIROME pin, this is also used as the external memory area. When a high-level signal is input to the IFIROB2 pin, this is also used as the external memory area, and the ROM area is located in the 1 MB area starting at address 0100000H.
 2. When data is written to the RAM area at address FFF0000H and below in 256 MB mode, data having the same contents is also written to the area at address 3FF0000H and below, which is indicated by "Same area" in the figure. The contents of these areas are linked (a memory access is performed from the RAM area at address 3FF0000H and below).

Caution Addresses 3FFF000H to 3FFFFFFH are an access prohibited area. The operation is not guaranteed when that area is accessed.

3.4 Areas

3.4.1 ROM area

If a high level is input to the IFIROME pin, the area of ROM that can be accessed when ROM is connected to VFB is set.

(1) ROM relocation function

A 1 MB area at addresses 00000000H to 000FFFFFFH or addresses 00100000H to 001FFFFFFH is reserved as the ROM area.

The area where it is to be located is selected according to the input level to the IFIROB2 pin.

(2) Interrupt/exception table

The NU85E increases the interrupt response speed by assigning fixed jump destination addresses corresponding to interrupts or exceptions.

This set of jump destination addresses is called the interrupt/exception table and is located at address 00000000H and following. When an interrupt/exception request is acknowledged, processing jumps to the jump destination address and the program that is written in memory beginning at that address is executed.

Remark When address 00000000H is set in the external memory area, prepare the jump destination address for jumping to the reset routine at address 00000000H of the external memory.

Figure 3-9. ROM Area

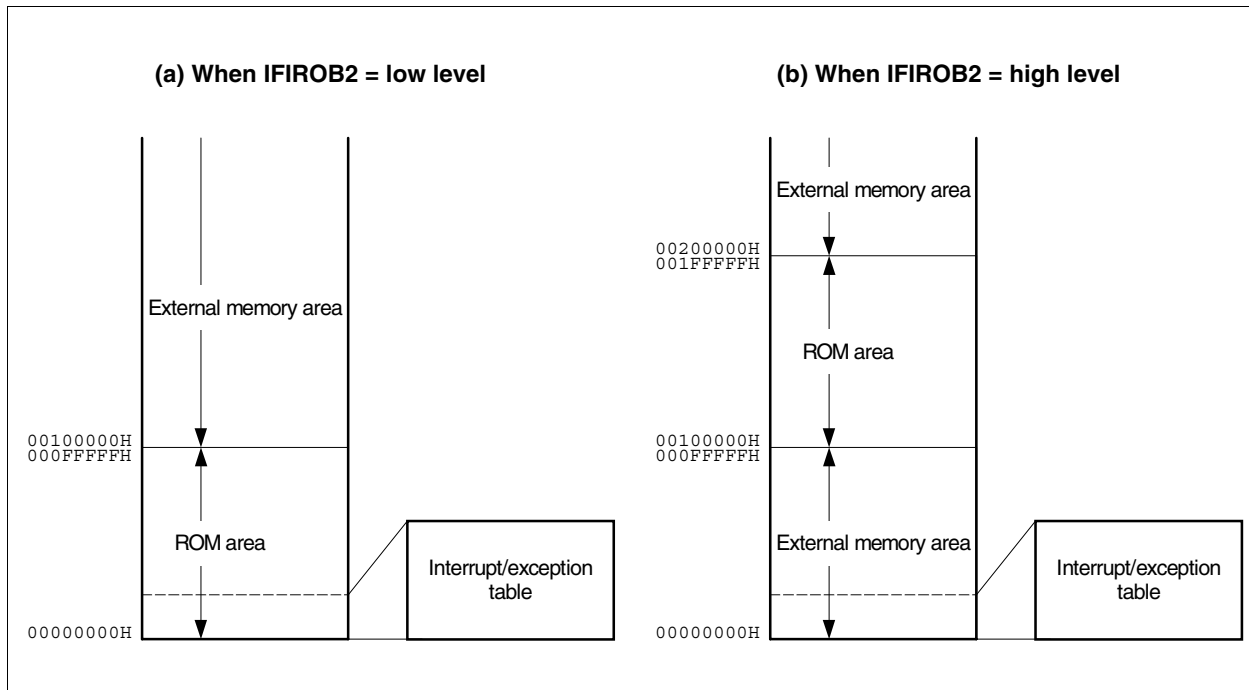


Table 3-3. Interrupt/Exception Table

Starting Address	Interrupt/Exception Source	Starting Address	Interrupt/Exception Source	Starting Address	Interrupt/Exception Source
00000000H	RESET	0000190H	INT17	00000310H	INT41
00000010H	NMI0	00001A0H	INT18	00000320H	INT42
00000020H	NMI1	00001B0H	INT19	00000330H	INT43
00000030H	NMI2	00001C0H	INT20	00000340H	INT44
00000040H	TRAP _{0n} (n = 0 to FH)	00001D0H	INT21	00000350H	INT45
00000050H	TRAP _{1n} (n = 0 to FH)	00001E0H	INT22	00000360H	INT46
00000060H	ILGOP	00001F0H	INT23	00000370H	INT47
00000080H	INT0	0000200H	INT24	00000380H	INT48
00000090H	INT1	0000210H	INT25	00000390H	INT49
000000A0H	INT2	0000220H	INT26	000003A0H	INT50
000000B0H	INT3	0000230H	INT27	000003B0H	INT51
000000C0H	INT4	0000240H	INT28	000003C0H	INT52
000000D0H	INT5	0000250H	INT29	000003D0H	INT53
000000E0H	INT6	0000260H	INT30	000003E0H	INT54
000000F0H	INT7	0000270H	INT31	000003F0H	INT55
00000100H	INT8	0000280H	INT32	00000400H	INT56
00000110H	INT9	0000290H	INT33	00000410H	INT57
00000120H	INT10	00002A0H	INT34	00000420H	INT58
00000130H	INT11	00002B0H	INT35	00000430H	INT59
00000140H	INT12	00002C0H	INT36	00000440H	INT60
00000150H	INT13	00002D0H	INT37	00000450H	INT61
00000160H	INT14	00002E0H	INT38	00000460H	INT62
00000170H	INT15	00002F0H	INT39	00000470H	INT63
00000180H	INT16	0000300H	INT40	–	–

Remark For the sources of interrupts or exceptions, see Table 8-1 Interrupt/Exception List.

3.4.2 RAM area

In 64 MB mode, the area at address 3FFEFFFH and below is reserved as the area for RAM connected to the VDB. In 256 MB mode, the address at FFFEFFFH and below is reserved.

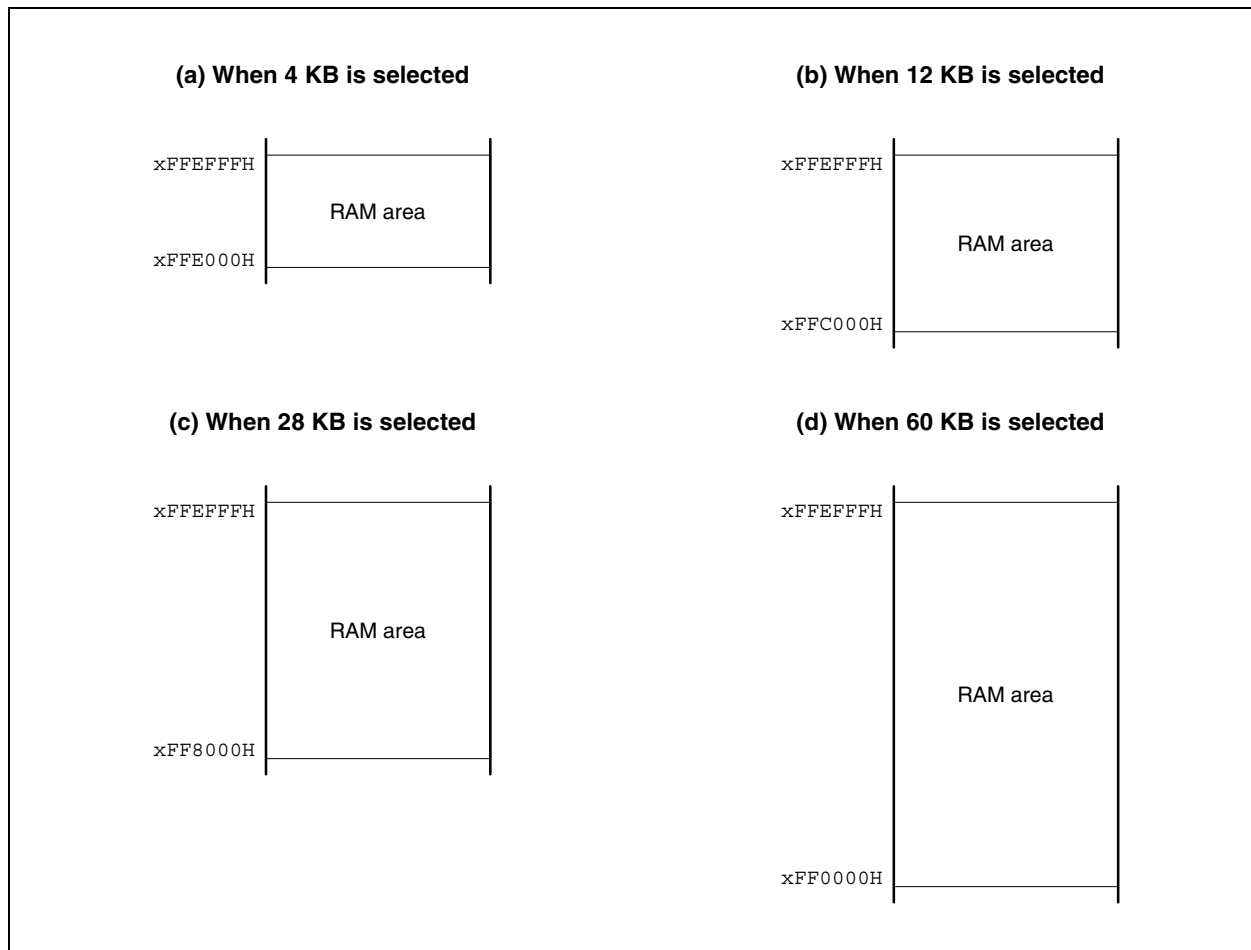
The size of the RAM area, which can be selected from among 4 KB, 12 KB, 28 KB, and 60 KB, is set according to the input levels to the IFIRA64, IFIRA32, and IFIRA16 pins.

Table 3-4. RAM Area Size Settings

IFIRA64	IFIRA32	IFIRA16	RAM Area Size
0	0	0	4 KB
0	0	1	12 KB
0	1	Arbitrary	28 KB
1	Arbitrary	Arbitrary	60 KB

Remark 0: low-level input 1: high-level input

Figure 3-10. RAM Area



Set as follows if the size of the RAM area to be used is other than 4 KB, 12 KB, 28 KB, or 60 KB.

(a) RAM area size = 0 KB (RAM-less)

Set the RAM area size to 4 KB and handle the VDB pins as indicated in **2.3 Recommended Connection of Unused Pins**.

(b) 0 KB < RAM area size < 4 KB

Set the RAM area size to 4 KB and use from the lower side addresses as RAM area.

(c) 4 KB < RAM area size < 12 KB

Set the RAM area size to 12 KB and use from the lower side addresses as RAM area.

(d) 12 KB < RAM area size < 28 KB

Set the RAM area size to 28 KB and use from the lower side addresses as RAM area.

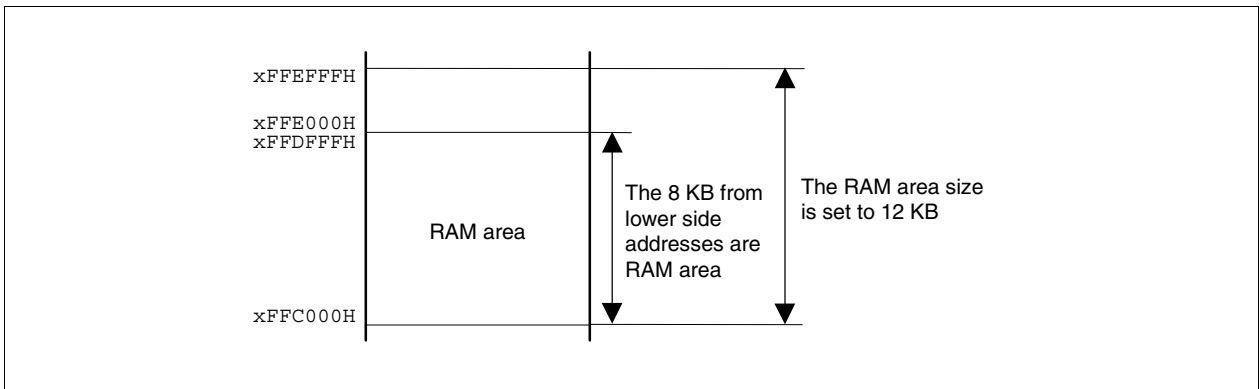
(e) 28 KB < RAM area size < 60 KB

Set the RAM area size to 60 KB and use from the lower side addresses as RAM area.

(f) 60 KB < RAM area size

A RAM area size exceeding 60 KB cannot be set.

Example Memory map when 8 KB RAM is used.

**3.4.3 Peripheral I/O area**

In 64 MB mode, the area at address `3FFFFFFH` and below is reserved as a peripheral I/O area. In 256 MB mode, the address at `FFFFFFFH` and below is reserved.

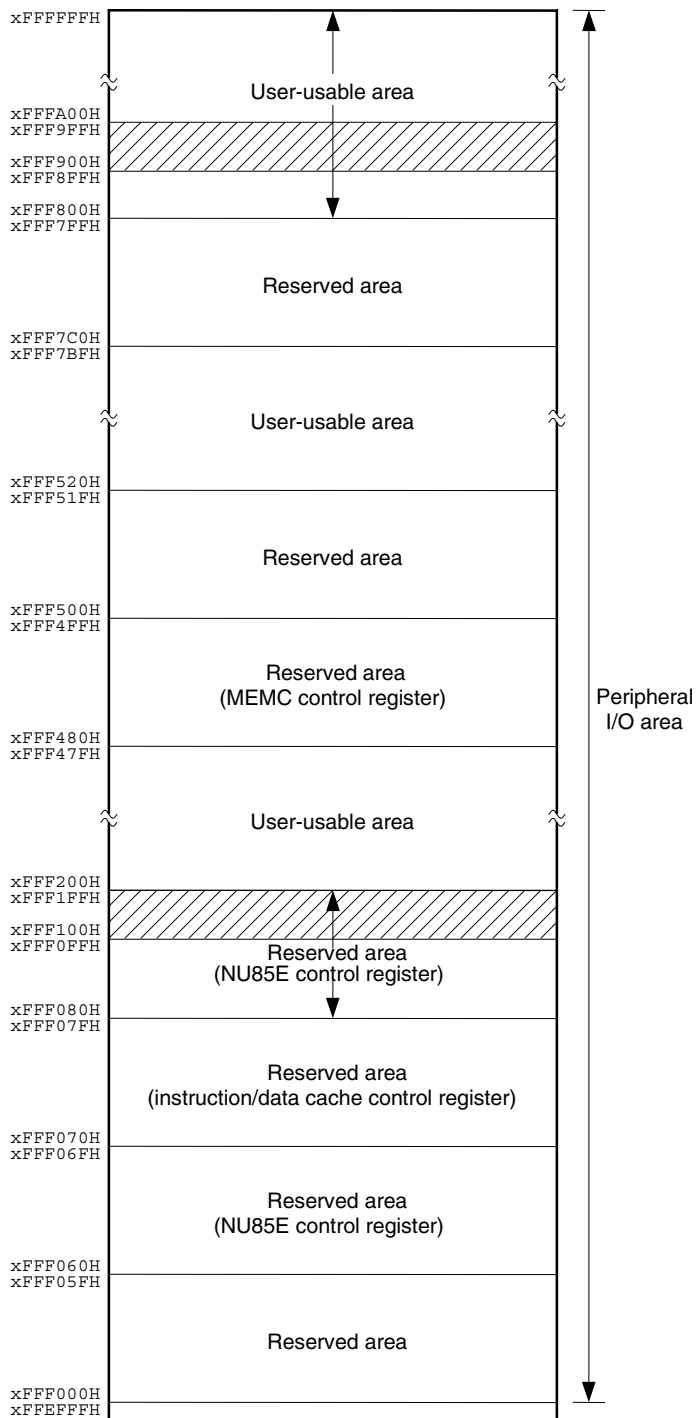
Peripheral I/O registers to which functions have been assigned such as status monitoring or specification of the operating mode of the NU85E, memory controller (MEMC), or instruction/data cache are located in this area.

For information about assigned registers, see **3.5 Peripheral I/O Registers**.

Caution User-defined addresses must be assigned to the following areas only (user-usable area); all other addresses are reserved and cannot therefore be used.

- `xFFF200H` to `xFFF47FH`
- `xFFF520H` to `xFFF7BFH`
- `xFFF800H` to `xFFFFFFFFH`

Figure 3-11. Peripheral I/O Area



Remark Interrupts are not acknowledged between the store instruction and the subsequent instruction for the shaded area (refer to 8.7 Periods When Interrupts Cannot Be Acknowledged).

3.4.4 External memory area

Access to the external memory area is made using the VDCSZ7 to VDCSZ0 signals assigned to each bank (see **4.2 Memory Banks**).

The “programmable peripheral I/O area”, which is independent of the peripheral I/O area, is also assigned to this area (see **4.4 Programmable Peripheral I/O Area Selection Function**).

Caution ROM, RAM, and peripheral I/O areas cannot be accessed as external memory areas.

3.5 Peripheral I/O Registers

- (1) Only the lower 12 bits of a 32-bit address are used for register address decoding, after being allocated to the 4 KB area of xxxxx000H to xxxxxFFFH.
- (2) The lowest bit of the address is not decoded. Therefore, when the register of an odd address ($2n + 1$ address) is byte-accessed, the register of an even address ($2n$) will be accessed.
- (3) Although word-accessible registers do not exist in the NU85E, halfword access using the lower and higher bits (in that order and ignoring the lowest 2) of a word area can be made twice to enable word access.
- (4) When byte-accessible registers are halfword-accessed, the higher 8 bits become undefined in a read operation, and the lower 8 bits of data are written to a register in a write operation.
- (5) Registers other than those that control the NU85E are incorporated in each macro (MEMC, instruction/data cache).

3.5.1 NU85E control registers

(1/4)

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF060H	Chip area select control register 0	CSC0	R/W			√	2C11H
FFFFF062H	Chip area select control register 1	CSC1	R/W			√	2C11H
FFFFF064H	Peripheral I/O area select control register	BPC	R/W			√	0000H
FFFFF066H	Bus size configuration register	BSC	R/W			√	0000H/ 5555H/ AAAAH
FFFFF068H	Endian configuration register	BEC	R/W			√	0000H
FFFFF06AH	Cache configuration register	BHC	R/W			√	0000H
FFFFF06EH	NPB strobe wait control register	VSWC	R/W	√	√		77H
FFFFF080H	DMA source address register 0L	DSA0L	R/W			√	Undefined
FFFFF082H	DMA source address register 0H	DSA0H	R/W			√	Undefined
FFFFF084H	DMA destination address register 0L	DDA0L	R/W			√	Undefined
FFFFF086H	DMA destination address register 0H	DDA0H	R/W			√	Undefined
FFFFF088H	DMA source address register 1L	DSA1L	R/W			√	Undefined
FFFFF08AH	DMA source address register 1H	DSA1H	R/W			√	Undefined
FFFFF08CH	DMA destination address register 1L	DDA1L	R/W			√	Undefined
FFFFF08EH	DMA destination address register 1H	DDA1H	R/W			√	Undefined
FFFFF090H	DMA source address register 2L	DSA2L	R/W			√	Undefined
FFFFF092H	DMA source address register 2H	DSA2H	R/W			√	Undefined
FFFFF094H	DMA destination address register 2L	DDA2L	R/W			√	Undefined
FFFFF096H	DMA destination address register 2H	DDA2H	R/W			√	Undefined
FFFFF098H	DMA source address register 3L	DSA3L	R/W			√	Undefined
FFFFF09AH	DMA source address register 3H	DSA3H	R/W			√	Undefined
FFFFF09CH	DMA destination address register 3L	DDA3L	R/W			√	Undefined
FFFFF09EH	DMA destination address register 3H	DDA3H	R/W			√	Undefined
FFFFF0C0H	DMA 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
FFFFF0F0H	DMA disable status register	DDIS	R	√	√		00H

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF0F2H	DMA restart register	DRST	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
FFFFF110H	Interrupt control register 0	PIC0	R/W	√	√		47H
FFFFF112H	Interrupt control register 1	PIC1	R/W	√	√		47H
FFFFF114H	Interrupt control register 2	PIC2	R/W	√	√		47H
FFFFF116H	Interrupt control register 3	PIC3	R/W	√	√		47H
FFFFF118H	Interrupt control register 4	PIC4	R/W	√	√		47H
FFFFF11AH	Interrupt control register 5	PIC5	R/W	√	√		47H
FFFFF11CH	Interrupt control register 6	PIC6	R/W	√	√		47H
FFFFF11EH	Interrupt control register 7	PIC7	R/W	√	√		47H
FFFFF120H	Interrupt control register 8	PIC8	R/W	√	√		47H
FFFFF122H	Interrupt control register 9	PIC9	R/W	√	√		47H
FFFFF124H	Interrupt control register 10	PIC10	R/W	√	√		47H
FFFFF126H	Interrupt control register 11	PIC11	R/W	√	√		47H
FFFFF128H	Interrupt control register 12	PIC12	R/W	√	√		47H
FFFFF12AH	Interrupt control register 13	PIC13	R/W	√	√		47H
FFFFF12CH	Interrupt control register 14	PIC14	R/W	√	√		47H
FFFFF12EH	Interrupt control register 15	PIC15	R/W	√	√		47H
FFFFF130H	Interrupt control register 16	PIC16	R/W	√	√		47H
FFFFF132H	Interrupt control register 17	PIC17	R/W	√	√		47H
FFFFF134H	Interrupt control register 18	PIC18	R/W	√	√		47H
FFFFF136H	Interrupt control register 19	PIC19	R/W	√	√		47H
FFFFF138H	Interrupt control register 20	PIC20	R/W	√	√		47H
FFFFF13AH	Interrupt control register 21	PIC21	R/W	√	√		47H
FFFFF13CH	Interrupt control register 22	PIC22	R/W	√	√		47H
FFFFF13EH	Interrupt control register 23	PIC23	R/W	√	√		47H
FFFFF140H	Interrupt control register 24	PIC24	R/W	√	√		47H

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF142H	Interrupt control register 25	PIC25	R/W	√	√		47H
FFFFFF144H	Interrupt control register 26	PIC26	R/W	√	√		47H
FFFFFF146H	Interrupt control register 27	PIC27	R/W	√	√		47H
FFFFFF148H	Interrupt control register 28	PIC28	R/W	√	√		47H
FFFFFF14AH	Interrupt control register 29	PIC29	R/W	√	√		47H
FFFFFF14CH	Interrupt control register 30	PIC30	R/W	√	√		47H
FFFFFF14EH	Interrupt control register 31	PIC31	R/W	√	√		47H
FFFFFF150H	Interrupt control register 32	PIC32	R/W	√	√		47H
FFFFFF152H	Interrupt control register 33	PIC33	R/W	√	√		47H
FFFFFF154H	Interrupt control register 34	PIC34	R/W	√	√		47H
FFFFFF156H	Interrupt control register 35	PIC35	R/W	√	√		47H
FFFFFF158H	Interrupt control register 36	PIC36	R/W	√	√		47H
FFFFFF15AH	Interrupt control register 37	PIC37	R/W	√	√		47H
FFFFFF15CH	Interrupt control register 38	PIC38	R/W	√	√		47H
FFFFFF15EH	Interrupt control register 39	PIC39	R/W	√	√		47H
FFFFFF160H	Interrupt control register 40	PIC40	R/W	√	√		47H
FFFFFF162H	Interrupt control register 41	PIC41	R/W	√	√		47H
FFFFFF164H	Interrupt control register 42	PIC42	R/W	√	√		47H
FFFFFF166H	Interrupt control register 43	PIC43	R/W	√	√		47H
FFFFFF168H	Interrupt control register 44	PIC44	R/W	√	√		47H
FFFFFF16AH	Interrupt control register 45	PIC45	R/W	√	√		47H
FFFFFF16CH	Interrupt control register 46	PIC46	R/W	√	√		47H
FFFFFF16EH	Interrupt control register 47	PIC47	R/W	√	√		47H
FFFFFF170H	Interrupt control register 48	PIC48	R/W	√	√		47H
FFFFFF172H	Interrupt control register 49	PIC49	R/W	√	√		47H
FFFFFF174H	Interrupt control register 50	PIC50	R/W	√	√		47H
FFFFFF176H	Interrupt control register 51	PIC51	R/W	√	√		47H
FFFFFF178H	Interrupt control register 52	PIC52	R/W	√	√		47H
FFFFFF17AH	Interrupt control register 53	PIC53	R/W	√	√		47H
FFFFFF17CH	Interrupt control register 54	PIC54	R/W	√	√		47H
FFFFFF17EH	Interrupt control register 55	PIC55	R/W	√	√		47H
FFFFFF180H	Interrupt control register 56	PIC56	R/W	√	√		47H
FFFFFF182H	Interrupt control register 57	PIC57	R/W	√	√		47H
FFFFFF184H	Interrupt control register 58	PIC58	R/W	√	√		47H
FFFFFF186H	Interrupt control register 59	PIC59	R/W	√	√		47H
FFFFFF188H	Interrupt control register 60	PIC60	R/W	√	√		47H
FFFFFF18AH	Interrupt control register 61	PIC61	R/W	√	√		47H
FFFFFF18CH	Interrupt control register 62	PIC62	R/W	√	√		47H

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF18EH	Interrupt control register 63	PIC63	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

3.5.2 Memory controller (MEMC) control registers

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFF480H	Bus cycle type configuration register 0	BCT0	R/W			√	8888H/0000H
FFFFF482H	Bus cycle type configuration register 1	BCT1	R/W			√	8888H/0000H
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 setting wait control register	ASC	R/W			√	FFFFH
FFFFF48CH	Bus cycle period control register	BCP	R/W	√	√		80H/00H
FFFFF49AH	Page ROM configuration register	PRC	R/W			√	7000H
FFFFF4A0H	SDRAM configuration register 0	SCR0	R/W			√	0000H
FFFFF4A2H	SDRAM refresh control register 0	RFS0	R/W			√	0000H
FFFFF4A4H	SDRAM configuration register 1	SCR1	R/W			√	0000H
FFFFF4A6H	SDRAM refresh control register 1	RFS1	R/W			√	0000H
FFFFF4A8H	SDRAM configuration register 2	SCR2	R/W			√	0000H
FFFFF4AAH	SDRAM refresh control register 2	RFS2	R/W			√	0000H
FFFFF4ACH	SDRAM configuration register 3	SCR3	R/W			√	0000H
FFFFF4AEH	SDRAM refresh control register 3	RFS3	R/W			√	0000H
FFFFF4B0H	SDRAM configuration register 4	SCR4	R/W			√	0000H
FFFFF4B2H	SDRAM refresh control register 4	RFS4	R/W			√	0000H
FFFFF4B4H	SDRAM configuration register 5	SCR5	R/W			√	0000H
FFFFF4B6H	SDRAM refresh control register 5	RFS5	R/W			√	0000H
FFFFF4B8H	SDRAM configuration register 6	SCR6	R/W			√	0000H
FFFFF4BAH	SDRAM refresh control register 6	RFS6	R/W			√	0000H
FFFFF4BCH	SDRAM configuration register 7	SCR7	R/W			√	0000H
FFFFF4BEH	SDRAM refresh control register 7	RFS7	R/W			√	0000H

3.5.3 Instruction cache control registers

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
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

- Notes**
1. This value becomes 0003H when the reset signal is active, and tag initialization starts automatically. The value changes to 0000H upon the completion of tag initialization.
 2. This value becomes 03H when the reset signal is active, and tag initialization starts automatically. The value changes to 00H upon the completion of tag initialization.

3.5.4 Data cache control registers

Address	Register Name	Symbol	R/W	Bit Units for Manipulation			After Reset
				1 Bit	8 Bits	16 Bits	
FFFFFF078H	Data cache control register	DCC	R/W			√	0003H ^{Note}
FFFFFF07CH	Data cache data configuration register	DCD	R/W			√	Undefined

- Note** This value becomes 0003H when the reset signal is active, and tag initialization starts automatically. The value changes to 0000H upon the completion of tag initialization.

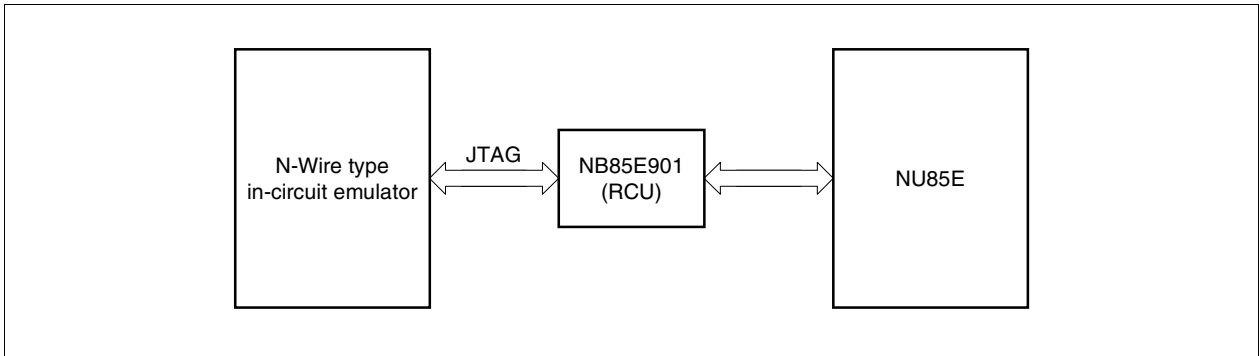
3.6 RCU Interface

3.6.1 Outline

The RCU interface is a bus interface for connecting the NU85E to the NB85E901 (run control unit (RCU)).

The RCU communicates with an N-Wire type in-circuit emulator by using JTAG and executes debug processing.

Figure 3-12. Connection of NU85E and N-Wire Type In-Circuit Emulator via RCU



3.6.2 On-chip debugging

By connecting the RCU to the NU85E, on-chip debugging (an on-board debug function) can be realized. The CPU of the NU85E is equipped with a breakpoint function, which can detect breakpoints based on the execution address, access address, access data, range (address mask), or 2-stage sequential execution, as well as a break interrupt function operable by external port input. Connection of the RCU to the NU85E allows not only debugging using these functions, but also makes possible the employment of a background monitor JTAG system ROM emulator and an N-Wire type in-circuit emulator.

CHAPTER 4 BCU

The bus control unit (BCU), which operates as a bus master on the VSB, controls the on-chip bus bridge (BBR), test interface control unit (TIC), and peripheral macros (bus slaves) such as the external memory controller (MEMC) connected to the VSB.

4.1 Features

- 32-bit independent I/O separated data bus
- One bus clock transfer between consecutive clock falling edges
- Data transfer in 8-bit, 16-bit, or 32-bit units on a 32-bit bus by means of the bus size function
- Bus arbitration for a multi-master system
- Programmable chip select function
- Programmable peripheral I/O area select function
- Endian setting function

4.2 Memory Banks

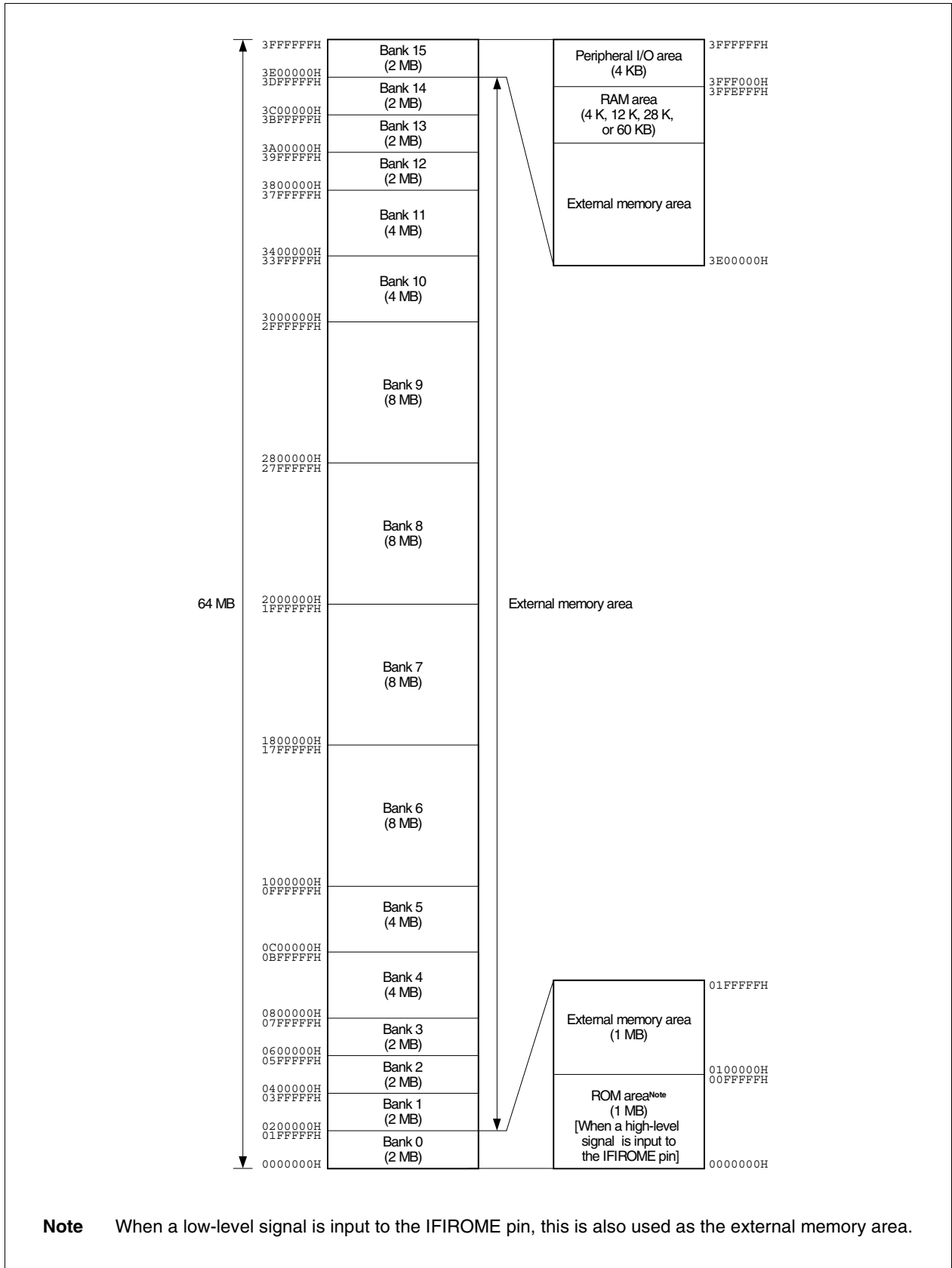
The data area is subdivided into multiple units called banks.

The BCU makes bus size, endian, and cache settings in terms of units called “CSn area”, which are arbitrary combinations of banks.

“CSn area” settings are made based on the VDCSZn signals corresponding to each bank (n = 7 to 0).

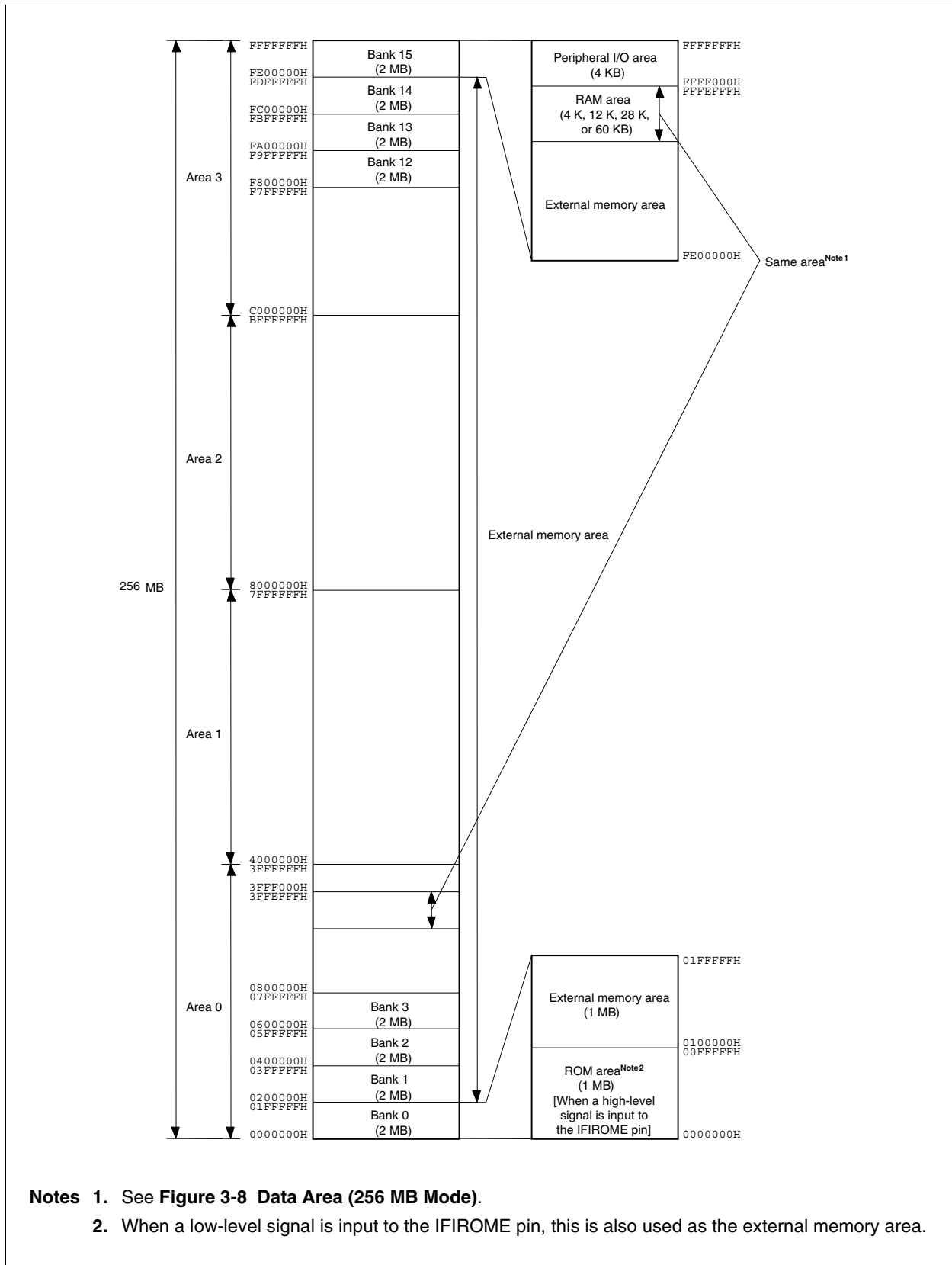
(1) Memory banks for 64 MB mode

The 64 MB data area is subdivided into memory banks with sizes of 2 MB, 4 MB, and 8 MB.



(2) Memory banks for 256 MB mode

The 256 MB data area is subdivided into four areas (area 0 to area 3), each of which contain memory banks of size 2 MB.



4.3 Programmable Chip Select Function

The VDCSZn signals corresponding to each bank of memory are set and the data area is subdivided into multiple CSn areas according to the chip area select control registers 0 and 1 (CSC0 and CSC1) (n = 7 to 0). The CSC0 and CSC1 registers can be read or written in 16-bit units.

When the VDCSZn signals for the same bank overlap due to the CSC0 and CSC1 register settings, the signal prioritization is as follows.

- VDCSZ0 > VDCSZ2 > VDCSZ1 > VDCSZ3
- VDCSZ7 > VDCSZ5 > VDCSZ6 > VDCSZ4

Figure 4-1. Chip Area Select Control Register 0 (CSC0)

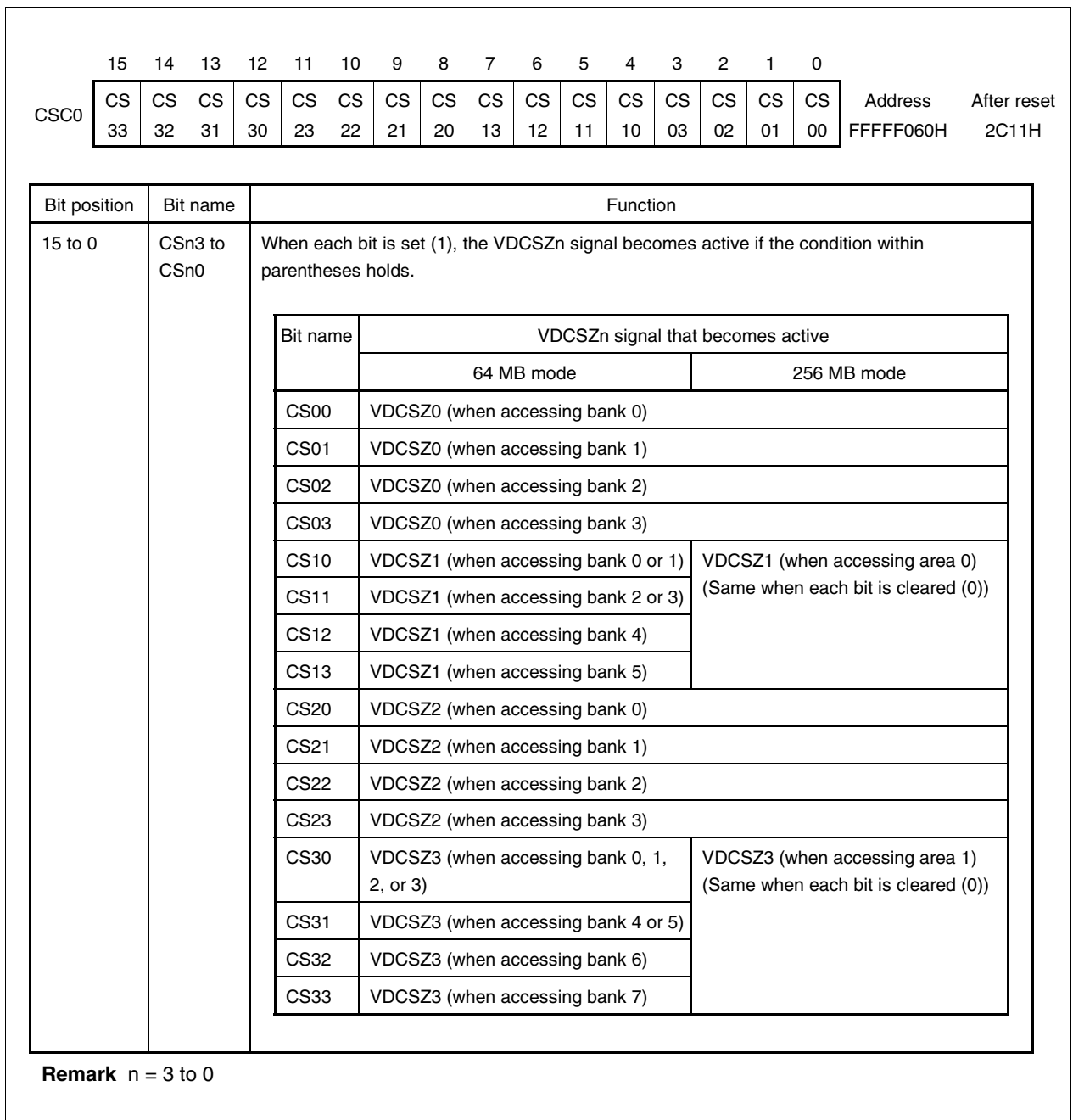


Figure 4-2. Chip Area Select Control Register 1 (CSC1)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	After reset
CSC1	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	CS	FFFFF062H	2C11H
	43	42	41	40	53	52	51	50	63	62	61	60	73	72	71	70		

Bit position	Bit name	Function																																															
15 to 0	CSn3 to CSn0	When each bit is set (1), the VDCSZn signal becomes active if the condition within parentheses holds. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th rowspan="2">Bit name</th> <th colspan="2">VDCSZn signal that becomes active</th> </tr> <tr> <th>64 MB mode</th> <th>256 MB mode</th> </tr> </thead> <tbody> <tr> <td>CS40</td> <td>VDCSZ4 (when accessing bank 12, 13, 14, or 15)</td> <td rowspan="4">VDCSZ4 (when accessing area 2) (Same when each bit is cleared (0))</td> </tr> <tr> <td>CS41</td> <td>VDCSZ4 (when accessing bank 10 or 11)</td> </tr> <tr> <td>CS42</td> <td>VDCSZ4 (when accessing bank 9)</td> </tr> <tr> <td>CS43</td> <td>VDCSZ4 (when accessing bank 8)</td> </tr> <tr> <td>CS50</td> <td colspan="2">VDCSZ5 (when accessing bank 15)</td> </tr> <tr> <td>CS51</td> <td colspan="2">VDCSZ5 (when accessing bank 14)</td> </tr> <tr> <td>CS52</td> <td colspan="2">VDCSZ5 (when accessing bank 13)</td> </tr> <tr> <td>CS53</td> <td colspan="2">VDCSZ5 (when accessing bank 12)</td> </tr> <tr> <td>CS60</td> <td>VDCSZ6 (when accessing bank 14 or 15)</td> <td rowspan="4">VDCSZ6 (when accessing area 3) (Same when each bit is cleared (0))</td> </tr> <tr> <td>CS61</td> <td>VDCSZ6 (when accessing bank 12 or 13)</td> </tr> <tr> <td>CS62</td> <td>VDCSZ6 (when accessing bank 11)</td> </tr> <tr> <td>CS63</td> <td>VDCSZ6 (when accessing bank 10)</td> </tr> <tr> <td>CS70</td> <td colspan="2">VDCSZ7 (when accessing bank 15)</td> </tr> <tr> <td>CS71</td> <td colspan="2">VDCSZ7 (when accessing bank 14)</td> </tr> <tr> <td>CS72</td> <td colspan="2">VDCSZ7 (when accessing bank 13)</td> </tr> <tr> <td>CS73</td> <td colspan="2">VDCSZ7 (when accessing bank 12)</td> </tr> </tbody> </table>	Bit name	VDCSZn signal that becomes active		64 MB mode	256 MB mode	CS40	VDCSZ4 (when accessing bank 12, 13, 14, or 15)	VDCSZ4 (when accessing area 2) (Same when each bit is cleared (0))	CS41	VDCSZ4 (when accessing bank 10 or 11)	CS42	VDCSZ4 (when accessing bank 9)	CS43	VDCSZ4 (when accessing bank 8)	CS50	VDCSZ5 (when accessing bank 15)		CS51	VDCSZ5 (when accessing bank 14)		CS52	VDCSZ5 (when accessing bank 13)		CS53	VDCSZ5 (when accessing bank 12)		CS60	VDCSZ6 (when accessing bank 14 or 15)	VDCSZ6 (when accessing area 3) (Same when each bit is cleared (0))	CS61	VDCSZ6 (when accessing bank 12 or 13)	CS62	VDCSZ6 (when accessing bank 11)	CS63	VDCSZ6 (when accessing bank 10)	CS70	VDCSZ7 (when accessing bank 15)		CS71	VDCSZ7 (when accessing bank 14)		CS72	VDCSZ7 (when accessing bank 13)		CS73	VDCSZ7 (when accessing bank 12)	
Bit name	VDCSZn signal that becomes active																																																
	64 MB mode	256 MB mode																																															
CS40	VDCSZ4 (when accessing bank 12, 13, 14, or 15)	VDCSZ4 (when accessing area 2) (Same when each bit is cleared (0))																																															
CS41	VDCSZ4 (when accessing bank 10 or 11)																																																
CS42	VDCSZ4 (when accessing bank 9)																																																
CS43	VDCSZ4 (when accessing bank 8)																																																
CS50	VDCSZ5 (when accessing bank 15)																																																
CS51	VDCSZ5 (when accessing bank 14)																																																
CS52	VDCSZ5 (when accessing bank 13)																																																
CS53	VDCSZ5 (when accessing bank 12)																																																
CS60	VDCSZ6 (when accessing bank 14 or 15)	VDCSZ6 (when accessing area 3) (Same when each bit is cleared (0))																																															
CS61	VDCSZ6 (when accessing bank 12 or 13)																																																
CS62	VDCSZ6 (when accessing bank 11)																																																
CS63	VDCSZ6 (when accessing bank 10)																																																
CS70	VDCSZ7 (when accessing bank 15)																																																
CS71	VDCSZ7 (when accessing bank 14)																																																
CS72	VDCSZ7 (when accessing bank 13)																																																
CS73	VDCSZ7 (when accessing bank 12)																																																

Remark n = 4 to 7

Examples 1. The following figure shows an example of CSC0 and CSC1 register settings for 64 MB mode and the memory map after the settings are made.

Figure 4-3. CSC0 and CSC1 Register Setting Example (64 MB Mode) (1/3)

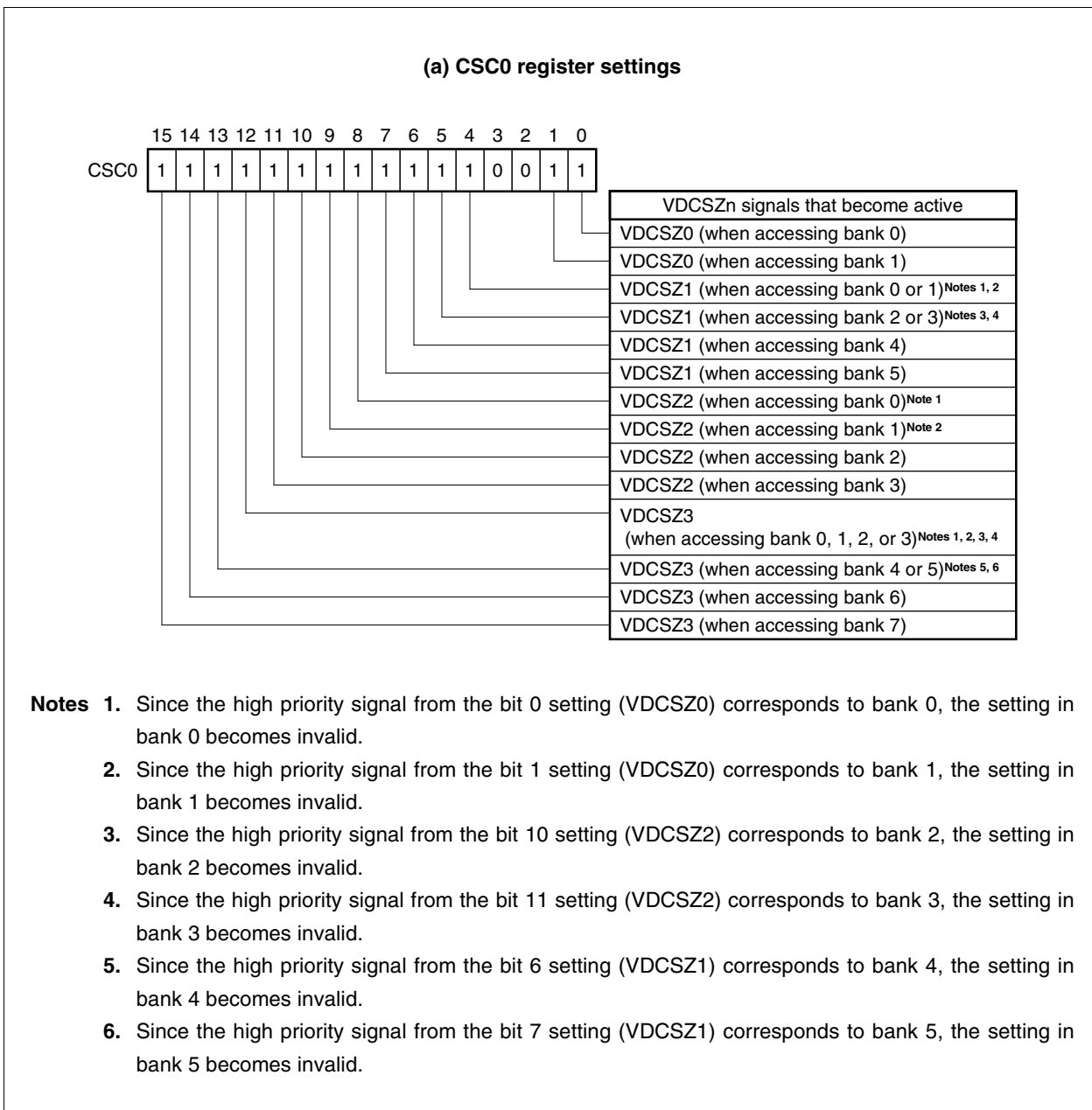


Figure 4-3. CSC0 and CSC1 Register Setting Example (64 MB Mode) (2/3)

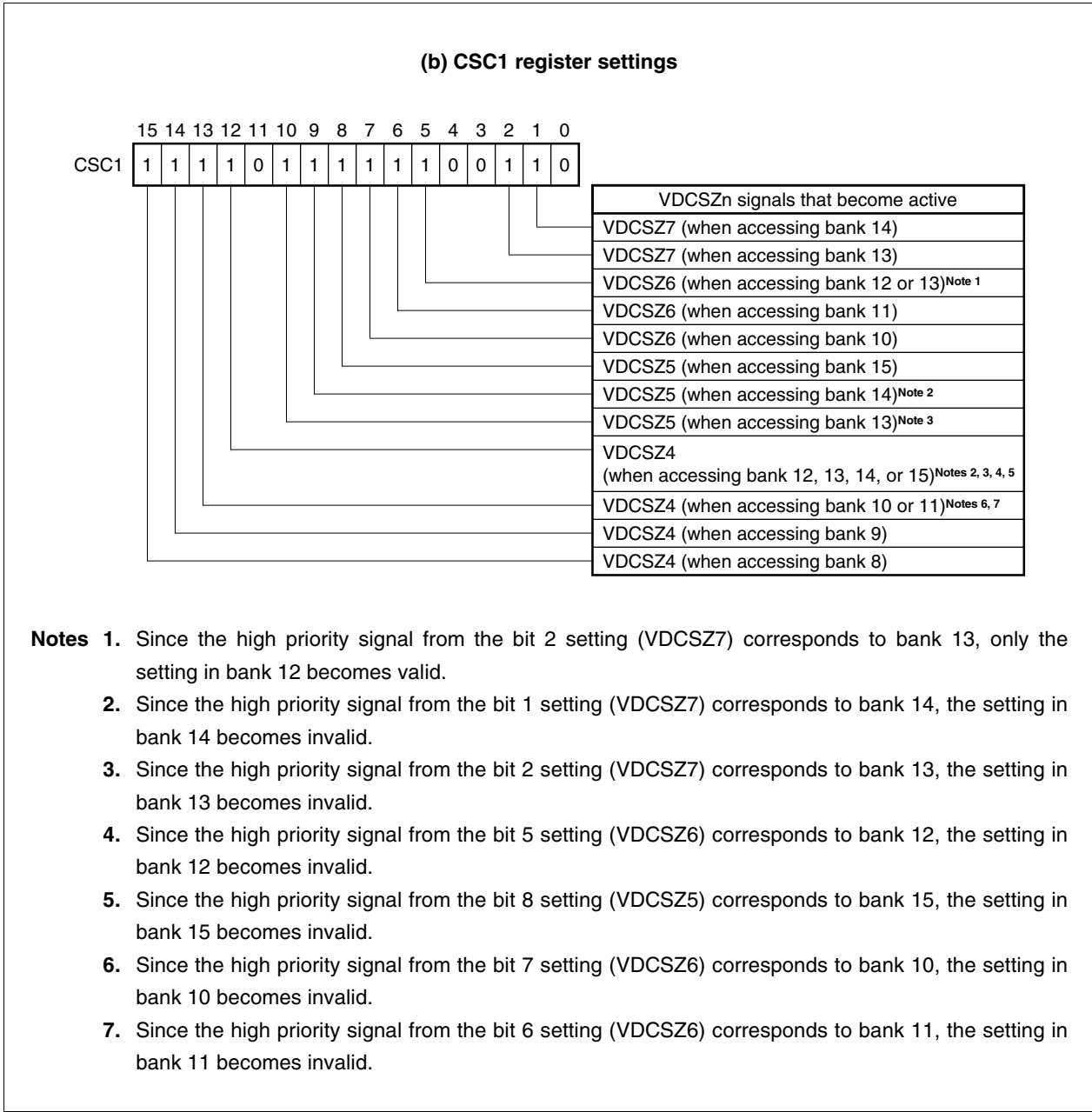
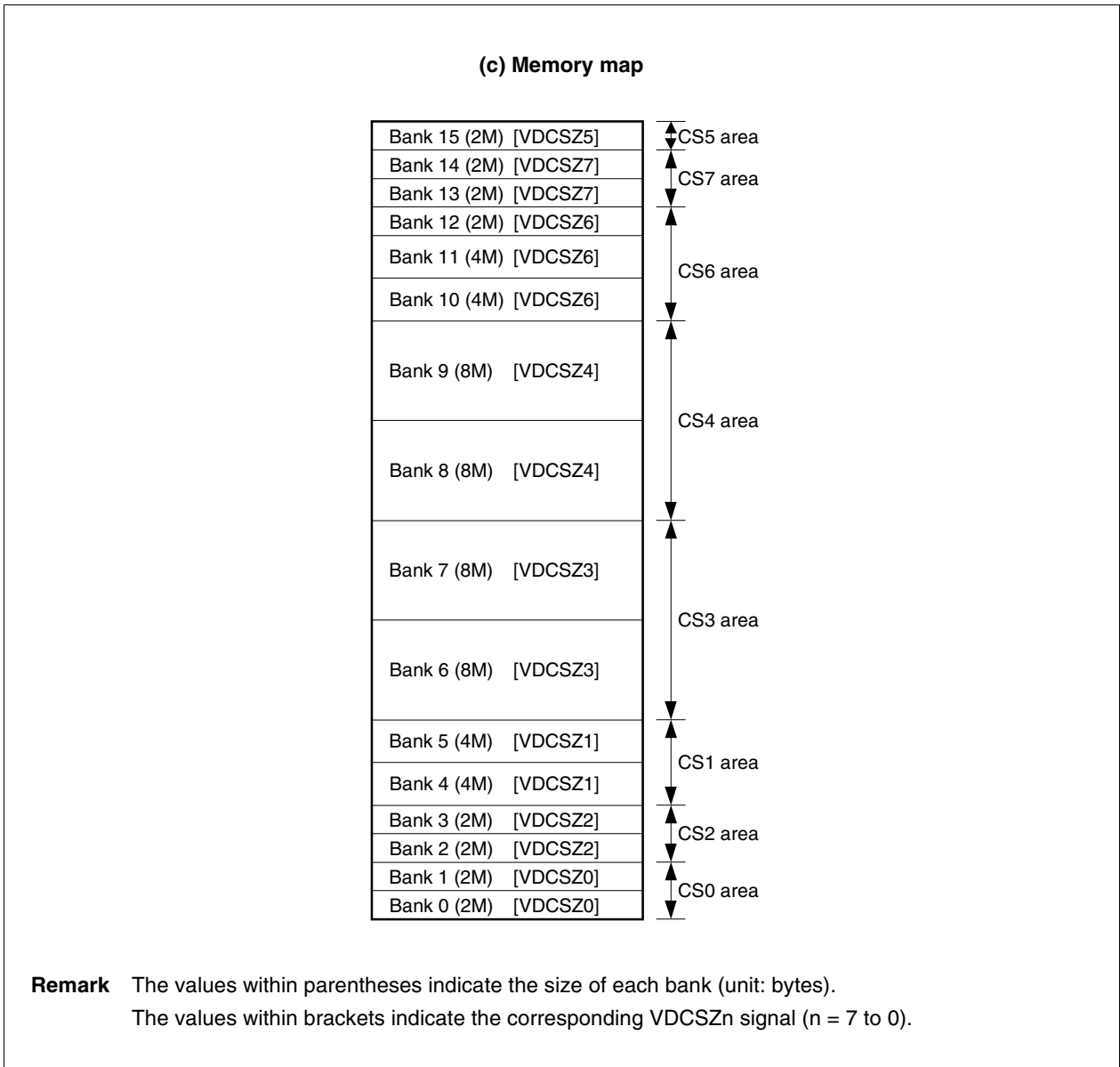


Figure 4-3. CSC0 and CSC1 Register Setting Example (64 MB Mode) (3/3)



Examples 2. The following figure shows an example of CSC0 and CSC1 register settings for 256 MB mode and the memory map after the settings are made.

Figure 4-4. CSC0 and CSC1 Register Setting Example (256 MB Mode) (1/2)

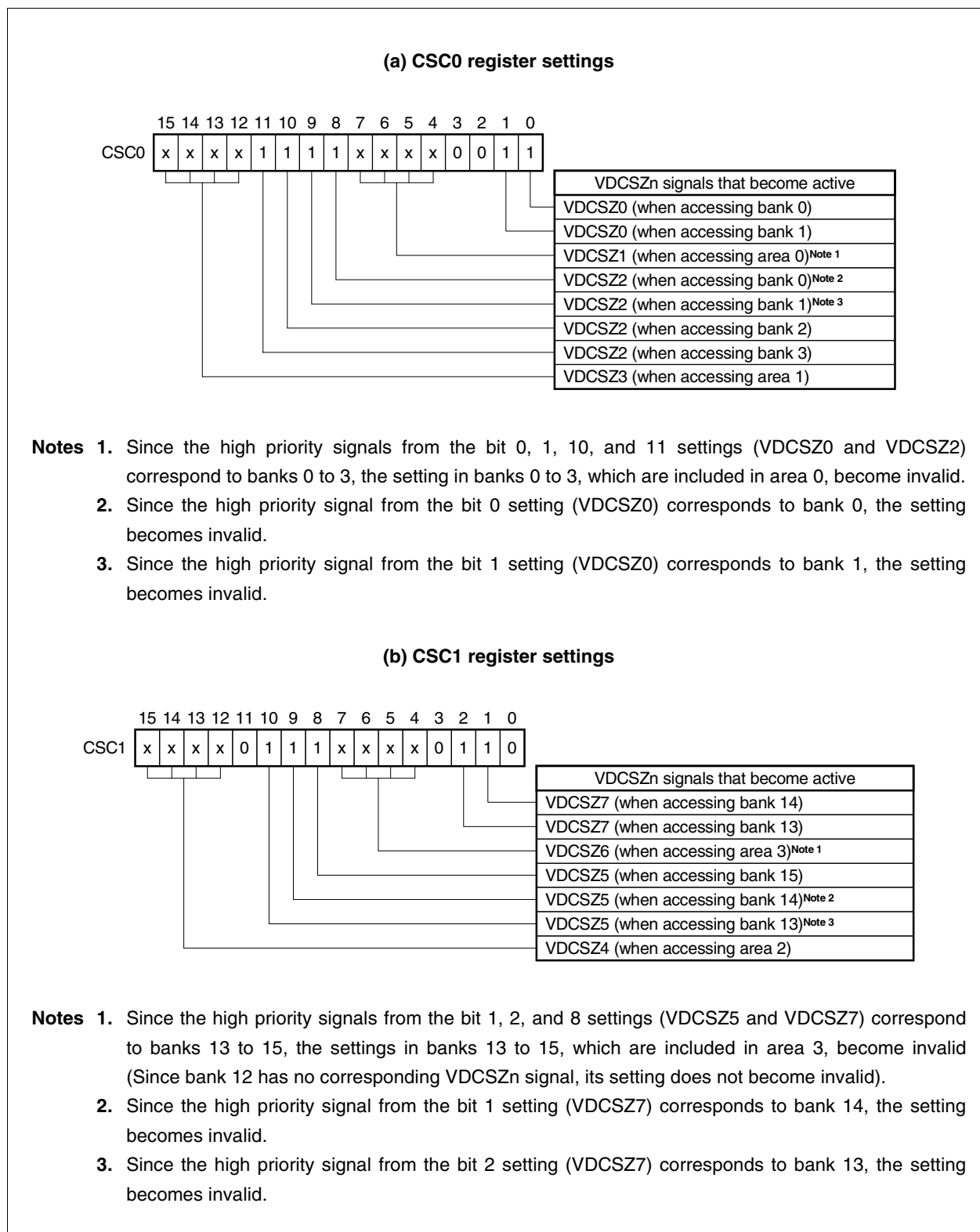
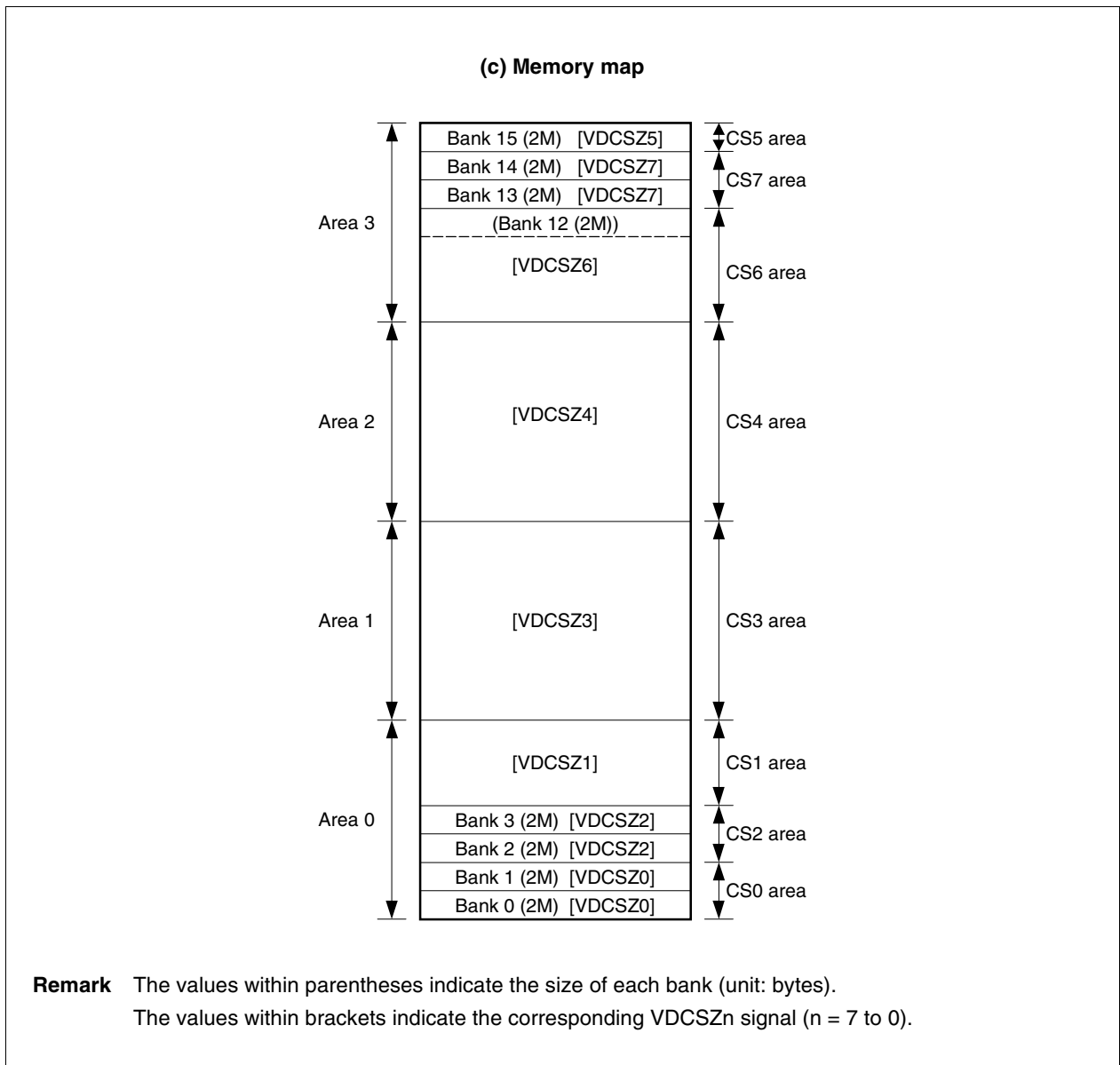


Figure 4-4. CSC0 and CSC1 Register Setting Example (256 MB Mode) (2/2)



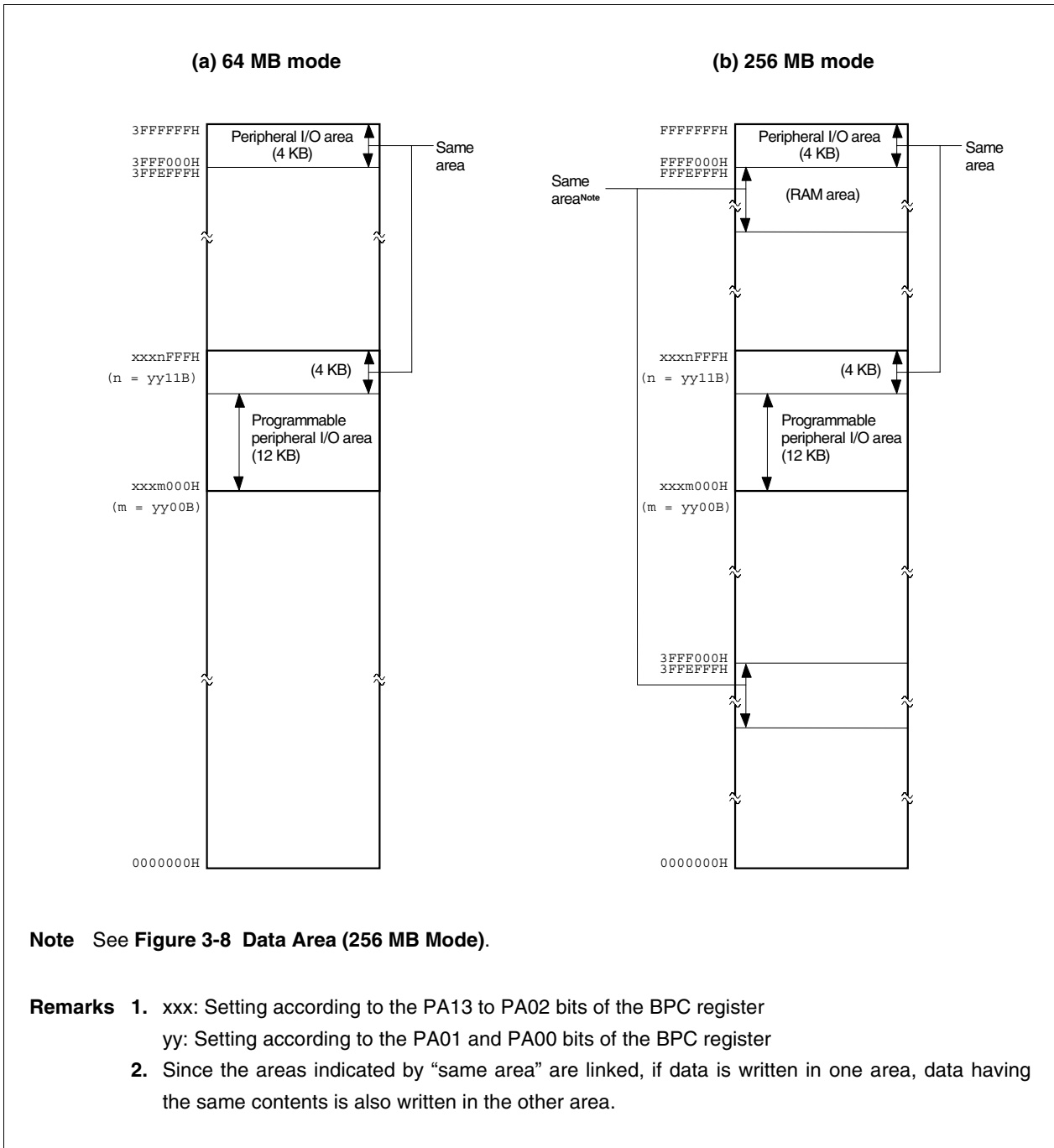
4.4 Programmable Peripheral I/O Area Selection Function

The NU85E has a 4 KB peripheral I/O area that is allocated in advance in the address space and a 12 KB programmable peripheral I/O area that can be allocated at arbitrary addresses according to register settings.

