

# H8S/2338 Series, H8S/2328 Series, H8S/2318 Series

Hardware Manual

— Specifications Common to All Series —

# HITACHI

ADE-602-171

Rev. 1.0

3/1/99

Hitachi, Ltd.



## Cautions






1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

# Organization of H8S/2338 Series, H8S/2328 Series, H8S/2318 Series Hardware Manual

The H8S/2338 Series, H8S/2328 Series, H8S/2318 Series Hardware Manual describes the operation of on-chip functions common to the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, in particular, and gives a detailed description of the related registers. Information specific to individual products, including pin arrangement, I/O ports, MCU operating modes (memory maps), interrupt vectors, bus control, and electrical characteristics, can be found in the Reference Manual for the relevant product.




**For product evaluation information, or comparative specification information for current users of Hitachi products**

For H8S/2338 Series, H8S/2328 Series, H8S/2318 Series specifications

Overview		1.1 Overview
Pin arrangement diagram		1.3 Pin Arrangement
Block diagrams of function modules		Section 6 Peripheral Block Diagrams
Pin functions		1.5 Pin Functions
Electrical characteristics		Section 7 Electrical Characteristics

**For detailed information on functions**

For details of the operation of individual modules




I/O port information		Section 5 I/O Ports
Interrupts and exception handling		Section 3 Exception Handling and Interrupt Controller
Pin functions		1.5 Pin Functions

For information on operating modes

List		1.4 Pin Functions in Each Operating Mode
Detailed descriptions		Section 2 MCU Operating Modes

**For use as design material**

For information on registers

List		Section 8 registers
To find a register from its address		8.1 List of Registers (Address Order)
To find register information by function		8.2 List of Registers (By Module)

For information on instructions

List		
Operation description and notes		H8S/2600 Series, H8S/2000 Series Programming Manual
Program examples		

# Preface

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are series of high-performance microcontrollers with a 32-bit H8S/2000 CPU core, and a set of on-chip supporting functions required for system configuration.

The H8S/2000 CPU can execute basic instructions in one state, and is provided with sixteen 16-bit general registers with a 32-bit internal configuration, and a concise and optimized instruction set. The CPU can handle a 16-Mbyte linear address space (architecturally 4 Gbytes). Programs based on the high-level language C can also be run efficiently.

The address space is divided into eight areas. The data bus width and access states can be selected for each of these areas, and various kinds of memory can be connected fast and easily.

Single-power-supply flash memory (F-ZTAT™) and mask ROM versions are available, providing a quick and flexible response to conditions from ramp-up through full-scale volume production, even for applications with frequently changing specifications.

On-chip supporting functions include a 16-bit timer pulse unit (TPU), programmable pulse generator (PPG), 8-bit timer, watchdog timer (WDT), serial communication interface (SCI), A/D converter, D/A converter, and I/O ports.

In addition, an on-chip DMA controller (DMAC) and data transfer controller (DTC) are provided, enabling high-speed data transfer without CPU intervention.

Use of the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series enables easy implementation of compact, high-performance systems capable of processing large volumes of data.

This manual describes the hardware of the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, covering features, specifications, registers and operation. It should be used in conjunction with the H8S/2338 Series Reference Manual, H8S/2328 Series Reference Manual, and H8S/2318 Series Reference Manual which contain information relating to the H8S/2338 Series and H8S/2328 Series hardware—including specifications and register descriptions—required for system design.

Refer to the H8S/2600 Series, H8S/2000 Series Programming Manual for a detailed description of the instruction set and programming-related information.

# Contents

Section 1	Overview .....	1
1.1	Overview.....	1
Section 2	Exception Handling.....	7
2.1	Overview.....	7
2.1.1	Exception Handling Types and Priority .....	7
2.1.2	Exception Handling Operation .....	8
2.1.3	Exception Vector Table.....	8
2.2	Reset .....	10
2.2.1	Overview .....	10
2.2.2	Reset Sequence .....	10
2.2.3	Interrupts after Reset .....	11
2.2.4	State of On-Chip Supporting Modules after Reset Release .....	11
2.3	Traces .....	12
2.4	Interrupts.....	13
2.5	Trap Instruction .....	14
2.6	Stack Status after Exception Handling .....	14
2.7	Notes on Use of the Stack.....	15
Section 3	Interrupt Controller .....	17
3.1	Overview.....	17
3.1.1	Features.....	17
3.1.2	Block Diagram.....	18
3.1.3	Pin Configuration.....	19
3.1.4	Register Configuration.....	19
3.2	Register Descriptions.....	20
3.2.1	System Control Register (SYSCR) .....	20
3.2.2	Interrupt Priority Registers A to K (IPRA to IPRK).....	21
3.2.3	IRQ Enable Register (IER) .....	22
3.2.4	IRQ Sense Control Registers H and L (ISCRH, ISCRL).....	23
3.2.5	IRQ Status Register (ISR).....	24
3.3	Interrupt Sources.....	25
3.3.1	External Interrupts.....	25
3.3.2	Internal Interrupts .....	26
3.3.3	Interrupt Exception Vector Table.....	26
3.4	Interrupt Operation .....	30
3.4.1	Interrupt Control Modes and Interrupt Operation .....	30
3.4.2	Interrupt Control Mode 0.....	33
3.4.3	Interrupt Control Mode 2.....	35

3.4.4	Interrupt Exception Handling Sequence .....	37
3.4.5	Interrupt Response Times .....	39
3.5	Usage Notes .....	40
3.5.1	Contention between Interrupt Generation and Disabling .....	40
3.5.2	Instructions that Disable Interrupts .....	41
3.5.3	Times when Interrupts are Disabled .....	41
3.5.4	Interrupts during Execution of EEPMOV Instruction .....	41
3.6	DTC and DMAC Activation by Interrupt .....	42
3.6.1	Overview .....	42
3.6.2	Block Diagram .....	42
3.6.3	Operation .....	43
<b>Section 4 Bus Controller .....</b>		<b>45</b>
4.1	Overview .....	45
4.1.1	Features .....	45
4.1.2	Block Diagram .....	47
4.1.3	Pin Configuration .....	48
4.1.4	Register Configuration .....	49
4.2	Register Descriptions .....	50
4.2.1	Bus Width Control Register (ABWCR) .....	50
4.2.2	Access State Control Register (ASTCR) .....	51
4.2.3	Wait Control Registers H and L (WCRH, WCRL) .....	52
4.2.4	Bus Control Register H (BCRH) .....	55
4.2.5	Bus Control Register L (BCRL) .....	57
4.2.6	Memory Control Register (MCR) .....	60
4.2.7	DRAM Control Register (DRAMCR) .....	62
4.2.8	Refresh Timer Counter (RTCNT) .....	64
4.2.9	Refresh Time Constant Register (RTCOR) .....	64
4.3	Overview of Bus Control .....	65
4.3.1	Area Partitioning .....	65
4.3.2	Bus Specifications .....	66
4.3.3	Memory Interfaces .....	67
4.3.4	Advanced Mode .....	68
4.3.5	Chip Select Signals .....	69
4.4	Basic Bus Interface .....	70
4.4.1	Overview .....	70
4.4.2	Data Size and Data Alignment .....	70
4.4.3	Valid Strokes .....	72
4.4.4	Basic Timing .....	73
4.4.5	Wait Control .....	81
4.5	DRAM Interface .....	83
4.5.1	Overview .....	83
4.5.2	Setting DRAM Space .....	83

4.5.3	Address Multiplexing .....	83
4.5.4	Data Bus .....	84
4.5.5	Pins Used for DRAM Interface .....	84
4.5.6	Basic Timing .....	85
4.5.7	Precharge State Control .....	86
4.5.8	Wait Control .....	87
4.5.9	Byte Access Control .....	89
4.5.10	Burst Operation .....	91
4.5.11	Refresh Control .....	94
4.6	DMAC Single Address Mode and DRAM Interface .....	97
4.6.1	When DDS = 1 .....	97
4.6.2	When DDS = 0 .....	98
4.7	Burst ROM Interface .....	99
4.7.1	Overview .....	99
4.7.2	Basic Timing .....	99
4.7.3	Wait Control .....	101
4.8	Idle Cycle.....	102
4.8.1	Operation.....	102
4.8.2	Pin States in Idle Cycle .....	106
4.9	Write Data Buffer Function .....	107
4.10	Bus Release.....	108
4.10.1	Overview .....	108
4.10.2	Operation.....	108
4.10.3	Pin States in External Bus Released State.....	109
4.10.4	Transition Timing.....	110
4.10.5	Usage Note .....	111
4.11	Bus Arbitration .....	111
4.11.1	Overview .....	111
4.11.2	Operation.....	111
4.11.3	Bus Transfer Timing .....	112
4.11.4	External Bus Release Usage Note.....	112
4.12	Resets and the Bus Controller.....	112
 Section 5 DMA Controller.....		113
5.1	Overview .....	113
5.1.1	Features .....	113
5.1.2	Block Diagram.....	114
5.1.3	Overview of Functions .....	115
5.1.4	Pin Configuration .....	117
5.1.5	Register Configuration .....	118
5.2	Register Descriptions (1) (Short Address Mode) .....	119
5.2.1	Memory Address Registers (MAR).....	120
5.2.2	I/O Address Register (IOAR).....	121

5.2.3	Execute Transfer Count Register (ETCR).....	121
5.2.4	DMA Control Register (DMACR).....	122
5.2.5	DMA Band Control Register (DMABCR).....	126
5.3	Register Descriptions (2) (Full Address Mode) .....	131
5.3.1	Memory Address Register (MAR) .....	131
5.3.2	I/O Address Register (IOAR) .....	131
5.3.3	Execute Transfer Count Register (ETCR).....	132
5.3.4	DMA Control Register (DMACR).....	133
5.3.5	DMA Band Control Register (DMABCR).....	137
5.4	Register Descriptions (3) .....	142
5.4.1	DMA Write Enable Register (DMAWER) .....	142
5.4.2	DMA Terminal Control Register (DMATCR).....	144
5.4.3	Module Stop Control Register (MSTPCR) .....	145
5.5	Operation .....	146
5.5.1	Transfer Modes .....	146
5.5.2	Sequential Mode .....	148
5.5.3	Idle Mode.....	151
5.5.4	Repeat Mode .....	154
5.5.5	Single Address Mode .....	158
5.5.6	Normal Mode.....	161
5.5.7	Block Transfer Mode.....	164
5.5.8	DMAC Activation Sources .....	170
5.5.9	Basic DMAC Bus Cycles .....	173
5.5.10	DMAC Bus Cycles (Dual Address Mode) .....	174
5.5.11	DMAC Bus Cycles (Single Address Mode) .....	182
5.5.12	Write Data Buffer Function.....	188
5.5.13	DMAC Multi-Channel Operation .....	189
5.5.14	Relation Between the DMAC and External Bus Requests, Refresh Cycles, and the DTC .....	190
5.5.15	NMI Interrupts and DMAC.....	191
5.5.16	Forced Termination of DMAC Operation.....	192
5.5.17	Clearing Full Address Mode .....	193
5.6	Interrupts.....	194
5.7	Usage Notes .....	195
Section 6	Data Transfer Controller .....	199
6.1	Overview.....	199
6.1.1	Features .....	199
6.1.2	Block Diagram.....	200
6.1.3	Register Configuration .....	201
6.2	Register Descriptions.....	202
6.2.1	DTC Mode Register A (MRA).....	202
6.2.2	DTC Mode Register B (MRB) .....	204



6.2.3	DTC Source Address Register (SAR) .....	205
6.2.4	DTC Destination Address Register (DAR) .....	205
6.2.5	DTC Transfer Count Register A (CRA) .....	205
6.2.6	DTC Transfer Count Register B (CRB) .....	206
6.2.7	DTC Enable Registers (DTCER) .....	206
6.2.8	DTC Vector Register (DTVECR) .....	207
6.2.9	Module Stop Control Register (MSTPCR) .....	208
6.3	Operation .....	209
6.3.1	Overview .....	209
6.3.2	Activation Sources .....	213
6.3.3	DTC Vector Table .....	214
6.3.4	Location of Register Information in Address Space .....	217
6.3.5	Normal Mode .....	218
6.3.6	Repeat Mode .....	219
6.3.7	Block Transfer Mode .....	220
6.3.8	Chain Transfer .....	222
6.3.9	Operation Timing .....	223
6.3.10	Number of DTC Execution States .....	224
6.3.11	Procedures for Using DTC .....	226
6.3.12	Examples of Use of the DTC .....	227
6.4	Interrupts .....	232
6.5	Usage Notes .....	232
<b>Section 7 16-Bit Timer Pulse Unit (TPU) .....</b>		<b>233</b>
7.1	Overview .....	233
7.1.1	Features .....	233
7.1.2	Block Diagram .....	237
7.1.3	Pin Configuration .....	238
7.1.4	Register Configuration .....	240
7.2	Register Descriptions .....	242
7.2.1	Timer Control Registers (TCR) .....	242
7.2.2	Timer Mode Registers (TMDR) .....	247
7.2.3	Timer I/O Control Registers (TIOR) .....	249
7.2.4	Timer Interrupt Enable Registers (TIER) .....	262
7.2.5	Timer Status Registers (TSR) .....	264
7.2.6	Timer Counters (TCNT) .....	268
7.2.7	Timer General Registers (TGR) .....	268
7.2.8	Timer Start Register (TSTR) .....	269
7.2.9	Timer Synchro Register (TSYR) .....	269
7.2.10	Module Stop Control Register (MSTPCR) .....	270
7.3	Interface to Bus Master .....	271
7.3.1	16-Bit Registers .....	271
7.3.2	8-Bit Registers .....	271

7.4	Operation .....	273
7.4.1	Overview .....	273
7.4.2	Basic Functions .....	274
7.4.3	Synchronous Operation .....	280
7.4.4	Buffer Operation .....	282
7.4.5	Cascaded Operation.....	286
7.4.6	PWM Modes .....	288
7.4.7	Phase Counting Mode .....	293
7.5	Interrupts.....	300
7.5.1	Interrupt Sources and Priorities.....	300
7.5.2	DTC/DMAC Activation .....	302
7.5.3	A/D Converter Activation .....	302
7.6	Operation Timing .....	303
7.6.1	Input/Output Timing .....	303
7.6.2	Interrupt Signal Timing.....	307
7.7	Usage Notes .....	311
<b>Section 8 Programmable Pulse Generator (PPG) .....</b>		<b>321</b>
8.1	Overview.....	321
8.1.1	Features .....	321
8.1.2	Block Diagram.....	322
8.1.3	Pin Configuration .....	323
8.1.4	Registers .....	324
8.2	Register Descriptions.....	325
8.2.1	Next Data Enable Registers H and L (NDERH, NDERL).....	325
8.2.2	Output Data Registers H and L (PODRH, PODRL).....	326
8.2.3	Next Data Registers H and L (NDRH, NDRL).....	327
8.2.4	Notes on NDR Access.....	327
8.2.5	PPG Output Control Register (PCR).....	329
8.2.6	PPG Output Mode Register (PMR).....	331
8.2.7	Port 1 Data Direction Register (P1DDR) .....	333
8.2.8	Port 2 Data Direction Register (P2DDR) .....	334
8.2.9	Module Stop Control Register (MSTPCR) .....	334
8.3	Operation .....	335
8.3.1	Overview .....	335
8.3.2	Output Timing .....	336
8.3.3	Normal Pulse Output.....	337
8.3.4	Non-Overlapping Pulse Output.....	339
8.3.5	Inverted Pulse Output .....	342
8.3.6	Pulse Output Triggered by Input Capture .....	343
8.4	Usage Notes .....	344
8.4.1	Operation of Pulse Output Pins .....	344
8.4.2	Note on Non-Overlapping Output.....	344

Section 9	8-Bit Timers .....	347
9.1	Overview.....	347
9.1.1	Features .....	347
9.1.2	Block Diagram.....	348
9.1.3	Pin Configuration .....	349
9.1.4	Register Configuration .....	349
9.2	Register Descriptions.....	350
9.2.1	Timer Counters 0 and 1 (TCNT0, TCNT1).....	350
9.2.2	Time Constant Registers A0 and A1 (TCORA0, TCORA1) .....	350
9.2.3	Time Constant Registers B0 and B1 (TCORB0, TCORB1).....	351
9.2.4	Time Control Registers 0 and 1 (TCR0, TCR1) .....	351
9.2.5	Timer Control/Status Registers 0 and 1 (TCSR0, TCSR1).....	353
9.2.6	Module Stop Control Register (MSTPCR) .....	356
9.3	Operation .....	357
9.3.1	TCNT Incrementation Timing.....	357
9.3.2	Compare Match Timing .....	358
9.3.3	Timing of TCNT External Reset.....	360
9.3.4	Timing of Overflow Flag (OVF) Setting.....	360
9.3.5	Operation with Cascaded Connection .....	361
9.4	Interrupts.....	362
9.4.1	Interrupt Sources and DTC Activation.....	362
9.4.2	A/D Converter Activation .....	362
9.5	Sample Application .....	363
9.6	Usage Notes .....	364
9.6.1	Contention between TCNT Write and Clear .....	364
9.6.2	Contention between TCNT Write and Increment .....	365
9.6.3	Contention between TCOR Write and Compare Match .....	366
9.6.4	Contention between Compare Matches A and B .....	367
9.6.5	Switching of Internal Clocks and TCNT Operation.....	367
9.6.6	Interrupts and Module Stop Mode.....	369
Section 10	Watchdog Timer .....	371
10.1	Overview.....	371
10.1.1	Features .....	371
10.1.2	Block Diagram.....	372
10.1.3	Pin Configuration .....	373
10.1.4	Register Configuration .....	373
10.2	Register Descriptions.....	374
10.2.1	Timer Counter (TCNT) .....	374
10.2.2	Timer Control/Status Register (TCSR) .....	374
10.2.3	Reset Control/Status Register (RSTCSR).....	376
10.2.4	Notes on Register Access .....	377
10.3	Operation .....	379

10.3.1	Operation in Watchdog Timer Mode .....	379
10.3.2	Operation in Interval Timer Mode .....	381
10.3.3	Timing of Overflow Flag (OVF) Setting.....	381
10.3.4	Timing of Watchdog Timer Overflow Flag (WOVF) Setting.....	382
10.4	Interrupts.....	382
10.5	Usage Notes .....	382
10.5.1	Contention between Timer Counter (TCNT) Write and Increment .....	382
10.5.2	Changing Value of CKS2 to CKS0.....	383
10.5.3	Switching between Watchdog Timer Mode and Interval Timer Mode.....	383
10.5.4	System Reset by $\overline{\text{WDTOVF}}$ Signal* .....	384
10.5.5	Internal Reset in Watchdog Timer Mode .....	384
<b>Section 11 Serial Communication Interface (SCI) .....</b>		<b>385</b>
11.1	Overview.....	385
11.1.1	Features .....	385
11.1.2	Block Diagram.....	387
11.1.3	Pin Configuration .....	388
11.1.4	Register Configuration .....	389
11.2	Register Descriptions.....	390
11.2.1	Receive Shift Register (RSR) .....	390
11.2.2	Receive Data Register (RDR) .....	390
11.2.3	Transmit Shift Register (TSR).....	391
11.2.4	Transmit Data Register (TDR) .....	391
11.2.5	Serial Mode Register (SMR).....	392
11.2.6	Serial Control Register (SCR).....	395
11.2.7	Serial Status Register (SSR).....	399
11.2.8	Bit Rate Register (BRR).....	402
11.2.9	Smart Card Mode Register (SCMR) .....	410
11.2.10	Module Stop Control Register (MSTPCR) .....	412
11.3	Operation .....	413
11.3.1	Overview .....	413
11.3.2	Operation in Asynchronous Mode.....	415
11.3.3	Multiprocessor Communication Function.....	426
11.3.4	Operation in Synchronous Mode.....	434
11.4	SCI Interrupts .....	443
11.5	Usage Notes .....	445
<b>Section 12 Smart Card Interface.....</b>		<b>453</b>
12.1	Overview.....	453
12.1.1	Features .....	453
12.1.2	Block Diagram.....	454
12.1.3	Pin Configuration .....	455
12.1.4	Register Configuration .....	456

12.2	Register Descriptions.....	457
12.2.1	Smart Card Mode Register (SCMR).....	457
12.2.2	Serial Status Register (SSR).....	458
12.2.3	Serial Mode Register (SMR).....	459
12.2.4	Serial Control Register (SCR).....	461
12.3	Operation.....	462
12.3.1	Overview.....	462
12.3.2	Pin Connections.....	462
12.3.3	Data Format.....	464
12.3.4	Register Settings.....	466
12.3.5	Clock.....	468
12.3.6	Data Transfer Operations.....	470
12.3.7	Operation in GSM Mode.....	477
12.3.8	Operation in Block Transfer Mode.....	478
12.4	Usage Notes.....	479
Section 13 A/D Converter (8 Analog Input Channel Version).....		483
13.1	Overview.....	483
13.1.1	Features.....	483
13.1.2	Block Diagram.....	485
13.1.3	Pin Configuration.....	486
13.1.4	Register Configuration.....	487
13.2	Register Descriptions.....	488
13.2.1	A/D Data Registers A to D (ADDRA to ADDR D).....	488
13.2.2	A/D Control/Status Register (ADCSR).....	489
13.2.3	A/D Control Register (ADCR).....	491
13.2.4	Module Stop Control Register (MSTPCR).....	492
13.3	Interface to Bus Master.....	493
13.4	Operation.....	494
13.4.1	Single Mode (SCAN = 0).....	494
13.4.2	Scan Mode (SCAN = 1).....	496
13.4.3	Input Sampling and A/D Conversion Time.....	498
13.4.4	External Trigger Input Timing.....	499
13.5	Interrupts.....	500
13.6	Usage Notes.....	501
Section 14 A/D Converter (12 Analog Input Channel Version).....		507
14.1	Overview.....	507
14.1.1	Features.....	507
14.1.2	Block Diagram.....	509
14.1.3	Pin Configuration.....	510
14.1.4	Register Configuration.....	511
14.2	Register Descriptions.....	512

14.2.1	A/D Data Registers A to D (ADDRA to ADDR D)	512
14.2.2	A/D Control/Status Register (ADCSR)	513
14.2.3	A/D Control Register (ADCR)	515
14.2.4	Module Stop Control Register (MSTPCR)	516
14.3	Interface to Bus Master	517
14.4	Operation	518
14.4.1	Single Mode (SCAN = 0)	518
14.4.2	Scan Mode (SCAN = 1)	520
14.4.3	Input Sampling and A/D Conversion Time	522
14.4.4	External Trigger Input Timing	523
14.5	Interrupts	524
14.6	Usage Notes	525
<b>Section 15 D/A Converter</b>		<b>527</b>
15.1	Overview	527
15.1.1	Features	527
15.1.2	Block Diagram	528
15.1.3	Pin Configuration	529
15.1.4	Register Configuration	529
15.2	Register Descriptions	530
15.2.1	D/A Data Registers 0 to 3 (DADR0 to DADR3)	530
15.2.2	D/A Control Registers 01 and 23 (DACR01, DACR23)	530
15.2.3	Module Stop Control Register (MSTPCR)	532
15.3	Operation	533
<b>Section 16 RAM</b>		<b>535</b>
16.1	Overview	535
16.1.1	Block Diagram	535
16.1.2	Register Configuration	536
16.2	Register Descriptions	536
16.2.1	System Control Register (SYSCR)	536
16.3	Operation	537
16.4	Usage Note	537
<b>Section 17 ROM</b>		<b>539</b>
17.1	Overview	539
17.1.1	Block Diagram	539
17.1.2	Register Configuration	540
17.2	Register Descriptions	540
17.2.1	Mode Control Register (MDCR)	540
17.2.2	Bus Control Register L (BCRL)	541
17.3	Operation	541
17.4	Overview of Flash Memory	544

17.4.1	Features .....	544
17.4.2	Overview .....	545
17.4.3	Flash Memory Operating Modes.....	546
17.4.4	On-Board Programming Modes .....	547
17.4.5	Flash Memory Emulation in RAM.....	549
17.4.6	Differences between Boot Mode and User Program Mode.....	550
17.4.7	Block Configuration .....	551
17.4.8	Pin Configuration .....	552
17.4.9	Register Configuration .....	553
17.5	Register Descriptions.....	554
17.5.1	Flash Memory Control Register 1 (FLMCR1).....	554
17.5.2	Flash Memory Control Register 2 (FLMCR2).....	557
17.5.3	Erase Block Register 1 (EBR1).....	558
17.5.4	Erase Block Registers 2 (EBR2).....	558
17.5.5	System Control Register 2 (SYSCR2) .....	559
17.5.6	RAM Emulation Register (RAMER) .....	560
17.6	On-Board Programming Modes .....	561
17.6.1	Boot Mode.....	562
17.6.2	User Program Mode .....	566
17.7	Programming/Erasing Flash Memory.....	568
17.7.1	Program Mode.....	568
17.7.2	Program-Verify Mode .....	569
17.7.3	Erase Mode.....	571
17.7.4	Erase-Verify Mode .....	571
17.8	Flash Memory Protection .....	573
17.8.1	Hardware Protection .....	573
17.8.2	Software Protection .....	573
17.8.3	Error Protection .....	574
17.9	Flash Memory Emulation in RAM.....	576
17.9.1	Emulation in RAM .....	576
17.9.2	RAM Overlap .....	577
17.10	Interrupt Handling when Programming/Erasing Flash Memory .....	578
17.11	Flash Memory PROM Mode .....	579
17.11.1	PROM Mode Setting.....	579
17.11.2	Socket Adapters and Memory Map.....	580
17.11.3	PROM Mode Operation .....	583
17.11.4	Memory Read Mode.....	584
17.11.5	Auto-Program Mode .....	587
17.11.6	Auto-Erase Mode.....	589
17.11.7	Status Read Mode.....	590
17.11.8	Status Polling.....	591
17.11.9	PROM Mode Transition Time.....	592
17.11.10	Notes On Memory Programming .....	592

17.12	Flash Memory Programming and Erasing Precautions .....	593
<b>Section 18 Clock Pulse Generator .....</b>		
18.1	Overview.....	599
18.1.1	Block Diagram.....	599
18.1.2	Register Configuration .....	600
18.2	Register Descriptions.....	600
18.2.1	System Clock Control Register (SCKCR) .....	600
18.3	Oscillator.....	602
18.3.1	Connecting a Crystal Resonator .....	602
18.3.2	External Clock Input .....	604
18.4	Duty Adjustment Circuit.....	606
18.5	Medium-Speed Clock Divider.....	606
18.6	Bus Master Clock Selection Circuit .....	606
<b>Section 19 Power-Down Modes .....</b>		
19.1	Overview.....	607
19.1.1	Register Configuration .....	608
19.2	Register Descriptions.....	609
19.2.1	Standby Control Register (SBYCR) .....	609
19.2.2	System Clock Control Register (SCKCR) .....	611
19.2.3	Module Stop Control Register (MSTPCR) .....	613
19.3	Medium-Speed Mode .....	614
19.4	Sleep Mode.....	615
19.5	Module Stop Mode .....	615
19.5.1	Module Stop Mode .....	615
19.5.2	Usage Notes.....	616
19.6	Software Standby Mode .....	617
19.6.1	Software Standby Mode .....	617
19.6.2	Clearing Software Standby Mode .....	617
19.6.3	Setting Oscillation Stabilization Time after Clearing Software Standby Mode ..	618
19.6.4	Software Standby Mode Application Example .....	618
19.6.5	Usage Notes.....	619
19.7	Hardware Standby Mode .....	620
19.7.1	Hardware Standby Mode.....	620
19.7.2	Hardware Standby Mode Timing .....	620
19.8	∅ Clock Output Disabling Function.....	621
<b>Appendix A Instruction Set.....</b>		
A.1	Instruction List.....	623
A.2	Instruction Codes .....	647
A.3	Operation Code Map.....	662
A.4	Number of States Required for Instruction Execution .....	666



A.5	Bus States During Instruction Execution .....	680
A.6	Condition Code Modification.....	694
Appendix B Internal I/O Registers .....		700
B.1	Addresses.....	700

# Section 1 Overview

## 1.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are series of microcomputers (MCUs: microcomputer units), built around the H8S/2000 CPU, employing Hitachi's proprietary architecture, and equipped with supporting functions on-chip.

The H8S/2000 CPU has an internal 32-bit architecture, is provided with sixteen 16-bit general registers and a concise, optimized instruction set designed for high-speed operation, and can address a 16-Mbyte linear address space. The instruction set is upward-compatible with H8/300 and H8/300H CPU instructions at the object-code level, facilitating migration from the H8/300, H8/300L, or H8/300H Series.

On-chip supporting functions required for system configuration include DMA controller (DMAC) and data transfer controller (DTC) bus masters, ROM and RAM memory, a 16-bit timer-pulse unit (TPU), programmable pulse generator (PPG), 8-bit timer, watchdog timer (WDT), serial communication interface (SCI), A/D converter, D/A converter, and I/O ports.

A high-functionality bus controller is also provided, enabling fast and easy connection of DRAM and other kinds of memory.

Single-power-supply flash memory (F-ZTAT™\*) and mask ROM versions are available, providing a quick and flexible response to conditions from ramp-up through full-scale volume production, even for applications with frequently changing specifications. ROM is connected to the CPU via a 16-bit data bus, enabling both byte and word data to be accessed in one state. Instruction fetching is thus speeded up, and processing speed increased.

The features of the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are shown in table 1-1.

Note: \* F-ZTAT is a trademark of Hitachi, Ltd.

**Table 1-1 Overview**

<b>Item</b>	<b>Specification</b>
CPU	<ul style="list-style-type: none"> <li>• General-register machine               <ul style="list-style-type: none"> <li>— Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)</li> </ul> </li> <li>• High-speed operation suitable for realtime control               <ul style="list-style-type: none"> <li>— Maximum clock rate: 20 MHz (25 MHz version in planning stage)</li> <li>— High-speed arithmetic operations                   <ul style="list-style-type: none"> <li>8/16/32-bit register-register add/subtract: 50 ns (at 20 MHz operation)</li> <li>16 × 16-bit register-register multiply: 1000 ns (at 20 MHz operation)</li> <li>32 ÷ 16-bit register-register divide: 1000 ns (at 20 MHz operation)</li> </ul> </li> </ul> </li> <li>• Instruction set suitable for high-speed operation               <ul style="list-style-type: none"> <li>— Sixty-five basic instructions</li> <li>— 8/16/32-bit move/arithmetic and logic instructions</li> <li>— Unsigned/signed multiply and divide instructions</li> <li>— Powerful bit-manipulation instructions</li> </ul> </li> <li>• CPU operating mode               <ul style="list-style-type: none"> <li>— Advanced mode: 16-Mbyte address space</li> </ul> </li> </ul>
Bus controller	<ul style="list-style-type: none"> <li>• Address space divided into 8 areas, with bus specifications settable independently for each area</li> <li>• Chip select output possible for each area</li> <li>• Choice of 8-bit or 16-bit access space for each area</li> <li>• 2-state or 3-state access space can be designated for each area</li> <li>• Number of program wait states can be set for each area</li> <li>• Burst ROM directly connectable</li> <li>• Maximum 8-Mbyte DRAM directly connectable (or use of interval timer possible)</li> <li>• External bus release function</li> </ul>
DMA controller (DMAC)	<ul style="list-style-type: none"> <li>• Choice of short address mode or full address mode</li> <li>• 4 channels in short address mode</li> <li>• 2 channels in full address mode</li> <li>• Transfer possible in repeat mode, block transfer mode, etc.</li> <li>• Single address mode transfer possible</li> <li>• Can be activated by internal interrupt</li> </ul>
Data transfer controller (DTC)	<ul style="list-style-type: none"> <li>• Can be activated by internal interrupt or software</li> <li>• Multiple transfers or multiple types of transfer possible for one activation source</li> <li>• Transfer possible in repeat mode, block transfer mode, etc.</li> <li>• Request can be sent to CPU for interrupt that activated DTC</li> </ul>

**Table 1-1 Overview (cont)**

<b>Item</b>	<b>Specification</b>
16-bit timer-pulse unit (TPU)	<ul style="list-style-type: none"><li>• 6-channel 16-bit timer</li><li>• Pulse I/O processing capability for up to 16 pins</li><li>• Automatic 2-phase encoder count capability</li></ul>
Programmable pulse generator (PPG)	<ul style="list-style-type: none"><li>• Maximum 16-bit pulse output possible with TPU as time base</li><li>• Output trigger selectable in 4-bit groups</li><li>• Non-overlap margin can be set</li><li>• Direct output or inverse output setting possible</li></ul>
8-bit timer, 2 channels	<ul style="list-style-type: none"><li>• 8-bit up-counter (external event count capability)</li><li>• Two time constant registers</li><li>• Two-channel connection possible</li></ul>
Watchdog timer	<ul style="list-style-type: none"><li>• Watchdog timer or interval timer selectable</li></ul>
Serial communication interface (SCI), 3 channels	<ul style="list-style-type: none"><li>• Asynchronous mode or synchronous mode selectable</li><li>• Multiprocessor communication function</li><li>• Smart card interface function</li></ul>
A/D converter	<ul style="list-style-type: none"><li>• Resolution: 10 bits</li><li>• Input: 8 or 12 channels</li><li>• High-speed conversion: 6.7 <math>\mu</math>s minimum conversion time (at 20 MHz operation)</li><li>• Single or scan mode selectable</li><li>• Sample-and-hold circuit</li><li>• A/D conversion can be activated by external trigger or timer trigger</li></ul>
D/A converter	<ul style="list-style-type: none"><li>• Resolution: 8 bits</li><li>• Output: 2 to 4 channels</li></ul>
Memory	<ul style="list-style-type: none"><li>• Flash memory, mask ROM</li><li>• High-speed static RAM</li></ul>
Interrupt controller	<ul style="list-style-type: none"><li>• Eight priority levels settable</li></ul>
Power-down state	<ul style="list-style-type: none"><li>• Medium-speed mode</li><li>• Sleep mode</li><li>• Module stop mode</li><li>• Software standby mode</li><li>• Hardware standby mode</li><li>• Variable clock division ratio</li></ul>

**Table 1-1 Overview (cont)**

Item	Specification					
Operating modes	• Eight MCU operating modes (F-ZTAT version)					
		<b>CPU</b>			<b>External Data Bus</b>	
	<b>Mode</b>	<b>Operating Mode</b>	<b>Description</b>	<b>On-Chip ROM</b>	<b>Initial Value</b>	<b>Maximum Value</b>
	0	—	—	—	—	—
	1	—	—	—	—	—
	2	—	—	—	—	—
	3	—	—	—	—	—
	4	Advanced	On-chip ROM disabled expansion mode	Disabled	16 bits	16 bits
	5	—	—	—	8 bits	16 bits
	6	—	On-chip ROM enabled expansion mode	Enabled	8 bits	16 bits
	7	—	Single-chip mode	—	—	—
	8	—	—	—	—	—
	9	—	—	—	—	—
	10	Advanced	Boot mode	Enabled	8 bits	16 bits
	11	—	—	—	—	—
	12	—	—	—	—	—
	13	—	—	—	—	—
	14	Advanced	User program mode	Enabled	8 bits	16 bits
	15	—	—	—	—	—

**Table 1-1 Overview (cont)**

Item	Specification					
	CPU			External Data Bus		
	Operating Mode	Description	On-Chip ROM	Initial Value	Maximum Value	
Operating modes	• Four MCU operating modes (ROMless and mask ROM versions)					
	0	—	—	—	—	
	1					
	2					
	3					
	4*	Advanced	On-chip ROM disabled expansion mode	Disabled	16 bits	16 bits
	5*		On-chip ROM disabled expansion mode	Disabled	8 bits	16 bits
	6		On-chip ROM enabled expansion mode	Enabled	8 bits	16 bits
	7		Single-chip mode	Enabled	—	—
	Note: * The ROMless version can use Mode 4 and 5.					
Clock pulse generator	• Built-in duty correction circuit					

# Section 2 Exception Handling

## 2.1 Overview

### 2.1.1 Exception Handling Types and Priority

As table 2-1 indicates, exception handling may be caused by a reset, trap instruction, or interrupt. Exception handling is prioritized as shown in table 2-1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Trap instruction exceptions are accepted at all times in the program execution state.

Exception handling sources, the stack structure, and the operation of the CPU vary depending on the interrupt control mode set by the INTM0 and INTM1 bits of SYSCR.

**Table 2-1 Exception Types and Priority**

Priority	Exception Type	Start of Exception Handling
High	Reset	Starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows.
	Trace* <sup>1</sup>	Starts when execution of the current instruction or exception handling ends, if the trace (T) bit is set to 1
	Interrupt	Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued* <sup>2</sup>
Low	Trap instruction (TRAPA)* <sup>3</sup>	Started by execution of a trap instruction (TRAPA)

- Notes:
1. Traces are enabled only in interrupt control mode 2. Trace exception handling is not executed after execution of an RTE instruction.
  2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
  3. Trap instruction exception handling requests are accepted at all times in the program execution state.

### 2.1.2 Exception Handling Operation

Exceptions originate from various sources. Trap instructions and interrupts are handled as follows:

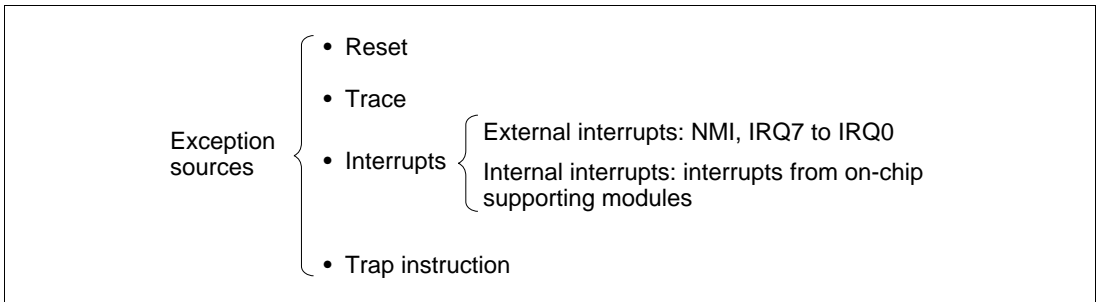
1. The program counter (PC), condition code register (CCR), and extend register (EXR) are pushed onto the stack.
2. The interrupt mask bits are updated. The T bit is cleared to 0.
3. A vector address corresponding to the exception source is generated, and program execution starts from that address.

For a reset exception, steps 2 and 3 above are carried out.

### 2.1.3 Exception Vector Table

The exception sources are classified as shown in figure 2-1. Different vector addresses are assigned to different exception sources.

Table 2-2 lists the exception sources and their vector addresses.



**Figure 2-1 Exception Sources**

In modes 6 and 7, the on-chip ROM available for use after a power-on reset is the 64-kbyte area comprising addresses H'000000 to H'00FFFF. Care is required when setting vector addresses. In this case, clearing the EAE bit in BCRL enables the 256-kbyte\* area comprising addresses H'000000 to H'03FFFF to be used.

Note: \* Depends on the model. See the relevant reference manual for details.



**Table 2-2 Exception Vector Table**

Exception Source		Vector Number	Vector Address* <sup>1</sup>
			Advanced Mode
Power-on reset		0	H'0000 to H'0003
Reserved		1	H'0004 to H'0007
Reserved for system use		2	H'0008 to H'000B
		3	H'000C to H'000F
		4	H'0010 to H'0013
Trace		5	H'0014 to H'0017
Reserved for system use		6	H'0018 to H'001B
External interrupt	NMI	7	H'001C to H'001F
Trap instruction (4 sources)		8	H'0020 to H'0023
		9	H'0024 to H'0027
		10	H'0028 to H'002B
		11	H'002C to H'002F
Reserved for system use		12	H'0030 to H'0033
		13	H'0034 to H'0037
		14	H'0038 to H'003B
		15	H'003C to H'003F
External interrupt	IRQ0	16	H'0040 to H'0043
	IRQ1	17	H'0044 to H'0047
	IRQ2	18	H'0048 to H'004B
	IRQ3	19	H'004C to H'004F
	IRQ4	20	H'0050 to H'0053
	IRQ5	21	H'0054 to H'0057
	IRQ6	22	H'0058 to H'005B
	IRQ7	23	H'005C to H'005F
Internal interrupt* <sup>2</sup>		24	H'0060 to H'0063
		91	H'016C to H'016F

Notes: 1. Lower 16 bits of the address.

2. For details of internal interrupt vectors, see section 3.3.3, Interrupt Exception Vector Table.

## 2.2 Reset

### 2.2.1 Overview

A reset has the highest exception priority.

When the  $\overline{\text{RES}}$  pin goes low, all processing halts and the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip enters the reset state. A reset initializes the internal state of the CPU and the registers of on-chip supporting modules. Immediately after a reset, interrupt control mode 0 is set.

Reset exception handling begins when the  $\overline{\text{RES}}$  pin changes from low to high.

A reset can also be caused by watchdog timer overflow. For details see section 10, Watchdog Timer.

### 2.2.2 Reset Sequence

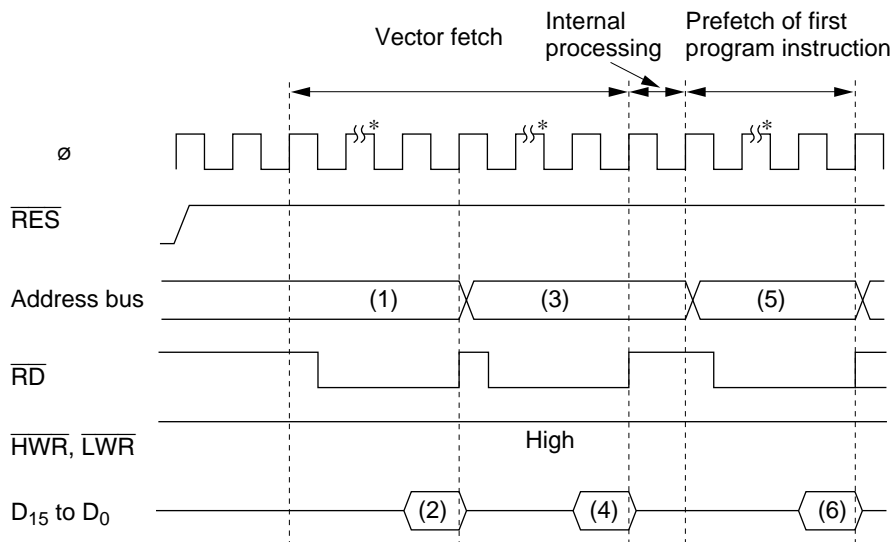
The H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip enters the reset state when the  $\overline{\text{RES}}$  pin goes low.

To ensure that the chip is reset, hold the  $\overline{\text{RES}}$  pin low for at least 20 ms at power-up. To reset the chip during operation, hold the  $\overline{\text{RES}}$  pin low for at least 20 states.

When the  $\overline{\text{RES}}$  pin goes high after being held low for the necessary time, the chip starts reset exception handling as follows:

1. The internal state of the CPU and the registers of the on-chip supporting modules are initialized, the T bit is cleared to 0 in EXR, and the I bit is set to 1 in EXR and CCR.
2. The reset exception vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Figure 2-2 shows an example of the reset sequence.



- (1), (3) Reset exception handling vector address ((1) = H'000000, (3) = H'000002)  
 (2), (4) Start address (contents of reset exception vector address)  
 (5) Start address ((5) = (2), (4))  
 (6) First program instruction

Note: \* 3 program wait states are inserted.

**Figure 2-2 Reset Sequence (Mode 4)**

### 2.2.3 Interrupts after Reset

If an interrupt is accepted after a reset but before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests, including NMI, are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: MOV.L #xx:32, SP).

### 2.2.4 State of On-Chip Supporting Modules after Reset Release

After reset release, MSTPCR is initialized to H'3FFF and all modules except the DMAC and DTC enter module stop mode. Consequently, on-chip supporting module registers cannot be read or written to. Register reading and writing is enabled when module stop mode is exited.

## 2.3 Traces

Traces are enabled in interrupt control mode 2. Trace mode is not activated in interrupt control mode 0, irrespective of the state of the T bit. For details of interrupt control modes, see section 3, Interrupt Controller.

If the T bit in EXR is set to 1, trace mode is activated. In trace mode, a trace exception occurs on completion of each instruction.

Trace mode is canceled by clearing the T bit in EXR to 0. It is not affected by interrupt masking.

Table 2-3 shows the state of CCR and EXR after execution of trace exception handling.

Interrupts are accepted even within the trace exception handling routine.

The T bit saved on the stack retains its value of 1, and when control is returned from the trace exception handling routine by the RTE instruction, trace mode resumes.

Trace exception handling is not carried out after execution of the RTE instruction.

**Table 2-3 Status of CCR and EXR after Trace Exception Handling**

Interrupt Control Mode	CCR		EXR	
	I	UI	I2 to I0	T
0	Trace exception handling cannot be used.			
2	1	—	—	0

### Legend

1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

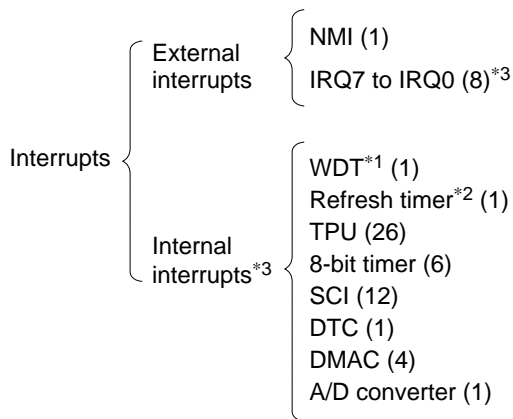
## 2.4 Interrupts

Interrupt exception handling can be requested by nine external sources (NMI, IRQ7 to IRQ0) and 52 internal sources in the on-chip supporting modules. Figure 2-3 classifies the interrupt sources and the number of interrupts of each type.

The on-chip supporting modules that can request interrupts include the watchdog timer (WDT), refresh timer, 16-bit timer-pulse unit (TPU), 8-bit timer, serial communication interface (SCI), data transfer controller (DTC), DMA controller (DMAC), and A/D converter. Each interrupt source has a separate vector address.

NMI is the highest-priority interrupt. Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to eight priority/mask levels to enable multiplexed interrupt control.

For details of interrupts, see section 3, Interrupt Controller.



Notes: Numbers in parentheses are the numbers of interrupt sources.

1. When the watchdog timer is used as an interval timer, it generates an interrupt request at each counter overflow.
2. When the refresh timer is used as an interval timer, it generates an interrupt request at each compare match.
3. The number of external interrupts and the modules provided on-chip differ from model to model; see the reference manual for the relevant model for details.

**Figure 2-3 Interrupt Sources and Number of Interrupts**

## 2.5 Trap Instruction

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state.

The TRAPA instruction fetches a start address from a vector table entry corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 2-4 shows the status of CCR and EXR after execution of trap instruction exception handling.

**Table 2-4 Status of CCR and EXR after Trap Instruction Exception Handling**

Interrupt Control Mode	CCR		EXR	
	I	UI	I2 to I0	T
0	1	—	—	—
2	1	—	—	0

### Legend

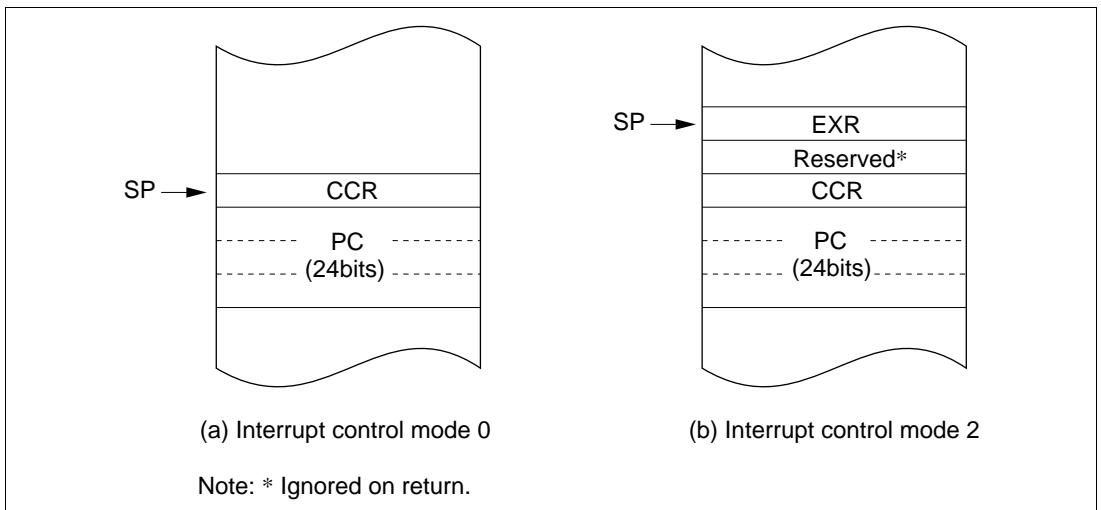
1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

## 2.6 Stack Status after Exception Handling

Figure 2-4 shows the stack after completion of trap instruction exception handling and interrupt exception handling.



**Figure 2-4 Stack Status after Exception Handling (Advanced Modes)**

## 2.7 Notes on Use of the Stack

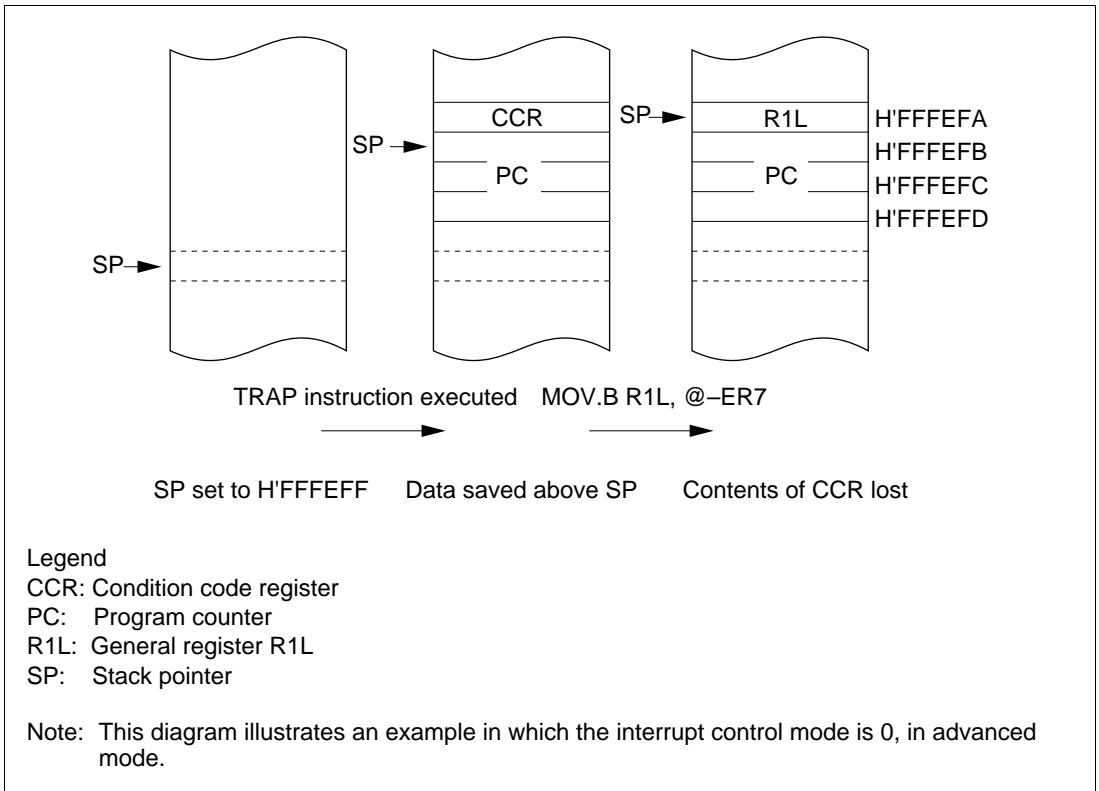
When accessing word data or longword data, the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series assume that the lowest address bit is 0. The stack should always be accessed by word transfer instruction or longword transfer instruction, and the value of the stack pointer (SP, ER7) should always be kept even. Use the following instructions to save registers:

```
PUSH.W  Rn    (or MOV.W Rn, @-SP)
PUSH.L  ERn   (or MOV.L ERn, @-SP)
```

Use the following instructions to restore registers:

```
POP.W   Rn    (or MOV.W @SP+, Rn)
POP.L   ERn   (or MOV.L @SP+, ERn)
```

Setting SP to an odd value may lead to a malfunction. Figure 2-5 shows an example of what happens when the SP value is odd.



**Figure 2-5 Operation when SP Value is Odd**

# Section 3 Interrupt Controller

## 3.1 Overview

### 3.1.1 Features

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series control interrupts by means of an interrupt controller. The interrupt controller has the following features. This chapter assumes the maximum number of interrupt sources available in these series—nine external interrupts and 52 internal interrupts. The number of interrupt sources differs from model to model; see the reference manual for the relevant model for details.

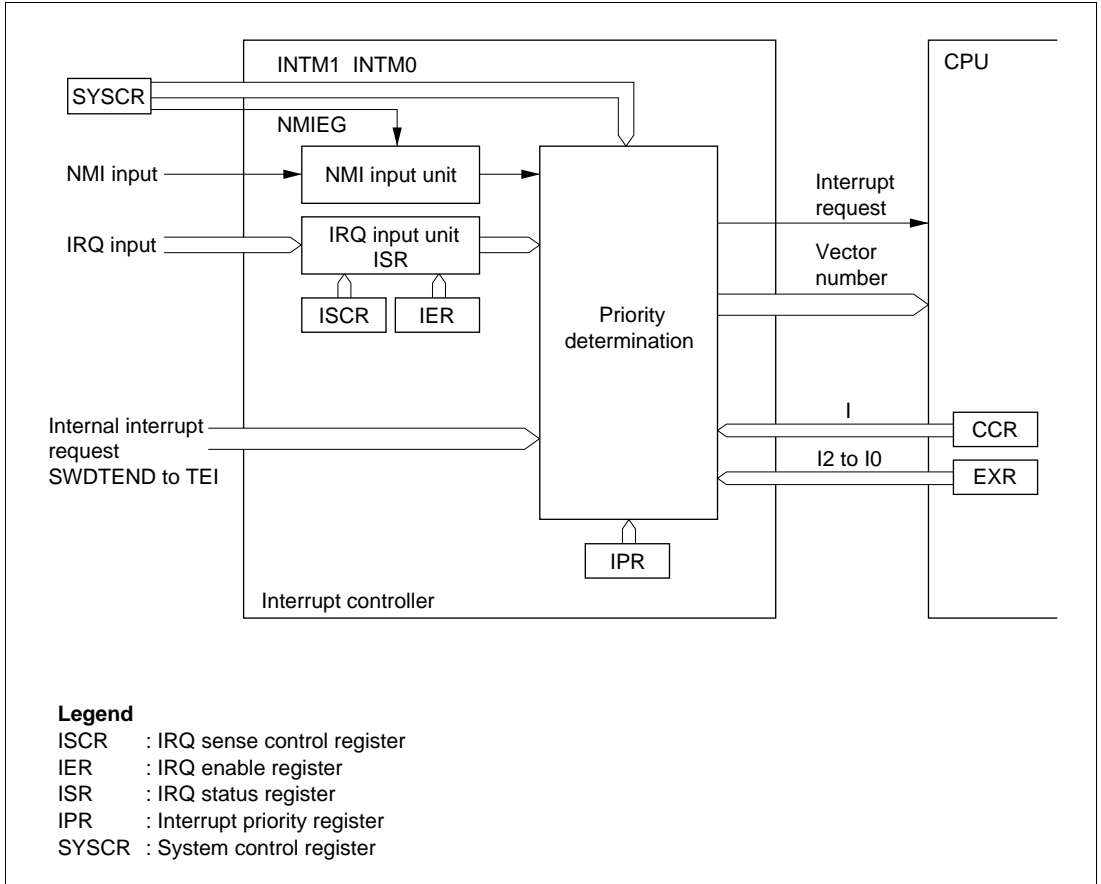
- Two interrupt control modes
  - Either of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR)
- Priorities settable with IPRs
  - Interrupt priority registers (IPRs) are provided for setting interrupt priorities. Eight priority levels can be set for each module for all interrupts except NMI
  - NMI is assigned the highest priority level of 8, and can be accepted at all times
- Independent vector addresses
  - All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine
- Nine external interrupt pins
  - NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge can be selected for NMI
  - Falling edge, rising edge, or both edge detection, or level sensing, can be selected for IRQ7 to IRQ0
- DTC and DMAC control
  - DTC and DMAC\* activation is controlled by means of interrupts

Note: \* Some models do not have an on-chip DMAC; see the reference manual for the relevant model for details.



### 3.1.2 Block Diagram

A block diagram of the interrupt controller is shown in Figure 3-1.



**Figure 3-1 Block Diagram of Interrupt Controller**

### 3.1.3 Pin Configuration

Table 3-1 summarizes the pins of the interrupt controller.

**Table 3-1 Interrupt Controller Pins**

Name	Symbol	I/O	Function
Nonmaskable interrupt	NMI	Input	Nonmaskable external interrupt; rising or falling edge can be selected
External interrupt requests 7 to 0	$\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$	Input	Maskable external interrupts; rising, falling, or both edges, or level sensing, can be selected

### 3.1.4 Register Configuration

Table 3-2 summarizes the registers of the interrupt controller.

**Table 3-2 Interrupt Controller Registers**

Name	Abbreviation	R/W	Initial Value	Address* <sup>1</sup>
System control register	SYSCR	R/W	H'01	H'FF39
IRQ sense control register H	ISCRH	R/W	H'00	H'FF2C
IRQ sense control register L	ISCR L	R/W	H'00	H'FF2D
IRQ enable register	IER	R/W	H'00	H'FF2E
IRQ status register	ISR	R/(W)* <sup>2</sup>	H'00	H'FF2F
Interrupt priority register A	IPRA	R/W	H'77	H'FEC4
Interrupt priority register B	IPRB	R/W	H'77	H'FEC5
Interrupt priority register C	IPRC	R/W	H'77	H'FEC6
Interrupt priority register D	IPRD	R/W	H'77	H'FEC7
Interrupt priority register E	IPRE	R/W	H'77	H'FEC8
Interrupt priority register F	IPRF	R/W	H'77	H'FEC9
Interrupt priority register G	IPRG	R/W	H'77	H'FECA
Interrupt priority register H	IPRH	R/W	H'77	H'FECB
Interrupt priority register I	IPRI	R/W	H'77	H'FECC
Interrupt priority register J	IPRJ	R/W	H'77	H'FECD
Interrupt priority register K	IPRK	R/W	H'77	H'FECE

Notes: 1. Lower 16 bits of the address.

2. Can only be written with 0 for flag clearing.

## 3.2 Register Descriptions

### 3.2.1 System Control Register (SYSCR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	INTM1	INTM0	NMIEG	LWROD	IRQPAS	RAME
Initial value :		0	0	0	0	0	0	0	1
R/W	:	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W

SYSCR is an 8-bit readable/writable register that selects the interrupt control mode, and the detected edge for NMI.

Only bits 5 to 3 are described here; for details of the other bits, see the MCU Operating Modes section in the reference manual for the relevant model.

SYSCR is initialized to H'01 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 5 and 4—Interrupt Control Mode 1 and 0 (INTM1, INTM0):** These bits select one of two interrupt control modes for the interrupt controller.

Bit 5 INTM1	Bit 4 INTM0	Interrupt Control Mode	Description
0	0	0	Interrupts are controlled by I bit (Initial value)
	1	—	Setting prohibited
1	0	2	Interrupts are controlled by bits I2 to I0, and IPR
	1	—	Setting prohibited

**Bit 3—NMI Edge Select (NMIEG):** Selects the input edge for the NMI pin.

Bit 3 NMIEG	Description
0	Interrupt request generated at falling edge of NMI input (Initial value)
1	Interrupt request generated at rising edge of NMI input

**Bit 1—IRQ Input Pin Select (IRQPAS):** Selects switching of the pins that can be used for input of  $\overline{\text{IRQ4}}$  to  $\overline{\text{IRQ7}}$ .  $\overline{\text{IRQ4}}$  to  $\overline{\text{IRQ7}}$  input is always performed from one of the ports.

Some models do not have an IRQPAS bit, and in those that do, the pins that can be used for  $\overline{\text{IRQ4}}$  to  $\overline{\text{IRQ7}}$  input differ from model to model; see the reference manual for the relevant model for details.

### 3.2.2 Interrupt Priority Registers A to K (IPRA to IPRK)

Bit	:	7	6	5	4	3	2	1	0
		—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0
Initial value :		0	1	1	1	0	1	1	1
R/W	:	—	R/W	R/W	R/W	—	R/W	R/W	R/W

The IPR registers are eleven 8-bit readable/writable registers that set priorities (levels 7 to 0) for interrupts other than NMI.

The correspondence between IPR settings and interrupt sources is shown in table 3-3.

The IPR registers set a priority (level 7 to 0) for each interrupt source other than NMI.

The IPR registers are initialized to H'77 by a reset and in hardware standby mode.

**Bits 7 and 3—Reserved:** Read-only bits, always read as 0.

**Table 3-3 Correspondence between Interrupt Sources and IPR Settings**

Register	Bits	
	6 to 4	2 to 0
IPRA	IRQ0	IRQ1
IPRB	IRQ2 IRQ3	IRQ4 IRQ5
IPRC	IRQ6 IRQ7	DTC
IPRD	Watchdog timer	Refresh timer
IPRE	—*	A/D converter
IPRF	TPU channel 0	TPU channel 1
IPRG	TPU channel 2	TPU channel 3
IPRH	TPU channel 4	TPU channel 5
IPRI	8-bit timer channel 0	8-bit timer channel 1
IPRJ	DMAC	SCI channel 0
IPRK	SCI channel 1	SCI channel 2

Note: \* Reserved bits.

As shown in table 3-3, multiple interrupts are assigned to one IPR. Setting a value in the range from H'0 to H'7 in the 3-bit groups of bits 6 to 4 and 2 to 0 sets the priority of the corresponding interrupt. The lowest priority level, level 0, is assigned by setting H'0, and the highest priority level, level 7, by setting H'7.

When interrupt requests are generated, the highest-priority interrupt according to the priority levels set in the IPR registers is selected. This interrupt level is then compared with the interrupt mask level set by the interrupt mask bits (I2 to I0) in the extend register (EXR) in the CPU, and if the priority level of the interrupt is higher than the set mask level, an interrupt request is issued to the CPU.

### 3.2.3 IRQ Enable Register (IER)

Bit	:	7	6	5	4	3	2	1	0
		IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IER is an 8-bit readable/writable register that controls enabling and disabling of interrupt requests IRQ7 to IRQ0.

IER is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 to 0—IRQ7 to IRQ0 Enable (IRQ7E to IRQ0E):** These bits select whether IRQ7 to IRQ0 are enabled or disabled.

Bit n IRQnE	Description	
0	IRQn interrupts disabled	(Initial value)
1	IRQn interrupts enabled	

(n = 7 to 0)

### 3.2.4 IRQ Sense Control Registers H and L (ISCRH, ISCLR)

#### ISCRH

Bit	:	15	14	13	12	11	10	9	8
		IRQ7SCB	IRQ7SCA	IRQ6SCB	IRQ6SCA	IRQ5SCB	IRQ5SCA	IRQ4SCB	IRQ4SCA
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### ISCLR

Bit	:	7	6	5	4	3	2	1	0
		IRQ3SCB	IRQ3SCA	IRQ2SCB	IRQ2SCA	IRQ1SCB	IRQ1SCA	IRQ0SCB	IRQ0SCA
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ISCR (composed of ISCRH and ISCLR) is a 16-bit readable/writable register that selects rising edge, falling edge, or both edge detection, or level sensing, for the input at pins  $\overline{IRQ7}$  to  $\overline{IRQ0}$ .

ISCR is initialized to H'0000 by a reset and in hardware standby mode.

**Bits 15 to 0:** IRQ7 Sense Control A and B (IRQ7SCA, IRQ7SCB) to IRQ0 Sense Control A and B (IRQ0SCA, IRQ0SCB)

#### Bits 15 to 0

IRQ7SCB to IRQ0SCB	IRQ7SCA to IRQ0SCA	Description
0	0	Interrupt request generated at $\overline{IRQ7}$ to $\overline{IRQ0}$ input low level (Initial value)
	1	Interrupt request generated at falling edge of $\overline{IRQ7}$ to $\overline{IRQ0}$ input
1	0	Interrupt request generated at rising edge of $\overline{IRQ7}$ to $\overline{IRQ0}$ input
	1	Interrupt request generated at both falling and rising edges of $\overline{IRQ7}$ to $\overline{IRQ0}$ input

### 3.2.5 IRQ Status Register (ISR)

Bit	:	7	6	5	4	3	2	1	0
		IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: \* Only 0 can be written, to clear the flag.

ISR is an 8-bit readable/writable register that indicates the status of IRQ7 to IRQ0 interrupt requests.

ISR is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 to 0—IRQ7 to IRQ0 flags (IRQ7F to IRQ0F):** These bits indicate the status of IRQ7 to IRQ0 interrupt requests.

Bit n	IRQnF	Description
0		[Clearing conditions] (Initial value)
		<ul style="list-style-type: none"> <li>• Cleared by reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag</li> <li>• When interrupt exception handling is executed when low-level detection is set (IRQnSCB = IRQnSCA = 0) and <math>\overline{\text{IRQn}}</math> input is high</li> <li>• When IRQn interrupt exception handling is executed when falling, rising, or both-edge detection is set (IRQnSCB = 1 or IRQnSCA = 1)</li> <li>• When the DTC is activated by an IRQn interrupt, and the DISEL bit in MRB of the DTC is cleared to 0</li> </ul>
1		[Setting conditions]
		<ul style="list-style-type: none"> <li>• When <math>\overline{\text{IRQn}}</math> input goes low when low-level detection is set (IRQnSCB = IRQnSCA = 0)</li> <li>• When a falling edge occurs in <math>\overline{\text{IRQn}}</math> input when falling edge detection is set (IRQnSCB = 0, IRQnSCA = 1)</li> <li>• When a rising edge occurs in <math>\overline{\text{IRQn}}</math> input when rising edge detection is set (IRQnSCB = 1, IRQnSCA = 0)</li> <li>• When a falling or rising edge occurs in <math>\overline{\text{IRQn}}</math> input when both-edge detection is set (IRQnSCB = IRQnSCA = 1)</li> </ul>

(n = 7 to 0)

### 3.3 Interrupt Sources

Interrupt sources comprise external interrupts (NMI and IRQ7 to IRQ0) and internal interrupts (52 sources).

#### 3.3.1 External Interrupts

There are nine external interrupts: NMI and IRQ7 to IRQ0. NMI and IRQ7 to IRQ0 can be used to restore the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip from software standby mode. (IRQ7 to IRQ3 can be designated for use as software standby mode clearing sources by setting the IRQ37S bit in SBYCR to 1.)

**NMI Interrupt:** NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the status of the CPU interrupt mask bits. The NMIEG bit in SYSCR can be used to select whether an interrupt is requested at a rising edge or a falling edge on the NMI pin.

The vector number for NMI interrupt exception handling is 7.

**IRQ7 to IRQ0 Interrupts:** Interrupts IRQ7 to IRQ0 are requested by an input signal at pins  $\overline{\text{IRQ7}}$  to  $\overline{\text{IRQ0}}$ . Interrupts IRQ7 to IRQ0 have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins  $\overline{\text{IRQ7}}$  to  $\overline{\text{IRQ0}}$ .
- Enabling or disabling of interrupt requests IRQ7 to IRQ0 can be selected with IER.
- The interrupt priority level can be set with IPR.
- The status of interrupt requests IRQ7 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

A block diagram of interrupts IRQ7 to IRQ0 is shown in figure 3-2.

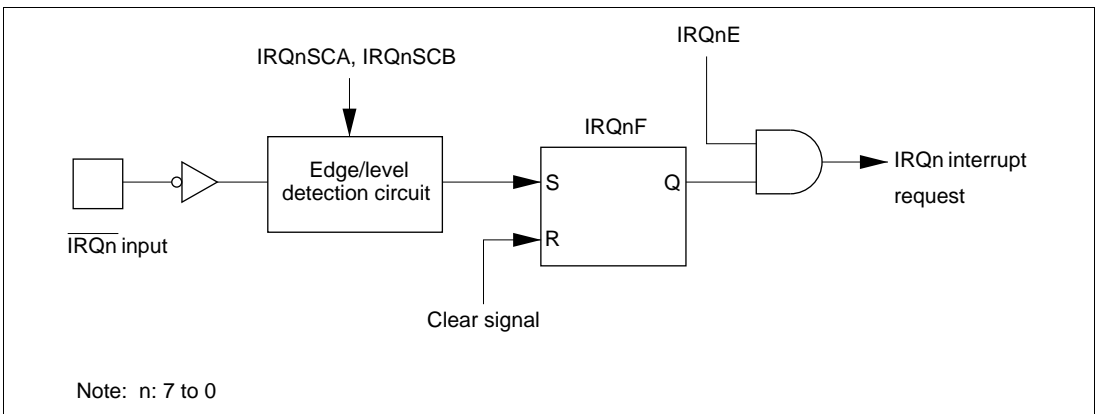
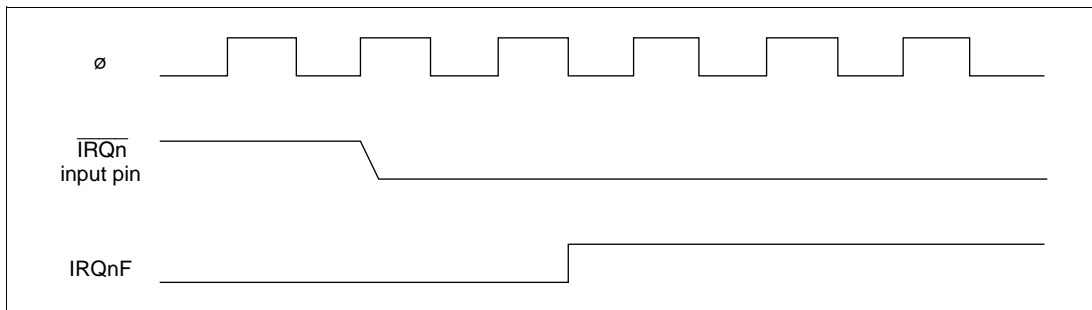


Figure 3-2 Block Diagram of Interrupts IRQ7 to IRQ0



Figure 3-3 shows the timing of setting IRQnF.



**Figure 3-3 Timing of Setting IRQnF**

The vector numbers for IRQ7 to IRQ0 interrupt exception handling are 23 to 16.

Detection of IRQ7 to IRQ0 interrupts does not depend on whether the relevant pin has been set for input or output. Therefore, when a pin is used as an external interrupt input pin, do not clear the corresponding DDR bit to 0 and use the pin as an I/O pin for another function. The pins that can be used for IRQ4 to IRQ7 interrupt input can be switched by means of the IRQPAS bit in SYSCR. The switched pins differ from model to model; see the reference manual for the relevant model for details.

### 3.3.2 Internal Interrupts

There are 52 sources for internal interrupts from on-chip supporting modules.

- For each on-chip supporting module there are flags that indicate the interrupt request status, and enable bits that select enabling or disabling of these interrupts. If both of these are set to 1 for a particular interrupt source, an interrupt request is issued to the interrupt controller.
- The interrupt priority level can be set by means of IPR.
- The DMAC and DTC can be activated by a TPU, SCI, or other interrupt request. When the DMAC or DTC is activated by an interrupt, the interrupt control mode and interrupt mask bits have no effect.

### 3.3.3 Interrupt Exception Vector Table

Table 3-4 shows interrupt exception handling sources, vector addresses, and interrupt priorities. For default priorities, the lower the vector number, the higher the priority.

Priorities among modules can be set by means of IPR. The situation when two or more modules are set to the same priority, and priorities within a module, are fixed as shown in table 3-4.

**Table 3-4 Interrupt Sources, Vector Addresses, and Interrupt Priorities**

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address*	IPR	Priority
NMI	External pin	7	H'001C		High ↑
IRQ0		16	H'0040	IPRA6 to 4	
IRQ1		17	H'0044	IPRA2 to 0	
IRQ2		18	H'0048	IPRB6 to 4	
IRQ3		19	H'004C		
IRQ4		20	H'0050	IPRB2 to 0	
IRQ5		21	H'0054		
IRQ6		22	H'0058	IPRC6 to 4	
IRQ7		23	H'005C		
SWDTEND (software-activated data transfer end)		DTC	24	H'0060	
WOVI (interval timer)	Watchdog timer	25	H'0064	IPRD6 to 4	
CMI (compare match)	Refresh controller	26	H'0068	IPRD2 to 0	
Reserved	—	27	H'006C	IPRE6 to 4	
ADI (A/D conversion end)	A/D	28	H'0070	IPRE2 to 0	
Reserved	—	29	H'0074		
		30	H'0078		
		31	H'007C		
TGI0A (TGR0A input capture/compare match)	TPU channel 0	32	H'0080	IPRF6 to 4	
TGI0B (TGR0B input capture/compare match)		33	H'0084		
TGI0C (TGR0C input capture/compare match)		34	H'0088		
TGI0D (TGR0D input capture/compare match)		35	H'008C		
TCI0V (overflow 0)		36	H'0090		
Reserved	—	37	H'0094		
		38	H'0098		
		39	H'009C		Low

Note: \* Lower 16 bits of the start address.

**Table 3-4 Interrupt Sources, Vector Addresses, and Interrupt Priorities (cont)**

<b>Interrupt Source</b>	<b>Origin of Interrupt Source</b>	<b>Vector Number</b>	<b>Vector Address*</b>	<b>IPR</b>	<b>Priority</b>
TGI1A (TGR1A input capture/compare match)	TPU channel 1	40	H'00A0	IPRF2 to 0	High ↑
TGI1B (TGR1B input capture/compare match)		41	H'00A4		
TCI1V (overflow 1)		42	H'00A8		
TCI1U (underflow 1)		43	H'00AC		
TGI2A (TGR2A input capture/compare match)	TPU channel 2	44	H'00B0	IPRG6 to 4	
TGI2B (TGR2B input capture/compare match)		45	H'00B4		
TCI2V (overflow 2)		46	H'00B8		
TCI2U (underflow 2)		47	H'00BC		
TGI3A (TGR3A input capture/compare match)	TPU channel 3	48	H'00C0	IPRG2 to 0	
TGI3B (TGR3B input capture/compare match)		49	H'00C4		
TGI3C (TGR3C input capture/compare match)		50	H'00C8		
TGI3D (TGR3D input capture/compare match)		51	H'00CC		
TCI3V (overflow 3)		52	H'00D0		
Reserved		—	53 54 55		
TGI4A (TGR4A input capture/compare match)	TPU channel 4	56	H'00E0	IPRH6 to 4	
TGI4B (TGR4B input capture/compare match)		57	H'00E4		
TCI4V (overflow 4)		58	H'00E8		
TCI4U (underflow 4)		59	H'00EC		
TGI5A (TGR5A input capture/compare match)	TPU channel 5	60	H'00F0	IPRH2 to 0	
TGI5B (TGR5B input capture/compare match)		61	H'00F4		
TCI5V (overflow 5)		62	H'00F8		
TCI5U (underflow 5)		63	H'00FC		

Note: \* Lower 16 bits of the start address.

**Table 3-4 Interrupt Sources, Vector Addresses, and Interrupt Priorities (cont)**

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address*	IPR	Priority
CMIA0 (compare match A0)	8-bit timer channel 0	64	H'0100	IPRI6 to 4	High ↑
CMIB0 (compare match B0)		65	H'0104		
OVI0 (overflow 0)		66	H'0108		
Reserved	—	67	H'010C		
CMIA1 (compare match A1)	8-bit timer channel 1	68	H'0110	IPRI2 to 0	
CMIB1 (compare match B1)		69	H'0114		
OVI1 (overflow 1)		70	H'0118		
Reserved	—	71	H'011C		
DEND0A (channel 0/channel 0A transfer end)	DMAC	72	H'0120	IPRJ6 to 4	
DEND0B (channel 0B transfer end)		73	H'0124		
DEND1A (channel 1/channel 1A transfer end)		74	H'0128		
DEND1B (channel 1B transfer end)		75	H'012C		
Reserved		—	76 77 78 79		H'0130 H'0134 H'0138 H'013C
ERI0 (receive error 0)	SCI channel 0	80	H'0140	IPRJ2 to 0	
RXI0 (reception completed 0)		81	H'0144		
TXI0 (transmit data empty 0)		82	H'0148		
TEI0 (transmission end 0)		83	H'014C		
ERI1 (receive error 1)	SCI channel 1	84	H'0150	IPRK6 to 4	
RXI1 (reception completed 1)		85	H'0154		
TXI1 (transmit data empty 1)		86	H'0158		
TEI1 (transmission end 1)		87	H'015C		
ERI2 (receive error 2)	SCI channel 2	88	H'0160	IPRK2 to 0	
RXI2 (reception completed 2)		89	H'0164		
TXI2 (transmit data empty 2)		90	H'0168		
TEI2 (transmission end 2)		91	H'016C		Low

Notes: Interrupt sources differ from model to model; see the reference manual for the relevant model for details.

\* Lower 16 bits of the start address.

## 3.4 Interrupt Operation

### 3.4.1 Interrupt Control Modes and Interrupt Operation

Interrupt operations in the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series differ depending on the interrupt control mode.

NMI interrupts are accepted at all times except in the reset state and the hardware standby state. In the case of IRQ interrupts and on-chip supporting module interrupts, an enable bit is provided for each interrupt. Clearing an enable bit to 0 disables the corresponding interrupt request. Interrupt sources for which the enable bits are set to 1 are controlled by the interrupt controller.

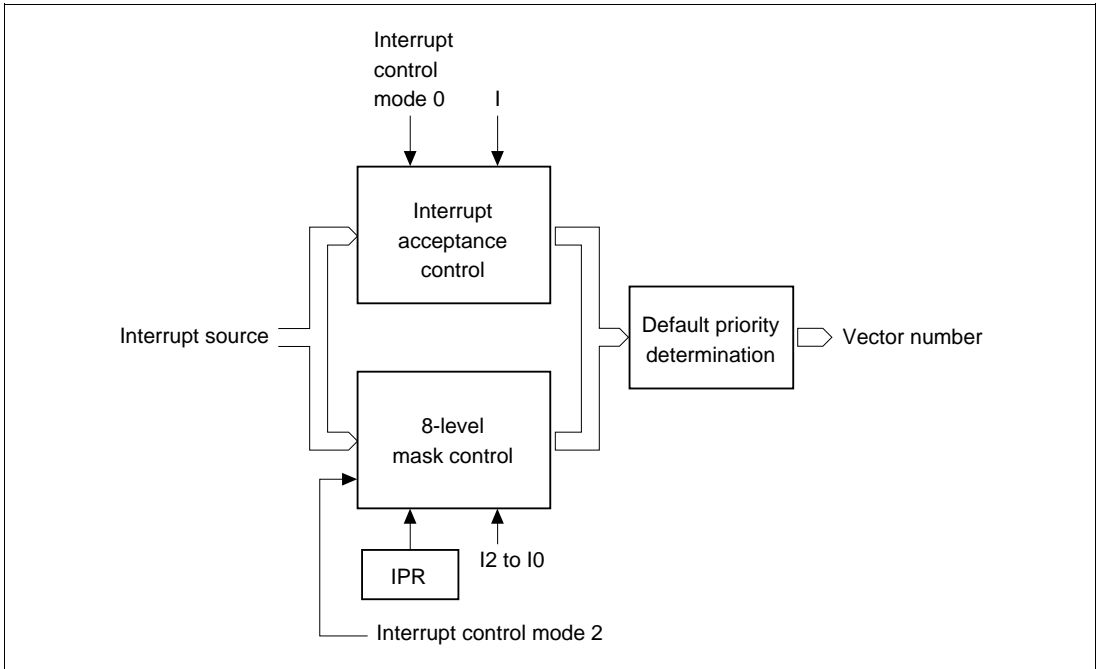
Table 3-5 shows the interrupt control modes.

The interrupt controller performs interrupt control according to the interrupt control mode set by the INTM1 and INTM0 bits in SYSCR, the priorities set in IPR, and the masking state indicated by the I bit in the CPU's CCR, and bits I2 to I0 in EXR.

**Table 3-5 Interrupt Control Modes**

Interrupt Control Mode	SYSCR		Priority Setting Registers	Interrupt Mask Bits	Description
	INTM1	INTM0			
0	0	0	—	I	Interrupt mask control is performed by the I bit.
—	—	1	—	—	Setting prohibited
2	1	0	IPR	I2 to I0	8-level interrupt mask control is performed by bits I2 to I0. 8 priority levels can be set with IPR.
—	—	1	—	—	Setting prohibited

Figure 3-4 shows a block diagram of the priority decision circuit.



**Figure 3-4 Block Diagram of Interrupt Control Operation**

**Interrupt Acceptance Control:** In interrupt control mode 0, interrupt acceptance is controlled by the I bit in CCR.

Table 3-6 shows the interrupts selected in each interrupt control mode.

**Table 3-6 Interrupts Selected in Each Interrupt Control Mode (1)**

Interrupt Control Mode	Interrupt Mask Bits	
	I	Selected Interrupts
0	0	All interrupts
	1	NMI interrupts
2	*	All interrupts

\* : Don't care

**8-Level Control:** In interrupt control mode 2, 8-level mask level determination is performed for the selected interrupts in interrupt acceptance control according to the interrupt priority level (IPR).

The interrupt source selected is the interrupt with the highest priority level, and whose priority level set in IPR is higher than the mask level.

**Table 3-7 Interrupts Selected in Each Interrupt Control Mode (2)**

Interrupt Control Mode	Selected Interrupts
0	All interrupts
2	Highest-priority-level (IPR) interrupt whose priority level is greater than the mask level (IPR > I2 to I0)

**Default Priority Determination:** When an interrupt is selected by 8-level control, its priority is determined and a vector number is generated.

If the same value is set for IPR, acceptance of multiple interrupts is enabled, and so only the interrupt source with the highest priority according to the preset default priorities is selected and has a vector number generated.

Interrupt sources with a lower priority than the accepted interrupt source are held pending.

Table 3-8 shows operations and control signal functions in each interrupt control mode.

**Table 3-8 Operations and Control Signal Functions in Each Interrupt Control Mode**

Interrupt Control Mode	Setting		Interrupt Acceptance Control	8-Level Control			Default Priority Determination	T (Trace)
	INTM1	INTM0	I	I2 to I0	IPR			
0	0	0	○ IM	X	—	—*2	○	—
2	1	0	X —*1	○	IM	PR	○	T

**Legend**

- : Interrupt operation control performed
- X : No operation. (All interrupts enabled)
- IM : Used as interrupt mask bit
- PR : Sets priority.
- : Not used.
- \*1 : Set to 1 when interrupt is accepted.
- \*2 : Keep the initial setting.

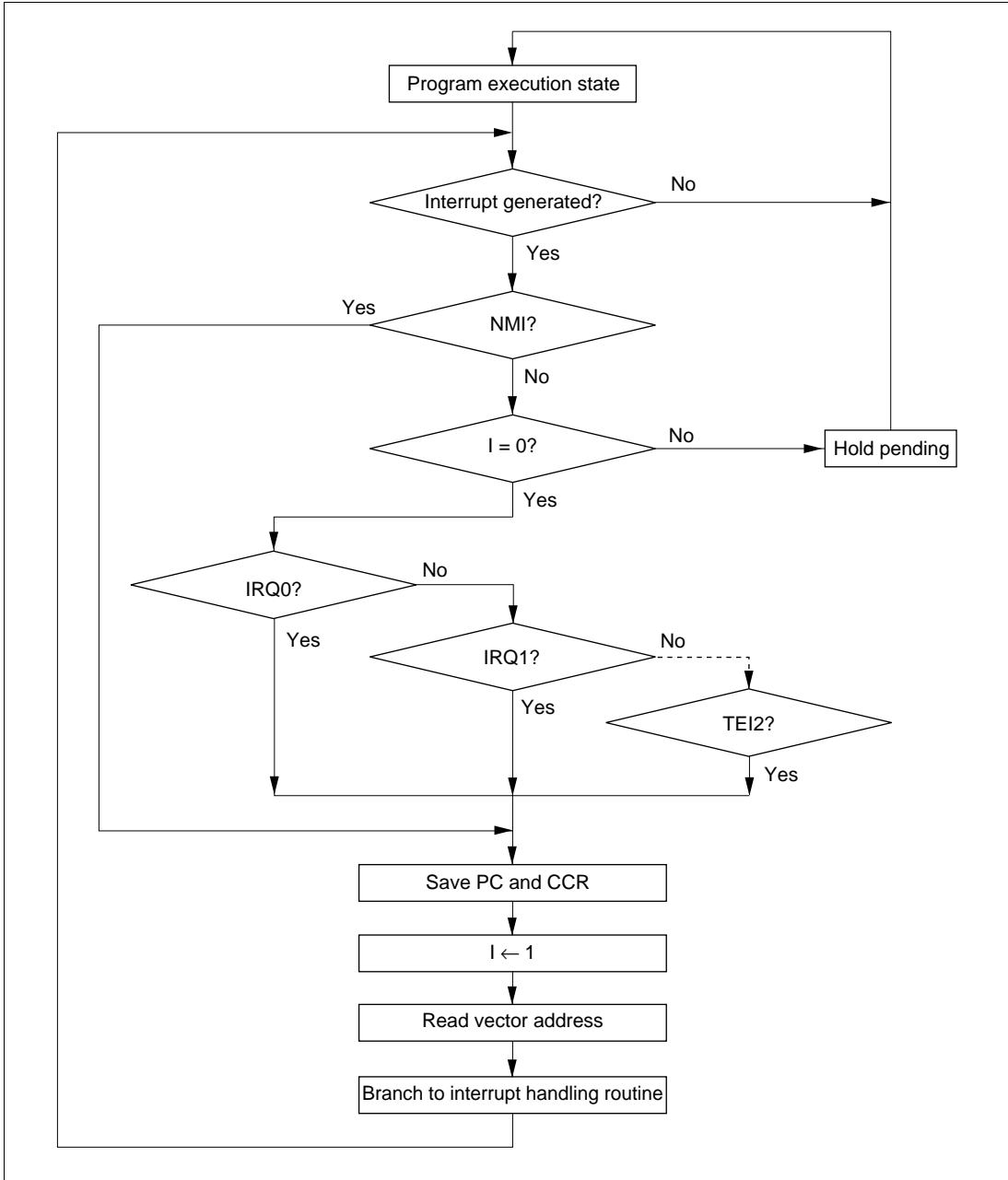
### 3.4.2 Interrupt Control Mode 0

Enabling and disabling of IRQ interrupts and on-chip supporting module interrupts can be set by means of the I bit in the CPU's CCR. Interrupts are enabled when the I bit is cleared to 0, and disabled when set to 1.

Figure 3-5 shows a flowchart of the interrupt acceptance operation in this case.

- [1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
- [2] The I bit is then referenced. If the I bit is cleared to 0, the interrupt request is accepted. If the I bit is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending.
- [3] Interrupt requests are sent to the interrupt controller, the highest-ranked interrupt according to the priority system is accepted, and other interrupt requests are held pending.
- [4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.
- [5] The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
- [6] Next, the I bit in CCR is set to 1. This masks all interrupts except NMI.
- [7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.





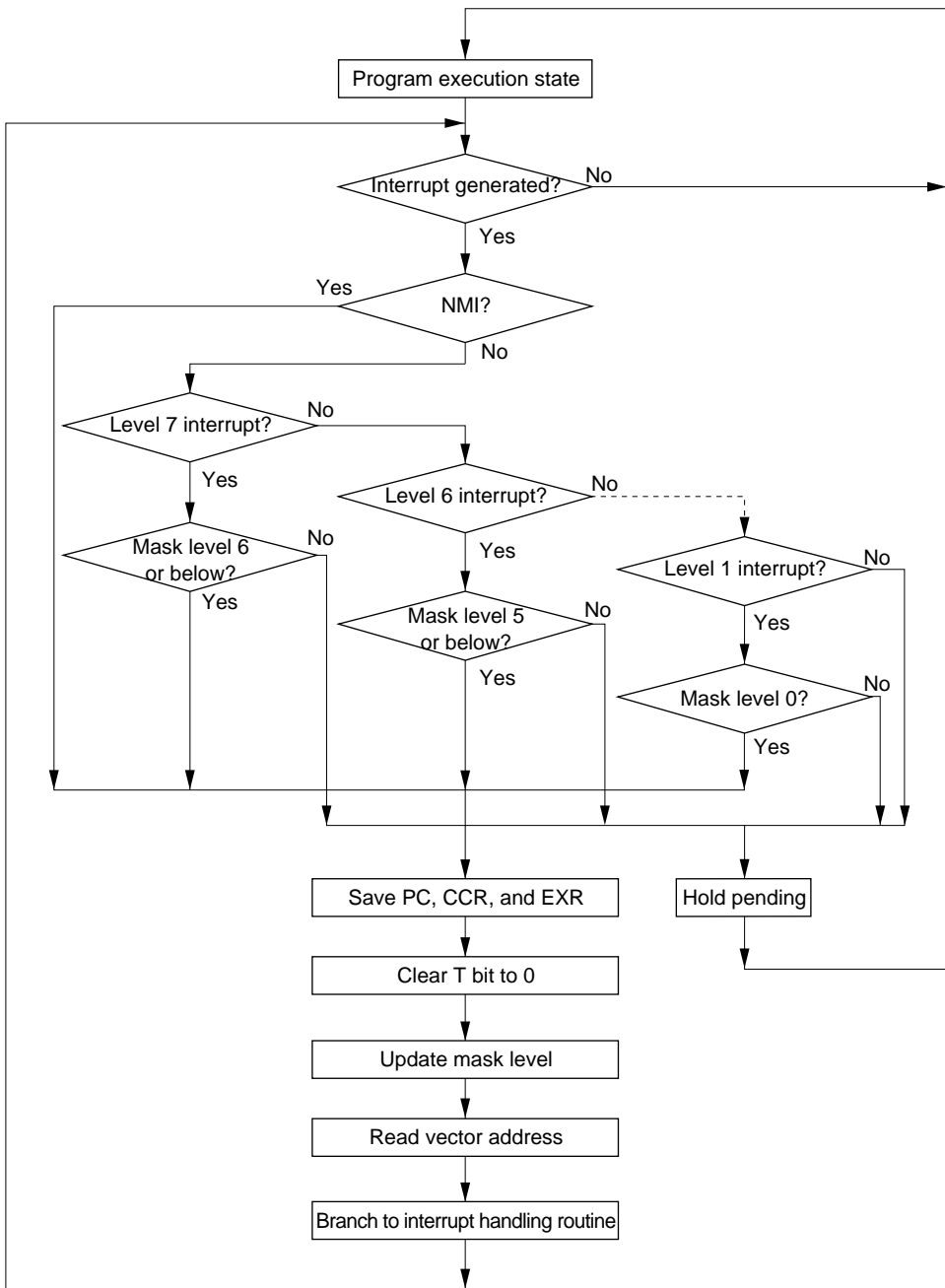
**Figure 3-5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

### 3.4.3 Interrupt Control Mode 2

Eight-level masking is implemented for IRQ interrupts and on-chip supporting module interrupts by comparing the interrupt mask level set by bits I2 to I0 of EXR in the CPU with IPR.

Figure 3-6 shows a flowchart of the interrupt acceptance operation in this case.

- [1] If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
- [2] When interrupt requests are sent to the interrupt controller, the interrupt with the highest priority according to the interrupt priority levels set in IPR is selected, and lower-priority interrupt requests are held pending. If a number of interrupt requests with the same priority are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 3-4 is selected.
- [3] Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. An interrupt request with a priority no higher than the mask level set at that time is held pending, and only an interrupt request with a priority higher than the interrupt mask level is accepted.
- [4] When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.
- [5] The PC, CCR, and EXR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
- [6] The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority level of the accepted interrupt.  
If the accepted interrupt is NMI, the interrupt mask level is set to H'7.
- [7] A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.



**Figure 3-6 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2**

### 3.4.4 Interrupt Exception Handling Sequence

Figure 3-7 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.

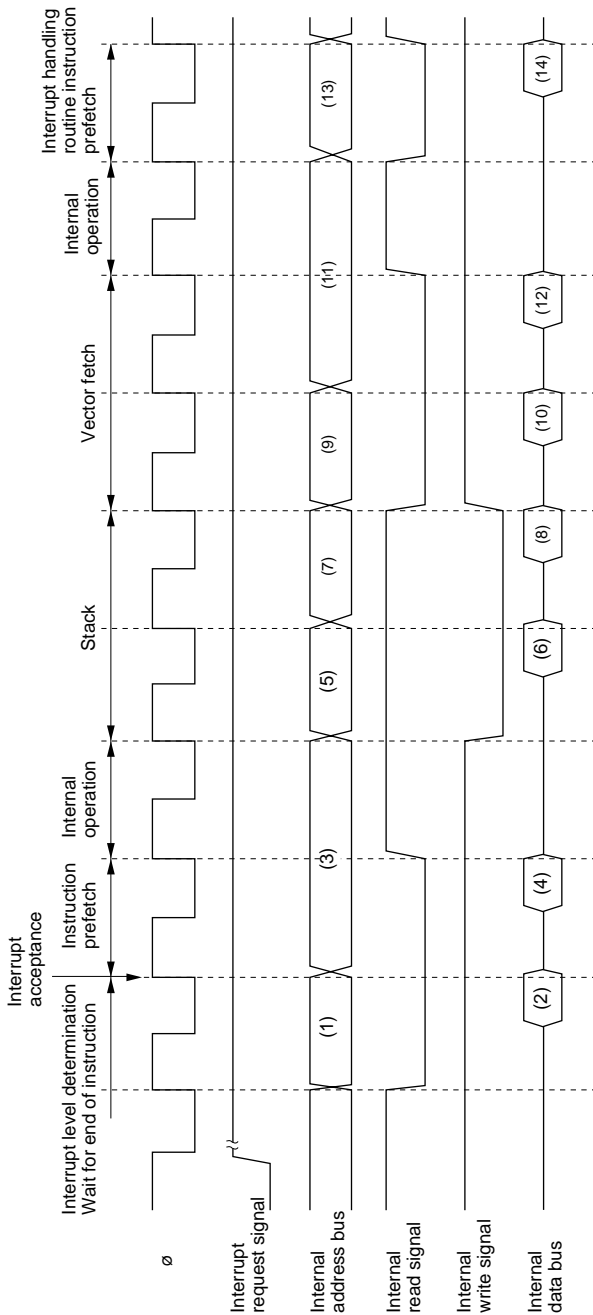


Figure 3-7 Interrupt Exception Handling

### 3.4.5 Interrupt Response Times

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are capable of fast word transfer instruction to on-chip memory, and the program area is provided in on-chip ROM and the stack area in on-chip RAM, enabling high-speed processing.

Table 3-9 shows interrupt response times—the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The execution status symbols used in table 3-9 are explained in table 3-10.

**Table 3-9 Interrupt Response Times**

No.	Item	Advanced Mode	
		INTM1 = 0	INTM1 = 1
1	Interrupt priority determination* <sup>1</sup>	3	3
2	Number of wait states until executing instruction ends* <sup>2</sup>	1 to 19+2·S <sub>I</sub>	1 to 19+2·S <sub>I</sub>
3	PC, CCR, EXR stack save	2·S <sub>K</sub>	3·S <sub>K</sub>
4	Vector fetch	2·S <sub>I</sub>	2·S <sub>I</sub>
5	Instruction fetch* <sup>3</sup>	2·S <sub>I</sub>	2·S <sub>I</sub>
6	Internal processing* <sup>4</sup>	2	2
Total (using on-chip memory)		12 to 32	13 to 33

- Notes: 1. Two states in case of internal interrupt.  
 2. Refers to MULXS and DIVXS instructions.  
 3. Prefetch after interrupt acceptance and interrupt handling routine prefetch.  
 4. Internal processing after interrupt acceptance and internal processing after vector fetch.

**Table 3-10 Number of States in Interrupt Handling Routine Execution**

Symbol		Internal Memory	Object of Access			
			External Device			
			8-Bit Bus		16-Bit Bus	
			2-State Access	3-State Access	2-State Access	3-State Access
Instruction fetch	S <sub>I</sub>	1	4	6+2m	2	3+m
Branch address read	S <sub>J</sub>					
Stack manipulation	S <sub>K</sub>					

**Legend**

m : Number of wait states in an external device access.

## 3.5 Usage Notes

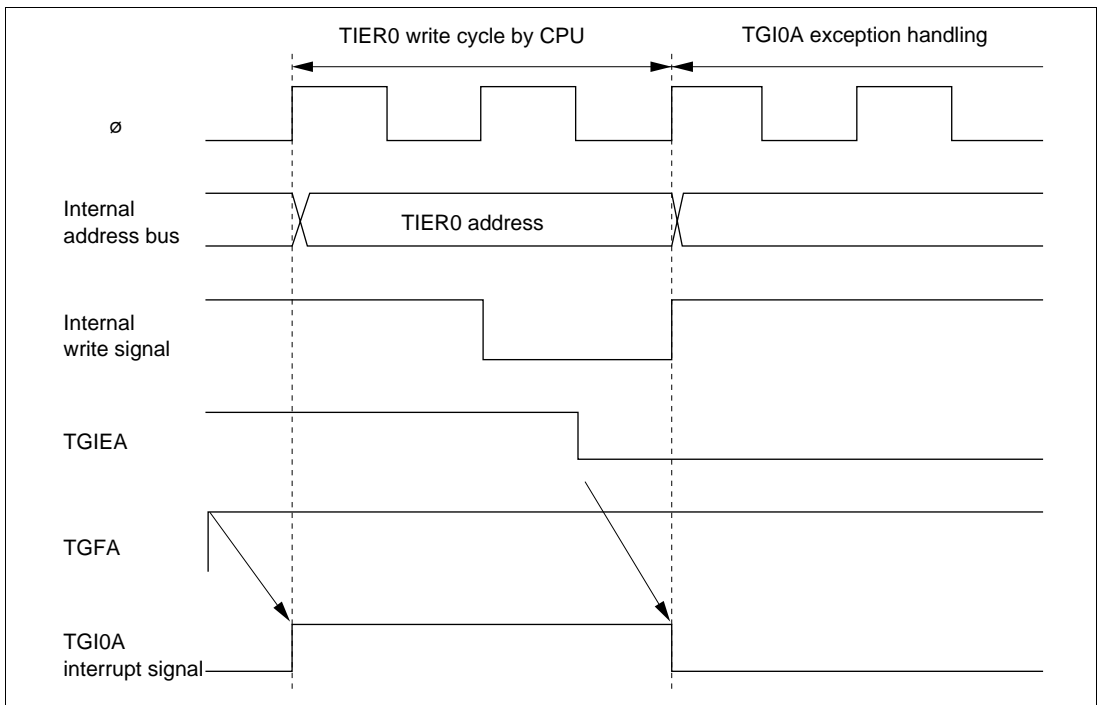
### 3.5.1 Contention between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupts, the disabling becomes effective after execution of the instruction.

In other words, when an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored.

The same also applies when an interrupt source flag is cleared.

Figure 3-8 shows an example in which the TGIEA bit in the TPU's TIER0 register is cleared to 0.



**Figure 3-8 Contention between Interrupt Generation and Disabling**

The above contention will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

### 3.5.2 Instructions that Disable Interrupts

Instructions that disable interrupts are LDC, ANDC, ORC, and XORC. After any of these instructions is executed, all interrupts including NMI are disabled and the next instruction is always executed. When the I bit is set by one of these instructions, the new value becomes valid two states after execution of the instruction ends.

### 3.5.3 Times when Interrupts are Disabled

There are times when interrupt acceptance is disabled by the interrupt controller.

The interrupt controller disables interrupt acceptance for a 3-state period after the CPU has updated the mask level with an LDC, ANDC, ORC, or XORC instruction.

### 3.5.4 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B instruction and the EEPMOV.W instruction.

With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the move is completed.

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at a break in the transfer cycle. The PC value saved on the stack in this case is the address of the next instruction.

Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

```
L1:  EEPMOV.W  
      MOV.W    R4,R4  
      BNE     L1
```



## 3.6 DTC and DMAC Activation by Interrupt

### 3.6.1 Overview

The DTC and DMAC can be activated by an interrupt. In this case, the following options are available. Some models do not have an on-chip DMAC; see the reference manual for the relevant model.

1. Interrupt request to CPU
2. Activation request to DTC
3. Activation request to DMAC
4. Selection of a number of the above

For details of interrupt requests that can be used with to activate the DTC or DMAC, see section 6, Data Transfer Controller, and section 5, DMA Controller.

### 3.6.2 Block Diagram

Figure 3-9 shows a block diagram of the DTC, DMAC, and interrupt controller.

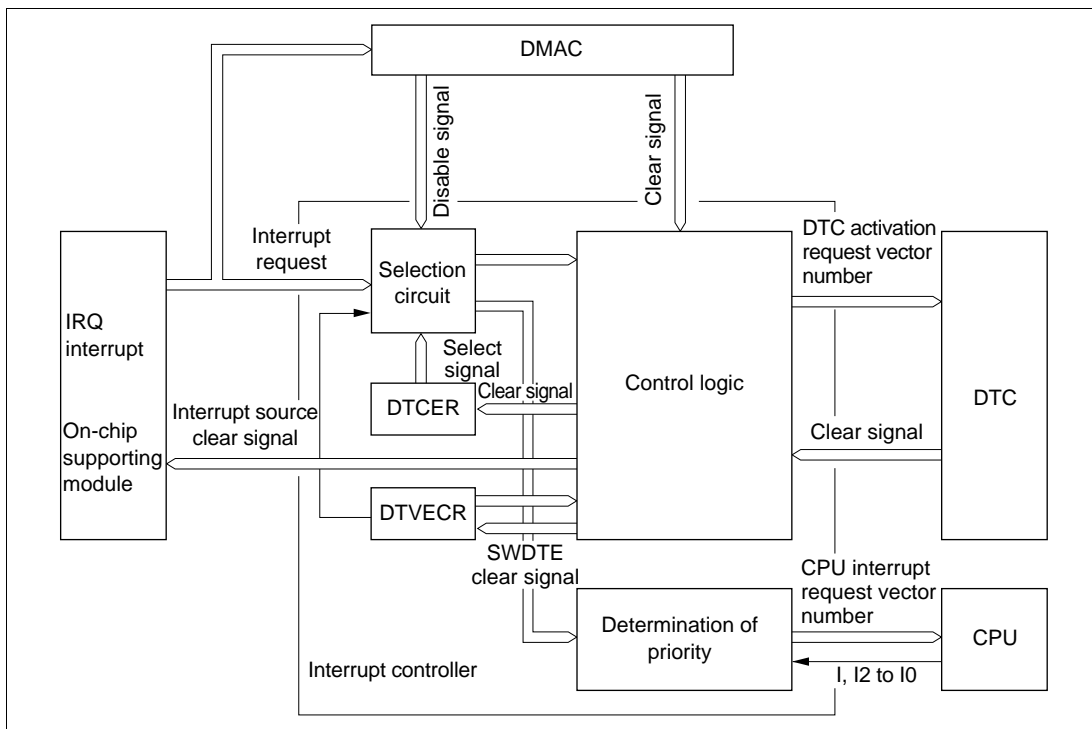


Figure 3-9 Interrupt Control for DTC and DMAC

### 3.6.3 Operation

The interrupt controller has three main functions in DTC and DMAC control.

**Selection of Interrupt Source:** With the DMAC, the activation source is input directly to each channel. The activation source for each DMAC channel is selected with bits DTF3 to DTF0 in DMACR. Whether the selected activation source is to be managed by the DMAC can be selected with the DTA bit of DMABCR. When the DTA bit is set to 1, the interrupt source constituting that DMAC activation source is not a DTC activation source or CPU interrupt source.

For interrupt sources other than interrupts managed by the DMAC, it is possible to select DTC activation request or CPU interrupt request with the DTCE bit of DTCEA to DTCEF in the DTC.

After a DTC data transfer, the DTCE bit can be cleared to 0 and an interrupt request sent to the CPU in accordance with the specification of the DISEL bit of MRB in the DTC.

When the DTC has performed the specified number of data transfers and the transfer counter value is zero, the DTCE bit is cleared to 0 and an interrupt request is sent to the CPU after the DTC data transfer.

**Determination of Priority:** The DTC activation source is selected in accordance with the default priority order, and is not affected by mask or priority levels. See section 5.6, Interrupts, and section 6.3.3, DTC Vector Table, for the respective priorities.

With the DMAC, the activation source is input directly to each channel.

**Operation Order:** If the same interrupt is selected as a DTC activation source and a CPU interrupt source, the DTC data transfer is performed first, followed by CPU interrupt exception handling.

If the same interrupt is selected as a DMAC activation source and a DTC activation source or CPU interrupt source, operations are performed for them independently according to their respective operating statuses and bus mastership priorities.

Table 3-11 summarizes interrupt source selection and interrupt source clearance control according to the settings of the DTA bit of DMABCR in the DMAC, the DTCE bit of DTCEA to DTCEF in the DTC, and the DISEL bit of MRB in the DTC.

**Table 3-11 Interrupt Source Selection and Clearing Control**

Settings			Interrupt Source Selection/Clearing Control		
DMAC	DTC		DMAC	DTC	CPU
DTA	DTCE	DISEL	DMAC	DTC	CPU
0	0	*	○	X	◎
	1	0	○	◎	X
		1	○	○	◎
1	*	*	◎	X	X

**Legend**

- ◎ : The relevant interrupt is used. Interrupt source clearing is performed.  
(The CPU should clear the source flag in the interrupt handling routine.)
- : The relevant interrupt is used. The interrupt source is not cleared.
- X : The relevant interrupt cannot be used.
- \* : Don't care

**Usage Note:** SCI and A/D converter interrupt sources are cleared when the DMAC or DTC reads or writes to the prescribed register, and are not dependent upon the DTA bit or DISEL bit.

# Section 4 Bus Controller

## 4.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have a built-in bus controller (BSC) that manages the external address space divided into eight areas. The bus specifications, such as bus width and number of access states, can be set independently for each area, enabling multiple memories to be connected easily.

The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters: the CPU, DMA controller (DMAC), and data transfer controller (DTC).

This section describes the bus controller with the maximum series specifications; see the reference manual for the relevant model for details of functions.

### 4.1.1 Features

The features of the bus controller are listed below.

- Manages external address space in area units
  - In advanced mode, manages the external space as 8 areas of 2 Mbytes
  - Bus specifications can be set independently for each area
  - DRAM/burst ROM interfaces can be set
- Basic bus interface
  - Chip select ( $\overline{CS0}$  to  $\overline{CS7}$ ) can be output for areas 0 to 7
  - 8-bit access or 16-bit access can be selected for each area
  - 2-state access or 3-state access can be selected for each area
  - Program wait states can be inserted for each area
- DRAM interface
  - DRAM interface can be set for areas 2 to 5 (in advanced mode)
  - Row address/column address multiplexed output (8/9/10 bits)
  - 2-CAS access method
  - Burst operation (fast page mode)
  - $T_p$  cycle insertion to secure RAS precharging time
  - Choice of CAS-before-RAS refreshing or self-refreshing
- Burst ROM interface
  - Burst ROM interface can be set for area 0
  - Choice of 1- or 2-state burst access

- Idle cycle insertion
  - An idle cycle can be inserted in case of an external read cycle between different areas
  - An idle cycle can be inserted when an external read cycle is immediately followed by an external write cycle
- Write buffer functions
  - External write cycle and internal access can be executed in parallel
  - DMAC single address mode and internal access can be executed in parallel
- Bus arbitration function
  - Includes a bus arbiter that arbitrates bus mastership among the CPU, DMAC, and DTC
- Other features
  - Refresh counter (refresh timer) can be used as an interval timer
  - External bus release function

## 4.1.2 Block Diagram

Figure 4-1 shows a block diagram of the bus controller.

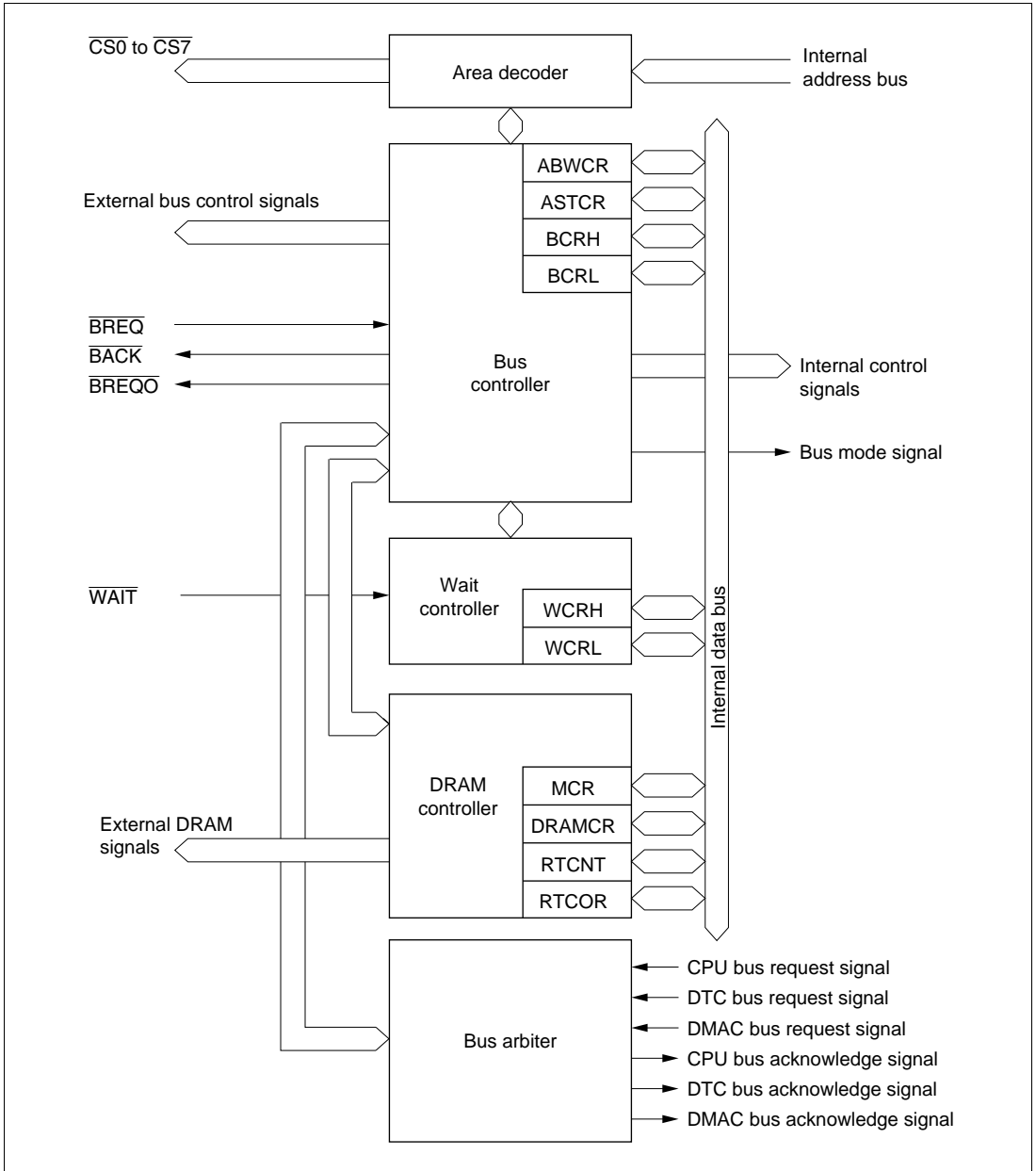


Figure 4-1 Block Diagram of Bus Controller

### 4.1.3 Pin Configuration

Table 4-1 summarizes the pins of the bus controller. The pins used for output of the various signals differ from model to model; see the reference manual for the relevant model for details.

**Table 4-1 Bus Controller Pins**

<b>Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
Address strobe	$\overline{AS}$	Output	Strobe signal indicating that address output on address bus is enabled.
Read	$\overline{RD}$	Output	Strobe signal indicating that external space is being read.
High write/write enable	$\overline{HWR}$	Output	Strobe signal indicating that external space is to be written, and upper half ( $D_{15}$ to $D_8$ ) of data bus is enabled. 2-CAS DRAM write enable signal.
Low write	$\overline{LWR}$	Output	Strobe signal indicating that external space is to be written, and lower half ( $D_7$ to $D_0$ ) of data bus is enabled.
Chip select 0	$\overline{CS0}$	Output	Strobe signal indicating that area 0 is selected.
Chip select 1	$\overline{CS1}$	Output	Strobe signal indicating that area 1 is selected.
Chip select 2/row address strobe 2	$\overline{CS2}$	Output	Strobe signal indicating that area 2 is selected. DRAM row address strobe signal when area 2 is in DRAM space.
Chip select 3/row address strobe 3	$\overline{CS3}$	Output	Strobe signal indicating that area 3 is selected. DRAM row address strobe signal when area 3 is in DRAM space.
Chip select 4/row address strobe 4	$\overline{CS4}$	Output	Strobe signal indicating that area 4 is selected. DRAM row address strobe signal when area 4 is in DRAM space.
Chip select 5/row address strobe 5	$\overline{CS5}$	Output	Strobe signal indicating that area 5 is selected. DRAM row address strobe signal when area 5 is in DRAM space.
Chip select 6	$\overline{CS6}$	Output	Strobe signal indicating that area 6 is selected.
Chip select 7	$\overline{CS7}$	Output	Strobe signal indicating that area 7 is selected.

**Table 4-1 Bus Controller Pins (cont)**

Name	Symbol	I/O	Function
Upper column address strobe	$\overline{\text{CAS}}$	Output	2-CAS DRAM upper column address strobe signal.
Lower column strobe	$\overline{\text{LCAS}}$	Output	DRAM lower column address strobe signal.
Wait	$\overline{\text{WAIT}}$	Input	Wait request signal when accessing external 3-state access space.
Bus request	$\overline{\text{BREQ}}$	Input	Request signal that releases bus to external device.
Bus request acknowledge	$\overline{\text{BACK}}$	Output	Acknowledge signal indicating that bus has been released.
Bus request output	$\overline{\text{BREQO}}$	Output	External bus request signal used when internal bus master accesses external space when external bus is released.

#### 4.1.4 Register Configuration

Table 4-2 summarizes the registers of the bus controller.

**Table 4-2 Bus Controller Registers**

Name	Abbreviation	R/W	Initial Value	
			Reset	Address* <sup>1</sup>
Bus width control register	ABWCR	R/W	H'FF/H'00* <sup>2</sup>	H'FED0
Access state control register	ASTCR	R/W	H'FF	H'FED1
Wait control register H	WCRH	R/W	H'FF	H'FED2
Wait control register L	WCRL	R/W	H'FF	H'FED3
Bus control register H	BCRH	R/W	H'D0	H'FED4
Bus control register L	BCRL	R/W	H'3C	H'FED5
Memory control register	MCR	R/W	H'00	H'FED6
DRAM control register	DRAMCR	R/W	H'00	H'FED7
Refresh timer counter	RTCNT	R/W	H'00	H'FED8
Refresh time constant register	RTCOR	R/W	H'FF	H'FED9

- Notes: 1. Lower 16 bits of the address.  
2. Determined by the MCU operating mode.



## 4.2 Register Descriptions

### 4.2.1 Bus Width Control Register (ABWCR)

Bit	:	7	6	5	4	3	2	1	0
		ABW7	ABW6	ABW5	ABW4	ABW3	ABW2	ABW1	ABW0
Modes 5 to 7									
Initial value :		1	1	1	1	1	1	1	1
R/W :		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Mode 4									
Initial value :		0	0	0	0	0	0	0	0
R/W :		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ABWCR is an 8-bit readable/writable register that designates each area for either 8-bit access or 16-bit access.

ABWCR sets the data bus width for the external memory space. The bus width for on-chip memory and internal I/O registers is fixed regardless of the settings in ABWCR.

After a reset and in hardware standby mode, ABWCR is initialized to H'FF in modes 5 to 7,\* and to H'00 in mode 4. It is not initialized in software standby mode.

Note: \* Modes 6 and 7 are not provided in the ROMless version.

**Bits 7 to 0—Area 7 to 0 Bus Width Control (ABW7 to ABW0):** These bits select whether the corresponding area is to be designated for 8-bit access or 16-bit access.

Bit n ABWn	Description
0	Area n is designated for 16-bit access
1	Area n is designated for 8-bit access

(n = 7 to 0)

## 4.2.2 Access State Control Register (ASTCR)

Bit	:	7	6	5	4	3	2	1	0
		AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0
Initial value :		1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ASTCR is an 8-bit readable/writable register that designates each area as either a 2-state access space or a 3-state access space.

ASTCR sets the number of access states for the external memory space. The number of access states for on-chip memory and internal I/O registers is fixed regardless of the settings in ASTCR.

ASTCR is initialized to H'FF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 7 to 0—Area 7 to 0 Access State Control (AST7 to AST0):** These bits select whether the corresponding area is to be designated as a 2-state access space or a 3-state access space.

Wait state insertion is enabled or disabled at the same time.

Bit n	Description
0	Area n is designated for 2-state access Wait state insertion in area n external space is disabled
1	Area n is designated for 3-state access Wait state insertion in area n external space is enabled

(Initial value)  
(n = 7 to 0)

### 4.2.3 Wait Control Registers H and L (WCRH, WCRL)

WCRH and WCRL are 8-bit readable/writable registers that select the number of program wait states for each area.

Program waits are not inserted in the case of on-chip memory or internal I/O registers.

WCRH and WCRL are initialized to H'FF by a reset and in hardware standby mode. They are not initialized in software standby mode.

#### WCRH

Bit	:	7	6	5	4	3	2	1	0
		W71	W70	W61	W60	W51	W50	W41	W40
Initial value :		1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Bits 7 and 6—Area 7 Wait Control 1 and 0 (W71, W70):** These bits select the number of program wait states when area 7 in external space is accessed while the AST7 bit in ASTCR is set to 1.

Bit 7 W71	Bit 6 W70	Description
0	0	Program wait not inserted when external space area 7 is accessed
	1	1 program wait state inserted when external space area 7 is accessed
1	0	2 program wait states inserted when external space area 7 is accessed
	1	3 program wait states inserted when external space area 7 is accessed (Initial value)

**Bits 5 and 4—Area 6 Wait Control 1 and 0 (W61, W60):** These bits select the number of program wait states when area 6 in external space is accessed while the AST6 bit in ASTCR is set to 1.

Bit 5 W61	Bit 4 W60	Description
0	0	Program wait not inserted when external space area 6 is accessed
	1	1 program wait state inserted when external space area 6 is accessed
1	0	2 program wait states inserted when external space area 6 is accessed
	1	3 program wait states inserted when external space area 6 is accessed (Initial value)

**Bits 3 and 2—Area 5 Wait Control 1 and 0 (W51, W50):** These bits select the number of program wait states when area 5 in external space is accessed while the AST5 bit in ASTCR is set to 1.

<b>Bit 3 W51</b>	<b>Bit 2 W50</b>	<b>Description</b>
0	0	Program wait not inserted when external space area 5 is accessed
	1	1 program wait state inserted when external space area 5 is accessed
1	0	2 program wait states inserted when external space area 5 is accessed
	1	3 program wait states inserted when external space area 5 is accessed (Initial value)

**Bits 1 and 0—Area 4 Wait Control 1 and 0 (W41, W40):** These bits select the number of program wait states when area 4 in external space is accessed while the AST4 bit in ASTCR is set to 1.

<b>Bit 1 W41</b>	<b>Bit 0 W40</b>	<b>Description</b>
0	0	Program wait not inserted when external space area 4 is accessed
	1	1 program wait state inserted when external space area 4 is accessed
1	0	2 program wait states inserted when external space area 4 is accessed
	1	3 program wait states inserted when external space area 4 is accessed (Initial value)

## WCRL

Bit	:	7	6	5	4	3	2	1	0
		W31	W30	W21	W20	W11	W10	W01	W00
Initial value :		1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Bits 7 and 6—Area 3 Wait Control 1 and 0 (W31, W30):** These bits select the number of program wait states when area 3 in external space is accessed while the AST3 bit in ASTCR is set to 1.

Bit 7 W31	Bit 6 W30	Description
0	0	Program wait not inserted when external space area 3 is accessed
	1	1 program wait state inserted when external space area 3 is accessed
1	0	2 program wait states inserted when external space area 3 is accessed
	1	3 program wait states inserted when external space area 3 is accessed (Initial value)

**Bits 5 and 4—Area 2 Wait Control 1 and 0 (W21, W20):** These bits select the number of program wait states when area 2 in external space is accessed while the AST2 bit in ASTCR is set to 1.

Bit 5 W21	Bit 4 W20	Description
0	0	Program wait not inserted when external space area 2 is accessed
	1	1 program wait state inserted when external space area 2 is accessed
1	0	2 program wait states inserted when external space area 2 is accessed
	1	3 program wait states inserted when external space area 2 is accessed (Initial value)

**Bits 3 and 2—Area 1 Wait Control 1 and 0 (W11, W10):** These bits select the number of program wait states when area 1 in external space is accessed while the AST1 bit in ASTCR is set to 1.

Bit 3 W11	Bit 2 W10	Description
0	0	Program wait not inserted when external space area 1 is accessed
	1	1 program wait state inserted when external space area 1 is accessed
1	0	2 program wait states inserted when external space area 1 is accessed
	1	3 program wait states inserted when external space area 1 is accessed (Initial value)

**Bits 1 and 0—Area 0 Wait Control 1 and 0 (W01, W00):** These bits select the number of program wait states when area 0 in external space is accessed while the AST0 bit in ASTCR is set to 1.

Bit 1 W01	Bit 0 W00	Description
0	0	Program wait not inserted when external space area 0 is accessed
	1	1 program wait state inserted when external space area 0 is accessed
1	0	2 program wait states inserted when external space area 0 is accessed
	1	3 program wait states inserted when external space area 0 is accessed (Initial value)

#### 4.2.4 Bus Control Register H (BCRH)

Bit	:	7	6	5	4	3	2	1	0
		ICIS1	ICIS0	BRSTRM	BRSTS1	BRSTS0	RMTS2	RMTS1	RMTS0
Initial value :		1	1	0	1	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BCRH is an 8-bit readable/writable register that selects enabling or disabling of idle cycle insertion, and the memory interface for areas 2 to 5 and area 0.

BCRH is initialized to H'D0 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Idle Cycle Insert 1 (ICIS1):** Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read cycles are performed in different areas.

Bit 7 ICIS1	Description
0	Idle cycle not inserted in case of successive external read cycles in different areas
1	Idle cycle inserted in case of successive external read cycles in different areas (Initial value)

**Bit 6—Idle Cycle Insert 0 (ICIS0):** Selects whether or not one idle cycle state is to be inserted between bus cycles when successive external read and external write cycles are performed .

Bit 6 ICIS0	Description
0	Idle cycle not inserted in case of successive external read and external write cycles
1	Idle cycle inserted in case of successive external read and external write cycles (Initial value)

**Bit 5—Burst ROM Enable (BRSTRM):** Selects whether area 0 is used as a burst ROM interface area.

Bit 5 BRSTRM	Description
0	Area 0 is basic bus interface area (Initial value)
1	Area 0 is burst ROM interface area

**Bit 4—Burst Cycle Select 1 (BRSTS1):** Selects the number of burst cycles for the burst ROM interface.

Bit 4 BRSTS1	Description
0	Burst cycle comprises 1 state
1	Burst cycle comprises 2 states (Initial value)

**Bit 3—Burst Cycle Select 0 (BRSTS0):** Selects the number of words that can be accessed in a burst ROM interface burst access.

**Bit 3**

BRSTS0	Description	
0	Max. 4 words in burst access	(Initial value)
1	Max. 8 words in burst access	

**Bits 2 to 0—RAM Type Select (RMTS2 to RMTS0):** These bits select the memory interface for areas 2 to 5 in advanced mode.

When DRAM space is selected, the relevant area is designated as a DRAM interface area.

Bit 2 RMTS2	Bit 1 RMTS1	Bit 0 RMTS0	Description Area 5	Area 4	Area 3	Area 2
0	0	0	Normal space			
		1	Normal space			DRAM space
	1	0	Normal space		DRAM space	
		1	DRAM space			
1	—	—	—			

Note: The  $\overline{\text{LCAS}}$  pin is used for the  $\overline{\text{LCAS}}$  signal on the 2-CAS DRAM interface. If it is wished to use  $\overline{\text{BREQO}}$  output and  $\overline{\text{WAIT}}$  input when using the  $\overline{\text{LCAS}}$  signal, use with another pin can be specified by means of the  $\overline{\text{WAITPS}}$  and  $\overline{\text{BREQOPS}}$  bits in  $\text{PFCR2}$ . Switching of these pins differs from model to model; see the reference manual for the relevant model for details.

**4.2.5 Bus Control Register L (BCRL)**

Bit	:	7	6	5	4	3	2	1	0
		BRLE	BREQOE	EAE	—	DDS	—	WDBE	WAITE
Initial value	:	0	0	1	1	1	1	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BCRL is an 8-bit readable/writable register that performs selection of the external bus-released state protocol, DMAC single address transfer, enabling or disabling of the write data buffer function, and enabling or disabling of  $\overline{\text{WAIT}}$  pin input.

BCRL is initialized to H'3C by a reset and in hardware standby mode. It is not initialized in software standby mode.



**Bit 7—Bus Release Enable (BRLE):** Enables or disables external bus release.

Bit 7	
BRLE	Description
0	External bus release is disabled. $\overline{\text{BREQ}}$ , $\overline{\text{BACK}}$ , and $\overline{\text{BREQO}}$ pins can be used as I/O ports (Initial value)
1	External bus release is enabled

**Bit 6—BREQO Pin Enable (BREQOE):** Outputs a signal that requests the external bus master to drop the bus request signal ( $\overline{\text{BREQ}}$ ) in the external bus release state, when an internal bus master performs an external space access, or when a refresh request is generated.

Bit 6	
BREQOE	Description
0	$\overline{\text{BREQO}}$ output disabled. $\overline{\text{BREQO}}$ pin can be used as I/O port (Initial value)
1	$\overline{\text{BREQO}}$ output enabled

**Bit 5—External Address Enable (EAE):** Selects whether addresses H'010000 to H'03FFFF\*<sup>1</sup> are to be internal addresses or external addresses.

Bit 5	
EAE	Description
0	Addresses H'010000 to H'03FFFF* <sup>1</sup> are in on-chip ROM
1	Addresses H'010000 to H'03FFFF* <sup>1</sup> are external addresses (external expansion mode) or a reserved area* <sup>2</sup> (single-chip mode) (Initial value)

- Notes:
1. The on-chip ROM area differs from model to model; see the reference manual for the relevant model for details.
  2. Reserved areas should not be accessed.

#### Bit 4—Reserved

**Bit 3—DACK Timing Select (DDS):** Selects the DMAC single address transfer bus timing for the DRAM interface.

#### Bit 3

DDS	Description
0	When DMAC single address transfer is performed in DRAM space, full access is always executed $\overline{\text{DACK}}$ signal goes low from $T_r$ or $T_1$ cycle
1	Burst access is possible when DMAC single address transfer is performed in DRAM space $\overline{\text{DACK}}$ signal goes low from $T_{c1}$ or $T_2$ cycle (Initial value)

#### Bit 2—Reserved

**Bit 1—Write Data Buffer Enable (WDBE):** Selects whether or not the write buffer function is used for an external write cycle or DMAC single address cycle.

#### Bit 1

WDBE	Description
0	Write data buffer function not used (Initial value)
1	Write data buffer function used

**Bit 0—WAIT Pin Enable (WAITE):** Selects enabling or disabling of wait input by the  $\overline{\text{WAIT}}$  pin.

#### Bit 0

WAITE	Description
0	Wait input by $\overline{\text{WAIT}}$ pin disabled. $\overline{\text{WAIT}}$ pin can be used as I/O port (Initial value)
1	Wait input by $\overline{\text{WAIT}}$ pin enabled

#### 4.2.6 Memory Control Register (MCR)

Bit	:	7	6	5	4	3	2	1	0
		TPC	BE	RCDM	–	MXC1	MXC0	RLW1	RLW0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MCR is an 8-bit readable/writable register that selects the DRAM strobe control method, number of precharge cycles, access mode, address multiplexing shift size, and the number of wait states inserted during refreshing, when areas 2 to 5 are designated as DRAM interface areas.

MCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—TP Cycle Control (TPC):** Selects whether a 1-state or 2-state precharge cycle ( $T_p$ ) is to be used when areas 2 to 5 designated as DRAM space are accessed.

Bit 7	
TPC	Description
0	1-state precharge cycle is inserted (Initial value)
1	2-state precharge cycle is inserted

**Bit 6—Burst Access Enable (BE):** Selects enabling or disabling of burst access to areas 2 to 5 designated as DRAM space. DRAM space burst access is performed in fast page mode.

Bit 6	
BE	Description
0	Burst disabled (always full access) (Initial value)
1	For DRAM space access, access in fast page mode

**Bit 5—RAS Down Mode (RCDM):** When areas 2 to 5 are designated as DRAM space and access to DRAM is interrupted, RCDM selects whether the next DRAM access is waited for with the  $\overline{\text{RAS}}$  signal held low (RAS down mode), or the  $\overline{\text{RAS}}$  signal is driven high again (RAS up mode).

Bit 5	
RCDM	Description
0	DRAM interface: RAS up mode selected (Initial value)
1	DRAM interface: RAS down mode selected

## Bit 4—Reserved

**Bits 3 and 2—Multiplex Shift Count 1 and 0 (MXC1, MXC0):** These bits select the size of the shift to the lower half of the row address in row address/column address multiplexing for the DRAM interface. In burst operation on the DRAM interface, these bits also select the row address to be used for comparison.

Bit 3 MXC1	Bit 2 MXC0	Description
0	0	8-bit shift (Initial value) <ul style="list-style-type: none"><li>• When 8-bit access space is designated: Row address <math>A_{23}</math> to <math>A_8</math> used for comparison</li><li>• When 16-bit access space is designated: Row address <math>A_{23}</math> to <math>A_9</math> used for comparison</li></ul>
	1	9-bit shift <ul style="list-style-type: none"><li>• When 8-bit access space is designated: Row address <math>A_{23}</math> to <math>A_9</math> used for comparison</li><li>• When 16-bit access space is designated: Row address <math>A_{23}</math> to <math>A_{10}</math> used for comparison</li></ul>
1	0	10-bit shift <ul style="list-style-type: none"><li>• When 8-bit access space is designated: Row address <math>A_{23}</math> to <math>A_{10}</math> used for comparison</li><li>• When 16-bit access space is designated: Row address <math>A_{23}</math> to <math>A_{11}</math> used for comparison</li></ul>
	1	—

**Bits 1 and 0—Refresh Cycle Wait Control 1 and 0 (RLW1, RLW0):** These bits select the number of wait states to be inserted in a DRAM interface CAS-before-RAS refresh cycle. This setting is used for all areas designated as DRAM space. Wait input by the  $\overline{\text{WAIT}}$  pin is disabled.

Bit 1 RLW1	Bit 0 RLW0	Description
0	0	No wait state inserted (Initial value)
	1	1 wait state inserted
1	0	2 wait states inserted
	1	3 wait states inserted

## 4.2.7 DRAM Control Register (DRAMCR)

Bit	:	7	6	5	4	3	2	1	0
		RFSHE	RCW	RMODE	CMF	CMIE	CKS2	CKS1	CKS0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DRAMCR is an 8-bit readable/writable register that selects the DRAM refresh mode and refresh counter clock, and controls the refresh timer.

DRAMCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Refresh Control (RFSHE):** Selects whether or not refresh control is performed. When refresh control is not performed, the refresh timer can be used as an interval timer.

Bit 7	
RFSHE	Description
0	Refresh control is not performed (Initial value)
1	Refresh control is performed

**Bit 6—RAS-CAS Wait (RCW):** Controls wait state insertion in DRAM interface CAS-before-RAS refreshing.

Bit 6	
RCW	Description
0	Wait state insertion in CAS-before-RAS refreshing disabled (Initial value) $\overline{\text{RAS}}$ falls in $T_{Rf}$ cycle
1	One wait state inserted in CAS-before-RAS refreshing $\overline{\text{RAS}}$ falls in $T_{Rc1}$ cycle

**Bit 5—Refresh Mode (RMODE):** When refresh control is performed ( $\text{RFSHE} = 1$ ), selects whether or not self-refresh control is performed in software standby mode.

Bit 5	
RMODE	Description
0	Self-refreshing is not performed in software standby mode (Initial value)
1	Self-refreshing is performed in software standby mode

**Bit 4—Compare Match Flag (CMF):** Status flag that indicates a match between the values of RTCNT and RTCOR.

When refresh control is performed ( $\text{RFSHE} = 1$ ), 1 should be written to the CMF bit when writing to DRAMCR.

Bit 4 CMF	Description
0	[Clearing condition] Cleared by reading the CMF flag when CMF = 1, then writing 0 to the CMF flag (Initial value)
1	[Setting condition] Set when RTCNT = RTCOR

**Bit 3—Compare Match Interrupt Enable (CMIE):** Enables or disables interrupt requests (CMI) by the CMF flag when the CMF flag in DRAMCR is set to 1.

When refresh control is performed (RFSHE = 1), the CMIE bit is always cleared to 0.

Bit 3 CMIE	Description
0	Interrupt request (CMI) by CMF flag disabled (Initial value)
1	Interrupt request (CMI) by CMF flag enabled

**Bits 2 to 0—Refresh Counter Clock Select (CKS2 to CKS0):** These bits select the clock to be input to RTCNT from among 7 internal clocks obtained by dividing the system clock ( $\phi$ ). When the input clock is selected with bits CKS2 to CKS0, RTCNT begins counting up.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description
0	0	0	Count operation disabled (Initial value)
		1	Count uses $\phi/2$
	1	0	Count uses $\phi/8$
		1	Count uses $\phi/32$
1	0	0	Count uses $\phi/128$
		1	Count uses $\phi/512$
	1	0	Count uses $\phi/2048$
		1	Count uses $\phi/4096$

#### 4.2.8 Refresh Timer Counter (RTCNT)

Bit	:	7	6	5	4	3	2	1	0
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RTCNT is an 8-bit readable/writable up-counter.

RTCNT counts up using the internal clock selected by bits CKS2 to CKS0 in DRAMCR.

When RTCNT matches RTCOR (compare match), the CMF flag in DRAMCR is set to 1 and RTCNT is cleared to H'00. If the RFSHE bit in DRAMCR is set to 1 at this time, a refresh cycle is started. Also, if the CMIE bit in DRAMCR is set to 1, a compare match interrupt (CMI) is generated.

RTCNT is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

#### 4.2.9 Refresh Time Constant Register (RTCOR)

Bit	:	7	6	5	4	3	2	1	0
Initial value :		1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RTCOR is an 8-bit readable/writable register that sets the period for compare match operations with RTCNT.

The values of RTCOR and RTCNT are constantly compared, and if they match, the CMF flag in DRAMCR is set to 1 and RTCNT is cleared to H'00.

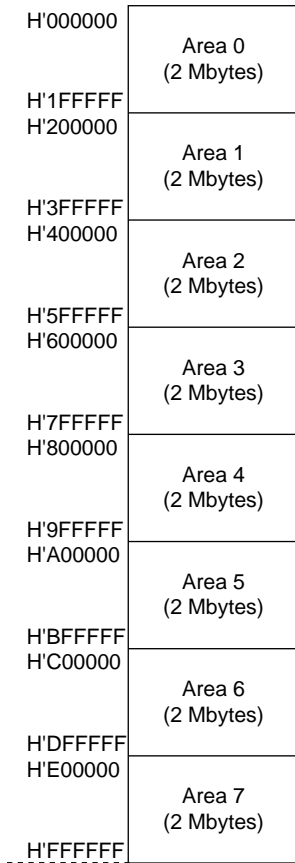
RTCOR is initialized to H'FF by a reset and in hardware standby mode. It is not initialized in software standby mode.

### 4.3 Overview of Bus Control

#### 4.3.1 Area Partitioning

In advanced mode, the bus controller partitions the 16-Mbyte address space into eight areas, 0 to 7, in 2-Mbyte units, and performs bus control for external space in area units. Figure 4-2 shows an outline of the memory map.

Chip select signals ( $\overline{CS0}$  to  $\overline{CS7}$ ) can be output for each area.



Advanced mode

**Figure 4-2 Overview of Area Partitioning**



### 4.3.2 Bus Specifications

The external space bus specifications consist of three elements: bus width, number of access states, and number of program wait states.

The bus width and number of access states for on-chip memory and internal I/O registers are fixed, and are not affected by the bus controller.

**Bus Width:** A bus width of 8 or 16 bits can be selected with ABWCR. An area for which an 8-bit bus is selected functions as an 8-bit access space, and an area for which a 16-bit bus is selected functions as a 16-bit access space.

If all areas are designated for 8-bit access, 8-bit bus mode is set; if any area is designated for 16-bit access, 16-bit bus mode is set. When the burst ROM interface is designated, 16-bit bus mode is always set.

**Number of Access States:** Two or three access states can be selected with ASTCR. An area for which 2-state access is selected functions as a 2-state access space, and an area for which 3-state access is selected functions as a 3-state access space.

With the DRAM interface and burst ROM interface, the number of access states may be determined without regard to ASTCR.

When 2-state access space is designated, wait insertion is disabled.

**Number of Program Wait States:** When 3-state access space is designated by ASTCR, the number of program wait states to be inserted automatically is selected with WCRH and WCRL. From 0 to 3 program wait states can be selected.

Table 4-3 shows the bus specifications for each basic bus interface area.

**Table 4-3 Bus Specifications for Each Area (Basic Bus Interface)**

ABWCR ABWn	ASTCR ASTn	WCRH, WCRL		Bus Specifications (Basic Bus Interface)		
		Wn1	Wn0	Bus Width	Access States	Program Wait States
0	0	—	—	16	2	0
	1	0	0		3	0
			1		1	
			0		2	
1	1	3				
1	0	—	—	8	2	0
	1	0	0		3	0
			1		1	
			0		2	
1	3					

### 4.3.3 Memory Interfaces

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series memory interfaces comprise a basic bus interface that allows direct connection of ROM, SRAM, and so on; a DRAM interface that allows direct connection of DRAM; and a burst ROM interface that allows direct connection of burst ROM. The interface can be selected independently for each area.

An area for which the basic bus interface is designated functions as normal space, an area for which the DRAM interface is designated functions as DRAM space, and an area for which the burst ROM interface is designated functions as burst ROM space.

#### 4.3.4 Advanced Mode

The initial state of each area is basic bus interface, 3-state access space. The initial bus width is selected according to the operating mode. The bus specifications described here cover basic items only, and the sections on each memory interface (4.4, 4.5, and 4.7) should be referred to for further details.

**Area 0:** Area 0 includes on-chip ROM\*, and in ROM-disabled expansion mode, all of area 0 is external space. In the ROM-enabled expansion mode, the space excluding on-chip ROM\* is external space.

When area 0 external space is accessed, the  $\overline{CS0}$  signal can be output.

Either basic bus interface or burst ROM interface can be selected for area 0.

Note: \* Only applies to versions with ROM.

**Areas 1 and 6:** In external expansion mode, all of area 1 and area 6 is external space.

When area 1 and 6 external space is accessed, the  $\overline{CS1}$  and  $\overline{CS6}$  pin signals respectively can be output.

Only the basic bus interface can be used for areas 1 and 6.

**Areas 2 to 5:** In external expansion mode, all of area 2 to area 5 is external space.

When area 2 to 5 external space is accessed, signals  $\overline{CS2}$  to  $\overline{CS5}$  can be output.

Basic bus interface or DRAM interface can be selected for areas 2 to 5. With the DRAM interface, signals  $\overline{CS2}$  to  $\overline{CS5}$  are used as  $\overline{RAS}$  signals.

**Area 7:** Area 7 includes the on-chip RAM and internal I/O registers. In external expansion mode, the space excluding the on-chip RAM and internal I/O registers is external space. The on-chip RAM is enabled when the RAME bit in the system control register (SYSCR) is set to 1; when the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding space becomes external space.

When area 7 external space is accessed, the  $\overline{CS7}$  signal can be output.

Only the basic bus interface can be used for the area 7 memory interface.

### 4.3.5 Chip Select Signals

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series can output chip select signals ( $\overline{CS0}$  to  $\overline{CS7}$ ) to areas 0 to 7, the signal being driven low when the corresponding external space area is accessed.

Figure 4-3 shows an example of  $\overline{CSn}$  ( $n = 0$  to 7) output timing.

Enabling or disabling of the  $\overline{CSn}$  signal is performed by setting the data direction register (DDR) for the port corresponding to the particular  $\overline{CSn}$  pin.

In ROM-disabled expansion mode, the  $\overline{CS0}$  pin is placed in the output state after a power-on reset. Pins  $\overline{CS1}$  to  $\overline{CS7}$  are placed in the input state after a power-on reset, and so the corresponding DDR bits should be set to 1 when outputting signals  $\overline{CS1}$  to  $\overline{CS7}$ .

In the ROM-enabled expansion mode, pins  $\overline{CS0}$  to  $\overline{CS7}$  are all placed in the input state after a power-on reset, and so the corresponding DDR bits should be set to 1 when outputting signals  $\overline{CS0}$  to  $\overline{CS7}$ .

For details, see the Reference Manual, I/O Ports section.

When areas 2 to 5 are designated as DRAM space, outputs  $\overline{CS2}$  to  $\overline{CS5}$  are used as  $\overline{RAS}$  signals.

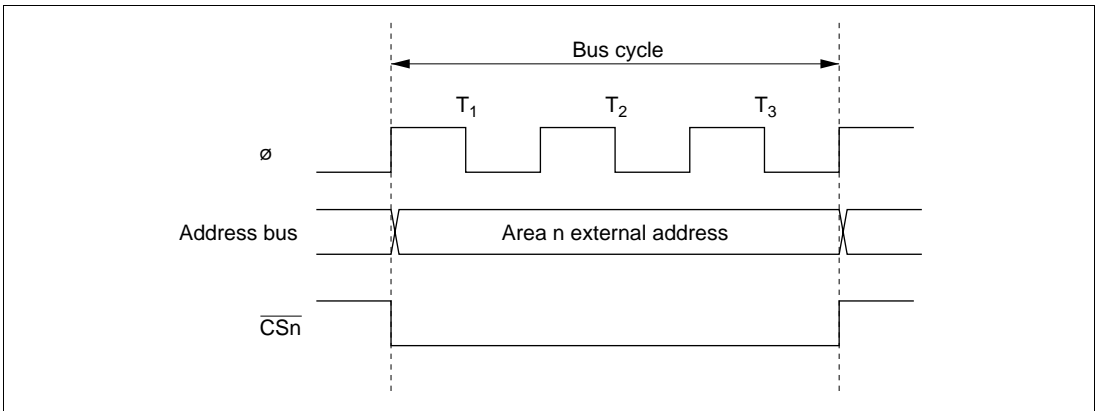


Figure 4-3  $\overline{CSn}$  Signal Output Timing ( $n = 0$  to 7)

## 4.4 Basic Bus Interface

### 4.4.1 Overview

The basic bus interface enables direct connection of ROM, SRAM, and so on.

The bus specifications can be selected with ABWCR, ASTCR, WCRH, and WCRL (see table 4-3).

### 4.4.2 Data Size and Data Alignment

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and when accessing external space, controls whether the upper data bus ( $D_{15}$  to  $D_8$ ) or lower data bus ( $D_7$  to  $D_0$ ) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space) and the data size.

**8-Bit Access Space:** Figure 4-4 illustrates data alignment control for the 8-bit access space. With the 8-bit access space, the upper data bus ( $D_{15}$  to  $D_8$ ) is always used for accesses. The amount of data that can be accessed at one time is one byte: a word transfer instruction is performed as two byte accesses, and a longword transfer instruction, as four byte accesses.

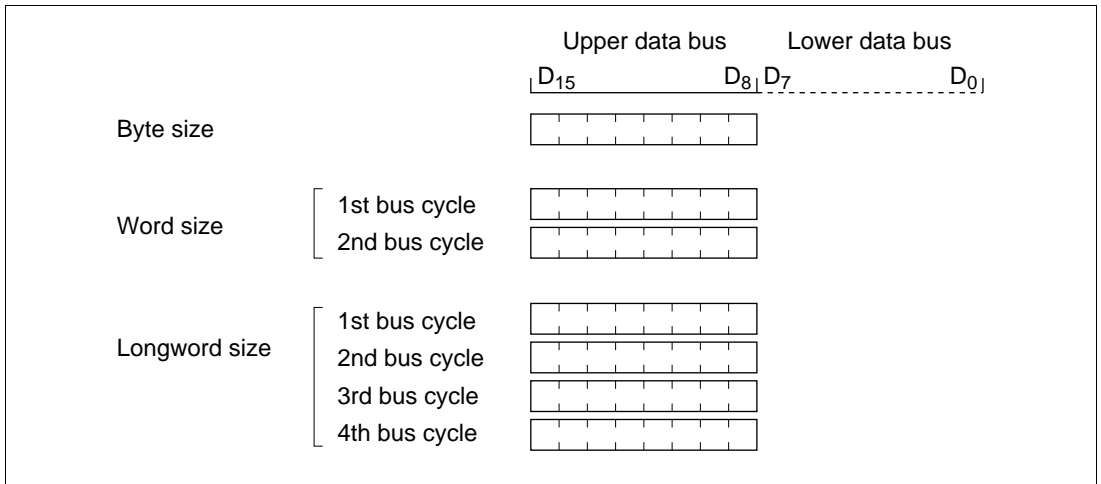
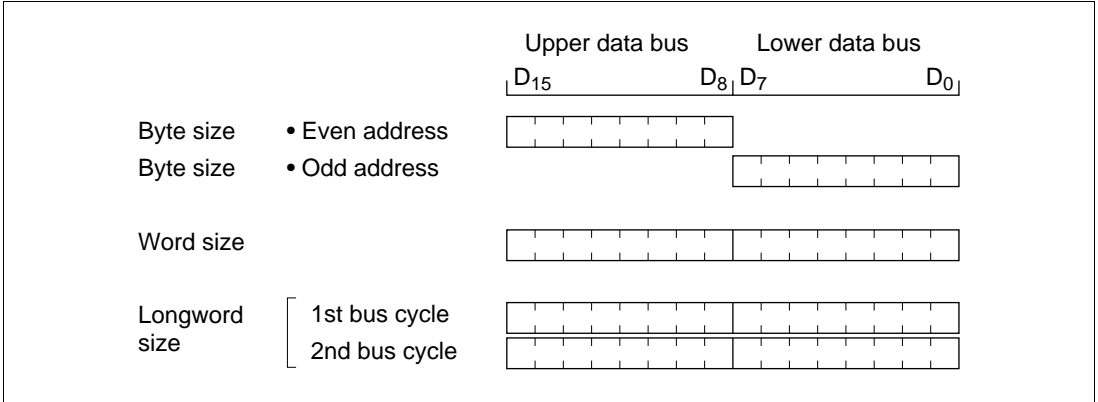


Figure 4-4 Access Sizes and Data Alignment Control (8-Bit Access Space)

**16-Bit Access Space:** Figure 4-5 illustrates data alignment control for the 16-bit access space. With the 16-bit access space, the upper data bus ( $D_{15}$  to  $D_8$ ) and lower data bus ( $D_7$  to  $D_0$ ) are used for accesses. The amount of data that can be accessed at one time is one byte or one word, and a longword transfer instruction is executed as two word transfer instructions.

In byte access, whether the upper or lower data bus is used is determined by whether the address is even or odd. The upper data bus is used for an even address, and the lower data bus for an odd address.



**Figure 4-5 Access Sizes and Data Alignment Control (16-Bit Access Space)**

### 4.4.3 Valid Strobes

Table 4-4 shows the data buses used and valid strobes for the access spaces.

In a read, the  $\overline{RD}$  signal is valid without discrimination between the upper and lower halves of the data bus.

In a write, the  $\overline{HWR}$  signal is valid for the upper half of the data bus, and the  $\overline{LWR}$  signal for the lower half.

**Table 4-4 Data Buses Used and Valid Strobes**

Area	Access Size	Read/Write	Address	Valid Strobe	Upper Data Bus (D <sub>15</sub> to D <sub>8</sub> )	Lower Data Bus (D <sub>7</sub> to D <sub>0</sub> )
8-bit access space	Byte	Read	—	$\overline{RD}$	Valid	Invalid
		Write	—	$\overline{HWR}$		Hi-Z
16-bit access space	Byte	Read	Even	$\overline{RD}$	Valid	Invalid
			Odd		Invalid	Valid
	Write	Even	$\overline{HWR}$	Valid	Hi-Z	
		Odd	$\overline{LWR}$	Hi-Z	Valid	
Word	Read	—	$\overline{RD}$	Valid	Valid	
		Write	—	$\overline{HWR}, \overline{LWR}$	Valid	Valid

Note: Hi-Z: High impedance

Invalid: Input state; input value is ignored.

#### 4.4.4 Basic Timing

**8-Bit 2-State Access Space:** Figure 4-6 shows the bus timing for an 8-bit 2-state access space. When an 8-bit access space is accessed, the upper half ( $D_{15}$  to  $D_8$ ) of the data bus is used.

The  $\overline{LWR}$  pin is fixed high. Wait states cannot be inserted.

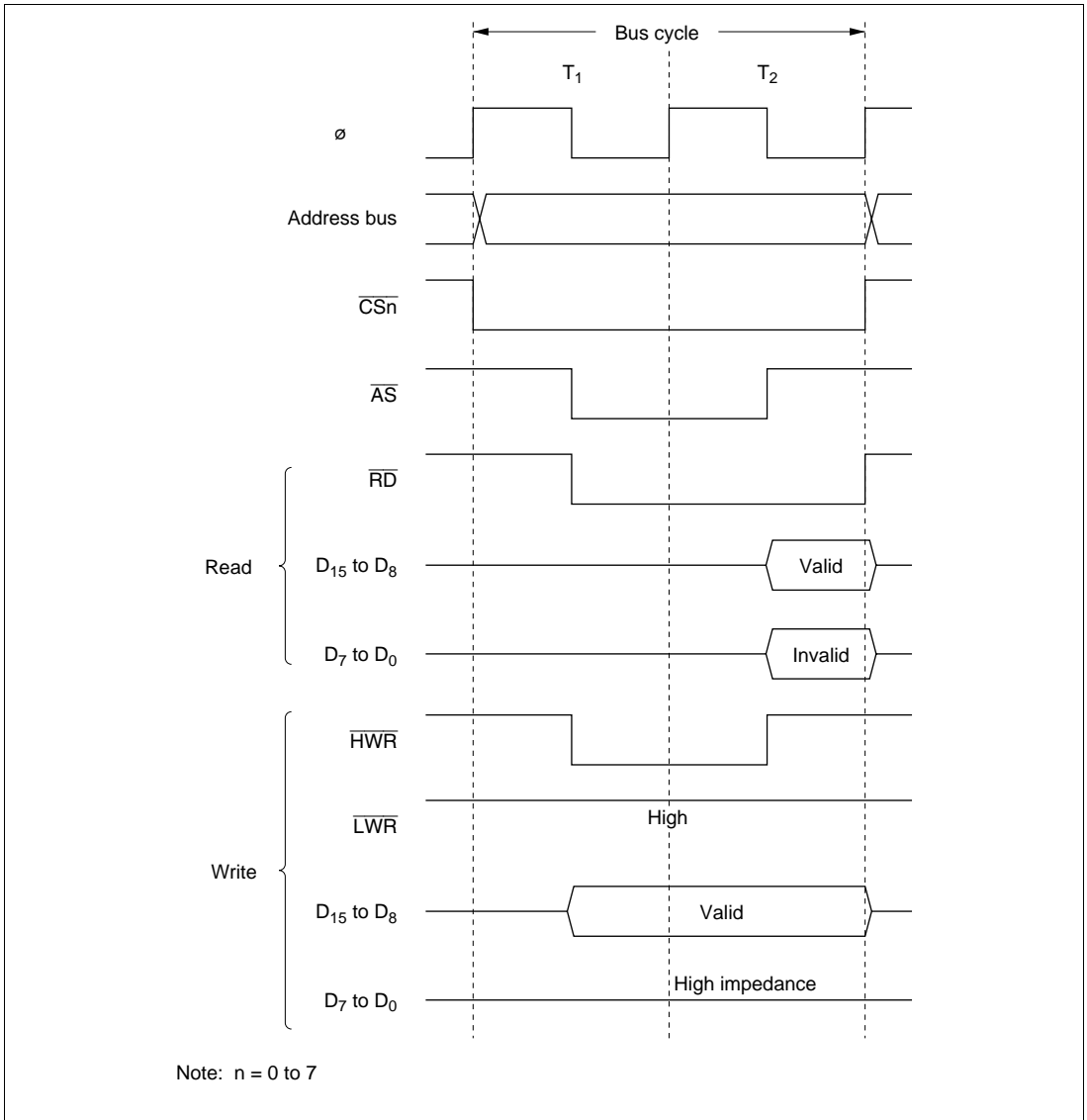
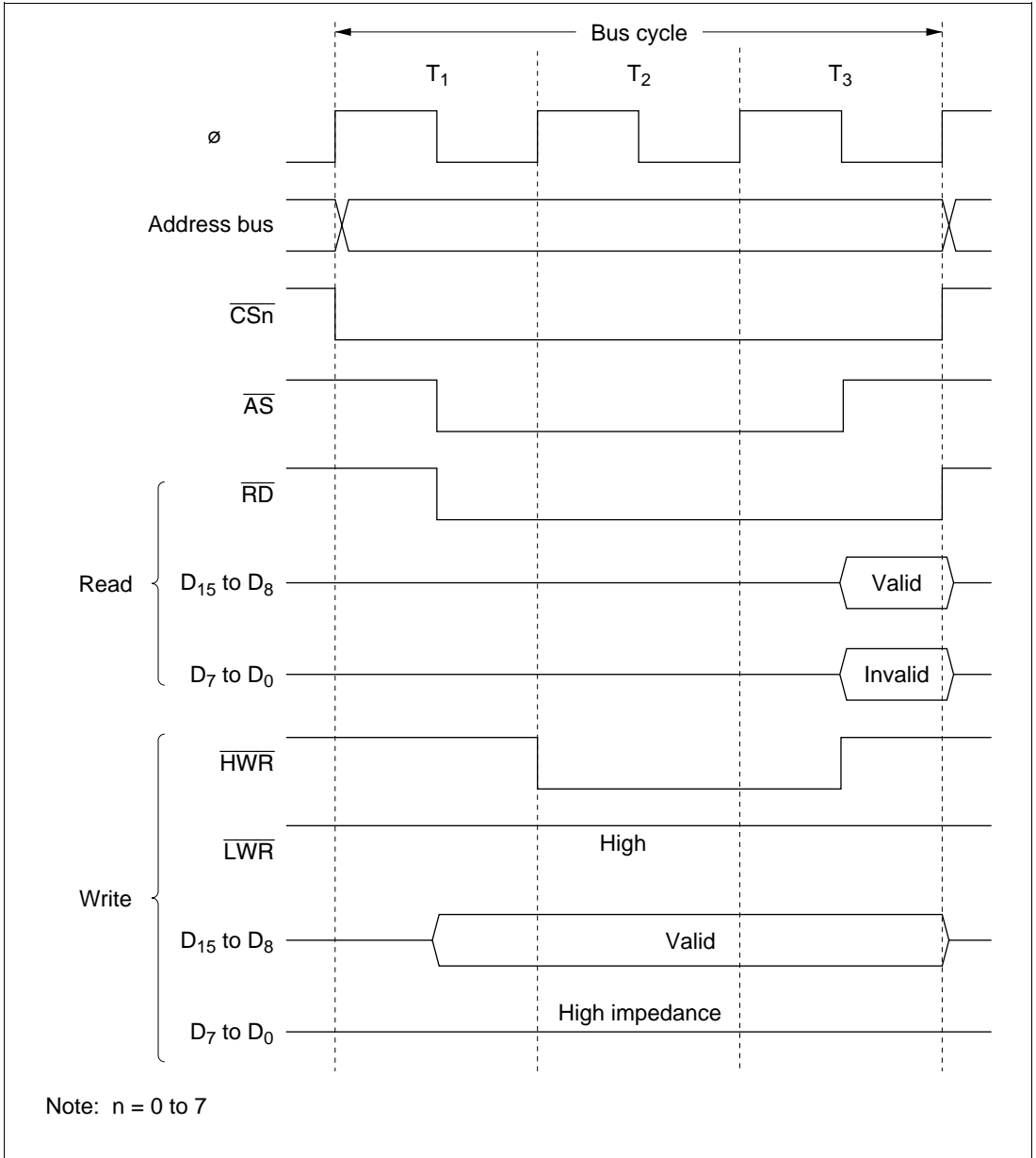


Figure 4-6 Bus Timing for 8-Bit 2-State Access Space



**8-Bit 3-State Access Space:** Figure 4-7 shows the bus timing for an 8-bit 3-state access space. When an 8-bit access space is accessed, the upper half ( $D_{15}$  to  $D_8$ ) of the data bus is used.

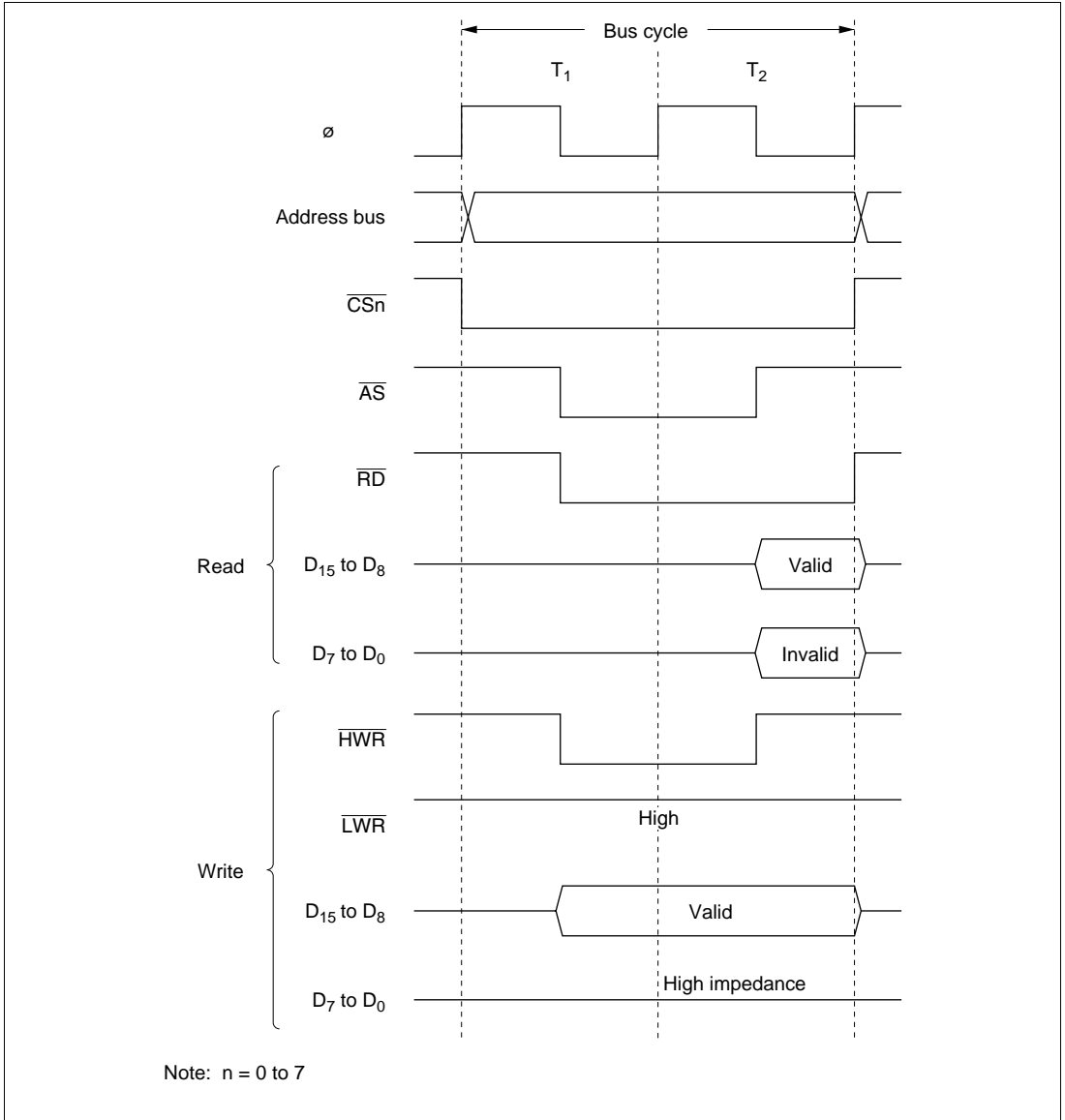
The  $\overline{LWR}$  pin is fixed high. Wait states can be inserted.



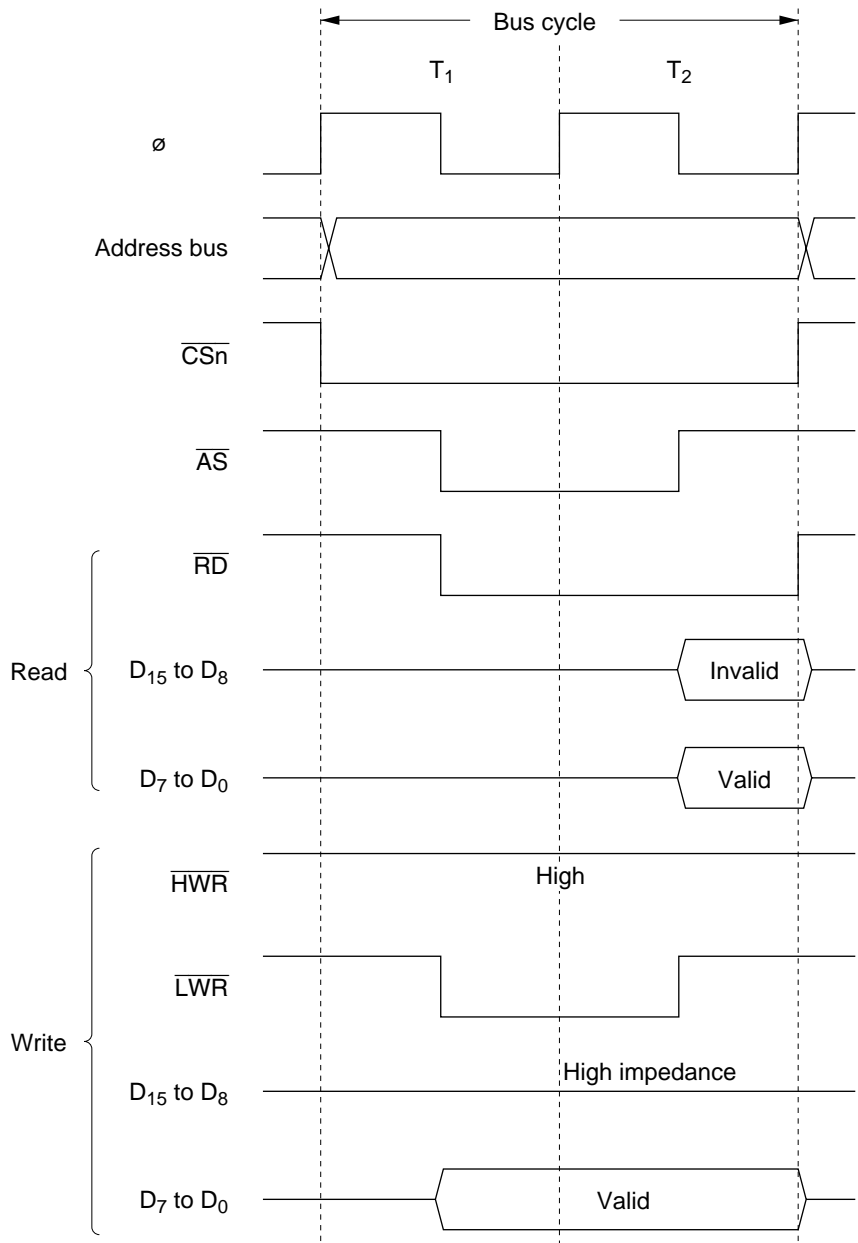
**Figure 4-7 Bus Timing for 8-Bit 3-State Access Space**

**16-Bit 2-State Access Space:** Figures 4-8 to 4-10 show bus timings for a 16-bit 2-state access space. When a 16-bit access space is accessed, the upper half ( $D_{15}$  to  $D_8$ ) of the data bus is used for the even address, and the lower half ( $D_7$  to  $D_0$ ) for the odd address.

Wait states cannot be inserted.

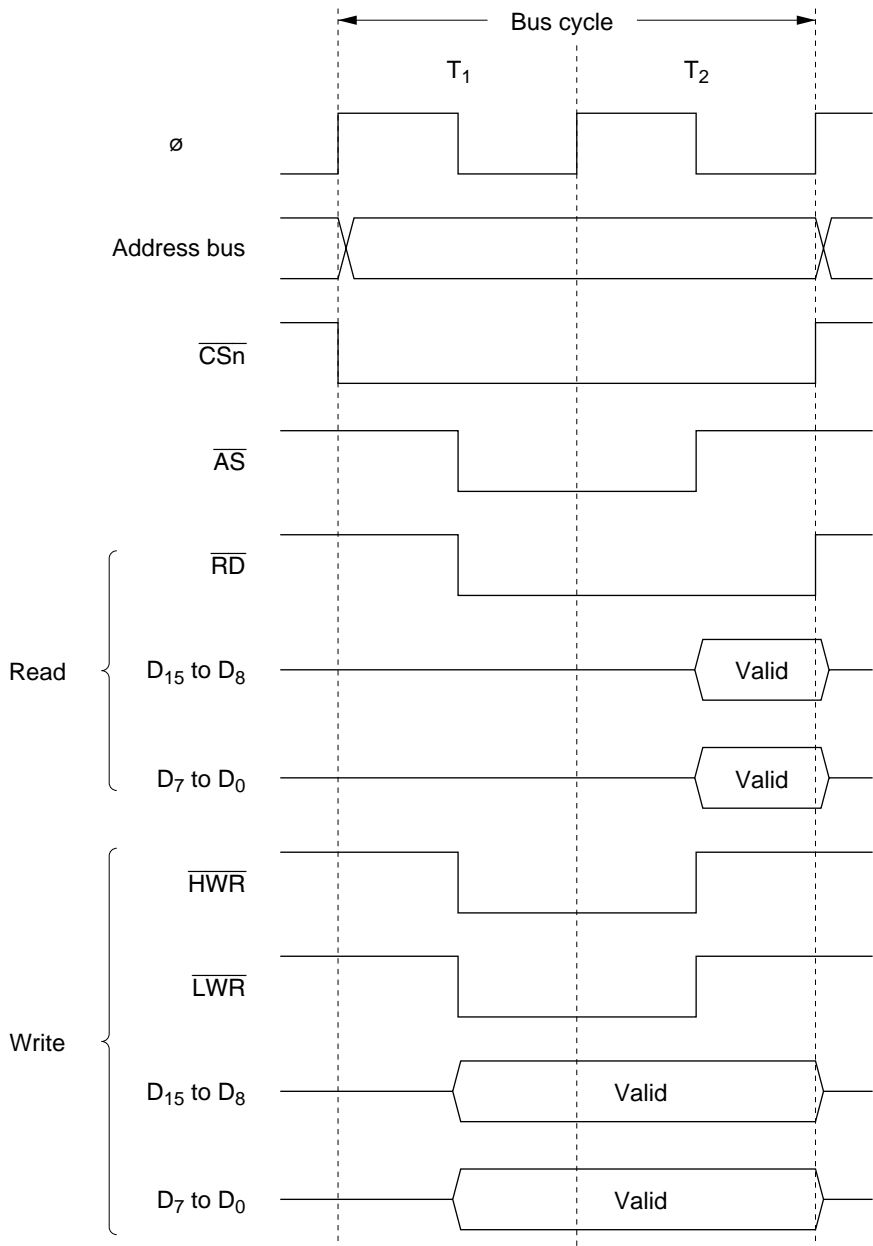


**Figure 4-8 Bus Timing for 16-Bit 2-State Access Space (1) (Even Address Byte Access)**



Note:  $n = 0$  to  $7$

Figure 4-9 Bus Timing for 16-Bit 2-State Access Space (2) (Odd Address Byte Access)

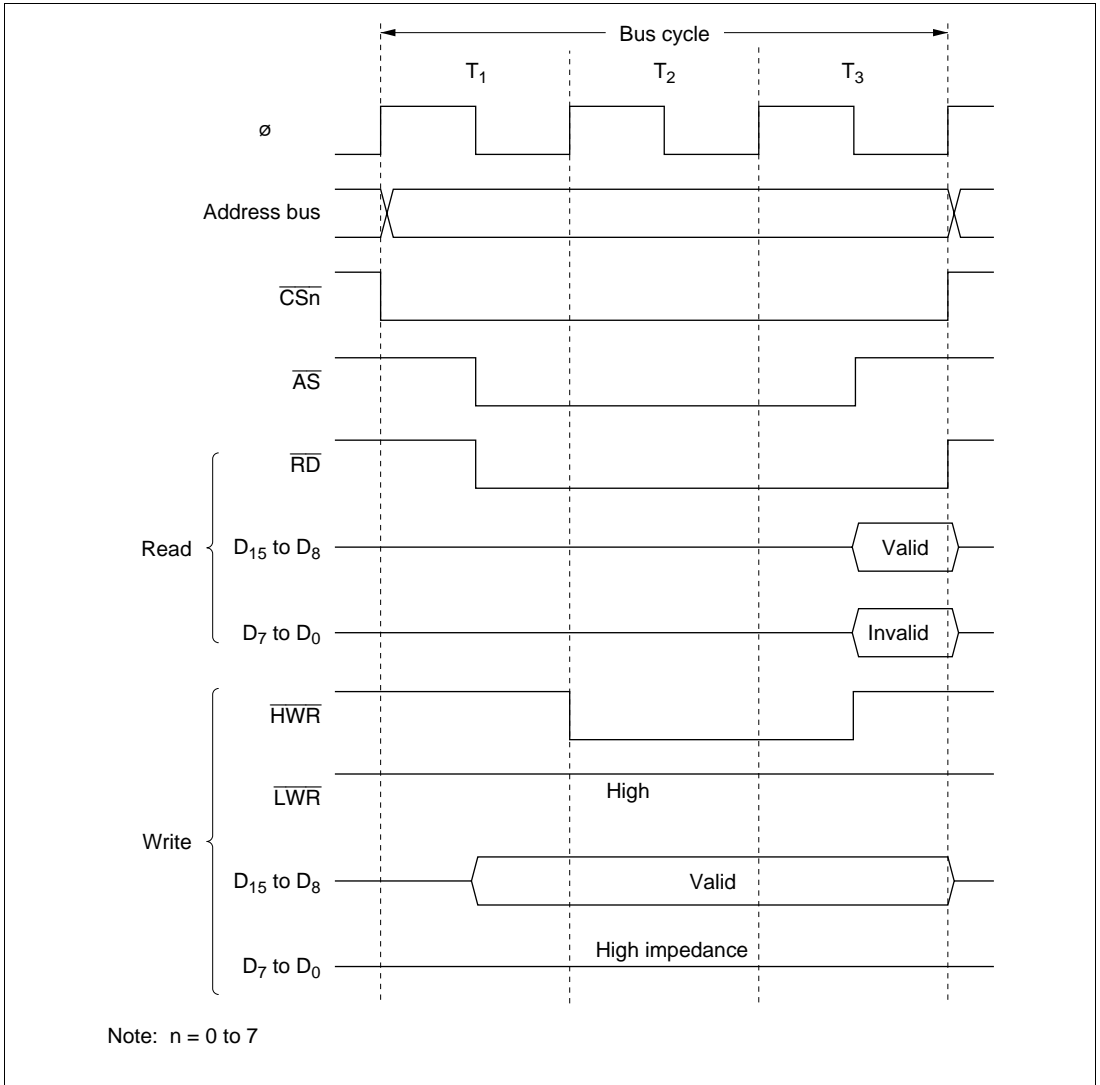


Note:  $n = 0$  to 7

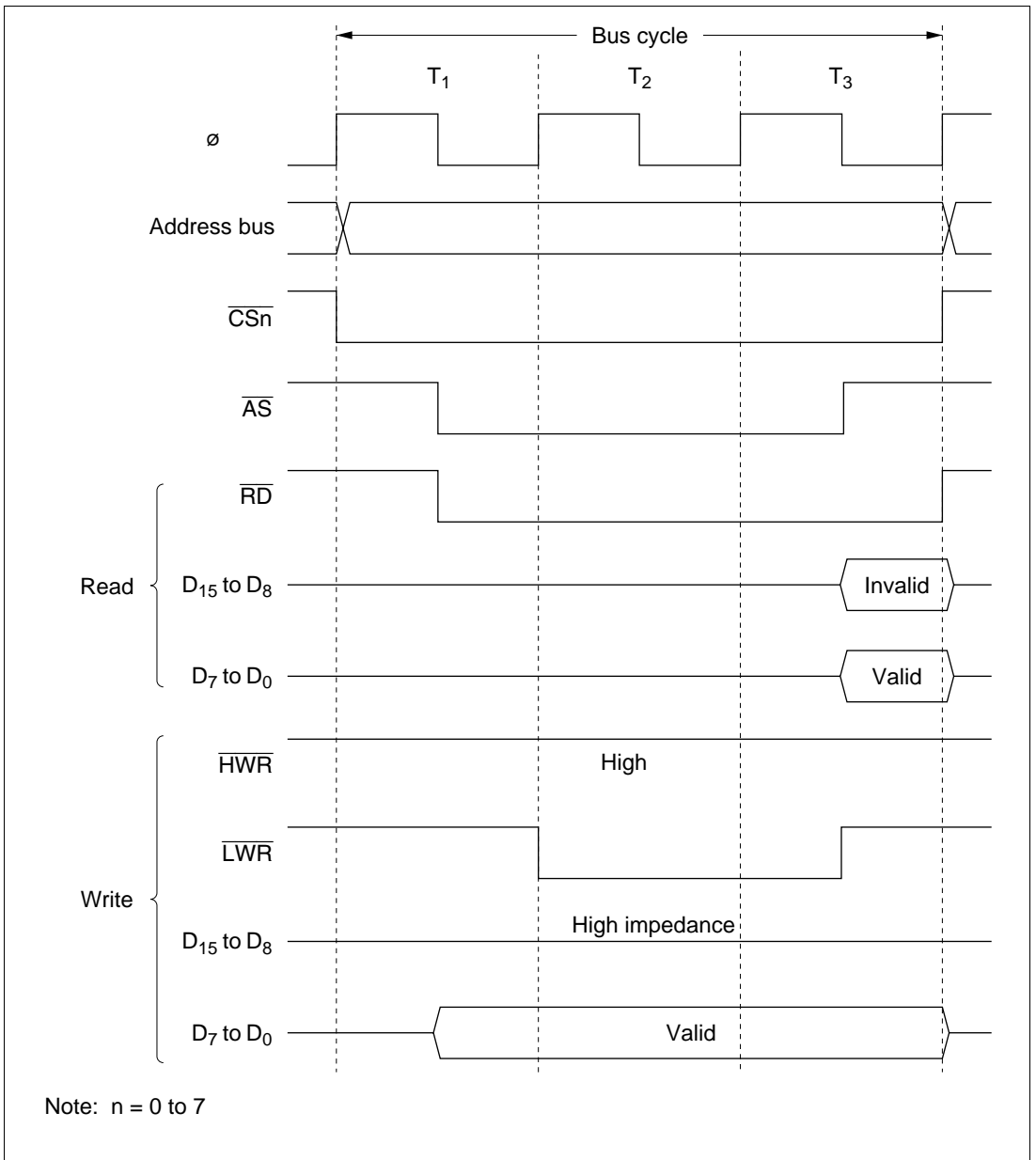
**Figure 4-10 Bus Timing for 16-Bit 2-State Access Space (3) (Word Access)**

**16-Bit 3-State Access Space:** Figures 4-11 to 4-13 show bus timings for a 16-bit 3-state access space. When a 16-bit access space is accessed, the upper half ( $D_{15}$  to  $D_8$ ) of the data bus is used for the even address, and the lower half ( $D_7$  to  $D_0$ ) for the odd address.

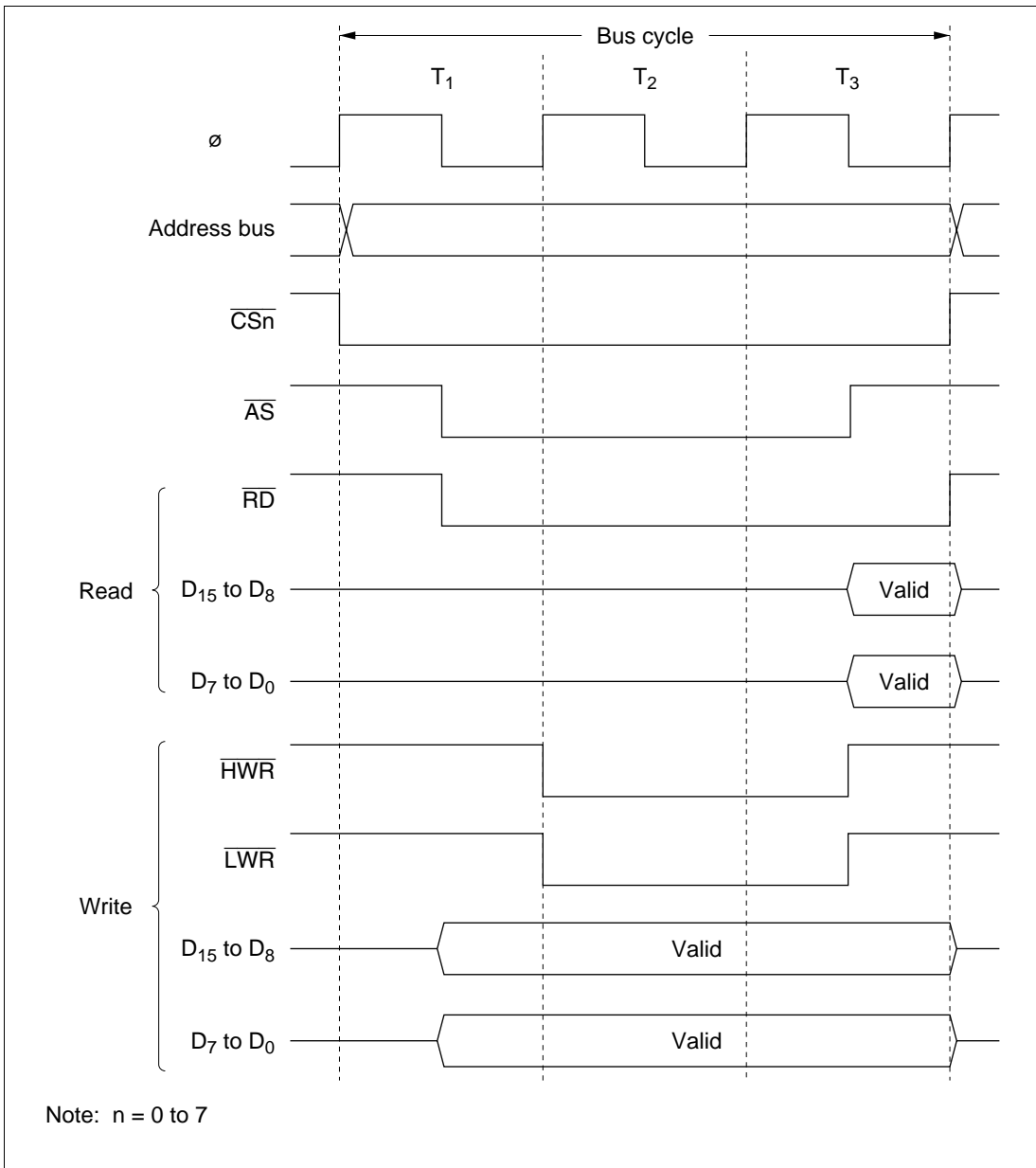
Wait states can be inserted.



