

**32**

# H8SX/1622 Group

## Hardware Manual

Renesas 32-Bit CISC Microcomputer  
H8SX Family / H8SX/1600 Series

H8SX/1622

R5F61622

All information contained in this material, including products and product specifications at the time of publication of this material, is subject to change by Renesas Technology Corp. without notice. Please review the latest information published by Renesas Technology Corp. through various means, including the Renesas Technology Corp. website (<http://www.renesas.com>).

Rev.1.00

Revision Date: Nov. 01, 2007

Renesas Technology  
[www.renesas.com](http://www.renesas.com)



## Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (<http://www.renesas.com>)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
  - (1) artificial life support devices or systems
  - (2) surgical implantations
  - (3) healthcare intervention (e.g., excision, administration of medication, etc.)
  - (4) any other purposes that pose a direct threat to human lifeRenesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

# General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

## 1. Handling of Unused Pins

Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

- The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

## 2. Processing at Power-on

The state of the product is undefined at the moment when power is supplied.

- The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

## 3. Prohibition of Access to Reserved Addresses

Access to reserved addresses is prohibited.

- The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

## 4. Clock Signals

After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

- When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

## 5. Differences between Products

Before changing from one product to another, i.e. to one with a different part number, confirm that the change will not lead to problems.

- The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

# How to Use This Manual

## 1. Objective and Target Users

This manual was written to explain the hardware functions and electrical characteristics of this LSI to the target users, i.e. those who will be using this LSI in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logic circuits, and microcomputers.

This manual is organized in the following items: an overview of the product, descriptions of the CPU, system control functions, and peripheral functions, electrical characteristics of the device, and usage notes.

When designing an application system that includes this LSI, take all points to note into account. Points to note are given in their contexts and at the final part of each section, and in the section giving usage notes.

The list of revisions is a summary of major points of revision or addition for earlier versions. It does not cover all revised items. For details on the revised points, see the actual locations in the manual.

The following documents have been prepared for the H8SX/1622 Group. Before using any of the documents, please visit our web site to verify that you have the most up-to-date available version of the document.

Document Type	Contents	Document Title	Document No.
Data Sheet	Overview of hardware and electrical characteristics	—	—
Hardware Manual	Hardware specifications (pin assignments, memory maps, peripheral specifications, electrical characteristics, and timing charts) and descriptions of operation	H8SX/1622 Group Hardware Manual	This manual
Software Manual	Detailed descriptions of the CPU and instruction set	H8SX Family Software Manual	REJ09B0102
Application Note	Examples of applications and sample programs	The latest versions are available from our web site.	
Renesas Technical Update	Preliminary report on the specifications of a product, document, etc.		

## 2. Description of Numbers and Symbols

Aspects of the notations for register names, bit names, numbers, and symbolic names in this manual are explained below.

### (1) Overall notation

In descriptions involving the names of bits and bit fields within this manual, the modules and registers to which the bits belong may be clarified by giving the names in the forms "module name"."register name"."bit name" or "register name"."bit name".

### (2) Register notation

The style "register name"\_"instance number" is used in cases where there is more than one instance of the same function or similar functions.

[Example] CMCSR\_0: Indicates the CMCSR register for the compare-match timer of channel 0.

### (3) Number notation

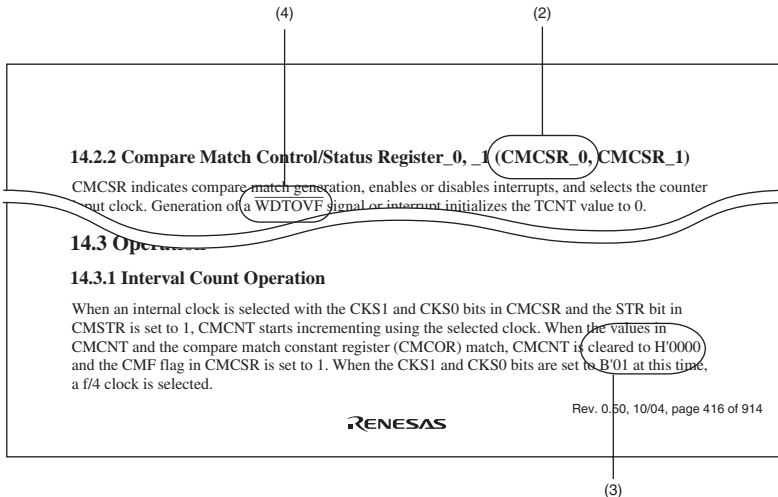
Binary numbers are given as B'nnnn (B' may be omitted if the number is obviously binary), hexadecimal numbers are given as H'nnnn or 0xnnnn, and decimal numbers are given as nnnn.

[Examples] Binary: B'11 or 11  
Hexadecimal: H'EFA0 or 0xEFA0  
Decimal: 1234

### (4) Notation for active-low

An overbar on the name indicates that a signal or pin is active-low.

[Example]  $\overline{\text{WDTOVF}}$

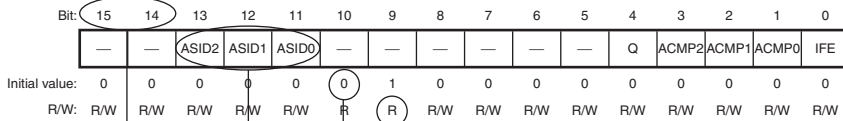


Note: The bit names and sentences in the above figure are examples and have nothing to do with the contents of this manual.

### 3. Description of Registers

Each register description includes a bit chart, illustrating the arrangement of bits, and a table of bits, describing the meanings of the bit settings. The standard format and notation for bit charts and tables are described below.

[Bit Chart]



[Table of Bits]

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved
14	—	0	R	Reserved
13 to 11	ASID2 to ASID0	All 0	R/W	Address Identifier These bits enable or disable the pin function.
10	—	0	R	Reserved This bit is always read as 0.
9	—	1	R	Reserved This bit is always read as 1.
8 to 0	—	0	—	Reserved

Note: The bit names and sentences in the above figure are examples, and have nothing to do with the contents of this manual.

- (1) Bit  
Indicates the bit number or numbers.  
In the case of a 32-bit register, the bits are arranged in order from 31 to 0. In the case of a 16-bit register, the bits are arranged in order from 15 to 0.
- (2) Bit name  
Indicates the name of the bit or bit field.  
When the number of bits has to be clearly indicated in the field, appropriate notation is included (e.g., ASID[3:0]).  
A reserved bit is indicated by "—".  
Certain kinds of bits, such as those of timer counters, are not assigned bit names. In such cases, the entry under Bit Name is blank.
- (3) Initial value  
Indicates the value of each bit immediately after a power-on reset, i.e., the initial value.  
0: The initial value is 0  
1: The initial value is 1  
—: The initial value is undefined
- (4) R/W  
For each bit and bit field, this entry indicates whether the bit or field is readable or writable, or both writing to and reading from the bit or field are impossible.  
The notation is as follows:  
R/W: The bit or field is readable and writable.  
R/(W): The bit or field is readable and writable.  
However, writing is only performed to flag clearing.  
R: The bit or field is readable.  
"R" is indicated for all reserved bits. When writing to the register, write the value under Initial Value in the bit chart to reserved bits or fields.  
W: The bit or field is writable.
- (5) Description  
Describes the function of the bit or field and specifies the values for writing.

#### 4. Description of Abbreviations

The abbreviations used in this manual are listed below.

- Abbreviations specific to this product

<b>Abbreviation</b>	<b>Description</b>
BSC	Bus controller
CPG	Clock pulse generator
DTC	Data transfer controller
INTC	Interrupt controller
PPG	Programmable pulse generator
SCI	Serial communication interface
TMR	8-bit timer
TPU	16-bit timer pulse unit
WDT	Watchdog timer
UBC	User break controller

- Abbreviations other than those listed above

<b>Abbreviation</b>	<b>Description</b>
ACIA	Asynchronous communication interface adapter
bps	Bits per second
DMA	Direct memory access
DMAC	Direct memory access controller
GSM	Global System for Mobile Communications
Hi-Z	High impedance
I/O	Input/output
LSB	Least significant bit
MSB	Most significant bit
NC	No connection
PLL	Phase-locked loop
PWM	Pulse width modulation
SIM	Subscriber Identity Module
UART	Universal asynchronous receiver/transmitter

All trademarks and registered trademarks are the property of their respective owners.



# Contents

Section 1	Overview	1
1.1	Features	1
1.1.1	Applications	1
1.1.2	Overview of Functions	2
1.2	List of Products	8
1.3	Block Diagram	9
1.4	Pin Descriptions	10
1.4.1	Pin Assignments	10
1.4.2	Pin Assignment for Each Operating Mode	12
1.4.3	Pin Functions	18
Section 2	CPU	25
2.1	Features	25
2.2	CPU Operating Modes	27
2.2.1	Normal Mode	27
2.2.2	Middle Mode	29
2.2.3	Advanced Mode	30
2.2.4	Maximum Mode	31
2.3	Instruction Fetch	33
2.4	Address Space	33
2.5	Registers	34
2.5.1	General Registers	35
2.5.2	Program Counter (PC)	36
2.5.3	Condition-Code Register (CCR)	37
2.5.4	Extended Control Register (EXR)	38
2.5.5	Vector Base Register (VBR)	39
2.5.6	Short Address Base Register (SBR)	39
2.5.7	Multiply-Accumulate Register (MAC)	39
2.5.8	Initial Values of CPU Registers	39
2.6	Data Formats	40
2.6.1	General Register Data Formats	40
2.6.2	Memory Data Formats	41
2.7	Instruction Set	42
2.7.1	Instructions and Addressing Modes	44
2.7.2	Table of Instructions Classified by Function	48
2.7.3	Basic Instruction Formats	58

2.8	Addressing Modes and Effective Address Calculation.....	59
2.8.1	Register Direct—Rn .....	59
2.8.2	Register Indirect—@ERn.....	60
2.8.3	Register Indirect with Displacement —@(d:2, ERn), @(d:16, ERn), or @(d:32, ERn).....	60
2.8.4	Index Register Indirect with Displacement—@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L).....	60
2.8.5	Register Indirect with Post-Increment, Pre-Decrement, Pre-Increment, or Post-Decrement—@ERn+, @-ERn, @+ERn, or @ERn-.....	61
2.8.6	Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32.....	62
2.8.7	Immediate—#xx .....	63
2.8.8	Program-Counter Relative—@(d:8, PC) or @(d:16, PC) .....	63
2.8.9	Program-Counter Relative with Index Register—@(RnL.B, PC), @(Rn.W, PC), or @(ERn.L, PC).....	63
2.8.10	Memory Indirect—@@aa:8 .....	64
2.8.11	Extended Memory Indirect—@@vec:7 .....	65
2.8.12	Effective Address Calculation .....	65
2.8.13	MOVA Instruction.....	67
2.9	Processing States .....	68
<b>Section 3 MCU Operating Modes .....</b>		<b>69</b>
3.1	Operating Mode Selection .....	69
3.2	Register Descriptions.....	70
3.2.1	Mode Control Register (MDCR) .....	70
3.2.2	System Control Register (SYSCR).....	72
3.3	Operating Mode Descriptions .....	74
3.3.1	Mode 1 .....	74
3.3.2	Mode 2.....	74
3.3.3	Mode 4.....	74
3.3.4	Mode 5.....	74
3.3.5	Mode 6.....	75
3.3.6	Mode 7.....	75
3.3.7	Pin Functions .....	76
3.4	Address Map.....	76
3.4.1	Address Map.....	76
<b>Section 4 Resets.....</b>		<b>79</b>
4.1	Types of Resets.....	79
4.2	Input/Output Pin .....	80
4.3	Register Descriptions.....	81

4.3.1	Reset Status Register (RSTSR).....	81
4.3.2	Reset Control/Status Register (RSTCSR).....	82
4.4	Pin Reset.....	83
4.5	Deep Software Standby Reset.....	83
4.6	Watchdog Timer Reset.....	83
4.7	Determination of Reset Generation Source.....	83
<b>Section 5 Exception Handling .....</b>		<b>85</b>
5.1	Exception Handling Types and Priority.....	85
5.2	Exception Sources and Exception Handling Vector Table.....	86
5.3	Reset.....	88
5.3.1	Reset Exception Handling.....	89
5.3.2	Interrupts after Reset.....	89
5.3.3	On-Chip Peripheral Functions after Reset Release.....	90
5.4	Traces.....	92
5.5	Address Error.....	93
5.5.1	Address Error Source.....	93
5.5.2	Address Error Exception Handling.....	94
5.6	Interrupts.....	95
5.6.1	Interrupt Sources.....	95
5.6.2	Interrupt Exception Handling.....	96
5.7	Instruction Exception Handling.....	97
5.7.1	Trap Instruction.....	97
5.7.2	Sleep Instruction.....	98
5.7.3	Illegal Instruction.....	99
5.8	Stack Status after Exception Handling.....	100
5.9	Usage Note.....	101
<b>Section 6 Interrupt Controller .....</b>		<b>103</b>
6.1	Features.....	103
6.2	Input/Output Pins.....	105
6.3	Register Descriptions.....	105
6.3.1	Interrupt Control Register (INTCR).....	106
6.3.2	CPU Priority Control Register (CPUPCR).....	107
6.3.3	Interrupt Priority Registers A to I, K, L, P to R (IPRA to IPRI, IPRK, IPRL, IPRP to IPRR).....	109
6.3.4	IRQ Enable Register (IER).....	111
6.3.5	IRQ Sense Control Registers H and L (ISCRH, ISCLR).....	113
6.3.6	IRQ Status Register (ISR).....	118
6.3.7	Software Standby Release IRQ Enable Register (SSIER).....	119

6.4	Interrupt Sources.....	120
6.4.1	External Interrupts .....	120
6.4.2	Internal Interrupts .....	121
6.5	Interrupt Exception Handling Vector Table.....	122
6.6	Interrupt Control Modes and Interrupt Operation.....	127
6.6.1	Interrupt Control Mode 0.....	127
6.6.2	Interrupt Control Mode 2.....	129
6.6.3	Interrupt Exception Handling Sequence .....	131
6.6.4	Interrupt Response Times .....	132
6.6.5	DTC and DMAC Activation by Interrupt.....	133
6.7	CPU Priority Control Function Over DTC and DMAC.....	136
6.8	Usage Notes.....	139
6.8.1	Conflict between Interrupt Generation and Disabling .....	139
6.8.2	Instructions that Disable Interrupts.....	140
6.8.3	Times when Interrupts are Disabled .....	140
6.8.4	Interrupts during Execution of EEPMOV Instruction .....	140
6.8.5	Interrupts during Execution of MOVMD and MOVSD Instructions.....	140
6.8.6	Interrupts of Peripheral Modules .....	141
Section 7 User Break Controller (UBC).....		143
7.1	Features.....	143
7.2	Block Diagram.....	144
7.3	Register Descriptions.....	145
7.3.1	Break Address Register n (BARA, BARB, BARC, BARD) .....	146
7.3.2	Break Address Mask Register n (BAMRA, BAMRB, BAMRC, BAMRD) ....	147
7.3.3	Break Control Register n (BRCRA, BRCRB, BRCRC, BRCRD) .....	148
7.4	Operation .....	150
7.4.1	Setting of Break Control Conditions.....	150
7.4.2	PC Break.....	150
7.4.3	Condition Match Flag .....	151
7.5	Usage Notes.....	152
Section 8 Bus Controller (BSC) .....		155
8.1	Features.....	155
8.2	Register Descriptions.....	158
8.2.1	Bus Width Control Register (ABWCR).....	159
8.2.2	Access State Control Register (ASTCR) .....	160
8.2.3	Wait Control Registers A and B (WTCRA, WTCRB) .....	161
8.2.4	Read Strobe Timing Control Register (RDNCR) .....	166
8.2.5	$\overline{\text{CS}}$ Assertion Period Control Registers (CSACR) .....	167

8.2.6	Idle Control Register (IDLCR) .....	170
8.2.7	Bus Control Register 1 (BCR1) .....	172
8.2.8	Bus Control Register 2 (BCR2) .....	174
8.2.9	Endian Control Register (ENDIANCR).....	175
8.2.10	SRAM Mode Control Register (SRAMCR) .....	176
8.2.11	Burst ROM Interface Control Register (BROMCR).....	177
8.2.12	Address/Data Multiplexed I/O Control Register (MPXCR) .....	179
8.3	Bus Configuration.....	180
8.4	Multi-Clock Function and Number of Access Cycles .....	181
8.5	External Bus.....	185
8.5.1	Input/Output Pins .....	185
8.5.2	Area Division.....	188
8.5.3	Chip Select Signals .....	189
8.5.4	External Bus Interface.....	190
8.5.5	Area and External Bus Interface .....	194
8.5.6	Endian and Data Alignment.....	199
8.6	Basic Bus Interface .....	202
8.6.1	Data Bus.....	202
8.6.2	I/O Pins Used for Basic Bus Interface .....	202
8.6.3	Basic Timing.....	203
8.6.4	Wait Control .....	209
8.6.5	Read Strobe ( $\overline{RD}$ ) Timing.....	211
8.6.6	Extension of Chip Select ( $\overline{CS}$ ) Assertion Period.....	212
8.6.7	$\overline{DACK}$ Signal Output Timing .....	214
8.7	Byte Control SRAM Interface .....	215
8.7.1	Byte Control SRAM Space Setting.....	215
8.7.2	Data Bus.....	215
8.7.3	I/O Pins Used for Byte Control SRAM Interface .....	216
8.7.4	Basic Timing.....	217
8.7.5	Wait Control .....	219
8.7.6	Read Strobe ( $\overline{RD}$ ).....	221
8.7.7	Extension of Chip Select ( $\overline{CS}$ ) Assertion Period.....	221
8.7.8	$\overline{DACK}$ Signal Output Timing .....	221
8.8	Burst ROM Interface .....	223
8.8.1	Burst ROM Space Setting.....	223
8.8.2	Data Bus.....	223
8.8.3	I/O Pins Used for Burst ROM Interface.....	224
8.8.4	Basic Timing.....	225
8.8.5	Wait Control .....	227
8.8.6	Read Strobe ( $\overline{RD}$ ) Timing.....	227

8.8.7	Extension of Chip Select ( $\overline{CS}$ ) Assertion Period.....	227
8.9	Address/Data Multiplexed I/O Interface.....	228
8.9.1	Address/Data Multiplexed I/O Space Setting .....	228
8.9.2	Address/Data Multiplex .....	228
8.9.3	Data Bus .....	228
8.9.4	I/O Pins Used for Address/Data Multiplexed I/O Interface .....	229
8.9.5	Basic Timing.....	230
8.9.6	Address Cycle Control.....	232
8.9.7	Wait Control .....	233
8.9.8	Read Strobe ( $\overline{RD}$ ) Timing .....	233
8.9.9	Extension of Chip Select ( $\overline{CS}$ ) Assertion Period.....	235
8.9.10	DACK Signal Output Timing .....	237
8.10	Idle Cycle.....	238
8.10.1	Operation .....	238
8.10.2	Pin States in Idle Cycle.....	247
8.11	Bus Release.....	248
8.11.1	Operation .....	248
8.11.2	Pin States in External Bus Released State .....	249
8.11.3	Transition Timing .....	250
8.12	Internal Bus.....	251
8.12.1	Access to Internal Address Space .....	251
8.13	Write Data Buffer Function .....	252
8.13.1	Write Data Buffer Function for External Data Bus .....	252
8.13.2	Write Data Buffer Function for Peripheral Modules .....	253
8.14	Bus Arbitration .....	254
8.14.1	Operation .....	254
8.14.2	Bus Transfer Timing .....	255
8.15	Bus Controller Operation in Reset.....	257
8.16	Usage Notes .....	257
<b>Section 9 DMA Controller (DMAC).....</b>		<b>259</b>
9.1	Features.....	259
9.2	Input/Output Pins.....	262
9.3	Register Descriptions.....	263
9.3.1	DMA Source Address Register (DSAR) .....	264
9.3.2	DMA Destination Address Register (DDAR) .....	265
9.3.3	DMA Offset Register (DOFR).....	266
9.3.4	DMA Transfer Count Register (DTCR) .....	267
9.3.5	DMA Block Size Register (DBSR) .....	268
9.3.6	DMA Mode Control Register (DMDR).....	269

9.3.7	DMA Address Control Register (DACR).....	278
9.3.8	DMA Module Request Select Register (DMRSR) .....	284
9.4	Transfer Modes.....	285
9.5	Operations.....	286
9.5.1	Address Modes .....	286
9.5.2	Transfer Modes .....	290
9.5.3	Activation Sources.....	295
9.5.4	Bus Access Modes .....	297
9.5.5	Extended Repeat Area Function .....	299
9.5.6	Address Update Function using Offset .....	301
9.5.7	Register during DMA Transfer .....	306
9.5.8	Priority of Channels .....	311
9.5.9	DMA Basic Bus Cycle.....	313
9.5.10	Bus Cycles in Dual Address Mode .....	314
9.5.11	Bus Cycles in Single Address Mode.....	323
9.6	DMA Transfer End .....	328
9.7	Relationship among DMAC and Other Bus Masters .....	331
9.7.1	CPU Priority Control Function Over DMAC .....	331
9.7.2	Bus Arbitration among DMAC and Other Bus Masters .....	332
9.8	Interrupt Sources.....	333
9.9	Usage Notes .....	336
9.9.1	DMAC Register Access During Operation.....	336
9.9.2	Settings of Module Stop Function .....	336
9.9.3	Activation by $\overline{\text{DREQ}}$ Falling Edge .....	336
9.9.4	Acceptation of Activation Source.....	337
<b>Section 10 Data Transfer Controller (DTC) .....</b>		<b>339</b>
10.1	Features.....	339
10.2	Register Descriptions.....	341
10.2.1	DTC Mode Register A (MRA) .....	342
10.2.2	DTC Mode Register B (MRB).....	343
10.2.3	DTC Source Address Register (SAR).....	345
10.2.4	DTC Destination Address Register (DAR).....	345
10.2.5	DTC Transfer Count Register A (CRA) .....	346
10.2.6	DTC Transfer Count Register B (CRB).....	346
10.2.7	DTC enable registers A to H (DTCERA to DTCERH) .....	347
10.2.8	DTC Control Register (DTCCR) .....	348
10.2.9	DTC Vector Base Register (DTCVBR).....	349
10.3	Activation Sources .....	349
10.4	Location of Transfer Information and DTC Vector Table.....	350

10.5	Operation .....	355
10.5.1	Bus Cycle Division .....	357
10.5.2	Transfer Information Read Skip Function .....	359
10.5.3	Transfer Information Writeback Skip Function.....	360
10.5.4	Normal Transfer Mode .....	360
10.5.5	Repeat Transfer Mode .....	361
10.5.6	Block Transfer Mode .....	363
10.5.7	Chain Transfer .....	364
10.5.8	Operation Timing.....	365
10.5.9	Number of DTC Execution Cycles .....	367
10.5.10	DTC Bus Release Timing .....	368
10.5.11	DTC Priority Level Control to the CPU .....	368
10.6	DTC Activation by Interrupt.....	369
10.7	Examples of Use of the DTC.....	370
10.7.1	Normal Transfer Mode .....	370
10.7.2	Chain Transfer .....	370
10.7.3	Chain Transfer when Counter = 0.....	371
10.8	Interrupt Sources.....	373
10.9	Usage Notes .....	373
10.9.1	Module Stop Function Setting .....	373
10.9.2	On-Chip RAM .....	373
10.9.3	DMAC Transfer End Interrupt.....	373
10.9.4	DTCE Bit Setting.....	373
10.9.5	Chain Transfer .....	374
10.9.6	Transfer Information Start Address, Source Address, and Destination Address .....	374
10.9.7	Transfer Information Modification .....	374
10.9.8	Endian Format .....	374
Section 11 I/O Ports.....		375
11.1	Register Descriptions.....	382
11.1.1	Data Direction Register (PnDDR) (n = 1 to 3, 6, A, D to F, H, and I).....	383
11.1.2	Data Register (PnDR) (n = 1 to 3, 6, A, D to F, H, and I) .....	384
11.1.3	Port Register (PORTn) (n = 1 to 6, A, D to F, H, and I).....	384
11.1.4	Input Buffer Control Register (PnICR) (n = 1 to 6, A, D to F, H, and I).....	385
11.1.5	Pull-Up MOS Control Register (PnPCR) (n = D to F, H, and I) .....	386
11.1.6	Open-Drain Control Register (PnODR) (n = 2 and F).....	388
11.2	Output Buffer Control.....	389
11.2.1	Port 1.....	389
11.2.2	Port 2.....	393



11.2.3	Port 3.....	397
11.2.4	Port 5.....	402
11.2.5	Port 6.....	403
11.2.6	Port A.....	406
11.2.7	Port D.....	410
11.2.8	Port E.....	410
11.2.9	Port F.....	412
11.2.10	Port H.....	415
11.2.11	Port I.....	416
11.3	Port Function Controller.....	424
11.3.1	Port Function Control Register 0 (PFCR0).....	424
11.3.2	Port Function Control Register 1 (PFCR1).....	425
11.3.3	Port Function Control Register 2 (PFCR2).....	426
11.3.4	Port Function Control Register 4 (PFCR4).....	428
11.3.5	Port Function Control Register 6 (PFCR6).....	429
11.3.6	Port Function Control Register 7 (PFCR7).....	430
11.3.7	Port Function Control Register 9 (PFCR9).....	431
11.3.8	Port Function Control Register B (PFCRB).....	433
11.3.9	Port Function Control Register C (PFCRC).....	434
11.4	Usage Notes.....	436
11.4.1	Notes on Input Buffer Control Register (ICR) Setting.....	436
11.4.2	Notes on Port Function Control Register (PFCR) Settings.....	436
Section 12 16-Bit Timer Pulse Unit (TPU).....		437
12.1	Features.....	437
12.2	Input/Output Pins.....	441
12.3	Register Descriptions.....	442
12.3.1	Timer Control Register (TCR).....	445
12.3.2	Timer Mode Register (TMDR).....	450
12.3.3	Timer I/O Control Register (TIOR).....	451
12.3.4	Timer Interrupt Enable Register (TIER).....	469
12.3.5	Timer Status Register (TSR).....	471
12.3.6	Timer Counter (TCNT).....	475
12.3.7	Timer General Register (TGR).....	475
12.3.8	Timer Start Register (TSTR).....	476
12.3.9	Timer Synchronous Register (TSYR).....	477
12.4	Operation.....	478
12.4.1	Basic Functions.....	478
12.4.2	Synchronous Operation.....	484
12.4.3	Buffer Operation.....	486

12.4.4	Cascaded Operation .....	490
12.4.5	PWM Modes .....	492
12.4.6	Phase Counting Mode .....	498
12.5	Interrupt Sources .....	504
12.6	DTC Activation .....	507
12.7	DMAC Activation .....	507
12.8	A/D Converter Activation .....	507
12.9	Operation Timing .....	508
12.9.1	Input/Output Timing .....	508
12.9.2	Interrupt Signal Timing .....	512
12.10	Usage Notes .....	516
12.10.1	Module Stop Function Setting .....	516
12.10.2	Input Clock Restrictions .....	516
12.10.3	Caution on Cycle Setting .....	517
12.10.4	Conflict between TCNT Write and Clear Operations .....	517
12.10.5	Conflict between TCNT Write and Increment Operations .....	518
12.10.6	Conflict between TGR Write and Compare Match .....	519
12.10.7	Conflict between Buffer Register Write and Compare Match .....	520
12.10.8	Conflict between TGR Read and Input Capture .....	521
12.10.9	Conflict between TGR Write and Input Capture .....	522
12.10.10	Conflict between Buffer Register Write and Input Capture .....	523
12.10.11	Conflict between Overflow/Underflow and Counter Clearing .....	524
12.10.12	Conflict between TCNT Write and Overflow/Underflow .....	525
12.10.13	Multiplexing of I/O Pins .....	525
12.10.14	Interrupts and Module Stop State .....	525
<b>Section 13 Programmable Pulse Generator (PPG) .....</b>		<b>527</b>
13.1	Features .....	527
13.2	Input/Output Pins .....	529
13.3	Register Descriptions .....	530
13.3.1	Next Data Enable Registers H, L (NDERH, NDERL) .....	530
13.3.2	Output Data Registers H, L (PODRH, PODRL) .....	532
13.3.3	Next Data Registers H, L (NDRH, NDRL) .....	533
13.3.4	PPG Output Control Register (PCR) .....	536
13.3.5	PPG Output Mode Register (PMR) .....	537
13.4	Operation .....	539
13.4.1	Output Timing .....	540
13.4.2	Sample Setup Procedure for Normal Pulse Output .....	541
13.4.3	Example of Normal Pulse Output (Example of 5-Phase Pulse Output) .....	542
13.4.4	Non-Overlapping Pulse Output .....	543

13.4.5	Sample Setup Procedure for Non-Overlapping Pulse Output .....	545
13.4.6	Example of Non-Overlapping Pulse Output (Example of 4-Phase Complementary Non-Overlapping Pulse Output).....	546
13.4.7	Inverted Pulse Output .....	548
13.4.8	Pulse Output Triggered by Input Capture .....	549
13.5	Usage Notes .....	550
13.5.1	Module Stop Function Setting .....	550
13.5.2	Operation of Pulse Output Pins.....	550
<b>Section 14 8-Bit Timers (TMR).....</b>		<b>551</b>
14.1	Features.....	551
14.2	Input/Output Pins .....	556
14.3	Register Descriptions .....	557
14.3.1	Timer Counter (TCNT).....	559
14.3.2	Time Constant Register A (TCORA).....	559
14.3.3	Time Constant Register B (TCORB) .....	560
14.3.4	Timer Control Register (TCR).....	560
14.3.5	Timer Counter Control Register (TCCR) .....	562
14.3.6	Timer Control/Status Register (TCSR).....	567
14.4	Operation .....	572
14.4.1	Pulse Output.....	572
14.4.2	Reset Input .....	573
14.5	Operation Timing.....	574
14.5.1	TCNT Count Timing .....	574
14.5.2	Timing of CMFA and CMFB Setting at Compare Match.....	575
14.5.3	Timing of Timer Output at Compare Match .....	575
14.5.4	Timing of Counter Clear by Compare Match .....	576
14.5.5	Timing of TCNT External Reset.....	576
14.5.6	Timing of Overflow Flag (OVF) Setting .....	577
14.6	Operation with Cascaded Connection.....	577
14.6.1	16-Bit Counter Mode .....	577
14.6.2	Compare Match Count Mode.....	578
14.7	Interrupt Sources.....	578
14.7.1	Interrupt Sources and DTC Activation .....	578
14.7.2	A/D Converter Activation.....	580
14.8	Usage Notes .....	581
14.8.1	Notes on Setting Cycle.....	581
14.8.2	Conflict between TCNT Write and Counter Clear.....	581
14.8.3	Conflict between TCNT Write and Increment.....	582
14.8.4	Conflict between TCOR Write and Compare Match.....	582

14.8.5	Conflict between Compare Matches A and B.....	583
14.8.6	Switching of Internal Clocks and TCNT Operation .....	583
14.8.7	Mode Setting with Cascaded Connection .....	585
14.8.8	Module Stop Function Setting .....	585
14.8.9	Interrupts in Module Stop State .....	585
<b>Section 15 Watchdog Timer (WDT) .....</b>		<b>587</b>
15.1	Features.....	587
15.2	Input/Output Pin .....	588
15.3	Register Descriptions.....	589
15.3.1	Timer Counter (TCNT).....	589
15.3.2	Timer Control/Status Register (TCSR).....	589
15.3.3	Reset Control/Status Register (RSTCSR).....	591
15.4	Operation .....	592
15.4.1	Watchdog Timer Mode.....	592
15.4.2	Interval Timer Mode.....	594
15.5	Interrupt Source .....	594
15.6	Usage Notes.....	595
15.6.1	Notes on Register Access .....	595
15.6.2	Conflict between Timer Counter (TCNT) Write and Increment.....	596
15.6.3	Changing Values of Bits CKS2 to CKS0.....	596
15.6.4	Switching between Watchdog Timer Mode and Interval Timer Mode.....	596
15.6.5	Internal Reset in Watchdog Timer Mode.....	597
15.6.6	System Reset by WDTOVF Signal.....	597
15.6.7	Transition to Watchdog Timer Mode or Software Standby Mode.....	597
<b>Section 16 Serial Communication Interface (SCI).....</b>		<b>599</b>
16.1	Features.....	599
16.2	Input/Output Pins.....	601
16.3	Register Descriptions.....	602
16.3.1	Receive Shift Register (RSR) .....	604
16.3.2	Receive Data Register (RDR).....	604
16.3.3	Transmit Data Register (TDR).....	604
16.3.4	Transmit Shift Register (TSR).....	605
16.3.5	Serial Mode Register (SMR) .....	605
16.3.6	Serial Control Register (SCR) .....	608
16.3.7	Serial Status Register (SSR) .....	614
16.3.8	Smart Card Mode Register (SCMR).....	622
16.3.9	Bit Rate Register (BRR) .....	623
16.3.10	Serial Extended Mode Register_2 (SEMR_2).....	633

16.4	Operation in Asynchronous Mode .....	635
16.4.1	Data Transfer Format.....	636
16.4.2	Receive Data Sampling Timing and Reception Margin in Asynchronous Mode .....	637
16.4.3	Clock.....	638
16.4.4	SCI Initialization (Asynchronous Mode).....	639
16.4.5	Serial Data Transmission (Asynchronous Mode) .....	640
16.4.6	Serial Data Reception (Asynchronous Mode).....	642
16.5	Multiprocessor Communication Function.....	646
16.5.1	Multiprocessor Serial Data Transmission .....	648
16.5.2	Multiprocessor Serial Data Reception .....	649
16.6	Operation in Clocked Synchronous Mode .....	652
16.6.1	Clock.....	652
16.6.2	SCI Initialization (Clocked Synchronous Mode).....	653
16.6.3	Serial Data Transmission (Clocked Synchronous Mode) .....	654
16.6.4	Serial Data Reception (Clocked Synchronous Mode).....	656
16.6.5	Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode).....	658
16.7	Operation in Smart Card Interface Mode.....	660
16.7.1	Sample Connection .....	660
16.7.2	Data Format (Except in Block Transfer Mode) .....	661
16.7.3	Block Transfer Mode .....	662
16.7.4	Receive Data Sampling Timing and Reception Margin.....	663
16.7.5	Initialization .....	664
16.7.6	Data Transmission (Except in Block Transfer Mode) .....	665
16.7.7	Serial Data Reception (Except in Block Transfer Mode).....	668
16.7.8	Clock Output Control.....	669
16.8	Interrupt Sources.....	671
16.8.1	Interrupts in Normal Serial Communication Interface Mode .....	671
16.8.2	Interrupts in Smart Card Interface Mode .....	672
16.9	Usage Notes .....	673
16.9.1	Module Stop Function Setting .....	673
16.9.2	Break Detection and Processing .....	673
16.9.3	Mark State and Break Detection .....	673
16.9.4	Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only).....	673
16.9.5	Relation between Writing to TDR and TDRE Flag .....	674
16.9.6	Restrictions on Using DTC or DMAC.....	674
16.9.7	SCI Operations during Mode Transitions .....	675

Section 17	I <sup>2</sup> C Bus Interface 2 (IIC2).....	679
17.1	Features.....	679
17.2	Input/Output Pins.....	681
17.3	Register Descriptions.....	682
17.3.1	I <sup>2</sup> C Bus Control Register A (IC CRA).....	683
17.3.2	I <sup>2</sup> C Bus Control Register B (IC CRB).....	684
17.3.3	I <sup>2</sup> C Bus Mode Register (IC MR).....	686
17.3.4	I <sup>2</sup> C Bus Interrupt Enable Register (IC IER).....	688
17.3.5	I <sup>2</sup> C Bus Status Register (IC SR).....	690
17.3.6	Slave Address Register (SAR).....	693
17.3.7	I <sup>2</sup> C Bus Transmit Data Register (IC DRT).....	694
17.3.8	I <sup>2</sup> C Bus Receive Data Register (IC DRR).....	694
17.3.9	I <sup>2</sup> C Bus Shift Register (IC DR).....	694
17.4	Operation.....	695
17.4.1	I <sup>2</sup> C Bus Format.....	695
17.4.2	Master Transmit Operation.....	696
17.4.3	Master Receive Operation.....	698
17.4.4	Slave Transmit Operation.....	700
17.4.5	Slave Receive Operation.....	703
17.4.6	Noise Canceler.....	704
17.4.7	Example of Use.....	705
17.5	Interrupt Request.....	709
17.6	Bit Synchronous Circuit.....	710
17.7	Usage Notes.....	711
17.7.1	Module Stop Function Setting.....	711
17.7.2	Issuance of Stop Condition and Repeated Start Condition.....	711
17.7.3	WAIT Bit.....	712
17.7.4	Restriction on Transfer Rate Setting Value in Multi-Master Mode.....	712
17.7.5	Restriction on Bit Manipulation when Setting the MST and TRS Bits in Multi-Master Mode.....	712
17.7.6	Notes on Master Receive Mode.....	712
Section 18	A/D Converter.....	713
18.1	Features.....	713
18.2	Input/Output Pins.....	715
18.3	Register Descriptions.....	715
18.3.1	A/D Data Registers A to H (ADDRA to ADDRH).....	716
18.3.2	A/D Control/Status Register (ADCSR).....	717
18.3.3	A/D Control Register (ADCR).....	719
18.4	Operation.....	721

18.4.1	Single Mode.....	721
18.4.2	Scan Mode .....	723
18.4.3	Input Sampling and A/D Conversion Time .....	725
18.4.4	External Trigger Input Timing.....	727
18.5	Interrupt Source .....	728
18.6	A/D Conversion Accuracy Definitions .....	728
18.7	Usage Notes .....	730
18.7.1	Module Stop Function Setting .....	730
18.7.2	A/D Input Hold Function in Software Standby Mode .....	730
18.7.3	Permissible Signal Source Impedance .....	730
18.7.4	Influences on Absolute Accuracy .....	731
18.7.5	Setting Range of Analog Power Supply and Other Pins .....	731
18.7.6	Notes on Board Design .....	732
18.7.7	Notes on Countermeasure against Noise.....	732
<b>Section 19 <math>\Delta\Sigma</math> A/D Converter.....</b>		<b>735</b>
19.1	Features.....	735
19.2	Input/Output Pins.....	737
19.3	Register Descriptions .....	738
19.3.1	$\Delta\Sigma$ A/D Mode Register (DSADMR).....	739
19.3.2	$\Delta\Sigma$ A/D Data Registers 0 to 5 (DSADDR0 to DSADDR5) .....	740
19.3.3	$\Delta\Sigma$ A/D Control/Status Register (DSADCSR).....	740
19.3.4	$\Delta\Sigma$ A/D Control Register (DSADCR).....	743
19.3.5	$\Delta\Sigma$ A/D Offset Cancel DAC Inputs 0 to 3 (DSADOF0 to DSADOF3).....	745
19.4	Operation .....	746
19.4.1	Procedure for Activating the $\Delta\Sigma$ A/D Converter .....	747
19.4.2	Selecting Analog Input Channels.....	748
19.4.3	Single Mode.....	749
19.4.4	Scan Mode .....	752
19.4.5	Flow of $\Delta\Sigma$ A/D Conversion Operation .....	754
19.4.6	Analog Input Sampling and A/D Conversion Time.....	756
19.4.7	External Trigger Input Timing.....	759
19.5	Interrupt Source .....	760
19.6	Usage Notes .....	761
19.6.1	Module Stop Function Setting .....	761
19.6.2	Settings for the Biasing Circuit.....	761
19.6.3	State of the $\Delta\Sigma$ A/D Converter in Software Standby Mode .....	762
19.6.4	Changing the Settings of $\Delta\Sigma$ A/D Converter Registers.....	762
19.6.5	DSE Bit.....	762

Section 20	D/A Converter .....	763
20.1	Features.....	763
20.2	Input/Output Pins.....	764
20.3	Register Descriptions.....	764
20.3.1	D/A Data Registers 0 and 1 (DADR0 and DADR1).....	764
20.3.2	D/A Control Register 01 (DACR01) .....	765
20.4	Operation .....	767
20.5	Usage Notes.....	768
20.5.1	Module Stop Function Setting .....	768
20.5.2	D/A Output Hold Function in Software Standby Mode.....	768
Section 21	RAM.....	769
Section 22	Flash Memory.....	771
22.1	Features.....	771
22.2	Mode Transition Diagram.....	773
22.3	Memory MAT Configuration .....	775
22.4	Block Structure .....	776
22.5	Programming/Erasing Interface .....	777
22.6	Input/Output Pins.....	779
22.7	Register Descriptions.....	779
22.7.1	Programming/Erasing Interface Registers .....	780
22.7.2	Programming/Erasing Interface Parameters .....	787
22.7.3	RAM Emulation Register (RAMER).....	799
22.8	On-Board Programming Mode .....	800
22.8.1	Boot Mode .....	800
22.8.2	User Program Mode.....	804
22.8.3	User Boot Mode.....	814
22.8.4	On-Chip Program and Storable Area for Program Data .....	818
22.9	Protection.....	824
22.9.1	Hardware Protection .....	824
22.9.2	Software Protection.....	825
22.9.3	Error Protection .....	825
22.10	Flash Memory Emulation Using RAM.....	827
22.11	Switching between User MAT and User Boot MAT.....	830
22.12	Programmer Mode .....	831
22.13	Standard Serial Communication Interface Specifications for Boot Mode .....	831
22.14	Usage Notes.....	860



Section 23	Clock Pulse Generator .....	863
23.1	Register Description .....	865
23.1.1	System Clock Control Register (SCKCR) .....	865
23.1.2	$\Delta\Sigma$ A/D Mode Register (DSADMR).....	868
23.2	Oscillator.....	869
23.2.1	Connecting Crystal Resonator .....	869
23.2.2	External Clock Input.....	870
23.3	PLL Circuit .....	871
23.4	Frequency Divider .....	871
23.4.1	1 $\phi$ , B $\phi$ , P $\phi$ Frequency Dividers.....	871
23.4.2	A $\phi$ Frequency Divider .....	871
23.5	Usage Notes .....	872
23.5.1	Notes on Clock Pulse Generator .....	872
23.5.2	Notes on Resonator .....	873
23.5.3	Notes on Board Design .....	873
Section 24	Power-Down Modes .....	875
24.1	Features.....	875
24.2	Register Descriptions .....	878
24.2.1	Standby Control Register (SBYCR) .....	878
24.2.2	Module Stop Control Registers A and B (MSTPCRA and MSTPCRB) .....	881
24.2.3	Module Stop Control Register C (MSTPCRC).....	884
24.2.4	Deep Standby Control Register (DPSBYCR).....	885
24.2.5	Deep Standby Wait Control Register (DPSWCR).....	888
24.2.6	Deep Standby Interrupt Enable Register (DPSIER) .....	890
24.2.7	Deep Standby Interrupt Flag Register (DPSIFR).....	891
24.2.8	Deep Standby Interrupt Edge Register (DPSIEGR) .....	893
24.2.9	Reset Status Register (RSTSR).....	894
24.2.10	Deep Standby Backup Register (DPSBKRn) .....	895
24.3	Multi-Clock Function .....	895
24.4	Module Stop State.....	895
24.5	Sleep Mode .....	896
24.5.1	Entry to Sleep Mode .....	896
24.5.2	Exit from Sleep Mode.....	896
24.6	All-Module-Clock-Stop Mode.....	897
24.7	Software Standby Mode.....	898
24.7.1	Entry to Software Standby Mode.....	898
24.7.2	Exit from Software Standby Mode .....	898
24.7.3	Setting Oscillation Settling Time after Exit from Software Standby Mode.....	899

24.7.4	Software Standby Mode Application Example .....	901
24.8	Deep Software Standby Mode .....	902
24.8.1	Entry to Deep Software Standby Mode .....	902
24.8.2	Exit from Deep Software Standby Mode .....	903
24.8.3	Pin State on Exit from Deep Software Standby Mode .....	904
24.8.4	B $\phi$ Operation after Exit from Deep Software Standby Mode .....	904
24.8.5	Setting Oscillation Settling Time after Exit from Deep Software Standby Mode .....	905
24.8.6	Deep Software Standby Mode Application Example .....	907
24.8.7	Flowchart of Deep Software Standby Mode Operation .....	911
24.9	Hardware Standby Mode .....	913
24.9.1	Transition to Hardware Standby Mode .....	913
24.9.2	Clearing Hardware Standby Mode .....	913
24.9.3	Hardware Standby Mode Timing .....	913
24.9.4	Timing Sequence at Power-On .....	914
24.10	Sleep Instruction Exception Handling .....	915
24.11	B $\phi$ Clock Output Control .....	918
24.12	Usage Notes .....	919
24.12.1	I/O Port Status .....	919
24.12.2	Current Consumption during Oscillation Settling Standby Period .....	919
24.12.3	Module Stop State of DMAC or DTC .....	919
24.12.4	On-Chip Peripheral Module Interrupts .....	919
24.12.5	Writing to MSTPCRA, MSTPCRB, and MSTPCRC .....	919
24.12.6	Control of Input Buffers by DIRQnE (n = 3 to 0) .....	920
24.12.7	Input Buffer Control by DIRQnE (n = 3 to 0) .....	920
24.12.8	B $\phi$ Output State .....	920
<b>Section 25 List of Registers .....</b>		<b>921</b>
25.1	Register Addresses (Address Order) .....	922
25.2	Register Bits .....	935
25.3	Register States in Each Operating Mode .....	954
<b>Section 26 Electrical Characteristics .....</b>		<b>967</b>
26.1	Electrical Characteristics .....	967
26.1.1	Absolute Maximum Ratings .....	967
26.1.2	DC Characteristics .....	968
26.1.3	AC Characteristics .....	973
26.1.4	A/D Conversion Characteristics .....	980
26.1.5	D/A Conversion Characteristics .....	980
26.1.6	$\Delta\Sigma$ A/D Conversion Characteristics .....	981

26.2	Timing Charts .....	986
26.3	Flash Memory Characteristics .....	1006
Appendix.....		1007
A.	Port States in Each Pin State.....	1007
B.	Product Lineup.....	1012
C.	Package Dimensions .....	1013
D.	Treatment of Unused Pins.....	1015
E.	Example of an External Circuit of $\Delta\Sigma$ A/D Converter.....	1018
Index .....		1019



# Section 1 Overview

## 1.1 Features

The core of each product in the H8SX/1622 Group of CISC (complex instruction set computer) microcomputers is an H8SX CPU, which has an internal 32-bit architecture. The H8SX CPU provides upward-compatibility with the CPUs of other Renesas Technology-original microcomputers; H8/300, H8/300H, and H8S.

As peripheral functions, each LSI of the Group includes a DMA controller, which enables high-speed data transfer, and a bus-state controller, which enables direct connection to different kinds of memory. The LSI of the Group also includes a  $\Delta\Sigma$  A/D converter specialized for sensor control, D/A converter, serial communication interfaces, and a multi-function timer that makes motor control easy. Together, the modules realize low-cost configurations for end systems. The power consumption of these modules are kept down dynamically by an on-chip power-management function.

### 1.1.1 Applications

Example of application: Consumer appliances

### 1.1.2 Overview of Functions

Table 1.1 gives an overview of the functions of the H8SX/1622 Group products.

**Table 1.1 Overview of Functions**

Classification	Module/ Function	Description
Memory	ROM	<ul style="list-style-type: none"> <li>ROM capacity: 256 Kbytes</li> </ul>
	RAM	<ul style="list-style-type: none"> <li>RAM capacity: 24 Kbytes</li> </ul>
CPU	CPU	<ul style="list-style-type: none"> <li>32-bit high-speed H8SX CPU (CISC type)</li> <li>Upward compatibility for H8/300, H8/300H, and H8S CPUs at object level</li> <li>Sixteen 16-bit general registers</li> <li>Eleven addressing modes</li> <li>4-Gbyte address space</li> <li>Program: 4 Gbytes available</li> <li>Data: 4 Gbytes available</li> <li>87 basic instructions, classifiable as bit arithmetic and logic instructions, multiply and divide instructions, bit manipulation instructions, multiply-and-accumulate instructions, and others</li> <li>Minimum instruction execution time: 20.0 ns (for an ADD instruction when running with system clock <math>f_{\phi} = 50</math> MHz and <math>V_{cc} = 3.0</math> to 3.6 V)</li> <li>On-chip multiplier (<math>16 \times 16 \rightarrow 32</math> bits)</li> <li>Supports multiply-and-accumulate instructions (<math>16 \times 16 + 32 \rightarrow 32</math> bits)</li> </ul>
		Operating mode

Classification	Module/ Function	Description
CPU	MCU operating mode	<p>Mode 1: User boot mode (selected by driving the MD2 and MD1 pins low and driving the MD0 pin high)</p> <p>Mode 2: Boot mode (selected by driving the MD2 and MD0 pins low and driving the MD1 pin high)</p> <p>Mode 4: On-chip ROM disabled external extended mode, 16-bit bus (selected by driving the MD1 and MD0 pins low and driving the MD2 pin high)</p> <p>Mode 5: On-chip ROM disabled external extended mode, 8-bit bus (selected by driving the MD1 pin low and driving the MD2 and MD0 pins high)</p> <p>Mode 6: On-chip ROM enabled external extended mode (selected by driving the MD0 pin low and driving the MD2 and MD1 pins high)</p> <p>Mode 7: Single-chip mode (can be externally extended) (selected by driving the MD2, MD1, and MD0 pins high)</p> <ul style="list-style-type: none"> <li>• Low power consumption state (transition driven by the SLEEP instruction)</li> </ul>
Interrupt (sources)	Interrupt controller (INTC)	<ul style="list-style-type: none"> <li>• Seventeen external interrupt pins (NMI, and <math>\overline{IRQ15}</math> to <math>\overline{IRQ0}</math>)</li> <li>• 80 internal interrupt sources</li> <li>• Two interrupt control modes (specified by the interrupt control register)</li> <li>• Eight priority orders specifiable (by setting the interrupt priority register)</li> <li>• Independent vector addresses</li> </ul>
	Break interrupt (UBC)	<ul style="list-style-type: none"> <li>• Break points on four channels</li> <li>• Address break can be set at the CPU instruction fetch cycle</li> </ul>

Classification	Module/ Function	Description
DMA	DMA controller (DMAC)	<ul style="list-style-type: none"> <li>• Two-channel DMA transfer available</li> <li>• Three activation methods (auto-request, on-chip module interrupt, external request)</li> <li>• Three transfer modes (normal transfer, repeat transfer, block transfer)</li> <li>• Dual or single address mode selectable</li> <li>• Extended repeat-area function</li> </ul>
	Data transfer controller (DTC)	<ul style="list-style-type: none"> <li>• Allows DMA transfer over 55 channels (number of DTC activation sources)</li> <li>• Activated by interrupt sources (chain transfer enabled)</li> <li>• Three transfer modes (normal transfer, repeat transfer, block transfer)</li> <li>• Short-address mode or full-address mode selectable</li> </ul>
External bus extension	Bus controller (BSC)	<ul style="list-style-type: none"> <li>• 16-Mbyte external address space</li> <li>• The external address space can be divided into eight areas, each of which is independently controllable <ul style="list-style-type: none"> <li>— Chip-select signals (<math>\overline{CS0}</math> to <math>\overline{CS7}</math>) can be output</li> <li>— Access in two or three states can be selected for each area</li> <li>— Program wait cycles can be inserted</li> <li>— The period of <math>\overline{CS}</math> assertion can be extended</li> <li>— Idle cycles can be inserted</li> </ul> </li> <li>• Bus arbitration function (arbitrates bus mastership among the internal CPU, DMAC and DTC, and external bus masters)</li> </ul> <hr/> <p>Bus formats</p> <ul style="list-style-type: none"> <li>• External memory interfaces (for the connection of ROM, burst ROM, SRAM, and byte control SRAM)</li> <li>• Address/data bus format: Support for both separate and multiplexed buses (8-bit access or 16-bit access)</li> </ul> <hr/> <ul style="list-style-type: none"> <li>• Endian conversion function for connecting devices in little-endian format</li> </ul>



Classification	Module/ Function	Description
Clock	Clock pulse generator (CPG)	<ul style="list-style-type: none"> <li>• One clock generation circuit available</li> <li>• Separate clock signals are provided for each of functional modules (detailed below) and each is independently specifiable (multi-clock function) <ul style="list-style-type: none"> <li>— System-intended data transfer modules, i.e. the CPU, are run by the system clock (<math>I\phi</math>): 8 to 50 MHz</li> <li>— On-chip peripheral functions are run by the peripheral module clock (<math>P\phi</math>): 8 to 35 MHz</li> <li>— External space modules are supplied with the external bus clock (<math>B\phi</math>): 8 to 50 MHz</li> <li>— <math>\Delta\Sigma</math> A/D converter is run by the clock for <math>\Delta\Sigma</math> A/D converter (<math>A\phi</math>): near 25 MHz</li> </ul> </li> <li>• Includes a PLL frequency multiplier and frequency dividers (including a divider for <math>A\phi</math>), so the operating frequency is selectable</li> <li>• Five power-down modes: Sleep mode, all-module-clock-stop mode, software standby mode, deep software standby mode, and hardware standby mode</li> </ul>
A/D converter	10-bit A/D converter (ADC)	<ul style="list-style-type: none"> <li>• 10-bit resolution <math>\times</math> eight input channels</li> <li>• Sample and hold function included</li> <li>• Conversion time: 5.33 <math>\mu</math>s per channel (with ADCLK at 7.5 MHz operation)</li> <li>• Two operating modes: single mode and scan mode</li> <li>• Three ways to start A/D conversion: by software, timer (TPU/TMR) trigger, and external trigger (Starting by TPU/TMR: This operation is available on the on-chip emulator but not available on other emulators.)</li> </ul>
	16-bit $\Delta\Sigma$ A/D converter ( $\Delta\Sigma$ AD)	<ul style="list-style-type: none"> <li>• 16-bit resolution</li> <li>• Six input channels (differential inputs on two channels)</li> <li>• Conversion time: 286 states</li> <li>• Two operating modes: single mode and scan mode</li> <li>• <math>\Delta\Sigma</math> modulation</li> <li>• Three ways to start A/D conversion: by software, timer (TPU/TMR) trigger, and external trigger (Starting by TPU/TMR: This operation is available on the on-chip emulator but not available on other emulators.)</li> <li>• Input voltage offset cancellation in two modes: by register setting and by differential input</li> </ul>

Classification	Module/ Function	Description
D/A converter	D/A converter (DAC)	<ul style="list-style-type: none"> <li>8-bit resolution × two output channels</li> <li>Output voltage: 0 V to Vref, maximum conversion time: 10 μs (with 20 pF load)</li> </ul>
Timer	8-bit timer (TMR)	<ul style="list-style-type: none"> <li>Eight channels of 8-bit timers (can be used as two channels of 16-bit timers)</li> <li>Select from among seven clock sources (six internal clocks and one external clock)</li> <li>Allows the output of pulse trains with a desired duty cycle or PWM signals</li> </ul>
	16-bit timer pulse unit (TPU)	<ul style="list-style-type: none"> <li>16 bits × six channels (general pulse timer unit)</li> <li>Select from among eight counter-input clocks for each channel</li> <li>Up to 16 pulse inputs and outputs</li> <li>Counter clear operation, simultaneous writing to multiple timer counters (TCNT), simultaneous clearing by compare match and input capture possible, simultaneous input/output for registers possible by counter synchronous operation, and up to 15-phase PWM output possible by combination with synchronous operation</li> <li>Buffered operation, cascaded operation (32 bits × two channels), and phase counting mode (two-phase encoder input) settable for each channel</li> <li>Input capture function supported</li> <li>Output compare function (by the output of compare match waveform) supported</li> </ul>
	Program-mable pulse generator (PPG)	<ul style="list-style-type: none"> <li>16-bit pulse output</li> <li>Four output groups, non-overlapping mode, and inverted output can be set</li> <li>Selectable output trigger signals; the PPG can operate in conjunction with the data transfer controller (DTC) and the DMA controller (DMAC)</li> </ul>
Watchdog timer	Watchdog timer (WDT)	<ul style="list-style-type: none"> <li>8 bits × one channel (selectable from eight counter input clocks)</li> <li>Switchable between watchdog timer mode and interval timer mode</li> </ul>
Serial interface	Serial communication interface (SCI)	<ul style="list-style-type: none"> <li>Five channels (select asynchronous or clocked synchronous serial communication mode)</li> <li>Full-duplex communication capability</li> <li>Select the desired bit rate and LSB-first or MSB-first transfer</li> </ul>
Smart card/ SIM		<ul style="list-style-type: none"> <li>The SCI module supports a smart card (SIM) interface.</li> </ul>

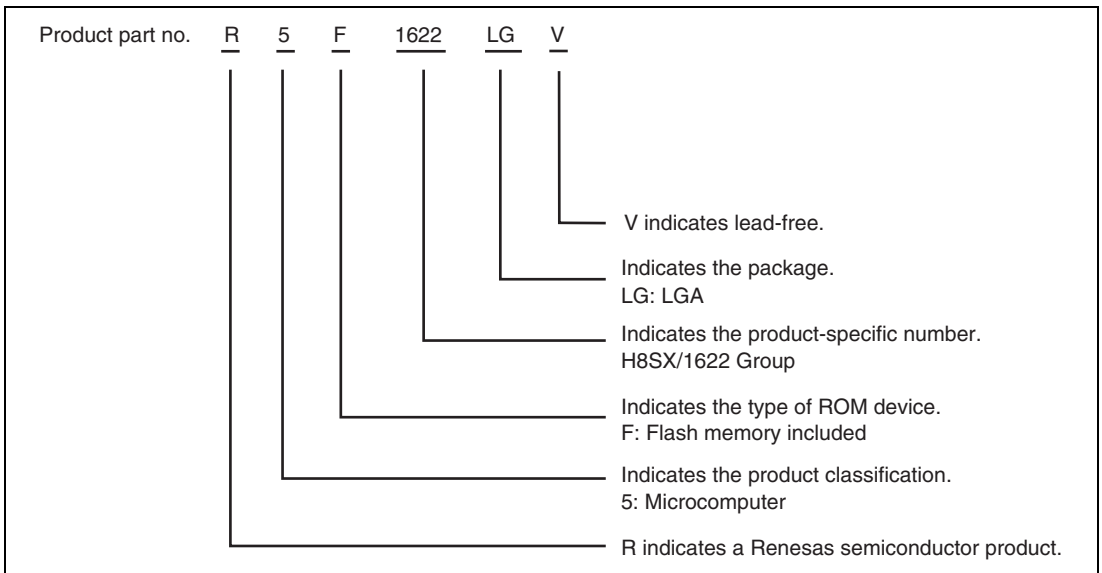
Classification	Module/ Function	Description
I <sup>2</sup> C bus interface	I <sup>2</sup> C bus interface 2 (IIC2)	<ul style="list-style-type: none"> <li>• Two channels</li> <li>• Bus can be directly driven (the SCL and SDL pins are NMOS open drains).</li> </ul>
I/O ports		<ul style="list-style-type: none"> <li>• 17 CMOS input-only pins</li> <li>• 74 CMOS input/output pins</li> <li>• Eight large-current drive pins (port 3)</li> <li>• 37 pull-up resistor-provided pins</li> <li>• 21 open-drain pins</li> </ul>
Package		<ul style="list-style-type: none"> <li>• LGA-145 package</li> <li>• LQFP-144 package</li> </ul>
Operating frequency/ Power supply voltage		<ul style="list-style-type: none"> <li>• Operating frequency: 8 to 50 MHz</li> <li>• Power supply voltage: V<sub>cc</sub> = PLLV<sub>cc</sub> = 3.0 to 3.6 V, AV<sub>cc</sub> = 3.0 to 3.6 V, AV<sub>ccP</sub> = AV<sub>ccA</sub> = AV<sub>ccD</sub> = 3.0 to 3.6 V</li> <li>• Flash program/erase voltage: 3.0 to 3.6 V</li> <li>• Supply current: <ul style="list-style-type: none"> <li>— 48 mA (typ.) (V<sub>cc</sub> = PLLV<sub>cc</sub> = 3.3 V, AV<sub>cc</sub> = 3.3 V, AV<sub>ccP</sub> = AV<sub>ccA</sub> = AV<sub>ccD</sub> = 3.3 V, I<sub>φ</sub> = B<sub>φ</sub> = 50 MHz, P<sub>φ</sub> = 25 MHz)</li> <li>— 46 mA (typ.) (V<sub>cc</sub> = PLLV<sub>cc</sub> = 3.3 V, AV<sub>cc</sub> = 3.3 V, AV<sub>ccP</sub> = AV<sub>ccA</sub> = AV<sub>ccD</sub> = 3.0 V, I<sub>φ</sub> = P<sub>φ</sub> = B<sub>φ</sub> = 35 MHz)</li> </ul> </li> </ul>
Operating ambient temperature (°C)		<ul style="list-style-type: none"> <li>• -20 to +75°C (regular specifications)</li> <li>• -40 to +85°C (wide-range specifications)</li> </ul>

## 1.2 List of Products

Table 1.2 is the list of products, and figure 1.1 shows how to read the product part No.

**Table 1.2 List of Products**

Product Part No.	ROM Capacity	RAM Capacity	Package	Remarks
R5F61622LGV	256 Kbytes	24 Kbytes	LGA-145	—
R5F61622FPV	256 Kbytes	24 Kbytes	LQFP-144	—



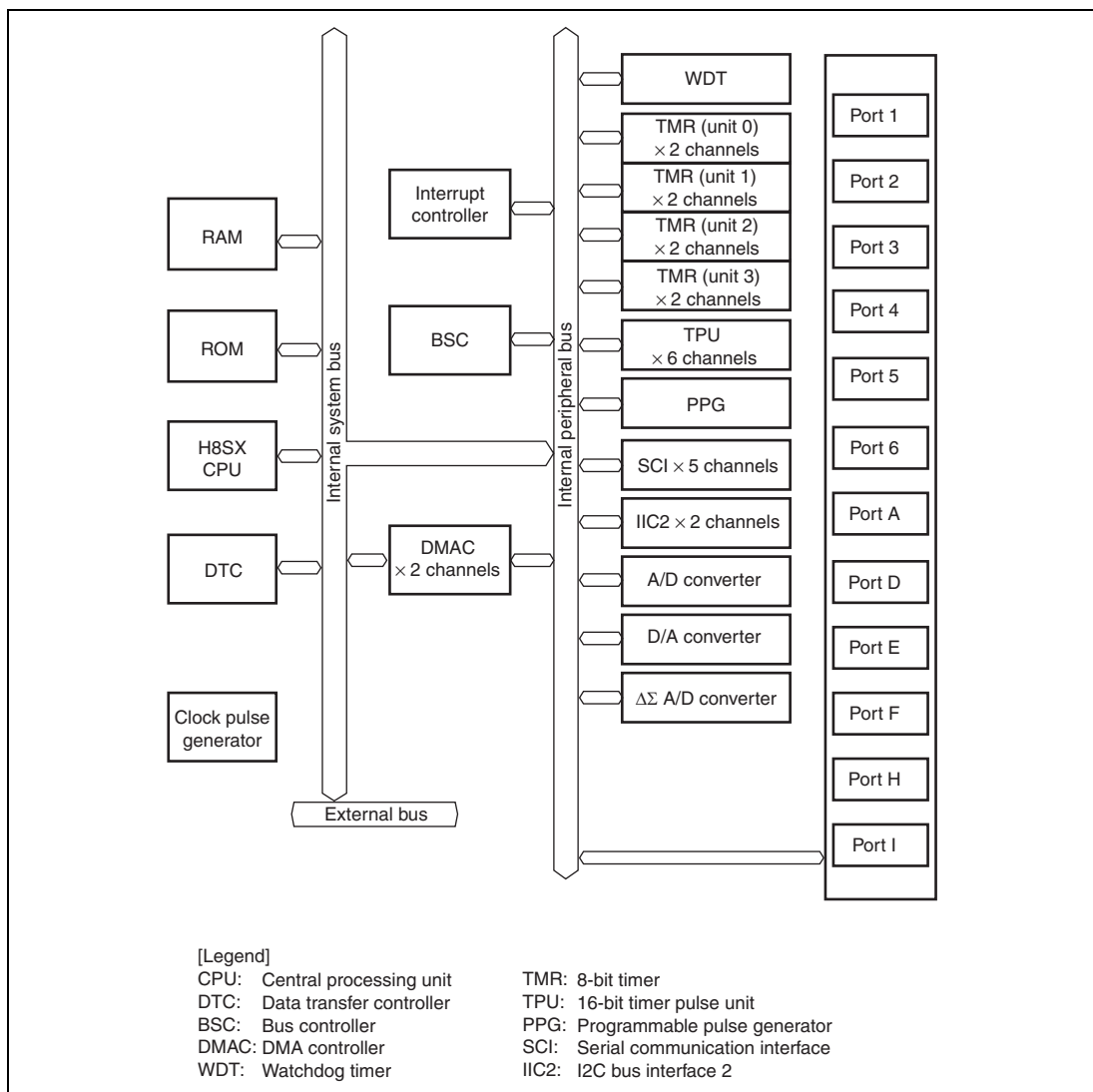
**Figure 1.1 How to Read the Product Part No.**

- Compact Package

Package	Code	Size	Pin Pitch
LGA-145	PTLG0145JB-A*	9.0 × 9.0 mm	0.65 mm
LQFP-144	PLQP0144KA-A*	20.0 × 20.0 mm	0.50 mm

Note: \* Lead-free version

## 1.3 Block Diagram



**Figure 1.2 Block Diagram**

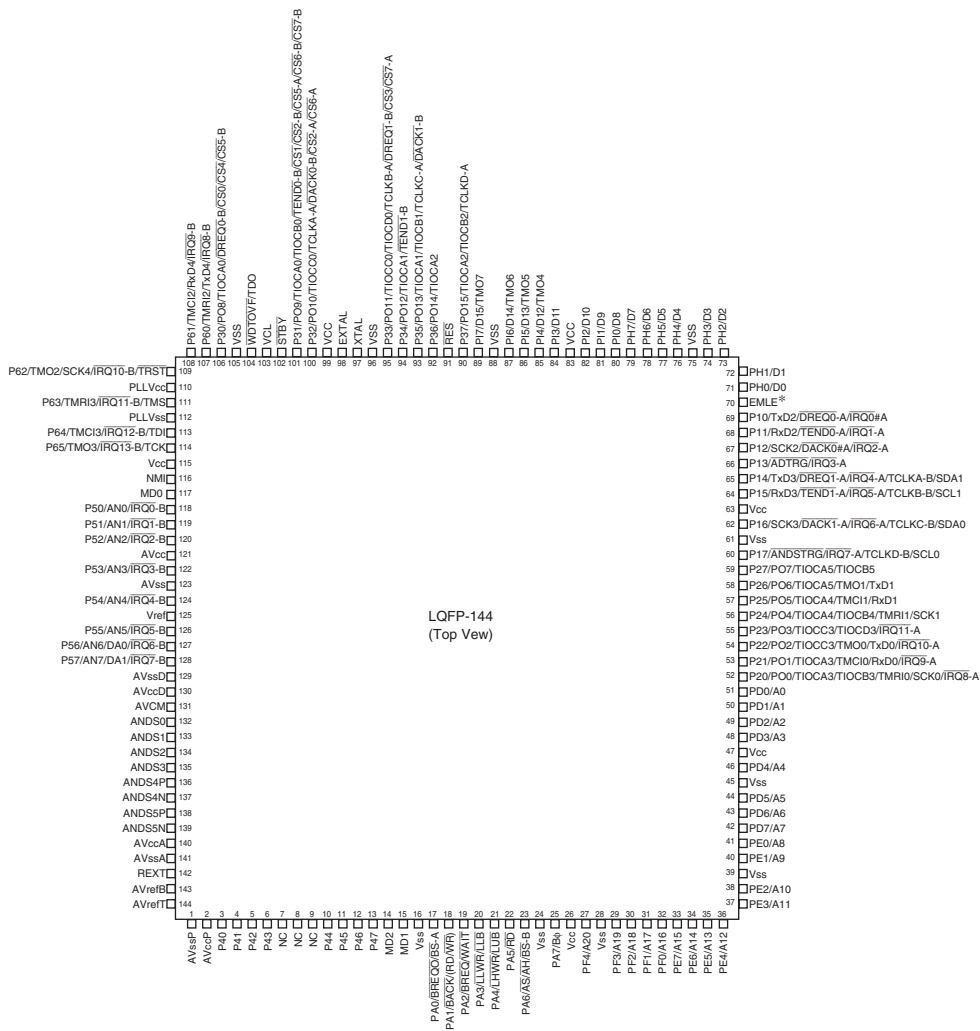
## 1.4 Pin Descriptions

### 1.4.1 Pin Assignments

	1	2	3	4	5	6	7	8	9	10	11	12	13
A	AVccP	AVrefT	AVrefB	AVccA	ANDS4N	ANDS1	AVssD	Vref	AVcc	MD0	P64	P62	PLLvcc
B	AVssP	P41	AVssA	ANDS5P	ANDS2	AVccD	P54	P52	P50	Vcc	P63	PLLvss	P61
C	P42	P40	P43	REXT	ANDS5N	ANDS3	AVCM	AVss	P51	NMI	P65	Vss	P30
D	NC	NC	NC	ANDS4P	ANDS0	P57	P56	P55	P53	WDTOVF	P60	P31	VCL
E	P47	P44	P45	MD2	NC	LGA (Top view)				Vss	$\overline{\text{STBY}}$	EXTAL	Vcc
F	PA0	P46	MD1	PA1	P33					P32	P36	XTAL	
G	PA4	Vss	PA6	PA2	P37					P34	PI7	P35	
H	PA7	PA5	PF4	PA3	PI5					Vss	PI4	$\overline{\text{RES}}$	
J	PF3	Vcc	PF1	Vss	Vcc					PI3	PI1	PI6	
K	PF0	PF2	PE6	Vss	PD1	P22	P23	P24	P17	P15	PI0	PH7	PI2
L	PE5	PE7	PD7	PD5	Vcc	PD0	P27	Vcc	P12	EMLE*	PH5	Vss	PH6
M	PE4	PE2	Vss	PD6	PD4	PD3	P20	P26	P16	P13	P10	PH4	PH2
N	PE1	PE3	PE0	Vss	PD2	P21	P25	Vss	P14	P11	PH0	PH1	PH3

Note: \* This pin is an on-chip emulator enable pin. Drive this pin low for the connection in normal operating mode. The on-chip emulator function is enabled by driving this pin high. When the on-chip emulator is in use, the P62, P63, P64, P65, and WDTOVF pins are dedicated pins for the on-chip emulator. For details on a connection example with the E10A, see E10A Emulator User's Manual.

**Figure 1.3 Pin Assignments (LGA-145)**



Note: \* This pin is an on-chip emulator enable pin. Drive this pin low for the connection in normal operating mode. The on-chip emulator function is enabled by driving this pin high. When the on-chip emulator is in use, the P62, P63, P64, P65, and WDT0VF pins are dedicated pins for the on-chip emulator. For details on a connection example with the E10A, see E10A Emulator User's Manual.

Figure 1.4 Pin Assignments (LQFP-144)

## 1.4.2 Pin Assignment for Each Operating Mode

**Table 1.3 Pin Assignment for Each Operating Mode**

Pin No.		Pin Name	
LQFP	LGA	Modes 1, 2, 6, 7	Modes 4 and 5
1	B1	AV <sub>ss</sub> P	AV <sub>ss</sub> P
2	A1	AV <sub>cc</sub> P	AV <sub>cc</sub> P
3	C2	P40	P40
4	B2	P41	P41
5	C1	P42	P42
6	C3	P43	P43
7	D2	NC	NC
8	D3	NC	NC
9	D1	NC	NC
10	E2	P44	P44
11	E3	P45	P45
12	F2	P46	P46
13	E1	P47	P47
14	E4	MD2	MD2
15	F3	MD1	MD1
16	G2	V <sub>ss</sub>	V <sub>ss</sub>
17	F1	PA0/BREQO/BS-A	PA0/BREQO/BS-A
18	F4	PA1/BACK/ (RD/WR)	PA1/BACK/ (RD/WR)
19	G4	PA2/BREQ/WAIT	PA2/BREQ/WAIT
20	H4	PA3/LLWR/LLB	PA3/LLWR/LLB
21	G1	PA4/LHWR/LUB	PA4/LHWR/LUB
22	H2	PA5/RD	PA5/RD
23	G3	PA6/AS/AH/BS-B	PA6/AS/AH/BS-B
24	J4	V <sub>ss</sub>	V <sub>ss</sub>
25	H1	PA7/Bφ	PA7/Bφ
26	J2	V <sub>cc</sub>	V <sub>cc</sub>



Pin No.			Pin Name
LQFP	LGA	Modes 1, 2, 6, 7	Modes 4 and 5
27	H3	PF4/A20	A20
28	K4	V <sub>ss</sub>	V <sub>ss</sub>
29	J1	PF3/A19	A19
30	K2	PF2/A18	A18
31	J3	PF1/A17	A17
32	K1	PF0/A16	A16
33	L2	PE7/A15	A15
34	K3	PE6/A14	A14
35	L1	PE5/A13	A13
36	M1	PE4/A12	A12
37	N2	PE3/A11	A11
38	M2	PE2/A10	A10
39	M3	V <sub>ss</sub>	V <sub>ss</sub>
40	N1	PE1/A9	A9
41	N3	PE0/A8	A8
42	L3	PD7/A7	A7
43	M4	PD6/A6	A6
44	L4	PD5/A5	A5
45	N4	V <sub>ss</sub>	V <sub>ss</sub>
46	M5	PD4/A4	A4
47	L5	V <sub>cc</sub>	V <sub>cc</sub>
48	M6	PD3/A3	A3
49	N5	PD2/A2	A2
50	K5	PD1/A1	A1
51	L6	PD0/A0	A0
52	M7	P20/PO0/TIOCA3/TIOCB3/TMRI0/SCK0/ IRQ8-A	P20/PO0/TIOCA3/TIOCB3/TMRI0/SCK0/ IRQ8-A
53	N6	P21/PO1/TIOCA3/TMCI0/RxD0/IRQ9-A	P21/PO1/TIOCA3/TMCI0/RxD0/IRQ9-A
54	K6	P22/PO2/TIOCC3/TMO0/TxD0/IRQ10-A	P22/PO2/TIOCC3/TMO0/TxD0/IRQ10-A
55	K7	P23/PO3/TIOCC3/TIOCD3/IRQ11-A	P23/PO3/TIOCC3/TIOCD3/IRQ11-A

Pin No.		Pin Name	
LQFP	LGA	Modes 1, 2, 6, 7	Modes 4 and 5
56	K8	P24/PO4/TIOCA4/TIOCB4/TMRI1/SCK1	P24/PO4/TIOCA4/TIOCB4/TMRI1/SCK1
57	N7	P25/PO5/TIOCA4/TMC11/RxD1	P25/PO5/TIOCA4/TMC11/RxD1
58	M8	P26/PO6/TIOCA5/TMO1/TxD1	P26/PO6/TIOCA5/TMO1/TxD1
59	L7	P27/PO7/TIOCA5/TIOCB5/ $\overline{\text{IRQ15}}$	P27/PO7/TIOCA5/TIOCB5/ $\overline{\text{IRQ15}}$
60	K9	P17/ $\overline{\text{ANDSTRG}}$ / $\overline{\text{IRQ7-A}}$ /TCLKD-B/SCL0	P17/ $\overline{\text{ANDSTRG}}$ / $\overline{\text{IRQ7-A}}$ /TCLKD-B/SCL0
61	N8	$V_{\text{SS}}$	$V_{\text{SS}}$
62	M9	P16/SCK3/ $\overline{\text{DACK1-A}}$ / $\overline{\text{IRQ6-A}}$ /TCLKC-B/SDA0	P16/SCK3/ $\overline{\text{DACK1-A}}$ / $\overline{\text{IRQ6-A}}$ /TCLKC-B/SDA0
63	L8	$V_{\text{CC}}$	$V_{\text{CC}}$
64	K10	P15/RxD3/ $\overline{\text{TEND1-A}}$ / $\overline{\text{IRQ5-A}}$ /TCLKB-B/SCL1	P15/RxD3/ $\overline{\text{TEND1-A}}$ / $\overline{\text{IRQ5-A}}$ /TCLKB-B/SCL1
65	N9	P14/TxD3/ $\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ /TCLKA-B/SDA1	P14/TxD3/ $\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ /TCLKA-B/SDA1
66	M10	P13/ $\overline{\text{ADTRG0}}$ / $\overline{\text{IRQ3-A}}$	P13/ $\overline{\text{ADTRG0}}$ / $\overline{\text{IRQ3-A}}$
67	L9	P12/SCK2/ $\overline{\text{DACK0-A}}$ / $\overline{\text{IRQ2-A}}$	P12/SCK2/ $\overline{\text{DACK0-A}}$ / $\overline{\text{IRQ2-A}}$
68	N10	P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}$	P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}$
69	M11	P10/TxD2/ $\overline{\text{DREQ0-A}}$ / $\overline{\text{IRQ0-A}}$	P10/TxD2/ $\overline{\text{DREQ0-A}}$ / $\overline{\text{IRQ0-A}}$
70	L10	EMLE	EMLE
71	N11	PH0/D0	PH0/D0
72	N12	PH1/D1	PH1/D1
73	M13	PH2/D2	PH2/D2
74	N13	PH3/D3	PH3/D3
75	L12	$V_{\text{SS}}$	$V_{\text{SS}}$
76	M12	PH4/D4	PH4/D4
77	L11	PH5/D5	PH5/D5
78	L13	PH6/D6	PH6/D6
79	K12	PH7/D7	PH7/D7
80	K11	PI0/D8	PI0/D8
81	J12	PI1/D9	PI1/D9
82	K13	PI2/D10	PI2/D10
83	J10	$V_{\text{CC}}$	$V_{\text{CC}}$
84	J11	PI3/D11	PI3/D11

Pin No.		Pin Name	
LQFP	LGA	Modes 1, 2, 6, 7	Modes 4 and 5
85	H12	PI4/D12/TMO4	PI4/D12/TMO4
86	H10	PI5/D13/TMO5	PI5/D13/TMO5
87	J13	PI6/D14/TMO6	PI6/D14/TMO6
88	H11	V <sub>ss</sub>	V <sub>ss</sub>
89	G12	PI7/D15/TMO7	PI7/D15/TMO7
90	G10	P37/PO15/TIOCA2/TIOCB2/TCLKD-A	P37/PO15/TIOCA2/TIOCB2/TCLKD-A
91	H13	$\overline{\text{RES}}$	$\overline{\text{RES}}$
92	F12	P36/PO14/TIOCA2	P36/PO14/TIOCA2
93	G13	P35/PO13/TIOCA1/TIOCB1/TCLKC-A/ DACK1-B	P35/PO13/TIOCA1/TIOCB1/TCLKC-A/ DACK1-B
94	G11	P34/PO12/TIOCA1/TEND1-B	P34/PO12/TIOCA1/TEND1-B
95	F10	P33/PO11/TIOCC0/TIOCD0/TCLKB-A/ DREQ1-B/CS3/CS7-A	P33/PO11/TIOCC0/TIOCD0/TCLKB-A/ DREQ1-B/CS3/CS7-A
96	E10	V <sub>ss</sub>	V <sub>ss</sub>
97	F13	XTAL	XTAL
98	E12	EXTAL	EXTAL
99	E13	V <sub>cc</sub>	V <sub>cc</sub>
100	F11	P32/PO10/TIOCC0/TCLKA-A/DACK0-B/ CS2-A/CS6-A	P32/PO10/TIOCC0/TCLKA-A/DACK0-B/ CS2-A/CS6-A
101	D12	P31/PO9/TIOCA0/TIOCB0/TEND0-B/ CS1/CS2-B/CS5-A/CS6-B/CS7-B	P31/PO9/TIOCA0/TIOCB0/TEND0-B/ CS1/CS2-B/CS5-A/CS6-B/CS7-B
102	E11	$\overline{\text{STBY}}$	$\overline{\text{STBY}}$
103	D10	$\overline{\text{WDTOVF}}$ /TDO	$\overline{\text{WDTOVF}}$ /TDO
104	D13	V <sub>cl</sub>	V <sub>cl</sub>
105	C12	V <sub>ss</sub>	V <sub>ss</sub>
106	C13	P30/PO8/TIOCA0/DREQ0-B/CS0/CS4/CS5-B	P30/PO8/TIOCA0/DREQ0-B/CS0/CS4/CS5-B
107	D11	P60/TMRI2/TxD4/ $\overline{\text{IRQ8}}$ -B	P60/TMRI2/TxD4/ $\overline{\text{IRQ8}}$ -B
108	B13	P61/TMCI2/RxD4/ $\overline{\text{IRQ9}}$ -B	P61/TMCI2/RxD4/ $\overline{\text{IRQ9}}$ -B
109	A12	P62/TMO2/SCK4/ $\overline{\text{IRQ10}}$ -B/TRST	P62/TMO2/SCK4/ $\overline{\text{IRQ10}}$ -B/TRST
110	A13	PLL <sub>V<sub>cc</sub></sub>	PLL <sub>V<sub>cc</sub></sub>
111	B11	P63/TMRI3/ $\overline{\text{IRQ11}}$ -B/TMS	P63/TMRI3/ $\overline{\text{IRQ11}}$ -B/TMS

Pin No.			Pin Name
LQFP	LGA	Modes 1, 2, 6, 7	Modes 4 and 5
112	B12	PLL <sub>V<sub>SS</sub></sub>	PLL <sub>V<sub>SS</sub></sub>
113	A11	P64/TMCI3/ $\overline{\text{IRQ12}}$ -B/TDI	P64/TMCI3/ $\overline{\text{IRQ12}}$ -B/TDI
114	C11	P65/TMO3/ $\overline{\text{IRQ13}}$ -B/TCK	P65/TMO3/ $\overline{\text{IRQ13}}$ -B/TCK
115	B10	V <sub>cc</sub>	V <sub>cc</sub>
116	C10	NMI	NMI
117	A10	MD0	MD0
118	B9	P50/AN0/ $\overline{\text{IRQ0}}$ -B	P50/AN0/ $\overline{\text{IRQ0}}$ -B
119	C9	P51/AN1/ $\overline{\text{IRQ1}}$ -B	P51/AN1/ $\overline{\text{IRQ1}}$ -B
120	B8	P52/AN2/ $\overline{\text{IRQ2}}$ -B	P52/AN2/ $\overline{\text{IRQ2}}$ -B
121	A9	AV <sub>cc</sub>	AV <sub>cc</sub>
122	D9	P53/AN3/ $\overline{\text{IRQ3}}$ -B	P53/AN3/ $\overline{\text{IRQ3}}$ -B
123	C8	AV <sub>ss</sub>	AV <sub>ss</sub>
124	B7	P54/AN4/ $\overline{\text{IRQ4}}$ -B	P54/AN4/ $\overline{\text{IRQ4}}$ -B
125	A8	Vref	Vref
126	D8	P55/AN5/ $\overline{\text{IRQ5}}$ -B	P55/AN5/ $\overline{\text{IRQ5}}$ -B
127	D7	P56/AN6/DA0/ $\overline{\text{IRQ6}}$ -B	P56/AN6/DA0/ $\overline{\text{IRQ6}}$ -B
128	D6	P57/AN7/DA1/ $\overline{\text{IRQ7}}$ -B	P57/AN7/DA1/ $\overline{\text{IRQ7}}$ -B
129	A7	AV <sub>ss</sub> D	AV <sub>ss</sub> D
130	B6	AV <sub>cc</sub> D	AV <sub>cc</sub> D
131	C7	AVCM	AVCM
132	D5	ANDS0	ANDS0
133	A6	ANDS1	ANDS1
134	B5	ANDS2	ANDS2
135	C6	ANDS3	ANDS3
136	D4	ANDS4P	ANDS4P
137	A5	ANDS4N	ANDS4N
138	B4	ANDS5P	ANDS5P
139	C5	ANDS5N	ANDS5N
140	A4	AV <sub>cc</sub> A	AV <sub>cc</sub> A

---

Pin No.			Pin Name
LQFP	LGA	Modes 1, 2, 6, 7	Modes 4 and 5
141	B3	AV <sub>SS</sub> A	AV <sub>SS</sub> A
142	C4	REXT	REXT
143	A3	AV <sub>ref</sub> B	AV <sub>ref</sub> B
144	A2	AV <sub>ref</sub> T	AV <sub>ref</sub> T
	E5	NC	NC

---

### 1.4.3 Pin Functions

**Table 1.4 Pin Functions**

Classification	Pin Name	I/O	Description
Power supply	V <sub>CC</sub>	Input	Power supply pin. Connect it to the system power supply.
	V <sub>CL</sub>	Input	Connect this pin to V <sub>SS</sub> via a 0.1- $\mu$ F capacitor (The capacitor should be placed close to the pin).
	V <sub>SS</sub>	Input	Ground pin. Connect it to the system power supply (0 V).
	PLL <sub>V<sub>CC</sub></sub>	Input	Power supply pin for the PLL circuit. Connect it to the system power supply.
	PLL <sub>V<sub>SS</sub></sub>	Input	Ground pin for the PLL circuit.
Clock	XTAL	Input	Pins for a crystal resonator. An external clock signal can be input through the EXTAL pin. For an example of this connection, see section 23, Clock Pulse Generator.
	EXTAL	Input	
	B $\phi$	Output	Outputs the system clock for external devices.
Operating mode control	MD2 to MD0	Input	Pins for setting the operating mode. The signal levels on these pins must not be changed during operation.
System control	$\overline{\text{RES}}$	Input	Reset signal input pin. This LSI enters the reset state when this signal goes low.
	$\overline{\text{STBY}}$	Input	This LSI enters hardware standby mode when this signal goes low.
	EMLE	Input	Input pin for the on-chip emulator enable signal. Input a high level when using the on-chip emulator, and input a low level when not using the on-chip emulator.
On-chip emulator	$\overline{\text{TRST}}$	Input	Pins for the on-chip emulator Driving the EMLE pin high makes these pins to function as on-chip emulator pins.
	TMS	Input	
	TDI	Input	
	TCK	Input	
	TDO	Output	
Address bus	A20 to A0	Output	Output pins for the address bits.
Data bus	D15 to D0	Input/output	Input and output for the bidirectional data bus. These pins also output addresses when accessing an address–data multiplexed I/O interface space.
Bus control	$\overline{\text{BREQ}}$	Input	External bus-master modules assert this signal to request the bus.
	$\overline{\text{BREQO}}$	Output	Internal bus-master modules assert this signal to request access to the external space via the bus in the external bus released state.

Classification	Pin Name	I/O	Description
Bus control	BACK	Output	Bus acknowledge signal, which indicates that the bus has been released.
	$\overline{BS-A/BS-B}$	Output	Indicates the start of a bus cycle.
	$\overline{AS}$	Output	Strobe signal which indicates that the output address on the address bus is valid in access to the basic bus interface or byte control SRAM interface space.
	$\overline{AH}$	Output	This signal is used to hold the address when accessing the address-data multiplexed I/O interface space.
	$\overline{RD}$	Output	Strobe signal which indicates that reading from the basic bus interface space is in progress.
	$\overline{RD/WR}$	Output	Indicates the direction (input or output) of the data bus.
	$\overline{LHWR}$	Output	Strobe signal which indicates that the higher-order byte (D15 to D8) is valid in access to the basic bus interface space.
	$\overline{LLWR}$	Output	Strobe signal which indicates that the lower-order byte (D7 to D0) is valid in access to the basic bus interface space.
	$\overline{LUB}$	Output	Strobe signal which indicates that the higher-order byte (D15 to D8) is valid in access to the byte control SRAM interface space.
	$\overline{LLB}$	Output	Strobe signal which indicates that the lower-order byte (D7 to D0) is valid in access to the byte control SRAM interface space.
	$\overline{CS0}$ $\overline{CS1}$ $\overline{CS2-A/CS2-B}$ $\overline{CS3}$ $\overline{CS4}$ $\overline{CS5-A/CS5-B}$ $\overline{CS6-A/CS6-B}$ $\overline{CS7-A/CS7-B}$	Output	Select signals for areas 0 to 7.
	WAIT	Input	Requests wait cycles in access to the external space.

Classification	Pin Name	I/O	Description
Interrupt	NMI	Input	Non-maskable interrupt request signal. When this pin is not in use, this signal must be fixed high.
	$\overline{\text{IRQ15}}$	Input	Maskable interrupt request signal.
	$\overline{\text{IRQ14}}$		
	$\overline{\text{IRQ13-A/IRQ13-B}}$		
	$\overline{\text{IRQ12-A/IRQ12-B}}$		
	$\overline{\text{IRQ11-A/IRQ11-B}}$		
	$\overline{\text{IRQ10-A/IRQ10-B}}$		
	$\overline{\text{IRQ9-A/IRQ9-B}}$		
	$\overline{\text{IRQ8-A/IRQ8-B}}$		
	$\overline{\text{IRQ7-A/IRQ7-B}}$		
	$\overline{\text{IRQ6-A/IRQ6-B}}$		
	$\overline{\text{IRQ5-A/IRQ5-B}}$		
	$\overline{\text{IRQ4-A/IRQ4-B}}$		
	$\overline{\text{IRQ3-A/IRQ3-B}}$		
$\overline{\text{IRQ2-A/IRQ2-B}}$			
$\overline{\text{IRQ1-A/IRQ1-B}}$			
$\overline{\text{IRQ0-A/IRQ0-B}}$			
DMA controller (DMAC)	$\overline{\text{DREQ0-A/DREQ0-B}}$	Input	Requests DMAC activation.
	$\overline{\text{DREQ1-A/DREQ1-B}}$		
	$\overline{\text{DACK0-A/DACK0-B}}$	Output	DMAC single address-transfer acknowledge signal.
	$\overline{\text{DACK1-A/DACK1-B}}$		
	$\overline{\text{TEND0-A/TEND0-B}}$	Output	Indicates end of data transfer by the DMAC.
	$\overline{\text{TEND1-A/TEND1-B}}$		
16-bit timer pulse unit (TPU)	$\overline{\text{TCLKA-A/TCLKA-B}}$	Input	Input pins for the external clock signals.
	$\overline{\text{TCLKB-A/TCLKB-B}}$		
	$\overline{\text{TCLKC-A/TCLKC-B}}$		
	$\overline{\text{TCLKD-A/TCLKD-B}}$		
	TIOCA0	Input/ output	Signals for TGRA_0 to TGRD_0. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCB0		
	TIOCC0		
	TIODC0		
	TIOCA1	Input/ output	Signals for TGRA_1 and TGRB_1. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCB1		



Classification	Pin Name	I/O	Description
16-bit timer pulse unit (TPU)	TIOCA2 TIOCB2	Input/ output	Signals for TGRA_2 and TGRB_2. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA3 TIOCB3 TIOCC3 TIOCD3	Input/ output	Signals for TGRA_3 to TGRD_3. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA4 TIOCB4	Input/ output	Signals for TGRA_4 and TGRB_4. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	TIOCA5 TIOCB5	Input/ output	Signals for TGRA_5 and TGRB_5. These pins are used as input capture inputs, output compare outputs, or PWM outputs.
	Programmable pulse generator (PPG)	PO15 to PO0	Output
8-bit timer (TMR)	TMO0 to TMO7	Output	Output pins for the compare match signals.
	TMCI0 to TMCI3	Input	Input pins for the external clock signals that drive for the counters.
	TMRI0 to TMRI3	Input	Input pins for the counter-reset signals.
Watchdog timer (WDT)	$\overline{\text{WDTOVF}}$	Output	Output pin for the counter-overflow signal in watchdog-timer mode.
Serial communication interface (SCI)	TxD0 to TxD4	Output	Output pins for data transmission.
	RxD0 to RxD4	Input	Input pins for data reception.
	SCK0 to SCK4	Input/ output	Input/output pins for clock signals.
I <sup>2</sup> C bus interface 2 (IIC2)	SCL0, SCL1	Input/ output	Input/output pins for clock signals for the IIC2. These pins can drive the bus directly with NMOS open-drain output.
	SDA0, SDA1	Input/ output	Input/output pins for data signals for the IIC2. These pins can drive the bus directly with NMOS open-drain output.

Classification	Pin Name	I/O	Description	
A/D converter	AN7 to AN0	Input	Input pins for the analog signals to be processed by the A/D converter.	
	ADTRG0	Input	Input pin for the external trigger signal that starts A/D conversion.	
D/A converter	DA1 DA0	Output	Output pins for the analog signals from the D/A converter.	
A/D converter, D/A converter	AV <sub>cc</sub>	Input	Analog power supply pin for the A/D and D/A converters. When the A/D and D/A converters are not in use, connect this pin to the system power supply.	
	AV <sub>ss</sub>	Input	Ground pin for the A/D and D/A converters. Connect this pin to the system power supply (0 V).	
	Vref	Input	Reference power supply pin for the A/D and D/A converters. When the A/D and D/A converters are not in use, connect this pin to the system power supply.	
$\Delta\Sigma$ A/D converter	ANDS5N ANDS5P ANDS4N ANDS4P ANDS3 ANDS2 ANDS1 ANDS0	Input	Analog input pins for the $\Delta\Sigma$ A/D converter.	
	ANDSTRG	Input	External trigger input pin for starting $\Delta\Sigma$ A/D conversion.	
	AV <sub>cc</sub> A	Input	Analog power supply pin for the $\Delta\Sigma$ A/D converter. When not using the $\Delta\Sigma$ A/D converter, connect this pin to the system power supply.	
	AV <sub>ss</sub> A	Input	Ground pin for the $\Delta\Sigma$ A/D converter. When not using the $\Delta\Sigma$ A/D converter, connect this pin to the system power supply (0 V).	
	AV <sub>cc</sub> D	Input	Analog power supply pin for the $\Delta\Sigma$ A/D converter. When not using the $\Delta\Sigma$ A/D converter, connect this pin to the system power supply.	
	AV <sub>ss</sub> D	Input	Ground pin for the $\Delta\Sigma$ A/D converter. When not using the $\Delta\Sigma$ A/D converter, connect this pin to the system power supply (0 V).	
	AV <sub>ref</sub> T AV <sub>ref</sub> B	Input	The same power as AV <sub>cc</sub> A and AV <sub>ss</sub> A is input to AV <sub>ref</sub> T and AV <sub>ref</sub> B, respectively. See section 19.2, Input/Output Pins, for details.	

Classification	Pin Name	I/O	Description
$\Delta\Sigma$ A/D converter	AVCM	Output	Connect a stabilizing capacitor between AVCM and AV <sub>SS</sub> A. See section 19.2, Input/Output Pins, for details.
	REXT	Output	Connect an external resistor between REXT and AV <sub>SS</sub> A. See section 19.2, Input/Output Pins, for details.
	AV <sub>CC</sub> P	Input	Analog power supply pin for the input buffers of the $\Delta\Sigma$ A/D converter.  When not using the $\Delta\Sigma$ A/D converter, connect this pin to the system power supply.
	AV <sub>SS</sub> P	Input	Ground pin for the input buffers of the $\Delta\Sigma$ A/D converter.  When not using the $\Delta\Sigma$ A/D converter, connect this pin to the system power supply (0 V).
I/O ports	P17 to P10	Input/output	8 input/output pins.
	P27 to P20	Input/output	8 input/output pins.
	P37 to P30	Input/output	8 input/output pins.
	P47 to P40	Input	8 input pins.
	P57 to P50	Input	8 input/output pins.
	P65 to P60	Input/output	6 input/output pins.
	PA7	Input	Input-only pin
	PA6 to PA0	Input/output	7 input/output pins.
	PD7 to PD0	Input/output	8 input/output pins.
	PE7 to PE0	Input/output	8 input/output pins.
	PF4 to PF0	Input/output	5 input/output pins.
	PH7 to PH0	Input/output	8 input/output pins.
	PI7 to PI0	Input/output	8 input/output pins.



## Section 2 CPU

The H8SX CPU is a high-speed CPU with an internal 32-bit architecture that is upward compatible with the H8/300, H8/300H, and H8S CPUs.

The H8SX CPU has sixteen 16-bit general registers, can handle a 4-Gbyte linear address space, and is ideal for a realtime control system.

### 2.1 Features

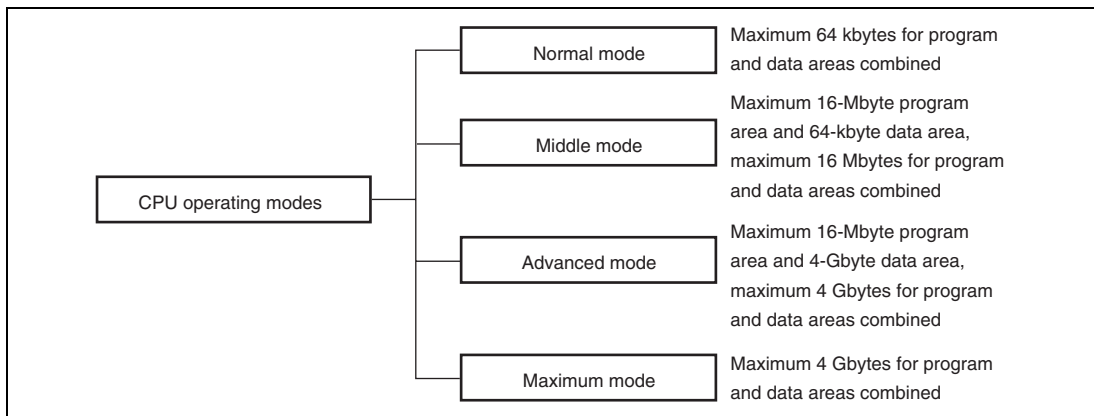
- Upward-compatible with H8/300, H8/300H, and H8S CPUs
  - Can execute object programs of these CPUs
- Sixteen 16-bit general registers
  - Also usable as sixteen 8-bit registers or eight 32-bit registers
- 87 basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Bit field transfer instructions
  - Powerful bit-manipulation instructions
  - Bit condition branch instructions
  - Multiply-and-accumulate instruction
- Eleven addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:2,ERn), @(d:16,ERn), or @(d:32,ERn)]
  - Index register indirect with displacement [@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)]
  - Register indirect with pre-/post-increment or pre-/post-decrement [ @+ERn, @-ERn, @ERn+, or @ERn- ]
  - Absolute address [ @aa:8, @aa:16, @aa:24, or @aa:32 ]
  - Immediate [ #xx:3, #xx:4, #xx:8, #xx:16, or #xx:32 ]
  - Program-counter relative [ @(d:8,PC) or @(d:16,PC) ]
  - Program-counter relative with index register [ @(RnL.B,PC), @(Rn.W,PC), or @(ERn.L,PC) ]
  - Memory indirect [ @@aa:8 ]
  - Extended memory indirect [ @@vec:7 ]

- Two base registers
  - Vector base register
  - Short address base register
- 4-Gbyte address space
  - Program: 4 Gbytes
  - Data: 4 Gbytes
- High-speed operation
  - All frequently-used instructions executed in one or two states
  - 8/16/32-bit register-register add/subtract: 1 state
  - $8 \times 8$ -bit register-register multiply: 1 state
  - $16 \div 8$ -bit register-register divide: 10 states
  - $16 \times 16$ -bit register-register multiply: 1 state
  - $32 \div 16$ -bit register-register divide: 18 states
  - $32 \times 32$ -bit register-register multiply: 5 states
  - $32 \div 32$ -bit register-register divide: 18 states
- Four CPU operating modes
  - Normal mode
  - Middle mode
  - Advanced mode
  - Maximum mode
- Power-down modes
  - Transition is made by execution of SLEEP instruction
  - Choice of CPU operating clocks

- Notes: 1. Advanced mode is only supported as the CPU operating mode of the H8SX/1622 Group. Normal, middle, and maximum modes are not supported.
2. The multiplier and divider are supported by the H8SX/1622 Group.

## 2.2 CPU Operating Modes

The H8SX CPU has four operating modes: normal, middle, advanced and maximum modes. These modes can be selected by the mode pins of this LSI.



**Figure 2.1 CPU Operating Modes**

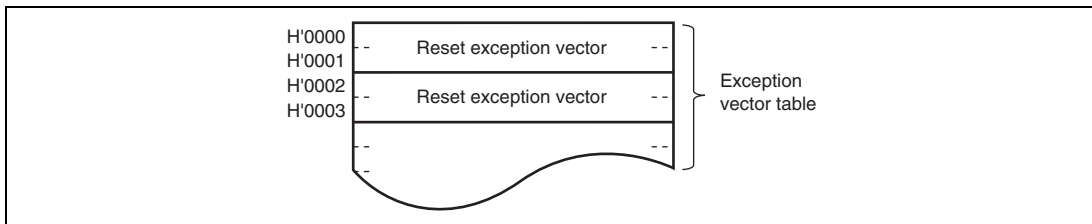
### 2.2.1 Normal Mode

The exception vector table and stack have the same structure as in the H8/300 CPU.

- **Address Space**  
The maximum address space of 64 kbytes can be accessed.
- **Extended Registers (En)**  
The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When the extended register En is used as a 16-bit register it can contain any value, even when the corresponding general register Rn is used as an address register. (If the general register Rn is referenced in the register indirect addressing mode with pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, the value in the corresponding extended register En will be affected.)
- **Instruction Set**  
All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.

- Exception Vector Table and Memory Indirect Branch Addresses

In normal mode, the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The structure of the exception vector table is shown in figure 2.2.

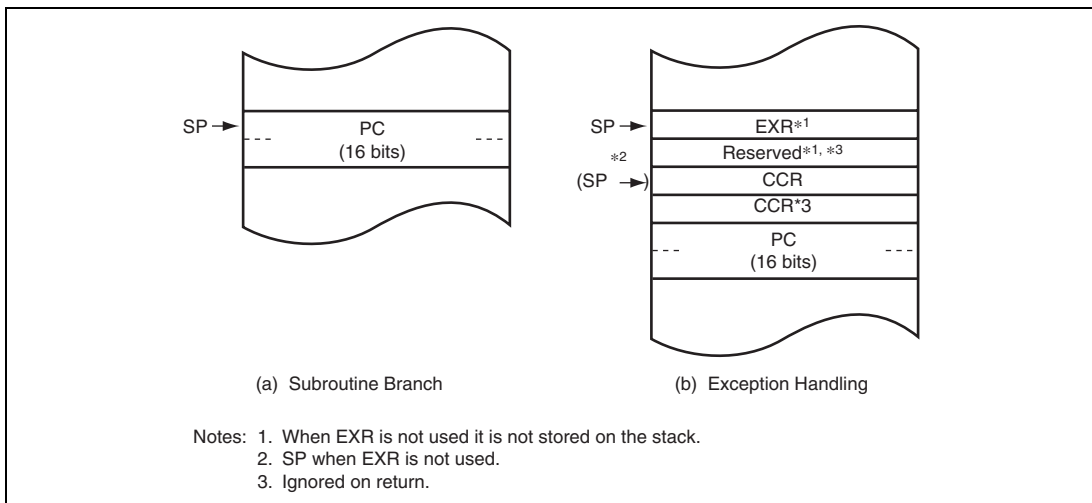


**Figure 2.2 Exception Vector Table (Normal Mode)**

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

- Stack Structure

The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.3. The PC contents are saved or restored in 16-bit units.



**Figure 2.3 Stack Structure (Normal Mode)**



## 2.2.2 Middle Mode

The program area in middle mode is extended to 16 Mbytes as compared with that in normal mode.

- Address Space

The maximum address space of 16 Mbytes can be accessed as a total of the program and data areas. For individual areas, up to 16 Mbytes of the program area or up to 64 kbytes of the data area can be allocated.

- Extended Registers (En)

The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When the extended register En is used as a 16-bit register (in other than the JMP and JSR instructions), it can contain any value even when the corresponding general register Rn is used as an address register. (If the general register Rn is referenced in the register indirect addressing mode with pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, the value in the corresponding extended register En will be affected.)

- Instruction Set

All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid and the upper eight bits are sign-extended.

- Exception Vector Table and Memory Indirect Branch Addresses

In middle mode, the top area starting at H'000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

In middle mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address. The upper eight bits are reserved and assumed to be H'00.

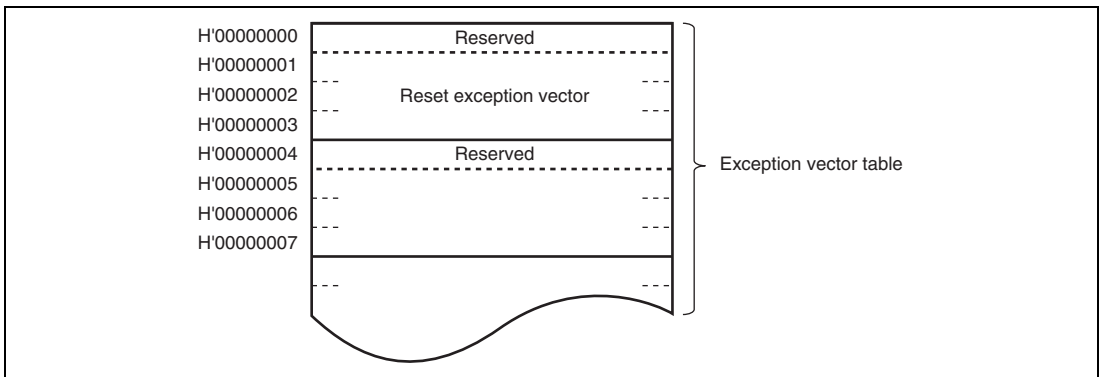
- Stack Structure

The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.

### 2.2.3 Advanced Mode

The data area is extended to 4 Gbytes as compared with that in middle mode.

- **Address Space**  
The maximum address space of 4 Gbytes can be linearly accessed. For individual areas, up to 16 Mbytes of the program area and up to 4 Gbytes of the data area can be allocated.
- **Extended Registers (En)**  
The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers or address registers.
- **Instruction Set**  
All instructions and addressing modes can be used.
- **Exception Vector Table and Memory Indirect Branch Addresses**  
In advanced mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.

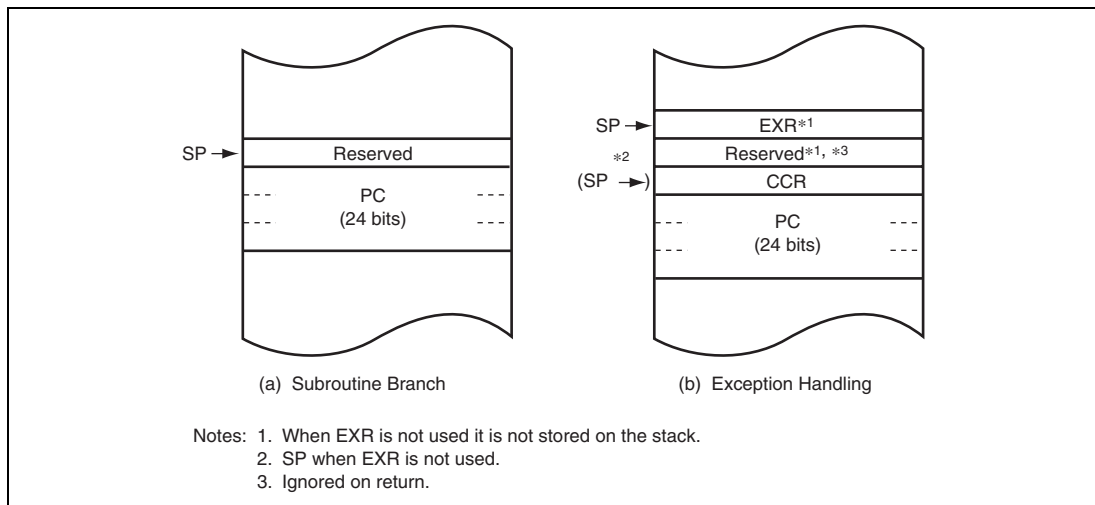


**Figure 2.4 Exception Vector Table (Middle and Advanced Modes)**

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location. In advanced mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address. The upper eight bits are reserved and assumed to be H'00.

- Stack Structure

The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.



**Figure 2.5 Stack Structure (Middle and Advanced Modes)**

### 2.2.4 Maximum Mode

The program area is extended to 4 Gbytes as compared with that in advanced mode.

- Address Space

The maximum address space of 4 Gbytes can be linearly accessed.

- Extended Registers (En)

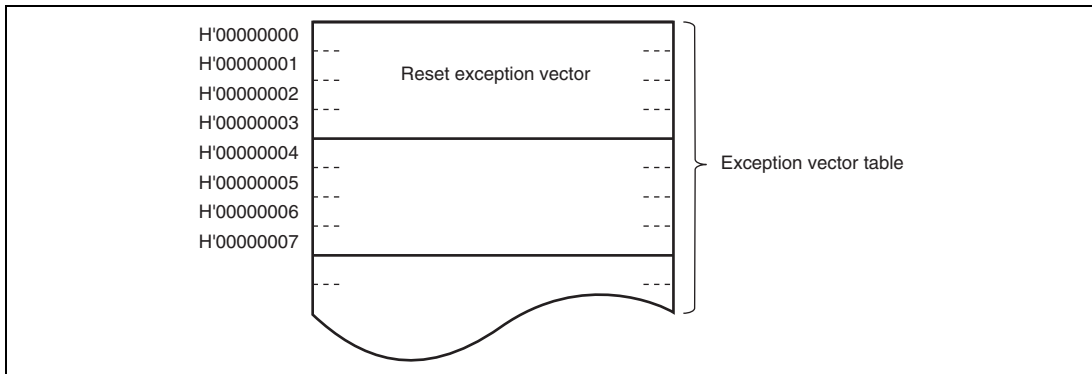
The extended registers (E0 to E7) can be used as 16-bit registers or as the upper 16-bit segments of 32-bit registers or address registers.

- Instruction Set

All instructions and addressing modes can be used.

- Exception Vector Table and Memory Indirect Branch Addresses

In maximum mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The structure of the exception vector table is shown in figure 2.6.

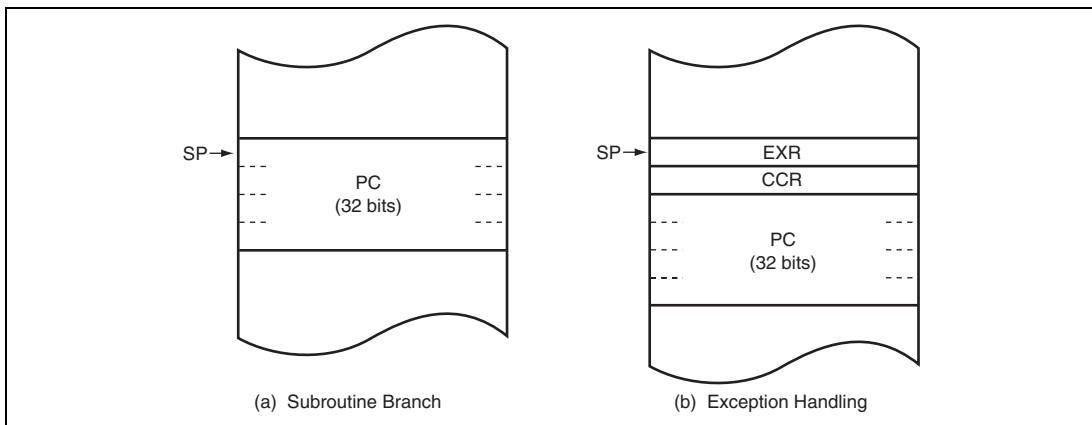


**Figure 2.6 Exception Vector Table (Maximum Modes)**

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location. In maximum mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address.

- Stack Structure

The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.7. The PC contents are saved or restored in 32-bit units. The EXR contents are saved or restored regardless of whether or not EXR is in use.



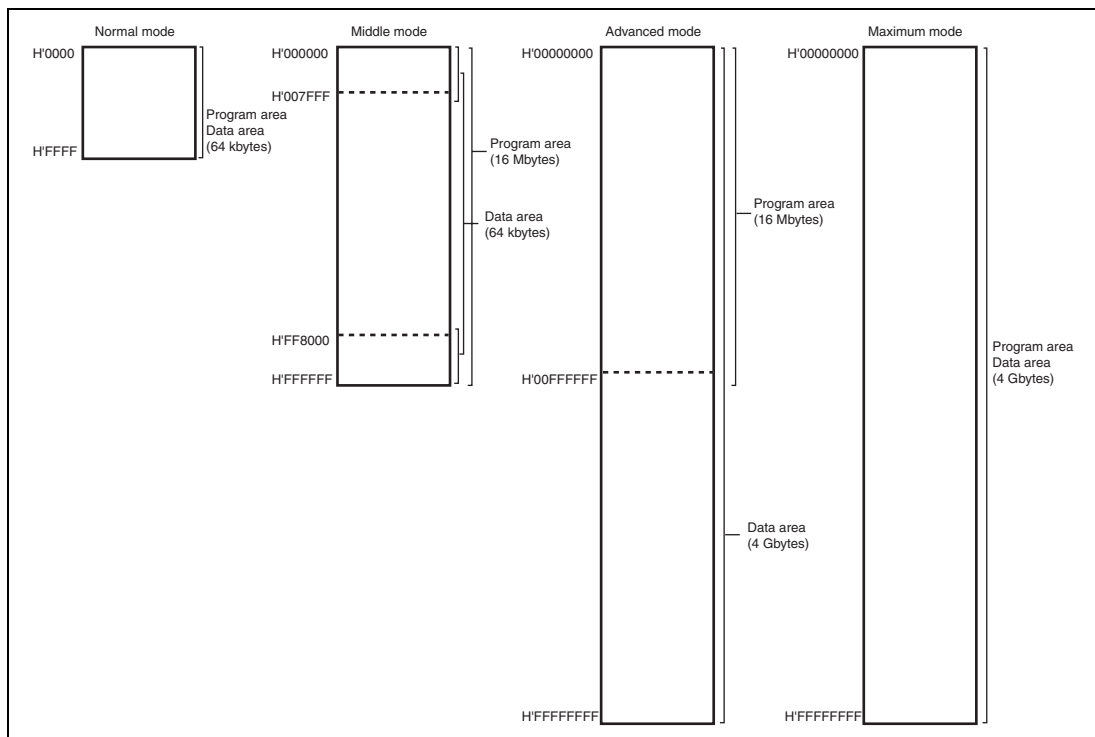
**Figure 2.7 Stack Structure (Maximum Mode)**

## 2.3 Instruction Fetch

The H8SX CPU has two modes for instruction fetch: 16-bit and 32-bit modes. It is recommended that the mode be set according to the bus width of the memory in which a program is stored. The instruction-fetch mode setting does not affect operation other than instruction fetch such as data accesses. Whether an instruction is fetched in 16- or 32-bit mode is selected by the FETCHMD bit in SYSCR. For details, see section 3.2.2, System Control Register (SYSCR).

## 2.4 Address Space

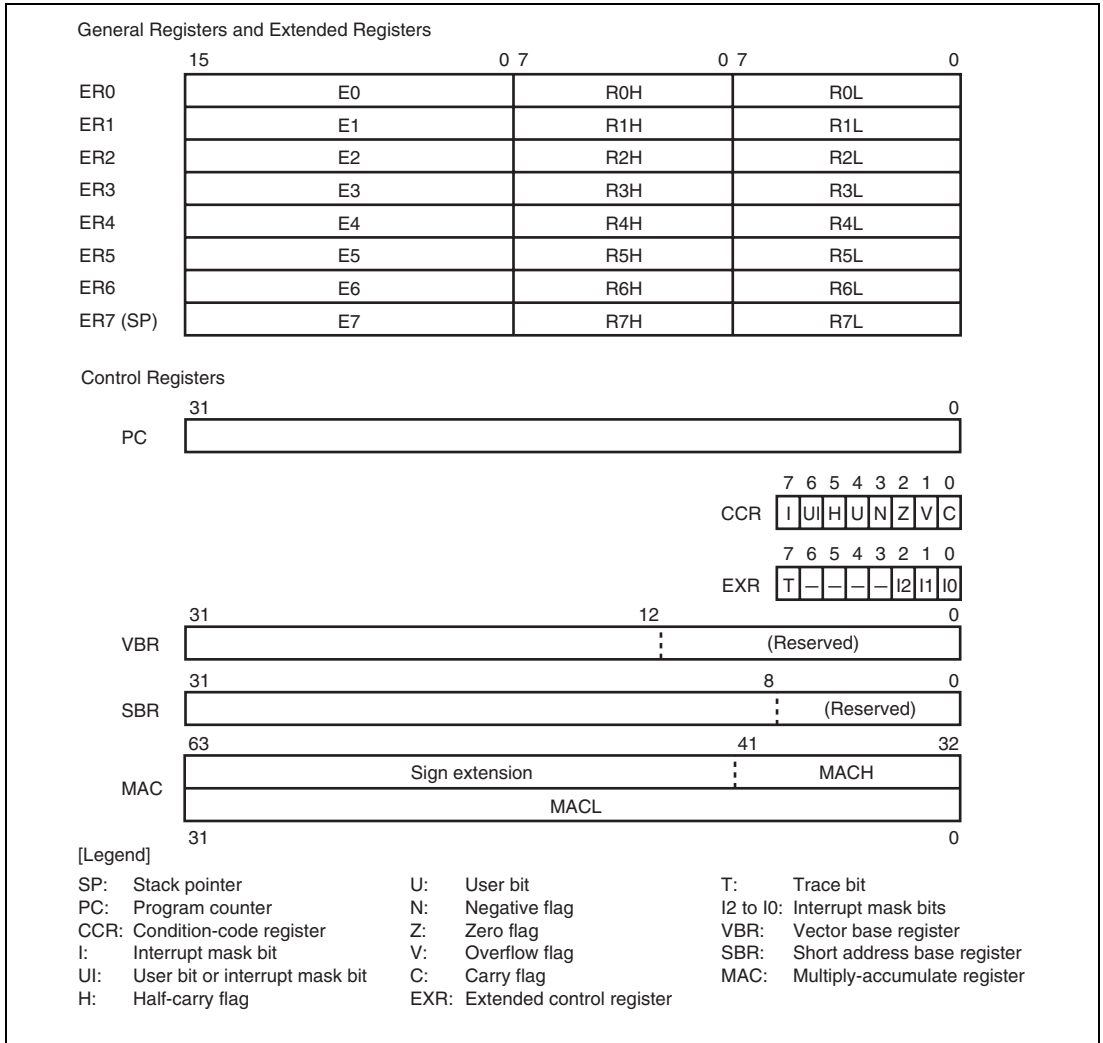
Figure 2.8 shows a memory map of the H8SX CPU. The address space differs depending on the CPU operating mode.



**Figure 2.8 Memory Map**

## 2.5 Registers

The H8SX CPU has the internal registers shown in figure 2.9. There are two types of registers: general registers and control registers. The control registers are the 32-bit program counter (PC), 8-bit extended control register (EXR), 8-bit condition-code register (CCR), 32-bit vector base register (VBR), 32-bit short address base register (SBR), and 64-bit multiply-accumulate register (MAC).



**Figure 2.9 CPU Registers**

## 2.5.1 General Registers

The H8SX CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.10 illustrates the usage of the general registers.

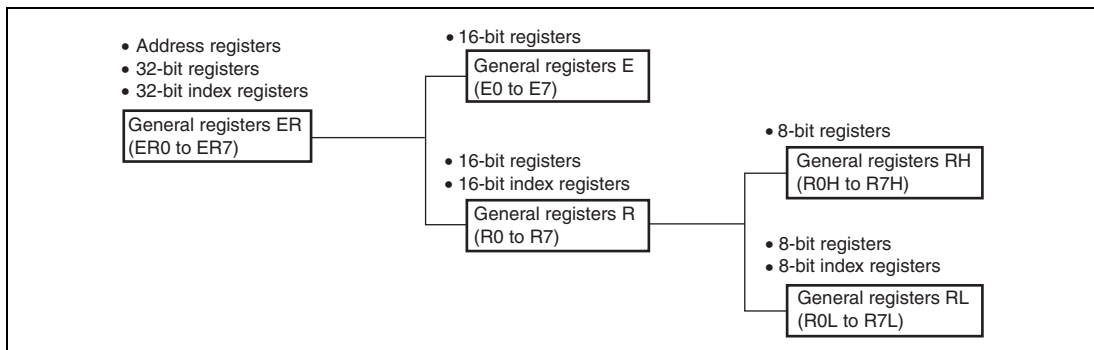
When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

When the general registers are used as 16-bit registers, the ER registers are divided into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

When the general registers are used as 8-bit registers, the R registers are divided into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

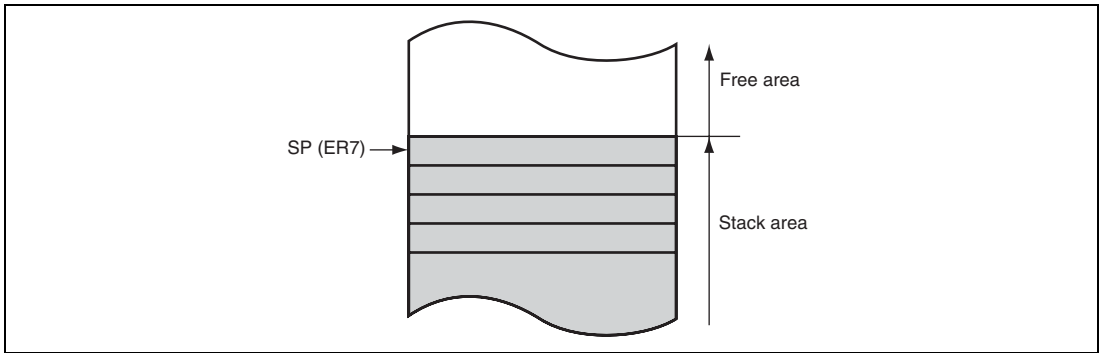
The general registers ER (ER0 to ER7), R (R0 to R7), and RL (R0L to R7L) are also used as index registers. The size in the operand field determines which register is selected.

The usage of each register can be selected independently.



**Figure 2.10 Usage of General Registers**

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine branches. Figure 2.11 shows the stack.



**Figure 2.11 Stack**

### 2.5.2 Program Counter (PC)

PC is a 32-bit counter that indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 16 bits (one word) or a multiple of 16 bits, so the least significant bit is ignored. (When the instruction code is fetched, the least significant bit is regarded as 0.)



### 2.5.3 Condition-Code Register (CCR)

CCR is an 8-bit register that contains internal CPU status information, including an interrupt mask (I) and user (UI, U) bits and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

Operations can be performed on the CCR bits by the LDC, STC, ANDC, ORC, and XORC instructions. The N, Z, V, and C flags are used as branch conditions for conditional branch (Bcc) instructions.

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	Interrupt Mask Bit Masks interrupts when set to 1. This bit is set to 1 at the start of an exception handling.
6	UI	Undefined	R/W	User Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
5	H	Undefined	R/W	Half-Carry Flag When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.
4	U	Undefined	R/W	User Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag Stores the value of the most significant bit (regarded as sign bit) of data.

Bit	Bit Name	Initial Value	R/W	Description
2	Z	Undefined	R/W	Zero Flag Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.
0	C	Undefined	R/W	Carry Flag Set to 1 when a carry occurs, and cleared to 0 otherwise. A carry has the following types: <ul style="list-style-type: none"> <li>• Carry from the result of addition</li> <li>• Borrow from the result of subtraction</li> <li>• Carry from the result of shift or rotation</li> </ul> The carry flag is also used as a bit accumulator by bit manipulation instructions.

#### 2.5.4 Extended Control Register (EXR)

EXR is an 8-bit register that contains the trace bit (T) and three interrupt mask bits (I2 to I0).

Operations can be performed on the EXR bits by the LDC, STC, ANDC, ORC, and XORC instructions.

For details, see section 5, Exception Handling.

Bit	Bit Name	Initial Value	R/W	Description
7	T	0	R/W	Trace Bit When this bit is set to 1, a trace exception is generated each time an instruction is executed. When this bit is cleared to 0, instructions are executed in sequence.
6 to 3	—	All 1	R/W	Reserved These bits are always read as 1.
2	I2	1	R/W	Interrupt Mask Bits
1	I1	1	R/W	These bits designate the interrupt mask level (0 to 7).
0	I0	1	R/W	

### 2.5.5 Vector Base Register (VBR)

VBR is a 32-bit register in which the upper 20 bits are valid. The lower 12 bits of this register are read as 0s. This register is a base address of the vector area for exception handlings other than a reset and a CPU address error (extended memory indirect is also out of the target). The initial value is H'00000000. The VBR contents are changed with the LDC and STC instructions.

### 2.5.6 Short Address Base Register (SBR)

SBR is a 32-bit register in which the upper 24 bits are valid. The lower eight bits are read as 0s. In 8-bit absolute address addressing mode (@aa:8), this register is used as the upper address. The initial value is H'FFFFFF00. The SBR contents are changed with the LDC and STC instructions.

### 2.5.7 Multiply-Accumulate Register (MAC)

MAC is a 64-bit register that stores the results of multiply-and-accumulate operations. It consists of two 32-bit registers denoted MACH and MACL. The lower 10 bits of MACH are valid; the upper bits are sign extended. The MAC contents are changed with the MAC, CLRMAC, LDMAC, and STMAC instructions.

### 2.5.8 Initial Values of CPU Registers

Reset exception handling loads the start address from the vector table into the PC, clears the T bit in EXR to 0, and sets the I bits in CCR and EXR to 1. The general registers, MAC, and the other bits in CCR are not initialized. In particular, the initial value of the stack pointer (ER7) is undefined. The SP should therefore be initialized using an MOV.L instruction executed immediately after a reset.

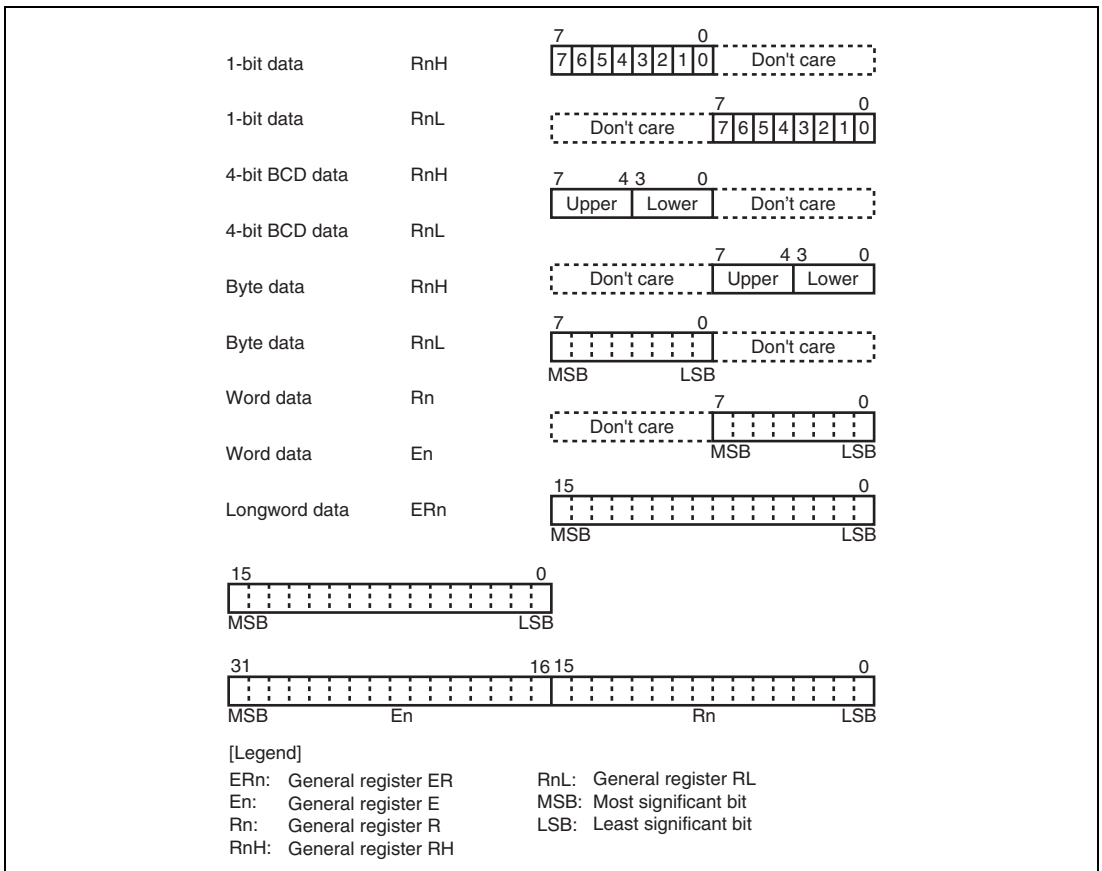
## 2.6 Data Formats

The H8SX CPU can process 1-bit, 4-bit BCD, 8-bit (byte), 16-bit (word), and 32-bit (longword) data.

Bit-manipulation instructions operate on 1-bit data by accessing bit  $n$  ( $n = 0, 1, 2, \dots, 7$ ) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

### 2.6.1 General Register Data Formats

Figure 2.12 shows the data formats in general registers.



**Figure 2.12 General Register Data Formats**

### 2.6.2 Memory Data Formats

Figure 2.13 shows the data formats in memory.

The H8SX CPU can access word data and longword data which are stored at any addresses in memory. When word data begins at an odd address or longword data begins at an address other than a multiple of 4, a bus cycle is divided into two or more accesses. For example, when longword data begins at an odd address, the bus cycle is divided into byte, word, and byte accesses. In this case, these accesses are assumed to be individual bus cycles.

However, instructions to be fetched, word and longword data to be accessed during execution of the stack manipulation, branch table manipulation, block transfer instructions, and MAC instruction should be located to even addresses.

When SP (ER7) is used as an address register to access the stack, the operand size should be word size or longword size.

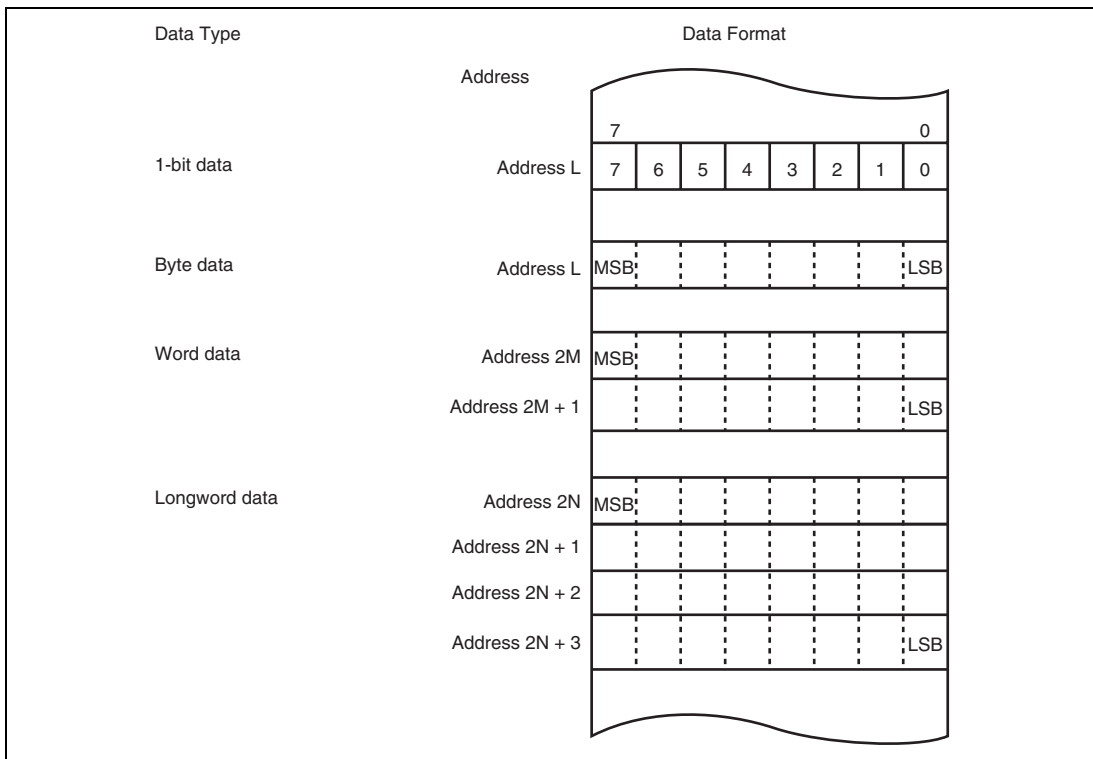


Figure 2.13 Memory Data Formats

## 2.7 Instruction Set

The H8SX CPU has 87 types of instructions. The instructions are classified by function as shown in table 2.1. The arithmetic operation, logic operation, shift, and bit manipulation instructions are called operation instruction in this manual.

**Table 2.1 Instruction Classification**

Function	Instructions	Size	Types
Data transfer	MOV	B/W/L	6
	MOVFPE* <sup>6</sup> , MOVTPE* <sup>6</sup>	B	
	POP, PUSH* <sup>1</sup>	W/L	
	LDM, STM	L	
	MOVA	B/W* <sup>2</sup>	
Block transfer	EPMOV	B	3
	MOVMD	B/W/L	
	MOVSD	B	
Arithmetic operations	ADD, ADDX, SUB, SUBX, CMP, NEG, INC, DEC	B/W/L	27
	DAA, DAS	B	
	ADDS, SUBS	L	
	MULXU, DIVXU, MULXS, DIVXS	B/W	
	MULU, DIVU, MULS, DIVS	W/L	
	MULU/U, MULS/U	L	
	EXTU, EXTS	W/L	
	TAS	B	
	MAC	—	
	LDMAC, STMAC	—	
	CLRMAC	—	
	Logic operations	AND, OR, XOR, NOT	
Shift	SHLL, SHLR, SHAL, SHAR, ROTL, ROTR, ROTXL, ROTXR	B/W/L	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAN, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	B	20
	BSET/EQ, BSET/NE, BCLR/EQ, BCLR/NE, BSTZ, BISTZ	B	
	BFLD, BFST	B	

Function	Instructions	Size	Types
Branch	BRA/BS, BRA/BC, BSR/BS, BSR/BC	B* <sup>3</sup>	9
	Bcc* <sup>5</sup> , JMP, BSR, JSR, RTS	—	
	RTS/L	L* <sup>5</sup>	
	BRA/S	—	
System control	TRAPA, RTE, SLEEP, NOP	—	10
	RTE/L	L* <sup>5</sup>	
	LDC, STC, ANDC, ORC, XORC	B/W/L	
		Total	87

## [Legend]

B: Byte size

W: Word size

L: Longword size

- Notes: 1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @-SP.  
 POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @-SP.
2. Size of data to be added with a displacement
  3. Size of data to specify a branch condition
  4. Bcc is the generic designation of a conditional branch instruction.
  5. Size of general register to be restored
  6. Not available in this LSI.

## 2.7.1 Instructions and Addressing Modes

Table 2.2 indicates the combinations of instructions and addressing modes that the H8SX CPU can use.

**Table 2.2 Combinations of Instructions and Addressing Modes (1)**

Classification	Instruction	Size	#xx	Rn	Addressing Mode						
					@ERn	@(d,ERn)	ERn.L)	@+ERn	@aa:8	@aa:16/ @aa:32	—
Data transfer	MOV	B/W/L	S	SD	SD	SD	SD	SD	SD	SD	
		B		S/D					S/D		
	MOVFP, MOVTP* <sup>12</sup>	B		S/D						S/D* <sup>1</sup>	
	POP, PUSH	W/L		S/D				S/D* <sup>2</sup>			
	LDM, STM	L		S/D				S/D* <sup>2</sup>			
	MOVA* <sup>4</sup>	B/W		S	S	S	S	S	S	S	
Block transfer	EEPMOV	B									SD* <sup>3</sup>
	MOVMD	B/W/L									SD* <sup>3</sup>
	MOVSD	B									SD* <sup>3</sup>
Arithmetic operations	ADD, CMP	B	S	D	D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD			SD
		W/L	S	SD	SD	SD	SD	SD			SD
	SUB	B	S		D	D	D	D	D	D	
		B			S	D	D	D	D	D	
		B			D	S	S	S	S	S	
		B				SD	SD	SD	SD		SD
		W/L	S	SD	SD	SD	SD	SD			SD
	ADDX, SUBX	B/W/L	S		SD						
		B/W/L	S			SD					
		B/W/L	S						SD* <sup>5</sup>		
	INC, DEC	B/W/L			D						
	ADDS, SUBS	L			D						
	DAA, DAS	B			D						
MULXU, DIVXU	B/W	S:4		SD							
MULU, DIVU	W/L	S:4		SD							



Classifi- cation	Instruction	Size	#xx	Rn	Addressing Mode						
					@ERn	@(d,ERn)	ERn.L)	@+ERn	@aa:8	@aa:32	—
Arithmetic operations	MULXS, DIVXS	B/W	S:4	SD							
	MULS, DIVS	W/L	S:4	SD							
	NEG	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D	D	D	
	EXTU, EXTS	W/L		D	D	D	D	D	D		
	TAS	B			D						
	MAC	—									
	CLRMAC	—									O
	LDMAC	—		S							
STMAC	—		D								
Logic operations	AND, OR, XOR	B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	NOT	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
Shift	SHLL, SHLR	B		D	D	D	D	D	D	D	
		B/W/L* <sup>6</sup>		D	D	D	D	D		D	
		B/W/L* <sup>7</sup>		D							
	SHAL, SHAR ROTL, ROTR ROTXL, ROTXR	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
Bit manipu- lation	BSET, BCLR, BNOT, BTST, BSET/cc, BCLR/cc	B		D	D				D	D	
		BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST, BSTZ, BISTZ	B		D	D				D	D

Classification	Instruction	Size	#xx	Rn	@ERn	Addressing Mode					
						@(d, ERn)	@(d, ERn) ERn.L)	@-ERn/ @ERn+/ @ERn-/ @+ERn	@aa:16/ @aa:32	—	
Bit manipulation	BFLD	B		D	S				S	S	
	BFST	B		S	D				D	D	
Branch	BRA/BS, BRA/BC* <sup>8</sup>	B			S				S	S	
	BSR/BS, BSR/BC* <sup>8</sup>	B			S				S	S	
System control	LDC (CCR, EXR)	B/W* <sup>9</sup>	S	S	S	S			S* <sup>10</sup>	S	
	LDC (VBR, SBR)	L			S						
	STC (CCR, EXR)	B/W* <sup>9</sup>		D	D	D			D* <sup>11</sup>	D	
	STC (VBR, SBR)	L			D						
	ANDC, ORC, XORC	B	S								
	SLEEP	—									
NOP	—										O

## [Legend]

d: d:16 or d:32

S: Can be specified as a source operand.

D: Can be specified as a destination operand.

SD: Can be specified as either a source or destination operand or both.

S/D: Can be specified as either a source or destination operand.

S:4: 4-bit immediate data can be specified as a source operand.

Notes: 1. Only @aa:16 is available.

2. @ERn+ as a source operand and @-ERn as a destination operand

3. Specified by ER5 as a source address and ER6 as a destination address for data transfer.

4. Size of data to be added with a displacement

5. Only @ERn- is available

6. When the number of bits to be shifted is 1, 2, 4, 8, or 16

7. When the number of bits to be shifted is specified by 5-bit immediate data or a general register

8. Size of data to specify a branch condition

9. Byte when immediate or register direct, otherwise, word

10. Only @ERn+ is available

11. Only @-ERn is available

12. Not available in this LSI.

**Table 2.2 Combinations of Instructions and Addressing Modes (2)**

Classifi- cation	Instruction	Size	Addressing Mode							
			@ERn	@(d,PC)	PC	@aa:24	@aa:32	@@aa:8	@@vec:7	
Branch	BRA/BS, BRA/BC	—		O						
	BSR/BS, BSR/BC	—		O						
	Bcc	—		O						
	BRA	—		O	O					
	BRA/S	—		O*						
	JMP	—	O				O	O	O	O
	BSR	—		O						
	JSR	—	O				O	O	O	O
	RTS, RTS/L	—								O
System control	TRAPA	—								O
	RTE, RTE/L	—								O

**[Legend]**

d: d:8 or d:16

Note: \* Only @(d:8, PC) is available.

## 2.7.2 Table of Instructions Classified by Function

Tables 2.4 to 2.11 summarize the instructions in each functional category. The notation used in these tables is defined in table 2.3.

**Table 2.3 Operation Notation**

Operation Notation	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extended control register
CCR	Condition-code register
VBR	Vector base register
SBR	Short address base 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
+	Addition
−	Subtraction
×	Multiplication
÷	Division
^	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Move
~	Logical not (logical complement)
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

Note: \* 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).

**Table 2.4 Data Transfer Instructions**

<b>Instruction</b>	<b>Size</b>	<b>Function</b>
MOV	B/W/L	#IMM → (EAd), (EAs) → (EAd) Transfers data between immediate data, general registers, and memory.
MOVFP*	B	(EAs) → Rd
MOVTPE*	B	Rs → (EAs)
POP	W/L	@SP+ → Rn Restores the data from the stack to a general register.
PUSH	W/L	Rn → @-SP Saves general register contents on the stack.
LDM	L	@SP+ → Rn (register list) Restores the data from the stack to multiple general registers. Two, three, or four general registers which have serial register numbers can be specified.
STM	L	Rn (register list) → @-SP Saves the contents of multiple general registers on the stack. Two, three, or four general registers which have serial register numbers can be specified.
MOVA	B/W	EA → Rd Zero-extends and shifts the contents of a specified general register or memory data and adds them with a displacement. The result is stored in a general register.

Note: Not available in this LSI.

**Table 2.5 Block Transfer Instructions**

<b>Instruction</b>	<b>Size</b>	<b>Function</b>
EEPMOV.B EEPMOV.W	B	Transfers a data block.  Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4 or R4L.
MOVMD.B	B	Transfers a data block.  Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4.
MOVMD.W	W	Transfers a data block.  Transfers word data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of word data to be transferred is specified by R4.
MOVMD.L	L	Transfers a data block.  Transfers longword data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of longword data to be transferred is specified by R4.
MOVSD.B	B	Transfers a data block with zero data detection.  Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4. When zero data is detected during transfer, the transfer stops and execution branches to a specified address.

**Table 2.6 Arithmetic Operation Instructions**

Instruction	Size	Function
ADD SUB	B/W/L	$(EAd) \pm \#IMM \rightarrow (EAd)$ , $(EAd) \pm (EAs) \rightarrow (EAd)$ Performs addition or subtraction on data between immediate data, general registers, and memory. Immediate byte data cannot be subtracted from byte data in a general register.
ADDX SUBX	B/W/L	$(EAd) \pm \#IMM \pm C \rightarrow (EAd)$ , $(EAd) \pm (EAs) \pm C \rightarrow (EAd)$ Performs addition or subtraction with carry on data between immediate data, general registers, and memory. The addressing mode which specifies a memory location can be specified as register indirect with post-decrement or register indirect.
INC DEC	B/W/L	$Rd \pm 1 \rightarrow Rd$ , $Rd \pm 2 \rightarrow Rd$ Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.)
ADDS SUBS	L	$Rd \pm 1 \rightarrow Rd$ , $Rd \pm 2 \rightarrow Rd$ , $Rd \pm 4 \rightarrow Rd$ Adds or subtracts the value 1, 2, or 4 to or from data in a general register.
DAA DAS	B	$Rd$ (decimal adjust) $\rightarrow Rd$ Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 2-digit 4-bit BCD data.
MULXU	B/W	$Rd \times Rs \rightarrow Rd$ Performs unsigned multiplication on data in two general registers: either 8 bits $\times$ 8 bits $\rightarrow$ 16 bits, or 16 bits $\times$ 16 bits $\rightarrow$ 32 bits.
MULU	W/L	$Rd \times Rs \rightarrow Rd$ Performs unsigned multiplication on data in two general registers: either 8 bits $\times$ 8 bits $\rightarrow$ 16 bits, or 16 bits $\times$ 16 bits $\rightarrow$ 32 bits.
MULU/U	L	$Rd \times Rs \rightarrow Rd$ Performs unsigned multiplication on data in two general registers (32 bits $\times$ 32 bits $\rightarrow$ upper 32 bits).
MULXS	B/W	$Rd \times Rs \rightarrow Rd$ Performs signed multiplication on data in two general registers: either 8 bits $\times$ 8 bits $\rightarrow$ 16 bits, or 16 bits $\times$ 16 bits $\rightarrow$ 32 bits.
MULS	W/L	$Rd \times Rs \rightarrow Rd$ Performs signed multiplication on data in two general registers: either 16 bits $\times$ 16 bits $\rightarrow$ 16 bits, or 32 bits $\times$ 32 bits $\rightarrow$ 32 bits.
MULS/U	L	$Rd \times Rs \rightarrow Rd$ Performs signed multiplication on data in two general registers (32 bits $\times$ 32 bits $\rightarrow$ upper 32 bits).
DIVXU	B/W	$Rd \div Rs \rightarrow Rd$ Performs unsigned division on data in two general registers: either 16 bits $\div$ 8 bits $\rightarrow$ 8-bit quotient and 8-bit remainder, or 32 bits $\div$ 16 bits $\rightarrow$ 16-bit quotient and 16-bit remainder.

Instruction	Size	Function
DIVU	W/L	$Rd \div Rs \rightarrow Rd$ Performs unsigned division on data in two general registers: either 16 bits $\div$ 16 bits $\rightarrow$ 16-bit quotient, or 32 bits $\div$ 32 bits $\rightarrow$ 32-bit quotient.
DIVXS	B/W	$Rd \div Rs \rightarrow Rd$ Performs signed division on data in two general registers: either 16 bits $\div$ 8 bits $\rightarrow$ 8-bit quotient and 8-bit remainder, or 32 bits $\div$ 16 bits $\rightarrow$ 16-bit quotient and 16-bit remainder.
DIVS	W/L	$Rd \div Rs \rightarrow Rd$ Performs signed division on data in two general registers: either 16 bits $\div$ 16 bits $\rightarrow$ 16-bit quotient, or 32 bits $\div$ 32 bits $\rightarrow$ 32-bit quotient.
CMP	B/W/L	(EAd) – #IMM, (EAd) – (EAs) Compares data between immediate data, general registers, and memory and stores the result in CCR.
NEG	B/W/L	$0 - (EAd) \rightarrow (EAd)$ Takes the two's complement (arithmetic complement) of data in a general register or the contents of a memory location.
EXTU	W/L	(EAd) (zero extension) $\rightarrow$ (EAd) Performs zero-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size. The lower 8 bits to word or longword, or the lower 16 bits to longword can be zero-extended.
EXTS	W/L	(EAd) (sign extension) $\rightarrow$ (EAd) Performs sign-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size. The lower 8 bits to word or longword, or the lower 16 bits to longword can be sign-extended.
TAS	B	@ERd – 0, 1 $\rightarrow$ (<bit 7> of @EAd) Tests memory contents, and sets the most significant bit (bit 7) to 1.
MAC	—	(EAs) $\times$ (EAd) + MAC $\rightarrow$ MAC Performs signed multiplication on memory contents and adds the result to MAC.
CLRMAC	—	$0 \rightarrow$ MAC Clears MAC to zero.
LDMAC	—	$Rs \rightarrow$ MAC Loads data from a general register to MAC.
STMAC	—	$MAC \rightarrow Rd$ Stores data from MAC to a general register.



**Table 2.7 Logic Operation Instructions**

Instruction	Size	Function
AND	B/W/L	$(EAd) \wedge \#IMM \rightarrow (EAd)$ , $(EAd) \wedge (EAs) \rightarrow (EAd)$ Performs a logical AND operation on data between immediate data, general registers, and memory.
OR	B/W/L	$(EAd) \vee \#IMM \rightarrow (EAd)$ , $(EAd) \vee (EAs) \rightarrow (EAd)$ Performs a logical OR operation on data between immediate data, general registers, and memory.
XOR	B/W/L	$(EAd) \oplus \#IMM \rightarrow (EAd)$ , $(EAd) \oplus (EAs) \rightarrow (EAd)$ Performs a logical exclusive OR operation on data between immediate data, general registers, and memory.
NOT	B/W/L	$\sim (EAd) \rightarrow (EAd)$ Takes the one's complement of the contents of a general register or a memory location.

**Table 2.8 Shift Operation Instructions**

Instruction	Size	Function
SHLL	B/W/L	$(EAd) \text{ (shift)} \rightarrow (EAd)$
SHLR		Performs a logical shift on the contents of a general register or a memory location.  The contents of a general register or a memory location can be shifted by 1, 2, 4, 8, or 16 bits. The contents of a general register can be shifted by any bits. In this case, the number of bits is specified by 5-bit immediate data or the lower 5 bits of the contents of a general register.
SHAL	B/W/L	$(EAd) \text{ (shift)} \rightarrow (EAd)$
SHAR		Performs an arithmetic shift on the contents of a general register or a memory location.  1-bit or 2-bit shift is possible.
ROTL	B/W/L	$(EAd) \text{ (rotate)} \rightarrow (EAd)$
ROTR		Rotates the contents of a general register or a memory location.  1-bit or 2-bit rotation is possible.
ROTXL	B/W/L	$(EAd) \text{ (rotate)} \rightarrow (EAd)$
ROTXR		Rotates the contents of a general register or a memory location with the carry bit.  1-bit or 2-bit rotation is possible.

**Table 2.9 Bit Manipulation Instructions**

<b>Instruction</b>	<b>Size</b>	<b>Function</b>
BSET	B	$1 \rightarrow (\text{<bit-No.> of <EAd>})$ Sets a specified bit in the contents of a general register or a memory location to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BSET/cc	B	if cc, $1 \rightarrow (\text{<bit-No.> of <EAd>})$ If the specified condition is satisfied, this instruction sets a specified bit in a memory location to 1. The bit number can be specified by 3-bit immediate data, or by the lower three bits of a general register. The Z flag status can be specified as a condition.
BCLR	B	$0 \rightarrow (\text{<bit-No.> of <EAd>})$ Clears a specified bit in the contents of a general register or a memory location to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BCLR/cc	B	if cc, $0 \rightarrow (\text{<bit-No.> of <EAd>})$ If the specified condition is satisfied, this instruction clears a specified bit in a memory location to 0. The bit number can be specified by 3-bit immediate data, or by the lower three bits of a general register. The Z flag status can be specified as a condition.
BNOT	B	$\sim (\text{<bit-No.> of <EAd>}) \rightarrow (\text{<bit-No.> of <EAd>})$ Inverts a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BTST	B	$\sim (\text{<bit-No.> of <EAd>}) \rightarrow Z$ Tests a specified bit in the contents of a general register or a memory location and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BAND	B	$C \wedge (\text{<bit-No.> of <EAd>}) \rightarrow C$ ANDs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIAND	B	$C \wedge [\sim (\text{<bit-No.> of <EAd>})] \rightarrow C$ ANDs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BOR	B	$C \vee (\text{<bit-No.> of <EAd>}) \rightarrow C$ ORs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.

Instruction	Size	Function
BIOR	B	$C \vee [\sim (\text{<bit-No.> of <EAd>})] \rightarrow C$ ORs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BXOR	B	$C \oplus (\text{<bit-No.> of <EAd>}) \rightarrow C$ Exclusive-ORs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIXOR	B	$C \oplus [\sim (\text{<bit-No.> of <EAd>})] \rightarrow C$ Exclusive-ORs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BLD	B	$(\text{<bit-No.> of <EAd>}) \rightarrow C$ Transfers a specified bit in the contents of a general register or a memory location to the carry flag. The bit number is specified by 3-bit immediate data.
BILD	B	$\sim (\text{<bit-No.> of <EAd>}) \rightarrow C$ Transfers the inverse of a specified bit in the contents of a general register or a memory location to the carry flag. The bit number is specified by 3-bit immediate data.
BST	B	$C \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the carry flag value to a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data.
BSTZ	B	$Z \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the zero flag value to a specified bit in the contents of a memory location. The bit number is specified by 3-bit immediate data.
BIST	B	$\sim C \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the inverse of the carry flag value to a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data.

Instruction	Size	Function
BISTZ	B	~ Z → (<bit-No.> of <EAd>) Transfers the inverse of the zero flag value to a specified bit in the contents of a memory location. The bit number is specified by 3-bit immediate data.
BFLD	B	(EAs) (bit field) → Rd Transfers a specified bit field in memory location contents to the lower bits of a specified general register.
BFST	B	Rs → (EAd) (bit field) Transfers the lower bits of a specified general register to a specified bit field in memory location contents.

**Table 2.10 Branch Instructions**

Instruction	Size	Function
BRA/BS BRA/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a specified address.
BSR/BS BSR/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a subroutine at a specified address.
Bcc	—	Branches to a specified address if the specified condition is satisfied.
BRA/S	—	Branches unconditionally to a specified address after executing the next instruction. The next instruction should be a 1-word instruction except for the block transfer and branch instructions.
JMP	—	Branches unconditionally to a specified address.
BSR	—	Branches to a subroutine at a specified address.
JSR	—	Branches to a subroutine at a specified address.
RTS	—	Returns from a subroutine.
RTS/L	—	Returns from a subroutine, restoring data from the stack to multiple general registers.

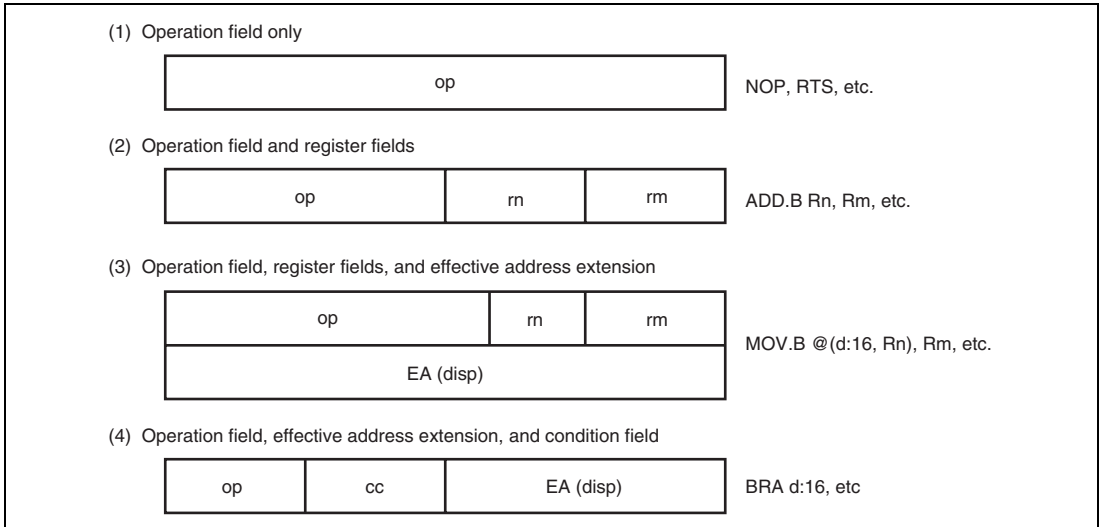
**Table 2.11 System Control Instructions**

<b>Instruction</b>	<b>Size</b>	<b>Function</b>
TRAPA	—	Starts trap-instruction exception handling.
RTE	—	Returns from an exception-handling routine.
RTE/L	—	Returns from an exception-handling routine, restoring data from the stack to multiple general registers.
SLEEP	—	Causes a transition to a power-down state.
LDC	B/W	#IMM → CCR, (EAs) → CCR, #IMM → EXR, (EAs) → EXR Loads immediate data or the contents of a general register or a memory location to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid.
	L	Rs → VBR, Rs → SBR Transfers the general register contents to VBR or SBR.
STC	B/W	CCR → (EAd), EXR → (EAd) Transfers the contents of CCR or EXR to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid.
	L	VBR → Rd, SBR → Rd Transfers the contents of VBR or SBR to a general register.
ANDC	B	CCR ∧ #IMM → CCR, EXR ∧ #IMM → EXR Logically ANDs the CCR or EXR contents with immediate data.
ORC	B	CCR ∨ #IMM → CCR, EXR ∨ #IMM → EXR Logically ORs the CCR or EXR contents with immediate data.
XORC	B	CCR ⊕ #IMM → CCR, EXR ⊕ #IMM → EXR Logically exclusive-ORs the CCR or EXR contents with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

### 2.7.3 Basic Instruction Formats

The H8SX CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

Figure 2.14 shows examples of instruction formats.



**Figure 2.14 Instruction Formats**

- **Operation Field**  
Indicates the function of the instruction, and specifies the addressing mode and operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.
- **Register Field**  
Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.
- **Effective Address Extension**  
8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.
- **Condition Field**  
Specifies the branch condition of Bcc instructions.

## 2.8 Addressing Modes and Effective Address Calculation

The H8SX CPU supports the 11 addressing modes listed in table 2.12. Each instruction uses a subset of these addressing modes.

Bit manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

**Table 2.12 Addressing Modes**

No.	Addressing Mode	Symbol
1	Register direct	Rn
2	Register indirect	@ERn
3	Register indirect with displacement	@(d:2,ERn)/@(d:16,ERn)/@(d:32,ERn)
4	Index register indirect with displacement	@(d:16, RnL.B)/@(d:16,Rn.W)/@(d:16,ERn.L) @(d:32, RnL.B)/@(d:32,Rn.W)/@(d:32,ERn.L)
5	Register indirect with post-increment	@ERn+
	Register indirect with pre-decrement	@-ERn
	Register indirect with pre-increment	@+ERn
	Register indirect with post-decrement	@ERn-
6	Absolute address	@aa:8/@aa:16/@aa:24/@aa:32
7	Immediate	#xx:3/#xx:4/#xx:8/#xx:16/#xx:32
8	Program-counter relative	@(d:8,PC)/@(d:16,PC)
9	Program-counter relative with index register	@(RnL.B,PC)/@(Rn.W,PC)/@(ERn.L,PC)
10	Memory indirect	@@aa:8
11	Extended memory indirect	@@vec:7

### 2.8.1 Register Direct—Rn

The operand value is the contents of an 8-, 16-, or 32-bit general register which is specified by the register field in the instruction code.

R0H to R7H and R0L to R7L can be specified as 8-bit registers.

R0 to R7 and E0 to E7 can be specified as 16-bit registers.

ER0 to ER7 can be specified as 32-bit registers.

### 2.8.2 Register Indirect—@ERn

The operand value is the contents of the memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code.

In advanced mode, if this addressing mode is used in a branch instruction, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

### 2.8.3 Register Indirect with Displacement —@(d:2, ERn), @(d:16, ERn), or @(d:32, ERn)

The operand value is the contents of a memory location which is pointed to by the sum of the contents of an address register (ERn) and a 16- or 32-bit displacement. ERn is specified by the register field of the instruction code. The displacement is included in the instruction code and the 16-bit displacement is sign-extended when added to ERn.

This addressing mode has a short format (@(d:2, ERn)). The short format can be used when the displacement is 1, 2, or 3 and the operand is byte data, when the displacement is 2, 4, or 6 and the operand is word data, or when the displacement is 4, 8, or 12 and the operand is longword data.

### 2.8.4 Index Register Indirect with Displacement—@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)

The operand value is the contents of a memory location which is pointed to by the sum of the following operation result and a 16- or 32-bit displacement: a specified bits of the contents of an address register (RnL, Rn, ERn) specified by the register field in the instruction code are zero-extended to 32-bit data and multiplied by 1, 2, or 4. The displacement is included in the instruction code and the 16-bit displacement is sign-extended when added to ERn. If the operand is byte data, ERn is multiplied by 1. If the operand is word or longword data, ERn is multiplied by 2 or 4, respectively.



### 2.8.5 Register Indirect with Post-Increment, Pre-Decrement, Pre-Increment, or Post-Decrement—@ERn+, @-ERn, @+ERn, or @ERn-

- Register indirect with post-increment—@ERn+

The operand value is the contents of a memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After the memory location is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access.
- Register indirect with pre-decrement—@-ERn

The operand value is the contents of a memory location which is pointed to by the following operation result: the value 1, 2, or 4 is subtracted from the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After that, the operand value is stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access.
- Register indirect with pre-increment—@+ERn

The operand value is the contents of a memory location which is pointed to by the following operation result: the value 1, 2, or 4 is added to the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After that, the operand value is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access.
- Register indirect with post-decrement—@ERn-

The operand value is the contents of a memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After the memory location is accessed, 1, 2, or 4 is subtracted from the address register contents and the remainder is stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access.

using this addressing mode, data to be written is the contents of the general register after calculating an effective address. If the same general register is specified in an instruction and two effective addresses are calculated, the contents of the general register after the first calculation of an effective address is used in the second calculation of an effective address.

#### Example 1:

```
MOV.W    R0, @ER0+
```

When ER0 before execution is H'12345678, H'567A is written at H'12345678.

**Example 2:**

MOV.B @ER0+, @ER0+

When ER0 before execution is H'00001000, H'00001000 is read and the contents is written at H'00001001.

After execution, ER0 is H'00001002.

**2.8.6 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32**

The operand value is the contents of a memory location which is pointed to by an absolute address included in the instruction code.

There are 8-bit (@aa:8), 16-bit (@aa:16), 24-bit (@aa:24), and 32-bit (@aa:32) absolute addresses.

To access the data area, the absolute address of 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) is used. For an 8-bit absolute address, the upper 24 bits are specified by SBR. For a 16-bit absolute address, the upper 16 bits are sign-extended. A 32-bit absolute address can access the entire address space.

To access the program area, the absolute address of 24 bits (@aa:24) or 32 bits (@aa:32) is used. For a 24-bit absolute address, the upper 8 bits are all assumed to be 0 (H'00).

Table 2.13 shows the accessible absolute address ranges.

**Table 2.13 Absolute Address Access Ranges**

<b>Absolute Address</b>	<b>Normal Mode</b>	<b>Middle Mode</b>	<b>Advanced Mode</b>	<b>Maximum Mode</b>
Data area	A consecutive 256-byte area (the upper address is set in SBR)			
8 bits (@aa:8)				
16 bits (@aa:16)	H'0000 to H'FFFF	H'000000 to H'007FFF,	H'00000000 to H'00007FFF, H'FFFF8000 to H'FFFFFFFF	
32 bits (@aa:32)	H'FF8000 to H'FFFFFF		H'00000000 to H'FFFFFFFF	
Program area				
24 bits (@aa:24)	H'000000 to H'FFFFFF		H'00000000 to H'00FFFFFF	
32 bits (@aa:32)			H'00000000 to H'00FFFFFF	H'00000000 to H'FFFFFFFF

### 2.8.7 Immediate—#xx

The operand value is 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) data included in the instruction code.

This addressing mode has short formats in which 3- or 4-bit immediate data can be used.

When the size of immediate data is less than that of the destination operand value (byte, word, or longword) the immediate data is zero-extended.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, for specifying a bit number. The BFLD and BFST instructions contain 8-bit immediate data in the instruction code, for specifying a bit field. The TRAPA instruction contains 2-bit immediate data in the instruction code, for specifying a vector address.