Registers for peripheral macros connected to the NPB or user logic can be freely located in the programmable peripheral I/O area.

Caution Be sure to allocate the programmable peripheral I/O area to a CSn area in which both little endian and instruction/data cache-prohibited settings have been made (n = 7 to 0).

Figure 4-5. Peripheral I/O Area and Programmable Peripheral I/O Area



Note See Figure 3-8 Data Area (256 MB Mode).

- Remarks**
- xxx: Setting according to the PA13 to PA02 bits of the BPC register
yy: Setting according to the PA01 and PA00 bits of the BPC register
 - Since the areas indicated by "same area" are linked, if data is written in one area, data having the same contents is also written in the other area.

The programmable peripheral I/O area can be used by specifying the higher 14 bits (bit 27 to bit 14) of the starting address in the PA00 to PA13 bits of the peripheral I/O area select control register (BPC) and setting (1) the PA15 bit. The BPC register can be read or written in 16-bit units.

The prioritization of the various CSn areas selected by the VDCSZn signals and the programmable peripheral I/O area is as follows (n = 7 to 0).

Programmable peripheral I/O area > Various CSn areas selected by VDCSZn signals

Cautions 1. In 64 MB mode, if the programmable peripheral I/O area overlaps the following areas, the programmable peripheral I/O area becomes invalid.

- Peripheral I/O area
- ROM area
- RAM area

2. In 256 MB mode, if the programmable peripheral I/O area overlaps the following areas, the programmable peripheral I/O area becomes invalid.

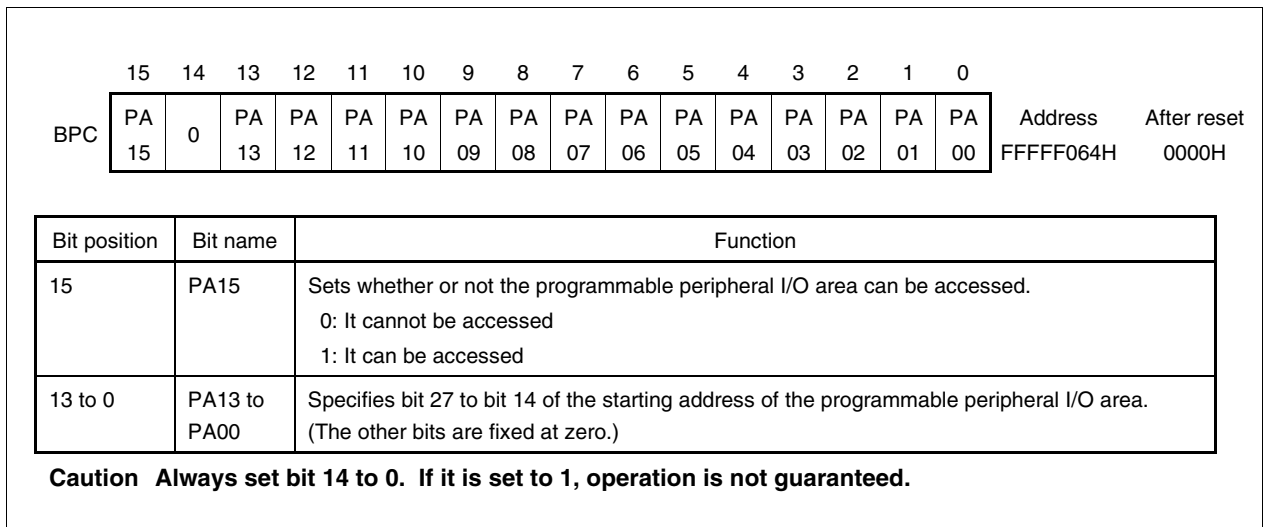
- Peripheral I/O area
- ROM area
- RAM area
- The area that is the same as the RAM area and that is located at address 3FFEFFFFH and below (See Figure 3-8 Data Area (256 MB Mode))

3. If there are no peripheral macros connected to the NPB or user logic, no programmable peripheral I/O area need be set (Set the BPC register to its after-reset value).

4. When accessing the programmable peripheral I/O area, the VDCSZn signals are all output as inactive (high level) (n = 7 to 0).

5. Programmable peripheral I/O area address setting is enabled only once. Do not change address in the middle of a program.

Figure 4-6. Peripheral I/O Area Select Control Register (BPC)

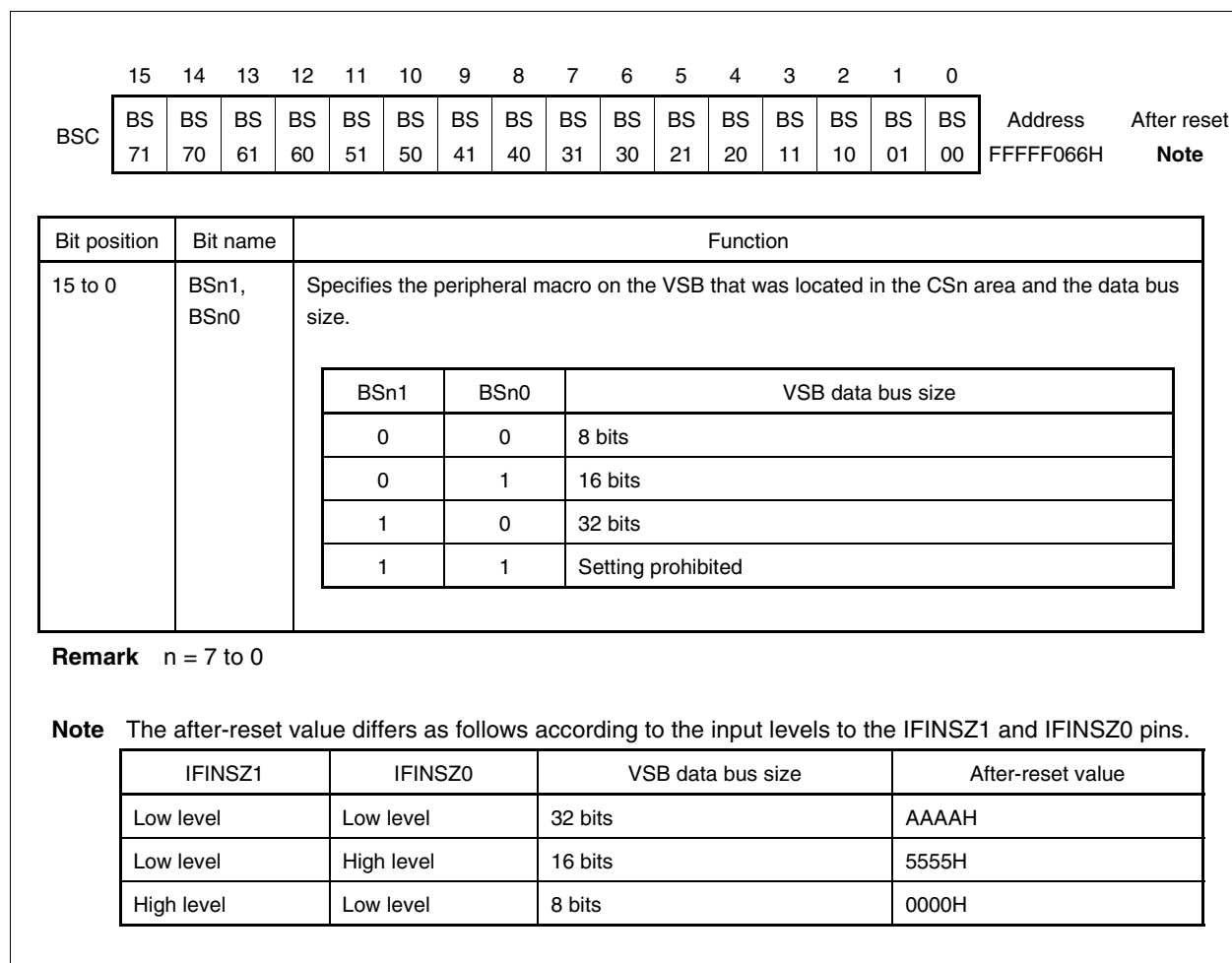


4.5 Bus Size Setting Function

The bus size setting function uses the bus size configuration register (BSC) to set the VSB data bus size for each CSn area selected by the chip select signals (VDCSZn) (see **Figures 4-3** and **4-4**) (n = 7 to 0).

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

Figure 4-7. Bus Size Configuration Register (BSC)



Example In a CSn area, when the boot ROM is 16-bit width and memories in other areas are 32-bit width, start up using 16-bit width in the initial state (input a low level to the IFINSZ1 pin and a high level to the IFINSZ0 pin) and then switch to 32-bit width via the BSC register.

4.6 Endian Setting Function

4.6.1 Endian configuration register (BEC)

The endian setting function uses the endian configuration register (BEC) to set the endian format of word data within memory for each CS_n area selected by the chip select signals (VDCSZ_n) (see **Figures 4-3** and **4-4**) (n = 7 to 0).

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

Cautions 1. Set the CS_n area specified as the programmable peripheral I/O area in the little endian format (n = 7 to 0).

2. Each of the following areas is fixed at little endian format. Any setting of the big endian format for these areas according to the BEC register is invalid.

- Peripheral I/O area
- ROM area
- RAM area
- The area that is the same as the RAM area and that is located at address 3FFEFFFFH and below (for 256 MB mode) (See Figure 3-8 Data Area (256 MB Mode))
- External memory fetch area (when VMBSTR signal is active)

Figure 4-8. Endian Configuration Register (BEC)

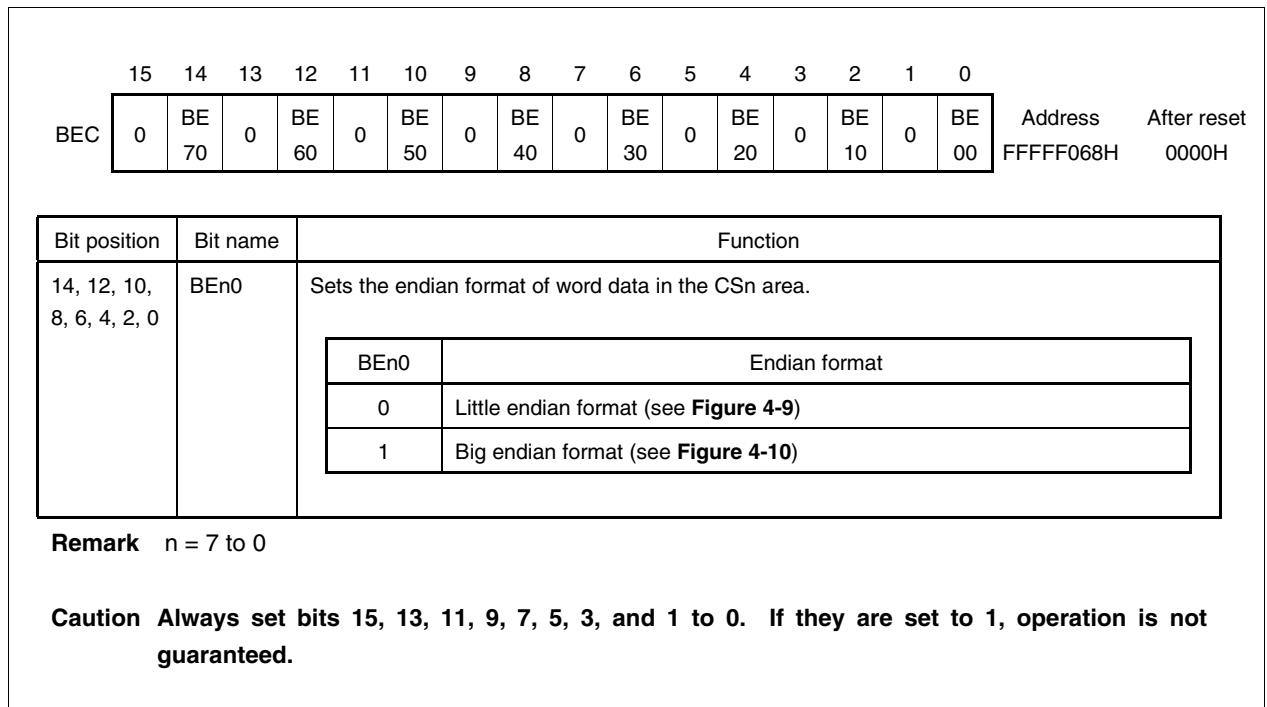
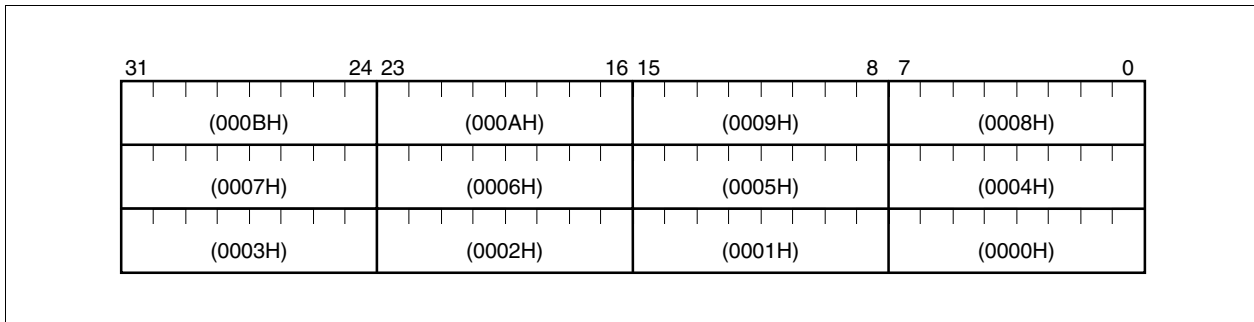
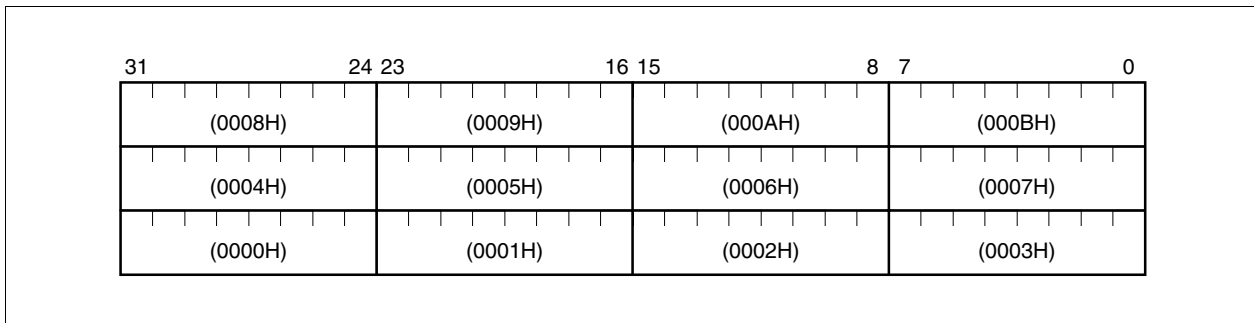


Figure 4-9. Word Data Little Endian Format Example**Figure 4-10. Word Data Big Endian Format Example**

4.6.2 Usage restrictions concerning big endian format with NEC development tools

(1) When using the debugger (ID850)

Only the memory window display supports the big endian format.

(2) When using the compiler (CA850)

(a) C language restrictions

(i) The following restrictions are attached to variables configured in a big endian space.

- <1> unions cannot be used.
- <2> bitfields cannot be used.
- <3> Accesses based on the cast (changed access size) cannot be used.
- <4> Variables with initial values cannot be used.

(ii) Because the access size may change due to optimization, it is necessary to specify the following optimization suppression options.

- For global optimization sections (opt850).....-Wo, -XTb
- For model-based optimization sections (impr850).....-Wi, +arg_reg_opt=OFF, +std_trans_opt=OFF

However, it is unnecessary to specify the above optimization suppression options when not using “cast” or “mask/shift” access^{Note}.

Note The condition is that patterns causing the following optimization are not used. It is extremely difficult to perform a perfect check on the user side in a state such as where all the patterns (especially in the model-based optimization section) are mixed together. The above optimization suppression options are therefore recommended.

<1> For global optimization section

- 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);

<2> For model-based optimization section

Usage whereby identical variables are accessed in different sizes

- Cast
- Mask
- Shift

Example

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

(b) Assembly language restrictions

Area-securing dummy instructions that are not byte size (.hword, .word, .float, .shword) cannot be used for variables configured in the big endian space.

4.7 Cache Configuration

The cache configuration register (BHC) is used to set the cache memory configuration for each CSn area selected by the chip select signals (VDCSZn) (see **Figures 4-3** and **4-4**) (n = 7 to 0).

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

Cautions 1. Be sure to disable the cache for big endian format CSn area or CSn areas set as the following areas (n = 7 to 0).

- ROM area
- RAM area
- Peripheral I/O area
- Programmable peripheral I/O area

2. The instruction/data cache enabled setting (BHn0/BHn1 bit = 1 (set)) is only valid when a low level is being input (cache enabled) to the IFIUNCH0 or IFIUNCH1 pin. In other cases, the instruction/data cache enabled setting is invalid even if the BHn0/BHn1 bit is set to 1.
3. When using the data cache, set this register after setting the data cache's data cache control register (DCC).

Figure 4-11. Cache Configuration Register (BHC)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																			
BHC	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	BH	71	70	61	60	51	50	41	40	31	30	21	20	11	10	01	00	Address	After reset	
																																		FFFFFF06AH	0000H

Bit position	Bit name	Function						
15, 13, 11, 9, 7, 5, 3, 1	BHn1	Sets whether or not the data cache located in the CSn area can be used. <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="width: 10%;">BHn1</th> <th style="width: 90%;">Data cache setting</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Cache disabled</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Cache enabled</td> </tr> </tbody> </table>	BHn1	Data cache setting	0	Cache disabled	1	Cache enabled
BHn1	Data cache setting							
0	Cache disabled							
1	Cache enabled							
14, 12, 10, 8, 6, 4, 2, 0	BHn0	Sets whether or not the instruction cache located in the CSn area can be used. <table border="1" style="width: 100%; margin-top: 10px;"> <thead> <tr> <th style="width: 10%;">BHn0</th> <th style="width: 90%;">Instruction cache setting</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td>Cache disabled</td> </tr> <tr> <td style="text-align: center;">1</td> <td>Cache enabled</td> </tr> </tbody> </table>	BHn0	Instruction cache setting	0	Cache disabled	1	Cache enabled
BHn0	Instruction cache setting							
0	Cache disabled							
1	Cache enabled							

Remark n = 7 to 0

4.8 BCU-Related Register Setting Examples

Figure 4-12 shows a BPC, BSC, BEC, and BHC register setting example, the corresponding settings for each CSn area, and the memory map when the data area has been set according to the contents of the example shown in Figure 4-3 **CSC0 and CSC1 Register Setting Example (64 MB Mode) (n = 7 to 0)**.

Figure 4-12. BPC, BSC, BEC, BHC Register Setting Example (1/3)

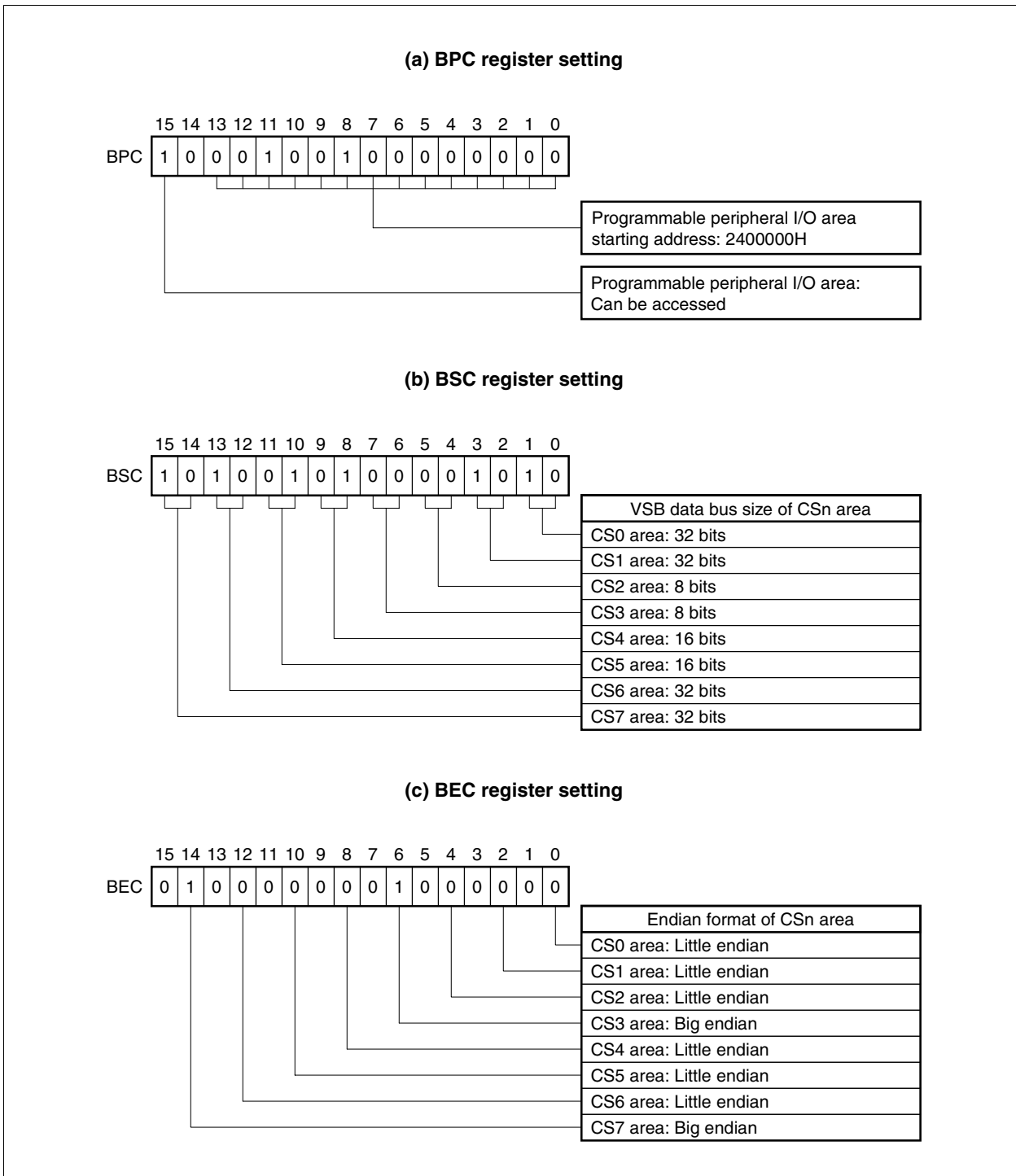


Figure 4-12. BPC, BSC, BEC, BHC Register Setting Example (2/3)

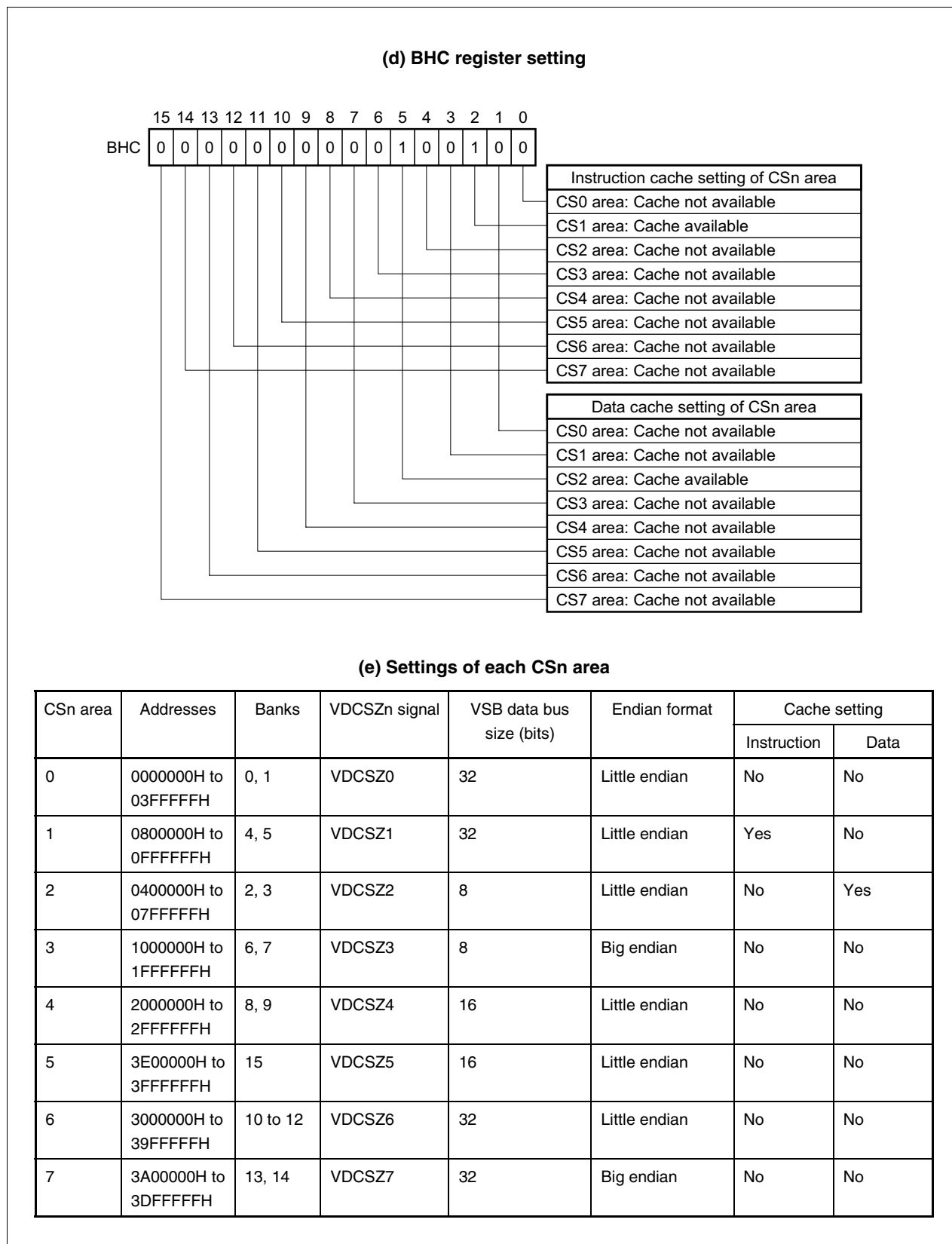
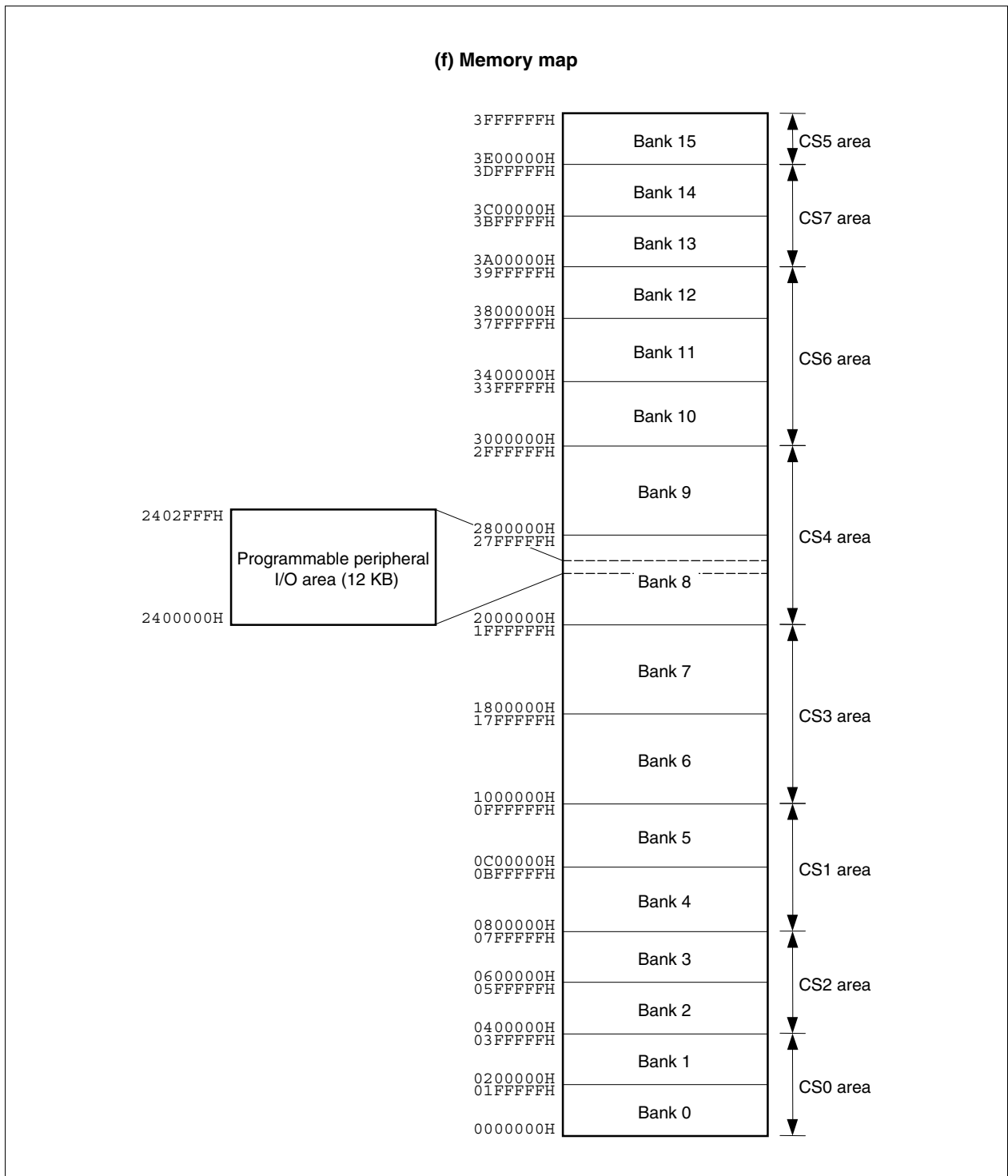


Figure 4-12. BPC, BSC, BEC, BHC Register Setting Example (3/3)

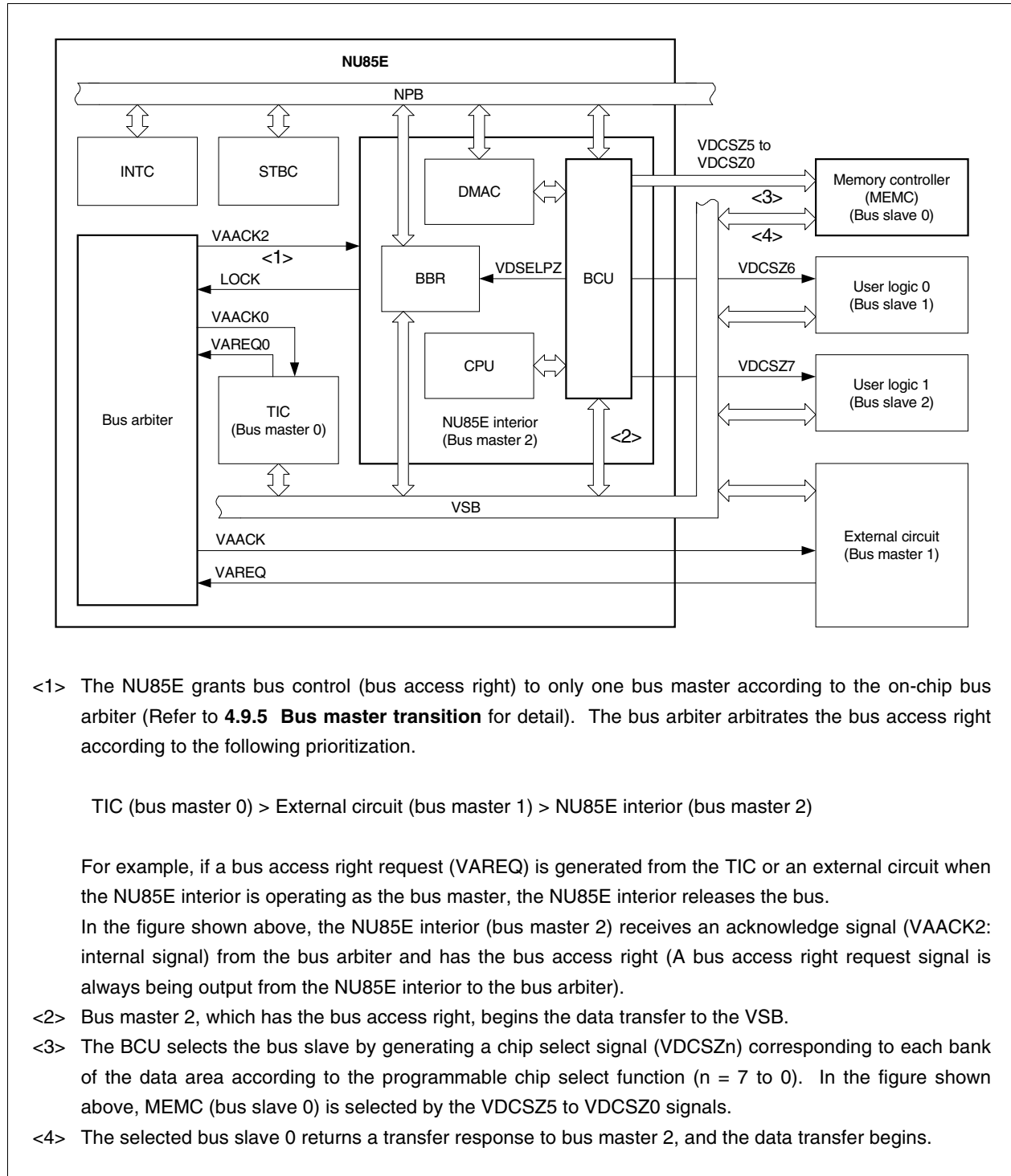


4.9 Data Transfer Using VSB

4.9.1 Data transfer example

This section uses the circuit shown in Figure 4-13 to explain the procedure for transferring data between bus masters and bus slaves connected to the VSB.

Figure 4-13. Example of Data Transfer Using VSB



4.9.2 Control signals output by bus master

When the NU85E operates as the bus master, the contents of the transfer that is currently being executed are indicated by outputting the various control signals indicated below (When the NU85E operates as a bus slave, the external bus master performs output, and this data is input to the NU85E as the VSxxxx signal).

However, the VxWAIT, VxAHLD, and VxLAST signals are output by the bus slave and input by the bus master (The signal names on the bus master side are VMWAIT, VMAHLD, and VMLAST, and the signal names on the bus slave side are VSWAIT, VSAHLD, and VSLAST).

(1) Transfer type

When the transfer begins, the bus master outputs the VMTTYP1 and VMTTYP0 signals to indicate the transfer type.

Table 4-1. VMTTYP1 and VMTTYP0 Signals

VMTTYP1	VMTTYP0	Transfer Type
0	0	Address-only transfer (transfer without data processing)
1	0	Non-sequential transfer (single transfer or burst transfer)
1	1	Sequential transfer (transfer in which the address currently being transferred is related to the previously transferred address)
0	1	(Reserved for future function expansion)

Remark 0: low-level 1: high-level

(2) Bus cycle type

The bus master indicates the current bus cycle status according to the VMCTYP2 to VMCTYP0 signals.

Table 4-2. VMCTYP2 to VMCTYP0 Signals

VMCTYP2	VMCTYP1	VMCTYP0	Bus Cycle Status
0	0	0	Opcode fetch
0	0	1	Data access
0	1	0	Misalign access ^{Note}
0	1	1	Read modify write access
1	0	0	Opcode fetch of jump address due to branch instruction
1	1	0	DMA 2-cycle transfer
1	1	1	DMA flyby transfer
1	0	1	(Reserved for future function expansion)

Note Output only when a high level is input to the IFIMAEN pin (misalign access enabled).

Remark 0: low-level 1: high-level

(3) Byte enable

The bus master uses the VMBENZ3 to VMBENZ0 signals to indicate the byte data among the data obtained by quartering the data bus (VBDI31 to VBDI0 and VBDO31 to VBDO0) into byte units.

Table 4-3. VMBENZ3 to VMBENZ0 Signals

Active (Low-Level Output) Signal	Enabled Byte Data
VMBENZ0	VBDI7 to VBDI0, VBDO7 to VBDO0
VMBENZ1	VBDI15 to VBDI8, VBDO15 to VBDO8
VMBENZ2	VBDI23 to VBDI16, VBDO23 to VBDO16
VMBENZ3	VBDI31 to VBDI24, VBDO31 to VBDO24

(4) Transfer size

The bus master uses the VMSIZE1 and VMSIZE0 signals to indicate the transfer size.

Table 4-4. VMSIZE1 and VMSIZE0 Signals

VMSIZE1	VMSIZE0	Explanation
0	0	Byte (8 bits)
0	1	Halfword (16 bits)
1	0	Word (32 bits)
1	1	(Reserved for future function expansion)

Remark 0: low-level 1: high-level

(5) Sequential status

The bus master uses the VMSEQ2 to VMSEQ0 signals to indicate the “burst transfer length” when a burst transfer starts, to indicate “continuous” during a burst transfer, and to indicate “single transfer” at the end of the burst transfer.

Table 4-5. VMSEQ2 to VMSEQ0 Signals

VMSEQ2	VMSEQ1	VMSEQ0	Sequential Status
0	0	0	Single transfer
0	0	1	Continuous (indicates that the next transfer address is related to the current transfer address) ^{Note}
0	1	0	Continuous 4 times (burst transfer length: 4)
0	1	1	Continuous 8 times (burst transfer length: 8)
1	0	0	Continuous 16 times (burst transfer length: 16)
1	0	1	Continuous 32 times (burst transfer length: 32)
1	1	0	Continuous 64 times (burst transfer length: 64)
1	1	1	Continuous 128 times (burst transfer length: 128)

Note This is output during continuous 2 times, or continuous 4, 8, 16, 32, 64, or 128 times transfer.

Remark 0: low-level 1: high-level

(6) Transfer response

The transfer response is indicated by the VMWAIT, VMAHLD, and VMLAST signals, which are output from the bus slave (The signal names on the bus slave side are VSWAIT, VSAHLD, and VSLAST). These signals become effective only while the VBCLK signal is low level.

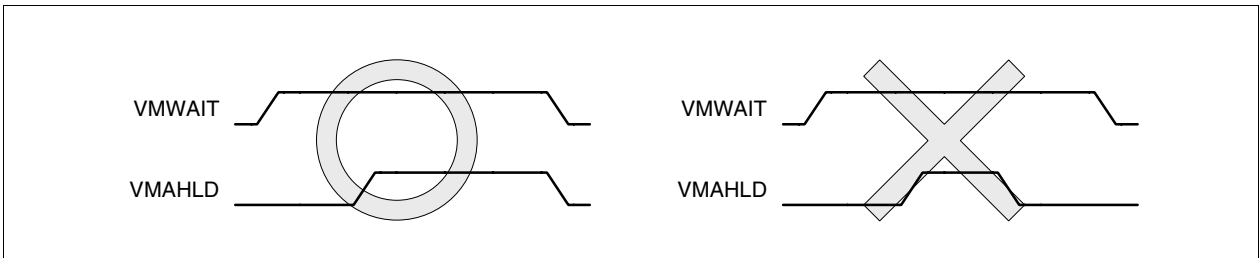
Table 4-6. VMWAIT, VMAHLD, and VMLAST Signals

VMWAIT	VMAHLD	VMLAST	Explanation
0	0	0	Status when the current transfer is completed (ready status)
0	0	1	Last response (burst transfer last response status)
1	0	0	Wait response (wait status)
1	1	0	Maintains address and control signal (address hold status)
Other than the above			(Reserved for future function expansion)

Remark 0: low-level 1: high-level

Caution Once the VMAHLD signal becomes active (1), hold the active level (1) until the VMWAIT signal becomes inactive (0).

It is not possible to return to the wait state from the address hold state during a bus cycle.



(7) Transfer direction

The bus master uses the VMWRITE signal to indicate the transfer direction. This signal outputs a high level during write access.

(8) Data bus direction control

The VBDC signal is the data input (VBDI31 to VBDI0) control signal output pin. This signal outputs a high level during read access.

The VBDV signal is the data output (VBDO31 to VBDO0) control signal output pin. This signal outputs a high level during write access.

Table 4-7. VBDC and VBDV Signals

VBDC	VBDV	Explanation
0	1	Read access
1	0	Write access

Remark 0: low-level 1: high-level

4.9.3 Read/write timing

(1) Read timing

Read data is output from the bus slave side in synchronization with the rising edge of the VBCLK signal immediately after the end of address output to the bus slave. Following this, the bus master fetches (samples) the data in synchronization with the next falling edge of the VBCLK signal.

However, if the VMAHLD signal has been input at an active level (high level), the bus slave outputs data in synchronization with the rising edge of the VBCLK signal immediately after the active-level VMAHLD was input, and the bus master fetches (samples) the data in synchronization with the next falling edge of the VBCLK signal.

(2) Write timing

Write data is output from the NU85E in synchronization with the falling edge of the VBCLK signal half clock after the address is output to the bus slave.

The following pages show the read/write timing of the bus master and slaves connected to the VSB. The diagrams show the timing seen from the NU85E side when the NU85E has bus access right.

Remark O mark: Sampling timing

A.x: Arbitrary address output from the VMA27 to VMA0 pins

D.x: I/O data for address "A.x"

XXX: Arbitrary level (for input), or undefined status (for output)

Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (1/12)

(a) 32-bit bus (single transfer, no waits)

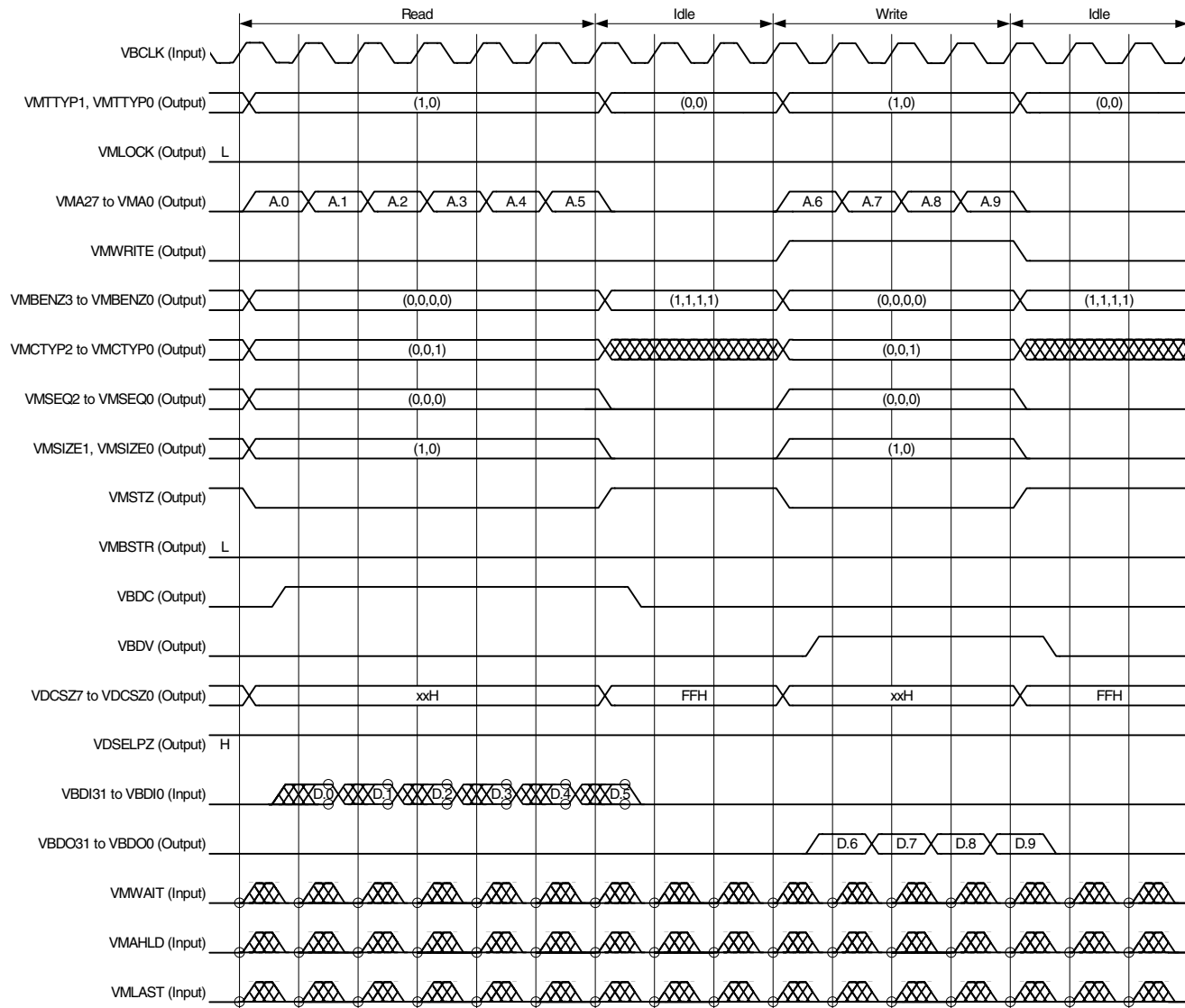


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (2/12)

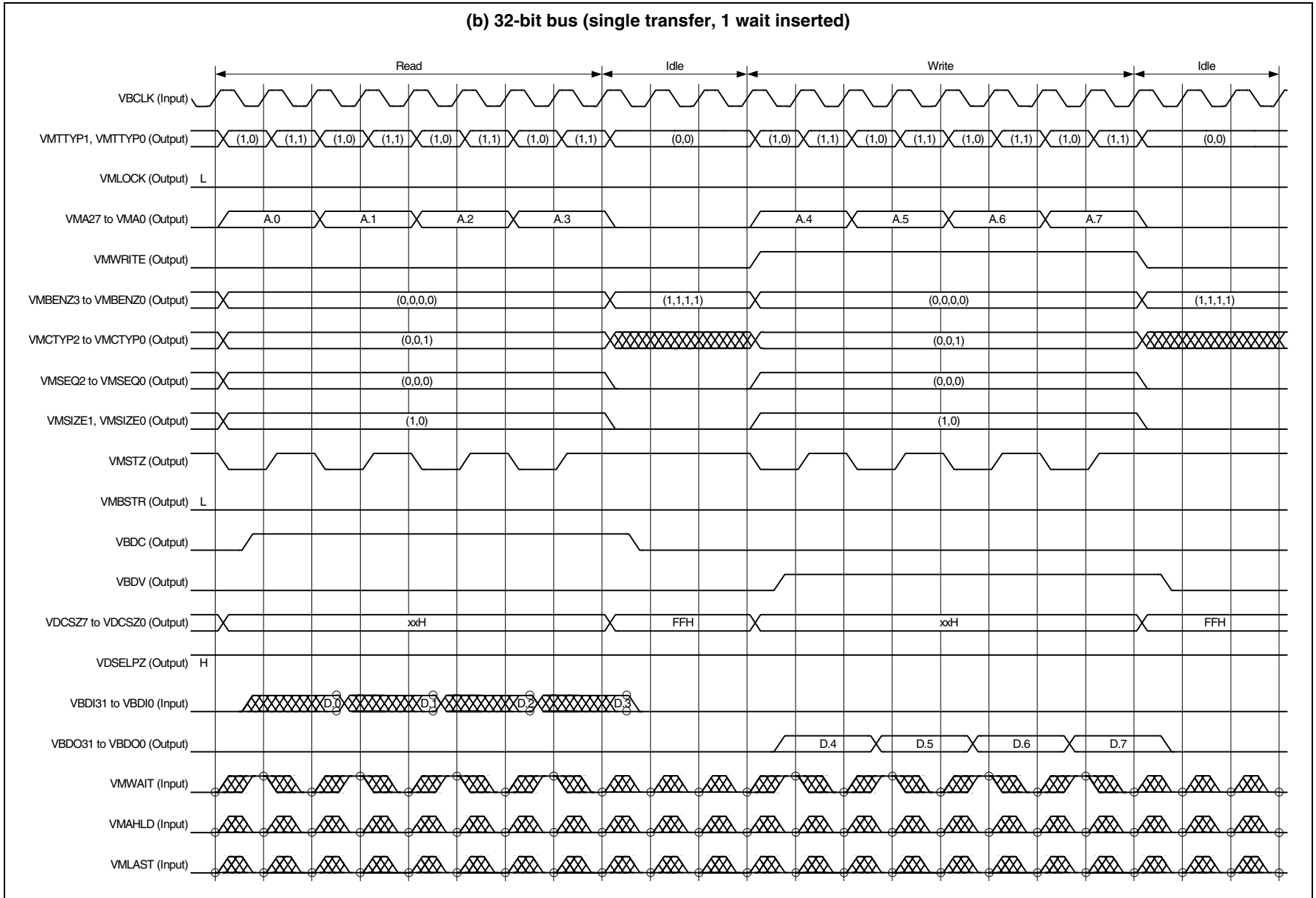


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (3/12)

(c) 32-bit bus (single transfer, 2 waits inserted)

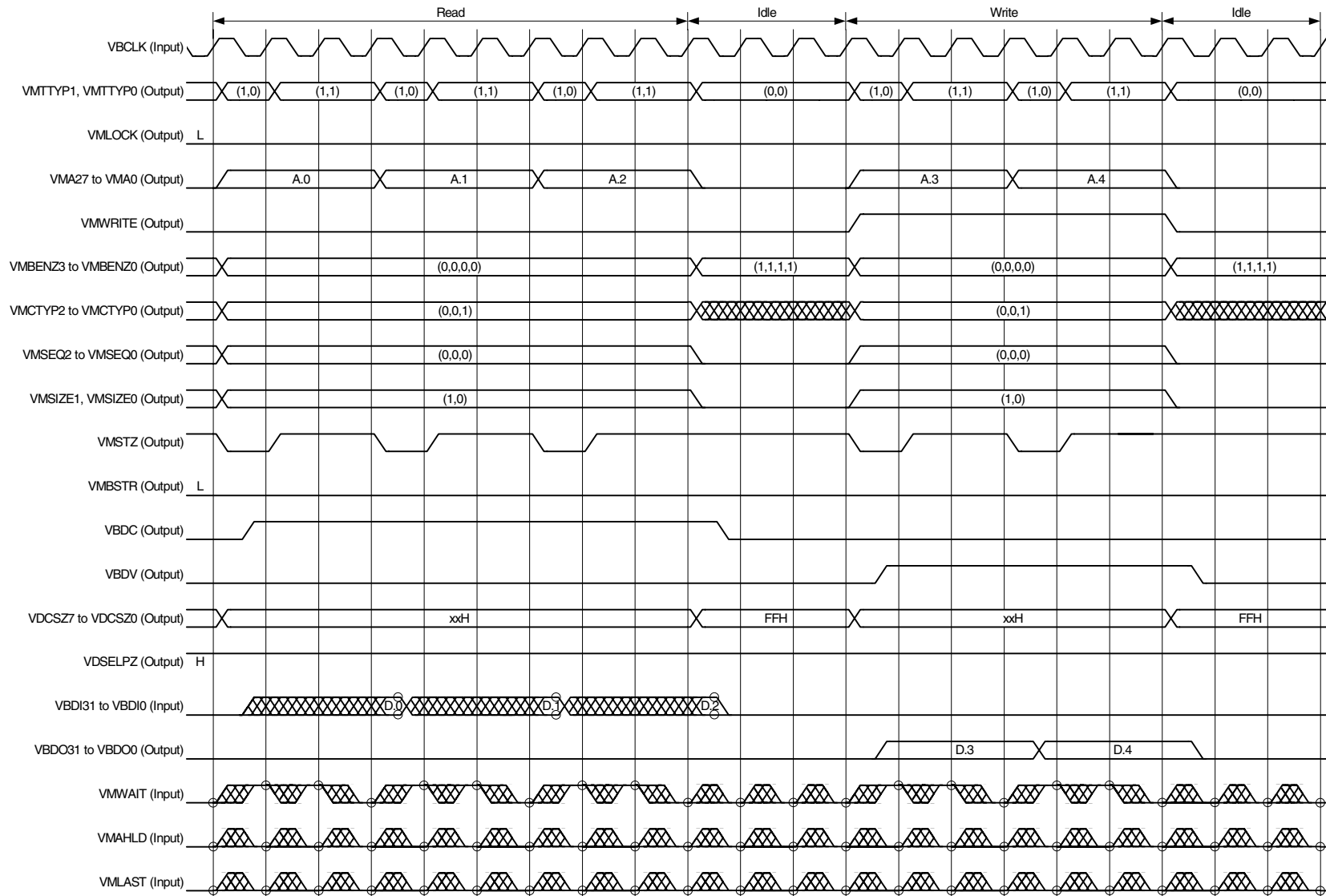


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (4/12)

(d) 32-bit bus (single transfer, 2 waits inserted, with address hold)

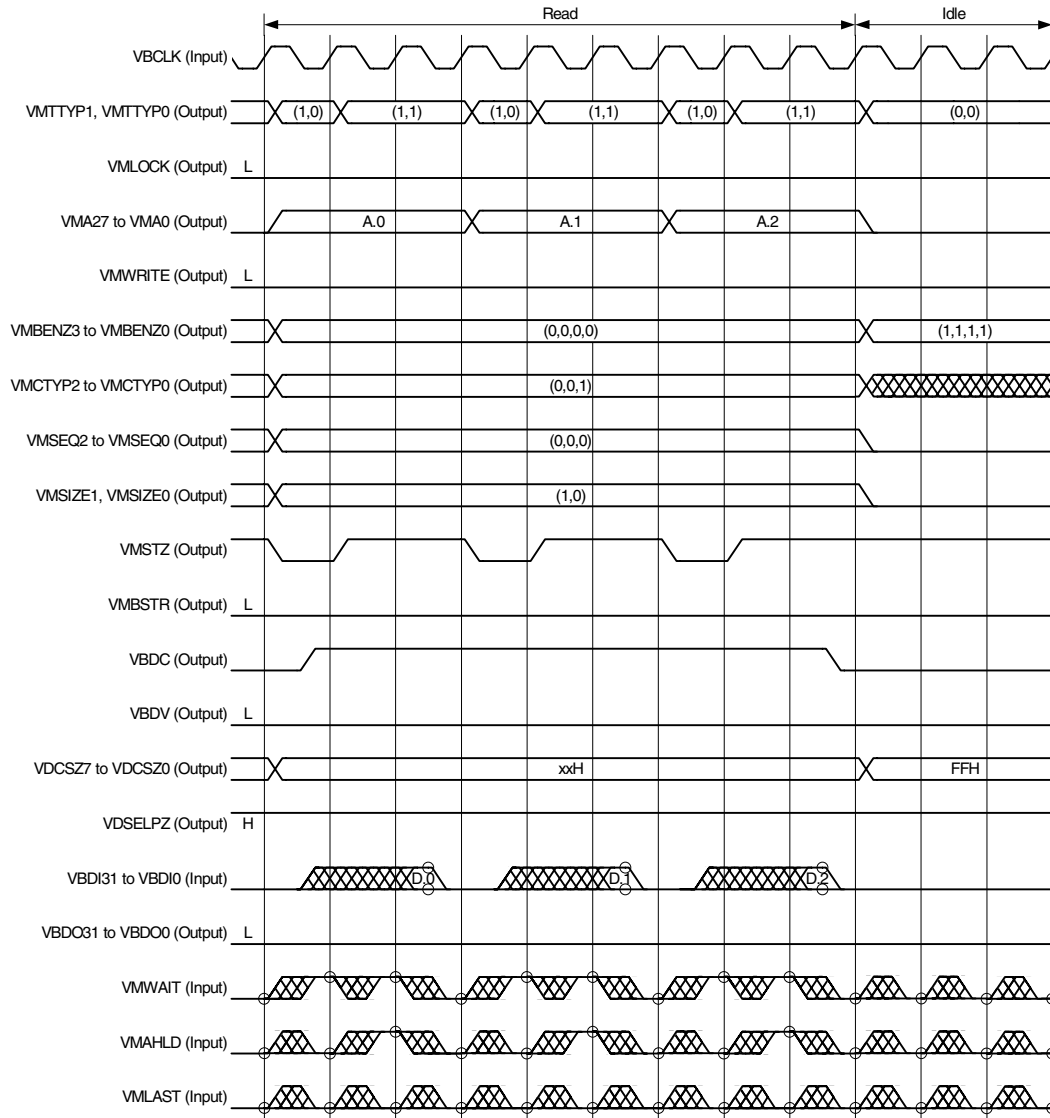


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (5/12)

(e) 32-bit bus (4-word sequential transfer, data access)

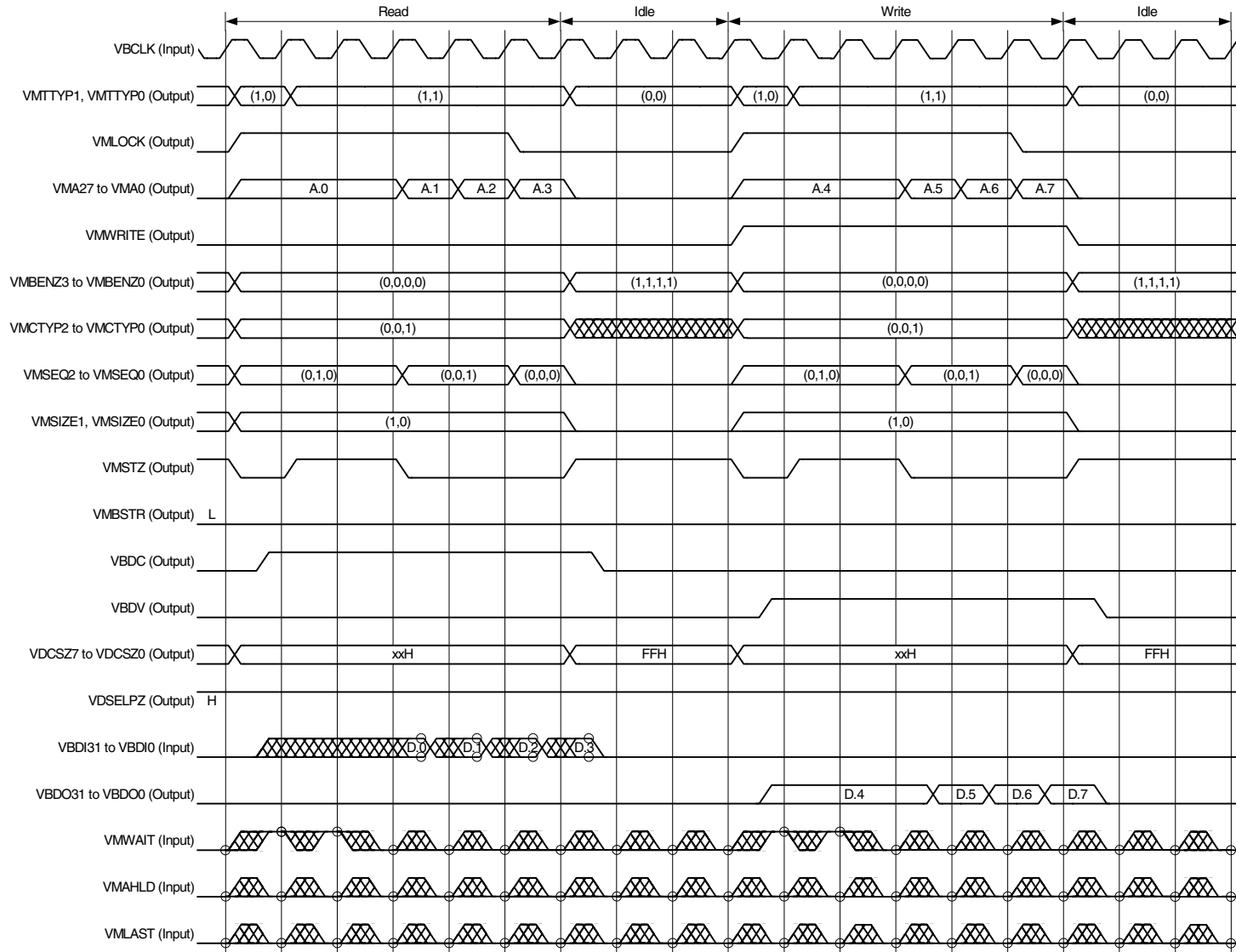


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (6/12)

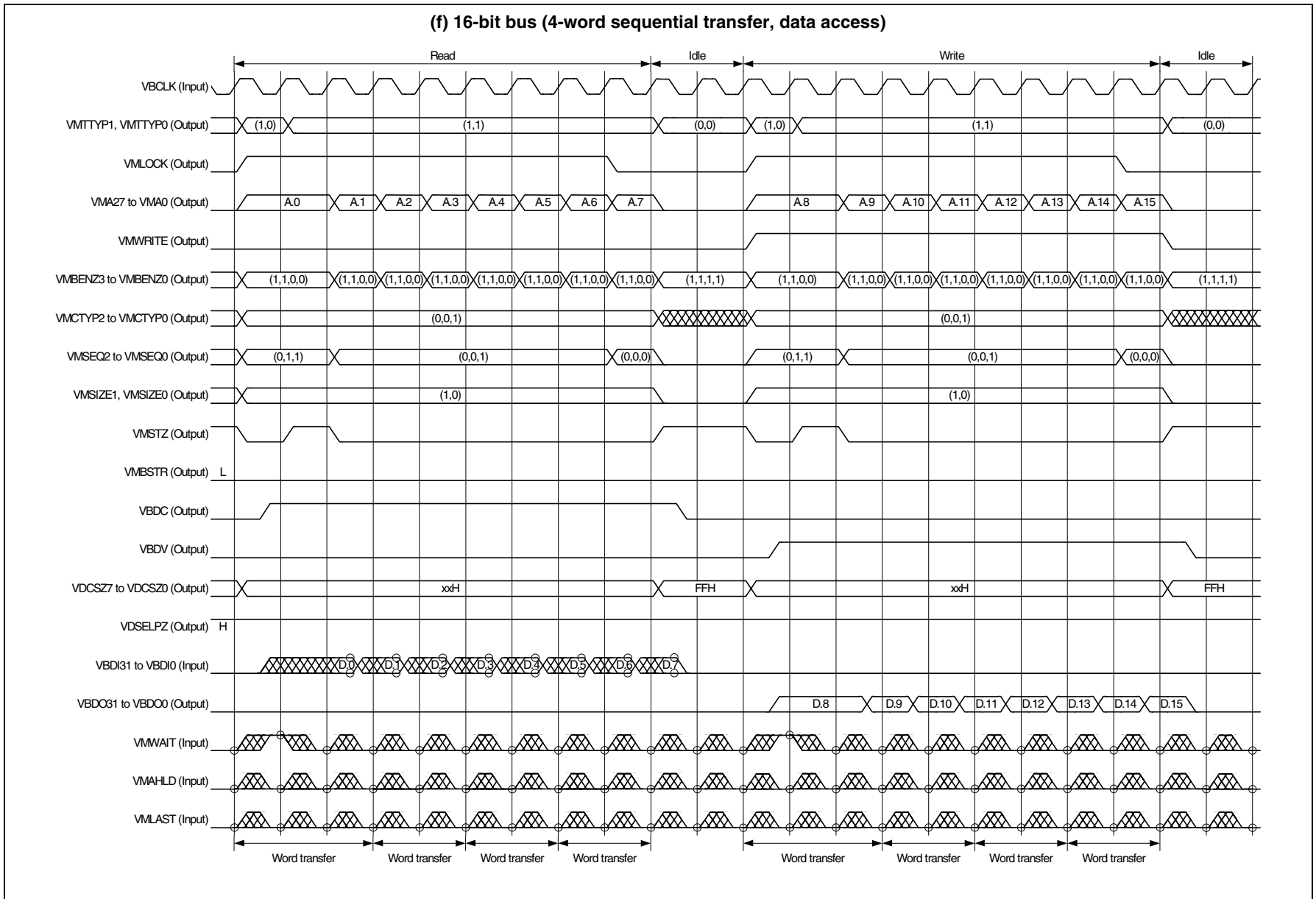


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (7/12)

(g) 8-bit bus (4-byte sequential transfer, external ROM fetch)

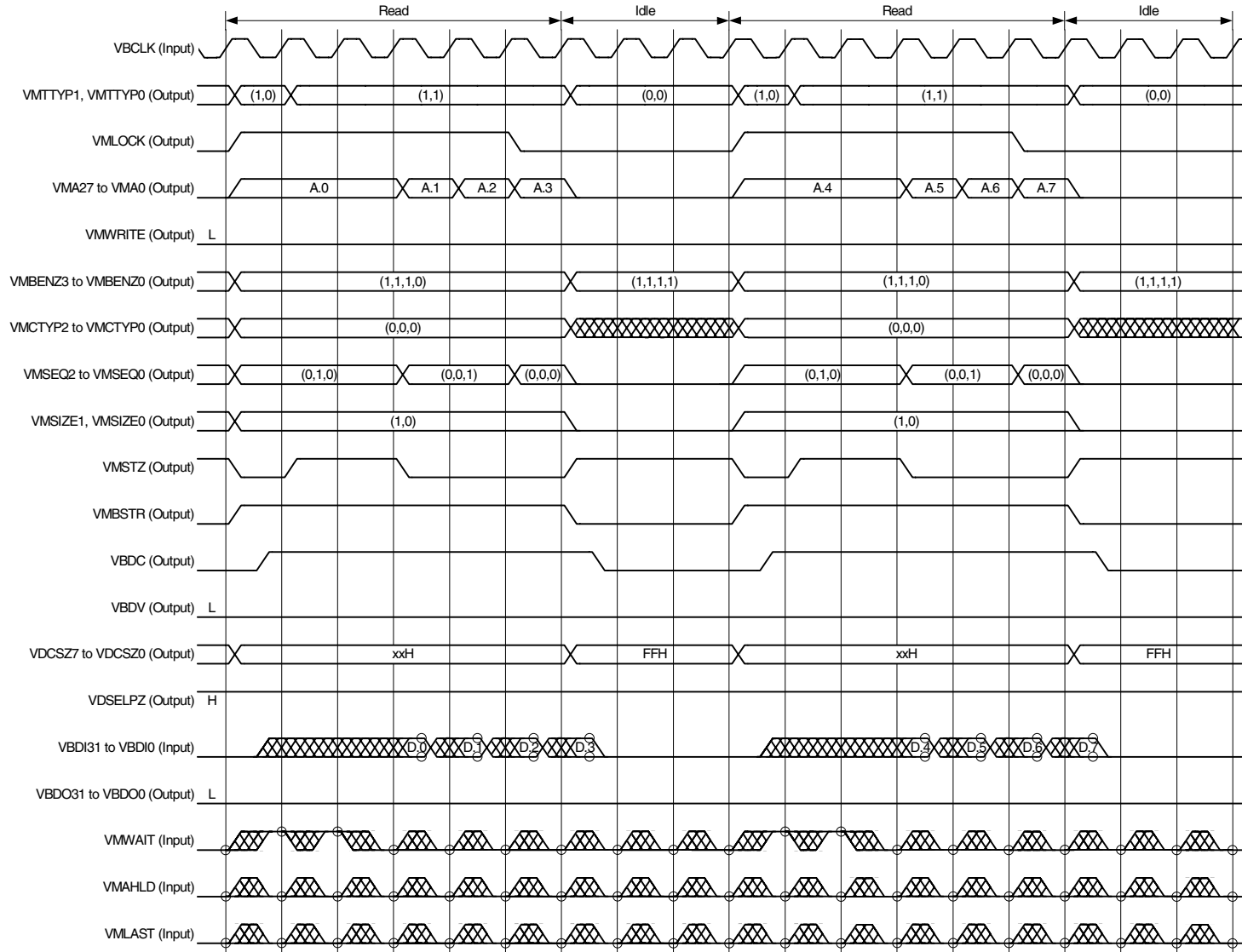


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (8/12)