**Figure 4-11 Bus Timing for 16-Bit 3-State Access Space (1) (Even Address Byte Access)**



**Figure 4-12 Bus Timing for 16-Bit 3-State Access Space (2) (Odd Address Byte Access)**



**Figure 4-13 Bus Timing for 16-Bit 3-State Access Space (3) (Word Access)**

#### 4.4.5 Wait Control

When accessing external space, the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series can extend the bus cycle by inserting one or more wait states ( $T_w$ ). There are two ways of inserting wait states: program wait insertion and pin wait insertion using the  $\overline{\text{WAIT}}$  pin.

**Program Wait Insertion:** From 0 to 3 wait states can be inserted automatically between the  $T_2$  state and  $T_3$  state on an individual area basis in 3-state access space, according to the settings of WCRH and WCRL.

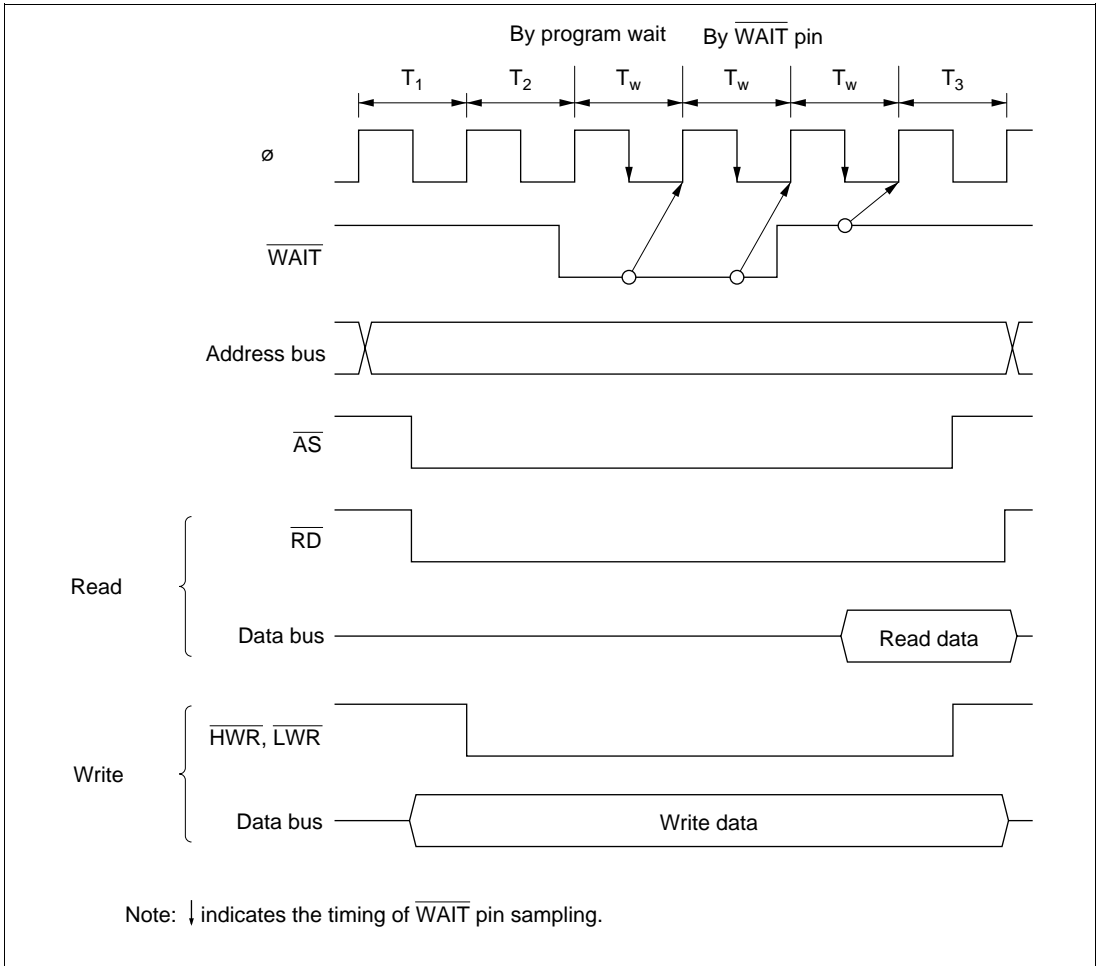
**Pin Wait Insertion:** Setting the WAITE bit in BCRL to 1 enables wait insertion by means of the  $\overline{\text{WAIT}}$  pin. When external space is accessed in this state, program wait insertion is first carried out according to the settings in WCRH and WCRL. Then, if the  $\overline{\text{WAIT}}$  pin is low at the falling edge of  $\phi$  in the last  $T_2$  or  $T_w$  state, a  $T_w$  state is inserted. If the  $\overline{\text{WAIT}}$  pin is held low,  $T_w$  states are inserted until it goes high.

This is useful when inserting four or more  $T_w$  states, or when changing the number of  $T_w$  states for different external devices.

The WAITE bit setting applies to all areas. With some models the pin that can be used for wait input can be switched by means of the WAITPS bit; see the reference manual for the relevant model to check the availability of this function.



Figure 4-14 shows an example of wait state insertion timing.



**Figure 4-14 Example of Wait State Insertion Timing**

The settings after a power-on reset are: 3-state access, 3 program wait state insertion, and WAIT input disabled.

## 4.5 DRAM Interface

### 4.5.1 Overview

When the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip is in advanced mode, external space areas 2 to 5 can be designated as DRAM space, and DRAM interfacing performed. With the DRAM interface, DRAM can be directly connected to the chip. A DRAM space of 2, 4, or 8 Mbytes can be set by means of bits RMTS2 to RMTS0 in BCRH. Burst operation is also possible, using fast page mode.

### 4.5.2 Setting DRAM Space

Areas 2 to 5 are designated as DRAM space by setting bits RMTS2 to RMTS0 in BCRH. The relation between the settings of bits RMTS2 to RMTS0 and DRAM space is shown in table 4-5. Possible DRAM space settings are: one area (area 2), two areas (areas 2 and 3), and four areas (areas 2 to 5).

**Table 4-5 Settings of Bits RMTS2 to RMTS0 and Corresponding DRAM Spaces**

RMTS2	RMTS1	RMTS0	Area 5	Area 4	Area 3	Area 2
0	0	1	Normal space			DRAM space
	1	0	Normal space			DRAM space
		1	DRAM space			

### 4.5.3 Address Multiplexing

With DRAM space, the row address and column address are multiplexed. In address multiplexing, the size of the shift of the row address is selected with bits MXC1 and MXC0 in MCR. Table 4-6 shows the relation between the settings of MXC1 and MXC0 and the shift size.

**Table 4-6 Address Multiplexing Settings by Bits MXC1 and MXC0**

	MCR		Shift Size	Address Pins															
	MXC1	MXC0		A <sub>23</sub> to A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>	A <sub>9</sub>	A <sub>8</sub>	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>		
Row address	0	0	8 bits	A <sub>23</sub> to A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>	A <sub>9</sub>	A <sub>8</sub>	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>		
		1	9 bits	A <sub>23</sub> to A <sub>13</sub>	A <sub>20</sub>	A <sub>19</sub>	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	A <sub>15</sub>	A <sub>14</sub>	A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>	A <sub>9</sub>	A <sub>8</sub>		
	1	0	10 bits	A <sub>23</sub> to A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>20</sub>	A <sub>19</sub>	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	A <sub>15</sub>	A <sub>14</sub>	A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>	A <sub>9</sub>	
		1	Setting prohibited	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	
Column address	—	—	—	A <sub>23</sub> to A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>	A <sub>9</sub>	A <sub>8</sub>	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>		

#### 4.5.4 Data Bus

If the bit in ABWCR corresponding to an area designated as DRAM space is set to 1, that area is designated as 8-bit DRAM space; if the bit is cleared to 0, the area is designated as 16-bit DRAM space. In 16-bit DRAM space, ×16-bit configuration DRAM can be connected directly.

In 8-bit DRAM space the upper half of the data bus,  $D_{15}$  to  $D_8$ , is enabled, while in 16-bit DRAM space both the upper and lower halves of the data bus,  $D_{15}$  to  $D_0$ , are enabled.

Access sizes and data alignment are the same as for the basic bus interface: see section 4.4.2, Data Size and Data Alignment.

#### 4.5.5 Pins Used for DRAM Interface

Table 4-7 shows the pins used for DRAM interfacing and their functions.

**Table 4-7 DRAM Interface Pins**

Pin	With DRAM Setting	Name	I/O	Function
HWR	$\overline{WE}$	Write enable	Output	When 2-CAS system is set, write enable for DRAM space access
$\overline{LCAS}$	$\overline{LCAS}$	Lower column address strobe	Output	Lower column address strobe for 16-bit DRAM space access
$\overline{CS2}$	$\overline{RAS2}$	Row address strobe 2	Output	Row address strobe when area 2 is designated as DRAM space
$\overline{CS3}$	$\overline{RAS3}$	Row address strobe 3	Output	Row address strobe when area 3 is designated as DRAM space
$\overline{CS4}$	$\overline{RAS4}$	Row address strobe 4	Output	Row address strobe when area 4 is designated as DRAM space
$\overline{CS5}$	$\overline{RAS5}$	Row address strobe 5	Output	Row address strobe when area 5 is designated as DRAM space
$\overline{CAS}$	$\overline{UCAS}$	Upper column address strobe	Output	Upper column address strobe for DRAM space access
$\overline{WAIT}$	$\overline{WAIT}$	Wait	Input	Wait request signal
$A_{12}$ to $A_0$	$A_{12}$ to $A_0$	Address pins	Output	Row address/column address multiplexed output
$D_{15}$ to $D_0$	$D_{15}$ to $D_0$	Data pins	I/O	Data input/output pins

## 4.5.6 Basic Timing

Figure 4-15 shows the basic access timing for DRAM space. The basic DRAM access timing is 4 states. Unlike the basic bus interface, the corresponding bits in ASTCR control only enabling or disabling of wait insertion, and do not affect the number of access states. When the corresponding bit in ASTCR is cleared to 0, wait states cannot be inserted in the DRAM access cycle.

The 4 states of the basic timing consist of one  $T_p$  (precharge cycle) state, one  $T_r$  (row address output cycle), and two  $T_c$  (column address output cycle) states,  $T_{c1}$  and  $T_{c2}$ .

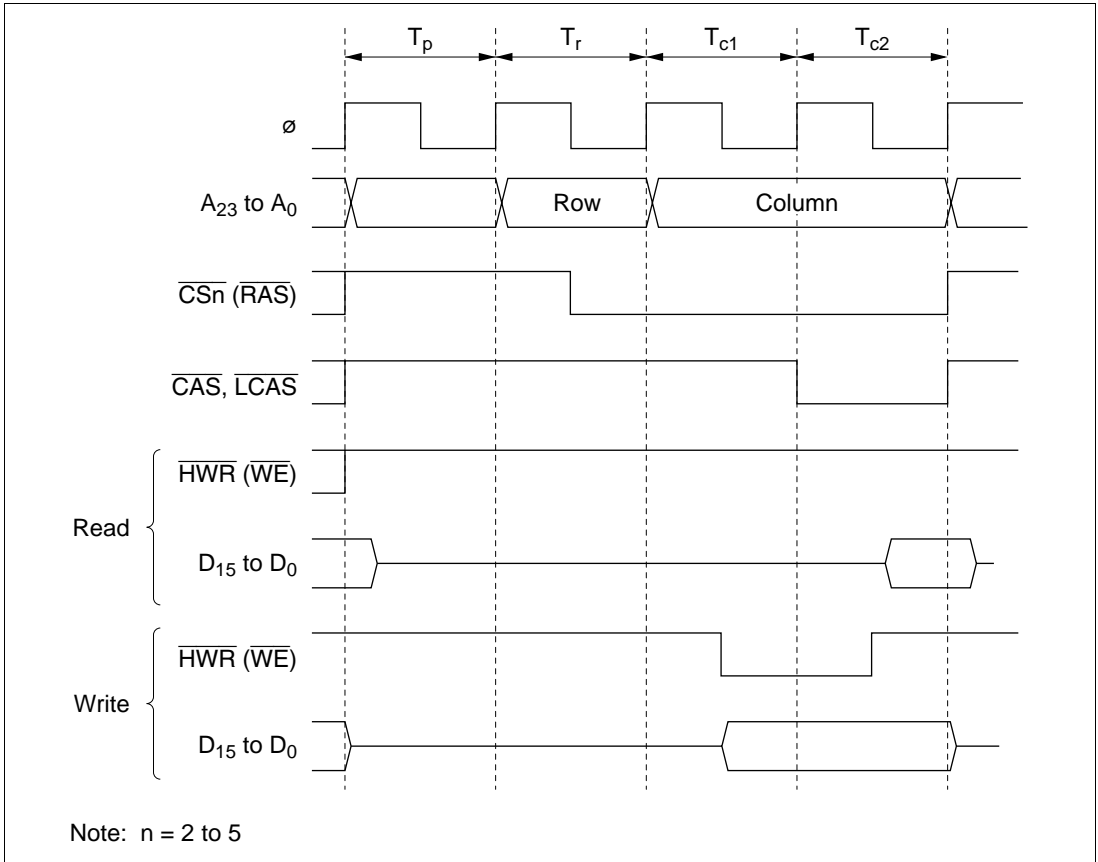


Figure 4-15 Basic Access Timing

### 4.5.7 Precharge State Control

When DRAM is accessed, an RAS precharging time must be secured. With the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, one  $T_p$  state is always inserted when DRAM space is accessed. This can be changed to two  $T_p$  states by setting the TPC bit in MCR to 1. Set the appropriate number of  $T_p$  cycles according to the DRAM connected and the operating frequency of the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip. Figure 4-16 shows the timing when two  $T_p$  states are inserted.

When the TCP bit is set to 1, two  $T_p$  states are also used for refresh cycles.

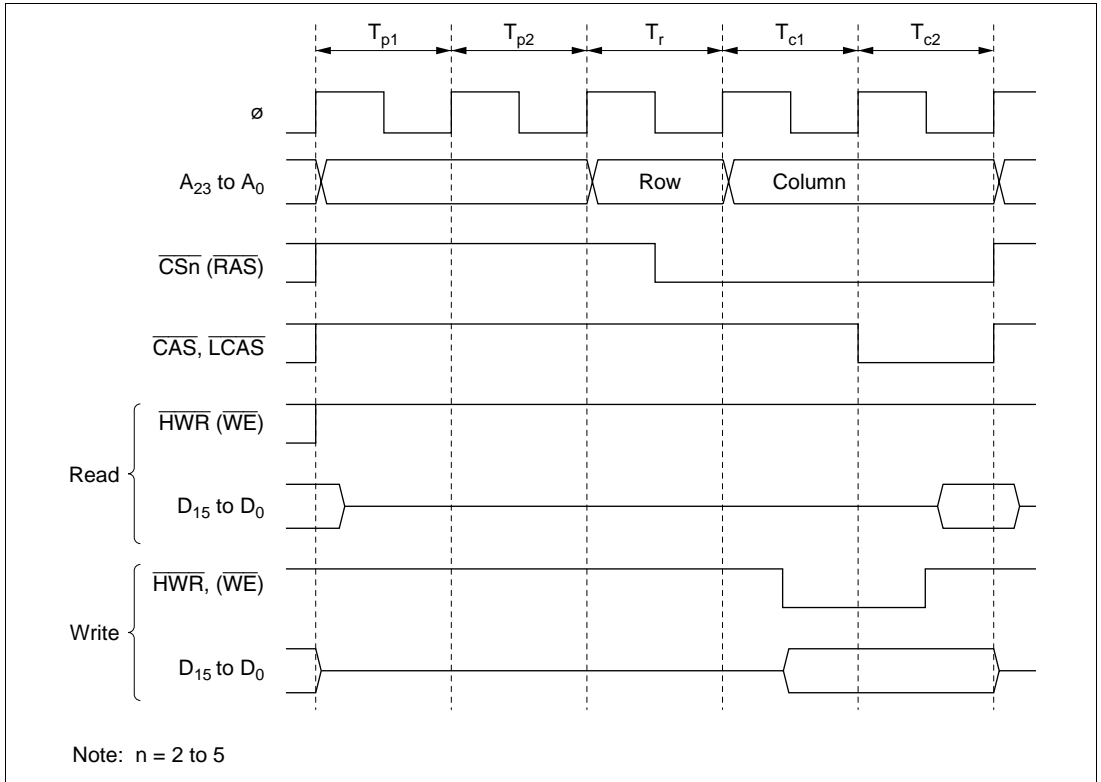


Figure 4-16 Timing with 2-State Precharge Cycle

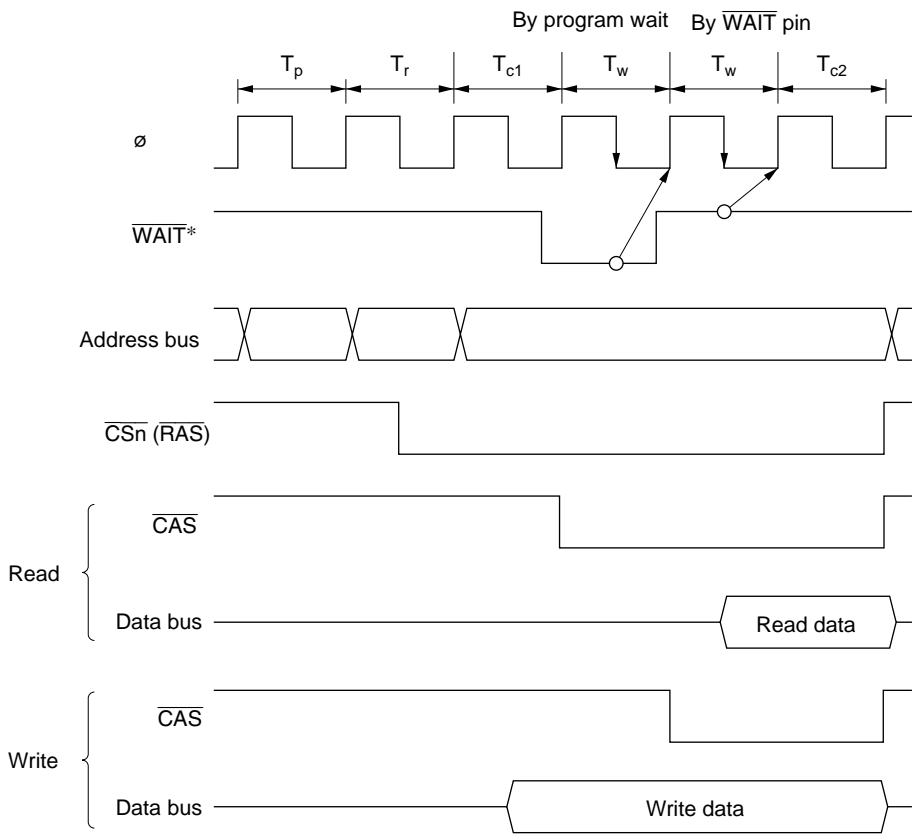
## 4.5.8 Wait Control

There are two ways of inserting wait states in a DRAM access cycle: program wait insertion and pin wait insertion using the  $\overline{\text{WAIT}}$  pin.

**Program Wait Insertion:** When the bit in ASTCR corresponding to an area designated as DRAM space is set to 1, from 0 to 3 wait states can be inserted automatically between the  $T_{c1}$  state and  $T_{c2}$  state, according to the settings of WCRH and WCRL.

**Pin Wait Insertion:** When the WAITE bit in BCRH is set to 1, wait input by means of the  $\overline{\text{WAIT}}$  pin is enabled regardless of the setting of the AST bit in ASTCR. When DRAM space is accessed in this state, a program wait is first inserted. If the  $\overline{\text{WAIT}}$  pin is low at the falling edge of  $\phi$  in the last  $T_{c1}$  or  $T_w$  state, another  $T_w$  state is inserted. If the  $\overline{\text{WAIT}}$  pin is held low,  $T_w$  states are inserted until it goes high. The wait input pin differs from model to model; see the reference manual for the relevant model for details.

Figure 4-17 shows an example of wait state insertion timing.



Notes: ↓ indicates the timing of  $\overline{\text{WAIT}}^*$  pin sampling.  
 n = 2 to 5

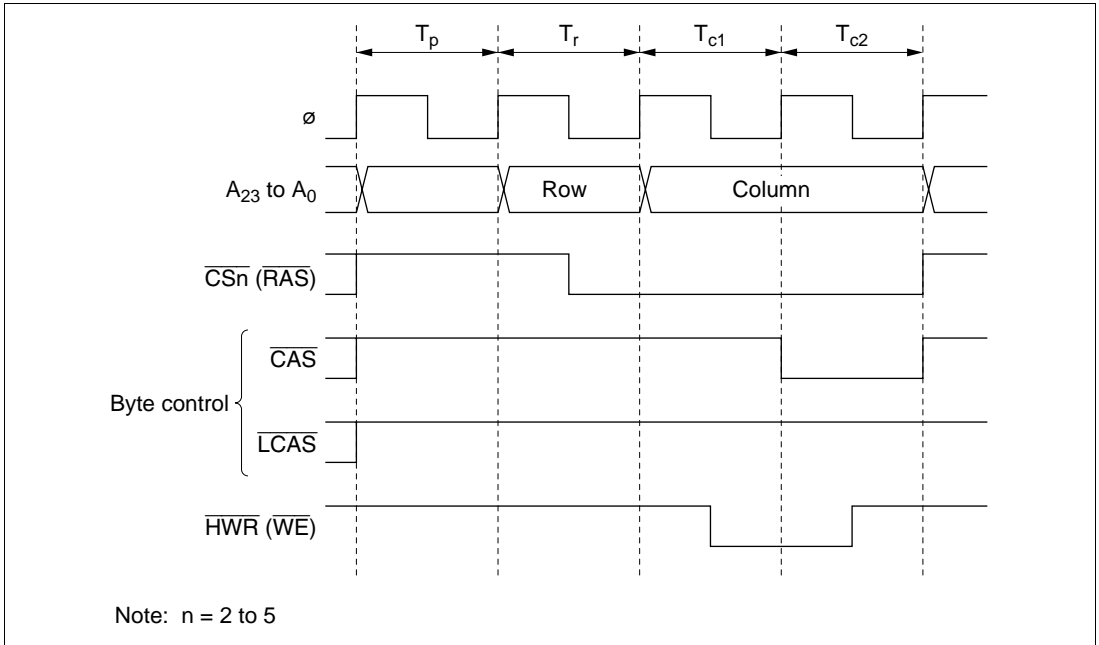
The pin that can be used for wait input, and the setting conditions, differ from model to model; see the reference manual for the relevant model for details.

**Figure 4-17 Example of Wait State Insertion Timing**

## 4.5.9 Byte Access Control

When DRAM with a  $\times 16$  configuration is connected, the 2-CAS system can be used for the control signals required for byte access.

Figure 4-18 shows the control timing in the 2-CAS system, and figure 4-19 shows an example of 2-CAS type DRAM connection.

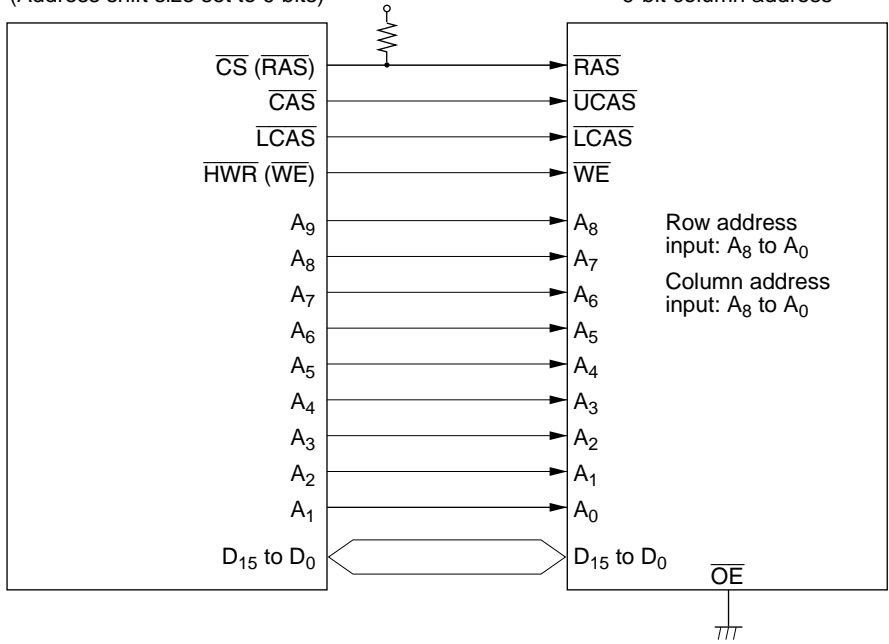


**Figure 4-18 2-CAS System Control Timing (Upper Byte Write Access)**



H8S/2338 Series, H8S/2328 Series,  
or H8S/2318 Series  
(Address shift size set to 9 bits)

2-CAS type 4-Mbit DRAM  
256-kbyte × 16-bit configuration  
9-bit column address

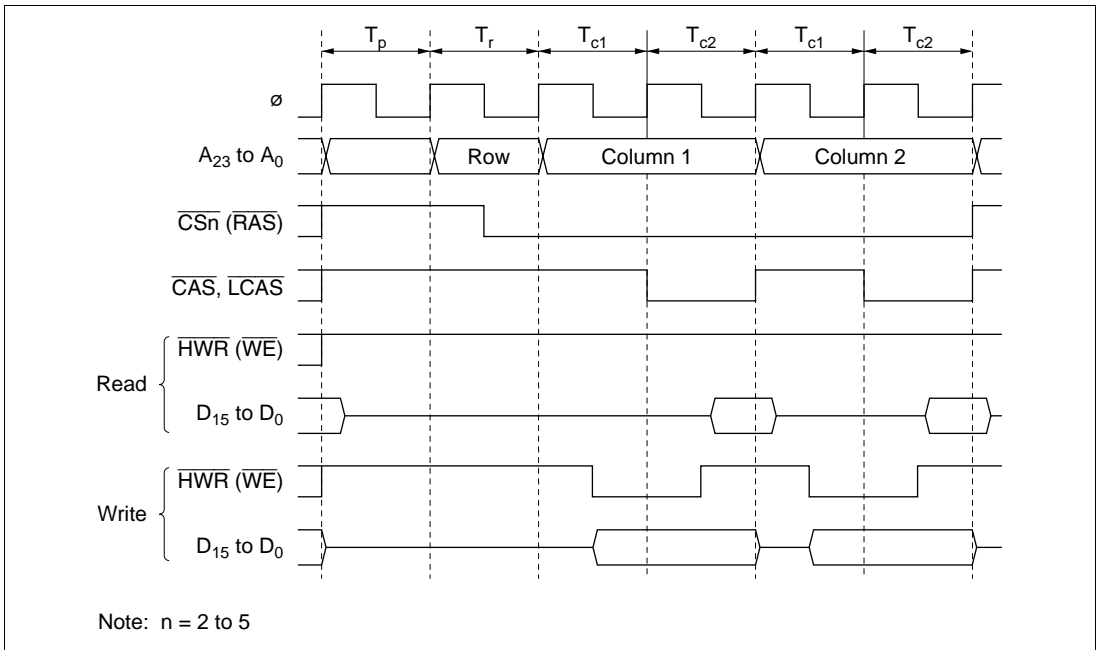


**Figure 4-19 Example of 2-CAS DRAM Connection**

#### 4.5.10 Burst Operation

With DRAM, in addition to full access (normal access) in which data is accessed by outputting a row address for each access, a fast page mode is also provided which can be used when making a number of consecutive accesses to the same row address. This mode enables fast (burst) access of data by simply changing the column address after the row address has been output. Burst access can be selected by setting the BE bit in MCR to 1.

**Burst Access (Fast Page Mode) Operation Timing:** Figure 4-20 shows the operation timing for burst access. When there are consecutive access cycles for DRAM space, the  $\overline{\text{CAS}}$  signal and column address output cycles (two states) continue as long as the row address is the same for consecutive access cycles. The row address used for the comparison is set with bits MXC1 and MXC0 in MCR.



**Figure 4-20 Operation Timing in Fast Page Mode**

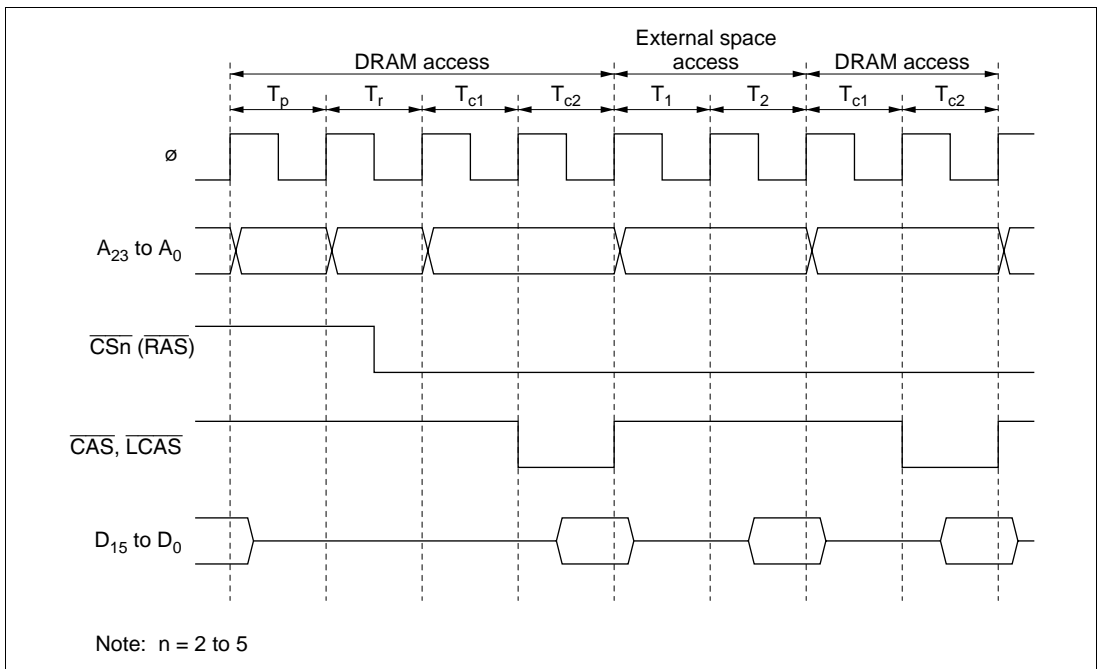
The bus cycle can also be extended in burst access by inserting wait states. The wait state insertion method and timing are the same as for full access. For details, see section 4.5.8, Wait Control.

**RAS Down Mode and RAS Up Mode:** Even when burst operation is selected, it may happen that access to DRAM space is not continuous, but is interrupted by access to another space. In this case, if the  $\overline{\text{RAS}}$  signal is held low during the access to the other space, burst operation can be resumed when the same row address in DRAM space is accessed again.

- RAS down mode

To select RAS down mode, set the RCDM bit in MCR to 1. If access to DRAM space is interrupted and another space is accessed, the  $\overline{\text{RAS}}$  signal is held low during the access to the other space, and burst access is performed if the row address of the next DRAM space access is the same as the row address of the previous DRAM space access. Figure 4-21 shows an example of the timing in RAS down mode.

Note, however, that the  $\overline{\text{RAS}}$  signal will go high if a refresh operation interrupts RAS down mode.

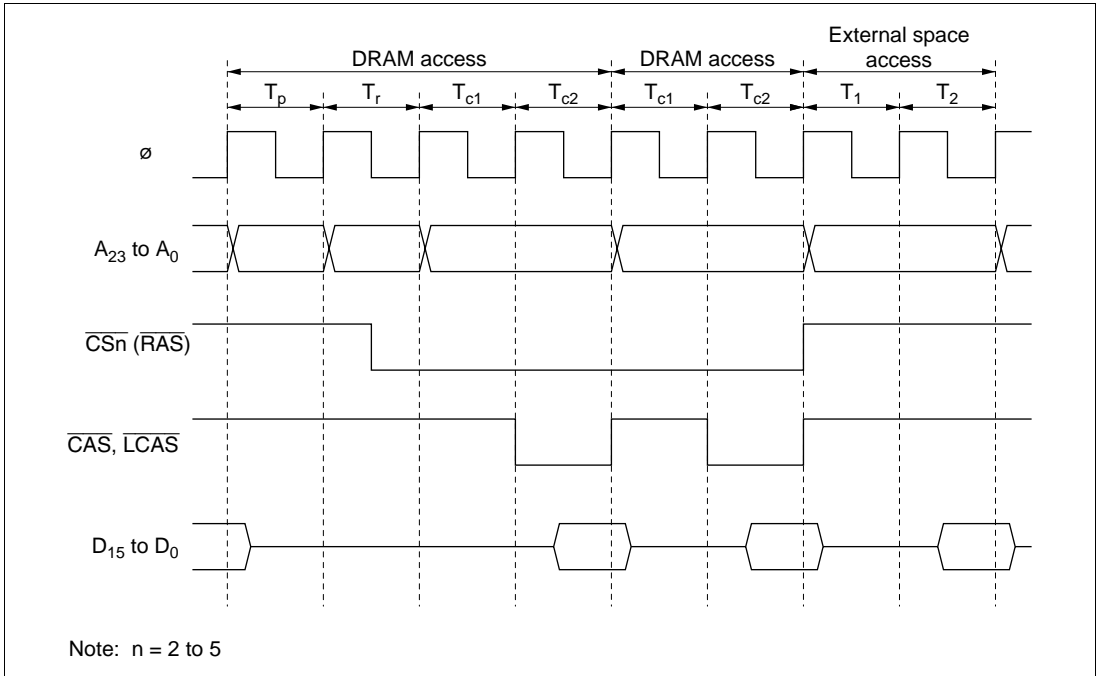


**Figure 4-21 Example of Operation Timing in RAS Down Mode**

- RAS up mode

To select RAS up mode, clear the RCDM bit in MCR to 0. Each time access to DRAM space is interrupted and another space is accessed, the  $\overline{\text{RAS}}$  signal goes high again. Burst operation is only performed if DRAM space is continuous. Figure 4-22 shows an example of the timing in RAS up mode.

In the case of burst ROM space access, the  $\overline{\text{RAS}}$  signal is not restored to the high level.



**Figure 4-22 Example of Operation Timing in RAS Up Mode**

#### 4.5.11 Refresh Control

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are provided with a DRAM refresh control function. Either of two refreshing methods can be selected: CAS-before-RAS (CBR) refreshing, or self-refreshing.

**CAS-before-RAS (CBR) Refreshing:** To select CBR refreshing, set the RFSHE bit in DRAMCR to 1, and clear the RMODE bit to 0.

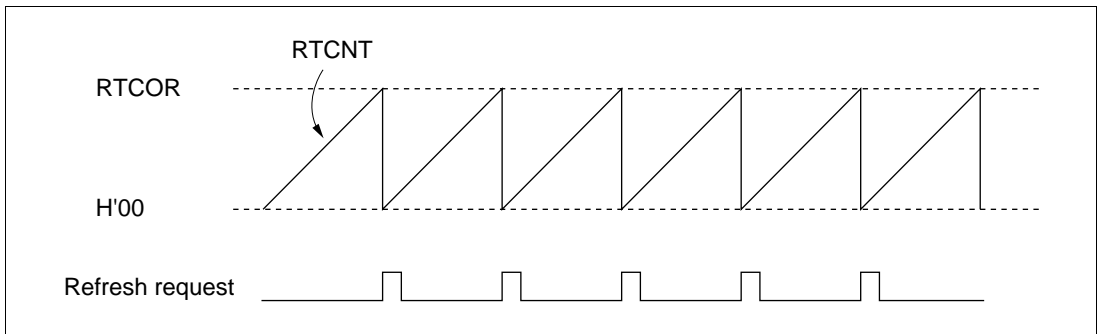
With CBR refreshing, RTCNT counts up using the input clock selected by bits CKS2 to CKS0 in DRAMCR, and when the count matches the value set in RTCOR (compare match), refresh control is performed. At the same time, RTCNT is reset and starts counting again from H'00. Refreshing is thus repeated at fixed intervals determined by RTCOR and bits CKS2 to CKS0. Set a value in RTCOR and bits CKS2 to CKS0 that will meet the refreshing interval specification for the DRAM used.

When bits CKS2 to CKS0 are set, RTCNT starts counting up. RTCNT and RTCOR settings should therefore be completed before setting bits CKS2 to CKS0.

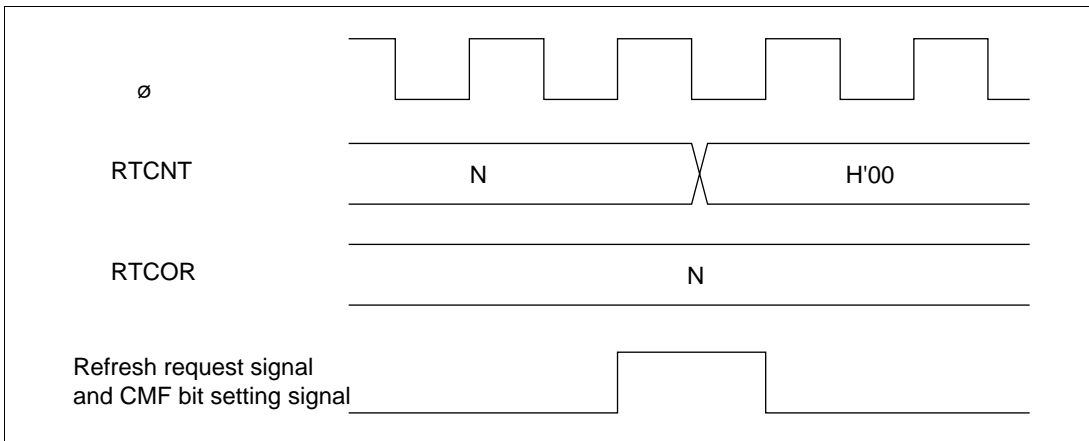
Do not clear the CMF flag when refresh control is being performed (RFSHE = 1).

RTCNT operation is shown in figure 4-23, compare match timing in figure 4-24, and CBR refresh timing in figure 4-25.

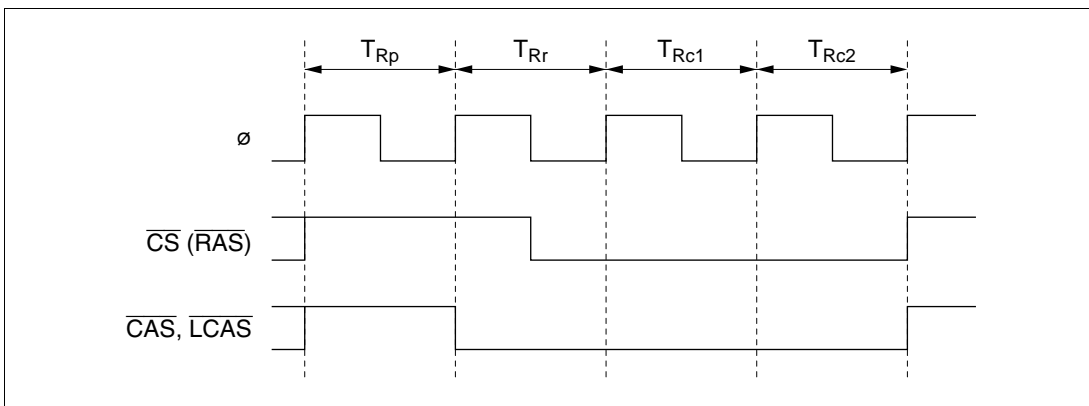
Access to other normal space can be performed during the CBR refresh interval.



**Figure 4-23 RTCNT Operation**



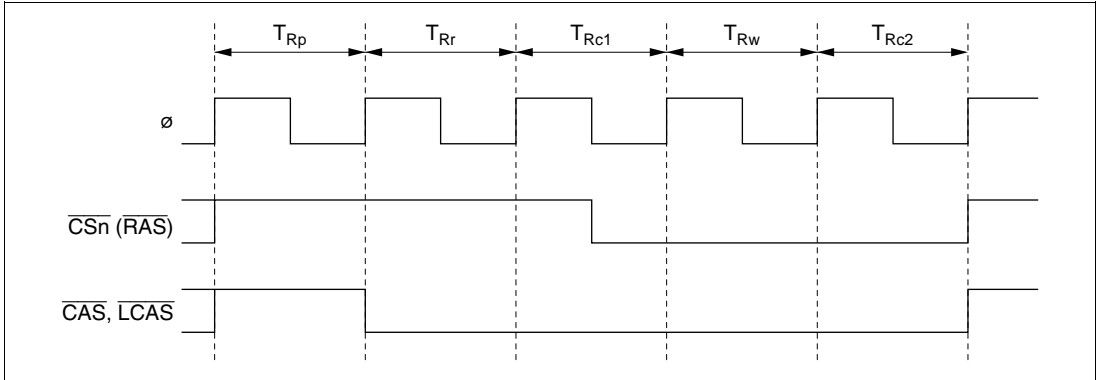
**Figure 4-24 Compare Match Timing**



**Figure 4-25 CBR Refresh Timing**

When the RCW bit is set to 1,  $\overline{\text{RAS}}$  signal output is delayed by one cycle. The width of the  $\overline{\text{RAS}}$  signal should be adjusted with bits RLW1 and RLW0. These bits are only enabled in refresh operations.

Figure 4-26 shows the timing when the RCW bit is set to 1.



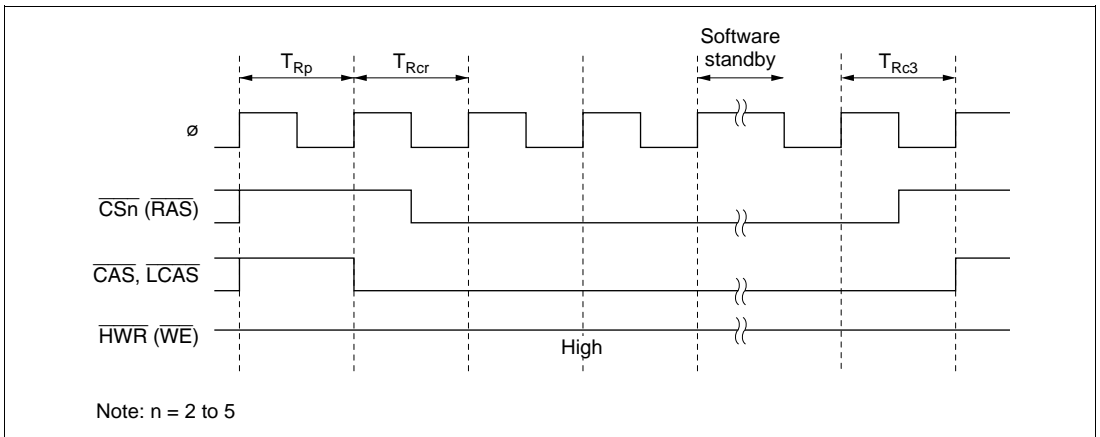
**Figure 4-26 CBR Refresh Timing (When RCW = 1, RLW1 = 0, RLW0 = 1)**

**Self-Refreshing:** A self-refresh mode (battery backup mode) is provided for DRAM as a kind of standby mode. In this mode, refresh timing and refresh addresses are generated within the DRAM.

To select self-refreshing, set the RFSHE bit and RMODE bit in DRAMCR to 1. Then, when a SLEEP instruction is executed to enter software standby mode, the  $\overline{\text{CAS}}$  and  $\overline{\text{RAS}}$  signals are output and DRAM enters self-refresh mode, as shown in figure 4-27.

When software standby mode is exited, the RMODE bit is cleared to 0 and self-refresh mode is cleared.

When switching to software standby mode, if there is a CBR refresh request, CBR refreshing is executed before self-refresh mode is entered.



**Figure 4-27 Self-Refresh Timing**

## 4.6 DMAC Single Address Mode and DRAM Interface

When burst mode is selected with the DRAM interface, the  $\overline{\text{DACK}}$  output timing can be selected with the DDS bit. When DRAM space is accessed in DMAC single address mode at the same time, this bit selects whether or not burst access is to be performed.

### 4.6.1 When DDS = 1

Burst access is performed by determining the address only, irrespective of the bus master. The  $\overline{\text{DACK}}$  output goes low from the  $T_{c1}$  state in the case of the DRAM interface.

Figure 4-28 shows the  $\overline{\text{DACK}}$  output timing for the DRAM interface when DDS = 1.

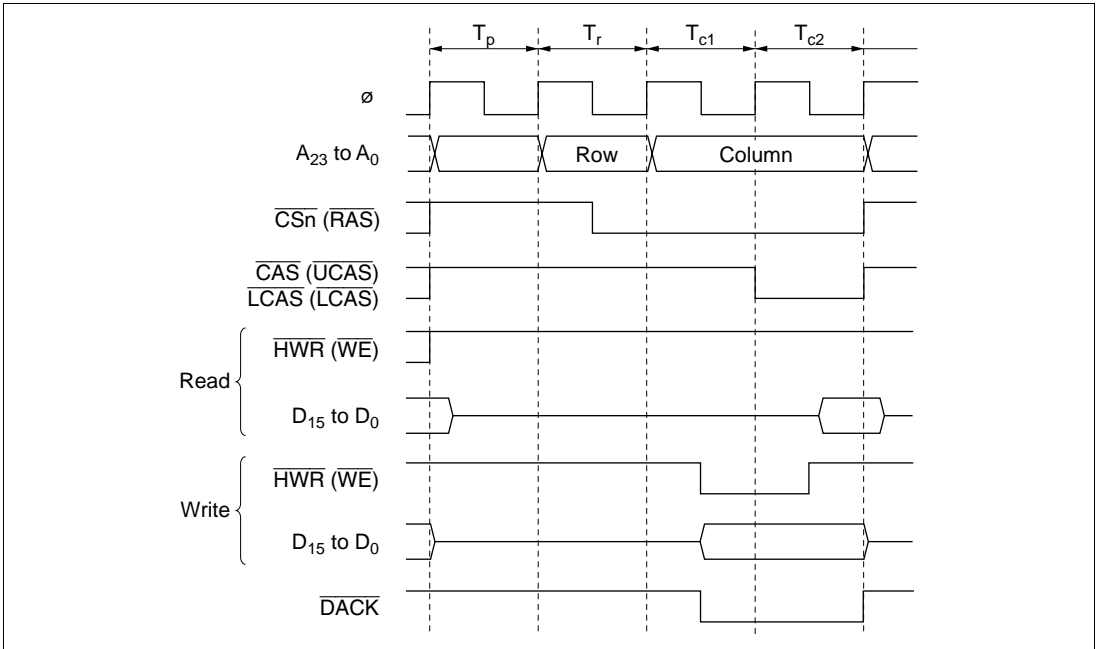


Figure 4-28  $\overline{\text{DACK}}$  Output Timing when DDS = 1 (Example of DRAM Access)



#### 4.6.2 When DDS = 0

When DRAM space is accessed in DMAC single address mode, full access (normal access) is always performed. The  $\overline{\text{DACK}}$  output goes low from the  $T_r$  state in the case of the DRAM interface.

In modes other than DMAC single address mode, burst access can be used when accessing DRAM space.

Figure 4-29 shows the  $\overline{\text{DACK}}$  output timing for the DRAM interface when DDS = 0.

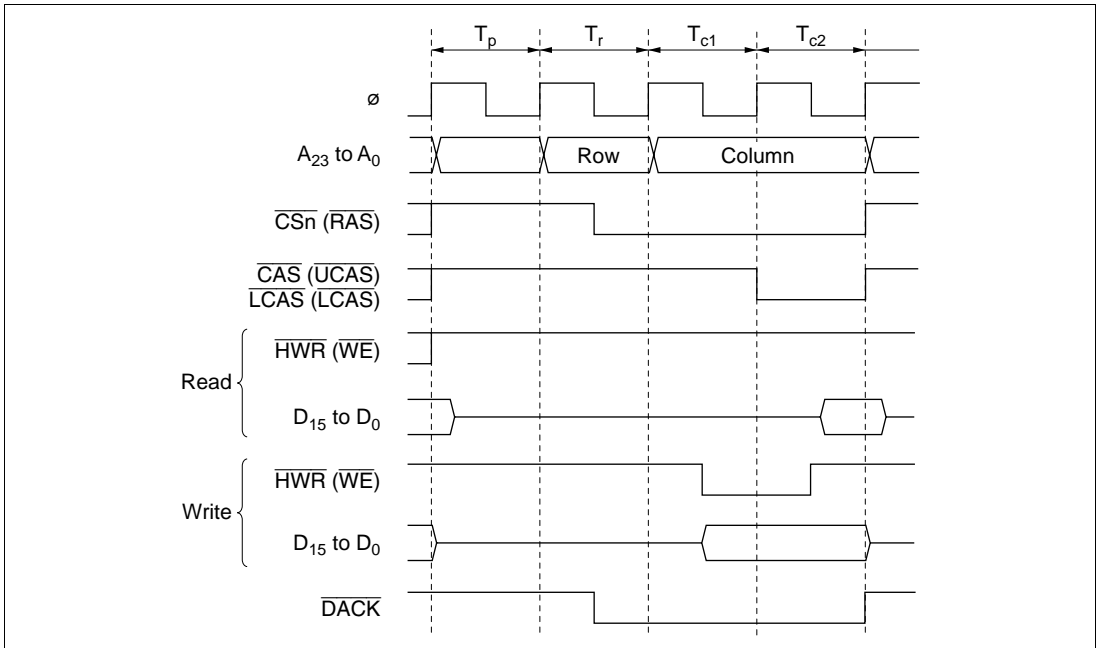


Figure 4-29  $\overline{\text{DACK}}$  Output Timing when DDS = 0 (Example of DRAM Access)

## 4.7 Burst ROM Interface

### 4.7.1 Overview

With the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, external space area 0 can be designated as burst ROM space, and burst ROM interfacing can be performed. The burst ROM space interface enables 16-bit configuration ROM with burst access capability to be accessed at high speed.

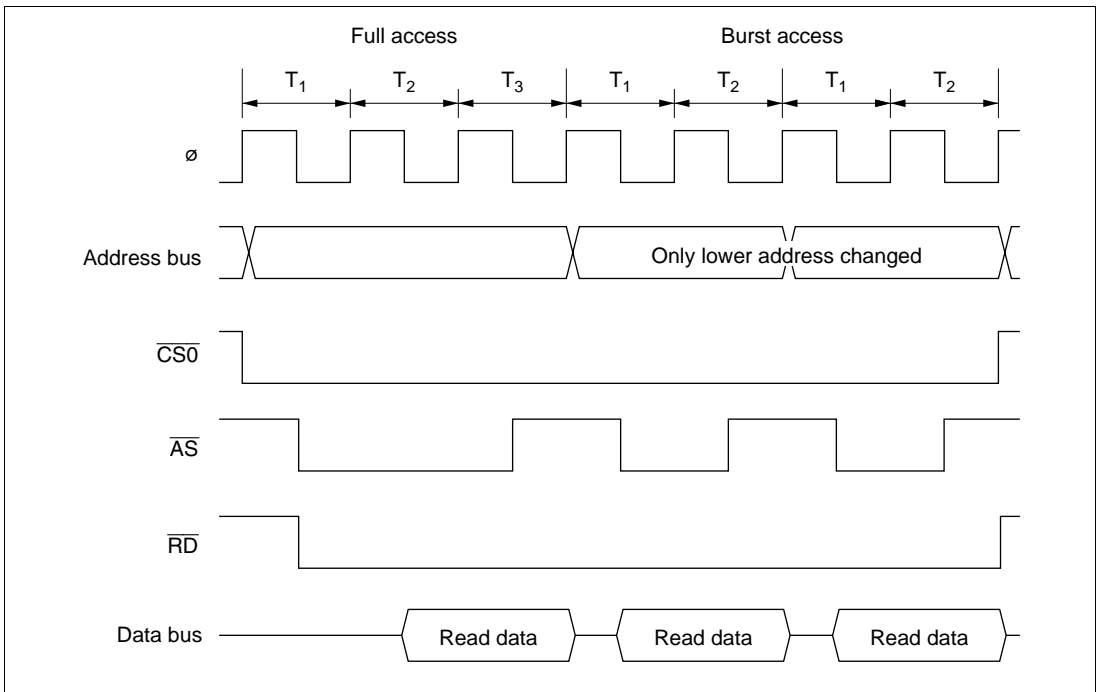
Area 0 can be designated as burst ROM space by means of the BRSTRM bit in BCRH. Consecutive burst accesses of a maximum of 4 words or 8 words can be performed for CPU instruction fetches only. One or two states can be selected for burst access.

### 4.7.2 Basic Timing

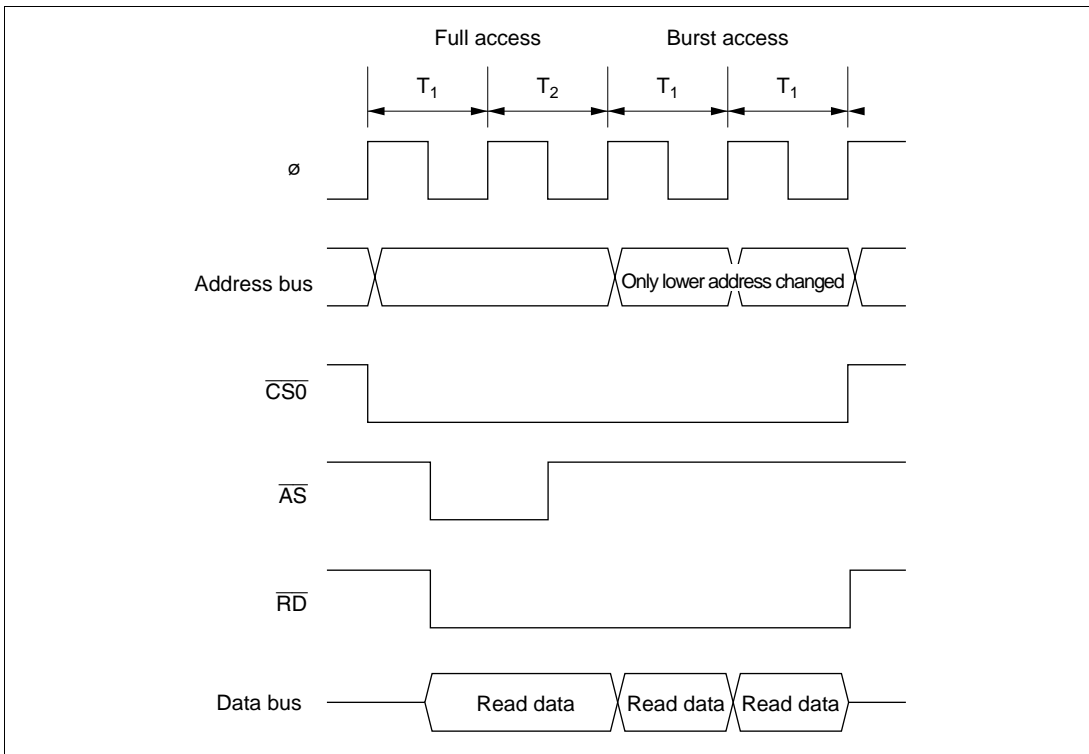
The number of states in the initial cycle (full access) of the burst ROM interface is in accordance with the setting of the AST0 bit in ASTCR. Also, when the AST0 bit is set to 1, wait state insertion is possible. One or two states can be selected for the burst cycle, according to the setting of the BRSTS1 bit in BCRH. Wait states cannot be inserted. When area 0 is designated as burst ROM space, it becomes 16-bit access space regardless of the setting of the ABW0 bit in ABWCR.

When the BRSTS0 bit in BCRH is cleared to 0, burst access of up to 4 words is performed; when the BRSTS0 bit is set to 1, burst access of up to 8 words is performed.

The basic access timing for burst ROM space is shown in figures 4-30 (a) and (b). The timing shown in figure 4-30 (a) is for the case where the AST0 and BRSTS1 bits are both set to 1, and that in figure 4-30 (b) is for the case where both these bits are cleared to 0.



**Figure 4-30 (a) Example of Burst ROM Access Timing (When  $AST0 = BRSTS1 = 1$ )**



**Figure 4-30 (b) Example of Burst ROM Access Timing (When  $AST0 = BRSTS1 = 0$ )**

### 4.7.3 Wait Control

As with the basic bus interface, either program wait insertion or pin wait insertion using the  $\overline{WAIT}$  pin can be used in the initial cycle (full access) of the burst ROM interface. See section 4.4.5, Wait Control.

Wait states cannot be inserted in a burst cycle.

## 4.8 Idle Cycle

### 4.8.1 Operation

When the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip accesses external space, it can insert a 1-state idle cycle ( $T_1$ ) between bus cycles in the following two cases: (1) when read accesses in different areas occur consecutively, and (2) when a write cycle occurs immediately after a read cycle. By inserting an idle cycle it is possible, for example, to avoid data collisions between ROM, with a long output floating time, and high-speed memory, I/O interfaces, and so on.

**Consecutive Reads in Different Areas:** If consecutive reads in different areas occur while the ICIS1 bit in BCRH is set to 1, an idle cycle is inserted at the start of the second read cycle. This is enabled in advanced mode.

Figure 4-31 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a read cycle from SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data collision is prevented.

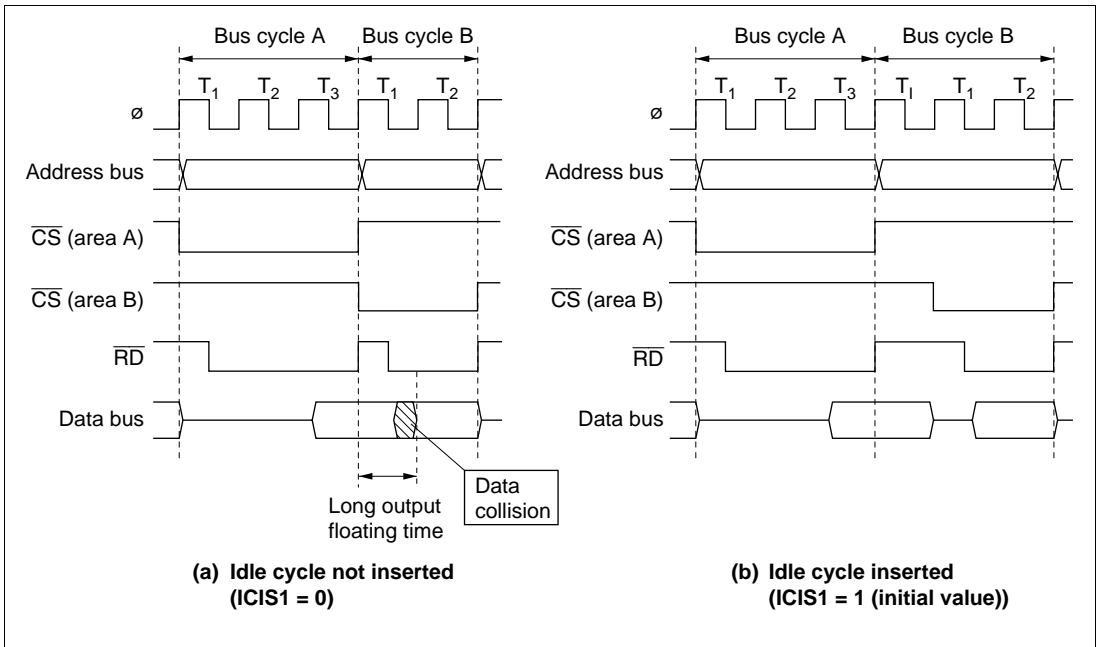
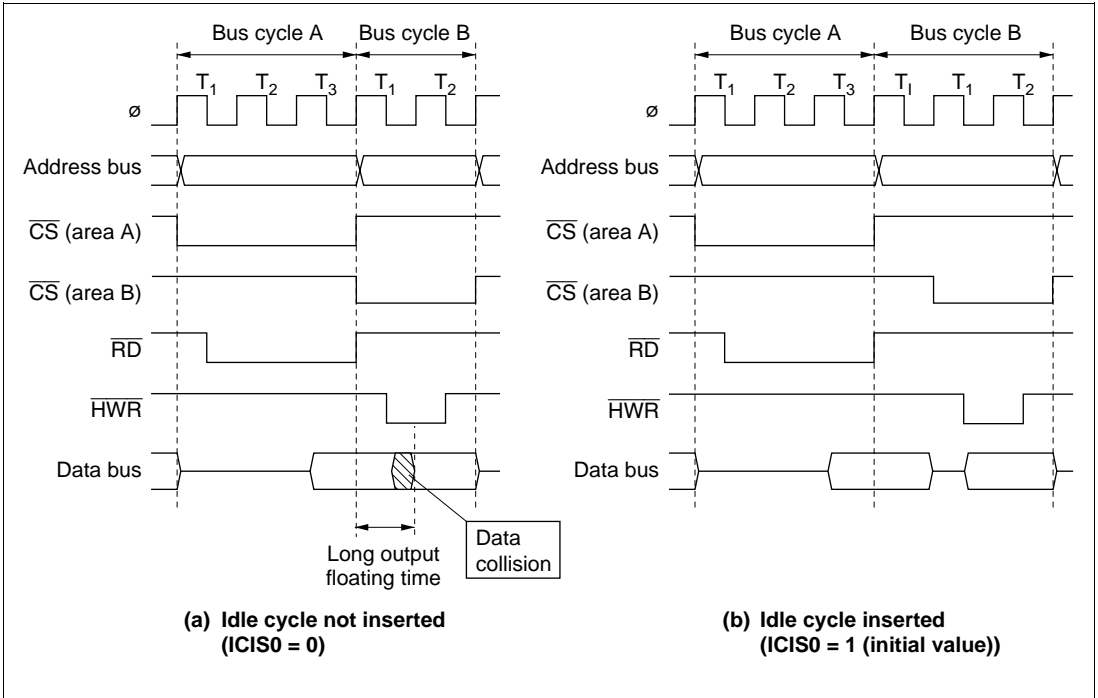


Figure 4-31 Example of Idle Cycle Operation (1)

**Write after Read:** If an external write occurs after an external read while the ICIS0 bit in BCRH is set to 1, an idle cycle is inserted at the start of the write cycle.

Figure 4-32 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data collision is prevented.



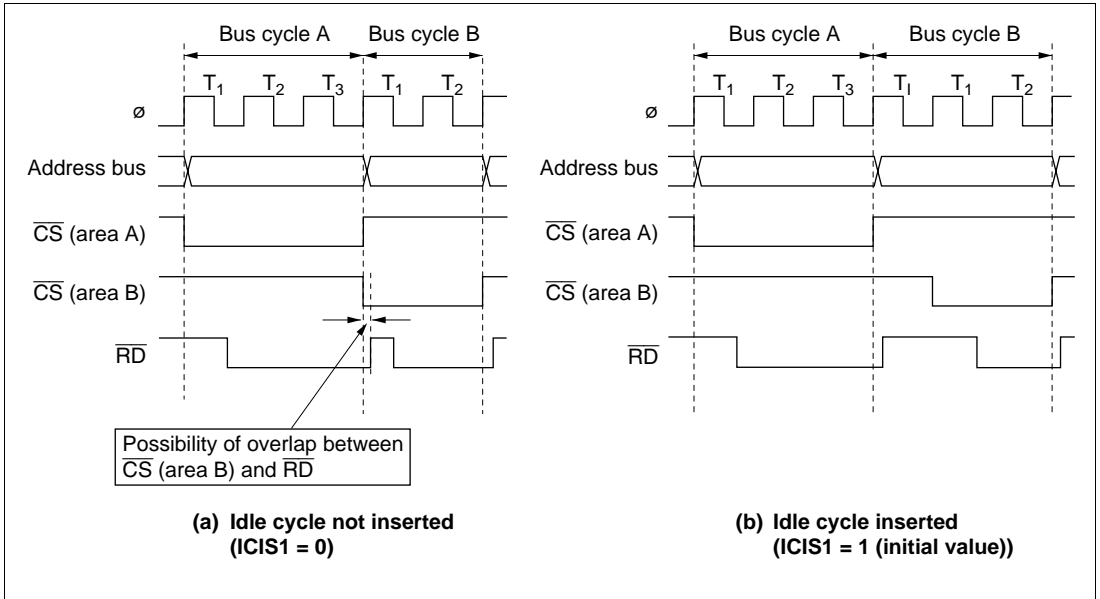
**Figure 4-32 Example of Idle Cycle Operation (2)**

**Relationship between Chip Select ( $\overline{CS}$ ) Signal and Read ( $\overline{RD}$ ) Signal:** Depending on the system's load conditions, the  $\overline{RD}$  signal may lag behind the  $\overline{CS}$  signal. An example is shown in figure 4.33.

In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the bus cycle A  $\overline{RD}$  signal and the bus cycle B  $\overline{CS}$  signal.

Setting idle cycle insertion, as in (b), however, will prevent any overlap between the  $\overline{RD}$  and  $\overline{CS}$  signals.

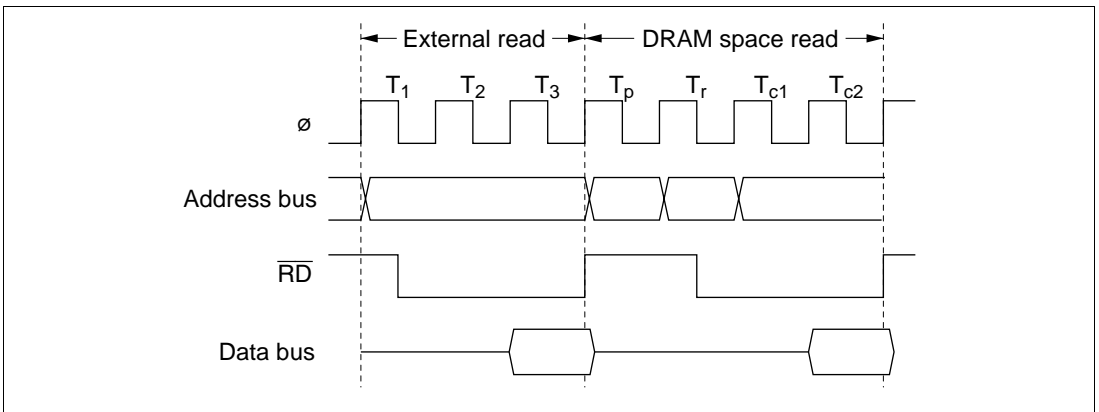
In the initial state after reset release, idle cycle insertion (b) is set.



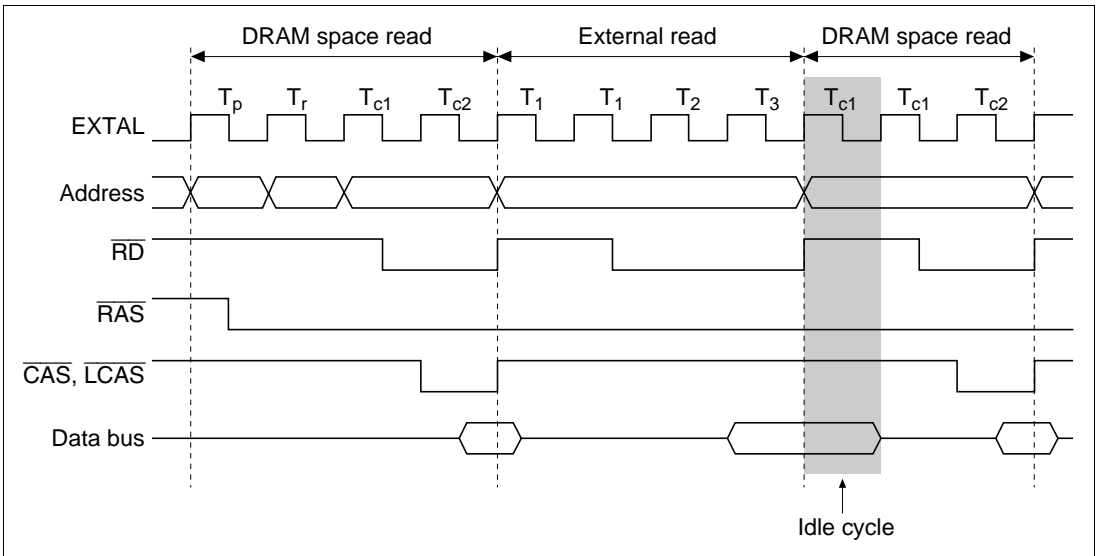
**Figure 4.33 Relationship between Chip Select ( $\overline{CS}$ ) and Read ( $\overline{RD}$ )**

**Usage Notes:** When DRAM space is accessed, the ICIS0 and ICIS1 bit settings are disabled. In the case of consecutive reads between different areas, for example, if the second access is a DRAM access, only a  $T_p$  cycle is inserted, and a  $T_1$  cycle is not. The timing in this case is shown in figure 4-34.

However, in burst access in RAS down mode these settings are enabled, and an idle cycle is inserted. The timing in this case is shown in figures 4-35 (a) and (b).

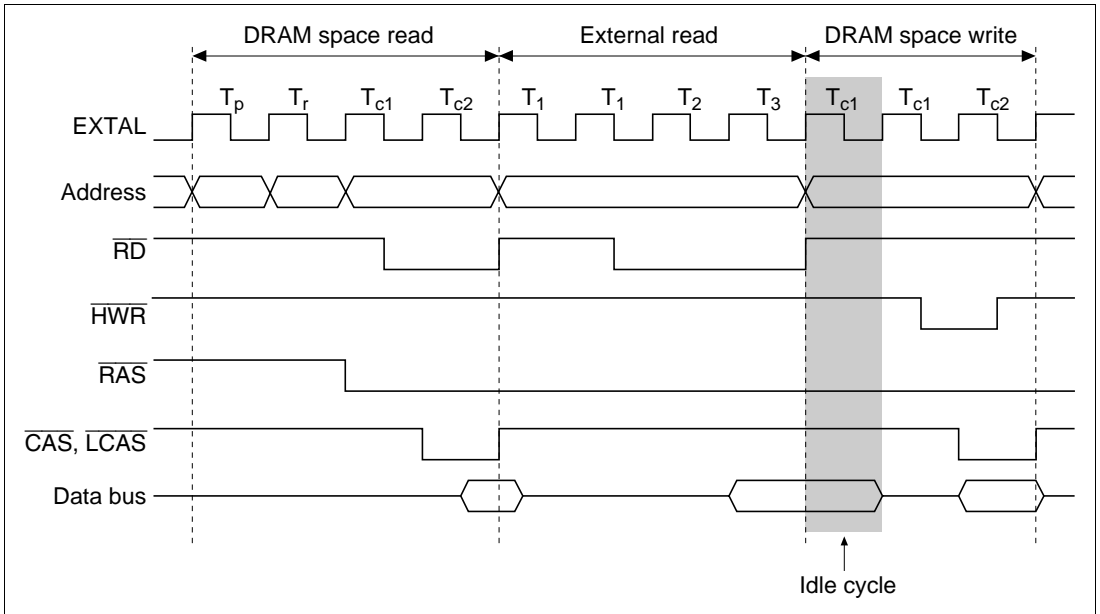


**Figure 4-34 Example of DRAM Access after External Read**



**Figure 4-35 (a) Example of Idle Cycle Operation in RAS Down Mode ( $ICIS1 = 1$ )**





**Figure 4-35 (b) Example of Idle Cycle Operation in RAS Down Mode (ICIS0 = 1)**

#### 4.8.2 Pin States in Idle Cycle

Table 4-8 shows the pin states in an idle cycle.

**Table 4-8 Pin States in Idle Cycle**

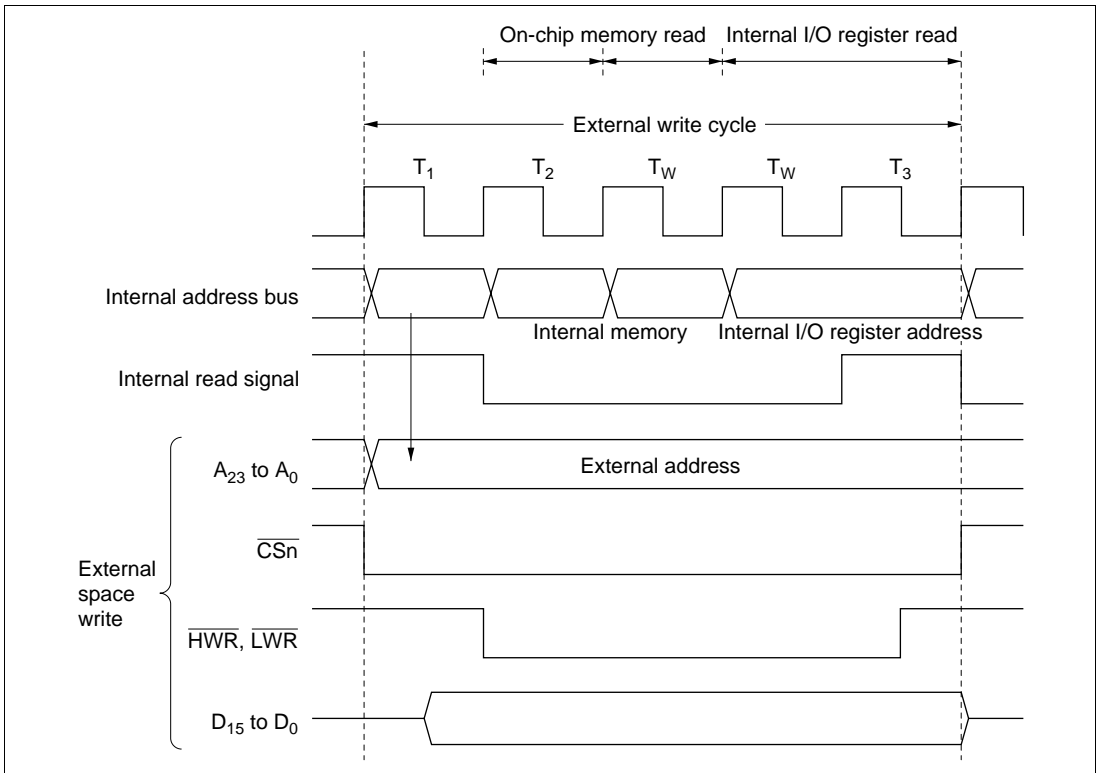
Pins	Pin State
$A_{23}$ to $A_0$	Contents of next bus cycle
$D_{15}$ to $D_0$	High impedance
$\overline{CSn}$	High*
$\overline{CAS}$	High
$\overline{AS}$	High
$\overline{RD}$	High
$\overline{HWR}$	High
$\overline{LWR}$	High
$\overline{DACKn}$	High

Note: \* Remains low in DRAM space RAS down mode or a refresh cycle.

## 4.9 Write Data Buffer Function

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have a write data buffer function in the external data bus. Using the write data buffer function enables external writes and DMA single address mode transfers to be executed in parallel with internal accesses. The write data buffer function is made available by setting the WDBE bit in BCRL to 1. Some models do not support the write data buffer function; see the reference manual for the relevant model for details.

Figure 4-36 shows an example of the timing when the write data buffer function is used. When this function is used, if an external write or DMA single address mode transfer continues for 2 states or longer, and there is an internal access next, only an external write is executed in the first state, but from the next state onward an internal access (on-chip memory or internal I/O register read) is executed in parallel with the external write rather than waiting until it ends.



**Figure 4-36 Example of Timing when Write Data Buffer Function is Used**

## 4.10 Bus Release

### 4.10.1 Overview

The H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip can release the external bus in response to a bus request from an external device. In the external bus released state, the internal bus master continues to operate as long as there is no external access.

If an internal bus master wants to make an external access in the external bus released state, or if a refresh request is generated, it can issue a bus request off-chip. The pin used for this bus request output ( $\overline{\text{BREQO}}$ ) differs from model to model; see the reference manual for the relevant model for details.

### 4.10.2 Operation

In external expansion mode, the bus can be released to an external device by setting the BRLE bit in BCRL to 1. Driving the  $\overline{\text{BREQ}}$  pin low issues an external bus request to the chip. When the  $\overline{\text{BREQ}}$  pin is sampled, at the prescribed timing the  $\overline{\text{BACK}}$  pin is driven low, and the address bus, data bus, and bus control signals are placed in the high-impedance state, establishing the external bus released state.

In the external bus released state, an internal bus master can perform accesses using the internal bus. When an internal bus master wants to make an external access, it temporarily defers activation of the bus cycle, and waits for the bus request from the external bus master to be dropped. Even if a refresh request is generated in the external bus released state, refresh control is deferred until the external bus master drops the bus request.

If the BREQOE bit in BCRL is set to 1, when an internal bus master wants to make an external access in the external bus released state, or when a refresh request is generated, the  $\overline{\text{BREQO}}$  pin is driven low and a request can be made off-chip to drop the bus request.

When the  $\overline{\text{BREQ}}$  pin is driven high, the  $\overline{\text{BACK}}$  pin is driven high at the prescribed timing and the external bus released state is terminated.

If an external bus release request and external access occur simultaneously, the order of priority is as follows:

(High) External bus release > Internal bus master external access (Low)

If a refresh request and external bus release request occur simultaneously, the order of priority is as follows:

(High) Refresh > External bus release (Low)

As a refresh and an external access by an internal bus master can be executed simultaneously, there is no relative order of priority for these two operations.

### 4.10.3 Pin States in External Bus Released State

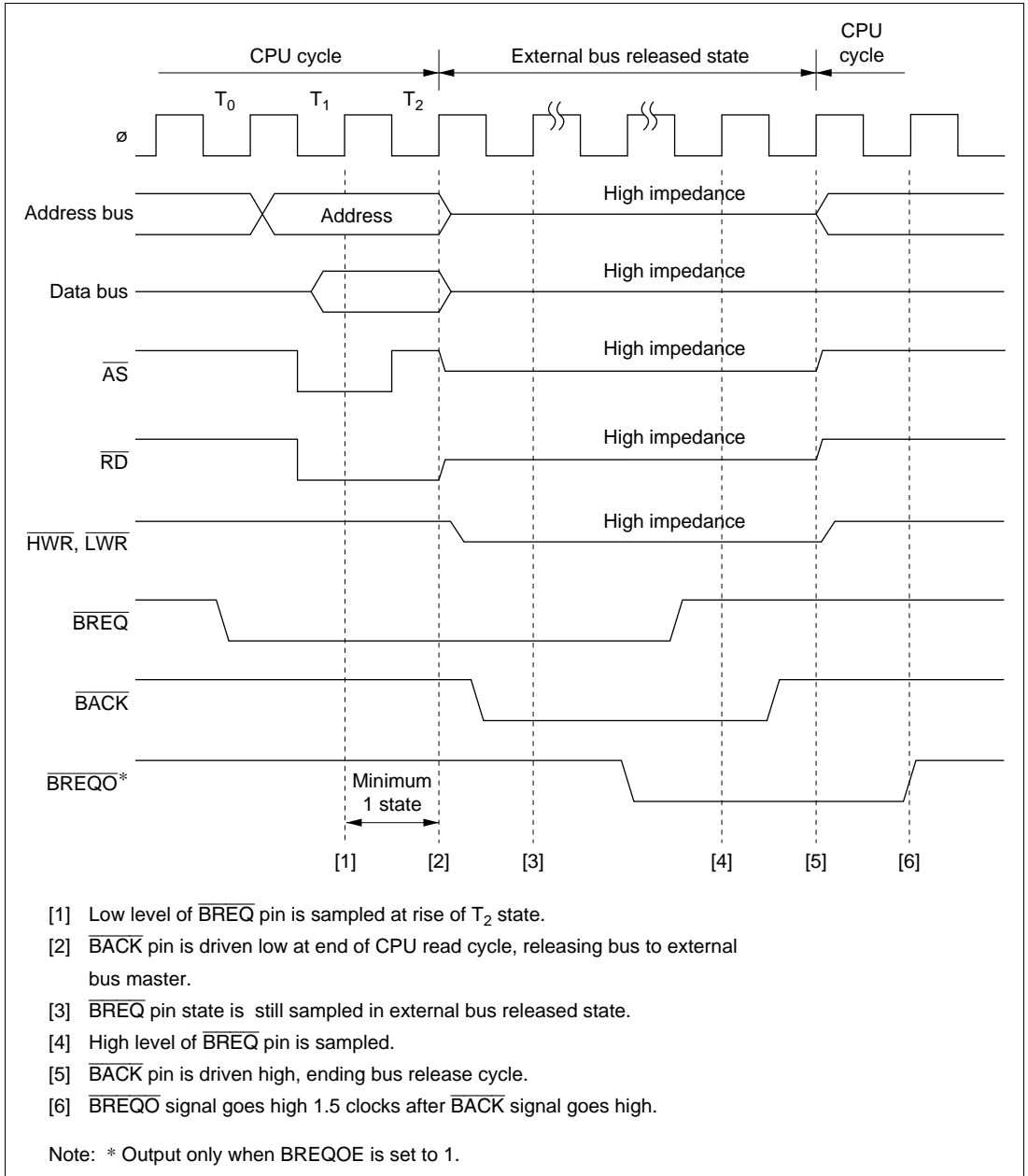
Table 4-9 shows the pin states in the external bus released state.

**Table 4-9 Pin States in Bus Released State**

<b>Pins</b>	<b>Pin State</b>
$A_{23}$ to $A_0$	High impedance
$D_{15}$ to $D_0$	High impedance
$\overline{CSn}$	High impedance
$\overline{CAS}$	High impedance
$\overline{AS}$	High impedance
$\overline{RD}$	High impedance
$\overline{HWR}$	High impedance
$\overline{LWR}$	High impedance
$\overline{DACKn}$	High

## 4.10.4 Transition Timing

Figure 4-37 shows the timing for transition to the bus released state.



**Figure 4-37 Bus Released State Transition Timing**

## 4.10.5 Usage Note

Do not set MSTPCR to H'FFFF or H'EFFF, since the external bus release function will halt if a transition is made to sleep mode when either of these settings has been made.

## 4.11 Bus Arbitration

### 4.11.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have a bus arbiter that arbitrates bus master operations.

There are three bus masters, the CPU, DTC, and DMAC, which perform read/write operations when they have possession of the bus. Each bus master requests the bus by means of a bus request signal. The bus arbiter determines priorities at the prescribed timing, and permits use of the bus by means of a bus request acknowledge signal. The selected bus master then takes possession of the bus and begins its operation.

### 4.11.2 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master making the request. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The order of priority of the bus masters is as follows:

(High) DMAC > DTC > CPU (Low)

An internal bus access by an internal bus master, external bus release, and refreshing, can be executed in parallel.

In the event of simultaneous external bus release request, refresh request, and internal bus master external access request generation, the order of priority is as follows:

(High) Refresh > External bus release (Low)

(High) External bus release > Internal bus master external access (Low)

As a refresh and an external access by an internal bus master can be executed simultaneously, there is no relative order of priority for these two operations.

### 4.11.3 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority than that of the bus master that has acquired the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific times at which each bus master can relinquish the bus.

**CPU:** The CPU is the lowest-priority bus master, and if a bus request is received from the DTC or DMAC, the bus arbiter transfers the bus to the bus master that issued the request. The timing for transfer of the bus is as follows:

- The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the operations. See Appendix A-5, Bus States During Instruction Execution, for timings at which the bus is not transferred.
- If the CPU is in sleep mode, it transfers the bus immediately.

**DTC:** The DTC sends the bus arbiter a request for the bus when an activation request is generated.

The DTC can release the bus after a vector read, a register information read (3 states), a single data transfer, or a register information write (3 states). It does not release the bus during a register information read (3 states), a single data transfer, or a register information write (3 states).

**DMAC:** The DMAC sends the bus arbiter a request for the bus when an activation request is generated.

In the case of an external request in short address mode or normal mode, and in cycle steal mode, the DMAC releases the bus after a single transfer.

In block transfer mode, it releases the bus after transfer of one block, and in burst mode, after completion of a transfer.

### 4.11.4 External Bus Release Usage Note

External bus release can be performed on completion of an external bus cycle. The  $\overline{RD}$  signal and the DRAM interface  $\overline{RAS}$  and  $\overline{CAS}$  signals remain low until the end of the external bus cycle. Therefore, when external bus release is performed, the  $\overline{RD}$ ,  $\overline{RAS}$ , and  $\overline{CAS}$  signals may change from the low level to the high-impedance state.

## 4.12 Resets and the Bus Controller

In a reset, the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip, including the bus controller, enters the reset state at that point, and any executing bus cycle is discontinued.

# Section 5 DMA Controller

## 5.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have a built-in DMA controller (DMAC) which can carry out data transfer on up to 4 channels.

### 5.1.1 Features

The features of the DMAC are listed below.

- Choice of short address mode or full address mode
  - Short address mode
    - Maximum of 4 channels can be used
    - Choice of dual address mode or single address mode
    - In dual address mode, one of the two addresses, transfer source and transfer destination, is specified as 24 bits and the other as 16 bits
    - In single address mode, transfer source or transfer destination address only is specified as 24 bits
    - In single address mode, transfer can be performed in one bus cycle
    - Choice of sequential mode, idle mode, or repeat mode for dual address mode and single address mode
  - Full address mode
    - Maximum of 2 channels can be used
    - Transfer source and transfer destination address specified as 24 bits
    - Choice of normal mode or block transfer mode
- 16-Mbyte address space can be specified directly
- Byte or word can be set as the transfer unit
- Activation sources: internal interrupt, external request, auto-request (depending on transfer mode)
  - Six 16-bit timer-pulse unit (TPU) compare match/input capture interrupts
  - Serial communication interface (SCIO, SCI1) transmission complete interrupt, reception complete interrupt
  - A/D converter conversion end interrupt
  - External request
  - Auto-request



- Module stop mode can be set
  - The initial setting enables DMAC registers to be accessed. DMAC operation is halted by setting module stop mode

### 5.1.2 Block Diagram

A block diagram of the DMAC is shown in figure 5-1.

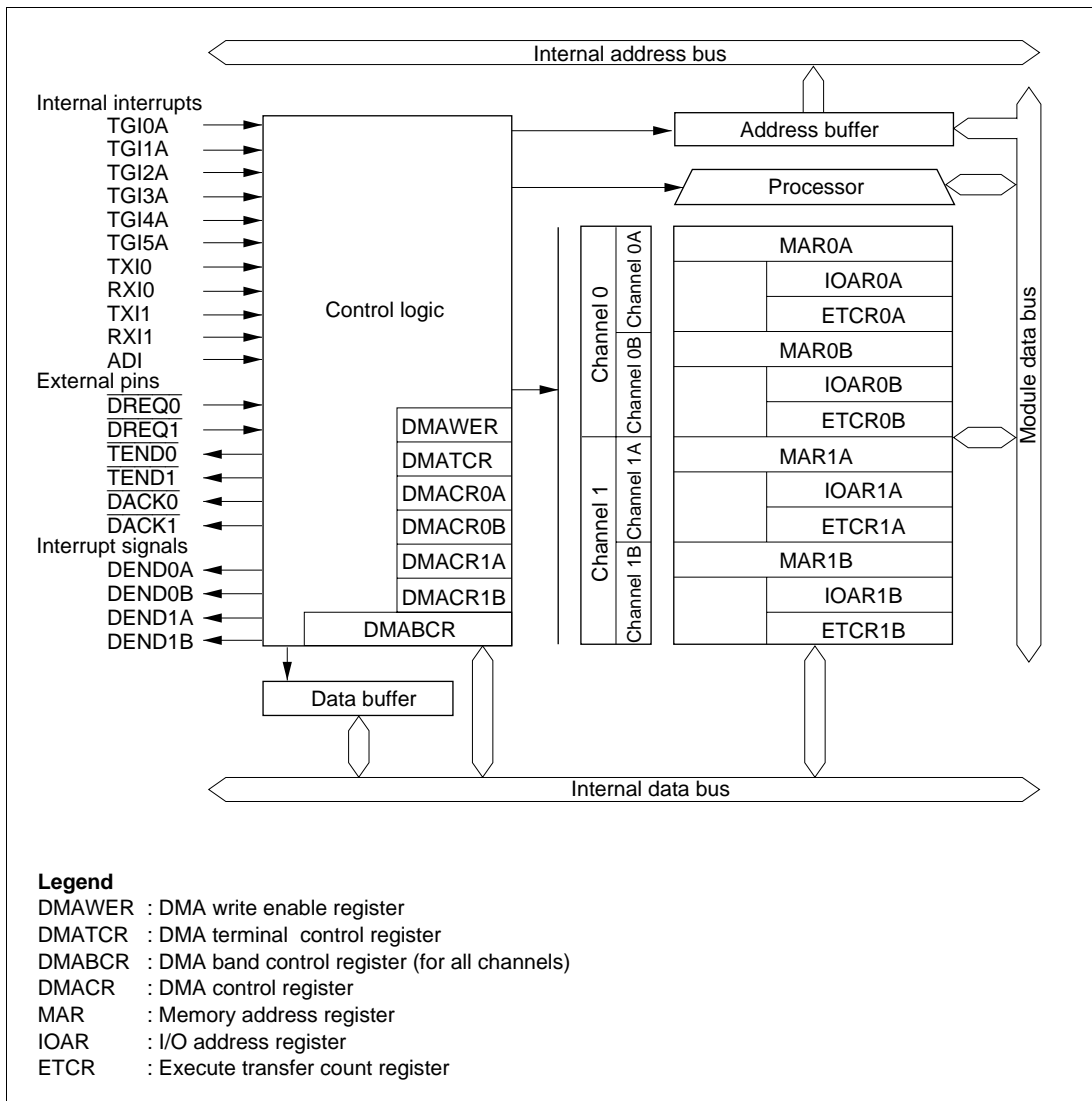


Figure 5-1 Block Diagram of DMAC

### 5.1.3 Overview of Functions

Tables 5-1 (1) and (2) summarize DMAC functions in short address mode and full address mode, respectively.

**Table 5-1 (1) Overview of DMAC Functions (Short Address Mode)**

Transfer Mode	Transfer Source	Address Register Bit Length	
		Source	Destination
Dual address mode	<ul style="list-style-type: none"> <li>TPU channel 0 to 5 compare match/input capture A interrupt</li> </ul>	24/16	16/24
<ul style="list-style-type: none"> <li>Sequential mode               <ul style="list-style-type: none"> <li>1-byte or 1-word transfer executed for one transfer request</li> <li>Memory address incremented/decremented by 1 or 2</li> <li>1 to 65536 transfers</li> </ul> </li> <li>Idle mode               <ul style="list-style-type: none"> <li>1-byte or 1-word transfer executed for one transfer request</li> <li>Memory address fixed</li> <li>1 to 65536 transfers</li> </ul> </li> <li>Repeat mode               <ul style="list-style-type: none"> <li>1-byte or 1-word transfer executed for one transfer request</li> <li>Memory address incremented/decremented by 1 or 2</li> <li>After specified number of transfers (1 to 256), initial state is restored and operation continues</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>SCI transmission complete interrupt</li> <li>SCI reception complete interrupt</li> <li>A/D converter conversion end interrupt</li> <li>External request</li> </ul>		
Single address mode	<ul style="list-style-type: none"> <li>External request</li> </ul>	24/ $\overline{\text{DACK}}$	$\overline{\text{DACK}}$ /24
<ul style="list-style-type: none"> <li>1-byte or 1-word transfer executed for one transfer request</li> <li>Transfer in 1 bus cycle using <math>\overline{\text{DACK}}</math> pin in place of address specifying I/O</li> <li>Specifiable for sequential, idle, and repeat modes</li> </ul>			

**Table 5-1 (2) Overview of DMAC Functions (Full Address Mode)**

Transfer Mode	Transfer Source	Address Register Bit Length	
		Source	Destination
<ul style="list-style-type: none"> <li>• Normal mode</li> <li>Auto-request               <ul style="list-style-type: none"> <li>— Transfer request retained internally</li> <li>— Transfers continue for the specified number of times (1 to 65536)</li> <li>— Choice of burst or cycle steal transfer</li> </ul> </li> <li>External request               <ul style="list-style-type: none"> <li>— 1-byte or 1-word transfer executed for one transfer request</li> <li>— 1 to 65536 transfers</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Auto-request</li> </ul>	24	24
<ul style="list-style-type: none"> <li>• Block transfer mode               <ul style="list-style-type: none"> <li>— Specified block size transfer executed for one transfer request</li> <li>— 1 to 65536 transfers</li> <li>— Either source or destination specifiable as block area</li> <li>— Block size: 1 to 256 bytes or words</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• External request</li> </ul>	24	24
	<ul style="list-style-type: none"> <li>• TPU channel 0 to 5 compare match/input capture A interrupt</li> <li>• SCI transmission complete interrupt</li> <li>• SCI reception complete interrupt</li> <li>• External request</li> <li>• A/D converter conversion end interrupt</li> </ul>		

## 5.1.4 Pin Configuration

Table 5-2 summarizes the DMAC pins. The pins used for output of the various signals differ from model to model; see the reference manual for the relevant model for details.

In short address mode, external request transfer, single address transfer, and transfer end output are not performed for channel A.

The DMA transfer acknowledge function is used in channel B single address mode in short address mode.

When the  $\overline{\text{DREQ}}$  pin is used, do not designate the corresponding port for output.

With regard to the  $\overline{\text{DACK}}$  pins, setting single address transfer automatically sets the corresponding port to output, functioning as a  $\overline{\text{DACK}}$  pin.

With regard to the  $\overline{\text{TEND}}$  pins, whether or not the corresponding port is used as a  $\overline{\text{TEND}}$  pin can be specified by means of a register setting.

**Table 5-2 DMAC Pins**

Channel	Pin Name	Symbol	I/O	Function
0	DMA request 0	$\overline{\text{DREQ0}}$	Input	DMAC channel 0 external request
	DMA transfer acknowledge 0	$\overline{\text{DACK0}}$	Output	DMAC channel 0 single address transfer acknowledge
	DMA transfer end 0	$\overline{\text{TEND0}}$	Output	DMAC channel 0 transfer end
1	DMA request 1	$\overline{\text{DREQ1}}$	Input	DMAC channel 1 external request
	DMA transfer acknowledge 1	$\overline{\text{DACK1}}$	Output	DMAC channel 1 single address transfer acknowledge
	DMA transfer end 1	$\overline{\text{TEND1}}$	Output	DMAC channel 1 transfer end

### 5.1.5 Register Configuration

Table 5-3 summarizes the DMAC registers.

**Table 5-3 DMAC Registers**

Channel	Name	Abbreviation	R/W	Initial Value	Address*	Bus Width
0	Memory address register 0A	MAR0A	R/W	Undefined	H'FEE0	16 bits
	I/O address register 0A	IOAR0A	R/W	Undefined	H'FEE4	16 bits
	Transfer count register 0A	ETCR0A	R/W	Undefined	H'FEE6	16 bits
	Memory address register 0B	MAR0B	R/W	Undefined	H'FEE8	16 bits
	I/O address register 0B	IOAR0B	R/W	Undefined	H'FEEC	16 bits
	Transfer count register 0B	ETCR0B	R/W	Undefined	H'FEEE	16 bits
1	Memory address register 1A	MAR1A	R/W	Undefined	H'FEF0	16 bits
	I/O address register 1A	IOAR1A	R/W	Undefined	H'FEF4	16 bits
	Transfer count register 1A	ETCR1A	R/W	Undefined	H'FEF6	16 bits
	Memory address register 1B	MAR1B	R/W	Undefined	H'FEF8	16 bits
	I/O address register 1B	IOAR1B	R/W	Undefined	H'FEFC	16 bits
	Transfer count register 1B	ETCR1B	R/W	Undefined	H'FEFE	16 bits
0, 1	DMA write enable register	DMAWER	R/W	H'00	H'FF00	8 bits
	DMA terminal control register	DMATCR	R/W	H'00	H'FF01	8 bits
	DMA control register 0A	DMACR0A	R/W	H'00	H'FF02	16 bits
	DMA control register 0B	DMACR0B	R/W	H'00	H'FF03	16 bits
	DMA control register 1A	DMACR1A	R/W	H'00	H'FF04	16 bits
	DMA control register 1B	DMACR1B	R/W	H'00	H'FF05	16 bits
	DMA band control register	DMABCR	R/W	H'0000	H'FF06	16 bits
	Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C	8 bits

Note: \* Lower 16 bits of the address.

## 5.2 Register Descriptions (1) (Short Address Mode)

Short address mode transfer can be performed for channels A and B independently.

Short address mode transfer is specified for each channel by clearing the FAE bit in DMABCR to 0, as shown in table 5-4. Short address mode or full address mode can be selected for channels 1 and 0 independently by means of bits FAE1 and FAE0.

**Table 5-4 Short Address Mode and Full Address Mode (For 1 Channel: Example of Channel 0)**

FAE0	Description																
0	Short address mode specified (channels A and B operate independently)																
Channel 0A	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td colspan="2" style="text-align: center;">MAR0A</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">IOAR0A</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">ETCR0A</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">DMACR0A</td></tr> </table> <div style="display: inline-block; vertical-align: middle; margin-left: 10px;"> <p>← Specifies transfer source/transfer destination address</p> <p>← Specifies transfer destination/transfer source address</p> <p>← Specifies number of transfers</p> <p>← Specifies transfer size, mode, activation source, etc.</p> </div>	MAR0A			IOAR0A		ETCR0A		DMACR0A								
MAR0A																	
	IOAR0A																
	ETCR0A																
	DMACR0A																
Channel 0B	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td colspan="2" style="text-align: center;">MAR0B</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">IOAR0B</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">ETCR0B</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">DMACR0B</td></tr> </table> <div style="display: inline-block; vertical-align: middle; margin-left: 10px;"> <p>← Specifies transfer source/transfer destination address</p> <p>← Specifies transfer destination/transfer source address</p> <p>← Specifies number of transfers</p> <p>← Specifies transfer size, mode, activation source, etc.</p> </div>	MAR0B			IOAR0B		ETCR0B		DMACR0B								
MAR0B																	
	IOAR0B																
	ETCR0B																
	DMACR0B																
1	Full address mode specified (channels A and B operate in combination)																
Channel 0	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td colspan="2" style="text-align: center;">MAR0A</td></tr> <tr><td colspan="2" style="text-align: center;">MAR0B</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">IOAR0A</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">IOAR0B</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">ETCR0A</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">ETCR0B</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">DMACR0A</td></tr> <tr><td style="width: 50%;"></td><td style="text-align: center;">DMACR0B</td></tr> </table> <div style="display: inline-block; vertical-align: middle; margin-left: 10px;"> <p>← Specifies transfer source address</p> <p>← Specifies transfer destination address</p> <p>← Not used</p> <p>← Not used</p> <p>← Specifies number of transfers</p> <p>← Specifies number of transfers (used in block transfer mode only)</p> <p>← Specifies transfer size, mode, activation source, etc.</p> </div>	MAR0A		MAR0B			IOAR0A		IOAR0B		ETCR0A		ETCR0B		DMACR0A		DMACR0B
MAR0A																	
MAR0B																	
	IOAR0A																
	IOAR0B																
	ETCR0A																
	ETCR0B																
	DMACR0A																
	DMACR0B																

## 5.2.1 Memory Address Registers (MAR)

Bit	:	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16																		
MAR	:	<table border="1" style="width:100%; height:20px; border-collapse: collapse;"> <tr> <td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;">—</td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td> </tr> </table>																—	—	—	—	—	—	—	—										
—	—	—	—	—	—	—	—																												
Initial value	:	0	0	0	0	0	0	0	0	*	*	*	*	*	*	*	*																		
R/W	:	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																		

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																		
MAR	:	<table border="1" style="width:100%; height:20px; border-collapse: collapse;"> <tr> <td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td><td style="width:5%;"> </td> </tr> </table>																																	
Initial value	:	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*																		
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																		

\*: Undefined

MAR is a 32-bit readable/writable register that specifies the transfer source address or destination address.

The upper 8 bits of MAR are reserved: they are always read as 0, and cannot be modified.

Whether MAR functions as the source address register or as the destination address register can be selected by means of the DTDIR bit in DMACR.

MAR is incremented or decremented each time a byte or word transfer is executed, so that the address specified by MAR is constantly updated. For details, see section 5.2.4, DMA Control Register (DMACR).

MAR is not initialized by a reset or in standby mode.

### 5.2.2 I/O Address Register (IOAR)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IOAR	:																
Initial value	:	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

IOAR is a 16-bit readable/writable register that specifies the lower 16 bits of the transfer source address or destination address. The upper 8 bits of the transfer address are automatically set to H'FF.

Whether IOAR functions as the source address register or as the destination address register can be selected by means of the DTDIR bit in DMACR.

IOAR is invalid in single address mode.

IOAR is not incremented or decremented each time a transfer is executed, so the address specified by IOAR is fixed.

IOAR is not initialized by a reset or in standby mode.

### 5.2.3 Execute Transfer Count Register (ETCR)

ETCR is a 16-bit readable/writable register that specifies the number of transfers. The setting of this register is different for sequential mode and idle mode on the one hand, and for repeat mode on the other.

#### Sequential Mode and Idle Mode

##### Transfer Counter (ETCR)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	:																
Initial value	:	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

In sequential mode and idle mode, ETCR functions as a 16-bit transfer counter (with a count range of 1 to 65536). ETCR is decremented by 1 each time a transfer is performed, and when the count reaches H'0000, the DTE bit in DMABCR is cleared, and transfer ends.



## Repeat Mode

### Transfer Number Storage (ETCRH)

Bit	:	15	14	13	12	11	10	9	8
Initial value :		*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

### Transfer Counter (ETCRL)

Bit	:	7	6	5	4	3	2	1	0
Initial value :		*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

In repeat mode, ETCR functions as transfer counter ETCRL (with a count range of 1 to 256) and transfer number storage register ETCRH. ETCRL is decremented by 1 each time a transfer is performed, and when the count reaches H'00, ETCRL is loaded with the value in ETCRH. At this point, MAR is automatically restored to the value it had when the count was started. The DTE bit in DMABCR is not cleared, and so transfers can be performed repeatedly until the DTE bit is cleared by the user.

ETCR is not initialized by a reset or in standby mode.

### 5.2.4 DMA Control Register (DMACR)

Bit	:	7	6	5	4	3	2	1	0
		DTSZ	DTID5	RPE	DTDIR	DTF3	DTF2	DTF1	DTF0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DMACR is an 8-bit readable/writable register that controls the operation of each DMAC channel.

DMACR is initialized to H'00 by a reset, and in standby mode.

**Bit 7—Data Transfer Size (DTSZ):** Selects the size of data to be transferred at one time.

Bit 7 DTSZ	Description	
0	Byte-size transfer	(Initial value)
1	Word-size transfer	

**Bit 6—Data Transfer Increment/Decrement (DTID):** Selects incrementing or decrementing of MAR after every data transfer in sequential mode or repeat mode.

In idle mode, MAR is neither incremented nor decremented.

Bit 6 DTID	Description	
0	MAR is incremented after a data transfer <ul style="list-style-type: none"> <li>• When DTSZ = 0, MAR is incremented by 1 after a transfer</li> <li>• When DTSZ = 1, MAR is incremented by 2 after a transfer</li> </ul>	(Initial value)
1	MAR is decremented after a data transfer <ul style="list-style-type: none"> <li>• When DTSZ = 0, MAR is decremented by 1 after a transfer</li> <li>• When DTSZ = 1, MAR is decremented by 2 after a transfer</li> </ul>	

**Bit 5—Repeat Enable (RPE):** Used in combination with the DTIE bit in DMABCR to select the mode (sequential, idle, or repeat) in which transfer is to be performed.

Bit 5 RPE	DMABCR DTIE	Description	
0	0	Transfer in sequential mode (no transfer end interrupt)	(Initial value)
	1	Transfer in sequential mode (with transfer end interrupt)	
1	0	Transfer in repeat mode (no transfer end interrupt)	
	1	Transfer in idle mode (with transfer end interrupt)	

For details of operation in sequential, idle, and repeat mode, see section 5.5.2, Sequential Mode, section 5.5.3, Idle Mode, and section 5.5.4, Repeat Mode.

**Bit 4—Data Transfer Direction (DTDIR):** Used in combination with the SAE bit in DMABCR to specify the data transfer direction (source or destination). The function of this bit is therefore different in dual address mode and single address mode.

DMABCR SAE	Bit 4 DTPDIR	Description
0	0	Transfer with MAR as source address and IOAR as destination address (Initial value)
	1	Transfer with IOAR as source address and MAR as destination address
1	0	Transfer with MAR as source address and $\overline{DACK}$ pin as write strobe
	1	Transfer with $\overline{DACK}$ pin as read strobe and MAR as destination address

**Bits 3 to 0—Data Transfer Factor (DTF3 to DTF0):** These bits select the data transfer factor (activation source). There are some differences in activation sources for channel A and for channel B.

### Channel A

Bit 3 DTF3	Bit 2 DTF2	Bit 1 DTF1	Bit 0 DTF0	Description		
0	0	0	0	— (Initial value)		
			1	Activated by A/D converter conversion end interrupt		
		1	0	—		
			1	—		
	1	0	0	0	Activated by SCI channel 0 transmission complete interrupt	
				1	Activated by SCI channel 0 reception complete interrupt	
		1	0	0	Activated by SCI channel 1 transmission complete interrupt	
				1	Activated by SCI channel 1 reception complete interrupt	
1	0	0	0	0	Activated by TPU channel 0 compare match/input capture A interrupt	
				1	Activated by TPU channel 1 compare match/input capture A interrupt	
			1	0	Activated by TPU channel 2 compare match/input capture A interrupt	
				1	Activated by TPU channel 3 compare match/input capture A interrupt	
		1	0	0	0	Activated by TPU channel 4 compare match/input capture A interrupt
					1	Activated by TPU channel 5 compare match/input capture A interrupt
			1	0	—	
				1	—	

## Channel B

Bit 3 DTF3	Bit 2 DTF2	Bit 1 DTF1	Bit 0 DTF0	Description	
0	0	0	0	— (Initial value)	
			1	Activated by A/D converter conversion end interrupt	
		1	0	Activated by $\overline{\text{DREQ}}$ pin falling edge input*	
			1	Activated by $\overline{\text{DREQ}}$ pin low-level input	
	1	0	0	Activated by SCI channel 0 transmission complete interrupt	
			1	Activated by SCI channel 0 reception complete interrupt	
		1	0	Activated by SCI channel 1 transmission complete interrupt	
			1	Activated by SCI channel 1 reception complete interrupt	
	1	0	0	0	Activated by TPU channel 0 compare match/input capture A interrupt
				1	Activated by TPU channel 1 compare match/input capture A interrupt
1			0	Activated by TPU channel 2 compare match/input capture A interrupt	
			1	Activated by TPU channel 3 compare match/input capture A interrupt	
1		0	0	Activated by TPU channel 4 compare match/input capture A interrupt	
			1	Activated by TPU channel 5 compare match/input capture A interrupt	
		1	0	—	
			1	—	

Note: \* Detected as a low level in the first transfer after transfer is enabled.

The same factor can be selected for more than one channel. In this case, activation starts with the highest-priority channel according to the relative channel priorities. For relative channel priorities, see section 5.5.13, DMAC Multi-Channel Operation.

## 5.2.5 DMA Band Control Register (DMABCR)

### DMABCRH

Bit	:	15	14	13	12	11	10	9	8
		FAE1	FAE0	SAE1	SAE0	DTA1B	DTA1A	DTA0B	DTA0A
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

### DMABCRL

Bit	:	7	6	5	4	3	2	1	0
		DTE1B	DTE1A	DTE0B	DTE0A	DTIE1B	DTIE1A	DTIE0B	DTIE0A
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DMABCR is a 16-bit readable/writable register that controls the operation of each DMAC channel.

DMABCR is initialized to H'0000 by a reset, and in hardware standby mode.

**Bit 15—Full Address Enable 1 (FAE1):** Specifies whether channel 1 is to be used in short address mode or full address mode.

In short address mode, channels 1A and 1B can be used as independent channels.

#### Bit 15

FAE1	Description
0	Short address mode (Initial value)
1	Full address mode

**Bit 14—Full Address Enable 0 (FAE0):** Specifies whether channel 0 is to be used in short address mode or full address mode.

In short address mode, channels 0A and 0B can be used as independent channels.

#### Bit 14

FAE0	Description
0	Short address mode (Initial value)
1	Full address mode

**Bit 13—Single Address Enable 1 (SAE1):** Specifies whether channel 1B is to be used for transfer in dual address mode or single address mode.

This bit is invalid in full address mode.

**Bit 13**

<b>SAE1</b>	<b>Description</b>	
0	Transfer in dual address mode	(Initial value)
1	Transfer in single address mode	

**Bit 12—Single Address Enable 0 (SAE0):** Specifies whether channel 0B is to be used for transfer in dual address mode or single address mode.

This bit is invalid in full address mode.

**Bit 12**

<b>SAE0</b>	<b>Description</b>	
0	Transfer in dual address mode	(Initial value)
1	Transfer in single address mode	

**Bits 11 to 8—Data Transfer Acknowledge (DTA):** These bits enable or disable clearing, when DMA transfer is performed, of the internal interrupt source selected by the data transfer factor setting.

When  $DTE = 1$  and  $DTA = 1$ , the internal interrupt source selected by the data transfer factor setting is cleared automatically by DMA transfer. When  $DTE = 1$  and  $DTA = 0$ , the internal interrupt source selected by the data transfer factor setting does not issue an interrupt request to the CPU or DTC.

When  $DTE = 0$  and  $DTA = 0$ , the internal interrupt source selected by the data transfer factor setting is not cleared when a transfer is performed, and can issue an interrupt request to the CPU or DTC in parallel. In this case, the interrupt source should be cleared by the CPU or DTC transfer.

When  $DTE = 0$ , the internal interrupt source selected by the data transfer factor setting issues an interrupt request to the CPU or DTC regardless of the DTA bit setting.

**Bit 11—Data Transfer Acknowledge 1B (DTA1B):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1B data transfer factor setting.

Bit 11 DTA1B	Description
0	Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value)
1	Clearing of selected internal interrupt source at time of DMA transfer is enabled

**Bit 10—Data Transfer Acknowledge 1A (DTA1A):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1A data transfer factor setting.

Bit 10 DTA1A	Description
0	Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value)
1	Clearing of selected internal interrupt source at time of DMA transfer is enabled

**Bit 9—Data Transfer Acknowledge 0B (DTA0B):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0B data transfer factor setting.

Bit 9 DTA0B	Description
0	Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value)
1	Clearing of selected internal interrupt source at time of DMA transfer is enabled

**Bit 8—Data Transfer Acknowledge 0A (DTA0A):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0A data transfer factor setting.

Bit 8 DTA0A	Description
0	Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value)
1	Clearing of selected internal interrupt source at time of DMA transfer is enabled

**Bits 7 to 4—Data Transfer Enable (DTE):** When DTE = 0, data transfer is disabled and the activation source selected by the data transfer factor setting is ignored. If the activation source is an internal interrupt, an interrupt request is issued to the CPU or DTC. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU or DTC.

The conditions for the DTE bit being cleared to 0 are as follows:

- When initialization is performed
- When the specified number of transfers have been completed in a transfer mode other than repeat mode
- When 0 is written to the DTE bit to forcibly abort the transfer, or for a similar reason

When DTE = 1, data transfer is enabled and the DMAC waits for a request by the activation source selected by the data transfer factor setting. When a request is issued by the activation source, DMA transfer is executed.

The condition for the DTE bit being set to 1 is as follows:

- When 1 is written to the DTE bit after the DTE bit is read as 0

**Bit 7—Data Transfer Enable 1B (DTE1B):** Enables or disables data transfer on channel 1B.

**Bit 7**

<b>DTE1B</b>	<b>Description</b>	
0	Data transfer disabled	(Initial value)
1	Data transfer enabled	

**Bit 6—Data Transfer Enable 1A (DTE1A):** Enables or disables data transfer on channel 1A.

**Bit 6**

<b>DTE1A</b>	<b>Description</b>	
0	Data transfer disabled	(Initial value)
1	Data transfer enabled	

**Bit 5—Data Transfer Enable 0B (DTE0B):** Enables or disables data transfer on channel 0B.

**Bit 5**

<b>DTE0B</b>	<b>Description</b>	
0	Data transfer disabled	(Initial value)
1	Data transfer enabled	

**Bit 4—Data Transfer Enable 0A (DTE0A):** Enables or disables data transfer on channel 0A.

**Bit 4**

<b>DTE0A</b>	<b>Description</b>	
0	Data transfer disabled	(Initial value)
1	Data transfer enabled	



**Bits 3 to 0—Data Transfer End Interrupt Enable (DTIE):** These bits enable or disable an interrupt to the CPU or DTC when transfer ends. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU or DTC.

A transfer end interrupt can be canceled either by clearing the DTIE bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the transfer counter and address register again, and then setting the DTE bit to 1.

**Bit 3—Data Transfer Interrupt Enable 1B (DTIE1B):** Enables or disables the channel 1B transfer end interrupt.

Bit 3 DTIE1B	Description	
0	Transfer end interrupt disabled	(Initial value)
1	Transfer end interrupt enabled	

**Bit 2—Data Transfer Interrupt Enable 1A (DTIE1A):** Enables or disables the channel 1A transfer end interrupt.

Bit 2 DTIE1A	Description	
0	Transfer end interrupt disabled	(Initial value)
1	Transfer end interrupt enabled	

**Bit 1—Data Transfer Interrupt Enable 0B (DTIE0B):** Enables or disables the channel 0B transfer end interrupt.

Bit 1 DTIE0B	Description	
0	Transfer end interrupt disabled	(Initial value)
1	Transfer end interrupt enabled	

**Bit 0—Data Transfer Interrupt Enable 0A (DTIE0A):** Enables or disables the channel 0A transfer end interrupt.

Bit 0 DTIE0A	Description	
0	Transfer end interrupt disabled	(Initial value)
1	Transfer end interrupt enabled	

## 5.3 Register Descriptions (2) (Full Address Mode)

Full address mode transfer is performed with channels A and B together. For details of full address mode setting, see table 5-4.

### 5.3.1 Memory Address Register (MAR)

Bit	:	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		—	—	—	—	—	—	—	—								
Initial value	:	0	0	0	0	0	0	0	0	*	*	*	*	*	*	*	*
R/W	:	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value	:	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

MAR is a 32-bit readable/writable register; MARA functions as the transfer source address register, and MARB as the destination address register.

MAR is composed of two 16-bit registers, MARH and MARL. The upper 8 bits of MARH are reserved: they are always read as 0, and cannot be modified.

MAR is incremented or decremented each time a byte or word transfer is executed, so that the source or destination memory address can be updated automatically. For details, see section 5.3.4, DMA Control Register (DMACR).

MAR is not initialized by a reset or in standby mode.

### 5.3.2 I/O Address Register (IOAR)

IOAR is not used in full address transfer.

### 5.3.3 Execute Transfer Count Register (ETCR)

ETCR is a 16-bit readable/writable register that specifies the number of transfers. The function of this register is different in normal mode and in block transfer mode.

ETCR is not initialized by a reset or in standby mode.

#### Normal Mode

ETCRA

##### Transfer Counter

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :		*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

In normal mode, ETCRA functions as a 16-bit transfer counter. ETCRA is decremented by 1 each time a transfer is performed, and transfer ends when the count reaches H'0000. ETCRB is not used at this time.

ETCRB

ETCRB is not used in normal mode.

#### Block Transfer Mode

ETCRA

##### Block Size Storage (ETCRAH)

Bit	:	15	14	13	12	11	10	9	8
Initial value :		*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

##### Block Size Counter (ETCRAL)

Bit	:	7	6	5	4	3	2	1	0
Initial value :		*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

## ETCRB

### Block Transfer Counter

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :		*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

\*: Undefined

In block transfer mode, ETCRAL functions as an 8-bit block size counter and ETCRAH holds the block size. ETCRAL is decremented each time a 1-byte or 1-word transfer is performed, and when the count reaches H'00, ETCRAL is loaded with the value in ETCRAH. So by setting the block size in ETCRAH and ETCRAL, it is possible to repeatedly transfer blocks consisting of any desired number of bytes or words.

ETCRB functions in block transfer mode, as a 16-bit block transfer counter. ETCRB is decremented by 1 each time a block is transferred, and transfer ends when the count reaches H'0000.

### 5.3.4 DMA Control Register (DMACR)

DMACR is a 16-bit readable/writable register that controls the operation of each DMAC channel. In full address mode, DMACRA and DMACRB have different functions.

DMACR is initialized to H'0000 by a reset, and in hardware standby mode.

#### DMACRA

Bit	:	15	14	13	12	11	10	9	8
		DTSZ	SAID	SAIDE	BLKDIR	BLKE	—	—	—
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### DMACRB

Bit	:	7	6	5	4	3	2	1	0
		—	DAID	DAIDE	—	DTF3	DTF2	DTF1	DTF0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Bit 15—Data Transfer Size (DTSZ):** Selects the size of data to be transferred at one time.

**Bit 15**

DTSZ	Description	
0	Byte-size transfer	(Initial value)
1	Word-size transfer	

**Bit 14—Source Address Increment/Decrement (SAID)**

**Bit 13—Source Address Increment/Decrement Enable (SAIDE):** These bits specify whether source address register MARA is to be incremented, decremented, or left unchanged, when data transfer is performed.

Bit 14 SAID	Bit 13 SAIDE	Description	
0	0	MARA is fixed	(Initial value)
	1	MARA is incremented after a data transfer <ul style="list-style-type: none"> <li>When DTSZ = 0, MARA is incremented by 1 after a transfer</li> <li>When DTSZ = 1, MARA is incremented by 2 after a transfer</li> </ul>	
1	0	MARA is fixed	
	1	MARA is decremented after a data transfer <ul style="list-style-type: none"> <li>When DTSZ = 0, MARA is decremented by 1 after a transfer</li> <li>When DTSZ = 1, MARA is decremented by 2 after a transfer</li> </ul>	

**Bit 12—Block Direction (BLKDIR)**

**Bit 11—Block Enable (BLKE):** These bits specify whether normal mode or block transfer mode is to be used. If block transfer mode is specified, the BLKDIR bit specifies whether the source side or the destination side is to be the block area.

Bit 12 BLKDIR	Bit 11 BLKE	Description	
0	0	Transfer in normal mode	(Initial value)
	1	Transfer in block transfer mode, destination side is block area	
1	0	Transfer in normal mode	
	1	Transfer in block transfer mode, source side is block area	

For operation in normal mode and block transfer mode, see section 5.5, Operation.

**Bits 10 to 7—Reserved:** Can be read or written to.

**Bit 6—Destination Address Increment/Decrement (DAID)**

**Bit 5—Destination Address Increment/Decrement Enable (DAIDE):** These bits specify whether destination address register MARB is to be incremented, decremented, or left unchanged, when data transfer is performed.

Bit 6 DAID	Bit 5 DAIDE	Description
0	0	MARB is fixed (Initial value)
	1	MARB is incremented after a data transfer <ul style="list-style-type: none"> <li>• When DTSZ = 0, MARB is incremented by 1 after a transfer</li> <li>• When DTSZ = 1, MARB is incremented by 2 after a transfer</li> </ul>
1	0	MARB is fixed
	1	MARB is decremented after a data transfer <ul style="list-style-type: none"> <li>• When DTSZ = 0, MARB is decremented by 1 after a transfer</li> <li>• When DTSZ = 1, MARB is decremented by 2 after a transfer</li> </ul>

**Bit 4—Reserved:** Can be read or written to.

**Bits 3 to 0—Data Transfer Factor (DTF3 to DTF0):** These bits select the data transfer factor (activation source). The factors that can be specified differ between normal mode and block transfer mode.

- Normal Mode

Bit 3 DTF3	Bit 2 DTF2	Bit 1 DTF1	Bit 0 DTF0	Description
0	0	0	0	— (Initial value)
			1	—
		1	0	Activated by $\overline{\text{DREQ}}$ pin falling edge input
			1	Activated by $\overline{\text{DREQ}}$ pin low-level input
	1	0	*	—
		1	0	Auto-request (cycle steal)
1	*	*	1	Auto-request (burst)
			*	—

\*: Don't care

- Block Transfer Mode

Bit 3 DTF3	Bit 2 DTF2	Bit 1 DTF1	Bit 0 DTF0	Description
0	0	0	0	— (Initial value)
			1	Activated by A/D converter conversion end interrupt
		1	0	Activated by $\overline{\text{DREQ}}$ pin falling edge input*
			1	Activated by $\overline{\text{DREQ}}$ pin low-level input
	1	0	0	Activated by SCI channel 0 transmission complete interrupt
			1	Activated by SCI channel 0 reception complete interrupt
		1	0	Activated by SCI channel 1 transmission complete interrupt
			1	Activated by SCI channel 1 reception complete interrupt
1	0	0	0	Activated by TPU channel 0 compare match/input capture A interrupt
			1	Activated by TPU channel 1 compare match/input capture A interrupt
		1	0	Activated by TPU channel 2 compare match/input capture A interrupt
			1	Activated by TPU channel 3 compare match/input capture A interrupt
	1	0	0	Activated by TPU channel 4 compare match/input capture A interrupt
			1	Activated by TPU channel 5 compare match/input capture A interrupt
		1	0	—
			1	—

Note: \* Detected as a low level in the first transfer after transfer is enabled.

The same factor can be selected for more than one channel. In this case, activation starts with the highest-priority channel according to the relative channel priorities. For relative channel priorities, see section 5.5.13, DMAC Multi-Channel Operation.

### 5.3.5 DMA Band Control Register (DMABCR)

#### DMABCRH

Bit	:	15	14	13	12	11	10	9	8
		FAE1	FAE0	—	—	DTA1	—	DTA0	—
Initial value	:	0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### DMABCRL

Bit	:	7	6	5	4	3	2	1	0
	:	DTME1	DTE1	DTME0	DTE0	DTIE1B	DTIE1A	DTIE0B	DTIE0A
Initial value	:	0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DMABCR is a 16-bit readable/writable register that controls the operation of each DMAC channel.

DMABCR is initialized to H'0000 by a reset, and in hardware standby mode.

**Bit 15—Full Address Enable 1 (FAE1):** Specifies whether channel 1 is to be used in short address mode or full address mode.

In full address mode, channels 1A and 1B are used together as a single channel.

#### Bit 15

FAE1	Description
0	Short address mode (Initial value)
1	Full address mode

**Bit 14—Full Address Enable 0 (FAE0):** Specifies whether channel 0 is to be used in short address mode or full address mode.

In full address mode, channels 0A and 0B are used together as a single channel.

#### Bit 14

FAE0	Description
0	Short address mode (Initial value)
1	Full address mode



**Bits 13 and 12—Reserved:** Can be read or written to.

**Bits 11 and 9—Data Transfer Acknowledge (DTA):** These bits enable or disable clearing, when DMA transfer is performed, of the internal interrupt source selected by the data transfer factor setting.

When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting is cleared automatically by DMA transfer. When DTE = 1 and DTA = 1, the internal interrupt source selected by the data transfer factor setting does not issue an interrupt request to the CPU or DTC.

When DTE = 1 and DTA = 0, the internal interrupt source selected by the data transfer factor setting is not cleared when a transfer is performed, and can issue an interrupt request to the CPU or DTC in parallel. In this case, the interrupt source should be cleared by the CPU or DTC transfer.

When DTE = 0, the internal interrupt source selected by the data transfer factor setting issues an interrupt request to the CPU or DTC regardless of the DTA bit setting.

The state of the DTME bit does not affect the above operations.

**Bit 11—Data Transfer Acknowledge 1 (DTA1):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 1 data transfer factor setting.

Bit 11 DTA1	Description
0	Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value)
1	Clearing of selected internal interrupt source at time of DMA transfer is enabled

**Bit 9—Data Transfer Acknowledge 0 (DTA0):** Enables or disables clearing, when DMA transfer is performed, of the internal interrupt source selected by the channel 0 data transfer factor setting.

Bit 9 DTA0	Description
0	Clearing of selected internal interrupt source at time of DMA transfer is disabled (Initial value)
1	Clearing of selected internal interrupt source at time of DMA transfer is enabled

**Bits 10 and 8—Reserved:** Can be read or written to.

**Bits 7 and 5—Data Transfer Master Enable (DTME):** Together with the DTE bit, these bits control enabling or disabling of data transfer on the relevant channel. When both the DTME bit and the DTE bit are set to 1, transfer is enabled for the channel.

If the relevant channel is in the middle of a burst mode transfer when an NMI interrupt is generated, the DTME bit is cleared, the transfer is interrupted, and bus mastership passes to the CPU. When the DTME bit is subsequently set to 1 again, the interrupted transfer is resumed. In block transfer mode, however, the DTME bit is not cleared by an NMI interrupt, and transfer is not interrupted.

The conditions for the DTME bit being cleared to 0 are as follows:

- When initialization is performed
- When NMI is input in burst mode
- When 0 is written to the DTME bit

The condition for DTME being set to 1 is as follows:

- When 1 is written to DTME after DTME is read as 0

**Bit 7—Data Transfer Master Enable 1 (DTME1):** Enables or disables data transfer on channel 1.

**Bit 7**

<b>DTME1</b>	<b>Description</b>
0	Data transfer disabled. In burst mode, cleared to 0 by an NMI interrupt (Initial value)
1	Data transfer enabled

**Bit 5—Data Transfer Master Enable 0 (DTME0):** Enables or disables data transfer on channel 0.

**Bit 5**

<b>DTME0</b>	<b>Description</b>
0	Data transfer disabled. In normal mode, cleared to 0 by an NMI interrupt (Initial value)
1	Data transfer enabled

**Bits 6 and 4—Data Transfer Enable (DTE):** When DTE = 0, data transfer is disabled and the activation source selected by the data transfer factor setting is ignored. If the activation source is an internal interrupt, an interrupt request is issued to the CPU or DTC. If the DTIE bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU.

The conditions for the DTE bit being cleared to 0 are as follows:

- When initialization is performed
- When the specified number of transfers have been completed
- When 0 is written to the DTE bit to forcibly abort the transfer, or for a similar reason

When DTE = 1 and DTME = 1, data transfer is enabled and the DMAC waits for a request by the activation source selected by the data transfer factor setting. When a request is issued by the activation source, DMA transfer is executed.

The condition for the DTE bit being set to 1 is as follows:

- When 1 is written to the DTE bit after the DTE bit is read as 0

**Bit 6—Data Transfer Enable 1 (DTE1):** Enables or disables data transfer on channel 1.

Bit 6 DTE1	Description	
0	Data transfer disabled	(Initial value)
1	Data transfer enabled	

**Bit 4—Data Transfer Enable 0 (DTE0):** Enables or disables data transfer on channel 0.

Bit 4 DTE0	Description	
0	Data transfer disabled	(Initial value)
1	Data transfer enabled	

**Bits 3 and 1—Data Transfer Interrupt Enable B (DTIEB):** These bits enable or disable an interrupt to the CPU or DTC when transfer is interrupted. If the DTIEB bit is set to 1 when DTME = 0, the DMAC regards this as indicating a break in the transfer, and issues a transfer break interrupt request to the CPU or DTC.

A transfer break interrupt can be canceled either by clearing the DTIEB bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the DTME bit to 1.

**Bit 3—Data Transfer Interrupt Enable 1B (DTIE1B):** Enables or disables the channel 1 transfer break interrupt.

**Bit 3**

<b>DTIE1B</b>	<b>Description</b>	
0	Transfer break interrupt disabled	(Initial value)
1	Transfer break interrupt enabled	

**Bit 1—Data Transfer Interrupt Enable 0B (DTIE0B):** Enables or disables the channel 0 transfer break interrupt.

**Bit 1**

<b>DTIE0B</b>	<b>Description</b>	
0	Transfer break interrupt disabled	(Initial value)
1	Transfer break interrupt enabled	

**Bits 2 and 0—Data Transfer End Interrupt Enable A (DTIEA):** These bits enable or disable an interrupt to the CPU or DTC when transfer ends. If the DTIEA bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU or DTC.

A transfer end interrupt can be canceled either by clearing the DTIEA bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the transfer counter and address register again, and then setting the DTE bit to 1.

**Bit 2—Data Transfer Interrupt Enable 1A (DTIE1A):** Enables or disables the channel 1 transfer end interrupt.

**Bit 2**

<b>DTIE1A</b>	<b>Description</b>	
0	Transfer end interrupt disabled	(Initial value)
1	Transfer end interrupt enabled	

**Bit 0—Data Transfer Interrupt Enable 0A (DTIE0A):** Enables or disables the channel 0 transfer end interrupt.

**Bit 0**

<b>DTIE0A</b>	<b>Description</b>	
0	Transfer end interrupt disabled	(Initial value)
1	Transfer end interrupt enabled	

## 5.4 Register Descriptions (3)

### 5.4.1 DMA Write Enable Register (DMAWER)

The DMAC can activate the DTC with a transfer end interrupt, rewrite the channel on which the transfer ended using a DTC chain transfer, and reactivate the DTC. DMAWER applies restrictions so that specific bits of DMACR for the specific channel, and also DMATCR and DMABCR, can be changed to prevent inadvertent rewriting of registers other than those for the channel concerned. The restrictions applied by DMAWER are valid for the DTC.

Figure 5-2 shows the transfer areas for activating the DTC with a channel 0A transfer end interrupt, and reactivating channel 0A. The address register and count register area is re-set by the first DTC transfer, then the control register area is re-set by the second DTC chain transfer.

When re-setting the control register area, perform masking by setting bits in DMAWER to prevent modification of the contents of the other channels.

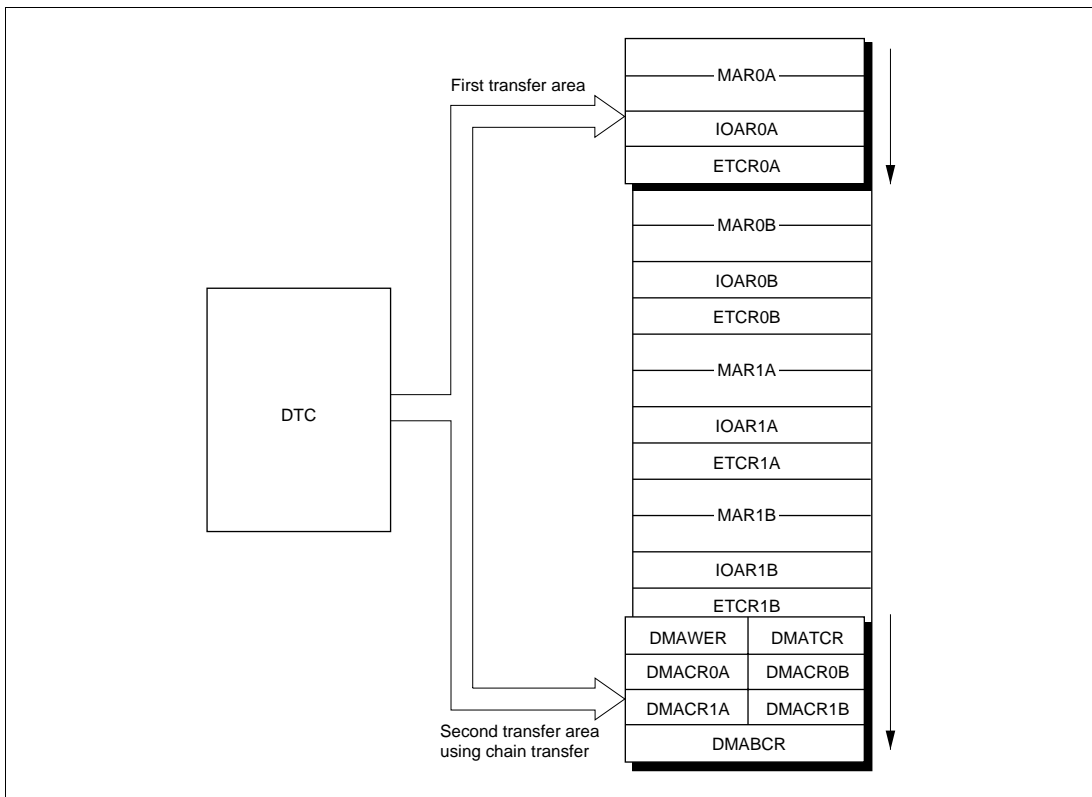


Figure 5-2 Areas for Register Re-Setting by DTC (Example: Channel 0A)

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	WE1B	WE1A	WE0B	WE0A
Initial value :		0	0	0	0	0	0	0	0
R/W	:	—	—	—	—	R/W	R/W	R/W	R/W

DMAWER is an 8-bit readable/writable register that controls enabling or disabling of writes to DMACR, DMABCR, and DMATCR by the DTC.

DMAWER is initialized to H'00 by a reset, and in hardware standby mode.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Write Enable 1B (WE1B):** Enables or disables writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR, by the DTC.

**Bit 3**

WE1B	Description
0	Writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR are disabled (Initial value)
1	Writes to all bits in DMACR1B, bits 11, 7, and 3 in DMABCR, and bit 5 in DMATCR are enabled

**Bit 2—Write Enable 1A (WE1A):** Enables or disables writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR, by the DTC.

**Bit 2**

WE1A	Description
0	Writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR are disabled (Initial value)
1	Writes to all bits in DMACR1A, and bits 10, 6, and 2 in DMABCR are enabled

**Bit 1—Write Enable 0B (WE0B):** Enables or disables writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR, by the DTC.

**Bit 1**

WE0B	Description
0	Writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR are disabled (Initial value)
1	Writes to all bits in DMACR0B, bits 9, 5, and 1 in DMABCR, and bit 4 in DMATCR are enabled

**Bit 0—Write Enable 0A (WE0A):** Enables or disables writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR, by the DTC.

Bit 0 WE0A	Description
0	Writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR are disabled (Initial value)
1	Writes to all bits in DMACR0A, and bits 8, 4, and 0 in DMABCR are enabled

Writes by the DTC to bits 15 to 12 (FAE and SAE) in DMABCR are invalid regardless of the DMAWER settings. These bits should be changed, if necessary, by CPU processing.

In writes by the DTC to bits 7 to 4 (DTE) in DMABCR, 1 can be written without first reading 0. To reactivate a channel set to full address mode, write 1 to both Write Enable A and Write Enable B for the channel to be reactivated.

MAR, IOAR, and ETCR are always write-enabled regardless of the DMAWER settings. When modifying these registers, the channel for which the modification is to be made should be halted.

#### 5.4.2 DMA Terminal Control Register (DMATCR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	TEE1	TEE0	—	—	—	—
Initial value :		0	0	0	0	0	0	0	0
R/W	:	—	—	R/W	R/W	—	—	—	—

DMATCR is an 8-bit readable/writable register that controls enabling or disabling of DMAC transfer end pin output. A port can be set for output automatically, and a transfer end signal output, by setting the appropriate bit.

DMATCR is initialized to H'00 by a reset, and in hardware standby mode.

**Bits 7 and 6—Reserved:** Read-only bits, always read as 0.

**Bit 5—Transfer End Enable 1 (TEE1):** Enables or disables transfer end pin 1 ( $\overline{\text{TEND1}}$ ) output.

Bit 5 TEE1	Description
0	$\overline{\text{TEND1}}$ pin output disabled (Initial value)
1	$\overline{\text{TEND1}}$ pin output enabled

**Bit 4—Transfer End Enable 0 (TEE0):** Enables or disables transfer end pin 0 ( $\overline{TEND0}$ ) output.

**Bit 4**

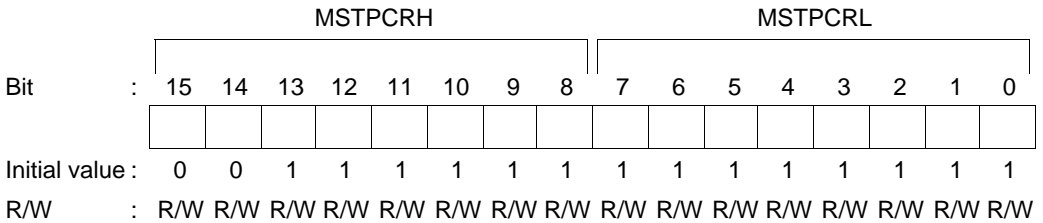
TEE0	Description	
0	$\overline{TEND0}$ pin output disabled	(Initial value)
1	$\overline{TEND0}$ pin output enabled	

The  $\overline{TEND}$  pins are assigned only to channel B in short address mode.

The transfer end signal indicates the transfer cycle in which the transfer counter reached 0, regardless of the transfer source. An exception is block transfer mode, in which the transfer end signal indicates the transfer cycle in which the block counter reached 0.

**Bits 3 to 0—Reserved:** Read-only bits, always read as 0.

**5.4.3 Module Stop Control Register (MSTPCR)**



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP15 bit in MSTPCR is set to 1, the DMAC operation stops at the end of the bus cycle and a transition is made to module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 15—Module Stop (MSTP15):** Specifies the DMAC module stop mode.

**Bits 15**

MSTP15	Description	
0	DMAC module stop mode cleared	(Initial value)
1	DMAC module stop mode set	



## 5.5 Operation

### 5.5.1 Transfer Modes

Table 5-5 lists the DMAC modes.

**Table 5-5 DMAC Transfer Modes**

Transfer Mode		Transfer Source	Remarks
Short address mode	Dual address mode	(1) Sequential mode	<ul style="list-style-type: none"> <li>Up to 4 channels can operate independently</li> <li>External request applies to channel B only</li> <li>Single address mode applies to channel B only</li> <li>Modes (1), (2), and (3) can also be specified for single address mode</li> </ul>
		(2) Idle mode	
		(3) Repeat mode	
	(4) Single address mode		
Full address mode	(5) Normal mode	<ul style="list-style-type: none"> <li>External request</li> <li>Auto-request</li> </ul>	<ul style="list-style-type: none"> <li>Max. 2-channel operation, combining channels A and B</li> <li>With auto-request, burst mode transfer or cycle steal transfer can be selected</li> </ul>
	(6) Block transfer mode	<ul style="list-style-type: none"> <li>TPU channel 0 to 5 compare match/input capture A interrupt</li> <li>SCI transmission complete interrupt</li> <li>SCI reception complete interrupt</li> <li>A/D converter conversion end interrupt</li> <li>External request</li> </ul>	

Operation in each mode is summarized below.

**Sequential Mode:** In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. One address is specified as 24 bits, and the other as 16 bits. The transfer direction is programmable.

**Idle Mode:** In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. One address is specified as 24 bits, and the other as 16 bits. The transfer source address and transfer destination address are fixed. The transfer direction is programmable.

**Repeat Mode:** In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. When the specified number of transfers have been completed, the addresses and transfer counter are restored to their original settings, and operation is continued. No interrupt request is sent to the CPU or DTC. One address is specified as 24 bits, and the other as 16 bits. The transfer direction is programmable.

**Single Address Mode:** In response to a single transfer request, the specified number of transfers are carried out between external memory and an external device, one byte or one word at a time. Unlike dual address mode, source and destination accesses are performed in parallel. Therefore, either the source or the destination is an external device which can be accessed with a strobe alone, using the  $\overline{\text{DACK}}$  pin. One address is specified as 24 bits, and for the other, the pin is set automatically. The transfer direction is programmable.

Sequential mode, idle mode, and repeat mode can also be specified for single address mode.

## Normal Mode

- Auto-request

By means of register settings only, the DMAC is activated, and transfer continues until the specified number of transfers have been completed. An interrupt request can be sent to the CPU or DTC when transfer is completed. Both addresses are specified as 24 bits.

- Cycle steal mode

The bus is released to another bus master after each byte or word transfer.

- Burst mode

The bus is held and transfer continued until the specified number of transfers have been completed.

- External request

In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. Both addresses are specified as 24 bits.

**Block Transfer Mode:** In response to a single transfer request, a block transfer of the specified block size is carried out. This is repeated the specified number of times, once each time there is a transfer request. At the end of each single block transfer, one address is restored to its original setting. An interrupt request can be sent to the CPU or DTC when the specified number of block transfers have been completed. Both addresses are specified as 24 bits.

### 5.5.2 Sequential Mode

Sequential mode can be specified by clearing the RPE bit in DMACR to 0. In sequential mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCR.

One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR.

Table 5-6 summarizes register functions in sequential mode.

**Table 5-6 Register Functions in Sequential Mode**

Register	Function		Initial Setting	Operation
	DTDIR = 0	DTDIR = 1		
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%;"></span> <span style="width: 80%; text-align: center;">MAR</span> <span style="width: 10%;"></span> </div> </div>	Source address register	Destination address register	Start address of transfer destination or transfer source	Incremented/decrypted every transfer
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>15</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%; text-align: center;">H'FF</span> <span style="width: 80%; text-align: center;">IOAR</span> <span style="width: 10%;"></span> </div> </div>	Destination address register	Source address register	Start address of transfer source or transfer destination	Fixed
<div style="display: flex; justify-content: space-between;"> <span>15</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%;"></span> <span style="width: 80%; text-align: center;">ETCR</span> <span style="width: 10%;"></span> </div> </div>	Transfer counter		Number of transfers	Decrypted every transfer; transfer ends when count reaches H'0000

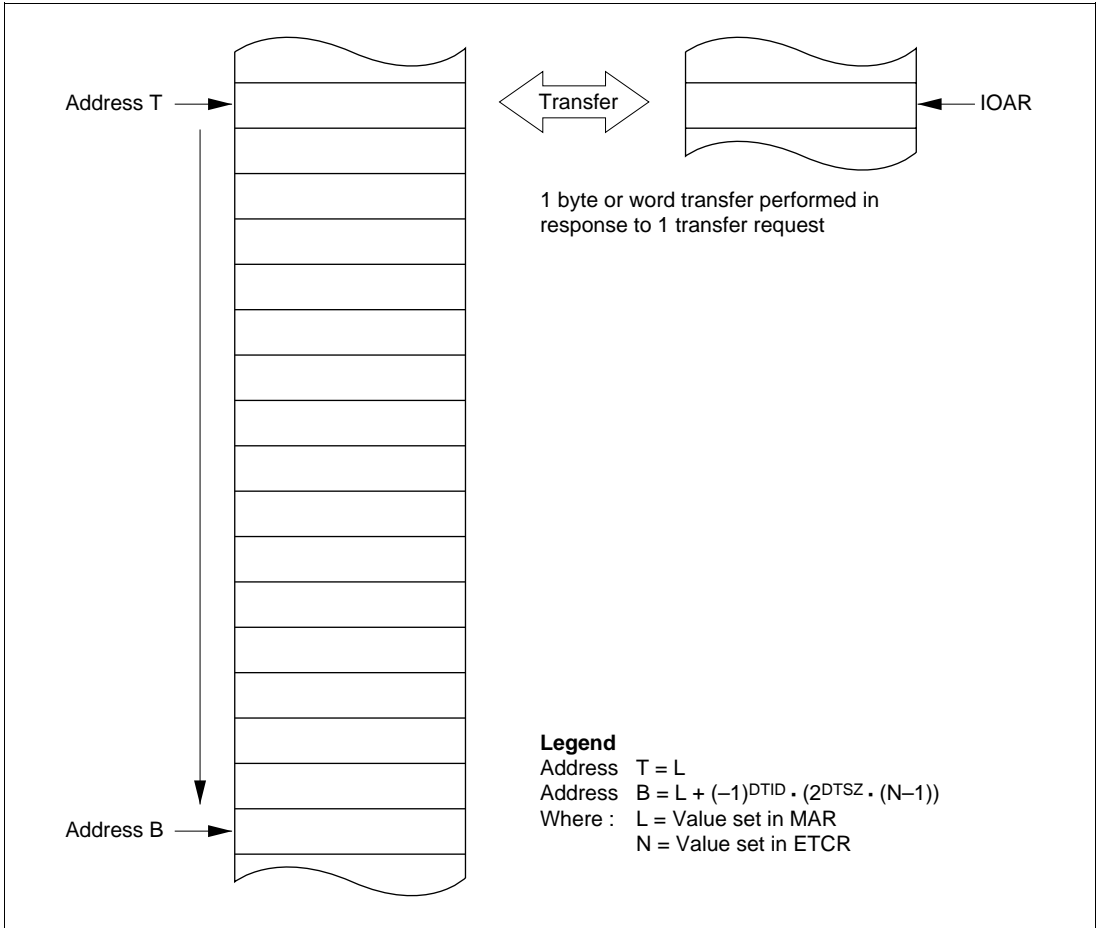
#### Legend

MAR : Memory address register  
 IOAR : I/O address register  
 ETCR : Execute transfer count register  
 DTDIR : Data transfer direction bit

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is incremented or decremented by 1 or 2 each time a byte or word is transferred.

IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF.

Figure 5-3 illustrates operation in sequential mode.



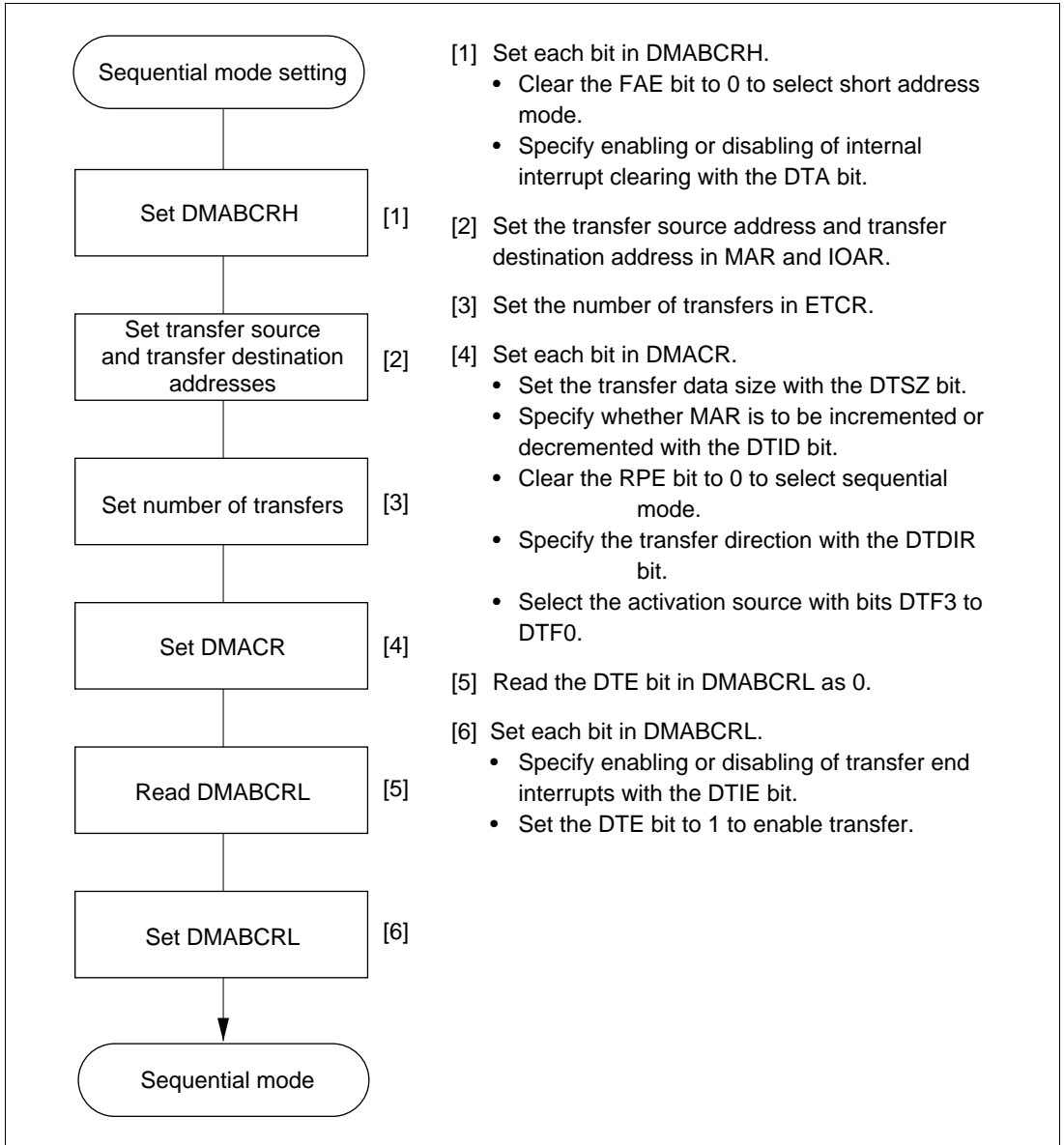
**Figure 5-3 Operation in Sequential Mode**

The number of transfers is specified as 16 bits in ETCR. ETCR is decremented by 1 each time a transfer is executed, and when its value reaches H'0000, the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU or DTC.

The maximum number of transfers, when H'0000 is set in ETCR, is 65,536.

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts. External requests can be set for channel B only.

Figure 5-4 shows an example of the setting procedure for sequential mode.



**Figure 5-4 Example of Sequential Mode Setting Procedure**

### 5.5.3 Idle Mode

Idle mode can be specified by setting the RPE bit and DTIE bit in DMACR to 1. In idle mode, one byte or word is transferred in response to a single transfer request, and this is executed the number of times specified in ETCR.

One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR.

Table 5-7 summarizes register functions in idle mode.

**Table 5-7 Register Functions in Idle Mode**

Register	Function		Initial Setting	Operation
	DTDIR = 0	DTDIR = 1		
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 15px; border-right: 1px dashed black;"></span> <span style="width: 100px; text-align: center;">MAR</span> <span style="width: 15px; border-left: 1px dashed black;"></span> </div> </div>	Source address register	Destination address register	Start address of transfer destination or transfer source	Fixed
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>15</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 40px; text-align: center;">H'FF</span> <span style="width: 100px; text-align: center;">IOAR</span> </div> </div>	Destination address register	Source address register	Start address of transfer source or transfer destination	Fixed
<div style="display: flex; justify-content: space-between;"> <span>15</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 100px; text-align: center;">ETCR</span> </div> </div>	Transfer counter		Number of transfers	Decremented every transfer; transfer ends when count reaches H'0000

#### Legend

MAR : Memory address register

IOAR : I/O address register

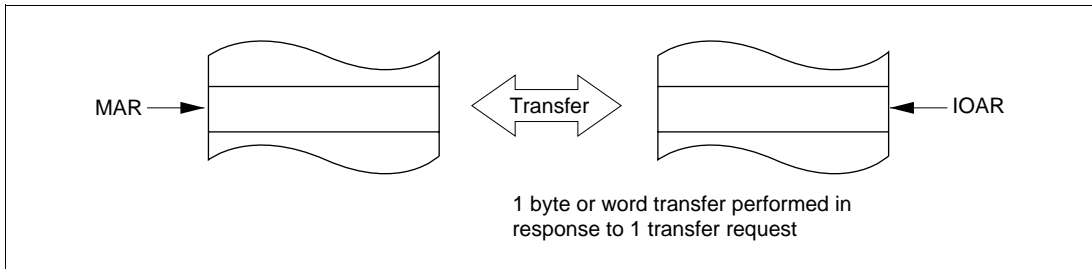
ETCR : Execute transfer count register

DTDIR : Data transfer direction bit

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is neither incremented nor decremented each time a byte or word is transferred.

IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF.

Figure 5-5 illustrates operation in idle mode.



**Figure 5-5 Operation in Idle Mode**

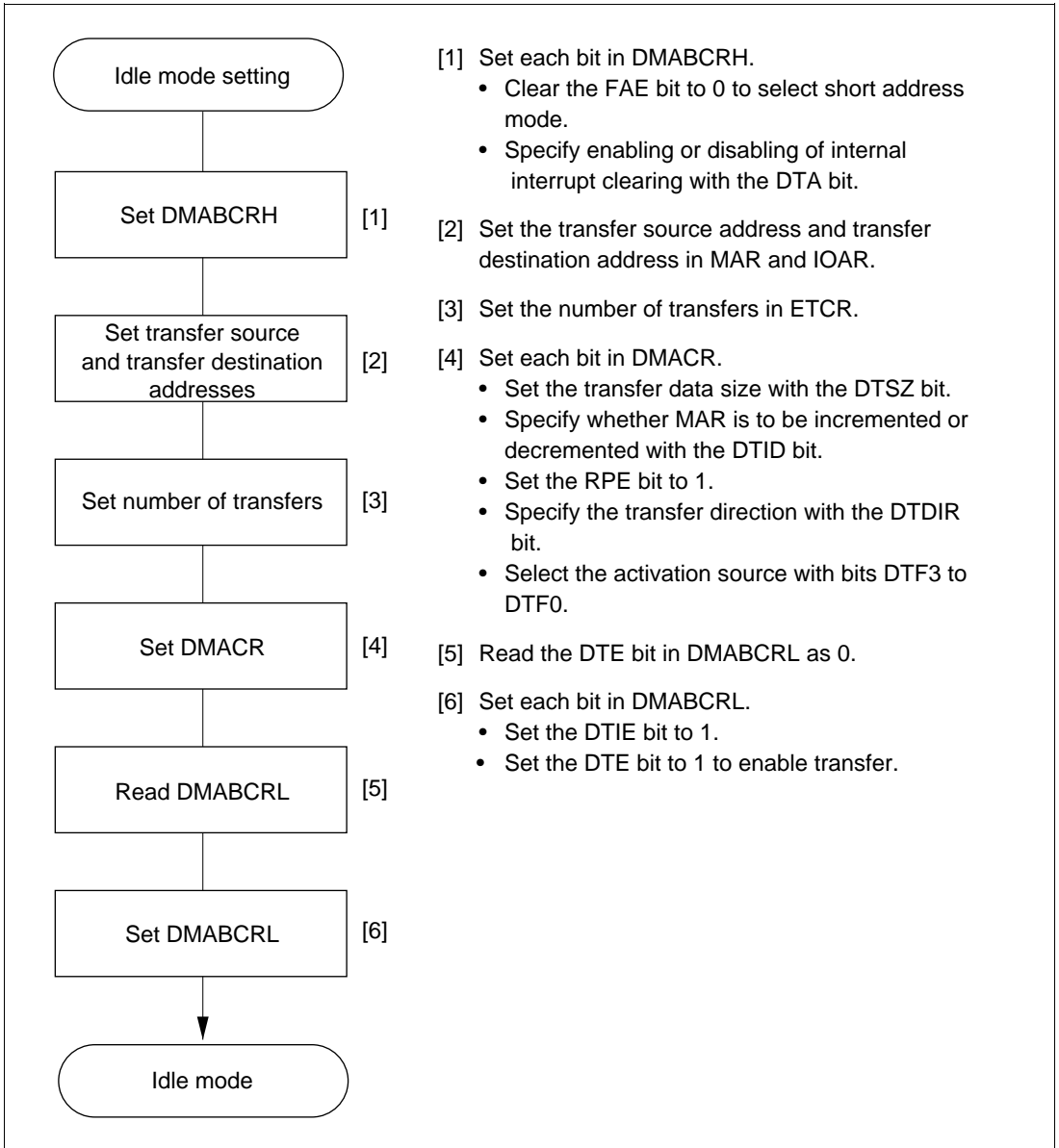
The number of transfers is specified as 16 bits in ETCR. ETCR is decremented by 1 each time a transfer is executed, and when its value reaches H'0000, the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU or DTC.

The maximum number of transfers, when H'0000 is set in ETCR, is 65,536.

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts. External requests can be set for channel B only.

When the DMAC is used in single address mode, only channel B can be set.

Figure 5-6 shows an example of the setting procedure for idle mode.



**Figure 5-6 Example of Idle Mode Setting Procedure**



## 5.5.4 Repeat Mode

Repeat mode can be specified by setting the RPE bit in DMACR to 1, and clearing the DTIE bit to 0. In repeat mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCR. On completion of the specified number of transfers, MAR and ETCRL are automatically restored to their original settings and operation continues.

One address is specified by MAR, and the other by IOAR. The transfer direction can be specified by the DTDIR bit in DMACR.

Table 5-8 summarizes register functions in repeat mode.

**Table 5-8 Register Functions in Repeat Mode**

Register	Function		Initial Setting	Operation
	DTDIR = 0	DTDIR = 1		
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%;"></span> <span style="width: 80%; text-align: center;">MAR</span> <span style="width: 10%;"></span> </div> </div>	Source address register	Destination address register	Start address of transfer destination or transfer source	Incremented/decrypted every transfer. Initial setting is restored when value reaches H'0000
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>15</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%; text-align: center;">H'FF</span> <span style="width: 80%; text-align: center;">IOAR</span> <span style="width: 10%;"></span> </div> </div>	Destination address register	Source address register	Start address of transfer source or transfer destination	Fixed
<div style="display: flex; justify-content: space-between;"> <span>7</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%;"></span> <span style="width: 80%; text-align: center;">ETCRH</span> <span style="width: 10%;"></span> </div> </div>	Holds number of transfers		Number of transfers	Fixed
-----				
<div style="display: flex; justify-content: space-between;"> <span>7</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <div style="display: flex; justify-content: space-between;"> <span style="width: 10%;"></span> <span style="width: 80%; text-align: center;">ETCRL</span> <span style="width: 10%;"></span> </div> </div>	Transfer counter		Number of transfers	Decrypted every transfer. Loaded with ETCRH value when count reaches H'00

### Legend

MAR : Memory address register

IOAR : I/O address register

ETCR : Execute transfer count register

DTDIR : Data transfer direction bit

MAR specifies the start address of the transfer source or transfer destination as 24 bits. MAR is incremented or decremented by 1 or 2 each time a byte or word is transferred.

IOAR specifies the lower 16 bits of the other address. The 8 bits above IOAR have a value of H'FF.

The number of transfers is specified as 8 bits by ETCRH and ETCRL. The maximum number of transfers, when H'00 is set in both ETCRH and ETCRL, is 256.

In repeat mode, ETCRL functions as the transfer counter, and ETCRH is used to hold the number of transfers. ETCRL is decremented by 1 each time a transfer is executed, and when its value reaches H'00, it is loaded with the value in ETCRH. At the same time, the value set in MAR is restored in accordance with the values of the DTSZ and DTID bits in DMACR. The MAR restoration operation is as shown below.

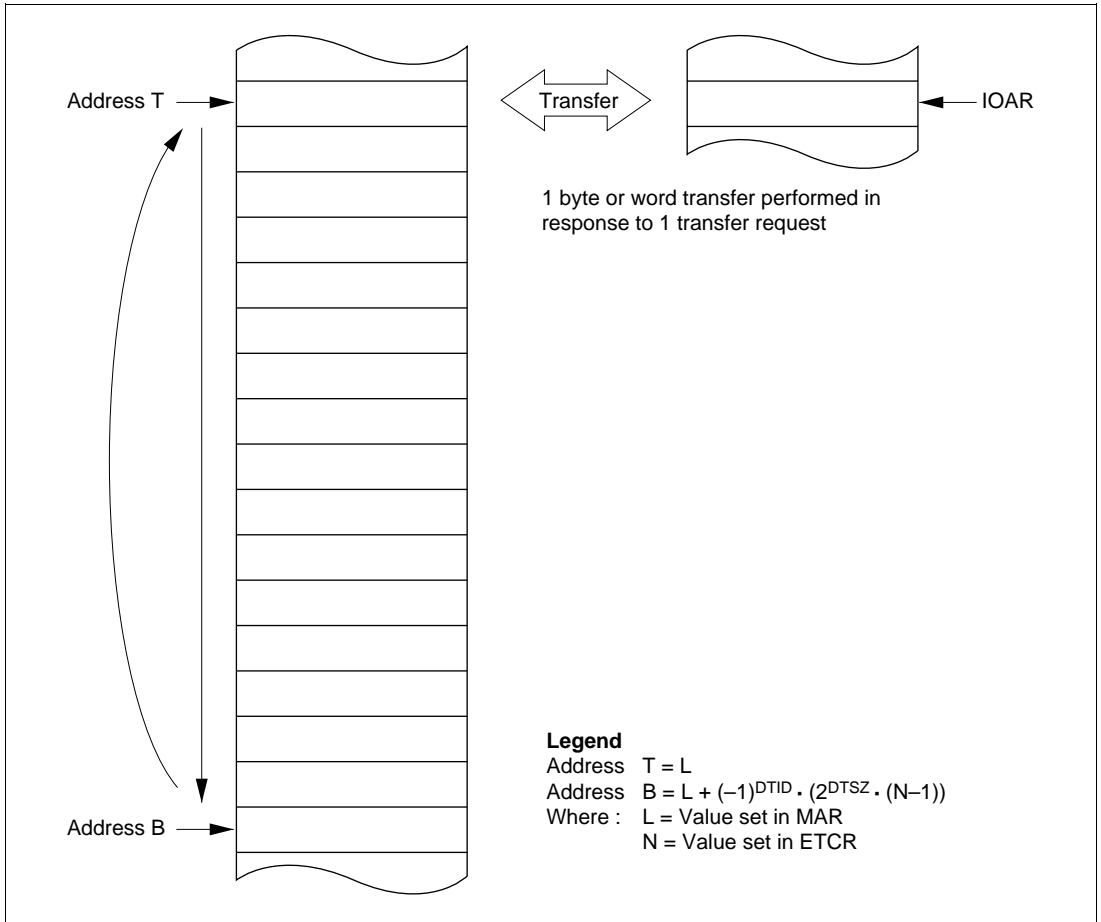
$$\text{MAR} = \text{MAR} - (-1)^{\text{DTID}} \cdot 2^{\text{DTSZ}} \cdot \text{ETCRH}$$

The same value should be set in ETCRH and ETCRL.

In repeat mode, operation continues until the DTE bit is cleared. To end the transfer operation, therefore, the DTE bit should be cleared to 0. A transfer end interrupt request is not sent to the CPU or DTC.

By setting the DTE bit to 1 again after it has been cleared, the operation can be restarted from the transfer after that terminated when the DTE bit was cleared.

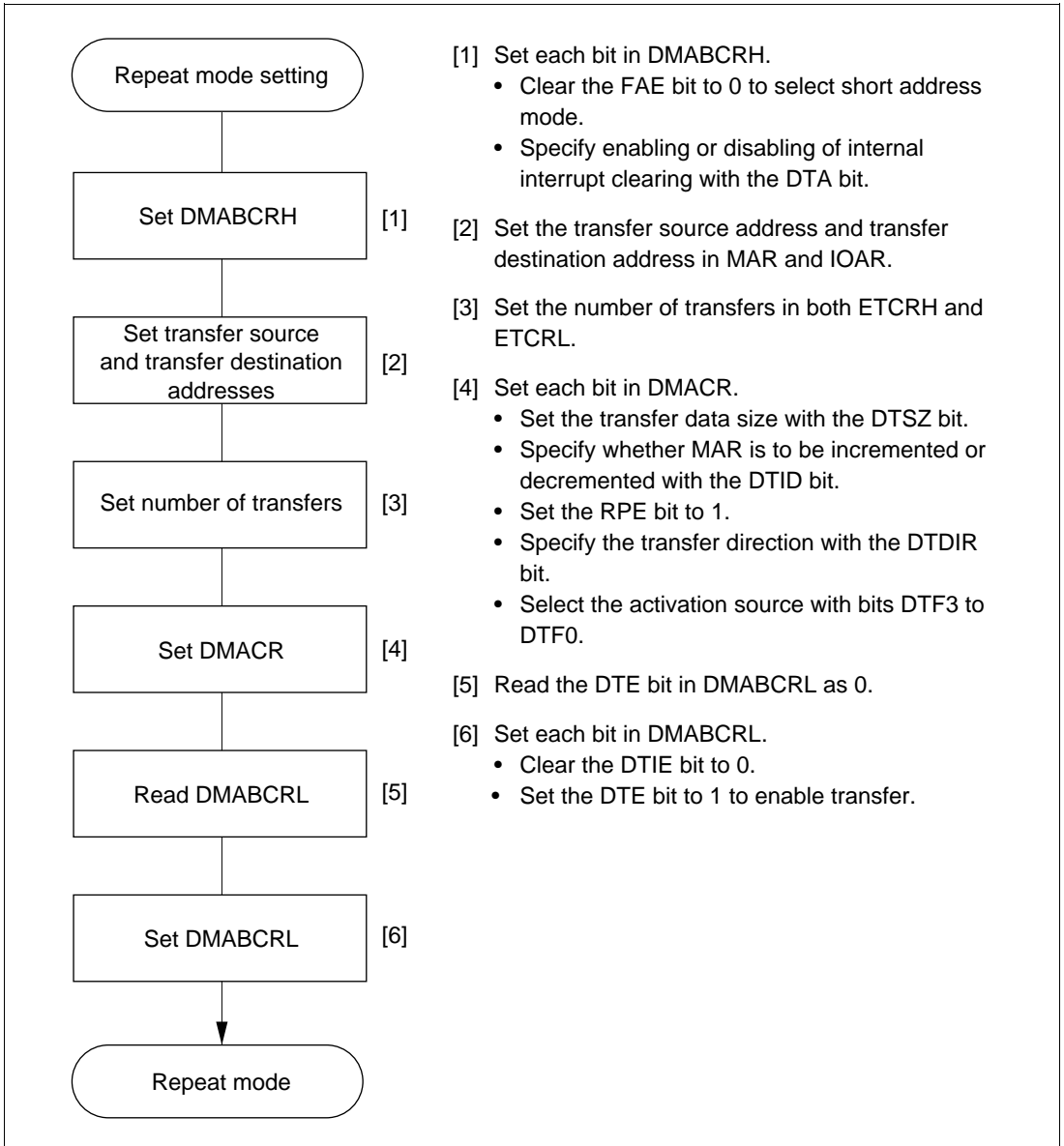
Figure 5-7 illustrates operation in repeat mode.



**Figure 5-7 Operation in Repeat mode**

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts. External requests can be set for channel B only.

Figure 5-8 shows an example of the setting procedure for repeat mode.



**Figure 5-8 Example of Repeat Mode Setting Procedure**

### 5.5.5 Single Address Mode

Single address mode can only be specified for channel B. This mode can be specified by setting the SAE bit in DMABCR to 1 in short address mode.

One address is specified by MAR, and the other is set automatically to the data transfer acknowledge pin ( $\overline{\text{DACK}}$ ). The transfer direction can be specified by the DTDIR bit in DMACR.

Table 5-9 summarizes register functions in single address mode.

**Table 5-9 Register Functions in Single Address Mode**

Register	Function		Initial Setting	Operation
	DTDIR = 0	DTDIR = 1		
<div style="display: flex; align-items: center;"> <span style="margin-right: 10px;">23</span> <div style="border: 1px solid black; padding: 2px; display: flex; align-items: center; justify-content: center;"> <span style="margin: 0 5px;">0</span> <span style="margin: 0 5px;">MAR</span> <span style="margin: 0 5px;">0</span> </div> </div>	Source address register	Destination address register	Start address of transfer destination or transfer source	*
$\overline{\text{DACK}}$ pin	Write strobe	Read strobe	(Set automatically by SAE bit; IOAR is invalid)	Strobe for external device
<div style="display: flex; align-items: center;"> <span style="margin-right: 10px;">15</span> <div style="border: 1px solid black; padding: 2px; display: flex; align-items: center; justify-content: center;"> <span style="margin: 0 5px;">0</span> <span style="margin: 0 5px;">ETCR</span> <span style="margin: 0 5px;">0</span> </div> </div>	Transfer counter		Number of transfers	*

#### Legend

MAR : Memory address register

IOAR : I/O address register

ETCR : Execute transfer register

DTDIR : Data transfer direction bit

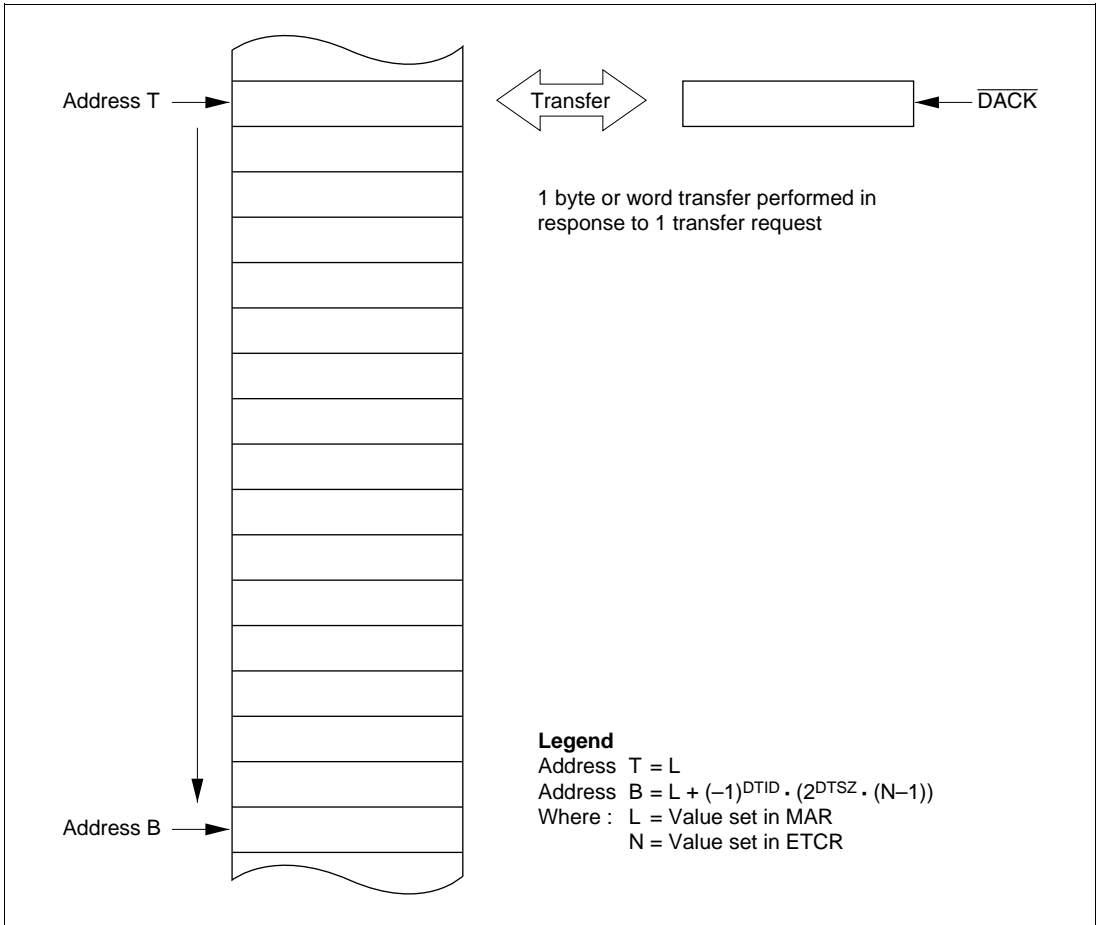
$\overline{\text{DACK}}$  : Data transfer acknowledge

Note: \* See the operation descriptions in sections 5.5.2, Sequential Mode, 5.5.3, Idle Mode, and 5.5.4, Repeat Mode.

MAR specifies the start address of the transfer source or transfer destination as 24 bits.

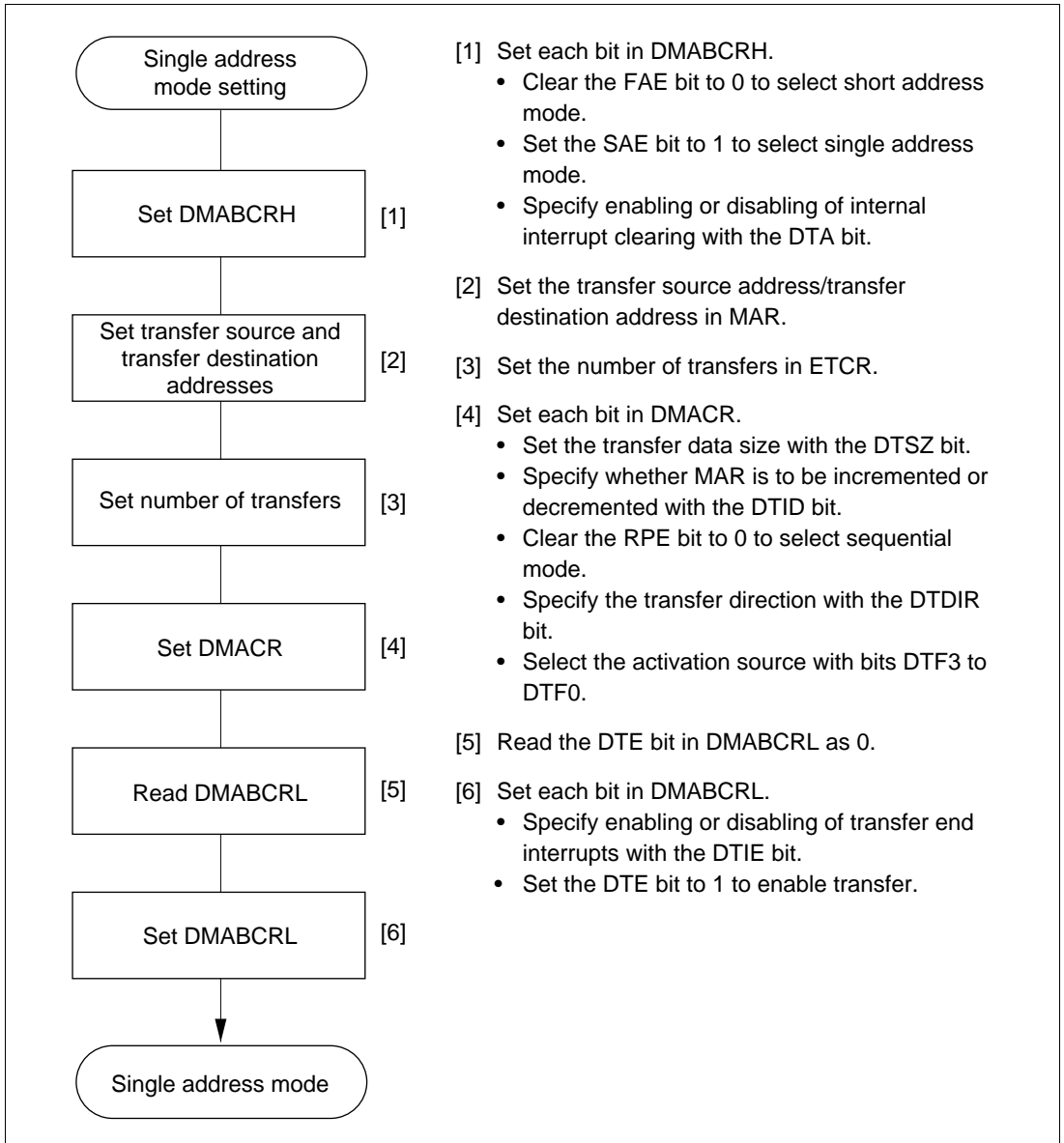
IOAR is invalid; in its place the strobe for external devices ( $\overline{\text{DACK}}$ ) is output.

Figure 5-9 illustrates operation in single address mode (when sequential mode is specified).



**Figure 5-9 Operation in Single Address Mode (When Sequential Mode is Specified)**

Figure 5-10 shows an example of the setting procedure for single address mode (when sequential mode is specified).



**Figure 5-10 Example of Single Address Mode Setting Procedure (When Sequential Mode is Specified)**


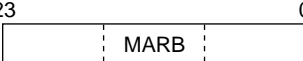

## 5.5.6 Normal Mode

In normal mode, transfer is performed with channels A and B used in combination. Normal mode can be specified by setting the FAE bit in DMABCR to 1 and clearing the BLKE bit in DMACRA to 0.

In normal mode, MAR is updated after each byte or word transfer in response to a single transfer request, and this is executed the number of times specified in ETCRA. The transfer source is specified by MARA, and the transfer destination by MARB.

Table 5-10 summarizes register functions in normal mode.

**Table 5-10 Register Functions in Normal Mode**

Register	Function	Initial Setting	Operation
23 	Source address register	Start address of transfer source	Incremented/decremented every transfer, or fixed
23 	Destination address register	Start address of transfer destination	Incremented/decremented every transfer, or fixed
15 	Transfer counter	Number of transfers	Decremented every transfer; transfer ends when count reaches H'0000

### Legend

- MARA : Memory address register A
- MARB : Memory address register B
- ETCRA : Execute transfer count register A

MARA and MARB specify the start addresses of the transfer source and transfer destination, respectively, as 24 bits. MAR can be incremented or decremented by 1 or 2 each time a byte or word is transferred, or can be fixed.

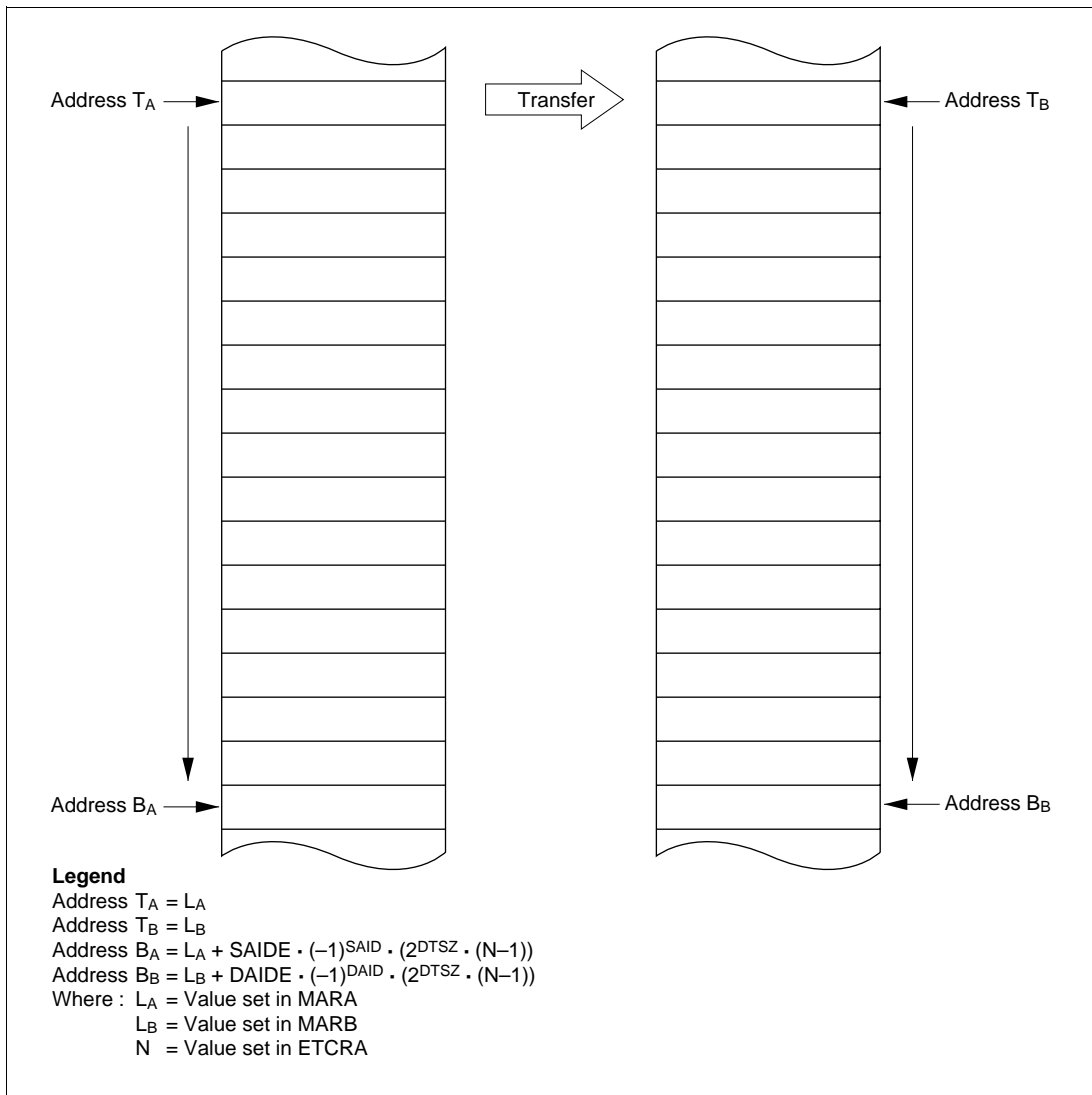
Incrementing, decrementing, or holding a fixed value can be set separately for MARA and MARB.

The number of transfers is specified by ETCRA as 16 bits. ETCRA is decremented each time a transfer is performed, and when its value reaches H'0000 the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this time, an interrupt request is sent to the CPU or DTC.

The maximum number of transfers, when H'0000 is set in ETCRA, is 65,536.



Figure 5-11 illustrates operation in normal mode.