### 2.8.8 Program-Counter Relative—@(d:8, PC) or @(d:16, PC)

This mode is used in the Bcc and BSR instructions. The operand value is a 32-bit branch address, which is the sum of an 8- or 16-bit displacement in the instruction code and the 32-bit address of the PC contents. The 8-bit or 16-bit displacement is sign-extended to 32 bits when added to the PC contents. The PC contents to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is -126 to +128 bytes (-63 to +64 words) or -32766 to +32768 bytes (-16383 to +16384 words) from the branch instruction. The resulting value should be an even number. In advanced mode, only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00).

### 2.8.9 Program-Counter Relative with Index Register—@(RnL.B, PC), @(Rn.W, PC), or @(ERn.L, PC)

This mode is used in the Bcc and BSR instructions. The operand value is a 32-bit branch address, which is the sum of the following operation result and the 32-bit address of the PC contents: the contents of an address register specified by the register field in the instruction code (RnL, Rn, or ERn) is zero-extended and multiplied by 2. The PC contents to which the displacement is added is the address of the first byte of the next instruction. In advanced mode, only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00).

### 2.8.10 Memory Indirect—@@aa:8

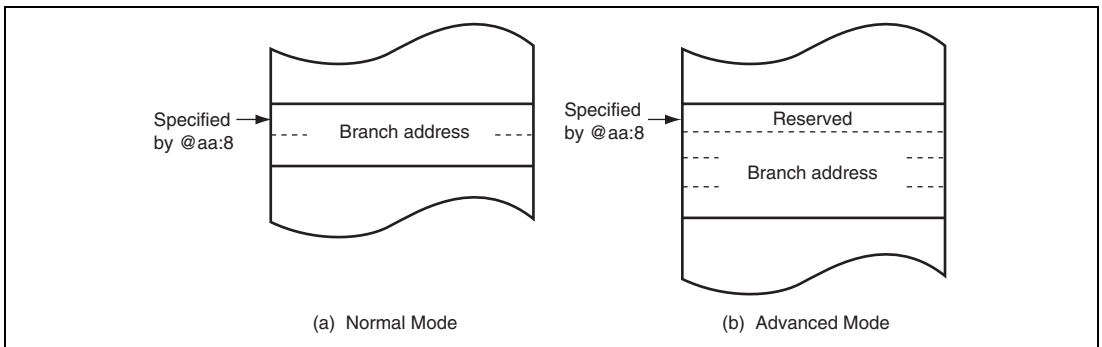
This mode can be used by the JMP and JSR instructions. The operand value is a branch address, which is the contents of a memory location pointed to by an 8-bit absolute address in the instruction code.

The upper bits of an 8-bit absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in other modes).

In normal mode, the memory location is pointed to by word-size data and the branch address is 16 bits long. In other modes, the memory location is pointed to by longword-size data. In middle or advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

Note that the top part of the address range is also used as the exception handling vector area. A vector address of an exception handling other than a reset or a CPU address error can be changed by VBR.

Figure 2.15 shows an example of specification of a branch address using this addressing mode.



**Figure 2.15 Branch Address Specification in Memory Indirect Mode**

### 2.8.11 Extended Memory Indirect—@@vec:7

This mode can be used by the JMP and JSR instructions. The operand value is a branch address, which is the contents of a memory location pointed to by the following operation result: the sum of 7-bit data in the instruction code and the value of H'80 is multiplied by 2 or 4.

The address range to store a branch address is H'0100 to H'01FF in normal mode and H'000200 to H'0003FF in other modes. In assembler notation, an address to store a branch address is specified.

In normal mode, the memory location is pointed to by word-size data and the branch address is 16 bits long. In other modes, the memory location is pointed to by longword-size data. In middle or advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

### 2.8.12 Effective Address Calculation

Tables 2.14 and 2.15 show how effective addresses are calculated in each addressing mode. The lower bits of the effective address are valid and the upper bits are ignored (zero extended or sign extended) according to the CPU operating mode.





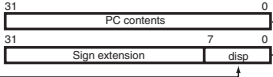
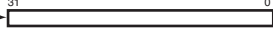
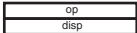
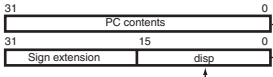
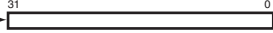

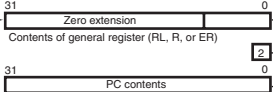


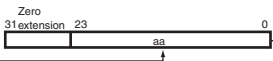
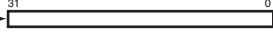

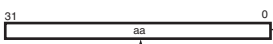
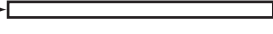

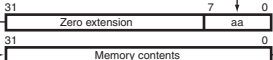


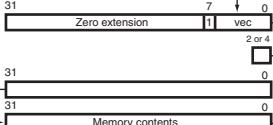

The valid bits in middle mode are as follows:

- The lower 16 bits of the effective address are valid and the upper 16 bits are sign-extended for the transfer and operation instructions.
- The lower 24 bits of the effective address are valid and the upper eight bits are zero-extended for the branch instructions.

Table 2.14 Effective Address Calculation for Transfer and Operation Instructions

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Immediate 		
2	Register direct 		
3	Register indirect 		
4	Register indirect with 16-bit displacement 		
	Register indirect with 32-bit displacement 		
5	Index register indirect with 16-bit displacement 		
	Index register indirect with 32-bit displacement 		
6	Register indirect with post-increment or post-decrement 		
	Register indirect with pre-increment or pre-decrement 		
7	8-bit absolute address 		
	16-bit absolute address 		
	32-bit absolute address 		

**Table 2.15 Effective Address Calculation for Branch Instructions**

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Register indirect 		
2	Program-counter relative with 8-bit displacement 		
	Program-counter relative with 16-bit displacement 		
3	Program-counter relative with index register 		
4	24-bit absolute address 		
	32-bit absolute address 		
5	Memory indirect 		
6	Extended memory indirect 		

### 2.8.13 MOVA Instruction

The MOVA instruction stores the effective address in a general register.

1. Firstly, data is obtained by the addressing mode shown in item 2 of table 2.14.
2. Next, the effective address is calculated using the obtained data as the index by the addressing mode shown in item 5 of table 2.14. The obtained data is used instead of the general register. The result is stored in a general register. For details, see H8SX Family Software Manual.

## 2.9 Processing States

The H8SX CPU has five main processing states: the reset state, exception-handling state, program execution state, bus-released state, and program stop state. Figure 2.16 indicates the state transitions.

- Reset state

In this state the CPU and internal peripheral modules are all initialized and stopped. When the  $\overline{\text{RES}}$  input goes low, all current processing stops and the CPU enters the reset state. All interrupts are masked in the reset state. Reset exception handling starts when the  $\overline{\text{RES}}$  signal changes from low to high. For details, see section 4, Resets and section 5, Exception Handling.

- Exception-handling state

The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to activation of an exception source, such as, a reset, trace, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception handling vector table and branches to that address. For further details, see section 4, Resets and section 5, Exception Handling.

- Program execution state

In this state the CPU executes program instructions in sequence.

- Bus-released state

The bus-released state occurs when the bus has been released in response to a bus request from a bus master other than the CPU. While the bus is released, the CPU halts operations.

- Program stop state

This is a power-down state in which the CPU stops operating. The program stop state occurs when a SLEEP instruction is executed or the CPU enters hardware standby mode. For details, see section 24, Power-Down Modes.

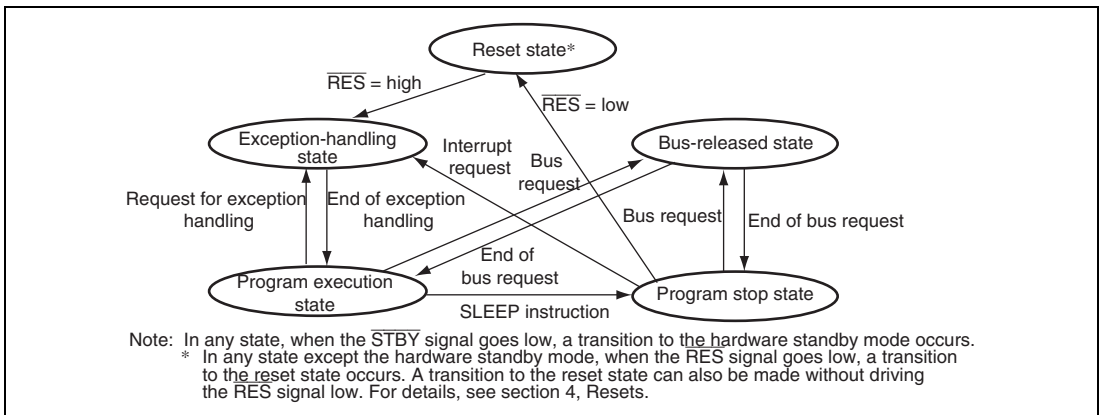


Figure 2.16 State Transitions



## Section 3 MCU Operating Modes

### 3.1 Operating Mode Selection

This LSI has six operating modes (modes 1, 2, 4, 5, 6, and 7). The operating mode is selected by the setting of mode pins MD2 to MD0. Table 3.1 lists MCU operating mode settings.

**Table 3.1 MCU Operating Mode Settings**

MCU Operating Mode	MD2	MD1	MD0	CPU Operating Mode	Address Space	LSI Initiation Mode	On-Chip ROM	External Data Bus Width	
								Default	Max.
1	0	0	1	Advanced mode	16 Mbytes	User boot mode	Enabled	—	16 bits
2	0	1	0			Boot mode	Enabled	—	16 bits
4	1	0	0			On-chip ROM disabled extended mode	Disabled	16 bits	16 bits
5	1	0	1			Disabled	8 bits	16 bits	
6	1	1	0			On-chip ROM enabled extended mode	Enabled	8 bits	16 bits
7	1	1	1			Single-chip mode	Enabled	—	16 bits

In this LSI, an advanced mode as the CPU operating mode and a 16-Mbyte address space are available. The initial external bus widths are eight or 16 bits. As the LSI initiation mode, the external extended mode, on-chip ROM initiation mode, or single-chip initiation mode can be selected.

Modes 1 and 2 are the user boot mode and the boot mode, respectively, in which the flash memory can be programmed and erased. For details on the user boot mode and boot mode, see section 22, Flash Memory.

Mode 7 is a single-chip initiation mode. All I/O ports can be used as general input/output ports. The external address space cannot be accessed in the initial state, but setting the EXPE bit in the system control register (SYSCR) to 1 enables to use the external address space. After the external address space is enabled, ports H and I can be used as a data bus, and ports D, E, and F can be used as an address output bus by specifying the data direction register (DDR) for each port.

Modes 4 to 6 are external extended modes, in which the external memory and devices can be accessed. In the external extended modes, the external address space can be designated as 8-bit or 16-bit address space for each area by the bus controller after starting program execution.

If 16-bit address space is designated for any one area, it is called the 16-bit bus widths mode. If 8-bit address space is designated for all areas, it is called the 8-bit bus width mode.

## 3.2 Register Descriptions

The following registers are related to the operating mode setting.

- Mode control register (MDCR)
- System control register (SYSCR)

### 3.2.1 Mode Control Register (MDCR)

MDCR indicates the current operating mode. When MDCR is read from, the states of signals MD2 to MD0 are latched. Latching is released by a reset.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	MDS3	MDS2	MDS1	MDS0
Initial Value	0	1	0	1	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R	R
Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	Undefined*	1	0	1	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R	R

Note: \* Determined by pins MD2 to MD0.

Bit	Bit Name	Initial Value	R/W	Descriptions
15	—	0	R	Reserved
14	—	1	R	These are read-only bits and cannot be modified.
13	—	0	R	
12	—	1	R	
11	MDS3	Undefined*	R	Mode Select 3 to 0
10	MDS2	Undefined*	R	These bits indicate the operating mode selected by the mode pins (MD2 to MD0) (see table 3.2). When MDCR is read, the signal levels input on pins MD2 to MD0 are latched into these bits. These latches are released by a reset.
9	MDS1	Undefined*	R	
8	MDS0	Undefined*	R	
7	—	Undefined*	R	Reserved
6	—	1	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	—	1	R	
3	—	Undefined*	R	
2	—	Undefined*	R	
1	—	Undefined*	R	
0	—	Undefined*	R	

Note: \* Determined by pins MD2 to MD0.

**Table 3.2 Settings of Bits MDS3 to MDS0**

MCU Operating Mode	Mode Pins			MDCR			
	MD2	MD1	MD0	MDS3	MDS2	MDS1	MDS0
1	0	0	1	1	1	0	1
2	0	1	0	1	1	0	0
4	1	0	0	0	0	1	0
5	1	0	1	0	0	0	1
6	1	1	0	0	1	0	1
7	1	1	1	0	1	0	0

### 3.2.2 System Control Register (SYSCR)

SYSCR controls MAC saturation operation, selects bus width mode for instruction fetch, sets external bus mode, enables/disables the on-chip RAM, and selects the DTC address mode.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	MACS	—	FETCHMD	—	EXPE	RAME
Initial Value	1	1	0	1	0	Undefined*	Undefined*	1
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	DTCMD	—
Initial Value	0	0	0	0	0	0	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* The initial value depends on the startup mode.

Bit	Bit Name	Initial Value	R/W	Descriptions
15	—	1	R/W	Reserved
14	—	1	R/W	These bits are always read as 1. The write value should always be 1.
13	MACS	0	R/W	<b>MAC Saturation Operation Control</b> Selects either saturation operation or non-saturation operation for the MAC instruction. 0: MAC instruction is non-saturation operation 1: MAC instruction is saturation operation
12	—	1	R/W	<b>Reserved</b> This bit is always read as 1. The write value should always be 1.
11	FETCHMD	0	R/W	<b>Instruction Fetch Mode Select</b> This LSI can prefetch an instruction in units of 16 bits or 32 bits. Select the bus width for instruction fetch depending on the used memory for the storage of programs* <sup>1</sup> . 0: 32-bit mode 1: 16-bit mode

Bit	Bit Name	Initial Value	R/W	Descriptions
10	—	Undefined* <sup>2</sup>	R	Reserved  This bit is fixed at 1 in on-chip ROM enabled mode, and 0 in on-chip ROM disabled mode. This bit cannot be changed.
9	EXPE	Undefined* <sup>2</sup>	R/W	External Bus Mode Enable  Selects external bus mode. In external extended mode, this bit is fixed at 1 and cannot be changed. In single-chip mode, the initial value of this bit is 0, and can be read from or written to.  When writing 0 to this bit after reading EXPE = 1, an external bus cycle should not be executed.  The external bus cycle may be carried out in parallel with the internal bus cycle depending on the setting of the write data buffer function.  0: External bus disabled 1: External bus enabled
8	RAME	1	R/W	RAM Enable  Enables or disables the on-chip RAM. This bit is initialized when the reset state is released. Do not write 0 during access to the on-chip RAM.  0: On-chip RAM disabled 1: On-chip RAM enabled
7 to 2	—	All 0	R/W	Reserved  These bits are always read as 0. The write value should always be 0.
1	DTCMD	1	R/W	DTC Mode Select  Selects DTC operating mode.  0: DTC is in full-address mode 1: DTC is in short address mode
0	—	1	R/W	Reserved  This bit is always read as 1. The write value should always be 1.

- Notes: 1. For details on instruction fetch mode, see section 2.3, Instruction Fetch.  
2. The initial value depends on the LSI initiation mode.  
EXPE = 1 because operating modes 4, 5, and 6 are external extended modes.

## 3.3 Operating Mode Descriptions

### 3.3.1 Mode 1

This is the user boot mode for the flash memory. The LSI operates in the same way as in mode 7 except for programming and erasing of the flash memory. For details, see section 22, Flash Memory.

### 3.3.2 Mode 2

This is the boot mode for the flash memory. The LSI operates in the same way as in mode 7 except for programming and erasing of the flash memory. For details, see section 22, Flash Memory.

### 3.3.3 Mode 4

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is disabled.

The initial bus width mode immediately after a reset is 16 bits, with 16-bit access to all areas. Ports D, E, and F function as an address bus, ports H and I function as a data bus, and parts of port A function as bus control signals. However, if all areas are designated as an 8-bit access space by the bus controller, the bus mode switches to eight bits, and only port H functions as a data bus.

### 3.3.4 Mode 5

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is disabled.

The initial bus width mode immediately after a reset is eight bits, with 8-bit access to all areas. Ports D, E, and F function as an address bus, port H functions as a data bus, and parts of port A function as bus control signals. However, if any area is designated as a 16-bit access space by the bus controller, the bus width mode switches to 16 bits, and ports H and I function as a data bus.

### 3.3.5 Mode 6

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is enabled.

The initial bus width mode immediately after a reset is eight bits, with 8-bit access to all areas. Ports D, E, and F function as input ports, but they can be used as an address bus by specifying the data direction register (DDR) for each port. For details, see section 11, I/O Ports. Port H functions as a data bus, and parts of port A function as bus control signals. However, if any area is designated as a 16-bit access space by the bus controller, the bus width mode switches to 16 bits, and ports H and I function as a data bus.

### 3.3.6 Mode 7

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the on-chip ROM is enabled.

All I/O ports can be used as general input/output ports. The external address space cannot be accessed in the initial state, but setting the EXPE bit in the system control register (SYSCR) to 1 enables the external address space. After the external address space is enabled, ports H and I can be used as a data bus, and ports D, E, and F can be used as an address output bus by specifying the data direction register (DDR) for each port. For details, see section 11, I/O Ports.

### 3.3.7 Pin Functions

Table 3.3 lists the pin functions in each operating mode.

**Table 3.3 Pin Functions in Each Operating Mode (Advanced Mode)**

Port		Mode 1	Mode 2	Mode 4	Mode 5	Mode 6	Mode 7
Port A	PA7	P*/C	P*/C	P/C*	P/C*	P/C*	P*/C
	PA6 to PA3	P*/C	P*/C	P/C*	P/C*	P/C*	P*/C
	PA2 to PA0	P*/C	P*/C	P*/C	P*/C	P*/C	P*/C
Port 3	P33 to P31	P*/C	P*/C	P*/C	P*/C	P*/C	P*/C
	P30	P*/C	P*/C	P/C*	P/C*	P*/C	P*/C
Port D		P*/A	P*/A	A	A	P*/A	P*/A
Port E		P*/A	P*/A	A	A	P*/A	P*/A
Port F	PF4 to PF0	P*/A	P*/A	P/A*	P/A*	P*/A	P*/A
Port H		P*/D	P*/D	D	D	D	P*/D
Port I		P*/D	P*/D	P/D*	P*/D	P*/D	P*/D

[Legend]

P: I/O port

A: Address bus output

D: Data bus input/output

C: Control signals, clock input/output

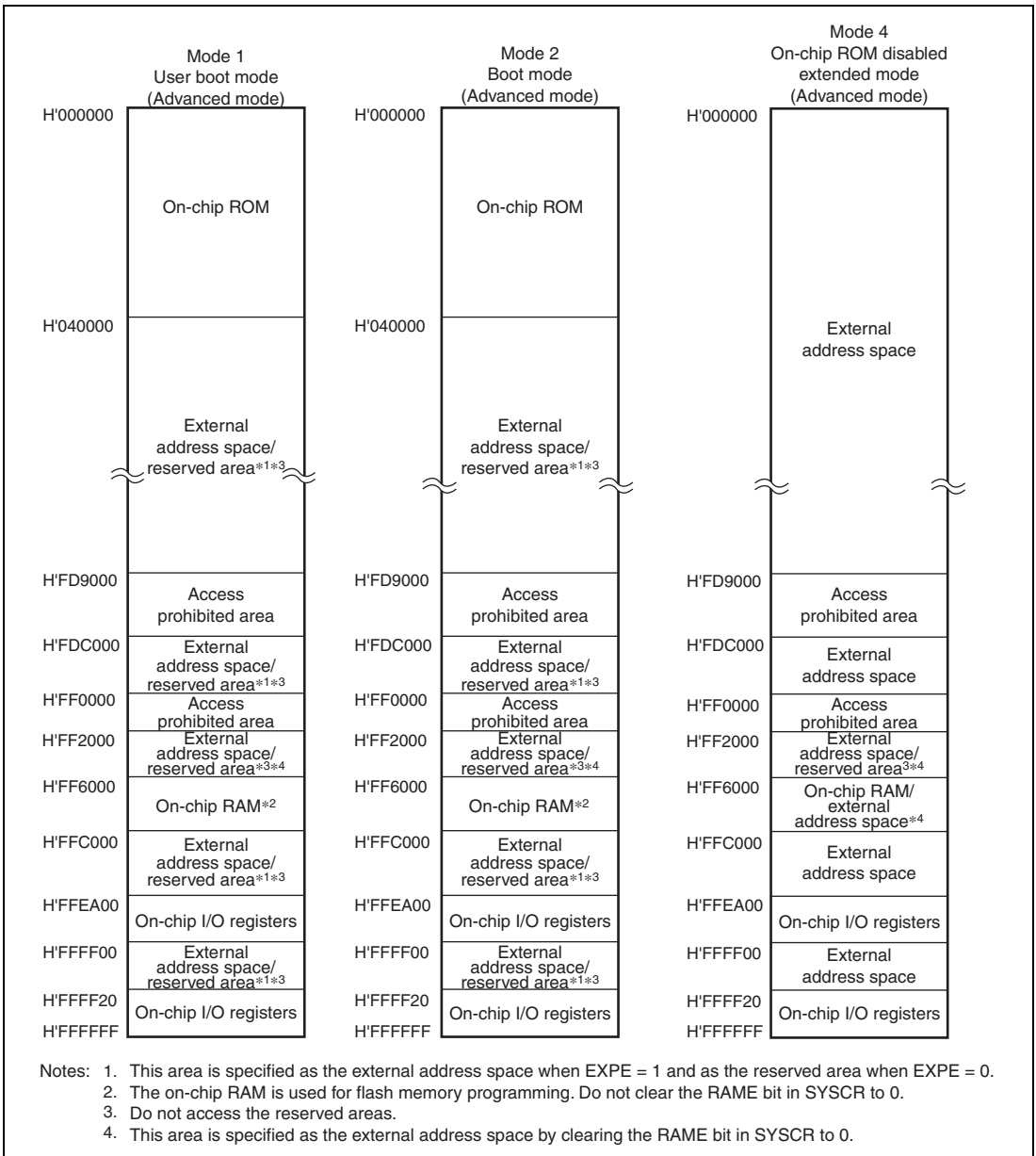
\*: Immediately after a reset

## 3.4 Address Map

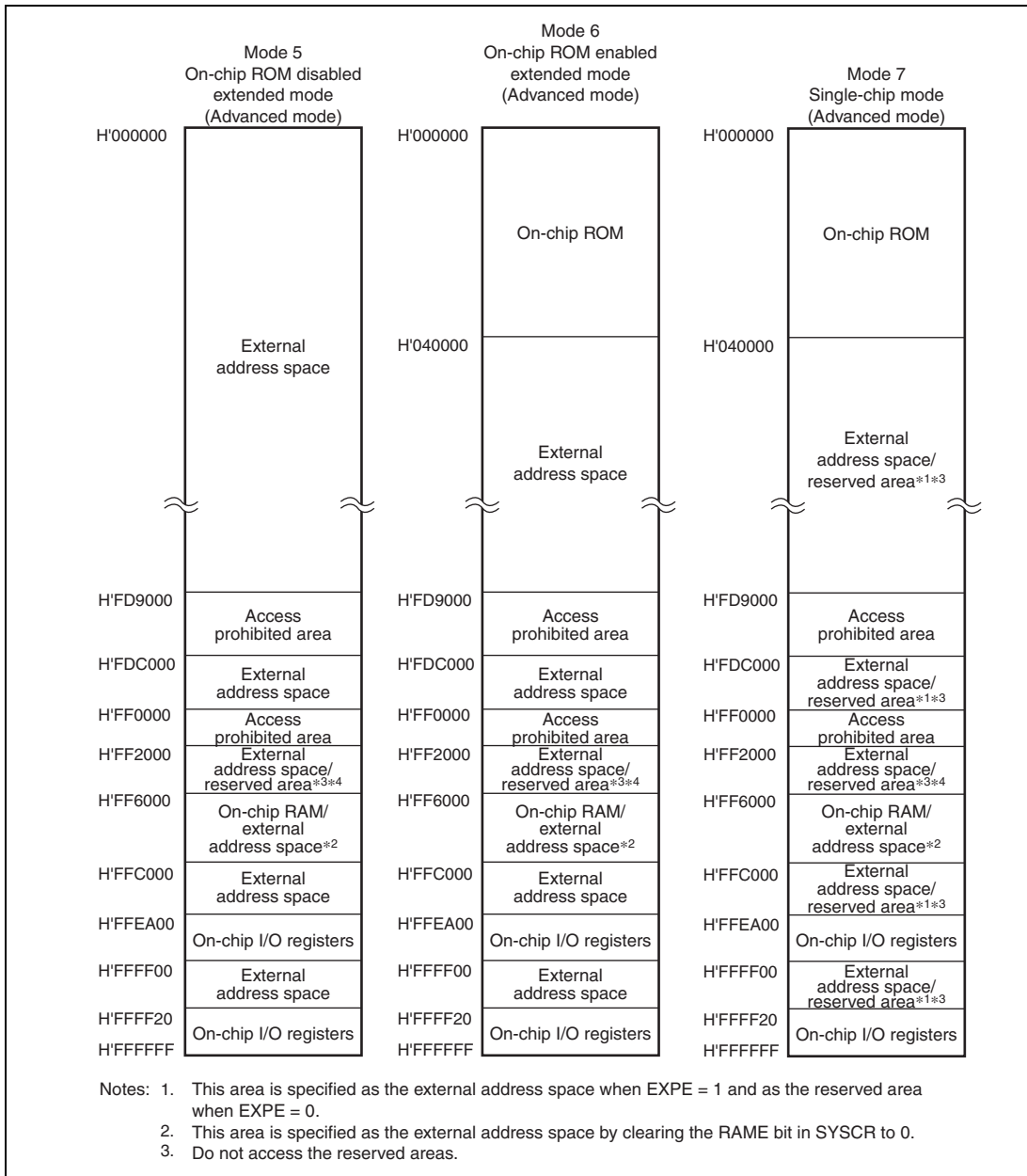
### 3.4.1 Address Map

Figures 3.1 and 3.2 show the address map in each operating mode.





**Figure 3.1 Address Map in Each Operating Mode of H8SX/1622 (1)**



**Figure 3.1 Address Map in Each Operating Mode of H8SX/1622 (2)**

## Section 4 Resets

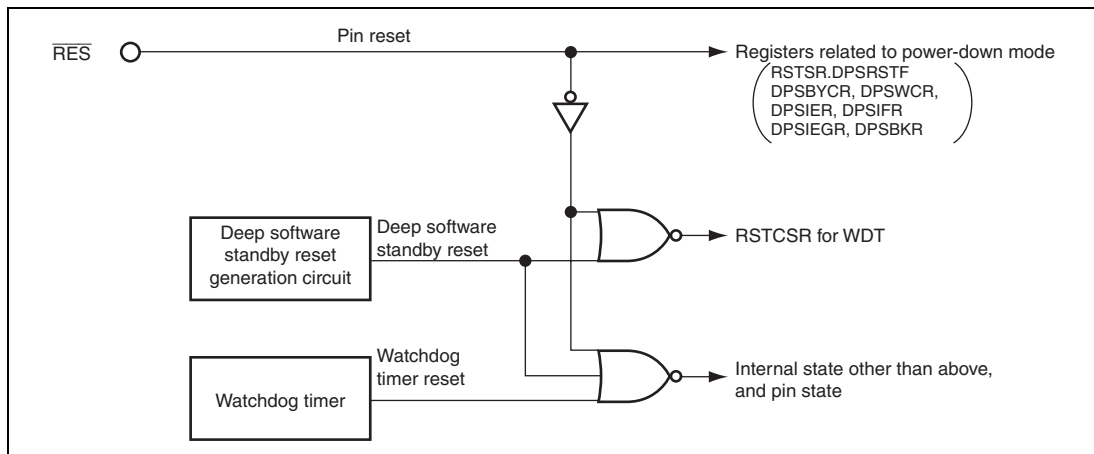
### 4.1 Types of Resets

There are three types of resets: a pin reset, deep software standby reset, and watchdog timer reset. Table 4.1 shows the reset names and sources.

The internal state and pins are initialized by a reset. Figure 4.1 shows the reset targets to be initialized.

**Table 4.1 Reset Names And Sources**

Reset Name	Source
Pin reset	Voltage input to the $\overline{\text{RES}}$ pin is driven low.
Deep software standby reset	Deep software standby mode is canceled by an interrupt.
Watchdog timer reset	The watchdog timer overflows.



**Figure 4.1 Block Diagram of Reset Circuit**

Note that some registers are not initialized by any of the resets. The following describes the CPU internal registers.

The PC, one of the CPU internal registers, is initialized by loading the start address from vector addresses with the reset exception handling. At this time, the T bit in EXR is cleared to 0 and the I bits in EXR and CCR are set to 1. The general registers, MAC, and other bits in CCR are not initialized.

The initial value of the SP (ER7) is undefined. The SP should be initialized using the MOV.L instruction immediately after a reset. For details, see section 2, CPU. For other registers that are not initialized by a reset, see register descriptions in each section.

When a reset is canceled, the reset exception handling is started. For the reset exception handling, see section 5.3, Reset.

## 4.2 Input/Output Pin

Table 4.2 shows the pin related to resets.

**Table 4.2 Pin Configuration**

<b>Pin Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
Reset	$\overline{\text{RES}}$	Input	Reset input

## 4.3 Register Descriptions

This LSI has the following registers for resets.

- Reset status register (RSTSR)
- Reset control/status register (RSTCSR)

### 4.3.1 Reset Status Register (RSTSR)

RSTSR indicates a source for generating an internal reset.

Bit	7	6	5	4	3	2	1	0
Bit name	DPSRSTF	—	—	—	—	—	—	—
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.

Bit	Bit Name	Initial Value	R/W	Description
7	DPSRSTF	0	R/(W)*	<p>Deep Software Standby Reset Flag</p> <p>Indicates that deep software standby mode is canceled by an external interrupt source specified with DPSIER or DPSIEGR and an internal reset is generated.</p> <p>[Setting condition]</p> <p>When deep software standby mode is canceled by an external interrupt source.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When this bit is read as 1 and then written by 0.</li> <li>• When a pin reset is generated.</li> </ul>
6 to 0	—	All 0	R/W	<p>Reserved</p> <p>These bits are always read as 0. The write value should always be 0.</p>

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

### 4.3.2 Reset Control/Status Register (RSTCSR)

RSTCSR controls an internal reset signal generated by the watchdog timer and selects the internal reset signal type. RSTCSR is initialized to H'1F by a pin reset or a deep software standby reset, but not by the internal reset signal generated by a WDT overflow.

Bit	7	6	5	4	3	2	1	0
Bit name	WOVF	RSTE	—	—	—	—	—	—
Initial value:	0	0	0	1	1	1	1	1
R/W:	R/(W)*	R/W	R/W	R	R	R	R	R

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

Bit	Bit Name	Initial Value	R/W	Description
7	WOVF	0	R/(W)*	<p>Watchdog Timer Overflow Flag</p> <p>This bit is set when TCNT overflows in watchdog timer mode, but not set in interval timer mode. Only 0 can be written to.</p> <p>[Setting condition]</p> <p>When TCNT overflows (H'FF → H'00) in watchdog timer mode.</p> <p>[Clearing condition]</p> <p>When this bit is read as 1 and then written by 0.</p> <p>(The flag must be read after writing of 0, when this bit is cleared by the CPU using an interrupt.)</p>
6	RSTE	0	R/W	<p>Reset Enable</p> <p>Selects whether or not the LSI internal state is reset by a TCNT overflow in watchdog timer mode.</p> <p>0: Internal state is not reset when TCNT overflows. (Although this LSI internal state is not reset, TCNT and TCSR of the WDT are reset.)</p> <p>1: Internal state is reset when TCNT overflows.</p>
5	—	0	R/W	<p>Reserved</p> <p>Although this bit is readable/writable, operation is not affected by this bit.</p>
4 to 0	—	1	R	<p>Reserved</p> <p>These are read-only bits but cannot be modified.</p>

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

## 4.4 Pin Reset

This is a reset generated by the  $\overline{\text{RES}}$  pin.

When the  $\overline{\text{RES}}$  pin is driven low, all the processing in progress is aborted and the LSI enters a reset state. In order to firmly reset the LSI, the  $\overline{\text{STBY}}$  pin should be set to high and the  $\overline{\text{RES}}$  pin should be held low at least for 20 ms at a power-on. During operation, the  $\overline{\text{RES}}$  pin should be held low at least for 20 states.

## 4.5 Deep Software Standby Reset

This is an internal reset generated when deep software standby mode is canceled by an interrupt.

When deep software standby mode is canceled, a deep software standby reset is generated, and simultaneously, clock oscillation starts. After the time specified with DPSWCR has elapsed, the deep software standby reset is canceled.

For details of the deep software standby reset, see section 24, Power-Down Modes.

## 4.6 Watchdog Timer Reset

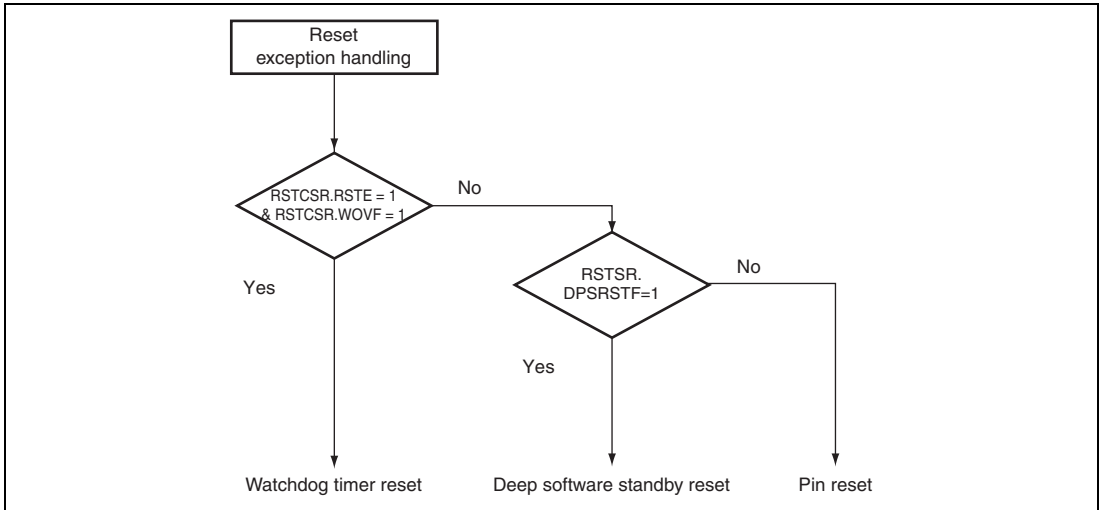
This is an internal reset generated by the watchdog timer.

When the RSTE bit in RSTCSR is set to 1, a watchdog timer reset is generated by a TCNT overflow. After a certain time, the watchdog timer reset is canceled.

For details of the watchdog timer reset, see section 15, Watchdog Timer (WDT).

## 4.7 Determination of Reset Generation Source

Reading RSTCSR and RSTSR determines which reset was used to execute the reset exception handling. Figure 4.2 shows an example the flow to identify a reset generation source.



**Figure 4.2 Example of Reset Generation Source Determination Flow**




## Section 5 Exception Handling

### 5.1 Exception Handling Types and Priority

As table 5.1 indicates, exception handling is caused by a reset, a trace, an address error, an interrupt, a trap instruction, and an illegal instruction (general illegal instruction or slot illegal instruction). Exception handling is prioritized as shown in table 5.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Exception sources, the stack structure, and operation of the CPU vary depending on the interrupt control mode. For details on the interrupt control mode, see section 6, Interrupt Controller.

**Table 5.1 Exception Types and Priority**

Priority	Exception Type	Exception Handling Start Timing
High  Low	Reset	Exception handling starts at the timing of low-to-high transition on the $\overline{\text{RES}}$ pin, watchdog timer overflow, or input of an external interrupt signal* <sup>4</sup> in deep standby mode. The CPU enters the reset state when the $\overline{\text{RES}}$ pin is low.
	Illegal instruction	Exception handling starts when an undefined code is executed.
	Trace* <sup>1</sup>	Exception handling starts after execution of the current instruction or exception handling when the trace (T) bit in EXR has been set to 1,
	Address error	After an address error has occurred, exception handling starts on completion of instruction execution.
	Interrupt	When an interrupt request has occurred, exception handling starts after execution of the current instruction or exception handling.* <sup>2</sup>
	Sleep instruction	Exception handling starts by execution of a sleep instruction (SLEEP) when the SSBY bit in SBYCR has been cleared to 0 and the SLPIE bit in SBYCR has been set to 1.
	Trap instruction* <sup>3</sup>	Exception handling starts by execution of a trap instruction (TRAPA).

- Notes:
- Traces are enabled only in interrupt control mode 2. Trace exception handling is not executed after execution of an RTE instruction.
  - Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
  - Trap instruction exception handling requests and sleep instruction exception handling requests are accepted at all times in program execution state.
  - The external interrupt input pins usable in deep software standby mode are IRQ3 to IRQ0 (IRQnA pins only) and NMI.

## 5.2 Exception Sources and Exception Handling Vector Table

Different vector table address offsets are assigned to different exception sources. The vector table addresses are calculated from the contents of the vector base register (VBR) and vector table address offset of the vector number. The start address of the exception service routine is fetched from the exception handling vector table indicated by this vector table address.

Table 5.2 shows the correspondence between the exception sources and vector table address offsets. Table 5.3 shows the calculation method of exception handling vector table addresses.

Since the usable modes differ depending on the product, for details on the available modes, see section 3, MCU Operating Modes.

**Table 5.2 Exception Handling Vector Table**

Exception Source	Vector Number	Vector Table Address Offset* <sup>1</sup>		
		Normal Mode* <sup>2</sup>	Advanced, Middle* <sup>2</sup> , Maximum* <sup>2</sup> Modes	
Reset	0	H'0000 to H'0001	H'0000 to H'0003	
Reserved for system use	1	H'0002 to H'0003	H'0004 to H'0007	
	2	H'0004 to H'0005	H'0008 to H'000B	
	3	H'0006 to H'0007	H'000C to H'000F	
Illegal instruction	4	H'0008 to H'0009	H'0010 to H'0013	
Trace	5	H'000A to H'000B	H'0014 to H'0017	
Reserved for system use	6	H'000C to H'000D	H'0018 to H'001B	
Interrupt (NMI)	7	H'000E to H'000F	H'001C to H'001F	
Trap instruction	(#0)	8	H'0010 to H'0011	H'0020 to H'0023
	(#1)	9	H'0012 to H'0013	H'0024 to H'0027
	(#2)	10	H'0014 to H'0015	H'0028 to H'002B
	(#3)	11	H'0016 to H'0017	H'002C to H'002F
CPU address error	12	H'0018 to H'0019	H'0030 to H'0033	
DMA address error* <sup>3</sup>	13	H'001A to H'001B	H'0034 to H'0037	
UBC break interrupt	14	H'001C to H'001D	H'0038 to H'003B	
Reserved for system use	15	H'001E to H'001F	H'003C to H'003F	
	17	H'0022 to H'0023	H'0044 to H'0047	
Sleep instruction	18	H'0024 to H'0025	H'0048 to H'004B	

Exception Source	Vector Number	Vector Table Address Offset* <sup>1</sup>		
		Normal Mode* <sup>2</sup>	Advanced, Middle* <sup>2</sup> , Maximum* <sup>2</sup> Modes	
Reserved for system use	19	H'0026 to H'0027	H'004C to H'004F	
	23	H'002E to H'002F	H'005C to H'005F	
User area (open space)	24	H'0030 to H'0031	H'0060 to H'0063	
	63	H'007E to H'007F	H'00FC to H'00FF	
External interrupt	IRQ0	64	H'0080 to H'0081	H'0100 to H'0103
	IRQ1	65	H'0082 to H'0083	H'0104 to H'0107
	IRQ2	66	H'0084 to H'0085	H'0108 to H'010B
	IRQ3	67	H'0086 to H'0087	H'010C to H'010F
	IRQ4	68	H'0088 to H'0089	H'0110 to H'0113
	IRQ5	69	H'008A to H'008B	H'0114 to H'0117
	IRQ6	70	H'008C to H'008D	H'0118 to H'011B
	IRQ7	71	H'008E to H'008F	H'011C to H'011F
	IRQ8	72	H'0090 to H'0091	H'0120 to H'0123
	IRQ9	73	H'0092 to H'0093	H'0124 to H'0127
	IRQ10	74	H'0094 to H'0095	H'0128 to H'012B
	IRQ11	75	H'0096 to H'0097	H'012C to H'012F
	IRQ12	76	H'0098 to H'0099	H'0130 to H'0133
	IRQ13	77	H'009A to H'009B	H'0134 to H'0137
	IRQ14	78	H'009C to H'009D	H'0138 to H'013B
	Internal interrupt* <sup>4</sup>	80	H'00A0 to H'00A1	H'0140 to H'0143
255		H'01FE to H'01FF	H'03FC to H'03FF	

- Notes:
1. Lower 16 bits of the address.
  2. Not available in this LSI.
  3. A DMA address error is generated by the DTC and DMAC.
  4. For details of internal interrupt vectors, see section 6.5, Interrupt Exception Handling Vector Table.

**Table 5.3 Calculation Method of Exception Handling Vector Table Address**

Exception Source	Calculation Method of Vector Table Address
Reset, CPU address error	Vector table address = (vector table address offset)
Other than above	Vector table address = VBR + (vector table address offset)

[Legend]

VBR: Vector base register

Vector table address offset: See table 5.2.

### 5.3 Reset

A reset has priority over any other exception. When the  $\overline{\text{RES}}$  pin goes low, all processing halts and this LSI enters the reset state. To ensure that this LSI is reset, hold the  $\overline{\text{RES}}$  pin low for at least 20 ms with the  $\overline{\text{STBY}}$  pin driven high when the power is turned on. When operation is in progress, hold the  $\overline{\text{RES}}$  pin low for at least 20 cycles.

In addition to the  $\overline{\text{RES}}$  pin, it is also possible to establish the reset state by two operations in the internal circuit. One of them is to use an overflow in the watchdog timer. The other is to use an external interrupt during deep software standby mode. For details, see section 4, Resets, section 15, Watchdog Timer (WDT), and section 24, Power-Down Modes.

A reset initializes the internal state of the CPU and the registers of the on-chip peripheral modules. The interrupt control mode is 0 immediately after a reset. However, there are registers that will not be initialized by issuing an internal reset based on the watchdog timer or by issuing an internal reset based on the external interrupt during deep software standby mode. For details, see section 4, Resets, section 15, Watchdog Timer (WDT), and section 24, Power-Down Modes.

### 5.3.1 Reset Exception Handling

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

1. The internal state of the CPU and the registers of the on-chip peripheral modules are initialized, VBR is cleared to H'00000000, the T bit is cleared to 0 in EXR, and the I bits are set to 1 in EXR and CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC. In this case, by reading the flags in the registers of individual functions, it is possible to determine whether the particular internal reset has been issued based on the watchdog timer or the external interrupt during deep software standby mode. For details, see section 4, Resets, section 15, Watchdog Timer (WDT), and section 24, Power-Down Modes.

Figures 5.1 and 5.2 show examples of the reset sequence.

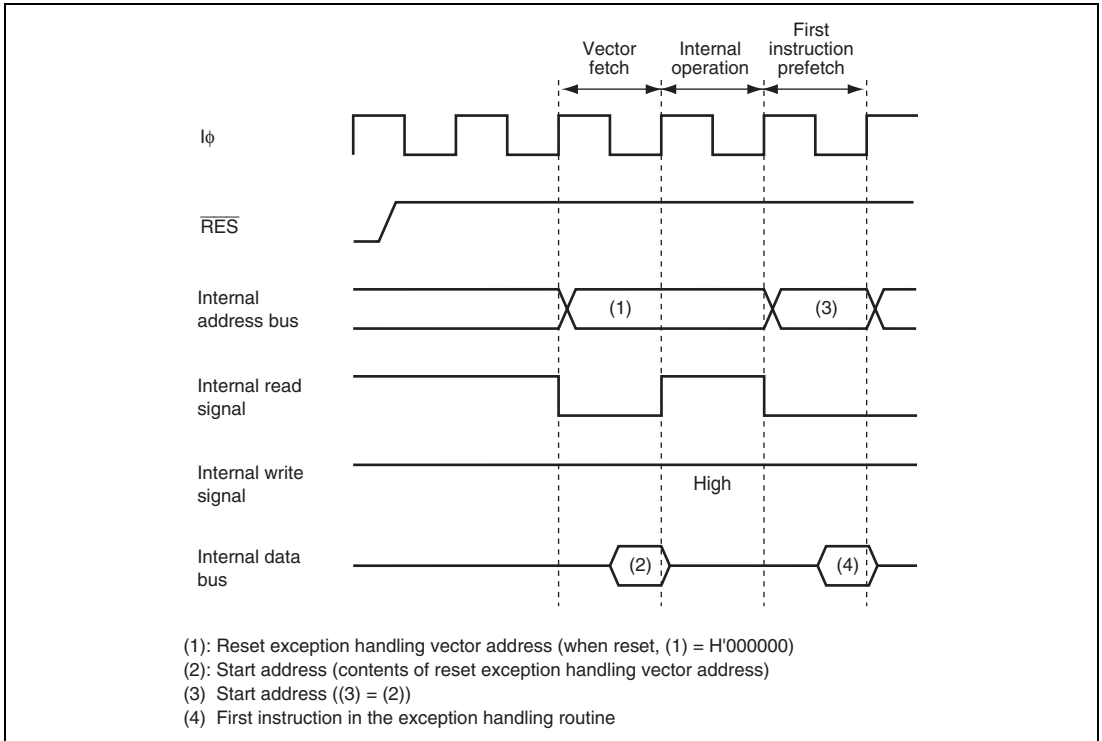
### 5.3.2 Interrupts after Reset

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`).

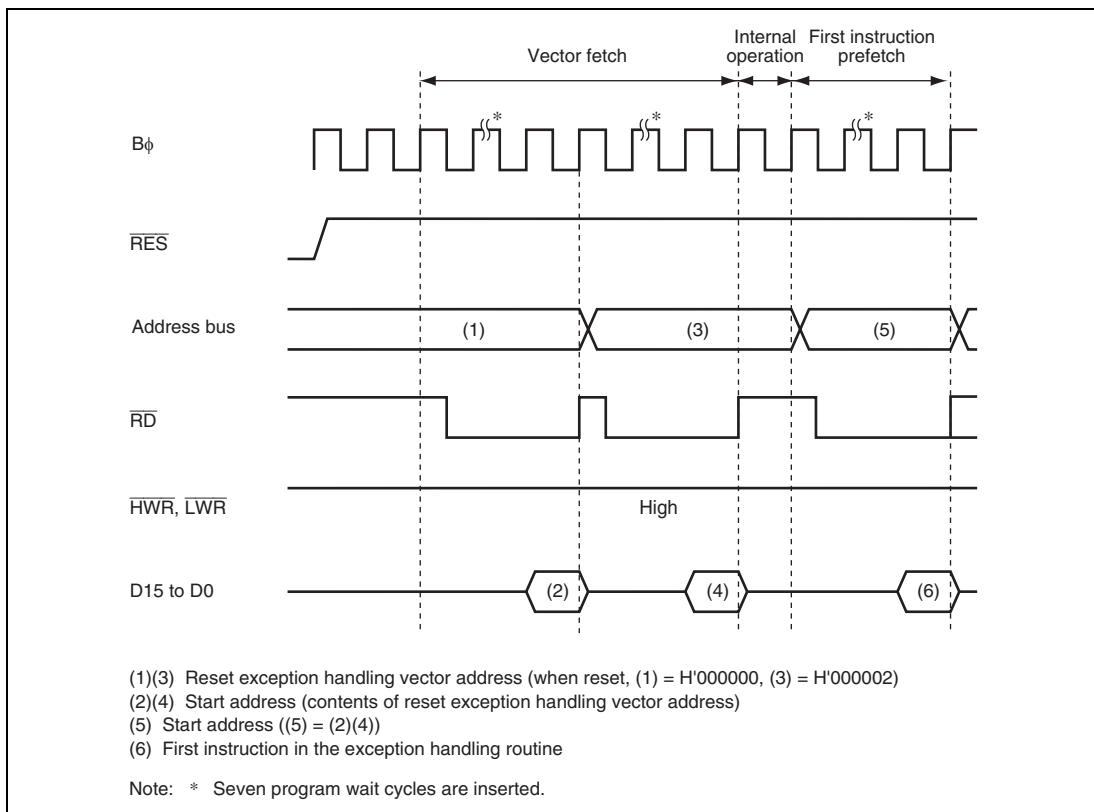
### 5.3.3 On-Chip Peripheral Functions after Reset Release

After the reset state is released, MSTPCRA and MSTPCRB are initialized to H'0FFF and H'FFFF, respectively, and all modules except the DTC and DMAC enter module stop mode.

Consequently, on-chip peripheral module registers cannot be read or written to. Register reading and writing is enabled when module stop mode is canceled.



**Figure 5.1 Reset Sequence (On-chip ROM Enabled Advanced Mode)**



**Figure 5.2 Reset Sequence**  
**(16-Bit External Access in On-chip ROM Disabled Advanced Mode)**

## 5.4 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. Before changing interrupt control modes, the T bit must be cleared. For details on interrupt control modes, see section 6, 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 not affected by interrupt masking by CCR. Table 5.4 shows the state of CCR and EXR after execution of trace exception handling. Trace mode is canceled by clearing the T bit in EXR to 0 during the trace exception handling. However, 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.

Interrupts are accepted even within the trace exception handling routine.

**Table 5.4 States 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 the previous value.



## 5.5 Address Error

### 5.5.1 Address Error Source

Instruction fetch, stack operation, or data read/write shown in table 5.5 may cause an address error.

**Table 5.5 Bus Cycle and Address Error**

Bus Cycle		Description	Address Error
Type	Bus Master		
Instruction fetch	CPU	Fetches instructions from even addresses	No (normal)
		Fetches instructions from odd addresses	Occurs
		Fetches instructions from areas other than on-chip peripheral module space* <sup>1</sup>	No (normal)
		Fetches instructions from on-chip peripheral module space* <sup>1</sup>	Occurs
		Fetches instructions from external memory space in single-chip mode	Occurs
		Fetches instructions from access prohibited area.* <sup>2</sup>	Occurs
Stack operation	CPU	Accesses stack when the stack pointer value is even address	No (normal)
		Accesses stack when the stack pointer value is odd	Occurs
Data read/write	CPU	Accesses word data from even addresses	No (normal)
		Accesses word data from odd addresses	No (normal)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to access prohibited area* <sup>2</sup>	Occurs
Data read/write	DTC or DMAC	Accesses word data from even addresses	No (normal)
		Accesses word data from odd addresses	No (normal)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to access prohibited area* <sup>2</sup>	Occurs
Single address transfer	DMAC	Address access space is the external memory space for single address transfer	No (normal)
		Address access space is not the external memory space for single address transfer	Occurs

Notes: 1. For on-chip peripheral module space, see section 8, Bus Controller (BSC).  
 2. For the access-prohibited area, see figure 3.1, Address Map (Advanced Mode) in section 3.4, Address Map.

### 5.5.2 Address Error Exception Handling

When an address error occurs, address error exception handling starts after the bus cycle causing the address error ends and current instruction execution completes. The address error exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the address error is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Even though an address error occurs during a transition to an address error exception handling, the address error is not accepted. This prevents an address error from occurring due to stacking for exception handling, thereby preventing infinitive stacking.

If the SP contents are not a multiple of 2 when an address error exception handling occurs, the stacked values (PC, CCR, and EXR) are undefined.

When an address error occurs, the following is performed to halt the DTC and DMAC.

- The ERR bit of DTCCR in the DTC is set to 1.
- The ERRF bit of DMDR\_0 in the DMAC is set to 1.
- The DTE bits of DMDRs for all channels in the DMAC are cleared to 0 to forcibly terminate transfer.

Table 5.6 shows the state of CCR and EXR after execution of the address error exception handling.

**Table 5.6 States of CCR and EXR after Address Error Exception Handling**

Interrupt Control Mode	CCR		EXR	
	I	UI	T	I2 to I0
0	1	—	—	—
2	1	—	0	7

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

## 5.6 Interrupts

### 5.6.1 Interrupt Sources

Interrupt sources are NMI, UBC break interrupt, IRQ0 to IRQ15, and on-chip peripheral modules, as shown in table 5.7.

**Table 5.7 Interrupt Sources**

Type	Source	Number of Sources
NMI	NMI pin (external input)	1
UBC break interrupt	User break controller (UBC)	1
IRQ0 to IRQ15	Pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ15}}$ (external input)	16
On-chip peripheral module	DMA controller (DMAC)	4
	Watchdog timer (WDT)	1
	A/D converter	1
	16-bit timer pulse unit (TPU)	26
	8-bit timer (TMR)	24
	Serial communications interface (SCI)	20
	IIC bus interface 2 (IIC2)	2
	$\Delta\Sigma$ A/D converter	1

Different vector numbers and vector table offsets are assigned to different interrupt sources. For vector number and vector table offset, see table 6.2, Interrupt Sources, Vector Address Offsets, and Interrupt Priority in section 6, Interrupt Controller.

## 5.6.2 Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign eight priority/mask levels to interrupts other than NMI to enable multiple-interrupt control. The source to start interrupt exception handling and the vector address differ depending on the product. For details, see section 6, Interrupt Controller.

The interrupt exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the interrupt source is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

## 5.7 Instruction Exception Handling

There are two instructions that cause exception handling: trap instruction and illegal instruction.

### 5.7.1 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 trap instruction exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the vector number specified in the TRAPA instruction is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

A start address is read from the vector table corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 5.8 shows the state of CCR and EXR after execution of trap instruction exception handling.

**Table 5.8 States 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 the previous value.

### 5.7.2 Sleep Instruction

The exception handling starts when a sleep instruction (SLEEP) is executed while the SSBY bit in SBYCR is clear (= 0) and the SLPIE bit in SBYCR is set (= 1). The exception handling caused by execution of a sleep instruction is always executable in the program execution state.

The following operations are performed by the CPU:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the sleep instruction is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

After execution of a sleep instruction, a bus master other than the CPU may have bus mastership. In this case, the exception handling starts at the point when the CPU gets bus mastership after the operation of the other bus master has ended.

Table 5.9 shows the state of CCR and EXR after execution of illegal instruction exception handling. See section 24.10, Sleep Instruction Exception Handling, for details.

**Table 5.9 States of CCR and EXR after Sleep Instruction Exception Handling**

Interrupt Control Mode	CCR		EXR	
	I	UI	T	I2 to I0
0	1	—	—	—
2	1	—	0	7

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

### 5.7.3 Illegal Instruction

The illegal instructions are general illegal instructions and slot illegal instructions. The exception handling by the general illegal instruction starts when an undefined code is executed. The exception handling by the slot illegal instruction starts when a particular instruction (e.g. its code length is two words or more, or it changes the PC contents) at a delay slot (immediately after a delayed branch instruction) is executed. The exception handling by the general illegal instruction and slot illegal instruction is always executable in the program execution state.

The exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the occurred exception is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Table 5.10 shows the state of CCR and EXR after execution of illegal instruction exception handling.

**Table 5.10 States of CCR and EXR after Illegal Instruction Exception Handling**

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

[Legend]

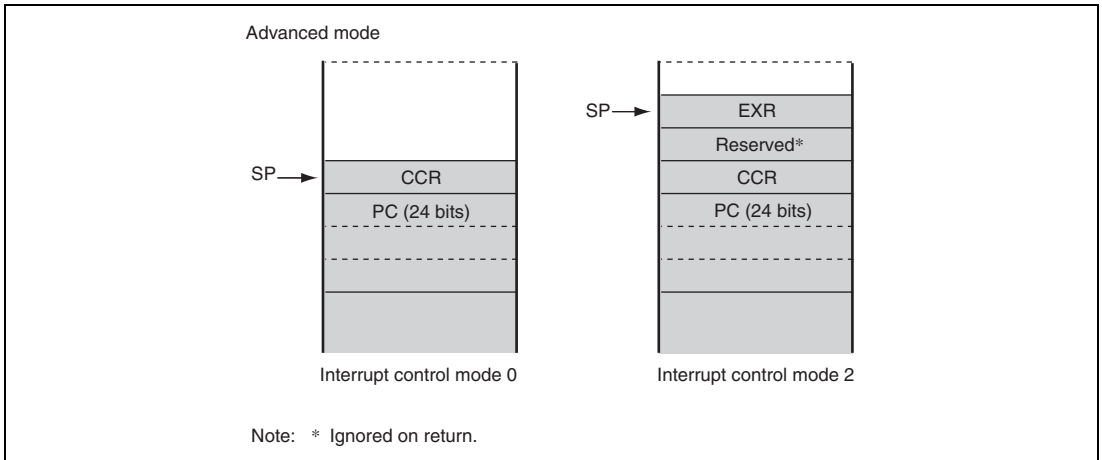
1: Set to 1

0: Cleared to 0

—: Retains the previous value.

## 5.8 Stack Status after Exception Handling

Figure 5.3 shows the stack after completion of exception handling.



**Figure 5.3 Stack Status after Exception Handling**



## 5.9 Usage Note

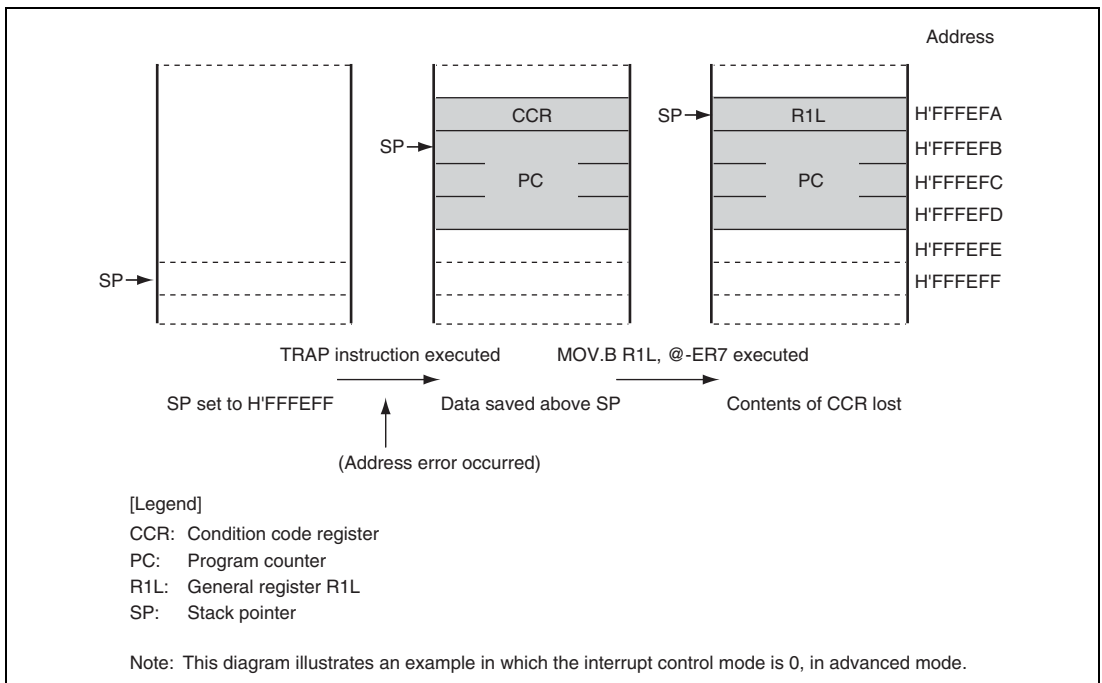
When performing stack-manipulating access, this LSI assumes that the lowest address bit is 0. The stack should always be accessed by a word transfer instruction or a longword transfer instruction, and the value of the stack pointer (SP: ER7) should always be kept even. Use the following instructions 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)

Performing stack manipulation while SP is set to an odd value leads to an address error. Figure 5.4 shows an example of operation when the SP value is odd.



**Figure 5.4 Operation when SP Value is Odd**



## Section 6 Interrupt Controller

### 6.1 Features

- Two interrupt control modes

Any of two interrupt control modes can be set by means of bits INTM1 and INTM0 in the interrupt control register (INTCR).
- Priority can be assigned by the interrupt priority register (IPR)

IPR provides for setting interrupt priority. Eight levels can be set for each module for all interrupts except for the interrupt requests listed below. The following eight interrupt requests are given priority of 8, therefore they are accepted at all times.

  - NMI
  - Illegal instruction
  - Trace
  - Trap instruction
  - CPU address error
  - DMA address error (occurred in the DTC and DMAC)
  - Sleep instruction
  - Break interrupt
- Independent vector addresses

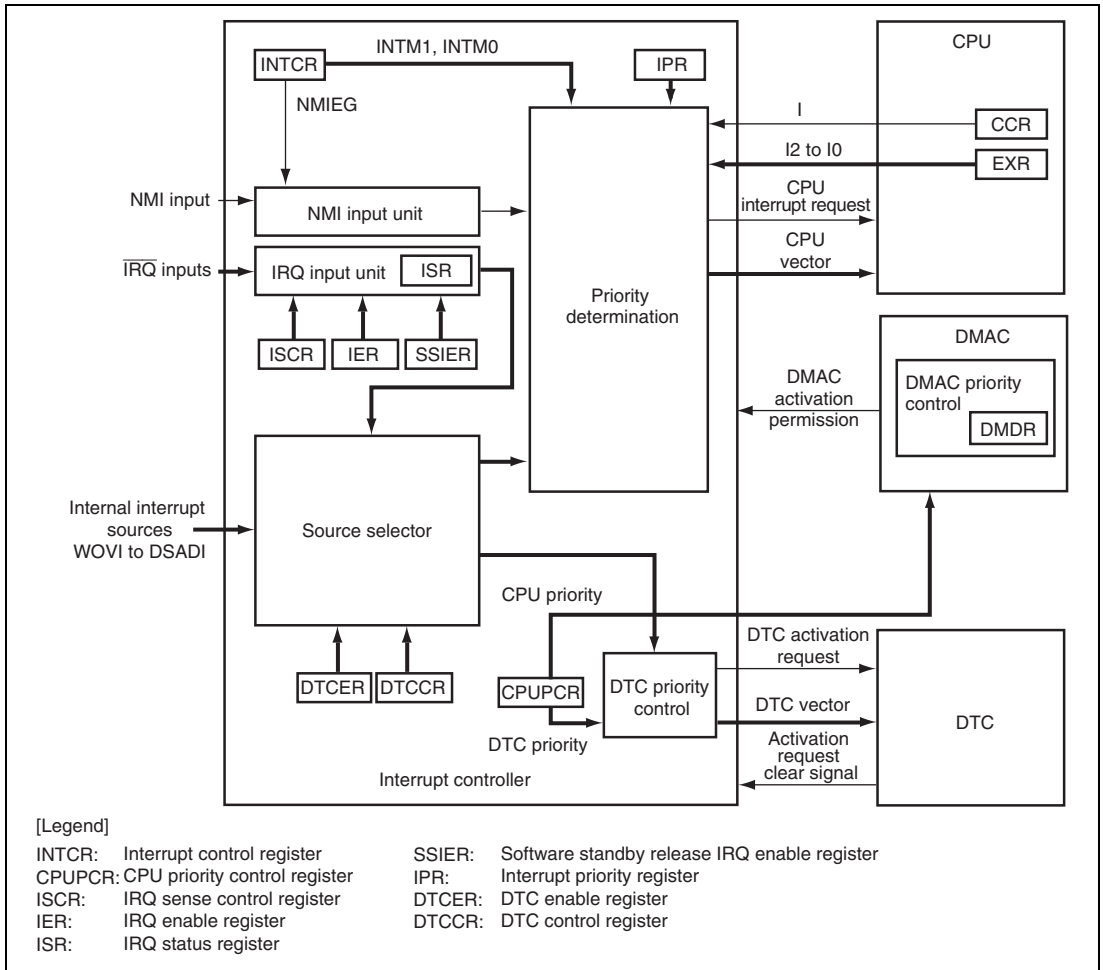
All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.
- Seventeen external interrupts

NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge detection can be selected for NMI. Falling edge, rising edge, or both edge detection, or level sensing, can be selected for  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ0}}$ .
- DTC and DMAC control

DTC and DMAC can be activated by means of interrupts.
- CPU priority control function

The priority levels can be assigned to the CPU, DTC, and DMAC. The priority level of the CPU can be automatically assigned on an exception generation. Priority can be given to the CPU interrupt exception handling over that of the DTC and DMAC transfer.

A block diagram of the interrupt controller is shown in figure 6.1.



**Figure 6.1 Block Diagram of Interrupt Controller**

## 6.2 Input/Output Pins

Table 6.1 shows the pin configuration of the interrupt controller.

**Table 6.1 Pin Configuration**

<b>Name</b>	<b>I/O</b>	<b>Function</b>
NMI	Input	Nonmaskable External Interrupt Rising or falling edge can be selected.
IRQ15 to IRQ0	Input	Maskable External Interrupts Rising, falling, or both edges, or level sensing, can be independently selected.

## 6.3 Register Descriptions

The interrupt controller has the following registers.

- Interrupt control register (INTCR)
- CPU priority control register (CPUPCR)
- Interrupt priority registers A to I, K, L, P to R (IPRA to IPRI, IPRK, IPRL, IPRP to IPRR)
- IRQ enable register (IER)
- IRQ sense control registers H and L (ISCRH, ISCRL)
- IRQ status register (ISR)
- Software standby release IRQ enable register (SSIER)

### 6.3.1 Interrupt Control Register (INTCR)

INTCR selects the interrupt control mode, and the detected edge for NMI.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	INTM1	INTM0	NMIEG	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These are read-only bits and cannot be modified.
5	INTM1	0	R/W	Interrupt Control Select Mode 1 and 0
4	INTM0	0	R/W	These bits select either of two interrupt control modes for the interrupt controller. 00: Interrupt control mode 0 Interrupts are controlled by I bit in CCR. 01: Setting prohibited. 10: Interrupt control mode 2 Interrupts are controlled by bits I2 to I0 in EXR, and IPR. 11: Setting prohibited.
3	NMIEG	0	R/W	NMI Edge Select Selects the input edge for the NMI pin. 0: Interrupt request generated at falling edge of NMI input 1: Interrupt request generated at rising edge of NMI input
2 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

### 6.3.2 CPU Priority Control Register (CPUPCR)

CPUPCR sets whether or not the CPU has priority over the DTC and DMAC. The interrupt exception handling by the CPU can be given priority over that of the DTC and DMAC transfer. The priority level of the DTC is set by bits DTCP2 to DTCP0 in CPUPCR. The priority level of the DMAC is set by the DMAC control register for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	CPUPCE	DTCP2	DTCP1	DTCP0	IPSETE	CPUP2	CPUP1	CPUP0
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 IPSETE bit is set to 1, the CPU priority is automatically updated, so these bits cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	CPUPCE	0	R/W	<p>CPU Priority Control Enable</p> <p>Controls the CPU priority control function. Setting this bit to 1 enables the CPU priority control over the DTC and DMAC.</p> <p>0: CPU always has the lowest priority 1: CPU priority control enabled</p>
6	DTCP2	0	R/W	DTC Priority Level 2 to 0
5	DTCP1	0	R/W	These bits set the DTC priority level.
4	DTCP0	0	R/W	<p>000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)</p>
3	IPSETE	0	R/W	<p>Interrupt Priority Set Enable</p> <p>Controls the function which automatically assigns the interrupt priority level of the CPU. Setting this bit to 1 automatically sets bits CPUP2 to CPUP0 by the CPU interrupt mask bit (I bit in CCR or bits I2 to I0 in EXR).</p> <p>0: Bits CPUP2 to CPUP0 are not updated automatically 1: The interrupt mask bit value is reflected in bits CPUP2 to CPUP0</p>

---

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
2	CPUP2	0	R/(W)*	CPU Priority Level 2 to 0
1	CPUP1	0	R/(W)*	These bits set the CPU priority level. When the CPUPCE is set to 1, the CPU priority control function over the DTC and DMAC becomes valid and the priority of CPU processing is assigned in accordance with the settings of bits CPUP2 to CPUP0.  000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
0	CPUP0	0	R/(W)*	

---

Note: \* When the IPSETE bit is set to 1, the CPU priority is automatically updated, so these bits cannot be modified.



### 6.3.3 Interrupt Priority Registers A to I, K, L, P to R (IPRA to IPRI, IPRK, IPRL, IPRP to IPRR)

IPR sets priority (levels 7 to 0) for interrupts other than NMI.

Setting a value in the range from B'000 to B'111 in the 3-bit groups of bits 14 to 12, 10 to 8, 6 to 4, and 2 to 0 assigns a priority level to the corresponding interrupt. For the correspondence between the interrupt sources and the IPR settings, see table 6.2.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	IPR14	IPR13	IPR12	—	IPR10	IPR9	IPR8
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This is a read-only bit and cannot be modified.
14	IPR14	1	R/W	Sets the priority level of the corresponding interrupt source.
13	IPR13	1	R/W	
12	IPR12	1	R/W	000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
11	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
10	IPR10	1	R/W	Sets the priority level of the corresponding interrupt source.
9	IPR9	1	R/W	
8	IPR8	1	R/W	000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
7	—	0	R	Reserved This is a read-only bit and cannot be modified.
6	IPR6	1	R/W	Sets the priority level of the corresponding interrupt source.
5	IPR5	1	R/W	
4	IPR4	1	R/W	000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
3	—	0	R	Reserved This is a read-only bit and cannot be modified.
2	IPR2	1	R/W	Sets the priority level of the corresponding interrupt source.
1	IPR1	1	R/W	
0	IPR0	1	R/W	000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)

### 6.3.4 IRQ Enable Register (IER)

IER enables interrupt requests IRQ15 to IRQ0. However, the bits of this register cannot set the IRQ interrupt requests (IRQ3 to IRQ0) to exit from deep software standby mode. For details, see section 24.2.6, Deep Standby Interrupt Enable Register (DPSIER).

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ15E	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E
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	7	6	5	4	3	2	1	0
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15E	0	R/W	IRQ15 Enable The IRQ15 interrupt request is enabled when this bit is 1.
14	IRQ14E	0	R/W	IRQ14 Enable The IRQ14 interrupt request is enabled when this bit is 1.
13	IRQ13E	0	R/W	IRQ13 Enable The IRQ13 interrupt request is enabled when this bit is 1.
12	IRQ12E	0	R/W	IRQ12 Enable The IRQ12 interrupt request is enabled when this bit is 1.
11	IRQ11E	0	R/W	IRQ11 Enable The IRQ11 interrupt request is enabled when this bit is 1.
10	IRQ10E	0	R/W	IRQ10 Enable The IRQ10 interrupt request is enabled when this bit is 1.
9	IRQ9E	0	R/W	IRQ9 Enable The IRQ9 interrupt request is enabled when this bit is 1.
8	IRQ8E	0	R/W	IRQ8 Enable The IRQ8 interrupt request is enabled when this bit is 1.
7	IRQ7E	0	R/W	IRQ7 Enable The IRQ7 interrupt request is enabled when this bit is 1.

Bit	Bit Name	Initial Value	R/W	Description
6	IRQ6E	0	R/W	IRQ6 Enable The IRQ6 interrupt request is enabled when this bit is 1.
5	IRQ5E	0	R/W	IRQ5 Enable The IRQ5 interrupt request is enabled when this bit is 1.
4	IRQ4E	0	R/W	IRQ4 Enable The IRQ4 interrupt request is enabled when this bit is 1.
3	IRQ3E	0	R/W	IRQ3 Enable* The IRQ3 interrupt request is enabled when this bit is 1.
2	IRQ2E	0	R/W	IRQ2 Enable* The IRQ2 interrupt request is enabled when this bit is 1.
1	IRQ1E	0	R/W	IRQ1 Enable* The IRQ1 interrupt request is enabled when this bit is 1.
0	IRQ0E	0	R/W	IRQ0 Enable* The IRQ0 interrupt request is enabled when this bit is 1.

Note: \* The bits of this register cannot set the IRQ interrupt requests to exit from deep software standby mode. For details, see section 24.2.6, Deep Standby Interrupt Enable Register (DPSIER).

### 6.3.5 IRQ Sense Control Registers H and L (ISCRH, ISCR L)

ISCRH and ISCR L select the source that generates an interrupt request from IRQ15 to IRQ0 input.

Upon changing the setting of ISCR, IRQnF (n = 0 to 15) in ISR is often set to 1 accidentally through an internal operation. In this case, an interrupt exception handling is executed if an IRQn interrupt request is enabled. In order to prevent such an accidental interrupt from occurring, the setting of ISCR should be changed while the IRQn interrupt is disabled, and then the IRQnF in ISR should be cleared to 0. However, the bits of this register cannot set the edge selections, IRQnSR (n = 3 to 0) and IRQnSF (n = 3 to 0), for IRQ interrupt requests to exit from deep software standby mode. For details, see section 24.2.8, Deep Standby Interrupt Edge Register (DPSIEGR).

#### • ISCRH

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ15SR	IRQ15SF	IRQ14SR	IRQ14SF	IRQ13SR	IRQ13SF	IRQ12SR	IRQ12SF
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	7	6	5	4	3	2	1	0
Bit Name	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF
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 L

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF
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	7	6	5	4	3	2	1	0
Bit Name	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF
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

- ISCRH

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15SR	0	R/W	IRQ15 Sense Control Rise
14	IRQ15SF	0	R/W	IRQ15 Sense Control Fall
				00: Interrupt request generated by low level of $\overline{\text{IRQ15}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ15}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ15}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ15}}$
13	IRQ14SR	0	R/W	IRQ14 Sense Control Rise
12	IRQ14SF	0	R/W	IRQ14 Sense Control Fall
				00: Interrupt request generated by low level of $\overline{\text{IRQ14}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ14}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ14}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ14}}$
11	IRQ13SR	0	R/W	IRQ13 Sense Control Rise
10	IRQ13SF	0	R/W	IRQ13 Sense Control Fall
				00: Interrupt request generated by low level of $\overline{\text{IRQ13}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ13}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ13}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ13}}$
9	IRQ12SR	0	R/W	IRQ12 Sense Control Rise
8	IRQ12SF	0	R/W	IRQ12 Sense Control Fall
				00: Interrupt request generated by low level of $\overline{\text{IRQ12}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ12}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ12}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ12}}$

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ11SR	0	R/W	IRQ11 Sense Control Rise
6	IRQ11SF	0	R/W	IRQ11 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ11}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ11}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ11}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ11}}$
5	IRQ10SR	0	R/W	IRQ10 Sense Control Rise
4	IRQ10SF	0	R/W	IRQ10 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ10}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ10}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ10}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ10}}$
3	IRQ9SR	0	R/W	IRQ9 Sense Control Rise
2	IRQ9SF	0	R/W	IRQ9 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ9}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ9}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ9}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ9}}$
1	IRQ8SR	0	R/W	IRQ8 Sense Control Rise
0	IRQ8SF	0	R/W	IRQ8 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ8}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ8}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ8}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ8}}$

- ISCR\_L

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ7SR	0	R/W	IRQ7 Sense Control Rise*
14	IRQ7SF	0	R/W	IRQ7 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ7}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ7}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ7}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ7}}$
13	IRQ6SR	0	R/W	IRQ6 Sense Control Rise*
12	IRQ6SF	0	R/W	IRQ6 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ6}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ6}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ6}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ6}}$
11	IRQ5SR	0	R/W	IRQ5 Sense Control Rise*
10	IRQ5SF	0	R/W	IRQ5 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ5}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ5}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ5}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ5}}$
9	IRQ4SR	0	R/W	IRQ4 Sense Control Rise*
8	IRQ4SF	0	R/W	IRQ4 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ4}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ4}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ4}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ4}}$



Bit	Bit Name	Initial Value	R/W	Description
7	IRQ3SR	0	R/W	IRQ3 Sense Control Rise*
6	IRQ3SF	0	R/W	IRQ3 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ3}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ3}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ3}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ3}}$
5	IRQ2SR	0	R/W	IRQ2 Sense Control Rise*
4	IRQ2SF	0	R/W	IRQ2 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ2}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ2}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ2}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ2}}$
3	IRQ1SR	0	R/W	IRQ1 Sense Control Rise*
2	IRQ1SF	0	R/W	IRQ1 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ1}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ1}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ1}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ1}}$
1	IRQ0SR	0	R/W	IRQ0 Sense Control Rise*
0	IRQ0SF	0	R/W	IRQ0 Sense Control Fall* 00: Interrupt request generated by low level of $\overline{\text{IRQ0}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ0}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ0}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ0}}$

Note: \* The bits of this register cannot set the edge selections for IRQ interrupt requests to exit from deep software standby mode. For details, see section 24.2.8, Deep Standby Interrupt Edge Register (DPSIEGR).

### 6.3.6 IRQ Status Register (ISR)

ISR is an IRQ15 to IRQ0 interrupt request register. However, the bits of this register cannot set the IRQ interrupt request flags, IRQnF (n = 3 to 0), to exit from deep software standby mode. For details, see section 24.2.7, Deep Standby Interrupt Flag Register (DPSIFR).

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ15F	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F
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	7	6	5	4	3	2	1	0
Bit Name	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. The bit manipulation instructions or memory operation instructions should be used to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15F	0	R/(W)* <sup>1</sup>	[Setting condition]
14	IRQ14F	0	R/(W)* <sup>1</sup>	<ul style="list-style-type: none"> <li>When the interrupt selected by ISCR occurs</li> </ul>
13	IRQ13F	0	R/(W)* <sup>1</sup>	[Clearing conditions]
12	IRQ12F	0	R/(W)* <sup>1</sup>	<ul style="list-style-type: none"> <li>Writing 0 after reading IRQnF = 1 (n = 11 to 0)</li> </ul>
11	IRQ11F	0	R/(W)* <sup>1</sup>	<ul style="list-style-type: none"> <li>When interrupt exception handling is executed while low-level sensing is selected and IRQn input is high</li> </ul>
10	IRQ10F	0	R/(W)* <sup>1</sup>	<ul style="list-style-type: none"> <li>When IRQn interrupt exception handling is executed while falling-, rising-, or both-edge sensing is selected</li> </ul>
9	IRQ9F	0	R/(W)* <sup>1</sup>	<ul style="list-style-type: none"> <li>When the DTC is activated by an IRQn interrupt, and the DISEL bit in MRB of the DTC is cleared to 0 (n = 15 to 0)</li> </ul>
8	IRQ8F	0	R/(W)* <sup>1</sup>	
7	IRQ7F	0	R/(W)* <sup>1</sup>	
6	IRQ6F	0	R/(W)* <sup>1</sup>	
5	IRQ5F	0	R/(W)* <sup>1</sup>	
4	IRQ4F	0	R/(W)* <sup>1</sup>	
3	IRQ3F* <sup>2</sup>	0	R/(W)* <sup>1</sup>	
2	IRQ2F* <sup>2</sup>	0	R/(W)* <sup>1</sup>	
1	IRQ1F* <sup>2</sup>	0	R/(W)* <sup>1</sup>	
0	IRQ0F* <sup>2</sup>	0	R/(W)* <sup>1</sup>	

Notes: 1. Only 0 can be written, to clear the flag.  
 2. The bits of this register cannot set the IRQ interrupt request flags, IRQnF (n = 3 to 0), to exit from deep software standby mode. For details, see section 24.2.7, Deep Standby Interrupt Flag Register (DPSIFR).

### 6.3.7 Software Standby Release IRQ Enable Register (SSIER)

SSIER selects the IRQ interrupt used to leave software standby mode.

The IRQ interrupt used to leave software standby mode should not be set as the DTC activation source.

Bit	15	14	13	12	11	10	9	8
Bit Name	SSI15	SSI14	SSI13	SSI12	SSI11	SSI10	SSI9	SSI8
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	7	6	5	4	3	2	1	0
Bit Name	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0
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	Bit Name	Initial Value	R/W	Description
15	SSI15	0	R/W	Software Standby Release IRQ Setting
14	SSI14	0	R/W	These bits select the IRQ <sub>n</sub> interrupt used to leave software standby mode (n = 15 to 0).
13	SSI13	0	R/W	
12	SSI12	0	R/W	0: An IRQ <sub>n</sub> request is not sampled in software standby mode
11	SSI11	0	R/W	1: When an IRQ <sub>n</sub> request occurs in software standby mode, this LSI leaves software standby mode after the oscillation settling time has elapsed
10	SSI10	0	R/W	
9	SSI9	0	R/W	
8	SSI8	0	R/W	
7	SSI7	0	R/W	
6	SSI6	0	R/W	
5	SSI5	0	R/W	
4	SSI4	0	R/W	
3	SSI3	0	R/W	
2	SSI2	0	R/W	
1	SSI1	0	R/W	
0	SSI0	0	R/W	

## 6.4 Interrupt Sources

### 6.4.1 External Interrupts

There are seventeen external interrupts: NMI and IRQ15 to IRQ0. These interrupts can be used to leave software standby mode.

For the external interrupt to exit from deep software standby mode, see section 24, Power-Down Modes.

#### (1) NMI Interrupts

Nonmaskable interrupt request (NMI) is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode or the settings of the CPU interrupt mask bits. The NMIEG bit in INTCR selects whether an interrupt is requested at the rising or falling edge on the NMI pin.

When an NMI interrupt is generated, the interrupt controller determines that an error has occurred, and performs the following procedure.

- Sets the ERR bit of DTCCR in the DTC to 1.
- Sets the ERRF bit of DMDR\_0 in the DMAC to 1
- Clears the DTE bits of DMDRs for all channels in the DMAC to 0 to forcibly terminate transfer

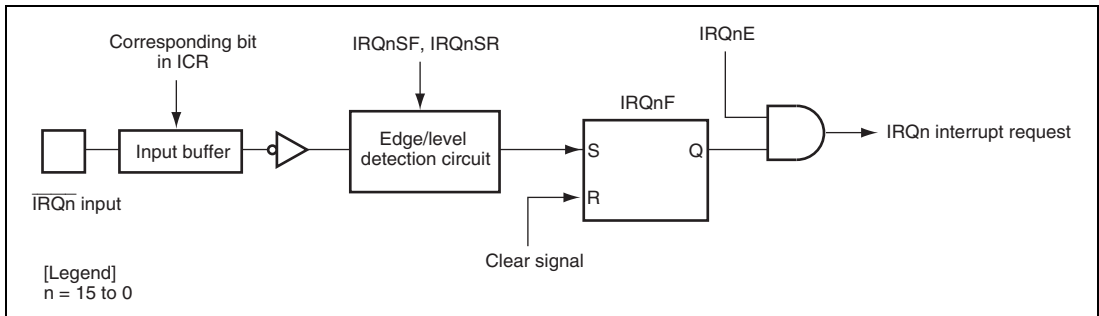
#### (2) IRQn Interrupts

An IRQn interrupt is requested by a signal input on pins  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ0}}$ .  $\overline{\text{IRQn}}$  (n = 15 to 0) 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, on pins  $\overline{\text{IRQn}}$ .
- Enabling or disabling of interrupt requests IRQn can be selected by IER.
- The interrupt priority can be set by IPR.
- The status of interrupt requests IRQn is indicated in ISR. ISR flags can be cleared to 0 by software. The bit manipulation instructions and memory operation instructions should be used to clear the flag.

Detection of  $IRQ_n$  interrupts is enabled through the P1ICR, P2ICR, P5ICR, and P6ICR register settings, and does not change regardless of the output setting. However, when a pin is used as an external interrupt input pin, the pin must not be used as an I/O pin for another function by clearing the corresponding DDR bit to 0.

A block diagram of interrupts  $IRQ_n$  is shown in figure 6.2.



**Figure 6.2 Block Diagram of Interrupts  $IRQ_n$**

When the  $IRQ$  sensing control in ISCR is set to a low level of signal  $\overline{IRQ_n}$ , the level of  $\overline{IRQ_n}$  should be held low until an interrupt handling starts. Then set the corresponding input signal  $\overline{IRQ_n}$  to high in the interrupt handling routine and clear the  $IRQ_nF$  to 0. Interrupts may not be executed when the corresponding input signal  $\overline{IRQ_n}$  is set to high before the interrupt handling begins.

### 6.4.2 Internal Interrupts

The sources for internal interrupts from on-chip peripheral modules have the following features:

- For each on-chip peripheral module there are flags that indicate the interrupt request status, and enable bits that enable or disable these interrupts. They can be controlled independently. When the enable bit is set to 1, an interrupt request is issued to the interrupt controller.
- The interrupt priority can be set by means of IPR.
- The DTC and DMAC can be activated by a TPU, SCI, or other interrupt request.
- The priority levels of DTC and DMAC activation can be controlled by the DTC and DMAC priority control functions.

## 6.5 Interrupt Exception Handling Vector Table

Table 6.2 lists interrupt exception handling sources, vector address offsets, and interrupt priority.

In the default priority order, a lower vector number corresponds to a higher priority. When interrupt control mode 2 is set, priority levels can be changed by setting the IPR contents. The priority for interrupt sources allocated to the same level in IPR follows the default priority, that is, they are fixed.

**Table 6.2 Interrupt Sources, Vector Address Offsets, and Interrupt Priority**

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
External pin	NMI	7	H'001C	—	High	—	—
UBC	Break interrupt	14	H'0038	—	↑	—	—
External pin	IRQ0	64	H'0100	IPRA14 to IPRA12		O	—
	IRQ1	65	H'0104	IPRA10 to IPRA8		O	—
	IRQ2	66	H'0108	IPRA6 to IPRA4		O	—
	IRQ3	67	H'010C	IPRA2 to IPRA0		O	—
	IRQ4	68	H'0110	IPRB14 to IPRB12		O	—
	IRQ5	69	H'0114	IPRB10 to IPRB8		O	—
	IRQ6	70	H'0118	IPRB6 to IPRB4		O	—
	IRQ7	71	H'011C	IPRB2 to IPRB0		O	—
	IRQ8	72	H'0120	IPRC14 to IPRC12		O	—
	IRQ9	73	H'0124	IPRC10 to IPRC8		O	—
	IRQ10	74	H'0128	IPRC6 to IPRC4		O	—
	IRQ11	75	H'012C	IPRC2 to IPRC0		O	—
	IRQ12	76	H'0130	IPRD14 to IPRD12		O	—
	IRQ13	77	H'0134	IPRD10 to IPRD8		O	—
	IRQ14	78	H'0138	IPRD6 to IPRD4		O	—
IRQ15	79	H'013C	IPRD2 to IPRD0	O	—		
—	Reserved for system use	80	H'0140	—	—	—	
WDT	WOVI	81	H'0144	IPRE10 to IPRE8	Low	—	—

Classification	Interrupt Source	Vector Number	Vector Address		IPR	Priority	DTC Activation	DMAC Activation
			Offset*					
—	Reserved for system use	82	H'0148	—	High	—	—	
		83	H'014C			—	—	
		84	H'0150			—	—	
		85	H'0154			—	—	
		86	H'0158			—	—	
		87	H'015C			—	—	
TPU_0	TGI0A	88	H'0160	IPRF6 to IPRF4		O	O	
	TGI0B	89	H'0164			O	—	
	TGI0C	90	H'0168			O	—	
	TGI0D	91	H'016C			O	—	
	TCI0V	92	H'0170			—	—	
TPU_1	TGI1A	93	H'0174	IPRF2 to IPRF0		O	O	
	TGI1B	94	H'0178			O	—	
	TCI1V	95	H'017C			—	—	
	TCI1U	96	H'0180			—	—	
TPU_2	TGI2A	97	H'0184	IPRG14 to IPRG12		O	O	
	TGI2B	98	H'0188			O	—	
	TCI2V	99	H'018C			—	—	
	TCI2U	100	H'0190			—	—	
TPU_3	TGI3A	101	H'0194	IPRG10 to IPRG8		O	O	
	TGI3B	102	H'0198			O	—	
	TGI3C	103	H'019C			O	—	
	TGI3D	104	H'01A0			O	—	
	TCI3V	105	H'01A4			—	—	
TPU_4	TGI4A	106	H'01A8	IPRG6 to IPRG4		O	O	
	TGI4B	107	H'01AC			O	—	
	TCI4V	108	H'01B0			—	—	
	TCI4U	109	H'01B4			Low	—	

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
TPU_5	TGI5A	110	H'01B8	IPRG2 to IPRG0	High	O	O
	TGI5B	111	H'01BC			O	—
	TCI5V	112	H'01C0			—	—
	TCI5U	113	H'01C4			—	—
—	Reserved for system use	114	H'01C8	—		—	—
		115	H'01CC			—	—
TMR_0	CMI0A	116	H'01D0	IPRH14 to IPRH12		O	—
	CMI0B	117	H'01D4			O	—
	OV0I	118	H'01D8			—	—
TMR_1	CMI1A	119	H'01DC	IPRH10 to IPRH8		O	—
	CMI1B	120	H'01E0			O	—
	OV1I	121	H'01E4			—	—
TMR_2	CMI2A	122	H'01E8	IPRH6 to IPRH4		O	—
	CMI2B	123	H'01EC			O	—
	OV2I	124	H'01F0			—	—
TMR_3	CMI3A	125	H'01F4	IPRH2 to IPRH0		O	—
	CMI3B	126	H'01F8			O	—
	OV3I	127	H'01FC			—	—
DMAC	DMTEND0	128	H'0200	IPRI14 to IPRI12		O	—
	DMTEND1	129	H'0204			IPRI10 to IPRI8	O
	Reserved for system use	130	H'0208	—			—
		131	H'020C			—	—
—	Reserved for system use	132	H'0210	—		—	—
		133	H'0214			—	—
		134	H'0218			—	—
		135	H'021C			—	—
DMAC	DMEEND0	136	H'0220	IPRK14 to IPRK12		O	—
	DMEEND1	137	H'0224			—	O
	Reserved for system use	138	H'0228	—			—
		139	H'022C			—	—

Low



Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
—	Reserved for system use	140	H'0230	—	High	—	—
		141	H'0234			—	—
		142	H'0238			—	—
		143	H'023C			—	—
SCI_0	ERI0	144	H'0240	IPRK6 to IPRK4	↑	—	—
	RXI0	145	H'0244			O	O
	TXI0	146	H'0248			O	O
	TEI0	147	H'024C			—	—
SCI_1	ERI1	148	H'0250	IPRK2 to IPRK0	↑	—	—
	RXI1	149	H'0254			O	O
	TXI1	150	H'0258			O	O
	TEI1	151	H'025C			—	—
SCI_2	ERI2	152	H'0260	IPRL14 to IPRL12	↑	—	—
	RXI2	153	H'0264			O	O
	TXI2	154	H'0268			O	O
	TEI2	155	H'026C			—	—
SCI_3	ERI3	156	H'0270	IPRL10 to IPRL8	↑	—	—
	RXI3	157	H'0274			O	O
	TXI3	158	H'0278			O	O
	TEI3	159	H'027C			—	—
SCI_4	ERI4	160	H'0280	IPRL6 to IPRL4	↑	—	—
	RXI4	161	H'0284			O	O
	TXI4	162	H'0288			O	O
	TEI4	163	H'028C			—	—
—	Reserved for system use	164	H'0290	—	↑	—	—
		199	H'031C			—	—
TMR_4	CMI4A	200	H'0320	IPRP10 to IPRP8	↑	O	—
	CMI4B	201	H'0324			O	—
	OV4I	202	H'0328			Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*	IPR	Priority	DTC Activation	DMAC Activation
TMR_5	CMI5A	203	H'032C	IPRP6 to IPRP4	High	O	—
	CMI5B	204	H'0330			O	—
	OV5I	205	H'0334			—	—
TMR_6	CMI6A	206	H'0338	IPRP2 to IPRP0		O	—
	CMI6B	207	H'033C			O	—
	OV6I	208	H'0340			—	—
TMR_7	CMI7A	209	H'0344	IPRQ14 to IPRQ12		O	—
	CMI7B	210	H'0348			O	—
	OV7I	211	H'034C			—	—
—	Reserved for system use	212	H'0350	—		—	—
		215	H'035C			—	—
IIC2	IICI0	216	H'0360	IPRQ6 to IPRQ4		—	—
	Reserved for system use	217	H'0364			—	—
	IICI1	218	H'0368			—	—
	Reserved for system use	219	H'036C			—	—
A/D	ADI0	220	H'0370	IPRQ2 to IPRQ0		—	O
	Reserved for system use	221	H'0374			—	—
		222	H'0378			—	—
		223	H'037C			—	—
$\Delta\Sigma$ A/D	DSADI	224	H'0380	IPRR14 to IPRR12		—	O
	Reserved for system use	225	H'0384			—	—
		226	H'0388			—	—
		227	H'038C			—	—
—	Reserved for system use	228	H'0390	—		—	—
		255	H'03FC			Low	—

Note: \* Lower 16 bits of the start address in advanced, middle, and maximum modes.

## 6.6 Interrupt Control Modes and Interrupt Operation

The interrupt controller has two interrupt control modes: interrupt control mode 0 and interrupt control mode 2. Interrupt operations differ depending on the interrupt control mode. The interrupt control mode is selected by INTCR. Table 6.3 shows the differences between interrupt control mode 0 and interrupt control mode 2.

**Table 6.3 Interrupt Control Modes**

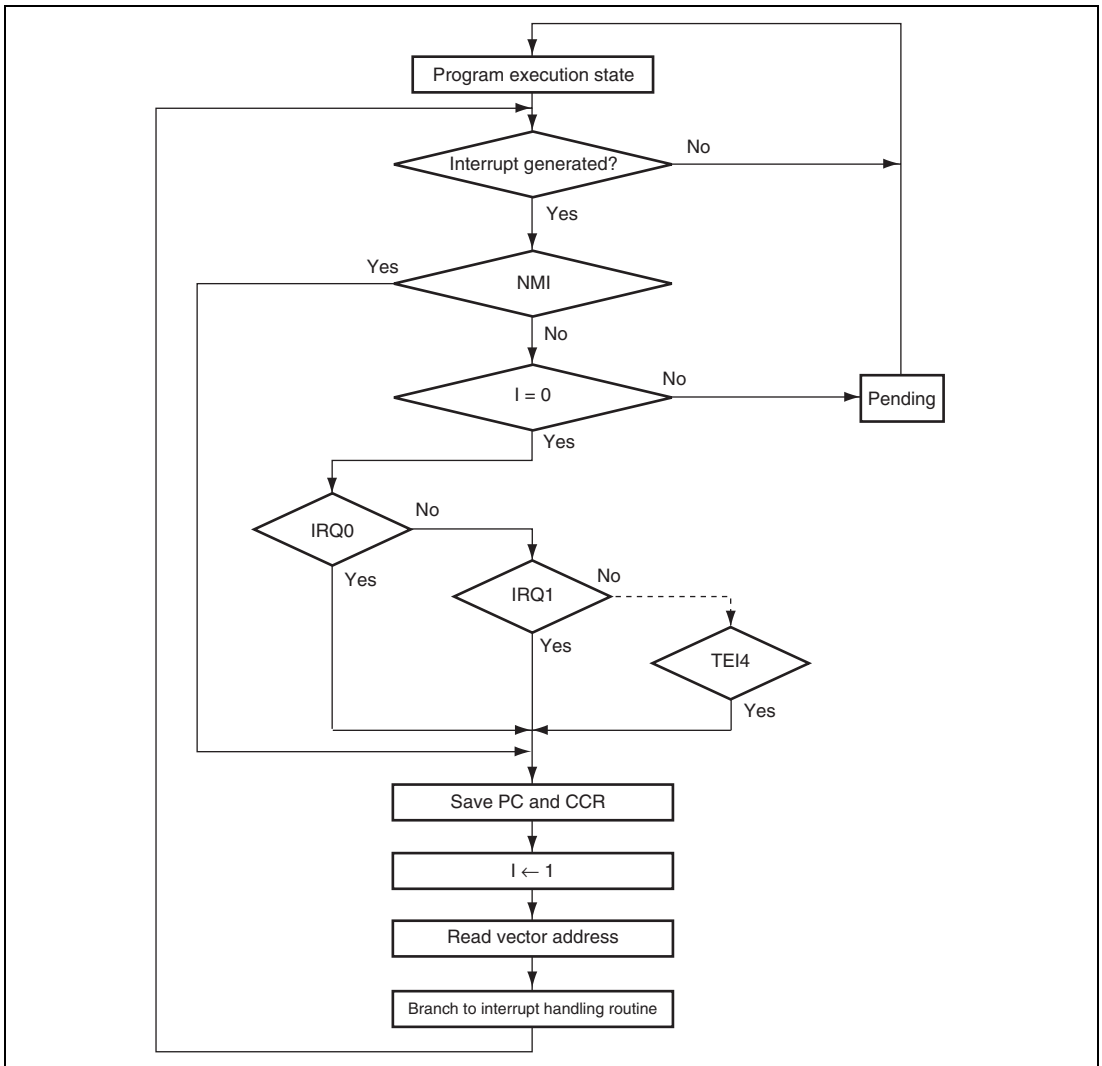
Interrupt Control Mode	Priority Setting Register	Interrupt Mask Bit	Description
0	Default	1	The priority levels of the interrupt sources are fixed default settings. The interrupts except for NMI is masked by the I bit.
2	IPR	I2 to I0	Eight priority levels can be set for interrupt sources except for NMI with IPR. 8-level interrupt mask control is performed by bits I2 to I0.

### 6.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests except for NMI are masked by the I bit in CCR of the CPU. Figure 6.3 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, the interrupt request is sent to the interrupt controller.
2. If the I bit in CCR is set to 1, NMI is accepted, and other interrupt requests are held pending. If the I bit is cleared to 0, an interrupt request is accepted.
3. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority, sends the request to the CPU, and holds other interrupt requests pending.
4. When the CPU accepts the interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR contents are saved to the stack area during the interrupt exception handling. The PC contents saved on the stack is 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. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

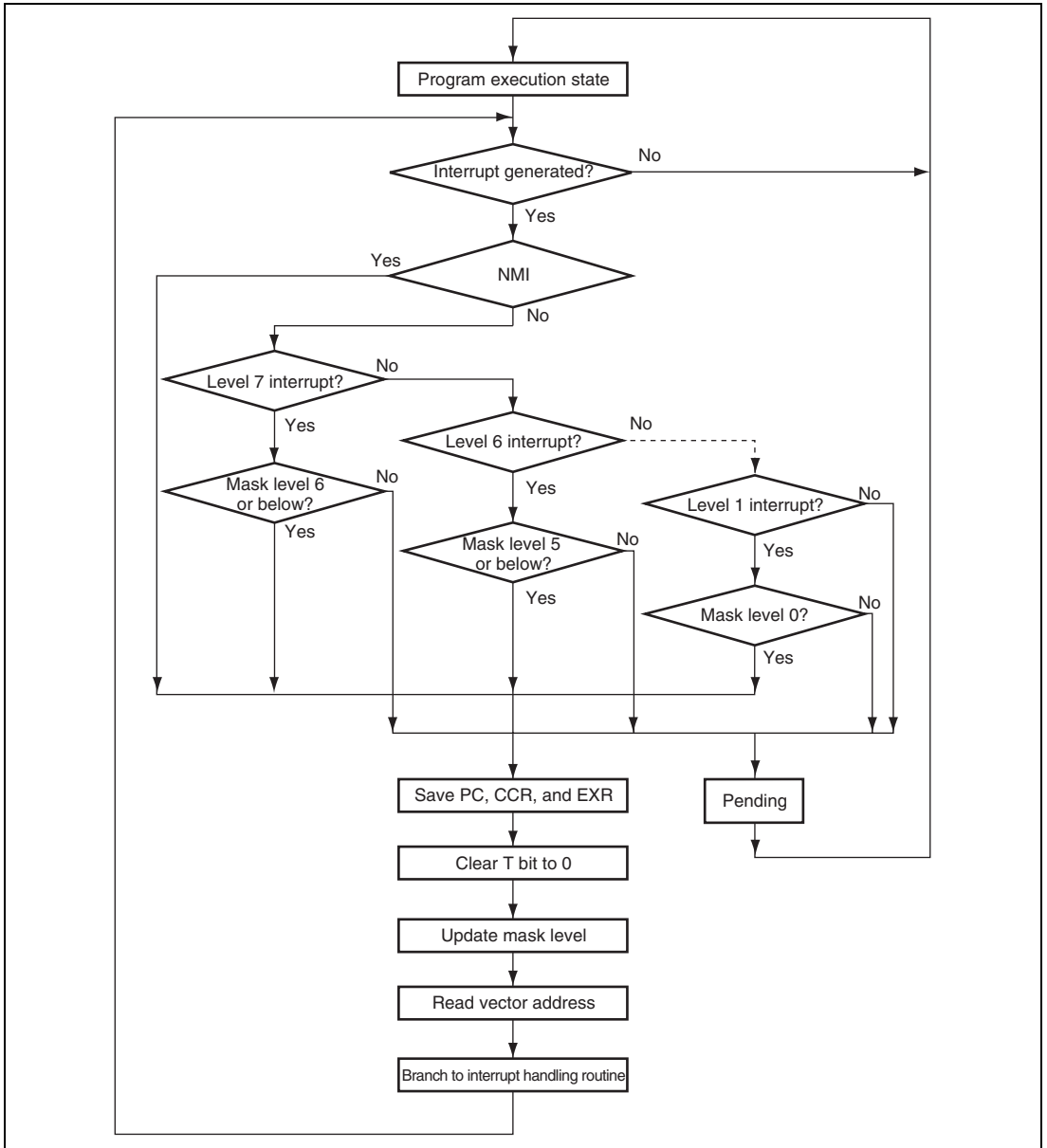


**Figure 6.3 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

## 6.6.2 Interrupt Control Mode 2

In interrupt control mode 2, interrupt requests except for NMI are masked by comparing the interrupt mask level (I2 to I0 bits) in EXR of the CPU and the IPR setting. There are eight levels in mask control. Figure 6.4 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority according to the IPR setting, and holds other interrupt requests pending. If multiple interrupt requests have the same priority, an interrupt request is selected according to the default setting shown in table 6.2.
3. Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. When the interrupt request does not have priority over the mask level set, it is held pending, and only an interrupt request with a priority over the interrupt mask level is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC, CCR, and EXR contents are saved to the stack area during interrupt exception handling. The PC saved on the stack is 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 of the accepted interrupt. If the accepted interrupt is NMI, the interrupt mask level is set to H'7.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.



**Figure 6.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2**

### 6.6.3 Interrupt Exception Handling Sequence

Figure 6.5 shows the interrupt exception handling sequence. The example is for the case where interrupt control mode 0 is set in maximum mode, and the program area and stack area are in on-chip memory.

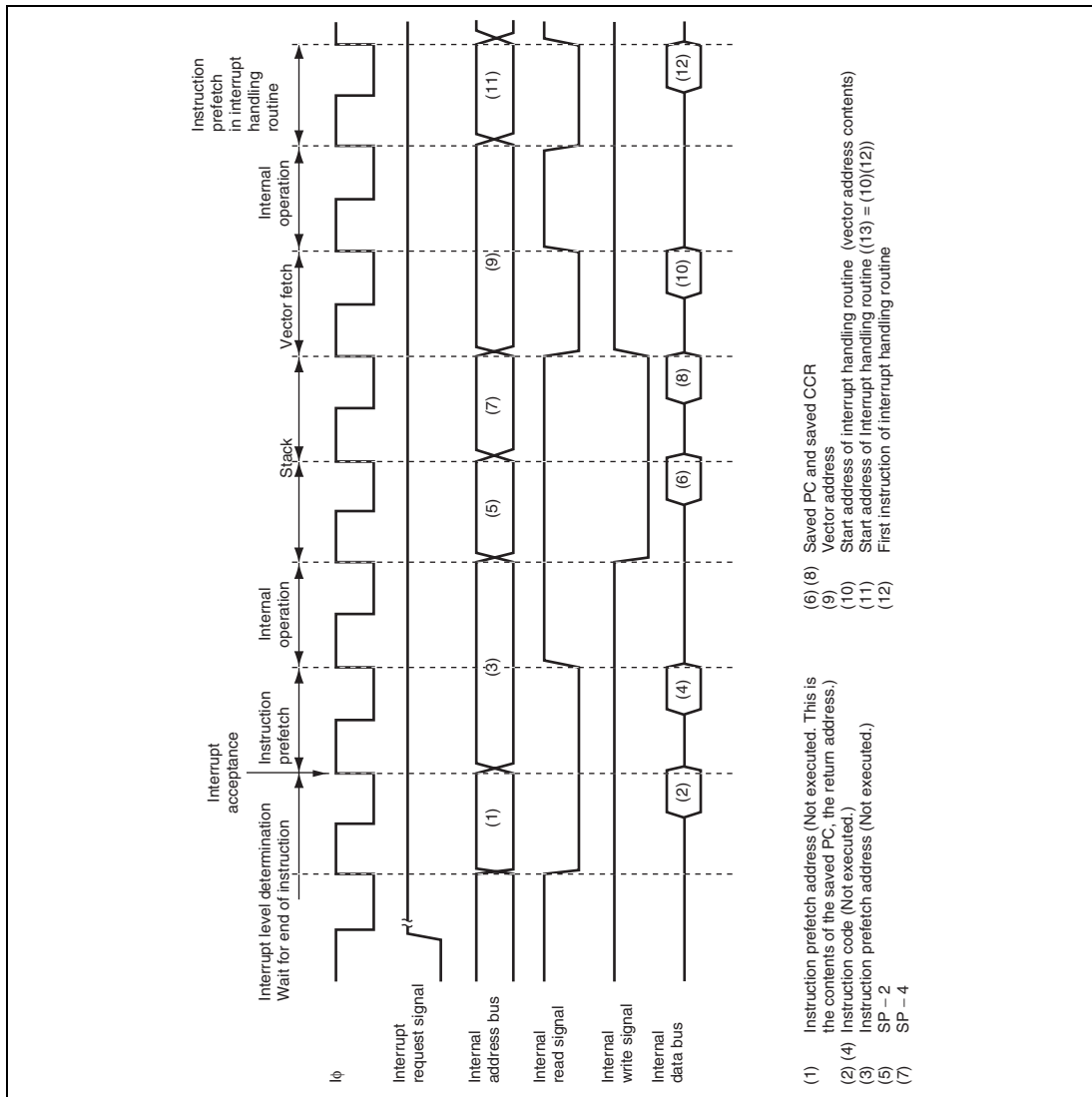


Figure 6.5 Interrupt Exception Handling

### 6.6.4 Interrupt Response Times

Table 6.4 shows interrupt response times – the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The symbols for execution states used in table 6.4 are explained in table 6.5.

This LSI is capable of fast word transfer to on-chip memory, so allocating the program area in on-chip ROM and the stack area in on-chip RAM enables high-speed processing.

**Table 6.4 Interrupt Response Times**

Execution State	Normal Mode* <sup>5</sup>		Advanced Mode		Maximum Mode* <sup>5</sup>	
	Interrupt Control Mode 0	Interrupt Control Mode 2	Interrupt Control Mode 0	Interrupt Control Mode 2	Interrupt Control Mode 0	Interrupt Control Mode 2
Interrupt priority determination* <sup>1</sup>				3		
Number of states until executing instruction ends* <sup>2</sup>				1 to 19 + 2·S <sub>i</sub>		
PC, CCR, EXR stacking	S <sub>k</sub> to 2·S <sub>k</sub> * <sup>6</sup>	2·S <sub>k</sub>	S <sub>k</sub> to 2·S <sub>k</sub> * <sup>6</sup>	2·S <sub>k</sub>	2·S <sub>k</sub>	2·S <sub>k</sub>
Vector fetch				S <sub>n</sub>		
Instruction fetch* <sup>3</sup>				2·S <sub>i</sub>		
Internal processing* <sup>4</sup>				2		
Total (using on-chip memory)	10 to 31	11 to 31	10 to 31	11 to 31	11 to 31	11 to 31

- Notes:
- Two states for an internal interrupt.
  - In the case of the MULXS or DIVXS instruction
  - Prefetch after interrupt acceptance or for an instruction in the interrupt handling routine.
  - Internal operation after interrupt acceptance or after vector fetch
  - Not available in this LSI.
  - When setting the SP value to 4n, the interrupt response time is S<sub>k</sub>; when setting to 4n + 2, the interrupt response time is 2·S<sub>k</sub>.



**Table 6.5 Number of Execution States in Interrupt Handling Routine**

Symbol	On-Chip Memory	Object of Access			
		External Device			
		8-Bit Bus		16-Bit Bus	
		2-State Access	3-State Access	2-State Access	3-State Access
Vector fetch $S_h$	1	8	$12 + 4m$	4	$6 + 2m$
Instruction fetch $S_l$	1	4	$6 + 2m$	2	$3 + m$
Stack manipulation $S_k$	1	8	$12 + 4m$	4	$6 + 2m$

[Legend]

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

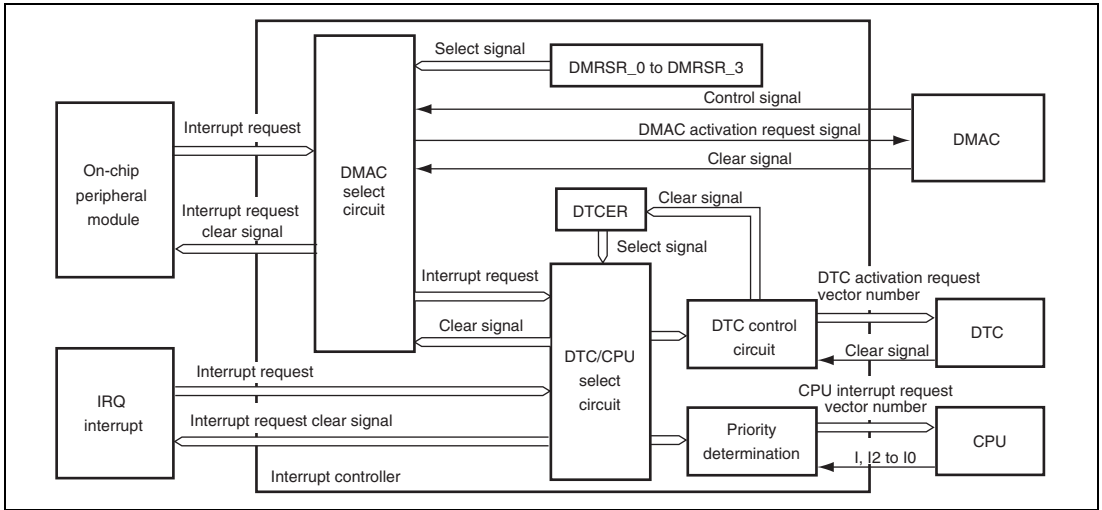
### 6.6.5 DTC and DMAC Activation by Interrupt

The DTC and DMAC can be activated by an interrupt. In this case, the following options are available:

- Interrupt request to the CPU
- Activation request to the DTC
- Activation request to the DMAC
- Combination of the above

For details on interrupt requests that can be used to activate the DTC and DMAC, see table 6.2, section 9, DMA Controller (DMAC), and section 10, Data Transfer Controller (DTC).

Figure 6.6 shows a block diagram of the DTC, DMAC, and interrupt controller.



**Figure 6.6 Block Diagram of DTC, DMAC, and Interrupt Controller**

### (1) Selection of Interrupt Sources

The activation source for each DMAC channel is selected by DMRSR. The selected activation source is input to the DMAC through the select circuit. When transfer by an on-chip module interrupt is enabled ( $DTF1 = 1$ ,  $DTF0 = 0$ , and  $DTE = 1$  in DMDR) and the DTA bit in DMDR is set to 1, the interrupt source selected for the DMAC activation source is controlled by the DMAC and cannot be used as a DTC activation source or CPU interrupt source.

Interrupt sources that are not controlled by the DMAC are set for DTC activation sources or CPU interrupt sources by the DTCE bit in DTCERA to DTCERH of the DTC.

Specifying the DISEL bit in MRB of the DTC generates an interrupt request to the CPU by clearing the DTCE bit to 0 after the individual DTC data transfer.

Note that when the DTC performs a predetermined number of data transfers and the transfer counter indicates 0, an interrupt request is made to the CPU by clearing the DTCE bit to 0 after the DTC data transfer.

When the same interrupt source is set as both the DTC and DMAC activation source and CPU interrupt source, the DTC and DMAC must be given priority over the CPU. If the IPSETE bit in CPUPCR is set to 1, the priority is determined according to the IPR setting. Therefore, the CPUP setting or the IPR setting corresponding to the interrupt source must be set to lower than or equal to the DTCP and DMAP setting. If the CPU is given priority over the DTC or DMAC, the DTC or DMAC may not be activated, and the data transfer may not be performed.

## (2) Priority Determination

The DTC activation source is selected according to the default priority, and the selection is not affected by its mask level or priority level. For respective priority levels, see table 8.1, Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs.

## (3) Operation Order

If the same interrupt is selected as both the DTC activation source and CPU interrupt source, the CPU interrupt exception handling is performed after the DTC data transfer. If the same interrupt is selected as the DTC or DMAC activation source or CPU interrupt source, respective operations are performed independently.

Table 6.6 lists the selection of interrupt sources and interrupt source clear control by setting the DTA bit in DMDR of the DMAC, the DTCE bit in DTCERA to DTCERH of the DTC, and the DISEL bit in MRB of the DTC.

**Table 6.6 Interrupt Source Selection and Clear Control**

DMAC Setting DTA	DTC Setting		Interrupt Source Selection/Clear Control		
	DTCE	DISEL	DMAC	DTC	CPU
0	0	*	O	X	√
		1	O	√	X
		0	O	O	√
1	*	1	√	X	X
		*			

[Legend]

- √: The corresponding interrupt is used. The interrupt source is cleared.  
(The interrupt source flag must be cleared in the CPU interrupt handling routine.)
- O: The corresponding interrupt is used. The interrupt source is not cleared.
- X: The corresponding interrupt is not available.
- \*: Don't care.

## (4) Usage Note

The interrupt sources of the SCI, and A/D converter are cleared according to the setting shown in table 6.6, when the DTC or DMAC reads/writes the prescribed register.

To initiate multiple channels for the DTC with the same interrupt, the same priority (DTCP = DMAP) should be assigned.

## 6.7 CPU Priority Control Function Over DTC and DMAC

The interrupt controller has a function to control the priority among the DTC, DMAC, and the CPU by assigning different priority levels to the DTC, DMAC, and CPU. Since the priority level can automatically be assigned to the CPU on an interrupt occurrence, it is possible to execute the CPU interrupt exception handling prior to the DTC or DMAC transfer.

The priority level of the CPU is assigned by bits CPUP2 to CPUP0 in CPUPCR. The priority level of the DTC is assigned by bits DTCP2 to DTCP0 in CPUPCR. The priority level of the DMAC is assigned by bits DMAP2 to DMAP0 in DMDR for each channel.

The priority control function over the DTC and DMAC is enabled by setting the CPUPCE bit in CPUPCR to 1. When the CPUPCE bit is 1, the DTC and DMAC activation sources are controlled according to the respective priority levels.

The DTC activation source is controlled according to the priority level of the CPU indicated by bits CPUP2 to CPUP0 and the priority level of the DTC indicated by bits DTCP2 to DTCP0. If the CPU has priority, the DTC activation source is held. The DTC is activated when the condition by which the activation source is held is cancelled (CPUPCE = 1 and value of bits CPUP2 to CPUP0 is greater than that of bits DTCP2 to DTCP0). The priority level of the DTC is assigned by the DTCP2 to DTCP0 bits regardless of the activation source.

For the DMAC, the priority level can be specified for each channel. The DMAC activation source is controlled according to the priority level of each DMAC channel indicated by bits DMAP2 to DMAP0 and the priority level of the CPU. If the CPU has priority, the DMAC activation source is held. The DMAC is activated when the condition by which the activation source is held is cancelled (CPUPCE = 1 and value of bits CPUP2 to CPUP0 is greater than that of bits DMAP2 to DMAP0). If different priority levels are specified for channels, the channels of the higher priority levels continue transfer and the activation sources for the channels of lower priority levels than that of the CPU are held.

There are two methods for assigning the priority level to the CPU by the IPSETE bit in CPUPCR. Setting the IPSETE bit to 1 enables a function to automatically assign the value of the interrupt mask bit of the CPU to the CPU priority level. Clearing the IPSETE bit to 0 disables the function to automatically assign the priority level. Therefore, the priority level is assigned directly by software rewriting bits CPUP2 to CPUP0. Even if the IPSETE bit is 1, the priority level of the CPU is software assignable by rewriting the interrupt mask bit of the CPU (I bit in CCR or I2 to I0 bits in EXR).

The priority level which is automatically assigned when the IPSETE bit is 1 differs according to the interrupt control mode.

In interrupt control mode 0, the I bit in CCR of the CPU is reflected in bit CPUP2. Bits CPUP1 and CPUP0 are fixed 0. In interrupt control mode 2, the values of bits I2 to I0 in EXR of the CPU are reflected in bits CPUP2 to CPUP0.

Table 6.7 shows the CPU priority control.

**Table 6.7 CPU Priority Control**

Interrupt Control Mode	Interrupt Priority	Interrupt Mask Bit	IPSETE in CPUPCR	Control Status	
				CPUP2 to CPUP0	Updating of CPUP2 to CPUP0
0	Default	I = any	0	B'111 to B'000	Enabled
		I = 0	1	B'000	Disabled
		I = 1		B'100	
2	IPR setting	I2 to I0	0	B'111 to B'000	Enabled
			1	I2 to I0	Disabled

Table 6.8 shows a setting example of the priority control function over the DTC and DMAC and the transfer request control state. A priority level can be independently set to each DMAC channel, but the table only shows one channel for example. Transfers through the DMAC channels can be separately controlled by assigning different priority levels for channels.

**Table 6.8 Example of Priority Control Function Setting and Control State**

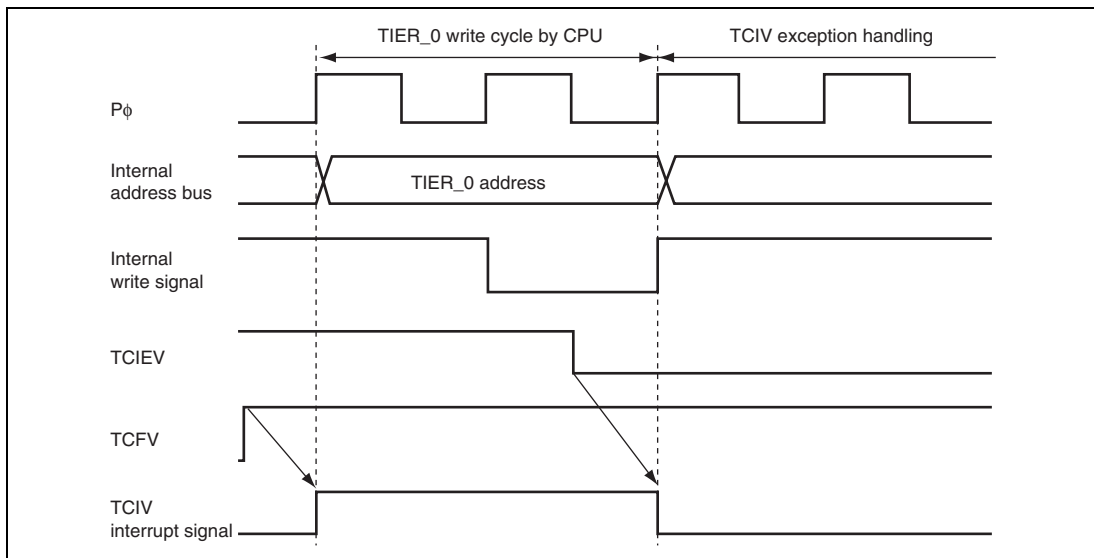
Interrupt Control Mode	CPUPCE in CPUPCR	CPUP2 to CPUP0	DTCP2 to DTCP0	DMAP2 to DMAP0	Transfer Request Control State	
					DTC	DMAC
0	0	Any	Any	Any	Enabled	Enabled
	1	B'000	B'000	B'000	Enabled	Enabled
		B'100	B'000	B'000	Masked	Masked
		B'100	B'000	B'011	Masked	Masked
		B'100	B'111	B'101	Enabled	Enabled
		B'000	B'111	B'101	Enabled	Enabled
2	0	Any	Any	Any	Enabled	Enabled
	1	B'000	B'000	B'000	Enabled	Enabled
		B'000	B'011	B'101	Enabled	Enabled
		B'011	B'011	B'101	Enabled	Enabled
		B'100	B'011	B'101	Masked	Enabled
		B'101	B'011	B'101	Masked	Enabled
		B'110	B'011	B'101	Masked	Masked
		B'111	B'011	B'101	Masked	Masked
		B'101	B'011	B'101	Masked	Enabled
		B'101	B'110	B'101	Enabled	Enabled

## 6.8 Usage Notes

### 6.8.1 Conflict between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to mask the interrupt, the masking becomes effective after execution of the instruction.

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 with priority over that interrupt, interrupt exception handling will be executed for the interrupt with priority, and another interrupt will be ignored. The same also applies when an interrupt source flag is cleared to 0. Figure 6.7 shows an example in which the TCIEV bit in TIER of the TPU is cleared to 0. The above conflict will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.



**Figure 6.7 Conflict between Interrupt Generation and Disabling**

Similarly, when an interrupt is requested immediately before the DTC enable bit is changed to activate the DTC, DTC activation and the interrupt exception handling by the CPU are both executed. When changing the DTC enable bit, make sure that an interrupt is not requested.

### 6.8.2 Instructions that Disable Interrupts

Instructions that disable interrupts immediately after execution 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.

### 6.8.3 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, and for a period of writing to the registers of the interrupt controller.