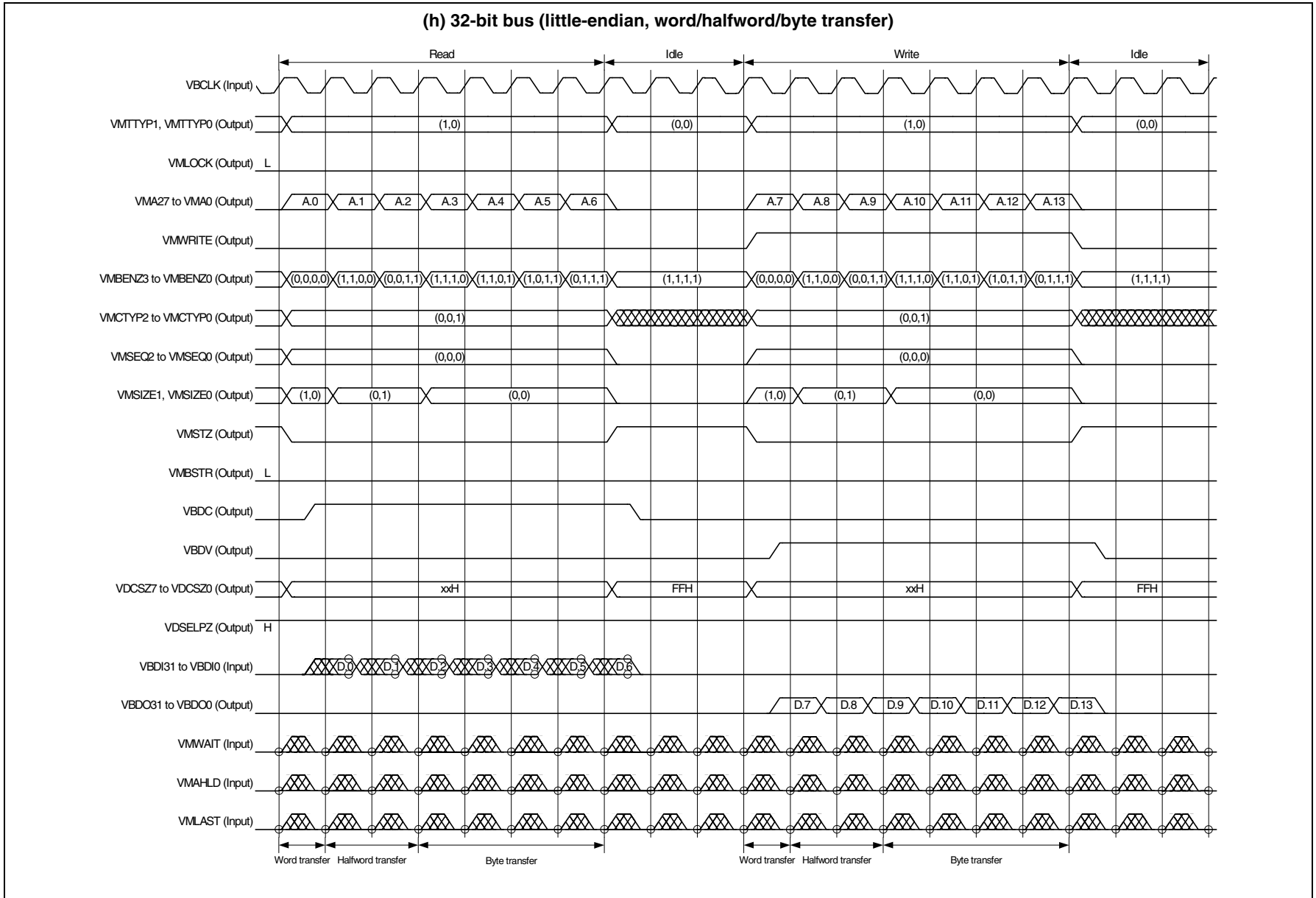


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (9/12)

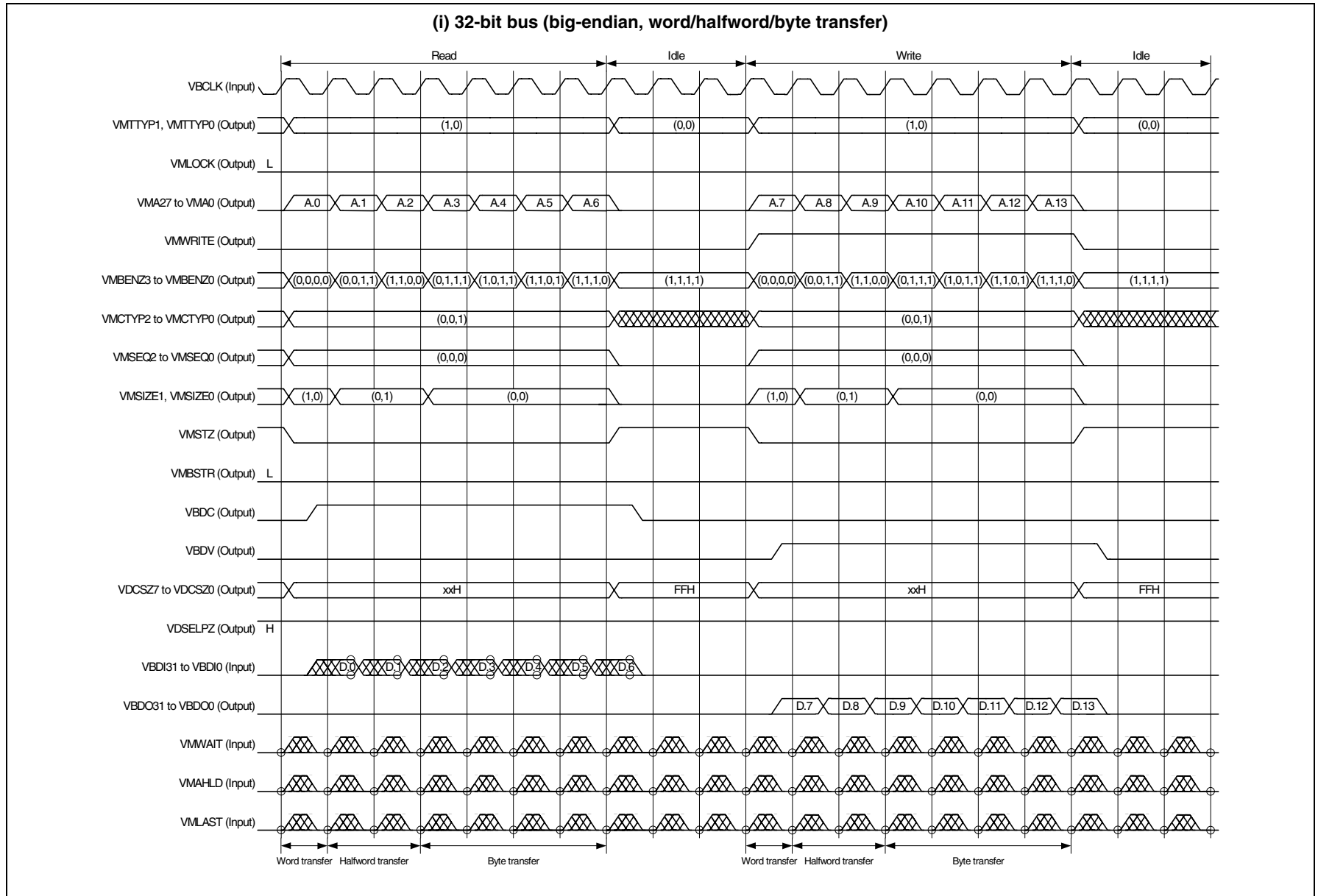


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (10/12)

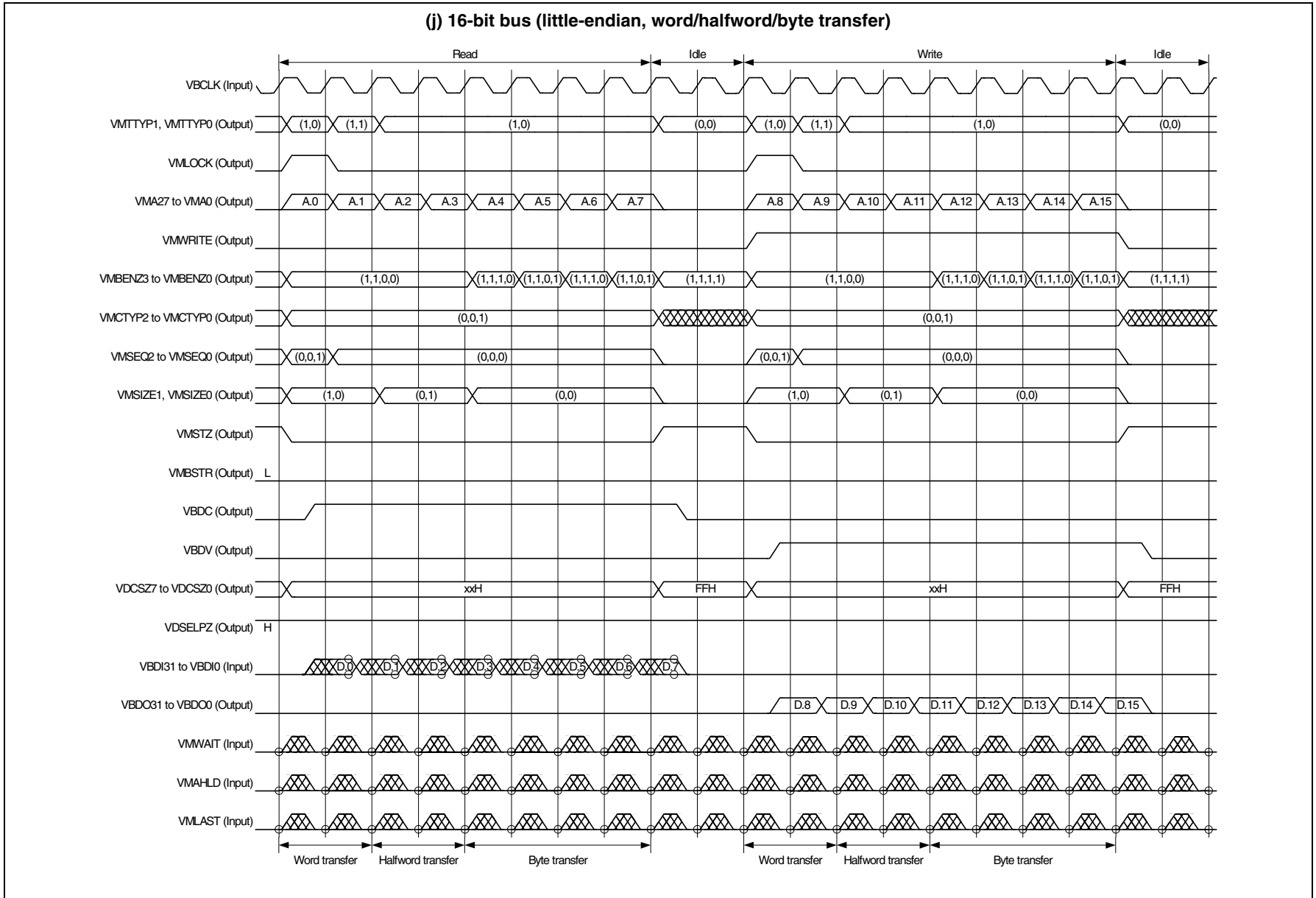


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (11/12)

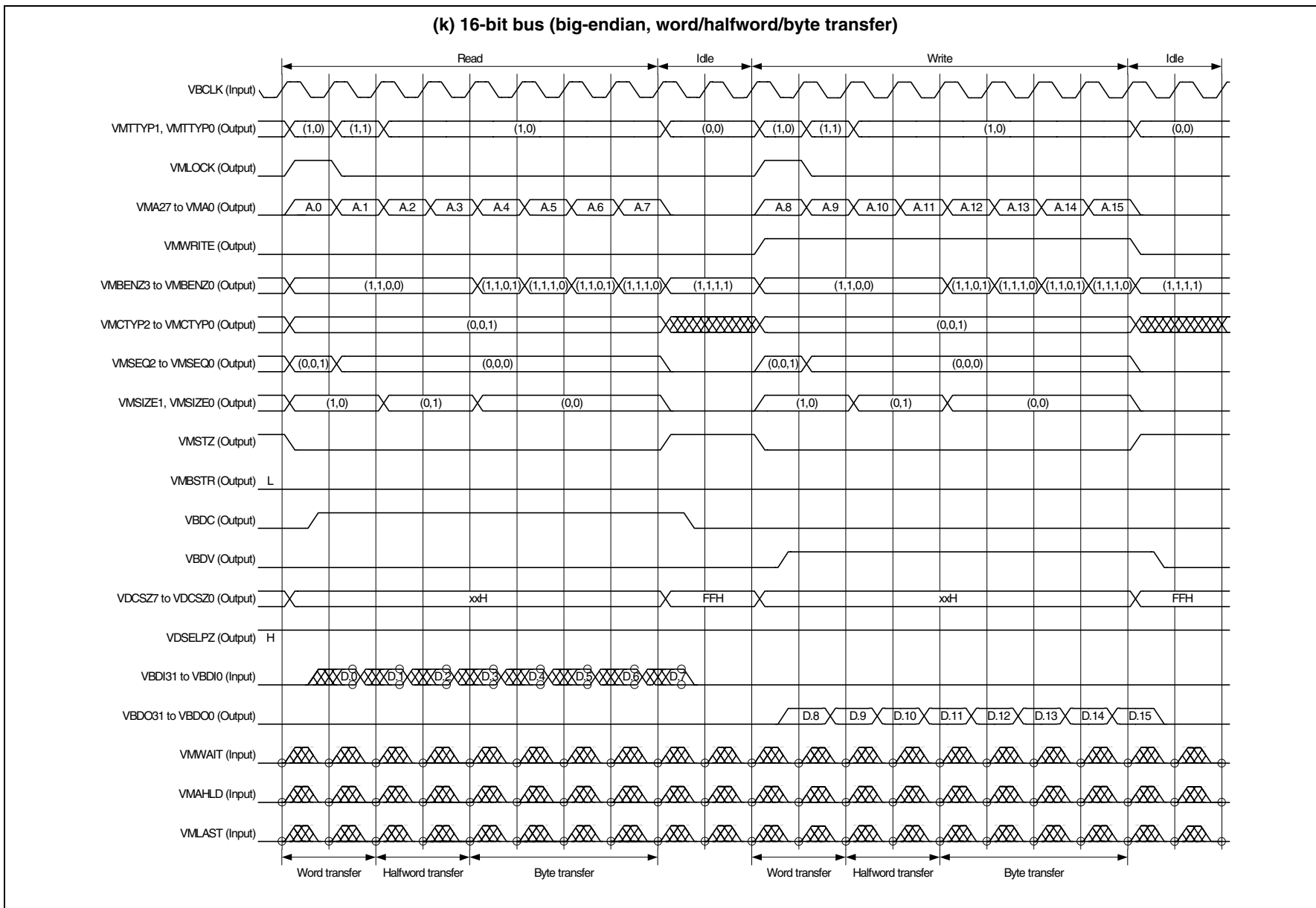
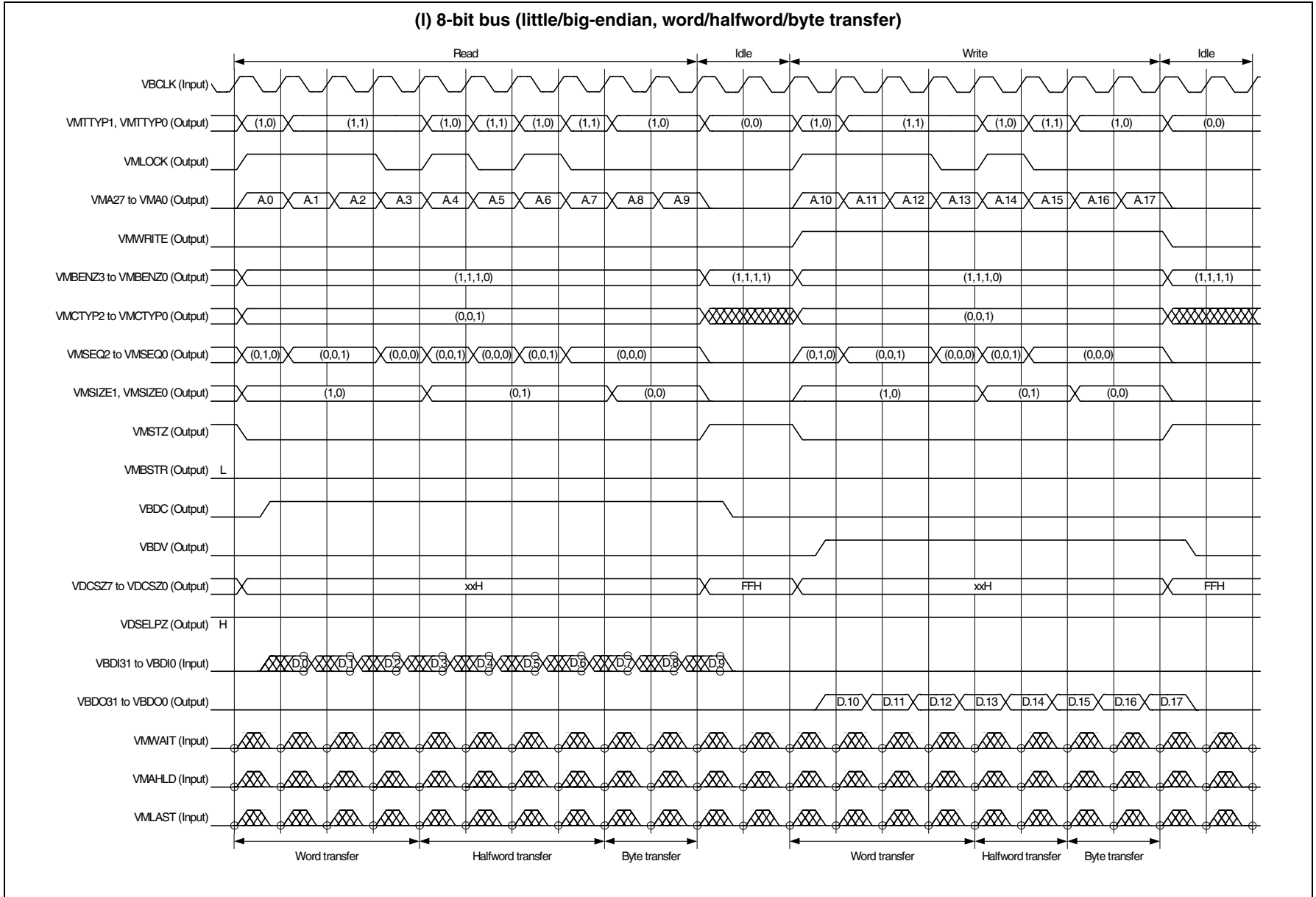


Figure 4-14. Read/Write Timing of Bus Slave Connected to VSB (12/12)



4.9.4 VSB read/write timing example

The read/write timing example of SRAM connected to the NT85E500 is shown below.

Figure 4-15. VSB Timing Example (1/2)

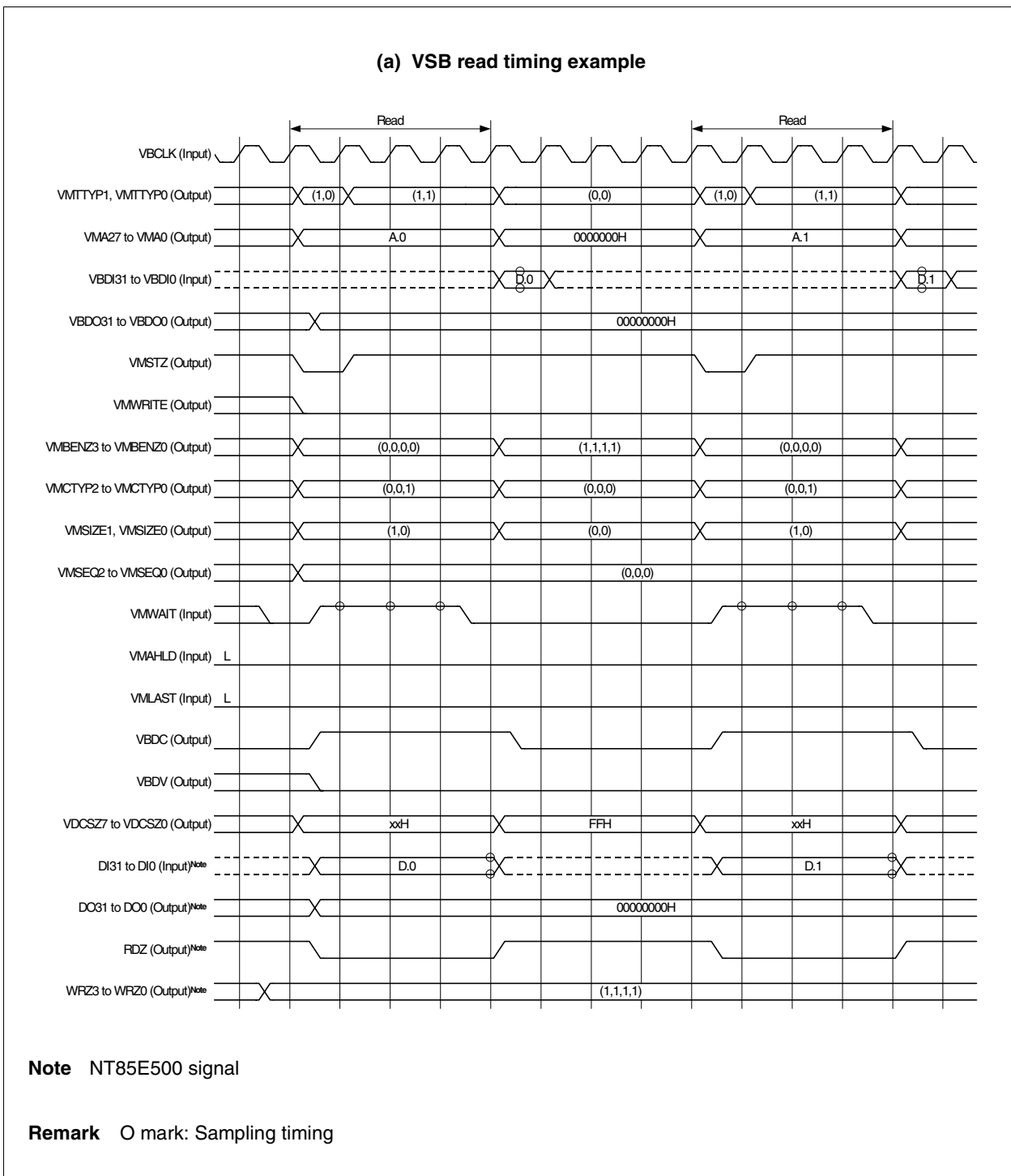
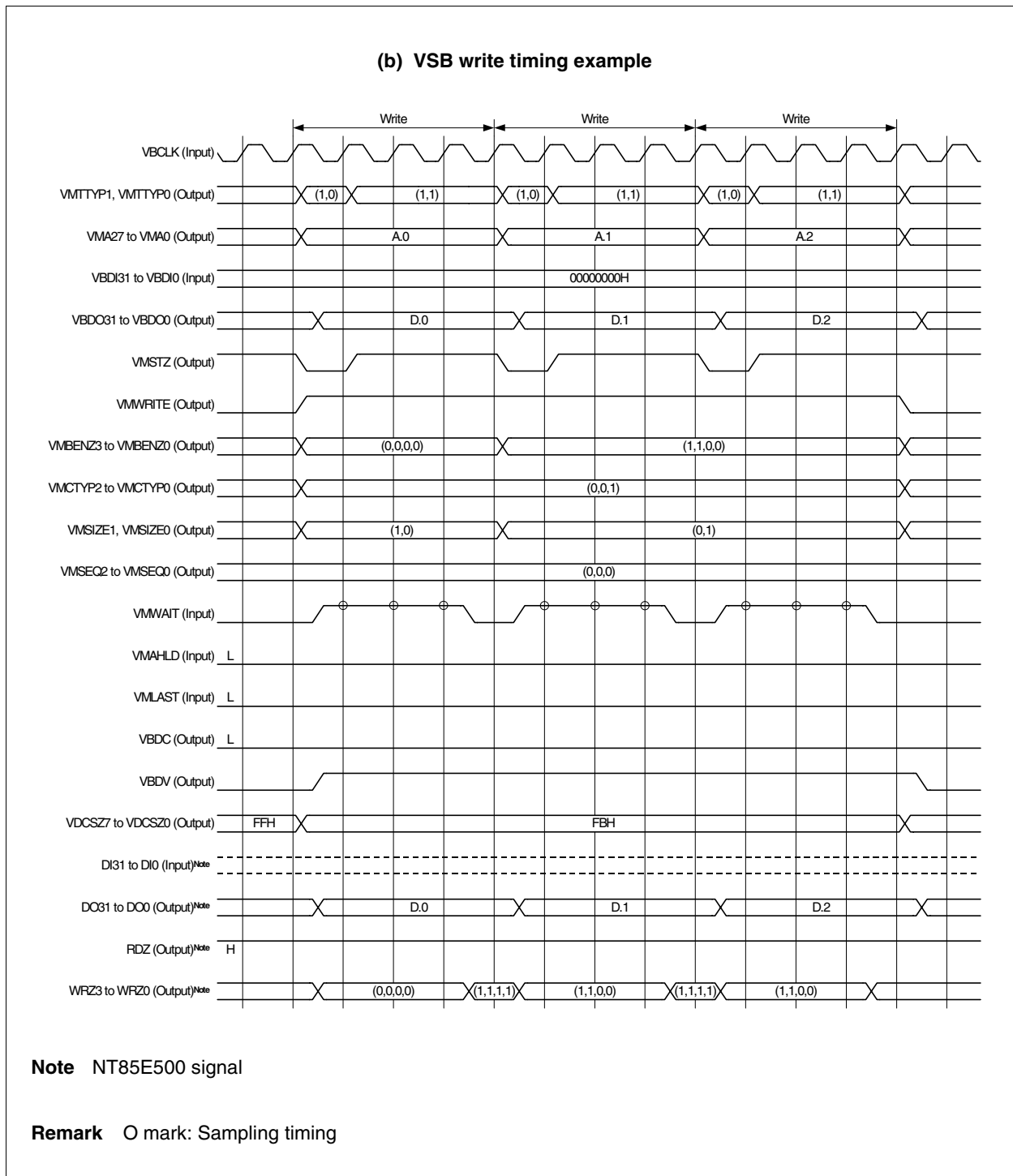


Figure 4-15. VSB Timing Example (2/2)

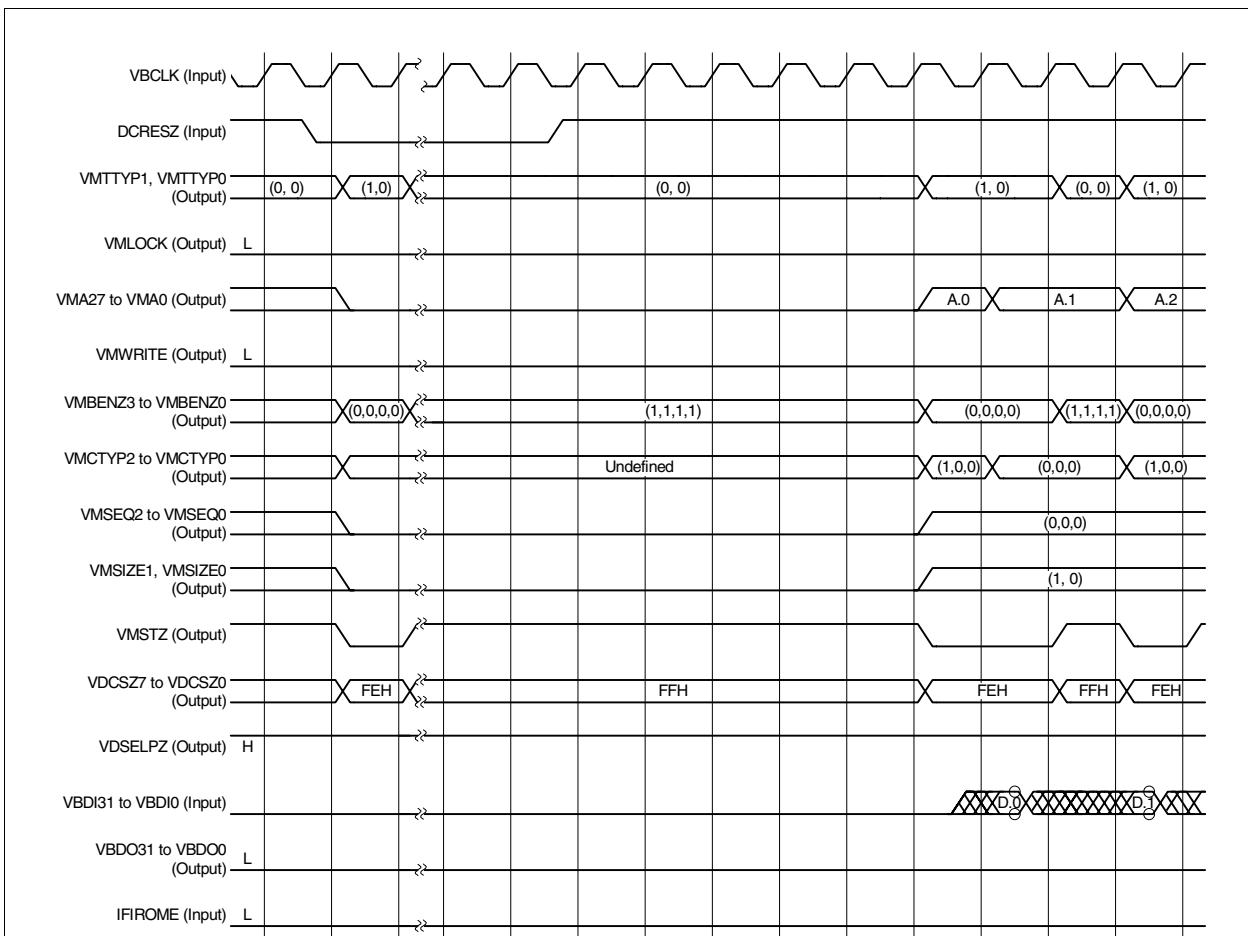


4.9.5 Reset timing

The reset timing of when a low level is input to the IFIROME pin (the connected ROM is used as external memory (via the VSB)) is shown below.

Caution Be sure to input the VBCLK signal continuously during the reset period (the period when DCRESZ is low level).

Figure 4-16. Reset Timing



Remarks 1. O mark: Sampling timing

- A.x: Arbitrary address output from the VMA27 to VMA0 pins
- D.x: Input data from address "A.x"
- XXX: Arbitrary input level

2. The timing seen from the NU85E when the NU85E has the bus access right is shown.

4.9.6 Bus master transition

There are five kinds of external bus cycles as shown below. Bus hold has the highest priority, followed by refresh cycle, DMA cycle, operand data access, and instruction fetch in that order.

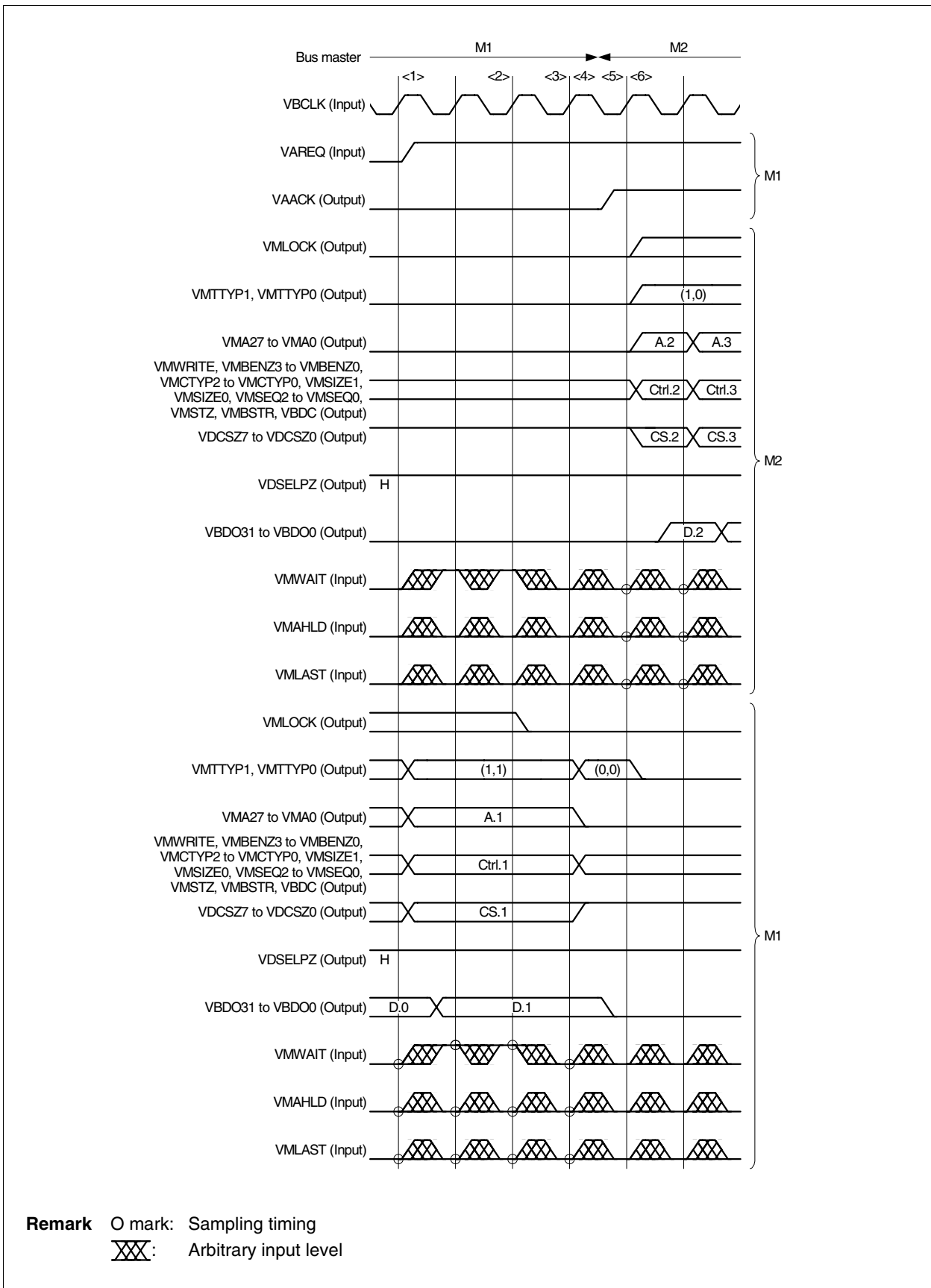
Priority	External Bus Cycle	Bus Master
High	Bus hold	External device
↑	Refresh cycle	SDRAM controller
	DMA cycle	DMA controller
↓	Operand data access	CPU
Low	Instruction fetch	CPU

The bus master transition procedure from master device (M1) which is operating as bus master to other master device (M2) is as follows.

- <1> M1, which operates as the bus master inputs a VSB access right request signal (VAREQ) from M2, another master device.
- <2> The bus arbiter within M1 goes into waiting for the ready response from the bus slave.
- <3> Upon completion of the current transfer, the bus slave returns a ready response.
- <4> The VMTTYP1 and VMTTYP0 signals of M1 indicate address-only transfer, and the VMLOCK, VDACSZ7 to VDACSZ0, and VDSELPZ signals are all ignored.
- <5> M1 returns an acknowledge signal (VAACK) for the VAREQ signal and a ready response to M2.
- <6> M2 becomes the bus master and data transfer on VSB starts.

Remark The ready response is when the VMWAIT, VMAHLD, and VMLAST signals are all in a low-level state.

Figure 4-17. Bus Master Transition Timing



4.9.7 Misalign access timing

The VSB access timing when misalign access is enabled (when a high level is input to the IFIMAEN pin) is shown below.

Figure 4-18. Misalign Access Timing (1/2)

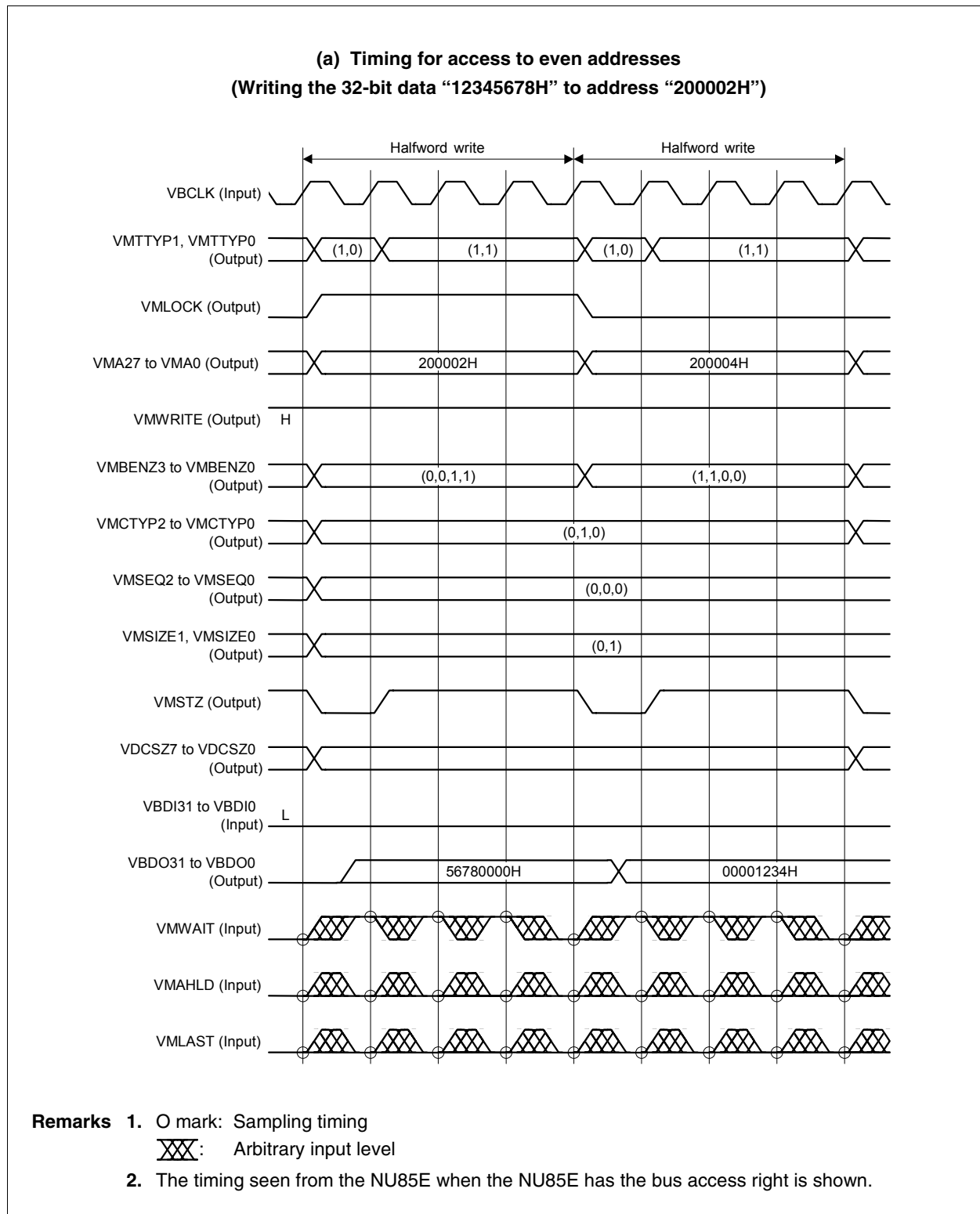
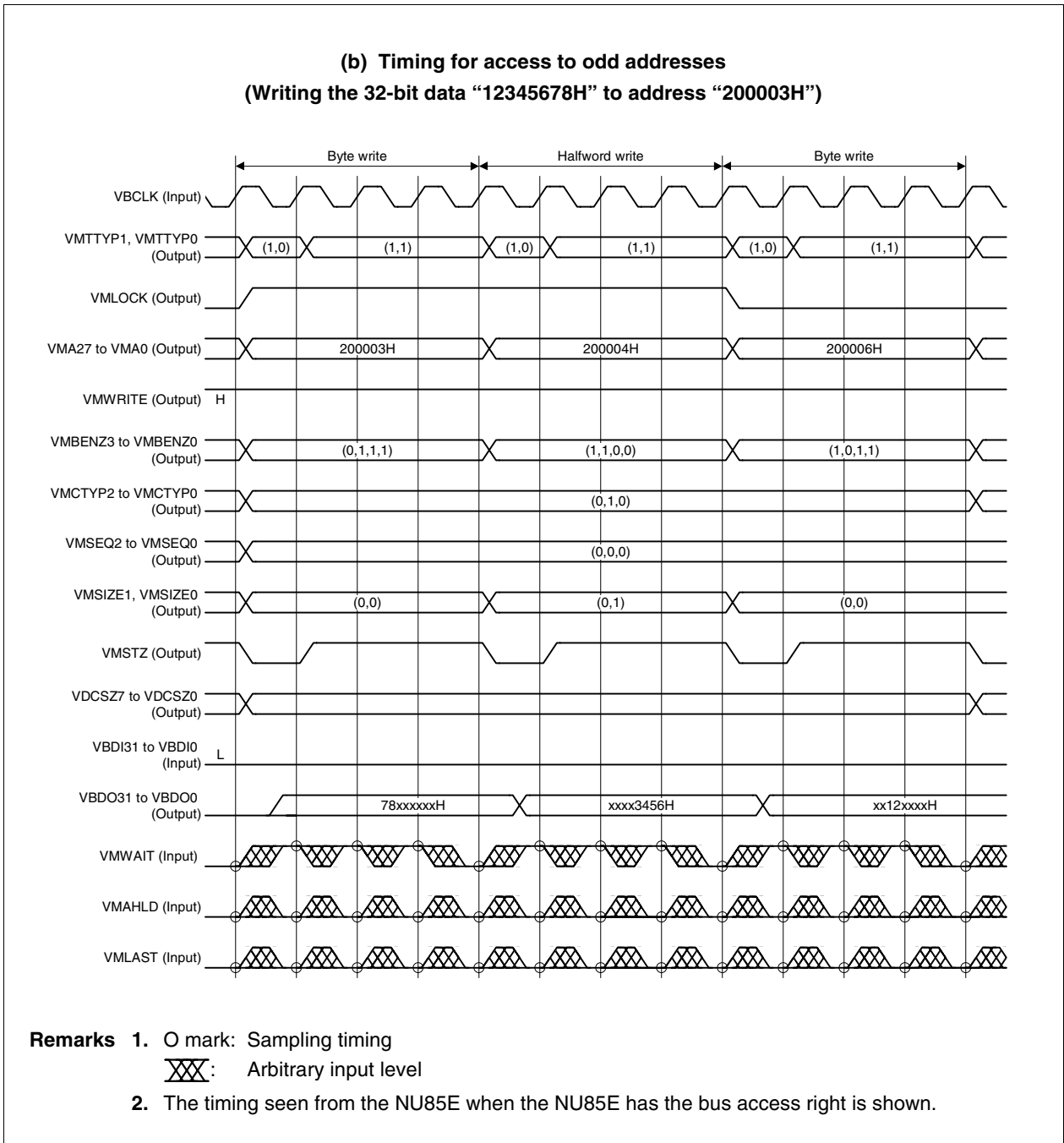


Figure 4-18. Misalign Access Timing (2/2)



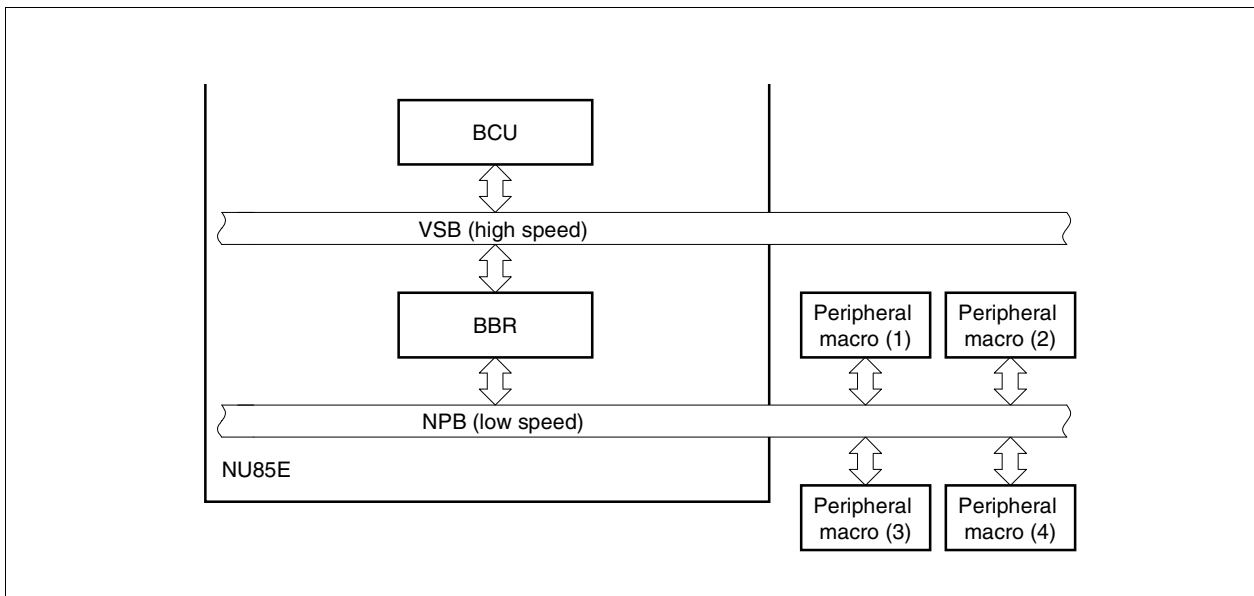
CHAPTER 5 BBR

The bus bridge (BBR) converts signals that are passed between the VSB and NPB.

The BBR sets up the following functions for peripheral macros that are connected to the NPB.

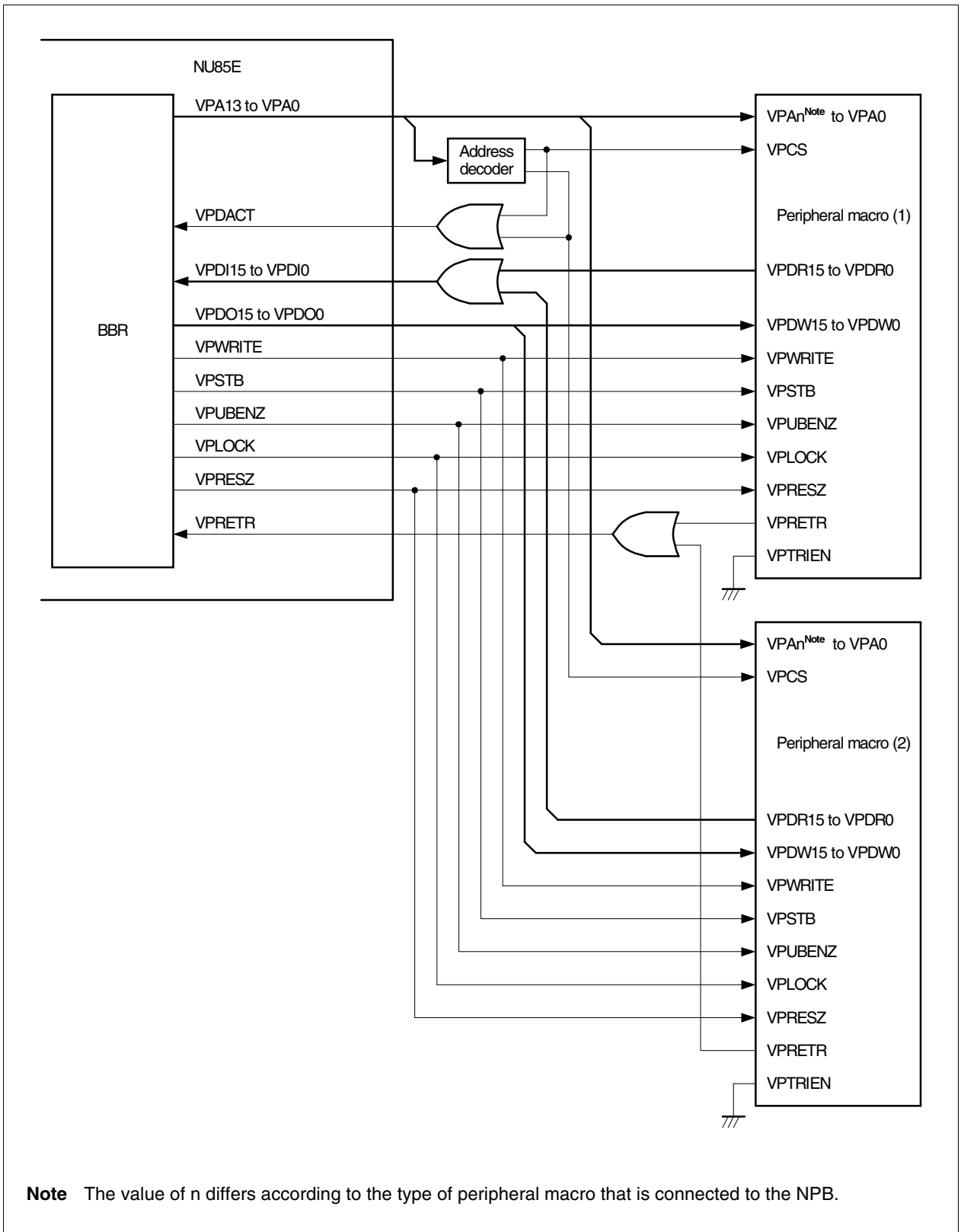
- Wait insertion function
- Retry function

Figure 5-1. NPB Connection Overview



The following figure shows a connection example connecting the NU85E and peripheral macros that are connected to the NPB.

Figure 5-2. NU85E and Peripheral Macro Connection Example



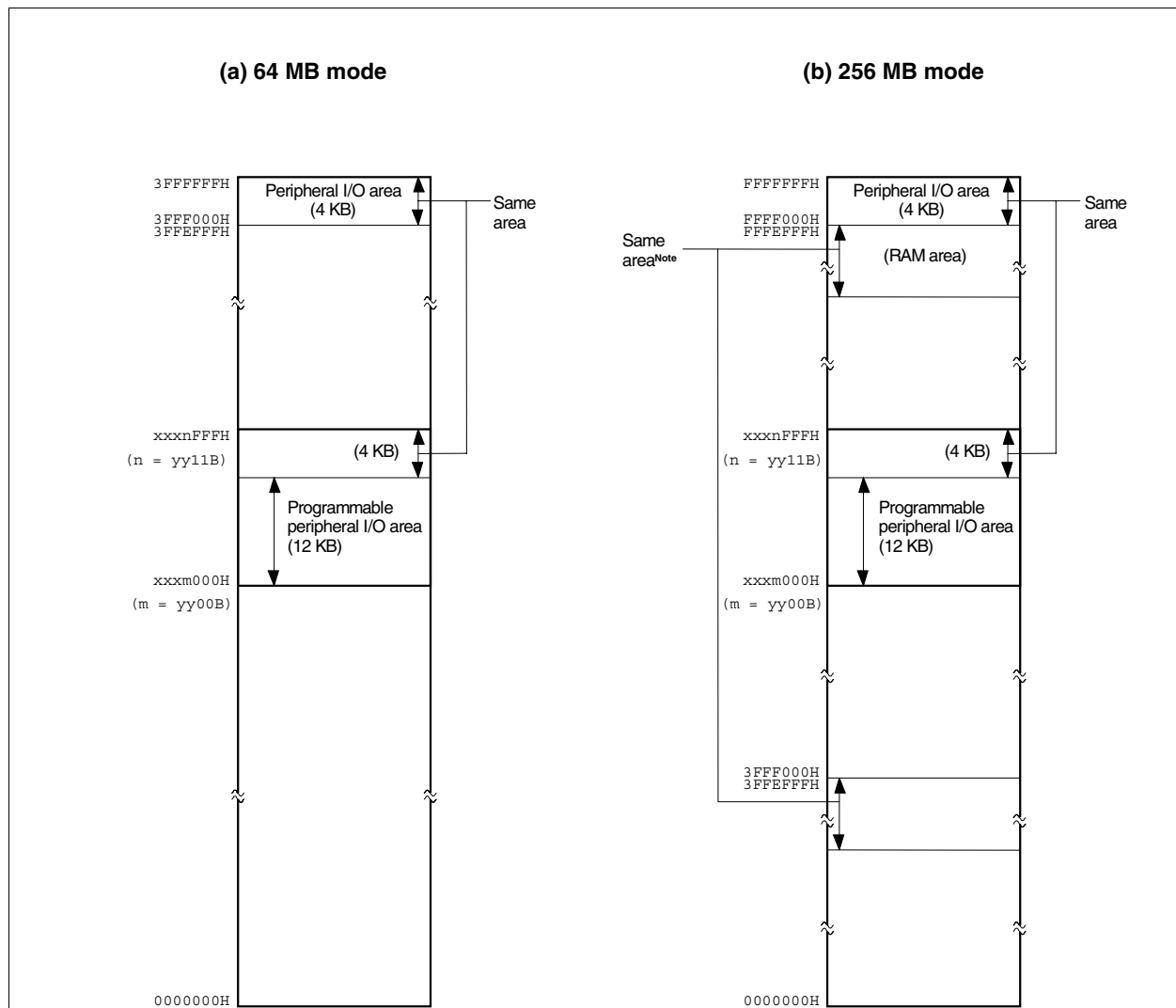
5.1 Programmable Peripheral I/O Area

The NU85E has a 4 KB peripheral I/O area that is allocated in advance in the address space and a 12 KB programmable peripheral I/O area that can be allocated at arbitrary addresses according to register settings (See 4.4 Programmable Peripheral I/O Area Selection Function).

If the peripheral I/O area or programmable peripheral I/O area in the memory map shown in Figure 5-3 is accessed, the NPB becomes active.

The programmable peripheral I/O area is set by the peripheral I/O area select control register (BPC).

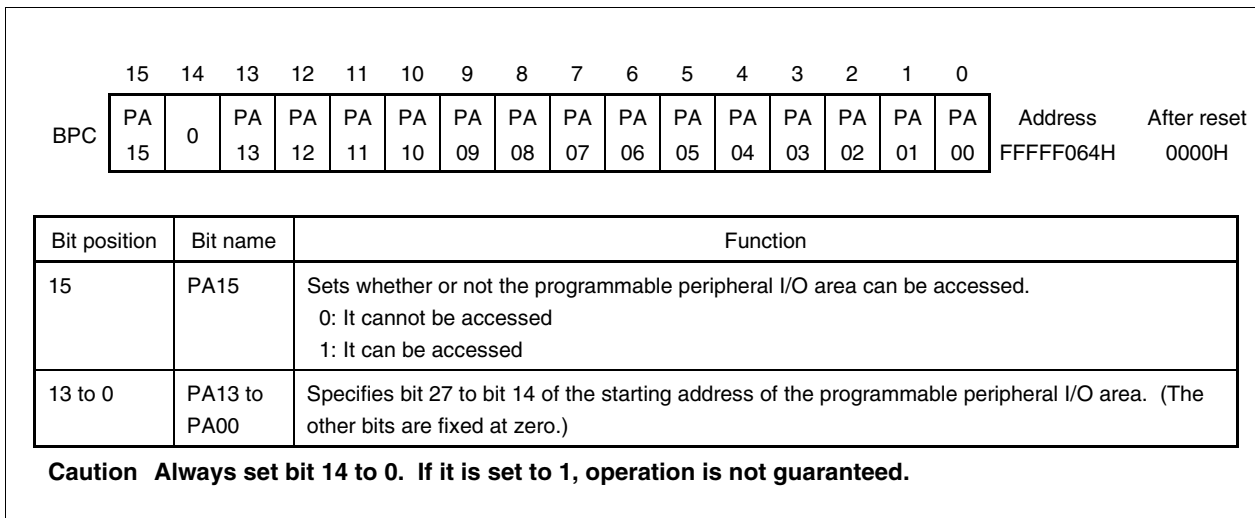
Figure 5-3. Peripheral I/O Area and Programmable Peripheral I/O Area



Note See Figure 3-8 Data Area (256 MB Mode).

- Remarks**
- xxx: Setting according to the PA13 to PA02 bits of the BPC register
 yy: Setting according to the PA01 and PA00 bits of the BPC register
 - Since the areas indicated by “same area” are linked, if data is written in one area, data having the same contents is also written in the other area.

Figure 5-4. Peripheral I/O Area Select Control Register (BPC)



Cautions 1. In 64 MB mode, if the programmable peripheral I/O area overlaps the following areas, the programmable peripheral I/O area becomes ineffective.

- Peripheral I/O area
- ROM area
- RAM area

2. In 256 MB mode, if the programmable peripheral I/O area overlaps the following areas, the programmable peripheral I/O area becomes ineffective.

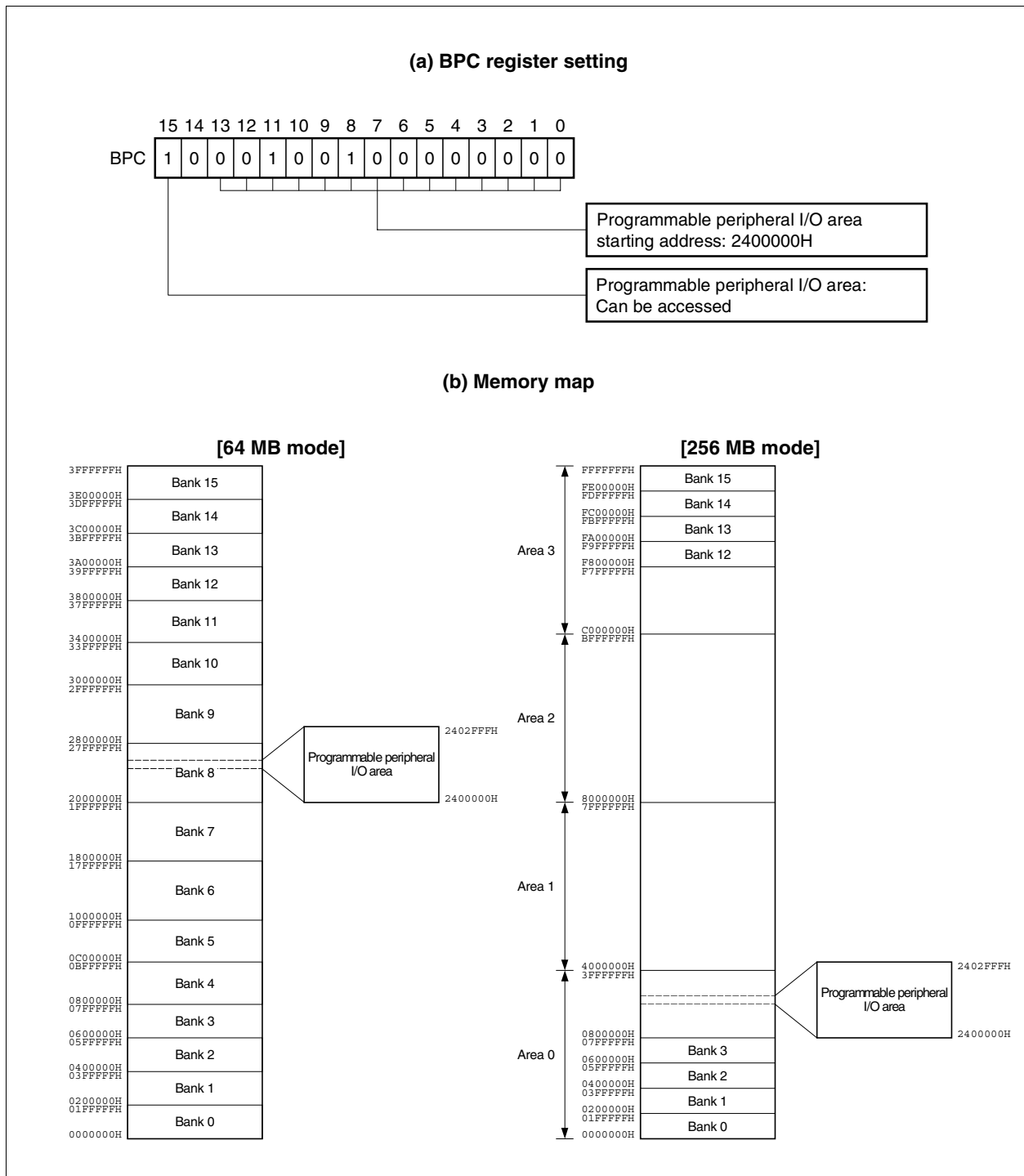
- Peripheral I/O area
- ROM area
- RAM area
- The area that is the same as the RAM area and that is located at address 3FFEFFFFH and below (See Figure 3-8 Data Area (256 MB Mode))

3. If no peripheral macros are connected to the NPB, no programmable peripheral I/O area need be set (Set the BPC register to its after-reset value).

4. The programmable peripheral I/O area address setting is enabled only once. Do not change addresses in the middle of a program.

Figure 5-5 shows a BPC register setting example and the memory map after the setting is made.

Figure 5-5. BPC Register Setting Example



5.2 Wait Insertion Function

The BBR is equipped with a wait insertion function for connection with low-speed peripheral macros that are connected to the NPB. The NPB strobe wait control register (VSWC) is used to set up this function.

The VSWC register sets the setup wait length and VPSTB wait length (see **Figure 5-6**). The number of waits can be set in the range from 0 to 7 clocks based on the internal system clock (VBCLK).

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

Figure 5-6. NPB Strobe Wait Control Register (VSWC) (1/2)

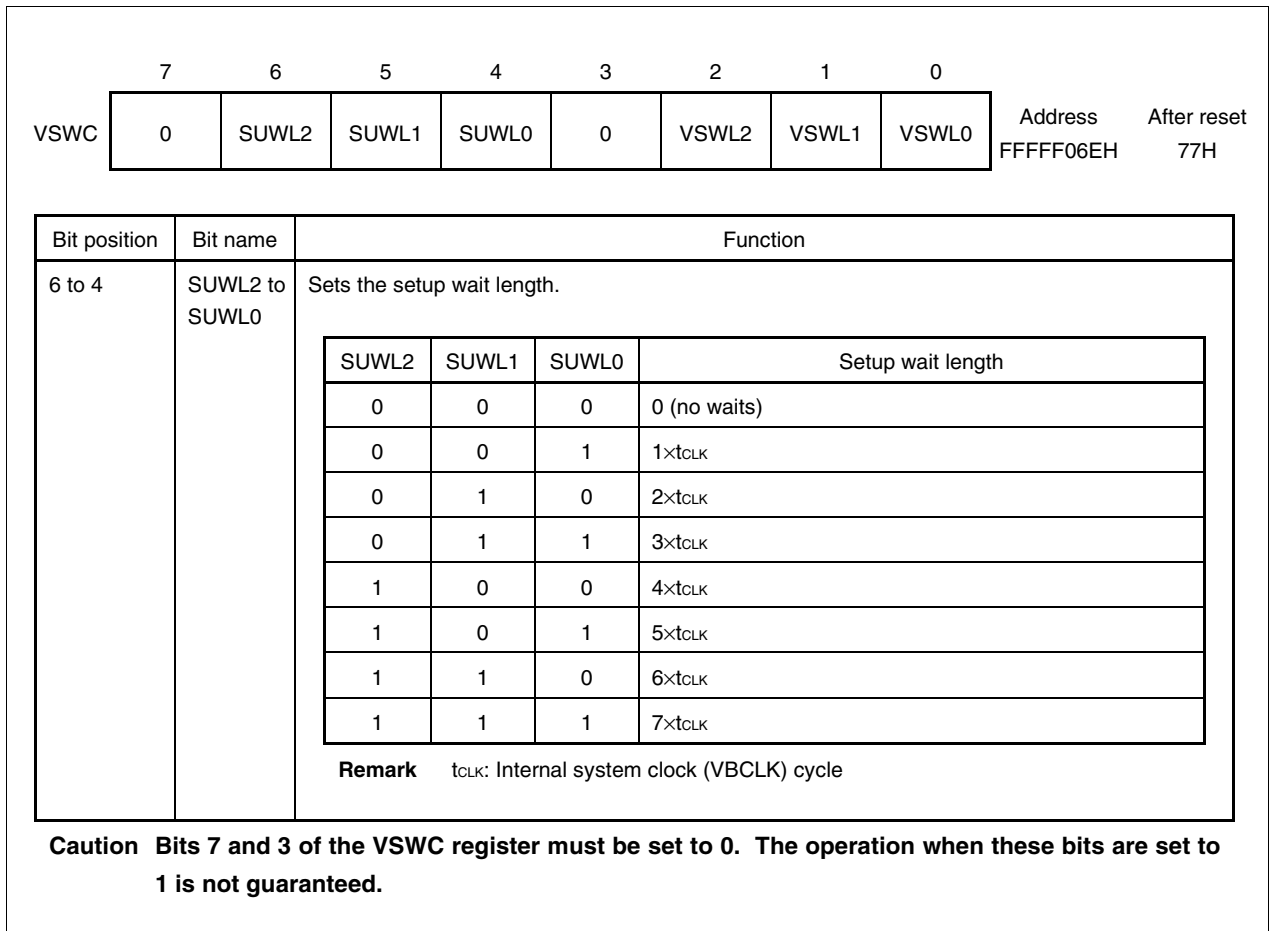
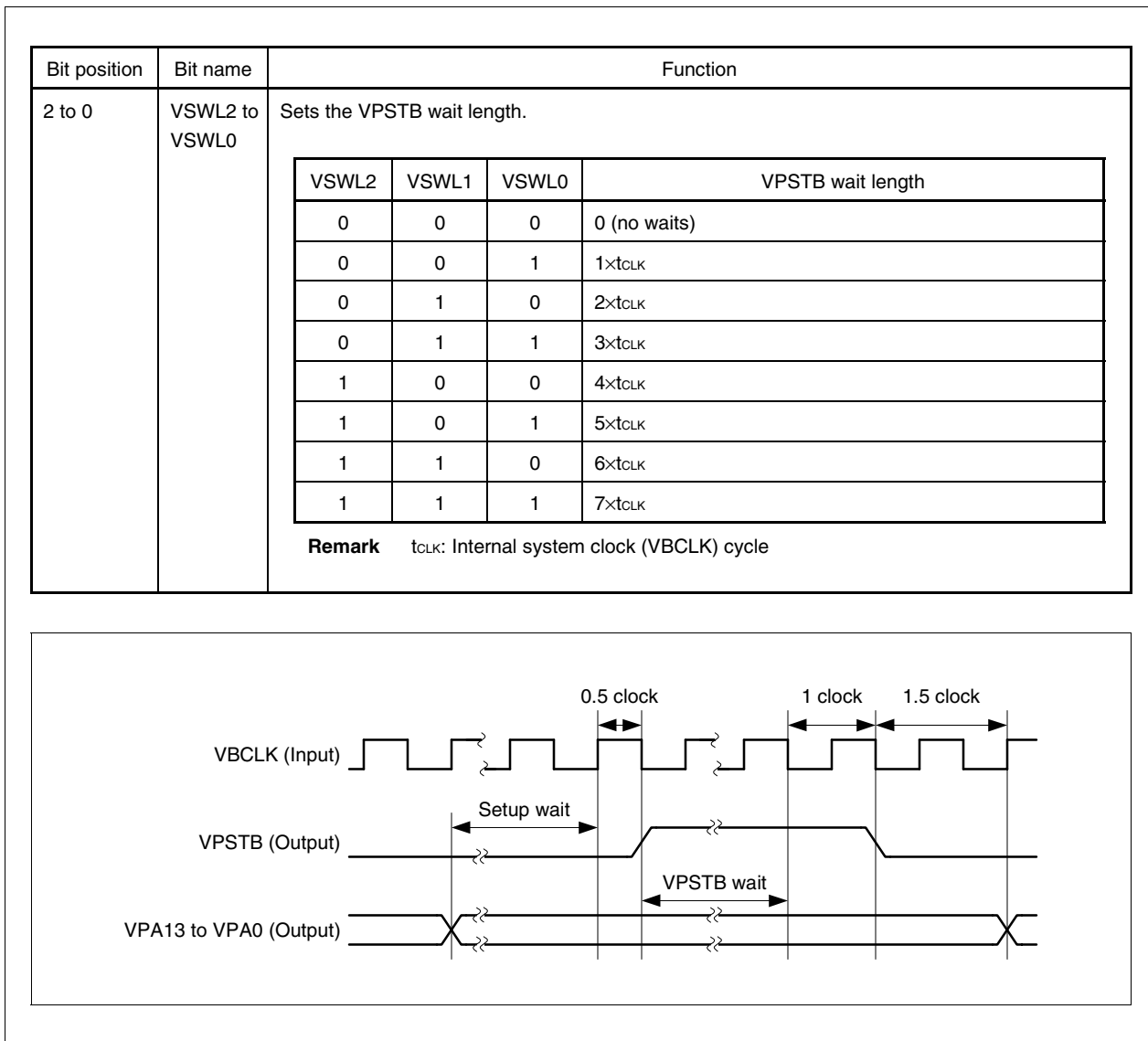


Figure 5-6. NPB Strobe Wait Control Register (VSWC) (2/2)



Be sure to set values for the setup wait and VPSTB wait lengths at each operation frequency that are the same as or greater than the number of waits shown in Table 5-1 below.