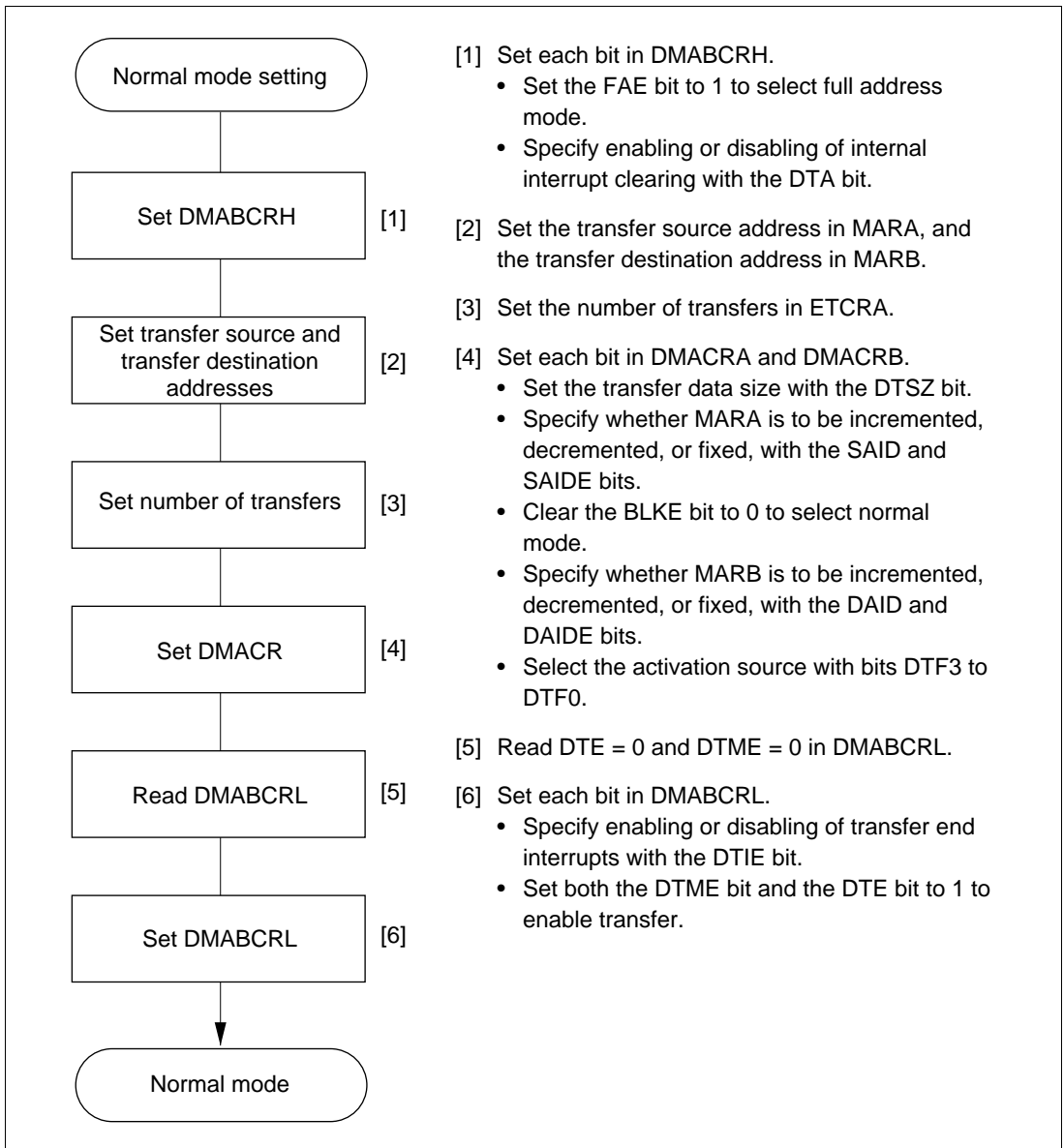
**Figure 5-11 Operation in Normal Mode**

Transfer requests (activation sources) are external requests and auto-requests.

With auto-request, the DMAC is only activated by register setting, and the specified number of transfers are performed automatically. With auto-request, cycle steal mode or burst mode can be selected. In cycle steal mode, the bus is released to another bus master each time a transfer is performed. In burst mode, the bus is held continuously until transfer ends.

For setting details, see section 5.3.4, DMA Control Register (DMACR).

Figure 5-12 shows an example of the setting procedure for normal mode.



**Figure 5-12 Example of Normal Mode Setting Procedure**

## 5.5.7 Block Transfer Mode

In block transfer mode, transfer is performed with channels A and B used in combination. Block transfer mode can be specified by setting the FAE bit in DMABCR and the BLKE bit in DMACRA to 1.

In block transfer mode, a transfer of the specified block size is carried out in response to a single transfer request, and this is executed the specified number of times. The transfer source is specified by MARA, and the transfer destination by MARB. Either the transfer source or the transfer destination can be selected as a block area (an area composed of a number of bytes or words).

Table 5-11 summarizes register functions in block transfer mode.

**Table 5-11 Register Functions in Block Transfer Mode**

Register	Function	Initial Setting	Operation
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 2px; margin: 2px 0;"> <span style="float: left;">23</span> <span style="float: right;">0</span> <div style="text-align: center; border: 1px solid black; padding: 2px;">MARA</div> </div>	Source address register	Start address of transfer source	Incremented/decremented every transfer, or fixed
<div style="display: flex; justify-content: space-between;"> <span>23</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 2px; margin: 2px 0;"> <span style="float: left;">23</span> <span style="float: right;">0</span> <div style="text-align: center; border: 1px solid black; padding: 2px;">MARB</div> </div>	Destination address register	Start address of transfer destination	Incremented/decremented every transfer, or fixed
<div style="display: flex; justify-content: space-between;"> <span>7</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 2px; margin: 2px 0;"> <span style="float: left;">7</span> <span style="float: right;">0</span> <div style="text-align: center; border: 1px solid black; padding: 2px;">ETCRAH</div> </div>	Holds block size	Block size	Fixed
<hr style="border-top: 1px dashed black;"/>			
<div style="display: flex; justify-content: space-between;"> <span>7</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 2px; margin: 2px 0;"> <span style="float: left;">7</span> <span style="float: right;">0</span> <div style="text-align: center; border: 1px solid black; padding: 2px;">ETCRAL</div> </div>	Block size counter	Block size	Decrement every transfer; ETCRH value copied when count reaches H'00
<div style="display: flex; justify-content: space-between;"> <span>15</span> <span>0</span> </div> <div style="border: 1px solid black; padding: 2px; margin: 2px 0;"> <span style="float: left;">15</span> <span style="float: right;">0</span> <div style="text-align: center; border: 1px solid black; padding: 2px;">ETCRB</div> </div>	Block transfer counter	Number of block transfers	Decrement every block transfer; transfer ends when count reaches H'0000

### Legend

- MARA : Memory address register A
- MARB : Memory address register B
- ETCRA : Execute transfer count register A
- ETCRB : Execute transfer count register B

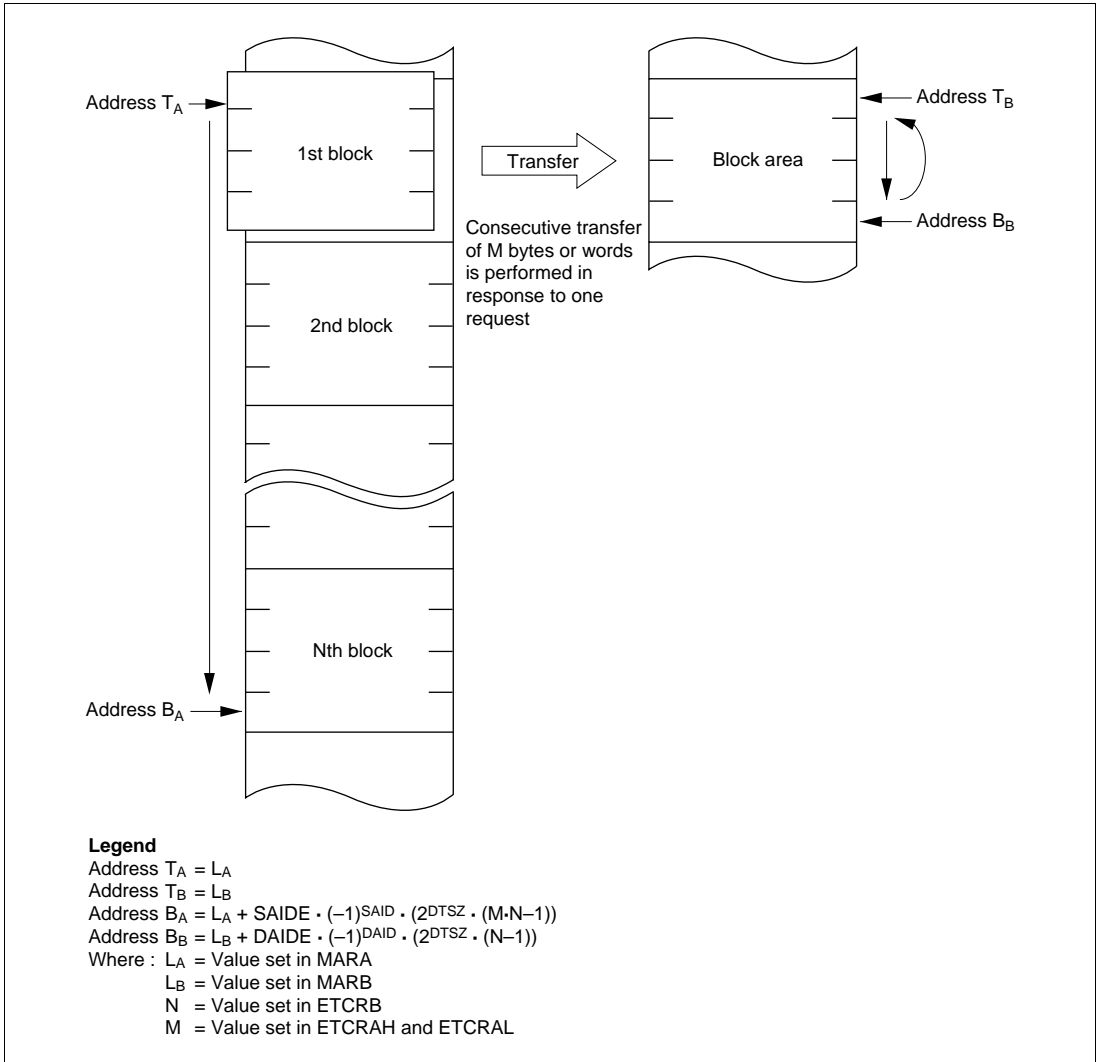
MARA and MARB specify the start addresses of the transfer source and transfer destination, respectively, as 24 bits. MAR can be incremented or decremented by 1 or 2 each time a byte or word is transferred, or can be fixed.

Incrementing, decrementing, or holding a fixed value can be set separately for MARA and MARB.

Whether a block is to be designated for MARA or for MARB is specified by the BLKDIR bit in DMACRA.

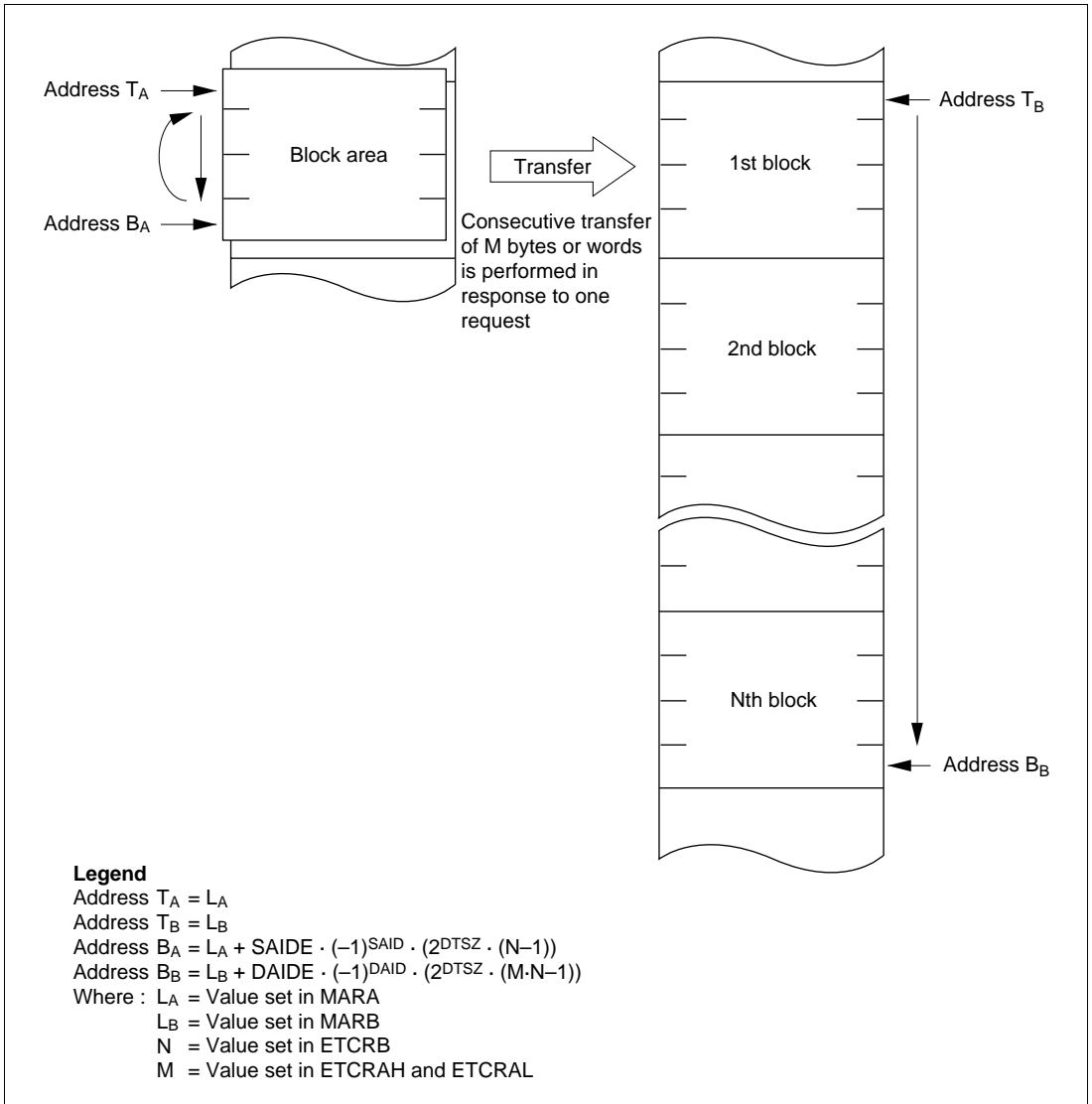
To specify the number of transfers, if M is the size of one block (where M = 1 to 256) and N transfers are to be performed (where N = 1 to 65,536), M is set in both ETCRAH and ETCRAL, and N in ETCRB.

Figure 5-13 illustrates operation in block transfer mode when MARB is designated as a block area.



**Figure 5-13 Operation in Block Transfer Mode (BLKDIR = 0)**

Figure 5-14 illustrates operation in block transfer mode when MARA is designated as a block area.

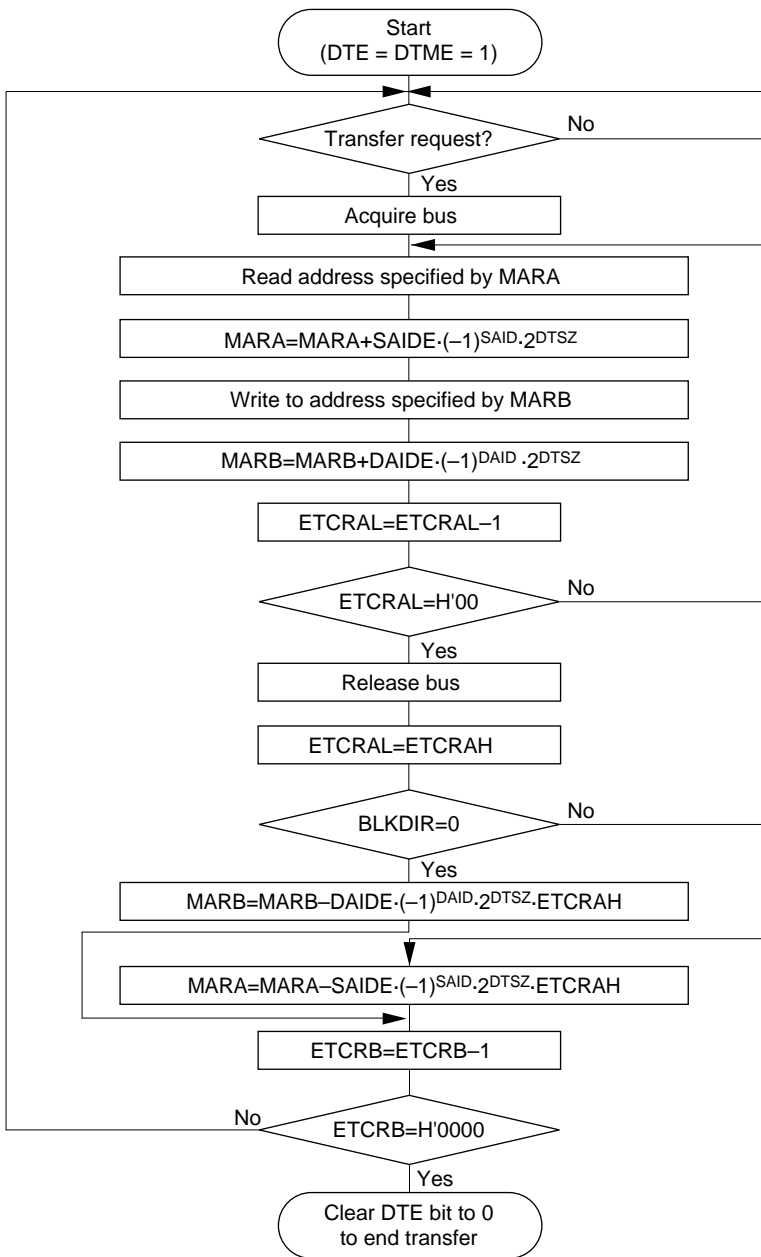


**Figure 5-14 Operation in Block Transfer Mode (BLKDIR = 1)**

ETCRAL is decremented by 1 each time a byte or word transfer is performed. In response to a single transfer request, burst transfer is performed until the value in ETCRAL reaches H'00. ETCRAL is then loaded with the value in ETCRAH. At this time, the value in the MAR register for which a block designation has been given by the BLKDIR bit in DMACRA is restored in accordance with the DTSZ, SAID/DAID, and SAIDE/DAIDE bits in DMACR.

ETCRB is decremented by 1 after every block transfer, and when the count reaches H'0000 the DTE bit is cleared and transfer ends. If the DTIE bit is set to 1 at this point, an interrupt request is sent to the CPU or DTC.

Figure 5-15 shows the operation flow in block transfer mode.

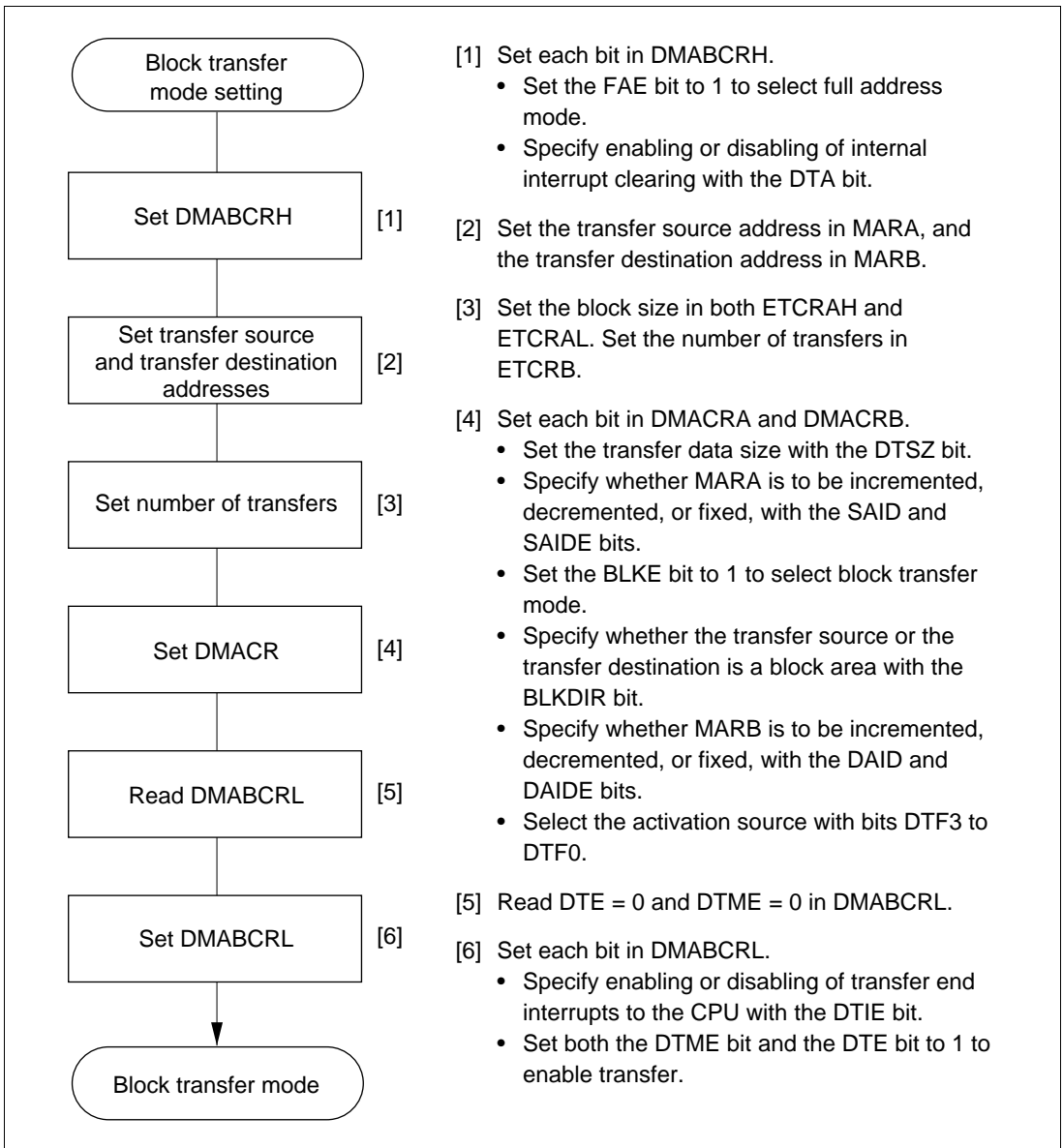


**Figure 5-15 Operation Flow in Block Transfer Mode**

Transfer requests (activation sources) consist of A/D converter conversion end interrupts, external requests, SCI transmission complete and reception complete interrupts, and TPU channel 0 to 5 compare match/input capture A interrupts.

For details, see section 5.3.4, DMA Control Register (DMACR).

Figure 5-16 shows an example of the setting procedure for block transfer mode.



**Figure 5-16 Example of Block Transfer Mode Setting Procedure**



## 5.5.8 DMAC Activation Sources

DMAC activation sources consist of internal interrupts, external requests, and auto-requests. The activation sources that can be specified depend on the transfer mode and the channel, as shown in table 5-12.

**Table 5-12 DMAC Activation Sources**

Activation Source		Short Address Mode		Full Address Mode	
		Channels 0A and 1A	Channels 0B and 1B	Normal Mode	Block Transfer Mode
Internal Interrupts	ADI	○	○	X	○
	TXI0	○	○	X	○
	RXI0	○	○	X	○
	TXI1	○	○	X	○
	RXI1	○	○	X	○
	TGI0A	○	○	X	○
	TGI1A	○	○	X	○
	TGI2A	○	○	X	○
	TGI3A	○	○	X	○
	TGI4A	○	○	X	○
	TGI5A	○	○	X	○
External Requests	DREQ pin falling edge input	X	○	○	○
	DREQ pin low-level input	X	○	○	○
Auto-request		X	X	○	X

### Legend

- : Can be specified
- X : Cannot be specified

**Activation by Internal Interrupt:** An interrupt request selected as a DMAC activation source can be sent simultaneously to the CPU and DTC. For details, see section 3, Interrupt Controller.

With activation by an internal interrupt, the DMAC accepts the request independently of the interrupt controller. Consequently, interrupt controller priority settings are irrelevant.

If the DMAC is activated by a CPU interrupt source or an interrupt source that is not used as a DTC activation source ( $DTA = 1$ ), the interrupt source flag is cleared automatically by the DMA transfer. With ADI, TXI, and RXI interrupts, however, the interrupt source flag is not cleared unless the prescribed register is accessed in a DMA transfer. If the same interrupt is used as an

activation source for more than one channel, the interrupt request flag is cleared when the highest-priority channel is activated first. Transfer requests for other channels are held pending in the DMAC, and activation is carried out in order of priority.

When  $DTE = 0$ , such as after completion of a transfer, a request from the selected activation source is not sent to the DMAC, regardless of the DTA bit. In this case, the relevant interrupt request is sent to the CPU or DTC.

In case of overlap with a CPU interrupt source or DTC activation source ( $DTA = 0$ ), the interrupt request flag is not cleared by the DMAC.

**Activation by External Request:** If an external request ( $\overline{DREQ}$  pin) is specified as an activation source, the relevant port should be set to input mode in advance.

Level sensing or edge sensing can be used for external requests.

External request operation in normal mode (short address mode or full address mode) is described below.

When edge sensing is selected, a 1-byte or 1-word transfer is executed each time a high-to-low transition is detected on the  $\overline{DREQ}$  pin. The next transfer may not be performed if the next edge is input before transfer is completed.

When level sensing is selected, the DMAC stands by for a transfer request while the  $\overline{DREQ}$  pin is held high. While the  $\overline{DREQ}$  pin is held low, transfers continue in succession, with the bus being released each time a byte or word is transferred. If the  $\overline{DREQ}$  pin goes high in the middle of a transfer, the transfer is interrupted and the DMAC stands by for a transfer request.

**Activation by Auto-Request:** Auto-request activation is performed by register setting only, and transfer continues to the end.

With auto-request activation, cycle steal mode or burst mode can be selected.

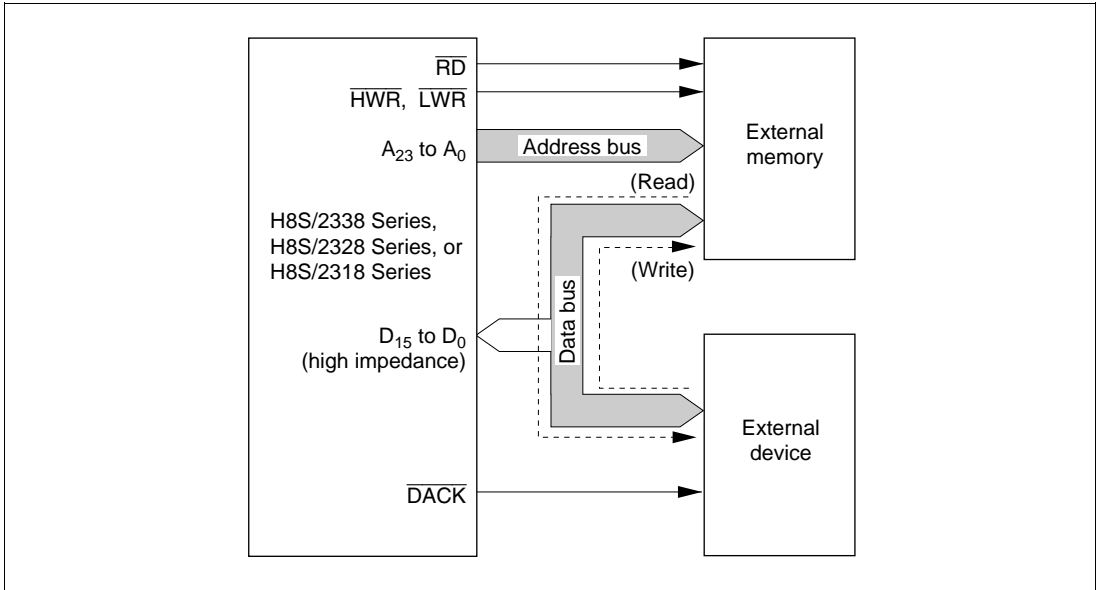
In cycle steal mode, the DMAC releases the bus to another bus master each time a byte or word is transferred. DMA and CPU cycles usually alternate.

In burst mode, the DMAC keeps possession of the bus until the end of the transfer, and transfer is performed continuously.

**Single Address Mode:** The DMAC can operate in dual address mode in which read cycles and write cycles are separate cycles, or single address mode in which read and write cycles are executed in parallel.

In dual address mode, transfer is performed with the source address and destination address specified separately.

In single address mode, on the other hand, transfer is performed between external space in which either the transfer source or the transfer destination is specified by an address, and an external device for which selection is performed by means of the  $\overline{\text{DACK}}$  strobe, without regard to the address. Figure 5-16 shows the data bus in single address mode.



**Figure 5-17 Data Bus in Single Address Mode**

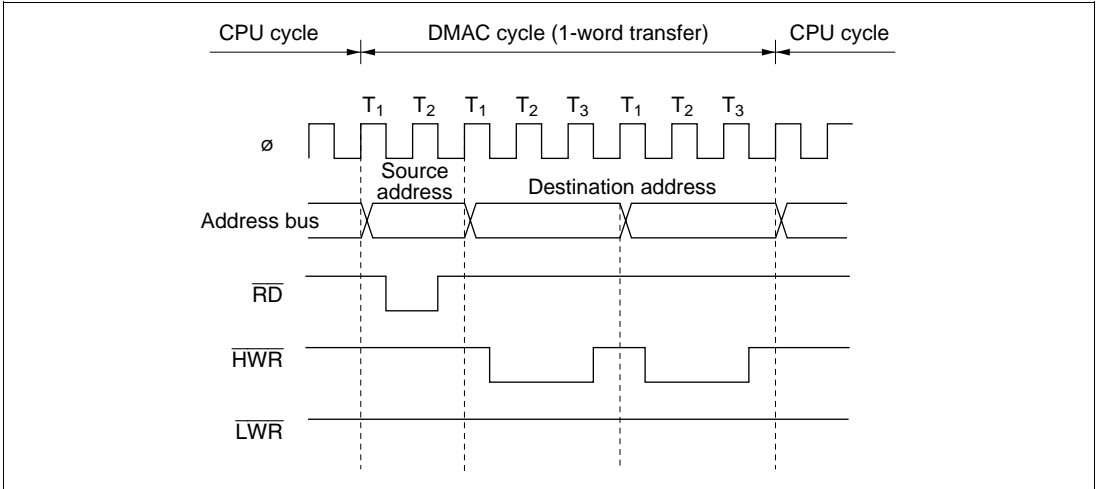
When using the DMAC for single address mode reading, transfer is performed from external memory to the external device, and the  $\overline{\text{DACK}}$  pin functions as a write strobe for the external device. When using the DMAC for single address mode writing, transfer is performed from the external device to external memory, and the  $\overline{\text{DACK}}$  pin functions as a read strobe for the external device. Since there is no directional control for the external device, one or other of the above single directions should be used.

Bus cycles in single address mode are in accordance with the settings of the bus controller for the external memory area. On the external device side,  $\overline{\text{DACK}}$  is output in synchronization with the address strobe. For details of bus cycles, see section 5.5.11, DMAC Bus Cycles (Single Address Mode).

Do not specify internal space for transfer addresses in single address mode.

### 5.5.9 Basic DMAC Bus Cycles

An example of the basic DMAC bus cycle timing is shown in figure 5-18. In this example, word-size transfer is performed from 16-bit, 2-state access space to 8-bit, 3-state access space. When the bus is transferred from the CPU to the DMAC, a source address read and destination address write are performed. The bus is not released in response to another bus request, etc., between these read and write operations. As with CPU cycles, DMA cycles conform to the bus controller settings.

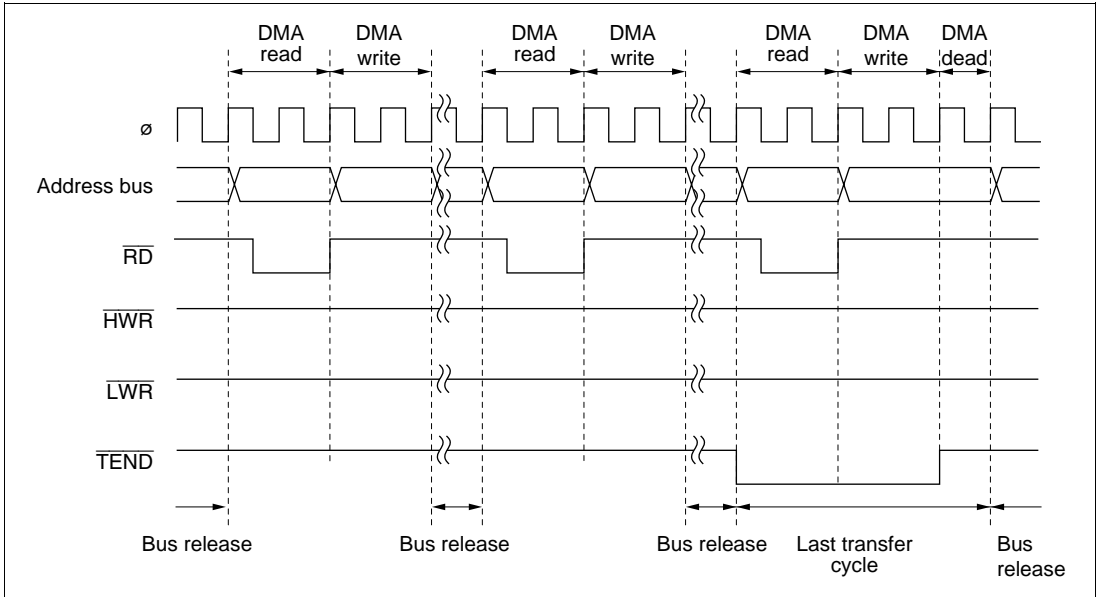


**Figure 5-18 Example of DMA Transfer Bus Timing**

The address is not output to the external address bus in an access to on-chip memory or an internal I/O register.

## 5.5.10 DMAC Bus Cycles (Dual Address Mode)

**Short Address Mode:** Figure 5-19 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and byte-size short address mode transfer (sequential/idle/repeat mode) is performed from external 8-bit, 2-state access space to internal I/O space.



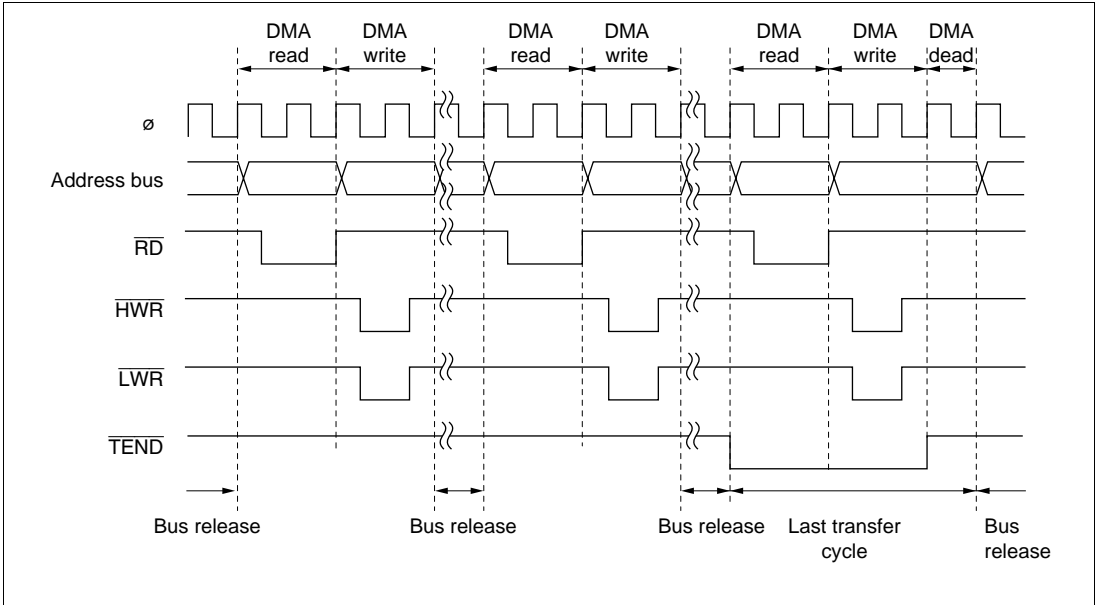
**Figure 5-19 Example of Short Address Mode Transfer**

A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released one or more bus cycles are executed by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

In repeat mode, when  $\overline{\text{TEND}}$  output is enabled,  $\overline{\text{TEND}}$  output goes low in the transfer cycle in which the transfer counter reaches 0.

**Full Address Mode (Cycle Steal Mode):** Figure 5-20 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and word-size full address mode transfer (cycle steal mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

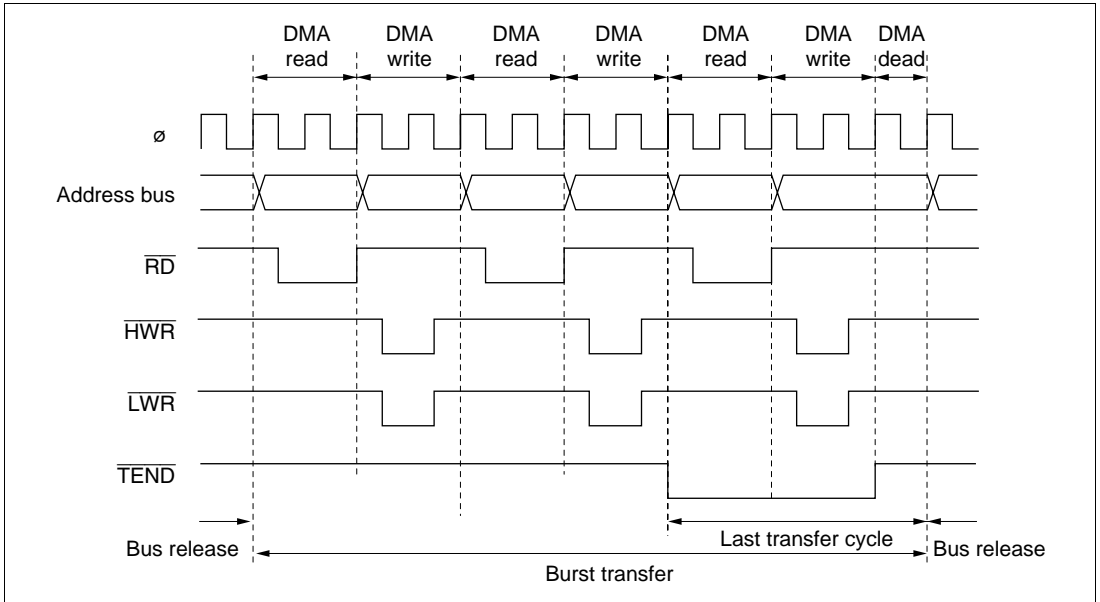


**Figure 5-20 Example of Full Address Mode (Cycle Steal) Transfer**

A one-byte or one-word transfer is performed, and after the transfer the bus is released. While the bus is released one bus cycle is executed by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

**Full Address Mode (Burst Mode):** Figure 5-21 shows a transfer example in which  $\overline{TEND}$  output is enabled and word-size full address mode transfer (burst mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.



**Figure 5-21 Example of Full Address Mode (Burst Mode) Transfer**

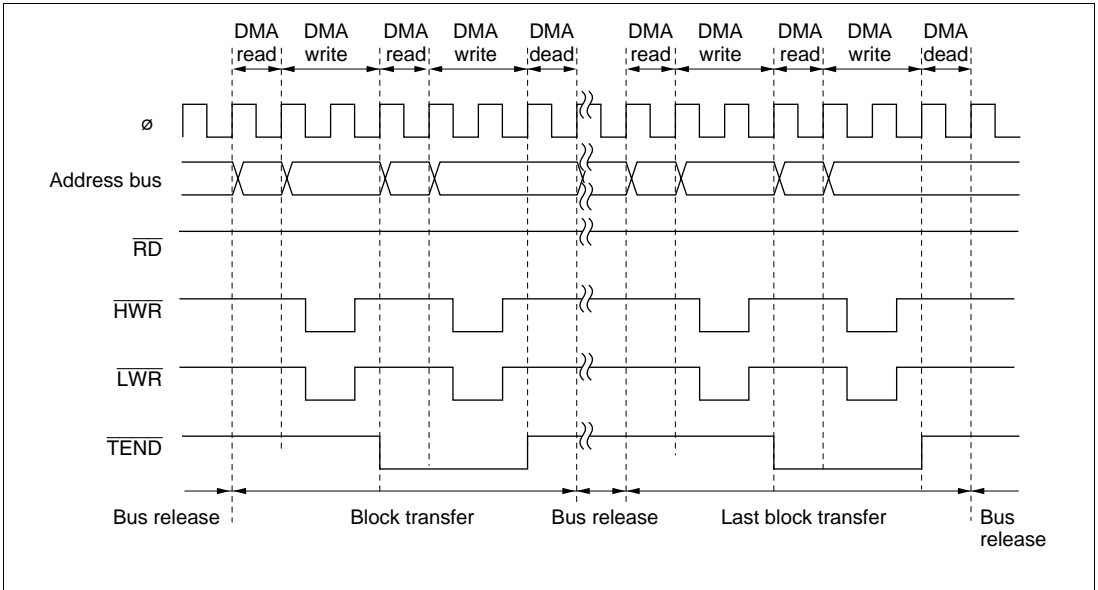
In burst mode, one-byte or one-word transfers are executed consecutively until transfer ends.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

If a request from another higher-priority channel is generated after burst transfer starts, that channel has to wait until the burst transfer ends.

If an NMI is generated while a channel designated for burst transfer is in the transfer enabled state, the DTME bit is cleared and the channel is placed in the transfer disabled state. If burst transfer has already been activated inside the DMAC, the bus is released on completion of a one-byte or one-word transfer within the burst transfer, and burst transfer is suspended. If the last transfer cycle of the burst transfer has already been activated inside the DMAC, execution continues to the end of the transfer even if the DTME bit is cleared.

**Full Address Mode (Block Transfer Mode):** Figure 5-22 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and word-size full address mode transfer (block transfer mode) is performed from internal 16-bit, 1-state access space to external 16-bit, 2-state access space.



**Figure 5-22 Example of Full Address Mode (Block Transfer Mode) Transfer**

A one-block transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released, one or more bus cycles are executed by the CPU or DTC.

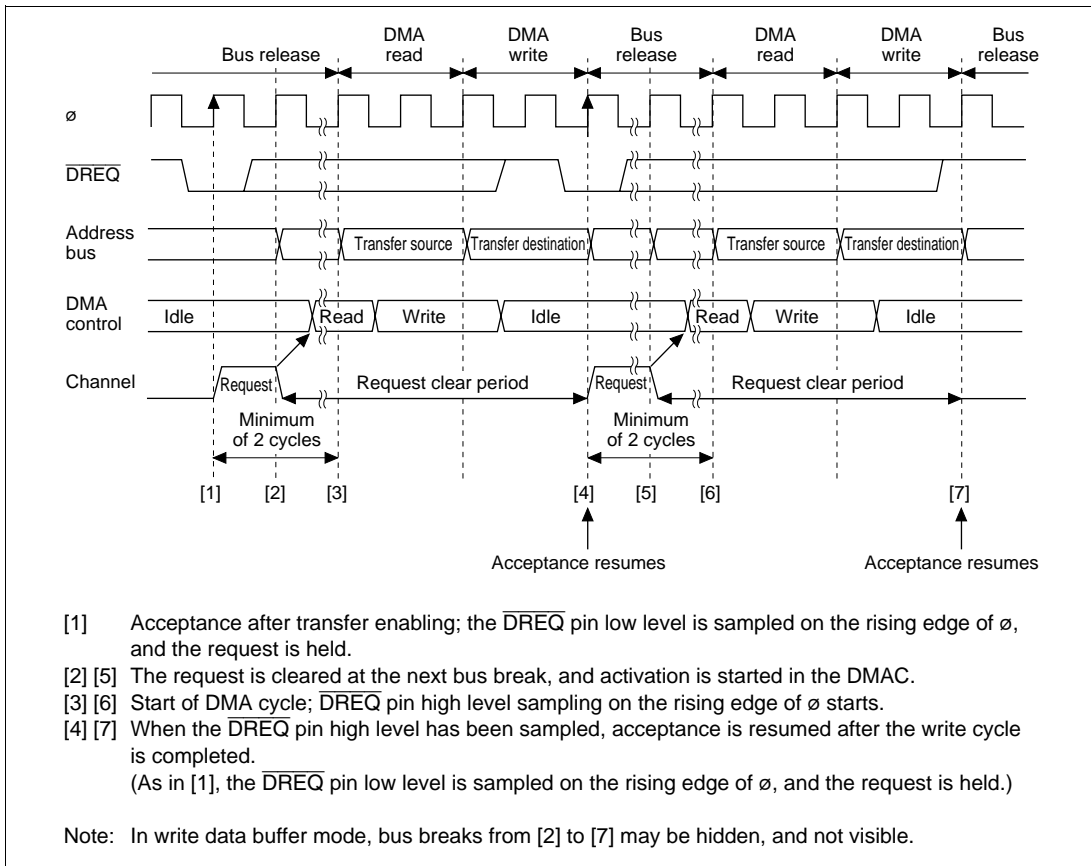
In the transfer end cycle of each block (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

One block is transmitted without interruption. NMI generation does not affect block transfer operation.



**DREQ Pin Falling Edge Activation Timing:** Set the DTA bit for the channel for which the  $\overline{\text{DREQ}}$  pin is selected to 1.

Figure 5-23 shows an example of  $\overline{\text{DREQ}}$  pin falling edge activated normal mode transfer.

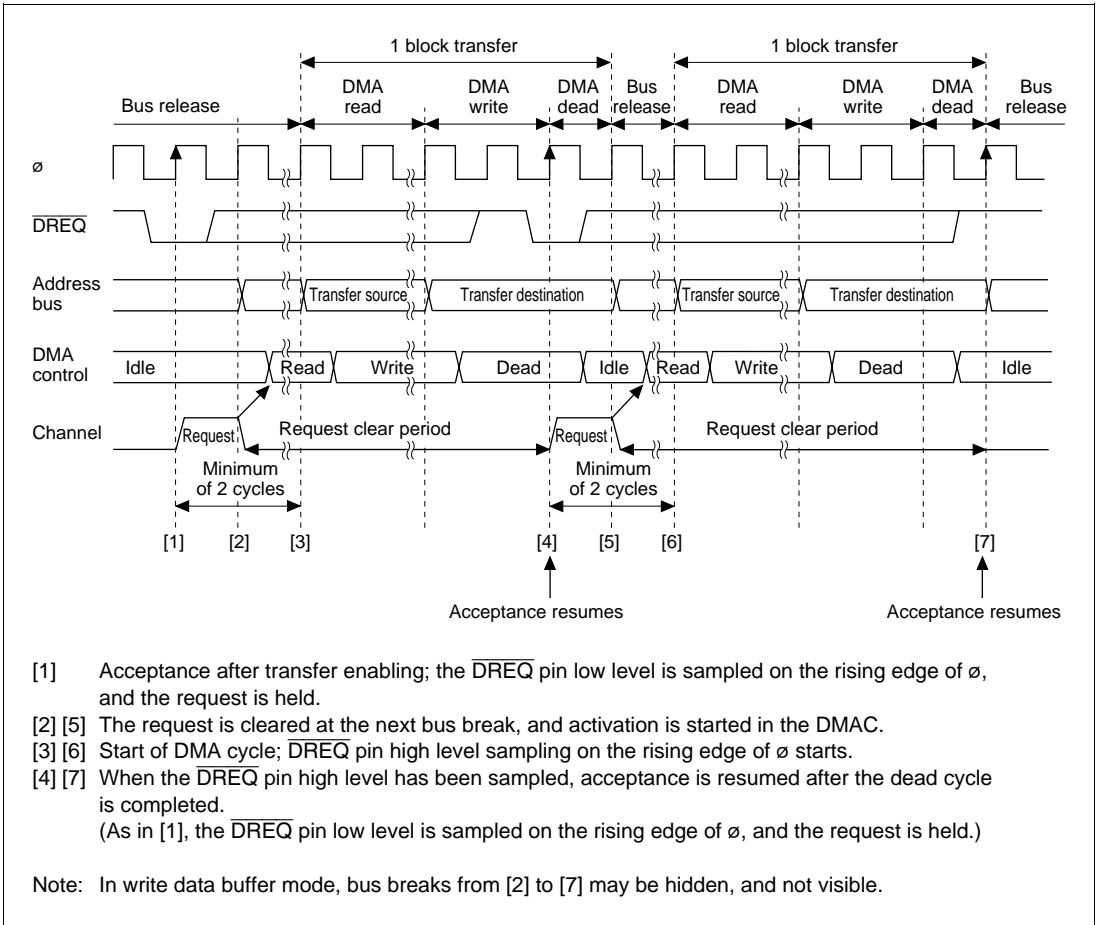


**Figure 5-23 Example of  $\overline{\text{DREQ}}$  Pin Falling Edge Activated Normal Mode Transfer**

$\overline{\text{DREQ}}$  pin sampling is performed every cycle, with the rising edge of the next  $\phi$  cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the  $\overline{\text{DREQ}}$  pin low level is sampled while acceptance by means of the  $\overline{\text{DREQ}}$  pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared, and  $\overline{\text{DREQ}}$  pin high level sampling for edge detection is started. If  $\overline{\text{DREQ}}$  pin high level sampling has been completed by the time the DMA write cycle ends, acceptance resumes after the end of the write cycle,  $\overline{\text{DREQ}}$  pin low level sampling is performed again, and this operation is repeated until the transfer ends.

Figure 5-24 shows an example of  $\overline{\text{DREQ}}$  pin falling edge activated block transfer mode transfer.



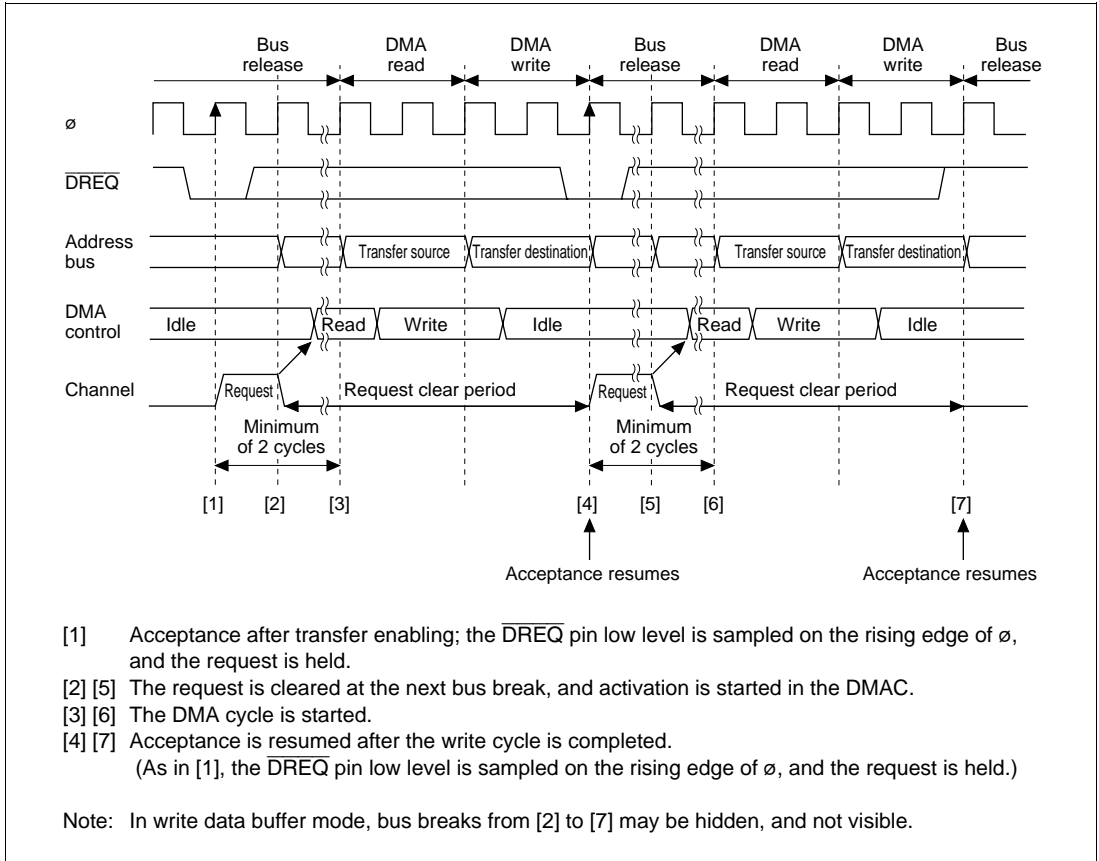
**Figure 5-24 Example of  $\overline{\text{DREQ}}$  Pin Falling Edge Activated Block Transfer Mode Transfer**

$\overline{\text{DREQ}}$  pin sampling is performed every cycle, with the rising edge of the next  $\phi$  cycle after the end of the DMABCRC write cycle for setting the transfer enabled state as the starting point.

When the  $\overline{\text{DREQ}}$  pin low level is sampled while acceptance by means of the  $\overline{\text{DREQ}}$  pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared, and  $\overline{\text{DREQ}}$  pin high level sampling for edge detection is started. If  $\overline{\text{DREQ}}$  pin high level sampling has been completed by the time the DMA dead cycle ends, acceptance resumes after the end of the dead cycle,  $\overline{\text{DREQ}}$  pin low level sampling is performed again, and this operation is repeated until the transfer ends.

**$\overline{\text{DREQ}}$  Level Activation Timing (Normal Mode):** Set the DTA bit for the channel for which the  $\overline{\text{DREQ}}$  pin is selected to 1.

Figure 5-25 shows an example of  $\overline{\text{DREQ}}$  level activated normal mode transfer.

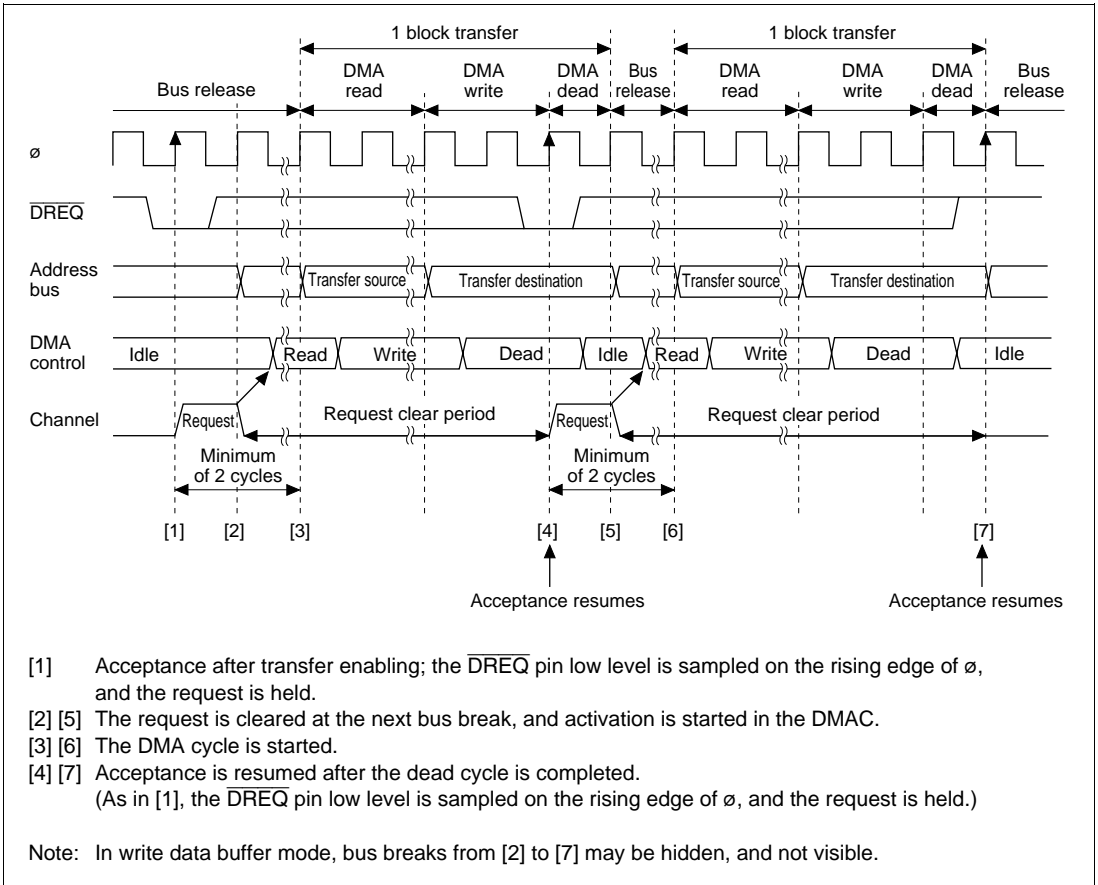


**Figure 5-25 Example of  $\overline{\text{DREQ}}$  Level Activated Normal Mode Transfer**

$\overline{\text{DREQ}}$  pin sampling is performed every cycle, with the rising edge of the next  $\phi$  cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the  $\overline{\text{DREQ}}$  pin low level is sampled while acceptance by means of the  $\overline{\text{DREQ}}$  pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. After the end of the write cycle, acceptance resumes,  $\overline{\text{DREQ}}$  pin low level sampling is performed again, and this operation is repeated until the transfer ends.

Figure 5-26 shows an example of  $\overline{\text{DREQ}}$  level activated block transfer mode transfer.



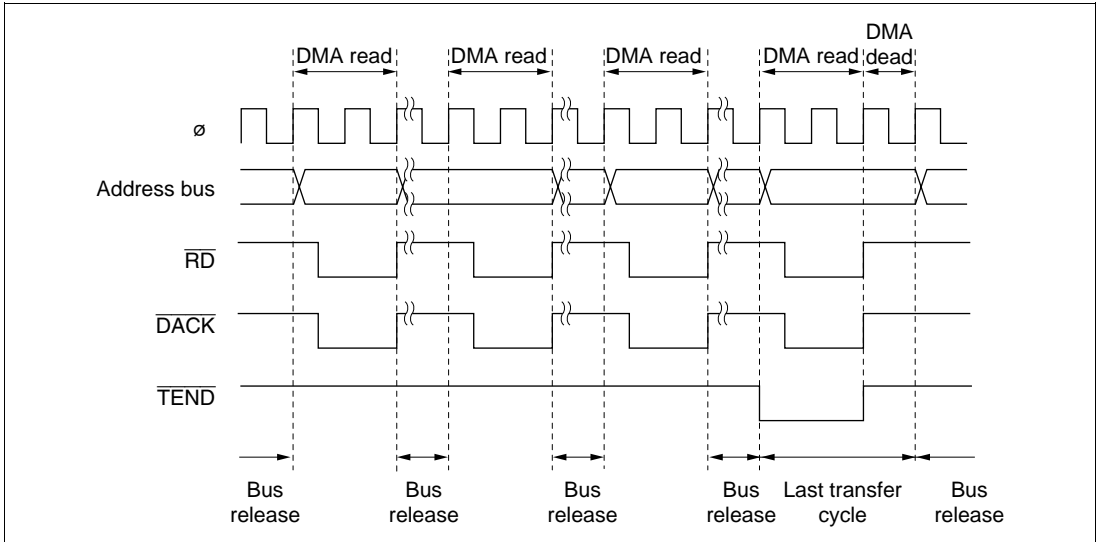
**Figure 5-26 Example of  $\overline{\text{DREQ}}$  Level Activated Block Transfer Mode Transfer**

$\overline{\text{DREQ}}$  pin sampling is performed every cycle, with the rising edge of the next  $\phi$  cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the  $\overline{\text{DREQ}}$  pin low level is sampled while acceptance by means of the  $\overline{\text{DREQ}}$  pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. After the end of the dead cycle, acceptance resumes,  $\overline{\text{DREQ}}$  pin low level sampling is performed again, and this operation is repeated until the transfer ends.

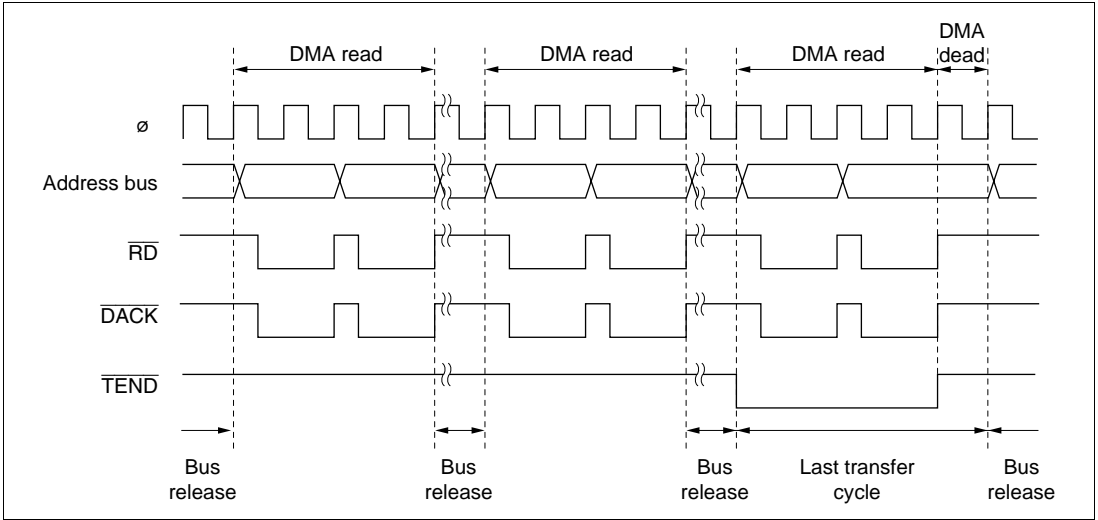
### 5.5.11 DMAC Bus Cycles (Single Address Mode)

**Single Address Mode (Read):** Figure 5-27 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and byte-size single address mode transfer (read) is performed from external 8-bit, 2-state access space to an external device.



**Figure 5-27 Example of Single Address Mode (Byte Read) Transfer**

Figure 5-28 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and word-size single address mode transfer (read) is performed from external 8-bit, 2-state access space to an external device.

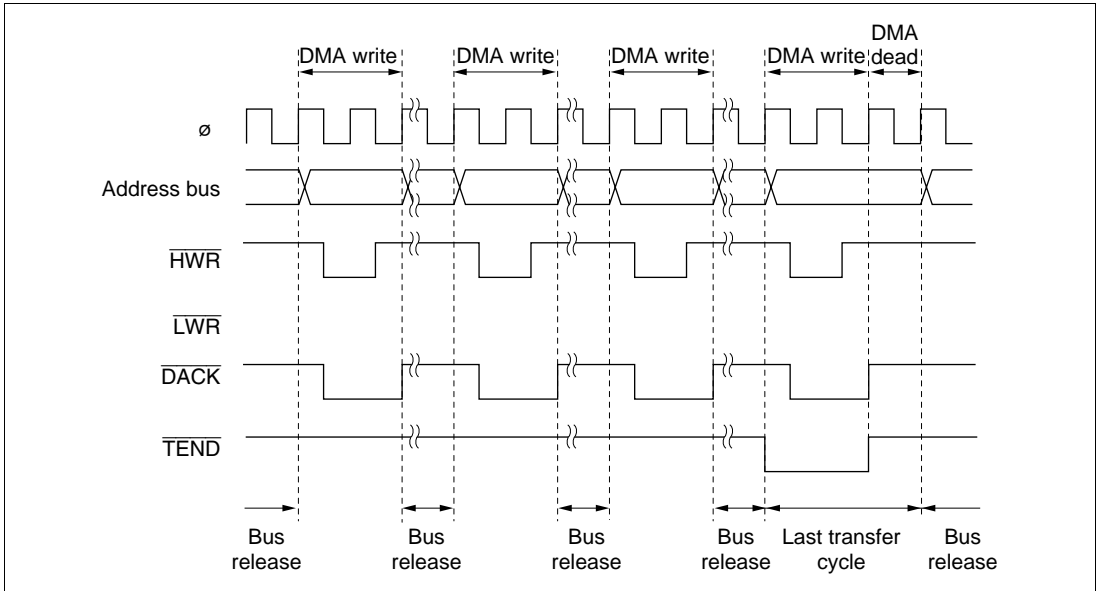


**Figure 5-28 Example of Single Address Mode (Word Read) Transfer**

A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released, one or more bus cycles are executed by the CPU or DTC.

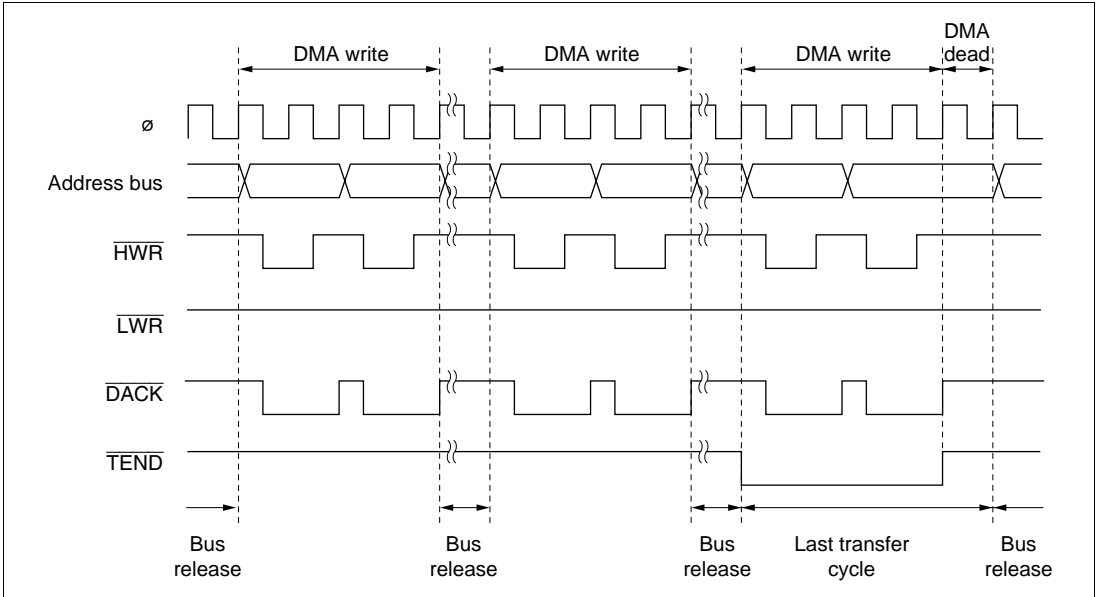
In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

**Single Address Mode (Write):** Figure 5-29 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and byte-size single address mode transfer (write) is performed from an external device to external 8-bit, 2-state access space.



**Figure 5-29 Example of Single Address Mode (Byte Write) Transfer**

Figure 5-30 shows a transfer example in which  $\overline{\text{TEND}}$  output is enabled and word-size single address mode transfer (write) is performed from an external device to external 8-bit, 2-state access space.



**Figure 5-30 Example of Single Address Mode (Word Write) Transfer**

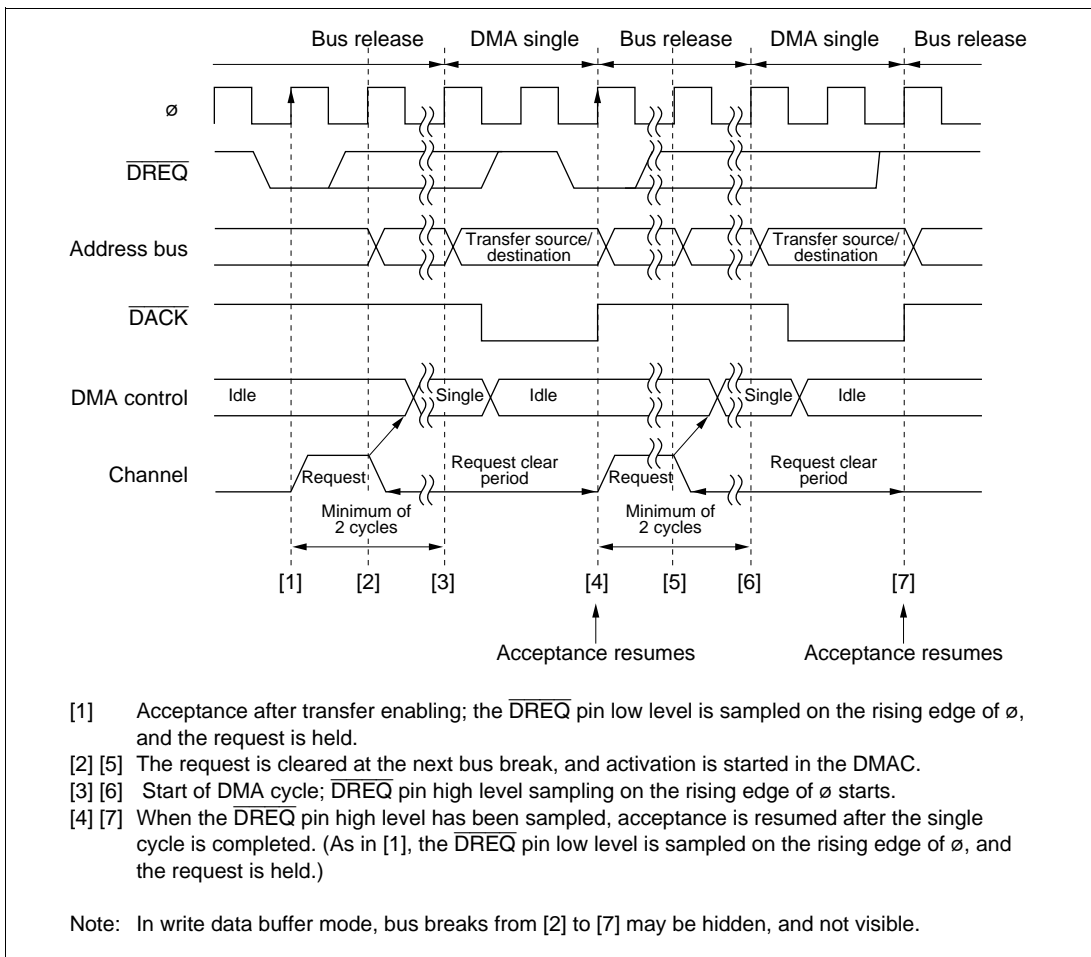
A one-byte or one-word transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released one or more bus cycles are executed by the CPU or DTC.

In the transfer end cycle (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.



**$\overline{\text{DREQ}}$  Pin Falling Edge Activation Timing:** Set the DTA bit for the channel for which the  $\overline{\text{DREQ}}$  pin is selected to 1.

Figure 5-31 shows an example of  $\overline{\text{DREQ}}$  pin falling edge activated single address mode transfer.



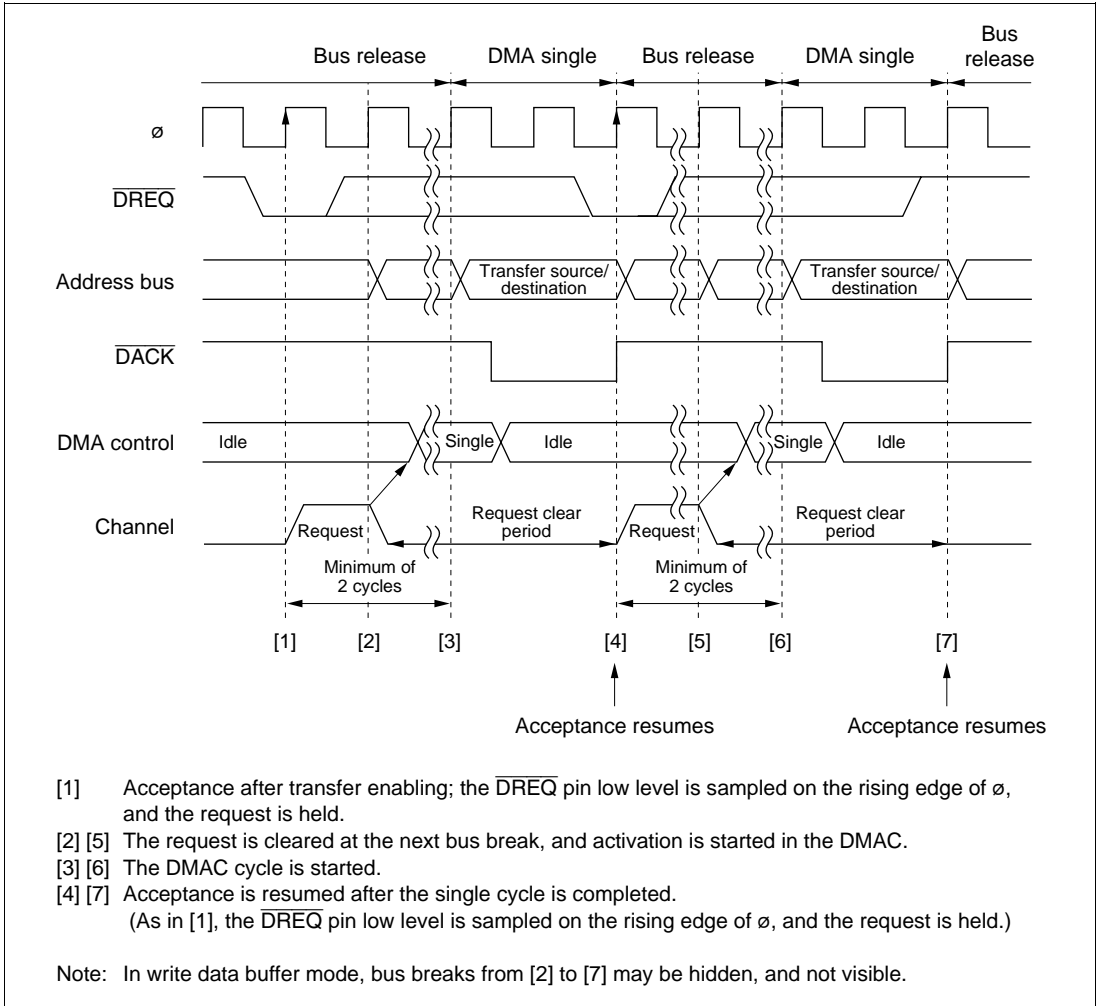
**Figure 5-31 Example of  $\overline{\text{DREQ}}$  Pin Falling Edge Activated Single Address Mode Transfer**

$\overline{\text{DREQ}}$  pin sampling is performed every cycle, with the rising edge of the next  $\phi$  cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the  $\overline{\text{DREQ}}$  pin low level is sampled while acceptance by means of the  $\overline{\text{DREQ}}$  pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared, and  $\overline{\text{DREQ}}$  pin high level sampling for edge detection is started. If  $\overline{\text{DREQ}}$  pin high level sampling has been completed by the time the DMA single cycle ends, acceptance resumes after the end of the single cycle,  $\overline{\text{DREQ}}$  pin low level sampling is performed again, and this operation is repeated until the transfer ends.

**$\overline{\text{DREQ}}$  Pin Low Level Activation Timing:** Set the DTA bit for the channel for which the  $\overline{\text{DREQ}}$  pin is selected to 1.

Figure 5-32 shows an example of  $\overline{\text{DREQ}}$  pin low level activated single address mode transfer.



**Figure 5-32 Example of  $\overline{\text{DREQ}}$  Pin Low Level Activated Single Address Mode Transfer**

$\overline{\text{DREQ}}$  pin sampling is performed every cycle, with the rising edge of the next  $\phi$  cycle after the end of the DMABCR write cycle for setting the transfer enabled state as the starting point.

When the  $\overline{\text{DREQ}}$  pin low level is sampled while acceptance by means of the  $\overline{\text{DREQ}}$  pin is possible, the request is held in the DMAC. Then, when activation is initiated in the DMAC, the request is cleared. After the end of the single cycle, acceptance resumes,  $\overline{\text{DREQ}}$  pin low level sampling is performed again, and this operation is repeated until the transfer ends.

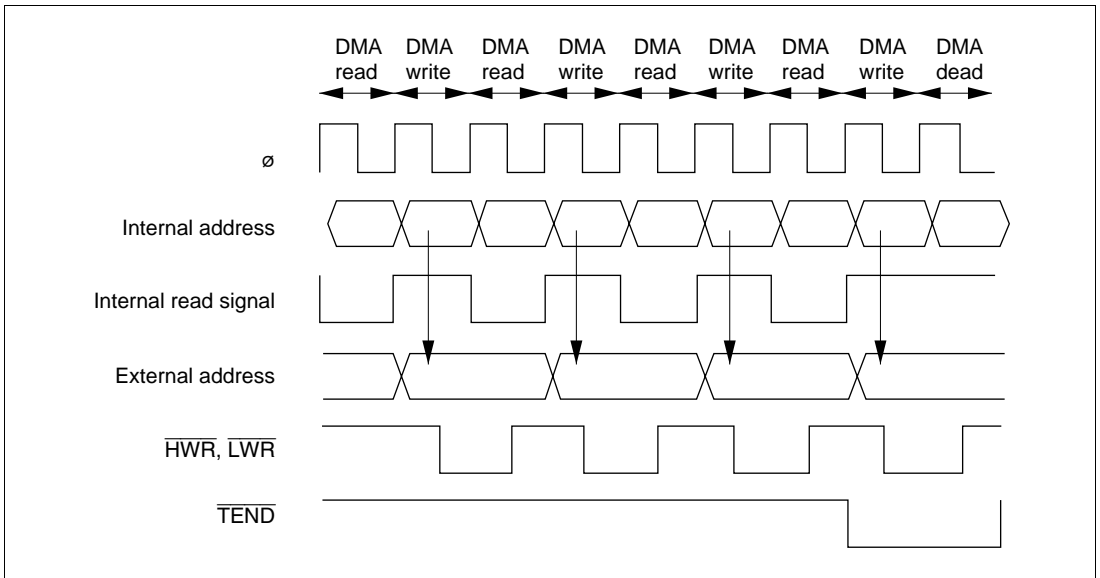
### 5.5.12 Write Data Buffer Function

DMAC internal-to-external dual address transfers and single address transfers can be executed at high speed using the write data buffer function, enabling system throughput to be improved.

When the WDBE bit of BCRL in the bus controller is set to 1, enabling the write data buffer function, dual address transfer external write cycles or single address transfers and internal accesses (on-chip memory or internal I/O registers) are executed in parallel. Internal accesses are independent of the bus master, and DMAC dead cycles are regarded as internal accesses.

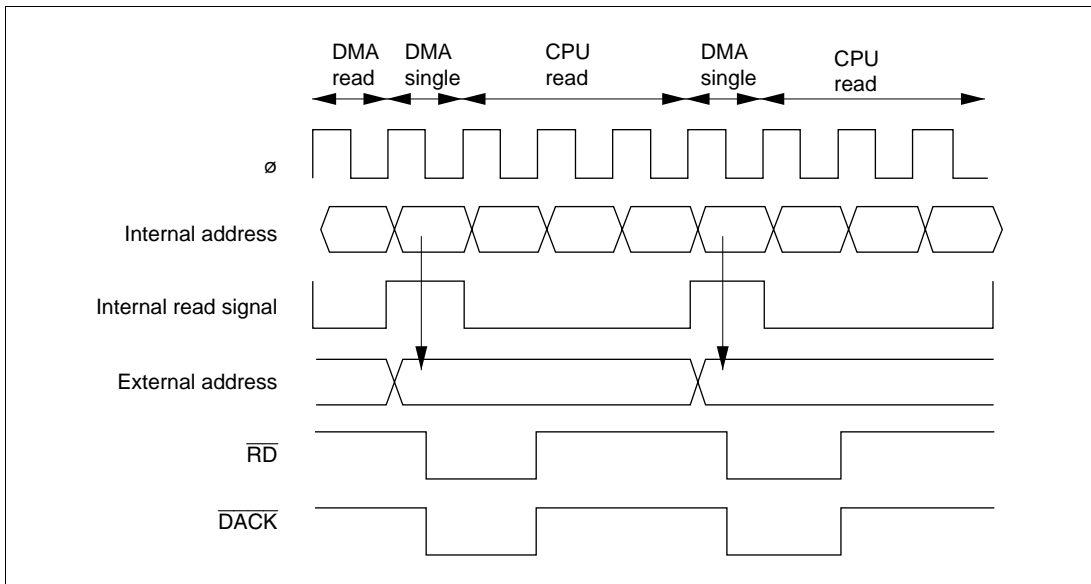
A low level can always be output from the  $\overline{\text{TEND}}$  pin if the bus cycle in which a low level is to be output is an external bus cycle. However, a low level is not output from the  $\overline{\text{TEND}}$  pin if the bus cycle in which a low level is to be output from the  $\overline{\text{TEND}}$  pin is an internal bus cycle, and an external write cycle is executed in parallel with this cycle.

Figure 5-33 shows an example of burst mode transfer from on-chip RAM to external memory using the write data buffer function.



**Figure 5-33 Example of Dual Address Transfer Using Write Data Buffer Function**

Figure 5-34 shows an example of single address transfer using the write data buffer function. In this example, the CPU program area is in on-chip memory.



**Figure 5-34 Example of Single Address Transfer Using Write Data Buffer Function**

When the write data buffer function is activated, the DMAC recognizes that the bus cycle concerned has ended, and starts the next operation. Therefore,  $\overline{\text{DREQ}}$  pin sampling is started one state after the start of the DMA write cycle or single address transfer.

### 5.5.13 DMAC Multi-Channel Operation

The DMAC channel priority order is: channel 0 > channel 1, and channel A > channel B. Table 5-13 summarizes the priority order for DMAC channels.

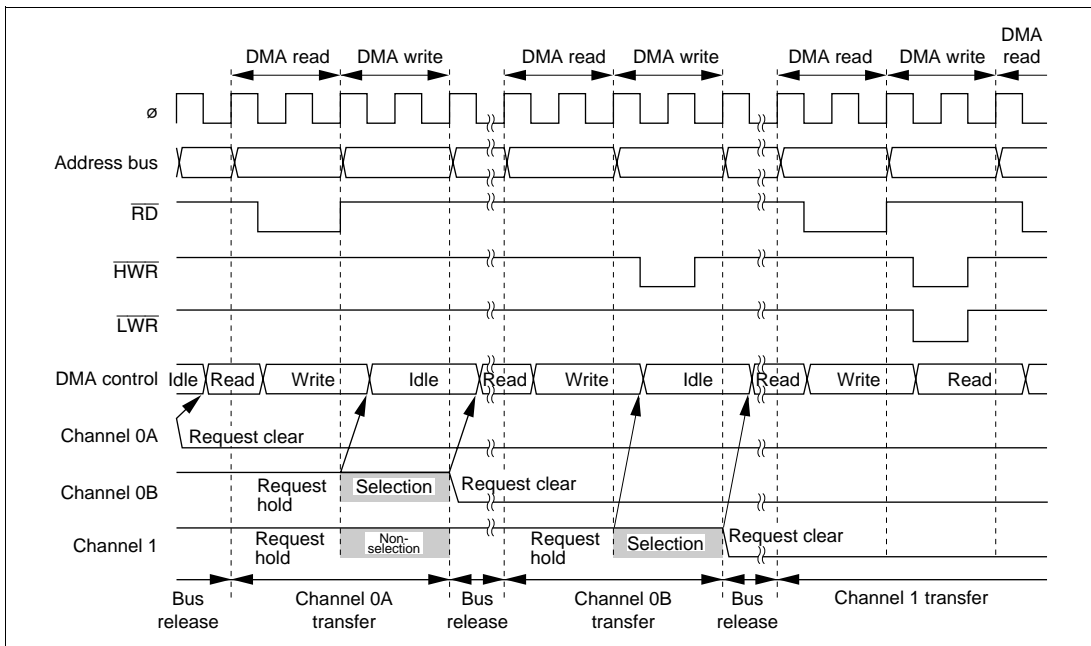
**Table 5-13 DMAC Channel Priority Order**

Short Address Mode	Full Address Mode	Priority
Channel 0A	Channel 0	High
Channel 0B		
Channel 1A	Channel 1	Low
Channel 1B		

If transfer requests are issued simultaneously for more than one channel, or if a transfer request for another channel is issued during a transfer, when the bus is released the DMAC selects the highest-priority channel from among those issuing a request according to the priority order shown in table 5-13.

During burst transfer, or when one block is being transferred in block transfer, the channel will not be changed until the end of the transfer.

Figure 5-35 shows a transfer example in which transfer requests are issued simultaneously for channels 0A, 0B, and 1.



**Figure 5-35 Example of Multi-Channel Transfer**

### 5.5.14 Relation Between the DMAC and External Bus Requests, Refresh Cycles, and the DTC

There can be no break between a DMA cycle read and a DMA cycle write. This means that a refresh cycle, external bus release cycle, or DTC cycle is not generated between the external read and external write in a DMA cycle.

In the case of successive read and write cycles, such as in burst transfer or block transfer, a refresh or external bus released state may be inserted after a write cycle. Since the DTC has a lower priority than the DMAC, the DTC does not operate until the DMAC releases the bus.

When DMA cycle reads or writes are accesses to on-chip memory or internal I/O registers, these DMA cycles can be executed at the same time as refresh cycles or external bus release. However, simultaneous operation may not be possible when a write buffer is used.

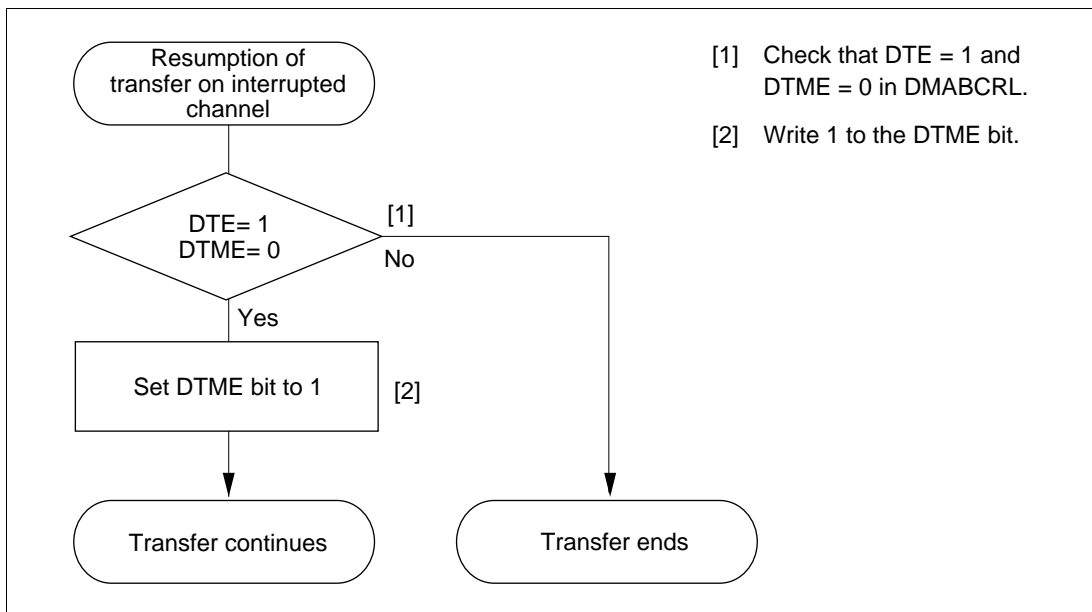
### 5.5.15 NMI Interrupts and DMAC

When an NMI interrupt is requested, burst mode transfer in full address mode is interrupted. An NMI interrupt does not affect the operation of the DMAC in other modes.

In full address mode, transfer is enabled for a channel when both the DTE bit and the DTME bit are set to 1. With burst mode setting, the DTME bit is cleared when an NMI interrupt is requested.

If the DTME bit is cleared during burst mode transfer, the DMAC discontinues transfer on completion of the 1-byte or 1-word transfer in progress, then releases the bus, which passes to the CPU.

The channel on which transfer was interrupted can be restarted by setting the DTME bit to 1 again. Figure 5-36 shows the procedure for continuing transfer when it has been interrupted by an NMI interrupt on a channel designated for burst mode transfer.



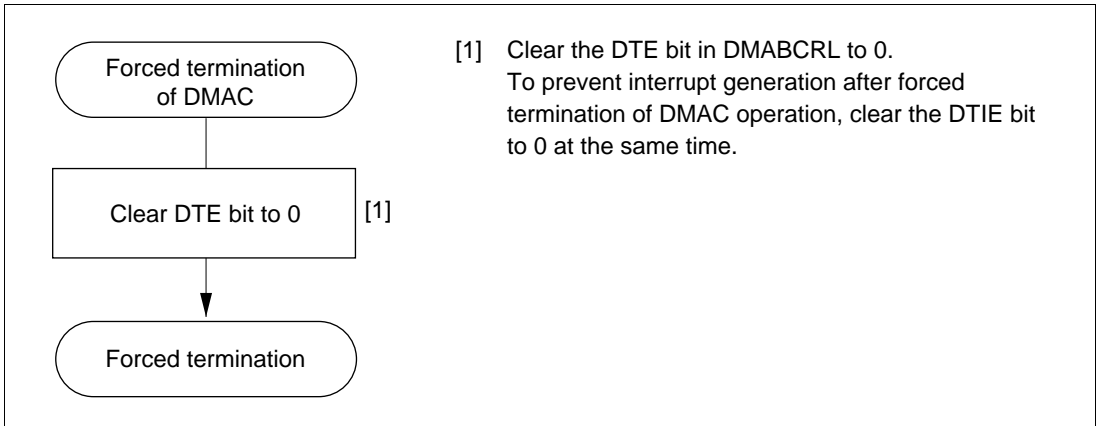
**Figure 5-36 Example of Procedure for Continuing Transfer on Channel Interrupted by NMI Interrupt**

### 5.5.16 Forced Termination of DMAC Operation

If the DTE bit for the channel currently operating is cleared to 0, the DMAC stops on completion of the 1-byte or 1-word transfer in progress. DMAC operation resumes when the DTE bit is set to 1 again.

In full address mode, the same applies to the DTME bit.

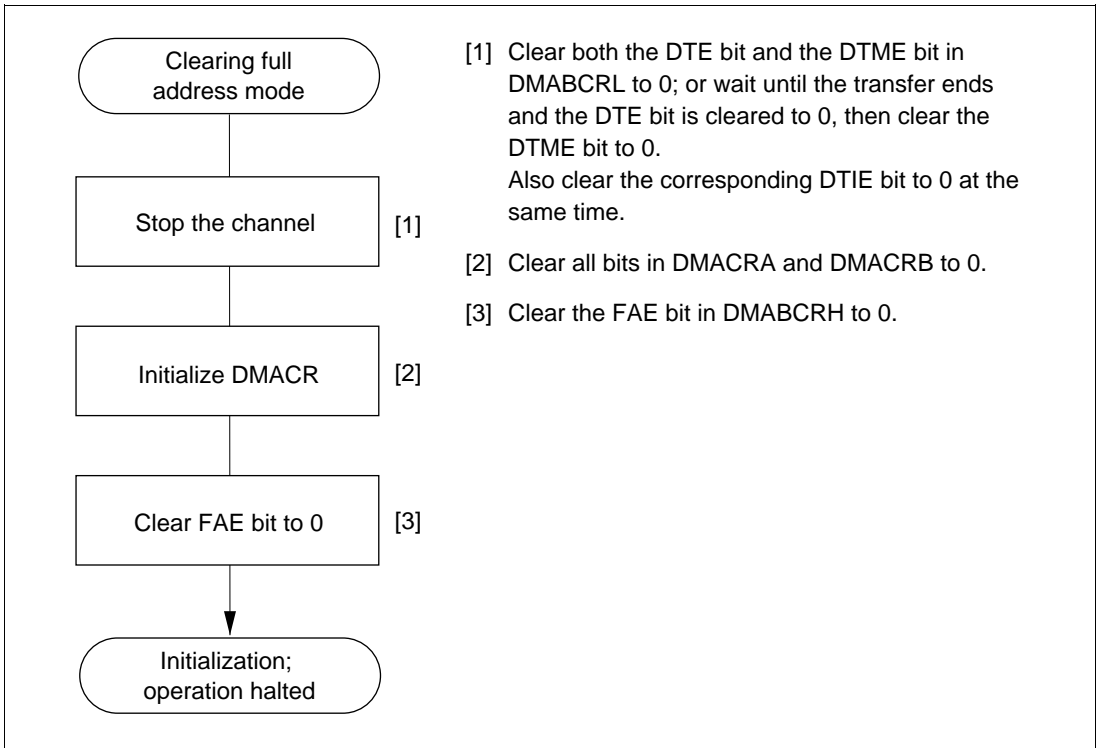
Figure 5-37 shows the procedure for forcibly terminating DMAC operation by software.



**Figure 5-37 Example of Procedure for Forcibly Terminating DMAC Operation**

### 5.5.17 Clearing Full Address Mode

Figure 5-38 shows the procedure for releasing and initializing a channel designated for full address mode. After full address mode has been cleared, the channel can be set to another transfer mode using the appropriate setting procedure.



**Figure 5-38 Example of Procedure for Clearing Full Address Mode**



## 5.6 Interrupts

The sources of interrupts generated by the DMAC are transfer end and transfer break. Table 5-13 shows the interrupt sources and their priority order.

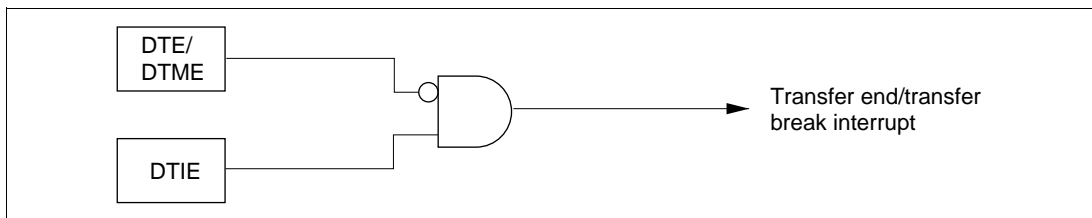
**Table 5-13 Interrupt Source Priority Order**

Interrupt Name	Interrupt Source		Interrupt Priority Order
	Short Address Mode	Full Address Mode	
DEND0A	Interrupt due to end of transfer on channel 0A	Interrupt due to end of transfer on channel 0	<div style="text-align: center;">           High            ↑            ↓            Low         </div>
DEND0B	Interrupt due to end of transfer on channel 0B	Interrupt due to break in transfer on channel 0	
DEND1A	Interrupt due to end of transfer on channel 1A	Interrupt due to end of transfer on channel 1	
DEND1B	Interrupt due to end of transfer on channel 1B	Interrupt due to break in transfer on channel 1	

Enabling or disabling of each interrupt source is set by means of the DTIE bit for the corresponding channel in DMABCR, and interrupts from each source are sent to the interrupt controller independently.

The relative priority of transfer end interrupts on each channel is decided by the interrupt controller, as shown in table 5-13.

Figure 5-39 shows a block diagram of a transfer end/transfer break interrupt. An interrupt is always generated when the DTIE bit is set to 1 while the DTE bit is cleared to 0.



**Figure 5-39 Block Diagram of Transfer End/Transfer Break Interrupt**

In full address mode, a transfer break interrupt is generated when the DTME bit is cleared to 0 while the DTIEB bit is set to 1.

In both short address mode and full address mode, DMABCR should be set so as to prevent the occurrence of a combination that constitutes a condition for interrupt generation during setting.

## 5.7 Usage Notes

**DMAC Register Access during Operation:** Except for forced termination, the operating (including transfer waiting state) channel setting should not be changed. The operating channel setting should only be changed when transfer is disabled.

Also, MAC registers should not be written to in a DMA transfer.

**Module Stop:** When the MSTP15 bit in MSTPCR is set to 1, the DMAC clock stops, and the module stop state is entered. However, 1 cannot be written to the MSTP15 bit if any of the DMAC channels is enabled. This setting should therefore be made when DMAC operation is stopped.

When the DMAC clock stops, DMAC register accesses can no longer be made. Since the following DMAC register settings are valid even in the module stop state, they should be invalidated, if necessary, before a module stop.

- Transfer end/break interrupt (DTE = 0 and DTIE = 1)
- $\overline{\text{TEND}}$  pin enable (TEE = 1)
- $\overline{\text{DACK}}$  pin enable (FAE = 0 and SAE = 1)

**Medium-Speed Mode:** When the DTA bit is 0, internal interrupt signals specified as DMAC transfer sources are edge-detected.

In medium-speed mode, the DMAC operates on a medium-speed clock, while on-chip supporting modules operate on a high-speed clock. Consequently, if the period in which the relevant interrupt source is cleared by the CPU, DTC, or another DMAC channel, and the next interrupt is generated, is less than one state with respect to the DMAC clock (bus master clock), edge detection may not be possible and the interrupt may be ignored.

Also, in medium-speed mode,  $\overline{\text{DREQ}}$  pin sampling is performed on the rising edge of the medium-speed clock.

**Write Data Buffer Function:** When the WDBE bit of BCRL in the bus controller is set to 1, enabling the write data buffer function, dual address transfer external write cycles or single address transfers and internal accesses (on-chip memory or internal I/O registers) are executed in parallel.

- Write Data Buffer Function and DMAC Register Setting

If the setting of a register that controls external accesses is changed during execution of an external access by means of the write data buffer function, the external access may not be performed normally. Registers that control external accesses should only be manipulated when external reads, etc., are used with DMAC operation disabled, and the operation is not performed in parallel with external access.

- Write Data Buffer Function and DMAC Operation Timing

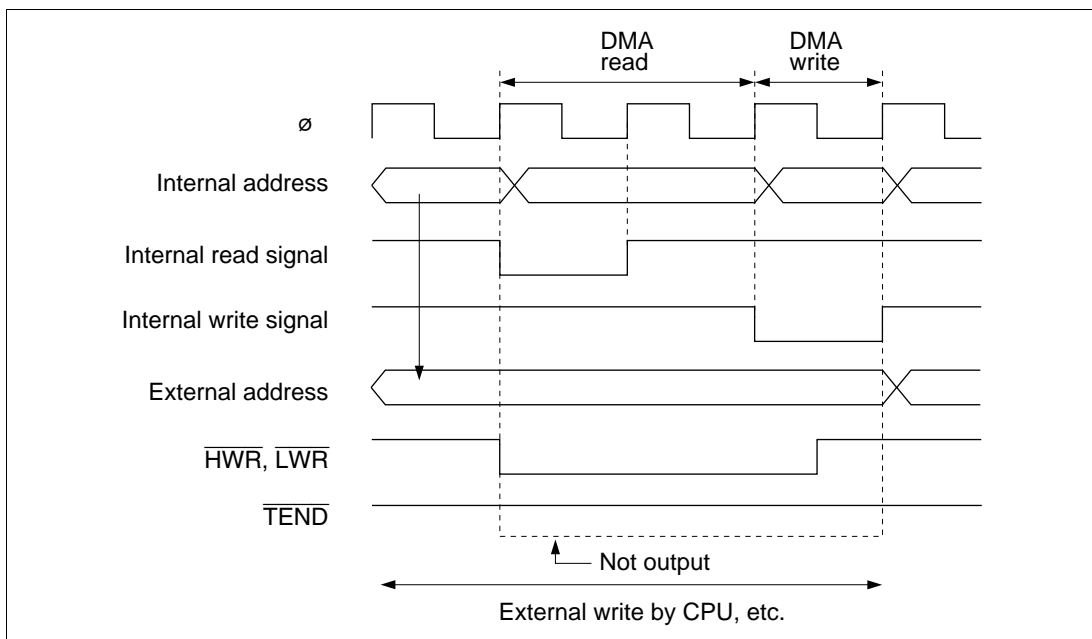
The DMAC can start its next operation during external access using the write data buffer function. Consequently, the  $\overline{\text{DREQ}}$  pin sampling timing,  $\overline{\text{TEND}}$  output timing, etc., are different from the case in which the write data buffer function is disabled. Also, internal bus cycles maybe hidden, and not visible.

- Write Data Buffer Function and  $\overline{\text{TEND}}$  Output

A low level is not output at the  $\overline{\text{TEND}}$  pin if the bus cycle in which a low level is to be output at the  $\overline{\text{TEND}}$  pin is an internal bus cycle, and an external write cycle is executed in parallel with this cycle. Note, for example, that a low level may not be output at the  $\overline{\text{TEND}}$  pin if the write data buffer function is used when data transfer is performed between an internal I/O register and on-chip memory.

If at least one of the DMAC transfer addresses is an external address, a low level is output at the  $\overline{\text{TEND}}$  pin.

Figure 5-40 shows an example in which a low level is not output at the  $\overline{\text{TEND}}$  pin.



**Figure 5-40 Example in Which Low Level is Not Output at  $\overline{\text{TEND}}$  Pin**

**Activation by Falling Edge on  $\overline{\text{DREQ}}$  Pin:**  $\overline{\text{DREQ}}$  pin falling edge detection is performed in synchronization with DMAC internal operations. The operation is as follows:

- [1] Activation request wait state: Waits for detection of a low level on the  $\overline{\text{DREQ}}$  pin, and switches to [2].
- [2] Transfer wait state: Waits for DMAC data transfer to become possible, and switches to [3].
- [3] Activation request disabled state: Waits for detection of a high level on the  $\overline{\text{DREQ}}$  pin, and switches to [1].

After DMAC transfer is enabled, a transition is made to [1]. Thus, initial activation after transfer is enabled is performed on detection of a low level.

**Activation Source Acceptance:** At the start of activation source acceptance, a low level is detected in both  $\overline{\text{DREQ}}$  pin falling edge sensing and low level sensing. Similarly, in the case of an internal interrupt, the interrupt request is detected. Therefore, a request is accepted from an internal interrupt or  $\overline{\text{DREQ}}$  pin low level that occurs before execution of the DMABCRL write to enable transfer.

When the DMAC is activated, take any necessary steps to prevent an internal interrupt or  $\overline{\text{DREQ}}$  pin low level remaining from the end of the previous transfer, etc.

**Internal Interrupt after End of Transfer:** When the DTE bit is cleared to 0 at the end of a transfer or by a forcible termination, the selected internal interrupt request will be sent to the CPU or DTC even if DTA is set to 1.

Also, if internal DMAC activation has already been initiated when operation is forcibly terminated, the transfer is executed but flag clearing is not performed for the selected internal interrupt even if DTA is set to 1.

An internal interrupt request following the end of transfer or a forcible termination should be handled by the CPU as necessary.

**Channel Re-Setting:** To reactivate a number of channels when multiple channels are enabled, use exclusive handling of transfer end interrupts, and perform DMABCR control bit operations exclusively.

Note, in particular, that in cases where multiple interrupts are generated between reading and writing of DMABCR, and a DMABCR operation is performed during new interrupt handling, the DMABCR write data in the original interrupt handling routine will be incorrect, and the write may invalidate the results of the operations by the multiple interrupts. Ensure that overlapping DMABCR operations are not performed by multiple interrupts, and that there is no separation between read and write operations by the use of a bit-manipulation instruction.

Also, when the DTE and DTME bits are cleared by the DMAC or are written with 0, they must first be read while cleared to 0 before the CPU can write a 1 to them.

# Section 6 Data Transfer Controller

## 6.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series include a data transfer controller (DTC). The DTC can be activated for data transfer by an interrupt or software.

### 6.1.1 Features

The features of the DTC are:

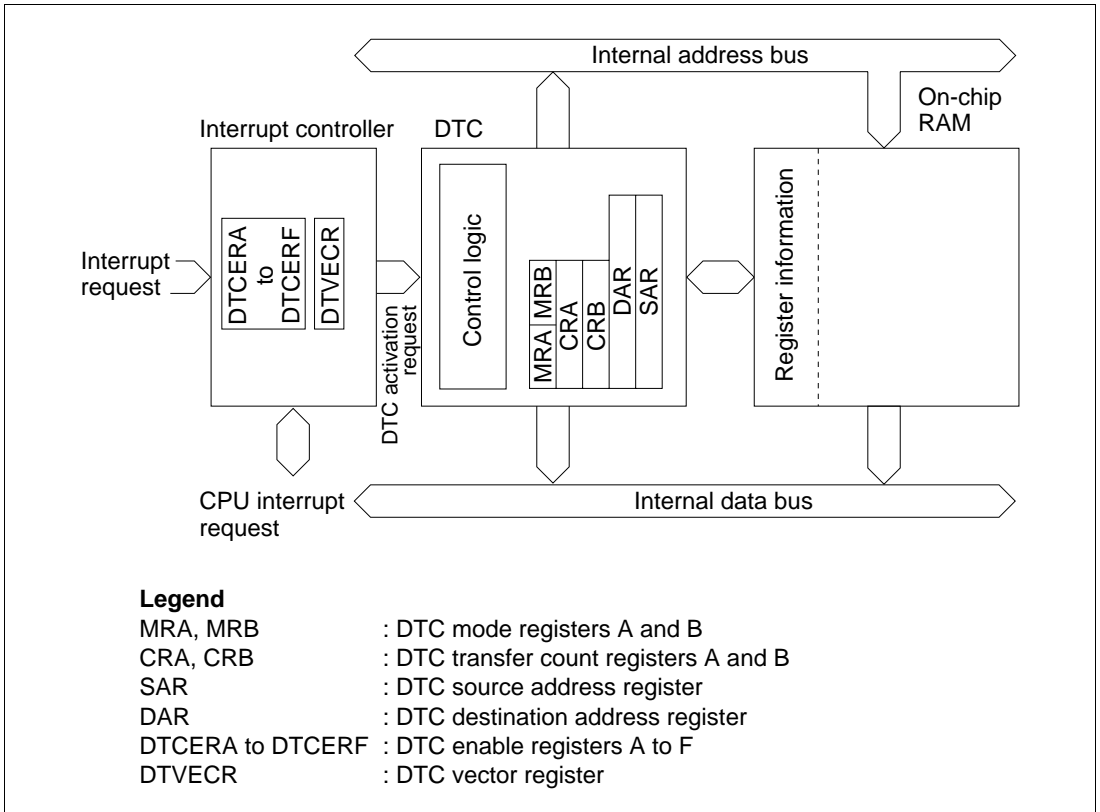
- Transfer possible over any number of channels
  - Transfer information is stored in memory
  - One activation source can trigger a number of data transfers (chain transfer)
  - Chain transfer execution can be set after data transfer (when counter = 0)
- Selection of transfer modes
  - Normal, repeat, and block transfer modes available
  - Incrementing, decrementing, and fixing of source and destination addresses can be selected
- Direct specification of 16-Mbyte address space possible
  - 24-bit transfer source and destination addresses can be specified
- Transfer can be set in byte or word units
- A CPU interrupt can be requested for the interrupt that activated the DTC
  - An interrupt request can be issued to the CPU after one data transfer ends
  - An interrupt request can be issued to the CPU after all the specified data transfers have ended
- Activation by software is possible
- Module stop mode can be set
  - The initial setting enables DTC registers to be accessed. DTC operation is halted by setting module stop mode

## 6.1.2 Block Diagram

Figure 6-1 shows a block diagram of the DTC.

The DTC's register information is stored in the on-chip RAM\*. A 32-bit bus connects the DTC to the on-chip RAM (1 kbyte), enabling 32-bit, 1-state reading and writing of DTC register information.

Note: \* When the DTC is used, the RAME bit in SYSCR must be set to 1.



**Figure 6-1 Block Diagram of DTC**

### 6.1.3 Register Configuration

Table 6-1 summarizes the DTC registers.

**Table 6-1 DTC Registers**

Name	Abbreviation	R/W	Initial Value	Address* <sup>1</sup>
DTC mode register A	MRA	—* <sup>2</sup>	Undefined	—* <sup>3</sup>
DTC mode register B	MRB	—* <sup>2</sup>	Undefined	—* <sup>3</sup>
DTC source address register	SAR	—* <sup>2</sup>	Undefined	—* <sup>3</sup>
DTC destination address register	DAR	—* <sup>2</sup>	Undefined	—* <sup>3</sup>
DTC transfer count register A	CRA	—* <sup>2</sup>	Undefined	—* <sup>3</sup>
DTC transfer count register B	CRB	—* <sup>2</sup>	Undefined	—* <sup>3</sup>
DTC enable registers	DTCER	R/W	H'00	H'FF30 to H'FF35
DTC vector register	DTVECR	R/W	H'00	H'FF37
Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. Lower 16 bits of the address.

2. Registers within the DTC cannot be read or written to directly.

3. Register information is located in on-chip RAM addresses H'F800 to H'FBFF. It cannot be located in external space. When the DTC is used, do not clear the RAME bit in SYSCR to 0.

## 6.2 Register Descriptions

### 6.2.1 DTC Mode Register A (MRA)

Bit	:	7	6	5	4	3	2	1	0
		SM1	SM0	DM1	DM0	MD1	MD0	DTS	Sz
Initial value	:	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined
R/W	:	—	—	—	—	—	—	—	—

MRA is an 8-bit register that controls the DTC operating mode.

**Bits 7 and 6—Source Address Mode 1 and 0 (SM1, SM0):** These bits specify whether SAR is to be incremented, decremented, or left fixed after a data transfer.

Bit 7 SM1	Bit 6 SM0	Description
0	—	SAR is fixed
1	0	SAR is incremented after a transfer (by +1 when Sz = 0; by +2 when Sz = 1)
	1	SAR is decremented after a transfer (by -1 when Sz = 0; by -2 when Sz = 1)

**Bits 5 and 4—Destination Address Mode 1 and 0 (DM1, DM0):** These bits specify whether DAR is to be incremented, decremented, or left fixed after a data transfer.

Bit 5 DM1	Bit 4 DM0	Description
0	—	DAR is fixed
1	0	DAR is incremented after a transfer (by +1 when Sz = 0; by +2 when Sz = 1)
	1	DAR is decremented after a transfer (by -1 when Sz = 0; by -2 when Sz = 1)



**Bits 3 and 2—DTC Mode (MD1, MD0):** These bits specify the DTC transfer mode.

<b>Bit 3 MD1</b>	<b>Bit 2 MD0</b>	<b>Description</b>
0	0	Normal mode
	1	Repeat mode
1	0	Block transfer mode
	1	—

**Bit 1—DTC Transfer Mode Select (DTS):** Specifies whether the source side or the destination side is set to be a repeat area or block area, in repeat mode or block transfer mode.

<b>Bit 1 DTS</b>	<b>Description</b>
0	Destination side is repeat area or block area
1	Source side is repeat area or block area

**Bit 0—DTC Data Transfer Size (Sz):** Specifies the size of data to be transferred.

<b>Bit 0 Sz</b>	<b>Description</b>
0	Byte-size transfer
1	Word-size transfer

## 6.2.2 DTC Mode Register B (MRB)

Bit	:	7	6	5	4	3	2	1	0
		CHNE	DISEL	CHNS	—	—	—	—	—
Initial value :		Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined
R/W	:	—	—	—	—	—	—	—	—

MRB is an 8-bit register that controls the DTC operating mode.

**Bit 7—DTC Chain Transfer Enable (CHNE):** Specifies chain transfer. With chain transfer, a number of data transfers can be performed consecutively in response to a single transfer request.

In data transfer with CHNE set to 1, determination of the end of the specified number of transfers, clearing of the interrupt source flag, and clearing of DTCER are not performed.

When CHNE is set to 1, the chain transfer condition can be selected with the CHNS bit.

### Bit 7

CHNE	Description
0	End of DTC data transfer (activation waiting state)
1	DTC chain transfer (new register information is read, then data is transferred)

**Bit 6—DTC Interrupt Select (DISEL):** Specifies whether interrupt requests to the CPU are disabled or enabled after a data transfer.

### Bit 6

DISEL	Description
0	After a data transfer ends, the CPU interrupt is disabled unless the transfer counter is 0 (the DTC clears the interrupt source flag of the activating interrupt to 0)
1	After a data transfer ends, the CPU interrupt is enabled (the DTC does not clear the interrupt source flag of the activating interrupt to 0)

**Bit 5—DTC Chain Transfer Select (CHNS):** Specifies the chain transfer condition when CHNE is 1.

Bit 7 CHNE	Bit 5 CHNS	Description
0	–	No chain transfer (DTC data transfer end, activation waiting state entered)
1	0	DTC chain transfer
1	1	Chain transfer only when transfer counter = 0

**Bits 4 to 0—Reserved:** These bits have no effect on DTC operation in the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, and should always be written with 0.

### 6.2.3 DTC Source Address Register (SAR)

Bit	:	23	22	21	20	19	---	4	3	2	1	0
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	---	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Initial value	:	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	---	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined
R/W	:	—	—	—	—	—	---	—	—	—	—	—

SAR is a 24-bit register that designates the source address of data to be transferred by the DTC. For word-size transfer, specify an even source address.

### 6.2.4 DTC Destination Address Register (DAR)

Bit	:	23	22	21	20	19	---	4	3	2	1	0
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	---	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Initial value	:	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	---	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined
R/W	:	—	—	—	—	—	---	—	—	—	—	—

DAR is a 24-bit register that designates the destination address of data to be transferred by the DTC. For word-size transfer, specify an even destination address.

### 6.2.5 DTC Transfer Count Register A (CRA)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Initial value	:	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined	Unde- fined
R/W	:	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		← CRAH →										← CRAL →					

CRA is a 16-bit register that designates the number of times data is to be transferred by the DTC.

In normal mode, the entire CRA register functions as a 16-bit transfer counter (1 to 65536). It is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

In repeat mode or block transfer mode, the CRA register is divided into two parts: the upper 8 bits (CRAH) and the lower 8 bits (CRAL). CRAH holds the number of transfers while CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are sent when the count reaches H'00. This operation is repeated.

### 6.2.6 DTC Transfer Count Register B (CRB)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :		Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-	Unde-
		fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-	fin-
R/W	:	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

CRB is a 16-bit register that designates the number of times data is to be transferred by the DTC in block transfer mode. It functions as a 16-bit transfer counter (1 to 65536) that is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

### 6.2.7 DTC Enable Registers (DTCER)

Bit	:	7	6	5	4	3	2	1	0
		DTCE7	DTCE6	DTCE5	DTCE4	DTCE3	DTCE2	DTCE1	DTCE0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The DTC enable registers comprise nine 8-bit readable/writable registers, DTCERA to DTCERI, with bits corresponding to the interrupt sources that can activate the DTC. These bits enable or disable DTC service for the corresponding interrupt sources.

The DTC enable registers are initialized to H'00 by a reset and in hardware standby mode.

## Bit n—DTC Activation Enable (DTCEn)

Bit n DTCEn	Description	
0	DTC activation by this interrupt is disabled [Clearing conditions] <ul style="list-style-type: none"><li>• When the DISEL bit is 1 and the data transfer has ended</li><li>• When the specified number of transfers have ended</li></ul>	(Initial value)
1	DTC activation by this interrupt is enabled [Holding condition] When the DISEL bit is 0 and the specified number of transfers have not ended	

(n = 7 to 0)

A DTCE bit can be set for each interrupt source that can activate the DTC. The correspondence between interrupt sources and DTCE bits is shown in table 6-4, together with the vector numbers generated by the interrupt controller.

For DTCE bit setting, read/write operations must be performed using bit-manipulation instructions such as BSET and BCLR. For the initial setting only, however, when multiple activation sources are set at one time, it is possible to disable interrupts and write after executing a dummy read on the relevant register.

### 6.2.8 DTC Vector Register (DTVECR)

Bit	:	7	6	5	4	3	2	1	0
		SWDTE	DTVEC6	DTVEC5	DTVEC4	DTVEC3	DTVEC2	DTVEC1	DTVEC0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)* <sup>1</sup>	R/(W)* <sup>2</sup>	R/(W)* <sup>2</sup>	R/(W)* <sup>2</sup>	R/(W)* <sup>2</sup>	R/(W)* <sup>2</sup>	R/(W)* <sup>2</sup>	R/(W)* <sup>2</sup>

- Notes: 1. A value of 1 can always be written to the SWDTE bit.  
2. Bits DTVEC6 to DTVEC0 can be written to when SWDTE = 0.

DTVECR is an 8-bit readable/writable register that enables or disables DTC activation by software, and sets a vector number for the software activation interrupt.

DTVECR is initialized to H'00 by a reset and in hardware standby mode.

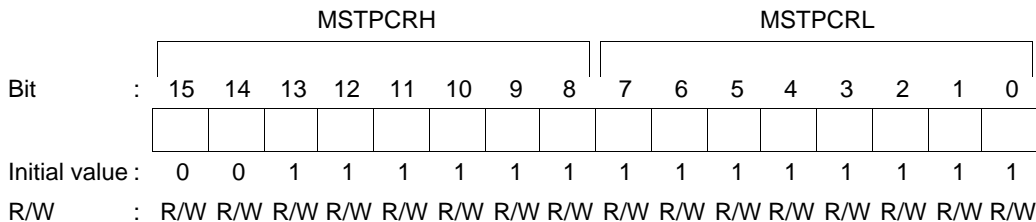
**Bit 7—DTC Software Activation Enable (SWDTE):** Enables or disables DTC activation by software.

Bit 7 SWDTE	Description
0	DTC software activation is disabled (Initial value) [Clearing condition] When the DISEL bit is 0 and the specified number of transfers have not ended
1	DTC software activation is enabled [Holding conditions] <ul style="list-style-type: none"> <li>• When the DISEL bit is 1 and data transfer has ended</li> <li>• When the specified number of transfers have ended</li> <li>• During data transfer due to software activation</li> </ul>

**Bits 6 to 0—DTC Software Activation Vectors 6 to 0 (DTVEC6 to DTVEC0):** These bits specify a vector number for DTC software activation.

The vector address is expressed as  $H'0400 + ((\text{vector number}) \ll 1)$ .  $\ll 1$  indicates a one-bit left-shift. For example, when  $DTVEC6$  to  $DTVEC0 = H'10$ , the vector address is  $H'0420$ .

### 6.2.9 Module Stop Control Register (MSTPCR)



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP14 bit in MSTPCR is set to 1, DTC operation stops at the end of the bus cycle and a transition is made to module stop mode. However, 1 cannot be written in the MSTP14 bit while the DTC is operating. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to  $H'3FFF$  by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 14—Module Stop (MSTP14):** Specifies the DTC module stop mode.

**Bit 14**

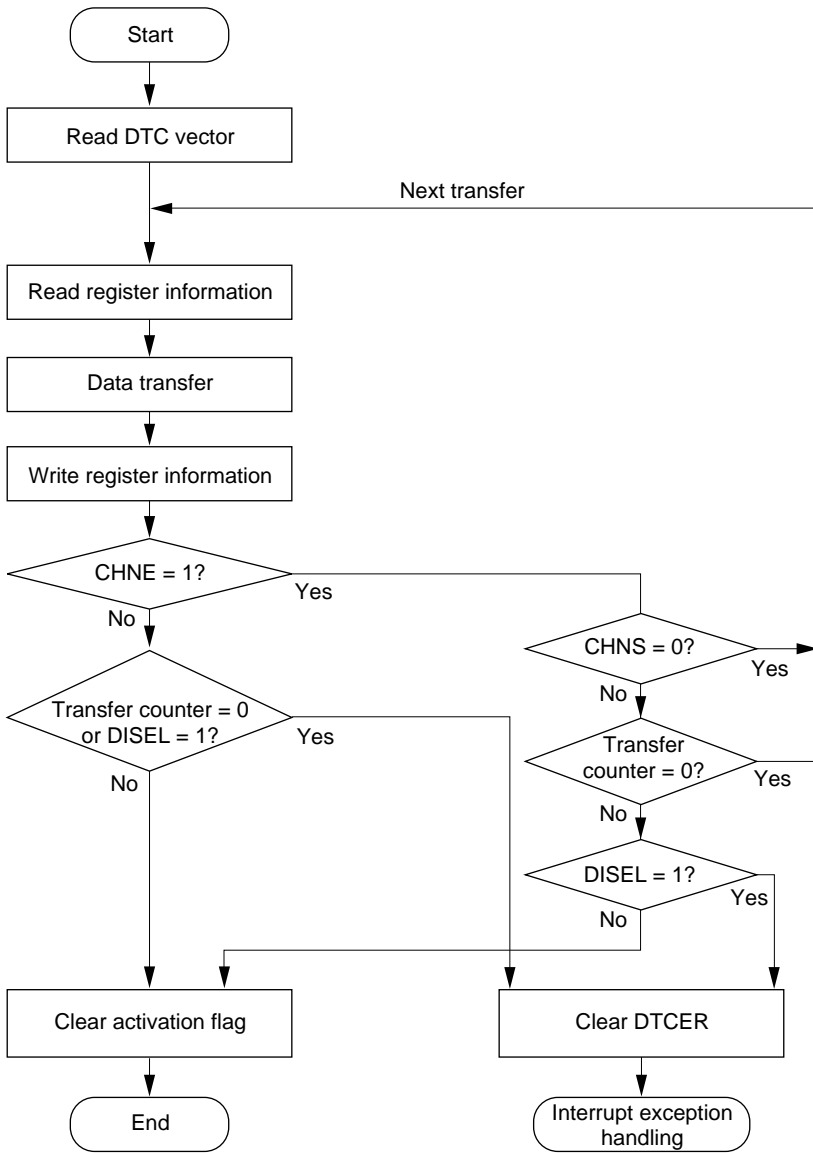
<b>MSTP14</b>	<b>Description</b>	
0	DTC module stop mode cleared	(Initial value)
1	DTC module stop mode set	

## 6.3 Operation

### 6.3.1 Overview

When activated, the DTC reads register information that is already stored in memory and transfers data on the basis of that register information. After the data transfer, it writes updated register information back to memory. Pre-storage of register information in memory makes it possible to transfer data over any required number of channels. Setting the CHNE bit to 1 makes it possible to perform a number of transfers with a single activation. A setting can also be made to have chain transfer performed only when the transfer counter value is 0. This enables DTC re-setting to be performed by the DTC itself.

Figure 6-2 shows a flowchart of DTC operation, and table 6-2 summarizes the chain transfer conditions (combinations for performing the second and third transfers are omitted).



**Figure 6-2 Flowchart of DTC Operation**



**Table 6-2 Chain Transfer Conditions**

1st Transfer				2nd Transfer				DTC Transfer
CHNE	CHNS	DISEL	CR	CHNE	CHNS	DISEL	CR	
0	—	0	Not 0	—	—	—	—	Ends at 1st transfer
0	—	0	0	—	—	—	—	Ends at 1st transfer
0	—	1	—	—	—	—	—	Interrupt request to CPU
1	0	—	—	0	—	0	Not 0	Ends at 2nd transfer
				0	—	0	0	Ends at 2nd transfer
				0	—	1	—	Interrupt request to CPU
1	1	0	Not 0	—	—	—	—	Ends at 1st transfer
1	1	—	0	0	—	0	Not 0	Ends at 2nd transfer
				0	—	0	0	Ends at 2nd transfer
				0	—	1	—	Interrupt request to CPU
1	1	1	Not 0	—	—	—	—	Ends at 1st transfer Interrupt request to CPU

The DTC transfer mode can be normal mode, repeat mode, or block transfer mode.

The 24-bit SAR designates the DTC transfer source address and the 24-bit DAR designates the transfer destination address. After each transfer, SAR and DAR are independently incremented, decremented, or left fixed.

Table 6-3 outlines the functions of the DTC.

**Table 6-3 DTC Functions**

<b>Transfer Mode</b>	<b>Activation Source*</b>	<b>Address Registers</b>	
		<b>Transfer Source</b>	<b>Transfer Destination</b>
<ul style="list-style-type: none"> <li>• Normal mode               <ul style="list-style-type: none"> <li>— One transfer request transfers one byte or one word</li> <li>— Memory addresses are incremented or decremented by 1 or 2</li> <li>— Up to 65,536 transfers possible</li> </ul> </li> <li>• Repeat mode               <ul style="list-style-type: none"> <li>— One transfer request transfers one byte or one word</li> <li>— Memory addresses are incremented or decremented by 1 or 2</li> <li>— After the specified number of transfers (1 to 256), the initial state resumes and operation continues</li> </ul> </li> <li>• Block transfer mode               <ul style="list-style-type: none"> <li>— One transfer request transfers a block of the specified size</li> <li>— Block size is from 1 to 256 bytes or words</li> <li>— Up to 65,536 transfers possible</li> <li>— A block area can be designated at either the source or destination</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• IRQ</li> <li>• TPU TGI</li> <li>• 8-bit timer CMI</li> <li>• SCI TXI or RXI</li> <li>• A/D converter ADI</li> <li>• DMAC DEND</li> <li>• Software</li> </ul>	24 bits	24 bits

Note: \* Activation sources depend on the on-chip modules provided with each model; see the reference manual for the relevant model for details.

### 6.3.2 Activation Sources

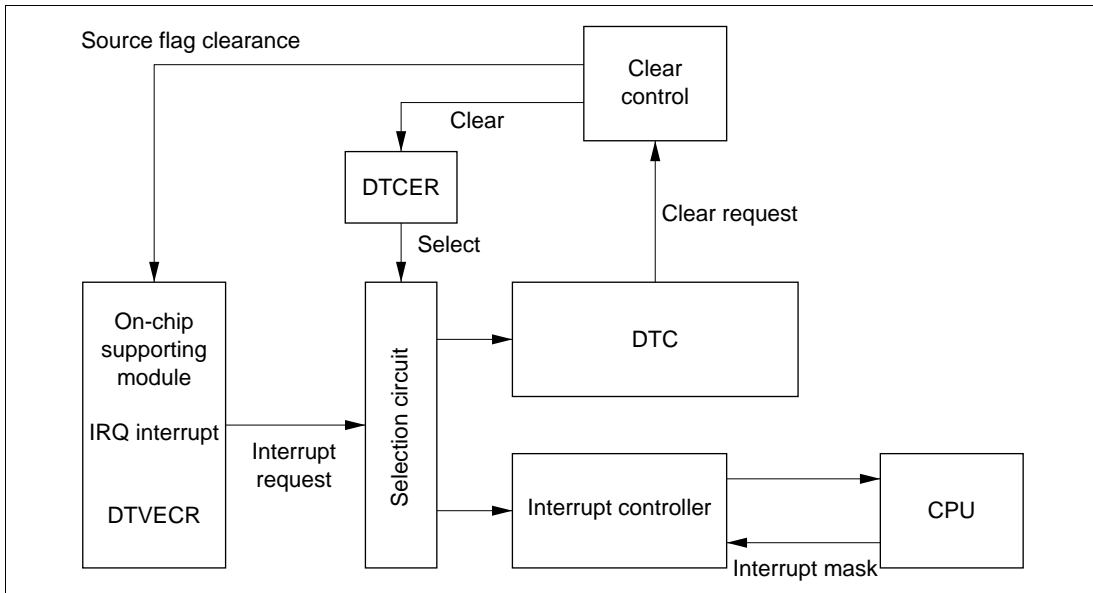
The DTC operates when activated by an interrupt or by a write to DTVECR by software. An interrupt request can be directed to the CPU or DTC, as designated by the corresponding DTCER bit. An interrupt becomes a DTC activation source when the corresponding bit is set to 1, and a CPU interrupt source when the bit is cleared to 0.

At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the activation source or corresponding DTCER bit is cleared. Table 6-4 shows activation source and DTCER clearance. The activation source flag, in the case of RXI0, for example, is the RDRF flag of SCIO.

**Table 6-4 Activation Source and DTCER Clearance**

<b>Activation Source</b>	<b>When the DISEL Bit Is 0 and the Specified Number of Transfers Have Not Ended</b>	<b>When the DISEL Bit Is 1, or when the Specified Number of Transfers Have Ended</b>
Software activation	The SWDTE bit is cleared to 0	<ul style="list-style-type: none"><li>• The SWDTE bit remains set to 1</li><li>• An interrupt is issued to the CPU</li></ul>
Interrupt activation	<ul style="list-style-type: none"><li>• The corresponding DTCER bit remains set to 1</li><li>• The activation source flag is cleared to 0</li></ul>	<ul style="list-style-type: none"><li>• The corresponding DTCER bit is cleared to 0</li><li>• The activation source flag remains set to 1</li><li>• A request is issued to the CPU for the activation source interrupt</li></ul>

Figure 6-3 shows a block diagram of activation source control. For details see section 3, Interrupt Controller.



**Figure 6-3 Block Diagram of DTC Activation Source Control**

When an interrupt has been designated a DTC activation source, existing CPU mask level and interrupt controller priorities have no effect. If there is more than one activation source at the same time, the DTC operates in accordance with the default priorities.

### 6.3.3 DTC Vector Table

Figure 6-4 shows the correspondence between DTC vector addresses and register information.

Table 6-5 shows the correspondence between activation, vector addresses, and DTCER bits. When the DTC is activated by software, the vector address is obtained from:  $H'0400 + (DTVECR[6:0] \ll 1)$  (where  $\ll 1$  indicates a 1-bit left shift). For example, if DTVECR is H'10, the vector address is H'0420.

The DTC reads the start address of the register information from the vector address set for each activation source, and then reads the register information from that start address. The register information can be placed at predetermined addresses in the on-chip RAM. The start address of the register information should be an integral multiple of four.

The configuration of the vector address is the same in both normal\* and advanced modes, a 2-byte unit being used in both cases. These two bytes specify the lower bits of the address in the on-chip RAM.

Note: \* Normal mode is not supported in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series.

**Table 6-5 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs**

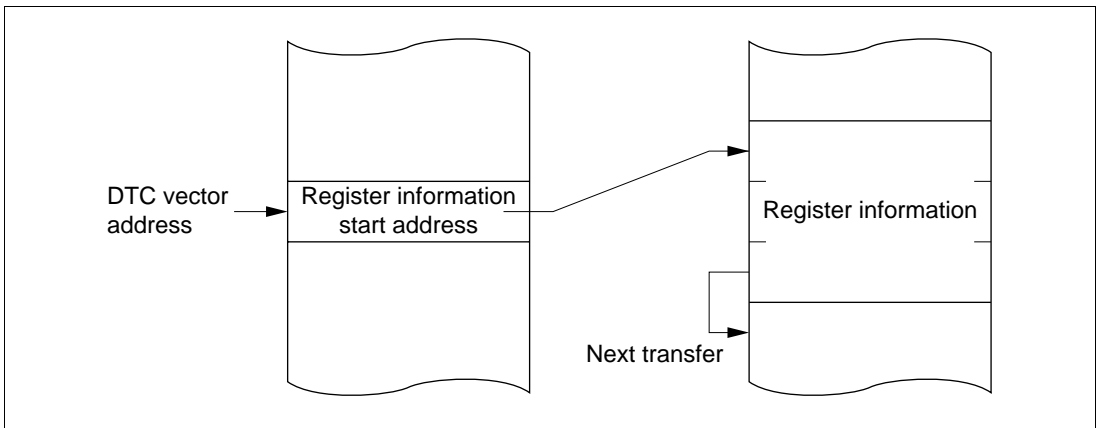
Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address	DTCE*	Priority	
Write to DTVECR	Software	DTVECR	H'0400+ (DTVECR [6:0]<<1)	—	High	
IRQ0	External pin	16	H'0420	DTCEA7	↑ High	
IRQ1		17	H'0422	DTCEA6		
IRQ2		18	H'0424	DTCEA5		
IRQ3		19	H'0426	DTCEA4		
IRQ4		20	H'0428	DTCEA3		
IRQ5		21	H'042A	DTCEA2		
IRQ6		22	H'042C	DTCEA1		
IRQ7		23	H'042E	DTCEA0		
ADI (A/D conversion end)	A/D	28	H'0438	DTCEB6	↑ High	
TGI0A (GR0A compare match/ input capture)	TPU channel 0	32	H'0440	DTCEB5		
TGI0B (GR0B compare match/ input capture)		33	H'0442	DTCEB4		
TGI0C (GR0C compare match/ input capture)		34	H'0444	DTCEB3		
TGI0D (GR0D compare match/ input capture)		35	H'0446	DTCEB2		
TGI1A (GR1A compare match/ input capture)		TPU channel 1	40	H'0450		DTCEB1
TGI1B (GR1B compare match/ input capture)	41		H'0452	DTCEB0		
TGI2A (GR2A compare match/ input capture)	TPU channel 2	44	H'0458	DTCEC7		
TGI2B (GR2B compare match/ input capture)		45	H'045A	DTCEC6		
						Low

**Table 6-5 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs (cont)**

<b>Interrupt Source</b>	<b>Origin of Interrupt Source</b>	<b>Vector Number</b>	<b>Vector Address</b>	<b>DTCE*</b>	<b>Priority</b>
TGI3A (GR3A compare match/ input capture)	TPU channel 3	48	H'0460	DTCEC5	High ↑ Low
TGI3B (GR3B compare match/ input capture)		49	H'0462	DTCEC4	
TGI3C (GR3C compare match/ input capture)		50	H'0464	DTCEC3	
TGI3D (GR3D compare match/ input capture)		51	H'0466	DTCEC2	
TGI4A (GR4A compare match/ input capture)	TPU channel 4	56	H'0470	DTCEC1	
TGI4B (GR4B compare match/ input capture)		57	H'0472	DTCEC0	
TGI5A (GR5A compare match/ input capture)	TPU channel 5	60	H'0478	DTCED5	
TGI5B (GR5B compare match/ input capture)		61	H'047A	DTCED4	
CMIA0	8-bit timer channel 0	64	H'0480	DTCED3	
CMIB0		65	H'0482	DTCED2	
CMIA1	8-bit timer channel 1	68	H'0488	DTCED1	
CMIB1		69	H'048A	DTCED0	
RXI0 (reception complete 0)	SCI channel 0	81	H'04A2	DTCEE3	
TXI0 (transmit data empty 0)		82	H'04A4	DTCEE2	
RXI1 (reception complete 1)	SCI channel 1	85	H'04AA	DTCEE1	
TXI1 (transmit data empty 1)		86	H'04AC	DTCEE0	
RXI2 (reception complete 2)	SCI channel 2	89	H'04B2	DTCEF7	
TXI2 (transmit data empty 2)		90	H'04B4	DTCEF6	

Note: \* DTCE bits with no corresponding interrupt are reserved, and should be written with 0.

Sources depend on the on-chip modules provided with each model; see the reference manual for the relevant model for details.



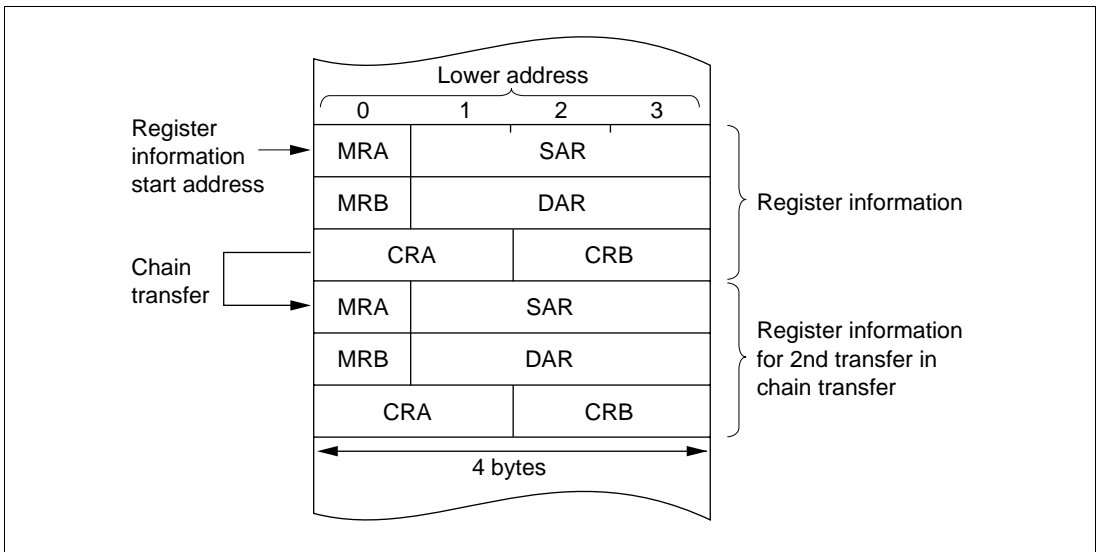
**Figure 6-4 Correspondence between DTC Vector Address and Register Information**

### 6.3.4 Location of Register Information in Address Space

Figure 6-5 shows how the register information should be located in the address space.

Locate the MRA, SAR, MRB, DAR, CRA, and CRB registers, in that order, from the start address of the register information (contents of the vector address). In the case of chain transfer, register information should be located in consecutive areas.

Locate the register information in the on-chip RAM (addresses: H'FFF800 to H'FFFBFF).



**Figure 6-5 Location of DTC Register Information in Address Space**

### 6.3.5 Normal Mode

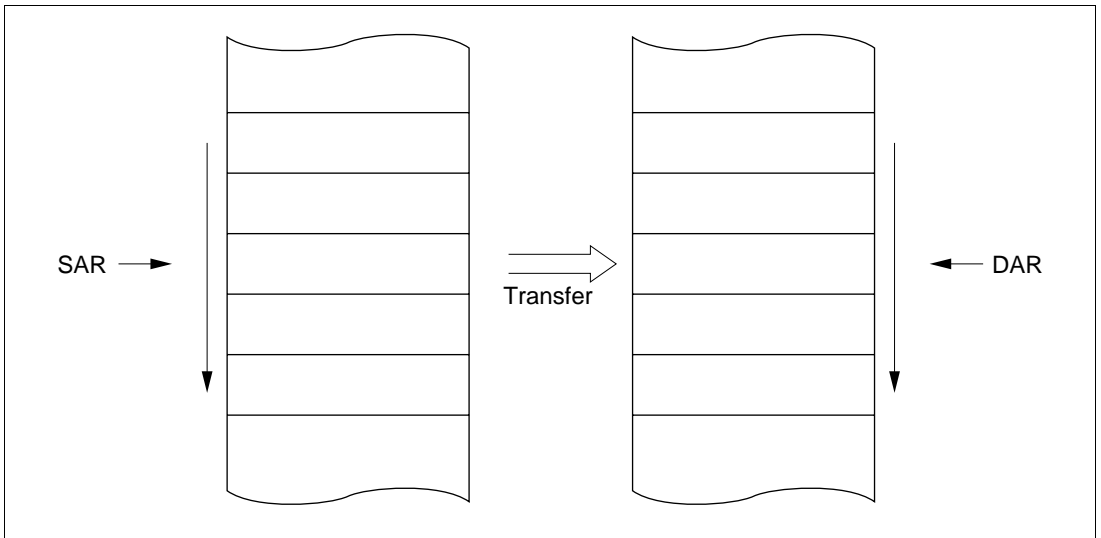
In normal mode, one operation transfers one byte or one word of data.

From 1 to 65,536 transfers can be specified. Once the specified number of transfers have ended, a CPU interrupt can be requested.

Table 6-6 lists the register information in normal mode and figure 6-6 shows the memory map in normal mode.

**Table 6-6 Register Information in Normal Mode**

Name	Abbreviation	Function
DTC source address register	SAR	Designates source address
DTC destination address register	DAR	Designates destination address
DTC transfer count register A	CRA	Designates transfer count
DTC transfer count register B	CRB	Not used



**Figure 6-6 Memory Map in Normal Mode**



### 6.3.6 Repeat Mode

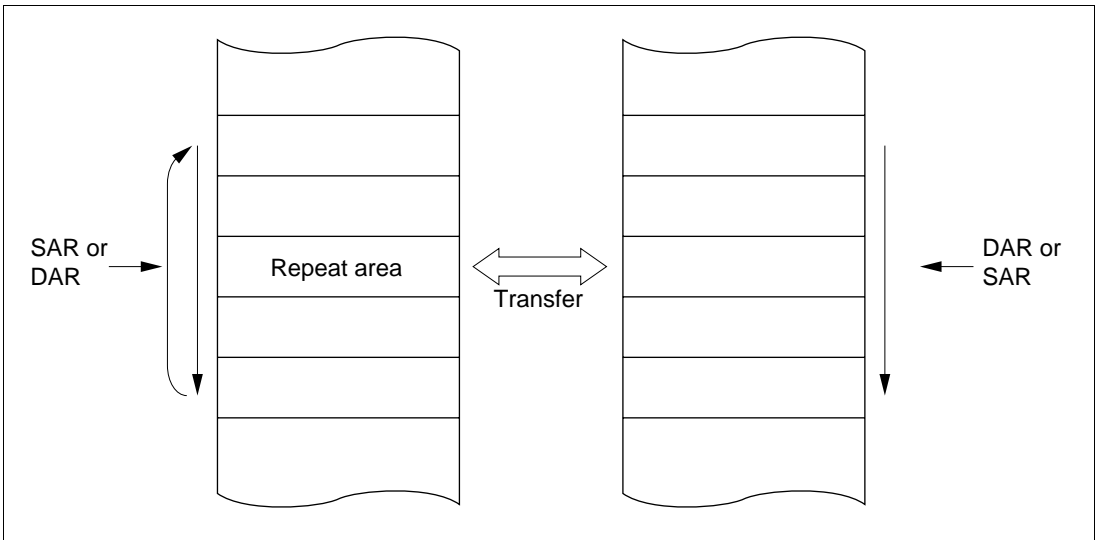
In repeat mode, one operation transfers one byte or one word of data.

From 1 to 256 transfers can be specified. Once the specified number of transfers have ended, the initial state of the transfer counter and the address register specified as the repeat area is restored, and transfer is repeated. In repeat mode the transfer counter value does not reach H'00, and therefore CPU interrupts cannot be requested when DISEL = 0.

Table 6-7 lists the register information in repeat mode and figure 6-7 shows the memory map in repeat mode.

**Table 6-7 Register Information in Repeat Mode**

Name	Abbreviation	Function
DTC source address register	SAR	Designates source address
DTC destination address register	DAR	Designates destination address
DTC transfer count register AH	CRAH	Holds number of transfers
DTC transfer count register AL	CRAL	Transfer counter
DTC transfer count register B	CRB	Not used



**Figure 6-7 Memory Map in Repeat Mode**

### 6.3.7 Block Transfer Mode

In block transfer mode, one operation transfers one block of data.

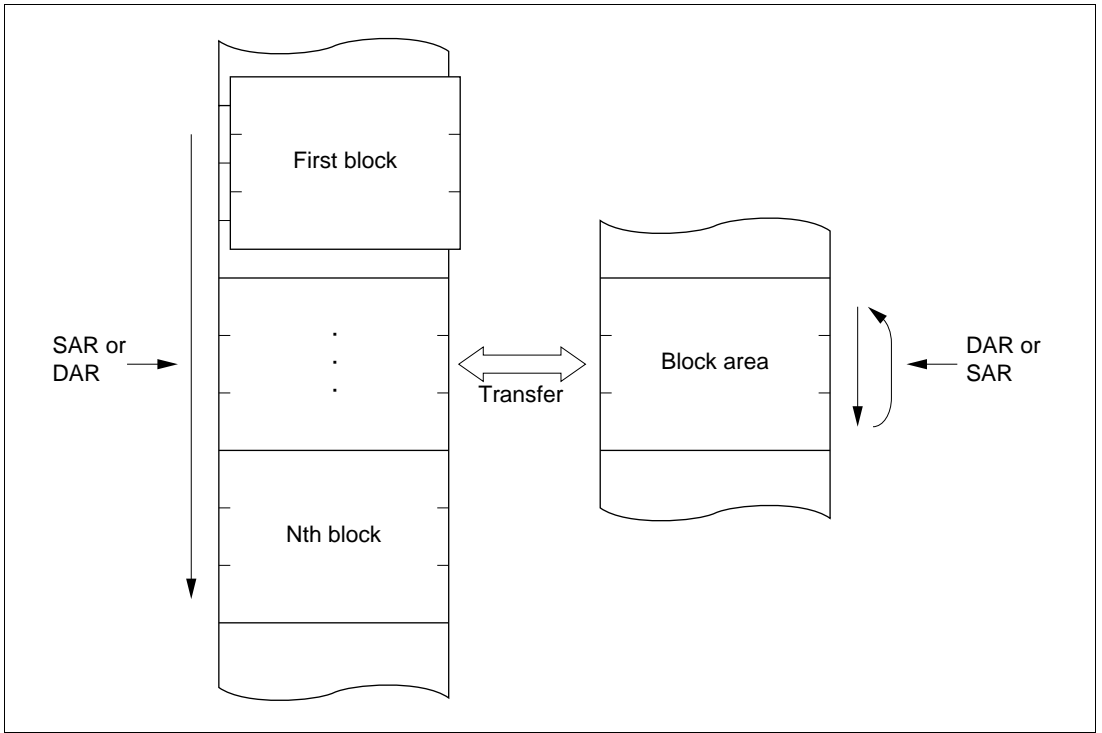
The block size is 1 to 256. When the transfer of one block ends, the initial state of the block size counter and the address register specified as the block area is restored. The other address register is then incremented, decremented, or left fixed.

From 1 to 65,536 transfers can be specified. Once the specified number of transfers have ended, a CPU interrupt is requested.

Table 6-8 lists the register information in block transfer mode and figure 6-8 shows the memory map in block transfer mode.

**Table 6-8 Register Information in Block Transfer Mode**

<b>Name</b>	<b>Abbreviation</b>	<b>Function</b>
DTC source address register	SAR	Designates transfer source address
DTC destination address register	DAR	Designates destination address
DTC transfer count register AH	CRAH	Holds block size
DTC transfer count register AL	CRAL	Block size counter
DTC transfer count register B	CRB	Transfer counter

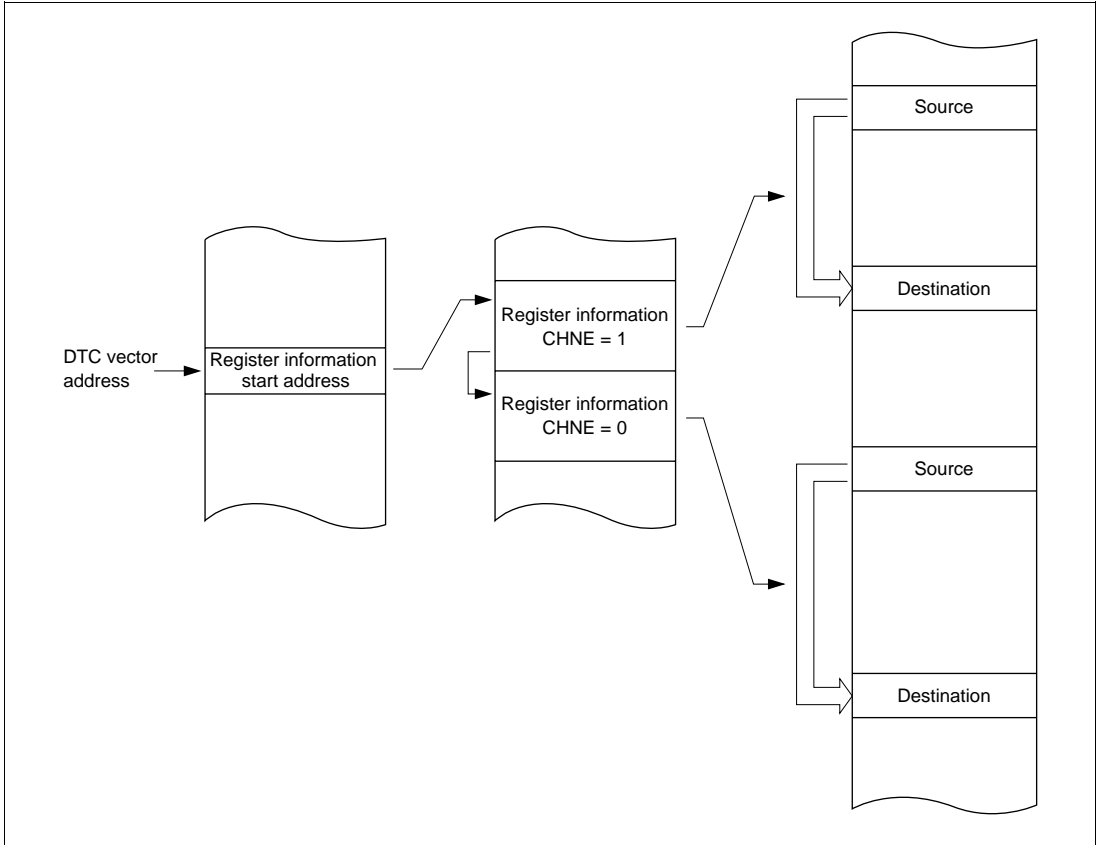


**Figure 6-8 Memory Map in Block Transfer Mode**

### 6.3.8 Chain Transfer

Setting the CHNE bit to 1 enables a number of data transfers to be performed consecutively in response to a single transfer request. It is also possible, by setting both the CHNE bit and CHNS bit to 1, to specify execution of chain transfer only when the transfer counter value is 0. SAR, DAR, CRA, CRB, MRA, and MRB, which define data transfers, can be set independently.

Figure 6-9 shows the memory map for chain transfer.

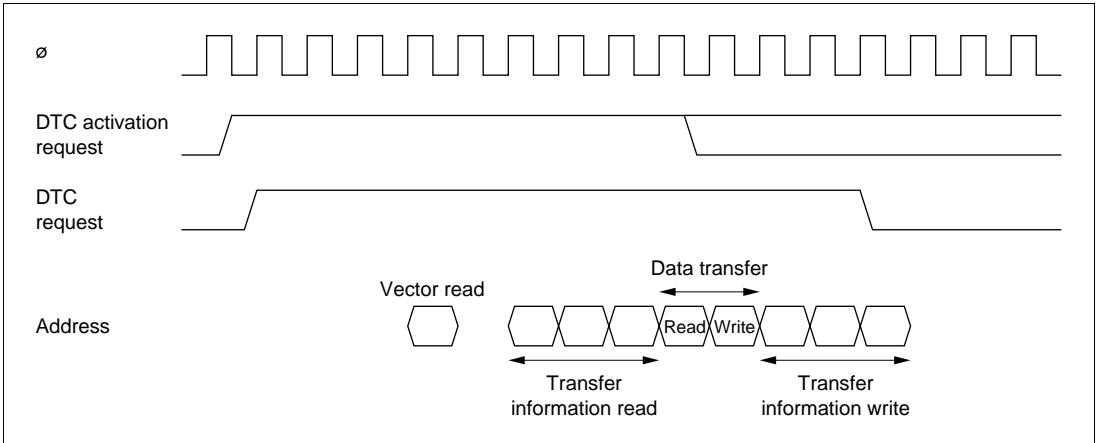


**Figure 6-9 Chain Transfer Memory Map**

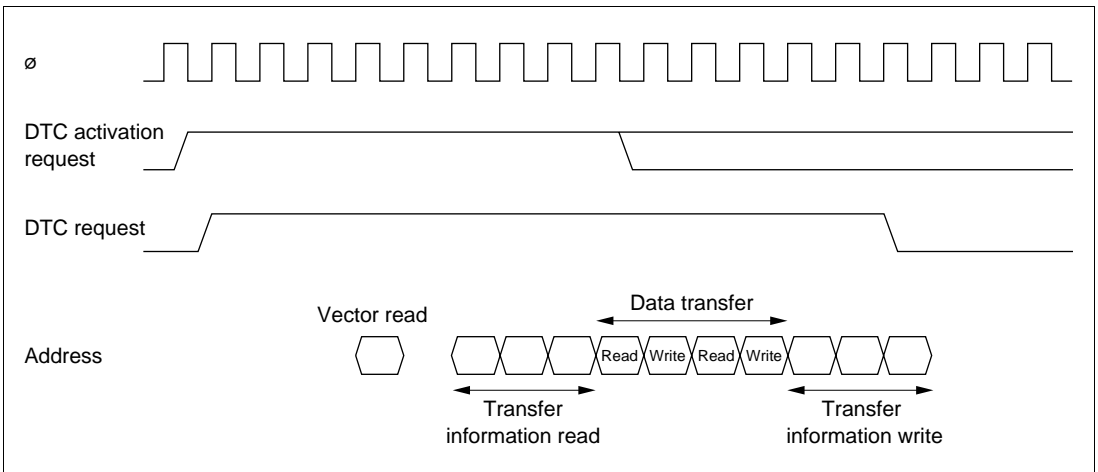
In the case of transfer with CHNE set to 1, an interrupt request to the CPU is not generated at the end of the specified number of transfers or by setting of the DISSEL bit to 1, and the interrupt source flag for the activation source is not affected.

### 6.3.9 Operation Timing

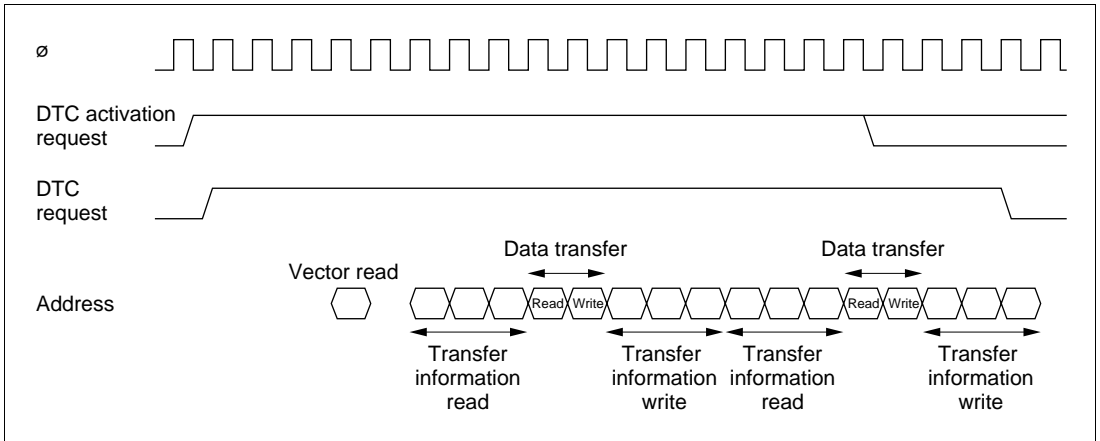
Figures 6-10 to 6-12 show examples of DTC operation timing.



**Figure 6-10 DTC Operation Timing (Example in Normal Mode or Repeat Mode)**



**Figure 6-11 DTC Operation Timing (Example of Block Transfer Mode, with Block Size of 2)**



**Figure 6-12 DTC Operation Timing (Example of Chain Transfer)**

### 6.3.10 Number of DTC Execution States

Table 6-9 lists execution phases for a single DTC data transfer, and table 6-10 shows the number of states required for each execution phase.

**Table 6-9 DTC Execution Phases**

Mode	Vector Read I	Register Information		Internal Operations M	
		Read/Write J	Data Read K		Data Write L
Normal	1	6	1	1	3
Repeat	1	6	1	1	3
Block transfer	1	6	N	N	3

N: Block size (initial setting of CRAH and CRAL)

**Table 6-10 Number of States Required for Each Execution Phase**

Access To:			On- Chip RAM	On- Chip ROM	On-Chip I/O Registers		External Devices			
Bus width			32	16	8	16	8	16		
Access states			1	1	2	2	2	3	2	3
Execution phase	Vector read	$S_I$	—	1	—	—	4	6+2m	2	3+m
	Register information read/write	$S_J$	1	—	—	—	—	—	—	—
	Byte data read	$S_K$	1	1	2	2	2	3+m	2	3+m
	Word data read	$S_K$	1	1	4	2	4	6+2m	2	3+m
	Byte data write	$S_L$	1	1	2	2	2	3+m	2	3+m
	Word data write	$S_L$	1	1	4	2	4	6+2m	2	3+m
Internal operation		$S_M$	1							

The number of execution states is calculated from the formula below. Note that  $\Sigma$  means the sum of all transfers activated by one activation event (the number in which the CHNE bit is set to 1, plus 1).

$$\text{Number of execution states} = I \cdot S_I + \Sigma (J \cdot S_J + K \cdot S_K + L \cdot S_L) + M \cdot S_M$$

For example, when the DTC vector address table is located in on-chip ROM, normal mode is set, and data is transferred from the on-chip ROM to an internal I/O register, the time required for the DTC operation is 13 states. The time from activation to the end of the data write is 10 states.

### 6.3.11 Procedures for Using DTC

**Activation by Interrupt:** The procedure for using the DTC with interrupt activation is as follows:

- [1] Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in the on-chip RAM.
- [2] Set the start address of the register information in the DTC vector address.
- [3] Set the corresponding bit in DTCER to 1.
- [4] Set the enable bits for the interrupt sources to be used as the activation sources to 1. The DTC is activated when an interrupt used as an activation source is generated.
- [5] After the end of one data transfer, or after the specified number of data transfers have ended, the DTCE bit is cleared to 0 and a CPU interrupt is requested. If the DTC is to continue transferring data, set the DTCE bit to 1.

**Activation by Software:** The procedure for using the DTC with software activation is as follows:

- [1] Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in the on-chip RAM.
- [2] Set the start address of the register information in the DTC vector address.
- [3] Check that the SWDTE bit is 0.
- [4] Write 1 to the SWDTE bit and the vector number to DTVECR.
- [5] Check the vector number written to DTVECR.
- [6] After the end of one data transfer, if the DISEL bit is 0 and a CPU interrupt is not requested, the SWDTE bit is cleared to 0. If the DTC is to continue transferring data, set the SWDTE bit to 1. When the DISEL bit is 1, or after the specified number of data transfers have ended, the SWDTE bit is held at 1 and a CPU interrupt is requested.



### 6.3.12 Examples of Use of the DTC

**Normal Mode:** An example is shown in which the DTC is used to receive 128 bytes of data via the SCI.

- [1] Set MRA to fixed source address ( $SM1 = SM0 = 0$ ), incrementing destination address ( $DM1 = 1$ ,  $DM0 = 0$ ), normal mode ( $MD1 = MD0 = 0$ ), and byte size ( $Sz = 0$ ). The DTS bit can have any value. Set MRB for one data transfer by one interrupt ( $CHNE = 0$ ,  $DISEL = 0$ ). Set the SCI RDR address in SAR, the start address of the RAM area where the data will be received in DAR, and 128 (H'0080) in CRA. CRB can be set to any value.
- [2] Set the start address of the register information at the DTC vector address.
- [3] Set the corresponding bit in DTCER to 1.
- [4] Set the SCI to the appropriate receive mode. Set the RIE bit in SCR to 1 to enable the reception complete (RXI) interrupt. Since the generation of a receive error during the SCI receive operation will disable subsequent reception, the CPU should be enabled to accept receive error interrupts.
- [5] Each time reception of one byte of data ends on the SCI, the RDRF flag in SSR is set to 1, an RXI interrupt is generated, and the DTC is activated. The receive data is transferred from RDR to RAM by the DTC. DAR is incremented and CRA is decremented. The RDRF flag is automatically cleared to 0.
- [6] When CRA becomes 0 after the 128 data transfers have ended, the RDRF flag is held at 1, the DTCE bit is cleared to 0, and an RXI interrupt request is sent to the CPU. The interrupt handling routine should perform wrap-up processing.

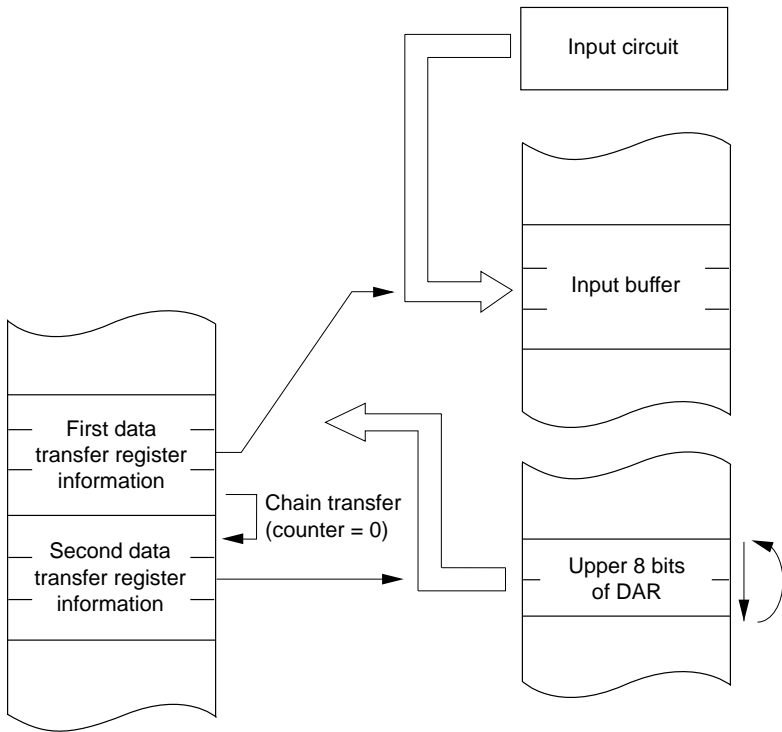
**Chain Transfer:** An example of DTC chain transfer is shown in which pulse output is performed using the PPG. Chain transfer can be used to perform pulse output data transfer and PPG output trigger cycle updating. Repeat mode transfer to the PPG's NDR is performed in the first half of the chain transfer, and normal mode transfer to the TPU's TGR in the second half. This is because clearing of the activation source and interrupt generation at the end of the specified number of transfers are restricted to the second half of the chain transfer (transfer when CHNE = 0).

- [1] Perform settings for transfer to the PPG's NDR. Set MRA to source address incrementing (SM1 = 1, SM0 = 0), fixed destination address (DM1 = DM0 = 0), repeat mode (MD1 = 0, MD0 = 1), and word size (Sz = 1). Set the source side as a repeat area (DTS = 1). Set MRB to chain mode (CHNE = 1, DISEL = 0). Set the data table start address in SAR, the NDRH address in DAR, and the data table size in CRAH and CRAL. CRB can be set to any value.
- [2] Perform settings for transfer to the TPU's TGR. Set MRA to source address incrementing (SM1 = 1, SM0 = 0), fixed destination address (DM1 = DM0 = 0), normal mode (MD1 = MD0 = 0), and word size (Sz = 1). Set the data table start address in SAR, the TGRA address in DAR, and the data table size in CRA. CRB can be set to any value.
- [3] Locate the TPU transfer register information consecutively after the NDR transfer register information.
- [4] Set the start address of the NDR transfer register information to the DTC vector address.
- [5] Set the bit corresponding to TGIA in DTCER to 1.
- [6] Set TGRA as an output compare register (output disabled) with TIOR, and enable the TGIA interrupt with TIER.
- [7] Set the initial output value in PODR, and the next output value in NDR. Set bits in DDR and NDER for which output is to be performed to 1. Using PCR, select the TPU compare match to be used as the output trigger.
- [8] Set the CST bit in TSTR to 1, and start the TCNT count operation.
- [9] Each time a TGRA compare match occurs, the next output value is transferred to NDR and the set value of the next output trigger period is transferred to TGRA. The activation source TGFA flag is cleared.
- [10] When the specified number of transfers are completed (the TPU transfer CRA value is 0), the TGFA flag is held at 1, the DTCE bit is cleared to 0, and a TGIA interrupt request is sent to the CPU. Wrap-up processing should be performed in the interrupt handling routine.

**Chain Transfer when Counter = 0:** By executing a second data transfer, and performing re-setting of the first data transfer, only when the counter value is 0, it is possible to perform 256 or more repeat transfers.

An example is shown in which a 128-kbyte input buffer is configured. The input buffer is assumed to have been set to start at lower address H'0000. Figure 6-13 shows the memory map.

- [1] For the first transfer, set the normal mode for input data. Set fixed transfer source address (G/A, etc.), CRA = H'0000 (64k times), and CHNE = 1, CHNS = 1, and DISEL = 0.
- [2] Prepare the upper 8-bit addresses of the start addresses for each of the 64k transfer start addresses for the first data transfer in a separate area (in ROM, etc.). For example, if the input buffer comprises H'200000 to H'21FFFF, prepare H'21 and H'20.
- [3] For the second transfer, set repeat mode (with the source side as the repeat area) for re-setting the transfer destination address for the first data transfer. Use the upper 8 bits of DAR in the first register information area as the transfer destination. Set CHNE = DISEL = 0. If the above input buffer is specified as H'200000 to H'21FFFF, set the transfer counter to 2.
- [4] Execute the first data transfer 64k times by means of interrupts. When the transfer counter for the first data transfer reaches 0, the second data transfer is started. Set the upper 8 bits of the transfer source address for the first data transfer to H'21. The lower 16 bits of the transfer destination address of the first data transfer and the transfer counter are H'0000.
- [5] Next, execute the first data transfer the 64k times specified for the first data transfer by means of interrupts. When the transfer counter for the first data transfer reaches 0, the second data transfer is started. Set the upper 8 bits of the transfer source address for the first data transfer to H'20. The lower 16 bits of the transfer destination address of the first data transfer and the transfer counter are H'0000.
- [6] Steps [4] and [5] are repeated endlessly. As repeat mode is specified for the second data transfer, an interrupt request is not sent to the CPU.



**Figure 6-13 Chain Transfer when Counter = 0**

**Software Activation:** An example is shown in which the DTC is used to transfer a block of 128 bytes of data by means of software activation. The transfer source address is H'1000 and the destination address is H'2000. The vector number is H'60, so the vector address is H'04C0.

- [1] Set MRA to incrementing source address (SM1 = 1, SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), block transfer mode (MD1 = 1, MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one block transfer by one interrupt (CHNE = 0). Set the transfer source address (H'1000) in SAR, the destination address (H'2000) in DAR, and 128 (H'8080) in CRA. Set 1 (H'0001) in CRB.
- [2] Set the start address of the register information at the DTC vector address (H'04C0).
- [3] Check that the SWDTE bit in DTVECR is 0. Check that there is currently no transfer activated by software.
- [4] Write 1 to the SWDTE bit and the vector number (H'60) to DTVECR. The write data is H'E0.
- [5] Read DTVECR again and check that it is set to the vector number (H'60). If it is not, this indicates that the write failed. This is presumably because an interrupt occurred between steps 3 and 4 and led to a different software activation. To activate this transfer, go back to step 3.
- [6] If the write was successful, the DTC is activated and a block of 128 bytes of data is transferred.
- [7] After the transfer, an SWDTEND interrupt occurs. The interrupt handling routine should clear the SWDTE bit to 0 and perform other wrap-up processing.

## 6.4 Interrupts

An interrupt request is issued to the CPU when the DTC finishes the specified number of data transfers, or a data transfer for which the DISEL bit was set to 1. In the case of interrupt activation, the interrupt set as the activation source is generated. These interrupts to the CPU are subject to CPU mask level and interrupt controller priority level control.

In the case of activation by software, a software activated data transfer end interrupt (SWDTEND) is generated.

When the DISEL bit is 1 and one data transfer has ended, or the specified number of transfers have ended, after data transfer ends, the SWDTE bit is held at 1 and an SWDTEND interrupt is generated. The interrupt handling routine should clear the SWDTE bit to 0.

When the DTC is activated by software, an SWDTEND interrupt is not generated during a data transfer wait or during data transfer even if the SWDTE bit is set to 1.

## 6.5 Usage Notes

**Module Stop:** When the MSTP14 bit in MSTPCR is set to 1, the DTC clock stops, and the DTC enters the module stop state. However, 1 cannot be written to the MSTP14 bit while the DTC is operating.

**On-Chip RAM:** The MRA, MRB, SAR, DAR, CRA, and CRB registers are all located in on-chip RAM. When the DTC is used, the RAME bit in SYSCR must not be cleared to 0.

**DMAC Transfer End Interrupt:** When DTC transfer is activated by a DMAC transfer end interrupt, regardless of the transfer counter and DISEL bit, the DMAC's DTE bit is not subject to DTC control, and the write data has priority. Consequently, an interrupt request may not be sent to the CPU when the DTC transfer counter reaches 0.

**DTCE Bit Setting:** For DTCE bit setting, read/write operations must be performed using bit-manipulation instructions such as BSET and BCLR. For the initial setting only, however, when multiple activation sources are set at one time, it is possible to disable interrupts and write after executing a dummy read on the relevant register.

**Chain Transfer:** When chain transfer is used, clearing of the activation source or DTCER is performed when the last of the chain of data transfers is executed. SCI and high-speed A/D converter interrupt/activation sources, on the other hand, are cleared when the DTC reads or writes to the prescribed register.

Therefore, when the DTC is activated by an interrupt or activation source, if a read/write of the relevant register is not included in the last chained data transfer, the interrupt or activation source will be retained.

# Section 7 16-Bit Timer Pulse Unit (TPU)

## 7.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have an on-chip 16-bit timer pulse unit (TPU) that comprises six 16-bit timer channels.

### 7.1.1 Features

- Maximum 16-pulse input/output
  - A total of 16 timer general registers (TGRs) are provided (four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5), each of which can be set independently as an output compare/input capture register
  - TGRC and TGRD for channels 0 and 3 can also be used as buffer registers
- Selection of 8 counter input clocks for each channel
- The following operations can be set for each channel:
  - Waveform output at compare match: Selection of 0, 1, or toggle output
  - Input capture function: Selection of rising edge, falling edge, or both edge detection
  - Counter clear operation: Counter clearing possible by compare match or input capture
  - Synchronous operation: Multiple timer counters (TCNT) can be written to simultaneously
  - Simultaneous clearing by compare match and input capture possible
  - Register simultaneous input/output possible by counter synchronous operation
  - PWM mode: Any PWM output duty can be set
  - Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
  - Input capture register double-buffering possible
  - Automatic rewriting of output compare register possible
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
  - Two-phase encoder pulse up/down-count possible
- Cascaded operation
  - Channel 2 (channel 5) input clock operates as 32-bit counter by setting channel 1 (channel 4) overflow/underflow
- Fast access via internal 16-bit bus
  - Fast access is possible via a 16-bit bus interface

- 26 interrupt sources
  - For channels 0 and 3, four compare match/input capture dual-function interrupts and one overflow interrupt can be requested independently
  - For channels 1, 2, 4, and 5, two compare match/input capture dual-function interrupts, one overflow interrupt, and one underflow interrupt can be requested independently
- Automatic transfer of register data
  - Block transfer, 1-word data transfer, and 1-byte data transfer possible by data transfer controller (DTC) or DMA controller (DMAC) activation
- Programmable pulse generator (PPG) output trigger can be generated
  - Channel 0 to 3 compare match/input capture signals can be used as PPG output trigger
- A/D converter conversion start trigger can be generated
  - Channel 0 to 5 compare match A/input capture A signals can be used as A/D converter conversion start trigger
- Module stop mode can be set
  - As the initial setting, TPU operation is halted. Register access is enabled by exiting module stop mode

Table 7-1 lists the functions of the TPU.



**Table 7-1 TPU Functions**

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
Count clock	$\phi/1$	$\phi/1$	$\phi/1$	$\phi/1$	$\phi/1$	$\phi/1$
	$\phi/4$	$\phi/4$	$\phi/4$	$\phi/4$	$\phi/4$	$\phi/4$
	$\phi/16$	$\phi/16$	$\phi/16$	$\phi/16$	$\phi/16$	$\phi/16$
	$\phi/64$	$\phi/64$	$\phi/64$	$\phi/64$	$\phi/64$	$\phi/64$
	TCLKA	$\phi/256$	$\phi/1024$	$\phi/256$	$\phi/1024$	$\phi/256$
	TCLKB	TCLKA	TCLKA	$\phi/1024$	TCLKA	TCLKA
	TCLKC	TCLKB	TCLKB	$\phi/4096$	TCLKC	TCLKC
	TCLKD		TCLKC	TCLKA		TCLKD
General registers	TGR0A	TGR1A	TGR2A	TGR3A	TGR4A	TGR5A
	TGR0B	TGR1B	TGR2B	TGR3B	TGR4B	TGR5B
General registers/ buffer registers	TGR0C	—	—	TGR3C	—	—
	TGR0D			TGR3D		
I/O pins	TIOCA0	TIOCA1	TIOCA2	TIOCA3	TIOCA4	TIOCA5
	TIOCB0	TIOCB1	TIOCB2	TIOCB3	TIOCB4	TIOCB5
	TIOCC0			TIOCC3		
	TIOCD0			TIOCD3		
Counter clear function	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture
Compare match output	0 output	○	○	○	○	○
	1 output	○	○	○	○	○
	Toggle output	○	○	○	○	○
Input capture function	○	○	○	○	○	○
Synchronous operation	○	○	○	○	○	○
PWM mode	○	○	○	○	○	○
Phase counting mode	—	○	○	—	○	○
Buffer operation	○	—	—	○	—	—

**Legend**

○ : Possible

— : Not possible

**Table 7-1 TPU Functions (cont)**

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
DMAC activation	TGR0A compare match or input capture	TGR1A compare match or input capture	TGR2A compare match or input capture	TGR3A compare match or input capture	TGR4A compare match or input capture	TGR5A compare match or input capture
DTC activation	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture
A/D conversion start trigger	TGR0A compare match or input capture	TGR1A compare match or input capture	TGR2A compare match or input capture	TGR3A compare match or input capture	TGR4A compare match or input capture	TGR5A compare match or input capture
PPG trigger	TGR0A/ TGR0B compare match or input capture	TGR1A/ TGR1B compare match or input capture	TGR2A/ TGR2B compare match or input capture	TGR3A/ TGR3B compare match or input capture	—	—
Interrupt sources	5 sources <ul style="list-style-type: none"> <li>• Compare match or input capture 0A</li> <li>• Compare match or input capture 0B</li> <li>• Compare match or input capture 0C</li> <li>• Compare match or input capture 0D</li> <li>• Overflow</li> </ul>	4 sources <ul style="list-style-type: none"> <li>• Compare match or input capture 1A</li> <li>• Compare match or input capture 1B</li> <li>• Overflow</li> <li>• Underflow</li> </ul>	4 sources <ul style="list-style-type: none"> <li>• Compare match or input capture 2A</li> <li>• Compare match or input capture 2B</li> <li>• Overflow</li> <li>• Underflow</li> </ul>	5 sources <ul style="list-style-type: none"> <li>• Compare match or input capture 3A</li> <li>• Compare match or input capture 3B</li> <li>• Compare match or input capture 3C</li> <li>• Compare match or input capture 3D</li> <li>• Overflow</li> </ul>	4 sources <ul style="list-style-type: none"> <li>• Compare match or input capture 4A</li> <li>• Compare match or input capture 4B</li> <li>• Overflow</li> <li>• Underflow</li> </ul>	4 sources <ul style="list-style-type: none"> <li>• Compare match or input capture 5A</li> <li>• Compare match or input capture 5B</li> <li>• Overflow</li> <li>• Underflow</li> </ul>

Legend

— : Not possible

## 7.1.2 Block Diagram

Figure 7-1 shows a block diagram of the TPU.

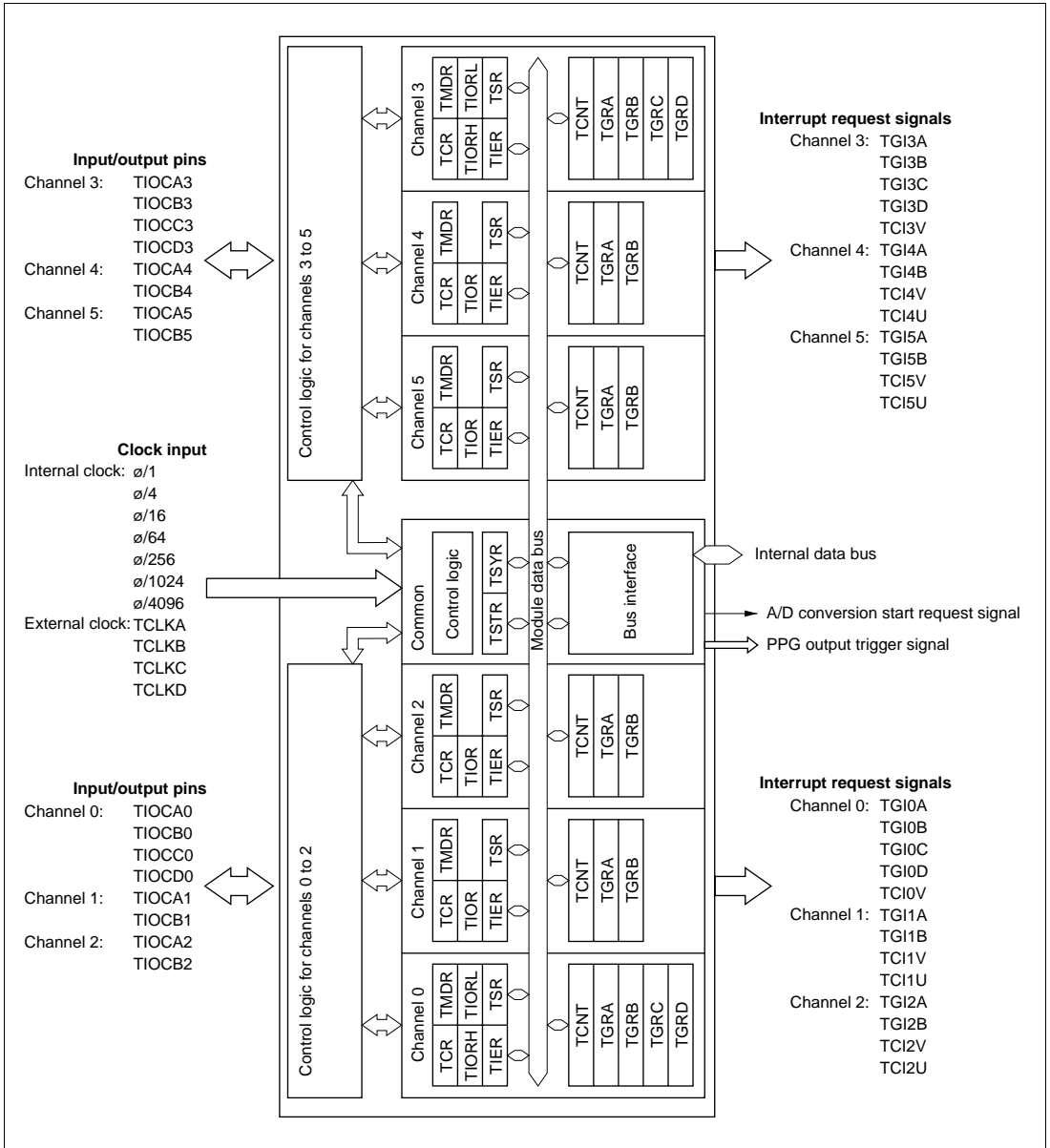


Figure 7-1 Block Diagram of TPU

### 7.1.3 Pin Configuration

Table 7-2 summarizes the TPU pins.

**Table 7-2 TPU Pins**

Channel	Name	Symbol	I/O	Function
All	Clock input A	TCLKA	Input	External clock A input pin (Channel 1 and 5 phase counting mode A phase input)
	Clock input B	TCLKB	Input	External clock B input pin (Channel 1 and 5 phase counting mode B phase input)
	Clock input C	TCLKC	Input	External clock C input pin (Channel 2 and 4 phase counting mode A phase input)
	Clock input D	TCLKD	Input	External clock D input pin (Channel 2 and 4 phase counting mode B phase input)
0	Input capture/out compare match A0	TIOCA0	I/O	TGR0A input capture input/output compare output/PWM output pin
	Input capture/out compare match B0	TIOCB0	I/O	TGR0B input capture input/output compare output/PWM output pin
	Input capture/out compare match C0	TIOCC0	I/O	TGR0C input capture input/output compare output/PWM output pin
	Input capture/out compare match D0	TIOCD0	I/O	TGR0D input capture input/output compare output/PWM output pin
1	Input capture/out compare match A1	TIOCA1	I/O	TGR1A input capture input/output compare output/PWM output pin
	Input capture/out compare match B1	TIOCB1	I/O	TGR1B input capture input/output compare output/PWM output pin
2	Input capture/out compare match A2	TIOCA2	I/O	TGR2A input capture input/output compare output/PWM output pin
	Input capture/out compare match B2	TIOCB2	I/O	TGR2B input capture input/output compare output/PWM output pin

**Table 7-2 TPU Pins (cont)**

<b>Channel</b>	<b>Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
3	Input capture/out compare match A3	TIOCA3	I/O	TGR3A input capture input/output compare output/PWM output pin
	Input capture/out compare match B3	TIOCB3	I/O	TGR3B input capture input/output compare output/PWM output pin
	Input capture/out compare match C3	TIOCC3	I/O	TGR3C input capture input/output compare output/PWM output pin
	Input capture/out compare match D3	TIOCD3	I/O	TGR3D input capture input/output compare output/PWM output pin
4	Input capture/out compare match A4	TIOCA4	I/O	TGR4A input capture input/output compare output/PWM output pin
	Input capture/out compare match B4	TIOCB4	I/O	TGR4B input capture input/output compare output/PWM output pin
5	Input capture/out compare match A5	TIOCA5	I/O	TGR5A input capture input/output compare output/PWM output pin
	Input capture/out compare match B5	TIOCB5	I/O	TGR5B input capture input/output compare output/PWM output pin

## 7.1.4 Register Configuration

Table 7-3 summarizes the TPU registers.