### 6.8.4 Interrupts during Execution of EEPMOV Instruction

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

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

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual 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
```

### 6.8.5 Interrupts during Execution of MOVMD and MOVSD Instructions

With the MOVMD or MOVSD instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual transfer cycle. The PC value saved on the stack in this case is the address of the MOVMD or MOVSD instruction. The transfer of the remaining data is resumed after returning from the interrupt handling routine.



### **6.8.6 Interrupts of Peripheral Modules**

To clear an interrupt source flag by the CPU using an interrupt function of a peripheral module, the flag must be read from after clearing within the interrupt processing routine. This makes the request signal synchronized with the peripheral module clock. For details, refer to section 23.5.1, Notes on Clock Pulse Generator.



## Section 7 User Break Controller (UBC)

The user break controller (UBC) generates a UBC break interrupt request each time the state of the program counter matches a specified break condition. The UBC break interrupt is a non-maskable interrupt and is always accepted, regardless of the interrupt control mode and the state of the interrupt mask bit of the CPU.

For each channel, the break control register (BRCR) and break address register (BAR) are used to specify the break condition as a combination of address bits and type of bus cycle.

Four break conditions are independently specifiable on four channels, A to D.

### 7.1 Features

- Number of break channels: four (channels A, B, C, and D)
- Break comparison conditions (each channel)
  - Address
  - Bus master (CPU cycle)
  - Bus cycle (instruction execution (PC break))
- After a break condition is satisfied, UBC break interrupt exception handling is executed immediately before execution of the instruction fetched from the specified address (PC break).
- Module stop state specifiable

## 7.2 Block Diagram

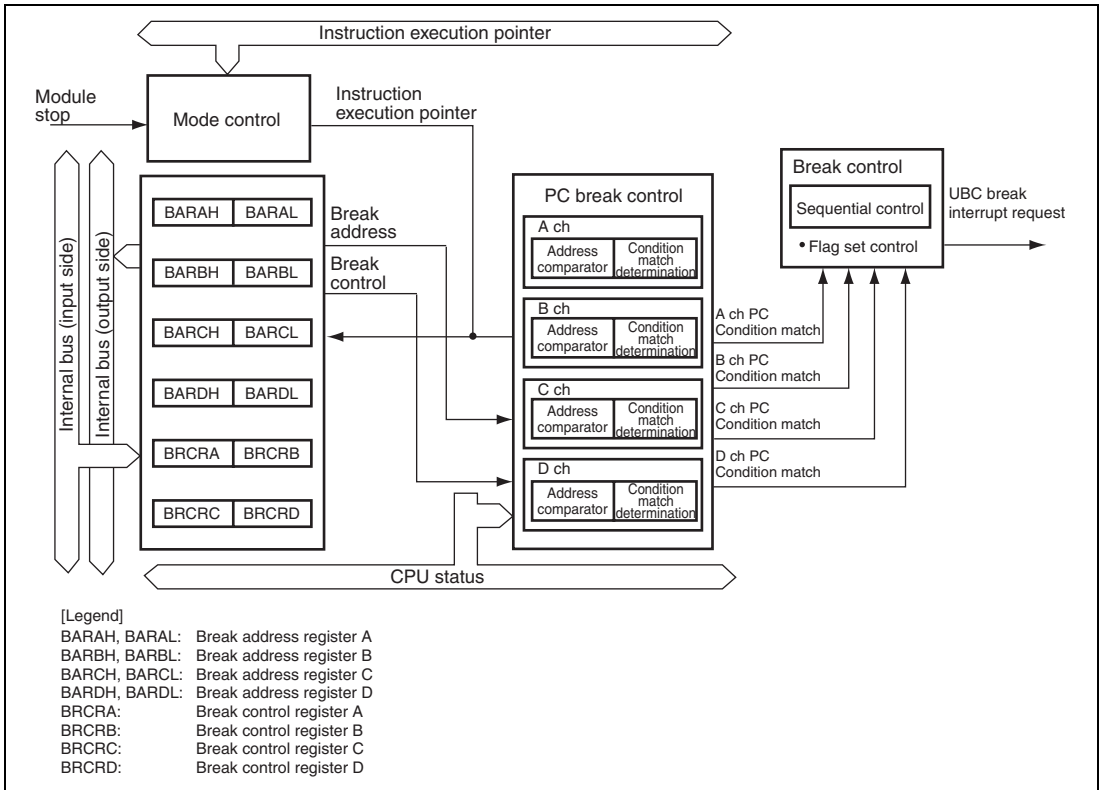


Figure 7.1 Block Diagram of UBC

## 7.3 Register Descriptions

Table 7.1 lists the register configuration of the UBC.

**Table 7.1 Register Configuration**

Register Name	Abbreviation	R/W	Initial Value	Address	Access Size
Break address register A	BARAH	R/W	H'0000	H'FFA00	16
	BARAL	R/W	H'0000	H'FFA02	16
Break address mask register A	BAMRAH	R/W	H'0000	H'FFA04	16
	BAMRAL	R/W	H'0000	H'FFA06	16
Break address register B	BARBH	R/W	H'0000	H'FFA08	16
	BARBL	R/W	H'0000	H'FFA0A	16
Break address mask register B	BAMRBH	R/W	H'0000	H'FFA0C	16
	BAMRBL	R/W	H'0000	H'FFA0E	16
Break address register C	BARCH	R/W	H'0000	H'FFA10	16
	BARCL	R/W	H'0000	H'FFA12	16
Break address mask register C	BAMRCH	R/W	H'0000	H'FFA14	16
	BAMRCL	R/W	H'0000	H'FFA16	16
Break address register D	BARDH	R/W	H'0000	H'FFA18	16
	BARDL	R/W	H'0000	H'FFA1A	16
Break address mask register D	BAMRDH	R/W	H'0000	H'FFA1C	16
	BAMRDL	R/W	H'0000	H'FFA1E	16
Break control register A	BRCRA	R/W	H'0000	H'FFA28	8/16
Break control register B	BRCRB	R/W	H'0000	H'FFA2C	8/16
Break control register C	BRCRC	R/W	H'0000	H'FFA30	8/16
Break control register D	BRCRD	R/W	H'0000	H'FFA34	8/16

### 7.3.1 Break Address Register n (BARA, BARB, BARC, BARD)

Each break address register n (BARn) consists of break address register nH (BARnH) and break address register nL (BARnL). Together, BARnH and BARnL specify the address used as a break condition on channel n of the UBC.

BARnH

Bit:	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	BARn31	BARn30	BARn29	BARn28	BARn27	BARn26	BARn25	BARn24	BARn23	BARn22	BARn21	BARn20	BARn19	BARn18	BARn17	BARn16
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

BARnL

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BARn15	BARn14	BARn13	BARn12	BARn11	BARn10	BARn9	BARn8	BARn7	BARn6	BARn5	BARn4	BARn3	BARn2	BARn1	BARn0
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

- BARnH

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	BARn31 to BARn16	All 0	R/W	Break Address n31 to 16 These bits hold the upper bit values (bits 31 to 16) for the address break-condition on channel n.

[Legend]

n = Channels A to D

- BARnL

Bit	Bit Name	Initial Value	R/W	Description
15 to 0	BARn15 to BARn0	All 0	R/W	Break Address n15 to 0 These bits hold the lower bit values (bits 15 to 0) for the address break-condition on channel n.

[Legend]

n = Channels A to D

### 7.3.2 Break Address Mask Register n (BAMRA, BAMRB, BAMRC, BAMRD)

Be sure to write H'FF00 0000 to break address mask register n (BAMRn). Operation is not guaranteed if another value is written here.

BAMRnH

Bit:	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	BAMRn31	BAMRn30	BAMRn29	BAMRn28	BAMRn27	BAMRn26	BAMRn25	BAMRn24	BAMRn23	BAMRn22	BAMRn21	BAMRn20	BAMRn19	BAMRn18	BAMRn17	BAMRn16
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

BAMRnL

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BAMRn15	BAMRn14	BAMRn13	BAMRn12	BAMRn11	BAMRn10	BAMRn9	BAMRn8	BAMRn7	BAMRn6	BAMRn5	BAMRn4	BAMRn3	BAMRn2	BAMRn1	BAMRn0
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

- BAMRnH

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	BAMRn31 to BAMRn16	All 0	R/W	Break Address Mask n31 to 16 Be sure to write H'FF00 here before setting a break condition in the break control register.

[Legend]

n = Channels A to D

- BAMRnL

Bit	Bit Name	Initial Value	R/W	Description
15 to 0	BAMRn15 to BAMRn0	All 0	R/W	Break Address Mask n15 to 0 Be sure to write H'0000 here before setting a break condition in the break control register.

[Legend]

n = Channels A to D

### 7.3.3 Break Control Register n (BRCRA, BRCRB, BRCRC, BRCRD)

BRCRA, BRCRB, BRCRC, and BRCRD are used to specify and control conditions for channels A, B, C, and D of the UBC.

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	CMFCPn	—	CPn2	CPn1	CPn0	—	—	—	IDn1	IDn0	RWn1	RWn0	—	—
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

[Legend]  
n = Channels A to D

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R/W	Reserved
14	—	0	R/W	These bits are always read as 0. The write value should always be 0.
13	CMFCPn	0	R/W	Condition Match CPU Flag UBC break source flag that indicates satisfaction of a specified CPU bus cycle condition. 0: The CPU cycle condition for channel n break requests has not been satisfied. 1: The CPU cycle condition for channel n break requests has been satisfied.
12	—	0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
11	CPn2	0	R/W	CPU Cycle Select
10	CPn1	0	R/W	These bits select CPU cycles as the bus cycle break condition for the given channel.
9	CPn0	0	R/W	000: Break requests will not be generated. 001: The bus cycle break condition is CPU cycles. 01x: Setting prohibited 1xx: Setting prohibited
8	—	0	R/W	Reserved
7	—	0	R/W	These bits are always read as 0. The write value should always be 0.
6	—	0	R/W	



Bit	Bit Name	Initial Value	R/W	Description
5	IDn1	0	R/W	Break Condition Select
4	IDn0	0	R/W	These bits select the PC break as the source of UBC break interrupt requests for the given channel. 00: Break requests will not be generated. 01: UBC break condition is the PC break. 1x: Setting prohibited
3	RWn1	0	R/W	Read Select
2	RWn0	0	R/W	These bits select read cycles as the bus cycle break condition for the given channel. 00: Break requests will not be generated. 01: The bus cycle break condition is read cycles. 1x: Setting prohibited
1	—	0	R/W	Reserved
0	—	0	R/W	These bits are always read as 0. The write value should always be 0.

[Legend]

n = Channels A to D

## 7.4 Operation

The UBC does not detect condition matches in standby states (sleep mode, all-module clock stop mode, software standby, deep software standby, and hardware standby modes).

### 7.4.1 Setting of Break Control Conditions

1. The address condition for the break is set in break address register n (BARn). A mask for the address is set in break address mask register n (BAMRn).
2. The bus and break conditions are set in break control register n (BRCRn). Bus conditions consist of CPU cycle, PC break, and reading. Condition comparison is not performed when the CPU cycle setting is CPn = B'000, the PC break setting is IDn = B'00, or the read setting is RWn = B'00.
3. The condition match CPU flag (CMFCPn) is set in the event of a break condition match on the corresponding channel. These flags are set when the break condition matches but are not cleared when it no longer does. To confirm setting of the same flag again, read the flag once from the break interrupt handling routine, and then write 0 to it (the flag is cleared by writing 0 to it after reading it as 1).

[Legend]

n = Channels A to D

### 7.4.2 PC Break

1. When specifying a PC break, specify the address as the first address of the required instruction. If the address for a PC break condition is not the first address of an instruction, a break will never be generated.
2. The break occurs after fetching and execution of the target instruction have been confirmed. In cases of contention between a break before instruction execution and a user maskable interrupt, priority is given to the break before instruction execution.
3. A break will not be generated even if a break before instruction execution is set in a delay slot.
4. The PC break condition is generated by specifying CPU cycles as the bus condition in break control register n (BRCRn.CPn0 = 1), PC break as the break condition (IDn0 = 1), and read cycles as the bus-cycle condition (RWn0 = 1).

[Legend]

n = Channels A to D

### 7.4.3 Condition Match Flag

Condition match flags are set when the break conditions match. The condition match flags of the UBC are listed in table 7.2.

**Table 7.2 List of Condition Match Flags**

<b>Register</b>	<b>Flag Bit</b>	<b>Source</b>
BRCRA	CMFCPA (bit 13)	Indicates that the condition matches in the CPU cycle for channel A
BRCRB	CMFCPB (bit 13)	Indicates that the condition matches in the CPU cycle for channel B
BRCRC	CMFCPC (bit 13)	Indicates that the condition matches in the CPU cycle for channel C
BRCRD	CMFCPD (bit 13)	Indicates that the condition matches in the CPU cycle for channel D

## 7.5 Usage Notes

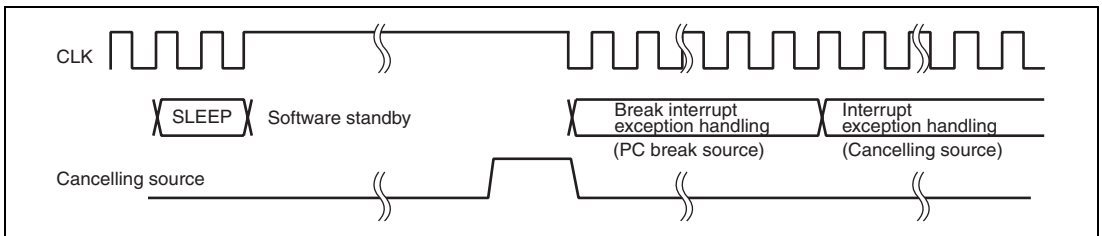
### 1. PC break usage note

- Contention between a SLEEP instruction (to place the chip in the sleep state or on software standby) and PC break

If a break before a PC break instruction is set for the instruction after a SLEEP instruction and the SLEEP instruction is executed with the SSBY bit cleared to 0, break interrupt exception handling is executed without sleep mode being entered. In this case, the instruction after the SLEEP instruction is executed after the RTE instruction.

When the SSBY bit is set to 1, break interrupt exception handling is executed after the oscillation settling time has elapsed subsequent to the transition to software standby mode.

When an interrupt is the canceling source, interrupt exception handling is executed after the RTE instruction, and the instruction following the SLEEP instruction is then executed.

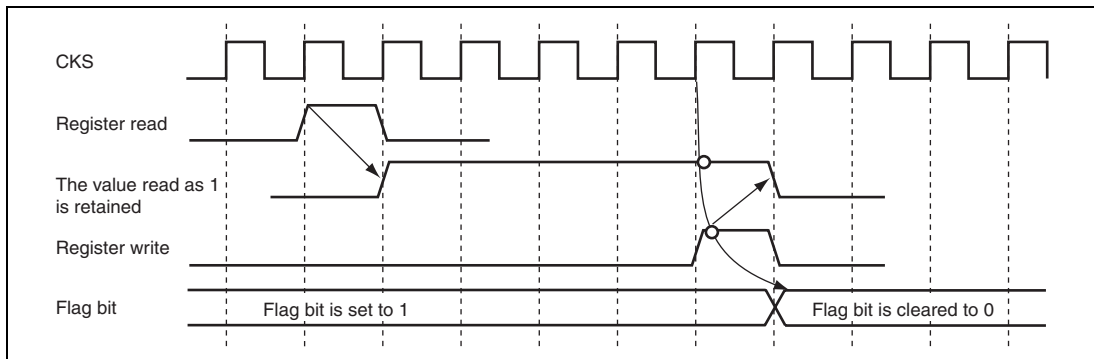


**Figure 7.2 Contention between SLEEP Instruction (Software Standby) and PC Break**

### 2. Prohibition on Setting of PC Break

- Setting of a UBC break interrupt for program within the UBC break interrupt handling routine is prohibited.

### 3. The procedure for clearing a UBC flag bit (condition match flag) is shown below. A flag bit is cleared by writing 0 to it after reading it as 1. As the register that contains the flag bits is accessible in byte units, bit manipulation instructions can be used.



**Figure 7.3 Flag Bit Clearing Sequence (Condition Match Flag)**

4. The valid range of break addresses in the MCU and CPU address modes is given in table 7.3.

— MCU operating mode/CPU mode and valid address range

In setting break addresses, MCU address mode and CPU mode need to be taken into account as shown below.

The mask must be set in the address mask register.

**Table 7.3 Valid Range of Break/Branch Addresses for MCU/CPU Address Modes**

	Advanced Mode	
	256 MB	16 MB
PC break address	The lower 24 bits are valid and the upper 8 bits are masked.	

5. If an illegal instruction is executed after setting break conditions for the UBC, an unexpected UBC break interrupt may occur depending on the value of the program counter and the internal bus cycle.



## Section 8 Bus Controller (BSC)

This LSI has an on-chip bus controller (BSC) that manages the external address space divided into eight areas.

The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters; CPU, DMAC, and DTC.

### 8.1 Features

- Manages external address space in area units  
 Manages the external address space divided into eight areas  
 Chip select signals ( $\overline{CS0}$  to  $\overline{CS7}$ ) can be output for each area  
 Bus specifications can be set independently for each area  
 8-bit access or 16-bit access can be selected for each area  
 Burst ROM, byte control SRAM, or address/data multiplexed I/O interface can be set  
 An endian conversion function is provided to connect a device of little endian
- Basic bus interface  
 This interface can be connected to the SRAM and ROM  
 2-state access or 3-state access can be selected for each area  
 Program wait cycles can be inserted for each area  
 Wait cycles can be inserted by the  $\overline{WAIT}$  pin.  
 Extension cycles can be inserted while  $\overline{CSn}$  is asserted for each area (n = 0 to 7)  
 The negation timing of the read strobe signal ( $\overline{RD}$ ) can be modified
- Byte control SRAM interface  
 Byte control SRAM interface can be set for areas 0 to 7  
 The SRAM that has a byte control pin can be directly connected
- Burst ROM interface  
 Burst ROM interface can be set for areas 0 and 1  
 Burst ROM interface parameters can be set independently for areas 0 and 1
- Address/data multiplexed I/O interface  
 Address/data multiplexed I/O interface can be set for areas 3 to 7

- Idle cycle insertion

Idle cycles can be inserted between external read accesses to different areas

Idle cycles can be inserted before the external write access after an external read access

Idle cycles can be inserted before the external read access after an external write access

Idle cycles can be inserted before the external access after a DMAC single address transfer (write access)

- Write buffer function

External write cycles and internal accesses can be executed in parallel

Write accesses to the on-chip peripheral module and on-chip memory accesses can be executed in parallel

DMAC single address transfers and internal accesses can be executed in parallel

- External bus release function

- Bus arbitration function

Includes a bus arbiter that arbitrates bus mastership among the CPU, DMAC, DTC, and external bus master

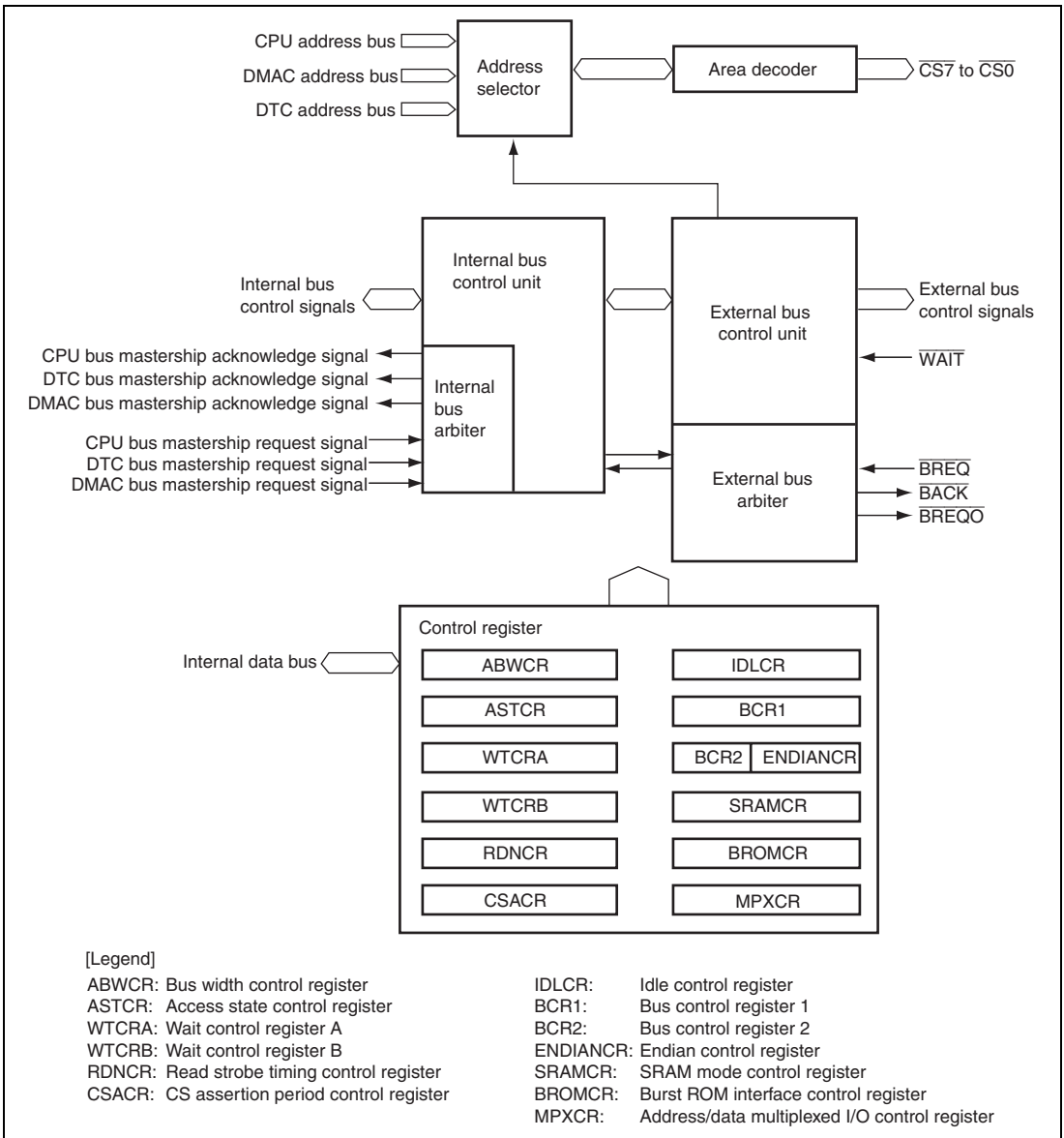
- Multi-clock function

The internal peripheral functions can be operated in synchronization with the peripheral module clock ( $P\phi$ ). Accesses to the external address space can be operated in synchronization with the external bus clock ( $B\phi$ ).

- The bus start ( $\overline{BS}$ ) and read/write ( $RD/\overline{WR}$ ) signals can be output.



A block diagram of the bus controller is shown in figure 8.1.



**Figure 8.1 Block Diagram of Bus Controller**

## 8.2 Register Descriptions

The bus controller has the following registers.

- Bus width control register (ABWCR)
- Access state control register (ASTCR)
- Wait control register A (WTCRA)
- Wait control register B (WTCRB)
- Read strobe timing control register (RDNCR)
- $\overline{\text{CS}}$  assertion period control register (CSACR)
- Idle control register (IDLCR)
- Bus control register 1 (BCR1)
- Bus control register 2 (BCR2)
- Endian control register (ENDIANCR)
- SRAM mode control register (SRAMCR)
- Burst ROM interface control register (BROMCR)
- Address/data multiplexed I/O control register (MPXCR)

### 8.2.1 Bus Width Control Register (ABWCR)

ABWCR specifies the data bus width for each area in the external address space.

Bit	15	14	13	12	11	10	9	8
Bit Name	ABWH7	ABWH6	ABWH5	ABWH4	ABWH3	ABWH2	ABWH1	ABWH0
Initial Value	1	1	1	1	1	1	1	1/0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1	ABWL0
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

Note: \* Initial value at 16-bit bus initiation is H'FEFF, and that at 8-bit bus initiation is H'FFFF.

Bit	Bit Name	Initial Value* <sup>1</sup>	R/W	Description
15	ABWH7	1	R/W	Area 7 to 0 Bus Width Control
14	ABWH6	1	R/W	These bits select whether the corresponding area is to be designated as 8-bit access space or 16-bit access space.
13	ABWH5	1	R/W	
12	ABWH4	1	R/W	ABWHn ABWLn (n = 7 to 0)
11	ABWH3	1	R/W	× 0: Setting prohibited
10	ABWH2	1	R/W	0 1: Area n is designated as 16-bit access space
9	ABWH1	1	R/W	1 1: Area n is designated as 8-bit access space* <sup>2</sup>
8	ABWL0	1/0	R/W	
7	ABWL7	1	R/W	
6	ABWL6	1	R/W	
5	ABWL5	1	R/W	
4	ABWL4	1	R/W	
3	ABWL3	1	R/W	
2	ABWL2	1	R/W	
1	ABWL1	1	R/W	
0	ABWL0	1	R/W	

[Legend]

×: Don't care

- Notes:
1. Initial value at 16-bit bus initiation is H'FEFF, and that at 8-bit bus initiation is H'FFFF.
  2. An address space specified as byte control SRAM interface must not be specified as 8-bit access space.

## 8.2.2 Access State Control Register (ASTCR)

ASTCR designates each area in the external address space as either 2-state access space or 3-state access space and enables/disables wait cycle insertion.

Bit	15	14	13	12	11	10	9	8
Bit Name	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
Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	AST7	1	R/W	Area 7 to 0 Access State Control
14	AST6	1	R/W	These bits select whether the corresponding area is to be designated as 2-state access space or 3-state access space. Wait cycle insertion is enabled or disabled at the same time.
13	AST5	1	R/W	
12	AST4	1	R/W	0: Area n is designated as 2-state access space Wait cycle insertion in area n access is disabled
11	AST3	1	R/W	
10	AST2	1	R/W	1: Area n is designated as 3-state access space Wait cycle insertion in area n access is enabled (n = 7 to 0)
9	AST1	1	R/W	
8	AST0	1	R/W	
7 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

### 8.2.3 Wait Control Registers A and B (WTCRA, WTCRB)

WTCRA and WTCRB select the number of program wait cycles for each area in the external address space.

#### • WTCRA

Bit	15	14	13	12	11	10	9	8
Bit Name	—	W72	W71	W70	—	W62	W61	W60
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	—	W52	W51	W50	—	W42	W41	W40
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

#### • WTCRB

Bit	15	14	13	12	11	10	9	8
Bit Name	—	W32	W31	W30	—	W22	W21	W20
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	—	W12	W11	W10	—	W02	W01	W00
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

- WTCRA

Bit	Bit Name	Initial Value	R/W	Description		
15	—	0	R	Reserved This is a read-only bit and cannot be modified.		
14	W72	1	R/W	Area 7 Wait Control 2 to 0		
13	W71	1	R/W	These bits select the number of program wait cycles when accessing area 7 while bit AST7 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted		
12	W70	1	R/W			
11	—	0	R		Reserved This is a read-only bit and cannot be modified.	
10	W62	1	R/W		Area 6 Wait Control 2 to 0	
9	W61	1	R/W		These bits select the number of program wait cycles when accessing area 6 while bit AST6 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted	
8	W60	1	R/W			
7	—	0	R			Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description	
6	W52	1	R/W	Area 5 Wait Control 2 to 0	
5	W51	1	R/W	These bits select the number of program wait cycles when accessing area 5 while bit AST5 in ASTCR is 1. 000: Program cycle wait not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted	
4	W50	1	R/W		
3	—	0	R		Reserved This is a read-only bit and cannot be modified.
2	W42	1	R/W		Area 4 Wait Control 2 to 0
1	W41	1	R/W		These bits select the number of program wait cycles when accessing area 4 while bit AST4 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
0	W40	1	R/W		

- WTCRB

Bit	Bit Name	Initial Value	R/W	Description		
15	—	0	R	Reserved This is a read-only bit and cannot be modified.		
14	W32	1	R/W	Area 3 Wait Control 2 to 0		
13	W31	1	R/W	These bits select the number of program wait cycles when accessing area 3 while bit AST3 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted		
12	W30	1	R/W			
11	—	0	R		Reserved This is a read-only bit and cannot be modified.	
10	W22	1	R/W		Area 2 Wait Control 2 to 0	
9	W21	1	R/W		These bits select the number of program wait cycles when accessing area 2 while bit AST2 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted	
8	W20	1	R/W			
7	—	0	R			Reserved This is a read-only bit and cannot be modified.



Bit	Bit Name	Initial Value	R/W	Description	
6	W12	1	R/W	Area 1 Wait Control 2 to 0	
5	W11	1	R/W	These bits select the number of program wait cycles when accessing area 1 while bit AST1 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted	
4	W10	1	R/W		
3	—	0	R		Reserved This is a read-only bit and cannot be modified.
2	W02	1	R/W		Area 0 Wait Control 2 to 0
1	W01	1	R/W		These bits select the number of program wait cycles when accessing area 0 while bit AST0 in ASTCR is 1. 000: Program wait cycle not inserted 001: 1 program wait cycle inserted 010: 2 program wait cycles inserted 011: 3 program wait cycles inserted 100: 4 program wait cycles inserted 101: 5 program wait cycles inserted 110: 6 program wait cycles inserted 111: 7 program wait cycles inserted
0	W00	1	R/W		

## 8.2.4 Read Strobe Timing Control Register (RDNCR)

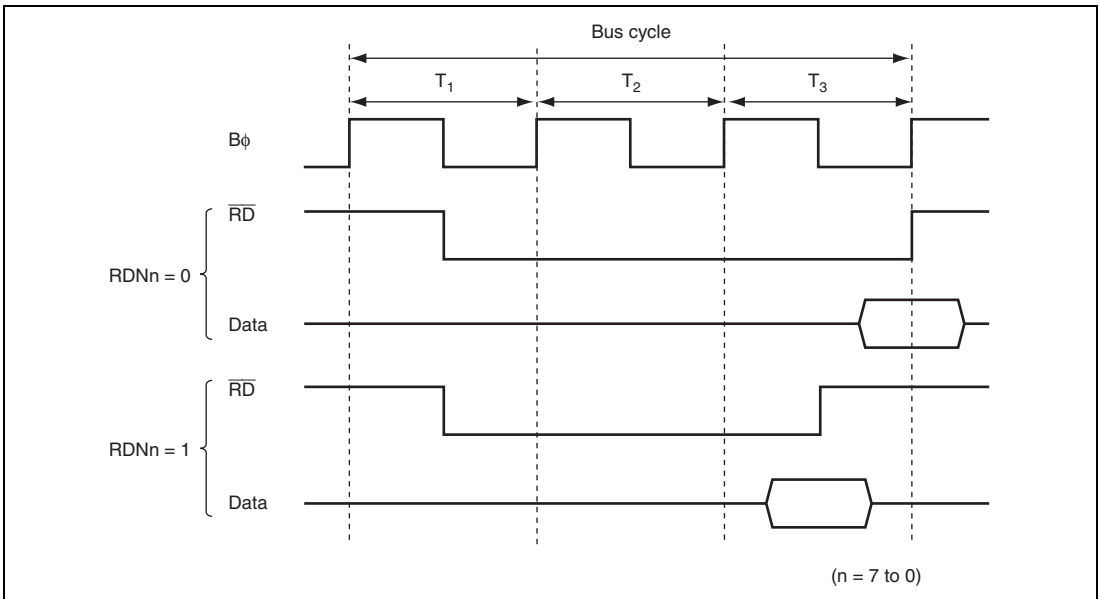
RDNCR selects the negation timing of the read strobe signal ( $\overline{RD}$ ) when reading the external address spaces specified as a basic bus interface or the address/data multiplexed I/O interface.

Bit	15	14	13	12	11	10	9	8
Bit Name	RDN7	RDN6	RDN5	RDN4	RDN3	RDN2	RDN1	RDN0
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	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	RDN7	0	R/W	Read Strobe Timing Control
14	RDN6	0	R/W	RDN7 to RDN0 set the negation timing of the read strobe in a corresponding area read access.
13	RDN5	0	R/W	
12	RDN4	0	R/W	As shown in figure 8.2, the read strobe for an area for which the RDNn bit is set to 1 is negated one half-cycle earlier than that for an area for which the RDNn bit is cleared to 0. The read data setup and hold time are also given one half-cycle earlier.
11	RDN3	0	R/W	
10	RDN2	0	R/W	
9	RDN1	0	R/W	0: In an area n read access, the $\overline{RD}$ signal is negated at the end of the read cycle 1: In an area n read access, the $\overline{RD}$ signal is negated one half-cycle before the end of the read cycle (n = 7 to 0)
8	RDN0	0	R/W	
7 to 0	—	All 0	R	Reserved

These are read-only bits and cannot be modified.

- Notes:
1. In an external address space which is specified as byte control SRAM interface, the RDNCR setting is ignored and the same operation when RDNn = 1 is performed.
  2. In an external address space which is specified as burst ROM interface, the RDNCR setting is ignored during CPU read accesses and the same operation when RDNn = 0 is performed.



**Figure 8.2 Read Strobe Negation Timing (Example of 3-State Access Space)**

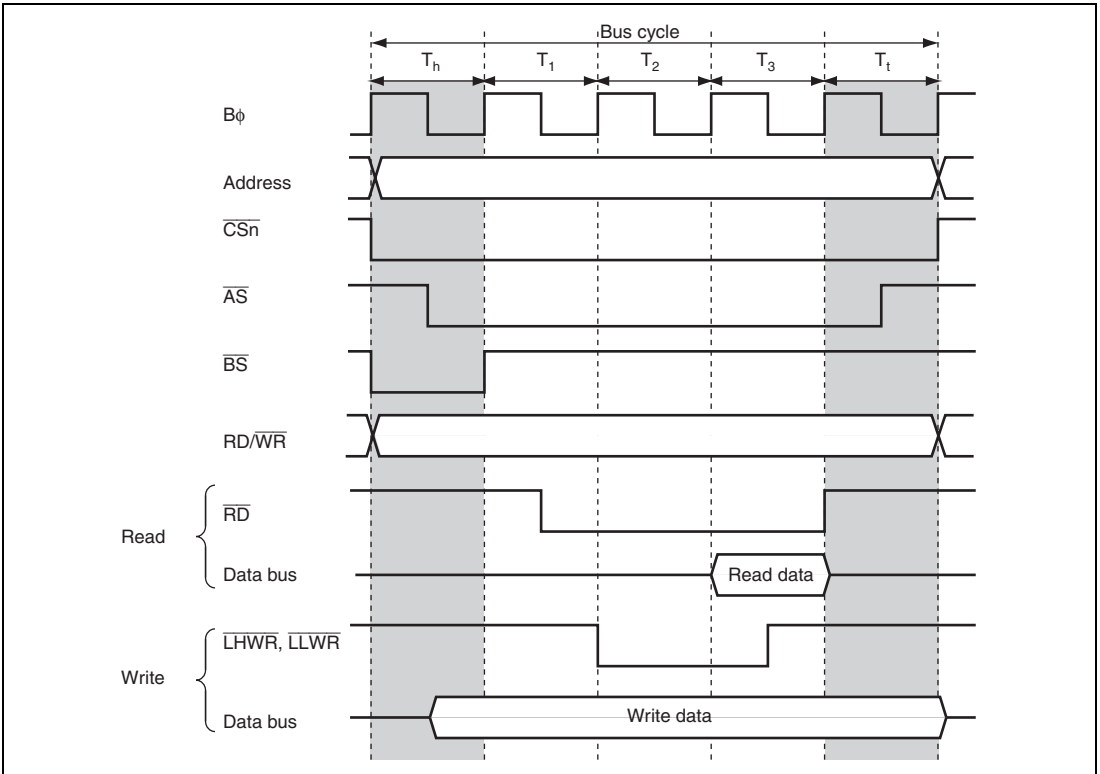
### 8.2.5 $\overline{\text{CS}}$ Assertion Period Control Registers (CSACR)

CSACR selects whether or not the assertion periods of the chip select signals ( $\overline{\text{CSn}}$ ) and address signals for the basic bus, byte-control SRAM, burst ROM, and address/data multiplexed I/O interface are to be extended. Extending the assertion period of the  $\overline{\text{CSn}}$  and address signals allows the setup time and hold time of read strobe ( $\overline{\text{RD}}$ ) and write strobe ( $\overline{\text{LHWR/LLWR}}$ ) to be assured and to make the write data setup time and hold time for the write strobe become flexible.

Bit	15	14	13	12	11	10	9	8
Bit Name	CSXH7	CSXH6	CSXH5	CSXH4	CSXH3	CSXH2	CSXH1	CSXH0
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	7	6	5	4	3	2	1	0
Bit Name	CSXT7	CSXT6	CSXT5	CSXT4	CSXT3	CSXT2	CSXT1	CSXT0
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	Bit Name	Initial Value	R/W	Description
15	CSXH7	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Control 1
14	CSXH6	0	R/W	These bits specify whether or not the Th cycle is to be inserted (see figure 8.3). When an area for which bit CSXHn is set to 1 is accessed, one Th cycle, in which the $\overline{CSn}$ and address signals are asserted, is inserted before the normal access cycle.
13	CSXH5	0	R/W	
12	CSXH4	0	R/W	
11	CSXH3	0	R/W	
10	CSXH2	0	R/W	
9	CSXH1	0	R/W	
8	CSXH0	0	R/W	
7	CSXT7	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Control 2
6	CSXT6	0	R/W	These bits specify whether or not the Tt cycle is to be inserted (see figure 8.3). When an area for which bit CSXTn is set to 1 is accessed, one Tt cycle, in which the $\overline{CSn}$ and address signals are retained, is inserted after the normal access cycle.
5	CSXT5	0	R/W	
4	CSXT4	0	R/W	
3	CSXT3	0	R/W	
2	CSXT2	0	R/W	
1	CSXT1	0	R/W	
0	CSXT0	0	R/W	

Note: \* In burst ROM interface, the CSXTn settings are ignored during CPU read accesses.



**Figure 8.3  $\overline{CS}$  and Address Assertion Period Extension**  
 (Example of Basic Bus Interface, 3-State Access Space, and  $RDN_n = 0$ )

## 8.2.6 Idle Control Register (IDLCR)

IDLCR specifies the idle cycle insertion conditions and the number of idle cycles.

Bit	15	14	13	12	11	10	9	8
Bit Name	IDLS3	IDLS2	IDLS1	IDLS0	IDLCB1	IDLCB0	IDLCA1	IDLCA0
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
Bit	7	6	5	4	3	2	1	0
Bit Name	IDLSEL7	IDLSEL6	IDLSEL5	IDLSEL4	IDLSEL3	IDLSEL2	IDLSEL1	IDLSEL0
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	Bit Name	Initial Value	R/W	Description
15	IDLS3	1	R/W	Idle Cycle Insertion 3 Inserts an idle cycle between the bus cycles when the DMAC single address transfer (write cycle) is followed by external access. 0: No idle cycle is inserted 1: An idle cycle is inserted
14	IDLS2	1	R/W	Idle Cycle Insertion 2 Inserts an idle cycle between the bus cycles when the external write cycle is followed by external read cycle. 0: No idle cycle is inserted 1: An idle cycle is inserted
13	IDLS1	1	R/W	Idle Cycle Insertion 1 Inserts an idle cycle between the bus cycles when the external read cycles of different areas continue. 0: No idle cycle is inserted 1: An idle cycle is inserted

Bit	Bit Name	Initial Value	R/W	Description
12	IDLS0	1	R/W	<p>Idle Cycle Insertion 0</p> <p>Inserts an idle cycle between the bus cycles when the external read cycle is followed by external write cycle.</p> <p>0: No idle cycle is inserted</p> <p>1: An idle cycle is inserted</p>
11	IDLCB1	1	R/W	Idle Cycle State Number Select B
10	IDLCB0	1	R/W	<p>Specifies the number of idle cycles to be inserted for the idle condition specified by IDLS1 and IDLS0.</p> <p>00: No idle cycle is inserted</p> <p>01: 2 idle cycles are inserted</p> <p>00: 3 idle cycles are inserted</p> <p>01: 4 idle cycles are inserted</p>
9	IDLCA1	1	R/W	Idle Cycle State Number Select A
8	IDLCA0	1	R/W	<p>Specifies the number of idle cycles to be inserted for the idle condition specified by IDLS3 to IDLS0.</p> <p>00: 1 idle cycle is inserted</p> <p>01: 2 idle cycles are inserted</p> <p>10: 3 idle cycles are inserted</p> <p>11: 4 idle cycles are inserted</p>
7	IDLSEL7	0	R/W	Idle Cycle Number Select
6	IDLSEL6	0	R/W	<p>Specifies the number of idle cycles to be inserted for each area for the idle insertion condition specified by IDLS1 and IDLS0.</p> <p>0: Number of idle cycles to be inserted for area n is specified by IDLCA1 and IDLCA0.</p> <p>1: Number of idle cycles to be inserted for area n is specified by IDLCB1 and IDLCB0.</p> <p>(n = 7 to 0)</p>
5	IDLSEL5	0	R/W	
4	IDLSEL4	0	R/W	
3	IDLSEL3	0	R/W	
2	IDLSEL2	0	R/W	
1	IDLSEL1	0	R/W	
0	IDLSEL0	0	R/W	

### 8.2.7 Bus Control Register 1 (BCR1)

BCR1 is used for selection of the external bus released state protocol, enabling/disabling of the write data buffer function, and enabling/disabling of the  $\overline{\text{WAIT}}$  pin input.

Bit	15	14	13	12	11	10	9	8
Bit Name	BRLE	BREQOE	—	—	—	—	WDBE	WAITE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DKC	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	BRLE	0	R/W	<p>External Bus Release Enable</p> <p>Enables/disables external bus release.</p> <p>0: External bus release disabled</p> <p><math>\overline{\text{BREQ}}</math>, <math>\overline{\text{BACK}}</math>, and <math>\overline{\text{BREQO}}</math> pins can be used as I/O ports</p> <p>1: External bus release enabled*</p> <p>For details, see section 11, I/O Ports.</p>
14	BREQOE	0	R/W	<p><math>\overline{\text{BREQO}}</math> Pin Enable</p> <p>Controls outputting the bus request signal (<math>\overline{\text{BREQO}}</math>) to the external bus master in the external bus released state when an internal bus master performs an external address space access.</p> <p>0: <math>\overline{\text{BREQO}}</math> output disabled</p> <p><math>\overline{\text{BREQO}}</math> pin can be used as I/O port</p> <p>1: <math>\overline{\text{BREQO}}</math> output enabled</p>



Bit	Bit Name	Initial Value	R/W	Description
13, 12	—	All 0	R	Reserved These are read-only bits and cannot be modified.
11, 10	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
9	WDBE	0	R/W	Write Data Buffer Enable The write data buffer function can be used for an external write cycle and a DMAC single address transfer cycle. The changed setting may not affect an external access immediately after the change. 0: Write data buffer function not used 1: Write data buffer function used
8	WAITE	0	R/W	$\overline{\text{WAIT}}$ Pin Enable Selects enabling/disabling of wait input by the $\overline{\text{WAIT}}$ pin. 0: Wait input by $\overline{\text{WAIT}}$ pin disabled $\overline{\text{WAIT}}$ pin can be used as I/O port 1: Wait input by $\overline{\text{WAIT}}$ pin enabled For details, see section 11, I/O Ports.
7	DKC	0	R/W	$\overline{\text{DACK}}$ Control Selects the timing of DMAC transfer acknowledge signal assertion. 0: $\overline{\text{DACK}}$ signal is asserted at the $B\phi$ falling edge 1: $\overline{\text{DACK}}$ signal is asserted at the $B\phi$ rising edge
6	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
5 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Note: When external bus release is enabled or input by the  $\overline{\text{WAIT}}$  pin is enabled, make sure to set the ICR bit to 1. For details, see section 11, I/O Ports.

### 8.2.8 Bus Control Register 2 (BCR2)

BCR2 is used for bus arbitration control of the CPU, DMAC, and DTC, and enabling/disabling of the write data buffer function to the peripheral modules.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	IBCCS	—	—	—	PWDBE
Initial Value	0	0	0	0	0	0	1	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These are read-only bits and cannot be modified.
5	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
4	IBCCS	0	R/W	Internal Bus Cycle Control Select Selects the internal bus arbiter function. 0: Releases the bus mastership according to the priority 1: Executes the bus cycles alternatively when a CPU bus mastership request conflicts with a DMAC or DTC bus mastership request
3, 2	—	All 0	R	Reserved These are read-only bits and cannot be modified.
1	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
0	PWDBE	0	R/W	Peripheral Module Write Data Buffer Enable Specifies whether or not to use the write data buffer function for the peripheral module write cycles. 0: Write data buffer function not used 1: Write data buffer function used

## 8.2.9 Endian Control Register (ENDIANCR)

ENDIANCR selects the endian format for each area of the external address space. Though the data format of this LSI is big endian, data can be transferred in the little endian format during external address space access.

Note that the data format for the areas used as a program area or a stack area should be big endian.

Bit	7	6	5	4	3	2	1	0
Bit Name	LE7	LE6	LE5	LE4	LE3	LE2	—	—
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	R

Bit	Bit Name	Initial Value	R/W	Description
7	LE7	0	R/W	Little Endian Select
6	LE6	0	R/W	Selects the endian for the corresponding area.
5	LE5	0	R/W	0: Data format of area n is specified as big endian
4	LE4	0	R/W	1: Data format of area n is specified as little endian
3	LE3	0	R/W	(n = 7 to 2)
2	LE2	0	R/W	
1, 0	—	All 0	R	Reserved
These are read-only bits and cannot be modified.				

### 8.2.10 SRAM Mode Control Register (SRAMCR)

SRAMCR specifies the bus interface of each area in the external address space as a basic bus interface or a byte control SRAM interface.

In areas specified as 8-bit access space by ABWCR, the SRAMCR setting is ignored and the byte control SRAM interface cannot be specified.

Bit	15	14	13	12	11	10	9	8
Bit Name	BCSEL7	BCSEL6	BCSEL5	BCSEL4	BCSEL3	BCSEL2	BCSEL1	BCSEL0
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	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	BCSEL7	0	R/W	Byte Control SRAM Interface Select
14	BCSEL6	0	R/W	Selects the bus interface for the corresponding area.
13	BCSEL5	0	R/W	When setting a bit to 1, the bus interface select bits in BROMCR and MPXCR must be cleared to 0.
12	BCSEL4	0	R/W	
11	BCSEL3	0	R/W	0: Area n is basic bus interface
10	BCSEL2	0	R/W	1: Area n is byte control SRAM interface
9	BCSEL1	0	R/W	(n = 7 to 0)
8	BCSEL0	0	R/W	
7 to 0	—	All 0	R	Reserved
These are read-only bits and cannot be modified.				

### 8.2.11 Burst ROM Interface Control Register (BROMCR)

BROMCR specifies the burst ROM interface.

Bit	15	14	13	12	11	10	9	8
Bit Name	BSRM0	BSTS02	BSTS01	BSTS00	—	—	BSWD01	BSWD00
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	BSRM1	BSTS12	BSTS11	BSTS10	—	—	BSWD11	BSWD10
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	BSRM0	0	R/W	Area 0 Burst ROM Interface Select Specifies the area 0 bus interface. To set this bit to 1, clear bit BCSEL0 in SRAMCR to 0. 0: Basic bus interface or byte-control SRAM interface 1: Burst ROM interface
14	BSTS02	0	R/W	Area 0 Burst Cycle Select
13	BSTS01	0	R/W	Specifies the number of burst cycles of area 0
12	BSTS00	0	R/W	000: 1 cycle 001: 2 cycles 010: 3 cycles 011: 4 cycles 100: 5 cycles 101: 6 cycles 110: 7 cycles 111: 8 cycles
11, 10	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
9	BSWD01	0	R/W	Area 0 Burst Word Number Select
8	BSWD00	0	R/W	Selects the number of words in burst access to the area 0 burst ROM interface 00: Up to 4 words (8 bytes) 01: Up to 8 words (16 bytes) 10: Up to 16 words (32 bytes) 11: Up to 32 words (64 bytes)
7	BSRM1	0	R/W	Area 1 Burst ROM Interface Select Specifies the area 1 bus interface as a basic interface or a burst ROM interface. To set this bit to 1, clear bit BCSEL1 in SRAMCR to 0. 0: Basic bus interface or byte-control SRAM interface 1: Burst ROM interface
6	BSTS12	0	R/W	Area 1 Burst Cycle Select
5	BSTS11	0	R/W	Specifies the number of cycles of area 1 burst cycle
4	BSTS10	0	R/W	000: 1 cycle 001: 2 cycles 010: 3 cycles 011: 4 cycles 100: 5 cycles 101: 6 cycles 110: 7 cycles 111: 8 cycles
3, 2	—	All 0	R	Reserved These are read-only bits and cannot be modified.
1	BSWD11	0	R/W	Area 1 Burst Word Number Select
0	BSWD10	0	R/W	Selects the number of words in burst access to the area 1 burst ROM interface 00: Up to 4 words (8 bytes) 01: Up to 8 words (16 bytes) 10: Up to 16 words (32 bytes) 11: Up to 32 words (64 bytes)

## 8.2.12 Address/Data Multiplexed I/O Control Register (MPXCR)

MPXCR specifies the address/data multiplexed I/O interface.

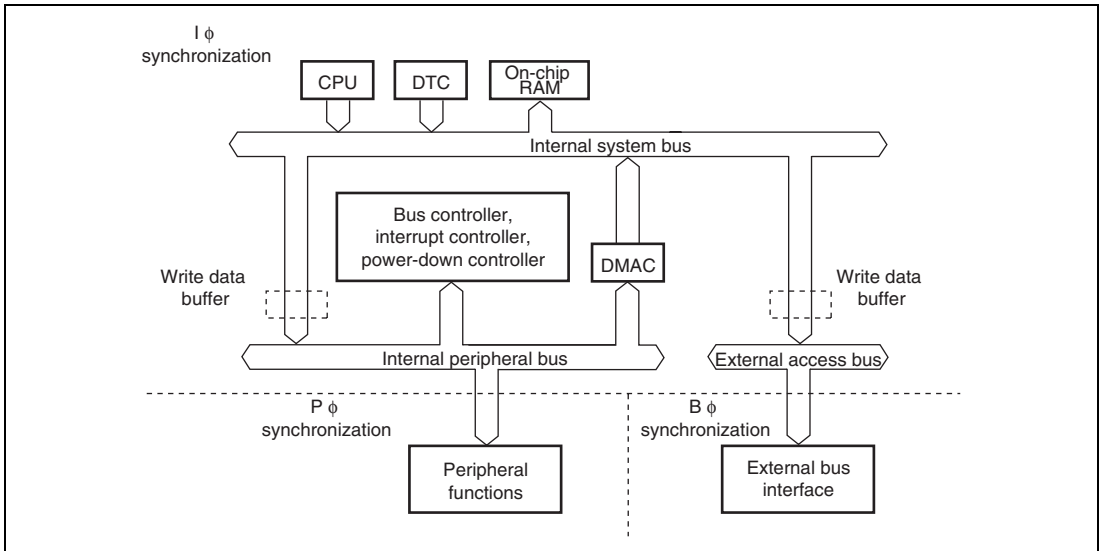
Bit	15	14	13	12	11	10	9	8
Bit Name	MPXE7	MPXE6	MPXE5	MPXE4	MPXE3	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R	R
Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	ADDEX
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	MPXE7	0	R/W	Address/Data Multiplexed I/O Interface Select
14	MPXE6	0	R/W	Specifies the bus interface for the corresponding area.
13	MPXE5	0	R/W	To set this bit to 1, clear the BCSELn bit in SRAMCR to 0.
12	MPXE4	0	R/W	0: Area n is specified as a basic interface or a byte control SRAM interface.
11	MPXE3	0	R/W	1: Area n is specified as an address/data multiplexed I/O interface (n = 7 to 3)
10 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	ADDEX	0	R/W	Address Output Cycle Extension Specifies whether a wait cycle is inserted for the address output cycle of address/data multiplexed I/O interface. 0: No wait cycle is inserted for the address output cycle 1: One wait cycle is inserted for the address output cycle

### 8.3 Bus Configuration

Figure 8.4 shows the internal bus configuration of this LSI. The internal bus of this LSI consists of the following three types.

- **Internal system bus**  
A bus that connects the CPU, DTC, DMAC, on-chip RAM, internal peripheral bus, and external access bus.
- **Internal peripheral bus**  
A bus that accesses registers in the bus controller, interrupt controller, and DMAC, and registers of peripheral modules such as SCI and timer.
- **External access cycle**  
A bus that accesses external devices via the external bus interface.



**Figure 8.4 Internal Bus Configuration**



## 8.4 Multi-Clock Function and Number of Access Cycles

The internal functions of this LSI operate synchronously with the system clock ( $I\phi$ ), the peripheral module clock ( $P\phi$ ), or the external bus clock ( $B\phi$ ). Table 8.1 shows the synchronization clock and their corresponding functions.

**Table 8.1 Synchronization Clocks and Their Corresponding Functions**

Synchronization Clock	Function Name
$I\phi$	MCU operating mode Interrupt controller Bus controller CPU DTC DMAC Internal memory Clock pulse generator Power down control UBC
$P\phi$	I/O ports TPU PPG TMR WDT SCI A/D D/A IIC2 $\Delta\Sigma$ A/D
$B\phi$	External bus interface

The frequency of each synchronization clock ( $I\phi$ ,  $P\phi$ , and  $B\phi$ ) is specified by the system clock control register (SCKCR) independently. For further details, see section 23, Clock Pulse Generator.

There will be cases when  $P\phi$  and  $B\phi$  are equal to  $I\phi$  and when  $P\phi$  and  $B\phi$  are different from  $I\phi$  according to the SCKCR specifications. In any case, access cycles for internal peripheral functions and external space is performed synchronously with  $P\phi$  and  $B\phi$ , respectively.

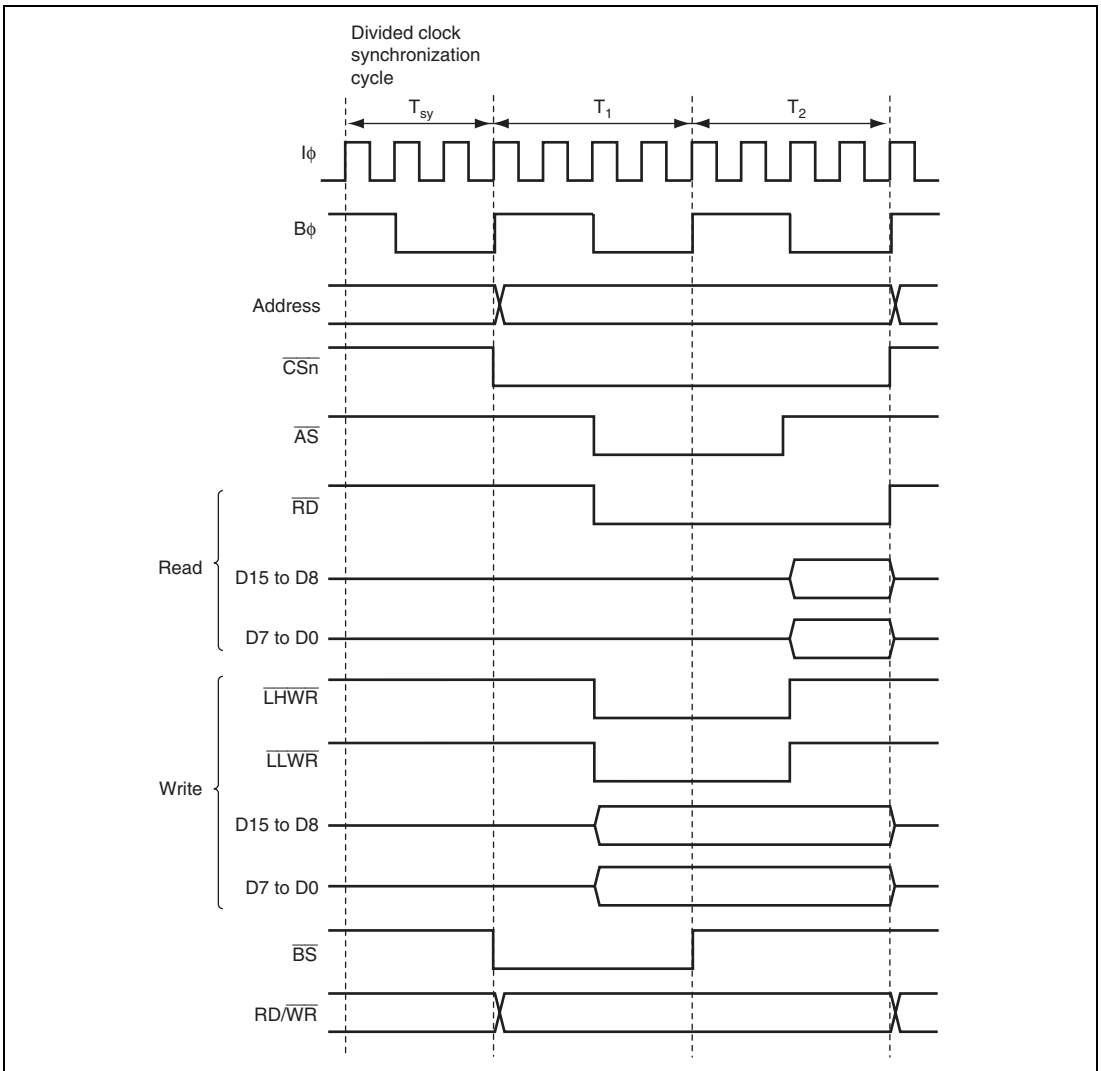
For example, in an external address space access where the frequency rate of  $I\phi$  and  $B\phi$  is  $n : 1$ , the operation is performed in synchronization with  $B\phi$ . In this case, external 2-state access space is  $2n$  cycles and external 3-state access space is  $3n$  cycles (no wait cycles is inserted) if the number of access cycles is counted based on  $I\phi$ .

If the frequencies of  $I\phi$ ,  $P\phi$  and  $B\phi$  are different, the start of bus cycle may not synchronize with  $P\phi$  or  $B\phi$  according to the bus cycle initiation timing. In this case, clock synchronization cycle ( $T_{sy}$ ) is inserted at the beginning of each bus cycle.

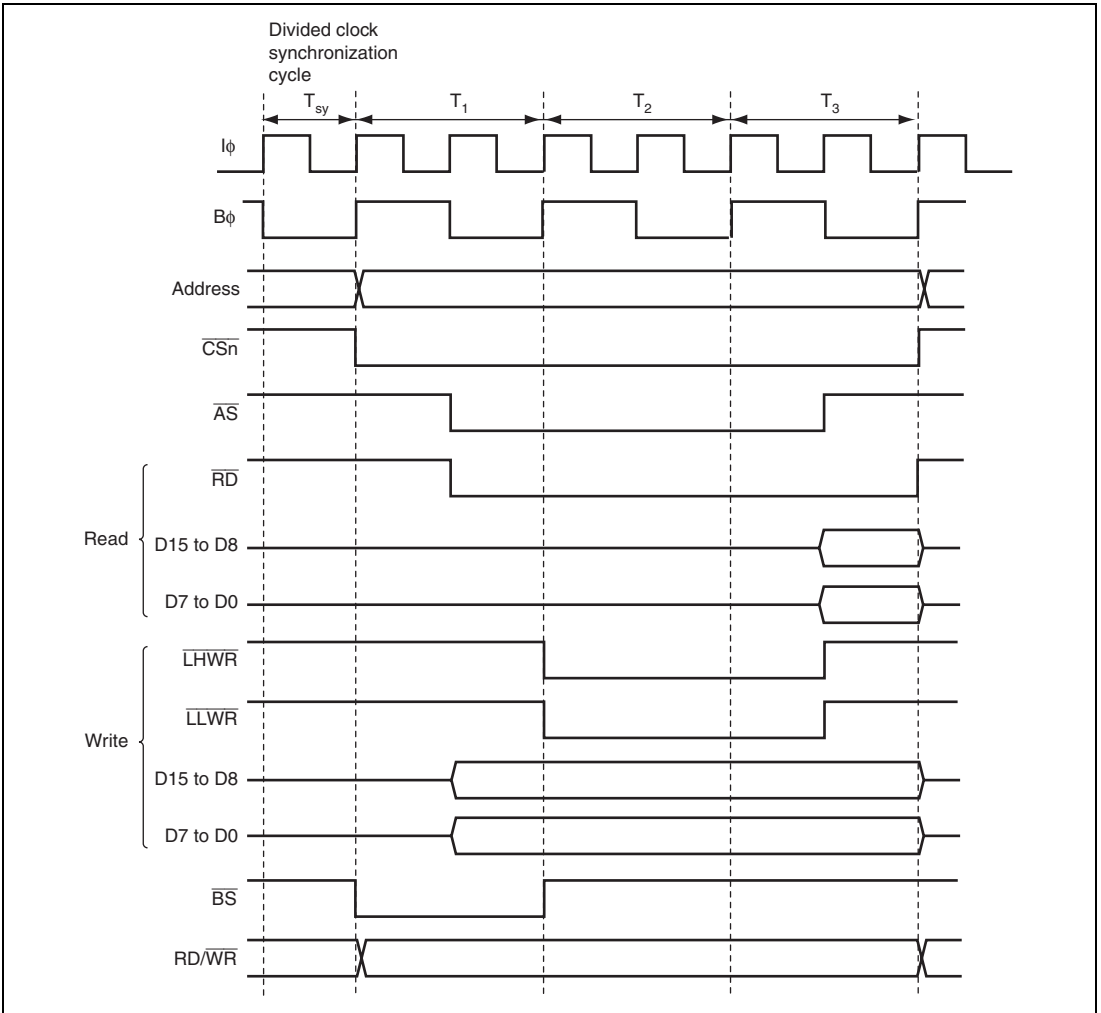
For example, if an external address space access occurs when the frequency rate of  $I\phi$  and  $B\phi$  is  $n : 1$ , 0 to  $n-1$  cycles of  $T_{sy}$  may be inserted. If an internal peripheral module access occurs when the frequency rate of  $I\phi$  and  $P\phi$  is  $m : 1$ , 0 to  $m-1$  cycles of  $T_{sy}$  may be inserted.

Figure 8.5 shows the external 2-state access timing when the frequency rate of  $I\phi$  and  $B\phi$  is  $4 : 1$ .

Figure 8.6 shows the external 3-state access timing when the frequency rate of  $I\phi$  and  $B\phi$  is  $2 : 1$ .



**Figure 8.5 System Clock: External Bus Clock = 4:1, External 2-State Access**



**Figure 8.6 System Clock: External Bus Clock = 2:1, External 3-State Access**

## 8.5 External Bus

### 8.5.1 Input/Output Pins

Table 8.2 shows the pin configuration of the bus controller and table 8.3 shows the pin functions on each interface.

**Table 8.2 Pin Configuration**

Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started
Address strobe/ address hold	$\overline{AS/AH}$	Output	<ul style="list-style-type: none"> <li>Strobe signal indicating that the basic bus, byte control SRAM, or burst ROM space is accessed and address output on address bus is enabled</li> <li>Signal to hold the address during access to the address/data multiplexed I/O interface</li> </ul>
Read strobe	$\overline{RD}$	Output	Strobe signal indicating that the basic bus, byte control SRAM, burst ROM, or address/data multiplexed I/O space is being read
Read/write	$RD/\overline{WR}$	Output	<ul style="list-style-type: none"> <li>Signal indicating the input or output direction</li> <li>Write enable signal of the SRAM during access to the byte control SRAM space</li> </ul>
Low-high write/ lower-upper byte select	$\overline{LHWR/LUB}$	Output	<ul style="list-style-type: none"> <li>Strobe signal indicating that the basic bus, burst ROM, or address/data multiplexed I/O space is written to, and the upper byte (D15 to D8) of data bus is enabled</li> <li>Strobe signal indicating that the byte control SRAM space is accessed, and the upper byte (D15 to D8) of data bus is enabled</li> </ul>
Low-low write/ lower-lower byte select	$\overline{LLWR/LLB}$	Output	<ul style="list-style-type: none"> <li>Strobe signal indicating that the basic bus, burst ROM, or address/data multiplexed I/O space is written to, and the lower byte (D7 to D0) of data bus is enabled</li> <li>Strobe signal indicating that the byte control SRAM space is accessed, and the lower byte (D7 to D0) of data bus is enabled</li> </ul>

<b>Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
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	$\overline{CS2}$	Output	Strobe signal indicating that area 2 is selected
Chip select 3	$\overline{CS3}$	Output	Strobe signal indicating that area 3 is selected
Chip select 4	$\overline{CS4}$	Output	Strobe signal indicating that area 4 is selected
Chip select 5	$\overline{CS5}$	Output	Strobe signal indicating that area 5 is selected
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
Wait	$\overline{WAIT}$	Input	Wait request signal when accessing external address space.
Bus request	$\overline{BREQ}$	Input	Request signal for release of bus to external bus master
Bus request acknowledge	$\overline{BACK}$	Output	Acknowledge signal indicating that bus has been released to external bus master
Bus request output	$\overline{BREQO}$	Output	External bus request signal used when internal bus master accesses external address space in the external-bus released state
Data transfer acknowledge 1 (DMAC_1)	$\overline{DACK1}$	Output	Data transfer acknowledge signal for DMAC_1 single address transfer
Data transfer acknowledge 0 (DMAC_0)	$\overline{DACK0}$	Output	Data transfer acknowledge signal for DMAC_0 single address transfer
External bus clock	$B\phi$	Output	External bus clock

**Table 8.3 Pin Functions in Each Interface**

Pin Name	Initial State			Basic Bus		Byte Control SRAM	Burst ROM		Address/Data Multiplexed I/O		Remarks
	16	8	Single-Chip	16	8	16	16	8	16	8	
	$B\phi$	Output	Output	—	O	O	O	O	O	O	
$\overline{CS0}$	Output	Output	—	O	O	O	O	O	—	—	
$\overline{CS1}$	—	—	—	O	O	O	O	O	—	—	
$\overline{CS2}$	—	—	—	O	O	O	—	—	—	—	
$\overline{CS3}$	—	—	—	O	O	O	—	—	O	O	
$\overline{CS4}$	—	—	—	O	O	O	—	—	O	O	
$\overline{CS5}$	—	—	—	O	O	O	—	—	O	O	
$\overline{CS6}$	—	—	—	O	O	O	—	—	O	O	
$\overline{CS7}$	—	—	—	O	O	O	—	—	O	O	
BS	—	—	—	O	O	O	O	O	O	O	
RD/WR	—	—	—	O	O	O	O	O	O	O	
$\overline{AS}$	Output	Output	—	O	O	O	O	O	—	—	
AH	—	—	—	—	—	—	—	—	O	O	
$\overline{RD}$	Output	Output	—	O	O	O	O	O	O	O	
$\overline{LHWR/LUB}$	Output	Output	—	O	—	O	O	—	O	—	
$\overline{LLWR/LLB}$	Output	Output	—	O	O	O	O	O	O	O	
$\overline{WAIT}$	—	—	—	O	O	O	O	O	O	O	Controlled by WAITE

[Legend]

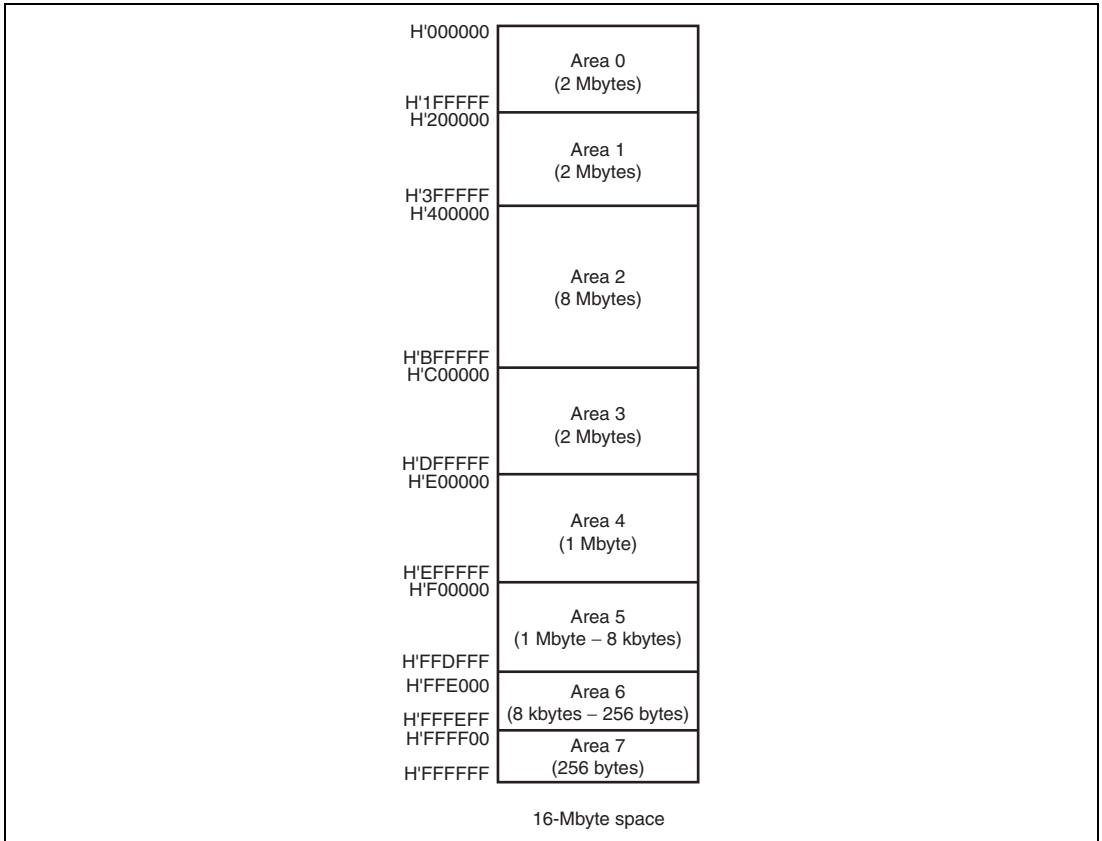
O: Used as a bus control signal

—: Not used as a bus control signal (used as a port input when initialized)

## 8.5.2 Area Division

The bus controller divides the 16-Mbyte address space into eight areas, and performs bus control for the external address space in area units. Chip select signals ( $\overline{CS0}$  to  $\overline{CS7}$ ) can be output for each area.

Figure 8.7 shows an area division of the 16-Mbyte address space. For details on address map, see section 3, MCU Operating Modes.



**Figure 8.7 Address Space Area Division**



### 8.5.3 Chip Select Signals

This LSI can output chip select signals ( $\overline{CS0}$  to  $\overline{CS7}$ ) for areas 0 to 7. The signal outputs low when the corresponding external address space area is accessed. Figure 8.8 shows an example of  $\overline{CSn}$  ( $n = 0$  to 7) signal output timing.

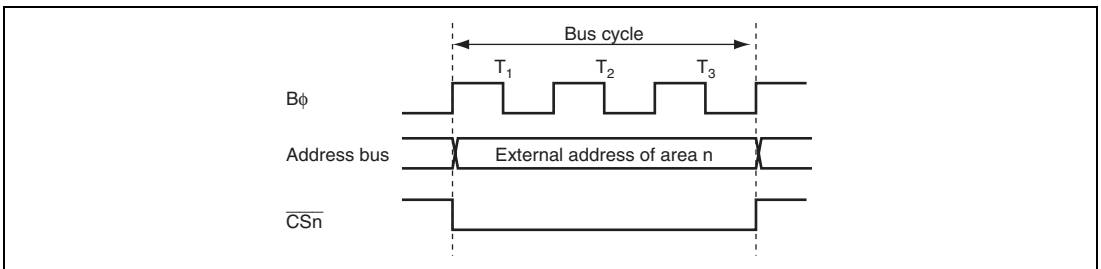
Enabling or disabling of  $\overline{CSn}$  signal output is set by the port function control register (PFCR). For details, see section 11.3, Port Function Controller.

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

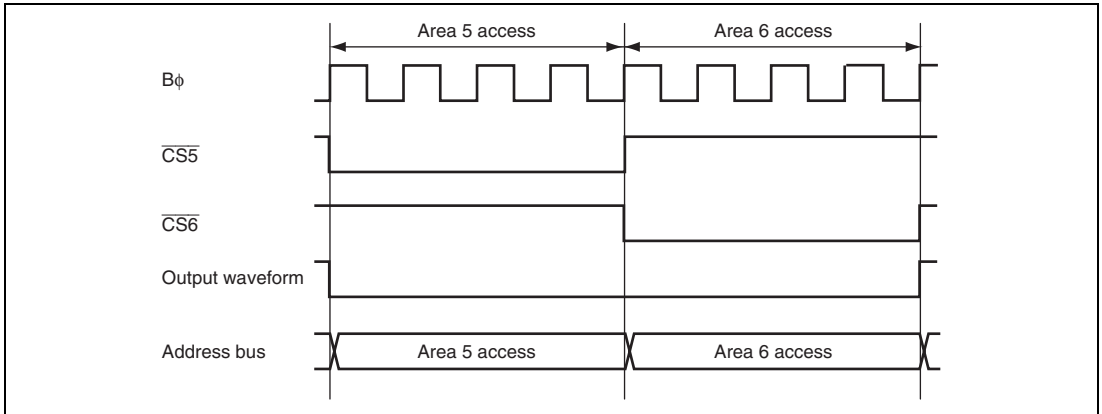
In on-chip ROM enabled extended mode, pins  $\overline{CS0}$  to  $\overline{CS7}$  are all placed in the input state after a reset and so the corresponding PFCR bits should be set to 1 when outputting signals  $\overline{CS0}$  to  $\overline{CS7}$ .

The PFCR can specify multiple  $\overline{CS}$  outputs for a pin. If multiple  $\overline{CSn}$  outputs are specified for a single pin by the PFCR,  $\overline{CS}$  to be output are generated by mixing all the  $\overline{CS}$  signals. In this case, the settings for the external bus interface areas in which the  $\overline{CSn}$  signals are output to a single pin should be the same.

Figure 8.9 shows the signal output timing when the  $\overline{CS}$  signals to be output to areas 5 and 6 are output to the same pin.



**Figure 8.8**  $\overline{CSn}$  Signal Output Timing ( $n = 0$  to 7)



**Figure 8.9 Timing When  $\overline{CS}$  Signal is Output to the Same Pin**

### 8.5.4 External Bus Interface

The type of the external bus interfaces, bus width, endian format, number of access cycles, and strobe assert/negate timings can be set for each area in the external address space. The bus width and the number of access cycles for both on-chip memory and internal I/O registers are fixed, and are not affected by the external bus settings.

#### (1) Type of External Bus Interface

Four types of external bus interfaces are provided and can be selected in area units. Table 8.4 shows each interface name, description, area name to be set for each interface. Table 8.5 shows the areas that can be specified for each interface. The initial state of each area is a basic bus interface.

**Table 8.4 Interface Names and Area Names**

Interface	Description	Area Name
Basic interface	Directly connected to ROM and RAM	Basic bus space
Byte control SRAM interface	Directly connected to byte SRAM with byte control pin	Byte control SRAM space
Burst ROM interface	Directly connected to the ROM that allows page access	Burst ROM space
Address/data multiplexed I/O interface	Directly connected to the peripheral LSI that requires address and data multiplexing	Address/data multiplexed I/O space

**Table 8.5 Areas Specifiable for Each Interface**

Interface	Related Registers	Areas							
		0	1	2	3	4	5	6	7
Basic interface	SRAMCR	0	0	0	0	0	0	0	0
Byte control SRAM interface		0	0	0	0	0	0	0	0
Burst ROM interface	BROMCR	0	0	—	—	—	—	—	—
Address/data multiplexed I/O interface	MPXCR	—	—	—	0	0	0	0	0

**(2) 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. In addition, the bus width of address/data multiplexed I/O space is 8 bits or 16 bits, and the bus width for the byte control SRAM space is 16 bits.

The initial state of the bus width is specified by the operating mode.

If all areas are designated as 8-bit access space, 8-bit bus mode is set; if any area is designated as 16-bit access space, 16-bit bus mode is set.

**(3) Endian Format**

Though the endian format of this LSI is big endian, data can be converted into little endian format when reading or writing to the external address space.

Areas 7 to 2 can be specified as either big endian or little endian format by the LE7 to LE2 bits in ENDIANCR.

The initial state of each area is the big endian format.

Note that the data format for the areas used as a program area or a stack area should be big endian.

**(4) Number of Access Cycles****(a) Basic Bus Interface**

The number of access cycles in the basic bus interface can be specified as two or three cycles by the ASTCR. An area specified as 2-state access is specified as 2-state access space; an area specified as 3-state access is specified as 3-state access space.

For the 2-state access space, a wait cycle insertion is disabled. For the 3-state access space, a program wait (0 to 7 cycles) specified by WTCRA and WTCRB or an external wait by  $\overline{\text{WAIT}}$  can be inserted.

$$\begin{aligned} &\text{Number of access cycles in the basic bus interface} \\ &= \text{number of basic cycles (2, 3) + number of program wait cycles (0 to 7)} \\ &\quad + \text{number of } \overline{\text{CS}} \text{ extension cycles (0, 1, 2)} \\ &\quad [+ \text{number of external wait cycles by the } \overline{\text{WAIT}} \text{ pin}] \end{aligned}$$

Assertion period of the chip select signal can be extended by CSACR.

**(b) Byte Control SRAM Interface**

The number of access cycles in the byte control SRAM interface is the same as that in the basic bus interface.

$$\begin{aligned} &\text{Number of access cycles in byte control SRAM interface} \\ &= \text{number of basic cycles (2, 3) + number of program wait cycles (0 to 7)} \\ &\quad + \text{number of } \overline{\text{CS}} \text{ extension cycles (0, 1, 2)} \\ &\quad [+ \text{number of external wait cycles by the } \overline{\text{WAIT}} \text{ pin}] \end{aligned}$$

**(c) Burst ROM Interface**

The number of access cycles at full access in the burst ROM interface is the same as that in the basic bus interface. The number of access cycles in the burst access can be specified as one to eight cycles by the BSTS bit in BROMCR.

$$\begin{aligned} &\text{Number of access cycles in the burst ROM interface} \\ &= \text{number of basic cycles (2, 3) + number of program wait cycles (0 to 7)} \\ &\quad + \text{number of } \overline{\text{CS}} \text{ extension cycles (0, 1)} \\ &\quad [+ \text{number of external wait cycles by the } \overline{\text{WAIT}} \text{ pin}] \\ &\quad + \text{number of burst access cycles (1 to 8)} \times \text{number of burst accesses (0 to 63)} \end{aligned}$$

**(d) Address/data multiplexed I/O interface**

The number of access cycles in data cycle of the address/data multiplexed I/O interface is the same as that in the basic bus interface. The number of access cycles in address cycle can be specified as two or three cycles by the ADDEX bit in MPXCR.

$$\begin{aligned}
 &\text{Number of access cycles in the address/data multiplexed I/O interface} \\
 &= \text{number of address output cycles (2, 3) + number of data output cycles (2, 3)} \\
 &\quad + \text{number of program wait cycles (0 to 7)} \\
 &\quad + \text{number of } \overline{\text{CS}} \text{ extension cycles (0, 1, 2)} \\
 &\quad [+ \text{number of external wait cycles by the } \overline{\text{WAIT}} \text{ pin}]
 \end{aligned}$$

Table 8.6 lists the number of access cycles for each interface.

**Table 8.6 Number of Access Cycles**

Basic bus interface	=	Th	+T1	+T2				+Tt	
	-	[0,1]	[1]	[1]				[0,1]	[2 to 4]
Byte control SRAM interface	=	Th	+T1	+T2	+Tpw	+Ttw	+T3	+Tt	
	-	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]	[3 to 12 + n]
Burst ROM interface	=	Th	+T1	+T2				+Tb	
	-	[0,1]	[1]	[1]				[(1 to 8) × m]	[(2 to 3) + (1 to 8) × m]
Address/data multiplexed I/O interface	=	Th	+T1	+T2	+Tpw	+Ttw	+T3	+Tt	
	-	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]	[3 to 12 + n]
interface	=	Tma	+Th	+T1	+T2			+Tt	
	-	[2,3]	[0,1]	[1]	[1]			[0,1]	[4 to 7]
		Tma	+Th	+T1	+T2	+Tpw	+Ttw	+T3	+Tt
		[2,3]	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]
									[5 to 15 + n]

[Legend]

Numbers: Number of access cycles

n: Pin wait (0 to ∞)

m: Number of burst accesses (0 to 63)

**(5) Strobe Assert/Negate Timings**

The assert and negate timings of the strobe signals can be modified as well as number of access cycles.

- Read strobe ( $\overline{\text{RD}}$ ) in the basic bus interface
- Chip select assertion period extension cycles in the basic bus interface
- Data transfer acknowledge ( $\overline{\text{DACK3}}$  to  $\overline{\text{DACK0}}$ ) output for DMAC single address transfers

### 8.5.5 Area and External Bus Interface

#### (1) Area 0

Area 0 includes on-chip ROM. All of area 0 is used as external address space in on-chip ROM disabled extended mode, and the space excluding on-chip ROM is external address space in on-chip ROM enabled extended mode.

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

Either of the basic bus interface, byte control SRAM interface, or burst ROM interface can be selected for area 0 by bit BSRM0 in BROMCR and bit BCSEL0 in SRAMCR. Table 8.7 shows the external interface of area 0.

**Table 8.7 Area 0 External Interface**

Interface	Register Setting	
	BSRM0 of BROMCR	BCSEL0 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Burst ROM interface	1	0
Setting prohibited	1	1

**(2) Area 1**

n externally extended mode, all of area 1 is external address space. In on-chip ROM enabled extended mode, the space excluding on-chip ROM is external address space.

When area 1 external address space is accessed, the  $\overline{CS1}$  signal can be output.

Either of the basic bus interface, byte control SRAM, or burst ROM interface can be selected for area 1 by bit BSRM1 in BROMCR and bit BCSEL1 in SRAMCR. Table 8.8 shows the external interface of area 1.

**Table 8.8 Area 1 External Interface**

Interface	Register Setting	
	BSRM1 of BROMCR	BCSEL1 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Burst ROM interface	1	0
Setting prohibited	1	1

**(3) Area 2**

In externally extended mode, all of area 2 is external address space.

When area 2 external address space is accessed, the  $\overline{CS2}$  signal can be output.

Either the basic bus interface or byte control SRAM interface can be selected for area 2 by bit BCSEL2 in SRAMCR. Table 8.9 shows the external interface of area 2.

**Table 8.9 Area 2 External Interface**

Interface	Register Setting
	BCSEL2 of SRAMCR
Basic bus interface	0
Byte control SRAM interface	1

**(4) Area 3**

In externally extended mode, all of area 3 is external address space.

When area 3 external address space is accessed, the  $\overline{CS3}$  signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 3 by bit MPXE3 in MPXCR and bit BCSEL3 in SRAMCR.

Table 8.10 shows the external interface of area 3.

**Table 8.10 Area 3 External Interface**

Interface	Register Setting	
	MPXE3 of MPXCR	BCSEL3 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

**(5) Area 4**

In externally extended mode, all of area 4 is external address space.

When area 4 external address space is accessed, the  $\overline{CS4}$  signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 4 by bit MPXE4 in MPXCR and bit BCSEL4 in SRAMCR.

Table 8.11 shows the external interface of area 4.

**Table 8.11 Area 4 External Interface**

Interface	Register Setting	
	MPXE4 of MPXCR	BCSEL4 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1



**(6) Area 5**

Area 5 includes the on-chip RAM and access prohibited spaces. In external extended mode, area 5, other than the on-chip RAM and access prohibited spaces, is external address space. Note that the on-chip RAM is enabled when the RAME bit in SYSCR are set to 1. If the RAME bit in SYSCR is cleared to 0, the on-chip RAM is disabled and the corresponding addresses are an external address space. For details, see section 3, MCU Operating Modes.

When area 5 external address space is accessed, the  $\overline{CS5}$  signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 5 by the MPXE5 bit in MPXCR and the BCSEL5 bit in SRAMCR. Table 8.12 shows the external interface of area 5.

**Table 8.12 Area 5 External Interface**

Interface	Register Setting	
	MPXE5 of MPXCR	BCSEL5 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

**(7) Area 6**

Area 6 includes internal I/O registers. In external extended mode, area 6 other than on-chip I/O register area is external address space.

When area 6 external address space is accessed, the  $\overline{CS6}$  signal can be output.

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 6 by the MPXE6 bit in MPXCR and the BCSEL6 bit in SRAMCR. Table 8.13 shows the external interface of area 6.

**Table 8.13 Area 6 External Interface**

Interface	Register Setting	
	MPXE6 of MPXCR	BCSEL6 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

**(8) Area 7**

Area 7 includes internal I/O registers. In external extended mode, area 7 other than internal I/O register area is external address space.

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

Either of the basic bus interface, byte control SRAM interface, or address/data multiplexed I/O interface can be selected for area 7 by the MPXE7 bit in MPXCR and the BCSEL7 bit in SRAMCR. Table 8.14 shows the external interface of area 7.

**Table 8.14 Area 7 External Interface**

Interface	Register Setting	
	MPXE7 of MPXCR	BCSEL7 of SRAMCR
Basic bus interface	0	0
Byte control SRAM interface	0	1
Address/data multiplexed I/O interface	1	0
Setting prohibited	1	1

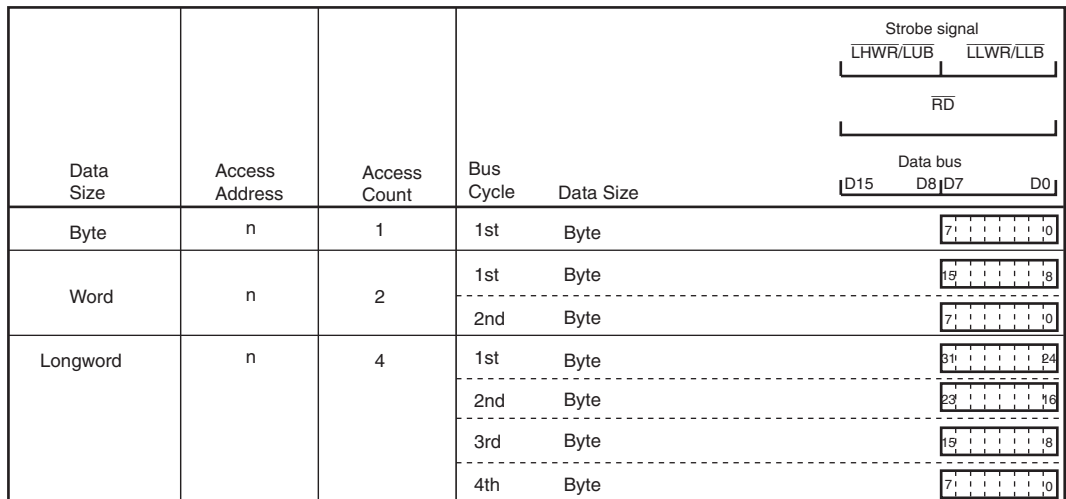
## 8.5.6 Endian and Data Alignment

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and controls whether the upper byte data bus (D15 to D8) or lower data bus (D7 to D0) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space), the data size, and endian format when accessing external address space.

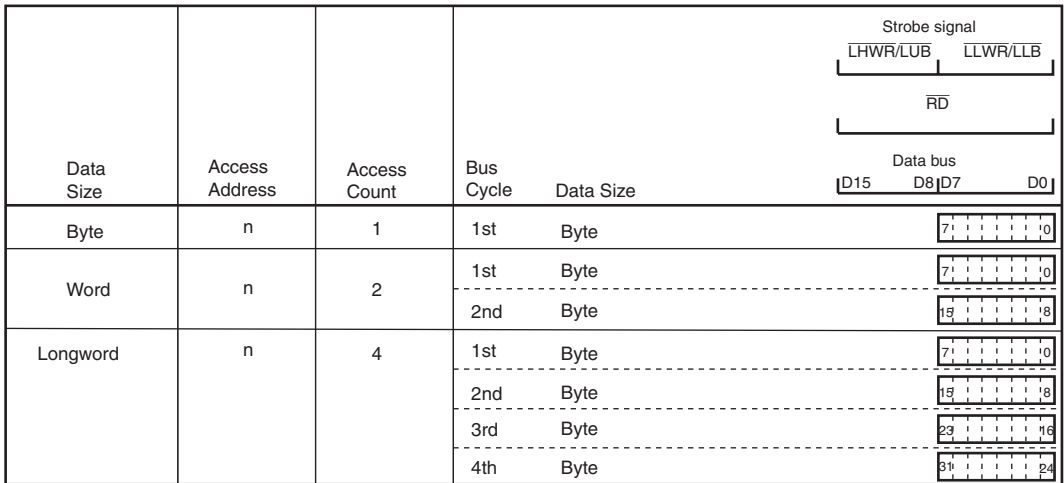
### (1) 8-Bit Access Space

With the 8-bit access space, the lower byte data bus (D7 to D0) is always used for access. The amount of data that can be accessed at one time is one byte: a word access is performed as two byte accesses, and a longword access, as four byte accesses.

Figures 8.10 and 8.11 illustrate data alignment control for the 8-bit access space. Figure 8.10 shows the data alignment when the data endian format is specified as big endian. Figure 8.11 shows the data alignment when the data endian format is specified as little endian.



**Figure 8.10 Access Sizes and Data Alignment Control for 8-Bit Access Space (Big Endian)**



**Figure 8.11 Access Sizes and Data Alignment Control for 8-Bit Access Space (Little Endian)**

## (2) 16-Bit Access Space

With the 16-bit access space, the upper byte data bus (D15 to D8) and lower byte data bus (D7 to D0) are used for accesses. The amount of data that can be accessed at one time is one byte or one word.

Figures 8.12 and 8.13 illustrate data alignment control for the 16-bit access space. Figure 8.12 shows the data alignment when the data endian format is specified as big endian. Figure 8.13 shows the data alignment when the data endian format is specified as little endian.

In big endian, byte access for an even address is performed by using the upper byte data bus and byte access for an odd address is performed by using the lower byte data bus.

In little endian, byte access for an even address is performed by using the lower byte data bus, and byte access for an odd address is performed by using the third byte data bus.

Access Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe signal		
					LHWR/LUB	LLWR/LLB	
					RD		
					Data bus		
					D15	D8 D7	D0
Byte	Even (2n)	1	1st	Byte	7 1 1 1 1 1 1 1 0		
	Odd (2n+1)	1	1st	Byte	7 1 1 1 1 1 1 1 0		
Word	Even (2n)	1	1st	Word	15 1 1 1 1 1 1 1 8 7 1 1 1 1 1 1 1 0		
	Odd (2n+1)	2	1st	Byte	15 1 1 1 1 1 1 1 8		
			2nd	Byte	7 1 1 1 1 1 1 1 0		
Longword	Even (2n)	2	1st	Word	31 1 1 1 1 1 1 1 24 23 1 1 1 1 1 1 1 16		
			2nd	Word	15 1 1 1 1 1 1 1 8 7 1 1 1 1 1 1 1 0		
	Odd (2n+1)	3	1st	Byte	31 1 1 1 1 1 1 1 24		
			2nd	Word	23 1 1 1 1 1 1 1 16 15 1 1 1 1 1 1 8		
			3rd	Byte	7 1 1 1 1 1 1 1 0		

**Figure 8.12 Access Sizes and Data Alignment Control for 16-Bit Access Space (Big Endian)**

Access Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe signal		
					LHWR/LUB	LLWR/LLB	
					RD		
					Data bus		
					D15	D8 D7	D0
Byte	Even (2n)	1	1st	Byte	7 1 1 1 1 1 1 1 0		
	Odd (2n+1)	1	1st	Byte	7 1 1 1 1 1 1 1 0		
Word	Even (2n)	1	1st	Word	15 1 1 1 1 1 1 1 8 7 1 1 1 1 1 1 1 0		
	Odd (2n+1)	2	1st	Byte	7 1 1 1 1 1 1 1 0		
			2nd	Byte	15 1 1 1 1 1 1 1 8		
Longword	Even (2n)	2	1st	Word	15 1 1 1 1 1 1 1 8 7 1 1 1 1 1 1 1 0		
			2nd	Word	31 1 1 1 1 1 1 1 24 23 1 1 1 1 1 1 1 16		
	Odd (2n+1)	3	1st	Byte	7 1 1 1 1 1 1 1 0		
			2nd	Word	23 1 1 1 1 1 1 1 16 15 1 1 1 1 1 1 8		
			3rd	Byte	31 1 1 1 1 1 1 1 24		

**Figure 8.13 Access Sizes and Data Alignment Control for 16-Bit Access Space (Little Endian)**

## 8.6 Basic Bus Interface

The basic bus interface can be connected directly to the ROM and SRAM. The bus specifications can be specified by the ABWCR, ASTCR, WTCRA, WTCRB, RDNCR, CSACR, and ENDIANCR.

### 8.6.1 Data Bus

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and controls whether the upper byte data bus (D15 to D8) or lower byte data bus (D7 to D0) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space), the data size, and endian format when accessing external address space. For details, see section 8.5.6, Endian and Data Alignment.

### 8.6.2 I/O Pins Used for Basic Bus Interface

Table 8.15 shows the pins used for basic bus interface.

**Table 8.15 I/O Pins for Basic Bus Interface**

Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started
Address strobe	$\overline{AS}^*$	Output	Strobe signal indicating that an address output on the address bus is valid during access
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Read/write	$\overline{RD}/\overline{WR}$	Output	Signal indicating the data bus input or output direction
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid during write access
Low-low write	$\overline{LLWR}$	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid during write access
Chip select 0 to 7	$\overline{CS0}$ to $\overline{CS7}$	Output	Strobe signal indicating that the area is selected
Wait	$\overline{WAIT}$	Input	Wait request signal used when an external address space is accessed

Note: \* When the address/data multiplexed I/O is selected, this pin only functions as the AH output and does not function as the AS output.

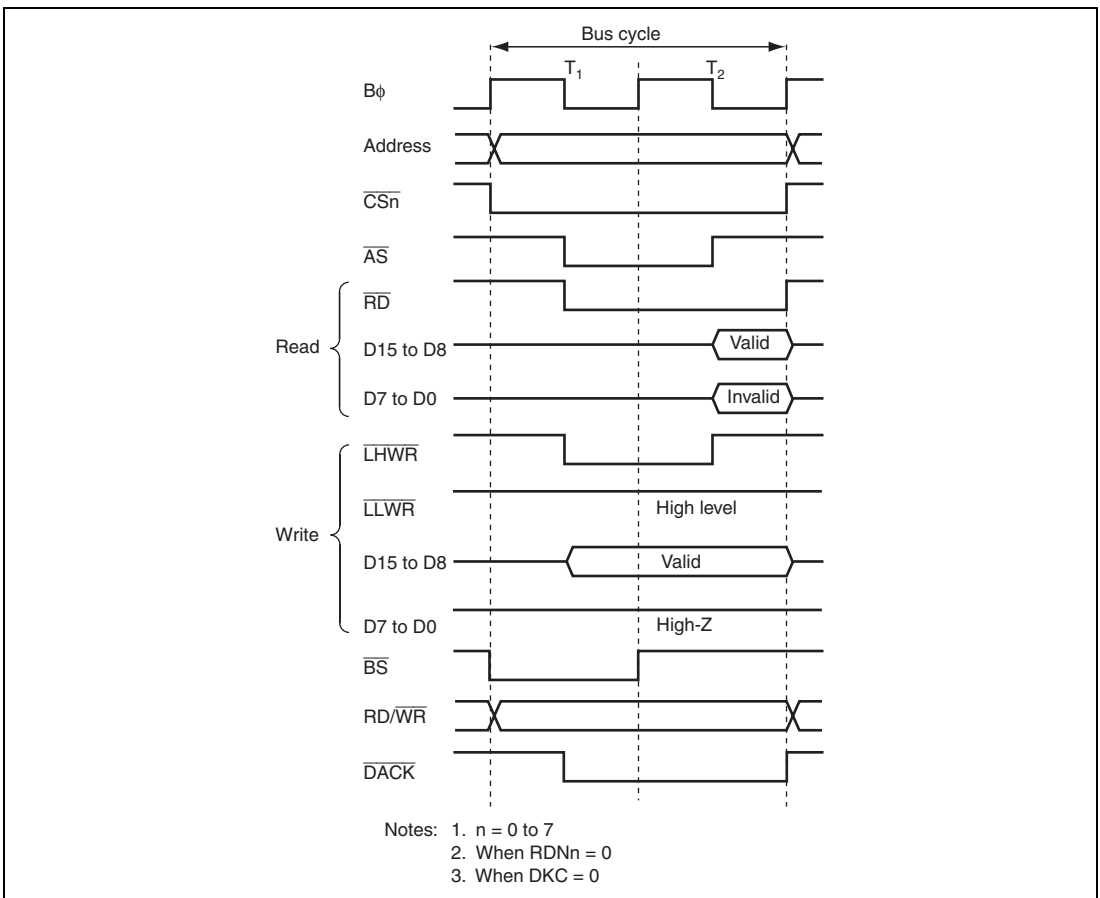
### 8.6.3 Basic Timing

This section describes the basic timing when the data is specified as big endian.

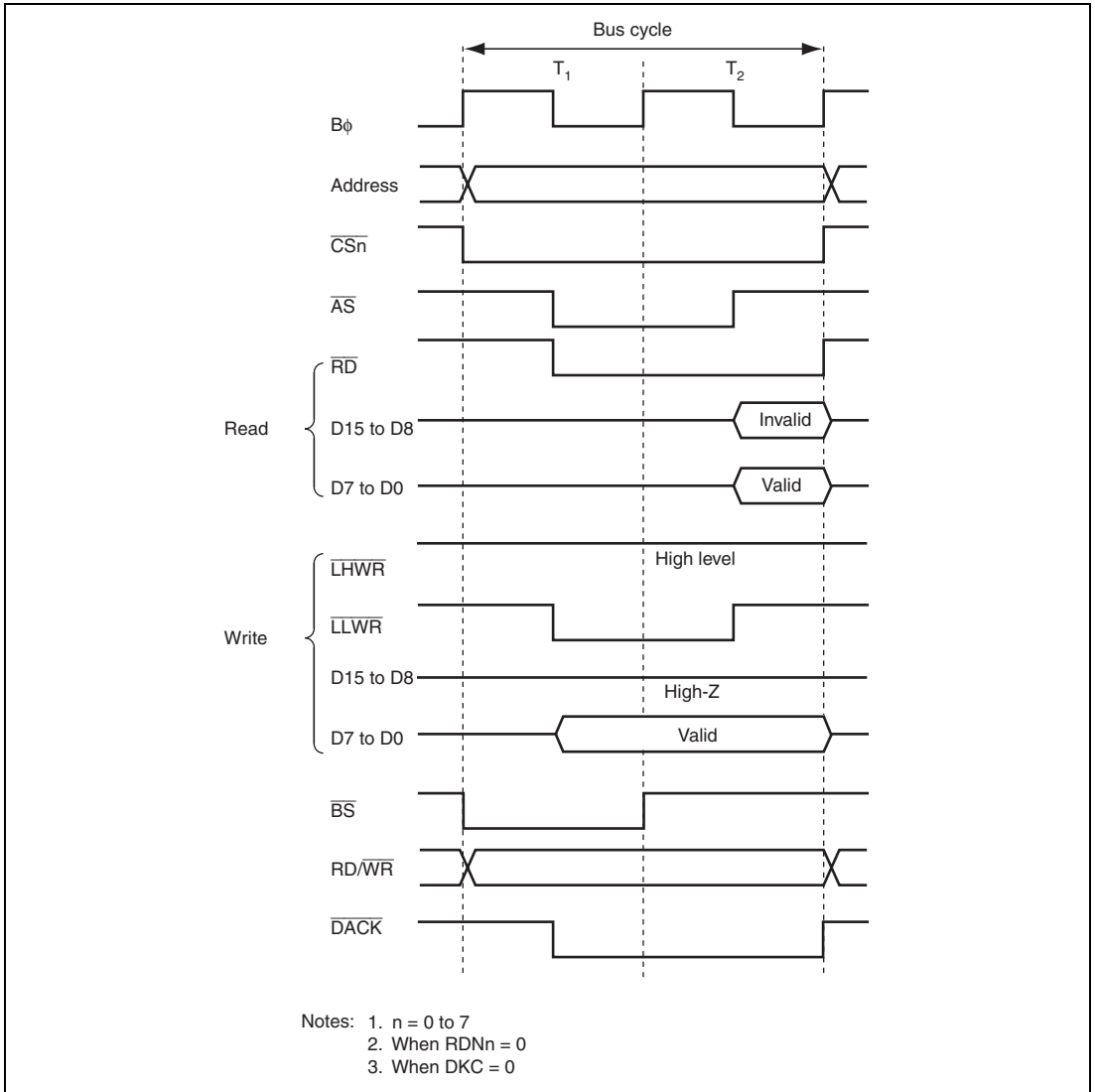
#### (1) 16-Bit 2-State Access Space

Figures 8.14 to 8.16 show the bus timing of 16-bit 2-state access space.

When accessing 16-bit access space, the upper byte data bus (D15 to D8) is used for even addresses access, and the lower byte data bus (D7 to D0) is used for odd addresses. No wait cycles can be inserted.

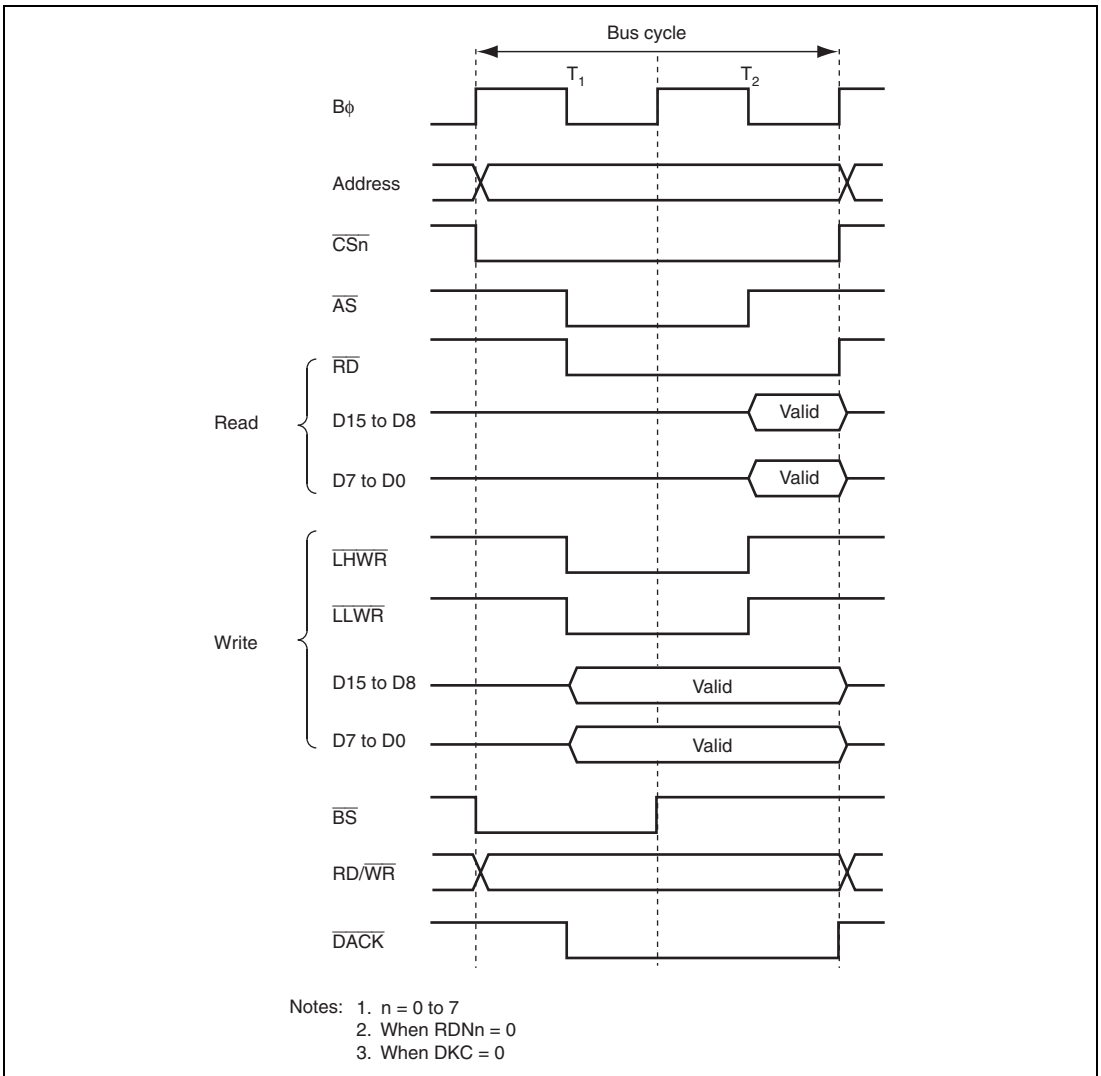


**Figure 8.14 16-Bit 2-State Access Space Bus Timing (Byte Access for Even Address)**



**Figure 8.15 16-Bit 2-State Access Space Bus Timing (Byte Access for Odd Address)**



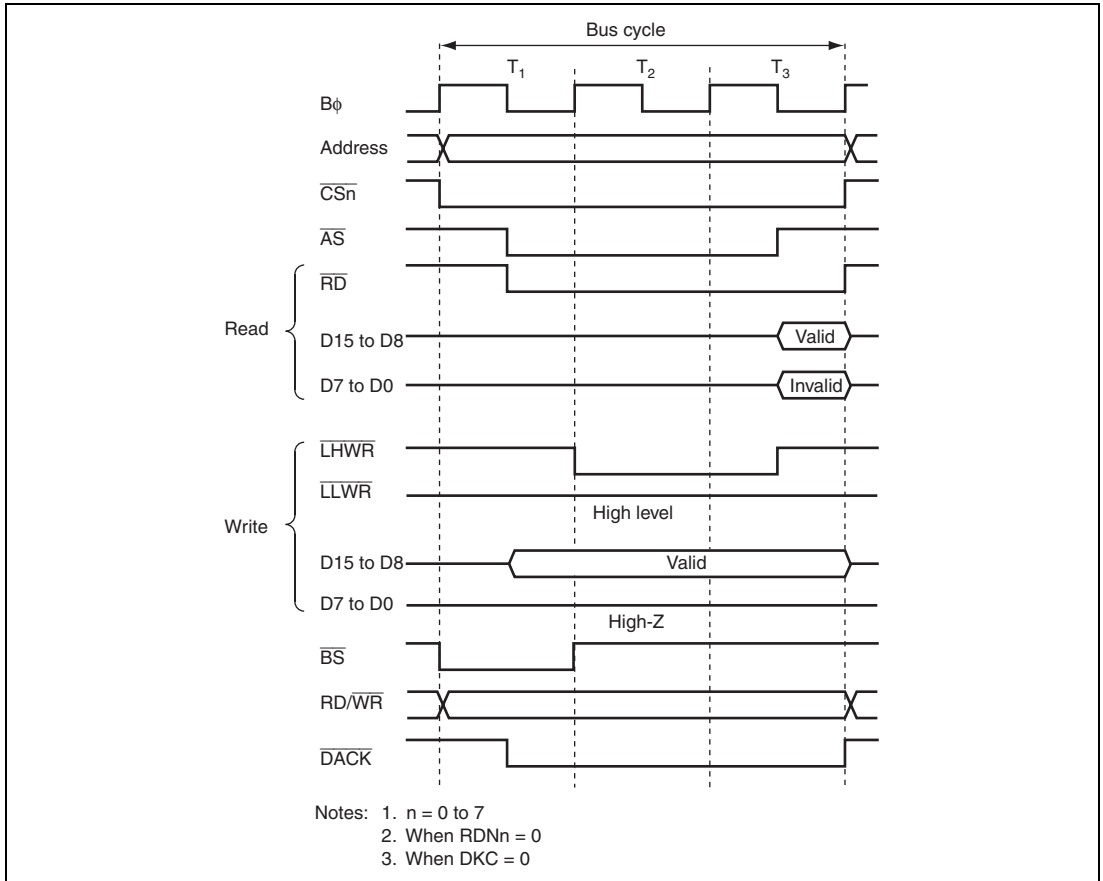


**Figure 8.16 16-Bit 2-State Access Space Bus Timing (Word Access for Even Address)**

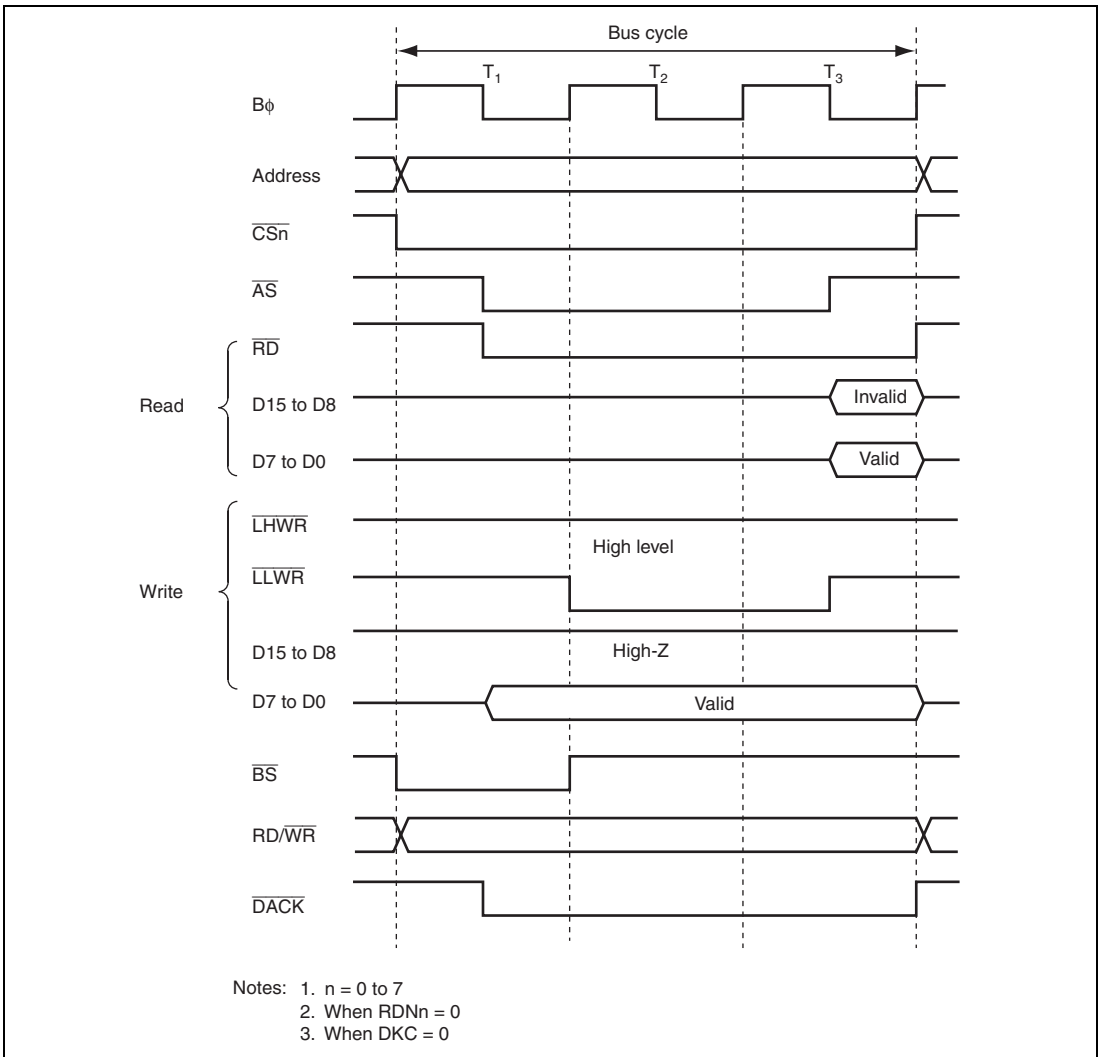
## (2) 16-Bit 3-State Access Space

Figures 8.17 to 8.19 show the bus timing of 16-bit 3-state access space.

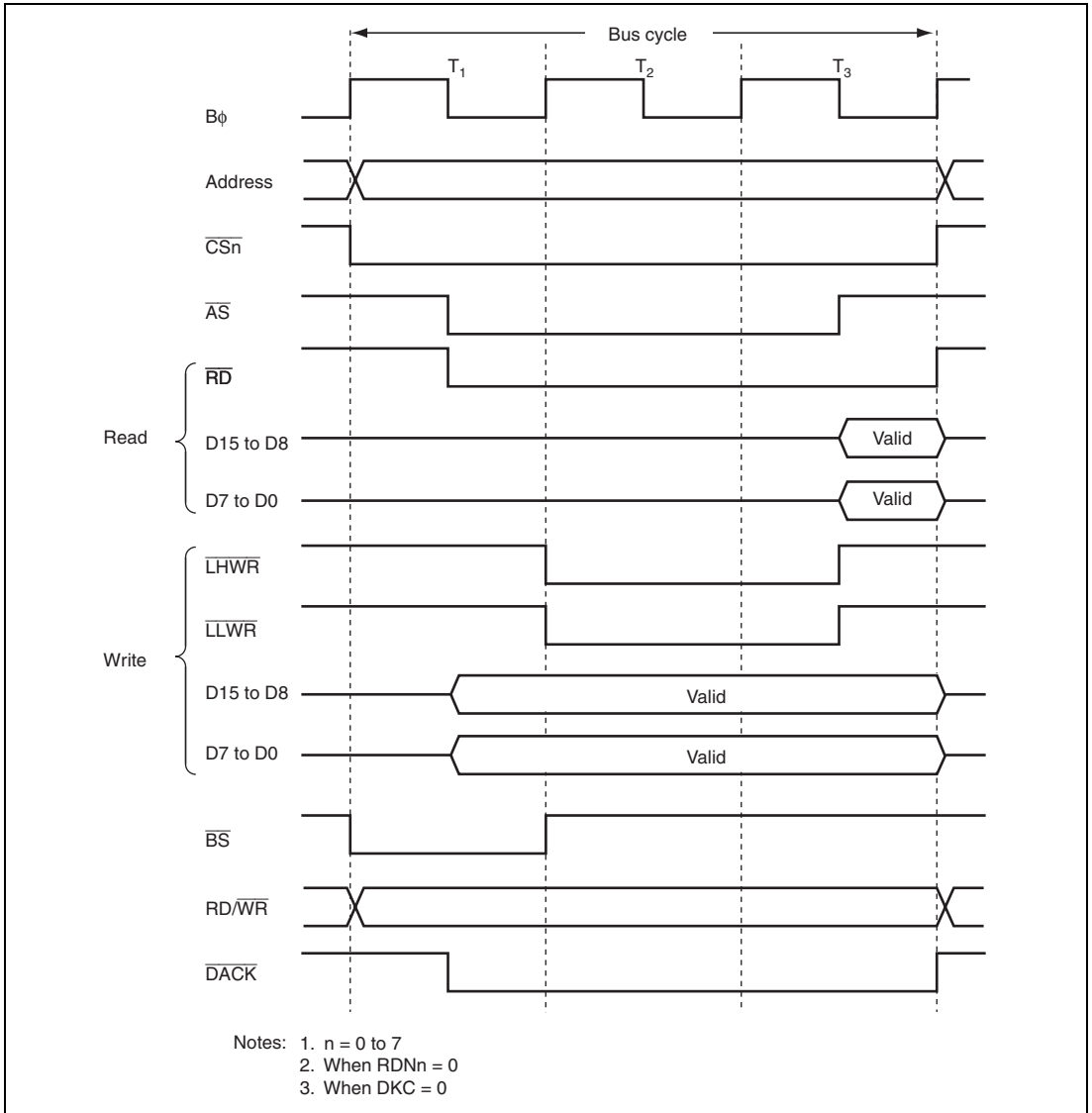
When accessing 16-bit access space, the upper byte data bus (D15 to D8) is used for even addresses, and the lower byte data bus (D7 to D0) is used for odd addresses. Wait cycles can be inserted.



**Figure 8.17 16-Bit 3-State Access Space Bus Timing (Byte Access for Even Address)**



**Figure 8.18 16-Bit 3-State Access Space Bus Timing (Word Access for Odd Address)**



**Figure 8.19 16-Bit 3-State Access Space Bus Timing (Word Access for Even Address)**

## 8.6.4 Wait Control

This LSI can extend the bus cycle by inserting wait cycles ( $T_w$ ) when the external address space is accessed. There are two ways of inserting wait cycles: program wait ( $T_{pw}$ ) insertion and pin wait ( $T_{tw}$ ) insertion using the  $\overline{\text{WAIT}}$  pin.

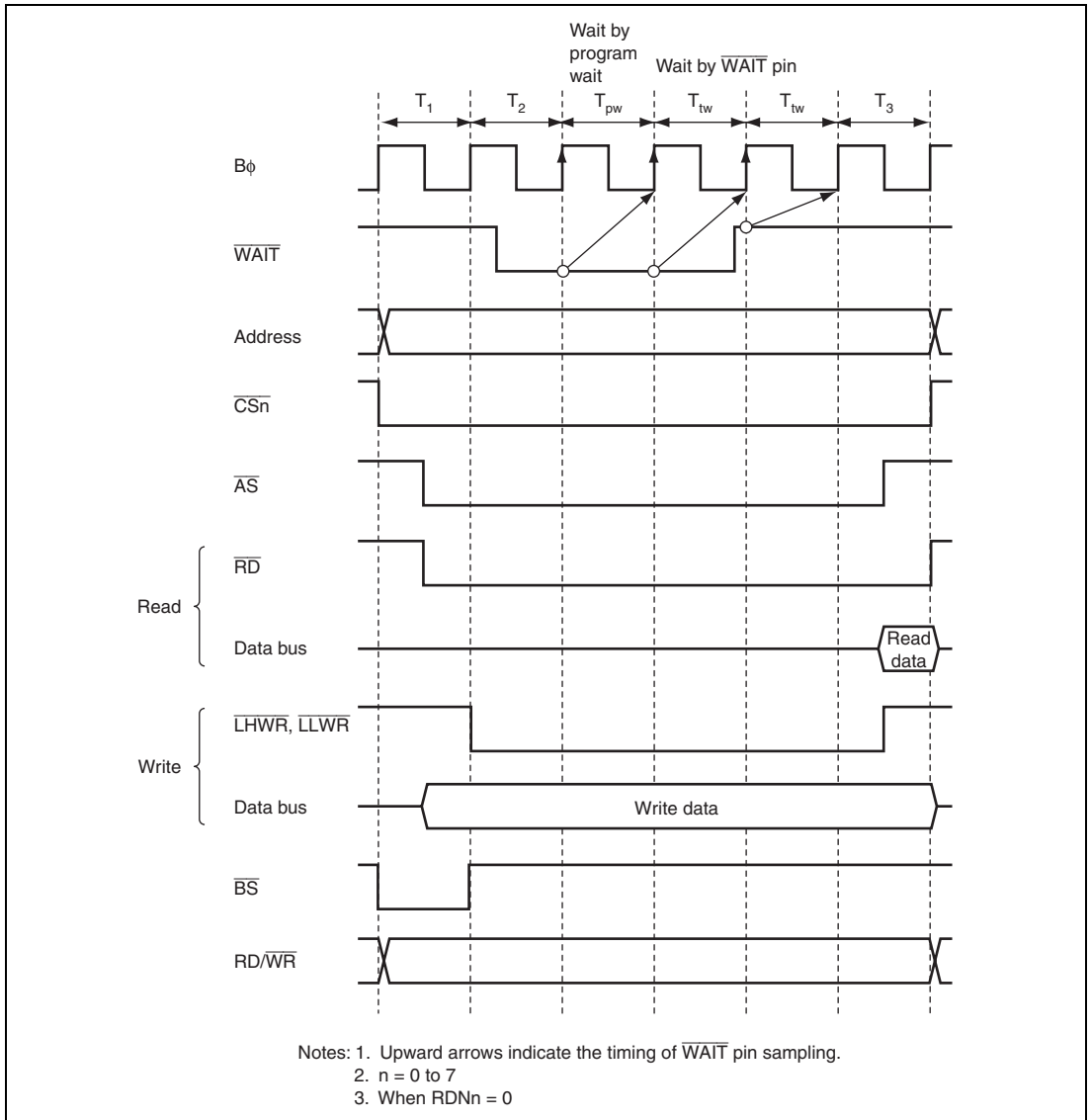
### (1) Program Wait Insertion

From 0 to 7 wait cycles can be inserted automatically between the  $T_2$  state and  $T_3$  state for 3-state access space, according to the settings in WTCRA and WTCRB.

### (2) Pin Wait Insertion

For 3-state access space, when the WAITE bit in BCR1 is set to 1 and the corresponding ICR bit is set to 1, wait input by means of the  $\overline{\text{WAIT}}$  pin is enabled. When the external address space is accessed in this state, a program wait ( $T_{pw}$ ) is first inserted according to the WTCRA and WTCRB settings. If the  $\overline{\text{WAIT}}$  pin is low at the falling edge of  $B\phi$  in the last  $T_2$  or  $T_{pw}$  cycle, another  $T_{tw}$  cycle is inserted until the  $\overline{\text{WAIT}}$  pin is brought high. The pin wait insertion is effective when the  $T_w$  cycles are inserted to seven cycles or more, or when the number of  $T_w$  cycles to be inserted is changed according to the external devices. The WAITE bit is common to all areas. For details on ICR, see section 11, I/O Ports.

Figure 8.20 shows an example of wait cycle insertion timing. After a reset, the 3-state access is specified, the program wait is inserted for seven cycles, and the  $\overline{\text{WAIT}}$  input is disabled.



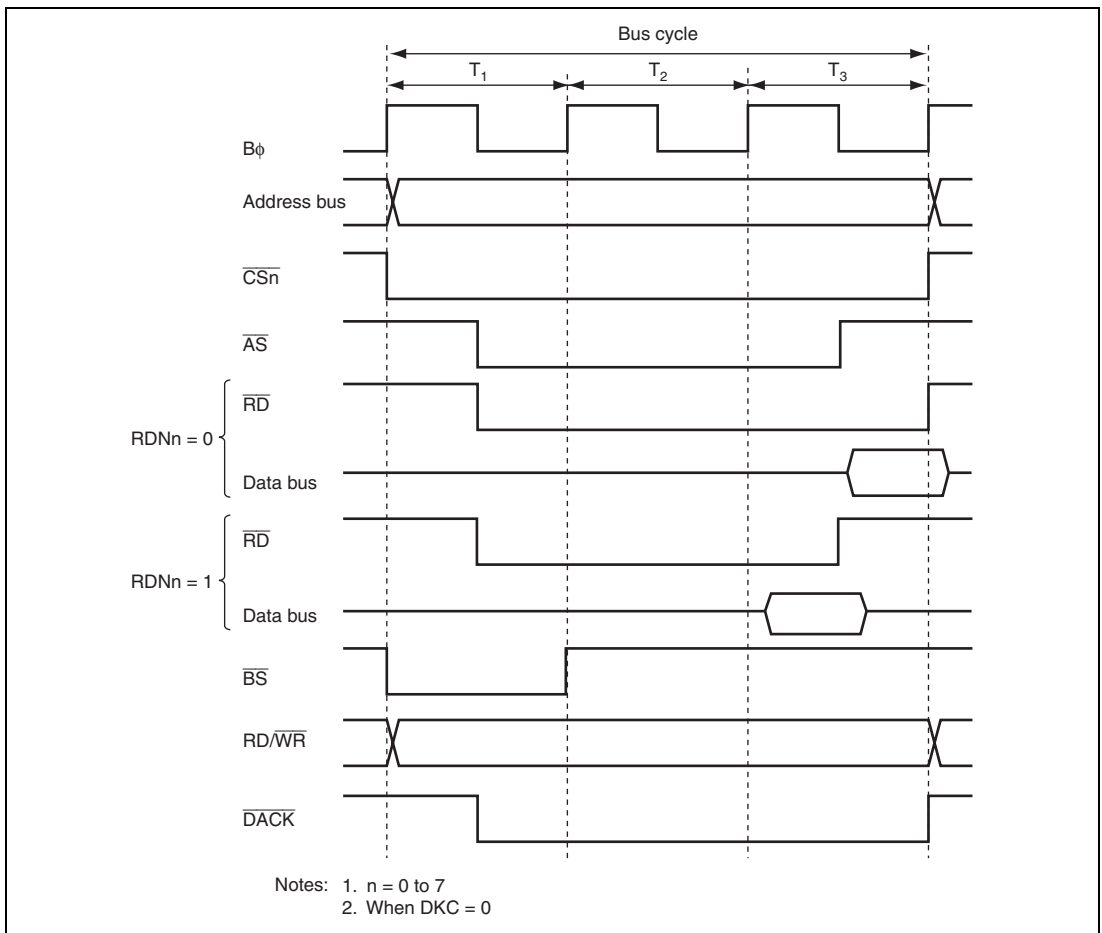
**Figure 8.20 Example of Wait Cycle Insertion Timing**

### 8.6.5 Read Strobe ( $\overline{RD}$ ) Timing

The read strobe timing can be modified in area units by setting bits RDN7 to RDN0 in RDNCR to 1.

Note that the  $\overline{RD}$  timing with respect to the  $\overline{DACK}$  rising edge will change if the read strobe timing is modified by setting RDNn to 1 when the DMAC is used in the single address mode.

Figure 8.21 shows an example of timing when the read strobe timing is changed in the basic bus 3-state access space.



**Figure 8.21 Example of Read Strobe Timing**

### 8.6.6 Extension of Chip Select ( $\overline{CS}$ ) Assertion Period

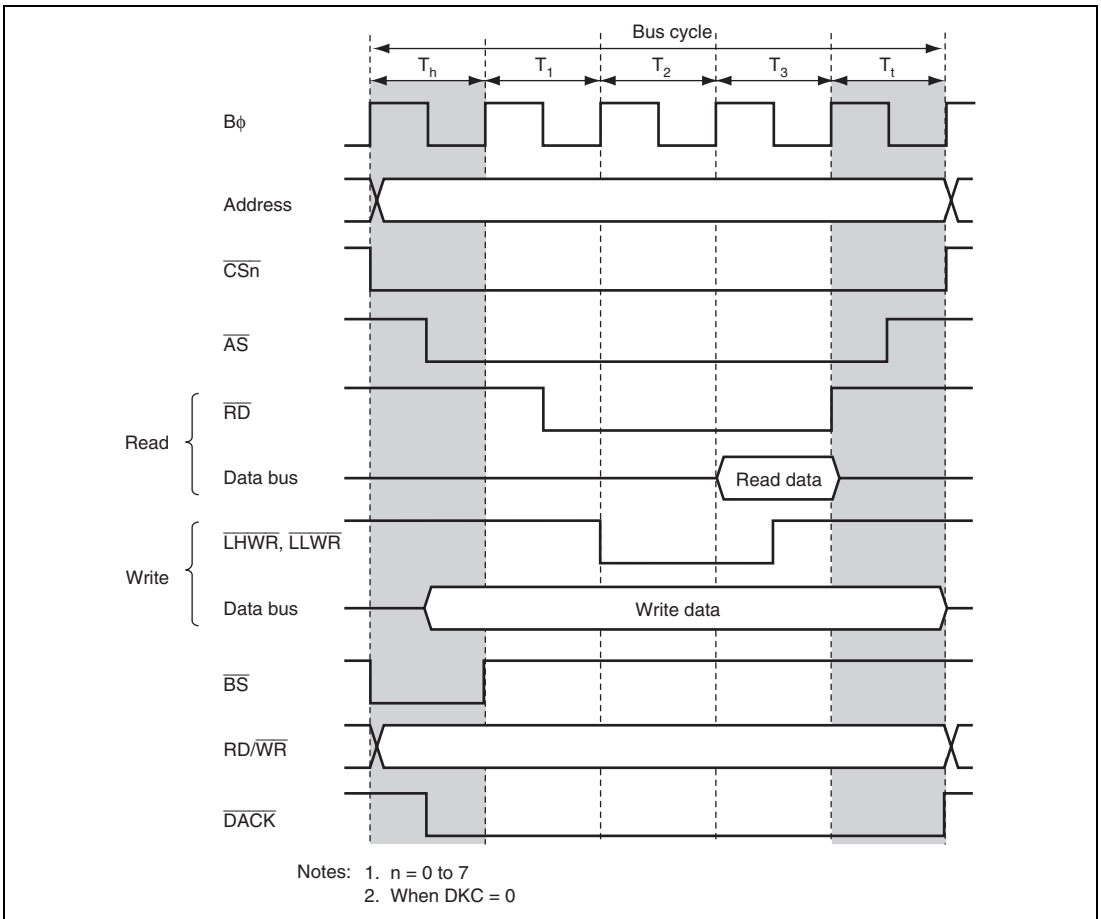
Some external I/O devices require a setup time and hold time between address and  $\overline{CS}$  signals and strobe signals such as  $\overline{RD}$ ,  $\overline{LHWR}$ , and  $\overline{LLWR}$ .

Settings can be made in CSACR to insert cycles in which only the  $\overline{CS}$ ,  $\overline{AS}$ , and address signals are asserted before and after a basic bus space access cycle. Extension of the  $\overline{CS}$  assertion period can be set in area units. With the  $\overline{CS}$  assertion extension period in write access, the data setup and hold times are less stringent since the write data is output to the data bus.

Figure 8.22 shows an example of the timing when the  $\overline{CS}$  assertion period is extended in basic bus 3-state access space.

Both extension cycle  $T_h$  inserted before the basic bus cycle and extension cycle  $T_t$  inserted after the basic bus cycle, or only one of these, can be specified for individual areas. Insertion or non-insertion can be specified for the  $T_h$  cycle with the upper eight bits (CSXH7 to CSXH0) in CSACR, and for the  $T_t$  cycle with the lower eight bits (CSXT7 to CSXT0).



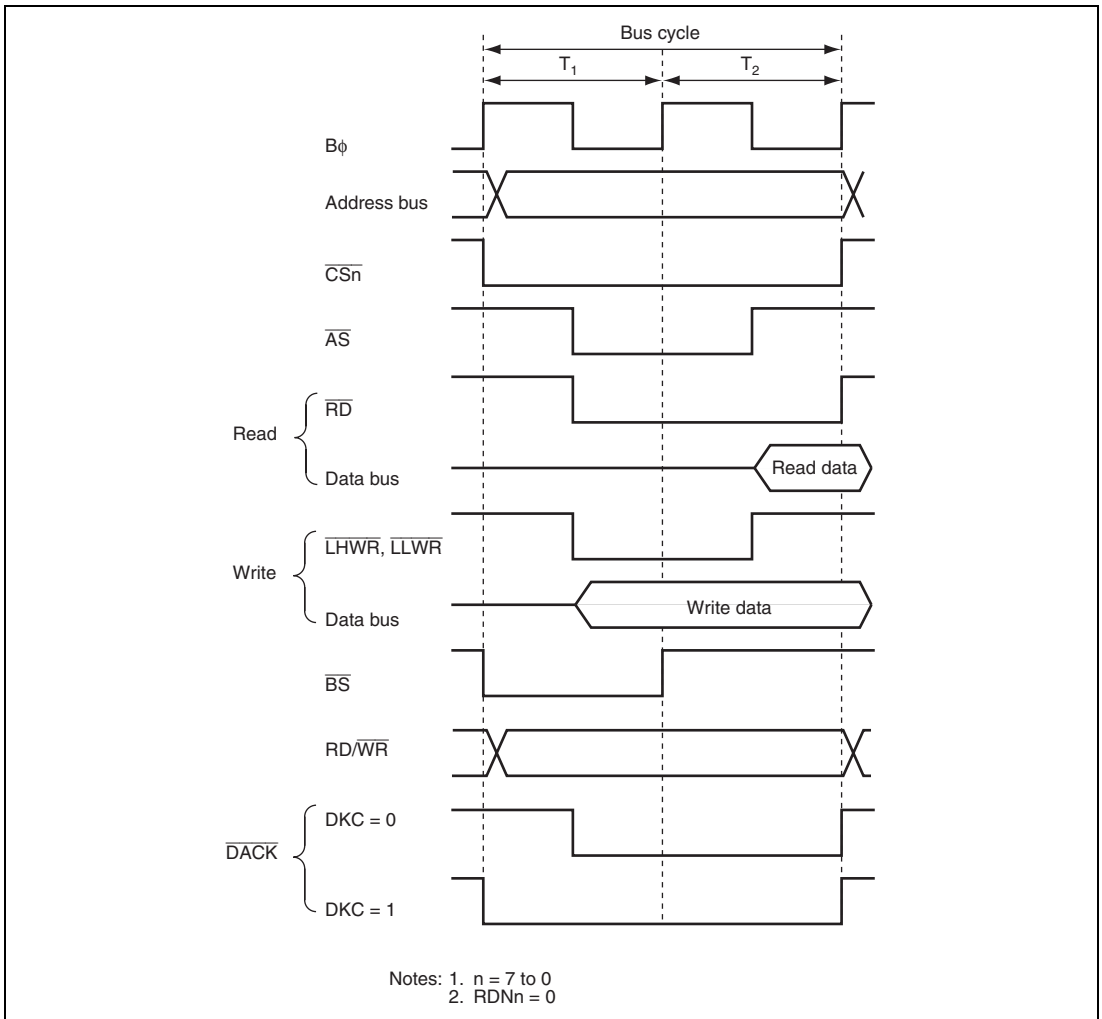


**Figure 8.22 Example of Timing when Chip Select Assertion Period is Extended**

### 8.6.7 $\overline{\text{DACK}}$ Signal Output Timing

For DMAC single address transfers, the  $\overline{\text{DACK}}$  signal assert timing can be modified by using the DKC bit in BCR1.

Figure 8.23 shows the  $\overline{\text{DACK}}$  signal output timing. Setting the DKC bit to 1 asserts the  $\overline{\text{DACK}}$  signal a half cycle earlier.



**Figure 8.23  $\overline{\text{DACK}}$  Signal Output Timing**

## 8.7 Byte Control SRAM Interface

The byte control SRAM interface is a memory interface for outputting a byte select strobe during a read or a write bus cycle. This interface has 16-bit data input/output pins and can be connected to the SRAM that has the upper byte select and the lower byte select strobes such as  $\overline{UB}$  and  $\overline{LB}$ .

The operation of the byte control SRAM interface is the same as the basic bus interface except that: the byte select strobes ( $\overline{LUB}$  and  $\overline{LLB}$ ) are output from the write strobe output pins ( $\overline{LHWR}$  and  $\overline{LLWR}$ ), respectively; the read strobe ( $\overline{RD}$ ) negation timing is a half cycle earlier than that in the case where  $RDNn = 0$  in the basic bus interface regardless of the  $RDNCR$  settings; and the  $\overline{RD}/\overline{WR}$  signal is used as write enable.

### 8.7.1 Byte Control SRAM Space Setting

Byte control SRAM interface can be specified for areas 0 to 7. Each area can be specified as byte control SRAM interface by setting bits  $BCSELn$  ( $n = 0$  to 7) in  $SRAMCR$ . For the area specified as burst ROM interface or address/data multiplexed I/O interface, the  $SRAMCR$  setting is invalid and byte control SRAM interface cannot be used.

### 8.7.2 Data Bus

The bus width of the byte control SRAM space can be specified as 16-bit byte control SRAM space according to bits  $ABWHn$  and  $ABWLn$  ( $n = 0$  to 7) in  $ABWCR$ . The area specified as 8-bit access space cannot be specified as the byte control SRAM space.

For the 16-bit byte control SRAM space, data bus (D15 to D0) is valid.

Access size and data alignment are the same as the basic bus interface. For details, see section 8.5.6, Endian and Data Alignment.

### 8.7.3 I/O Pins Used for Byte Control SRAM Interface

Table 8.16 shows the pins used for the byte control SRAM interface.

In the byte control SRAM interface, write strobe signals ( $\overline{\text{LHWR}}$  and  $\overline{\text{LLWR}}$ ) are output from the byte select strobes. The  $\text{RD}/\overline{\text{WR}}$  signal is used as a write enable signal.

**Table 8.16 I/O Pins for Byte Control SRAM Interface**

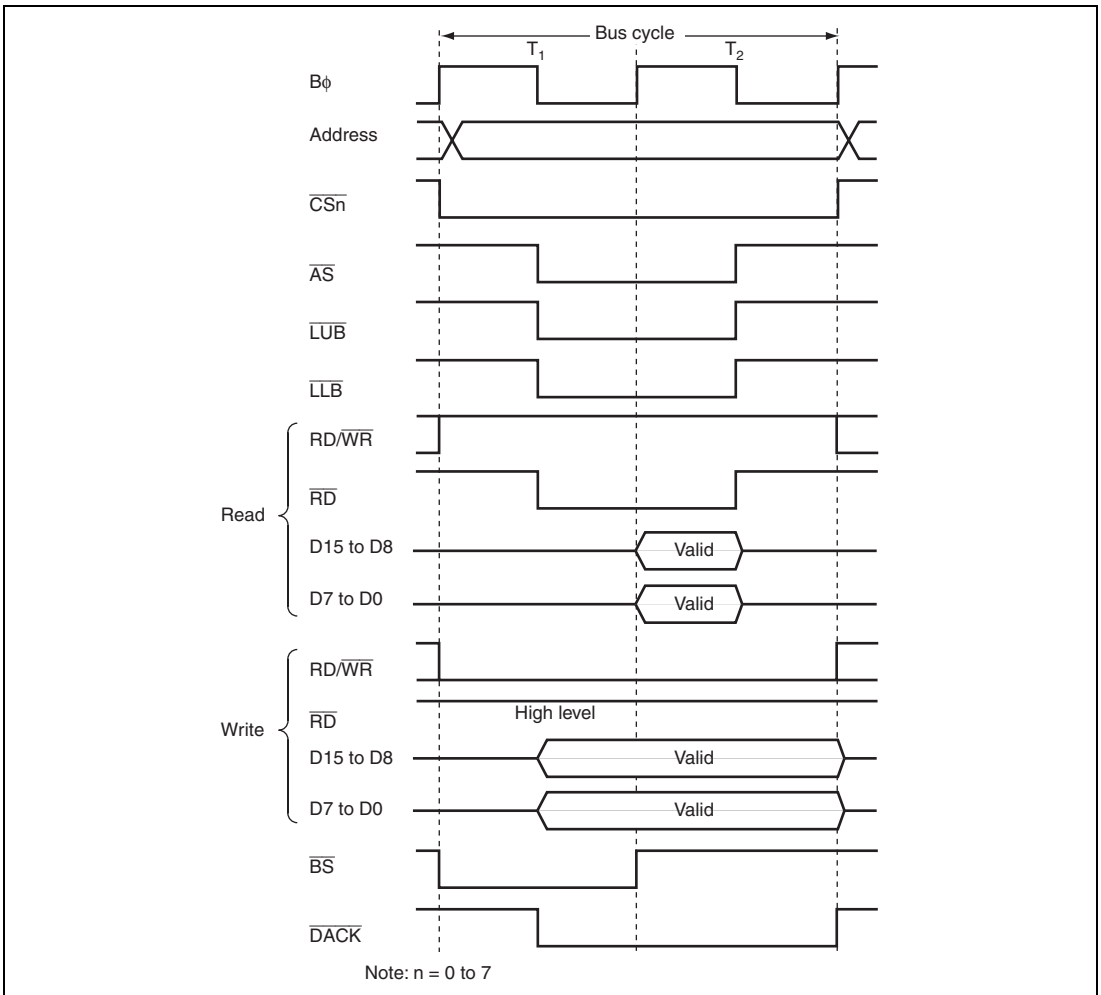
Pin	When Byte Control SRAM is Specified	Name	I/O	Function
$\overline{\text{AS/AH}}$	$\overline{\text{AS}}$	Address strobe	Output	Strobe signal indicating that the address output on the address bus is valid when a basic bus interface space or byte control SRAM space is accessed
$\overline{\text{CSn}}$	$\overline{\text{CSn}}$	Chip select	Output	Strobe signal indicating that area n is selected
$\overline{\text{RD}}$	$\overline{\text{RD}}$	Read strobe	Output	Output enable for the SRAM when the byte control SRAM space is accessed
$\text{RD}/\overline{\text{WR}}$	$\text{RD}/\overline{\text{WR}}$	Read/write	Output	Write enable signal for the SRAM when the byte control SRAM space is accessed
$\overline{\text{LHWR/LUB}}$	$\overline{\text{LUB}}$	Lower-upper byte select	Output	Upper byte select when the 16-bit byte control SRAM space is accessed
$\overline{\text{LLWR/LLB}}$	$\overline{\text{LLB}}$	Lower-lower byte select	Output	Lower byte select when the 16-bit byte control SRAM space is accessed
$\overline{\text{WAIT}}$	$\overline{\text{WAIT}}$	Wait	Input	Wait request signal used when an external address space is accessed
A20 to A0	A20 to A0	Address pin	Output	Address output pin
D15 to D0	D15 to D0	Data pin	Input/output	Data input/output pin

## 8.7.4 Basic Timing

### (1) 2-State Access Space

Figure 8.24 shows the bus timing when the byte control SRAM space is specified as a 2-state access space.

Data buses used for 16-bit access space is the same as those in basic bus interface. No wait cycles can be inserted.

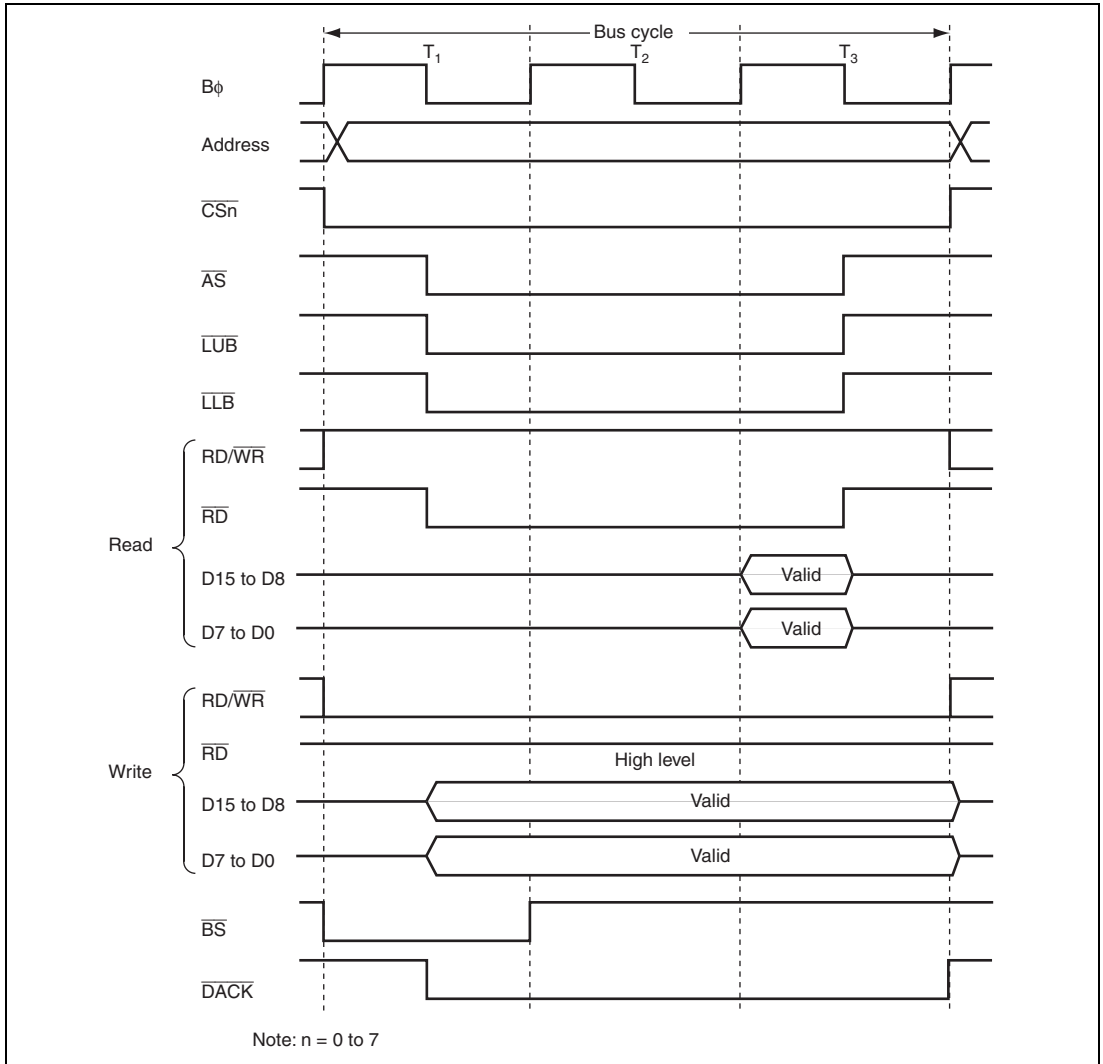


**Figure 8.24 16-Bit 2-State Access Space Bus Timing**

**(2) 3-State Access Space**

Figure 8.25 shows the bus timing when the byte control SRAM space is specified as a 3-state access space.

Data buses used for 16-bit access space is the same as those in the basic bus interface. Wait cycles can be inserted.



**Figure 8.25 16-Bit 3-State Access Space Bus Timing**

### 8.7.5 Wait Control

The bus cycle can be extended for the byte control SRAM interface by inserting wait cycles ( $T_w$ ) in the same way as the basic bus interface.

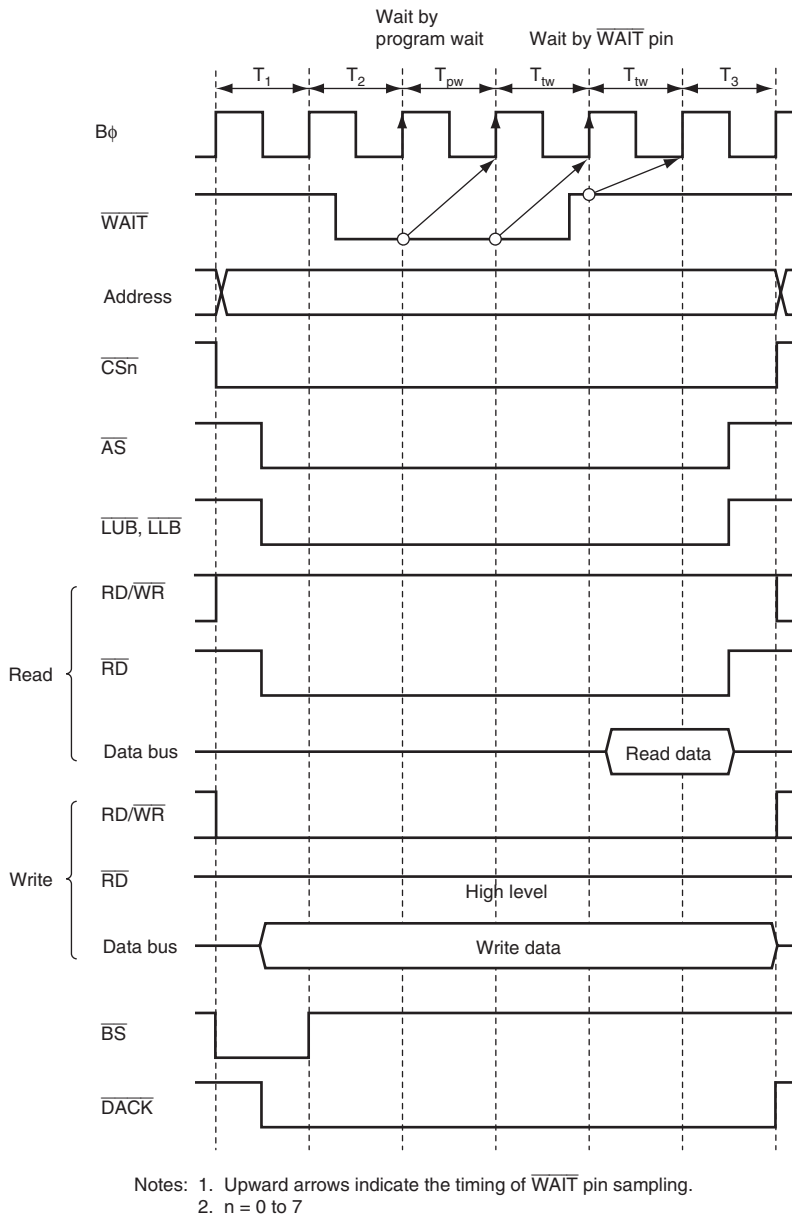
#### (1) Program Wait Insertion

From 0 to 7 wait cycles can be inserted automatically between T2 cycle and T3 cycle for the 3-state access space in area units, according to the settings in WTCRA and WTCRB.

#### (2) Pin Wait Insertion

For 3-state access space, when the WAITE bit in BCR1 is set to 1, the corresponding DDR bit is cleared to 0, and the ICR bit is set to 1, wait input by means of the  $\overline{\text{WAIT}}$  pin is enabled. For details on DDR and ICR, see section 11, I/O Ports.

Figure 8.26 shows an example of wait cycle insertion timing.



**Figure 8.26 Example of Wait Cycle Insertion Timing**



### 8.7.6 Read Strobe ( $\overline{\text{RD}}$ )

When the byte control SRAM space is specified, the RDNCR setting for the corresponding space is invalid.

The read strobe negation timing is the same timing as when  $\text{RDNn} = 1$  in the basic bus interface. Note that the  $\overline{\text{RD}}$  timing with respect to the  $\overline{\text{DACK}}$  rising edge becomes different.

### 8.7.7 Extension of Chip Select ( $\overline{\text{CS}}$ ) Assertion Period

In the byte control SRAM interface, the extension cycles can be inserted before and after the bus cycle in the same way as the basic bus interface. For details, see section 8.6.6, Extension of Chip Select ( $\overline{\text{CS}}$ ) Assertion Period.

### 8.7.8 $\overline{\text{DACK}}$ Signal Output Timing

For DMAC single address transfers, the  $\overline{\text{DACK}}$  signal assert timing can be modified by using the DKC bit in BCR1.

Figure 8.27 shows the  $\overline{\text{DACK}}$  signal output timing. Setting the DKC bit to 1 asserts the  $\overline{\text{DACK}}$  signal a half cycle earlier.

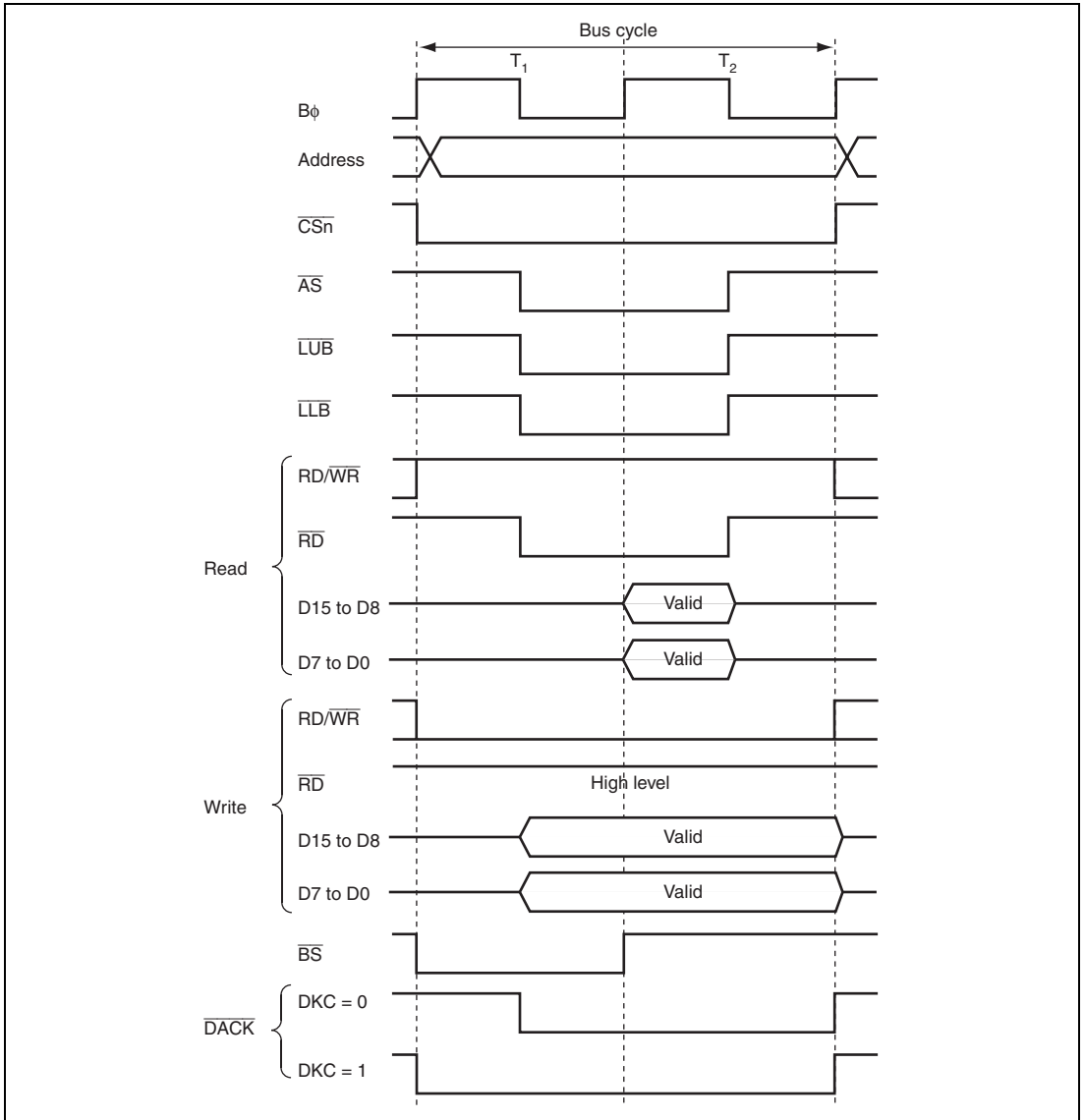


Figure 8.27  $\overline{\text{DACK}}$  Signal Output Timing

## 8.8 Burst ROM Interface

In this LSI, external address space areas 0 and 1 can be designated as burst ROM space, and burst ROM interfacing performed. The burst ROM interface enables ROM with page access capability to be accessed at high speed.

Areas 1 and 0 can be designated as burst ROM space by means of bits BSRM1 and BSRM0 in BROMCR. Consecutive burst accesses of up to 32 words can be performed, according to the setting of bits BSWDn1 and BSWDn0 ( $n = 0, 1$ ) in BROMCR. From one to eight cycles can be selected for burst access.

Settings can be made independently for area 0 and area 1.

In the burst ROM interface, the burst access covers only CPU read accesses. Other accesses are performed with the similar method to the basic bus interface.

### 8.8.1 Burst ROM Space Setting

Burst ROM interface can be specified for areas 0 and 1. Areas 0 and 1 can be specified as burst ROM space by setting bits BSRMn ( $n = 0, 1$ ) in BROMCR.

### 8.8.2 Data Bus

The bus width of the burst ROM space can be specified as 8-bit or 16-bit burst ROM interface space according to the ABWHn and ABWLn bits ( $n = 0, 1$ ) in ABWCR.

For the 8-bit bus width, data bus (D7 to D0) is valid. For the 16-bit bus width, data bus (D15 to D0) is valid.

Access size and data alignment are the same as the basic bus interface. For details, see section 8.5.6, Endian and Data Alignment.

### 8.8.3 I/O Pins Used for Burst ROM Interface

Table 8.17 shows the pins used for the burst ROM interface.

**Table 8.17 I/O Pins Used for Burst ROM Interface**

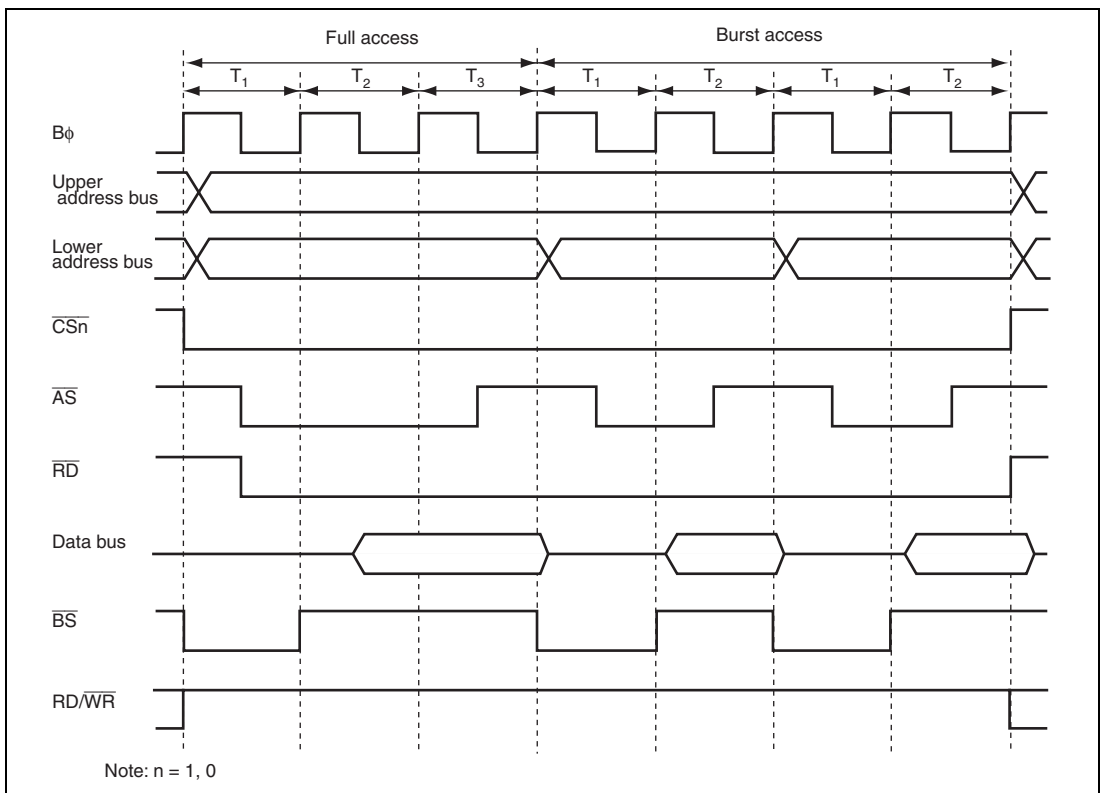
Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started.
Address strobe	$\overline{AS}$	Output	Strobe signal indicating that an address output on the address bus is valid during access
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Read/write	$\overline{RD}/\overline{WR}$	Output	Signal indicating the data bus input or output direction
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid during write access
Low-low write	$\overline{LLWR}$	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid during write access
Chip select 0 to 7	$\overline{CS0}$ to $\overline{CS7}$	Output	Strobe signal indicating that the area is selected
Wait	$\overline{WAIT}$	Input	Wait request signal used when an external address space is accessed

### 8.8.4 Basic Timing

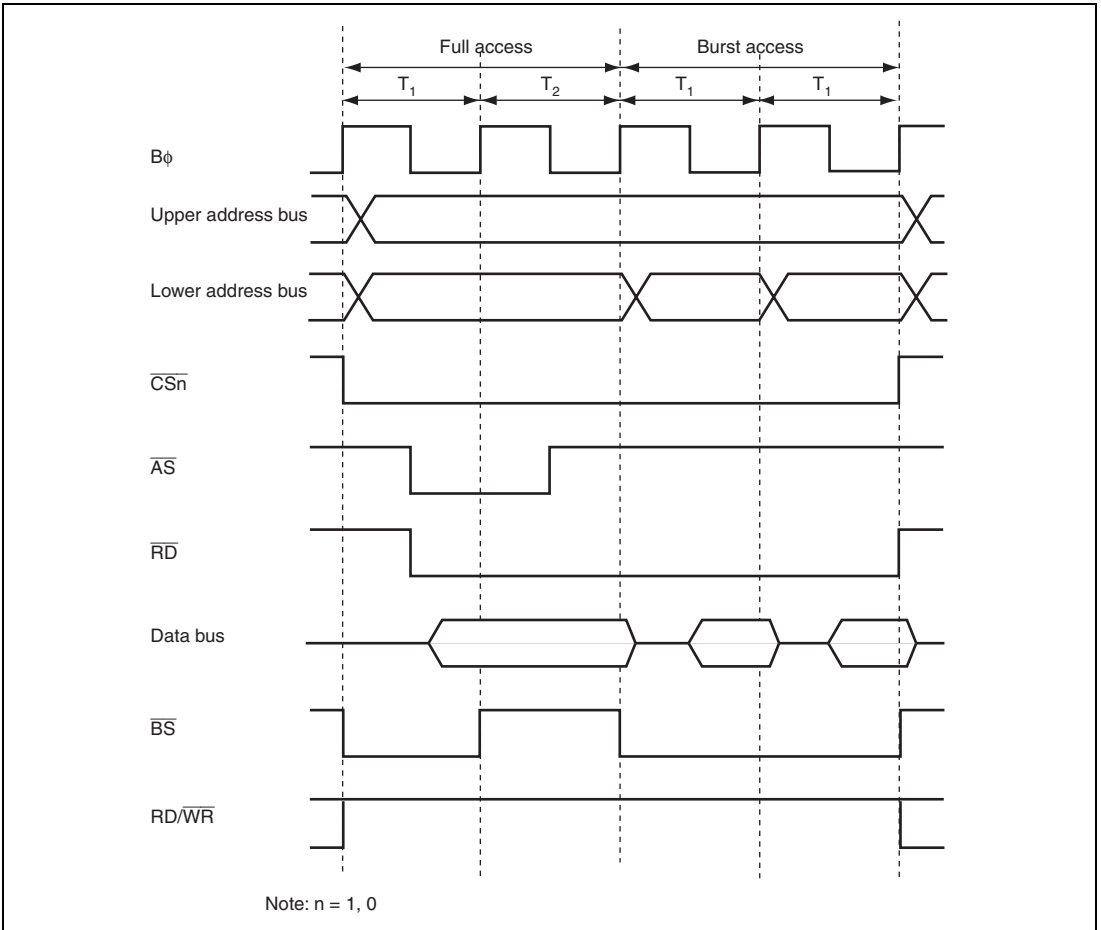
The number of access cycles in the initial cycle (full access) on the burst ROM interface is determined by the basic bus interface settings in ABWCR, ASTCR, WTCRA, WTCRB, and bits CSX $H_n$  in CSACR ( $n = 0$  to 7). When area 0 or area 1 designated as burst ROM space is read by the CPU, the settings in RDNCR and bits CSXT $n$  in CSACR ( $n = 0$  to 7) are ignored.

From one to eight cycles can be selected for the burst cycle, according to the settings of bits BSTS02 to BSTS00 and BSTS12 to BSTS10 in BROMCR. Wait cycles cannot be inserted. In addition, 4-word, 8-word, 16-word, or 32-word consecutive burst access can be performed according to the settings of BSTS01, BSTS00, BSTS11, and BSTS10 bits in BROMCR.

The basic access timing for burst ROM space is shown in figures 8.28 and 8.29.



**Figure 8.28 Example of Burst ROM Access Timing ( $AST_n = 1$ , Two Burst Cycles)**



**Figure 8.29 Example of Burst ROM Access Timing (ASTn = 0, One Burst Cycle)**

### 8.8.5 Wait Control

As with the basic bus interface, either program wait insertion or pin wait insertion by the  $\overline{\text{WAIT}}$  pin can be used in the initial cycle (full access) on the burst ROM interface. See section 8.6.4, Wait Control. Wait cycles cannot be inserted in a burst cycle.

### 8.8.6 Read Strobe ( $\overline{\text{RD}}$ ) Timing

When the burst ROM space is read by the CPU, the RDNCR setting for the corresponding space is invalid.

The read strobe negation timing is the same timing as when  $\text{RDNn} = 0$  in the basic bus interface.

### 8.8.7 Extension of Chip Select ( $\overline{\text{CS}}$ ) Assertion Period

In the burst ROM interface, the extension cycles can be inserted in the same way as the basic bus interface.

For the burst ROM space, the burst access can be enabled only in read access by the CPU. In this case, the setting of the corresponding  $\text{CSXTn}$  bit in  $\text{CSACR}$  is ignored and an extension cycle can be inserted only before the full access cycle. Note that no extension cycle can be inserted before or after the burst access cycles.

In accesses other than read accesses by the CPU, the burst ROM space is equivalent to the basic bus interface space. Accordingly, extension cycles can be inserted before and after the burst access cycles.

## 8.9 Address/Data Multiplexed I/O Interface

If areas 3 to 7 of external address space are specified as address/data multiplexed I/O space in this LSI, the address/data multiplexed I/O interface can be performed. In the address/data multiplexed I/O interface, peripheral LSIs that require the multiplexed address/data can be connected directly to this LSI.

### 8.9.1 Address/Data Multiplexed I/O Space Setting

Address/data multiplexed I/O interface can be specified for areas 3 to 7. Each area can be specified as the address/data multiplexed I/O space by setting bits MPXEn (n = 3 to 7) in MPXCR.

### 8.9.2 Address/Data Multiplex

In the address/data multiplexed I/O space, data bus is multiplexed with address bus. Table 8.18 shows the relationship between the bus width and address output.

**Table 8.18 Address/Data Multiplex**

Bus Width	Cycle	Data Pins															
		PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
8 bits	Address	-	-	-	-	-	-	-	-	A7	A6	A5	A4	A3	A2	A1	A0
	Data	-	-	-	-	-	-	-	-	D7	D6	D5	D4	D3	D2	D1	D0
16 bits	Address	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
	Data	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

### 8.9.3 Data Bus

The bus width of the address/data multiplexed I/O space can be specified for either 8-bit access space or 16-bit access space by the ABWHn and ABWLn bits (n = 3 to 7) in ABWCR.

For the 8-bit access space, D7 to D0 are valid for both address and data. For 16-bit access space, D15 to D0 are valid for both address and data. If the address/data multiplexed I/O space is accessed, the corresponding address will be output to the address bus.

For details on access size and data alignment, see section 8.5.6, Endian and Data Alignment.



### 8.9.4 I/O Pins Used for Address/Data Multiplexed I/O Interface

Table 8.19 shows the pins used for the address/data multiplexed I/O Interface.

**Table 8.19 I/O Pins for Address/Data Multiplexed I/O Interface**

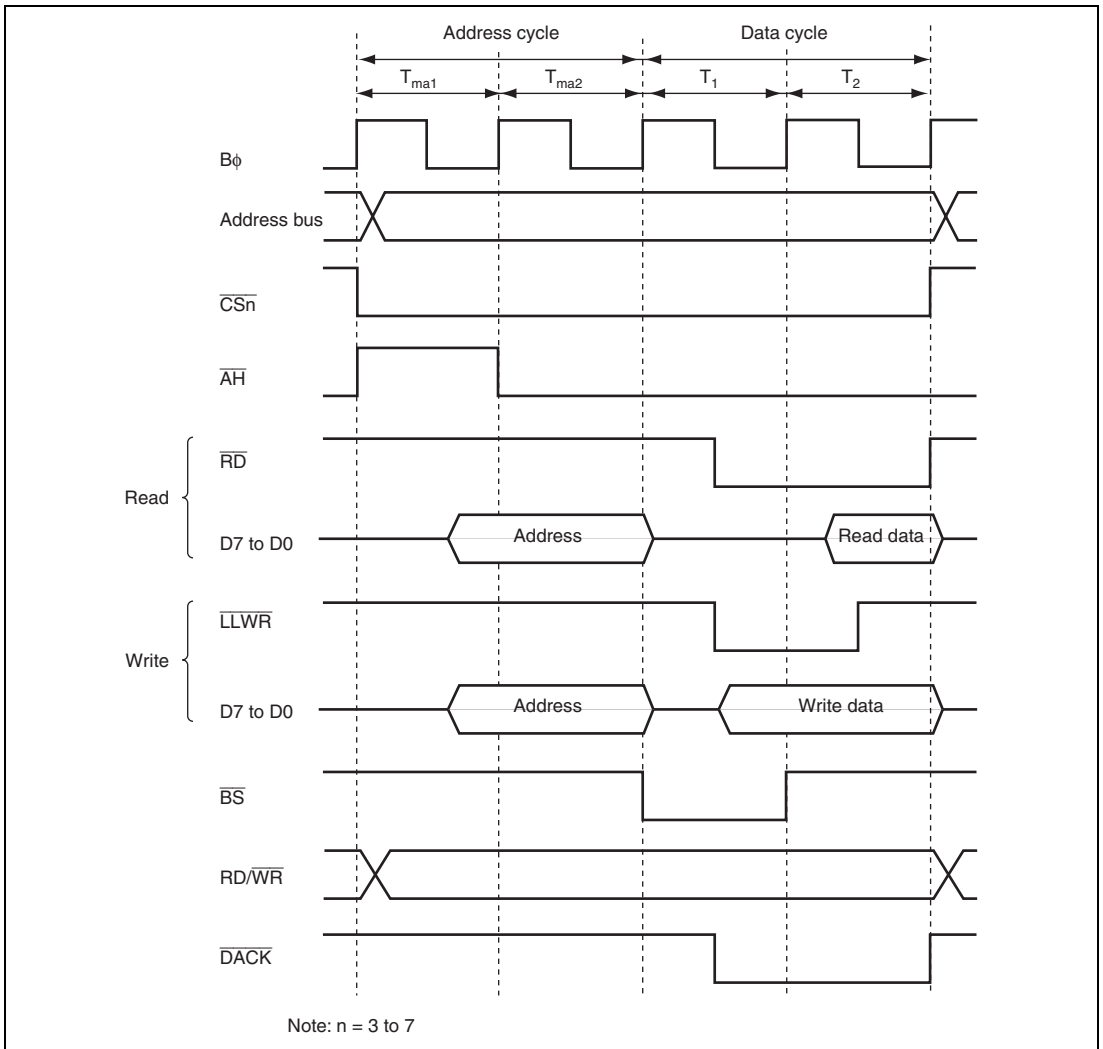
Pin	When Byte Control SRAM is Specified	Name	I/O	Function
$\overline{CSn}$	$\overline{CSn}$	Chip select	Output	Chip select (n = 3 to 7) when area n is specified as the address/data multiplexed I/O space
$\overline{AS/AH}$	$\overline{AH}^*$	Address hold	Output	Signal to hold an address when the address/data multiplexed I/O space is specified
$\overline{RD}$	$\overline{RD}$	Read strobe	Output	Signal indicating that the address/data multiplexed I/O space is being read
$\overline{LHWR/LUB}$	$\overline{LHWR}$	Low-high write	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid when the address/data multiplexed I/O space is written
$\overline{LLWR/LLB}$	$\overline{LLWR}$	Low-low write	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid when the address/data multiplexed I/O space is written
D15 to D0	D15 to D0	Address/data	Input/output	Address and data multiplexed pins for the address/data multiplexed I/O space.  Only D7 to D0 are valid when the 8-bit space is specified. D15 to D0 are valid when the 16-bit space is specified.
A20 to A0	A20 to A0	Address	Output	Address output pin
$\overline{WAIT}$	$\overline{WAIT}$	Wait	Input	Wait request signal used when the external address space is accessed
$\overline{BS}$	$\overline{BS}$	Bus cycle start	Output	Signal to indicate the bus cycle start
$\overline{RD/\overline{WR}}$	$\overline{RD/\overline{WR}}$	Read/write	Output	Signal indicating the data bus input or output direction

Note: \* The  $\overline{AH}$  output is multiplexed with the  $\overline{AS}$  output. At the timing that an area is specified as address/data multiplexed I/O, this pin starts to function as the  $\overline{AH}$  output meaning that this pin cannot be used as the  $\overline{AS}$  output. At this time, when other areas set to the basic bus interface is accessed, this pin does not function as the  $\overline{AS}$  output. Until an area is specified as address/data multiplexed I/O, be aware that this pin functions as the  $\overline{AS}$  output.

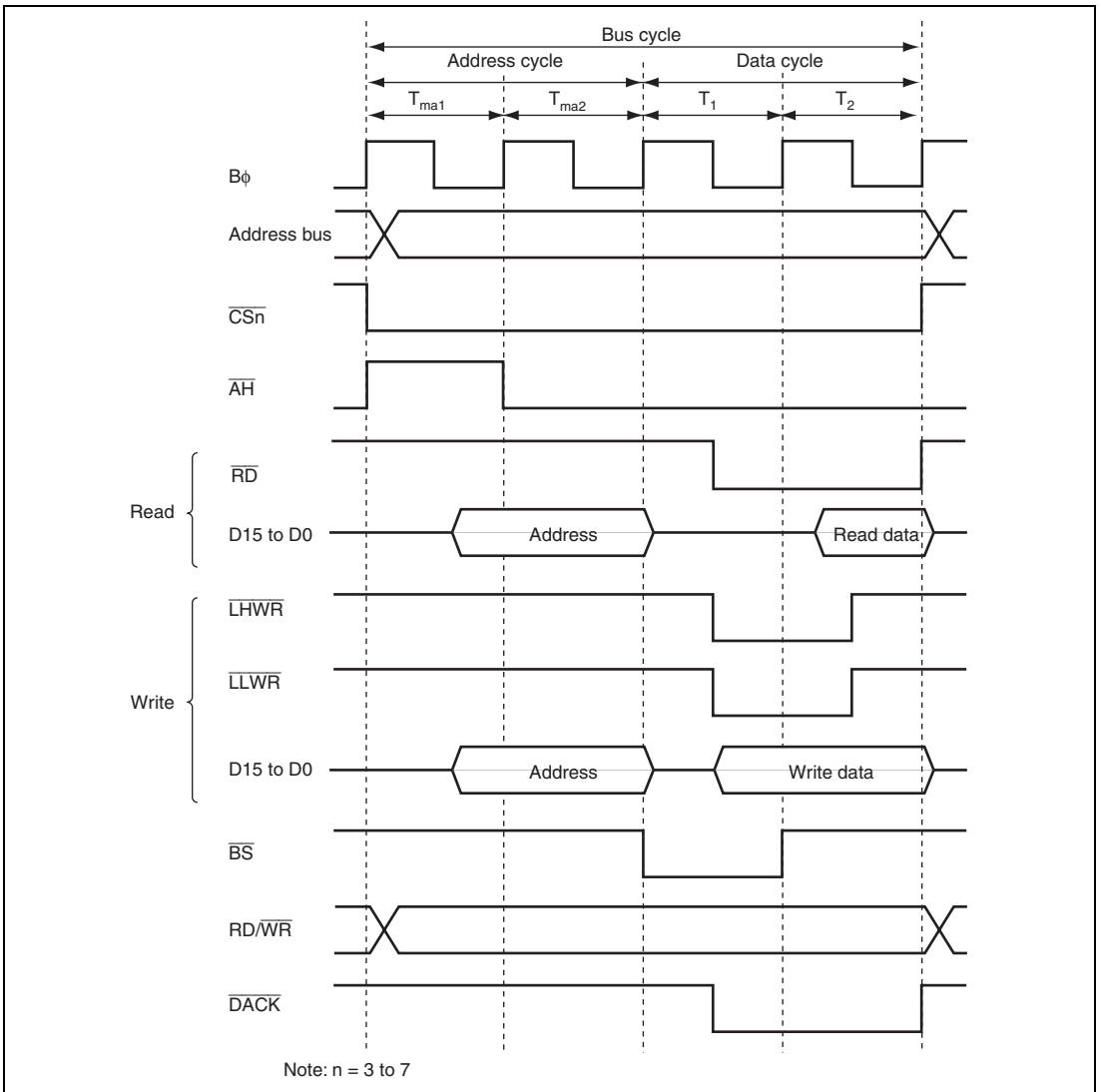
### 8.9.5 Basic Timing

The bus cycle in the address/data multiplexed I/O interface consists of an address cycle and a data cycle. The data cycle is based on the basic bus interface timing specified by the ABWCR, ASTCR, WTCRA, WTCRB, RDNCR, and CSACR.

Figures 8.30 and 8.31 show the basic access timings.



**Figure 8.30 8-Bit Access Space Access Timing (ABWHn = 1, ABWLn = 1)**

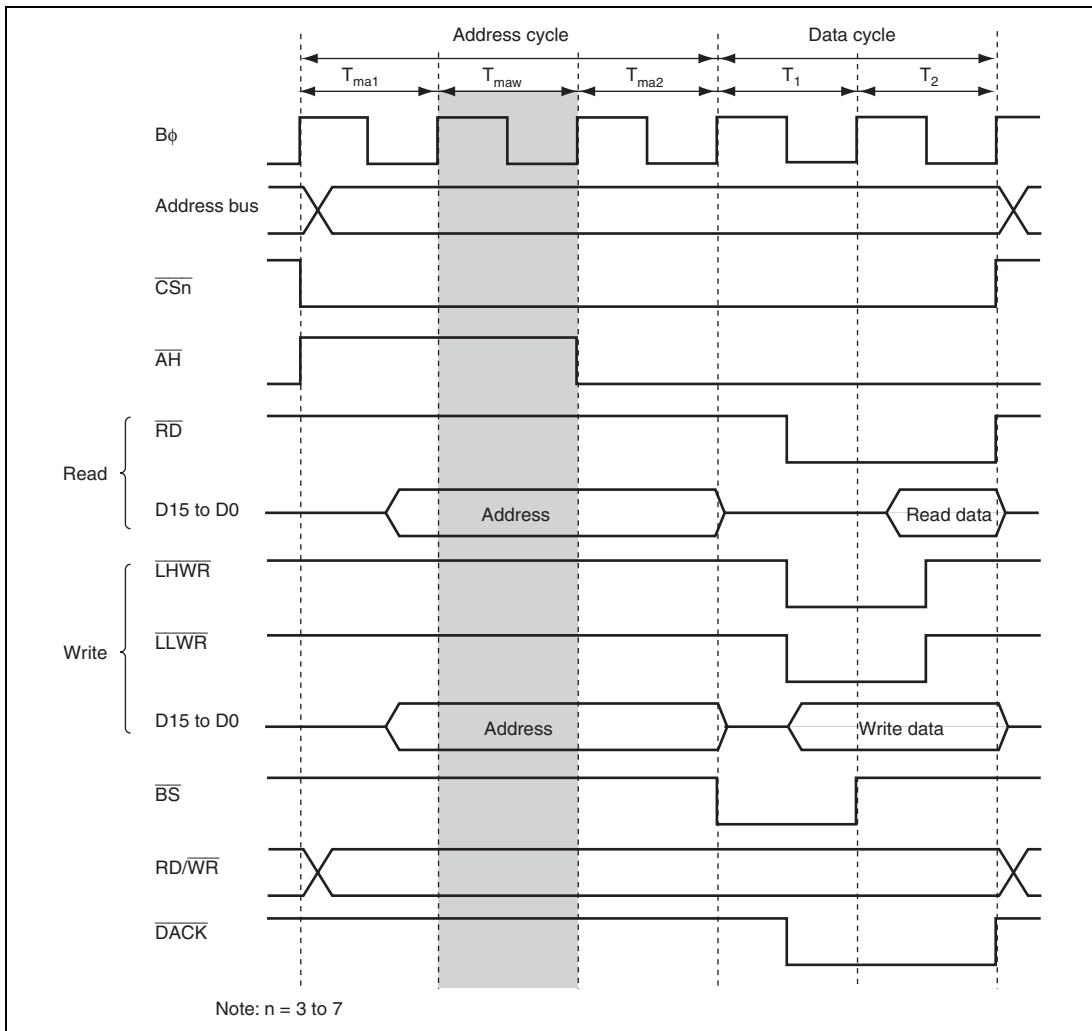


**Figure 8.31 16-Bit Access Space Access Timing (ABWHn = 0, ABWLn = 1)**

### 8.9.6 Address Cycle Control

An extension cycle ( $T_{maw}$ ) can be inserted between  $T_{ma1}$  and  $T_{ma2}$  cycles to extend the  $\overline{AH}$  signal output period by setting the  $\overline{ADDEX}$  bit in  $MPXCR$ . By inserting the  $T_{maw}$  cycle, the address setup for  $\overline{AH}$  and the  $\overline{AH}$  minimum pulse width can be assured.

Figure 8.32 shows the access timing when the address cycle is three cycles.



**Figure 8.32 Access Timing of 3 Address Cycles ( $\overline{ADDEX} = 1$ )**

### 8.9.7 Wait Control

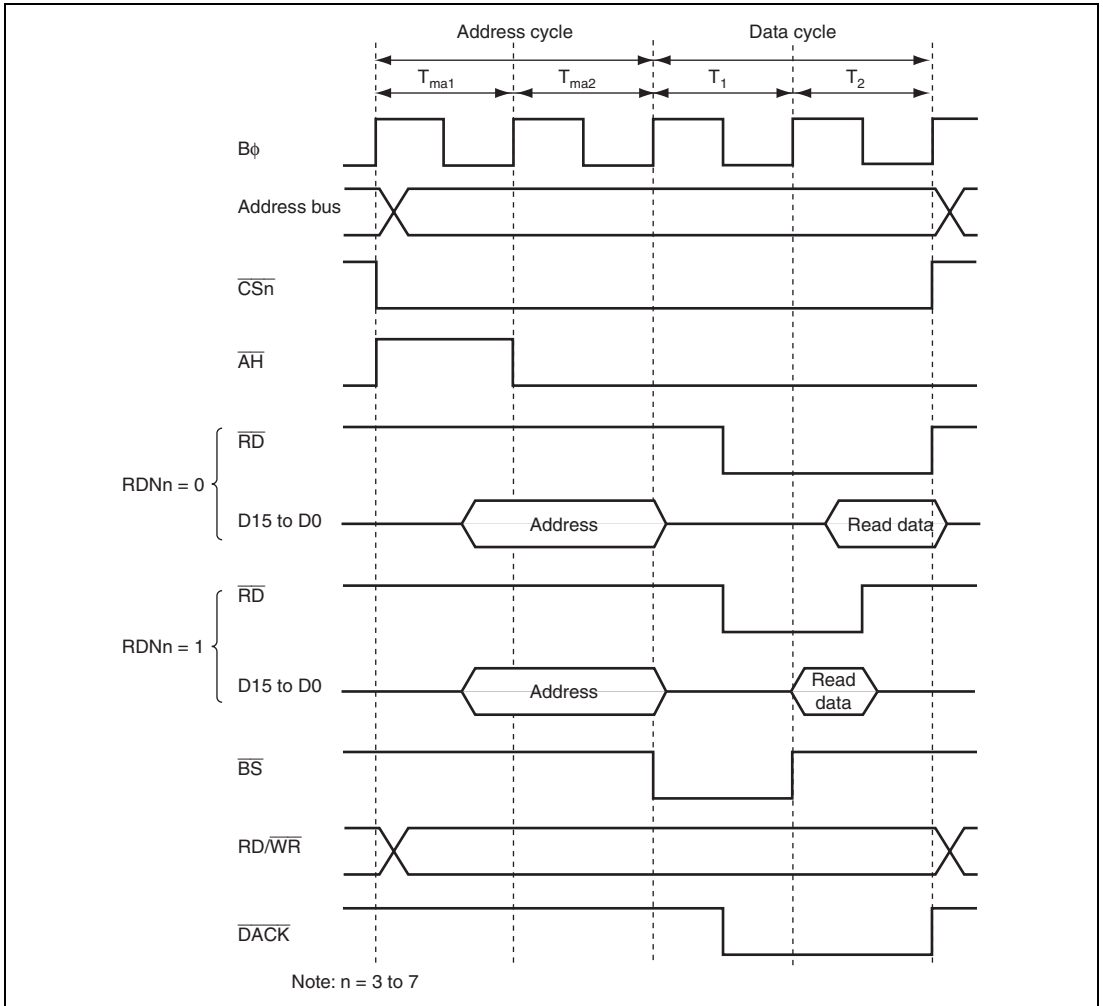
In the data cycle of the address/data multiplexed I/O interface, program wait insertion and pin wait insertion by the  $\overline{\text{WAIT}}$  pin are enabled in the same way as in the basic bus interface. For details, see section 8.6.4, Wait Control.

Wait control settings do not affect the address cycles.

### 8.9.8 Read Strobe ( $\overline{\text{RD}}$ ) Timing

In the address/data multiplexed I/O interface, the read strobe timing of data cycles can be modified in the same way as in basic bus interface. For details, see section 8.6.5, Read Strobe ( $\overline{\text{RD}}$ ) Timing.

Figure 8.33 shows an example when the read strobe timing is modified.

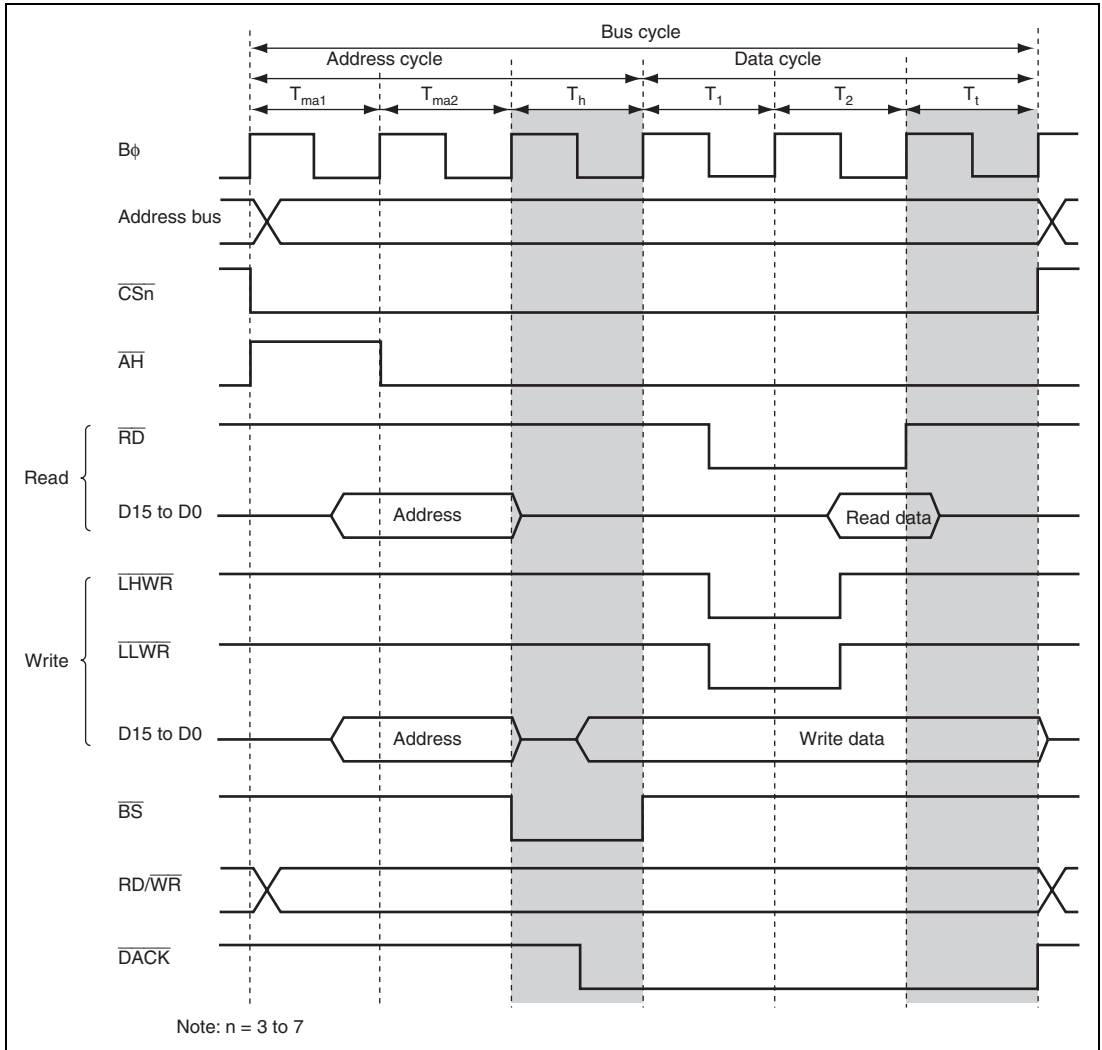


**Figure 8.33 Read Strobe Timing**

### 8.9.9 Extension of Chip Select ( $\overline{CS}$ ) Assertion Period

In the address/data multiplexed interface, the extension cycles can be inserted before and after the bus cycle. For details, see section 8.6.6, Extension of Chip Select ( $\overline{CS}$ ) Assertion Period.

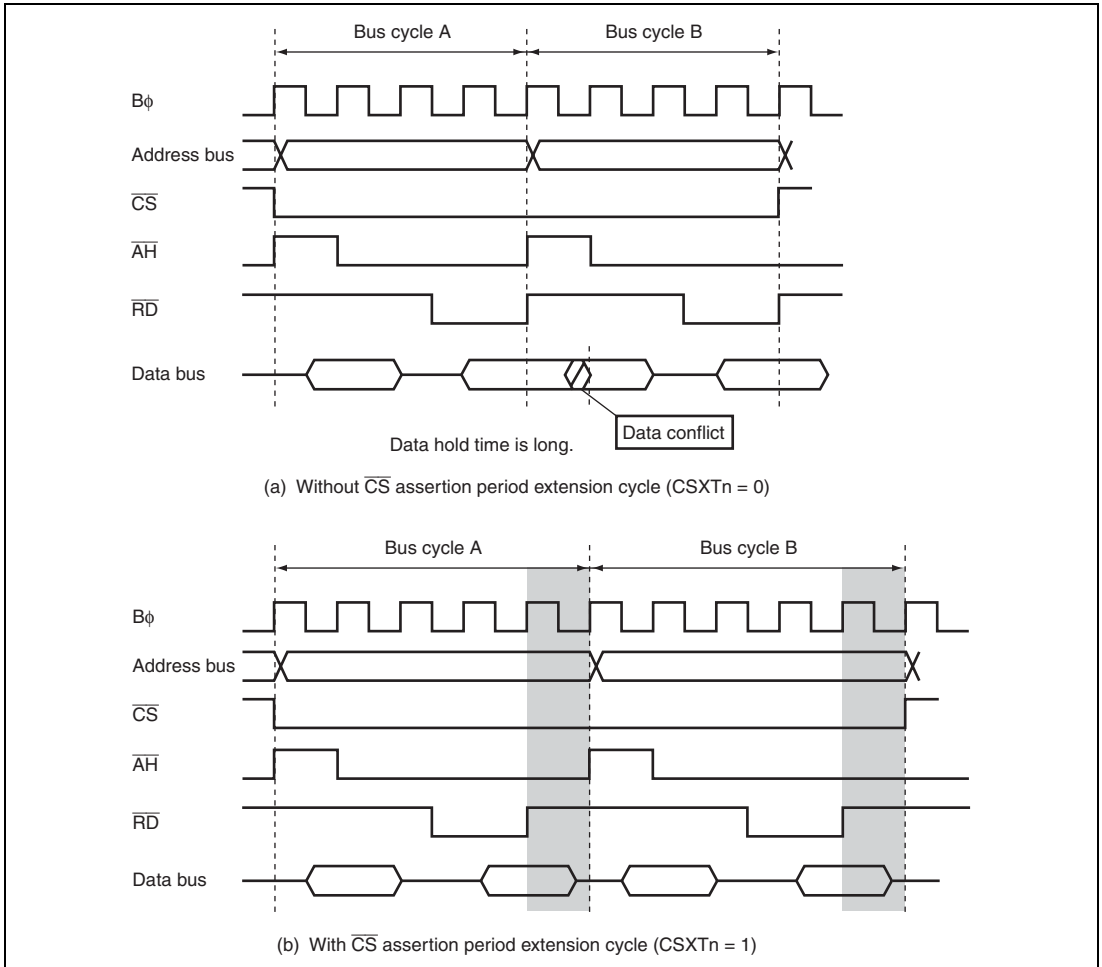
Figure 8.34 shows an example of the chip select ( $\overline{CS}$ ) assertion period extension timing.



**Figure 8.34 Chip Select ( $\overline{CS}$ ) Assertion Period Extension Timing in Data Cycle**

When consecutively reading from the same area connected to a peripheral LSI whose data hold time is long, data outputs from the peripheral LSI and this LSI may conflict. Inserting the chip select assertion period extension cycle after the access cycle can avoid the data conflict.

Figure 8.35 shows an example of the operation. In the figure, both bus cycles A and B are read access cycles to the address/data multiplexed I/O space. An example of the data conflict is shown in (a), and an example of avoiding the data conflict by the  $\overline{\text{CS}}$  assertion period extension cycle in (b).



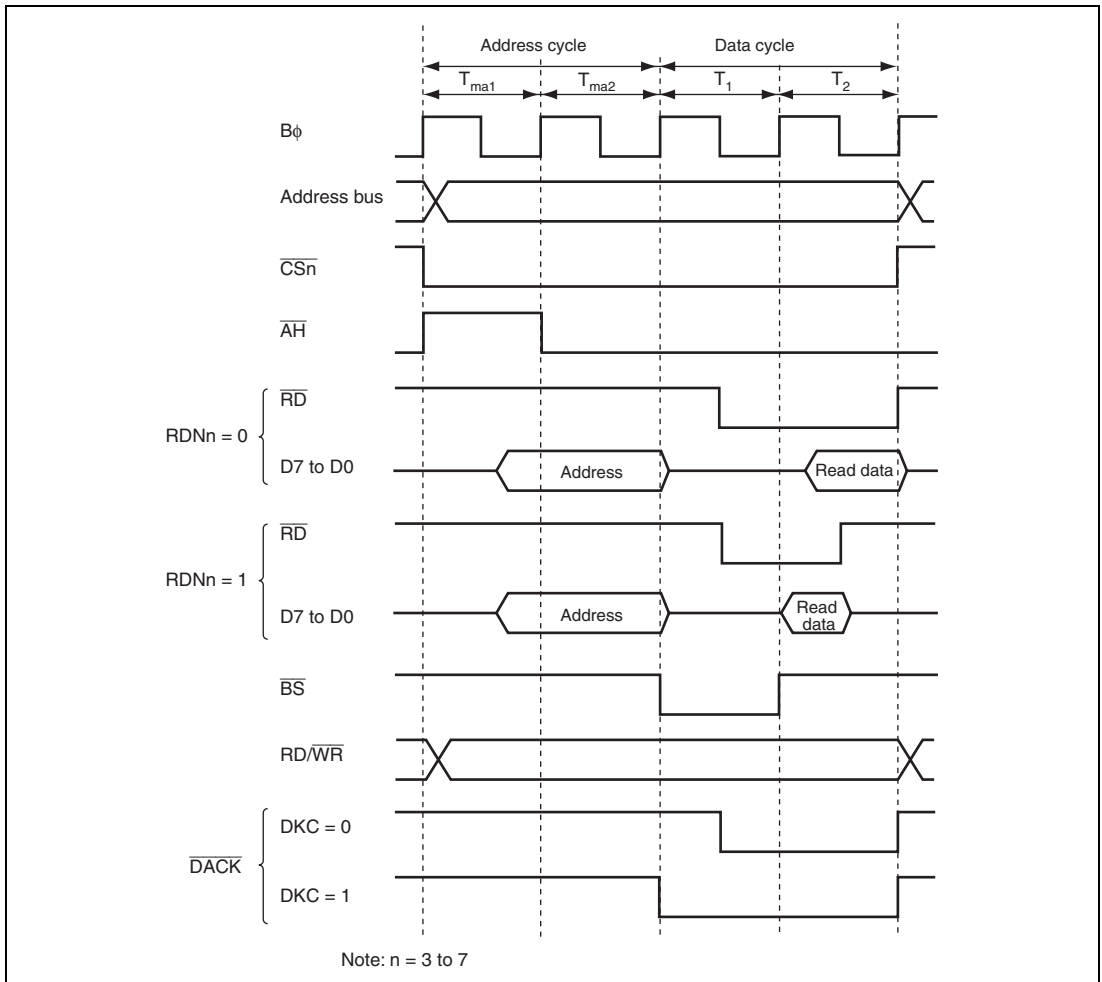
**Figure 8.35 Consecutive Read Accesses to Same Area  
(Address/Data Multiplexed I/O Space)**



### 8.9.10 $\overline{\text{DACK}}$ Signal Output Timing

For DMAC single address transfers, the  $\overline{\text{DACK}}$  signal assert timing can be modified by using the DKC bit in BCR1.

Figure 8.36 shows the  $\overline{\text{DACK}}$  signal output timing. Setting the DKC bit to 1 asserts the  $\overline{\text{DACK}}$  signal a half cycle earlier.



**Figure 8.36  $\overline{\text{DACK}}$  Signal Output Timing**

## 8.10 Idle Cycle

In this LSI, idle cycles can be inserted between the consecutive external accesses. By inserting the idle cycle, data conflicts between ROM read cycle whose output floating time is long and an access cycle from/to high-speed memory or I/O interface can be prevented.

### 8.10.1 Operation

When this LSI consecutively accesses external address space, it can insert an idle cycle between bus cycles in the following four cases. These conditions are determined by the sequence of read and write and previously accessed area.