Table 5-1. Setting of Setup Wait, VPSTB Wait Lengths at Each Operation Frequency

Wait Length	Operation Frequency				
	To 25 MHz	To 33 MHz	To 50 MHz	To 81 MHz	To 100 MHz
Setup wait length (set using bits SUWL2 to SUWL0)	1	1	1	2	2
VPSTB wait length (set using bits VSWL2 to VSWL0)	1	2	4	5	6

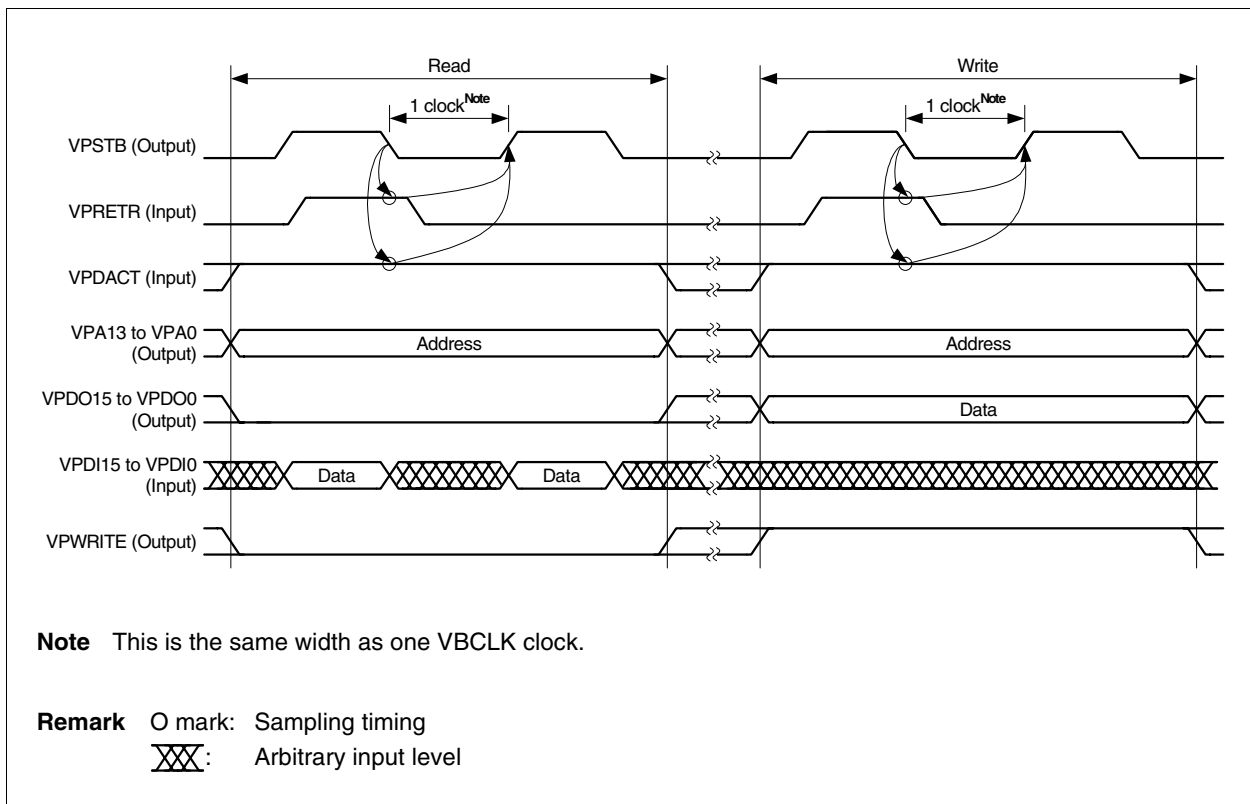
Caution These setting values are not guaranteed, so be sure to set the number of waits appropriate to the system after verifying operation.

5.3 Retry Function

The retry function, which repeats read or write processing according to a retry request signal (VPRETR) from a peripheral macro on the NPB, is used in situations such as when the data setup time is insufficient.

If a high-level signal is being input to the VPRETR and VPDACT pins at the falling edge of the VPSTB signal, the VPSTB signal rises again and the read or write operation is repeated.

Figure 5-7. Retry Function



5.4 NPB Read/Write Timing

Figure 5-8 to Figure 5-13 show the basic read/write timing of NPB, Figure 5-14 shows a timing example for read/write access to a bus slave connected to the NU85E and NPB, and Figure 5-15 shows a timing example of write access to a peripheral I/O register. Each one of these figures shows the timing as seen from the NU85E when the NU85E has the bus access right.

- Remark** O mark: Sampling timing
 A.x: Arbitrary address output from the VPA13 to VPA0 pins
 D.x: I/O data for address “A.x”
 XXX: Signal in undefined state (for output signal), arbitrary level (for input signal)

Figure 5-8. Halfword Access Timing

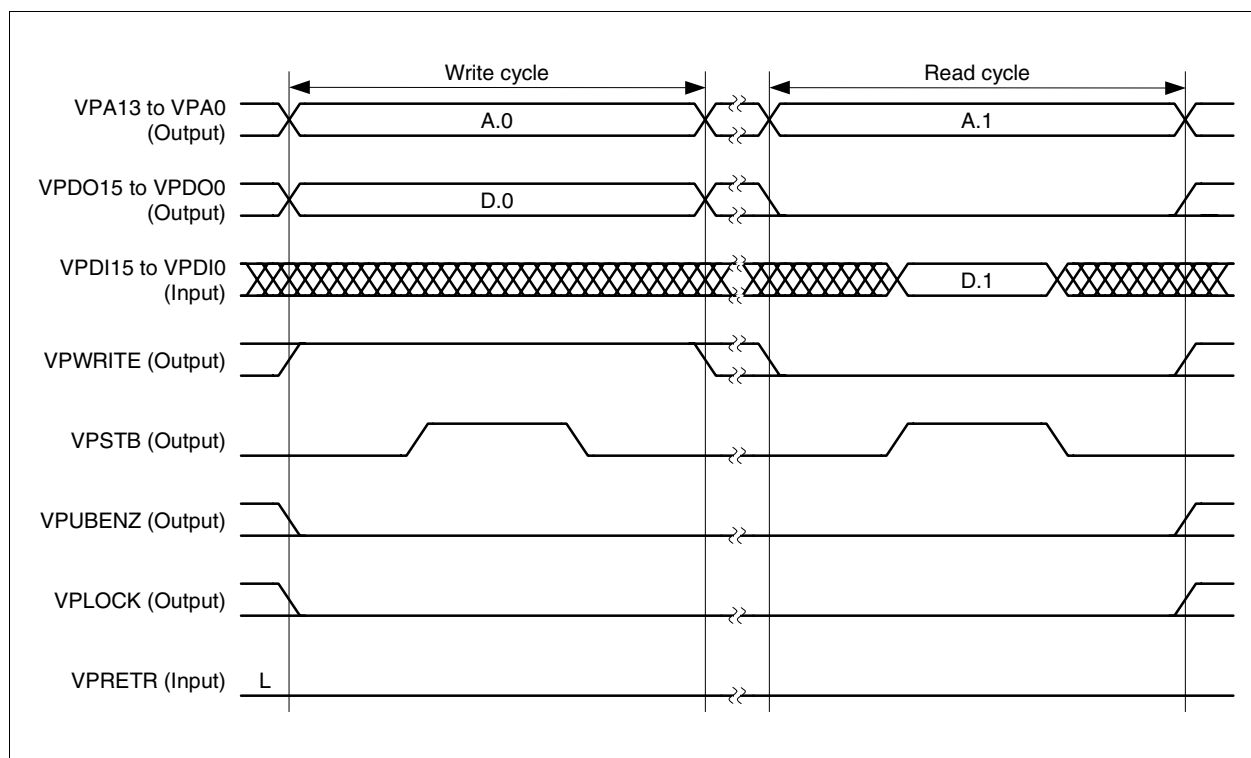


Figure 5-9. Timing of Byte Access to Odd Address

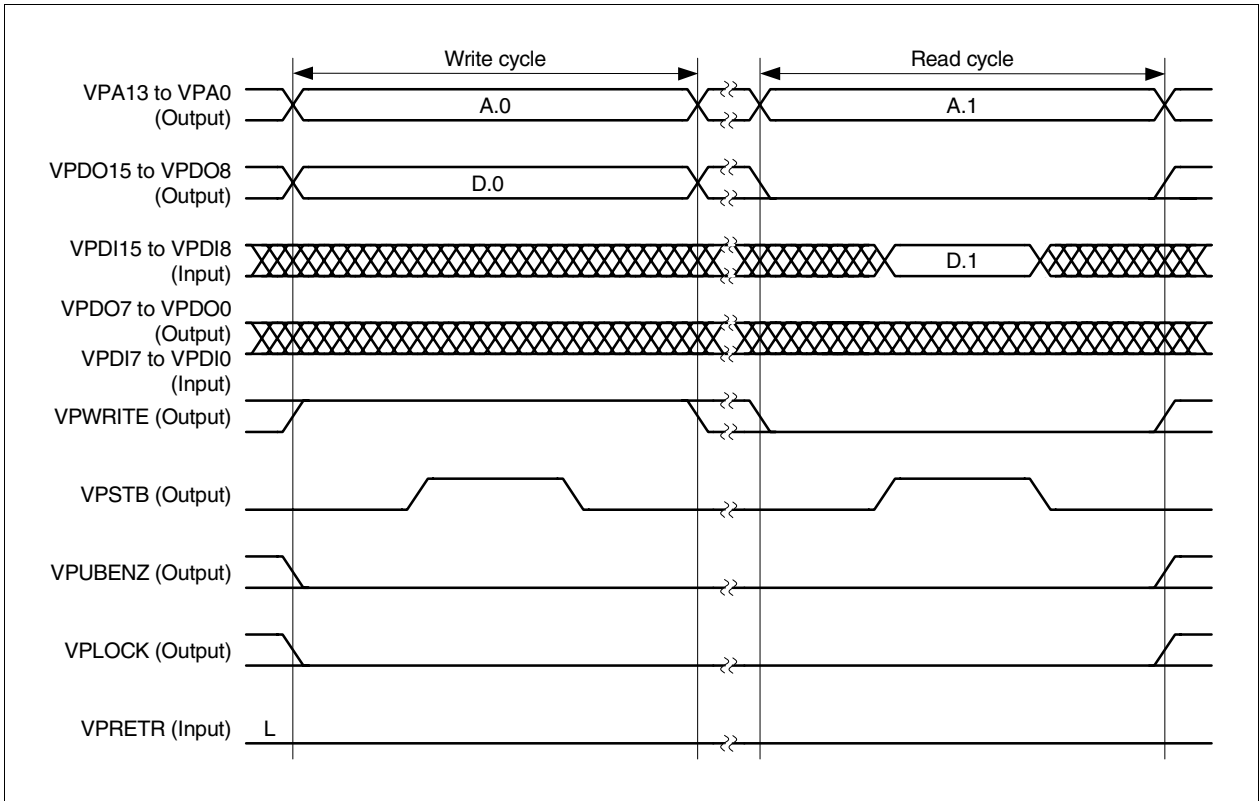


Figure 5-10. Timing of Byte Access to Even Address

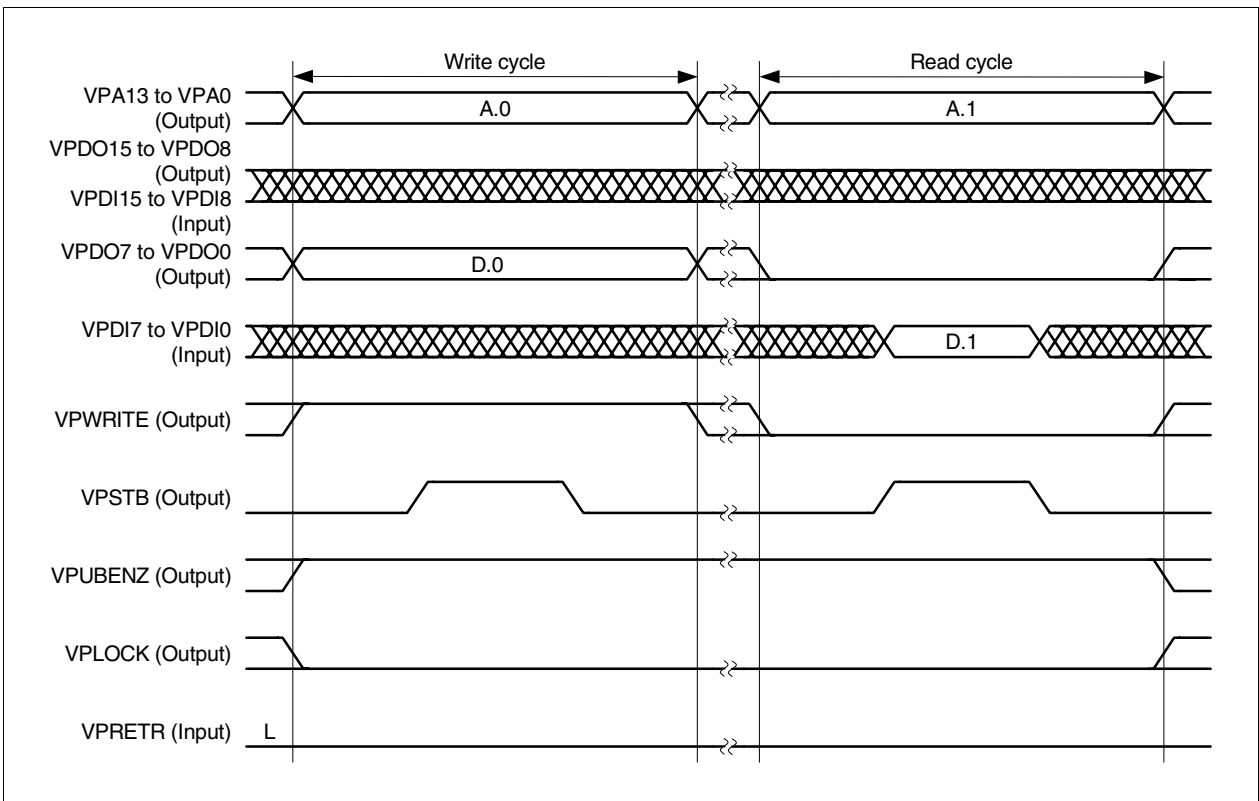


Figure 5-11. Read Modify Write Timing

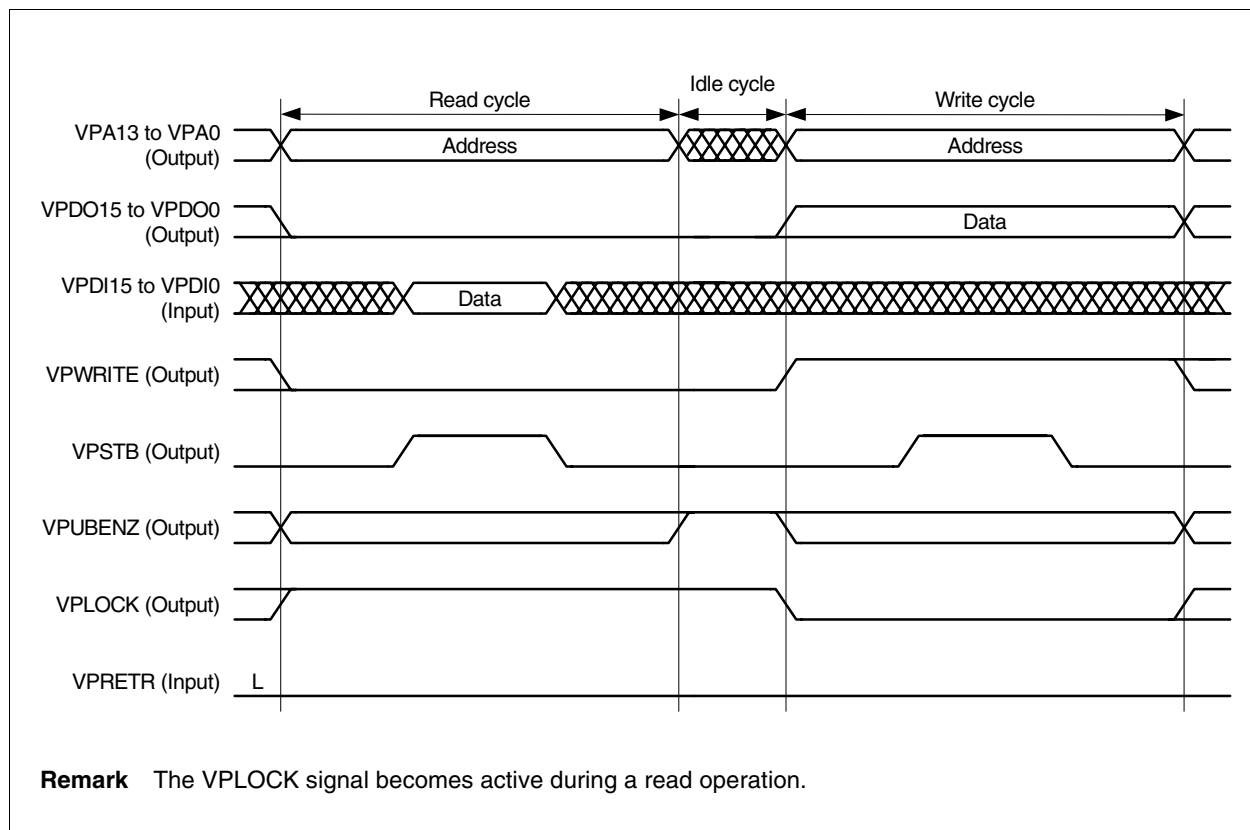


Figure 5-12. Retry Timing (Write)

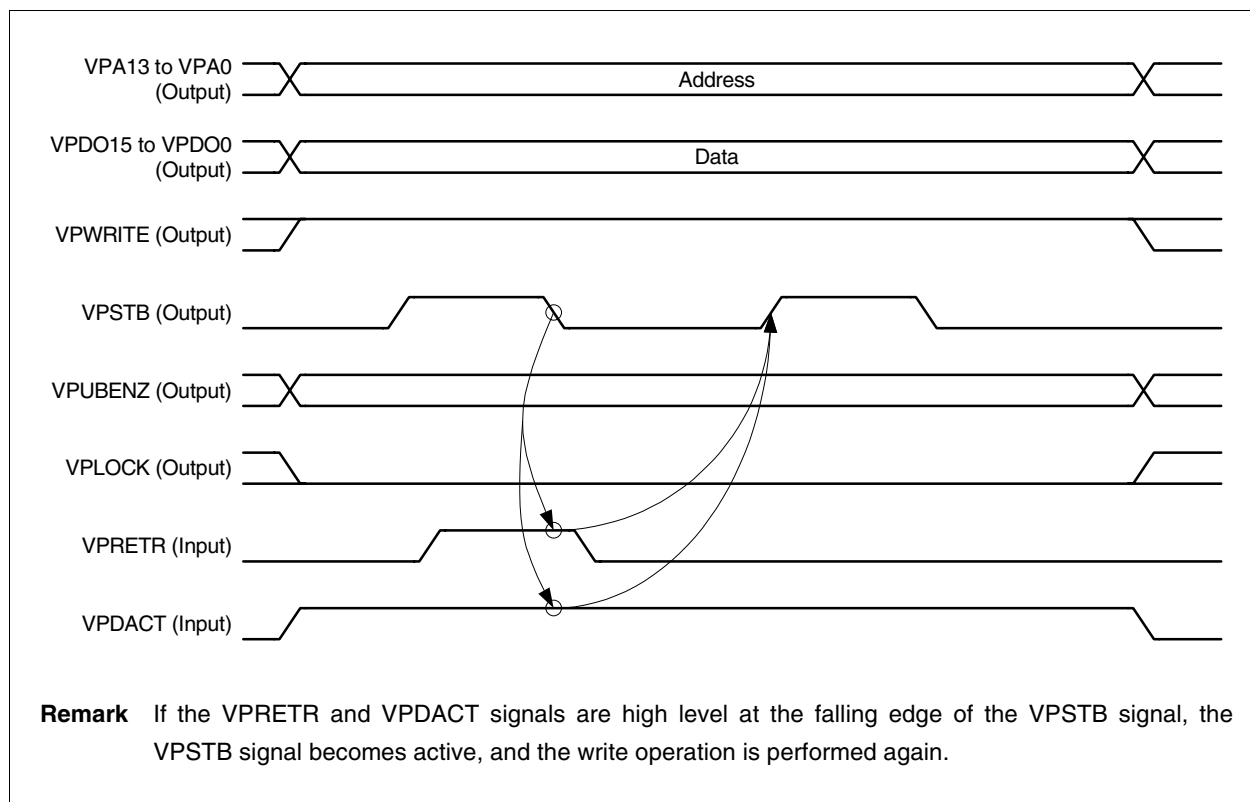


Figure 5-13. Retry Timing (Read)

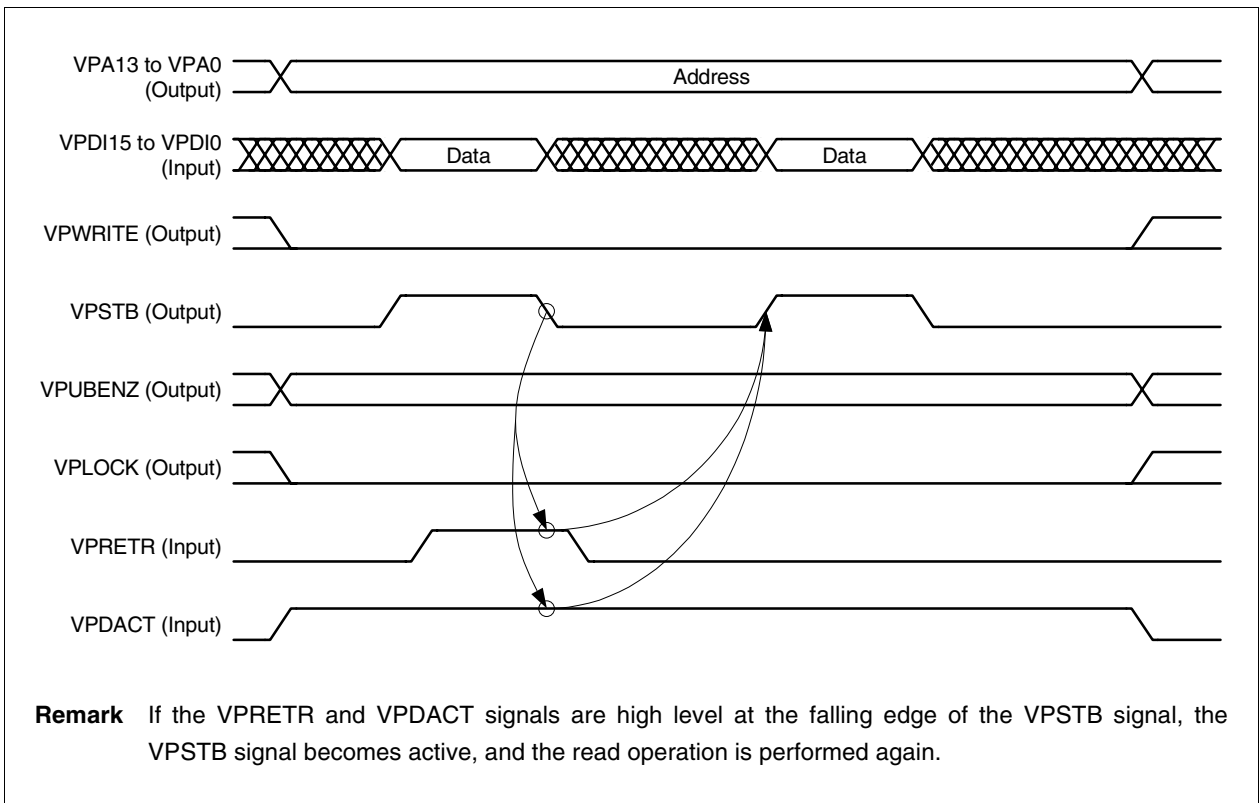


Figure 5-14. Read/Write Timing of Bus Slave Connected to NPB (1/4)

(a) Example of timing of word-data write to NPB peripheral macro (programmable peripheral I/O area)

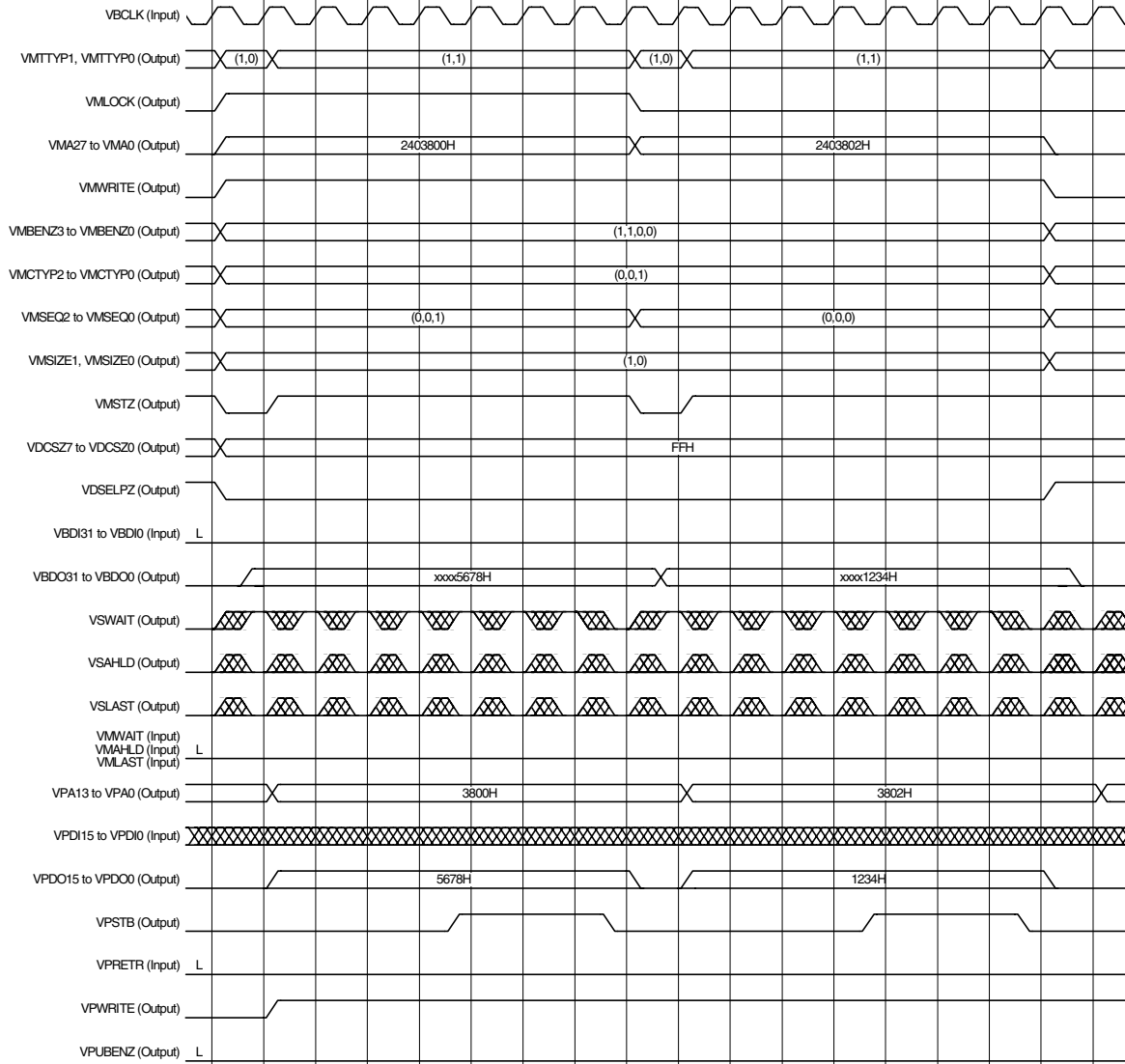


Figure 5-14. Read/Write Timing of Bus Slave Connected to NPB (2/4)

(b) Example of timing of halfword-data write to NPB peripheral macro (programmable peripheral I/O area)

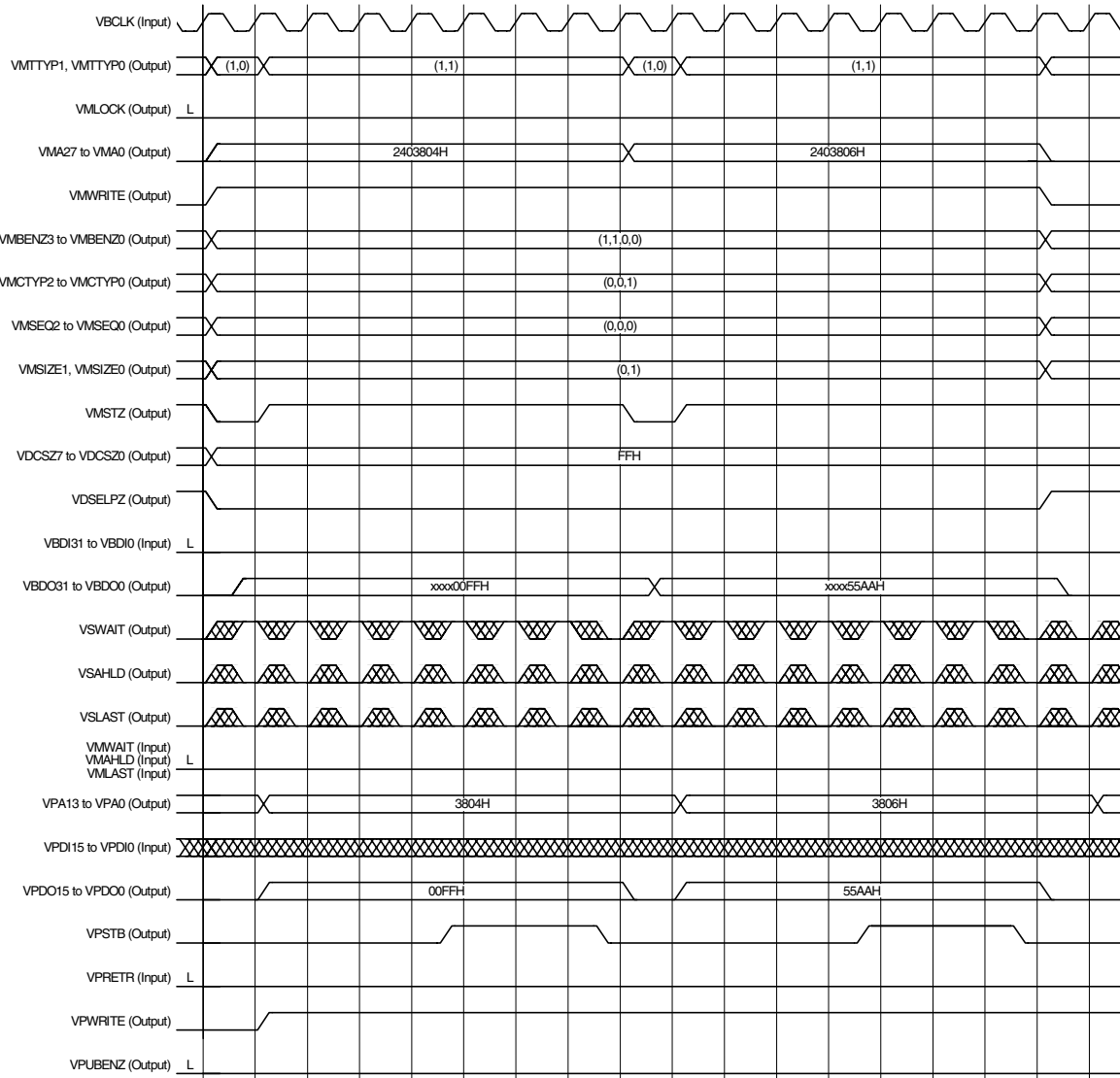


Figure 5-14. Read/Write Timing of Bus Slave Connected to NPB (3/4)

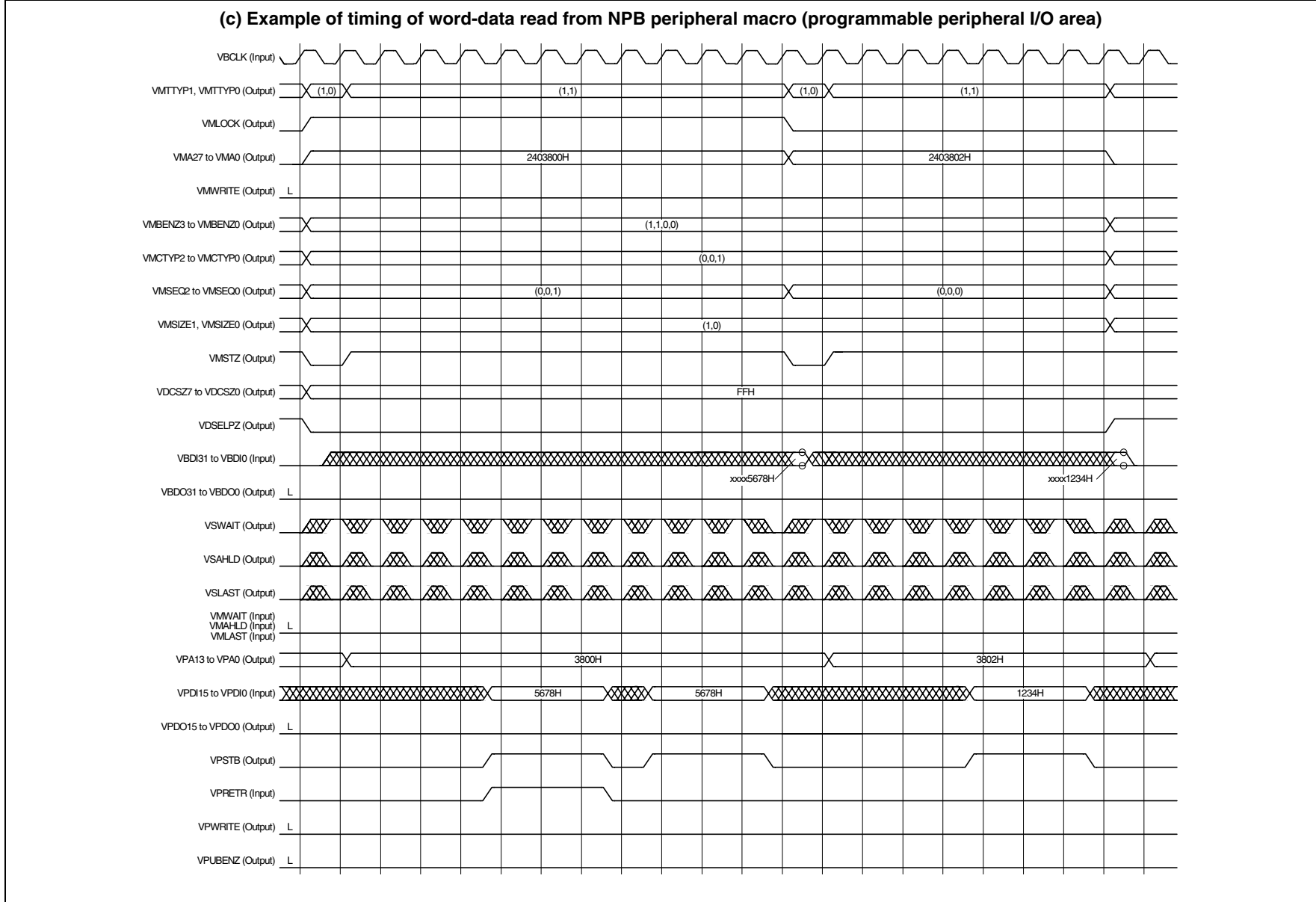


Figure 5-14. Read/Write Timing of Bus Slave Connected to NPB (4/4)

(d) Example of timing of halfword-data read from NPB peripheral macro (programmable peripheral I/O area)

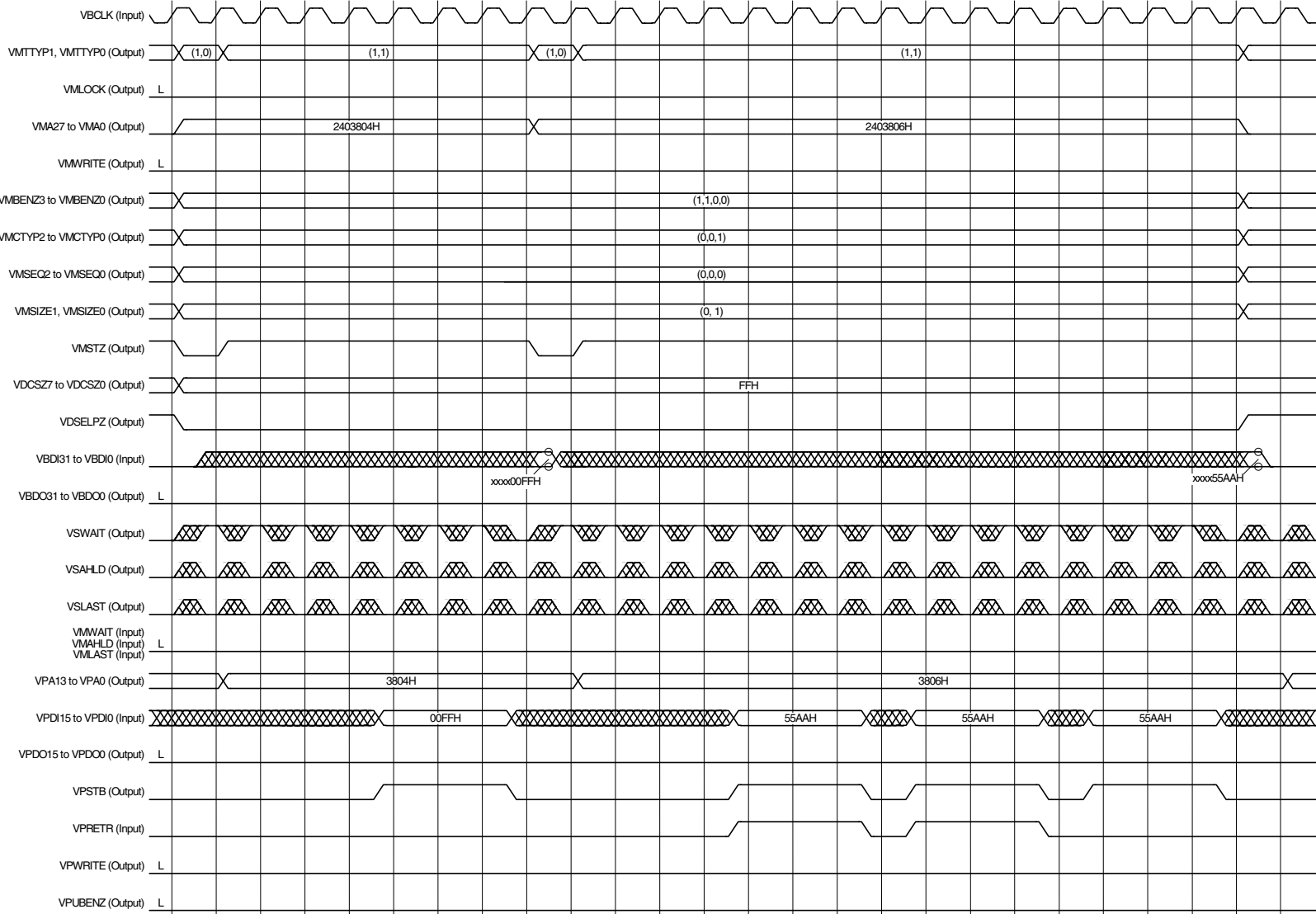
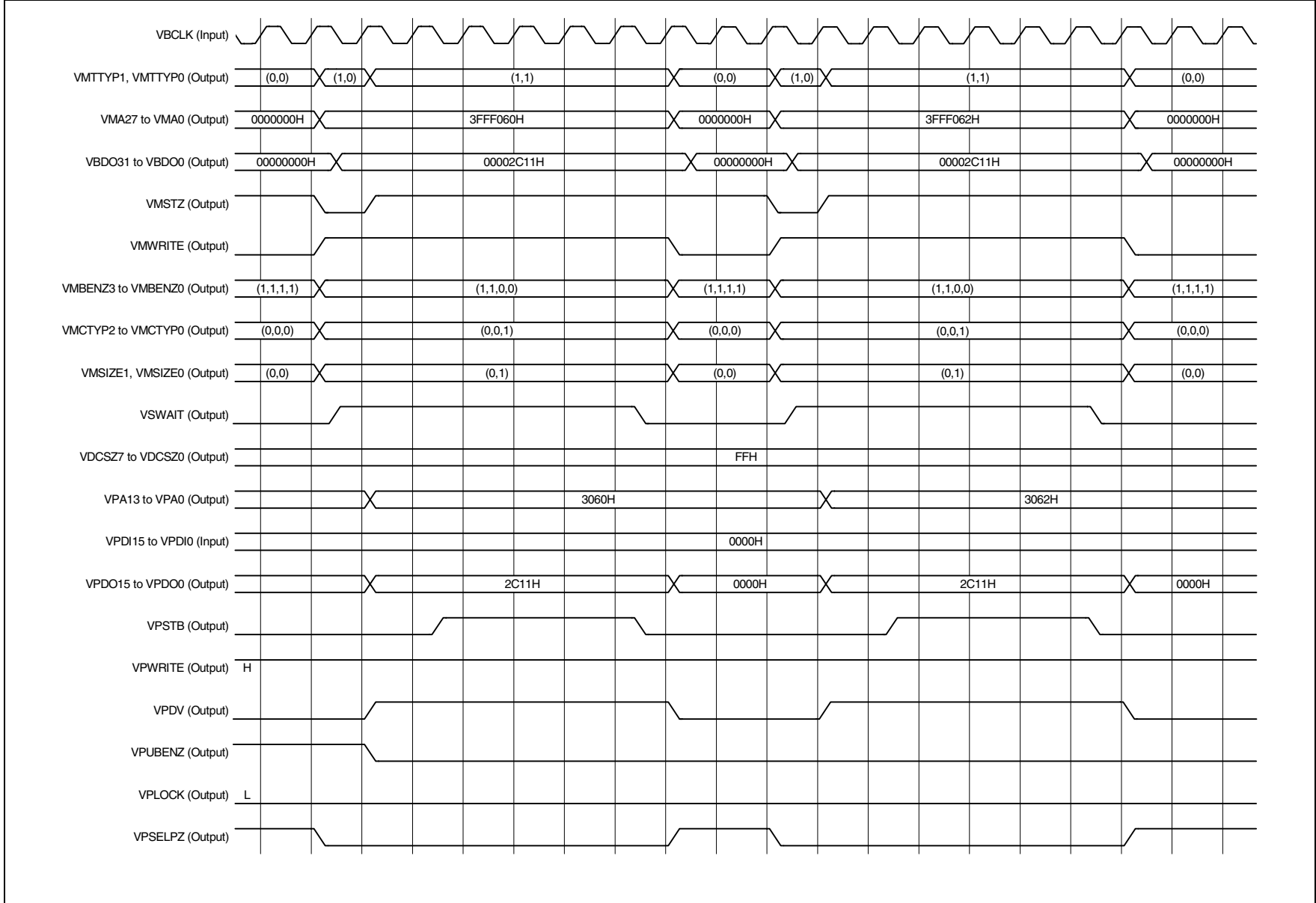


Figure 5-15. NPB Write Timing (Example of Timing of Data Write to CSC0 and CSC1 Registers)



★ 5.5 Precautions**• NPB access from external master to NU85E**

BBR does not provide a bus sizing function. Therefore, NPB access from the external bus master of the VSB to the NU85E as a slave must be executed with the bus size of the VSB set to 16 bits.

CHAPTER 6 STBC

The standby control unit (STBC) implements the various power save functions of the NU85E by controlling the external clock generator (CG).

6.1 Power Save Function

The power save function has the following three modes.

(1) HALT mode

This mode, which stops the supply of clocks only to the CPU, is set by executing a special-purpose instruction (HALT instruction). Since the supply of clocks to internal units other than the CPU continues, operation of the NU85E internal peripheral I/O that do not depend on the CPU instruction processing continues. The power consumption of the overall system can be reduced by intermittent operation that is achieved due to a combination of HALT mode and normal operation mode.

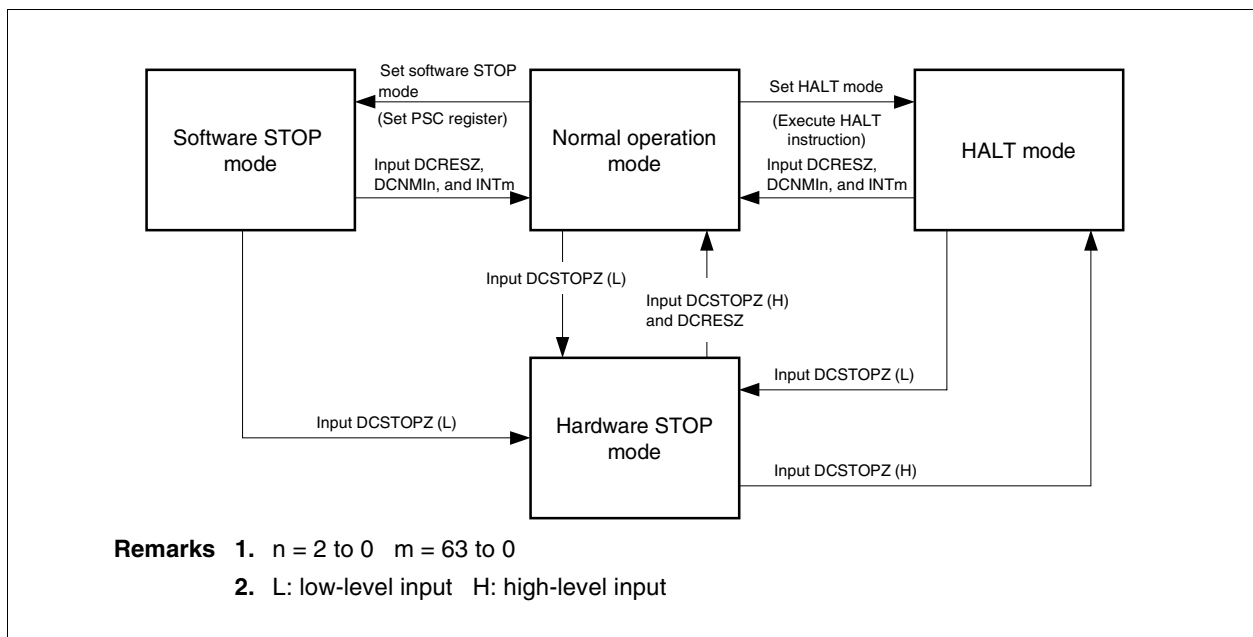
(2) Software STOP mode

This mode, which stops the overall system by stopping the external clock generator, is set by means of a PSC register setting. The system enters an ultra-low power consumption state in which only leak current is lost.

(3) Hardware STOP mode

This mode, which stops the overall system by stopping the external clock generator, is set by inputting the DCSTOPZ signal. The system enters an ultra-low power consumption state in which only leak current is lost.

Figure 6-1. Power Save Function State Transition Diagram



6.2 Control Registers

6.2.1 Power save control register (PSC)

The PSC is an 8-bit register that controls the power save function.

If interrupts are enabled according to the NMI2M to NMI0M and INTM bit settings, software STOP mode can be canceled by an interrupt request (except when interrupt servicing is disabled by the interrupt mask register (IMR0 to IMR3)).

Software STOP mode is specified by setting the STP bit.

This register can only be written by using a specific procedure so that its settings are not mistakenly overwritten due to erroneous program execution.

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

Caution Do not set the PSC register by transferring data using the DMAC. To set this register, always use a store instruction (ST or SST) or a bit manipulation instruction (SET1, CLR1, or NOT1 instruction).

Figure 6-2. Power Save Control Register (PSC)

	7	6	5	4	3	2	1	0	Address	After reset
PSC	NMI2M	NMI1M	NMI0M	INTM	0	0	STP	0	FFFF1FEH	00H

Bit position	Bit name	Function
7	NMI2M	Masks non-maskable interrupt requests (NMI2) from the DCNMI2 pin. ^{Note} 0: Enables NMI2 requests 1: Disables NMI2 requests
6	NMI1M	Masks non-maskable interrupt requests (NMI1) from the DCNMI1 pin. ^{Note} 0: Enables NMI1 requests 1: Disables NMI1 requests
5	NMI0M	Masks non-maskable interrupt requests (NMI0) from the DCNMI0 pin. ^{Note} 0: Enables NMI0 requests 1: Disables NMI0 requests
4	INTM	Masks maskable interrupt requests (INT63 to INT0) from the INT63 to INT0 pins. ^{Note} 0: Enables INT63 to INT0 requests 1: Disables INT63 to INT0 requests
1	STP	Specifies software STOP mode. When this bit is set (1), software STOP mode is set. When software STOP mode is canceled, this bit is automatically cleared (0).

Note The setting is valid in software STOP mode only.

- Cautions 1.** If the NMI2M to NMI0M and INTM bits are set (1) at the same time as the STP bit, the settings of the NMI2M to NMI0M and INTM bits are invalid. Therefore, if there are unmasked interrupt requests pending when software STOP mode is entered, be sure to set (1) those interrupt request bits (NMI2M to NMI0M and INTM) before setting (1) the STP bit.
- 2.** Because an interrupt request that occurs while the NMI2M to NMI0M and INTM bits are set (1) is invalid (it is not held pending), software STOP mode cannot be canceled.

Use the procedure shown below to set data in the PSC register.

- <1> Write the data that is to be set in the PSC register to an arbitrary general-purpose register (see **3.2.1 Program registers**).
- <2> Use the store instruction (ST or SST instruction) to write the contents of the general-purpose register, which had been prepared in step <1>, to the command register (PRCMD).
- <3> Use the following instructions to write the contents of the general-purpose register, which had been prepared in step <1>, to the PSC register (Do this immediately after writing the contents of the general-purpose register in the PRCMD register).
 - Store instruction (ST or SST instruction)
 - Bit manipulation instruction (SET1, CLR1, or NOT1 instruction)
- <4> If the NU85E switches to software STOP mode, insert NOP instructions (five or more instructions).

Examples 1.

```

<1> mov    0x02, r11
      movea base_address, r0, r20 ; base_address = FFFF000H
<2> st.b   r11, PRCMD[r20]      ; PRCMD = 01FCH
<3> st.b   r11, PSC[r20]       ; PSC = 01FEH
<4> nop
      nop
      nop
      nop
      nop

```

2.

```

<1> mov    0x02, r11
      movea 0xF1FCH, r0, r20
      movea 0xF1FEH, r0, r21
<2> st.b   r11, 0x0[r20]       ; r20 = FFFFF1FCH (= PRCMD)
<3> st.b   r11, 0x0[r21]       ; r21 = FFFFF1FEH (= PSC)
<4> nop
      nop
      nop
      nop
      nop

```

No special procedure is required to read the contents of the PSC register.

- Remarks**
1. Interrupts are not acknowledged for store instructions for the PRCMD register.
 2. Steps <2> and <3> above are assumed to occur consecutively. If another instruction is placed between the instructions described in steps <2> and <3>, then when the interrupt is acknowledged for that instruction, the setting may not be established, causing abnormal operation.
 3. Although the data written in the PRCMD register is dummy data, use the same value (data) as the value of the general-purpose register used for setting data in a specific register (step <3> in the examples above) even when writing to the PRCMD register (step <2> in the examples above). This is similar to using a general-purpose register for addressing.
 - ★ 4. To enable interrupts immediately after the software STOP mode is entered, insert the EI instruction between the <1> mov and <2> st.b instruction

★ **Remarks 5.** The following shows the operation when a non-maskable interrupt or maskable interrupt is requested while a NOP instruction is being executed.

- If a non-maskable or maskable interrupt is requested before SWSTOPRQ becomes active, the interrupt servicing is immediately executed.
- If a non-maskable or maskable interrupt is requested after SWSTOPRQ became active, the STOP mode is canceled by the requested interrupt after the STOP mode is entered, in the same way as cancellation by a normal interrupt.

6.2.2 Command register (PRCMD)

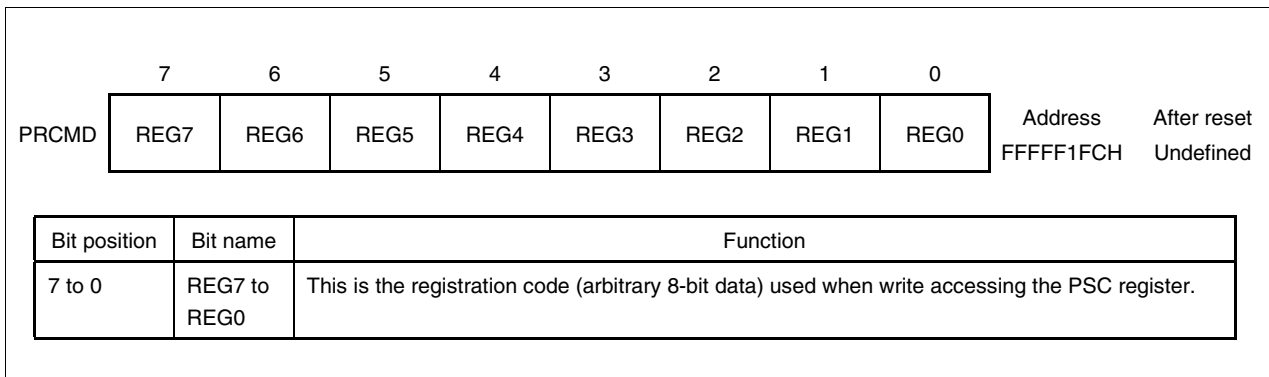
The command register (PRCMD) is used to set protection for write operations to the PSC register so that the application system is not halted unexpectedly due to erroneous program execution.

Only the first write operation to the PSC register is valid after a registration code (arbitrary 8-bit data) is written to the PRCMD register. Since the register value can be rewritten only by a predetermined procedure, illegal write operations to the PSC register are rejected.

Data can be written in the PRCMD register only in 8-bit units. During reading, the value is undefined.

Caution Do not set the PRCMD register by transferring data using the DMAC. To set this register, always use a store instruction (ST or SST).

Figure 6-3. Command Register (PRCMD)



6.3 HALT Mode

In HALT mode, the operation clock of the CPU is stopped. Since the supply of clocks to internal units other than the CPU continues, operation continues. The power consumption of the overall system can be reduced by setting the NU85E to HALT mode while the CPU is idle.

(1) Setting and operation status

The NU85E is switched to HALT mode by the HALT instruction.

Although program execution stops in HALT mode, the contents of all registers and of RAM immediately before HALT mode began are maintained. Also, operation continues for all NU85E-internal peripheral I/O that does not depend on CPU instruction processing.

Caution Insert at least five NOP instructions after the HALT instruction.

(2) Cancellation of HALT mode

HALT mode is canceled by a non-maskable interrupt request, an unmasked maskable interrupt request, or the input of the DCRESZ signal.

(a) Cancellation by interrupt request

HALT mode is canceled by a non-maskable interrupt request or by an unmasked maskable interrupt request regardless of the priority. The following table shows the operation performed after HALT mode is canceled.

Table 6-1. Operation After HALT Mode Is Canceled by Interrupt Request

Cancellation Source	Interrupt Enabled (EI) State	Interrupt Disabled (DI) State
Non-maskable interrupt request	Branch to handler address	
Maskable interrupt request	Branch to handler address or execution of next instruction	Execution of next instruction

The operation differs as follows if HALT mode was set within the interrupt servicing routine.

<1> When a low priority interrupt request is generated

Only HALT mode is canceled. The interrupt request is not acknowledged (pending).

<2> When a high priority interrupt request (including a non-maskable interrupt request) is generated

HALT mode is canceled and the interrupt request is acknowledged.

(b) Cancellation by DCRESZ signal input

This is the same as a normal reset operation.

Caution Be sure to input the DCRESZ signal so that the setup and hold times referenced to the VBCLK signal are satisfied.

6.4 Software STOP Mode

In software STOP mode, the CPU operation clock and the clock generator are stopped. The overall system is stopped, and ultra-low power consumption is achieved in which only leak current is lost.

(1) Setting and operation status

The NU85E 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 register.

Although program execution stops in software STOP mode, the contents of all registers and of RAM immediately before software STOP mode began are maintained. The operation of all NU85E-internal peripheral I/O is also stopped.

(2) Cancellation of software STOP mode

Software STOP mode is canceled by a non-maskable interrupt request, an unmasked maskable interrupt request, or the input of a DCRESZ signal.

(a) Cancellation by interrupt request

Software STOP mode is canceled by a non-maskable interrupt request not masked by the PSC register or by an unmasked maskable interrupt request regardless of the priority. The following table shows the operation performed after software STOP mode is canceled.

Caution An interrupt request that occurs while the NMI2M to NMI0M and INTM bits of the power save control register (PSC) are set (interrupt disabled), is invalid (software STOP mode is not canceled).

Table 6-2. Operation After Software STOP Mode Is Canceled by Interrupt Request

Cancellation Source	Interrupt Enabled (EI) State	Interrupt Disabled (DI) State
Non-maskable interrupt request	Branch to handler address	
Maskable interrupt request	Branch to handler address or execution of next instruction	Execution of next instruction

The operation shown in Table 6-3 is performed if software STOP mode was set within the interrupt servicing routine.

Table 6-3. Operation After Setting Software STOP Mode in Interrupt Servicing Routine

Interrupt Servicing Routine Type When Software STOP Mode Is Set	Cancellation Source		Operation
		Priority ^{Note 1}	
Maskable interrupt	Maskable interrupt request	Low	Software STOP mode is canceled and the interrupt request is not acknowledged (pending).
		Same	
		High (ID = 1) ^{Note 2}	
	Non-maskable interrupt request	High (ID = 0) ^{Note 3}	Software STOP mode is canceled and the interrupt request is acknowledged.
Non-maskable interrupt	Maskable interrupt request	–	Software STOP mode is canceled and the interrupt request is not acknowledged (pending).
	Non-maskable interrupt request	Low	
		Same	
		High	Software STOP mode is canceled and the interrupt request is acknowledged.

Notes 1. The priority order of the interrupts when software STOP mode is set (interrupts that were under servicing).

2. When the ID bit of the PSW is 1 (interrupt acknowledgement disabled)

3. When the ID bit of the PSW is 0 (interrupt acknowledgement enabled)

★ **Remark** Cancellation of software STOP mode by NMI is performed regardless of the NP bit value in the PSW.

(b) Cancellation by DCRESZ signal input

This is the same as a normal reset operation.

Caution Be sure to input the DCRESZ signal so that the setup and hold times referenced to the VBCLK signal are satisfied.

6.5 Hardware STOP Mode

In hardware STOP mode, the CPU operation clock and the clock generator are stopped. The overall system is stopped, and ultra-low power consumption is achieved in which only leak current is lost.

(1) Setting and operation status

The NU85E is switched to hardware STOP mode by inputting a low-level signal to the DCSTOPZ pin. The NU85E is switched to hardware STOP mode even if a low-level signal is input to the DCSTOPZ pin when the NU85E is in HALT mode or software STOP mode.

Although program execution stops in hardware STOP mode, the contents of all registers and of RAM immediately before hardware STOP mode began are maintained. The operation of all NU85E-internal peripheral I/O is also stopped.

Remark The NU85E may not switch to hardware STOP mode correctly if the DCSTOPZ input becomes active (low level) due to a read modify write, misalign access, etc. while the VMLOCK signal is locked. If the DCSTOPZ input becomes low level in the bus lock state, an internal CPU of the NU85E is stopped, but the HWSTOPRQ signal, which controls the external clock generator, does not become active because the slave device connected to the locked bus may require clock supply. Consequently, clock is not stopped and the NU85E will not switch to hardware STOP mode.

If the system must be switched to hardware STOP mode when the DCSTOPZ input is low level, mask the DCSTOPZ input by the VMLOCK signal to avoid switching to hardware STOP mode while the bus is locked.

(2) Cancellation of hardware STOP mode

Hardware STOP mode is canceled by inputting the DCSTOPZ or DCRESZ signal.

(a) Cancellation by DCSTOPZ signal input

Hardware STOP mode is canceled when the input to the DCSTOPZ pin goes from low level to high level.

The mode to which the NU85E switches after hardware STOP mode is canceled differs as follows according to the status in effect before hardware STOP mode was set.

Table 6-4. Status After Cancellation of Hardware STOP Mode

Before Hardware STOP Mode Was Set	After Hardware STOP Mode Was Canceled
Normal operation mode	Normal operation mode
Software STOP mode	Normal operation mode
HALT mode	HALT mode

(b) Cancellation by DCRESZ signal input

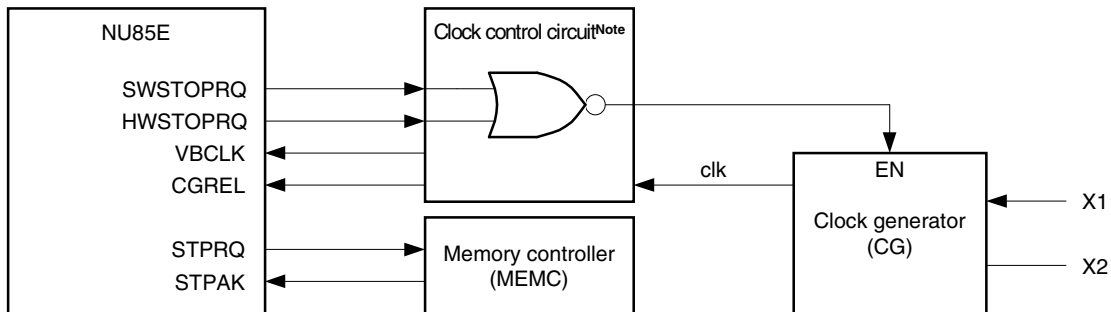
This is the same as a normal reset operation.

Caution Be sure to input the DCRESZ signal so that the setup and hold times referenced to the VBCLK signal are satisfied.

6.6 Clock Control in Software/Hardware STOP Mode

The NU85E and clock control circuit are connected as follows.

Figure 6-4. Connection of NU85E and Clock Control Circuit



Note Design the clock control circuit as a user logic. Also, include a circuit for ensuring the oscillation stabilization time (see **Figures 6-5** and **6-6**).

Caution In a system in which the MEMC is not connected to the NU85E, handle the STPAK pin in either of the following ways.

- Always input a high level.
- Connect user logic that outputs a high level to the STPAK pin to the STPRQ output (high level) of the NU85E.

If a high level is not input to the STPAK pin, the HWSTOPRQ and SWSTOPRQ signals do not become active and shifting the STOP mode becomes impossible.

(1) Clock control when setting or canceling software STOP mode**(a) When setting software STOP mode (after software STOP mode is set by setting the STP bit of the PSC register)**

- <1> Set the STOP mode request signal (STPRQ) to active (high level) and output it to the memory controller.
- <2> Input the active level (high level) of the acknowledge signal (STPAK) from the memory controller that received the STPRQ signal.
- <3> Set the software STOP mode request signal (SWSTOPRQ) to active (high level) and output it to the clock control circuit (Use this SWSTOPRQ signal to stop the VBCLK output from the clock control circuit).

(b) When canceling software STOP mode

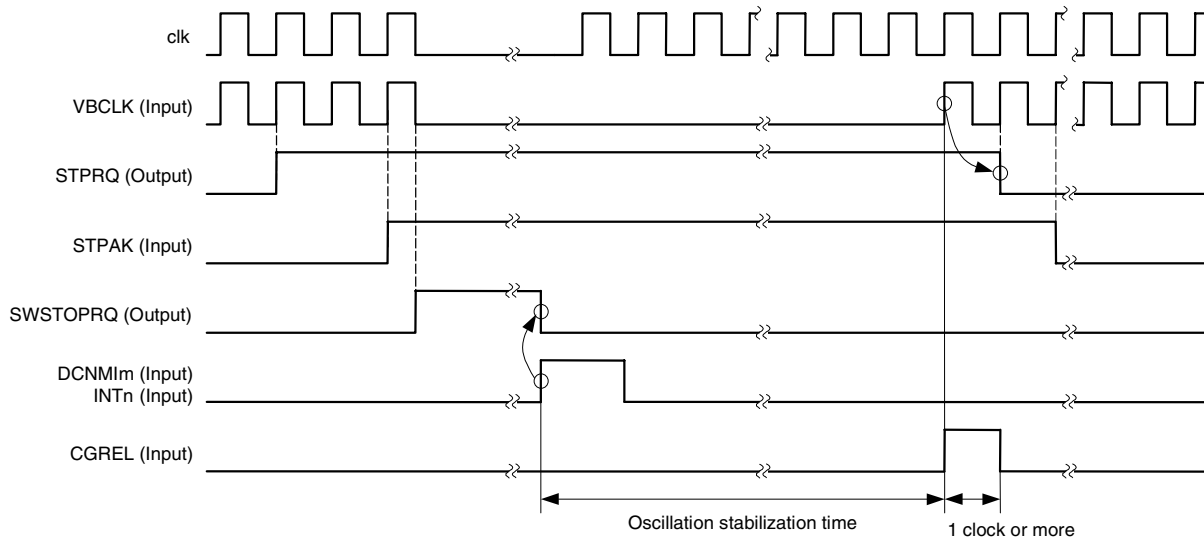
- <1> Input a non-maskable interrupt request (NMI_m), unmasked maskable interrupt request (INT_n), or the DCRESZ signal (m = 2 to 0, n = 63 to 0).
- <2> Set the software STOP mode request signal (SWSTOPRQ) to inactive (low level) and output it to the clock control circuit (clock generator starts operation).
- <3> After the oscillation stabilization time, input the active level (high level) of the CGREL signal from the clock control circuit simultaneous with the VBCLK signal (The input of the VBCLK signal returns the STPRQ and STPAK outputs to low level).
- ★ <4> After inputting the VBCLK signal, input a high level to the DCRESZ signal.

Caution Input an active level (high level) to the CGREL pin for one clock or more.
When setting the software STOP mode again, be sure to input an inactive level (low level) to the CGREL pin before setting.

★ **Remark** A level latch is used for the DCRESZ signal, which can therefore be input asynchronously to VBCLK.

Figure 6-5. Software STOP Mode Set/Cancel Timing Example

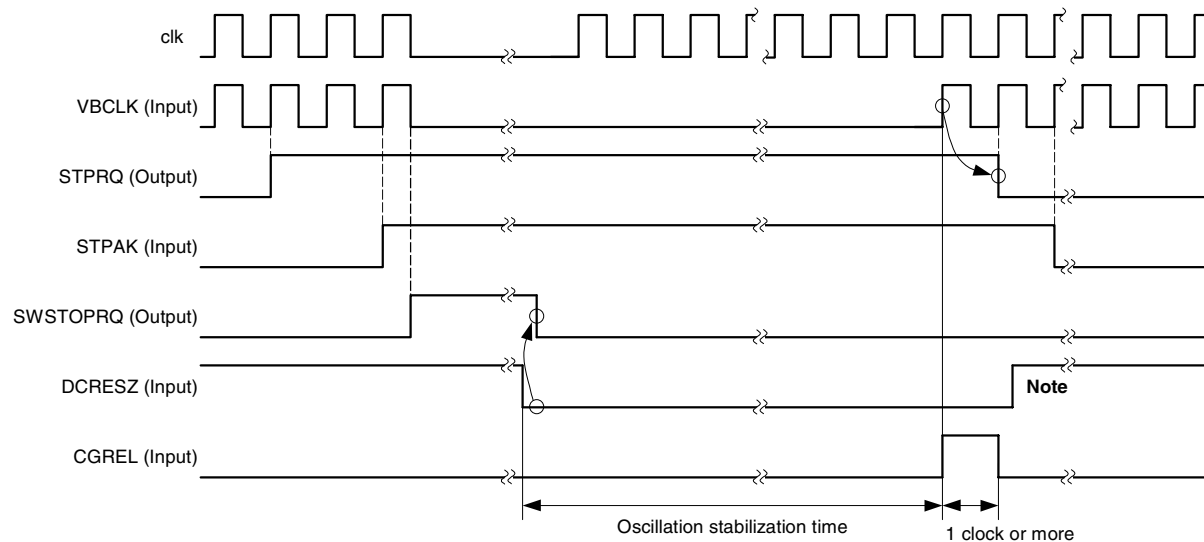
(a) When software STOP mode is canceled by DCNMIm or INTn input



Remarks 1. The DCNMIm and INTn inputs are detected at the rising edge and the interrupt request is held in the CPU.

2. m = 2 to 0, n = 63 to 0

(b) When software STOP mode is canceled by DCRESZ input



Note Input a high level to the DCRESZ pin after restarting input of VBCLK.

(2) Clock control when setting or canceling hardware STOP mode**(a) When setting hardware STOP mode**

- <1> Input the active level (low level) of the DCSTOPZ signal.
- <2> Set the STOP mode request signal (STPRQ) to active (high level) and output it to the memory controller.
- <3> Input the active level (high level) of the acknowledge signal (STPAK) from the memory controller that received the STPRQ signal.
- <4> Set the hardware STOP mode request signal (HWSTOPRQ) to active (high level) and output it to the clock control circuit (Use this HWSTOPRQ signal to stop the VBCLK output from the clock control circuit).