**Table 7-3 TPU Registers**

Channel	Name	Abbreviation	R/W	Initial Value	Address *1
0	Timer control register 0	TCR0	R/W	H'00	H'FFD0
	Timer mode register 0	TMDR0	R/W	H'C0	H'FFD1
	Timer I/O control register 0H	TIOR0H	R/W	H'00	H'FFD2
	Timer I/O control register 0L	TIOR0L	R/W	H'00	H'FFD3
	Timer interrupt enable register 0	TIER0	R/W	H'40	H'FFD4
	Timer status register 0	TSR0	R/(W)*2	H'C0	H'FFD5
	Timer counter 0	TCNT0	R/W	H'0000	H'FFD6
	Timer general register 0A	TGR0A	R/W	H'FFFF	H'FFD8
	Timer general register 0B	TGR0B	R/W	H'FFFF	H'FFDA
	Timer general register 0C	TGR0C	R/W	H'FFFF	H'FFDC
	Timer general register 0D	TGR0D	R/W	H'FFFF	H'FFDE
1	Timer control register 1	TCR1	R/W	H'00	H'FFE0
	Timer mode register 1	TMDR1	R/W	H'C0	H'FFE1
	Timer I/O control register 1	TIOR1	R/W	H'00	H'FFE2
	Timer interrupt enable register 1	TIER1	R/W	H'40	H'FFE4
	Timer status register 1	TSR1	R/(W)*2	H'C0	H'FFE5
	Timer counter 1	TCNT1	R/W	H'0000	H'FFE6
	Timer general register 1A	TGR1A	R/W	H'FFFF	H'FFE8
	Timer general register 1B	TGR1B	R/W	H'FFFF	H'FFEA
2	Timer control register 2	TCR2	R/W	H'00	H'FFF0
	Timer mode register 2	TMDR2	R/W	H'C0	H'FFF1
	Timer I/O control register 2	TIOR2	R/W	H'00	H'FFF2
	Timer interrupt enable register 2	TIER2	R/W	H'40	H'FFF4
	Timer status register 2	TSR2	R/(W)*2	H'C0	H'FFF5
	Timer counter 2	TCNT2	R/W	H'0000	H'FFF6
	Timer general register 2A	TGR2A	R/W	H'FFFF	H'FFF8
	Timer general register 2B	TGR2B	R/W	H'FFFF	H'FFFA

**Table 7-3 TPU Registers (cont)**

Channel	Name	Abbreviation	R/W	Initial Value	Address* <sup>1</sup>
3	Timer control register 3	TCR3	R/W	H'00	H'FE80
	Timer mode register 3	TMDR3	R/W	H'C0	H'FE81
	Timer I/O control register 3H	TIOR3H	R/W	H'00	H'FE82
	Timer I/O control register 3L	TIOR3L	R/W	H'00	H'FE83
	Timer interrupt enable register 3	TIER3	R/W	H'40	H'FE84
	Timer status register 3	TSR3	R/(W)* <sup>2</sup>	H'C0	H'FE85
	Timer counter 3	TCNT3	R/W	H'0000	H'FE86
	Timer general register 3A	TGR3A	R/W	H'FFFF	H'FE88
	Timer general register 3B	TGR3B	R/W	H'FFFF	H'FE8A
	Timer general register 3C	TGR3C	R/W	H'FFFF	H'FE8C
	Timer general register 3D	TGR3D	R/W	H'FFFF	H'FE8E
4	Timer control register 4	TCR4	R/W	H'00	H'FE90
	Timer mode register 4	TMDR4	R/W	H'C0	H'FE91
	Timer I/O control register 4	TIOR4	R/W	H'00	H'FE92
	Timer interrupt enable register 4	TIER4	R/W	H'40	H'FE94
	Timer status register 4	TSR4	R/(W)* <sup>2</sup>	H'C0	H'FE95
	Timer counter 4	TCNT4	R/W	H'0000	H'FE96
	Timer general register 4A	TGR4A	R/W	H'FFFF	H'FE98
	Timer general register 4B	TGR4B	R/W	H'FFFF	H'FE9A
5	Timer control register 5	TCR5	R/W	H'00	H'FEA0
	Timer mode register 5	TMDR5	R/W	H'C0	H'FEA1
	Timer I/O control register 5	TIOR5	R/W	H'00	H'FEA2
	Timer interrupt enable register 5	TIER5	R/W	H'40	H'FEA4
	Timer status register 5	TSR5	R/(W)* <sup>2</sup>	H'C0	H'FEA5
	Timer counter 5	TCNT5	R/W	H'0000	H'FEA6
	Timer general register 5A	TGR5A	R/W	H'FFFF	H'FEA8
	Timer general register 5B	TGR5B	R/W	H'FFFF	H'FEAA
All	Timer start register	TSTR	R/W	H'00	H'FFC0
	Timer synchro register	TSYR	R/W	H'00	H'FFC1
	Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. Lower 16 bits of the address.

2. Can only be written with 0 for flag clearing.

## 7.2 Register Descriptions

### 7.2.1 Timer Control Registers (TCR)

#### Channel 0: TCR0

#### Channel 3: TCR3

Bit	:	7	6	5	4	3	2	1	0
		CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### Channel 1: TCR1

#### Channel 2: TCR2

#### Channel 4: TCR4

#### Channel 5: TCR5

Bit	:	7	6	5	4	3	2	1	0
		—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The TCR registers are 8-bit registers that control the TCNT channels. The TPU has six TCR registers, one for each of channels 0 to 5. The TCR registers are initialized to H'00 by a reset and in hardware standby mode.

TCR register settings should be made only when TCNT operation is stopped.



**Bits 7, 6, and 5—Counter Clear 2, 1, and 0 (CCLR2, CCLR1, CCLR0):** These bits select the TCNT counter clearing source.

Channel	Bit 7 CCLR2	Bit 6 CCLR1	Bit 5 CCLR0	Description
0, 3	0	0	0	TCNT clearing disabled (Initial value)
			1	TCNT cleared by TGRA compare match/input capture
			0	TCNT cleared by TGRB compare match/input capture
	1	0	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation * <sup>1</sup>
			1	TCNT clearing disabled
			0	TCNT cleared by TGRC compare match/input capture * <sup>2</sup>
1	1	0	TCNT cleared by TGRD compare match/input capture * <sup>2</sup>	
		1	TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation * <sup>1</sup>	

Channel	Bit 7 Reserved* <sup>3</sup>	Bit 6 CCLR1	Bit 5 CCLR0	Description
1, 2, 4, 5	0	0	0	TCNT clearing disabled (Initial value)
			1	TCNT cleared by TGRA compare match/input capture
			0	TCNT cleared by TGRB compare match/input capture
			1	TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation * <sup>1</sup>

- Notes:
1. Synchronous operation setting is performed by setting the SYNC bit in TSYR to 1.
  2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.
  3. Bit 7 is reserved in channels 1, 2, 4, and 5. It is always read as 0 and cannot be modified.

**Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0):** These bits select the input clock edge. When the input clock is counted using both edges, the input clock period is halved (e.g.  $\phi/4$  both edges =  $\phi/2$  rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority.

Bit 4 CKEG1	Bit 3 CKEG0	Description	
0	0	Count at rising edge	(Initial value)
	1	Count at falling edge	
1	—	Count at both edges	

Note: Internal clock edge selection is valid when the input clock is  $\phi/4$  or slower. This setting is ignored if the input clock is  $\phi/1$ , or when overflow/underflow of another channel is selected.

**Bits 2, 1, and 0—Time Prescaler 2, 1, and 0 (TPSC2 to TPSC0):** These bits select the TCNT counter clock. The clock source can be selected independently for each channel. Table 7-4 shows the clock sources that can be set for each channel.

**Table 7-4 TPU Clock Sources**

Channel	Internal Clock							External Clock				Overflow/ Underflow on Another Channel
	$\phi/1$	$\phi/4$	$\phi/16$	$\phi/64$	$\phi/256$	$\phi/1024$	$\phi/4096$	TCLKA	TCLKB	TCLKC	TCLKD	
0	○	○	○	○				○	○	○	○	
1	○	○	○	○	○			○	○			○
2	○	○	○	○		○		○	○	○		
3	○	○	○	○	○	○	○	○				
4	○	○	○	○		○		○		○		○
5	○	○	○	○	○			○		○	○	

**Legend**

- : Setting
- Blank : No setting

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
0	0	0	0	Internal clock: counts on $\phi/1$ (Initial value)
			1	Internal clock: counts on $\phi/4$
		1	0	Internal clock: counts on $\phi/16$
			1	Internal clock: counts on $\phi/64$
	1	0	0	External clock: counts on TCLKA pin input
			1	External clock: counts on TCLKB pin input
		1	0	External clock: counts on TCLKC pin input
			1	External clock: counts on TCLKD pin input

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
1	0	0	0	Internal clock: counts on $\phi/1$ (Initial value)
			1	Internal clock: counts on $\phi/4$
		1	0	Internal clock: counts on $\phi/16$
			1	Internal clock: counts on $\phi/64$
	1	0	0	External clock: counts on TCLKA pin input
			1	External clock: counts on TCLKB pin input
		1	0	Internal clock: counts on $\phi/256$
			1	Counts on TCNT2 overflow/underflow

Note: This setting is ignored when channel 1 is in phase counting mode.

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
2	0	0	0	Internal clock: counts on $\phi/1$ (Initial value)
			1	Internal clock: counts on $\phi/4$
		1	0	Internal clock: counts on $\phi/16$
			1	Internal clock: counts on $\phi/64$
	1	0	0	External clock: counts on TCLKA pin input
			1	External clock: counts on TCLKB pin input
		1	0	External clock: counts on TCLKC pin input
			1	Internal clock: counts on $\phi/1024$

Note: This setting is ignored when channel 2 is in phase counting mode.

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
3	0	0	0	Internal clock: counts on $\emptyset/1$ (Initial value)
			1	Internal clock: counts on $\emptyset/4$
		1	0	Internal clock: counts on $\emptyset/16$
			1	Internal clock: counts on $\emptyset/64$
	1	0	0	External clock: counts on TCLKA pin input
			1	Internal clock: counts on $\emptyset/1024$
		1	0	Internal clock: counts on $\emptyset/256$
			1	Internal clock: counts on $\emptyset/4096$

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
4	0	0	0	Internal clock: counts on $\emptyset/1$ (Initial value)
			1	Internal clock: counts on $\emptyset/4$
		1	0	Internal clock: counts on $\emptyset/16$
			1	Internal clock: counts on $\emptyset/64$
	1	0	0	External clock: counts on TCLKA pin input
			1	External clock: counts on TCLKC pin input
		1	0	Internal clock: counts on $\emptyset/1024$
			1	Counts on TCNT5 overflow/underflow

Note: This setting is ignored when channel 4 is in phase counting mode.

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
5	0	0	0	Internal clock: counts on $\emptyset/1$ (Initial value)
			1	Internal clock: counts on $\emptyset/4$
		1	0	Internal clock: counts on $\emptyset/16$
			1	Internal clock: counts on $\emptyset/64$
	1	0	0	External clock: counts on TCLKA pin input
			1	External clock: counts on TCLKC pin input
		1	0	Internal clock: counts on $\emptyset/256$
			1	External clock: counts on TCLKD pin input

Note: This setting is ignored when channel 5 is in phase counting mode.

## 7.2.2 Timer Mode Registers (TMDR)

### Channel 0: TMDR0

### Channel 3: TMDR3

Bit	:	7	6	5	4	3	2	1	0
		—	—	BFB	BFA	MD3	MD2	MD1	MD0
Initial value :		1	1	0	0	0	0	0	0
R/W	:	—	—	R/W	R/W	R/W	R/W	R/W	R/W

### Channel 1: TMDR1

### Channel 2: TMDR2

### Channel 4: TMDR4

### Channel 5: TMDR5

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	MD3	MD2	MD1	MD0
Initial value :		1	1	0	0	0	0	0	0
R/W	:	—	—	—	—	R/W	R/W	R/W	R/W

The TMDR registers are 8-bit readable/writable registers that are used to set the operating mode for each channel. The TPU has six TMDR registers, one for each channel. The TMDR registers are initialized to H'C0 by a reset and in hardware standby mode.

TMDR register settings should be made only when TCNT operation is stopped.

**Bits 7 and 6—Reserved:** Read-only bits, always read as 1.

**Bit 5—Buffer Operation B (BFB):** Specifies whether TGRB is to operate in the normal way, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD input capture/output compare is not generated.

In channels 1, 2, 4, and 5, which have no TGRD, bit 5 is reserved. It is always read as 0 and cannot be modified.

#### Bit 5

BFB	Description	
0	TGRB operates normally	(Initial value)
1	TGRB and TGRD used together for buffer operation	

**Bit 4—Buffer Operation A (BFA):** Specifies whether TGRA is to operate in the normal way, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated.

In channels 1, 2, 4, and 5, which have no TGRC, bit 4 is reserved. It is always read as 0 and cannot be modified.

Bit 4 BFA	Description	
0	TGRA operates normally	(Initial value)
1	TGRA and TGRC used together for buffer operation	

**Bits 3 to 0—Modes 3 to 0 (MD3 to MD0):** These bits are used to set the timer operating mode.

Bit 3 MD3* <sup>1</sup>	Bit 2 MD2* <sup>2</sup>	Bit 1 MD1	Bit 0 MD0	Description			
0	0	0	0	Normal operation	(Initial value)		
			1	Reserved			
	1	0	1	0	PWM mode 1		
				1	PWM mode 2		
			0	1	0	Phase counting mode 1	
					1	Phase counting mode 2	
		0	1	0	Phase counting mode 3		
				1	Phase counting mode 4		
1	*	*	*	—			

\*: Don't care

- Notes: 1. MD3 is a reserved bit. In a write, it should always be written with 0.  
 2. Phase counting mode cannot be set for channels 0 and 3. For these channels, 0 should always be written to MD2.

### 7.2.3 Timer I/O Control Registers (TIOR)

**Channel 0: TIOR0H**

**Channel 1: TIOR1**

**Channel 2: TIOR2**

**Channel 3: TIOR3H**

**Channel 4: TIOR4**

**Channel 5: TIOR5**

Bit	:	7	6	5	4	3	2	1	0
		IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Channel 0: TIOR0L**

**Channel 3: TIOR3L**

Bit	:	7	6	5	4	3	2	1	0
		IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Note:** When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

The TIOR registers are 8-bit registers that control the TGR registers. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. The TIOR registers are initialized to H'00 by a reset and in hardware standby mode.

Care is required since TIOR is affected by the TMDR setting. The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified.

**Bits 7 to 4— I/O Control B3 to B0 (IOB3 to IOB0)****I/O Control D3 to D0 (IOD3 to IOD0):**

Bits IOB3 to IOB0 specify the function of TGRB.

Bits IOD3 to IOD0 specify the function of TGRD.

Channel	Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description
0	0	0	0	0	TGR0B is Output disabled (Initial value)
				1	output compare register
				0	Initial output is 0 output
				1	Toggle output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1 output
				0	0 output at compare match
				1	1 output at compare match
	1	0	0	0	TGR0B is Capture input source is TIOCB0 pin
				1	input capture register
				*	Input capture at rising edge
				*	Input capture at falling edge
1	*	*	*	Input capture at both edges	
			*	Capture input source is channel 1/count clock	
			*	Input capture at TCNT1 count-up/count-down* <sup>1</sup>	
			*		

\*: Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000 and  $\emptyset/1$  is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.



Channel	Bit 7 IOD3	Bit 6 IOD2	Bit 5 IOD1	Bit 4 IOD0	Description
0	0	0	0	0	TGR0D is Output disabled (Initial value)
				1	output
				0	Initial output is 0
				1	compare register* <sup>2</sup>
	1	0	0	0	0 output at compare match
				1	1 output at compare match
				0	Toggle output at compare match
				1	Output disabled
1	0	0	0	Initial output is 1	
			1	0 output at compare match	
			0	1 output at compare match	
			1	Toggle output at compare match	
1	0	0	0	TGR0D is Capture input	
			1	input source is	
			*	capture register* <sup>2</sup>	
			*	TIOCD0 pin	
1	0	0	*	Input capture at rising edge	
			*	Input capture at falling edge	
1	0	0	*	Input capture at both edges	
			*	Input capture at TCNT1 count-up/count-down* <sup>1</sup>	
1	0	0	*	Capture input source is channel 1/count clock	
			*		

\*: Don't care

- Notes:
1. When bits TPSC2 to TPSC0 in TCR1 are set to B'000 and  $\emptyset/1$  is used as the TCNT1 count clock, this setting is invalid and input capture is not generated.
  2. When the BFB bit in TMDR0 is set to 1 and TGR0D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Channel	Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description	
1	0	0	0	0	TGR1B is Output disabled (Initial value)	
				1	output	
				0	Initial output is 0	
				1	output at compare match	
	1	0	0	0	TGR1B is output compare register	
				1	0	1 output at compare match
				1	1	Toggle output at compare match
				0	Output disabled	
	1	0	0	0	TGR1B is output compare register	
				1	1	Initial output is 1
				0	0	0 output at compare match
				1	1	1 output at compare match
1	0	0	0	TGR1B is capture input source is TIOCB1 pin		
			1	*	Input capture at rising edge	
			1	*	Input capture at falling edge	
			1	*	Input capture at both edges	
1	0	0	0	TGR1B is capture input source is TGR0C compare match/ input capture		
			1	*	Input capture at generation of TGR0C compare match/ input capture	
			1	*	TGR0C compare match/ input capture	
			1	*	Input capture	

\*: Don't care

Channel	Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description	
2	0	0	0	0	TGR2B is Output disabled (Initial value)	
				1	output	
				0	Initial output is 0	
				1	output at compare match	
	1	0	0	0	TGR2B is output compare register	
				1	0	1 output at compare match
				1	1	Toggle output at compare match
				0	Output disabled	
	1	0	0	0	TGR2B is output compare register	
				1	1	Initial output is 1
				0	0	0 output at compare match
				1	1	1 output at compare match
1	*	0	0	TGR2B is capture input source is TIOCB2 pin		
			1	*	Input capture at rising edge	
			1	*	Input capture at falling edge	
			1	*	Input capture at both edges	
1	*	0	0	TGR2B is capture input source is TGR0C compare match/ input capture		
			1	*	Input capture at generation of TGR0C compare match/ input capture	
			1	*	TGR0C compare match/ input capture	
			1	*	Input capture	

\*: Don't care

Channel	Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description		
3	0	0	0	0	TGR3B is Output disabled (Initial value)		
				1	output	Initial output is 0	0 output at compare match
				0	compare register	output	1 output at compare match
				1		Toggle output at compare match	
	1	0	0	0	Output disabled		
				1	Initial output is 1	0 output at compare match	
				0	output	1 output at compare match	
				1		Toggle output at compare match	
				0	TGR3B is Capture input	Input capture at rising edge	
				1	input source is TIOCB3 pin	Input capture at falling edge	
1	0	0	0	capture register	Input capture at both edges		
			1	*	Capture input source is channel 4/count clock	Input capture at TCNT4 count-up/count-down* <sup>1</sup>	
			*	*			
			*	*			

\*: Don't care

Note: 1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and  $\emptyset/1$  is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.

Channel	Bit 7 IOD3	Bit 6 IOD2	Bit 5 IOD1	Bit 4 IOD0	Description
3	0	0	0	0	Output disabled (Initial value)
				1	Initial output is 0
				0	0 output at compare match
				1	1 output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1
				0	0 output at compare match
				1	1 output at compare match
1	0	0	0	Capture input source is TIOCD3 pin	
			1	Input capture at rising edge	
			*	Input capture at falling edge	
			*	Input capture at both edges	
1	*	*	*	Capture input source is channel 4/count clock	Input capture at TCNT4 count-up/count-down* <sup>1</sup>

\*: Don't care

- Notes:
1. When bits TPSC2 to TPSC0 in TCR4 are set to B'000 and  $\emptyset/1$  is used as the TCNT4 count clock, this setting is invalid and input capture is not generated.
  2. When the BFB bit in TMDR3 is set to 1 and TGR3D is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Channel	Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description			
4	0	0	0	0	TGR4B is Output disabled (Initial value)			
				1	output	Initial output is 0	0 output at compare match	
				0	compare register	0 output	1 output at compare match	
			1	0	0	1	Toggle output at compare match	
					1	0	Output disabled	
					1	0	Initial output is 1	0 output at compare match
	1	0	0	0	TGR4B is Capture input source is TIOCB4 pin			
				1	capture register	1 output at compare match	Input capture at rising edge	
				1	*	Input capture at falling edge		
			1	*	*	0	Capture input source is TGR3C compare match/ input capture	Input capture at both edges
						0	0	Input capture at generation of TGR3C compare match/ input capture
						0	0	1 output at compare match

\*: Don't care

Channel	Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description			
5	0	0	0	0	TGR5B is Output disabled (Initial value)			
				1	output	Initial output is 0	0 output at compare match	
				0	compare register	0 output	1 output at compare match	
			1	0	0	0	1	Toggle output at compare match
						1	0	Output disabled
						1	0	Initial output is 1
	1	*	0	0	TGR5B is Capture input source is TIOCB5 pin			
				1	capture register	1 output at compare match	Input capture at rising edge	
				1	*	Input capture at falling edge		
			1	*	0	0	Capture input source is TGR3C compare match/ input capture	Input capture at both edges
						0	0	Input capture at generation of TGR3C compare match/ input capture
						0	0	1 output at compare match

\*: Don't care

**Bits 3 to 0— I/O Control A3 to A0 (IOA3 to IOA0)****I/O Control C3 to C0 (IOC3 to IOC0):**

IOA3 to IOA0 specify the function of TGRA.

IOC3 to IOC0 specify the function of TGRC.

Channel	Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description
0	0	0	0	0	TGR0A is Output disabled (Initial value)
				1	output compare register
				0	Initial output is 0 output
				1	Toggle output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1 output
				0	0 output at compare match
				1	1 output at compare match
	1	0	0	0	TGR0A is Capture input source is TIOCA0 pin
				1	input capture register
				*	Input capture at rising edge
				*	Input capture at falling edge
1	*	*	*	Input capture at both edges	
			*	Input capture at TCNT1 count-up/count-down	
			*	Capture input source is channel 1/ count clock	
			*		

\*: Don't care

Channel	Bit 3 IOC3	Bit 2 IOC2	Bit 1 IOC1	Bit 0 IOC0	Description
0	0	0	0	0	TGR0C is Output disabled (Initial value)
				1	output Initial output is 0
				0	compare 0 output at compare match
				1	register* <sup>1</sup> 1 output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1
				0	output 0 output at compare match
				1	1 output at compare match
	1	0	0	0	TGR0C is Capture input
				1	input source is
				*	capture TIOCC0 pin
				*	register* <sup>1</sup> Input capture at both edges
1	*	*	0	Capture input	
			1	source is channel	
			0	Input capture at TCNT1	
			1	count-up/count-down	
1	*	*	0	1/count clock	
			1		
			0		
			1		

\*: Don't care

Note: 1. When the BFA bit in TMDR0 is set to 1 and TGR0C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Channel	Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description
1	0	0	0	0	TGR1A is Output disabled (Initial value)
				1	output compare register
				0	Initial output is 0 output
				1	Toggle output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1 output
				0	0 output at compare match
				1	1 output at compare match
	1	0	0	0	TGR1A is Capture input source is TIOCA1 pin
				1	capture register
				*	Input capture at rising edge
				*	Input capture at falling edge
1	*	*	*	Input capture at both edges	
			*	Input capture at generation of channel 0/TGR0A compare match/ input capture	

\*: Don't care

Channel	Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description
2	0	0	0	0	TGR2A is Output disabled (Initial value)
				1	output compare register
				0	Initial output is 0 output
				1	Toggle output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1 output
				0	0 output at compare match
				1	1 output at compare match
	1	*	0	0	TGR2A is Capture input source is TIOCA2 pin
				1	capture register
				*	Input capture at rising edge
				*	Input capture at falling edge
1	*	0	*	Input capture at both edges	

\*: Don't care



Channel	Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description		
3	0	0	0	0	TGR3A is Output disabled (Initial value)		
				1	output	Initial output is 0	0 output at compare match
				0	compare register	output	1 output at compare match
				1		Toggle output at compare match	
	1	0	0	0	Output disabled		
				1	Initial output is 1	0 output at compare match	
				0	output	1 output at compare match	
				1		Toggle output at compare match	
				0	TGR3A is Capture input	Input capture at rising edge	
				1	input source is TIOCA3 pin	Input capture at falling edge	
1	0	0	0	capture register	Input capture at both edges		
			1	*	Input capture at TCNT4 count-up/count-down		
1	1	*	*	Capture input source is channel 4/count clock	Input capture at TCNT4 count-up/count-down		

\*: Don't care

Channel	Bit 3 IOC3	Bit 2 IOC2	Bit 1 IOC1	Bit 0 IOC0	Description
3	0	0	0	0	Output disabled (Initial value)
				1	Initial output is 0 output
				0	0 output at compare match
				1	1 output at compare match
	1	0	0	0	Output disabled
				1	Initial output is 1 output
				0	0 output at compare match
				1	1 output at compare match
				0	Toggle output at compare match
				1	Toggle output at compare match
1	0	0	0	TGR3C is input capture register*1	
			1	Capture input source is TIOCC3 pin	
			*	Input capture at rising edge	
			*	Input capture at falling edge	
1	*	*	*	Input capture at both edges	
			*	Input capture at TCNT4 count-up/count-down 4/count clock	

\*: Don't care

Note: 1. When the BFA bit in TMDR3 is set to 1 and TGR3C is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Channel	Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description		
4	0	0	0	0	TGR4A is Output disabled (Initial value)		
				1	output	Initial output is 0	0 output at compare match
				0	compare register	output	1 output at compare match
				1		Toggle output at compare match	
	1	0	0	0	Output disabled		
				1	Initial output is 1	0 output at compare match	
				0	output	1 output at compare match	
				1	Toggle output at compare match		
1	0	0	0	TGR4A is Capture input			
			1	input source is TIOCA4 pin	Input capture at rising edge		
			0	capture register	Input capture at falling edge		
			1	*	Input capture at both edges		
	1	*	*	Capture input source is TGR3A	Input capture at generation of TGR3A compare match/input capture		

\*: Don't care

Channel	Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description		
5	0	0	0	0	TGR5A is Output disabled (Initial value)		
				1	output	Initial output is 0	0 output at compare match
				0	compare register	output	1 output at compare match
				1		Toggle output at compare match	
	1	0	0	0	Output disabled		
				1	Initial output is 1	0 output at compare match	
				0	output	1 output at compare match	
				1	Toggle output at compare match		
1	*	0	0	TGR5A is Capture input			
			1	input source is TIOCA5 pin	Input capture at rising edge		
			0	capture register	Input capture at falling edge		
		1	*	Input capture at both edges			

\*: Don't care

## 7.2.4 Timer Interrupt Enable Registers (TIER)

### Channel 0: TIER0

### Channel 3: TIER3

Bit	:	7	6	5	4	3	2	1	0
		TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA
Initial value :		0	1	0	0	0	0	0	0
R/W	:	R/W	—	—	R/W	R/W	R/W	R/W	R/W

### Channel 1: TIER1

### Channel 2: TIER2

### Channel 4: TIER4

### Channel 5: TIER5

Bit	:	7	6	5	4	3	2	1	0
		TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA
Initial value :		0	1	0	0	0	0	0	0
R/W	:	R/W	—	R/W	R/W	—	—	R/W	R/W

The TIER registers are 8-bit registers that control enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel. The TIER registers are initialized to H'40 by a reset and in hardware standby mode.

**Bit 7—A/D Conversion Start Request Enable (TTGE):** Enables or disables generation of A/D conversion start requests by TGRA input capture/compare match.

Bit 7 TTGE	Description
0	A/D conversion start request generation disabled (Initial value)
1	A/D conversion start request generation enabled

**Bit 6—Reserved:** Read-only bit, always read as 1.

**Bit 5—Underflow Interrupt Enable (TCIEU):** Enables or disables interrupt requests (TCIU) by the TCFU bit when the TCFU bit in TSR is set to 1 in channels 1 and 2.

In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.

**Bit 5**

<b>TCIEU</b>	<b>Description</b>	
0	Interrupt requests (TCIU) by TCFU disabled	(Initial value)
1	Interrupt requests (TCIU) by TCFU enabled	

**Bit 4—Overflow Interrupt Enable (TCIEV):** Enables or disables interrupt requests (TCIV) by the TCFV bit when the TCFV bit in TSR is set to 1.

**Bit 4**

<b>TCIEV</b>	<b>Description</b>	
0	Interrupt requests (TCIV) by TCFV disabled	(Initial value)
1	Interrupt requests (TCIV) by TCFV enabled	

**Bit 3—TGR Interrupt Enable D (TGIED):** Enables or disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.

**Bit 3**

<b>TGIED</b>	<b>Description</b>	
0	Interrupt requests (TGID) by TGFD disabled	(Initial value)
1	Interrupt requests (TGID) by TGFD enabled	

**Bit 2—TGR Interrupt Enable C (TGIEC):** Enables or disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

**Bit 2**

<b>TGIEC</b>	<b>Description</b>	
0	Interrupt requests (TGIC) by TGFC disabled	(Initial value)
1	Interrupt requests (TGIC) by TGFC enabled	

**Bit 1—TGR Interrupt Enable B (TGIEB):** Enables or disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1.

Bit 1	
TGIEB	Description
0	Interrupt requests (TGIB) by TGFB disabled (Initial value)
1	Interrupt requests (TGIB) by TGFB enabled

**Bit 0—TGR Interrupt Enable A (TGIEA):** Enables or disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1.

Bit 0	
TGIEA	Description
0	Interrupt requests (TGIA) by TGFA disabled (Initial value)
1	Interrupt requests (TGIA) by TGFA enabled

## 7.2.5 Timer Status Registers (TSR)

### Channel 0: TSR0

### Channel 3: TSR3

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA
Initial value :		1	1	0	0	0	0	0	0
R/W	:	—	—	—	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: \* Only 0 can be written, to clear the flag.

### Channel 1: TSR1

### Channel 2: TSR2

### Channel 4: TSR4

### Channel 5: TSR5

Bit	:	7	6	5	4	3	2	1	0
		TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA
Initial value :		1	1	0	0	0	0	0	0
R/W	:	R	—	R/(W)*	R/(W)*	—	—	R/(W)*	R/(W)*

Note: \* Only 0 can be written, to clear the flag.

The TSR registers are 8-bit registers that indicate the status of each channel. The TPU has six TSR registers, one for each channel. The TSR registers are initialized to H'00 by a reset and in hardware standby mode.

**Bit 7—Count Direction Flag (TCFD):** Status flag that shows the direction in which TCNT counts in channels 1, 2, 4, and 5.

In channels 0 and 3, bit 7 is reserved. It is always read as 1 and cannot be modified.

Bit 7 TCFD	Description	
0	TCNT counts down	
1	TCNT counts up	(Initial value)

**Bit 6—Reserved:** Read-only bit, always read as 1.

**Bit 5—Underflow Flag (TCFU):** Status flag that indicates that TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting mode.

In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.

Bit 5 TCFU	Description	
0	[Clearing condition] When 0 is written to TCFU after reading TCFU = 1	(Initial value)
1	[Setting condition] When the TCNT value underflows (changes from H'0000 to H'FFFF)	

**Bit 4—Overflow Flag (TCFV):** Status flag that indicates that TCNT overflow has occurred.

Bit 4 TCFV	Description	
0	[Clearing condition] When 0 is written to TCFV after reading TCFV = 1	(Initial value)
1	[Setting condition] When the TCNT value overflows (changes from H'FFFF to H'0000)	

**Bit 3—Input Capture/Output Compare Flag D (TGFD):** Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.

<b>Bit 3 TGFD</b>	<b>Description</b>
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• When DTC is activated by TGID interrupt while DISEL bit of MRB in DTC is 0</li><li>• When 0 is written to TGFD after reading TGFD = 1</li></ul>
1	[Setting conditions] <ul style="list-style-type: none"><li>• When TCNT = TGRD while TGRD is functioning as output compare register</li><li>• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register</li></ul>

**Bit 2—Input Capture/Output Compare Flag C (TGFC):** Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

<b>Bit 2 TGFC</b>	<b>Description</b>
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• When DTC is activated by TGIC interrupt while DISEL bit of MRB in DTC is 0</li><li>• When 0 is written to TGFC after reading TGFC = 1</li></ul>
1	[Setting conditions] <ul style="list-style-type: none"><li>• When TCNT = TGRC while TGRC is functioning as output compare register</li><li>• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register</li></ul>



**Bit 1—Input Capture/Output Compare Flag B (TGFB):** Status flag that indicates the occurrence of TGRB input capture or compare match.

Bit 1 TGFB	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• When DTC is activated by TGIB interrupt while DISEL bit of MRB in DTC is 0</li><li>• When 0 is written to TGFB after reading TGFB = 1</li></ul>
1	[Setting conditions] <ul style="list-style-type: none"><li>• When TCNT = TGRB while TGRB is functioning as output compare register</li><li>• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register</li></ul>

**Bit 0—Input Capture/Output Compare Flag A (TGFA):** Status flag that indicates the occurrence of TGRA input capture or compare match.

Bit 0 TGFA	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• When DTC is activated by TGIA interrupt while DISEL bit of MRB in DTC is 0</li><li>• When DMAC is activated by TGIA interrupt while DTA bit of DMABCR in DMAC is 1</li><li>• When 0 is written to TGFA after reading TGFA = 1</li></ul>
1	[Setting conditions] <ul style="list-style-type: none"><li>• When TCNT = TGRA while TGRA is functioning as output compare register</li><li>• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register</li></ul>

## 7.2.6 Timer Counters (TCNT)

**Channel 0: TCNT0 (up-counter)**

**Channel 1: TCNT1 (up/down-counter\*)**

**Channel 2: TCNT2 (up/down-counter\*)**

**Channel 3: TCNT3 (up-counter)**

**Channel 4: TCNT4 (up/down-counter\*)**

**Channel 5: TCNT5 (up/down-counter\*)**

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* These counters can be used as up/down-counters only in phase counting mode or when counting overflow/underflow on another channel. In other cases they function as up-counters.

The TCNT registers are 16-bit counters. The TPU has six TCNT counters, one for each channel.

The TCNT counters are initialized to H'0000 by a reset and in hardware standby mode.

The TCNT counters cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

## 7.2.7 Timer General Registers (TGR)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The TGR registers are 16-bit registers with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channels 0 and 3 and two each for channels 1, 2, 4, and 5. TGRC and TGRD for channels 0 and 3 can also be designated for operation as buffer registers\*. The TGR registers are initialized to H'FFFF by a reset and in hardware standby mode.

The TGR registers cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

Note: \* TGR buffer register combinations are TGRA—TGRC and TGRB—TGRD.

### 7.2.8 Timer Start Register (TSTR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	CST5	CST4	CST3	CST2	CST1	CST0
Initial value	:	0	0	0	0	0	0	0	0
R/W	:	—	—	R/W	R/W	R/W	R/W	R/W	R/W

TSTR is an 8-bit readable/writable register that selects operation/stoppage for channels 0 to 5. TSTR is initialized to H'00 by a reset, and in hardware standby mode. When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

**Bits 7 and 6—Reserved:** Must always be written with 0.

**Bits 5 to 0—Counter Start 5 to 0 (CST5 to CST0):** These bits select operation or stoppage for TCNT.

Bit n	Description
CSTn	
0	TCNTn count operation is stopped (Initial value)
1	TCNTn performs count operation

n = 5 to 0

Note: If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value.

### 7.2.9 Timer Synchro Register (TSYR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
Initial value	:	0	0	0	0	0	0	0	0
R/W	:	—	—	R/W	R/W	R/W	R/W	R/W	R/W

TSYR is an 8-bit readable/writable register that selects independent operation or synchronous operation for the channel 0 to 4 TCNT counters. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

TSYR is initialized to H'00 by a reset and in hardware standby mode.

**Bits 7 and 6—Reserved:** Must always be written with 0.

**Bits 5 to 0—Timer Synchro 5 to 0 (SYNC5 to SYNC0):** These bits select whether operation is independent of or synchronized with other channels.

When synchronous operation is selected, synchronous presetting of multiple channels\*<sup>1</sup>, and synchronous clearing through counter clearing on another channel\*<sup>2</sup> are possible.

Bit n SYNCn	Description
0	TCNTn operates independently (TCNT presetting/clearing is unrelated to other channels) (Initial value)
1	TCNTn performs synchronous operation TCNT synchronous presetting/synchronous clearing is possible

n = 5 to 0

- Notes: 1. To set synchronous operation, the SYNC bits for at least two channels must be set to 1.  
2. To set synchronous clearing, in addition to the SYNC bit, the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.

### 7.2.10 Module Stop Control Register (MSTPCR)

MSTPCRH								MSTPCRL								
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP13 bit in MSTPCR is set to 1, TPU operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 13—Module Stop (MSTP13):** Specifies the TPU module stop mode.

Bit 13 MSTP13	Description
0	TPU module stop mode cleared
1	TPU module stop mode set (Initial value)

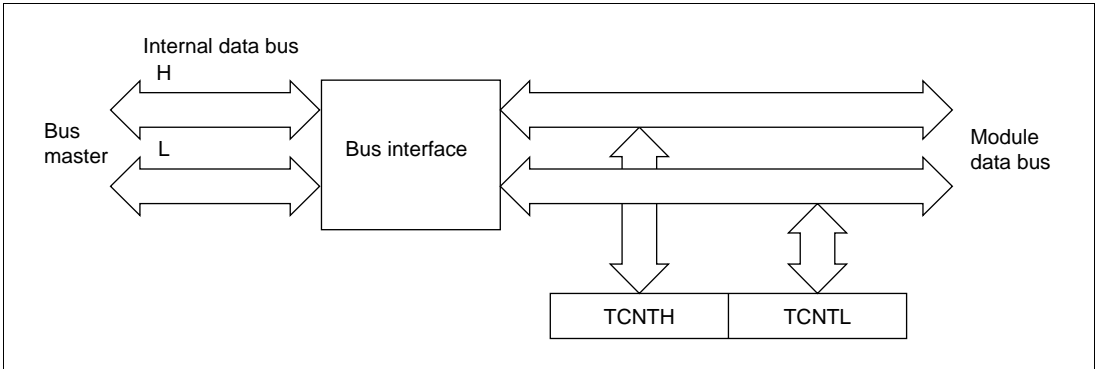
## 7.3 Interface to Bus Master

### 7.3.1 16-Bit Registers

TCNT and TGR are 16-bit registers. As the data bus to the bus master is 16 bits wide, these registers can be read and written to in 16-bit units.

These registers cannot be read or written to in 8-bit units; 16-bit access must always be used.

An example of 16-bit register access operation is shown in figure 7-2.

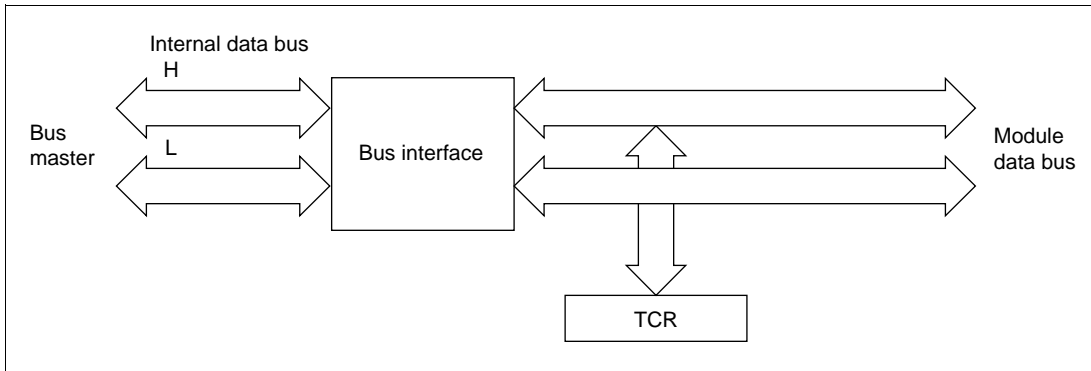


**Figure 7-2 16-Bit Register Access Operation [Bus Master ↔ TCNT (16 Bits)]**

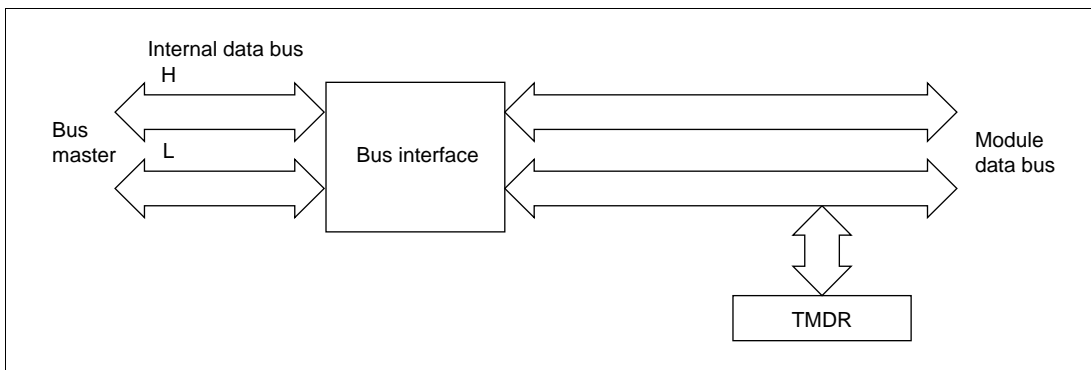
### 7.3.2 8-Bit Registers

Registers other than TCNT and TGR are 8-bit. As the data bus to the CPU is 16 bits wide, these registers can be read and written to in 16-bit units. They can also be read and written to in 8-bit units.

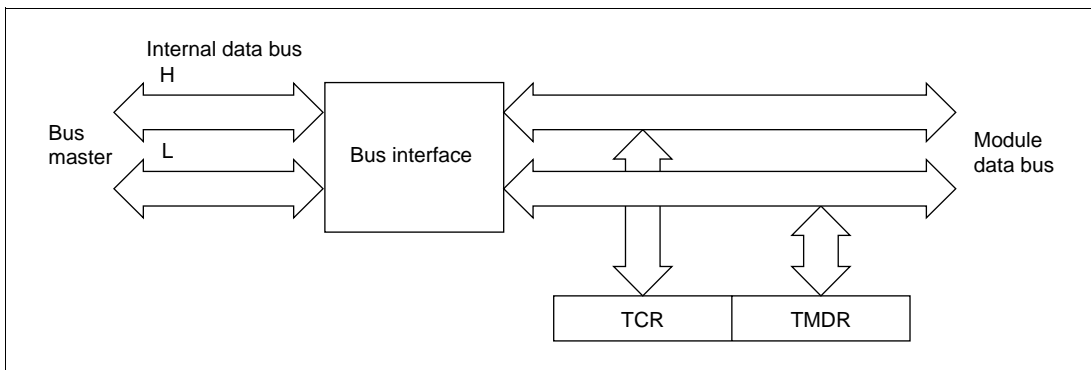
Examples of 8-bit register access operation are shown in figures 7-3, 7-4, and 7-5.



**Figure 7-3 8-Bit Register Access Operation [Bus Master ↔ TCR (Upper 8 Bits)]**



**Figure 7-4 8-Bit Register Access Operation [Bus Master ↔ TMDR (Lower 8 Bits)]**



**Figure 7-5 8-Bit Register Access Operation [Bus Master ↔ TCR and TMDR (16 Bits)]**

## 7.4 Operation

### 7.4.1 Overview

Operation in each mode is outlined below.

**Normal Operation:** Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, synchronous counting, and external event counting.

Each TGR can be used as an input capture register or output compare register.

**Synchronous Operation:** When synchronous operation is designated for a channel, TCNT for that channel performs synchronous presetting. That is, when TCNT for a channel designated for synchronous operation is rewritten, the TCNT counters for the other channels are also rewritten at the same time. Synchronous clearing of the TCNT counters is also possible by setting the timer synchronization bits in TSYR for channels designated for synchronous operation.

#### Buffer Operation

- When TGR is an output compare register  
When a compare match occurs, the value in the buffer register for the relevant channel is transferred to TGR.
- When TGR is an input capture register  
When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in TGR is transferred to the buffer register.

**Cascaded Operation:** The channel 1 counter (TCNT1) and channel 2 counter (TCNT2), or the channel 4 counter (TCNT4) and channel 5 counter (TCNT5), can be connected together to operate as a 32-bit counter.

**PWM Mode:** In this mode, a PWM waveform is output. The output level can be set by means of TIOR. A PWM waveform with a duty of between 0% and 100% can be output, according to the setting of each TGR register.

**Phase Counting Mode:** In this mode, TCNT is incremented or decremented by detecting the phases of two clocks input from the external clock input pins in channels 1, 2, 4, and 5. When phase counting mode is set, the corresponding TCLK pin functions as the clock pin, and TCNT performs up/down-counting.

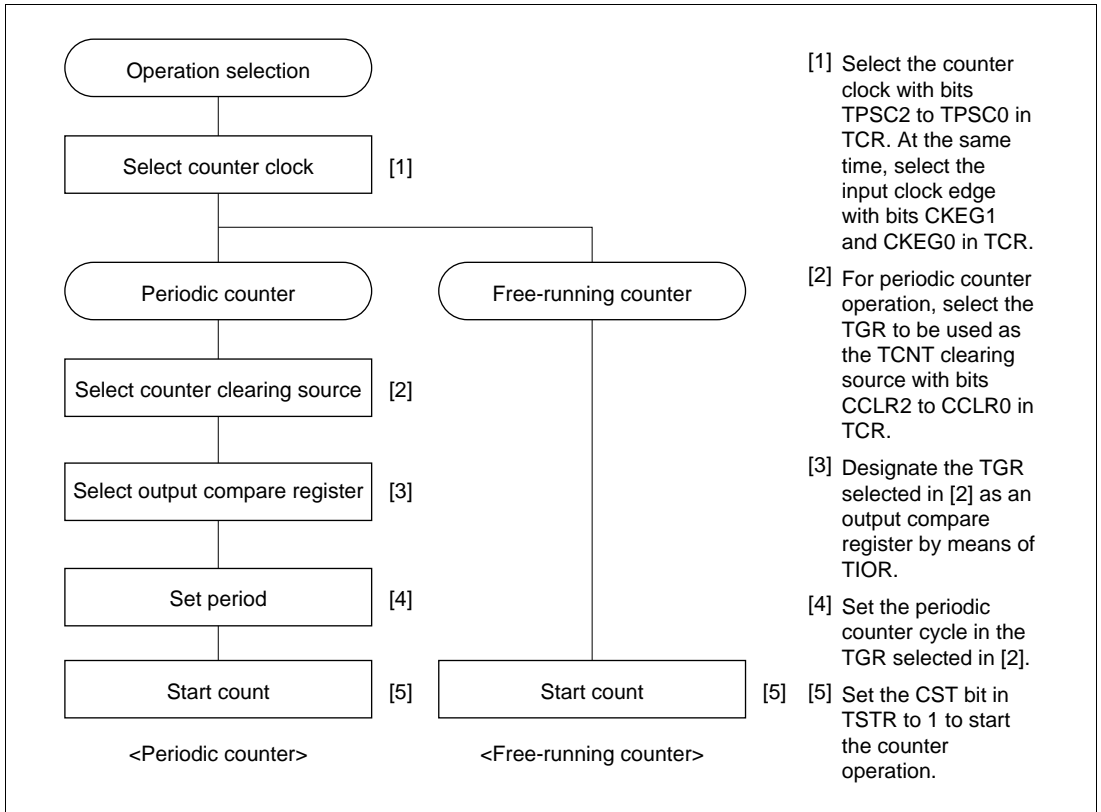
This can be used for two-phase encoder pulse input.

## 7.4.2 Basic Functions

**Counter Operation:** When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

- Example of count operation setting procedure

Figure 7-6 shows an example of the count operation setting procedure.



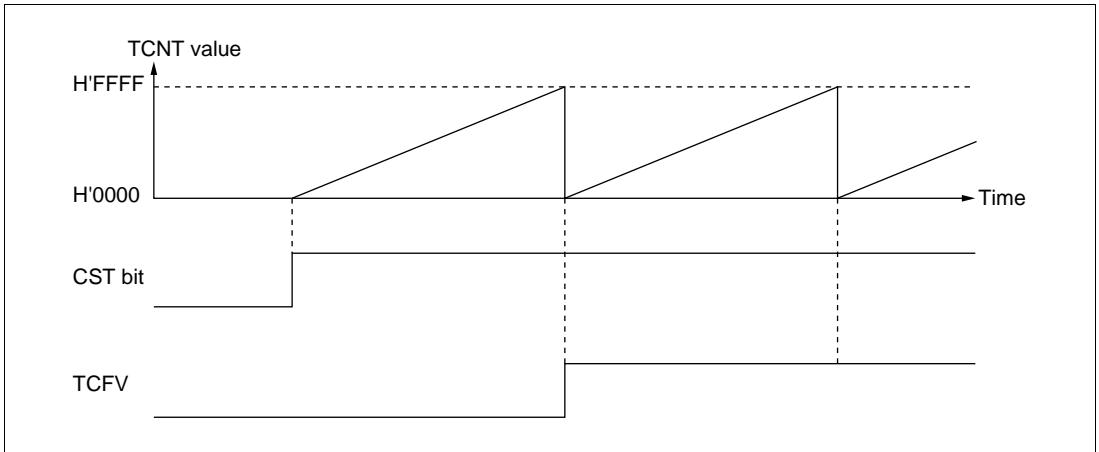
**Figure 7-6 Example of Counter Operation Setting Procedure**



- Free-running count operation and periodic count operation

Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000.

Figure 7-7 illustrates free-running counter operation.

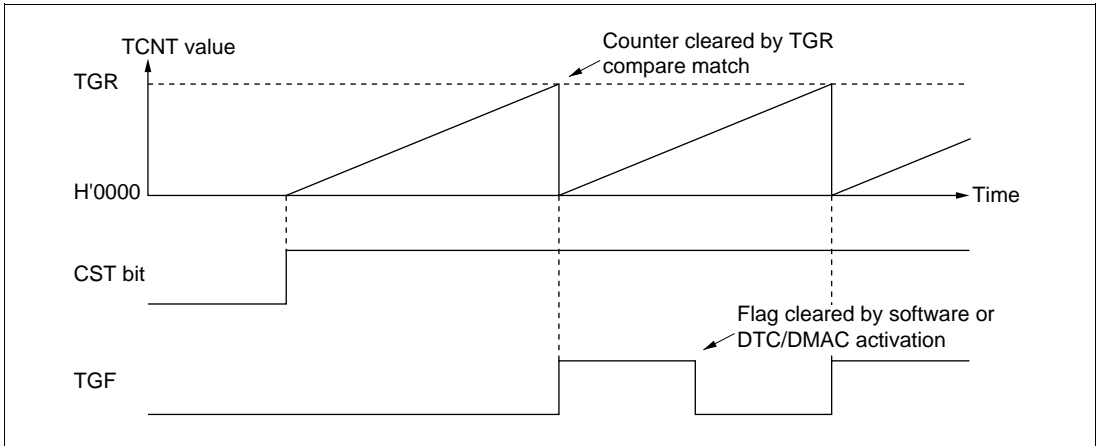


**Figure 7-7 Free-Running Counter Operation**

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts up-count operation as a periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000.

Figure 7-8 illustrates periodic counter operation.

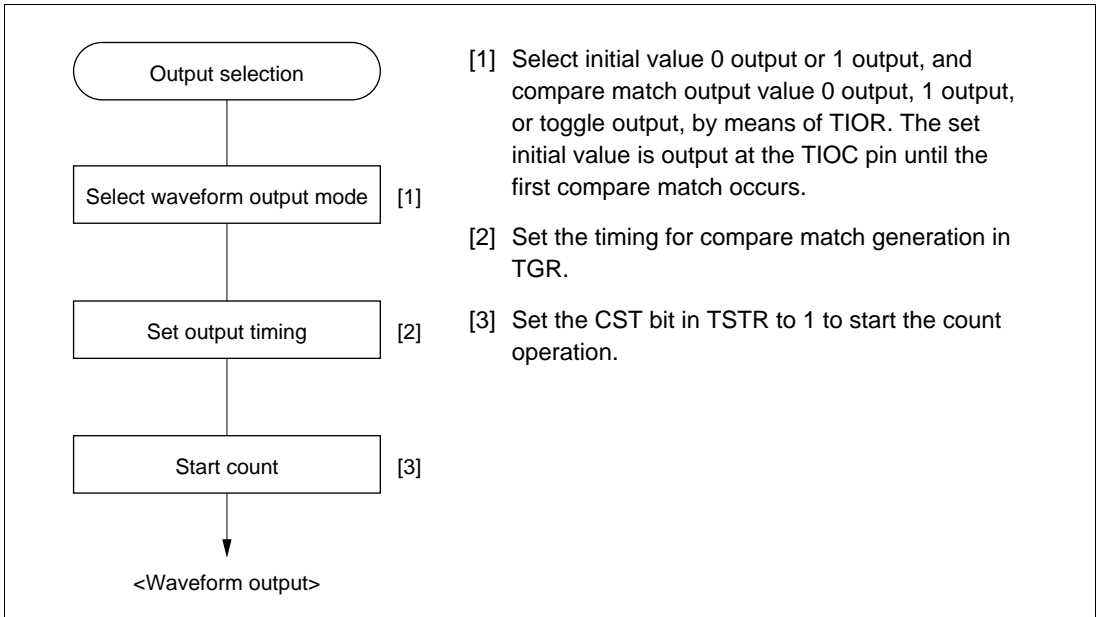


**Figure 7-8 Periodic Counter Operation**

**Waveform Output by Compare Match:** The TPU can perform 0, 1, or toggle output from the corresponding output pin using compare match.

- Example of setting procedure for waveform output by compare match

Figure 7-9 shows an example of the setting procedure for waveform output by compare match

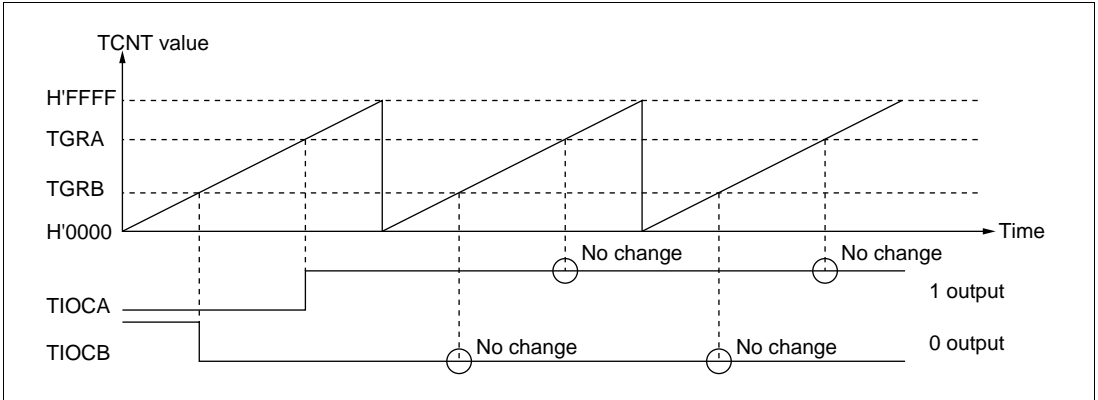


**Figure 7-9 Example of Setting Procedure for Waveform Output by Compare Match**

- Examples of waveform output operation

Figure 7-10 shows an example of 0 output/1 output.

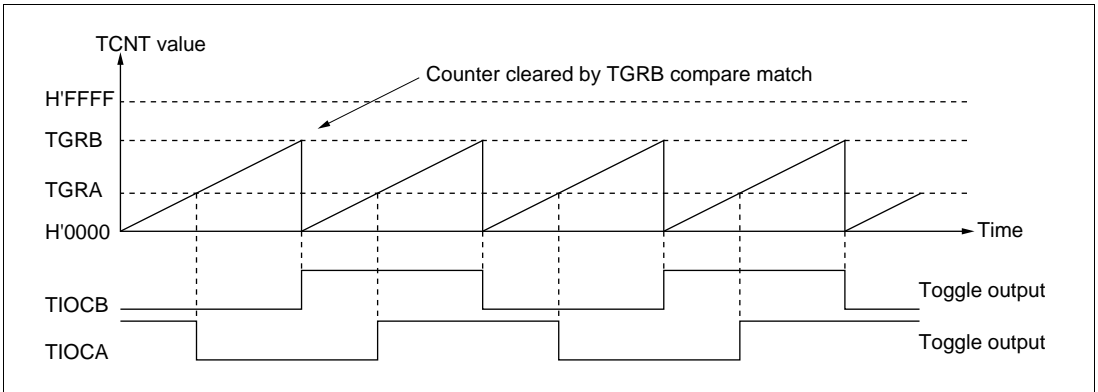
In this example TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level coincide, the pin level does not change.



**Figure 7-10 Example of 0 Output/1 Output Operation**

Figure 7-11 shows an example of toggle output.

In this example TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.



**Figure 7-11 Example of Toggle Output Operation**

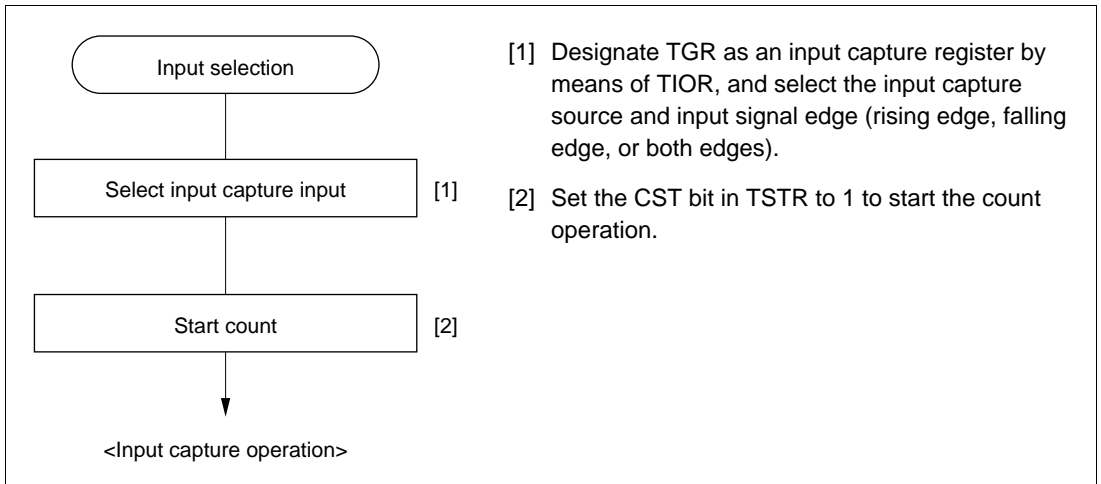
**Input Capture Function:** The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detected edge. For channels 0, 1, 3, and 4, it is also possible to specify another channel's counter input clock or compare match signal as the input capture source.

Note: When another channel's counter input clock is used as the input capture input for channels 0 and 3,  $\phi/1$  should not be selected as the counter input clock used for input capture input. Input capture will not be generated if  $\phi/1$  is selected.

- Example of input capture operation setting procedure

Figure 7-12 shows an example of the input capture operation setting procedure.

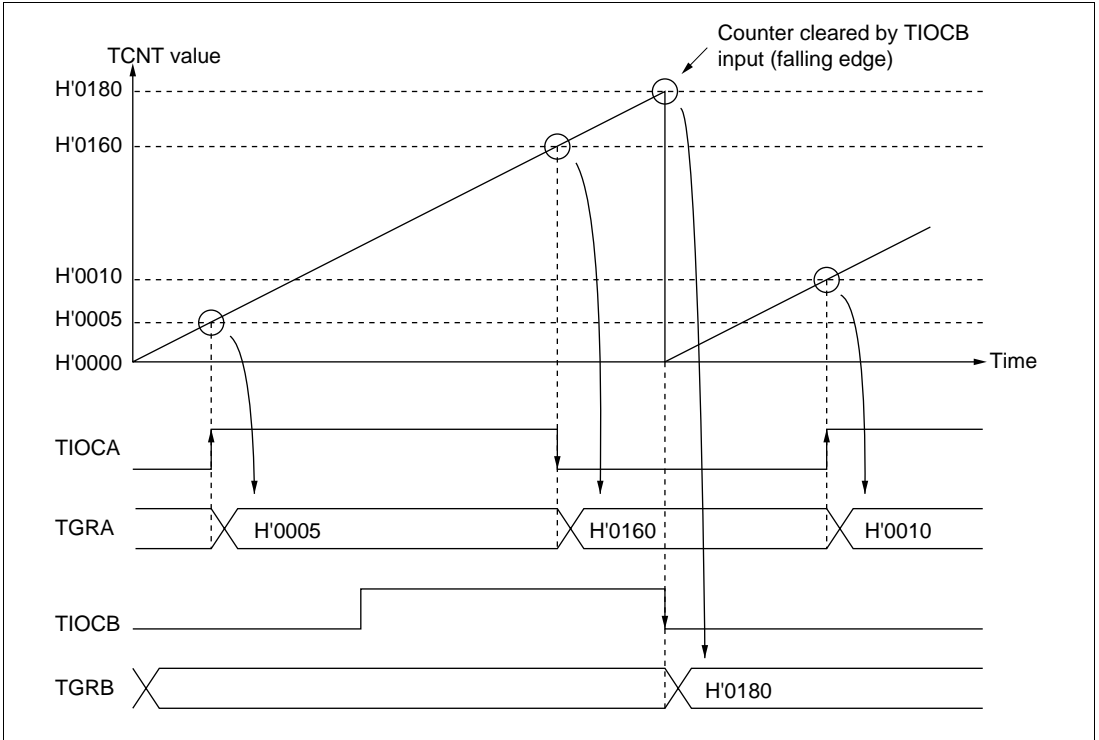


**Figure 7-12 Example of Input Capture Operation Setting Procedure**

- Example of input capture operation

Figure 7-13 shows an example of input capture operation.

In this example both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.



**Figure 7-13 Example of Input Capture Operation**

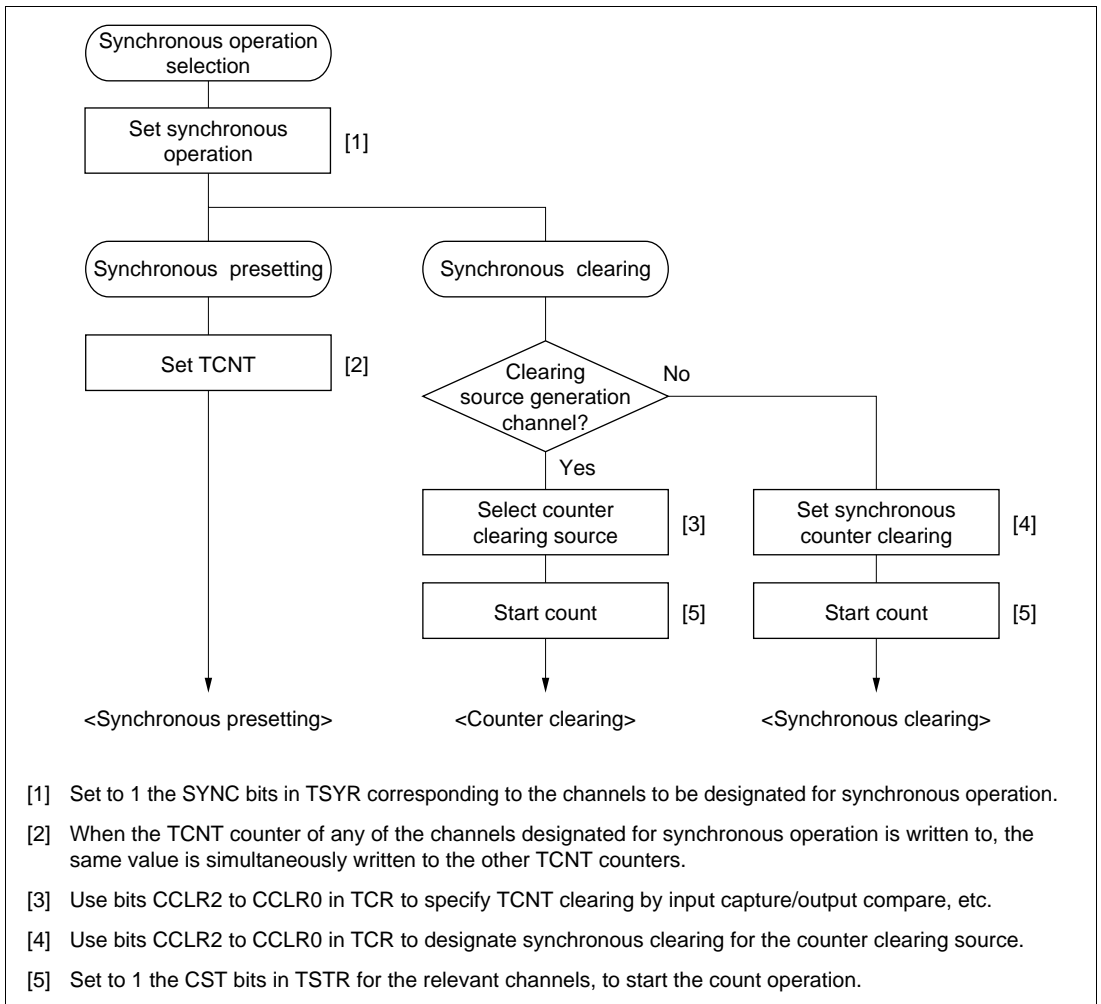
### 7.4.3 Synchronous Operation

In synchronous operation, the values in a number of TCNT counters can be rewritten simultaneously (synchronous presetting). Also, a number of TCNT counters can be cleared simultaneously by making the appropriate setting in TCR (synchronous clearing).

Synchronous operation enables TGR to be incremented with respect to a single time base.

Channels 0 to 5 can all be designated for synchronous operation.

**Example of Synchronous Operation Setting Procedure:** Figure 7-14 shows an example of the synchronous operation setting procedure.



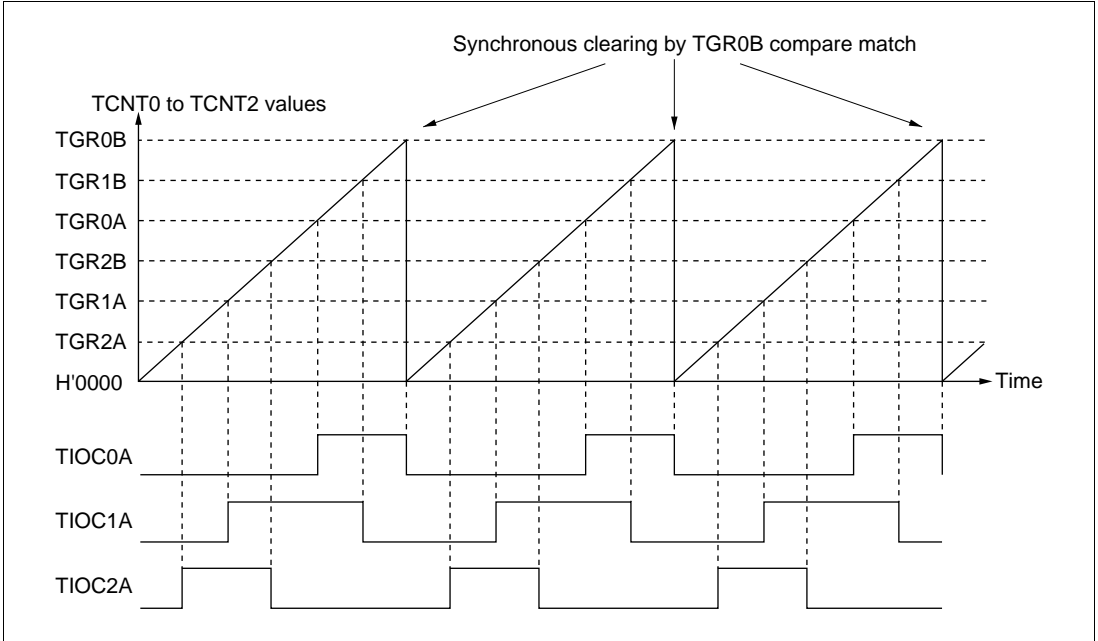
**Figure 7-14 Example of Synchronous Operation Setting Procedure**

**Example of Synchronous Operation:** Figure 7-15 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGR0B compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source.

Three-phase PWM waveforms are output from pins TIOC0A, TIOC1A, and TIOC2A. At this time, synchronous presetting, and synchronous clearing by TGR0B compare match, is performed for channel 0 to 2 TCNT counters, and the data set in TGR0B is used as the PWM cycle.

For details of PWM modes, see section 7.4.6, PWM Modes.



**Figure 7-15 Example of Synchronous Operation**

## 7.4.4 Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

Buffer operation differs depending on whether TGR has been designated as an input capture register or as a compare match register.

Table 7-5 shows the register combinations used in buffer operation.

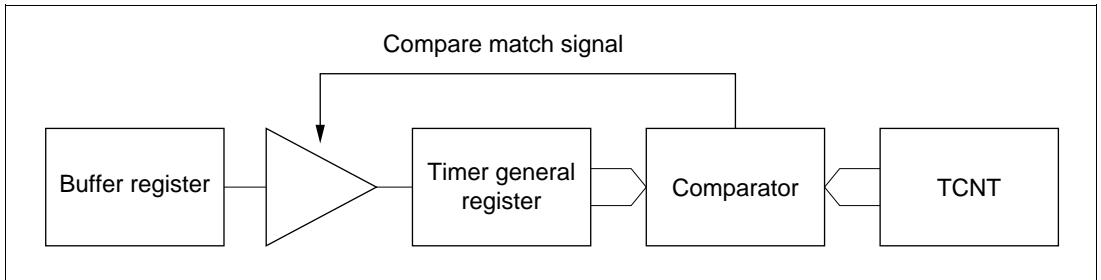
**Table 7-5 Register Combinations in Buffer Operation**

Channel	Timer General Register	Buffer Register
0	TGR0A	TGR0C
	TGR0B	TGR0D
3	TGR3A	TGR3C
	TGR3B	TGR3D

- When TGR is an output compare register

When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.

This operation is illustrated in figure 7-16.



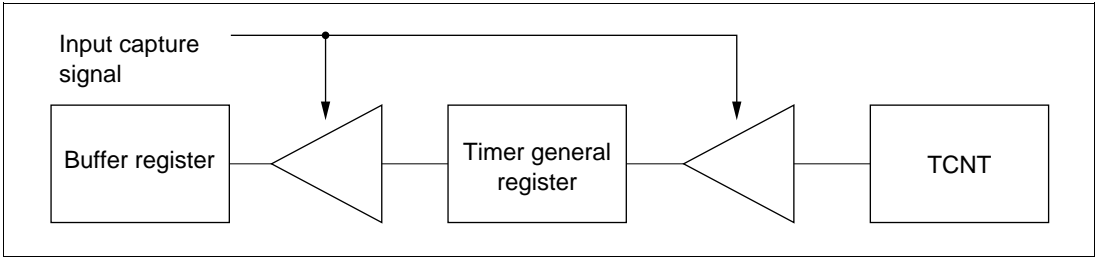
**Figure 7-16 Compare Match Buffer Operation**



- When TGR is an input capture register

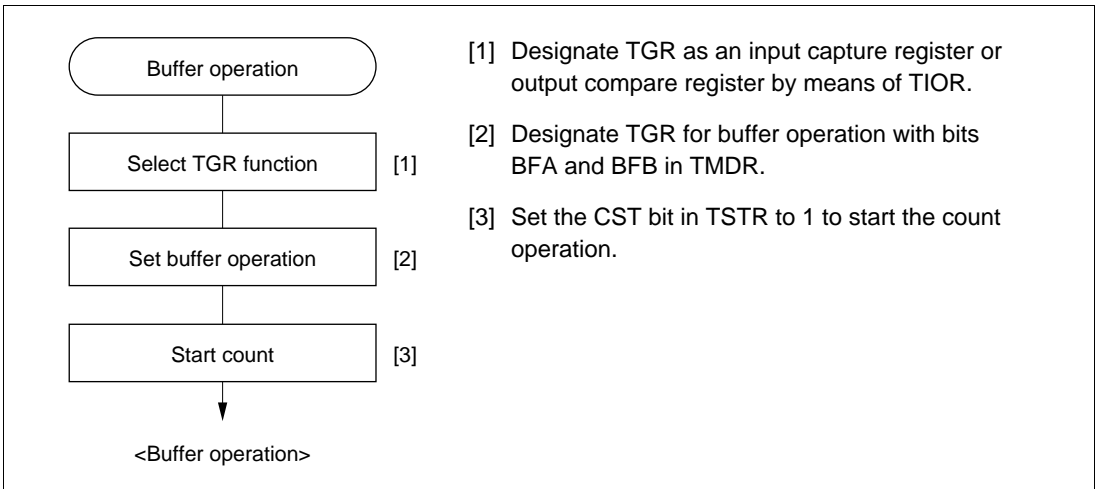
When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in the timer general register is transferred to the buffer register.

This operation is illustrated in figure 7-17.



**Figure 7-17 Input Capture Buffer Operation**

**Example of Buffer Operation Setting Procedure:** Figure 7-18 shows an example of the buffer operation setting procedure.



**Figure 7-18 Example of Buffer Operation Setting Procedure**

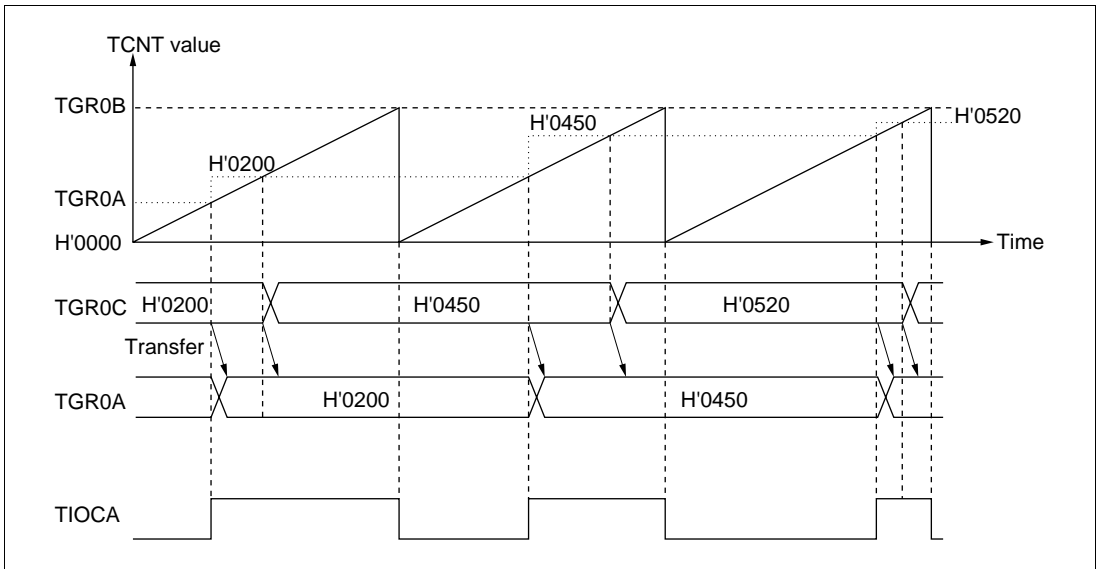
## Examples of Buffer Operation

- When TGR is an output compare register

Figure 7-19 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B.

As buffer operation has been set, when compare match A occurs the output changes and the value in buffer register TGRB is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

For details of PWM modes, see section 7.4.6, PWM Modes.



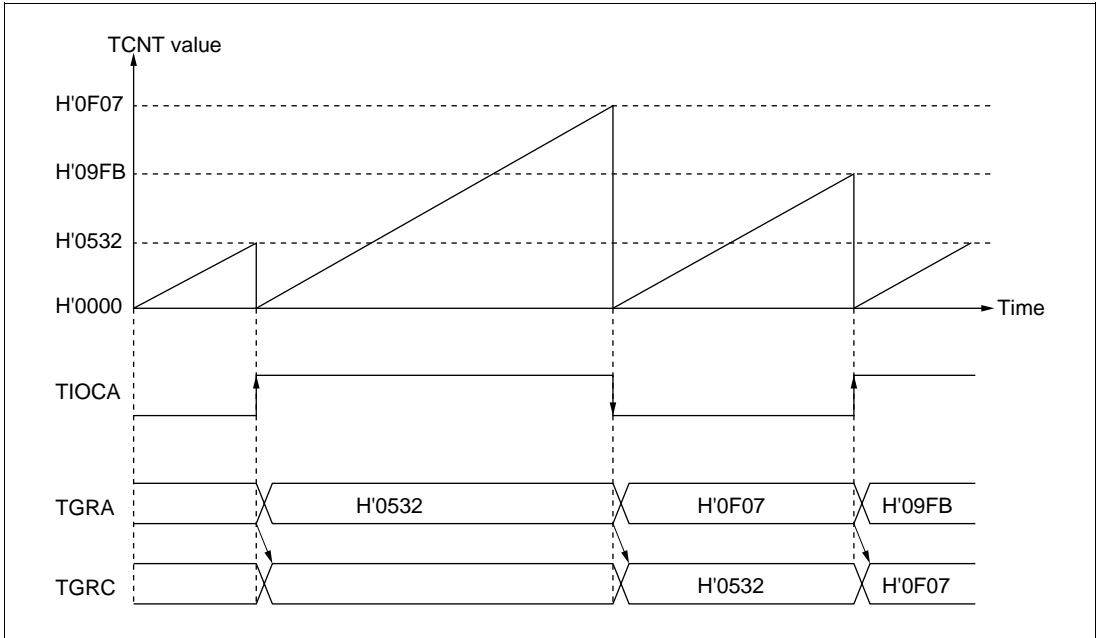
**Figure 7-19 Example of Buffer Operation (1)**

- When TGR is an input capture register

Figure 7-20 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC.

Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge.

As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.



**Figure 7-20 Example of Buffer Operation (2)**

## 7.4.5 Cascaded Operation

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter.

This function works by counting the channel 1 (channel 4) counter clock upon overflow/underflow of TCNT2 (TCNT5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

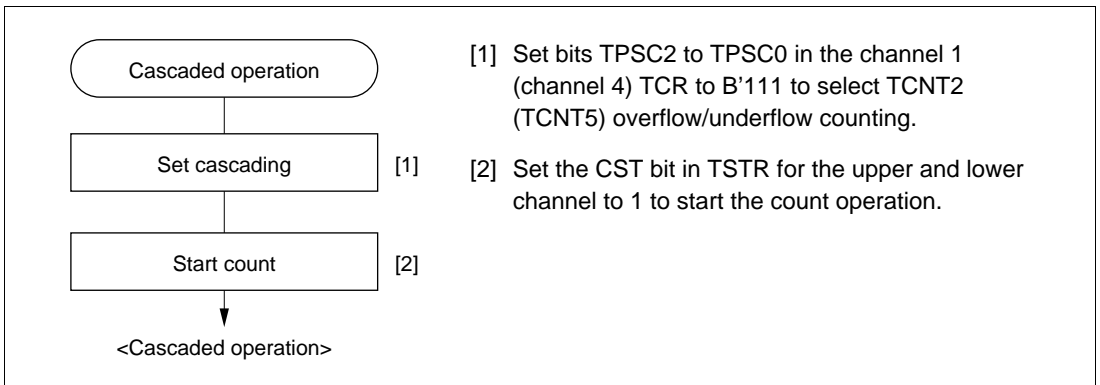
Table 7-6 shows the register combinations used in cascaded operation.

Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

**Table 7-6 Cascaded Combinations**

Combination	Upper 16 Bits	Lower 16 Bits
Channels 1 and 2	TCNT1	TCNT2
Channels 4 and 5	TCNT4	TCNT5

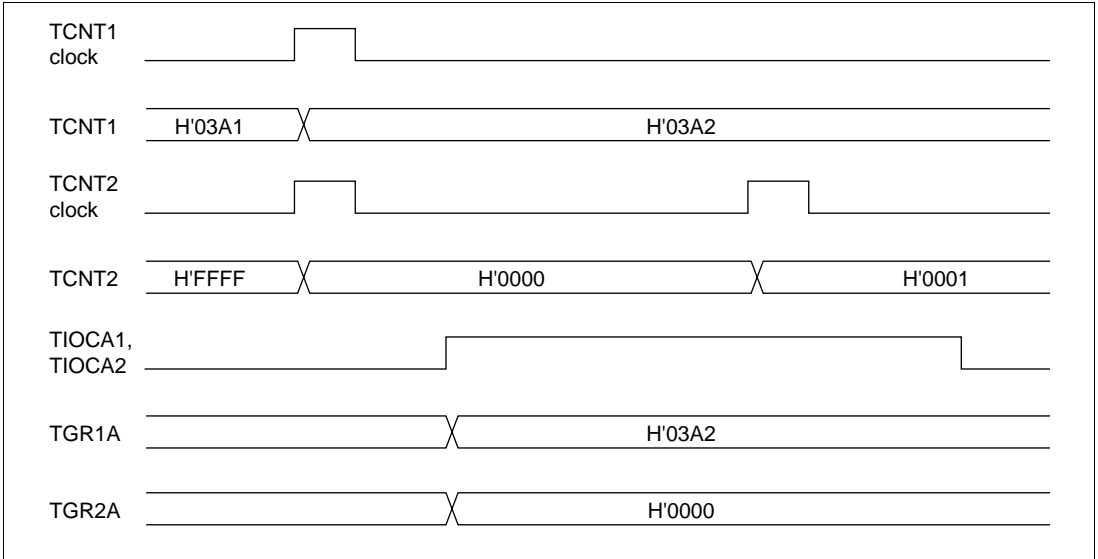
**Example of Cascaded Operation Setting Procedure:** Figure 7-21 shows an example of the setting procedure for cascaded operation.



**Figure 7-21 Cascaded Operation Setting Procedure**

**Examples of Cascaded Operation:** Figure 7-22 illustrates the operation when counting upon TCNT2 overflow/underflow has been set for TCNT1, TGR1A and TGR2A have been designated as input capture registers, and TIOC pin rising edge has been selected.

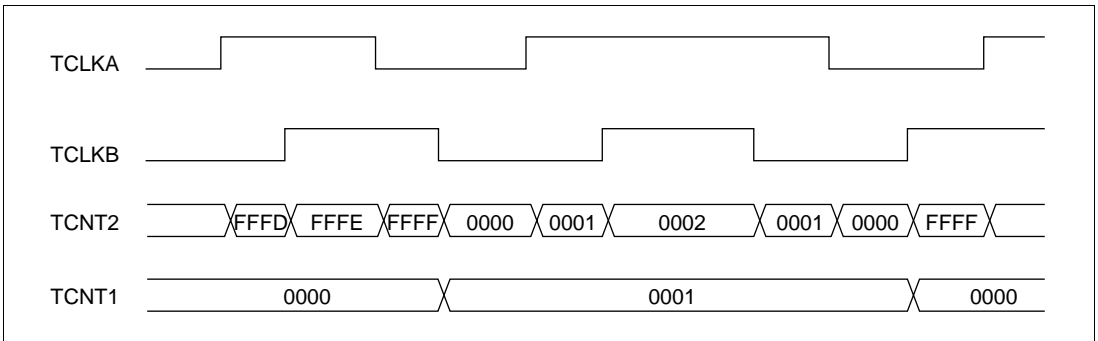
When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGR1A, and the lower 16 bits to TGR2A.



**Figure 7-22 Example of Cascaded Operation (1)**

Figure 7-23 illustrates the operation when counting upon TCNT2 overflow/underflow has been set for TCNT1, and phase counting mode has been designated for channel 2.

TCNT1 is incremented by TCNT2 overflow and decremented by TCNT2 underflow.



**Figure 7-23 Example of Cascaded Operation (2)**

## 7.4.6 PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0, 1, or toggle output can be selected as the output level in response to compare match of each TGR.

Designating TGR compare match as the counter clearing source enables the period to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible.

There are two PWM modes, as described below.

- PWM mode 1

PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The output specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR is output from the TIOCA and TIOCC pins at compare matches A and C, and the output specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR is output at compare matches B and D. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs.

In PWM mode 1, a maximum 8-phase PWM output is possible.

- PWM mode 2

PWM output is generated using one TGR as the period register and the others as duty registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronization register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the period and duty registers are identical, the output value does not change when a compare match occurs.

In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

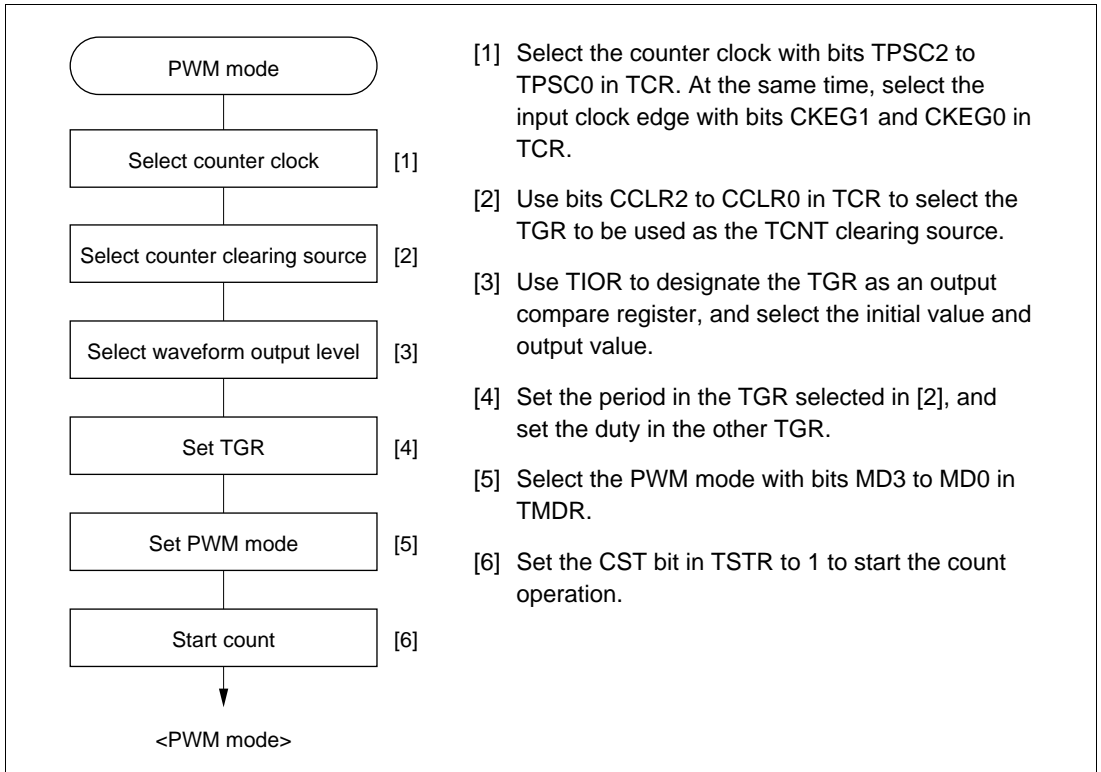
The correspondence between PWM output pins and registers is shown in table 7-7.

**Table 7-7 PWM Output Registers and Output Pins**

Channel	Registers	Output Pins	
		PWM Mode 1	PWM Mode 2
0	TGR0A	TIOCA0	TIOCA0
	TGR0B		TIOCB0
	TGR0C	TIOCC0	TIOCC0
	TGR0D		TIOCD0
1	TGR1A	TIOCA1	TIOCA1
	TGR1B		TIOCB1
2	TGR2A	TIOCA2	TIOCA2
	TGR2B		TIOCB2
3	TGR3A	TIOCA3	TIOCA3
	TGR3B		TIOCB3
	TGR3C	TIOCC3	TIOCC3
	TGR3D		TIOCD3
4	TGR4A	TIOCA4	TIOCA4
	TGR4B		TIOCB4
5	TGR5A	TIOCA5	TIOCA5
	TGR5B		TIOCB5

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the period is set.

**Example of PWM Mode Setting Procedure:** Figure 7-24 shows an example of the PWM mode setting procedure.



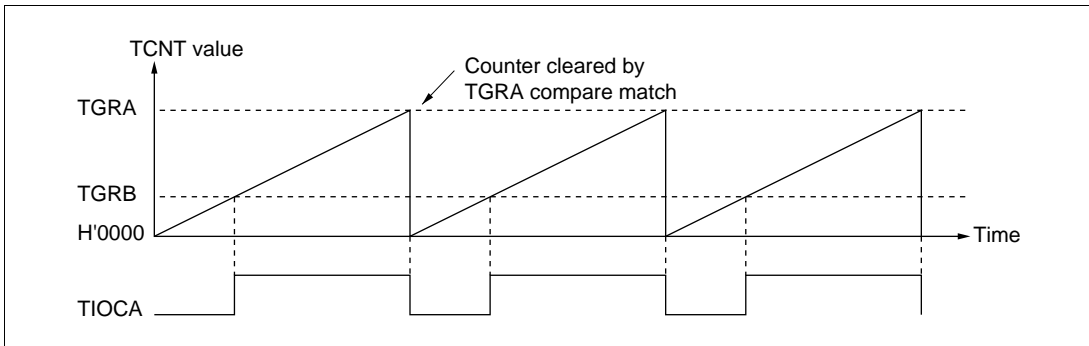
**Figure 7-24 Example of PWM Mode Setting Procedure**

**Examples of PWM Mode Operation:** Figure 7-25 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value.

In this case, the value set in TGRA is used as the period, and the value set in TGRB as the duty.



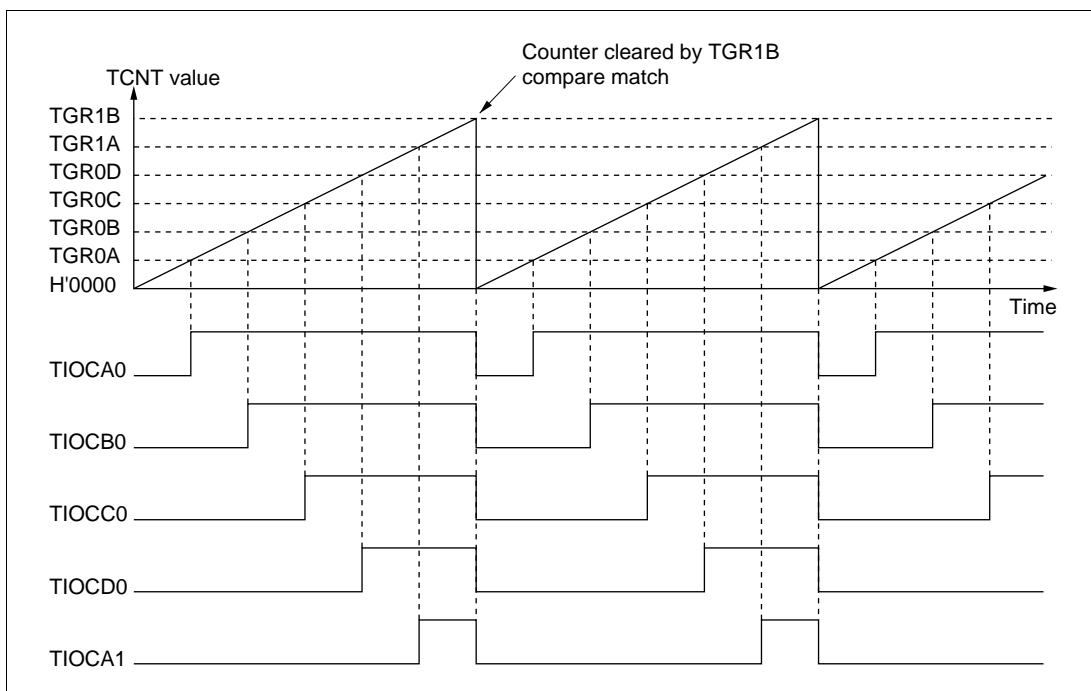


**Figure 7-25 Example of PWM Mode Operation (1)**

Figure 7-26 shows an example of PWM mode 2 operation.

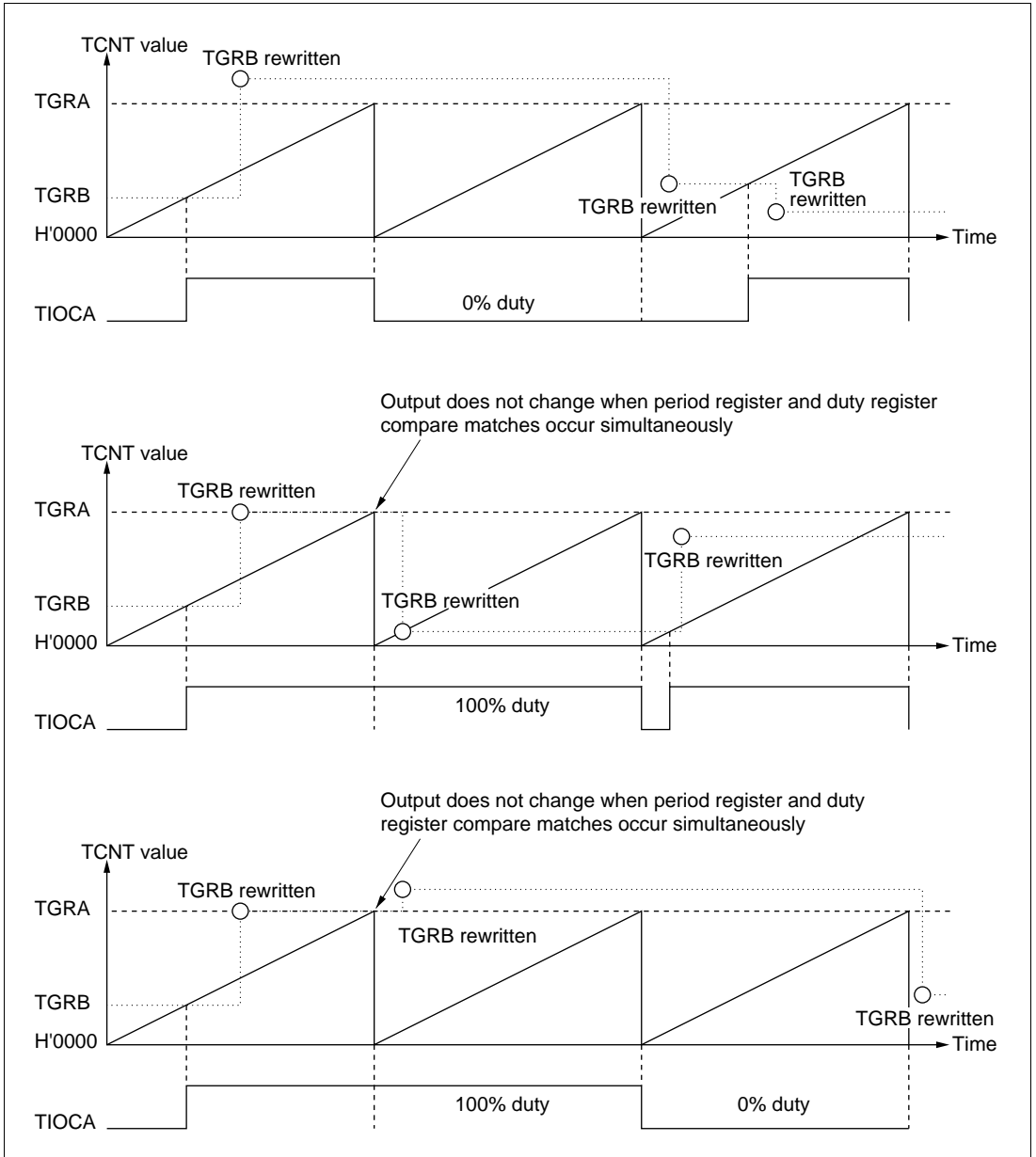
In this example, synchronous operation is designated for channels 0 and 1, TGR1B compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGR0A to TGR0D, TGR1A), to output a 5-phase PWM waveform.

In this case, the value set in TGR1B is used as the period, and the values set in the other TGR registers as the duty.



**Figure 7-26 Example of PWM Mode Operation (2)**

Figure 7-27 shows examples of PWM waveform output with 0% duty and 100% duty in PWM mode.



**Figure 7-27 Examples of PWM Mode Operation (3)**

### 7.4.7 Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1, 2, 4, and 5.

When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used.

When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set.

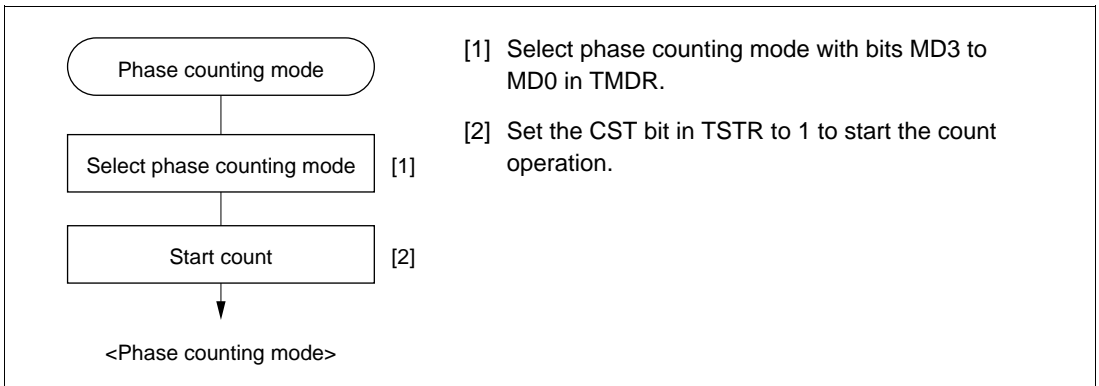
The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 7-8 shows the correspondence between external clock pins and channels.

**Table 7-8 Phase Counting Mode Clock Input Pins**

Channels	External Clock Pins	
	A-Phase	B-Phase
When channel 1 or 5 is set to phase counting mode	TCLKA	TCLKB
When channel 2 or 4 is set to phase counting mode	TCLKC	TCLKD

**Example of Phase Counting Mode Setting Procedure:** Figure 7-28 shows an example of the phase counting mode setting procedure.

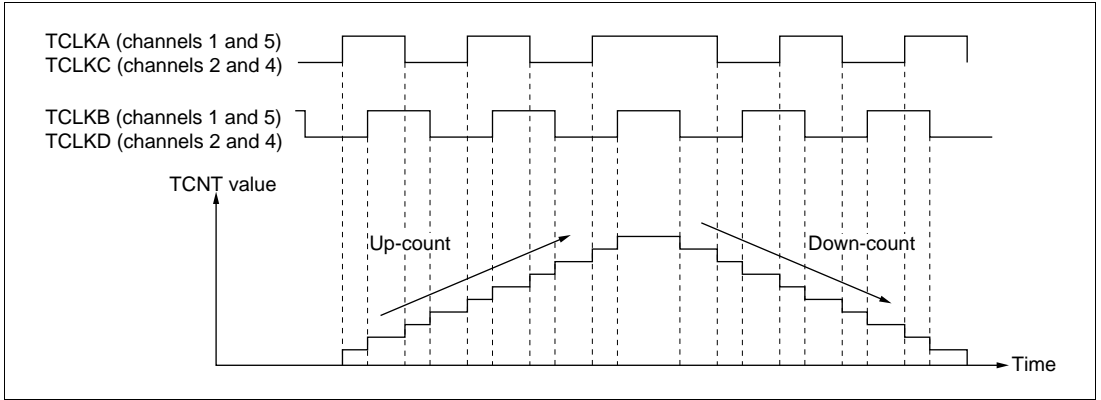


**Figure 7-28 Example of Phase Counting Mode Setting Procedure**

**Examples of Phase Counting Mode Operation:** In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

- Phase counting mode 1

Figure 7-29 shows an example of phase counting mode 1 operation, and table 7-9 summarizes the TCNT up/down-count conditions.



**Figure 7-29 Example of Phase Counting Mode 1 Operation**

**Table 7-9 Up/Down-Count Conditions in Phase Counting Mode 1**

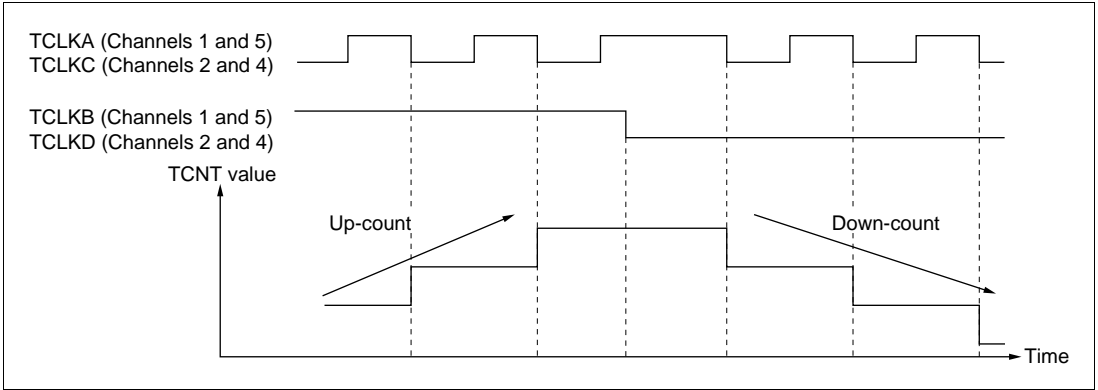
TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level		Up-count
Low level		
	Low level	
	High level	
High level		Down-count
Low level		
	High level	
	Low level	

**Legend**

- : Rising edge
- : Falling edge

- Phase counting mode 2

Figure 7-30 shows an example of phase counting mode 2 operation, and table 7-10 summarizes the TCNT up/down-count conditions.



**Figure 7-30 Example of Phase Counting Mode 2 Operation**

**Table 7-10 Up/Down-Count Conditions in Phase Counting Mode 2**

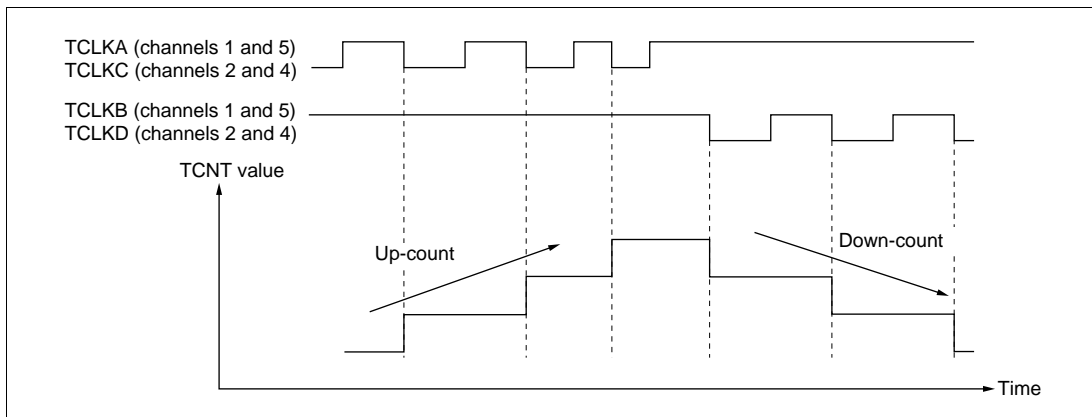
TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level		Don't care
Low level		Don't care
	Low level	Don't care
	High level	Up-count
High level		Don't care
Low level		Don't care
	High level	Don't care
	Low level	Down-count

**Legend**

- : Rising edge
- : Falling edge

- Phase counting mode 3

Figure 7-31 shows an example of phase counting mode 3 operation, and table 7-11 summarizes the TCNT up/down-count conditions.



**Figure 7-31 Example of Phase Counting Mode 3 Operation**

**Table 7-11 Up/Down-Count Conditions in Phase Counting Mode 3**

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level		Don't care
Low level		Don't care
	Low level	Don't care
	High level	Up-count
High level		Down-count
Low level		Don't care
	High level	Don't care
	Low level	Don't care

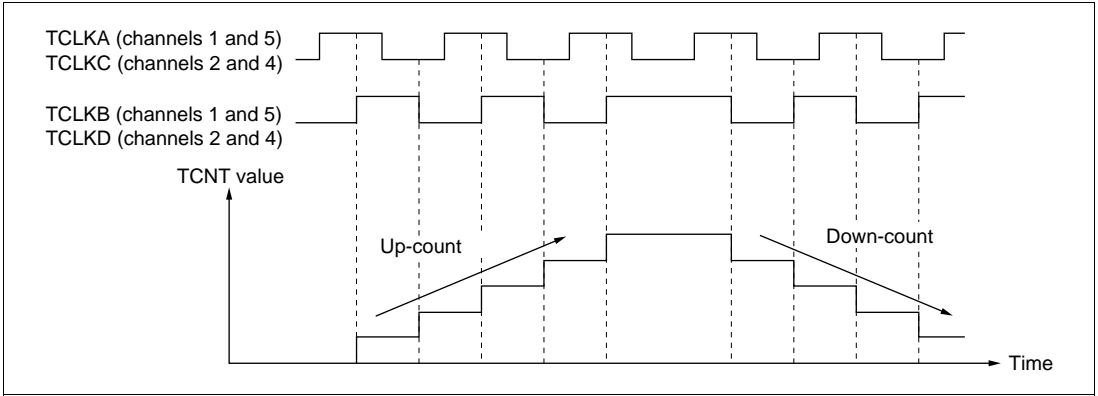
**Legend**

: Rising edge

: Falling edge

- Phase counting mode 4

Figure 7-32 shows an example of phase counting mode 4 operation, and table 7-12 summarizes the TCNT up/down-count conditions.



**Figure 7-32 Example of Phase Counting Mode 4 Operation**

**Table 7-12 Up/Down-Count Conditions in Phase Counting Mode 4**

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level		Up-count
Low level		Up-count
	Low level	Don't care
	High level	Don't care
High level		Down-count
Low level		Down-count
	High level	Don't care
	Low level	Don't care

**Legend**

- : Rising edge
- : Falling edge

**Phase Counting Mode Application Example:** Figure 7-33 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

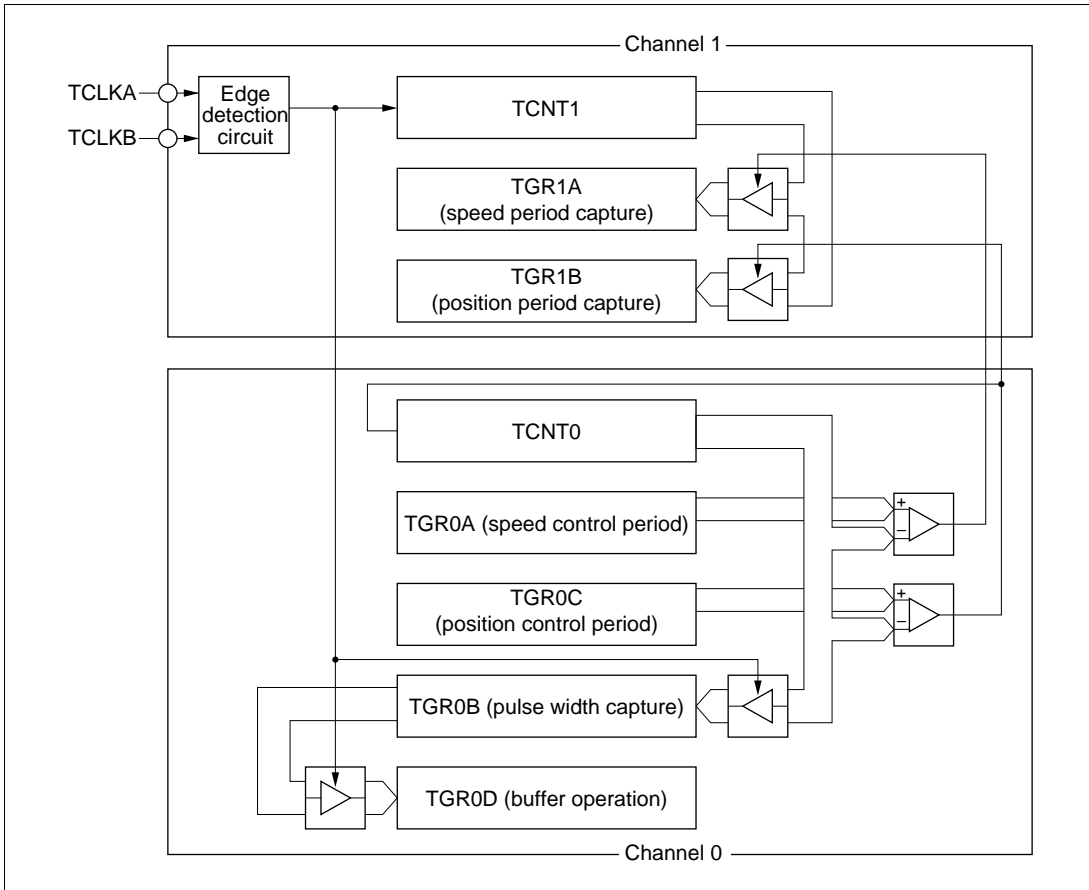
Channel 1 is set to phase counting mode 1, and the encoder pulse A-phase and B-phase are input to TCLKA and TCLKB.

Channel 0 operates with TCNT counter clearing by TGR0C compare match; TGR0A and TGR0C are used for the compare match function, and are set with the speed control period and position control period. TGR0B is used for input capture, with TGR0B and TGR0D operating in buffer mode. The channel 1 counter input clock is designated as the TGR0B input capture source, and detection of the pulse width of 2-phase encoder 4-multiplication pulses is performed.

TGR1A and TGR1B for channel 1 are designated for input capture, channel 0 TGR0A and TGR0C compare matches are selected as the input capture source, and store the up/down-counter values for the control periods.

This procedure enables accurate position/speed detection to be achieved.





**Figure 7-33 Phase Counting Mode Application Example**

## 7.5 Interrupts

### 7.5.1 Interrupt Sources and Priorities

There are three kinds of TPU interrupt source: TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disable bit, allowing generation of interrupt request signals to be enabled or disabled individually.

When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0.

Relative channel priorities can be changed by the interrupt controller, but the priority order within a channel is fixed. For details, see section 3, Interrupt Controller.

Table 7-13 lists the TPU interrupt sources.

**Table 7-13 TPU Interrupts**

Channel	Interrupt Source	Description	DMAC Activation	DTC Activation	Priority
0	TGI0A	TGR0A input capture/compare match	Possible	Possible	High ↑
	TGI0B	TGR0B input capture/compare match	Not possible	Possible	
	TGI0C	TGR0C input capture/compare match	Not possible	Possible	
	TGI0D	TGR0D input capture/compare match	Not possible	Possible	
	TCI0V	TCNT0 overflow	Not possible	Not possible	
1	TGI1A	TGR1A input capture/compare match	Possible	Possible	
	TGI1B	TGR1B input capture/compare match	Not possible	Possible	
	TCI1V	TCNT1 overflow	Not possible	Not possible	
	TCI1U	TCNT1 underflow	Not possible	Not possible	
2	TGI2A	TGR2A input capture/compare match	Possible	Possible	
	TGI2B	TGR2B input capture/compare match	Not possible	Possible	
	TCI2V	TCNT2 overflow	Not possible	Not possible	
	TCI2U	TCNT2 underflow	Not possible	Not possible	
3	TGI3A	TGR3A input capture/compare match	Possible	Possible	
	TGI3B	TGR3B input capture/compare match	Not possible	Possible	
	TGI3C	TGR3C input capture/compare match	Not possible	Possible	
	TGI3D	TGR3D input capture/compare match	Not possible	Possible	
	TCI3V	TCNT3 overflow	Not possible	Not possible	
4	TGI4A	TGR4A input capture/compare match	Possible	Possible	
	TGI4B	TGR4B input capture/compare match	Not possible	Possible	
	TCI4V	TCNT4 overflow	Not possible	Not possible	
	TCI4U	TCNT4 underflow	Not possible	Not possible	
5	TGI5A	TGR5A input capture/compare match	Possible	Possible	Low
	TGI5B	TGR5B input capture/compare match	Not possible	Possible	
	TCI5V	TCNT5 overflow	Not possible	Not possible	
	TCI5U	TCNT5 underflow	Not possible	Not possible	

Note: This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

**Input Capture/Compare Match Interrupt:** An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1 by the occurrence of a TGR input capture/compare match on a particular channel. The interrupt request is cleared by clearing the TGF flag to 0. The TPU has 16 input capture/compare match interrupts, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

**Overflow Interrupt:** An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to 1 by the occurrence of TCNT overflow on a channel. The interrupt request is cleared by clearing the TCFV flag to 0. The TPU has six overflow interrupts, one for each channel.

**Underflow Interrupt:** An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has four underflow interrupts, one each for channels 1, 2, 4, and 5.

### 7.5.2 DTC/DMAC Activation

**DTC Activation:** The DTC can be activated by the TGR input capture/compare match interrupt for a channel. For details, see section 6, Data Transfer Controller.

A total of 16 TPU input capture/compare match interrupts can be used as DTC activation sources, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

**DMAC Activation:** The DMAC can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 5, DMA Controller.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as DMAC activation sources, one for each channel.

### 7.5.3 A/D Converter Activation

The A/D converter can be activated by the TGRA input capture/compare match for a channel.

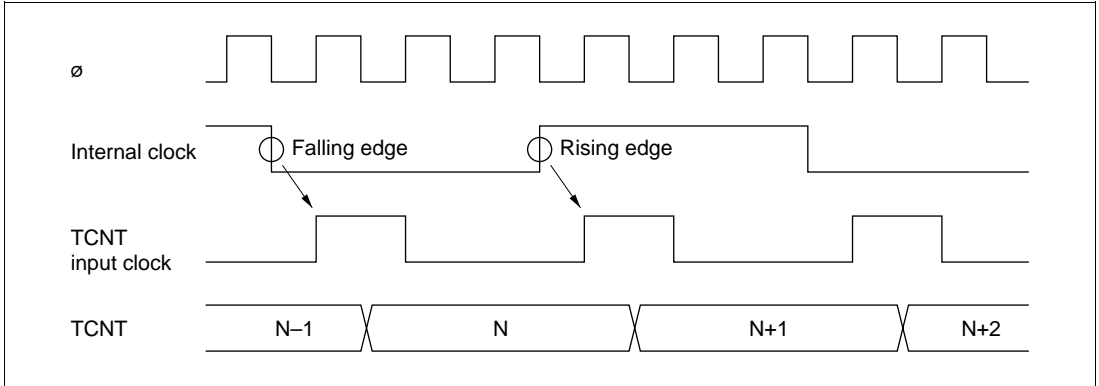
If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

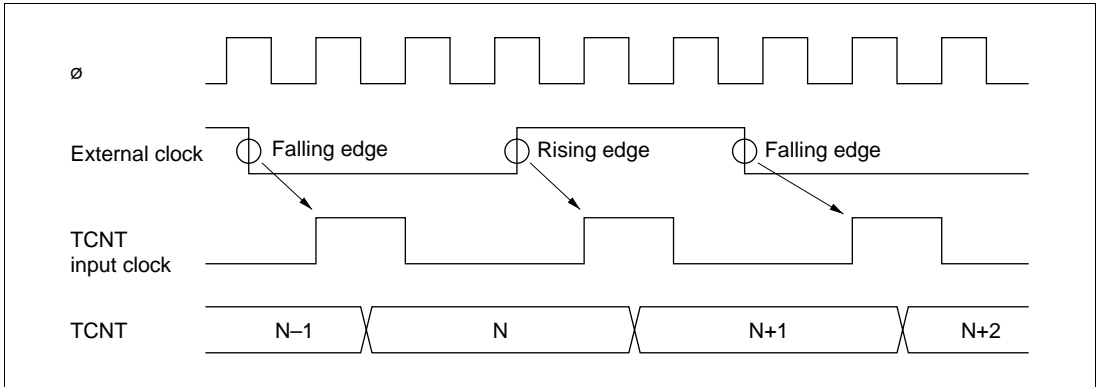
## 7.6 Operation Timing

### 7.6.1 Input/Output Timing

**TCNT Count Timing:** Figure 7-34 shows TCNT count timing in internal clock operation, and figure 7-35 shows TCNT count timing in external clock operation.



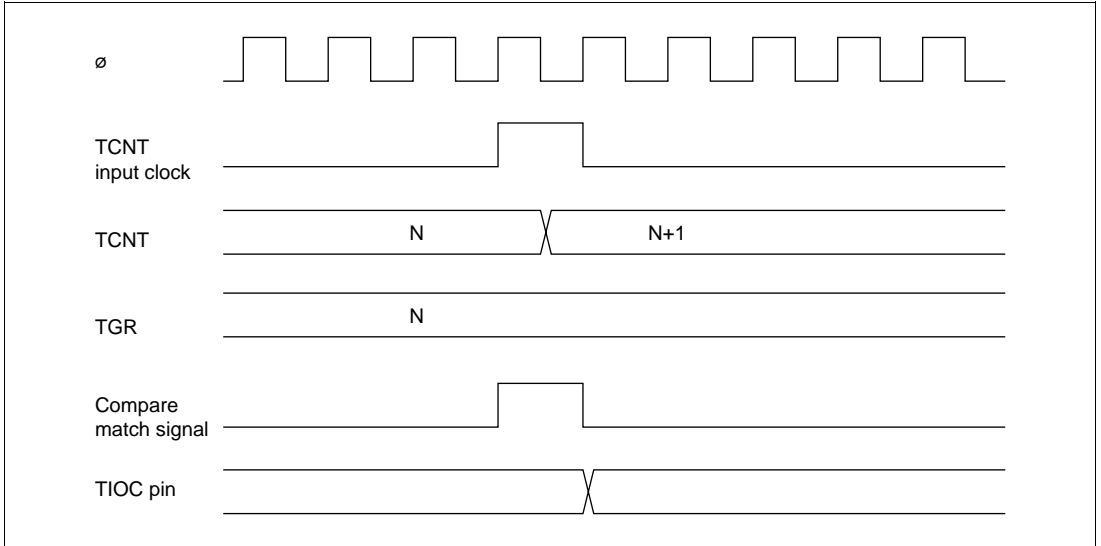
**Figure 7-34 Count Timing in Internal Clock Operation**



**Figure 7-35 Count Timing in External Clock Operation**

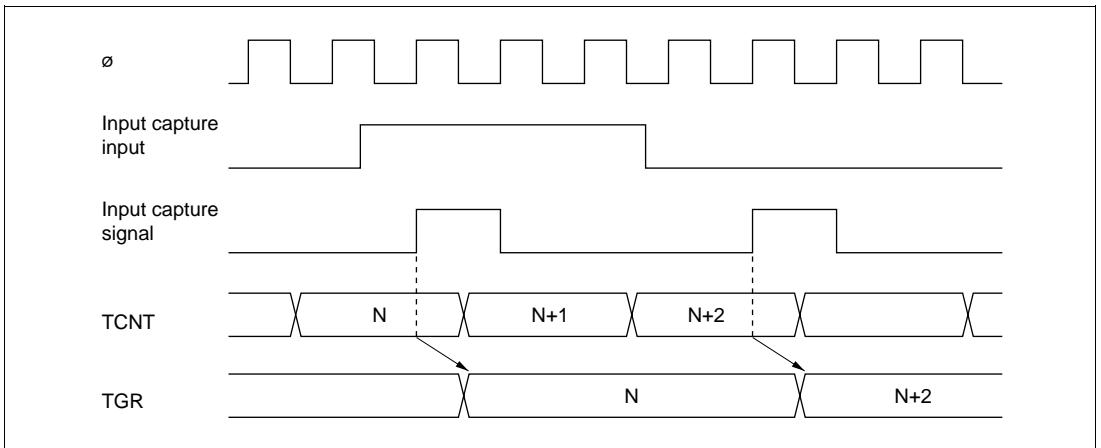
**Output Compare Output Timing:** A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin. After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

Figure 7-36 shows output compare output timing.



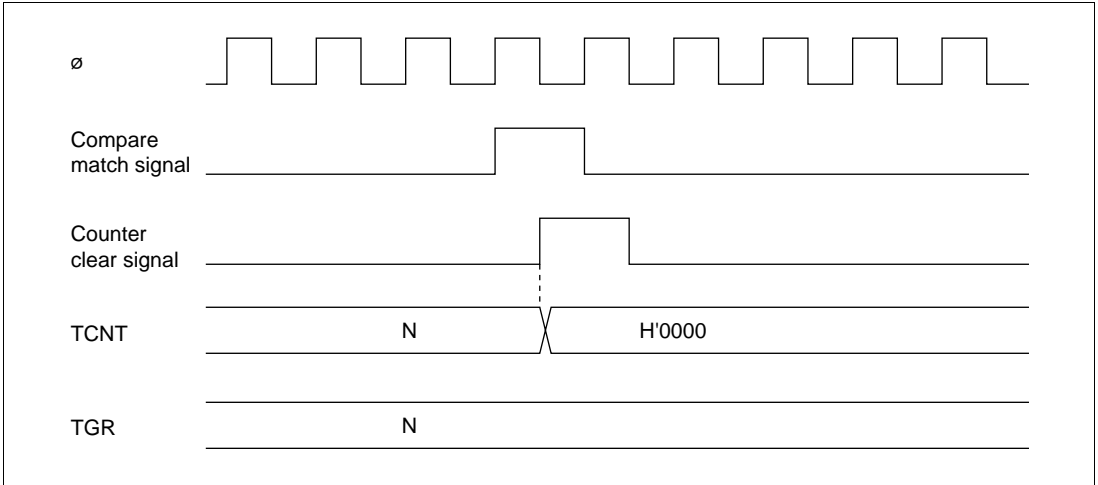
**Figure 7-36 Output Compare Output Timing**

**Input Capture Signal Timing:** Figure 7-37 shows input capture signal timing.

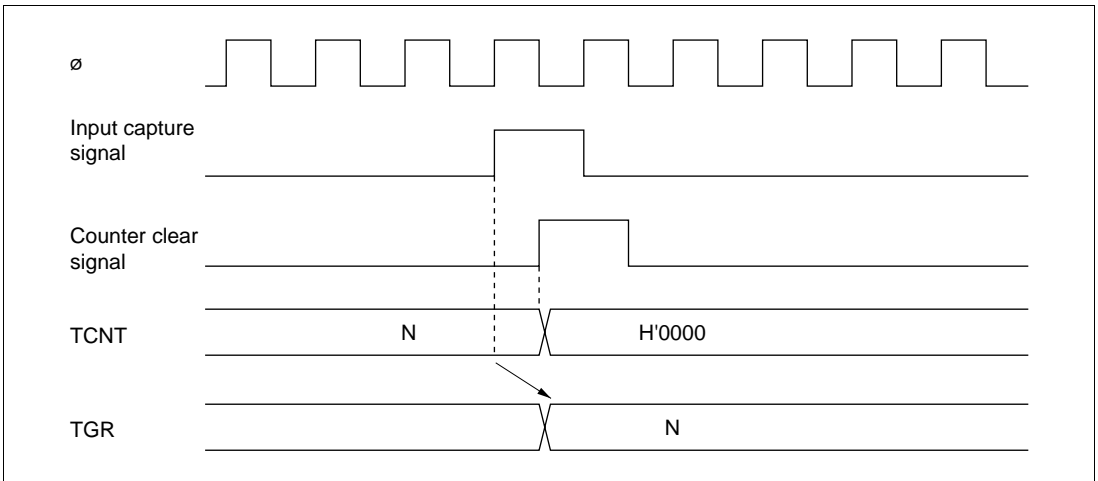


**Figure 7-37 Input Capture Input Signal Timing**

**Timing for Counter Clearing by Compare Match/Input Capture:** Figure 7-38 shows the timing when counter clearing by compare match occurrence is specified, and figure 7-39 shows the timing when counter clearing by input capture occurrence is specified.

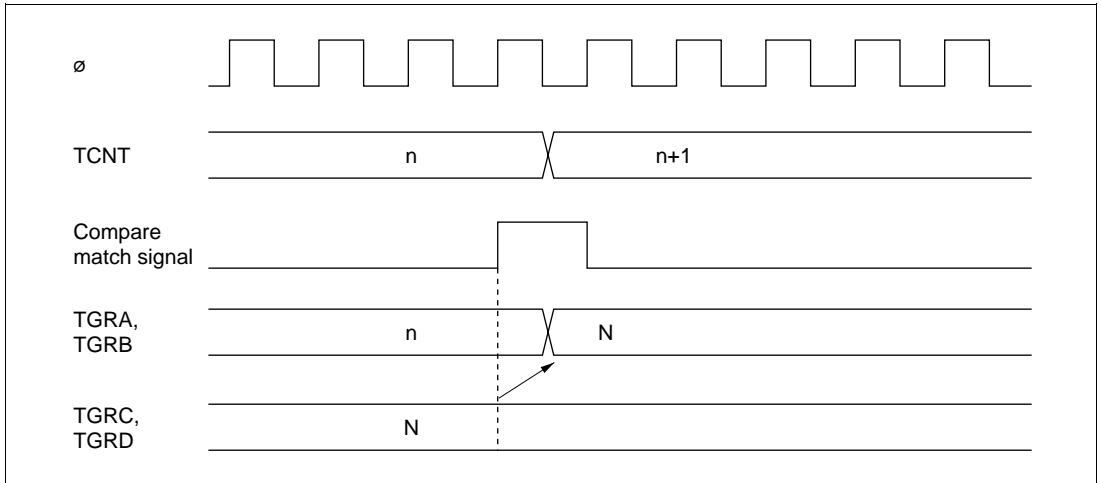


**Figure 7-38 Counter Clear Timing (Compare Match)**

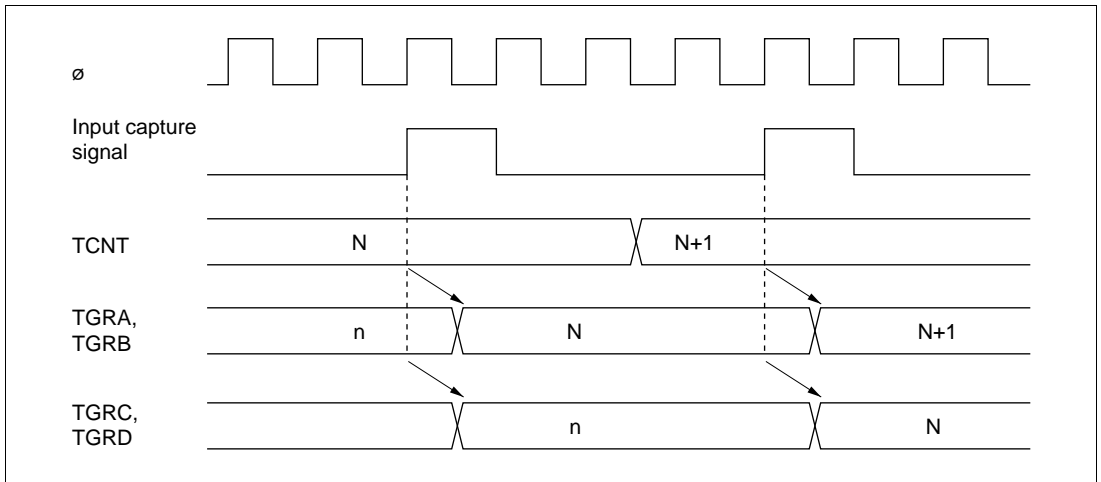


**Figure 7-39 Counter Clear Timing (Input Capture)**

**Buffer Operation Timing:** Figures 7-40 and 7-41 show the timing in buffer operation.



**Figure 7-40 Buffer Operation Timing (Compare Match)**

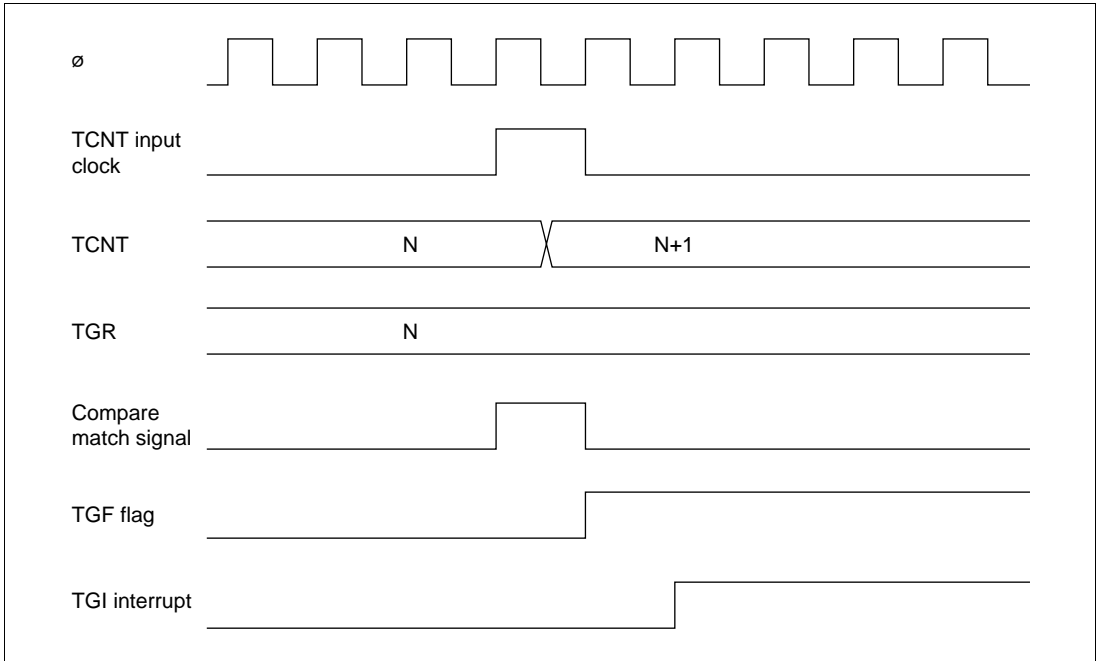


**Figure 7-41 Buffer Operation Timing (Input Capture)**



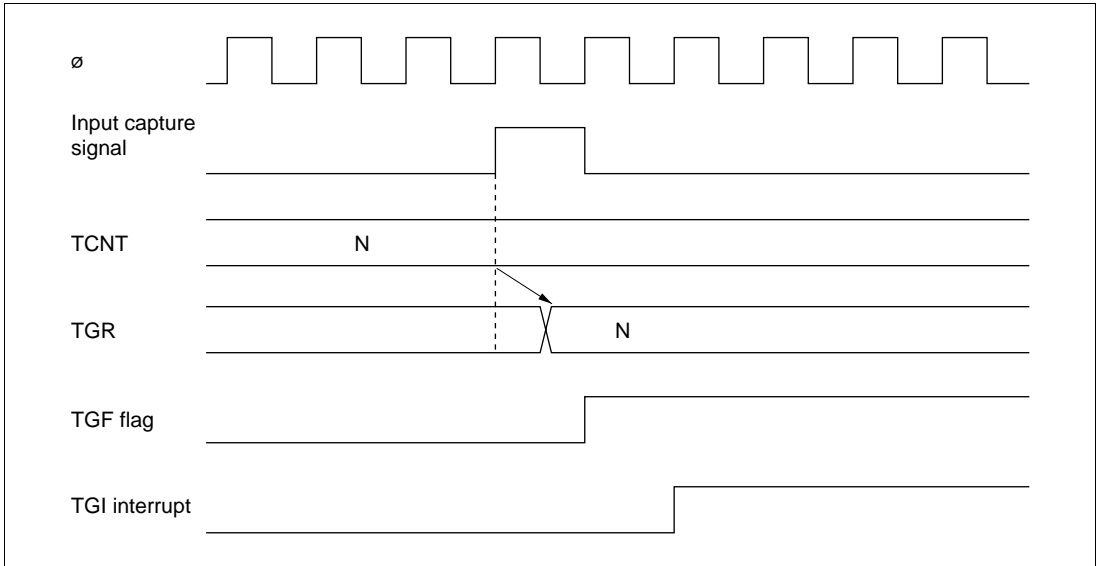
## 7.6.2 Interrupt Signal Timing

**TGF Flag Setting Timing in Case of Compare Match:** Figure 7-42 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and TGI interrupt request signal timing.



**Figure 7-42 TGI Interrupt Timing (Compare Match)**

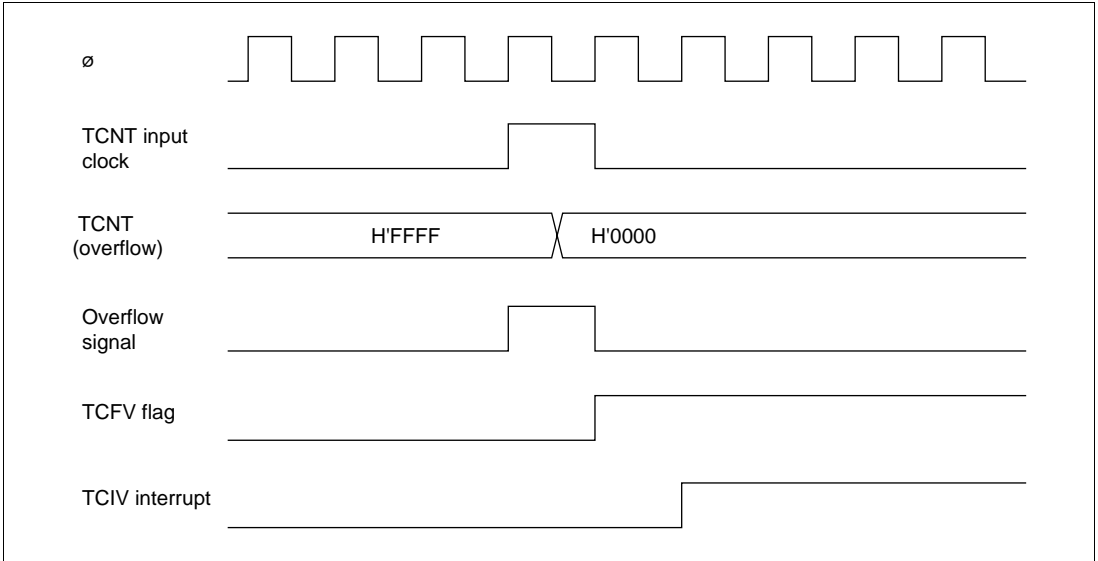
**TGF Flag Setting Timing in Case of Input Capture:** Figure 7-43 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and TGI interrupt request signal timing.



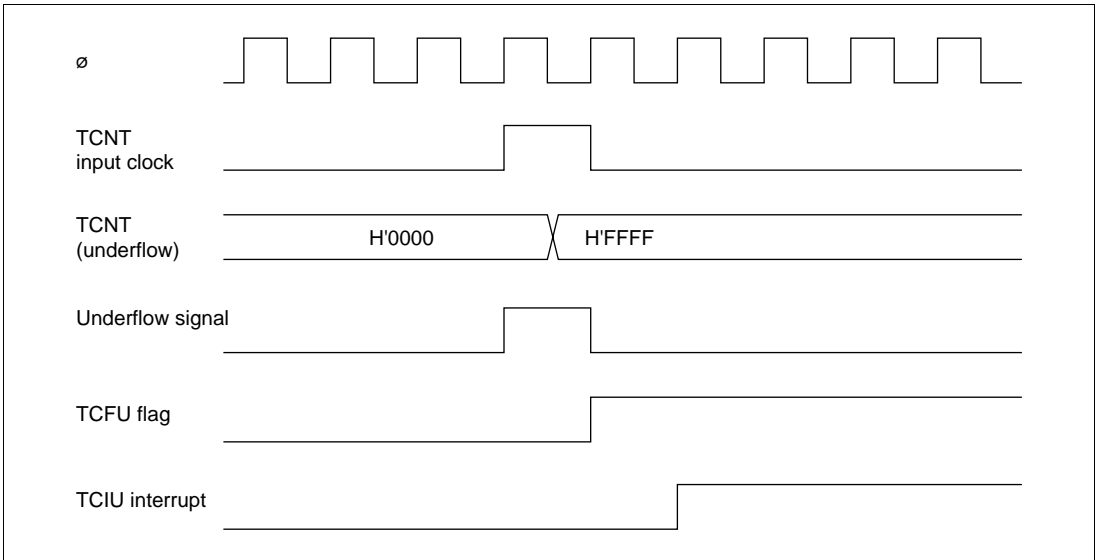
**Figure 7-43 TGI Interrupt Timing (Input Capture)**

**TCFV Flag/TCFU Flag Setting Timing:** Figure 7-44 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and TCIV interrupt request signal timing.

Figure 7-45 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and TCIU interrupt request signal timing.

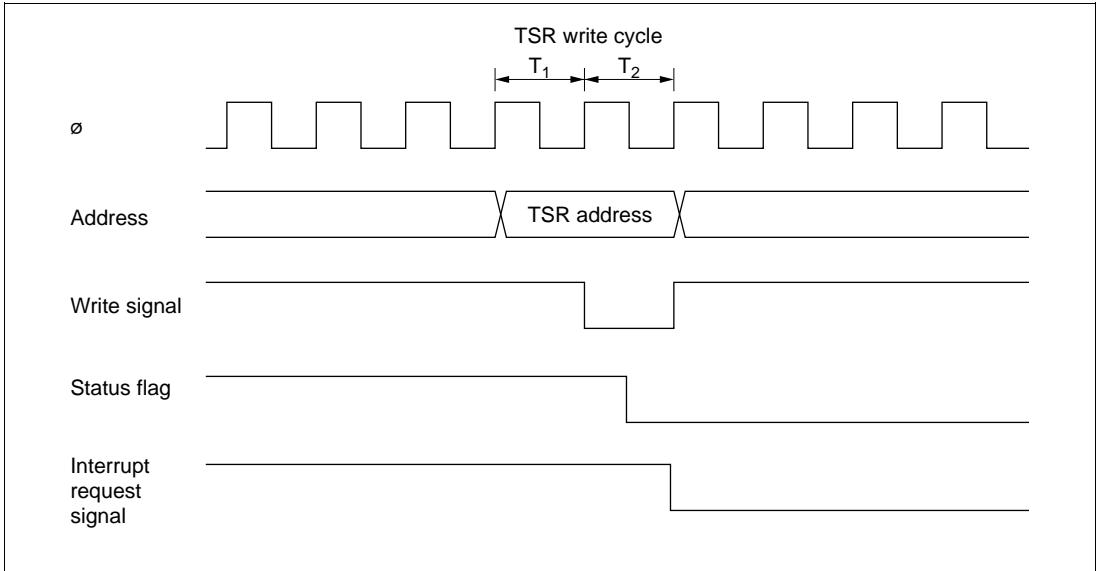


**Figure 7-44 TCIV Interrupt Setting Timing**

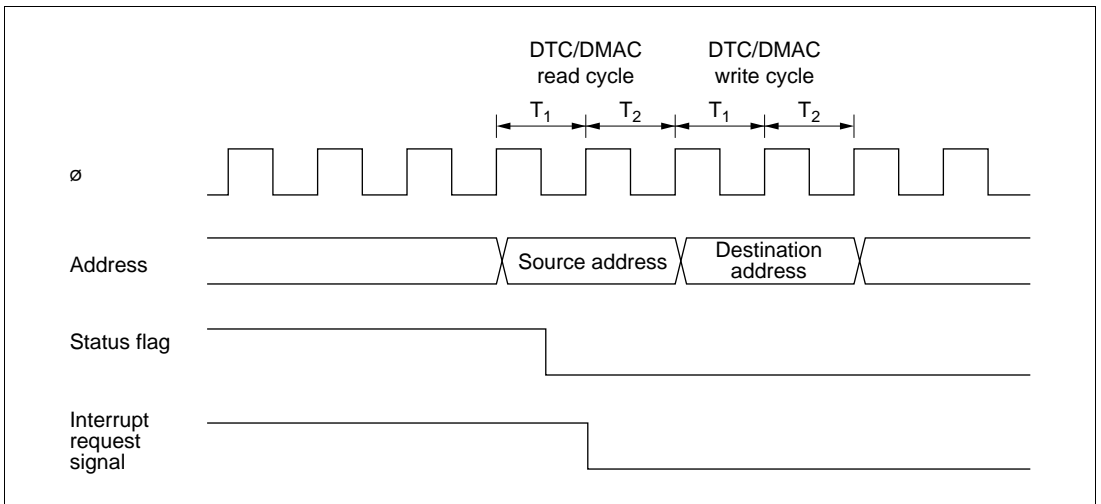


**Figure 7-45 TCIU Interrupt Setting Timing**

**Status Flag Clearing Timing:** After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DTC or DMAC is activated, the flag is cleared automatically. Figure 7-46 shows the timing for status flag clearing by the CPU, and figure 7-47 shows the timing for status flag clearing by the DTC or DMAC.



**Figure 7-46 Timing for Status Flag Clearing by CPU**



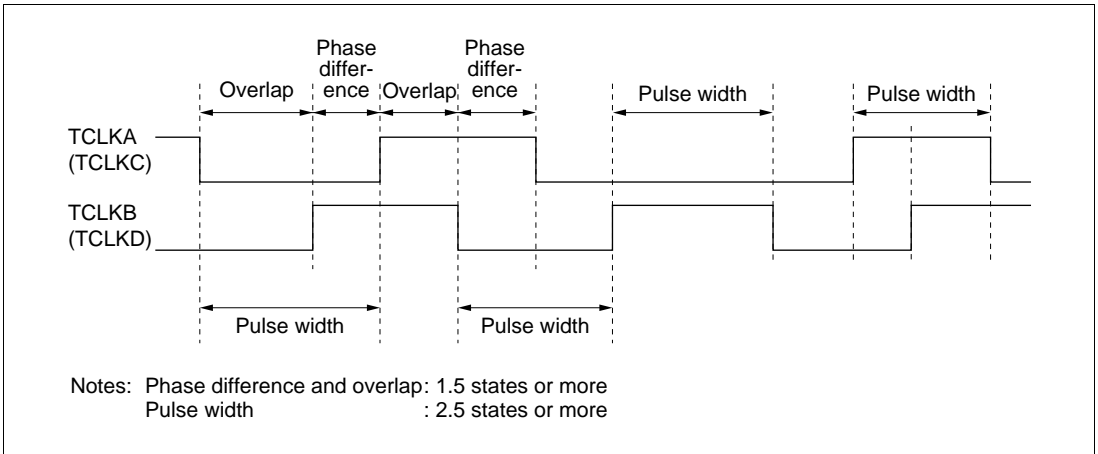
**Figure 7-47 Timing for Status Flag Clearing by DTC/DMAC Activation**

## 7.7 Usage Notes

Note that the kinds of operation and contention described below can occur during TPU operation.

**Input Clock Restrictions:** The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width.

In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 7-48 shows the input clock conditions in phase counting mode.



**Figure 7-48 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode**

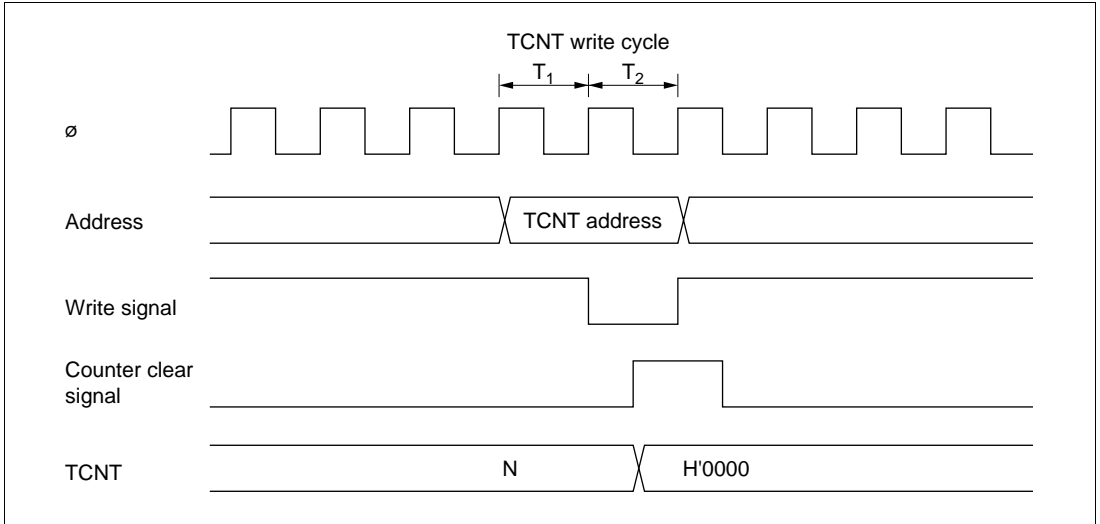
**Caution on Period Setting:** When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

$$f = \frac{\phi}{(N + 1)}$$

Where  $f$  : Counter frequency  
 $\phi$  : Operating frequency  
 $N$  : TGR set value

**Contention between TCNT Write and Clear Operations:** If the counter clear signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed.

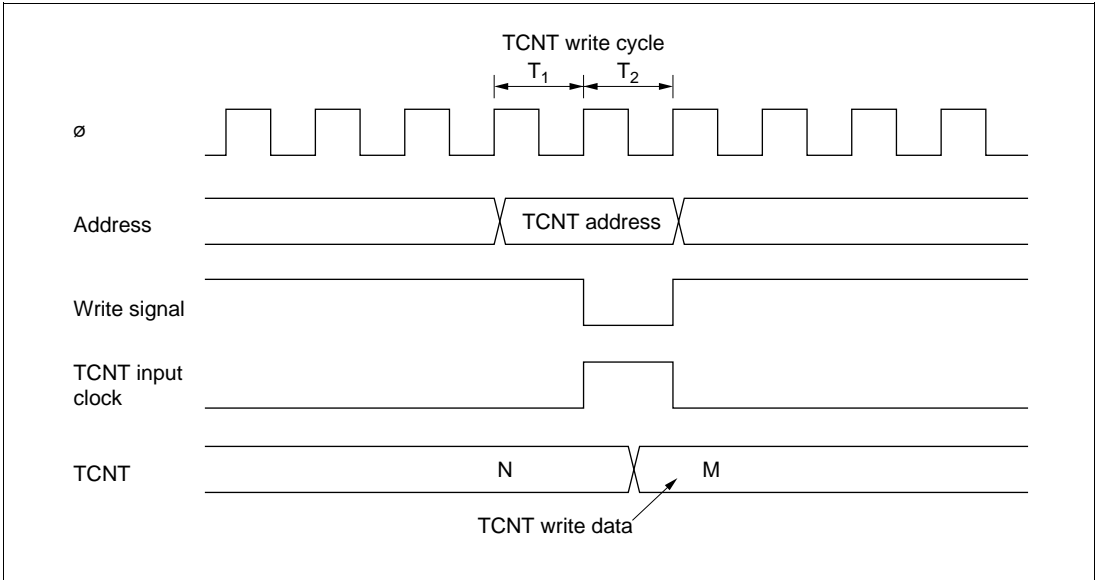
Figure 7-49 shows the timing in this case.



**Figure 7-49** Contention between TCNT Write and Clear Operations

**Contention between TCNT Write and Increment Operations:** If incrementing occurs in the  $T_2$  state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented.

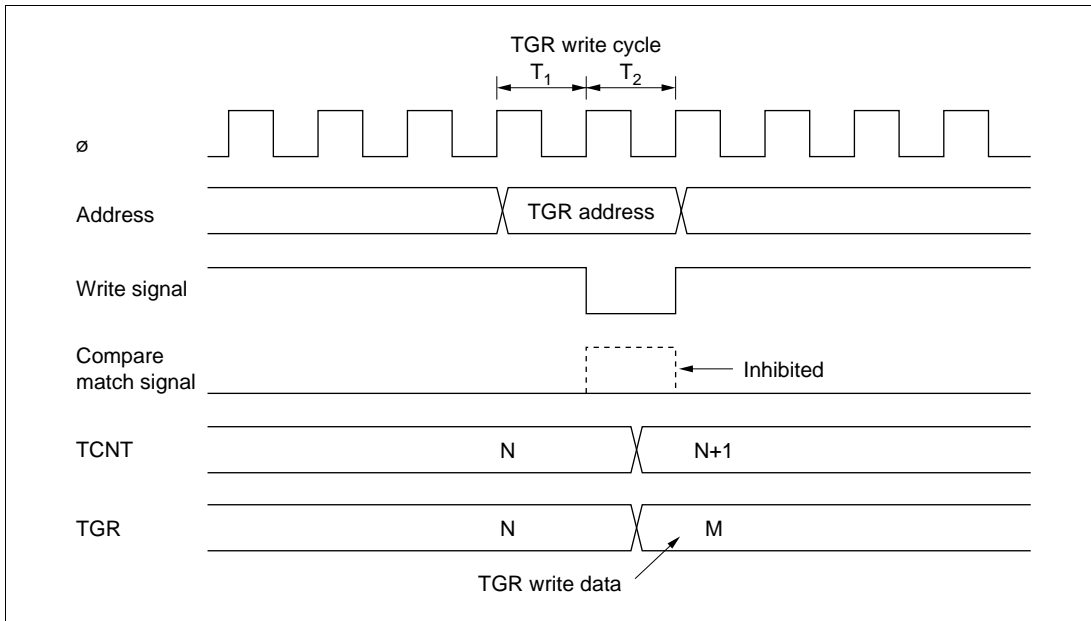
Figure 7-50 shows the timing in this case.



**Figure 7-50** Contention between TCNT Write and Increment Operations

**Contention between TGR Write and Compare Match:** If a compare match occurs in the  $T_2$  state of a TGR write cycle, the TGR write takes precedence and the compare match signal is inhibited. A compare match does not occur even if the same value as before is written.

Figure 7-51 shows the timing in this case.

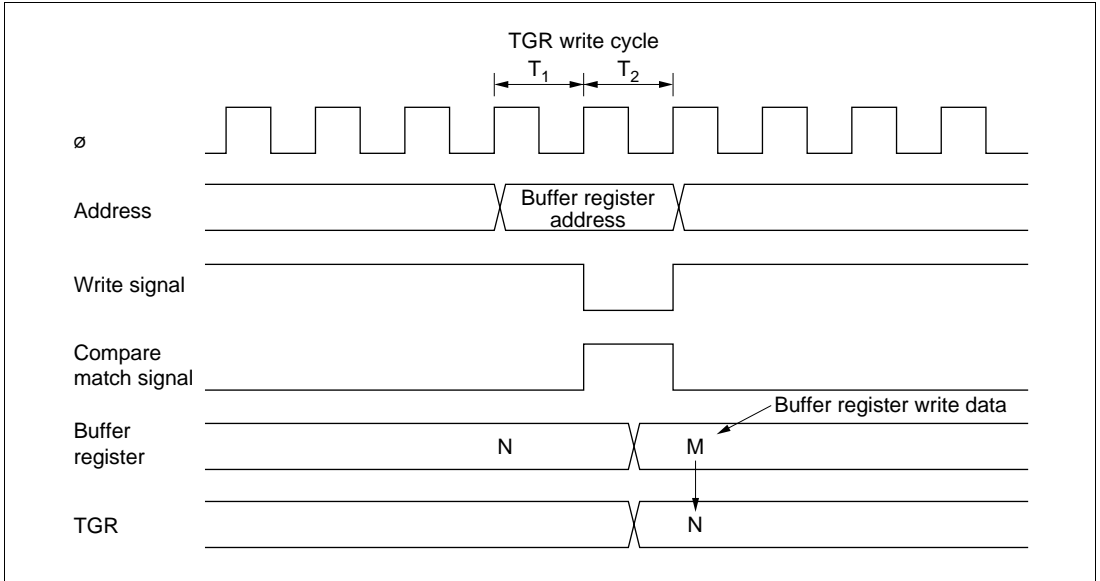


**Figure 7-51 Contention between TGR Write and Compare Match**



**Contention between Buffer Register Write and Compare Match:** If a compare match occurs in the  $T_2$  state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the data prior to the write.

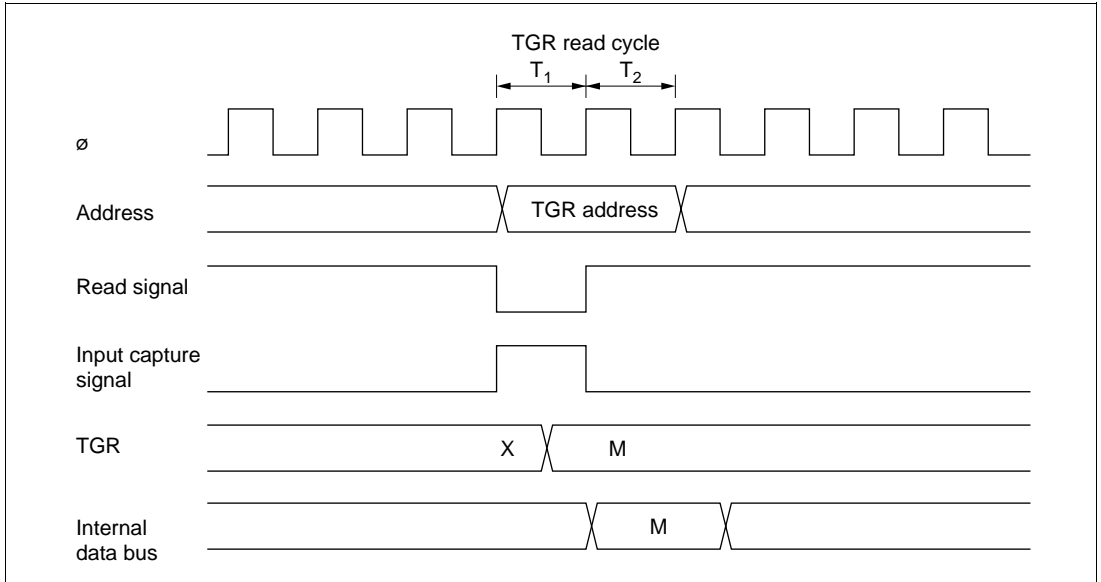
Figure 7-52 shows the timing in this case.



**Figure 7-52** Contention between Buffer Register Write and Compare Match

**Contention between TGR Read and Input Capture:** If the input capture signal is generated in the  $T_1$  state of a TGR read cycle, the data that is read will be the data after input capture transfer.

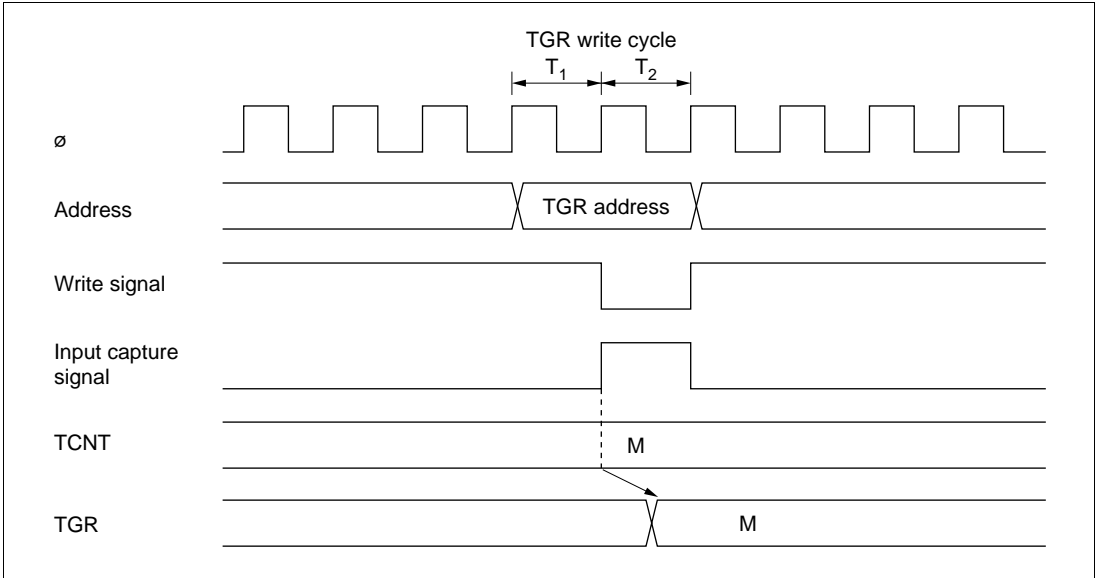
Figure 7-53 shows the timing in this case.



**Figure 7-53 Contention between TGR Read and Input Capture**

**Contention between TGR Write and Input Capture:** If the input capture signal is generated in the  $T_2$  state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

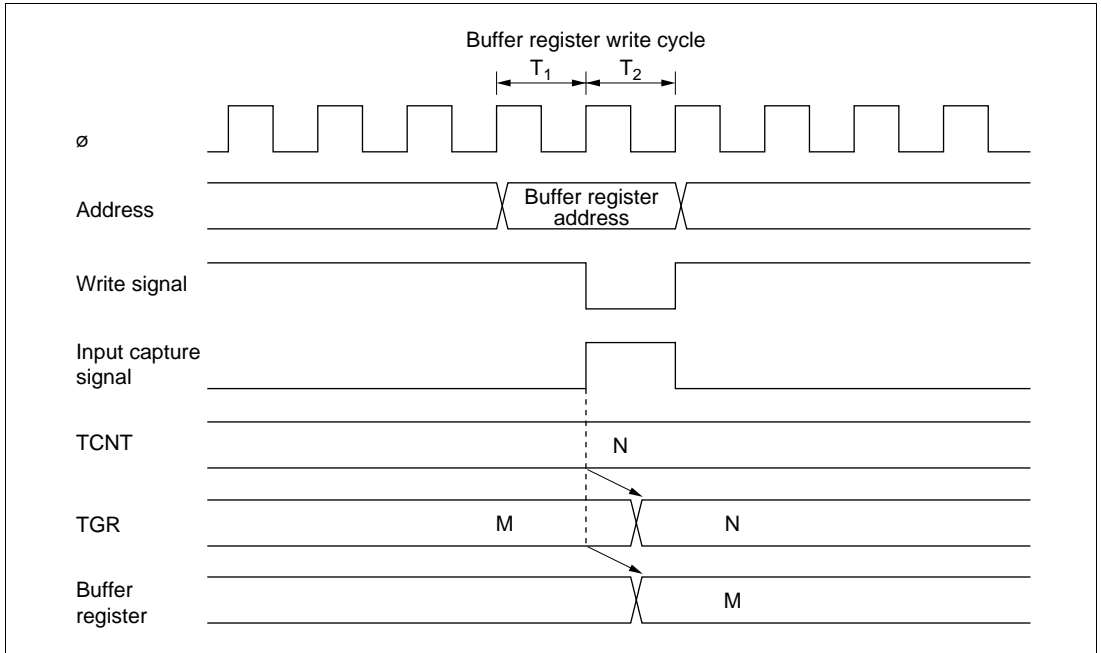
Figure 7-54 shows the timing in this case.



**Figure 7-54 Contention between TGR Write and Input Capture**

**Contention between Buffer Register Write and Input Capture:** If the input capture signal is generated in the  $T_2$  state of a buffer write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

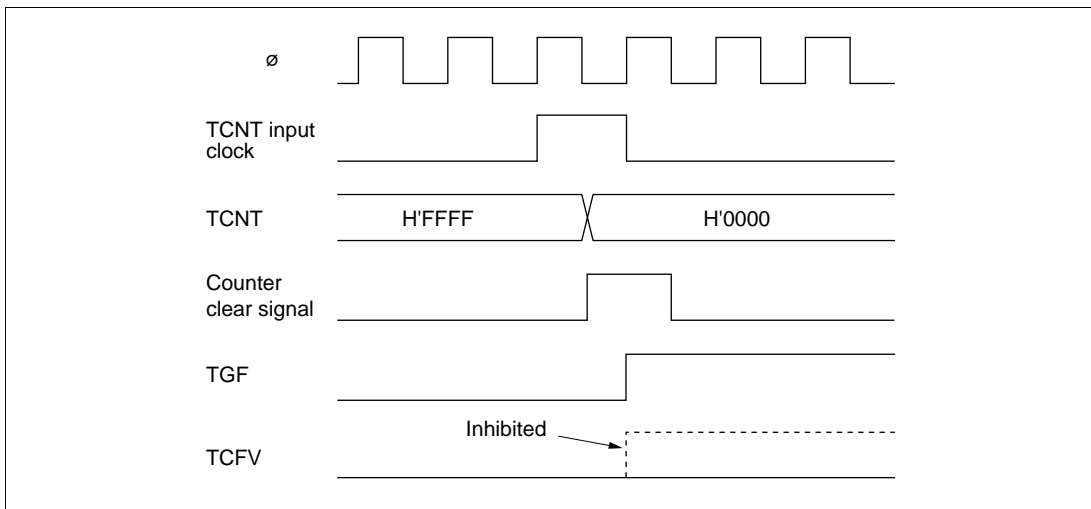
Figure 7-55 shows the timing in this case.



**Figure 7-55 Contention between Buffer Register Write and Input Capture**

**Contention between Overflow/Underflow and Counter Clearing:** If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

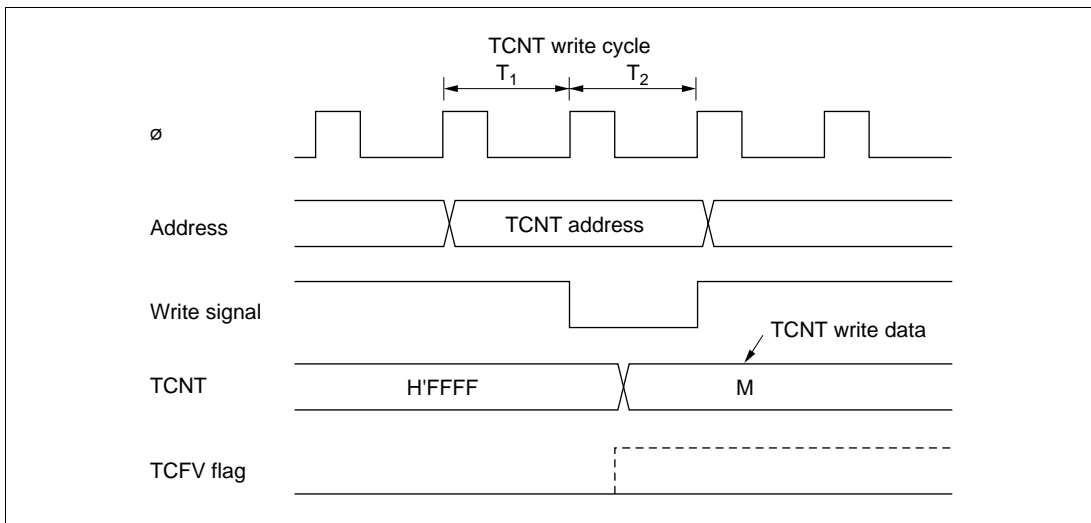
Figure 7-56 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.



**Figure 7-56** Contention between Overflow and Counter Clearing

**Contention between TCNT Write and Overflow/Underflow:** If there is an up-count or down-count in the  $T_2$  state of a TCNT write cycle, and overflow/underflow occurs, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 7-57 shows the operation timing when there is contention between TCNT write and overflow.



**Figure 7-57 Contention between TCNT Write and Overflow**

**Multiplexing of I/O Pins:** In the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

**Interrupts and Module Stop Mode:** If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

# Section 8 Programmable Pulse Generator (PPG)

## 8.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have a built-in programmable pulse generator (PPG) that provides pulse outputs by using the 16-bit timer-pulse unit (TPU) as a time base. The PPG pulse outputs are divided into 4-bit groups (group 3 to group 0) that can operate both simultaneously and independently.

### 8.1.1 Features

PPG features are listed below.

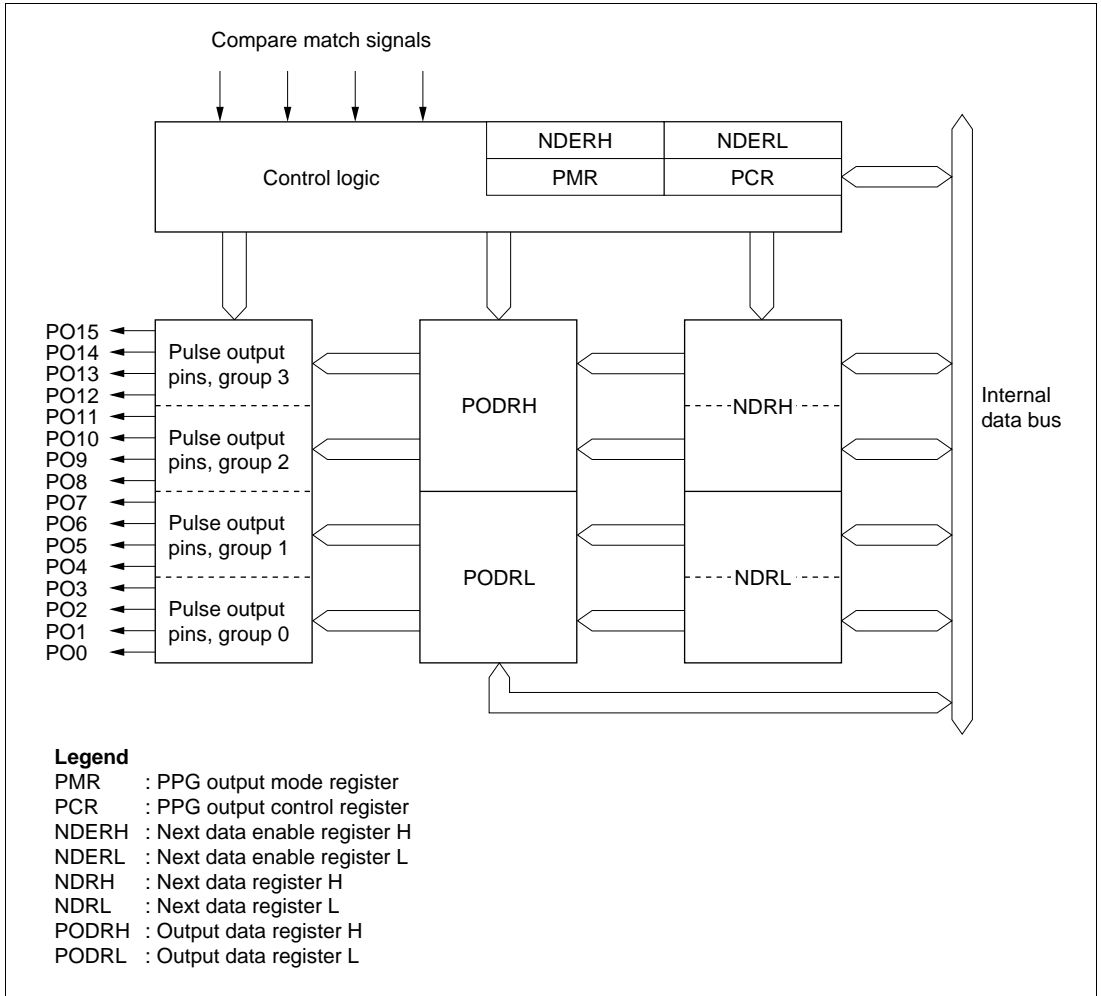
- 16-bit output data
  - Maximum 16-bit data can be output, and output can be enabled on a bit-by-bit basis
- Four output groups
  - Output trigger signals can be selected in 4-bit groups to provide up to four different 4-bit outputs
- Selectable output trigger signals
  - Output trigger signals can be selected for each group from the compare match signals of four TPU channels
- Non-overlap mode
  - A non-overlap margin can be provided between pulse outputs
- Can operate together with the data transfer controller (DTC) and DMA controller (DMAC)\*
  - The compare match signals selected as output trigger signals can activate the DTC or DMAC for sequential output of data without CPU intervention

Note: \* Some models do not have an on-chip DMAC; see the reference manual for the relevant model for details.

- Inverted output can be set
  - Inverted data can be output for each group
- Module stop mode can be set
  - As the initial setting, PPG operation is halted. Register access is enabled by exiting module stop mode

## 8.1.2 Block Diagram

Figure 8-1 shows a block diagram of the PPG.



**Figure 8-1 Block Diagram of PPG**



### 8.1.3 Pin Configuration

Table 8-1 summarizes the PPG pins.

**Table 8-1 PPG Pins**

<b>Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
Pulse output 0	PO0	Output	Group 0 pulse output
Pulse output 1	PO1	Output	
Pulse output 2	PO2	Output	
Pulse output 3	PO3	Output	Group 1 pulse output
Pulse output 4	PO4	Output	
Pulse output 5	PO5	Output	
Pulse output 6	PO6	Output	
Pulse output 7	PO7	Output	Group 2 pulse output
Pulse output 8	PO8	Output	
Pulse output 9	PO9	Output	
Pulse output 10	PO10	Output	
Pulse output 11	PO11	Output	
Pulse output 12	PO12	Output	Group 3 pulse output
Pulse output 13	PO13	Output	
Pulse output 14	PO14	Output	
Pulse output 15	PO15	Output	

## 8.1.4 Registers

Table 8-2 summarizes the PPG registers.

**Table 8-2 PPG Registers**

Name	Abbreviation	R/W	Initial Value	Address* <sup>1</sup>
PPG output control register	PCR	R/W	H'FF	H'FF46
PPG output mode register	PMR	R/W	H'F0	H'FF47
Next data enable register H	NDERH	R/W	H'00	H'FF48
Next data enable register L	NDERL	R/W	H'00	H'FF49
Output data register H	PODRH	R/(W)* <sup>2</sup>	H'00	H'FF4A
Output data register L	PODRL	R/(W)* <sup>2</sup>	H'00	H'FF4B
Next data register H	NDRH	R/W	H'00	H'FF4C* <sup>3</sup> H'FF4E
Next data register L	NDRL	R/W	H'00	H'FF4D* <sup>3</sup> H'FF4F
Port 1 data direction register	P1DDR	W	H'00	H'FEB0
Port 2 data direction register	P2DDR	W	H'00	H'FEB1
Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. Lower 16 bits of the address.

2. Bits used for pulse output cannot be written to.

3. When the same output trigger is selected for pulse output groups 2 and 3 by the PCR setting, the NDRH address is H'FF4C. When the output triggers are different, the NDRH address is H'FF4E for group 2 and H'FF4C for group 3.

Similarly, when the same output trigger is selected for pulse output groups 0 and 1 by the PCR setting, the NDRL address is H'FF4D. When the output triggers are different, the NDRL address is H'FF4F for group 0 and H'FF4D for group 1.

## 8.2 Register Descriptions

### 8.2.1 Next Data Enable Registers H and L (NDERH, NDERL)

#### NDERH

Bit	:	7	6	5	4	3	2	1	0
		NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### NDERL

Bit	:	7	6	5	4	3	2	1	0
		NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

NDERH and NDERL are 8-bit readable/writable registers that enable or disable pulse output on a bit-by-bit basis.

If a bit is enabled for pulse output by NDERH or NDERL, the NDR value is automatically transferred to the corresponding PODR bit when the TPU compare match event specified by PCR occurs, updating the output value. If pulse output is disabled, the bit value is not transferred from NDR to PODR and the output value does not change.

NDERH and NDERL are each initialized to H'00 by a reset and in hardware standby mode. They are not initialized in software standby mode.

**NDERH Bits 7 to 0—Next Data Enable 15 to 8 (NDER15 to NDER8):** These bits enable or disable pulse output on a bit-by-bit basis.

#### Bits 7 to 0

NDER15 to NDER8	Description
0	Pulse outputs PO15 to PO8 are disabled (NDR15 to NDR8 are not transferred to POD15 to POD8) (Initial value)
1	Pulse outputs PO15 to PO8 are enabled (NDR15 to NDR8 are transferred to POD15 to POD8)

**NDERL Bits 7 to 0—Next Data Enable 7 to 0 (NDER7 to NDER0):** These bits enable or disable pulse output on a bit-by-bit basis.

**Bits 7 to 0**

<b>NDER7 to NDER0</b>	<b>Description</b>
0	Pulse outputs PO7 to PO0 are disabled (NDR7 to NDR0 are not transferred to POD7 to POD0) (Initial value)
1	Pulse outputs PO7 to PO0 are enabled (NDR7 to NDR0 are transferred to POD7 to POD0)

**8.2.2 Output Data Registers H and L (PODRH, PODRL)**

**PODRH**

Bit	:	7	6	5	4	3	2	1	0
		POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

**PODRL**

Bit	:	7	6	5	4	3	2	1	0
		POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: \* A bit that has been set for pulse output by NDER is read-only.

PODRH and PODRL are 8-bit readable/writable registers that store output data for use in pulse output.

### 8.2.3 Next Data Registers H and L (NDRH, NDRL)

NDRH and NDRL are 8-bit readable/writable registers that store the next data for pulse output. During pulse output, the contents of NDRH and NDRL are transferred to the corresponding bits in PODRH and PODRL when the TPU compare match event specified by PCR occurs. The NDRH and NDRL addresses differ depending on whether pulse output groups have the same output trigger or different output triggers. For details see section 8.2.4, Notes on NDR Access.

NDRH and NDRL are each initialized to H'00 by a reset and in hardware standby mode. They are not initialized in software standby mode.

### 8.2.4 Notes on NDR Access

The NDRH and NDRL addresses differ depending on whether pulse output groups have the same output trigger or different output triggers.

**Same Trigger for Pulse Output Groups:** If pulse output groups 2 and 3 are triggered by the same compare match event, the NDRH address is H'FF4C. The upper 4 bits belong to group 3 and the lower 4 bits to group 2. Address H'FF4E consists entirely of reserved bits that cannot be modified and are always read as 1.

#### Address H'FF4C

Bit	:	7	6	5	4	3	2	1	0
		NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8
Initial value	:	0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### Address H'FF4E

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	—	—	—	—
Initial value	:	1	1	1	1	1	1	1	1
R/W	:	—	—	—	—	—	—	—	—

If pulse output groups 0 and 1 are triggered by the same compare match event, the NDRL address is H'FF4D. The upper 4 bits belong to group 1 and the lower 4 bits to group 0. Address H'FF4F consists entirely of reserved bits that cannot be modified and are always read as 1.

### Address H'FF4D

Bit	:	7	6	5	4	3	2	1	0
		NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

### Address H'FF4F

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	—	—	—	—
Initial value :		1	1	1	1	1	1	1	1
R/W	:	—	—	—	—	—	—	—	—

**Different Triggers for Pulse Output Groups:** If pulse output groups 2 and 3 are triggered by different compare match events, the address of the upper 4 bits in NDRH (group 3) is H'FF4C and the address of the lower 4 bits (group 2) is H'FF4E. Bits 3 to 0 of address H'FF4C and bits 7 to 4 of address H'FF4E are reserved bits that cannot be modified and are always read as 1.

### Address H'FF4C

Bit	:	7	6	5	4	3	2	1	0
		NDR15	NDR14	NDR13	NDR12	—	—	—	—
Initial value :		0	0	0	0	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	—	—	—	—

### Address H'FF4E

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	NDR11	NDR10	NDR9	NDR8
Initial value :		1	1	1	1	0	0	0	0
R/W	:	—	—	—	—	R/W	R/W	R/W	R/W

If pulse output groups 0 and 1 are triggered by different compare match event, the address of the upper 4 bits in NDRL (group 1) is H'FF4D and the address of the lower 4 bits (group 0) is H'FF4F. Bits 3 to 0 of address H'FF4D and bits 7 to 4 of address H'FF4F are reserved bits that cannot be modified and are always read as 1.

## Address H'FF4D

Bit	:	7	6	5	4	3	2	1	0
		NDR7	NDR6	NDR5	NDR4	—	—	—	—
Initial value :		0	0	0	0	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	—	—	—	—

## Address H'FF4F

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	NDR3	NDR2	NDR1	NDR0
Initial value :		1	1	1	1	0	0	0	0
R/W	:	—	—	—	—	R/W	R/W	R/W	R/W

### 8.2.5 PPG Output Control Register (PCR)

Bit	:	7	6	5	4	3	2	1	0
		G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0
Initial value :		1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PCR is an 8-bit readable/writable register that selects output trigger signals for PPG outputs on a group-by-group basis.

PCR is initialized to H'FF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 7 and 6—Group 3 Compare Match Select 1 and 0 (G3CMS1, G3CMS0):** These bits select the compare match that triggers pulse output group 3 (pins PO15 to PO12).

		Description
Bit 7	Bit 6	
G3CMS1	G3CMS0	Output Trigger for Pulse Output Group 3
0	0	Compare match in TPU channel 0
	1	Compare match in TPU channel 1
1	0	Compare match in TPU channel 2
	1	Compare match in TPU channel 3 (Initial value)

**Bits 5 and 4—Group 2 Compare Match Select 1 and 0 (G2CMS1, G2CMS0):** These bits select the compare match that triggers pulse output group 2 (pins PO11 to PO8).

		<b>Description</b>
Bit 5 G2CMS1	Bit 4 G2CMS0	<b>Output Trigger for Pulse Output Group 2</b>
0	0	Compare match in TPU channel 0
	1	Compare match in TPU channel 1
1	0	Compare match in TPU channel 2
	1	Compare match in TPU channel 3 (Initial value)

**Bits 3 and 2—Group 1 Compare Match Select 1 and 0 (G1CMS1, G1CMS0):** These bits select the compare match that triggers pulse output group 1 (pins PO7 to PO4).

		<b>Description</b>
Bit 3 G1CMS1	Bit 2 G1CMS0	<b>Output Trigger for Pulse Output Group 1</b>
0	0	Compare match in TPU channel 0
	1	Compare match in TPU channel 1
1	0	Compare match in TPU channel 2
	1	Compare match in TPU channel 3 (Initial value)

**Bits 1 and 0—Group 0 Compare Match Select 1 and 0 (G0CMS1, G0CMS0):** These bits select the compare match that triggers pulse output group 0 (pins PO3 to PO0).

		<b>Description</b>
Bit 1 G0CMS1	Bit 0 G0CMS0	<b>Output Trigger for Pulse Output Group 0</b>
0	0	Compare match in TPU channel 0
	1	Compare match in TPU channel 1
1	0	Compare match in TPU channel 2
	1	Compare match in TPU channel 3 (Initial value)



## 8.2.6 PPG Output Mode Register (PMR)

Bit	:	7	6	5	4	3	2	1	0
		G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV
Initial value :		1	1	1	1	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PMR is an 8-bit readable/writable register that selects pulse output inversion and non-overlapping operation for each group.

The output trigger period of a non-overlapping operation PPG output waveform is set in TGRB and the non-overlap margin is set in TGRA. The output values change at compare match A and B.

For details, see section 8.3.4, Non-Overlapping Pulse Output.

PMR is initialized to H'F0 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Group 3 Inversion (G3INV):** Selects direct output or inverted output for pulse output group 3 (pins PO15 to PO12).

### Bit 7

G3INV	Description
0	Inverted output for pulse output group 3 (low-level output at pin for a 1 in PODRH)
1	Direct output for pulse output group 3 (high-level output at pin for a 1 in PODRH) (Initial value)

**Bit 6—Group 2 Inversion (G2INV):** Selects direct output or inverted output for pulse output group 2 (pins PO11 to PO8).

### Bit 6

G2INV	Description
0	Inverted output for pulse output group 2 (low-level output at pin for a 1 in PODRH)
1	Direct output for pulse output group 2 (high-level output at pin for a 1 in PODRH) (Initial value)

**Bit 5—Group 1 Inversion (G1INV):** Selects direct output or inverted output for pulse output group 1 (pins PO7 to PO4).

Bit 5 G1INV	Description
0	Inverted output for pulse output group 1 (low-level output at pin for a 1 in PODRL)
1	Direct output for pulse output group 1 (high-level output at pin for a 1 in PODRL) (Initial value)

**Bit 4—Group 0 Inversion (G0INV):** Selects direct output or inverted output for pulse output group 0 (pins PO3 to PO0).

Bit 4 G0INV	Description
0	Inverted output for pulse output group 0 (low-level output at pin for a 1 in PODRL)
1	Direct output for pulse output group 0 (high-level output at pin for a 1 in PODRL) (Initial value)

**Bit 3—Group 3 Non-Overlap (G3NOV):** Selects normal or non-overlapping operation for pulse output group 3 (pins PO15 to PO12).

Bit 3 G3NOV	Description
0	Normal operation in pulse output group 3 (output values updated at compare match A in the selected TPU channel) (Initial value)
1	Non-overlapping operation in pulse output group 3 (independent 1 and 0 output at compare match A or B in the selected TPU channel)

**Bit 2—Group 2 Non-Overlap (G2NOV):** Selects normal or non-overlapping operation for pulse output group 2 (pins PO11 to PO8).