1. When read cycles of different areas in the external address space occur consecutively
2. When an external write cycle occurs immediately after an external read cycle
3. When an external read cycle occurs immediately after an external write cycle
4. When an external access occurs immediately after a DMAC single address transfer (write cycle)

Up to four idle cycles can be inserted under the conditions shown above. The number of idle cycles to be inserted should be specified to prevent data conflicts between the output data from a previously accessed device and data from a subsequently accessed device.

Under conditions 1 and 2, which are the conditions to insert idle cycles after read, the number of idle cycles can be selected from setting A specified by bits IDLCA1 and IDLCA0 in IDLCR or setting B specified by bits IDLCB1 and IDLCB0 in IDLCR: Setting A can be selected from one to four cycles, and setting B can be selected from one or two to four cycles. Setting A or B can be specified for each area by setting bits IDLSEL7 to IDLSEL0 in IDLCR. Note that bits IDLSEL7 to IDLSEL0 correspond to the previously accessed area of the consecutive accesses.

The number of idle cycles to be inserted under conditions 3 and 4, which are conditions to insert idle cycles after write, can be determined by setting A as described above.

After the reset release, IDLCR is initialized to four idle cycle insertion under all conditions 1 to 4 shown above.

Table 8.20 shows the correspondence between conditions 1 to 4 and number of idle cycles to be inserted for each area. Table 8.21 shows the correspondence between the number of idle cycles to be inserted specified by settings A and B, and number of cycles to be inserted.

**Table 8.20 Number of Idle Cycle Insertion Selection in Each Area**

Insertion Condition	Bit Settings		Area for Previous Access								
	n	IDLSn	IDLSELn	0	1	2	3	4	5	6	7
		Setting	n = 0 to 7								
Consecutive reads in different areas	1	0	—	Invalid							
		1	0	A	A	A	A	A	A	A	A
			1	B	B	B	B	B	B	B	B
Write after read	0	0	—	Invalid							
		1	0	A	A	A	A	A	A	A	A
			1	B	B	B	B	B	B	B	B
Read after write	2	0	—	Invalid							
		1		A							
External access after single address transfer	3	0	—	Invalid							
		1		A							

[Legend]

A: Number of idle cycle insertion A is selected.

B: Number of idle cycle insertion B is selected.

Invalid: No idle cycle is inserted for the corresponding condition.

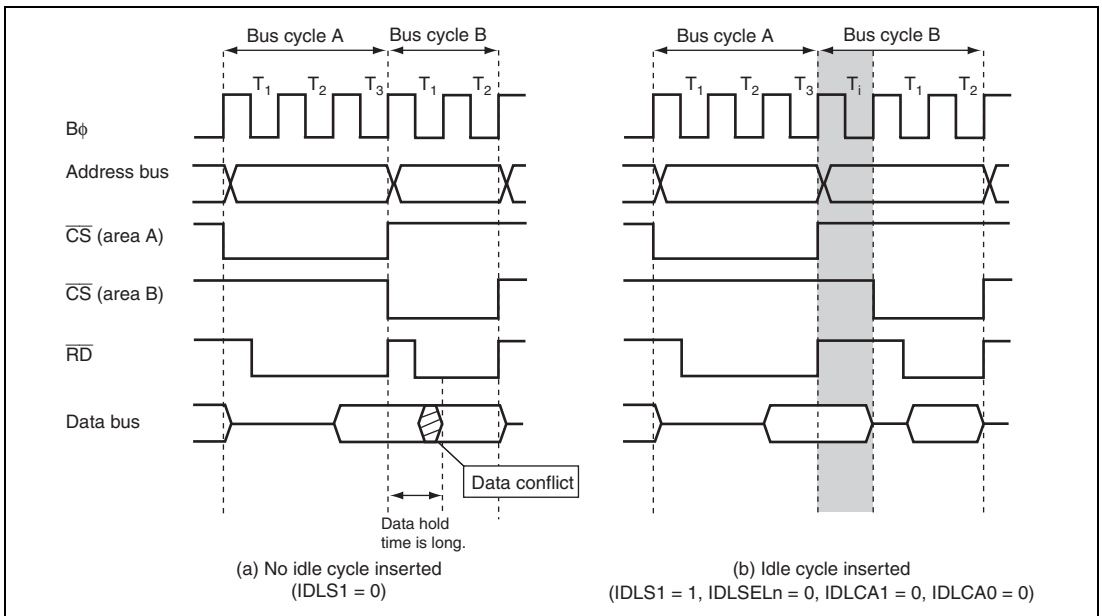
**Table 8.21 Number of Idle Cycle Insertions**

Bit Settings				Number of Cycles
A		B		
IDLCA1	IDLCA0	IDLCB1	IDLCB0	
—	—	0	0	0
0	0	—	—	1
0	1	0	1	2
1	0	1	0	3
1	1	1	1	4

### (1) Consecutive Reads in Different Areas

If consecutive reads in different areas occur while bit IDLS1 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 when bit IDLSELn in IDLCR is cleared to 0, or bits IDLCB1 and IDLCB0 when bit IDLSELn is set to 1 are inserted at the start of the second read cycle (n = 0 to 7).

Figure 8.37 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a read cycle for SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data conflict is prevented.

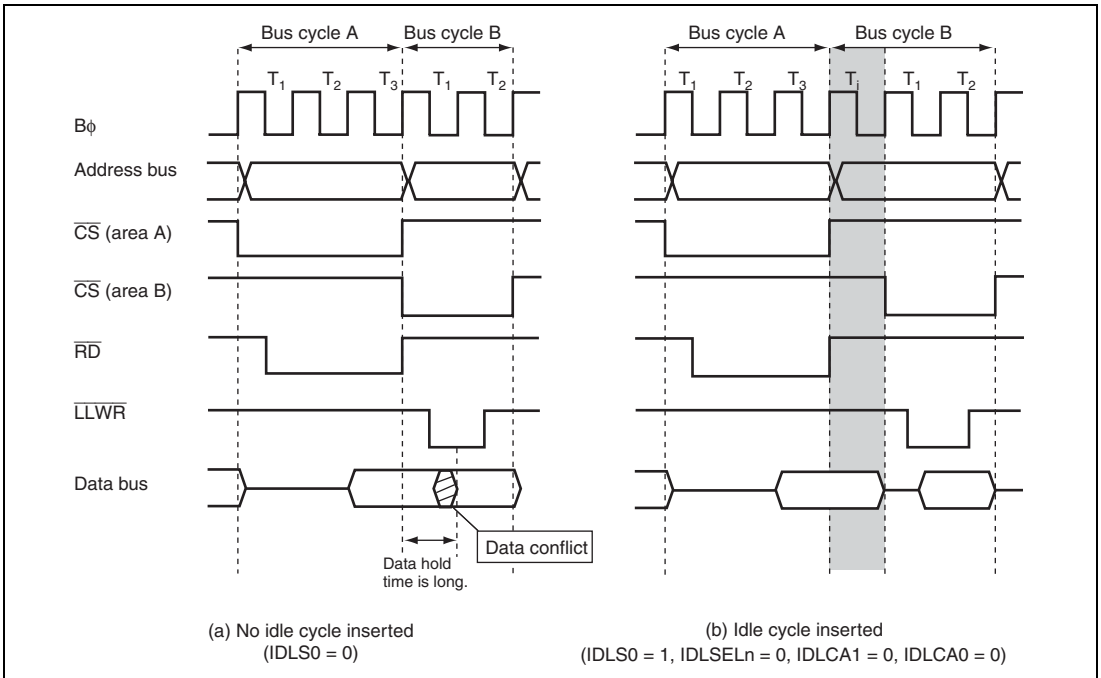


**Figure 8.37 Example of Idle Cycle Operation (Consecutive Reads in Different Areas)**

## (2) Write after Read

If an external write occurs after an external read while bit IDLS0 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 when bit IDLSELn in IDLCR is cleared to 0 when IDLSELn = 0, or bits IDLCB1 and IDLCB0 when IDLSELn is set to 1 are inserted at the start of the write cycle (n = 0 to 7).

Figure 8.38 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for 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 conflict occurs in bus cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data conflict is prevented.

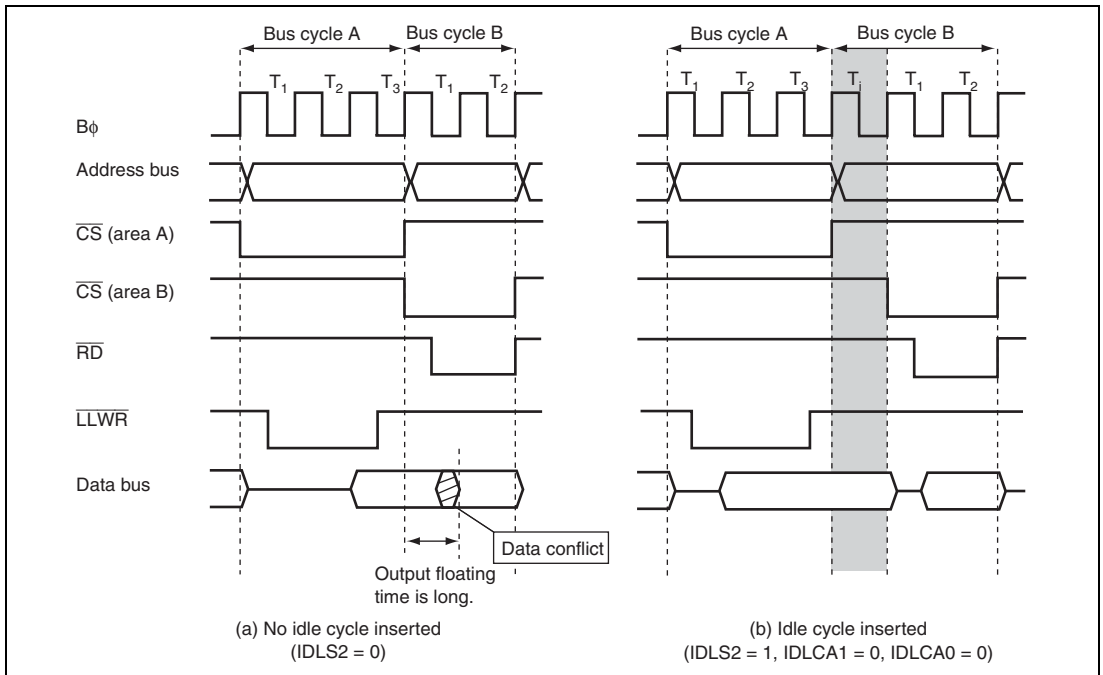


**Figure 8.38 Example of Idle Cycle Operation (Write after Read)**

### (3) Read after Write

If an external read occurs after an external write while bit IDLS2 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 are inserted at the start of the read cycle ( $n = 0$  to 7).

Figure 8.39 shows an example of the operation in this case. In this example, bus cycle A is a CPU write cycle and bus cycle B is a read cycle from the SRAM. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the CPU write data and read data from an SRAM device. In (b), an idle cycle is inserted, and a data conflict is prevented.

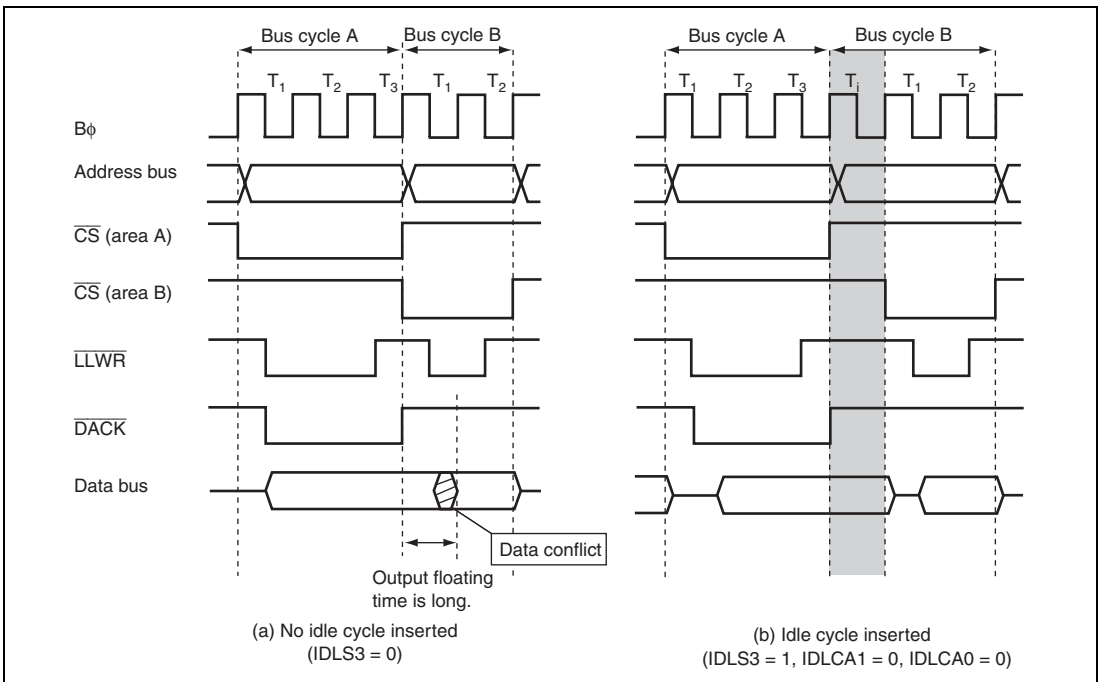


**Figure 8.39 Example of Idle Cycle Operation (Read after Write)**

#### (4) External Access after Single Address Transfer Write

If an external access occurs after a single address transfer write while bit IDLS3 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 are inserted at the start of the external access ( $n = 0$  to 7).

Figure 8.40 shows an example of the operation in this case. In this example, bus cycle A is a single address transfer (write cycle) and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the external device write data and this LSI write data. In (b), an idle cycle is inserted, and a data conflict is prevented.

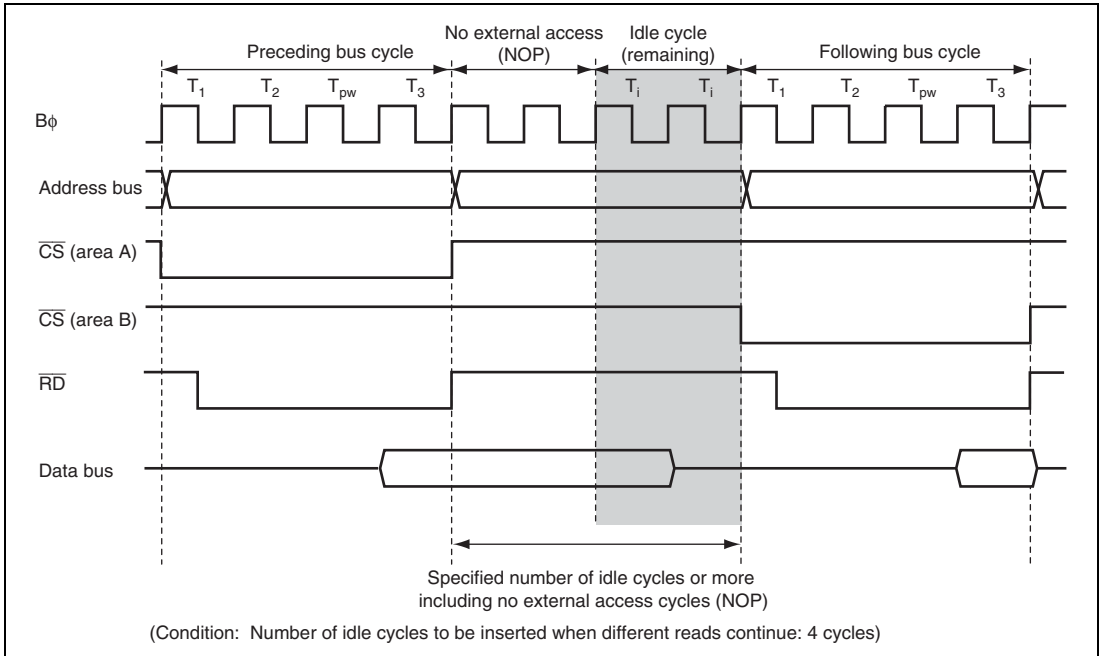


**Figure 8.40 Example of Idle Cycle Operation (Write after Single Address Transfer Write)**

### (5) External NOP Cycles and Idle Cycles

A cycle in which an external space is not accessed due to internal operations is called an external NOP cycle. Even when an external NOP cycle occurs between consecutive external bus cycles, an idle cycle can be inserted. In this case, the number of external NOP cycles is included in the number of idle cycles to be inserted.

Figure 8.41 shows an example of external NOP and idle cycle insertion.

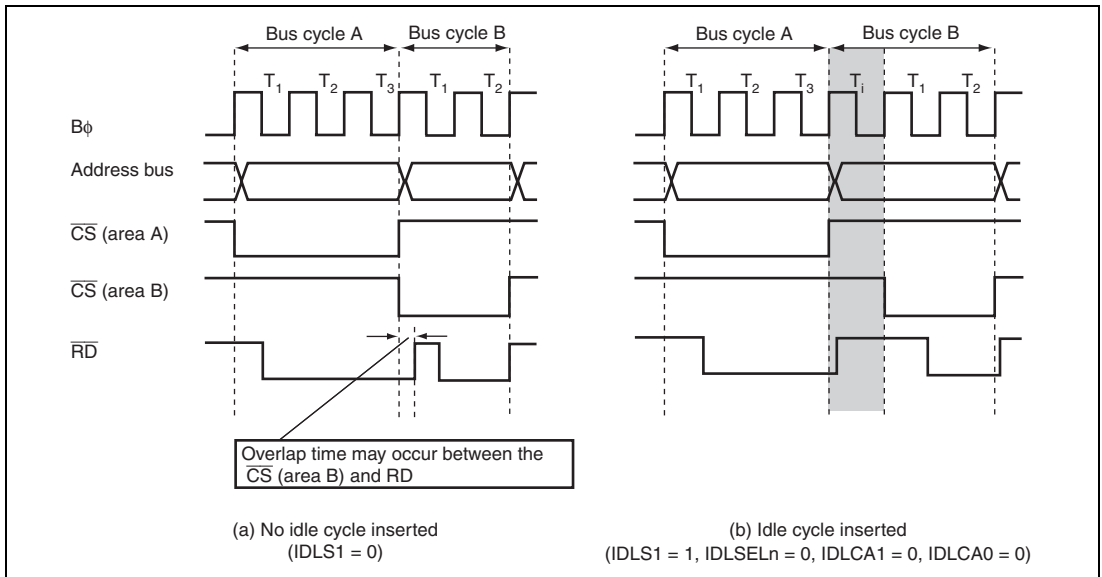


**Figure 8.41 Idle Cycle Insertion Example**



## (6) 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 8.42. In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the  $\overline{RD}$  signal in bus cycle A and the  $\overline{CS}$  signal in bus cycle B. 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 indicated in (b) is set.



**Figure 8.42 Relationship between Chip Select ( $\overline{CS}$ ) and Read ( $\overline{RD}$ )**

**Table 8.22 Idle Cycles in Mixed Accesses to Normal Space**

Previous Access	Next Access	IDLS				IDLSEL	IDLCA		IDLCB		Idle Cycle
		3	2	1	0	7 to 0	1	0	1	0	
Normal space read	Normal space read	—	—	0	—	—	—	—	—	—	Disabled
		—	—	1	—	0	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
						1	—	—	0	0	0 cycle inserted
									0	1	2 cycle inserted
									1	0	3 cycles inserted
									1	1	4 cycles inserted
Normal space read	Normal space write	—	—	—	0	—	—	—	—	—	Disabled
		—	—	—	1	0	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
						1	—	—	0	0	0 cycle inserted
									0	1	2 cycle inserted
									1	0	3 cycles inserted
									1	1	4 cycles inserted
Normal space write	Normal space read	—	0	—	—	—	—	—	—	—	Disabled
		—	1	—	—	—	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
Single address transfer write	Normal space read	0	—	—	—	—	—	—	—	—	Disabled
		1	—	—	—	—	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted

### 8.10.2 Pin States in Idle Cycle

Table 8.23 shows the pin states in an idle cycle.

**Table 8.23 Pin States in Idle Cycle**

<b>Pins</b>	<b>Pin State</b>
A20 to A0	Contents of following bus cycle
D15 to D0	High impedance
$\overline{CS}_n$ (n = 7 to 0)	High
$\overline{AS}$	High
$\overline{RD}$	High
$\overline{BS}$	High
$\overline{RD}/\overline{WR}$	High
$\overline{AH}$	Low
$\overline{LHWR}$ , $\overline{LLWR}$	High
$\overline{DACK}_n$ (n = 1 to 0)	High

## 8.11 Bus Release

This LSI can release the external bus in response to a bus request from an external device. In the external bus released state, internal bus masters continue to operate as long as there is no external access.

In addition, in the external bus released state, the  $\overline{\text{BREQO}}$  signal can be driven low to output a bus request externally.

### 8.11.1 Operation

In external extended mode, when the BRLE bit in BCR1 is set to 1 and the ICR bits for the corresponding pin are set to 1, the bus can be released to the external. Driving the  $\overline{\text{BREQ}}$  pin low issues an external bus request to this LSI. 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. For details on DDR and ICR, see section 11, I/O Ports.

In the external bus released state, the CPU, DTC, and DMAC can access the internal space using the internal bus. When the CPU, DTC, or DMAC attempts to access the external address space, it temporarily defers initiation of the bus cycle, and waits for the bus request from the external bus master to be canceled.

If the BREQOE bit in BCR1 is set to 1, the  $\overline{\text{BREQO}}$  pin can be driven low when any of the following requests are issued, to request cancellation of the bus request externally.

- When the CPU, DTC, or DMAC attempts to access the external address space
- When a SLEEP instruction is executed to place the chip in software standby mode or all-module-clock-stop mode
- When SCKCR is written to for setting the clock frequency

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

(High) External bus release > External access by CPU, DTC, or DMAC (Low)

### 8.11.2 Pin States in External Bus Released State

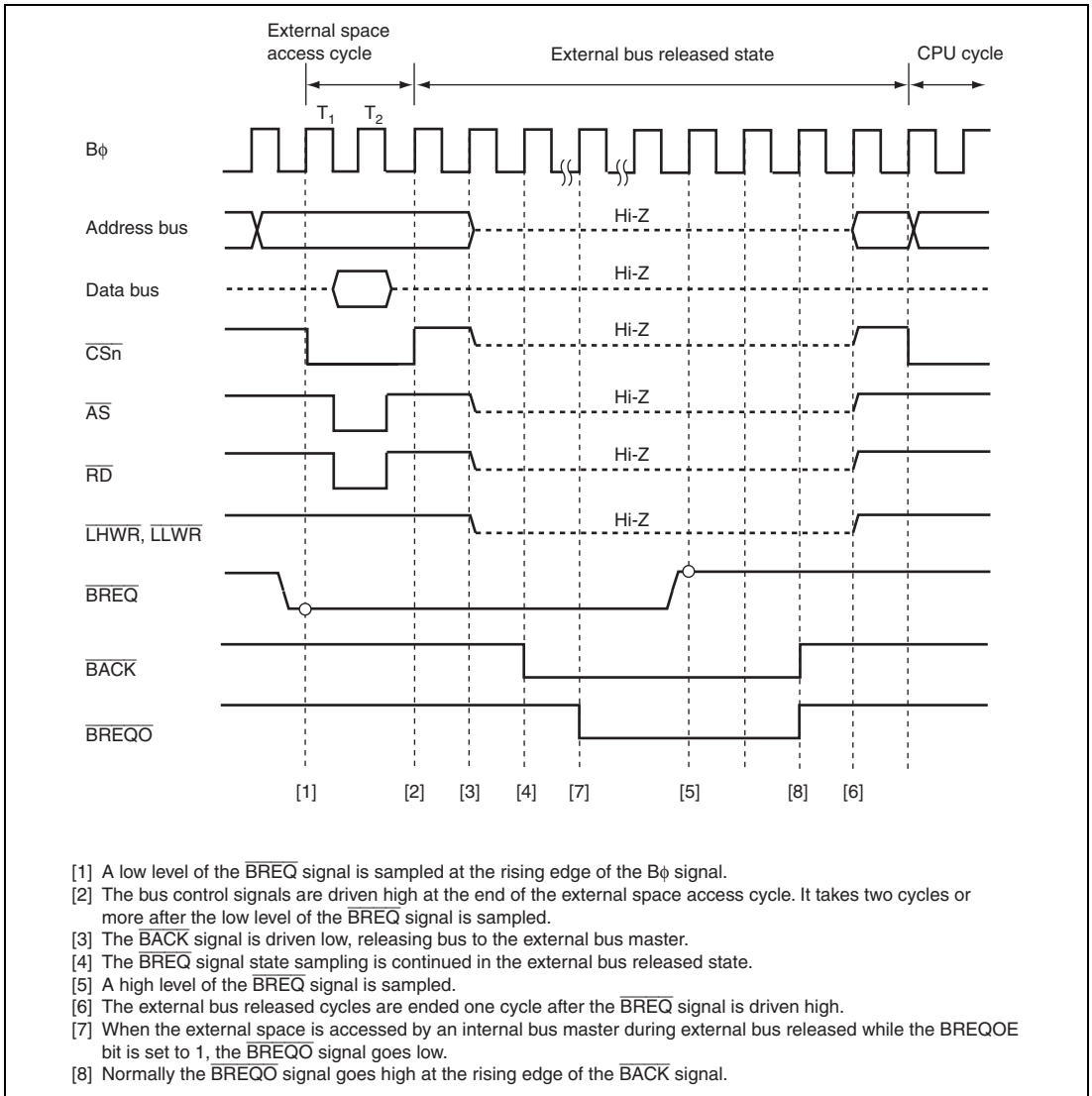
Table 8.24 shows pin states in the external bus released state.

**Table 8.24 Pin States in Bus Released State**

<b>Pins</b>	<b>Pin State</b>
A20 to A0	High impedance
D15 to D0	High impedance
$\overline{BS}$	High impedance
$\overline{CSn}$ (n = 7 to 0)	High impedance
$\overline{AS}$	High impedance
$\overline{AH}$	High impedance
$RD/\overline{WR}$	High impedance
$\overline{RD}$	High impedance
$\overline{LUB}$ , $\overline{LLB}$	High impedance
$\overline{LHWR}$ , $\overline{LLWR}$	High impedance
$\overline{DACKn}$ (n = 1 to 0)	High

### 8.11.3 Transition Timing

Figure 8.43 shows the timing for transition to the bus released state.



**Figure 8.43 Bus Released State Transition Timing**

## 8.12 Internal Bus

### 8.12.1 Access to Internal Address Space

The internal address spaces of this LSI are the on-chip ROM space, on-chip RAM space, and register space for the on-chip peripheral modules. The number of cycles necessary for access differs according to the space.

Table 8.25 shows the number of access cycles for each on-chip memory space.

**Table 8.25 Number of Access Cycles for On-Chip Memory Spaces**

Access Space	Access	Number of Access Cycles
On-chip ROM space	Read	One I $\phi$ cycle
	Write	Three I $\phi$ cycles
On-chip RAM space	Read	One I $\phi$ cycle
	Write	One I $\phi$ cycle

In access to the registers for on-chip peripheral modules, the number of access cycles differs according to the register to be accessed. When the dividing ratio of the operating clock of a bus master and that of a peripheral module is 1 : n, synchronization cycles using a clock divided by 0 to n-1 are inserted for register access in the same way as for external bus clock division.

Table 8.26 lists the number of access cycles for registers of on-chip peripheral modules.

**Table 8.26 Number of Access Cycles for Registers of On-Chip Peripheral Modules**

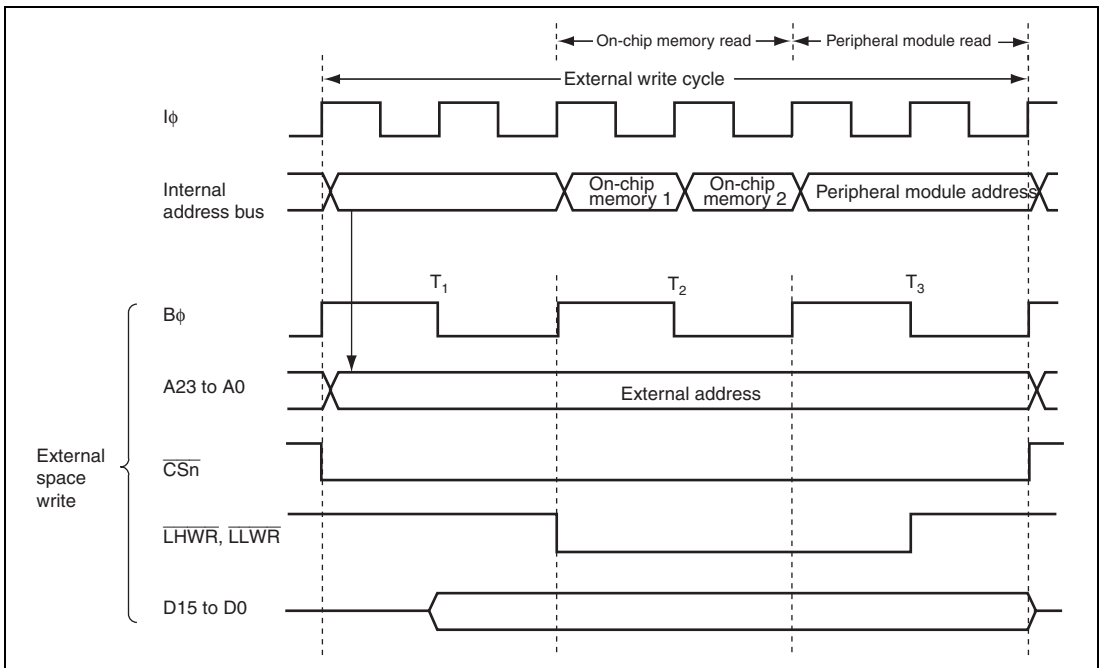
Module to be Accessed	Number of Cycles		
	Read	Write	Write Data Buffer Function
DMAC and UBC registers	Two I $\phi$		Disabled
MCU operating mode, clock pulse generator, power-down control registers, interrupt controller, bus controller, DTC registers	Two I $\phi$	Three I $\phi$	Disabled
I/O port registers of PFCR and WDT	Two P $\phi$	Three P $\phi$	Disabled
I/O port registers other than PFCR, TPU, PPG, TMR, SCI0 to SCI4, and D/A registers	Two P $\phi$		Enabled
A/D, $\Delta\Sigma$ A/D	Three P $\phi$		Enabled

## 8.13 Write Data Buffer Function

### 8.13.1 Write Data Buffer Function for External Data Bus

This LSI has a write data buffer function for the external data bus. Using the write data buffer function enables internal accesses in parallel with external writes or DMAC single address transfers. The write data buffer function is made available by setting the WDBE bit to 1 in BCR1.

Figure 8.44 shows an example of the timing when the write data buffer function is used. When this function is used, if an external address space write or a DMAC single address transfer continues for two cycles or longer, and there is an internal access next, an external write only is executed in the first two cycles. However, from the next cycle onward, internal accesses (on-chip memory or internal I/O register read/write) and the external address space write rather than waiting until it ends are executed in parallel.



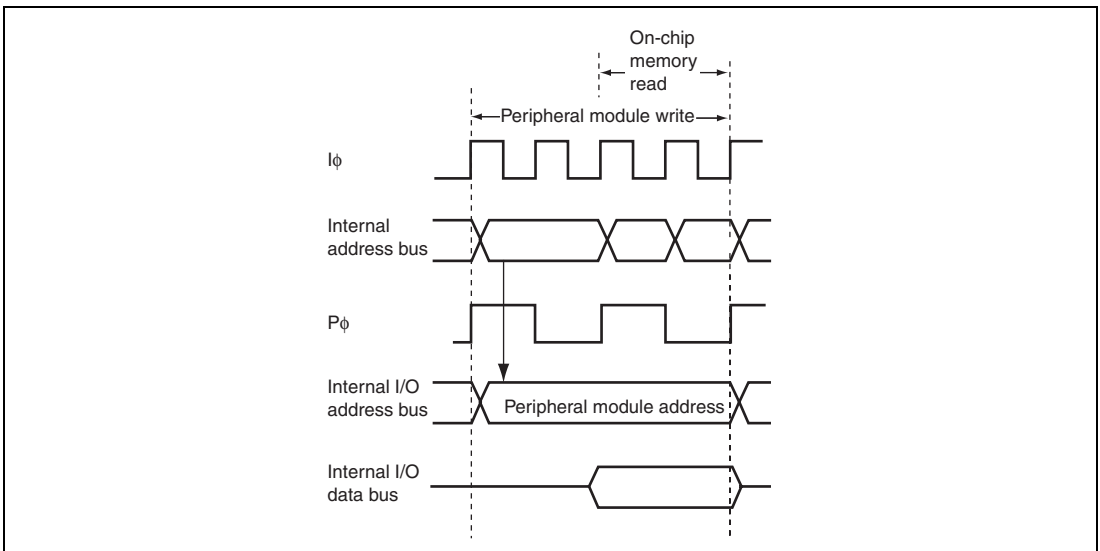
**Figure 8.44** Example of Timing when Write Data Buffer Function is Used



### 8.13.2 Write Data Buffer Function for Peripheral Modules

This LSI has a write data buffer function for the peripheral module access. Using the write data buffer function enables peripheral module writes and on-chip memory or external access to be executed in parallel. The write data buffer function is made available by setting the PWDBE bit in BCR2 to 1. For details on the on-chip peripheral module registers, see table 8.26, Number of Access Cycles for Registers of On-Chip Peripheral Modules in section 8.12, Internal Bus.

Figure 8.45 shows an example of the timing when the write data buffer function is used. When this function is used, if an internal I/O register write continues for two cycles or longer and then there is an on-chip RAM, an on-chip ROM, or an external access, internal I/O register write only is performed in the first two cycles. However, from the next cycle onward an internal memory or an external access and internal I/O register write are executed in parallel rather than waiting until it ends.



**Figure 8.45 Example of Timing when Peripheral Module Write Data Buffer Function is Used**

## 8.14 Bus Arbitration

This LSI has bus arbiters that arbitrate bus mastership operations (bus arbitration). This LSI incorporates internal access and external access bus arbiters that can be used and controlled independently. The internal bus arbiter handles the CPU, DTC, and DMAC accesses. The external bus arbiter handles the external access by the CPU, DTC, and DMAC and external bus release request (external bus master).

The bus arbiters determine priorities at the prescribed timing, and permit use of the bus by means of the bus request acknowledge signal.

### 8.14.1 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. 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 priority of the internal bus arbitration:

(High) DMAC > DTC > CPU (Low)

The priority of the external bus arbitration:

(High) External bus release request > External access by the CPU, DTC, and DMAC (Low)

If the DMAC or DTC accesses continue, the CPU can be given priority over the DMAC or DTC to execute the bus cycles alternatively between them by setting the IBCCS bit in BCR2. In this case, the priority between the DMAC and DTC does not change.

An internal bus access by the CPU, DTC, or DMAC and an external bus access by an external bus release request can be executed in parallel.

### 8.14.2 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority over that of the bus master that has taken control of the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific timings at which each bus master can release the bus.

#### (1) 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 at the end of the bus cycle. In sleep mode, the bus is transferred synchronously with the clock.

Note, however, that the bus cannot be transferred in the following cases.

- The word or longword access is performed in some divisions.
- Stack handling is performed in multiple bus cycles.
- Transfer data read or write by memory transfer instructions, block transfer instructions, or TAS instruction.

(In the block transfer instructions, the bus can be transferred in the write cycle and the following transfer data read cycle.)

- From the target read to write in the bit manipulation instructions or memory operation instructions.

(In an instruction that performs no write operation according to the instruction condition, up to a cycle corresponding the write cycle)

#### (2) DTC

The DTC sends the internal bus arbiter a request for the bus when an activation request is generated. When the DTC accesses an external bus space, the DTC first takes control of the bus from the internal bus arbiter and then requests a bus to the external bus arbiter.

Once the DTC takes control of the bus, the DTC continues the transfer processing cycles. If a bus master whose priority is higher than the DTC requests the bus, the DTC transfers the bus to the higher priority bus master. If the IBCCS bit in BCR2 is set to 1, the DTC transfers the bus to the CPU.

Note, however, that the bus cannot be transferred in the following cases.

- During transfer information read
- During the first data transfer
- During transfer information write back

The DTC releases the bus when the consecutive transfer cycles completed.

### (3) DMAC

The DMAC sends the internal bus arbiter a request for the bus when an activation request is generated. When the DMAC accesses an external bus space, the DMAC first takes control of the bus from the internal bus arbiter and then requests a bus to the external bus arbiter.

After the DMAC takes control of the bus, it may continue the transfer processing cycles or release the bus at the end of every bus cycle depending on the conditions.

The DMAC continues transfers without releasing the bus in the following case:

- Between the read cycle in the dual-address mode and the write cycle corresponding to the read cycle

If no bus master of a higher priority than the DMAC requests the bus and the IBCCS bit in BCR2 is cleared to 0, the DMAC continues transfers without releasing the bus in the following cases:

- During 1-block transfers in the block transfer mode
- During transfers in the burst mode

In other cases, the DMAC transfers the bus at the end of the bus cycle.

### (4) External Bus Release

When the  $\overline{\text{BREQ}}$  pin goes low and an external bus release request is issued while the BRLE bit in BCR1 is set to 1 with the corresponding ICR bit set to 1, a bus request is sent to the bus arbiter.

External bus release can be performed on completion of an external bus cycle.

## 8.15 Bus Controller Operation in Reset

In a reset, this LSI, including the bus controller, enters the reset state immediately, and any executing bus cycle is aborted.

## 8.16 Usage Notes

### (1) Setting Registers

The BSC registers must be specified before accessing the external address space. In on-chip ROM disabled mode, the BSC registers must be specified before accessing the external address space for other than an instruction fetch access.

### (2) External Bus Release Function and All-Module-Clock-Stop Mode

In this LSI, if the ACSE bit in MSTPCRA is set to 1, and then a SLEEP instruction is executed with the setting for all peripheral module clocks to be stopped (MSTPCRA and MSTPCRB = H'FFFFFFF) or for operation of the 8-bit timer module alone (MSTPCRA = H'[E to 0]FFFFFF), and a transition is made to the sleep state, the all-module-clock-stop mode is entered in which the clock is also stopped for the bus controller and I/O ports. For details, see section 24, Power-Down Modes.

In this state, the external bus release function is halted. To use the external bus release function in sleep mode, the ACSE bit in MSTPCR must be cleared to 0. Conversely, if a SLEEP instruction to place the chip in all-module-clock-stop mode is executed in the external bus released state, the transition to all-module-clock-stop mode is deferred and performed until after the bus is recovered.

### (3) External Bus Release Function and Software Standby

In this LSI, internal bus master operation does not stop even while the bus is released, as long as the program is running in on-chip ROM, etc., and no external access occurs. If a SLEEP instruction to place the chip in software standby mode is executed while the external bus is released, the transition to software standby mode is deferred and performed after the bus is recovered.

Also, since clock oscillation halts in software standby mode, if the  $\overline{\text{BREQ}}$  signal goes low in this mode, indicating an external bus release request, the request cannot be answered until the chip has recovered from the software standby mode.

Note that the  $\overline{\text{BACK}}$  and  $\overline{\text{BREQO}}$  pins are both in the high-impedance state in software standby mode.

#### (4) $\overline{\text{BREQO}}$ Output Timing

When the BREQOE bit is set to 1 and the  $\overline{\text{BREQO}}$  signal is output, both the  $\overline{\text{BREQO}}$  and  $\overline{\text{BACK}}$  signals may go low simultaneously.

This will occur if the next external access request occurs while internal bus arbitration is in progress after the chip samples a low level of the  $\overline{\text{BREQ}}$  signal.

## Section 9 DMA Controller (DMAC)

This LSI includes a 2-channel DMA controller (DMAC).

### 9.1 Features

- Maximum of 4-G byte address space can be accessed
- Byte, word, or longword can be set as data transfer unit
- Maximum of 4-G bytes (4,294,967,295 bytes) can be set as total transfer size  
Supports free-running mode in which total transfer size setting is not needed
- DMAC activation methods are auto-request, on-chip module interrupt, and external request.
  - Auto request: CPU activates (cycle stealing or burst access can be selected)
  - On-chip module interrupt: Interrupt requests from on-chip peripheral modules can be selected as an activation source
  - External request: Low level or falling edge detection of the  $\overline{\text{DREQ}}$  signal can be selected. External request is available for the two channels.  
In block transfer mode, low level detection is only available.
- Dual or single address mode can be selected as address mode
  - Dual address mode: Both source and destination are specified by addresses
  - Single address mode: Either source or destination is specified by the  $\overline{\text{DREQ}}$  signal and the other is specified by address
- Normal, repeat, or block transfer can be selected as transfer mode
  - Normal transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  - Repeat transfer mode: One byte, one word, or one longword data is transferred at a single transfer request  
Repeat size of data is transferred and then a transfer address returns to the transfer start address  
Up to 65536 transfers (65,536 bytes/words/longwords) can be set as repeat size
  - Block transfer mode: One block data is transferred at a single transfer request  
Up to 65,536 bytes/words/longwords can be set as block size

- Extended repeat area function which repeats the addressees within a specified area using the transfer address with the fixed upper bits (ring buffer transfer can be performed, as an example) is available

One bit (two bytes) to 27 bits (128 Mbytes) for transfer source and destination can be set as extended repeat areas

- Address update can be selected from fixed address, offset addition, and increment or decrement by 1, 2, or 4

Address update by offset addition enables to transfer data at addresses which are not placed continuously

- Word or longword data can be transferred to an address which is not aligned with the respective boundary

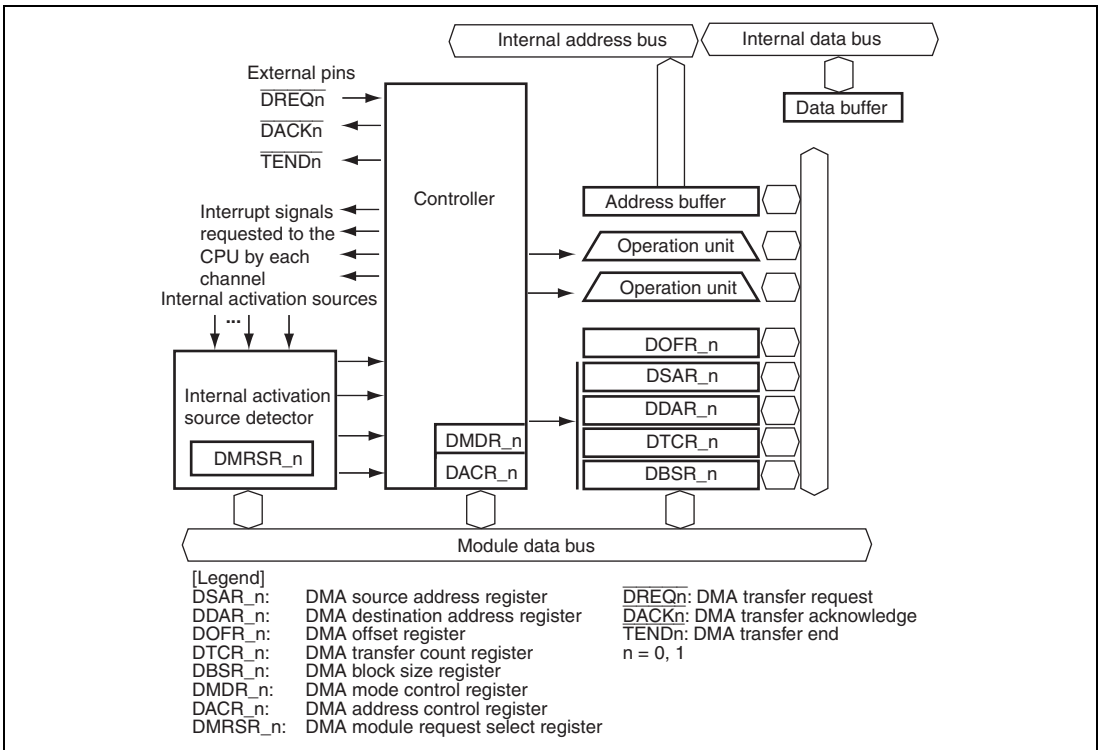
Data is divided according to its address (byte or word) when it is transferred

- Two types of interrupts can be requested to the CPU

A transfer end interrupt is generated after the number of data specified by the transfer counter is transferred. A transfer escape end interrupt is generated when the remaining total transfer size is less than the transfer data size at a single transfer request, when the repeat size of data transfer is completed, or when the extended repeat area overflows.



A block diagram of the DMAC is shown in figure 9.1.



**Figure 9.1 Block Diagram of DMAC**

## 9.2 Input/Output Pins

Table 9.1 shows the pin configuration of the DMAC.

**Table 9.1 Pin Configuration**

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

## 9.3 Register Descriptions

The DMAC has the following registers.

### Channel 0:

- DMA source address register\_0 (DSAR\_0)
- DMA destination address register\_0 (DDAR\_0)
- DMA offset register\_0 (DOFR\_0)
- DMA transfer count register\_0 (DTCR\_0)
- DMA block size register\_0 (DBSR\_0)
- DMA mode control register\_0 (DMDR\_0)
- DMA address control register\_0 (DACR\_0)
- DMA module request select register\_0 (DMRSR\_0)

### Channel 1:

- DMA source address register\_1 (DSAR\_1)
- DMA destination address register\_1 (DDAR\_1)
- DMA offset register\_1 (DOFR\_1)
- DMA transfer count register\_1 (DTCR\_1)
- DMA block size register\_1 (DBSR\_1)
- DMA mode control register\_1 (DMDR\_1)
- DMA address control register\_1 (DACR\_1)
- DMA module request select register\_1 (DMRSR\_1)

### 9.3.1 DMA Source Address Register (DSAR)

DSAR is a 32-bit readable/writable register that specifies the transfer source address. DSAR updates the transfer source address every time data is transferred. When DDAR is specified as the destination address (the DIRS bit in DACR is 1) in single address mode, DSAR is ignored.

Although DSAR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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
Bit Name								
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	7	6	5	4	3	2	1	0
Bit Name								
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

### 9.3.2 DMA Destination Address Register (DDAR)

DDAR is a 32-bit readable/writable register that specifies the transfer destination address. DDAR updates the transfer destination address every time data is transferred. When DSAR is specified as the source address (the DIRS bit in DACR is 0) in single address mode, DDAR is ignored.

Although DDAR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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
Bit Name								
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	7	6	5	4	3	2	1	0
Bit Name								
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

### 9.3.3 DMA Offset Register (DOFR)

DOFR is a 32-bit readable/writable register that specifies the offset to update the source and destination addresses. Although different values are specified for individual channels, the same values must be specified for the source and destination sides of a single channel.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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
Bit Name								
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	7	6	5	4	3	2	1	0
Bit Name								
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

### 9.3.4 DMA Transfer Count Register (DTCR)

DTCR is a 32-bit readable/writable register that specifies the size of data to be transferred (total transfer size).

To transfer 1-byte data in total, set H'00000001 in DTCR. When H'00000000 is set in this register, it means that the total transfer size is not specified and data is transferred with the transfer counter stopped (free running mode). When H'FFFFFFF is set, the total transfer size is 4 Gbytes (4,294,967,295), which is the maximum size. While data is being transferred, this register indicates the remaining transfer size. The value corresponding to its data access size is subtracted every time data is transferred (byte: -1, word: -2, and longword: -4).

Although DTCR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
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	23	22	21	20	19	18	17	16
Bit Name								
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
Bit Name								
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	7	6	5	4	3	2	1	0
Bit Name								
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

### 9.3.5 DMA Block Size Register (DBSR)

DBSR specifies the repeat size or block size. DBSR is enabled in repeat transfer mode and block transfer mode and is disabled in normal transfer mode.

Bit	31	30	29	28	27	26	25	24
Bit Name	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24
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	23	22	21	20	19	18	17	16
Bit Name	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16
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
Bit Name	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8
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	7	6	5	4	3	2	1	0
Bit Name	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
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	Bit Name	Initial Value	R/W	Description
31 to 16	BKSZH31 to BKSZH16	Undefined	R/W	Specify the repeat size or block size. When H'0001 is set, the repeat or block size is one byte, one word, or one longword. When H'0000 is set, it means the maximum value (refer to table 9.2). While the DMA is in operation, the setting is fixed.
15 to 0	BKSZ15 to BKSZ0	Undefined	R/W	Indicate the remaining repeat or block size while the DMA is in operation. The value is decremented by 1 every time data is transferred. When the remaining size becomes 0, the value of the BKSZH bits is loaded. Set the same value as the BKSZH bits.



**Table 9.2 Data Access Size, Valid Bits, and Settable Size**

Mode	Data Access Size	BKSZH Valid Bits	BKSZ Valid Bits	Settable Size (Byte)
Repeat transfer and block transfer	Byte	31 to 16	15 to 0	1 to 65,536
	Word			2 to 131,072
	Longword			4 to 262,144

### 9.3.6 DMA Mode Control Register (DMDR)

DMDR controls the DMAC operation.

- DMDR\_0

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
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	R
Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	ERRF	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/(W)*	R	R/(W)*	R/(W)*
Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Note: \* Only 0 can be written to this bit after having been read as 1, to clear the flag.

- DMDR\_1

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
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	R
Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	—	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R/(W)*	R/(W)*
Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAPO
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Note: \* Only 0 can be written to this bit after having been read as 1, to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
31	DTE	0	R/W	<p><b>Data Transfer Enable</b></p> <p>Enables/disables a data transfer for the corresponding channel. When this bit is set to 1, it indicates that the DMAC is in operation.</p> <p>Setting this bit to 1 starts a transfer when the auto-request is selected. When the on-chip module interrupt or external request is selected, a transfer request after setting this bit to 1 starts the transfer. While data is being transferred, clearing this bit to 0 stops the transfer.</p> <p>In block transfer mode, if writing 0 to this bit while data is being transferred, this bit is cleared to 0 after the current 1-block size data transfer.</p> <p>If an event which stops (sustains) a transfer occurs externally, this bit is automatically cleared to 0 to stop the transfer.</p> <p>Operating modes and transfer methods must not be changed while this bit is set to 1.</p> <p>0: Disables a data transfer 1: Enables a data transfer (DMA is in operation)</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When the specified total transfer size of transfers is completed</li> <li>• When a transfer is stopped by an overflow interrupt by a repeat size end</li> <li>• When a transfer is stopped by an overflow interrupt by an extended repeat size end</li> <li>• When a transfer is stopped by a transfer size error interrupt</li> <li>• When clearing this bit to 0 to stop a transfer</li> </ul> <p>In block transfer mode, this bit changes after the current block transfer.</p> <ul style="list-style-type: none"> <li>• When an address error or an NMI interrupt is requested</li> <li>• In the reset state or hardware standby mode</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
30	DACKE	0	R/W	<p><math>\overline{\text{DACK}}</math> Signal Output Enable</p> <p>Enables/disables the <math>\overline{\text{DACK}}</math> signal output in single address mode. This bit is ignored in dual address mode.</p> <p>0: Enables <math>\overline{\text{DACK}}</math> signal output</p> <p>1: Disables <math>\overline{\text{DACK}}</math> signal output</p>
29	TENDE	0	R/W	<p><math>\overline{\text{TEND}}</math> Signal Output Enable</p> <p>Enables/disables the <math>\overline{\text{TEND}}</math> signal output.</p> <p>0: Enables <math>\overline{\text{TEND}}</math> signal output</p> <p>1: Disables <math>\overline{\text{TEND}}</math> signal output</p>
28	—	0	R/W	<p>Reserved</p> <p>Initial value should not be changed.</p>
27	DREQS	0	R/W	<p><math>\overline{\text{DREQ}}</math> Select</p> <p>Selects whether a low level or the falling edge of the <math>\overline{\text{DREQ}}</math> signal used in external request mode is detected. When a block transfer is performed in external request mode, clear this bit to 0.</p> <p>0: Low level detection</p> <p>1: Falling edge detection (the first transfer after a transfer enabled is detected on a low level)</p>
26	NRD	0	R/W	<p>Next Request Delay</p> <p>Selects the accepting timing of the next transfer request.</p> <p>0: Starts accepting the next transfer request after completion of the current transfer</p> <p>1: Starts accepting the next transfer request one cycle after completion of the current transfer</p>
25, 24	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>
23	ACT	0	R	<p>Active State</p> <p>Indicates the operating state for the channel.</p> <p>0: Waiting for a transfer request or a transfer disabled state by clearing the DTE bit to 0</p> <p>1: Active state</p>
22 to 20	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>

Bit	Bit Name	Initial Value	R/W	Description
19	ERRF	0	R/(W)*	<p>System Error Flag</p> <p>Indicates that an address error or an NMI interrupt has been generated. This bit is available only in DMDR_0. Setting this bit to 1 prohibits writing to the DTE bit for all the channels. This bit is reserved in DMDR_1 to DMDR_3. It is always read as 0 and cannot be modified.</p> <p>0: An address error or an NMI interrupt has not been generated</p> <p>1: An address error or an NMI interrupt has been generated</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When clearing to 0 after reading ERRF = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When an address error or an NMI interrupt has been generated</li> </ul> <p>However, when an address error or an NMI interrupt has been generated in DMAC module stop state, this bit is not set to 1.</p>
18	—	0	R	<p>Reserved</p> <p>This bit is always read as 0 and cannot be modified.</p>
17	ESIF	0	R/(W)*	<p>Transfer Escape Interrupt Flag</p> <p>Indicates that a transfer escape end interrupt has been requested. A transfer escape end means that a transfer is terminated before the transfer counter reaches 0.</p> <p>0: A transfer escape end interrupt has not been requested</p> <p>1: A transfer escape end interrupt has been requested</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When setting the DTE bit to 1</li> <li>When clearing to 0 before reading ESIF = 1</li> </ul> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>When a transfer size error interrupt is requested</li> <li>When a repeat size end interrupt is requested</li> <li>When a transfer end interrupt by an extended repeat area overflow is requested</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
16	DTIF	0	R/(W)*	<p>Data Transfer Interrupt Flag</p> <p>Indicates that a transfer end interrupt by the transfer counter has been requested.</p> <p>0: A transfer end interrupt by the transfer counter has not been requested</p> <p>1: A transfer end interrupt by the transfer counter has been requested</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When setting the DTE bit to 1</li> <li>• When clearing to 0 after reading DTIF = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When DTCR reaches 0 and the transfer is completed</li> </ul>
15	DTSZ1	0	R/W	Data Access Size 1 and 0
14	DTSZ0	0	R/W	<p>Select the data access size for a transfer.</p> <p>00: Byte size (eight bits)</p> <p>01: Word size (16 bits)</p> <p>10: Longword size (32 bits)</p> <p>11: Setting prohibited</p>
13	MDS1	0	R/W	Transfer Mode Select 1 and 0
12	MDS0	0	R/W	<p>Select the transfer mode.</p> <p>00: Normal transfer mode</p> <p>01: Block transfer mode</p> <p>10: Repeat transfer mode</p> <p>11: Setting prohibited</p>

Bit	Bit Name	Initial Value	R/W	Description
11	TSEIE	0	R/W	<p>Transfer Size Error Interrupt Enable</p> <p>Enables/disables a transfer size error interrupt.</p> <p>When the next transfer is requested while this bit is set to 1 and the contents of the transfer counter is less than the size of data to be transferred at a single transfer request, the DTE bit is cleared to 0. At this time, the ESIF bit is set to 1 to indicate that a transfer size error interrupt has been requested.</p> <p>The sources of a transfer size error are as follows:</p> <ul style="list-style-type: none"> <li>• In normal or repeat transfer mode, the total transfer size set in DTCR is less than the data access size</li> <li>• In block transfer mode, the total transfer size set in DTCR is less than the block size</li> </ul> <p>0: Disables a transfer size error interrupt request 1: Enables a transfer size error interrupt request</p>
10	—	0	R	<p>Reserved</p> <p>This bit is always read as 0 and cannot be modified.</p>
9	ESIE	0	R/W	<p>Transfer Escape Interrupt Enable</p> <p>Enables/disables a transfer escape end interrupt request. When the ESIF bit is set to 1 with this bit set to 1, a transfer escape end interrupt is requested to the CPU or DTC. The transfer end interrupt request is cleared by clearing this bit or the ESIF bit to 0.</p> <p>0: Disables a transfer escape end interrupt 1: Enables a transfer escape end interrupt</p>
8	DTIE	0	R/W	<p>Data Transfer Interrupt Enable</p> <p>Enables/disables a transfer end interrupt request by the transfer counter. When the DTIF bit is set to 1 with this bit set to 1, a transfer end interrupt is requested to the CPU or DTC. The transfer end interrupt request is cleared by clearing this bit or the DTIF bit to 0.</p> <p>0: Disables a transfer end interrupt 1: Enables a transfer end interrupt</p>

Bit	Bit Name	Initial Value	R/W	Description
7	DTF1	0	R/W	Data Transfer Factor 1 and 0
6	DTF0	0	R/W	Select a DMAC activation source. When the on-chip peripheral module setting is selected, the interrupt source should be selected by DMRSR. When the external request setting is selected, the sampling method should be selected by the DREQS bit.  00: Auto request (cycle stealing) 01: Auto request (burst access) 10: On-chip module interrupt 11: External request
5	DTA	0	R/W	Data Transfer Acknowledge  This bit is valid in DMA transfer by the on-chip module interrupt source. This bit enables or disables to clear the source flag selected by DMRSR.  0: To clear the source in DMA transfer is disabled. Since the on-chip module interrupt source is not cleared in DMA transfer, it should be cleared by the CPU or DTC transfer.  1: To clear the source in DMA transfer is enabled. Since the on-chip module interrupt source is cleared in DMA transfer, it does not require an interrupt by the CPU or DTC transfer.
4, 3	—	All 0	R	Reserved  These bits are always read as 0 and cannot be modified.



Bit	Bit Name	Initial Value	R/W	Description
2	DMAP2	0	R/W	DMA Priority Level 2 to 0
1	DMAP1	0	R/W	Select the priority level of the DMAC when using the CPU priority control function over DTC and DMAC.
0	DMAP0	0	R/W	When the CPU has priority over the DMAC, the DMAC masks a transfer request and waits for the timing when the CPU priority becomes lower than the DMAC priority. The priority levels can be set to the individual channels. This bit is valid when the CPUPCE bit in CPUPCR is set to 1.  000: Priority level 0 (low) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (high)

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

### 9.3.7 DMA Address Control Register (DACR)

DACR specifies the operating mode and transfer method.

Bit	31	30	29	28	27	26	25	24
Bit Name	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R	R/W	R/W	R/W
Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	SAT1	SAT0	—	—	DAT1	DAT0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W
Bit	15	14	13	12	11	10	9	8
Bit Name	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31	AMS	0	R/W	Address Mode Select Selects address mode from single or dual address mode. In single address mode, the $\overline{\text{DACK}}$ pin is enabled according to the DACKC bit. 0: Dual address mode 1: Single address mode
30	DIRS	0	R/W	Single Address Direction Select Specifies the data transfer direction in single address mode. This bit is ignored in dual address mode. 0: Specifies DSAR as source address 1: Specifies DDAR as destination address
29 to 27	—	0	R/W	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
26	RPTIE	0	R/W	<p>Repeat Size End Interrupt Enable</p> <p>Enables/disables a repeat size end interrupt request.</p> <p>In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat-size data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested. Even when the repeat area is not specified (ARS1 = 1 and ARS0 = 0), a repeat size end interrupt after a 1-block data transfer can be requested.</p> <p>In addition, in block transfer mode, when the next transfer is requested after 1-block data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested.</p> <p>0: Disables a repeat size end interrupt 1: Enables a repeat size end interrupt</p>
25	ARS1	0	R/W	Area Select 1 and 0
24	ARS0	0	R/W	<p>Specify the block area or repeat area in block or repeat transfer mode.</p> <p>00: Specify the block area or repeat area on the source address 01: Specify the block area or repeat area on the destination address 10: Do not specify the block area or repeat area 11: Setting prohibited</p>
23, 22	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>
21	SAT1	0	R/W	Source Address Update Mode 1 and 0
20	SAT0	0	R/W	<p>Select the update method of the source address (DSAR). When DSAR is not specified as the transfer source in single address mode, this bit is ignored.</p> <p>00: Source address is fixed 01: Source address is updated by adding the offset 10: Source address is updated by adding 1, 2, or 4 according to the data access size 11: Source address is updated by subtracting 1, 2, or 4 according to the data access size</p>

Bit	Bit Name	Initial Value	R/W	Description
19, 18	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
17	DAT1	0	R/W	Destination Address Update Mode 1 and 0
16	DAT0	0	R/W	Select the update method of the destination address (DDAR). When DDAR is not specified as the transfer destination in single address mode, this bit is ignored. 00: Destination address is fixed 01: Destination address is updated by adding the offset 10: Destination address is updated by adding 1, 2, or 4 according to the data access size 11: Destination address is updated by subtracting 1, 2, or 4 according to the data access size
15	SARIE	0	R/W	Interrupt Enable for Source Address Extended Area Overflow Enables/disables an interrupt request for an extended area overflow on the source address. When an extended repeat area overflow on the source address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the source address is requested. When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which a transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped. When the extended repeat area is not specified, this bit is ignored. 0: Disables an interrupt request for an extended area overflow on the source address 1: Enables an interrupt request for an extended area overflow on the source address
14, 13	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
12	SARA4	0	R/W	Source Address Extended Repeat Area
11	SARA3	0	R/W	Specify the extended repeat area on the source address (DSAR). With the extended repeat area, the specified lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2.  When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively.  When an overflow in the extended repeat area occurs with the SARIE bit set to 1, an interrupt can be requested. Table 9.3 shows the settings and areas of the extended repeat area.
10	SARA2	0	R/W	
9	SARA1	0	R/W	
8	SARA0	0	R/W	
7	DARIE	0	R/W	Destination Address Extended Repeat Area Overflow Interrupt Enable  Enables/disables an interrupt request for an extended area overflow on the destination address.  When an extended repeat area overflow on the destination address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the destination address is requested.  When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which the transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped.  When the extended repeat area is not specified, this bit is ignored.  0: Disables an interrupt request for an extended area overflow on the destination address 1: Enables an interrupt request for an extended area overflow on the destination address
6, 5	—	All 0	R	Reserved  These bits are always read as 0 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
4	DARA4	0	R/W	Destination Address Extended Repeat Area
3	DARA3	0	R/W	Specify the extended repeat area on the destination address (DDAR). With the extended repeat area, the specified lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2.  When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively.  When an overflow in the extended repeat area occurs with the DARIE bit set to 1, an interrupt can be requested. Table 9.3 shows the settings and areas of the extended repeat area.
2	DARA2	0	R/W	
1	DARA1	0	R/W	
0	DARA0	0	R/W	

**Table 9.3 Settings and Areas of Extended Repeat Area**

<b>SARA4 to SARA0 or DARA4 to DARA0</b>	<b>Extended Repeat Area</b>
00000	Not specified
00001	2 bytes specified as extended repeat area by the lower 1 bit of the address
00010	4 bytes specified as extended repeat area by the lower 2 bits of the address
00011	8 bytes specified as extended repeat area by the lower 3 bits of the address
00100	16 bytes specified as extended repeat area by the lower 4 bits of the address
00101	32 bytes specified as extended repeat area by the lower 5 bits of the address
00110	64 bytes specified as extended repeat area by the lower 6 bits of the address
00111	128 bytes specified as extended repeat area by the lower 7 bits of the address
01000	256 bytes specified as extended repeat area by the lower 8 bits of the address
01001	512 bytes specified as extended repeat area by the lower 9 bits of the address
01010	1 kbyte specified as extended repeat area by the lower 10 bits of the address
01011	2 kbytes specified as extended repeat area by the lower 11 bits of the address
01100	4 kbytes specified as extended repeat area by the lower 12 bits of the address
01101	8 kbytes specified as extended repeat area by the lower 13 bits of the address
01110	16 kbytes specified as extended repeat area by the lower 14 bits of the address
01111	32 kbytes specified as extended repeat area by the lower 15 bits of the address
10000	64 kbytes specified as extended repeat area by the lower 16 bits of the address
10001	128 kbytes specified as extended repeat area by the lower 17 bits of the address
10010	256 kbytes specified as extended repeat area by the lower 18 bits of the address
10011	512 kbytes specified as extended repeat area by the lower 19 bits of the address
10100	1 Mbyte specified as extended repeat area by the lower 20 bits of the address
10101	2 Mbytes specified as extended repeat area by the lower 21 bits of the address
10110	4 Mbytes specified as extended repeat area by the lower 22 bits of the address
10111	8 Mbytes specified as extended repeat area by the lower 23 bits of the address
11000	16 Mbytes specified as extended repeat area by the lower 24 bits of the address
11001	32 Mbytes specified as extended repeat area by the lower 25 bits of the address
11010	64 Mbytes specified as extended repeat area by the lower 26 bits of the address
11011	128 Mbytes specified as extended repeat area by the lower 27 bits of the address
111××	Setting prohibited

[Legend]

×: Don't care

### 9.3.8 DMA Module Request Select Register (DMRSR)

DMRSR is an 8-bit readable/writable register that specifies the on-chip module interrupt source. The vector number of the interrupt source is specified in eight bits. However, 0 is regarded as no interrupt source. For the vector numbers of the interrupt sources, refer to table 9.5.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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



## 9.4 Transfer Modes

Table 9.4 shows the DMAC transfer modes. The transfer modes can be specified to the individual channels.

**Table 9.4 Transfer Modes**

Address Mode	Transfer mode	Activation Source	Common Function	Address Register	
				Source	Destination
Dual address	<ul style="list-style-type: none"> <li>Normal transfer</li> <li>Repeat transfer</li> <li>Block transfer</li> </ul> Repeat or block size = 1 to 65,536 bytes, 1 to 65,536 words, or 1 to 65,536 longwords	<ul style="list-style-type: none"> <li>Auto request (activated by CPU)</li> <li>On-chip module interrupt</li> <li>External request</li> </ul>	<ul style="list-style-type: none"> <li>Total transfer size: 1 to 4 Gbytes or not specified</li> <li>Offset addition</li> <li>Extended repeat area function</li> </ul>	DSAR	DDAR
Single address	<ul style="list-style-type: none"> <li>Instead of specifying the source or destination address registers, data is directly transferred from/to the external device using the <math>\overline{DACK}</math> pin</li> <li>The same settings as above are available other than address register setting (e.g., above transfer modes can be specified)</li> <li>One transfer can be performed in one bus cycle (the types of transfer modes are the same as those of dual address modes)</li> </ul>			$\overline{DSAR}/\overline{DACK}$	$\overline{DACK}/\overline{DDAR}$

When the auto request setting is selected as the activation source, the cycle stealing or burst access can be selected. When the total transfer size is not specified (DTCR = H'00000000), the transfer counter is stopped and the transfer is continued without the limitation of the transfer count.

## 9.5 Operations

### 9.5.1 Address Modes

#### (1) Dual Address Mode

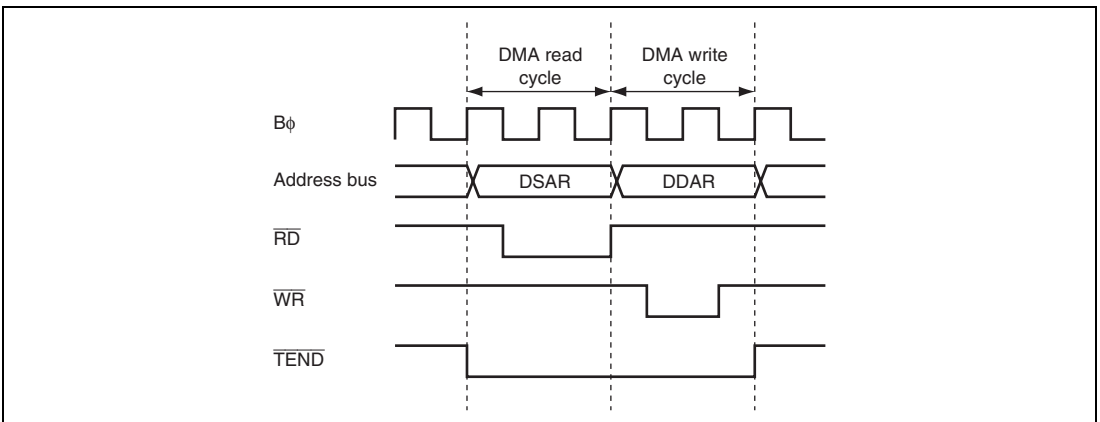
In dual address mode, the transfer source address is specified in DSAR and the transfer destination address is specified in DDAR. A transfer at a time is performed in two bus cycles (when the data bus width is less than the data access size or the access address is not aligned with the boundary of the data access size, the number of bus cycles are needed more than two because one bus cycle is divided into multiple bus cycles).

In the first bus cycle, data at the transfer source address is read and in the next cycle, the read data is written to the transfer destination address.

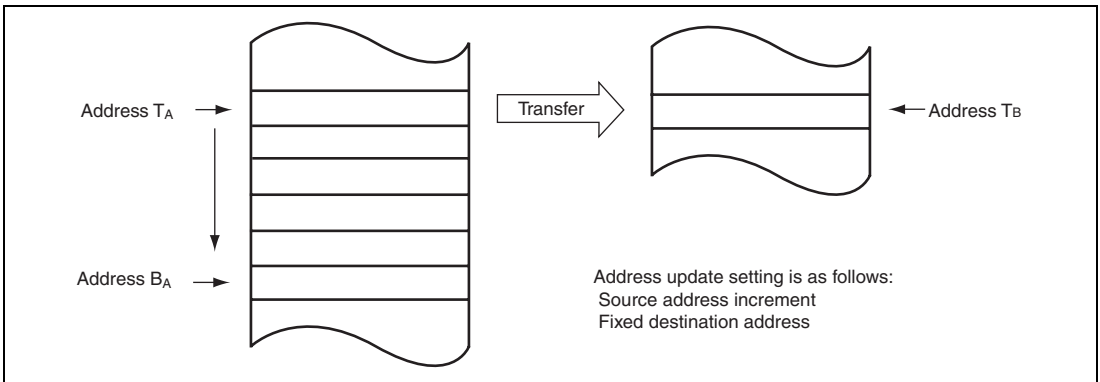
The read and write cycles are not separated. Other bus cycles (bus cycle by other bus masters, refresh cycle, and external bus release cycle) are not generated between read and write cycles.

The  $\overline{TEND}$  signal output is enabled or disabled by the TEND bit in DMDR. The  $\overline{TEND}$  signal is output in two bus cycles. When an idle cycle is inserted before the bus cycle, the  $\overline{TEND}$  signal is also output in the idle cycle. The  $\overline{DACK}$  signal is not output.

Figure 9.2 shows an example of the signal timing in dual address mode and figure 9.3 shows the operation in dual address mode.



**Figure 9.2 Example of Signal Timing in Dual Address Mode**



**Figure 9.3 Operations in Dual Address Mode**

## (2) Single Address Mode

In single address mode, data between an external device and an external memory is directly transferred using the  $\overline{DACK}$  pin instead of DSAR or DDAR. A transfer at a time is performed in one bus cycle. In this mode, the data bus width must be the same as the data access size. For details on the data bus width, see section 8, Bus Controller (BSC).

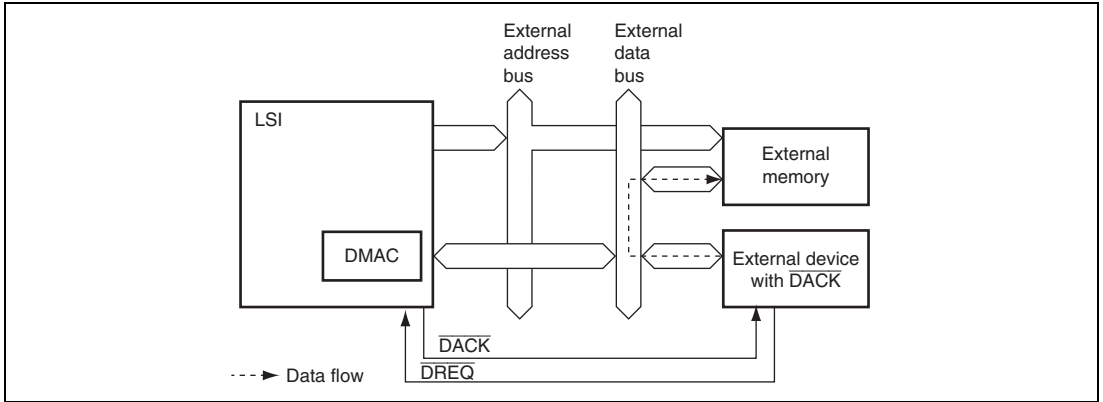
The DMAC accesses an external device as the transfer source or destination by outputting the strobe signal ( $\overline{DACK}$ ) to the external device with  $\overline{DACK}$  and accesses the other transfer target by outputting the address. Accordingly, the DMA transfer is performed in one bus cycle. Figure 9.4 shows an example of a transfer between an external memory and an external device with the  $\overline{DACK}$  pin. In this example, the external device outputs data on the data bus and the data is written to the external memory in the same bus cycle.

The transfer direction is decided by the DIRS bit in DACR which specifies an external device with the  $\overline{DACK}$  pin as the transfer source or destination. When DIRS = 0, data is transferred from an external memory (DSAR) to an external device with the  $\overline{DACK}$  pin. When DIRS = 1, data is transferred from an external device with the  $\overline{DACK}$  pin to an external memory (DDAR). The settings of registers which are not used as the transfer source or destination are ignored.

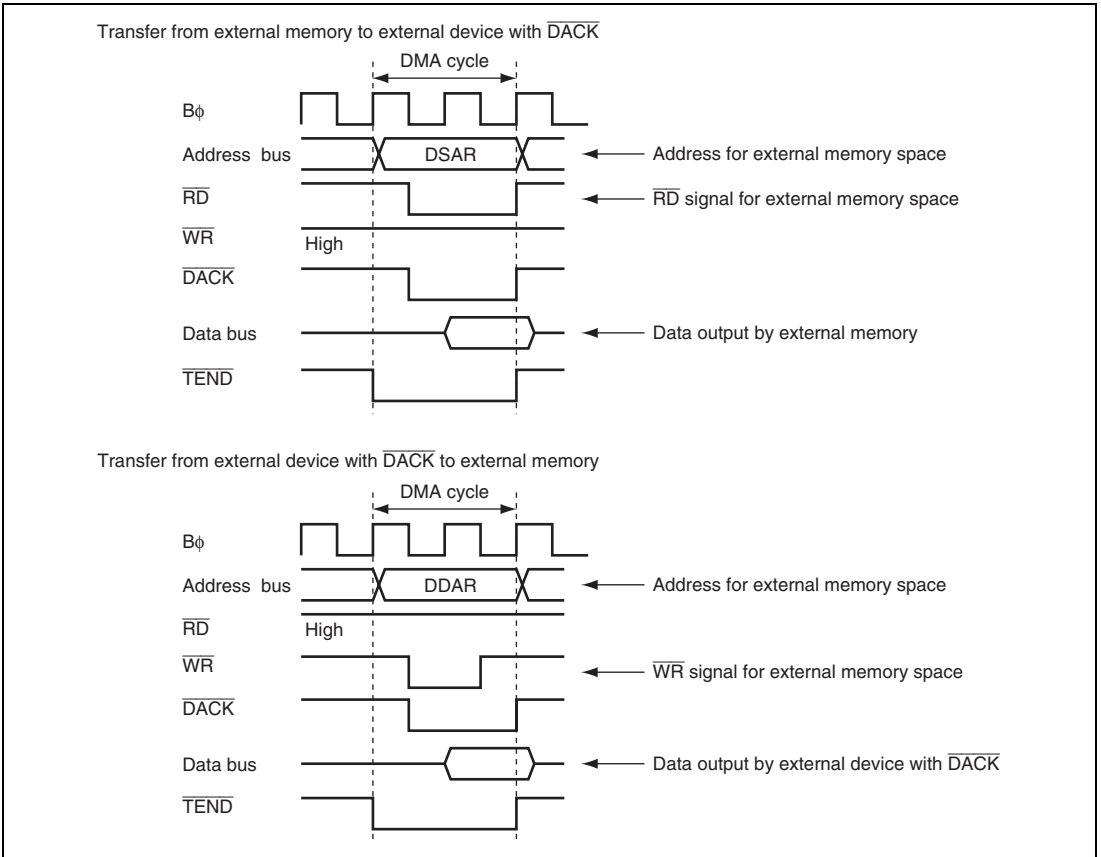
The  $\overline{DACK}$  signal output is enabled in single address mode by the DACKE bit in DMDR. The  $\overline{DACK}$  signal is low active.

The  $\overline{TEND}$  signal output is enabled or disabled by the TENDE bit in DMDR. The  $\overline{TEND}$  signal is output in one bus cycle. When an idle cycle is inserted before the bus cycle, the  $\overline{TEND}$  signal is also output in the idle cycle.

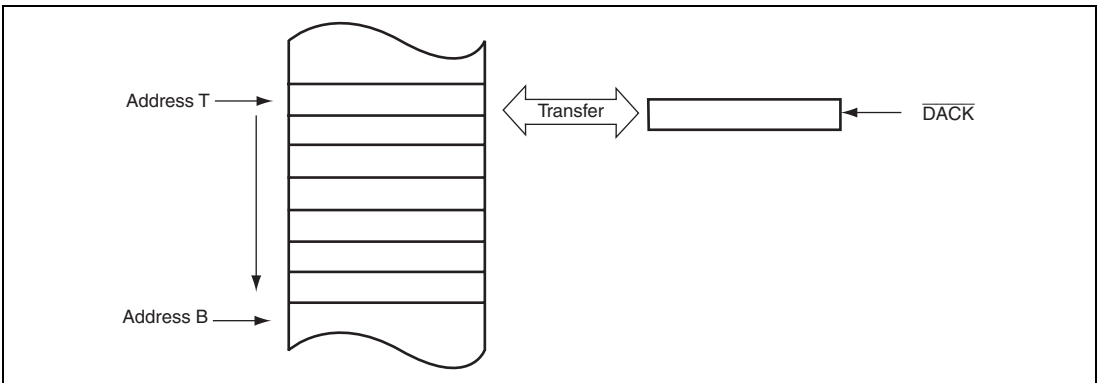
Figure 9.5 shows an example of timing charts in single address mode and figure 9.6 shows an example of operation in single address mode.



**Figure 9.4 Data Flow in Single Address Mode**



**Figure 9.5 Example of Signal Timing in Single Address Mode**



**Figure 9.6 Operations in Single Address Mode**

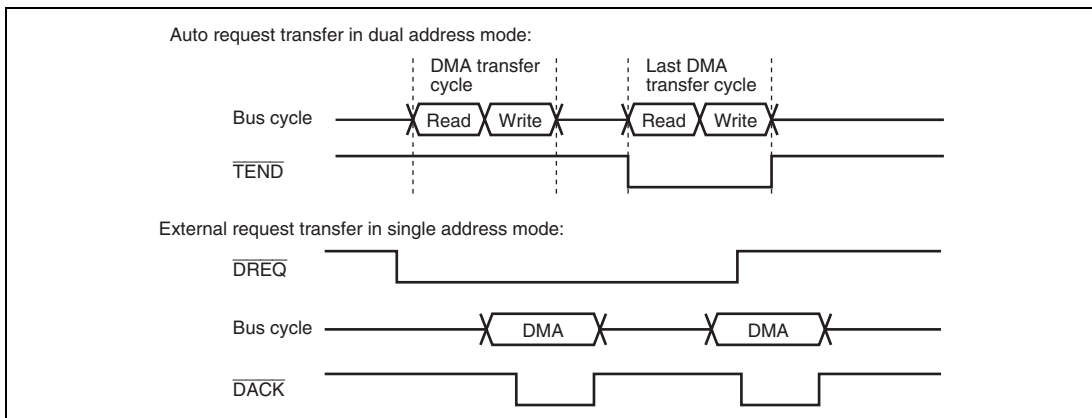
## 9.5.2 Transfer Modes

### (1) Normal Transfer Mode

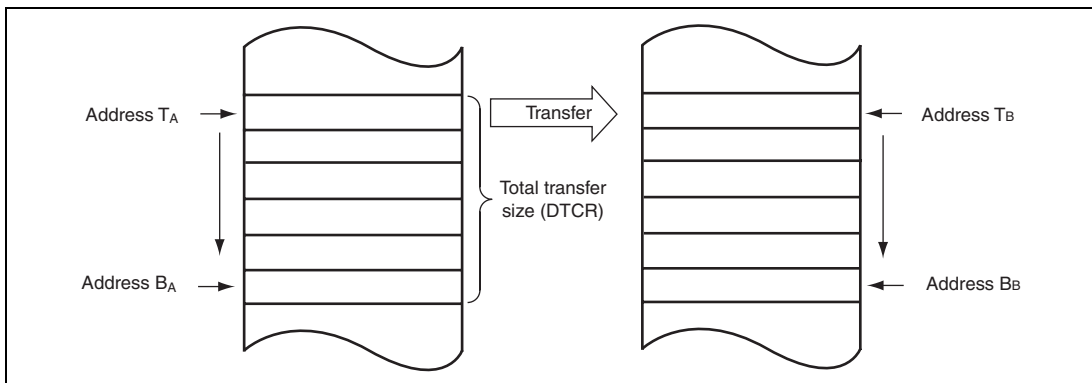
In normal transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. DBSR is ignored in normal transfer mode.

The  $\overline{TEND}$  signal is output only in the last DMA transfer. The  $\overline{DACK}$  signal is output every time a transfer request is received and a transfer starts.

Figure 9.7 shows an example of the signal timing in normal transfer mode and figure 9.8 shows the operation in normal transfer mode.



**Figure 9.7 Example of Signal Timing in Normal Transfer Mode**



**Figure 9.8 Operations in Normal Transfer Mode**

## (2) Repeat Transfer Mode

In repeat transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. The repeat size can be specified in DBSR up to  $65536 \times$  data access size.

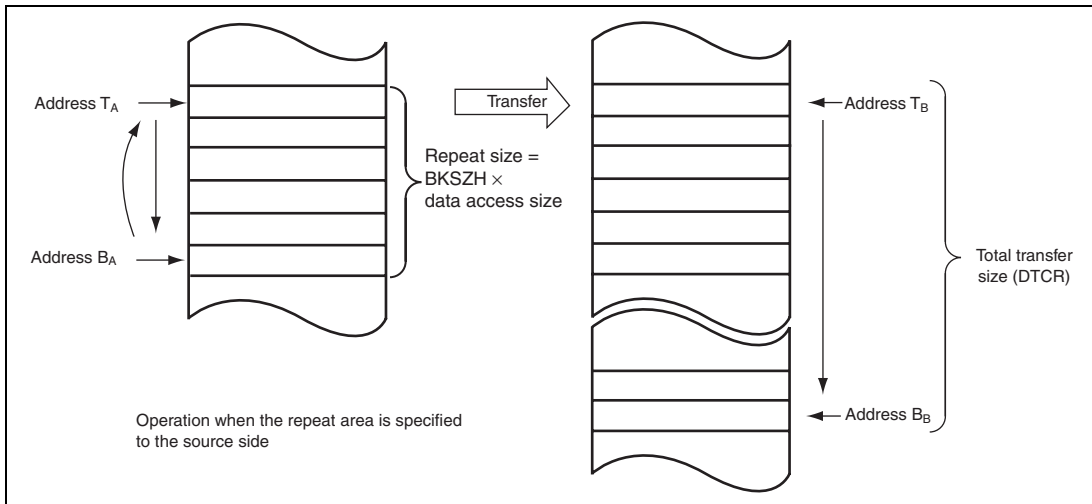
The repeat area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the repeat area returns to the transfer start address when the repeat size of transfers is completed. This operation is repeated until the total transfer size specified in DTCR is completed. When H'00000000 is specified in DTCR, it is regarded as the free running mode and repeat transfer is continued until the DTE bit in DMDR is cleared to 0.

In addition, a DMA transfer can be stopped and a repeat size end interrupt can be requested to the CPU or DTC when the repeat size of transfers is completed. When the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit is set to 1, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1 to complete the transfer. At this time, an interrupt is requested to the CPU or DTC when the ESIE bit in DMDR is set to 1.

The timings of the  $\overline{TEND}$  and  $\overline{DACK}$  signals are the same as in normal transfer mode.

Figure 9.9 shows the operation in repeat transfer mode while dual address mode is set.

When the repeat area is specified as neither source nor destination address side, the operation is the same as the normal transfer mode operation shown in figure 9.8. In this case, a repeat size end interrupt can also be requested to the CPU when the repeat size of transfers is completed.



**Figure 9.9 Operations in Repeat Transfer Mode**

### (3) Block Transfer Mode

In block transfer mode, one block size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as total transfer size by DTCR. The block size can be specified in DBSR up to  $65536 \times$  data access size.

While one block of data is being transferred, transfer requests from other channels are suspended. When the transfer is completed, the bus is released to the other bus master.

The block area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the block area returns to the transfer start address when the block size of data is completed. When the block area is specified as neither source nor destination address side, the operation continues without returning the address to the transfer start address. A repeat size end interrupt can be requested.

The  $\overline{TEND}$  signal is output every time 1-block data is transferred in the last DMA transfer cycle. When the external request is selected as an activation source, the low level detection of the  $\overline{DREQ}$  signal ( $DREQS = 0$ ) should be selected.

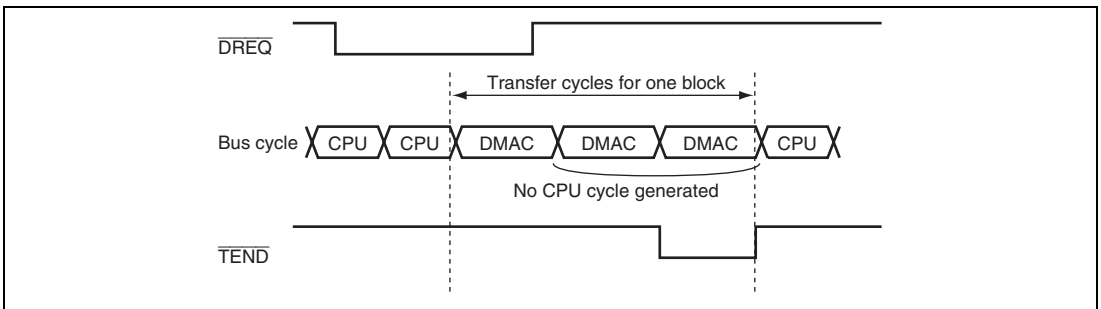
When an interrupt request by an extended repeat area overflow is used in block transfer mode, settings should be selected carefully. For details, see section 9.5.5, Extended Repeat Area Function.



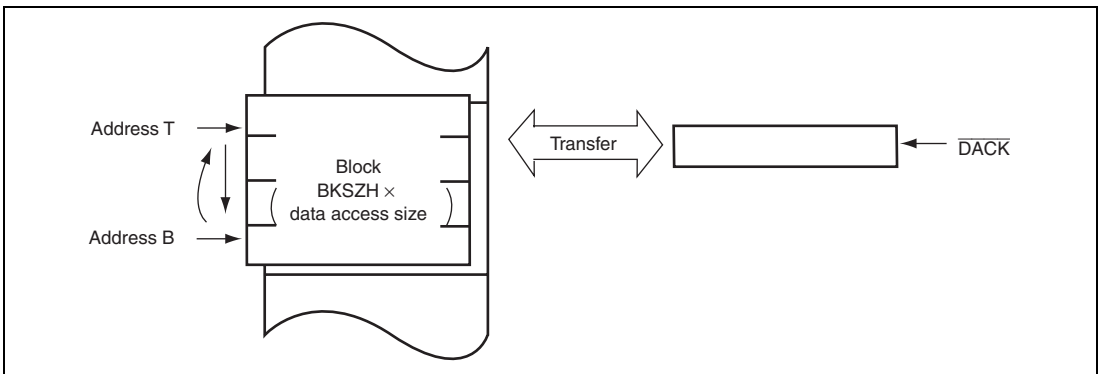
Figure 9.10 shows an example of the DMA transfer timing in block transfer mode. The transfer conditions are as follows:

- Address mode: single address mode
- Data access size: byte
- 1-block size: three bytes

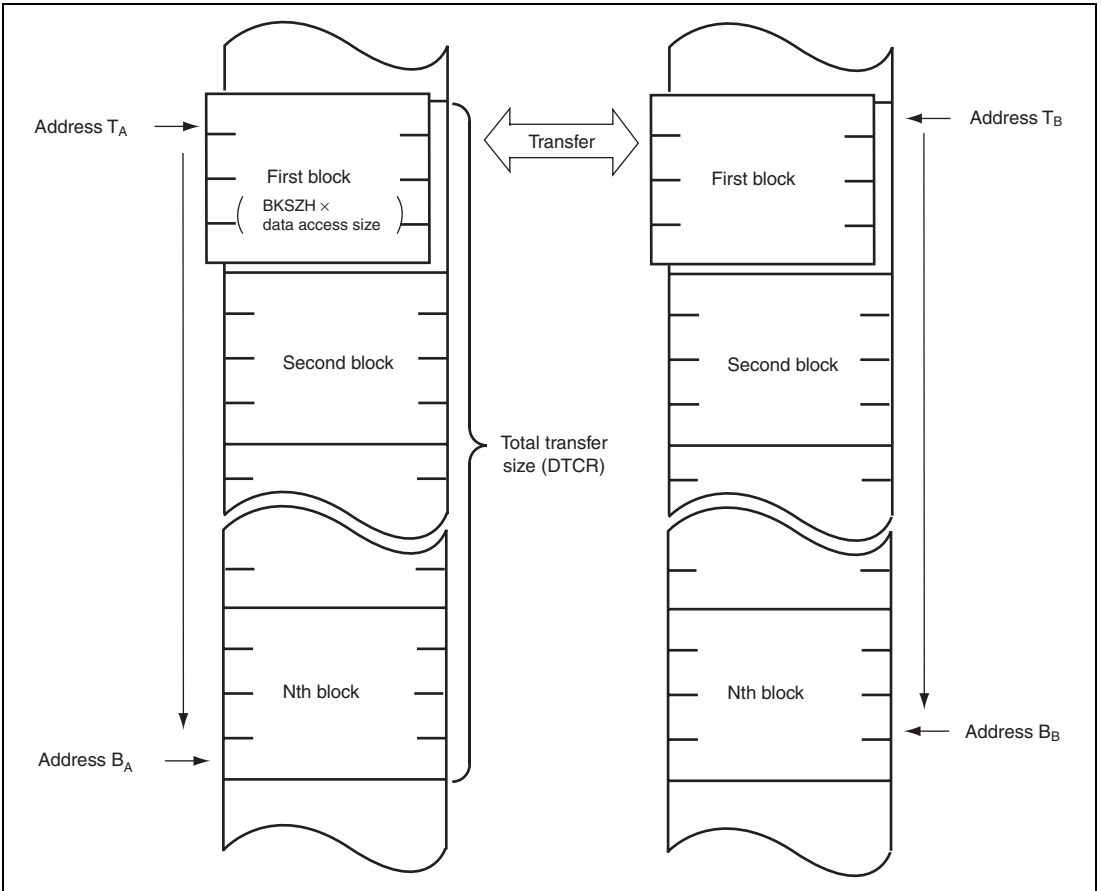
The block transfer mode operations in single address mode and in dual address mode are shown in figures 9.11 and 9.12, respectively.



**Figure 9.10 Operations in Block Transfer Mode**



**Figure 9.11 Operation in Single Address Mode in Block Transfer Mode (Block Area Specified)**



**Figure 9.12 Operation in Dual Address Mode in Block Transfer Mode  
(Block Area Not Specified)**

### 9.5.3 Activation Sources

The DMAC is activated by an auto request, an on-chip module interrupt, and an external request. The activation source is specified by bits DTF1 and DTF0 in DMDR.

#### (1) Activation by Auto Request

The auto request activation is used when a transfer request from an external device or an on-chip peripheral module is not generated such as a transfer between memory and memory or between memory and an on-chip peripheral module which does not request a transfer. A transfer request is automatically generated inside the DMAC. In auto request activation, setting the DTE bit in DMDR starts a transfer. The bus mode can be selected from cycle stealing and burst modes.

#### (2) Activation by On-Chip Module Interrupt

An interrupt request from an on-chip peripheral module (on-chip peripheral module interrupt) is used as a transfer request. When a DMA transfer is enabled ( $DTE = 1$ ), the DMA transfer is started by an on-chip module interrupt.

The activation source of the on-chip module interrupt is selected by the DMA module request select register (DMRSR). The activation sources are specified to the individual channels. Table 9.5 is a list of on-chip module interrupts for the DMAC. The interrupt request selected as the activation source can generate an interrupt request simultaneously to the CPU or DTC. For details, refer to section 6, Interrupt Controller.

The DMAC receives interrupt requests by on-chip peripheral modules independent of the interrupt controller. Therefore, the DMAC is not affected by priority given in the interrupt controller.

When the DMAC is activated while  $DTA = 1$ , the interrupt request flag is automatically cleared by a DMA transfer. If multiple channels use a single transfer request as an activation source, when the channel having priority is activated, the interrupt request flag is cleared. In this case, other channels may not be activated because the transfer request is not held in the DMAC.

When the DMAC is activated while  $DTA = 0$ , the interrupt request flag is not cleared by the DMAC and should be cleared by the CPU or DTC transfer.

When an activation source is selected while  $DTE = 0$ , the activation source does not request a transfer to the DMAC. It requests an interrupt to the CPU or DTC.

In addition, make sure that an interrupt request flag as an on-chip module interrupt source is cleared to 0 before writing 1 to the DTE bit.

**Table 9.5 List of On-Chip Module Interrupts to DMAC**

<b>On-Chip Module Interrupt Source</b>	<b>On-Chip Module</b>	<b>DMRSR (Vector Number)</b>
TGI0A (TGI0A input capture/compare match)	TPU_0	88
TGI1A (TGI1A input capture/compare match)	TPU_1	93
TGI2A (TGI2A input capture/compare match)	TPU_2	97
TGI3A (TGI3A input capture/compare match)	TPU_3	101
TGI4A (TGI4A input capture/compare match)	TPU_4	106
TGI5A (TGI5A input capture/compare match)	TPU_5	110
RX10 (receive data full interrupt from SCI channel 0)	SCI_0	145
TX10 (transmit data empty interrupt from SCI channel 0)	SCI_0	146
RX11 (receive data full interrupt from SCI channel 1)	SCI_1	149
TX11 (transmit data empty interrupt from SCI channel 1)	SCI_1	150
RX12 (receive data full interrupt from SCI channel 2)	SCI_2	153
TX12 (transmit data empty interrupt from SCI channel 2)	SCI_2	154
RX13 (receive data full interrupt from SCI channel 3)	SCI_3	157
TX13 (transmit data empty interrupt from SCI channel 3)	SCI_3	158
RX14 (receive data full interrupt from SCI channel 4)	SCI_4	161
TX14 (transmit data empty interrupt from SCI channel 4)	SCI_4	162
ADIO (conversion end interrupt from A/D converter )	ADIO	220
DSADI (conversion end interrupt from $\Delta\Sigma$ A/D converter)	DSADI	224

### (3) Activation by External Request

A transfer is started by a transfer request signal ( $\overline{\text{DREQ}}$ ) from an external device. When a DMA transfer is enabled ( $\text{DTE} = 1$ ), the DMA transfer is started by the  $\overline{\text{DREQ}}$  assertion. When a DMA transfer between on-chip peripheral modules is performed, select an activation source from the auto request and on-chip module interrupt (the external request cannot be used).

A transfer request signal is input to the  $\overline{\text{DREQ}}$  pin. The  $\overline{\text{DREQ}}$  signal is detected on the falling edge or low level. Whether the falling edge or low level detection is used is selected by the  $\text{DREQS}$  bit in  $\text{DMDR}$ . To perform a block transfer, select the low level detection ( $\text{DREQS} = 0$ ).

When an external request is selected as an activation source, clear the  $\text{DDR}$  bit to 0 and set the  $\text{ICR}$  bit to 1 for the corresponding pin. For details, see section 11, I/O Ports.

#### 9.5.4 Bus Access Modes

There are two types of bus access modes: cycle stealing and burst.

When an activation source is the auto request, the cycle stealing or burst mode is selected by bit  $\text{DTF0}$  in  $\text{DMDR}$ . When an activation source is the on-chip module interrupt or external request, the cycle stealing mode is selected.

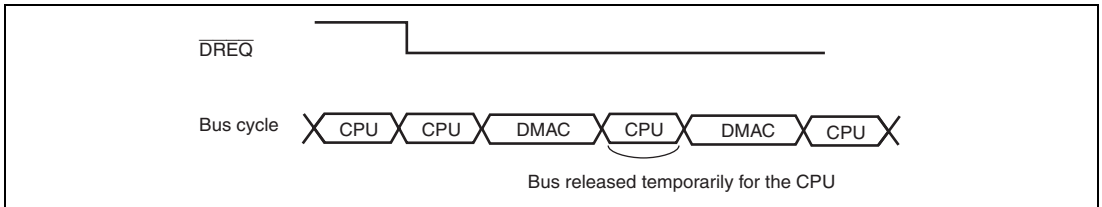
##### (1) Cycle Stealing Mode

In cycle stealing mode, the  $\text{DMAC}$  releases the bus every time one unit of transfers (byte, word, longword, or 1-block size) is completed. After that, when a transfer is requested, the  $\text{DMAC}$  obtains the bus to transfer 1-unit data and then releases the bus on completion of the transfer. This operation is continued until the transfer end condition is satisfied.

When a transfer is requested to another channel during a DMA transfer, the  $\text{DMAC}$  releases the bus and then transfers data for the requested channel. For details on operations when a transfer is requested to multiple channels, see section 9.5.8, Priority of Channels.

Figure 9.13 shows an example of timing in cycle stealing mode. The transfer conditions are as follows:

- Address mode: Single address mode
- Sampling method of the  $\overline{\text{DREQ}}$  signal: Low level detection



**Figure 9.13 Example of Timing in Cycle Stealing Mode**

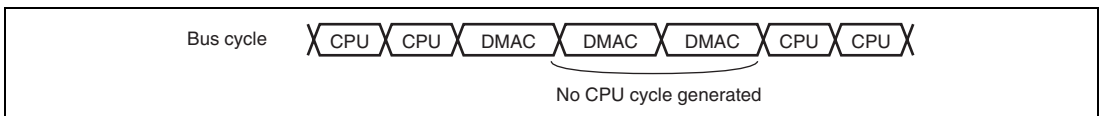
## (2) Burst Access Mode

In burst mode, once it takes the bus, the DMAC continues a transfer without releasing the bus until the transfer end condition is satisfied. Even if a transfer is requested from another channel having priority, the transfer is not stopped once it is started. The DMAC releases the bus in the next cycle after the transfer for the channel in burst mode is completed. This is similarly to operation in cycle stealing mode. However, setting the IBCCS bit in IBCR of the bus controller makes the DMAC release the bus to pass the bus to another bus master.

In block transfer mode, the burst mode setting is ignored (operation is the same as that in burst mode during one block of transfers). The DMAC is always operated in cycle stealing mode.

Clearing the DTE bit in DMDR stops a DMA transfer. A transfer requested before the DTE bit is cleared to 0 by the DMAC is executed. When an interrupt by a transfer size error, a repeat size end, or an extended repeat area overflow occurs, the DTE bit is cleared to 0 and the transfer ends.

Figure 9.14 shows an example of timing in burst mode.



**Figure 9.14 Example of Timing in Burst Mode**

### 9.5.5 Extended Repeat Area Function

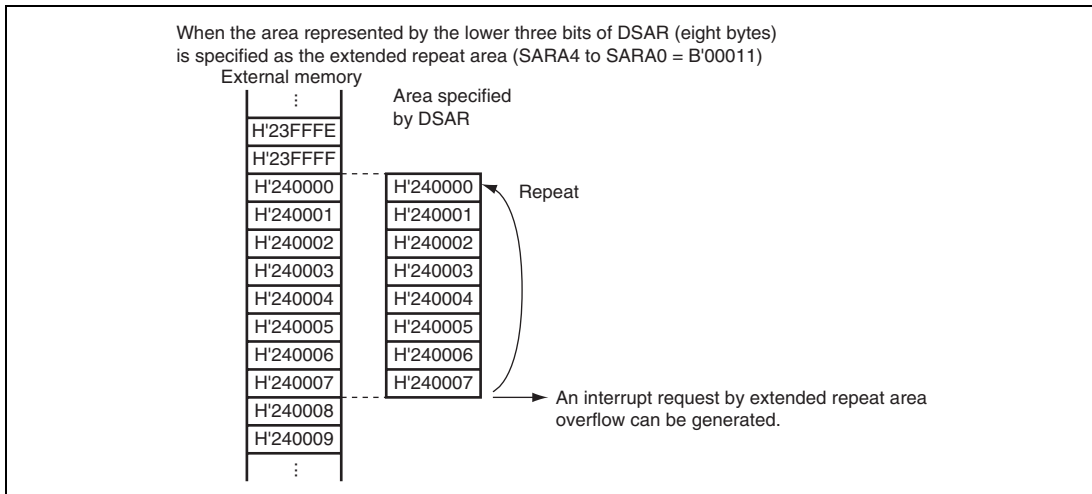
The source and destination address sides can be specified as the extended repeat area. The contents of the address register repeat addresses within the area specified as the extended repeat area. For example, to use a ring buffer as the transfer target, the contents of the address register should return to the start address of the buffer every time the contents reach the end address of the buffer (overflow on the ring buffer address). This operation can automatically be performed using the extended repeat area function of the DMAC.

The extended repeat areas can be specified independently to the source address register (DSAR) and destination address register (DDAR).

The extended repeat area on the source address is specified by bits SARA4 to SARA0 in DACR. The extended repeat area on the destination address is specified by bits DARA4 to DARA0 in DACR. The extended repeat area sizes for each side can be specified independently.

A DMA transfer is stopped and an interrupt by an extended repeat area overflow can be requested to the CPU when the contents of the address register reach the end address of the extended repeat area. When an overflow on the extended repeat area set in DSAR occurs while the SARIE bit in DACR is set to 1, the ESIF bit in DMDR is set to 1 and the DTE bit in DMDR is cleared to 0 to stop the transfer. At this time, if the ESIE bit in DMDR is set to 1, an interrupt by an extended repeat area overflow is requested to the CPU. When the DARIE bit in DACR is set to 1, an overflow on the extended repeat area set in DDAR occurs, meaning that the destination side is a target. During the interrupt handling, setting the DTE bit in DMDR resumes the transfer.

Figure 9.15 shows an example of the extended repeat area operation.



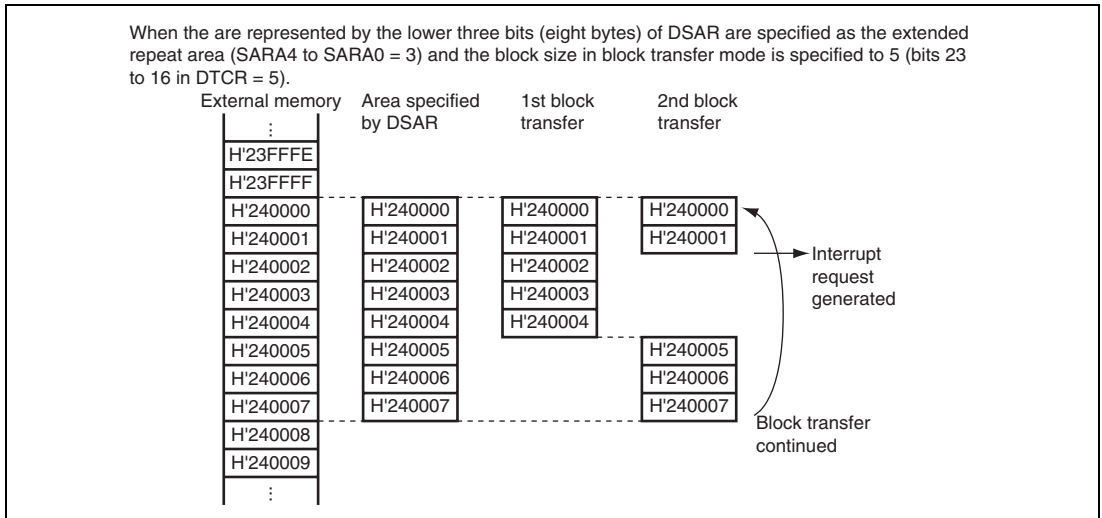
**Figure 9.15 Example of Extended Repeat Area Operation**

When an interrupt by an extended repeat area overflow is used in block transfer mode, the following should be taken into consideration.

When a transfer is stopped by an interrupt by an extended repeat area overflow, the address register must be set so that the block size is a power of 2 or the block size boundary is aligned with the extended repeat area boundary. When an overflow on the extended repeat area occurs during a transfer of one block, the interrupt by the overflow is suspended and the transfer overruns.



Figure 9.16 shows examples when the extended repeat area function is used in block transfer mode.

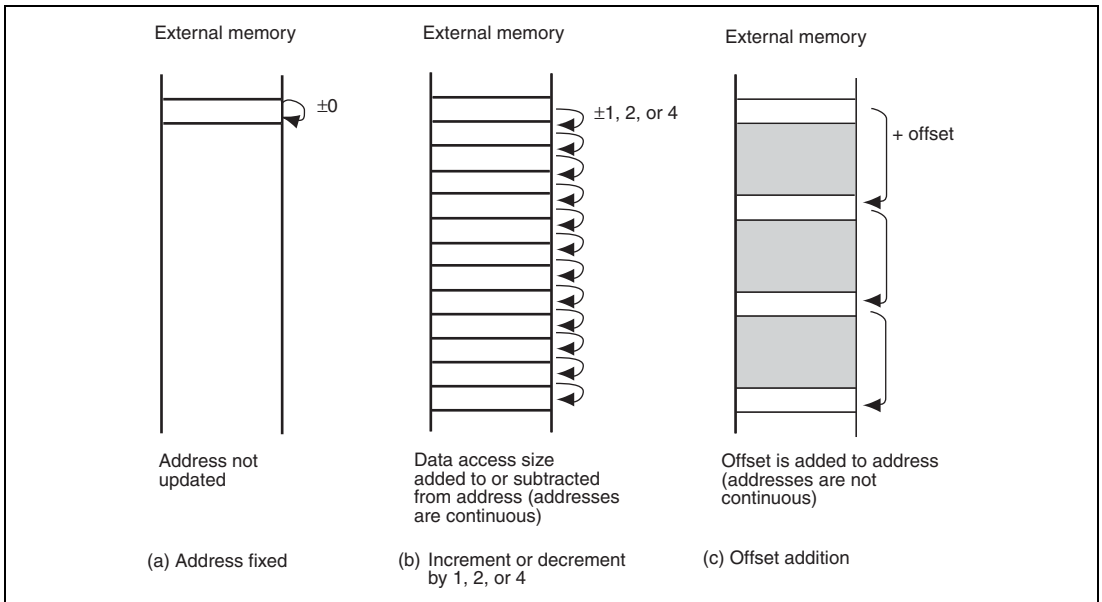


**Figure 9.16 Example of Extended Repeat Area Function in Block Transfer Mode**

### 9.5.6 Address Update Function using Offset

The source and destination addresses are updated by fixing, increment/decrement by 1, 2, or 4, or offset addition. When the offset addition is selected, the offset specified by the offset register (DOFR) is added to the address every time the DMAC transfers the data access size of data. This function realizes a data transfer where addresses are allocated to separated areas.

Figure 9.17 shows the address update method.



**Figure 9.17 Address Update Method**

In item (a), Address fixed, the transfer source or destination address is not updated indicating the same address.

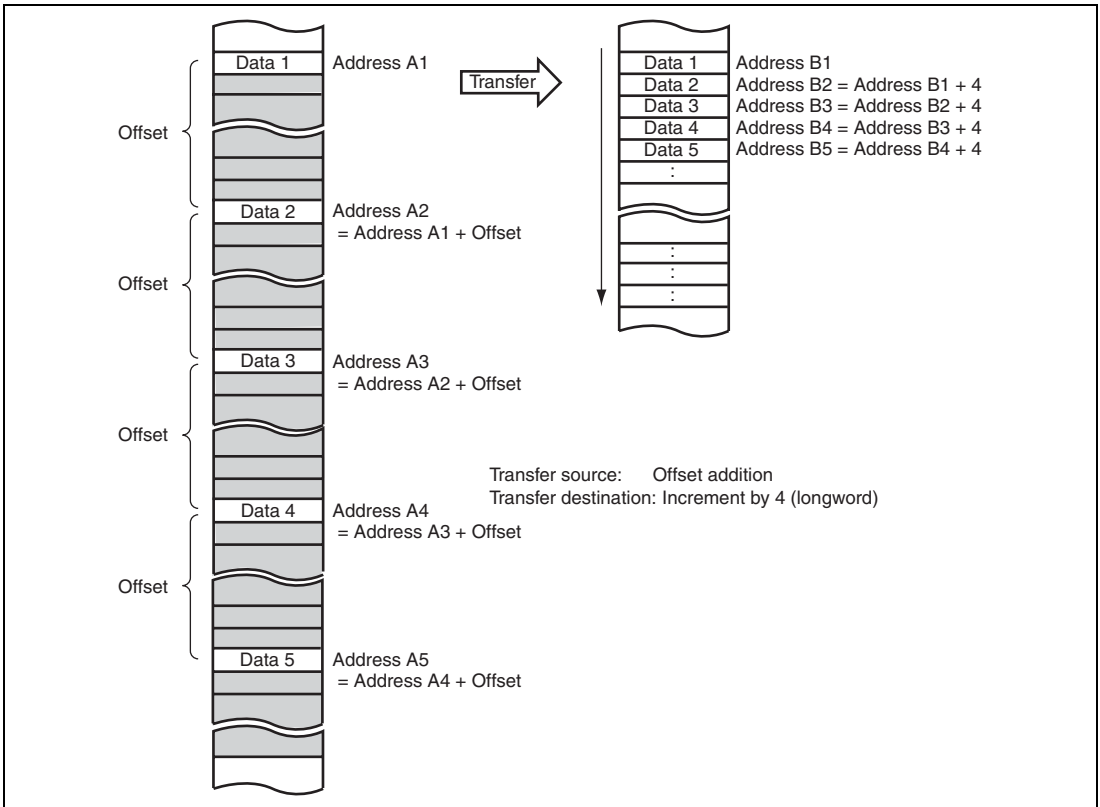
In item (b), Increment or decrement by 1, 2, or 4, the transfer source or destination address is incremented or decremented by the value according to the data access size at each transfer. Byte, word, or longword can be specified as the data access size. The value of 1 for byte, 2 for word, and 4 for longword is used for updating the address. This operation realizes the data transfer placed in consecutive areas.

In item (c), Offset addition, the address update does not depend on the data access size. The offset specified by DOFR is added to the address every time the DMAC transfers data of the data access size.

The address is calculated by the offset set in DOFR and the contents of DSAR and DDAR. Although the DMAC calculates only addition, an offset subtraction can be realized by setting the negative value in DOFR. In this case, the negative value must be 2's complement.

## (1) Basic Transfer Using Offset

Figure 9.18 shows a basic operation of a transfer using the offset addition.

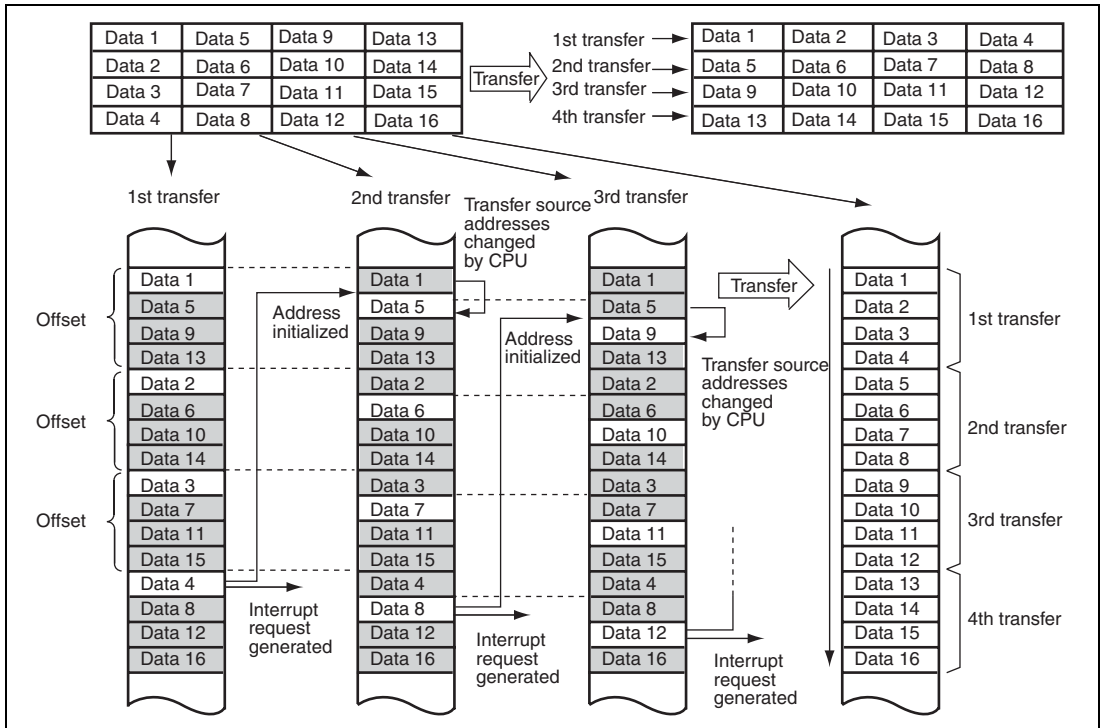


**Figure 9.18 Operation of Offset Addition**

In figure 9.18, the offset addition is selected as the transfer source address update and increment or decrement by 1, 2, or 4 is selected as the transfer destination address. The address update means that data at the address which is away from the previous transfer source address by the offset is read from. The data read from the address away from the previous address is written to the consecutive area in the destination side.

**(2) XY Conversion Using Offset**

Figure 9.19 shows the XY conversion using the offset addition in repeat transfer mode.

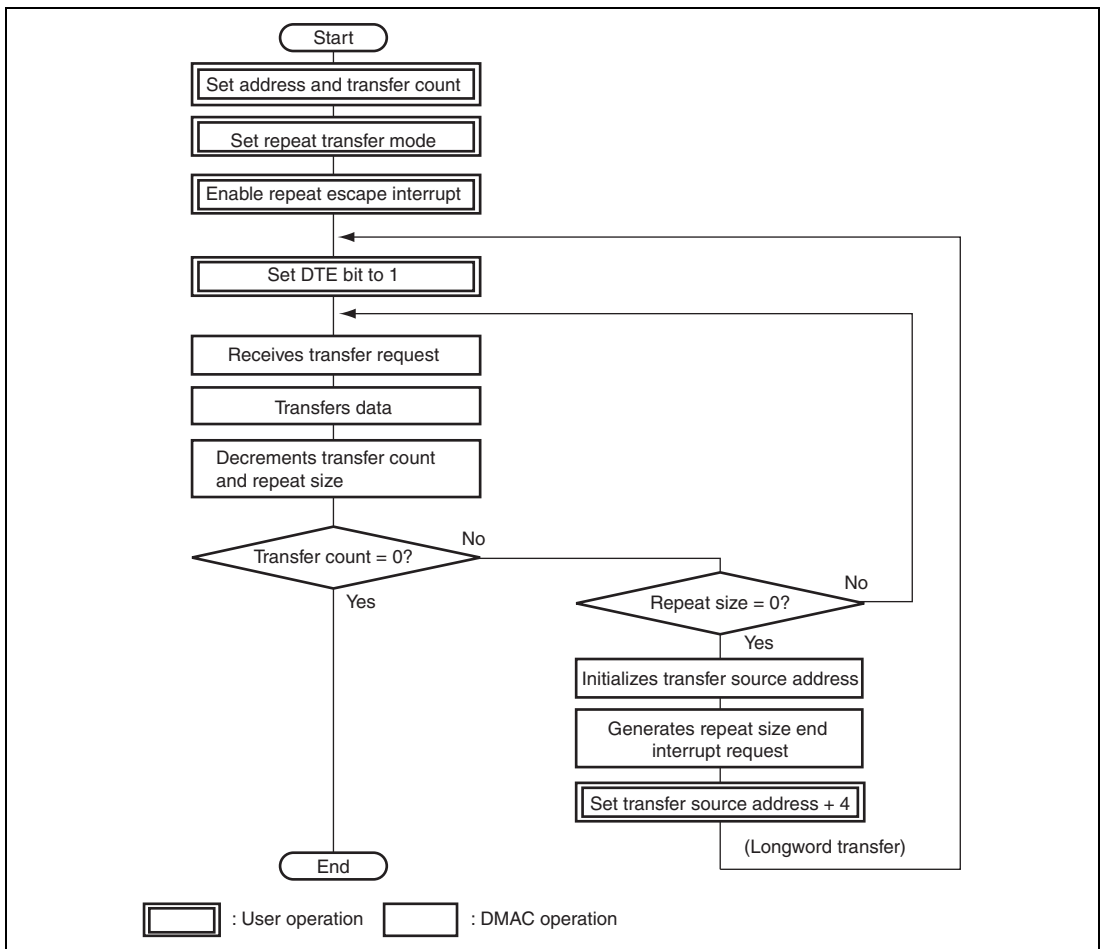


**Figure 9.19 XY Conversion Operation Using Offset Addition in Repeat Transfer Mode**

In figure 9.19, the source address side is specified to the repeat area by DACR and the offset addition is selected. The offset value is set to  $4 \times$  data access size (when the data access size is longword, H'00000010 is set in DOFR, as an example). The repeat size is set to  $4 \times$  data access size (when the data access size is longword, the repeat size is set to  $4 \times 4 = 16$  bytes, as an example). The increment or decrement by 1, 2, or 4 is specified as the transfer destination address. A repeat size end interrupt is requested when the repeat size of transfers is completed.

When a transfer starts, the transfer source address is added to the offset every time data is transferred. The transfer data is written to the destination continuous addresses. When data 4 is transferred meaning that the repeat size of transfers is completed, the transfer source address returns to the transfer start address (address of data 1 on the transfer source) and a repeat size end interrupt is requested. While this interrupt stops the transfer temporarily, the contents of DSAR are written to the address of data 5 by the CPU (when the data access size is longword, write the data 1 address + 4). When the DTE bit in DMDR is set to 1, the transfer is resumed from the state when the transfer is stopped. Accordingly, operations are repeated and the transfer source data is transposed to the destination area (XY conversion).

Figure 9.20 shows a flowchart of the XY conversion.



**Figure 9.20 XY Conversion Flowchart Using Offset Addition in Repeat Transfer Mode**

### (3) Offset Subtraction

When setting the negative value in DOFR, the offset value must be 2's complement. The 2's complement is obtained by the following formula.

2's complement of offset =  $1 + \sim\text{offset}$  ( $\sim$ : bit inversion)

Example: 2's complement of H'0001FFFF  
= H'FFFE0000 + H'00000001  
= H'FFFE0001

The value of 2's complement can be obtained by the NEG.L instruction.

#### 9.5.7 Register during DMA Transfer

The DMAC registers are updated by a DMA transfer. The value to be updated differs according to the other settings and transfer state. The registers to be updated are DSAR, DDAR, DTCR, bits BKSZH and BKSZ in DBSR, and the DTE, ACT, ERRF, ESIF, and DTIF bits in DMDR.

##### (1) DMA Source Address Register

When the transfer source address set in DSAR is accessed, the contents of DSAR are output and then are updated to the next address.

The increment or decrement can be specified by bits SAT1 and SAT0 in DACR. When SAT1 and SAT0 = B'00, the address is fixed. When SAT1 and SAT0 = B'01, the address is added with the offset. When SAT1 and SAT0 = B'10, the address is incremented. When SAT1 and SAT0 = B'11, the address is decremented. The size of increment or decrement depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the source address is word or longword, when the source address is not aligned with the word or longword boundary, the read bus cycle is divided into byte or word cycles. While data of one word or one longword is being read, the size of increment or decrement is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After one word or one longword of data is read, the address when the read cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the source address side, the source address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the source address side, operation follows the setting. The upper address bits are fixed and is not affected by the address update.

While data is being transferred, DSAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DSAR during the transfer may be updated regardless of the access by the CPU. Moreover, DSAR for the channel being transferred must not be written to.

## (2) DMA Destination Address Register

When the transfer destination address set in DDAR is accessed, the contents of DDAR are output and then are updated to the next address.

The increment or decrement can be specified by bits DAT1 and DAT0 in DACR. When DAT1 and DAT0 = B'00, the address is fixed. When DAT1 and DAT0 = B'01, the address is added with the offset. When DAT1 and DAT0 = B'10, the address is incremented. When DAT1 and DAT0 = B'11, the address is decremented. The incrementing or decrementing size depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the destination address is word or longword, when the destination address is not aligned with the word or longword boundary, the write bus cycle is divided into byte and word cycles. While one word or one longword of data is being written, the incrementing or decrementing size is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After the one word or one longword of data is written, the address when the write cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the destination address side, the destination address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the destination address side, operation follows the setting. The upper address bits are fixed and is not affected by the address update.

While data is being transferred, DDAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DDAR during the transfer may be updated regardless of the access by the CPU. Moreover, DDAR for the channel being transferred must not be written to.

### **(3) DMA Transfer Count Register (DTCR)**

A DMA transfer decrements the contents of DTCR by the transferred bytes. When byte data is transferred, DTCR is decremented by 1. When word data is transferred, DTCR is decremented by 2. When longword data is transferred, DTCR is decremented by 4. However, when DTCR = 0, the contents of DTCR are not changed since the number of transfers is not counted.

While data is being transferred, all the bits of DTCR may be changed. DTCR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DTCR during the transfer may be updated regardless of the access by the CPU. Moreover, DTCR for the channel being transferred must not be written to.

When a conflict occurs between the address update by DMA transfer and write access by the CPU, the CPU has priority. When a conflict occurs between change from 1, 2, or 4 to 0 in DTCR and write access by the CPU (other than 0), the CPU has priority in writing to DTCR. However, the transfer is stopped.

### **(4) DMA Block Size Register (DBSR)**

DBSR is enabled in block or repeat transfer mode. Bits 31 to 16 in DBSR function as BKSZH and bits 15 to 0 in DBSR function as BKSZ. The BKSZH bits (16 bits) store the block size and repeat size and its value is not changed. The BKSZ bits (16 bits) function as a counter for the block size and repeat size and its value is decremented every transfer by 1. When the BKSZ value is to change from 1 to 0 by a DMA transfer, 0 is not stored but the BKSZH value is loaded into the BKSZ bits.

Since the upper 16 bits of DBSR are not updated, DBSR can be accessed in words.

DBSR for the channel being transferred must not be written to.



## (5) DTE Bit in DMDR

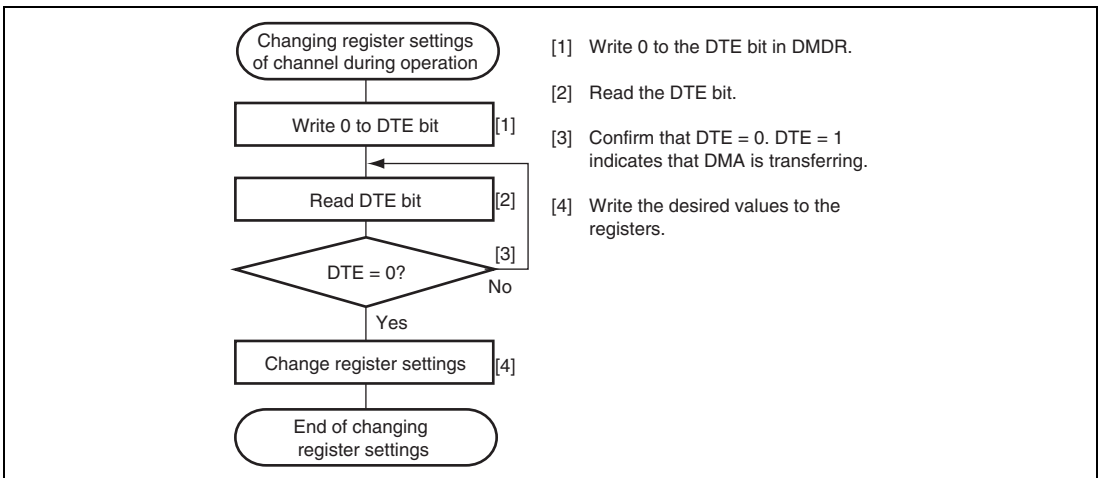
Although the DTE bit in DMDR enables or disables data transfer by the CPU write access, it is automatically cleared to 0 according to the DMA transfer state by the DMAC.

The conditions for clearing the DTE bit by the DMAC are as follows:

- When the total size of transfers is completed
- When a transfer is completed by a transfer size error interrupt
- When a transfer is completed by a repeat size end interrupt
- When a transfer is completed by an extended repeat area overflow interrupt
- When a transfer is stopped by an NMI interrupt
- When a transfer is stopped by an address error
- Reset state
- Hardware standby mode
- When a transfer is stopped by writing 0 to the DTE bit

Writing to the registers for the channels when the corresponding DTE bit is set to 1 is prohibited (except for the DTE bit). When changing the register settings after writing 0 to the DTE bit, confirm that the DTE bit has been cleared to 0.

Figure 9.21 show the procedure for changing the register settings for the channel being transferred.



**Figure 9.21 Procedure for Changing Register Setting For Channel being Transferred**

**(6) ACT Bit in DMDR**

The ACT bit in DMDR indicates whether the DMAC is in the idle or active state. When DTE = 0 or DTE = 1 and the DMAC is waiting for a transfer request, the ACT bit is 0. Otherwise (the DMAC is in the active state), the ACT bit is 1. When individual transfers are stopped by writing 0 and the transfer is not completed, the ACT bit retains 1.

In block transfer mode, even if individual transfers are stopped by writing 0 to the DTE bit, the 1-block size of transfers is not stopped. The ACT bit retains 1 from writing 0 to the DTE bit to completion of a 1-block size transfer.

In burst mode, up to three times of DMA transfer are performed from the cycle in which the DTE bit is written to 0. The ACT bit retains 1 from writing 0 to the DTE bit to completion of DMA transfer.

**(7) ERRF Bit in DMDR**

When an address error or an NMI interrupt occur, the DMAC clears the DTE bits for all the channels to stop a transfer. In addition, it sets the ERRF bit in DMDR\_0 to 1 to indicate that an address error or an NMI interrupt has occurred regardless of whether or not the DMAC is in operation.

**(8) ESIF Bit in DMDR**

When an interrupt by an transfer size error, a repeat size end, or an extended repeat area overflow is requested, the ESIF bit in DMDR is set to 1. When both the ESIF and ESIE bits are set to 1, a transfer escape interrupt is requested to the CPU or DTC.

The ESIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle of the interrupt source is completed.

The ESIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 9.8, Interrupt Sources.

## (9) DTIF Bit in DMDR

The DTIF bit in DMDR is set to 1 after the total transfer size of transfers is completed. When both the DTIF and DTIE bits in DMDR are set to 1, a transfer end interrupt by the transfer counter is requested to the CPU or DTC.

The DTIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle is completed.

The DTIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 9.8, Interrupt Sources.

### 9.5.8 Priority of Channels

The channels of the DMAC are given following priority levels: channel 0 > channel 1. Table 9.6 shows the priority levels among the DMAC channels.

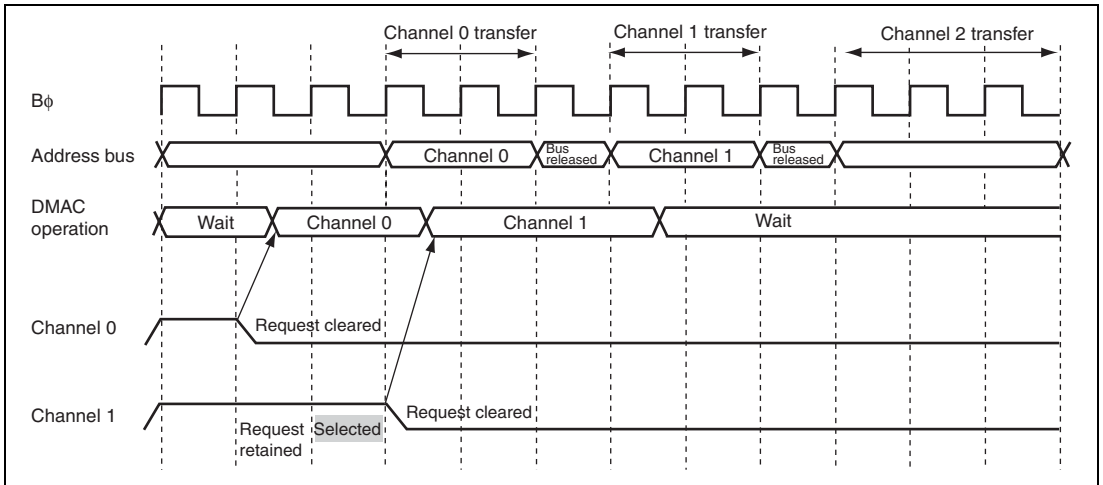
**Table 9.6 Priority among DMAC Channels**

<b>Channel</b>	<b>Priority</b>
Channel 0	High
Channel 1	Low

The channel having highest priority other than the channel being transferred is selected when a transfer is requested from other channels. The selected channel starts the transfer after the channel being transferred releases the bus. At this time, when a bus master other than the DMAC requests the bus, the cycle for the bus master is inserted.

In a burst transfer or a block transfer, channels are not switched.

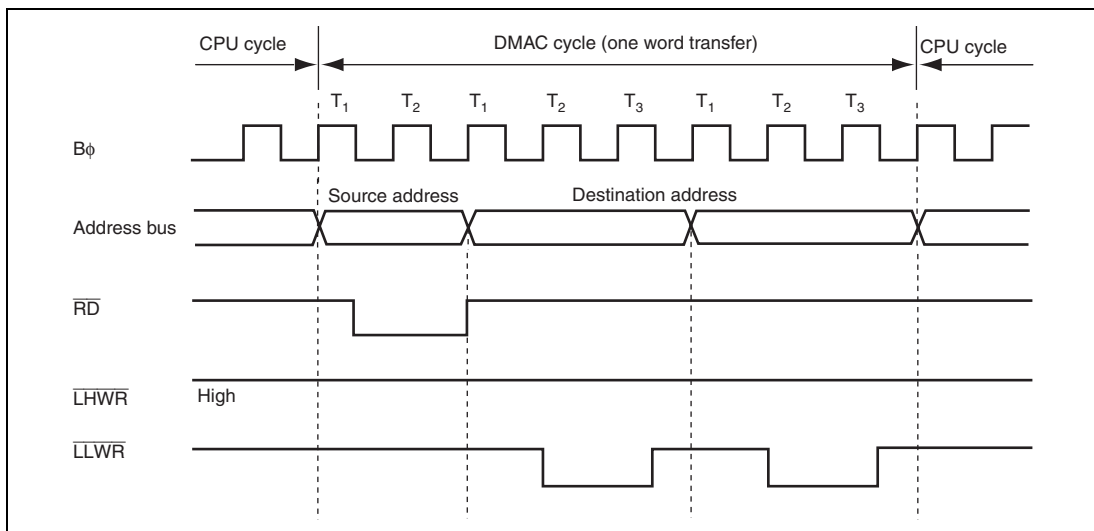
Figure 9.22 shows a transfer example when multiple transfer requests from channels 0 and 1.



**Figure 9.22 Example of Timing for Channel Priority**

### 9.5.9 DMA Basic Bus Cycle

Figure 9.23 shows an examples of signal timing of a basic bus cycle. In figure 9.23, data is transferred in words from the 16-bit 2-state access space to the 8-bit 3-state access space. When the bus mastership is passed from the DMAC to the CPU, data is read from the source address and it is written to the destination address. The bus is not released between the read and write cycles by other bus requests. DMAC bus cycles follows the bus controller settings.



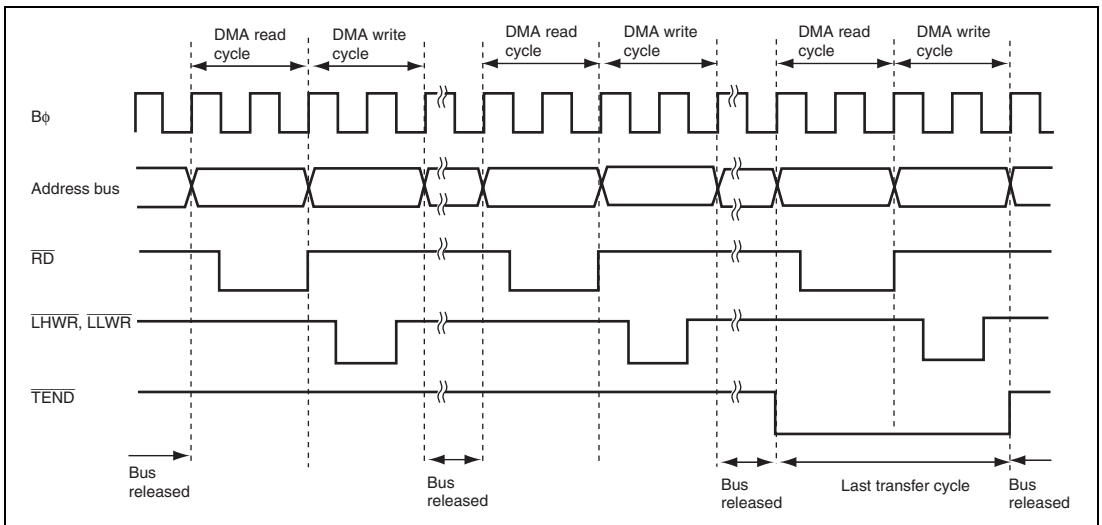
**Figure 9.23 Example of Bus Timing of DMA Transfer**

### 9.5.10 Bus Cycles in Dual Address Mode

#### (1) Normal Transfer Mode (Cycle Stealing Mode)

In cycle stealing mode, the bus is released every time one transfer size of data (one byte, one word, or one longword) is completed. One bus cycle or more by the CPU or DTC are executed in the bus released cycles.

In figure 9.24, the  $\overline{\text{TEND}}$  signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by cycle stealing.

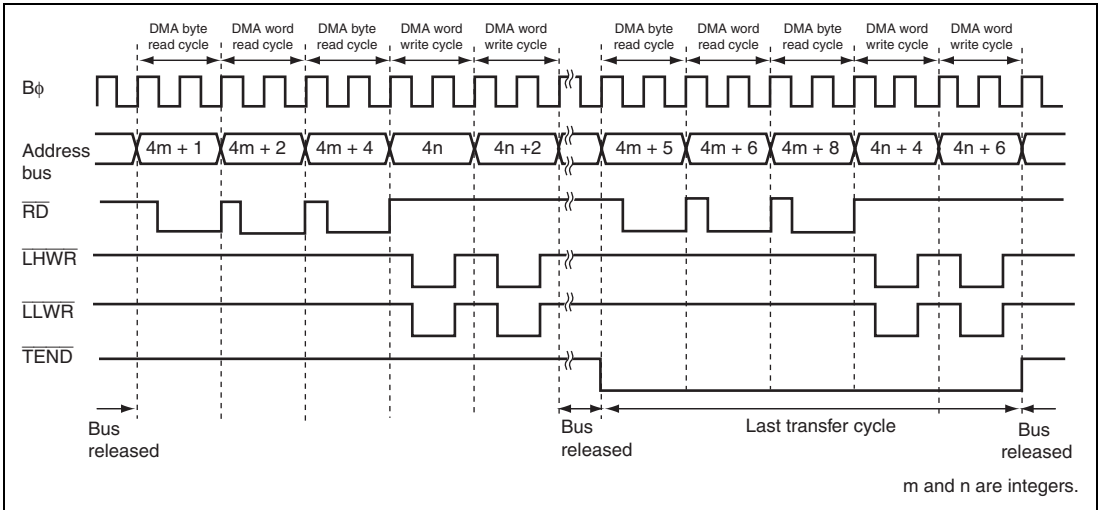


**Figure 9.24 Example of Transfer in Normal Transfer Mode by Cycle Stealing**

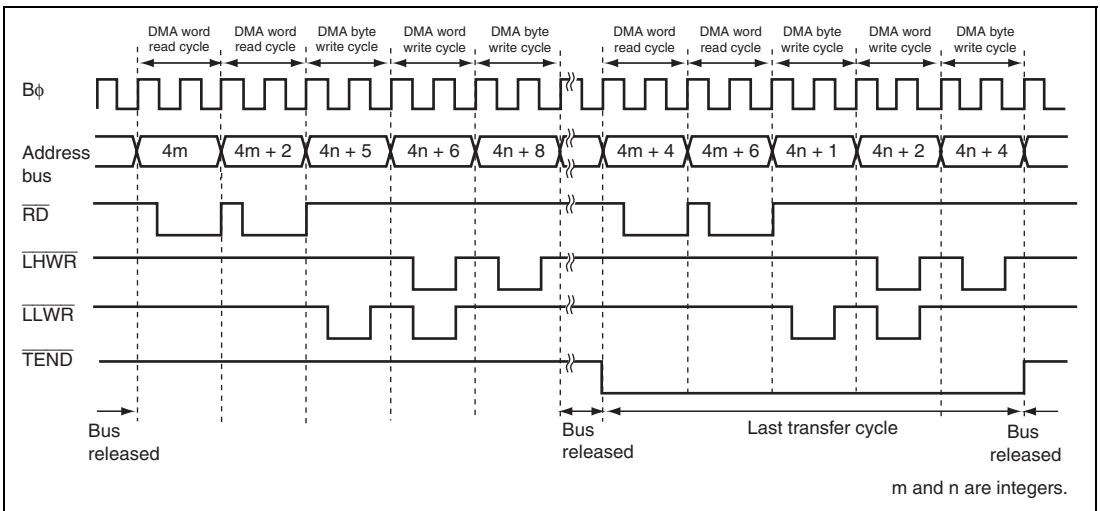
In figures 9.25 and 9.26, the  $\overline{\text{TEND}}$  signal output is enabled and data is transferred in longwords from the external 16-bit 2-state access space to the 16-bit 2-state access space in normal transfer mode by cycle stealing.

In figure 9.25, the transfer source (DSAR) is not aligned with a longword boundary and the transfer destination (DDAR) is aligned with a longword boundary.

In figure 9.26, the transfer source (DSAR) is aligned with a longword boundary and the transfer destination (DDAR) is not aligned with a longword boundary.



**Figure 9.25 Example of Transfer in Normal Transfer Mode by Cycle Stealing  
(Transfer Source DSAR = Odd Address and Source Address Increment)**



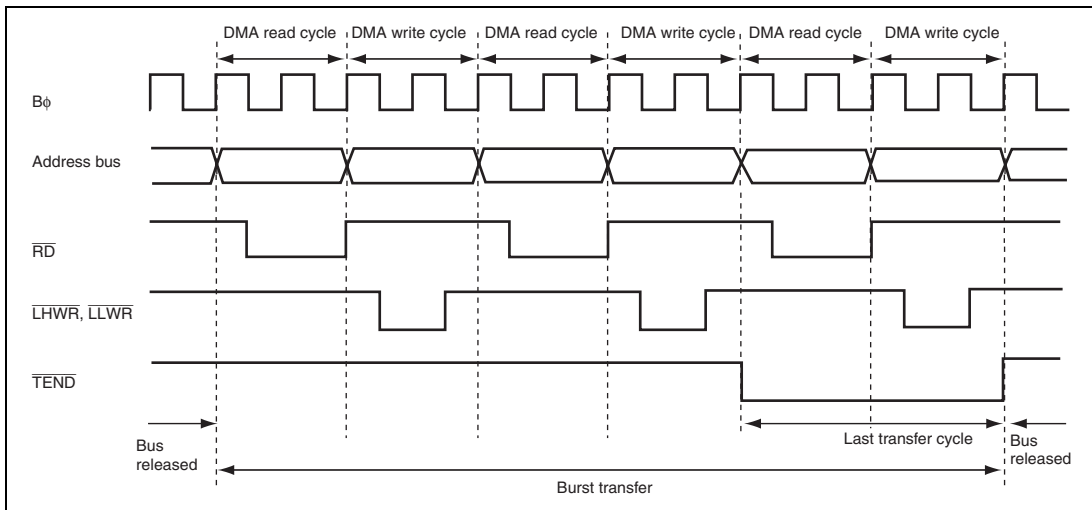
**Figure 9.26 Example of Transfer in Normal Transfer Mode by Cycle Stealing  
(Transfer Destination DDAR = Odd Address and Destination Address Decrement)**

## (2) Normal Transfer Mode (Burst Mode)

In burst mode, one byte, one word, or one longword of data continues to be transferred until the transfer end condition is satisfied.

When a burst transfer starts, a transfer request from a channel having priority is suspended until the burst transfer is completed.

In figure 9.27, the  $\overline{\text{TEND}}$  signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by burst access.



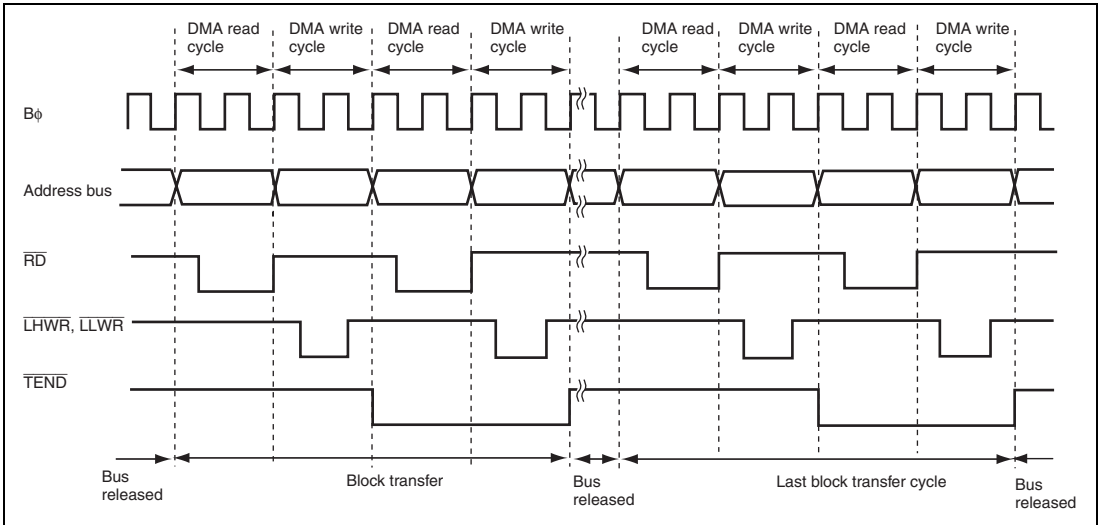
**Figure 9.27 Example of Transfer in Normal Transfer Mode by Burst Access**



### (3) Block Transfer Mode

In block transfer mode, the bus is released every time a 1-block size of transfers at a single transfer request is completed.

In figure 9.28, the  $\overline{TEND}$  signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in block transfer mode.



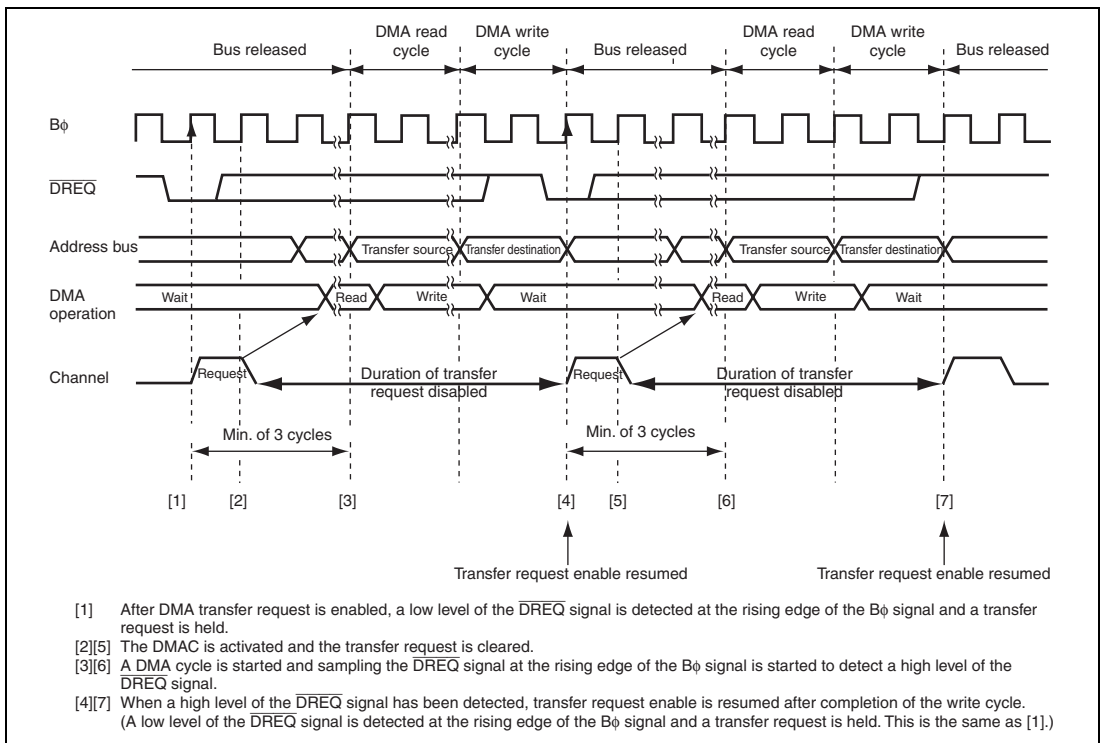
**Figure 9.28 Example of Transfer in Block Transfer Mode**

#### (4) Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 9.29 shows an example of normal transfer mode activated by the  $\overline{\text{DREQ}}$  signal falling edge.

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the  $\overline{\text{DREQ}}$  signal for falling edge detection. If a high level of the  $\overline{\text{DREQ}}$  signal has been detected until completion of the DMA write cycle, receiving the next transfer request resumes and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.

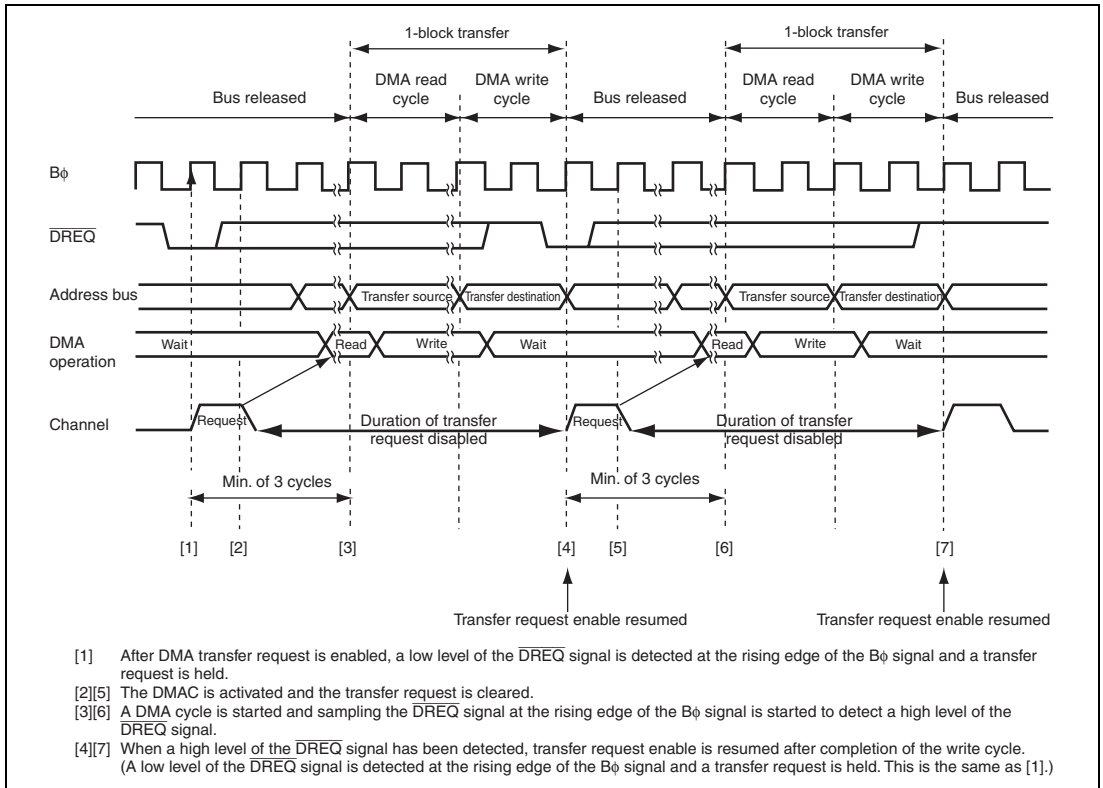


**Figure 9.29 Example of Transfer in Normal Transfer Mode Activated by  $\overline{\text{DREQ}}$  Falling Edge**

Figure 9.30 shows an example of block transfer mode activated by the  $\overline{\text{DREQ}}$  signal falling edge.

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the  $\overline{\text{DREQ}}$  signal for falling edge detection. If a high level of the  $\overline{\text{DREQ}}$  signal has been detected until completion of the DMA write cycle, receiving DMA transfer request resumes and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



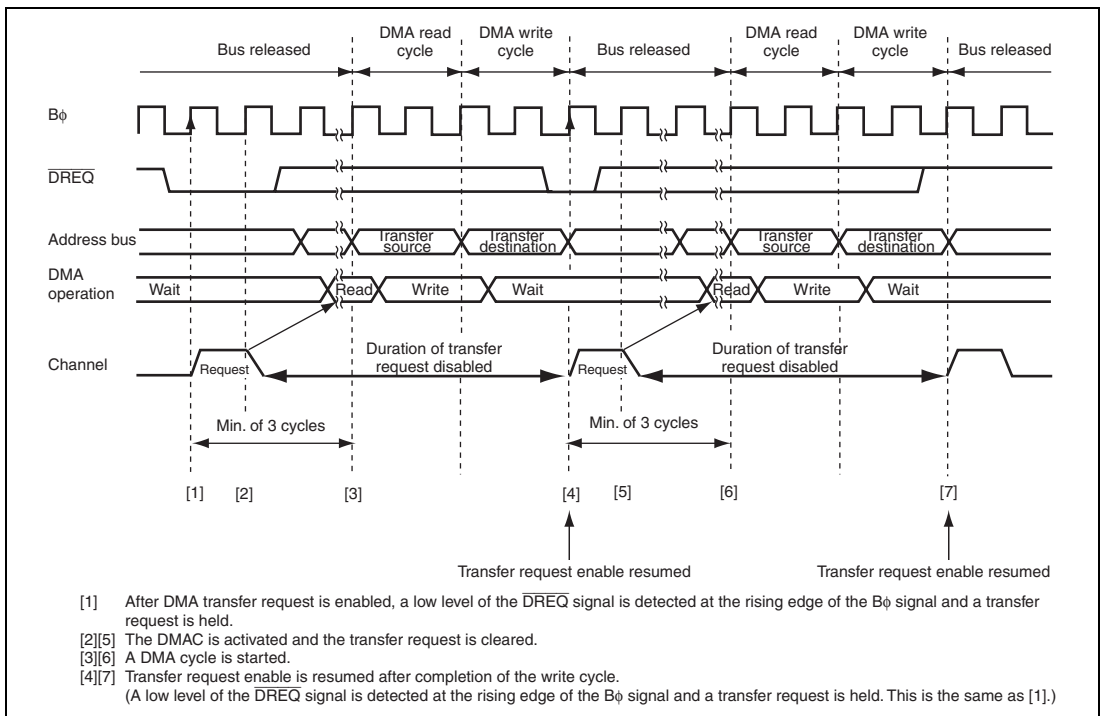
**Figure 9.30 Example of Transfer in Block Transfer Mode Activated by  $\overline{\text{DREQ}}$  Falling Edge**

### (5) Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 9.31 shows an example of normal transfer mode activated by the  $\overline{\text{DREQ}}$  signal low level.

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.

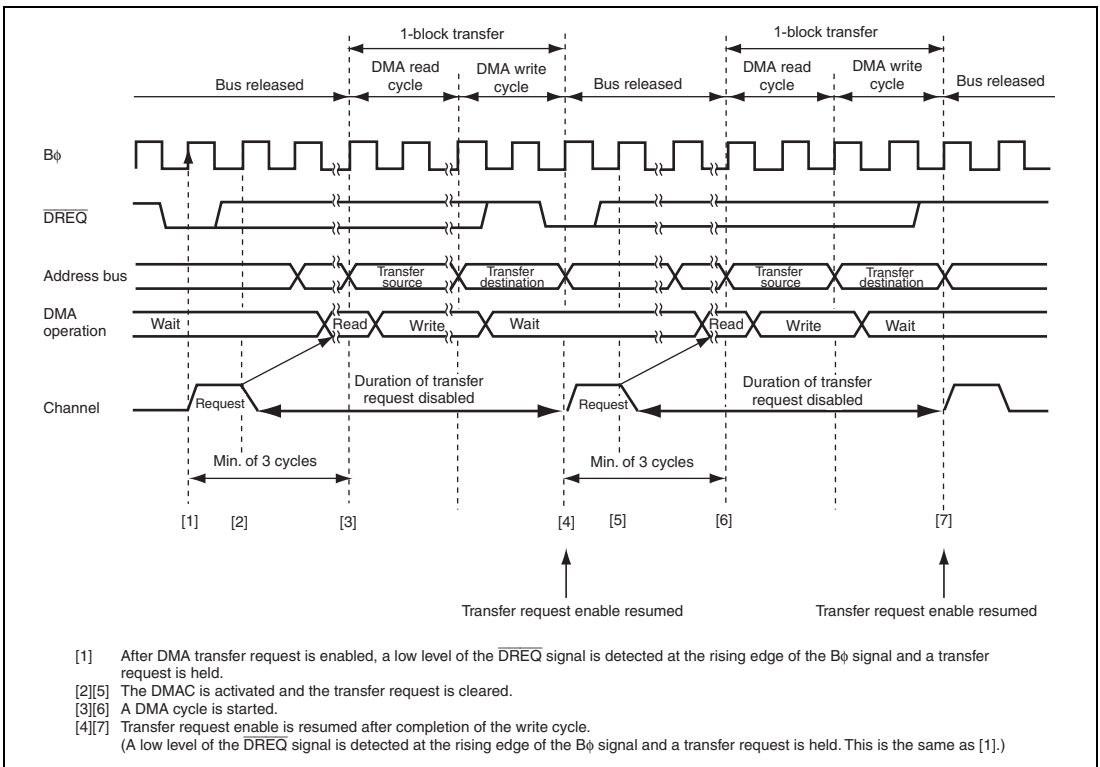


**Figure 9.31 Example of Transfer in Normal Transfer Mode Activated by  $\overline{\text{DREQ}}$  Low Level**

Figure 9.32 shows an example of block transfer mode activated by the  $\overline{\text{DREQ}}$  signal low level.