(b) When canceling hardware STOP mode

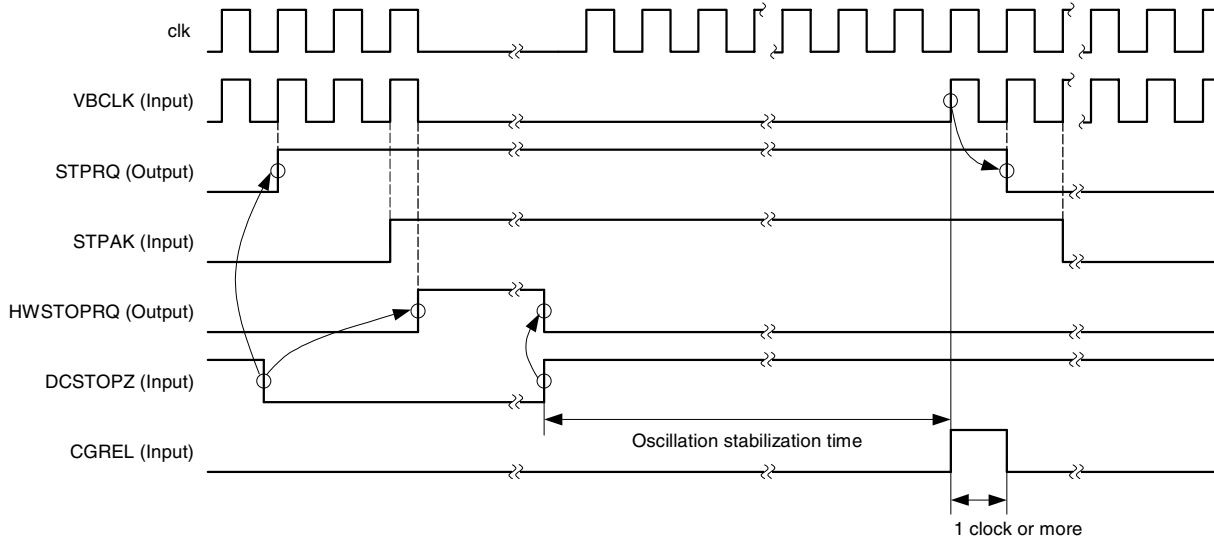
- <1> Input the DCRESZ signal or the inactive level (high level) of the DCSTOPZ signal.
- <2> Set the hardware STOP mode request signal (HWSTOPRQ) to inactive (low level) and output it to the clock control circuit (clock generator starts operation).
- <3> After the oscillation stabilization time, input the active level (high level) of the CGREL signal from the clock control circuits simultaneous with the VBCLK signal (The input of the VBCLK signal returns the STPRQ and STPAK outputs to low level).

Caution Input an active level (high level) to the CGREL pin for one clock or more.
When setting the hardware STOP mode again, be sure to input an inactive level (low level) to the CGREL pin before setting.

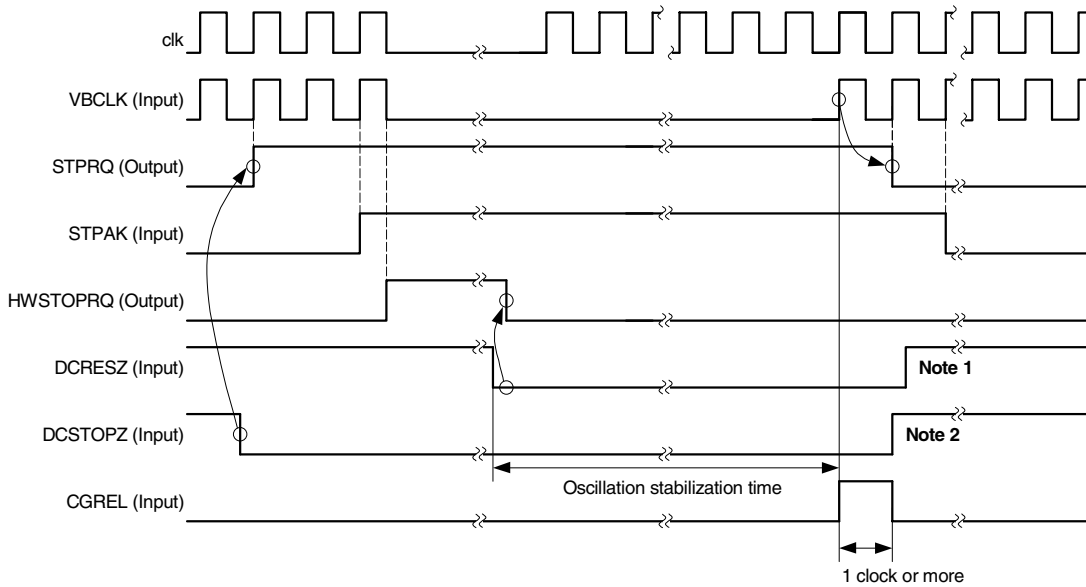
★ **Remark** A level latch is used for the DCRESZ signal, which can therefore be input asynchronously to VBCLK.

Figure 6-6. Hardware STOP Mode Set/Cancel Timing Example

(a) When hardware STOP mode is canceled by DCSTOPZ input



(b) When hardware STOP mode is canceled by DCRESZ input



- Notes**
1. Input a high level to the DCRESZ pin after restarting input of VBCLK.
 2. Input a high level to the DCSTOPZ pin before inputting a high level to the DCRESZ pin.

CHAPTER 7 DMAC

The DMA control unit (DMAC) controls data transfers between memory and peripheral macros or between memory and memory based on DMA transfer requests issued according to the DMARQ3 to DMARQ0 pins or software triggers (memory means RAM or external memory).

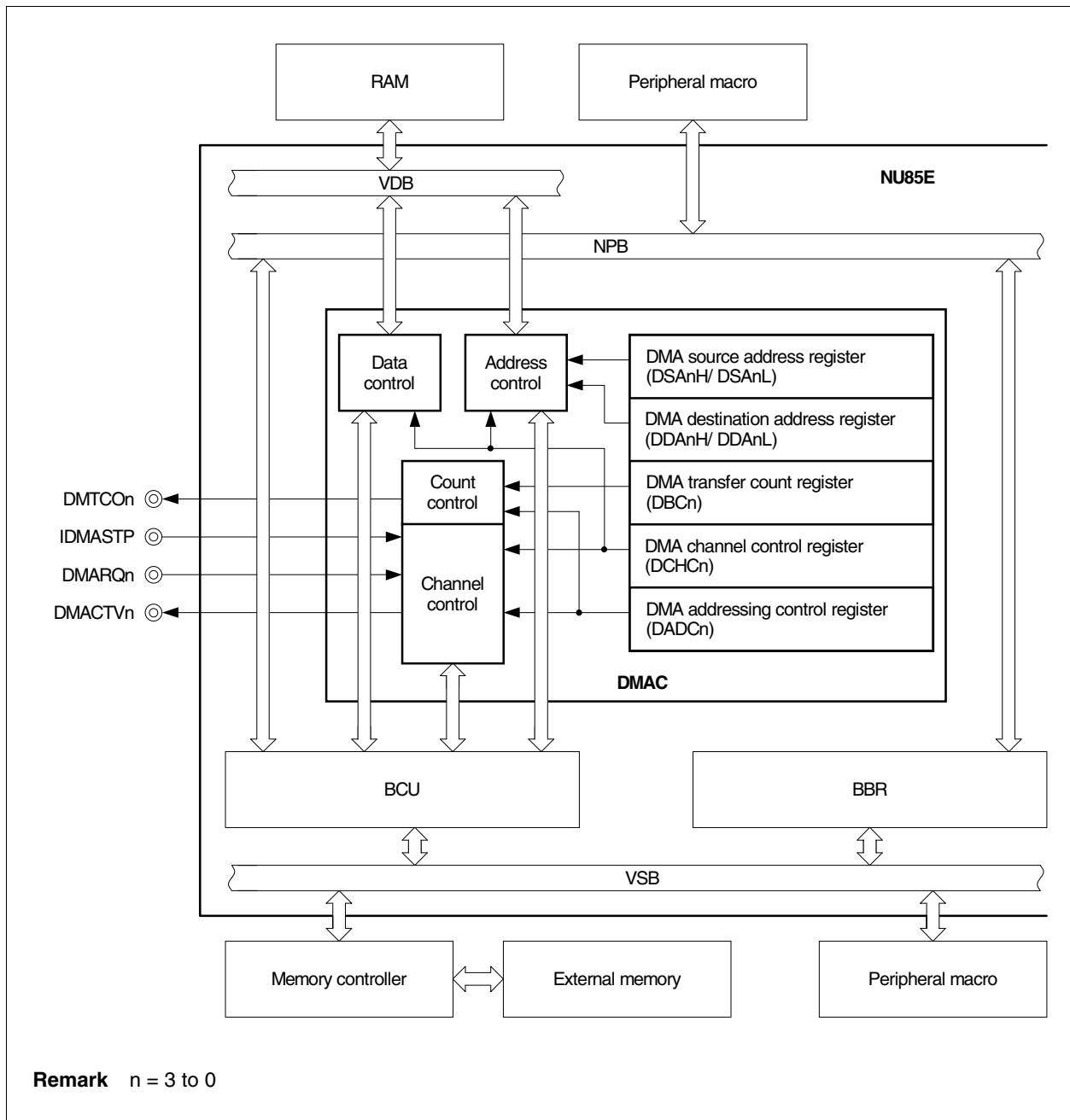
7.1 Features

- Four independent DMA channels
- Transfer units: 8 bits, 16 bits, or 32 bits
- Maximum transfer count: 65,536 (2^{16})
- Two transfer types
 - Flyby (1-cycle) transfer
 - Two-cycle transfer
- Four transfer modes
 - Single transfer mode
 - Single-step transfer mode
 - Line transfer mode (four bus cycle transfer mode)
(in 2-cycle transfer, the operation from read to write is repeated four times)
 - Block transfer mode
- Transfer requests
 - Requests by DMARQ3 to DMARQ0 pin input
 - Requests by software
- Transfer objects
 - Between RAM^{Note} and peripheral macros
 - Between RAM^{Note} and external memory
 - Between RAM^{Note} and RAM^{Note}
 - Between external memory and peripheral macros
 - Between external memory and external memory (Transfer between little endian area and big endian area is possible)

Note RAM directly connected to the VDB (refer to **7.2 Configuration**)

- Terminal count output signals (DMTCO3 to DMTCO0)
- Next address setting function

7.2 Configuration



7.3 Transfer Objects

(1) Transfer types

Table 7-1 shows the relationships between transfer types and transfer objects.

Caution Operation is not guaranteed when a transfer is performed using a combination of transfer source and transfer destination marked by an “No” in Table 7-1.

Table 7-1. Relationships Between Transfer Type and Transfer Object

		Transfer Destination					
		Two-Cycle Transfer			Flyby Transfer		
		VSB	NPB	RAM	VSB	NPB	RAM
Transfer Source	VSB	Yes	Yes	Yes	Yes ^{Note}	No	No
	NPB	Yes	Yes	Yes	No	No	No
	RAM	Yes	Yes	Yes	No	No	No

Note The transfer can be performed only when using the MEMC (NT85E500) associated with the flyby transfer.

Remark Yes: Transfer enabled
 No: Transfer disabled
 VSB: External memory or peripheral macro on the VSB
 NPB: Peripheral macro on the NPB
 RAM: RAM directly connected to the VDB

(2) Wait function

Table 7-2 shows the relationships between the wait function and transfer objects.

Table 7-2. Relationships Between Wait Function and Transfer Object

Transfer Object	Wait Function
VSB	Set by MEMC (NT85E500, NT85E502)
NPB	Set by VSWC register
RAM	No wait

7.4 DMA Channel Priorities

DMA channel prioritization is fixed as follows.

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

This prioritization is only valid in the TI state. During a block transfer, the channel used for transfer is never switched.

During a single-step transfer, if a higher priority DMA transfer request is generated during the period when the bus is released (TI), the higher priority DMA transfer is performed.

7.5 Control Registers

7.5.1 DMA source address registers 0 to 3 (DSA0 to DSA3)

These registers are used to set the DMA transfer source addresses (28 bits each) for DMA channels n (n = 0 to 3). They are divided into two 16-bit registers, DSA_nH and DSA_nL, respectively.

Since they are two-stage FIFO-configuration buffer registers, the transfer source address of a new DMA transfer can be set during a DMA transfer (See 7.6 Next Address Setting Function).

When a flyby transfer is set according to the TTYP bit of the DMA addressing control registers n (DADC_n), the external memory addresses are set by the DSA_n registers. At this time, any settings of the DMA destination address registers n (DDA_n) are ignored.

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

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

Figure 7-1. DMA Source Address Registers 0H to 3H (DSA0H to DSA3H)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DSA0H	IR	0	0	0	SA 27	SA 26	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	Address FFFFFF082H	After reset Undefined
DSA1H	IR	0	0	0	SA 27	SA 26	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	Address FFFFFF08AH	After reset Undefined
DSA2H	IR	0	0	0	SA 27	SA 26	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	Address FFFFFF092H	After reset Undefined
DSA3H	IR	0	0	0	SA 27	SA 26	SA 25	SA 24	SA 23	SA 22	SA 21	SA 20	SA 19	SA 18	SA 17	SA 16	Address FFFFFF09AH	After reset Undefined

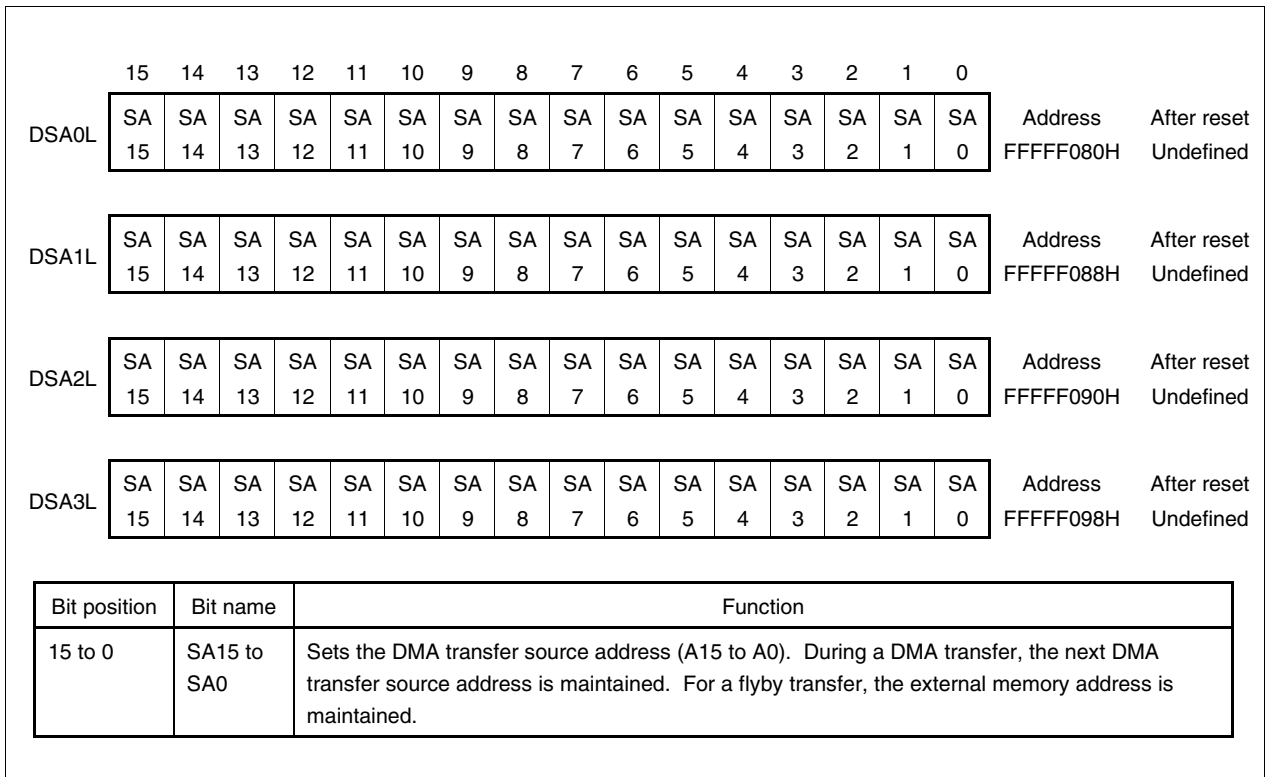
Bit position	Bit name	Function
15	IR	Specifies the DMA transfer source. 0: External memory or peripheral macro 1: RAM
11 to 0	SA27 to SA16	Sets the DMA transfer source address (A27 to A16). During a DMA transfer, the next DMA transfer source address is maintained. For a flyby transfer, the external memory address is maintained.

Caution Bits 14 to 12 of the DSA0H to DSA3H registers must be set to 0. The operation when these bits are set to 1 is not guaranteed.

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

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

Figure 7-2. DMA Source Address Registers 0L to 3L (DSA0L to DSA3L)



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

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

Since they are two-stage FIFO-configuration buffer registers, the transfer destination address of a new DMA transfer can be set during a DMA transfer (See 7.6 Next Address Setting Function).

When a flyby transfer is set according to the TTYP bit of the DMA addressing control registers n (DADCn), any setting of these registers are ignored.

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

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

Figure 7-3. DMA Destination Address Registers 0H to 3H (DDA0H to DDA3H)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
DDA0H	IR	0	0	0	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	Address	After reset
					27	26	25	24	23	22	21	20	19	18	17	16	FFFFF086H	Undefined
DDA1H	IR	0	0	0	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	Address	After reset
					27	26	25	24	23	22	21	20	19	18	17	16	FFFFF08EH	Undefined
DDA2H	IR	0	0	0	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	Address	After reset
					27	26	25	24	23	22	21	20	19	18	17	16	FFFFF096H	Undefined
DDA3H	IR	0	0	0	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	Address	After reset
					27	26	25	24	23	22	21	20	19	18	17	16	FFFFF09EH	Undefined

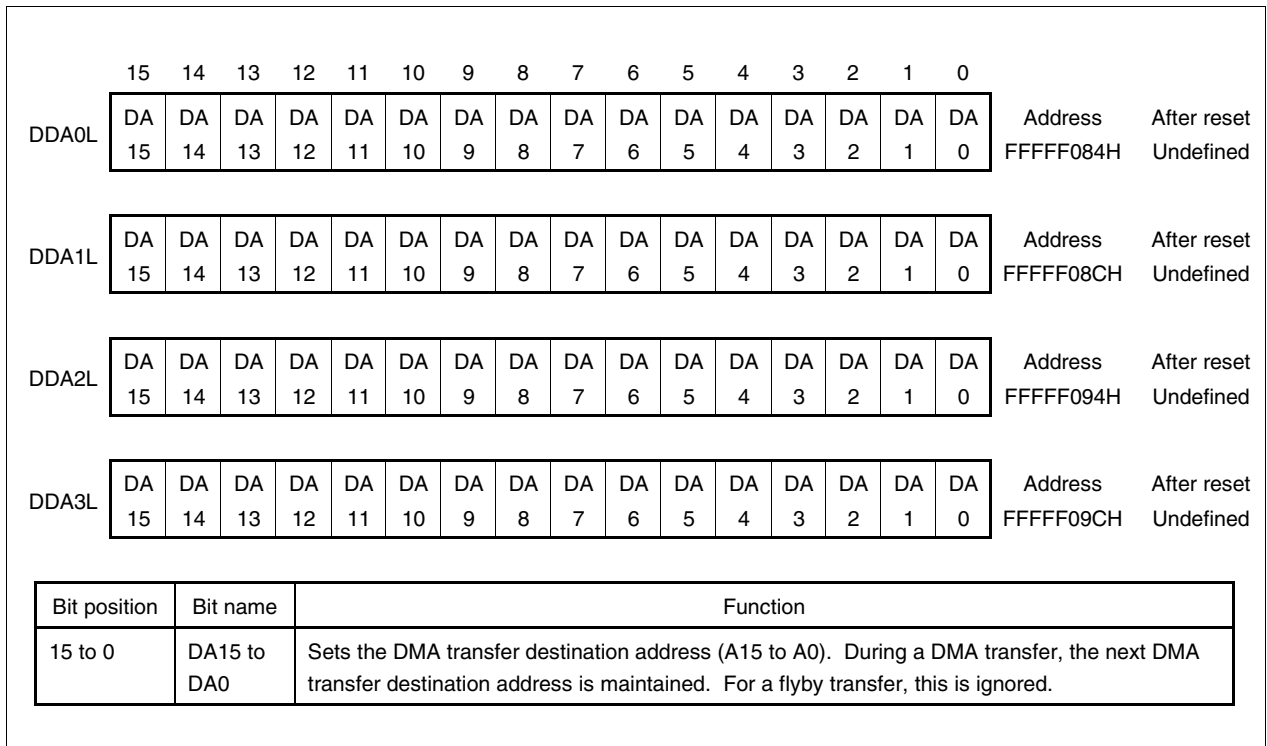
Bit position	Bit name	Function
15	IR	Specifies the DMA transfer destination. 0: External memory or peripheral macro 1: RAM
11 to 0	DA27 to DA16	Sets the DMA transfer destination address (A27 to A16). During a DMA transfer, the next DMA transfer destination address is maintained. For a flyby transfer, this is ignored.

Caution Bits 14 to 12 of the DDA0H to DDA3H registers must be set to 0. The operation when these bits are set to 1 is not guaranteed.

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

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

Figure 7-4. DMA Destination Address Registers 0L to 3L (DDA0L to DDA3L)



7.5.3 DMA transfer count registers 0 to 3 (DBC0 to DBC3)

These 16-bit registers are used to set the transfer counts for DMA channels n (n = 0 to 3). These registers maintain the remaining transfer count during a DMA transfer.

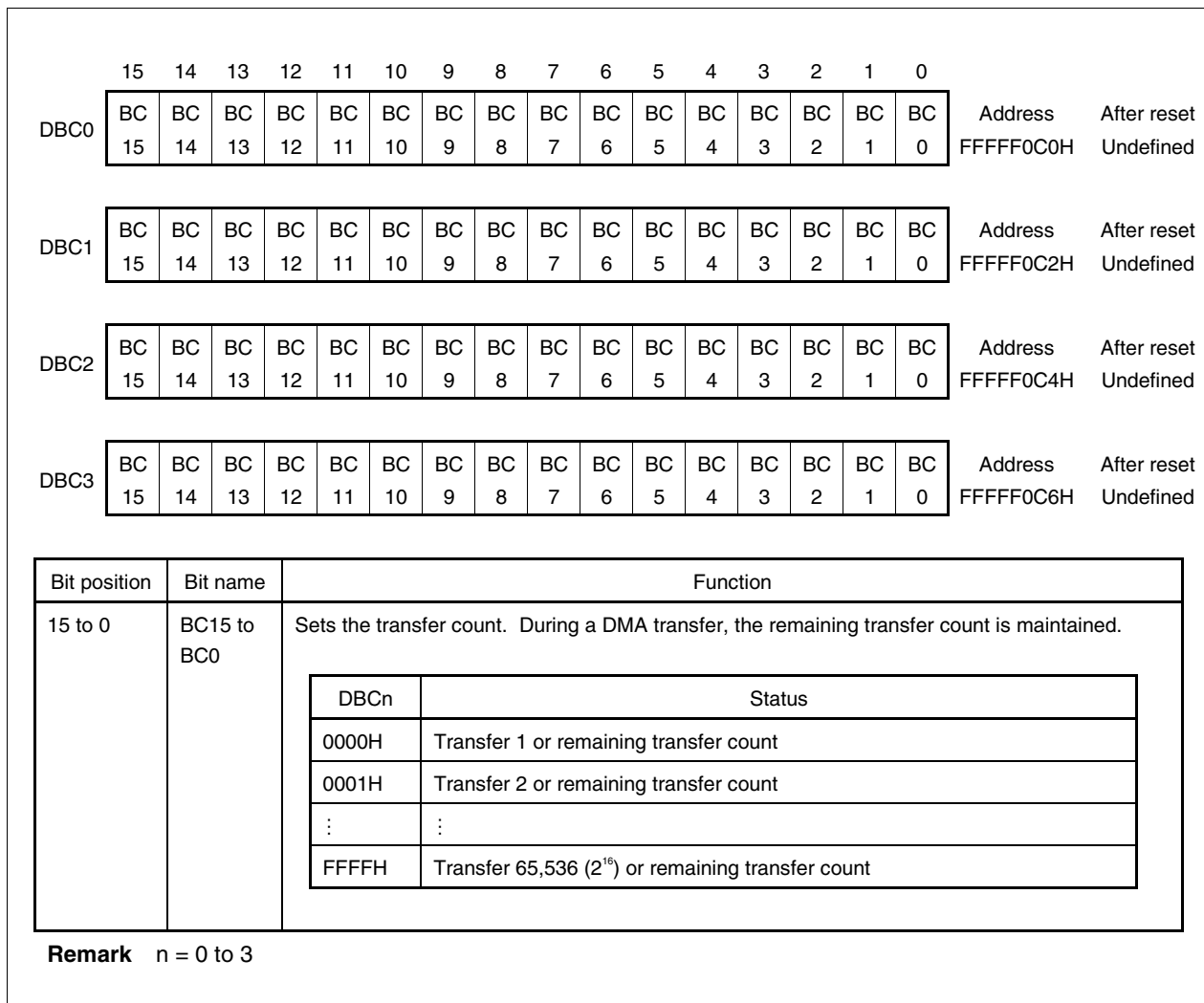
Since they are two-stage FIFO-configuration buffer registers, the transfer count of a new DMA transfer can be set during a DMA transfer (See 7.6 Next Address Setting Function).

These registers are decremented by 1 for each transfer that is performed. Transfer ends when a borrow occurs.

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

Note that in the case of line transfers, when the DBCn register is 0003H (4 transfers) this becomes one line transfer. For a setting in which the transfer count cannot be divided by 4, the sections that can be line transferred are (line) transferred first, then the remaining indivisible section is transferred as single transfers.

Figure 7-5. DMA Transfer Count Registers 0 to 3 (DBC0 to DBC3)



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

These 16-bit registers are used to control the DMA transfer operation mode for DMA channels n (n = 0 to 3). These registers can be read or written in 16-bit units.

Caution These registers cannot be accessed during a DMA transfer.

Figure 7-6. DMA Addressing Control Registers 0 to 3 (DADC0 to DADC3) (1/2)

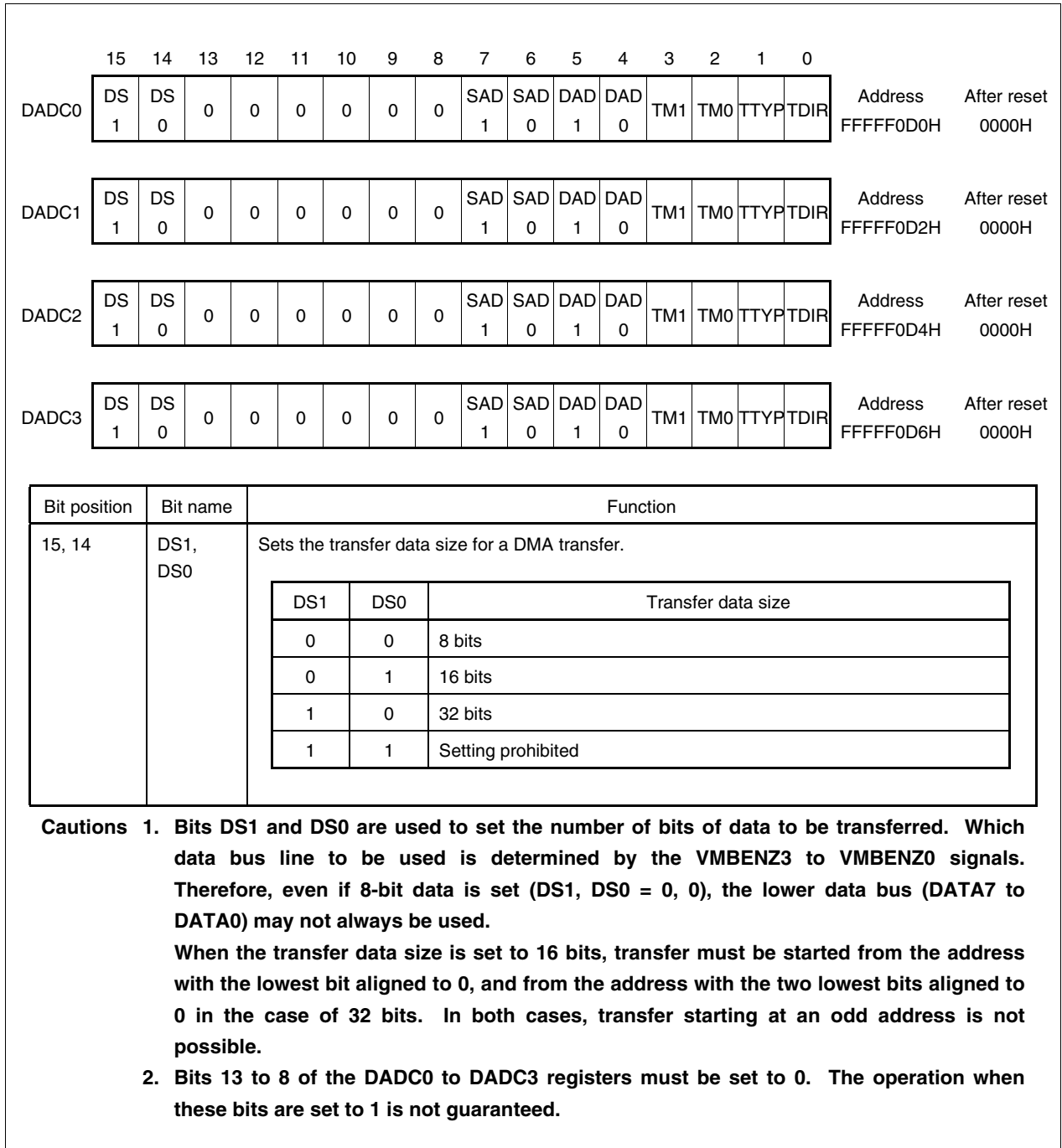


Figure 7-6. DMA Addressing Control Registers 0 to 3 (DADC0 to DADC3) (2/2)

Bit position	Bit name	Function															
7, 6	SAD1, SAD0	<p>Sets the count direction of the transfer source addresses for DMA channels n (n = 0 to 3).</p> <table border="1"> <thead> <tr> <th>SAD1</th> <th>SAD0</th> <th>Count direction</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Increment</td> </tr> <tr> <td>0</td> <td>1</td> <td>Decrement</td> </tr> <tr> <td>1</td> <td>0</td> <td>Fixed</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	SAD1	SAD0	Count direction	0	0	Increment	0	1	Decrement	1	0	Fixed	1	1	Setting prohibited
SAD1	SAD0	Count direction															
0	0	Increment															
0	1	Decrement															
1	0	Fixed															
1	1	Setting prohibited															
5, 4	DAD1, DAD0	<p>Sets the count direction of the transfer destination addresses for DMA channels n (n = 0 to 3).</p> <table border="1"> <thead> <tr> <th>DAD1</th> <th>DAD0</th> <th>Count direction</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Increment</td> </tr> <tr> <td>0</td> <td>1</td> <td>Decrement</td> </tr> <tr> <td>1</td> <td>0</td> <td>Fixed</td> </tr> <tr> <td>1</td> <td>1</td> <td>Setting prohibited</td> </tr> </tbody> </table>	DAD1	DAD0	Count direction	0	0	Increment	0	1	Decrement	1	0	Fixed	1	1	Setting prohibited
DAD1	DAD0	Count direction															
0	0	Increment															
0	1	Decrement															
1	0	Fixed															
1	1	Setting prohibited															
3, 2	TM1, TMO	<p>Sets the transfer mode used for DMA transfers.</p> <table border="1"> <thead> <tr> <th>TM1</th> <th>TMO</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>Line transfer mode</td> </tr> <tr> <td>1</td> <td>1</td> <td>Block transfer mode</td> </tr> </tbody> </table>	TM1	TMO	Transfer mode	0	0	Single transfer mode	0	1	Single-step transfer mode	1	0	Line transfer mode	1	1	Block transfer mode
TM1	TMO	Transfer mode															
0	0	Single transfer mode															
0	1	Single-step transfer mode															
1	0	Line transfer mode															
1	1	Block transfer mode															
1	TTYP	<p>Sets the DMA transfer type.</p> <p>0: Two-cycle transfer 1: Flyby transfer^{Note}</p>															
0	TDIR	<p>Sets the transfer direction used for transfers between peripheral macros and external memory. The setting is valid only for flyby transfers and is ignored for 2-cycle transfers.</p> <p>0: External memory to peripheral macro (read) 1: Peripheral macro to external memory (write)</p>															

Remark n = 0 to 3

Note Valid only when using the MEMC associated with the flyby transfer.

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

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

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

Figure 7-7. DMA Channel Control Registers 0 to 3 (DCHC0 to DCHC3) (1/2)

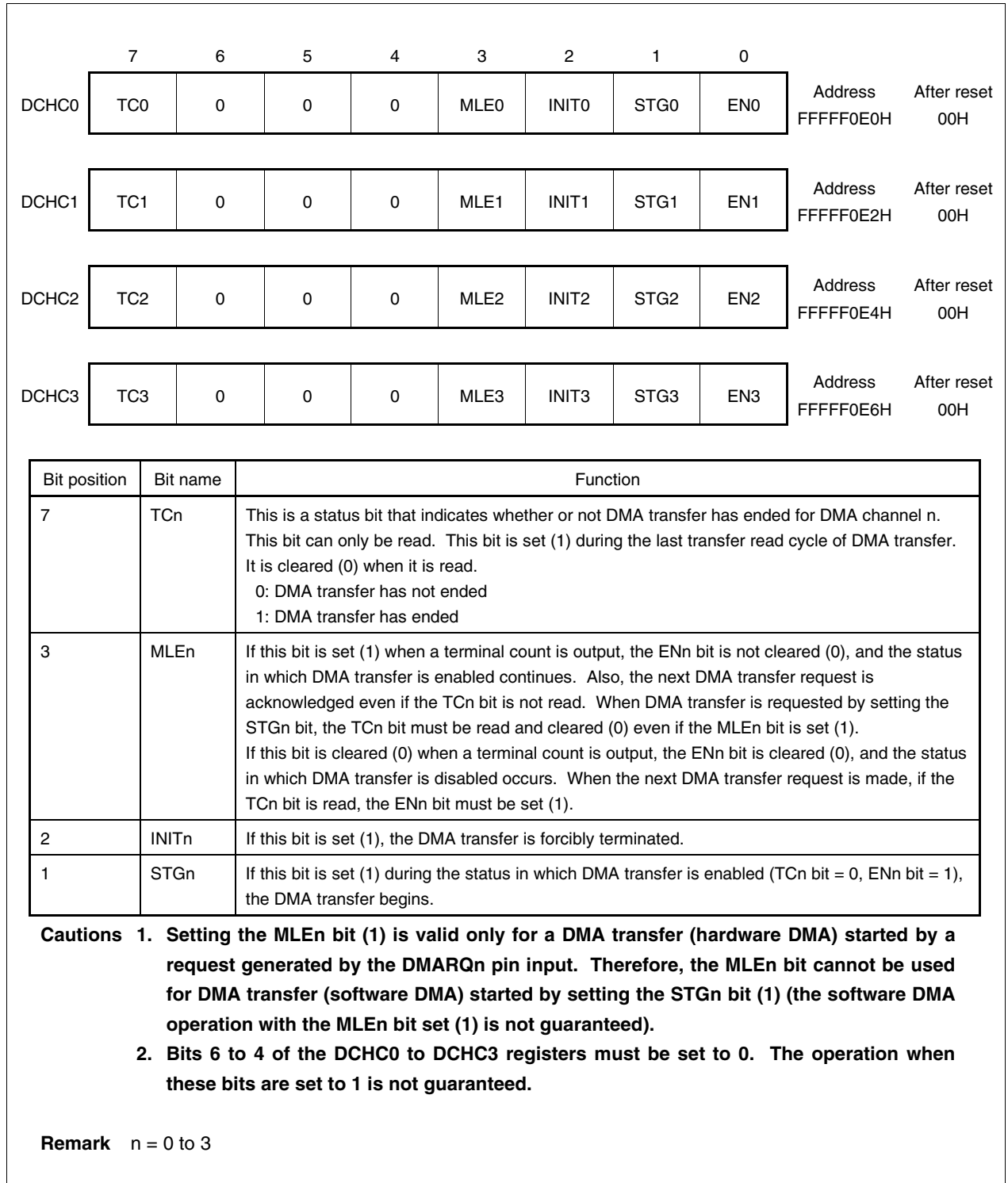


Figure 7-7. DMA Channel Control Registers 0 to 3 (DCHC0 to DCHC3) (2/2)

Bit position	Bit name	Function
0	ENn	Sets whether DMA transfer is enabled or disabled for DMA channel n. This bit is cleared (0) when the DMA transfer is completed. It is also cleared (0) when an IDMASTP signal is input or when transfer is forcibly terminated by setting (1) the INITn bit. 0: DMA transfer is disabled 1: DMA transfer is enabled

Remark n = 0 to 3

7.5.6 DMA disable status register (DDIS)

This register maintains the contents of the ENn bit of the DCHCn register when an IDMASTP signal is input (n = 0 to 3).

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

Figure 7-8. DMA Disable Status Register (DDIS)

	7	6	5	4	3	2	1	0		
DDIS	0	0	0	0	CH3	CH2	CH1	CH0	Address	After reset
									FFFFFF0FH	00H

Bit position	Bit name	Function
3 to 0	CH3 to CH0	Reflects the contents of the ENn bit of the DCHCn register when an IDMASTP signal is input. The contents of this register are maintained until the next IDMASTP signal is input or a system reset occurs.

Caution Bits 7 to 4 of the DDIS register must be set to 0. The operation when these bits are set to 1 is not guaranteed.

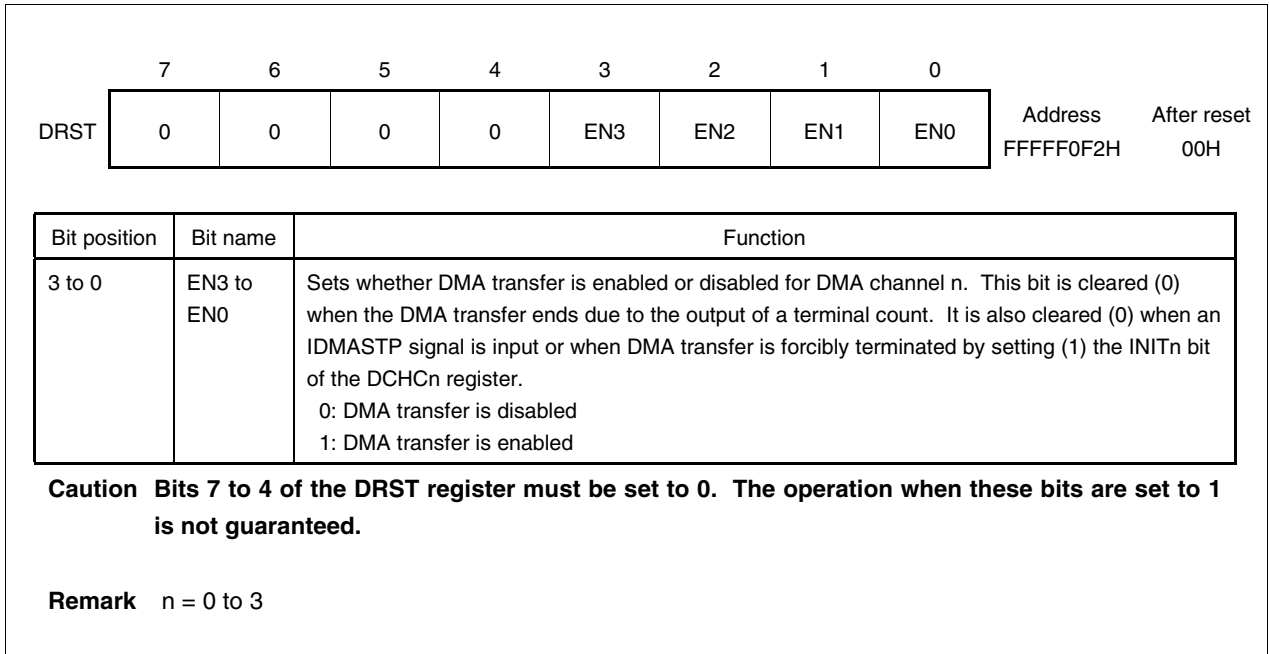
Remark n = 0 to 3

7.5.7 DMA restart register (DRST)

This register is used to restart a DMA transfer that was forcibly interrupted by inputting an IDMASTP signal. The ENn bits of this register are linked respectively with the ENn bits of the DCHCn registers (n = 0 to 3). After a DMA transfer was forcibly interrupted by inputting the IDMASTP signal, the DMA channel for which the transfer was interrupted is confirmed from the contents of the DDIS register, and the DMA transfer can be restarted by setting (1) the ENn bit of the corresponding DMA channel.

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

Figure 7-9. DMA Restart Register (DRST)



7.6 Next Address Setting Function

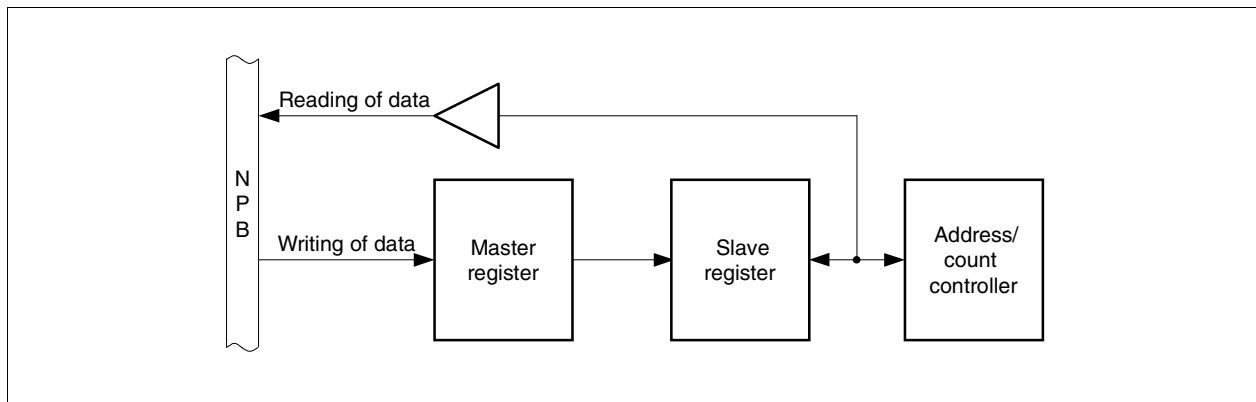
The DMA source address registers (DSAnH and DSAnL), DMA destination address registers (DDAnH and DDAnL), and DMA transfer count registers (DBCn) are two-stage FIFO-configuration buffer registers ($n = 0$ to 3).

When a terminal count signal (DMTCON) is output, these registers are automatically rewritten with the values that had just been set before the signal is output.

Therefore, if a new DMA transfer is set for these registers during a DMA transfer, the transfer can begin only when the ENn bit of the DCHCn register is set (1).

Figure 7-10 shows the buffer register configuration.

Figure 7-10. Buffer Register Configuration



7.7 DMA Bus State

7.7.1 Bus state types

DMAC bus cycles consist of the 13 states shown below.

(1) T1 state

This is an idle state in which there is no access request.

The DMARQ3 to DMARQ0 signals are sampled at the rising edge of the VBCLK signal.

(2) T0 state

This is a DMA transfer ready state (There is a DMA transfer request, and the bus access right has been acquired for the first DMA transfer).

(3) T1R state

This is the state to which the DMAC moves first for a 2-cycle transfer read.

Address driving begins. After the T1R state, the DMAC always transitions to the T2R state.

(4) T1RI state

This is a state in which the DMAC is awaiting an acknowledge signal for an external memory read request.

After the last T1RI state, the DMAC always transitions to the T2R state.

(5) T2R state

This is a wait state or the last state of a 2-cycle transfer read.

In the last T2R state, read data is sampled. After the read data is sampled, the DMAC always transitions to the T1W state.

(6) T2RI state

This is a DMA transfer ready state for a DMA transfer to RAM (The bus access right has been acquired for a DMA transfer to RAM).

After the last T2RI state, the DMAC always transitions to the T1W state.

(7) T1W state

This is the state to which the DMAC moves first for a 2-cycle transfer write.

Address driving begins. After the T1W state, the DMAC always transitions to the T2W state.

(8) T1WI state

This is a state in which the DMAC is awaiting an acknowledge signal for an external memory write request.

After the last T1WI state, the DMAC always transitions to the T2W state.

(9) T2W state

This is a wait state or the last state of a 2-cycle transfer write.

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

(10) T1FH state

This is the basic state of a flyby transfer and is the execution cycle of that transfer.

After the T1FH state, the DMAC transitions to the T2FH state.

(11) T1FHI state

This is the last state of a flyby transfer, and the DMAC is awaiting the end of the transfer. After the T1FHI state, the bus is released, and the DMAC transitions to the TE state.

(12) T2FH state

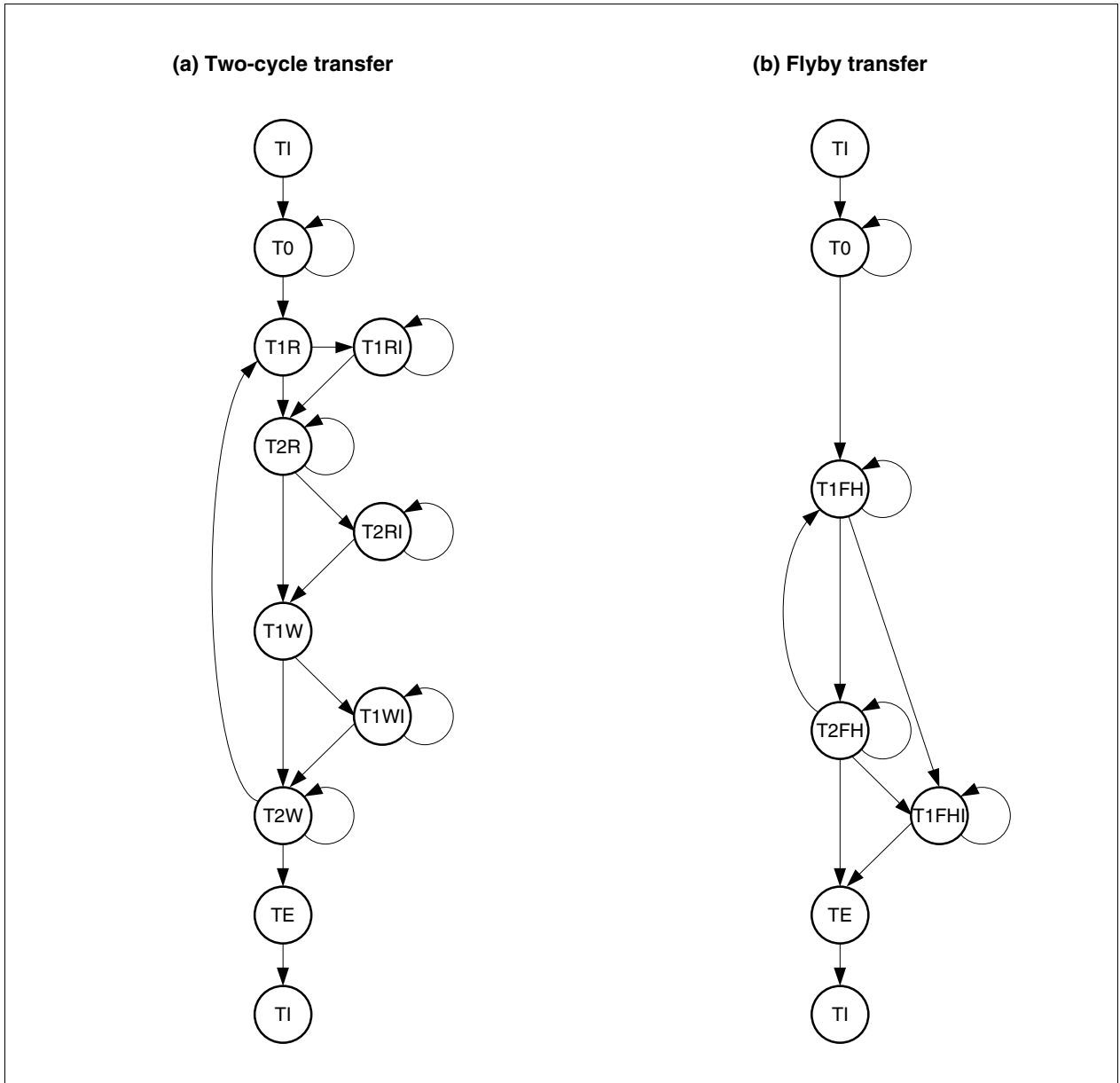
This is the state in which the DMAC judges whether or not to continue flyby transfers. If the next transfer is executed in block transfer mode, the DMAC moves to the T1FH state after the T2FH state. In other modes, if a wait has occurred, the DMAC transitions to the T1FHI state. If no wait has occurred, the bus is released, and the DMAC transitions to the TE state.

(13) TE state

This is the state in which the DMA transfer is completed. The DMAC generates a terminal count signal (DMTCON) and initializes other types of internal signals ($n = 3$ to 0). After the TE state, the DMAC always transitions to the TI state.

7.7.2 DMAC bus cycle state transitions

Figure 7-11. DMAC Bus Cycle State Transition Diagram



7.8 Transfer Modes

7.8.1 Single transfer mode

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

If a higher priority DMA transfer request is generated while the DMAC has released the bus, the higher priority DMA transfer request always takes precedence. However, if a lower priority DMA transfer request is generated within one clock after the end of a single transfer, even if the previous higher priority DMA transfer request signal stays active, this request is not prioritized, and the next DMA transfer after the bus is released for the CPU is a transfer based on the newly generated, lower priority DMA transfer request.

Figures 7-12 to 7-15 show examples for single transfer.

Figure 7-12. Single Transfer Example 1

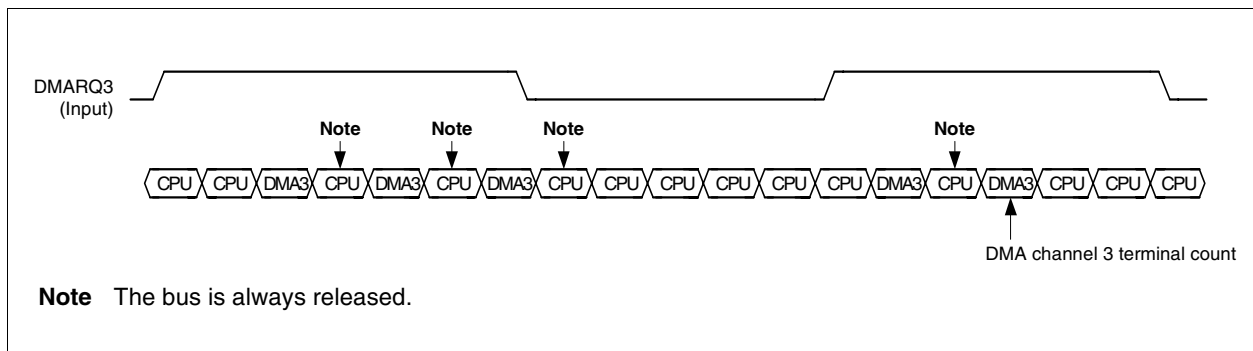


Figure 7-13 shows a single transfer mode example in which a higher priority DMA transfer request is generated. DMA channels 0 to 2 are used for a block transfer, and channel 3 is used for a single transfer.

Figure 7-13. Single Transfer Example 2

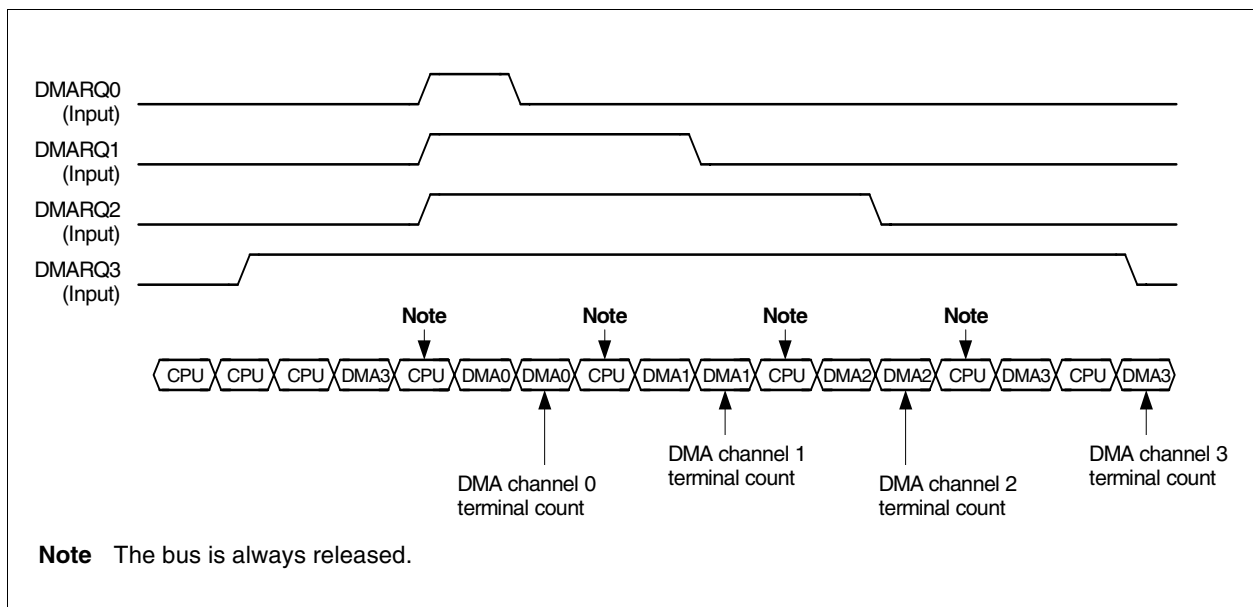


Figure 7-14 shows a single transfer mode example in which a lower priority DMA transfer request is generated within one clock after the end of a single transfer. DMA channels 0, 3 are used for a single transfer. When two DMA transfer request signals are activated at the same time, the two DMA transfers are performed alternately.

Figure 7-14. Single Transfer Example 3

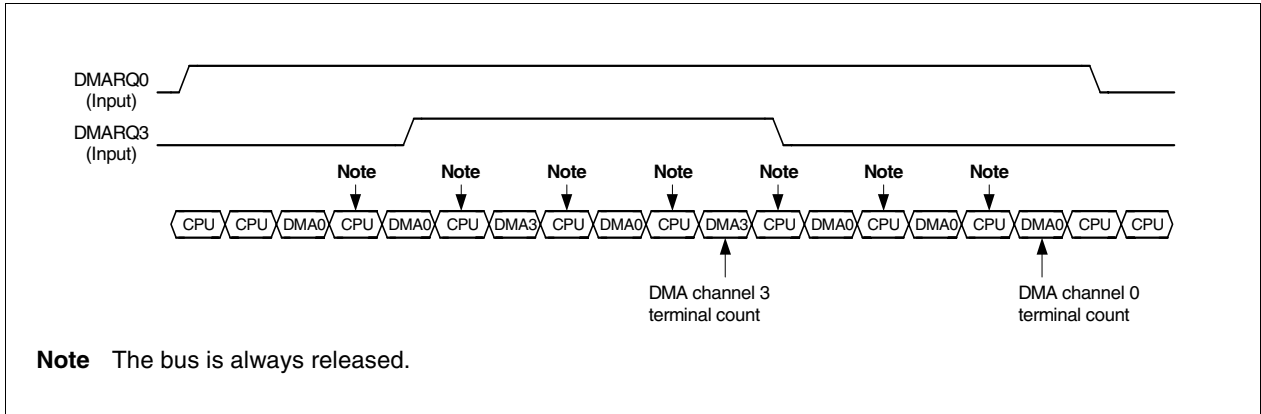
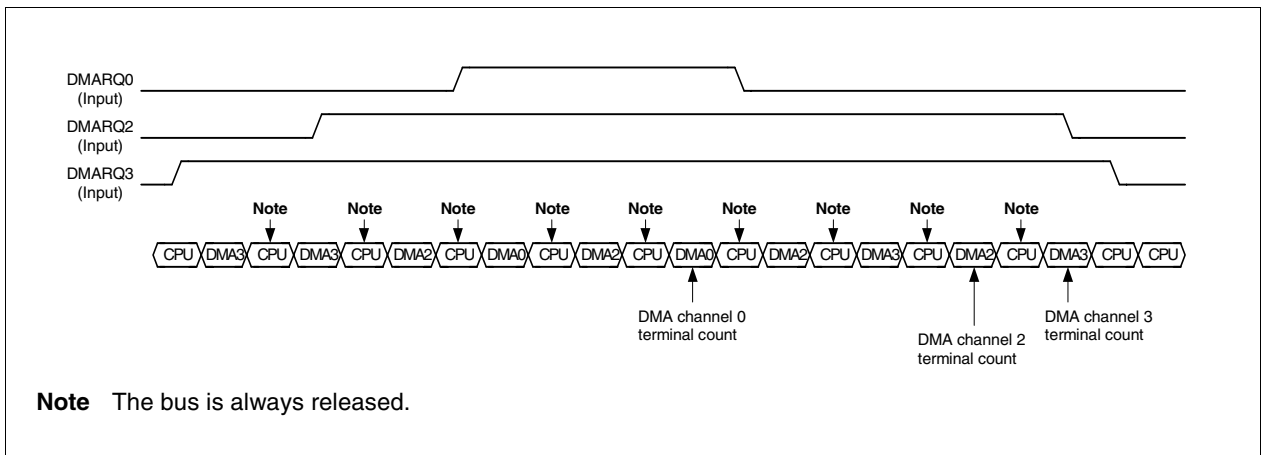


Figure 7-15 shows a single transfer mode example in which two or more lower priority DMA transfer requests are generated within one clock after the end of a single transfer. DMA channels 0, 2, and 3 are used for a single transfer. When three or more DMA transfer request signals are activated at the same time, always the two highest priority DMA transfers are performed alternately.

Figure 7-15. Single Transfer Example 4



7.8.2 Single-step transfer mode

In single-step transfer mode, the DMAC releases the bus after each byte, halfword, or word transfer. Once a DMA transfer request signal (DMARQ3 to DMARQ0) is received, this operation continues until a terminal count occurs.

If a higher priority DMA transfer request is generated while the DMAC has released the bus, the higher priority DMA transfer request always takes precedence.

Figures 7-16 and 7-17 show single-step transfer mode examples.

Figure 7-16. Single-Step Transfer Example 1

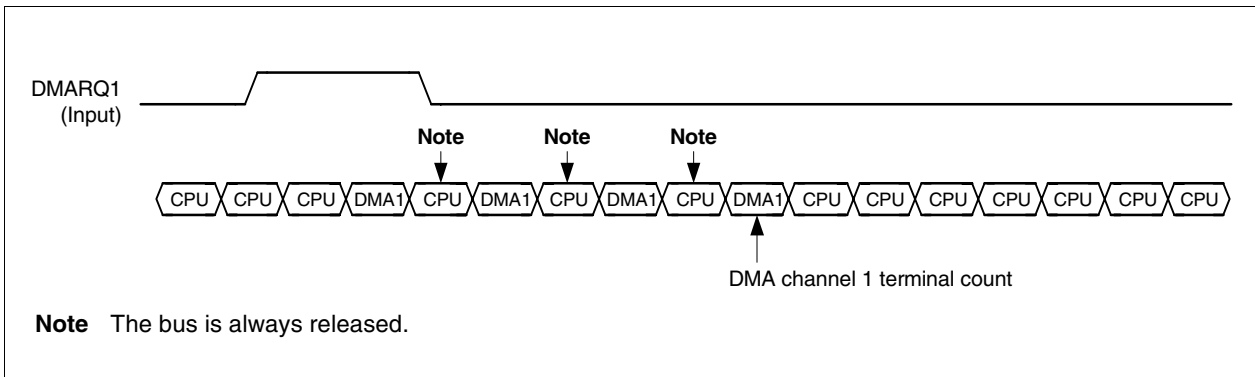
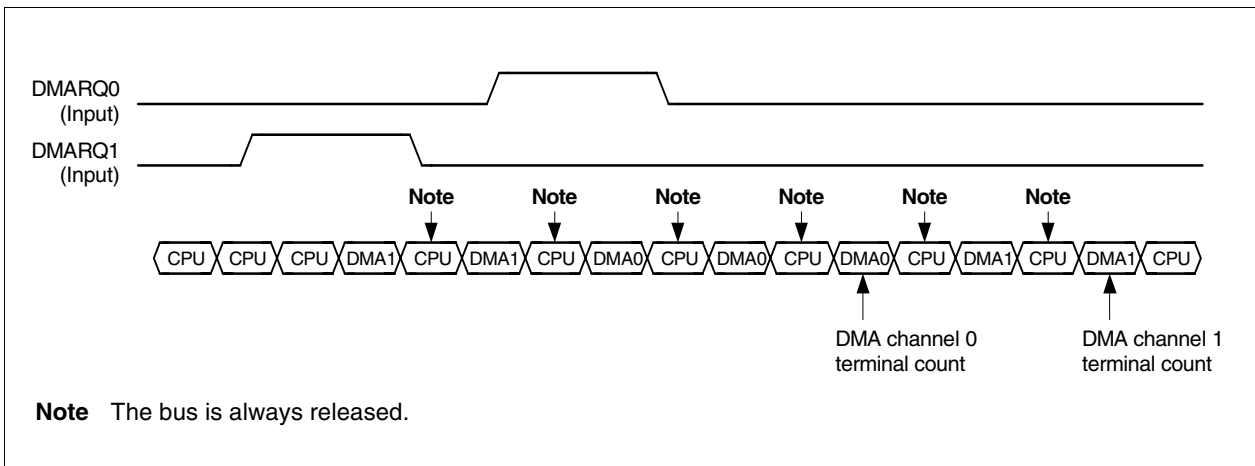


Figure 7-17. Single-Step Transfer Example 2



7.8.3 Line transfer mode

In line transfer mode, the DMAC releases the bus after every four byte, halfword, or word transfers. If there is a subsequent DMA transfer request, four transfers are performed again. This operation continues until a terminal count occurs. In 2-cycle transfer, the operation from read to write is repeated four times.

If a higher priority DMA transfer request is generated while the DMAC has released the bus, the higher priority DMA transfer request always takes precedence. However, if a lower priority DMA transfer request is generated within one clock after the end of a line transfer, even if the previous higher priority DMA transfer request signal stays active, this request is not prioritized, and the next DMA transfer after the bus is released for the CPU is a transfer based on the newly generated, lower priority DMA transfer request.

Figures 7-18 to 7-21 show examples for line transfer.

Figure 7-18. Line Transfer Example 1

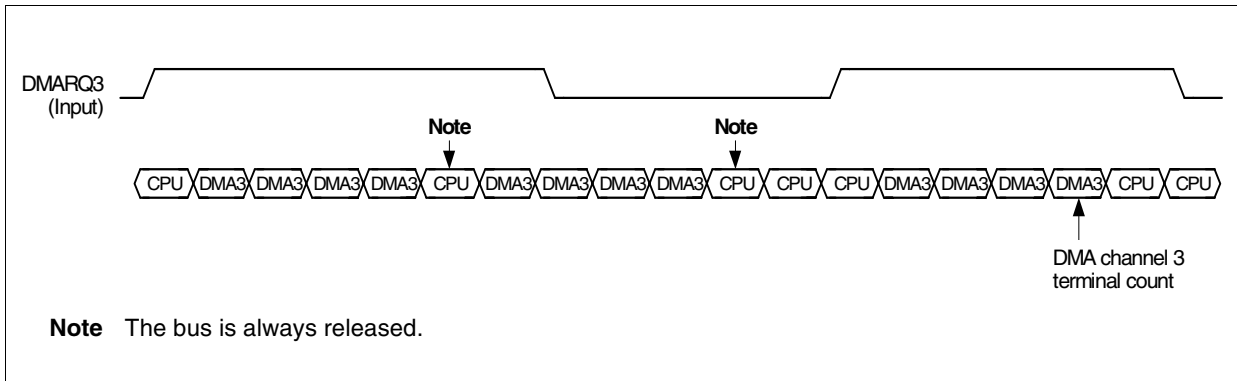
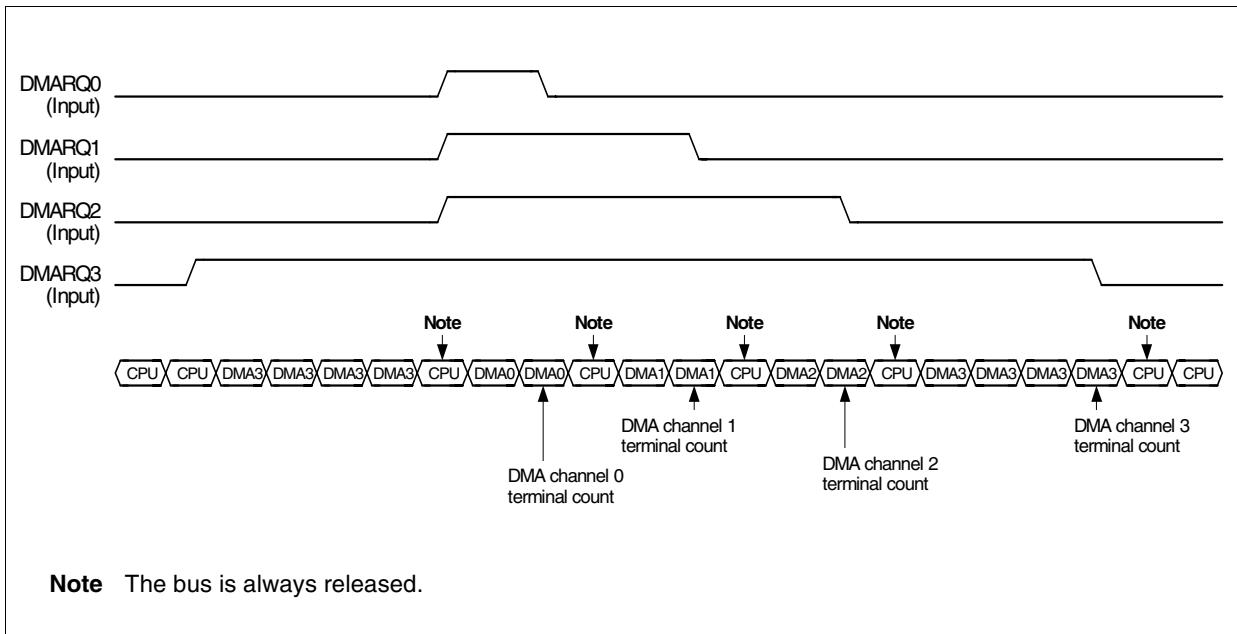


Figure 7-19 shows a line transfer mode example in which a higher priority DMA transfer request is generated. DMA channels 0 to 2 are used for a block transfer, and channel 3 is used for a line transfer.

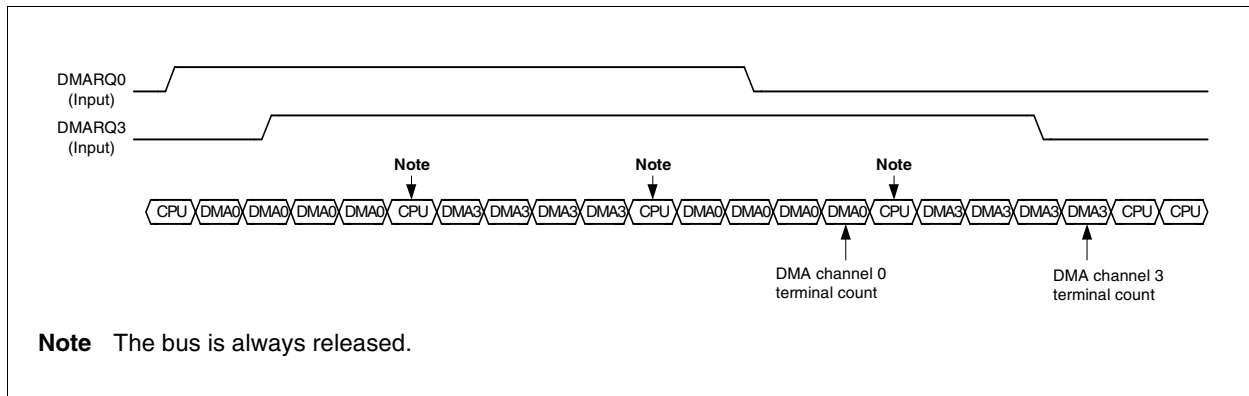
Figure 7-19. Line Transfer Example 2



Figures 7-20 and 7-21 show line transfer mode examples in which a lower priority DMA transfer request is generated within one clock after the end of a line transfer. When two DMA transfer request signals are activated at the same time, the two DMA transfers are performed alternately.

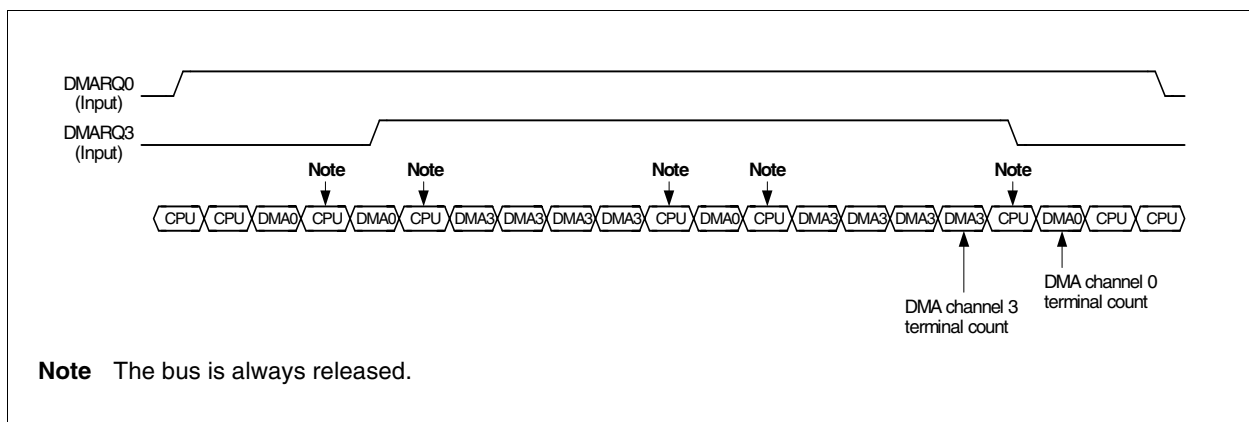
DMA channels 0 and 3 in Figure 7-20 are used for line transfer.