Bit 2 G2NOV	Description
0	Normal operation in pulse output group 2 (output values updated at compare match A in the selected TPU channel) (Initial value)
1	Non-overlapping operation in pulse output group 2 (independent 1 and 0 output at compare match A or B in the selected TPU channel)

**Bit 1—Group 1 Non-Overlap (G1NOV):** Selects normal or non-overlapping operation for pulse output group 1 (pins PO7 to PO4).

**Bit 1**

G1NOV	Description
0	Normal operation in pulse output group 1 (output values updated at compare match A in the selected TPU channel) (Initial value)
1	Non-overlapping operation in pulse output group 1 (independent 1 and 0 output at compare match A or B in the selected TPU channel)

**Bit 0—Group 0 Non-Overlap (G0NOV):** Selects normal or non-overlapping operation for pulse output group 0 (pins PO3 to PO0).

**Bit 0**

G0NOV	Description
0	Normal operation in pulse output group 0 (output values updated at compare match A in the selected TPU channel) (Initial value)
1	Non-overlapping operation in pulse output group 0 (independent 1 and 0 output at compare match A or B in the selected TPU channel)

**8.2.7 Port 1 Data Direction Register (P1DDR)**

Bit	7	6	5	4	3	2	1	0
	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR
Initial value :	0	0	0	0	0	0	0	0
R/W :	W	W	W	W	W	W	W	W

P1DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 1.

Port 1 is multiplexed with pins PO15 to PO8. Bits corresponding to pins used for PPG output must be set to 1. For further information about P1DDR, see the I/O Port section in the reference manual for the relevant model.

## 8.2.8 Port 2 Data Direction Register (P2DDR)

Bit	:	7	6	5	4	3	2	1	0
		P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR
Initial value :		0	0	0	0	0	0	0	0
R/W	:	W	W	W	W	W	W	W	W

P2DDR is an 8-bit write-only register, the individual bits of which specify input or output for the pins of port 2.

Port 2 is multiplexed with pins PO7 to PO0. Bits corresponding to pins used for PPG output must be set to 1. For further information about P2DDR, see the I/O Port section in the reference manual for the relevant model.

## 8.2.9 Module Stop Control Register (MSTPCR)

		MSTPCRH								MSTPCRL							
Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :		0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP11 bit in MSTPCR is set to 1, PPG operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 11—Module Stop (MSTP11):** Specifies the PPG module stop mode.

Bit 11	MSTP11	Description
0		PPG module stop mode cleared
1		PPG module stop mode set (Initial value)

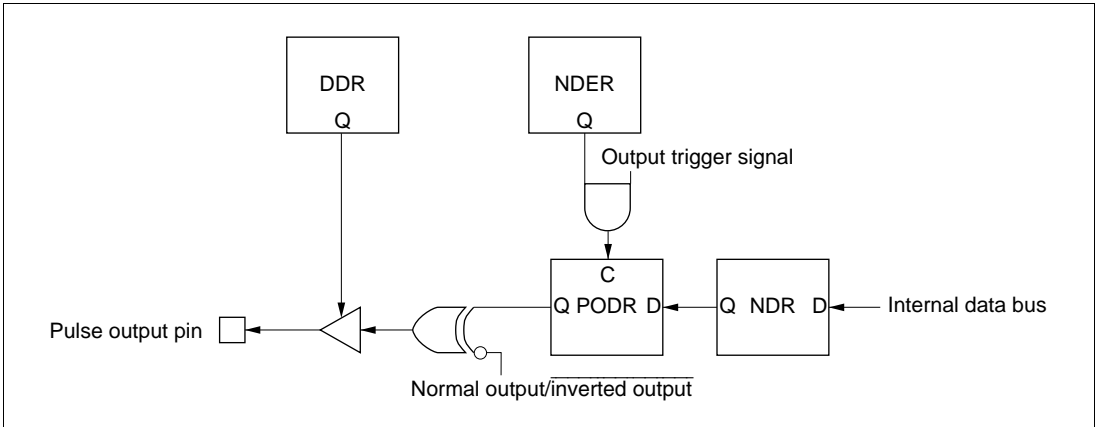
## 8.3 Operation

### 8.3.1 Overview

PPG pulse output is enabled when the corresponding bits in P1DDR, P2DDR, and NDER are set to 1. In this state the corresponding PODR contents are output.

When the compare match event specified by PCR occurs, the corresponding NDR bit contents are transferred to PODR to update the output values.

Figure 8-2 illustrates the PPG output operation and table 8-3 summarizes the PPG operating conditions.



**Figure 8-2 PPG Output Operation**

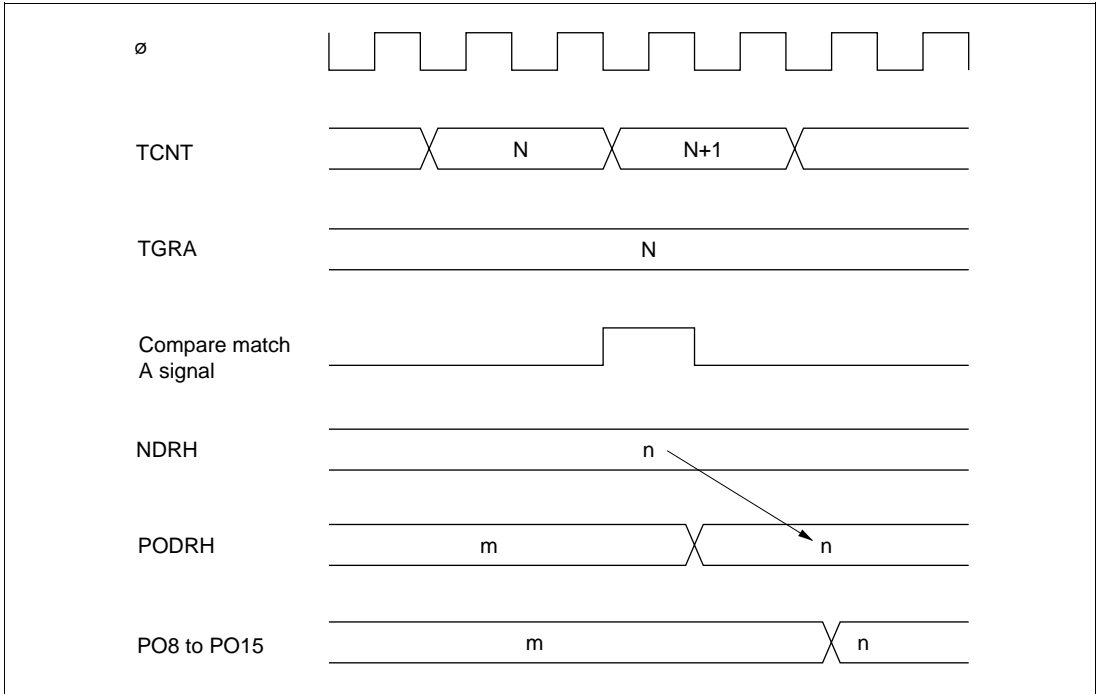
**Table 8-3 PPG Operating Conditions**

NDR	DDR	Pin Function
0	0	Generic input port
	1	Generic output port
1	0	Generic input port (but the PODR bit is a read-only bit, and when compare match occurs, the NDR bit value is transferred to the PODR bit)
	1	PPG pulse output

Sequential output of data of up to 16 bits is possible by writing new output data to NDR before the next compare match. For details of non-overlapping operation, see section 8.3.4, Non-Overlapping Pulse Output.

### 8.3.2 Output Timing

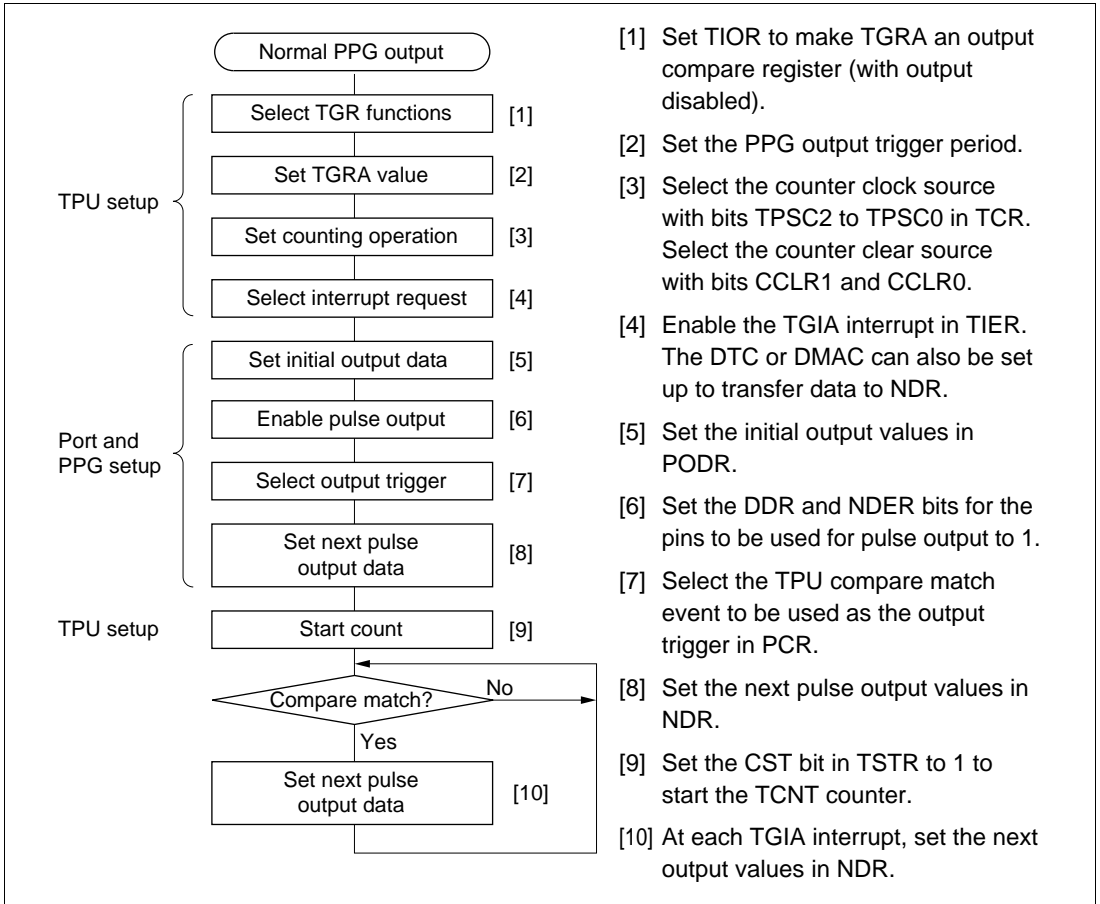
If pulse output is enabled, NDR contents are transferred to PODR and output when the specified compare match event occurs. Figure 8-3 shows the timing of these operations for the case of normal output in groups 2 and 3, triggered by compare match A.



**Figure 8-3** Timing of Transfer and Output of NDR Contents (Example)

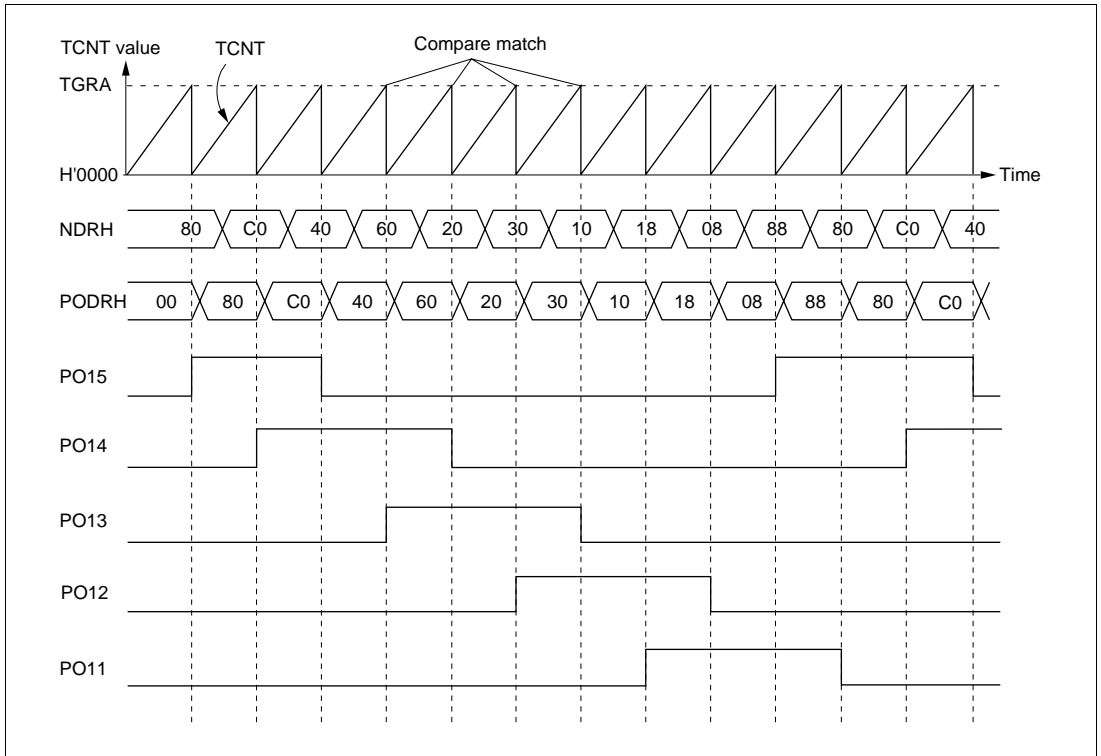
### 8.3.3 Normal Pulse Output

**Sample Setup Procedure for Normal Pulse Output:** Figure 8-4 shows a sample procedure for setting up normal pulse output.



**Figure 8-4 Setup Procedure for Normal Pulse Output (Example)**

**Example of Normal Pulse Output (Example of Five-Phase Pulse Output):** Figure 8-5 shows an example in which pulse output is used for cyclic five-phase pulse output.



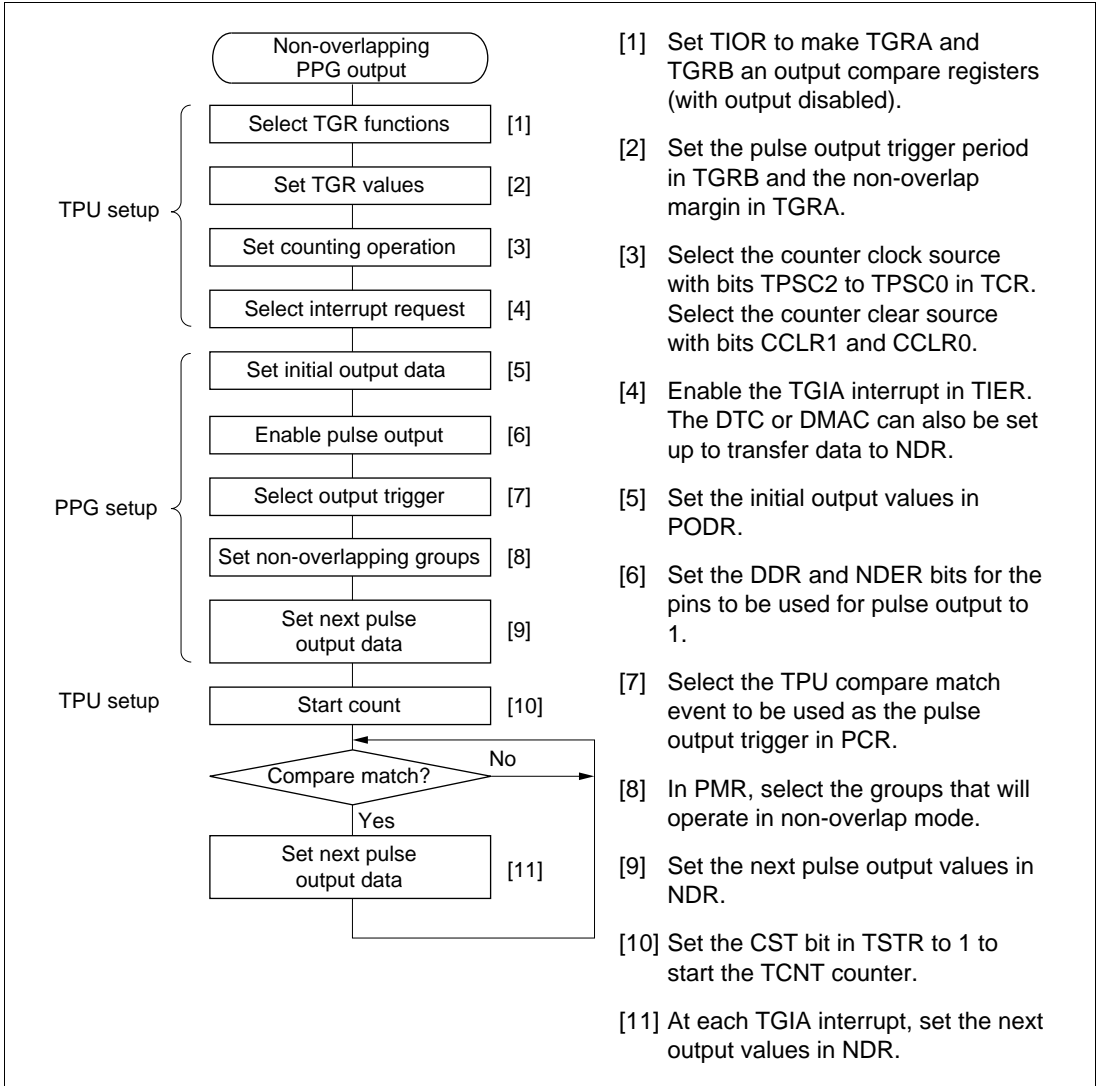
**Figure 8-5 Normal Pulse Output Example (Five-Phase Pulse Output)**

- [1] Set up the TPU channel to be used as the output trigger channel so that TGRA is an output compare register and the counter will be cleared by compare match A. Set the trigger period in TGRA and set the TGIEA bit in TIER to 1 to enable the compare match A (TGIA) interrupt.
- [2] Write H'F8 in P1DDR and NDRH, and set the G3CMS1, G3CMS0, G2CMS1, and G2CMS0 bits in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Write output data H'80 in NDRH.
- [3] The timer counter in the TPU channel starts. When compare match A occurs, the NDRH contents are transferred to PODRH and output. The TGIA interrupt handling routine writes the next output data (H'C0) in NDRH.
- [4] Five-phase overlapping pulse output (one or two phases active at a time) can be obtained subsequently by writing H'40, H'60, H'20, H'30, H'10, H'18, H'08, H'88... at successive TGIA interrupts. If the DTC or DMAC is set for activation by this interrupt, pulse output can be obtained without imposing a load on the CPU.



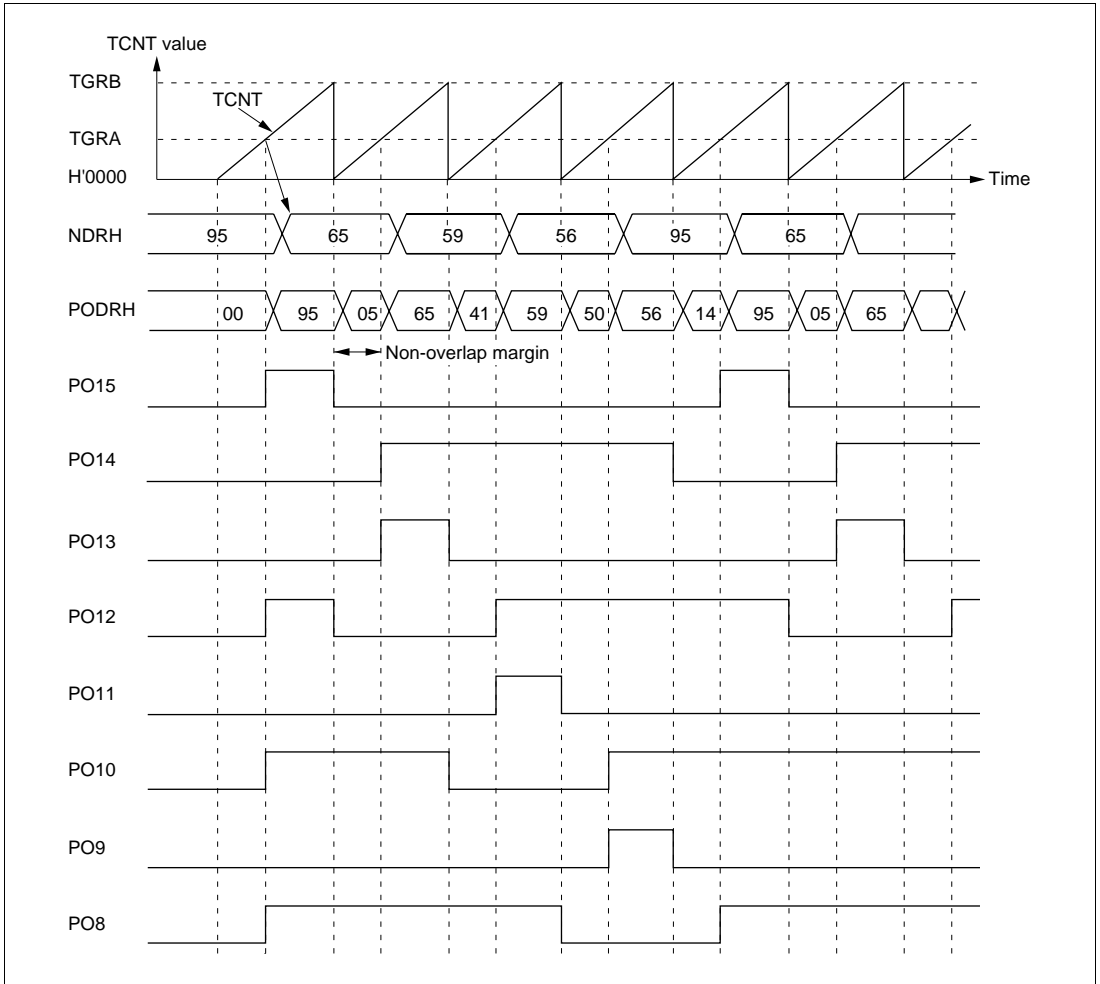
### 8.3.4 Non-Overlapping Pulse Output

**Sample Setup Procedure for Non-Overlapping Pulse Output:** Figure 8-6 shows a sample procedure for setting up non-overlapping pulse output.



**Figure 8-6 Setup Procedure for Non-Overlapping Pulse Output (Example)**

**Example of Non-Overlapping Pulse Output (Example of Four-Phase Complementary Non-Overlapping Output):** Figure 8-7 shows an example in which pulse output is used for four-phase complementary non-overlapping pulse output.



**Figure 8-7 Non-Overlapping Pulse Output Example (Four-Phase Complementary)**

- [1] Set up the TPU channel to be used as the output trigger channel so that TGRA and TGRB are output compare registers. Set the trigger period in TGRB and the non-overlap margin in TGRA, and set the counter to be cleared by compare match B. Set the TGIEA bit in TIER to 1 to enable the TGIA interrupt.
- [2] Write H'FF in P1DDR and NDERH, and set the G3CMS1, G3CMS0, G2CMS1, and G2CMS0 bits in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Set the G3NOV and G2NOV bits in PMR to 1 to select non-overlapping output. Write output data H'95 in NDRH.
- [3] The timer counter in the TPU channel starts. When a compare match with TGRB occurs, outputs change from 1 to 0. When a compare match with TGRA occurs, outputs change from 0 to 1 (the change from 0 to 1 is delayed by the value set in TGRA). The TGIA interrupt handling routine writes the next output data (H'65) in NDRH.
- [4] Four-phase complementary non-overlapping pulse output can be obtained subsequently by writing H'59, H'56, H'95... at successive TGIA interrupts. If the DTC or DMAC is set for activation by this interrupt, pulse output can be obtained without imposing a load on the CPU.

### 8.3.5 Inverted Pulse Output

If the G3INV, G2INV, G1INV, and G0INV bits in PMR are cleared to 0, values that are the inverse of the PODR contents can be output.

Figure 8-8 shows the outputs when G3INV and G2INV are cleared to 0, in addition to the settings of figure 8-7.

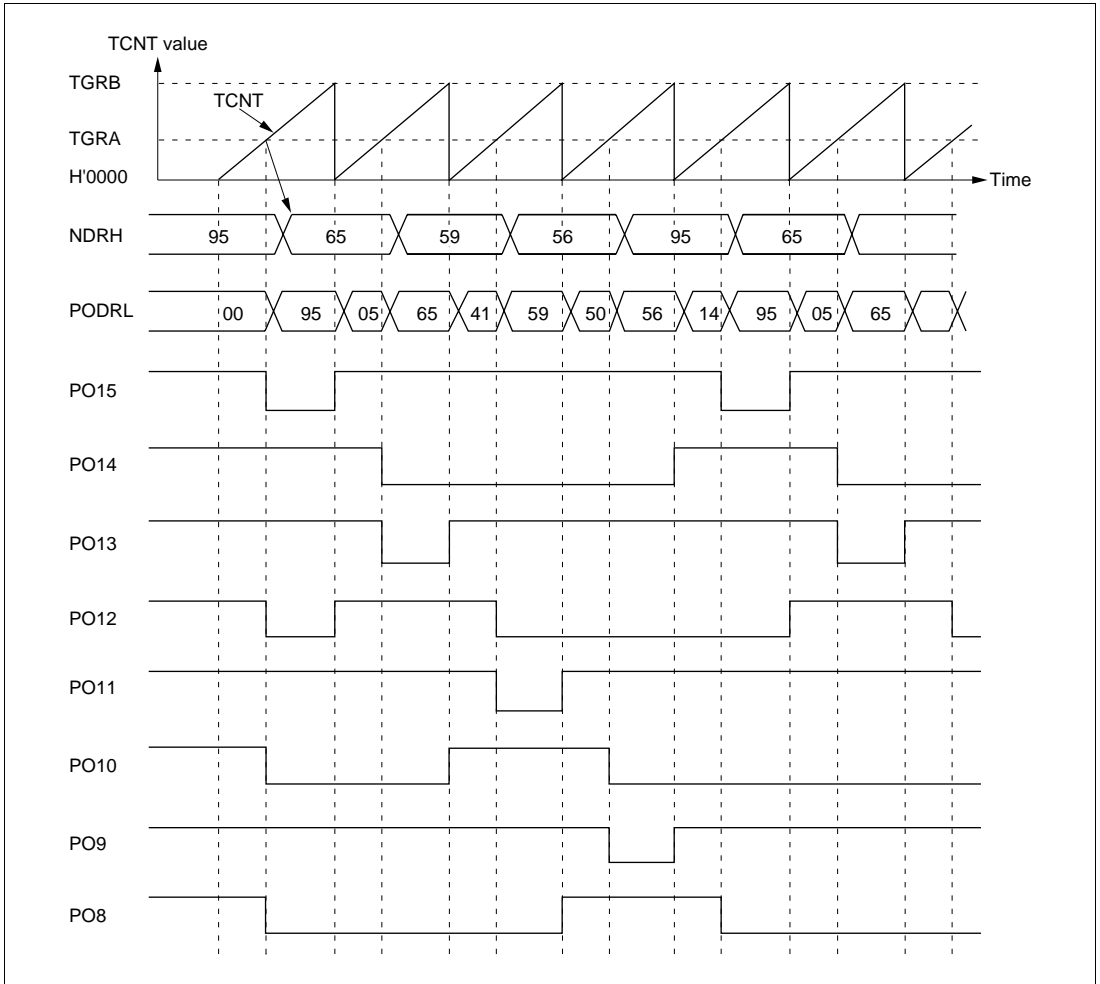
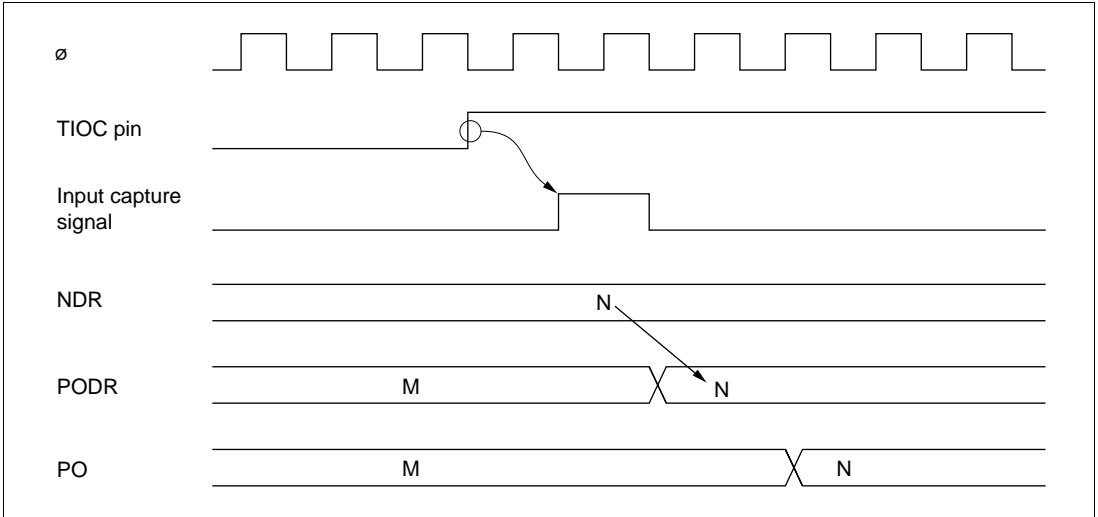


Figure 8-8 Inverted Pulse Output (Example)

### 8.3.6 Pulse Output Triggered by Input Capture

Pulse output can be triggered by TPU input capture as well as by compare match. If TGRA functions as an input capture register in the TPU channel selected by PCR, pulse output will be triggered by the input capture signal.

Figure 8-9 shows the timing of this output.



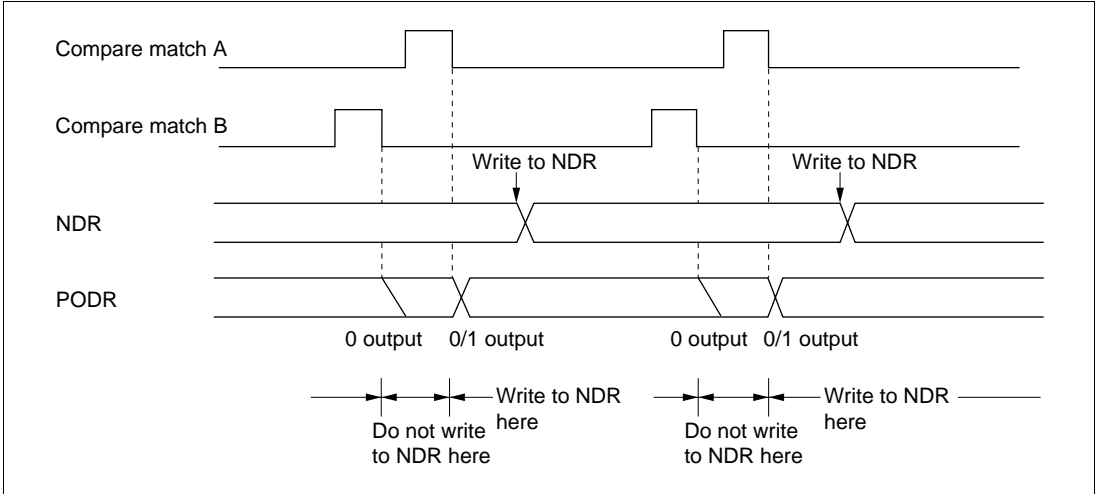
**Figure 8-9 Pulse Output Triggered by Input Capture (Example)**



Therefore, 0 data can be transferred ahead of 1 data by making compare match B occur before compare match A. The NDR contents should not be altered during the interval from compare match B to compare match A (the non-overlap margin).

This can be accomplished by having the TGIA interrupt handling routine write the next data in NDR, or by having the TGIA interrupt activate the DTC or DMAC. Note, however, that the next data must be written before the next compare match B occurs.

Figure 8-11 shows the timing of this operation.



**Figure 8-11 Non-Overlapping Operation and NDR Write Timing**

# Section 9 8-Bit Timers

## 9.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series include an 8-bit timer module with two channels (TMR0 and TMR1). Each channel has an 8-bit counter (TCNT) and two time constant registers (TCORA and TCORB) that are constantly compared with the TCNT value to detect compare match events. The 8-bit timer module can thus be used for a variety of functions, including pulse output with an arbitrary duty cycle.

### 9.1.1 Features

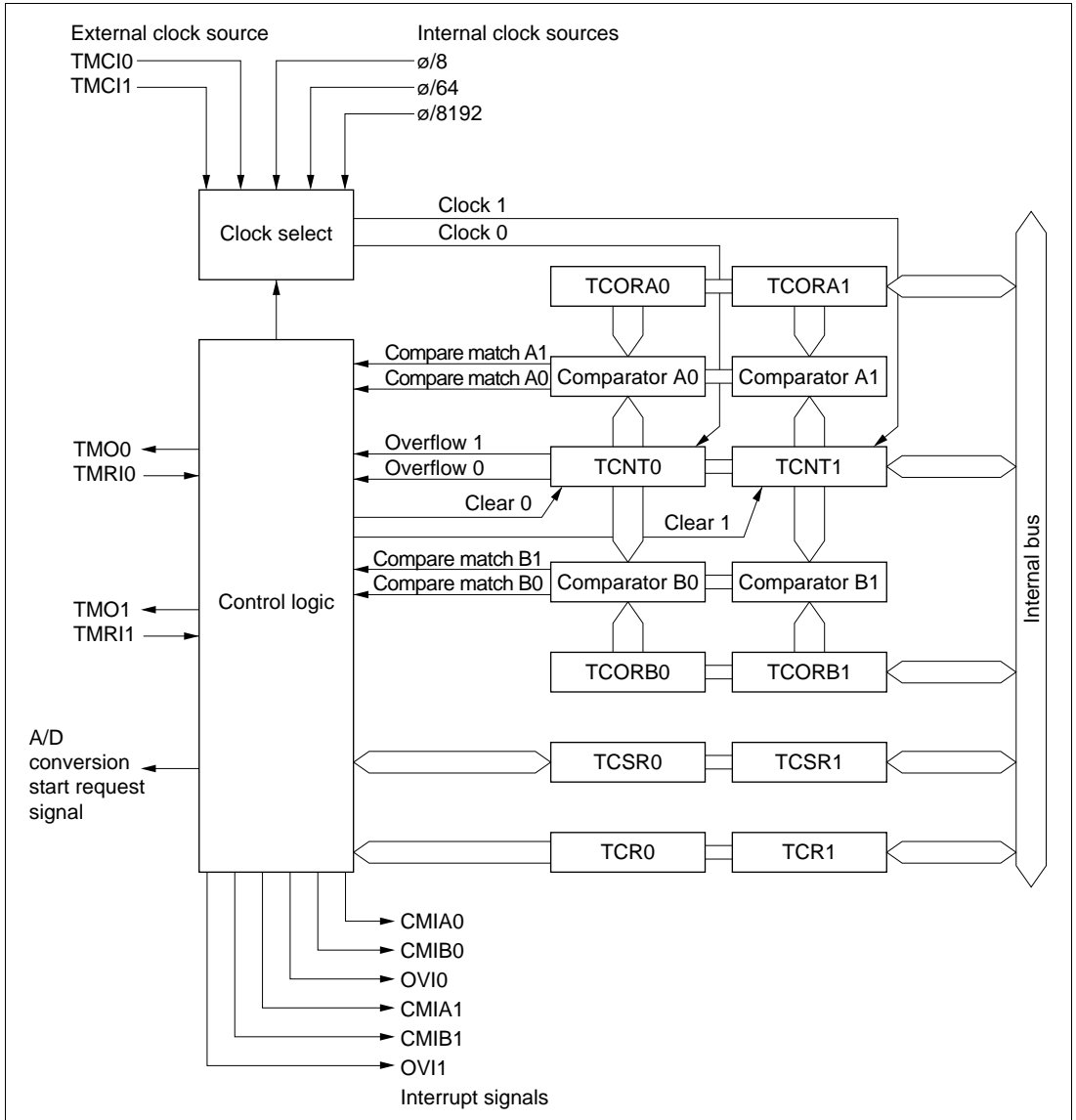
The features of the 8-bit timer module are listed below.

- Selection of four clock sources  
The counters can be driven by one of three internal clock signals ( $\phi/8$ ,  $\phi/64$ , or  $\phi/8192$ ) or an external clock input (enabling use as an external event counter)
- Selection of three ways to clear the counters  
The counters can be cleared on compare match A or B, or by an external reset signal
- Timer output control by a combination of two compare match signals  
The timer output signal in each channel is controlled by a combination of two independent compare match signals, enabling the timer to generate output waveforms with an arbitrary duty cycle or PWM output
- Provision for cascading of two channels
  - Operation as a 16-bit timer is possible, using channel 0 for the upper 8 bits and channel 1 for the lower 8 bits (16-bit count mode)
  - Channel 1 can be used to count channel 0 compare matches (compare match count mode)
- Three independent interrupts  
Compare match A and B and overflow interrupts can be requested independently
- A/D converter conversion start trigger can be generated  
Channel 0 compare match A signal can be used as an A/D converter conversion start trigger
- Module stop mode can be set
  - As the initial setting, 8-bit timer operation is halted. Register access is enabled by exiting module stop mode



## 9.1.2 Block Diagram

Figure 9-1 shows a block diagram of the 8-bit timer module.



**Figure 9-1 Block Diagram of 8-Bit Timer Module**

### 9.1.3 Pin Configuration

Table 9-1 summarizes the input and output pins of the 8-bit timer module.

**Table 9-1 Input and Output Pins of 8-Bit Timer**

Channel	Name	Symbol	I/O	Function
0	Timer output pin 0	TMO0	Output	Outputs at compare match
	Timer clock input pin 0	TMC10	Input	Inputs external clock for counter
	Timer reset input pin 0	TMR10	Input	Inputs external reset to counter
1	Timer output pin 1	TMO1	Output	Outputs at compare match
	Timer clock input pin 1	TMC11	Input	Inputs external clock for counter
	Timer reset input pin 1	TMR11	Input	Inputs external reset to counter

### 9.1.4 Register Configuration

Table 9-2 summarizes the registers of the 8-bit timer module.

**Table 9-2 8-Bit Timer Registers**

Channel	Name	Abbreviation	R/W	Initial value	Address*1
0	Timer control register 0	TCR0	R/W	H'00	H'FFB0
	Timer control/status register 0	TCSR0	R/(W)*2	H'00	H'FFB2
	Time constant register A0	TCORA0	R/W	H'FF	H'FFB4
	Time constant register B0	TCORB0	R/W	H'FF	H'FFB6
	Timer counter 0	TCNT0	R/W	H'00	H'FFB8
1	Timer control register 1	TCR1	R/W	H'00	H'FFB1
	Timer control/status register 1	TCSR1	R/(W)*2	H'10	H'FFB3
	Time constant register A1	TCORA1	R/W	H'FF	H'FFB5
	Time constant register B1	TCORB1	R/W	H'FF	H'FFB7
	Timer counter 1	TCNT1	R/W	H'00	H'FFB9
All	Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

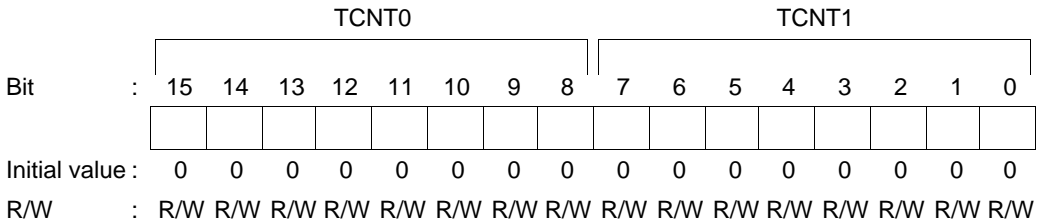
Notes: 1. Lower 16 bits of the address

2. Only 0 can be written to bits 7 to 5, to clear these flags.

Each pair of registers for channel 0 and channel 1 is a 16-bit register with the upper 8 bits for channel 0 and the lower 8 bits for channel 1, so they can be accessed together by a word transfer instruction.

## 9.2 Register Descriptions

### 9.2.1 Timer Counters 0 and 1 (TCNT0, TCNT1)



TCNT0 and TCNT1 are 8-bit readable/writable up-counters that increment on pulses generated from an internal or external clock source. This clock source is selected by clock select bits CKS2 to CKS0 in TCR. The CPU can read or write to TCNT0 and TCNT1 at all times.

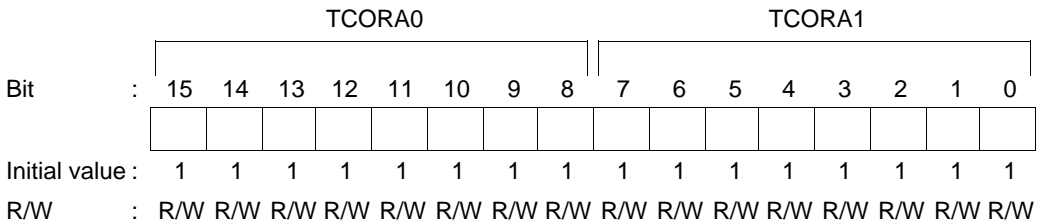
TCNT0 and TCNT1 comprise a single 16-bit register, so they can be accessed together by a word transfer instruction.

TCNT0 and TCNT1 can be cleared by an external reset input or by a compare match signal. Which signal is to be used for clearing is selected by clock clear bits CCLR1 and CCLR0 in TCR.

When a timer counter overflows from H'FF to H'00, OVF in TCSR is set to 1.

TCNT0 and TCNT1 are each initialized to H'00 by a reset and in hardware standby mode.

### 9.2.2 Time Constant Registers A0 and A1 (TCORA0, TCORA1)



TCORA0 and TCORA1 are 8-bit readable/writable registers. TCORA0 and TCORA1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction.

TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding CMFA flag in TCSR is set. Note, however, that comparison is disabled during the T2 state of a TCOR write cycle.

The timer output can be freely controlled by these compare match signals and the settings of bits OS1 and OS0 in TCSR.

TCORA0 and TCORA1 are each initialized to H'FF by a reset and in hardware standby mode.

### 9.2.3 Time Constant Registers B0 and B1 (TCORB0, TCORB1)

	TCORB0	TCORB1
Bit	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Initial value :	1 1 1 1 1 1 1 1	1 1 1 1 1 1 1 1
R/W	R/W R/W R/W R/W R/W R/W R/W R/W	R/W R/W R/W R/W R/W R/W R/W R/W

TCORB0 and TCORB1 are 8-bit readable/writable registers. TCORB0 and TCORB1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction.

TCORB is continually compared with the value in TCNT. When a match is detected, the corresponding CMFB flag in TCSR is set. Note, however, that comparison is disabled during the T2 state of a TCOR write cycle.

The timer output can be freely controlled by these compare match signals and the settings of output select bits OS3 and OS2 in TCSR.

TCORB0 and TCORB1 are each initialized to H'FF by a reset and in hardware standby mode.

### 9.2.4 Time Control Registers 0 and 1 (TCR0, TCR1)

Bit	7	6	5	4	3	2	1	0
	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
Initial value :	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TCR0 and TCR1 are 8-bit readable/writable registers that select the clock source and the time at which TCNT is cleared, and enable interrupts.

TCR0 and TCR1 are each initialized to H'00 by a reset and in hardware standby mode.

For details of this timing, see section 9.3, Operation.

**Bit 7—Compare Match Interrupt Enable B (CMIEB):** Selects whether CMFB interrupt requests (CMIB) are enabled or disabled when the CMFB flag in TCSR is set to 1.

Bit 7 CMIEB	Description	
0	CMFB interrupt requests (CMIB) are disabled	(Initial value)
1	CMFB interrupt requests (CMIB) are enabled	

**Bit 6—Compare Match Interrupt Enable A (CMIEA):** Selects whether CMFA interrupt requests (CMIA) are enabled or disabled when the CMFA flag in TCSR is set to 1.

Bit 6 CMIEA	Description	
0	CMFA interrupt requests (CMIA) are disabled	(Initial value)
1	CMFA interrupt requests (CMIA) are enabled	

**Bit 5—Timer Overflow Interrupt Enable (OVIE):** Selects whether OVF interrupt requests (OVI) are enabled or disabled when the OVF flag in TCSR is set to 1.

Bit 5 OVIE	Description	
0	OVF interrupt requests (OVI) are disabled	(Initial value)
1	OVF interrupt requests (OVI) are enabled	

**Bits 4 and 3—Counter Clear 1 and 0 (CCLR1 and CCLR0):** These bits select the method by which TCNT is cleared: by compare match A or B, or by an external reset input.

Bit 4 CCLR1	Bit 3 CCLR0	Description	
0	0	Clearing is disabled	(Initial value)
	1	Clear by compare match A	
1	0	Clear by compare match B	
	1	Clear by rising edge of external reset input	

**Bits 2 to 0—Clock Select 2 to 0 (CKS2 to CKS0):** These bits select whether the clock input to TCNT is an internal or external clock.

Three internal clocks can be selected, all divided from the system clock ( $\phi$ ):  $\phi/8$ ,  $\phi/64$ , and  $\phi/8192$ . The falling edge of the selected internal clock triggers the count.

When use of an external clock is selected, three types of count can be selected: at the rising edge, the falling edge, and both rising and falling edges.

Some functions differ between channel 0 and channel 1.

Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Description
0	0	0	Clock input disabled (Initial value)
		1	Internal clock, counted at falling edge of $\phi/8$
	1	0	Internal clock, counted at falling edge of $\phi/64$
		1	Internal clock, counted at falling edge of $\phi/8192$
1	0	0	For channel 0: count at TCNT1 overflow signal* For channel 1: count at TCNT0 compare match A*
		1	External clock, counted at rising edge
	1	0	External clock, counted at falling edge
		1	External clock, counted at both rising and falling edges

Note: \* If the count input of channel 0 is the TCNT1 overflow signal and that of channel 1 is the TCNT0 compare match signal, no incrementing clock is generated. Do not use this setting.

### 9.2.5 Timer Control/Status Registers 0 and 1 (TCSR0, TCSR1)

#### TCSR0

Bit	:	7	6	5	4	3	2	1	0
		CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	R/W	R/W	R/W	R/W	R/W

#### TCSR1

Bit	:	7	6	5	4	3	2	1	0
		CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0
Initial value :		0	0	0	1	0	0	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	—	R/W	R/W	R/W	R/W

Note: \* Only 0 can be written to bits 7 to 5, to clear these flags.

TCSR0 and TCSR1 are 8-bit registers that display compare match and overflow statuses, and control compare match output.

TCSR0 is initialized to H'00, and TCSR1 to H'10, by a reset and in hardware standby mode.

**Bit 7—Compare Match Flag B (CMFB):** Status flag indicating whether the values of TCNT and TCORB match.

Bit 7 CMFB	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• Cleared by reading CMFB when CMFB = 1, then writing 0 to CMFB</li><li>• When DTC is activated by CMIB interrupt while DISEL bit of MRB in DTC is 0</li></ul>
1	[Setting condition] Set when TCNT matches TCORB

**Bit 6—Compare Match Flag A (CMFA):** Status flag indicating whether the values of TCNT and TCORA match.

Bit 6 CMFA	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• Cleared by reading CMFA when CMFA = 1, then writing 0 to CMFA</li><li>• When DTC is activated by CMIA interrupt while DISEL bit of MRB in DTC is 0</li></ul>
1	[Setting condition] Set when TCNT matches TCORA

**Bit 5—Timer Overflow Flag (OVF):** Status flag indicating that TCNT has overflowed (changed from H'FF to H'00).

Bit 5 OVF	Description
0	[Clearing condition] (Initial value) <ul style="list-style-type: none"><li>• Cleared by reading OVF when OVF = 1, then writing 0 to OVF</li></ul>
1	[Setting condition] Set when TCNT overflows from H'FF to H'00

**Bit 4—A/D Trigger Enable (ADTE) (TCSR0 Only):** Selects enabling or disabling of A/D converter start requests by compare match A.

In TCSR1, this bit is reserved: it is always read as 1 and cannot be modified.

**Bit 4**

ADTE	Description	
0	A/D converter start requests by compare match A are disabled	(Initial value)
1	A/D converter start requests by compare match A are enabled	

**Bits 3 to 0—Output Select 3 to 0 (OS3 to OS0):** These bits specify how the timer output level is to be changed by a compare match of TCOR and TCNT.

Bits OS3 and OS2 select the effect of compare match B on the output level, bits OS1 and OS0 select the effect of compare match A on the output level, and both of them can be controlled independently.

Note, however, that priorities are set such that: toggle output > 1 output > 0 output. If compare matches occur simultaneously, the output changes according to the compare match with the higher priority.

Timer output is disabled when bits OS3 to OS0 are all 0.

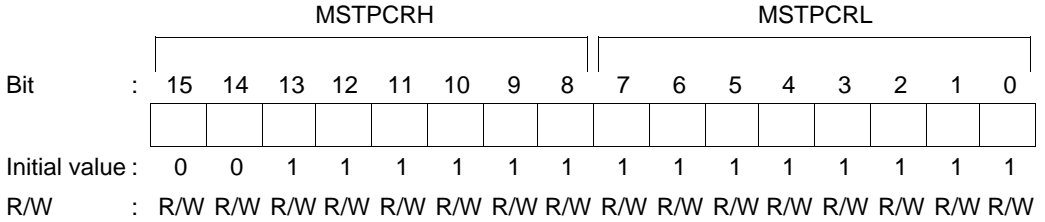
After a reset, the timer output is 0 until the first compare match event occurs.

Bit 3 OS3	Bit 2 OS2	Description	
0	0	No change when compare match B occurs	(Initial value)
	1	0 is output when compare match B occurs	
1	0	1 is output when compare match B occurs	
	1	Output is inverted when compare match B occurs (toggle output)	

Bit 1 OS1	Bit 0 OS0	Description	
0	0	No change when compare match A occurs	(Initial value)
	1	0 is output when compare match A occurs	
1	0	1 is output when compare match A occurs	
	1	Output is inverted when compare match A occurs (toggle output)	



## 9.2.6 Module Stop Control Register (MSTPCR)



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP12 bit in MSTPCR is set to 1, the 8-bit timer operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 12—Module Stop (MSTP12):** Specifies the 8-bit timer module stop mode.

### Bit 12

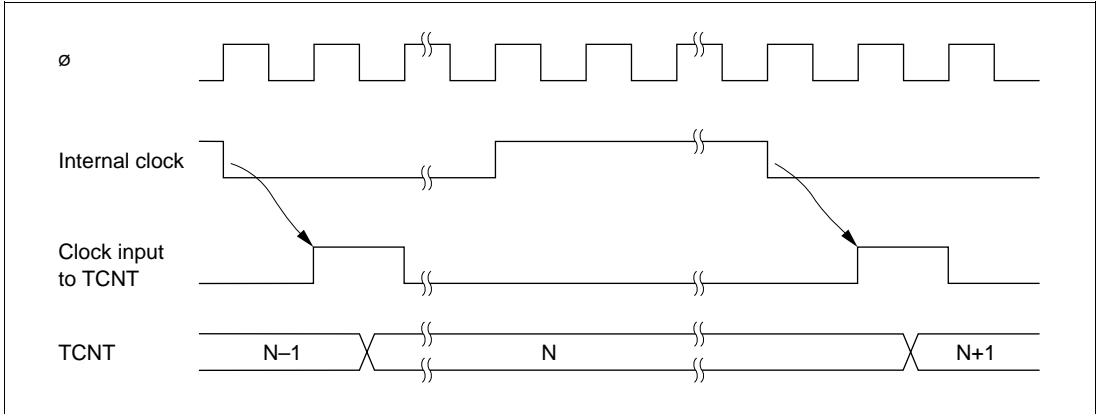
MSTP12	Description
0	8-bit timer module stop mode cleared
1	8-bit timer module stop mode set <span style="float: right;">(Initial value)</span>

## 9.3 Operation

### 9.3.1 TCNT Incrementation Timing

TCNT is incremented by input clock pulses (either internal or external).

**Internal Clock:** Three different internal clock signals ( $\phi/8$ ,  $\phi/64$ , or  $\phi/8192$ ) divided from the system clock ( $\phi$ ) can be selected, by setting bits CKS2 to CKS0 in TCR. Figure 9-2 shows the count timing.

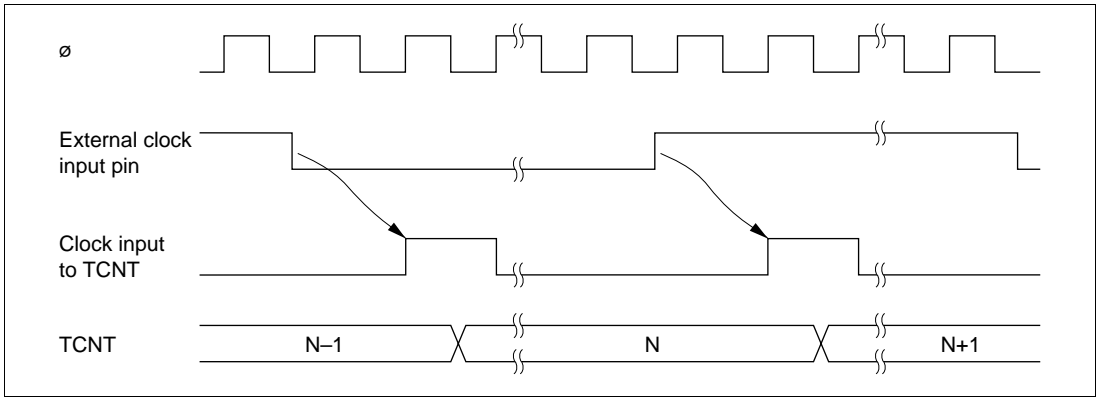


**Figure 9-2 Count Timing for Internal Clock Input**

**External Clock:** Three incrementation methods can be selected by setting bits CKS2 to CKS0 in TCR: at the rising edge, the falling edge, and both rising and falling edges.

Note that the external clock pulse width must be at least 1.5 states for incrementation at a single edge, and at least 2.5 states for incrementation at both edges. The counter will not increment correctly if the pulse width is less than these values.

Figure 9-3 shows the timing of incrementation at both edges of an external clock signal.

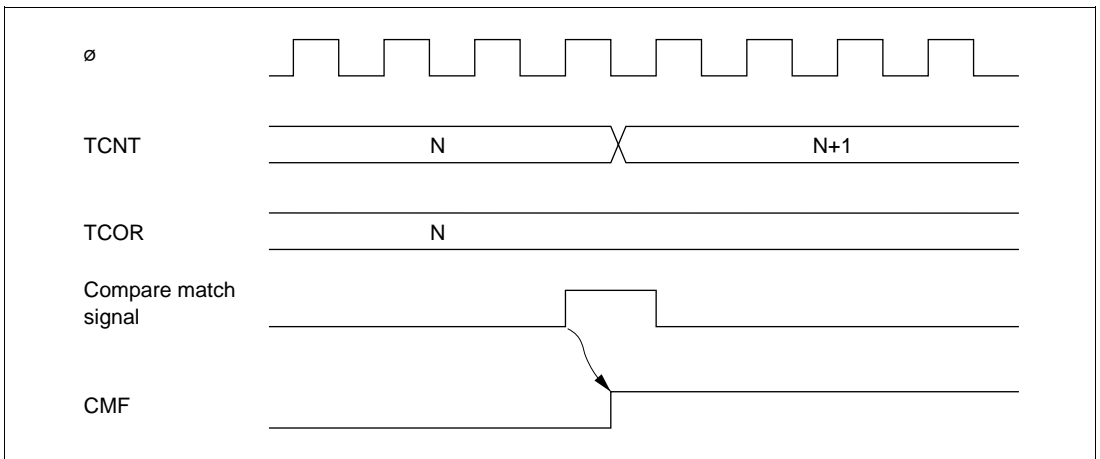


**Figure 9-3 Count Timing for External Clock Input**

### 9.3.2 Compare Match Timing

**Setting of Compare Match Flags A and B (CMFA, CMFB):** The CMFA and CMFB flags in TCSR are set to 1 by a compare match signal generated when the TCOR and TCNT values match. The compare match signal is generated at the last state in which the match is true, just before the timer counter is updated.

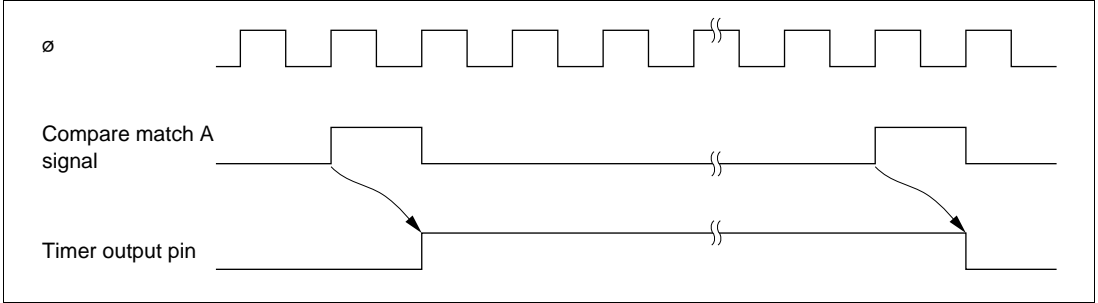
Therefore, when TCOR and TCNT match, the compare match signal is not generated until the next incrementation clock input. Figure 9-4 shows this timing.



**Figure 9-4 Timing of CMF Setting**

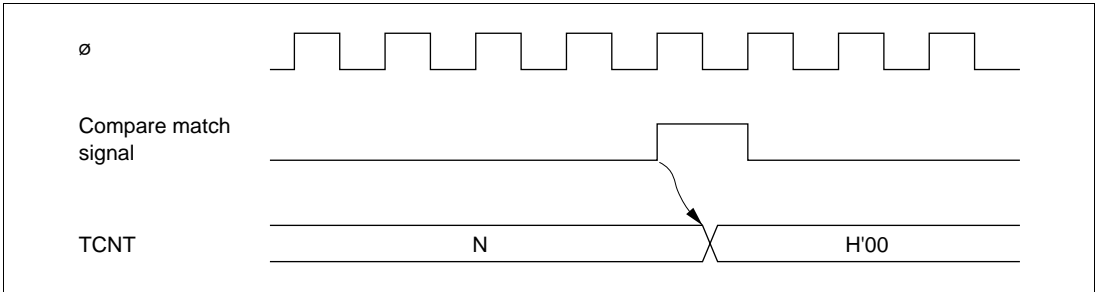
**Timer Output Timing:** When compare match A or B occurs, the timer output changes as specified by bits OS3 to OS0 in TCSR. Depending on these bits, the output can remain the same, change to 0, change to 1, or toggle.

Figure 9-5 shows the timing when the output is set to toggle at compare match A.



**Figure 9-5 Timing of Timer Output**

**Timing of Compare Match Clear:** The timer counter is cleared when compare match A or B occurs, depending on the setting of the CCLR1 and CCLR0 bits in TCR. Figure 9-6 shows the timing of this operation.



**Figure 9-6 Timing of Compare Match Clear**

### 9.3.3 Timing of TCNT External Reset

TCNT is cleared at the rising edge of an external reset input, depending on the settings of the CCLR1 and CCLR0 bits in TCR. The clear pulse width must be at least 1.5 states. Figure 9-7 shows the timing of this operation.

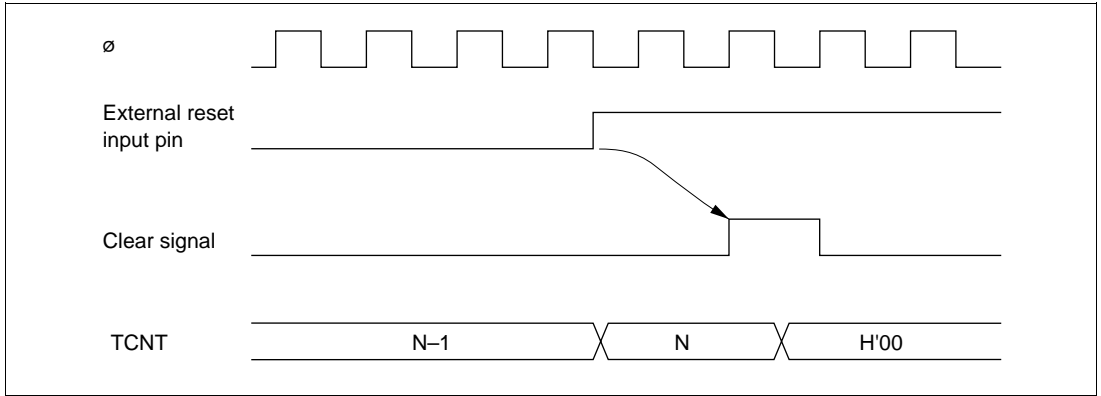


Figure 9-7 Timing of Clearance by External Reset

### 9.3.4 Timing of Overflow Flag (OVF) Setting

The OVF in TCSR is set to 1 when TCNT overflows (changes from H'FF to H'00). Figure 9-8 shows the timing of this operation.

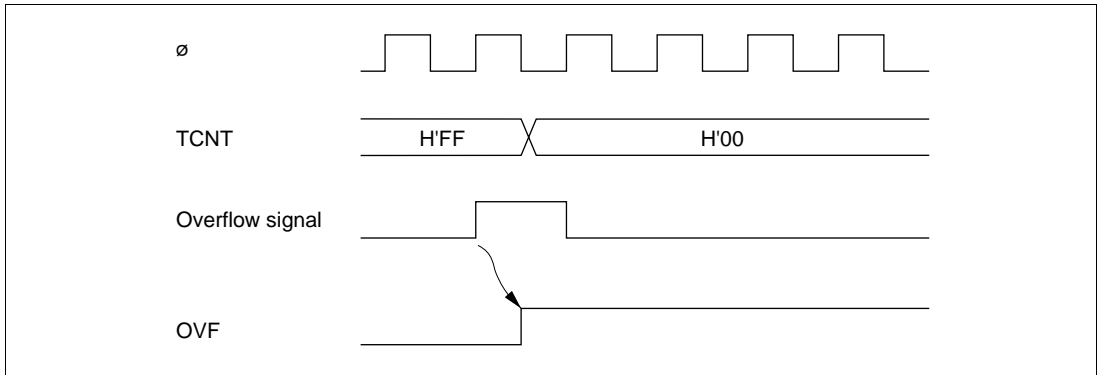


Figure 9-8 Timing of OVF Setting

### 9.3.5 Operation with Cascaded Connection

If bits CKS2 to CKS0 in either TCR0 or TCR1 are set to B'100, the 8-bit timers of the two channels are cascaded. With this configuration, a single 16-bit timer could be used (16-bit timer mode) or compare matches of the 8-bit channel 0 could be counted by the timer of channel 1 (compare match counter mode). In this case, the timer operates as below.

**16-Bit Counter Mode:** When bits CKS2 to CKS0 in TCR0 are set to B'100, the timer functions as a single 16-bit timer with channel 0 occupying the upper 8 bits and channel 1 occupying the lower 8 bits.

- Setting of compare match flags
  - The CMF flag in TCSR0 is set to 1 when a 16-bit compare match event occurs.
  - The CMF flag in TCSR1 is set to 1 when a lower 8-bit compare match event occurs.
- Counter clear specification
  - If the CCLR1 and CCLR0 bits in TCR0 have been set for counter clear at compare match, the 16-bit counter (TCNT0 and TCNT1 together) is cleared when a 16-bit compare match event occurs. The 16-bit counter (TCNT0 and TCNT1 together) is cleared even if counter clear by the TMRI0 pin has also been set.
  - The settings of the CCLR1 and CCLR0 bits in TCR1 are ignored. The lower 8 bits cannot be cleared independently.
- Pin output
  - Control of output from the TMO0 pin by bits OS3 to OS0 in TCSR0 is in accordance with the 16-bit compare match conditions.
  - Control of output from the TMO1 pin by bits OS3 to OS0 in TCSR1 is in accordance with the lower 8-bit compare match conditions.

**Compare Match Counter Mode:** When bits CKS2 to CKS0 in TCR1 are B'100, TCNT1 counts compare match A's for channel 0.

Channels 0 and 1 are controlled independently. Conditions such as setting of the CMF flag, generation of interrupts, output from the TMO pin, and counter clear are in accordance with the settings for each channel.

**Usage Note:** If the 16-bit counter mode and compare match counter mode are set simultaneously, the input clock pulses for TCNT0 and TCNT1 are not generated and thus the counters will stop operating. Software should therefore avoid using both these modes.

## 9.4 Interrupts

### 9.4.1 Interrupt Sources and DTC Activation

There are three 8-bit timer interrupt sources: CMIA, CMIB, and OVI. Their relative priorities are shown in table 9-3. Each interrupt source is set as enabled or disabled by the corresponding interrupt enable bit in TCR, and independent interrupt requests are sent for each to the interrupt controller. It is also possible to activate the DTC by means of CMIA and CMIB interrupts.

**Table 9-3 8-Bit Timer Interrupt Sources**

Channel	Interrupt Source	Description	DTC Activation	Priority
0	CMIA0	Interrupt by CMFA	Possible	High ↑
	CMIB0	Interrupt by CMFB	Possible	
	OVI0	Interrupt by OVF	Not possible	
1	CMIA1	Interrupt by CMFA	Possible	Low ↑
	CMIB1	Interrupt by CMFB	Possible	
	OVI1	Interrupt by OVF	Not possible	

Note: This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

### 9.4.2 A/D Converter Activation

The A/D converter can be activated only by channel 0 compare match A.

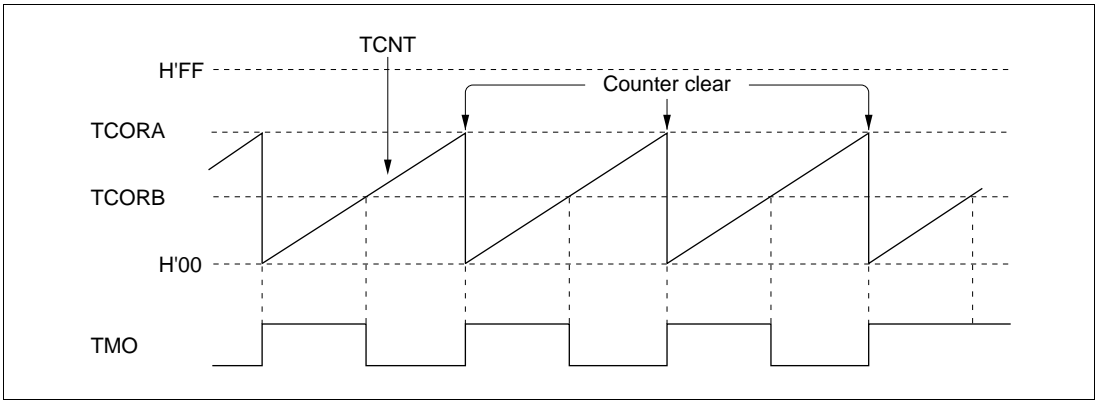
If the ADTE bit in TCSR0 is set to 1 when the CMFA flag is set to 1 by the occurrence of channel 0 compare match A, a request to start A/D conversion is sent to the A/D converter. If the 8-bit timer conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

## 9.5 Sample Application

In the example below, the 8-bit timer is used to generate a pulse output with a selected duty cycle, as shown in figure 9-9. The control bits are set as follows:

- [1] In TCR, bit CCLR1 is cleared to 0 and bit CCLR0 is set to 1 so that the timer counter is cleared when its value matches the constant in TCORA.
- [2] In TCSR, bits OS3 to OS0 are set to B'0110, causing the output to change to 1 at a TCORA compare match and to 0 at a TCORB compare match.

With these settings, the 8-bit timer provides output of pulses at a rate determined by TCORA with a pulse width determined by TCORB. No software intervention is required.



**Figure 9-9 Example of Pulse Output**

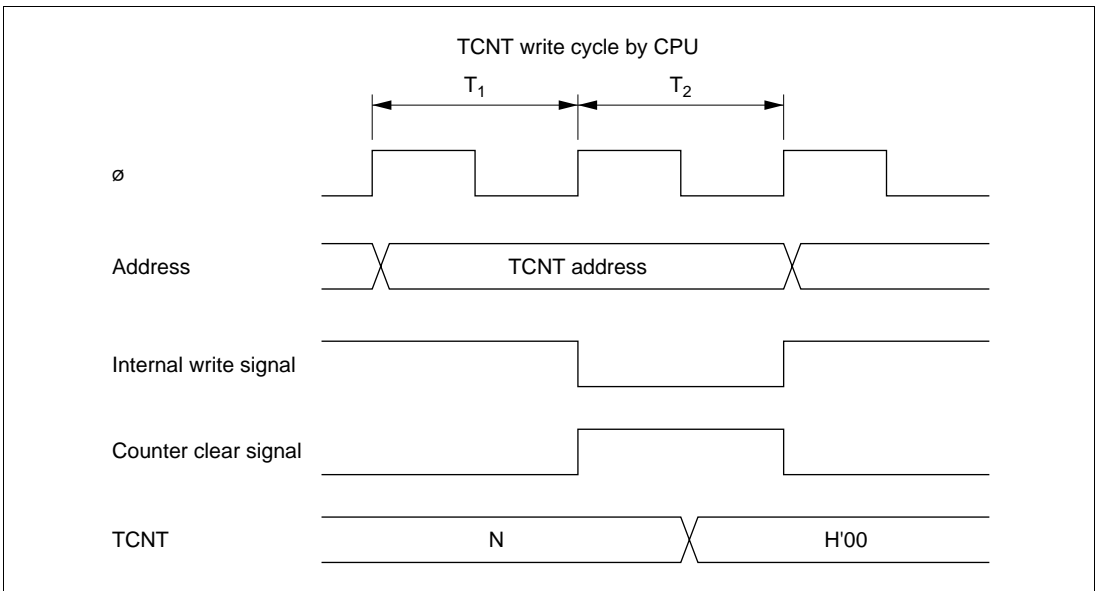
## 9.6 Usage Notes

Note that the following kinds of contention can occur in the 8-bit timer module.

### 9.6.1 Contention between TCNT Write and Clear

If a timer counter clock pulse is generated during the  $T_2$  state of a TCNT write cycle, the clear takes priority, so that the counter is cleared and the write is not performed.

Figure 9-10 shows this operation.



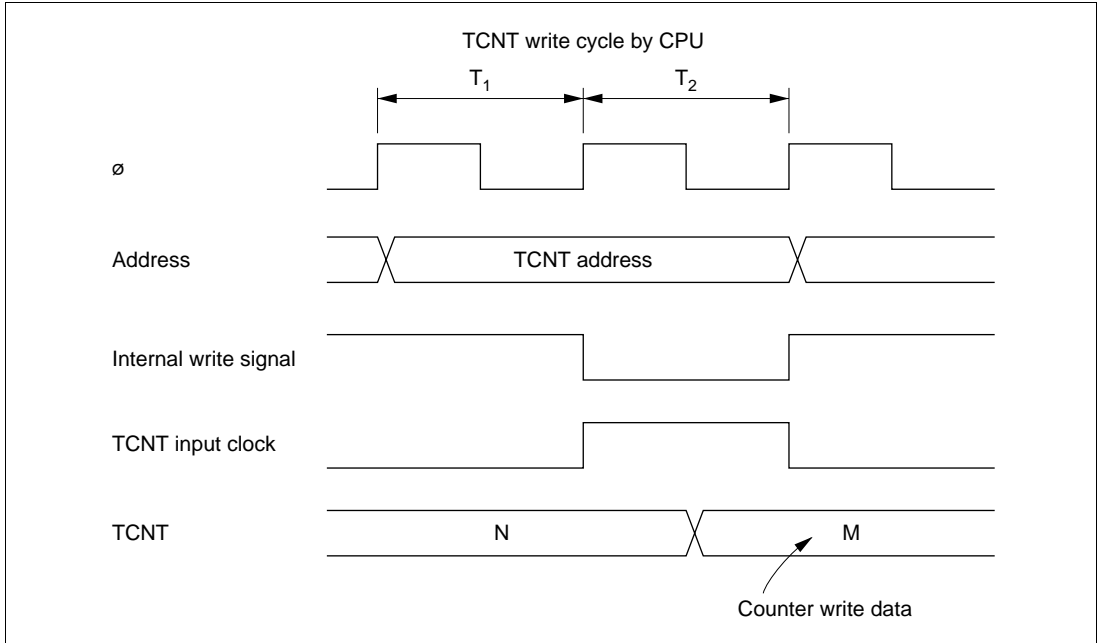
**Figure 9-10 Contention between TCNT Write and Clear**



## 9.6.2 Contention between TCNT Write and Increment

If a timer counter clock pulse is generated during the  $T_2$  state of a TCNT write cycle, the write takes priority and the counter is not incremented.

Figure 9-11 shows this operation.

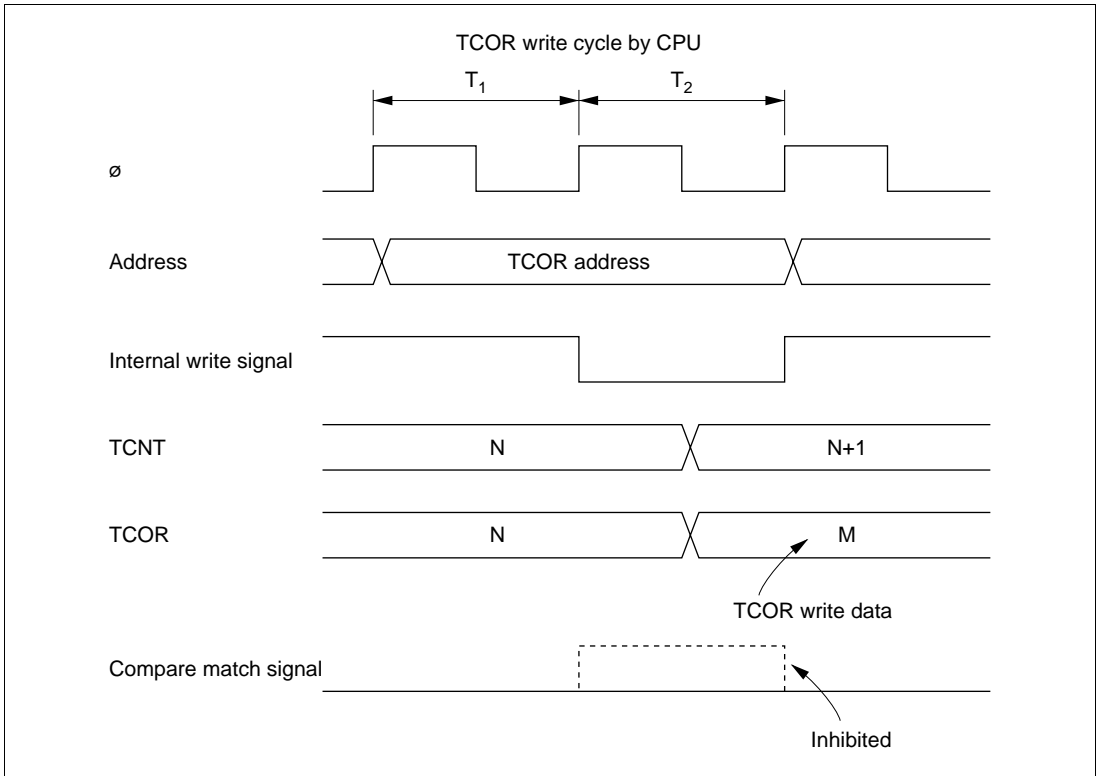


**Figure 9-11 Contention between TCNT Write and Increment**

### 9.6.3 Contention between TCOR Write and Compare Match

During the  $T_2$  state of a TCOR write cycle, the TCOR write has priority and the compare match signal is inhibited even if a compare match event occurs.

Figure 9-12 shows this operation.



**Figure 9-12** Contention between TCOR Write and Compare Match

#### 9.6.4 Contention between Compare Matches A and B

If compare match events A and B occur at the same time, the 8-bit timer operates in accordance with the priorities for the output statuses set for compare match A and compare match B, as shown in table 9-4.

**Table 9-4 Timer Output Priorities**

Output Setting	Priority
Toggle output	High
1 output	▲ ↑
0 output	
No change	Low

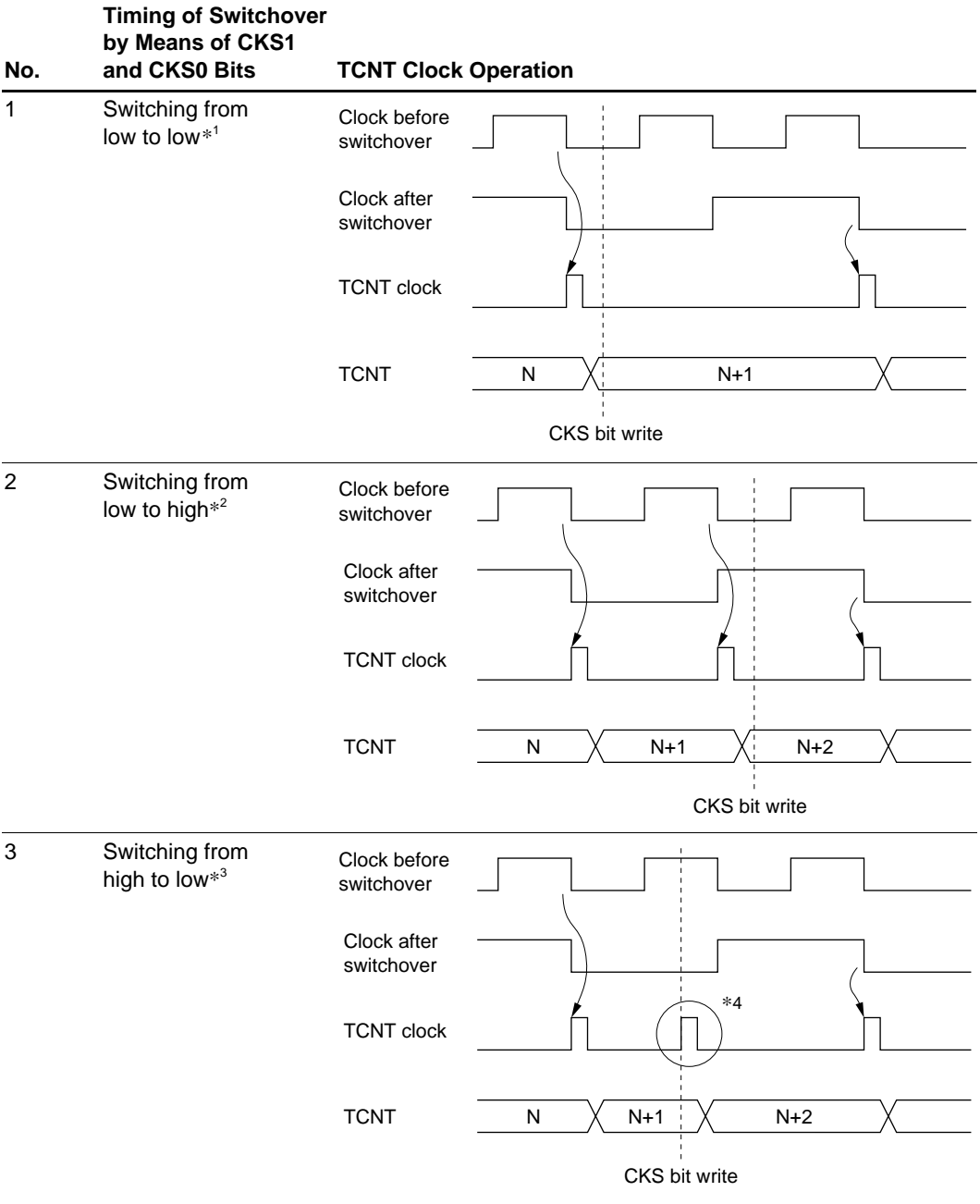
#### 9.6.5 Switching of Internal Clocks and TCNT Operation

TCNT may increment erroneously when the internal clock is switched over. Table 9-5 shows the relationship between the timing at which the internal clock is switched (by writing to the CKS1 and CKS0 bits) and the TCNT operation.

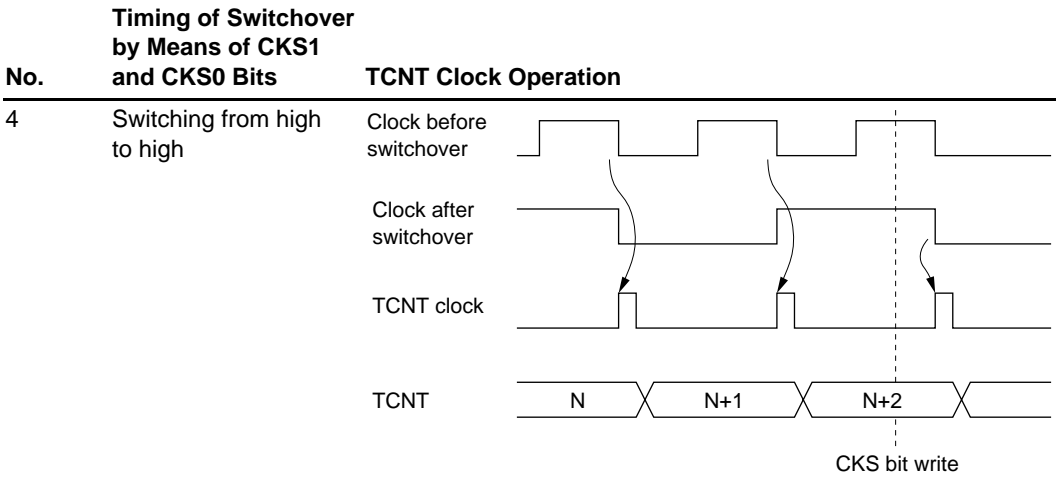
When the TCNT clock is generated from an internal clock, the falling edge of the internal clock pulse is detected. If clock switching causes a change from high to low level, as shown in case 3 in table 9-5, a TCNT clock pulse is generated on the assumption that the switchover is a falling edge. This increments TCNT.

The erroneous incrementation can also happen when switching between internal and external clocks.

**Table 9-5 Switching of Internal Clock and TCNT Operation**



**Table 9-5 Switching of Internal Clock and TCNT Operation (cont)**



- Notes:
1. Includes switching from low to stop, and from stop to low.
  2. Includes switching from stop to high.
  3. Includes switching from high to stop.
  4. Generated on the assumption that the switchover is a falling edge; TCNT is incremented.

### 9.6.6 Interrupts and Module Stop Mode

If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

# Section 10 Watchdog Timer

## 10.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have a single-channel on-chip watchdog timer (WDT) for monitoring system operation. The WDT outputs an overflow signal ( $\overline{\text{WDTOVF}}$ )\* if a system crash prevents the CPU from writing to the timer counter, allowing it to overflow. At the same time, the WDT can also generate an internal reset signal for the chip.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

### 10.1.1 Features

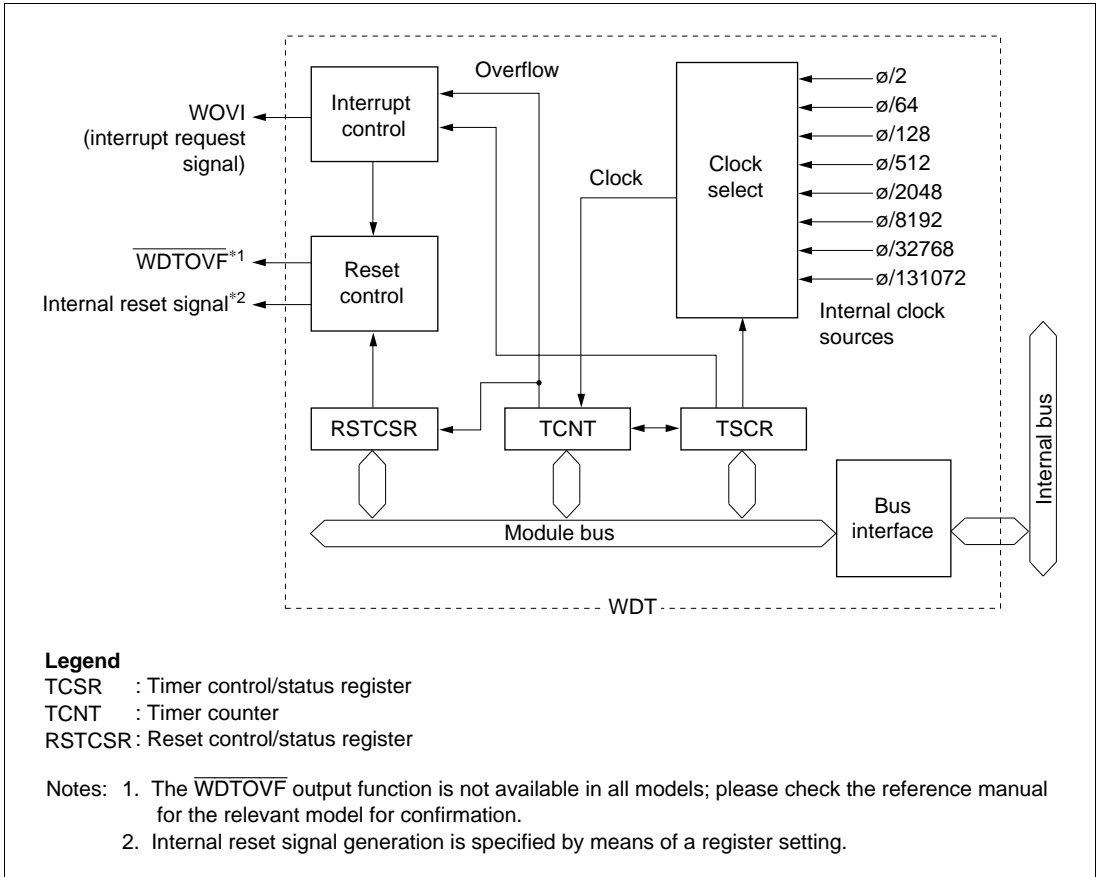
WDT features are listed below.

- Switchable between watchdog timer mode and interval timer mode
- $\overline{\text{WDTOVF}}$  output when in watchdog timer mode\*  
If the counter overflows, the WDT outputs  $\overline{\text{WDTOVF}}$ . It is possible to select whether or not the entire H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip is reset at the same time
- Interrupt generation when in interval timer mode  
If the counter overflows, the WDT generates an interval timer interrupt
- Choice of eight counter clock sources

Note: \* The  $\overline{\text{WDTOVF}}$  output function is not available in all models; please check the reference manual for the relevant model for confirmation.

## 10.1.2 Block Diagram

Figure 10-1 shows a block diagram of the WDT.



**Figure 10-1 Block Diagram of WDT**

### 10.1.3 Pin Configuration

Table 10-1 describes the WDT output pin.

**Table 10-1 WDT Pin**

Name	Symbol	I/O	Function
Watchdog timer overflow	WDTOVF*	Output	Outputs counter overflow signal in watchdog timer mode

Note: \* The WDTOVF output function is not available in all models; please check the reference manual for the relevant model for confirmation.

### 10.1.4 Register Configuration

The WDT has three registers, as summarized in table 10-2. These registers control clock selection, WDT mode switching, and the reset signal.

**Table 10-2 WDT Registers**

Name	Abbreviation	R/W	Initial Value	Address* <sup>1</sup>	
				Write* <sup>2</sup>	Read
Timer control/status register	TCSR	R/(W)* <sup>3</sup>	H'18	H'FFBC	H'FFBC
Timer counter	TCNT	R/W	H'00	H'FFBC	H'FFBD
Reset control/status register	RSTCSR	R/(W)* <sup>3</sup>	H'1F	H'FFBE	H'FFBF

- Notes:
1. Lower 16 bits of the address.
  2. For details of write operations, see section 10.2.4, Notes on Register Access.
  3. Only a write of 0 is permitted to bit 7, to clear the flag.



## 10.2 Register Descriptions

### 10.2.1 Timer Counter (TCNT)

Bit	:	7	6	5	4	3	2	1	0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TCNT is an 8-bit readable/writable\*<sup>1</sup> up-counter.

When the TME bit is set to 1 in TCSR, TCNT starts counting pulses generated from the internal clock source selected by bits CKS2 to CKS0 in TCSR. When the count overflows (changes from H'FF to H'00), either the watchdog timer overflow signal ( $\overline{\text{WDTOVF}}$ )\*<sup>2</sup> or an interval timer interrupt (WOVI) is generated, depending on the mode selected by the  $\overline{\text{WT/IT}}$  bit in TCSR.

TCNT is initialized to H'00 by a reset, in hardware standby mode, or when the TME bit is cleared to 0. It is not initialized in software standby mode.

- Notes:
1. TCNT is write-protected by a password to prevent accidental overwriting. For details see section 10.2.4, Notes on Register Access.
  2. The  $\overline{\text{WDTOVF}}$  output function is not available in all models; please check the reference manual for the relevant model for confirmation.

### 10.2.2 Timer Control/Status Register (TCSR)

Bit	:	7	6	5	4	3	2	1	0
		OVF	$\overline{\text{WT/IT}}$	TME	—	—	CKS2	CKS1	CKS0
Initial value :		0	0	0	1	1	0	0	0
R/W	:	R/(W)*	R/W	R/W	—	—	R/W	R/W	R/W

Note: \* Only 0 can be written, to clear the flag.

TCSR is an 8-bit readable/writable\* register. Its functions include selecting the clock source to be input to TCNT, and the timer mode.

TCR is initialized to H'18 by a reset and in hardware standby mode. It is not initialized in software standby mode.

Note: \* TCSR is write-protected by a password to prevent accidental overwriting. For details see section 10.2.4, Notes on Register Access.

**Bit 7—Overflow Flag (OVF):** Indicates that TCNT has overflowed from H'FF to H'00, when in interval timer mode. This flag cannot be set during watchdog timer operation.

Bit 7 OVF	Description	
0	[Clearing condition] Cleared by reading TCSR when OVF = 1, then writing 0 to OVF	(Initial value)
1	[Setting condition] Set when TCNT overflows (changes from H'FF to H'00) in interval timer mode	

**Bit 6—Timer Mode Select (WT/IT):** Selects whether the WDT is used as a watchdog timer or interval timer. If used as an interval timer, the WDT generates an interval timer interrupt request (WOVI) when TCNT overflows. If used as a watchdog timer, the WDT generates the  $\overline{\text{WDTOVF}}$  signal\*<sup>1</sup> when TCNT overflows.

Bit 6 WT/IT	Description	
0	Interval timer: Sends the CPU an interval timer interrupt request (WOVI) when TCNT overflows	(Initial value)
1	Watchdog timer: Generates the $\overline{\text{WDTOVF}}$ signal* <sup>1</sup> when TCNT overflows* <sup>2</sup>	

- Notes: 1. The  $\overline{\text{WDTOVF}}$  output function is not available in all models; please check the reference manual for the relevant model for confirmation.  
2. For details of the case where TCNT overflows in watchdog timer mode, see section 10.2.3, Reset Control/Status Register (RSTCSR).

**Bit 5—Timer Enable (TME):** Selects whether TCNT runs or is halted.

Bit 5 TME	Description	
0	TCNT is initialized to H'00 and halted	(Initial value)
1	TCNT counts	

**Bits 4 and 3—Reserved:** Read-only bits, always read as 1.

**Bits 2 to 0:** Clock Select 2 to 0 (CKS2 to CKS0): These bits select one of eight internal clock sources, obtained by dividing the system clock ( $\phi$ ), for input to TCNT.

			Description	
Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Clock	Overflow Period (when $\phi = 20 \text{ MHz}$ )*
0	0	0	$\phi/2$ (Initial value)	25.6 $\mu\text{s}$
		1	$\phi/64$	819.2 $\mu\text{s}$
	1	0	$\phi/128$	1.6 ms
		1	$\phi/512$	6.6 ms
1	0	0	$\phi/2048$	26.2 ms
		1	$\phi/8192$	104.9 ms
	1	0	$\phi/32768$	419.4 ms
		1	$\phi/131072$	1.68 s

Note: \* The overflow period is the time from when TCNT starts counting up from H'00 until overflow occurs.

### 10.2.3 Reset Control/Status Register (RSTCSR)

Bit	:	7	6	5	4	3	2	1	0
		WOVF	RSTE	—	—	—	—	—	—
Initial value :		0	0	0	1	1	1	1	1
R/W	:	R/(W)*	R/W	R/W	—	—	—	—	—

Note: \* Only 0 can be written, to clear the flag.

RSTCSR is an 8-bit readable/writable\* register that controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal.

RSTCSR is initialized to H'1F by a reset signal from the  $\overline{\text{RES}}$  pin, but not by the WDT internal reset signal caused by overflows.

Note: \* RSTCSR is write-protected by a password to prevent accidental overwriting. For details see section 10.2.4, Notes on Register Access.

**Bit 7—Watchdog Overflow Flag (WOVF):** Indicates that TCNT has overflowed (changed from H'FF to H'00) during watchdog timer operation. This bit is not set in interval timer mode.

Bit 7 WOVF	Description	
0	[Clearing condition] Cleared by reading TCSR when WOVF = 1, then writing 0 to WOVF	(Initial value)
1	[Setting condition] Set when TCNT overflows (changes from H'FF to H'00) during watchdog timer operation	

**Bit 6—Reset Enable (RSTE):** Specifies whether or not a reset signal is generated in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip if TCNT overflows during watchdog timer operation.

Bit 6 RSTE	Description	
0	Reset signal is not generated if TCNT overflows*	(Initial value)
1	Reset signal is generated if TCNT overflows	

Note: \* The modules within the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip are not reset, but TCNT and TCSR within the WDT are reset.

**Bit 5—Reserved:** Read-only bit.

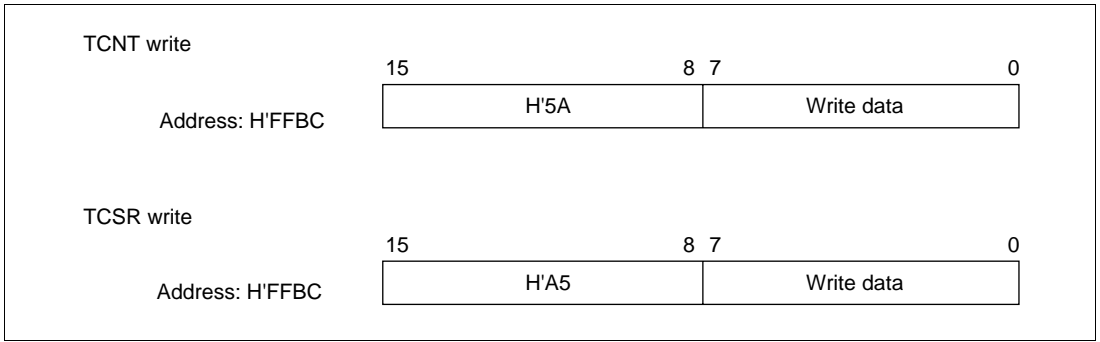
**Bits 4 to 0—Reserved:** Read-only bits, always read as 1.

#### 10.2.4 Notes on Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in being more difficult to write to. The procedures for writing to and reading these registers are given below.

**Writing to TCNT and TCSR:** These registers must be written to by a word transfer instruction. They cannot be written to with byte instructions.

Figure 10-2 shows the format of data written to TCNT and TCSR. TCNT and TCSR both have the same write address. For a write to TCNT, the upper byte of the written word must contain H'5A and the lower byte must contain the write data. For a write to TCSR, the upper byte of the written word must contain H'A5 and the lower byte must contain the write data. This transfers the write data from the lower byte to TCNT or TCSR.

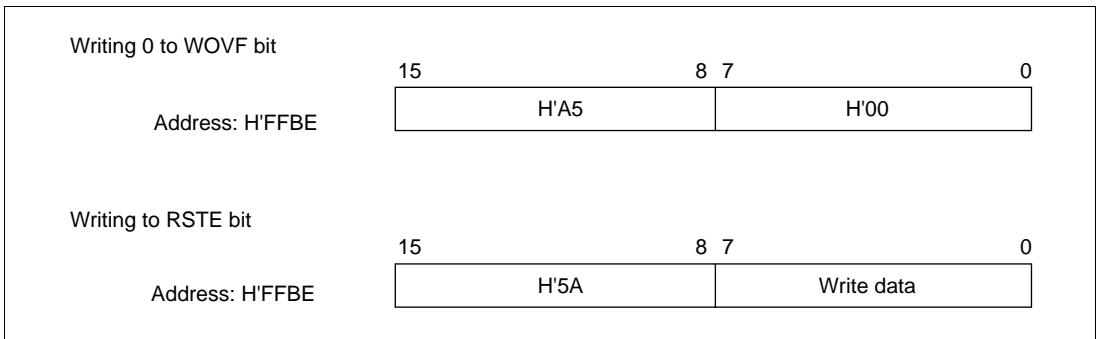


**Figure 10-2 Writing to TCNT and TCSR**

**Writing to RSTCSR:** RSTCSR must be written to by a word transfer instruction to address H'FFBE. It cannot be written to with byte instructions.

Figure 10-3 shows the format of data written to RSTCSR. The method of writing 0 to the WOVF bit differs from that for writing to the RSTE bit.

To write 0 to the WOVF bit, the write data must have H'A5 in the upper byte and H'00 in the lower byte. This clears the WOVF bit to 0, but has no effect on the RSTE bit. To write to the RSTE bit, the upper byte must contain H'5A and the lower byte must contain the write data. This writes the value in bit 6 of the lower byte into the RSTE bit, but has no effect on the WOVF bit.



**Figure 10-3 Writing to RSTCSR**

**Reading TCNT, TCSR, and RSTCSR:** These registers are read in the same way as other registers. The read addresses are H'FFBC for TCSR, H'FFBD for TCNT, and H'FFBF for RSTCSR.

## 10.3 Operation

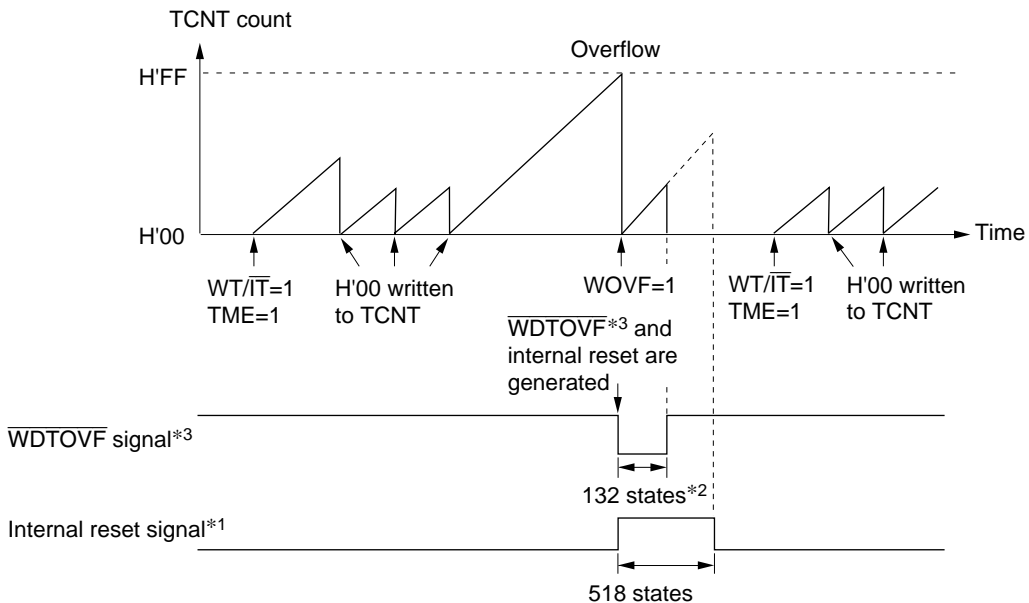
### 10.3.1 Operation in Watchdog Timer Mode

To use the WDT as a watchdog timer, set the  $\overline{WT/IT}$  and TME bits to 1. Software must prevent TCNT overflows by rewriting the TCNT value (normally be writing H'00) before overflow occurs. This ensures that TCNT does not overflow while the system is operating normally. If TCNT overflows without being rewritten because of a system crash or other error, the  $\overline{WDTOVF}$  signal\* is output. This is shown in figure 10-4. This  $\overline{WDTOVF}$  signal\* can be used to reset the system. The  $\overline{WDTOVF}$  signal\* is output for 132 states when RSTE = 1, and for 130 states when RSTE = 0.

If TCNT overflows when 1 is set in the RSTE bit in RSTCSR, a signal that resets the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip internally is generated at the same time as the  $\overline{WDTOVF}$  signal\*<sup>1</sup>. The internal reset signal is output for 518 states.

If a reset caused by a signal input to the  $\overline{RES}$  pin occurs at the same time as a reset caused by a WDT overflow, the  $\overline{RES}$  pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.

Note: \* The  $\overline{WDTOVF}$  output function is not available in all models; please check the reference manual for the relevant model for confirmation.



**Legend**

$WT/\overline{IT}$  : Timer mode select bit  
 $TME$  : Timer enable bit

- Notes: 1. The internal reset signal is generated only if the RSTE bit is set to 1.  
 2. 130 states when the RSTE bit is cleared to 0.  
 3. The  $\overline{WDTOVF}$  output function is not available in all models; please check the reference manual for the relevant model for confirmation.

**Figure 10-4 Operation in Watchdog Timer Mode**

### 10.3.2 Operation in Interval Timer Mode

To use the WDT as an interval timer, clear the  $WT/\overline{IT}$  bit in TCSR to 0 and set the TME bit to 1. An interval timer interrupt (WOVI) is generated each time TCNT overflows, provided that the WDT is operating as an interval timer, as shown in figure 10-5. This function can be used to generate interrupt requests at regular intervals.

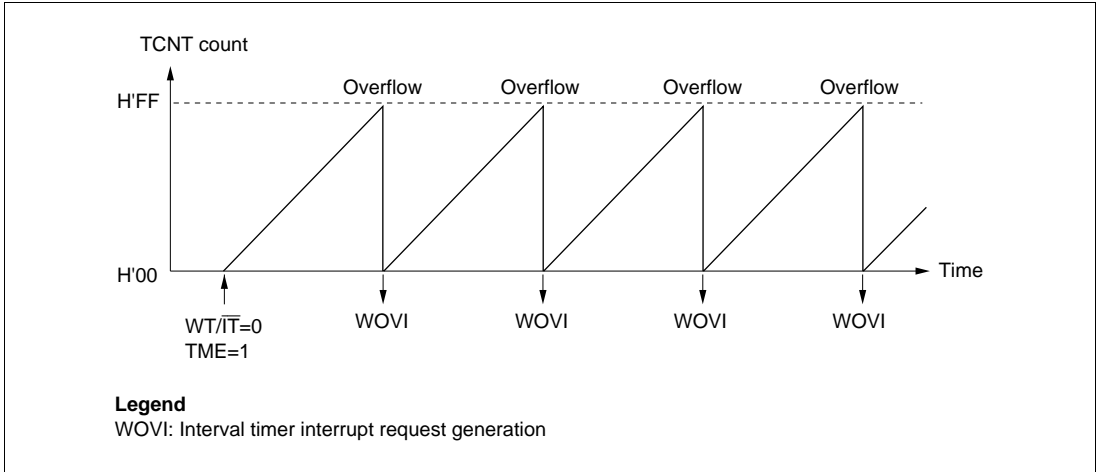


Figure 10-5 Operation in Interval Timer Mode

### 10.3.3 Timing of Overflow Flag (OVF) Setting

The OVF flag is set to 1 if TCNT overflows during interval timer operation. At the same time, an interval timer interrupt (WOVI) is requested. This timing is shown in figure 10-6.

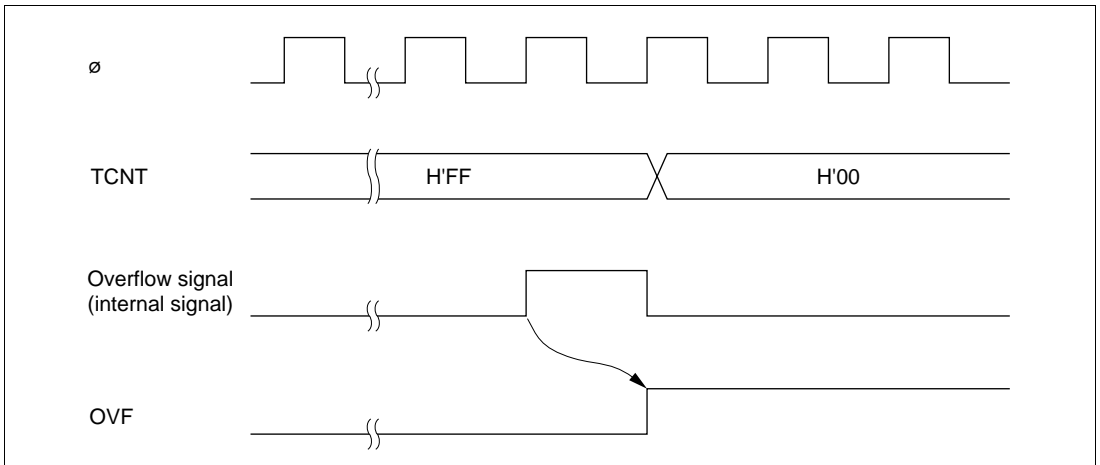


Figure 10-6 Timing of OVF Setting



### 10.3.4 Timing of Watchdog Timer Overflow Flag (WOVF) Setting

The WOVF flag is set to 1 if TCNT overflows during watchdog timer operation. At the same time, the  $\overline{\text{WDTOVF}}$  signal goes low. If TCNT overflows while the RSTE bit in RSTCSR is set to 1, an internal reset signal is generated for the entire H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip. Figure 10-7 shows the timing in this case.

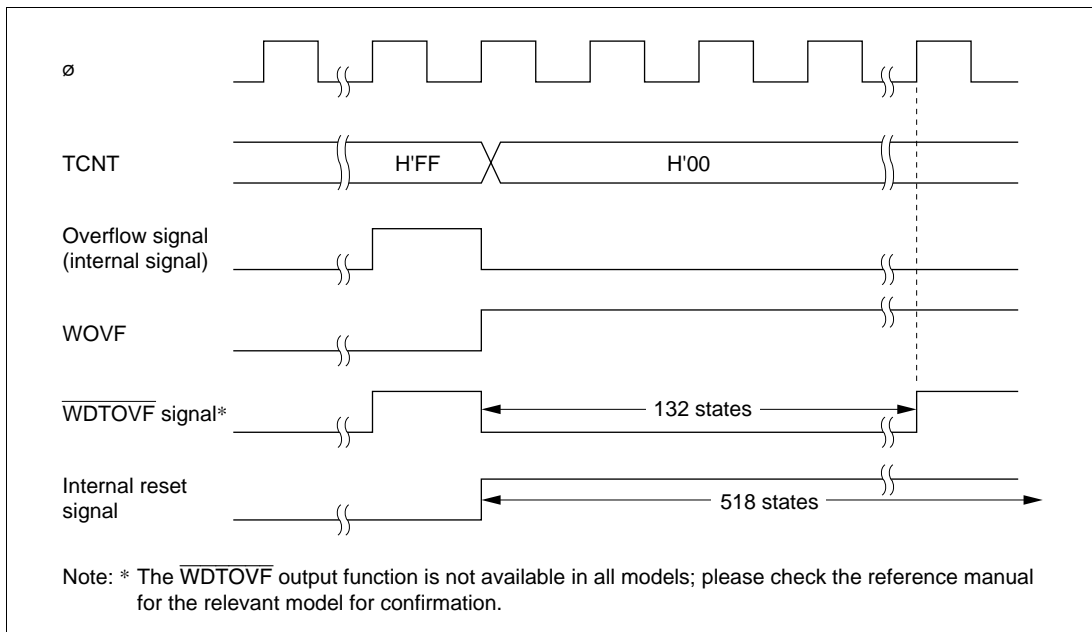


Figure 10-7 Timing of WOVF Setting

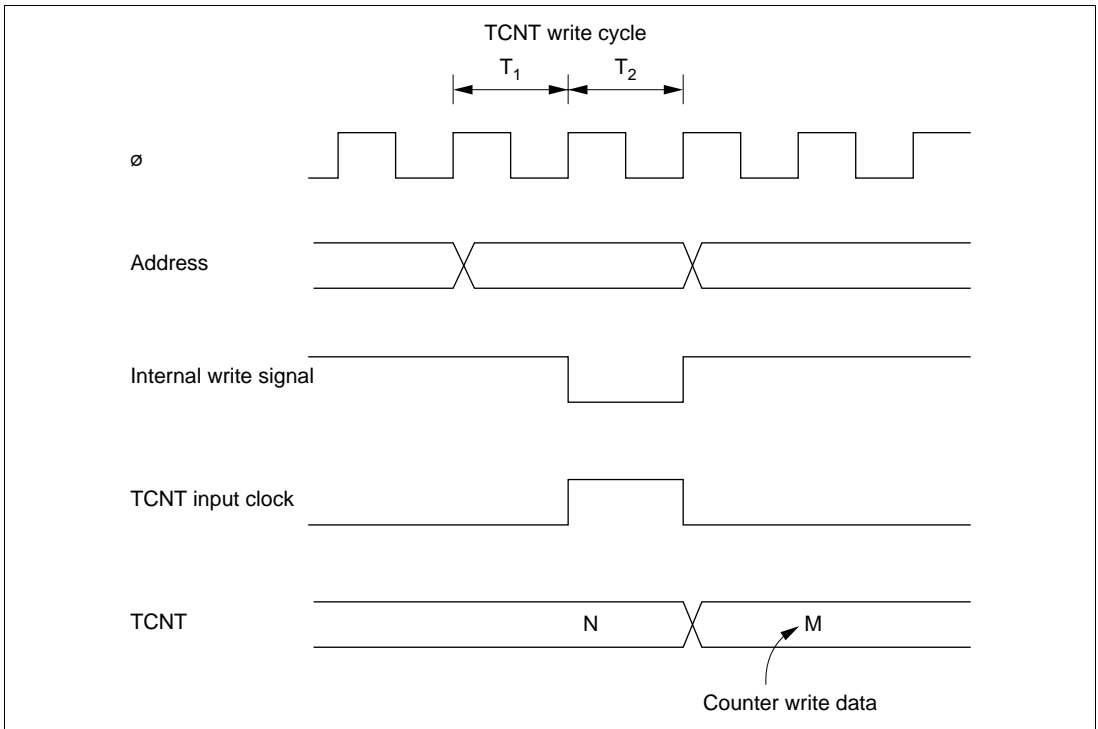
## 10.4 Interrupts

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR.

## 10.5 Usage Notes

### 10.5.1 Contention between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the  $T_2$  state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 10-8 shows this operation.



**Figure 10-8 Contention between TCNT Write and Increment**

### 10.5.2 Changing Value of CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors may occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before changing the value of bits CKS2 to CKS0.

### 10.5.3 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer, or vice versa, while the WDT is operating, errors may occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

### 10.5.4 System Reset by $\overline{\text{WDTOVF}}$ Signal\*

If the  $\overline{\text{WDTOVF}}$  output signal\* is input to the  $\overline{\text{RES}}$  pin of the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip, the chip will not be initialized correctly. Make sure that the  $\overline{\text{WDTOVF}}$  signal\* is not input logically to the  $\overline{\text{RES}}$  pin. To reset the entire system by means of the  $\overline{\text{WDTOVF}}$  signal\*, use the circuit shown in figure 10-9.

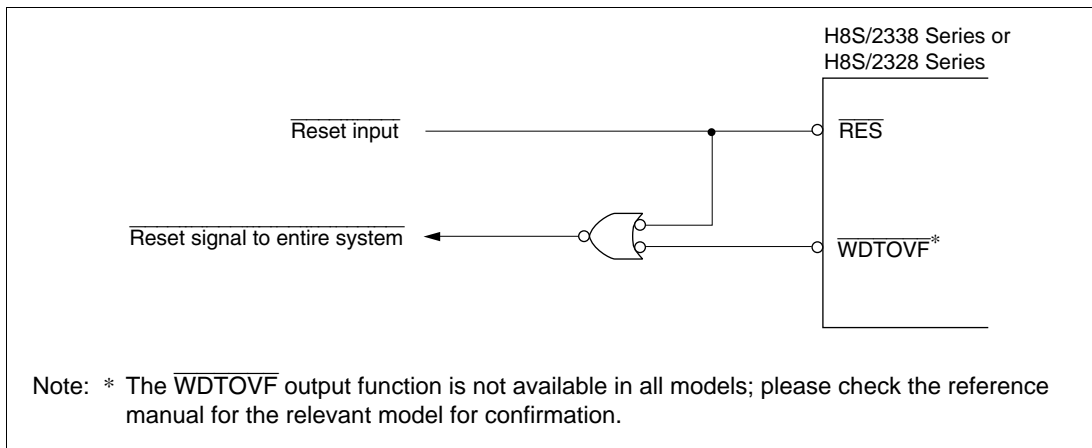


Figure 10-9 Circuit for System Reset by  $\overline{\text{WDTOVF}}$  Signal (Example)

### 10.5.5 Internal Reset in Watchdog Timer Mode

The H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip is not reset internally if TCNT overflows while the RSTE bit is cleared to 0 during watchdog timer operation, but TCNT and TCSR of the WDT are reset.

TCNT, TCSR, and RSTCR cannot be written to while the  $\overline{\text{WDTOVF}}$  signal\* is low. Also note that a read of the WOVF flag is not recognized during this period. To clear the WOVF flag, therefore, read TCSR after the  $\overline{\text{WDTOVF}}$  signal\* goes high, then write 0 to the WOVF flag.

Note: \* The  $\overline{\text{WDTOVF}}$  output function is not available in all models; please check the reference manual for the relevant model for confirmation.

# Section 11 Serial Communication Interface (SCI)

## 11.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are equipped with a serial communication interface (SCI) that can handle both asynchronous and synchronous serial communication. A function is also provided for serial communication between processors (multiprocessor communication function).

### 11.1.1 Features

SCI features are listed below.

- Choice of asynchronous or synchronous serial communication mode

#### Asynchronous mode

- Serial data communication executed using an asynchronous system in which synchronization is achieved character by character
- Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA)
- A multiprocessor communication function is provided that enables serial data communication with a number of processors
- Choice of 12 serial data transfer formats
  - Data length : 7 or 8 bits
  - Stop bit length : 1 or 2 bits
  - Parity : Even, odd, or none
  - Multiprocessor bit : 1 or 0
- Receive error detection : Parity, overrun, and framing errors
- Break detection : Break can be detected by reading the RxD pin level directly in case of a framing error

#### Synchronous mode

- Serial data communication synchronized with a clock
- Serial data communication can be carried out with other chips that have a synchronous communication function
- One serial data transfer format
  - Data length : 8 bits
- Receive error detection : Overrun errors detected

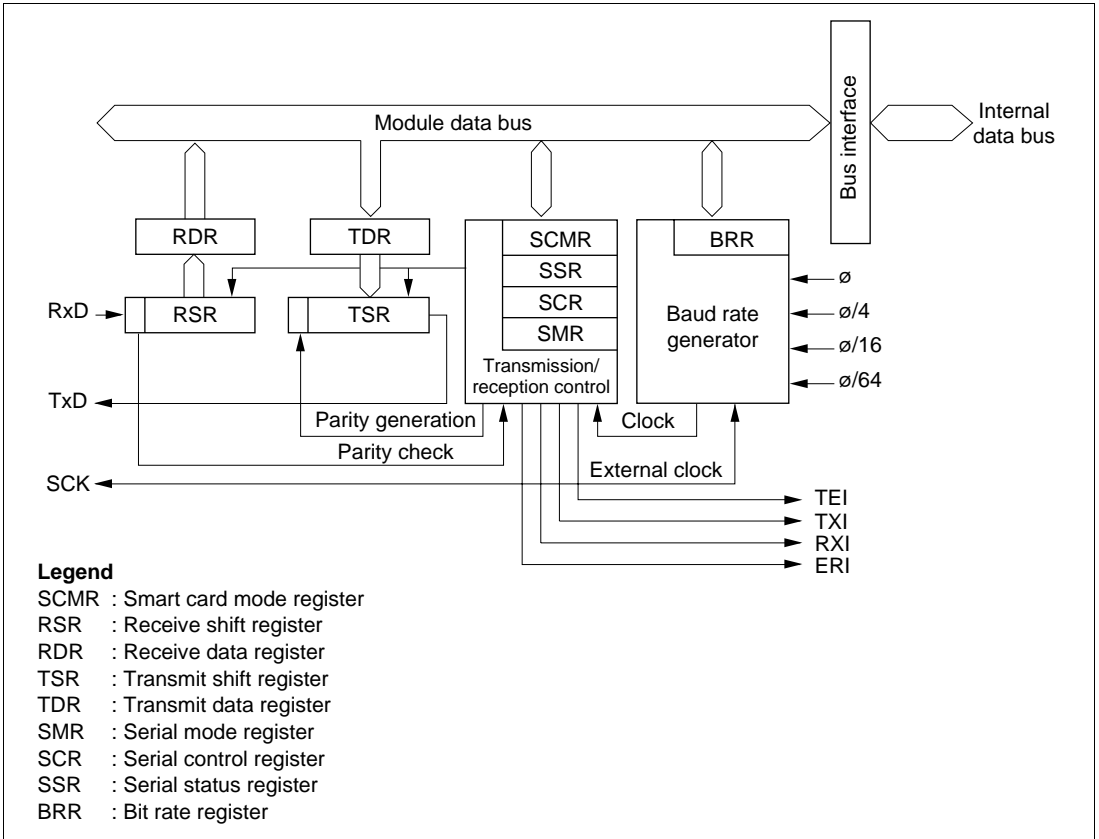
- Full-duplex communication capability
  - The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously
  - Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data
- Choice of LSB-first or MSB-first transfer
  - Can be selected regardless of the communication mode\*<sup>1</sup> (except in the case of asynchronous mode 7-bit data)
- Built-in baud rate generator allows any bit rate to be selected
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK pin
- Four interrupt sources
  - Four interrupt sources—transmit-data-empty, transmit-end, receive-data-full, and receive error—that can issue requests independently
  - The transmit-data-empty and receive-data-full interrupts can activate the DMA controller (DMAC)\*<sup>2</sup> or data transfer controller (DTC) to execute data transfer
- Module stop mode can be set
  - As the initial setting, SCI operation is halted. Register access is enabled by exiting module stop mode

Notes: 1. Descriptions in this section refer to LSB-first transfer.

2. Some models do not have an on-chip DMAC; please check the reference manual for the relevant model for confirmation.

## 11.1.2 Block Diagram

Figure 11-1 shows a block diagram of the SCI.



**Figure 11-1 Block Diagram of SCI**

### 11.1.3 Pin Configuration

Table 11-1 shows the serial pins for each SCI channel.

**Table 11-1 SCI Pins**

<b>Channel*</b>	<b>Pin Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
0	Serial clock pin 0	SCK0	I/O	SCI0 clock input/output
	Receive data pin 0	RxD0	Input	SCI0 receive data input
	Transmit data pin 0	TxD0	Output	SCI0 transmit data output
1	Serial clock pin 1	SCK1	I/O	SCI1 clock input/output
	Receive data pin 1	RxD1	Input	SCI1 receive data input
	Transmit data pin 1	TxD1	Output	SCI1 transmit data output
2	Serial clock pin 2	SCK2	I/O	SCI2 clock input/output
	Receive data pin 2	RxD2	Input	SCI2 receive data input
	Transmit data pin 2	TxD2	Output	SCI2 transmit data output

Note: \* The number of channels differs from model to model; see the reference manual for the relevant model for details.

## 11.1.4 Register Configuration

The SCI has the internal registers shown in table 11-2. These registers are used to specify asynchronous mode or synchronous mode, the data format, and the bit rate, and to control the transmitter/receiver.

**Table 11-2 SCI Registers**

Channel* <sup>1</sup>	Name	Abbreviation	R/W	Initial Value	Address* <sup>2</sup>
0	Serial mode register 0	SMR0	R/W	H'00	H'FF78
	Bit rate register 0	BRR0	R/W	H'FF	H'FF79
	Serial control register 0	SCR0	R/W	H'00	H'FF7A
	Transmit data register 0	TDR0	R/W	H'FF	H'FF7B
	Serial status register 0	SSR0	R/(W)* <sup>3</sup>	H'84	H'FF7C
	Receive data register 0	RDR0	R	H'00	H'FF7D
	Smart card mode register 0	SCMR0	R/W	H'F2	H'FF7E
1	Serial mode register 1	SMR1	R/W	H'00	H'FF80
	Bit rate register 1	BRR1	R/W	H'FF	H'FF81
	Serial control register 1	SCR1	R/W	H'00	H'FF82
	Transmit data register 1	TDR1	R/W	H'FF	H'FF83
	Serial status register 1	SSR1	R/(W)* <sup>3</sup>	H'84	H'FF84
	Receive data register 1	RDR1	R	H'00	H'FF85
	Smart card mode register 1	SCMR1	R/W	H'F2	H'FF86
2	Serial mode register 2	SMR2	R/W	H'00	H'FF88
	Bit rate register 2	BRR2	R/W	H'FF	H'FF89
	Serial control register 2	SCR2	R/W	H'00	H'FF8A
	Transmit data register 2	TDR2	R/W	H'FF	H'FF8B
	Serial status register 2	SSR2	R/(W)* <sup>3</sup>	H'84	H'FF8C
	Receive data register 2	RDR2	R	H'00	H'FF8D
	Smart card mode register 2	SCMR2	R/W	H'F2	H'FF8E
All	Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. The number of channels differs from model to model; see the reference manual for the relevant model for details.

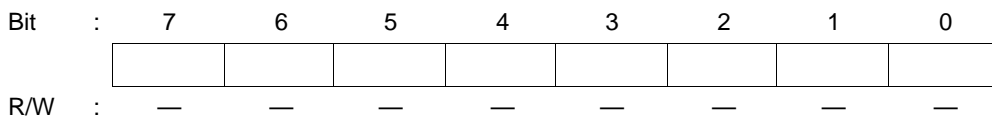
2. Lower 16 bits of the address.

3. Can only be written with 0 for flag clearing.



## 11.2 Register Descriptions

### 11.2.1 Receive Shift Register (RSR)



RSR is a register used to receive serial data.

The SCI sets serial data input from the RxD pin in RSR in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to RDR automatically.

RSR cannot be directly read or written to by the CPU.

### 11.2.2 Receive Data Register (RDR)



RDR is a register that stores received serial data.

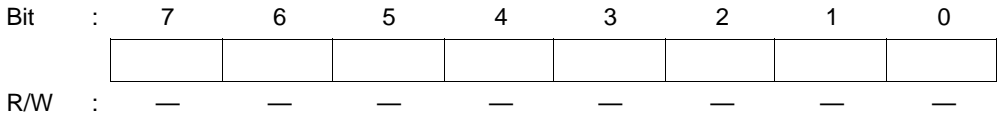
When the SCI has received one byte of serial data, it transfers the received serial data from RSR to RDR where it is stored, and completes the receive operation. After this, RSR is receive-enabled.

Since RSR and RDR function as a double buffer in this way, continuous receive operations can be performed.

RDR is a read-only register, and cannot be written to by the CPU.

RDR is initialized to H'00 by a reset, and in standby mode or module stop mode.

### 11.2.3 Transmit Shift Register (TSR)



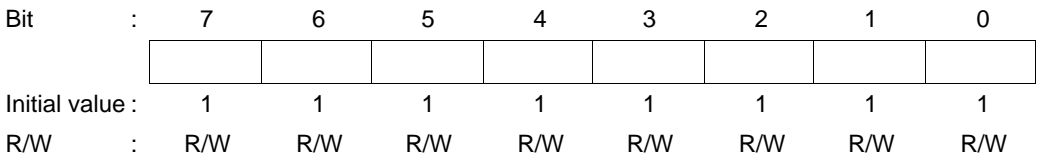
TSR is a register used to transmit serial data.

To perform serial data transmission, the SCI first transfers transmit data from TDR to TSR, then sends the data to the TxD pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from TDR to TSR, and transmission started, automatically. However, data transfer from TDR to TSR is not performed if the TDRE bit in SSR is set to 1.

TSR cannot be directly read or written to by the CPU.

### 11.2.4 Transmit Data Register (TDR)



TDR is an 8-bit register that stores data for serial transmission.

When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts serial transmission. Continuous serial transmission can be carried out by writing the next transmit data to TDR during serial transmission of the data in TSR.

TDR can be read or written to by the CPU at all times.

TDR is initialized to H'FF by a reset, and in standby mode or module stop mode.

## 11.2.5 Serial Mode Register (SMR)

Bit	:	7	6	5	4	3	2	1	0
		C/ $\bar{A}$	CHR	PE	O/ $\bar{E}$	STOP	MP	CKS1	CKS0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SMR is an 8-bit register used to set the SCI's serial transfer format and select the baud rate generator clock source.

SMR can be read or written to by the CPU at all times.

SMR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode and module stop mode it retains its previous state.

**Bit 7—Communication Mode (C/ $\bar{A}$ ):** Selects asynchronous mode or synchronous mode as the SCI operating mode.

Bit 7 C/ $\bar{A}$	Description
0	Asynchronous mode (Initial value)
1	Synchronous mode

**Bit 6—Character Length (CHR):** Selects 7 or 8 bits as the data length in asynchronous mode. In synchronous mode, a fixed data length of 8 bits is used regardless of the CHR setting.

Bit 6 CHR	Description
0	8-bit data (Initial value)
1	7-bit data*

Note: \* When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted, and it is not possible to choose between LSB-first or MSB-first transfer.

**Bit 5—Parity Enable (PE):** In asynchronous mode, selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. In synchronous mode and with a multiprocessor format, parity bit addition and checking is not performed, regardless of the PE bit setting.

**Bit 5**

PE	Description
0	Parity bit addition and checking disabled (Initial value)
1	Parity bit addition and checking enabled*

Note: \* When the PE bit is set to 1, the parity (even or odd) specified by the  $O/\bar{E}$  bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the  $O/\bar{E}$  bit.

**Bit 4—Parity Mode ( $O/\bar{E}$ ):** Selects either even or odd parity for use in parity addition and checking.

The  $O/\bar{E}$  bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking, in asynchronous mode. The  $O/\bar{E}$  bit setting is invalid in synchronous mode, and when parity addition and checking is disabled in asynchronous mode.

**Bit 4**

$O/\bar{E}$	Description
0	Even parity* <sup>1</sup> (Initial value)
1	Odd parity* <sup>2</sup>

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is even.

2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is odd.

**Bit 3—Stop Bit Length (STOP):** Selects 1 or 2 bits as the stop bit length in asynchronous mode. The STOP bits setting is only valid in asynchronous mode. If synchronous mode is set the STOP bit setting is invalid since stop bits are not added.

<b>Bit 3 STOP</b>	<b>Description</b>
0	1 stop bit: In transmission, a single 1-bit (stop bit) is added to the end of a transmit character before it is sent. (Initial value)
1	2 stop bits: In transmission, two 1-bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character.

**Bit 2—Multiprocessor Mode (MP):** Selects multiprocessor format. When multiprocessor format is selected, the PE bit and  $O/\bar{E}$  bit parity settings are invalid. The MP bit setting is only valid in asynchronous mode; it is invalid in synchronous mode.

For details of the multiprocessor communication function, see section 11.3.3, Multiprocessor Communication Function.

<b>Bit 2 MP</b>	<b>Description</b>
0	Multiprocessor function disabled (Initial value)
1	Multiprocessor format selected

**Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0):** These bits select the clock source for the baud rate generator. The clock source can be selected from  $\emptyset$ ,  $\emptyset/4$ ,  $\emptyset/16$ , and  $\emptyset/64$ , according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 11.2.8, Bit Rate Register.

Bit 1 CKS1	Bit 0 CKS0	Description	
0	0	$\emptyset$ clock	(Initial value)
	1	$\emptyset/4$ clock	
1	0	$\emptyset/16$ clock	
	1	$\emptyset/64$ clock	

### 11.2.6 Serial Control Register (SCR)

Bit	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value :	0	0	0	0	0	0	0	0
R/W :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCR is a register that performs enabling or disabling of SCI transfer operations, serial clock output in asynchronous mode, and interrupt requests, and selection of the serial clock source.

SCR can be read or written to by the CPU at all times.

SCR is initialized to H'00 by a reset and in hardware standby mode. In software standby mode and module stop mode it retains its previous state.

**Bit 7—Transmit Interrupt Enable (TIE):** Enables or disables transmit-data-empty interrupt (TXI) request generation when serial transmit data is transferred from TDR to TSR and the TDRE flag in SSR is set to 1.

Bit 7 TIE	Description	
0	Transmit-data-empty interrupt (TXI) requests disabled*	(Initial value)
1	Transmit-data-empty interrupt (TXI) requests enabled	

Note:\* TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or by clearing the TIE bit to 0.

**Bit 6—Receive Interrupt Enable (RIE):** Enables or disables receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request generation when serial receive data is transferred from RSR to RDR and the RDRF flag in SSR is set to 1.

Bit 6	
RIE	Description
0	Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request disabled* (Initial value)
1	Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request enabled

Note:\* RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or by clearing the RIE bit to 0.

**Bit 5—Transmit Enable (TE):** Enables or disables the start of serial transmission by the SCI.

Bit 5	
TE	Description
0	Transmission disabled* <sup>1</sup> (Initial value)
1	Transmission enabled* <sup>2</sup>

Notes: 1. The TDRE flag in SSR is fixed at 1.  
 2. In this state, serial transmission is started when transmit data is written to TDR and the TDRE flag in SSR is cleared to 0.  
 SMR setting must be performed to decide the transfer format before setting the TE bit to 1.

**Bit 4—Receive Enable (RE):** Enables or disables the start of serial reception by the SCI.

Bit 4	
RE	Description
0	Reception disabled* <sup>1</sup> (Initial value)
1	Reception enabled* <sup>2</sup>

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.  
 2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in synchronous mode.  
 SMR setting must be performed to decide the transfer format before setting the RE bit to 1.

**Bit 3—Multiprocessor Interrupt Enable (MPIE):** Enables or disables multiprocessor interrupts. The MPIE bit setting is only valid in asynchronous mode when the MP bit in SMR is set to 1.

The MPIE bit setting is invalid in synchronous mode or when the MP bit is cleared to 0.

**Bit 3**

<b>MPIE</b>	<b>Description</b>
0	Multiprocessor interrupts disabled (normal reception performed) (Initial value) [Clearing conditions] <ul style="list-style-type: none"><li>• When the MPIE bit is cleared to 0</li><li>• When data with MPB = 1 is received</li></ul>
1	Multiprocessor interrupts enabled* Receive interrupt (RXI) requests, receive-error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received.

Note: \* When receive data including MPB = 0 is received, receive data transfer from RSR to RDR, receive error detection, and setting of the RDRF, FER, and ORER flags in SSR, is not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR are set to 1) and FER and ORER flag setting is enabled.

**Bit 2—Transmit End Interrupt Enable (TEIE):** Enables or disables transmit-end interrupt (TEI) request generation when there is no valid transmit data in TDR in MSB data transmission.

**Bit 2**

<b>TEIE</b>	<b>Description</b>
0	Transmit end interrupt (TEI) request disabled* (Initial value)
1	Transmit end interrupt (TEI) request enabled*

Note: \* TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or by clearing the TEIE bit to 0.



**Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0):** These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin. The combination of the CKE1 and CKE0 bits determines whether the SCK pin functions as an I/O port, the serial clock output pin, or the serial clock input pin.

The setting of the CKE0 bit, however, is only valid for internal clock operation (CKE1 = 0) in asynchronous mode. The CKE0 bit setting is invalid in synchronous mode, and in the case of external clock operation (CKE1 = 1). Set CKE1 and CKE0 before determining the SCI operating mode with SMR.

For details of clock source selection, see table 11-9 in section 11-3, Operation.

Bit 1 CKE1	Bit 0 CKE0	Description	
0	0	Asynchronous mode	Internal clock/SCK pin functions as I/O port* <sup>1</sup>
		Synchronous mode	Internal clock/SCK pin functions as serial clock output
	1	Asynchronous mode	Internal clock/SCK pin functions as clock output* <sup>2</sup>
		Synchronous mode	Internal clock/SCK pin functions as serial clock output
1	0	Asynchronous mode	External clock/SCK pin functions as clock input* <sup>3</sup>
		Synchronous mode	External clock/SCK pin functions as serial clock input
	1	Asynchronous mode	External clock/SCK pin functions as clock input* <sup>3</sup>
		Synchronous mode	External clock/SCK pin functions as serial clock input

- Notes:
1. Initial value
  2. Outputs a clock of the same frequency as the bit rate.
  3. Inputs a clock with a frequency 16 times the bit rate.

## 11.2.7 Serial Status Register (SSR)

Bit	:	7	6	5	4	3	2	1	0
		TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value :		1	0	0	0	0	1	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: \* Only 0 can be written, to clear the flag.

SSR is an 8-bit register containing status flags that indicate the operating status of the SCI, and multiprocessor bits.

SSR can be read or written to by the CPU at all times. However, 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER. Also note that in order to clear these flags they must be read as 1 beforehand. The TEND flag and MPB flag are read-only flags and cannot be modified.

SSR is initialized to H'84 by a reset, and in standby mode or module stop mode.

**Bit 7—Transmit Data Register Empty (TDRE):** Indicates that data has been transferred from TDR to TSR and the next serial data can be written to TDR.

### Bit 7

TDRE	Description
0	[Clearing conditions] <ul style="list-style-type: none"> <li>When 0 is written to TDRE after reading TDRE = 1</li> <li>When the DMAC or DTC is activated by a TXI interrupt and writes data to TDR</li> </ul>
1	[Setting conditions] (Initial value) <ul style="list-style-type: none"> <li>When the TE bit in SCR is 0</li> <li>When data is transferred from TDR to TSR and data can be written to TDR</li> </ul>

**Bit 6—Receive Data Register Full (RDRF):** Indicates that the received data is stored in RDR.

### Bit 6

RDRF	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"> <li>When 0 is written to RDRF after reading RDRF = 1</li> <li>When the DMAC or DTC is activated by an RXI interrupt and reads data from RDR</li> </ul>
1	[Setting condition] <p>When serial reception ends normally and receive data is transferred from RSR to RDR</p>

Note: RDR and the RDRF flag are not affected and retain their previous values when an error is detected during reception or when the RE bit in SCR is cleared to 0.

If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error will occur and the receive data will be lost.

**Bit 5—Overrun Error (ORER):** Indicates that an overrun error occurred during reception, causing abnormal termination.

Bit 5 ORER	Description	
0	[Clearing condition] When 0 is written to ORER after reading ORER = 1	(Initial value)* <sup>1</sup>
1	[Setting condition] When the next serial reception is completed while RDRF = 1* <sup>2</sup>	

Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.  
2. The receive data prior to the overrun error is retained in RDR, and the data received subsequently is lost. Also, subsequent serial reception cannot be continued while the ORER flag is set to 1. In synchronous mode, serial transmission cannot be continued, either.

**Bit 4—Framing Error (FER):** Indicates that a framing error occurred during reception in asynchronous mode, causing abnormal termination.

Bit 4 FER	Description	
0	[Clearing condition] When 0 is written to FER after reading FER = 1	(Initial value)* <sup>1</sup>
1	[Setting condition] When the SCI checks the stop bit at the end of the receive data when reception ends, and the stop bit is 0 * <sup>2</sup>	

Notes: 1. The FER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.  
2. In 2-stop-bit mode, only the first stop bit is checked for a value of 0; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the FER flag is set to 1. In synchronous mode, serial transmission cannot be continued, either.

**Bit 3—Parity Error (PER):** Indicates that a parity error occurred during reception using parity addition in asynchronous mode, causing abnormal termination.

Bit 3 PER	Description
0	[Clearing condition] (Initial value)* <sup>1</sup> When 0 is written to PER after reading PER = 1
1	[Setting condition] When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the $O/\bar{E}$ bit in SMR* <sup>2</sup>

- Notes: 1. The PER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.
2. If a parity error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. In synchronous mode, serial transmission cannot be continued, either.

**Bit 2—Transmit End (TEND):** Indicates that there is no valid data in TDR when the last bit of the transmit character is sent, and transmission has been ended.

The TEND flag is read-only and cannot be modified.

Bit 2 TEND	Description
0	[Clearing conditions] <ul style="list-style-type: none"> <li>When 0 is written to TDRE after reading TDRE = 1</li> <li>When the DMAC or DTC is activated by a TXI interrupt and writes data to TDR</li> </ul>
1	[Setting conditions] (Initial value) <ul style="list-style-type: none"> <li>When the TE bit in SCR is 0</li> <li>When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character</li> </ul>

**Bit 1—Multiprocessor Bit (MPB):** When reception is performed using multiprocessor format in asynchronous mode, MPB stores the multiprocessor bit in the receive data.

MPB is a read-only bit, and cannot be modified.

Bit 1 MPB	Description
0	[Clearing condition] (Initial value)* When data with a 0 multiprocessor bit is received
1	[Setting condition] When data with a 1 multiprocessor bit is received

Note: \* Retains its previous state when the RE bit in SCR is cleared to 0 with multiprocessor format.

**Bit 0—Multiprocessor Bit Transfer (MPBT):** When transmission is performed using multiprocessor format in asynchronous mode, MPBT stores the multiprocessor bit to be added to the transmit data.

The MPBT bit setting is invalid when multiprocessor format is not used, when not transmitting, and in synchronous mode.

Bit 0 MPBT	Description
0	Data with a 0 multiprocessor bit is transmitted (Initial value)
1	Data with a 1 multiprocessor bit is transmitted

### 11.2.8 Bit Rate Register (BRR)

Bit	:	7	6	5	4	3	2	1	0
Initial value :		1	1	1	1	1	1	1	1
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BRR is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SMR.

BRR can be read or written to by the CPU at all times.

BRR is initialized to H'FF by a reset and in hardware standby mode. In software standby mode and module stop mode it retains its previous state.

As baud rate generator control is performed independently for each channel, different values can be set for each channel.

Table 11-3 shows sample BRR settings in asynchronous mode, and table 11-4 shows sample BRR settings in synchronous mode.

**Table 11-3 BRR Settings for Various Bit Rates (Asynchronous Mode)**

Bit Rate (bits/s)	$\phi = 2 \text{ MHz}$			$\phi = 2.097152 \text{ MHz}$			$\phi = 2.4576 \text{ MHz}$			$\phi = 3 \text{ MHz}$		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
	110	1	141	0.03	1	148	-0.04	1	174	-0.26	1	212
150	1	103	0.16	1	108	0.21	1	127	0.00	1	155	0.16
300	0	207	0.16	0	217	0.21	0	255	0.00	1	77	0.16
600	0	103	0.16	0	108	0.21	0	127	0.00	0	155	0.16
1200	0	51	0.16	0	54	-0.70	0	63	0.00	0	77	0.16
2400	0	25	0.16	0	26	1.14	0	31	0.00	0	38	0.16
4800	0	12	0.16	0	13	-2.48	0	15	0.00	0	19	-2.34
9600	0	6	—	0	6	-2.48	0	7	0.00	0	9	-2.34
19200	0	2	—	0	2	—	0	3	0.00	0	4	-2.34
31250	0	1	0.00	0	1	—	0	1	—	0	2	0.00
38400	0	1	—	0	1	—	0	1	0.00	—	—	—

Bit Rate (bits/s)	$\phi = 3.6864 \text{ MHz}$			$\phi = 4 \text{ MHz}$			$\phi = 4.9152 \text{ MHz}$			$\phi = 5 \text{ MHz}$		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
	110	2	64	0.70	2	70	0.03	2	86	0.31	2	88
150	1	191	0.00	1	207	0.16	1	255	0.00	2	64	0.16
300	1	95	0.00	1	103	0.16	1	127	0.00	1	129	0.16
600	0	191	0.00	0	207	0.16	0	255	0.00	1	64	0.16
1200	0	95	0.00	0	103	0.16	0	127	0.00	0	129	0.16
2400	0	47	0.00	0	51	0.16	0	63	0.00	0	64	0.16
4800	0	23	0.00	0	25	0.16	0	31	0.00	0	32	-1.36
9600	0	11	0.00	0	12	0.16	0	15	0.00	0	15	1.73
19200	0	5	0.00	0	6	—	0	7	0.00	0	7	1.73
31250	—	—	—	0	3	0.00	0	4	-1.70	0	4	0.00
38400	0	2	0.00	0	2	—	0	3	0.00	0	3	1.73

**Table 11-3 BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)**

Bit Rate (bits/s)	$\phi = 6 \text{ MHz}$			$\phi = 6.144 \text{ MHz}$			$\phi = 7.3728 \text{ MHz}$			$\phi = 8 \text{ MHz}$		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	106	-0.44	2	108	0.08	2	130	-0.07	2	141	0.03
150	2	77	0.16	2	79	0.00	2	95	0.00	2	103	0.16
300	1	155	0.16	1	159	0.00	1	191	0.00	1	207	0.16
600	1	77	0.16	1	79	0.00	1	95	0.00	1	103	0.16
1200	0	155	0.16	0	159	0.00	0	191	0.00	0	207	0.16
2400	0	77	0.16	0	79	0.00	0	95	0.00	0	103	0.16
4800	0	38	0.16	0	39	0.00	0	47	0.00	0	51	0.16
9600	0	19	-2.34	0	19	0.00	0	23	0.00	0	25	0.16
19200	0	9	-2.34	0	9	0.00	0	11	0.00	0	12	0.16
31250	0	5	0.00	0	5	2.40	—	—	—	0	7	0.00
38400	0	4	-2.34	0	4	0.00	0	5	0.00	—	—	—

Bit Rate (bits/s)	$\phi = 9.8304 \text{ MHz}$			$\phi = 10 \text{ MHz}$			$\phi = 12 \text{ MHz}$			$\phi = 12.288 \text{ MHz}$		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	174	-0.26	2	177	-0.25	2	212	0.03	2	217	0.08
150	2	127	0.00	2	129	0.16	2	155	0.16	2	159	0.00
300	1	255	0.00	2	64	0.16	2	77	0.16	2	79	0.00
600	1	127	0.00	1	129	0.16	1	155	0.16	1	159	0.00
1200	0	255	0.00	1	64	0.16	1	77	0.16	1	79	0.00
2400	0	127	0.00	0	129	0.16	0	155	0.16	0	159	0.00
4800	0	63	0.00	0	64	0.16	0	77	0.16	0	79	0.00
9600	0	31	0.00	0	32	-1.36	0	38	0.16	0	39	0.00
19200	0	15	0.00	0	15	1.73	0	19	-2.34	0	19	0.00
31250	0	9	-1.70	0	9	0.00	0	11	0.00	0	11	2.40
38400	0	7	0.00	0	7	1.73	0	9	-2.34	0	9	0.00

**Table 11-3 BRR Settings for Various Bit Rates (Asynchronous Mode) (cont)**

Bit Rate (bits/s)	$\phi = 14$ MHz			$\phi = 14.7456$ MHz			$\phi = 16$ MHz			$\phi = 17.2032$ MHz		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	248	-0.17	3	64	0.70	3	70	0.03	3	75	0.48
150	2	181	0.16	2	191	0.00	2	207	0.16	2	223	0.00
300	2	90	0.16	2	95	0.00	2	103	0.16	2	111	0.00
600	1	181	0.16	1	191	0.00	1	207	0.16	1	223	0.00
1200	1	90	0.16	1	95	0.00	1	103	0.16	1	111	0.00
2400	0	181	0.16	0	191	0.00	0	207	0.16	0	223	0.00
4800	0	90	0.16	0	95	0.00	0	103	0.16	0	111	0.00
9600	0	45	-0.93	0	47	0.00	0	51	0.16	0	55	0.00
19200	0	22	-0.93	0	23	0.00	0	25	0.16	0	27	0.00
31250	0	13	0.00	0	14	-1.70	0	15	0.00	0	16	1.20
38400	0	10	—	0	11	0.00	0	12	0.16	0	13	0.00

Bit Rate (bits/s)	$\phi = 18$ MHz			$\phi = 19.6608$ MHz			$\phi = 20$ MHz			$\phi = 25$ MHz*		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	79	-0.12	3	86	0.31	3	88	-0.25	3	110	-0.02
150	2	233	0.16	2	255	0.00	3	64	0.16	3	80	-0.47
300	2	116	0.16	2	127	0.00	2	129	0.16	2	162	0.15
600	1	233	0.16	1	255	0.00	2	64	0.16	2	80	-0.47
1200	1	116	0.16	1	127	0.00	1	129	0.16	1	162	0.15
2400	0	233	0.16	0	255	0.00	1	64	0.16	1	80	-0.47
4800	0	116	0.16	0	127	0.00	0	129	0.16	0	162	0.15
9600	0	58	-0.69	0	63	0.00	0	64	0.16	0	80	-0.47
19200	0	28	1.02	0	31	0.00	0	32	-1.36	0	40	-0.76
31250	0	17	0.00	0	19	-1.70	0	19	0.00	0	24	1.00
38400	0	14	-2.34	0	15	0.00	0	15	1.73	0	19	1.73

Note: \* In planning stage



**Table 11-4 BRR Settings for Various Bit Rates (Synchronous Mode)**

Bit Rate (bits/s)	$\phi = 2$ MHz		$\phi = 4$ MHz		$\phi = 8$ MHz		$\phi = 10$ MHz		$\phi = 16$ MHz		$\phi = 20$ MHz		$\phi = 25$ MHz*2	
	n	N	n	N	n	N	n	N	n	N	n	N	n	N
110	3	70												
250	2	124	2	249	3	124	—	—	3	249				
500	1	249	2	124	2	249	—	—	3	124	—	—		
1 k	1	124	1	249	2	124	—	—	2	249	—	—	3	97
2.5 k	0	199	1	99	1	199	1	249	2	99	2	124	2	155
5 k	0	99	0	199	1	99	1	124	1	199	1	249	2	77
10 k	0	49	0	99	0	199	0	249	1	99	1	124	1	155
25 k	0	19	0	39	0	79	0	99	0	159	0	199	0	249
50 k	0	9	0	19	0	39	0	49	0	79	0	99	0	124
100 k	0	4	0	9	0	19	0	24	0	39	0	49	0	62
250 k	0	1	0	3	0	7	0	9	0	15	0	19	0	24
500 k	0	0*1	0	1	0	3	0	4	0	7	0	9	—	—
1 M			0	0*1	0	1			0	3	0	4	—	—
2.5 M							0	0*1			0	1	—	—
5 M											0	0*1	—	—

Note: As far as possible, the setting should be made so that the error is no more than 1%.

Blank : Cannot be set.

— : Can be set, but there will be a degree of error.

\*1 : Continuous transfer is not possible.

\*2 : In planning stage

The BRR setting is found from the following formulas.

Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Synchronous mode:

$$N = \frac{\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bits/s)

N: BRR setting for baud rate generator ( $0 \leq N \leq 255$ )

$\phi$ : Operating frequency (MHz)

n: Baud rate generator input clock ( $n = 0$  to 3)

(See the table below for the relation between n and the clock.)

n	Clock	SMR Setting	
		CKS1	CKS0
0	$\phi$	0	0
1	$\phi/4$	0	1
2	$\phi/16$	1	0
3	$\phi/64$	1	1

The bit rate error in asynchronous mode is found from the following formula:

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 11-5 shows the maximum bit rate for each frequency in asynchronous mode. Tables 11-6 and 11-7 show the maximum bit rates with external clock input.

**Table 11-5 Maximum Bit Rate for Each Frequency (Asynchronous Mode)**

<b>ø (MHz)</b>	<b>Maximum Bit Rate (bits/s)</b>	<b>n</b>	<b>N</b>
2	62500	0	0
2.097152	65536	0	0
2.4576	76800	0	0
3	93750	0	0
3.6864	115200	0	0
4	125000	0	0
4.9152	153600	0	0
5	156250	0	0
6	187500	0	0
6.144	192000	0	0
7.3728	230400	0	0
8	250000	0	0
9.8304	307200	0	0
10	312500	0	0
12	375000	0	0
12.288	384000	0	0
14	437500	0	0
14.7456	460800	0	0
16	500000	0	0
17.2032	537600	0	0
18	562500	0	0
19.6608	614400	0	0
20	625000	0	0
25*	781250	0	0

Note: \* In planning stage

**Table 11-6 Maximum Bit Rate with External Clock Input (Asynchronous Mode)**

<b>ø (MHz)</b>	<b>External Input Clock (MHz)</b>	<b>Maximum Bit Rate (bits/s)</b>
2	0.5000	31250
2.097152	0.5243	32768
2.4576	0.6144	38400
3	0.7500	46875
3.6864	0.9216	57600
4	1.0000	62500
4.9152	1.2288	76800
5	1.2500	78125
6	1.5000	93750
6.144	1.5360	96000
7.3728	1.8432	115200
8	2.0000	125000
9.8304	2.4576	153600
10	2.5000	156250
12	3.0000	187500
12.288	3.0720	192000
14	3.5000	218750
14.7456	3.6864	230400
16	4.0000	250000
17.2032	4.3008	268800
18	4.5000	281250
19.6608	4.9152	307200
20	5.0000	312500
25*	6.2500	390625

Note: \* In planning stage

**Table 11-7 Maximum Bit Rate with External Clock Input (Synchronous Mode)**

$\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bits/s)
2	0.3333	333333.3
4	0.6667	666666.7
6	1.0000	1000000.0
8	1.3333	1333333.3
10	1.6667	1666666.7
12	2.0000	2000000.0
14	2.3333	2333333.3
16	2.6667	2666666.7
18	3.0000	3000000.0
20	3.3333	3333333.3
25*	4.1667	4166666.7

Note: \* In planning stage

### 11.2.9 Smart Card Mode Register (SCMR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	SDIR	SINV	—	SMIF
Initial value :		1	1	1	1	0	0	1	0
R/W	:	—	—	—	—	R/W	R/W	—	R/W

SCMR selects LSB-first or MSB-first transfer by means of bit SDIR. Except in the case of asynchronous mode 7-bit data, LSB-first or MSB-first transfer can be selected regardless of the serial communication mode. The descriptions in this chapter refer to LSB-first transfer.

For details of the other bits in SCMR, see 12.2.1, Smart Card Mode Register (SCMR).

SCMR is initialized to H'F2 by a reset and in hardware standby mode. In software standby mode and module stop mode it retains its previous state.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 1.

**Bit 3—Smart Card Data Transfer Direction (SDIR):** Selects the serial/parallel conversion format.

This bit is valid when 8-bit data is used as the transmit/receive format.

**Bit 3**

<b>SDIR</b>	<b>Description</b>	
0	TDR contents are transmitted LSB-first Receive data is stored in RDR LSB-first	(Initial value)
1	TDR contents are transmitted MSB-first Receive data is stored in RDR MSB-first	

**Bit 2—Smart Card Data Invert (SINV):** Specifies inversion of the data logic level. The SINV bit does not affect the logic level of the parity bit(s); parity bit inversion requires inversion of the O/ $\bar{E}$  bit in SMR.

**Bit 2**

<b>SINV</b>	<b>Description</b>	
0	TDR contents are transmitted without modification Receive data is stored in RDR without modification	(Initial value)
1	TDR contents are inverted before being transmitted Receive data is stored in RDR in inverted form	

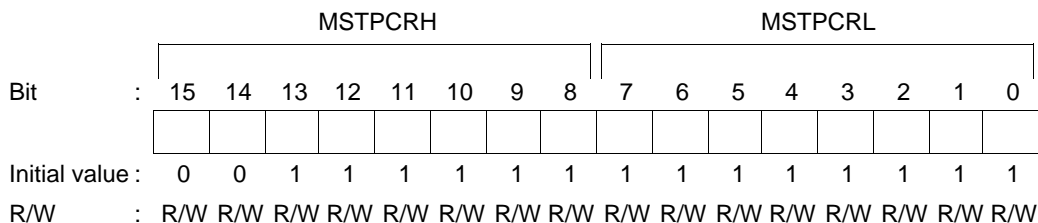
**Bit 1—Reserved:** Read-only bit, always read as 1.

**Bit 0—Smart Card Interface Mode Select (SMIF):** When the smart card interface operates as a normal SCI, 0 should be written to this bit.

**Bit 0**

<b>SMIF</b>	<b>Description</b>	
0	Operates as normal SCI (smart card interface function disabled)	(Initial value)
1	Smart card interface function enabled	

## 11.2.10 Module Stop Control Register (MSTPCR)



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the corresponding bit of bits MSTP7 to MSTP5 is set to 1, SCI operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Module Stop (MSTP7):** Specifies the SCI channel 2 module stop mode.

Bit 7 MSTP7	Description
0	SCI channel 2 module stop mode cleared
1	SCI channel 2 module stop mode set <span style="float: right;">(Initial value)</span>

**Bit 6—Module Stop (MSTP6):** Specifies the SCI channel 1 module stop mode.

Bit 6 MSTP6	Description
0	SCI channel 1 module stop mode cleared
1	SCI channel 1 module stop mode set <span style="float: right;">(Initial value)</span>

**Bit 5—Module Stop (MSTP5):** Specifies the SCI channel 0 module stop mode.

Bit 5 MSTP5	Description
0	SCI channel 0 module stop mode cleared
1	SCI channel 0 module stop mode set <span style="float: right;">(Initial value)</span>

## 11.3 Operation

### 11.3.1 Overview

The SCI can carry out serial communication in two modes: asynchronous mode in which synchronization is achieved character by character, and synchronous mode in which synchronization is achieved with clock pulses.

Selection of asynchronous or synchronous mode and the transmission format is made using SMR as shown in table 11-8. The SCI clock is determined by a combination of the  $C/\overline{A}$  bit in SMR and the CKE1 and CKE0 bits in SCR, as shown in table 11-9.

#### Asynchronous Mode

- Data length: Choice of 7 or 8 bits
- Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing, parity, and overrun errors, and breaks, during reception
- Choice of internal or external clock as SCI clock source
  - When internal clock is selected:

The SCI operates on the baud rate generator clock and a clock with the same frequency as the bit rate can be output
  - When external clock is selected:

A clock with a frequency of 16 times the bit rate must be input (the built-in baud rate generator is not used)

#### Synchronous Mode

- Transfer format: Fixed 8-bit data
- Detection of overrun errors during reception
- Choice of internal or external clock as SCI clock source
  - When internal clock is selected:

The SCI operates on the baud rate generator clock and a serial clock is output off-chip
  - When external clock is selected:

The built-in baud rate generator is not used, and the SCI operates on the input serial clock



**Table 11-8 SMR Settings and Serial Transfer Format Selection**

SMR Settings						SCI Transfer Format			
Bit 7	Bit 6	Bit 2	Bit 5	Bit 3	Mode	Data Length	Multi-processor Bit	Parity Bit	Stop Bit Length
C/A	CHR	MP	PE	STOP					
0	0	0	0	0	Asynchronous mode	8-bit data	No	No	1 bit
				1					2 bits
				0	Yes				1 bit
				1					2 bits
	1	0	0	0	mode (multi-processor format)	7-bit data	No	No	1 bit
				1					2 bits
		1	0	0		Yes			1 bit
				1					2 bits
0	1	—	0	Asynchronous mode (multi-processor format)	8-bit data	Yes	No	1 bit	
								1	2 bits
	1	—	0		0			7-bit data	1 bit
									1
1	—	—	—	Synchronous mode	8-bit data	No	None		

**Table 11-9 SMR and SCR Settings and SCI Clock Source Selection**

SMR		SCR Setting		SCI Transmit/Receive Clock		
Bit 7	Bit 1	Bit 0	Mode	Clock Source	SCK Pin Function	
C/A	CKE1	CKE0				
0	0	0	Asynchronous mode	Internal	SCI does not use SCK pin	
		1				Outputs clock with same frequency as bit rate
	1	0	External		Inputs clock with frequency of 16 times the bit rate	
		1				
1	0	0	Synchronous mode	Internal	Outputs serial clock	
		1				
	1	0		External	Inputs serial clock	
		1				

### 11.3.2 Operation in Asynchronous Mode

In asynchronous mode, characters are sent or received, each preceded by a start bit indicating the start of communication and one or two stop bits indicating the end of communication. Serial communication is thus carried out with synchronization established on a character-by-character basis.

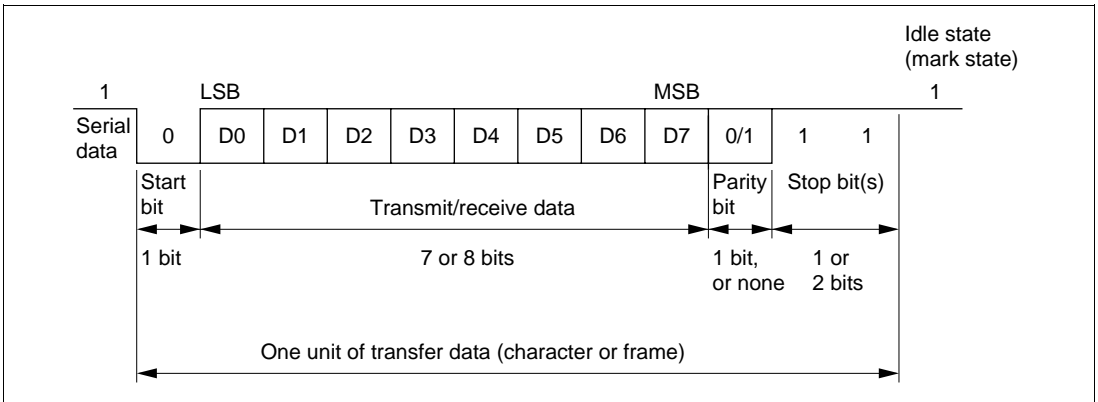
Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 11-2 shows the general format for asynchronous serial communication.

In asynchronous serial communication, the communication line is usually held in the mark state (high level). The SCI monitors the communication line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication.

One serial communication character consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, the SCI performs synchronization at the falling edge of the start bit in reception. The SCI samples the data on the 8th pulse of a clock with a frequency of 16 times the length of one bit, so that the transfer data is latched at the center of each bit.



**Figure 11-2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)**

**Data Transfer Format:** Table 11-10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting.

**Table 11-10 Serial Transfer Formats (Asynchronous Mode)**

SMR Settings				Serial Transfer Format and Frame Length														
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12			
0	0	0	0	S	8-bit data								STOP					
0	0	0	1	S	8-bit data								STOP	STOP				
0	1	0	0	S	8-bit data								P	STOP				
0	1	0	1	S	8-bit data								P	STOP	STOP			
1	0	0	0	S	7-bit data							STOP						
1	0	0	1	S	7-bit data							STOP	STOP					
1	1	0	0	S	7-bit data							P	STOP					
1	1	0	1	S	7-bit data							P	STOP	STOP				
0	—	1	0	S	8-bit data								MPB	STOP				
0	—	1	1	S	8-bit data								MPB	STOP	STOP			
1	—	1	0	S	7-bit data							MPB	STOP					
1	—	1	1	S	7-bit data							MPB	STOP	STOP				

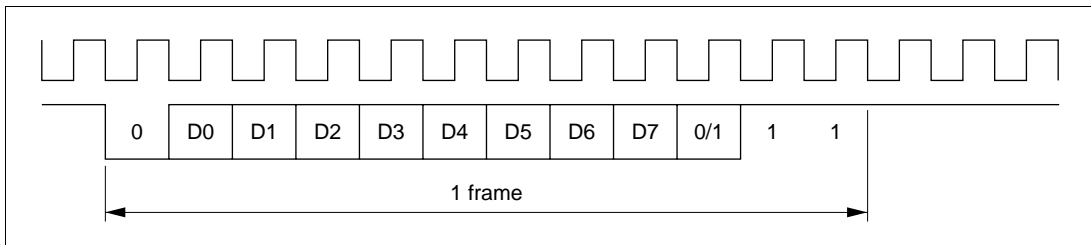
**Legend**

- S : Start bit
- STOP : Stop bit
- P : Parity bit
- MPB : Multiprocessor bit

**Clock:** Either an internal clock generated by the built-in baud rate generator or an external clock input at the SCK pin can be selected as the SCI's serial clock, according to the setting of the  $\overline{C/\overline{A}}$  bit in SMR and the CKE1 and CKE0 bits in SCR. For details of SCI clock source selection, see table 11-9.

When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is at the center of each transmit data bit, as shown in figure 11-3.



**Figure 11-3 Relation between Output Clock and Transfer Data Phase  
(Asynchronous Mode)**

## Data Transfer Operations

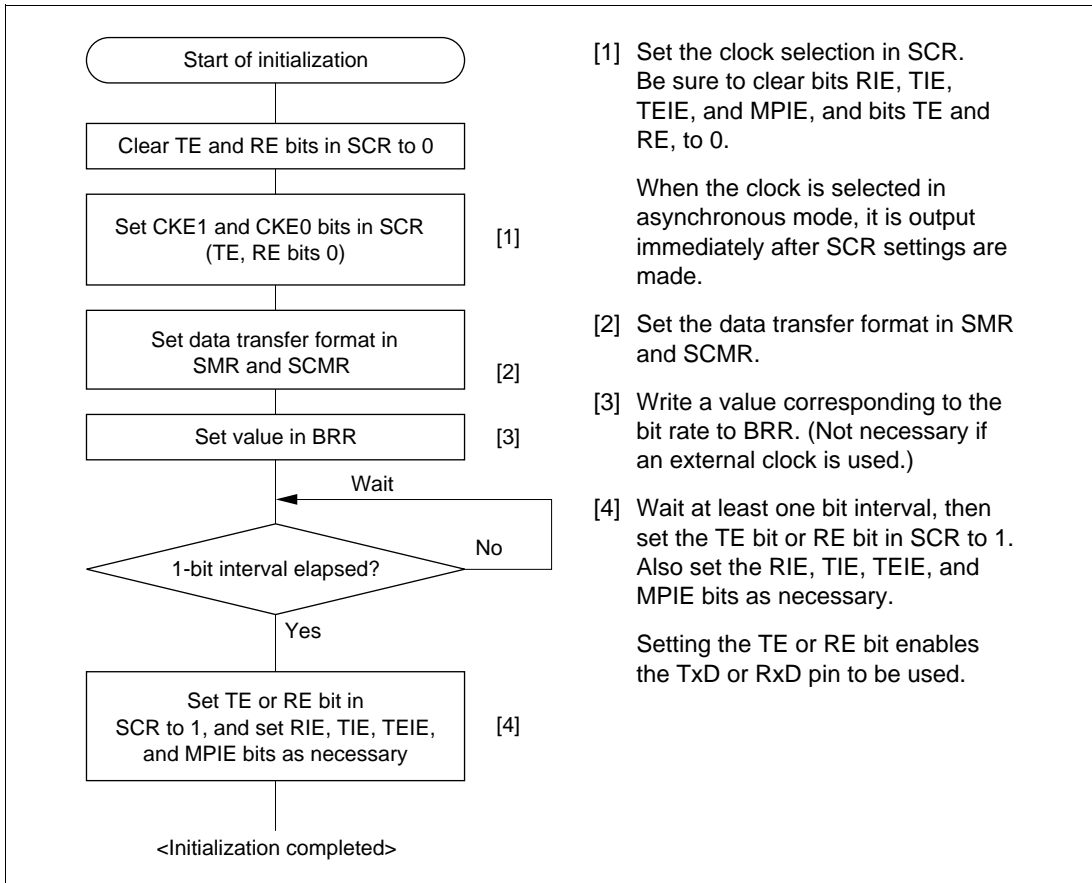
- SCI initialization (asynchronous mode)

Before transmitting or receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 11-4 shows a sample SCI initialization flowchart.

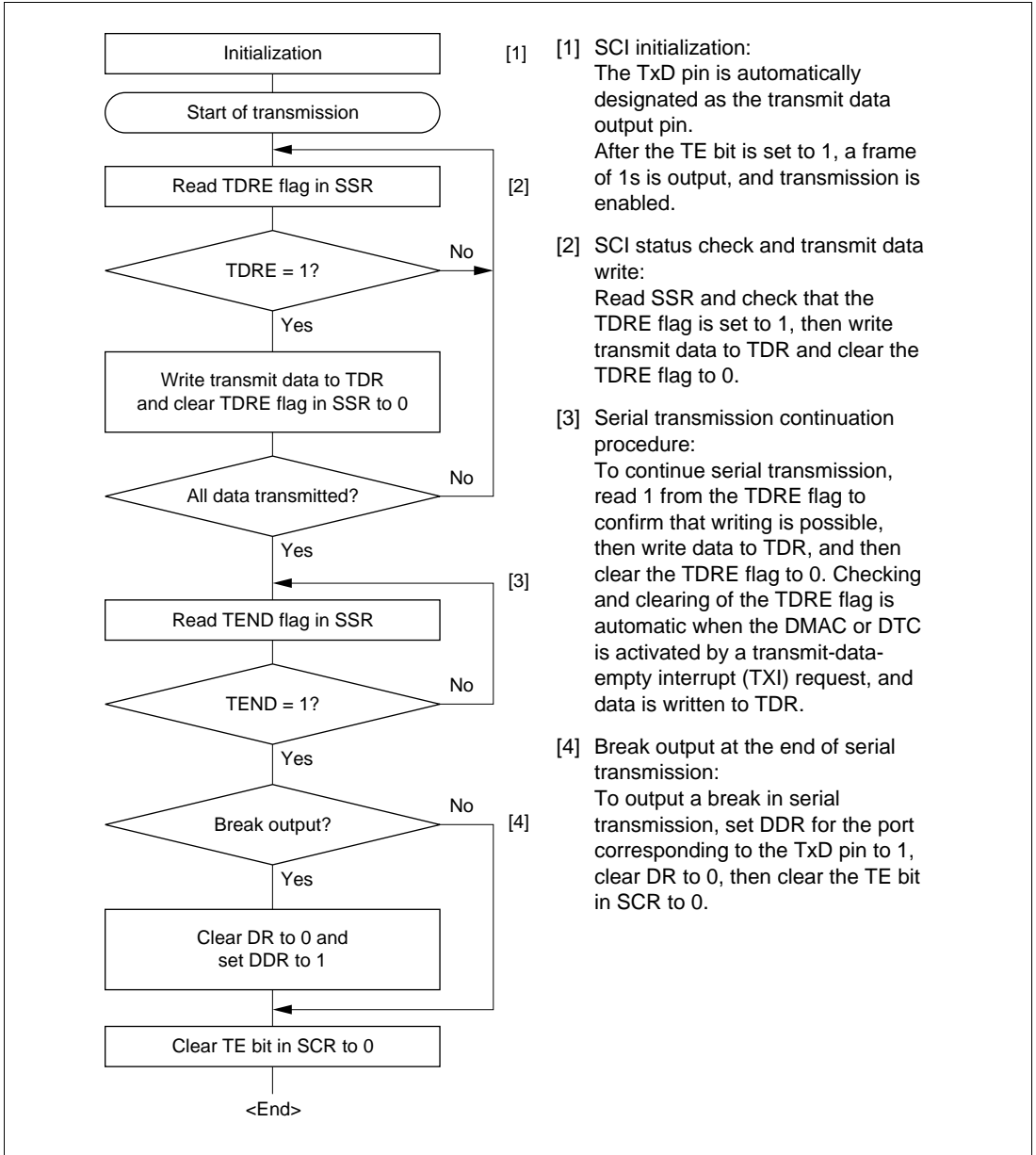


**Figure 11-4 Sample SCI Initialization Flowchart**

- Serial data transmission (asynchronous mode)

Figure 11-5 shows a sample flowchart for serial transmission.

The following procedure should be used for serial data transmission.



**Figure 11-5 Sample Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Parity bit or multiprocessor bit:

One parity bit (even or odd parity), or one multiprocessor bit is output.

A format in which neither a parity bit nor a multiprocessor bit is output can also be selected.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

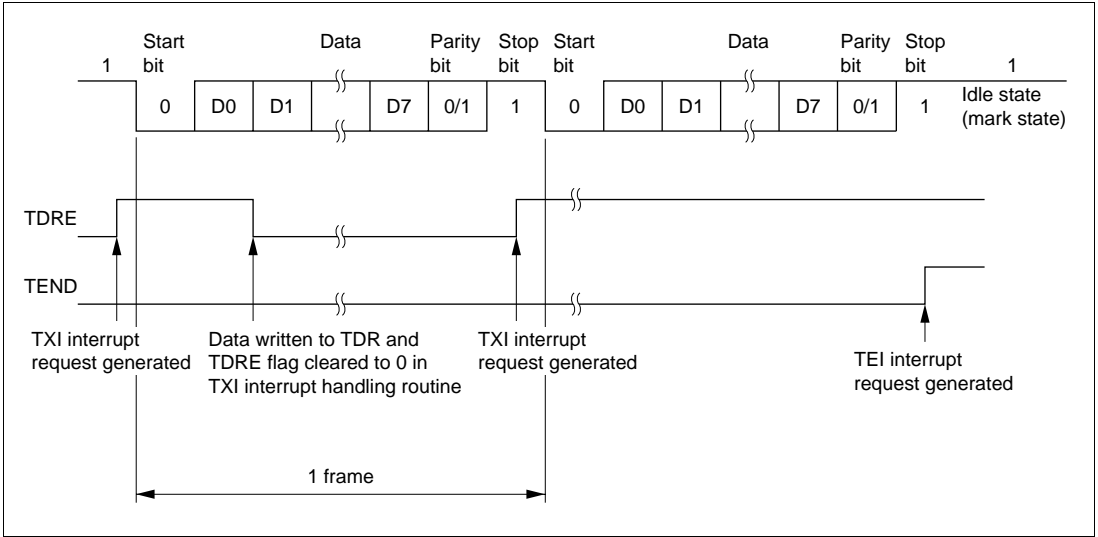
1 is output continuously until the start bit that starts the next transmission is sent.

[3] The SCI checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output continuously. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 11-6 shows an example of the operation for transmission in asynchronous mode.



**Figure 11-6 Example of Transmit Operation in Asynchronous Mode  
(Example with 8-Bit Data, Parity, One Stop Bit)**



- Serial data reception (asynchronous mode)

Figure 11-7 shows a sample flowchart for serial reception.

The following procedure should be used for serial data reception.

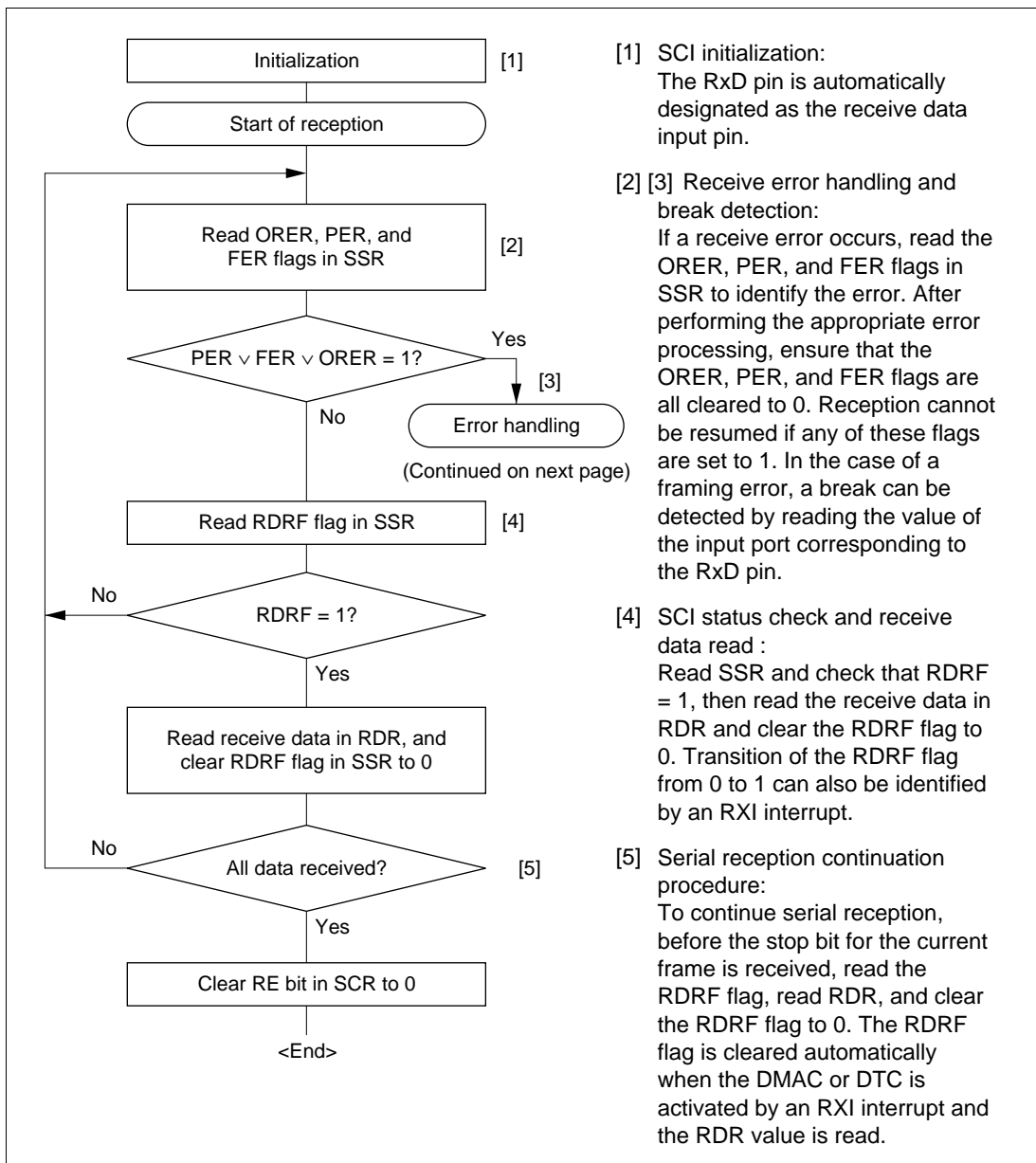


Figure 11-7 Sample Serial Reception Flowchart

[3]

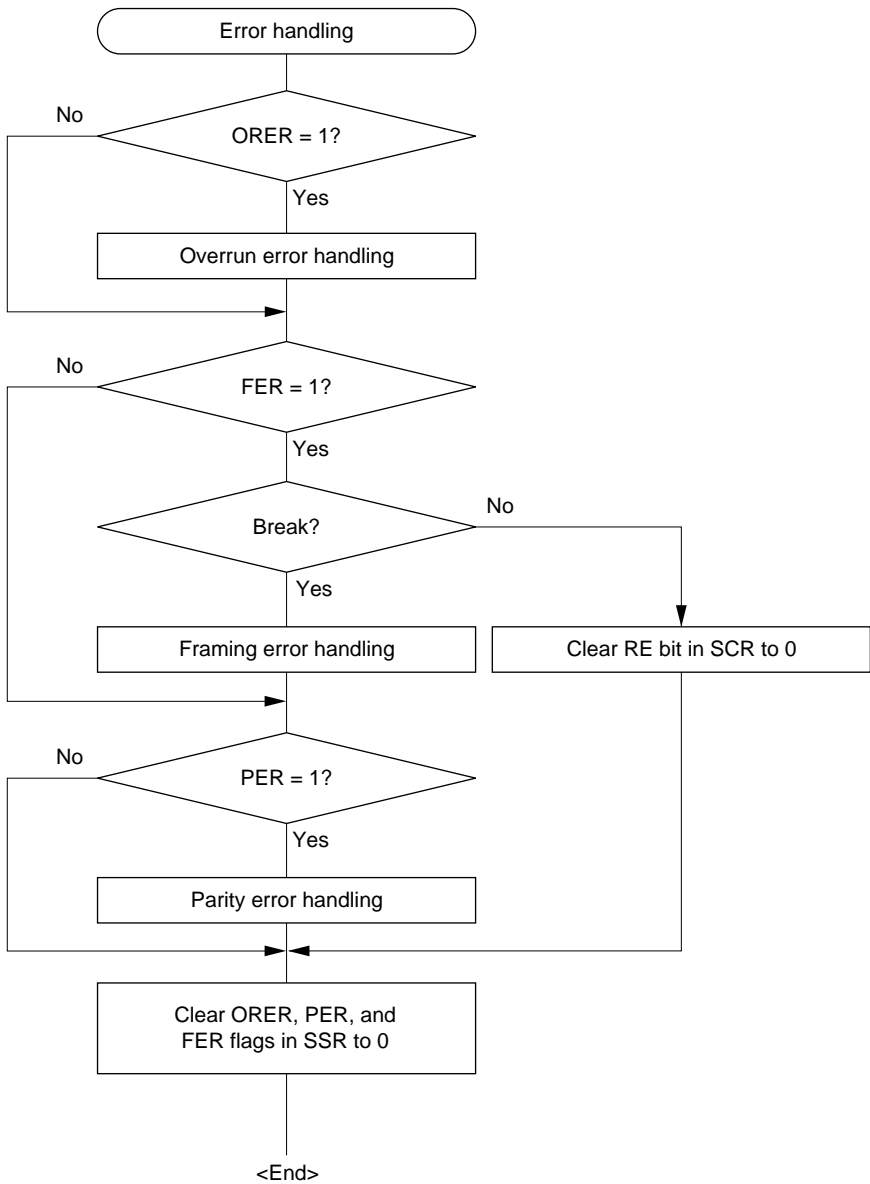


Figure 11-7 Sample Serial Reception Flowchart (cont)

In serial reception, the SCI operates as described below.

[1] The SCI monitors the communication line, and if a 0 stop bit is detected, performs internal synchronization and starts reception.

[2] The received data is stored in RSR in LSB-to-MSB order.

[3] The parity bit and stop bit are received.

After receiving these bits, the SCI carries out the following checks.

[a] Parity check:

The SCI checks whether the number of 1 bits in the receive data agrees with the parity (even or odd) set in the  $O/\bar{E}$  bit in SMR.

[b] Stop bit check:

The SCI checks whether the stop bit is 1.

If there are two stop bits, only the first is checked.

[c] Status check:

The SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from RSR to RDR.

If all the above checks are passed, the RDRF flag is set to 1, and the receive data is stored in RDR.

If a receive error\* is detected in the error check, the operation is as shown in table 11-11.

Note: \* Subsequent receive operations cannot be performed when a receive error has occurred.

Also note that the RDRF flag is not set to 1 in reception, and so the error flags must be cleared to 0.

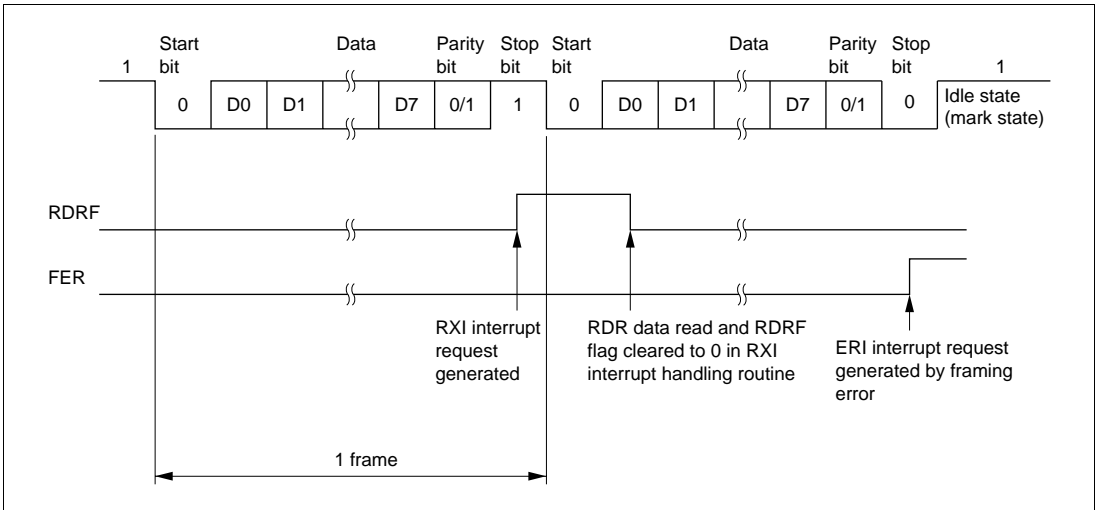
[4] If the RIE bit in SCR is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR is set to 1 when the ORER, PER, or FER flag changes to 1, a receive-error interrupt (ERI) request is generated.