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



**Figure 9.32 Example of Transfer in Block Transfer Mode Activated by  $\overline{\text{DREQ}}$  Low Level**

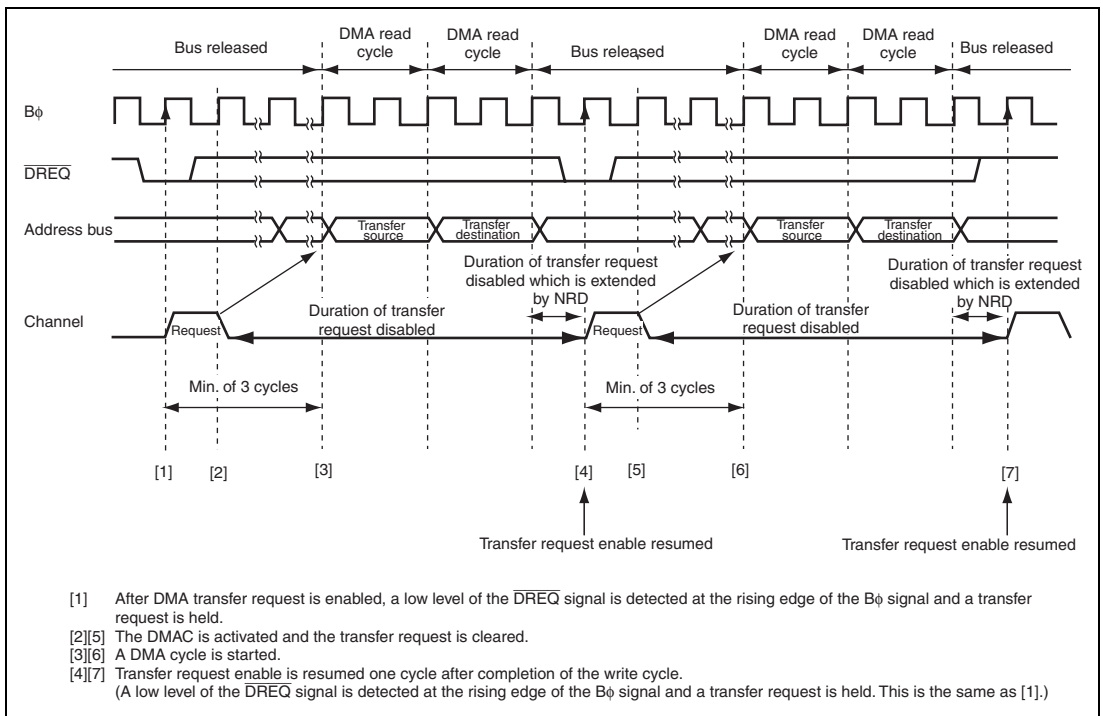
**(6) Activation Timing by  $\overline{\text{DREQ}}$  Low Level with  $\text{NRD} = 1$** 

When the  $\text{NRD}$  bit in  $\text{DMDR}$  is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 9.33 shows an example of normal transfer mode activated by the  $\overline{\text{DREQ}}$  signal low level with  $\text{NRD} = 1$ .

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the  $\text{DTE}$  bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the  $\text{DMAC}$ . When the  $\text{DMAC}$  is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



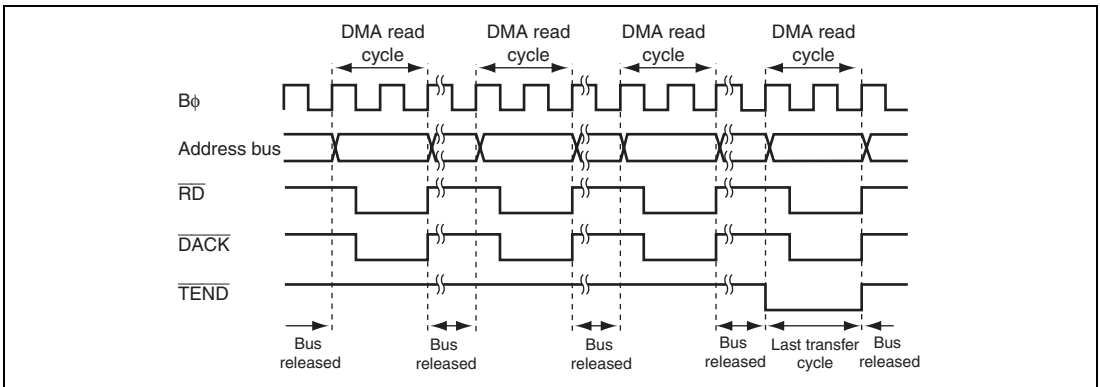
**Figure 9.33 Example of Transfer in Normal Transfer Mode Activated by  $\overline{\text{DREQ}}$  Low Level with  $\text{NRD} = 1$**

### 9.5.11 Bus Cycles in Single Address Mode

#### (1) Single Address Mode (Read and Cycle Stealing)

In single address mode, one byte, one word, or one longword of data is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU or DTC are executed in the bus released cycles.

In figure 9.34, the  $\overline{\text{TEND}}$  signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (read).

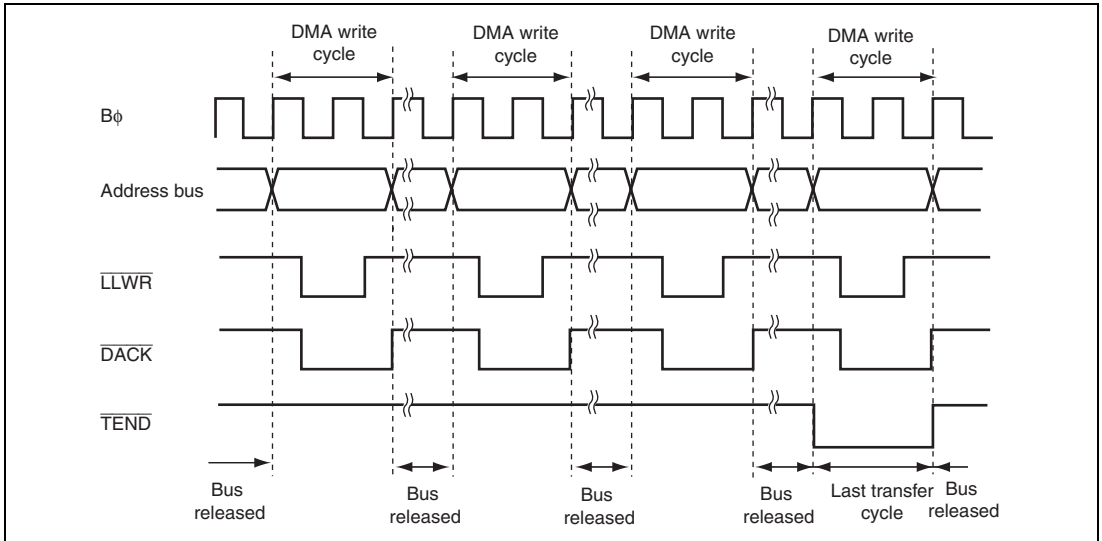


**Figure 9.34 Example of Transfer in Single Address Mode (Byte Read)**

## (2) Single Address Mode (Write and Cycle Stealing)

In single address mode, data of one byte, one word, or one longword is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU or DTC are executed in the bus released cycles.

In figure 9.35, the  $\overline{TEND}$  signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (write).



**Figure 9.35 Example of Transfer in Single Address Mode (Byte Write)**

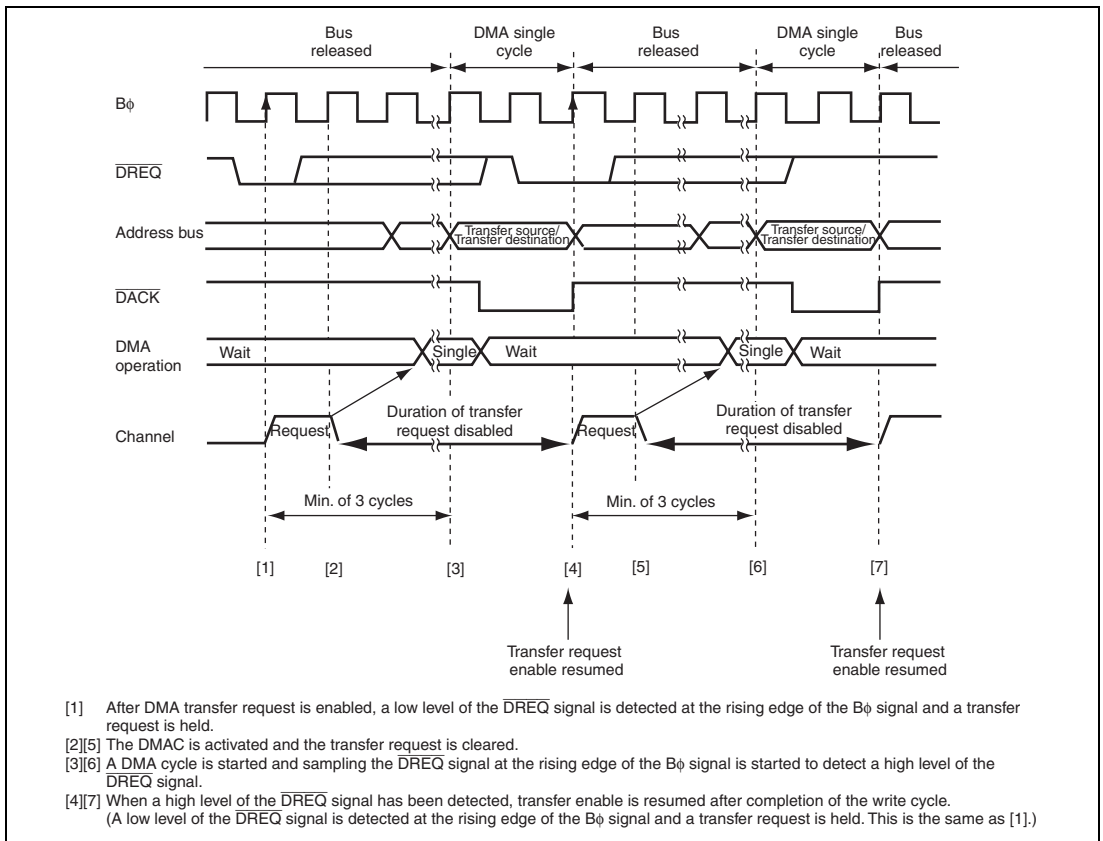


### (3) Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 9.36 shows an example of single address mode activated by the  $\overline{\text{DREQ}}$  signal falling edge.

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the  $\overline{\text{DREQ}}$  signal for falling edge detection. If a high level of the  $\overline{\text{DREQ}}$  signal has been detected until completion of the single cycle, receiving the next transfer request resumes and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



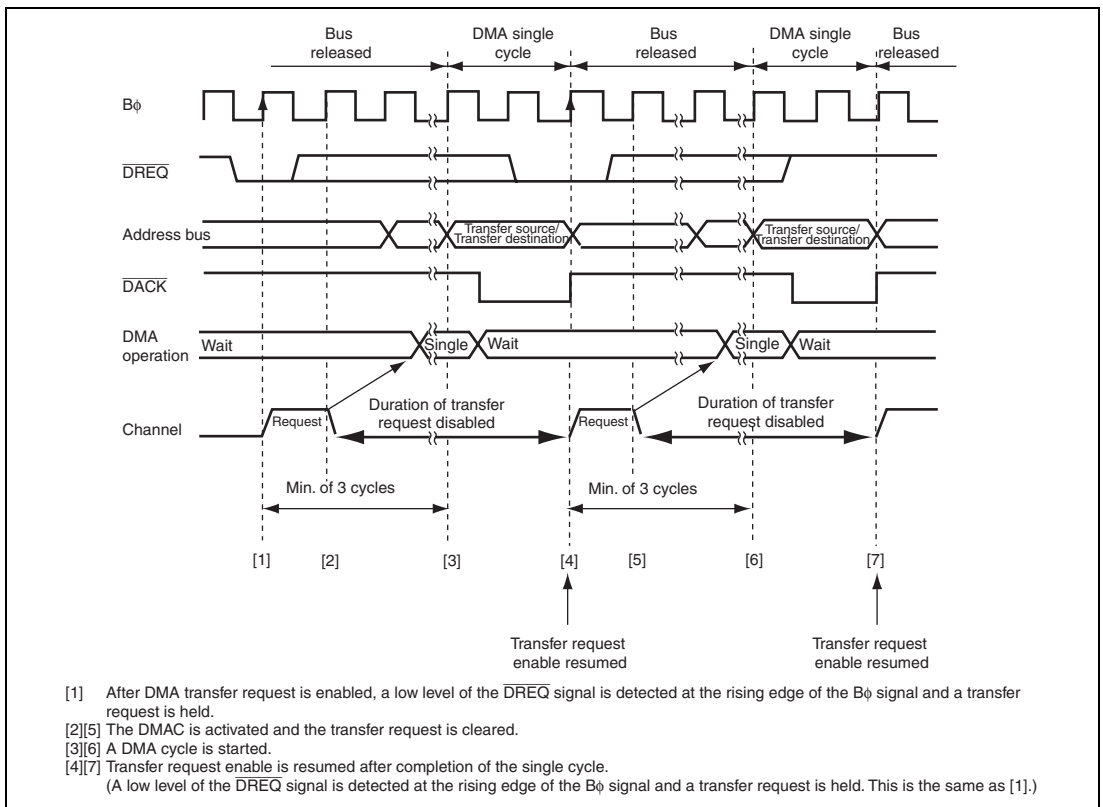
**Figure 9.36 Example of Transfer in Single Address Mode Activated by  $\overline{\text{DREQ}}$  Falling Edge**

#### (4) Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 9.37 shows an example of normal transfer mode activated by the  $\overline{\text{DREQ}}$  signal low level.

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the single cycle and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



**Figure 9.37 Example of Transfer in Single Address Mode Activated by  $\overline{\text{DREQ}}$  Low Level**

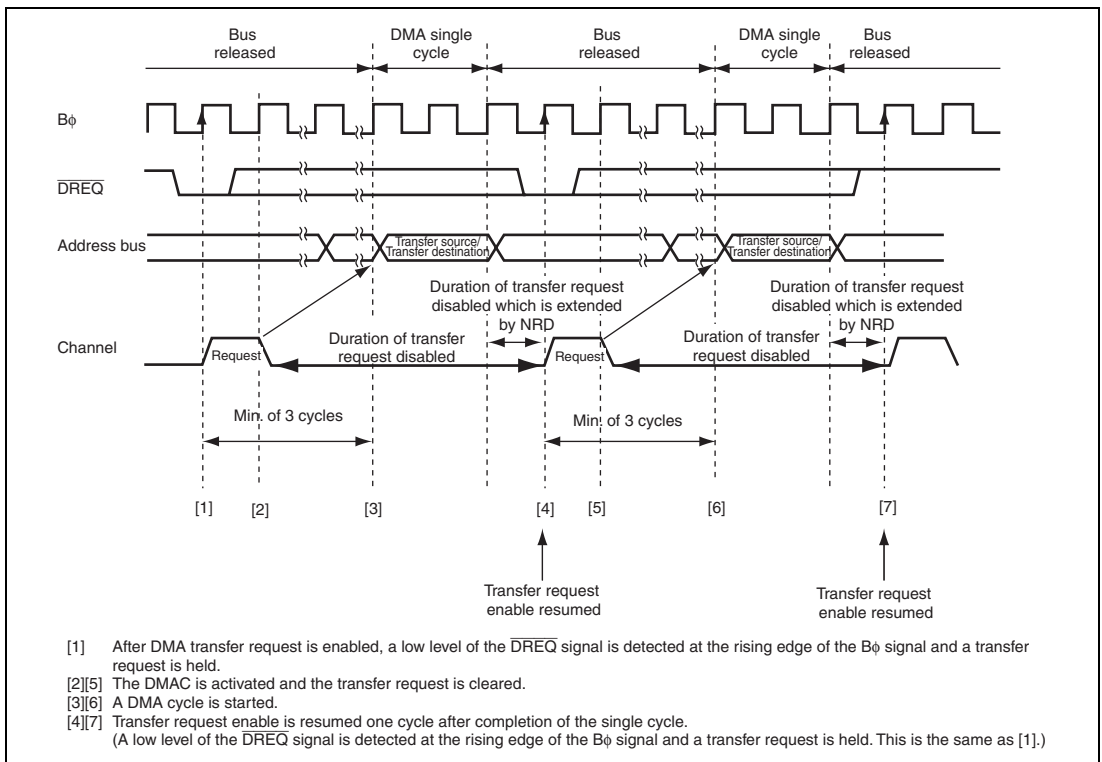
### (5) Activation Timing by $\overline{\text{DREQ}}$ Low Level with $\text{NRD} = 1$

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 9.38 shows an example of single address mode activated by the  $\overline{\text{DREQ}}$  signal low level with  $\text{NRD} = 1$ .

The  $\overline{\text{DREQ}}$  signal is sampled every cycle from the next rising edge of the  $\text{B}\phi$  signal immediately after the DTE bit write cycle.

When a low level of the  $\overline{\text{DREQ}}$  signal is detected while a transfer request by the  $\overline{\text{DREQ}}$  signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after one cycle of the transfer request duration inserted by  $\text{NRD} = 1$  on completion of the single cycle and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



**Figure 9.38 Example of Transfer in Single Address Mode Activated by  $\overline{\text{DREQ}}$  Low Level with  $\text{NRD} = 1$**

## 9.6 DMA Transfer End

Operations on completion of a transfer differ according to the transfer end condition. DMA transfer completion is indicated that the DTE and ACT bits in DMDR are changed from 1 to 0.

### (1) Transfer End by DTCR Change from 1, 2, or 4, to 0

When DTCR is changed from 1, 2, or 4 to 0, a DMA transfer for the channel is completed. The DTE bit in DMDR is cleared to 0 and the DTIF bit in DMDR is set to 1. At this time, when the DTIE bit in DMDR is set to 1, a transfer end interrupt by the transfer counter is requested. When the DTCR value is 0 before the transfer, the transfer is not stopped.

### (2) Transfer End by Transfer Size Error Interrupt

When the following conditions are satisfied while the TSEIE bit in DMDR is set to 1, a transfer size error occurs and a DMA transfer is terminated. At this time, the DTE bit in DMR is cleared to 0 and the ESIF bit in DMDR is set to 1.

- In normal transfer mode and repeat transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the data access size
- In block transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the block size

When the TSEIE bit in DMDR is cleared to 0, data is transferred until the DTCR value reaches 0. A transfer size error is not generated. Operation in each transfer mode is shown below.

- In normal transfer mode and repeat transfer mode, when the DTCR value is less than the data access size, data is transferred in bytes
- In block transfer mode, when the DTCR value is less than the block size, the specified size of data in DTCR is transferred instead of transferring the block size of data. The transfer is performed in bytes.

### **(3) Transfer End by Repeat Size End Interrupt**

In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit in DACR is set to 1, a repeat size end interrupt is requested. When the interrupt is requested to complete DMA transfer, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1. Under this condition, setting the DTE bit to 1 resumes the transfer.

In block transfer mode, when the next transfer is requested after completion of a 1-block size data transfer, a repeat size end interrupt can be requested.

### **(4) Transfer End by Interrupt on Extended Repeat Area Overflow**

When an overflow on the extended repeat area occurs while the extended repeat area is specified and the SARIE or DARIE bit in DACR is set to 1, an interrupt by an extended repeat area overflow is requested. When the interrupt is requested, the DMA transfer is terminated, the DTE bit in DMDR is cleared to 0, and the ESIF bit in DMDR is set to 1.

In dual address mode, even if an interrupt by an extended repeat area overflow occurs during a read cycle, the following write cycle is performed.

In block transfer mode, even if an interrupt by an extended repeat area overflow occurs during a 1-block transfer, the remaining data is transferred. The transfer is not terminated by an extended repeat area overflow interrupt unless the current transfer is complete.

### **(5) Transfer End by Clearing DTE Bit in DMDR**

When the DTE bit in DMDR is cleared to 0 by the CPU, a transfer is completed after the current DMA cycle and a DMA cycle in which the transfer request is accepted are completed.

In block transfer mode, a DMA transfer is completed after 1-block data is transferred.

**(6) Transfer End by NMI Interrupt**

When an NMI interrupt is requested, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR\_0 is set to 1. When an NMI interrupt is requested during a DMA transfer, the transfer is forced to stop. To perform DMA transfer after an NMI interrupt is requested, clear the ERRF bit to 0 and then set the DTE bits for the channels to 1.

The transfer end timings after an NMI interrupt is requested are shown below.

**(a) Normal Transfer Mode and Repeat Transfer Mode**

In dual address mode, a DMA transfer is completed after completion of the write cycle for one transfer unit.

In single address mode, a DMA transfer is completed after completion of the bus cycle for one transfer unit.

**(b) Block Transfer Mode**

A DMA transfer is forced to stop. Since a 1-block size of transfers is not completed, operation is not guaranteed.

In dual address mode, the write cycle corresponding to the read cycle is performed. This is similar to (a) in normal transfer mode.

**(7) Transfer End by Address Error**

When an address error occurs, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR\_0 is set to 1. When an address error occurs during a DMA transfer, the transfer is forced to stop. To perform a DMA transfer after an address error occurs, clear the ERRF bit to 0 and then set the DTE bits for the channels.

The transfer end timing after an address error is the same as that after an NMI interrupt.

**(8) Transfer End by Hardware Standby Mode or Reset**

The DMAC is initialized by a reset and a transition to the hardware standby mode. A DMA transfer is not guaranteed.

## 9.7 Relationship among DMAC and Other Bus Masters

### 9.7.1 CPU Priority Control Function Over DMAC

The CPU priority control function over DMAC can be used according to the CPU priority control register (CPUPCR) setting. For details, see section 6.7, CPU Priority Control Function Over DTC and DMAC.

The priority level of the DMAC is specified by bits DMAP2 to DMAP0 and can be specified for each channel.

The priority level of the CPU is specified by bits CPUP2 to CPUP0. The value of bits CPUP2 to CPUP0 is updated according to the exception handling priority.

If the CPU priority control is enabled by the CPUPCE bit in CPUPCR, when the CPU has priority over the DMAC, a transfer request for the corresponding channel is masked and the transfer is not activated. When another channel has priority over or the same as the CPU, a transfer request is received regardless of the priority between channels and the transfer is activated.

The transfer request masked by the CPU priority control function is suspended. When the transfer channel is given priority over the CPU by changing priority levels of the CPU or channel, the transfer request is received and the transfer is resumed. Writing 0 to the DTE bit clears the suspended transfer request.

When the CPUPCE bit is cleared to 0, it is regarded as the lowest priority.

### 9.7.2 Bus Arbitration among DMAC and Other Bus Masters

When DMA transfer cycles are consecutively performed, bus cycles of other bus masters may be inserted between the transfer cycles. The DMAC can release the bus temporarily to pass the bus to other bus masters.

The consecutive DMA transfer cycles may not be divided according to the transfer mode settings to achieve high-speed access.

The read and write cycles of a DMA transfer are not separated. Refreshing, external bus release, and on-chip bus master (CPU or DTC) cycles are not inserted between the read and write cycles of a DMA transfer.

In block transfer mode and an auto request transfer by burst access, bus cycles of the DMA transfer are consecutively performed. For this duration, since the DMAC has priority over the CPU and DTC, accesses to the external space is suspended (the IBCCS bit in the bus control register 2 (BCR2) is cleared to 0).

When the bus is passed to another channel or an auto request transfer by cycle stealing, bus cycles of the DMAC and on-chip bus master are performed alternatively.

When the arbitration function among the DMAC and on-chip bus masters is enabled by setting the IBCCS bit in BCR2, the bus is used alternatively except the bus cycles which are not separated. For details, see section 8, Bus Controller (BSC).

A conflict may occur between external space access of the DMAC and an external bus release cycle. Even if a burst or block transfer is performed by the DMAC, the transfer is stopped temporarily and a cycle of external bus release is inserted by the BSC according to the external bus priority (when the CPU external access and the DTC external access do not have priority over a DMAC transfer, the transfers are not operated until the DMAC releases the bus).

In dual address mode, the DMAC releases the external bus after the external space write cycle. Since the read and write cycles are not separated, the bus is not released.

An internal space (on-chip memory and internal I/O registers) access of the DMAC and an external bus release cycle may be performed at the same time.



## 9.8 Interrupt Sources

The DMAC interrupt sources are a transfer end interrupt by the transfer counter and a transfer escape end interrupt which is generated when a transfer is terminated before the transfer counter reaches 0. Table 9.7 shows interrupt sources and priority.

**Table 9.7 Interrupt Sources and Priority**

Abbr.	Interrupt Sources	Priority
DMTEND0	Transfer end interrupt by channel 0 transfer counter	High
DMTEND1	Transfer end interrupt by channel 1 transfer counter	
DMEEND0	Interrupt by channel 0 transfer size error	
	Interrupt by channel 0 repeat size end	
	Interrupt by channel 0 extended repeat area overflow on source address	
	Interrupt by channel 0 extended repeat area overflow on destination address	
DMEEND1	Interrupt by channel 1 transfer size error	
	Interrupt by channel 1 repeat size end	
	Interrupt by channel 1 extended repeat area overflow on source address	
	Interrupt by channel 1 extended repeat area overflow on destination address	

Each interrupt is enabled or disabled by the DTIE and ESIE bits in DMDR for the corresponding channel. A DMTEND interrupt is generated by the combination of the DTIF and DTIE bits in DMDR. A DMEEND interrupt is generated by the combination of the ESIF and ESIE bits in DMDR. The DMEEND interrupt sources are not distinguished. The priority among channels are decided by the interrupt controller and it is shown in table 9.7. For details, see section 6, Interrupt Controller.

Each interrupt source is specified by the interrupt enable bit in the register for the corresponding channel. A transfer end interrupt by the transfer counter, a transfer size error interrupt, a repeat size end interrupt, an interrupt by an extended repeat area overflow on the source address, and an interrupt by an extended repeat area overflow on the destination address are enabled or disabled by the DTIE bit in DMDR, the TSEIE bit in DMDR, the RPTIE bit in DACR, SARIE bit in DACR, and the DARIE bit in DACR, respectively.

A transfer end interrupt by the transfer counter is generated when the DTIF bit in DMDR is set to 1. The DTIF bit is set to 1 when DTCR becomes 0 by a transfer while the DTIE bit in DMDR is set to 1.

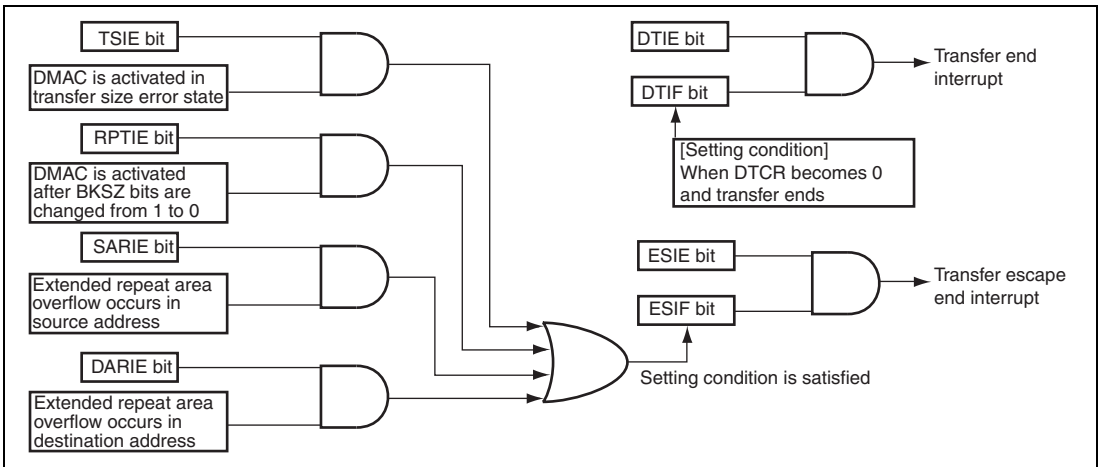
An interrupt other than the transfer end interrupt by the transfer counter is generated when the ESIF bit in DMDR is set to 1. The ESIF bit is set to 1 when the conditions are satisfied by a transfer while the enable bit is set to 1.

A transfer size error interrupt is generated when the next transfer cannot be performed because the DTCR value is less than the data access size, meaning that the data access size of transfers cannot be performed. In block transfer mode, the block size is compared with the DTCR value for transfer error decision.

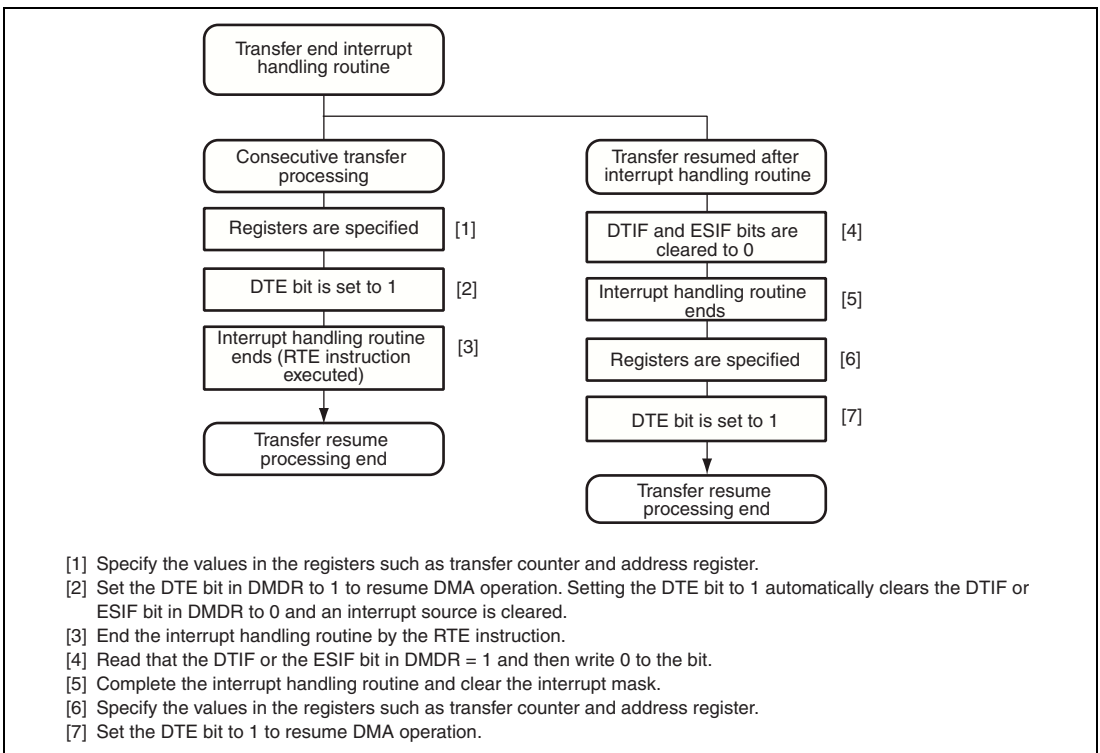
A repeat size end interrupt is generated when the next transfer is requested after completion of the repeat size of transfers in repeat transfer mode. Even when the repeat area is not specified in the address register, the transfer can be stopped periodically according to the repeat size. At this time, when a transfer end interrupt by the transfer counter is generated, the ESIF bit is set to 1.

An interrupt by an extended repeat area overflow on the source and destination addresses is generated when the address exceeds the extended repeat area (overflow). At this time, when a transfer end interrupt by the transfer counter, the ESIF bit is set to 1.

Figure 9.39 is a block diagram of interrupts and interrupt flags. To clear an interrupt, clear the DTIF or ESIF bit in DMDR to 0 in the interrupt handling routine or continue the transfer by setting the DTE bit in DMDR after setting the register. Figure 9.40 shows procedure to resume the transfer by clearing a interrupt.



**Figure 9.39** Interrupt and Interrupt Sources



**Figure 9.40** Procedure Example of Resuming Transfer by Clearing Interrupt Source

## 9.9 Usage Notes

### 9.9.1 DMAC Register Access During Operation

Except for clearing the DTE bit in DMDR, the settings for channels being transferred (including waiting state) must not be changed. The register settings must be changed during the transfer prohibited state.

### 9.9.2 Settings of Module Stop Function

The DMAC operation can be enabled or disabled by the module stop control register. The DMAC is enabled by the initial value.

Setting bit MSTPA13 in MSTPCRA stops the clock supplied to the DMAC and the DMAC enters the module stop state. However, when a transfer for a channel is enabled or when an interrupt is being requested, bit MSTPA13 cannot be set to 1. Clear the DTE bit to 0, clear the DTIF or DTIE bit in DMDR to 0, and then set bit MSTPA13.

When the clock is stopped, the DMAC registers cannot be accessed. However, the following register settings are valid in the module stop state. Disable them before entering the module stop state, if necessary.

- TENDE bit in DMDR is 1 (the TEND signal output enabled)
- DACKE bit in DMDR is 1 (the DACK signal output enabled)

### 9.9.3 Activation by $\overline{\text{DREQ}}$ Falling Edge

The  $\overline{\text{DREQ}}$  falling edge detection is synchronized with the DMAC internal operation.

- A. Activation request waiting state: Waiting for detecting the  $\overline{\text{DREQ}}$  low level. A transition to 2. is made.
- B. Transfer waiting state: Waiting for a DMAC transfer. A transition to 3. is made.
- C. Transfer prohibited state: Waiting for detecting the  $\overline{\text{DREQ}}$  high level. A transition to 1. is made.

After a DMAC transfer enabled, a transition to 1. is made. Therefore, the  $\overline{\text{DREQ}}$  signal is sampled by low level detection at the first activation after a DMAC transfer enabled.

#### 9.9.4 Acceptation of Activation Source

At the beginning of an activation source reception, a low level is detected regardless of the setting of  $\overline{\text{DREQ}}$  falling edge or low level detection. Therefore, if the  $\overline{\text{DREQ}}$  signal is driven low before setting DMDR, the low level is received as a transfer request.

When the DMAC is activated, clear the  $\overline{\text{DREQ}}$  signal of the previous transfer.



## Section 10 Data Transfer Controller (DTC)

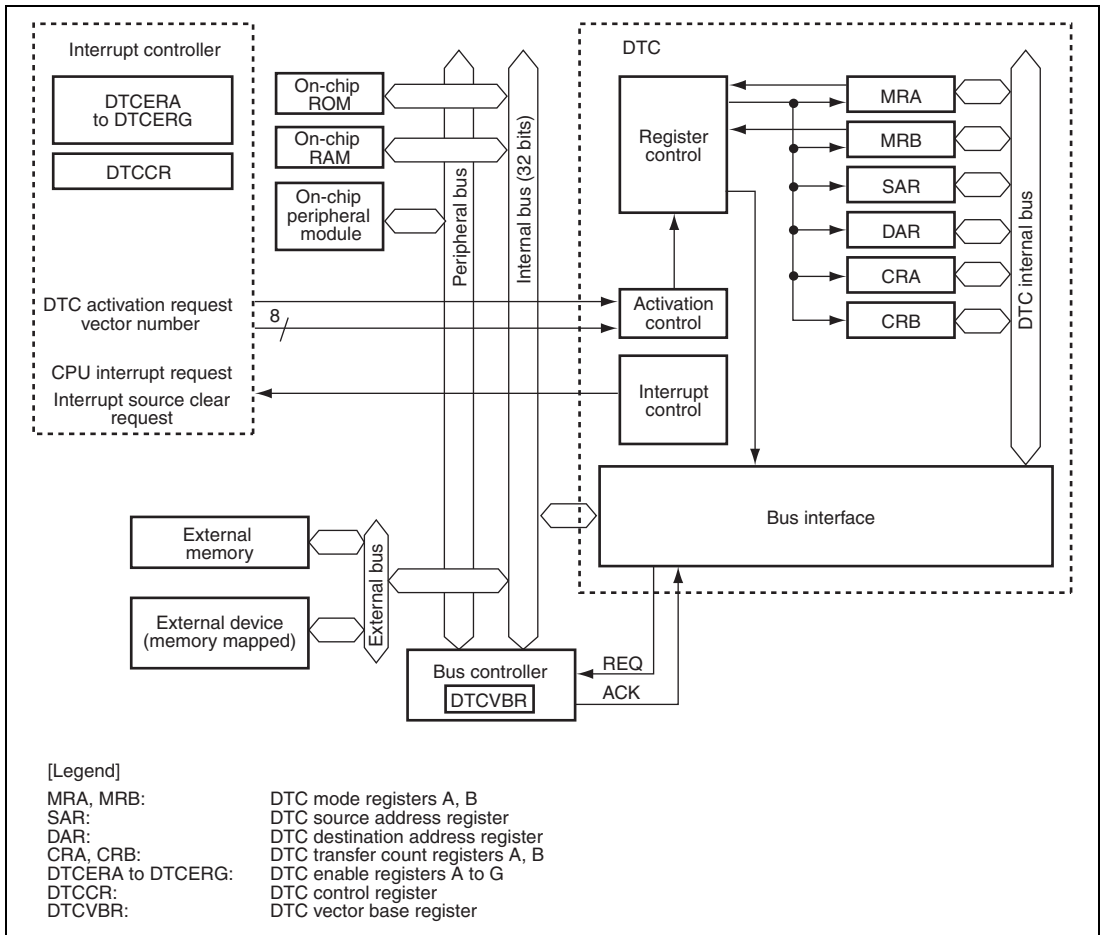
This LSI includes a data transfer controller (DTC). The DTC can be activated to transfer data by an interrupt request.

### 10.1 Features

- Transfer possible over any number of channels:
  - Multiple data transfer enabled for one activation source (chain transfer)
  - Chain transfer specifiable after data transfer (when the counter is 0)
- Three transfer modes
  - Normal/repeat/block transfer modes selectable
  - Transfer source and destination addresses can be selected from increment/decrement/fixed
- Short address mode or full address mode selectable
  - Short address mode
    - Transfer information is located on a 3-longword boundary
    - The transfer source and destination addresses can be specified by 24 bits to select a 16-Mbyte address space directly
  - Full address mode
    - Transfer information is located on a 4-longword boundary
    - The transfer source and destination addresses can be specified by 32 bits to select a 4-Gbyte address space directly
- Size of data for data transfer can be specified as byte, word, or longword
  - The bus cycle is divided if an odd address is specified for a word or longword transfer.
  - The bus cycle is divided if address  $4n + 2$  is specified for a longword transfer.
- A CPU interrupt can be requested for the interrupt that activated the DTC
  - A CPU interrupt can be requested after one data transfer completion
  - A CPU interrupt can be requested after the specified data transfer completion
- Read skip of the transfer information specifiable
- Writeback skip executed for the fixed transfer source and destination addresses
- Module stop state specifiable

Figure 10.1 shows a block diagram of the DTC. The DTC transfer information can be allocated to the data area\*. When the transfer information is allocated to the on-chip RAM, a 32-bit bus connects the DTC to the on-chip RAM, enabling 32-bit/1-state reading and writing of the DTC transfer information.

Note: \* When the transfer information is stored in the on-chip RAM, the RAME bit in SYSCR must be set to 1.



**Figure 10.1 Block Diagram of DTC**



## 10.2 Register Descriptions

DTC has the following registers.

- DTC mode register A (MRA)
- DTC mode register B (MRB)
- DTC source address register (SAR)
- DTC destination address register (DAR)
- DTC transfer count register A (CRA)
- DTC transfer count register B (CRB)

These six registers MRA, MRB, SAR, DAR, CRA, and CRB cannot be directly accessed by the CPU. The contents of these registers are stored in the data area as transfer information. When a DTC activation request occurs, the DTC reads a start address of transfer information that is stored in the data area according to the vector address, reads the transfer information, and transfers data. After the data transfer, it writes a set of updated transfer information back to the data area.

- DTC enable registers A to G (DTCERA to DTCERG)
- DTC control register (DTCCR)
- DTC vector base register (DTCVBR)

### 10.2.1 DTC Mode Register A (MRA)

MRA selects DTC operating mode. MRA cannot be accessed directly by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name	MD1	MD0	Sz1	Sz0	SM1	SM0	—	—
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	—	—	—	—	—	—	—	—

Bit	Bit Name	Initial Value	R/W	Description
7	MD1	Undefined	—	DTC Mode 1 and 0
6	MD0	Undefined	—	Specify DTC transfer mode. 00: Normal mode 01: Repeat mode 10: Block transfer mode 11: Setting prohibited
5	Sz1	Undefined	—	DTC Data Transfer Size 1 and 0
4	Sz0	Undefined	—	Specify the size of data to be transferred. 00: Byte-size transfer 01: Word-size transfer 10: Longword-size transfer 11: Setting prohibited
3	SM1	Undefined	—	Source Address Mode 1 and 0
2	SM0	Undefined	—	Specify an SAR operation after a data transfer. 0x: SAR is fixed (SAR writeback is skipped) 10: SAR is incremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10) 11: SAR is decremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10)
1, 0	<input type="checkbox"/>	Undefined	<input type="checkbox"/>	Reserved The write value should always be 0.

[Legend]

X: Don't care

## 10.2.2 DTC Mode Register B (MRB)

MRB selects DTC operating mode. MRB cannot be accessed directly by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name	CHNE	CHNS	DISEL	DTS	DM1	DM0	—	—
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	—	—	—	—	—	—	—	—

Bit	Bit Name	Initial Value	R/W	Description
7	CHNE	Undefined	—	<p>DTC Chain Transfer Enable</p> <p>Specifies the chain transfer. For details, see section 10.5.7, Chain Transfer. The chain transfer condition is selected by the CHNS bit.</p> <p>0: Disables the chain transfer 1: Enables the chain transfer</p>
6	CHNS	Undefined	—	<p>DTC Chain Transfer Select</p> <p>Specifies the chain transfer condition. If the following transfer is a chain transfer, the completion check of the specified transfer count is not performed and activation source flag or DTCER is not cleared.</p> <p>0: Chain transfer every time 1: Chain transfer only when transfer counter = 0</p>
5	DISEL	Undefined	—	<p>DTC Interrupt Select</p> <p>When this bit is set to 1, a CPU interrupt request is generated every time after a data transfer ends. When this bit is set to 0, a CPU interrupt request is only generated when the specified number of data transfer ends.</p>
4	DTS	Undefined	—	<p>DTC Transfer Mode Select</p> <p>Specifies either the source or destination as repeat or block area during repeat or block transfer mode.</p> <p>0: Specifies the destination as repeat or block area 1: Specifies the source as repeat or block area</p>

Bit	Bit Name	Initial Value	R/W	Description
3	DM1	Undefined	—	Destination Address Mode 1 and 0
2	DM0	Undefined	—	Specify a DAR operation after a data transfer. 0X: DAR is fixed (DAR writeback is skipped) 10: DAR is incremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10) 11: SAR is decremented after a transfer (by 1 when Sz1 and Sz0 = B'00; by 2 when Sz1 and Sz0 = B'01; by 4 when Sz1 and Sz0 = B'10)
1, 0	—	Undefined	—	Reserved The write value should always be 0.

[Legend]

X: Don't care

### 10.2.3 DTC Source Address Register (SAR)

SAR is a 32-bit register that designates the source address of data to be transferred by the DTC.

In full address mode, 32 bits of SAR are valid. In short address mode, the lower 24 bits of SAR is valid and bits 31 to 24 are ignored. At this time, the upper eight bits are filled with the value of bit 23.

If a word or longword access is performed while an odd address is specified in SAR or if a longword access is performed while address  $4n + 2$  is specified in SAR, the bus cycle is divided into multiple cycles to transfer data. For details, see section 10.5.1, Bus Cycle Division.

SAR cannot be accessed directly from the CPU.

### 10.2.4 DTC Destination Address Register (DAR)

DAR is a 32-bit register that designates the destination address of data to be transferred by the DTC.

In full address mode, 32 bits of DAR are valid. In short address mode, the lower 24 bits of DAR is valid and bits 31 to 24 are ignored. At this time, the upper eight bits are filled with the value of bit 23.

If a word or longword access is performed while an odd address is specified in DAR or if a longword access is performed while address  $4n + 2$  is specified in DAR, the bus cycle is divided into multiple cycles to transfer data. For details, see section 10.5.1, Bus Cycle Division.

DAR cannot be accessed directly from the CPU.

### 10.2.5 DTC Transfer Count Register A (CRA)

CRA is a 16-bit register that designates the number of times data is to be transferred by the DTC.

In normal transfer mode, CRA functions as a 16-bit transfer counter (1 to 65,536). It is decremented by 1 every time data is transferred, and bit DTCEn (n = 15 to 0) corresponding to the activation source is cleared and then an interrupt is requested to the CPU when the count reaches H'0000. The transfer count is 1 when CRA = H'0001, 65,535 when CRA = H'FFFF, and 65,536 when CRA = H'0000.

In repeat transfer mode, CRA is divided into two parts: the upper eight bits (CRAH) and the lower eight 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 to CRAL when the count reaches H'00. The transfer count is 1 when CRAH = CRAL = H'01, 255 when CRAH = CRAL = H'FF, and 256 when CRAH = CRAL = H'00.

In block transfer mode, CRA is divided into two parts: the upper eight bits (CRAH) and the lower eight bits (CRAL). CRAH holds the block size while CRAL functions as an 8-bit block-size counter (1 to 256 for byte, word, or longword). CRAL is decremented by 1 every time a byte (word or longword) data is transferred, and the contents of CRAH are sent to CRAL when the count reaches H'00. The block size is 1 byte (word or longword) when CRAH = CRAL = H'01, 255 bytes (words or longwords) when CRAH = CRAL = H'FF, and 256 bytes (words or longwords) when CRAH = CRAL = H'00.

CRA cannot be accessed directly from the CPU.

### 10.2.6 DTC Transfer Count Register B (CRB)

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 65,536) that is decremented by 1 every time data is transferred, and bit DTCEn (n = 15 to 0) corresponding to the activation source is cleared and then an interrupt is requested to the CPU when the count reaches H'0000. The transfer count is 1 when CRB = H'0001, 65,535 when CRB = H'FFFF, and 65,536 when CRB = H'0000.

CRB is not available in normal and repeat modes and cannot be accessed directly by the CPU.

## 10.2.7 DTC enable registers A to H (DTCERA to DTCERH)

DTCER, which is comprised of eight registers, DTCERA to DTCERH, is a register that specifies DTC activation interrupt sources. The correspondence between interrupt sources and DTCE bits is shown in table 10.1. Use bit manipulation instructions such as BSET and BCLR to read or write a DTCE bit. If all interrupts are masked, multiple activation sources can be set at one time (only at the initial setting) by writing data after executing a dummy read on the relevant register.

Bit	15	14	13	12	11	10	9	8
Bit Name	DTCE15	DTCE14	DTCE13	DTCE12	DTCE11	DTCE10	DTCE9	DTCE8
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	7	6	5	4	3	2	1	0
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
15	DTCE15	0	R/W	DTC Activation Enable 15 to 0
14	DTCE14	0	R/W	Setting this bit to 1 specifies a relevant interrupt source to a DTC activation source.
13	DTCE13	0	R/W	
12	DTCE12	0	R/W	[Clearing conditions]
11	DTCE11	0	R/W	• When writing 0 to the bit to be cleared after reading 1
10	DTCE10	0	R/W	• When the DISEL bit is 1 and the data transfer has ended
9	DTCE9	0	R/W	
8	DTCE8	0	R/W	• When the specified number of transfers have ended
7	DTCE7	0	R/W	These bits are not cleared when the DISEL bit is 0 and the specified number of transfers have not ended
6	DTCE6	0	R/W	
5	DTCE5	0	R/W	
4	DTCE4	0	R/W	
3	DTCE3	0	R/W	
2	DTCE2	0	R/W	
1	DTCE1	0	R/W	
0	DTCE0	0	R/W	

### 10.2.8 DTC Control Register (DTCCR)

DTCCR specifies transfer information read skip.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	RRS	RCHNE	—	—	ERR
Initial Value	0	0	0	0	0	0	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.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
4	RRS	0	R/W	DTC Transfer Information Read Skip Enable Controls the vector address read and transfer information read. A DTC vector number is always compared with the vector number for the previous activation. If the vector numbers match and this bit is set to 1, the DTC data transfer is started without reading a vector address and transfer information. If the previous DTC activation is a chain transfer, the vector address read and transfer information read are always performed. 0: Transfer read skip is not performed. 1: Transfer read skip is performed when the vector numbers match.
3	RCHNE	0	R/W	Chain Transfer Enable After DTC Repeat Transfer Enables/disables the chain transfer while transfer counter (CRAL) is 0 in repeat transfer mode. In repeat transfer mode, the CRAH value is written to CRAL when CRAL is 0. Accordingly, chain transfer may not occur when CRAL is 0. If this bit is set to 1, the chain transfer is enabled when CRAH is written to CRAL. 0: Disables the chain transfer after repeat transfer 1: Enables the chain transfer after repeat transfer



Bit	Bit Name	Initial Value	R/W	Description
2, 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	ERR	0	R/(W)*	Transfer Stop Flag Indicates that an address error or an NMI interrupt occurs. If an address error or an NMI interrupt occurs, the DTC stops. 0: No interrupt occurs 1: An interrupt occurs [Clearing condition] <ul style="list-style-type: none"> <li>When writing 0 after reading 1</li> </ul>

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

### 10.2.9 DTC Vector Base Register (DTCVBR)

DTCVBR is a 32-bit register that specifies the base address for vector table address calculation. Bits 31 to 28 and bits 11 to 0 are fixed 0 and cannot be written to. The initial value of DTCVBR is H'00000000.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Bit Name																
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/W	R/W	R/W	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
Bit Name																
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	R	R	R	R	R	R	R	R	R	R	R

### 10.3 Activation Sources

The DTC is activated by an interrupt request. The interrupt source is selected by DTCER. A DTC activation source can be selected by setting the corresponding bit in DTCER; the CPU interrupt source can be selected by clearing the corresponding bit in DTCER. At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the activation source interrupt flag or corresponding DTCER bit is cleared.

## 10.4 Location of Transfer Information and DTC Vector Table

Locate the transfer information in the data area. The start address of transfer information should be located at the address that is a multiple of four ( $4n$ ). Otherwise, the lower two bits are ignored during access ([1:0] = B'00.) Transfer information can be located in either short address mode (three longwords) or full address mode (four longwords). The DTCMD bit in SYSCR specifies either short address mode (DTCMD = 1) or full address mode (DTCMD = 0). For details, see section 3.2.2, System Control Register (SYSCR). Transfer information located in the data area is shown in figure 10.2

The DTC reads the start address of transfer information from the vector table according to the activation source, and then reads the transfer information from the start address. Figure 10.3 shows correspondences between the DTC vector address and transfer information.

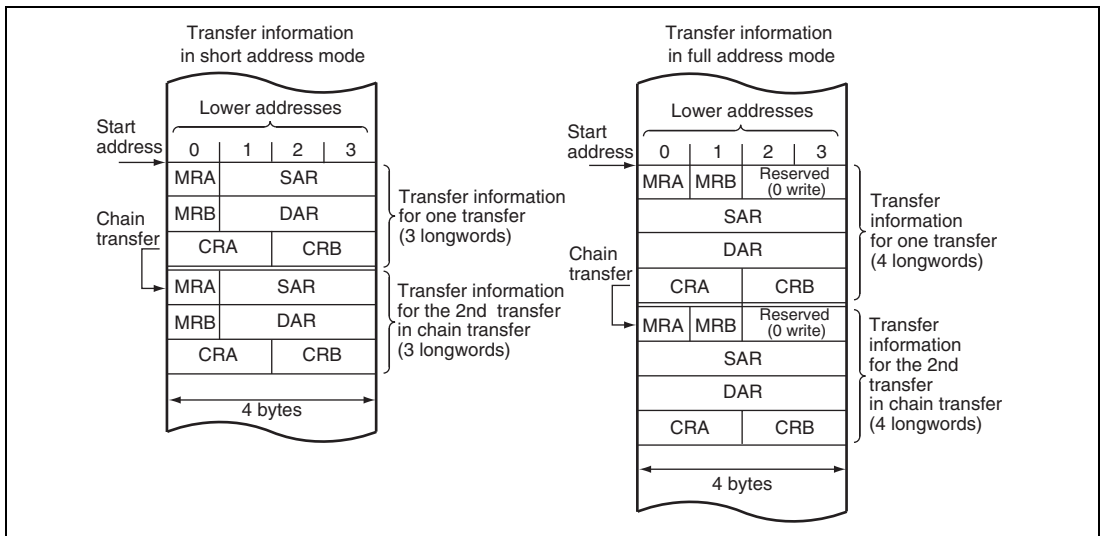
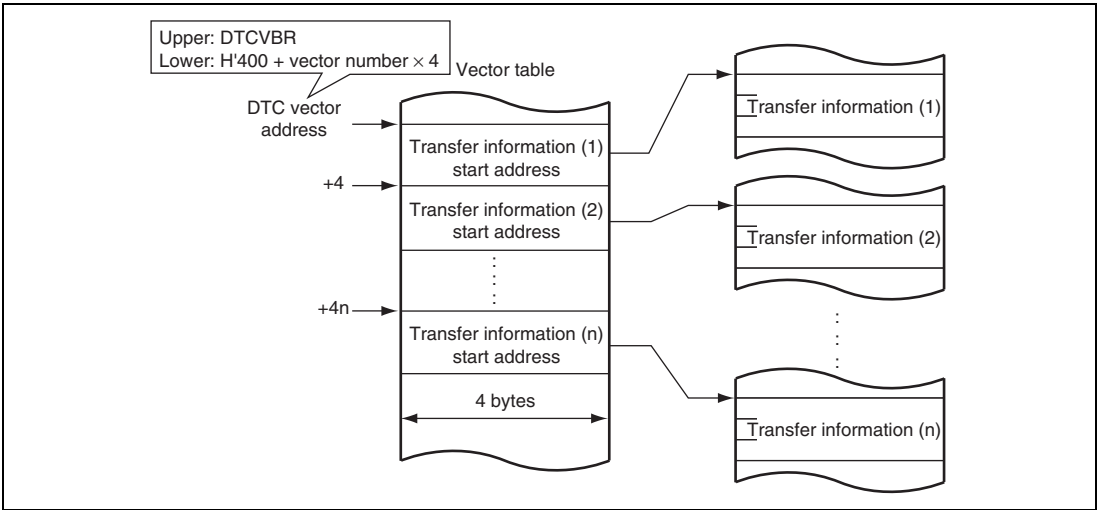


Figure 10.2 Transfer Information on Data Area



**Figure 10.3 Correspondence between DTC Vector Address and Transfer Information**



Origin of Activation Source	Activation Source	Vector Number	DTC Vector Address Offset	DTCE*	Priority
TPU_4	TGI4A	106	H'5A8	DTCEB1	High
	TGI4B	107	H'5AC	DTCEB0	
TPU_5	TGI5A	110	H'5B8	DTCEC15	↑ ↓ Low
	TGI5B	111	H'5BC	DTCEC14	
TMR_0	CMI0A	116	H'5D0	DTCEC13	
	CMI0B	117	H'5D4	DTCEC12	
TMR_1	CMI1A	119	H'5DC	DTCEC11	
	CMI1B	120	H'5E0	DTCEC10	
TMR_2	CMI2A	122	H'5E8	DTCEC9	
	CMI2B	123	H'5EC	DTCEC8	
TMR_3	CMI3A	125	H'5F4	DTCEC7	
	CMI3B	126	H'5F8	DTCEC6	
DMAC	DMTEND0	128	H'600	DTCEC5	
	DMTEND1	129	H'604	DTCEC4	
DMAC	DMEEND0	136	H'620	DTCED13	
	DMEEND1	137	H'624	DTCED12	
SCI_0	RXI0	145	H'644	DTCED5	
	TXI0	146	H'648	DTCED4	
SCI_1	RXI1	149	H'654	DTCED3	
	TXI1	150	H'658	DTCED2	
SCI_2	RXI2	153	H'664	DTCED1	
	TXI2	154	H'668	DTCED0	
SCI_3	RXI3	157	H'674	DTCEE15	
	TXI3	158	H'678	DTCEE14	
SCI_4	RXI4	161	H'684	DTCEE13	
	TXI4	162	H'688	DTCEE12	Low

Origin of Activation Source	Activation Source	Vector Number	DTC Vector Address Offset	DTCE*	Priority
TMR_4	CMI4A	200	H'720	DTCEF3	High
	CMI4B	201	H'724	DTCEF2	
TMR_5	CMI5A	202	H'72C	DTCEF1	↑ ↓ Low
	CMI5B	203	H'730	DTCEF0	
TMR_6	CMI6A	204	H'738	DTCEG15	
	CMI6B	205	H'73C	DTCEG14	
TMR_7	CMI7A	206	H'744	DTCEG13	
	CMI7B	207	H'748	DTCEG12	

Note: \* The DTCE bits with no corresponding interrupt are reserved, and the write value should always be 0. To leave software standby mode or all-module-clock-stop mode with an interrupt, write 0 to the corresponding DTCE bit.

## 10.5 Operation

The DTC stores transfer information in the data area. When activated, the DTC reads transfer information that is stored in the data area and transfers data on the basis of that transfer information. After the data transfer, it writes updated transfer information back to the data area. Since transfer information is in the data area, it is possible to transfer data over any required number of channels. There are three transfer modes: normal, repeat, and block.

The DTC specifies the source address and destination address in SAR and DAR, respectively. After a transfer, SAR and DAR are incremented, decremented, or fixed independently.

Table 10.2 shows the DTC transfer modes.

**Table 10.2 DTC Transfer Modes**

<b>Transfer Mode</b>	<b>Size of Data Transferred at One Transfer Request</b>	<b>Memory Address Increment or Decrement</b>	<b>Transfer Count</b>
Normal	1 byte/word/longword	Incremented/decremented by 1, 2, or 4, or fixed	1 to 65536
Repeat* <sup>1</sup>	1 byte/word/longword	Incremented/decremented by 1, 2, or 4, or fixed	1 to 256* <sup>3</sup>
Block* <sup>2</sup>	Block size specified by CRAH (1 to 256 bytes/words/longwords)	Incremented/decremented by 1, 2, or 4, or fixed	1 to 65536

Notes: 1. Either source or destination is specified to repeat area.  
 2. Either source or destination is specified to block area.  
 3. After transfer of the specified transfer count, initial state is recovered to continue the operation.

Setting the CHNE bit in MRB to 1 makes it possible to perform a number of transfers with a single activation (chain transfer). Setting the CHNS bit in MRB to 1 can also be made to have chain transfer performed only when the transfer counter value is 0.

Figure 10.4 shows a flowchart of DTC operation, and table 10.3 summarizes the chain transfer conditions (combinations for performing the second and third transfers are omitted).

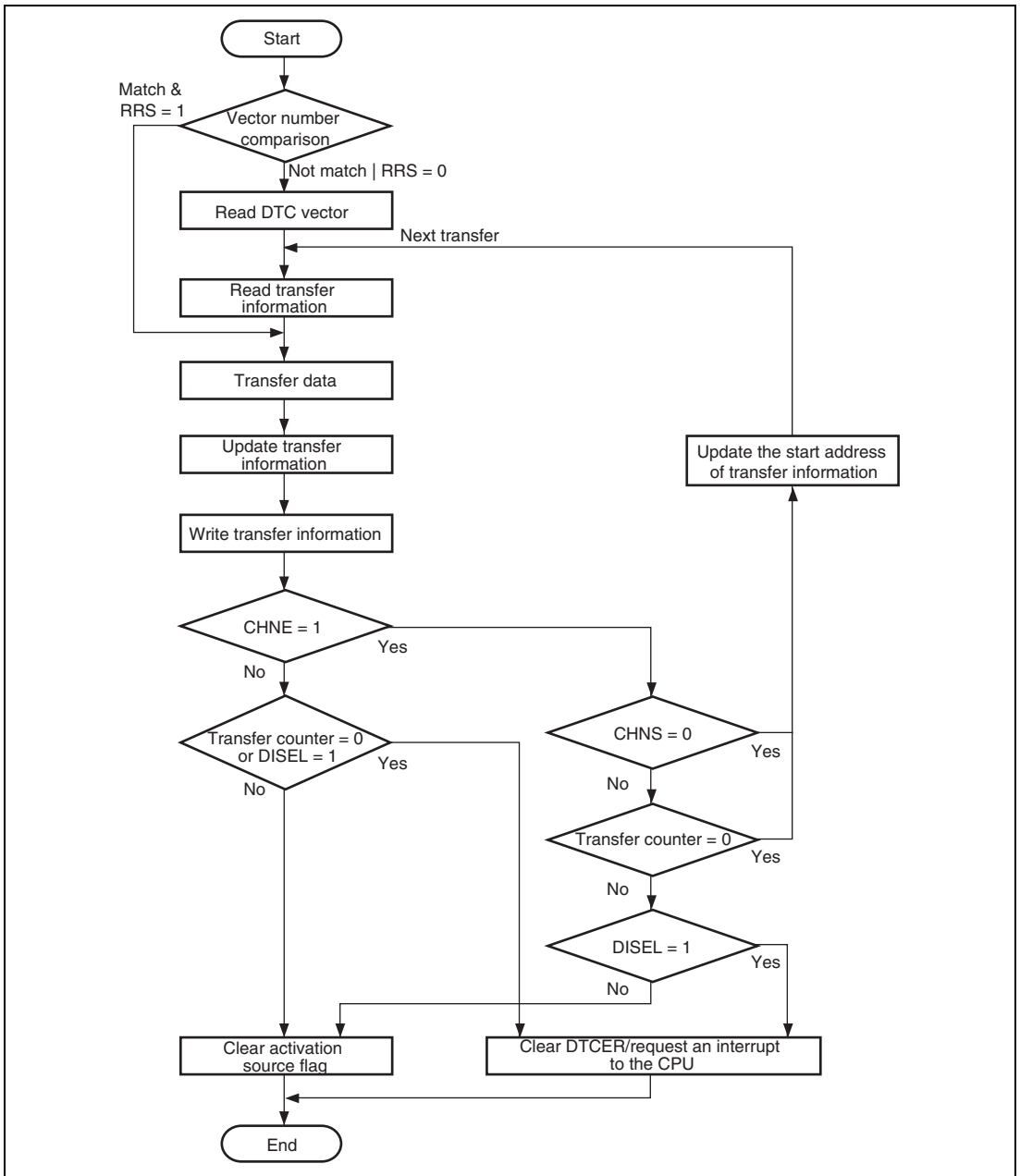


Figure 10.4 Flowchart of DTC Operation



**Table 10.3 Chain Transfer Conditions**

1st Transfer				2nd Transfer				DTC Transfer
CHNE	CHNS	DISEL	Transfer Counter* <sup>1</sup>	CHNE	CHNS	DISEL	Transfer Counter* <sup>1</sup>	
0	—	0	Not 0	—	—	—	—	Ends at 1st transfer
0	—	0	0* <sup>2</sup>	—	—	—	—	Ends at 1st transfer
0	—	1	—	—	—	—	—	Interrupt request to CPU
1	0	—	—	0	—	0	Not 0	Ends at 2nd transfer
				0	—	0	0* <sup>2</sup>	Ends at 2nd transfer
				0	—	1	—	Interrupt request to CPU
1	1	0	Not 0	—	—	—	—	Ends at 1st transfer
1	1	—	0* <sup>2</sup>	0	—	0	Not 0	Ends at 2nd transfer
				0	—	0	0* <sup>2</sup>	Ends at 2nd transfer
				0	—	1	—	Interrupt request to CPU
1	1	1	Not 0	—	—	—	—	Ends at 1st transfer
								Interrupt request to CPU

Notes: 1. CRA in normal mode transfer, CRAL in repeat transfer mode, or CRB in block transfer mode

2. When the contents of the CRAH is written to the CRAL in repeat transfer mode

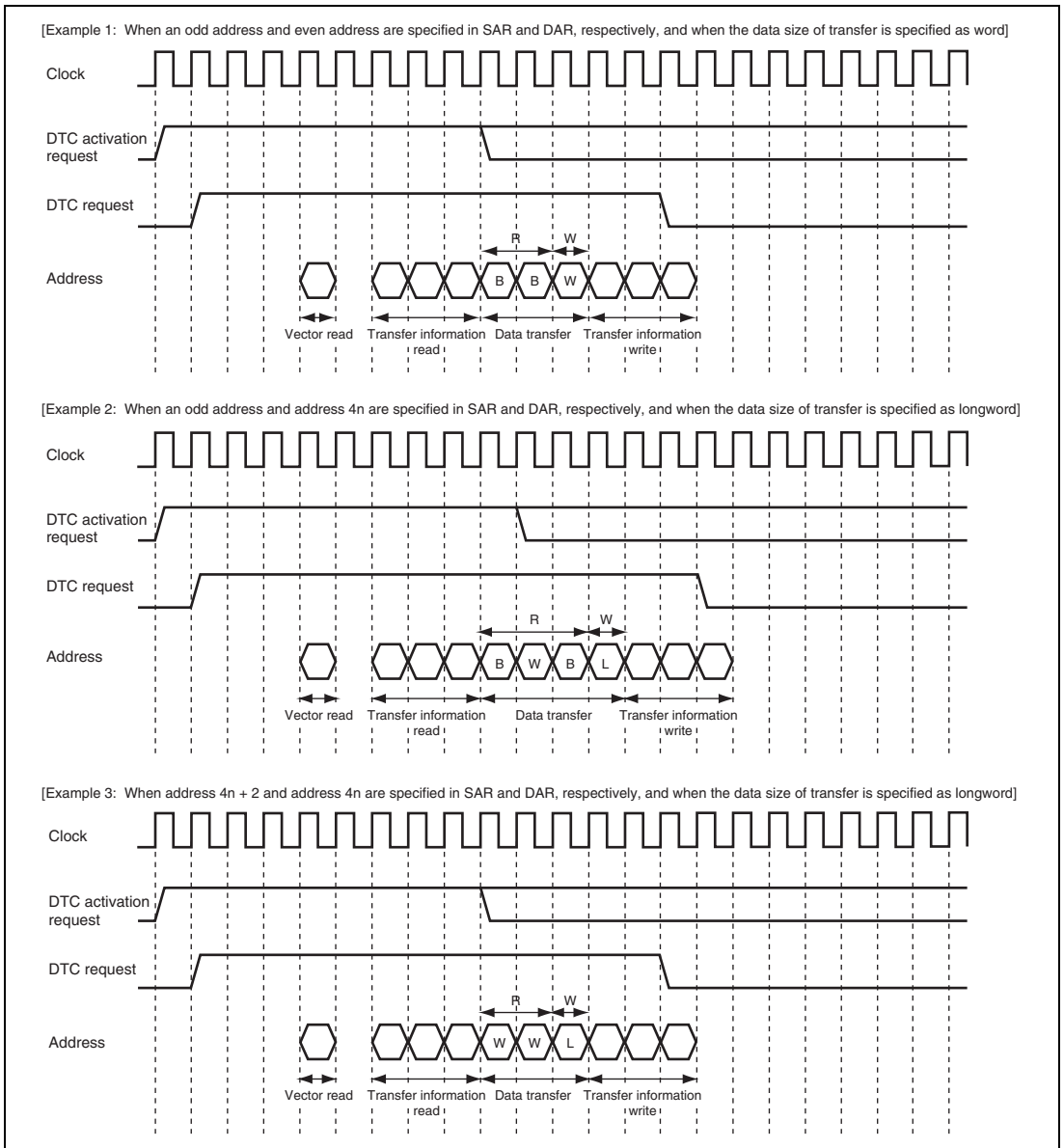
### 10.5.1 Bus Cycle Division

When the transfer data size is word and the SAR and DAR values are not a multiple of 2, the bus cycle is divided and the transfer data is read from or written to in bytes. Similarly, when the transfer data size is longword and the SAR and DAR values are not a multiple of 4, the bus cycle is divided and the transfer data is read from or written to in words.

Table 10.4 shows the relationship among, SAR, DAR, transfer data size, bus cycle divisions, and access data size. Figure 10.5 shows the bus cycle division example.

**Table 10.4 Number of Bus Cycle Divisions and Access Size**

SAR and DAR Values	Specified Data Size		
	Byte (B)	Word (W)	Longword (LW)
Address 4n	1 (B)	1 (W)	1 (LW)
Address 2n + 1	1 (B)	2 (B-B)	3 (B-W-B)
Address 4n + 2	1 (B)	1 (W)	2 (W-W)

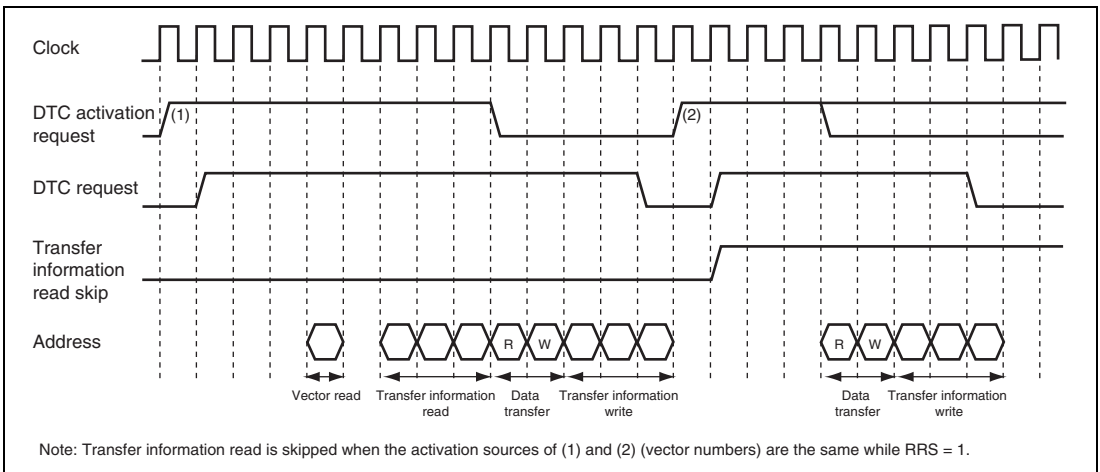


**Figure 10.5 Bus Cycle Division Example**

## 10.5.2 Transfer Information Read Skip Function

By setting the RRS bit of DTCCR, the vector address read and transfer information read can be skipped. The current DTC vector number is always compared with the vector number of previous activation. If the vector numbers match when RRS = 1, a DTC data transfer is performed without reading the vector address and transfer information. If the previous activation is a chain transfer, the vector address read and transfer information read are always performed. Figure 10.6 shows the transfer information read skip timing.

To modify the vector table and transfer information, temporarily clear the RRS bit to 0, modify the vector table and transfer information, and then set the RRS bit to 1 again. When the RRS bit is cleared to 0, the stored vector number is deleted, and the updated vector table and transfer information are read at the next activation.



**Figure 10.6 Transfer Information Read Skip Timing**

### 10.5.3 Transfer Information Writeback Skip Function

By specifying bit SM1 in MRA and bit DM1 in MRB to the fixed address mode, a part of transfer information will not be written back. This function is performed regardless of short or full address mode. Table 10.5 shows the transfer information writeback skip condition and writeback skipped registers. Note that the CRA and CRB are always written back regardless of the short or full address mode. In addition in full address mode, the writeback of the MRA and MRB are always skipped.

**Table 10.5 Transfer Information Writeback Skip Condition and Writeback Skipped Registers**

SM1	DM1	SAR	DAR
0	0	Skipped	Skipped
0	1	Skipped	Written back
1	0	Written back	Skipped
1	1	Written back	Written back

### 10.5.4 Normal Transfer Mode

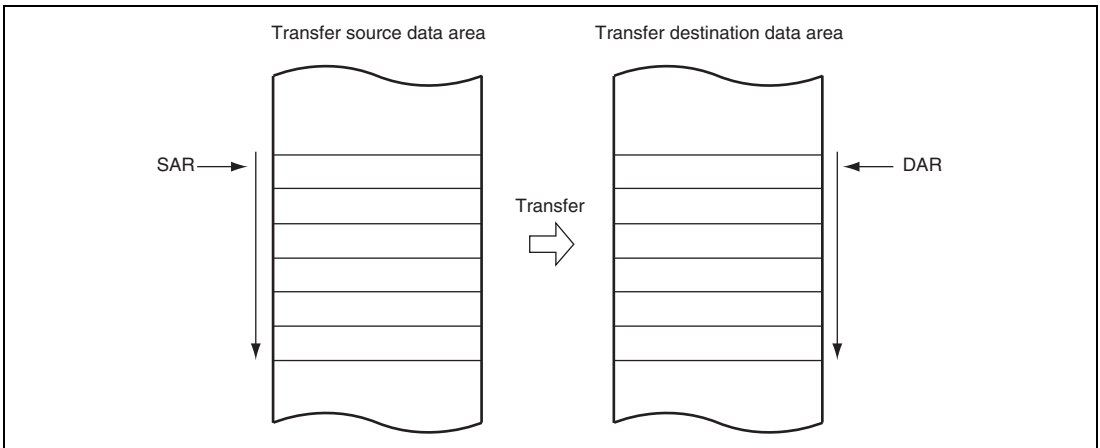
In normal transfer mode, one operation transfers one byte, one word, or one longword of data. From 1 to 65,536 transfers can be specified. The transfer source and destination addresses can be specified as incremented, decremented, or fixed. When the specified number of transfers ends, an interrupt can be requested to the CPU.

Table 10.6 lists the register function in normal transfer mode. Figure 10.7 shows the memory map in normal transfer mode.

**Table 10.6 Register Function in Normal Transfer Mode**

Register	Function	Written Back Value
SAR	Source address	Incremented/decremented/fixed*
DAR	Destination address	Incremented/decremented/fixed*
CRA	Transfer count A	CRA – 1
CRB	Transfer count B	Not updated

Note: \* Transfer information writeback is skipped.



**Figure 10.7 Memory Map in Normal Transfer Mode**

### 10.5.5 Repeat Transfer Mode

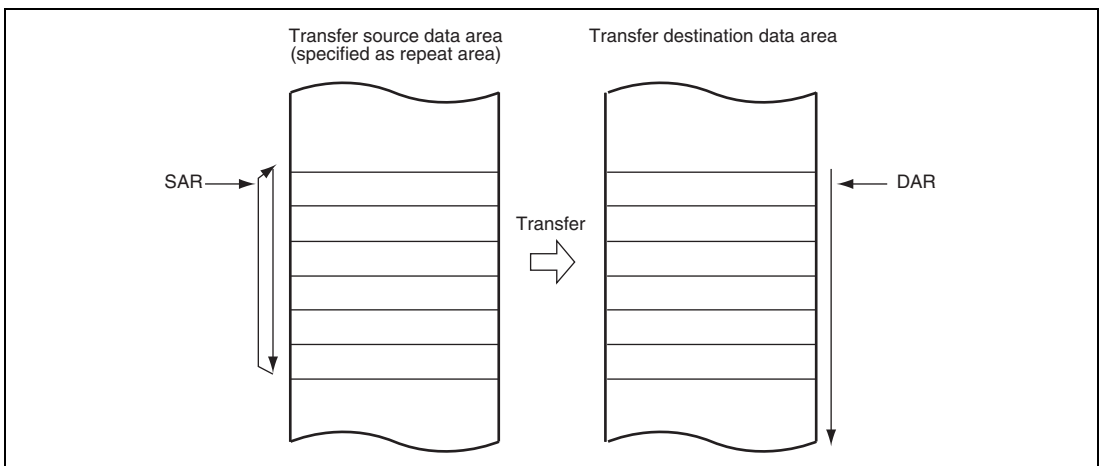
In repeat transfer mode, one operation transfers one byte, one word, or one longword of data. By the DTS bit in MRB, either the source or destination can be specified as a repeat area. From 1 to 256 transfers can be specified. When the specified number of transfers ends, the transfer counter and address register specified as the repeat area is restored to the initial state, and transfer is repeated. The other address register is then incremented, decremented, or left fixed. In repeat transfer mode, the transfer counter (CRAL) is updated to the value specified in CRAH when CRAL becomes H'00. Thus the transfer counter value does not reach H'00, and therefore a CPU interrupt cannot be requested when DISEL = 0.

Table 10.7 lists the register function in repeat transfer mode. Figure 10.8 shows the memory map in repeat transfer mode.

**Table 10.7 Register Function in Repeat Transfer Mode**

Register	Function	Written Back Value	
		CRAL is not 1	CRAL is 1
SAR	Source address	Incremented/decremented/ fixed*	DTS = 0: Incremented/ decremented/fixed*  DTS = 1: SAR initial value
DAR	Destination address	Incremented/decremented/ fixed*	DTS = 0: DAR initial value  DTS = 1: Incremented/ decremented/fixed*
CRAH	Transfer count storage	CRAH	CRAH
CRAL	Transfer count A	CRAL – 1	CRAH
CRB	Transfer count B	Not updated	Not updated

Note: \* Transfer information writeback is skipped.



**Figure 10.8 Memory Map in Repeat Transfer Mode  
(When Transfer Source is Specified as Repeat Area)**

### 10.5.6 Block Transfer Mode

In block transfer mode, one operation transfers one block of data. Either the transfer source or the transfer destination is designated as a block area by the DTS bit in MRB.

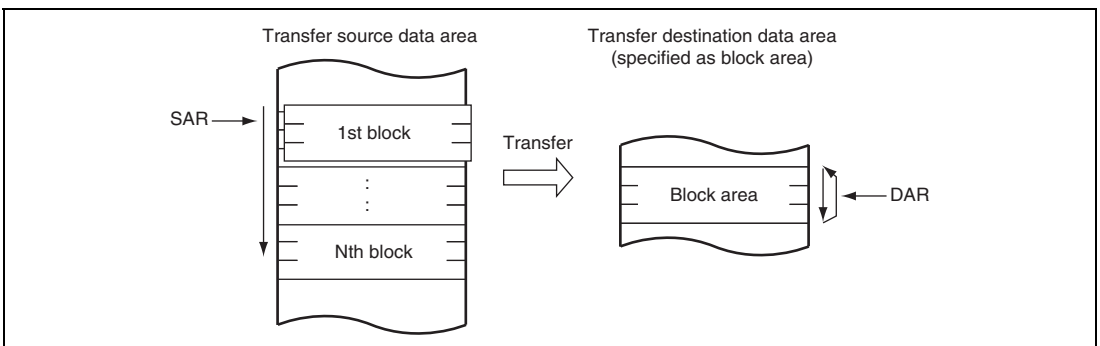
The block size is 1 to 256 bytes (1 to 256 words, or 1 to 256 longwords). When the transfer of one block ends, the block size counter (CRAL) and address register (SAR when DTS = 1 or DAR when DTS = 0) specified as the block area is restored to the initial state. The other address register is then incremented, decremented, or left fixed. From 1 to 65,536 transfers can be specified. When the specified number of transfers ends, an interrupt is requested to the CPU.

Table 10.8 lists the register function in block transfer mode. Figure 10.9 shows the memory map in block transfer mode.

**Table 10.8 Register Function in Block Transfer Mode**

Register	Function	Written Back Value
SAR	Source address	DTS = 0: Incremented/decremented/fixed* DTS = 1: SAR initial value
DAR	Destination address	DTS = 0: DAR initial value DTS = 1: Incremented/decremented/fixed*
CRAH	Block size storage	CRAH
CRAL	Block size counter	CRAH
CRB	Block transfer counter	CRB – 1

Note: \* Transfer information writeback is skipped.



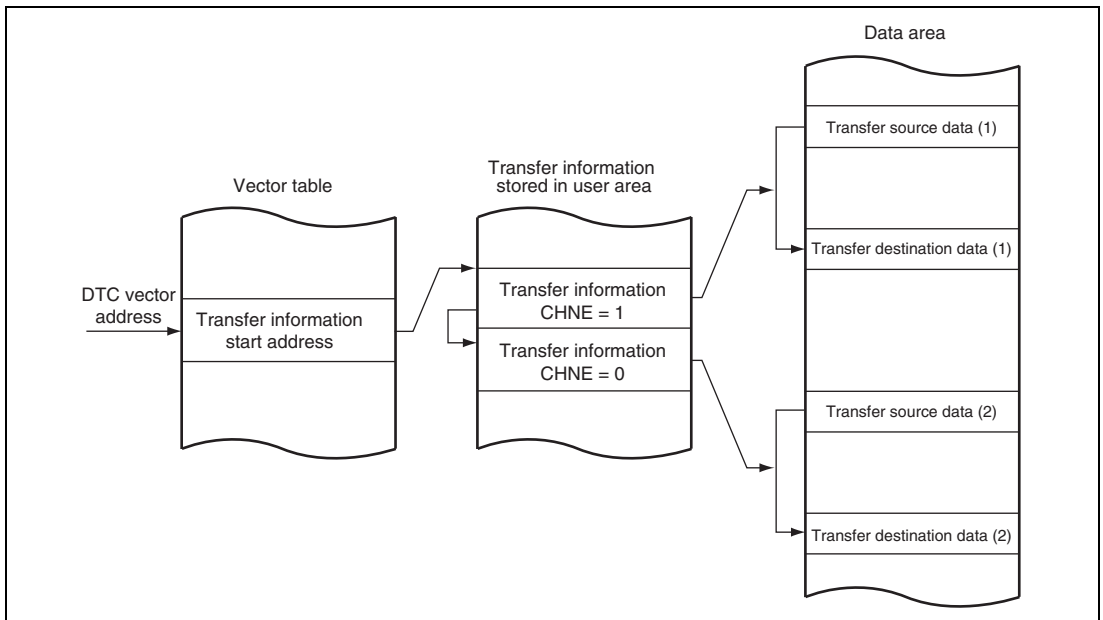
**Figure 10.9 Memory Map in Block Transfer Mode  
(When Transfer Destination is Specified as Block Area)**

### 10.5.7 Chain Transfer

Setting the CHNE bit in MRB to 1 enables a number of data transfers to be performed consecutively in response to a single transfer request. Setting the CHNE and CHNS bits in MRB set to 1 enables a chain transfer only when the transfer counter reaches 0. SAR, DAR, CRA, CRB, MRA, and MRB, which define data transfers, can be set independently. Figure 10.10 shows the chain transfer operation.

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 the DISEL bit to 1, and the interrupt source flag for the activation source and DTCER are not affected.

In repeat transfer mode, setting the RCHNE bit in DTCCR and the CHNE and CHNS bits in MRB to 1 enables a chain transfer after transfer with transfer counter = 1 has been completed.

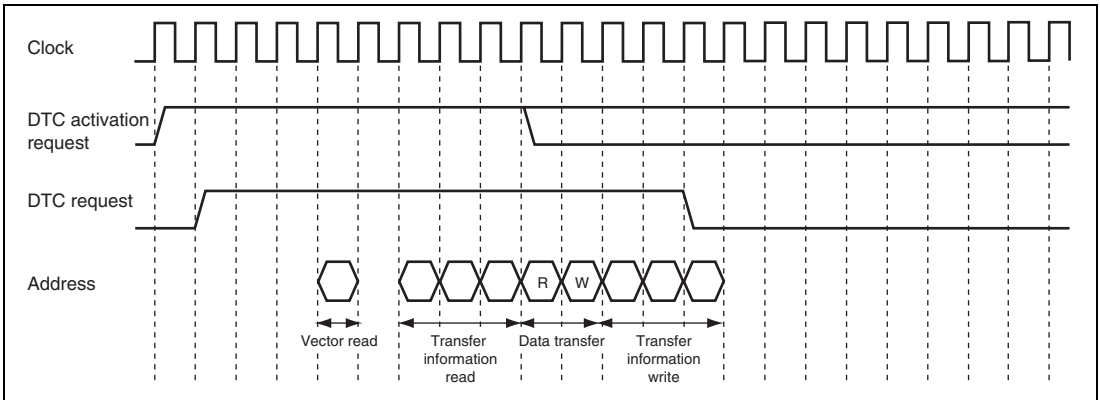


**Figure 10.10 Operation of Chain Transfer**



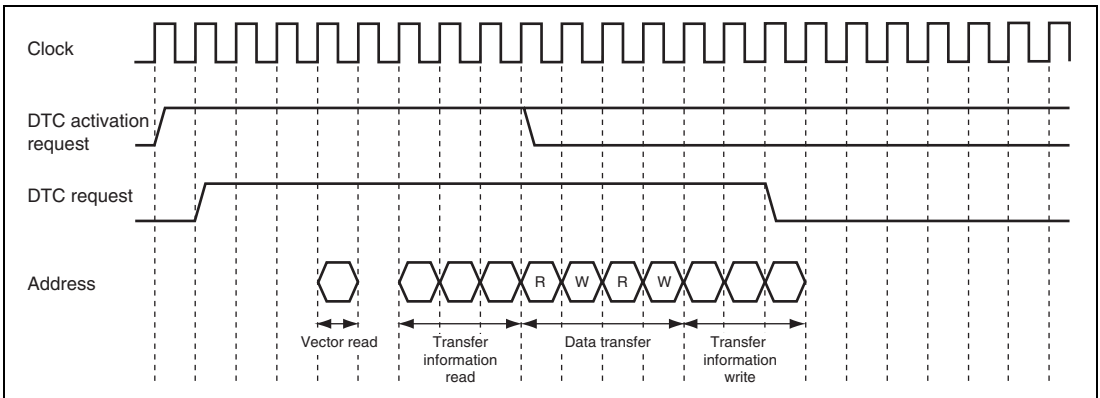
## 10.5.8 Operation Timing

Figures 10.11 to 10.14 show the DTC operation timings.



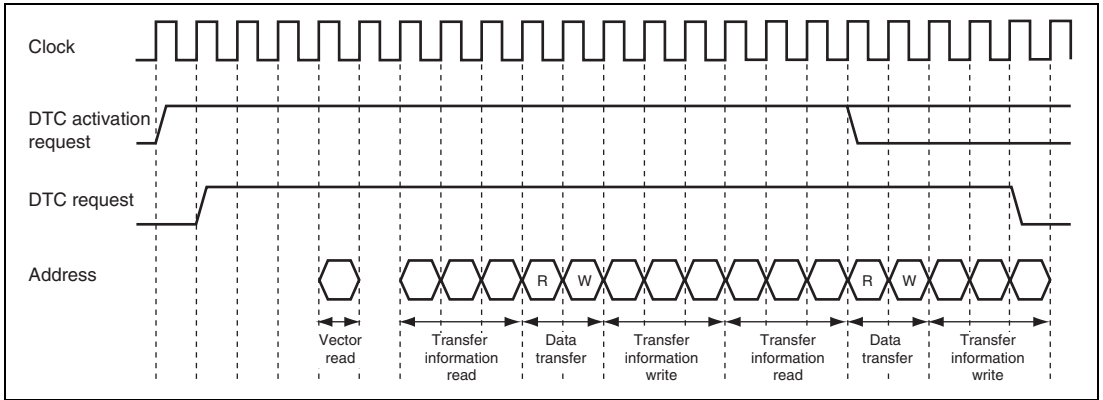
**Figure 10.11 DTC Operation Timing**

**(Example of Short Address Mode in Normal Transfer Mode or Repeat Transfer Mode)**

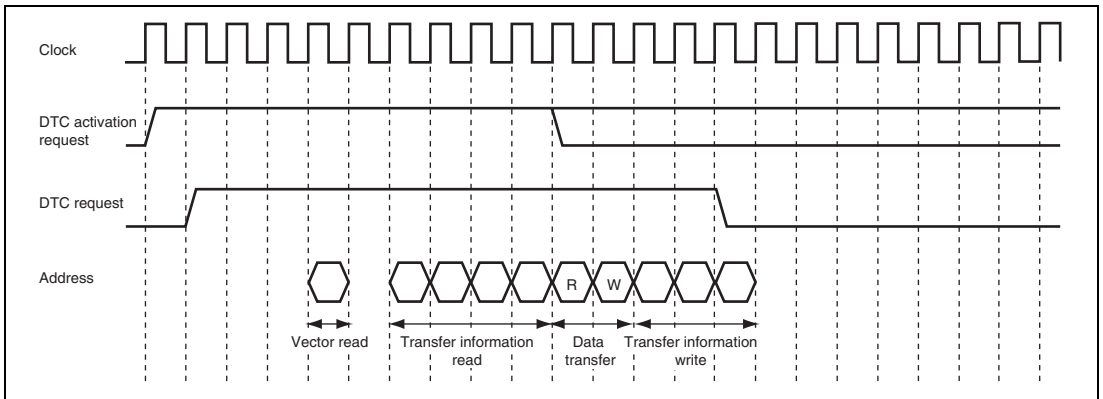


**Figure 10.12 DTC Operation Timing**

**(Example of Short Address Mode in Block Transfer Mode with Block Size of 2)**



**Figure 10.13 DTC Operation Timing (Example of Short Address Mode in Chain Transfer)**



**Figure 10.14 DTC Operation Timing (Example of Full Address Mode in Normal Transfer Mode or Repeat Transfer Mode)**

### 10.5.9 Number of DTC Execution Cycles

Table 10.9 shows the execution status for a single DTC data transfer, and table 10.10 shows the number of cycles required for each execution.

**Table 10.9 DTC Execution Status**

Mode	Vector Read I	Transfer Information Read J			Transfer Information Write L			Data Read L		Data Write M			Internal Operation N			
		0* <sup>1</sup>	4* <sup>2</sup>	3* <sup>3</sup>	0* <sup>1</sup>	3* <sup>2,3</sup>	2* <sup>4</sup>	1* <sup>5</sup>	3* <sup>6</sup>	2* <sup>7</sup>	1	3* <sup>6</sup>	2* <sup>7</sup>	1	1	0* <sup>1</sup>
Normal	1	0* <sup>1</sup>	4* <sup>2</sup>	3* <sup>3</sup>	0* <sup>1</sup>	3* <sup>2,3</sup>	2* <sup>4</sup>	1* <sup>5</sup>	3* <sup>6</sup>	2* <sup>7</sup>	1	3* <sup>6</sup>	2* <sup>7</sup>	1	1	0* <sup>1</sup>
Repeat	1	0* <sup>1</sup>	4* <sup>2</sup>	3* <sup>3</sup>	0* <sup>1</sup>	3* <sup>2,3</sup>	2* <sup>4</sup>	1* <sup>5</sup>	3* <sup>6</sup>	2* <sup>7</sup>	1	3* <sup>6</sup>	2* <sup>7</sup>	1	1	0* <sup>1</sup>
Block transfer	1	0* <sup>1</sup>	4* <sup>2</sup>	3* <sup>3</sup>	0* <sup>1</sup>	3* <sup>2,3</sup>	2* <sup>4</sup>	1* <sup>5</sup>	3•P * <sup>6</sup>	2•P* <sup>7</sup>	1•P	3•P * <sup>6</sup>	2•P* <sup>7</sup>	1•P	1	0* <sup>1</sup>

[Legend]

P: Block size (CRAH and CRAL value)

- Note:
1. When transfer information read is skipped
  2. In full address mode operation
  3. In short address mode operation
  4. When the SAR or DAR is in fixed mode
  5. When the SAR and DAR are in fixed mode
  6. When a longword is transferred while an odd address is specified in the address register
  7. When a word is transferred while an odd address is specified in the address register or when a longword is transferred while address  $4n + 2$  is specified

**Table 10.10 Number of Cycles Required for Each Execution State**

Object to be Accessed		On-Chip		On-Chip I/O			External Devices			
		RAM	ROM	Registers						
Bus width		32	32	8	16	32	8	16		
Access cycles		1	1	2	2	2	2	3	2	3
Execution status	Vector read $S_i$	1	1	□	□	□	8	12 + 4m	4	6 + 2m
	Transfer information read $S_j$	1	1	□	□	□	8	12 + 4m	4	6 + 2m
	Transfer information write $S_k$	1	1	□	□	□	8	12 + 4m	4	6 + 2m
	Byte data read $S_L$	1	1	2	2	2	2	3 + m	2	3 + m
	Word data read $S_L$	1	1	4	2	2	4	4 + 2m	2	3 + m
	Longword data read $S_L$	1	1	8	4	2	8	12 + 4m	4	6 + 2m
	Byte data write $S_M$	1	1	2	2	2	2	3 + m	2	3 + m
	Word data write $S_M$	1	1	4	2	2	4	4 + 2m	2	3 + m
	Longword data write $S_M$	1	1	8	4	2	8	12 + 4m	4	6 + 2m
	Internal operation $S_N$								1	

[Legend]

m: Number of wait cycles 0 to 7 (For details, see section 8, Bus Controller (BSC).)

The number of execution cycles 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).

$$\text{Number of execution cycles} = I \cdot S_i + \Sigma (J \cdot S_j + K \cdot S_k + L \cdot S_L + M \cdot S_M) + N \cdot S_N$$

### 10.5.10 DTC Bus Release Timing

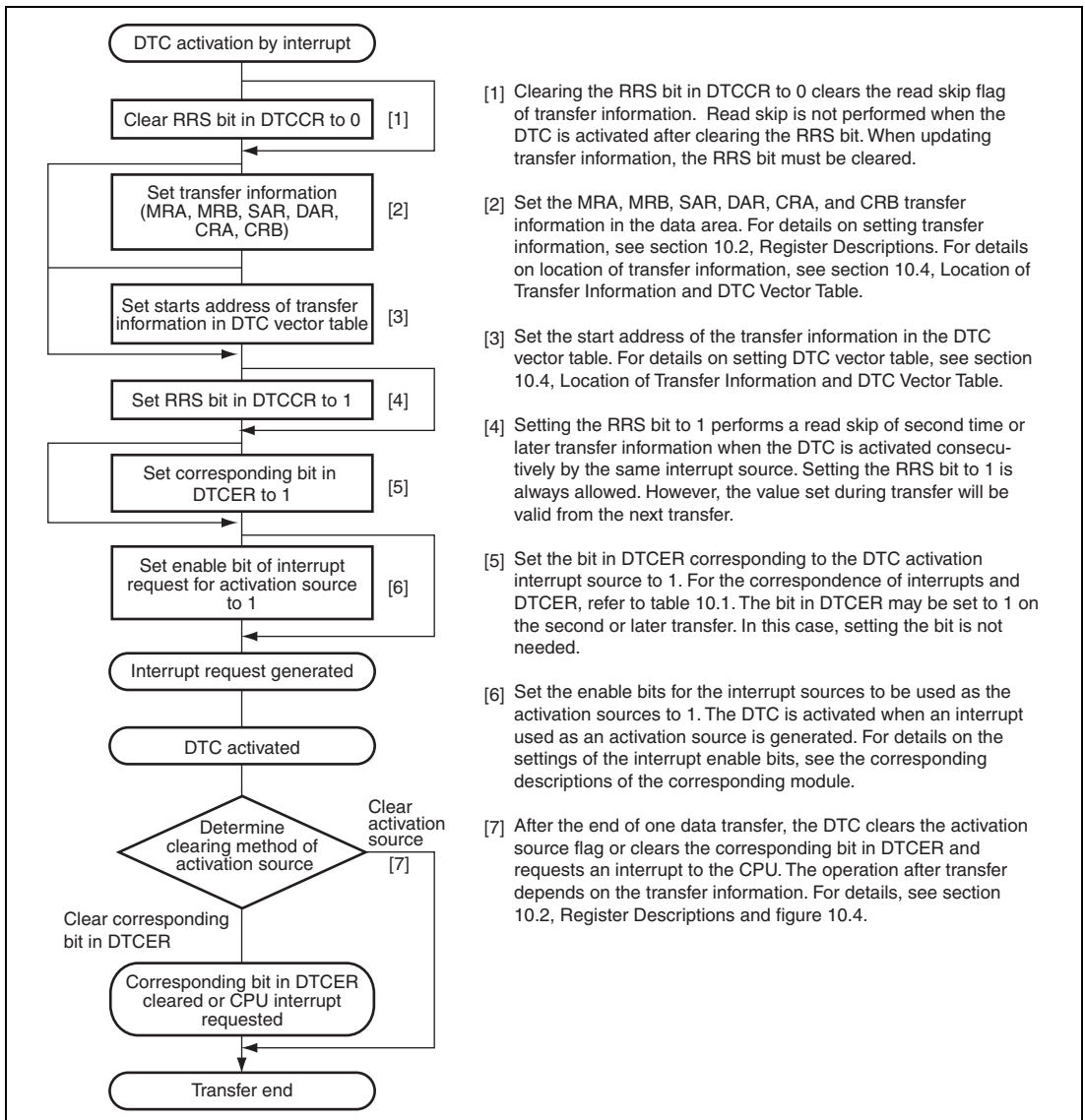
The DTC requests the bus mastership to the bus arbiter when an activation request occurs. The DTC releases the bus after a vector read, transfer information read, a single data transfer, or transfer information writeback. The DTC does not release the bus during transfer information read, single data transfer, or transfer information writeback.

### 10.5.11 DTC Priority Level Control to the CPU

The priority of the DTC activation sources over the CPU can be controlled by the CPU priority level specified by bits CPUP2 to CPUP0 in CPUPCR and the DTC priority level specified by bits DTCP2 to DTCP0. For details, see section 6, Interrupt Controller.

## 10.6 DTC Activation by Interrupt

The procedure for using the DTC with interrupt activation is shown in figure 10.15.



- [1] Clearing the RRS bit in DTCCR to 0 clears the read skip flag of transfer information. Read skip is not performed when the DTC is activated after clearing the RRS bit. When updating transfer information, the RRS bit must be cleared.
- [2] Set the MRA, MRB, SAR, DAR, CRA, and CRB transfer information in the data area. For details on setting transfer information, see section 10.2, Register Descriptions. For details on location of transfer information, see section 10.4, Location of Transfer Information and DTC Vector Table.
- [3] Set the start address of the transfer information in the DTC vector table. For details on setting DTC vector table, see section 10.4, Location of Transfer Information and DTC Vector Table.
- [4] Setting the RRS bit to 1 performs a read skip of second time or later transfer information when the DTC is activated consecutively by the same interrupt source. Setting the RRS bit to 1 is always allowed. However, the value set during transfer will be valid from the next transfer.
- [5] Set the bit in DTCER corresponding to the DTC activation interrupt source to 1. For the correspondence of interrupts and DTCER, refer to table 10.1. The bit in DTCER may be set to 1 on the second or later transfer. In this case, setting the bit is not needed.
- [6] 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. For details on the settings of the interrupt enable bits, see the corresponding descriptions of the corresponding module.
- [7] After the end of one data transfer, the DTC clears the activation source flag or clears the corresponding bit in DTCER and requests an interrupt to the CPU. The operation after transfer depends on the transfer information. For details, see section 10.2, Register Descriptions and figure 10.4.

## 10.7 Examples of Use of the DTC

### 10.7.1 Normal Transfer 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 transfer mode ( $MD1 = MD0 = 0$ ), and byte size ( $Sz1 = Sz0 = 0$ ). The DTS bit can have any value. Set MRB for one data transfer by one interrupt ( $CHNE = 0, DISEL = 0$ ). Set the RDR address of the SCI 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 transfer information for an RXI interrupt 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 receive end (RXI) interrupt. Since the generation of a receive error during the SCI reception 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. Termination processing should be performed in the interrupt handling routine.

### 10.7.2 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  $CHNE = 0$ ).

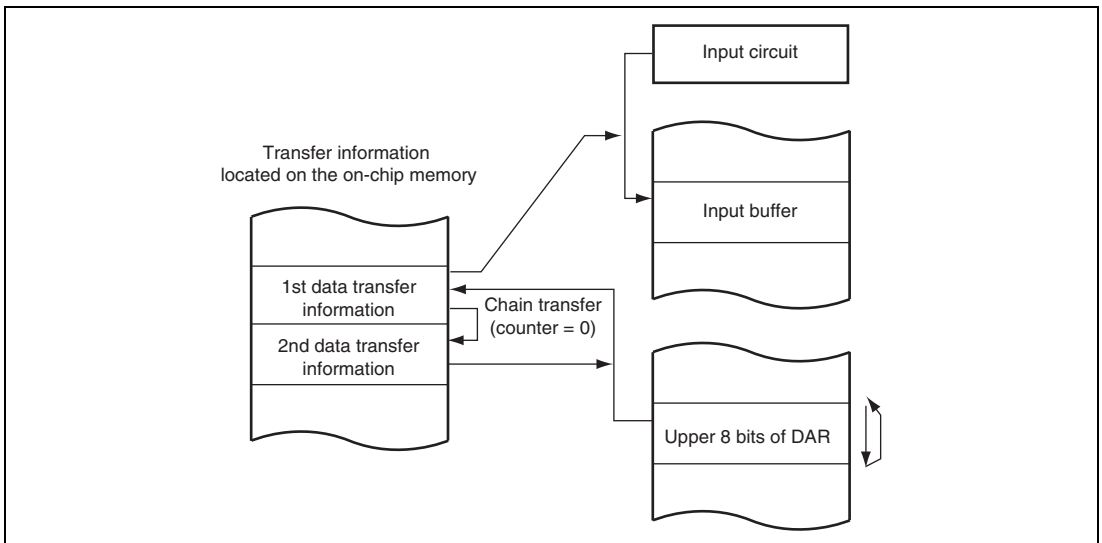
1. Perform settings for transfer to the PPG's NDR. Set MRA to source address incrementing ( $SM1 = 1$ ,  $SM0 = 0$ ), fixed destination address ( $DM1 = DM0 = 0$ ), repeat mode ( $MD1 = 0$ ,  $MD0 = 1$ ), and word size ( $Sz1 = 0$ ,  $Sz0 = 1$ ). Set the source side as a repeat area ( $DTS = 1$ ). Set MRB to chain transfer mode ( $CHNE = 1$ ,  $CHNS = 0$ ,  $DISEL = 0$ ). 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 ( $Sz1 = 0$ ,  $Sz0 = 1$ ). 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 information consecutively after the NDR transfer information.
4. Set the start address of the NDR transfer information to the DTC vector address.
5. Set the bit corresponding to the TGIA interrupt in DTCEM 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. Termination processing should be performed in the interrupt handling routine.

### 10.7.3 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 10.16 shows the chain transfer when the counter value is 0.

1. For the first transfer, set the normal transfer mode for input data. Set the fixed transfer source address,  $CRA = H'0000$  (65,536 times),  $CHNE = 1$ ,  $CHNS = 1$ , and  $DISEL = 0$ .
2. Prepare the upper 8-bit addresses of the start addresses for 65,536-transfer units for the first data transfer in a separate area (in ROM, etc.). For example, if the input buffer is configured at addresses  $H'200000$  to  $H'21FFFF$ , prepare  $H'21$  and  $H'20$ .
3. For the second transfer, set repeat transfer mode (with the source side as the repeat area) for re-setting the transfer destination address for the first data transfer. Use the upper eight bits of  $DAR$  in the first transfer 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 65536 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 eight 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 65536 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 eight 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, no interrupt request is sent to the CPU.



**Figure 10.16 Chain Transfer when Counter = 0**



## 10.8 Interrupt Sources

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 priority level control in the interrupt controller.

## 10.9 Usage Notes

### 10.9.1 Module Stop Function Setting

Operation of the DTC can be disabled or enabled using the module stop control register. The initial setting is for operation of the DTC to be enabled. Register access is disabled by setting the module stop state. The module stop state cannot be set while the DTC is activated. For details, refer to section 24, Power-Down Modes.

### 10.9.2 On-Chip RAM

Transfer information can be located in on-chip RAM. In this case, the RAME bit in SYSCR must not be cleared to 0.

### 10.9.3 DMAC Transfer End Interrupt

When the DTC is activated by a DMAC transfer end interrupt, the DTE bit of DMDR is not controlled by the DTC but its value is modified with the write data regardless of the transfer counter value and DISEL bit setting. Accordingly, even if the DTC transfer counter value becomes 0, no interrupt request may be sent to the CPU in some cases.

### 10.9.4 DTCE Bit Setting

For DTCE bit setting, use bit manipulation instructions such as BSET and BCLR. If all interrupts are disabled, multiple activation sources can be set at one time (only at the initial setting) by writing data after executing a dummy read on the relevant register.

### **10.9.5 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. At this time, SCI and A/D converter interrupt/activation sources, are cleared when the DTC reads or writes to the relevant 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.

### **10.9.6 Transfer Information Start Address, Source Address, and Destination Address**

The transfer information start address to be specified in the vector table should be address  $4n$ . If an address other than address  $4n$  is specified, the lower 2 bits of the address are regarded as 0s.

The source and destination addresses specified in SAR and DAR, respectively, will be transferred in the divided bus cycles depending on the address and data size.

### **10.9.7 Transfer Information Modification**

When  $IBCCS = 1$  and the DMAC is used, clear the IBCCS bit to 0 and then set to 1 again before modifying the DTC transfer information in the CPU exception handling routine initiated by a DTC transfer end interrupt.

### **10.9.8 Endian Format**

The DTC supports big and little endian formats. The endian formats used when transfer information is written to and when transfer information is read from by the DTC must be the same.

---

## Section 11 I/O Ports

Table 11.1 summarizes the port functions. The pins of each port also have other functions such as input/output pins of on-chip peripheral modules or external interrupt input pins. Each I/O port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, a port register (PORT) used to read the pin states, and an input buffer control register (ICR) that controls input buffer on/off. Port 5 does not have a DR or a DDR register.

Ports D to F and H and I have internal input pull-up MOSs and a pull-up MOS control register (PCR) that controls the on/off state of the input pull-up MOSs.

Ports 2 and F include an open-drain control register (ODR) that controls on/off of the output buffer PMOSs.

All of the I/O ports can drive a single TTL load with a capacitive component of up to 30 pF and drive Darlington transistors when functioning as output ports.

Port 2 have pins for Schmitt-trigger inputs. Schmitt-trigger input is enabled for pins of other ports when they are used as IRQ, TPU, TMR, or IIC2 inputs.

**Table 11.1 Port Functions**

Port	Description	Bit	Function			Schmitt-Trigger Input*1	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 1	General I/O port function multiplexed with interrupt input, SCI I/O, DMAC I/O, A/D converter input, TPU input, and IIC2 I/O	7	P17/SCL0	$\overline{\text{IRQ7-A}}$ / TCLKD-B/ $\overline{\text{ANDSTRG}}$	—	$\overline{\text{IRQ7-A}}$ , TCLKD-B, SCL0	—	—
		6	P16/SDA0/SCK3	$\overline{\text{IRQ6-A}}$ / TCLKC-B	$\overline{\text{DACK1-A}}$	$\overline{\text{IRQ6-A}}$ , TCLKC-B, SDA0		
		5	P15/SCL1	$\overline{\text{IRQ5-A}}$ / TCLKB-B/ RxD3	$\overline{\text{TEND1-A}}$	$\overline{\text{IRQ5-A}}$ , TCLKB-B, SCL1		
		4	P14/SDA1	$\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ / TCLKA-B	TxD3	$\overline{\text{IRQ4-A}}$ , TCLKA-B, SDA1		
		3	P13	$\overline{\text{ADTRG0}}$ / $\overline{\text{IRQ3-A}}$	—	$\overline{\text{IRQ3-A}}$		
		2	P12/SCK2	$\overline{\text{IRQ2-A}}$	$\overline{\text{DACK0-A}}$	$\overline{\text{IRQ2-A}}$		
		1	P11	RxD2/ $\overline{\text{IRQ1-A}}$	$\overline{\text{TEND0-A}}$	$\overline{\text{IRQ1-A}}$		
		0	P10	$\overline{\text{DREQ0-A}}$ / $\overline{\text{IRQ0-A}}$	TxD2	$\overline{\text{IRQ0-A}}$		

Port	Description	Bit	Function			Schmitt-Trigger Input*1	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 2	General I/O port function multiplexed with interrupt input, PPG output, TPU I/O, TMR I/O, and SCI I/O	7	P27/TIOCB5	TIOCA5/ $\overline{\text{IRQ15}}$	PO7	P27, TIOCB5, TIOCA5, $\overline{\text{IRQ15}}$	—	O
		6	P26/TIOCA5	$\overline{\text{IRQ14}}$	PO6/TMO1/TxD1	P26, TIOCA5, $\overline{\text{IRQ14}}$		
		5	P25/TIOCA4	TMCI1/RxD1/ $\overline{\text{IRQ13-A}}$	PO5	P25, TIOCA4, TMC11, $\overline{\text{IRQ13-A}}$		
		4	P24/TIOCB4/SCK1	TIOCA4/TMRI1/ $\overline{\text{IRQ12-A}}$	PO4	P24, TIOCB4, TIOCA4, TMRI1, $\overline{\text{IRQ12-A}}$		
		3	P23/TIOCD3	$\overline{\text{IRQ11-A}}$ /TIOCC3	PO3	P23, TIOCD3, $\overline{\text{IRQ11-A}}$		
		2	P22/TIOCC3	$\overline{\text{IRQ10-A}}$	PO2/TMO0/TxD0	All input functions		
		1	P21/TIOCA3	TMCI0/RxD0/ $\overline{\text{IRQ9-A}}$	PO1	P21, $\overline{\text{IRQ9-A}}$ , TIOCA3, TMC10		
		0	P20/TIOCB3/SCK0	TIOCA3/TMRI0/ $\overline{\text{IRQ8-A}}$	PO0	P20, $\overline{\text{IRQ8-A}}$ , TIOCB3, TIOCA3, TMRI0		

Port	Description	Bit	Function			Schmitt-Trigger Input*1	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 3	General I/O port function multiplexed with bus control output, PPG output, DMAC I/O, and TPU I/O	7	P37/TIOCB2	TIOCA2/TCLKD-A	PO15	P37, TIOCB2, TIOCA2, TCLKD-A	—	O
		6	P36/TIOCA2	—	PO14	P36, TIOCA2		
		5	P35/TIOCB1	TIOCA1/TCLKC-A	PO13/ $\overline{\text{DACK1}}$ -B	P35, TIOCB1		
		4	P34/TIOCA1	—	PO12/ $\overline{\text{TEND1}}$ -B	P34, TIOCA1		
		3	P33/TIOCD0	TIOCC0/ TCLKB-A/ $\overline{\text{DREQ1}}$ -B	PO11/ $\overline{\text{CS3}}$ / $\overline{\text{CS7}}$ -A	P33, TIOCD0, TIOCC0, TCKB-A		
		2	P32/TIOCC0	TCLKA-A	PO10/ $\overline{\text{DACK0}}$ -B/ $\overline{\text{CS2}}$ -A/ $\overline{\text{CS6}}$ -A	P32, TIOCC0, TCLKA-A		
		1	P31/TIOCB0	TIOCA0	PO9/ $\overline{\text{TEND0}}$ -B/ $\overline{\text{CS1}}$ / $\overline{\text{CS2}}$ -B/ $\overline{\text{CS5}}$ -A/ $\overline{\text{CS6}}$ -B/ $\overline{\text{CS7}}$ -B	P31, TIOCB0, TIOCA0		
		0	P30/TIOCA0	$\overline{\text{DREQ0}}$ -B	PO8/ $\overline{\text{CS0}}$ / $\overline{\text{CS4}}$ / $\overline{\text{CS5}}$ -B	P30, TIOCA0		
Port 4	General input port	7	—	P47	—	—	—	—
		6	—	P46	—	—		
		5	—	P45	—	—		
		4	—	P44	—	—		
		3	—	P43	—	—		
		2	—	P42	—	—		
		1	—	P41	—	—		
		0	—	P40	—	—		

Port	Description	Bit	I/O	Function		Schmitt-Trigger Input*1	Input Pull-up MOS Function	Open-Drain Output Function
				Input	Output			
Port 5	General input port function multiplexed with interrupt input, A/D converter input, and D/A converter output	7	—	P57/AN7/ $\overline{\text{IRQ7}}$ -B	DA1	$\overline{\text{IRQ7}}$ -B	—	—
		6	—	P56/AN6/ $\overline{\text{IRQ6}}$ -B	DA0	$\overline{\text{IRQ6}}$ -B	—	—
		5	—	P55/AN5/ $\overline{\text{IRQ5}}$ -B	—	$\overline{\text{IRQ5}}$ -B	—	—
		4	—	P54/AN4/ $\overline{\text{IRQ4}}$ -B	—	$\overline{\text{IRQ4}}$ -B	—	—
		3	—	P53/AN3/ $\overline{\text{IRQ3}}$ -B	—	$\overline{\text{IRQ3}}$ -B	—	—
		2	—	P52/AN2/ $\overline{\text{IRQ2}}$ -B	—	$\overline{\text{IRQ2}}$ -B	—	—
		1	—	P51/AN1/ $\overline{\text{IRQ1}}$ -B	—	$\overline{\text{IRQ1}}$ -B	—	—
		0	—	P50/AN0/ $\overline{\text{IRQ0}}$ -B	—	$\overline{\text{IRQ0}}$ -B	—	—
Port 6	General I/O port function multiplexed with TMR I/O, SCI I/O, H-UDI input, and interrupt input	7	—	—	—	—	—	—
		6	—	—	—	—	—	—
		5	P65	$\overline{\text{IRQ13}}$ -B	TMO3	$\overline{\text{IRQ13}}$ -B	—	—
		4	P64	TMCI3/ $\overline{\text{IRQ12}}$ -B	—	TMCI3, $\overline{\text{IRQ12}}$ -B	—	—
		3	P63	TMRI3/ $\overline{\text{IRQ11}}$ -B	—	TMRI3, $\overline{\text{IRQ11}}$ -B	—	—
		2	P62/SCK4	$\overline{\text{IRQ10}}$ -B	TMO2	$\overline{\text{IRQ10}}$ -B	—	—
		1	P61	TMCI2/RxD4/ $\overline{\text{IRQ9}}$ -B	—	TMCI2, $\overline{\text{IRQ9}}$ -B	—	—
		0	P60	TMRI2/ $\overline{\text{IRQ8}}$ -B	TxD4	TMRI2, $\overline{\text{IRQ8}}$ -B	—	—
Port A	General I/O port function multiplexed with system clock output and bus control I/O	7	—	PA7	B $\phi$	—	—	—
		6	PA6	—	$\overline{\text{AS/AH/BS}}$ -B	—	—	
		5	PA5	—	$\overline{\text{RD}}$	—	—	
		4	PA4	—	$\overline{\text{LHWR/LUB}}$	—	—	
		3	PA3	—	$\overline{\text{LLWR/LLB}}$	—	—	
		2	PA2	$\overline{\text{BREQ/WAIT}}$	—	—	—	
		1	PA1	—	$\overline{\text{BACK}}$ / (RD/WR)	—	—	
		0	PA0	—	$\overline{\text{BREQ/BS-A}}$	—	—	

Port	Description	Bit	Function		Schmitt-Trigger Input* <sup>1</sup>	Input Pull-up MOS Function	Open-Drain Output Function	
			I/O	Input				Output
Port D	General I/O port function multiplexed with address output	7	PD7	—	A7	—	O	—
		6	PD6	—	A6	—		
		5	PD5	—	A5	—		
		4	PD4	—	A4	—		
		3	PD3	—	A3	—		
		2	PD2	—	A2	—		
		1	PD1	—	A1	—		
		0	PD0	—	A0	—		
Port E	General I/O port function multiplexed with address output	7	PE7	—	A15	—	O	—
		6	PE6	—	A14			
		5	PE5	—	A13			
		4	PE4	—	A12			
		3	PE3	—	A11			
		2	PE2	—	A10			
		1	PE1	—	A9			
		0	PE0	—	A8			
Port F	General I/O port function multiplexed with address output	7	—	—	—	—	O	O
		6	—	—	—			
		5	—	—	—			
		4	PF4	—	A20			
		3	PF3	—	A19			
		2	PF2	—	A18			
		1	PF1	—	A17			
		0	PF0	—	A16			



Port	Description	Bit	I/O	Function		Schmitt-Trigger Input*1	Input Pull-up MOS Function	Open-Drain Output Function
				Input	Output			
Port H	General I/O port function multiplexed with bi-directional data bus	7	PH7/D7*2	—	—	—	O	—
		6	PH6/D6*2	—	—	—		
		5	PH5/D5*2	—	—	—		
		4	PH4/D4*2	—	—	—		
		3	PH3/D3*2	—	—	—		
		2	PH2/D2*2	—	—	—		
		1	PH1/D1*2	—	—	—		
		0	PH0/D0*2	—	—	—		
Port I	General I/O port function multiplexed with bi-directional data bus	7	PI7/D15*2	—	TMO7	—	O	—
		6	PI6/D14*2	—	TMO6	—		
		5	PI5/D13*2	—	TMO5	—		
		4	PI4/D12*2	—	TMO4	—		
		3	PI3/D11*2	—	—	—		
		2	PI2/D10*2	—	—	—		
		1	PI1/D9*2	—	—	—		
		0	PI0/D8*2	—	—	—		

- Notes:
1. Pins without Schmitt-trigger input buffer have CMOS input buffer.
  2. Addresses are also output when accessing to the address/data multiplexed I/O space.

## 11.1 Register Descriptions

Table 11.2 lists each port registers.

**Table 11.2 Register Configuration in Each Port**

Port	Number of Pins	Registers					
		DDR	DR	PORT	ICR	PCR	ODR
Port 1	8	O	O	O	O	—	—
Port 2	8	O	O	O	O	—	O
Port 3	8	O	O	O	O	—	—
Port 4	8	—	—	O	O	—	—
Port 5	8	—	—	O	O	—	—
Port 6* <sup>1</sup>	6	O	O	O	O	—	—
Port A	8	O	O	O	O	—	—
Port D	8	O	O	O	O	O	—
Port E	8	O	O	O	O	O	—
Port F* <sup>2</sup>	5	O	O	O	O	O	O
Port H	8	O	O	O	O	O	—
Port I	8	O	O	O	O	O	—

[Legend]

O: Register exists

—: No register exists

- Notes:
1. The lower six bits are valid and the upper two bits are reserved. The write value should always be the initial value.
  2. The lower five bits are valid and the upper three bits are reserved. The write value should always be the initial value.

### 11.1.1 Data Direction Register (PnDDR) (n = 1 to 3, 6, A, D to F, H, and I)

DDR is an 8-bit write-only register that specifies the port input or output for each bit. A read from the DDR is invalid and DDR is always read as an undefined value.

When the general I/O port function is selected, the corresponding pin functions as an output port by setting the corresponding DDR bit to 1; the corresponding pin functions as an input port by clearing the corresponding DDR bit to 0.

The initial DDR values are shown in table 11.3.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DDR	Pn6DDR	Pn5DDR	Pn4DDR	Pn3DDR	Pn2DDR	Pn1DDR	Pn0DDR
Initial Value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Note: The lower six bits are valid and the upper two bits are reserved for port 6 registers.  
The lower five bits are valid and the upper three bits are reserved for port F registers.

**Table 11.3 Startup Mode and Initial Value**

Port	Startup Mode	
	External Extended Mode	Single-Chip Mode
Port A	H'80	H'00
Other ports	H'00	

### 11.1.2 Data Register (PnDR) (n = 1 to 3, 6, A, D to F, H, and I)

DR is an 8-bit readable/writable register that stores the output data of the pins to be used as the general output port.

The initial value of DR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DR	Pn6DR	Pn5DR	Pn4DR	Pn3DR	Pn2DR	Pn1DR	Pn0DR
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: The lower six bits are valid and the upper two bits are reserved for port 6 registers.  
The lower five bits are valid and the upper three bits are reserved for port F registers.

### 11.1.3 Port Register (PORTn) (n = 1 to 6, A, D to F, H, and I)

PORT is an 8-bit read-only register that reflects the port pin status. A write to PORT is invalid.

When PORT is read, the DR bits that correspond to the respective DDR bits set to 1 are read and the status of each pin whose corresponding DDR bit is cleared to 0 is also read regardless of the ICR value.

The initial value of PORT is undefined and is determined based on the port pin status.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	R	R	R	R	R	R	R	R

Note: The lower six bits are valid and the upper two bits are reserved for port 6 registers.  
The lower five bits are valid and the upper three bits are reserved for port F registers.

### 11.1.4 Input Buffer Control Register (PnICR) (n = 1 to 6, A, D to F, H, and I)

ICR is an 8-bit readable/writable register that controls the port input buffers.

For bits in ICR set to 1, the input buffers of the corresponding pins are valid. For bits in ICR cleared to 0, the input buffers of the corresponding pins are invalid and the input signals are fixed high.

When the pin functions as an input for the peripheral modules, the corresponding bits should be set to 1. The initial value should be written to a bit whose corresponding pin is not used as an input or is used as an analog input/output pin.

If the bits in ICR have been cleared to 0, the pin state is not reflected to the peripheral modules.

When PORT is read, the pin status is always read regardless of the ICR value.

If ICR is modified, an internal edge may occur depending on the pin status. Accordingly, ICR should be modified when the corresponding input pins are not used. For example, in  $\overline{\text{IRQ}}$  input, modify ICR while the corresponding interrupt is disabled, clear the IRQF flag in ISR of the interrupt controller to 0, and then enable the corresponding interrupt. If an edge occurs after the ICR setting, the edge should be cancelled.

The initial value of ICR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ICR	Pn6ICR	Pn5ICR	Pn4ICR	Pn3ICR	Pn2ICR	Pn1ICR	Pn0ICR
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: The lower six bits are valid and the upper two bits are reserved for port 6 registers.  
The lower five bits are valid and the upper three bits are reserved for port F registers.

### 11.1.5 Pull-Up MOS Control Register (PnPCR) (n = D to F, H, and I)

PCR is an 8-bit readable/writable register that controls on/off of the port input pull-up MOS.

If a bit in PCR is set to 1 while the pin is in input state, the input pull-up MOS corresponding to the bit in PCR is turned on. Table 11.4 shows the input pull-up MOS status.

The initial value of PCR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7PCR	Pn6PCR	Pn5PCR	Pn4PCR	Pn3PCR	Pn2PCR	Pn1PCR	Pn0PCR
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

**Table 11.4 Input Pull-Up MOS State**

Port	Pin State	Reset	Hardware Standby Mode	Deep Software Standby Mode (IOKEEP = 0)	Software Standby Mode	Deep Software Standby Mode (IOKEEP = 1)	Other Operation	
Port D	Address output					OFF		
	Port output					OFF		
	Port input	OFF				ON/OFF		
Port E	Address output					OFF		
	Port output					OFF		
	Port input	OFF				ON/OFF		
Port F	Address output					OFF		
	Port output					OFF		
	Port input	OFF				ON/OFF		
Port H	Data input/output					OFF		
	Port output					OFF		
	Port input	OFF				ON/OFF		
Port I	Peripheral module output					OFF		
	Data input/output					OFF		
	Port output					OFF		
	Port input	OFF				ON/OFF		

[Legend]

OFF: The input pull-up MOS is always off.

ON/OFF: If PCR is set to 1, the input pull-up MOS is on; if PCR is cleared to 0, the input pull-up MOS is off.

### 11.1.6 Open-Drain Control Register (PnODR) (n = 2 and F)

ODR is an 8-bit readable/writable register that selects the open-drain output function.

If a bit in ODR is set to 1, the pin corresponding to that bit in ODR functions as an NMOS open-drain output. If a bit in ODR is cleared to 0, the pin corresponding to that bit in ODR functions as a CMOS output.

The initial value of ODR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ODR	Pn6ODR	Pn5ODR	Pn4ODR	Pn3ODR	Pn2ODR	Pn1ODR	Pn0ODR
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: The lower five bits are valid and the upper three bits are reserved for port F registers.



## 11.2 Output Buffer Control

This section describes the output priority of each pin.

The name of each peripheral module pin is followed by "\_OE". This (for example: MIOCA4\_OE) indicates whether the output of the corresponding function is valid (1) or if another setting is specified (0). Table 11.5 lists each port output signal's valid setting. For details on the corresponding output signals, see the register description of each peripheral module. If the name of each peripheral module pin is followed by A or B, the pin function can be modified by the port function control register (PFCR). For details, see section 11.3, Port Function Controller.

For a pin whose initial value changes according to the activation mode, "Initial value E" indicates the initial value when the LSI is started up in external extended mode and "Initial value S" indicates the initial value when the LSI is started in single-chip mode.

### 11.2.1 Port 1

#### (1) P17/ANDSTRG/IRQ7-A/TCLKD-B/SCL0

The pin function is switched as shown below according to the combination of the IIC2 register and P17DDR bit settings.

Module Name	Pin Function	Setting	
		IIC2	I/O Port
		SCL0_OE	P17DDR
IIC2	SCL0 I/O	1	—
I/O port	P17 output	0	1
	P17 input (initial value)	0	0

**(2) P16/SCK3/ $\overline{\text{DACK1}}$ -A/ $\overline{\text{IRQ6}}$ -A/TCLKC-B/SDA0**

The pin function is switched as shown below according to the combination of the DMAC, SCI, and IIC2 register settings and P16DDR bit setting.

Module Name	Pin Function	Setting			
		DMAC	IIC2	SCI	I/O Port
		$\overline{\text{DACK1A\_OE}}$	SDA0_OE	SCK3_OE	P16DDR
DMAC	$\overline{\text{DACK1}}$ -A output	1	-	—	—
IIC2	SDA0 I/O	0	1	—	—
SCI	SCK3 I/O	0	0	1	—
I/O port	P16 output	0	0	0	1
	P16 input (initial value)	0	0	0	0

**(3) P15/RxD3/ $\overline{\text{TEND1}}$ -A/ $\overline{\text{IRQ5}}$ -A/TCLKB-B/SCL1**

The pin function is switched as shown below according to the combination of the DMAC and IIC2 register settings and P15DDR bit setting.

Module Name	Pin Function	Setting		
		DMAC	IIC2	I/O Port
		$\overline{\text{TEND1A\_OE}}$	SCL1_OE	P15DDR
DMAC	$\overline{\text{TEND1}}$ -A output	1	—	—
IIC2	SCL1 I/O	0	1	—
I/O port	P15 output	0	0	1
	P15 input (initial value)	0	0	0

**(4) P14/TxD3/DREQ1-A/ $\overline{\text{IRQ4}}$ -A/TCLKA-B/SDA1**

The pin function is switched as shown below according to the combination of the SCI and IIC2 register settings and P14DDR bit setting.

Module Name	Pin Function	Setting		
		SCI	IIC2	I/O Port
		TxD3_OE	SDA1_OE	P14DDR
SCI	TxD3 output	1	—	—
IIC2	SDA1 I/O	0	1	—
I/O port	P14 output	0	0	1
	P14 input (initial value)	0	0	0

**(5) P13/ $\overline{\text{ADTRG0}}$ / $\overline{\text{IRQ3}}$ -A**

The pin function is switched as shown below according to the P13DDR bit setting.

Module Name	Pin Function	Setting
		I/O Port
		P13DDR
I/O port	P13 output	1
	P13 input (initial value)	0

**(6) P12/SCK2/ $\overline{\text{DACK0-A}}$ / $\overline{\text{IRQ2-A}}$** 

The pin function is switched as shown below according to the combination of the DMAC and SCI register settings and P12DDR bit setting.

Module Name	Pin Function	Setting		
		DMAC	SCI	I/O Port
		$\overline{\text{DACK0A\_OE}}$	SCK2_OE	P12DDR
DMAC	$\overline{\text{DACK0-A}}$ output	1	—	—
SCI	SCK2 output	0	1	—
I/O port	P12 output	0	0	1
	P12 input (initial value)	0	0	0

**(7) P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}$** 

The pin function is switched as shown below according to the combination of the DMAC register setting and P11DDR bit setting.

Module Name	Pin Function	Setting	
		DMAC	I/O Port
		$\overline{\text{TEND0A\_OE}}$	P11DDR
DMAC	$\overline{\text{TEND0-A}}$ output	1	—
I/O port	P11 output	0	1
	P11 input (initial value)	0	0

**(8) P10/TxD2/DREQ0-A/ $\overline{\text{IRQ0-A}}$** 

The pin function is switched as shown below according to the combination of the SCI register setting and P10DDR bit setting.

Module Name	Pin Function	Setting	
		SCI	I/O Port
		TxD2_OE	P10DDR
SCI	TxD2 output	1	—
I/O port	P10 output	0	1
	P10 input (initial value)	0	0

**11.2.2 Port 2****(1) P27/PO7/TIOCA5/TIOCB5/ $\overline{\text{IRQ15}}$** 

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P27DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCB5_OE	PO7_OE	P27DDR
TPU	TIOCB5 output	1	—	—
PPG	PO7 output	0	1	—
I/O port	P27 output	0	0	1
	P27 input (initial value)	0	0	0

**(2) P26/PO6/TIOCA5/TMO1/TxD1/ $\overline{\text{TRQ14}}$** 

The pin function is switched as shown below according to the combination of the TPU, TMR, SCI, and PPG register settings and P26DDR bit setting.

Module Name	Pin Function	Setting				
		TPU	TMR	SCI	PPG	I/O Port
		TIOCA5_OE	TMO1_OE	TxD1_OE	PO6_OE	P26DDR
TPU	TIOCA5 output	1	—	—	—	—
TMR	TMO1 output	0	1	—	—	—
SCI	TxD1 output	0	0	1	—	—
PPG	PO6 output	0	0	0	1	—
I/O port	P26 output	0	0	0	0	1
	P26 input (initial value)	0	0	0	0	0

**(3) P25/PO5/TIOCA4/TMC11/RxD1/ $\overline{\text{TRQ13-A}}$** 

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P25DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCA4_OE	PO5_OE	P25DDR
TPU	TIOCA4 output	1	—	—
PPG	PO5 output	0	1	—
I/O port	P25 output	0	0	1
	P25 input (initial value)	0	0	0

**(4) P24/PO4/TIOCA4/TIOCB4/TMRI1/SCK1/ $\overline{\text{IRQ12}}$ -A**

The pin function is switched as shown below according to the combination of the TPU, SCI, and PPG register settings and P24DDR bit setting.

Module Name	Pin Function	Setting			
		TPU	SCI	PPG	I/O Port
		TIOCB4_OE	SCK1_OE	PO4_OE	P24DDR
TPU	TIOCB4 output	1	—	—	—
SCI	SCK1 output	0	1	—	—
PPG	PO4 output	0	0	1	—
I/O port	P24 output	0	0	0	1
	P24 input (initial value)	0	0	0	0

**(5) P23/PO3/TIOCC3/TIOCD3/ $\overline{\text{IRQ11}}$ -A**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P23DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCD3_OE	PO3_OE	P23DDR
TPU	TIOCD3 output	1	—	—
PPG	PO3 output	0	1	—
I/O port	P23 output	0	0	1
	P23 input (initial value)	0	0	0

**(6) P22 /PO2/TIOCC3/TMO0/TxD0/ $\overline{\text{IRQ10}}$ -A**

The pin function is switched as shown below according to the combination of the TPU, TMR, SCI, and PPG register settings and P22DDR bit setting.

Module Name	Pin Function	Setting				
		TPU	TMR	SCI	PPG	I/O Port
		TIOCC3_OE	TMO0_OE	TxD0_OE	PO2_OE	P22DDR
TPU	TIOCC3 output	1	—	—	—	—
TMR	TMO0 output	0	1	—	—	—
SCI	TxD0 output	0	0	1	—	—
PPG	PO2 output	0	0	0	1	—
I/O port	P22 output	0	0	0	0	1
	P22 input (initial value)	0	0	0	0	0

**(7) P21/PO1/TIOCA3/TMCI0/RxD0/ $\overline{\text{IRQ9}}$ -A**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P21DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCA3_OE	PO1_OE	P21DDR
TPU	TIOCA3 output	1	—	—
PPG	PO1 output	0	1	—
I/O port	P21 output	0	0	1
	P21 input (initial value)	0	0	0



**(8) P20/PO0/TIOCA3/TIOCB3/TMRI0/SCK0/IRQ8-A**

The pin function is switched as shown below according to the combination of the TPU, SCI, and PPG register settings and P20DDR bit setting.

Module Name	Pin Function	Setting			
		TPU	SCI	PPG	I/O Port
		TIOCB3_OE	SCK0_OE	PO0_OE	P20DDR
TPU	TIOCB3 output	1	—	—	—
SCI	SCK0 output	0	1	—	—
PPG	PO0 output	0	0	1	—
I/O port	P20 output	0	0	0	1
	P20 input (initial value)	0	0	0	0

**11.2.3 Port 3****(1) P37/PO15/TIOCA2/TIOCB2/TCLKD-A**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P37DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCB2_OE	PO15_OE	P37DDR
TPU	TIOCB2 output	1	—	—
PPG	PO15 output	0	1	—
I/O port	P37 output	0	0	1
	P37 input (initial value)	0	0	0

**(2) P36/PO14/TIOCA2**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P36DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCA2_OE	PO14_OE	P36DDR
TPU	TIOCA2 output	1	—	—
PPG	PO14 output	0	1	—
I/O port	P36 output	0	0	1
	P36 input (initial value)	0	0	0

**(3) P35/PO13/TIOCA1/TIOCB1/TCLKC-A/ $\overline{\text{DACK1-B}}$** 

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P35DDR bit setting.

Module Name	Pin Function	Setting			
		DMAC	TPU	PPG	I/O Port
		$\overline{\text{DACK1B}}_{\text{OE}}$	TIOCB1_OE	PO13_OE	P35DDR
DMAC	$\overline{\text{DACK1-B}}$ output	1	—	—	—
TPU	TIOCB1 output	0	1	—	—
PPG	PO13 output	0	0	1	—
I/O port	P35 output	0	0	0	1
	P35 input (initial value)	0	0	0	0

**(4) P34/PO12/TIOCA1/TEND1-B**

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P34DDR bit setting.

Module Name	Pin Function	Setting			
		DMAC	TPU	PPG	I/O Port
		TEND1B_OE	TIOCA1_OE	PO12_OE	P34DDR
DMAC	TEND1-B output	1	—	—	—
TPU	TIOCA1 output	0	1	—	—
PPG	PO12 output	0	0	1	—
I/O port	P34 output	0	0	0	1
	P34 input (initial value)	0	0	0	0

**(5) P33/PO11/TIOCC0/TIOCD0/TCLKB-A/DREQ1-B/CS3/CS7-A**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P33DDR bit setting.

Module Name	Pin Function	Setting				
		I/O Port		TPU	PPG	I/O Port
		CS3_OE	CS7A_OE	TIOCD0_OE	PO11_OE	P33DDR
Bus controller	CS3 output*	1	—	—	—	—
	CS7A output*	—	1	—	—	—
TPU	TIOCD0 output	0	0	1	—	—
PPG	PO11 output	0	0	0	1	—
I/O port	P33 output	0	0	0	0	1
	P33 input (initial value)	0	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(6) P32/PO10/TIOCC0/TCLKA-A/ $\overline{\text{DACK0-B}}$ / $\overline{\text{CS2-A}}$ / $\overline{\text{CS6-A}}$** 

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P32DDR bit setting.

Module Name	Pin Function	Setting					
		I/O Port		DMAC	TPU	PPG	I/O Port
		$\overline{\text{CS2A\_OE}}$	$\overline{\text{CS6A\_OE}}$	$\overline{\text{DACK0B\_OE}}$	TIOCC0_OE	PO10_OE	P32DDR
Bus controller	$\overline{\text{CS2-A}}$ output*	1	—	—	—	—	—
	$\overline{\text{CS6-A}}$ output*	—	1	—	—	—	—
DMAC	$\overline{\text{DACK0-B}}$ output	0	0	1	—	—	—
TPU	TIOCC0 output	0	0	0	1	—	—
PPG	PO10 output	0	0	0	0	1	—
I/O port	P32 output	0	0	0	0	0	1
	P32 input (initial value)	0	0	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(7) P31/PO9/TIOCA0/TIOCB0/TEND0-B/CS1/CS2-B/CS5-A/CS6-B/CS7-B**

The pin function is switched as shown below according to the combination of the DMAC, TPU, and PPG register settings and P31DDR bit setting.

Module Name	Pin Function	Setting								
		I/O Port					DMAC	TPU	PPG	I/O Port
		CS1_OE	CS2B_OE	CS5A_OE	CS6B_OE	CS7B_OE	TEND0B_OE	TIOCB0_OE	PO9_OE	P31DDR
Bus controller	CS1 output*	1	—	—	—	—	—	—	—	—
	CS2-B output*	—	1	—	—	—	—	—	—	—
	CS5-A output*	—	—	1	—	—	—	—	—	—
	CS6-B output*	—	—	—	1	—	—	—	—	—
	CS7-B output*	—	—	—	—	1	—	—	—	—
DMAC	TEND0-B output	0	0	0	0	0	1	—	—	—
TPU	TIOCB0 output	0	0	0	0	0	0	1	—	—
PPG	PO9 output	0	0	0	0	0	0	0	1	—
I/O port	P31 output	0	0	0	0	0	0	0	0	1
	P31 input (initial value)	0	0	0	0	0	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(8) P30/PO8/ $\overline{\text{DREQ0}}$ -B/ $\overline{\text{TIOCA0}}$ / $\overline{\text{CS0}}$ / $\overline{\text{CS4-A}}$ / $\overline{\text{CS5-B}}$** 

The pin function is switched as shown below according to the combination of the TPU and PPG register settings and P30DDR bit setting.

Module Name	Pin Function	Setting					
		I/O Port		TPU	PPG	I/O Port	
		$\overline{\text{CS0\_OE}}$	$\overline{\text{CS4A\_OE}}$	$\overline{\text{CS5B\_OE}}$	$\overline{\text{TIOCA0\_OE}}$	$\overline{\text{PO8\_OE}}$	P30DDR
Bus controller	$\overline{\text{CS0}}$ output* (initial value E)	1	—	—	—	—	—
	$\overline{\text{CS4-A}}$ output*	—	1	—	—	—	—
	$\overline{\text{CS5-B}}$ output*	—	—	1	—	—	—
TPU	$\overline{\text{TIOCA0}}$ output	0	0	0	1	—	—
PPG	PO8 output	0	0	0	0	1	—
I/O port	P30 output	0	0	0	0	0	1
	P30 input (initial value S)	0	0	0	0	0	0

[Legend]

Initial value E: Initial value in on-chip ROM disabled external extended mode

Initial value S: Initial value in other modes

Note: \* Valid in external extended mode (EXPE = 1)

**11.2.4 Port 5****(1) P57/AN7/DA1/ $\overline{\text{IRQ7}}$ -B**

Module Name	Pin Function
-------------	--------------

D/A converter	DA1 output
---------------	------------

**(2) P56/AN6/DA0/ $\overline{\text{IRQ6}}$ -B**

Module Name	Pin Function
-------------	--------------

D/A converter	DA0 output
---------------	------------

## 11.2.5 Port 6

### (1) P65/TMO3/ $\overline{\text{IRQ13}}$

The pin function is switched as shown below according to the combination of the TMR register setting and P65DDR bit setting.

Module Name	Pin Function	Setting	
		TMR	I/O Port
		TMO3_OE	P65DDR
TMR	TMO3 output	1	—
I/O port	P65 output	0	1
	P65 input (initial value)	0	0

### (2) P64/TMCI3/ $\overline{\text{IRQ12-B}}$

The pin function is switched as shown below according to the P64DDR bit setting.

Module Name	Pin Function	Setting	
		I/O Port	
		P64DDR	
I/O port	P64 output	1	
	P64 input (initial value)	0	

**(3) P63/TMRI3/ $\overline{\text{IRQ11}}$** 

The pin function is switched as shown below according to the P63DDR bit setting.

Module Name	Pin Function	Setting	
		I/O Port	P63DDR
I/O port	P63 output	1	
	P63 input (initial value)	0	

**(4) P62/TMO2/SCK4/ $\overline{\text{IRQ10}}$** 

The pin function is switched as shown below according to the combination of the TMR and SCI register settings and P62DDR bit setting.

Module Name	Pin Function	Setting		
		TMR	SCI	I/O Port
		TMO2_OE	SCK4_OE	P62DDR
TMR	TMO2 output	1	—	—
SCI	SCK4 output	0	1	—
I/O port	P62 output	0	0	1
	P62 input (initial value)	0	0	0



**(5) P61/TMCI2/RxD4/ $\overline{\text{IRQ9}}$ -B**

The pin function is switched as shown below according to the P61DDR bit setting.

Module Name	Pin Function	Setting	
		I/O Port	P61DDR
I/O port	P61 output		1
	P61 input (initial value)		0

**(6) P60/TMRI2/TxD4/ $\overline{\text{IRQ8}}$ -B**

The pin function is switched as shown below according to the combination of the SCI register setting and P60DDR bit setting.

Module Name	Pin Function	Setting	
		SCI TxD4_OE	I/O Port P60DDR
SCI	TxD4 output	1	—
I/O port	P60 output	0	1
	P60 input (initial value)	0	0

## 11.2.6 Port A

### (1) PA7/B $\phi$

The pin function is switched as shown below according to the PA7DDR bit setting.

Module Name	Pin Function	Setting	
		I/O Port	
		PA7DDR	
I/O port	B $\phi$ output (initial value E)	1	
	PA7 input (initial value S)	0	

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

### (2) PA6/ $\overline{AS}$ / $\overline{AH}$ / $\overline{BS-B}$

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, bus controller register, port function control register (PFCR), and PA6DDR bit settings.

Module Name	Pin Function	Setting			
		Bus Controller		I/O Port	
		$\overline{AH\_OE}$	$\overline{BS-B\_OE}$	$\overline{AS\_OE}$	PA6DDR
Bus controller	$\overline{AH}$ output*	1	—	—	—
	$\overline{BS-B}$ output*	0	1	—	—
	$\overline{AS}$ output* (initial value E)	0	0	1	—
I/O port	PA6 output	0	0	0	1
	PA6 input (initial value S)	0	0	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

Note: \* Valid in external extended mode (EXPE = 1)

**(3) PA5/ $\overline{RD}$** 

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, and PA5DDR bit settings.

Module Name	Pin Function	Setting	
		MCU Operating Mode	I/O Port
		EXPE	PA5DDR
Bus controller	$\overline{RD}$ output* (initial value E)	1	—
I/O port	PA5 output	0	1
	PA5 input (initial value S)	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

Note: \* Valid in external extended mode (EXPE = 1)

**(4) PA4/ $\overline{LHWR}$ / $\overline{LUB}$** 

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, bus controller register, port function control register (PFCR), and PA4DDR bit settings.

Module Name	Pin Function	Setting		
		Bus Controller	I/O Port	
		$\overline{LUB\_OE}$ * <sup>2</sup>	$\overline{LHWR\_OE}$ * <sup>2</sup>	PA4DDR
Bus controller	$\overline{LUB}$ output* <sup>1</sup>	1	—	—
	$\overline{LHWR}$ output* <sup>1</sup> (initial value E)	—	1	—
I/O port	PA4 output	0	0	1
	PA4 input (initial value S)	0	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

Notes: 1. Valid in external extended mode (EXPE = 1)

2. When the byte control SRAM space is accessed while the byte control SRAM space is specified or while LHWR $\overline{OE}$  = 1, this pin functions as the  $\overline{LUB}$  output; otherwise, the  $\overline{LHWR}$  output.

**(5) PA3/ $\overline{\text{LLB}}$ / $\overline{\text{LLWR}}$ / $\overline{\text{LLB}}$** 

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, bus controller register, and PA3DDR bit settings.