Figure 7-20. Line Transfer Example 3



DMA channel 0 in Figure 7-21 is used for a single transfer, and channel 3 is used for a line transfer.

Figure 7-21. Line Transfer Example 4



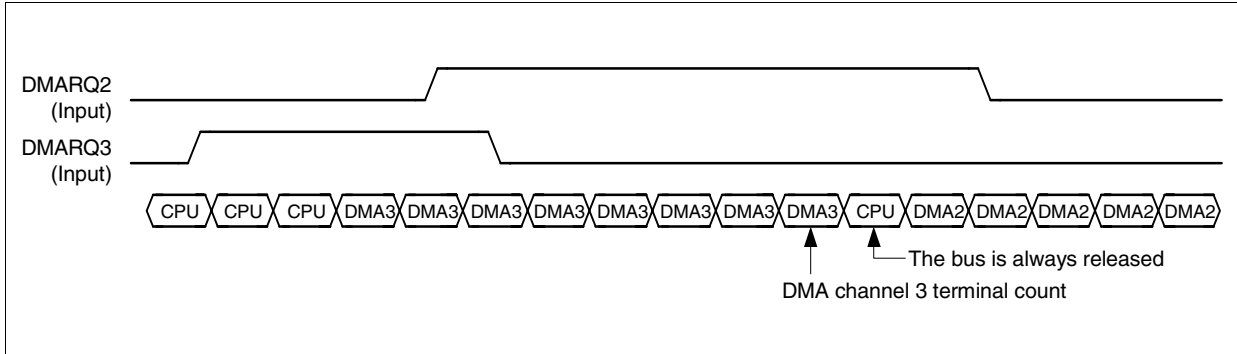
7.8.4 Block transfer mode

In block transfer mode, once transfer begins, the transfers continue without releasing the bus until a terminal count occurs. No other DMA transfer requests are acknowledged during a block transfer.

After the block transfer ends and the DMAC has released the bus, another DMA transfer can be acknowledged. Although it is prohibited to insert a CPU bus cycle during a block transfer, bus mastership can be transferred even during a block transfer in response to a request by the external bus master (including SDRAM refresh).

Figure 7-22 shows a block transfer mode example. It is a block transfer mode example in which a higher priority DMA transfer request is generated. DMA channels 2 and 3 are used for a block transfer.

Figure 7-22. Block Transfer Example



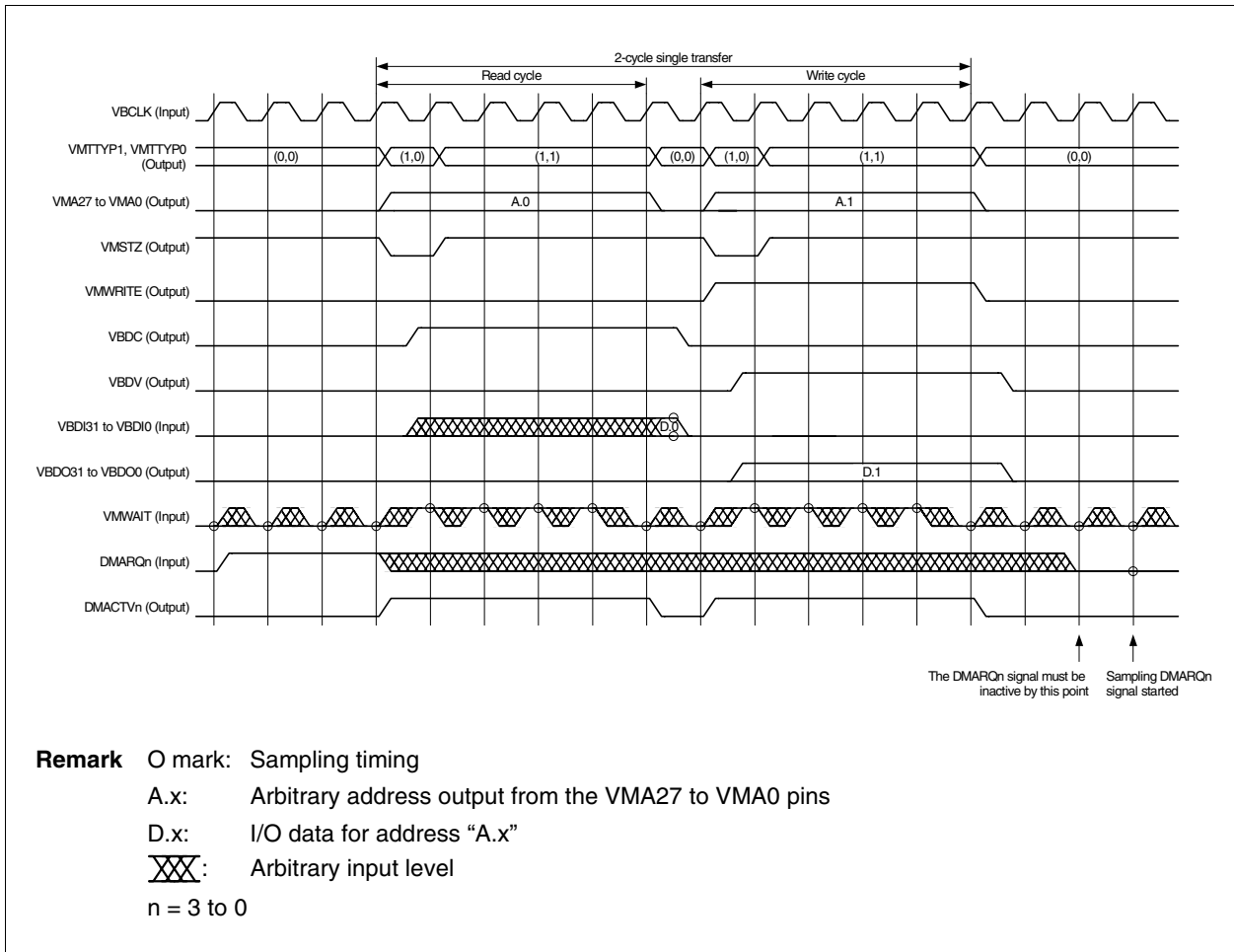
★ 7.8.5 One-time transfer when executing single transfers using DMARQn signal

(1) Two-cycle transfer

When executing single transfers to the external memory using the DMARQn signal input, the next DMARQn signal is acknowledged when its sampling is started at the rise of VBCLK three clocks following the completion of the write cycle of the current 2-cycle transfer. Actually, when the specified DMARQn setup time is satisfied after VBCLK falls 2.5 clocks after completion of the write cycle, the next DMARQn signal request is acknowledged. Therefore, in order to transfer only once, it is recommended that the DMARQn signal be made inactive within 2 clocks of the end of the write cycle of a 2-cycle single transfer (n = 3 to 0).

During a DMA transfer in which the destination is the RAM connected to the VDB, the DMACTVn signal does not become active during transfer to RAM, so if the transfer destination (write cycle) is RAM, the timing when the write cycle ends cannot be determined (n = 3 to 0). When executing a single transfer, whether from memory to RAM or from RAM to memory, the DMACTVn signal becomes active during the memory transfer. In this case, therefore, it is recommended that the DMARQn signal be made inactive within 2 clocks after the DMACTVn signal becomes inactive.

Figure 7-23. One-Time Transfer When Executing Single Transfers Using DMARQn Signal



(2) Flyby transfer

Similar to the 2-cycle transfer, the next DMARQn signal is acknowledged when its sampling is started at the rise of VBCLK three clocks following the completion of the write cycle of the current 2-cycle transfer. Actually, when the specified DMARQn setup time is satisfied after VBCLK falls 2.5 clocks after completion of the write cycle, the next DMARQn signal request is acknowledged. Therefore, in order to transfer only once, it is recommended that the DMARQn signal be made inactive within 2 clocks of the end of the write cycle of a 2-cycle single transfer (n = 3 to 0).

7.9 Transfer Types

7.9.1 Two-cycle transfer

In a 2-cycle transfer, data is transferred in 2 cycles: a read cycle (transfer source to DMAC) and a write cycle (DMAC to transfer destination).

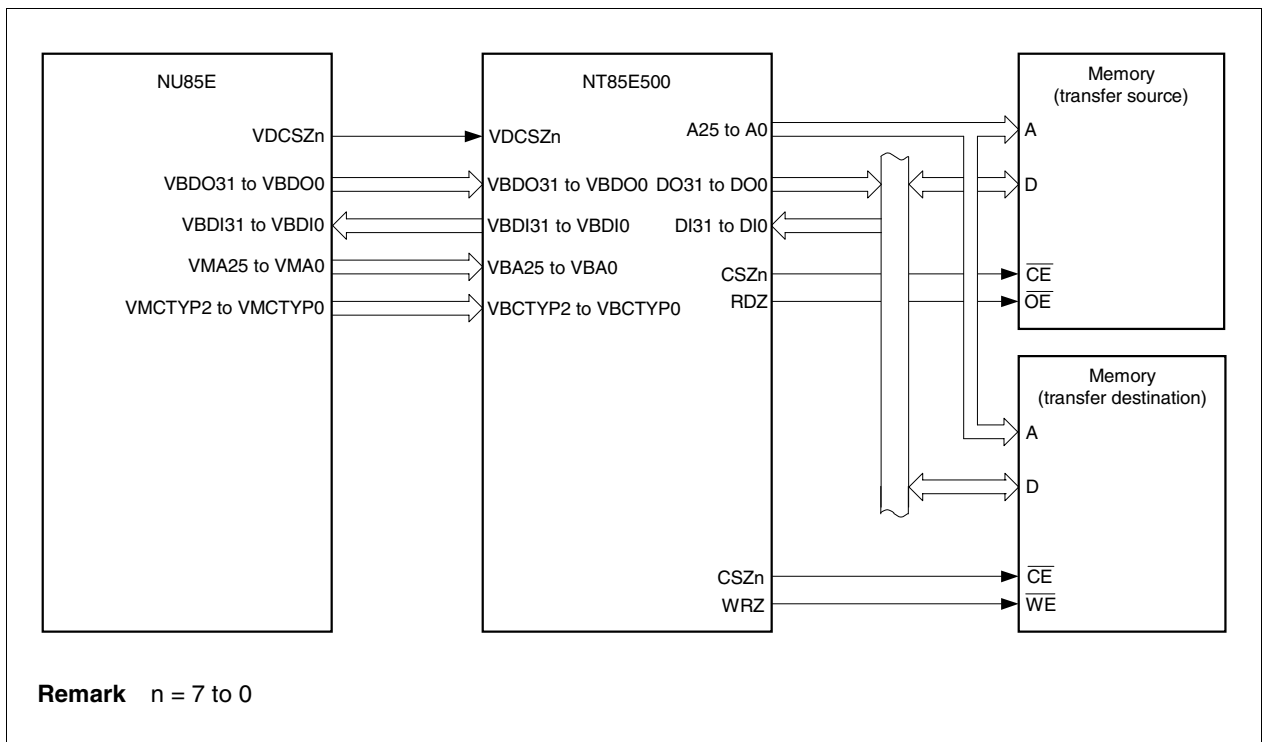
In the first cycle, the transfer source address is output to read data from the transfer source to the DMAC. In the second cycle, the transfer destination address is output to write data from the DMAC to the transfer destination.

The signals indicating 2-cycle DMA transfer (1, 1, 0) are output from the VMCTYP2 to VMCTYP0 pins.

Caution A one-clock idle cycle is always inserted between a read cycle and a write cycle.

★

Figure 7-24. Example of Two-Cycle Transfer



7.9.2 Flyby transfer

A flyby transfer can be executed only when the MEMC supports flyby transfers.

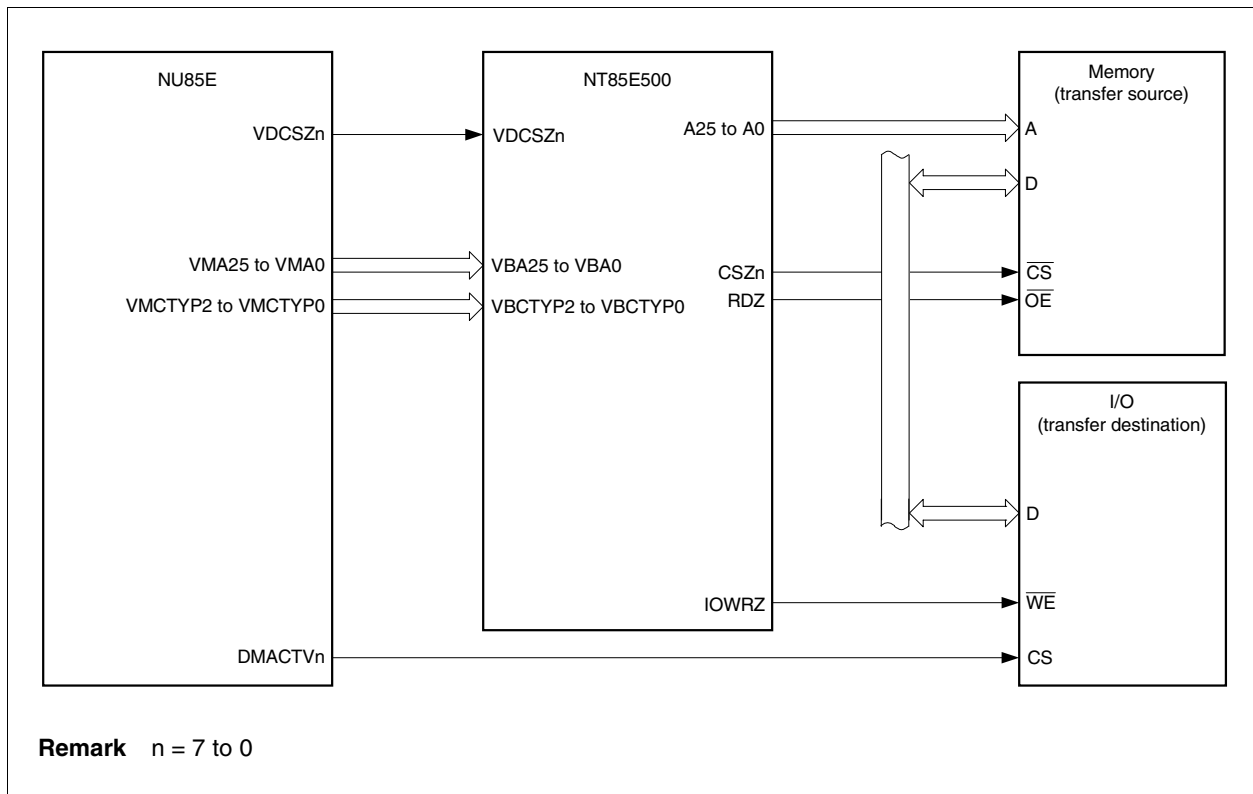
- ★ Flyby transfer executes a transfer from memory to I/O or from I/O to memory in one cycle. The NU85ET always outputs the address on the memory side set in the DSA_nH or DSA_nL registers for either transfer from memory to I/O or from I/O to memory (n = 3 to 0).

The strobe signal to the memory or to the external I/O simultaneously makes the RDZ/IOWRZ signals and WRZ/IORDZ active during transfer from memory to I/O and from I/O to memory, respectively. Signals indicating DMA flyby transfer (1, 1, 1) are also output from the VMCTYP2 to VMCTYP0 pins. Only the data bus on the memory side of the memory controller is used for data, so the VBDI31 to VBDI0 and VBDO31 to VBDO0 signals, which are for VSB data, are not used.

The external I/O is selected according to the DMACTV3 to DMACTV0 signals.

Caution When NA85E535 is used as a memory controller, flyby transfer with SDRAM is possible, except in a system in which the SDRAM controller (NT85E502) is connected to the NT85E500.

★ **Figure 7-25. Example of Flyby Transfer (Memory to I/O)**



7.10 DMA Transfer Start Factors

DMA transfer can be started by the following two factors.

(1) Request by external pin (DMARQn)

If the ENn bit of the DCHCn register is set to 1 and the TCn bit is set to 0, the DMARQn signal becomes active in TI state (n = 3 to 0). If the DMARQn signal becomes active in TI state, the DMAC moves to T0 state and DMA transfer begins.

(2) Request by software

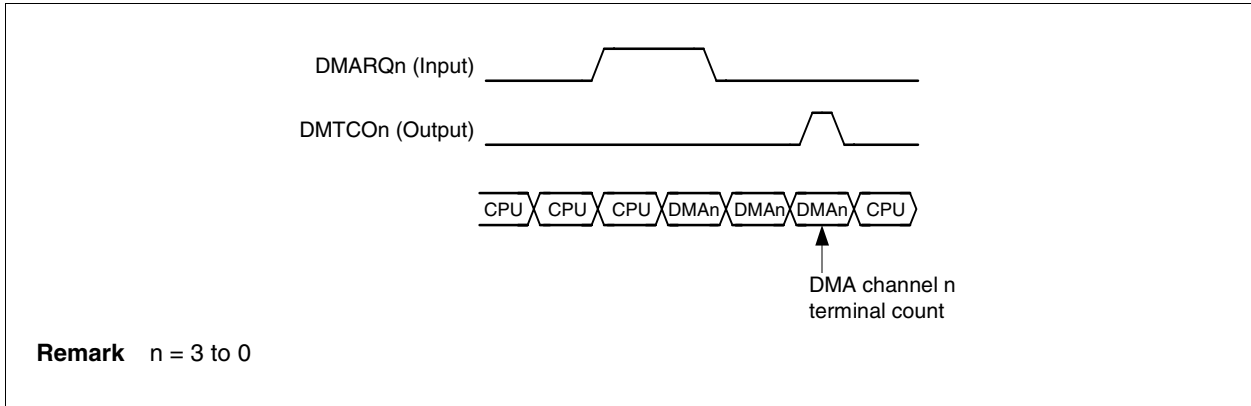
If the STGn, ENn, and TCn bits of the DCHCn register are set as follows, DMA transfer begins (n = 0 to 3).

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

7.11 Terminal Count Output When DMA Transfer Is Complete

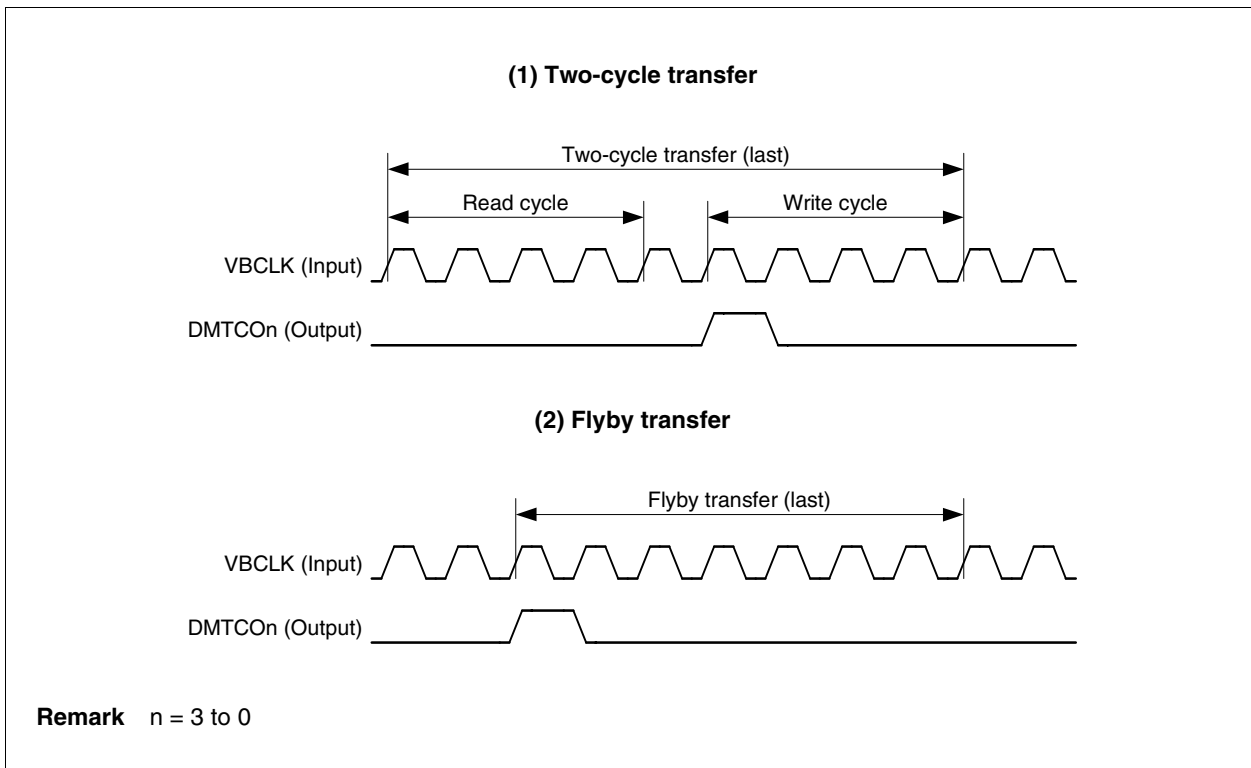
The terminal count signal (DMTCOn) becomes active for only one clock in the final DMA transfer cycle (n = 3 to 0).

Figure 7-26. Timing Example of Terminal Count Signals (DMTCO3 to DMTCO0)



During 2-cycle transfer, the signal becomes active for one clock at the beginning of the last write cycle.
 During flyby transfer, the signal becomes active for one clock at the beginning of the last transfer cycle.

★ **Figure 7-27. Example of Terminal Count Signal Output (DMTCO3 to DMTCO0)**



7.12 Forcible Interruption

DMA transfer can be forcibly interrupted by inputting the IDMASTP signal during the DMA transfer.

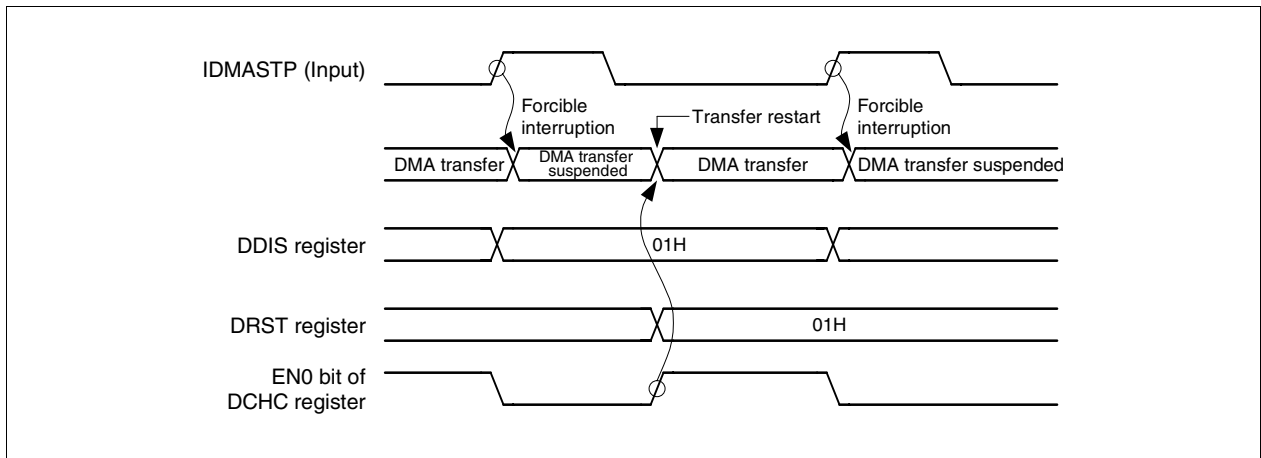
At this time, the DMAC clears (0) the ENn bit of the DCHCn register of all channels to set the state in which DMA transfer is disabled, completes the DMA transfer that was being executed when the IDMASTP signal was input, and the bus releases to the CPU (n = 0 to 3).

For single-step transfer mode, block transfer mode, or line transfer mode, the DMA transfer request is maintained in the DMAC. When the ENn bit is set (1), the DMA transfer is restarted from the point at which the DMA transfer was interrupted.

For single transfer mode, when the ENn bit is set (1), the next DMA transfer request is acknowledged and DMA transfer begins.

Caution To forcibly interrupt DMA transfer and stop the next transfer from occurring, the IDMASTP signal must be made active before the end of the DMA transfer currently under execution. Moreover, although it is possible to restart DMA transfer following an interruption, this transfer cannot be executed under new settings (new conditions). Execute DMA transfer under new settings either after the end of the current transfer or after transfer has been forcibly terminated by setting the INITn bit of the DCHCn register (n = 0 to 3).

Figure 7-28. DMA Transfer Forcible Interruption Example



7.13 Forcible Termination

By setting (1) the INITn bit of the DCHCn register during a DMA transfer, it is possible to forcibly terminate the DMA transfer under execution. The following is an example of the operation of a forcible termination (n = 0 to 3).

Caution The setting (1) of the INITn bit is performed when the VSB has been released to the CPU (n = 0 to 3). Therefore, because the VSB is locked until the DMA transfer has completely finished in a block transfer using the VSB, it is not possible to exercise a forcible termination during this transfer.

Figure 7-29. DMA Transfer Forcible Termination Example (1/2)

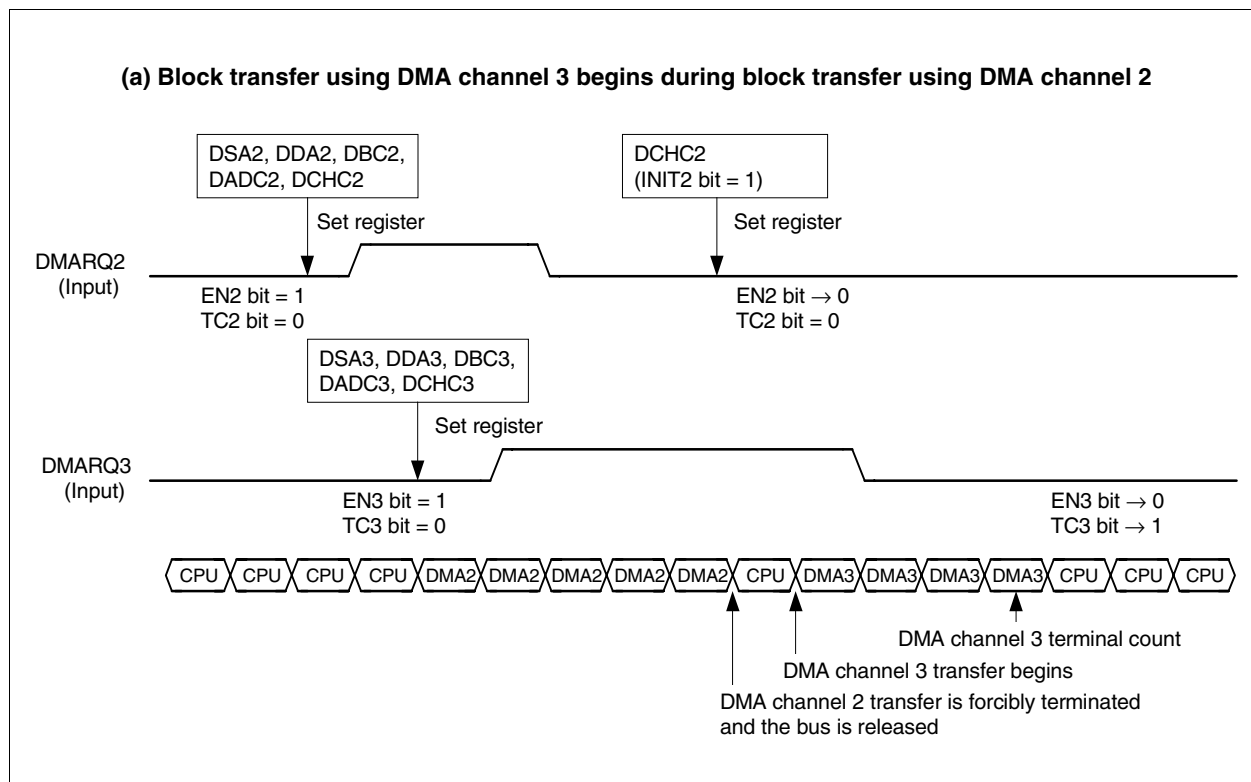
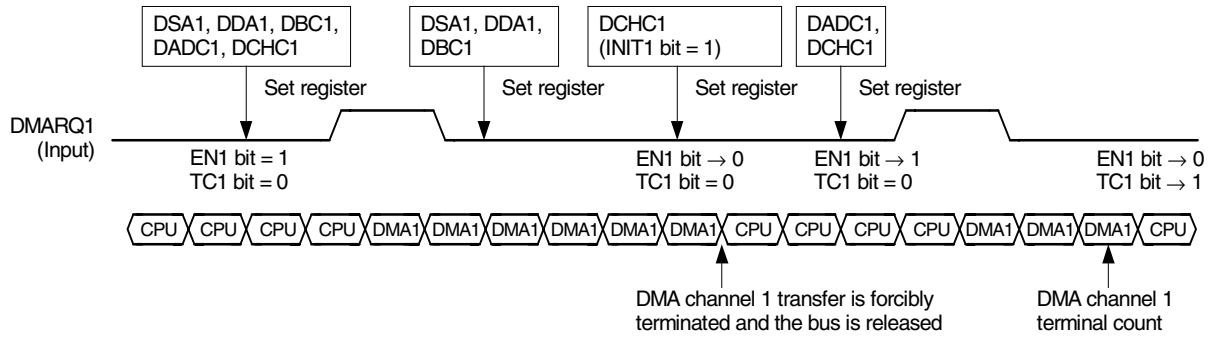


Figure 7-29. DMA Transfer Forcible Termination Example (2/2)

(b) The transfer is forcibly terminated during block transfer using DMA channel 1 and a transfer with another condition is executed



★ **Remark** Since the DSA_n, DDA_n, and DBC_n registers are buffer registers with an FIFO configuration, the values are retained even after a forcible termination. Also, the next transfer condition can be set even during a DMA transfer. However, a setting in the DADC_n register is ignored (See 7.6 Next Address Setting Function).

7.14 DMA Transfer Timing Examples

Examples of the DMA transfer timing in each transfer mode are shown in the following pages.

The NT85E500 and the NT85E502 are provided as MEMCs for the NU85E. This section gives examples in the case that the NT85E500 and the NT85E502 are used.

(1) Two-cycle transfer

Figures 7-30 to 7-33 show examples of the timing of 2-cycle transfers between external SRAMs connected to the MEMC (NT85E500). Figures 7-34 and 7-35 show examples of the timing of 2-cycle transfers between RAM connected to the VDB and SDRAM connected to the MEMC (NT85E502).

- Remarks**
1. The levels of the broken-line portions of the VMCTYP2 to VMCTYP0, VMSEQ2 to VMSEQ0, VMSIZE1, VMSIZE0, and DI0 signals indicate the undefined state.
 2. The O marks indicate the sampling timing.
 3. n = 3 to 0

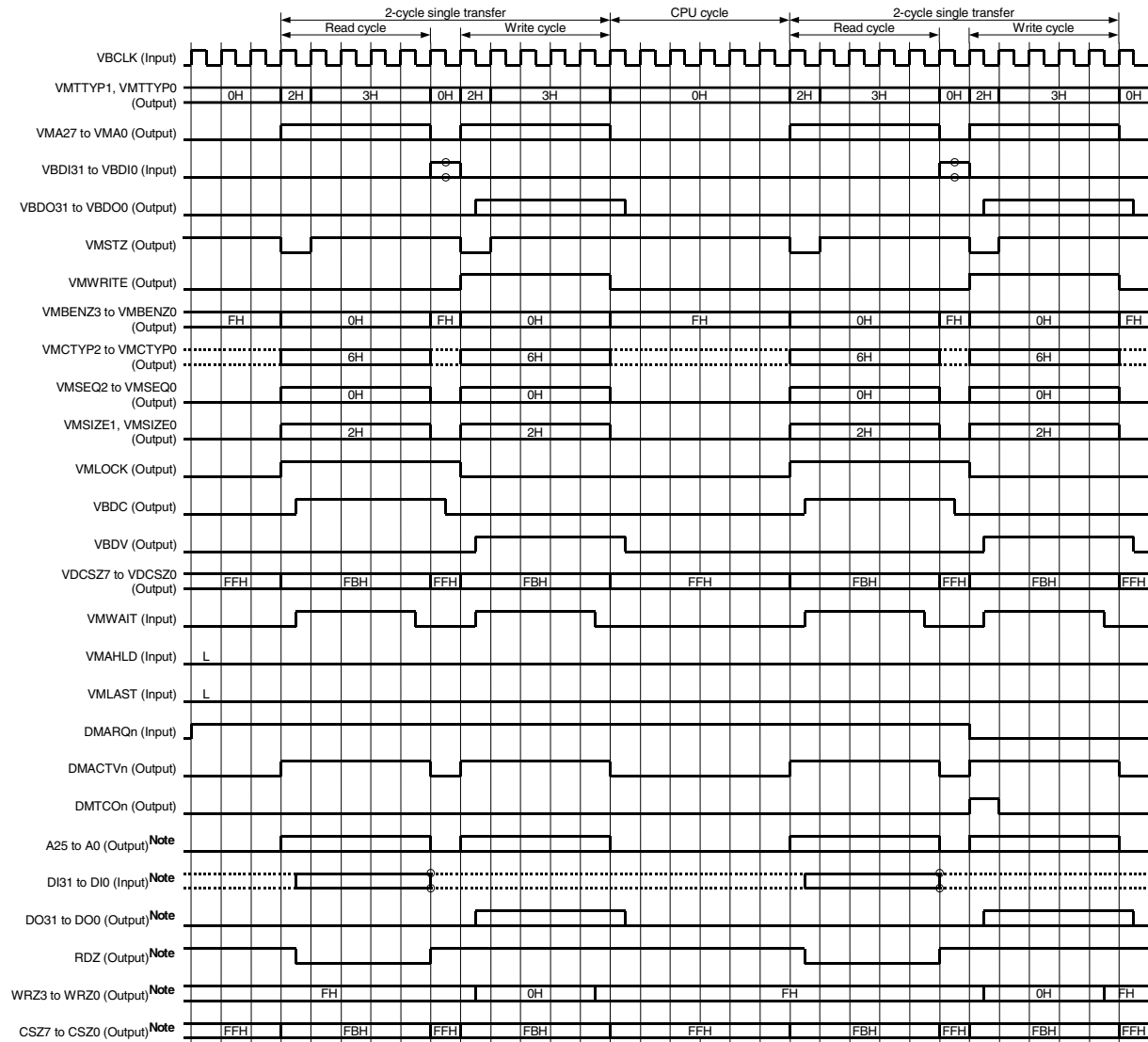
Figure 7-30 shows an example of the timing of a 2-cycle single transfer (between external SRAMs connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0001H (2 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 7377H (CS2 wait states = 3)

Note An NT85E500 register.

Figure 7-30. Example of Two-Cycle Single Transfer Timing (Between External SRAMs Connected to NT85E500)



Note These are NT85E500 signals.

Figure 7-31 shows an example of the timing of a 2-cycle single-step transfer (between external SRAMs connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0002H (3 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 7377H (CS2 wait states = 3)

Note An NT85E500 register.

Figure 7-31. Example of Two-Cycle Single-Step Transfer Timing (Between External SRAMs Connected to NT85E500)



Note These are NT85E500 signals.

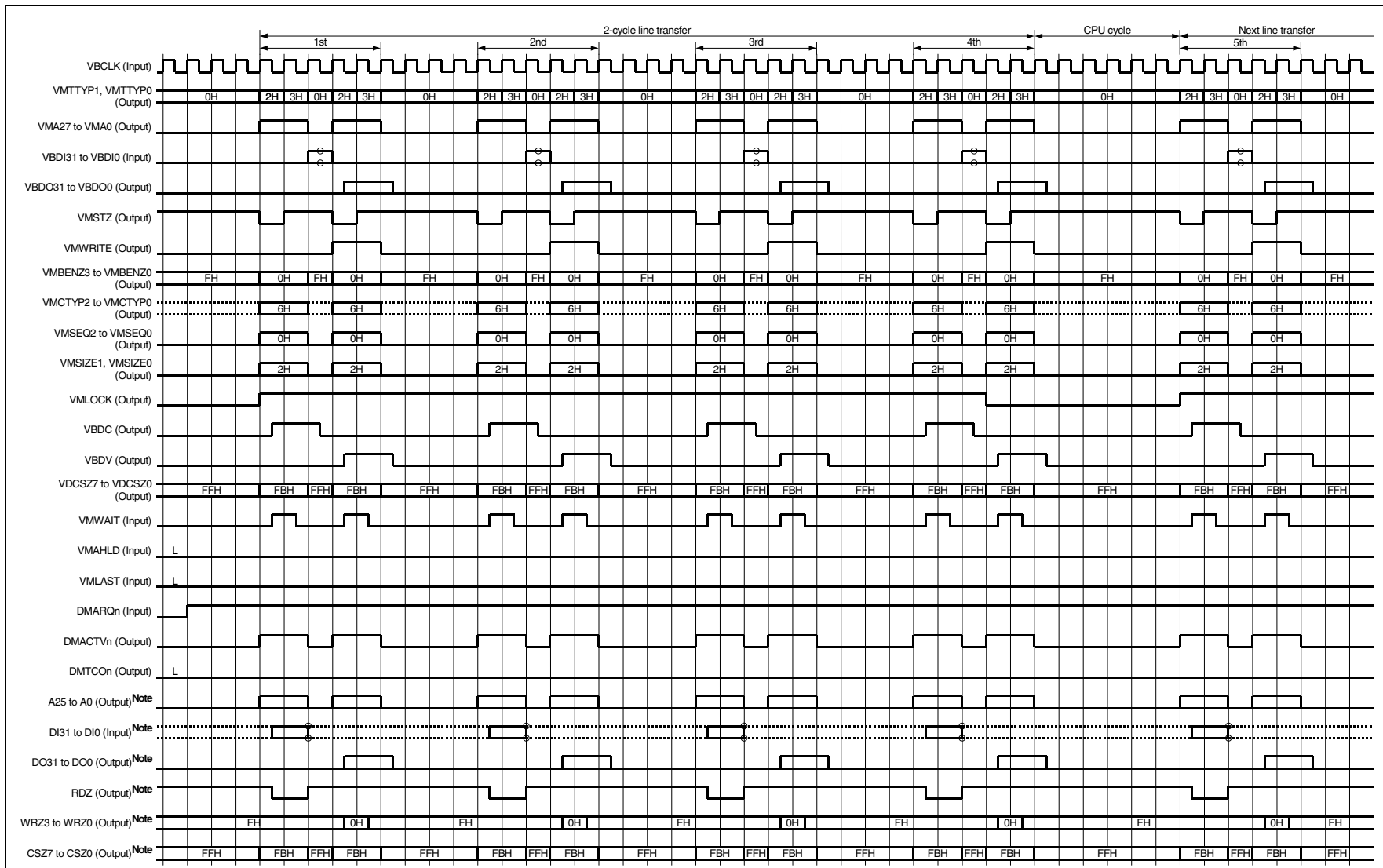
Figure 7-32 shows an example of the timing of a 2-cycle line transfer (between the external SRAMs connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0007H (8 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 7077H (No CS2 wait states)

Note An NT85E500 register.

Figure 7-32. Example of Two-Cycle Line Transfer Timing (Between External SRAMs Connected to NT85E500)



Note These are NT85E500 signals.

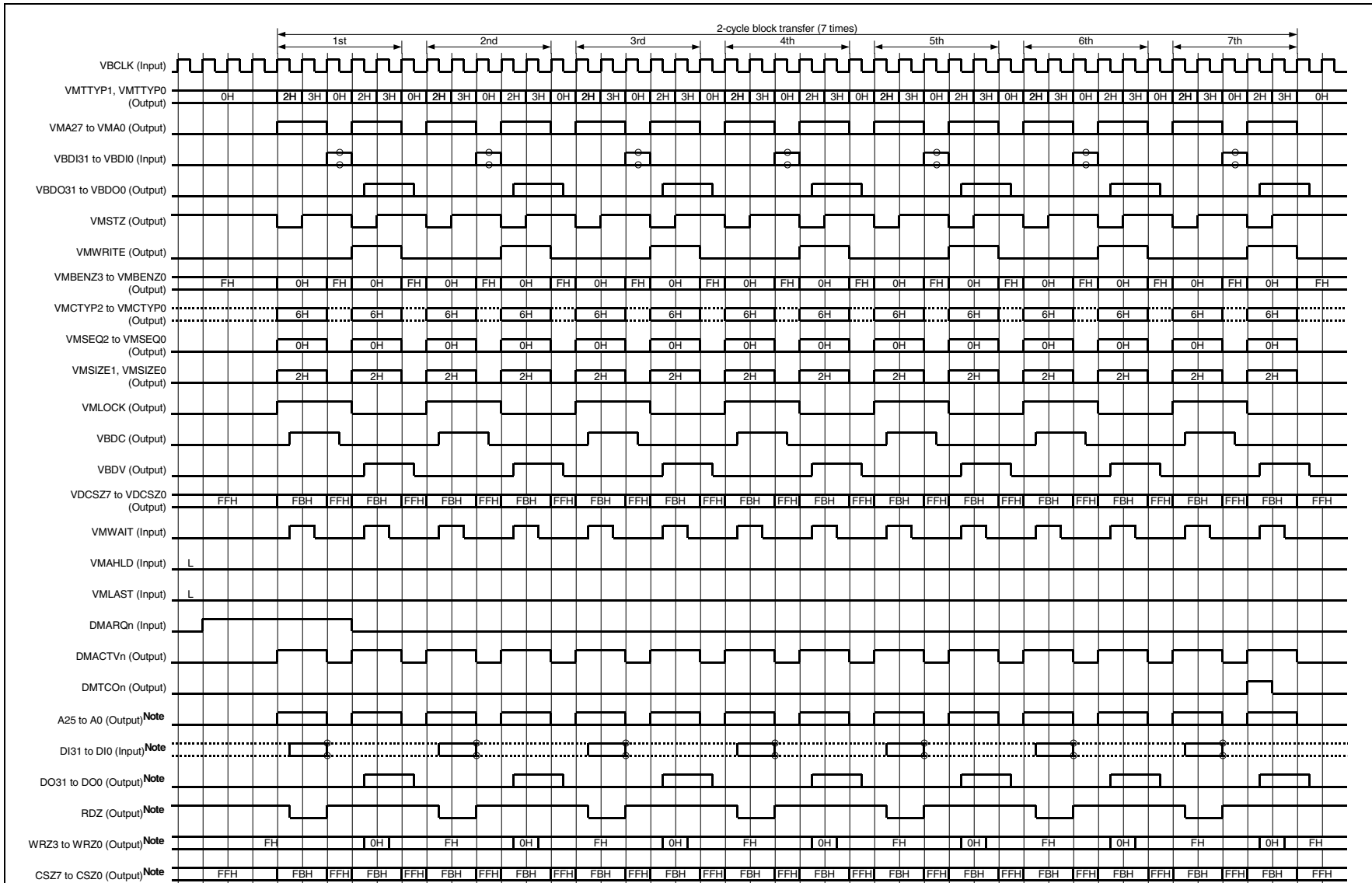
Figure 7-33 shows an example of the timing of a 2-cycle block transfer (between the external SRAMs connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0006H (7 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 7077H (No CS2 wait states)

Note An NT85E500 register.

Figure 7-33. Example of Two-Cycle Block Transfer Timing (Between External SRAMs Connected to NT85E500)



Note These are NT85E500 signals.

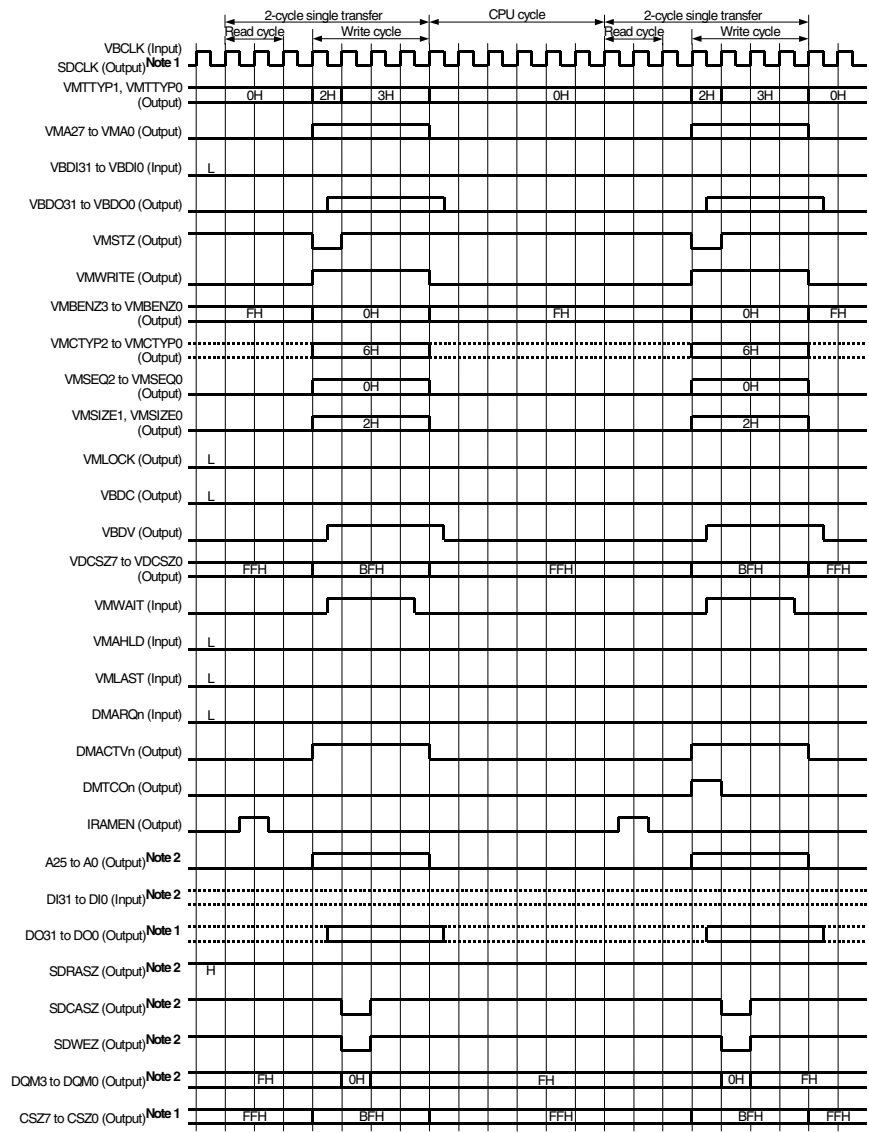
Figure 7-34 shows an example of the timing of a 2-cycle single transfer (from RAM connected to the VDB to SDRAM connected to the NT85E502). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0001H (2 transfers)
- SCRn register^{Note} = 2062H (CAS latency = 2,
number of wait states = 1,
address shift width = 2 bits (32-bit data bus),
low address width = 11 bits,
address multiplexed width = 10 bits)

Note An NT85E502 register.

Figure 7-34. Example of Two-Cycle Single Transfer Timing (from RAM Connected to VDB to SDRAM Connected to NT85E502)



Notes 1. These are NT85E500 signals.

2. These are NT85E502 signals.

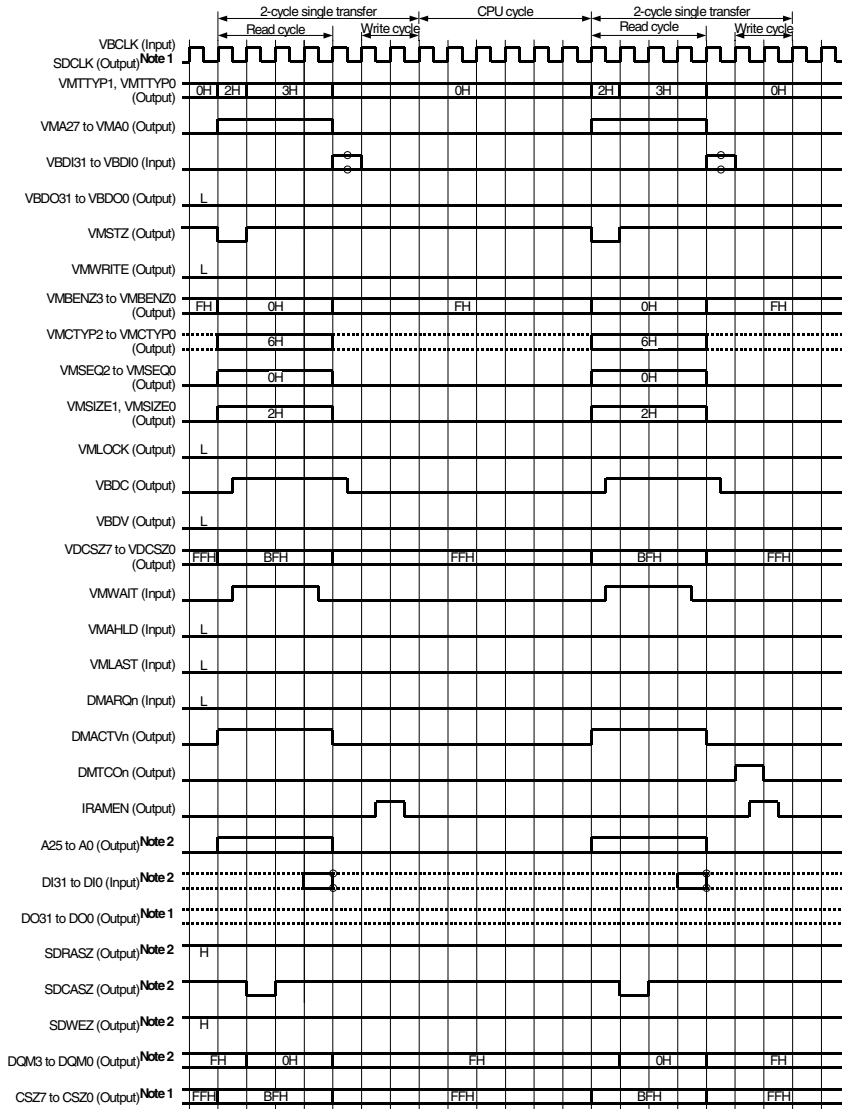
Figure 7-35 shows an example of the timing of a 2-cycle single transfer (from SDRAM connected to the NT85E502 to RAM connected to the VDB). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0001H (2 transfers)
- SCRn register^{Note} = 2062H (CAS latency = 2,
number of wait states = 1,
address shift width = 2 bits (32-bit data bus),
low address width = 11 bits,
address multiplexed width = 10 bits)

Note An NT85E502 register.

Figure 7-35. Example of Two-Cycle Single Transfer Timing (from SDRAM Connected to NT85E502 to RAM Connected to VDB)



Notes 1. These are NT85E500 signals.

2. These are NT85E502 signals.

(2) Flyby transfers

Figures 7-36 to 7-41 show examples of the timing of flyby transfers between external SRAM and external I/O connected to the MEMC (NT85E500). The flyby transfer consists of the following states.

- T1, T2 states: These are basic states for accessing the NT85E500.
- T3 state: This is a basic state added for flyby transfer.
- TA state: This is an address setting wait state inserted by means of a setting in the NT85E500's ASC register.
- TI state: This is an idle state inserted by means of a setting in the NT85E500's BCC register.
- TW state: This is a wait state inserted by means of a setting in the NT85E500's DWC0 register.

Remarks 1. The levels of the broken-line portions of the VMCTYP2 to VMCTYP0, VMSEQ2 to VMSEQ0, VMSIZE1, VMSIZE0, and DI31 to DI0 signals indicate the undefined state.

2. n = 3 to 0

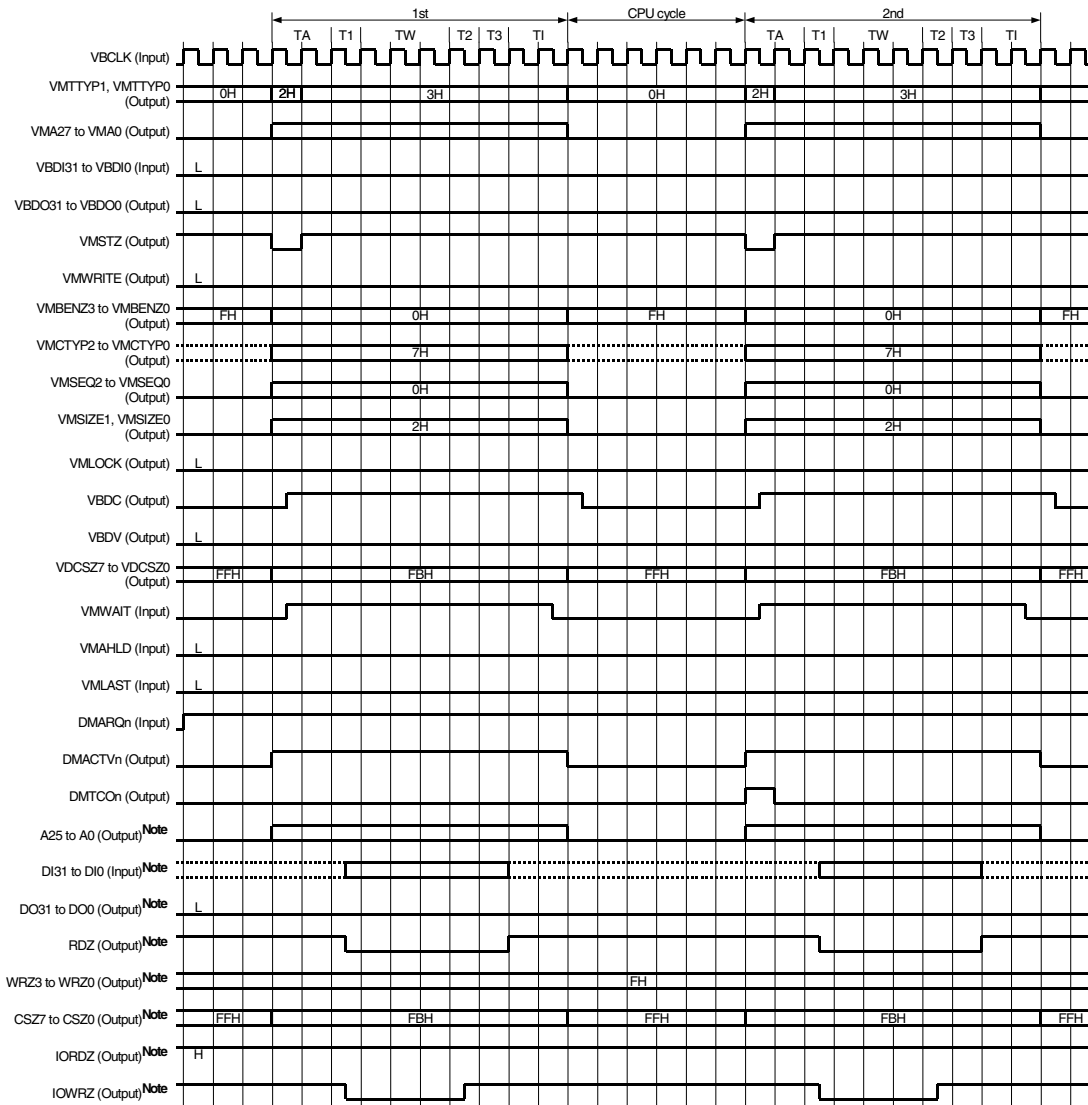
Figure 7-36 shows an example of the timing of a flyby single transfer (from external SRAM to external I/O connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0001H (2 transfers)
- ASC register^{Note} = FFEFH (CS2 address setting wait states = 2)
- BCC register^{Note} = FFEFH (CS2 idle states = 2)
- DWC0 register^{Note} = 7377H (CS2 wait states = 3)

Note An NT85E500 register.

Figure 7-36. Example of Flyby Single Transfer Timing (from External SRAM to External I/O Connected to NT85E500)



Note These are NT85E500 signals.

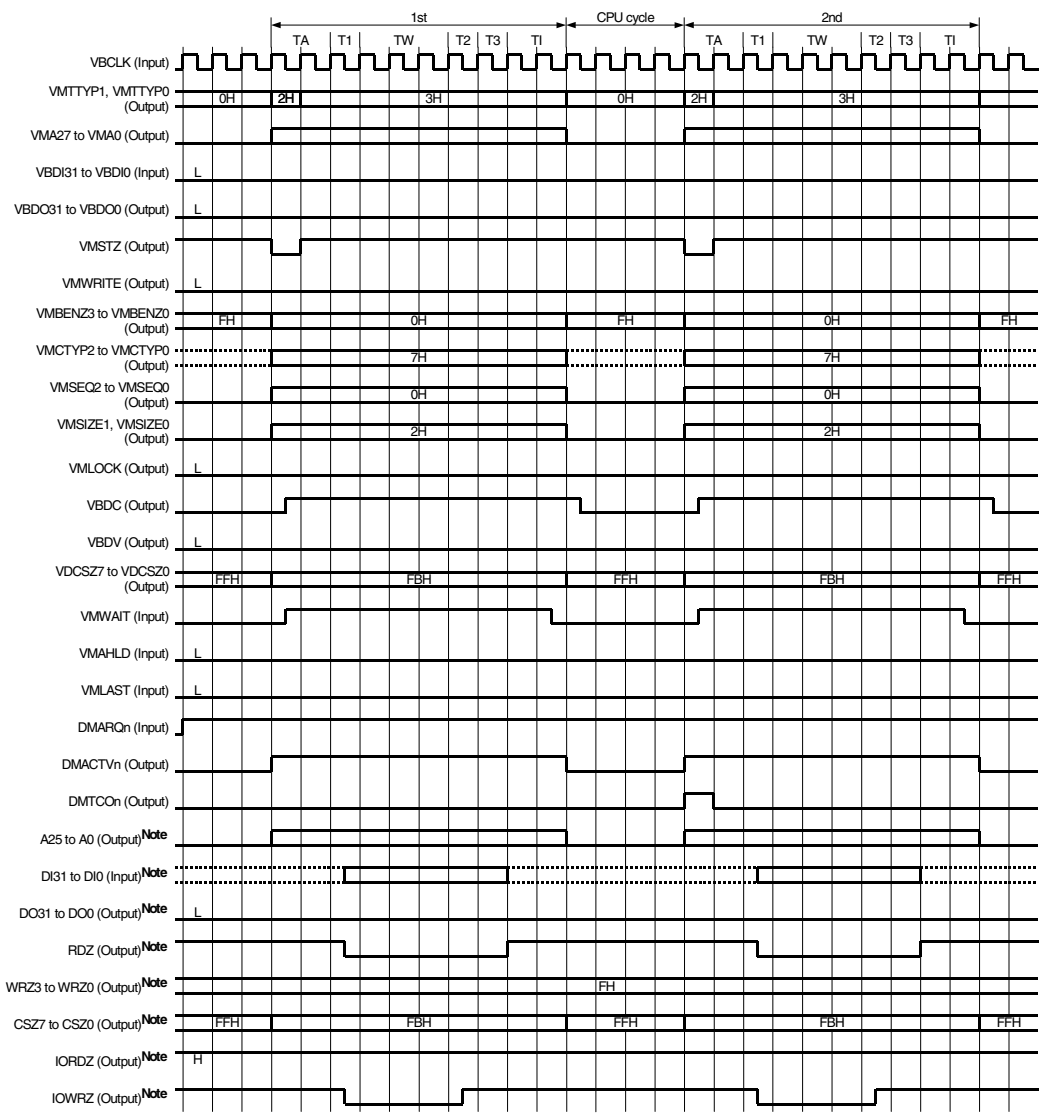
Figure 7-37 shows an example of the timing of a flyby single-step transfer (from external SRAM to external I/O connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0001H (2 transfers)
- ASC register^{Note} = FFEFH (CS2 address setting wait states = 2)
- BCC register^{Note} = FFEFH (CS2 idle states = 2)
- DWC0 register^{Note} = 7377H (CS2 wait states = 3)

Note An NT85E500 register.

Figure 7-37. Example of Flyby Single-Step Transfer Timing (from External SRAM to External I/O Connected to NT85E500)



Note These are NT85E500 signals.

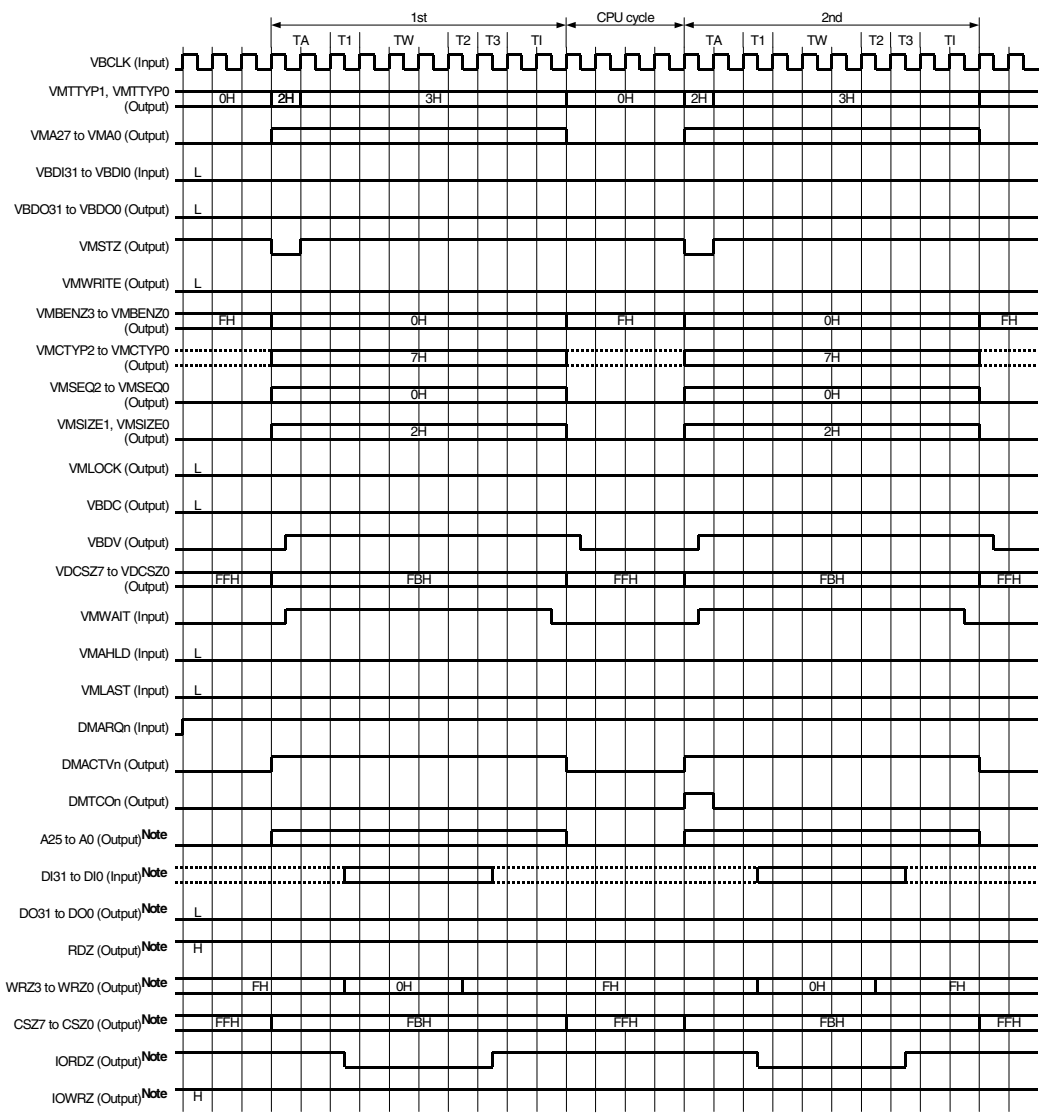
Figure 7-38 shows an example of the timing of a flyby single-step transfer (from external I/O to external SRAM connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0001H (2 transfers)
- ASC register^{Note} = FFEFH (CS2 address setting wait states = 2)
- BCC register^{Note} = FFEFH (CS2 idle states = 2)
- DWC0 register^{Note} = 7377H (CS2 wait states = 3)

Note An NT85E500 register.

Figure 7-38. Example of Flyby Single-Step Transfer Timing (from External I/O to External SRAM Connected to NT85E500)



Note These are NT85E500 signals.

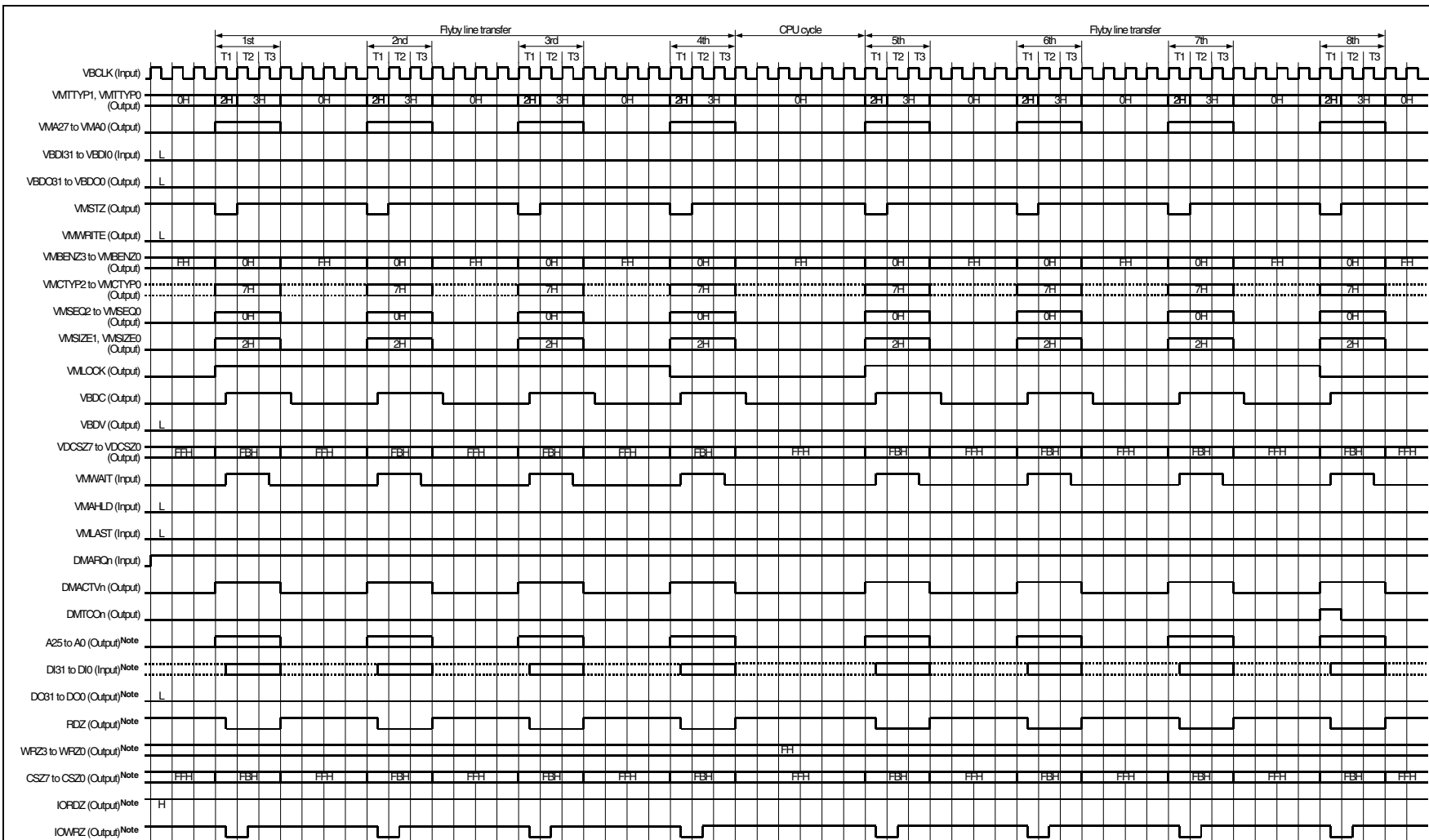
Figure 7-39 shows an example of the timing of a flyby line transfer (from external SRAM to external I/O connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0007H (8 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 0000H (No wait states)

Note An NT85E500 register.

Figure 7-39. Example of Flyby Line Transfer Timing (from External SRAM to External I/O Connected to NT85E500)



Note These are NT85E500 signals.