**Table 11-11 Receive Error Conditions**

Receive Error	Abbreviation	Condition	Data Transfer
Overrun error	ORER	When the next data reception is completed while the RDRF flag in SSR is set to 1	Receive data is not transferred from RSR to RDR
Framing error	FER	When the stop bit is 0	Receive data is transferred from RSR to RDR
Parity error	PER	When the received data differs from the parity (even or odd) set in SMR	Receive data is transferred from RSR to RDR

Figure 11-8 shows an example of the operation for reception in asynchronous mode.



**Figure 11-8 Example of SCI Receive Operation  
(Example with 8-Bit Data, Parity, One Stop Bit)**

### 11.3.3 Multiprocessor Communication Function

The multiprocessor communication function performs serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data, in asynchronous mode. Use of this function enables data transfer to be performed among a number of processors sharing a single serial communication line.

When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code.

The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added.

The receiving station skips the data until data with a 1 multiprocessor bit is sent.

When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip the data until data with a 1 multiprocessor bit is again received. In this way, data communication is carried out among a number of processors.

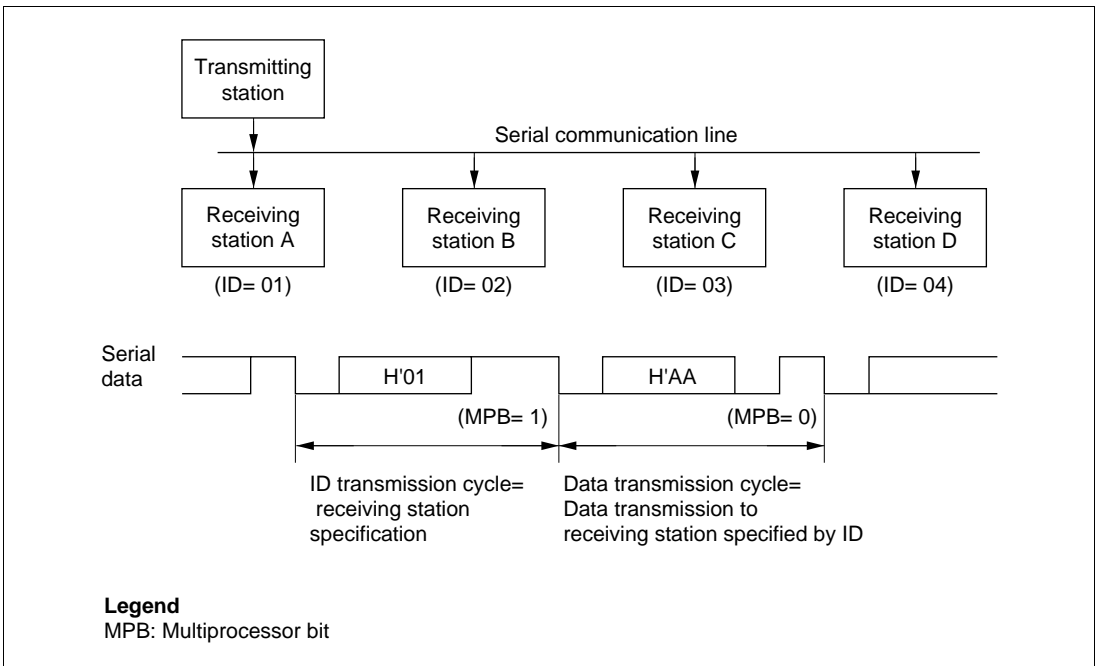
Figure 11-9 shows an example of inter-processor communication using the multiprocessor format.

**Data Transfer Formats:** There are four data transfer formats.

When the multiprocessor format is specified, the parity bit specification is invalid.

For details, see table 11-10.

**Clock:** See the section on asynchronous mode.



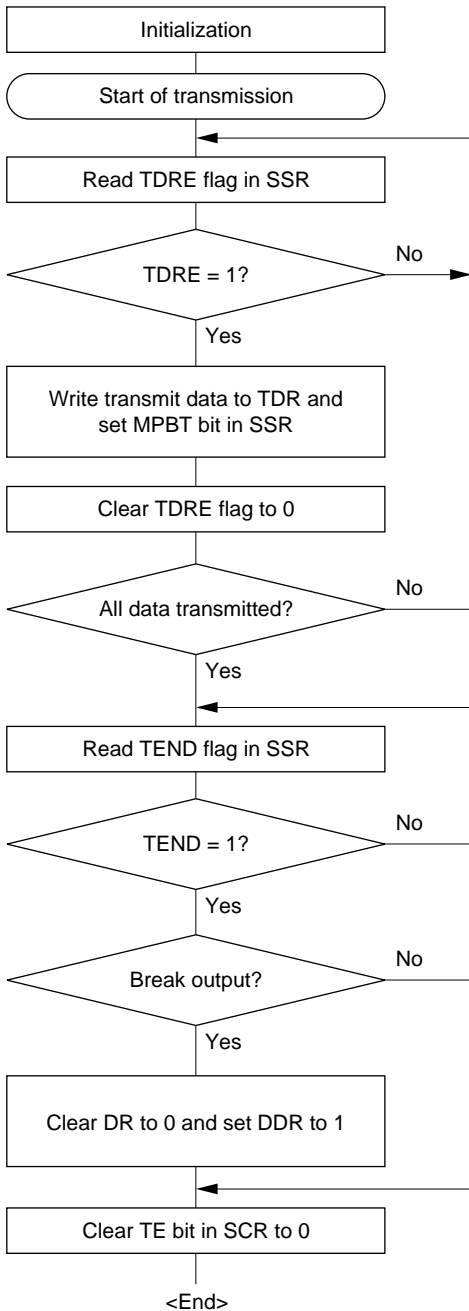
**Figure 11-9 Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)**

### Data Transfer Operations

- Multiprocessor serial data transmission

Figure 11-10 shows a sample flowchart for multiprocessor serial data transmission.

The following procedure should be used for multiprocessor serial data transmission.



- [1] [1] SCI initialization:  
The TxD pin is automatically designated as the transmit data output pin.  
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.
- [2] [2] SCI status check and transmit data write:  
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. Set the MPBT bit in SSR to 0 or 1.  
Finally, clear the TDRE flag to 0.
- [3] [3] Serial transmission continuation procedure:  
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. Checking and clearing of the TDRE flag is automatic when the DMAC or DTC is activated by a transmit-data-empty interrupt (TXI) request, and data is written to TDR.
- [4] [4] Break output at the end of serial transmission:  
To output a break in serial transmission, set the port DDR to 1, clear DR to 0, then clear the TE bit in SCR to 0.

**Figure 11-10 Sample Multiprocessor Serial Transmission Flowchart**

In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated.

The serial transmit data is sent from the TxD pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Multiprocessor bit

One multiprocessor bit (MPBT value) is output.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

1 is output continuously until the start bit that starts the next transmission is sent.

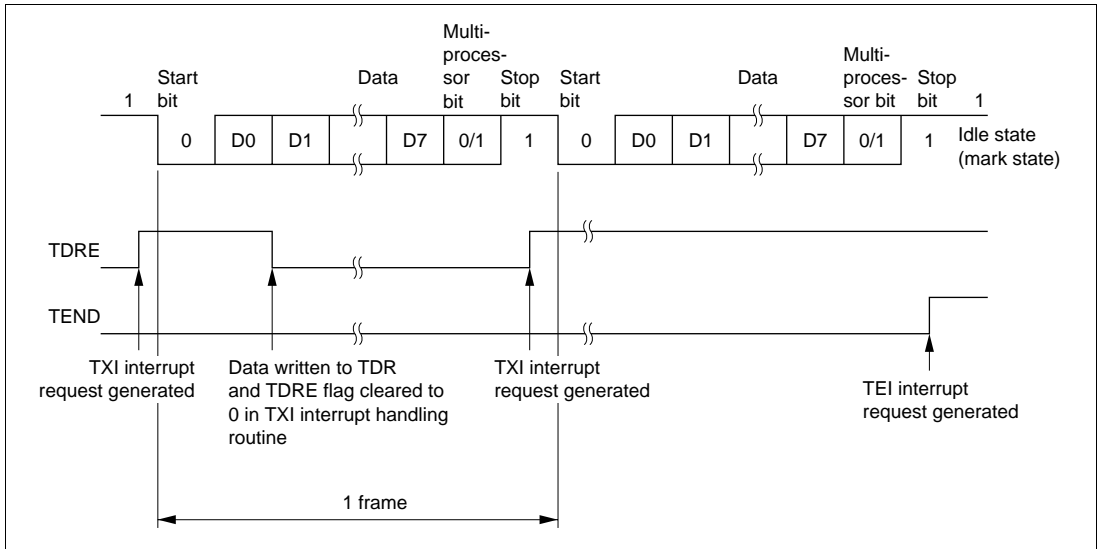
[3] The SCI checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output continuously. If the TEIE bit in SCR is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.



Figure 11-11 shows an example of SCI operation for transmission using the multiprocessor format.

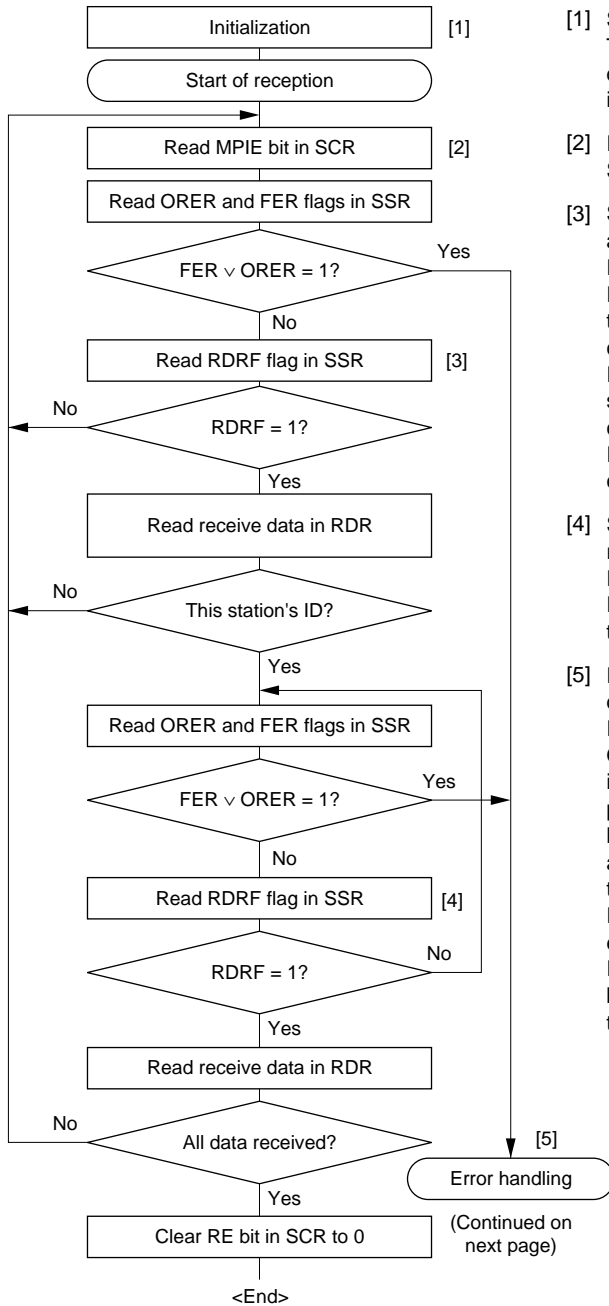


**Figure 11-11 Example of SCI Transmit Operation  
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

- Multiprocessor serial data reception

Figure 11-12 shows a sample flowchart for multiprocessor serial reception.

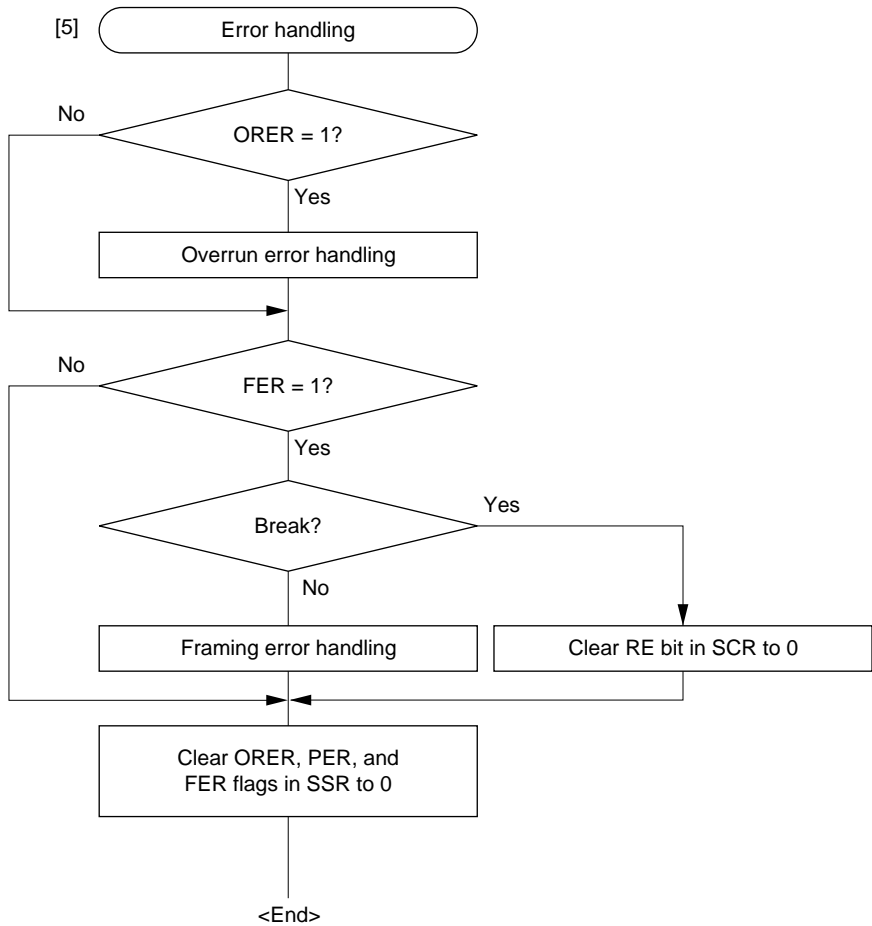
The following procedure should be used for multiprocessor serial data reception.



- [1] SCI initialization:  
The RxD pin is automatically designated as the receive data input pin.
- [2] ID reception cycle:  
Set the MPIE bit in SCR to 1.
- [3] SCI status check, ID reception and comparison:  
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID. If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0. If the data is this station's ID, clear the RDRF flag to 0.
- [4] SCI status check and data reception:  
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.
- [5] Receive error handling and break detection:  
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error handling, ensure that the ORER and FER flags are both cleared to 0. Reception cannot be resumed if either of these flags is set to 1. In the case of a framing error, a break can be detected by reading the RxD pin value.

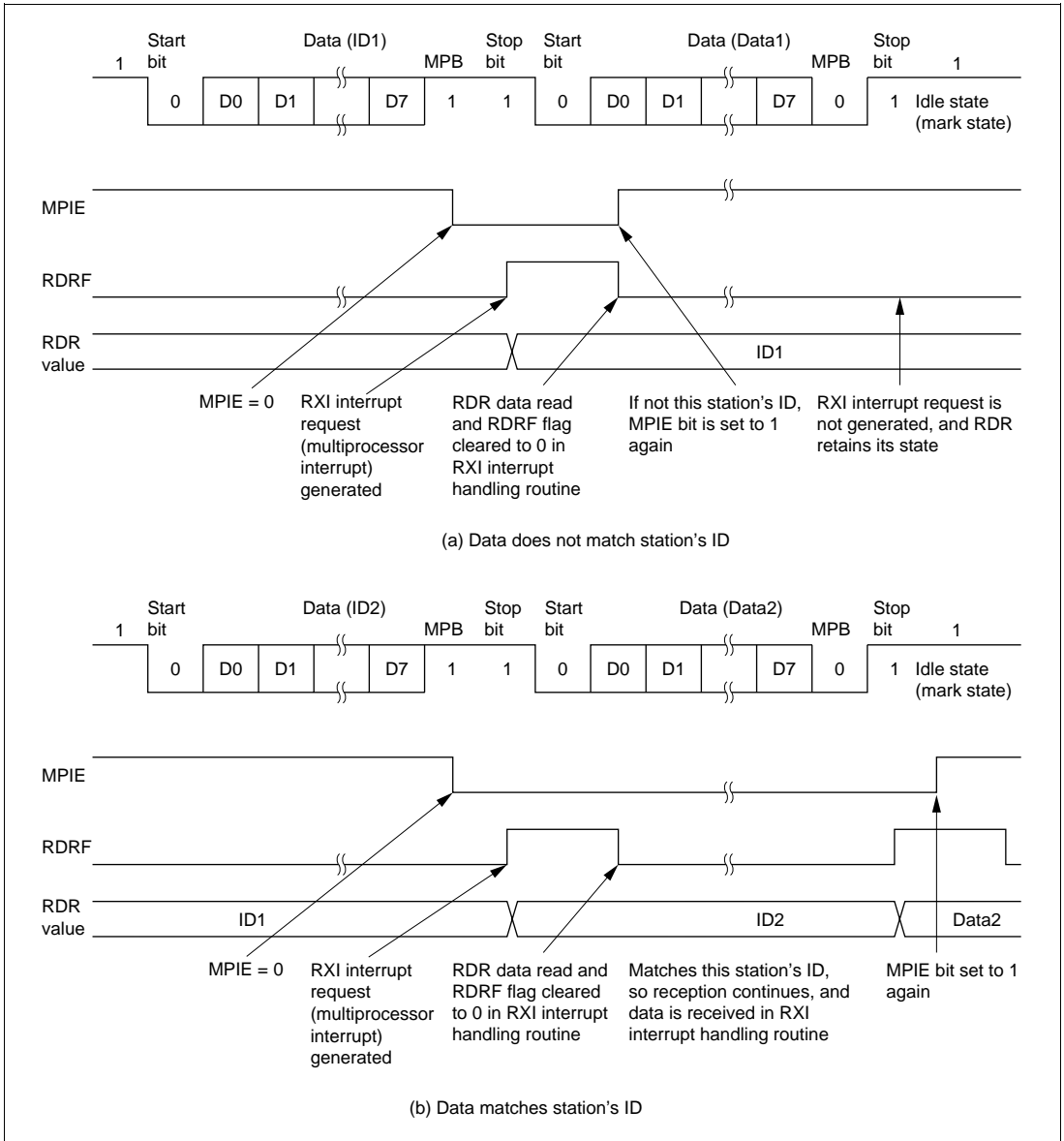
(Continued on next page)

Figure 11-12 Sample Multiprocessor Serial Reception Flowchart



**Figure 11-12 Sample Multiprocessor Serial Reception Flowchart (cont)**

Figure 11-13 shows an example of SCI operation for multiprocessor format reception.



**Figure 11-13 Example of SCI Receive Operation  
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**



**Clock:** Either an internal clock generated by the built-in baud rate generator or an external serial clock input at the SCK pin can be selected, according to the setting of the  $\overline{C/A}$  bit in SMR and the CKE1 and CKE0 bits in SCR. For details of SCI clock source selection, see table 11-9.

When the SCI is operated on an internal clock, the serial clock is output from the SCK pin.

Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. When only receive operations are performed, however, the serial clock is output until an overrun error occurs or the RE bit is cleared to 0. To perform receive operations in units of one character, an external clock should be selected as the clock source.

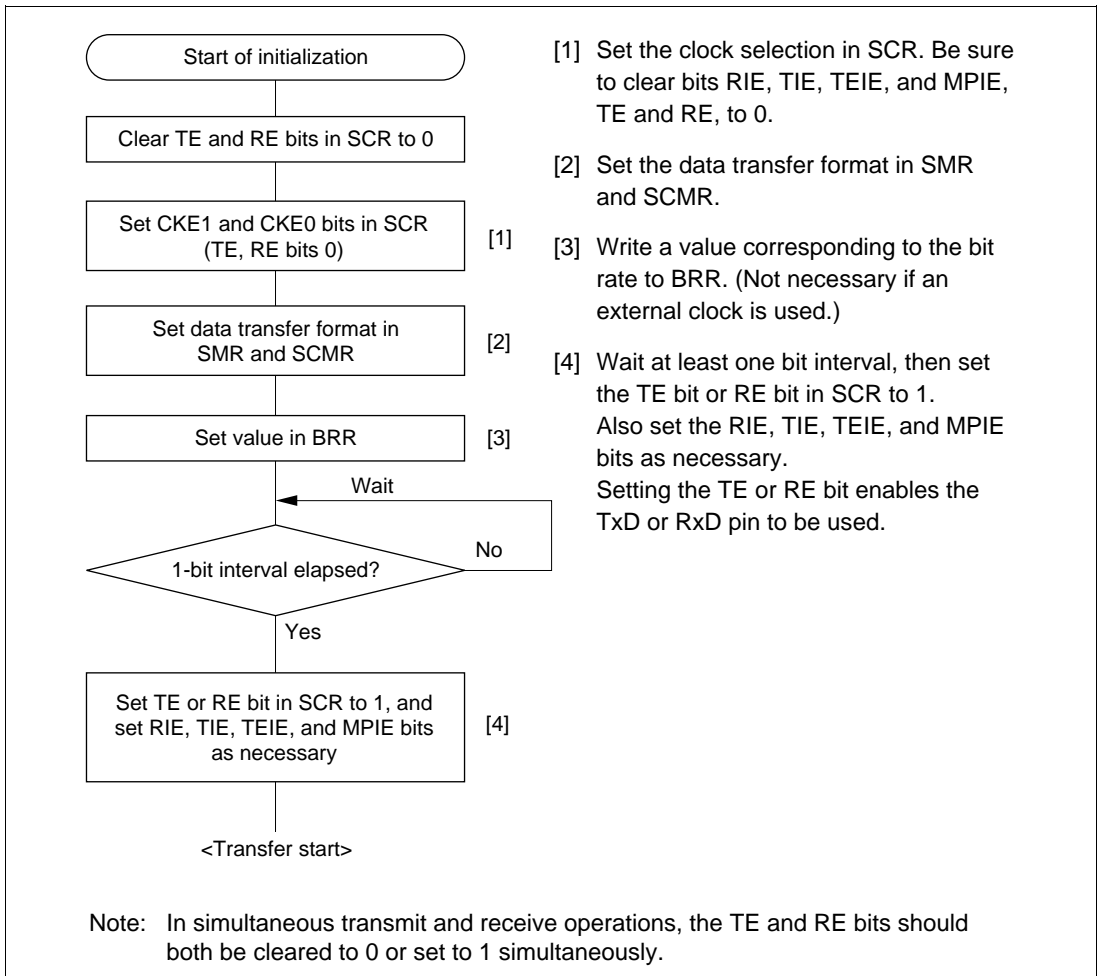
## Data Transfer Operations

- SCI initialization (synchronous mode)

Before transmitting or receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

Figure 11-15 shows a sample SCI initialization flowchart.

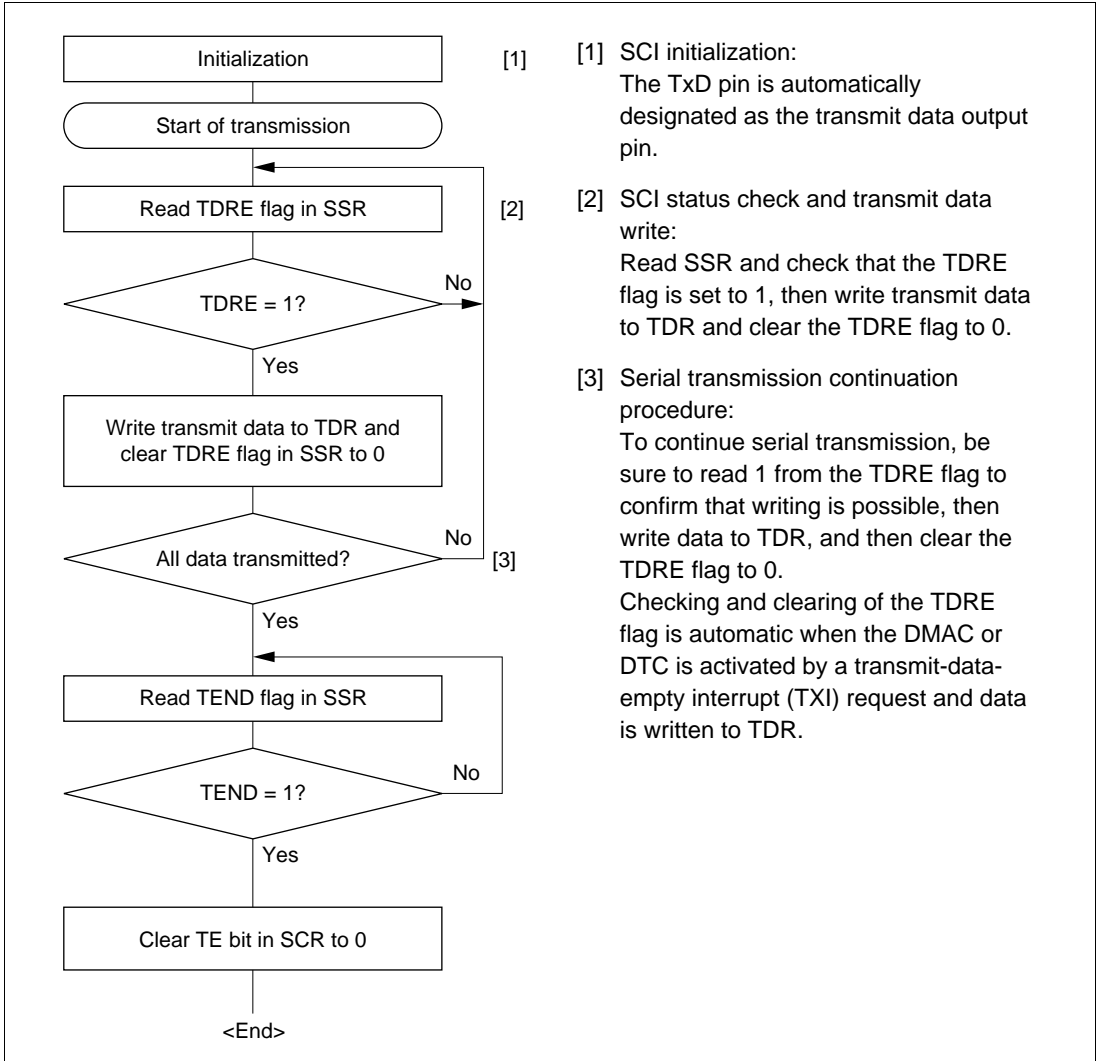


**Figure 11-15 Sample SCI Initialization Flowchart**

- Serial data transmission (synchronous mode)

Figure 11-16 shows a sample flowchart for serial transmission.

The following procedure should be used for serial data transmission.



**Figure 11-16 Sample Serial Transmission Flowchart**



In serial transmission, the SCI operates as described below.

[1] The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

[2] After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated.

When clock output mode has been set, the SCI outputs 8 serial clock pulses. When use of an external clock has been specified, data is output synchronized with the input clock.

The serial transmit data is sent from the TxD pin starting with the LSB (bit 0) and ending with the MSB (bit 7).

[3] The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).

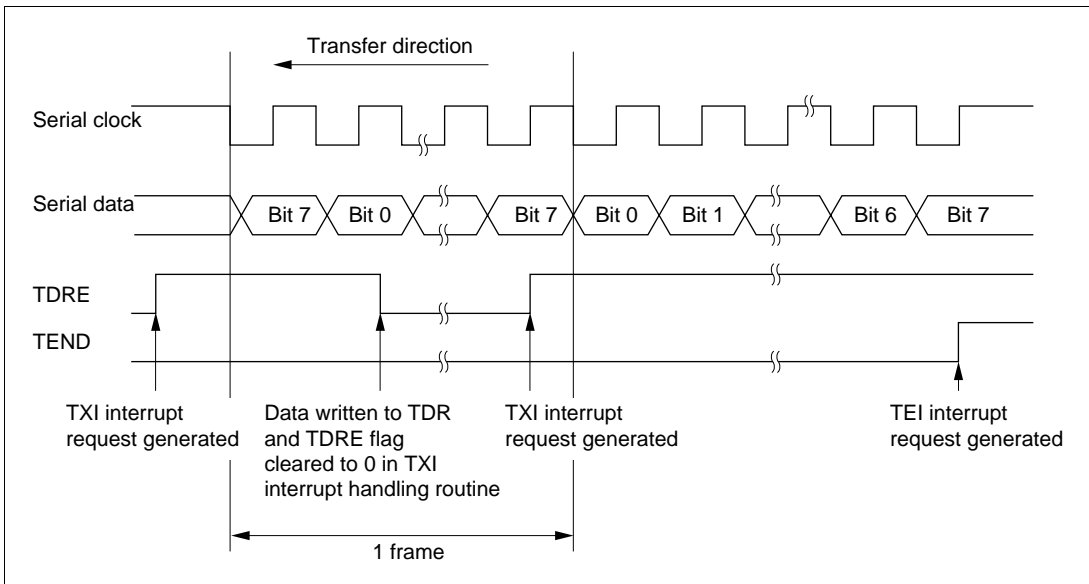
If the TDRE flag is cleared to 0, data is transferred from TDR to TSR, and serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, the MSB (bit 7) is sent, and the TxD pin maintains its state.

If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

[4] After completion of serial transmission, the SCK pin is fixed.

Figure 11-17 shows an example of SCI operation in transmission.



**Figure 11-17 Example of SCI Transmit Operation**

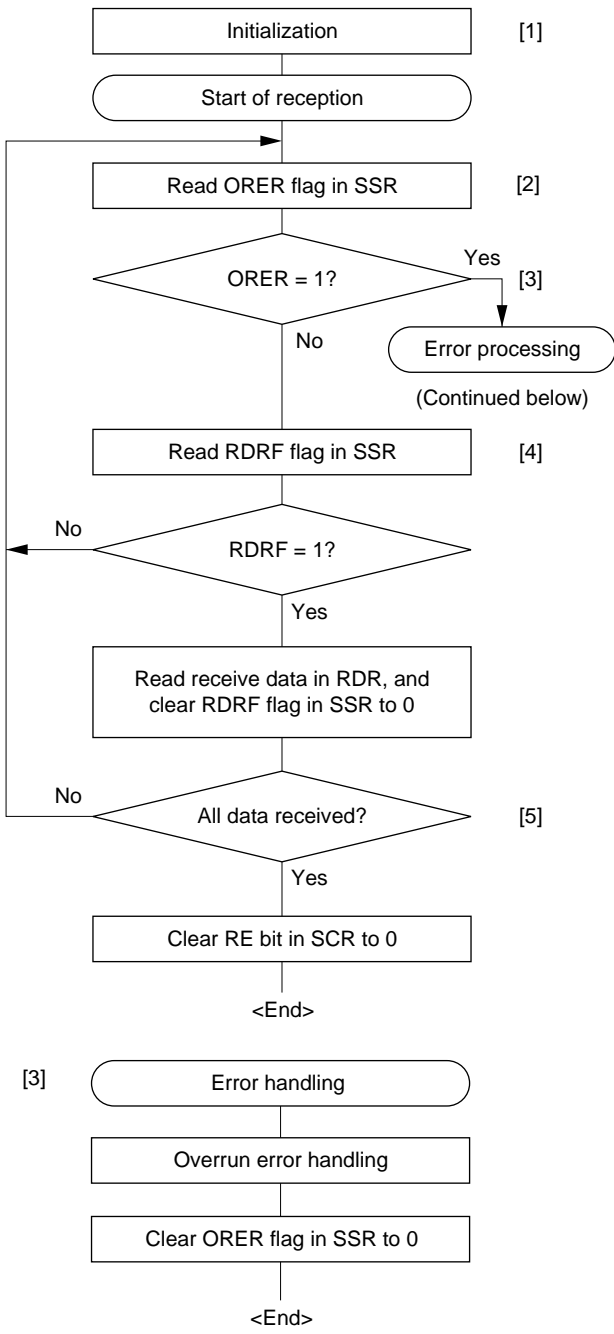
- Serial data reception (synchronous mode)

Figure 11-18 shows a sample flowchart for serial reception.

The following procedure should be used for serial data reception.

When changing the operating mode from asynchronous to synchronous, be sure to check that the ORER, PER, and FER flags are all cleared to 0.

The RDRF flag will not be set if the FER or PER flag is set to 1, and neither transmit nor receive operations will be possible.



- [1] SCI initialization:  
The RxD pin is automatically designated as the receive data input pin.
- [2] [3] Receive error handling:  
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error handling, clear the ORER flag to 0. Transfer cannot be resumed if the ORER flag is set to 1.
- [4] SCI status check and receive data read:  
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
- [5] Serial reception continuation procedure:  
To continue serial reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. The RDRF flag is cleared automatically when the DMAC or DTC is activated by a receive-data-full interrupt (RXI) request and the RDR value is read.

**Figure 11-18 Sample Serial Reception Flowchart**

In serial reception, the SCI operates as described below.

[1] The SCI performs internal initialization in synchronization with serial clock input or output.

[2] The received data is stored in RSR in LSB-to-MSB order.

After reception, the SCI checks whether the RDRF flag is 0 and the receive data can be transferred from RSR to RDR.

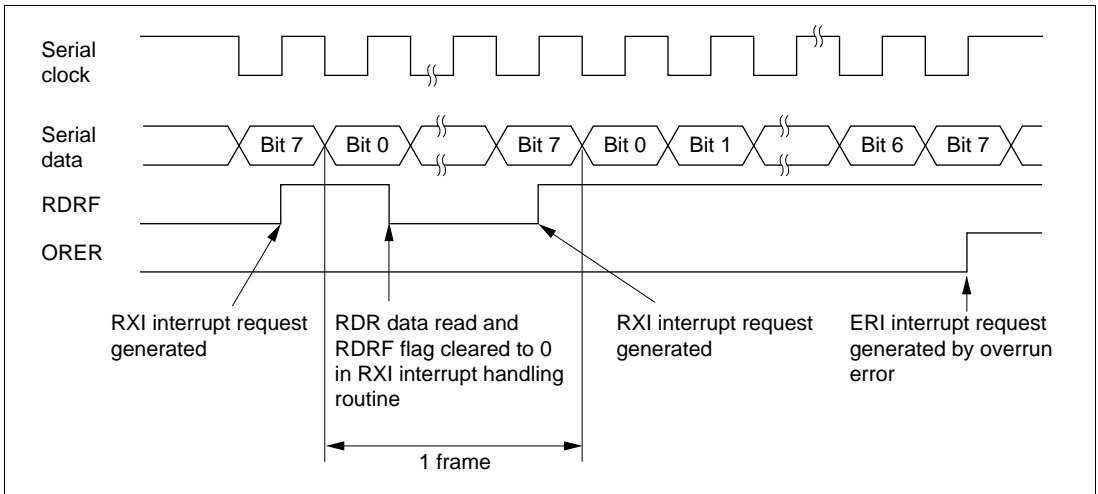
If this check is passed, the RDRF flag is set to 1, and the receive data is stored in RDR. If a receive error is detected in the error check, the operation is as shown in table 11-11.

Neither transmit nor receive operations can be performed subsequently when a receive error has been found in the error check.

[3] If the RIE bit in SCR is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR is set to 1 when the ORER flag changes to 1, a receive-error interrupt (ERI) request is generated.

Figure 11-19 shows an example of SCI operation in reception.

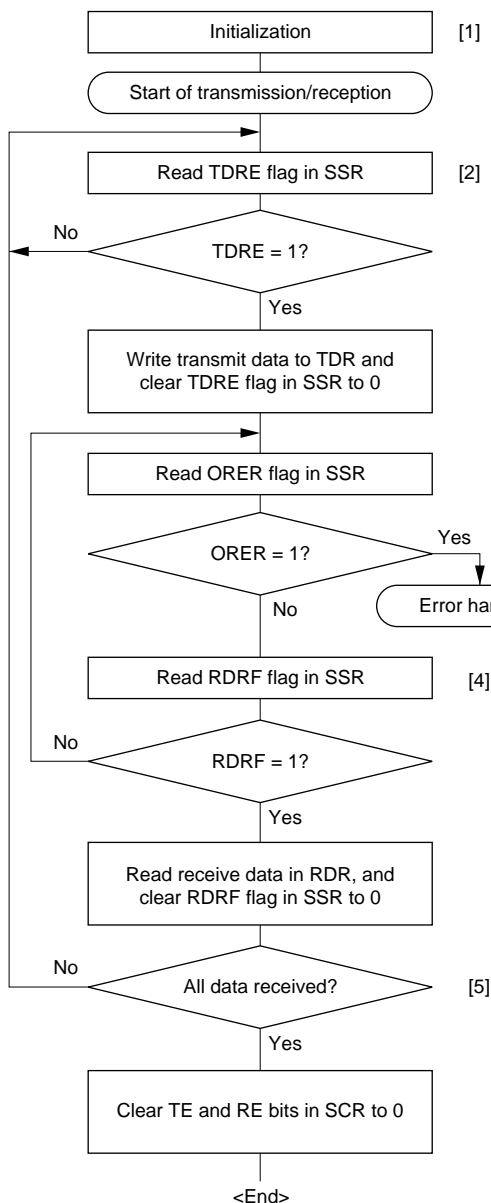


**Figure 11-19 Example of SCI Receive Operation**

- Simultaneous serial data transmission and reception (synchronous mode)

Figure 11-20 shows a sample flowchart for simultaneous serial transmit and receive operations.

The following procedure should be used for simultaneous serial data transmit and receive operations.



- [1] SCI initialization:  
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.
- [2] SCI status check and transmit data write:  
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.
- [3] Receive error handling:  
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error handling, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.
- [4] SCI status check and receive data read:  
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
- [5] Serial transmission/reception continuation procedure:  
To continue serial transmission/reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0. Checking and clearing of the TDRE flag is automatic when the DMAC or DTC is activated by a transmit-data-empty interrupt (TXI) request and data is written to TDR. Also, the RDRF flag is cleared automatically when the DMAC or DTC is activated by a receive-data-full interrupt (RXI) request and the RDR value is read.

Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE and RE bits to 0, then set both these bits to 1 simultaneously.

Figure 11-20 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations

## 11.4 SCI Interrupts

The SCI has four interrupt sources: the transmit-end interrupt (TEI) request, receive-error interrupt (ERI) request, receive-data-full interrupt (RXI) request, and transmit-data-empty interrupt (TXI) request. Table 11-12 shows the interrupt sources and their relative priorities. Individual interrupt sources can be enabled or disabled with the TIE, RIE, and TEIE bits in the SCR. Each kind of interrupt request is sent to the interrupt controller independently.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt can activate the DMAC or DTC to perform data transfer. The TDRE flag is cleared to 0 automatically when data transfer is performed by the DMAC or DTC. The DMAC and DTC cannot be activated by a TEI interrupt request.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DMAC or DTC to perform data transfer. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DMAC or DTC. The DMAC and DTC cannot be activated by an ERI interrupt request.

Also note that the DMAC cannot be activated by an SCI channel 2 interrupt.



## 11.5 Usage Notes

The following points should be noted when using the SCI.

**Relation between Writes to TDR and the TDRE Flag:** The TDRE flag in SSR is a status flag that indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR regardless of the state of the TDRE flag. However, if new data is written to TDR when the TDRE flag is cleared to 0, the data stored in TDR will be lost since it has not yet been transferred to TSR. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to TDR.

**Operation when Multiple Receive Errors Occur Simultaneously:** If a number of receive errors occur at the same time, the state of the status flags in SSR is as shown in table 11-13. If there is an overrun error, data is not transferred from RSR to RDR, and the receive data is lost.

**Table 11-13 State of SSR Status Flags and Transfer of Receive Data**

SSR Status Flags				Receive Data Transfer	Receive Error Status
RDRF	ORER	FER	PER	from RSR to RDR	
1	1	0	0	X	Overrun error
0	0	1	0	○	Framing error
0	0	0	1	○	Parity error
1	1	1	0	X	Overrun error + framing error
1	1	0	1	X	Overrun error + parity error
0	0	1	1	○	Framing error + parity error
1	1	1	1	X	Overrun error + framing error + parity error

Notes: ○: Receive data is transferred from RSR to RDR.

X: Receive data is not transferred from RSR to RDR.



**Break Detection and Processing (Asynchronous Mode Only):** When framing error (FER) detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set, and the parity error flag (PER) may also be set.

Note that, since the SCI continues the receive operation after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

**Sending a Break (Asynchronous Mode Only):** The TxD pin has a dual function as an I/O port whose direction (input or output) is determined by DR and DDR. This can be used to send a break.

Between serial transmission initialization and setting of the TE bit to 1, the mark state is replaced by the value of DR (the pin does not function as the TxD pin until the TE bit is set to 1). Therefore, DDR and DR for the port corresponding to the TxD pin should first be set to 1.

To send a break during serial transmission, first clear DR to 0, then clear the TE bit to 0.

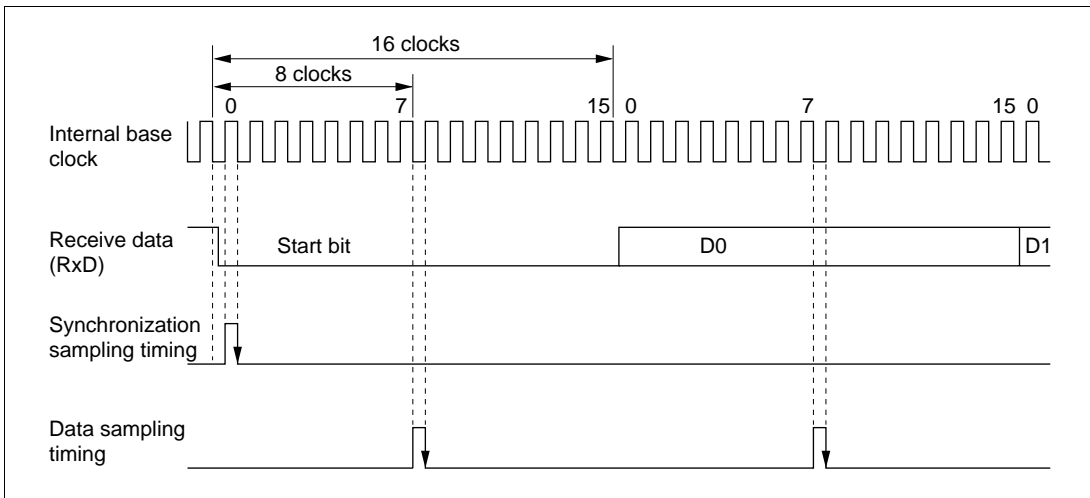
When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

**Receive Error Flags and Transmit Operations (Synchronous Mode Only):** Transmission cannot be started when a receive error flag (ORER, PER, or FER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission.

Note also that receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

**Receive Data Sampling Timing and Receive Margin in Asynchronous Mode:** In asynchronous mode, the SCI operates on a base clock with a frequency of 16 times the transfer rate.

In reception, the SCI samples the falling edge of the start bit using the base clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the base clock. This is illustrated in figure 11-21.



**Figure 11-21 Receive Data Sampling Timing in Asynchronous Mode**

Thus the receive margin in asynchronous mode is given by formula (1) below.

$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\%$$

... Formula (1)

- Where
- M : Receive margin (%)
  - N : Ratio of bit rate to clock (N = 16)
  - D : Clock duty (D = 0 to 1.0)
  - L : Frame length (L = 9 to 12)
  - F : Absolute value of clock rate deviation

Assuming values of F = 0 and D = 0.5 in formula (1), a receive margin of 46.875% is given by formula (2) below.

When D = 0.5 and F = 0,

$$M = \left( 0.5 - \frac{1}{2 \times 16} \right) \times 100\%$$

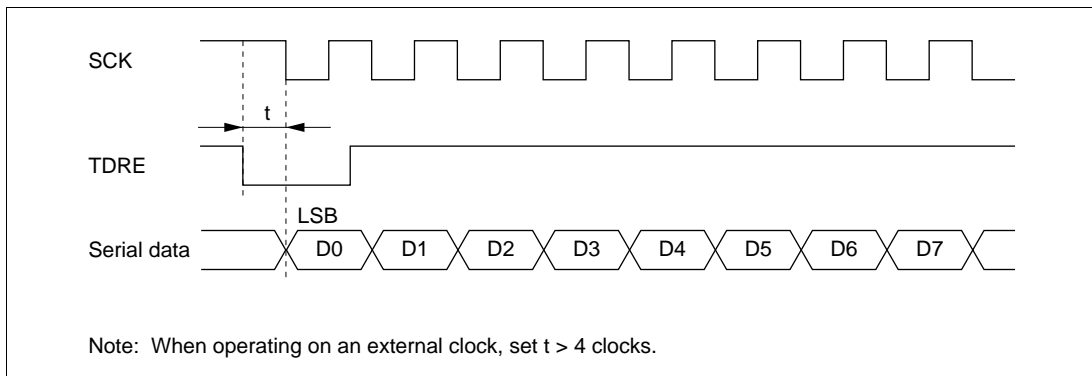
$$= 46.875\%$$

... Formula (2)

However, this is a theoretical value, and a margin of 20% to 30% should be allowed in system design.

## Restrictions on Use of DMAC or DTC

- When an external clock source is used as the serial clock, the transmit clock should not be input until at least 5  $\phi$  clock cycles after TDR is updated by the DMAC or DTC. Misoperation may occur if the transmit clock is input within 4  $\phi$  clocks after TDR is updated. (Figure 11-22)
- When RDR is read by the DMAC or DTC, be sure to set the activation source to the relevant SCI receive-data-full interrupt (RXI).



**Figure 11-22 Example of Synchronous Transmission Using DTC**

## Operation in Case of Mode Transition

### • Transmission

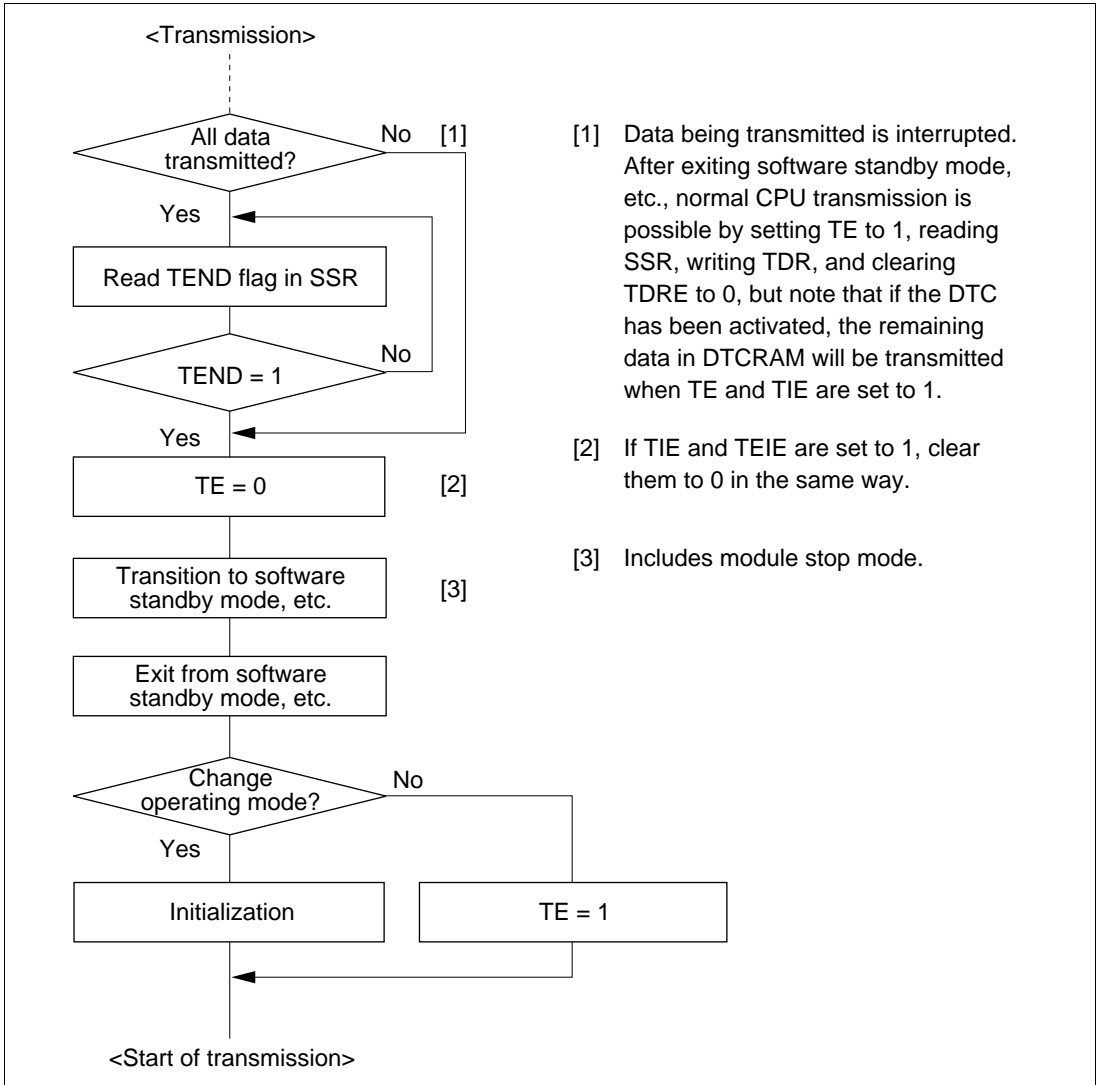
Operation should be stopped (by clearing TE, TIE, and TEIE to 0) before making a module stop mode or software standby mode transition. TSR, TDR, and SSR are reset. The output pin states in module stop mode or software standby mode depend on the port settings, and becomes high-level output after the relevant mode is cleared. If a transition is made during transmission, the data being transmitted will be undefined. When transmitting without changing the transmit mode after the relevant mode is cleared, transmission can be started by setting TE to 1 again, and performing the following sequence: SSR read → TDR write → TDRE clearance. To transmit with a different transmit mode after clearing the relevant mode, the procedure must be started again from initialization. Figure 11-23 shows a sample flowchart for mode transition during transmission. Port pin states are shown in figures 11-24 and 11-25. Operation should also be stopped (by clearing TE, TIE, and TEIE to 0) before making a transition from transmission by DTC transfer to module stop mode or software standby mode transition. To perform transmission with the DTC after the relevant mode is cleared, setting TE and TIE to 1 will set the TXI flag and start DTC transmission.

- Reception

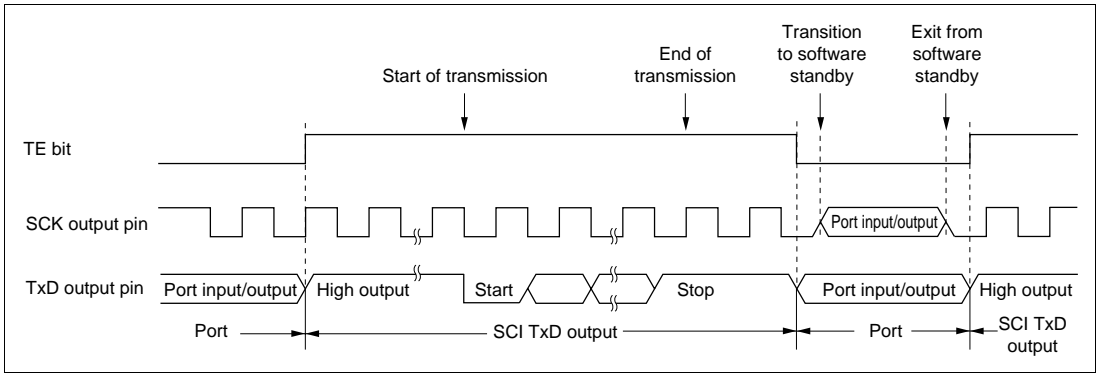
Receive operation should be stopped (by clearing RE to 0) before making a module stop mode or software standby mode transition. RSR, RDR, and SSR are reset. If a transition is made without stopping operation, the data being received will be invalid.

To continue receiving without changing the reception mode after the relevant mode is cleared, set RE to 1 before starting reception. To receive with a different receive mode, the procedure must be started again from initialization.

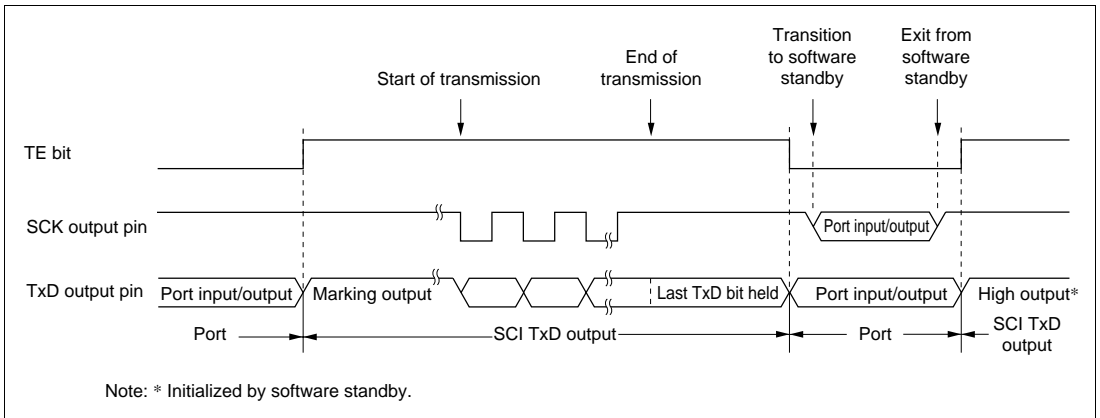
Figure 11-26 shows a sample flowchart for mode transition during reception.



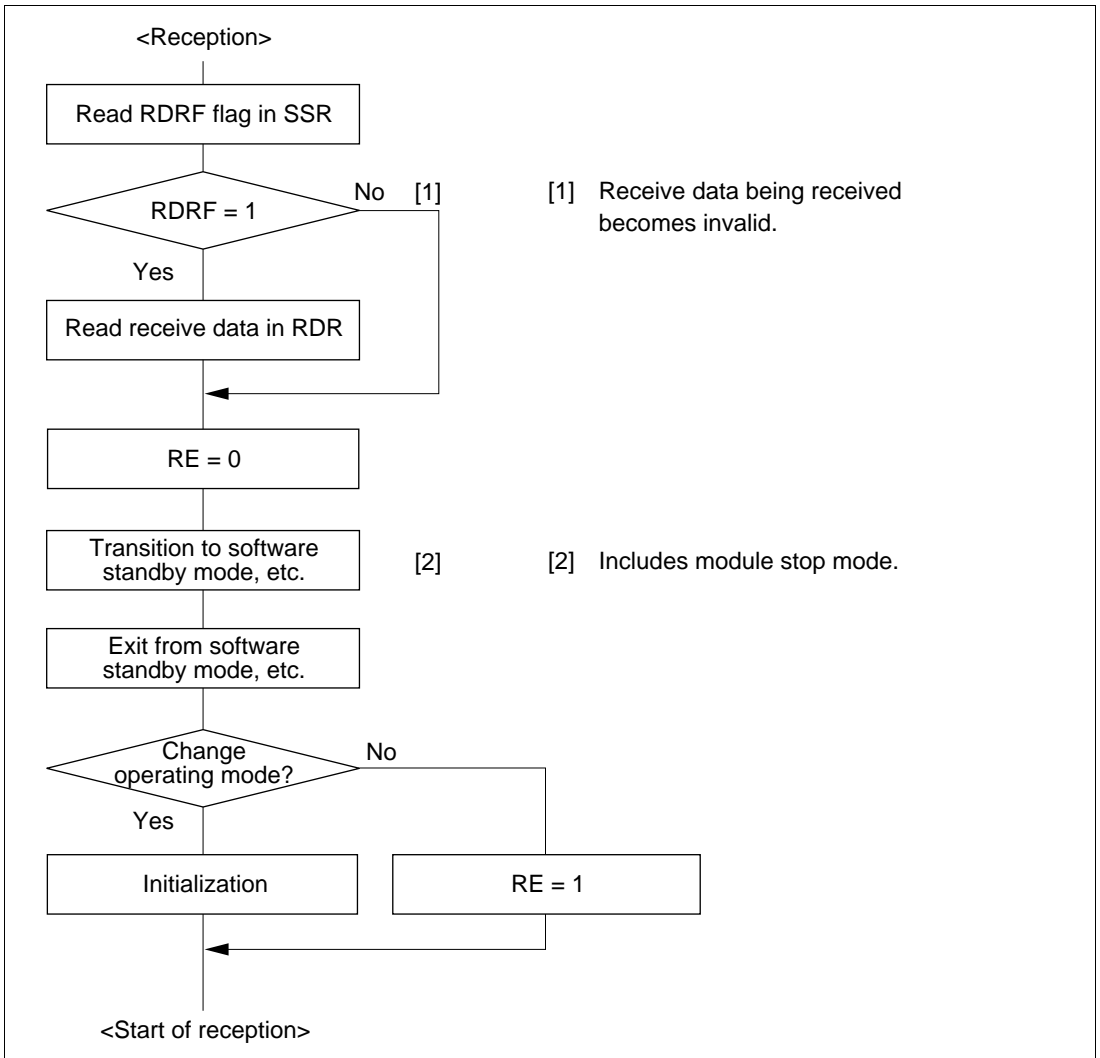
**Figure 11-23 Sample Flowchart for Mode Transition during Transmission**



**Figure 11-24 Asynchronous Transmission Using Internal Clock**



**Figure 11-25 Synchronous Transmission Using Internal Clock**



**Figure 11-26 Sample Flowchart for Mode Transition during Reception**

# Section 12 Smart Card Interface

## 12.1 Overview

The SCI supports an IC card (smart card) interface conforming to ISO/IEC 7816-3 (identification card) as a serial communication interface extension function.

Switching between the normal serial communication interface and the smart card interface is carried out by means of a register setting.

### 12.1.1 Features

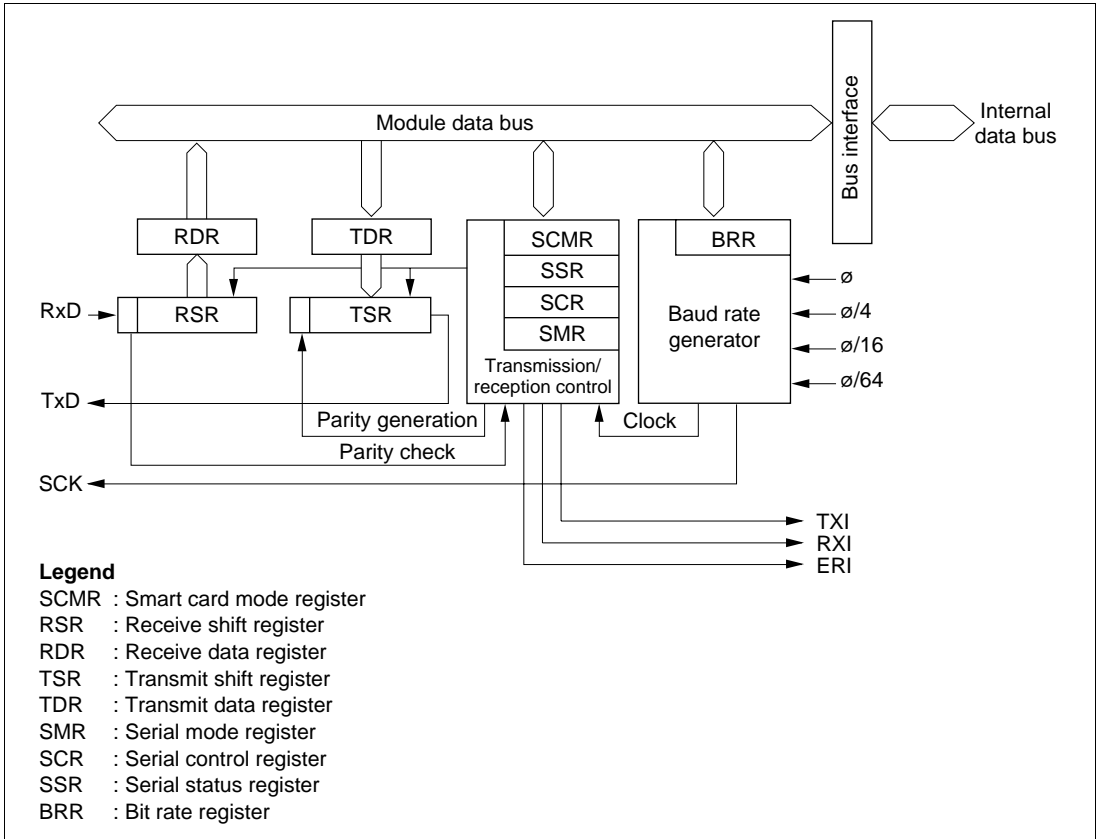
Features of the smart card interface supported by the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series are as follows.

- Asynchronous mode
  - Data length: 8 bits
  - Parity bit generation and checking
  - Transmission of error signal (parity error) in receive mode
  - Error signal detection and automatic data retransmission in transmit mode
  - Direct convention and inverse convention both supported
- Built-in baud rate generator allows any bit rate to be selected
- Three interrupt sources
  - Three interrupt sources (transmit-data-empty, receive-data-full, and transmit/receive-error) that can issue requests independently
  - The transmit-data-empty and receive-data-full interrupts can activate the DMA controller (DMAC)\* or data transfer controller (DTC) to execute data transfer

Note: \* Some models do not support a DMAC; please check the reference manual for the relevant model for confirmation.

## 12.1.2 Block Diagram

Figure 12-1 shows a block diagram of the smart card interface.



**Figure 12-1 Block Diagram of Smart Card Interface**



### 12.1.3 Pin Configuration

Table 12-1 shows the smart card interface pin configuration.

**Table 12-1 Smart Card Interface Pins**

<b>Channel*</b>	<b>Pin Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
0	Serial clock pin 0	SCK0	I/O	SCI0 clock input/output
	Receive data pin 0	RxD0	Input	SCI0 receive data input
	Transmit data pin 0	TxD0	Output	SCI0 transmit data output
1	Serial clock pin 1	SCK1	I/O	SCI1 clock input/output
	Receive data pin 1	RxD1	Input	SCI1 receive data input
	Transmit data pin 1	TxD1	Output	SCI1 transmit data output
2	Serial clock pin 2	SCK2	I/O	SCI2 clock input/output
	Receive data pin 2	RxD2	Input	SCI2 receive data input
	Transmit data pin 2	TxD2	Output	SCI2 transmit data output

Note: \* The number of channels differs from model to model; see the reference manual for the relevant model for details.

## 12.1.4 Register Configuration

Table 12-2 shows the registers used by the smart card interface. Details of SMR, BRR, SCR, TDR, RDR, and MSTPCR are the same as for the normal SCI function: see the register descriptions in section 11, Serial Communication Interface.

**Table 12-2 Smart Card Interface Registers**

Channel* <sup>1</sup>	Name	Abbreviation	R/W	Initial Value	Address* <sup>2</sup>
0	Serial mode register 0	SMR0	R/W	H'00	H'FF78
	Bit rate register 0	BRR0	R/W	H'FF	H'FF79
	Serial control register 0	SCR0	R/W	H'00	H'FF7A
	Transmit data register 0	TDR0	R/W	H'FF	H'FF7B
	Serial status register 0	SSR0	R/(W)* <sup>3</sup>	H'84	H'FF7C
	Receive data register 0	RDR0	R	H'00	H'FF7D
	Smart card mode register 0	SCMR0	R/W	H'F2	H'FF7E
1	Serial mode register 1	SMR1	R/W	H'00	H'FF80
	Bit rate register 1	BRR1	R/W	H'FF	H'FF81
	Serial control register 1	SCR1	R/W	H'00	H'FF82
	Transmit data register 1	TDR1	R/W	H'FF	H'FF83
	Serial status register 1	SSR1	R/(W)* <sup>3</sup>	H'84	H'FF84
	Receive data register 1	RDR1	R	H'00	H'FF85
	Smart card mode register 1	SCMR1	R/W	H'F2	H'FF86
2	Serial mode register 2	SMR2	R/W	H'00	H'FF88
	Bit rate register 2	BRR2	R/W	H'FF	H'FF89
	Serial control register 2	SCR2	R/W	H'00	H'FF8A
	Transmit data register 2	TDR2	R/W	H'FF	H'FF8B
	Serial status register 2	SSR2	R/(W)* <sup>3</sup>	H'84	H'FF8C
	Receive data register 2	RDR2	R	H'00	H'FF8D
	Smart card mode register 2	SCMR2	R/W	H'F2	H'FF8E
All	Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. The number of channels differs from model to model; see the reference manual for the relevant model for details.

2. Lower 16 bits of the address.

3. Can only be written with 0 for flag clearing.

## 12.2 Register Descriptions

Registers added with the smart card interface and bits for which the function changes are described here.

### 12.2.1 Smart Card Mode Register (SCMR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	SDIR	SINV	—	SMIF
Initial value :		1	1	1	1	0	0	1	0
R/W	:	—	—	—	—	R/W	R/W	—	R/W

SCMR is an 8-bit readable/writable register that selects the smart card interface function.

SCMR is initialized to HF2 by a reset and in hardware standby mode. In software standby mode and module stop mode it retains its previous state.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 1.

**Bit 3—Smart Card Data Transfer Direction (SDIR):** Selects the serial/parallel conversion format.

#### Bit 3

SDIR	Description
0	TDR contents are transmitted LSB-first Receive data is stored in RDR LSB-first (Initial value)
1	TDR contents are transmitted MSB-first Receive data is stored in RDR MSB-first

**Bit 2—Smart Card Data Invert (SINV):** Specifies inversion of the data logic level. This function is used together with the SDIR bit for communication with an inverse convention card. The SINV bit does not affect the logic level of the parity bit. For parity-related setting procedures, see section 12.3.4, Register Settings.

#### Bit 2

SINV	Description
0	TDR contents are transmitted as they are Receive data is stored as it is in RDR (Initial value)
1	TDR contents are inverted before being transmitted Receive data is stored in inverted form in RDR

**Bit 1—Reserved:** Read-only bit, always read as 1.

**Bit 0—Smart Card Interface Mode Select (SMIF):** Enables or disables the smart card interface function.

Bit 0 SMIF	Description
0	Smart card interface function is disabled (Initial value)
1	Smart card interface function is enabled

### 12.2.2 Serial Status Register (SSR)

Bit	:	7	6	5	4	3	2	1	0
		TDRE	RDRF	ORER	ERS	PER	TEND	MPB	MPBT
Initial value :		1	0	0	0	0	1	0	0
R/W	:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: \* Only 0 can be written to bits 7 to 3, to clear these flags.

Bit 4 of SSR has a different function in smart card interface mode. Coupled with this, the setting conditions for bit 2, TEND, are also different.

**Bits 7 to 5**—Operate in the same way as for the normal SCI. For details, see section 11.2.7, Serial Status Register (SSR).

**Bit 4—Error Signal Status (ERS):** In smart card interface mode, bit 4 indicates the status of the error signal sent back from the receiving end in transmission. Framing errors are not detected in smart card interface mode.

Bit 4 ERS	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"><li>• Upon reset, and in standby mode or module stop mode</li><li>• When 0 is written to ERS after reading ERS = 1</li></ul>
1	[Setting condition] When the low level of the error signal is sampled

Note: Clearing the TE bit in SCR to 0 does not affect the ERS flag, which retains its previous state.

**Bits 3 to 0**—Operate in the same way as for the normal SCI. For details, see section 11.2.7, Serial Status Register (SSR).

However, the setting conditions for the TEND bit, are as shown below.

**Bit 2**

TEND	Description
0	[Clearing conditions] <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE = 1</li> <li>• When the DMAC or DTC is activated by a TXI interrupt and writes data to TDR</li> </ul>
1	[Setting conditions] <span style="float: right;">(Initial value)</span> <ul style="list-style-type: none"> <li>• Upon reset, and in standby mode or module stop mode</li> <li>• When the TE bit in SCR is 0 and the ERS bit is also 0</li> <li>• When TDRE = 1 and ERS = 0 (normal transmission) 2.5 etu after transmission of a 1-byte serial character when GM = 0 and BLK = 0</li> <li>• When TDRE = 1 and ERS = 0 (normal transmission) 1.5 etu after transmission of a 1-byte serial character when GM = 0 and BLK = 1</li> <li>• When TDRE = 1 and ERS = 0 (normal transmission) 1.0 etu after transmission of a 1-byte serial character when GM = 1 and BLK = 0</li> <li>• When TDRE = 1 and ERS = 0 (normal transmission) 1.0 etu after transmission of a 1-byte serial character when GM = 1 and BLK = 1</li> </ul>

Note: etu: Elementary time unit (time for transfer of 1 bit)

**12.2.3 Serial Mode Register (SMR)**

Bit	7	6	5	4	3	2	1	0
	GM	BLK	PE*	O/ $\bar{E}$	BCP1	BCP0	CKS1	CKS0
Initial value :	0	0	0	0	0	0	0	0
R/W :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* When the smart card interface is used, set a value of 1 in bit 5.

The function of bits 7, 6, 3, and 2 of SMR changes in smart card interface mode.

**Bit 7—GSM Mode (GM):** Sets the smart card interface function to GSM mode.

This bit is cleared to 0 when the normal smart card interface is used. In GSM mode, this bit is set to 1, the timing of setting of the TEND flag that indicates transmission completion is advanced, and clock output control mode addition is performed. The contents of the clock output control mode addition are specified by bits 1 and 0 of the serial control register (SCR).

Bit 7 GM	Description
0	Normal smart card interface mode operation (Initial value) <ul style="list-style-type: none"> <li>TEND flag generation 12.5 etu (11.5 etu in block transfer mode) after beginning of start bit</li> <li>Clock output on/off control only</li> </ul>
1	GSM mode smart card interface mode operation <ul style="list-style-type: none"> <li>TEND flag generation 11.0 etu after beginning of start bit</li> <li>High/low fixing control possible in addition to clock output on/off control (set by SCR)</li> </ul>

Note: etu: Elementary time unit (time for transfer of 1 bit)

**Bit 6—Block Transfer Mode (BLK):** Selects block transfer mode.

Bit 6 BLK	Description
0	Normal smart card interface mode operation (Initial value) <ul style="list-style-type: none"> <li>Error signal transmission/detection and automatic data retransmission performed</li> <li>TXI interrupt generated by TEND flag</li> <li>TEND flag set 12.5 etu after start of transmission (11.0 etu in GSM mode)</li> </ul>
1	Block transfer mode operation <ul style="list-style-type: none"> <li>Error signal transmission/detection and automatic data retransmission not performed</li> <li>TXI interrupt generated by TDRE flag</li> <li>TEND flag set 11.5 etu after start of transmission (11.0 etu in GSM mode)</li> </ul>

**Bits 3 and 2—Base Clock Pulse 1 and 2 (BCP1, BCP0):** These bits specify the number of base clock periods in a 1-bit transfer interval on the smart card interface.

Bit 3 BCP1	Bit 2 BCP0	Description
0	0	32 clock periods (Initial value)
	1	64 clock periods
1	0	372 clock periods
	1	256 clock periods

**Bits 5, 4, 1, and 0—**Operate in the same way as for the normal SCI. For details, see section 11.2.5, Serial Mode Register (SMR).

## 12.2.4 Serial Control Register (SCR)

Bit	:	7	6	5	4	3	2	1	0
		TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

In smart card interface mode, the function of bits 1 and 0 of SCR changes when bit 7 of the serial mode register (SMR) is set to 1.

**Bits 7 to 2**—Operate in the same way as for the normal SCI. For details, see section 11.2.6, Serial Control Register (SCR).

**Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0):** These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin.

In smart card interface mode, in addition to the normal switching between clock output enabling and disabling, the clock output can be specified as being fixed high or low.

SCMR	SMR	SCR Setting		
SMIF	C/A, GM	CKE1	CKE0	SCK Pin Function
0	See the SCI specification			
1	0	0	0	Operates as port I/O pin
1	0	0	1	Outputs clock as SCK output pin
1	1	0	0	Operates as SCK output pin, with output fixed low
1	1	0	1	Outputs clock as SCK output pin
1	1	1	0	Operates as SCK output pin, with output fixed high
1	1	1	1	Outputs clock as SCK output pin

## 12.3 Operation

### 12.3.1 Overview

The main functions of the smart card interface are as follows.

- One frame consists of 8-bit data plus a parity bit.
- In transmission, a guard time of at least 2 etu (1 etu in block transfer mode) (elementary time unit: the time for transfer of one bit) is left between the end of the parity bit and the start of the next frame.
- If a parity error is detected during reception, a low error signal level is output for one etu period, 10.5 etu after the start bit. (This does not apply to block transfer mode.)
- If the error signal is sampled during transmission, the same data is transmitted automatically after the elapse of 2 etu or longer. (This does not apply to block transfer mode.)
- Only asynchronous communication is supported; there is no synchronous communication function.

### 12.3.2 Pin Connections

Figure 12-2 shows a schematic diagram of smart card interface related pin connections.

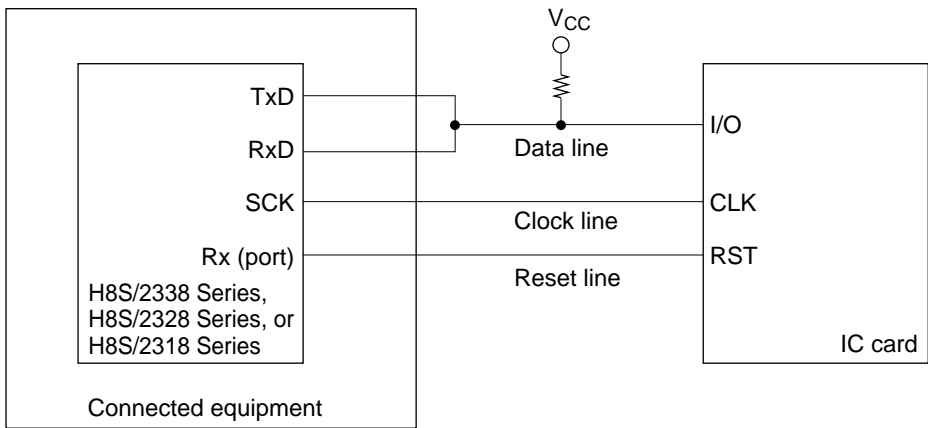
In communication with an IC card, since both transmission and reception are carried out on a single data communication line, the chip's TxD pin and RxD pin should both be connected to the line, as shown in the figure. The data communication line should be pulled up to the  $V_{CC}$  power supply with a resistor.

When the clock generated on the smart card interface is used by an IC card, the SCK pin output is input to the CLK pin of the IC card. No connection is needed if the IC card uses an internal clock.

Chip port output is used as the reset signal.

Other pins must normally be connected to the power supply or ground.



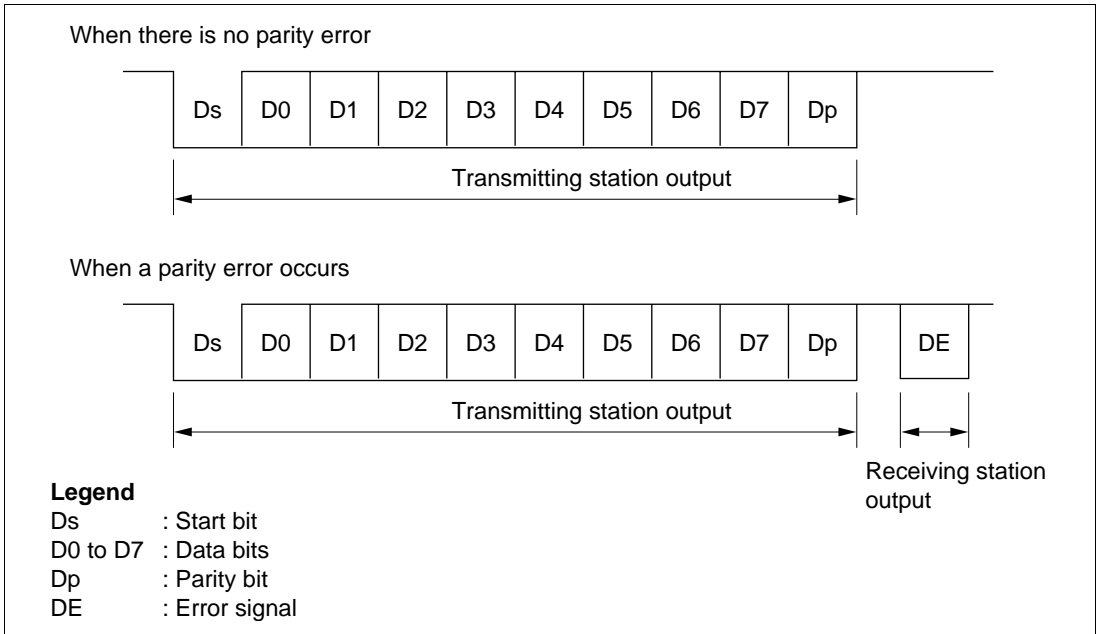


**Figure 12-2 Schematic Diagram of Smart Card Interface Pin Connections**

Note: If an IC card is not connected, and the TE and RE bits are both set to 1, closed transmission/reception is possible, enabling self-diagnosis to be carried out.

### 12.3.3 Data Format

**Normal Transfer Mode:** Figure 12-3 shows the smart card interface data format in the normal transfer mode. In reception in this mode, a parity check is carried out on each frame. If an error is detected an error signal is sent back to the transmitting end, and retransmission of the data is requested. If an error signal is sampled during transmission, the same data is retransmitted.



**Figure 12-3 Smart Card Interface Data Format**

The operation sequence is as follows.

- [1] When the data line is not in use it is in the high-impedance state, and is fixed high with a pull-up resistor.
- [2] The transmitting station starts transfer of one frame of data. The data frame starts with a start bit (Ds, low-level), followed by 8 data bits (D0 to D7) and a parity bit (Dp).
- [3] With the smart card interface, the data line then returns to the high-impedance state. The data line is pulled high with a pull-up resistor.
- [4] The receiving station carries out a parity check.  
If there is no parity error and the data is received normally, the receiving station waits for reception of the next data.  
If a parity error occurs, however, the receiving station outputs an error signal (DE, low-level) to request retransmission of the data. After outputting the error signal for the prescribed length of time, the receiving station places the signal line in the high-impedance state again. The signal line is pulled high again by a pull-up resistor.
- [5] If the transmitting station does not receive an error signal, it proceeds to transmit the next data frame.  
If it does receive an error signal, however, it returns to step [2] and retransmits the data in which the error occurred.

**Block Transfer Mode:** The operation sequence in block transfer mode is as follows.

- [1] When the data line is not in use it is in the high-impedance state, and is fixed high with a pull-up resistor.
- [2] The transmitting station starts transfer of one frame of data. The data frame starts with a start bit (Ds, low-level), followed by 8 data bits (D0 to D7) and a parity bit (Dp).
- [3] With the smart card interface, the data line then returns to the high-impedance state. The data line is pulled high with a pull-up resistor.
- [4] The receiving station carries out a parity check, but does not output an error signal even if an error has occurred. Since subsequent receive operations cannot be carried out if an error occurs, the error flag must be cleared to 0 before the parity bit for the next frame is received.
- [5] The transmitting station proceeds to transmit the next data frame.

## 12.3.4 Register Settings

Table 12-3 shows a bit map of the registers used by the smart card interface.

Bits indicated as 0 or 1 must be set to the value shown. The setting of other bits is described below.

**Table 12-3 Smart Card Interface Register Settings**

Register	Bit							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SMR	GM	BLK	1	O/ $\bar{E}$	BCP1	BCP0	CKS1	CKS0
BRR	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
SCR	TIE	RIE	TE	RE	0	0	CKE1*	CKE0
TDR	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
SSR	TDRE	RDRF	ORER	ERS	PER	TEND	0	0
RDR	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
SCMR	—	—	—	—	SDIR	SINV	—	SMIF

Notes: — : Unused bit.

\*: The CKE1 bit must be cleared to 0 when the GM bit in SMR is cleared to 0.

**SMR Settings:** The GM bit is cleared to 0 in normal smart card interface mode, and set to 1 in GSM mode. The O/ $\bar{E}$  bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

Bits CKS1 and CKS0 select the clock source of the built-in baud rate generator, and bits BCP1 and BCP0 select the number of base clock cycles during transfer of one bit. For details, see section 12.3.5, Clock.

The BLK bit is cleared to 0 when using the normal smart card interface mode, and set to 1 when using block transfer mode.

**BRR Setting:** BRR is used to set the bit rate. See section 12.3.5, Clock, for the method of calculating the value to be set.

**SCR Settings:** The function of the TIE, RIE, TE, and RE bits is the same as for the normal SCI. For details, see section 11, Serial Communication Interface.

Bits CKE1 and CKE0 specify the clock output. When the GM bit in SMR is cleared to 0, set these bits to B'00 if a clock is not to be output, or to B'01 if a clock is to be output. When the GM bit in SMR is set to 1, clock output is performed. The clock output can also be fixed high or low.

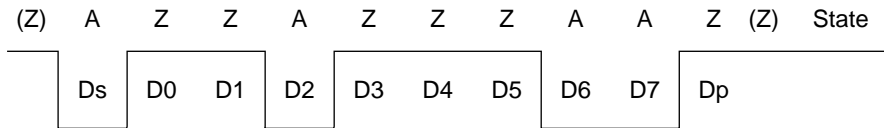
**Smart Card Mode Register (SCMR) Settings:** The SDIR bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

The SINV bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

The SMIF bit is set to 1 when the smart card interface is used.

Examples of register settings and the waveform of the start character are shown below for the two types of IC card (direct convention and inverse convention).

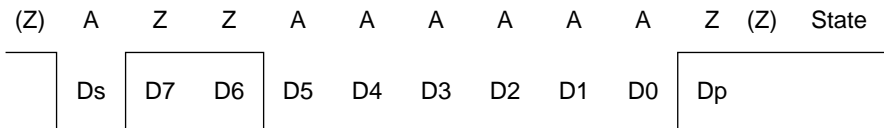
- Direct convention ( $SDIR = SINV = O/\bar{E} = 0$ )



With the direct convention type, the logic 1 level corresponds to state Z and the logic 0 level to state A, and transfer is performed in LSB-first order. The start character data above is H'3B.

The parity bit is 1 since even parity is stipulated for the smart card.

- Inverse convention ( $SDIR = SINV = O/\bar{E} = 1$ )



With the inverse convention type, the logic 1 level corresponds to state A and the logic 0 level to state Z, and transfer is performed in MSB-first order. The start character data above is H'3F.

The parity bit is 0, corresponding to state Z, since even parity is stipulated for the smart card.

With the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, inversion specified by the SINV bit applies only to the data bits, D7 to D0. For parity bit inversion, the  $O/\bar{E}$  bit in SMR should be set to odd parity mode (the same applies to both transmission and reception).

### 12.3.5 Clock

Only an internal clock generated by the built-in baud rate generator can be used as the transmit/receive clock for the smart card interface. The bit rate is set with BRR and the CKS1, CKS0, BCP1, and BCP0 bits in SMR. The formula for calculating the bit rate is as shown below. Table 12-5 shows some sample bit rates.

If clock output is selected by setting CKE0 to 1, the clock is output from the SCK pin. The clock frequency is determined by the bit rate and the setting of bits BCP1 and BCP0.

$$B = \frac{\phi}{S \times 2^{2n+1} \times (N + 1)} \times 10^6$$

Where: N = Value set in BRR ( $0 \leq N \leq 255$ )

B = Bit rate (bits/s)

$\phi$  = Operating frequency (MHz)

n = See table 12-4

S = Number of internal clock cycles in 1-bit period set by bits BCP1 and BCP0

**Table 12-4 Correspondence between n and CKS1, CKS0**

n	CKS1	CKS0
0	0	0
1		1
2	1	0
3		1

**Table 12-5 Examples of Bit Rate B (bits/s) for Various BRR Settings  
(When n = 0 and S = 372)**

N	$\phi$ (MHz)							
	10.00	10.714	13.00	14.285	16.00	18.00	20.00	25.00*
0	13441	14400	17473	19200	21505	24194	26882	33602
1	6720	7200	8737	9600	10753	12097	13441	16801
2	4480	4800	5824	6400	7168	8065	8961	11201

Note: Bit rates are rounded to the nearest whole number.

\* In planning stage

The method of calculating the value to be set in the bit rate register (BRR) from the operating frequency and bit rate, on the other hand, is shown below. N is an integer,  $0 \leq N \leq 255$ , and the smaller error is specified.

$$N = \frac{\phi}{S \times 2^{2n+1} \times B} \times 10^6 - 1$$

**Table 12-6 Examples of BRR Settings for Bit Rate B (bits/s) (When n = 0 and S = 372)**

Bits/s	$\phi$ (MHz)																			
	7.1424		10.00		10.7136		13.00		14.2848		16.00		18.00		20.00		25.00*			
	N	Error	N	Error	N	Error	N	Error	N	Error	N	Error	N	Error	N	Error	N	Error		
9600	0	0.00	1	30	1	25	1	8.99	1	0.00	1	12.01	2	15.99	2	6.60	3	12.49		

Note: \* In planning stage

**Table 12-7 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode)  
(When S = 372)**

$\phi$ (MHz)	Maximum Bit Rate (bits/s)	N	n
7.1424	9600	0	0
10.00	13441	0	0
10.7136	14400	0	0
13.00	17473	0	0
14.2848	19200	0	0
16.00	21505	0	0
18.00	24194	0	0
20.00	26882	0	0
25.00*	33602	0	0

Note: \* In planning stage

The bit rate error is given by the following formula:

$$\text{Error (\%)} = \left( \frac{\phi}{S \times 2^{2n+1} \times B \times (N + 1)} \times 10^6 - 1 \right) \times 100$$

### 12.3.6 Data Transfer Operations

**Initialization:** Before transmitting or receiving data, initialize the SCI as described below. Initialization is also necessary when switching from transmit mode to receive mode, or vice versa.

[1] Clear the TE and RE bits in SCR to 0.

[2] Clear the error flags ERS, PER, and ORER in SSR to 0.

[3] Set the GM, BLK,  $O/\bar{E}$ , BCP1, BCP0, CKS1, and CKS0 bits in SMR, and set the PE bit to 1.

[4] Set the SMIF, SDIR, and SINV bits in SCMR.

When the SMIF bit is set to 1, the TxD and RxD pins are both switched from ports to SCI pins, and are placed in the high-impedance state.

[5] Set the value corresponding to the bit rate in BRR.

[6] Set the CKE1 and CKE0 bits in SCR. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0.

If the CKE0 bit is set to 1, the clock is output from the SCK pin.

[7] Wait at least one bit interval, then set the TIE, RIE, TE, and RE bits in SCR. Do not set the TE bit and RE bit at the same time, except for self-diagnosis.



**Serial Data Transmission (Except Block Transfer Mode):** As data transmission in smart card mode involves error signal sampling and retransmission processing, the processing procedure is different from that for the normal SCI. Figure 12-4 shows a flowchart for transmitting, and figure 12-5 shows the relation between a transmit operation and the internal registers.

- [1] Perform smart card interface mode initialization as described above in Initialization.
- [2] Check that the ERS error flag in SSR is cleared to 0.
- [3] Repeat steps [2] and [3] until it can be confirmed that the TEND flag in SSR is set to 1.
- [4] Write the transmit data to TDR, clear the TDRE flag to 0, and perform the transmit operation. The TEND flag is cleared to 0.
- [5] When transmitting data continuously, go back to step [2].
- [6] To end transmission, clear the TE bit to 0.

With the above processing, interrupt handling or data transfer by the DMAC or DTC is possible.

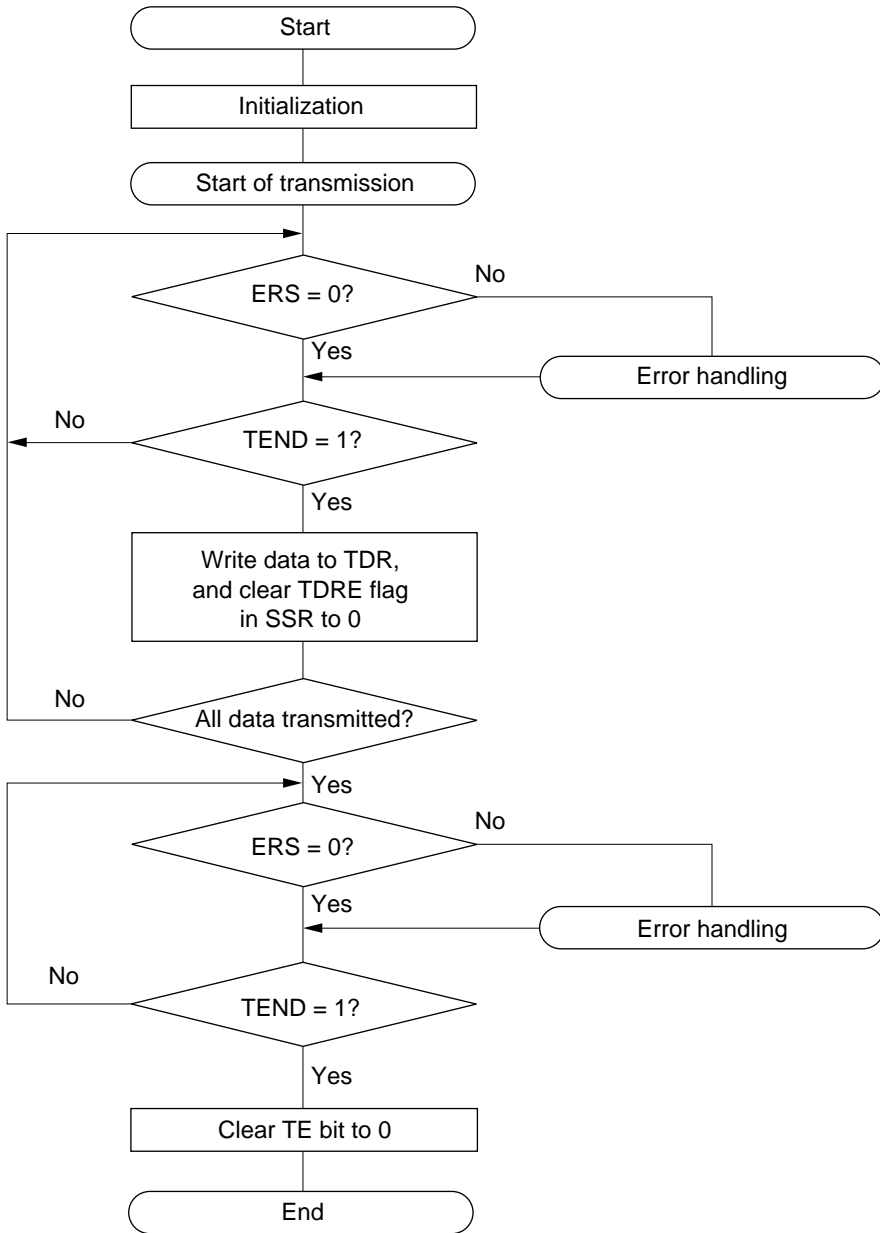
If transmission ends and the TEND flag is set to 1 while the TIE bit is set to 1 and interrupt requests are enabled, a transmit-data-empty interrupt (TXI) request will be generated. If an error occurs in transmission and the ERS flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a transmit/receive-error interrupt (ERI) request will be generated.

The timing for setting the TEND flag depends on the value of the GM bit in SMR. The TEND flag setting timing is shown in figure 12-6.

If the DMAC or DTC is activated by a TXI request, the number of bytes set in the DMAC or DTC can be transmitted automatically, including automatic retransmission.

For details, see Interrupt Operations and Data Transfer Operation by DMAC or DTC below.

**Note:** For details of operation in block transfer mode, see section 11.3.2, Operation in Asynchronous Mode.

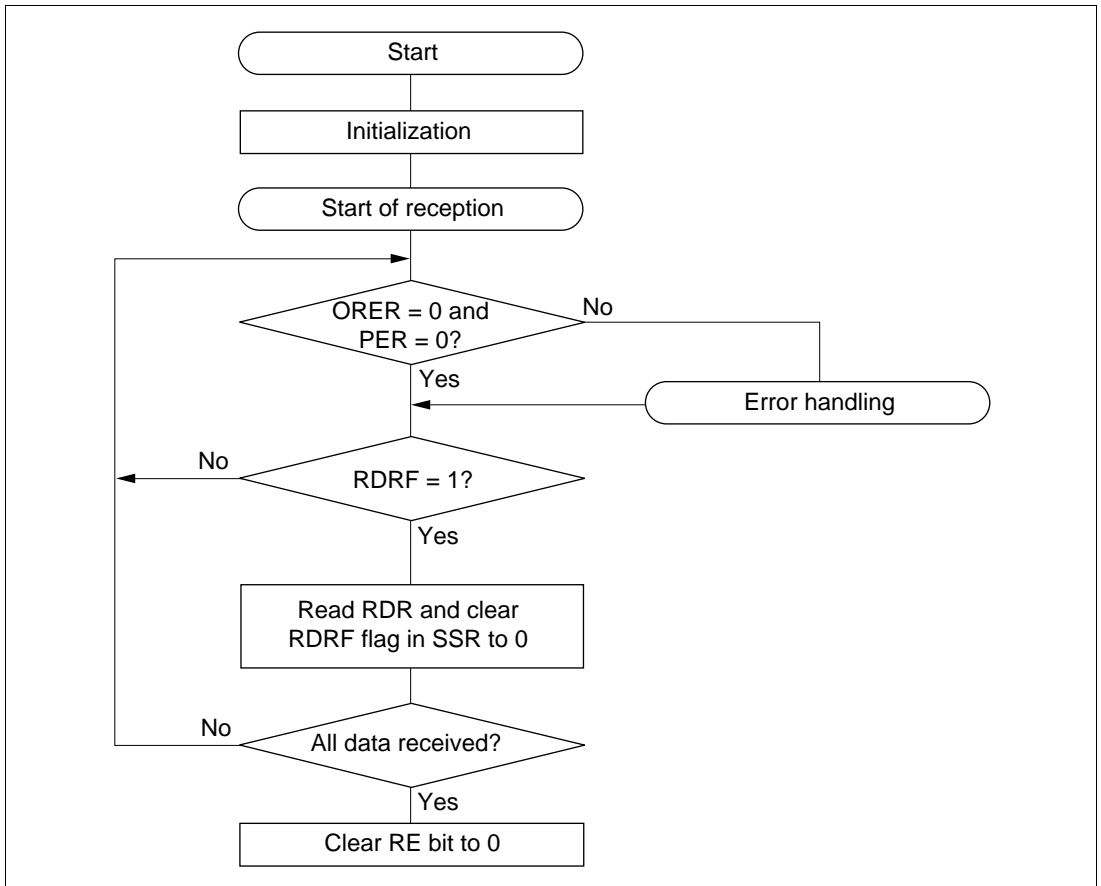


**Figure 12-4 Sample Transmission Flowchart**



**Serial Data Reception (Except Block Transfer Mode):** Data reception in smart card mode uses the same processing procedure as for the normal SCI. Figure 12-7 shows an example of the transmission processing flow.

- [1] Perform smart card interface mode initialization as described above in Initialization.
- [2] Check that the ORER flag and PER flag in SSR are cleared to 0. If either is set, perform the appropriate receive error handling, then clear both the ORER and the PER flag to 0.
- [3] Repeat steps [2] and [3] until it can be confirmed that the RDRF flag is set to 1.
- [4] Read the receive data from RDR.
- [5] When receiving data continuously, clear the RDRF flag to 0 and go back to step [2].
- [6] To end reception, clear the RE bit to 0.



**Figure 12-7 Sample Reception Flowchart**

With the above processing, interrupt handling or data transfer by the DMAC or DTC is possible.

If reception ends and the RDRF flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a receive data full interrupt (RXI) request will be generated. If an error occurs in reception and either the ORER flag or the PER flag is set to 1, a transmit/receive-error interrupt (ERI) request will be generated.

If the DMAC or DTC is activated by an RXI request, the receive data in which the error occurred is skipped, and only the number of bytes of receive data set in the DMAC or DTC are transferred.

For details, see Interrupt Operation and Data Transfer Operation by DMAC or DTC below.

If a parity error occurs during reception and the PER is set to 1, the received data is still transferred to RDR, and therefore this data can be read.

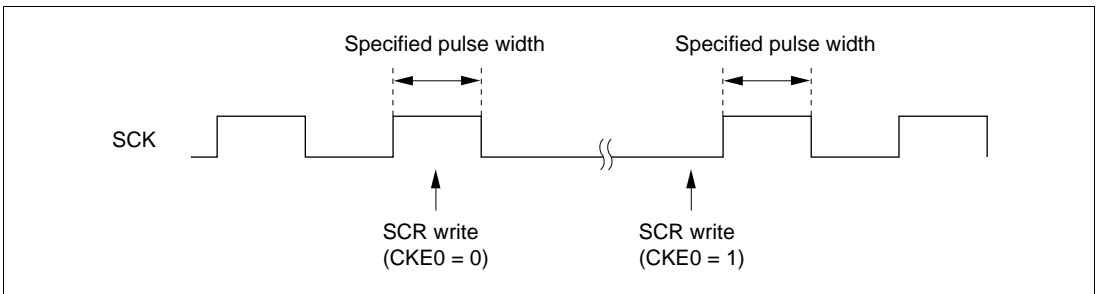
Note: For details of operation in block transfer mode, see section 11.3.2, Operation in Asynchronous Mode.

**Mode Switching Operation:** When switching from receive mode to transmit mode, first confirm that the receive operation has been completed, then start from initialization, clearing RE bit to 0 and setting TE bit to 1. The RDRF flag or the PER and ORER flags can be used to check that the receive operation has been completed.

When switching from transmit mode to receive mode, first confirm that the transmit operation has been completed, then start from initialization, clearing TE bit to 0 and setting RE bit to 1. The TEND flag can be used to check that the transmit operation has been completed.

**Fixing Clock Output:** When the GSM bit in SMR is set to 1, the clock output can be fixed with bits CKE1 and CKE0 in SCR. At this time, the minimum clock pulse width can be made the specified width.

Figure 12-8 shows the timing for fixing the clock output. In this example, GSM is set to 1, CKE1 is cleared to 0, and the CKE0 bit is controlled.



**Figure 12-8 Timing for Fixing Clock Output**

**Interrupt Operation (Except Block Transfer Mode):** There are three interrupt sources in smart card interface mode: transmit-data-empty interrupt (TXI) requests, transmit/receive-error interrupt (ERI) requests, and receive-data-full interrupt (RXI) requests. The transmit-end interrupt (TEI) request is not used in this mode.

When the TEND flag in SSR is set to 1, a TXI interrupt request is generated.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated.

When any of flags ORER, PER, and ERS in SSR is set to 1, an ERI interrupt request is generated. The relationship between the operating states and interrupt sources is shown in table 12-8.

Note: For details of operation in block transfer mode, see section 11.4, SCI Interrupts.

**Table 12-8 Smart Card Mode Operating States and Interrupt Sources**

Operating State		Flag	Enable Bit	Interrupt Source	DTC Activation	DMAC Activation
Transmit Mode	Normal operation	TEND	TIE	TXI	Possible	Possible
	Error	ERS	RIE	ERI	Not possible	Not possible
Receive Mode	Normal operation	RDRF	RIE	RXI	Possible	Possible
	Error	PER, ORER	RIE	ERI	Not possible	Not possible

**Data Transfer Operation by DMAC or DTC:** In smart card mode, as with the normal SCI, transfer can be carried out using the DMAC or DTC. In a transmit operation, the TDRE flag is also set to 1 at the same time as the TEND flag in SSR, and a TXI interrupt is generated. If the TXI request is designated beforehand as a DMAC or DTC activation source, the DMAC or DTC will be activated by the TXI request, and transfer of the transmit data will be carried out. The TDRE and TEND flags are automatically cleared to 0 when data transfer is performed by the DMAC or DTC. In the event of an error, the SCI retransmits the same data automatically. The TEND flag remains cleared to 0 during this time, and the DMAC is not activated. Thus, the number of bytes specified by the SCI and DMAC are transmitted automatically even in retransmission following an error. However, the ERS flag is not cleared automatically when an error occurs, and so the RIE bit should be set to 1 beforehand so that an ERI request will be generated in the event of an error, and the ERS flag will be cleared.

When performing transfer using the DMAC or DTC, it is essential to set and enable the DMAC or DTC before carrying out SCI setting. For details of the DMAC and DTC setting procedures, see section 5, DMA Controller (DMAC), and section 6, Data Transfer Controller (DTC).

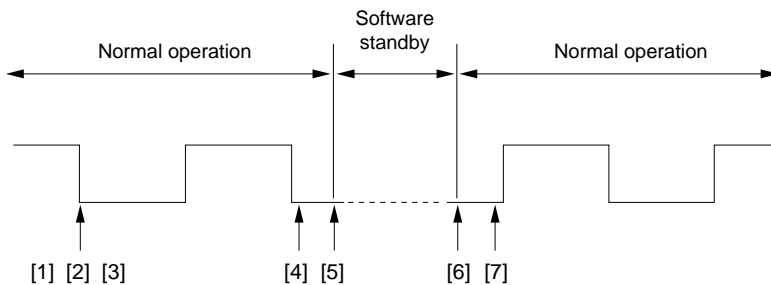
In a receive operation, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. If the RXI request is designated beforehand as a DMAC or DTC activation source, the DMAC or DTC will be activated by the RXI request, and transfer of the receive data will be carried out. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DMAC or DTC. If an error occurs, an error flag is set but the RDRF flag is not. Consequently, the DMAC or DTC is not activated, but instead, an ERI interrupt request is sent to the CPU. Therefore, the error flag should be cleared.

Note: For details of operation in block transfer mode, see section 11.4, SCI Interrupts.

### 12.3.7 Operation in GSM Mode

**Switching the Mode:** When switching between smart card interface mode and software standby mode, the following switching procedure should be followed in order to maintain the clock duty.

- When changing from smart card interface mode to software standby mode
  - [1] Set the data register (DR) and data direction register (DDR) corresponding to the SCK pin to the value for the fixed output state in software standby mode.
  - [2] Write 0 to the TE bit and RE bit in the serial control register (SCR) to halt the transmit/receive operation. At the same time, set the CKE1 bit to the value for the fixed output state in software standby mode.
  - [3] Write 0 to the CKE0 bit in SCR to halt the clock.
  - [4] Wait for one serial clock period.  
During this interval, clock output is fixed at the specified level, with the duty preserved.
  - [5] Write H'00 to SMR and SCMR.
  - [6] Make the transition to the software standby state.
- When returning to smart card interface mode from software standby mode
  - [7] Exit the software standby state.
  - [8] Set the CKE1 bit in SCR to the value for the fixed output state (current SCK pin state) when software standby mode is initiated.
  - [9] Set smart card interface mode and output the clock. Signal generation is started with the normal duty.



**Figure 12-9 Clock Halt and Restart Procedure**

**Powering On:** To secure the clock duty from power-on, the following switching procedure should be followed.

- [1] The initial state is port input and high impedance. Use a pull-up resistor or pull-down resistor to fix the potential.
- [2] Fix the SCK pin to the specified output level with the CKE1 bit in SCR.
- [3] Set SMR and SCMR, and switch to smart card mode operation.
- [4] Set the CKE0 bit in SCR to 1 to start clock output.

### 12.3.8 Operation in Block Transfer Mode

Operation in block transfer mode is the same as in SCI asynchronous mode, except for the following points. For details, see section 11.3.2, Operation in Asynchronous Mode.

**Data Format:** The data format is 8 bits with parity. There is no stop bit, but there is a guard time of 2 or more bits (1 or more bits in reception).

Also, except during transmission (with start bit, data bits, and parity bit), the transmission pins go to the high-impedance state, so the signal lines must be fixed high with a pull-up resistor.

**Transmit/Receive Clock:** Only an internal clock generated by the built-in baud rate generator can be used as the transmit/receive clock. The number of basic clock periods in a 1-bit transfer interval can be set to 32, 64, 372, or 256 with bits BCP1 and BCP0. For details, see section 12.3.5, Clock.

**ERS (FER) Flag:** As with the normal smart card interface, the ERS flag indicates the error signal status, but since error signal transmission and reception is not performed, this flag is always cleared to 0.

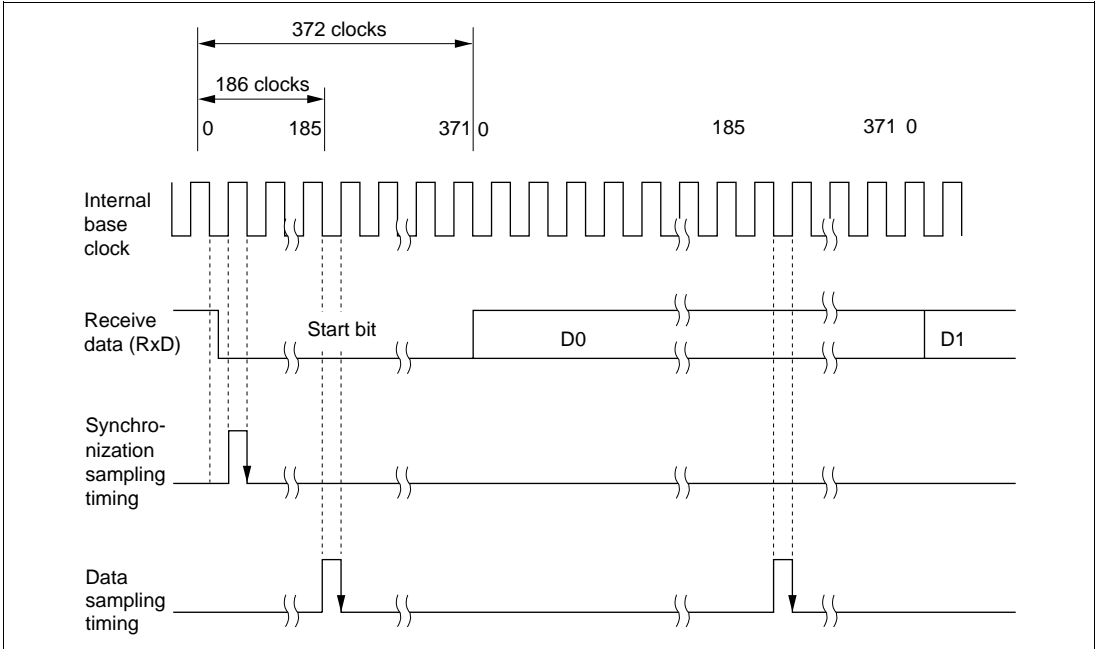
## 12.4 Usage Notes

The following points should be noted when using the SCI as a smart card interface.



**Receive Data Sampling Timing and Receive Margin in Smart Card Interface Mode:** In smart card interface mode, the SCI operates on a base clock with a frequency of 32, 64, 372, or 256 times the transfer rate (determined by bits BCP1 and BCP0).

In reception, the SCI samples the falling edge of the start bit using the base clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 16th, 32nd, 186th, or 128th pulse of the base clock. Use of a 372-times clock is illustrated in figure 12-10.



**Figure 12-10 Receive Data Sampling Timing in Smart Card Mode  
(When Using 372-Times Clock)**

Thus the receive margin in asynchronous mode is given by the following formula.

$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\%$$

Where M: Receive margin (%)

N: Ratio of bit rate to clock (N = 32, 64, 372, 256)

D: Clock duty (D = 0 to 1.0)

L: Frame length (L = 10)

F: Absolute value of clock frequency deviation

Assuming values of F = 0, D = 0.5, and N=372 in the above formula, the receive margin formula is as follows.

When  $D = 0.5$  and  $F = 0$ ,

$$M = (0.5 - 1/2 \times 372) \times 100\%$$

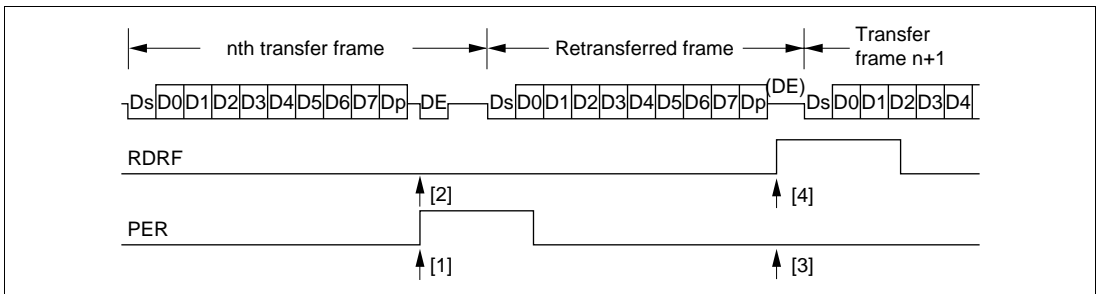
$$= 49.866\%$$

**Retransfer Operations (Except Block Transfer Mode):** Retransfer operations are performed by the SCI in receive mode and transmit mode as described below.

- Retransfer operation when SCI is in receive mode

Figure 12-11 illustrates the retransfer operation when the SCI is in receive mode.

- [1] If an error is found when the received parity bit is checked, the PER bit in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an ERI interrupt request is generated. The PER bit in SSR should be kept cleared to 0 until the next parity bit is sampled.
- [2] The RDRF bit in SSR is not set for a frame in which an error has occurred.
- [3] If no error is found when the received parity bit is checked, the PER bit in SSR is not set.
- [4] If no error is found when the received parity bit is checked, the receive operation is judged to have been completed normally, and the RDRF flag in SSR is automatically set to 1. If the RIE bit in SCR is enabled at this time, an RXI interrupt request is generated. If DMAC or DTC data transfer by an RXI source is enabled, the contents of RDR can be read automatically. When the RDR data is read by the DMAC or DTC, the RDRF flag is automatically cleared to 0.
- [5] When a normal frame is received, the pin retains the high-impedance state at the timing for error signal transmission.



**Figure 12-11 Retransfer Operation in SCI Receive Mode**

- Retransfer operation when SCI is in transmit mode

Figure 12-12 illustrates the retransfer operation when the SCI is in transmit mode.

- [6] If an error signal is sent back from the receiving end after transmission of one frame is completed, the ERS bit in SSR is set to 1. If the RIE bit in SCR is enabled at this time, an ERI

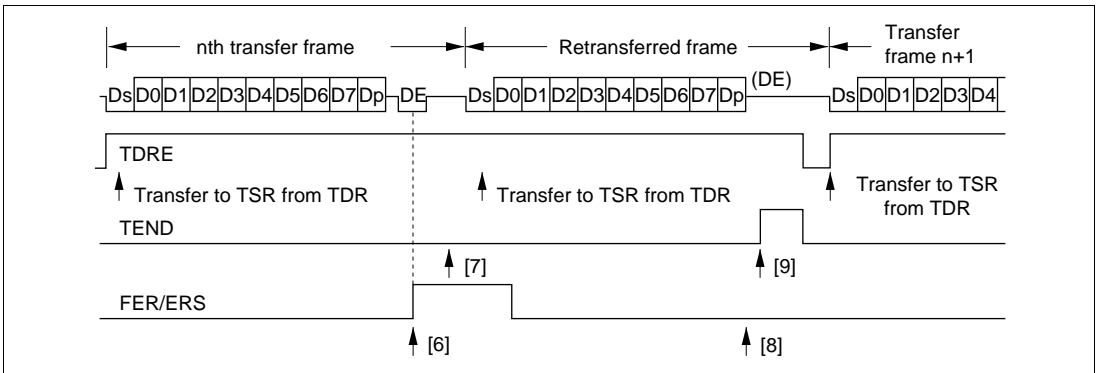
interrupt request is generated. The ERS bit in SSR should be kept cleared to 0 until the next parity bit is sampled.

[7] The TEND bit in SSR is not set for a frame for which an error signal indicating an abnormality is received.

[8] If an error signal is not sent back from the receiving end, the ERS bit in SSR is not set.

[9] If an error signal is not sent back from the receiving end, transmission of one frame, including a retransfer, is judged to have been completed, and the TEND bit in SSR is set to 1. If the TIE bit in SCR is enabled at this time, a TXI interrupt request is generated.

If data transfer by the DMAC or DTC by means of the TXI source is enabled, the next data can be written to TDR automatically. When data is written to TDR by the DMAC or DTC, the TDRE bit is automatically cleared to 0.



**Figure 12-12 Retransfer Operation in SCI Transmit Mode**

# Section 13 A/D Converter (8 Analog Input Channel Version)

## 13.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series incorporate a successive-approximations type 10-bit A/D converter that allows up to eight analog input channels to be selected.

### 13.1.1 Features

A/D converter features are listed below

- 10-bit resolution
- Eight input channels
- Settable analog conversion voltage range
  - Conversion of analog voltages with the reference voltage pin ( $V_{ref}$ ) as the analog reference voltage
- High-speed conversion
  - Minimum conversion time: 6.7  $\mu$ s per channel (at 20 MHz operation)
- Choice of single mode or scan mode
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels
- Four data registers
  - Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three kinds of conversion start
  - Choice of software or timer conversion start trigger (TPU or 8-bit timer), or  $\overline{ADTRG}$  pin
- A/D conversion end interrupt generation
  - A/D conversion end interrupt (ADI) request can be generated at the end of A/D conversion
  - The DMA controller (DMAC)\* or data transfer controller (DTC) can be activated for data transfer by an interrupt

Note: \* Some models do not have an on-chip DMAC; please check the reference manual for the relevant model for confirmation.

- Module stop mode can be set
  - As the initial setting, A/D converter operation is halted. Register access is enabled by exiting module stop mode.

### 13.1.2 Block Diagram

Figure 13-1 shows a block diagram of the A/D converter.

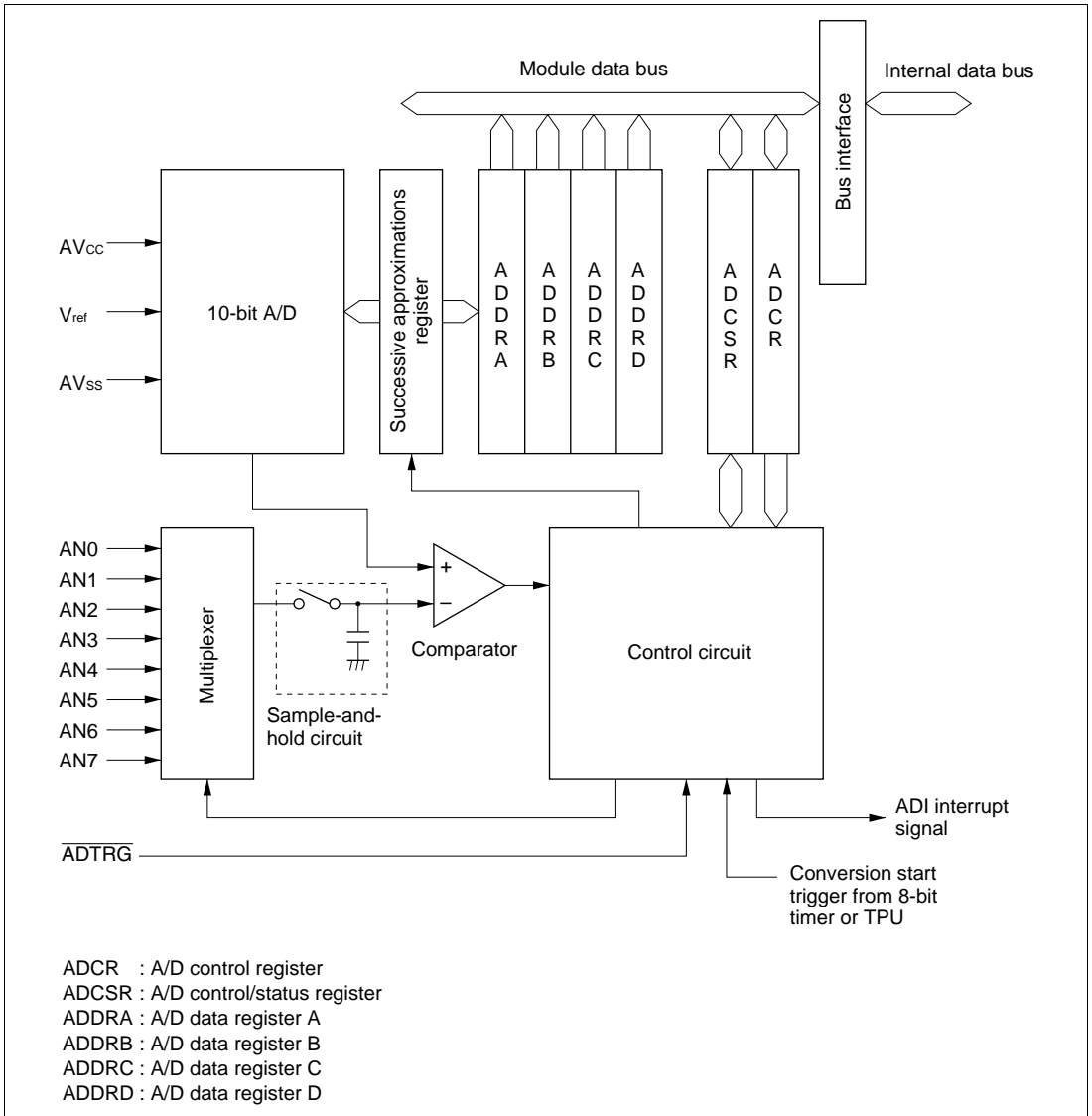


Figure 13-1 Block Diagram of A/D Converter

### 13.1.3 Pin Configuration

Table 13-1 summarizes the input pins used by the A/D converter.

The  $AV_{CC}$  and  $AV_{SS}$  pins are the power supply pins for the analog block in the A/D converter. The  $V_{ref}$  pin is the A/D conversion reference voltage pin.

The eight analog input pins are divided into two groups: group 0 (AN0 to AN3), and group 1 (AN4 to AN7).

**Table 13-1 A/D Converter Pins**

Pin Name	Symbol	I/O	Function
Analog power supply pin	$AV_{CC}$	Input	Analog block power supply
Analog ground pin	$AV_{SS}$	Input	Analog block ground and A/D conversion reference voltage
Reference voltage pin	$V_{ref}$	Input	A/D conversion reference voltage
Analog input pin 0	AN0	Input	Group 0 analog inputs
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin 3	AN3	Input	
Analog input pin 4	AN4	Input	Group 1 analog inputs
Analog input pin 5	AN5	Input	
Analog input pin 6	AN6	Input	
Analog input pin 7	AN7	Input	
A/D external trigger input pin	$\overline{ADTRG}$	Input	External trigger input for starting A/D conversion

### 13.1.4 Register Configuration

Table 13-2 summarizes the registers of the A/D converter.

**Table 13-2 A/D Converter Registers**

<b>Name</b>	<b>Abbreviation</b>	<b>R/W</b>	<b>Initial Value</b>	<b>Address*<sup>1</sup></b>
A/D data register AH	ADDRAH	R	H'00	H'FF90
A/D data register AL	ADDRAL	R	H'00	H'FF91
A/D data register BH	ADDRBH	R	H'00	H'FF92
A/D data register BL	ADDRBL	R	H'00	H'FF93
A/D data register CH	ADDRCH	R	H'00	H'FF94
A/D data register CL	ADDRCL	R	H'00	H'FF95
A/D data register DH	ADDRDH	R	H'00	H'FF96
A/D data register DL	ADDRDL	R	H'00	H'FF97
A/D control/status register	ADCSR	R/(W)* <sup>2</sup>	H'00	H'FF98
A/D control register	ADCR	R/W	H'3F	H'FF99
Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. Lower 16 bits of the address.

2. Bit 7 can only be written with 0 for flag clearing.

## 13.2 Register Descriptions

### 13.2.1 A/D Data Registers A to D (ADDRA to ADDR D)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	—	—	—	—	—	—
Initial value	:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

There are four 16-bit read-only ADDR registers, ADDRA to ADDR D, used to store the results of A/D conversion.

The 10-bit data resulting from A/D conversion is transferred to the ADDR register for the selected channel and stored there. The upper 8 bits of the converted data are transferred to the upper byte (bits 15 to 8) of ADDR, and the lower 2 bits are transferred to the lower byte (bits 7 and 6) and stored. Bits 5 to 0 are always read as 0.

The correspondence between the analog input channels and ADDR registers is shown in table 13-3.

The ADDR registers can always be read by the CPU. The upper byte can be read directly, but for the lower byte, data transfer is performed via a temporary register (TEMP). For details, see section 13.3, Interface to Bus Master.

The ADDR registers are initialized to H'0000 by a reset, and in standby mode or module stop mode.

**Table 13-3 Analog Input Channels and Corresponding ADDR Registers**

Analog Input Channel		A/D Data Register
Group 0	Group 1	
AN0	AN4	ADDRA
AN1	AN5	ADDRB
AN2	AN6	ADDRC
AN3	AN7	ADDRD



### 13.2.2 A/D Control/Status Register (ADCSR)

Bit	:	7	6	5	4	3	2	1	0
		ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* Only 0 can be written to bit 7, to clear this flag.

ADCSR is an 8-bit readable/writable register that controls A/D conversion operations and shows the status of the operation.

ADCSR is initialized to H'00 by a reset, and in hardware standby mode or module stop mode.

**Bit 7—A/D End Flag (ADF):** Status flag that indicates the end of A/D conversion.

Bit 7 ADF	Description	
0	[Clearing conditions]	(Initial value)
	<ul style="list-style-type: none"> <li>When 0 is written to the ADF flag after reading ADF = 1</li> <li>When the DMAC or DTC is activated by an ADI interrupt and ADDR is read</li> </ul>	
1	[Setting conditions]	
	<ul style="list-style-type: none"> <li>Single mode: When A/D conversion ends</li> <li>Scan mode: When A/D conversion ends on all specified channels</li> </ul>	

**Bit 6—A/D Interrupt Enable (ADIE):** Selects enabling or disabling of interrupt (ADI) requests at the end of A/D conversion.

Bit 6 ADIE	Description	
0	A/D conversion end interrupt (ADI) request disabled	(Initial value)
1	A/D conversion end interrupt (ADI) request enabled	

**Bit 5—A/D Start (ADST):** Selects starting or stopping of A/D conversion. Holds a value of 1 during A/D conversion.

The ADST bit can be set to 1 by software, a timer conversion start trigger, or the A/D external trigger input pin ( $\overline{\text{ADTRG}}$ ).

**Bit 5**

ADST	Description
0	• A/D conversion stopped (Initial value)
1	• Single mode: A/D conversion is started. Cleared to 0 automatically when conversion on the specified channel ends • Scan mode: A/D conversion is started. Conversion continues sequentially on the selected channels until ADST is cleared to 0 by software, a reset, or a transition to standby mode or module stop mode

**Bit 4—Scan Mode (SCAN):** Selects single mode or scan mode as the A/D conversion operating mode. See section 13.4, Operation, for details of single mode and scan mode operation. Only set the SCAN bit while conversion is stopped (ADST = 0).

**Bit 4**

SCAN	Description
0	Single mode (Initial value)
1	Scan mode

**Bit 3—Clock Select (CKS):** Used together with the CKS1 bit in ADCR to set the A/D conversion time. Only change the conversion time while conversion is stopped (ADST = 0).

ADCR3 CKS1	Bit 3 CKS	Description
0	0	Conversion time = 530 states (max.)
	1	Conversion time = 68 states (max.)
1	0	Conversion time = 266 states (max.) (Initial value)
	1	Conversion time = 134 states (max.)

**Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0):** These bits are used together with the SCAN bit to select the analog input channels.

Only set the input channel(s) while conversion is stopped (ADST = 0).

Group Selection	Channel Selection		Description	
	CH2	CH1	CH0	Single Mode (SCAN = 0) Scan Mode (SCAN = 1)
0	0	0	AN0 (Initial value)	AN0
		1	AN1	AN0, AN1
	1	0	AN2	AN0 to AN2
		1	AN3	AN0 to AN3
1	0	0	AN4	AN4
		1	AN5	AN4, AN5
	1	0	AN6	AN4 to AN6
		1	AN7	AN4 to AN7

### 13.2.3 A/D Control Register (ADCR)

Bit	:	7	6	5	4	3	2	1	0
		TRGS1	TRGS0	—	—	CKS1	CH3	—	—
Initial value :		0	0	1	1	1	1	1	1
R/W	:	R/W	R/W	—	—	R/W	R/W	—	—

ADCR is an 8-bit readable/writable register that enables or disables external triggering of A/D conversion operations.

ADCR is initialized to H'3F by a reset, and in standby mode or module stop mode.

**Bits 7 and 6—Timer Trigger Select 1 and 0 (TRGS1, TRGS0):** These bits select enabling or disabling of the start of A/D conversion by a trigger signal. Only set bits TRGS1 and TRGS0 while conversion is stopped (ADST = 0).

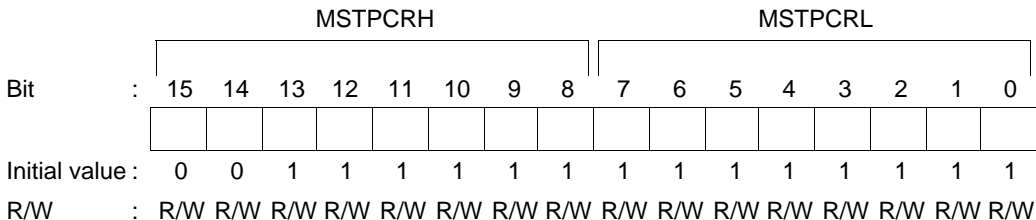
Bit 7 TRGS1	Bit 6 TRGS0	Description
0	0	A/D conversion start by external trigger is disabled (Initial value)
	1	A/D conversion start by external trigger (TPU) is enabled
1	0	A/D conversion start by external trigger (8-bit timer) is enabled
	1	A/D conversion start by external trigger pin ( $\overline{ADTRG}$ ) is enabled

**Bits 5, 4, 1, and 0—Reserved:** Read-only bits, always read as 1.

**Bit 3—Clock Select 1 (CKS1):** Used together with the CKS bit in ADCSR to set the A/D conversion time. See the description of the CKS bit for details.

**Bit 2—Channel Select 3 (CH3):** Reserved. A value of 1 must be written to this bit.

### 13.2.4 Module Stop Control Register (MSTPCR)



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP9 bit in MSTPCR is set to 1, A/D converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 9—Module Stop (MSTP9):** Specifies the A/D converter module stop mode.

Bit 9 MSTP9	Description
0	A/D converter module stop mode cleared
1	A/D converter module stop mode set <span style="float: right;">(Initial value)</span>

### 13.3 Interface to Bus Master

ADDRA to ADDR<sub>D</sub> are 16-bit registers, and the data bus to the bus master is 8 bits wide. Therefore, in accesses by the bus master, the upper byte is accessed directly, but the lower byte is accessed via a temporary register (TEMP).

A data read from ADDR is performed as follows. When the upper byte is read, the upper byte value is transferred to the CPU and the lower byte value is transferred to TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the CPU.

When reading ADDR, always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, incorrect data may be obtained.

Figure 13-2 shows the data flow for ADDR access.

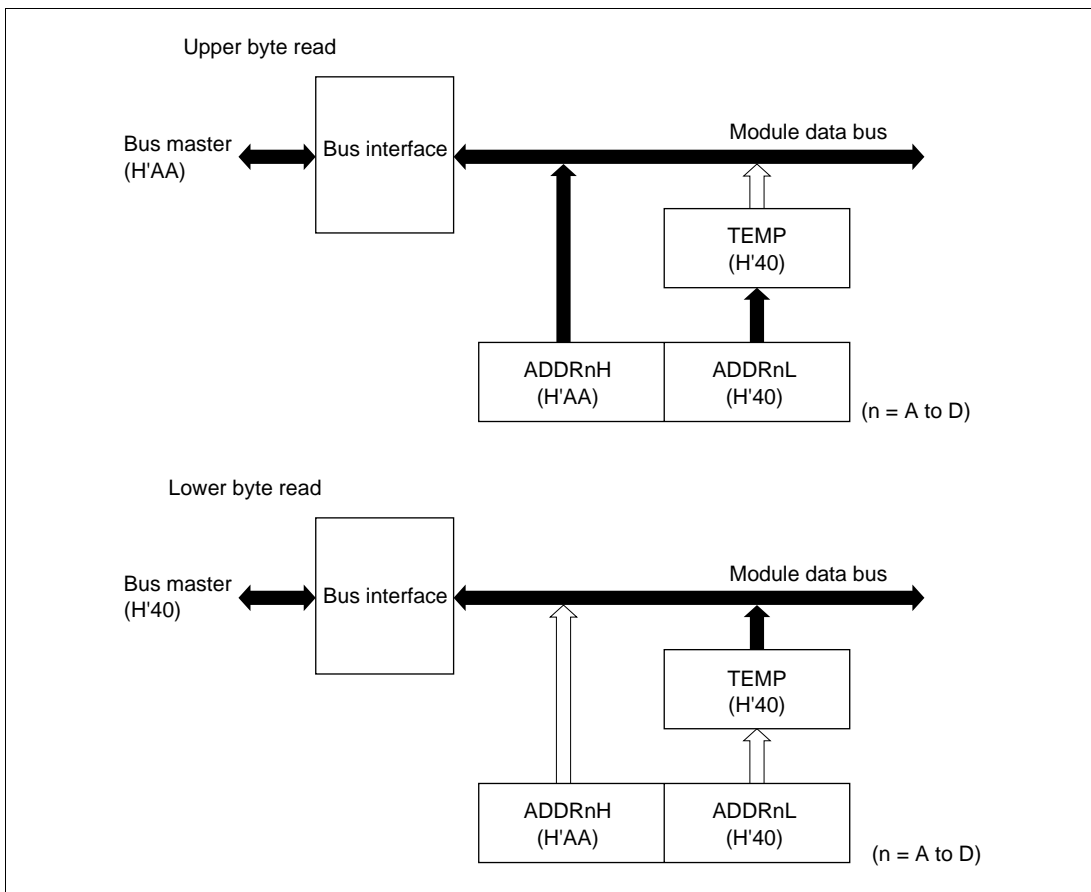


Figure 13-2 ADDR Access Operation (Reading H'AA40)

## 13.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution. It has two operating modes: single mode and scan mode.

### 13.4.1 Single Mode (SCAN = 0)

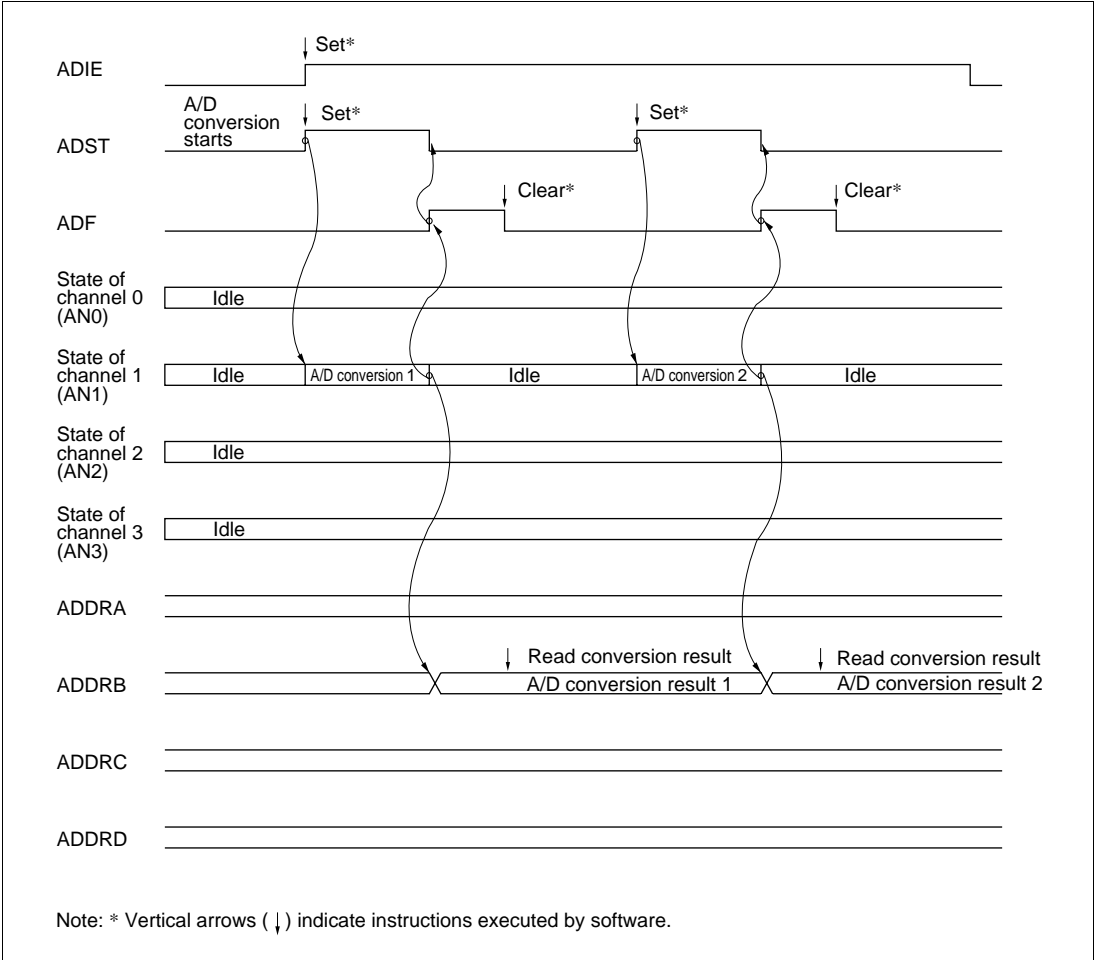
Single mode is selected when A/D conversion is to be performed on a single channel only. A/D conversion is started when the ADST bit is set to 1 by software or by external trigger input. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends.

On completion of conversion, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by writing 0 to it after reading ADCSR.

When the operating mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the operating mode or input channel is changed.

Typical operations when channel 1 (AN1) is selected in single mode are described next. Figure 13-3 shows a timing diagram for this example.

- [1] Single mode is selected (SCAN = 0), input channel AN1 is selected (CH2 = 0, CH1 = 0, CH0 = 1), the A/D interrupt is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).
- [2] When A/D conversion is completed, the result is transferred to ADDR0. At the same time the ADF flag is set to 1, the ADST bit is cleared to 0, and the A/D converter becomes idle.
- [3] Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.
- [4] The A/D interrupt handling routine starts.
- [5] The routine reads ADCSR, then writes 0 to the ADF flag.
- [6] The routine reads and processes the conversion result (ADDR0).
- [7] Execution of the A/D interrupt handling routine ends. After that, if the ADST bit is set to 1, A/D conversion starts again and steps [2] to [7] are repeated.



**Figure 13-3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)**

### 13.4.2 Scan Mode (SCAN = 1)

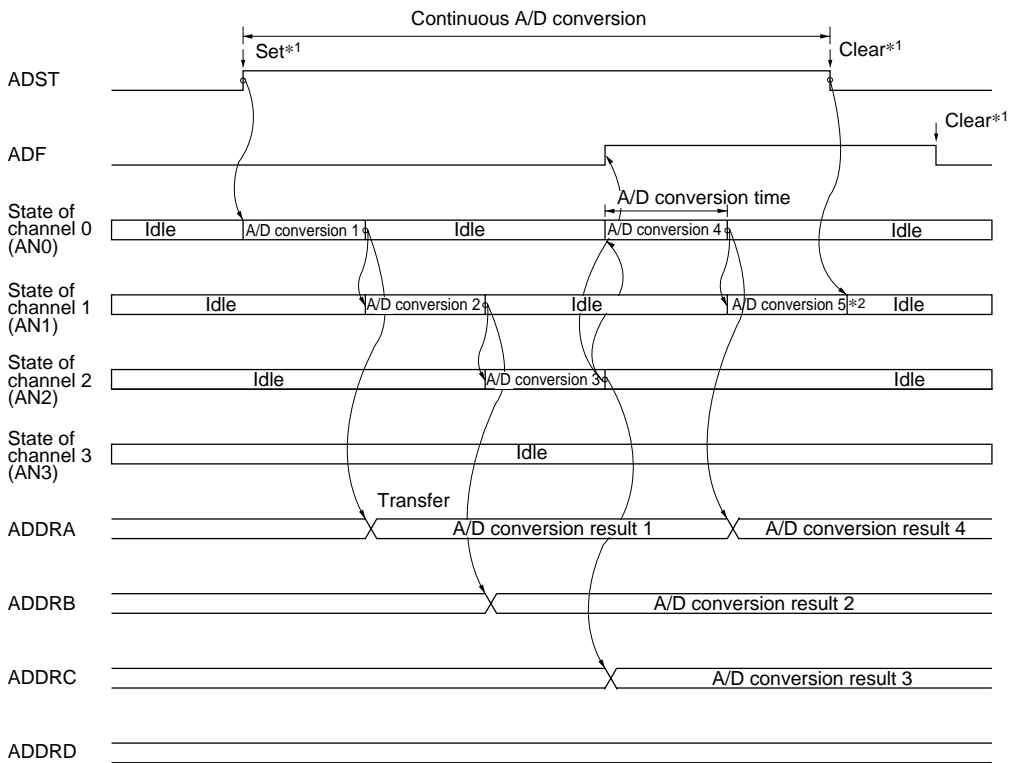
Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit is set to 1 by software, or by timer or external trigger input, A/D conversion starts on the first channel in the group (AN0). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1) starts immediately. A/D conversion continues cyclically on the selected channels until the ADST bit is cleared to 0. The conversion results are transferred for storage into the ADDR registers corresponding to the channels.

When the operating mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the operating mode or input channel is changed.

Typical operations when three channels (AN0 to AN2) are selected in scan mode are described next. Figure 13-4 shows a timing diagram for this example.

- [1] Scan mode is selected (SCAN = 1), scan group 0 is selected (CH2 = 0), analog input channels AN0 to AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1)
- [2] When A/D conversion of the first channel (AN0) is completed, the result is transferred to ADDRA. Next, conversion of the second channel (AN1) starts automatically.
- [3] Conversion proceeds in the same way through the third channel (AN2).
- [4] When conversion of all the selected channels (AN0 to AN2) is completed, the ADF flag is set to 1 and conversion of the first channel (AN0) starts again. If the ADIE bit is set to 1 at this time, an ADI interrupt is requested after A/D conversion ends.
- [5] Steps [2] to [4] are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).





- Notes: 1. Vertical arrows ( $\downarrow$ ) indicate instructions executed by software.  
 2. Data currently being converted is ignored.

**Figure 13-4 Example of A/D Converter Operation  
 (Scan Mode, Channels AN0 to AN2 Selected)**

### 13.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input at a time  $t_D$  after the ADST bit is set to 1, then starts conversion. Figure 13-5 shows the A/D conversion timing. Table 13-4 indicates the A/D conversion time.

As indicated in figure 13-5, the A/D conversion time includes  $t_D$  and the input sampling time. The length of  $t_D$  varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 13-4.

In scan mode, the values given in table 13-4 apply to the first conversion time. In the second and subsequent conversions the conversion time is as shown in table 13-5.

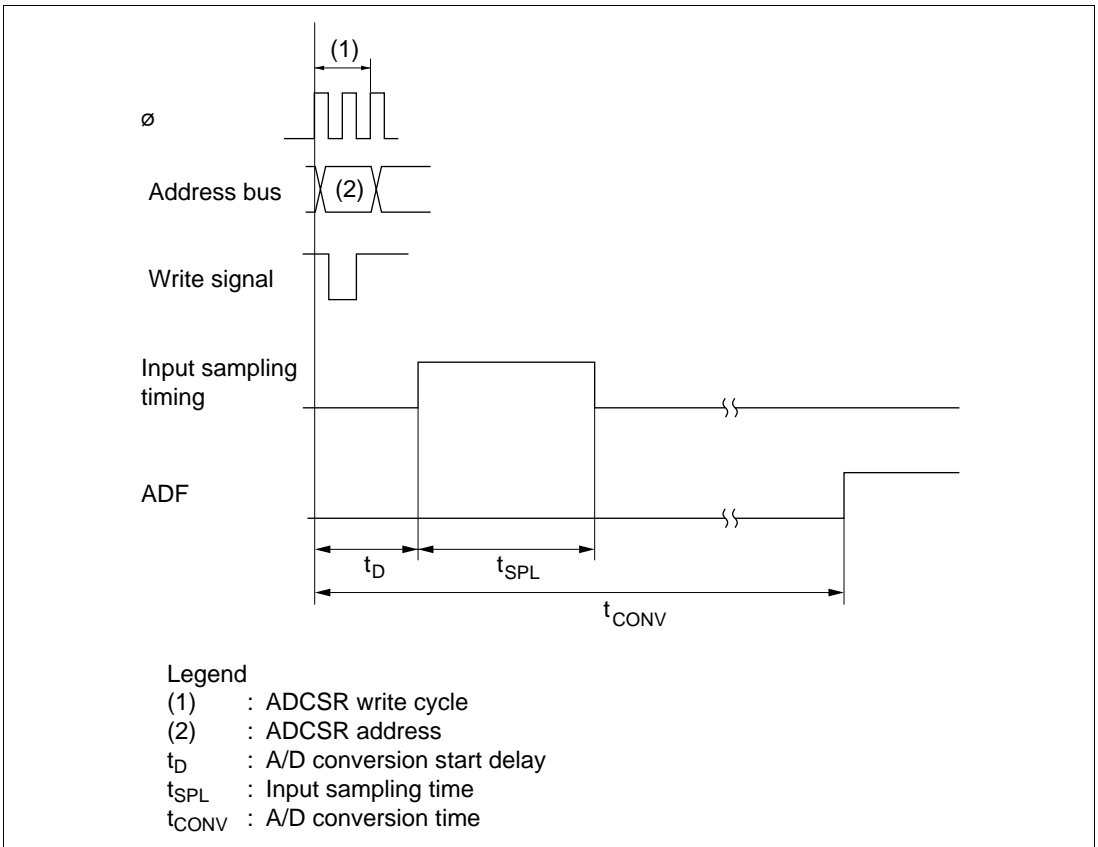


Figure 13-5 A/D Conversion Timing

**Table 13-4 A/D Conversion Time (Single Mode)**

Item	Symbol	CKS1 = 0						CKS1 = 1					
		CKS = 0			CKS = 1			CKS = 0			CKS = 1		
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max
A/D conversion start delay	$t_D$	18	—	33	4	—	5	10	—	17	6	—	9
Input sampling time	$t_{SPL}$	—	127	—	—	15	—	—	63	—	—	31	—
A/D conversion time	$t_{CONV}$	55	—	530	67	—	68	259	—	266	131	—	134

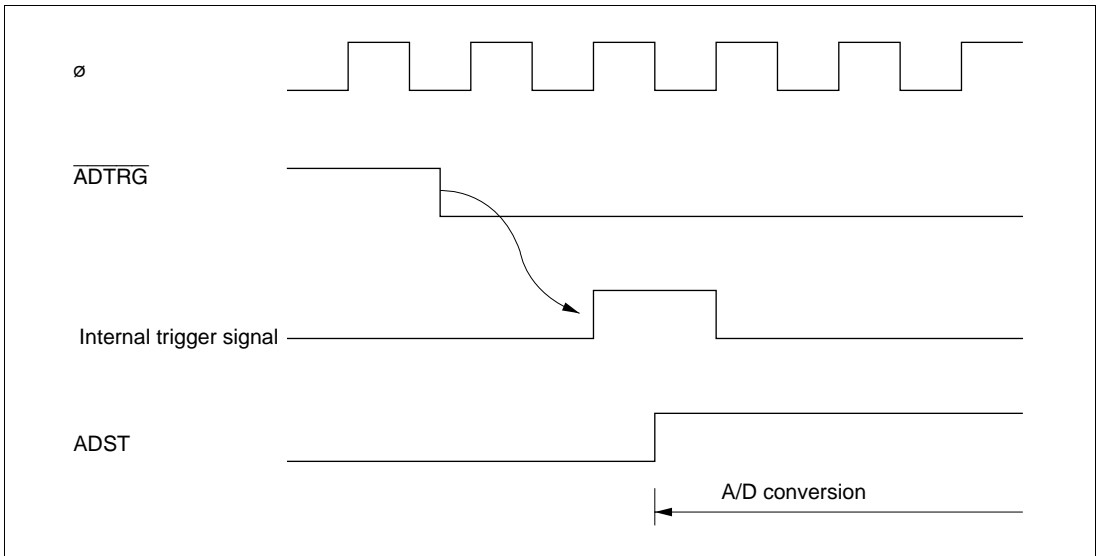
Note: Values in the table are the number of states.

**Table 13-5 A/D Conversion Time (Scan Mode)**

CKS1	CKS	Conversion Time (States)
0	0	512 (Fixed)
	1	64 (Fixed)
1	0	256 (Fixed)
	1	128 (Fixed)

### 13.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to B'11 in ADCR, external trigger input is enabled at the  $\overline{ADTRG}$  pin. A falling edge at the  $\overline{ADTRG}$  pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 13-6 shows the timing.



**Figure 13-6 External Trigger Input Timing**

## 13.5 Interrupts

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. ADI interrupt requests can be enabled or disabled by means of the ADIE bit in ADCSR.

The DTC or DMAC can be activated by an ADI interrupt. Having the converted data read by the DTC or DMAC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

The A/D converter interrupt source is shown in table 13-6.

**Table 13-6 A/D Converter Interrupt Source**

Interrupt Source	Description	DTC Activation	DMAC Activation
ADI	Interrupt due to end of conversion	Possible	Possible

## 13.6 Usage Notes

The following points should be noted when using the A/D converter.

### Setting Range of Analog Power Supply and Other Pins

#### 1. Analog input voltage range

The voltage applied to analog input pins ANn during A/D conversion should be in the range  $AV_{SS} \leq ANn \leq V_{ref}$ .

#### 2. Relation between $AV_{CC}$ , $AV_{SS}$ and $V_{CC}$ , $V_{SS}$

As the relationship between  $AV_{CC}$ ,  $AV_{SS}$  and  $V_{CC}$ ,  $V_{SS}$ , set  $AV_{SS} = V_{SS}$ . If the A/D converter is not used, the  $AV_{CC}$  and  $AV_{SS}$  pins must not be left open.

#### 3. $V_{ref}$ input range

The analog reference voltage input at the  $V_{ref}$  pin should be set in the range  $V_{ref} \leq AV_{CC}$ .

If conditions 1, 2, and 3 above are not met, the reliability of the device may be adversely affected.

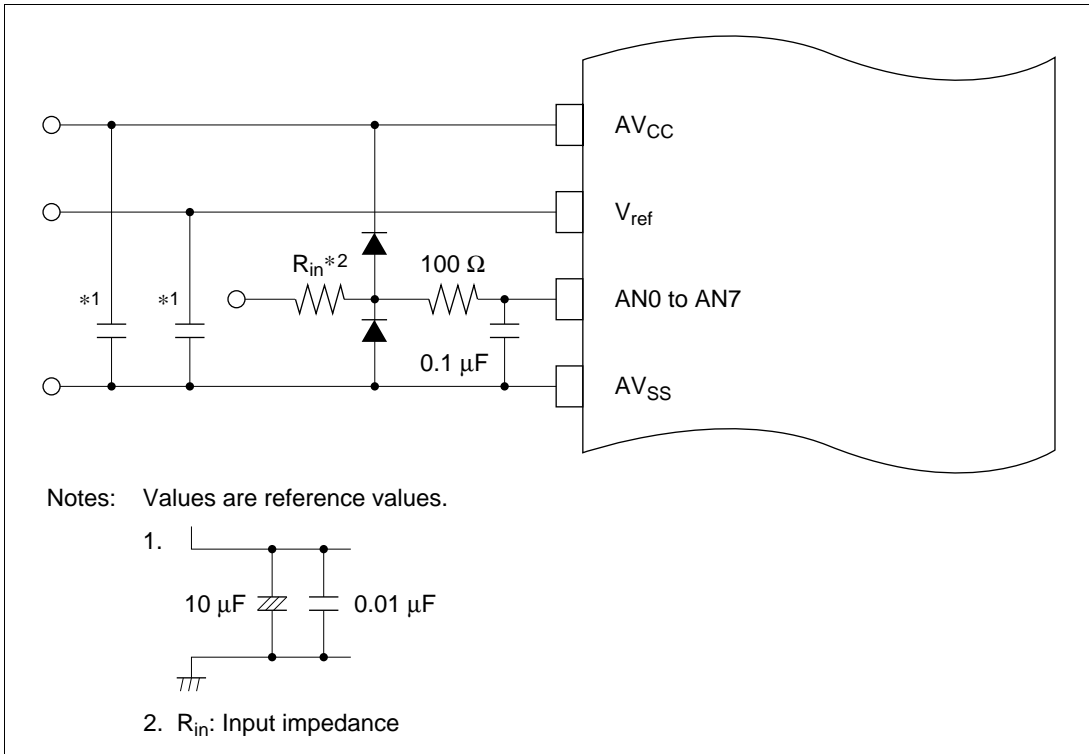
**Notes on Board Design:** In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values.

Also, digital circuitry must be isolated from the analog input signals (AN0 to AN7), analog reference power supply ( $V_{ref}$ ), and analog power supply ( $AV_{CC}$ ) by the analog ground ( $AV_{SS}$ ). Also, the analog ground ( $AV_{SS}$ ) should be connected at one point to a stable digital ground ( $V_{SS}$ ) on the board.

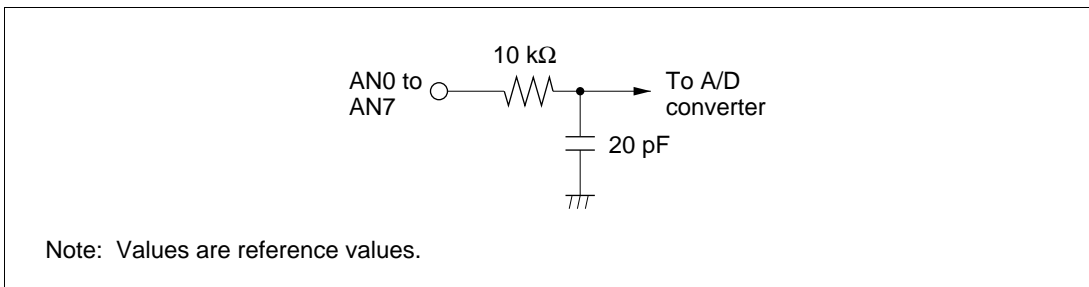
**Notes on Noise Countermeasures:** A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN7) and analog reference power supply ( $V_{ref}$ ) should be connected between  $AV_{CC}$  and  $AV_{SS}$  as shown in figure 13-7.

Also, the bypass capacitors connected to  $AV_{CC}$  and  $V_{ref}$  and the filter capacitor connected to AN0 to AN7 must be connected to  $AV_{SS}$ .

If a filter capacitor is connected as shown in figure 13-7, the input currents at the analog input pins (AN0 to AN7) are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance ( $R_{in}$ ), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.



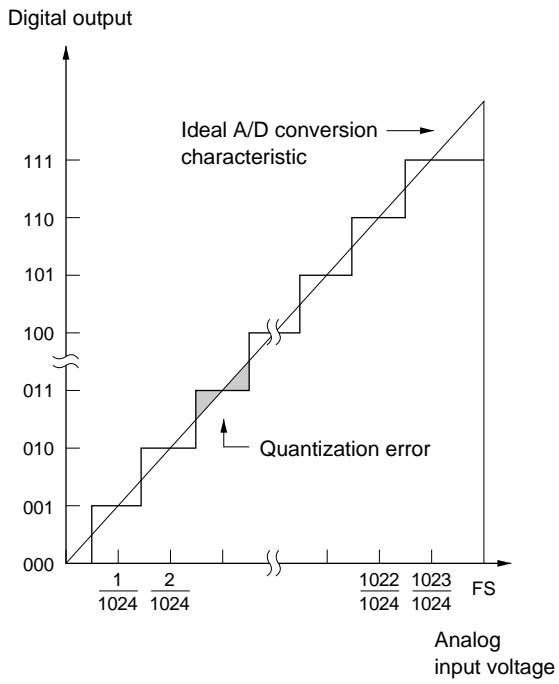
**Figure 13-7 Example of Analog Input Protection Circuit**



**Figure 13-8 Analog Input Pin Equivalent Circuit**

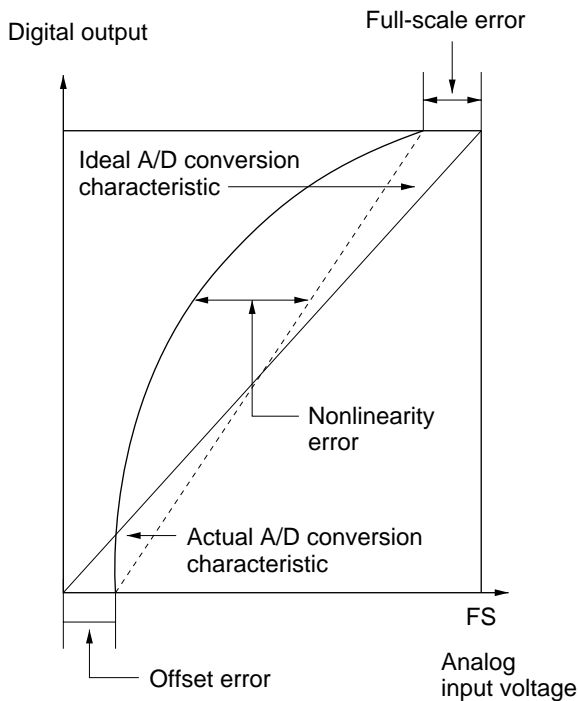
**A/D Conversion Precision Definitions:** H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series  
A/D conversion precision definitions are given below.

- Resolution  
The number of A/D converter digital output codes
- Offset error  
The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'0000000000 to B'0000000001 (see figure 13-10).
- Full-scale error  
The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 to B'1111111111 (see figure 13-10).
- Quantization error  
The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 13-9).
- Nonlinearity error  
The error with respect to the ideal A/D conversion characteristic between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error.
- Absolute precision  
The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.



**Figure 13-9 A/D Conversion Precision Definitions (1)**





**Figure 13-10 A/D Conversion Precision Definitions (2)**

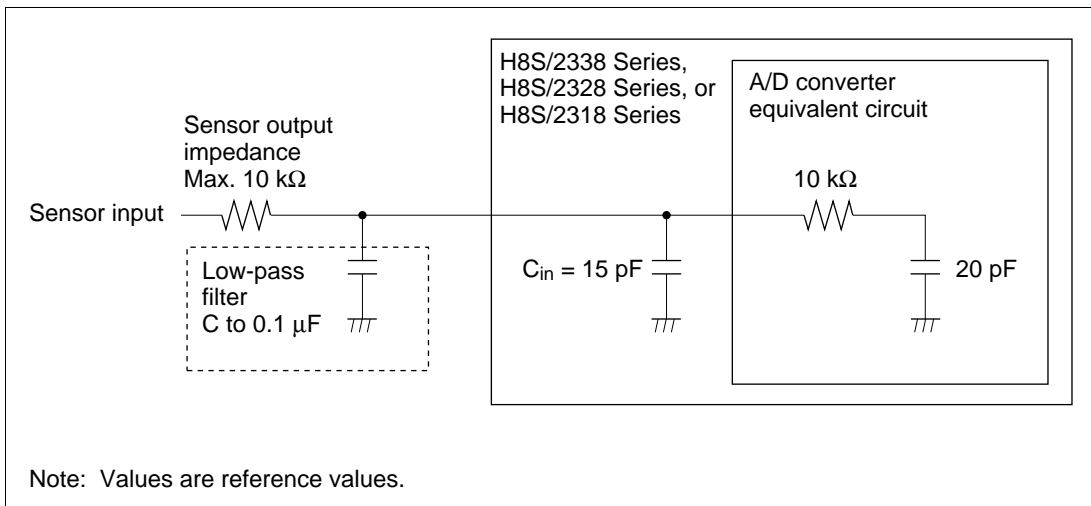
**Permissible Signal Source Impedance:** H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series analog input is designed so that conversion precision is guaranteed for an input signal for which the signal source impedance is 10 k $\Omega$  or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 10 k $\Omega$ , charging may be insufficient and it may not be possible to guarantee the A/D conversion precision.

If a large capacitance is provided externally, the input load will essentially comprise only the internal input resistance of 10 k $\Omega$ , and the signal source impedance is ignored. However, since a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (e.g., 5 mV/ $\mu$ s or greater).

When converting a high-speed analog signal, a low-impedance buffer should be inserted.

**Influences on Absolute Precision:** Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute precision. Be sure to make the connection to an electrically stable GND such as  $AV_{SS}$ .

Care is also required to insure that filter circuits do not communicate with digital signals on the mounting board, so acting as antennas.



**Figure 13-11 Example of Analog Input Circuit**

# Section 14 A/D Converter (12 Analog Input Channel Version)

## 14.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series incorporate a successive-approximations type 10-bit A/D converter that allows up to twelve analog input channels to be selected.

### 14.1.1 Features

A/D converter features are listed below

- 10-bit resolution
- Twelve input channels
- Settable analog conversion voltage range
  - Conversion of analog voltages with the reference voltage pin ( $V_{ref}$ ) as the analog reference voltage
- High-speed conversion
  - Minimum conversion time: 6.7  $\mu$ s per channel (at 20 MHz operation)
- Choice of single mode or scan mode
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels
- Four data registers
  - Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three kinds of conversion start
  - Choice of software or timer conversion start trigger (TPU or 8-bit timer), or  $\overline{ADTRG}$  pin
- A/D conversion end interrupt generation
  - A/D conversion end interrupt (ADI) request can be generated at the end of A/D conversion
  - The DMA controller (DMAC)\* or data transfer controller (DTC) can be activated for data transfer by an interrupt

Note: \* Some models do not have an on-chip DMAC; please check the reference manual for the relevant model for confirmation.

- Module stop mode can be set
  - As the initial setting, A/D converter operation is halted. Register access is enabled by exiting module stop mode.

### 14.1.2 Block Diagram

Figure 14-1 shows a block diagram of the A/D converter.

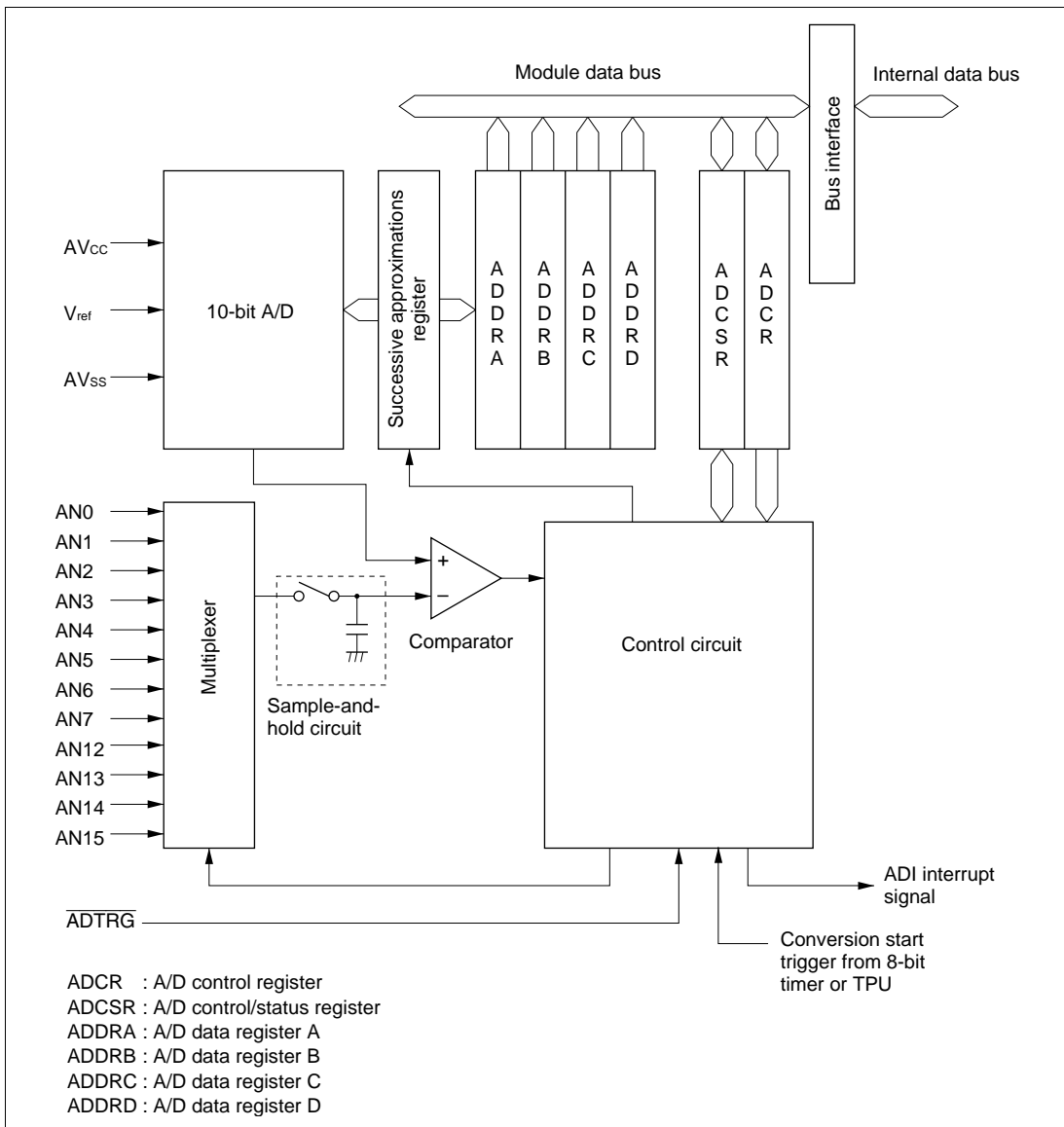


Figure 14-1 Block Diagram of A/D Converter

### 14.1.3 Pin Configuration

Table 14-1 summarizes the input pins used by the A/D converter.

The  $AV_{CC}$  and  $AV_{SS}$  pins are the power supply pins for the analog block in the A/D converter. The  $V_{ref}$  pin is the A/D conversion reference voltage pin.

The twelve analog input pins are divided into two channel sets and two groups: channel set 0 (AN0 to AN7), channel set 1 (AN12 to AN15), group 0 (AN0 to AN3), and group 1 (AN4 to AN7, AN12 to AN15).

**Table 14-1 A/D Converter Pins**

Pin Name	Symbol	I/O	Function
Analog power supply pin	$AV_{CC}$	Input	Analog block power supply
Analog ground pin	$AV_{SS}$	Input	Analog block ground and A/D conversion reference voltage
Reference voltage pin	$V_{ref}$	Input	A/D conversion reference voltage
Analog input pin 0	AN0	Input	Channel set 0 (CH3 = 1), group 0 analog inputs
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin 3	AN3	Input	
Analog input pin 4	AN4	Input	Channel set 0 (CH3 = 1), group 1 analog inputs
Analog input pin 5	AN5	Input	
Analog input pin 6	AN6	Input	
Analog input pin 7	AN7	Input	
Analog input pin 12	AN12	Input	Channel set 1 (CH3 = 0), group 1 analog inputs
Analog input pin 13	AN13	Input	
Analog input pin 14	AN14	Input	
Analog input pin 15	AN15	Input	
A/D external trigger input pin	$\overline{ADTRG}$	Input	External trigger input for starting A/D conversion

## 14.1.4 Register Configuration

Table 14-2 summarizes the registers of the A/D converter.

**Table 14-2 A/D Converter Registers**

<b>Name</b>	<b>Abbreviation</b>	<b>R/W</b>	<b>Initial Value</b>	<b>Address*<sup>1</sup></b>
A/D data register AH	ADDRAH	R	H'00	H'FF90
A/D data register AL	ADDRAL	R	H'00	H'FF91
A/D data register BH	ADDRBH	R	H'00	H'FF92
A/D data register BL	ADDRBL	R	H'00	H'FF93
A/D data register CH	ADDRCH	R	H'00	H'FF94
A/D data register CL	ADDRCL	R	H'00	H'FF95
A/D data register DH	ADDRDH	R	H'00	H'FF96
A/D data register DL	ADDRDL	R	H'00	H'FF97
A/D control/status register	ADCSR	R/(W)* <sup>2</sup>	H'00	H'FF98
A/D control register	ADCR	R/W	H'3F	H'FF99
Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Notes: 1. Lower 16 bits of the address.

2. Bit 7 can only be written with 0 for flag clearing.

## 14.2 Register Descriptions

### 14.2.1 A/D Data Registers A to D (ADDRA to ADDR D)

Bit	:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	—	—	—	—	—	—
Initial value	:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

There are four 16-bit read-only ADDR registers, ADDRA to ADDR D, used to store the results of A/D conversion.

The 10-bit data resulting from A/D conversion is transferred to the ADDR register for the selected channel and stored there. The upper 8 bits of the converted data are transferred to the upper byte (bits 15 to 8) of ADDR, and the lower 2 bits are transferred to the lower byte (bits 7 and 6) and stored. Bits 5 to 0 are always read as 0.

The correspondence between the analog input channels and ADDR registers is shown in table 14-3.

The ADDR registers can always be read by the CPU. The upper byte can be read directly, but for the lower byte, data transfer is performed via a temporary register (TEMP). For details, see section 14.3, Interface to Bus Master.

The ADDR registers are initialized to H'0000 by a reset, and in standby mode or module stop mode.

**Table 14-3 Analog Input Channels and Corresponding ADDR Registers**

Analog Input Channel				
Channel Set 0 (CH3 = 1)		Channel Set 1 (CH3 = 0)		A/D Data Register
Group 0 (CH2 = 0)	Group 1 (CH2 = 1)	Group 0 (CH2 = 0)	Group 1 (CH2 = 1)	
AN0	AN4	Setting prohibited	AN12	ADDRA
AN1	AN5	Setting prohibited	AN13	ADDRB
AN2	AN6	Setting prohibited	AN14	ADDRC
AN3	AN7	Setting prohibited	AN15	ADDRD

## 14.2.2 A/D Control/Status Register (ADCSR)

Bit	:	7	6	5	4	3	2	1	0
		ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* Only 0 can be written to bit 7, to clear this flag.

ADCSR is an 8-bit readable/writable register that controls A/D conversion operations and shows the status of the operation.

ADCSR is initialized to H'00 by a reset, and in hardware standby mode or module stop mode.

**Bit 7—A/D End Flag (ADF):** Status flag that indicates the end of A/D conversion.

Bit 7 ADF	Description
0	[Clearing conditions] (Initial value) <ul style="list-style-type: none"> <li>When 0 is written to the ADF flag after reading ADF = 1</li> <li>When the DMAC or DTC is activated by an ADI interrupt and ADDR is read</li> </ul>
1	[Setting conditions] <ul style="list-style-type: none"> <li>Single mode: When A/D conversion ends</li> <li>Scan mode: When A/D conversion ends on all specified channels</li> </ul>

**Bit 6—A/D Interrupt Enable (ADIE):** Selects enabling or disabling of interrupt (ADI) requests at the end of A/D conversion.

Bit 6 ADIE	Description
0	A/D conversion end interrupt (ADI) request disabled (Initial value)
1	A/D conversion end interrupt (ADI) request enabled



**Bit 5—A/D Start (ADST):** Selects starting or stopping of A/D conversion. Holds a value of 1 during A/D conversion.

The ADST bit can be set to 1 by software, a timer conversion start trigger, or the A/D external trigger input pin ( $\overline{\text{ADTRG}}$ ).

**Bit 5**

ADST	Description
0	• A/D conversion stopped (Initial value)
1	<ul style="list-style-type: none"> <li>• Single mode: A/D conversion is started. Cleared to 0 automatically when conversion on the specified channel ends</li> <li>• Scan mode: A/D conversion is started. Conversion continues sequentially on the selected channels until ADST is cleared to 0 by software, a reset, or a transition to standby mode or module stop mode</li> </ul>

**Bit 4—Scan Mode (SCAN):** Selects single mode or scan mode as the A/D conversion operating mode. See section 14.4, Operation, for details of single mode and scan mode operation. Only set the SCAN bit while conversion is stopped (ADST = 0).

**Bit 4**

SCAN	Description
0	Single mode (Initial value)
1	Scan mode

**Bit 3—Clock Select (CKS):** Used together with the CKS1 bit in ADCR to set the A/D conversion time. Only change the conversion time while conversion is stopped (ADST = 0).

**ADCR Bit 3 Bit 3**

CKS1	CKS	Description
0	0	Conversion time = 530 states (max.)
	1	Conversion time = 68 states (max.)
1	0	Conversion time = 266 states (max.) (Initial value)
	1	Conversion time = 134 states (max.)

**Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0):** These bits are used together with the SCAN bit to select the analog input channels.

Only set the input channel(s) while conversion is stopped (ADST = 0).

Channel Selection				Description	
CH3	CH2	CH1	CH0	Single Mode (SCAN = 0)	Scan Mode (SCAN = 1)
0	0	0	0	Setting prohibited	Setting prohibited
			1		
		1	0		
			1		
			1		
1	0	0	0	AN12	AN12
			1	AN13	AN12, AN13
		1	0	AN14	AN12 to AN14
			1	AN15	AN12 to AN15
			1	AN15	AN12 to AN15
1	0	0	0	AN0 (Initial value)	AN0
			1	AN1	AN0, AN1
		1	0	AN2	AN0 to AN2
			1	AN3	AN0 to AN3
	1	0	0	AN4	AN4
			1	AN5	AN4, AN5
		1	0	AN6	AN4 to AN6
		1	AN7	AN4 to AN7	

### 14.2.3 A/D Control Register (ADCR)

Bit	:	7	6	5	4	3	2	1	0
		TRGS1	TRGS0	—	—	CKS1	CH3	—	—
Initial value :		0	0	1	1	1	1	1	1
R/W	:	R/W	R/W	—	—	R/W	R/W	—	—

ADCR is an 8-bit readable/writable register that enables or disables external triggering of A/D conversion operations.

ADCR is initialized to H'3F by a reset, and in standby mode or module stop mode.

**Bits 7 and 6—Timer Trigger Select 1 and 0 (TRGS1, TRGS0):** These bits select enabling or disabling of the start of A/D conversion by a trigger signal. Only set bits TRGS1 and TRGS0 while conversion is stopped (ADST = 0).

Bit 7 TRGS1	Bit 6 TRGS0	Description
0	0	A/D conversion start by external trigger is disabled (Initial value)
	1	A/D conversion start by external trigger (TPU) is enabled
1	0	A/D conversion start by external trigger (8-bit timer) is enabled
	1	A/D conversion start by external trigger pin ( $\overline{\text{ADTRG}}$ ) is enabled

**Bits 5, 4, 1, and 0—Reserved:** Read-only bits, always read as 1.

**Bit 3—Clock Select 1 (CKS1):** Used together with the CKS bit in ADCSR to set the A/D conversion time. See the description of the CKS bit for details.

**Bit 2—Channel Select 3 (CH3):** Used together with bits CH2, CH1, and CH0 in ADCSR to select the analog input channel(s). See the description of bits CH2, CH1, and CH0 for details.

#### 14.2.4 Module Stop Control Register (MSTPCR)

MSTPCRH								MSTPCRL								
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Initial value :	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP9 bit in MSTPCR is set to 1, A/D converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 9—Module Stop (MSTP9):** Specifies the A/D converter module stop mode.

Bit 9 MSTP9	Description
0	A/D converter module stop mode cleared
1	A/D converter module stop mode set (Initial value)

## 14.3 Interface to Bus Master

ADDRA to ADDR<sub>D</sub> are 16-bit registers, and the data bus to the bus master is 8 bits wide. Therefore, in accesses by the bus master, the upper byte is accessed directly, but the lower byte is accessed via a temporary register (TEMP).

A data read from ADDR is performed as follows. When the upper byte is read, the upper byte value is transferred to the CPU and the lower byte value is transferred to TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the CPU.

When reading ADDR, always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, incorrect data may be obtained.

Figure 14-2 shows the data flow for ADDR access.

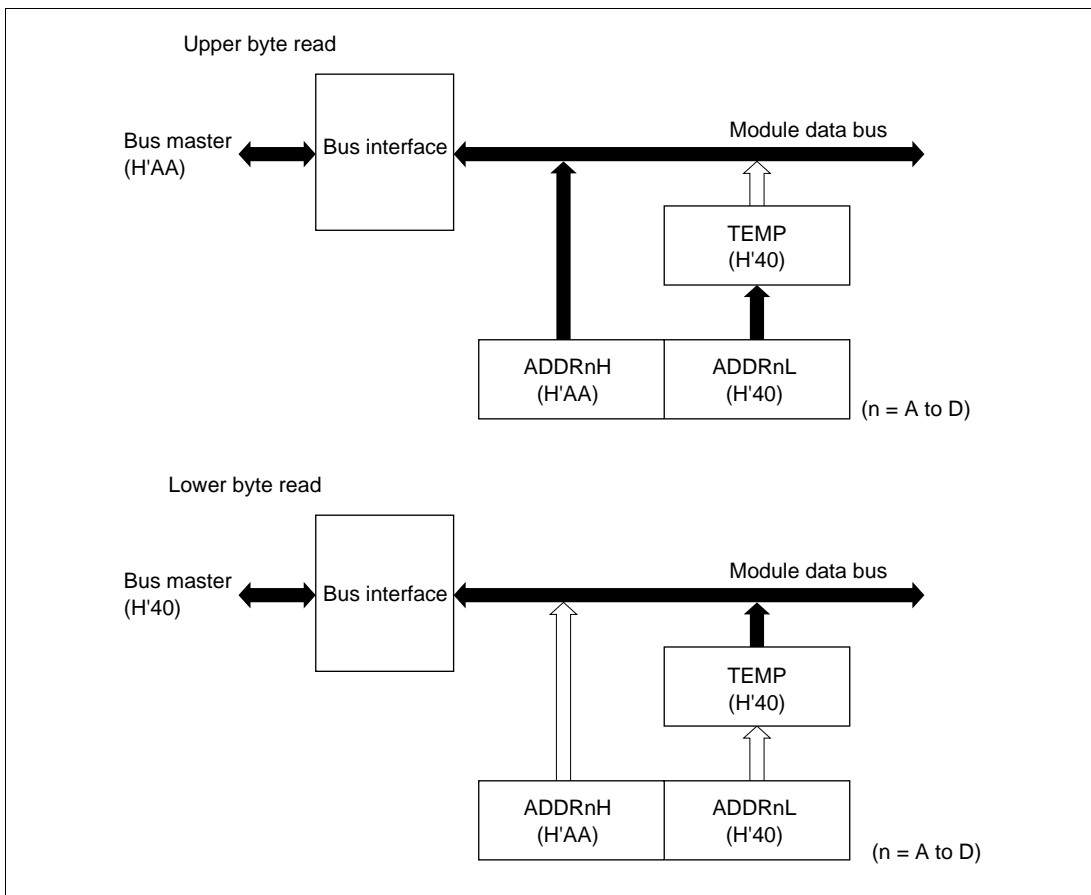


Figure 14-2 ADDR Access Operation (Reading H'AA40)

## 14.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution. It has two operating modes: single mode and scan mode.

### 14.4.1 Single Mode (SCAN = 0)

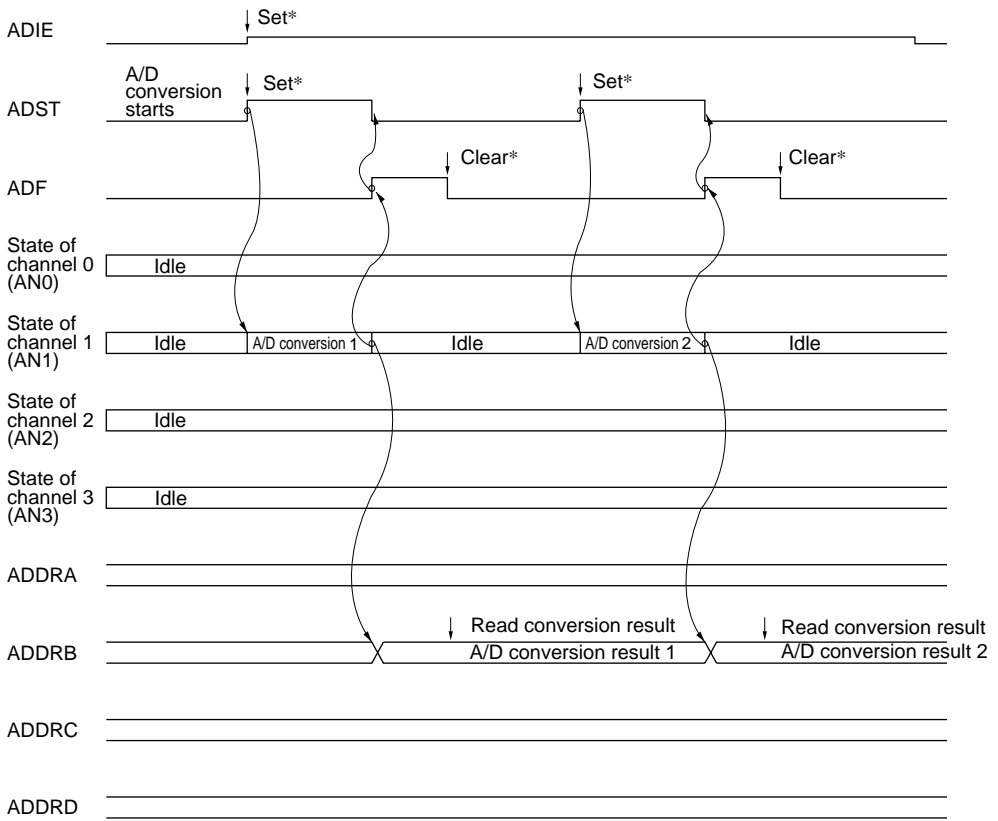
Single mode is selected when A/D conversion is to be performed on a single channel only. A/D conversion is started when the ADST bit is set to 1 by software or by external trigger input. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends.

On completion of conversion, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by writing 0 to it after reading ADCSR.

When the operating mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the operating mode or input channel is changed.

Typical operations when channel 1 (AN1) is selected in single mode are described next. Figure 14-3 shows a timing diagram for this example.

- [1] Single mode is selected (SCAN = 0), input channel AN1 is selected (CH3 = 1, CH2 = 0, CH1 = 0, CH0 = 1), the A/D interrupt is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).
- [2] When A/D conversion is completed, the result is transferred to ADDR0. At the same time the ADF flag is set to 1, the ADST bit is cleared to 0, and the A/D converter becomes idle.
- [3] Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.
- [4] The A/D interrupt handling routine starts.
- [5] The routine reads ADCSR, then writes 0 to the ADF flag.
- [6] The routine reads and processes the conversion result (ADDR0).
- [7] Execution of the A/D interrupt handling routine ends. After that, if the ADST bit is set to 1, A/D conversion starts again and steps [2] to [7] are repeated.



Note: \* Vertical arrows (↓) indicate instructions executed by software.

**Figure 14-3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)**

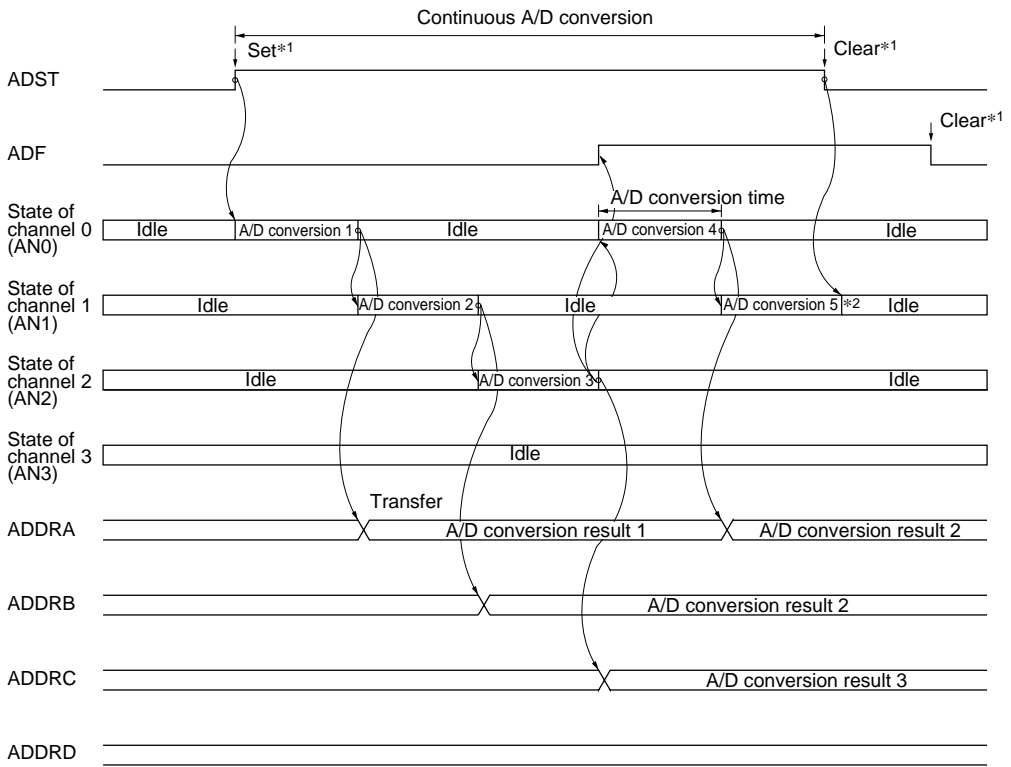
#### 14.4.2 Scan Mode (SCAN = 1)

Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit is set to 1 by software, or by timer or external trigger input, A/D conversion starts on the first channel in the group (AN0). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1) starts immediately. A/D conversion continues cyclically on the selected channels until the ADST bit is cleared to 0. The conversion results are transferred for storage into the ADDR registers corresponding to the channels.