Module Name	Pin Function	Setting		
		Bus Controller		I/O Port
		$\overline{\text{LLB}}_{\text{OE}}^{*2}$	$\overline{\text{LLWR}}_{\text{OE}}^{*2}$	PA3DDR
Bus controller	$\overline{\text{LLB}}$ output* <sup>1</sup>	1	—	—
	$\overline{\text{LLWR}}$ output* <sup>1</sup> (initial value E)	—	1	—
I/O port	PA3 output	0	0	1
	PA3 input (initial value S)	0	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

Notes: 1. Valid in external extended mode (EXPE = 1)

2. If the byte control SRAM space is accessed, this pin functions as the  $\overline{\text{LLB}}$  output; otherwise, the  $\overline{\text{LLWR}}$ .

**(6) PA2/ $\overline{\text{BREQ}}$ / $\overline{\text{WAIT}}$** 

The pin function is switched as shown below according to the combination of the bus controller register setting and PA2DDR bit setting.

Module Name	Pin Function	Setting		
		Bus Controller		I/O Port
		BCR_BRLE	BCR_WAITE	PA2DDR
Bus controller	$\overline{\text{BREQ}}$ input	1	—	—
	$\overline{\text{WAIT}}$ input	0	1	—
I/O port	PA2 output	0	0	1
	PA2 input (initial value)	0	0	0

**(7) PA1/ $\overline{\text{BACK}}$ / $\overline{\text{RD}}$ / $\overline{\text{WR}}$** 

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, bus controller register, port function control register (PFCR), and PA1DDR bit settings.

Module Name	Pin Function	Setting			
		Bus Controller		I/O Port	
		$\overline{\text{BACK}}_{\text{OE}}$	Byte control SRAM Selection	$\overline{\text{RD}}/\overline{\text{WR}}_{\text{OE}}$	PA1DDR
Bus controller	$\overline{\text{BACK}}$ output*	1	—	—	—
	$\overline{\text{RD}}/\overline{\text{WR}}$ output*	0	1	—	—
		0	0	1	—
I/O port	PA1 output	0	0	0	1
	PA1 input (initial value)	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(8) PA0/ $\overline{\text{BREQO}}$ / $\overline{\text{BS-A}}$** 

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, bus controller register, port function control register (PFCR), and PA0DDR bit settings.

Module Name	Pin Function	Setting		
		I/O Port	Bus Controller	I/O Port
		$\overline{\text{BSA}}_{\text{OE}}$	$\overline{\text{BREQO}}_{\text{OE}}$	PA0DDR
Bus controller	$\overline{\text{BS-A}}$ output*	1	—	—
	$\overline{\text{BREQO}}$ output*	0	1	—
I/O port	PA0 output	0	0	1
	PA0 input (initial value)	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

### 11.2.7 Port D

#### (1) PD7/A7, PD6/A6, PD5/A5, PD4/A4, PD3/A3, PD2/A2, PD1/A1, PD0/A0

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, PEnDDR bit settings.

Module Name	Pin Function	MCU Operating Mode	Setting
			I/O Port PEnDDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PEn output	Single-chip mode*	1
	PEn input (initial value)	Modes other than on-chip ROM disabled extended mode	0

[Legend]

n: 0 to 7

Note: \* Address output is enabled by setting PDnDDR = 1 in external extended mode (EXPE = 1).

### 11.2.8 Port E

#### (1) PE7/A15

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, and PE7DDR bit settings.

Module Name	Pin Function	MCU Operating Mode	Setting
			I/O Port PE7DDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PE7 output	Single-chip mode*	1
	PE7 input (initial value)	Modes other than on-chip ROM disabled extended mode	0

Note: \* Address output is enabled by setting PE7DDR = 1 in external extended mode (EXPE = 1).

**(2) PE6/A14**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, and PE6DDR bit settings.

Module Name	Pin Function	Setting	
		MCU Operating Mode	I/O Port PE6DDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PE6 output	Single-chip mode*	1
	PE6 input (initial value)	Modes other than on-chip ROM disabled extended mode	0

Note: \* Address output is enabled by setting PE6DDR = 1 in external extended mode (EXPE = 1).

**(3) PE5/A13**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, and PE5DDR bit settings.

Module Name	Pin Function	Setting	
		MCU Operating Mode	I/O Port PE5DDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PE5 output	Single-chip mode*	1
	PE5 input (initial value)	Modes other than on-chip ROM disabled extended mode	0

Note: \* Address output is enabled by setting PE5DDR = 1 in external extended mode (EXPE = 1).

**(4) PE4/A12, PE3/A11, PE3/A11, PE2/A10, PE1/A9, PE0/A8**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, and PEnDDR bit settings.

Module Name	Pin Function	MCU Operating Mode	Setting	
			I/O Port	PEnDDR
Bus controller	Address output	On-chip ROM disabled extended mode	—	
		On-chip ROM enabled extended mode	1	
I/O port	PEn output	Single-chip mode*	1	
	PEn input (initial value)	Modes other than on-chip ROM disabled extended mode	0	

[Legend]

n: 0 to 4

Note: \* Address output is enabled by setting PEnDDR = 1 in external extended mode (EXPE = 1).

**11.2.9 Port F****(1) PF4/A20**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, port function control register (PFCR), and PF4DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			I/O Port A20_OE	I/O Port PF4DDR
On-chip ROM disabled extended mode	Bus controller	A20 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A20 output*	1	—
		I/O port	0	1
	I/O port	PF4 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)



**(2) PF3/A19**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, port function control register (PFCR), and PF3DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			I/O Port	I/O Port
			A19_OE	PF3DDR
On-chip ROM disabled extended mode	Bus controller	A19 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A19 output*	1	—
	I/O port	PF3 output	0	1
		PF3 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(3) PF2/A18**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, port function control register (PFCR), and PF2DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			I/O Port	I/O Port
			A18_OE	PF2DDR
On-chip ROM disabled extended mode	Bus controller	A18 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A18 output*	1	—
	I/O port	PF2 output	0	1
		PF2 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(4) PF1/A17**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, port function control register (PFCR), and PF1DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			I/O Port	I/O Port
			A17_OE	PF1DDR
On-chip ROM disabled extended mode	Bus controller	A17 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A17 output*	1	—
	I/O port	PF1 output	0	1
		PF1 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

**(5) PF0/A16**

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, port function control register (PFCR), and PF0DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			I/O Port	I/O Port
			A16_OE	PF0DDR
On-chip ROM disabled extended mode	Bus controller	A16 output	—	—
Modes other than on-chip ROM disabled extended mode	Bus controller	A16 output*	1	—
	I/O port	PF0 output	0	1
		PF0 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

## 11.2.10 Port H

### (1) PI7/D15, PI6/D14, PI5/D13, PI4/D12, PI3/D11, PI2/D10, PI1/D9, PI0/D8

The pin function is switched as shown below according to the combination of the operating mode, EXPE bit, and PHnDDR bit settings.

Module Name	Pin Function	Setting	
		MCU Operating Mode	I/O Port
		EXPE	PHnDDR
Bus controller	Data I/O* (initial value E)	1	—
I/O port	PHn output	0	1
	PHn input (initial value S)	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

n: 0 to 7

Note: \* Valid in external extended mode (EXPE = 1)

### 11.2.11 Port I

#### (1) PI7/D15/TMO7, PI6/D14/TMO6, PI5/D13/TMO5, PI4/D12/TMO4

The pin function is switched as shown below according to the combination of the operating mode, bus mode, EXPE bit, TMR register, and PInDDR bit settings.

Module Name	Pin Function	Setting		
		Bus Controller	TMR	I/O Port
		16-Bit Bus Mode	TMO <sub>n</sub> _OE	PInDDR
Bus controller	Data I/O* (initial value E)	1	0	0
TMR	TMO <sub>n</sub> output	0	1	—
I/O port	PIn output	0	0	1
	PIn input (initial value S)	0	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

n: 4 to 7

Note: \* Valid in external extended mode (EXPE = 1)

**(2) PI3/D11, PI2/D10, PI1/D9, PI0/D8**

The pin function is switched as shown below according to the combination of the operating mode, bus mode, EXPE bit, and PInDDR bit settings.

Module Name	Pin Function	Setting	
		Bus Controller	I/O Port
		16-Bit Bus Mode	PInDDR
Bus controller	Data I/O* (initial value E)	1	—
I/O port	PIn output	0	1
	PIn input (initial value S)	0	0

[Legend]

Initial value E: Initial value in external extended mode

Initial value S: Initial value in single-chip mode

n: 0 to 3

Note: \* Valid in external extended mode (EXPE = 1)

**Table 11.5 Available Output Signals and Settings in Each Port**

Port	Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P1	7	SCL0_OE	SCL0	ICCRA.ICE = 1
	6	DACK1A_OE	DACK1	FPCR7.DMAS1[A,B] = 00 DACR.AMS = 1, DMDR.DACKE = 1
		SDA0_OE	SDA0	ICCRA.ICE = 1
		SCK3_OE	SCK3	When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 while SMR.GM = 0, SCR.CKE[1, 0] = 01 or while SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE[1, 0] = 01 or while SMR.C/A = 1, SCR.CKE1 = 0
5	TEND1A_OE	TEND1	FPCR7.DMAS1[A,B] = 00	DMDR.TENDE = 1
	SCL1_OE	SCL1		ICCRA.ICE = 1
4	TxD3_OE	TxD3		SCR.TE = 1, IrCR.IrE = 0
	SDA1_OE	SDA1		ICCRA.ICE = 1
3	—	—	—	—
2	DACK0A_OE	DACK0	FPCR7.DMAS0[A,B] = 00	DACR.AMS = 1, DMDR.DACKE = 1
	SCK2_OE	SCK2		When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 while SMR.GM = 0, SCR.CKE[1, 0] = 01 or while SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE[1, 0] = 01 or while SMR.C/A = 1, SCR.CKE1 = 0
1	TEND0A_OE	TEND0	PFCR7.DMAS0[A,B] = 00	DMDR.TENDE = 1
0	TxD2_OE	TxD2		SCR.TE = 1

Port	Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P2 7	TIOCB5_OE	TIOCB5		TPU.TIOR5.IOB3 = 0, TPU.TIOR5.IOB[1,0] = 01/10/11
	PO7_OE	PO7		NDERL.NDER7 = 1
6	TIOCA5_OE	TIOCA5		TPU.TIOR5.IOA3 = 0, TPU.TIOR5.IOA[1,0] = 01/10/11
	TMO1_OE	TMO1		TCSR.OS3, 2 = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	TxD1_OE	TxD1		SCR.TE = 1
	PO6_OE	PO6		NDERL.NDER6 = 1
5	TIOCA4_OE	TIOCA4		TPU.TIOR4.IOA3 = 0, TPU.TIOR4.IOA[1,0] = 01/10/11
	PO5_OE	PO5		NDERL.NDER5 = 1
4	TIOCB4_OE	TIOCB4		TPU.TIOR4.IOB3 = 0, TPU.TIOR4.IOB[1,0] = 01/10/11
	SCK1_OE	SCK1		When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 while SMR.GM = 0, SCR.CKE[1, 0] = 01 or while SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE[1, 0] = 01 or while SMR.C/A = 1, SCR.CKE1 = 0
	PO4_OE	PO4		NDERL.NDER4 = 1
3	TIOCD3_OE	TIOCD3		TPU.TMDR.BFB = 0, TPU.TIORL3.IOD3 = 0, TPU.TIORL3.IOD[1,0] = 01/10/11
	PO3_OE	PO3		NDERL.NDER3 = 1
2	TIOCC3_OE	TIOCC3		TPU.TMDR.BFA = 0, TPU.TIORL3.IOC3 = 0, TPU.TIORL3.IOD[1,0] = 01/10/11
	TMO0_OE	TMO0		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	TxD0_OE	TxD0		SCR.TE = 1
	PO2_OE	PO2		NDERL.NDER2 = 1
1	TIOCA3_OE	TIOCA3		TPU.TIORH3.IOA3 = 0, TPU.TIORH3.IOA[1,0] = 01/10/11
	PO1_OE	PO1		NDERL.NDER1 = 1

Port	Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P2	0	TIOCB3_OE	TIOCB3	TPU.TIORH3.IOB3 = 0, TPU.TIORH3.IOB[1,0] = 01/10/11
		SCK0_OE	SCK0	When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 while SMR.GM = 0, SCR.CKE[1, 0] = 01 or while SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE[1, 0] = 01 or while SMR.C/A = 1, SCR.CKE1 = 0
		PO0_OE	PO0	NDERL.NDER0 = 1
P3	7	TIOCB2_OE	TIOCB2	TPU.TIOR2.IOB3 = 0, TPU.TIOR2.IOB[1,0] = 01/10/11
		PO15_OE	PO15	NDERH.NDER15 = 1
	6	TIOCA2_OE	TIOCA2	TPU.TIOR2.IOA3 = 0, TPU.TIOR2.IOA[1,0] = 01/10/11
		PO14_OE	PO14	NDERH.NDER14 = 1
	5	$\overline{\text{DACK1B}}_O\text{E}$	$\overline{\text{DACK1}}$	PFCR7.DMAS1[A,B] = 01 DACR.AMS = 1, DMDR.DACKE = 1
		TIOCB1_OE	TIOCB1	TPU.TIOR1.IOB3 = 0, TPU.TIOR1.IOB[1,0] = 01/10/11
		PO13_OE	PO13	NDERH.NDER13 = 1
	4	$\overline{\text{TEND1B}}_O\text{E}$	$\overline{\text{TEND1}}$	PFCR7.DMAS1[A,B] = 01 DMDR.TENDE = 1
		TIOCA1_OE	TIOCA1	TPU.TIOR1.IOA3 = 0, TPU.TIOR1.IOA[1,0] = 01/10/11
		PO12_OE	PO12	NDERH.NDER12 = 1
	3	$\overline{\text{CS3}}_O\text{E}$	$\overline{\text{CS3}}$	SYSCR.EXPE = 1, PFCR0.CS3E = 1
		$\overline{\text{CS7A}}_O\text{E}$	$\overline{\text{CS7}}$	PFCR1.CS7S[A,B] = 00 SYSCR.EXPE = 1, PFCR0.CS7E = 1
	TIOCD0_OE	TIOCD0	TPU.TMDR.BFB = 0, TPU.TIORL0.IOD3 = 0, TPU.TIORL0.IOD[1,0] = 01/10/11	
	PO11_OE	PO11	NDERH.NDER11 = 1	
2	$\overline{\text{CS2A}}_O\text{E}$	$\overline{\text{CS2}}$	PFCR2.CS2S = 0 SYSCR.EXPE = 1, PFCR0.CS2E = 1	
	$\overline{\text{CS6A}}_O\text{E}$	$\overline{\text{CS6}}$	PFCR1.CS6S[A,B] = 00 SYSCR.EXPE = 1, PFCR0.CS6E = 1	
	$\overline{\text{DACK0B}}_O\text{E}$	$\overline{\text{DACK0}}$	PFCR7.DMAS0[A,B] = 01 DACR.AMS = 1, DMDR.DACKE = 1	
	TIOCC0_OE	TIOCC0	TPU.TMDR.BFA = 0, TPU.TIORL0.IOC3 = 0, TPU.TIORL0.IOD[1,0] = 01/10/11	
	PO10_OE	PO10	NDERH.NDER10 = 1	



Port	Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P3	1	$\overline{CS1\_OE}$	$\overline{CS1}$	SYSCR.EXPE = 1, PFCR0.CS1E = 1
		$\overline{CS2B\_OE}$	$\overline{CS2}$	PFCR2.CS2S = 1 SYSCR.EXPE = 1, PFCR0.CS2E = 1
		$\overline{CS5A\_OE}$	$\overline{CS5}$	PFCR1.CS5S[A,B] = 00 SYSCR.EXPE = 1, PFCR0.CS5E = 1
		$\overline{CS6B\_OE}$	$\overline{CS6}$	PFCR1.CS6S[A,B] = 01 SYSCR.EXPE = 1, PFCR0.CS6E = 1
		$\overline{CS7B\_OE}$	$\overline{CS7}$	PFCR1.CS7S[A,B] = 01 SYSCR.EXPE = 1, PFCR0.CS7E = 1
		$\overline{TEND0B\_OE}$	$\overline{TEND0}$	PFCR7.DMAS0[A,B] = 01 DMDR.TENDE = 1
		$\overline{TIOCB0\_OE}$	$\overline{TIOCB0}$	TPU.TIORH0.IOB3 = 0, TPU.TIORH0.IOB[1,0] = 01/10/11
	PO9_OE	PO9	NDERH.NDER9 = 1	
0		$\overline{CS0\_OE}$	$\overline{CS0}$	SYSCR.EXPE = 1, PFCR0.CS0E = 1
		$\overline{CS4\_OE}$	$\overline{CS4}$	SYSCR.EXPE = 1, PFCR0.CS4E = 1
		$\overline{CS5B\_OE}$	$\overline{CS5}$	PFCR1.CS5S[A,B] = 01 SYSCR.EXPE = 1, PFCR0.CS5E = 1
		$\overline{TIOCA0\_OE}$	$\overline{TIOCA0}$	TPU.TIORH0.IOA3 = 0, TPU.TIORH0.IOA[1,0] = 01/10/11
	PO8_OE	PO8	NDERH.NDER8 = 1	
P6	5	TMO3_OE	TMO3	TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	2	TMO2_OE	TMO2	TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
		SCK4_OE	SCK4	When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 while SMR.GM = 0, SCR.CKE[1, 0] = 01 or while SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE[1, 0] = 01 or while SMR.C/A = 1, SCR.CKE1 = 0
0	TxD4_OE	TxD4	SCR.TE = 1	

Port	Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PA	7	B $\phi$ _OE	B $\phi$	PADDR.PA7DDR = 1, SCKCR.POSEL1 = 0
	6	AH_OE	AH	SYSCR.EXPE = 1, MPXCR.MPXEn (n = 7 to 3) = 1
		BSB_OE	BS	PFCR2.BSS = 1, SYSCR.EXPE = 1, PFCR2.BSE = 1
		AS_OE	AS	SYSCR.EXPE = 1, PFCR2.ASOE = 1
	5	RD_OE	RD	SYSCR.EXPE = 1
	4	LUB_OE	LUB	SYSCR.EXPE = 1, PFCR6.LHWROE = 1, or SRAMCR.BCSELn = 1
		LHWR_OE	LHWR	SYSCR.EXPE = 1, PFCR6.LHWROE = 1
	3	LLB_OE	LLB	SYSCR.EXPE = 1, SRAMCR.BCSELn = 1
		LLWR_OE	LLWR	SYSCR.EXPE = 1
	1	BACK_OE	BACK	SYSCR.EXPE = 1, BCR1.BRLE = 1
(RD/WR)_OE		RD/WR	SYSCR.EXPE = 1, PFCR2.REWRE = 1, or SRAMCR.BCSELn = 1	
0	BSA_OE	BS	PFCR2.BSS = 0, SYSCR.EXPE = 1, PFCR2.BSE = 1	
	BREQO_OE	BREQO	SYSCR.EXPE = 1, BCR1.BRLE = 1, BCR1.BREQOE = 1	
PD	7	A7_OE	A7	SYSCR.EXPE = 1, PDDDR.PD7DDR = 1
	6	A6_OE	A6	SYSCR.EXPE = 1, PDDDR.PD6DDR = 1
	5	A5_OE	A5	SYSCR.EXPE = 1, PDDDR.PD5DDR = 1
	4	A4_OE	A4	SYSCR.EXPE = 1, PDDDR.PD4DDR = 1
	3	A3_OE	A3	SYSCR.EXPE = 1, PDDDR.PD3DDR = 1
	2	A2_OE	A2	SYSCR.EXPE = 1, PDDDR.PD2DDR = 1
	1	A1_OE	A1	SYSCR.EXPE = 1, PDDDR.PD1DDR = 1
	0	A0_OE	A0	SYSCR.EXPE = 1, PDDDR.PD0DDR = 1
PE	7	A15_OE	A15	SYSCR.EXPE = 1, PDDDR.PE7DDR = 1
	6	A14_OE	A14	SYSCR.EXPE = 1, PDDDR.PE6DDR = 1
	5	A13_OE	A13	SYSCR.EXPE = 1, PDDDR.PE5DDR = 1
	4	A12_OE	A12	SYSCR.EXPE = 1, PDDDR.PE4DDR = 1
	3	A11_OE	A11	SYSCR.EXPE = 1, PDDDR.PE3DDR = 1
	2	A10_OE	A10	SYSCR.EXPE = 1, PDDDR.PE2DDR = 1
	1	A9_OE	A9	SYSCR.EXPE = 1, PDDDR.PE1DDR = 1
	0	A8_OE	A8	SYSCR.EXPE = 1, PDDDR.PE0DDR = 1

Port	Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PF 4	A20_OE	A20		SYSCR.EXPE = 1, PFCR4.A20E = 1
	A19_OE	A19		SYSCR.EXPE = 1, PFCR4.A19E = 1
	A18_OE	A18		SYSCR.EXPE = 1, PFCR4.A18E = 1
	A17_OE	A17		SYSCR.EXPE = 1, PFCR4.A17E = 1
	A16_OE	A16		SYSCR.EXPE = 1, PFCR4.A16E = 1
PH 7	D7_E	D7		SYSCR.EXPE = 1
	D6_E	D6		SYSCR.EXPE = 1
	D5_E	D5		SYSCR.EXPE = 1
	D4_E	D4		SYSCR.EXPE = 1
	D3_E	D3		SYSCR.EXPE = 1
	D2_E	D2		SYSCR.EXPE = 1
	D1_E	D1		SYSCR.EXPE = 1
	D0_E	D0		SYSCR.EXPE = 1
PI 7	D15_E	D15		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	TMO7_OE	TMO7		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	D14_E	D14		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	TMO6_OE	TMO6		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	D13_E	D13		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	TMO5_OE	TMO5		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	D12_E	D12		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	TMO4_OE	TMO4		TCSR.OS[3,2] = 01/10/11 or TCSR.OS[1,0] = 01/10/11
	D11_E	D11		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	D10_E	D10		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	D9_E	D9		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	D8_E	D8		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01

## 11.3 Port Function Controller

The port function controller controls the I/O ports.

The port function controller incorporates the following registers.

- Port function control register 0 (PFCR0)
- Port function control register 1 (PFCR1)
- Port function control register 2 (PFCR2)
- Port function control register 4 (PFCR4)
- Port function control register 6 (PFCR6)
- Port function control register 7 (PFCR7)
- Port function control register 9 (PFCR9)
- Port function control register B (PFCRB)
- Port function control register C (PFCRC)

### 11.3.1 Port Function Control Register 0 (PFCR0)

PFCR0 enables/disables the  $\overline{CS}$  output.

Bit	7	6	5	4	3	2	1	0
Bit Name	CS7E	CS6E	CS5E	CS4E	CS3E	CS2E	CS1E	CS0E
Initial Value	0	0	0	0	0	0	0	Undefined*
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* 1 in external extended mode, 0 in other modes.

Bit	Bit Name	Initial Value	R/W	Description
7	CS7E	0	R/W	CS7 to CS0 Enable
6	CS6E	0	R/W	These bits enable/disable the corresponding $\overline{CSn}$ output.
5	CS5E	0	R/W	
4	CS4E	0	R/W	0: Pin functions as I/O port
3	CS3E	0	R/W	1: Pin functions as $\overline{CSn}$ output pin (n = 7 to 0)
2	CS2E	0	R/W	
1	CS1E	0	R/W	
0	CS0E	Undefined*	R/W	

Note: \* 1 in external extended mode, 0 in other modes.

### 11.3.2 Port Function Control Register 1 (PFCR1)

PFCR1 selects the  $\overline{CS}$  output pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	CS7SA	CS7SB	CS6SA	CS6SB	CS5SA	CS5SB	CS4SA	CS4SB
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	Bit Name	Initial Value	R/W	Description
7	CS7SA*	0	R/W	$\overline{CS7}$ Output Pin Select
6	CS7SB*	0	R/W	Selects the output pin for $\overline{CS7}$ when $\overline{CS7}$ output is enabled (CS7E = 1) 00: Specifies pin PB3 as $\overline{CS7}$ -A output 01: Specifies pin PB1 as $\overline{CS7}$ -B output 10: (Setting prohibited) 11: (Setting prohibited)
5	CS6SA*	0	R/W	$\overline{CS6}$ Output Pin Select
4	CS6SB*	0	R/W	Selects the output pin for $\overline{CS6}$ when $\overline{CS6}$ output is enabled (CS6E = 1) 00: Specifies pin PB2 as $\overline{CS6}$ -A output 01: Specifies pin PB1 as $\overline{CS6}$ -B output 10: (Setting prohibited) 11: (Setting prohibited)
3	CS5SA*	0	R/W	$\overline{CS5}$ Output Pin Select
2	CS5SB*	0	R/W	Selects the output pin for $\overline{CS5}$ when $\overline{CS5}$ output is enabled (CS5E = 1) 00: Specifies pin PB1 as $\overline{CS5}$ -A output 01: Specifies pin PB0 as $\overline{CS5}$ -B output 10: (Setting prohibited) 11: (Setting prohibited)

Bit	Bit Name	Initial Value	R/W	Description
1	CS4SA*	0	R/W	$\overline{\text{CS4}}$ Output Pin Select
0	CS4SB*	0	R/W	Selects the output pin for $\overline{\text{CS4}}$ when $\overline{\text{CS4}}$ output is enabled (CS4E = 1) 00: Specifies pin PB0 as $\overline{\text{CS4}}$ -A output 01: (Setting prohibited) 10: (Setting prohibited) 11: (Setting prohibited)

Note: \* If multiple  $\overline{\text{CS}}$  outputs are specified to a single pin according to the  $\overline{\text{CSn}}$  output pin select bits (n = 4 to 7), multiple  $\overline{\text{CS}}$  signals are output from the pin. For details, see section 8.5.3, Chip Select Signals.

### 11.3.3 Port Function Control Register 2 (PFCR2)

PFCR1 selects the  $\overline{\text{CS}}$  output pin, enables/disables bus control I/O, and selects the bus control I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	CS2S	BSS	BSE	—	RDWRE	ASOE	—
Initial Value	0	0	0	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6	CS2S* <sup>1</sup>	0	R/W	$\overline{\text{CS2}}$ Output Pin Select Selects the output pin for $\overline{\text{CS2}}$ when $\overline{\text{CS2}}$ output is enabled (CS2E = 1) 0: Specifies pin PB2 as $\overline{\text{CS2}}$ -A output pin 1: Specifies pin PB1 as $\overline{\text{CS2}}$ -B output pin

Bit	Bit Name	Initial Value	R/W	Description
5	BSS	0	R/W	$\overline{BS}$ Output Pin Select Selects the $\overline{BS}$ output pin 0: Specifies pin PA0 as $\overline{BS}$ -A output pin 1: Specifies pin PA6 as $\overline{BS}$ -B output pin
4	BSE	0	R/W	$\overline{BS}$ Output Enable Enables/disables the $\overline{BS}$ output 0: Disables the $\overline{BS}$ output 1: Enables the $\overline{BS}$ output
3	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
2	RD $\overline{WR}$ * <sup>2</sup>	0	R/W	RD/ $\overline{WR}$ Output Enable Enables/disables the RD/ $\overline{WR}$ output 0: Disables the RD/ $\overline{WR}$ output 1: Enables the RD/ $\overline{WR}$ output
1	ASOE	1	R/W	$\overline{AS}$ Output Enable Enables/disables the $\overline{AS}$ output 0: Specifies pin PA6 as I/O port 1: Specifies pin PA6 as $\overline{AS}$ output pin
0	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

- Notes:
1. If multiple  $\overline{CS}$  outputs are specified to a single pin according to the  $\overline{CS2}$  output pin select bit, multiple  $\overline{CS}$  signals are output from the pin. For details, see section 8.5.3, Chip Select Signals.
  2. If an area is specified as a byte control SDRAM space, the pin functions as RD/ $\overline{WR}$  output.

### 11.3.4 Port Function Control Register 4 (PFCR4)

PFCR4 enables/disables the address output.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	A20E	A19E	A18E	A17E	A16E
Initial Value	0	0	0	0/1*	0/1*	0/1*	0/1*	0/1*
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
4	A20E	0/1*	R/W	Address A20 Enable Enables/disables the address output (A20). 0: Disables the A20 output 1: Enables the A20 output
3	A19E	0/1*	R/W	Address A19 Enable Enables/disables the address output (A19). 0: Disables the A19 output 1: Enables the A19 output
2	A18E	0/1*	R/W	Address A18 Enable Enables/disables the address output (A18). 0: Disables the A18 output 1: Enables the A18 output
1	A17E	0/1*	R/W	Address A17 Enable Enables/disables the address output (A17). 0: Disables the A17 output 1: Enables the A17 output
0	A16E	0/1*	R/W	Address A16 Enable Enables/disables the address output (A16). 0: Disables the A16 output 1: Enables the A16 output

Note: \* The initial value differs according to the set operating mode: 1 for operating modes in which on-chip ROM is disabled, and 0 for those in which on-chip ROM is enabled.



### 11.3.5 Port Function Control Register 6 (PFCR6)

PFCR6 selects the TPU clock input pin.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	LHWROE	—	—	TCLKS	—	—	—
Initial Value	1	1	1	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
6	LHWROE	1	R/W	$\overline{\text{LHWR}}$ Output Enable Enables/disables $\overline{\text{LHWR}}$ output (valid in external extended mode). 0: Specifies pin PA4 as I/O port 1: Specifies pin PA4 as $\overline{\text{LHWR}}$ output pin
5	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
4	—	0	R	Reserved This is a read-only bit and cannot be modified.
3	TCLKS	0	R/W	TPU External Clock Input Pin Select Selects the TPU external clock input pins. 0: Specifies pins P32, P33, P35, and P37 as external clock inputs 1: Specifies pins P14 to P17 as external clock inputs
2 to 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

### 11.3.6 Port Function Control Register 7 (PFCR7)

PFCR7 selects the DMAC I/O pins ( $\overline{\text{DREQ}}$ ,  $\overline{\text{DACK}}$ , and  $\overline{\text{TEND}}$ ).

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	DMAS1A	DMAS1B	DMAS0A	DMAS0B
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	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
3	DMAS1A	0	R/W	DMAC Control Pin Select
2	DMAS1B	0	R/W	Selects the I/O port to control DMAC_1. 00: Specifies pins P14 to P16 as DMAC control pins 01: Specifies pins P33 to P35 as DMAC control pins 10: Setting prohibited 11: Setting prohibited
1	DMAS0A	0	R/W	DMAC Control Pin Select
0	DMAS0B	0	R/W	Selects the I/O port to control DMAC_0. 00: Specifies pins P10 to P12 as DMAC control pins 01: Specifies pins P30 to P32 as DMAC control pins 10: Setting prohibited 11: Setting prohibited

### 11.3.7 Port Function Control Register 9 (PFCR9)

PFCR9 selects the multiple functions for the TPU I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B
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	Bit Name	Initial Value	R/W	Description
7	TPUMS5	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA5 function 0: Specifies pin P26 as output compare output and input capture 1: Specifies P27 as input capture input and P26 as output compare
6	TPUMS4	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA4 function 0: Specifies P25 as output compare output and input capture 1: Specifies P24 as input capture input and P25 as output compare
5	TPUMS3A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA3 function 0: Specifies P21 as output compare output and input capture 1: Specifies P20 as input capture input and P21 as output compare
4	TPUMS3B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC3 function 0: Specifies P22 as output compare output and input capture 1: Specifies P23 as input capture input and P22 as output compare

Bit	Bit Name	Initial Value	R/W	Description
3	TPUMS2	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCA2 function</p> <p>0: Specifies P36 as output compare output and input capture</p> <p>1: Specifies P37 as input capture input and P36 as output compare</p>
2	TPUMS1	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCA1 function</p> <p>0: Specifies P34 as output compare output and input capture</p> <p>1: Specifies P35 as input capture input and P34 as output compare</p>
1	TPUMS0A	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCA0 function</p> <p>0: Specifies P30 as output compare output and input capture</p> <p>1: Specifies P31 as input capture input and P30 as output compare</p>
0	TPUMS0B	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCC0 function</p> <p>0: Specifies P32 as output compare output and input capture</p> <p>1: Specifies P33 as input capture input and P32 as output compare</p>

### 11.3.8 Port Function Control Register B (PFCRB)

PFCRB selects the input pins for  $\overline{\text{IRQ13}}$  to  $\overline{\text{IRQ8}}$ .

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	ITS13	ITS12	ITS11	ITS10	ITS9	ITS8
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	Bit Name	Initial Value	R/W	Description
7 to 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	ITS13	0	R/W	$\overline{\text{IRQ13}}$ Pin Select Selects an input pin for $\overline{\text{IRQ13}}$ . 0: Selects pin P23 as $\overline{\text{IRQ13}}$ -A input 1: Selects pin P63 as $\overline{\text{IRQ13}}$ -B input
4	ITS12	0	R/W	$\overline{\text{IRQ12}}$ Pin Select Selects an input pin for $\overline{\text{IRQ12}}$ . 0: Selects pin P23 as $\overline{\text{IRQ12}}$ -A input 1: Selects pin P63 as $\overline{\text{IRQ12}}$ -B input
3	ITS11	0	R/W	$\overline{\text{IRQ11}}$ Pin Select Selects an input pin for $\overline{\text{IRQ11}}$ . 0: Selects pin P23 as $\overline{\text{IRQ11}}$ -A input 1: Selects pin P63 as $\overline{\text{IRQ11}}$ -B input
2	ITS10	0	R/W	$\overline{\text{IRQ10}}$ Pin Select Selects an input pin for $\overline{\text{IRQ10}}$ . 0: Selects pin P22 as $\overline{\text{IRQ10}}$ -A input 1: Selects pin P62 as $\overline{\text{IRQ10}}$ -B input

Bit	Bit Name	Initial Value	R/W	Description
1	ITS9	0	R/W	$\overline{\text{IRQ9}}$ Pin Select
0	ITS8	0	R/W	$\overline{\text{IRQ8}}$ Pin Select Selects an input pin for $\overline{\text{IRQ8}}$ . 0: Selects pin P20 as $\overline{\text{IRQ8}}$ -A input 1: Selects pin P60 as $\overline{\text{IRQ8}}$ -B input

### 11.3.9 Port Function Control Register C (PFCRC)

PFCRC selects input pins for  $\overline{\text{IRQ7}}$  to  $\overline{\text{IRQ0}}$ .

Bit	7	6	5	4	3	2	1	0
Bit Name	ITS7	ITS6	ITS5	ITS4	ITS3	ITS2	ITS1	ITS0
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	Bit Name	Initial Value	R/W	Description
7	ITS7	0	R/W	$\overline{\text{IRQ7}}$ Pin Select Selects an input pin for $\overline{\text{IRQ7}}$ . 0: Selects pin P17 as $\overline{\text{IRQ7}}$ -A input 1: Selects pin P57 as $\overline{\text{IRQ7}}$ -B output
6	ITS6	0	R/W	$\overline{\text{IRQ6}}$ Pin Select Selects an input pin for $\overline{\text{IRQ6}}$ . 0: Selects pin P16 as $\overline{\text{IRQ6}}$ -A input 1: Selects pin P56 as $\overline{\text{IRQ6}}$ -B output
5	ITS5	0	R/W	$\overline{\text{IRQ5}}$ Pin Select Selects an input pin for $\overline{\text{IRQ5}}$ . 0: Selects pin P15 as $\overline{\text{IRQ5}}$ -A input 1: Selects pin P55 as $\overline{\text{IRQ5}}$ -B output

Bit	Bit Name	Initial Value	R/W	Description
4	ITS4	0	R/W	<p><math>\overline{\text{IRQ4}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ4}}</math>.</p> <p>0: Selects pin P14 as <math>\overline{\text{IRQ4}}</math>-A input</p> <p>1: Selects pin P54 as <math>\overline{\text{IRQ4}}</math>-B output</p>
3	ITS3	0	R/W	<p><math>\overline{\text{IRQ3}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ3}}</math>.</p> <p>0: Selects pin P13 as <math>\overline{\text{IRQ3}}</math>-A input</p> <p>1: Selects pin P53 as <math>\overline{\text{IRQ3}}</math>-B output</p>
2	ITS2	0	R/W	<p><math>\overline{\text{IRQ2}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ2}}</math>.</p> <p>0: Selects pin P12 as <math>\overline{\text{IRQ2}}</math>-A input</p> <p>1: Selects pin P52 as <math>\overline{\text{IRQ2}}</math>-B output</p>
1	ITS1	0	R/W	<p><math>\overline{\text{IRQ1}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ1}}</math>.</p> <p>0: Selects pin P11 as <math>\overline{\text{IRQ1}}</math>-A input</p> <p>1: Selects pin P51 as <math>\overline{\text{IRQ1}}</math>-B output</p>
0	ITS0	0	R/W	<p><math>\overline{\text{IRQ0}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ0}}</math>.</p> <p>0: Selects pin P10 as <math>\overline{\text{IRQ0}}</math>-A input</p> <p>1: Selects pin P50 as <math>\overline{\text{IRQ0}}</math>-B output</p>

## 11.4 Usage Notes

### 11.4.1 Notes on Input Buffer Control Register (ICR) Setting

1. When changing the ICR setting, the LSI may malfunction due to an edge that is internally generated according to the pin states. To change the ICR setting, fix the pin high or disable the input function by setting the peripheral module allocated to the corresponding pin.
2. If an input is enabled by setting ICR while multiple input functions are assigned to the pin, the pin state is reflected in all the inputs. Care must be taken for each module settings for unused input functions.
3. When a pin is used as an output, data to be output from the pin will be latched as the pin state if the input by the ICR setting is enabled. To use the pin as an output, disable the input function for the pin by setting ICR.

### 11.4.2 Notes on Port Function Control Register (PFCR) Settings

1. The port function controller controls the I/O ports. To set the input/output to each pin, select the input/output destination and then enable input/output.
2. When changing the input pin, an edge may be generated if the previous pin level differs from the pin level after the change, causing an unintended malfunction. To change the input pin, follow the procedure below.
  - A. Disable the input function by the setting of the peripheral module corresponding to the pin to be changed.
  - B. Select the input pin by the setting of PFCR.
  - C. Enable the input function by the setting of the peripheral module corresponding to the pin to be changed.
3. If a pin function has both a selection bit that modifies the input/output destination and an enable bit that enables the pin function, first specify the input/output destination by the selection bit and then enable the pin function by the enable bit.



## Section 12 16-Bit Timer Pulse Unit (TPU)

This LSI has an on-chip 16-bit timer pulse unit (TPU) that comprises six 16-bit timer channels.

Table 12.1 lists the 16-bit timer unit functions and figure 12.1 is a block diagram.

### 12.1 Features

- Maximum 16-pulse input/output
- Selection of eight counter input clocks for each channel
- The following operations can be set for each channel:
  - Waveform output at compare match
  - Input capture function
  - Counter clear operation
  - Synchronous operations:
    - Multiple timer counters (TCNT) can be written to simultaneously
    - Simultaneous clearing by compare match and input capture possible
    - Simultaneous input/output for registers possible by counter synchronous operation
    - Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
- Cascaded operation
- Fast access via internal 16-bit bus
- 26 interrupt sources
- Automatic transfer of register data
- Programmable pulse generator (PPG) output trigger can be generated
- Conversion start trigger for the A/D converter and  $\Delta\Sigma$  A/D converter can be generated
- Module stop state specifiable

**Table 12.1 TPU Functions**

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
Count clock	P $\phi$ /1	P $\phi$ /1	P $\phi$ /1	P $\phi$ /1	P $\phi$ /1	P $\phi$ /1
	P $\phi$ /4	P $\phi$ /4	P $\phi$ /4	P $\phi$ /4	P $\phi$ /4	P $\phi$ /4
	P $\phi$ /16	P $\phi$ /16	P $\phi$ /16	P $\phi$ /16	P $\phi$ /16	P $\phi$ /16
	P $\phi$ /64	P $\phi$ /64	P $\phi$ /64	P $\phi$ /64	P $\phi$ /64	P $\phi$ /64
	TCLKA	P $\phi$ /256	P $\phi$ /1024	P $\phi$ /256	P $\phi$ /1024	P $\phi$ /256
	TCLKB	TCLKA	TCLKA	P $\phi$ /1024	TCLKA	TCLKA
	TCLKC	TCLKB	TCLKB	P $\phi$ /4096	TCLKC	TCLKC
TCLKD	TCNT2	TCLKC	TCLKA	TCNT5	TCLKD	
General registers (TGR)	TGRA_0	TGRA_1	TGRA_2	TGRA_3	TGRA_4	TGRA_5
	TGRB_0	TGRB_1	TGRB_2	TGRB_3	TGRB_4	TGRB_5
General registers/ buffer registers	TGRC_0	—	—	TGRC_3	—	—
	TGRD_0			TGRD_3		
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	O	O	O	O	O
	1 output	O	O	O	O	O
	Toggle output	O	O	O	O	O
Input capture function	O	O	O	O	O	O
Synchronous operation	O	O	O	O	O	O
PWM mode	O	O	O	O	O	O
Phase counting mode	—	O	O	—	O	O
Buffer operation	O	—	—	O	—	—

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
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
DMAC activation	TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture	TGRA_5 compare match or input capture
A/D converter trigger $\Delta\Sigma$ A/D converter trigger	TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture	TGRA_5 compare match or input capture
PPG trigger	TGRA_0/ TGRB_0 compare match or input capture	TGRA_1/ TGRB_1 compare match or input capture	TGRA_2/ TGRB_2 compare match or input capture	TGRA_3/ TGRB_3 compare match or input capture	—	—
Interrupt sources	5 sources  Compare match or input capture 0A  Compare match or input capture 0B  Compare match or input capture 0C  Compare match or input capture 0D  Overflow	4 sources  Compare match or input capture 1A  Compare match or input capture 1B  Overflow Underflow	4 sources  Compare match or input capture 2A  Compare match or input capture 2B  Overflow Underflow	5 sources  Compare match or input capture 3A  Compare match or input capture 3B  Compare match or input capture 3C  Compare match or input capture 3D  Overflow	4 sources  Compare match or input capture 4A  Compare match or input capture 4B  Overflow Underflow	4 sources  Compare match or input capture 5A  Compare match or input capture 5B  Overflow Underflow

## [Legend]

○ : Possible

— : Not possible

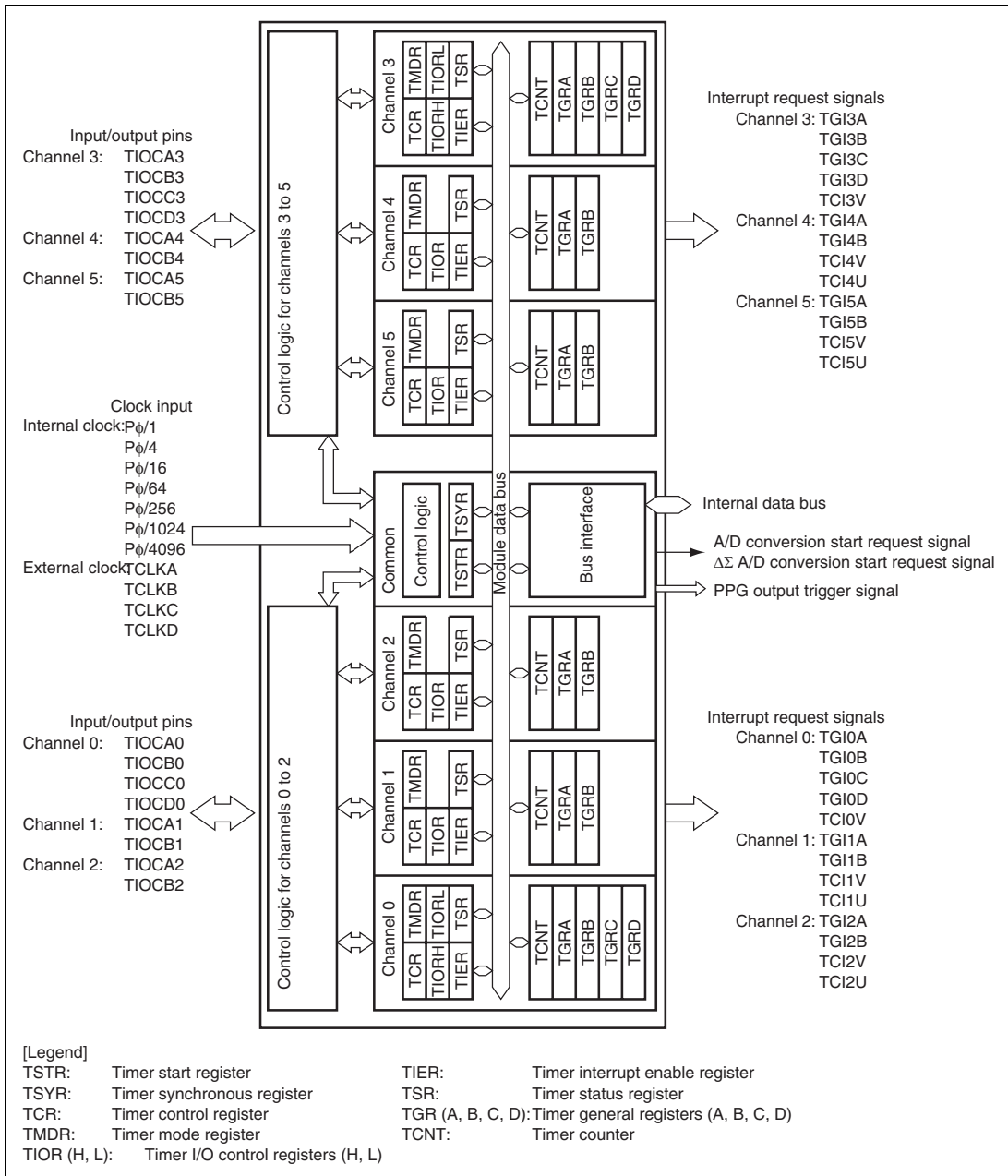


Figure 12.1 Block Diagram of TPU

## 12.2 Input/Output Pins

Table 12.2 shows TPU pin configurations.

**Table 12.2 Pin Configuration**

Channel	Symbol	I/O	Function
All	TCLKA	Input	External clock A input pin (Channel 1 and 5 phase counting mode A phase input)
	TCLKB	Input	External clock B input pin (Channel 1 and 5 phase counting mode B phase input)
	TCLKC	Input	External clock C input pin (Channel 2 and 4 phase counting mode A phase input)
	TCLKD	Input	External clock D input pin (Channel 2 and 4 phase counting mode B phase input)
0	TIOCA0	I/O	TGRA_0 input capture input/output compare output/PWM output pin
	TIOCB0	I/O	TGRB_0 input capture input/output compare output/PWM output pin
	TIOCC0	I/O	TGRC_0 input capture input/output compare output/PWM output pin
	TIOCD0	I/O	TGRD_0 input capture input/output compare output/PWM output pin
1	TIOCA1	I/O	TGRA_1 input capture input/output compare output/PWM output pin
	TIOCB1	I/O	TGRB_1 input capture input/output compare output/PWM output pin
2	TIOCA2	I/O	TGRA_2 input capture input/output compare output/PWM output pin
	TIOCB2	I/O	TGRB_2 input capture input/output compare output/PWM output pin
3	TIOCA3	I/O	TGRA_3 input capture input/output compare output/PWM output pin
	TIOCB3	I/O	TGRB_3 input capture input/output compare output/PWM output pin
	TIOCC3	I/O	TGRC_3 input capture input/output compare output/PWM output pin
	TIOCD3	I/O	TGRD_3 input capture input/output compare output/PWM output pin
4	TIOCA4	I/O	TGRA_4 input capture input/output compare output/PWM output pin
	TIOCB4	I/O	TGRB_4 input capture input/output compare output/PWM output pin
5	TIOCA5	I/O	TGRA_5 input capture input/output compare output/PWM output pin
	TIOCB5	I/O	TGRB_5 input capture input/output compare output/PWM output pin

## 12.3 Register Descriptions

The TPU has the following registers in each channel.

### Channel 0:

- Timer control register\_0 (TCR\_0)
- Timer mode register\_0 (TMDR\_0)
- Timer I/O control register H\_0 (TIORH\_0)
- Timer I/O control register L\_0 (TIORL\_0)
- Timer interrupt enable register\_0 (TIER\_0)
- Timer status register\_0 (TSR\_0)
- Timer counter\_0 (TCNT\_0)
- Timer general register A\_0 (TGRA\_0)
- Timer general register B\_0 (TGRB\_0)
- Timer general register C\_0 (TGRC\_0)
- Timer general register D\_0 (TGRD\_0)

### Channel 1:

- Timer control register\_1 (TCR\_1)
- Timer mode register\_1 (TMDR\_1)
- Timer I/O control register\_1 (TIOR\_1)
- Timer interrupt enable register\_1 (TIER\_1)
- Timer status register\_1 (TSR\_1)
- Timer counter\_1 (TCNT\_1)
- Timer general register A\_1 (TGRA\_1)
- Timer general register B\_1 (TGRB\_1)

**Channel 2:**

- Timer control register\_2 (TCR\_2)
- Timer mode register\_2 (TMDR\_2)
- Timer I/O control register\_2 (TIOR\_2)
- Timer interrupt enable register\_2 (TIER\_2)
- Timer status register\_2 (TSR\_2)
- Timer counter\_2 (TCNT\_2)
- Timer general register A\_2 (TGRA\_2)
- Timer general register B\_2 (TGRB\_2)

**Channel 3:**

- Timer control register\_3 (TCR\_3)
- Timer mode register\_3 (TMDR\_3)
- Timer I/O control register H\_3 (TIORH\_3)
- Timer I/O control register L\_3 (TIORL\_3)
- Timer interrupt enable register\_3 (TIER\_3)
- Timer status register\_3 (TSR\_3)
- Timer counter\_3 (TCNT\_3)
- Timer general register A\_3 (TGRA\_3)
- Timer general register B\_3 (TGRB\_3)
- Timer general register C\_3 (TGRC\_3)
- Timer general register D\_3 (TGRD\_3)

**Channel 4:**

- Timer control register\_4 (TCR\_4)
- Timer mode register\_4 (TMDR\_4)
- Timer I/O control register\_4 (TIOR\_4)
- Timer interrupt enable register\_4 (TIER\_4)
- Timer status register\_4 (TSR\_4)
- Timer counter\_4 (TCNT\_4)
- Timer general register A\_4 (TGRA\_4)
- Timer general register B\_4 (TGRB\_4)

### **Channel 5:**

- Timer control register\_5 (TCR\_5)
- Timer mode register\_5 (TMDR\_5)
- Timer I/O control register\_5 (TIOR\_5)
- Timer interrupt enable register\_5 (TIER\_5)
- Timer status register\_5 (TSR\_5)
- Timer counter\_5 (TCNT\_5)
- Timer general register A\_5 (TGRA\_5)
- Timer general register B\_5 (TGRB\_5)

### **Common Registers:**

- Timer start register (TSTR)
- Timer synchronous register (TSYR)



### 12.3.1 Timer Control Register (TCR)

TCR controls the TCNT operation for each channel. The TPU has a total of six TCR registers, one for each channel. TCR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
7	CCLR2	0	R/W	Counter Clear 2 to 0
6	CCLR1	0	R/W	These bits select the TCNT counter clearing source. See tables 12.3 and 12.4 for details.
5	CCLR0	0	R/W	
4	CKEG1	0	R/W	Clock Edge 1 and 0
3	CKEG0	0	R/W	These bits select the input clock edge. For details, see table 12.5. When the input clock is counted using both edges, the input clock period is halved (e.g. $P\phi/4$ both edges = $P\phi/2$ 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. Internal clock edge selection is valid when the input clock is $P\phi/4$ or slower. This setting is ignored if the input clock is $P\phi/1$ , or when overflow/underflow of another channel is selected.
2	TPSC2	0	R/W	
1	TPSC1	0	R/W	These bits select the TCNT counter clock. The clock source can be selected independently for each channel. See tables 12.6 to 12.11 for details. To select the external clock as the clock source, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 11, I/O Ports.
0	TPSC0	0	R/W	

**Table 12.3 CCLR2 to CCLR0 (Channels 0 and 3)**

Channel	Bit 7 CCLR2	Bit 6 CCLR1	Bit 5 CCLR0	Description
0, 3	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* <sup>1</sup>
	1	0	0	TCNT clearing disabled
	1	0	1	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	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* <sup>1</sup>

- Notes: 1. Synchronous operation is selected 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.

**Table 12.4 CCLR2 to CCLR0 (Channels 1, 2, 4, and 5)**

Channel	Bit 7 Reserved* <sup>2</sup>	Bit 6 CCLR1	Bit 5 CCLR0	Description
1, 2, 4, 5	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/synchronous operation* <sup>1</sup>

- Notes: 1. Synchronous operation is selected by setting the SYNC bit in TSYR to 1.  
 2. Bit 7 is reserved in channels 1, 2, 4, and 5. It is always read as 0 and cannot be modified.

**Table 12.5 Input Clock Edge Selection**

Clock Edge Selection		Input Clock	
CKEG1	CKEG0	Internal Clock	External Clock
0	0	Counted at falling edge	Counted at rising edge
0	1	Counted at rising edge	Counted at falling edge
1	X	Counted at both edges	Counted at both edges

[Legend]

X: Don't care

**Table 12.6 TPSC2 to TPSC0 (Channel 0)**

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
0	0	0	0	Internal clock: counts on P $\phi$ /1
	0	0	1	Internal clock: counts on P $\phi$ /4
	0	1	0	Internal clock: counts on P $\phi$ /16
	0	1	1	Internal clock: counts on P $\phi$ /64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	External clock: counts on TCLKD pin input

**Table 12.7 TPSC2 to TPSC0 (Channel 1)**

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
1	0	0	0	Internal clock: counts on P $\phi$ /1
	0	0	1	Internal clock: counts on P $\phi$ /4
	0	1	0	Internal clock: counts on P $\phi$ /16
	0	1	1	Internal clock: counts on P $\phi$ /64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	Internal clock: counts on P $\phi$ /256
	1	1	1	Counts on TCNT2 overflow/underflow

Note: This setting is ignored when channel 1 is in phase counting mode.

**Table 12.8 TPSC2 to TPSC0 (Channel 2)**

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
2	0	0	0	Internal clock: counts on P $\phi$ /1
	0	0	1	Internal clock: counts on P $\phi$ /4
	0	1	0	Internal clock: counts on P $\phi$ /16
	0	1	1	Internal clock: counts on P $\phi$ /64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	Internal clock: counts on P $\phi$ /1024

Note: This setting is ignored when channel 2 is in phase counting mode.

**Table 12.9 TPSC2 to TPSC0 (Channel 3)**

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
3	0	0	0	Internal clock: counts on P $\phi$ /1
	0	0	1	Internal clock: counts on P $\phi$ /4
	0	1	0	Internal clock: counts on P $\phi$ /16
	0	1	1	Internal clock: counts on P $\phi$ /64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	Internal clock: counts on P $\phi$ /1024
	1	1	0	Internal clock: counts on P $\phi$ /256
	1	1	1	Internal clock: counts on P $\phi$ /4096

**Table 12.11 TPSC2 to TPSC0 (Channel 4)**

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
4	0	0	0	Internal clock: counts on P $\phi$ /1
	0	0	1	Internal clock: counts on P $\phi$ /4
	0	1	0	Internal clock: counts on P $\phi$ /16
	0	1	1	Internal clock: counts on P $\phi$ /64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on P $\phi$ /1024
	1	1	1	Counts on TCNT5 overflow/underflow

Note: This setting is ignored when channel 4 is in phase counting mode.

**Table 12.11 TPSC2 to TPSC0 (Channel 5)**

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
5	0	0	0	Internal clock: counts on P $\phi$ /1
	0	0	1	Internal clock: counts on P $\phi$ /4
	0	1	0	Internal clock: counts on P $\phi$ /16
	0	1	1	Internal clock: counts on P $\phi$ /64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on P $\phi$ /256
	1	1	1	External clock: counts on TCLKD pin input

Note: This setting is ignored when channel 5 is in phase counting mode.

### 12.3.2 Timer Mode Register (TMDR)

TMDR sets the operating mode for each channel. The TPU has six TMDR registers, one for each channel. TMDR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	BFB	BFA	MD3	MD2	MD1	MD0
Initial Value	1	1	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	R	Reserved These are read-only bits and cannot be modified.
5	BFB	0	R/W	Buffer Operation B Specifies whether TGRB is to normally operate, 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. 0: TGRB operates normally 1: TGRB and TGRD used together for buffer operation
4	BFA	0	R/W	Buffer Operation A Specifies whether TGRA is to normally operate, 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. 0: TGRA operates normally 1: TGRA and TGRC used together for buffer operation
3	MD3	0	R/W	Modes 3 to 0
2	MD2	0	R/W	Set the timer operating mode.
1	MD1	0	R/W	MD3 is a reserved bit. The write value should always be 0. See table 12.12 for details.
0	MD0	0	R/W	

**Table 12.12 MD3 to MD0**

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
0	0	0	1	Reserved
0	0	1	0	PWM mode 1
0	0	1	1	PWM mode 2
0	1	0	0	Phase counting mode 1
0	1	0	1	Phase counting mode 2
0	1	1	0	Phase counting mode 3
0	1	1	1	Phase counting mode 4
1	X	X	X	—

[Legend]

X: Don't care

- Notes:
1. MD3 is a reserved bit. The write value should always be 0.
  2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

### 12.3.3 Timer I/O Control Register (TIOR)

TIOR controls TGR. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. 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.

When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

To designate the input capture pin in TIOR, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 11, I/O Ports.

- TIORH\_0, TIOR\_1, TIOR\_2, TIORH\_3, TIOR\_4, TIOR\_5

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- TIORL\_0, TORL\_3

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- TIORH\_0, TIOR\_1, TIOR\_2, TIORH\_3, TIOR\_4, TIOR\_5

Bit	Bit Name	Initial Value	R/W	Description
7	IOB3	0	R/W	I/O Control B3 to B0
6	IOB2	0	R/W	Specify the function of TGRB.
5	IOB1	0	R/W	For details, see tables 12.13, 12.15, 12.16, 12.17, 12.19, and 12.20.
4	IOB0	0	R/W	
3	IOA3	0	R/W	I/O Control A3 to A0
2	IOA2	0	R/W	Specify the function of TGRA.
1	IOA1	0	R/W	For details, see tables 12.21, 12.23, 12.24, 12.25, 12.27, and 12.28.
0	IOA0	0	R/W	

- TIORL\_0, TIORL\_3

Bit	Bit Name	Initial Value	R/W	Description
7	IOD3	0	R/W	I/O Control D3 to D0
6	IOD2	0	R/W	Specify the function of TGRD.
5	IOD1	0	R/W	For details, see tables 12.14 and 12.18.
4	IOD0	0	R/W	
3	IOC3	0	R/W	I/O Control C3 to C0
2	IOC2	0	R/W	Specify the function of TGRC.
1	IOC1	0	R/W	For details, see tables 12.22 and 12.26.
0	IOC0	0	R/W	



Table 12.13 TIORH\_0

				Description		
Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	TGRB_0 Function	TIOCB0 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCB0 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCB0 pin Input capture at falling edge
1	0	1	x	Capture input source is TIOCB0 pin Input capture at both edges		
1	1	x	x	Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down*		

[Legend]

X: Don't care

Note: \* When bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and P $\phi$ /1 is used as the TCNT\_1 count clock, this setting is invalid and input capture is not generated.

Table 12.14 TIORL\_0

Bit 7 IOD3	Bit 6 IOD2	Bit 5 IOD1	Bit 4 IOD0	Description		
				TGRD_0 Function	TIOCD0 Pin Function	
0	0	0	0	Output compare register* <sup>2</sup>	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register* <sup>2</sup>	Capture input source is TIOCD0 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCD0 pin Input capture at falling edge
1	0	1	X	Capture input source is TIOCD0 pin Input capture at both edges		
1	1	X	X	Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down* <sup>1</sup>		

[Legend]

X: Don't care

- Notes: 1. When bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and P<sub>φ</sub>/1 is used as the TCNT\_1 count clock, this setting is invalid and input capture is not generated.
2. When the BFB bit in TMDR\_0 is set to 1 and TGRD\_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 12.15 TIOR\_1

				Description		
Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	TGRB_1 Function	TIOCB1 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCB1 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCB1 pin Input capture at falling edge
1	0	1	X	Capture input source is TIOCB1 pin Input capture at both edges		
1	1	X	X	TGRC_0 compare match/input capture Input capture at generation of TGRC_0 compare match/input capture		

[Legend]

X: Don't care

Table 12.16 TIOR\_2

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	TGRB_2 Function	Description	
					TIOCB2 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	X	0	0		Input capture register	Capture input source is TIOCB2 pin Input capture at rising edge
1	X	0	1			Capture input source is TIOCB2 pin Input capture at falling edge
1	X	1	X	Capture input source is TIOCB2 pin Input capture at both edges		

[Legend]

X: Don't care

Table 12.17 TIORH\_3

				Description		
Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	TGRB_3 Function	TIOCB3 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCB3 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCB3 pin Input capture at falling edge
1	0	1	x	Capture input source is TIOCB3 pin Input capture at both edges		
1	1	x	x	Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down*		

[Legend]

X: Don't care

Note: When bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and P $\phi$ /1 is used as the TCNT\_4 count clock, this setting is invalid and input capture is not generated.

Table 12.18 TIORL\_3

				Description		
Bit 7 IOD3	Bit 6 IOD2	Bit 5 IOD1	Bit 4 IOD0	TGRD_3 Function	TIOCD3 Pin Function	
0	0	0	0	Output compare register* <sup>2</sup>	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register* <sup>2</sup>	Capture input source is TIOCD3 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCD3 pin Input capture at falling edge
1	0	1	x	Capture input source is TIOCD3 pin Input capture at both edges		
1	1	x	x	Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down* <sup>1</sup>		

[Legend]

X: Don't care

- Notes: 1. When bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and P<sub>φ</sub>/1 is used as the TCNT\_4 count clock, this setting is invalid and input capture is not generated.
2. When the BFB bit in TMDR\_3 is set to 1 and TGRD\_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 12.19 TIOR\_4

				Description		
Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	TGRB_4 Function	TIOCB4 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCB4 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCB4 pin Input capture at falling edge
1	0	1	X	Capture input source is TIOCB4 pin Input capture at both edges		
1	1	X	X	Capture input source is TGRC_3 compare match/input capture Input capture at generation of TGRC_3 compare match/input capture		

[Legend]

X: Don't care

Table 12.20 TIOR\_5

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description		
				TGRB_5 Function	TIOCB5 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	X	0	0		Input capture register	Capture input source is TIOCB5 pin Input capture at rising edge
1	X	0	1			Capture input source is TIOCB5 pin Input capture at falling edge
1	X	1	X	Capture input source is TIOCB5 pin Input capture at both edges		

[Legend]

X: Don't care



Table 12.21 TIORH\_0

				Description		
Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	TGRA_0 Function	TIOCA0 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCA0 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCA0 pin Input capture at falling edge
1	0	1	X	Capture input source is TIOCA0 pin Input capture at both edges		
1	1	X	X	Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down*		

[Legend]

X: Don't care

Note: \* When bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and P $\phi$ /1 is used as the TCNT\_1 count clock, this setting is invalid and input capture is not generated.

Table 12.22 TIORL\_0

Bit 3 IOC3	Bit 2 IOC2	Bit 1 IOC1	Bit 0 IOC0	Description		
				TGRC_0 Function	TIOCC0 Pin Function	
0	0	0	0	Output compare register* <sup>2</sup>	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register* <sup>2</sup>	Capture input source is TIOCC0 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCC0 pin Input capture at falling edge
1	0	1	X			Capture input source is TIOCC0 pin Input capture at both edges
1	1	X	X			Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down* <sup>1</sup>

[Legend]

X: Don't care

- Notes: 1. When bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and P<sub>φ</sub>/1 is used as the TCNT\_1 count clock, this setting is invalid and input capture is not generated.
2. When the BFA bit in TMDR\_0 is set to 1 and TGRC\_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 12.23 TIOR\_1

				Description	
Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	TGRA_1 Function	TIOCA1 Pin Function
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA1 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA1 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA1 pin Input capture at both edges
1	1	X	X		Capture input source is TGRA_0 compare match/input capture Input capture at generation of channel 0/TGRA_0 compare match/input capture

[Legend]

X: Don't care

Table 12.24 TIOR\_2

Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	TGRA_2 Function	Description	
					TIOCA2 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	X	0	0		Input capture register	Capture input source is TIOCA2 pin Input capture at rising edge
1	X	0	1			Capture input source is TIOCA2 pin Input capture at falling edge
1	X	1	X	Capture input source is TIOCA2 pin Input capture at both edges		

[Legend]

X: Don't care

Table 12.25 TIORH\_3

				Description		
Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	TGRA_3 Function	TIOCA3 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCA3 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCA3 pin Input capture at falling edge
1	0	1	X	Capture input source is TIOCA3 pin Input capture at both edges		
1	1	X	X	Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down*		

[Legend]

X: Don't care

Note: \* When bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and P $\phi$ /1 is used as the TCNT\_4 count clock, this setting is invalid and input capture is not generated.

Table 12.26 TIORL\_3

				Description	
Bit 3 IOC3	Bit 2 IOC2	Bit 1 IOC1	Bit 0 IOC0	TGRC_3 Function	TIOCC3 Pin Function
0	0	0	0	Output compare register* <sup>2</sup>	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register* <sup>2</sup>	Capture input source is TIOCC3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCC3 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCC3 pin Input capture at both edges
1	1	X	X		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down* <sup>1</sup>

## [Legend]

X: Don't care

- Note:
1. When bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and P<sub>φ</sub>/1 is used as the TCNT\_4 count clock, this setting is invalid and input capture is not generated.
  2. When the BFA bit in TMDR\_3 is set to 1 and TGRC\_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 12.27 TIOR\_4

Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	Description		
				TGRA_4 Function	TIOCA4 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	0	0	0		Input capture register	Capture input source is TIOCA4 pin Input capture at rising edge
1	0	0	1			Capture input source is TIOCA4 pin Input capture at falling edge
1	0	1	X	Capture input source is TIOCA4 pin Input capture at both edges		
1	1	X	X	Capture input source is TGRA_3 compare match/input capture Input capture at generation of TGRA_3 compare match/input capture		

[Legend]

X: Don't care

Table 12.28 TIOR\_5

Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	TGRA_5 Function	Description	
					TIOCA5 Pin Function	
0	0	0	0	Output compare register	Output disabled	
0	0	0	1		Initial output is 0 output 0 output at compare match	
0	0	1	0		Initial output is 0 output 1 output at compare match	
0	0	1	1		Initial output is 0 output Toggle output at compare match	
0	1	0	0		Output disabled	
0	1	0	1		Initial output is 1 output 0 output at compare match	
0	1	1	0		Initial output is 1 output 1 output at compare match	
0	1	1	1		Initial output is 1 output Toggle output at compare match	
1	X	0	0		Input capture register	Input capture source is TIOCA5 pin Input capture at rising edge
1	X	0	1			Input capture source is TIOCA5 pin Input capture at falling edge
1	X	1	X	Input capture source is TIOCA5 pin Input capture at both edges		

[Legend]

X: Don't care



### 12.3.4 Timer Interrupt Enable Register (TIER)

TIER controls enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TTGE	—	TCIEU	TCIEV	TGIED	TCIEC	TGIEB	TGIEA
Initial Value	0	1	0	0	0	0	0	0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
7	TTGE	0	R/W	<p>A/D Conversion Start Request Enable</p> <p>Enables/disables generation of A/D conversion and <math>\Delta\Sigma</math> A/D conversion start requests by TGRA input capture/compare match.</p> <p>0: A/D conversion and <math>\Delta\Sigma</math> A/D conversion start request generation disabled</p> <p>1: A/D conversion and <math>\Delta\Sigma</math> A/D conversion start request generation enabled</p>
6	—	1	R	<p>Reserved</p> <p>This is a read-only bit and cannot be modified.</p>
5	TCIEU	0	R/W	<p>Underflow Interrupt Enable</p> <p>Enables/disables interrupt requests (TCIU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1, 2, 4, and 5.</p> <p>In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.</p> <p>0: Interrupt requests (TCIU) by TCFU disabled</p> <p>1: Interrupt requests (TCIU) by TCFU enabled</p>
4	TCIEV	0	R/W	<p>Overflow Interrupt Enable</p> <p>Enables/disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1.</p> <p>0: Interrupt requests (TCIV) by TCFV disabled</p> <p>1: Interrupt requests (TCIV) by TCFV enabled</p>

Bit	Bit Name	Initial value	R/W	Description
3	TGIED	0	R/W	<p>TGR Interrupt Enable D</p> <p>Enables/disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3.</p> <p>In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.</p> <p>0: Interrupt requests (TGID) by TGFD bit disabled 1: Interrupt requests (TGID) by TGFD bit enabled</p>
2	TGIEC	0	R/W	<p>TGR Interrupt Enable C</p> <p>Enables/disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3.</p> <p>In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.</p> <p>0: Interrupt requests (TGIC) by TGFC bit disabled 1: Interrupt requests (TGIC) by TGFC bit enabled</p>
1	TGIEB	0	R/W	<p>TGR Interrupt Enable B</p> <p>Enables/disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1.</p> <p>0: Interrupt requests (TGIB) by TGFB bit disabled 1: Interrupt requests (TGIB) by TGFB bit enabled</p>
0	TGIEA	0	R/W	<p>TGR Interrupt Enable A</p> <p>Enables/disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1.</p> <p>0: Interrupt requests (TGIA) by TGFA bit disabled 1: Interrupt requests (TGIA) by TGFA bit enabled</p>

### 12.3.5 Timer Status Register (TSR)

TSR indicates the status of each channel. The TPU has six TSR registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TCFD	—	TCFU	TCFV	TGFD	TGFC	TGFB	TGFA
Initial Value	1	1	0	0	0	0	0	0
R/W	R	R	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: \* Only 0 can be written to bits 5 to 0, to clear flags.

Bit	Bit Name	Initial value	R/W	Description
7	TCFD	1	R	Count Direction Flag 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. 0: TCNT counts down 1: TCNT counts up
6	—	1	R	Reserved This is a read-only bit and cannot be modified.
5	TCFU	0	R/(W)*	Underflow Flag Status flag that indicates that a 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. [Setting condition] When the TCNT value underflows (changes from H'0000 to H'FFFF) [Clearing condition] When a 0 is written to TCFU after reading TCFU = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
4	TCFV	0	R/(W)*	<p>Overflow Flag</p> <p>Status flag that indicates that a TCNT overflow has occurred.</p> <p>[Setting condition]</p> <p>When the TCNT value overflows (changes from H'FFFF to H'0000)</p> <p>[Clearing condition]</p> <p>When a 0 is written to TCFV after reading TCFV = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
3	TGFD	0	R/(W)*	<p>Input Capture/Output Compare Flag D</p> <p>Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3.</p> <p>In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.</p> <p>[Setting conditions]</p> <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> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When DTC is activated by a TGID interrupt while the DISEL bit in MRB of DTC is 0</li> <li>• When 0 is written to TGFD after reading TGFD = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>

Bit	Bit Name	Initial value	R/W	Description
2	TGFC	0	R/(W)*	<p>Input Capture/Output Compare Flag C</p> <p>Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3.</p> <p>In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.</p> <p>[Setting conditions]</p> <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> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When DTC is activated by a TGIC interrupt while the DISEL bit in MRB of DTC is 0</li> <li>• When 0 is written to TGFC after reading TGFC = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>
1	TGFB	0	R/(W)*	<p>Input Capture/Output Compare Flag B</p> <p>Status flag that indicates the occurrence of TGRB input capture or compare match.</p> <p>[Setting conditions]</p> <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> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When DTC is activated by a TGIB interrupt while the DISEL bit in MRB of DTC is 0</li> <li>• When 0 is written to TGFB after reading TGFB = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>

Bit	Bit Name	Initial value	R/W	Description
0	TGFA	0	R/(W)*	<p>Input Capture/Output Compare Flag A</p> <p>Status flag that indicates the occurrence of TGRA input capture or compare match.</p> <p>[Setting conditions]</p> <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> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When DTC is activated by a TGIA interrupt while the DISEL bit in MRB of DTC is 0</li> <li>• When DMAC is activated by a TGIA interrupt while the DTA bit in DMDR of DMAC is 1</li> <li>• When 0 is written to TGFA after reading TGFA = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>

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

### 12.3.6 Timer Counter (TCNT)

TCNT is a 16-bit readable/writable counter. The TPU has six TCNT counters, one for each channel.

TCNT is initialized to H'0000 by a reset or in hardware standby mode.

TCNT cannot be accessed in 8-bit units. TCNT must always be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8
Bit Name								
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	7	6	5	4	3	2	1	0
Bit Name								
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

### 12.3.7 Timer General Register (TGR)

TGR is a 16-bit readable/writable register 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 cannot be accessed in 8-bit units; they must always be accessed in 16-bit units. TGR and buffer register combinations during buffer operations are TGRA–TGRC and TGRB–TGRD.

Bit	15	14	13	12	11	10	9	8
Bit Name								
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
Bit	7	6	5	4	3	2	1	0
Bit Name								
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

### 12.3.8 Timer Start Register (TSTR)

TSTR starts or stops operation for channels 0 to 5. When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	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	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	CST5	0	R/W	Counter Start 5 to 0
4	CST4	0	R/W	These bits select operation or stoppage for TCNT.
3	CST3	0	R/W	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.
2	CST2	0	R/W	
1	CST1	0	R/W	
0	CST0	0	R/W	
				0: TCNT_5 to TCNT_0 count operation is stopped 1: TCNT_5 to TCNT_0 performs count operation



### 12.3.9 Timer Synchronous Register (TSYR)

TSYR selects independent operation or synchronous operation for the TCNT counters of channels 0 to 5. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	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	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	SYNC5	0	R/W	Timer Synchronization 5 to 0
4	SYNC4	0	R/W	These bits select whether operation is independent of or synchronized with other channels.
3	SYNC3	0	R/W	When synchronous operation is selected, synchronous presetting of multiple channels, and synchronous clearing through counter clearing on another channel are possible.
2	SYNC2	0	R/W	
1	SYNC1	0	R/W	
0	SYNC0	0	R/W	To set synchronous operation, the SYNC bits for at least two channels must be set to 1. 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. 0: TCNT_5 to TCNT_0 operate independently (TCNT presetting/clearing is unrelated to other channels) 1: TCNT_5 to TCNT_0 perform synchronous operation (TCNT synchronous presetting/synchronous clearing is possible)

## 12.4 Operation

### 12.4.1 Basic Functions

Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, periodic counting, and external event counting.

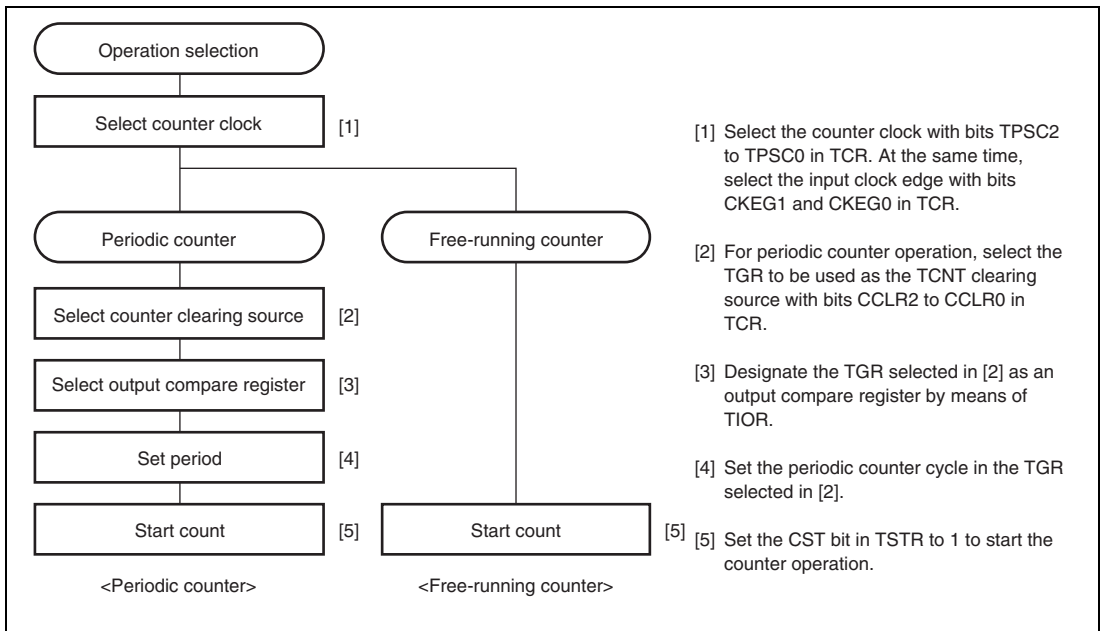
Each TGR can be used as an input capture register or output compare register.

#### (1) 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.

##### (a) Example of count operation setting procedure

Figure 12.2 shows an example of the count operation setting procedure.

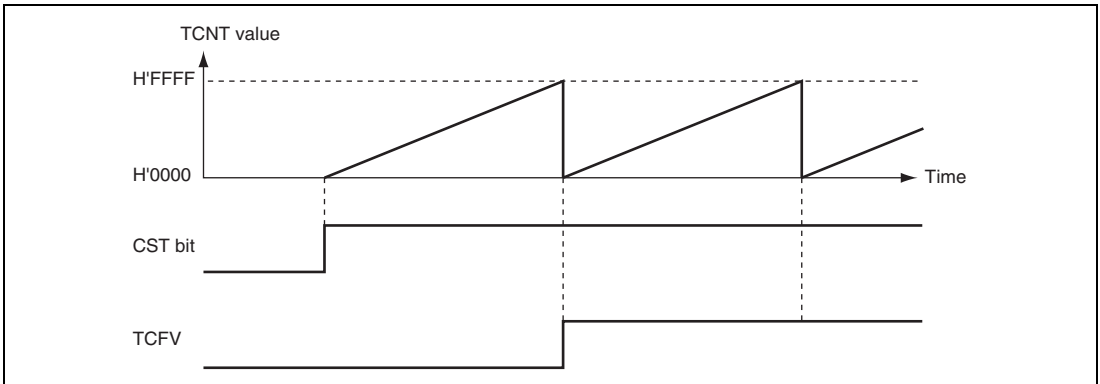


**Figure 12.2 Example of Counter Operation Setting Procedure**

### (b) 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 (changes 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 12.3 illustrates free-running counter operation.

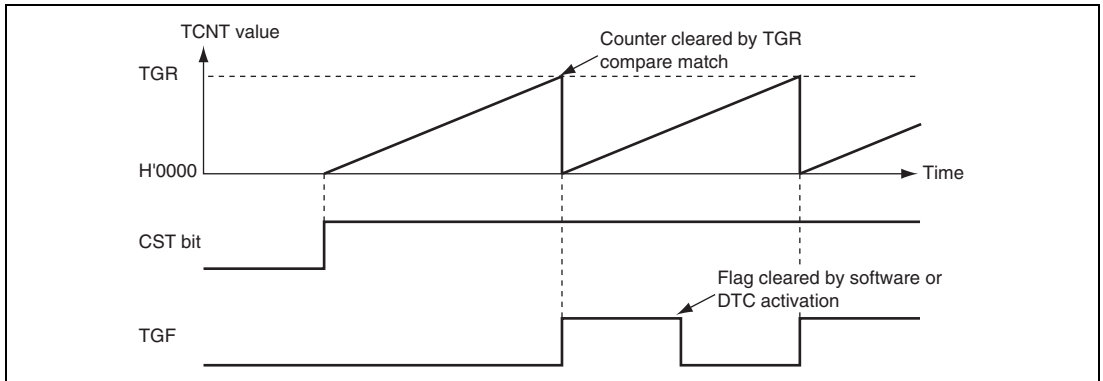


**Figure 12.3 Free-Running Counter Operation**

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 count-up 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 12.4 illustrates periodic counter operation.



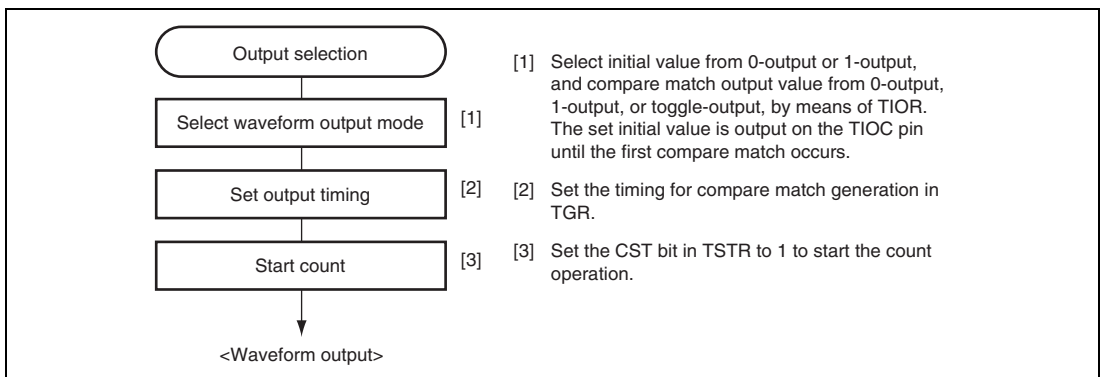
**Figure 12.4 Periodic Counter Operation**

## (2) Waveform Output by Compare Match

The TPU can perform 0, 1, or toggle output from the corresponding output pin using a compare match.

### (a) Example of setting procedure for waveform output by compare match

Figure 12.5 shows an example of the setting procedure for waveform output by a compare match.

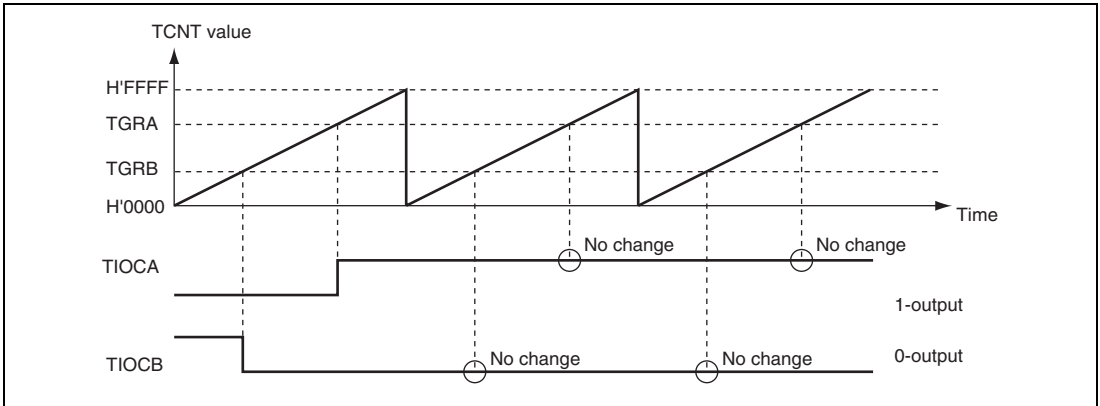


**Figure 12.5 Example of Setting Procedure for Waveform Output by Compare Match**

**(b) Examples of waveform output operation**

Figure 12.6 shows an example of 0-output and 1-output.

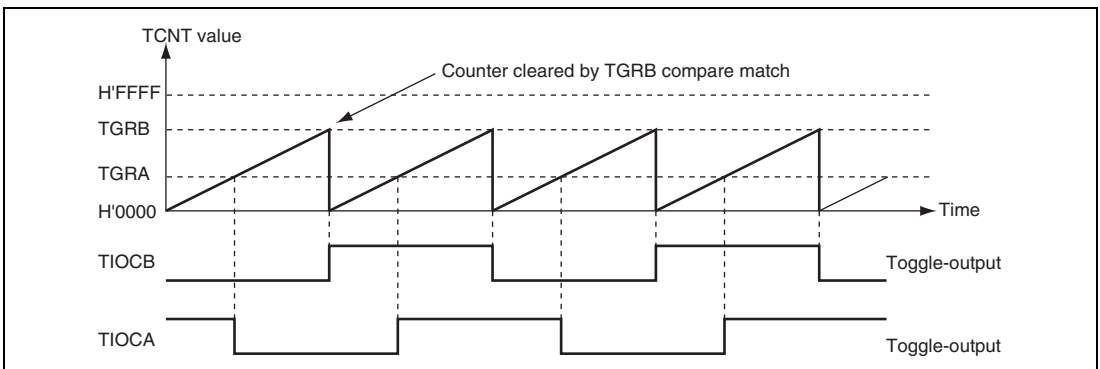
In this example, TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level match, the pin level does not change.



**Figure 12.6 Example of 0-Output/1-Output Operation**

Figure 12.7 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 12.7 Example of Toggle Output Operation**

### (3) Input Capture Function

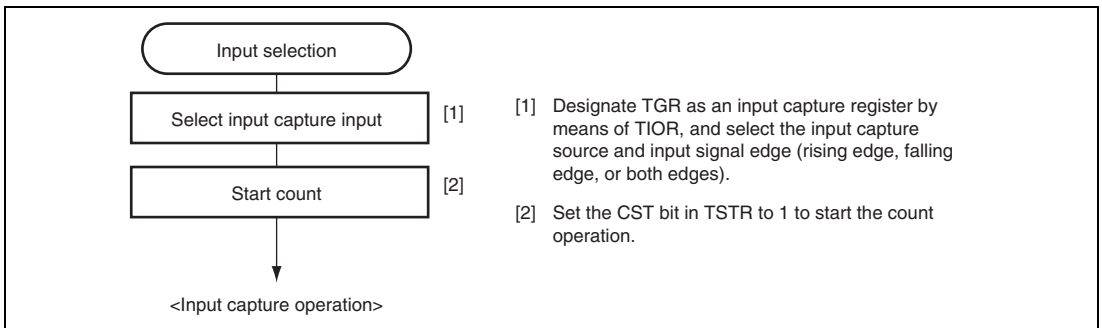
The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detection 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, P $\phi$ /1 should not be selected as the counter input clock used for input capture input. Input capture will not be generated if P $\phi$ /1 is selected.

#### (a) Example of setting procedure for input capture operation

Figure 12.8 shows an example of the setting procedure for input capture operation.

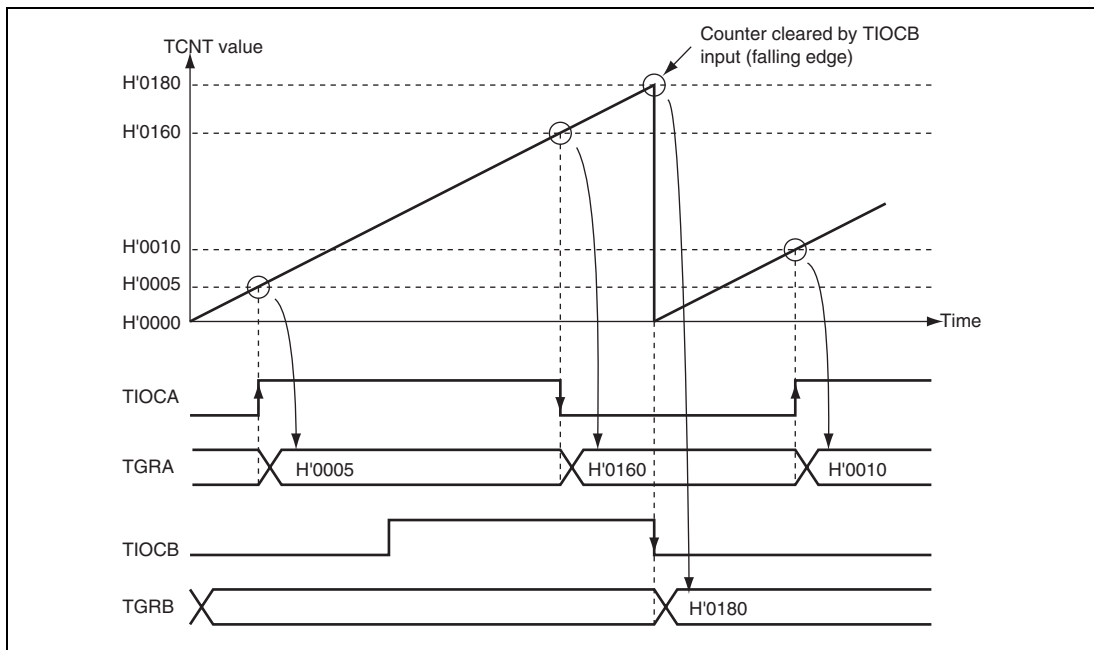


**Figure 12.8 Example of Setting Procedure for Input Capture Operation**

**(b) Example of input capture operation**

Figure 12.9 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 12.9 Example of Input Capture Operation**

## 12.4.2 Synchronous Operation

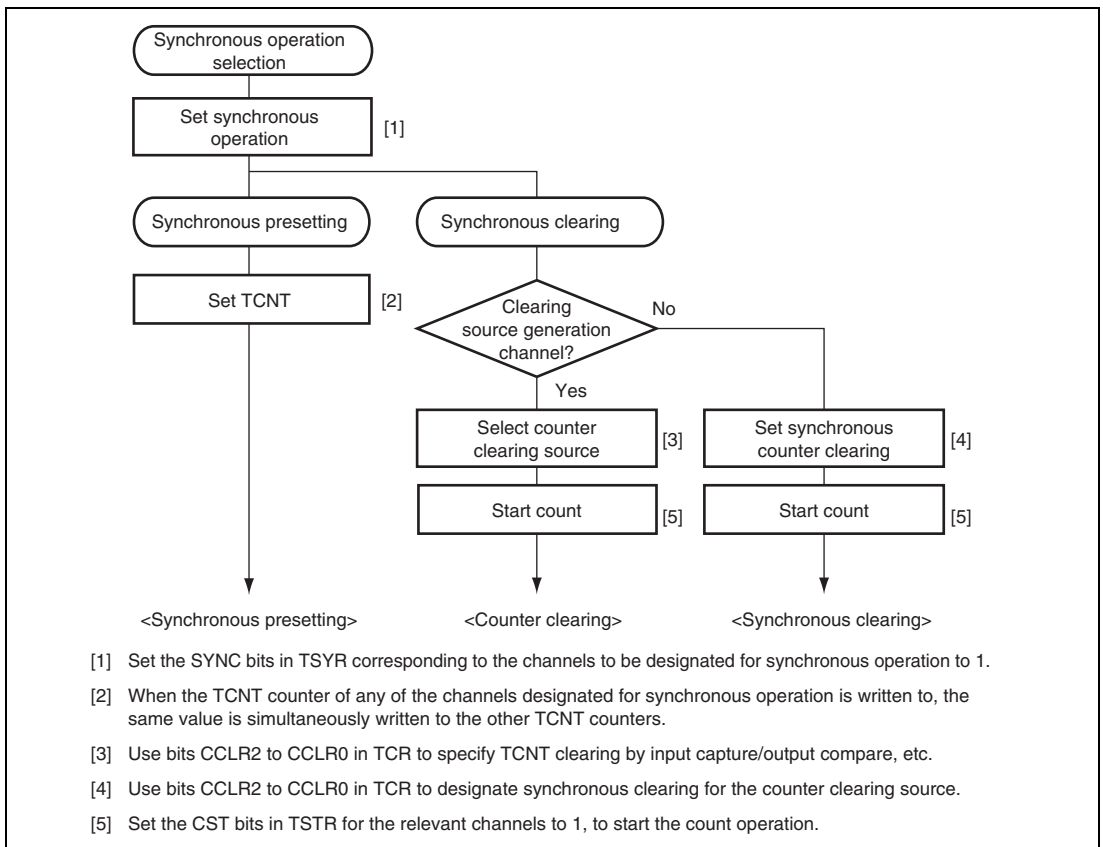
In synchronous operation, the values in multiple TCNT counters can be rewritten simultaneously (synchronous presetting). Also, multiple TCNT counters can be cleared simultaneously (synchronous clearing) by making the appropriate setting in TCR.

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.

### (1) Example of Synchronous Operation Setting Procedure

Figure 12.10 shows an example of the synchronous operation setting procedure.



**Figure 12.10 Example of Synchronous Operation Setting Procedure**



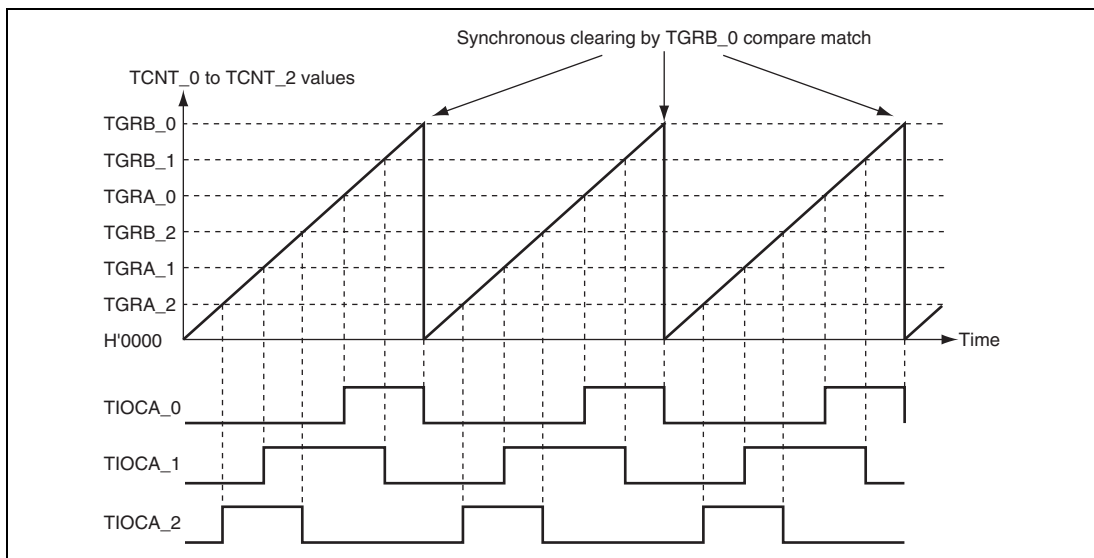
## (2) Example of Synchronous Operation

Figure 12.11 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGRB\_0 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 TIOCA0, TIOCA1, and TIOCA2. At this time, synchronous presetting and synchronous clearing by TGRB\_0 compare match are performed for channel 0 to 2 TCNT counters, and the data set in TGRB\_0 is used as the PWM cycle.

For details on PWM modes, see section 12.4.5, PWM Modes.



**Figure 12.11 Example of Synchronous Operation**

### 12.4.3 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 a compare match register.

Table 12.29 shows the register combinations used in buffer operation.

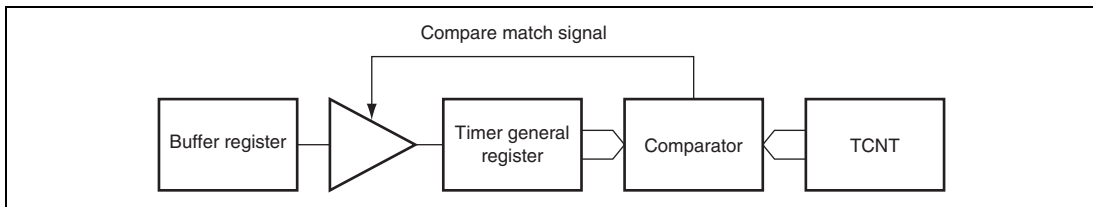
**Table 12.29 Register Combinations in Buffer Operation**

Channel	Timer General Register	Buffer Register
0	TGRA_0	TGRC_0
	TGRB_0	TGRD_0
3	TGRA_3	TGRC_3
	TGRB_3	TGRD_3

- 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 12.12.

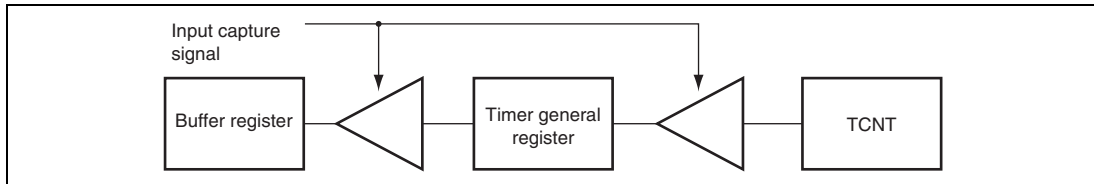


**Figure 12.12 Compare Match Buffer Operation**

- 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 TGR is transferred to the buffer register.

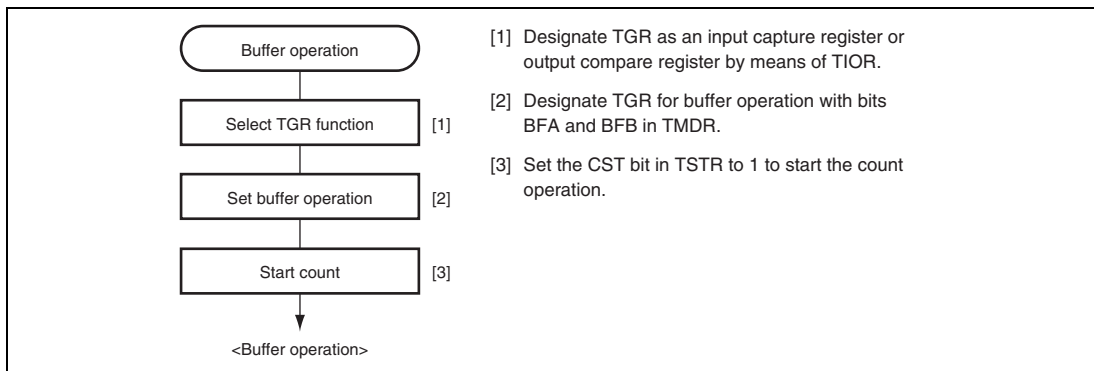
This operation is illustrated in figure 12.13.



**Figure 12.13 Input Capture Buffer Operation**

### (1) Example of Buffer Operation Setting Procedure

Figure 12.14 shows an example of the buffer operation setting procedure.



**Figure 12.14 Example of Buffer Operation Setting Procedure**

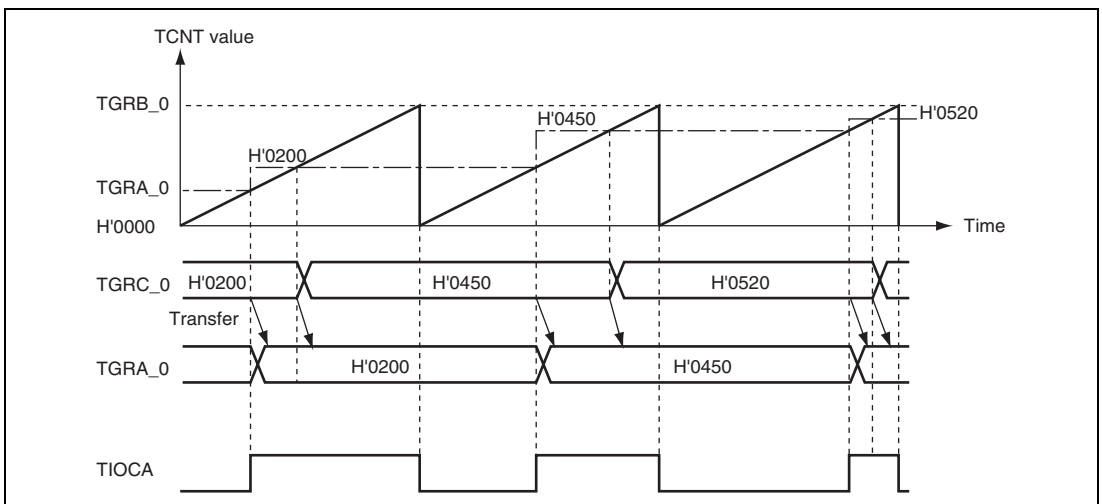
## (2) Examples of Buffer Operation

### (a) When TGR is an output compare register

Figure 12.15 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 TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

For details on PWM modes, see section 12.4.5, PWM Modes.



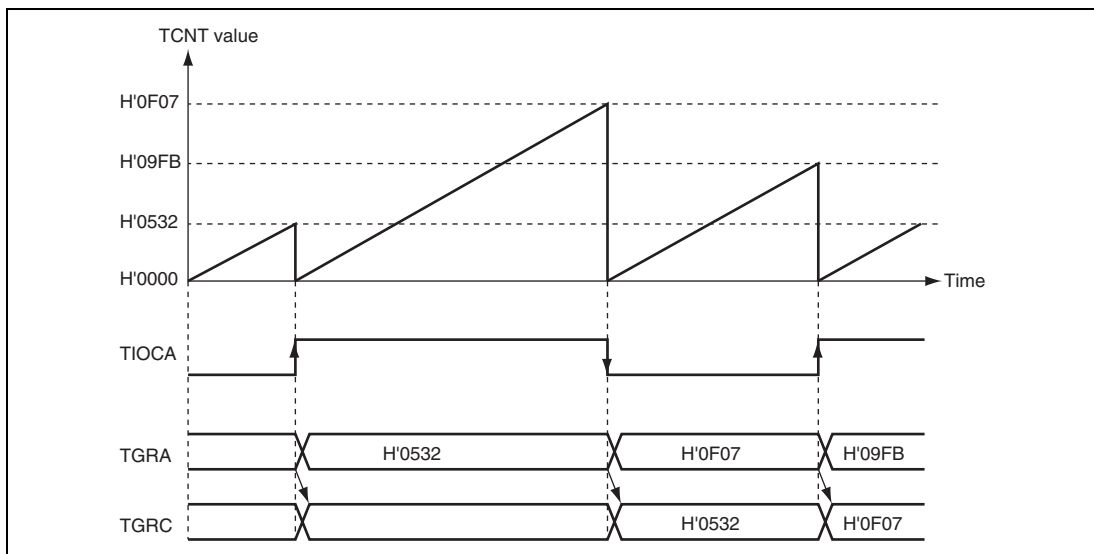
**Figure 12.15 Example of Buffer Operation (1)**

**(b) When TGR is an input capture register**

Figure 12.16 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 12.16 Example of Buffer Operation (2)**

### 12.4.4 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 at overflow/underflow of TCNT\_2 (TCNT\_5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

Table 12.30 shows the register combinations used in cascaded operation.

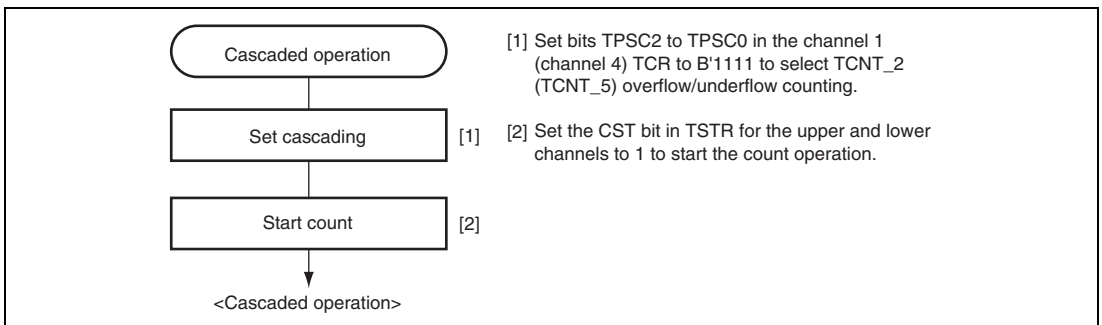
Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

**Table 12.30 Cascaded Combinations**

Combination	Upper 16 Bits	Lower 16 Bits
Channels 1 and 2	TCNT_1	TCNT_2
Channels 4 and 5	TCNT_4	TCNT_5

#### (1) Example of Cascaded Operation Setting Procedure

Figure 12.17 shows an example of the setting procedure for cascaded operation.

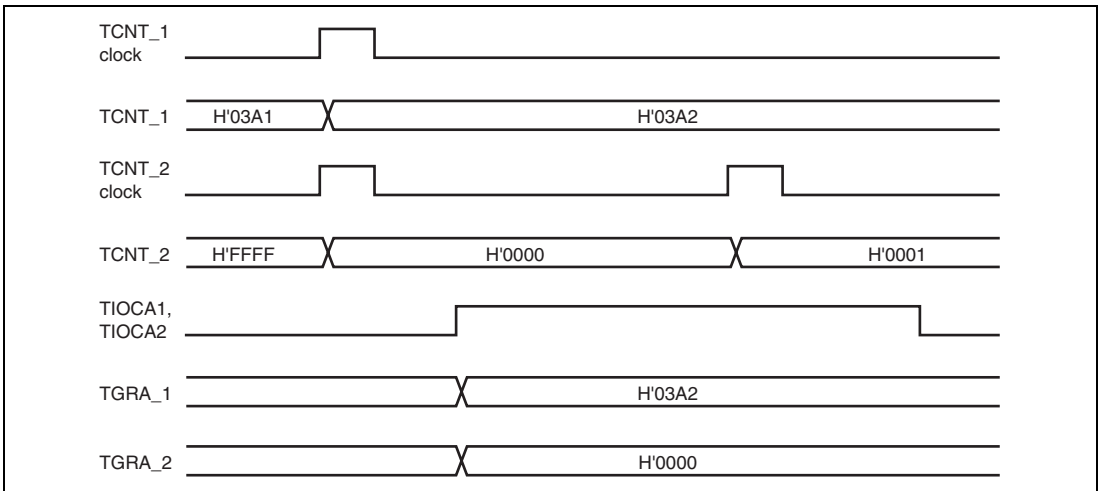


**Figure 12.17 Example of Cascaded Operation Setting Procedure**

## (2) Examples of Cascaded Operation

Figure 12.18 illustrates the operation when counting upon TCNT\_2 overflow/underflow has been set for TCNT\_1, TGRA\_1 and TGRA\_2 have been designated as input capture registers, and the TIOC pin rising edge has been selected.

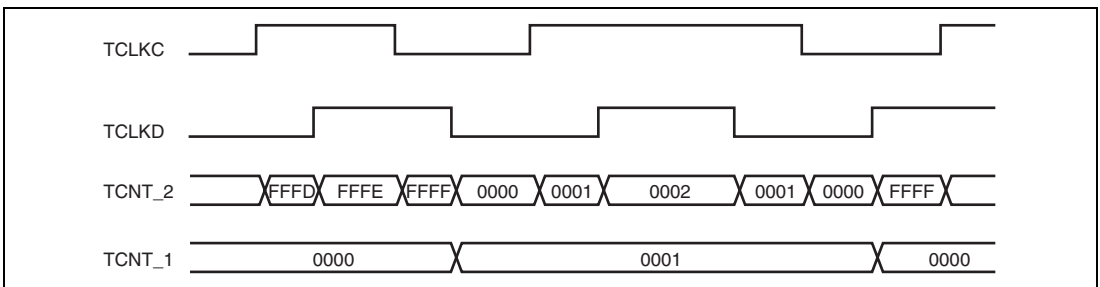
When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGRA\_1, and the lower 16 bits to TGRA\_2.



**Figure 12.18 Example of Cascaded Operation (1)**

Figure 12.19 illustrates the operation when counting upon TCNT\_2 overflow/underflow has been set for TCNT\_1, and phase counting mode has been designated for channel 2.

TCNT\_1 is incremented by TCNT\_2 overflow and decremented by TCNT\_2 underflow.



**Figure 12.19 Example of Cascaded Operation (2)**

### 12.4.5 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.

Settings of TGR registers can output a PWM waveform in the range of 0% to 100% duty cycle.

Designating TGR compare match as the counter clearing source enables the cycle 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.

#### (a) PWM mode 1

PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The outputs specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR are output from the TIOCA and TIOCC pins at compare matches A and C, respectively. The outputs specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR are output at compare matches B and D, respectively. 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.

#### (b) PWM mode 2

PWM output is generated using one TGR as the cycle register and the others as duty cycle registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronous register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty cycle 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 12.31.

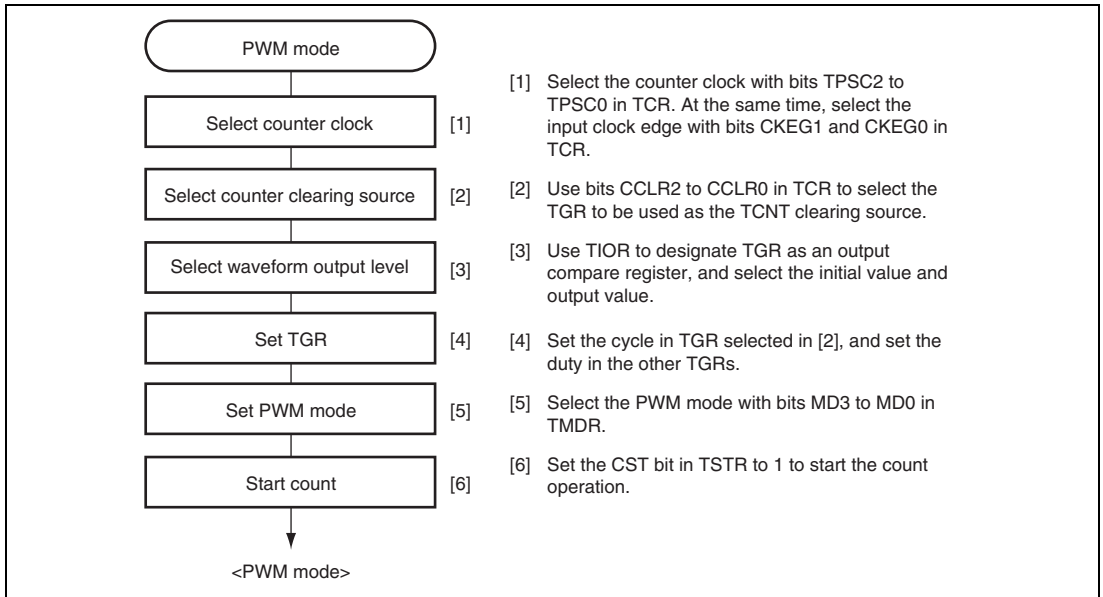
**Table 12.31 PWM Output Registers and Output Pins**

Channel	Registers	Output Pins	
		PWM Mode 1	PWM Mode 2
0	TGRA_0	TIOCA0	TIOCA0
	TGRB_0		TIOCB0
	TGRC_0	TIOCC0	TIOCC0
	TGRD_0		TIOCD0
1	TGRA_1	TIOCA1	TIOCA1
	TGRB_1		TIOCB1
2	TGRA_2	TIOCA2	TIOCA2
	TGRB_2		TIOCB2
3	TGRA_3	TIOCA3	TIOCA3
	TGRB_3		TIOCB3
	TGRC_3	TIOCC3	TIOCC3
	TGRD_3		TIOCD3
4	TGRA_4	TIOCA4	TIOCA4
	TGRB_4		TIOCB4
5	TGRA_5	TIOCA5	TIOCA5
	TGRB_5		TIOCB5

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the cycle is set.

## (2) Example of PWM Mode Setting Procedure

Figure 12.20 shows an example of the PWM mode setting procedure.



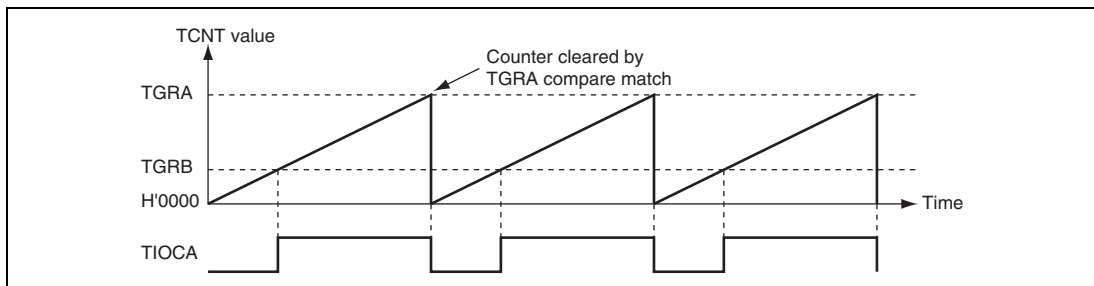
**Figure 12.20 Example of PWM Mode Setting Procedure**

### (3) Examples of PWM Mode Operation

Figure 12.21 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 cycle, and the value set in TGRB register as the duty cycle.

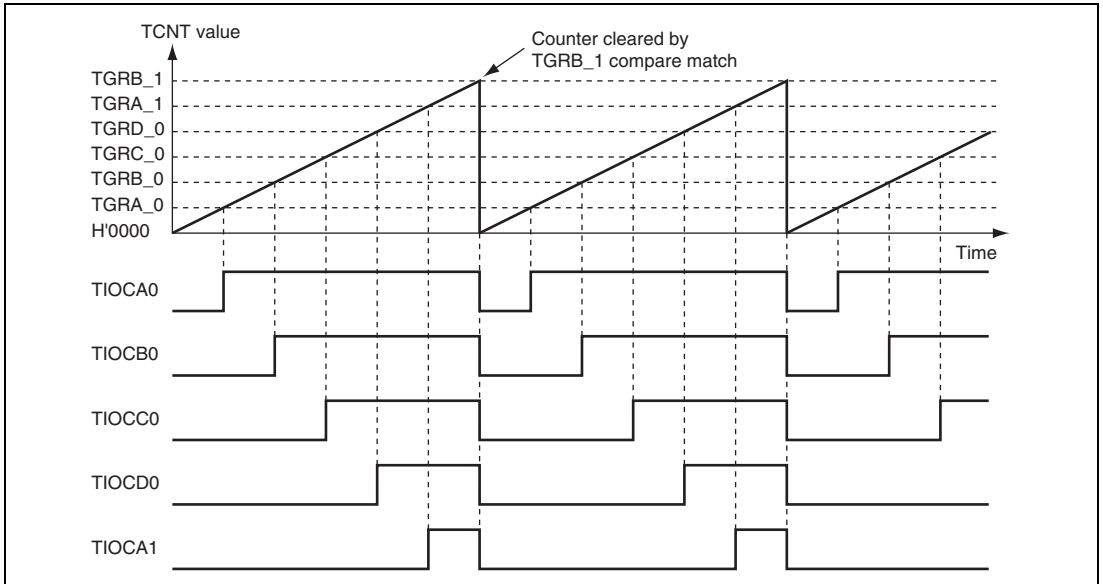


**Figure 12.21 Example of PWM Mode Operation (1)**

Figure 12.22 shows an example of PWM mode 2 operation.

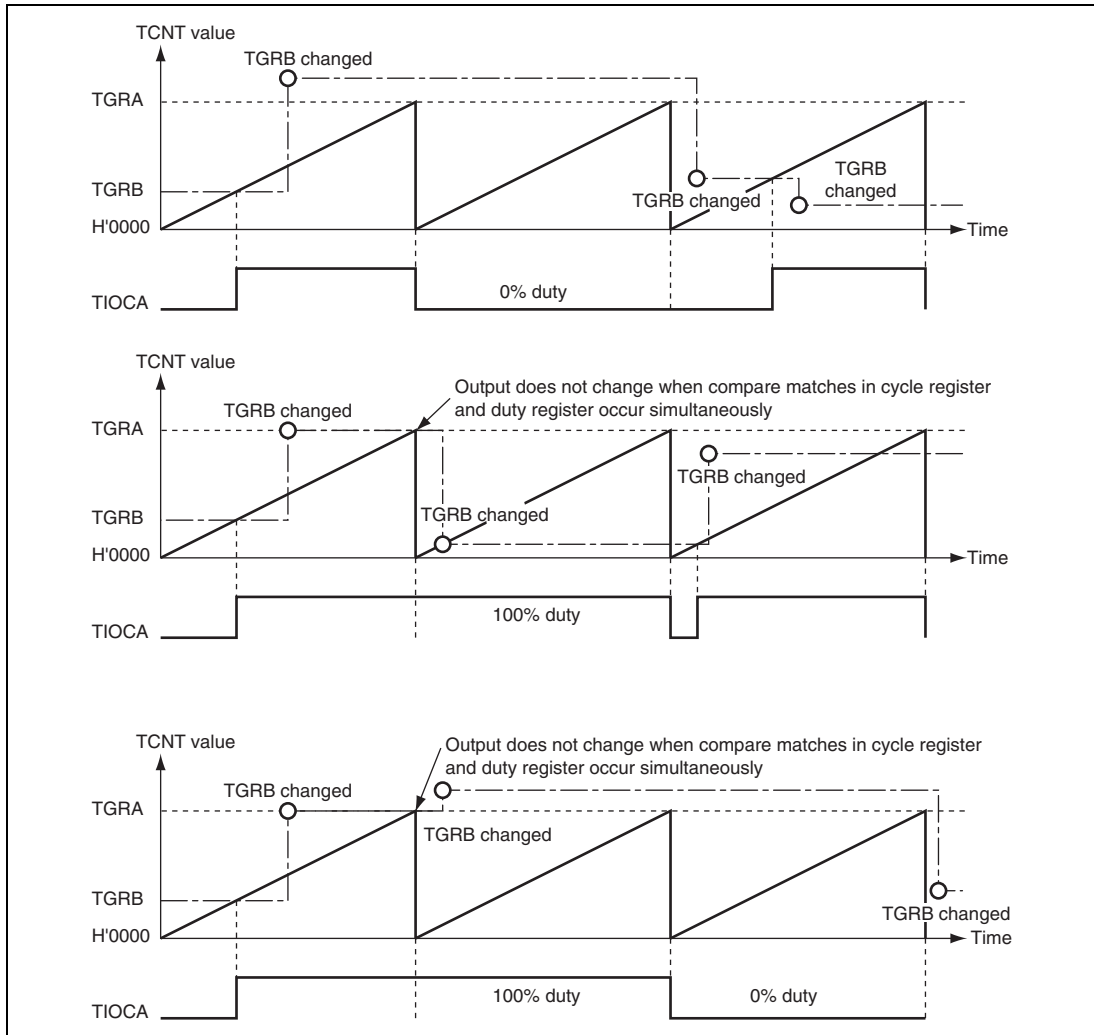
In this example, synchronous operation is designated for channels 0 and 1, TGRB\_1 compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGRA\_0 to TGRD\_0, TGRA\_1), to output a 5-phase PWM waveform.

In this case, the value set in TGRB\_1 is used as the cycle, and the values set in the other TGRs as the duty cycle.



**Figure 12.22 Example of PWM Mode Operation (2)**

Figure 12.23 shows examples of PWM waveform output with 0% duty cycle and 100% duty cycle in PWM mode.



**Figure 12.23 Example of PWM Mode Operation (3)**

### 12.4.6 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.

This can be used for two-phase encoder pulse input.

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.

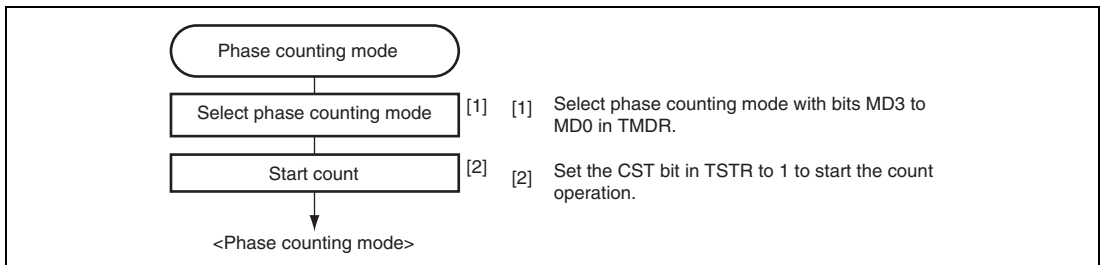
Table 12.32 shows the correspondence between external clock pins and channels.

**Table 12.32 Clock Input Pins in Phase Counting Mode**

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

#### (1) Example of Phase Counting Mode Setting Procedure

Figure 12.24 shows an example of the phase counting mode setting procedure.



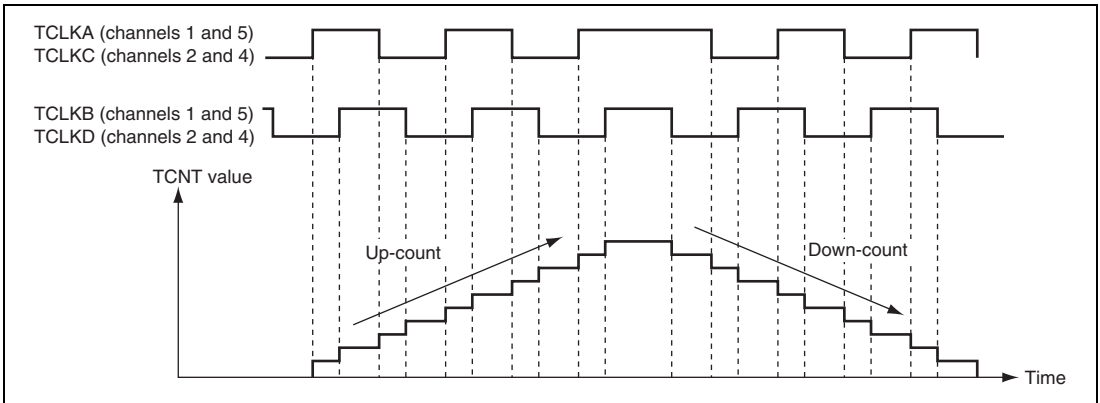
**Figure 12.24 Example of Phase Counting Mode Setting Procedure**

## (2) 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.

### (a) Phase counting mode 1

Figure 12.25 shows an example of phase counting mode 1 operation, and table 12.33 summarizes the TCNT up/down-count conditions.



**Figure 12.25 Example of Phase Counting Mode 1 Operation**

**Table 12.33 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	Down-count
	High level	
High level		Down-count
Low level		
	High level	Up-count
	Low level	

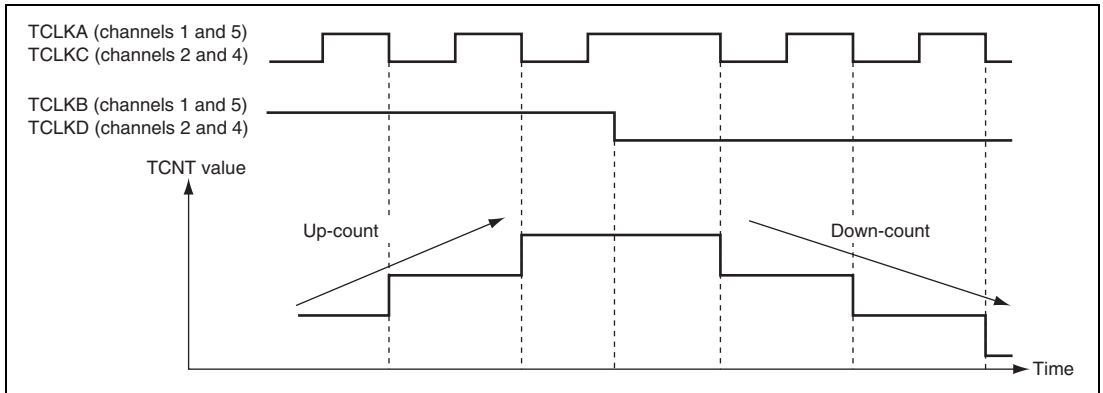
[Legend]

: Rising edge

: Falling edge

**(b) Phase counting mode 2**

Figure 12.26 shows an example of phase counting mode 2 operation, and table 12.34 summarizes the TCNT up/down-count conditions.



**Figure 12.26 Example of Phase Counting Mode 2 Operation**

**Table 12.34 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

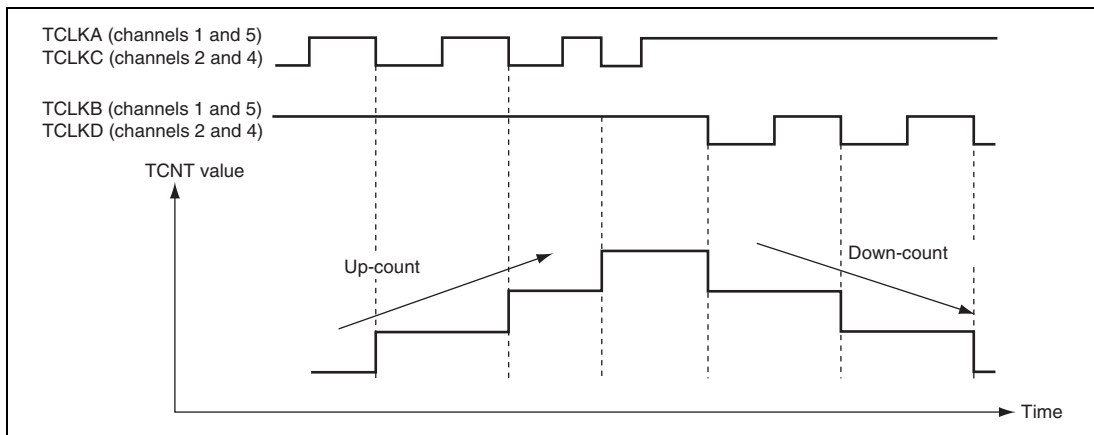
[Legend]

: Rising edge  
: Falling edge



**(c) Phase counting mode 3**

Figure 12.27 shows an example of phase counting mode 3 operation, and table 12.35 summarizes the TCNT up/down-count conditions.



**Figure 12.27 Example of Phase Counting Mode 3 Operation**

**Table 12.35 Up/Down-Count Conditions in Phase Counting Mode 3**

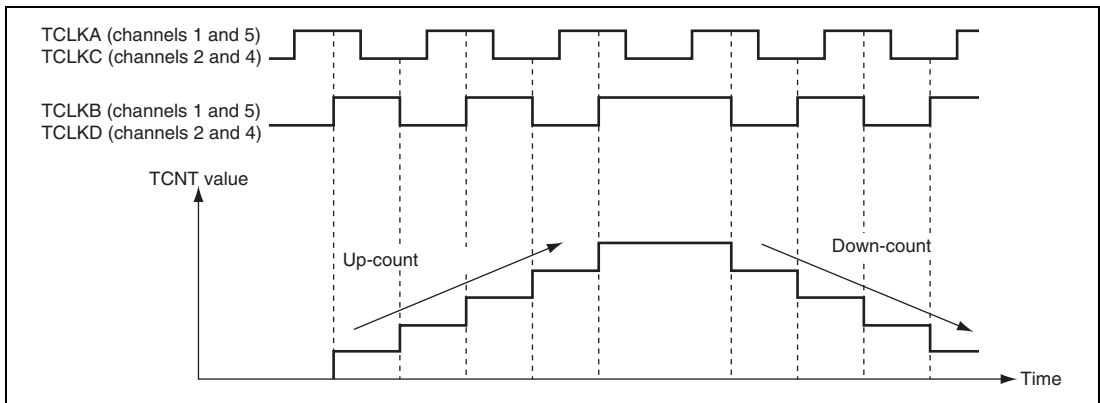
TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level		Don't care
Low level		Don't care
	Low level	Don't care
	High level	Up-count
High level		Down-count
Low level		Don't care
	High level	Don't care
	Low level	Don't care

[Legend]

: Rising edge  
: Falling edge

**(d) Phase counting mode 4**

Figure 12.28 shows an example of phase counting mode 4 operation, and table 12.36 summarizes the TCNT up/down-count conditions.



**Figure 12.28 Example of Phase Counting Mode 4 Operation**

**Table 12.36 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

### (3) Phase Counting Mode Application Example

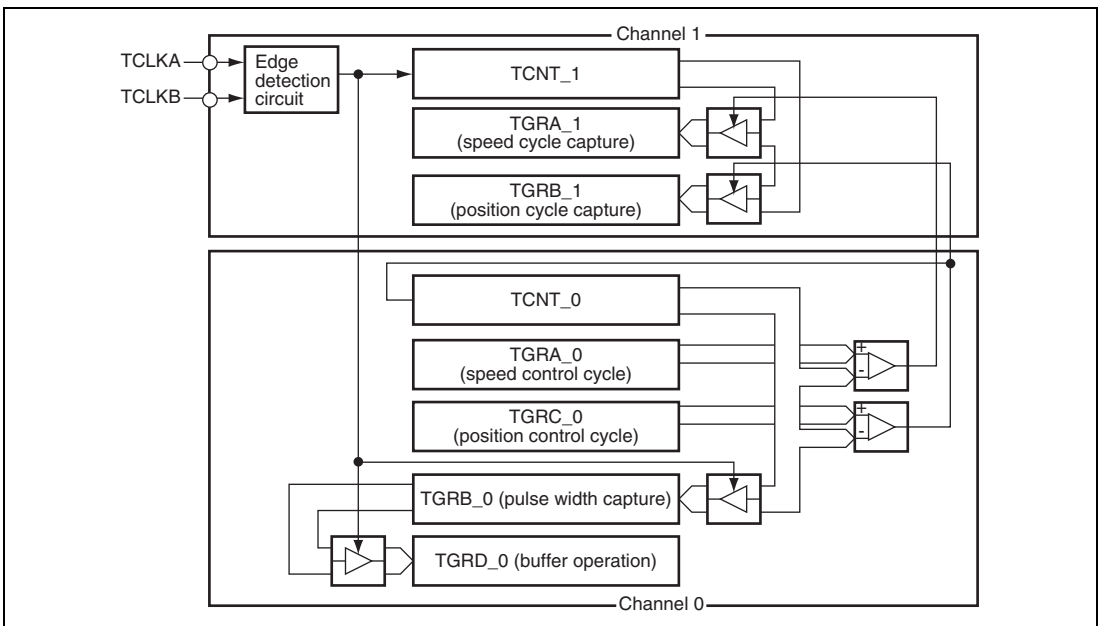
Figure 12.29 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

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 TGRC\_0 compare match; TGRA\_0 and TGRC\_0 are used for the compare match function and are set with the speed control cycle and position control cycle. TGRB\_0 is used for input capture, with TGRB\_0 and TGRD\_0 operating in buffer mode. The channel 1 counter input clock is designated as the TGRB\_0 input capture source, and the pulse width of 2-phase encoder 4-multiplication pulses is detected.

TGRA\_1 and TGRB\_1 for channel 1 are designated for input capture, channel 0 TGRA\_0 and TGRC\_0 compare matches are selected as the input capture source, and the up/down-counter values for the control cycles are stored.

This procedure enables accurate position/speed detection to be achieved.



**Figure 12.29 Phase Counting Mode Application Example**

## 12.5 Interrupt Sources

There are three kinds of TPU interrupt sources: 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 priority levels can be changed by the interrupt controller, but the priority within a channel is fixed. For details, see section 6, Interrupt Controller.

Table 12.37 lists the TPU interrupt sources.

**Table 12.37 TPU Interrupts**

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation
0	TGI0A	TGRA_0 input capture/ compare match	TGFA_0	Possible	Possible
	TGI0B	TGRB_0 input capture/ compare match	TGFB_0	Possible	Not possible
	TGI0C	TGRC_0 input capture/ compare match	TGFC_0	Possible	Not possible
	TGI0D	TGRD_0 input capture/ compare match	TGFD_0	Possible	Not possible
	TCI0V	TCNT_0 overflow	TCFV_0	Not possible	Not possible
1	TGI1A	TGRA_1 input capture/ compare match	TGFA_1	Possible	Possible
	TGI1B	TGRB_1 input capture/ compare match	TGFB_1	Possible	Not possible
	TCI1V	TCNT_1 overflow	TCFV_1	Not possible	Not possible
	TCI1U	TCNT_1 underflow	TCFU_1	Not possible	Not possible

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation
2	TGI2A	TGRA_2 input capture/ compare match	TGFA_2	Possible	Possible
	TGI2B	TGRB_2 input capture/ compare match	TGFB_2	Possible	Not possible
	TCI2V	TCNT_2 overflow	TCFV_2	Not possible	Not possible
	TCI2U	TCNT_2 underflow	TCFU_2	Not possible	Not possible
3	TGI3A	TGRA_3 input capture/ compare match	TGFA_3	Possible	Possible
	TGI3B	TGRB_3 input capture/ compare match	TGFB_3	Possible	Not possible
	TGI3C	TGRC_3 input capture/ compare match	TGFC_3	Possible	Not possible
	TGI3D	TGRD_3 input capture/ compare match	TGFD_3	Possible	Not possible
	TCI3V	TCNT_3 overflow	TCFV_3	Not possible	Not possible
4	TGI4A	TGRA_4 input capture/ compare match	TGFA_4	Possible	Possible
	TGI4B	TGRB_4 input capture/ compare match	TGFB_4	Possible	Not possible
	TCI4V	TCNT_4 overflow	TCFV_4	Not possible	Not possible
	TCI4U	TCNT_4 underflow	TCFU_4	Not possible	Not possible
5	TGI5A	TGRA_5 input capture/ compare match	TGFA_5	Possible	Possible
	TGI5B	TGRB_5 input capture/ compare match	TGFB_5	Possible	Not possible
	TCI5V	TCNT_5 overflow	TCFV_5	Not possible	Not possible
	TCI5U	TCNT_5 underflow	TCFU_5	Not possible	Not possible

Note: This table shows the initial state immediately after a reset. The relative channel priority levels can be changed by the interrupt controller.

### **(1) 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 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.

### **(2) 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 a 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.

### **(3) 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 a 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.

## 12.6 DTC Activation

The DTC can be activated by the TGR input capture/compare match interrupt for a channel. For details, see section 10, Data Transfer Controller (DTC).

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.

## 12.7 DMAC Activation

The DMAC can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 9, DMA Controller (DMAC).

In TPU, one in each channel, totally six TGRA input capture/compare match interrupts can be used as DMAC activation sources.

## 12.8 A/D Converter Activation

The TGRA input capture/compare match for each channel can activate the A/D converter and  $\Delta\Sigma$  A/D converter.

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 and  $\Delta\Sigma$  A/D converter. If the TPU conversion start trigger has been selected on the A/D converter or  $\Delta\Sigma$  A/D converter side at this time, A/D conversion is started.

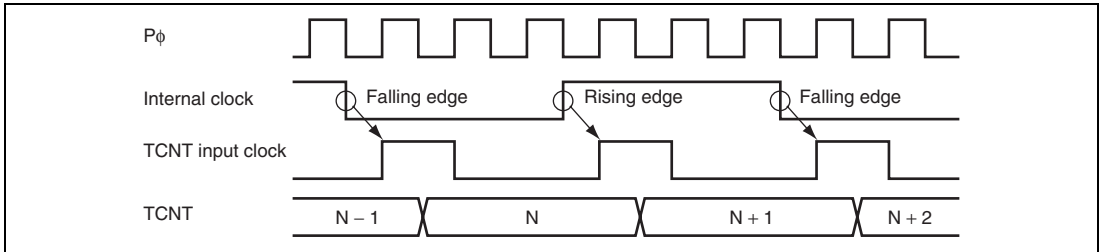
In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

## 12.9 Operation Timing

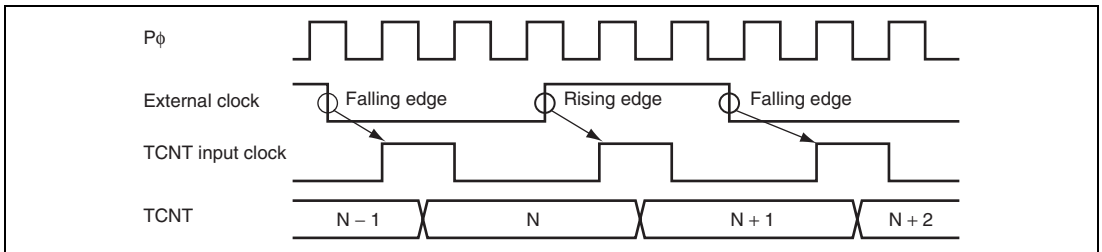
### 12.9.1 Input/Output Timing

#### (1) TCNT Count Timing

Figure 12.30 shows TCNT count timing in internal clock operation, and figure 12.31 shows TCNT count timing in external clock operation.



**Figure 12.30 Count Timing in Internal Clock Operation**



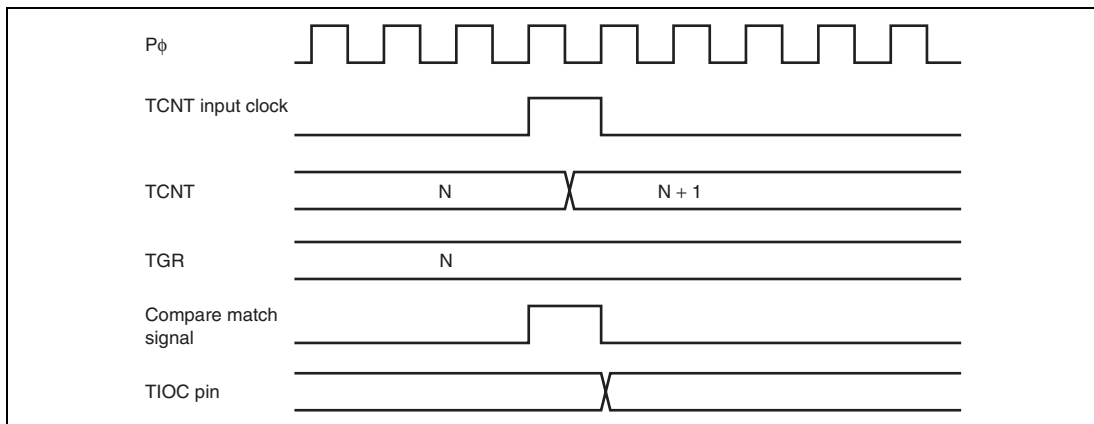
**Figure 12.31 Count Timing in External Clock Operation**



## (2) 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 (TIOC pin). After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

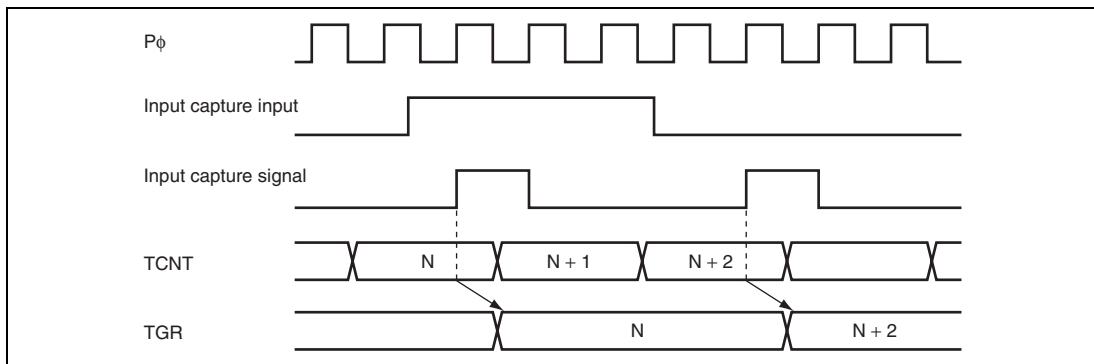
Figure 12.32 shows output compare output timing.



**Figure 12.32 Output Compare Output Timing**

## (3) Input Capture Signal Timing

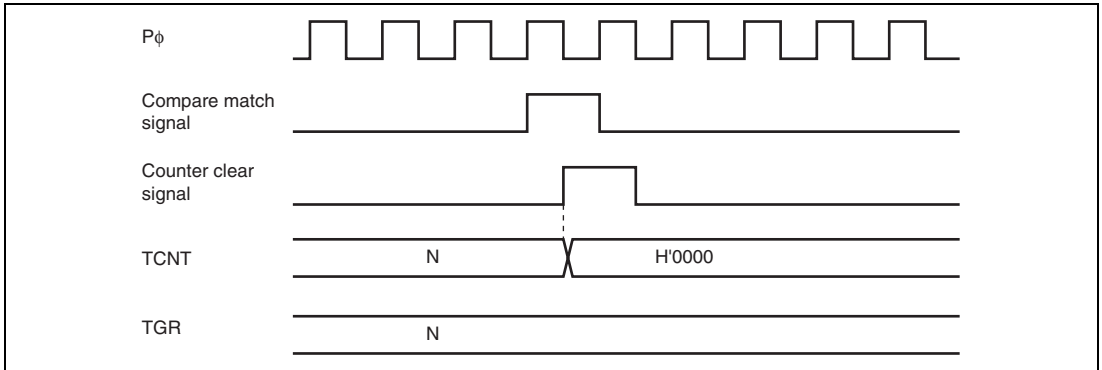
Figure 12.33 shows input capture signal timing.



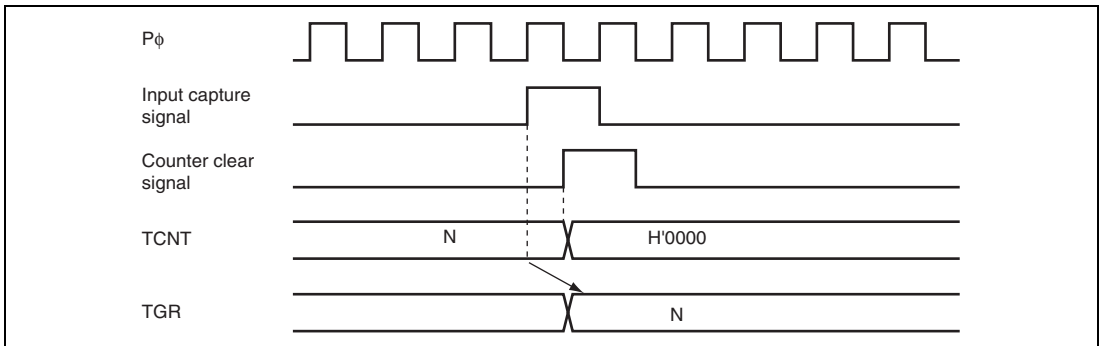
**Figure 12.33 Input Capture Input Signal Timing**

**(4) Timing for Counter Clearing by Compare Match/Input Capture**

Figure 12.34 shows the timing when counter clearing by compare match occurrence is specified, and figure 12.35 shows the timing when counter clearing by input capture occurrence is specified.



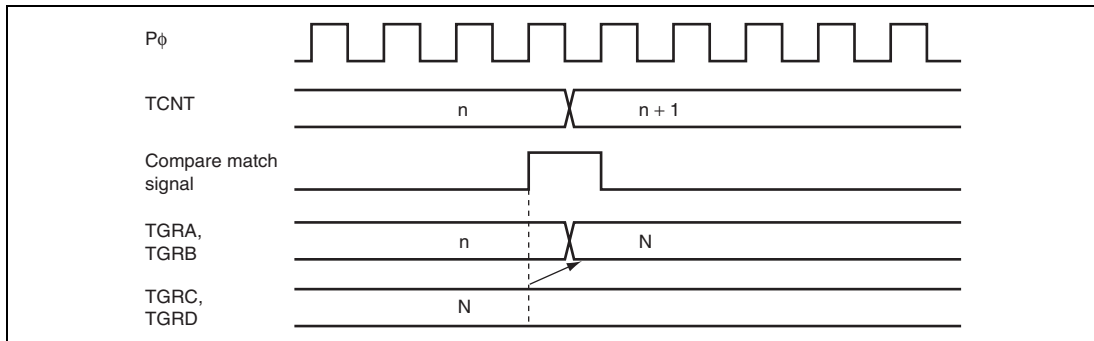
**Figure 12.34 Counter Clear Timing (Compare Match)**



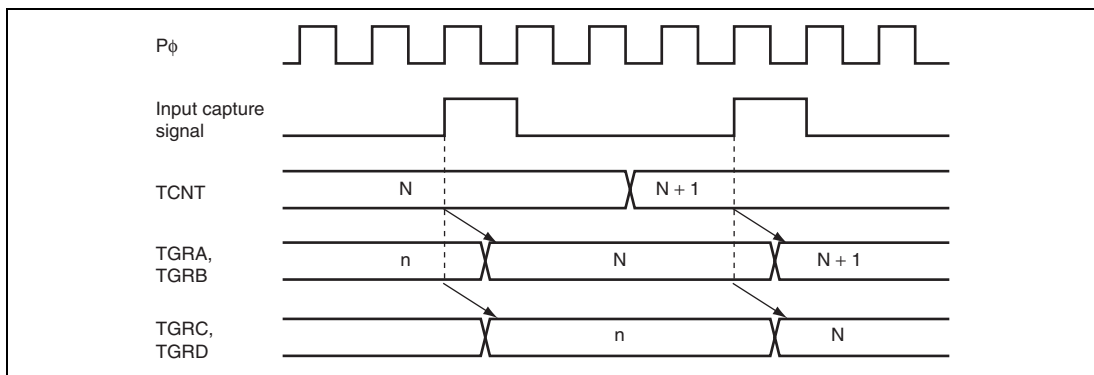
**Figure 12.35 Counter Clear Timing (Input Capture)**

## (5) Buffer Operation Timing

Figures 12.36 and 12.37 show the timings in buffer operation.



**Figure 12.36 Buffer Operation Timing (Compare Match)**

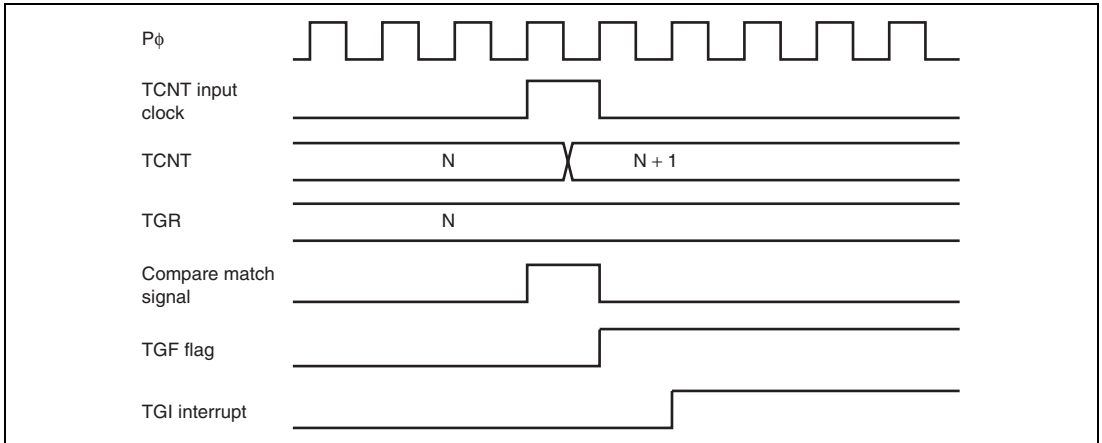


**Figure 12.37 Buffer Operation Timing (Input Capture)**

## 12.9.2 Interrupt Signal Timing

### (1) TGF Flag Setting Timing in Case of Compare Match

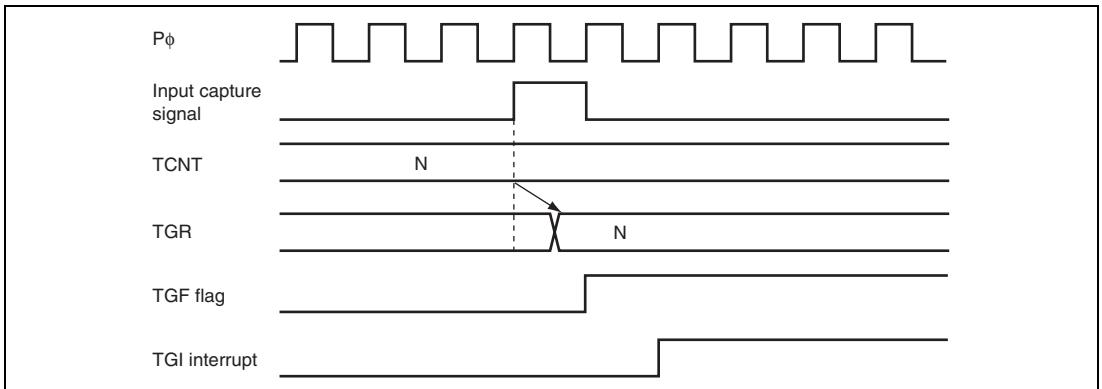
Figure 12.38 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and the TGI interrupt request signal timing.



**Figure 12.38 TGI Interrupt Timing (Compare Match)**

### (2) TGF Flag Setting Timing in Case of Input Capture

Figure 12.39 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and the TGI interrupt request signal timing.

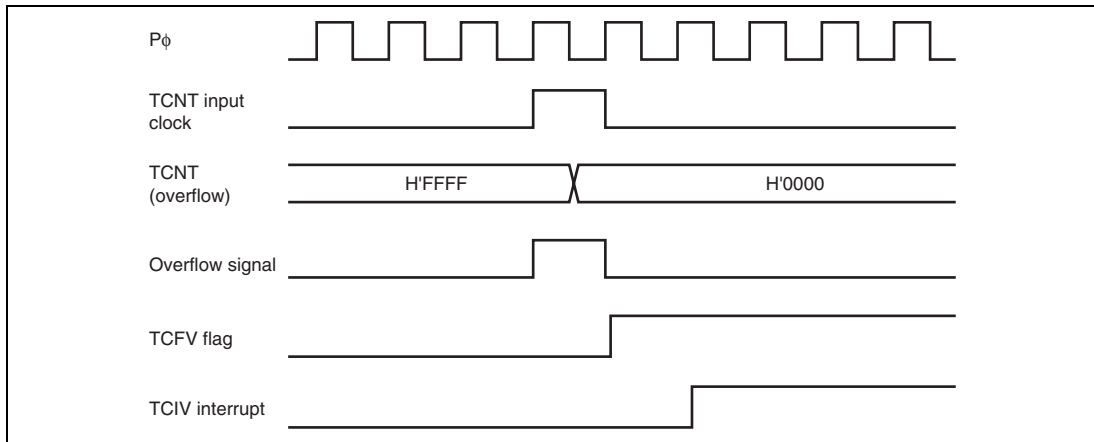


**Figure 12.39 TGI Interrupt Timing (Input Capture)**

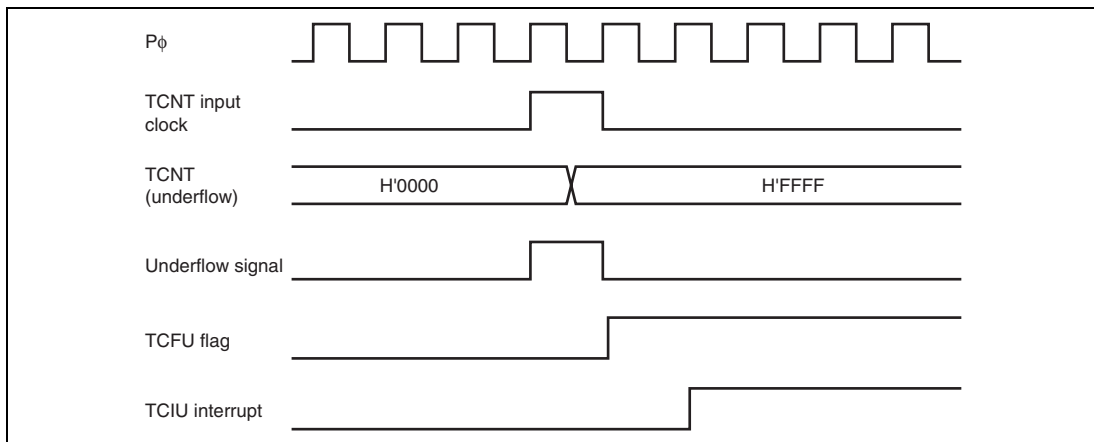
### (3) TCFV Flag/TCFU Flag Setting Timing

Figure 12.40 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and the TCIV interrupt request signal timing.

Figure 12.41 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and the TCIU interrupt request signal timing.



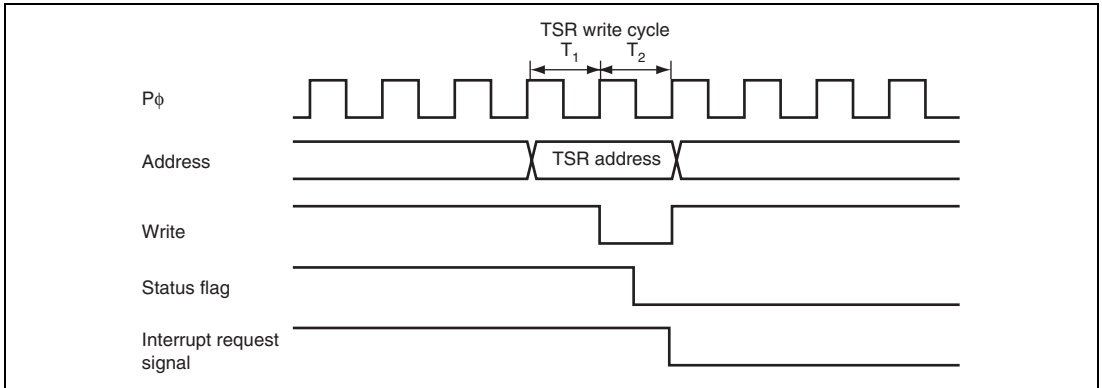
**Figure 12.40 TCIV Interrupt Setting Timing**



**Figure 12.41 TCIU Interrupt Setting Timing**

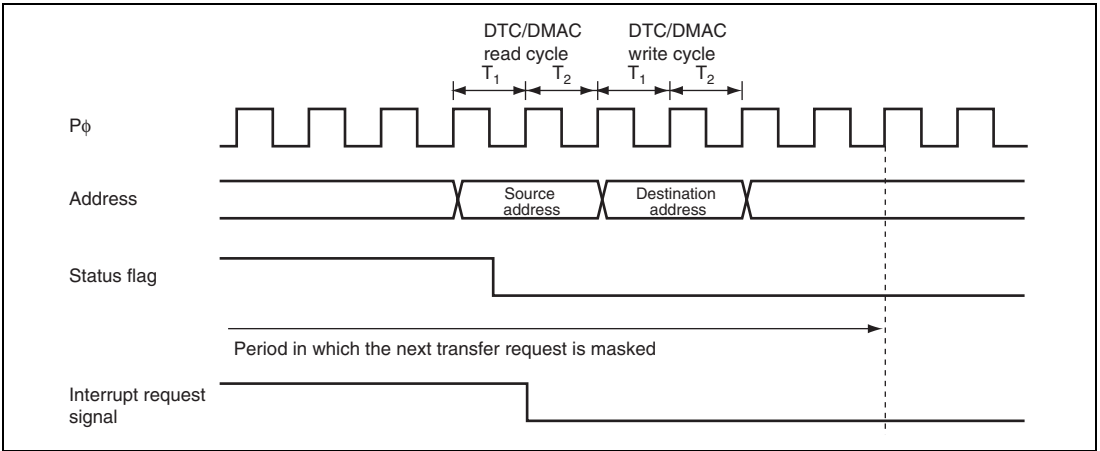
#### (4) Status Flag Clearing Timing

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 12.42 shows the timing for status flag clearing by the CPU, and figures 12.43 and 12.44 show the timing for status flag clearing by the DTC or DMAC.

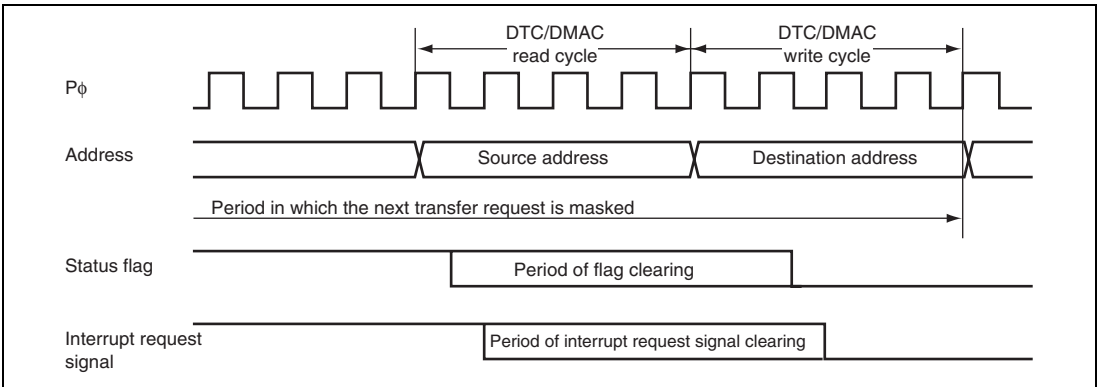


**Figure 12.42 Timing for Status Flag Clearing by CPU**

The status flag and interrupt request signal are cleared in synchronization with  $P\phi$  after the DTC or DMAC transfer has started, as shown in figure 12.43. If conflict occurs for clearing the status flag and interrupt request signal due to activation of multiple DTC or DMAC transfers, it will take up to five clock cycles ( $P\phi$ ) for clearing them, as shown in figure 12.44. The next transfer request is masked for a longer period of either a period until the current transfer ends or a period for five clock cycles ( $P\phi$ ) from the beginning of the transfer. Note that in the DTC transfer, the status flag may be cleared during outputting the destination address.



**Figure 12.43 Timing for Status Flag Clearing by DTC or DMAC Activation (1)**



**Figure 12.44 Timing for Status Flag Clearing by DTC or DMAC Activation (2)**

## 12.10 Usage Notes

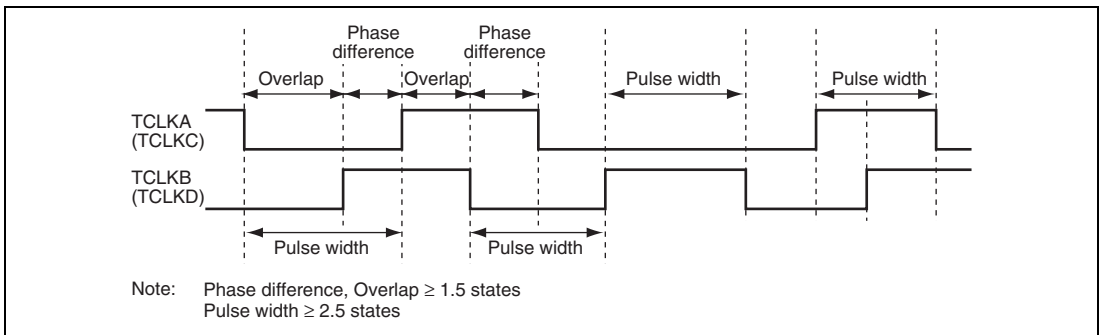
### 12.10.1 Module Stop Function Setting

Operation of the TPU can be disabled or enabled using the module stop control register. The initial setting is for operation of the TPU to be halted. Register access is enabled by clearing module stop state. For details, see section 24, Power-Down Modes.