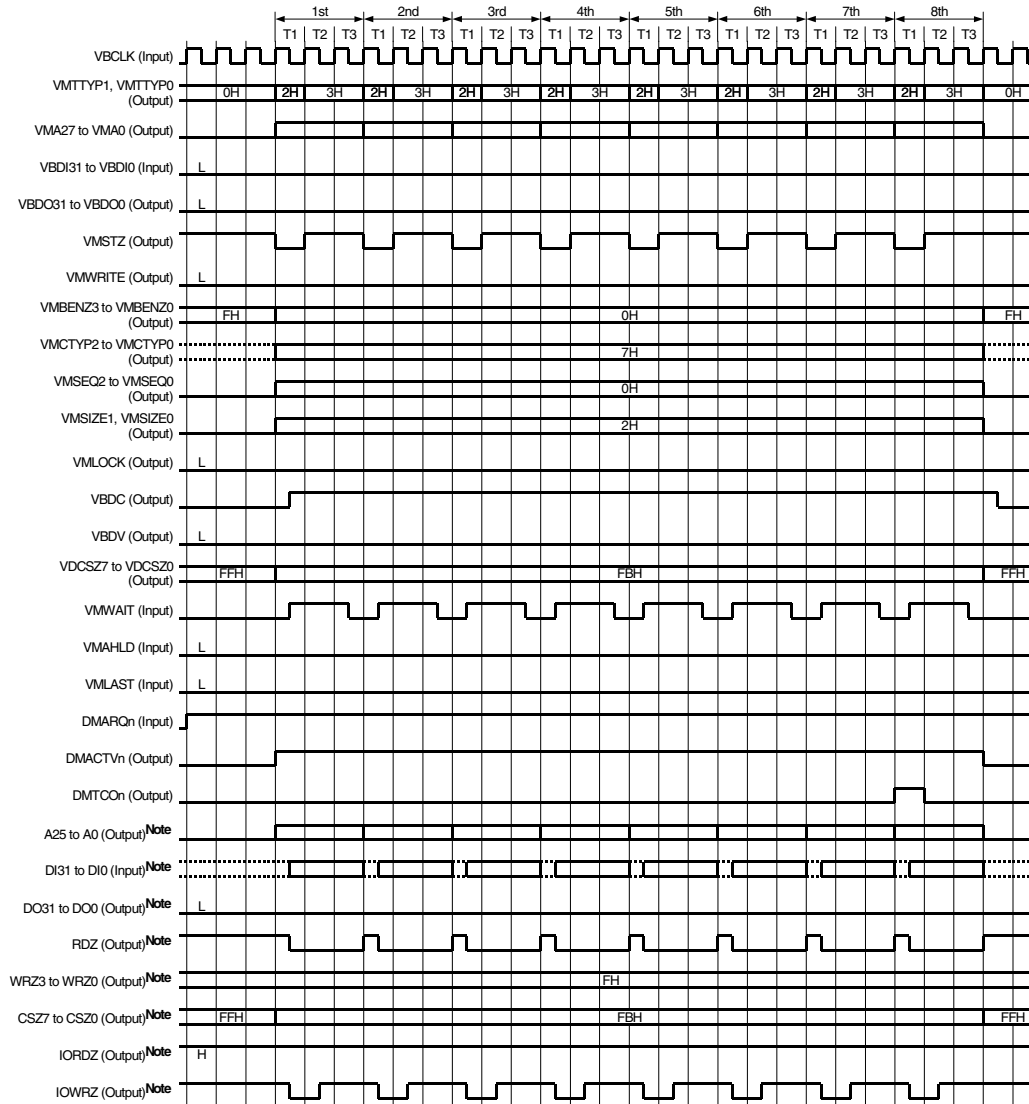
Figure 7-40 shows an example of the timing of a flyby block transfer (from external SRAM to external I/O connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0007H (8 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 0000H (No wait states)

Note An NT85E500 register.

Figure 7-40. Example of Flyby Block Transfer Timing (from External SRAM to External I/O Connected to NT85E500)



Note These are NT85E500 signals.

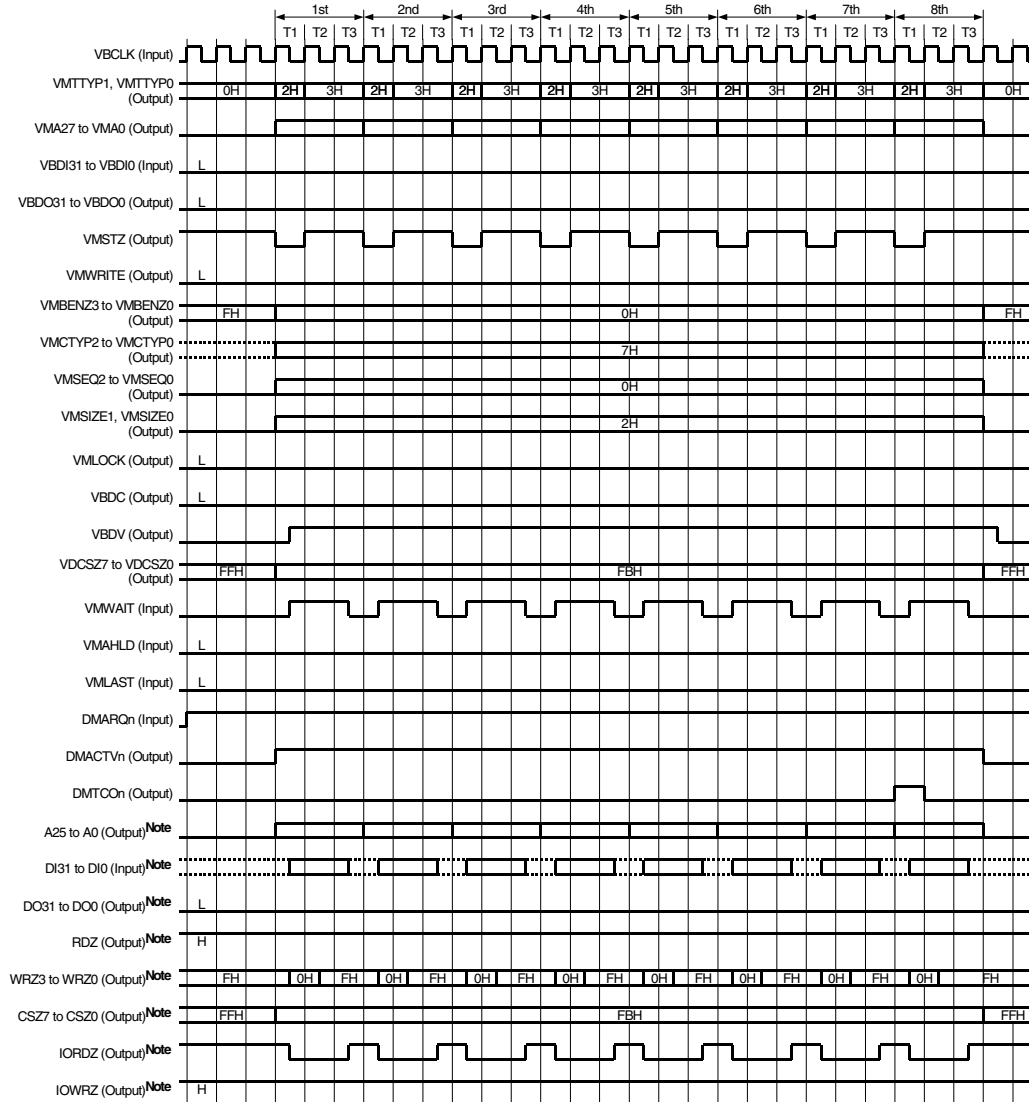
Figure 7-41 shows an example of the timing of a flyby block transfer (from external I/O to external SRAM connected to the NT85E500). The settings of the registers in this figure are as follows.

[Register settings]

- DBCn register = 0007H (8 transfers)
- ASC register^{Note} = 0000H (No address setting wait states)
- BCC register^{Note} = 0000H (No idle states)
- DWC0 register^{Note} = 0000H (No wait states)

Note An NT85E500 register.

Figure 7-41. Example of Flyby Block Transfer Timing (from External I/O to External SRAM Connected to NT85E500)



Note These are NT85E500 signals.

7.15 Precautions

(1) Memory boundary

Operation is not guaranteed if the address of the transfer source or transfer destination is outside of the area for the DMA object (external memory, RAM, or peripheral macro) during a DMA transfer.

(2) Misalign data transfer

DMA transfer of misalign data with a 32-bit or 16-bit bus width is not supported.

(3) Intervals related to DMA transfer

The overhead before a DMA transfer and the minimum number of clocks required for a DMA transfer are shown below.

- From the acknowledgement of the DMARQn signal until the rising edge of the DMACTVn signal (n = 3 to 0): 3 clocks
- ★ • From when the DMARQn signal is acknowledged until the rising edge of the IRAMEN signal for transfer from RAM to VSB (n = 3 to 0): 3.5 clocks
- ★ • Access to RAM connected to VDB: 1 clock

In the case of external memory access, these depend on the connected MEMC and the external memory. An example is shown below.

Example When SRAM is accessed using the MEMC (NT85E500)

Transfer Type	Conditions	Transfer Mode	Minimum Clock Number												
Two-cycle	<ul style="list-style-type: none"> • Time between start of read cycle and end of write cycle • The transfer time of one transfer for single and single-step transfers, and four transfers for a line transfer. • The combinations of transfer sources and destinations are as follows. <Transfer source> <Transfer destination> <table style="margin-left: 20px; border: none;"> <tr><td>VSB</td><td>→</td><td>VSB</td></tr> <tr><td>VSB</td><td>→</td><td>RAM</td></tr> <tr><td>RAM</td><td>→</td><td>VSB</td></tr> <tr><td>RAM</td><td>→</td><td>RAM</td></tr> </table> 	VSB	→	VSB	VSB	→	RAM	RAM	→	VSB	RAM	→	RAM	Single	5 clocks
		VSB	→	VSB											
		VSB	→	RAM											
	RAM	→	VSB												
	RAM	→	RAM												
	Single-step	5 clocks													
Line	32 clocks														
Time in which bus is released to CPU	Single	6 clocks													
	Single-step	4 clocks													
	Line	6 clocks													
Flyby	Transfer time of one transfer from SRAM to I/O, and from I/O to SRAM	–	3 clocks												

(4) CPU access during DMA transfer

The CPU can access external memory, peripheral macros, or RAM for which no DMA transfer is being performed.

- ★ The DMAC has a higher VSB bus access right priority than the CPU, so the access from the CPU to the VSB generated during the DMA transfer must wait until the DMA transfer is complete and the bus is available for the CPU. However, while DMA transfer is being performed between the external memory and peripheral macro, the CPU can access the RAM. Also, the CPU can access the external memory and peripheral macro using the VSB when DMA transfer is being performed between RAMs that are directly connected to the VDB.

(5) DMA transfer end interrupt

The DMA transfer end interrupt is not generated when DMA transfer is complete. If the generation of an interrupt coinciding with the completion of transfer is required, input the DMATCO_n signal to the INT_m pin and perform maskable interrupt servicing (n = 3 to 0, m = 63 to 0).

(6) DMARQ_n signal retention

The DMARQ_n signal must retain the request until the DMACTV_n signal becomes active.

If the DMARQ_n signal is made inactive before the DMACTV_n signal becomes active, DMA transfer may not be executed (n = 3 to 0).

(7) VMLOCK signal

If the destination of the DMA transfer is the RAM connected to the VDB, the VMLOCK signal will not become active in any transfer mode (single transfer, etc.) (the VMLOCK signal becomes active only when two or more VSB cycles are generated during 2-cycle transfer).

If the destination of the DMA transfer is the VSB, the VMLOCK signal stays active in the line transfer mode, in either 2-cycle transfer or flyby transfer, until the fourth DMA transfer is performed. Therefore, during a DMA line transfer in which the destination is the VSB, the VSB is locked until one line transfer is complete and the bus is retained. During 2-cycle single transfer, single-step transfer, or block transfer, the VMLOCK signal becomes inactive for each DMA transfer, therefore other VSB requests (VAREQ), such as SDRAM refresh, that have a higher priority are acknowledged and the bus can be released. During block transfer, the VSB bus access right is relinquished in the middle of transfer, and the remaining transfer is executed when the bus access right becomes available again.

CHAPTER 8 INTC

The interrupt control unit (INTC), which can process interrupt requests generated for a total of 67 sources, processes various types of interrupt requests from external sources. In addition, exception processing can be started (exception trap) due to a TRAP instruction (software exception) or due to the generation of an exception event (fetching of an illegal opcode).

An interrupt is an event that is generated independently of program execution, and an exception is an event that is generated dependent on program execution. Generally, the processing of an exception takes precedence over the processing of an interrupt.

8.1 Features

- Interrupt
 - Non-maskable interrupt: 3 sources
 - Maskable interrupt: 64 sources
 - 8 levels programmable priorities (maskable interrupt)
 - Multiple interrupt control according to priority
 - Mask specification to each maskable interrupt request
- Exception
 - Software exception: 32 sources
 - Exception trap: 1 source (illegal opcode exception)

These interrupt/exception sources are listed in Table 8-1.

Table 8-1. Interrupt/Exception List (1/3)

Type	Classifi- cation	Interrupt/Exception Source			Default Priority	Exception Code	Handler Address	Restored PC
		Name	Control Register	Generating Source				
Reset	Interrupt	RESET	–	DCRESZ input	–	0000H	00000000H	Undefined
Non-maskable	Interrupt	NMI0	–	DCNMI0 input	–	0010H	00000010H	nextPC
	Interrupt	NMI1	–	DCNMI1 input	–	0020H	00000020H	nextPC
	Interrupt	NMI2	–	DCNMI2 input	–	0030H	00000030H	nextPC
Software exception	Exception	TRAP0n ^{Note}	–	TRAP instruction	–	004nH	00000040H	nextPC
	Exception	TRAP1n ^{Note}	–	TRAP instruction	–	005nH	00000050H	nextPC
Exception trap	Exception	ILGOP	–	Illegal opcode	–	0060H	00000060H	nextPC
Maskable	Interrupt	INT0	PIC0	INT0 input	0	0080H	00000080H	nextPC
	Interrupt	INT1	PIC1	INT1 input	1	0090H	00000090H	nextPC
	Interrupt	INT2	PIC2	INT2 input	2	00A0H	000000A0H	nextPC
	Interrupt	INT3	PIC3	INT3 input	3	00B0H	000000B0H	nextPC
	Interrupt	INT4	PIC4	INT4 input	4	00C0H	000000C0H	nextPC
	Interrupt	INT5	PIC5	INT5 input	5	00D0H	000000D0H	nextPC
	Interrupt	INT6	PIC6	INT6 input	6	00E0H	000000E0H	nextPC

Note n: value of 0 to FH

Table 8-1. Interrupt/Exception List (2/3)

Type	Classification	Interrupt/Exception Source			Default Priority	Exception Code	Handler Address	Restored PC
		Name	Control Register	Generating Source				
Maskable	Interrupt	INT7	PIC7	INT7 input	7	00F0H	00000F0H	nextPC
	Interrupt	INT8	PIC8	INT8 input	8	0100H	00000100H	nextPC
	Interrupt	INT9	PIC9	INT9 input	9	0110H	00000110H	nextPC
	Interrupt	INT10	PIC10	INT10 input	10	0120H	00000120H	nextPC
	Interrupt	INT11	PIC11	INT11 input	11	0130H	00000130H	nextPC
	Interrupt	INT12	PIC12	INT12 input	12	0140H	00000140H	nextPC
	Interrupt	INT13	PIC13	INT13 input	13	0150H	00000150H	nextPC
	Interrupt	INT14	PIC14	INT14 input	14	0160H	00000160H	nextPC
	Interrupt	INT15	PIC15	INT15 input	15	0170H	00000170H	nextPC
	Interrupt	INT16	PIC16	INT16 input	16	0180H	00000180H	nextPC
	Interrupt	INT17	PIC17	INT17 input	17	0190H	00000190H	nextPC
	Interrupt	INT18	PIC18	INT18 input	18	01A0H	000001A0H	nextPC
	Interrupt	INT19	PIC19	INT19 input	19	01B0H	000001B0H	nextPC
	Interrupt	INT20	PIC20	INT20 input	20	01C0H	000001C0H	nextPC
	Interrupt	INT21	PIC21	INT21 input	21	01D0H	000001D0H	nextPC
	Interrupt	INT22	PIC22	INT22 input	22	01E0H	000001E0H	nextPC
	Interrupt	INT23	PIC23	INT23 input	23	01F0H	000001F0H	nextPC
	Interrupt	INT24	PIC24	INT24 input	24	0200H	00000200H	nextPC
	Interrupt	INT25	PIC25	INT25 input	25	0210H	00000210H	nextPC
	Interrupt	INT26	PIC26	INT26 input	26	0220H	00000220H	nextPC
	Interrupt	INT27	PIC27	INT27 input	27	0230H	00000230H	nextPC
	Interrupt	INT28	PIC28	INT28 input	28	0240H	00000240H	nextPC
	Interrupt	INT29	PIC29	INT29 input	29	0250H	00000250H	nextPC
	Interrupt	INT30	PIC30	INT30 input	30	0260H	00000260H	nextPC
	Interrupt	INT31	PIC31	INT31 input	31	0270H	00000270H	nextPC
	Interrupt	INT32	PIC32	INT32 input	32	0280H	00000280H	nextPC
	Interrupt	INT33	PIC33	INT33 input	33	0290H	00000290H	nextPC
	Interrupt	INT34	PIC34	INT34 input	34	02A0H	000002A0H	nextPC
	Interrupt	INT35	PIC35	INT35 input	35	02B0H	000002B0H	nextPC
	Interrupt	INT36	PIC36	INT36 input	36	02C0H	000002C0H	nextPC
	Interrupt	INT37	PIC37	INT37 input	37	02D0H	000002D0H	nextPC
	Interrupt	INT38	PIC38	INT38 input	38	02E0H	000002E0H	nextPC
	Interrupt	INT39	PIC39	INT39 input	39	02F0H	000002F0H	nextPC
	Interrupt	INT40	PIC40	INT40 input	40	0300H	00000300H	nextPC
	Interrupt	INT41	PIC41	INT41 input	41	0310H	00000310H	nextPC
	Interrupt	INT42	PIC42	INT42 input	42	0320H	00000320H	nextPC
	Interrupt	INT43	PIC43	INT43 input	43	0330H	00000330H	nextPC

Table 8-1. Interrupt/Exception List (3/3)

Type	Classification	Interrupt/Exception Source			Default Priority	Exception Code	Handler Address	Restored PC
		Name	Control Register	Generating Source				
Maskable	Interrupt	INT44	PIC44	INT44 input	44	0340H	00000340H	nextPC
	Interrupt	INT45	PIC45	INT45 input	45	0350H	00000350H	nextPC
	Interrupt	INT46	PIC46	INT46 input	46	0360H	00000360H	nextPC
	Interrupt	INT47	PIC47	INT47 input	47	0370H	00000370H	nextPC
	Interrupt	INT48	PIC48	INT48 input	48	0380H	00000380H	nextPC
	Interrupt	INT49	PIC49	INT49 input	49	0390H	00000390H	nextPC
	Interrupt	INT50	PIC50	INT50 input	50	03A0H	000003A0H	nextPC
	Interrupt	INT51	PIC51	INT51 input	51	03B0H	000003B0H	nextPC
	Interrupt	INT52	PIC52	INT52 input	52	03C0H	000003C0H	nextPC
	Interrupt	INT53	PIC53	INT53 input	53	03D0H	000003D0H	nextPC
	Interrupt	INT54	PIC54	INT54 input	54	03E0H	000003E0H	nextPC
	Interrupt	INT55	PIC55	INT55 input	55	03F0H	000003F0H	nextPC
	Interrupt	INT56	PIC56	INT56 input	56	0400H	00000400H	nextPC
	Interrupt	INT57	PIC57	INT57 input	57	0410H	00000410H	nextPC
	Interrupt	INT58	PIC58	INT58 input	58	0420H	00000420H	nextPC
	Interrupt	INT59	PIC59	INT59 input	59	0430H	00000430H	nextPC
	Interrupt	INT60	PIC60	INT60 input	60	0440H	00000440H	nextPC
	Interrupt	INT61	PIC61	INT61 input	61	0450H	00000450H	nextPC
	Interrupt	INT62	PIC62	INT62 input	62	0460H	00000460H	nextPC
	Interrupt	INT63	PIC63	INT63 input	63	0470H	00000470H	nextPC

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

Restored PC: This is the PC value saved in EIPC or FEPC upon activation of interrupt servicing or exception processing. Note, however, that the restored PC when a non-maskable or maskable interrupt is acknowledged while one of the following instructions is being executed does not become the nextPC (if an interrupt is acknowledged during interrupt execution, execution stops, and then resumes after the interrupt servicing has finished).

- Load instructions (SLD.B, SLD.BU, SLD.H, SLD.HU, SLD.W)
- Division instructions (DIV, DIVH, DIVU, DIVHU)
- PREPARE, DISPOSE instructions (only if an interrupt is generated before the stack pointer is updated)

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

- 2.** The execution address of the illegal instruction when an illegal opcode exception occurs is calculated as follows: (Restored PC – 4)

8.2 Non-Maskable Interrupts (NMI)

A non-maskable interrupt request (NMI) is acknowledged unconditionally even if the NU85E is in an interrupt disabled (DI) state.

A non-maskable interrupt request is generated according to DCNMIn pin input ($n = 2$ to 0). When a rising edge is input to the DCNMIn pin, a non-maskable interrupt (NMI n) is generated.

If multiple non-maskable interrupts are generated at the same time, the highest priority servicing is executed according to the following priority order (the lower priority interrupts are ignored).

NMI2 > NMI1 > NMI0

Note that if an NMI0, NMI1, or NMI2 request is generated while NMI0 is being serviced, the servicing is executed as follows.

(1) If an NMI0 request is generated while NMI0 is being serviced

The new NMI0 request is held pending regardless of the value of the PSW's NP bit. The pending NMI0 request is acknowledged after servicing of the current NMI0 request has finished (after execution of the RETI instruction).

(2) If an NMI1 request is generated while NMI0 is being serviced

If the PSW's NP bit remains set (1) while NMI0 is being serviced, the new NMI1 request is held pending. The pending NMI1 request is acknowledged after servicing of the current NMI0 request has finished (after execution of the RETI instruction).

If the PSW's NP bit is cleared (0) while NMI0 is being serviced, the newly generated NMI1 request is executed (NMI0 servicing is halted).

(3) If an NMI2 request is generated while NMI0 is being serviced

The new NMI2 request is executed, regardless of the value of the PSW's NP bit (NMI0 servicing is halted).

- ★ **Cautions**
1. **When a non-maskable interrupt request (NMI) is generated, the values of the PC and PSW are saved in the registers (FEPC and FEPSW) for saving the status when an NMI occurs, but in this case, only NMI0 can be normally restored by the RETI instruction. Even if NMI1 and NMI2, which assume an emergency use such as watchdog, are restored by the RETI instruction, the INTC cannot determine the priority of the following interrupts. Therefore, when NMI1 or NMI2, and other maskable interrupts are input with a miniscule time lag, maskable interrupt requests other than NMI1 and NMI2 may be deleted.**
When NMI1 or NMI2 is generated while NMI0 is being serviced, FEPC is overwritten. When NMI0 servicing has been restored after the NMI1 and NMI2 servicing, the main routine cannot successfully be restored from the NMI0 servicing and an endless loop occurs. In the case of NMI2, a newly generated NMI2 request is executed regardless of the value of the NP bit in the PSW.
Therefore, NMI1 and NMI2 cannot be restored.
 2. **If interrupt servicing by NMI1 or NMI2 is continued without the RETI instruction being executed, hang up will not occur, but none of the following interrupt requests will be acknowledged because multiple interrupts are disabled.**
- ★

Figure 8-1. Example of Non-Maskable Interrupt Request Acknowledgement Operation (1/2)

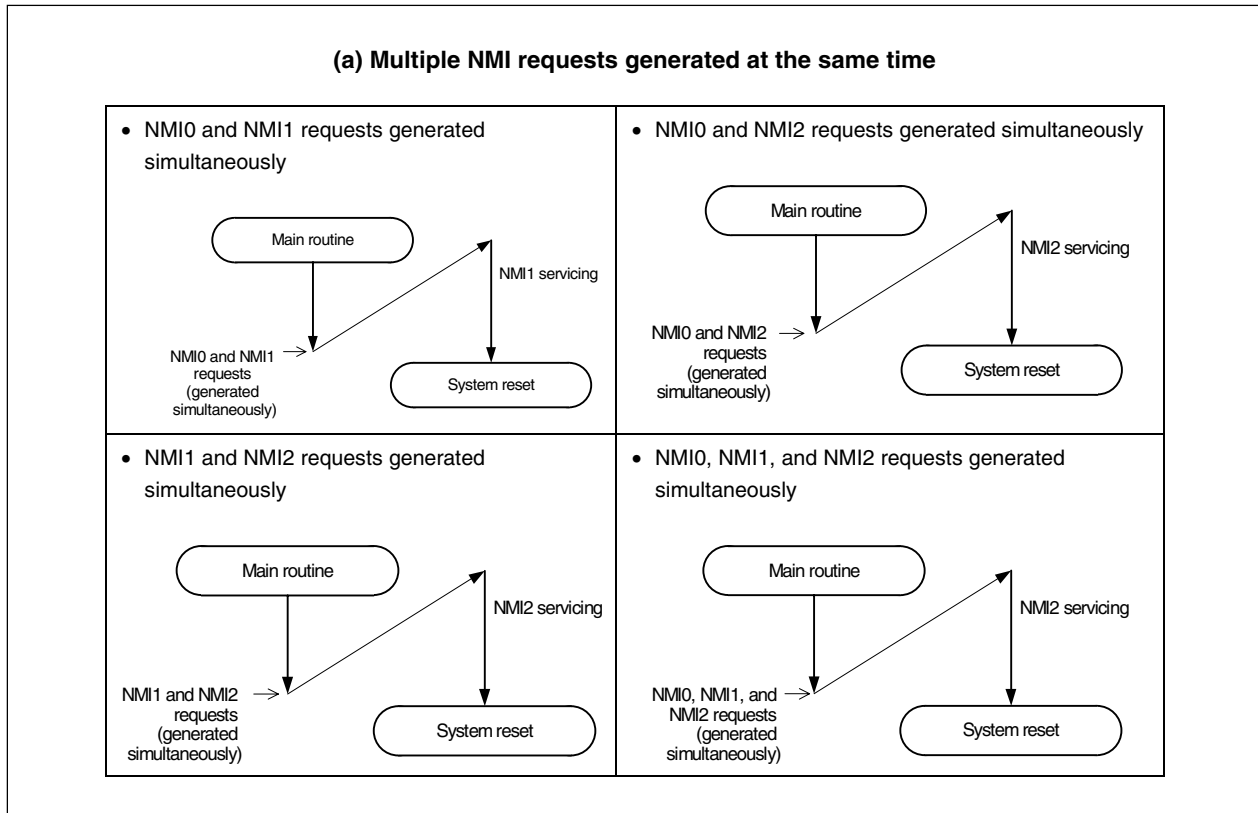
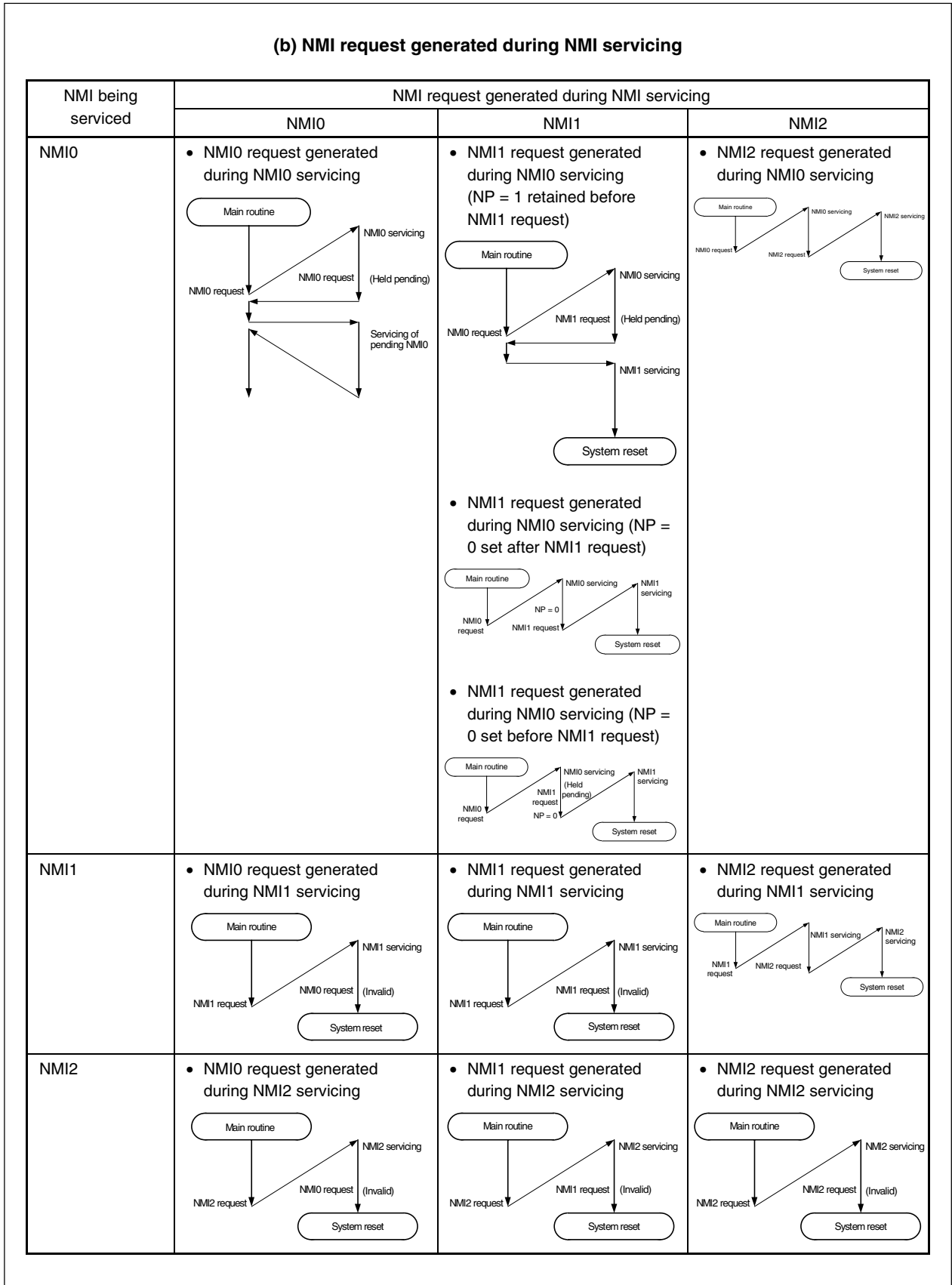


Figure 8-1. Example of Non-Maskable Interrupt Request Acknowledgement Operation (2/2)



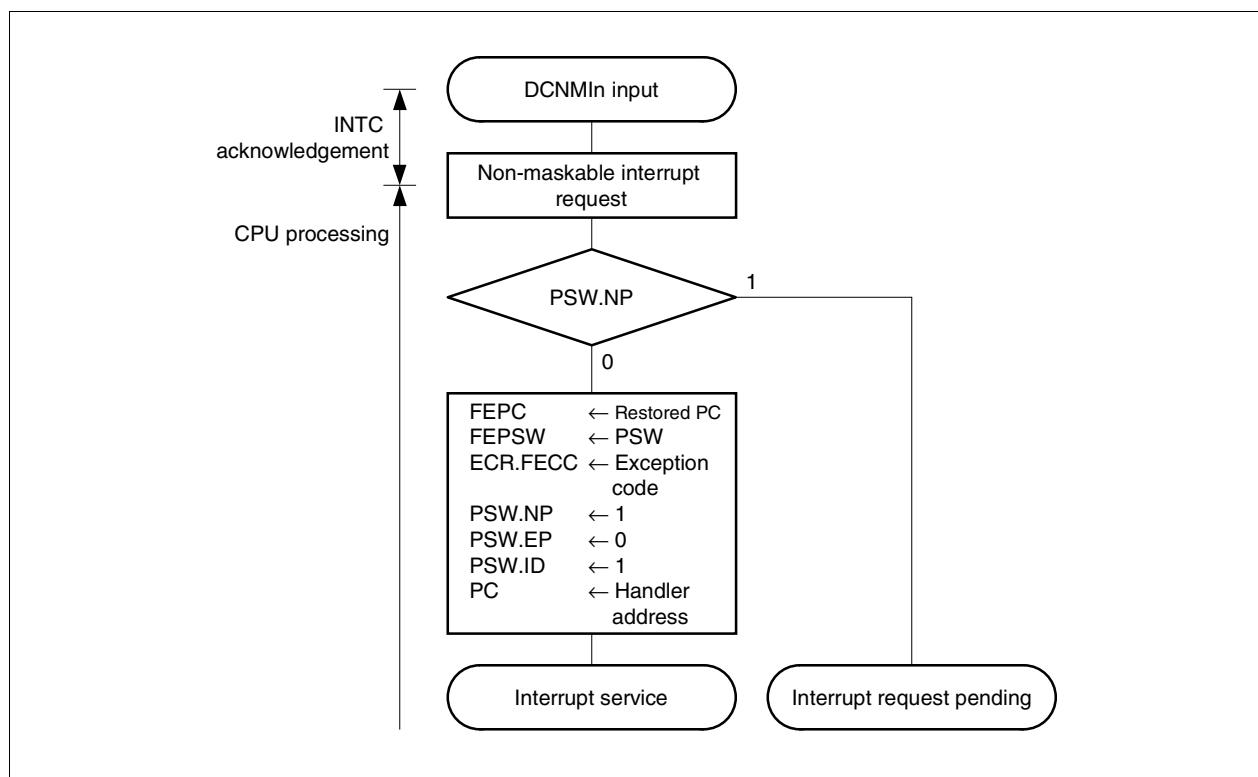
8.2.1 Operation

If a non-maskable interrupt is generated according to DCNMIn input, the CPU performs the following processing and shifts control to the handler routine ($n = 2$ to 0).

- <1> Save the restored PC in the FEPC.
- <2> Save the current PSW in the FEPSW.
- <3> Write the exception code in the higher halfword (FECC) of the ECR.
- <4> Set the NP and ID bits of the PSW and clear the EP bit.
- <5> Set the handler address for the non-maskable interrupt in the PC and shift control.

Figure 8-2 shows the processing format of non-maskable interrupt service.

Figure 8-2. Non-Maskable Interrupt Processing Format



8.2.2 Restore

(1) NMIO

Control is returned from NMIO servicing according to the RETI instruction.

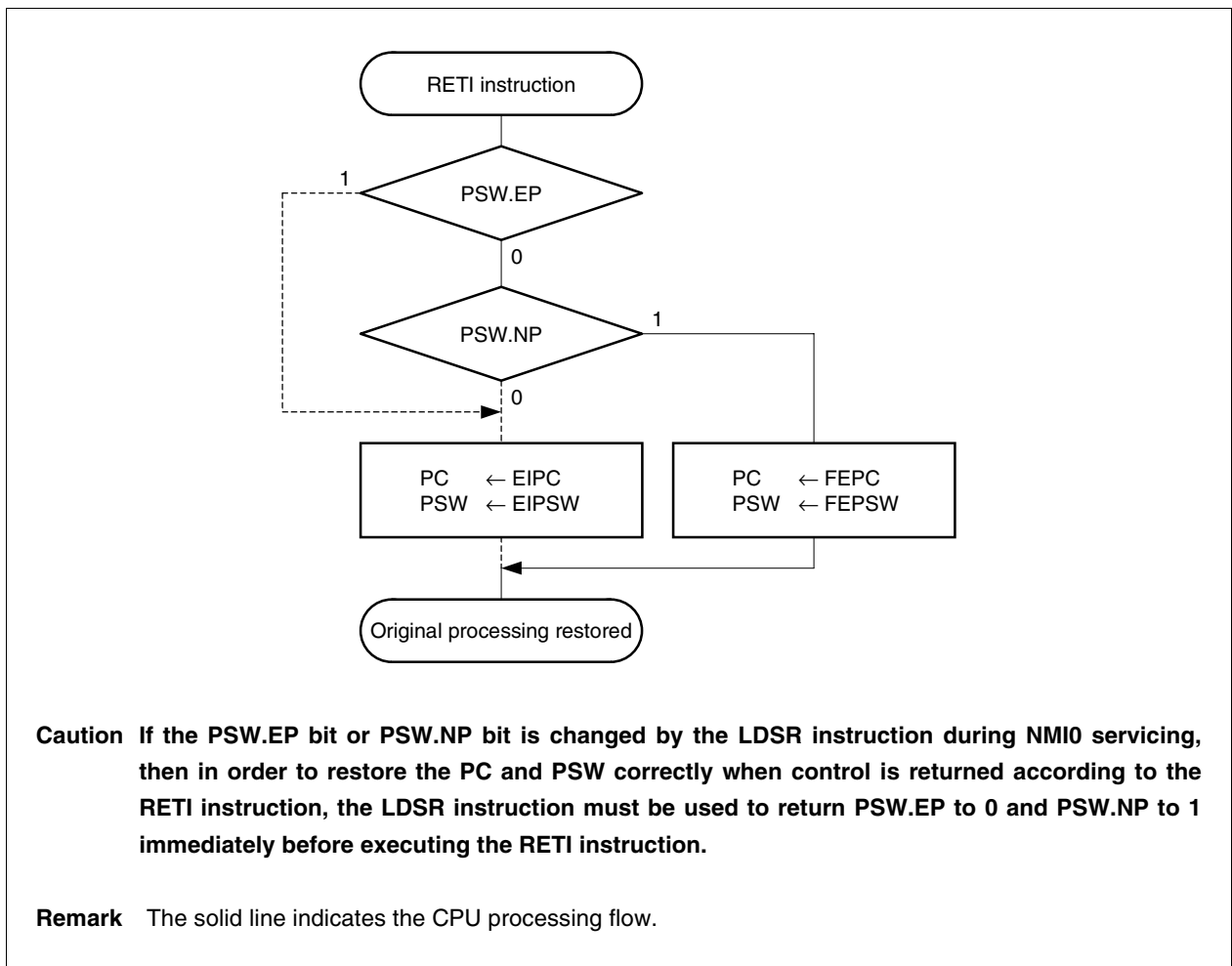
When the RETI instruction is executed, the CPU performs the following processing and shifts control to the restored PC address.

<1> Since the EP bit of the PSW is 0 and the NP bit is 1, fetch the restored PC and PSW from the FEPC and FEPSW.

<2> Shift control to the fetched restored PC address and PSW status.

Figure 8-3 shows the processing format of the RETI instruction.

Figure 8-3. RETI Instruction Processing Format



(2) NMI1, NMI2

Restoring by RETI instruction is not possible. Perform a system reset according to DCRESZ input after interrupt servicing.

8.3 Maskable Interrupts

A maskable interrupt request is an interrupt request for which the acknowledgement of the interrupt can be masked according to the interrupt control register. There are 64 interrupt sources for maskable interrupts.

A maskable interrupt request is generated according to INTn pin input (n = 63 to 0). When a rising edge is input to the INTn pin, a maskable interrupt (INTn) is generated.

If multiple maskable interrupt requests are generated at the same time, their priorities are determined according to the default priorities. In addition to the default priority, eight interrupt priority levels can be set according to the interrupt control register (programmable priority control).

When an interrupt request is acknowledged, interrupt disabled (DI) state is set, and the acknowledgement of subsequent maskable interrupt requests is disabled.

If the EI instruction is executed during an interrupt service routine, interrupt enabled (EI) state is set, and the acknowledgement of interrupt requests having higher priorities than the priority level of the currently acknowledged interrupt request (specified by the interrupt control register) is enabled. Interrupts having the same priority level cannot be nested.

However, the following processing is required for multiple interrupt service.

- <1> Save the EIPC and EIPSW in memory or general-purpose registers before executing the EI instruction.
- <2> Before executing the RETI instruction, execute the DI instruction and return the values that were saved in step <1> to the EIPC and EIPSW.

8.3.1 Operation

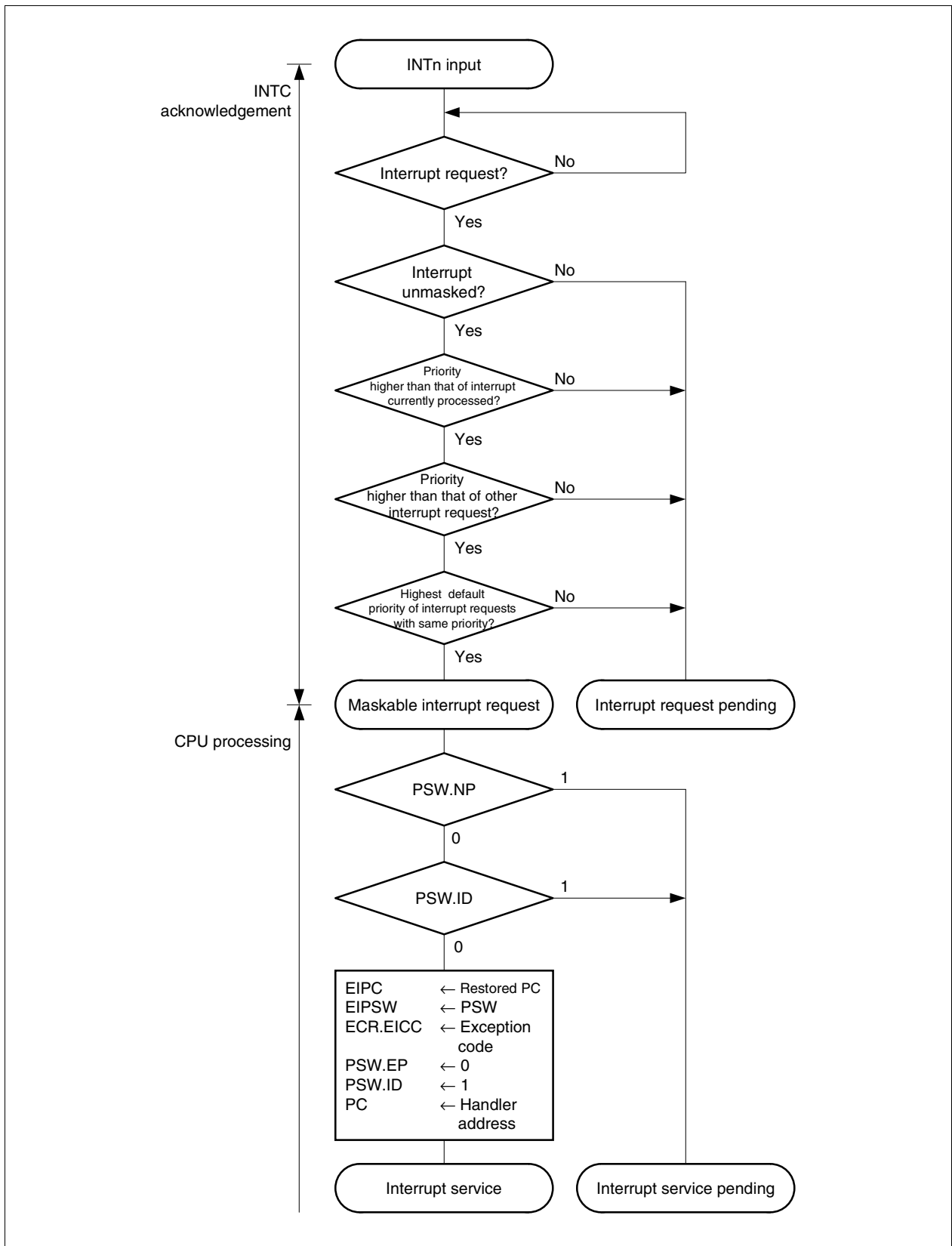
If a maskable interrupt is generated according to INTn input, the CPU performs the following processing and shifts control to the handler routine.

- <1> Save the restored PC in the EIPC.
- <2> Save the current PSW in the EIPSW.
- <3> Write the exception code in the lower halfword (EICC) of the ECR.
- <4> Set the ID bit of the PSW and clear the EP bit.
- <5> Set the handler address for the interrupt in the PC and shift control.

An INTn input that is masked by the INTC and an INTn input that was generated while another interrupt was being serviced (PSW.NP = 1 or PSW.ID = 1) are kept pending within the INTC. In this case, if the mask is canceled or the RETI and LDSR instructions are used to set PSW.NP to 0 and PSW.ID to 0, the new maskable interrupt service is started according to the INTn input that had been pending.

Figure 8-4 shows the processing format of maskable interrupt service.

Figure 8-4. Maskable Interrupt Processing Format



8.3.2 Restore

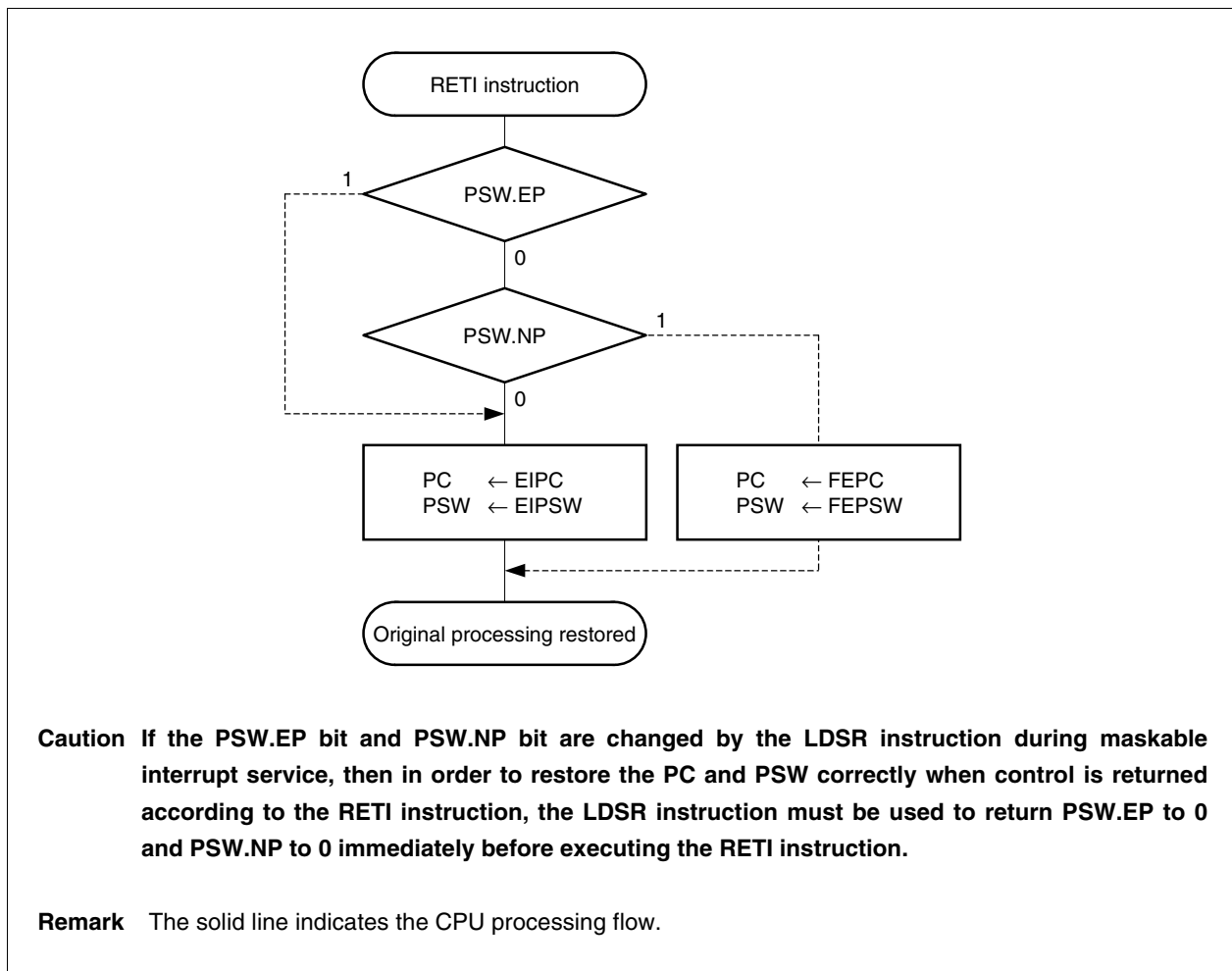
Control is returned from maskable interrupt service according to the RETI instruction.

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

- <1> Since the EP bit of the PSW is 0 and the NP bit is 0, fetch the restored PC and PSW from the EIPC and EIPSW.
- <2> Shift control to the fetched restored PC address and PSW status.

Figure 8-5 shows the processing format of the RETI instruction.

Figure 8-5. RETI Instruction Processing Format



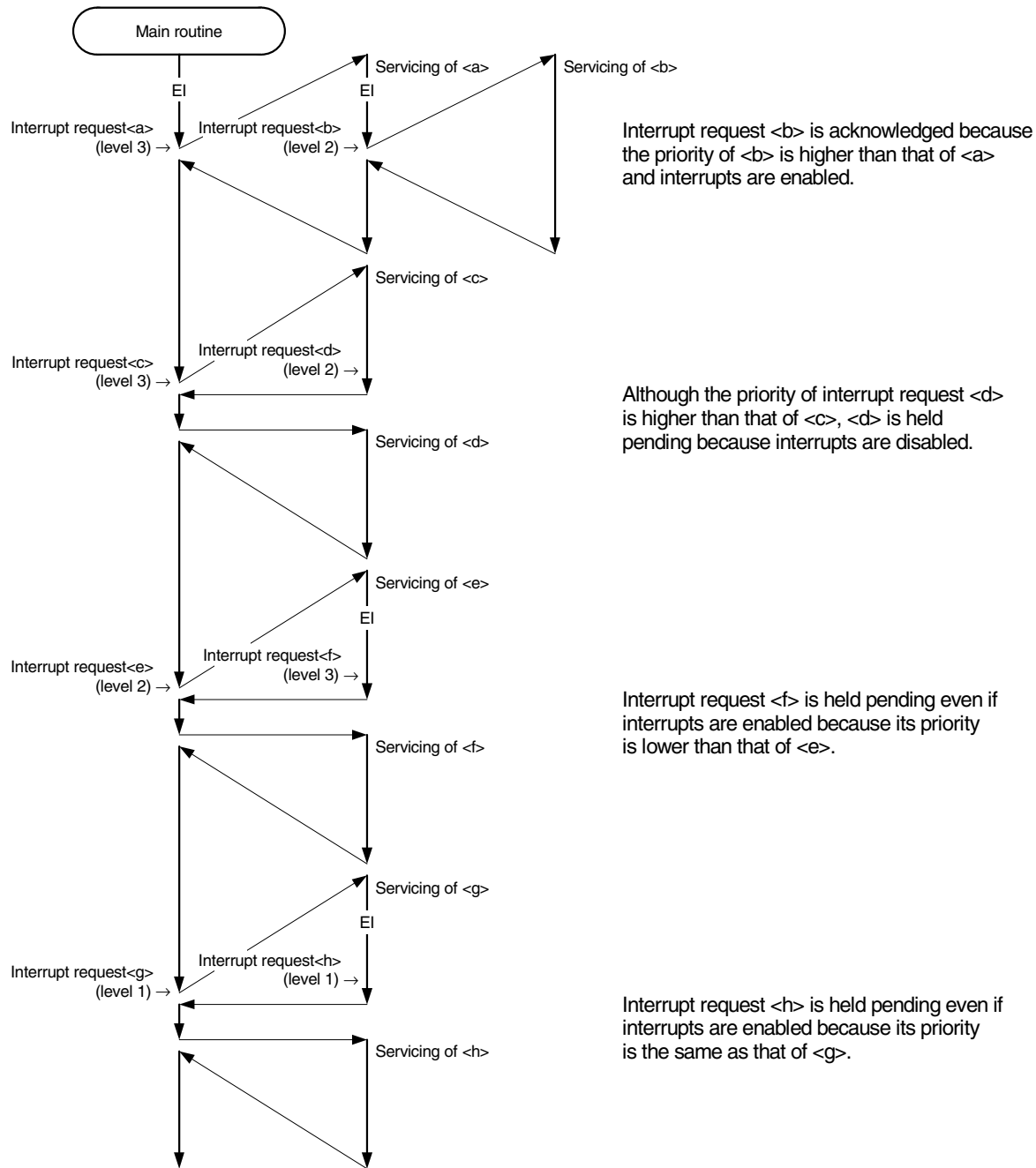
8.3.3 Maskable interrupt priorities

The INTC provides multiple interrupt service that acknowledges another interrupt while an interrupt is being serviced. Multiple interrupts can be controlled according to priorities.

Priority control includes control according to default priorities and programmable priority control according to the interrupt control register (PICn). For priority control according to default priorities, if multiple interrupts having the same priority level according to the PICn register are generated at the same time, the interrupts are serviced according to the priorities (default priorities) that have been assigned in advance to each interrupt request (see **Table 8-1 Interrupt/Exception List**). For programmable priority control, the interrupt requests are divided into eight levels according to PICn register settings.

When an interrupt is acknowledged, the ID flag of the PSW is automatically set (1). Therefore, to use multiple interrupt service, clear (0) the ID flag (such as by executing the EI instruction during the interrupt service program) to set the interrupt enable state.

Figure 8-6. Servicing Example in Which Another Interrupt Request Is Issued During Interrupt Servicing (1/2)



- Remarks**
1. <a> to <u> in the figure represent dummy names that are assigned to distinguish between the interrupt requests.
 2. Higher or lower default priorities mentioned in the figure indicate relative priorities between two interrupt requests.

Caution To use multiple interrupt servicing, the contents of the EIPC and EIPSW registers must be saved.

Figure 8-6. Servicing Example in Which Another Interrupt Request Is Issued During Interrupt Servicing (2/2)

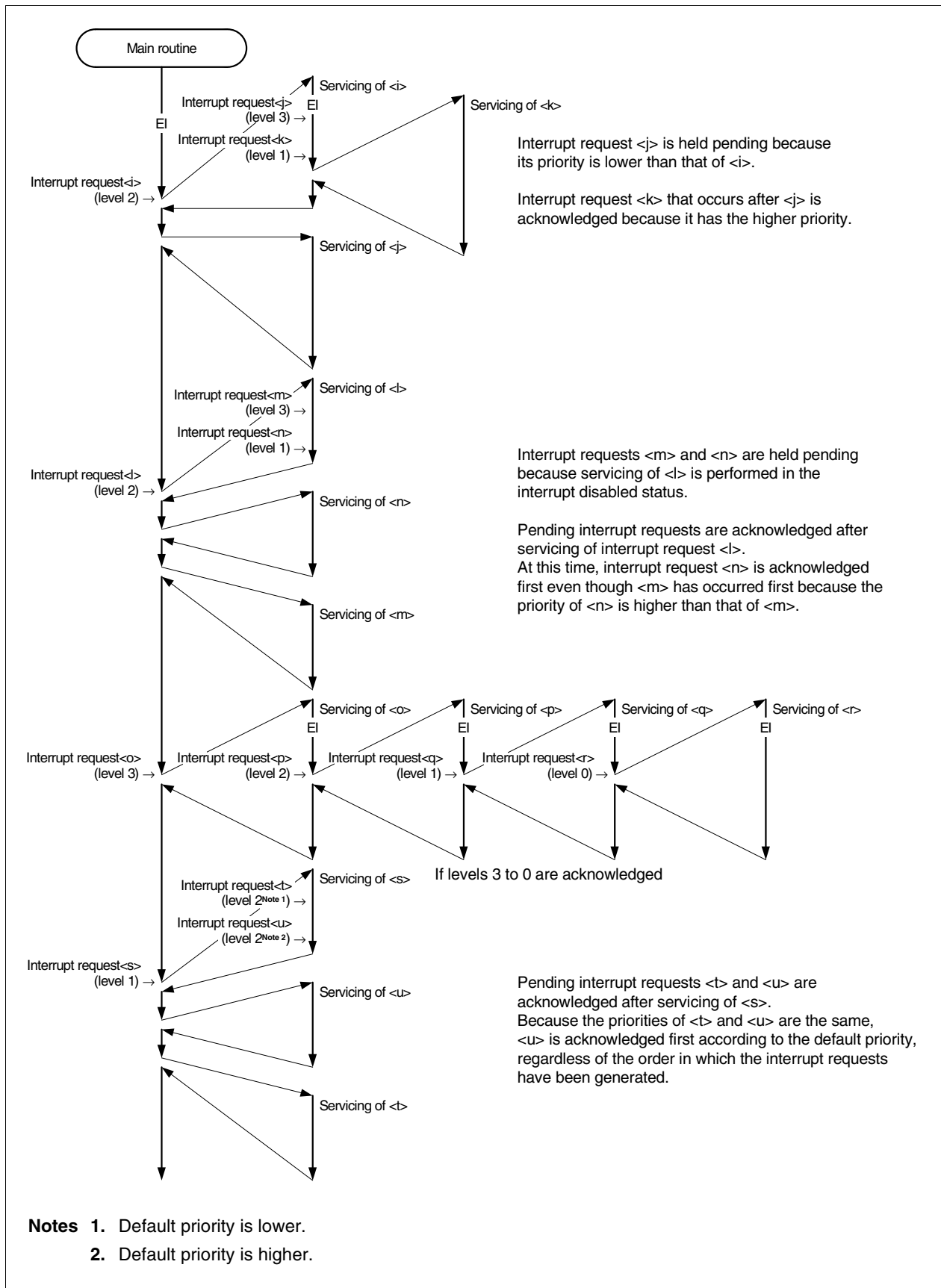
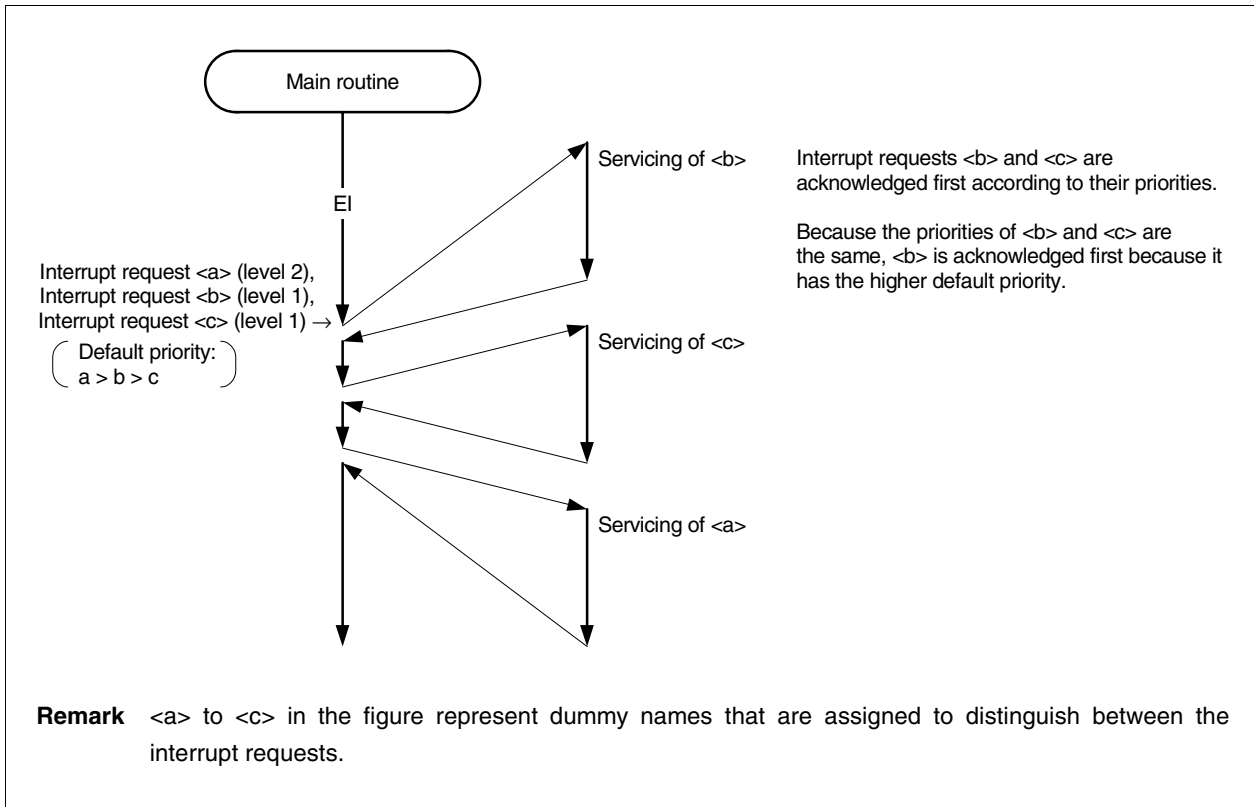


Figure 8-7. Servicing Example for Simultaneously Issued Interrupt Requests



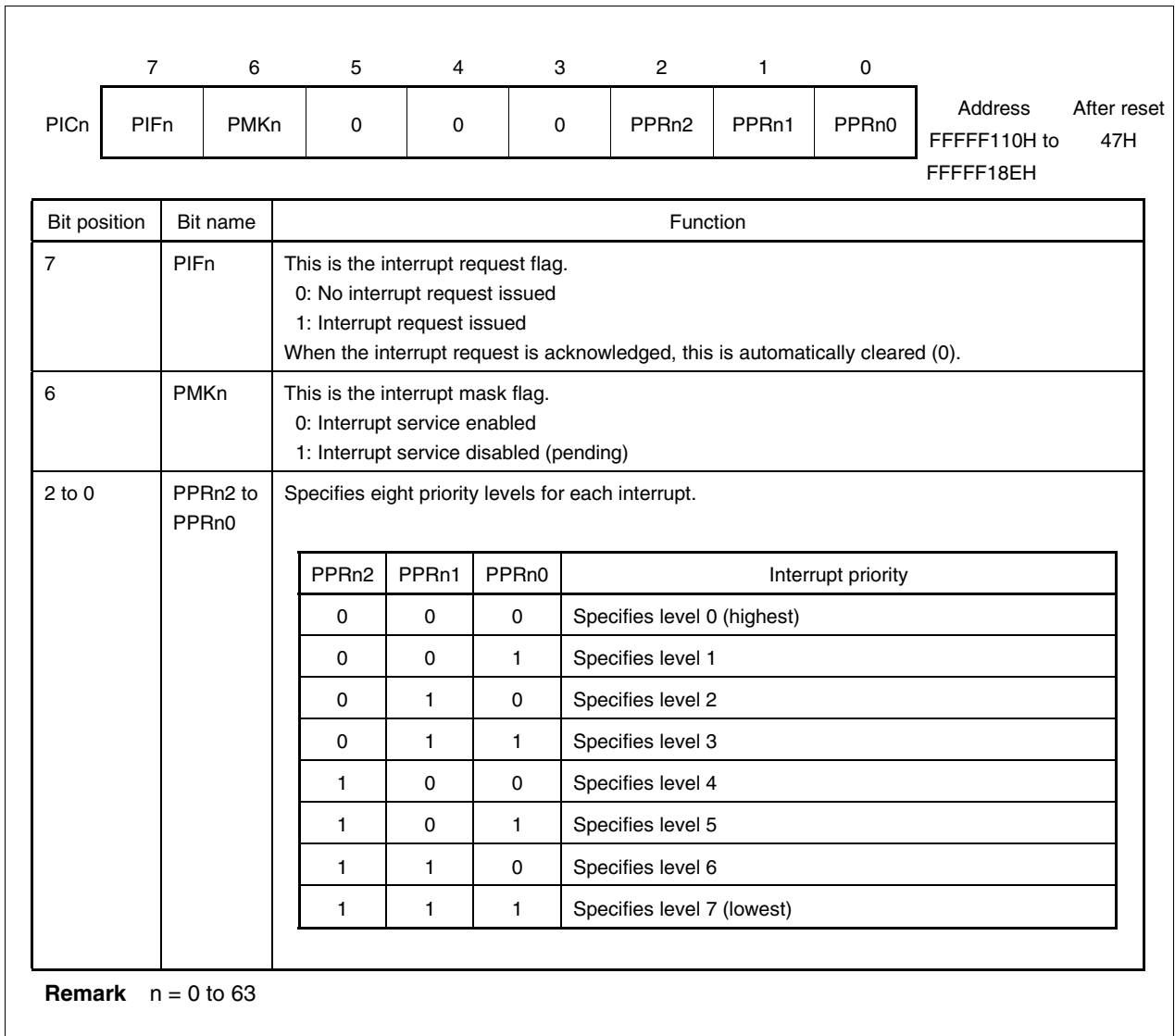
8.3.4 Control registers

(1) Interrupt control registers 0 to 63 (PIC0 to PIC63)

The interrupt control registers, which are assigned to each interrupt request (maskable interrupt), set control conditions for each interrupt.

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

Figure 8-8. Interrupt Control Registers 0 to 63 (PIC0 to PIC63)



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

The interrupt mask registers maintain the mask status of each maskable interrupt.

The PMKn bit of this register and the PMKn bit of the PICn register are linked (n = 0 to 63).

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

When using the higher 8 bits of the IMRm register as the IMRmH register, and the lower 8 bits as the IMRmL register, the IMRm register can be read or written in 8-bit or 1-bit units.

Figure 8-9. Interrupt Mask Registers 0 to 3 (IMR0 to IMR3)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
IMR0	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	Address	After reset
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	FFFFFF100H	FFFFH
IMR1	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	Address	After reset
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	FFFFFF102H	FFFFH
IMR2	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	Address	After reset
	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	FFFFFF104H	FFFFH
IMR3	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	PMK	Address	After reset
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	FFFFFF106H	FFFFH

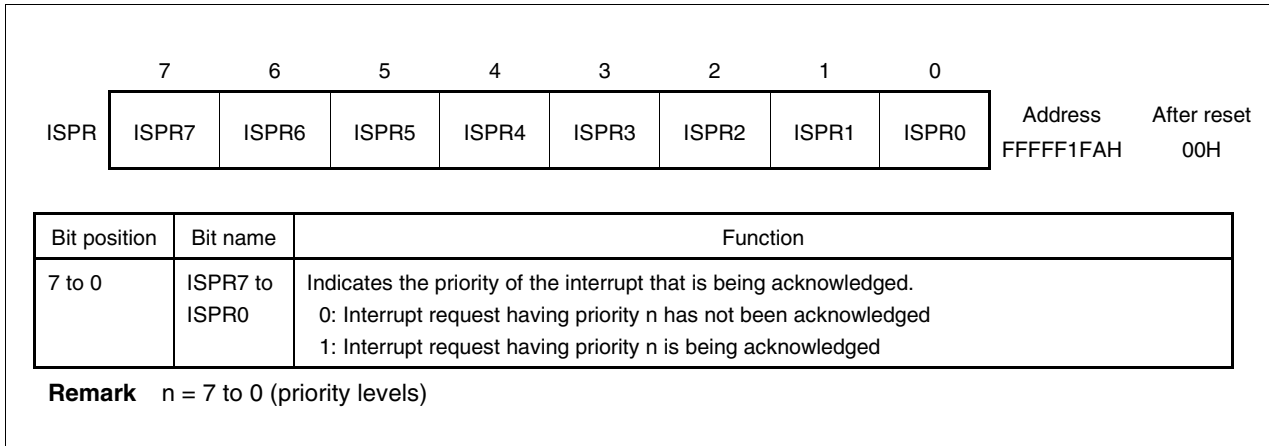
(3) In-service priority register (ISPR)

This register maintains the priority level of the maskable interrupt that is being acknowledged. When an interrupt request is acknowledged, the bit corresponding to the priority level of that interrupt request is set (1) and maintained while the interrupt is being serviced.

When the RETI instruction is executed, the bit corresponding to the interrupt request having the highest priority among the bits that are set (1) within the ISPR register is automatically cleared (0). However, it is not cleared (0) when control returns from non-maskable interrupt service or exception processing.

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

Figure 8-10. In-Service Priority Register (ISPR)

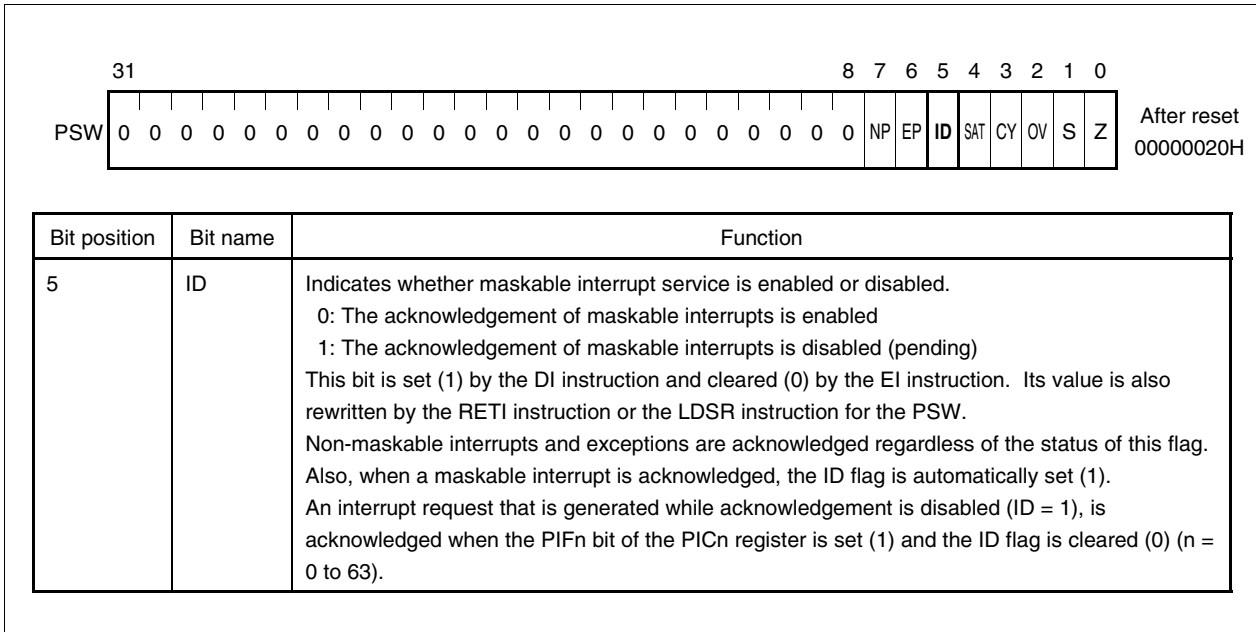


8.3.5 Maskable interrupt status flag (ID)

This flag, which controls the operation status of maskable interrupts, stores information indicating whether the acknowledgement of interrupt requests is enabled or disabled.