When the operating mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the operating mode or input channel is changed.

Typical operations when three channels (AN0 to AN2) are selected in scan mode are described next. Figure 14-4 shows a timing diagram for this example.

- [1] Scan mode is selected (SCAN = 1), channel set 0 is selected (CH3 = 0), scan group 0 is selected (CH2 = 0), analog input channels AN0 to AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1)
- [2] When A/D conversion of the first channel (AN0) is completed, the result is transferred to ADDR0. Next, conversion of the second channel (AN1) starts automatically.
- [3] Conversion proceeds in the same way through the third channel (AN2).
- [4] When conversion of all the selected channels (AN0 to AN2) is completed, the ADF flag is set to 1 and conversion of the first channel (AN0) starts again. If the ADIE bit is set to 1 at this time, an ADI interrupt is requested after A/D conversion ends.
- [5] Steps [2] to [4] are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).



- Notes: 1. Vertical arrows ( $\downarrow$ ) indicate instructions executed by software.  
 2. Data currently being converted is ignored.

**Figure 14-4 Example of A/D Converter Operation  
 (Scan Mode, Channels AN0 to AN2 Selected)**



### 14.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input at a time  $t_D$  after the ADST bit is set to 1, then starts conversion. Figure 14-5 shows the A/D conversion timing. Table 14-4 indicates the A/D conversion time.

As indicated in figure 14-5, the A/D conversion time includes  $t_D$  and the input sampling time. The length of  $t_D$  varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 14-4.

In scan mode, the values given in table 14-4 apply to the first conversion time. In the second and subsequent conversions the conversion time is as shown in table 14-5.

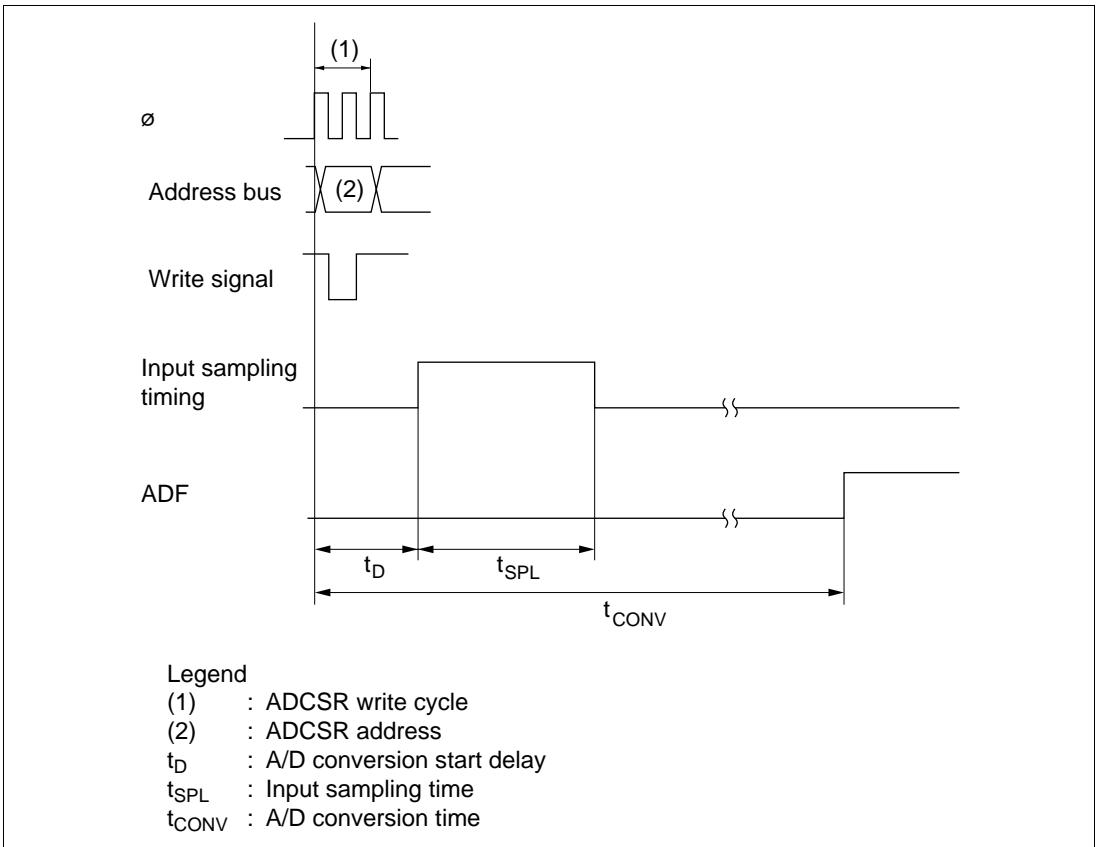


Figure 14-5 A/D Conversion Timing

**Table 14-4 A/D Conversion Time (Single Mode)**

Item	Symbol	CKS1 = 0						CKS1 = 1					
		CKS = 0			CKS = 1			CKS = 0			CKS = 1		
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max
A/D conversion start delay	$t_D$	18	—	33	4	—	5	10	—	17	6	—	9
Input sampling time	$t_{SPL}$	—	127	—	—	15	—	—	63	—	—	31	—
A/D conversion time	$t_{CONV}$	55	—	530	67	—	68	259	—	266	131	—	134

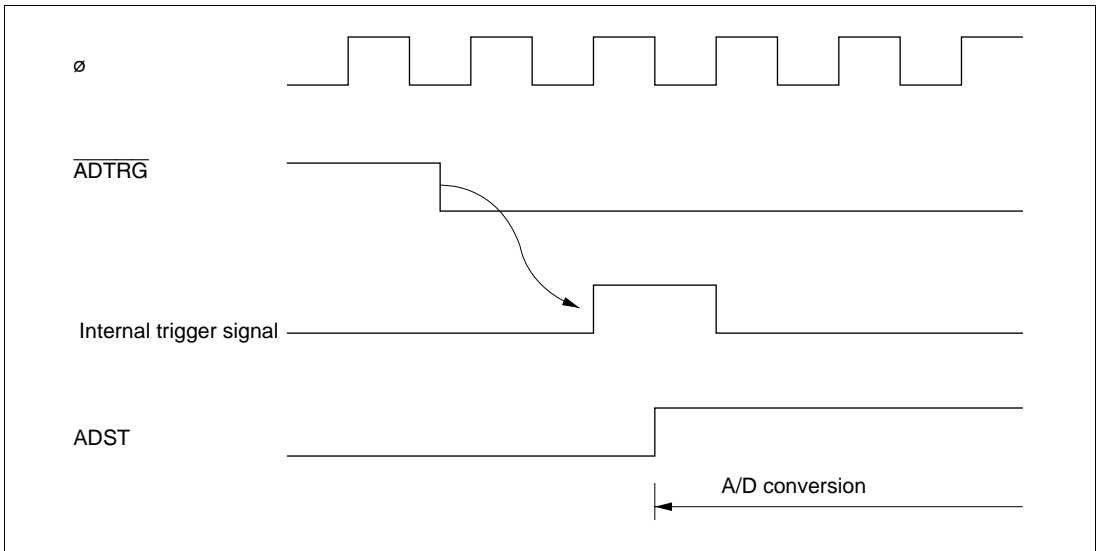
Note: Values in the table are the number of states.

**Table 14-5 A/D Conversion Time (Scan Mode)**

CKS1	CKS	Conversion Time (States)
0	0	512 (Fixed)
	1	64 (Fixed)
1	0	256 (Fixed)
	1	128 (Fixed)

#### 14.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to B'11 in ADCR, external trigger input is enabled at the  $\overline{ADTRG}$  pin. A falling edge at the  $\overline{ADTRG}$  pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 14-6 shows the timing.



**Figure 14-6 External Trigger Input Timing**

## 14.5 Interrupts

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. ADI interrupt requests can be enabled or disabled by means of the ADIE bit in ADCSR.

The DTC or DMAC can be activated by an ADI interrupt. Having the converted data read by the DTC or DMAC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

The A/D converter interrupt source is shown in table 14-6.

**Table 14-6 A/D Converter Interrupt Source**

Interrupt Source	Description	DTC Activation	DMAC Activation
ADI	Interrupt due to end of conversion	Possible	Possible

## 14.6 Usage Notes

The following points should be noted when using the A/D converter.

### Setting Range of Analog Power Supply and Other Pins

#### 1. Analog input voltage range

The voltage applied to analog input pins ANn during A/D conversion should be in the range  $AV_{SS} \leq ANn \leq V_{ref}$ .

#### 2. Relation between $AV_{CC}$ , $AV_{SS}$ and $V_{CC}$ , $V_{SS}$

As the relationship between  $AV_{CC}$ ,  $AV_{SS}$  and  $V_{CC}$ ,  $V_{SS}$ , set  $AV_{SS} = V_{SS}$ . If the A/D converter is not used, the  $AV_{CC}$  and  $AV_{SS}$  pins must not be left open.

#### 3. $V_{ref}$ input range

The analog reference voltage input at the  $V_{ref}$  pin should be set in the range  $V_{ref} \leq AV_{CC}$ .

If conditions 1, 2, and 3 above are not met, the reliability of the device may be adversely affected.

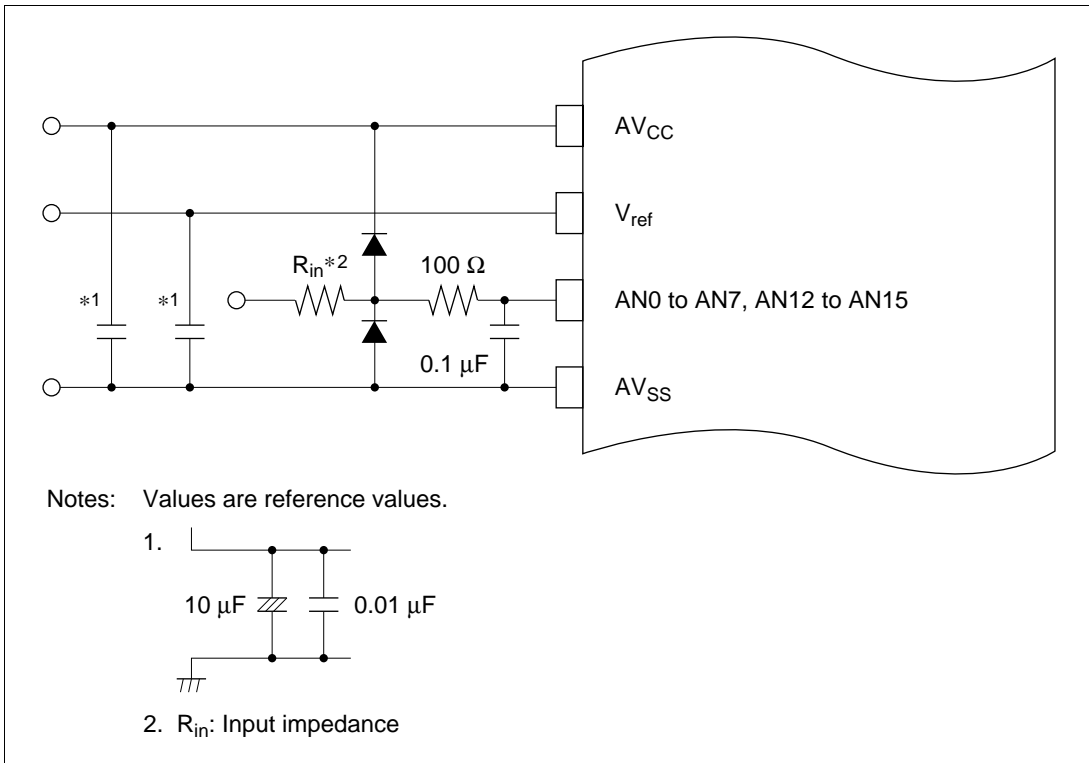
**Notes on Board Design:** In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values.

Also, digital circuitry must be isolated from the analog input signals (AN0 to AN7 and AN12 to AN15), analog reference power supply ( $V_{ref}$ ), and analog power supply ( $AV_{CC}$ ) by the analog ground ( $AV_{SS}$ ). Also, the analog ground ( $AV_{SS}$ ) should be connected at one point to a stable digital ground ( $V_{SS}$ ) on the board.

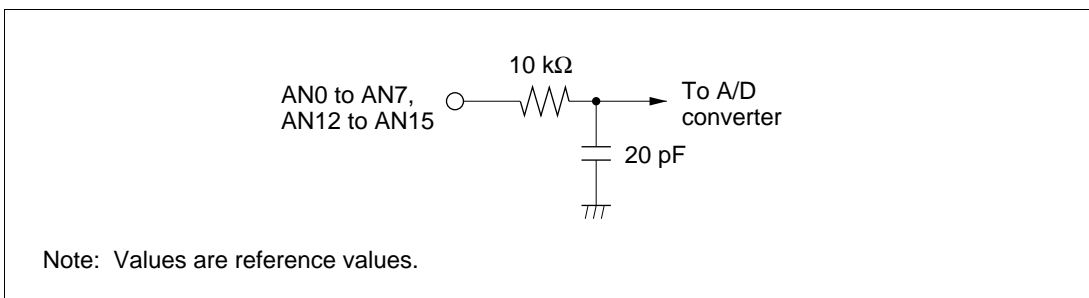
**Notes on Noise Countermeasures:** A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN7 and AN12 to AN15) and analog reference power supply ( $V_{ref}$ ) should be connected between  $AV_{CC}$  and  $AV_{SS}$  as shown in figure 14-7.

Also, the bypass capacitors connected to  $AV_{CC}$  and  $V_{ref}$  and the filter capacitor connected to AN0 to AN7 must be connected to  $AV_{SS}$ .

If a filter capacitor is connected as shown in figure 14-7, the input currents at the analog input pins (AN0 to AN7 and AN12 to AN15) are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance ( $R_{in}$ ), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.



**Figure 14-7 Example of Analog Input Protection Circuit**



**Figure 14-8 Analog Input Pin Equivalent Circuit**

**A/D Conversion Precision Definitions:** See this item in section 13.6.

**Permissible Signal Source Impedance:** See this item in section 13.6.

**Influences on Absolute Precision:** See this item in section 13.6.

# Section 15 D/A Converter

## 15.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series include an 8-bit resolution D/A converter with from two to four analog signal output channels.

The number of output channels differs from model to model; please check the reference manual for the relevant model for confirmation.

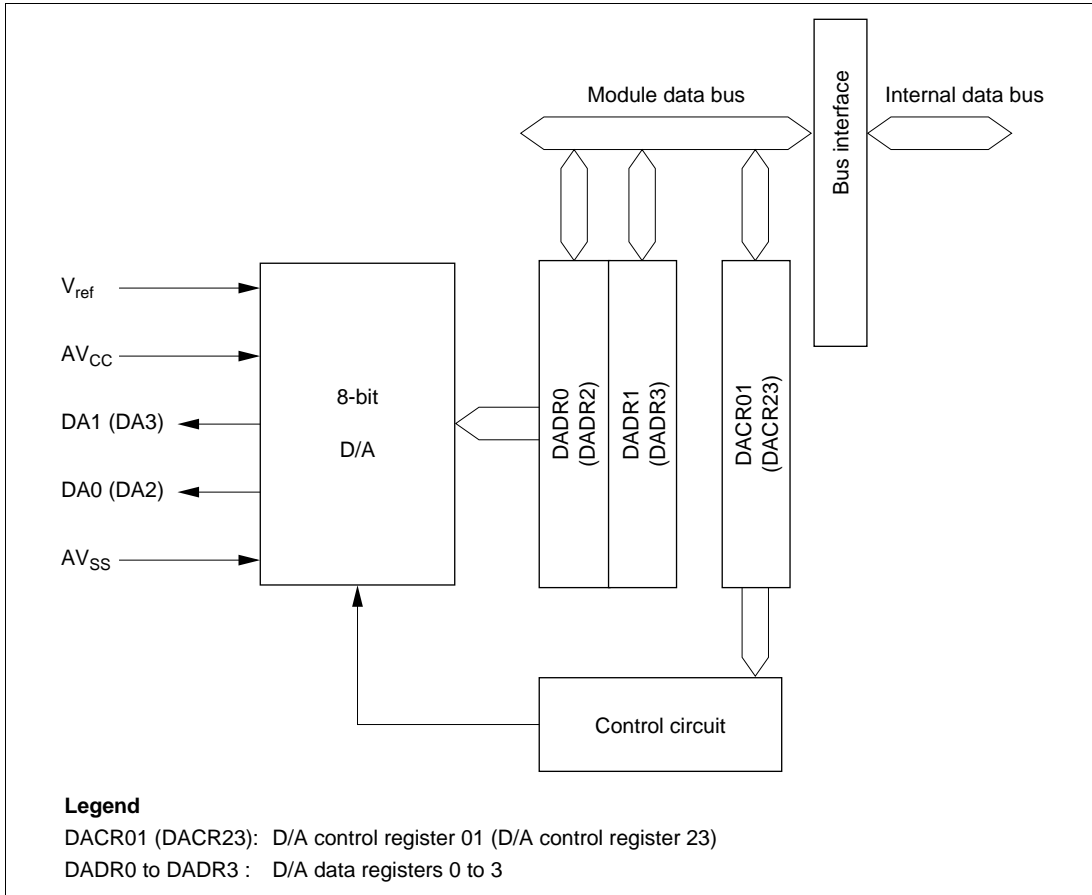
### 15.1.1 Features

D/A converter features are listed below.

- 8-bit resolution
- Two to four output channels
- Maximum conversion time of 10  $\mu$ s (with 20 pF load)
- Output voltage of 0 V to  $V_{ref}$
- D/A output hold function in software standby mode
- Module stop mode can be set
  - As the initial setting, D/A converter operation is halted. Register access is enabled by exiting module stop mode.

## 15.1.2 Block Diagram

Figure 15-1 shows a block diagram of the D/A converter.



**Figure 15-1 Block Diagram of D/A Converter**

### 15.1.3 Pin Configuration

Table 15-1 summarizes the input and output pins of the D/A converter.

**Table 15-1 Pin Configuration**

Pin Name	Symbol	I/O	Function
Analog power pin	$AV_{CC}$	Input	Analog power source
Analog ground pin	$AV_{SS}$	Input	Analog ground and reference voltage
Analog output pin 0	DA0	Output	Channel 0 analog output
Analog output pin 1	DA1	Output	Channel 1 analog output
Analog output pin 2	DA2	Output	Channel 2 analog output
Analog output pin 3	DA3	Output	Channel 3 analog output
Reference voltage pin	$V_{ref}$	Input	Analog reference voltage

### 15.1.4 Register Configuration

Table 15-2 summarizes the registers of the D/A converter.

**Table 15-2 D/A Converter Registers**

Channels	Name	Abbreviation	R/W	Initial Value	Address*
0, 1	D/A data register 0	DADR0	R/W	H'00	H'FFA4
	D/A data register 1	DADR1	R/W	H'00	H'FFA5
	D/A control register 01	DACR01	R/W	H'1F	H'FFA6
2, 3	D/A data register 2	DADR2	R/W	H'00	H'FFA8
	D/A data register 3	DADR3	R/W	H'00	H'FFA9
	D/A control register 23	DACR12	R/W	H'1F	H'FFAA
Common	Module stop control register	MSTPCR	R/W	H'3FFF	H'FF3C

Note:\* Lower 16 bits of the address.



## 15.2 Register Descriptions

### 15.2.1 D/A Data Registers 0 to 3 (DADR0 to DADR3)

Bit	:	7	6	5	4	3	2	1	0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DADR0 to DADR3 are 8-bit readable/writable registers that store data for conversion.

Whenever output is enabled, the values in DADR0 and DADR1 are converted and output from the analog output pins.

DADR0 to DADR3 are each initialized to H'00 by a reset and in hardware standby mode.

### 15.2.2 D/A Control Registers 01 and 23 (DACR01, DACR23)

Bit	:	7	6	5	4	3	2	1	0
		DAOE1	DAOE0	DAE	—	—	—	—	—
Initial value :		0	0	0	1	1	1	1	1
R/W	:	R/W	R/W	R/W	—	—	—	—	—

DACR01 and DACR23 are 8-bit readable/writable registers that control the operation of the D/A converter.

DACR01 and DACR23 are each initialized to H'1F by a reset and in hardware standby mode.

**Bit 7—D/A Output Enable 1 (DAOE1):** Controls D/A conversion and analog output.

Bit 7	Description
0	Analog output DA1 (DA3) is disabled (Initial value)
1	Channel 1 (channel 3) D/A conversion is enabled; analog output DA1 (DA3) is enabled

**Bit 6—D/A Output Enable 0 (DAOE0):** Controls D/A conversion and analog output.

**Bit 6**

DAOE0	Description
0	Analog output DA0 (DA2) is disabled (Initial value)
1	Channel 0 (channel 2) D/A conversion is enabled; analog output DA0 (DA2) is enabled

**Bit 5—D/A Enable (DAE):** Used together with the DAOE0 and DAOE1 bits to control D/A conversion. When the DAE bit is cleared to 0, channel 0 and 1 D/A conversions are controlled independently. When the DAE bit is set to 1, channel 0 and 1 D/A conversions are controlled together.

Output of conversion results is always controlled independently by the DAOE0 and DAOE1 bits.

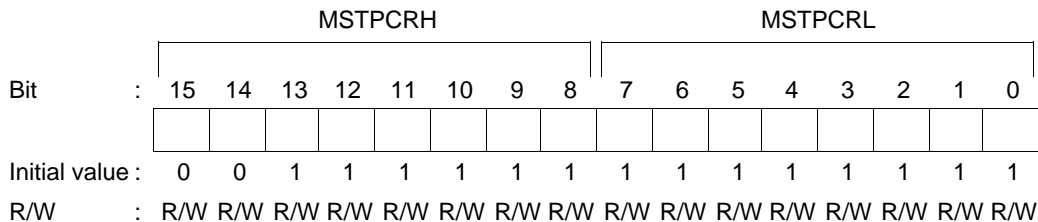
Bit 7 DAOE1	Bit 6 DAOE0	Bit 5 DAE	Description
0	0	*	Channel 0 and 1 (channel 2 and 3) D/A conversions disabled
		1	Channel 0 (channel 2) D/A conversion enabled Channel 1 (channel 3) D/A conversion disabled
		1	Channel 0 and 1 (channel 2 and 3) D/A conversions enabled
1	0	0	Channel 0 (channel 2) D/A conversion disabled Channel 1 (channel 3) D/A conversion enabled
		1	Channel 0 and 1 (channel 2 and 3) D/A conversions enabled
		*	Channel 0 and 1 (channel 2 and 3) D/A conversions enabled

\*: Don't care

If the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip enters software standby mode when D/A conversion is enabled, the D/A output is held and the analog power current is the same as during D/A conversion. When it is necessary to reduce the analog power current in software standby mode, clear both the DAOE0 and DAOE1 bits to 0 to disable D/A output.

**Bits 4 to 0—Reserved:** Read-only bits, always read as 1.

### 15.2.3 Module Stop Control Register (MSTPCR)



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

When the MSTP10 bit or MSTP4 bit in MSTPCR is set to 1, D/A converter operation stops at the end of the bus cycle and a transition is made to module stop mode. Registers cannot be read or written to in module stop mode. For details, see section 19.5, Module Stop Mode.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 10—Module Stop (MSTP10):** Specifies the D/A converter channel 0 and 1 module stop mode.

Bit 10 MSTP10	Description	
0	D/A converter (channel 0 and 1) module stop mode cleared	
1	D/A converter (channel 0 and 1) module stop mode set	(Initial value)

**Bit 4—Module Stop (MSTP4):** Specifies the D/A converter channel 2 and 3 module stop mode.

Bit 4 MSTP4	Description	
0	D/A converter (channel 2 and 3) module stop mode cleared	
1	D/A converter (channel 2 and 3) module stop mode set	(Initial value)

## 15.3 Operation

The D/A converter includes D/A conversion circuits for two channels, each of which can operate independently.

D/A conversion is performed continuously while enabled by DACR. If either DADR0 or DADR1 is written to, the new data is immediately converted. The conversion result is output by setting the corresponding DAOE0 or DAOE1 bit to 1.

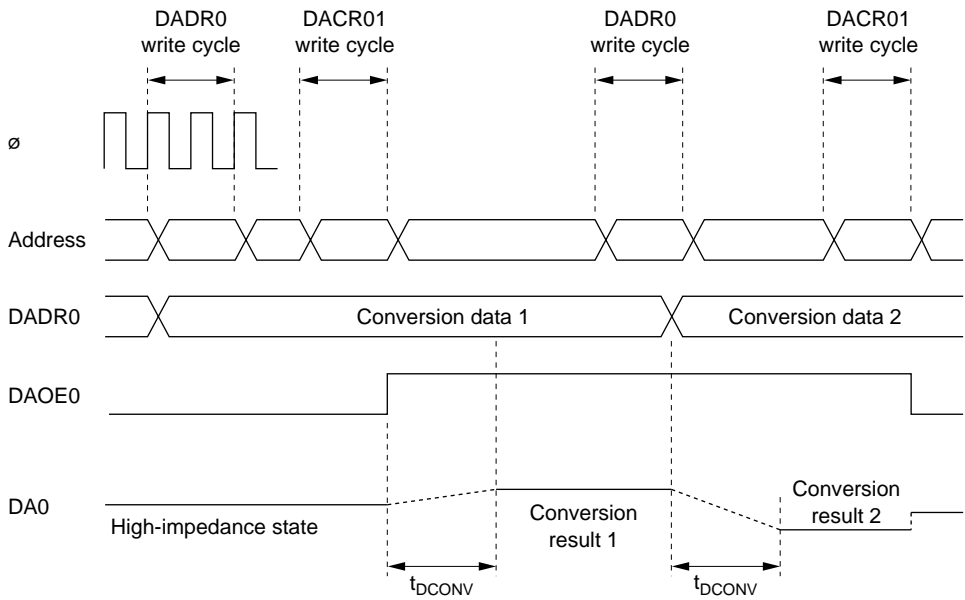
The operation example described in this section concerns D/A conversion on channel 0. Figure 15-2 shows the timing of this operation.

- [1] Write the conversion data to DADR0.
- [2] Set the DAOE0 bit in DACR01 to 1. D/A conversion is started and the DA0 pin becomes an output pin. The conversion result is output after the conversion time has elapsed. The output value is expressed by the following formula:

$$\frac{\text{DADR contents}}{256} \times V_{\text{ref}}$$

The conversion results are output continuously until DADR0 is written to again or the DAOE0 bit is cleared to 0.

- [3] If DADR0 is written to again, the new data is immediately converted. The new conversion result is output after the conversion time has elapsed.
- [4] If the DAOE0 bit is cleared to 0, the DA0 pin becomes an input pin.



**Legend**

$t_{DCONV}$ : D/A conversion time

**Figure 15-2 Example of D/A Converter Operation**

# Section 16 RAM

## 16.1 Overview

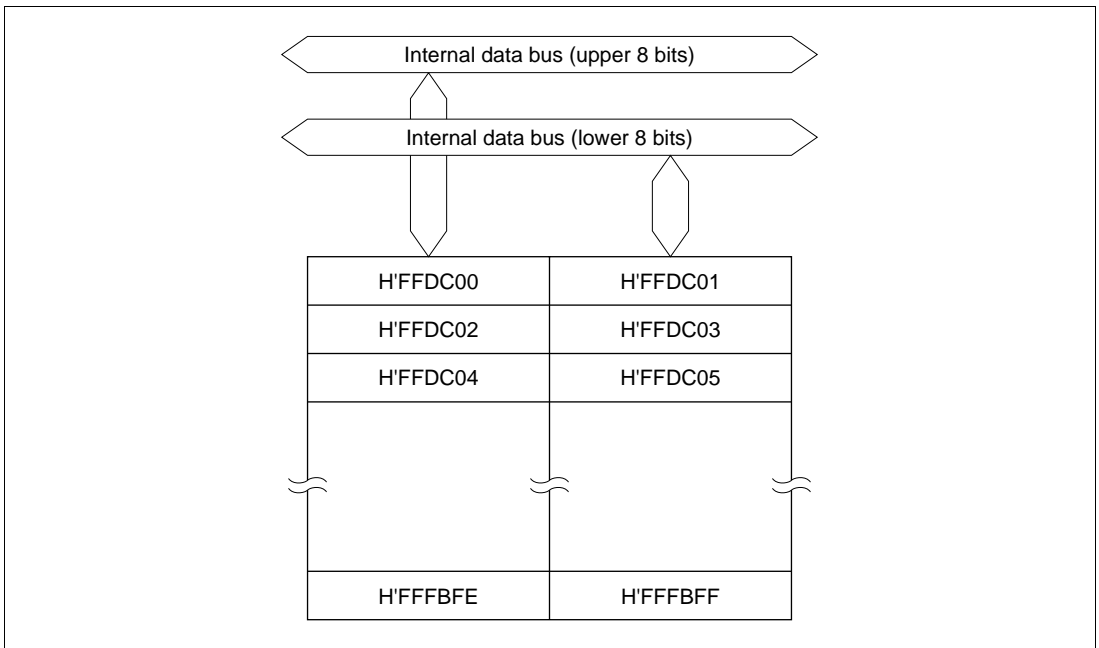
The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data. This makes it possible to perform fast word data transfer.

The on-chip RAM can be enabled or disabled by means of the RAM enable bit (RAME) in the system control register (SYSCR).

The amount of on-chip RAM differs from model to model; please check the reference manual for the relevant model for confirmation.

### 16.1.1 Block Diagram

Figure 16-1 shows a block diagram of 8 kbytes of on-chip RAM.



**Figure 16-1 Block Diagram of RAM (8 kbytes)**

## 16.1.2 Register Configuration

The on-chip RAM is controlled by SYSCR. Table 16-1 shows the address and initial value of SYSCR.

**Table 16-1 RAM Register**

Name	Abbreviation	R/W	Initial Value	Address*
System control register	SYSCR	R/W	H'01	H'FF39

Note: \* Lower 16 bits of the address.

## 16.2 Register Descriptions

### 16.2.1 System Control Register (SYSCR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	INTM1	INTM0	NMIEG	LWROD	IRQPAS	RAME
Initial value :		0	0	0	0	0	0	0	1
R/W	:	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W

The on-chip RAM is enabled or disabled by the RAME bit in SYSCR. For details of other bits in SYSCR, see section 3.2.1, System Control Register (SYSCR).

**Bit 0—RAM Enable (RAME):** Enables or disables the on-chip RAM. The RAME bit is initialized when the reset state is released. It is not initialized in software standby mode.

Bit 0 RAME	Description
0	On-chip RAM is disabled
1	On-chip RAM is enabled (Initial value)

## 16.3 Operation

When the RAME bit is set to 1, accesses to addresses H'FFDC00 to H'FFFBFF are directed to the on-chip RAM. When the RAME bit is cleared to 0, the off-chip address space is accessed.

Since the on-chip RAM is connected to the CPU by an internal 16-bit data bus, it can be written to and read in byte or word units. Each type of access can be performed in one state.

Even addresses use the upper 8 bits, and odd addresses use the lower 8 bits. Word data must start at an even address.

## 16.4 Usage Note

DTC register information can be located in addresses H'FFF800 to H'FFFBFF. When the DTC is used, the RAME bit must not be cleared to 0.



# Section 17 ROM

## 17.1 Overview

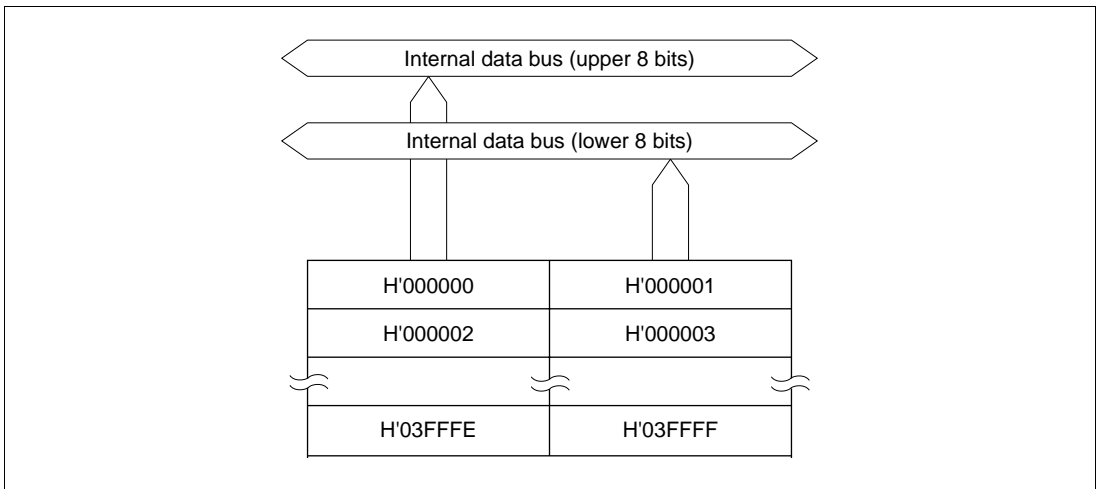
The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have 256 kbyte of on-chip flash memory, or 256, 128, or 32 kbytes of on-chip mask ROM. The ROM is connected to the bus master via a 16-bit data bus, enabling both byte and word data to be accessed in one state. Instruction fetching is thus speeded up, and processing speed increased.

The on-chip ROM is enabled and disabled by means of the mode pins ( $MD_2$ ,  $MD_1$ , and  $MD_0$ ) and the EAE bit in BCRL.

The flash memory version of the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series can be erased and programmed with a PROM programmer, as well as on-board.

### 17.1.1 Block Diagram

Figure 17-1 shows a block diagram of 256 kbytes of on-chip ROM.



**Figure 17-1 Block Diagram of ROM (256 kbytes)**

## 17.1.2 Register Configuration

The operating mode of the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip is controlled by the mode pins and the BCRL register. The ROM-related registers are shown in table 17-1.

**Table 17-1 ROM Registers**

Register Name	Abbreviation	R/W	Initial Value	Address*
Mode control register	MDCR	R/W	Undefined	H'FF3B
Bus controller register	BCRL	R/W	Undefined	H'FED5

Note: \* Lower 16 bits of the address.

## 17.2 Register Descriptions

### 17.2.1 Mode Control Register (MDCR)

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	—	MDS2	MDS1	MDS0
Initial value :		1	0	0	0	0	—*	—*	—*
R/W	:	—	—	—	—	—	R	R	R

Note: \* Determined by pins MD<sub>2</sub> to MD<sub>0</sub>.

MDCR is an 8-bit read-only register used to monitor the current operating mode of the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip.

**Bit 7—Reserved:** Read-only bit, always read as 1.

**Bits 6 to 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—Mode Select 2 to 0 (MDS2 to MDS0):** These bits indicate the input levels at pins MD<sub>2</sub> to MD<sub>0</sub> (the current operating mode). Bits MDS2 to MDS0 correspond to pins MD<sub>2</sub> to MD<sub>0</sub>. MDS2 to MDS0 are read-only bits, and cannot be modified. The mode pin (MD<sub>2</sub> to MD<sub>0</sub>) input levels are latched into these bits when MDCR is read. These latches are canceled by a power-on reset.

## 17.2.2 Bus Control Register L (BCRL)

Bit	:	7	6	5	4	3	2	1	0
		BRLE	BREQ0E	EAE	—	DDS	—	WDBE	WAITE
Initial value :		0	0	1	1	1	1	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Enabling or disabling of part of the on-chip ROM area in the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series can be selected by means of the EAE bit in BCRL. For details of the other bits in BCRL, see section 4.2.5, Bus Control Register L (BCRL).

**Bit 5—External Address Enable (EAE):** Designates addresses H'010000 to H'03FFFF as either internal or external addresses.

### Bit 5

EAE	Description
0	Addresses H'010000 to H'03FFFF* <sup>1</sup> are in on-chip ROM
1	Addresses H'010000 to H'03FFFF* <sup>1</sup> are external addresses (in external expanded mode) or a reserved area* <sup>2</sup> (in single-chip mode) (Initial value)

- Notes: 1. The on-chip ROM area differs from model to model; please check the reference manual for the relevant model for confirmation.  
2. A reserved area must not be accessed.

## 17.3 Operation

The on-chip ROM is connected to the CPU by a 16-bit data bus, and both byte and word data can be accessed in one state. Even addresses are connected to the upper 8 bits, and odd addresses to the lower 8 bits. Word data must start at an even address.

The on-chip ROM is enabled and disabled by setting the mode pins (MD<sub>2</sub>, MD<sub>1</sub>, and MD<sub>0</sub>) and the EAE bit in BCRL. These settings are shown in table 17-2.

**Table 17-2 Operating Modes and ROM (F-ZTAT Version)**

Mode	Operating Mode	Mode Pins				BCRL	
		FWE	MD2	MD1	MD0	EAE	On-Chip ROM
0	—	0	0	0	0	—	—
1					1		
2				1	0		
3					1		
4	Advanced expanded mode with on-chip ROM disabled		1	0	0	—	Disabled
5	Advanced expanded mode with on-chip ROM disabled				1		
6	Advanced expanded mode with on-chip ROM enabled			1	0	0	Enabled (256 kbytes)* <sup>1</sup>
						1	Enabled (64 kbytes)
7	Advanced single-chip mode				1	0	Enabled (256 kbytes)* <sup>1</sup>
						1	Enabled (64 kbytes)
8	—	1	0	0	0	—	—
9					1		
10	Boot mode (advanced expanded mode with on-chip ROM enabled)* <sup>3</sup>			1	0	0	Enabled (256 kbytes)* <sup>2</sup>
						1	Enabled (64 kbytes)
11	Boot mode (advanced single-chip mode)* <sup>4</sup>				1	0	Enabled (256 kbytes)* <sup>2</sup>
						1	Enabled (64 kbytes)
12	—		1	0	0	—	—
13					1		
14	User program mode (advanced expanded mode with on-chip ROM enabled)* <sup>3</sup>			1	0	0	Enabled (256 kbytes)* <sup>1</sup>
						1	Enabled (64 kbytes)
15	User program mode (advanced single-chip mode)* <sup>4</sup>				1	0	Enabled (256 kbytes)* <sup>1</sup>
						1	Enabled (64 kbytes)

Notes: 1. Note that in modes 6, 7, 14, and 15, the on-chip ROM that can be used after a power-on reset is the 64-kbyte area from H'000000 to H'00FFFF.

2. Note that in the mode 10 and mode 11 boot modes, the on-chip ROM that can be used immediately after all flash memory is erased by the boot program is the 64-kbyte area from H'000000 to H'00FFFF.
3. Apart from the fact that flash memory can be erased and programmed, operation is the same as in advanced expanded mode with on-chip ROM enabled.
4. Apart from the fact that flash memory can be erased and programmed, operation is the same as in advanced single-chip mode.

**Table 17-3 Operating Modes and ROM (Mask ROM Version)**

Mode	Operating Mode	Mode Pins			BCRL	
		MD2	MD1	MD0	EAE	On-Chip ROM
0	—	0	0	0	—	—
1				1		
2			1	0		
3				1		
4	Advanced expanded mode with on-chip ROM disabled	1	0	0	—	Disabled
5	Advanced expanded mode with on-chip ROM disabled			1		
6	Advanced expanded mode with on-chip ROM enabled		1	0	0 1	Enabled (256 kbytes)* Enabled (64 kbytes)
7	Advanced single-chip mode			1	0 1	Enabled (256 kbytes)* Enabled (64 kbytes)

Note: \* Note that in modes 6 and 7, the on-chip ROM that can be used after a power-on reset is the 64-kbyte area from H'000000 to H'00FFFF.

## 17.4 Overview of Flash Memory

### 17.4.1 Features

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have 256 kbytes of on-chip flash memory. The features of the flash memory are summarized below.

- Four flash memory operating modes
  - Program mode
  - Erase mode
  - Program-verify mode
  - Erase-verify mode
- Programming/erase methods

The flash memory is programmed 128 bytes at a time. Erasing is performed by block erase (in single-block units). To erase the entire flash memory, the individual blocks must be erased sequentially. Block erasing can be performed as required on 4-kbyte, 32-kbyte, and 64-kbyte blocks.
- Programming/erase times

The flash memory programming time is T.B.D. ms (typ.) for simultaneous 128-byte programming, equivalent to T.B.D.  $\mu$ s (typ.) per byte, and the erase time is T.B.D. ms (typ.).
- Reprogramming capability

The flash memory can be reprogrammed up to 100 times.
- On-board programming modes

There are two modes in which flash memory can be programmed/erased/verified on-board:

  - Boot mode
  - User program mode
- Automatic bit rate adjustment

With data transfer in boot mode, the bit rate of the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip can be automatically adjusted to match the transfer bit rate of the host.
- Flash memory emulation by RAM

Part of the RAM area can be overlapped onto flash memory, to emulate flash memory updates in real time.
- Protect modes

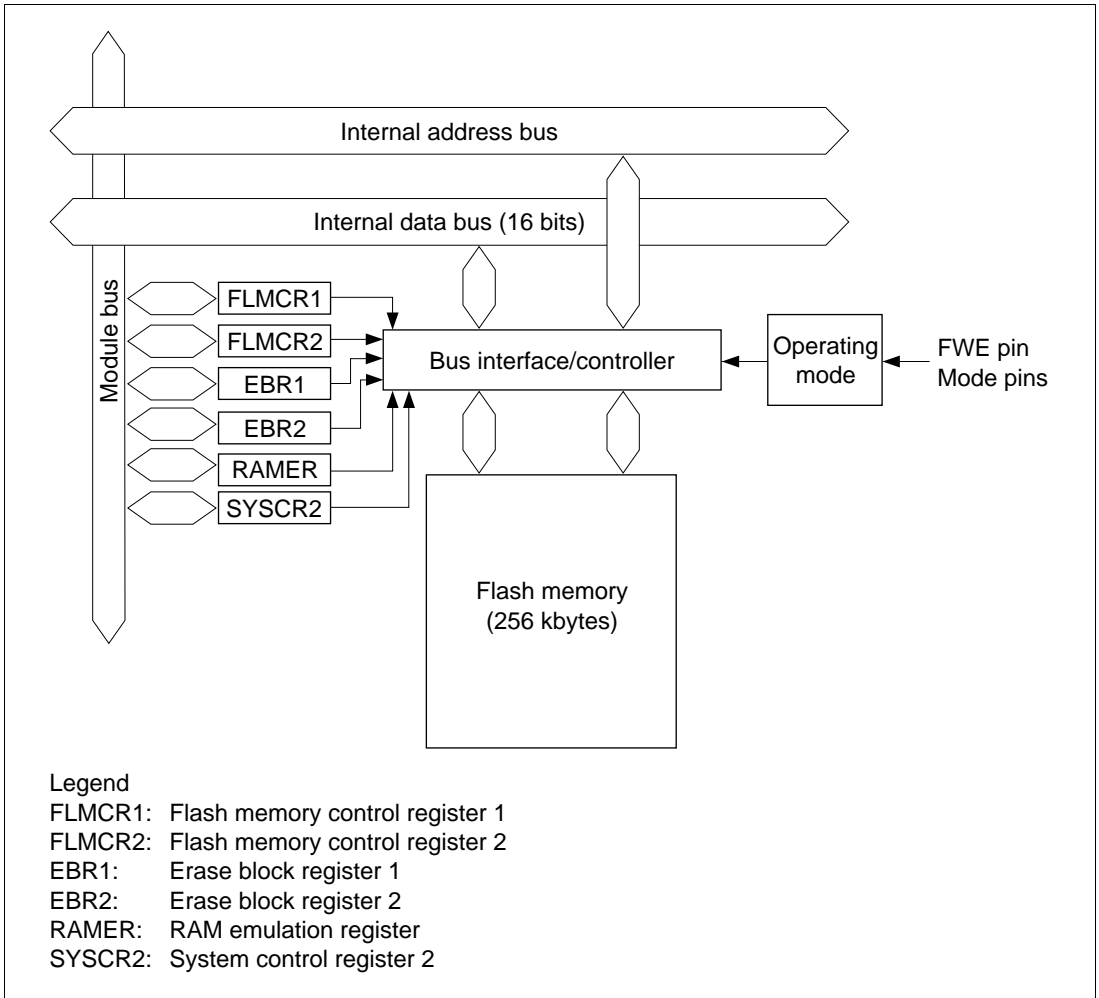
There are three protect modes, hardware, software, and error protect, which allow protected status to be designated for flash memory program/erase/verify operations.

- PROM mode

Flash memory can be programmed/erased in PROM mode, using a PROM programmer, as well as in on-board programming mode.

## 17.4.2 Overview

### Block Diagram



**Figure 17-2 Block Diagram of Flash Memory**

### 17.4.3 Flash Memory Operating Modes

**Mode Transitions:** When the mode pins and the FWE pin are set in the reset state and a reset-start is executed, the chip enters one of the operating modes shown in figure 17-3. In user mode, flash memory can be read but not programmed or erased.

Flash memory can be programmed and erased in boot mode, user program mode, and PROM mode.

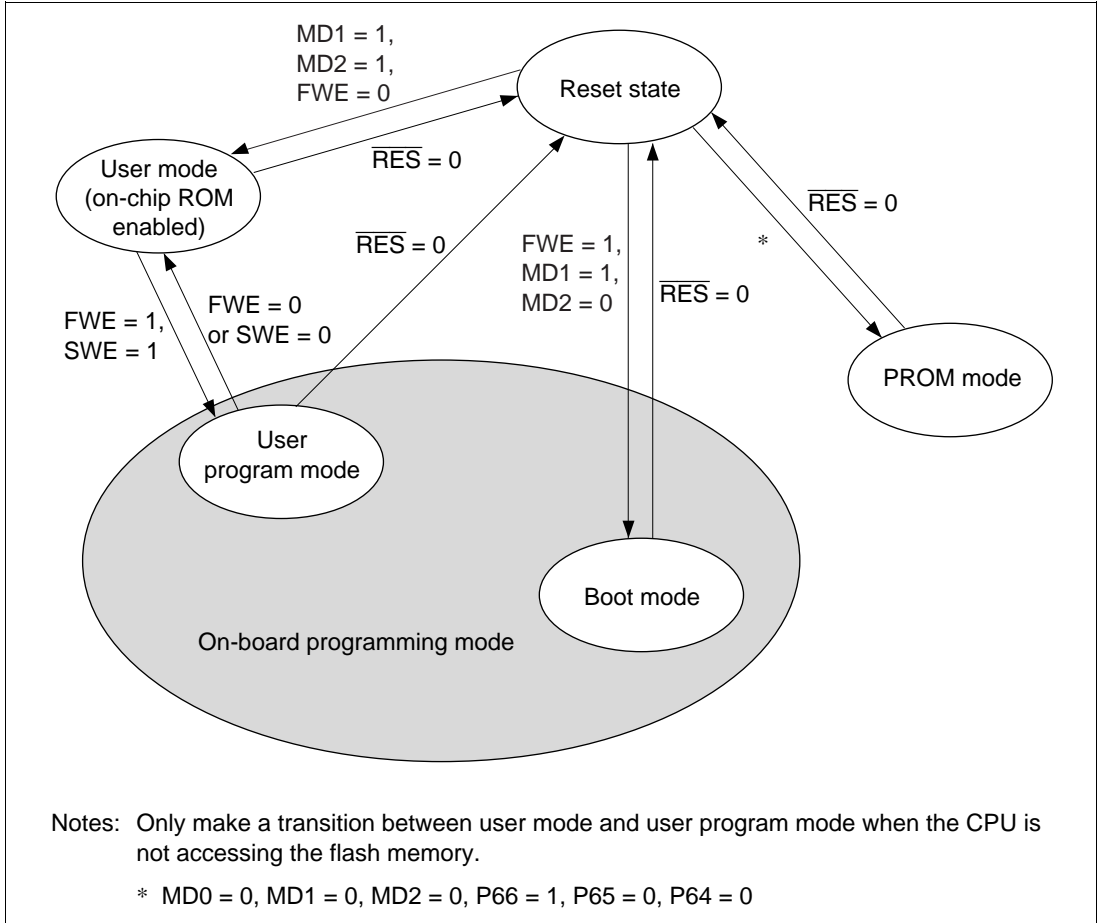


Figure 17-3 Flash Memory Mode Transitions

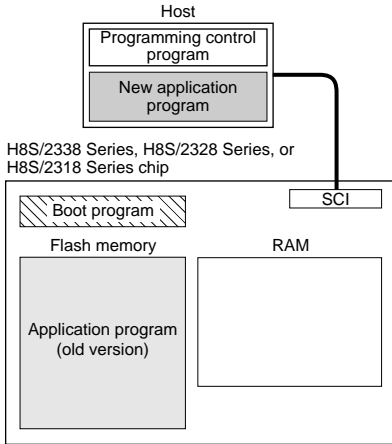


## 17.4.4 On-Board Programming Modes

- Boot mode

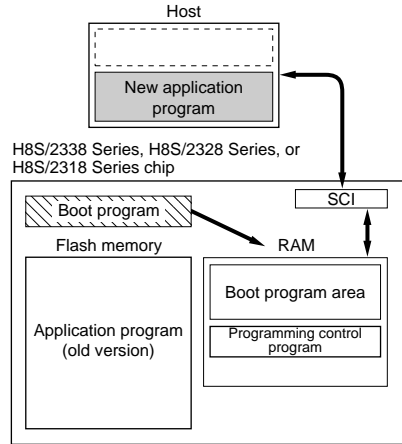
### 1. Initial state

The old program version or data remains written in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.



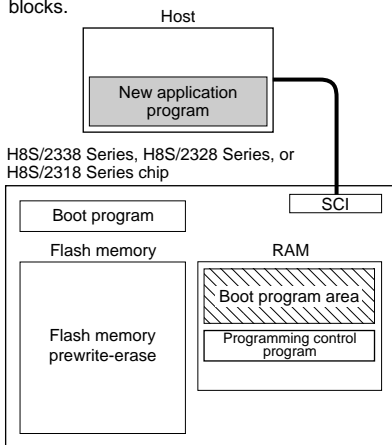
### 2. Programming control program transfer

When boot mode is entered, the boot program in the chip (originally incorporated in the chip) is started and the programming control program in the host is transferred to RAM via SCI communication. The boot program required for flash memory erasing is automatically transferred to the RAM boot program area.



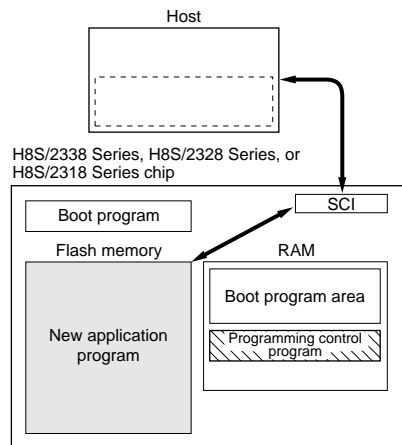
### 3. Flash memory initialization

The erase program in the boot program area (in RAM) is executed, and the flash memory is initialized (to H'FF). In boot mode, entire flash memory erasure is performed, without regard to blocks.



### 4. Writing new application program

The programming control program transferred from the host to RAM is executed, and the new application program in the host is written into the flash memory.




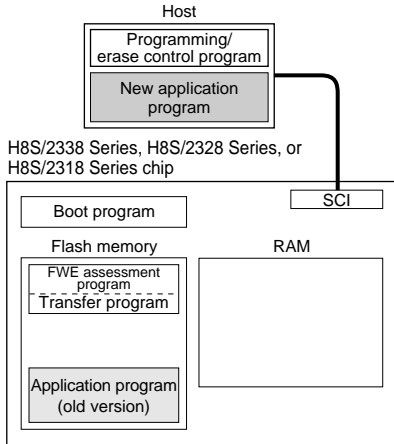
 Program execution state

Figure 17-4 Boot Mode

- User program mode

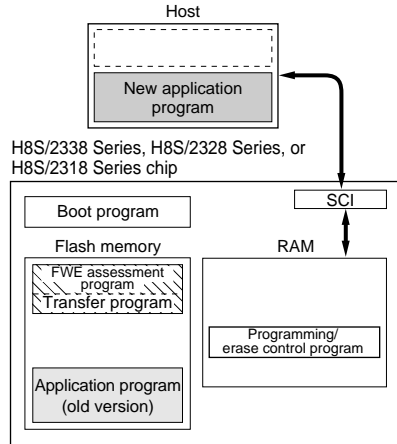
1. Initial state

(1) The FWE assessment program that confirms that the FWE pin has been driven high, and (2) the program that will transfer the programming/erase control program to on-chip RAM should be written into the flash memory by the user beforehand. (3) The programming/erase control program should be prepared in the host or in the flash memory.



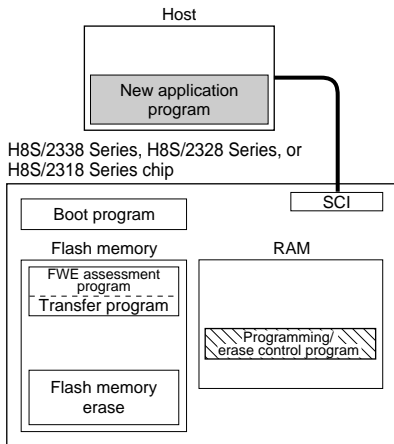
2. Programming/erase control program transfer

When the FWE pin is driven high, user software confirms this fact, executes the transfer program in the flash memory, and transfers the programming/erase control program to RAM.



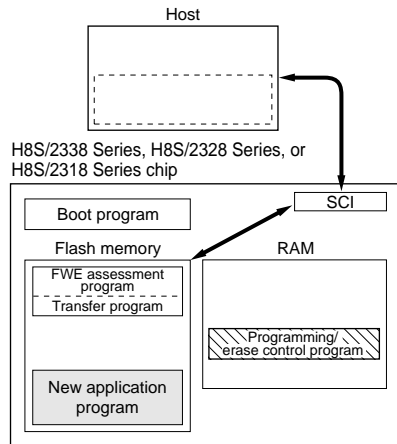
3. Flash memory initialization

The programming/erase program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.



4. Writing new application program

Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.

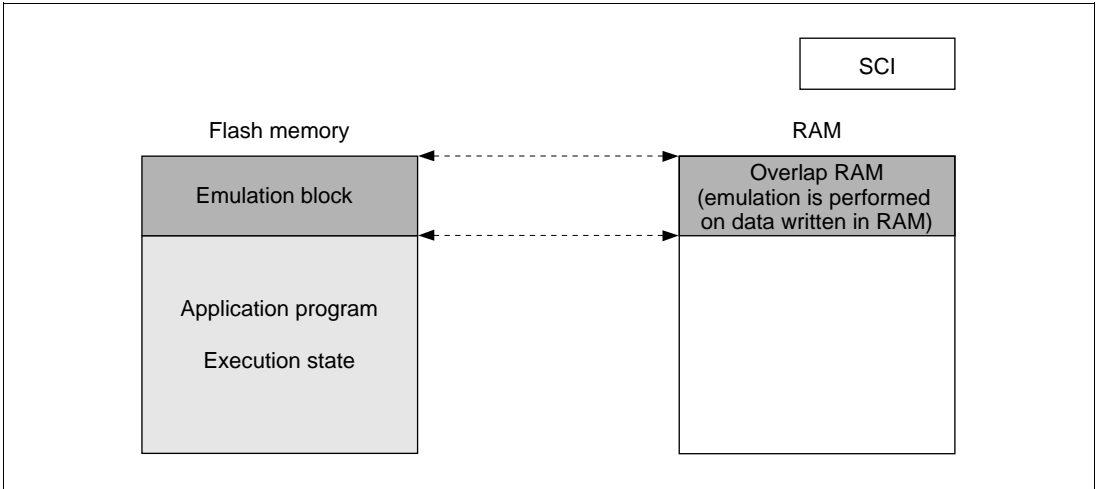


Program execution state

**Figure 17-5 User Program Mode (Example)**

## 17.4.5 Flash Memory Emulation in RAM

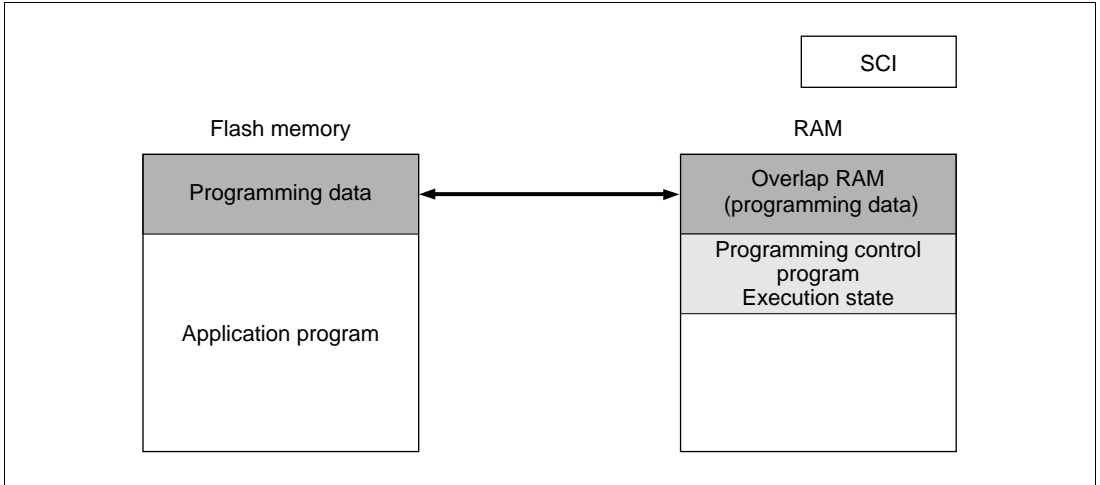
**Reading Overlap RAM Data in User Mode and User Program Mode:** Emulation should be performed in user mode or user program mode. When the emulation block set in RAMER is accessed while the emulation function is being executed, data written in the overlap RAM is read.



**Figure 17-6 Reading Overlap RAM Data in User Mode and User Program Mode**

**Writing Overlap RAM Data in User Program Mode:** When overlap RAM data is confirmed, the RAMS bit is cleared, RAM overlap is released, and writes should actually be performed to the flash memory.

When the programming control program is transferred to RAM, ensure that the transfer destination and the overlap RAM do not overlap, as this will cause data in the overlap RAM to be rewritten.



**Figure 17-7 Writing Overlap RAM Data in User Program Mode**

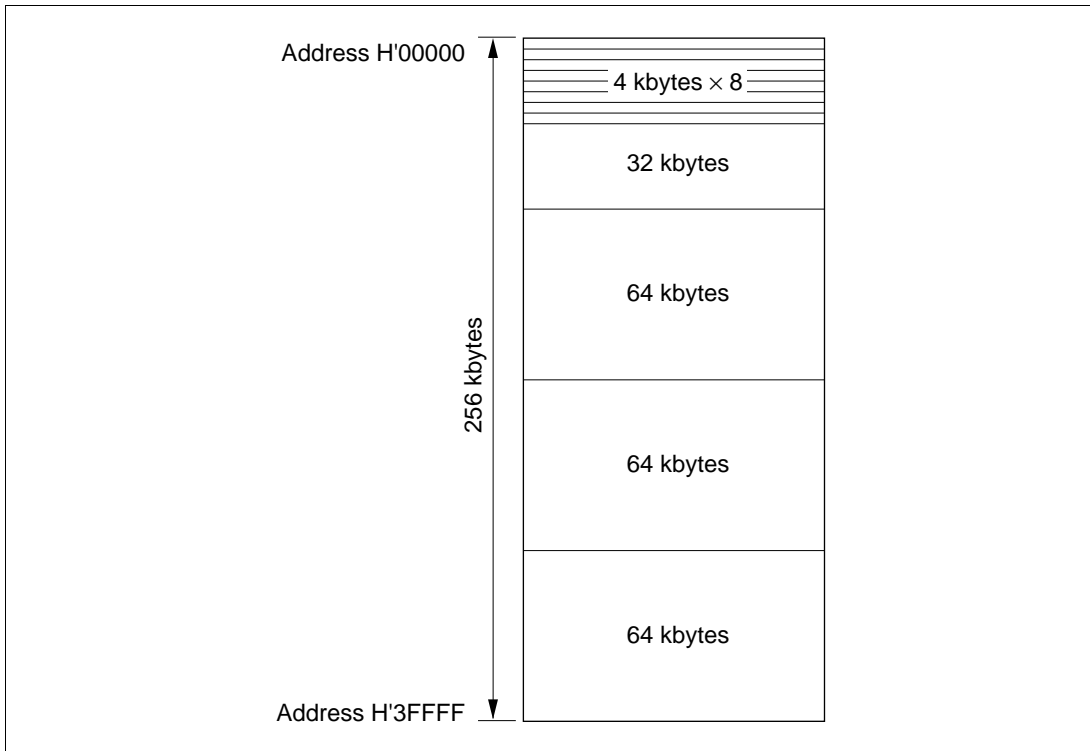
#### 17.4.6 Differences between Boot Mode and User Program Mode

	<b>Boot Mode</b>	<b>User Program Mode</b>
Entire memory erase	Yes	Yes
Block erase	No	Yes
Programming control program*	Program/program-verify	Erase/erase-verify/program/ program-verify emulation

Note: \* To be provided by the user, in accordance with the recommended algorithm.

### 17.4.7 Block Configuration

The flash memory is divided into three 64-kbyte blocks, one 32-kbyte block, and eight 4-kbyte blocks.



**Figure 17-8 Flash Memory Block Configuration**

## 17.4.8 Pin Configuration

The flash memory is controlled by means of the pins shown in table 17-4.

**Table 17-4 Flash Memory Pins**

<b>Pin Name</b>	<b>Abbreviation</b>	<b>I/O</b>	<b>Function</b>
Reset	$\overline{\text{RES}}$	Input	Reset
Flash write enable	FWE	Input	Flash program/erase protection by hardware
Mode 2	MD2	Input	Sets MCU operating mode
Mode 1	MD1	Input	Sets MCU operating mode
Mode 0	MD0	Input	Sets MCU operating mode
Port 64	P64	Input	Sets MCU operating mode in writer mode
Port 65	P65	Input	Sets MCU operating mode in writer mode
Port 66	P66	Input	Sets MCU operating mode in writer mode
Transmit data	TxD1	Output	Serial transmit data output
Receive data	RxD1	Input	Serial receive data input

## 17.4.9 Register Configuration

The registers used to control the on-chip flash memory when enabled are shown in table 17-5. In order to access the FLMCR1, FLMCR2, EBR1, and EBR2 registers, the FLSHE bit must be set to 1 in SYSCR2 (except RAMER).

**Table 17-5 Flash Memory Registers**

Register Name	Abbreviation	R/W	Initial Value	Address* <sup>1</sup>
Flash memory control register 1	FLMCR1* <sup>6</sup>	R/W* <sup>3</sup>	H'00* <sup>4</sup>	H'FFC8* <sup>2</sup>
Flash memory control register 2	FLMCR2* <sup>6</sup>	R/W* <sup>3</sup>	H'00* <sup>5</sup>	H'FFC9* <sup>2</sup>
Erase block register 1	EBR1* <sup>6</sup>	R/W* <sup>3</sup>	H'00* <sup>5</sup>	H'FFCA* <sup>2</sup>
Erase block register 2	EBR2* <sup>6</sup>	R/W* <sup>3</sup>	H'00* <sup>5</sup>	H'FFCB* <sup>2</sup>
System control register 2	SYSCR2* <sup>7</sup>	R/W	H'00	H'FF42
RAM emulation register	RAMER	R/W	H'00	H'FEDB

Notes: 1. Lower 16 bits of the address.

2. Flash memory. Registers selection is performed by the FLSHE bit in system control register 2 (SYSCR2).
3. In modes in which the on-chip flash memory is disabled, a read will return H'00, and writes are invalid. Writes are also disabled when the FWE bit is cleared to 0 in FLMCR1.
4. When a high level is input to the FWE pin, the initial value is H'80.
5. When a low level is input to the FWE pin, or if a high level is input and the SWE bit in FLMCR1 is not set, these registers are initialized to H'00.
6. FLMCR1, FLMCR2, EBR1, and EBR2 are 8-bit registers. Only byte accesses are valid for these registers, the access requiring 2 states.
7. The SYSCR2 register can only be used in the F-ZTAT version. In the mask ROM version this register will return an undefined value if read, and cannot be modified.

## 17.5 Register Descriptions

### 17.5.1 Flash Memory Control Register 1 (FLMCR1)

Bit	:	7	6	5	4	3	2	1	0
		FWE	SWE	ESU	PSU	EV	PV	E	P
Initial value :		1/0	0	0	0	0	0	0	0
R/W	:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

FLMCR1 is an 8-bit register used for flash memory operating mode control. Program-verify mode or erase-verify mode for addresses H'00000 to H'3FFFF is entered by setting SWE to 1 when FWE = 1, then setting the EV or PV bit. Program mode for addresses H'00000 to H'3FFFF is entered by setting SWE to 1 when FWE = 1, then setting the PSU bit, and finally setting the P bit. Erase mode for addresses H'00000 to H'3FFFF is entered by setting SWE to 1 when FWE = 1, then setting the ESU bit, and finally setting the E bit. FLMCR1 is initialized by a power-on reset, and in hardware standby mode and software standby mode. Its initial value is H'80 when a high level is input to the FWE pin, and H'00 when a low level is input. When on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

Writes to the SWE bit in FLMCR1 are enabled only when FWE = 1; writes to bits ESU, PSU, EV, and PV only when FWE = 1 and SWE = 1; writes to the E bit only when FWE = 1, SWE = 1, and ESU = 1; and writes to the P bit only when FWE = 1, SWE = 1, and PSU = 1.

**Bit 7—Flash Write Enable Bit (FWE):** Sets hardware protection against flash memory programming/erasing.

Bit 7	
FWE	Description
0	When a low level is input to the FWE pin (hardware-protected state)
1	When a high level is input to the FWE pin

**Bit 6—Software Write Enable Bit (SWE):** Enables or disables flash memory programming and erasing. This bit should be set when setting bits 5 to 0, EBR1 bits 7 to 0, and EBR2 bits 3 to 0.

When SWE = 1, the flash memory can only be read in program-verify or erase-verify mode.



<b>Bit 6</b> <b>SWE</b>	<b>Description</b>
0	Writes disabled (Initial value)
1	Writes enabled [Setting condition] When FWE = 1

**Bit 5—Erase Setup Bit (ESU):** Prepares for a transition to erase mode. Do not set the SWE, PSU, EV, PV, E, or P bit at the same time.

<b>Bit 5</b> <b>ESU</b>	<b>Description</b>
0	Erase setup cleared (Initial value)
1	Erase setup [Setting condition] When FWE = 1 and SWE = 1

**Bit 4—Program Setup Bit (PSU):** Prepares for a transition to program mode. Do not set the SWE, ESU, EV, PV, E, or P bit at the same time.

<b>Bit 4</b> <b>PSU</b>	<b>Description</b>
0	Program setup cleared (Initial value)
1	Program setup [Setting condition] When FWE = 1 and SWE = 1

**Bit 3—Erase-Verify (EV):** Selects erase-verify mode transition or clearing. Do not set the SWE, ESU, PSU, PV, E, or P bit at the same time.

<b>Bit 3</b> <b>EV</b>	<b>Description</b>
0	Erase-verify mode cleared (Initial value)
1	Transition to erase-verify mode [Setting condition] When FWE = 1 and SWE = 1

**Bit 2—Program-Verify (PV):** Selects program-verify mode transition or clearing. Do not set the SWE, ESU, PSU, EV, E, or P bit at the same time.

<b>Bit 2 PV</b>	<b>Description</b>	
0	Program-verify mode cleared	(Initial value)
1	Transition to program-verify mode [Setting condition] When FWE = 1 and SWE = 1	

**Bit 1—Erase (E):** Selects erase mode transition or clearing. Do not set the SWE, ESU, PSU, EV, PV, or P bit at the same time.

<b>Bit 1 E</b>	<b>Description</b>	
0	Erase mode cleared	(Initial value)
1	Transition to erase mode [Setting condition] When FWE = 1, SWE = 1, and ESU = 1	

**Bit 0—Program (P):** Selects program mode transition or clearing. Do not set the SWE, PSU, ESU, EV, PV, or E bit at the same time.

<b>Bit 0 P</b>	<b>Description</b>	
0	Program mode cleared	(Initial value)
1	Transition to program mode [Setting condition] When FWE = 1, SWE = 1, and PSU = 1	

### 17.5.2 Flash Memory Control Register 2 (FLMCR2)

Bit	:	7	6	5	4	3	2	1	0
		FLER	—	—	—	—	—	—	—
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R	—	—	—	—	—	—	—

FLMCR2 is an 8-bit register that controls the flash memory operating modes. FLMCR2 is initialized to H'00 by a power-on reset, and in hardware standby mode and software standby mode.

When on-chip flash memory is disabled, a read will return H'00 and writes are invalid.

**Bit 7—Flash Memory Error (FLER):** Indicates that an error has occurred during an operation on flash memory (programming or erasing). When FLER is set to 1, flash memory goes to the error-protection state.

#### Bit 7

FLER	Description
0	Flash memory is operating normally (Initial value) Flash memory program/erase protection (error protection) is disabled [Clearing condition] Power-on reset or hardware standby mode
1	An error has occurred during flash memory programming/erasing Flash memory program/erase protection (error protection) is enabled [Setting condition] See section 17.8.3, Error Protection

**Bits 6 to 0—Reserved:** Read-only bits, always read as 0.

### 17.5.3 Erase Block Register 1 (EBR1)

Bit	:	7	6	5	4	3	2	1	0
EBR1		EB7	EB6	EB5	EB4	EB3	EB2	EB1	EB0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

EBR1 is an 8-bit register that specifies the flash memory erase area block by block. EBR1 is initialized to H'00 by a power-on reset, in hardware standby mode and software standby mode, when a low level is input to the FWE pin, and when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. When a bit in EBR1 is set, the corresponding block can be erased. Other blocks are erase-protected. Set only one bit in EBR1 and EBR2 together (setting more than one bit will automatically clear all EBR1 and EBR2 bits to 0). When on-chip flash memory is disabled, a read will return H'00 and writes are invalid.

The flash memory block configuration is shown in table 17-6.

### 17.5.4 Erase Block Registers 2 (EBR2)

Bit	:	7	6	5	4	3	2	1	0
EBR2		—	—	—	—	EB11	EB10	EB9	EB8
Initial value :		0	0	0	0	0	0	0	0
R/W	:	—	—	—	—	R/W	R/W	R/W	R/W

EBR2 is an 8-bit register that specifies the flash memory erase area block by block. EBR2 is initialized to H'00 by a power-on reset, in hardware standby mode and software standby mode, when a low level is input to the FWE pin, and when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. When a bit in EBR2 is set, the corresponding block can be erased. Other blocks are erase-protected. Set only one bit in EBR2 and EBR1 together (setting more than one bit will automatically clear all EBR1 and EBR2 bits to 0). Bits 7 to 4 are reserved: they are always read as 0 and cannot be modified. When on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

The flash memory block configuration is shown in table 17-6.

**Table 17-6 Flash Memory Erase Blocks**

Block (Size)	Address
EB0 (4 kbytes)	H'000000 to H'000FFF
EB1 (4 kbytes)	H'001000 to H'001FFF
EB2 (4 kbytes)	H'002000 to H'002FFF
EB3 (4 kbytes)	H'003000 to H'003FFF
EB4 (4 kbytes)	H'004000 to H'004FFF
EB5 (4 kbytes)	H'005000 to H'005FFF
EB6 (4 kbytes)	H'006000 to H'006FFF
EB7 (4 kbytes)	H'007000 to H'007FFF
EB8 (32 kbytes)	H'008000 to H'00FFFF
EB9 (64 kbytes)	H'010000 to H'01FFFF
EB10 (64 kbytes)	H'020000 to H'02FFFF
EB11 (64 kbytes)	H'030000 to H'03FFFF

### 17.5.5 System Control Register 2 (SYSCR2)

Bit	:	7	6	5	4	3	2	1	0
		—	—	—	—	FLSHE	—	—	—
Initial value	:	0	0	0	0	0	0	0	0
R/W	:	—	—	—	—	R/W	—	—	—

SYSCR2 is an 8-bit readable/writable register that performs on-chip flash memory control.

SYSCR2 is initialized to H'00 by a reset and in hardware standby mode.

SYSCR2 can only be used in the F-ZTAT version. In the mask ROM version this register will return an undefined value if read, and cannot be modified.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—Flash Memory Control Register Enable (FLSHE):** Controls CPU access to the flash memory control registers (FLMCR1, FLMCR2, EBR1, and EBR2). Writing 1 to the FLSHE bit enables the flash memory control registers to be read and written to. Clearing FLSHE to 0 designates these registers as unselected (the register contents are retained).

Bit 3 FLSHE	Description
0	Flash control registers are not selected for addresses H'FFFFC8 to H'FFFFCB (Initial value)
1	Flash control registers are selected for addresses H'FFFFC8 to H'FFFFCB

**Bits 2 to 0—Reserved:** Read-only bits, always read as 0.

### 17.5.6 RAM Emulation Register (RAMER)

Bit	7	6	5	4	3	2	1	0
	—	—	—	—	RAMS	RAM2	RAM1	RAM0
Initial value :	0	0	0	0	0	0	0	0
R/W :	—	—	—	—	R/W	R/W	R/W	R/W

RAMER specifies the area of flash memory to be overlapped with part of RAM when emulating real-time flash memory programming. RAMER is initialized to H'00 by a power-on reset and in hardware standby mode. It is not initialized in software standby mode. RAMER settings should be made in user mode or user program mode.

Flash memory area divisions are shown in table 17-7. To ensure correct operation of the emulation function, the ROM for which RAM emulation is performed should not be accessed immediately after this register has been modified. Normal execution of an access immediately after register modification is not guaranteed.

**Bits 7 to 4—Reserved:** Read-only bits, always read as 0.

**Bit 3—RAM Select (RAMS):** Specifies selection or non-selection of flash memory emulation in RAM. When RAMS = 1, all flash memory blocks are program/erase-protected.

Bit 3 RAMS	Description
0	Emulation not selected Program/erase-protection of all flash memory blocks is disabled (Initial value)
1	Emulation selected Program/erase-protection of all flash memory blocks is enabled

**Bits 2 to 0—Flash Memory Area Selection (RAM2 to RAM0):** These bits are used together with bit 3 to select the flash memory area to be overlapped with RAM. (See table 17-7.)

**Table 17-7 Flash Memory Area Divisions**

RAM Area	Block Name	RAMS	RAM2	RAM1	RAM0
H'FFDC00 to H'FFEBFF	RAM area, 4 kbytes	0	*	*	*
H'000000 to H'000FFF	EB0 (4 kbytes)	1	0	0	0
H'001000 to H'001FFF	EB1 (4 kbytes)	1	0	0	1
H'002000 to H'002FFF	EB2 (4 kbytes)	1	0	1	0
H'003000 to H'003FFF	EB3 (4 kbytes)	1	0	1	1
H'004000 to H'004FFF	EB4 (4 kbytes)	1	1	0	0
H'005000 to H'005FFF	EB5 (4 kbytes)	1	1	0	1
H'006000 to H'006FFF	EB6 (4 kbytes)	1	1	1	0
H'007000 to H'007FFF	EB7 (4 kbytes)	1	1	1	1

\*: Don't care

## 17.6 On-Board Programming Modes

When pins are set to on-board programming mode, program/erase/verify operations can be performed on the on-chip flash memory. There are two on-board programming modes: boot mode and user program mode. The pin settings for transition to each of these modes are shown in table 17-8. For a diagram of the transitions to the various flash memory modes, see figure 17-3.

**Table 17-8 Setting On-Board Programming Modes**

MCU Mode	Mode	Pins			
		FWE	MD2	MD1	MD0
Boot mode	Advanced expanded mode with on-chip ROM enabled	1	0	1	0
	Advanced single-chip mode				1
User program mode*	Advanced expanded mode with on-chip ROM enabled	1	1	1	0
	Advanced single-chip mode				1

Note: \* Normally, user mode should be used. Set the FWE pin to 1 to make a transition to user program mode before performing a program/erase/verify operation.

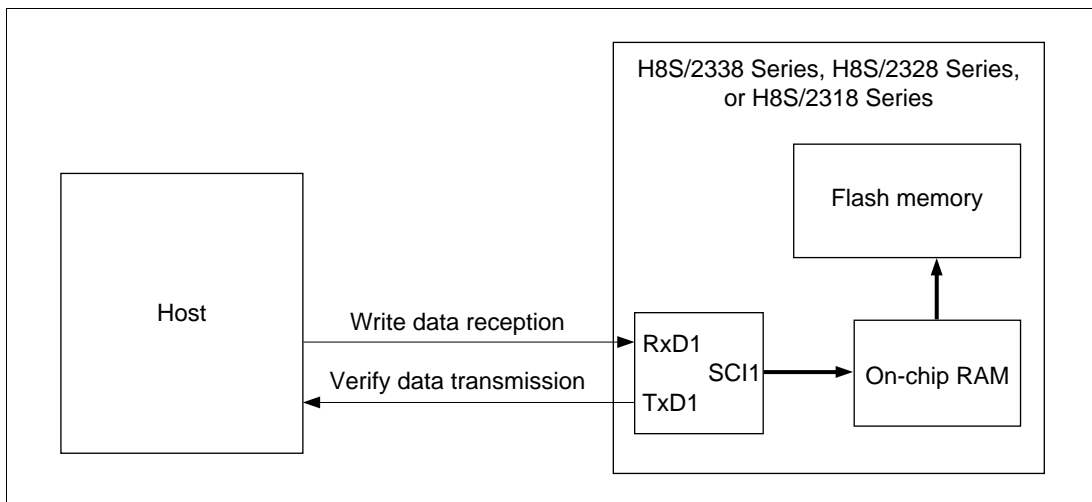
## 17.6.1 Boot Mode

When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. The channel 1 SCI to be used is set to asynchronous mode.

When a reset-start is executed after the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip's pins have been set to boot mode, the boot program built into the chip is started and the programming control program prepared in the host is serially transmitted to the chip via the SCI. In the chip, the programming control program received via the SCI is written into the programming control program area in on-chip RAM. After the transfer is completed, control branches to the start address of the programming control program area and the programming control program execution state is entered (flash memory programming is performed).

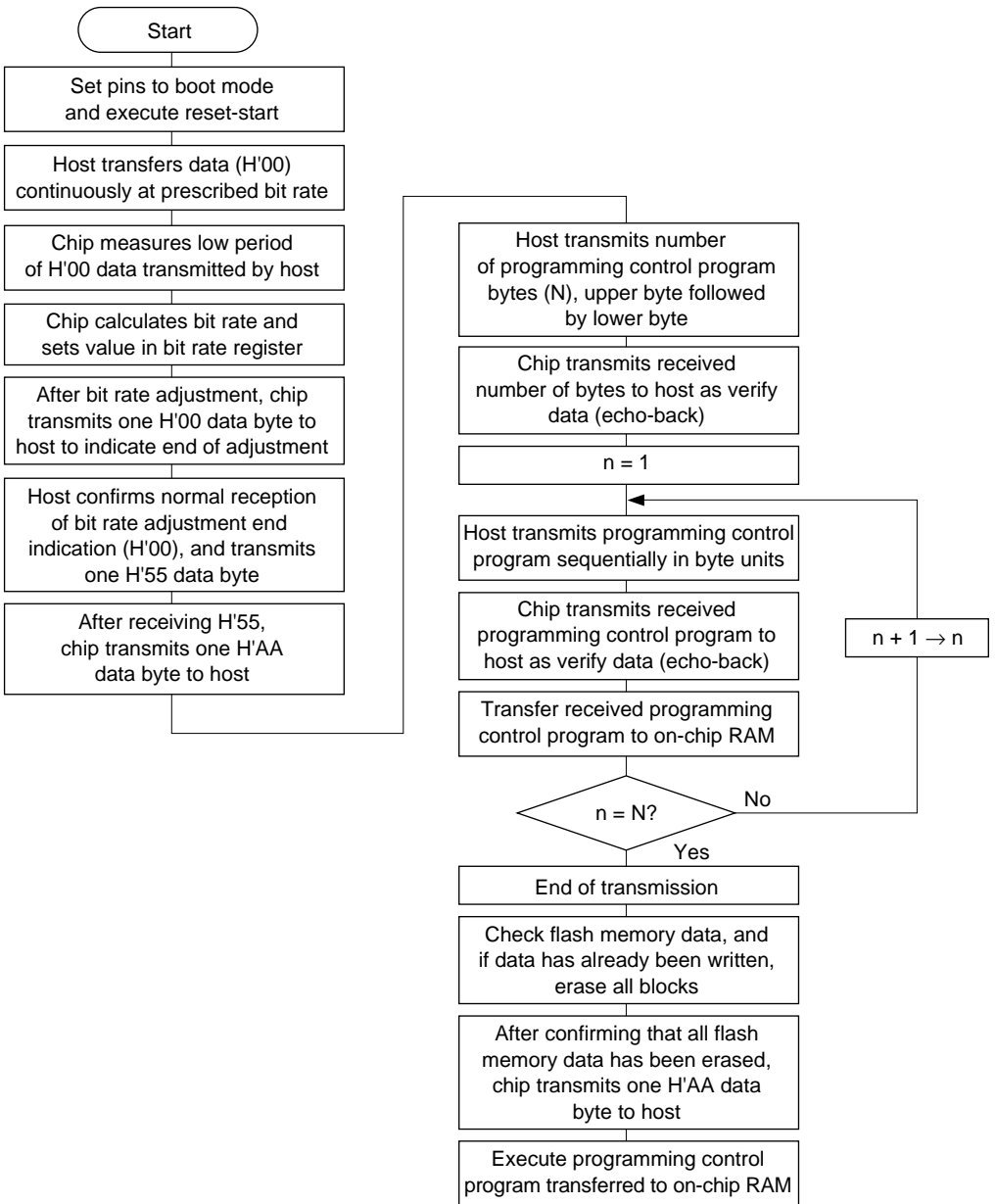
The transferred programming control program must therefore include coding that follows the programming algorithm given later.

The system configuration in boot mode is shown in figure 17-9, and the boot program mode execution procedure in figure 17-10.



**Figure 17-9 System Configuration in Boot Mode**



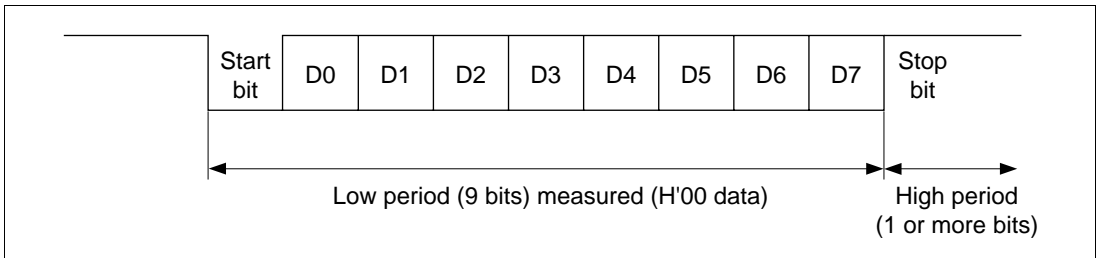


Note: If a memory cell does not operate normally and cannot be erased, one H'FF byte is transmitted as an erase error, and the erase operation and subsequent operations are halted.

**Figure 17-10 Boot Mode Execution Procedure**

**Automatic SCI Bit Rate Adjustment:** When boot mode is initiated, the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip measures the low period of the asynchronous SCI communication data (H'00) transmitted continuously from the host. The SCI transmit/receive format should be set as follows: 8-bit data, 1 stop bit, no parity. The chip calculates the bit rate of the transmission from the host from the measured low period, and transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the chip. If reception cannot be performed normally, initiate boot mode again (reset), and repeat the above operations. Depending on the host's transmission bit rate and the chip's system clock frequency, there will be a discrepancy between the bit rates of the host and the chip. To ensure correct SCI operation, the host's transfer bit rate should be set to 9,600 or 19,200 bps.

Table 17-9 shows typical host transfer bit rates and system clock frequencies for which automatic adjustment of the MCU's bit rate is possible. The boot program should be executed within this system clock range.

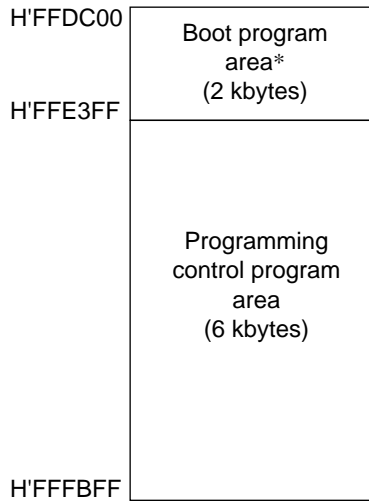


**Figure 17-11 Automatic SCI Bit Rate Adjustment**

**Table 17-9 System Clock Frequencies for which Automatic Adjustment of H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series Bit Rate is Possible**

Host Bit Rate	System Clock Frequency for which Automatic Adjustment of H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series Bit Rate is Possible
19,200 bps	T.B.D.
9,600 bps	T.B.D.

**On-Chip RAM Area Divisions in Boot Mode:** In boot mode, the 2-kbyte area from H'FFDC00 to H'FFE3FF is reserved for use by the boot program, as shown in figure 17-12. The area to which the programming control program is transferred is H'FFE400 to H'FFFBFF. The boot program area can be used when the programming control program transferred into RAM enters the execution state. A stack area should be set up as required.



Note: \* The boot program area cannot be used until a transition is made to the execution state for the programming control program transferred to RAM. Note that the boot program remains stored in this area after a branch is made to the programming control program.

**Figure 17-12 RAM Areas in Boot Mode**

### Notes on Use of Boot Mode

- When the chip comes out of reset in boot mode, it measures the low-level period of the input at the SCI's RxD1 pin. The reset should end with RxD1 high. After the reset ends, it takes approximately 100 states before the chip is ready to measure the low-level period of the RxD1 pin.
- In boot mode, if any data has been programmed into the flash memory (if all data is not 1), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, such as the first time on-board programming is performed, or if the program activated in user program mode is accidentally erased.
- Interrupts cannot be used while the flash memory is being programmed or erased.
- The RxD1 and TxD1 pins should be pulled up on the board.

- Before branching to the programming control program (RAM area H'FFE400 to H'FFFBFF), the chip terminates transmit and receive operations by the on-chip SCI (channel 1) (by clearing the RE and TE bits in SCR to 0), but the adjusted bit rate value remains set in BRR. The transmit data output pin, TxD1, goes to the high-level output state (P31DDR = 1, P31DR = 1). The contents of the CPU's internal general registers are undefined at this time, so these registers must be initialized immediately after branching to the programming control program. In particular, since the stack pointer (SP) is used implicitly in subroutine calls, etc., a stack area must be specified for use by the programming control program. Initial settings must also be made for the other on-chip registers.
- Boot mode can be entered by making the pin settings shown in table 17-8 and executing a reset-start. Boot mode can be cleared by driving the reset pin low, waiting at least 20 states, then setting the FWE pin and mode pins, and executing reset release\*<sup>1</sup>. Boot mode can also be cleared by a WDT overflow reset. Do not change the mode pin input levels in boot mode, and do not drive the FWE pin low while the boot program is being executed or while flash memory is being programmed or erased\*<sup>2</sup>.
- If the mode pin input levels are changed (for example, from low to high) during a reset, the state of ports with multiplexed address functions and bus control output pins ( $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{HWR}$ ) will change according to the change in the microcomputer's operating mode\*<sup>3</sup>. Therefore, care must be taken to make pin settings to prevent these pins from becoming output signal pins during a reset, or to prevent collision with signals outside the microcomputer.

- Notes:
1. Mode pins and FWE pin input must satisfy the mode programming setup time ( $t_{MDS} = 200$  ns) with respect to the reset release timing, as shown in figures 17-29 to 17-31.
  2. For further information on FWE application and disconnection, see section 17.13, Flash Memory Programming and Erasing Precautions.
  3. See the I/O Ports section in the reference manual for the relevant model.

## 17.6.2 User Program Mode

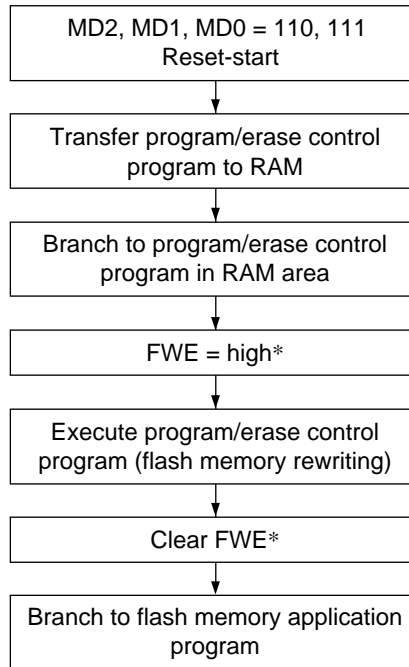
When set to user program mode, the chip can program and erase its flash memory by executing a user program/erase control program. Therefore, on-board reprogramming of the on-chip flash memory can be carried out by providing on-board means of FWE control and supply of programming data, and storing a program/erase control program in part of the program area as necessary.

To select user program mode, select a mode that enables the on-chip flash memory (mode 6 or 7), and apply a high level to the FWE pin. In this mode, on-chip supporting modules other than flash memory operate as they normally would in modes 6 and 7.

The flash memory itself cannot be read while the SWE bit is set to 1 to perform programming or erasing, so the control program that performs programming and erasing should be run in on-chip RAM or external memory.

Figure 17-13 shows the procedure for executing the program/erase control program when transferred to on-chip RAM.

Write the FWE assessment program and transfer program (and the program/erase control program if necessary) beforehand



**Note:** Do not apply a constant high level to the FWE pin. Apply a high level to the FWE pin only when the flash memory is programmed or erased. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

\* For further information on FWE application and disconnection, see section 17.13, Flash Memory Programming and Erasing Precautions.

**Figure 17-13 User Program Mode Execution Procedure**

## 17.7 Programming/Erasing Flash Memory

In the on-board programming modes, flash memory programming and erasing is performed by software, using the CPU. There are four flash memory operating modes: program mode, erase mode, program-verify mode, and erase-verify mode. Transitions to these modes can be made for addresses H'00000 to H'3FFFF by setting the PSU, ESU, P, E, PV, and EV bits in FLMCR1.

The flash memory cannot be read while being programmed or erased. Therefore, the program that controls flash memory programming/erasing (the programming control program) should be located and executed in on-chip RAM or external memory.

- Notes:
1. Operation is not guaranteed if setting/resetting of the SWE, ESU, PSU, EV, PV, E, and P bits in FLMCR1 is executed by a program in flash memory.
  2. When programming or erasing, set FWE to 1 (programming/erasing will not be executed if FWE = 0).
  3. Perform programming in the erased state. Do not perform additional programming on previously programmed addresses.

### 17.7.1 Program Mode

Follow the procedure shown in the program/program-verify flowchart in figure 17-14 to write data or programs to flash memory. Performing program operations according to this flowchart will enable data or programs to be written to flash memory without subjecting the device to voltage stress or sacrificing program data reliability. Programming should be carried out 128 bytes at a time.

For the wait times ( $x$ ,  $y$ ,  $z1$ ,  $z2$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\varepsilon$ ,  $\eta$ ) after bits are set or cleared in flash memory control register 1 (FLMCR1) and the maximum number of programming operations ( $N$ ), see the Flash Memory Characteristics section in the reference manual for the relevant model.

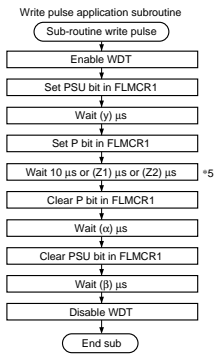
Following the elapse of ( $x$ )  $\mu$ s or more after the SWE bit is set to 1 in flash memory control register 1 (FLMCR1), 128-byte program data is stored in the program data area and reprogram data area, and the 128-byte data in the reprogram data area is written consecutively to the write addresses. The lower 8 bits of the first address written to must be H'00 or H'80. 128 consecutive byte data transfers are performed. The program address and program data are latched in the flash memory. A 128-byte data transfer must be performed even if writing fewer than 128 bytes; in this case, H'FF data must be written to the extra addresses.

Next, the watchdog timer is set to prevent overprogramming in the event of program runaway, etc. Set a value greater than ( $y + z2 + \alpha + \beta$ ) ms as the WDT overflow period. After this, preparation for program mode (program setup) is carried out by setting the PSU bit in FLMCR1, and after the elapse of ( $y$ )  $\mu$ s or more, the operating mode is switched to program mode by setting the P bit in FLMCR1. The time during which the P bit is set is the flash memory programming time. Set the programming time according to the table in the programming flowchart.

## 17.7.2 Program-Verify Mode

In program-verify mode, the data written in program mode is read to check whether it has been correctly written in the flash memory.

After the elapse of a given programming time, the programming mode is exited (the P bit in FLMCR1 is cleared to 0, then the PSU bit is cleared to 0 at least ( $\alpha$ )  $\mu$ s later). Next, the watchdog timer is cleared after the elapse of ( $\beta$ )  $\mu$ s or more, and the operating mode is switched to program-verify mode by setting the PV bit in FLMCR1. Before reading in program-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of ( $\gamma$ )  $\mu$ s or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least ( $\epsilon$ )  $\mu$ s after the dummy write before performing this read operation. Next, the originally written data is compared with the verify data, and reprogram data is computed (see figure 17-14) and transferred to the reprogram data area. After 128 bytes of data have been verified, exit program-verify mode, wait for at least ( $\eta$ )  $\mu$ s, then clear the SWE bit in FLMCR1 to 0. If reprogramming is necessary, set program mode again, and repeat the program/program-verify sequence as before. However, ensure that the program/program-verify sequence is not repeated more than (N) times on the same bits.

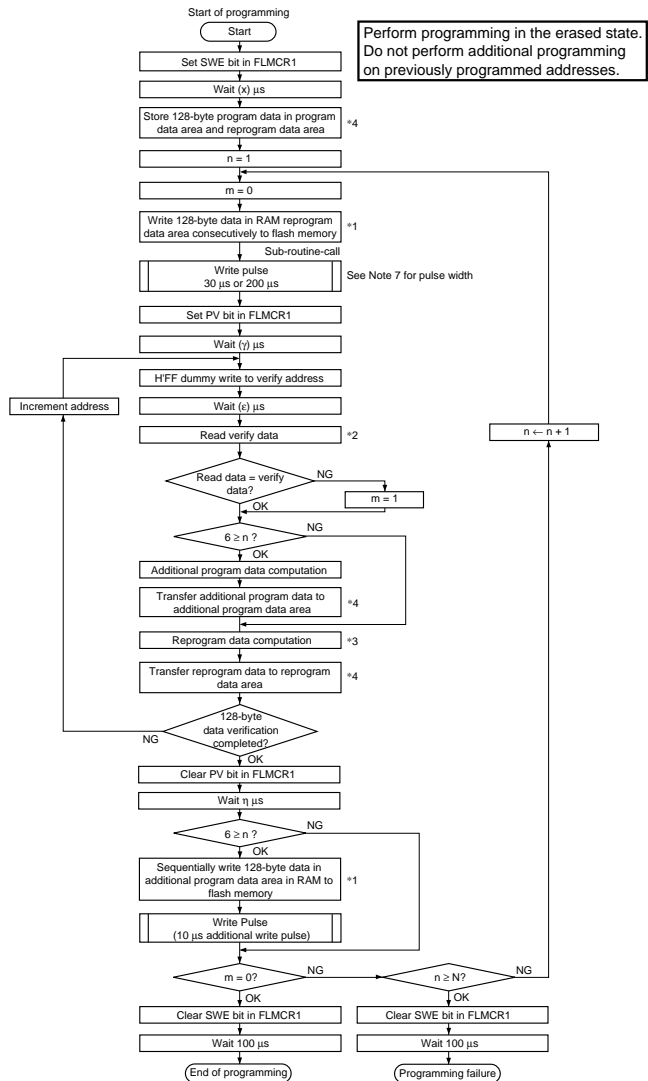


Note 7: Write Pulse Width

Number of Writes (n)	Write Time (z) usec
1	z1
2	z1
3	z1
4	z1
5	z2
6	z2
7	z2
8	z2
9	z2
10	z2
11	z2
12	z2
13	z2
.	.
998	z2
999	z2
1000	z2

Note: Use a 10 μs write pulse for additional programming.

RAM
Program data storage area (128 bytes)
Reprogram data storage area (128 bytes)
Additional program data storage area (128 bytes)



Notes: 1. Data transfer is performed by byte transfer. The lower 8 bits of the first address written to must be H'00 or H'80. A 128-byte data transfer must be performed even if writing fewer than 128 bytes; in this case, HFF data must be written to the extra addresses.

2. Verify data is read in 16-bit (W) units.

3. Even bits for which programming has been completed in the 128-byte p programming loop will be subjected to additional programming if they fail the subsequent verify operation.

4. A 128-byte area for storing program data and a 128-byte area for storing reprogram data must be provided in RAM. The contents of the reprogram data area are modified as programming proceeds.

5. The write pulse varies as programming proceeds. A write pulse of (z1) or (z2) μs should be applied according to the progress of the programming operation. See Note 7 for the pulse widths.

6. For the values of x, y, z1, z2, α, β, γ, ε, η, and N, see the Flash Memory Characteristics section in the reference manual for the relevant model.

Program Data Operation Chart

Original Data (D)	Verify Data (V)	Reprogram Data (X)	Comments
0	0	1	Programming completed
0	1	0	Programming incomplete; reprogram
1	0	1	Still in erased state; no action
1	1	1	

Additional Program Data Operation Chart

Reprogram Data (X)	Verify Data (V)	Additional Program Data (Y)	Comments
0	0	0	Additional programming executed
0	1	1	Additional programming not executed
1	0	0	Additional programming not executed
1	1	1	Additional programming not executed

Figure 17-14 Program/Program-Verify Flowchart



### 17.7.3 Erase Mode

Flash memory erasing should be performed block by block following the procedure shown in the erase/erase-verify flowchart (single-block erase) shown in figure 17-15.

For the wait times ( $x$ ,  $y$ ,  $z$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\epsilon$ ,  $\eta$ ) after bits are set or cleared in flash memory control register 1 (FLMCR1) and the maximum number of programming operations ( $N$ ), see the Flash Memory Characteristics section in the reference manual for the relevant model.

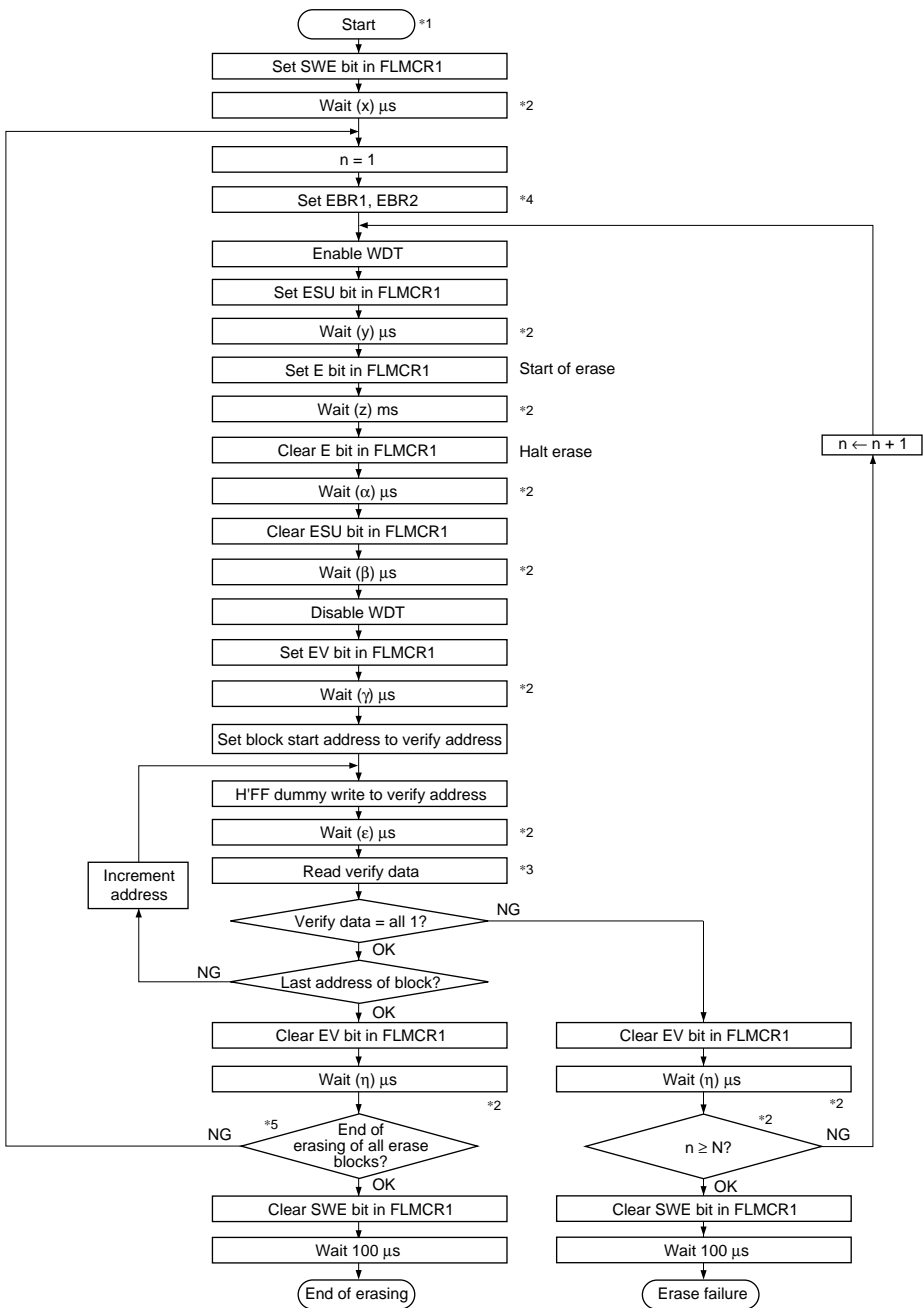
To perform data or program erasure, make a 1 bit setting for the flash memory area to be erased in erase block register 1 or 2 (EBR1 or EBR2) at least ( $x$ )  $\mu$ s after setting the SWE bit to 1 in flash memory control register 1 (FLMCR1). Next, the watchdog timer is set to prevent overerasing in the event of program runaway, etc. Set a value greater than ( $y + z + \alpha + \beta$ ) ms as the WDT overflow period. After this, preparation for erase mode (erase setup) is carried out by setting the ESU bit in FLMCR1, and after the elapse of ( $y$ )  $\mu$ s or more, the operating mode is switched to erase mode by setting the E bit in FLMCR1. The time during which the E bit is set is the flash memory erase time. Ensure that the erase time does not exceed ( $z$ ) ms.

Note: With flash memory erasing, prewriting (setting all data in the memory to be erased to 0) is not necessary before starting the erase procedure.

### 17.7.4 Erase-Verify Mode

In erase-verify mode, data is read after memory has been erased to check whether it has been correctly erased.

After the elapse of the erase time, erase mode is exited (the E bit in FLMCR1 is cleared to 0, then the ESU bit in FLMCR1 is cleared to 0 at least ( $\alpha$ )  $\mu$ s later), the watchdog timer is cleared after the elapse of ( $\beta$ )  $\mu$ s or more, and the operating mode is switched to erase-verify mode by setting the EV bit in FLMCR1. Before reading in erase-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of ( $\gamma$ )  $\mu$ s or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least ( $\epsilon$ )  $\mu$ s after the dummy write before performing this read operation. If the read data has been erased (all 1), a dummy write is performed to the next address, and erase-verify is performed. If the read data has not been erased, set erase mode again, and repeat the erase/erase-verify sequence in the same way. However, ensure that the erase/erase-verify sequence is not repeated more than ( $N$ ) times. When verification is completed, exit erase-verify mode, and wait for at least ( $\eta$ )  $\mu$ s. If erasure has been completed on all the erase blocks, clear the SWE bit in FLMCR1 to 0. If there are any unerased blocks, make a 1 bit setting for the flash memory area to be erased, and repeat the erase/erase-verify sequence in the same way.



- Notes: 1. Prewriting (setting erase block data to all 0) is not necessary.  
 2. The values of x, y, z, α, β, γ, ε, η, and N are shown in the Flash Memory Characteristics section.  
 3. Verify data is read in 16-bit (W) units.  
 4. Set only one bit in EBR1 or EBR2. More than one bit cannot be set.  
 5. Erasing is performed in block units. To erase a number of blocks, the individual blocks must be erased sequentially.

Figure 17-15 Erase/Erase-Verify Flowchart

## 17.8 Flash Memory Protection

There are three kinds of flash memory program/erase protection: hardware protection, software protection, and error protection.

### 17.8.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted. Settings in flash memory control registers 1 and 2 (FLMCR1, FLMCR2) and erase block registers 1 and 2 (EBR1, EBR2) are reset. (See table 17-10.)

**Table 17-10 Hardware Protection**

Item	Description	Functions	
		Program	Erase
FWE pin protection	<ul style="list-style-type: none"><li>When a low level is input to the FWE pin, FLMCR1, FLMCR2, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered.</li></ul>	Yes	Yes
Reset/standby protection	<ul style="list-style-type: none"><li>In a reset (including a WDT overflow reset) and in standby mode, FLMCR1, FLMCR2, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered.</li><li>In a reset via the <math>\overline{\text{RES}}</math> pin, the reset state is not entered unless the <math>\overline{\text{RES}}</math> pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the <math>\overline{\text{RES}}</math> pin low for the <math>\overline{\text{RES}}</math> pulse width specified in the AC Characteristics section.</li></ul>	Yes	Yes

### 17.8.2 Software Protection

Software protection can be implemented by setting the SWE bit in flash memory control register 1 (FLMCR1), erase block registers 1 and 2 (EBR1, EBR2), and the RAMS bit in the RAM emulation register (RAMER). When software protection is in effect, setting the P or E bit in FLMCR1 does not cause a transition to program mode or erase mode. (See table 17-11.)

**Table 17-11 Software Protection**

Item	Description	Functions	
		Program	Erase
SWE bit protection	<ul style="list-style-type: none"> <li>Clearing the SWE bit to 0 in FLMCR1 sets the program/erase-protected state for area H'00000 to H'3FFFF (Execute in on-chip RAM or external memory.)</li> </ul>	Yes	Yes
Block specification protection	<ul style="list-style-type: none"> <li>Erase protection can be set for individual blocks by settings in erase block registers 1 and 2 (EBR1, EBR2).</li> <li>Setting EBR1 and EBR2 to H'00 places all blocks in the erase-protected state.</li> </ul>	—	Yes
Emulation protection	<ul style="list-style-type: none"> <li>Setting the RAMS bit to 1 in the RAM emulation register (RAMER) places all blocks in the program/erase-protected state.</li> </ul>	Yes	Yes

### 17.8.3 Error Protection

In error protection, an error is detected when MCU runaway occurs during flash memory programming/erasing, or operation is not performed in accordance with the program/erase algorithm, and the program/erase operation is aborted. Aborting the program/erase operation prevents damage to the flash memory due to overprogramming or overerasing.

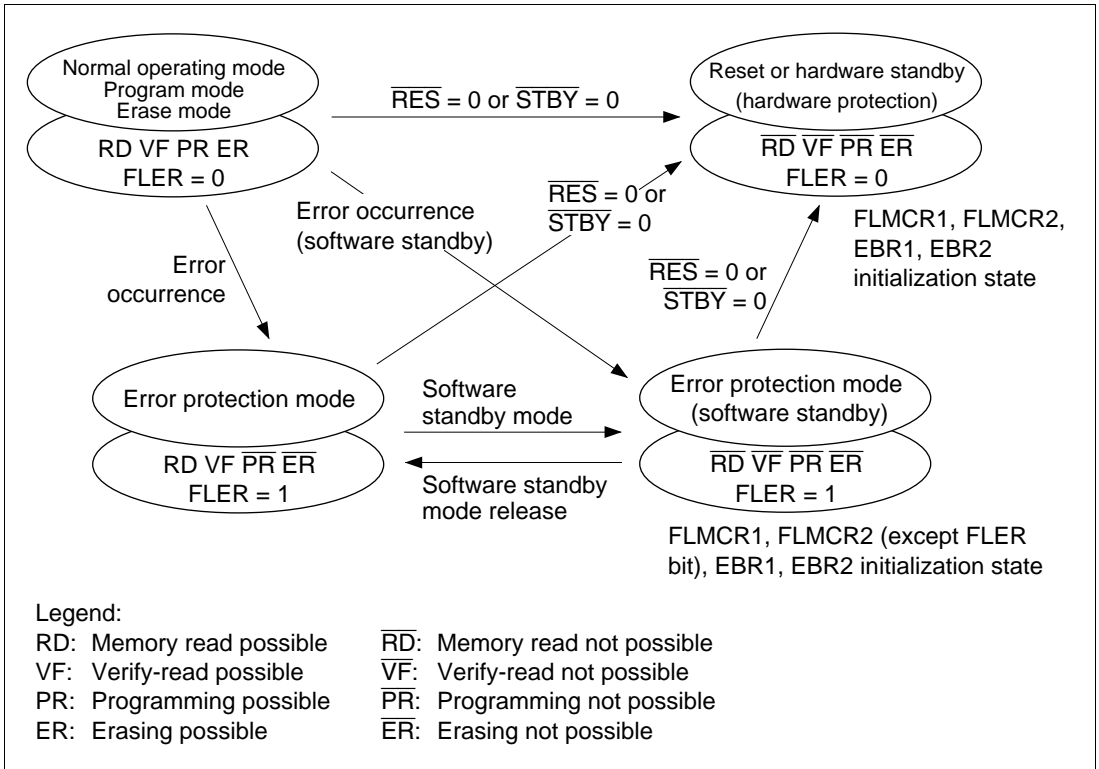
If the MCU malfunctions during flash memory programming/erasing, the FLER bit is set to 1 in FLMCR2 and the error protection state is entered. The FLMCR1, FLMCR2, EBR1, and EBR2 settings are retained, but program mode or erase mode is aborted at the point at which the error occurred. Program mode or erase mode cannot be re-entered by re-setting the P or E bit. However, PV and EV bit setting is enabled, and a transition can be made to verify mode.

FLER bit setting conditions are as follows:

- When flash memory is read during programming/erasing (including a vector read or instruction fetch)
- Immediately after exception handling (excluding a reset) during programming/erasing
- When a SLEEP instruction (including software standby) is executed during programming/erasing
- When the CPU loses the bus during programming/erasing

Error protection is released only by a reset and in hardware standby mode.

Figure 17-16 shows the flash memory state transition diagram.



**Figure 17-16 Flash Memory State Transitions**

## 17.9 Flash Memory Emulation in RAM

### 17.9.1 Emulation in RAM

Making a setting in the RAM emulation register (RAMER) enables part of RAM to be overlapped onto the flash memory area so that data to be written to flash memory can be emulated in RAM in real time. After the RAMER setting has been made, accesses can be made from the flash memory area or the RAM area overlapping flash memory. Emulation can be performed in user mode and user program mode. Figure 17-17 shows an example of emulation of real-time flash memory programming.

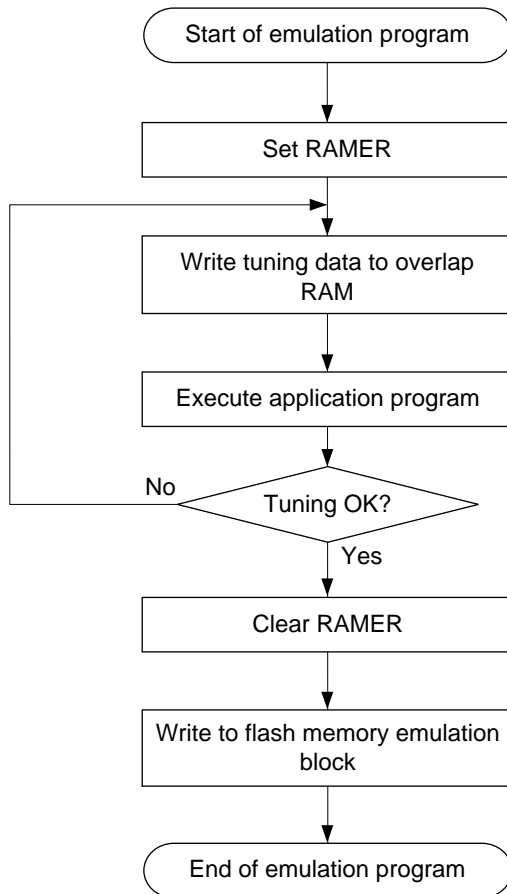
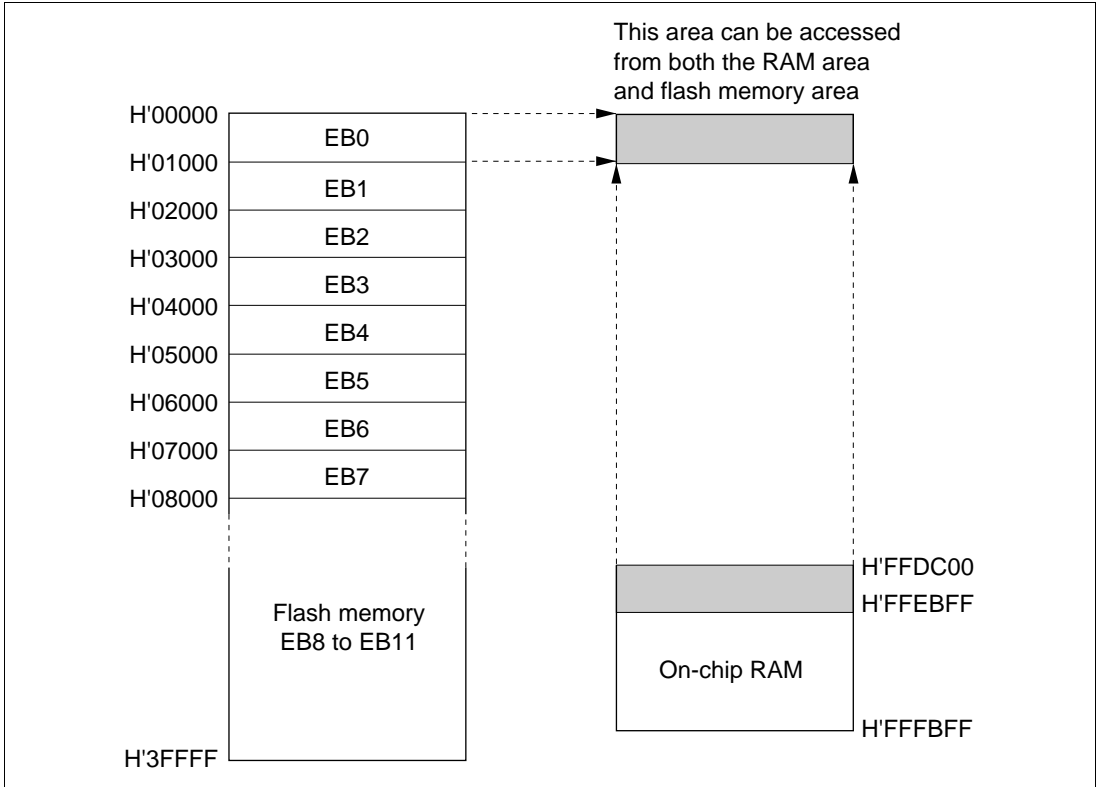


Figure 17-17 Flowchart for Flash Memory Emulation in RAM

## 17.9.2 RAM Overlap

An example in which flash memory block area EB1 is overlapped is shown below.



**Figure 17.18 Example of RAM Overlap Operation**

### Example in Which Flash Memory Block Area EB1 is Overlapped

1. Set bits RAMS, RAM2, RAM1, and RAM0 in RAMER to 1, 0, 0, 1, to overlap part of RAM onto the area (EB1) for which real-time programming is required.
2. Real-time programming is performed using the overlapping RAM.
3. After the program data has been confirmed, the RAMS bit is cleared, releasing RAM overlap.
4. The data written in the overlapping RAM is written into the flash memory space (EB1).

Notes: 1. When the RAMS bit is set to 1, program/erase protection is enabled for all blocks regardless of the value of RAM2, RAM1, and RAM0 (emulation protection). In this state, setting the P or E bit in flash memory control register 1 (FLMCR1) will not cause a transition to program mode or erase mode. When actually programming a flash memory area, the RAMS bit should be cleared to 0.

2. A RAM area cannot be erased by execution of software in accordance with the erase algorithm while flash memory emulation in RAM is being used.
3. Block area EB0 includes the vector table. When performing RAM emulation, the vector table is needed by the overlap RAM.

## 17.10 Interrupt Handling when Programming/Erasing Flash Memory

All interrupts, including NMI input, are disabled when flash memory is being programmed or erased (when the P or E bit is set in FLMCR1), and while the boot program is executing in boot mode\*<sup>1</sup>, to give priority to the program or erase operation. There are three reasons for this:

1. Interrupt during programming or erasing might cause a violation of the programming or erasing algorithm, with the result that normal operation could not be assured.
2. In the interrupt exception handling sequence during programming or erasing, the vector would not be read correctly\*<sup>2</sup>, possibly resulting in MCU runaway.
3. If an interrupt occurred during boot program execution, it would not be possible to execute the normal boot mode sequence.

For these reasons, in on-board programming mode alone there are conditions for disabling interrupts, as an exception to the general rule. However, this provision does not guarantee normal erasing and programming or MCU operation. All requests, including NMI, must therefore be restricted inside and outside the MCU when programming or erasing flash memory. The NMI interrupt is also disabled in the error-protection state while the P or E bit remains set in FLMCR1.

- Notes:
1. Interrupt requests must be disabled inside and outside the MCU until the programming control program has completed programming.
  2. The vector may not be read correctly in this case for the following two reasons:
    - If flash memory is read while being programmed or erased (while the P or E bit is set in FLMCR1), correct read data will not be obtained (undetermined values will be returned).
    - If the interrupt entry in the vector table has not been programmed yet, interrupt exception handling will not be executed correctly.



## 17.11 Flash Memory PROM Mode

### 17.11.1 PROM Mode Setting

Programs and data can be written and erased in PROM mode as well as in the on-board programming modes. In PROM mode, the on-chip ROM can be freely programmed using a PROM programmer that supports the Hitachi microcomputer device type with 256-kbyte on-chip flash memory (FZTAT256V3A). Flash memory read mode, auto-program mode, auto-erase mode, and status read mode are supported with this device type. In auto-program mode, auto-erase mode, and status read mode, a status polling procedure is used, and in status read mode, detailed internal signals are output after execution of an auto-program or auto-erase operation.

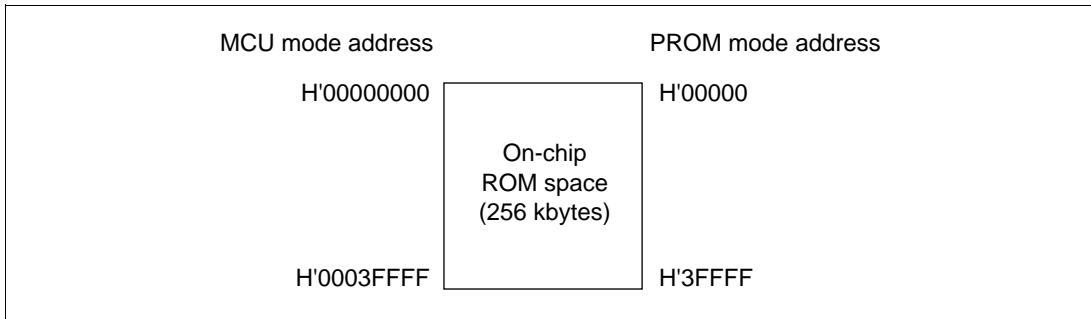
Table 17-12 shows PROM mode pin settings.

**Table 17-12 PROM Mode Pin Settings**

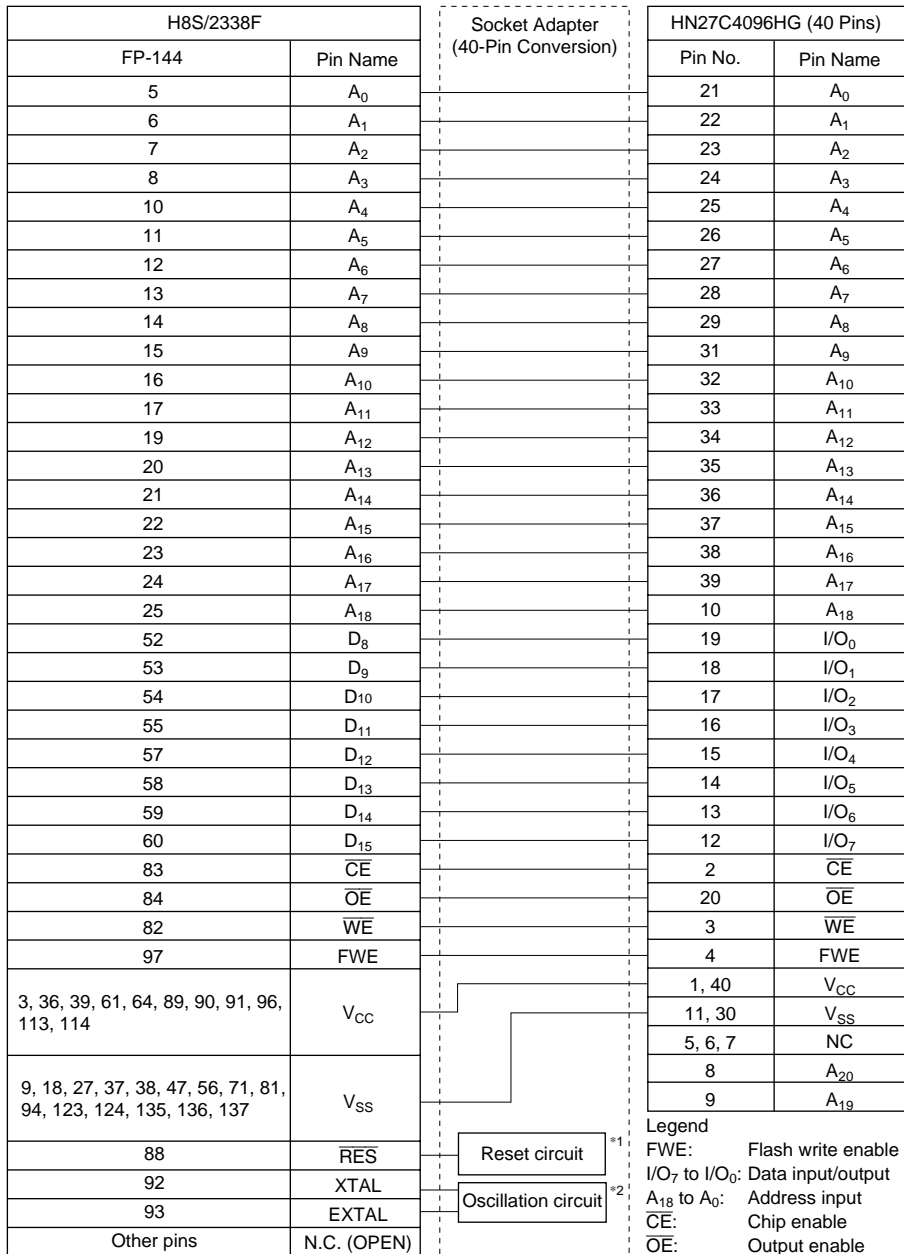
<b>Pin Names</b>	<b>Settings/External Circuit Connection</b>
Mode pins: MD <sub>2</sub> , MD <sub>1</sub> , MD <sub>0</sub>	Low-level input
Mode setting pins: P66, P65, P64	High-level input to P66, low-level input to P65 and P64
FWE pin	High-level input (in auto-program and auto-erase modes)
$\overline{\text{STBY}}$ pin	High-level input (do not select hardware standby mode)
$\overline{\text{RES}}$ pin	Reset circuit
XTAL, EXTAL pins	Oscillator circuit
Other pins requiring setting: P32, P25	High-level input to P32, low-level input to P25

### 17.11.2 Socket Adapters and Memory Map

In PROM mode, a socket adapter is connected to the chip as shown in figures 17-20 and 17-21. Figure 17-19 shows the on-chip ROM memory map and figures 17-20 and 17-21 show the socket adapter pin assignments.



**Figure 17-19 Memory Map in PROM Mode**

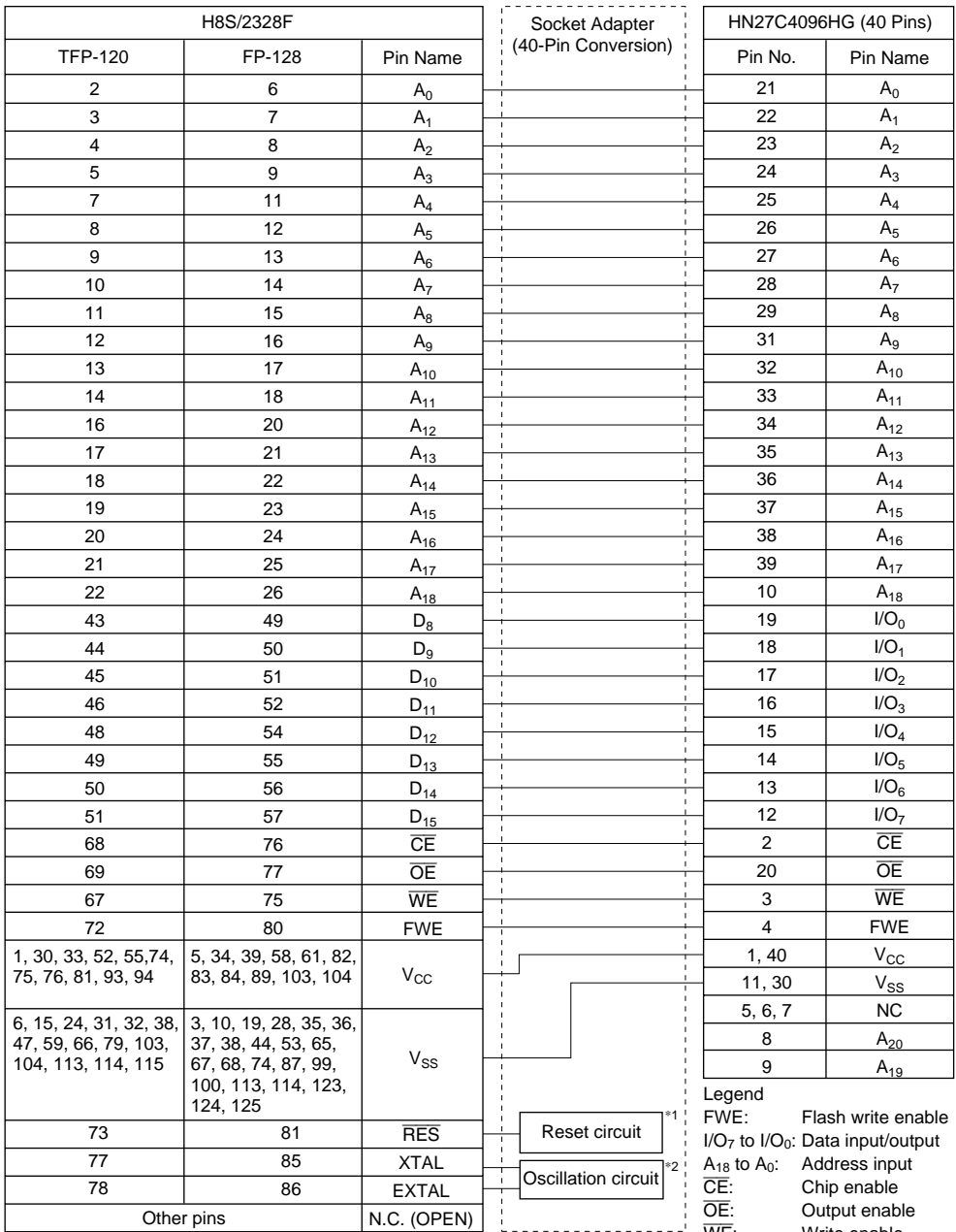


Legend  
FWE: Flash write enable  
I/O<sub>7</sub> to I/O<sub>0</sub>: Data input/output  
A<sub>18</sub> to A<sub>0</sub>: Address input  
 $\overline{CE}$ : Chip enable  
 $\overline{OE}$ : Output enable  
 $\overline{WE}$ : Write enable

- Notes: 1. A reset oscillation stabilization time ( $t_{osc1}$ ) of at least 10 ms is required.  
2. A 12 MHz crystal resonator should be used.

This figure shows pin assignments, and does not show the entire socket adapter circuit.

**Figure 17-20 H8S/2338F Socket Adapter Pin Assignments**



Legend  
FWE: Flash write enable  
I/O<sub>7</sub> to I/O<sub>0</sub>: Data input/output  
A<sub>18</sub> to A<sub>0</sub>: Address input  
 $\overline{CE}$ : Chip enable  
 $\overline{OE}$ : Output enable  
 $\overline{WE}$ : Write enable

- Notes: 1. A reset oscillation stabilization time ( $t_{OSC1}$ ) of at least 10 ms is required.  
2. A 12 MHz crystal resonator should be used.

This figure shows pin assignments, and does not show the entire socket adapter circuit.

**Figure 17-21 H8S/2328F Socket Adapter Pin Assignments**

### 17.11.3 PROM Mode Operation

Table 17-13 shows how the different operating modes are set when using PROM mode, and table 17-14 lists the commands used in PROM mode. Details of each mode are given below.

**Memory Read Mode:** Memory read mode supports byte reads.

**Auto-Program Mode:** Auto-program mode supports programming of 128 bytes at a time. Status polling is used to confirm the end of auto-programming.

**Auto-Erase Mode:** Auto-erase mode supports automatic erasing of the entire flash memory. Status polling is used to confirm the end of auto-erasing.

**Status Read Mode:** Status polling is used for auto-programming and auto-erasing, and normal termination can be confirmed by reading the I/O<sub>6</sub> signal. In status read mode, error information is output if an error occurs.

**Table 17-13 Settings for Each Operating Mode in PROM Mode**

Mode	Pin Names					
	FWE	CE	OE	WE	I/O <sub>7</sub> to I/O <sub>0</sub>	A <sub>18</sub> to A <sub>0</sub>
Read	H or L	L	L	H	Data output	Ain
Output disable	H or L	L	H	H	Hi-z	X
Command write	H or L <sup>*3</sup>	L	H	L	Data input	Ain <sup>*2</sup>
Chip disable <sup>*1</sup>	H or L	H	X	X	Hi-z	X

Legend:

H: High level

L: Low level

Hi-z: High impedance

X: Don't care

- Notes:
1. Chip disable is not a standby state; internally, it is an operation state.
  2. Ain indicates that there is also address input in auto-program mode.
  3. For command writes when making a transition to auto-program or auto-erase mode, input a high level to the FWE pin.

**Table 17-14 PROM Mode Commands**

Command Name	Number of Cycles	1st Cycle			2nd Cycle		
		Mode	Address	Data	Mode	Address	Data
Memory read mode	1 + n	Write	X	H'00	Read	RA	Dout
Auto-program mode	129	Write	X	H'40	Write	PA	Din
Auto-erase mode	2	Write	X	H'20	Write	X	H'20
Status read mode	2	Write	X	H'71	Write	X	H'71

Legend:

RA: Read address

PA: Program address

- Notes:
1. In auto-program mode, 129 cycles are required for command writing by a simultaneous 128-byte write.
  2. In memory read mode, the number of cycles depends on the number of address write cycles (n).

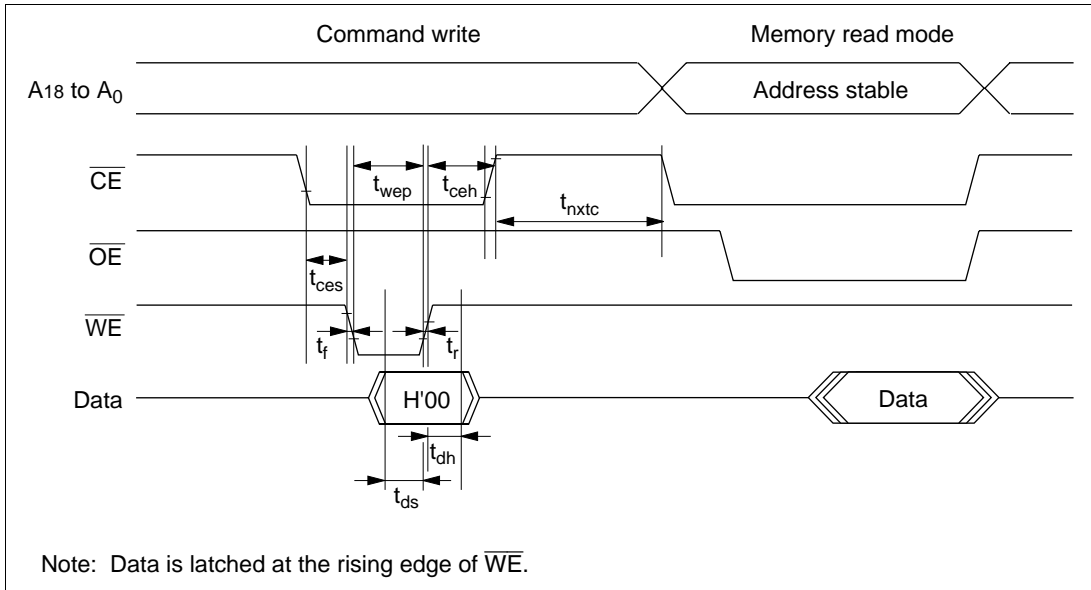
#### 17.11.4 Memory Read Mode

- After the end of an auto-program, auto-erase, or status read operation, the command wait state is entered. To read memory contents, a transition must be made to memory read mode by means of a command write before the read is executed.
- Command writes can be performed in memory read mode, just as in the command wait state.
- Once memory read mode has been entered, consecutive reads can be performed.
- After power-on, memory read mode is entered.

**Table 17-15 AC Characteristics in Memory Read Mode**

(Conditions:  $V_{CC} = 3.3\text{ V} \pm 0.3\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

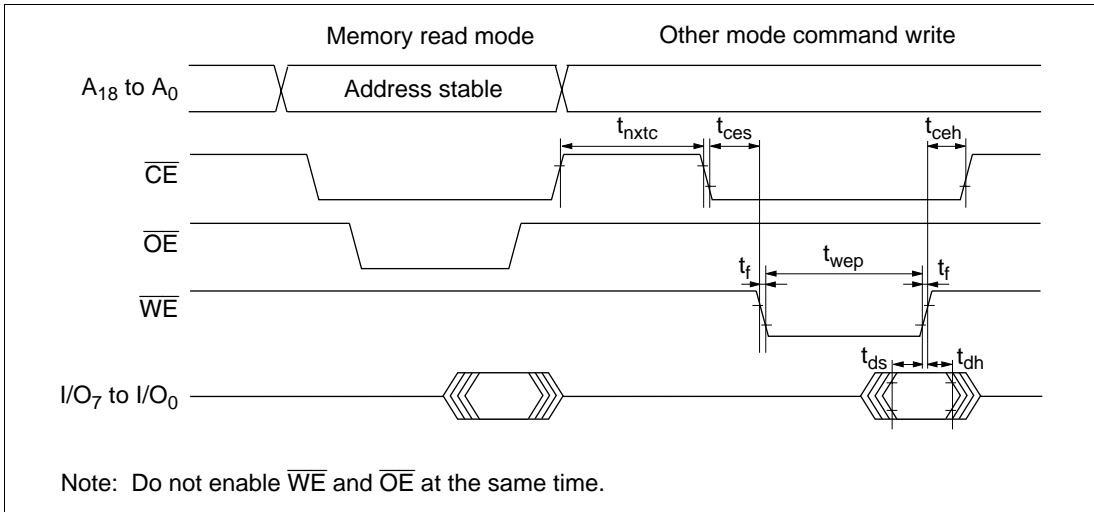
Item	Symbol	Min	Max	Unit	Notes
Command write cycle	$t_{nxtc}$	20		$\mu\text{s}$	
$\overline{\text{CE}}$ hold time	$t_{ceh}$	0		ns	
$\overline{\text{CE}}$ setup time	$t_{ces}$	0		ns	
Data hold time	$t_{dh}$	50		ns	
Data setup time	$t_{ds}$	50		ns	
Write pulse width	$t_{wep}$	70		ns	
$\overline{\text{WE}}$ rise time	$t_r$		30	ns	
$\overline{\text{WE}}$ fall time	$t_f$		30	ns	



**Figure 17-22 Memory Read Mode Timing Waveforms after Command Write**

**Table 17-16 AC Characteristics when Entering Another Mode from Memory Read Mode**  
 (Conditions:  $V_{CC} = 3.3\text{ V} \pm 0.3\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

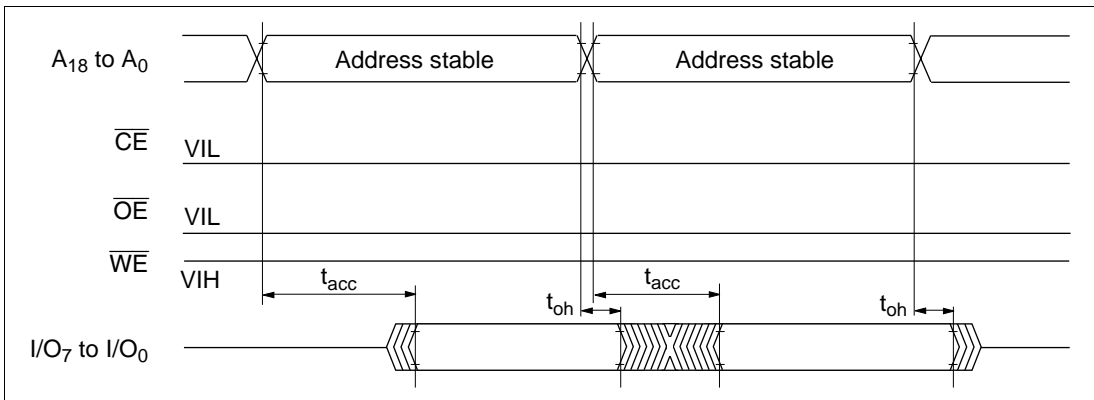
Item	Symbol	Min	Max	Unit	Notes
Command write cycle	$t_{nxtc}$	20		$\mu\text{s}$	
$\overline{CE}$ hold time	$t_{ceh}$	0		ns	
$\overline{CE}$ setup time	$t_{ces}$	0		ns	
Data hold time	$t_{dh}$	50		ns	
Data setup time	$t_{ds}$	50		ns	
Write pulse width	$t_{wep}$	70		ns	
$\overline{WE}$ rise time	$t_r$		30	ns	
$\overline{WE}$ fall time	$t_f$		30	ns	



**Figure 17-23 Timing Waveforms when Entering Another Mode from Memory Read Mode**

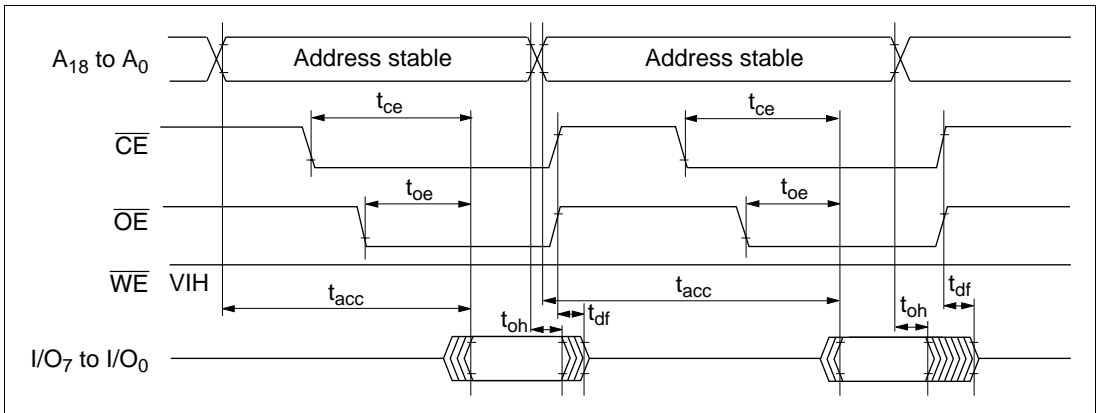
**Table 17-17 AC Characteristics in Memory Read Mode**  
 (Conditions:  $V_{CC} = 3.3\text{ V} \pm 0.3\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Max	Unit	Notes
Access time	$t_{acc}$		20	$\mu\text{s}$	
$\overline{CE}$ output delay time	$t_{ce}$		150	ns	
$\overline{OE}$ output delay time	$t_{oe}$		150	ns	
Output disable delay time	$t_{df}$		100	ns	
Data output hold time	$t_{oh}$	5		ns	



**Figure 17-24 Timing Waveforms for  $\overline{CE}/\overline{OE}$  Enable State Read**





**Figure 17-25 Timing Waveforms for  $\overline{\text{CE}}/\overline{\text{OE}}$  Clocked Read**

### 17.11.5 Auto-Program Mode

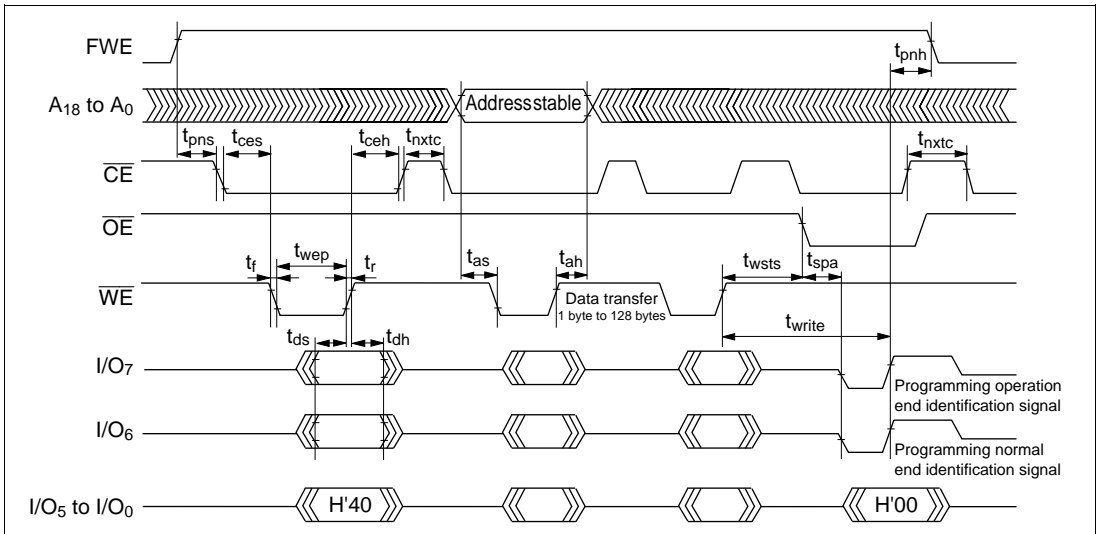
- In auto-program mode, 128 bytes are programmed simultaneously. For this purpose, 128 consecutive byte data transfers should be performed.
- A 128-byte data transfer must be performed even if writing fewer than 128 bytes; in this case, H'FF data must be written to the extra addresses.
- The lower 7 bits of the transfer address must be held low. If an invalid address is input, memory programming will be started but a programming error will occur.
- Memory address transfer is executed in the second cycle (figure 17-26). Do not perform transfer later than the second cycle.
- Do not perform a command write during a programming operation.
- Perform one auto-programming operation for a 128-byte block for each address. One or more additional programming operations cannot be carried out on address blocks that have already been programmed.
- Confirm normal end of auto-programming by checking I/O<sub>6</sub>. Alternatively, status read mode can also be used for this purpose (the I/O<sub>7</sub> status polling pin is used to identify the end of an auto-program operation).
- Status polling I/O<sub>6</sub> and I/O<sub>7</sub> information is retained until the next command write. As long as the next command write has not been performed, reading is possible by enabling  $\overline{\text{CE}}$  and  $\overline{\text{OE}}$ .

## AC Characteristics

**Table 17-18 AC Characteristics in Auto-Program Mode**

(Conditions:  $V_{CC} = 3.3\text{ V} \pm 0.3\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	$t_{nxtc}$	20		$\mu\text{s}$	
$\overline{\text{CE}}$ hold time	$t_{ceh}$	0		ns	
$\overline{\text{CE}}$ setup time	$t_{ces}$	0		ns	
Data hold time	$t_{dh}$	50		ns	
Data setup time	$t_{ds}$	50		ns	
Write pulse width	$t_{wep}$	70		ns	
Status polling start time	$t_{wsts}$	1		ms	
Status polling access time	$t_{spa}$		150	ns	
Address setup time	$t_{as}$	0		ns	
Address hold time	$t_{ah}$	60		ns	
Memory write time	$t_{write}$	1	3000	ms	
$\overline{\text{WE}}$ rise time	$t_r$		30	ns	
$\overline{\text{WE}}$ fall time	$t_f$		30	ns	
Write setup time	$t_{pns}$	100		ns	
Write end setup time	$t_{pnh}$	100		ns	



**Figure 17-26 Auto-Program Mode Timing Waveforms**

## 17.11.6 Auto-Erase Mode

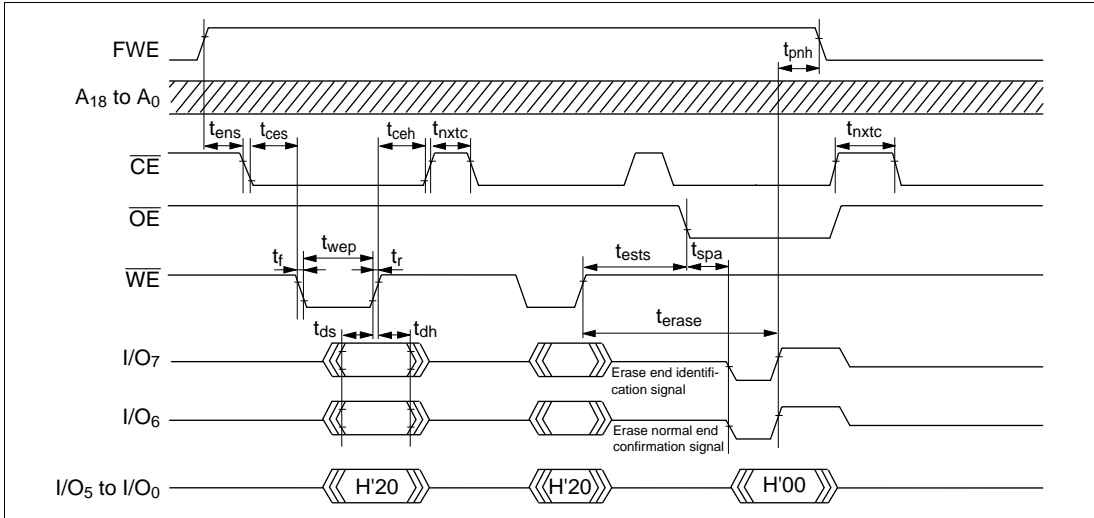
- Auto-erase mode supports only total memory erasing.
- Do not perform a command write during auto-erasing.
- Confirm normal end of auto-erasing by checking I/O<sub>6</sub>. Alternatively, status read mode can also be used for this purpose (the I/O<sub>7</sub> status polling pin is used to identify the end of an auto-erase operation).
- Status polling I/O<sub>6</sub> and I/O<sub>7</sub> pin information is retained until the next command write. As long as the next command write has not been performed, reading is possible by enabling  $\overline{\text{CE}}$  and  $\overline{\text{OE}}$ .

### AC Characteristics

**Table 17-19 AC Characteristics in Auto-Erase Mode**

(Conditions:  $V_{CC} = 3.3 \text{ V} \pm 0.3 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	$t_{\text{nxtc}}$	20		$\mu\text{s}$	
$\overline{\text{CE}}$ hold time	$t_{\text{ceh}}$	0		ns	
$\overline{\text{CE}}$ setup time	$t_{\text{ces}}$	0		ns	
Data hold time	$t_{\text{dh}}$	50		ns	
Data setup time	$t_{\text{ds}}$	50		ns	
Write pulse width	$t_{\text{wep}}$	70		ns	
Status polling start time	$t_{\text{ests}}$	1		ms	
Status polling access time	$t_{\text{s pa}}$		150	ns	
Memory erase time	$t_{\text{erase}}$	100	40000	ms	
$\overline{\text{WE}}$ rise time	$t_{\text{r}}$		30	ns	
$\overline{\text{WE}}$ fall time	$t_{\text{f}}$		30	ns	
Erase setup time	$t_{\text{ens}}$	100		ns	
Erase end setup time	$t_{\text{enh}}$	100		ns	



**Figure 17-27 Auto-Erase Mode Timing Waveforms**

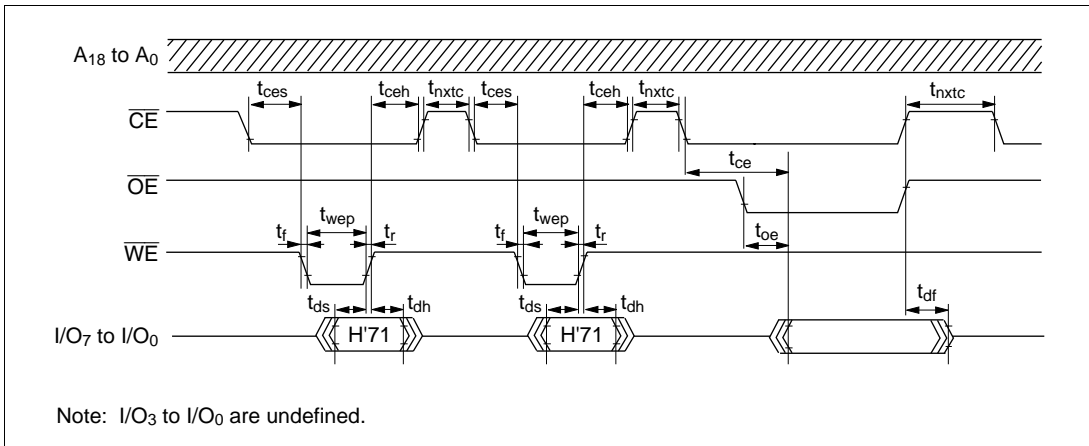
### 17.11.7 Status Read Mode

- Status read mode is used to identify what type of abnormal end has occurred. Use this mode when an abnormal end occurs in auto-program mode or auto-erase mode.
- The return code is retained until a command write for other than status read mode is performed.

**Table 17-20 AC Characteristics in Status Read Mode**

(Conditions:  $V_{CC} = 3.3 \text{ V} \pm 0.3 \text{ V}$ ,  $V_{SS} = 0 \text{ V}$ ,  $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$ )

Item	Symbol	Min	Max	Unit	Notes
Command write cycle	$t_{nxtc}$	20		$\mu\text{s}$	
$\overline{\text{CE}}$ hold time	$t_{ceh}$	0		ns	
$\overline{\text{CE}}$ setup time	$t_{ces}$	0		ns	
Data hold time	$t_{dh}$	50		ns	
Data setup time	$t_{ds}$	50		ns	
Write pulse width	$t_{wep}$	70		ns	
$\overline{\text{OE}}$ output delay time	$t_{oe}$		150	ns	
Disable delay time	$t_{df}$		100	ns	
$\overline{\text{CE}}$ output delay time	$t_{ce}$		150	ns	
$\overline{\text{WE}}$ rise time	$t_r$		30	ns	
$\overline{\text{WE}}$ fall time	$t_f$		30	ns	



**Figure 17-28 Status Read Mode Timing Waveforms**

**Table 17-21 Status Read Mode Return Commands**

Pin Name	I/O <sub>7</sub>	I/O <sub>6</sub>	I/O <sub>5</sub>	I/O <sub>4</sub>	I/O <sub>3</sub>	I/O <sub>2</sub>	I/O <sub>1</sub>	I/O <sub>0</sub>
Attribute	Normal end identification	Command error	Programming error	Erase error	—	—	Programming or erase count exceeded	Effective address error
Initial value	0	0	0	0	0	0	0	0
Indications	Normal end: 0 Abnormal end: 1	Command error: 1 Otherwise: 0	Programming error: 1 Otherwise: 0	Erase error: 1 Otherwise: 0	—	—	Count exceeded: 1 Otherwise: 0	Effective address error: 1 Otherwise: 0

Note: I/O<sub>2</sub> and I/O<sub>3</sub> are undefined.

### 17.11.8 Status Polling

- The I/O<sub>7</sub> status polling flag indicates the operating status in auto-program or auto-erase mode.
- The I/O<sub>6</sub> status polling flag indicates a normal or abnormal end in auto-program or auto-erase mode.

**Table 17-22 Status Polling Output Truth Table**

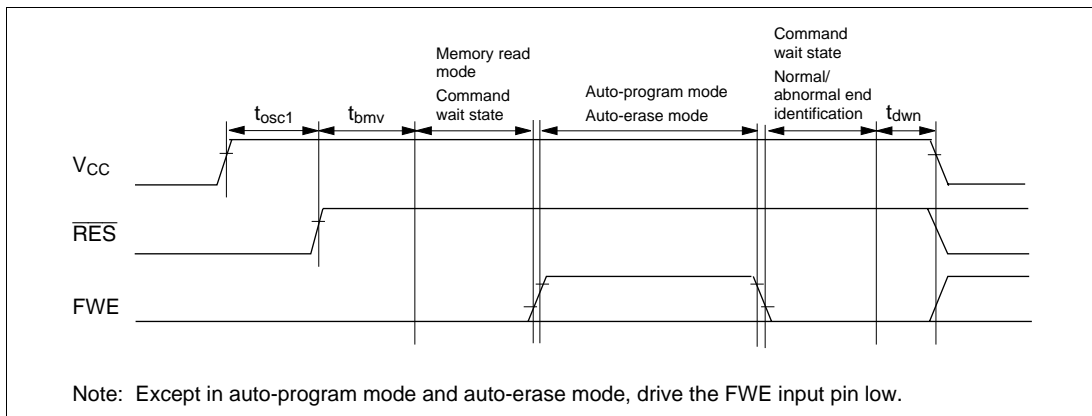
Pin Names	Internal Operation in Progress	Abnormal End	—	Normal End
I/O <sub>7</sub>	0	1	0	1
I/O <sub>6</sub>	0	0	1	1
I/O <sub>0</sub> to I/O <sub>5</sub>	0	0	0	0

### 17.11.9 PROM Mode Transition Time

Commands cannot be accepted during the oscillation stabilization period or the PROM mode setup period. After the PROM mode setup time, a transition is made to memory read mode.

**Table 17-23 Command Wait State Transition Time Specifications**

Item	Symbol	Min	Max	Unit	Notes
Standby release (oscillation stabilization time)	$t_{osc1}$	30	—	ms	
PROM mode setup time	$t_{bmv}$	10	—	ms	
$V_{CC}$ hold time	$t_{dwn}$	0	—	ms	



**Figure 17-29 Oscillation Stabilization Time, Writer Mode Setup Time, and Power Supply Fall Sequence**

### 17.11.10 Notes On Memory Programming

- When programming addresses which have previously been programmed, carry out auto-erasing before auto-programming.
- When performing programming using writer mode on a chip that has been programmed/erased in an on-board programming mode, auto-erasing is recommended before carrying out auto-programming.

- Notes:
1. The flash memory is initially in the erased state when the device is shipped by Hitachi. For other chips for which the erasure history is unknown, it is recommended that auto-erasing be executed to check and supplement the initialization (erase) level.
  2. Auto-programming should be performed once only on the same address block. Additional programming cannot be carried out on address blocks that have already been programmed.

## 17.12 Flash Memory Programming and Erasing Precautions

Precautions concerning the use of on-board programming mode, the RAM emulation function, and PROM mode are summarized below.

**Use the specified voltages and timing for programming and erasing:** Applied voltages in excess of the rating can permanently damage the device. Use a PROM programmer that supports the Hitachi microcomputer device type with 256-kbyte on-chip flash memory (FZTAT256V3A).

Do not select the HN27C4096 setting for the PROM programmer, and only use the specified socket adapter. Failure to observe these points may result in damage to the device.

**Powering on and off (see figures 17-30 to 17-32):** Do not apply a high level to the FWE pin until  $V_{CC}$  has stabilized. Also, drive the FWE pin low before turning off  $V_{CC}$ .

When applying or disconnecting  $V_{CC}$  power, fix the FWE pin low and place the flash memory in the hardware protection state.

The power-on and power-off timing requirements should also be satisfied in the event of a power failure and subsequent recovery.

**FWE application/disconnection (see figures 17-30 to 17-32):** FWE application should be carried out when MCU operation is in a stable condition. If MCU operation is not stable, fix the FWE pin low and set the protection state.

The following points must be observed concerning FWE application and disconnection to prevent unintentional programming or erasing of flash memory:

- Apply FWE when the  $V_{CC}$  voltage has stabilized within its rated voltage range.  
Apply FWE when oscillation has stabilized (after the elapse of the oscillation stabilization time).
- In boot mode, apply and disconnect FWE during a reset.
- In user program mode, FWE can be switched between high and low level regardless of the reset state. FWE input can also be switched during execution of a program in flash memory.
- Do not apply FWE if program runaway has occurred.
- Disconnect FWE only when the SWE, ESU, PSU, EV, PV, P, and E bits in FLMCR1 are cleared.

Make sure that the SWE, ESU, PSU, EV, PV, P, and E bits are not set by mistake when applying or disconnecting FWE.

**Do not apply a constant high level to the FWE pin:** Apply a high level to the FWE pin only when programming or erasing flash memory. A system configuration in which a high level is constantly applied to the FWE pin should be avoided. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

**Use the recommended algorithm when programming and erasing flash memory:** The recommended algorithm enables programming and erasing to be carried out without subjecting the device to voltage stress or sacrificing program data reliability. When setting the P or E bit in FLMCR1, the watchdog timer should be set beforehand as a precaution against program runaway, etc.

**Do not set or clear the SWE bit during execution of a program in flash memory:** Wait for at least 100  $\mu$ s after clearing the SWE bit before executing a program or reading data in flash memory. When the SWE bit is set, data in flash memory can be rewritten, but when SWE = 1, flash memory can only be read in program-verify or erase-verify mode. Access flash memory only for verify operations (verification during programming/erasing). Also, do not clear the SWE bit during programming, erasing, or verifying.

Similarly, when using the RAM emulation function while a high level is being input to the FWE pin, the SWE bit must be cleared before executing a program or reading data in flash memory.

However, the RAM area overlapping flash memory space can be read and written to regardless of whether the SWE bit is set or cleared.

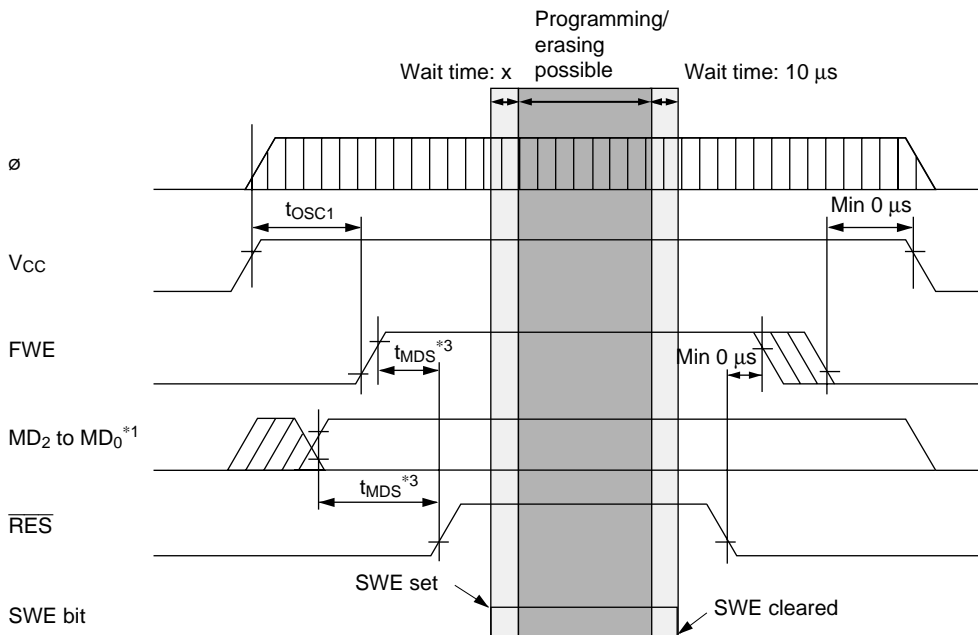
**Do not use interrupts while flash memory is being programmed or erased:** All interrupt requests, including NMI, should be disabled during FWE application to give priority to program/erase operations.

**Do not perform additional programming. Erase the memory before reprogramming:** In on-board programming, perform only one programming operation on a 128-byte programming unit block. In PROM mode, too, perform only one programming operation on a 128-byte programming unit block. Programming should be carried out with the entire programming unit block erased.

**Before programming, check that the chip is correctly mounted in the PROM programmer:** Overcurrent damage to the device can result if the index marks on the PROM programmer socket, socket adapter, and chip are not correctly aligned.

**Do not touch the socket adapter or chip during programming:** Touching either of these can cause contact faults and write errors.



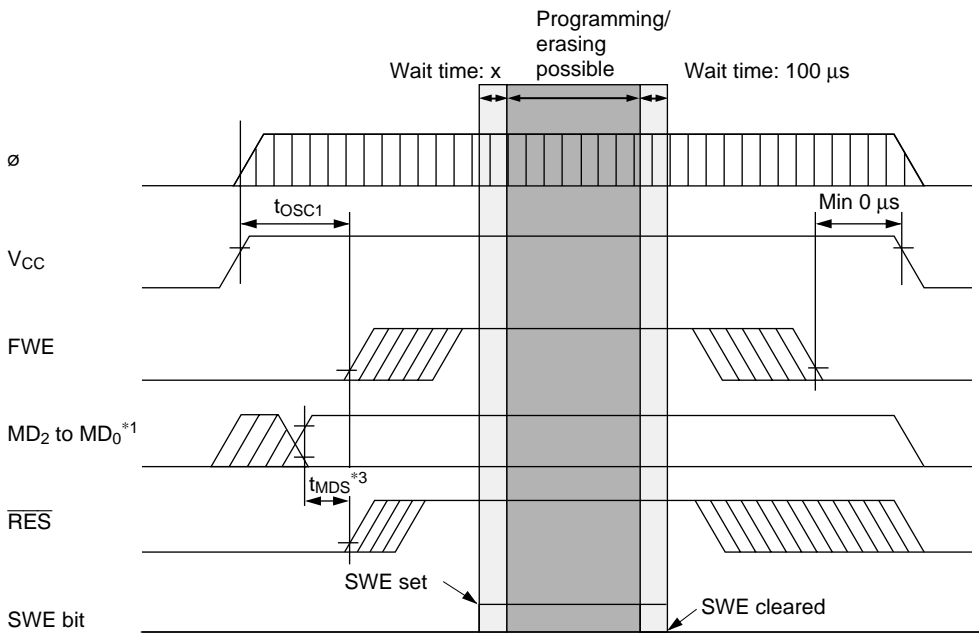


□ Period during which flash memory access is prohibited (x: Wait time after setting SWE bit)<sup>\*2</sup>

■ Period during which flash memory can be programmed (Execution of program in flash memory prohibited, and data reads other than verify operations prohibited)

- Notes:
1. Except when switching modes, the level of the mode pins ( $MD_2$  to  $MD_0$ ) must be fixed until power-off by pulling the pins up or down.
  2. See the Flash Memory Characteristics section in the reference manual for the relevant model.
  3. Mode programming setup time  $t_{MDS}$  (min) = 200 ns

**Figure 17-30 Power-On/Off Timing (Boot Mode)**

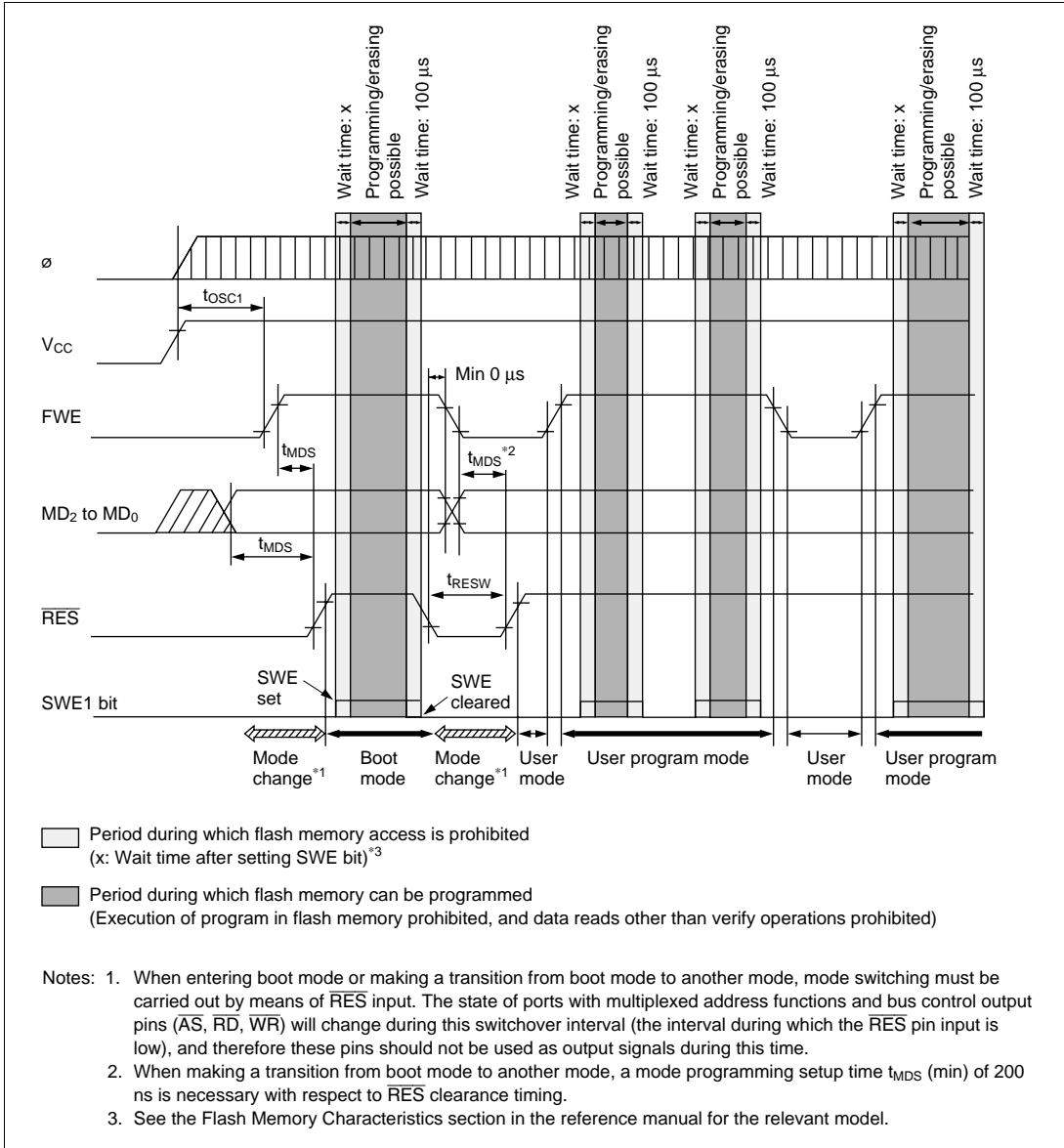


□ Period during which flash memory access is prohibited (x: Wait time after setting SWE bit)<sup>\*2</sup>

■ Period during which flash memory can be programmed (Execution of program in flash memory prohibited, and data reads other than verify operations prohibited)

- Notes:
1. Except when switching modes, the level of the mode pins (MD<sub>2</sub> to MD<sub>0</sub>) must be fixed until power-off by pulling the pins up or down.
  2. See the Flash Memory Characteristics section in the reference manual for the relevant model.
  3. Mode programming setup time  $t_{MDS}$  (min) = 200 ns

**Figure 17-31 Power-On/Off Timing (User Program Mode)**



**Figure 17-32 Mode Transition Timing**  
**(Example: Boot Mode → User Mode ↔ User Program Mode)**

# Section 18 Clock Pulse Generator

## 18.1 Overview

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have an on-chip clock pulse generator (CPG) that generates the system clock ( $\phi$ ), the bus master clock, and internal clocks.

The clock pulse generator consists of an oscillator circuit, a duty adjustment circuit, a medium-speed clock divider, and a bus master clock selection circuit.

In the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series, the CPG has a medium-speed mode in which the bus master runs on a medium-speed clock and the other supporting modules run on the high-speed clock, and a function that allows the medium-speed mode to be disabled and the clock division ratio to be changed for the entire chip. A clock from  $\phi/2$  to  $\phi/32$  can be selected.

### 18.1.1 Block Diagram

Figure 18-1 shows a block diagram of the clock pulse generator.

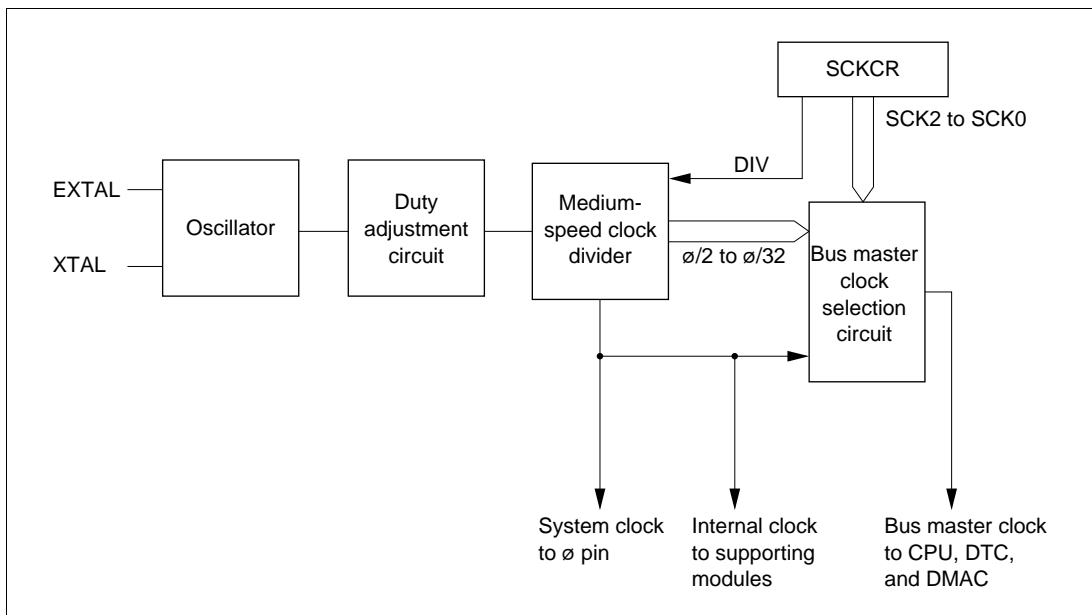


Figure 18-1 Block Diagram of Clock Pulse Generator

## 18.1.2 Register Configuration

The clock pulse generator is controlled by SCKCR. Table 18-1 shows the register configuration.

**Table 18-1 Clock Pulse Generator Register**

Name	Abbreviation	R/W	Initial Value	Address*
System clock control register	SCKCR	R/W	H'00	H'FF3A

Note:\* Lower 16 bits of the address.

## 18.2 Register Descriptions

### 18.2.1 System Clock Control Register (SCKCR)

Bit	:	7	6	5	4	3	2	1	0
		PSTOP	—	DIV	—	—	SCK2	SCK1	SCK0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	—	—	R/W	R/W	R/W

SCKCR is an 8-bit readable/writable register that controls  $\phi$  clock output, the medium-speed mode in which the bus master runs on a medium-speed clock and the other supporting modules run on the high-speed clock, and a function that allows the medium-speed mode to be disabled and the clock division ratio to be changed for the entire chip.

SCKCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7— $\phi$  Clock Output Disable (PSTOP):** Controls  $\phi$  output.

Bit 7 PSTOP	Description			
	Normal Operation	Sleep Mode	Software Standby Mode	Hardware Standby Mode
0	$\phi$ output (Initial value)	$\phi$ output	Fixed high	High impedance
1	Fixed high	Fixed high	Fixed high	High impedance

**Bit 6—Reserved:** This bit can be read or written to, but only 0 should be written.

**Bit 5—Division Ratio Select (DIV):** When the DIV bit is set to 1, the medium-speed mode is disabled and a clock obtained using the division ratio set with bits SCK2 to SCK0 is supplied to the entire chip. In this way, the current dissipation within the chip is reduced in proportion to the division ratio. As the frequency of  $\phi$  changes, the following points must be noted.

- The division ratio set with bits SCK2 to SCK0 should be selected so as to fall within the guaranteed operation range of clock cycle time  $t_{cyc}$  given in the AC timing table in the Electrical Characteristics section. Ensure that  $\phi_{min} = 2 \text{ MHz}$ , and the condition  $\phi < 2 \text{ MHz}$  does not arise.
- All internal modules basically operate on  $\phi$ . Note, therefore, that time processing involving the timers, the SCI, etc., will change when the division ratio changes. The wait time when software standby is cleared will also change in line with a change in the division ratio.
- The division ratio can be changed while the chip is operating. The clock output from the  $\phi$  pin will also change when the division ratio is changed. The frequency of the clock output from the  $\phi$  pin in this case will be as follows:

$$\phi = \text{EXTAL} \times n$$

Where: EXTAL: Crystal resonator or external clock frequency

n: Division ratio ( $n = \phi/2, \phi/4, \text{ or } \phi/8$ )

- Do not set the DIV bit and bits SCK2 to SCK0 simultaneously. First set the DIV bit, then bits SCK2 to SCK0.

#### Bit 5

DIV	Description
0	When bits SCK2 to SCK0 are set to other than high-speed mode, medium-speed mode is set (Initial value)
1	When bits SCK2 to SCK0 are set to other than high-speed mode, a divided clock is supplied to the entire chip

**Bits 4 and 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—System Clock Select 2 to 0 (SCK2 to SCK0):** When the DIV bit is cleared to 0, these bits select the medium-speed mode; when the DIV bit is set to 1, they select the division ratio of the clock supplied to the entire chip.

Bit 2 SCK2	Bit 1 SCK1	Bit 0 SCK0	Description	
			DIV = 0	DIV = 1
0	0	0	Bus master is in high-speed mode (Initial value)	Bus master is in high-speed mode (Initial value)
		1	Medium-speed clock is $\phi/2$	Clock supplied to entire chip is $\phi/2$
	1	0	Medium-speed clock is $\phi/4$	Clock supplied to entire chip is $\phi/4$
1	0	1	Medium-speed clock is $\phi/8$	Clock supplied to entire chip is $\phi/8$
		0	Medium-speed clock is $\phi/16$	—
	1	—	—	—

## 18.3 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

### 18.3.1 Connecting a Crystal Resonator

**Circuit Configuration:** A crystal resonator can be connected as shown in the example in figure 18-2. Select the damping resistance  $R_d$  according to table 18-2. An AT-cut parallel-resonance crystal should be used.

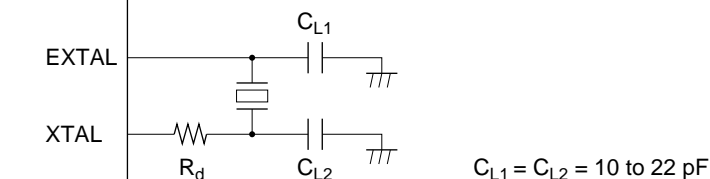


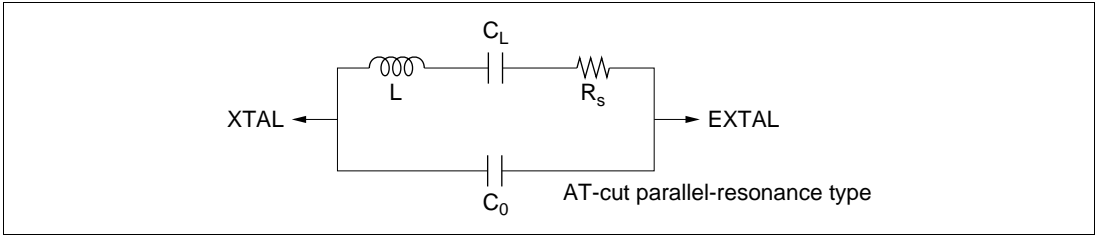
Figure 18-2 Connection of Crystal Resonator (Example)

Table 18-2 Damping Resistance Value

Frequency (MHz)	2	4	8	12	16	20	25*
$R_d$ ( $\Omega$ )	6.8 k	500	200	0	0	0	0

Note: \* In planning stage

**Crystal Resonator:** Figure 18-3 shows the equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 18-3 and the same resonance frequency as the system clock ( $\phi$ ).



**Figure 18-3 Crystal Resonator Equivalent Circuit**

**Table 18-3 Crystal Resonator Characteristics**

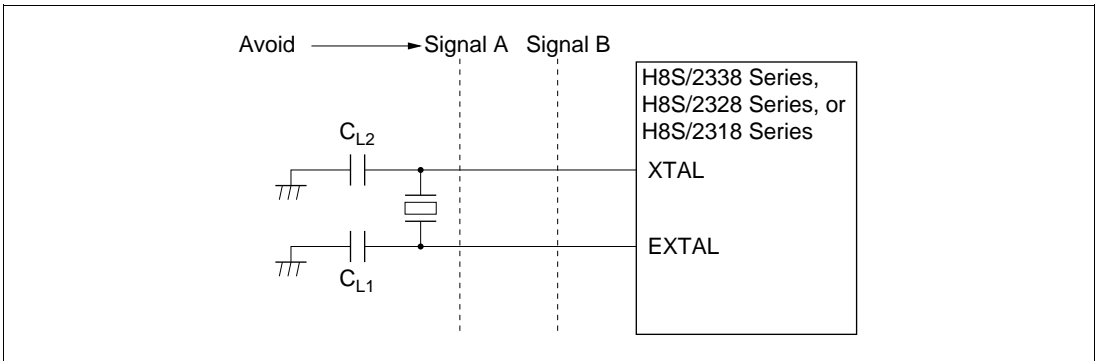
Frequency (MHz)	2	4	8	12	16	20	25*
$R_s$ max ( $\Omega$ )	500	120	80	60	50	40	40
$C_0$ max (pF)	7	7	7	7	7	7	7

Note: \* In planning stage

**Notes on Board Design:** When a crystal resonator is connected, the following points should be noted:

Other signal lines should be routed away from the oscillator circuit to prevent induction from interfering with correct oscillation. See figure 18-4.

When designing the board, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins.



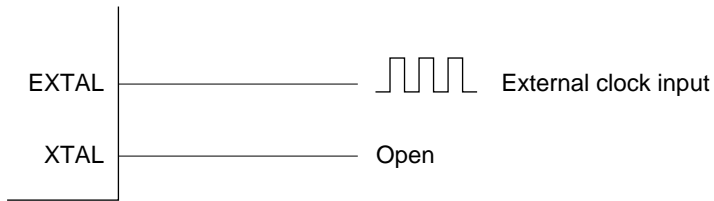
**Figure 18-4 Example of Incorrect Board Design**



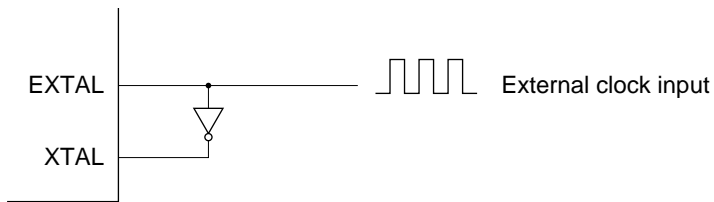
### 18.3.2 External Clock Input

**Circuit Configuration:** An external clock signal can be input as shown in the examples in figure 18-5. If the XTAL pin is left open, make sure that stray capacitance is no more than 10 pF.