### 12.10.2 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 12.45 shows the input clock conditions in phase counting mode.



**Figure 12.45 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode**



### 12.10.3 Caution on Cycle 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:

$$f = \frac{P\phi}{(N + 1)}$$

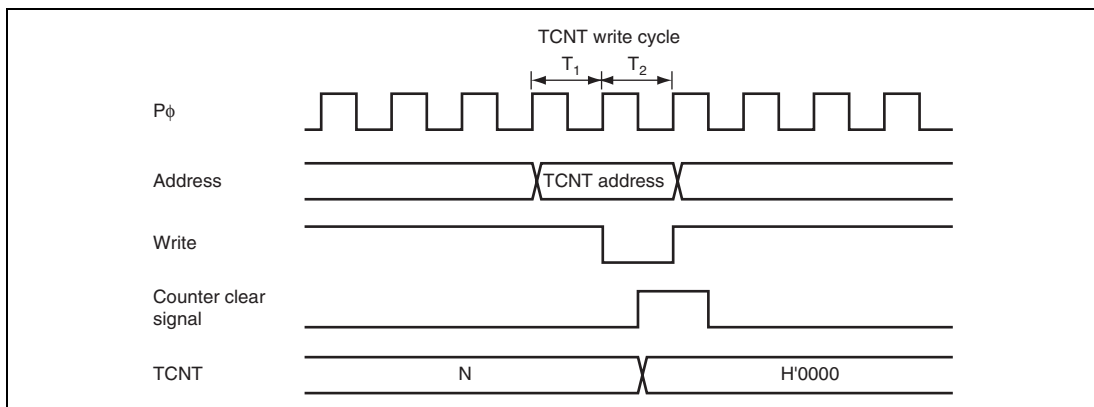
f: Counter frequency

P $\phi$ : Operating frequency

N: TGR set value

### 12.10.4 Conflict between TCNT Write and Clear Operations

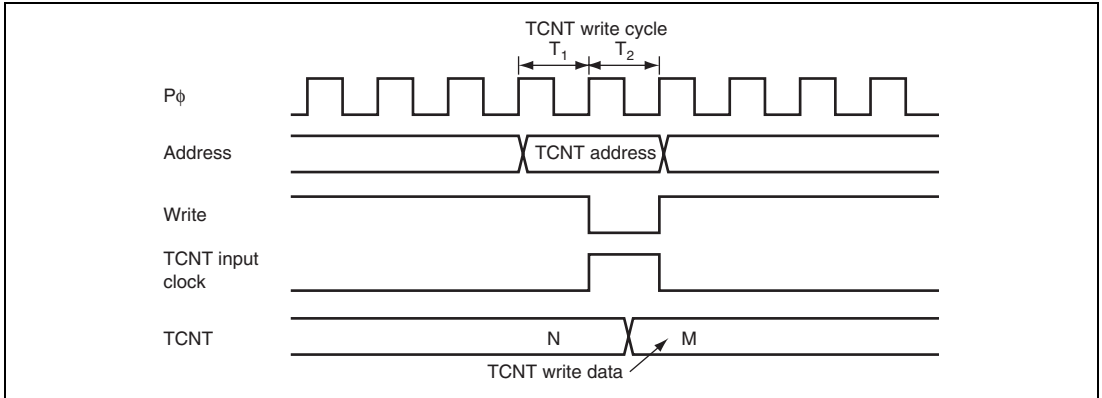
If the counter clearing signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed. Figure 12.46 shows the timing in this case.



**Figure 12.46 Conflict between TCNT Write and Clear Operations**

### 12.10.5 Conflict between TCNT Write and Increment Operations

If incrementing occurs in the T2 state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented. Figure 12.47 shows the timing in this case.

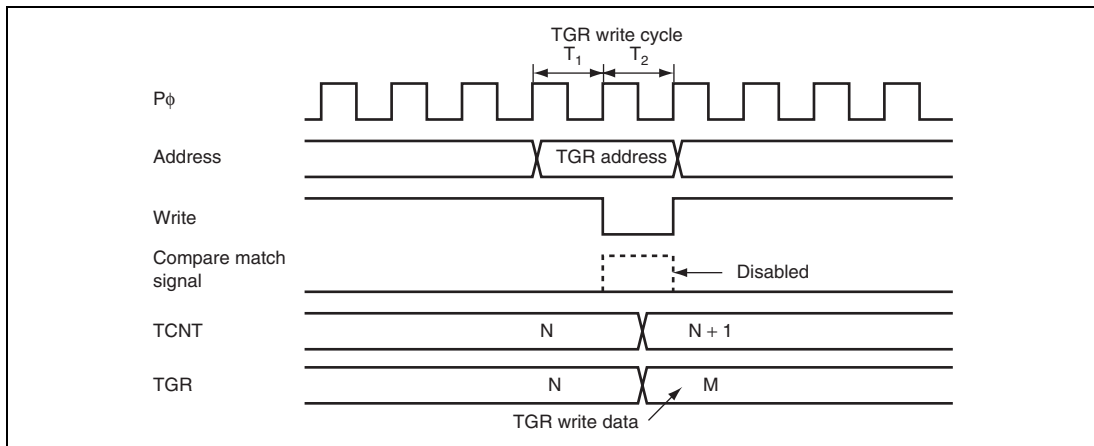


**Figure 12.47 Conflict between TCNT Write and Increment Operations**

### 12.10.6 Conflict between TGR Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the TGR write takes precedence and the compare match signal is disabled. A compare match also does not occur when the same value as before is written.

Figure 12.48 shows the timing in this case.

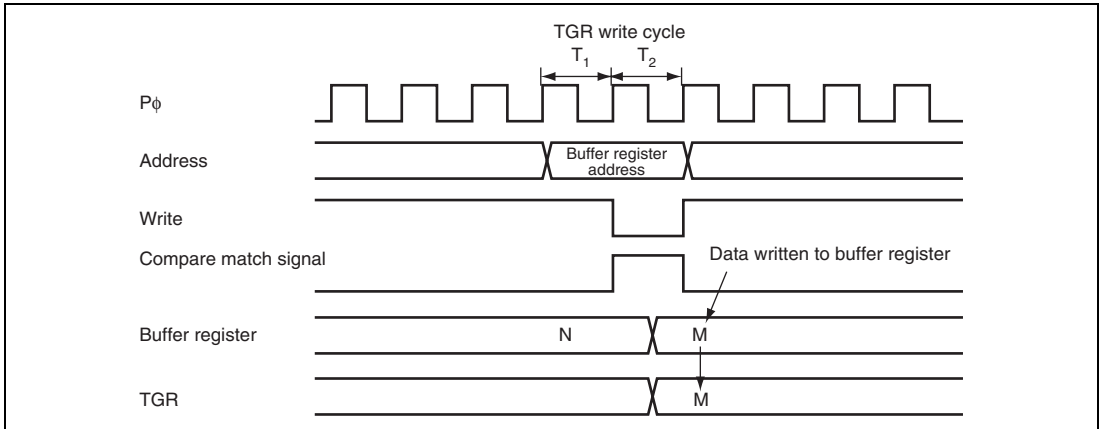


**Figure 12.48 Conflict between TGR Write and Compare Match**

### 12.10.7 Conflict between Buffer Register Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the write data.

Figure 12.49 shows the timing in this case.

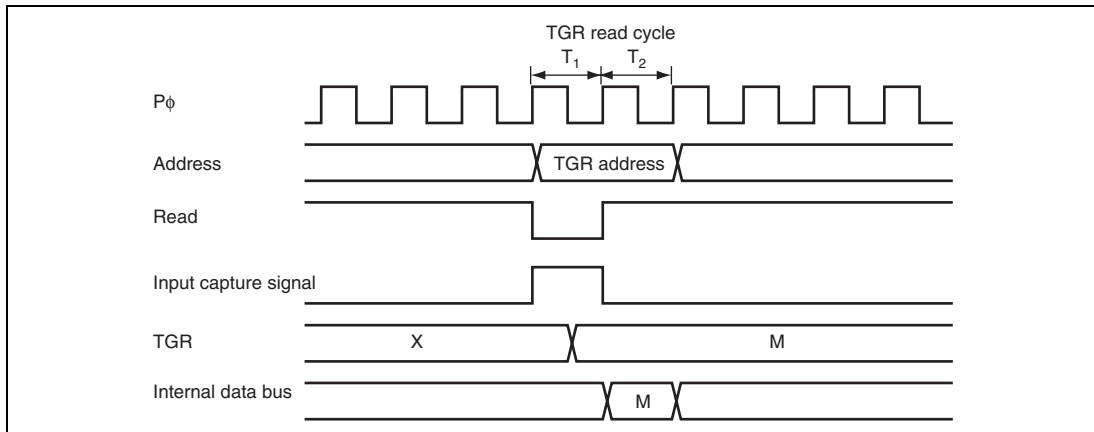


**Figure 12.49 Conflict between Buffer Register Write and Compare Match**

### 12.10.8 Conflict between TGR Read and Input Capture

If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is read will be the data after input capture transfer.

Figure 12.50 shows the timing in this case.

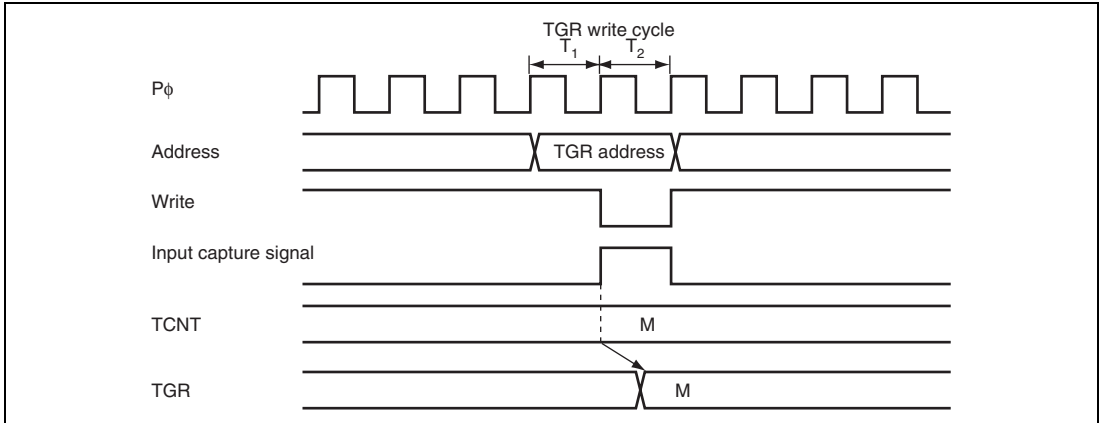


**Figure 12.50 Conflict between TGR Read and Input Capture**

### 12.10.9 Conflict between TGR Write and Input Capture

If the input capture signal is generated in the T2 state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

Figure 12.51 shows the timing in this case.

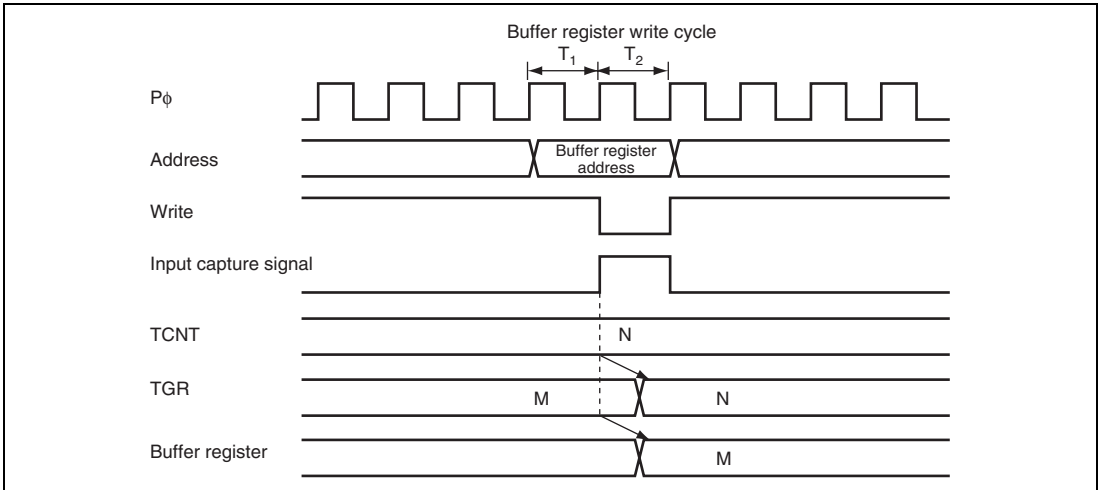


**Figure 12.51 Conflict between TGR Write and Input Capture**

### 12.10.10 Conflict between Buffer Register Write and Input Capture

If the input capture signal is generated in the T2 state of a buffer register write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

Figure 12.52 shows the timing in this case.

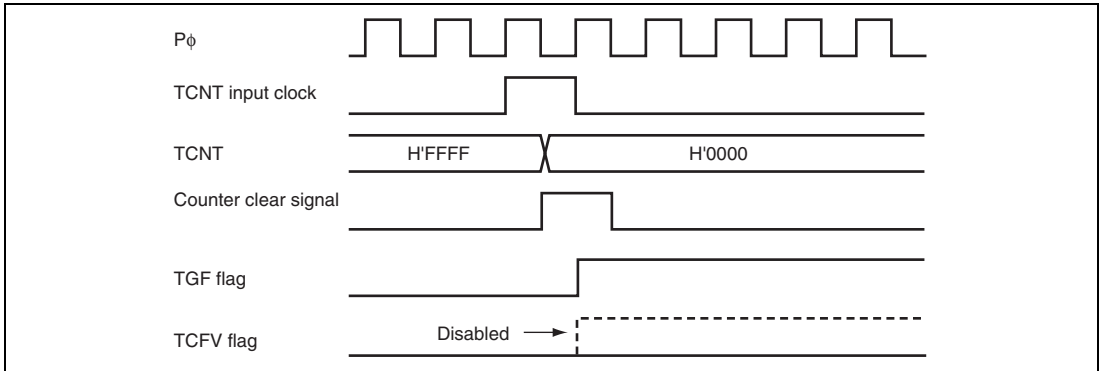


**Figure 12.52 Conflict between Buffer Register Write and Input Capture**

### 12.10.11 Conflict 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 12.53 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.



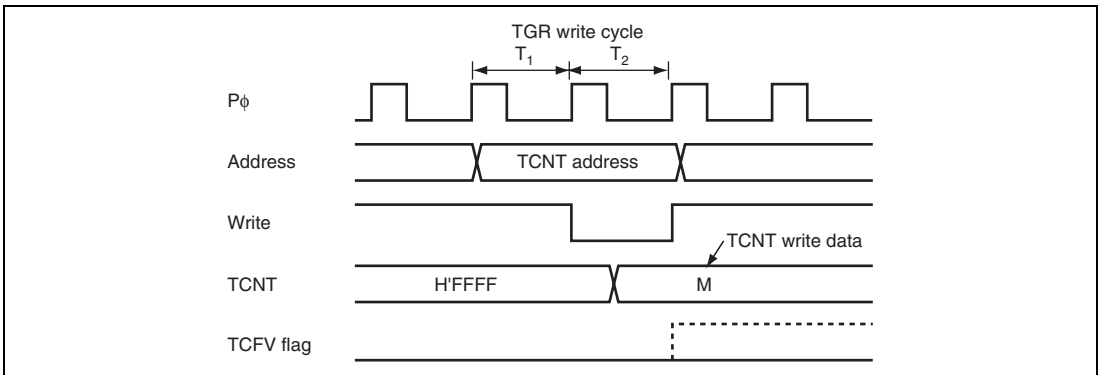
**Figure 12.53 Conflict between Overflow and Counter Clearing**



### 12.10.12 Conflict between TCNT Write and Overflow/Underflow

If an overflow/underflow occurs due to increment/decrement in the T2 state of a TCNT write cycle, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 12.54 shows the operation timing when there is conflict between TCNT write and overflow.



**Figure 12.54 Conflict between TCNT Write and Overflow**

### 12.10.13 Multiplexing of I/O Pins

In this LSI, 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.

### 12.10.14 Interrupts and Module Stop State

If module stop state 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 state.

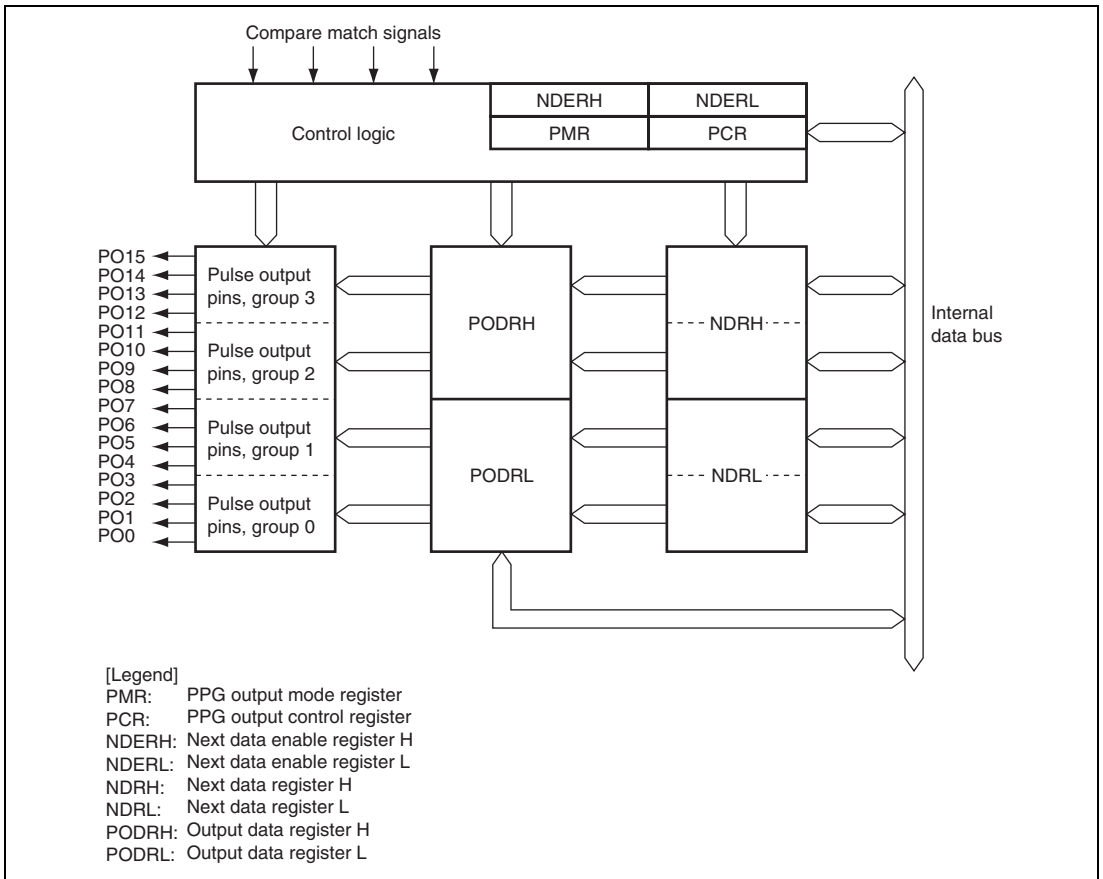


## Section 13 Programmable Pulse Generator (PPG)

The programmable pulse generator (PPG) 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 (groups 3 to 0) that can operate both simultaneously and independently. Figure 13.1 shows a block diagram of the PPG.

### 13.1 Features

- 16-bit output data
- Four output groups
- Selectable output trigger signals
- Non-overlapping mode
- Can operate together with the data transfer controller (DTC) and DMA controller (DMAC)
- Inverted output can be set
- Module stop state specifiable



**Figure 13.1 Block Diagram of PPG**

## 13.2 Input/Output Pins

Table 13.1 shows the PPG pin configuration.

**Table 13.1 Pin Configuration**

Pin Name	I/O	Function
PO15	Output	Group 3 pulse output
PO14	Output	
PO13	Output	
PO12	Output	
PO11	Output	Group 2 pulse output
PO10	Output	
PO9	Output	
PO8	Output	
PO7	Output	Group 1 pulse output
PO6	Output	
PO5	Output	
PO4	Output	
PO3	Output	Group 0 pulse output
PO2	Output	
PO1	Output	
PO0	Output	

### 13.3 Register Descriptions

The PPG has the following registers.

- Next data enable register H (NDERH)
- Next data enable register L (NDERL)
- Output data register H (PODRH)
- Output data register L (PODRL)
- Next data register H (NDRH)
- Next data register L (NDRL)
- PPG output control register (PCR)
- PPG output mode register (PMR)

#### 13.3.1 Next Data Enable Registers H, L (NDERH, NDERL)

NDERH and NDERL enable/disable pulse output on a bit-by-bit basis.

- NDERH

Bit	7	6	5	4	3	2	1	0
Bit Name	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
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
7	NDER15	0	R/W	Next Data Enable 15 to 8
6	NDER14	0	R/W	When a bit is set to 1, the value in the corresponding NDRH bit is transferred to the PODRH bit by the selected output trigger. Values are not transferred from NDRH to PODRH for cleared bits.
5	NDER13	0	R/W	
4	NDER12	0	R/W	
3	NDER11	0	R/W	
2	NDER10	0	R/W	
1	NDER9	0	R/W	
0	NDER8	0	R/W	

- NDERL

Bit	Bit Name	Initial Value	R/W	Description
7	NDER7	0	R/W	Next Data Enable 7 to 0
6	NDER6	0	R/W	When a bit is set to 1, the value in the corresponding NDRL bit is transferred to the PODRL bit by the selected output trigger. Values are not transferred from NDRL to PODRL for cleared bits.
5	NDER5	0	R/W	
4	NDER4	0	R/W	
3	NDER3	0	R/W	
2	NDER2	0	R/W	
1	NDER1	0	R/W	
0	NDER0	0	R/W	

### 13.3.2 Output Data Registers H, L (PODRH, PODRL)

PODRH and PODRL store output data for use in pulse output. A bit that has been set for pulse output by NDER is read-only and cannot be modified.

- PODRH

Bit	7	6	5	4	3	2	1	0
Bit Name	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
Bit Name	POD7	POD6	POD5	POD4	POD3	POD2	POD2	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

- PODRH

Bit	Bit Name	Initial Value	R/W	Description
7	POD15	0	R/W	Output Data Register 15 to 8
6	POD14	0	R/W	For bits which have been set to pulse output by NDERH, the output trigger transfers NDRH values to this register during PPG operation. While NDERH is set to 1, the CPU cannot write to this register. While NDERH is cleared, the initial output value of the pulse can be set.
5	POD13	0	R/W	
4	POD12	0	R/W	
3	POD11	0	R/W	
2	POD10	0	R/W	
1	POD9	0	R/W	
0	POD8	0	R/W	



- PODRL

Bit	Bit Name	Initial Value	R/W	Description
7	POD7	0	R/W	Output Data Register 7 to 0
6	POD6	0	R/W	For bits which have been set to pulse output by NDERL, the output trigger transfers NDRL values to this register during PPG operation. While NDERL is set to 1, the CPU cannot write to this register. While NDERL is cleared, the initial output value of the pulse can be set.
5	POD5	0	R/W	
4	POD4	0	R/W	
3	POD3	0	R/W	
2	POD2	0	R/W	
1	POD1	0	R/W	
0	POD0	0	R/W	

### 13.3.3 Next Data Registers H, L (NDRH, NDRL)

NDRH and NDRL store the next data for pulse output. The NDR addresses differ depending on whether pulse output groups have the same output trigger or different output triggers.

- NDRH

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- NDRL

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- NDRH

If pulse output groups 2 and 3 have the same output trigger, all eight bits are mapped to the same address and can be accessed at one time, as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR15	0	R/W	Next Data Register 15 to 8
6	NDR14	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3	NDR11	0	R/W	
2	NDR10	0	R/W	
1	NDR9	0	R/W	
0	NDR8	0	R/W	

If pulse output groups 2 and 3 have different output triggers, the upper four bits and lower four bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR15	0	R/W	Next Data Register 15 to 12
6	NDR14	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3 to 0	—	All 1	—	

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
3	NDR11	0	R/W	Next Data Register 11 to 8
2	NDR10	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
1	NDR9	0	R/W	
0	NDR8	0	R/W	

- NDRL

If pulse output groups 0 and 1 have the same output trigger, all eight bits are mapped to the same address and can be accessed at one time, as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR7	0	R/W	Next Data Register 7 to 0
6	NDR6	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3	NDR3	0	R/W	
2	NDR2	0	R/W	
1	NDR1	0	R/W	
0	NDR0	0	R/W	

If pulse output groups 0 and 1 have different output triggers, the upper four bits and lower four bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR7	0	R/W	Next Data Register 7 to 4
6	NDR6	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3 to 0	—	All 1	—	

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
3	NDR3	0	R/W	Next Data Register 3 to 0
2	NDR2	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
1	NDR1	0	R/W	
0	NDR0	0	R/W	

### 13.3.4 PPG Output Control Register (PCR)

PCR selects output trigger signals on a group-by-group basis. For details on output trigger selection, refer to section 13.3.5, PPG Output Mode Register (PMR).

Bit	7	6	5	4	3	2	1	0
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
7	G3CMS1	1	R/W	Group 3 Compare Match Select 1 and 0
6	G3CMS0	1	R/W	These bits select output trigger of pulse output group 3. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
5	G2CMS1	1	R/W	Group 2 Compare Match Select 1 and 0
4	G2CMS0	1	R/W	These bits select output trigger of pulse output group 2. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
3	G1CMS1	1	R/W	Group 1 Compare Match Select 1 and 0
2	G1CMS0	1	R/W	These bits select output trigger of pulse output group 1. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
1	G0CMS1	1	R/W	Group 0 Compare Match Select 1 and 0
0	G0CMS0	1	R/W	These bits select output trigger of pulse output group 0. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3

### 13.3.5 PPG Output Mode Register (PMR)

PMR selects the pulse output mode of the PPG for each group. If inverted output is selected, a low-level pulse is output when PODRH is 1 and a high-level pulse is output when PODRH is 0. If non-overlapping operation is selected, PPG updates its output values at compare match A or B of the TPU that becomes the output trigger. For details, refer to section 13.4.4, Non-Overlapping Pulse Output.

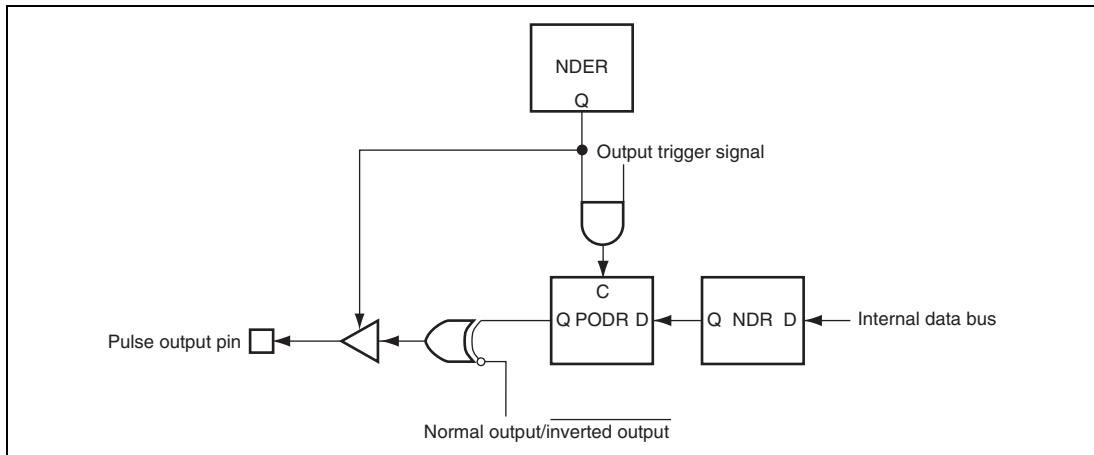
Bit	7	6	5	4	3	2	1	0
Bit Name	G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV
Initial Value	1	1	1	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	G3INV	1	R/W	Group 3 Inversion Selects direct output or inverted output for pulse output group 3. 0: Inverted output 1: Direct output
6	G2INV	1	R/W	Group 2 Inversion Selects direct output or inverted output for pulse output group 2. 0: Inverted output 1: Direct output
5	G1INV	1	R/W	Group 1 Inversion Selects direct output or inverted output for pulse output group 1. 0: Inverted output 1: Direct output
4	G0INV	1	R/W	Group 0 Inversion Selects direct output or inverted output for pulse output group 0. 0: Inverted output 1: Direct output

Bit	Bit Name	Initial Value	R/W	Description
3	G3NOV	0	R/W	<p>Group 3 Non-Overlap</p> <p>Selects normal or non-overlapping operation for pulse output group 3.</p> <p>0: Normal operation (output values updated at compare match A in the selected TPU channel)</p> <p>1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)</p>
2	G2NOV	0	R/W	<p>Group 2 Non-Overlap</p> <p>Selects normal or non-overlapping operation for pulse output group 2.</p> <p>0: Normal operation (output values updated at compare match A in the selected TPU channel)</p> <p>1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)</p>
1	G1NOV	0	R/W	<p>Group 1 Non-Overlap</p> <p>Selects normal or non-overlapping operation for pulse output group 1.</p> <p>0: Normal operation (output values updated at compare match A in the selected TPU channel)</p> <p>1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)</p>
0	G0NOV	0	R/W	<p>Group 0 Non-Overlap</p> <p>Selects normal or non-overlapping operation for pulse output group 0.</p> <p>0: Normal operation (output values updated at compare match A in the selected TPU channel)</p> <p>1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)</p>

## 13.4 Operation

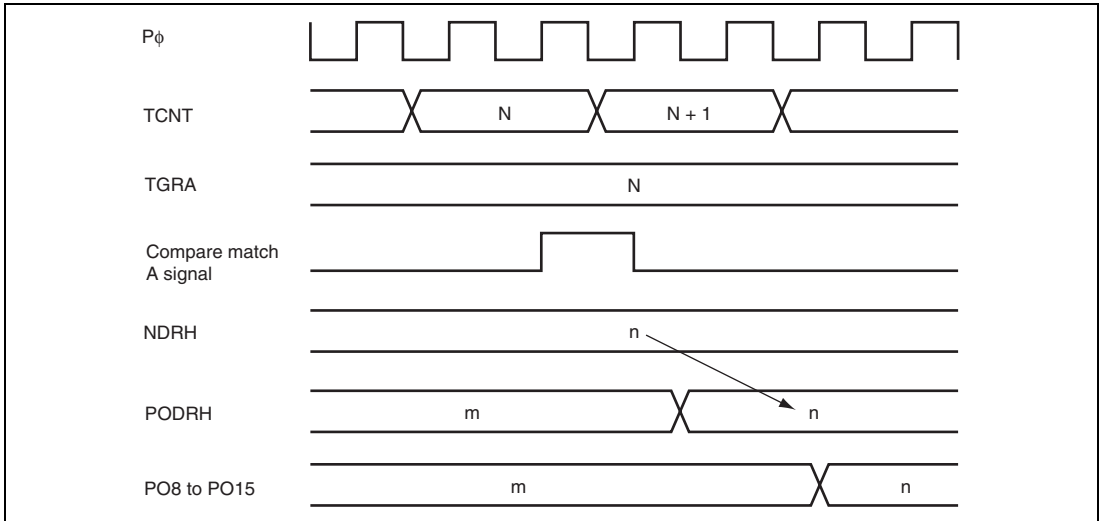
Figure 13.2 shows a schematic diagram of the PPG. PPG pulse output is enabled when the corresponding bits in NDER are set to 1. An initial output value is determined by its corresponding PODR initial setting. When the compare match event specified by PCR occurs, the corresponding NDR bit contents are transferred to PODR to update the output values. Sequential output of data of up to 16 bits is possible by writing new output data to NDR before the next compare match.



**Figure 13.2 Schematic Diagram of PPG**

### 13.4.1 Output Timing

If pulse output is enabled, the NDR contents are transferred to PODR and output when the specified compare match event occurs. Figure 13.3 shows the timing of these operations for the case of normal output in groups 2 and 3, triggered by compare match A.

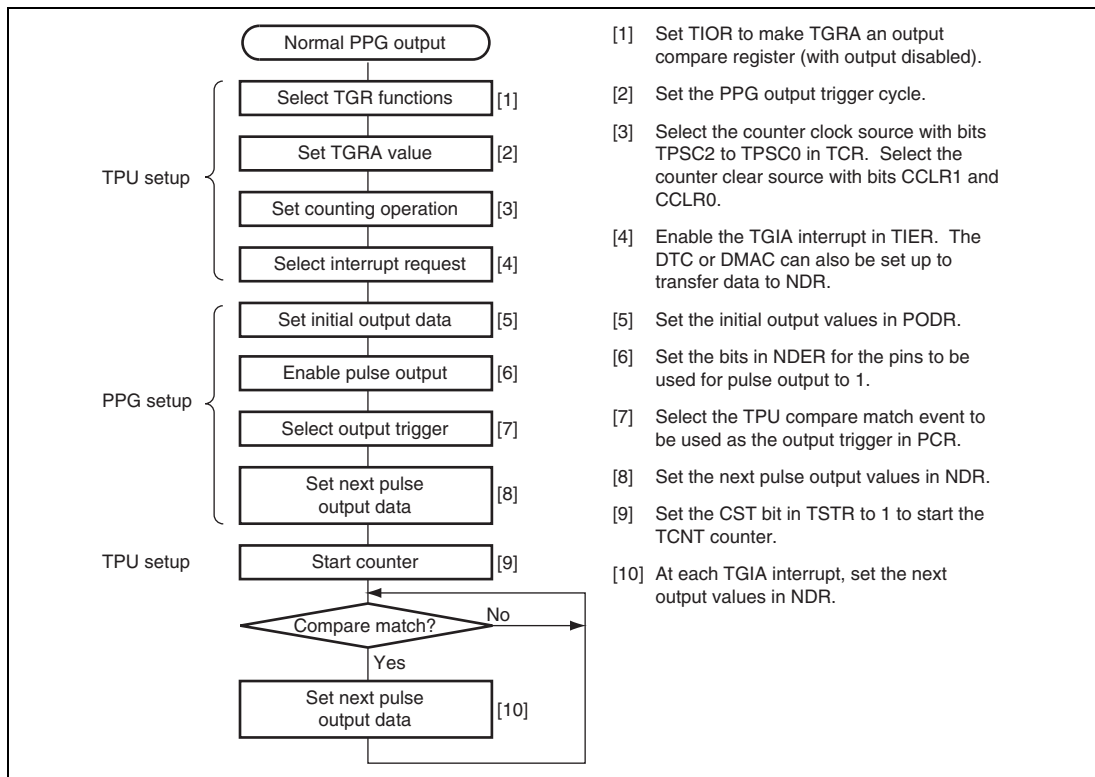


**Figure 13.3 Timing of Transfer and Output of NDR Contents (Example)**



### 13.4.2 Sample Setup Procedure for Normal Pulse Output

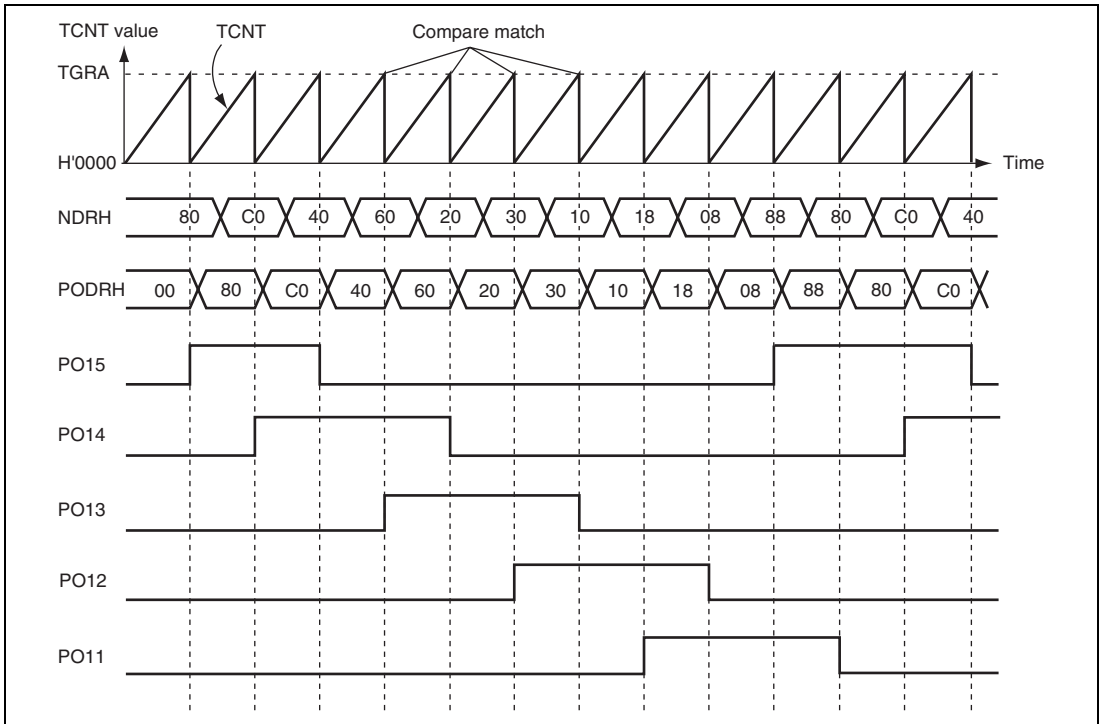
Figure 13.4 shows a sample procedure for setting up normal pulse output.



**Figure 13.4 Setup Procedure for Normal Pulse Output (Example)**

### 13.4.3 Example of Normal Pulse Output (Example of 5-Phase Pulse Output)

Figure 13.5 shows an example in which pulse output is used for cyclic 5-phase pulse output.



**Figure 13.5 Normal Pulse Output Example (5-Phase Pulse Output)**

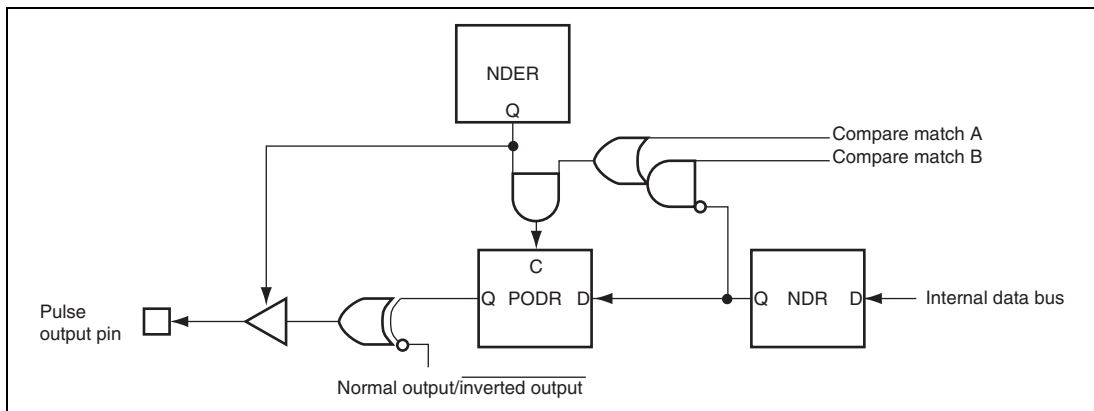
1. Set up TGRA in TPU which is used as the output trigger to be an output compare register. Set a cycle in TGRA so the counter will be cleared by compare match A. Set the TGIEA bit in TIER to 1 to enable the compare match/input capture A (TGIA) interrupt.
2. Write H'F8 to NDERH, and set bits G3CMS1, G3CMS0, G2CMS1, and G2CMS0 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. 5-phase 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.
5. If the DTC or DMAC is set for activation by the TGIA interrupt, pulse output can be obtained without imposing a load on the CPU.

### 13.4.4 Non-Overlapping Pulse Output

During non-overlapping operation, transfer from NDR to PODR is performed as follows:

- At compare match A, the NDR bits are always transferred to PODR.
- At compare match B, the NDR bits are transferred only if their value is 0. The NDR bits are not transferred if their value is 1.

Figure 13.6 illustrates the non-overlapping pulse output operation.



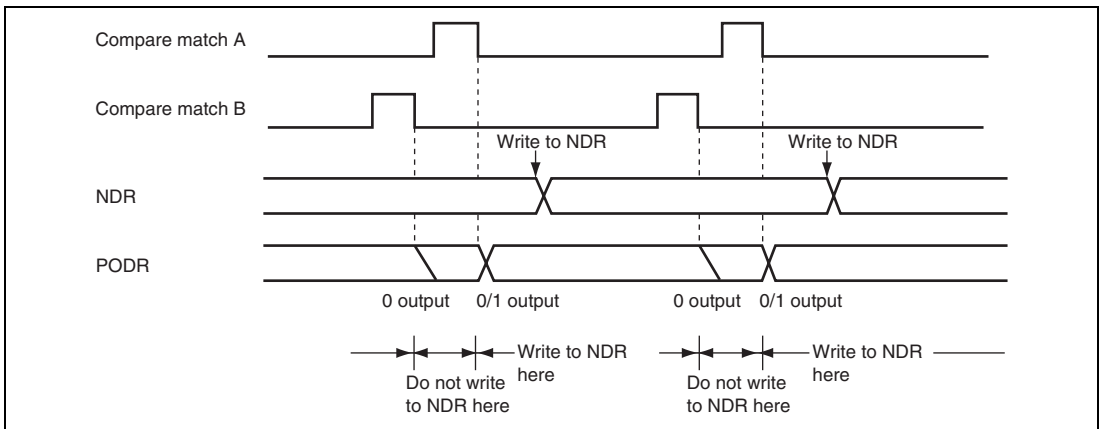
**Figure 13.6 Non-Overlapping Pulse Output**

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-overlapping 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.

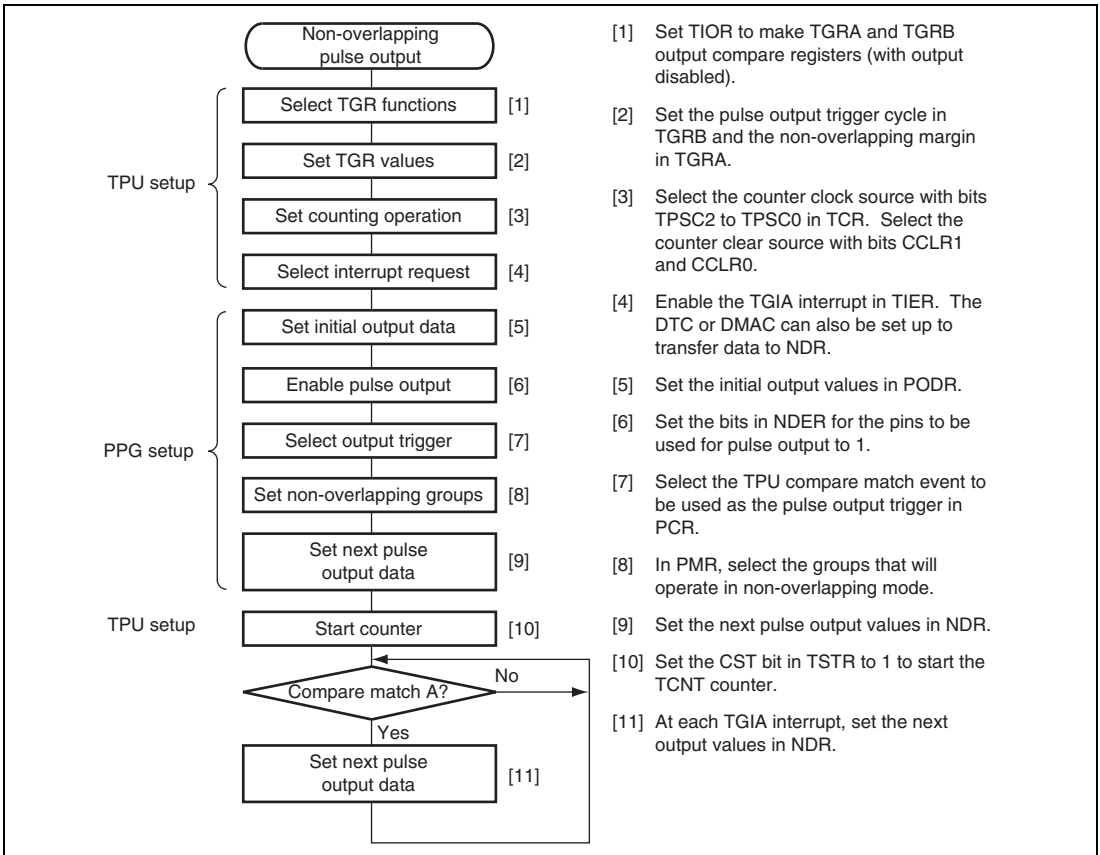
Figure 13.7 shows the timing of this operation.



**Figure 13.7 Non-Overlapping Operation and NDR Write Timing**

### 13.4.5 Sample Setup Procedure for Non-Overlapping Pulse Output

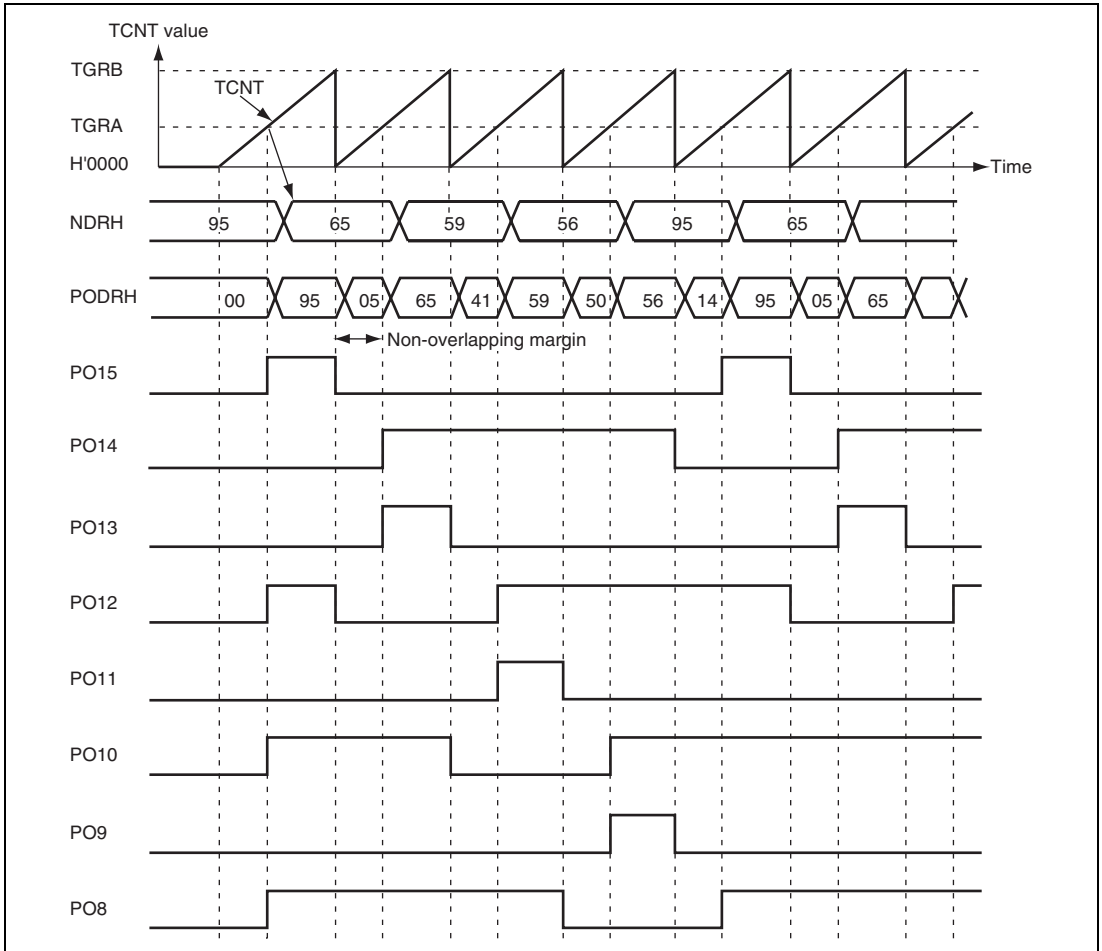
Figure 13.8 shows a sample procedure for setting up non-overlapping pulse output.



**Figure 13.8 Setup Procedure for Non-Overlapping Pulse Output (Example)**

### 13.4.6 Example of Non-Overlapping Pulse Output (Example of 4-Phase Complementary Non-Overlapping Pulse Output)

Figure 13.9 shows an example in which pulse output is used for 4-phase complementary non-overlapping pulse output.



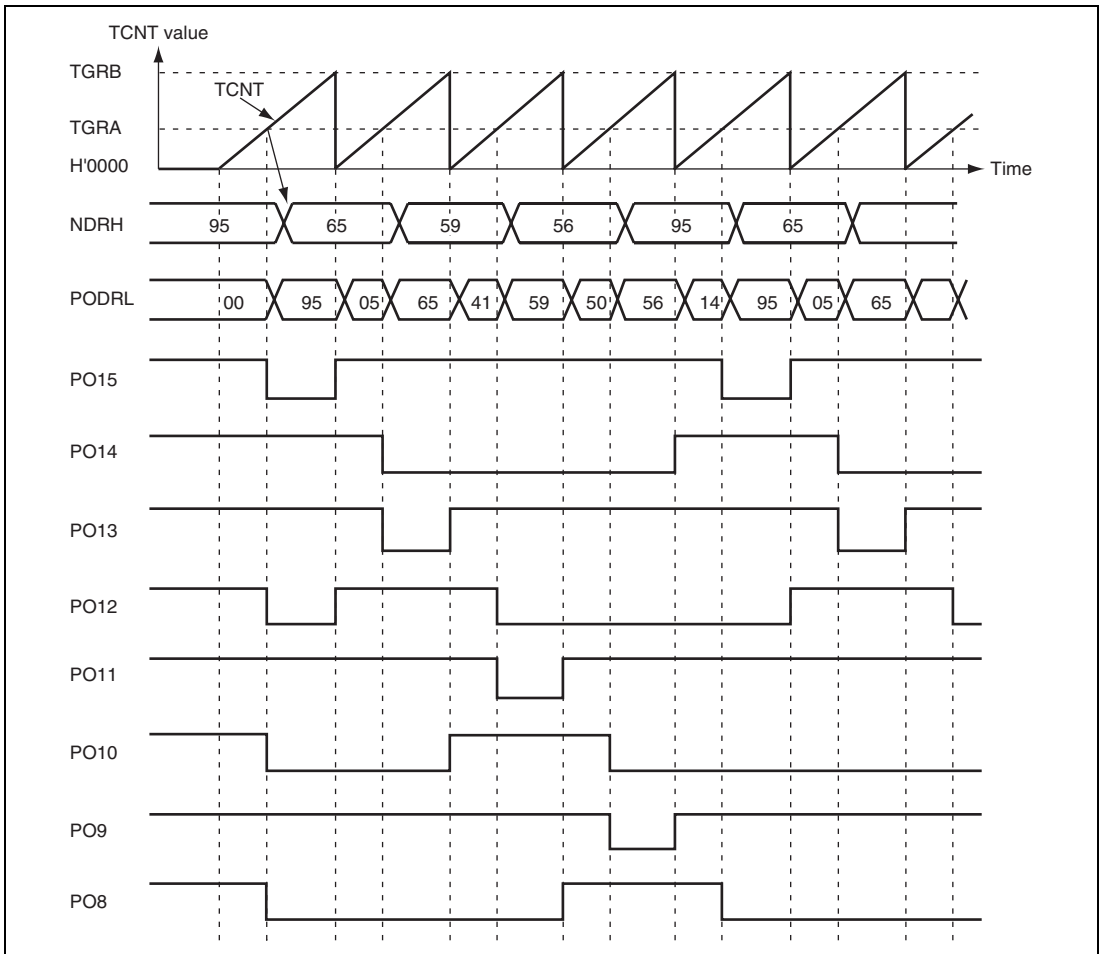
**Figure 13.9 Non-Overlapping Pulse Output Example (4-Phase Complementary)**

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 cycle in TGRB and the non-overlapping 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 to NDERH, and set bits G3CMS1, G3CMS0, G2CMS1, and G2CMS0 in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Set bits G3NOV and G2NOV in PMR to 1 to select non-overlapping pulse output. Write output data H'95 to 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) to NDRH.
4. 4-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 a TGIA interrupt, pulse can be output without imposing a load on the CPU.

### 13.4.7 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 13.10 shows the outputs when the G3INV and G2INV bits are cleared to 0, in addition to the settings of figure 13.9.



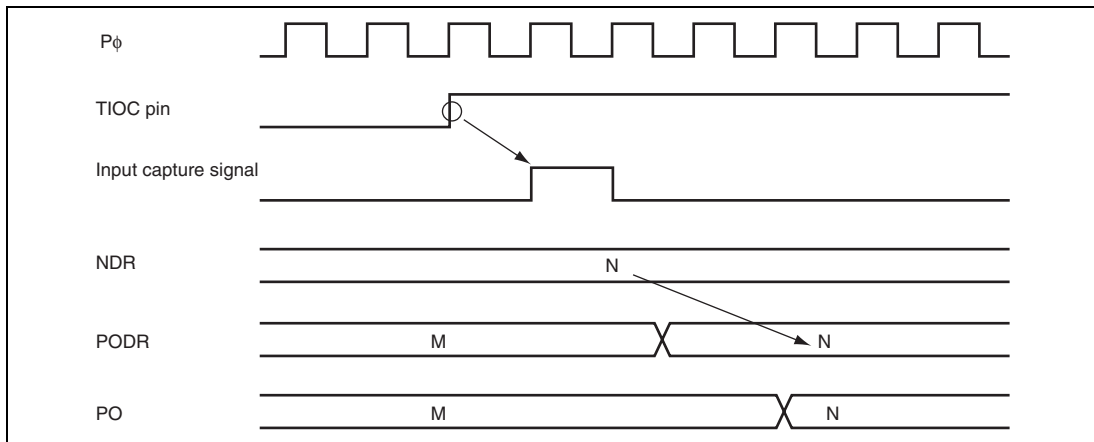
**Figure 13.10 Inverted Pulse Output (Example)**



### 13.4.8 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 13.11 shows the timing of this output.



**Figure 13.11 Pulse Output Triggered by Input Capture (Example)**

## 13.5 Usage Notes

### 13.5.1 Module Stop Function Setting

PPG operation can be disabled or enabled using the module stop control register. The initial value is for PPG operation to be halted. Register access is enabled by clearing module stop state. For details, refer to section 24, Power-Down Modes.

### 13.5.2 Operation of Pulse Output Pins

Pins PO0 to PO15 are also used for other peripheral functions such as the TPU. When output by another peripheral function is enabled, the corresponding pins cannot be used for pulse output. Note, however, that data transfer from NDR bits to PODR bits takes place, regardless of the usage of the pins.

Pin functions should be changed only under conditions in which the output trigger event will not occur.

## Section 14 8-Bit Timers (TMR)

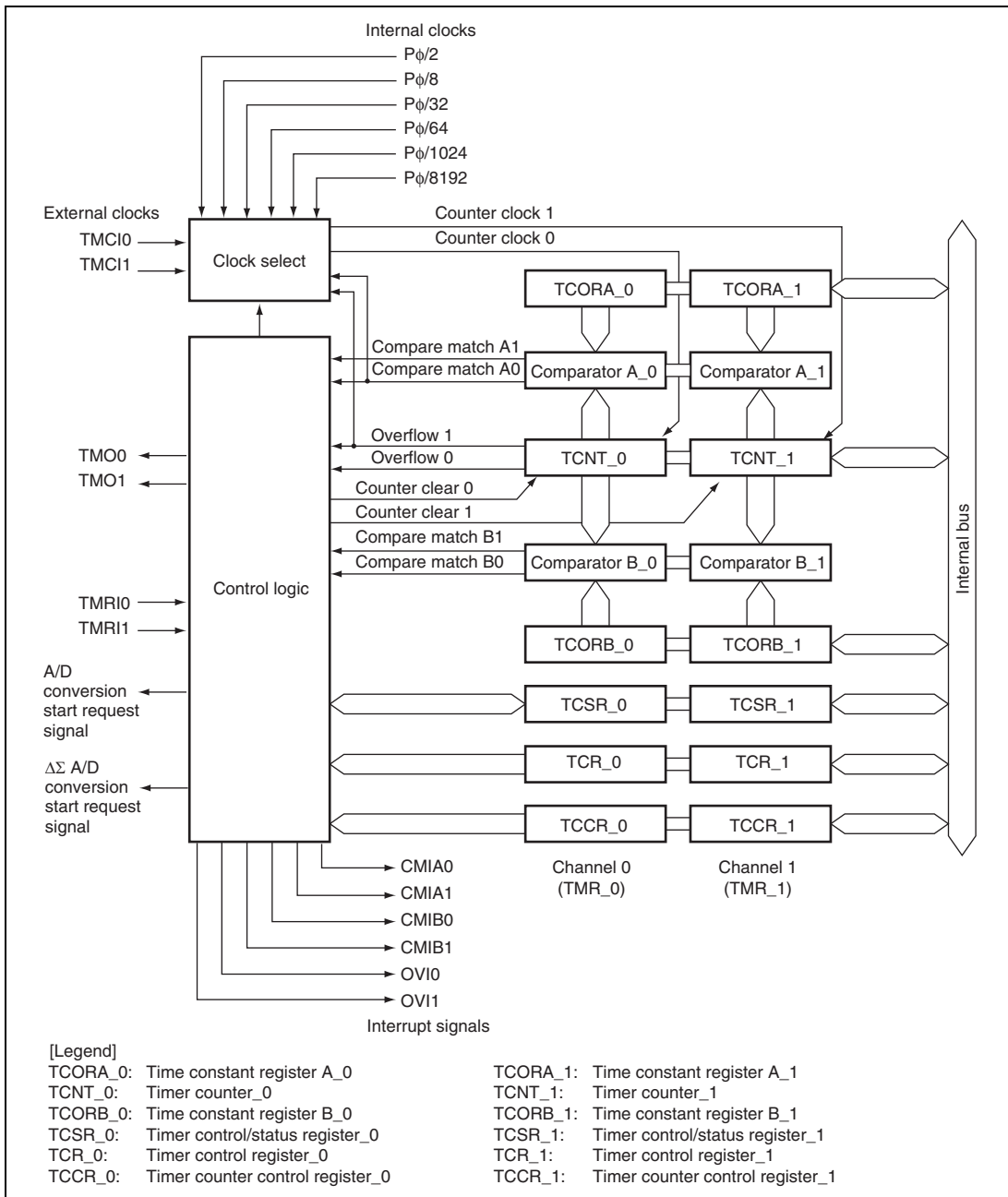
This LSI has four units (unit 0 to unit 3) of an on-chip 8-bit timer module that comprise two 8-bit counter channels, totaling eight channels. The 8-bit timer module can be used to count external events and also be used as a multifunction timer in a variety of applications, such as generation of counter reset, interrupt requests, and pulse output with a desired duty cycle using a compare-match signal with two registers.

Figures 14.1 to 14.4 show block diagrams of the 8-bit timer module (unit 0 to unit 3).

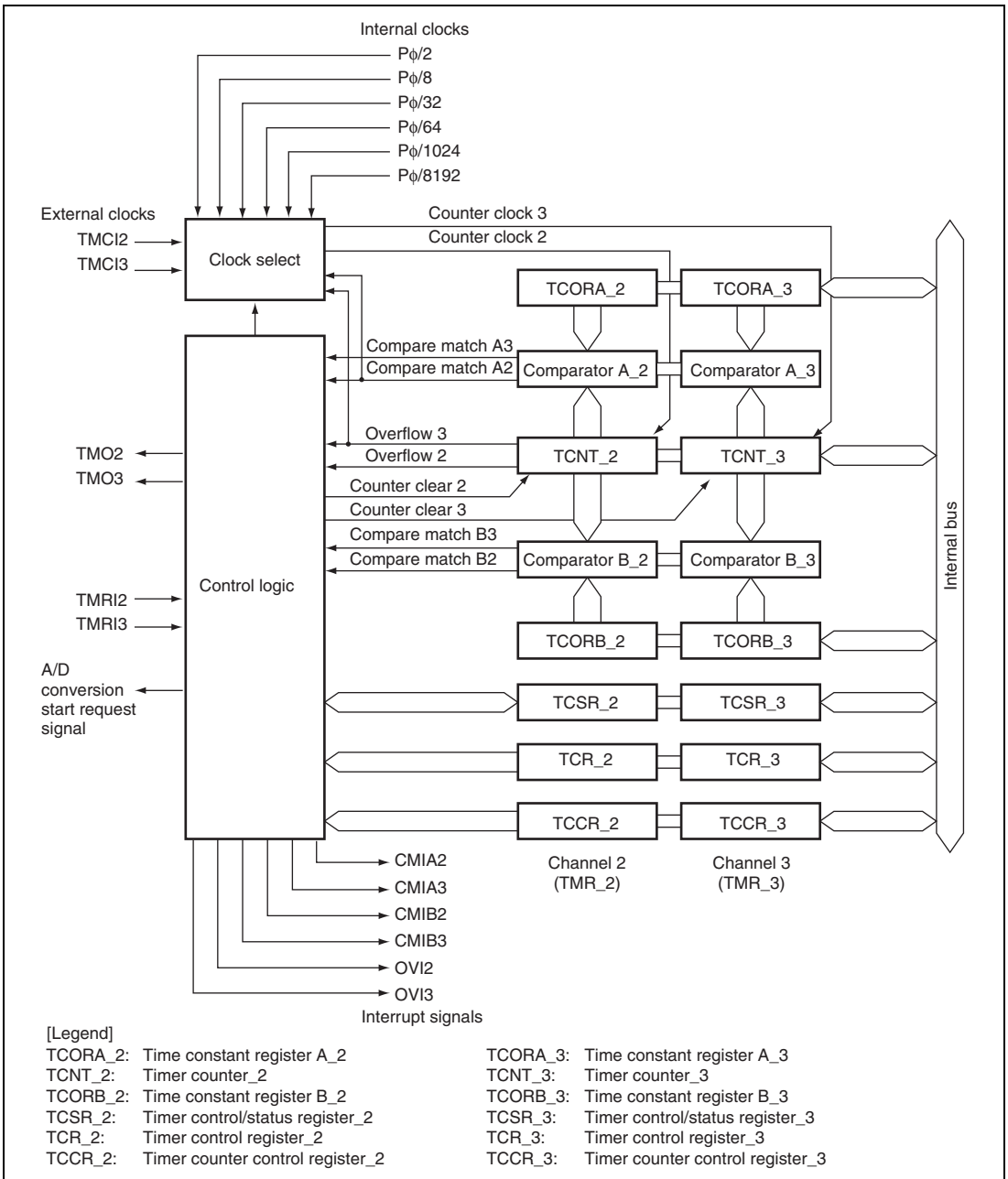
This section describes unit 0 (channels 0 and 1) and unit 2 (channels 4 and 5). Unit 0 and unit 1 have the same functions. Unit 2 and unit 3 have the same functions as unit 0 and unit 1, except that units 2 and 3 do not have the TMRI and TMCI pins.

### 14.1 Features

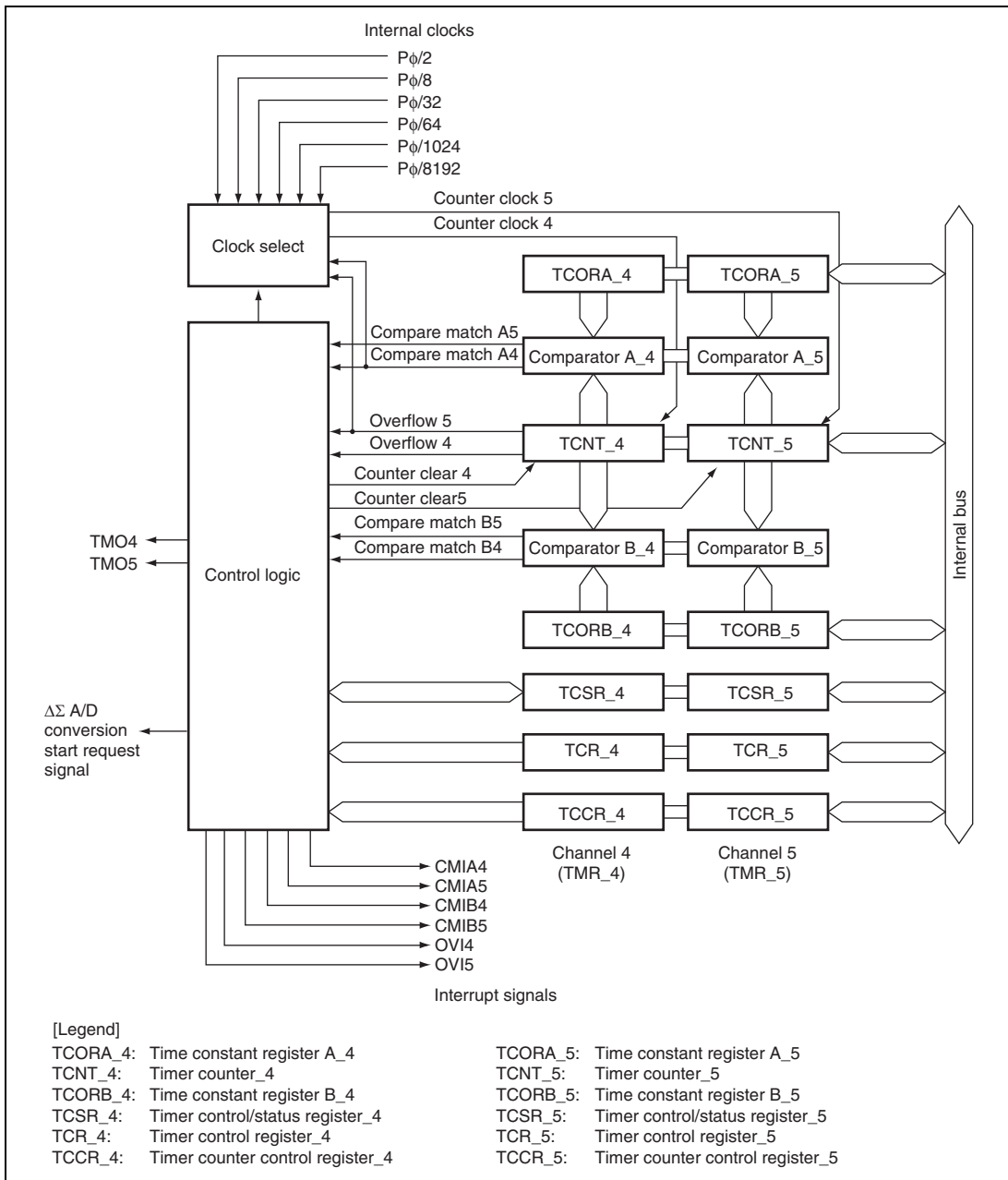
- Selection of seven clock sources  
The counters can be driven by one of six internal clock signals (P $\phi$ /2, P $\phi$ /8, P $\phi$ /32, P $\phi$ /64, P $\phi$ /1024, or P $\phi$ /8192) or an external clock input (only internal clock available in units 2 and 3: P $\phi$ /2, P $\phi$ /8, P $\phi$ /32, P $\phi$ /64, P $\phi$ /1024, and P $\phi$ /8192).
- 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. (This is available only in unit 0 and unit 1.)
- 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 output pulses with a desired duty cycle or PWM output.
- Cascading of two channels  
Operation as a 16-bit timer is possible, using TMR\_0 for the upper 8 bits and TMR\_1 for the lower 8 bits (16-bit count mode).  
TMR\_1 can be used to count TMR\_0 compare matches (compare match count mode).
- Three interrupt sources  
Compare match A, compare match B, and overflow interrupts can be requested independently.
- Generation of trigger to start A/D converter conversion (available in unit 0 and unit 1 only)
- Generation of trigger to start  $\Delta\Sigma$  A/D converter conversion (available in unit 0 and unit 2 only)
- Module stop state specifiable



**Figure 14.1 Block Diagram of 8-Bit Timer Module (Unit 0)**



**Figure 14.2 Block Diagram of 8-Bit Timer Module (Unit 1)**



**Figure 14.3 Block Diagram of 8-Bit Timer Module (Unit 2)**

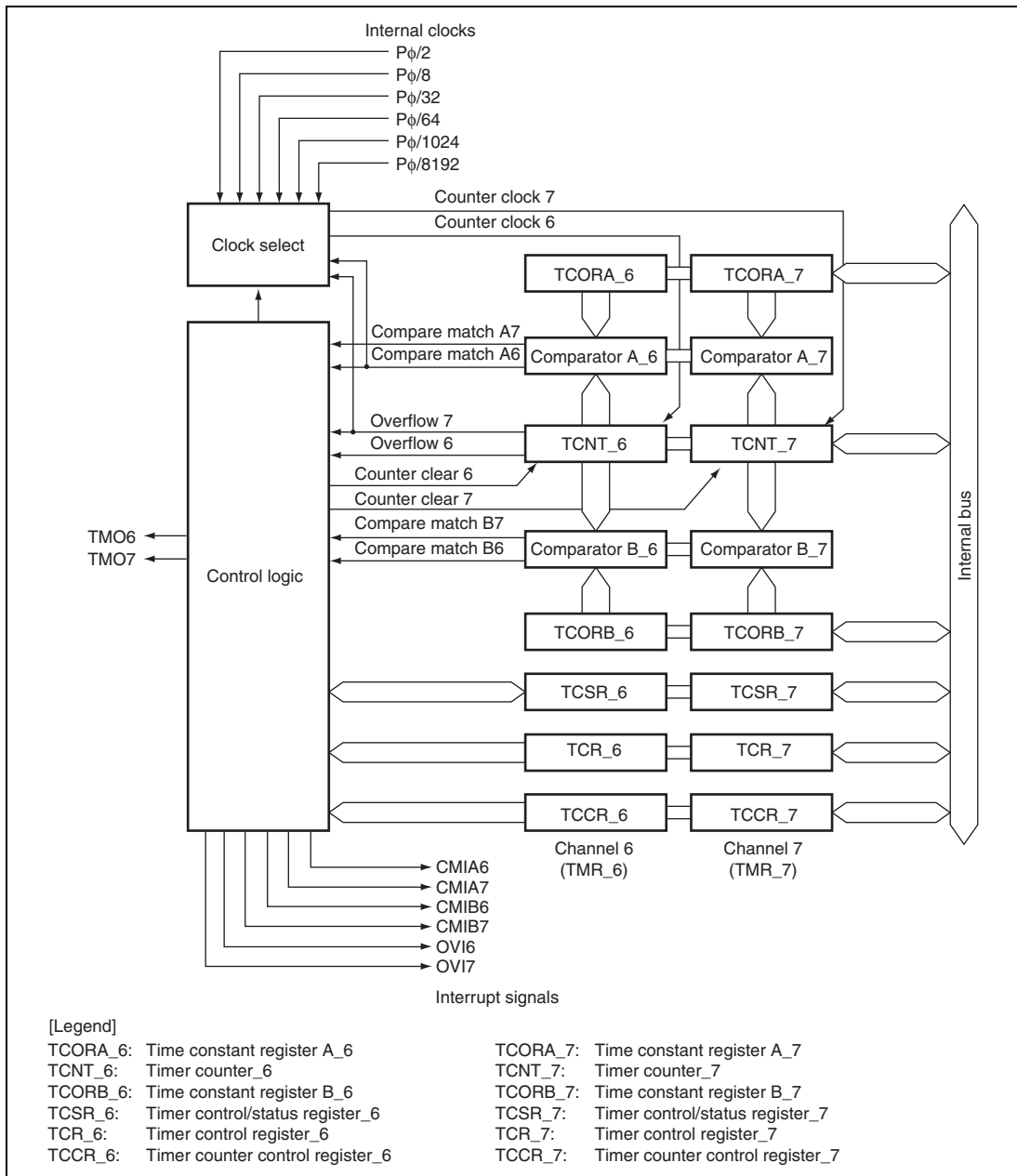


Figure 14.4 Block Diagram of 8-Bit Timer Module (Unit 3)

## 14.2 Input/Output Pins

Table 14.1 shows the pin configuration of the TMR.

**Table 14.1 Pin Configuration**

Unit	Channel	Name	Symbol	I/O	Function
0	0	Timer output pin	TMO0	Output	Outputs compare match
		Timer clock input pin	TMC10	Input	Inputs external clock for counter
		Timer reset input pin	TMRI0	Input	Inputs external reset to counter
	1	Timer output pin	TMO1	Output	Outputs compare match
		Timer clock input pin	TMC11	Input	Inputs external clock for counter
		Timer reset input pin	TMRI1	Input	Inputs external reset to counter
1	2	Timer output pin	TMO2	Output	Outputs compare match
		Timer clock input pin	TMC12	Input	Inputs external clock for counter
		Timer reset input pin	TMRI2	Input	Inputs external reset to counter
	3	Timer output pin	TMO3	Output	Outputs compare match
		Timer clock input pin	TMC13	Input	Inputs external clock for counter
		Timer reset input pin	TMRI3	Input	Inputs external reset to counter
2	4	Timer output pin	TMO4	Output	Outputs compare match
	5	Timer output pin	TMO5	Output	Outputs compare match
3	6	Timer output pin	TMO6	Output	Outputs compare match
	7	Timer output pin	TMO7	Output	Outputs compare match



## 14.3 Register Descriptions

The TMR has the following registers.

### Unit 0:

- Channel 0 (TMR\_0):
  - Timer counter\_0 (TCNT\_0)
  - Time constant register A\_0 (TCORA\_0)
  - Time constant register B\_0 (TCORB\_0)
  - Timer control register\_0 (TCR\_0)
  - Timer counter control register\_0 (TCCR\_0)
  - Timer control/status register\_0 (TCSR\_0)
- Channel 1 (TMR\_1):
  - Timer counter\_1 (TCNT\_1)
  - Time constant register A\_1 (TCORA\_1)
  - Time constant register B\_1 (TCORB\_1)
  - Timer control register\_1 (TCR\_1)
  - Timer counter control register\_1 (TCCR\_1)
  - Timer control/status register\_1 (TCSR\_1)

### Unit 1:

- Channel 2 (TMR\_2):
  - Timer counter\_2 (TCNT\_2)
  - Time constant register A\_2 (TCORA\_2)
  - Time constant register B\_2 (TCORB\_2)
  - Timer control register\_2 (TCR\_2)
  - Timer counter control register\_2 (TCCR\_2)
  - Timer control/status register\_2 (TCSR\_2)
- Channel 3 (TMR\_3):
  - Timer counter\_3 (TCNT\_3)
  - Time constant register A\_3 (TCORA\_3)
  - Time constant register B\_3 (TCORB\_3)
  - Timer control register\_3 (TCR\_3)
  - Timer counter control register\_3 (TCCR\_3)
  - Timer control/status register\_3 (TCSR\_3)

**Unit 2:**

- Channel 4 (TMR\_4):
  - Timer counter\_4 (TCNT\_4)
  - Time constant register A\_4 (TCORA\_4)
  - Time constant register B\_4 (TCORB\_4)
  - Timer control register\_4 (TCR\_4)
  - Timer counter control register\_4 (TCCR\_4)
  - Timer control/status register\_4 (TCSR\_4)
- Channel 5 (TMR\_5):
  - Timer counter\_5 (TCNT\_5)
  - Time constant register A\_5 (TCORA\_5)
  - Time constant register B\_5 (TCORB\_5)
  - Timer control register\_5 (TCR\_5)
  - Timer counter control register\_5 (TCCR\_5)
  - Timer control/status register\_5 (TCSR\_5)

**Unit 3:**

- Channel 6 (TMR\_6):
  - Timer counter\_6 (TCNT\_6)
  - Time constant register A\_6 (TCORA\_6)
  - Time constant register B\_6 (TCORB\_6)
  - Timer control register\_6 (TCR\_6)
  - Timer counter control register\_6 (TCCR\_6)
  - Timer control/status register\_6 (TCSR\_6)
- Channel 7 (TMR\_7):
  - Timer counter\_7 (TCNT\_7)
  - Time constant register A\_7 (TCORA\_7)
  - Time constant register B\_7 (TCORB\_7)
  - Timer control register\_7 (TCR\_7)
  - Timer counter control register\_7 (TCCR\_7)
  - Timer control/status register\_7 (TCSR\_7)

### 14.3.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT\_0 and TCNT\_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction. Bits CKS2 to CKS0 in TCR and bits ICKS1 and ICKS0 in TCCR are used to select a clock. TCNT can be cleared by an external reset input signal, compare match A signal, or compare match B signal. Which signal to be used for clearing is selected by bits CCLR1 and CCLR0 in TCR. When TCNT overflows from H'FF to H'00, bit OVF in TCSR is set to 1. TCNT is initialized to H'00.

Bit	TCNT_0								TCNT_1							
Bit Name	7	6	5	4	3	2	1	0	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

### 14.3.2 Time Constant Register A (TCORA)

TCORA is an 8-bit readable/writable register. TCORA\_0 and TCORA\_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction. The value in TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding CMFA flag in TCSR is set to 1. Note however that comparison is disabled during the T2 state of a TCORA write cycle. The timer output from the TMO pin can be freely controlled by this compare match signal (compare match A) and the settings of bits OS1 and OS0 in TCSR. TCORA is initialized to H'FF.

Bit	TCORA_0								TCORA_1							
Bit Name	7	6	5	4	3	2	1	0	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

### 14.3.3 Time Constant Register B (TCORB)

TCORB is an 8-bit readable/writable register. TCORB\_0 and TCORB\_1 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 to 1. Note however that comparison is disabled during the T2 state of a TCORB write cycle. The timer output from the TMO pin can be freely controlled by this compare match signal (compare match B) and the settings of bits OS3 and OS2 in TCSR. TCORB is initialized to H'FF.

Bit	TCORB_0								TCORB_1							
Bit Name	7	6	5	4	3	2	1	0	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

### 14.3.4 Timer Control Register (TCR)

TCR selects the TCNT clock source and the condition for clearing TCNT, and enables/disables interrupt requests.

Bit	7	6	5	4	3	2	1	0
Bit Name	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

Bit	Bit Name	Initial Value	R/W	Description
7	CMIEB	0	R/W	Compare Match Interrupt Enable B Selects whether CMFB interrupt requests (CMIB) are enabled or disabled when the CMFB flag in TCSR is set to 1.* <sup>2</sup> 0: CMFB interrupt requests (CMIB) are disabled 1: CMFB interrupt requests (CMIB) are enabled

Bit	Bit Name	Initial Value	R/W	Description
6	CMIEA	0	R/W	Compare Match Interrupt Enable A Selects whether CMFA interrupt requests (CMIA) are enabled or disabled when the CMFA flag in TCSR is set to 1. * <sup>2</sup> 0: CMFA interrupt requests (CMIA) are disabled 1: CMFA interrupt requests (CMIA) are enabled
5	OVIE	0	R/W	Timer Overflow Interrupt Enable* <sup>3</sup> Selects whether OVF interrupt requests (OVI) are enabled or disabled when the OVF flag in TCSR is set to 1. 0: OVF interrupt requests (OVI) are disabled 1: OVF interrupt requests (OVI) are enabled
4	CCLR1	0	R/W	Counter Clear 1 and 0* <sup>1</sup>
3	CCLR0	0	R/W	These bits select the method by which TCNT is cleared. 00: Clearing is disabled 01: Cleared by compare match A 10: Cleared by compare match B 11: Cleared at rising edge (TMRIS in TCCR is cleared to 0) of the external reset input or when the external reset input is high (TMRIS in TCCR is set to 1) * <sup>3</sup>
2	CKS2	0	R/W	Clock Select 2 to 0* <sup>1</sup>
1	CKS1	0	R/W	These bits select the clock input to TCNT and count condition. See table 14.2.
0	CKS0	0	R/W	

- Notes:
1. To use an external reset or external clock, the DDR and ICR bits in the corresponding pin should be set to 0 and 1, respectively. For details, see section 11, I/O Ports.
  2. In unit 2 and unit 3, one interrupt signal is used for CMIEB or CMIEA. For details, see section 14.7, Interrupt Sources.
  3. Available only in unit 0 and unit 1.

### 14.3.5 Timer Counter Control Register (TCCR)

TCCR selects the TCNT internal clock source and controls external reset input.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	TMRIS	—	ICKS1	ICKS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R	Reserved These bits are always read as 0. It should not be set to 0.
3	TMRIS	0	R/W	Timer Reset Input Select* Selects an external reset input when the CCLR1 and CCLR0 bits in TCR are B'11. 0: Cleared at rising edge of the external reset 1: Cleared when the external reset is high
2	—	0	R	Reserved This bit is always read as 0. It should not be set to 0.
1	ICKS1	0	R/W	Internal Clock Select 1 and 0
0	ICKS0	0	R/W	These bits in combination with bits CKS2 to CKS0 in TCR select the internal clock. See table 14.2.

Note: \* Available only in unit 0 and unit 1. The write value should always be 0 in unit 2 and unit 3.

**Table 14.2 Clock Input to TCNT and Count Condition (Unit 0)**

Channel	TCR			TCCR		Description	
	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0		
TMR_0	0	0	0	—	—	Clock input prohibited	
	0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /2.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /2.	
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.	
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.	
	1	0	0	—	—	Counts at TCNT_1 overflow signal* <sup>1</sup> .	
	TMR_1	0	0	0	—	—	Clock input prohibited
		0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.
0					1	Uses internal clock. Counts at rising edge of P $\phi$ /2.	
1					0	Uses internal clock. Counts at falling edge of P $\phi$ /8.	
1					1	Uses internal clock. Counts at falling edge of P $\phi$ /2.	
0		1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.	
0		1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.	
1		0	0	—	—	Counts at TCNT_0 compare match A* <sup>1</sup> .	
All		1	0	1	—	—	Uses external clock. Counts at rising edge* <sup>2</sup> .
		1	1	0	—	—	Uses external clock. Counts at falling edge* <sup>2</sup> .
	1	1	1	—	—	Uses external clock. Counts at both rising and falling edges* <sup>2</sup> .	

Notes: 1. If the clock input of channel 0 is the TCNT\_1 overflow signal and that of channel 1 is the TCNT\_0 compare match signal, no incrementing clock is generated. Do not use this setting.

2. To use the external clock, the DDR and ICR bits in the corresponding pin should be set to 0 and 1, respectively. For details, see section 11, I/O Ports.

**Table 14.3 Clock Input to TCNT and Count Condition (Unit 1)**

Channel	TCR			TCCR		Description	
	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0		
TMR_2	0	0	0	—	—	Clock input prohibited	
	0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /2.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /2.	
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.	
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.	
	1	0	0	—	—	Counts at TCNT_3 overflow signal* <sup>1</sup> .	
	TMR_3	0	0	0	—	—	Clock input prohibited
		0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.
0					1	Uses internal clock. Counts at rising edge of P $\phi$ /2.	
1					0	Uses internal clock. Counts at falling edge of P $\phi$ /8.	
1					1	Uses internal clock. Counts at falling edge of P $\phi$ /2.	
0		1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.	
0		1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.	
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.	
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.	
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.	
1		0	0	—	—	Counts at TCNT_2 compare match A* <sup>1</sup> .	
All		1	0	1	—	—	Uses external clock. Counts at rising edge* <sup>2</sup> .
		1	1	0	—	—	Uses external clock. Counts at falling edge* <sup>2</sup> .
	1	1	1	—	—	Uses external clock. Counts at both rising and falling edges* <sup>2</sup> .	

- Notes: 1. If the clock input of channel 2 is the TCNT\_3 overflow signal and that of channel 3 is the TCNT\_2 compare match signal, no incrementing clock is generated. Do not use this setting.
2. To use the external clock, the DDR and ICR bits in the corresponding pin should be set to 0 and 1, respectively. For details, see section 11, I/O Ports.



**Table 14.4 Clock Input to TCNT and Count Condition (Unit 2)**

Channel	TCR			TCCR		Description
	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0	
TMR_4	0	0	0	—	—	Clock input prohibited
	0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /2.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /2.
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.
	1	0	0	—	—	Counts at TCNT_5 overflow signal*.
TMR_5	0	0	0	—	—	Clock input prohibited
	0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /2.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /2.
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.
	1	0	0	—	—	Counts at TCNT_4 compare match A*.
All	1	0	1	—	—	Setting prohibited
	1	1	0	—	—	Setting prohibited
	1	1	1	—	—	Setting prohibited

Note: \* If the clock input of channel 4 is the TCNT\_5 overflow signal and that of channel 5 is the TCNT\_4 compare match signal, no incrementing clock is generated. Do not use this setting.

**Table 14.5 Clock Input to TCNT and Count Condition (Unit 3)**

Channel	TCR			TCCR		Description
	Bit 2 CKS2	Bit 1 CKS1	Bit 0 CKS0	Bit 1 ICKS1	Bit 0 ICKS0	
TMR_6	0	0	0	—	—	Clock input prohibited
	0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /2.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /2.
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.
1	0	0	—	—	Counts at TCNT_7 overflow signal*.	
TMR_7	0	0	0	—	—	Clock input prohibited
	0	0	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /2.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /2.
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /64.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /32.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /64.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /32.
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P $\phi$ /8192.
				0	1	Uses internal clock. Counts at rising edge of P $\phi$ /1024.
				1	0	Uses internal clock. Counts at falling edge of P $\phi$ /8192.
				1	1	Uses internal clock. Counts at falling edge of P $\phi$ /1024.
1	0	0	—	—	Counts at TCNT_6 compare match A*.	
All	1	0	1	—	—	Setting prohibited
	1	1	0	—	—	Setting prohibited
	1	1	1	—	—	Setting prohibited

Note: \* If the clock input of channel 6 is the TCNT\_7 overflow signal and that of channel 7 is the TCNT\_6 compare match signal, no incrementing clock is generated. Do not use this setting.

### 14.3.6 Timer Control/Status Register (TCSR)

TCSR displays status flags, and controls compare match output.

- TCSR\_0

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- TCSR\_1

Bit	7	6	5	4	3	2	1	0
Bit Name	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	R/W	R/W	R/W	R/W

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

- TCSR\_0

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)* <sup>1</sup>	Compare Match Flag B [Setting condition] <ul style="list-style-type: none"> <li>• When TCNT matches TCORB</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>• When writing 0 after reading CMFB = 1 (When this flag is cleared by the CPU in the interrupt handling, be sure to read the flag after writing 0 to it.)</li> <li>• When the DTC is activated by a CMIB interrupt while the DISEL bit in MRB of the DTC is 0*<sup>3</sup></li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
6	CMFA	0	R/(W)* <sup>1</sup>	<p>Compare Match Flag A</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When TCNT matches TCORA</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When writing 0 after reading CMFA = 1 (When this flag is cleared by the CPU in the interrupt handling, be sure to read the flag after writing 0 to it.)</li> <li>When the DTC is activated by a CMIA interrupt while the DISEL bit in MRB in the DTC is 0*<sup>3</sup></li> </ul>
5	OVF	0	R/(W)* <sup>1</sup>	<p>Timer Overflow Flag</p> <p>[Setting condition]</p> <p>When TCNT overflows from H'FF to H'00</p> <p>[Clearing condition]</p> <p>When writing 0 after reading OVF = 1 (When this flag is cleared by the CPU in the interrupt handling, be sure to read the flag after writing 0 to it.)</p>
4	ADTE	0	R/W	<p>A/D Trigger Enable*<sup>3</sup></p> <p>Selects enabling or disabling of A/D converter start requests by compare match A.</p> <p>0: A/D converter start requests by compare match A are disabled</p> <p>1: A/D converter start requests by compare match A are enabled</p>
3	OS3	0	R/W	Output Select 3 and 2* <sup>2</sup>
2	OS2	0	R/W	<p>These bits select a method of TMO pin output when compare match B of TCORB and TCNT occurs.</p> <p>00: No change when compare match B occurs</p> <p>01: 0 is output when compare match B occurs</p> <p>10: 1 is output when compare match B occurs</p> <p>11: Output is inverted when compare match B occurs (toggle output)</p>

Bit	Bit Name	Initial Value	R/W	Description
1	OS1	0	R/W	Output Select 1 and 0* <sup>2</sup>
0	OS0	0	R/W	These bits select a method of TMO pin output when compare match A of TCORA and TCNT occurs. 00: No change when compare match A occurs 01: 0 is output when compare match A occurs 10: 1 is output when compare match A occurs 11: Output is inverted when compare match A occurs (toggle output)

- Notes:
1. Only 0 can be written to bits 7 to 5, to clear these flags.
  2. Timer output is disabled when bits OS3 to OS0 are all 0. Timer output is 0 until the first compare match occurs after a reset.
  3. For the corresponding A/D converter channels, see section 18, A/D Converter.

- TCSR\_1

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)* <sup>1</sup>	Compare Match Flag B [Setting condition] <ul style="list-style-type: none"> <li>When TCNT matches TCORB</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>When writing 0 after reading CMFB = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>When the DTC is activated by a CMIB interrupt while the DISEL bit in MRB of the DTC is 0*<sup>3</sup></li> </ul>
6	CMFA	0	R/(W)* <sup>1</sup>	Compare Match Flag A [Setting condition] <ul style="list-style-type: none"> <li>When TCNT matches TCORA</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>When writing 0 after reading CMFA = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>When the DTC is activated by a CMIA interrupt while the DISEL bit in MRB of the DTC is 0*<sup>3</sup></li> </ul>
5	OVF	0	R/(W)* <sup>1</sup>	Timer Overflow Flag [Setting condition] <p>When TCNT overflows from H'FF to H'00</p> [Clearing condition] <p>Cleared by reading OVF when OVF = 1, then writing 0 to OVF</p> (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial Value	R/W	Description
4	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
3	OS3	0	R/W	Output Select 3 and 2* <sup>2</sup>
2	OS2	0	R/W	These bits select a method of TMO pin output when compare match B of TCORB and TCNT occurs. 00: No change when compare match B occurs 01: 0 is output when compare match B occurs 10: 1 is output when compare match B occurs 11: Output is inverted when compare match B occurs (toggle output)
1	OS1	0	R/W	Output Select 1 and 0* <sup>2</sup>
0	OS0	0	R/W	These bits select a method of TMO pin output when compare match A of TCORA and TCNT occurs. 00: No change when compare match A occurs 01: 0 is output when compare match A occurs 10: 1 is output when compare match A occurs 11: Output is inverted when compare match A occurs (toggle output)

- Notes:
1. Only 0 can be written to bits 7 to 5, to clear these flags.
  2. Timer output is disabled when bits OS3 to OS0 are all 0. Timer output is 0 until the first compare match occurs after a reset.
  3. Available only in unit 0 and unit 1.

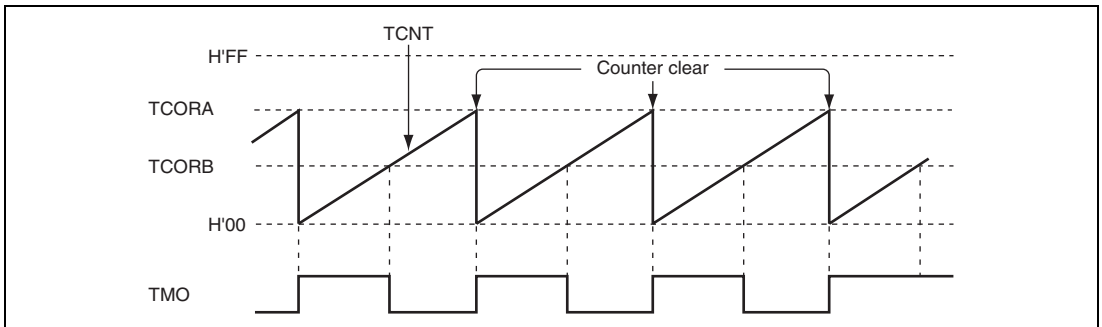
## 14.4 Operation

### 14.4.1 Pulse Output

Figure 14.5 shows an example of the 8-bit timer being used to generate a pulse output with a desired duty cycle. The control bits are set as follows:

1. Clear the bit CCLR1 in TCR to 0 and set the bit CCLR0 in TCR to 1 so that TCNT is cleared at a TCORA compare match.
2. Set the bits OS3 to OS0 in TCSR 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 pulses output at a cycle determined by TCORA with a pulse width determined by TCORB. No software intervention is required. The timer output is 0 until the first compare match occurs after a reset.



**Figure 14.5 Example of Pulse Output**

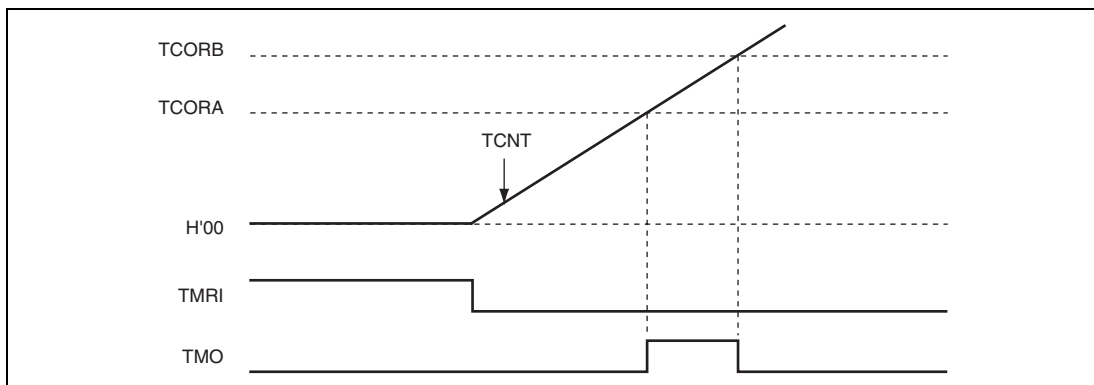


### 14.4.2 Reset Input

Figure 14.6 shows an example of the 8-bit timer being used to generate a pulse which is output after a desired delay time from a TMRI input. The control bits are set as follows:

1. Set both bits CCLR1 and CCLR0 in TCR to 1 and set the TMRIS bit in TCCR to 1 so that TCNT is cleared at the high level input of the TMRI signal.
2. In TCSR, set bits OS3 to OS0 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 pulses output at a desired delay time from a TMRI input determined by TCORA and with a pulse width determined by TCORB and TCORA.

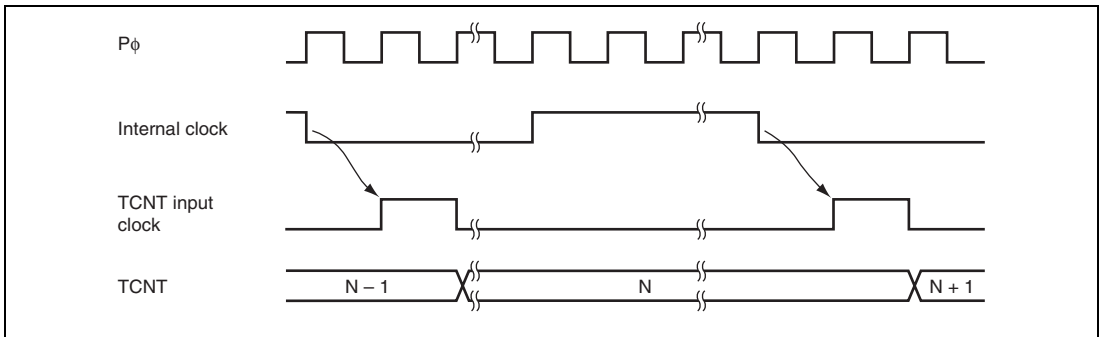


**Figure 14.6 Example of Reset Input**

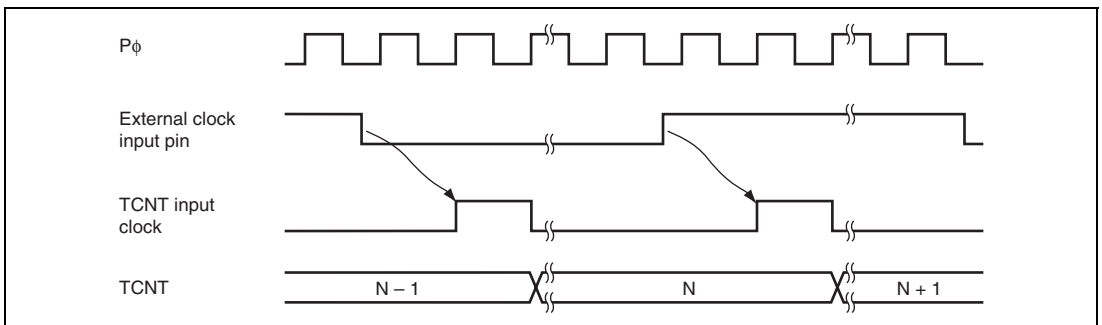
## 14.5 Operation Timing

### 14.5.1 TCNT Count Timing

Figure 14.7 shows the TCNT count timing for internal clock input. Figure 14.8 shows the TCNT count timing for external clock input. Note that the external clock pulse width must be at least 1.5 states for increment at a single edge, and at least 2.5 states for increment at both edges. The counter will not increment correctly if the pulse width is less than these values.



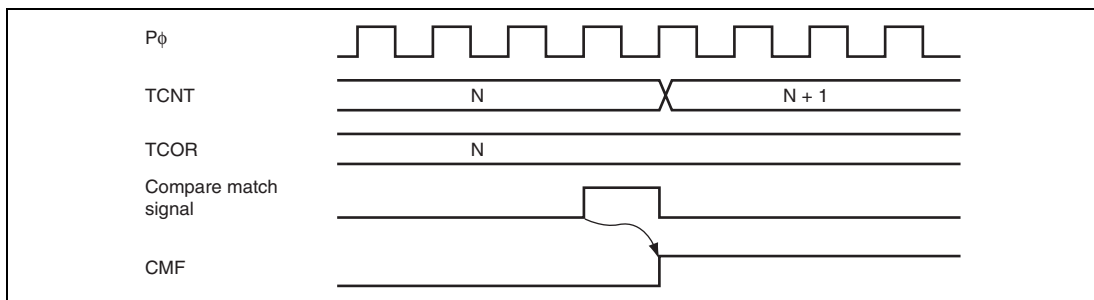
**Figure 14.7** Count Timing for Internal Clock Input



**Figure 14.8** Count Timing for External Clock Input

### 14.5.2 Timing of CMFA and CMFB Setting at Compare Match

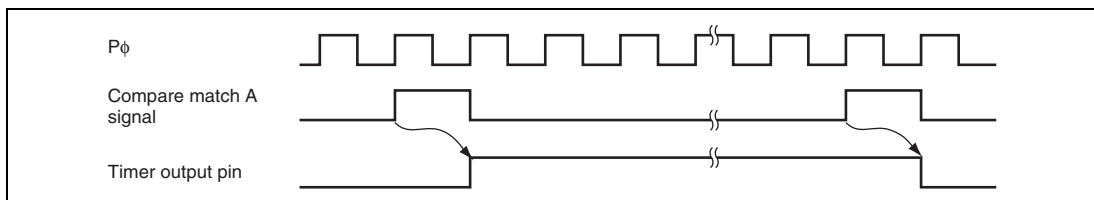
The CMFA and CMFB flags in TCSR are set to 1 by a compare match signal generated when the TCOR and TCNT values match. The compare match signal is generated at the last state in which the match is true, just before the timer counter is updated. Therefore, when the TCOR and TCNT values match, the compare match signal is not generated until the next TCNT clock input. Figure 14.9 shows this timing.



**Figure 14.9 Timing of CMF Setting at Compare Match**

### 14.5.3 Timing of Timer Output at Compare Match

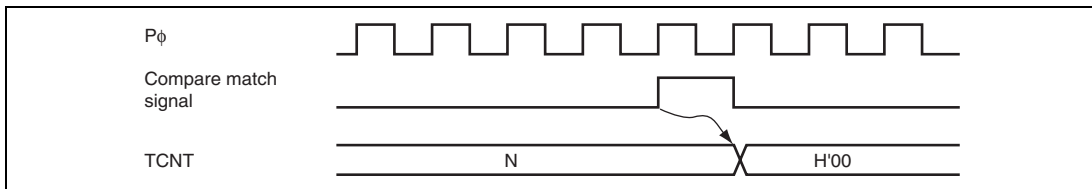
When a compare match signal is generated, the timer output changes as specified by the bits OS3 to OS0 in TCSR. Figure 14.10 shows the timing when the timer output is toggled by the compare match A signal.



**Figure 14.10 Timing of Toggled Timer Output at Compare Match A**

### 14.5.4 Timing of Counter Clear by Compare Match

TCNT is cleared when compare match A or B occurs, depending on the settings of the bits CCLR1 and CCLR0 in TCR. Figure 14.11 shows the timing of this operation.

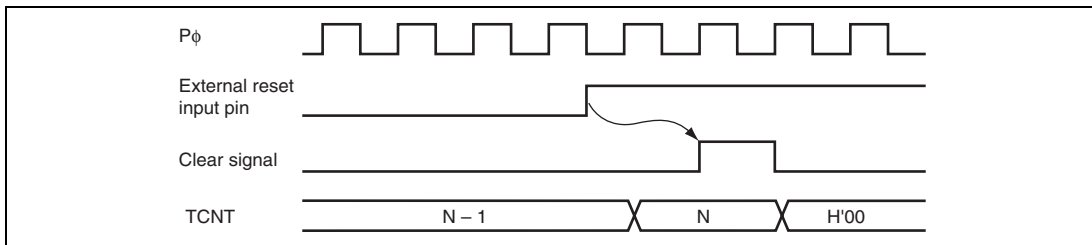


**Figure 14.11 Timing of Counter Clear by Compare Match**

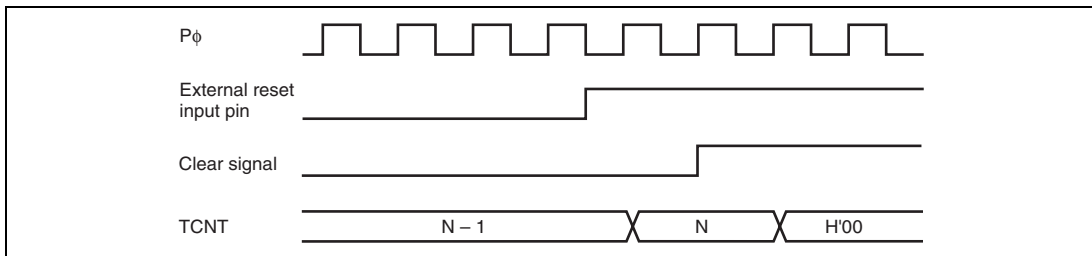
### 14.5.5 Timing of TCNT External Reset\*

TCNT is cleared at the rising edge or high level of an external reset input, depending on the settings of bits CCLR1 and CCLR0 in TCR. The clear pulse width must be at least 2 states. Figure 14.12 and Figure 14.13 shows the timing of this operation.

Note: \* Clearing by an external reset is available only in units 0 and 1.



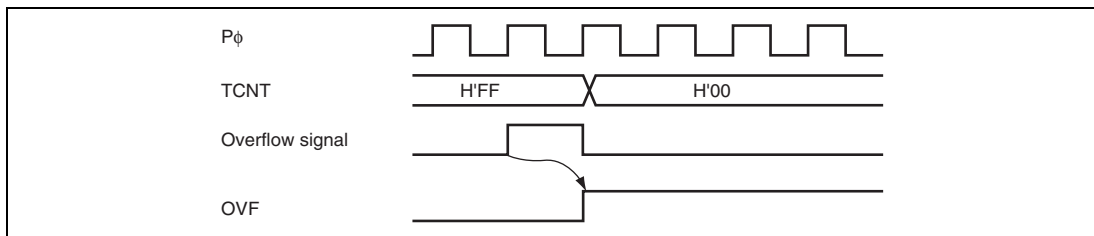
**Figure 14.12 Timing of Clearance by External Reset (Rising Edge)**



**Figure 14.13 Timing of Clearance by External Reset (High Level)**

### 14.5.6 Timing of Overflow Flag (OVF) Setting

The OVF bit in TCSR is set to 1 when TCNT overflows (changes from H'FF to H'00). Figure 14.14 shows the timing of this operation.



**Figure 14.14 Timing of OVF Setting**

## 14.6 Operation with Cascaded Connection

If the bits CKS2 to CKS0 in either TCR\_0 or TCR\_1 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 counter mode) or compare matches of the 8-bit channel 0 could be counted by the timer of channel 1 (compare match count mode).

### 14.6.1 16-Bit Counter Mode

When the bits CKS2 to CKS0 in TCR\_0 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.

#### (1) Setting of Compare Match Flags

- The CMF flag in TCSR\_0 is set to 1 when a 16-bit compare match event occurs.
- The CMF flag in TCSR\_1 is set to 1 when a lower 8-bit compare match event occurs.

#### (2) Counter Clear Specification

- If the CCLR1 and CCLR0 bits in TCR\_0 have been set for counter clear at compare match, the 16-bit counter (TCNT\_0 and TCNT\_1 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 been set.
- The settings of the CCLR1 and CCLR0 bits in TCR\_1 are ignored. The lower 8 bits cannot be cleared independently.

### (3) Pin Output

- Control of output from the TMO0 pin by the bits OS3 to OS0 in TCSR\_0 is in accordance with the 16-bit compare match conditions.
- Control of output from the TMO1 pin by the bits OS3 to OS0 in TCSR\_1 is in accordance with the lower 8-bit compare match conditions.

#### 14.6.2 Compare Match Count Mode

When the bits CKS2 to CKS0 in TCR\_1 are set to B'100, TCNT\_1 counts compare match A 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.

## 14.7 Interrupt Sources

### 14.7.1 Interrupt Sources and DTC Activation

- Interrupt in units 0 to 3  
There are three interrupt sources for the 8-bit timers (TMR\_0 to TMR\_7): CMIA, CMIB, and OVI. Their interrupt sources and priorities are shown in table 14.6. Each interrupt source is enabled or disabled by the corresponding interrupt enable bit in TCR or TCSR, 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 14.6 Interrupt Sources for 8-Bit Timers (TMR\_0 to TMR\_7) (in Units 0 to 3)**

Signal Name	Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority
CMIA0	CMIA0	TCORA_0 compare match	CMFA	Possible	High
CMIB0	CMIB0	TCORB_0 compare match	CMFB	Possible	↑ High
OVI0	OVI0	TCNT_0 overflow	OVF	Not possible	Low
CMIA1	CMIA1	TCORA_1 compare match	CMFA	Possible	High
CMIB1	CMIB1	TCORB_1 compare match	CMFB	Possible	↑ High
OVI1	OVI1	TCNT_1 overflow	OVF	Not possible	Low
CMIA2	CMIA2	TCORA_2 compare match	CMFA	Possible	High
CMIB2	CMIB2	TCORB_2 compare match	CMFB	Possible	↑ High
OVI2	OVI2	TCNT_2 overflow	OVF	Not possible	Low
CMIA3	CMIA3	TCORA_3 compare match	CMFA	Possible	High
CMIB3	CMIB3	TCORB_3 compare match	CMFB	Possible	↑ High
OVI3	OVI3	TCNT_3 overflow	OVF	Not possible	Low
CMIA4	CMIA4	TCORA_4 compare match	CMFA	Possible	High
CMIB4	CMIB4	TCORB_4 compare match	CMFB	Possible	↑ High
OVI4	OVI4	TCNT_4 overflow	OVF	Not possible	Low
CMIA5	CMIA5	TCORA_5 compare match	CMFA	Possible	High
CMIB5	CMIB5	TCORB_5 compare match	CMFB	Possible	↑ High
OVI5	OVI5	TCNT_5 overflow	OVF	Not possible	Low
CMIA6	CMIA6	TCORA_6 compare match	CMFA	Possible	High
CMIB6	CMIB6	TCORB_6 compare match	CMFB	Possible	↑ High
OVI6	OVI6	TCNT_6 overflow	OVF	Not possible	Low
CMIA7	CMIA7	TCORA_7 compare match	CMFA	Possible	High
CMIB7	CMIB7	TCORB_7 compare match	CMFB	Possible	↑ High
OVI7	OVI7	TCNT_7 overflow	OVF	Not possible	Low

### 14.7.2 A/D Converter Activation

The A/D converter can be activated by a compare match A of TMR\_0 or TMR\_2, whereas the  $\Delta\Sigma$  A/D converter can be activated by a compare match A of TMR\_0 or TMR\_4.

If the ADTE bit is set to 1 when the CMFA flag in TCSR\_0, TCSR\_2, or TCSR\_4 is set to 1 by the occurrence of a compare match A of TMR\_0, TMR\_2, or TMR\_4, a request to start A/D conversion is sent to the A/D converter or  $\Delta\Sigma$  A/D converter. If the 8-bit timer conversion start trigger has been selected on the A/D converter or  $\Delta\Sigma$  A/D converter side at this time, A/D conversion is started.



## 14.8 Usage Notes

### 14.8.1 Notes on Setting Cycle

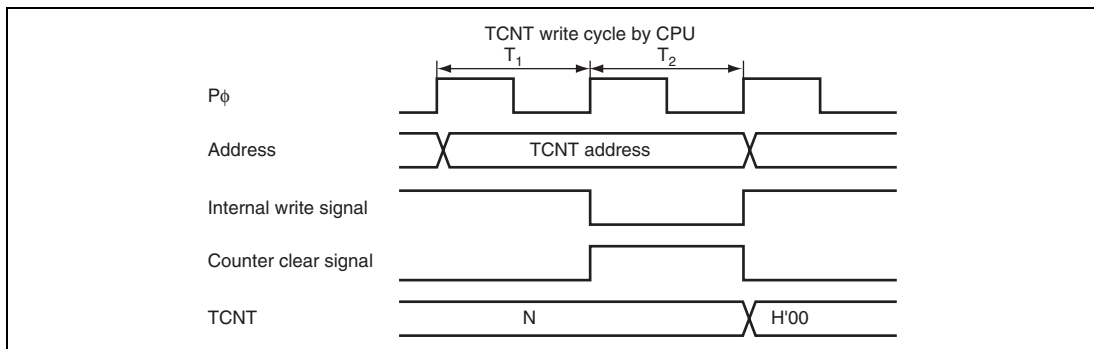
If the compare match is selected for counter clear, TCNT is cleared at the last state in the cycle in which the values of TCNT and TCOR match. TCNT updates the counter value at this last state. Therefore, the counter frequency is obtained by the following formula.

$$f = \phi / (N + 1)$$

f: Counter frequency  
 $\phi$ : Operating frequency  
 N: TCOR value

### 14.8.2 Conflict between TCNT Write and Counter Clear

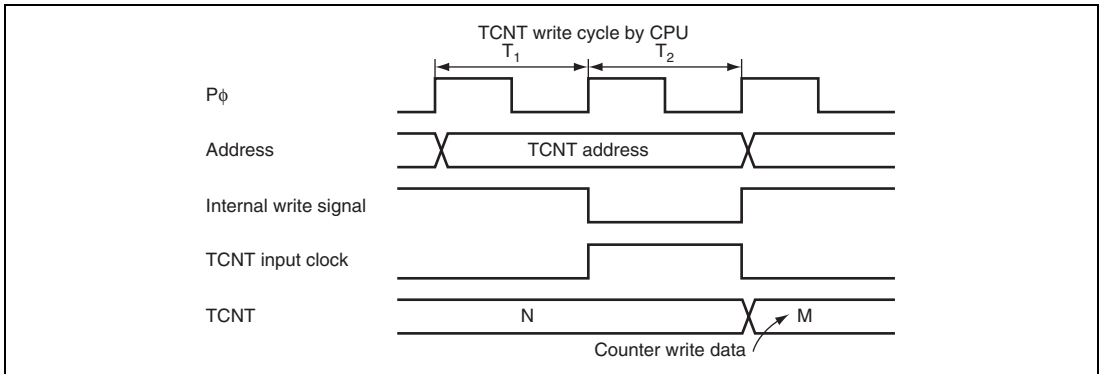
If a counter clear signal is generated during the  $T_2$  state of a TCNT write cycle, the clear takes priority and the write is not performed as shown in figure 14.15.



**Figure 14.15 Conflict between TCNT Write and Clear**

### 14.8.3 Conflict between TCNT Write and Increment

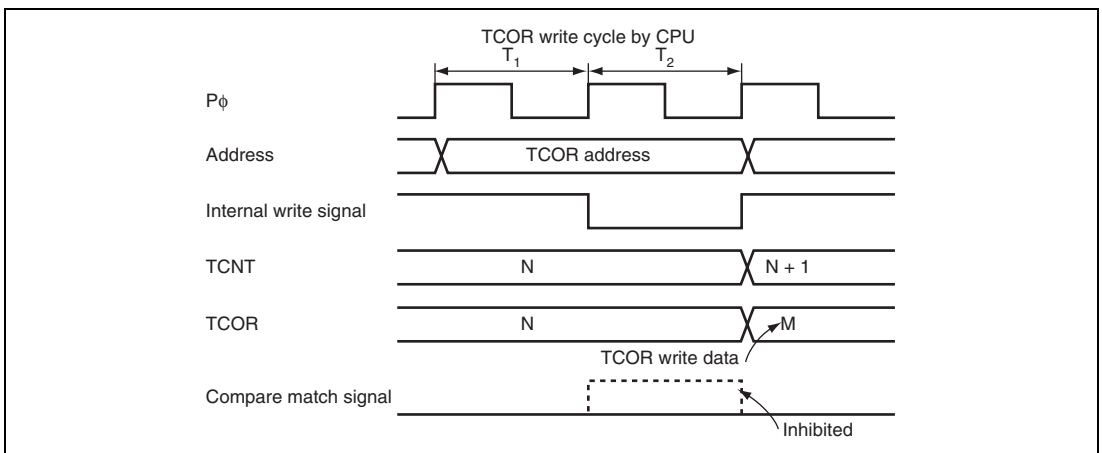
If a TCNT input clock pulse is generated during the  $T_2$  state of a TCNT write cycle, the write takes priority and the counter is not incremented as shown in figure 14.16.



**Figure 14.16 Conflict between TCNT Write and Increment**

### 14.8.4 Conflict between TCOR Write and Compare Match

If a compare match event occurs during the  $T_2$  state of a TCOR write cycle, the TCOR write takes priority and the compare match signal is inhibited as shown in figure 14.17.



**Figure 14.17 Conflict between TCOR Write and Compare Match**

### 14.8.5 Conflict 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 14.7.

**Table 14.7 Timer Output Priorities**

Output Setting	Priority
Toggle output	High
1-output	↑
0-output	
No change	Low

### 14.8.6 Switching of Internal Clocks and TCNT Operation

TCNT may be incremented erroneously depending on when the internal clock is switched. Table 14.8 shows the relationship between the timing at which the internal clock is switched (by writing to the bits CKS1 and CKS0) and the TCNT operation.

When the TCNT clock is generated from an internal clock, the rising or falling edge of the internal clock pulse are always monitored. Table 14.8 assumes that the falling edge is selected. If the signal levels of the clocks before and after switching change from high to low as shown in item 3, the change is considered as the falling edge. Therefore, a TCNT clock pulse is generated and TCNT is incremented. This is similar to when the rising edge is selected.

The erroneous increment of TCNT can also happen when switching between rising and falling edges of the internal clock, and when switching between internal and external clocks.

**Table 14.8 Switching of Internal Clock and TCNT Operation**

No.	Timing to Change CKS1 and CKS0 Bits	TCNT Clock Operation
1	Switching from low to low* <sup>1</sup>	<p data-bbox="517 231 620 438">Clock before switchover Clock after switchover TCNT input clock TCNT</p> <p data-bbox="698 454 848 478">CKS bits changed</p>
2	Switching from low to high* <sup>2</sup>	<p data-bbox="517 494 620 694">Clock before switchover Clock after switchover TCNT input clock TCNT</p> <p data-bbox="818 710 969 734">CKS bits changed</p>
3	Switching from high to low* <sup>3</sup>	<p data-bbox="517 750 620 949">Clock before switchover Clock after switchover TCNT input clock TCNT</p> <p data-bbox="758 853 788 877">*4</p> <p data-bbox="758 965 909 989">CKS bits changed</p>
4	Switching from high to high	<p data-bbox="517 1005 620 1204">Clock before switchover Clock after switchover TCNT input clock TCNT</p> <p data-bbox="884 1220 1035 1244">CKS bits changed</p>

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 because the change of the signal levels is considered as a falling edge; TCNT is incremented.

### **14.8.7 Mode Setting with Cascaded Connection**

If 16-bit counter mode and compare match count mode are specified at the same time, input clocks for TCNT\_0 and TCNT\_1 are not generated, and the counter stops. Do not specify 16-bit counter mode and compare match count mode simultaneously.

### **14.8.8 Module Stop Function Setting**

Operation of the TMR can be disabled or enabled using the module stop control register. The initial setting is for operation of the TMR to be halted. Register access is enabled by clearing the module stop state. For details, see section 24, Power-Down Modes.

### **14.8.9 Interrupts in Module Stop State**

If the TMR enters the module stop state after it has requested an interrupt, the source of interrupt to the CPU or the DTC activation source cannot be cleared. TMR interrupts should therefore be disabled before the TMR enters the module stop state.



## Section 15 Watchdog Timer (WDT)

The watchdog timer (WDT) is an 8-bit timer that outputs an overflow signal ( $\overline{\text{WDTOVF}}$ ) if a system crash, etc. prevents the CPU from writing to the timer counter, thus allowing it to overflow. At the same time, the WDT can also generate an internal reset signal.

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.

Figure 15.1 shows a block diagram of the WDT.

### 15.1 Features

- Selectable from eight counter input clocks
- Switchable between watchdog timer mode and interval timer mode
  - In watchdog timer mode
    - If the counter overflows, the WDT outputs  $\overline{\text{WDTOVF}}$ . It is possible to select whether or not the entire LSI is reset at the same time.
  - In interval timer mode
    - If the counter overflows, the WDT generates an interval timer interrupt (WOVI).

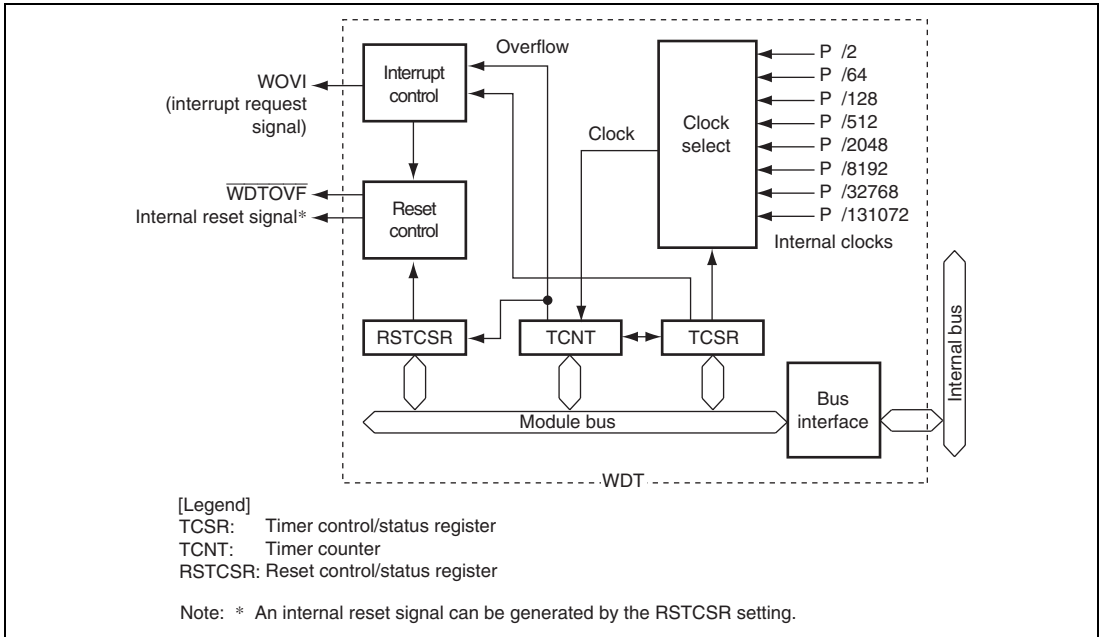


Figure 15.1 Block Diagram of WDT

## 15.2 Input/Output Pin

Table 15.1 shows the WDT pin configuration.

Table 15.1 Pin Configuration

Name	Symbol	I/O	Function
Watchdog timer overflow	WDTOVF	Output	Outputs a counter overflow signal in watchdog timer mode



## 15.3 Register Descriptions

The WDT has the following three registers. To prevent accidental overwriting, TCSR, TCNT, and RSTCSR have to be written to in a method different from normal registers. For details, see section 15.6.1, Notes on Register Access.

- Timer counter (TCNT)
- Timer control/status register (TCSR)
- Reset control/status register (RSTCSR)

### 15.3.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 when the TME bit in TCSR is cleared to 0.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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

### 15.3.2 Timer Control/Status Register (TCSR)

TCSR selects the clock source to be input to TCNT, and the timer mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	OVF	WT/ $\overline{IT}$	TME	—	—	CKS2	CKS1	CKS0
Initial Value	0	0	0	1	1	0	0	0
R/W	R/(W)*	R/W	R/W	R	R	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*	<p>Overflow Flag</p> <p>Indicates that TCNT has overflowed in interval timer mode. Only 0 can be written to this bit, to clear the flag.</p> <p>[Setting condition]</p> <p>When TCNT overflows in interval timer mode (changes from H'FF to H'00)</p> <p>When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset.</p> <p>[Clearing condition]</p> <p>Cleared by reading TCSR when OVF = 1, then writing 0 to OVF</p> <p>(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
6	WT/IT	0	R/W	<p>Timer Mode Select</p> <p>Selects whether the WDT is used as a watchdog timer or interval timer.</p> <p>0: Interval timer mode</p> <p>    When TCNT overflows, an interval timer interrupt (WOVI) is requested.</p> <p>1: Watchdog timer mode</p> <p>    When TCNT overflows, the <math>\overline{\text{WDTOVF}}</math> signal is output.</p>
5	TME	0	R/W	<p>Timer Enable</p> <p>When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00.</p>
4, 3	—	All 1	R	<p>Reserved</p> <p>These are read-only bits and cannot be modified.</p>
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	Select the clock source to be input to TCNT. The overflow cycle for $P\phi = 20$ MHz is indicated in parentheses.
0	CKS0	0	R/W	<p>000: Clock <math>P\phi/2</math> (cycle: 25.6 <math>\mu\text{s}</math>)</p> <p>001: Clock <math>P\phi/64</math> (cycle: 819.2 <math>\mu\text{s}</math>)</p> <p>010: Clock <math>P\phi/128</math> (cycle: 1.6 ms)</p> <p>011: Clock <math>P\phi/512</math> (cycle: 6.6 ms)</p> <p>100: Clock <math>P\phi/2048</math> (cycle: 26.2 ms)</p> <p>101: Clock <math>P\phi/8192</math> (cycle: 104.9 ms)</p> <p>110: Clock <math>P\phi/32768</math> (cycle: 419.4 ms)</p> <p>111: Clock <math>P\phi/131072</math> (cycle: 1.68 s)</p>

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

### 15.3.3 Reset Control/Status Register (RSTCSR)

RSTCSR 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 WDT overflows.

Bit	7	6	5	4	3	2	1	0
Bit Name	WOVF	RSTE	—	—	—	—	—	—
Initial Value	0	0	0	1	1	1	1	1
R/W	R/(W)*	R/W	R/W	R	R	R	R	R

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

Bit	Bit Name	Initial Value	R/W	Description
7	WOVF	0	R/(W)*	<p>Watchdog Timer Overflow Flag</p> <p>This bit is set when TCNT overflows in watchdog timer mode. This bit cannot be set in interval timer mode, and only 0 can be written.</p> <p>[Setting condition]</p> <p>When TCNT overflows (changed from H'FF to H'00) in watchdog timer mode</p> <p>[Clearing condition]</p> <p>Reading RSTCSR when WOVF = 1, and then writing 0 to WOVF</p>
6	RSTE	0	R/W	<p>Reset Enable</p> <p>Specifies whether or not this LSI is internally reset if TCNT overflows during watchdog timer operation.</p> <p>0: LSI is not reset even if TCNT overflows (Though this LSI is not reset, TCNT and TCSR in WDT are reset)</p> <p>1: LSI is reset if TCNT overflows</p>

Bit	Bit Name	Initial Value	R/W	Description
5	—	0	R/W	Reserved Although this bit is readable/writable, reading from or writing to this bit does not affect operation.
4 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

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

## 15.4 Operation

### 15.4.1 Watchdog Timer Mode

To use the WDT in watchdog timer mode, set both the  $\overline{WT/IT}$  and TME bits in TCSR to 1.

During watchdog timer operation, if TCNT overflows without being rewritten because of a system crash or other error, the  $\overline{WDTOVF}$  signal is output. This ensures that TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally H'00 is written) before overflow occurs. This  $\overline{WDTOVF}$  signal can be used to reset the LSI internally in watchdog timer mode.

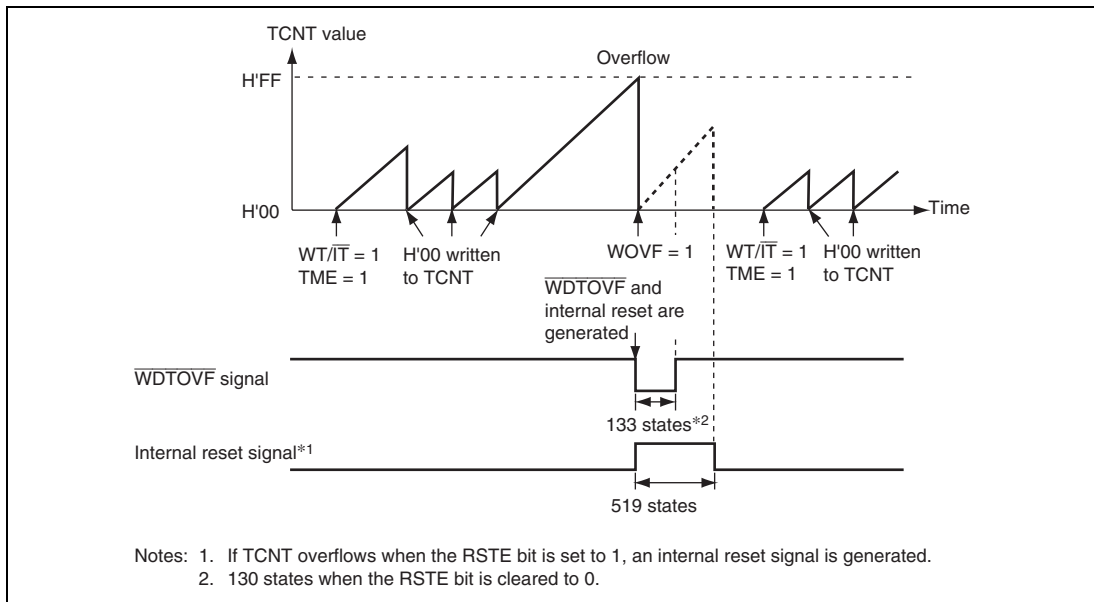
If TCNT overflows when the RSTE bit in RSTCSR is set to 1, a signal that resets this LSI internally is generated at the same time as the  $\overline{WDTOVF}$  signal. If a reset caused by a signal input to the  $\overline{RES}$  pin occurs at the same time as a reset caused by a WDT overflow, the  $\overline{RES}$  pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.

The  $\overline{WDTOVF}$  signal is output for 133 cycles of  $P\phi$  when RSTE = 1 in RSTCSR, and for 130 cycles of  $P\phi$  when RSTE = 0 in RSTCSR. The internal reset signal is output for 519 cycles of  $P\phi$ .

When RSTE = 1, an internal reset signal is generated. Since the system clock control register (SCKCR) is initialized, the multiplication ratio of  $P\phi$  becomes the initial value.

When RSTE = 0, an internal reset signal is not generated. Neither SCKCR nor the multiplication ratio of  $P\phi$  is changed.

When TCNT overflows in watchdog timer mode, the WOVF bit in RSTCSR is set to 1. If TCNT overflows when the RSTE bit in RSTCSR is set to 1, an internal reset signal is generated for the entire LSI.



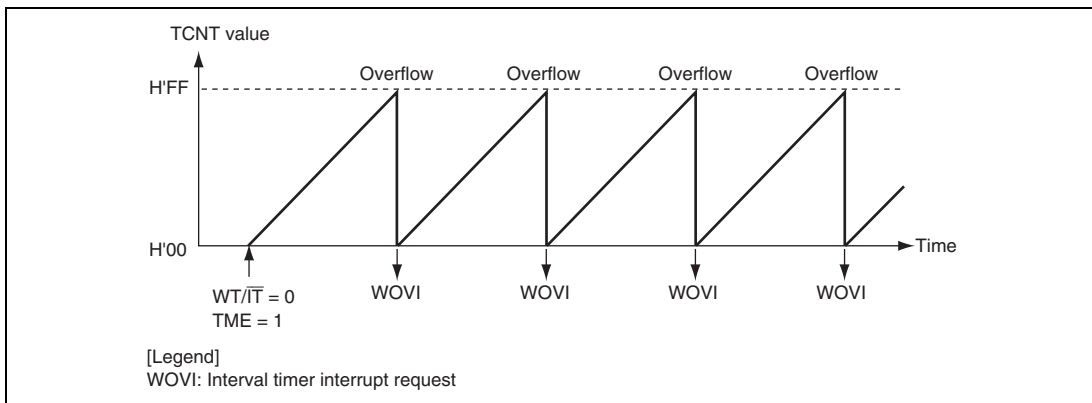
**Figure 15.2 Operation in Watchdog Timer Mode**

### 15.4.2 Interval Timer Mode

To use the WDT as an interval timer, set the  $\overline{WT/IT}$  bit to 0 and the TME bit to 1 in TCSR.

When the WDT is used as an interval timer, an interval timer interrupt (WOVI) is generated each time the TCNT overflows. Therefore, an interrupt can be generated at intervals.

When the TCNT overflows in interval timer mode, an interval timer interrupt (WOVI) is requested at the same time the OVF bit in the TCSR is set to 1.



**Figure 15.3 Operation in Interval Timer Mode**

## 15.5 Interrupt Source

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. The OVF flag must be cleared to 0 in the interrupt handling routine.

**Table 15.2 WDT Interrupt Source**

Name	Interrupt Source	Interrupt Flag	DTC Activation
WOVI	TCNT overflow	OVF	Impossible

## 15.6 Usage Notes

### 15.6.1 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.

#### (1) Writing to TCNT, TCSR, and RSTCSR

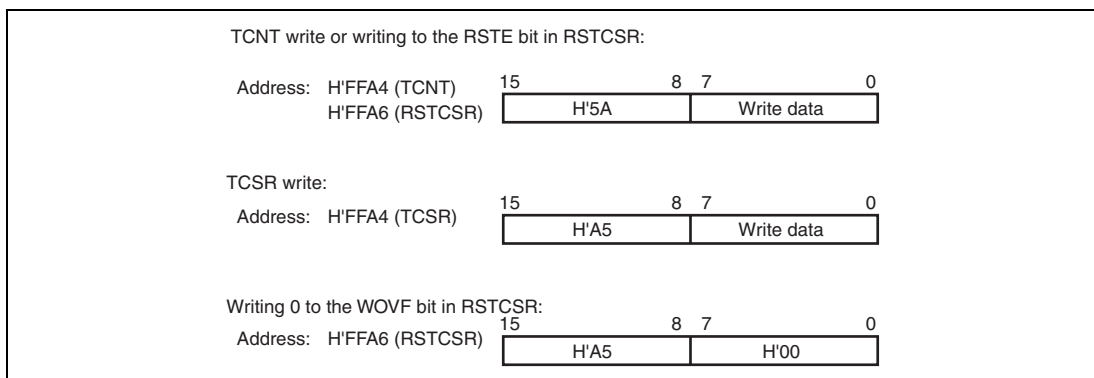
TCNT and TCSR must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

For writing, TCNT and TCSR are assigned to the same address. Accordingly, perform data transfer as shown in figure 15.4. The transfer instruction writes the lower byte data to TCNT or TCSR.

To write to RSTCSR, execute a word transfer instruction for address H'FFA6. A byte transfer instruction cannot be used to write to RSTCSR.

The method of writing 0 to the WOVF bit in RSTCSR differs from that of writing to the RSTE bit in RSTCSR. Perform data transfer as shown in figure 15.4.

At data transfer, the transfer instruction clears the WOVF bit to 0, but has no effect on the RSTE bit. To write to the RSTE bit, perform data transfer as shown in figure 15.4. In this case, the transfer instruction writes the value in bit 6 of the lower byte to the RSTE bit, but has no effect on the WOVF bit.



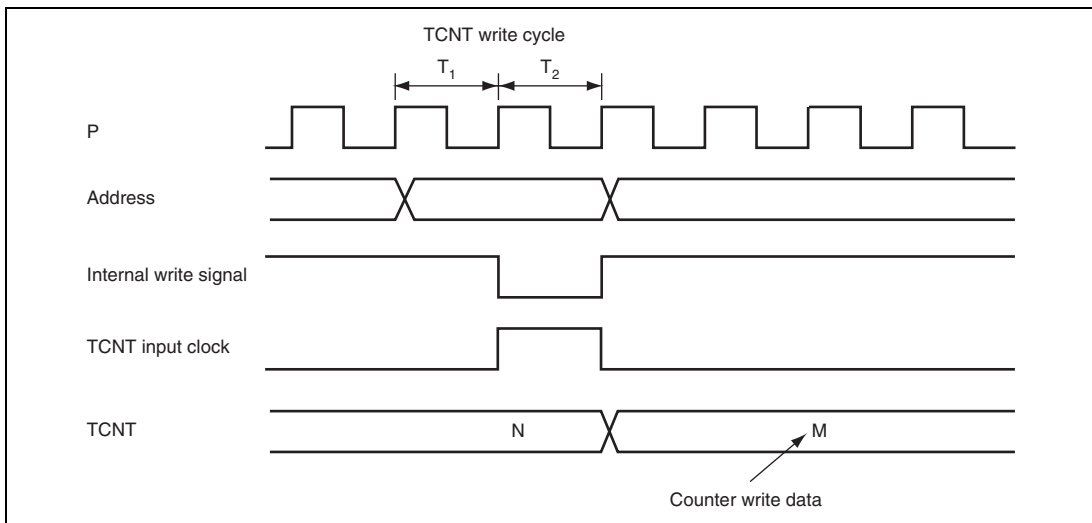
**Figure 15.4 Writing to TCNT, TCSR, and RSTCSR**

## (2) Reading from TCNT, TCSR, and RSTCSR

These registers can be read from in the same way as other registers. For reading, TCSR is assigned to address H'FFA4, TCNT to address H'FFA5, and RSTCSR to address H'FFA7.

### 15.6.2 Conflict between Timer Counter (TCNT) Write and Increment

If a TCNT clock pulse is generated during the T2 cycle of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 15.5 shows this operation.



**Figure 15.5 Conflict between TCNT Write and Increment**

### 15.6.3 Changing Values of Bits CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before the values of bits CKS2 to CKS0 are changed.

### 15.6.4 Switching between Watchdog Timer Mode and Interval Timer Mode

If the timer mode is switched from watchdog timer mode to interval timer mode while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before switching the timer mode.



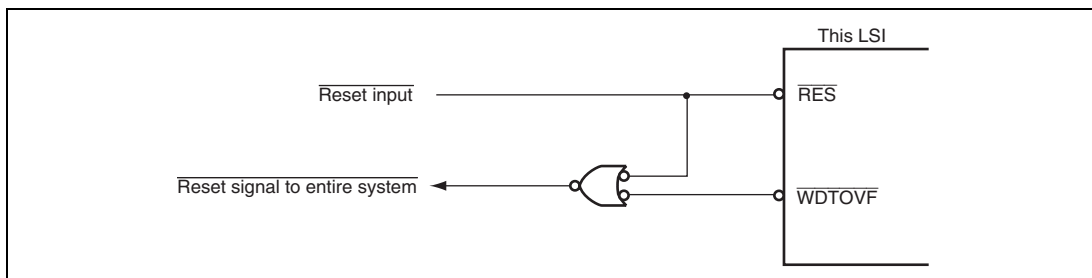
### 15.6.5 Internal Reset in Watchdog Timer Mode

This LSI is not reset internally if TCNT overflows while the RSTE bit is cleared to 0 during watchdog timer mode 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.

### 15.6.6 System Reset by $\overline{\text{WDTOVF}}$ Signal

If the  $\overline{\text{WDTOVF}}$  signal is input to the  $\overline{\text{RES}}$  pin, this LSI 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 a circuit like that shown in figure 15.6.



**Figure 15.6** Circuit for System Reset by  $\overline{\text{WDTOVF}}$  Signal (Example)

### 15.6.7 Transition to Watchdog Timer Mode or Software Standby Mode

When the WDT operates in watchdog timer mode, a transition to software standby mode is not made even when the SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1. Instead, a transition to sleep mode is made.

To transit to software standby mode, the SLEEP instruction must be executed after halting the WDT (clearing the TME bit to 0).

When the WDT operates in interval timer mode, a transition to software standby mode is made through execution of the SLEEP instruction when the SSBY bit in SBYCR is set to 1.



## Section 16 Serial Communication Interface (SCI)

This LSI has five independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clocked synchronous serial communication. Asynchronous 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 function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface supporting ISO/IEC 7816-3 (Identification Card) as an extended asynchronous communication mode. Figure 16.1 shows a block diagram of the SCI.

### 16.1 Features

- Choice of asynchronous or clocked synchronous serial communication mode
- 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.

- On-chip baud rate generator allows any bit rate to be selected

The external clock can be selected as a transfer clock source (except for the smart card interface).

- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode 7-bit data)
- Four interrupt sources

The interrupt sources are transmit-end, transmit-data-empty, receive-data-full, and receive error. The transmit-data-empty and receive-data-full interrupt sources can activate the DTC or DMAC.

- Module stop state specifiable

#### Asynchronous Mode:

- Data length: 7 or 8 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- 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

- Average transfer rate generator (SCI\_2 only)  
10.667-MHz operation: 460.606 kbps or 115.152 kbps can be selected  
16-MHz operation: 720 kbps, 460.784 kbps, or 115.196 kbps can be selected  
32-MHz operation: 720 kbps

### Clocked Synchronous Mode:

- Data length: 8 bits
- Receive error detection: Overrun errors

### Smart Card Interface:

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on receiving an error signal during transmission
- Both direct convention and inverse convention are supported

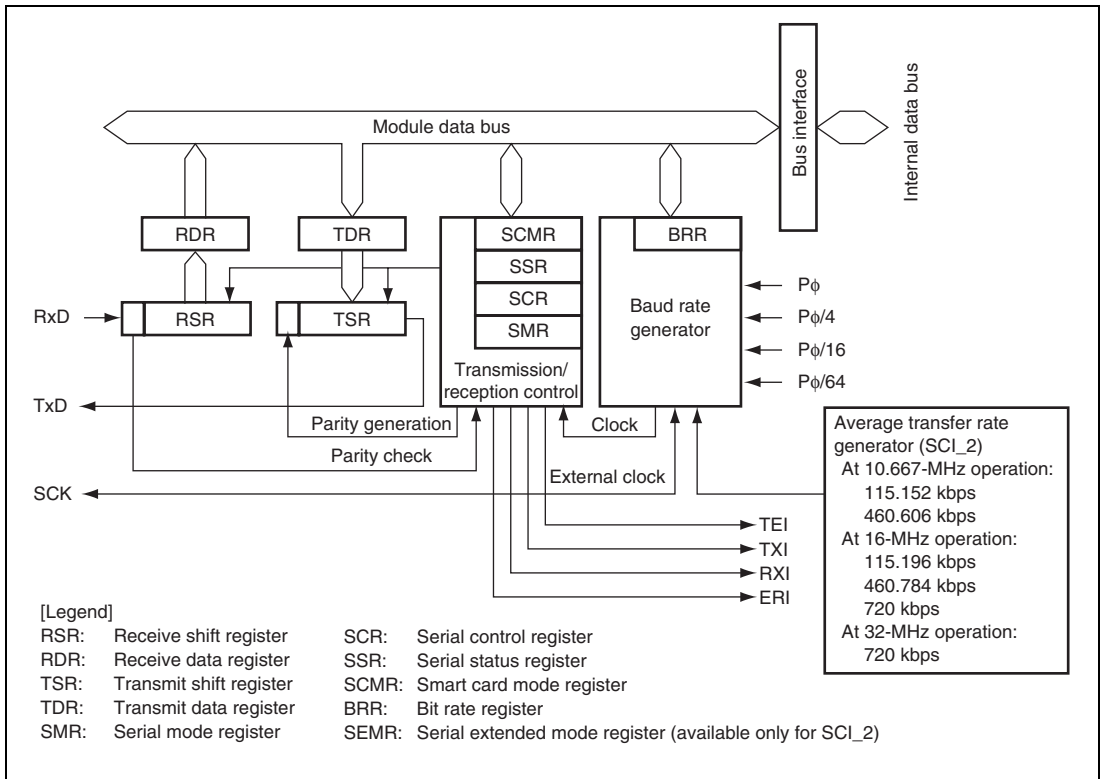


Figure 16.1 Block Diagram of SCI

## 16.2 Input/Output Pins

Table 16.1 lists the pin configuration of the SCI.

**Table 16.1 Pin Configuration**

Channel	Pin Name*	I/O	Function
0	SCK0	I/O	Channel 0 clock input/output
	RxD0	Input	Channel 0 receive data input
	TxD0	Output	Channel 0 transmit data output
1	SCK1	I/O	Channel 1 clock input/output
	RxD1	Input	Channel 1 receive data input
	TxD1	Output	Channel 1 transmit data output
2	SCK2	I/O	Channel 2 clock input/output
	RxD2	Input	Channel 2 receive data input
	TxD2	Output	Channel 2 transmit data output
3	SCK3	I/O	Channel 3 clock input/output
	RxD3	Input	Channel 3 receive data input
	TxD3	Output	Channel 3 transmit data output
4	SCK4	I/O	Channel 4 clock input/output
	RxD4	Input	Channel 4 receive data input
	TxD4	Output	Channel 4 transmit data output

Note: \* Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

## 16.3 Register Descriptions

The SCI has the following registers. Some bits in the serial mode register (SMR), serial status register (SSR), and serial control register (SCR) have different functions in different modes—normal serial communication interface mode and smart card interface mode; therefore, the bits are described separately for each mode in the corresponding register sections.

### Channel 0:

- Receive shift register\_0 (RSR\_0)
- Transmit shift register\_0 (TSR\_0)
- Receive data register\_0 (RDR\_0)
- Transmit data register\_0 (TDR\_0)
- Serial mode register\_0 (SMR\_0)
- Serial control register\_0 (SCR\_0)
- Serial status register\_0 (SSR\_0)
- Smart card mode register\_0 (SCMR\_0)
- Bit rate register\_0 (BRR\_0)

### Channel 1:

- Receive shift register\_1 (RSR\_1)
- Transmit shift register\_1 (TSR\_1)
- Receive data register\_1 (RDR\_1)
- Transmit data register\_1 (TDR\_1)
- Serial mode register\_1 (SMR\_1)
- Serial control register\_1 (SCR\_1)
- Serial status register\_1 (SSR\_1)
- Smart card mode register\_1 (SCMR\_1)
- Bit rate register\_1 (BRR\_1)

**Channel 2:**

- Receive shift register\_2 (RSR\_2)
- Transmit shift register\_2 (TSR\_2)
- Receive data register\_2 (RDR\_2)
- Transmit data register\_2 (TDR\_2)
- Serial mode register\_2 (SMR\_2)
- Serial control register\_2 (SCR\_2)
- Serial status register\_2 (SSR\_2)
- Smart card mode register\_2 (SCMR\_2)
- Bit rate register\_2 (BRR\_2)
- Serial extended mode register\_2 (SEMR\_2) (SCI\_2 only)

**Channel 3:**

- Receive shift register\_3 (RSR\_3)
- Transmit shift register\_3 (TSR\_3)
- Receive data register\_3 (RDR\_3)
- Transmit data register\_3 (TDR\_3)
- Serial mode register\_3 (SMR\_3)
- Serial control register\_3 (SCR\_3)
- Serial status register\_3 (SSR\_3)
- Smart card mode register\_3 (SCMR\_3)
- Bit rate register\_3 (BRR\_3)

**Channel 4:**

- Receive shift register\_4 (RSR\_4)
- Transmit shift register\_4 (TSR\_4)
- Receive data register\_4 (RDR\_4)
- Transmit data register\_4 (TDR\_4)
- Serial mode register\_4 (SMR\_4)
- Serial control register\_4 (SCR\_4)
- Serial status register\_4 (SSR\_4)
- Smart card mode register\_4 (SCMR\_4)
- Bit rate register\_4 (BRR\_4)

### 16.3.1 Receive Shift Register (RSR)

RSR is a shift register which is used to receive serial data input from the RxD pin and converts it into parallel data. When one frame of data has been received, it is transferred to RDR automatically. RSR cannot be directly accessed by the CPU.

### 16.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one frame of serial data, it transfers the received serial data from RSR to RDR where it is stored. This allows RSR to receive the next data. Since RSR and RDR function as a double buffer in this way, continuous receive operations can be performed. After confirming that the RDRF bit in SSR is set to 1, read RDR only once. RDR cannot be written to by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

### 16.3.3 Transmit Data Register (TDR)

TDR is an 8-bit register that stores transmit data. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structures of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR when one frame of data is transmitted, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read from or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR for only once after confirming that the TDRE bit in SSR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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



### 16.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first automatically transfers transmit data from TDR to TSR, and then sends the data to the TxD pin. TSR cannot be directly accessed by the CPU.

### 16.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock source. Some bits in SMR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	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

#### Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	C/ $\bar{A}$	0	R/W	Communication Mode 0: Asynchronous mode 1: Clocked synchronous mode
6	CHR	0	R/W	Character Length (valid only in asynchronous mode) 0: Selects 8 bits as the data length. 1: Selects 7 bits as the data length. LSB-first is fixed and the MSB (bit 7) in TDR is not transmitted in transmission. In clocked synchronous mode, a fixed data length of 8 bits is used.

Bit	Bit Name	Initial Value	R/W	Description
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. For a multiprocessor format, parity bit addition and checking are not performed regardless of the PE bit setting.
4	O/ $\bar{E}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity.
3	STOP	0	R/W	Stop Bit Length (valid only in asynchronous mode) Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.
2	MP	0	R/W	Multiprocessor Mode (valid only in asynchronous mode) When this bit is set to 1, the multiprocessor function is enabled. The PE bit and O/ $\bar{E}$ bit settings are invalid in multiprocessor mode.
1	CKS1	0	R/W	Clock Select 1, 0
0	CKS0	0	R/W	These bits select the clock source for the baud rate generator. 00: P $\phi$ clock (n = 0) 01: P $\phi$ /4 clock (n = 1) 10: P $\phi$ /16 clock (n = 2) 11: P $\phi$ /64 clock (n = 3) For the relation between the settings of these bits and the baud rate, see section 16.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 16.3.9, Bit Rate Register (BRR)).

**Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):**

Bit	Bit Name	Initial Value	R/W	Description
7	GM	0	R/W	GSM Mode  Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 etu from the start and the clock output control function is appended. For details, see sections 16.7.6, Data Transmission (Except in Block Transfer Mode) and 16.7.8, Clock Output Control.
6	BLK	0	R/W	Setting this bit to 1 allows block transfer mode operation. For details, see section 16.7.3, Block Transfer Mode.
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode)  When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. Set this bit to 1 in smart card interface mode.
4	O/ $\bar{E}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode)  0: Selects even parity 1: Selects odd parity  For details on the usage of this bit in smart card interface mode, see section 16.7.2, Data Format (Except in Block Transfer Mode).
3	BCP1	0	R/W	Basic Clock Pulse 1,0
2	BCP0	0	R/W	These bits select the number of basic clock cycles in a 1-bit data transfer time in smart card interface mode.  00: 32 clock cycles (S = 32) 01: 64 clock cycles (S = 64) 10: 372 clock cycles (S = 372) 11: 256 clock cycles (S = 256)  For details, see section 16.7.4, Receive Data Sampling Timing and Reception Margin. S is described in section 16.3.9, Bit Rate Register (BRR).

Bit	Bit Name	Initial Value	R/W	Description
1	CKS1	0	R/W	Clock Select 1,0
0	CKS0	0	R/W	These bits select the clock source for the baud rate generator. 00: P $\phi$ clock (n = 0) 01: P $\phi$ /4 clock (n = 1) 10: P $\phi$ /16 clock (n = 2) 11: P $\phi$ /64 clock (n = 3) For the relation between the settings of these bits and the baud rate, see section 16.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 16.3.9, Bit Rate Register (BRR)).

Note:  $t_{etu}$  (Elementary Time Unit): 1-bit transfer time

### 16.3.6 Serial Control Register (SCR)

SCR is a register that enables/disables the following SCI transfer operations and interrupt requests, and selects the transfer clock source. For details on interrupt requests, see section 16.8, Interrupt Sources. Some bits in SCR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	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

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	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

**Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):**

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	<p>Transmit Interrupt Enable</p> <p>When this bit is set to 1, a TXI interrupt request is enabled.</p> <p>A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.</p>
6	RIE	0	R/W	<p>Receive Interrupt Enable</p> <p>When this bit is set to 1, RXI and ERI interrupt requests are enabled.</p> <p>RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.</p>
5	TE	0	R/W	<p>Transmit Enable</p> <p>When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format.</p> <p>If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.</p>
4	RE	0	R/W	<p>Receive Enable</p> <p>When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format.</p> <p>Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.</p>

Bit	Bit Name	Initial Value	R/W	Description
3	MPIE	0	R/W	<p>Multiprocessor Interrupt Enable (valid only when the MP bit in SMR is 1 in asynchronous mode)</p> <p>When this bit is set to 1, receive data in which the multiprocessor bit is 0 is skipped, and setting of the RDRF, FER, and ORER status flags in SSR is disabled. On receiving data in which the multiprocessor bit is 1, this bit is automatically cleared and normal reception is resumed. For details, see section 16.5, Multiprocessor Communication Function.</p> <p>When receive data including MPB = 0 in SSR is being received, transfer of the received data from RSR to RDR, detection of reception errors, and the settings of RDRF, FER, and ORER flags in SSR are not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is automatically cleared to 0, and RXI and ERI interrupt requests (in the case where the TIE and RIE bits in SCR are set to 1) and setting of the FER and ORER flags are enabled.</p>
2	TEIE	0	R/W	<p>Transmit End Interrupt Enable</p> <p>When this bit is set to 1, a TEI interrupt request is enabled. A TEI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0 in order to clear the TEND flag to 0, or by clearing the TEIE bit to 0.</p>

Bit	Bit Name	Initial Value	R/W	Description
1	CKE1	0	R/W	Clock Enable 1, 0 (for SCI_0, 1, 3, 4)
0	CKE0	0	R/W	<p>These bits select the clock source and SCK pin function.</p> <ul style="list-style-type: none"> <li>Asynchronous mode</li> </ul> <p>00: On-chip baud rate generator The SCK pin can be used as an I/O port pin.</p> <p>01: On-chip baud rate generator The SCK pin outputs a clock with the same frequency as the bit rate.</p> <p>1X: External clock A clock with a frequency 16 times the bit rate should be input from the SCK pin.</p> <ul style="list-style-type: none"> <li>Clocked synchronous mode</li> </ul> <p>0X: Internal clock The SCK pin functions as a clock output pin.</p> <p>1X: External clock The SCK pin functions as a clock input pin.</p> <p>Clock Enable 1, 0 (for SCI_2)</p> <p>These bits select the clock source and SCK pin function.</p> <ul style="list-style-type: none"> <li>Asynchronous mode</li> </ul> <p>00: On-chip baud rate generator The SCK pin can be used as an I/O port pin.</p> <p>01: On-chip baud rate generator The SCK pin outputs a clock with the same frequency as the bit rate.</p> <p>1X: External clock or average transfer rate generator</p> <ul style="list-style-type: none"> <li>When using an external clock, a clock with a frequency 16 times the bit rate should be input from the SCK pin.</li> <li>Average transfer rate generator is used.</li> </ul> <ul style="list-style-type: none"> <li>Clocked synchronous mode</li> </ul> <p>0X: Internal clock The SCK pin functions as a clock output pin.</p> <p>1X: External clock The SCK pin functions as a clock input pin.</p>

Note: X: Don't care

**Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):**

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	<p>Transmit Interrupt Enable</p> <p>When this bit is set to 1, a TXI interrupt request is enabled.</p> <p>A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.</p>
6	RIE	0	R/W	<p>Receive Interrupt Enable</p> <p>When this bit is set to 1, RXI and ERI interrupt requests are enabled.</p> <p>RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.</p>
5	TE	0	R/W	<p>Transmit Enable</p> <p>When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format.</p> <p>If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.</p>
4	RE	0	R/W	<p>Receive Enable</p> <p>When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format.</p> <p>Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.</p>
3	MPIE	0	R/W	<p>Multiprocessor Interrupt Enable (valid only when the MP bit in SMR is 1 in asynchronous mode)</p> <p>Write 0 to this bit in smart card interface mode.</p>



Bit	Bit Name	Initial Value	R/W	Description
2	TEIE	0	R/W	Transmit End Interrupt Enable Write 0 to this bit in smart card interface mode.
1	CKE1	0	R/W	Clock Enable 1, 0
0	CKE0	0	R/W	These bits control the clock output from the SCK pin. In GSM mode, clock output can be dynamically switched. For details, see section 16.7.8, Clock Output Control. <ul style="list-style-type: none"> <li>When GM in SMR = 0 <p>00: Output disabled (SCK pin functions as I/O port.)</p> <p>01: Clock output</p> <p>1X: Reserved</p> </li> <li>When GM in SMR = 1 <p>00: Output fixed low</p> <p>01: Clock output</p> <p>10: Output fixed high</p> <p>11: Clock output</p> </li> </ul>

### 16.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. TDRE, RDRF, ORER, PER, and FER can only be cleared. Some bits in SSR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	FRE	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.

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	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 clear the flag.

**Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):**

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	<p>Transmit Data Register Empty</p> <p>Indicates whether TDR contains transmit data.</p> <p>[Setting conditions]</p> <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</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>• When a TXI interrupt request is issued allowing DMAC or DTC to write data to TDR</li> </ul>
6	RDRF	0	R/(W)*	<p>Receive Data Register Full</p> <p>Indicates whether receive data is stored in RDR.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When serial reception ends normally and receive data is transferred from RSR to RDR</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to RDRF after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>• When an RXI interrupt request is issued allowing DMAC or DTC to read data from RDR</li> </ul> <p>The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.</p> <p>Note that when the next serial reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.</p>

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	<p>Overrun Error</p> <p>Indicates that an overrun error has occurred during reception and the reception ends abnormally.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the next serial reception is completed while RDRF = 1</li> </ul> <p>In RDR, receive data prior to an overrun error occurrence is retained, but data received after the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to ORER after reading ORER = 1</li> </ul> <p>Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.</p> <p>(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
4	FER	0	R/(W)*	<p>Framing Error</p> <p>Indicates that a framing error has occurred during reception in asynchronous mode and the reception ends abnormally.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the stop bit is 0</li> </ul> <p>In 2-stop-bit mode, only the first stop bit is checked whether it is 1 but the second stop bit is not checked. Note that receive data when the framing error occurs is transferred to RDR, however, the RDRF flag is not set. In addition, when the FER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to FER after reading FER = 1</li> </ul> <p>Even when the RE bit in SCR is cleared, the FER flag is not affected and retains its previous value.</p> <p>(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	<p>Parity Error</p> <p>Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a parity error is detected during reception Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to PER after reading PER = 1 Even when the RE bit in SCR is cleared, the PER bit is not affected and retains its previous value. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>
2	TEND	1	R	<p>Transmit End</p> <p>[Setting conditions]</p> <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 transmit character</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written to TDRE after reading TDRE = 1</li> <li>When a TXI interrupt request is issued allowing DMAC or DTC to write data to TDR</li> </ul>
1	MPB	0	R	<p>Multiprocessor Bit</p> <p>Stores the multiprocessor bit value in the receive frame. When the RE bit in SCR is cleared to 0 its previous state is retained.</p>
0	MPBT	0	R/W	<p>Multiprocessor Bit Transfer</p> <p>Sets the multiprocessor bit value to be added to the transmit frame.</p>

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

**Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):**

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	<p>Transmit Data Register Empty</p> <p>Indicates whether TDR contains transmit data.</p> <p>[Setting conditions]</p> <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</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>• When a TXI interrupt request is issued allowing DMAC or DTC to write data to TDR</li> </ul>
6	RDRF	0	R/(W)*	<p>Receive Data Register Full</p> <p>Indicates whether receive data is stored in RDR.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When serial reception ends normally and receive data is transferred from RSR to RDR</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to RDRF after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>• When an RXI interrupt request is issued allowing DMAC or DTC to read data from RDR</li> </ul> <p>The RDRF flag is not affected and retains its previous value even when the RE bit in SCR is cleared to 0.</p> <p>Note that when the next reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.</p>

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	<p>Overrun Error</p> <p>Indicates that an overrun error has occurred during reception and the reception ends abnormally.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the next serial reception is completed while RDRF = 1</li> </ul> <p>In RDR, the receive data prior to an overrun error occurrence is retained, but data received following the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to ORER after reading ORER = 1</li> </ul> <p>Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.</p> <p>(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
4	ERS	0	R/(W)*	<p>Error Signal Status</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a low error signal is sampled</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to ERS after reading ERS = 1</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	<p>Parity Error</p> <p>Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"><li>When a parity error is detected during reception Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.</li></ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"><li>When 0 is written to PER after reading PER = 1 Even when the RE bit in SCR is cleared, the PER flag is not affected and retains its previous value. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li></ul>

---



Bit	Bit Name	Initial Value	R/W	Description
2	TEND	1	R	<p>Transmit End</p> <p>This bit is set to 1 when no error signal is sent from the receiving side and the next transmit data is ready to be transferred to TDR.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When both the TE and ERS bits in SCR are 0</li> <li>• When ERS = 0 and TDRE = 1 after a specified time passed after completion of 1-byte data transfer. The set timing depends on the register setting as follows: <ul style="list-style-type: none"> <li>When GM = 0 and BLK = 0, 2.5 etu after transmission start</li> <li>When GM = 0 and BLK = 1, 1.5 etu after transmission start</li> <li>When GM = 1 and BLK = 0, 1.0 etu after transmission start</li> <li>When GM = 1 and BLK = 1, 1.0 etu after transmission start</li> </ul> </li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE = 1</li> <li>• When a TXI interrupt request is issued allowing DMAC or DTC to write the next data to TDR</li> </ul>
1	MPB	0	R	<p>Multiprocessor Bit</p> <p>Not used in smart card interface mode.</p>
0	MPBT	0	R/W	<p>Multiprocessor Bit Transfer</p> <p>Write 0 to this bit in smart card interface mode.</p>

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

### 16.3.8 Smart Card Mode Register (SCMR)

SCMR selects smart card interface mode and its format.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	SDIR	SINV	—	SMIF
Initial Value	1	1	1	1	0	0	1	0
R/W	R	R	R	R	R/W	R/W	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These are read-only bits and cannot be modified.
3	SDIR	0	R/W	Smart Card Data Transfer Direction Selects the serial/parallel conversion format. 0: Transfer with LSB-first 1: Transfer with MSB-first This bit is valid only when the 8-bit data format is used for transmission/reception; when the 7-bit data format is used, data is always transmitted/received with LSB-first.
2	SINV	0	R/W	Smart Card Data Invert Inverts the transmit/receive data logic level. This bit does not affect the logic level of the parity bit. To invert the parity bit, invert the O $\bar{E}$ bit in SMR. 0: TDR contents are transmitted as they are. Receive data is stored as it is in RDR. 1: TDR contents are inverted before being transmitted. Receive data is stored in inverted form in RDR.
1	—	1	R	Reserved This is a read-only bit and cannot be modified.
0	SMIF	0	R/W	Smart Card Interface Mode Select When this bit is set to 1, smart card interface mode is selected. 0: Normal asynchronous or clocked synchronous mode 1: Smart card interface mode