It is assigned to bit 5 of the program status word (PSW).

Figure 8-11. Program Status Word (PSW)



8.4 Software Exception

A software exception, which is an exception that is generated when the CPU executes the TRAP instruction, can always be acknowledged.

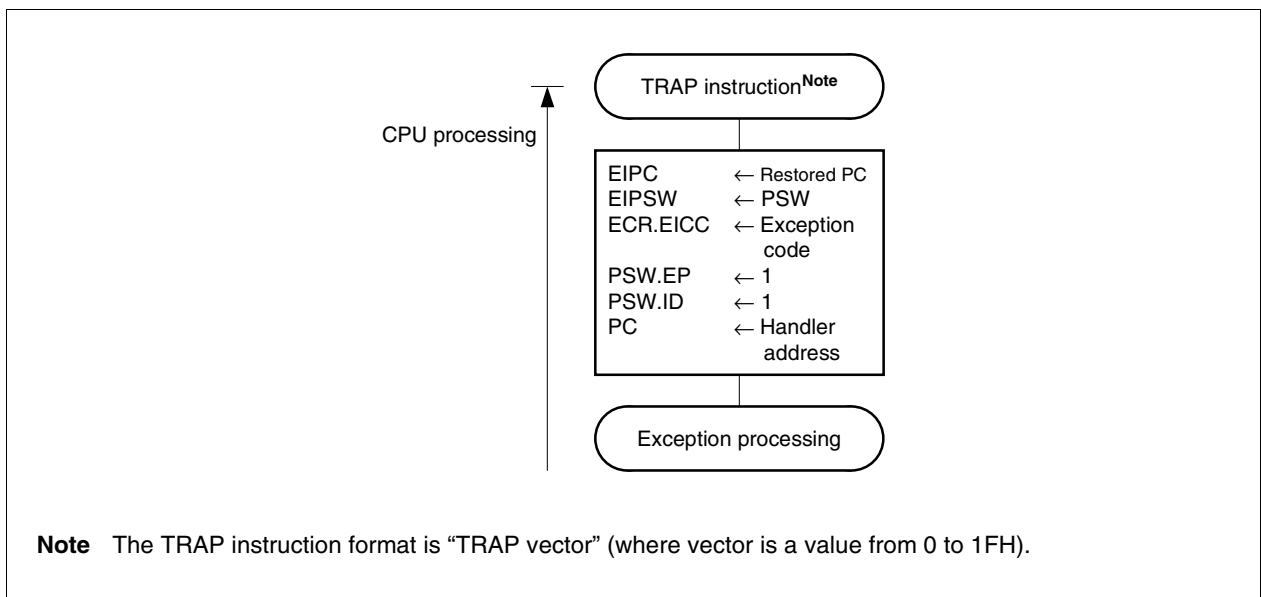
8.4.1 Operation

If a software exception is generated, the CPU performs the following processing and shifts control to the handler routine.

- <1> Save the restored PC in the EIPC.
- <2> Save the current PSW in the EIPSW.
- <3> Write the exception code in the lower 16 bits (EICC) of the ECR (interrupt source).
- <4> Set the EP and ID bits of the PSW.
- <5> Set the handler address (00000040H or 00000050H) for the software exception in the PC and shift control.

Figure 8-12 shows the processing format of software exception processing.

Figure 8-12. Software Exception Processing Format



The handler address is determined by the TRAP instruction operand (vector). When vector is 0 to 0FH, the address is 00000040H. When vector is 10H to 1FH, the address is 00000050H.

8.4.2 Restore

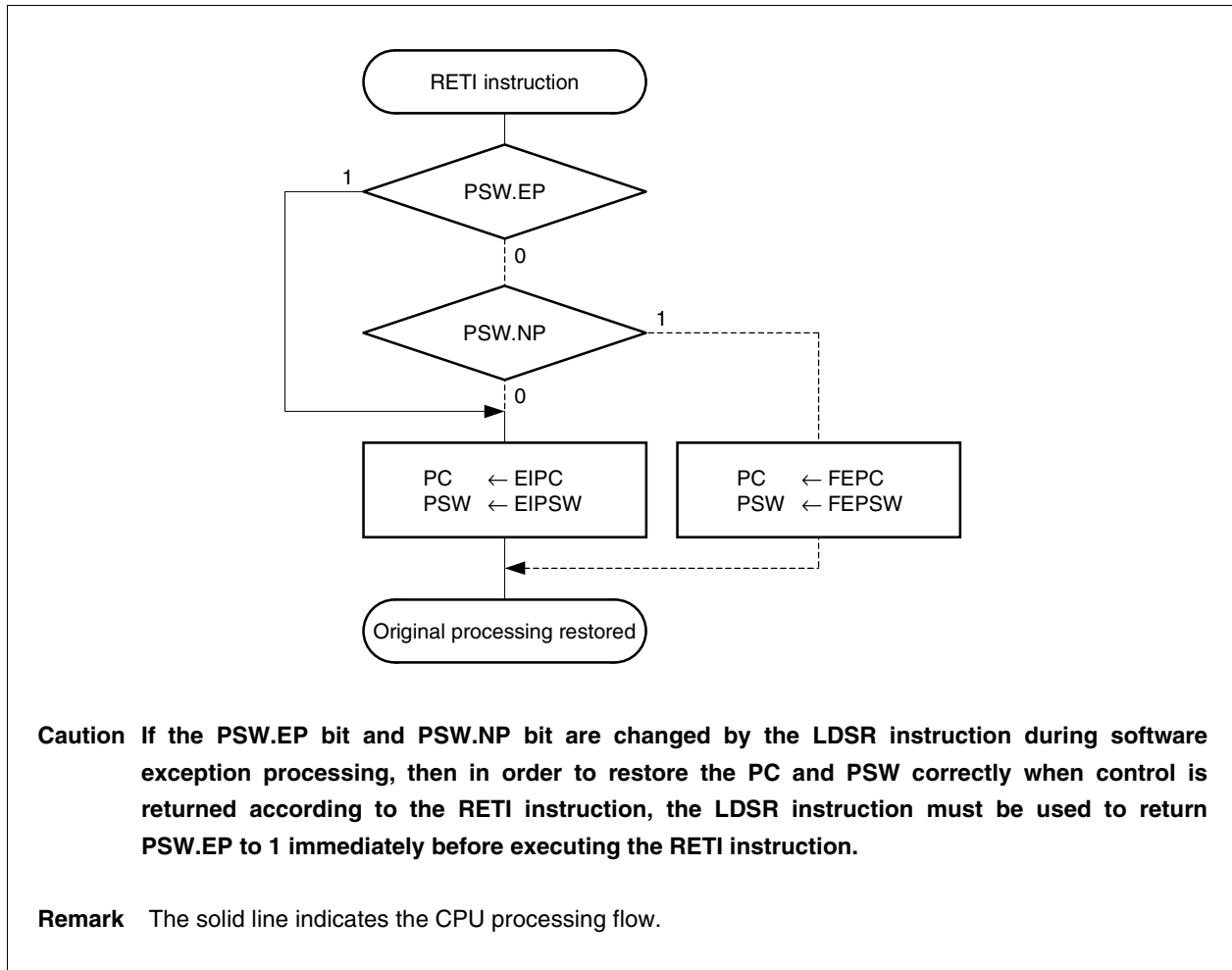
Control is returned from software exception processing according to the RETI instruction.

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

- <1> Since the EP bit of the PSW is 1, fetch the restored PC and PSW from the EIPC and EIPSW.
- <2> Shift control to the fetched restored PC address and PSW status.

Figure 8-13 shows the processing format of the RETI instruction.

Figure 8-13. RETI Instruction Processing Format



8.5 Exception Trap

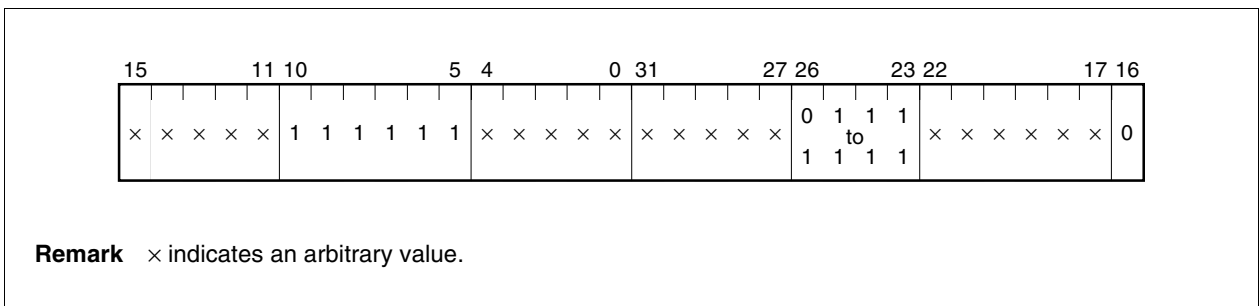
The exception trap is an interrupt that is requested when the illegal execution of an instruction occurs. In the NU85E, the illegal opcode exception (ILGOP: Illegal opcode trap) is assigned for the exception trap.

An illegal opcode exception is generated when the sub-opcode of the instruction to be executed next is an illegal opcode.

8.5.1 Illegal opcode

The illegal opcode, which has a 32-bit long instruction format, is defined as an arbitrary opcode in which bits 10 to 5 are 11111B, bits 26 to 23 are 0111B to 1111B, and bit 16 is 0B.

Figure 8-14. Illegal Opcode



Caution Since a new instruction may be assigned in the future for the illegal opcode, we recommend that this opcode not be used.

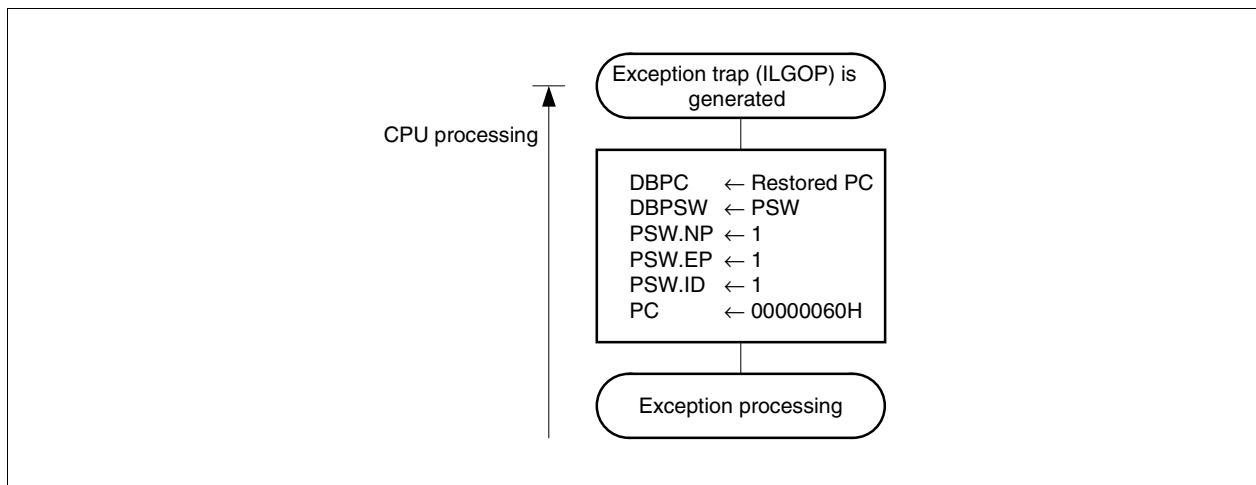
8.5.2 Operation

If an exception trap is generated, the CPU performs the following processing and shifts control to the handler routine.

- <1> Save the restored PC in the DBPC.
- <2> Save the current PSW in the DBPSW.
- <3> Set the NP, EP, and ID bits of the PSW.
- <4> Set the handler address (00000060H) for the exception trap in the PC and shift control.

Figure 8-15 shows the processing format of exception trap processing.

Figure 8-15. Exception Trap Processing Format



8.5.3 Restore

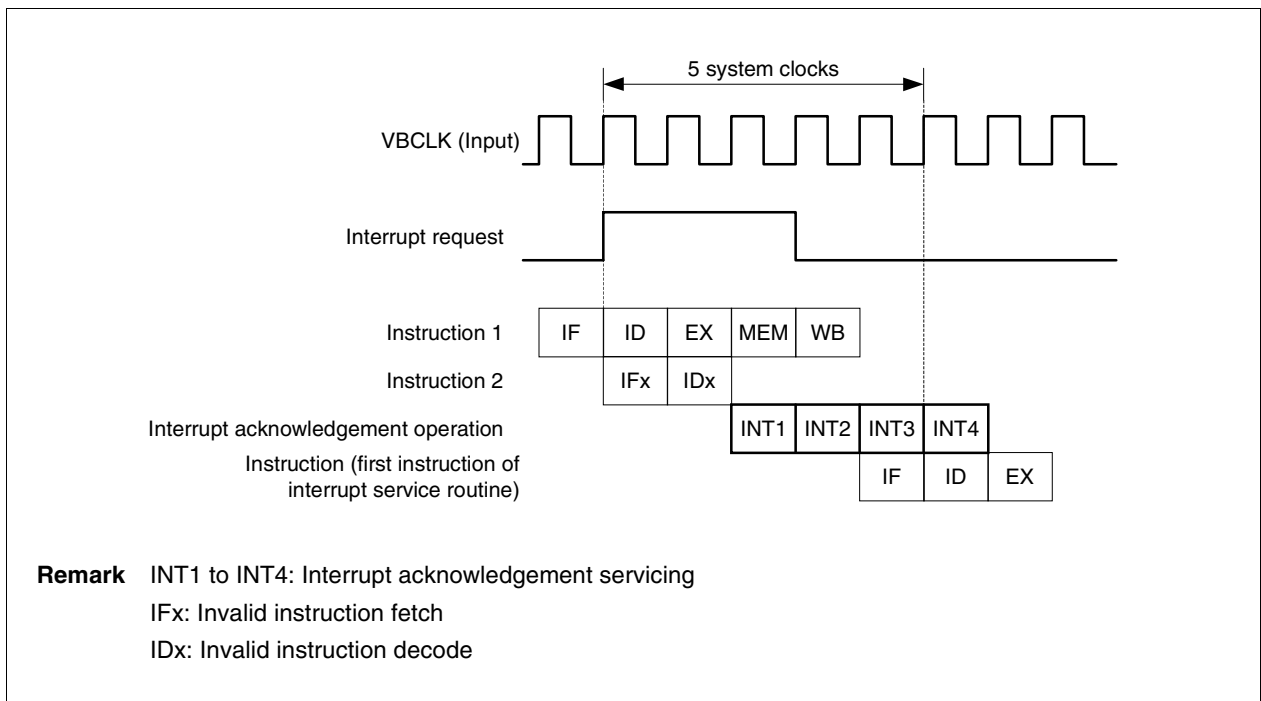
Control cannot return from an exception trap. Perform a system reset according to DCRESZ input.

8.6 Interrupt Response Time

Except in the following cases, the interrupt response time is a minimum of 5 clocks. To input interrupt requests continuously, leave a space of at least 5 clocks between interrupt request inputs.

- During software or hardware STOP mode
- When an external bus is accessed
- When there are two or more successive interrupt request non-sampling instructions (see **8.7 Periods When Interrupts Cannot Be Acknowledged**).
- When the interrupt control register is accessed

Figure 8-16. Example of Pipeline Operation When Interrupt Request Is Acknowledged (Outline)



8.7 Periods When Interrupts Cannot Be Acknowledged

An interrupt is acknowledged while an instruction is being executed. However, an interrupt is not acknowledged between an interrupt request non-sampling instruction and the subsequent instruction (the interrupt is held pending).

The interrupt request non-sampling instructions are as follows.

- EI instruction
- DI instruction
- LDSR reg2, 0x5 instruction (for PSW)
- Store instruction for specific area (xFFF100H to xFFF1FFH^{Note}, xFFF900H to xFFF9FFH)

Note The IMR0 to IMR3, PIC0 to PIC63, ISPR, PRCMD, and PSC registers are allocated to a part of this area.

CHAPTER 9 TEST FUNCTION

The NU85E is equipped with an on-chip test interface control unit (TIC) for testing the NU85E itself or connected peripheral macros via the test buses (TBI39 to TBI0 and TBO34 to TBO0). The test buses are effective when the TEST and BUNRI signals are active.

9.1 Test Pins

9.1.1 Test bus pins (TBI39 to TBI0 and TBO34 to TBO0)

The test bus pins are used in place of normal pins when the NU85E is in unit test mode.

Always extend these pins outside of the ASIC (they can be used in combination with normal pins).

For details, refer to the various cell-based IC family design manuals.

9.1.2 BUNRI and TEST pins

These pins are used to select normal, unit test, or standby test mode.

Table 9-1. List of Test Mode Settings

BUNRI Pin Input Level	TEST Pin Input Level	Mode
Low level	Arbitrary	Normal mode
High level	Low level	Standby test mode
High level	High level	Unit test mode

(1) Normal mode

This is the mode the user normally uses.

When a low-level signal is being input to the BUNRI pin, the pins other than the test pins are enabled, and the NU85E is in normal mode. At this time, input to the TBI39 to TBI0 pins is ignored, and the TBO34 to TBO0 pins are set to high impedance.

(2) Unit test mode and standby test mode

When a high-level signal is being input to the BUNRI pin, the NU85E is in test mode. The two types of test mode are unit test mode and standby test mode.

Circuits should be designed so that floating or bus contention does not occur for the pins constituting the bus (excluding test pins) during unit or standby test mode (For the pin status in each mode, see **2.4 Pin Status**).

(a) Unit test mode

When a high-level signal is being input to the BUNRI and TEST pins, the input from the TBI39 to TBI0 pins is enabled in their place. Also, the test result is output from the TBO34 to TBO0 pins.

Input/output signals from the following pins are also valid in test mode, and operate in the same way as in normal mode. Accordingly, in test mode, be sure to handle these pins as indicated in **9.4 Handling of Each Pin in Test Mode**.

- VSB pins
- NPB pins
- VFB pins
- VDB pins
- Instruction cache pins
- Data cache pins
- RCU pins

Caution Unit test mode is the mode used by NEC to perform testing. Test patterns are provided by NEC.

(b) Standby test mode

When a high-level signal is being input to the BUNRI pin and a low-level signal is being input to the TEST pin, the NU85E is in standby test mode.

The input to the TBI39 to TBI0 pins is ignored, and the TBO34 to TBO0 pins are set to high impedance.

9.1.3 BUNRIOUT pin

The level input to the BUNRI pin is output as is from the BUNRIOUT pin. To support the test bus automatic connection tool, use the output from the BUNRIOUT pin and not that from the BUNRI pin if BUNRI signal logic is required for user circuit separation during the core testing or in places that are not targets of test bus automatic connection such as the cache.

9.2 List of Test Interface Signals

Signal Name	I/O	Function
PHTDIN1, PHTDIN0	Output	Dedicated test signals output to peripheral macros
PHTDO1, PHTDO0	Input	Dedicated test signals input from peripheral macros
TESEN	Output	Enable signal output for setting peripheral macros to test mode
VPTCLK	Output	Peripheral macro test clock output
PHTEST	Output	Status signal output pin indicating peripheral test mode status
TMODE1	Output	Test mode selection output
TMODE0	Output	These are NEC reserved pins. Leave them open.
TBREDZ	Output	

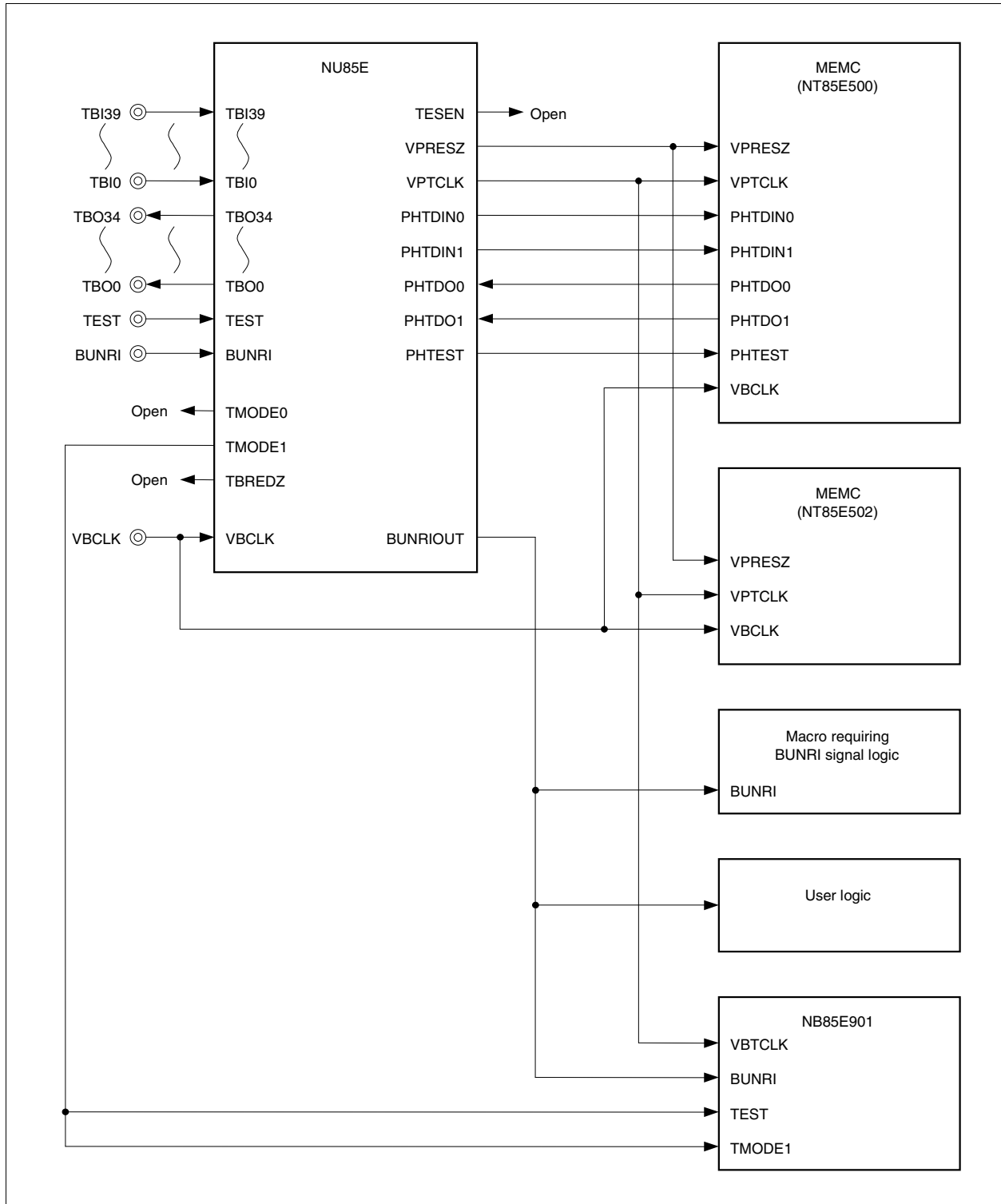
Caution The above signals are only required for tests performed at NEC.

9.3 Example of Connection of Peripheral Macro in Test Mode

The NPB peripheral macro, MEMC (NT85E500, NT85E502), instruction cache, and data cache supported by NEC are tested via the NU85E.

An example of the connections between the NU85E, the NPB peripheral macro, and the MEMC is shown below.

★ **Figure 9-1. Peripheral Macro Connection Example**



9.4 Handling of Each Pin in Test Mode

(1) Pins other than those for test mode

(a) Input pins

Input a low level to the VAREQ pin. Special handling is not required for pins other than the VAREQ pin (handle as in normal mode).

(b) Output pins

Special handling is not required (handle as in normal mode).

(2) Test mode pins (except TBI39 to TBI0, TBO34 to TBO0, BUNRI, TEST, and BUNRIOUT)

Handle the pins for test mode as indicated below.

Pin Name	I/O	Connection Method		
		When MEMC Is Connected	When Cache Is Connected	When Neither MEMC nor Cache Is Connected
PHTDOn	Input	Connect to the PHTDOn pin of the NT85E500.	–	Input low level.
PHTDINn	Output	Connect to the PHTDINn pin of the NT85E500.	–	Leave open.
VPRESZ	Output	Connect to the VPRESZ pin of the NT85E500, NT85E502.	Connect to the VPRESZ pin.	
VPTCLK	Output	Connect to the VPTCLK pin of the NT85E500, NT85E502.	Connect to the VPTCLK pin.	
TESEN	Output	–	–	
PHTEST	Output	Connect to the PHTEST pin of the NT85E500.	–	
TMODEn, TBREDZ	Output	Leave open.		

★ Remark n = 1, 0

(3) Precautions when NB85E901 is connected

When the NB85E901 (RCU) is connected to the NU85E, the following pins are used in the unit test mode. All of these pins should be attached off chip as external pins.

- TBI39 to TBI0^{Note 1}
- TBO34 to TBO0^{Note 1}
- TEST^{Note 1}
- BUNRI
- DCK^{Note 2}
- DRSTZ^{Note 2}
- DMS^{Note 2}
- DDI^{Note 2}
- DDO^{Note 2}
- DBINT^{Notes 1, 2}

Notes 1. Can be used as an alternate function pin with a pin used in normal mode.

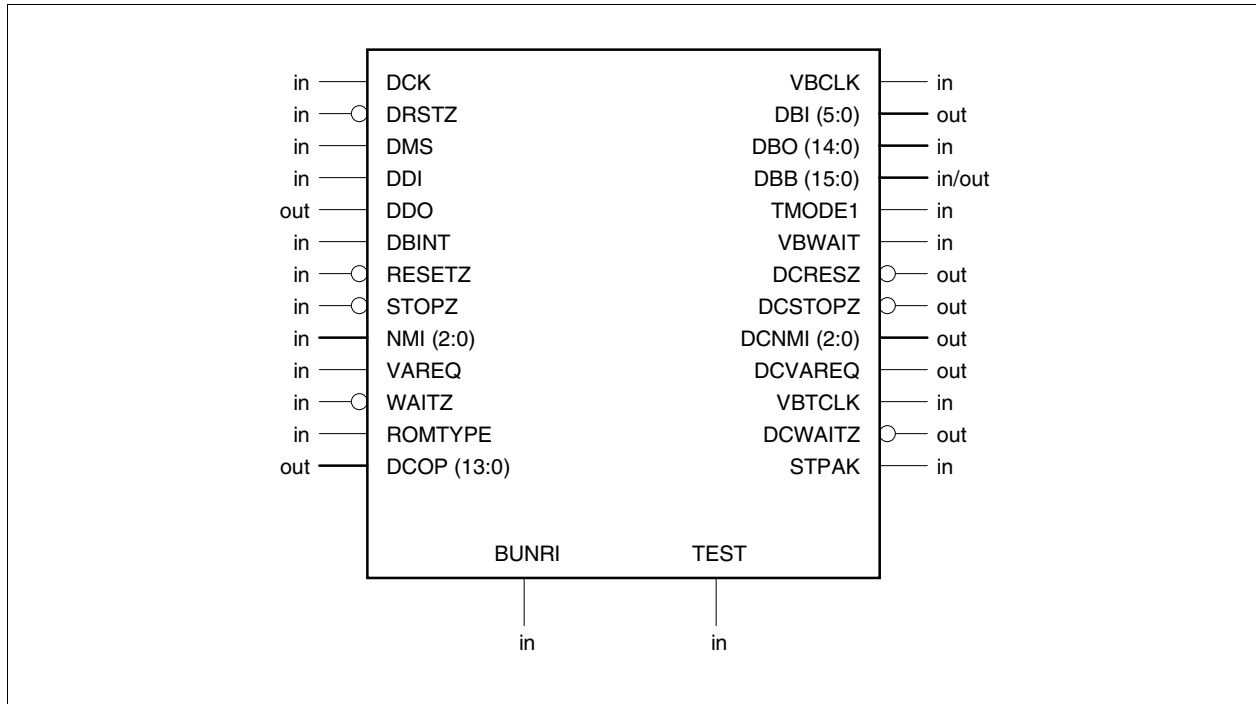
2. Pins of the NB85E901 (For details, refer to **CHAPTER 10 NB85E901.**)

CHAPTER 10 NB85E901

(Under Development)

The NB85E901 (RCU: Run Control Unit) is a run control unit that realizes the execution of JTAG communication and debug processing. Connection of the NB85E901 with an N-Wire type in-circuit emulator (N-Wire type IE) makes it possible to perform on-chip debugging on the NU85E.

10.1 Symbol Diagram



10.2 Pin Functions

10.2.1 Pin function list

Pin Name		I/O	Function
N-Wire type IE connection pins	DCK	Input	Clock input for RCU
	DRSTZ	Input	Reset input for RCU
	DMS	Input	Debug mode selection input
	DDI	Input	Debug data input
	DDO	Output	Debug data output
	DBINT	Input	External debug interrupt input
System control pins	RESETZ	Input	System reset input
	STOPZ	Input	Hardware STOP mode request input
	NMI2 to NMI0	Input	Non-maskable interrupt input
	VAREQ	Input	Bus access right request input
	WAITZ	Input	Wait request input
	ROMTYPE	Input	NEC reserved pin (input low level)
	DCOP13 to DCOP0	Output	NEC reserved pin (leave open)
NU85E connection pins	VBCLK	Input	System clock input
	DBI5 to DBI0	Output	Debug control output
	DBO14 to DBO0	Input	Debug control input
	DBB15 to DBB0	I/O	Debug control I/O
	TMODE1	Input	Test mode selection input
	VBWAIT	Input	Wait response input
	DCRESZ	Output	Reset output
	DCSTOPZ	Output	Hardware STOP mode request output
	DCNMI2 to DCNMI0	Output	Non-maskable interrupt output
	DCVAREQ	Output	Bus access right request output
	VBTCLK	Input	Clock input for testing
	Peripheral connection pins	DCWAITZ	Output
STPAK		Input	STOP mode request acknowledge input
Test mode pins	BUNRI	Input	Normal/test mode selection input
	TEST	Input	Test mode selection input

10.2.2 Pin functions

(1) N-Wire type IE connection pins

Caution N-Wire type IE connection pins (DCK, DRSTZ, DMS, DDI, DDO, DBINT) must be attached off the chip as external pins since they are used in the unit test mode of the NB85E901. Do not use these pins as alternate function pins (however, the DBINT pin can be used as the alternate function of a pin other than the TBI39 to TBI0, TBO34 to TBO0, TEST, BUNRI, DCK, DRSTZ, DMS, DDI, and DDO pins).

(a) **DCK (input)**

This is the pin to which the clock for the RCU is input from the N-Wire type IE.

(b) **DRSTZ (input)**

This is the RCU reset input pin. The RCU is reset asynchronously when a low level is input.

(c) **DMS (input)**

This is the pin to which the debug mode selection is input from the N-Wire type IE.

(d) **DDI (input)**

This is the pin to which the debug data is input from the N-Wire type IE.

(e) **DDO (output)**

This is the pin from which the debug data is output to the N-Wire type IE.

(f) **DBINT (input)**

This is the external debug interrupt input pin. An active level (high level) is input when shifting to the debug mode by an external request.

(2) System control pins

(a) **RESETZ (input)**

This is the system reset input pin.

(b) **STOPZ (input)**

This is the hardware STOP mode request input pin.

(c) **NMI2 to NMI0 (input)**

These are non-maskable interrupt input pins.

(d) **VAREQ (input)**

This is the bus access right request input pin.

(e) **WAITZ (input)**

This is the external wait request input pin.

(f) ROMTYPE (input)

This is an NEC reserved pin. Always input a low level.

(g) DCOP13 to DCOP0 (output)

These are NEC reserved pins. Leave them open.

(3) NU85E connection pins**(a) VBCLK (input)**

This is the system clock input pin.

(b) DBI5 to DBI0 (output)

These are debug control output pins. Connect them to pins DBI5 to DBI0 on the NU85E.

(c) DBO14 to DBO0 (input)

These are debug control input pins. Connect them to pins DBO14 to DBO0 on the NU85E.

(d) DBB15 to DBB0 (I/O)

These are debug control I/O pins. Connect them to pins DBB15 to DBB0 on the NU85E.

(e) TMODE1 (input)

This is the test mode selection input pin. Connect it to the TMODE1 pin on the NU85E.

(f) VBWAIT (input)

This is the wait response input pin.

(g) DCRESZ (output)

This is the reset output pin. Connect it to the DCRESZ pin on the NU85E.

(h) DCSTOPZ (output)

This is the hardware STOP mode request output pin. Connect it to the DCSTOPZ pin on the NU85E.

(i) DCNMI2 to DCNMI0 (output)

These are non-maskable interrupt output pins. Connect them to pins DCNMI2 to DCNMI0 on the NU85E.

(j) DCVAREQ (output)

This is the bus access right request output pin. Connect it to the VAREQ pin on the NU85E.

(k) VBTCLK (input)

This is the test clock input pin. Connect it to the VPTCLK pin on the NU85E.

(4) Peripheral connection pins**(a) DCWAITZ (output)**

This is the external wait request output pin.

(b) STPAK (input)

This is the STOP mode request acknowledge input pin. Input the STPAK signal from the memory controller.

(5) Test mode pins**(a) BUNRI (input)**

This is the input pin for selecting normal mode or test mode.

(b) TEST (input)

This is the test bus control input pin. Connect it to the TMODE1 pin on the NU85E.

10.2.3 Recommended connection of unused pins

	Pin Name	I/O	Recommended Connection Method
N-Wire type IE connection pins	DCK, DRSTZ, DMS, DDI	Input	–
	DDO	Output	–
	DBINT	Input	Input a low level.
System control pins	RESETZ	Input	–
	STOPZ, WAITZ	Input	Input a high level.
	NMI2 to NMI0, VAREQ, ROMTYPE	Input	Input a low level.
	DCOP13 to DCOP0	Output	Leave open.
NU85E connection pins	VBCLK, DBO14 to DBO0, TMODE1, VBTCLK	Input	–
	DBI5 to DBI0	Output	–
	DBB15 to DBB0	I/O	–
	VBWAIT	Input	Input a low level.
	DCRESZ, DCSTOPZ, DCNMI2 to DCNMI0, DCVAREQ	Output	Leave open.
Peripheral connection pins	DCWAITZ	Output	Leave open.
	STPAK	Input	Input the STPRQ signal of the NU85E.
Test mode pins	BUNRI, TEST	Input	–

10.2.4 Pin status

The following table shows the status in each operating mode of the pins that have output functions.

Table 10-1. Pin Status in Each Operating Mode

Pin Name	Pin Status					
	Reset ^{Note}	Software STOP Mode	Hardware STOP Mode	HALT Mode	Standby Test Mode	Unit Test Mode
DDO	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
DCOP13 to DCOP10	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
DCOP9	L/L	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
DCOP8 to DCOP3	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
DCOP2	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates
DCOP1, DCOP0	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
DBI5	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates
DBI4 to DBI2	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
DBI1	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates
DBI0	L/H	L/H	L/H	L/H	L/H	L/H
DBB15 to DBB0	Retained/ Retained	Retained/ Retained	Retained/ Retained	Retained/ Retained	Retained/ Retained	Retained/ Operates
DCRESZ	RESETZ	RESETZ	RESETZ	RESETZ	RESETZ	Undefined
DCSTOPZ	STOPZ	STOPZ	STOPZ	STOPZ	STOPZ	Undefined
DCNMI2 to DCNMI0	NMI2 to NMI0	NMI2 to NMI0	NMI2 to NMI0	NMI2 to NMI0	NMI2 to NMI0	Undefined
DCVAREQ	VAREQ	VAREQ	VAREQ	VAREQ	VAREQ	Undefined
DCWAITZ	WAITZ	WAITZ	WAITZ	WAITZ	WAITZ	Undefined

Note When a low level is input to the DCRESZ pin and an external clock is input to the VBCLK pin.

Remarks 1. L: Low-level output

H: High-level output

Retained: Retains previous status

- The item to the left of the slash (/) indicates the status when a low level is input to the DRSTZ pin, and the item to right indicates the status when a high level is input to the DRSTZ pin.
- The status of the DCRESZ, DCSTOPZ, DCNMI2 to DCNMI0, DCVAREQ, and DCWAITZ pins indicates the status either when a low level is input to the DRSTZ pin or when a high level is input to the DRSTZ pin and the external input signals (RESETZ, STOPZ, NMI2 to NMI0, VAREQ, and WAITZ) have not been masked.

Caution The following input pins must be set according to the table below in the respective operating modes.

Pin Name	Pin Status					
	Reset ^{Note}	Software STOP Mode	Hardware STOP Mode	HALT Mode	Standby Test Mode	Unit Test Mode
DRSTZ	L/H	L/H	L/H	L/H	L/H	L/H
DDI	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates	H/Operates
DBINT	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates	L/Operates
ROMTYPE	L	L	L	L	L	L

Note When a low level is input to the DCRESZ pin and an external clock is input to the VBCLK pin.

Remarks 1. L: Low-level output

H: High-level output

Retained: Retains previous status

- 2.** The item to the left of the slash (/) indicates the status when a low level is input to the DRSTZ pin, and the item to right indicates the status when a high level is input to the DRSTZ pin.

10.3 Debug Function

(1) Debug interface

Communication with the host machine is performed via the N-Wire type IE using the DCK, DRSTZ, DMS, DDI, and DDO signals. JTAG communication specification is used for the interface. The boundary scan function is not supported.

(2) On-chip debug

By connecting with the N-Wire type IE, it is possible to debug the NB85E901 on the NU85E chip. For details on the above connection, refer to **10.5 N-Wire Type IE Connection**.

(3) Forcible reset function

The NB85E901 unit can be forcibly reset.

(4) Break reset function

The CPU can be started in debug mode immediately after CPU reset release.

(5) Forcible break function

Execution of the user program can be forcibly interrupted. Note that the illegal opcode exception handler (start address: 00000060H) cannot be used.

(6) Debug interrupt interface

The forcible break function can be executed by inputting a high level to the DBINT pin.

Remark It is also possible to release the HALT, software STOP, and hardware STOP modes by DBINT input.

(7) Breakpoint function

Execution of the user program can be interrupted at an arbitrary address. Also, data access to an arbitrary address can be interrupted. Note that the illegal opcode exception handler (start address: 00000060H) cannot be used.

There are two kinds of instruction/access alternate breakpoints: a pre-execution break and a post-access execution break.

(8) Debug monitor function

A debug-dedicated memory space, which is different to the user memory space, is used during debugging (background monitor format). Execution of the user program can be started from an arbitrary address.

It is also possible to read/write the user resource (such as memory and I/O) and download the user program during a user program interruption.

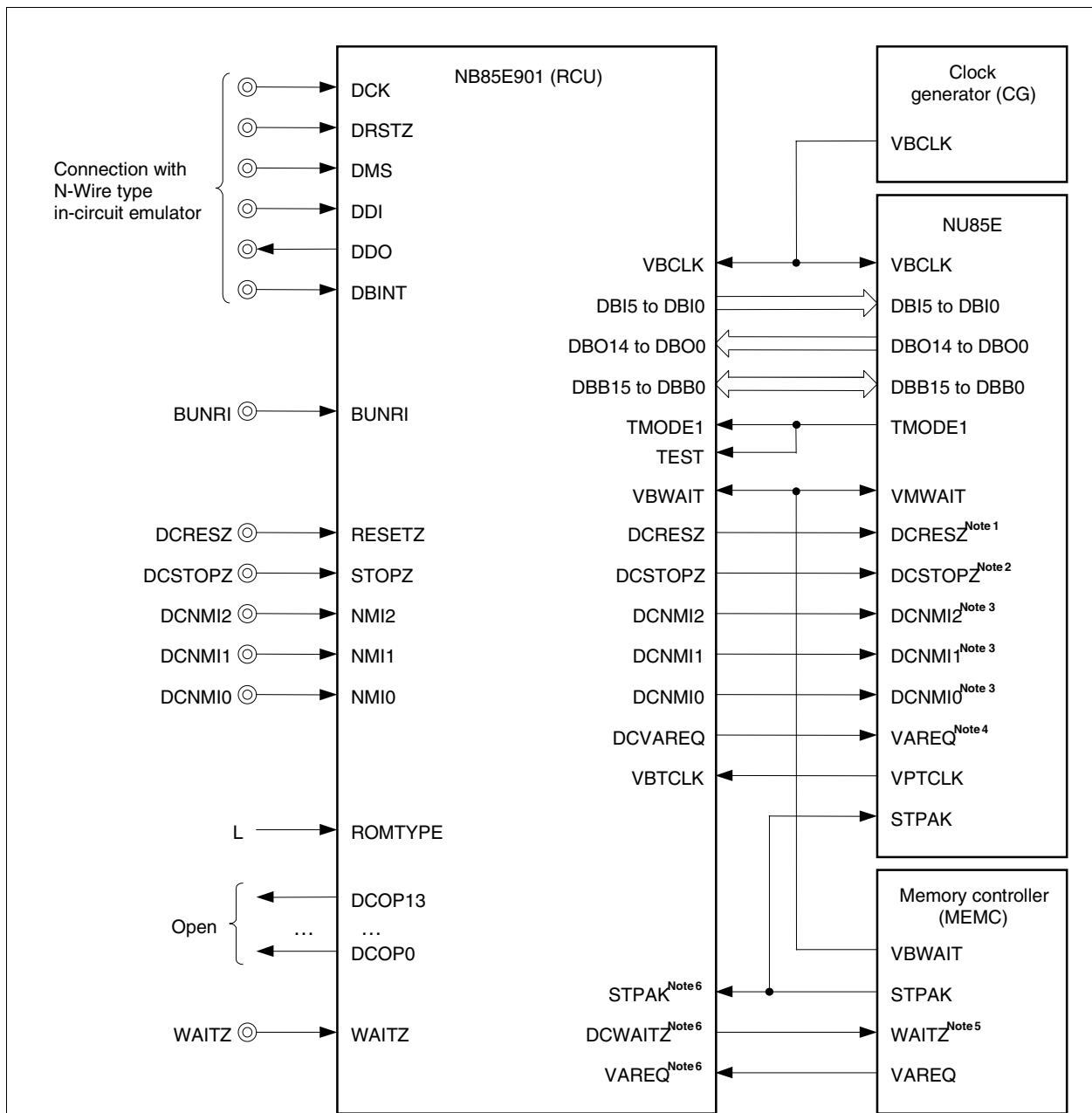
(9) Mask function

The external input signals (RESETZ, STOPZ, NMI2 to NMI0, VAREQ, WAITZ) can be masked.

10.4 NU85E Connection Example

Figure 10-1 shows an example of the connection between the NB85E901 and the NU85E.

Figure 10-1. NB85E901 and NU85E Connection Example

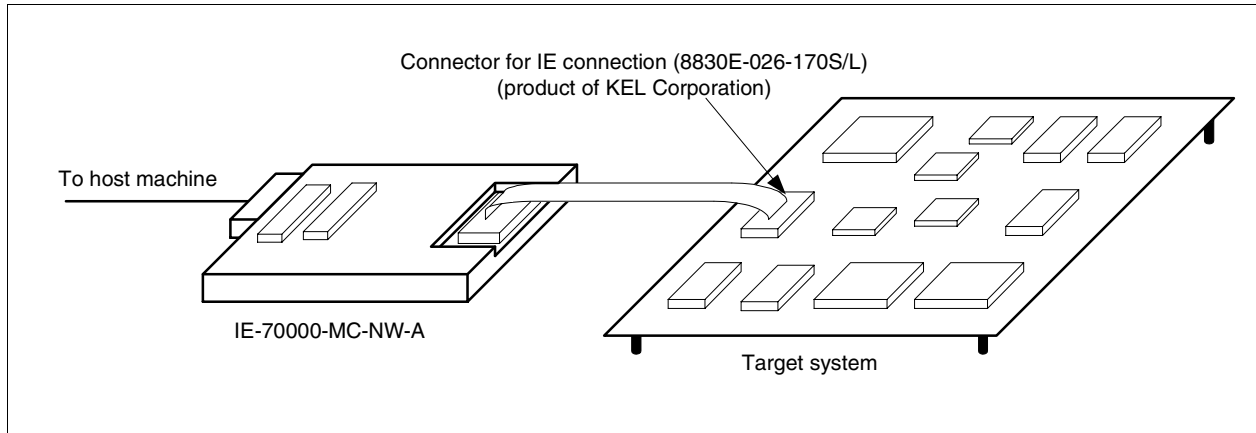


- Notes**
1. Input the signal input from the RESETZ pin of the RCU and output from the DCRESZ pin.
 2. Input the signal input from the STOPZ pin of the RCU and output from the DCSTOPZ pin.
 3. Input the signal input from the NMI_n pin of the RCU and output from the DCNMI_n pin (n = 2 to 0).
 4. Input the signal input from the VAREQ pin of the RCU and output from the DCVAREQ pin.
 5. Input the signal input from the WAITZ pin of the RCU and output from the DCWAITZ pin.
 6. If the MEMC is not used, process as shown in **10.2.3 Recommended connection of unused pins.**

10.5 N-Wire Type IE Connection

In order to connect the N-Wire type IE (IE-70000-MC-NW-A), it is necessary to mount a connector for IE connection and a connection circuit on the target system.

Figure 10-2. N-Wire Type IE Connection



10.5.1 IE connector (target system side)

Figure 10-3 shows the pin layout of the IE connector (target system side), and Table 10-2 describes the pin functions.

Remark The recommended connectors are as follows.

- 8830E-026-170S (product of KEL Corporation): 26-pin straight type
- 8830E-026-170L (product of KEL Corporation): 26-pin right angle type

Figure 10-3. IE Connector Pin Layout Diagram (Target System Side)

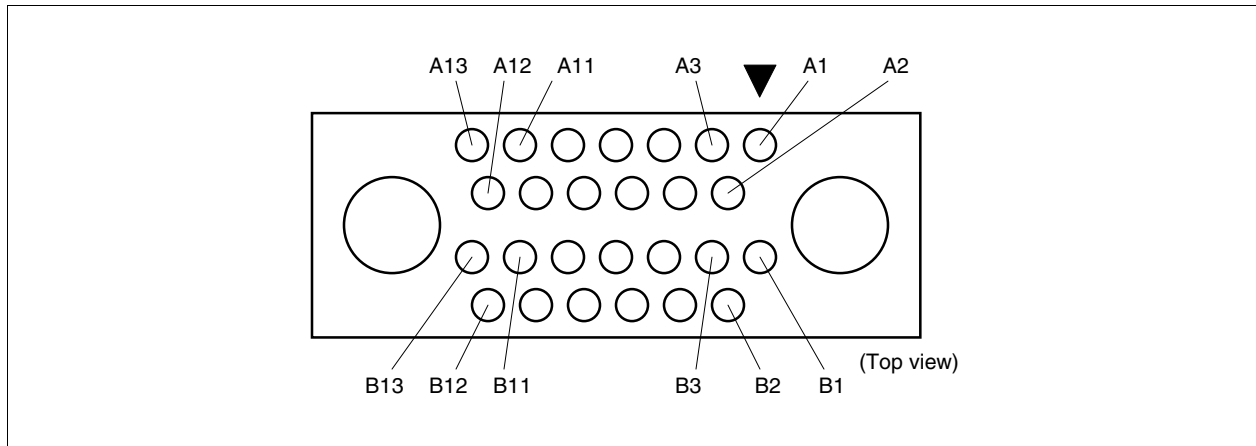


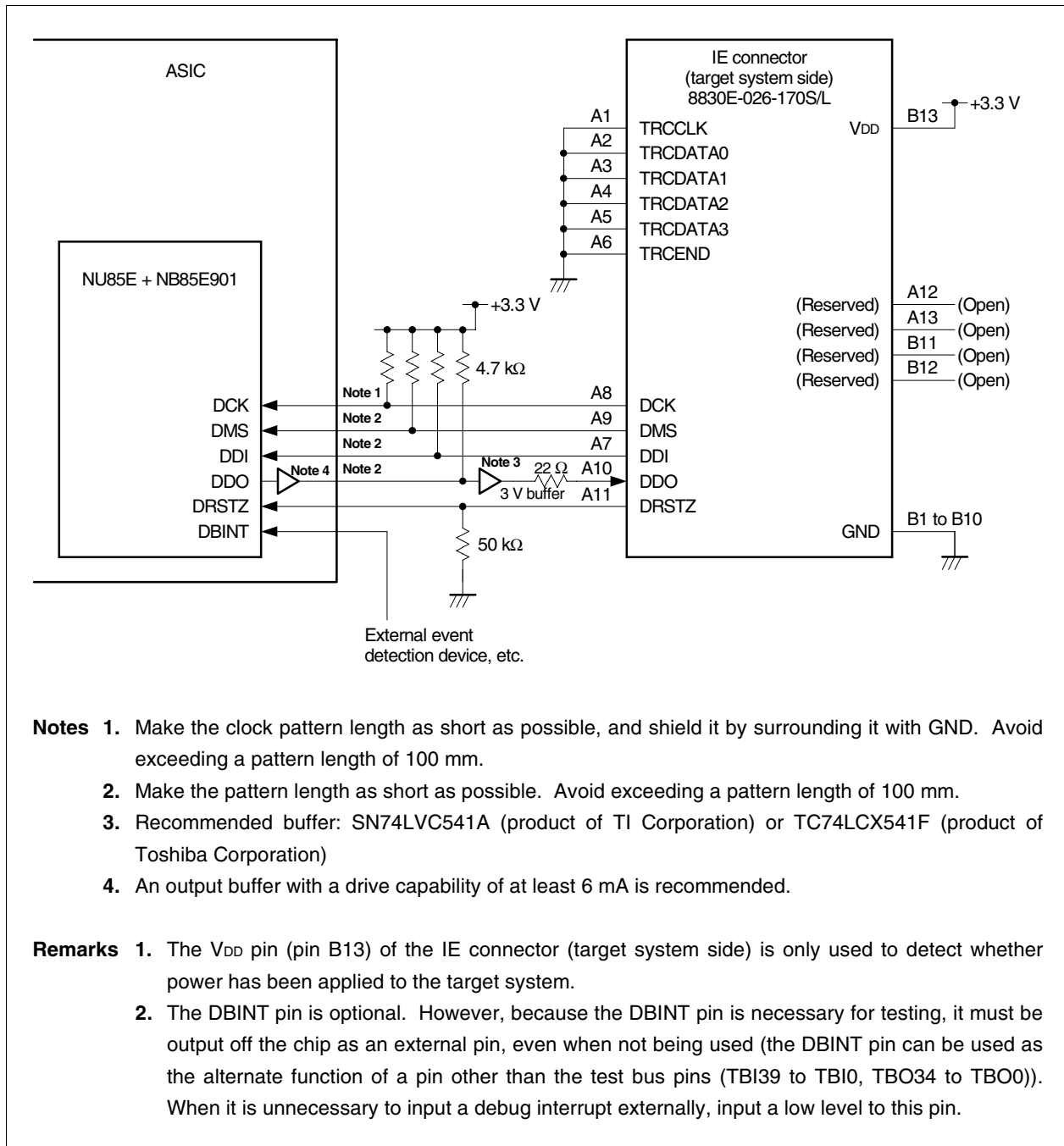
Table 10-2. IE Connector Pin Functions (Target System Side)

Pin No.	Pin Name	I/O	Pin Function
A1	TRCCLK	Input	Trace clock input
A2	TRCDATA0	Input	Trace data 0 input
A3	TRCDATA1	Input	Trace data 1 input
A4	TRCDATA2	Input	Trace data 2 input
A5	TRCDATA3	Input	Trace data 3 input
A6	TRCEND	Input	Trace data end input
A7	DDI	Output	Debug serial interface data output
A8	DCK	Output	Debug serial interface clock output
A9	DMS	Output	Debug serial interface transfer mode selection output
A10	DDO	Input	Debug serial interface data input
A11	DRSTZ	Output	DCU reset output
A12	(Reserved)	–	(Leave open)
A13	(Reserved)	–	(Leave open)
B1	GND	–	–
B2	GND	–	–
B3	GND	–	–
B4	GND	–	–
B5	GND	–	–
B6	GND	–	–
B7	GND	–	–
B8	GND	–	–
B9	GND	–	–
B10	GND	–	–
B11	(Reserved)	–	(Leave open)
B12	(Reserved)	–	(Leave open)
B13	V _{DD}	–	+3.3 V input (for monitoring target power supply application)

10.5.2 Example of recommended circuit when connecting NB85E901 and NU85E

Figure 10-4 shows an example of the circuit recommended for IE connector section (target system side).

Figure 10-4. Example of Recommended Circuit for IE Connection (NU85E + NB85E901)



APPENDIX A ROM/RAM ACCESS TIMING

Figure A-1. ROM Access Timing

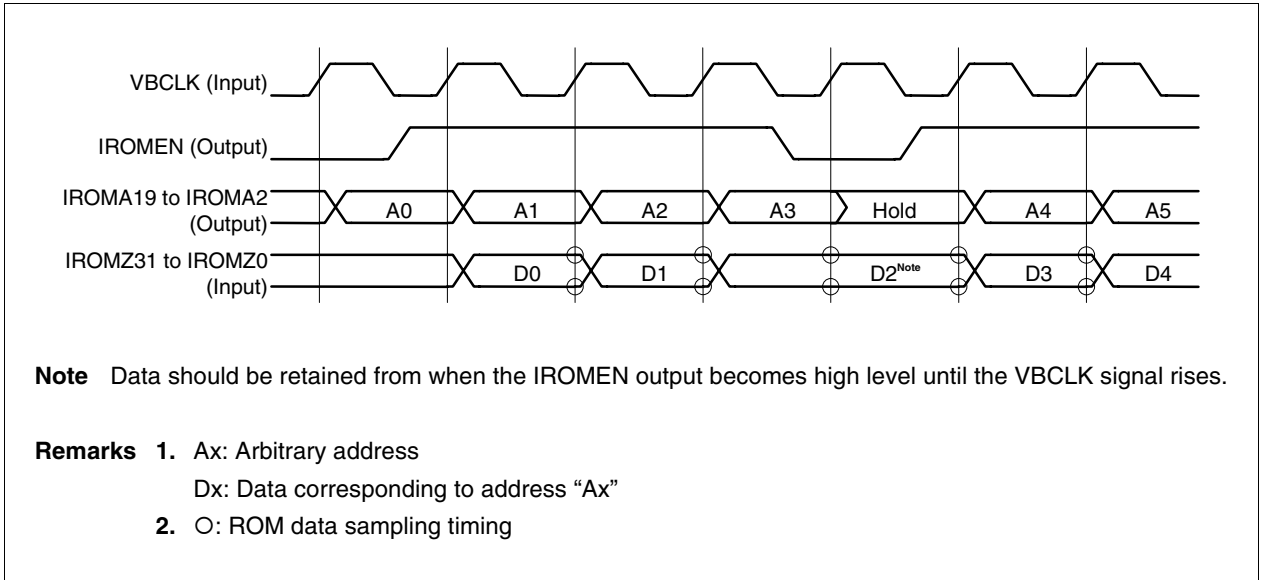
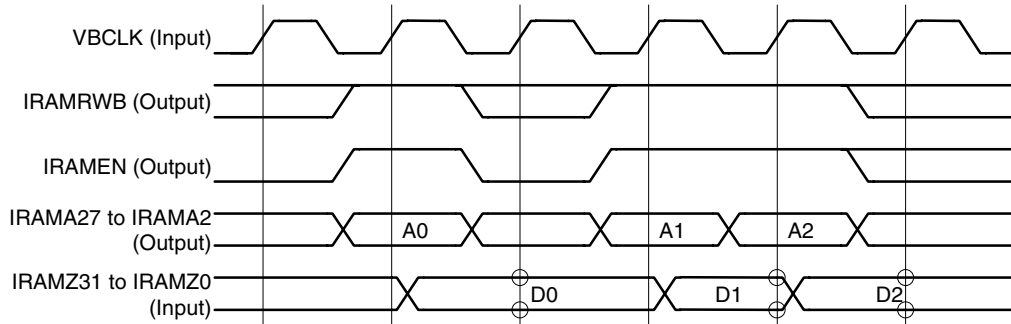


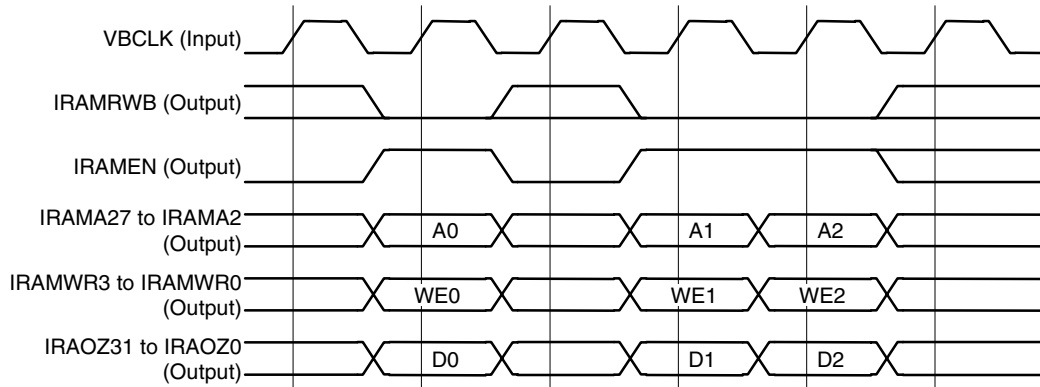
Figure A-2. RAM Access Timing

(a) Read timing



- Remarks**
1. Ax: Arbitrary address
Dx: Data corresponding to address "Ax"
 2. O: RAM data sampling timing

(b) Write timing



- Remark**
- Ax: Arbitrary address
 - Dx: Data corresponding to address "Ax"

APPENDIX B INDEX

[A]

Address space	58
Application system example	18

[B]

BBR	118
BC15 to BC0	156
BCU	74
BCUNCH	39
BCU-related register setting examples	91
BEC	87
BE _n 0	87
BHC	90
BH _n 0	90
BH _n 1	90
Block transfer mode	171
BPC	85, 121
BSC	86
BS _n 1, BS _n 0	86
BUNRI	44
BUNRIOUT	44
Bus size configuration register	86
Bus size setting function	86

[C]

Cache configuration	90
Cache configuration register	90
CGREL	35
CH3 to CH0	160
Chip area select control register 0	77
Chip area select control register 1	78
Clock control	144
Command register	139
CPU	51
CSC0	77
CSC1	78
CS _n 3 to CS _n 0	77, 78
CTBP	55
CTPC	55
CTPSW	55
CY	57

[D]

DA15 to DA0	155
-------------------	-----

DA27 to DA16	154
DAD1, DAD0	158
DADC0 to DADC3	157
Data area	60
Data transfer using VSB	94
DBB15 to DBB0	41
DBC0 to DBC3	156
DBI5 to DBI0	41
DBO14 to DBO0	41
DBPC	55
DBPSW	55
DCHC0 to DCHC3	159
DCNMI2 to DCNMI0	36
DCRESZ	34
DCSTOPZ	35
DDA0 to DDA3	154
DDIS	160
Debug function	242
DMA addressing control registers 0 to 3	157
DMA bus state	163
DMA channel control registers 0 to 3	159
DMA channel priorities	151
DMA destination address registers 0 to 3	154
DMA disable status register	160
DMA restart register	161
DMA source address registers 0 to 3	152
DMA transfer count registers 0 to 3	156
DMA transfer start factors	175
DMA transfer timing examples	180
DMAC	149
DMAC bus cycle state transitions	165
DMACTV3 to DMACTV0	36
DMARQ3 to DMARQ0	36
DMTCO3 to DMTCO0	36
DRST	161
DS1, DS0	157
DSA0 to DSA3	152

[E]

ECR	55, 56
EICC	56
EIPC	55
EIPSW	55
EN3 to EN0	161

Endian configuration register	87	IDRRQ	39
Endian setting function.....	87	IDWRQ.....	39
ENn	160	IDEA27 to IDEA0	40
EP	57	IDED31 to IDED0	40
EVAD15 to EVAD0.....	41	IDES	40
EVASTB.....	41	IDHUM.....	40
EVCLRIP.....	41	IDMASTP.....	36
EVDSTB.....	41	IDRETR	40
EVIEN	41	IDRRDY	40
EVINTAK.....	41	IDSEQ2	39
EVINTLV6 to EVINTLV0	42	IDSEQ4	39
EVINTRQ	41	IDUNCH.....	40
EVIREL	41	IFID256.....	43
EVLKRT	41	IFIMAEN.....	43
EVOEN.....	41	IFIMODE3, IFIMODE2.....	44
Example of connection of peripheral macro in test mode	232	IFINSZ1, IFINSZ0.....	43
Exception trap	227	IFIRA64, IFIRA32, IFIRA16	42
External memory	18	IFIRABE.....	44
External memory area.....	67	IFIRASE.....	44
[F]		IFIROB2.....	42
FCOMB	44	IFIROBE	44
FECC	56	IFIROME.....	42
FEPC	55	IFIROPR.....	44
FEPSW	55	IFIUNCH0	44
Flyby transfer	174	IFIUNCH1	43
[G]		IFIUSWE.....	44
General-purpose registers	53	IFIWRTH.....	43
[H]		IIAACK.....	38
HALT mode.....	140	IIBTFT.....	38
Handling of each pin in test mode.....	233	IIDLEF	38
Hardware STOP mode.....	143	IIDRRQ.....	38
HWSTOPRQ	35	IIEA25 to IIEA2	38
[I]		IIEDI31 to IIEDI0.....	38
IBAACK.....	38	IIRCAN	39
IBBTFT.....	39	Illegal opcode	227
IBDLE3 to IBDLE0	38	IMR0 to IMR3	222
IBDRDY.....	38	INITn.....	159
IBDRRQ	38	In-service priority register	223
IBEA25 to IBEA2.....	38	INT63 to INT0	36
IBEDI31 to IBEDI0	38	INTC	206
ID	57, 224	Internal block diagram	22
IDAACK.....	39	Internal units	23
IDDARQ	39	Interrupt control registers 0 to 63	221
IDDRDY	40	Interrupt mask registers 0 to 3.....	222
		Interrupt response time.....	229
		Interrupt source register	55, 56
		Interrupt/exception list	206
		Interrupt/exception table.....	62

INTM	137
IR	152, 154
IRAMA27 to IRAMA2	37
IRAMEN	37
IRAMRWB	37
IRAMWR3 to IRAMWRO	37
IRAMWT	37
IRAMZ31 to IRAMZ0	37
IRAOZ31 to IRAOZ0	37
IROMA19 to IROMA2	36
IROMAE	36
IROMCS	36
IROMEN	36
IROMIA	36
IROMWT	36
IROMZ31 to IROMZ0	36
IRRSA	40
ISPR	223
ISPR7 to ISPR0	223
[L]	
Line transfer mode	169
[M]	
Maskable interrupt priorities	217
Maskable interrupts	214
Memory banks	74
MLEn	159
[N]	
NB85E901	234
NB85E901 and NU85E connection example	243
Next address setting function	162
NMI	209
NMI0M	137
NMI1M	137
NMI2M	137
Non-maskable interrupts	209
Normal mode	230
NP	57
NPB	17
NPB strobe wait control register	123
N-Wire type IE connection	244
[O]	
On-chip debugging	73
OV	57
[P]	
PA13 to PA00	85, 121
PA15	85, 121
PC	53, 54
Periods when interrupts cannot be acknowledged	229
Peripheral I/O area	65
Peripheral I/O area select control register	85, 121
Peripheral I/O registers	67
PHEVA	44
PHTDIN1, PHTDIN0	44
PHTDO1, PHTDO0	44
PHTEST	45
PIC0 to PIC63	221
PIFn	221
Pin functions	25
Pin status	48
PMKn	221, 222
Power save control register	137
Power save function	136
PPRn2 to PPRn0	221
PRCMD	139
Program area	59
Program counter	53, 54
Program registers	53
Program status word	55, 57
Programmable chip select function	77
Programmable peripheral I/O area	120
Programmable peripheral I/O area selection function	83
PSC	137
PSW	55, 57
[R]	
r0 to r31	53
RAM	18
RAM area	64
RCU interface	73
Recommended connection of unused pins	46
REG7 to REG0	139
Retry function	125
ROM	18
ROM area	62
ROM relocation function	62
ROM/RAM access timing	247
[S]	
S	57

SA15 to SA0	153	Two-cycle transfer	173
SA27 to SA16	152	[U]	
SAD1, SAD0	158	Unit test mode	231
SAT	57	[V]	
Single transfer mode	166	VAACK	30
Single-step transfer mode	168	VAPREQ	30
Software exception	225	VAREQ	30
Software STOP mode	141	VBCLK	35
Standby test mode	231	VBDC	34
STBC	136	VBDI31 to VBDI0	30
STGn	159	VBDO31 to VBDO0	30
STP	137	VBDV	34
STPAK	35	VDB	18
STPRQ	35	VDCSZ7 to VDCSZ0	34
SUWL2 to SUWL0	123	VDELPZ	33
SWSTOPRQ	35	VFB	18
Symbol diagram	21	VMA27 to VMA0	30
System registers	55	VMAHLD	33
[T]		VMBENZ3 to VMBENZ0	31
T0 state	163	VMBSTR	33
T1FH state	163	VMCTYP2 to VMCTYP0	32
T1FHI state	164	VMLAST	33
T1R state	163	VMLOCK	31
T1RI state	163	VMSEQ2 to VMSEQ0	32
T1W state	163	VMSIZE1, VMSIZE0	31
T1WI state	163	VMSTZ	30
T2FH state	164	VMTTYP1, VMTTYP0	30
T2R state	163	VMWAIT	33
T2RI state	163	VMWRITE	31
T2W state	163	VPA13 to VPA0	29
TBI39 to TBI0	44	VPDACT	29
TBO34 to TBO0	44	VPDI15 to VPDI0	29
TBREDZ	45	VPDO15 to VPDO0	29
TCn	159	VPDV	29
TDIR	158	VPLOCK	29
TE state	164	VPRESZ	45
Terminal count output when DMA transfer is complete	176	VPRETR	29
TESEN	44	VPSTB	29
TEST	44	VPTCLK	44
Test function	230	VPUBENZ	29
TI state	163	VPWRITE	29
TM1, TM0	158	VSA13 to VSA0	30
TMODE0	45	VSAHLD	33
TMODE1	45	VSB	17
Transfer objects	151	VSBENZ1	31
TTYP	158	VSLAST	33

VSLOCK 31
VSSELPZ..... 33
VSSTZ 30
VSWAIT 33
VSWC 123
VSWL2 to VSWL0 124
VSWRITE 31

[W]

Wait insertion function 123

[Z]

Z 57

APPENDIX C REVISION HISTORY

The following shows the major revisions in the previous edition (2nd edition). The numbers in the Pages column indicate those in the previous edition.

(1) From 1st to 2nd

Pages	Description
p.29	Modification of 2.2.1 (4) VPWRITE
p.29	Modification of 2.2.1 (10) VPDV
p.36	Modification of 2.2.4 (1) IDMASTP
p.37	Modification of 2.2.7 (1) IRAMA27 to IRAMA2
p.45	Modification of 2.2.13 (12) TMODE1
p.46	Modification of 2.3 Recommended Connection of Unused Pins
p.66	Modification of Figure 3-11 Peripheral I/O Area
p.96	Modification of Table 4-3 VMBENZ3 to VMBENZ0 Signals
p.104	Modification of Figure 4-14 (f) 16-bit bus (4-word sequential transfer, data access)
p.114	Addition of bus priority in 4.9.6 Bus master transition
p.124	Modification of Table 5-1 Setting of Setup Wait, VPSTB Wait Lengths at Each Operation Frequency
p.131	Modification of Figure 5-14 (b) Example of timing of halfword-data write to NPB peripheral macro (programmable peripheral I/O area)
p.134	Modification of Figure 5-15 NPB Write Timing (Example of Timing of Data Write to CSC0 and CSC1 Registers)
p.137	Addition of Caution 2 in 6.2.1 Power save control register (PSC)
p.141	Modification of 6.4 (2) (a) Cancellation by interrupt request
p.143	Modification of Remark in 6.5 (1) Setting and operation status
p.148	Modification of Figure 6-6 Hardware STOP Mode Set/Cancel Timing Example
p.171	Modification of 7.8.4 Block transfer mode
p.174	Modification of 7.9.2 Flyby transfer
pp.176 to 179, 181, 183, 185, 187, 189, 191, 193, 195, 197	Modification of VMSEQ2 to VMSEQ0, VMSIZE1, and VMSIZE0 timing in Figures 7-27 to 7-38
p.229	Modification of 8.7 Periods When Interrupts Cannot Be Acknowledged
p.231	Addition of Caution in 9.1.2 (2) (a) Unit test mode
p.231	Modification of 9.1.3 BUNRIOUT pin
p.232	Modification of Figure 9-1 Peripheral Macro Connection Example
p.233	Modification of 9.4 (2) Test mode pins
p.238	Modification of 10.2.2 (5) TEST
p.240	Modification of Figure 10-1 NB85E901 and NU85E Connection Example

Facsimile Message

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

From:

Name

Company

Tel.

FAX

Address

Thank you for your kind support.

North America

NEC Electronics Inc.
Corporate Communications Dept.
Fax: +1-800-729-9288
+1-408-588-6130

Hong Kong, Philippines, Oceania

NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Asian Nations except Philippines

NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Europe

NEC Electronics (Europe) GmbH
Market Communication Dept.
Fax: +49-211-6503-274

Korea

NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: +82-2-528-4411

Japan

NEC Semiconductor Technical Hotline
Fax: +81- 44-435-9608

South America

NEC do Brasil S.A.
Fax: +55-11-6462-6829

Taiwan

NEC Electronics Taiwan Ltd.
Fax: +886-2-2719-5951

I would like to report the following error/make the following suggestion:

Document title: _____

Document number: _____

Page number: _____

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Technical Accuracy	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Organization	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>