In example (b), make sure that the external clock is held high in standby mode.



(a) XTAL pin left open



(b) Complementary clock input at XTAL pin

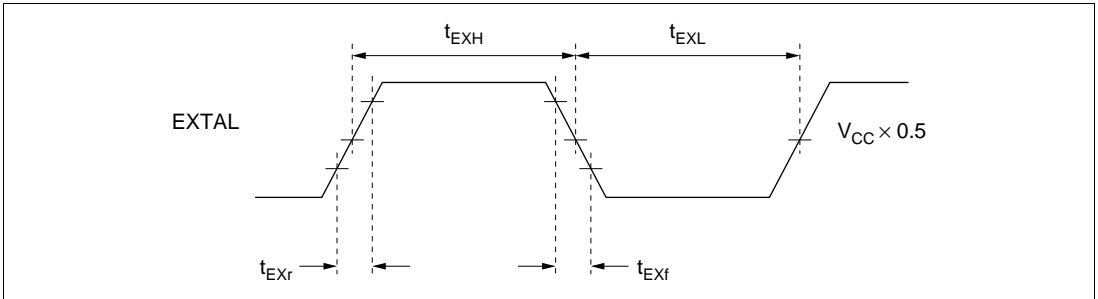
**Figure 18-5 External Clock Input (Examples)**

**External Clock:** The external clock signal should have the same frequency as the system clock ( $\phi$ ).

Table 18-4 and figure 18-6 show the input conditions for the external clock.

**Table 18-4 External Clock Input Conditions**

Item	Symbol	$V_{CC} = 2.7\text{ V}$ to $3.3\text{ V}$		$V_{CC} = 3.0\text{ V}$ to $3.6\text{ V}$		Unit	Test Conditions	
		Min	Max	Min	Max			
External clock input low pulse width	$t_{EXL}$	20	—	10	—	ns	Figure 18-6	
External clock input high pulse width	$t_{EXH}$	20	—	10	—	ns		
External clock rise time	$t_{EXr}$	—	5	—	5	ns		
External clock fall time	$t_{EXf}$	—	5	—	5	ns		
Clock low pulse width level	$t_{CL}$	0.4	0.6	0.4	0.6	$t_{cyc}$	$\phi \geq 5\text{ MHz}$	Figure 22-4
		80	—	80	—	ns	$\phi < 5\text{ MHz}$	
Clock high pulse width level	$t_{CH}$	0.4	0.6	0.4	0.6	$t_{cyc}$	$\phi \geq 5\text{ MHz}$	
		80	—	80	—	ns	$\phi < 5\text{ MHz}$	



**Figure 18-6 External Clock Input Timing**

## 18.4 Duty Adjustment Circuit

When the oscillator frequency is 5 MHz or higher, the duty adjustment circuit adjusts the duty cycle of the clock signal from the oscillator to generate the system clock ( $\phi$ ).

## 18.5 Medium-Speed Clock Divider

The medium-speed clock divider divides the system clock to generate  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , and  $\phi/32$ .

## 18.6 Bus Master Clock Selection Circuit

The bus master clock selection circuit selects the system clock ( $\phi$ ) or one of the medium-speed clocks ( $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , or  $\phi/32$ ) to be supplied to the bus master, according to the settings of the SCK2 to SCK0 bits in SCKCR.

# Section 19 Power-Down Modes

## 19.1 Overview

In addition to the normal program execution state, the H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series have five power-down modes in which operation of the CPU and oscillator is halted and power dissipation is reduced. Low-power operation can be achieved by individually controlling the CPU, on-chip supporting modules, and so on.

The H8S/2338 Series, H8S/2328 Series, and H8S/2318 Series operating modes are as follows:

1. High-speed mode
2. Medium-speed mode
3. Sleep mode
4. Module stop mode
5. Software standby mode
6. Hardware standby mode

Of these, 2 to 6 are power-down modes. Sleep mode is a CPU mode, medium-speed mode is a CPU and bus master mode, and module stop mode is an on-chip supporting module mode (including bus masters other than the CPU). A combination of these modes can be set.

After a reset, the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip is in high-speed mode.

Table 19-1 shows the conditions for transition to the various modes, the status of the CPU, on-chip supporting modules, etc., and the method of clearing each mode.

**Table 19-1 Operating Modes**

Operating Mode	Transition Condition	Clearing Condition	Oscillator	CPU		Modules		I/O Ports
					Registers	Registers	Registers	
High speed mode	Control register		Functions	High speed	Function	High speed	Function	High speed
Medium-speed mode	Control register		Functions	Medium speed	High/medium speed * <sup>1</sup>	Function	Function	High speed
Sleep mode	Instruction	Interrupt	Functions	Halted	Retained	High speed	Function	High speed
Module stop mode	Control register		Functions	High/medium speed	Function	Halted	Retained/reset * <sup>2</sup>	Retained
Software standby mode	Instruction	External interrupt	Halted	Halted	Retained	Halted	Retained/reset * <sup>2</sup>	Retained
Hardware standby mode	Pin		Halted	Halted	Undefined	Halted	Reset	High impedance

- Notes: 1. The bus master operates on the medium-speed clock, and other on-chip supporting modules on the high-speed clock.
2. Some SCI registers and the A/D converter are reset, and other on-chip supporting modules retain their states.

### 19.1.1 Register Configuration

Power-down modes are controlled by the SBYCR, SCKCR, and MSTPCR registers. Table 19-2 summarizes these registers.

**Table 19-2 Power-Down Mode Registers**

Name	Abbreviation	R/W	Initial Value	Address*
Standby control register	SBYCR	R/W	H'08	H'FF38
System clock control register	SCKCR	R/W	H'00	H'FF3A
Module stop control register H	MSTPCRH	R/W	H'3F	H'FF3C
Module stop control register L	MSTPCRL	R/W	H'FF	H'FF3D

Note: \* Lower 16 bits of the address.

## 19.2 Register Descriptions

### 19.2.1 Standby Control Register (SBYCR)

Bit	:	7	6	5	4	3	2	1	0
		SSBY	STS2	STS1	STS0	OPE	—	—	IRQ3TS
Initial value :		0	0	0	0	1	0	0	0
R/W	:	R/W	R/W	R/W	R/W	R/W	—	—	R/W

SBYCR is an 8-bit readable/writable register that performs software standby mode control.

SBYCR is initialized to H'08 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7—Software Standby (SSBY):** Specifies a transition to software standby mode. Remains set to 1 when software standby mode is released by an external interrupt, and a transition is made to normal operation. The SSBY bit should be cleared by writing 0 to it.

#### Bit 7

SSBY	Description
0	Transition to sleep mode after execution of SLEEP instruction (Initial value)
1	Transition to software standby mode after execution of SLEEP instruction

**Bits 6 to 4—Standby Timer Select 2 to 0 (STS2 to STS0):** These bits select the time the MCU waits for the clock to stabilize when software standby mode is cleared by an external interrupt. With crystal oscillation, refer to table 19-4 and make a selection according to the operating frequency so that the standby time is at least 8 ms (the oscillation stabilization time). With an external clock, any selection can be made\*.

Note: \* Except in the F-ZTAT version.

Bit 6 STS2	Bit 5 STS1	Bit 4 STS0	Description
0	0	0	Standby time = 8192 states (Initial value)
		1	Standby time = 16384 states
	1	0	Standby time = 32768 states
		1	Standby time = 65536 states
1	0	0	Standby time = 131072 states
		1	Standby time = 262144 states
	1	0	Reserved
		1	Standby time = 16 states*

Note: \* Not available in the F-ZTAT version.

**Bit 3—Output Port Enable (OPE):** Specifies whether the output of the address bus and bus control signals ( $\overline{CS0}$  to  $\overline{CS7}$ ,  $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{HWR}$ ,  $\overline{LWR}$ ,  $\overline{CAS}$ ) is retained or set to the high-impedance state in software standby mode.

Bit 3 OPE	Description
0	In software standby mode, address bus and bus control signals are high-impedance
1	In software standby mode, address bus and bus control signals retain output state (Initial value)

**Bits 2 and 1—Reserved:** Read-only bits, always read as 0.

**Bit 0—IRQ37 Software Standby Clear Select (IRQ37S):** Specifies whether inputs  $\overline{IRQ3}$  to  $\overline{IRQ7}$  can be used as software standby mode clearing sources in addition to the usual sources, NMI and  $\overline{IRQ0}$  to  $\overline{IRQ2}$  inputs.

Bit 0 IRQ37S	Description
0	Inputs $\overline{IRQ3}$ to $\overline{IRQ7}$ cannot be used as software standby mode clearing sources (Initial value)
1	Inputs $\overline{IRQ3}$ to $\overline{IRQ7}$ can be used as software standby mode clearing sources

## 19.2.2 System Clock Control Register (SCKCR)

Bit	:	7	6	5	4	3	2	1	0
		PSTOP	—	DIV	—	—	SCK2	SCK1	SCK0
Initial value :		0	0	0	0	0	0	0	0
R/W	:	R/W	R/W	R/W	—	—	R/W	R/W	R/W

SCKCR is an 8-bit readable/writable register that controls  $\phi$  clock output, the medium-speed mode in which the bus master runs on a medium-speed clock and the other supporting modules run on the high-speed clock, and a function that allows the medium-speed mode to be disabled and the clock division ratio to be changed for the entire chip.

SCKCR is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bit 7— $\phi$  Clock Output Disable (PSTOP):** Controls  $\phi$  output.

Bit 7 PSTOP	Description			
	Normal Operating Mode	Sleep Mode	Software Standby Mode	Hardware Standby Mode
0	$\phi$ output (Initial value)	$\phi$ output	Fixed high	High impedance
1	Fixed high	Fixed high	Fixed high	High impedance

**Bit 6—Reserved:** This bit can be read or written to, but only 0 should be written.

**Bit 5—Division Ratio Select (DIV):** When the DIV bit is set to 1, the medium-speed mode is disabled and a clock obtained using the division ratio set with bits SCK2 to SCK0 is supplied to the entire chip. In this way, the current dissipation within the chip is reduced in proportion to the division ratio. As the frequency of  $\phi$  changes, the following points must be noted.

- The division ratio set with bits SCK2 to SCK0 should be selected so as to fall within the guaranteed operation range of clock cycle time  $t_{cyc}$  given in the AC timing table in the Electrical Characteristics section. Ensure that  $\phi_{min} = 2$  MHz, and the condition  $\phi < 2$  MHz does not arise.
- All internal modules basically operate on  $\phi$ . Note, therefore, that time processing involving the timers, the SCI, etc., will change when the division ratio changes. The wait time when software standby is cleared will also change in line with a change in the division ratio.



- The division ratio can be changed while the chip is operating. The clock output from the  $\phi$  pin will also change when the division ratio is changed. The frequency of the clock output from the  $\phi$  pin in this case will be as follows:

$$\phi = \text{EXTAL} \times n$$

Where: EXTAL: Crystal resonator or external clock frequency

n: Division ratio ( $n = \phi/2, \phi/4, \text{ or } \phi/8$ )

- Do not set the DIV bit and bits SCK2 to SCK0 simultaneously. First set the DIV bit, then bits SCK2 to SCK0.

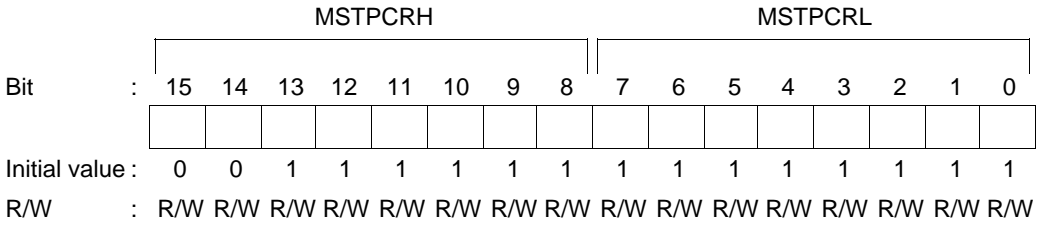
Bit 5 DIV	Description
0	When bits SCK2 to SCK0 are set to other than high-speed mode, medium-speed mode is set (Initial value)
1	When bits SCK2 to SCK0 are set to other than high-speed mode, a divided clock is supplied to the entire chip

**Bits 4 and 3—Reserved:** Read-only bits, always read as 0.

**Bits 2 to 0—System Clock Select 2 to 0 (SCK2 to SCK0):** When the DIV bit is cleared to 0, these bits select the bus master clock; when the DIV bit is set to 1, they select the division ratio of the clock supplied to the entire chip.

Bit 2 SCK2	Bit 1 SCK1	Bit 0 SCK0	Description	
			DIV = 0	DIV = 1
0	0	0	Bus master is in high-speed mode (Initial value)	Bus master is in high-speed mode (Initial value)
		1	Medium-speed clock is $\phi/2$	Clock supplied to entire chip is $\phi/2$
	1	0	Medium-speed clock is $\phi/4$	Clock supplied to entire chip is $\phi/4$
		1	Medium-speed clock is $\phi/8$	Clock supplied to entire chip is $\phi/8$
1	0	0	Medium-speed clock is $\phi/16$	—
		1	Medium-speed clock is $\phi/32$	—
	1	—	—	—

### 19.2.3 Module Stop Control Register (MSTPCR)



MSTPCR is a 16-bit readable/writable register that performs module stop mode control.

MSTPCR is initialized to H'3FFF by a reset and in hardware standby mode. It is not initialized in software standby mode.

**Bits 15 to 0—Module Stop (MSTP 15 to MSTP 0):** These bits specify module stop mode. See table 19-3 for the method of selecting on-chip supporting modules.

#### Bits 15 to 0

MSTP15 to MSTP0	Description
0	Module stop mode cleared
1	Module stop mode set

## 19.3 Medium-Speed Mode

When the SCK2 to SCK0 bits in SCKCR are set to 1, the operating mode changes to medium-speed mode as soon as the current bus cycle ends. In medium-speed mode, the CPU operates on the operating clock ( $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , or  $\phi/32$ ) specified by the SCK2 to SCK0 bits. The bus masters other than the CPU (the DMAC and DTC) also operate in medium-speed mode. On-chip supporting modules other than the bus masters always operate on the high-speed clock ( $\phi$ ).

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if  $\phi/4$  is selected as the operating clock, on-chip memory is accessed in 4 states, and internal I/O registers in 8 states.

Medium-speed mode is cleared by clearing all of bits SCK2 to SCK0 to 0. A transition is made to high-speed mode and medium-speed mode is cleared at the end of the current bus cycle.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, a transition is made to sleep mode. When sleep mode is cleared by an interrupt, medium-speed mode is restored.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, a transition is made to software standby mode. When software standby mode is cleared by an external interrupt, medium-speed mode is restored.

When the  $\overline{\text{RES}}$  pin is driven low, a transition is made to the reset state, and medium-speed mode is cleared. The same applies in the case of a reset caused by overflow of the watchdog timer.

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

Figure 19-1 shows the timing for transition to and clearance of medium-speed mode.

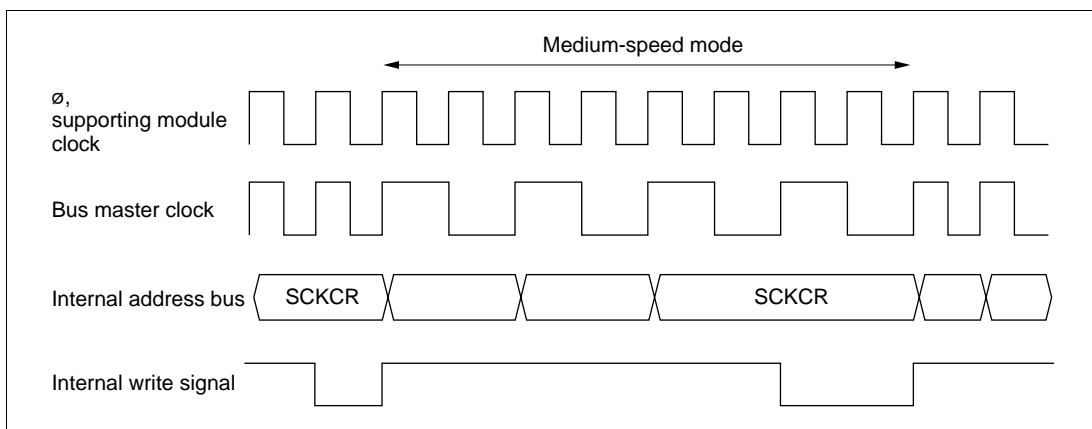


Figure 19-1 Medium-Speed Mode Transition and Clearance Timing

## 19.4 Sleep Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other supporting modules do not stop.

Sleep mode is cleared by a reset or any interrupt, and the CPU returns to the normal program execution state via the exception handling state. Sleep mode is not cleared if interrupts are disabled, or if interrupts other than NMI are masked by the CPU.

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

## 19.5 Module Stop Mode

### 19.5.1 Module Stop Mode

Module stop mode can be set for individual on-chip supporting modules.

When the corresponding MSTP bit in MSTPCR is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

Table 19-3 shows MSTP bits and the corresponding on-chip supporting modules.

When the corresponding MSTP bit is cleared to 0, module stop mode is cleared and the module starts operating at the end of the bus cycle. In module stop mode, the internal states of modules other than the SCI and A/D converter are retained.

After reset clearance, all modules other than DMAC and DTC are in module stop mode.

When an on-chip supporting module is in module stop mode, read/write access to its registers is disabled.

Do not make a transition to sleep mode with MSTPCR set to H'FFFF or H'EEEE, as this will halt operation of the bus controller.

**Table 19-3 MSTP Bits and Corresponding On-Chip Supporting Modules**

Register	Bit	Module
MSTPCRH	MSTP15	DMA controller (DMAC)
	MSTP14	Data transfer controller (DTC)
	MSTP13	16-bit timer-pulse unit (TPU)
	MSTP12	8-bit timer module
	MSTP11	Programmable pulse generator (PPG)
	MSTP10	D/A converter (channels 0 and 1)
	MSTP9	A/D converter
	MSTP8	—
MSTPCRL	MSTP7	Serial communication interface (SCI) channel 2
	MSTP6	Serial communication interface (SCI) channel 1
	MSTP5	Serial communication interface (SCI) channel 0
	MSTP4	D/A converter (channels 2 and 3)
	MSTP3	—
	MSTP2	—
	MSTP1	—
	MSTP0	—

Note: Bits 8 and 3 to 0 can be read or written to, but do not affect operation.

### 19.5.2 Usage Notes

**DMAC\*/DTC Module Stop:** Depending on the operating status of the DMAC\* or DTC, the MSTP15 and MSTP14 bits may not be set to 1. Setting of the DMAC\* or DTC module stop mode should be carried out only when the respective module is not activated.

For details, refer to section 5, DMA Controller, and section 6, Data Transfer Controller.

**On-Chip Supporting Module Interrupts:** Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC\* or DTC activation source. Interrupts should therefore be disabled before entering module stop mode.

**Writing to MSTPCR:** MSTPCR should only be written to by the CPU.

Note: \* Some models do not have an on-chip DMAC; see the reference manual for the relevant model for confirmation.

## 19.6 Software Standby Mode

### 19.6.1 Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip supporting modules, and oscillator all stop. However, the contents of the CPU's internal registers, RAM data, and the states of on-chip supporting modules other than the SCI and A/D converter, and I/O ports, are retained. Whether the address bus and bus control signals are placed in the high-impedance state or retain the output state can be specified by the OPE bit in SBYCR.

In this mode the oscillator stops, and therefore power dissipation is significantly reduced.

### 19.6.2 Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, or pins  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ7}}$ \*), or by means of the  $\overline{\text{RES}}$  pin or  $\overline{\text{STBY}}$  pin.

**Clearing with an Interrupt:** When an NMI or  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ7}}$ \* interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SYSCR, stable clocks are supplied to the entire H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip, software standby mode is cleared, and interrupt exception handling is started.

When clearing software standby mode with an  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ7}}$ \* interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ7}}$ \* is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side or has been designated as a DTC activation source.

Note: \* Setting the  $\overline{\text{IRQ37S}}$  bit to 1 enables  $\overline{\text{IRQ3}}$  to  $\overline{\text{IRQ7}}$  to be used as software standby mode clearing sources.

**Clearing with the  $\overline{\text{RES}}$  Pin:** When the  $\overline{\text{RES}}$  pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip. Note that the  $\overline{\text{RES}}$  pin must be held low until clock oscillation stabilizes. When the  $\overline{\text{RES}}$  pin goes high, the CPU begins reset exception handling.

**Clearing with the  $\overline{\text{STBY}}$  Pin:** When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

### 19.6.3 Setting Oscillation Stabilization Time after Clearing Software Standby Mode

Bits STS2 to STS0 in SBYCR should be set as described below.

**Using a Crystal Oscillator:** Set bits STS2 to STS0 so that the standby time is at least 8 ms (the oscillation stabilization time).

Table 19-4 shows the standby times for different operating frequencies and settings of bits STS2 to STS0.

**Table 19-4 Oscillation Stabilization Time Settings**

STS2	STS1	STS0	Standby Time	25 MHz*	20 MHz	16 MHz	12 MHz	10 MHz	8 MHz	6 MHz	4 MHz	2 MHz	Unit
0	0	0	8192 states	0.32	0.41	0.51	0.68	0.8	1.0	1.3	2.0	4.1	ms
		1	16384 states	0.65	0.82	1.0	1.3	1.6	2.0	2.7	4.1	8.2	
	1	0	32768 states	1.3	1.6	2.0	2.7	3.3	4.1	5.5	8.2	16.4	
		1	65536 states	2.6	3.3	4.1	5.5	6.6	8.2	10.9	16.4	32.8	
1	0	0	131072 states	5.2	6.6	8.2	10.9	13.1	16.4	21.8	32.8	65.5	
		1	262144 states	10.4	13.1	16.4	21.8	26.2	32.8	43.6	65.6	131.2	
	1	0	Reserved	—	—	—	—	—	—	—	—	—	
		1	16 states	0.6	0.8	1.0	1.3	1.6	2.0	2.7	4.0	8.0	

  : Recommended time setting

Note: \* In planning stage

**Using an External Clock:** Any value can be set. Normally, use of the minimum time is recommended.\*

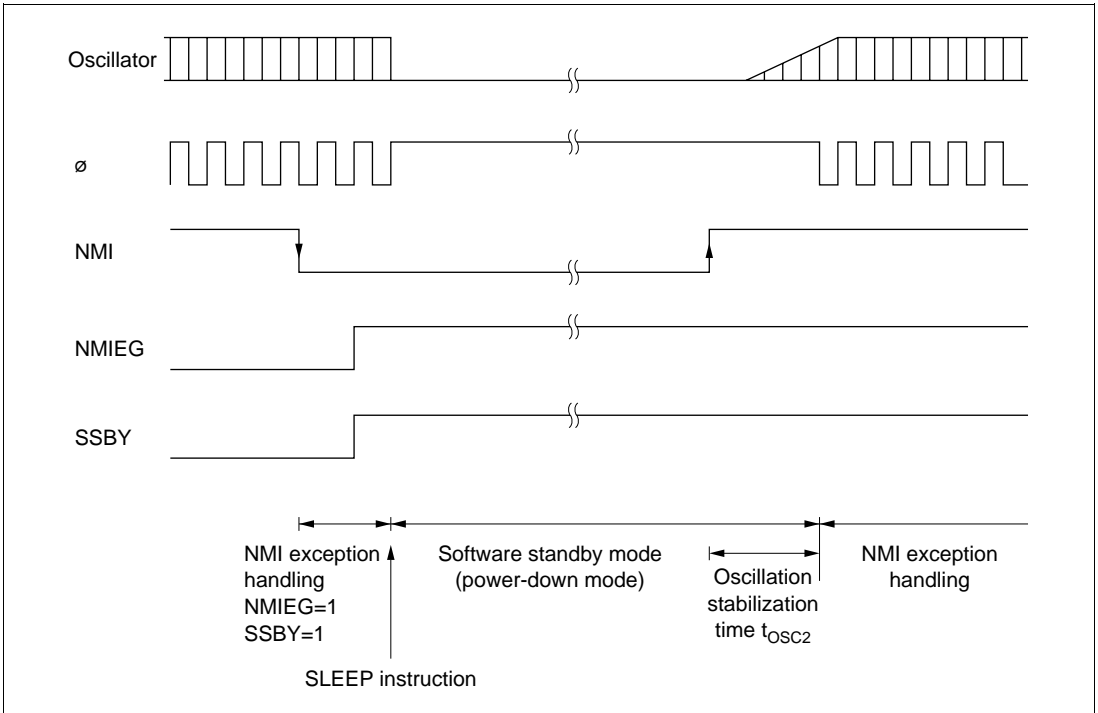
Note: \* The 16-state standby time cannot be used in the F-ZTAT version; a standby time of 8192 states or longer should be used.

### 19.6.4 Software Standby Mode Application Example

Figure 19-2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in SYSCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.



**Figure 19-2 Software Standby Mode Application Example**

### 19.6.5 Usage Notes

**I/O Port Status:** In software standby mode, I/O port states are retained. If the OPE bit is set to 1, the address bus and bus control signal output is also retained. Therefore, there is no reduction in current dissipation for the output current when a high-level signal is output.

**Current Dissipation during Oscillation Stabilization Wait Period:** Current dissipation increases during the oscillation stabilization wait period.

**Write Data Buffer Function:** The write data buffer function and software standby mode cannot be used at the same time. When the write data buffer function is used, the WDBE bit in BCRL should be cleared to 0 to cancel the write data buffer function before entering software standby mode. Also check that external writes have finished, by reading external addresses, etc., before executing a SLEEP instruction to enter software standby mode. See section 4.9, Write Data Buffer Function, for details of the write data buffer function.



## 19.7 Hardware Standby Mode

### 19.7.1 Hardware Standby Mode

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power dissipation. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the  $\overline{\text{STBY}}$  pin low.

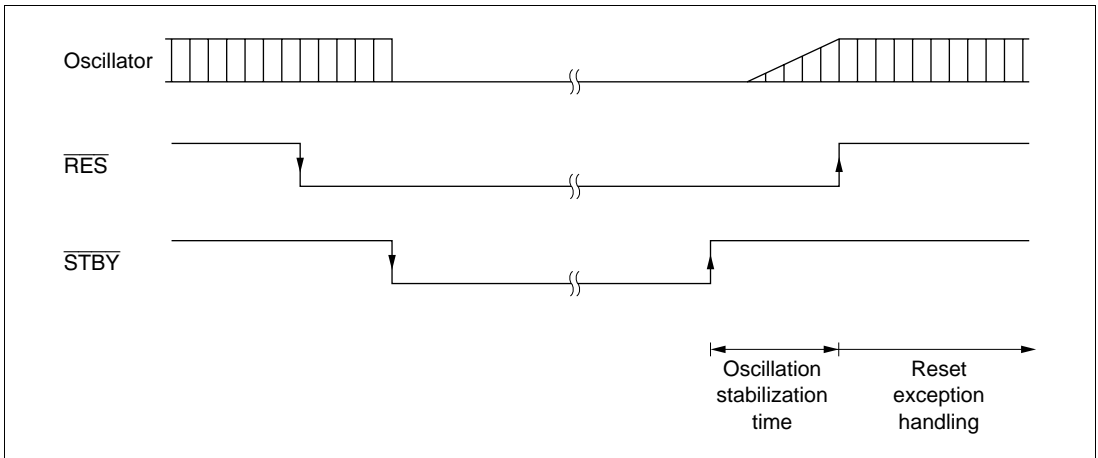
Do not change the state of the mode pins ( $\text{MD}_2$  to  $\text{MD}_0$ ) while the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series chip is in hardware standby mode.

Hardware standby mode is cleared by means of the  $\overline{\text{STBY}}$  pin and the  $\overline{\text{RES}}$  pin. When the  $\overline{\text{STBY}}$  pin is driven high while the  $\overline{\text{RES}}$  pin is low, the reset state is set and clock oscillation is started. Ensure that the  $\overline{\text{RES}}$  pin is held low until the clock oscillator stabilizes (at least 8 ms—the oscillation stabilization time—when using a crystal oscillator). When the  $\overline{\text{RES}}$  pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

### 19.7.2 Hardware Standby Mode Timing

Figure 19-3 shows an example of hardware standby mode timing.

When the  $\overline{\text{STBY}}$  pin is driven low after the  $\overline{\text{RES}}$  pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the  $\overline{\text{STBY}}$  pin high, waiting for the oscillation stabilization time, then changing the  $\overline{\text{RES}}$  pin from low to high.



**Figure 19-3 Hardware Standby Mode Timing**

## 19.8 $\phi$ Clock Output Disabling Function

Output of the  $\phi$  clock can be controlled by means of the PSTOP bit in SCKCR, and DDR for the corresponding port. When the PSTOP bit is set to 1, the  $\phi$  clock stops at the end of the bus cycle, and  $\phi$  output goes high.  $\phi$  clock output is enabled when the PSTOP bit is cleared to 0. When DDR for the corresponding port is cleared to 0,  $\phi$  clock output is disabled and input port mode is set. Table 19-5 shows the state of the  $\phi$  pin in each processing state.

**Table 19-5  $\phi$  Pin State in Each Processing State**

DDR	0	1	
PSTOP	—	0	1
Hardware standby mode	High impedance		
Software standby mode	High impedance	Fixed high	
Sleep mode	High impedance	$\phi$ output	Fixed high
Normal operating state	High impedance	$\phi$ output	Fixed high

# Appendix A Instruction Set

## A.1 Instruction List

### Operand Notation

Rd	General register (destination)* <sup>1</sup>
Rs	General register (source)* <sup>1</sup>
Rn	General register* <sup>1</sup>
ERn	General register (32-bit register)
MAC	Multiply-and-accumulate register (32-bit register)* <sup>2</sup>
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extended control register
CCR	Condition-code register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#IMM	Immediate data
disp	Displacement
+	Add
-	Subtract
×	Multiply
÷	Divide
^	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Transfer from the operand on the left to the operand on the right, or transition from the state on the left to the state on the right
¬	Logical NOT (logical complement)
( ) < >	Contents of operand
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

- Notes: 1. General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).
2. The MAC register cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series.

## Condition Code Notation

### Symbol

↕	Changes according to the result of the instruction
*	Undetermined (no guaranteed value)
0	Always cleared to 0
1	Always set to 1
—	Not affected by execution of the instruction

Table A-1 Instruction Set

(1) Data Transfer Instructions

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1			
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@aa		I	H	N	Z	V		C		
																			Advanced	
MOV	MOV.B #xx:8,Rd	B	2									#xx:8→Rd8	—	—	↑	↓	0	—	1	
	MOV.B Rs,Rd	B		2								Rs8→Rd8	—	—	↑	↓	0	—	1	
	MOV.B @ERs,Rd	B			2							@ERs→Rd8	—	—	↑	↓	0	—	2	
	MOV.B @(d:16,ERs),Rd	B				4						@(d:16,ERs)→Rd8	—	—	↑	↓	0	—	3	
	MOV.B @(d:32,ERs),Rd	B					8					@(d:32,ERs)→Rd8	—	—	↑	↓	0	—	5	
	MOV.B @ERs+,Rd	B						2				@ERs→Rd8,ERs32+1→ERs32	—	—	↑	↓	0	—	3	
	MOV.B @aa:8,Rd	B							2			@aa:8→Rd8	—	—	↑	↓	0	—	2	
	MOV.B @aa:16,Rd	B								4		@aa:16→Rd8	—	—	↑	↓	0	—	3	
	MOV.B @aa:32,Rd	B									6	@aa:32→Rd8	—	—	↑	↓	0	—	4	
	MOV.B Rs,@ERd	B			2							Rs8→@ERd	—	—	↑	↓	0	—	2	
	MOV.B Rs,@(d:16,ERd)	B				4						Rs8→@(d:16,ERd)	—	—	↑	↓	0	—	3	
	MOV.B Rs,@(d:32,ERd)	B					8					Rs8→@(d:32,ERd)	—	—	↑	↓	0	—	5	
	MOV.B Rs,@-ERd	B						2				ERd32-1→ERd32,Rs8→@ERd	—	—	↑	↓	0	—	3	
	MOV.B Rs,@aa:8	B								2		Rs8→@aa:8	—	—	↑	↓	0	—	2	
	MOV.B Rs,@aa:16	B									4	Rs8→@aa:16	—	—	↑	↓	0	—	3	
	MOV.B Rs,@aa:32	B										6	Rs8→@aa:32	—	—	↑	↓	0	—	4
	MOV.W #xx:16,Rd	W	4										#xx:16→Rd16	—	—	↑	↓	0	—	2
MOV.W Rs,Rd	W		2									Rs16→Rd16	—	—	↑	↓	0	—	1	
MOV.W @ERs,Rd	W			2								@ERs→Rd16	—	—	↑	↓	0	—	2	

Table A-1 Instruction Set (cont)

## (1) Data Transfer Instructions (cont)

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1	
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa		I	H	N	Z	V		C
														Advanced				
MOV	MOV.W @(d:16,ERs),Rd	W			4						@(d:16,ERs)→Rd16	—	—	↑	↓	0	—	3
	MOV.W @(d:32,ERs),Rd	W			8						@(d:32,ERs)→Rd16	—	—	↑	↓	0	—	5
	MOV.W @ERs+,Rd	W				2					@ERs→Rd16,ERs32+2→ERs32	—	—	↑	↓	0	—	3
	MOV.W @aa:16,Rd	W					4				@aa:16→Rd16	—	—	↑	↓	0	—	3
	MOV.W @aa:32,Rd	W					6				@aa:32→Rd16	—	—	↑	↓	0	—	4
	MOV.W Rs,@ERd	W		2							Rs16→@ERd	—	—	↑	↓	0	—	2
	MOV.W Rs,@(d:16,ERd)	W			4						Rs16→@(d:16,ERd)	—	—	↑	↓	0	—	3
	MOV.W Rs,@(d:32,ERd)	W			8						Rs16→@(d:32,ERd)	—	—	↑	↓	0	—	5
	MOV.W Rs,@-ERd	W				2					ERd32-2→ERd32,Rs16→@ERd	—	—	↑	↓	0	—	3
	MOV.W Rs,@aa:16	W					4				Rs16→@aa:16	—	—	↑	↓	0	—	3
	MOV.W Rs,@aa:32	W					6				Rs16→@aa:32	—	—	↑	↓	0	—	4
	MOV.L #xx:32,ERd	L	6								#xx:32→ERd32	—	—	↑	↓	0	—	3
	MOV.L ERs,ERd	L		2							ERs32→ERd32	—	—	↑	↓	0	—	1
	MOV.L @ERs,ERd	L			4						@ERs→ERd32	—	—	↑	↓	0	—	4
	MOV.L @(d:16,ERs),ERd	L				6					@(d:16,ERs)→ERd32	—	—	↑	↓	0	—	5
	MOV.L @(d:32,ERs),ERd	L				10					@(d:32,ERs)→ERd32	—	—	↑	↓	0	—	7
	MOV.L @ERs+,ERd	L				4					@ERs→ERd32,ERs32+4→@ERs32	—	—	↑	↓	0	—	5
MOV.L @aa:16,ERd	L					6				@aa:16→ERd32	—	—	↑	↓	0	—	5	
MOV.L @aa:32,ERd	L					8				@aa:32→ERd32	—	—	↑	↓	0	—	6	

**Table A-1 Instruction Set (cont)**

**(1) Data Transfer Instructions (cont)**

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1							
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa			I	H	N	Z	V	C	Advanced					
MOV	MOV.L ERs, @ERd	L		4													ERs32→@ERd	—	—	↑	↓	0	—	4
	MOV.L ERs, @(d:16,ERd)	L			6												ERs32→@(d:16,ERd)	—	—	↑	↓	0	—	5
	MOV.L ERs, @(d:32,ERd)	L			10												ERs32→@(d:32,ERd)	—	—	↑	↓	0	—	7
	MOV.L ERs, @-ERd	L				4											ERd32-4→ERd32,ERs32→@ERd	—	—	↑	↓	0	—	5
	MOV.L ERs, @aa:16	L					6										ERs32→@aa:16	—	—	↑	↓	0	—	5
	MOV.L ERs, @aa:32	L						8									ERs32→@aa:32	—	—	↑	↓	0	—	6
POP	POP.W Rn	W								2							@SP→Rn16,SP+2→SP	—	—	↑	↓	0	—	3
	POP.L ERn	L								4							@SP→ERn32,SP+4→SP	—	—	↑	↓	0	—	5
PUSH	PUSH.W Rn	W								2							SP-2→SP,Rn16→@SP	—	—	↑	↓	0	—	3
	PUSH.L ERn	L								4							SP-4→SP,ERn32→@SP	—	—	↑	↓	0	—	5
LDM	LDM @SP+,(ERm-ERn)	L								4							(@SP→ERn32,SP+4→SP) Repeated for each register restored	—	—	—	—	—	—	7/9/11 [1]
STM	STM (ERm-ERn),@-SP	L								4							(SP-4→SP,ERn32→@SP) Repeated for each register saved	—	—	—	—	—	—	7/9/11 [1]
MOVFP	MOVFP @aa:16,Rd	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series															[2]							
MOVTP	MOVTP Rs,@aa:16	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series															[2]							

Table A-1 Instruction Set

## (2) Arithmetic Instructions

Mnemonic		Addressing Mode/ Instruction Length (Bytes)									Operation	Condition Code					No. of States*1				
		Operand Size	#xx	Rn	@ERn	@ (d, ERn)	@ -ERn/@ERn+	@aa	@ (d, PC)	@ @aa			I	H	N	Z		V	C		
													Advanced								
ADD	ADD.B #xx:8,Rd	B	2															1			
	ADD.B Rs,Rd	B		2														1			
	ADD.W #xx:16,Rd	W	4											[3]	↑	↓	↑	↓	2		
	ADD.W Rs,Rd	W		2											[3]	↑	↓	↑	↓	1	
	ADD.L #xx:32,ERd	L	6												[4]	↑	↓	↑	↓	3	
	ADD.L ERs,ERd	L		2											[4]	↑	↓	↑	↓	1	
ADDX	ADDX #xx:8,Rd	B	2												↑	↓	[5]	↑	↓	1	
	ADDX Rs,Rd	B		2											↑	↓	[5]	↑	↓	1	
ADDS	ADDS #1,ERd	L		2											—	—	—	—	—	1	
	ADDS #2,ERd	L		2											—	—	—	—	—	1	
	ADDS #4,ERd	L		2											—	—	—	—	—	1	
INC	INC.B Rd	B		2											—	—	↑	↓	↑	—	1
	INC.W #1,Rd	W		2											—	—	↑	↓	↑	—	1
	INC.W #2,Rd	W		2											—	—	↑	↓	↑	—	1
	INC.L #1,ERd	L		2											—	—	↑	↓	↑	—	1
	INC.L #2,ERd	L		2											—	—	↑	↓	↑	—	1
DAA	DAA Rd	B		2											—	*	↑	↓	*	↑	1
SUB	SUB.B Rs,Rd	B		2											—	↑	↓	↑	↓	↑	1
	SUB.W #xx:16,Rd	W	4												—	[3]	↑	↓	↑	↓	2



Table A-1 Instruction Set (cont)

(2) Arithmetic Instructions (cont)

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa			I	H	N	Z	V		C
			Advanced																
SUB	SUB.W Rs,Rd	W	2																1
	SUB.L #xx:32,ERd	L	6																3
	SUB.L ERs,ERd	L	2																1
SUBX	SUBX #xx:8,Rd	B	2																1
	SUBX Rs,Rd	B	2																1
SUBS	SUBS #1,ERd	L	2																1
	SUBS #2,ERd	L	2																1
	SUBS #4,ERd	L	2																1
DEC	DEC.B Rd	B	2																1
	DEC.W #1,Rd	W	2																1
	DEC.W #2,Rd	W	2																1
	DEC.L #1,ERd	L	2																1
	DEC.L #2,ERd	L	2																1
DAS	DAS Rd	B	2										*				*		1
MULXU	MULXU.B Rs,Rd	B	2																12
	MULXU.W Rs,ERd	W	2																20
MULXS	MULXS.B Rs,Rd	B	4																13
	MULXS.W Rs,ERd	W	4																21

Table A-1 Instruction Set (cont)

## (2) Arithmetic Instructions (cont)

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1							
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa		I	H	N	Z	V		C						
			Advanced																					
DIVXU	DIVXU.B Rs,Rd	B	2													Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (unsigned division)	—	—	[6]	[7]	—	—	12	
	DIVXU.W Rs,ERd	W	2													ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (unsigned division)	—	—	[6]	[7]	—	—	20	
DIVXS	DIVXS.B Rs,Rd	B	4													Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (signed division)	—	—	[8]	[7]	—	—	13	
	DIVXS.W Rs,ERd	W	4													ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (signed division)	—	—	[8]	[7]	—	—	21	
CMP	CMP.B #xx:8,Rd	B	2													Rd8-#xx:8	—	↑	↓	↓	↓	↓	↓	1
	CMP.B Rs,Rd	B	2													Rd8-Rs8	—	↑	↓	↓	↓	↓	↓	1
	CMP.W #xx:16,Rd	W	4													Rd16-#xx:16	—	[3]	↑	↓	↓	↓	↓	2
	CMP.W Rs,Rd	W	2													Rd16-Rs16	—	[3]	↑	↓	↓	↓	↓	1
	CMP.L #xx:32,ERd	L	6													ERd32-#xx:32	—	[4]	↑	↓	↓	↓	↓	3
	CMP.L ERs,ERd	L	2													ERd32-ERs32	—	[4]	↑	↓	↓	↓	↓	1
NEG	NEG.B Rd	B	2													0-Rd8→Rd8	—	↑	↓	↓	↓	↓	↓	1
	NEG.W Rd	W	2													0-Rd16→Rd16	—	↑	↓	↓	↓	↓	↓	1
	NEG.L ERd	L	2													0-ERd32→ERd32	—	↑	↓	↓	↓	↓	↓	1
EXTU	EXTU.W Rd	W	2													0→(<bits 15 to 8> of Rd16)	—	—	0	↑	0	—	—	1
	EXTU.L ERd	L	2													0→(<bits 31 to 16> of ERd32)	—	—	0	↑	0	—	—	1

**Table A-1 Instruction Set (cont)**

**(2) Arithmetic Instructions (cont)**

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1		
			#xx	Rn	@ERn	@(d,ERn)	@_ERn/@ERn+	@aa	@(d,PC)	@_aa			I	H	N	Z	V	C	Advanced
EXTS	EXTS.W Rd	W	2														1		
	EXTS.L ERd	L	2														1		
TAS	TAS @ERd	B		4													4		
MAC	MAC @ERn+, @ERm+	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series														[2]			
CLRMAC	CLRMAC																		
LDMAC	LDMAC ERs,MACH																		
	LDMAC ERs,MACL																		
STMAC	STMAC MACH,ERd																		
	STMAC MACL,ERd																		



**Table A-1 Instruction Set**

**(4) Shift Instructions**

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1	
			#xx	Rn	@ ERn	@ (d,ERn)	@ -ERn/@ ERn+	@ aa	@ (d,PC)	@ @aa			I	H	N	Z		V
														Advanced				
SHAL	SHAL.B Rd	B	2											↑	↓	↓	↓	1
	SHAL.B #2,Rd	B	2											↑	↓	↓	↓	1
	SHAL.W Rd	W	2											↑	↓	↓	↓	1
	SHAL.W #2,Rd	W	2								C MSB ← LSB			↑	↓	↓	↓	1
	SHAL.L ERd	L	2											↑	↓	↓	↓	1
	SHAL.L #2,ERd	L	2											↑	↓	↓	↓	1
SHAR	SHAR.B Rd	B	2											↑	↓	0	↓	1
	SHAR.B #2,Rd	B	2											↑	↓	0	↓	1
	SHAR.W Rd	W	2											↑	↓	0	↓	1
	SHAR.W #2,Rd	W	2								MSB → LSB C			↑	↓	0	↓	1
	SHAR.L ERd	L	2											↑	↓	0	↓	1
	SHAR.L #2,ERd	L	2											↑	↓	0	↓	1
SHLL	SHLL.B Rd	B	2											↑	↓	0	↓	1
	SHLL.B #2,Rd	B	2											↑	↓	0	↓	1
	SHLL.W Rd	W	2											↑	↓	0	↓	1
	SHLL.W #2,Rd	W	2								C MSB ← LSB			↑	↓	0	↓	1
	SHLL.L ERd	L	2											↑	↓	0	↓	1
	SHLL.L #2,ERd	L	2											↑	↓	0	↓	1

Table A-1 Instruction Set (cont)

## (4) Shift Instructions (cont)

Mnemonic		Addressing Mode/ Instruction Length (Bytes)							Operation	Condition Code					No. of States*1				
		Operand Size	#xx	Rn	@ERn	@ (d,ERn)	@-ERn/@ERn+	@aa		@ (d,PC)	@@aa		I	H		N	Z	V	C
SHLR	SHLR.B Rd	B	2															1	
	SHLR.B #2,Rd	B	2															1	
	SHLR.W Rd	W	2															1	
	SHLR.W #2,Rd	W	2															1	
	SHLR.L ERd	L	2															1	
	SHLR.L #2,ERd	L	2															1	
ROTXL	ROTXL.B Rd	B	2															1	
	ROTXL.B #2,Rd	B	2															1	
	ROTXL.W Rd	W	2															1	
	ROTXL.W #2,Rd	W	2															1	
	ROTXL.L ERd	L	2															1	
	ROTXL.L #2,ERd	L	2															1	
ROTXR	ROTXR.B Rd	B	2															1	
	ROTXR.B #2,Rd	B	2															1	
	ROTXR.W Rd	W	2															1	
	ROTXR.W #2,Rd	W	2															1	
	ROTXR.L ERd	L	2															1	
	ROTXR.L #2,ERd	L	2															1	

Table A-1 Instruction Set (cont)

(4) Shift Instructions (cont)

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1		
			#xx	Rn	@ ERn	@ (d,ERn)	@ -ERn/@ ERn+	@ aa	@ (d,PC)	@ @aa			I	H	N	Z		V	C
			ROTL	ROTL.B Rd	B	2													
	ROTL.B #2,Rd	B	2														1		
	ROTL.W Rd	W	2														1		
	ROTL.W #2,Rd	W	2														1		
	ROTL.L ERd	L	2														1		
	ROTL.L #2,ERd	L	2														1		
ROTR	ROTR.B Rd	B	2														1		
	ROTR.B #2,Rd	B	2														1		
	ROTR.W Rd	W	2														1		
	ROTR.W #2,Rd	W	2														1		
	ROTR.L ERd	L	2														1		
	ROTR.L #2,ERd	L	2														1		

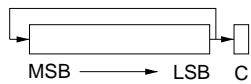
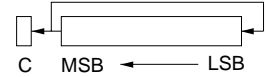


Table A-1 Instruction Set

## (5) Bit-Manipulation Instructions

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code						No. of States*1
			#xx	Rn	@ERn	@(d,ERn)	@_ERn/@ERn+	@aa	@(d,PC)	@@aa		I	H	N	Z	V	C	
														Advanced				
BSET	BSET #xx:3,Rd	B	2														1	
	BSET #xx:3,@ERd	B		4													4	
	BSET #xx:3,@aa:8	B						4									4	
	BSET #xx:3,@aa:16	B						6									5	
	BSET #xx:3,@aa:32	B						8									6	
	BSET Rn,Rd	B	2														1	
	BSET Rn,@ERd	B		4													4	
	BSET Rn,@aa:8	B						4									4	
	BSET Rn,@aa:16	B						6									5	
	BSET Rn,@aa:32	B						8									6	
BCLR	BCLR #xx:3,Rd	B	2														1	
	BCLR #xx:3,@ERd	B		4													4	
	BCLR #xx:3,@aa:8	B						4									4	
	BCLR #xx:3,@aa:16	B						6									5	
	BCLR #xx:3,@aa:32	B						8									6	
	BCLR Rn,Rd	B	2														1	
	BCLR Rn,@ERd	B		4													4	
	BCLR Rn,@aa:8	B						4									4	
	BCLR Rn,@aa:16	B						6									5	



**Table A-1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)							Operation	Condition Code					No. of States* <sup>1</sup>			
			#xx	Rn	@ERn	@ (d,ERn)	@-ERn/@ERn+	@aa	@ (d,PC)		@aa		I	H	N	Z	V	C	Advanced
BCLR	BCLR Rn, @aa:32	B						8									6		
BNOT	BNOT #xx:3, Rd	B	2														1		
	BNOT #xx:3, @ERd	B		4													4		
	BNOT #xx:3, @aa:8	B						4									4		
	BNOT #xx:3, @aa:16	B						6									5		
	BNOT #xx:3, @aa:32	B						8									6		
	BNOT Rn, Rd	B	2														1		
	BNOT Rn, @ERd	B		4													4		
	BNOT Rn, @aa:8	B						4									4		
	BNOT Rn, @aa:16	B						6									5		
	BNOT Rn, @aa:32	B						8									6		
BTST	BTST #xx:3, Rd	B	2											↓			1		
	BTST #xx:3, @ERd	B		4										↓			3		
	BTST #xx:3, @aa:8	B						4						↓			3		
	BTST #xx:3, @aa:16	B						6						↓			4		

Table A-1 Instruction Set (cont)

## (5) Bit-Manipulation Instructions (cont)

	Mnemonic	Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1					
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa			I	H	N	Z		V	C			
			Advanced																			
BTST	BTST #xx:3,@aa:32	B						8							¬ (#xx:3 of @aa:32)→Z	—	—	—	↓	—	—	5
	BTST Rn,Rd	B	2												¬ (Rn8 of Rd8)→Z	—	—	—	↓	—	—	1
	BTST Rn,@ERd	B		4											¬ (Rn8 of @ERd)→Z	—	—	—	↓	—	—	3
	BTST Rn,@aa:8	B						4							¬ (Rn8 of @aa:8)→Z	—	—	—	↓	—	—	3
	BTST Rn,@aa:16	B						6							¬ (Rn8 of @aa:16)→Z	—	—	—	↓	—	—	4
	BTST Rn,@aa:32	B						8							¬ (Rn8 of @aa:32)→Z	—	—	—	↓	—	—	5
BLD	BLD #xx:3,Rd	B	2												(#xx:3 of Rd8)→C	—	—	—	—	—	↓	1
	BLD #xx:3,@ERd	B		4											(#xx:3 of @ERd)→C	—	—	—	—	—	↓	3
	BLD #xx:3,@aa:8	B						4							(#xx:3 of @aa:8)→C	—	—	—	—	—	↓	3
	BLD #xx:3,@aa:16	B						6							(#xx:3 of @aa:16)→C	—	—	—	—	—	↓	4
	BLD #xx:3,@aa:32	B						8							(#xx:3 of @aa:32)→C	—	—	—	—	—	↓	5
BILD	BILD #xx:3,Rd	B	2												¬ (#xx:3 of Rd8)→C	—	—	—	—	—	↓	1
	BILD #xx:3,@ERd	B		4											¬ (#xx:3 of @ERd)→C	—	—	—	—	—	↓	3
	BILD #xx:3,@aa:8	B						4							¬ (#xx:3 of @aa:8)→C	—	—	—	—	—	↓	3
	BILD #xx:3,@aa:16	B						6							¬ (#xx:3 of @aa:16)→C	—	—	—	—	—	↓	4
	BILD #xx:3,@aa:32	B						8							¬ (#xx:3 of @aa:32)→C	—	—	—	—	—	↓	5
BST	BST #xx:3,Rd	B	2												C→(#xx:3 of Rd8)	—	—	—	—	—	—	1
	BST #xx:3,@ERd	B		4											C→(#xx:3 of @ERd)	—	—	—	—	—	—	4
	BST #xx:3,@aa:8	B						4							C→(#xx:3 of @aa:8)	—	—	—	—	—	—	4

**Table A-1 Instruction Set (cont)**

**(5) Bit-Manipulation Instructions (cont)**

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code					No. of States*1		
			#xx	Rn	@ERn	@ (d,ERn)	@ -ERn/@ERn+	@aa	@ (d,PC)	@aa			I	H	N	Z	V	C	Advanced
BST	BST #xx:3,@aa:16	B						6									5		
	BST #xx:3,@aa:32	B						8									6		
BIST	BIST #xx:3,Rd	B	2								$\neg C \rightarrow (\#xx:3 \text{ of } Rd8)$						1		
	BIST #xx:3,@ERd	B		4							$\neg C \rightarrow (\#xx:3 \text{ of } @ERd)$						4		
	BIST #xx:3,@aa:8	B						4			$\neg C \rightarrow (\#xx:3 \text{ of } @aa:8)$						4		
	BIST #xx:3,@aa:16	B						6			$\neg C \rightarrow (\#xx:3 \text{ of } @aa:16)$						5		
	BIST #xx:3,@aa:32	B						8			$\neg C \rightarrow (\#xx:3 \text{ of } @aa:32)$						6		
	BAND	BAND #xx:3,Rd	B	2								$C \wedge (\#xx:3 \text{ of } Rd8) \rightarrow C$					$\updownarrow$	1	
	BAND #xx:3,@ERd	B		4							$C \wedge (\#xx:3 \text{ of } @ERd) \rightarrow C$					$\updownarrow$	3		
	BAND #xx:3,@aa:8	B						4			$C \wedge (\#xx:3 \text{ of } @aa:8) \rightarrow C$					$\updownarrow$	3		
	BAND #xx:3,@aa:16	B						6			$C \wedge (\#xx:3 \text{ of } @aa:16) \rightarrow C$					$\updownarrow$	4		
	BAND #xx:3,@aa:32	B						8			$C \wedge (\#xx:3 \text{ of } @aa:32) \rightarrow C$					$\updownarrow$	5		
BIAND	BIAND #xx:3,Rd	B	2								$C \wedge [\neg (\#xx:3 \text{ of } Rd8)] \rightarrow C$					$\updownarrow$	1		
	BIAND #xx:3,@ERd	B		4							$C \wedge [\neg (\#xx:3 \text{ of } @ERd)] \rightarrow C$					$\updownarrow$	3		
	BIAND #xx:3,@aa:8	B						4			$C \wedge [\neg (\#xx:3 \text{ of } @aa:8)] \rightarrow C$					$\updownarrow$	3		
	BIAND #xx:3,@aa:16	B						6			$C \wedge [\neg (\#xx:3 \text{ of } @aa:16)] \rightarrow C$					$\updownarrow$	4		
	BIAND #xx:3,@aa:32	B						8			$C \wedge [\neg (\#xx:3 \text{ of } @aa:32)] \rightarrow C$					$\updownarrow$	5		
BOR	BOR #xx:3,Rd	B	2								$C \vee (\#xx:3 \text{ of } Rd8) \rightarrow C$					$\updownarrow$	1		
	BOR #xx:3,@ERd	B		4							$C \vee (\#xx:3 \text{ of } @ERd) \rightarrow C$					$\updownarrow$	3		

Table A-1 Instruction Set (cont)

## (5) Bit-Manipulation Instructions (cont)

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)							Operation	Condition Code					No. of States <sup>*1</sup>			
			#xx	Rn	@ERn	@ (d,ERn)	@-ERn/@ERn+	@aa	@ (d,PC)		@aa		I	H	N	Z	V	C	Advanced
BOR	BOR #xx:3,@aa:8	B						4							↓		3		
	BOR #xx:3,@aa:16	B						6							↓		4		
	BOR #xx:3,@aa:32	B						8							↓		5		
BIOR	BIOR #xx:3,Rd	B	2												↓		1		
	BIOR #xx:3,@ERd	B		4											↓		3		
	BIOR #xx:3,@aa:8	B						4							↓		3		
	BIOR #xx:3,@aa:16	B						6							↓		4		
	BIOR #xx:3,@aa:32	B						8							↓		5		
BXOR	BXOR #xx:3,Rd	B	2												↓		1		
	BXOR #xx:3,@ERd	B		4											↓		3		
	BXOR #xx:3,@aa:8	B						4							↓		3		
	BXOR #xx:3,@aa:16	B						6							↓		4		
	BXOR #xx:3,@aa:32	B						8							↓		5		
BIXOR	BIXOR #xx:3,Rd	B	2												↓		1		
	BIXOR #xx:3,@ERd	B		4											↓		3		
	BIXOR #xx:3,@aa:8	B						4							↓		3		
	BIXOR #xx:3,@aa:16	B						6							↓		4		
	BIXOR #xx:3,@aa:32	B						8							↓		5		

**Table A-1 Instruction Set**

**(6) Branch Instructions**

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)							Operation	Branching Condition	Condition Code					No. of States*1		
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)			@@aa	I	H	N	Z		V	C
														Advanced					
Bcc	BRA d:8(BT d:8)	—							2	if condition is true then PC←PC+d else next;	Always	—	—	—	—	—	—	2	
	BRA d:16(BT d:16)	—							4			—	—	—	—	—	—	—	3
	BRN d:8(BF d:8)	—							2		Never	—	—	—	—	—	—	—	2
	BRN d:16(BF d:16)	—							4			—	—	—	—	—	—	—	3
	BHI d:8	—							2		CvZ=0	—	—	—	—	—	—	—	2
	BHI d:16	—							4			—	—	—	—	—	—	—	3
	BLS d:8	—							2		CvZ=1	—	—	—	—	—	—	—	2
	BLS d:16	—							4			—	—	—	—	—	—	—	3
	BCC d:8(BHS d:8)	—							2		C=0	—	—	—	—	—	—	—	2
	BCC d:16(BHS d:16)	—							4			—	—	—	—	—	—	—	3
	BCS d:8(BLO d:8)	—							2		C=1	—	—	—	—	—	—	—	2
	BCS d:16(BLO d:16)	—							4			—	—	—	—	—	—	—	3
	BNE d:8	—							2		Z=0	—	—	—	—	—	—	—	2
	BNE d:16	—							4			—	—	—	—	—	—	—	3
	BEQ d:8	—							2		Z=1	—	—	—	—	—	—	—	2
	BEQ d:16	—							4			—	—	—	—	—	—	—	3
BVC d:8	—							2	V=0	—	—	—	—	—	—	—	2		
BVC d:16	—							4		—	—	—	—	—	—	—	3		

Table A-1 Instruction Set (cont)

## (6) Branch Instructions (cont)

Mnemonic		Operand Size	Addressing Mode/ Instruction Length (Bytes)							Operation	Branching Condition	Condition Code					No. of States* <sup>1</sup>		
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)			@@aa		I	H	N		Z	V
													Advanced						
Bcc	BVS d:8	—						2			V=1	—	—	—	—	—	—	2	
	BVS d:16	—						4			V=1	—	—	—	—	—	—	3	
	BPL d:8	—						2			N=0	—	—	—	—	—	—	2	
	BPL d:16	—						4			N=0	—	—	—	—	—	—	3	
	BMI d:8	—						2			N=1	—	—	—	—	—	—	2	
	BMI d:16	—						4			N=1	—	—	—	—	—	—	3	
	BGE d:8	—						2			N@V=0	—	—	—	—	—	—	2	
	BGE d:16	—						4			N@V=0	—	—	—	—	—	—	3	
	BLT d:8	—						2			N@V=1	—	—	—	—	—	—	2	
	BLT d:16	—						4			N@V=1	—	—	—	—	—	—	3	
	BGT d:8	—						2			Zv(N@V)=0	—	—	—	—	—	—	2	
	BGT d:16	—						4			Zv(N@V)=0	—	—	—	—	—	—	3	
	BLE d:8	—						2			Zv(N@V)=1	—	—	—	—	—	—	2	
	BLE d:16	—						4			Zv(N@V)=1	—	—	—	—	—	—	3	

**Table A-1 Instruction Set (cont)**

**(6) Branch Instructions (cont)**

Mnemonic		Addressing Mode/ Instruction Length (Bytes)									Operation	Condition Code					No. of States* <sup>1</sup>		
		Operand Size	#xx	Rn	@ ERn	@ (d,ERn)	@ -ERn/@ ERn+	@ aa	@ (d,PC)	@ @aa			I	H	N	Z	V	C	Advanced
													—	—	—	—	—	—	—
JMP	JMP @ERn	—			2													2	
	JMP @aa:24	—						4										3	
	JMP @ @aa:8	—								2								5	
BSR	BSR d:8	—							2									4	
	BSR d:16	—						4										5	
JSR	JSR @ERn	—			2													4	
	JSR @aa:24	—						4										5	
	JSR @ @aa:8	—								2								6	
RTS	RTS	—									2							5	

**Table A-1 Instruction Set**  
**(7) System Control Instructions**

Mnemonic	Addressing Mode/ Instruction Length (Bytes)	Operand Size	Addressing Mode/ Instruction Length (Bytes)							Operation	Condition Code					No. of States*1							
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)		@@aa	I	H	N	Z		V	C					
														Advanced									
TRAPA	TRAPA #xx:2	—														PC→@-SP,CCR→@-SP, EXR→@-SP,<vector>→PC	1	—	—	—	—	—	8 [9]
RTE	RTE	—														EXR←@SP+,CCR←@SP+, PC←@SP+	↑	↑	↑	↑	↑	↑	5 [9]
SLEEP	SLEEP	—														Transition to power-down state	—	—	—	—	—	—	2
LDC	LDC #xx:8,CCR	B	2													#xx:8→CCR	↑	↑	↑	↑	↑	↑	1
	LDC #xx:8,EXR	B	4													#xx:8→EXR	—	—	—	—	—	—	2
	LDC Rs,CCR	B	2													Rs8→CCR	↑	↑	↑	↑	↑	↑	1
	LDC Rs,EXR	B	2													Rs8→EXR	—	—	—	—	—	—	1
	LDC @ERs,CCR	W		4												@ERs→CCR	↑	↑	↑	↑	↑	↑	3
	LDC @ERs,EXR	W		4												@ERs→EXR	—	—	—	—	—	—	3
	LDC @(d:16,ERs),CCR	W			6											@(d:16,ERs)→CCR	↑	↑	↑	↑	↑	↑	4
	LDC @(d:16,ERs),EXR	W			6											@(d:16,ERs)→EXR	—	—	—	—	—	—	4
	LDC @(d:32,ERs),CCR	W			10											@(d:32,ERs)→CCR	↑	↑	↑	↑	↑	↑	6
	LDC @(d:32,ERs),EXR	W			10											@(d:32,ERs)→EXR	—	—	—	—	—	—	6
	LDC @ERs+,CCR	W				4										@ERs→CCR,ERs32+2→ERs32	↑	↑	↑	↑	↑	↑	4
	LDC @ERs+,EXR	W				4										@ERs→EXR,ERs32+2→ERs32	—	—	—	—	—	—	4
	LDC @aa:16,CCR	W					6									@aa:16→CCR	↑	↑	↑	↑	↑	↑	4
	LDC @aa:16,EXR	W					6									@aa:16→EXR	—	—	—	—	—	—	4
LDC @aa:32,CCR	W						8								@aa:32→CCR	↑	↑	↑	↑	↑	↑	5	
LDC @aa:32,EXR	W						8								@aa:32→EXR	—	—	—	—	—	—	5	



**Table A-1 Instruction Set (cont)**

**(7) System Control Instructions (cont)**

	Mnemonic	Operand Size	Addressing Mode/ Instruction Length (Bytes)							Operation	Condition Code					No. of States*1				
			#xx	Rn	@ ERn	@ (d,ERn)	@ -ERn/@ ERn+	@ aa	@ (d,PC)		@ @aa	I	H	N	Z		V	C		
														Advanced						
STC	STC CCR,Rd	B	2										CCR→Rd8	—	—	—	—	—	—	1
	STC EXR,Rd	B	2										EXR→Rd8	—	—	—	—	—	—	1
	STC CCR,@ERd	W		4									CCR→@ERd	—	—	—	—	—	—	3
	STC EXR,@ERd	W		4									EXR→@ERd	—	—	—	—	—	—	3
	STC CCR,@(d:16,ERd)	W			6								CCR→@(d:16,ERd)	—	—	—	—	—	—	4
	STC EXR,@(d:16,ERd)	W			6								EXR→@(d:16,ERd)	—	—	—	—	—	—	4
	STC CCR,@(d:32,ERd)	W			10								CCR→@(d:32,ERd)	—	—	—	—	—	—	6
	STC EXR,@(d:32,ERd)	W			10								EXR→@(d:32,ERd)	—	—	—	—	—	—	6
	STC CCR,@-ERd	W				4							ERd32-2→ERd32,CCR→@ERd	—	—	—	—	—	—	4
	STC EXR,@-ERd	W				4							ERd32-2→ERd32,EXR→@ERd	—	—	—	—	—	—	4
	STC CCR,@aa:16	W					6						CCR→@aa:16	—	—	—	—	—	—	4
	STC EXR,@aa:16	W					6						EXR→@aa:16	—	—	—	—	—	—	4
STC CCR,@aa:32	W						8					CCR→@aa:32	—	—	—	—	—	—	5	
STC EXR,@aa:32	W						8					EXR→@aa:32	—	—	—	—	—	—	5	
ANDC	ANDC #xx:8,CCR	B	2										CCR^#xx:8→CCR	↑	↓	↑	↓	↑	↓	1
	ANDC #xx:8,EXR	B	4										EXR^#xx:8→EXR	—	—	—	—	—	—	2
ORC	ORC #xx:8,CCR	B	2										CCR∨#xx:8→CCR	↑	↑	↑	↑	↑	↑	1
	ORC #xx:8,EXR	B	4										EXR∨#xx:8→EXR	—	—	—	—	—	—	2
XORC	XORC #xx:8,CCR	B	2										CCR@#xx:8→CCR	↑	↑	↑	↑	↑	↑	1
	XORC #xx:8,EXR	B	4										EXR@#xx:8→EXR	—	—	—	—	—	—	2
NOP	NOP	—									2		PC←PC+2	—	—	—	—	—	—	1

**Table A-1 Instruction Set**  
**(8) Block Transfer Instructions**

Mnemonic		Addressing Mode/ Instruction Length (Bytes)								Operation	Condition Code						No. of States*1	
		Operand Size									I	H	N	Z	V	C	Advanced	
		#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa									
EEPMOV	EEPMOV.B	—								4	if R4L≠0 Repeat @ER5→@ER6 ER5+1→ER5 ER6+1→ER6 R4L-1→R4L Until R4L=0 else next;	—	—	—	—	—	—	4+2n *2
	EEPMOV.W	—								4	if R4≠0 Repeat @ER5→@ER6 ER5+1→ER5 ER6+1→ER6 R4-1→R4 Until R4=0 else next;	—	—	—	—	—	—	4+2n *2

Notes: 1. The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.

2. n is the initial value of R4L or R4.

[1] Seven states for saving or restoring two registers, nine states for three registers, or eleven states for four registers.

[2] Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series.

[3] Set to 1 when a carry or borrow occurs at bit 11; otherwise cleared to 0.

[4] Set to 1 when a carry or borrow occurs at bit 27; otherwise cleared to 0.

[5] Retains its previous value when the result is zero; otherwise cleared to 0.

[6] Set to 1 when the divisor is negative; otherwise cleared to 0.

[7] Set to 1 when the divisor is zero; otherwise cleared to 0.

[8] Set to 1 when the quotient is negative; otherwise cleared to 0.

[9] One additional state is required for execution when EXR is valid.

## A.2 Instruction Codes

Table A-2 shows the instruction codes.

Table A-2 Instruction Codes

Instruction	Mnemonic	Size	Instruction Format											
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte		
ADD	ADD.B #xx:8,Rd	B	8	rd	IMM									
	ADD.B Rs,Rd	B	0	8	rs	rd								
	ADD.W #xx:16,Rd	W	7	9	1	rd	IMM							
	ADD.W Rs,Rd	W	0	9	rs	rd								
	ADD.L #xx:32,ERd	L	7	A	1	0	erd	IMM						
ADD.L ERs,ERd	L	0	A	1	ers	0	erd							
ADDS	ADDS #1,ERd	L	0	B	0	0	erd							
	ADDS #2,ERd	L	0	B	8	0	erd							
	ADDS #4,ERd	L	0	B	9	0	erd							
ADDX	ADDX #xx:8,Rd	B	9	rd	IMM									
	ADDX Rs,Rd	B	0	E	rs	rd								
AND	AND.B #xx:8,Rd	B	E	rd	IMM									
	AND.B Rs,Rd	B	1	6	rs	rd								
	AND.W #xx:16,Rd	W	7	9	6	rd	IMM							
	AND.W Rs,Rd	W	6	6	rs	rd								
	AND.L #xx:32,ERd	L	7	A	6	0	erd	IMM						
AND.L ERs,ERd	L	0	1	F	0	6	6	0	ers	0	erd			
ANDC	ANDC #xx:8,CCR	B	0	6	IMM									
	ANDC #xx:8,EXR	B	0	1	4	1	0	6	IMM					
BAND	BAND #xx:3,Rd	B	7	6	0	IMM	rd							
	BAND #xx:3,@ERd	B	7	C	0	erd	0	7	6	0	IMM	0		
	BAND #xx:3,@aa:8	B	7	E		abs		7	6	0	IMM	0		
	BAND #xx:3,@aa:16	B	6	A	1	0		abs		7	6	0	IMM	0
	BAND #xx:3,@aa:32	B	6	A	3	0		abs			7	6	0	IMM
Bcc	BRA d:8 (BT d:8)	—	4	0	disp									
	BRA d:16 (BT d:16)	—	5	8	0	0	disp							
	BRN d:8 (BF d:8)	—	4	1	disp									
	BRN d:16 (BF d:16)	—	5	8	1	0	disp							



Table A-2 Instruction Codes (cont)

Instruction	Mnemonic	Size	Instruction Format												
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte			
BCLR	BCLR #xx:3,Rd	B	7	2	0:IMM	rd									
	BCLR #xx:3,@ERd	B	7	D	0:erd	0	7	2	0:IMM	0					
	BCLR #xx:3,@aa:8	B	7	F	abs	7	2	0:IMM	0						
	BCLR #xx:3,@aa:16	B	6	A	1	8	abs		7	2	0:IMM	0			
	BCLR #xx:3,@aa:32	B	6	A	3	8	abs				7	2	0:IMM	0	
	BCLR Rn,Rd	B	6	2	rn	rd									
	BCLR Rn,@ERd	B	7	D	0:erd	0	6	2	rn	0					
	BCLR Rn,@aa:8	B	7	F	abs	6	2	rn	0						
	BCLR Rn,@aa:16	B	6	A	1	8	abs		6	2	rn	0			
BCLR Rn,@aa:32	B	6	A	3	8	abs				6	2	rn	0		
BIAND	BIAND #xx:3,Rd	B	7	6	1:IMM	rd									
	BIAND #xx:3,@ERd	B	7	C	0:erd	0	7	6	1:IMM	0					
	BIAND #xx:3,@aa:8	B	7	E	abs	7	6	1:IMM	0						
	BIAND #xx:3,@aa:16	B	6	A	1	0	abs		7	6	1:IMM	0			
	BIAND #xx:3,@aa:32	B	6	A	3	0	abs				7	6	1:IMM	0	
BILD	BILD #xx:3,Rd	B	7	7	1:IMM	rd									
	BILD #xx:3,@ERd	B	7	C	0:erd	0	7	7	1:IMM	0					
	BILD #xx:3,@aa:8	B	7	E	abs	7	7	1:IMM	0						
	BILD #xx:3,@aa:16	B	6	A	1	0	abs		7	7	1:IMM	0			
	BILD #xx:3,@aa:32	B	6	A	3	0	abs				7	7	1:IMM	0	
BIOR	BIOR #xx:3,Rd	B	7	4	1:IMM	rd									
	BIOR #xx:3,@ERd	B	7	C	0:erd	0	7	4	1:IMM	0					
	BIOR #xx:3,@aa:8	B	7	E	abs	7	4	1:IMM	0						
	BIOR #xx:3,@aa:16	B	6	A	1	0	abs		7	4	1:IMM	0			
	BIOR #xx:3,@aa:32	B	6	A	3	0	abs				7	4	1:IMM	0	

**Table A-2 Instruction Codes (cont)**

Instruction	Mnemonic	Size	Instruction Format																	
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte								
BIST	BIST #xx:3,Rd	B	6	7	1:IMM	rd														
	BIST #xx:3,@ERd	B	7	D	0:erd	0	6	7	1:IMM	0										
	BIST #xx:3,@aa:8	B	7	F	abs	6	7	1:IMM	0											
	BIST #xx:3,@aa:16	B	6	A	1	8	abs		6	7	1:IMM	0								
	BIST #xx:3,@aa:32	B	6	A	3	8	abs				6	7	1:IMM	0						
BIXOR	BIXOR #xx:3,Rd	B	7	5	1:IMM	rd														
	BIXOR #xx:3,@ERd	B	7	C	0:erd	0	7	5	1:IMM	0										
	BIXOR #xx:3,@aa:8	B	7	E	abs	7	5	1:IMM	0											
	BIXOR #xx:3,@aa:16	B	6	A	1	0	abs		7	5	1:IMM	0								
	BIXOR #xx:3,@aa:32	B	6	A	3	0	abs				7	5	1:IMM	0						
BLD	BLD #xx:3,Rd	B	7	7	0:IMM	rd														
	BLD #xx:3,@ERd	B	7	C	0:erd	0	7	7	0:IMM	0										
	BLD #xx:3,@aa:8	B	7	E	abs	7	7	0:IMM	0											
	BLD #xx:3,@aa:16	B	6	A	1	0	abs		7	7	0:IMM	0								
	BLD #xx:3,@aa:32	B	6	A	3	0	abs				7	7	0:IMM	0						
BNOT	BNOT #xx:3,Rd	B	7	1	0:IMM	rd														
	BNOT #xx:3,@ERd	B	7	D	0:erd	0	7	1	0:IMM	0										
	BNOT #xx:3,@aa:8	B	7	F	abs	7	1	0:IMM	0											
	BNOT #xx:3,@aa:16	B	6	A	1	8	abs		7	1	0:IMM	0								
	BNOT #xx:3,@aa:32	B	6	A	3	8	abs				7	1	0:IMM	0						
	BNOT Rn,Rd	B	6	1	rn	rd														
	BNOT Rn,@ERd	B	7	D	0:erd	0	6	1	rn	0										
	BNOT Rn,@aa:8	B	7	F	abs	6	1	rn	0											
	BNOT Rn,@aa:16	B	6	A	1	8	abs		6	1	rn	0								
	BNOT Rn,@aa:32	B	6	A	3	8	abs				6	1	rn	0						

Table A-2 Instruction Codes (cont)

Instruction	Mnemonic	Size	Instruction Format												
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte			
BOR	BOR #xx:3,Rd	B	7	4	0:IMM	rd									
	BOR #xx:3,@ERd	B	7	C	0:erd	0	7	4	0:IMM	0					
	BOR #xx:3,@aa:8	B	7	E	abs	7	4	0:IMM	0						
	BOR #xx:3,@aa:16	B	6	A	1	0	abs		7	4	0:IMM	0			
	BOR #xx:3,@aa:32	B	6	A	3	0	abs				7	4	0:IMM	0	
BSET	BSET #xx:3,Rd	B	7	0	0:IMM	rd									
	BSET #xx:3,@ERd	B	7	D	0:erd	0	7	0	0:IMM	0					
	BSET #xx:3,@aa:8	B	7	F	abs	7	0	0:IMM	0						
	BSET #xx:3,@aa:16	B	6	A	1	8	abs		7	0	0:IMM	0			
	BSET #xx:3,@aa:32	B	6	A	3	8	abs				7	0	0:IMM	0	
	BSET Rn,Rd	B	6	0	rn	rd									
	BSET Rn,@ERd	B	7	D	0:erd	0	6	0	rn	0					
	BSET Rn,@aa:8	B	7	F	abs	6	0	rn	0						
	BSET Rn,@aa:16	B	6	A	1	8	abs		6	0	rn	0			
	BSET Rn,@aa:32	B	6	A	3	8	abs				6	0	rn	0	
BSR	BSR d:8	—	5	5	disp										
	BSR d:16	—	5	C	0	0	disp								
BST	BST #xx:3,Rd	B	6	7	0:IMM	rd									
	BST #xx:3,@ERd	B	7	D	0:erd	0	6	7	0:IMM	0					
	BST #xx:3,@aa:8	B	7	F	abs	6	7	0:IMM	0						
	BST #xx:3,@aa:16	B	6	A	1	8	abs		6	7	0:IMM	0			
	BST #xx:3,@aa:32	B	6	A	3	8	abs				6	7	0:IMM	0	
BTST	BTST #xx:3,Rd	B	7	3	0:IMM	rd									
	BTST #xx:3,@ERd	B	7	C	0:erd	0	7	3	0:IMM	0					
	BTST #xx:3,@aa:8	B	7	E	abs	7	3	0:IMM	0						
	BTST #xx:3,@aa:16	B	6	A	1	0	abs		7	3	0:IMM	0			
	BTST #xx:3,@aa:32	B	6	A	3	0	abs				7	3	0:IMM	0	
	BTST Rn,Rd	B	6	3	rn	rd									
	BTST Rn,@ERd	B	7	C	0:erd	0	6	3	rn	0					



**Table A-2 Instruction Codes (cont)**

Instruction	Mnemonic	Size	Instruction Format												
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte			
BTST	BTST Rn,@aa:8	B	7	E	abs	6	3	rn	0						
	BTST Rn,@aa:16	B	6	A	1	0	abs			6	3	rn	0		
	BTST Rn,@aa:32	B	6	A	3	0	abs				6	3	rn	0	
BXOR	BXOR #xx:3,Rd	B	7	5	0:IMM: rd										
	BXOR #xx:3,@ERd	B	7	C	0:erd 0	7	5	0:IMM: 0							
	BXOR #xx:3,@aa:8	B	7	E	abs	7	5	0:IMM: 0							
	BXOR #xx:3,@aa:16	B	6	A	1	0	abs			7	5	0:IMM: 0			
	BXOR #xx:3,@aa:32	B	6	A	3	0	abs				7	5	0:IMM: 0		
CLRMAC	CLRMAC	—	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series												
CMP	CMP.B #xx:8,Rd	B	A	rd	IMM										
	CMP.B Rs,Rd	B	1	C	rs	rd									
	CMP.W #xx:16,Rd	W	7	9	2	rd	IMM								
	CMP.W Rs,Rd	W	1	D	rs	rd									
	CMP.L #xx:32,ERd	L	7	A	2	0:erd	IMM								
	CMP.L ERs,ERd	L	1	F	1:ers	0:erd									
DAA	DAA Rd	B	0	F	0	rd									
DAS	DAS Rd	B	1	F	0	rd									
DEC	DEC.B Rd	B	1	A	0	rd									
	DEC.W #1,Rd	W	1	B	5	rd									
	DEC.W #2,Rd	W	1	B	D	rd									
	DEC.L #1,ERd	L	1	B	7	0:erd									
	DEC.L #2,ERd	L	1	B	F	0:erd									
DIVXS	DIVXS.B Rs,Rd	B	0	1	D	0	5	1	rs	rd					
	DIVXS.W Rs,ERd	W	0	1	D	0	5	3	rs	0:erd					
DIVXU	DIVXU.B Rs,Rd	B	5	1	rs	rd									
	DIVXU.W Rs,ERd	W	5	3	rs	0:erd									
EEPMOV	EEPMOV.B	—	7	B	5	C	5	9	8	F					
	EEPMOV.W	—	7	B	D	4	5	9	8	F					

Table A-2 Instruction Codes (cont)

Instruction	Mnemonic	Size	Instruction Format																	
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte								
EXTS	EXTS.W Rd	W	1	7	D	rd														
	EXTS.L ERd	L	1	7	F	0:erd														
EXTU	EXTU.W Rd	W	1	7	5	rd														
	EXTU.L ERd	L	1	7	7	0:erd														
INC	INC.B Rd	B	0	A	0	rd														
	INC.W #1,Rd	W	0	B	5	rd														
	INC.W #2,Rd	W	0	B	D	rd														
	INC.L #1,ERd	L	0	B	7	0:erd														
	INC.L #2,ERd	L	0	B	F	0:erd														
JMP	JMP @ERn	—	5	9	0:ern	0														
	JMP @aa:24	—	5	A	abs															
	JMP @ @aa:8	—	5	B	abs															
JSR	JSR @ERn	—	5	D	0:ern	0														
	JSR @aa:24	—	5	E	abs															
	JSR @ @aa:8	—	5	F	abs															
LDC	LDC #xx:8,CCR	B	0	7	IMM															
	LDC #xx:8,EXR	B	0	1	4	1	0	7	IMM											
	LDC Rs,CCR	B	0	3	0	rs														
	LDC Rs,EXR	B	0	3	1	rs														
	LDC @ERs,CCR	W	0	1	4	0	6	9	0:ers	0										
	LDC @ERs,EXR	W	0	1	4	1	6	9	0:ers	0										
	LDC @(d:16,ERs),CCR	W	0	1	4	0	6	F	0:ers	0	disp									
	LDC @(d:16,ERs),EXR	W	0	1	4	1	6	F	0:ers	0	disp									
	LDC @(d:32,ERs),CCR	W	0	1	4	0	7	8	0:ers	0	6	B	2	0	disp					
	LDC @(d:32,ERs),EXR	W	0	1	4	1	7	8	0:ers	0	6	B	2	0	disp					
	LDC @ERs+,CCR	W	0	1	4	0	6	D	0:ers	0										
	LDC @ERs+,EXR	W	0	1	4	1	6	D	0:ers	0										
	LDC @aa:16,CCR	W	0	1	4	0	6	B	0	0	abs									
LDC @aa:16,EXR	W	0	1	4	1	6	B	0	0	abs										

**Table A-2 Instruction Codes (cont)**

Instruction	Mnemonic	Size	Instruction Format													
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte				
LDC	LDC @aa:32,CCR	W	0	1	4	0	6	B	2	0	abs					
	LDC @aa:32,EXR	W	0	1	4	1	6	B	2	0	abs					
LDM	LDM.L @SP+, (ERn-ERn+1)	L	0	1	1	0	6	D	7	0;ern+1						
	LDM.L @SP+, (ERn-ERn+2)	L	0	1	2	0	6	D	7	0;ern+2						
	LDM.L @SP+, (ERn-ERn+3)	L	0	1	3	0	6	D	7	0;ern+3						
LDMAC	LDMAC ERs, MACH	L	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series													
	LDMAC ERs, MACL	L														
MAC	MAC @ERn+, @ERm+	—														
MOV	MOV.B #xx:8,Rd	B	F	rd	IMM											
	MOV.B Rs,Rd	B	0	C	rs rd											
	MOV.B @ERs,Rd	B	6	8	0;ers rd											
	MOV.B @(d:16,ERs),Rd	B	6	E	0;ers rd	disp										
	MOV.B @(d:32,ERs),Rd	B	7	8	0;ers 0	6	A	2	rd	disp						
	MOV.B @ERs+,Rd	B	6	C	0;ers rd											
	MOV.B @aa:8,Rd	B	2	rd	abs											
	MOV.B @aa:16,Rd	B	6	A	0 rd	abs										
	MOV.B @aa:32,Rd	B	6	A	2 rd	abs										
	MOV.B Rs, @ERd	B	6	8	1;erd rs											
	MOV.B Rs, @(d:16,ERd)	B	6	E	1;erd rs	disp										
	MOV.B Rs, @(d:32,ERd)	B	7	8	0;erd 0	6	A	A	rs	disp						
	MOV.B Rs, @-ERd	B	6	C	1;erd rs											
	MOV.B Rs, @aa:8	B	3	rs	abs											
	MOV.B Rs, @aa:16	B	6	A	8 rs	abs										
	MOV.B Rs, @aa:32	B	6	A	A rs	abs										
	MOV.W #xx:16,Rd	W	7	9	0 rd	IMM										
	MOV.W Rs,Rd	W	0	D	rs rd											
	MOV.W @ERs,Rd	W	6	9	0;ers rd											
	MOV.W @(d:16,ERs),Rd	W	6	F	0;ers rd	disp										
MOV.W @(d:32,ERs),Rd	W	7	8	0;ers 0	6	B	2	rd	disp							

Table A-2 Instruction Codes (cont)

Instruction	Mnemonic	Size	Instruction Format																	
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte								
MOV	MOV.W @ERs+,Rd	W	6	D	0:ers	rd														
	MOV.W @aa:16,Rd	W	6	B	0	rd	abs													
	MOV.W @aa:32,Rd	W	6	B	2	rd	abs													
	MOV.W Rs,@ERd	W	6	9	1:erd	rs														
	MOV.W Rs,@(d:16,ERd)	W	6	F	1:erd	rs	disp													
	MOV.W Rs,@(d:32,ERd)	W	7	8	0:erd	0	6	B	A	rs	disp									
	MOV.W Rs,@-ERd	W	6	D	1:erd	rs														
	MOV.W Rs,@aa:16	W	6	B	8	rs	abs													
	MOV.W Rs,@aa:32	W	6	B	A	rs	abs													
	MOV.L #xx:32,Rd	L	7	A	0	0:erd	IMM													
	MOV.L ERs,ERd	L	0	F	1:ers	0:erd														
	MOV.L @ERs,ERd	L	0	1	0	0	6	9	0:ers	0:erd										
	MOV.L @(d:16,ERs),ERd	L	0	1	0	0	6	F	0:ers	0:erd	disp									
	MOV.L @(d:32,ERs),ERd	L	0	1	0	0	7	8	0:ers	0	6	B	2	0:erd	disp					
	MOV.L @ERs+,ERd	L	0	1	0	0	6	D	0:ers	0:erd										
	MOV.L @aa:16,ERd	L	0	1	0	0	6	B	0	0:erd	abs									
	MOV.L @aa:32,ERd	L	0	1	0	0	6	B	2	0:erd	abs									
	MOV.L ERs,@ERd	L	0	1	0	0	6	9	1:erd	0:ers										
	MOV.L ERs,@(d:16,ERd)	L	0	1	0	0	6	F	1:erd	0:ers	disp									
	MOV.L ERs,@(d:32,ERd)*	L	0	1	0	0	7	8	0:erd	0	6	B	A	0:ers	disp					
MOV.L ERs,@-ERd	L	0	1	0	0	6	D	1:erd	0:ers											
MOV.L ERs,@aa:16	L	0	1	0	0	6	B	8	0:ers	abs										
MOV.L ERs,@aa:32	L	0	1	0	0	6	B	A	0:ers	abs										
MOVFPPE	MOVFPPE @aa:16,Rd	B	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series																	
MOVTPPE	MOVTPPE Rs,@aa:16	B																		
MULXS	MULXS.B Rs,Rd	B	0	1	C	0	5	0	rs	rd										
	MULXS.W Rs,ERd	W	0	1	C	0	5	2	rs	0:erd										
MULXU	MULXU.B Rs,Rd	B	5	0	rs	rd														
	MULXU.W Rs,ERd	W	5	2	rs	0:erd														

**Table A-2 Instruction Codes (cont)**

Instruction	Mnemonic	Size	Instruction Format										
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte	
NEG	NEG.B Rd	B	1	7	8	rd							
	NEG.W Rd	W	1	7	9	rd							
	NEG.L ERd	L	1	7	B	:0:erd							
NOP	NOP	—	0	0	0	0							
NOT	NOT.B Rd	B	1	7	0	rd							
	NOT.W Rd	W	1	7	1	rd							
	NOT.L ERd	L	1	7	3	:0:erd							
OR	OR.B #xx:8,Rd	B	C	rd	IMM								
	OR.B Rs,Rd	B	1	4	rs	rd							
	OR.W #xx:16,Rd	W	7	9	4	rd	IMM						
	OR.W Rs,Rd	W	6	4	rs	rd							
	OR.L #xx:32,ERd	L	7	A	4	:0:erd	IMM						
	OR.L ERs,ERd	L	0	1	F	0	6	4	0:ers	:0:erd			
ORC	ORC #xx:8,CCR	B	0	4	IMM								
	ORC #xx:8,EXR	B	0	1	4	1	0	4	IMM				
POP	POP.W Rn	W	6	D	7	rn							
	POP.L ERn	L	0	1	0	0	6	D	7	:0:ern			
PUSH	PUSH.W Rn	W	6	D	F	rn							
	PUSH.L ERn	L	0	1	0	0	6	D	F	:0:ern			
ROTL	ROTL.B Rd	B	1	2	8	rd							
	ROTL.B #2, Rd	B	1	2	C	rd							
	ROTL.W Rd	W	1	2	9	rd							
	ROTL.W #2, Rd	W	1	2	D	rd							
	ROTL.L ERd	L	1	2	B	:0:erd							
	ROTL.L #2, ERd	L	1	2	F	:0:erd							

Table A-2 Instruction Codes (cont)

Instruction	Mnemonic	Size	Instruction Format																		
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte									
ROTR	ROTR.B Rd	B	1	3	8	rd															
	ROTR.B #2, Rd	B	1	3	C	rd															
	ROTR.W Rd	W	1	3	9	rd															
	ROTR.W #2, Rd	W	1	3	D	rd															
	ROTR.L ERd	L	1	3	B	0:erd															
	ROTR.L #2, ERd	L	1	3	F	0:erd															
ROTXL	ROTXL.B Rd	B	1	2	0	rd															
	ROTXL.B #2, Rd	B	1	2	4	rd															
	ROTXL.W Rd	W	1	2	1	rd															
	ROTXL.W #2, Rd	W	1	2	5	rd															
	ROTXL.L ERd	L	1	2	3	0:erd															
	ROTXL.L #2, ERd	L	1	2	7	0:erd															
ROTXR	ROTXR.B Rd	B	1	3	0	rd															
	ROTXR.B #2, Rd	B	1	3	4	rd															
	ROTXR.W Rd	W	1	3	1	rd															
	ROTXR.W #2, Rd	W	1	3	5	rd															
	ROTXR.L ERd	L	1	3	3	0:erd															
	ROTXR.L #2, ERd	L	1	3	7	0:erd															
RTE	RTE	—	5	6	7	0															
RTS	RTS	—	5	4	7	0															
SHAL	SHAL.B Rd	B	1	0	8	rd															
	SHAL.B #2, Rd	B	1	0	C	rd															
	SHAL.W Rd	W	1	0	9	rd															
	SHAL.W #2, Rd	W	1	0	D	rd															
	SHAL.L ERd	L	1	0	B	0:erd															
	SHAL.L #2, ERd	L	1	0	F	0:erd															

**Table A-2 Instruction Codes (cont)**

Instruction	Mnemonic	Size	Instruction Format																	
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte								
SHAR	SHAR.B Rd	B	1	1	8	rd														
	SHAR.B #2, Rd	B	1	1	C	rd														
	SHAR.W Rd	W	1	1	9	rd														
	SHAR.W #2, Rd	W	1	1	D	rd														
	SHAR.L ERd	L	1	1	B	0:erd														
	SHAR.L #2, ERd	L	1	1	F	0:erd														
SHLL	SHLL.B Rd	B	1	0	0	rd														
	SHLL.B #2, Rd	B	1	0	4	rd														
	SHLL.W Rd	W	1	0	1	rd														
	SHLL.W #2, Rd	W	1	0	5	rd														
	SHLL.L ERd	L	1	0	3	0:erd														
	SHLL.L #2, ERd	L	1	0	7	0:erd														
SHLR	SHLR.B Rd	B	1	1	0	rd														
	SHLR.B #2, Rd	B	1	1	4	rd														
	SHLR.W Rd	W	1	1	1	rd														
	SHLR.W #2, Rd	W	1	1	5	rd														
	SHLR.L ERd	L	1	1	3	0:erd														
	SHLR.L #2, ERd	L	1	1	7	0:erd														
SLEEP	SLEEP	—	0	1	8	0														
STC	STC.B CCR,Rd	B	0	2	0	rd														
	STC.B EXR,Rd	B	0	2	1	rd														
	STC.W CCR,@ERd	W	0	1	4	0	6	9	1:erd	0										
	STC.W EXR,@ERd	W	0	1	4	1	6	9	1:erd	0										
	STC.W CCR,@(d:16,ERd)	W	0	1	4	0	6	F	1:erd	0	disp									
	STC.W EXR,@(d:16,ERd)	W	0	1	4	1	6	F	1:erd	0	disp									
	STC.W CCR,@(d:32,ERd)	W	0	1	4	0	7	8	0:erd	0	6	B	A	0	disp					
	STC.W EXR,@(d:32,ERd)	W	0	1	4	1	7	8	0:erd	0	6	B	A	0	disp					
	STC.W CCR,@-ERd	W	0	1	4	0	6	D	1:erd	0										
	STC.W EXR,@-ERd	W	0	1	4	1	6	D	1:erd	0										

Table A-2 Instruction Codes (cont)

Instruction	Mnemonic	Size	Instruction Format											
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte		
STC	STC.W CCR,@aa:16	W	0	1	4	0	6	B	8	0	abs			
	STC.W EXR,@aa:16	W	0	1	4	1	6	B	8	0	abs			
	STC.W CCR,@aa:32	W	0	1	4	0	6	B	A	0	abs			
	STC.W EXR,@aa:32	W	0	1	4	1	6	B	A	0	abs			
STM	STM.L(ERn-ERn+1), @-SP	L	0	1	1	0	6	D	F	0:ern				
	STM.L(ERn-ERn+2), @-SP	L	0	1	2	0	6	D	F	0:ern				
	STM.L(ERn-ERn+3), @-SP	L	0	1	3	0	6	D	F	0:ern				
STMAC	STMAC MACH,ERd	L	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series											
	STMAC MACL,ERd	L												
SUB	SUB.B Rs,Rd	B	1	8	rs	rd								
	SUB.W #xx:16,Rd	W	7	9	3	rd	IMM							
	SUB.W Rs,Rd	W	1	9	rs	rd								
	SUB.L #xx:32,ERd	L	7	A	3	0:erd	IMM							
	SUB.L ERs,ERd	L	1	A	1:ers	0:erd								
SUBS	SUBS #1,ERd	L	1	B	0	0:erd								
	SUBS #2,ERd	L	1	B	8	0:erd								
	SUBS #4,ERd	L	1	B	9	0:erd								
SUBX	SUBX #xx:8,Rd	B	B	rd	IMM									
	SUBX Rs,Rd	B	1	E	rs	rd								
TAS	TAS @ERd	B	0	1	E	0	7	B	0:erd	C				
TRAPA	TRAPA #x:2	—	5	7	00:IMM	0								
XOR	XOR.B #xx:8,Rd	B	D	rd	IMM									
	XOR.B Rs,Rd	B	1	5	rs	rd								
	XOR.W #xx:16,Rd	W	7	9	5	rd	IMM							
	XOR.W Rs,Rd	W	6	5	rs	rd								
	XOR.L #xx:32,ERd	L	7	A	5	0:erd	IMM							
	XOR.L ERs,ERd	L	0	1	F	0	6	5	0:ers	0:erd				



**Table A-2 Instruction Codes (cont)**

Instruction	Mnemonic	Size	Instruction Format										
			1st byte	2nd byte	3rd byte	4th byte	5th byte	6th byte	7th byte	8th byte	9th byte	10th byte	
XORC	XORC #xx:8,CCR	B	0	5	IMM								
	XORC #xx:8,EXR	B	0	1	4	1	0	5	IMM				

Note: \* Bit 7 of the 4th byte of the MOV.L ERs, @(d:32,ERd) instruction can be either 1 or 0.

**Legend**

- IMM: Immediate data (2, 3, 8, 16, or 32 bits)
- abs: Absolute address (8, 16, 24, or 32 bits)
- disp: Displacement (8, 16, or 32 bits)
- rs, rd, rn: Register field (4 bits specifying an 8-bit or 16-bit register. The symbols rs, rd, and rn correspond to operand symbols Rs, Rd, and Rn.)
- ers, erd, ern, erm: Register field (3 bits specifying an address register or 32-bit register. The symbols ers, erd, ern, and erm correspond to operand symbols ERs, ERd, ERn, and ERm.)

The register fields specify general registers as follows.

Address Register 32-Bit Register		16-Bit Register		8-Bit Register	
Register Field	General Register	Register Field	General Register	Register Field	General Register
000	ER0	0000	R0	0000	R0H
001	ER1	0001	R1	0001	R1H
•	•	•	•	•	•
•	•	•	•	•	•
•	•	•	•	•	•
111	ER7	0111	R7	0111	R7H
		1000	E0	1000	R0L
		1001	E1	1001	R1L
		•	•	•	•
		•	•	•	•
		•	•	•	•
		1111	E7	1111	R7L

# A.3 Operation Code Map

Table A-3 shows the operation code map.

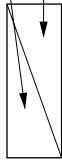
**Table A-3 Operation Code Map (1)**

Instruction code

Instruction when most significant bit of BH is 0.

Instruction when most significant bit of BH is 1.

1st byte		2nd byte	
AH	AL	BH	BL



AL/AH	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	Table A.3(2)	STC	LDC	ORC	XORC	ANDC	LDC	ADD	ADD	Table A.3(2)	Table A.3(2)	MOV	ADDX		Table A.3(2)
1	Table A.3(2)	Table A.3(2)	STMAC	LDMAC	OR	XOR	AND	Table A.3(2)	SUB	SUB	Table A.3(2)	Table A.3(2)	CMP	SUBX		Table A.3(2)
2	MOV.B															
3	MOV.B															
4	BRA	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU	MULXU	DIVXU	RTS	BSR	RTE	TRAPA	Table A.3(2)		JMP		BSR		JSR	
6	BSET	BNOT	BCLR	BTST	OR	XOR	AND	BSL	BSR	MOV	Table A.3(2)					
7					BOR	BXOR	BAND	BLD	MOV	MOV	Table A.3(2)	EEP	MOV			Table A.3(3)
8	ADD															
9	ADDX															
A	CMP															
B	SUBX															
C	OR															
D	XOR															
E	AND															
F	MOV															

Note: \* Cannot be used in the H8S/2338 Series, H8S/2328 Series, H8S/2318 Series, or H8S/2318 Series.

**Table A-3 Operation Code Map (2)**

Instruction code		1st byte		2nd byte	
AH	AL	BH	BL		

BH AH AL	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
01	MOV	LDM	STM		LDC		MAC*		SLEEP		CLRMAC*		Table A.3(3)	Table A.3(3)	TAS	Table A.3(3)
0A	INC								ADD							
0B	ADDS					INC		INC	ADDS					INC		INC
0F	DAA								MOV							
10	SHLL		SHLL			SHLL		SHAL	SHAL			SHAL		SHAL		
11	SHLR		SHLR			SHLR		SHAR	SHAR			SHAR		SHAR		
12	ROTXL		ROTXL			ROTXL		ROTL	ROTL			ROTL		ROTL		
13	ROTXR		ROTXR			ROTXR		ROTR	ROTR			ROTR		ROTR		
17	NOT		NOT		EXTU		EXTU		NEG	NEG		EXTS		EXTS		
1A	DEC								SUB							
1B	SUBS					DEC	DEC		SUBS			DEC		DEC		
1F	DAS								CMP							
58	BRA	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
6A	MOV	Table A.3(4)	MOV	Table A.3(4)	MOVFP*				MOV	MOV		MOVTPE*				
79	MOV	ADD	CMP	SUB	OR	XOR	AND									
7A	MOV	ADD	CMP	SUB	OR	XOR	AND									

Note: \* Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series.

**Table A-3 Operation Code Map (3)**

Instruction code	1st byte		2nd byte		3rd byte		4th byte	
	AH	AL	BH	BL	CH	CL	DH	DL



Instruction when most significant bit of DH is 0.

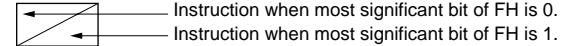
Instruction when most significant bit of DH is 1.

CL AH AL BH BL CH	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	01C05	MULXS		MULXS												
01D05		DIVXS		DIVXS												
01F06					OR	XOR	AND									
7Cr06 *1				BTST												
7Cr07 *1				BTST	BOR BIOR	BXOR BIXOR	BAND BIAND	BLD BILD								
7Dr06 *1	BSET	BNOT	BCLR					BST BIST								
7Dr07 *1	BSET	BNOT	BCLR													
7Eaa6 *2				BTST												
7Eaa7 *2				BTST	BOR BIOR	BXOR BIXOR	BAND BIAND	BLD BILD								
7Faa6 *2	BSET	BNOT	BCLR					BST BIST								
7Faa7 *2	BSET	BNOT	BCLR													

- Notes: 1. r is the register specification field.  
2. aa is the absolute address specification.

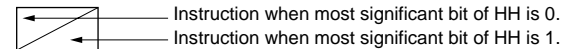
**Table A-3 Operation Code Map (4)**

Instruction code	1st byte		2nd byte		3rd byte		4th byte		5th byte		6th byte	
	AH	AL	BH	BL	CH	CL	DH	DL	EH	EL	FH	FL



<b>EL</b> AHALBHBLCHCLDHDLEH	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
6A10aaaa6*				BTST												
6A10aaaa7*					BOR	BXOR	BAND	BLD								
6A18aaaa6*	BSET	BNOT	BCLR		BIOR	BIXOR	BIAND	BST	BILD							
6A18aaaa7*									BST	BIST						

Instruction code	1st byte		2nd byte		3rd byte		4th byte		5th byte		6th byte		7th byte		8th byte	
	AH	AL	BH	BL	CH	CL	DH	DL	EH	EL	FH	FL	GH	GL	HH	HL



<b>GL</b> AHALBHBL... FHFLGH	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
6A30aaaaaaaa6*				BTST												
6A30aaaaaaaa7*					BOR	BXOR	BAND	BLD								
6A38aaaaaaaa6*	BSET	BNOT	BCLR		BIOR	BIXOR	BIAND	BST	BILD							
6A38aaaaaaaa7*										BST	BIST					

Note: \* aa is the absolute address specification.

## A.4 Number of States Required for Instruction Execution

The tables in this section can be used to calculate the number of states required for instruction execution by the CPU. Table A-5 indicates the number of instruction fetch, data read/write, and other cycles occurring in each instruction. Table A-4 indicates the number of states required for each cycle. The number of states required for execution of an instruction can be calculated from these two tables as follows:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

**Examples:** Advanced mode, program code and stack located in external memory, on-chip supporting modules accessed in two states with 8-bit bus width, external devices accessed in three states with one wait state and 16-bit bus width.

### 1. BSET #0, @FFFFC7:8

From table A-5:

$$I = L = 2, \quad J = K = M = N = 0$$

From table A-4:

$$S_I = 4, \quad S_L = 2$$

$$\text{Number of states required for execution} = 2 \times 4 + 2 \times 2 = 12$$

### 2. JSR @@30

From table A-5:

$$I = J = K = 2, \quad L = M = N = 0$$

From table A-4:

$$S_I = S_J = S_K = 4$$

$$\text{Number of states required for execution} = 2 \times 4 + 2 \times 4 + 2 \times 4 = 24$$

**Table A-4 Number of States per Cycle**

Cycle		Access Conditions						
		On-Chip Memory	On-Chip Supporting Module		External Device			
			8-Bit Bus	16-Bit Bus	8-Bit Bus		16-Bit Bus	
				2-State Access	3-State Access	2-State Access	3-State Access	
Instruction fetch	$S_i$	1	4	2	4	6 + 2m	2	3 + m
Branch address read	$S_j$							
Stack operation	$S_k$							
Byte data access	$S_l$		2		2	3 + m		
Word data access	$S_m$		4		4	6 + 2m		
Internal operation	$S_n$	1	1	1	1	1	1	1

Legend

m: Number of wait states inserted into external device access

**Table A-5 Number of Cycles in Instruction Execution**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address Read	Operation	Data Access	Data Access	Operation
		I	J	K	L	M	N
ADD	ADD.B #xx:8,Rd	1					
	ADD.B Rs,Rd	1					
	ADD.W #xx:16,Rd	2					
	ADD.W Rs,Rd	1					
	ADD.L #xx:32,ERd	3					
	ADD.L ERs,ERd	1					
ADDS	ADDS #1/2/4,ERd	1					
ADDX	ADDX #xx:8,Rd	1					
	ADDX Rs,Rd	1					
AND	AND.B #xx:8,Rd	1					
	AND.B Rs,Rd	1					
	AND.W #xx:16,Rd	2					
	AND.W Rs,Rd	1					
	AND.L #xx:32,ERd	3					
	AND.L ERs,ERd	2					
ANDC	ANDC #xx:8,CCR	1					
	ANDC #xx:8,EXR	2					
BAND	BAND #xx:3,Rd	1					
	BAND #xx:3,@ERd	2			1		
	BAND #xx:3,@aa:8	2			1		
	BAND #xx:3,@aa:16	3			1		
	BAND #xx:3,@aa:32	4			1		
Bcc	BRA d:8 (BT d:8)	2					
	BRN d:8 (BF d:8)	2					
	BHI d:8	2					
	BLS d:8	2					
	BCC d:8 (BHS d:8)	2					
	BCS d:8 (BLO d:8)	2					
	BNE d:8	2					
	BEQ d:8	2					
	BVC d:8	2					
	BVS d:8	2					
	BPL d:8	2					



**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address	Operation	Data	Data	
		I	J	K	L	M	N
Bcc	BMI d:8	2					
	BGE d:8	2					
	BLT d:8	2					
	BGT d:8	2					
	BLE d:8	2					
	BRA d:16 (BT d:16)	2					1
	BRN d:16 (BF d:16)	2					1
	BHI d:16	2					1
	BLS d:16	2					1
	BCC d:16 (BHS d:16)	2					1
	BCS d:16 (BLO d:16)	2					1
	BNE d:16	2					1
	BEQ d:16	2					1
	BVC d:16	2					1
	BVS d:16	2					1
	BPL d:16	2					1
	BMI d:16	2					1
	BGE d:16	2					1
	BLT d:16	2					1
	BGT d:16	2					1
BLE d:16	2					1	
BCLR	BCLR #xx:3,Rd	1					
	BCLR #xx:3,@ERd	2			2		
	BCLR #xx:3,@aa:8	2			2		
	BCLR #xx:3,@aa:16	3			2		
	BCLR #xx:3,@aa:32	4			2		
	BCLR Rn,Rd	1					
	BCLR Rn,@ERd	2			2		
	BCLR Rn,@aa:8	2			2		
	BCLR Rn,@aa:16	3			2		
	BCLR Rn,@aa:32	4			2		

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address	Operation	Data	Data	Operation
		I	J	K	L	M	N
BIAND	BIAND #xx:3,Rd	1					
	BIAND #xx:3,@ERd	2			1		
	BIAND #xx:3,@aa:8	2			1		
	BIAND #xx:3,@aa:16	3			1		
	BIAND #xx:3,@aa:32	4			1		
BILD	BILD #xx:3,Rd	1					
	BILD #xx:3,@ERd	2			1		
	BILD #xx:3,@aa:8	2			1		
	BILD #xx:3,@aa:16	3			1		
	BILD #xx:3,@aa:32	4			1		
BIOR	BIOR #xx:8,Rd	1					
	BIOR #xx:8,@ERd	2			1		
	BIOR #xx:8,@aa:8	2			1		
	BIOR #xx:8,@aa:16	3			1		
	BIOR #xx:8,@aa:32	4			1		
BIST	BIST #xx:3,Rd	1					
	BIST #xx:3,@ERd	2			2		
	BIST #xx:3,@aa:8	2			2		
	BIST #xx:3,@aa:16	3			2		
	BIST #xx:3,@aa:32	4			2		
BIXOR	BIXOR #xx:3,Rd	1					
	BIXOR #xx:3,@ERd	2			1		
	BIXOR #xx:3,@aa:8	2			1		
	BIXOR #xx:3,@aa:16	3			1		
	BIXOR #xx:3,@aa:32	4			1		
BLD	BLD #xx:3,Rd	1					
	BLD #xx:3,@ERd	2			1		
	BLD #xx:3,@aa:8	2			1		
	BLD #xx:3,@aa:16	3			1		
	BLD #xx:3,@aa:32	4			1		

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction Execution Cycles					
		Instruction Fetch	Branch Address Read	Stack Operation	Byte Data Access	Word Data Access	Internal Operation
		I	J	K	L	M	N
BNOT	BNOT #xx:3,Rd	1					
	BNOT #xx:3,@ERd	2			2		
	BNOT #xx:3,@aa:8	2			2		
	BNOT #xx:3,@aa:16	3			2		
	BNOT #xx:3,@aa:32	4			2		
	BNOT Rn,Rd	1					
	BNOT Rn,@ERd	2			2		
	BNOT Rn,@aa:8	2			2		
	BNOT Rn,@aa:16	3			2		
BNOT Rn,@aa:32	4			2			
BOR	BOR #xx:3,Rd	1					
	BOR #xx:3,@ERd	2			1		
	BOR #xx:3,@aa:8	2			1		
	BOR #xx:3,@aa:16	3			1		
	BOR #xx:3,@aa:32	4			1		
BSET	BSET #xx:3,Rd	1					
	BSET #xx:3,@ERd	2			2		
	BSET #xx:3,@aa:8	2			2		
	BSET #xx:3,@aa:16	3			2		
	BSET #xx:3,@aa:32	4			2		
	BSET Rn,Rd	1					
	BSET Rn,@ERd	2			2		
	BSET Rn,@aa:8	2			2		
	BSET Rn,@aa:16	3			2		
BSET Rn,@aa:32	4			2			
BSR	BSR d:8    Advanced	2		2			
	BSR d:16    Advanced	2		2			1
BST	BST #xx:3,Rd	1					
	BST #xx:3,@ERd	2			2		
	BST #xx:3,@aa:8	2			2		
	BST #xx:3,@aa:16	3			2		
	BST #xx:3,@aa:32	4			2		

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address Read	Operation	Data Access	Data Access	Operation
		I	J	K	L	M	N
BTST	BTST #xx:3,Rd	1					
	BTST #xx:3,@ERd	2			1		
	BTST #xx:3,@aa:8	2			1		
	BTST #xx:3,@aa:16	3			1		
	BTST #xx:3,@aa:32	4			1		
	BTST Rn,Rd	1					
	BTST Rn,@ERd	2			1		
	BTST Rn,@aa:8	2			1		
	BTST Rn,@aa:16	3			1		
	BTST Rn,@aa:32	4			1		
BXOR	BXOR #xx:3,Rd	1					
	BXOR #xx:3,@ERd	2			1		
	BXOR #xx:3,@aa:8	2			1		
	BXOR #xx:3,@aa:16	3			1		
	BXOR #xx:3,@aa:32	4			1		
CLRMAC	CLRMAC	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series					
CMP	CMP.B #xx:8,Rd	1					
	CMP.B Rs,Rd	1					
	CMP.W #xx:16,Rd	2					
	CMP.W Rs,Rd	1					
	CMP.L #xx:32,ERd	3					
	CMP.L ERs,ERd	1					
DAA	DAA Rd	1					
DAS	DAS Rd	1					
DEC	DEC.B Rd	1					
	DEC.W #1/2,Rd	1					
	DEC.L #1/2,ERd	1					
DIVXS	DIVXS.B Rs,Rd	2					11
	DIVXS.W Rs,ERd	2					19
DIVXU	DIVXU.B Rs,Rd	1					11
	DIVXU.W Rs,ERd	1					19

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction Fetch						
		I	J	K	L	M	N	
EEMOV	EEMOV.B	2			2n+2*2			
	EEMOV.W	2			2n+2*2			
EXTS	EXTS.W Rd	1						
	EXTS.L ERd	1						
EXTU	EXTU.W Rd	1						
	EXTU.L ERd	1						
INC	INC.B Rd	1						
	INC.W #1/2,Rd	1						
	INC.L #1/2,ERd	1						
JMP	JMP @ERn	2						
	JMP @aa:24	2					1	
	JMP @@aa:8 Advanced	2	2				1	
JSR	JSR @ERn Advanced	2		2				
	JSR @aa:24 Advanced	2		2			1	
	JSR @@aa:8 Advanced	2	2	2				
LDC	LDC #xx:8,CCR	1						
	LDC #xx:8,EXR	2						
	LDC Rs,CCR	1						
	LDC Rs,EXR	1						
	LDC @ERs,CCR	2					1	
	LDC @ERs,EXR	2					1	
	LDC @(d:16,ERs),CCR	3					1	
	LDC @(d:16,ERs),EXR	3					1	
	LDC @(d:32,ERs),CCR	5					1	
	LDC @(d:32,ERs),EXR	5					1	
	LDC @ERs+,CCR	2					1	1
	LDC @ERs+,EXR	2					1	1
	LDC @aa:16,CCR	3					1	
	LDC @aa:16,EXR	3					1	
	LDC @aa:32,CCR	4					1	
	LDC @aa:32,EXR	4					1	

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address Read	Operation	Data Access	Data Access	Operation
		I	J	K	L	M	N
LDM	LDM.L @SP+, (ERn-ERn+1)	2		4			1
	LDM.L @SP+, (ERn-ERn+2)	2		6			1
	LDM.L @SP+, (ERn-ERn+3)	2		8			1
LDMAC	LDMAC ERs,MACH LDMAC ERs,MACL	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series					
MAC	MAC @ERn+,@ERm+	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series					
MOV	MOV.B #xx:8,Rd	1					
	MOV.B Rs,Rd	1					
	MOV.B @ERs,Rd	1			1		
	MOV.B @(d:16,ERs),Rd	2			1		
	MOV.B @(d:32,ERs),Rd	4			1		
	MOV.B @ERs+,Rd	1			1		1
	MOV.B @aa:8,Rd	1			1		
	MOV.B @aa:16,Rd	2			1		
	MOV.B @aa:32,Rd	3			1		
	MOV.B Rs,@ERd	1			1		
	MOV.B Rs,@(d:16,ERd)	2			1		
	MOV.B Rs,@(d:32,ERd)	4			1		
	MOV.B Rs,@-ERd	1			1		1
	MOV.B Rs,@aa:8	1			1		
	MOV.B Rs,@aa:16	2			1		
	MOV.B Rs,@aa:32	3			1		
	MOV.W #xx:16,Rd	2					
	MOV.W Rs,Rd	1					
	MOV.W @ERs,Rd	1					1
	MOV.W @(d:16,ERs),Rd	2					1
	MOV.W @(d:32,ERs),Rd	4					1
MOV.W @ERs+,Rd	1					1	1
MOV.W @aa:16,Rd	2					1	
MOV.W @aa:32,Rd	3					1	
MOV.W Rs,@ERd	1					1	

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal	
		Fetch	Address	Operation	Data	Data		
		I	J	K	L	M	N	
MOV	MOV.W Rs,@(d:16,ERd)	2				1		
	MOV.W Rs,@(d:32,ERd)	4				1		
	MOV.W Rs,@-ERd	1				1	1	
	MOV.W Rs,@aa:16	2				1		
	MOV.W Rs,@aa:32	3				1		
	MOV.L #xx:32,ERd	3						
	MOV.L ERs,ERd	1						
	MOV.L @ERs,ERd	2					2	
	MOV.L @(d:16,ERs),ERd	3					2	
	MOV.L @(d:32,ERs),ERd	5					2	
	MOV.L @ERs+,ERd	2					2	1
	MOV.L @aa:16,ERd	3					2	
	MOV.L @aa:32,ERd	4					2	
	MOV.L ERs,@ERd	2					2	
	MOV.L ERs,@(d:16,ERd)	3					2	
	MOV.L ERs,@(d:32,ERd)	5					2	
	MOV.L ERs,@-ERd	2					2	1
MOV.L ERs,@aa:16	3					2		
MOV.L ERs,@aa:32	4					2		
MOVFP	MOVFP @:aa:16,Rd	Can not be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series						
MOVTPE	MOVTPE Rs,@:aa:16							
MULXS	MULXS.B Rs,Rd	2					11	
	MULXS.W Rs,ERd	2					19	
MULXU	MULXU.B Rs,Rd	1					11	
	MULXU.W Rs,ERd	1					19	
NEG	NEG.B Rd	1						
	NEG.W Rd	1						
	NEG.L ERd	1						
NOP	NOP	1						
NOT	NOT.B Rd	1						
	NOT.W Rd	1						
	NOT.L ERd	1						

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address Read	Operation	Data Access	Data Access	Operation
		I	J	K	L	M	N
OR	OR.B #xx:8,Rd	1					
	OR.B Rs,Rd	1					
	OR.W #xx:16,Rd	2					
	OR.W Rs,Rd	1					
	OR.L #xx:32,ERd	3					
	OR.L ERs,ERd	2					
ORC	ORC #xx:8,CCR	1					
	ORC #xx:8,EXR	2					
POP	POP.W Rn	1				1	1
	POP.L ERn	2				2	1
PUSH	PUSH.W Rn	1				1	1
	PUSH.L ERn	2				2	1
ROTL	ROTL.B Rd	1					
	ROTL.B #2,Rd	1					
	ROTL.W Rd	1					
	ROTL.W #2,Rd	1					
	ROTL.L ERd	1					
	ROTL.L #2,ERd	1					
ROTR	ROTR.B Rd	1					
	ROTR.B #2,Rd	1					
	ROTR.W Rd	1					
	ROTR.W #2,Rd	1					
	ROTR.L ERd	1					
	ROTR.L #2,ERd	1					
ROTXL	ROTXL.B Rd	1					
	ROTXL.B #2,Rd	1					
	ROTXL.W Rd	1					
	ROTXL.W #2,Rd	1					
	ROTXL.L ERd	1					
	ROTXL.L #2,ERd	1					



**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address	Operation	Data	Data	Operation
		I	J	K	L	M	N
ROTXR	ROTXR.B Rd	1					
	ROTXR.B #2,Rd	1					
	ROTXR.W Rd	1					
	ROTXR.W #2,Rd	1					
	ROTXR.L ERd	1					
	ROTXR.L #2,ERd	1					
RTE	RTE	2		2/3*1			1
RTS	RTS      Advanced	2		2			1
SHAL	SHAL.B Rd	1					
	SHAL.B #2,Rd	1					
	SHAL.W Rd	1					
	SHAL.W #2,Rd	1					
	SHAL.L ERd	1					
	SHAL.L #2,ERd	1					
SHAR	SHAR.B Rd	1					
	SHAR.B #2,Rd	1					
	SHAR.W Rd	1					
	SHAR.W #2,Rd	1					
	SHAR.L ERd	1					
	SHAR.L #2,ERd	1					
SHLL	SHLL.B Rd	1					
	SHLL.B #2,Rd	1					
	SHLL.W Rd	1					
	SHLL.W #2,Rd	1					
	SHLL.L ERd	1					
	SHLL.L #2,ERd	1					
SHLR	SHLR.B Rd	1					
	SHLR.B #2,Rd	1					
	SHLR.W Rd	1					
	SHLR.W #2,Rd	1					
	SHLR.L ERd	1					
	SHLR.L #2,ERd	1					
SLEEP	SLEEP	1					1

**Table A-5 Number of Cycles in Instruction Execution (cont)**

Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address	Operation	Data	Data	
		I	J	K	L	M	N
STC	STC.B CCR,Rd	1					
	STC.B EXR,Rd	1					
	STC.W CCR,@ERd	2				1	
	STC.W EXR,@ERd	2				1	
	STC.W CCR,@(d:16,ERd)	3				1	
	STC.W EXR,@(d:16,ERd)	3				1	
	STC.W CCR,@(d:32,ERd)	5				1	
	STC.W EXR,@(d:32,ERd)	5				1	
	STC.W CCR,@-ERd	2				1	1
	STC.W EXR,@-ERd	2				1	1
	STC.W CCR,@aa:16	3				1	
	STC.W EXR,@aa:16	3				1	
	STC.W CCR,@aa:32	4				1	
	STC.W EXR,@aa:32	4				1	
STM	STM.L (ERn-ERn+1), @-SP	2		4			1
	STM.L (ERn-ERn+2), @-SP	2		6			1
	STM.L (ERn-ERn+3), @-SP	2		8			1
STMAC	STMAC MACH,ERd STMAC MACL,ERd	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series					
SUB	SUB.B Rs,Rd	1					
	SUB.W #xx:16,Rd	2					
	SUB.W Rs,Rd	1					
	SUB.L #xx:32,ERd	3					
	SUB.L ERs,ERd	1					
SUBS	SUBS #1/2/4,ERd	1					
SUBX	SUBX #xx:8,Rd	1					
	SUBX Rs,Rd	1					
TAS	TAS @ERd	2			2		
TRAPA	TRAPA #x:2 Advanced	2	2	2/3*1			2

**Table A-5 Number of Cycles in Instruction Execution (cont)**

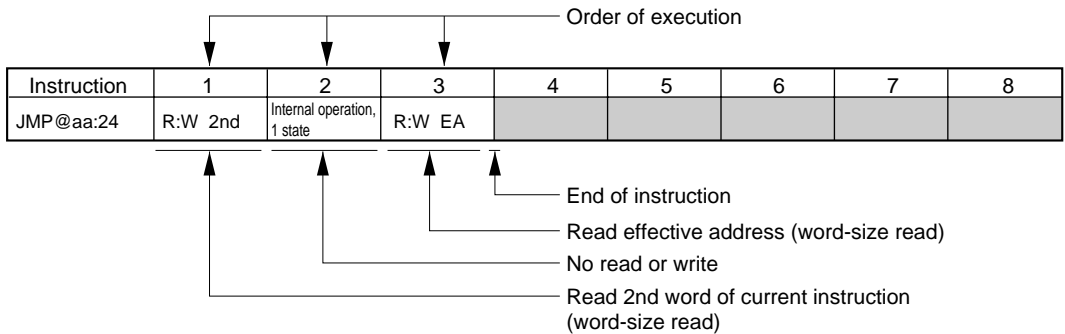
Instruction	Mnemonic	Instruction	Branch	Stack	Byte	Word	Internal
		Fetch	Address	Operation	Data	Data	Operation
		I	J	K	L	M	N
XOR	XOR.B #xx:8,Rd	1					
	XOR.B Rs,Rd	1					
	XOR.W #xx:16,Rd	2					
	XOR.W Rs,Rd	1					
	XOR.L #xx:32,ERd	3					
	XOR.L ERs,ERd	2					
XORC	XORC #xx:8,CCR	1					
	XORC #xx:8,EXR	2					

Notes: 1. 2 when EXR is invalid, 3 when EXR is valid.  
2. When n bytes of data are transferred.

## A.5 Bus States During Instruction Execution

Table A-6 indicates the types of cycles that occur during instruction execution by the CPU. See table A-4 for the number of states per cycle.

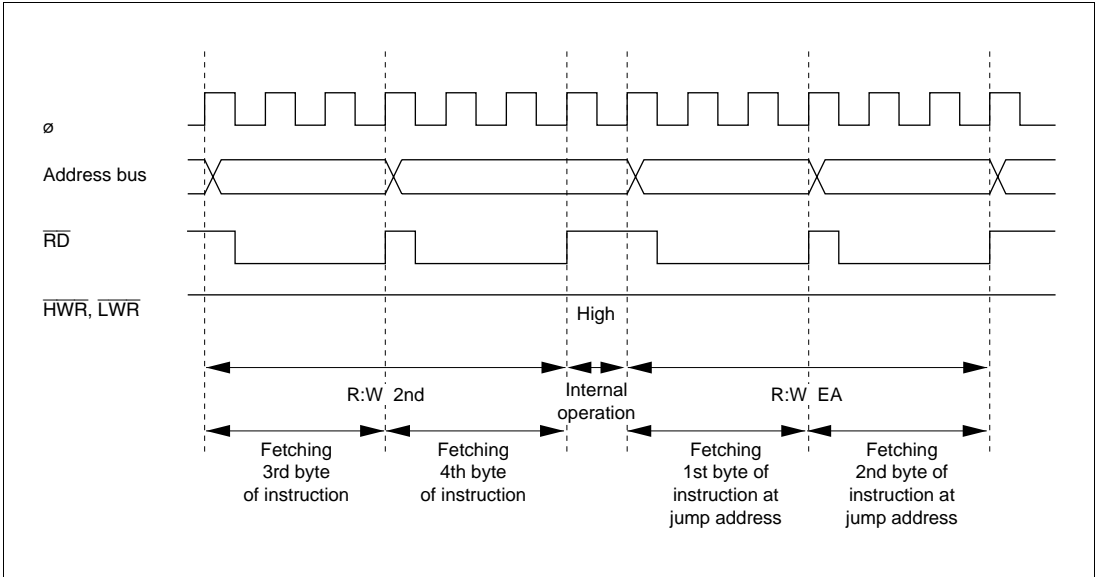
### How to Read the Table:



### Legend

R:B	Byte-size read
R:W	Word-size read
W:B	Byte-size write
W:W	Word-size write
:M	Transfer of the bus is not performed immediately after this cycle
2nd	Address of 2nd word (3rd and 4th bytes)
3rd	Address of 3rd word (5th and 6th bytes)
4th	Address of 4th word (7th and 8th bytes)
5th	Address of 5th word (9th and 10th bytes)
NEXT	Address of next instruction
EA	Effective address
VEC	Vector address

Figure A-1 shows timing waveforms for the address bus and the  $\overline{RD}$ ,  $\overline{HWR}$ , and  $\overline{LWR}$  signals during execution of the above instruction with an 8-bit bus, using three-state access with no wait states.



**Figure A-1 Address Bus,  $\overline{RD}$ ,  $\overline{HWR}$ , and  $\overline{LWR}$  Timing  
(8-Bit Bus, Three-State Access, No Wait States)**

**Table A-6 Instruction Execution Cycles**

Instruction	1	2	3	4	5	6	7	8	9
ADD.B #xx:8,Rd	R:W NEXT								
ADD.B Rs,Rd	R:W NEXT								
ADD.W #xx:16,Rd	R:W 2nd	R:W NEXT							
ADD.W Rs,Rd	R:W NEXT								
ADD.L #xx:32,ERd	R:W 2nd	R:W 3rd	R:W NEXT						
ADD.L ERs,ERd	R:W NEXT								
ADDS #1/2/4,ERd	R:W NEXT								
ADDX #xx:8,Rd	R:W NEXT								
ADDX Rs,Rd	R:W NEXT								
AND.B #xx:8,Rd	R:W NEXT								
AND.B Rs,Rd	R:W NEXT								
AND.W #xx:16,Rd	R:W 2nd	R:W NEXT							
AND.W Rs,Rd	R:W NEXT								
AND.L #xx:32,ERd	R:W 2nd	R:W 3rd	R:W NEXT						
AND.L ERs,ERd	R:W 2nd	R:W NEXT							
ANDC #xx:8,CCR	R:W NEXT								
ANDC #xx:8,EXR	R:W 2nd	R:W NEXT							
BAND #xx:3,Rd	R:W NEXT								
BAND #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BAND #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BAND #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BAND #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BRA d:8 (BT d:8)	R:W NEXT	R:W EA							
BRN d:8 (BF d:8)	R:W NEXT	R:W EA							
BHI d:8	R:W NEXT	R:W EA							
BLS d:8	R:W NEXT	R:W EA							
BCC d:8 (BHS d:8)	R:W NEXT	R:W EA							
BCS d:8 (BLO d:8)	R:W NEXT	R:W EA							
BNE d:8	R:W NEXT	R:W EA							
BEQ d:8	R:W NEXT	R:W EA							
BVC d:8	R:W NEXT	R:W EA							
BVS d:8	R:W NEXT	R:W EA							
BPL d:8	R:W NEXT	R:W EA							
BMI d:8	R:W NEXT	R:W EA							
BGE d:8	R:W NEXT	R:W EA							
BLT d:8	R:W NEXT	R:W EA							
BGT d:8	R:W NEXT	R:W EA							

**Table A-6 Instruction Execution Cycles (cont)**

Instruction	1	2	3	4	5	6	7	8	9
BLE d:8	R:W NEXT	R:W EA							
BRA d:16 (BT d:16)	R:W 2nd	Internal operation, 1 state	R:W EA						
BRN d:16 (BF d:16)	R:W 2nd	Internal operation, 1 state	R:W EA						
BHI d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BLS d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BCC d:16 (BHS d:16)	R:W 2nd	Internal operation, 1 state	R:W EA						
BCS d:16 (BLO d:16)	R:W 2nd	Internal operation, 1 state	R:W EA						
BNE d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BEQ d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BVC d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BVS d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BPL d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BMI d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BGE d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BLT d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BGT d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BLE d:16	R:W 2nd	Internal operation, 1 state	R:W EA						
BCLR #xx:3,Rd	R:W NEXT								
BCLR #xx:3,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BCLR #xx:3,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BCLR #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				

Table A-6 Instruction Execution Cycles (cont)

Instruction	1	2	3	4	5	6	7	8	9
BCLR #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BCLR Rn,Rd	R:W NEXT								
BCLR Rn,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BCLR Rn,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BCLR Rn,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BCLR Rn,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BIAND #xx:3,Rd	R:W NEXT								
BIAND #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BIAND #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BIAND #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BIAND #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BILD #xx:3,Rd	R:W NEXT								
BILD #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BILD #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BILD #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BILD #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BIOR #xx:3,Rd	R:W NEXT								
BIOR #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BIOR #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BIOR #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BIOR #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BIST #xx:3,Rd	R:W NEXT								
BIST #xx:3,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BIST #xx:3,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BIST #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BIST #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BIXOR #xx:3,Rd	R:W NEXT								
BIXOR #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BIXOR #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BIXOR #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BIXOR #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BLD #xx:3,Rd	R:W NEXT								
BLD #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BLD #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BLD #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BLD #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BNOT #xx:3,Rd	R:W NEXT								



**Table A-6 Instruction Execution Cycles (cont)**

Instruction	1	2	3	4	5	6	7	8	9
BNOT #xx:3,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BNOT #xx:3,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BNOT #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BNOT #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BNOT Rn,Rd	R:W NEXT								
BNOT Rn,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BNOT Rn,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BNOT Rn,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BNOT Rn,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BOR #xx:3,Rd	R:W NEXT								
BOR #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BOR #xx:3,@aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BOR #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BOR #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BSET #xx:3,Rd	R:W NEXT								
BSET #xx:3,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BSET #xx:3,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BSET #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BSET #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BSET Rn,Rd	R:W NEXT								
BSET Rn,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BSET Rn,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BSET Rn,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BSET Rn,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BSR d:8	Advanced	R:W NEXT	R:W EA	W:W:M stack (H)	W:W stack (L)				
BSR d:16	Advanced	R:W 2nd	Internal operation, 1 state	R:W EA	W:W:M stack (H)	W:W stack (L)			
BST #xx:3,Rd	R:W NEXT								
BST #xx:3,@ERd	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BST #xx:3,@aa:8	R:W 2nd	R:B:M EA	R:W:M NEXT	W:B EA					
BST #xx:3,@aa:16	R:W 2nd	R:W 3rd	R:B:M EA	R:W:M NEXT	W:B EA				
BST #xx:3,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B:M EA	R:W:M NEXT	W:B EA			
BTST #xx:3,Rd	R:W NEXT								
BTST #xx:3,@ERd	R:W 2nd	R:B EA	R:W:M NEXT						

Table A-6 Instruction Execution Cycles (cont)

Instruction	1	2	3	4	5	6	7	8	9
BTST #xx:3, @aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BTST #xx:3, @aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BTST #xx:3, @aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BTST Rn, Rd	R:W NEXT								
BTST Rn, @ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BTST Rn, @aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BTST Rn, @aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BTST Rn, @aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
BXOR #xx:3, Rd	R:W NEXT								
BXOR #xx:3, @ERd	R:W 2nd	R:B EA	R:W:M NEXT						
BXOR #xx:3, @aa:8	R:W 2nd	R:B EA	R:W:M NEXT						
BXOR #xx:3, @aa:16	R:W 2nd	R:W 3rd	R:B EA	R:W:M NEXT					
BXOR #xx:3, @aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:B EA	R:W:M NEXT				
CLRMAC	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series								
CMP.B #xx:8, Rd	R:W NEXT								
CMP.B Rs, Rd	R:W NEXT								
CMP.W #xx:16, Rd	R:W 2nd	R:W NEXT							
CMP.W Rs, Rd	R:W NEXT								
CMP.L #xx:32, ERd	R:W 2nd	R:W 3rd	R:W NEXT						
CMP.L ERs, ERd	R:W NEXT								
DAA Rd	R:W NEXT								
DAS Rd	R:W NEXT								
DEC.B Rd	R:W NEXT								
DEC.W #1/2, Rd	R:W NEXT								
DEC.L #1/2, ERd	R:W NEXT								
DIVXS.B Rs, Rd	R:W 2nd	R:W NEXT	Internal operation, 11 states						
DIVXS.W Rs, ERd	R:W 2nd	R:W NEXT	Internal operation, 19 states						
DIVXU.B Rs, Rd	R:W NEXT	Internal operation, 11 states							
DIVXU.W Rs, ERd	R:W NEXT	Internal operation, 19 states							
EEMOV.B	R:W 2nd	R:B EAs*1	R:B EAd*1	R:B EAs*2	W:B EAd*2	R:W NEXT			
EEMOV.W	R:W 2nd	R:B EAs*1	R:B EAd*1	R:B EAs*2	W:B EAd*2	R:W NEXT			
EXTS.W Rd	R:W NEXT			← Repeated n times*2 →					
EXTS.L ERd	R:W NEXT								
EXTU.W Rd	R:W NEXT								
EXTU.L ERd	R:W NEXT								
INC.B Rd	R:W NEXT								

**Table A-6 Instruction Execution Cycles (cont)**

Instruction		1	2	3	4	5	6	7	8	9
INC.W #1/2,Rd		R:W NEXT								
INC.L #1/2,ERd		R:W NEXT								
JMP @ERn		R:W NEXT	R:W EA							
JMP @aa:24		R:W 2nd	Internal operation, 1 state	R:W EA						
JMP @@aa:8	Advanced	R:W NEXT	R:W:M aa:8	R:W aa:8	Internal operation, 1 state	R:W EA				
JSR @ERn	Advanced	R:W NEXT	R:W EA	W:W:M stack (H)	W:W stack (L)					
JSR @aa:24	Advanced	R:W 2nd	Internal operation, 1 state	R:W EA	W:W:M stack (H)	W:W stack (L)				
JSR @@aa:8	Advanced	R:W NEXT	R:W:M aa:8	R:W aa:8	W:W:M stack (H)	W:W stack (L)	R:W EA			
LDC #xx:8,CCR		R:W NEXT								
LDC #xx:8,EXR		R:W 2nd	R:W NEXT							
LDC Rs,CCR		R:W NEXT								
LDC Rs,EXR		R:W NEXT								
LDC @ERs,CCR		R:W 2nd	R:W NEXT	R:W EA						
LDC @ERs,EXR		R:W 2nd	R:W NEXT	R:W EA						
LDC @(d:16,ERs),CCR		R:W 2nd	R:W 3rd	R:W NEXT	R:W EA					
LDC @(d:16,ERs),EXR		R:W 2nd	R:W 3rd	R:W NEXT	R:W EA					
LDC @(d:32,ERs),CCR		R:W 2nd	R:W 3rd	R:W 4th	R:W 5th	R:W NEXT	R:W EA			
LDC @(d:32,ERs),EXR		R:W 2nd	R:W 3rd	R:W 4th	R:W 5th	R:W NEXT	R:W EA			
LDC @ERs+,CCR		R:W 2nd	R:W NEXT	Internal operation, 1 state	R:W EA					
LDC @ERs+,EXR		R:W 2nd	R:W NEXT	Internal operation, 1 state	R:W EA					
LDC @aa:16,CCR		R:W 2nd	R:W 3rd	R:W NEXT	R:W EA					
LDC @aa:16,EXR		R:W 2nd	R:W 3rd	R:W NEXT	R:W EA					
LDC @aa:32,CCR		R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	R:W EA				
LDC @aa:32,EXR		R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	R:W EA				
LDM.L @SP+, (ERn-ERn+1)		R:W 2nd	R:W:M NEXT	Internal operation, 1 state	R:W:M stack (H)*3	R:W stack (L)*3				

**Table A-6 Instruction Execution Cycles (cont)**

Instruction	1	2	3	4	5	6	7	8	9
LDM.L @SP+,(ERn-ERn+2)	R:W 2nd	R:W NEXT	Internal operation, 1 state	R:W:M stack (H)*3	R:W stack (L)*3				
LDM.L @SP+,(ERn-ERn+3)	R:W 2nd	R:W NEXT	Internal operation, 1 state	R:W:M stack (H)*3	R:W stack (L)*3				
LDMAC ERs,MACH	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series								
LDMAC ERs,MACL									
MAC @ERn+,@ERm+									
MOV.B #xx:8,Rd	R:W NEXT								
MOV.B Rs,Rd	R:W NEXT								
MOV.B @ERs,Rd	R:W NEXT	R:B EA							
MOV.B @(d:16,ERs),Rd	R:W 2nd	R:W NEXT	R:B EA						
MOV.B @(d:32,ERs),Rd	R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	R:B EA				
MOV.B @ERs+,Rd	R:W NEXT	Internal operation, 1 state	R:B EA						
MOV.B @aa:8,Rd	R:W NEXT	R:B EA							
MOV.B @aa:16,Rd	R:W 2nd	R:W NEXT	R:B EA						
MOV.B @aa:32,Rd	R:W 2nd	R:W 3rd	R:W NEXT	R:B EA					
MOV.B Rs,@ERd	R:W NEXT	W:B EA							
MOV.B Rs,@(d:16,ERd)	R:W 2nd	R:W NEXT	W:B EA						
MOV.B Rs,@(d:32,ERd)	R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	W:B EA				
MOV.B Rs,@-ERd	R:W NEXT	Internal operation, 1 state	W:B EA						
MOV.B Rs,@aa:8	R:W NEXT	W:B EA							
MOV.B Rs,@aa:16	R:W 2nd	R:W NEXT	W:B EA						
MOV.B Rs,@aa:32	R:W 2nd	R:W 3rd	R:W NEXT	W:B EA					
MOV.W #xx:16,Rd	R:W 2nd	R:W NEXT							
MOV.W Rs,Rd	R:W NEXT								
MOV.W @ERs,Rd	R:W NEXT	R:W EA							
MOV.W @(d:16,ERs),Rd	R:W 2nd	R:W NEXT	R:W EA						
MOV.W @(d:32,ERs),Rd	R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	R:W EA				
MOV.W @ERs+,Rd	R:W NEXT	Internal operation, 1 state	R:W EA						
MOV.W @aa:16,Rd	R:W 2nd	R:W NEXT	R:W EA						
MOV.W @aa:32,Rd	R:W 2nd	R:W 3rd	R:W NEXT	R:B EA					
MOV.W Rs,@ERd	R:W NEXT	W:W EA							

**Table A-6 Instruction Execution Cycles (cont)**

Instruction	1	2	3	4	5	6	7	8	9
MOV.W Rs,@(d:16,ERd)	R:W 2nd	R:W NEXT	W:W EA						
MOV.W Rs,@(d:32,ERd)	R:W 2nd	R:W 3rd	R:E 4th	R:W NEXT	W:W EA				
MOV.W Rs,@-ERd	R:W NEXT	Internal operation, 1 state	W:W EA						
MOV.W Rs,@aa:16	R:W 2nd	R:W NEXT	W:W EA						
MOV.W Rs,@aa:32	R:W 2nd	R:W 3rd	R:W NEXT	W:W EA					
MOV.L #xx:32,ERd	R:W 2nd	R:W 3rd	R:W NEXT						
MOV.L ERs,ERd	R:W NEXT								
MOV.L @ERs,ERd	R:W 2nd	R:W:M NEXT	R:W:M EA	R:W EA+2					
MOV.L @(d:16,ERs),ERd	R:W 2nd	R:W:M 3rd	R:W NEXT	R:W:M EA	R:W EA+2				
MOV.L @(d:32,ERs),ERd	R:W 2nd	R:W:M 3rd	R:W:M 4th	R:W 5th	R:W NEXT	R:W:M EA	R:W EA+2		
MOV.L @ERs+,ERd	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	R:W:M EA	R:W EA+2				
MOV.L @aa:16,ERd	R:W 2nd	R:W:M 3rd	R:W NEXT	R:W:M EA	R:W EA+2				
MOV.L @aa:32,ERd	R:W 2nd	R:W:M 3rd	R:W 4th	R:W NEXT	R:W:M EA	R:W EA+2			
MOV.L ERs,@ERd	R:W 2nd	R:W:M NEXT	W:W:M EA	W:W EA+2					
MOV.L ERs,@(d:16,ERd)	R:W 2nd	R:W:M 3rd	R:W NEXT	W:W:M EA	W:W EA+2				
MOV.L ERs,@(d:32,ERd)	R:W 2nd	R:W:M 3rd	R:W:M 4th	R:W 5th	R:W NEXT	W:W:M EA	W:W EA+2		
MOV.L ERs,@-ERd	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	W:W:M EA	W:W EA+2				
MOV.L ERs,@aa:16	R:W 2nd	R:W:M 3rd	R:W NEXT	W:W:M EA	W:W EA+2				
MOV.L ERs,@aa:32	R:W 2nd	R:W:M 3rd	R:W 4th	R:W NEXT	W:W:M EA	W:W EA+2			
MOVFPPE @aa:16,Rd	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series								
MOVTPPE Rs,@aa:16									
MULXS.B Rs,Rd	R:W 2nd	R:W NEXT	Internal operation, 11 states						
MULXS.W Rs,ERd	R:W 2nd	R:W NEXT	Internal operation, 19 states						
MULXU.B Rs,Rd	R:W NEXT	Internal operation, 11 states							
MULXU.W Rs,ERd	R:W NEXT	Internal operation, 19 states							
NEG.B Rd	R:W NEXT								
NEG.W Rd	R:W NEXT								
NEG.L ERd	R:W NEXT								
NOP	R:W NEXT								
NOT.B Rd	R:W NEXT								
NOT.W Rd	R:W NEXT								
NOT.L ERd	R:W NEXT								
OR.B #xx:8,Rd	R:W NEXT								
OR.B Rs,Rd	R:W NEXT								

**Table A-6 Instruction Execution Cycles (cont)**

Instruction	1	2	3	4	5	6	7	8	9
OR.W #xx:16,Rd	R:W 2nd	R:W NEXT							
OR.W Rs,Rd	R:W NEXT								
OR.L #xx:32,ERd	R:W 2nd	R:W 3rd	R:W NEXT						
OR.L ERs,ERd	R:W 2nd	R:W NEXT							
ORC #xx:8,CCR	R:W NEXT								
ORC #xx:8,EXR	R:W 2nd	R:W NEXT							
POP.W Rn	R:W NEXT	Internal operation, 1 state	R:W EA						
POP.L ERn	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	R:W:M EA	R:W EA+2				
PUSH.W Rn	R:W NEXT	Internal operation, 1 state	W:W EA						
PUSH.L ERn	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	W:W:M EA	W:W EA+2				
ROTL.B Rd	R:W NEXT								
ROTL.B #2,Rd	R:W NEXT								
ROTL.W Rd	R:W NEXT								
ROTL.W #2,Rd	R:W NEXT								
ROTL.L ERd	R:W NEXT								
ROTL.L #2,ERd	R:W NEXT								
ROTR.B Rd	R:W NEXT								
ROTR.B #2,Rd	R:W NEXT								
ROTR.W Rd	R:W NEXT								
ROTR.W #2,Rd	R:W NEXT								
ROTR.L ERd	R:W NEXT								
ROTR.L #2,ERd	R:W NEXT								
ROTXL.B Rd	R:W NEXT								
ROTXL.B #2,Rd	R:W NEXT								
ROTXL.W Rd	R:W NEXT								
ROTXL.W #2,Rd	R:W NEXT								
ROTXL.L ERd	R:W NEXT								
ROTXL.L #2,ERd	R:W NEXT								
ROTXR.B Rd	R:W NEXT								
ROTXR.B #2,Rd	R:W NEXT								
ROTXR.W Rd	R:W NEXT								
ROTXR.W #2,Rd	R:W NEXT								
ROTXR.L ERd	R:W NEXT								

**Table A-6 Instruction Execution Cycles (cont)**

Instruction		1	2	3	4	5	6	7	8	9
ROTXR.L #2,ERd		R:W NEXT								
RTE		R:W NEXT	R:W stack (EXR)	R:W stack (H)	R:W stack (L)	Internal operation, 1 state	R:W*4			
RTS	Advanced	R:W NEXT	R:W:M stack (H)	R:W stack (L)	Internal operation, 1 state	R:W*4				
SHAL.B Rd		R:W NEXT								
SHAL.B #2,Rd		R:W NEXT								
SHAL.W Rd		R:W NEXT								
SHAL.W #2,Rd		R:W NEXT								
SHAL.L ERd		R:W NEXT								
SHAL.L #2,ERd		R:W NEXT								
SHAR.B Rd		R:W NEXT								
SHAR.B #2,Rd		R:W NEXT								
SHAR.W Rd		R:W NEXT								
SHAR.W #2,Rd		R:W NEXT								
SHAR.L ERd		R:W NEXT								
SHAR.L #2,ERd		R:W NEXT								
SHLL.B Rd		R:W NEXT								
SHLL.B #2,Rd		R:W NEXT								
SHLL.W Rd		R:W NEXT								
SHLL.W #2,Rd		R:W NEXT								
SHLL.L ERd		R:W NEXT								
SHLL.L #2,ERd		R:W NEXT								
SHLR.B Rd		R:W NEXT								
SHLR.B #2,Rd		R:W NEXT								
SHLR.W Rd		R:W NEXT								
SHLR.W #2,Rd		R:W NEXT								
SHLR.L ERd		R:W NEXT								
SHLR.L #2,ERd		R:W NEXT								
SLEEP		R:W NEXT	Internal operation:M							
STC CCR,Rd		R:W NEXT								
STC EXR,Rd		R:W NEXT								
STC CCR,@ERd		R:W 2nd	R:W NEXT	W:W EA						
STC EXR,@ERd		R:W 2nd	R:W NEXT	W:W EA						
STC CCR,@(d:16,ERd)		R:W 2nd	R:W 3rd	R:W NEXT	W:W EA					

Table A-6 Instruction Execution Cycles (cont)

Instruction	1	2	3	4	5	6	7	8	9	
STC EXR,@(d:16,ERd)	R:W 2nd	R:W 3rd	R:W NEXT	W:W EA						
STC CCR,@(d:32,ERd)	R:W 2nd	R:W 3rd	R:W 4th	R:W 5th	R:W NEXT	W:W EA				
STC EXR,@(d:32,ERd)	R:W 2nd	R:W 3rd	R:W 4th	R:W 5th	R:W NEXT	W:W EA				
STC CCR,@-ERd	R:W 2nd	R:W NEXT	Internal operation, 1 state	W:W EA						
STC EXR,@-ERd	R:W 2nd	R:W NEXT	Internal operation, 1 state	W:W EA						
STC CCR,@aa:16	R:W 2nd	R:W 3rd	R:W NEXT	W:W EA						
STC EXR,@aa:16	R:W 2nd	R:W 3rd	R:W NEXT	W:W EA						
STC CCR,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	W:W EA					
STC EXR,@aa:32	R:W 2nd	R:W 3rd	R:W 4th	R:W NEXT	W:W EA					
STM.L(ERn-ERn+1),@-SP	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	W:W:M stack (H) <sup>*3</sup>	W:W stack (L) <sup>*3</sup>					
STM.L(ERn-ERn+2),@-SP	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	W:W:M stack (H) <sup>*3</sup>	W:W stack (L) <sup>*3</sup>					
STM.L(ERn-ERn+3),@-SP	R:W 2nd	R:W:M NEXT	Internal operation, 1 state	W:W:M stack (H) <sup>*3</sup>	W:W stack (L) <sup>*3</sup>					
STMACH,ERd	Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series									
STMACH,ERd										
SUB.B Rs,Rd	R:W NEXT									
SUB.W #xx:16,Rd	R:W 2nd	R:W NEXT								
SUB.W Rs,Rd	R:W NEXT									
SUB.L #xx:32,ERd	R:W 2nd	R:W 3rd	R:W NEXT							
SUB.L ERs,ERd	R:W NEXT									
SUBS #1/2/4,ERd	R:W NEXT									
SUBX #xx:8,Rd	R:W NEXT									
SUBX Rs,Rd	R:W NEXT									
TAS @ERd	R:W 2nd	R:W NEXT	R:B:M EA	W:B EA						
TRAPA #x:2	Advanced	R:W NEXT	Internal operation, 1 state	W:W stack (L)	W:W stack (H)	W:W stack (EXR)	R:W:M VEC	R:W VEC+2	Internal operation, 1 state	R:W <sup>*7</sup>
XOR.B #xx8,Rd	R:W NEXT									
XOR.B Rs,Rd	R:W NEXT									
XOR.W #xx:16,Rd	R:W 2nd	R:W NEXT								
XOR.W Rs,Rd	R:W NEXT									
XOR.L #xx:32,ERd	R:W 2nd	R:W 3rd	R:W NEXT							



**Table A-6 Instruction Execution Cycles (cont)**

Instruction		1	2	3	4	5	6	7	8	9
XOR.L ERs,ERd		R:W 2nd	R:W NEXT							
XORC #xx:8,CCR		R:W NEXT								
XORC #xx:8,EXR		R:W 2nd	R:W NEXT							
Reset exception handling	Advanced	R:W VEC	R:W VEC+2	Internal operation, 1 state	R:W*5					
Interrupt exception handling	Advanced	R:W*6	Internal operation, 1 state	W:W stack (L)	W:W stack (H)	W:W stack (EXR)	R:W:M VEC	R:W VEC+2	Internal operation, 1 state	R:W*7

- Notes:
1. EAs is the contents of ER5. EAd is the contents of ER6.
  2. EAs is the contents of ER5. EAd is the contents of ER6. Both registers are incremented by 1 after execution of the instruction. n is the initial value of R4L or R4. If n = 0, these bus cycles are not executed.
  3. Repeated two times to save or restore two registers, three times for three registers, or four times for four registers.
  4. Start address after return.
  5. Start address of the program.
  6. Prefetch address, equal to two plus the PC value pushed onto the stack. In recovery from sleep mode or software standby mode the read operation is replaced by an internal operation.
  7. Start address of the interrupt handling routine.

## A.6 Condition Code Modification

This section indicates the effect of each CPU instruction on the condition code. The notation used in the table is defined below.

$$m = \begin{cases} 31 & \text{for longword operands} \\ 15 & \text{for word operands} \\ 7 & \text{for byte operands} \end{cases}$$

$S_i$	The $i$ -th bit of the source operand
$D_i$	The $i$ -th bit of the destination operand
$R_i$	The $i$ -th bit of the result
$D_n$	The specified bit in the destination operand
—	Not affected
$\updownarrow$	Modified according to the result of the instruction (see definition)
0	Always cleared to 0
1	Always set to 1
*	Undetermined (no guaranteed value)
Z'	Z flag before instruction execution
C'	C flag before instruction execution

**Table A-7 Condition Code Modification**

Instruction	H	N	Z	V	C	Definition
ADD	↓	↓	↓	↓	↓	$H = S_{m-4} \cdot D_{m-4} + D_{m-4} \cdot \overline{R_{m-4}} + S_{m-4} \cdot \overline{R_{m-4}}$ $N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ $V = S_m \cdot D_m \cdot \overline{R_m} + \overline{S_m} \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot D_m + D_m \cdot \overline{R_m} + S_m \cdot \overline{R_m}$
ADDS	—	—	—	—	—	
ADDX	↓	↓	↓	↓	↓	$H = S_{m-4} \cdot D_{m-4} + D_{m-4} \cdot \overline{R_{m-4}} + S_{m-4} \cdot \overline{R_{m-4}}$ $N = R_m$ $Z = Z' \cdot \overline{R_m} \cdot \dots \cdot \overline{R_0}$ $V = S_m \cdot D_m \cdot \overline{R_m} + \overline{S_m} \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot D_m + D_m \cdot \overline{R_m} + S_m \cdot \overline{R_m}$
AND	—	↓	↓	0	—	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$
ANDC	↓	↓	↓	↓	↓	Stores the corresponding bits of the result. No flags change when the operand is EXR.
BAND	—	—	—	—	↓	$C = C' \cdot D_n$
Bcc	—	—	—	—	—	
BCLR	—	—	—	—	—	
BIAND	—	—	—	—	↓	$C = C' \cdot \overline{D_n}$
BILD	—	—	—	—	↓	$C = \overline{D_n}$
BIOR	—	—	—	—	↓	$C = C' + \overline{D_n}$
BIST	—	—	—	—	—	
BIXOR	—	—	—	—	↓	$C = C' \cdot D_n + \overline{C'} \cdot \overline{D_n}$
BLD	—	—	—	—	↓	$C = D_n$
BNOT	—	—	—	—	—	
BOR	—	—	—	—	↓	$C = C' + D_n$
BSET	—	—	—	—	—	
BSR	—	—	—	—	—	
BST	—	—	—	—	—	
BTST	—	—	↓	—	—	$Z = \overline{D_n}$
BXOR	—	—	—	—	↓	$C = C' \cdot \overline{D_n} + \overline{C'} \cdot D_n$
CLRMAC						Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series

**Table A-7 Condition Code Modification (cont)**

Instruction	H	N	Z	V	C	Definition
CMP	↑	↑	↑	↑	↑	$H = S_{m-4} \cdot \overline{D_{m-4}} + \overline{D_{m-4}} \cdot R_{m-4} + S_{m-4} \cdot R_{m-4}$ $N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ $V = \overline{S_m} \cdot D_m \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot \overline{D_m} + \overline{D_m} \cdot R_m + S_m \cdot R_m$
DAA	*	↑	↑	*	↑	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ C: decimal arithmetic carry
DAS	*	↑	↑	*	↑	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ C: decimal arithmetic borrow
DEC	—	↑	↑	↑	—	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ $V = D_m \cdot \overline{R_m}$
DIVXS	—	↑	↑	—	—	$N = S_m \cdot \overline{D_m} + \overline{S_m} \cdot D_m$ $Z = \overline{S_m} \cdot \overline{S_{m-1}} \cdot \dots \cdot \overline{S_0}$
DIVXU	—	↑	↑	—	—	$N = S_m$ $Z = \overline{S_m} \cdot \overline{S_{m-1}} \cdot \dots \cdot \overline{S_0}$
EEPMOV	—	—	—	—	—	
EXTS	—	↑	↑	0	—	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$
EXTU	—	0	↑	0	—	$Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$
INC	—	↑	↑	↑	—	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ $V = \overline{D_m} \cdot R_m$
JMP	—	—	—	—	—	
JSR	—	—	—	—	—	
LDC	↑	↑	↑	↑	↑	Stores the corresponding bits of the result. No flags change when the operand is EXR.
LDM	—	—	—	—	—	
LDMAC						Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series
MAC						

**Table A-7 Condition Code Modification (cont)**

Instruction	H	N	Z	V	C	Definition
MOV	—	↓	↓	0	—	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$
MOVFPPE						Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series
MOVTPE						
MULXS	—	↓	↓	—	—	N = R2m Z = $R2m \cdot \overline{R2m-1} \cdot \dots \cdot \overline{R0}$
MULXU	—	—	—	—	—	
NEG	↓	↓	↓	↓	↓	H = Dm-4 + Rm-4 N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ V = Dm · Rm C = Dm + Rm
NOP	—	—	—	—	—	
NOT	—	↓	↓	0	—	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$
OR	—	↓	↓	0	—	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$
ORC	↓	↓	↓	↓	↓	Stores the corresponding bits of the result. No flags change when the operand is EXR.
POP	—	↓	↓	0	—	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$
PUSH	—	↓	↓	0	—	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$
ROTL	—	↓	↓	0	↓	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = Dm (1-bit shift) or C = Dm-1 (2-bit shift)
ROTR	—	↓	↓	0	↓	N = Rm Z = $\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = D0 (1-bit shift) or C = D1 (2-bit shift)

**Table A-7 Condition Code Modification (cont)**

Instruction	H	N	Z	V	C	Definition
ROTXL	—	↓	↓	0	↓	N = Rm $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = Dm (1-bit shift) or C = Dm-1 (2-bit shift)
ROTXR	—	↓	↓	0	↓	N = Rm $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = D0 (1-bit shift) or C = D1 (2-bit shift)
RTE	↓	↓	↓	↓	↓	Stores the corresponding bits of the result.
RTS	—	—	—	—	—	
SHAL	—	↓	↓	↓	↓	N = Rm $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $V = \overline{Dm} \cdot \overline{Dm-1} + \overline{Dm} \cdot \overline{Dm-1}$ (1-bit shift) $V = \overline{Dm} \cdot \overline{Dm-1} \cdot \overline{Dm-2} \cdot \overline{Dm} \cdot \overline{Dm-1} \cdot \overline{Dm-2}$ (2-bit shift) C = Dm (1-bit shift) or C = Dm-1 (2-bit shift)
SHAR	—	↓	↓	0	↓	N = Rm $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = D0 (1-bit shift) or C = D1 (2-bit shift)
SHLL	—	↓	↓	0	↓	N = Rm $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = Dm (1-bit shift) or C = Dm-1 (2-bit shift)
SHLR	—	0	↓	0	↓	N = Rm $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C = D0 (1-bit shift) or C = D1 (2-bit shift)
SLEEP	—	—	—	—	—	
STC	—	—	—	—	—	
STM	—	—	—	—	—	
STMAC						Cannot be used in the H8S/2338 Series, H8S/2328 Series, or H8S/2318 Series

**Table A-7 Condition Code Modification (cont)**

<b>Instruction</b>	<b>H</b>	<b>N</b>	<b>Z</b>	<b>V</b>	<b>C</b>	<b>Definition</b>
SUB	↑	↑	↑	↑	↑	$H = S_{m-4} \cdot \overline{D_{m-4}} + \overline{D_{m-4}} \cdot R_{m-4} + S_{m-4} \cdot R_{m-4}$ $N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$ $V = \overline{S_m} \cdot D_m \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot \overline{D_m} + \overline{D_m} \cdot R_m + S_m \cdot R_m$
SUBS	—	—	—	—	—	
SUBX	↑	↑	↑	↑	↑	$H = S_{m-4} \cdot \overline{D_{m-4}} + \overline{D_{m-4}} \cdot R_{m-4} + S_{m-4} \cdot R_{m-4}$ $N = R_m$ $Z = Z' \cdot \overline{R_m} \cdot \dots \cdot \overline{R_0}$ $V = \overline{S_m} \cdot D_m \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot \overline{D_m} + \overline{D_m} \cdot R_m + S_m \cdot R_m$
TAS	—	↑	↑	0	—	$N = D_m$ $Z = \overline{D_m} \cdot \overline{D_{m-1}} \cdot \dots \cdot \overline{D_0}$
TRAPA	—	—	—	—	—	
XOR	—	↑	↑	0	—	$N = R_m$ $Z = \overline{R_m} \cdot \overline{R_{m-1}} \cdot \dots \cdot \overline{R_0}$
XORC	↑	↑	↑	↑	↑	Stores the corresponding bits of the result. No flags change when the operand is EXR.

# Appendix B Internal I/O Registers

## B.1 Addresses

Address	Register									Module Name	Data Bus Width	
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
H'F800	MRA	SM1	SM0	DM1	DM0	MD1	MD0	DTS	Sz	DTC	16/32* bits	
to	SAR											
H'FBFF												
	MRB	CHNE	DISEL	CHNS	—	—	—	—	—			
	DAR											
	CRA											
	CRB											
H'FE80	TCR3	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU3		16 bits
H'FE81	TMDR3	—	—	BFB	BFA	MD3	MD2	MD1	MD0			
H'FE82	TIOR3H	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0			
H'FE83	TIOR3L	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0			
H'FE84	TIER3	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA			
H'FE85	TSR3	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA			
H'FE86	TCNT3											
H'FE87												
H'FE88	TGR3A											
H'FE89												
H'FE8A	TGR3B											
H'FE8B												
H'FE8C	TGR3C											
H'FE8D												
H'FE8E	TGR3D											
H'FE8F												

Note: \* Located in on-chip RAM. The bus width is 32 bits when the DTC accesses this area as register information, and 16 bits otherwise.



Address	Register										Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
H'FE90	TCR4	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0		TPU4	16 bits
H'FE91	TMDR4	—	—	—	—	MD3	MD2	MD1	MD0			
H'FE92	TIOR4	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0			
H'FE94	TIER4	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA			
H'FE95	TSR4	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA			
H'FE96	TCNT4											
H'FE97												
H'FE98	TGR4A											
H'FE99												
H'FE9A	TGR4B											
H'FE9B												
H'FEA0	TCR5	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0		TPU5	16 bits
H'FEA1	TMDR5	—	—	—	—	MD3	MD2	MD1	MD0			
H'FEA2	TIOR5	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0			
H'FEA4	TIER5	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA			
H'FEA5	TSR5	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA			
H'FEA6	TCNT5											
H'FEA7												
H'FEA8	TGR5A											
H'FEA9												
H'FEAA	TGR5B											
H'FEAB												
H'FEB0	P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR		Ports	8 bits
H'FEB1	P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR			
H'FEB2	P3DDR	—	—	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR			
H'FEB4	P5DDR	—	—	—	—	P53DDR	P52DDR	P51DDR	P50DDR			
H'FEB5	P6DDR	P67DDR	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR			
H'FEB6	P7DDR	—	—	P75DDR	P74DDR	P73DDR	P72DDR	P71DDR	P70DDR			
H'FEB7	P8DDR	—	P86DDR	P85DDR	P84DDR	P83DDR	P82DDR	P81DDR	P80DDR			
H'FEB8	P9DDR	P97DDR	P96DDR	P95DDR	P94DDR	P93DDR	P92DDR	—	—			
H'FEB9	PADDR	PA7DDR	PA6DDR	PA5DDR	PA4DDR	PA3DDR	PA2DDR	PA1DDR	PA0DDR			
H'FEBA	PBDDR	PB7DDR	PB6DDR	PB5DDR	PB4DDR	PB3DDR	PB2DDR	PB1DDR	PB0DDR			
H'FEBB	PCDDR	PC7DDR	PC6DDR	PC5DDR	PC4DDR	PC3DDR	PC2DDR	PC1DDR	PC0DDR			
H'FEBC	PDDDR	PD7DDR	PD6DDR	PD5DDR	PD4DDR	PD3DDR	PD2DDR	PD1DDR	PD0DDR			
H'FEBD	PEDDR	PE7DDR	PE6DDR	PE5DDR	PE4DDR	PE3DDR	PE2DDR	PE1DDR	PE0DDR			
H'FEBE	PFDDR	PF7DDR	PF6DDR	PF5DDR	PF4DDR	PF3DDR	PF2DDR	PF1DDR	PF0DDR			
H'FEBF	PGDDR	—	—	—	PG4DDR	PG3DDR	PG2DDR	PG1DDR	PG0DDR			

Address	Register									Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'FEC4	IPRA	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0	Interrupt controller	8 bits
H'FEC5	IPRB	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FEC6	IPRC	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FEC7	IPRD	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FEC8	IPRE	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FEC9	IPRF	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FECA	IPRG	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FECB	IPRH	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FECC	IPRI	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FECD	IPRJ	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0		
H'FECE	IPRK	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0	Bus controller	8 bits
H'FED0	ABWCR	ABW7	ABW6	ABW5	ABW4	ABW3	ABW2	ABW1	ABW0		
H'FED1	ASTCR	AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0		
H'FED2	WCRH	W71	W70	W61	W60	W51	W50	W41	W40		
H'FED3	WCRL	W31	W30	W21	W20	W11	W10	W01	W00		
H'FED4	BCRH	ICIS1	ICIS0	BRSTRM	BRSTS1	BRSTS0	RMTS2	RMTS1	RMST0		
H'FED5	BCRL	BRLE	BREQOE	EAE	—	DDS	—	WDBE	WAITE		
H'FED6	MCR	TPC	BE	RCDM	—	MXC1	MXC0	RLW1	RLW0		
H'FED7	DRAMCR	RFSHE	RCW	RMODE	CMF	CMIE	CKS2	CKS1	CKS0		
H'FED8	RTCNT										
H'FED9	RTCOR										
H'FEDB	RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0	DMAC	16 bits
H'FEE0	MAR0AH	—	—	—	—	—	—	—	—		
H'FEE1											
H'FEE2	MAR0AL										
H'FEE3											
H'FEE4	IOAR0A										
H'FEE5											
H'FEE6	ETCR0A										
H'FEE7											
H'FEE8	MAR0BH	—	—	—	—	—	—	—	—		
H'FEE9											
H'FEEA	MAR0BL										
H'FEEB											
H'FEEC	IOAR0B										
H'FEED											
H'FEEE	ETCR0B										
H'FEEF											

Address	Register									Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'FEF0	MAR1AH	—	—	—	—	—	—	—	—	DMAC	16 bits
H'FEF1											
H'FEF2	MAR1AL										
H'FEF3											
H'FEF4	IOAR1A										
H'FEF5											
H'FEF6	ETCR1A										
H'FEF7											
H'FEF8	MAR1BH	—	—	—	—	—	—	—	—		
H'FEF9											
H'FEFA	MAR1BL										
H'FEFB											
H'FEFC	IOAR1B										
H'FEFD											
H'FEFE	ETCR1B										
H'FEFF											
H'FF00	DMAWER	—	—	—	—	WE1B	WE1A	WE0B	WE0A		8 bits
H'FF01	DMATCR	—	—	TEE1	TEE0	—	—	—	—		
H'FF02	DMACR0A	DTSZ	DTID	RPE	DTDIR	DTF3	DTF2	DTF1	DTF0	Short address mode	16 bits
		DTSZ	SAID	SAIDE	BLKDIR	BLKE	—	—	—	Full address mode	
H'FF03	DMACR0B	DTSZ	DTID	RPE	DTDIR	DTF3	DTF2	DTF1	DTF0	Short address mode	
		—	DAID	DAIDE	—	DTF3	DTF2	DTF1	DTF0	Full address mode	
H'FF04	DMACR1A	DTSZ	DTID	RPE	DTDIR	DTF3	DTF2	DTF1	DTF0	Short address mode	
		DTSZ	SAID	SAIDE	BLKDIR	BLKE	—	—	—	Full address mode	
H'FF05	DMACR1B	DTSZ	DTID	RPE	DTDIR	DTF3	DTF2	DTF1	DTF0	Short address mode	
		—	DAID	DAIDE	—	DTF3	DTF2	DTF1	DTF0	Full address mode	
H'FF06	DMABCRH	FAE1	FAE0	SAE1	SAE0	DTA1B	DTA1A	DTA0B	DTA0A	Short address mode	
		FAE1	FAE0	—	—	DTA1	—	DTA0	—	Full address mode	
H'FF07	DMABCR L	DTE1B	DTE1A	DTE0B	DTE0A	DTIE1B	DTIE1A	DTIE0B	DTIE0A	Short address mode	
		DTME1	DTE1	DTME0	DTE0	DTIE1B	DTIE1A	DTIE0B	DTIE0A	Full address mode	

Address	Register									Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'FF2C	ISCRH	IRQ7SCB	IRQ7SCA	IRQ6SCB	IRQ6SCA	IRQ5SCB	IRQ5SCA	IRQ4SCB	IRQ4SCA	Interrupt controller	8 bits
H'FF2D	ISCR_L	IRQ3SCB	IRQ3SCA	IRQ2SCB	IRQ2SCA	IRQ1SCB	IRQ1SCA	IRQ0SCB	IRQ0SCA		
H'FF2E	IER	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E		
H'FF2F	ISR	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F		
H'FF30 to H'FF35	DTCE7 to DTCE0	DTCE7	DTCE6	DTCE5	DTCE4	DTCE3	DTCE2	DTCE1	DTCE0	DTC	8 bits
H'FF37	DTVECR	SWDTE	DTVEC6	DTVEC5	DTVEC4	DTVEC3	DTVEC2	DTVEC1	DTVEC0		
H'FF38	SBYCR	SSBY	STS2	STS1	STS0	OPE	—	—	IRQ37S	Power-down mode	8 bits
H'FF39	SYSCR	—	—	INTM1	INTM0	NMIEG	LWROD	IRQPAS	RAME	MCU	8 bits
H'FF3A	SCKCR	PSTOP	—	DIV	—	—	SCK2	SCK1	SCK0	Clock pulse generator	8 bits
H'FF3B	MDCR	—	—	—	—	—	MDS2	MDS1	MDS0	MCU	8 bits
H'FF3C	MSTPCR_H	MSTP15	MSTP14	MSTP13	MSTP12	MSTP11	MSTP10	MSTP9	MSTP8	Power-down mode	8 bits
H'FF3D	MSTPCR_L	MSTP7	MSTP6	MSTP5	MSTP4	MSTP3	MSTP2	MSTP1	MSTP0		
H'FF42	SYSCR2	—	—	—	—	FLSHE	—	—	—	MCU	8 bits
	*2										
H'FF44	Reserved	—	—	—	—	—	—	—	—	Reserved	—
H'FF45	PFCR1	—	—	—	—	A23E	A22E	A21E	A20E	Ports	8 bits
H'FF46	PCR	G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0	PPG	8 bits
H'FF47	PMR	G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV		
H'FF48	NDERH	NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8		
H'FF49	NDERL	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0		
H'FF4A	PODRH	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8		
H'FF4B	PODRL	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0		
H'FF4C*1	NDRH	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8		
H'FF4D*1	NDRL	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0		
H'FF4E*1	NDRH	—	—	—	—	NDR11	NDR10	NDR9	NDR8		
H'FF4F*1	NDRL	—	—	—	—	NDR3	NDR2	NDR1	NDR0		
H'FF50	PORT1	P17	P16	P15	P14	P13	P12	P11	P10	Ports	8 bits
H'FF51	PORT2	P27	P26	P25	P24	P23	P22	P21	P20		
H'FF52	PORT3	—	—	P35	P34	P33	P32	P31	P30		

- Notes:
1. If the pulse output group 2 and pulse output group 3 output triggers are the same according to the PCR setting, the NDRH address will be H'FF4C, and if different, the address of NDRH for group 2 will be H'FF4E, and that for group 3 will be H'FF4C. Similarly, if the pulse output group 0 and pulse output group 1 output triggers are the same according to the PCR setting, the NDRL address will be H'FF4D, and if different, the address of NDRL for group 0 will be H'FF4F, and that for group 1 will be H'FF4D.
  2. Available only in the F-ZTAT version.

Address	Register									Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'FF53	PORT4	P47	P46	P45	P44	P43	P42	P41	P40	Ports	8 bits
H'FF54	PORT5	P57	P56	P55	P54	P53	P52	P51	P50		
H'FF55	PORT6	P67	P66	P65	P64	P63	P62	P61	P60		
H'FF56	PORT7	—	—	P75	P74	P73	P72	P71	P70		
H'FF57	PORT8	—	P86	P85	P84	P83	P82	P81	P80		
H'FF58	PORT9	P97	P96	P95	P94	P93	P92	—	—		
H'FF59	PORTA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0		
H'FF5A	PORTB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0		
H'FF5B	PORTC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0		
H'FF5C	PORTD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0		
H'FF5D	PORTE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0		
H'FF5E	PORTF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0		
H'FF5F	PORTG	—	—	—	PG4	PG3	PG2	PG1	PG0		
H'FF60	P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR		
H'FF61	P2DR	P27DR	P26DR	P25DR	P24DR	P23DR	P22DR	P21DR	P20DR		
H'FF62	P3DR	—	—	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR		
H'FF64	P5DR	—	—	—	—	P53DR	P52DR	P51DR	P50DR		
H'FF65	P6DR	P67DR	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR		
H'FF66	P7DR	—	—	P75DR	P74DR	P73DR	P72DR	P71DR	P70DR		
H'FF67	P8DR	—	P86DR	P85DR	P84DR	P83DR	P82DR	P81DR	P80DR		
H'FF68	P9DR	P97DR	P96DR	P95DR	P94DR	P93DR	P92DR	—	—		
H'FF69	PADR	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR		
H'FF6A	PBDR	PB7DR	PB6DR	PB5DR	PB4DR	PB3DR	PB2DR	PB1DR	PB0DR		
H'FF6B	PCDR	PC7DR	PC6DR	PC5DR	PC4DR	PC3DR	PC2DR	PC1DR	PC0DR		
H'FF6C	PDDR	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR		
H'FF6D	PEDR	PE7DR	PE6DR	PE5DR	PE4DR	PE3DR	PE2DR	PE1DR	PE0DR		
H'FF6E	PFDR	PF7DR	PF6DR	PF5DR	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR		
H'FF6F	PGDR	—	—	—	PG4DR	PG3DR	PG2DR	PG1DR	PG0DR		
H'FF70	PAPCR	PA7PCR	PA6PCR	PA5PCR	PA4PCR	PA3PCR	PA2PCR	PA1PCR	PA0PCR		
H'FF71	PBPCR	PB7PCR	PB6PCR	PB5PCR	PB4PCR	PB3PCR	PB2PCR	PB1PCR	PB0PCR		
H'FF72	PCPCR	PC7PCR	PC6PCR	PC5PCR	PC4PCR	PC3PCR	PC2PCR	PC1PCR	PC0PCR		
H'FF73	PDPCR	PD7PCR	PD6PCR	PD5PCR	PD4PCR	PD3PCR	PD2PCR	PD1PCR	PD0PCR		
H'FF74	PEPCR	PE7PCR	PE6PCR	PE5PCR	PE4PCR	PE3PCR	PE2PCR	PE1PCR	PE0PCR		
H'FF76	P3ODR	—	—	P35ODR	P34ODR	P33ODR	P32ODR	P31ODR	P30ODR		
H'FF77	PAODR	PA7ODR	PA6ODR	PA5ODR	PA4ODR	PA3ODR	PA2ODR	PA1ODR	PA0ODR		

Address	Register									Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'FF78	SMR0	C/ $\bar{A}$ / GM* <sup>3</sup>	CHR/ BLK* <sup>4</sup>	PE	O/ $\bar{E}$	STOP/ BCP1* <sup>5</sup>	MP/ BCP0* <sup>6</sup>	CKS1	CKS0	SCI0, smart card interface 0	8 bits
H'FF79	BRR0										
H'FF7A	SCR0	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0		
H'FF7B	TDR0										
H'FF7C	SSR0	TDRE	RDRF	ORER	FER/ ERS* <sup>7</sup>	PER	TEND	MPB	MPBT		
H'FF7D	RDR0										
H'FF7E	SCMR0	—	—	—	—	SDIR	SINV	—	SMIF		
H'FF80	SMR1	C/ $\bar{A}$ / GM* <sup>3</sup>	CHR/ BLK* <sup>4</sup>	PE	O/ $\bar{E}$	STOP/ BCP1* <sup>5</sup>	MP/ BCP0* <sup>6</sup>	CKS1	CKS0	SCI1, smart card interface 1	8 bits
H'FF81	BRR1										
H'FF82	SCR1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0		
H'FF83	TDR1										
H'FF84	SSR1	TDRE	RDRF	ORER	FER/ ERS* <sup>7</sup>	PER	TEND	MPB	MPBT		
H'FF85	RDR1										
H'FF86	SCMR1	—	—	—	—	SDIR	SINV	—	SMIF		
H'FF88	SMR2	C/ $\bar{A}$ / GM* <sup>3</sup>	CHR/ BLK* <sup>4</sup>	PE	O/ $\bar{E}$	STOP/ BCP1* <sup>5</sup>	MP/ BCP0* <sup>6</sup>	CKS1	CKS0	SCI2, smart card interface 2	8 bits
H'FF89	BRR2										
H'FF8A	SCR2	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0		
H'FF8B	TDR2										
H'FF8C	SSR2	TDRE	RDRF	ORER	FER/ ERS* <sup>7</sup>	PER	TEND	MPB	MPBT		
H'FF8D	RDR2										
H'FF8E	SCMR2	—	—	—	—	SDIR	SINV	—	SMIF		

- Notes:
3. Functions as C/ $\bar{A}$  for SCI use, and as GM for smart card interface use.
  4. Functions as CHR for SCI use, and as BLK for smart card interface use.
  5. Functions as STOP for SCI use, and as BCP1 for smart card interface use.
  6. Functions as MP for SCI use, and as BCP0 for smart card interface use.
  7. Functions as FER for SCI use, and as ERS for smart card interface use.

Address	Register									Module Name	Data Bus Width		
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0				
H'FF90	ADDRAH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D converter	8 bits		
H'FF91	ADDRAL	AD1	AD0	—	—	—	—	—	—				
H'FF92	ADDRBH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2				
H'FF93	ADDRBL	AD1	AD0	—	—	—	—	—	—				
H'FF94	ADDRCH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2				
H'FF95	ADDRCL	AD1	AD0	—	—	—	—	—	—				
H'FF96	ADDRDH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2				
H'FF97	ADDRDL	AD1	AD0	—	—	—	—	—	—				
H'FF98	ADCSR	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0				
H'FF99	ADCR	TRGS1	TRGS0	—	—	CKS1	CH3	—	—				
H'FFA4	DADR0									D/A converter	8 bits		
H'FFA5	DADR1												
H'FFA6	DACR	DAOE1	DAOE0	DAE	—	—	—	—	—				
H'FFA8	DADR2												
H'FFA9	DADR3												
H'FFAA	DACR23	DAOE1	DAOE0	DAE	—	—	—	—	—				
H'FFAC	PFCR2	WAITPS	BREQOPS	CS167E	CS25E	ASOD	—	—	—			8 bits	
H'FFB0	TCR0	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0			8-bit timer channels 0, 1	16 bits
H'FFB1	TCR1	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0				
H'FFB2	TCSR0	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0				
H'FFB3	TCSR1	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0				
H'FFB4	TCORA0												
H'FFB5	TCORA1												
H'FFB6	TCORB0												
H'FFB7	TCORB1												
H'FFB8	TCNT0												
H'FFB9	TCNT1												
H'FFBC	TCSR	OVF	WT/IT	TME	—	—	CKS2	CKS1	CKS0	WDT	16 bits		
(read)													
H'FFBD	TCNT												
(read)													
H'FFBF	RSTCSR	WOVF	RSTE	—	—	—	—	—	—				
(read)													
H'FFC0	TSTR	—	—	CST5	CST4	CST3	CST2	CST1	CST0	TPU	16 bits		
H'FFC1	TSYR	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0				

Register										Module Name	Data Bus Width
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
H'FFC8**	FLMCR1	FWE	SWE	ESU	PSU	EV	PV	E	P	FLASH	8 bits
H'FFC9**	FLMCR2	FLER	—	—	—	—	—	—	—		
H'FFCA**	EBR1	EB7	EB6	EB5	EB4	EB3	EB2	EB1	EB0		
H'FFCB**	EBR2	—	—	—	—	EB11	EB10	EB9	EB8		
H'FFD0	TCR0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU0	16 bits
H'FFD1	TMDR0	—	—	BFB	BFA	MD3	MD2	MD1	MD0		
H'FFD2	TIOR0H	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0		
H'FFD3	TIOR0L	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0		
H'FFD4	TIER0	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA		
H'FFD5	TSR0	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA		
H'FFD6	TCNT0										
H'FFD7											
H'FFD8	TGR0A										
H'FFD9											
H'FFDA	TGR0B										
H'FFDB											
H'FFDC	TGR0C										
H'FFDD											
H'FFDE	TGR0D										
H'FFDF											
H'FFE0	TCR1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU1	16 bits
H'FFE1	TMDR1	—	—	—	—	MD3	MD2	MD1	MD0		
H'FFE2	TIOR1	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0		
H'FFE4	TIER1	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA		
H'FFE5	TSR1	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA		
H'FFE6	TCNT1										
H'FFE7											
H'FFE8	TGR1A										
H'FFE9											
H'FFEA	TGR1B										
H'FFEB											

Notes: 8. Available only in the F-ZTAT version.



Address	Register										Module Name	Data Bus Width
	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0			
H'FFF0	TCR2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0		TPU2	16 bits
H'FFF1	TMDR2	—	—	—	—	MD3	MD2	MD1	MD0			
H'FFF2	TIOR2	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0			
H'FFF4	TIER2	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA			
H'FFF5	TSR2	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA			
H'FFF6	TCNT2											
H'FFF7												
H'FFF8	TGR2A											
H'FFF9												
H'FFFA	TGR2B											
H'FFFB												

---

## **H8S/2338 Series, H8S/2328 Series, H8S/2318 Series Hardware Manual**

Publication Date: 1st Edition, March 1999

Published by: Electronic Devices Sales & Marketing Group  
Semiconductor & Integrated Circuits Group  
Hitachi, Ltd.

Edited by: Technical Documentation Group  
UL Media Co., Ltd.

Copyright © Hitachi, Ltd., 1999. All rights reserved. Printed in Japan.