### 16.3.9 Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 16.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clocked synchronous mode, and smart card interface mode. The initial value of BRR is H'FF, and it can be read from or written to by the CPU at all times.

**Table 16.2 Relationships between N Setting in BRR and Bit Rate B**

Mode	Bit Rate	Error
Asynchronous mode	$N = \frac{P\phi \times 10^6}{64 \times 2^{2n-1} \times B} - 1$	$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1 \right\} \times 100$
Clocked synchronous mode	$N = \frac{P\phi \times 10^6}{8 \times 2^{2n-1} \times B} - 1$	
Smart card interface mode	$N = \frac{P\phi \times 10^6}{S \times 2^{2n+1} \times B} - 1$	$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N+1)} - 1 \right\} \times 100$

[Legend]

B: Bit rate (bit/s)

N: BRR setting for baud rate generator ( $0 \leq N \leq 255$ )

$P\phi$ : Operating frequency (MHz)

n and S: Determined by the SMR settings shown in the following table.

SMR Setting			SMR Setting		
CKS1	CKS0	n	BCP1	BCP0	S
0	0	0	0	0	32
0	1	1	0	1	64
1	0	2	1	0	372
1	1	3	1	1	256

Table 16.3 shows sample N settings in BRR in normal asynchronous mode. Table 16.4 shows the maximum bit rate settable for each operating frequency. Tables 16.6 and 16.8 show sample N settings in BRR in clocked synchronous mode and smart card interface mode, respectively. In smart card interface mode, the number of basic clock cycles S in a 1-bit data transfer time can be selected. For details, see section 16.7.4, Receive Data Sampling Timing and Reception Margin. Tables 16.5 and 16.7 show the maximum bit rates with external clock input.

When the ABCS bit in serial extended mode register\_2 is set to 1 in asynchronous mode, the bit rates for SCI\_2 are double the bit rates shown in table 16.3.

**Table 16.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (1)**

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	8			9.8304			10			12		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	141	0.03	2	174	-0.26	2	177	-0.25	2	212	0.03
150	2	103	0.16	2	127	0.00	2	129	0.16	2	155	0.16
300	1	207	0.16	1	255	0.00	2	64	0.16	2	77	0.16
600	1	103	0.16	1	127	0.00	1	129	0.16	1	155	0.16
1200	0	207	0.16	0	255	0.00	1	64	0.16	1	77	0.16
2400	0	103	0.16	0	127	0.00	0	129	0.16	0	155	0.16
4800	0	51	0.16	0	63	0.00	0	64	0.16	0	77	0.16
9600	0	25	0.16	0	31	0.00	0	32	-1.36	0	38	0.16
19200	0	12	0.16	0	15	0.00	0	15	1.73	0	19	-2.34
31250	0	7	0.00	0	9	-1.70	0	9	0.00	0	11	0.00
38400	—	—	—	0	7	0.00	0	7	1.73	0	9	-2.34

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	12.288			14			14.7456			16		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	217	0.08	2	248	-0.17	3	64	0.70	3	70	0.03
150	2	159	0.00	2	181	0.16	2	191	0.00	2	207	0.16
300	2	79	0.00	2	90	0.16	2	95	0.00	2	103	0.16
600	1	159	0.00	1	181	0.16	1	191	0.00	1	207	0.16
1200	1	79	0.00	1	90	0.16	1	95	0.00	1	103	0.16
2400	0	159	0.00	0	181	0.16	0	191	0.00	0	207	0.16
4800	0	79	0.00	0	90	0.16	0	95	0.00	0	103	0.16
9600	0	39	0.00	0	45	-0.93	0	47	0.00	0	51	0.16
19200	0	19	0.00	0	22	-0.93	0	23	0.00	0	25	0.16
31250	0	11	2.40	0	13	0.00	0	14	-1.70	0	15	0.00
38400	0	9	0.00	—	—	—	0	11	0.00	0	12	0.16

Note: For SCI\_2, the table shows the examples for the case when the ABCS bit in SEMR\_2 is cleared to 0. When the ABCS bit is set to 1, the bit rates are doubled.

**Table 16.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (2)**

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	17.2032			18			19.6608			20		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	75	0.48	3	79	-0.12	3	86	0.31	3	88	-0.25
150	2	223	0.00	2	233	0.16	2	255	0.00	3	64	0.16
300	2	111	0.00	2	116	0.16	2	127	0.00	2	129	0.16
600	1	223	0.00	1	233	0.16	1	255	0.00	2	64	0.16
1200	1	111	0.00	1	116	0.16	1	127	0.00	1	129	0.16
2400	0	223	0.00	0	233	0.16	0	255	0.00	1	64	0.16
4800	0	111	0.00	0	116	0.16	0	127	0.00	0	129	0.16
9600	0	55	0.00	0	58	-0.69	0	63	0.00	0	64	0.16
19200	0	27	0.00	0	28	1.02	0	31	0.00	0	32	-1.36
31250	0	16	1.20	0	17	0.00	0	19	-1.70	0	19	0.00
38400	0	13	0.00	0	14	-2.34	0	15	0.00	0	15	1.73

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	25			30			33			35		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	110	-0.02	3	132	0.13	3	145	0.33	3	154	0.23
150	3	80	0.47	3	97	-0.35	3	106	0.39	3	113	-0.06
300	2	162	-0.15	2	194	0.16	2	214	-0.07	2	227	-0.06
600	2	80	0.47	2	97	-0.35	2	106	0.39	2	113	-0.06
1200	1	162	-0.15	1	194	0.16	1	214	-0.07	1	227	-0.06
2400	1	80	0.47	1	97	-0.35	1	106	0.39	1	113	-0.06
4800	0	162	-0.15	0	194	0.16	0	214	-0.07	0	227	-0.06
9600	0	80	0.47	0	97	-0.35	0	106	0.39	0	113	-0.06
19200	0	40	-0.76	0	48	-0.35	0	53	-0.54	0	56	-0.06
31250	0	24	0.00	0	29	0	0	32	0	0	34	0.00
38400	0	19	1.73	0	23	1.73	0	26	-0.54	0	27	-1.73

Note: For SCI\_2, the table shows the examples for the case when the ABCS bit in SEMR\_2 is cleared to 0. When the ABCS bit is set to 1, the bit rates are doubled.

**Table 16.4 Maximum Bit Rate for Each Operating Frequency (Asynchronous Mode)**

<b>P<math>\phi</math> (MHz)</b>	<b>Maximum Bit Rate (bit/s)</b>	<b>n</b>	<b>N</b>
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
30	937500	0	0
33	1031250	0	0
35	1093750	0	0

Note: For SCI\_2, the table shows the examples for the case when the ABCS bit in SEMR\_2 is cleared to 0. When the ABCS bit is set to 1, the bit rates are doubled.



**Table 16.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)**

<b>P<math>\phi</math> (MHz)</b>	<b>External Input Clock (MHz)</b>	<b>Maximum Bit Rate (bit/s)</b>
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
30	7.5000	468750
33	8.2500	515625
35	8.7500	546875

Note: For SCI\_2, the table shows the examples for the case when the ABCS bit in SEMR\_2 is cleared to 0. When the ABCS bit is set to 1, the bit rates are doubled.

**Table 16.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)**

Bit Rate (bit/s)	Operating Frequency P <sub>φ</sub> (MHz)							
	8		10		16		20	
	n	N	n	N	n	N	n	N
110								
250	3	124	—	—	3	249		
500	2	249	—	—	3	124	—	—
1k	2	124	—	—	2	249	—	—
2.5k	1	199	1	249	2	99	2	124
5k	1	99	1	124	1	199	1	249
10k	0	199	0	249	1	99	1	124
25k	0	79	0	99	0	159	0	199
50k	0	39	0	49	0	79	0	99
100k	0	19	0	24	0	39	0	49
250k	0	7	0	9	0	15	0	19
500k	0	3	0	4	0	7	0	9
1M	0	1			0	3	0	4
2.5M			0	0*			0	1
5M							0	0*

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)							
	25		30		33		35	
	n	N	n	N	n	N	n	N
110								
250								
500			3	233				
1k	3	97	3	116	3	128	3	136
2.5k	2	155	2	187	2	205	2	218
5k	2	77	2	93	2	102	2	108
10k	1	155	1	187	1	205	1	218
25k	0	249	1	74	1	82	1	87
50k	0	124	0	149	0	164	0	174
100k	0	62	0	74	0	82	0	87
250k	0	24	0	29	0	32	0	34
500k	—	—	0	14	—	—	—	—
1M	—	—	—	—	—	—	—	—
2.5M	—	—	0	2	—	—	—	—
5M	—	—	—	—	—	—	—	—

[Legend]

Space: Setting prohibited.

—: Can be set, but there will be error.

\*: Continuous transmission or reception is not possible.

**Table 16.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)**

P $\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	P $\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	1.3333	1333333.3	20	3.3333	3333333.3
10	1.6667	1666666.7	25	4.1667	4166666.7
12	2.0000	2000000.0	30	5.0000	5000000.0
14	2.3333	2333333.3	33	5.5000	5500000.0
16	2.6667	2666666.7	35	5.8336	5833625.0
18	3.0000	3000000.0			

**Table 16.8 BRR Settings for Various Bit Rates**  
(Smart Card Interface Mode, n = 0, S = 372)

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	7.1424			10.00			10.7136			13.00		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	0	0.00	0	1	30	0	1	25	0	1	8.99

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	14.2848			16.00			18.00			20.00		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	1	0.00	0	1	12.01	0	2	15.99	0	2	6.66

Bit Rate (bit/s)	Operating Frequency P $\phi$ (MHz)											
	25.00			30.00			33.00			35.00		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	3	12.49	0	3	5.01	0	4	7.59	0	4	1.99

**Table 16.9 Maximum Bit Rate for Each Operating Frequency**  
(Smart Card Interface Mode, S = 372)

P $\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N	P $\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N
7.1424	9600	0	0	18.00	24194	0	0
10.00	13441	0	0	20.00	26882	0	0
10.7136	14400	0	0	25.00	33602	0	0
13.00	17473	0	0	30.00	40323	0	0
14.2848	19200	0	0	33.00	44355	0	0
16.00	21505	0	0	35.00	47043	0	0

### 16.3.10 Serial Extended Mode Register\_2 (SEMR\_2)

SEMR\_2 selects the clock source for SCI\_2 in asynchronous mode. The basic clock is automatically specified when the average transfer rate operation is selected.

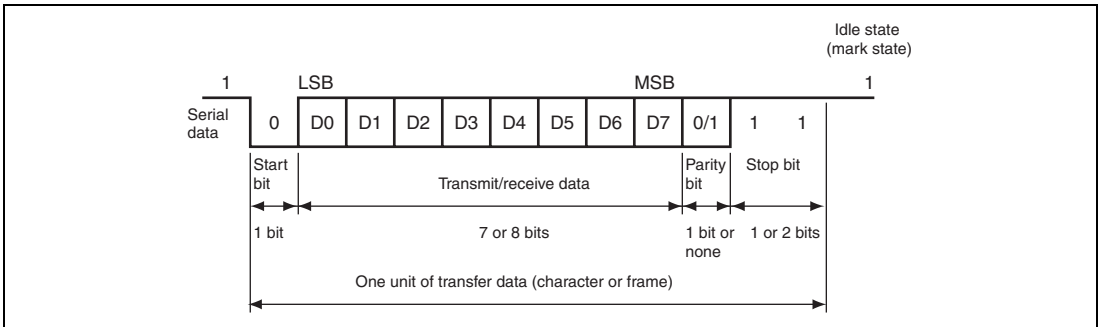
Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	ABCS	ACS2	ACS1	ACS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6 to 4	—	All 0	R	Reserved These are read-only bits and cannot be modified.
3	ABCS	0	R/W	Asynchronous Mode Basic Clock Select (valid only in asynchronous mode) Selects the basic clock. 0: The basic clock has a frequency 16 times the transfer rate 1: The basic clock has a frequency 8 times the transfer rate

Bit	Bit Name	Initial Value	R/W	Description
2	ACS2	0	R/W	Asynchronous Mode Clock Source Select (valid when CKE1 = 1 in asynchronous mode)
1	ACS1	0	R/W	These bits select the clock source for the average transfer rate function. When the average transfer rate function is enabled, the basic clock is automatically specified regardless of the ABCS bit value.  000: External clock input  001: 115.152 kbps of average transfer rate specific to $P\phi = 10.667$ MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate)  010: 460.606 kbps of average transfer rate specific to $P\phi = 10.667$ MHz is selected (operated using the basic clock with a frequency 8 times the transfer rate)  011: 720 kbps of average transfer rate specific to $P\phi = 32$ MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate)  100: Setting prohibited  101: 115.196 kbps of average transfer rate specific to $P\phi = 16$ MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate)  110: 460.784 kbps of average transfer rate specific to $P\phi = 16$ MHz is selected (operated using the basic clock with a frequency 16 times the transfer rate)  111: 720 kbps of average transfer rate specific to $P\phi = 16$ MHz is selected (operated using the basic clock with a frequency 8 times the transfer rate)  The average transfer rate only supports operating frequencies of 10.667 MHz, 16 MHz, and 32 MHz.
0	ACS0	0	R/W	

## 16.4 Operation in Asynchronous Mode

Figure 16.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transmit/receive data, a parity bit, and finally stop bits (high level). 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. 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 transmission and reception.



**Figure 16.2 Data Format in Asynchronous Communication  
(Example with 8-Bit Data, Parity, Two Stop Bits)**

### 16.4.1 Data Transfer Format

Table 16.10 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. For details on the multiprocessor bit, see section 16.5, Multiprocessor Communication Function.

**Table 16.10 Serial Transfer Formats (Asynchronous Mode)**

SMR Settings				Serial Transmit/Receive 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



### 16.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the bit rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Since receive data is sampled at the rising edge of the 8th pulse of the basic clock, data is latched at the middle of each bit, as shown in figure 16.3. Thus the reception margin in asynchronous mode is determined by formula (1) below.

$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100 \quad [\%] \quad \dots \quad \text{Formula (1)}$$

M: Reception margin

N: Ratio of bit rate to clock (N = 16)

D: Duty cycle of clock (D = 0.5 to 1.0)

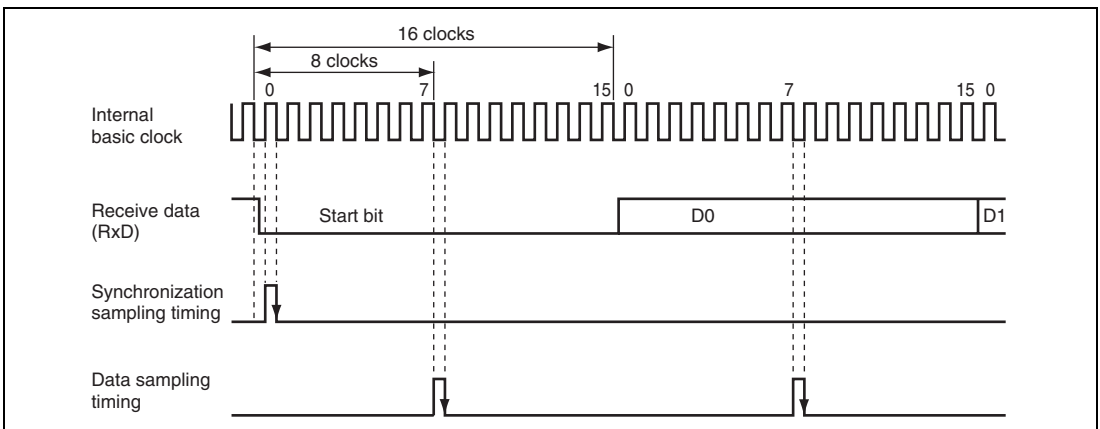
L: Frame length (L = 9 to 12)

F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in formula (1), the reception margin is determined by the formula below.

$$M = \left( 0.5 - \frac{1}{2 \times 16} \right) \times 100 [\%] = 46.875\%$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.



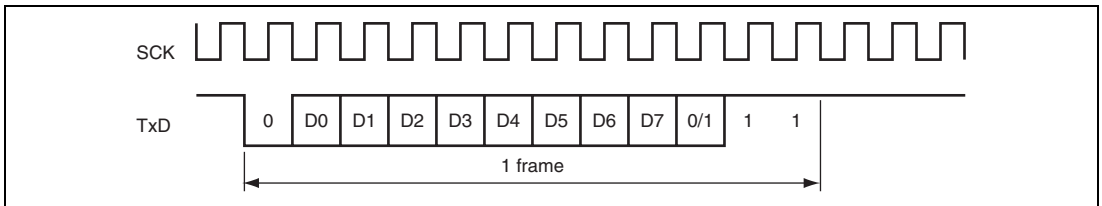
**Figure 16.3 Receive Data Sampling Timing in Asynchronous Mode**

Note: For SCI\_2, the above description shows the example for the case when the ABCS bit in SEMR\_2 is cleared to 0. When the ABCS bit is set to 1, the basic clock has 8 times the frequency of the bit rate and the received data are sampled on the fourth rising edge of the basic clock.

### 16.4.3 Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input to the SCK pin can be selected as the SCI's transfer clock, according to the setting of the C/A bit in SMR and the CKE1 and CKE0 bits in SCR. When an external clock is input to the SCK pin, the clock frequency should be 16 times the bit rate used.

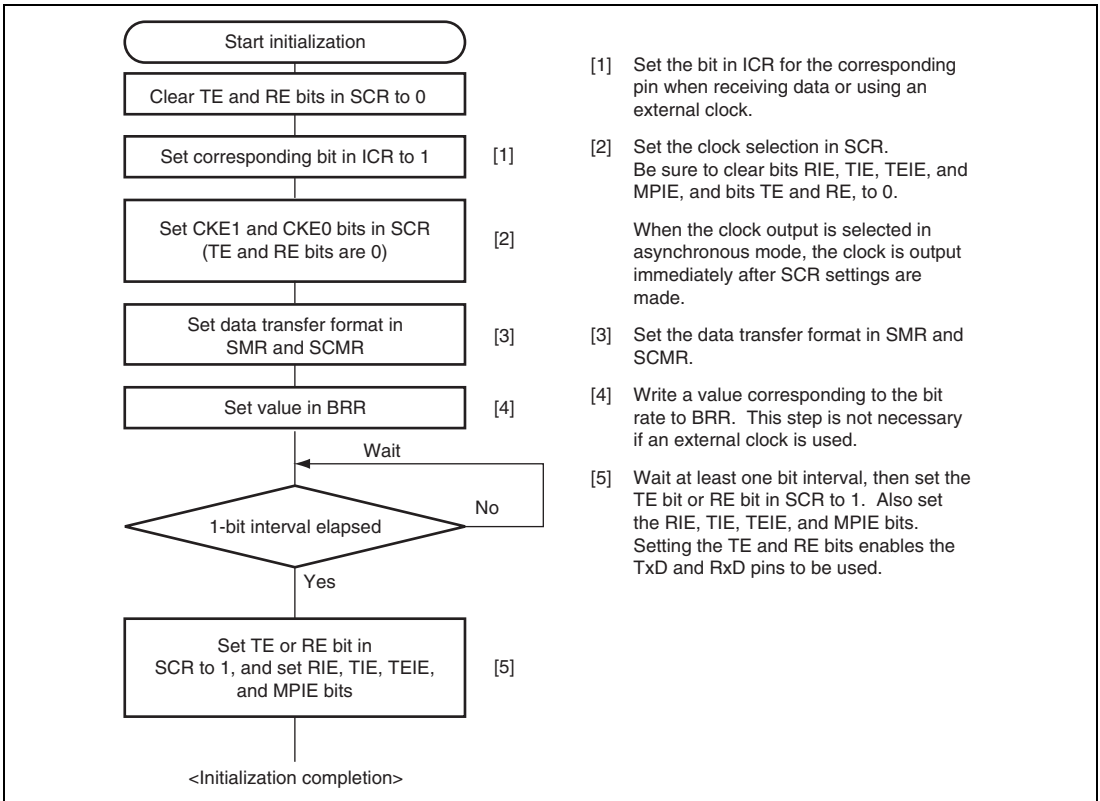
When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 16.4.



**Figure 16.4 Phase Relation between Output Clock and Transmit Data (Asynchronous Mode)**

### 16.4.4 SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 16.5. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization.



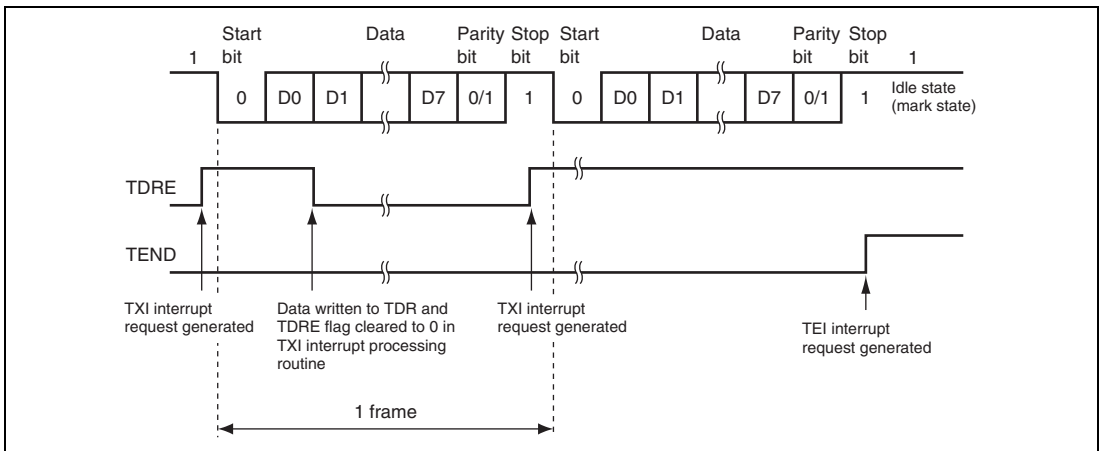
**Figure 16.5 Sample SCI Initialization Flowchart**

### 16.4.5 Serial Data Transmission (Asynchronous Mode)

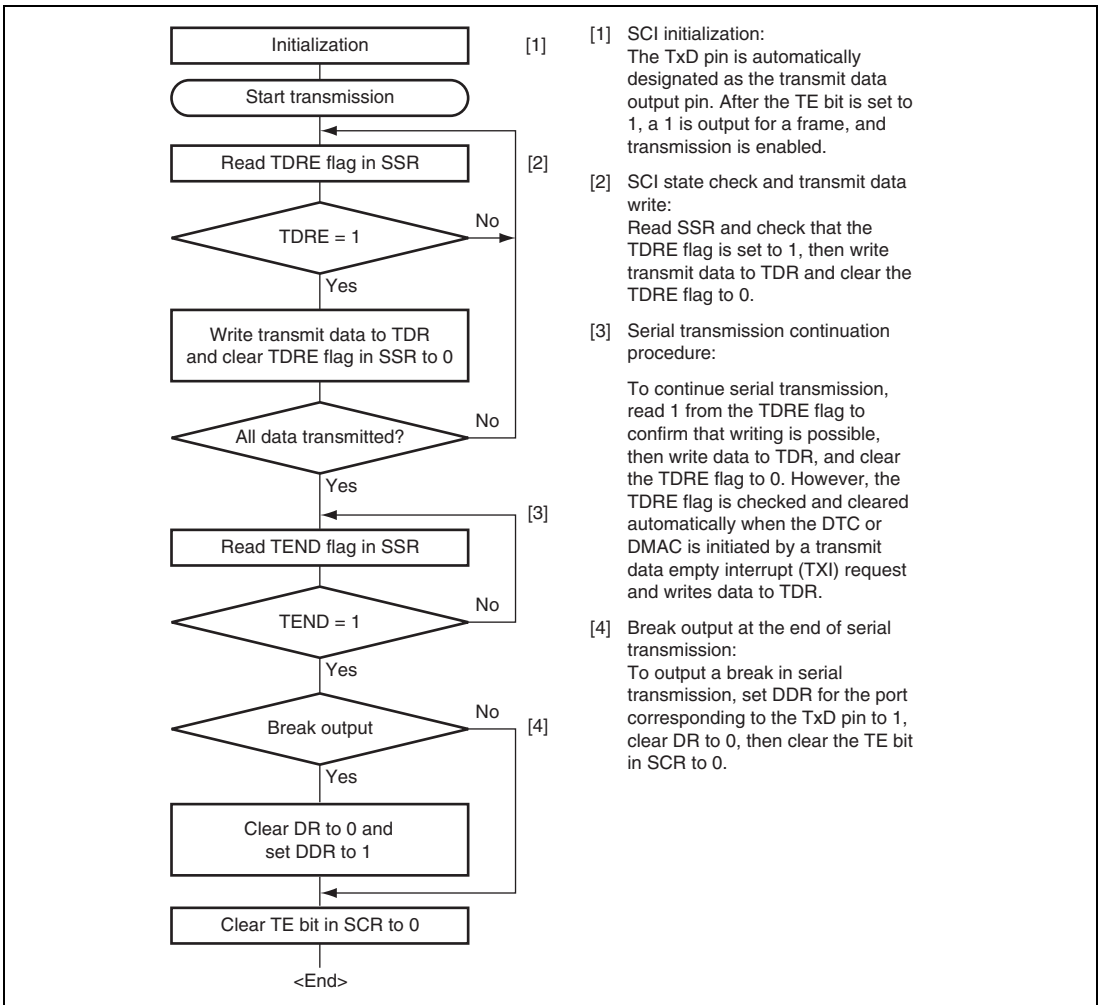
Figure 16.6 shows an example of the operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is cleared to 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 TXIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit or multiprocessor bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the next transmit data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 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. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 16.7 shows a sample flowchart for transmission in asynchronous mode.



**Figure 16.6 Example of Operation for Transmission in Asynchronous Mode  
(Example with 8-Bit Data, Parity, One Stop Bit)**

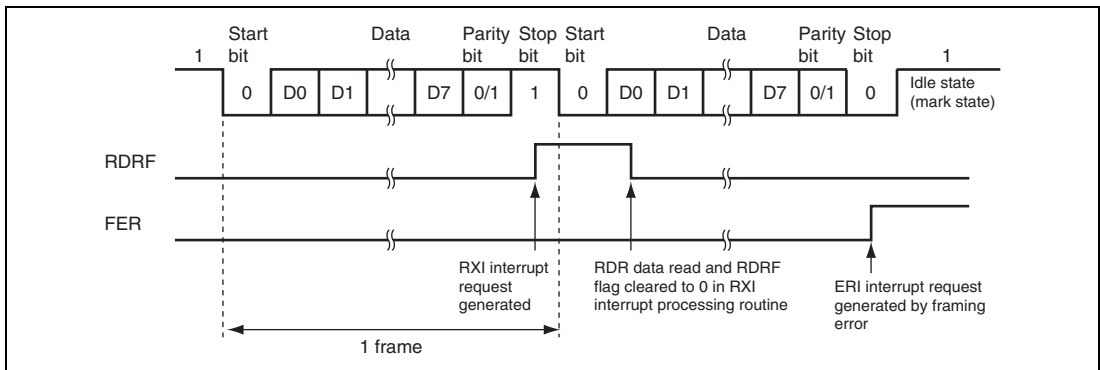


**Figure 16.7 Sample Serial Transmission Flowchart**

### 16.4.6 Serial Data Reception (Asynchronous Mode)

Figure 16.8 shows an example of the operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, stores receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.



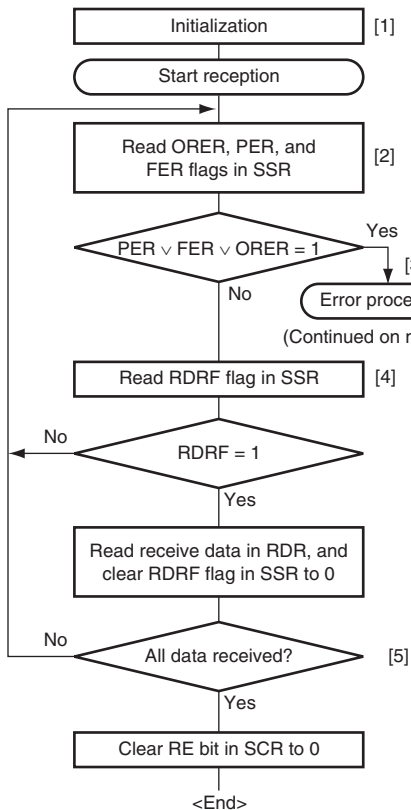
**Figure 16.8 Example of SCI Operation for Reception**  
(Example with 8-Bit Data, Parity, One Stop Bit)

Table 16.11 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 16.9 shows a sample flowchart for serial data reception.

**Table 16.11 SSR Status Flags and Receive Data Handling**

SSR Status Flag				Receive Data	Receive Error Type
RDRF*	ORER	FER	PER		
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

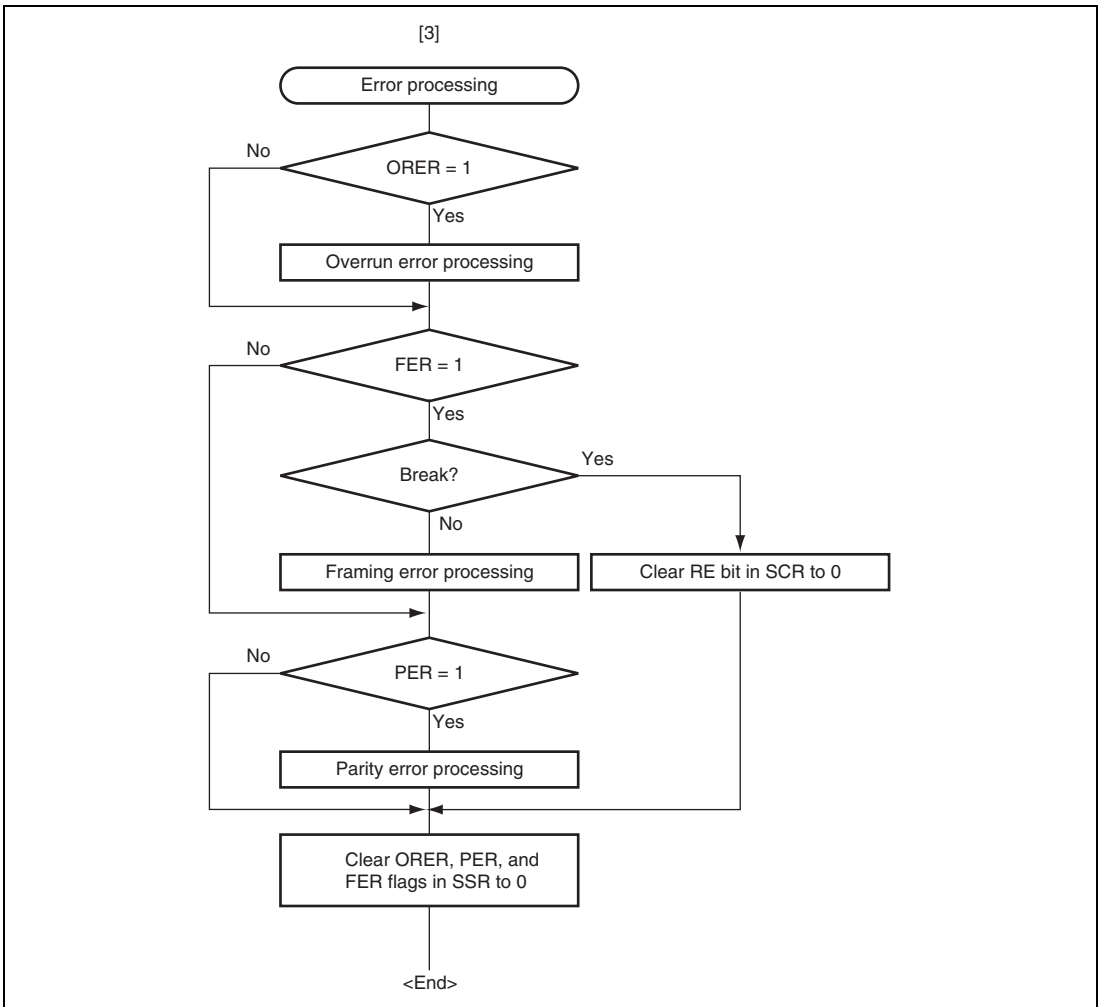
Note: \* The RDRF flag retains the state it had before data reception.



- [1] SCI initialization:  
The RxD pin is automatically designated as the receive data input pin.
- [2] [3] Receive error processing and break detection:  
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.
- [4] SCI state check and receive data read:  
Read SSR and check that RDRF = 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 stop bit for the current frame is received, read the RDRF flag and RDR, and clear the RDRF flag to 0. However, the RDRF flag is cleared automatically when the DTC or DMAC is initiated by an RXI interrupt and reads data from RDR.

**Figure 16.9 Sample Serial Reception Flowchart (1)**





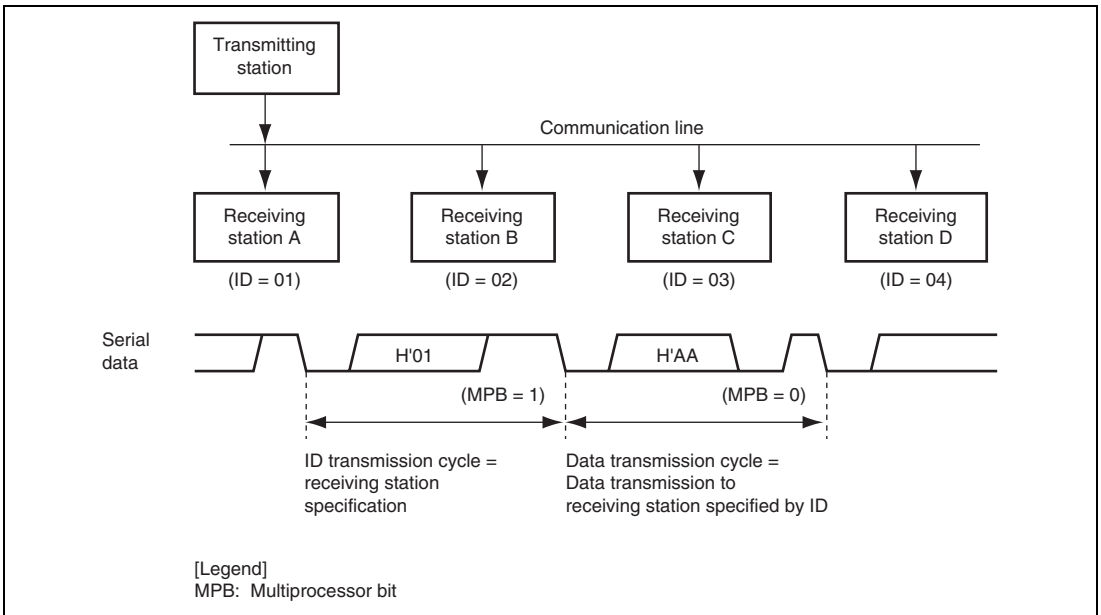
**Figure 16.9 Sample Serial Reception Flowchart (2)**

## 16.5 Multiprocessor Communication Function

Use of the multiprocessor communication function enables data transfer to be performed among a number of processors sharing communication lines by means of asynchronous serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data. 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 for the specified receiving station. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle. If the multiprocessor bit is 1, the cycle is an ID transmission cycle, and if the multiprocessor bit is 0, the cycle is a data transmission cycle. Figure 16.10 shows an example of inter-processor communication using the multiprocessor format. The transmitting station first sends data which includes the ID code of the receiving station and a multiprocessor bit set to 1. It then transmits transmit data added with a multiprocessor bit cleared to 0. The receiving station skips 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 data until data with a 1 multiprocessor bit is again received.

The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set to 1, transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status flags, RDRF, FER, and ORER in SSR to 1 are prohibited until data with a 1 multiprocessor bit is received. On reception of a receive character with a 1 multiprocessor bit, the MPB bit in SSR is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt is generated.

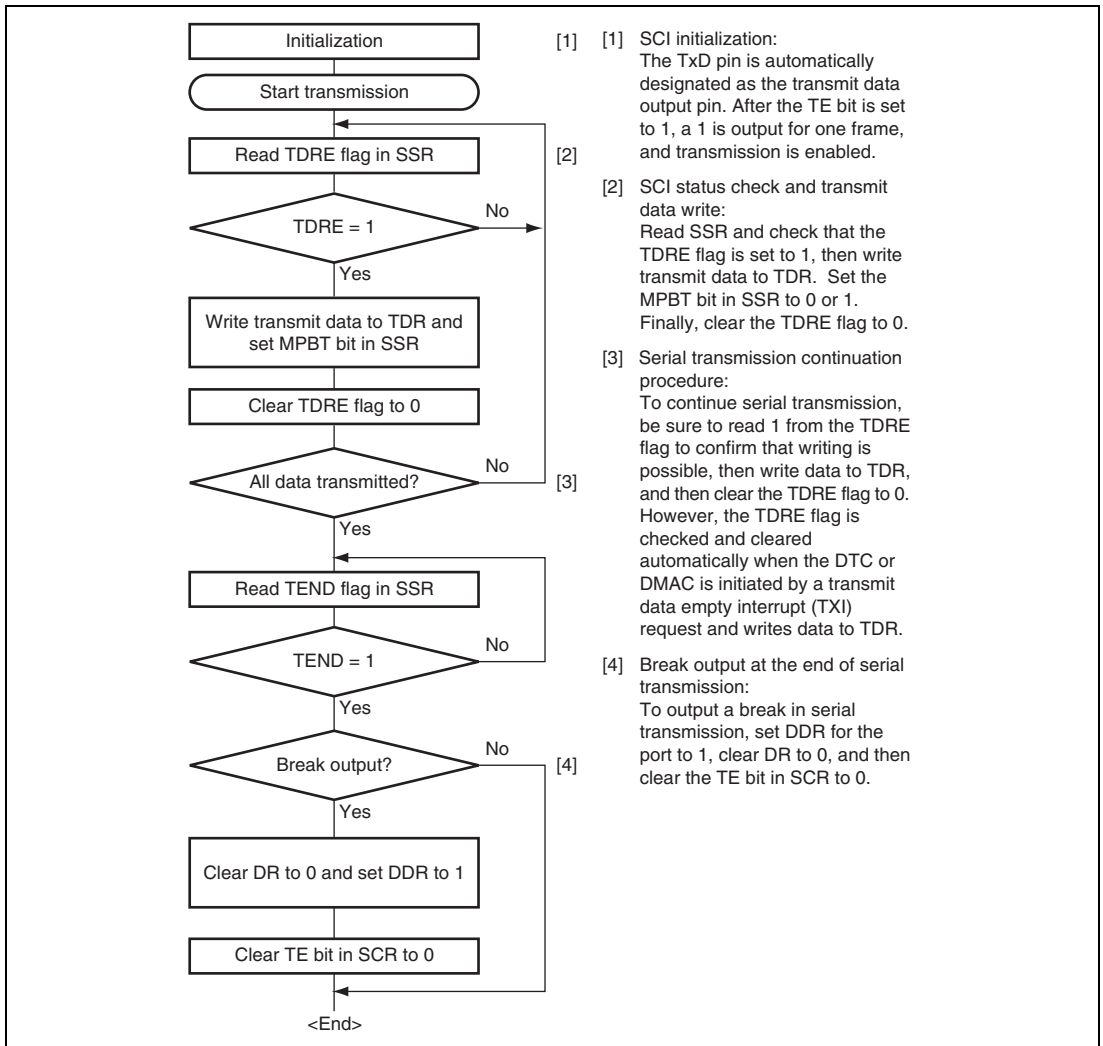
When the multiprocessor format is selected, the parity bit setting is invalid. All other bit settings are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.



**Figure 16.10 Example of Communication Using Multiprocessor Format  
(Transmission of Data H'AA to Receiving Station A)**

### 16.5.1 Multiprocessor Serial Data Transmission

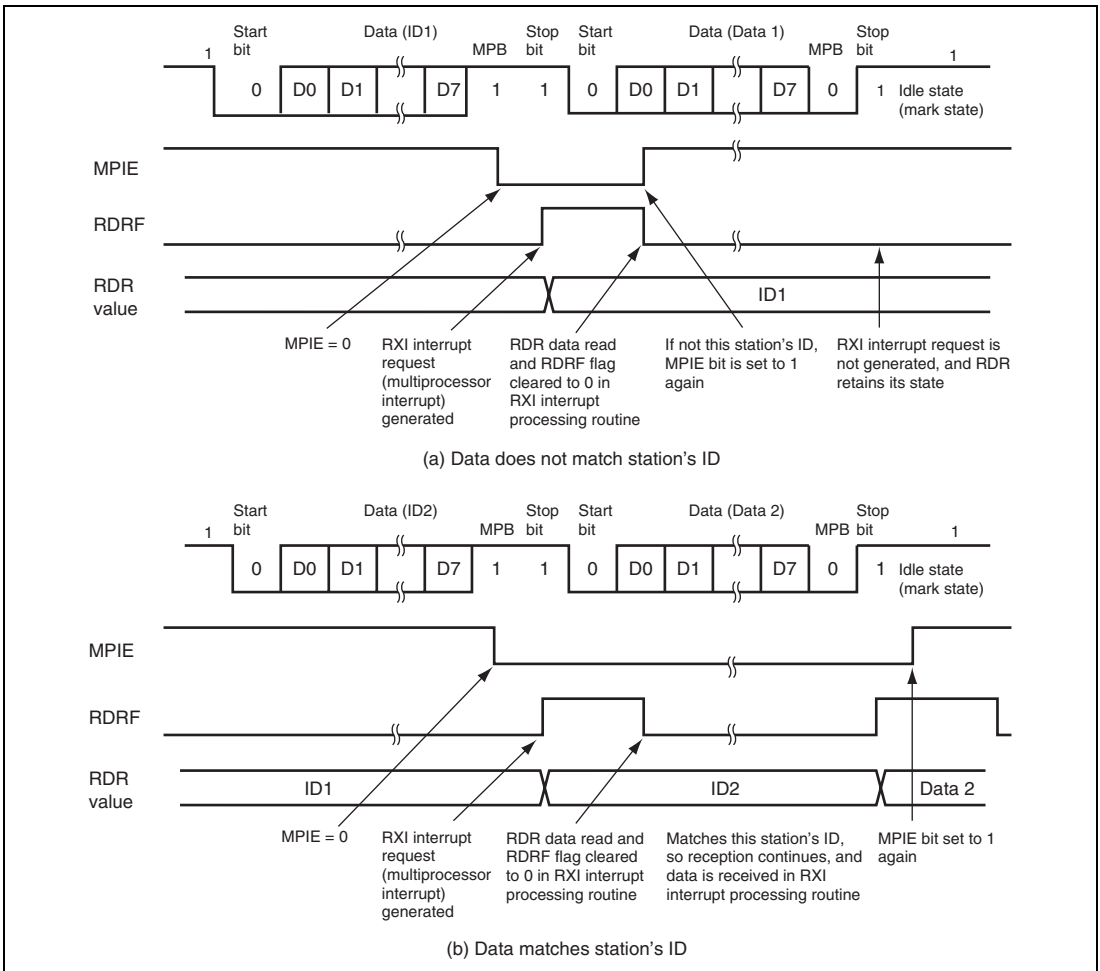
Figure 16.11 shows a sample flowchart for multiprocessor serial data transmission. For an ID transmission cycle, set the MPBT bit in SSR to 1 before transmission. For a data transmission cycle, clear the MPBT bit in SSR to 0 before transmission. All other SCI operations are the same as those in asynchronous mode.



**Figure 16.11 Sample Multiprocessor Serial Transmission Flowchart**

## 16.5.2 Multiprocessor Serial Data Reception

Figure 16.13 shows a sample flowchart for multiprocessor serial data reception. If the MPIE bit in SCR is set to 1, data is skipped until data with a 1 multiprocessor bit is sent. On receiving data with a 1 multiprocessor bit, the receive data is transferred to RDR. An RXI interrupt request is generated at this time. All other SCI operations are the same as in asynchronous mode. Figure 16.12 shows an example of SCI operation for multiprocessor format reception.



**Figure 16.12 Example of SCI Operation for Reception  
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

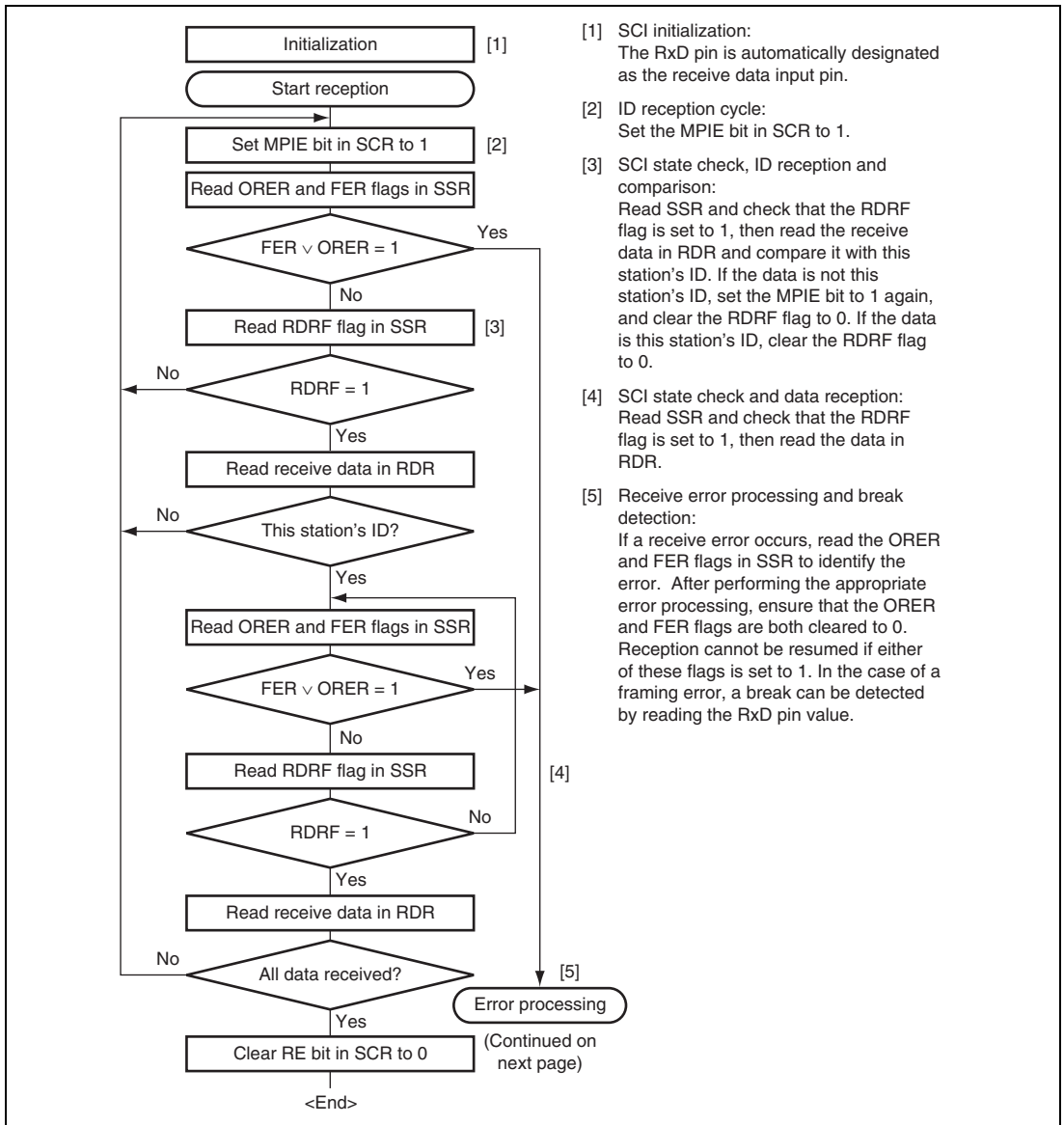
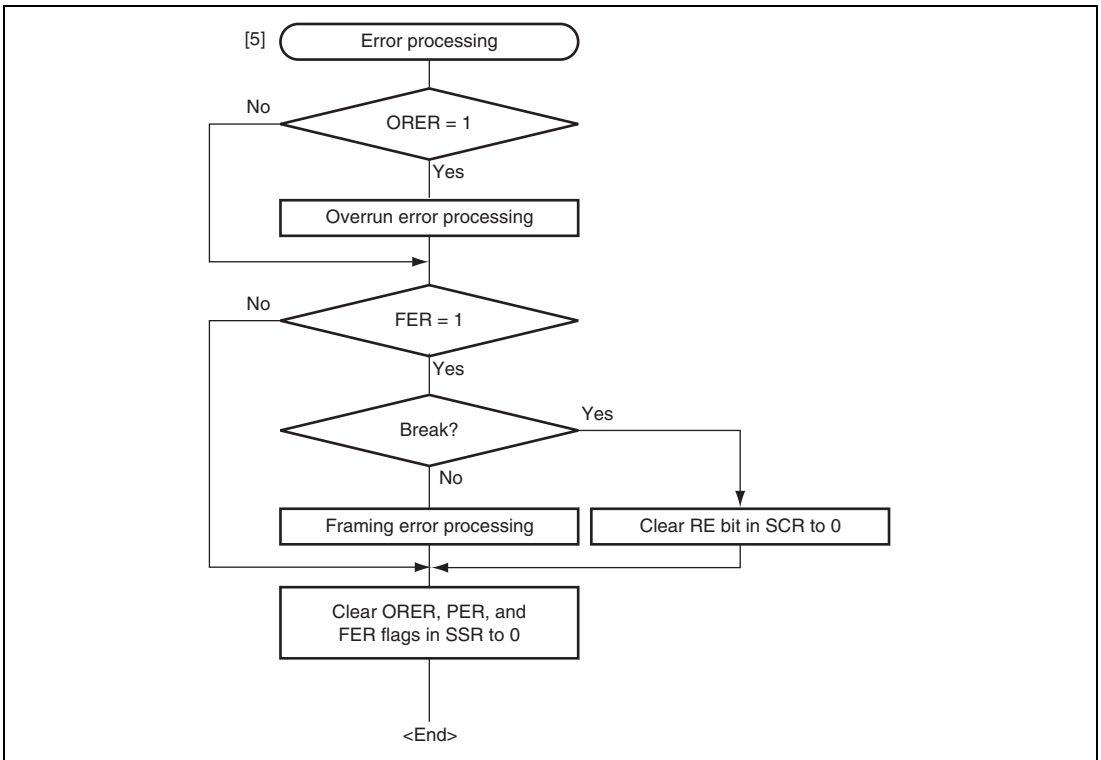


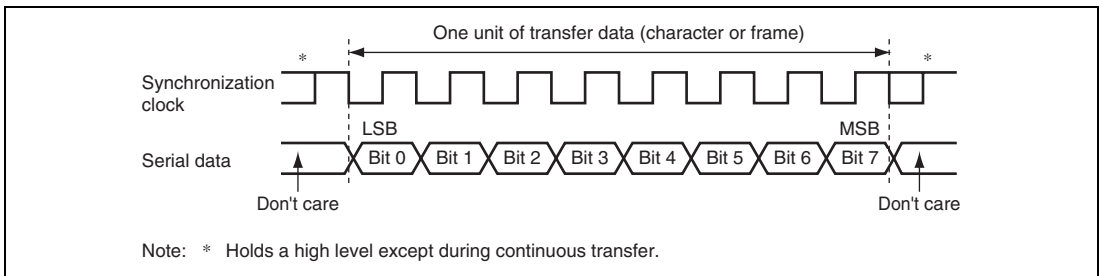
Figure 16.13 Sample Multiprocessor Serial Reception Flowchart (1)



**Figure 16.13 Sample Multiprocessor Serial Reception Flowchart (2)**

## 16.6 Operation in Clocked Synchronous Mode

Figure 16.14 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses. One character in transfer data consists of 8-bit data. In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the MSB output state. In clocked synchronous mode, no parity bit or multiprocessor bit is added. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that the next transmit data can be written during transmission or the previous receive data can be read during reception, enabling continuous data transfer.



**Figure 16.14 Data Format in Clocked Synchronous Communication (LSB-First)**

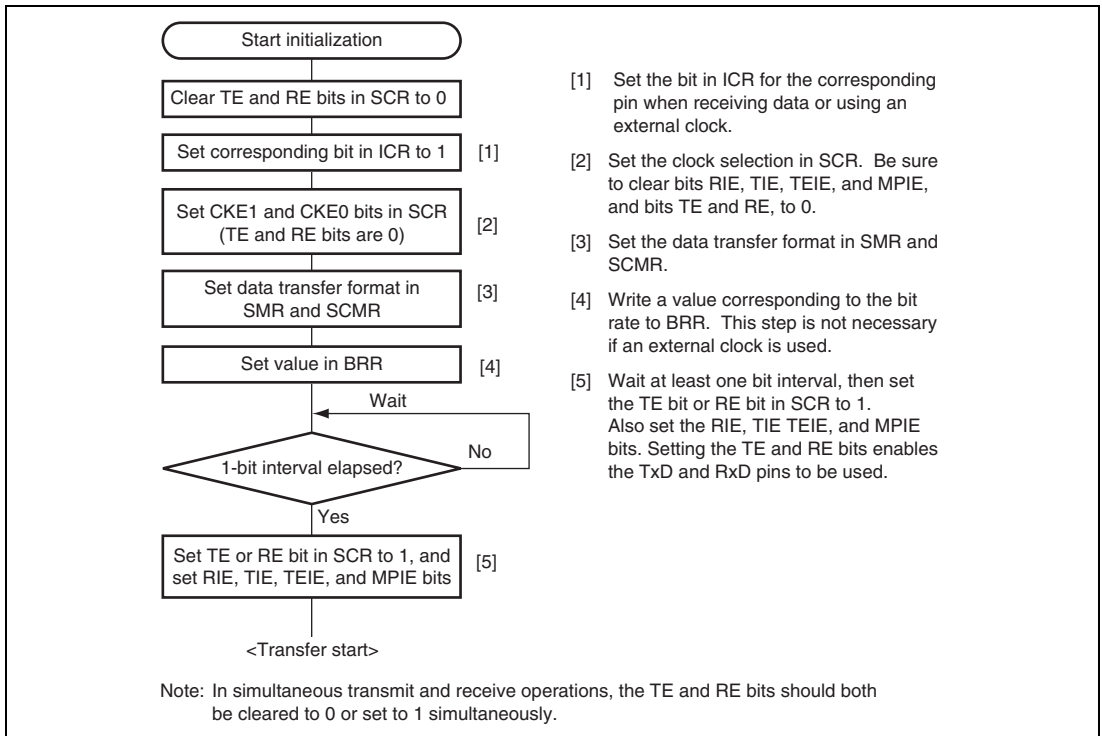
### 16.6.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCR. When the SCI is operated on an internal clock, the synchronization clock is output from the SCK pin. Eight synchronization clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. Note that in the case of reception only, the synchronization clock is output until an overrun error occurs or until the RE bit is cleared to 0.



## 16.6.2 SCI Initialization (Clocked Synchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 16.15. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. However, clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR.



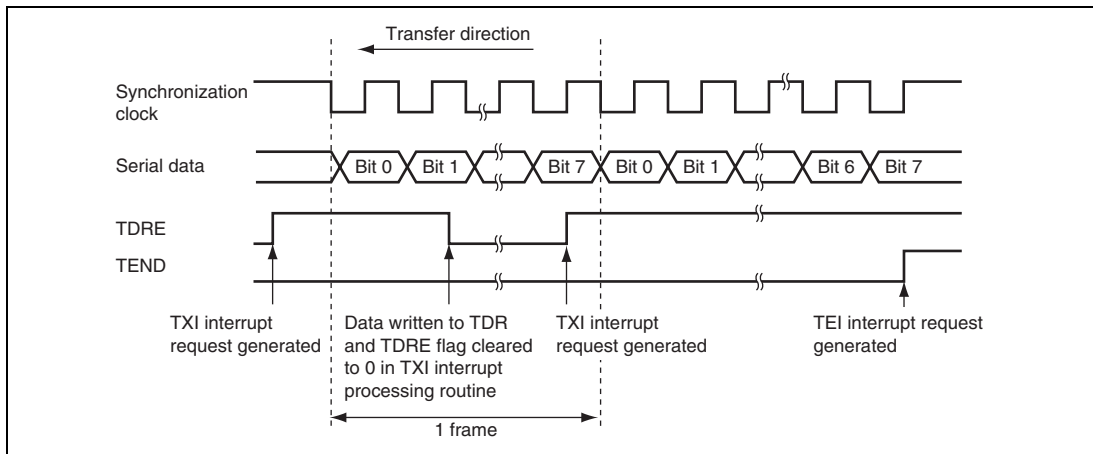
**Figure 16.15 Sample SCI Initialization Flowchart**

### 16.6.3 Serial Data Transmission (Clocked Synchronous Mode)

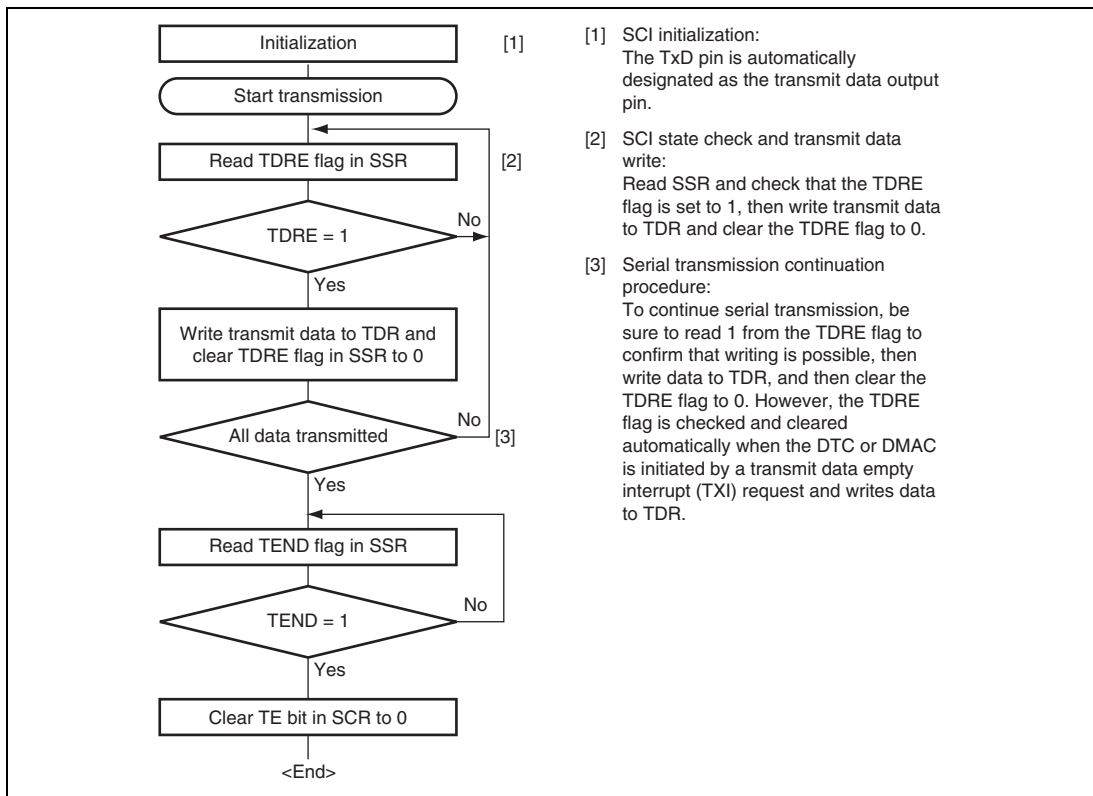
Figure 16.16 shows an example of the operation for transmission in clocked synchronous mode. In transmission, the SCI operates as described below.

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 in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. 8-bit data is sent from the TxD pin synchronized with the output clock when clock output mode has been specified and synchronized with the input clock when use of an external clock has been specified.
4. The SCI checks the TDRE flag at the timing for sending the last bit.
5. If the TDRE flag is cleared to 0, the next transmit data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TxD pin retains the output state of the last bit. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated. The SCK pin is fixed high.

Figure 16.17 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.



**Figure 16.16 Example of Operation for Transmission in Clocked Synchronous Mode**

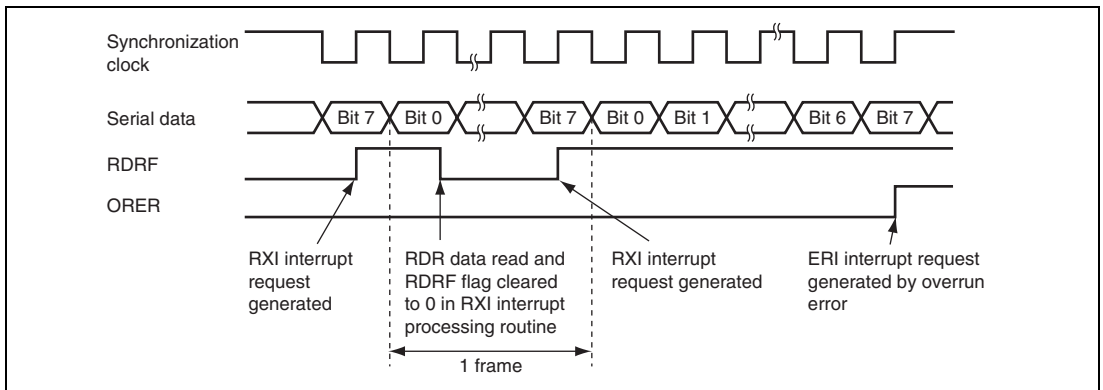


**Figure 16.17 Sample Serial Transmission Flowchart**

### 16.6.4 Serial Data Reception (Clocked Synchronous Mode)

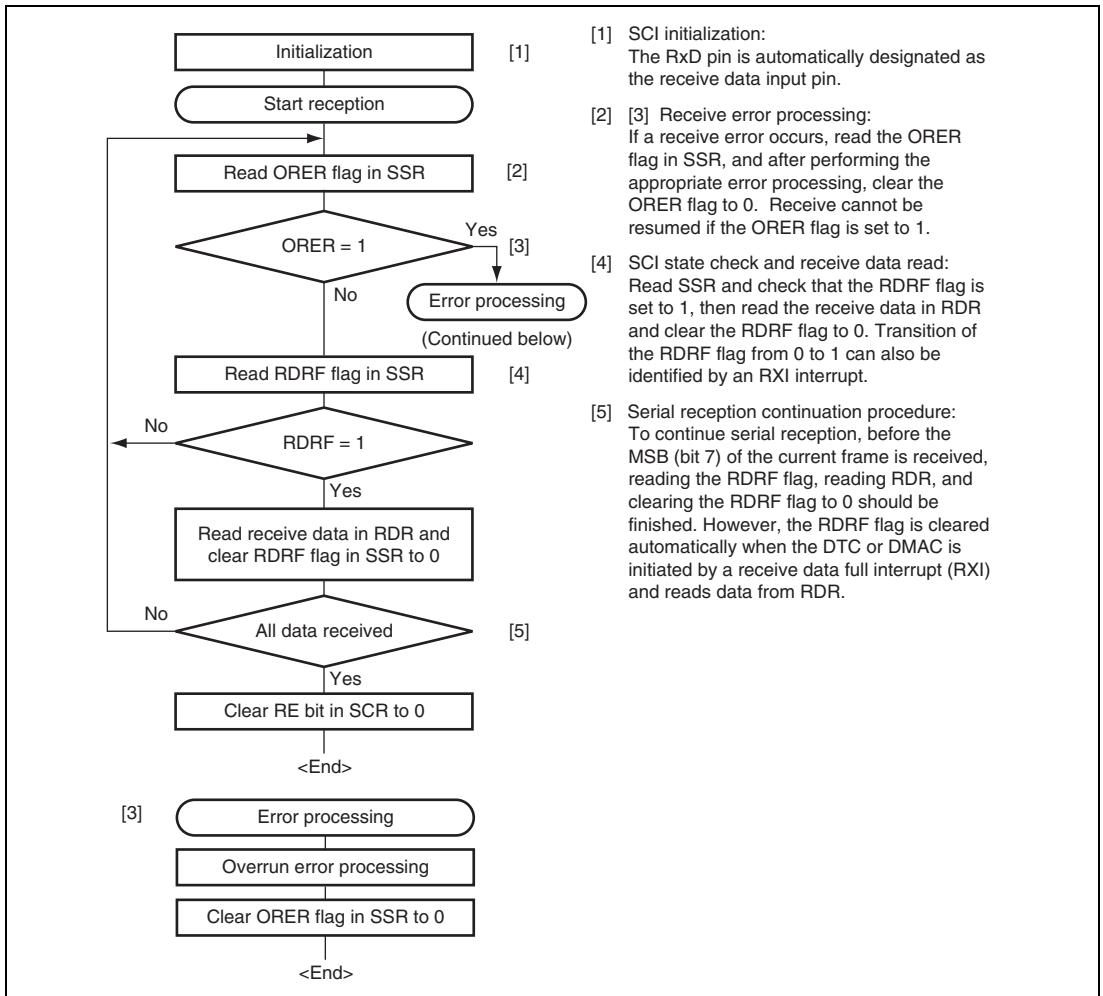
Figure 16.18 shows an example of SCI operation for reception in clocked synchronous mode. In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with a synchronization clock input or output, starts receiving data, and stores the receive data in RSR.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.



**Figure 16.18 Example of Operation for Reception in Clocked Synchronous Mode**

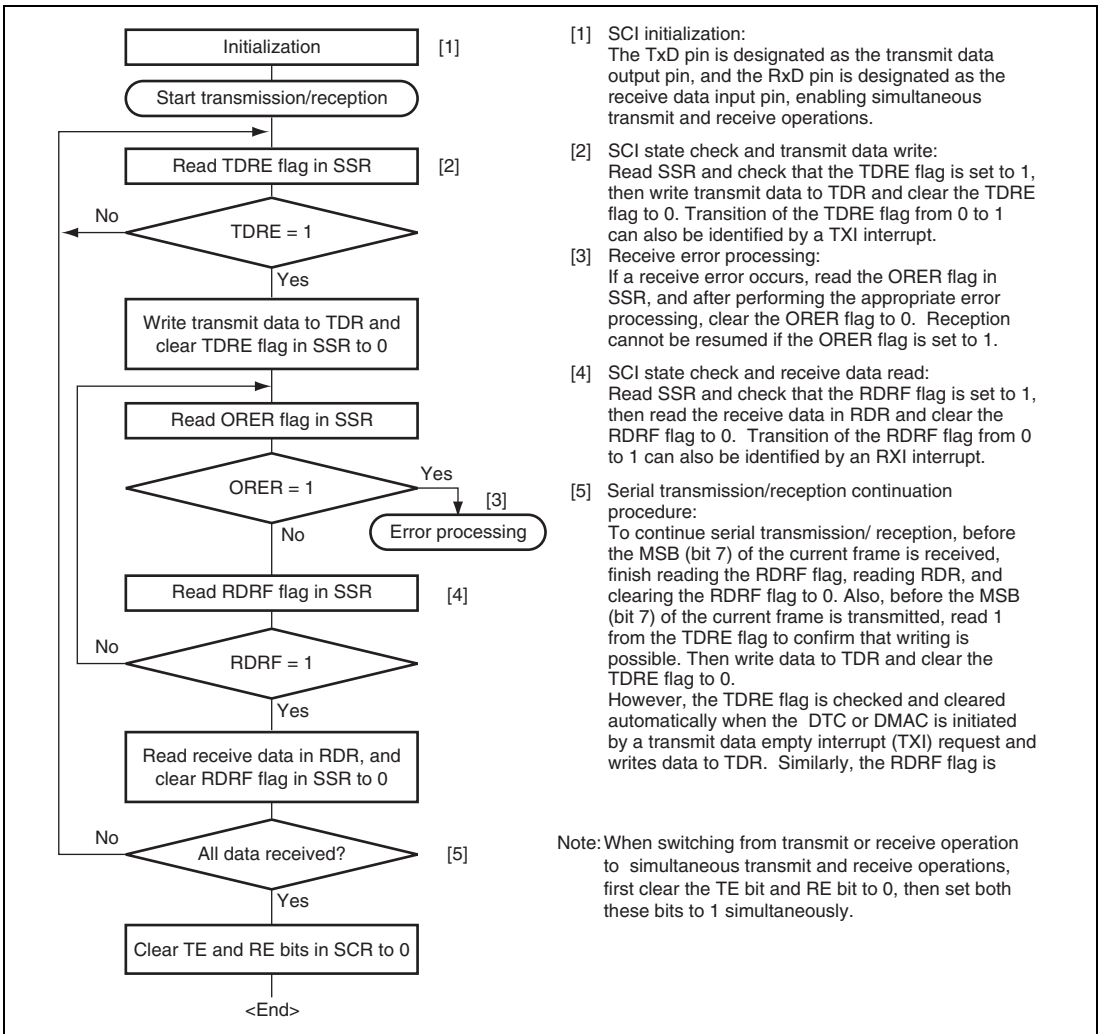
Transfer cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 16.19 shows a sample flowchart for serial data reception.



**Figure 16.19 Sample Serial Reception Flowchart**

### **16.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode)**

Figure 16.20 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags are set to 1, clear the TE bit to 0. Then simultaneously set both the TE and RE bits to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI has finished reception, clear the RE bit to 0. Then after checking that the RDRF bit and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set both the TE and RE bits to 1 with a single instruction.



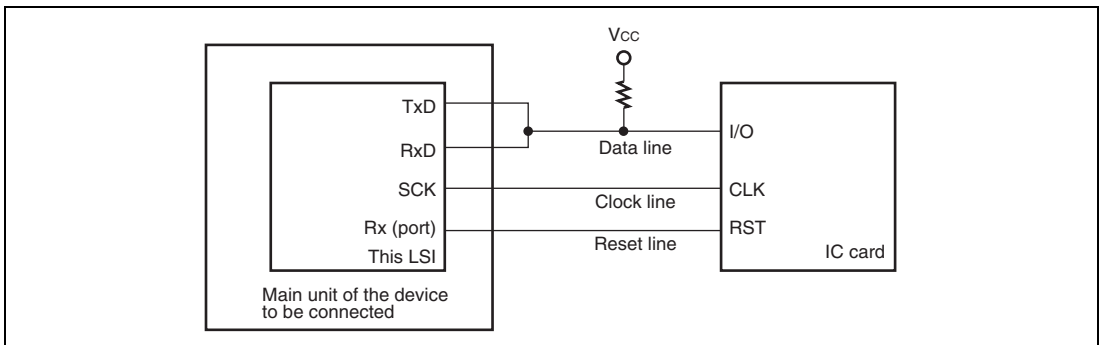
**Figure 16.20 Sample Flowchart of Simultaneous Serial Transmission and Reception**

## 16.7 Operation in Smart Card Interface Mode

The SCI supports the IC card (smart card) interface, supporting the ISO/IEC 7816-3 (Identification Card) standard, as an extended serial communication interface function. Smart card interface mode can be selected using the appropriate register.

### 16.7.1 Sample Connection

Figure 16.21 shows a sample connection between the smart card and this LSI. As in the figure, since this LSI communicates with the IC card using a single transmission line, interconnect the TxD and RxD pins and pull up the data transmission line to  $V_{CC}$  using a resistor. Setting the RE and TE bits to 1 with the IC card not connected enables closed transmission/reception allowing self diagnosis. To supply the IC card with the clock pulses generated by the SCI, input the SCK pin output to the CLK pin of the IC card. A reset signal can be supplied via the output port of this LSI.



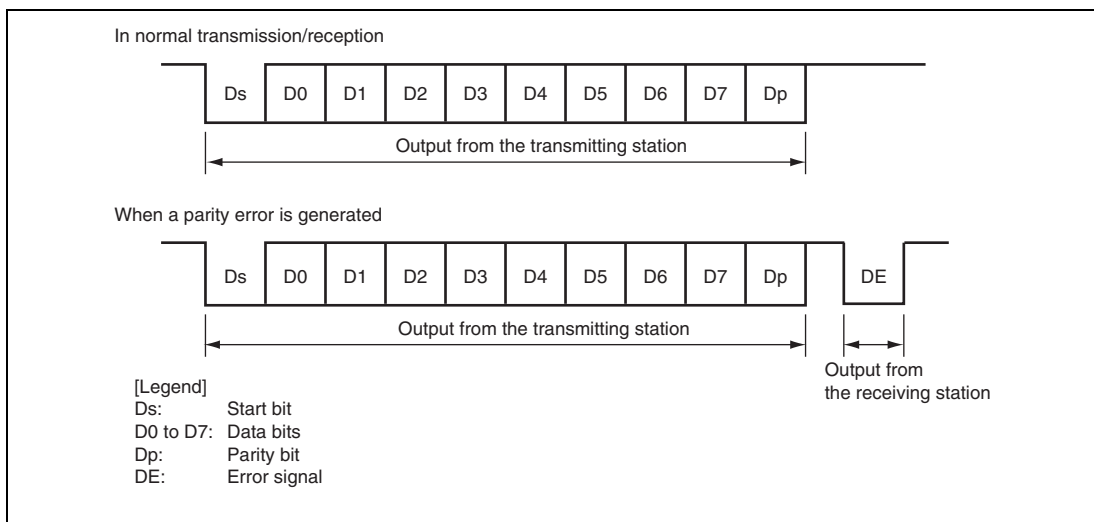
**Figure 16.21 Pin Connection for Smart Card Interface**



## 16.7.2 Data Format (Except in Block Transfer Mode)

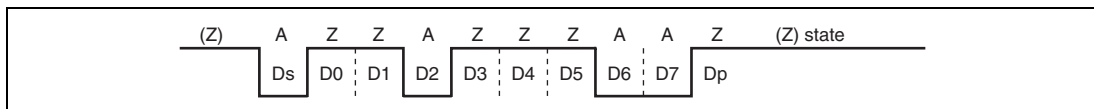
Figure 16.22 shows the data transfer formats in smart card interface mode.

- One frame contains 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etu (elementary time unit: time required for transferring one bit) is secured as a guard time after the end of the parity bit before the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etu has passed from the start bit.
- If an error signal is sampled during transmission, the same data is automatically re-transmitted after at least 2 etu.



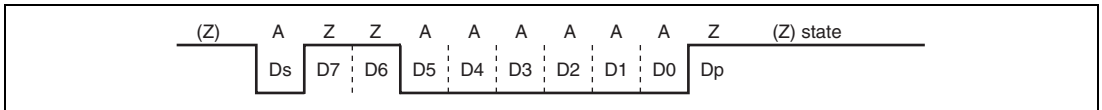
**Figure 16.22 Data Formats in Normal Smart Card Interface Mode**

For communication with the IC cards of the direct convention and inverse convention types, follow the procedure below.



**Figure 16.23 Direct Convention (SDIR = SINV = O/E = 0)**

For the direct convention type, logic levels 1 and 0 correspond to states Z and A, respectively, and data is transferred with LSB-first as the start character, as shown in figure 16.23. Therefore, data in the start character in the figure is H'3B. When using the direct convention type, write 0 to both the SDIR and SINV bits in SCMR. Write 0 to the  $O\bar{E}$  bit in SMR in order to use even parity, which is prescribed by the smart card standard.



**Figure 16.24 Inverse Convention (SDIR = SINV =  $O\bar{E}$  = 1)**

For the inverse convention type, logic levels 1 and 0 correspond to states A and Z, respectively and data is transferred with MSB-first as the start character, as shown in figure 16.24. Therefore, data in the start character in the figure is H'3F. When using the inverse convention type, write 1 to both the SDIR and SINV bits in SCMR. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to state Z. Since the SNIV bit of this LSI only inverts data bits D7 to D0, write 1 to the  $O\bar{E}$  bit in SMR to invert the parity bit in both transmission and reception.

### 16.7.3 Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following respects.

- Even if a parity error is detected during reception, no error signal is output. Since the PER bit in SSR is set by error detection, clear the PER bit before receiving the parity bit of the next frame.
- During transmission, at least 1 etu is secured as a guard time after the end of the parity bit before the start of the next frame.
- Since the same data is not re-transmitted during transmission, the TEND flag is set 11.5 etu after transmission start.
- Although the ERS flag in block transfer mode displays the error signal status as in normal smart card interface mode, the flag is always read as 0 because no error signal is transferred.

### 16.7.4 Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the on-chip baud rate generator can be used as a transfer clock in smart card interface mode. In this mode, the SCI can operate on a basic clock with a frequency of 32, 64, 372, or 256 times the bit rate according to the BCP1 and BCP0 bit settings (the frequency is always 16 times the bit rate in normal asynchronous mode). At reception, the falling edge of the start bit is sampled using the basic clock in order to perform internal synchronization. Receive data is sampled on the 16th, 32nd, 186th and 128th rising edges of the basic clock so that it can be latched at the middle of each bit as shown in figure 16.25. The reception margin here is determined by the following formula.

$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\%$$

M: Reception margin (%)

N: Ratio of bit rate to clock (N = 32, 64, 372, 256)

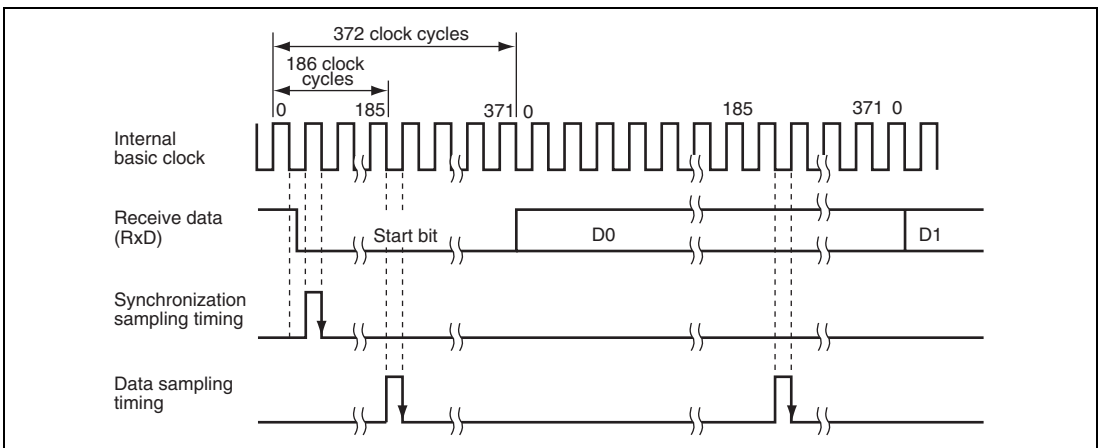
D: Duty cycle of clock (D = 0 to 1.0)

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 reception margin is determined by the formula below.

$$M = \left( 0.5 - \frac{1}{2 \times 372} \right) \times 100\% = 49.866\%$$



**Figure 16.25 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency is 372 Times the Bit Rate)**

### 16.7.5 Initialization

Before transmitting and receiving data, initialize the SCI using the following procedure. Initialization is also necessary before switching from transmission to reception and vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Set the ICR bit of the corresponding pin to 1.
3. Clear the error flags ERS, PER, and ORER in SSR to 0.
4. Set the GM, BLK,  $O/\bar{E}$ , BCP1, BCP0, CKS1, and CKS0 bits in SMR appropriately. Also set the PE bit to 1.
5. Set the SMIF, SDIR, and SINV bits in SCMR appropriately. When the DDR corresponding to the TxD pin is cleared to 0, the TxD and RxD pins are changed from port pins to SCI pins, placing the pins into high impedance state.
6. Set the value corresponding to the bit rate in BRR.
7. Set the CKE1 and CKE0 bits in SCR appropriately. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0 simultaneously.

When the CKE0 bit is set to 1, the SCK pin is allowed to output clock pulses.

8. Set the TIE, RIE, TE, and RE bits in SCR appropriately after waiting for at least a 1-bit interval. Setting the TE and RE bits to 1 simultaneously is prohibited except for self diagnosis.

To switch from reception to transmission, first verify that reception has completed, then initialize the SCI. At the end of initialization, RE and TE should be set to 0 and 1, respectively. Reception completion can be verified by reading the RDRF, PER, or ORER flag. To switch from transmission to reception, first verify that transmission has completed, then initialize the SCI. At the end of initialization, TE and RE should be set to 0 and 1, respectively. Transmission completion can be verified by reading the TEND flag.

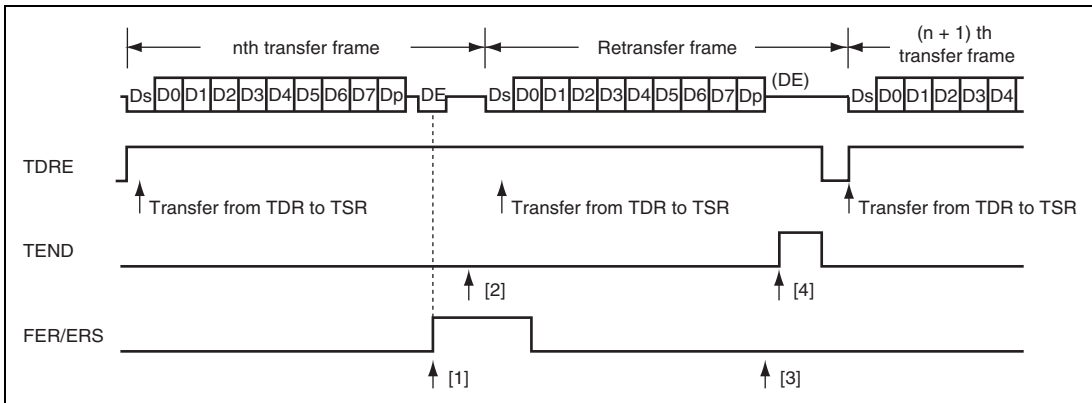
### 16.7.6 Data Transmission (Except in Block Transfer Mode)

Data transmission in smart card interface mode (except in block transfer mode) is different from that in normal serial communication interface mode in that an error signal is sampled and data can be re-transmitted. Figure 16.26 shows the data re-transfer operation during transmission.

1. If an error signal from the receiving end is sampled after one frame of data has been transmitted, the ERS bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the ERS bit to 0 before the next parity bit is sampled.
2. For the frame in which an error signal is received, the TEND bit in SSR is not set to 1. Data is re-transferred from TDR to TSR allowing automatic data retransmission.
3. If no error signal is returned from the receiving end, the ERS bit in SSR is not set to 1.
4. In this case, one frame of data is determined to have been transmitted including re-transfer, and the TEND bit in SSR is set to 1. Here, a TXI interrupt request is generated if the TIE bit in SCR is set to 1. Writing transmit data to TDR starts transmission of the next data.

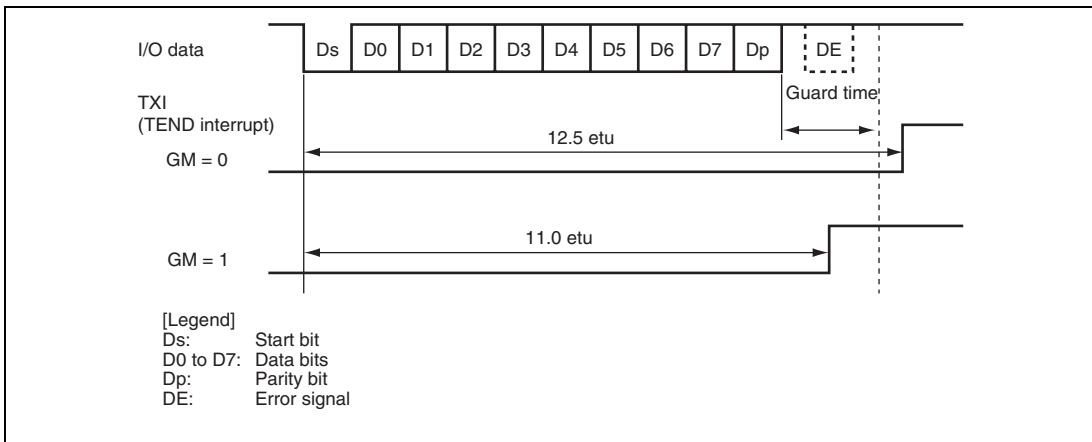
Figure 16.28 shows a sample flowchart for transmission. All the processing steps are automatically performed using a TXI interrupt request to activate the DTC or DMAC. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request if the TIE bit in SCR has been set to 1. This activates the DTC or DMAC by a TXI request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DTC or DMAC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DTC or DMAC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, TEND remains as 0, thus not activating the DTC or DMAC. Therefore, the SCI and DTC or DMAC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DTC or DMAC, be sure to set and enable the DTC or DMAC prior to making SCI settings. For DTC or DMAC settings, see section 9, DMA Controller (DMAC) and section 10, Data Transfer Controller (DTC).

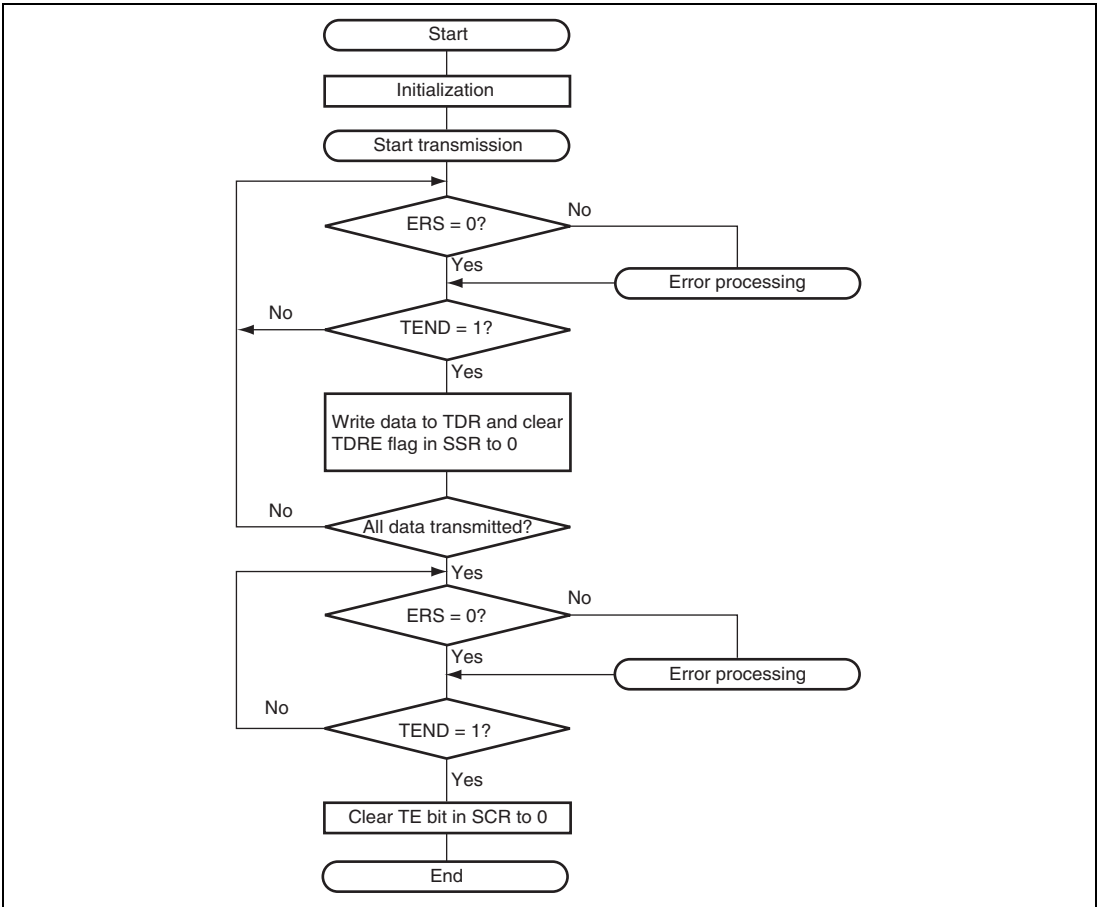


**Figure 16.26 Data Re-Transfer Operation in SCI Transmission Mode**

Note that the TEND flag is set in different timings depending on the GM bit setting in SMR. Figure 16.27 shows the TEND flag set timing.



**Figure 16.27 TEND Flag Set Timing during Transmission**



**Figure 16.28 Sample Transmission Flowchart**

### 16.7.7 Serial Data Reception (Except in Block Transfer Mode)

Data reception in smart card interface mode is similar to that in normal serial communication interface mode. Figure 16.29 shows the data re-transfer operation during reception.

1. If a parity error is detected in receive data, the PER bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the PER bit to 0 before the next parity bit is sampled.
2. For the frame in which a parity error is detected, the RDRF bit in SSR is not set to 1.
3. If no parity error is detected, the PER bit in SSR is not set to 1.
4. In this case, data is determined to have been received successfully, and the RDRF bit in SSR is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set to 1.

Figure 16.30 shows a sample flowchart for reception. All the processing steps are automatically performed using an RXI interrupt request to activate the DTC or DMAC. In reception, setting the RIE bit to 1 allows an RXI interrupt request to be generated when the RDRF flag is set to 1. This activates the DTC or DMAC by an RXI request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DTC or DMAC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DTC or DMAC. If an error occurs during reception, i.e., either the ORER or PER flag is set to 1, a transmit/receive error interrupt (ERI) request is generated and the error flag must be cleared. If an error occurs, the DTC or DMAC is not activated and receive data is skipped, therefore, the number of bytes of receive data specified in the DTC or DMAC is transferred. Even if a parity error occurs and the PER bit is set to 1 in reception, receive data is transferred to RDR, thus allowing the data to be read.

Note: For operations in block transfer mode, see section 16.4, Operation in Asynchronous Mode.

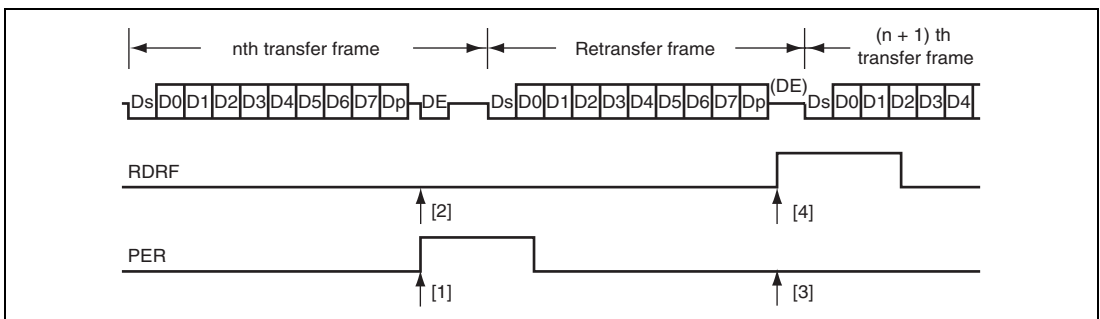
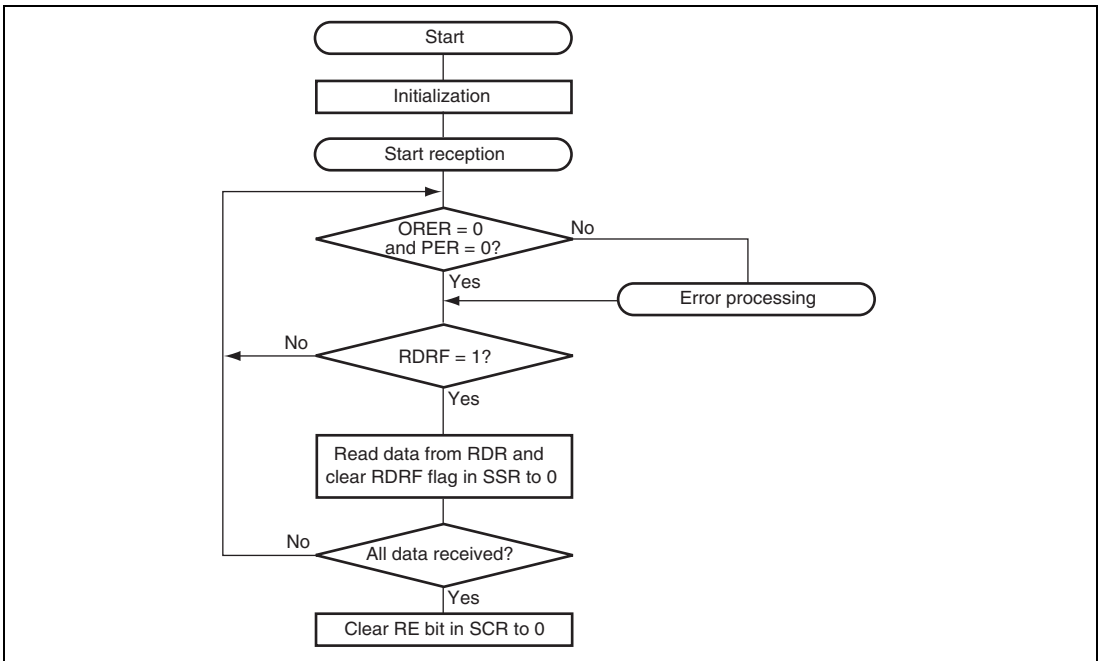


Figure 16.29 Data Re-Transfer Operation in SCI Reception Mode



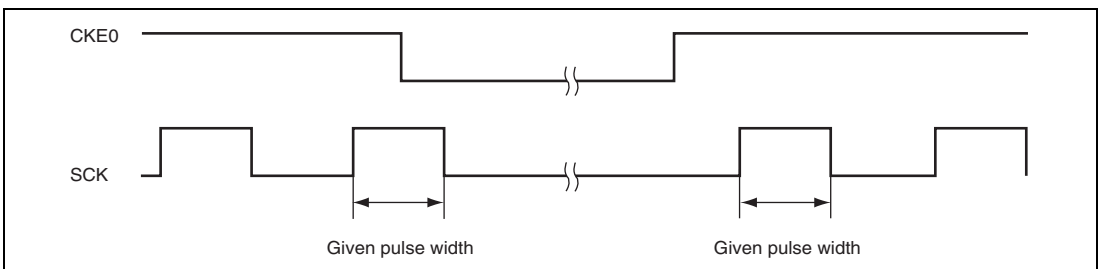


**Figure 16.30 Sample Reception Flowchart**

### 16.7.8 Clock Output Control

Clock output can be fixed using the CKE1 and CKE0 bits in SCR when the GM bit in SMR is set to 1. Specifically, the minimum width of a clock pulse can be specified.

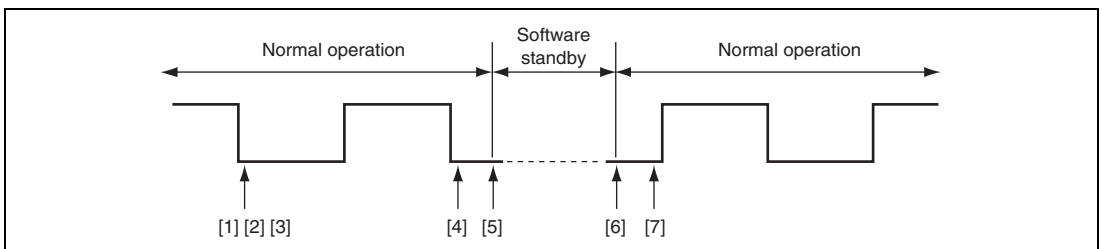
Figure 16.31 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 0.



**Figure 16.31 Clock Output Fixing Timing**

At power-on and transitions to/from software standby mode, use the following procedure to secure the appropriate clock duty cycle.

- At power-on
  - To secure the appropriate clock duty cycle simultaneously with power-on, use the following procedure.
    1. Initially, port input is enabled in the high-impedance state. To fix the potential level, use a pull-up or pull-down resistor.
    2. Fix the SCK pin to the specified output using the CKE1 bit in SCR.
    3. Set SMR and SCMR to enable smart card interface mode.
      - Set the CKE0 bit in SCR to 1 to start clock output.
- At mode switching
  - At transition 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 values for the output fixed state in software standby mode.
    2. Write 0 to the TE and RE bits in SCR to stop transmission/reception. Simultaneously, set the CKE1 bit to the value for the output fixed state in software standby mode.
    3. Write 0 to the CKE0 bit in SCR to stop the clock.
    4. Wait for one cycle of the serial clock. In the mean time, the clock output is fixed to the specified level with the duty cycle retained.
    5. Make the transition to software standby mode.
  - At transition from smart card interface mode to software standby mode
    6. Clear software standby mode.
    7. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty cycle is then generated.



**Figure 16.32 Clock Stop and Restart Procedure**

## 16.8 Interrupt Sources

### 16.8.1 Interrupts in Normal Serial Communication Interface Mode

Table 16.12 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

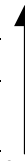
When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt request can activate the DTC or DMAC to allow data transfer. The TDRE flag is automatically cleared to 0 at data transfer by the DTC or DMAC.

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 DTC or DMAC to allow data transfer. The RDRF flag is automatically cleared to 0 at data transfer by the DTC or DMAC.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared to 0 simultaneously by the TXI interrupt processing routine, the SCI cannot branch to the TEI interrupt processing routine later.

**Table 16.12 SCI Interrupt Sources**

Name	Interrupt Source	Interrupt Flag	DMAC Activation	DTC Activation	Priority
ERI	Receive error	ORER, FER, or PER	Not possible	Not possible	High
RXI	Receive data full	RDRF	Possible	Possible	
TXI	Transmit data empty	TDRE	Possible	Possible	
TEI	Transmit end	TEND	Not possible	Not possible	Low



## 16.8.2 Interrupts in Smart Card Interface Mode

Table 16.13 shows the interrupt sources in smart card interface mode. A transmit end (TEI) interrupt request cannot be used in this mode.

**Table 16.13 SCI Interrupt Sources**

Name	Interrupt Source	Interrupt Flag	DMAC Activation	DTC Activation	Priority
ERI	Receive error or error signal detection	ORER, PER, or ERS	Not possible	Not possible	High
RXI	Receive data full	RDRF	Possible	Possible	
TXI	Transmit data empty	TDRE	Possible	Possible	Low

↑  
↓

Data transmission/reception using the DTC or DMAC is also possible in smart card interface mode, similar to in the normal SCI mode. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt. This activates the DTC or DMAC by a TXI request thus allowing transfer of transmit data if the TXI request is specified as a source of DTC or DMAC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DTC or DMAC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DTC or DMAC. Therefore, the SCI and DTC or DMAC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag in SSR, which is set at error occurrence, is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit in SCR to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DTC or DMAC, be sure to set and enable the DTC or DMAC prior to making SCI settings. For DTC or DMAC settings, see section 9, DMA Controller (DMAC) and section 10, Data Transfer Controller (DTC).

In reception, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. This activates the DTC or DMAC by an RXI request thus allowing transfer of receive data if the RXI request is specified as a source of DTC or DMAC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DTC or DMAC. If an error occurs, the RDRF flag is not set but the error flag is set. Therefore, the DTC or DMAC is not activated and an ERI interrupt request is issued to the CPU instead; the error flag must be cleared.

## 16.9 Usage Notes

### 16.9.1 Module Stop Function Setting

Operation of the SCI can be disabled or enabled using the module stop control register. The initial setting is for operation of the SCI to be halted. Register access is enabled by clearing the module stop state. For details, see section 24, Power-Down Modes.

### 16.9.2 Break Detection and Processing

When framing error 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 PER flag may also be set. Note that, since the SCI continues the receive operation even after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

### 16.9.3 Mark State and Break Detection

When the TE bit is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line in mark state (the state of 1) until TE is set to 1, set both DDR and DR to 1. Since the TE bit is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set DDR to 1 and DR to 0, and 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.

### 16.9.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) 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 the receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

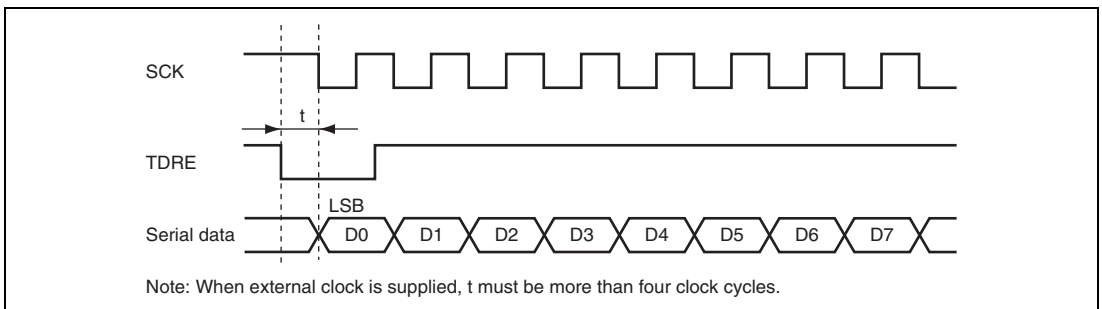
### 16.9.5 Relation between Writing to TDR and TDRE Flag

The TDRE flag in SSR is a status flag which 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 irrespective of the TDRE flag status. However, if new data is written to TDR when the TDRE flag is 0, that is, when the previous data has not been transferred to TSR yet, the previous data in TDR is lost. Be sure to write transmit data to TDR after verifying that the TDRE flag is set to 1.

### 16.9.6 Restrictions on Using DTC or DMAC

- When the external clock source is used as a synchronization clock, update TDR by the DTC or DMAC and wait for at least five  $P\phi$  clock cycles before allowing the transmit clock to be input. If the transmit clock is input within four clock cycles after TDR modification, the SCI may malfunction (figure 16.33).
- When using the DTC or DMAC to read RDR, be sure to set the receive end interrupt (RXI) as the DTC or DMAC activation source.



**Figure 16.33 Sample Transmission using DTC in Clocked Synchronous Mode**

## 16.9.7 SCI Operations during Mode Transitions

### (1) Transmission

Before making the transition to module stop state or software standby mode, stop the transmit operations ( $TE = TIE = TEIE = 0$ ). TSR, TDR, and SSR are reset. The states of the output pins during module stop state or software standby mode depend on the port settings, and the pins output a high-level signal after mode cancellation. If the transition is made during data transmission, the data being transmitted will be undefined.

To transmit data in the same transmission mode after mode cancellation, set the TE bit to 1, read SSR, write to TDR, clear TDRE in this order, and then start transmission. To transmit data in a different transmission mode, initialize the SCI first.

Figure 16.34 shows a sample flowchart for mode transition during transmission. Figures 16.35 and 16.36 show the port pin states during mode transition.

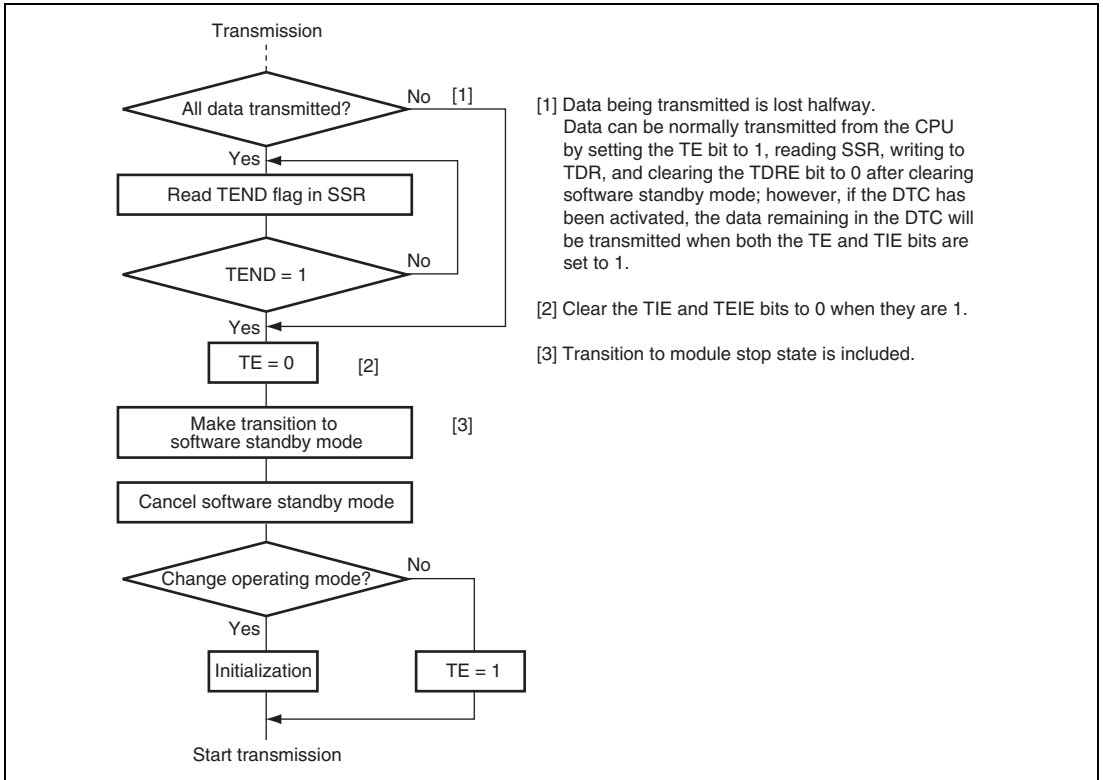
Before making the transition from the transmission mode using DTC transfer to module stop state or software standby mode, stop all transmit operations ( $TE = TIE = TEIE = 0$ ). Setting the TE and TIE bits to 1 after mode cancellation sets the TXI flag to start transmission using the DTC.

### (2) Reception

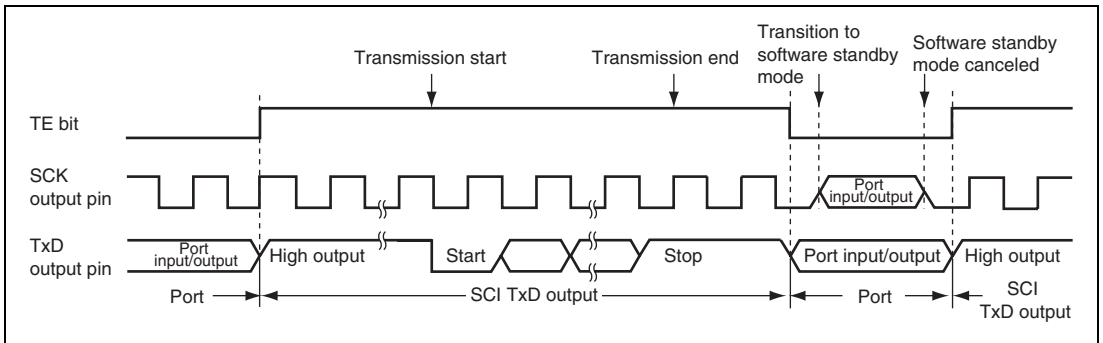
Before making the transition to module stop state or software standby mode, stop the receive operations ( $RE = 0$ ). RSR, RDR, and SSR are reset. If transition is made during data reception, the data being received will be invalid.

To receive data in the same reception mode after mode cancellation, set the RE bit to 1, and then start reception. To receive data in a different reception mode, initialize the SCI first.

Figure 16.37 shows a sample flowchart for mode transition during reception.

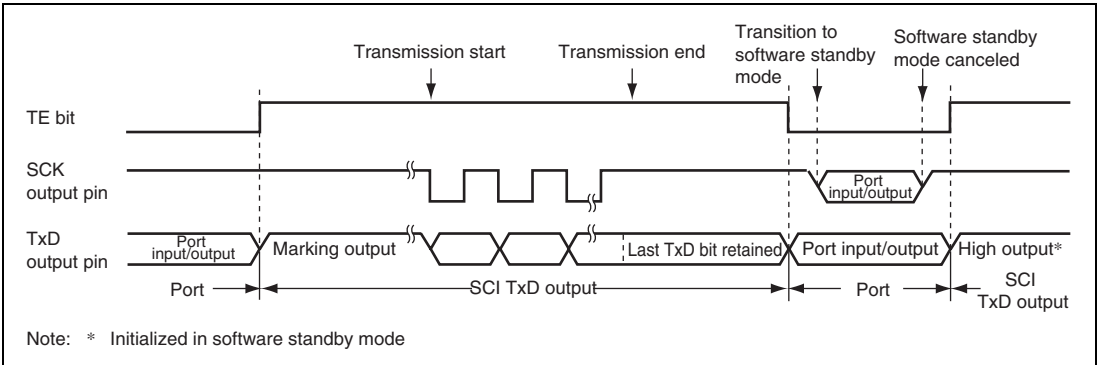


**Figure 16.34 Sample Flowchart for Mode Transition during Transmission**

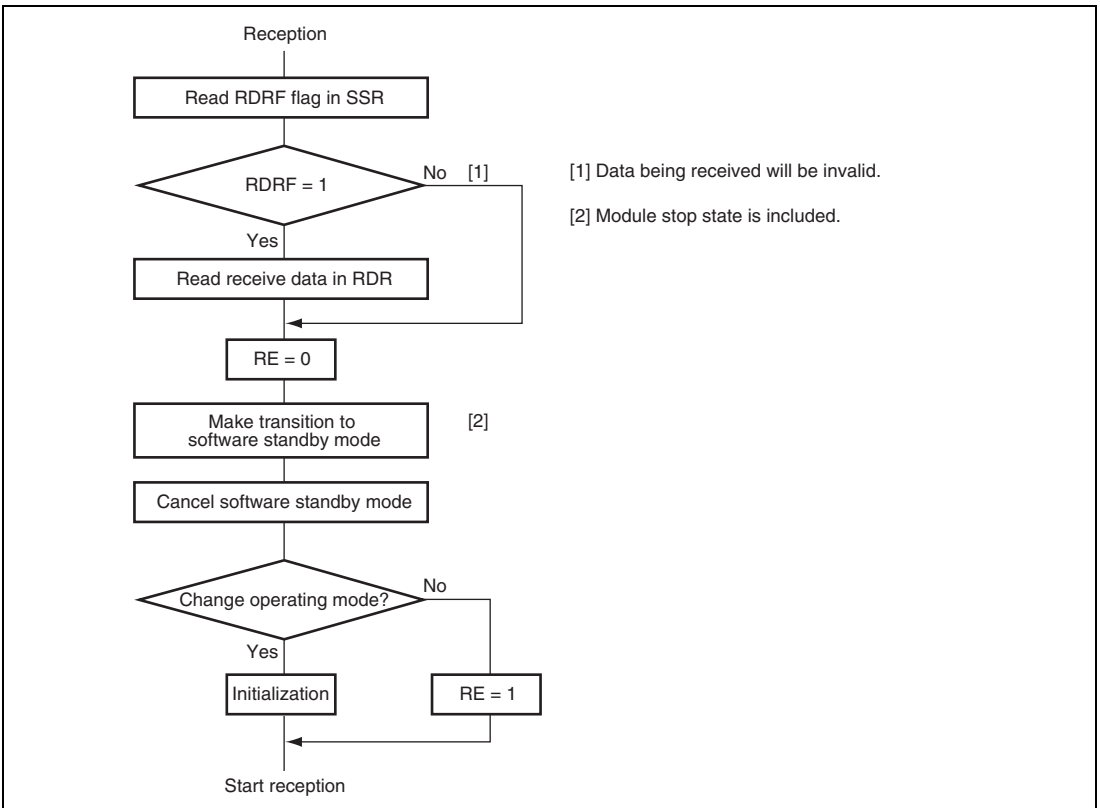


**Figure 16.35 Port Pin States during Mode Transition (Internal Clock, Asynchronous Transmission)**





**Figure 16.36 Port Pin States during Mode Transition (Internal Clock, Clocked Synchronous Transmission)**



**Figure 16.37 Sample Flowchart for Mode Transition during Reception**



## Section 17 I<sup>2</sup>C Bus Interface 2 (IIC2)

This LSI has a two-channel I<sup>2</sup>C bus interface.

The I<sup>2</sup>C bus interface conforms to and provides a subset of the Philips I<sup>2</sup>C bus (inter-IC bus) interface functions. The register configuration that controls the I<sup>2</sup>C bus differs partly from the Philips configuration, however.

Figure 17.1 shows the block diagram of the I<sup>2</sup>C bus interface 2. Figure 17.2 shows an example of I/O pin connections to external circuits.

### 17.1 Features

- Continuous transmission/reception  
Since the shift register, transmit data register, and receive data register are independent from each other, the continuous transmission/reception can be performed.
- Start and stop conditions generated automatically in master mode
- Selection of acknowledge output levels when receiving
- Automatic loading of acknowledge bit when transmitting
- Bit synchronization/wait function  
In master mode, the state of SCL is monitored per bit, and the timing is synchronized automatically. If transmission or reception is not yet possible, drive the SCL signal low until preparations are completed
- Six interrupt sources  
Transmit-data-empty (including slave-address match), transmit-end, receive-data-full (including slave-address match), arbitration lost, NACK detection, and stop condition detection
- Direct bus drive  
Two pins, the SCL and SDA pins function as NMOS open-drain outputs.
- Module stop state specifiable

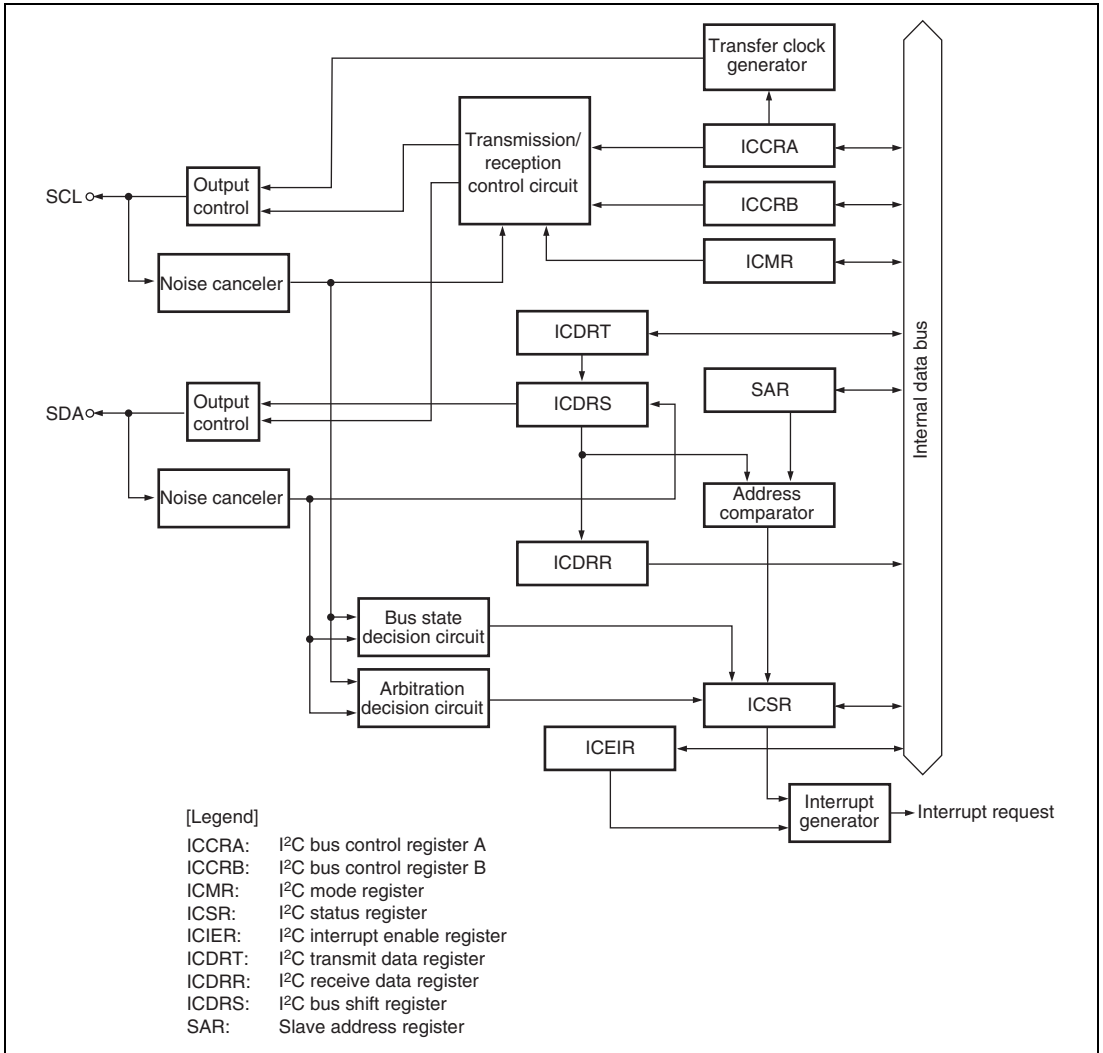
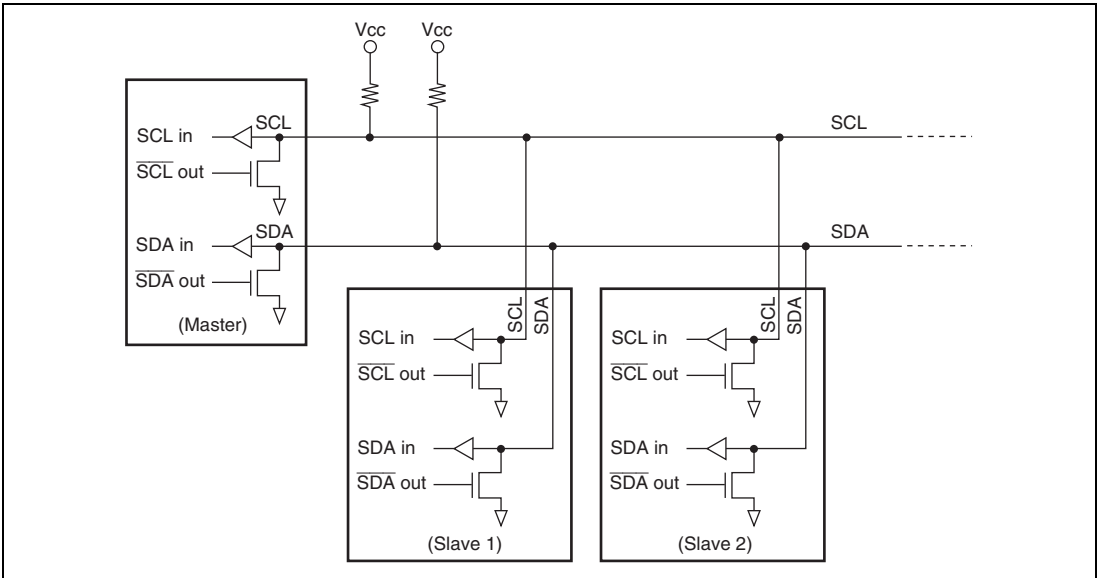


Figure 17.1 Block Diagram of I<sup>2</sup>C Bus Interface 2



**Figure 17.2** Connections to the External Circuit by the I/O Pins

## 17.2 Input/Output Pins

Table 17.1 shows the pin configuration of the I<sup>2</sup>C bus interface 2.

**Table 17.1** Pin Configuration of the I<sup>2</sup>C Bus Interface 2

Channel	Abbreviation	I/O	Function
0	SCL0	I/O	Channel 0 serial clock I/O pin
	SDA0	I/O	Channel 0 serial data I/O pin
1	SCL1	I/O	Channel 1 serial clock I/O pin
	SDA1	I/O	Channel 1 serial data I/O pin

Note: The pin symbols are represented as SCL and SDA; channel numbers are omitted in this manual.

## 17.3 Register Descriptions

The I<sup>2</sup>C bus interface 2 has the following registers.

Channel 0:

- I<sup>2</sup>C bus control register A\_0 (ICCRA\_0)
- I<sup>2</sup>C bus control register B\_0 (ICCRB\_0)
- I<sup>2</sup>C bus mode register\_0 (ICMR\_0)
- I<sup>2</sup>C bus interrupt enable register\_0 (ICIER\_0)
- I<sup>2</sup>C bus status register\_0 (ICSR\_0)
- Slave address register\_0 (SAR\_0)
- I<sup>2</sup>C bus transmit data register\_0 (ICDRT\_0)
- I<sup>2</sup>C bus receive data register\_0 (ICDRR\_0)
- I<sup>2</sup>C bus shift register\_0 (ICDRS\_0)

Channel 1:

- I<sup>2</sup>C bus control register A\_1 (ICCRA\_1)
- I<sup>2</sup>C bus control register B\_1 (ICCRB\_1)
- I<sup>2</sup>C bus mode register\_1 (ICMR\_1)
- I<sup>2</sup>C bus interrupt enable register\_1 (ICIER\_1)
- I<sup>2</sup>C bus status register\_1 (ICSR\_1)
- Slave address register\_1 (SAR\_1)
- I<sup>2</sup>C bus transmit data register\_1 (ICDRT\_1)
- I<sup>2</sup>C bus receive data register\_1 (ICDRR\_1)
- I<sup>2</sup>C bus shift register\_1 (ICDRS\_1)

### 17.3.1 I<sup>2</sup>C Bus Control Register A (ICCRA)

ICCRA enables or disables I<sup>2</sup>C bus interface, controls transmission or reception, and selects master or slave mode, transmission or reception, and transfer clock frequency in master mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	ICE	RCVD	MST	TRS	CKS3	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

Bit	Bit Name	Initial Value	R/W	Description
7	ICE	0	R/W	<p>I<sup>2</sup>C Bus Interface Enable</p> <p>0: This module is halted (SCL and SDA pins are used as the port function)</p> <p>1: This bit is enabled for transfer operations (SCL and SDA pins are bus drive state)</p>
6	RCVD	0	R/W	<p>Reception Disable</p> <p>This bit enables or disables the next operation when TRS is 0 and ICDRR is read.</p> <p>0: Enables next reception</p> <p>1: Disables next reception</p>
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	<p>Transmit/Receive Select</p> <p>When arbitration is lost in master mode, MST and TRS are both reset by hardware, causing a transition to slave receive mode. Modification of the TRS bit should be made between transfer frames.</p> <p>Operating modes are described below according to MST and TRS combination.</p> <p>00: Slave receive mode</p> <p>01: Slave transmit mode</p> <p>10: Master receive mode</p> <p>11: Master transmit mode</p>
3	CKS3	0	R/W	Transfer Clock Select 3 to 0
2	CKS2	0	R/W	These bits are valid only in master mode. Make setting according to the required transfer rate. For details on the transfer rate, see table 17.2.
1	CKS1	0	R/W	
0	CKS0	0	R/W	

**Table 17.2 Transfer Rate**

Bit 3	Bit 2	Bit 1	Bit 0	Clock	Transfer Rate					
					P $\phi$ = 8 MHz	P $\phi$ = 10 MHz	P $\phi$ = 20 MHz	P $\phi$ = 25 MHz	P $\phi$ = 33 MHz	P $\phi$ = 35 MHz
0	0	0	0	P $\phi$ /28	286 kHz	357 kHz	714 kHz	893 kHz	1179 kHz	1250 kHz
			1	P $\phi$ /40	200 kHz	250 kHz	500 kHz	625 kHz	825 kHz	875 kHz
		1	0	P $\phi$ /48	167 kHz	208 kHz	417 kHz	521 kHz	688 kHz	729 kHz
			1	P $\phi$ /64	125 kHz	156 kHz	313 kHz	391 kHz	516 kHz	546 kHz
	1	0	0	P $\phi$ /168	47.6 kHz	59.5 kHz	119 kHz	149 kHz	196 kHz	208 kHz
			1	P $\phi$ /100	80.0 kHz	100 kHz	200 kHz	250 kHz	330 kHz	350 kHz
		1	0	P $\phi$ /112	71.4 kHz	89.3 kHz	179 kHz	223 kHz	295 kHz	312 kHz
			1	P $\phi$ /128	62.5 kHz	78.1 kHz	156 kHz	195 kHz	258 kHz	273 kHz
1	0	0	0	P $\phi$ /56	143 kHz	179 kHz	357 kHz	446 kHz	589 kHz	625 kHz
			1	P $\phi$ /80	100 kHz	125 kHz	250 kHz	313 kHz	413 kHz	437 kHz
		1	0	P $\phi$ /96	83.3 kHz	104 kHz	208 kHz	260 kHz	344 kHz	364 kHz
			1	P $\phi$ /128	62.5 kHz	78.1 kHz	156 kHz	195 kHz	258 kHz	273 kHz
	1	0	0	P $\phi$ /336	23.8 kHz	29.8 kHz	59.5 kHz	74.4 kHz	98.2 kHz	104 kHz
			1	P $\phi$ /200	40.0 kHz	50.0 kHz	100 kHz	125 kHz	165 kHz	175 kHz
		1	0	P $\phi$ /224	35.7 kHz	44.6 kHz	89.3 kHz	112 kHz	147 kHz	156 kHz
			1	P $\phi$ /256	31.3 kHz	39.1 kHz	78.1 kHz	97.7 kHz	129 kHz	136 kHz

### 17.3.2 I<sup>2</sup>C Bus Control Register B (ICCRB)

ICCRB issues start/stop condition, manipulates the SDA pin, monitors the SCL pin, and controls reset in the I<sup>2</sup>C control module.

Bit	7	6	5	4	3	2	1	0
Bit Name	BBSY	SCP	SDAO	—	SCLO	—	IICRST	—
Initial Value	0	1	1	1	1	1	0	1
R/W	R/W	R/W	R	R/W	R	—	R/W	—



Bit	Bit Name	Initial Value	R/W	Description
7	BBSY	0	R/W	<p>Bus Busy</p> <p>This bit indicates whether the I<sup>2</sup>C bus is occupied or released and to issue start and stop conditions in master mode. This bit is set to 1 when the SDA level changes from high to low under the condition of SCL = high, assuming that the start condition has been issued. This bit is cleared to 0 when the SDA level changes from low to high under the condition of SDA = high, assuming that the stop condition has been issued. Follow this procedure also when re-transmitting a start condition. To issue a start or stop condition, use the MOV instruction.</p>
6	SCP	1	R/W	<p>Start/Stop Condition Issue</p> <p>This bit controls the issuance of start or stop condition in master mode.</p> <p>To issue a start condition, write 1 to BBSY and 0 to SCP. A re-transmit start condition is issued in the same way. To issue a stop condition, write 0 to BBSY and 0 to SCP. This bit is always read as 1. If 1 is written, the data is not stored.</p>
5	SDAO	1	R	<p>This bit monitors the output level of SDA.</p> <p>0: When reading, the SDA pin outputs a low level 1: When reading the SDA pin outputs a high level</p>
4	—	1	R/W	<p>Reserved</p> <p>The write value should always be 1.</p>
3	SCLO	1	R	<p>This bit monitors the SCL output level.</p> <p>When reading and SCLO is 1, the SCL pin outputs a high level. When reading and SCLO is 0, the SCL pin outputs a low level.</p>
2	—	1	—	<p>Reserved</p> <p>This bit is always read as 0.</p>
1	IICRST	0	R/W	<p>IIC Control Module Reset</p> <p>This bit reset the IIC control module except the I<sup>2</sup>C registers. If hang-up occurs because of communication failure during I<sup>2</sup>C operation, by setting this bit to 1, the</p>
0	—	1	—	<p>Reserved</p> <p>This bit is always read as 1.</p>

### 17.3.3 I<sup>2</sup>C Bus Mode Register (ICMR)

ICMR selects MSB first or LSB first, controls the master mode wait and selects the number of transfer bits.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	WAIT	—	—	BCWP	BC2	BC1	BC0
Initial Value	0	0	1	1	1	0	0	0
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved The write value should always be 0.
6	WAIT	0	R/W	Wait Insertion This bit selects whether to insert a wait after data transfer except for the acknowledge bit. When this bit is set to 1, after the falling of the clock for the last data bit, the low period is extended for two transfer clocks. When this bit is cleared to 0, data and the acknowledge bit are transferred consecutively with no waits inserted. The setting of this bit is invalid in slave mode.
5	—	1	—	Reserved
4	—	1	—	These bits are always read as 1.
3	BCWP	1	R/W	BC Write Protect This bit controls the modification of the BC2 to BC0 bits. When modifying, this bit should be cleared to 0 and the MOV instruction should be used. 0: When writing, the values of BC2 to BC0 are set 1: When reading, 1 is always read When writing, the settings of BC2 to BC0 are invalid.

Bit	Bit Name	Initial Value	R/W	Description
2	BC2	0	R/W	Bit Counter 2 to 0
1	BC1	0	R/W	These bits specify the number of bits to be transferred next. The settings of these bits should be made during intervals between transfer frames. When setting these bits to a value other than 000, the setting should be made while the SCL line is low. The value return to 000 at the end of a data transfer including the acknowledge bit.  000: 9 001: 2 010: 3 011: 4 100: 5 101: 6 110: 7 111: 8  I <sup>2</sup> C control module can be reset without setting the ports and initializing the registers.
0	BC0	0	R/W	

### 17.3.4 I<sup>2</sup>C Bus Interrupt Enable Register (ICIER)

ICIER enables or disables interrupt sources and the acknowledge bits, sets the acknowledge bits to be transferred, and confirms the acknowledge bit to be received.

Bit	7	6	5	4	3	2	1	0
Bit Name	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR	ACKBT
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	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	<p>Transmit Interrupt Enable</p> <p>When the TDRE bit in ICSR is set to 1, this bit enables or disables the transmit data empty interrupt (TXI) request.</p> <p>0: Transmit data empty interrupt (TXI) request is disabled</p> <p>1: Transmit data empty interrupt (TXI) request is enabled</p>
6	TEIE	0	R/W	<p>Transmit End Interrupt Enable</p> <p>This bit enables or disables the transmit end interrupt (TEI) request at the rising of the ninth clock while the TDRE bit in ICSR is set to 1. The TEI request can be canceled by clearing the TEND bit or the TEIE bit to 0.</p> <p>0: Transmit end interrupt (TEI) request is disabled</p> <p>1: Transmit end interrupt (TEI) request is enabled</p>
5	RIE	0	R/W	<p>Receive Interrupt Enable</p> <p>This bit enables or disables the receive full interrupt (RXI) request when receive data is transferred from ICDRS to ICDRR and the RDRF bit in ICSR is set to 1. The RXI request can be canceled by clearing the RDRF or RIE bit to 0.</p> <p>0: Receive data full interrupt (RXI) request is disabled</p> <p>1: Receive data full interrupt (RXI) request is enabled</p>

Bit	Bit Name	Initial Value	R/W	Description
4	NAKIE	0	R/W	<p>NACK Receive Interrupt Enable</p> <p>This bit enables or disables the NACK receive interrupt (NAKI) request when the NACKF and AL bits in ICSR are set to 1. The NAKI request can be canceled by clearing the NACKF or AL bit, or the NAKIE bit to 0.</p> <p>0: NACK receive interrupt (NAKI) request is disabled 1: NACK receive interrupt (NAKI) request is enabled</p>
3	STIE	0	R/W	<p>Stop Condition Detection Interrupt Enable</p> <p>0: Stop condition detection interrupt (STPI) request is disabled 1: Stop condition detection interrupt (STPI) request is enabled</p>
2	ACKE	0	R/W	<p>Acknowledge Bit Decision Select</p> <p>0: The value of the acknowledge bit is ignored and continuous transfer is performed 1: If the acknowledge bit is 1, continuous transfer is suspended</p>
1	ACKBR	0	R	<p>Receive Acknowledge</p> <p>In transmit mode, this bit stores the acknowledge data that are returned by the receive device. This bit cannot be modified.</p> <p>0: Receive acknowledge = 0 1: Receive acknowledge = 1</p>
0	ACKBT	0	R/W	<p>Transmit Acknowledge</p> <p>In receive mode, this bit specifies the bit to be sent at the acknowledge timing.</p> <p>0: 0 is sent at the acknowledge timing 1: 1 is sent at the acknowledge timing</p>

### 17.3.5 I<sup>2</sup>C Bus Status Register (ICSR)

ICSR confirms the interrupt request flags and status.

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	TEND	RDRF	NACKF	STOP	AL	AAS	ADZ
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	Bit Name	Initial Value	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting conditions] <ul style="list-style-type: none"> <li>When data is transferred from ICDRT to ICDRS and ICDRT becomes empty</li> <li>When the TRS bit is set</li> <li>When a start condition (that includes a retransmit condition) is issued</li> <li>When the slave receive mode shifts to the slave transmit mode</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>When data is written to ICDRT</li> </ul>
6	TEND	0	R/W	Transmit End [Setting condition] <ul style="list-style-type: none"> <li>When the ninth clock of SCL rises while the TDRE flag is 1</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading TEND = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>When data is written to ICDRT</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
5	RDRF	0	R/W	<p>Receive Data Register Full</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When receive data is transferred from ICDRS to ICRRR</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>When data is read from ICRRR</li> </ul>
4	NACKF	0	R/W	<p>No Acknowledge Detection Flag</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When no acknowledge is detected from the receive device in transmission while the ACKF bit in ICIEP is set to 1</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading NACKF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>
3	STOP	0	R/W	<p>Stop Condition Detection Flag</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>In master mode, when a stop condition is detected after frame transfer</li> <li>In slave mode, when a stop condition is detected after a general call or after the slave address that came as the first byte after detection of a start condition has matched the address set in SAR</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading STOP = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	AL	0	R/W	<p>Arbitration Lost Flag</p> <p>This flag indicates that arbitration was lost in master mode.</p> <p>When two or more master devices attempt to seize the bus at nearly the same time, the I<sup>2</sup>C bus monitors SDA, and if the I<sup>2</sup>C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When the internal SDA and the SDA pin level disagree at the rising of SCL in master transmit mode</li> <li>• When the SDA pin outputs a high level in master mode while a start condition is detected</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>• When 0 is written to this bit after reading AL = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>
1	AAS	0	R/W	<p>Slave Address Recognition Flag</p> <p>In slave receive mode, this flag is set to 1 when the first frame following a start condition matches bits SVA6 to SVA0 in SAR.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When the slave address is detected in slave receive mode</li> <li>• When the general call address is detected in slave receive mode</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>• When 0 is written to this bit after reading AAS = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>



Bit	Bit Name	Initial Value	R/W	Description
0	ADZ	0	R/W	<p>General Call Address Recognition Flag</p> <p>This bit is valid in slave receive mode.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the general call address is detected in slave receive mode</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading ADZ = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> </ul>

### 17.3.6 Slave Address Register (SAR)

SAR sets the slave address. In slave mode, if the upper 7 bits of SAR match the upper 7 bits of the first frame received after a start condition, the LSI operates as the slave device.

Bit	7	6	5	4	3	2	1	0
Bit Name	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	—
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	Bit Name	Initial Value	R/W	Description
7 to 1	SVA6 to SVA0	0	R/W	<p>Slave Address 6 to 0</p> <p>These bits set a unique address differing from the addresses of other slave devices connected to the I<sup>2</sup>C bus.</p>
0	—	0	R/W	<p>Reserved</p> <p>Although this bit is readable/writable, only 0 should be written to.</p>

### 17.3.7 I<sup>2</sup>C Bus Transmit Data Register (ICDRT)

ICDRT is an 8-bit readable/writable register that stores the transmit data. When ICDRT detects a space in the I<sup>2</sup>C bus shift register, it transfers the transmit data which has been written to ICDRT to ICDRS and starts transmitting data. If the next data is written to ICDRT during transmitting data to ICDRS, continuous transmission is possible.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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

### 17.3.8 I<sup>2</sup>C Bus Receive Data Register (ICDRR)

ICDRR is an 8-bit read-only register that stores the receive data. When one byte of data has been received, ICDRR transfers the receive data from ICDRS to ICDRR and the next data can be received. ICDRR is a receive-only register; therefore, this register cannot be written to by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

### 17.3.9 I<sup>2</sup>C Bus Shift Register (ICDRS)

ICDRS is an 8-bit write-only register that is used to transmit/receive data. In transmission, data is transferred from ICDRT to ICDRS and the data is sent from the SDA pin. In reception, data is transferred from ICDRS to ICDRR after one byte of data is received. This register cannot be read from the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

## 17.4 Operation

### 17.4.1 I<sup>2</sup>C Bus Format

Figure 17.3 shows the I<sup>2</sup>C bus formats. Figure 17.4 shows the I<sup>2</sup>C bus timing. The first frame following a start condition always consists of 8 bits.

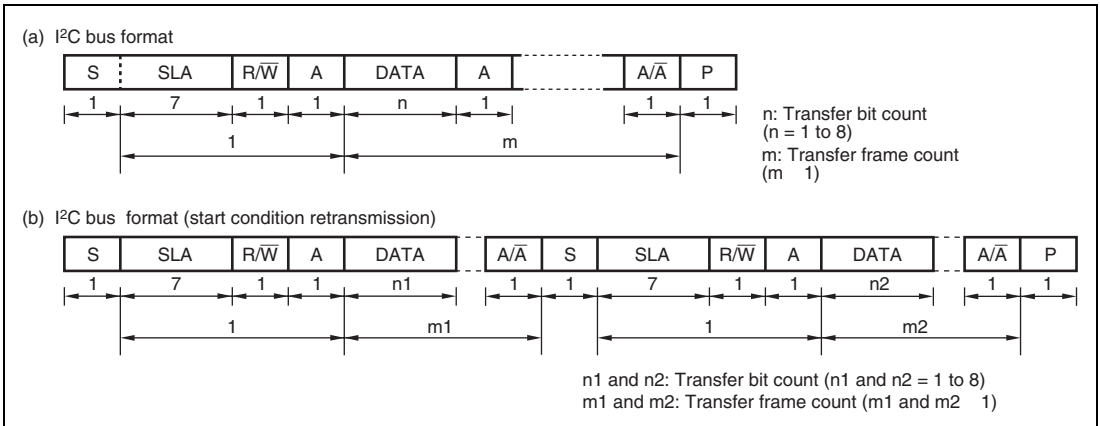


Figure 17.3 I<sup>2</sup>C Bus Formats

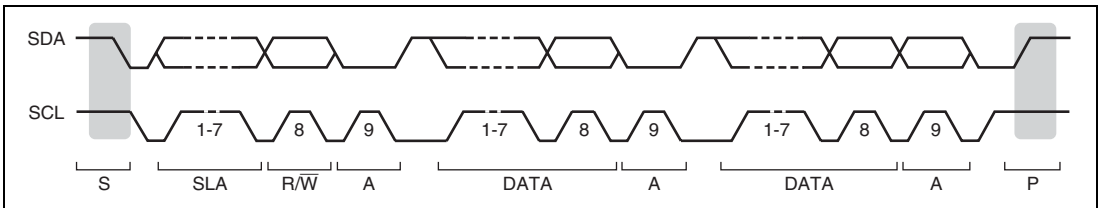


Figure 17.4 I<sup>2</sup>C Bus Timing

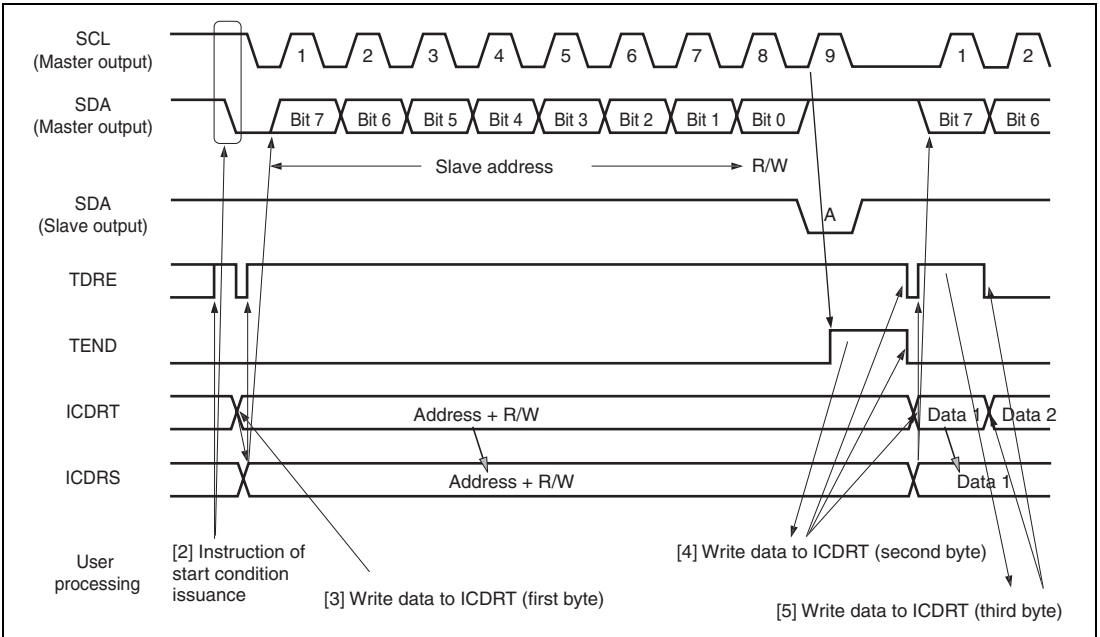
#### [Legend]

- S: Start condition. The master device drives SDA from high to low while SCL is high.
- SLA: Slave address
- R/W: Indicates the direction of data transfer; from the slave device to the master device when R/W is 1, or from the master device to the slave device when R/W is 0.
- A: Acknowledge. The receive device drives SDA low.
- DATA: Transferred data
- P: Stop condition. The master device drives SDA from low to high while SCL is high.

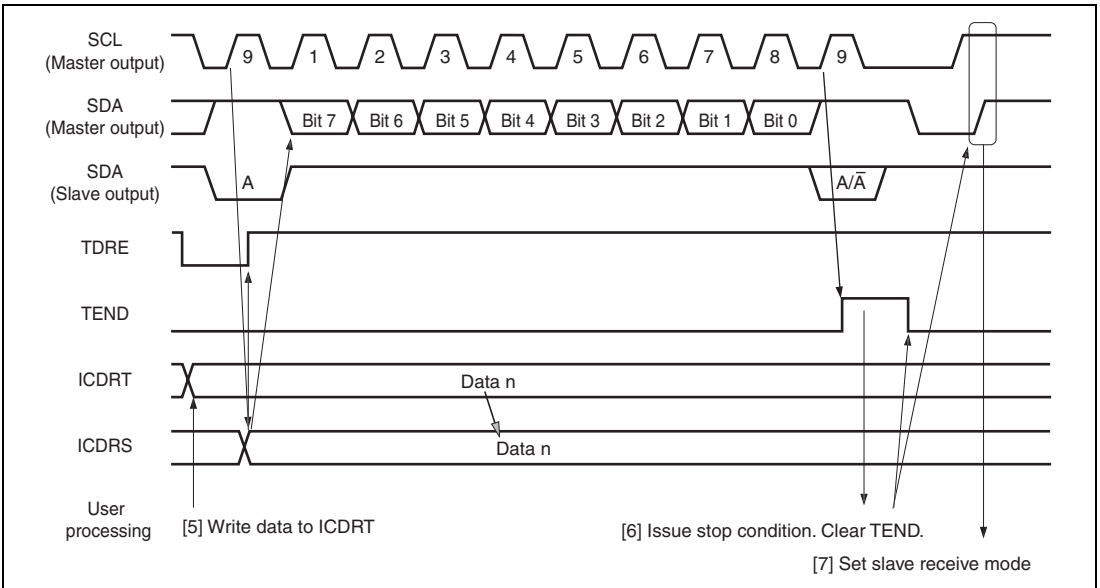
## 17.4.2 Master Transmit Operation

In I<sup>2</sup>C bus format master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device return an acknowledge signal. Figures 17.5 and 17.6 show the operating timings in master transmit mode. The transmission procedure and operations in master transmit mode are described below.

1. Set the ICR bit in the corresponding register to 1. Set the ICE bit in ICCRA to 1. Set the WAIT bit in ICMR and the CKS3 to CKS0 bits in ICCRA to 1. (initial setting)
2. Read the BSSY flag in ICCRB to confirm that the bus is free. Set the MST and TRS bits in ICCRA to select master transmit mode. Then, write 1 to BBSY and 0 to SCP using the MOV instruction. (The start condition is issued.) This generates the start condition.
3. After confirming that TDRE in ICSR has been set, write the transmit data (the first byte shows the slave address and R/W) to ICDRT. After this, when TDRE is automatically cleared to 0, data is transferred from ICDRT to ICDRS. TDRE is set again.
4. When transmission of one byte data is completed while TDRE is 1, TEND in ICSR is set to 1 at the rising of the ninth transmit clock pulse. Read the ACKBR bit in ICIER to confirm that the slave device has been selected. Then, write the second byte data to ICDRT. When ACKBR is 1, the slave device has not been acknowledged, so issue a stop condition. To issue the stop condition, write 0 to BBSY and SCP using the MOV instruction. SCL is fixed to a low level until the transmit data is prepared or the stop condition is issued.
5. The transmit data after the second byte is written to ICDRT every time TDRE is set.
6. Write the number of bytes to be transmitted to ICDRT. Wait until TEND is set (the end of last byte data transmission) while TDRE is 1, or wait for NACK (NACKF in ICSR is 1) from the receive device while CKE in ICIER is 1. Then, issue the stop condition to clear TEND or NACKF.
7. When the STOP bit in ICSR is set to 1, the operation returns to the slave receive mode.



**Figure 17.5 Master Transmit Mode Operation Timing 1**

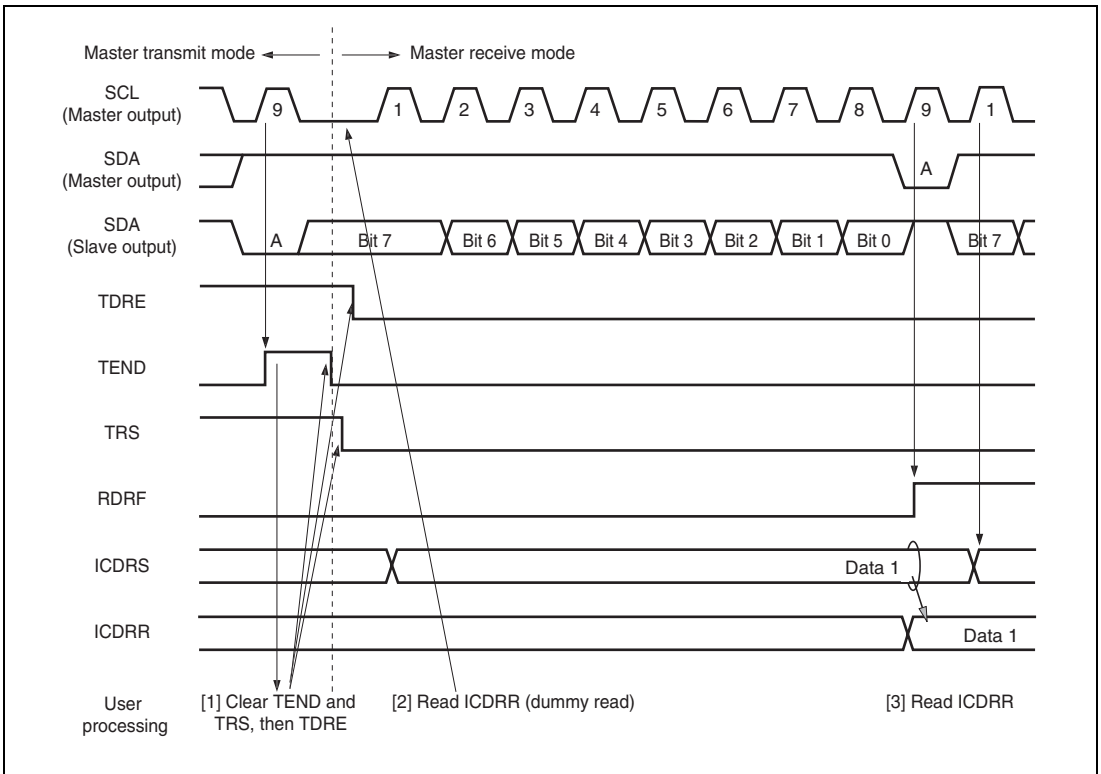


**Figure 17.6 Master Transmit Mode Operation Timing 2**

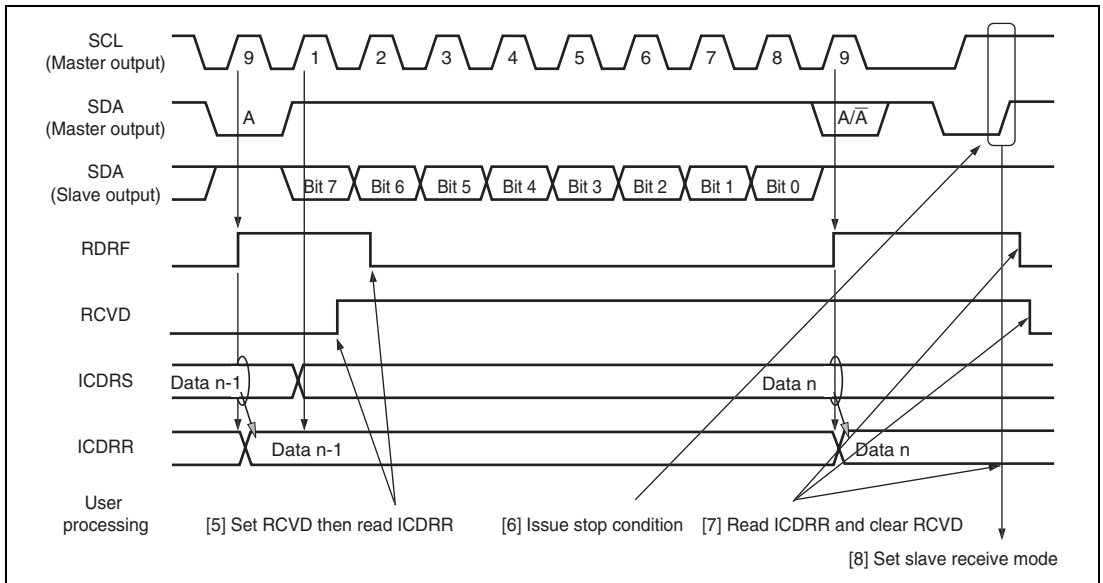
### 17.4.3 Master Receive Operation

In master receive mode, the master device outputs the receive clock, receives data from the slave device, and returns an acknowledge signal. Figures 17.7 and 17.8 show the operation timings in master receive mode. The reception procedure and operations in master receive mode are shown below.

1. Clear the TEND bit in ICSR to 0, then clear the TRS bit in ICCRA to 0 to switch from master transmit mode to master receive mode. Then, clear the TDRE bit to 0.
2. When ICDDR is read (dummy read), reception is started, the receive clock pulse is output, and data is received, in synchronization with the internal clock. The master mode outputs the level specified by the ACKBT in ICIER to SDA, at the ninth receive clock pulse.
3. After the reception of the first frame data is completed, the RDRF bit in ICSR is set to 1 at the rising of the ninth receive clock pulse. At this time, the received data is read by reading ICDRR. At the same time, RDRF is cleared.
4. The continuous reception is performed by reading ICDRR and clearing RDRF to 0 every time RDRF is set. If the eighth receive clock pulse falls after reading ICDRR by other processing while RDRF is 1, SCL is fixed to a low level until ICDRR is read.
5. If the next frame is the last receive data, set the RCVD bit in ICCR1 before reading ICDRR. This enables the issuance of the stop condition after the next reception.
6. When the RDRF bit is set to 1 at the rising of the ninth receive clock pulse, the stop condition is issued.
7. When the STOP bit in ICSR is set to 1, read ICDRR and clear RCVD to 0.
8. The operation returns to the slave receive mode.



**Figure 17.7 Master Receive Mode Operation Timing 1**



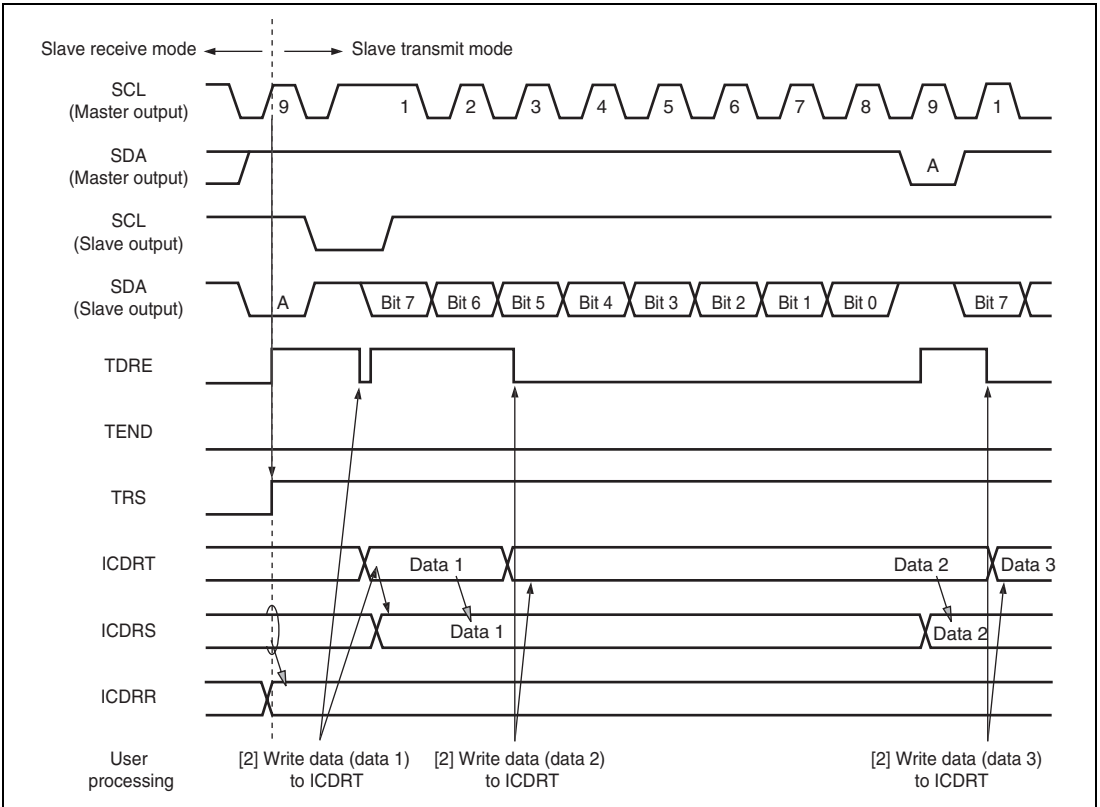
**Figure 17.8 Master Receive Mode Operation Timing 2**

#### 17.4.4 Slave Transmit Operation

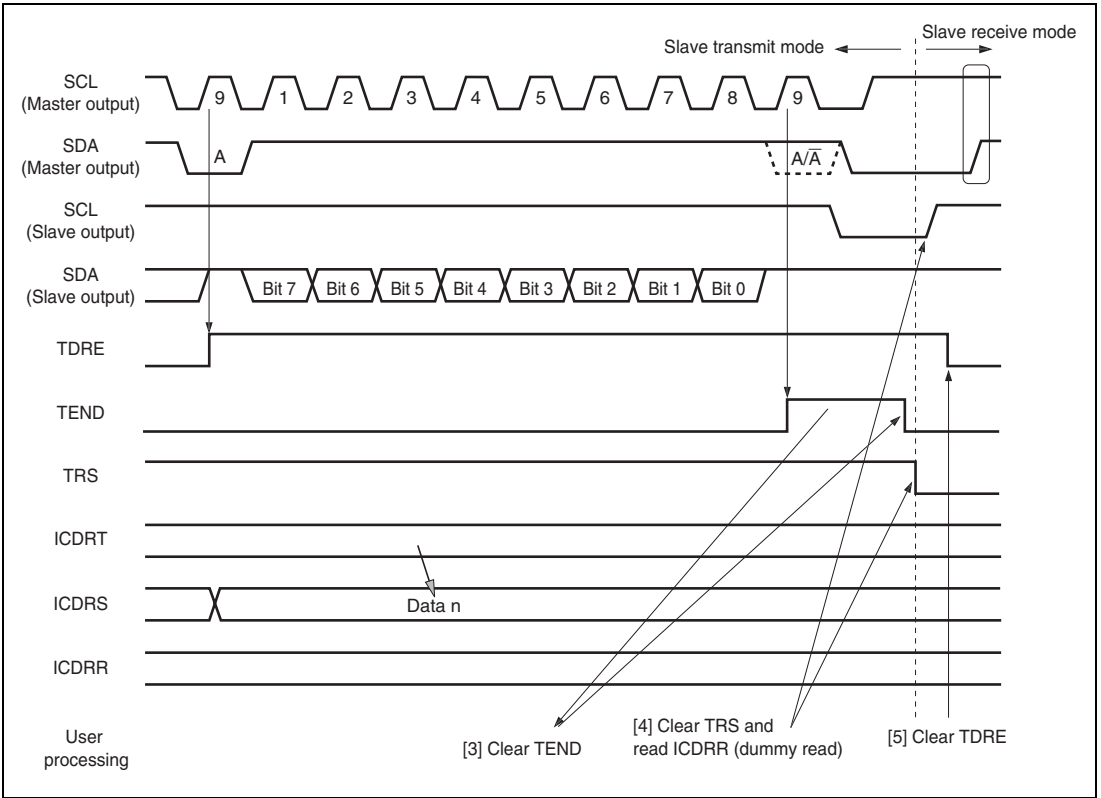
In slave transmit mode, the slave device outputs the transmit data, and the master device outputs the receive clock pulse and returns an acknowledge signal. Figures 17.9 and 16.10 show the operation timings in slave transmit mode. The transmission procedure and operations in slave transmit mode are described below.

1. Set the ICR bit in the corresponding register to 1, then set the ICE bit in ICCRA to 1. Set the ACKBIT in ICIER, and perform other initial settings. Set the MST and TRS bits in ICCRA to select slave receive mode, and wait until the slave address matches.
2. When the slave address matches in the first frame following the detection of the start condition, the slave device outputs the level specified by ACKBT in ICIER to SDA, at the rising of the ninth clock pulse. At this time, if the eighth bit data ( $R/\bar{W}$ ) is 1, TRS in ICCRA and TDRE in ICSR are set to 1, and the mode changes to slave transmit mode automatically. The continuous transmission is performed by writing the transmit data to ICDRT every time TDRE is set.
3. If TDRE is set after writing the last transmit data to ICDRT, wait until TEND in ICSR is set to 1, with TDRE = 1. When TEND is set, clear TEND.
4. Clear TRS for end processing, and read ICDRR (dummy read) to free SCL.
5. Clear TDRE.





**Figure 17.9 Slave Transmit Mode Operation Timing 1**

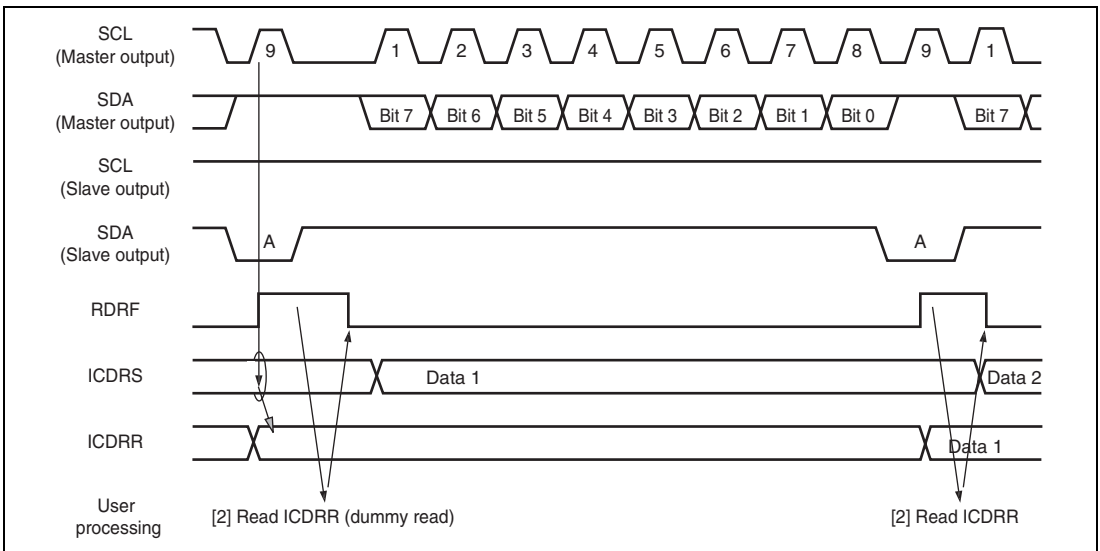


**Figure 17.10 Slave Transmit Mode Operation Timing 2**

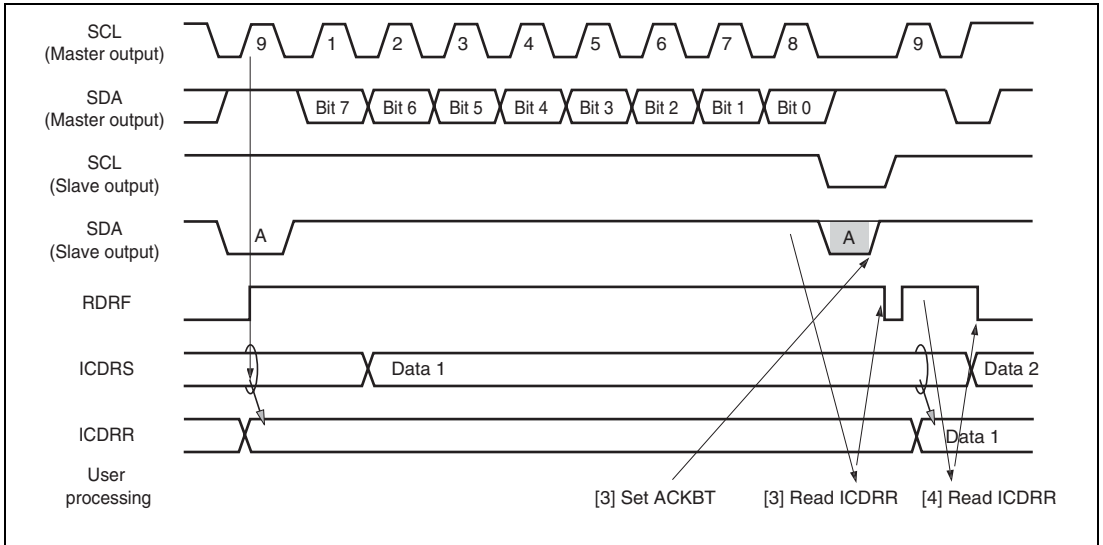
### 17.4.5 Slave Receive Operation

In slave receive mode, the master device outputs the transmit clock and the transmit data, and the slave device returns an acknowledge signal. Figures 17.11 and 17.12 show the operation timings in slave receive mode. The reception procedure and operations in slave receive mode are described below.

1. Set the ICR bit in the corresponding register to 1. Then, set the ICE bit in ICCRA to 1. Set the ACKBT bit in ICIER and perform other initial settings. Set the MST and TRS bits in ICCRA to select slave receive mode and wait until the slave address matches.
2. When the slave address matches in the first frame following detection of the start condition, the slave address outputs the level specified by ACKBT in ICIER to SDA, at the rising of the ninth clock pulse. At the same time, RDRF in ICSR is set to read ICDRR (dummy read). (Since the read data shows the slave address and  $R/\bar{W}$ , it is not used).
3. Read ICDRR every time RDRF is set. If the eighth clock pulse falls while RDRF is 1, SCL is fixed to a low level until ICDRR is read. The change of the acknowledge (ACKBT) setting before reading ICDRR to be returned to the master device is reflected in the next transmit frame.
4. The last byte data is read by reading ICDRR.



**Figure 17.11 Slave Receive Mode Operation Timing 1**

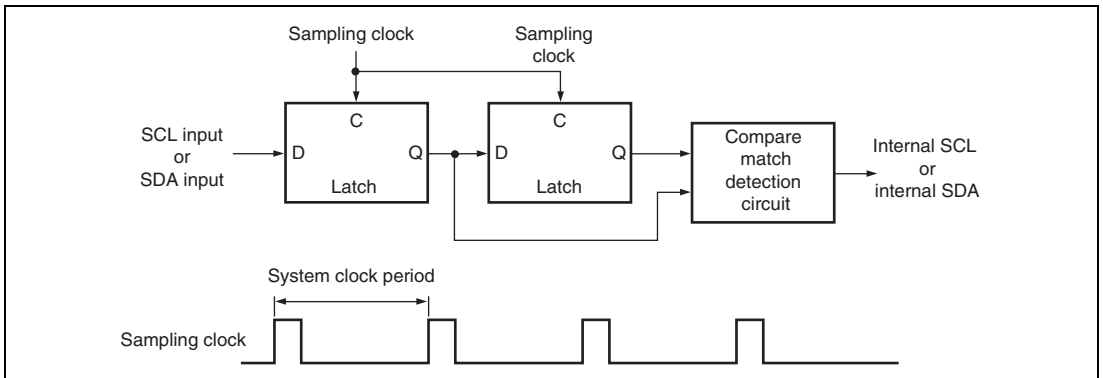


**Figure 17.12 Slave Receive Mode Operation Timing 2**

**17.4.6 Noise Canceler**

The logic levels at the SCL and SDA pins are routed through the noise cancelers before being latched internally. Figure 17.13 shows a block diagram of the noise canceler circuit.

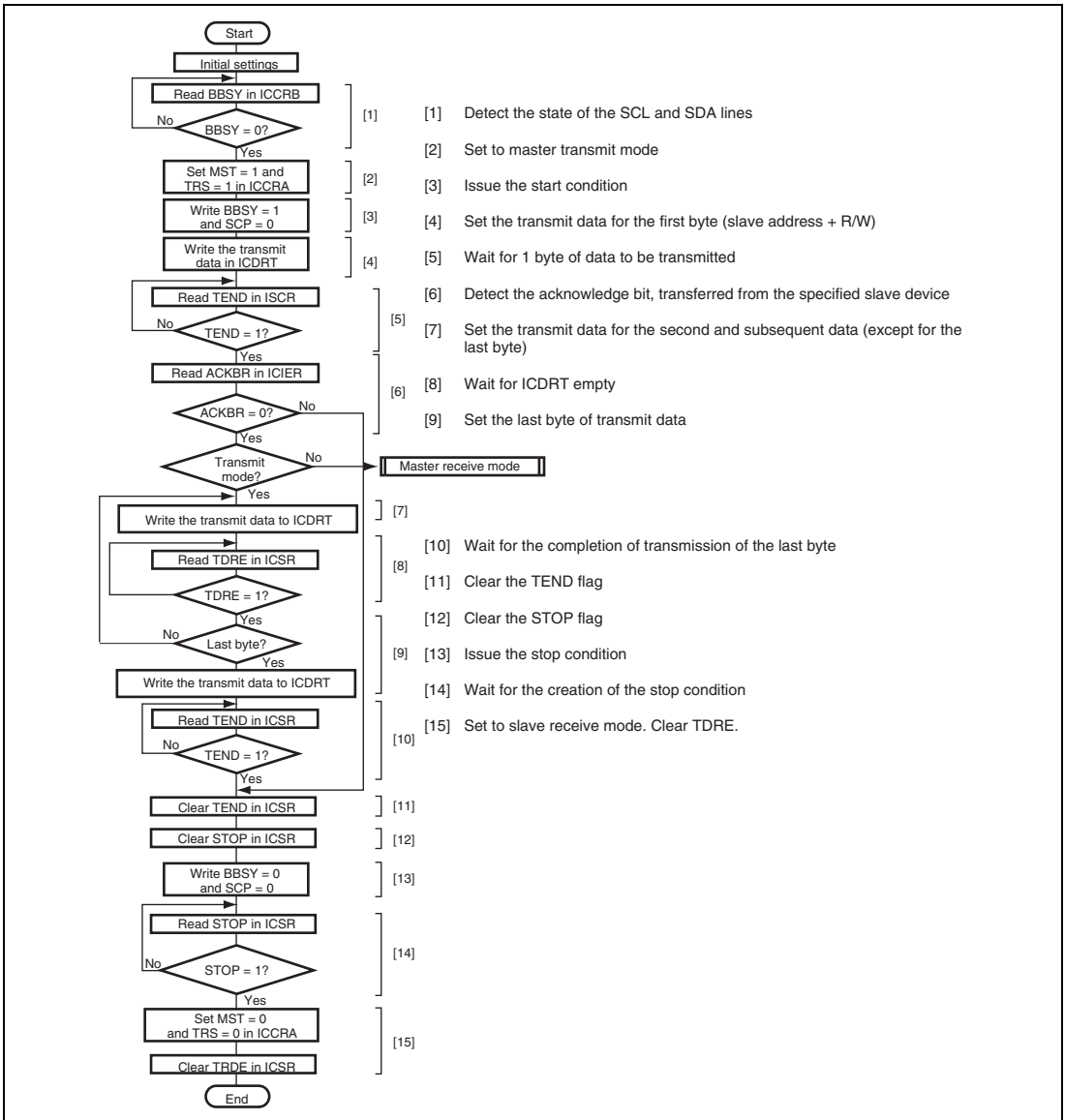
The noise canceler consists of two cascaded latches and a match detector. The signal input to SCL (or SDA) is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.



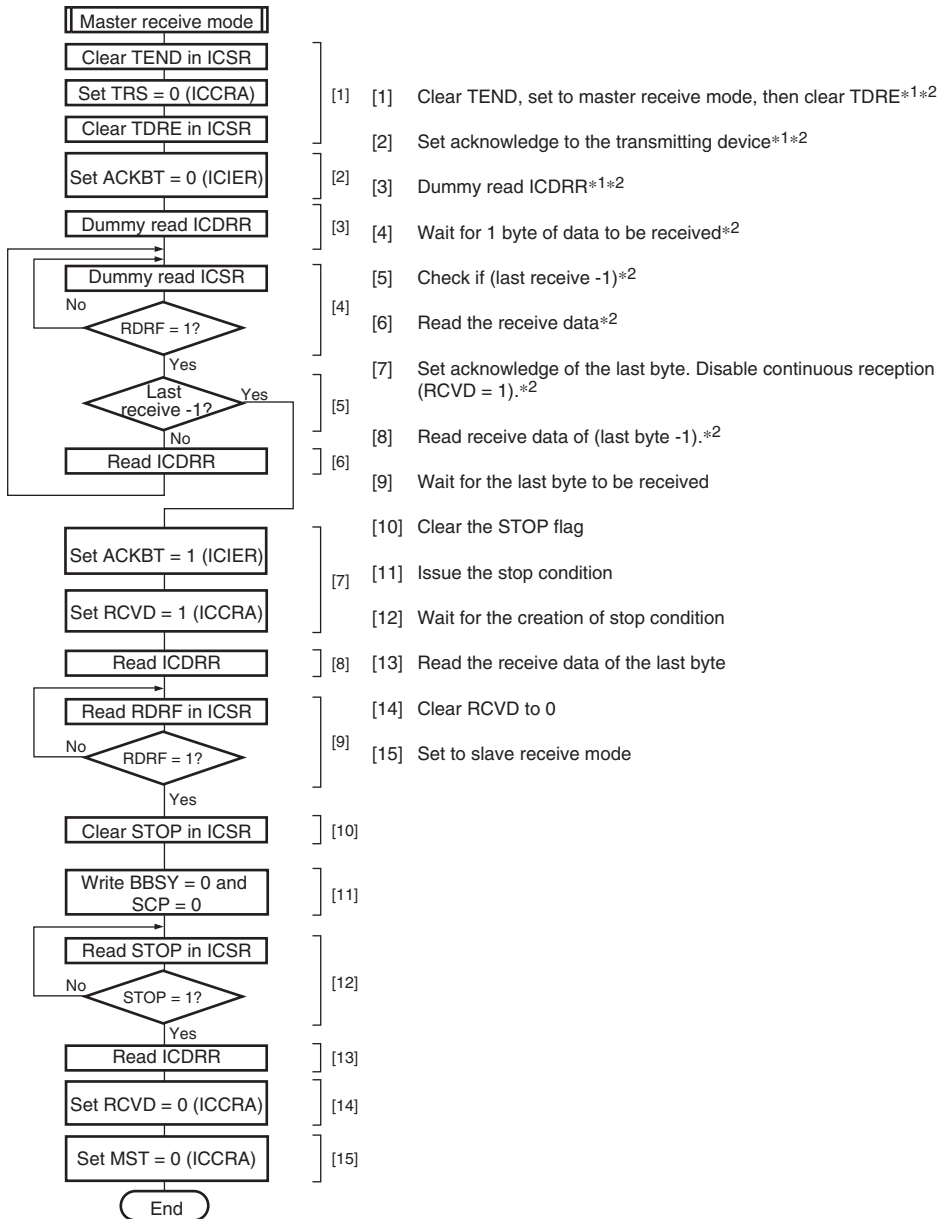
**Figure 17.13 Block Diagram of Noise Canceler**

### 17.4.7 Example of Use

Sample flowcharts in respective modes that use the I<sup>2</sup>C bus interface are shown in figures 17.14 to 17.17.

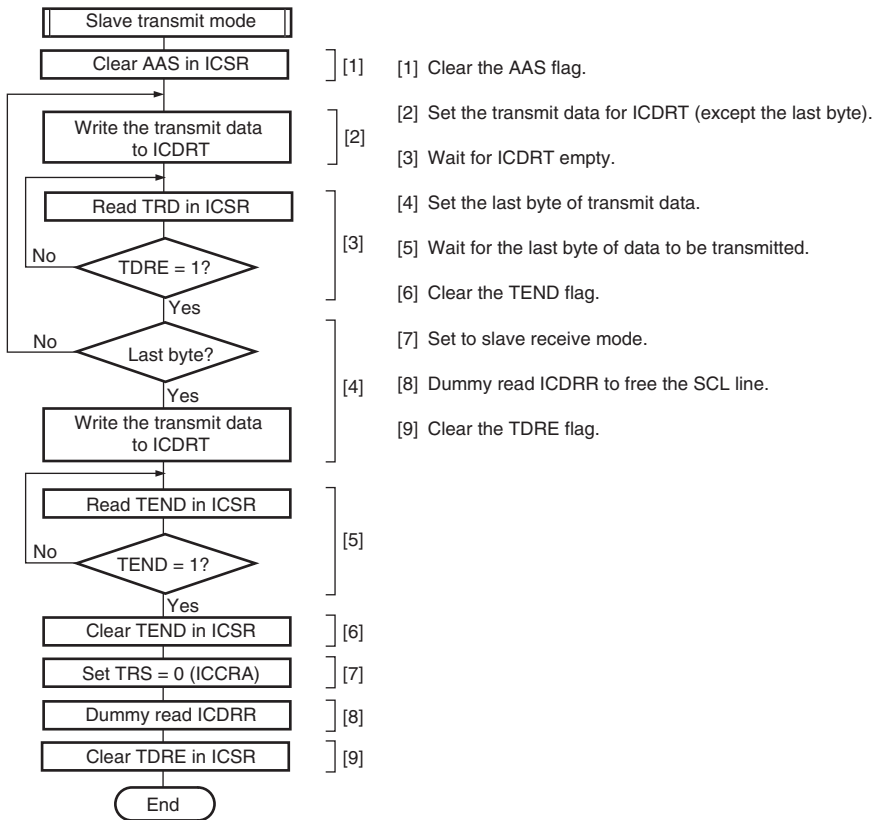


**Figure 17.14 Sample Flowchart of Master Transmit Mode**

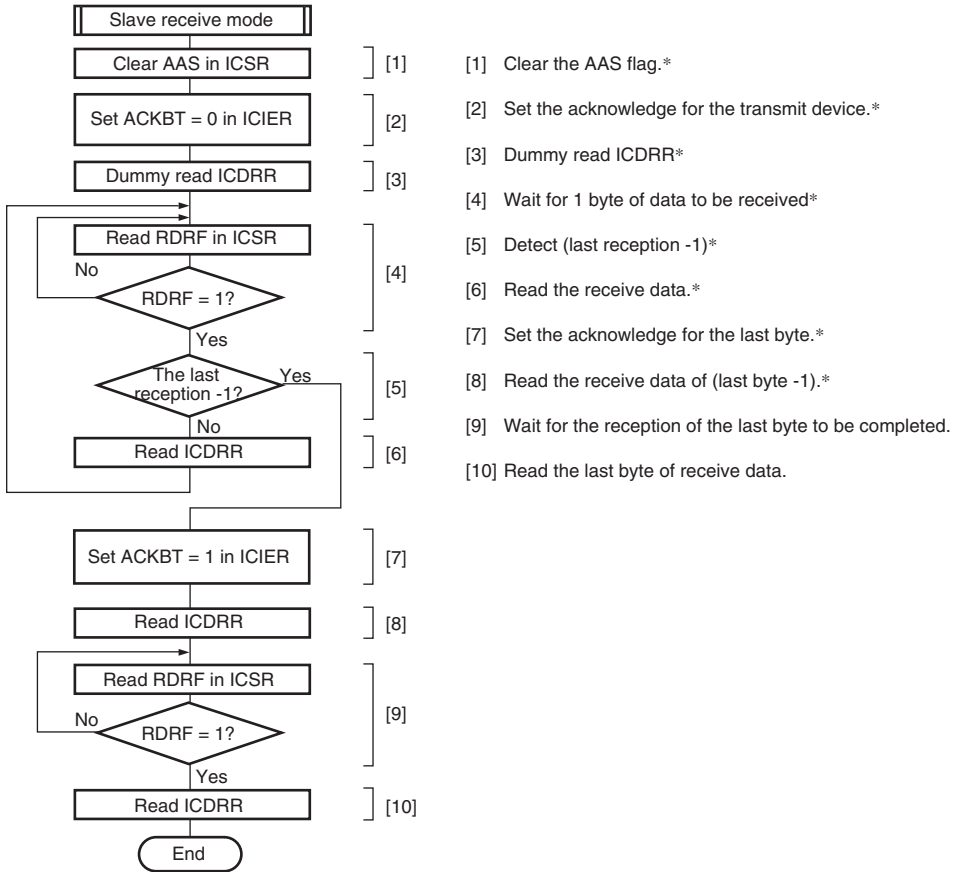


Note: 1. Do not generate an interrupt during steps [1] to [3].  
 2. For one-byte reception, steps [2] to [6] do not need to be executed. After step [1], execute step [7]. In step [8], read ICDRR (dummy read).

**Figure 17.15 Sample Flowchart for Master Receive Mode**



**Figure 17.16 Sample Flowchart for Slave Transmit Mode**



Note: \* For one-byte reception, steps [2] to [6] do not need to be executed. After step [1], execute step [7].  
 In step [8], read ICDRR (dummy read).

**Figure 17.17 Sample Flowchart for Slave Receive Mode**



## 17.5 Interrupt Request

There are six interrupt requests in this module; transmit data empty, transmit end, receive data full, NACK detection, STOP recognition, and arbitration lost. Table 17.3 shows the contents of each interrupt request.

**Table 17.3 Interrupt Requests**

Interrupt Request	Abbreviation	Interrupt Condition
Transmit Data Empty	TXI	$(TDRE = 1) \cdot (TIE = 1)$
Transmit End	TEI	$(TEND = 1) \cdot (TEIE = 1)$
Receive Data Full	RXI	$(RDRF = 1) \cdot (RIE = 1)$
Stop Recognition	STPI	$(STOP = 1) \cdot (STIE = 1)$
NACK Detection	NAKI	$\{(NACKF = 1) + (AL = 1)\} \cdot (NAKIE = 1)$
Arbitration Lost		

## 17.6 Bit Synchronous Circuit

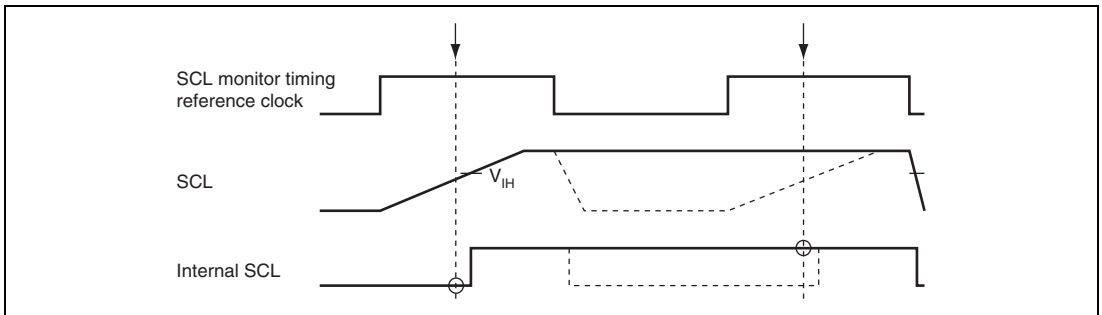
This module has a possibility that the high-level period is shortened in the two states described below.

In master mode,

- When SCL is driven low by the slave device
- When the rising speed of SCL is lowered by the load on the SCL line (load capacitance or pull-up resistance)

Therefore, this module monitors SCL and communicates bit by bit in synchronization.

Figure 17.18 shows the timing of the bit synchronous circuit, and table 17.4 shows the time when SCL output changes from low to Hi-Z and the period which SCL is monitored.



**Figure 17.18 Timing of the Bit Synchronous Circuit**

**Table 17.4 Time for Monitoring SCL**

CKS3	CKS2	Time for Monitoring SCL
0	0	7.5 t <sub>cyc</sub>
	1	19.5 t <sub>cyc</sub>
1	0	17.5 t <sub>cyc</sub>
	1	41.5 t <sub>cyc</sub>

## 17.7 Usage Notes

### 17.7.1 Module Stop Function Setting

Operation of the IIC2 can be disabled or enabled using the module stop control register. The initial setting is for operation of the IIC2 to be halted. Register access is enabled by clearing the module stop state. For details, see section 24, Power-Down Modes.

### 17.7.2 Issuance of Stop Condition and Repeated Start Condition

Confirm the ninth falling edge of the clock before issuing a stop or a repeated start condition. The ninth falling edge can be confirmed by monitoring the SCLO bit in the I<sup>2</sup>C bus control register B (ICCRB).

If a stop or a repeated start condition is issued at certain timing in either of the following cases, the stop or repeated start condition may be issued incorrectly.

- The rising time of the SCL signal exceeds the time given in section 17.6, Bit Synchronous Circuit, because of the load on the SCL bus (load capacitance or pull-up resistance).
- The bit synchronous circuit is activated because a slave device holds the SCL bus low during the eighth clock.

### 17.7.3 WAIT Bit

The WAIT bit in the I<sup>2</sup>C bus mode register (ICMR) must be held 0. If the WAIT bit is set to 1, when a slave device holds the SCL signal low more than one transfer clock cycle during the eighth clock, the high level period of the ninth clock may be shorter than a given period.

### 17.7.4 Restriction on Transfer Rate Setting Value in Multi-Master Mode

When the I<sup>2</sup>C transfer rate of this LSI is slower than that of any other master, the SCL signal may be output with an unexpected pulse width. To avoid this phenomenon, set the I<sup>2</sup>C transfer rate of this LSI to a value that is equal to or higher than 1/1.8 times the transfer rate of the fastest master. For example, if the fastest rate of other master is 400 kbps, the I<sup>2</sup>C transfer rate of this LSI should be at least 223 kbps (= 400/1.8).

### 17.7.5 Restriction on Bit Manipulation when Setting the MST and TRS Bits in Multi-Master Mode

If the MST and TRS bits are manipulated sequentially to select master transmit mode, a conflict state (for example, the AL bit in ICSR is set to 1 in master transmit mode (MST = 1, TRS = 1)) can result depending on the timing of arbitration lost that might occur during execution of the bit manipulation instruction for the TRS bit. This phenomenon can be avoided by the following operations.

- In multi-master mode, use the MOV instruction to set the MST and TRS bits.
- If arbitration is lost, check to see whether both MST and TRS bits have been cleared to 0. If both bits are not clear, clear them to 0.

### 17.7.6 Notes on Master Receive Mode

In master receive mode, when the value of RDRF is 1 at the falling edge of the eighth clock pulse, the SCL signal is pulled low. If ICDRR is read near the falling edge of the eighth clock pulse, SCL is fixed to low only during the eighth clock cycle of the next received data and, after that, SCL is released even if ICDRR is not read, which allows the ninth clock pulse to be output. As a result, some data fails to be received. This phenomenon can be avoided by the following operations.

- In master receive mode, read ICDRR before the rising edge of the eighth clock pulse.
- In master receive mode, set the RCVD bit to 1 and perform byte-wise communication.

## Section 18 A/D Converter

This LSI includes a successive approximation type 10-bit A/D converter that allows up to eight analog input channels to be selected.

Figure 18.1 shows a block diagram of the A/D converter.

### 18.1 Features

- 10-bit resolution
- Eight input channels
- Conversion cycles: 5.33  $\mu$ s per channel (with ADCLK at 7.5 MHz operation)
- Two kinds of operating modes
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels, or 1 to 8 channels
- Eight data registers
  - A/D conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three types of conversion start
  - Conversion can be started by software, a conversion start trigger from the 16-bit timer pulse unit (TPU)\* or 8-bit timer (TMR)\*, or an external trigger signal.
- Interrupt source
  - A/D conversion end interrupt (ADI) request can be generated.
- Module stop state specifiable

Note: \* Starting by a trigger from the TPU/TMR is available on the on-chip emulator but not available on other emulators.

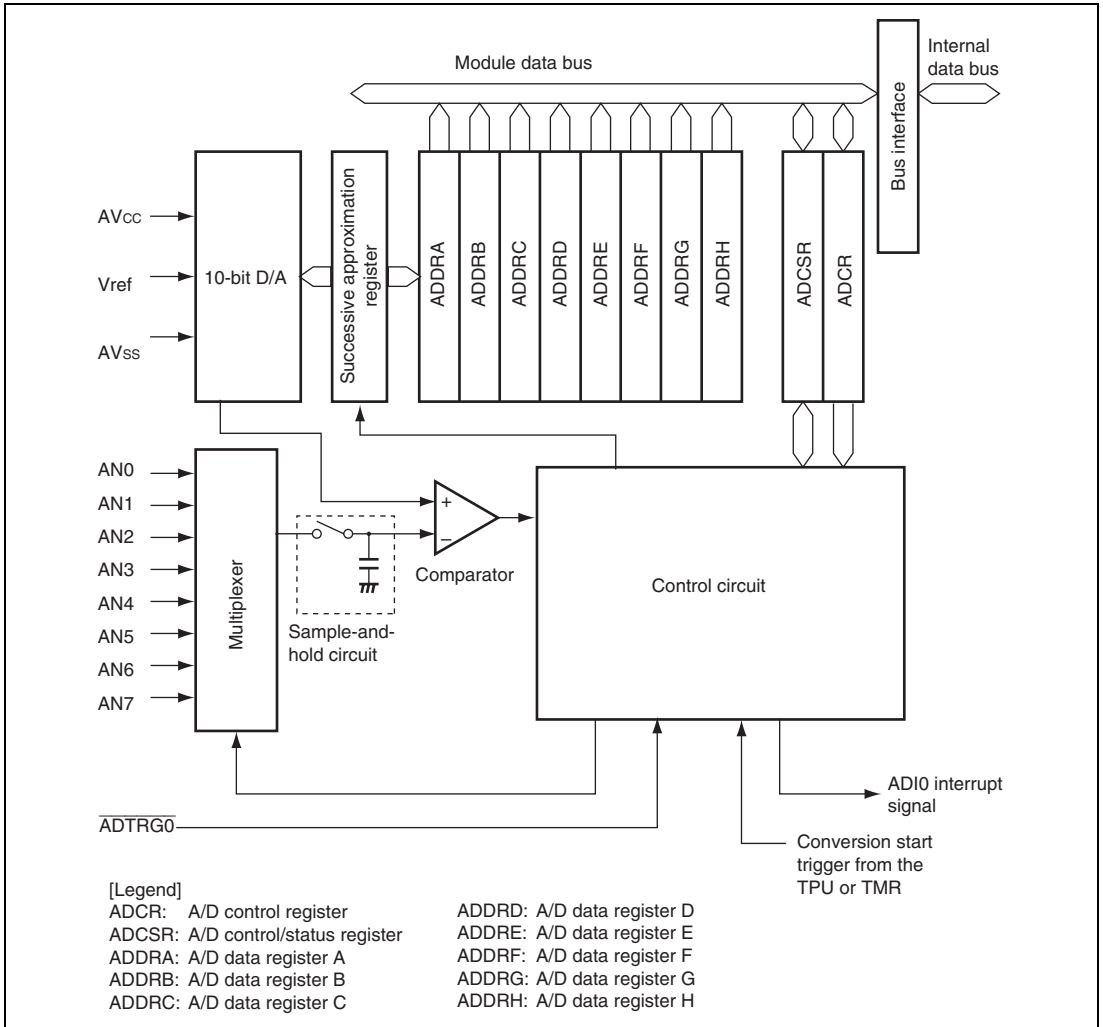


Figure 18.1 Block Diagram of A/D Converter

## 18.2 Input/Output Pins

Table 18.1 shows the pin configuration of the A/D converter.

**Table 18.1 Pin Configuration**

Pin Name	Symbol	I/O	Function
Analog input pin 0	AN0	Input	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	
Analog input pin 5	AN5	Input	
Analog input pin 6	AN6	Input	
Analog input pin 7	AN7	Input	
A/D external trigger input pin	ADTRG0	Input	External trigger input for starting A/D conversion
Analog power supply pin	AV <sub>CC</sub>	Input	Analog block power supply
Analog ground pin	AV <sub>SS</sub>	Input	Analog block ground
Reference voltage pin	Vref	Input	A/D conversion reference voltage

## 18.3 Register Descriptions

The A/D converter has the following registers.

- A/D data register A (ADDRA)
- A/D data register B (ADDRB)
- A/D data register C (ADDRC)
- A/D data register D (ADDRD)
- A/D data register E (ADDRE)
- A/D data register F (ADDRF)
- A/D data register G (ADDRG)
- A/D data register H (ADDRH)
- A/D control/status register (ADCSR)
- A/D control register (ADCR)

### 18.3.1 A/D Data Registers A to H (ADDRA to ADDRH)

There are eight 16-bit read-only ADDR registers, ADDRA to ADDRH, used to store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 18.2.

The converted 10-bit data is stored in bits 15 to 6. The lower 6-bit data is always read as 0.

The data bus between the CPU and the A/D converter has a 16-bit width. The data can be read directly from the CPU. ADDR must not be accessed in 8-bit units and must be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Name											—	—	—	—	—	—
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

**Table 18.2 Analog Input Channels and Corresponding ADDR Registers**

Analog Input Channel	A/D Data Register Which Stores Conversion Result
AN0	ADDRA
AN1	ADDRB
AN2	ADDRC
AN3	ADDRD
AN4	ADDRE
AN5	ADDRF
AN6	ADDRG
AN7	ADDRH



### 18.3.2 A/D Control/Status Register (ADCSR)

ADCSR controls A/D conversion operations.

Bit	7	6	5	4	3	2	1	0
Bit Name	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)*	R/W	R/W	R	R/W	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W)*	<p>A/D End Flag</p> <p>A status flag that indicates the end of A/D conversion.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>When A/D conversion ends in single mode</li> <li>When A/D conversion ends on all specified channels in scan mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written after reading ADF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>When the DTC or DMAC is activated by an ADI interrupt and ADDR is read</li> </ul>
6	ADIE	0	R/W	<p>A/D Interrupt Enable</p> <p>When this bit is set to 1, ADI interrupts by ADF are enabled.</p>
5	ADST	0	R/W	<p>A/D Start</p> <p>Clearing this bit to 0 stops A/D conversion, and the A/D converter enters wait state.</p> <p>Setting this bit to 1 starts A/D conversion. In single mode, this bit is cleared to 0 automatically when A/D conversion on the specified channel ends. In scan mode, A/D conversion continues sequentially on the specified channels until this bit is cleared to 0 by software, a reset, or hardware standby mode.</p>

Bit	Bit Name	Initial Value	R/W	Description
4	—	0	R	Reserved This is a read-only bit and cannot be modified.
3	CH3	0	R/W	Channel Select 3 to 0
2	CH2	0	R/W	Selects analog input together with bits SCANE and SCANS in ADCR.
1	CH1	0	R/W	<ul style="list-style-type: none"> <li>• When SCANE = 0 and SCANS = X               <ul style="list-style-type: none"> <li>0000: AN0</li> <li>0001: AN1</li> <li>0010: AN2</li> <li>0011: AN3</li> <li>0100: AN4</li> <li>0101: AN5</li> <li>0110: AN6</li> <li>0111: AN7</li> <li>1XXX: Setting prohibited</li> </ul> </li> <li>• When SCANE = 1 and SCANS = 0               <ul style="list-style-type: none"> <li>0000: AN0</li> <li>0001: AN0 and AN1</li> <li>0010: AN0 to AN2</li> <li>0011: AN0 to AN3</li> <li>0100: AN4</li> <li>0101: AN4 and AN5</li> <li>0110: AN4 to AN6</li> <li>0111: AN4 to AN7</li> <li>1XXX: Setting prohibited</li> </ul> </li> <li>• When SCANE = 1 and SCANS = 1               <ul style="list-style-type: none"> <li>0000: AN0</li> <li>0001: AN0 and AN1</li> <li>0010: AN0 to AN2</li> <li>0011: AN0 to AN3</li> <li>0100: AN0 to AN4</li> <li>0101: AN0 to AN5</li> <li>0110: AN0 to AN6</li> <li>0111: AN0 to AN7</li> <li>1XXX: Setting prohibited</li> </ul> </li> </ul>
0	CH0	0	R/W	

## [Legend]

X: Don't care

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

### 18.3.3 A/D Control Register (ADCR)

ADCR enables A/D conversion to be started by an external trigger input.

Bit	7	6	5	4	3	2	1	0
Bit Name	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	ADSTCLR	EXTRGS
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	Bit Name	Initial Value	R/W	Description
7	TRGS1	0	R/W	Timer Trigger Select 1 and 0, Extended Trigger Select
6	TRGS0	0	R/W	These bits select enabling or disabling of the start of A/D conversion by a trigger signal.
0	EXTRGS	0	R/W	<p>000: A/D conversion start by external trigger is disabled.</p> <p>010: A/D conversion start by conversion trigger from TPU is enabled.</p> <p>100: A/D conversion start by conversion trigger from TMR is enabled.</p> <p>110: A/D conversion start by the <math>\overline{\text{ADTRG0}}</math> pin is enabled.*</p> <p>001: Setting prohibited</p> <p>011: Setting prohibited</p> <p>101: Setting prohibited</p> <p>111: Setting prohibited</p>
5	SCANE	0	R/W	Scan Mode
4	SCANS	0	R/W	<p>These bits select the A/D conversion operating mode.</p> <p>0X: Single mode</p> <p>10: Scan mode. A/D conversion is performed continuously for channels 1 to 4.</p> <p>11: Scan mode. A/D conversion is performed continuously for channels 1 to 8.</p>

Bit	Bit Name	Initial Value	R/W	Description
3	CKS1	0	R/W	Clock Select 1 and 0
2	CKS0	0	R/W	<p>These bits set the A/D conversion time.</p> <p>Set the A/D conversion time while the ADST bit in ADCSR is 0, and then set the conversion mode.</p> <p>00: Conversion time = 46 states (max), ADCLK = <math>P\phi</math></p> <p>01: Conversion time = 87 states (max), ADCLK = <math>P\phi/2</math></p> <p>10: Conversion time = 168 states (max), ADCLK = <math>P\phi/4</math></p> <p>11: Conversion time = 332 states (max), ADCLK = <math>P\phi/8</math></p>
1	ADSTCLR	0	R/W	<p>A/D Start Clear</p> <p>This bit sets automatic clearing of the ADST bit in scan mode.</p> <p>0: Prohibits automatic clearing of the ADST bit in scan mode.</p> <p>1: Performs automatic clearing in scan mode if all the selected channels complete A/D conversion.</p>

## [Legend]

X: Don't care

Note: \* To set A/D conversion to start by the ADTRG0 pin, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 11, I/O Ports.

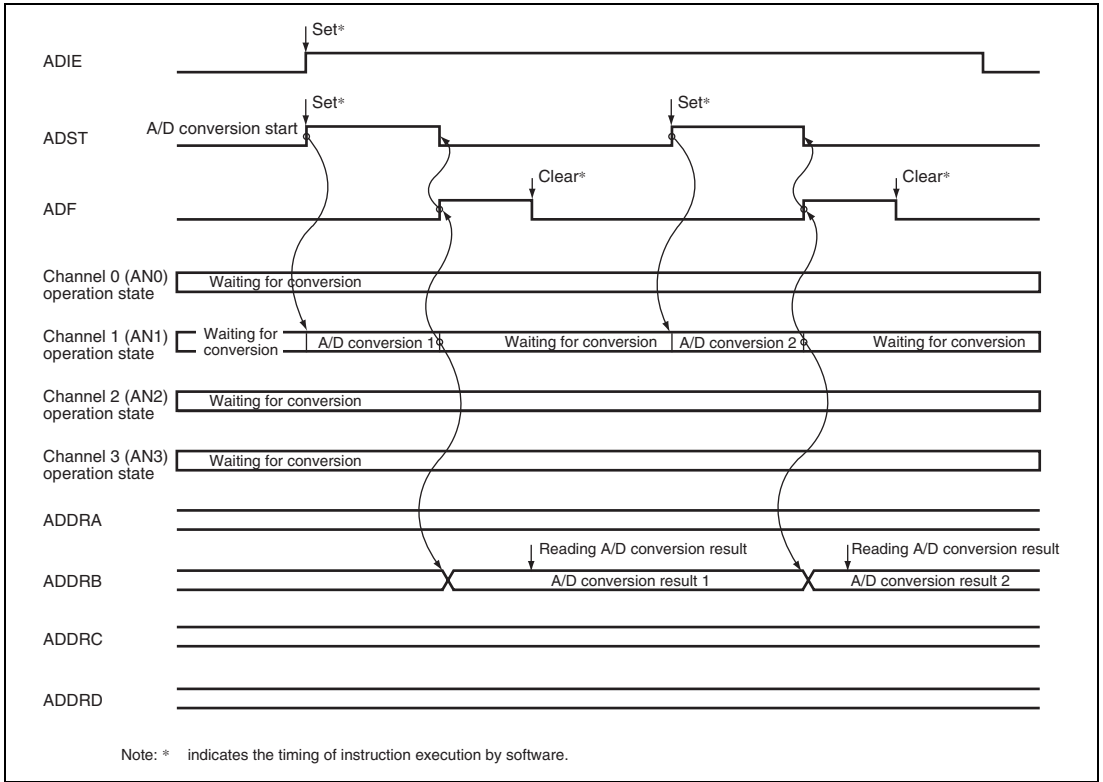
## 18.4 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes: single mode and scan mode. When configuring the A/D converter, first set the clock for A/D conversion. Before changing the operating mode or analog input channel, clear the ADST bit in ADCSR to 0 to stop A/D conversion to prevent incorrect operation. The ADST bit can be set to 1 at the same time as the operating mode or analog input channel is changed.

### 18.4.1 Single Mode

In single mode, A/D conversion is to be performed only once on the analog input of the specified single channel.

1. A/D conversion for the selected channel is started when the ADST bit in ADCSR is set to 1 by software, TPU, TMR, or an external trigger input.
2. When A/D conversion is completed, the A/D conversion result is transferred to the corresponding A/D data register of the channel.
3. When A/D conversion is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when A/D conversion ends. The A/D converter enters wait state. If the ADST bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters wait state.



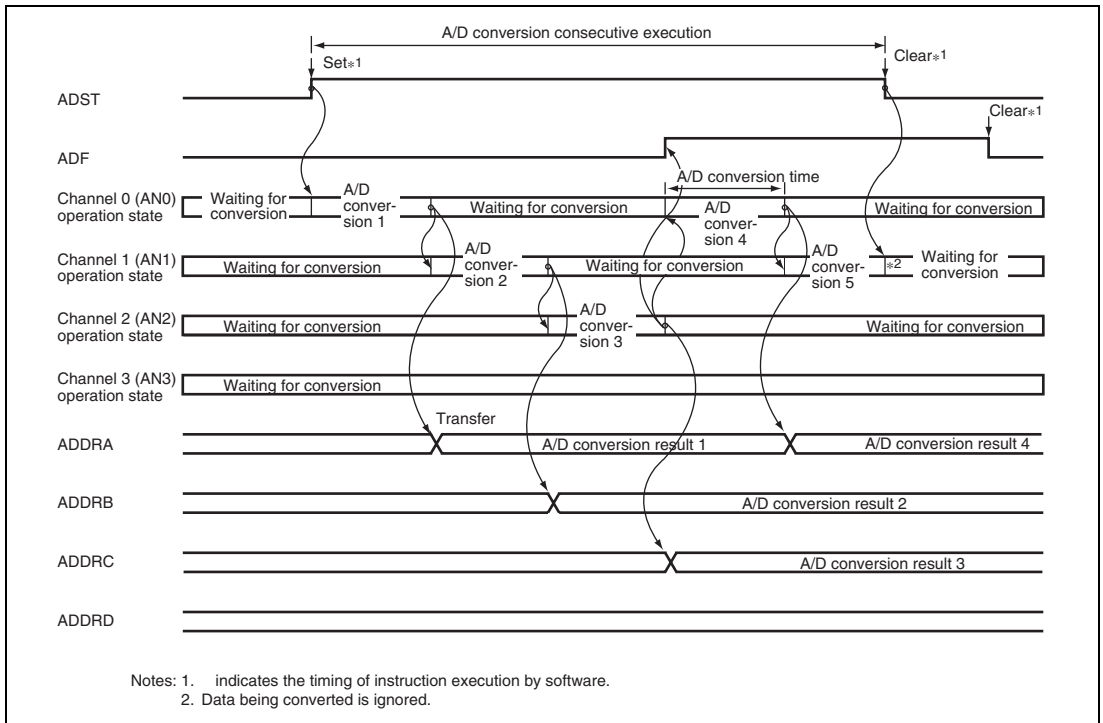
**Figure 18.2 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)**

## 18.4.2 Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the analog inputs of the specified channels up to four or eight\* channels. Two types of scan mode are provided, that is, continuous scan mode where A/D conversion is repeatedly performed and one-cycle scan mode where A/D conversion is performed for the specified channels for one cycle.

### (1) Continuous Scan Mode

1. When the ADST bit in ADCSR is set to 1 by software, TPU, TMR, or an external trigger input, A/D conversion starts on the first channel in the specified channel group. Consecutive A/D conversion on a maximum of four channels (SCANE and SCANS = B'10) or on a maximum of eight channels (SCANE and SCANS = B'11) can be selected. When consecutive A/D conversion is performed on four channels, A/D conversion starts on AN0 when CH3 and CH2 = B'00, whereas starts on AN4 when CH3 and CH2 = B'01. When consecutive A/D conversion is performed on eight channels, A/D conversion starts on AN0 when CH3 = B'0.
2. When A/D conversion for each channel is completed, the A/D conversion result is sequentially transferred to the corresponding ADDR of each channel.
3. When A/D conversion of all selected channels is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. A/D conversion of the first channel in the group starts again.
4. The ADST bit is not cleared automatically, and steps 2 to 3 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters wait state. If the ADST bit is later set to 1, A/D conversion starts again from the first channel in the group.



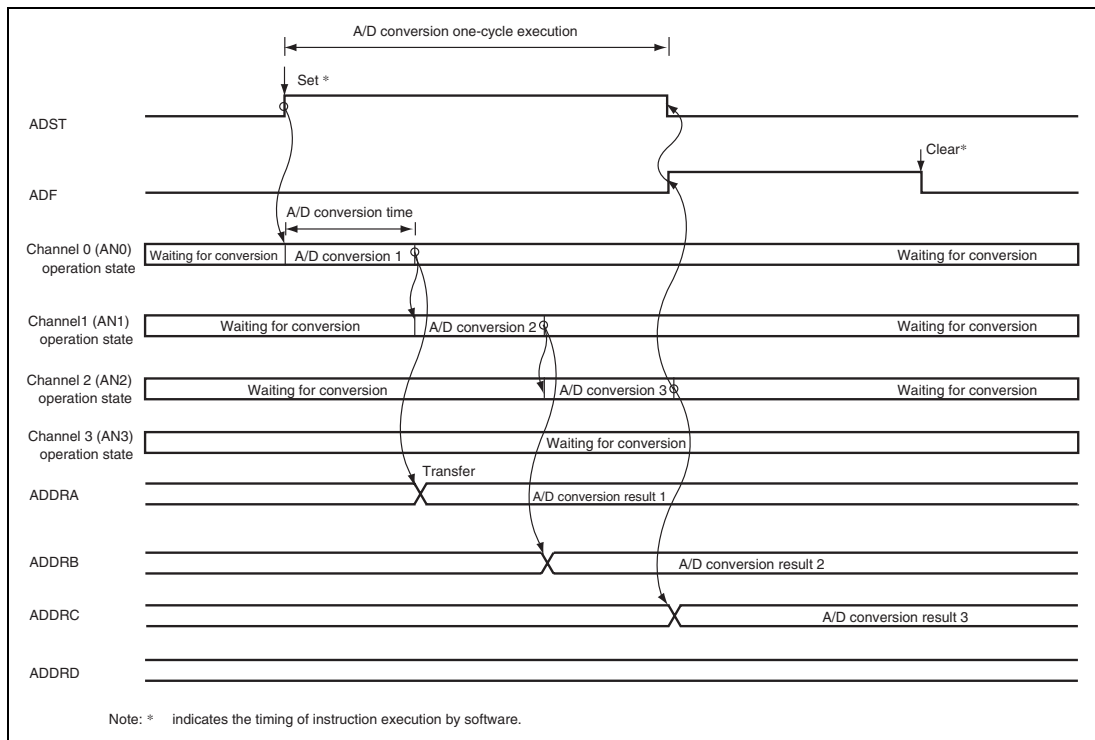
**Figure 18.3 Example of A/D Conversion  
(Continuous Scan Mode, Three Channels (AN0 to AN2) Selected)**

## (2) One-Cycle Scan Mode

1. Set the ADSTCLR bit in ADCR to 1.
2. When the ADST bit in ADCSR is set to 1 by software, TPU, TMR, or an external trigger input, A/D conversion starts on the first channel in the specified channel group. Consecutive A/D conversion on a maximum of four channels (SCANE and SCANS = B'10) or on a maximum of eight channels (SCANE and SCANS = B'11) can be selected. When consecutive A/D conversion is performed on four channels, A/D conversion starts on AN0 when CH3 and CH2 = B'00, whereas starts on AN4 when CH3 and CH2 = B'01. When consecutive A/D conversion is performed on eight channels, A/D conversion starts on AN0 when CH3 = B'0.
3. When A/D conversion for each channel is completed, the A/D conversion result is sequentially transferred to the corresponding ADDR of each channel.
4. When A/D conversion of all selected channels is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.



5. The ADST bit is automatically cleared when A/D conversion is completed for all of the channels that have been selected. A/D conversion stops and the A/D converter enters a wait state.



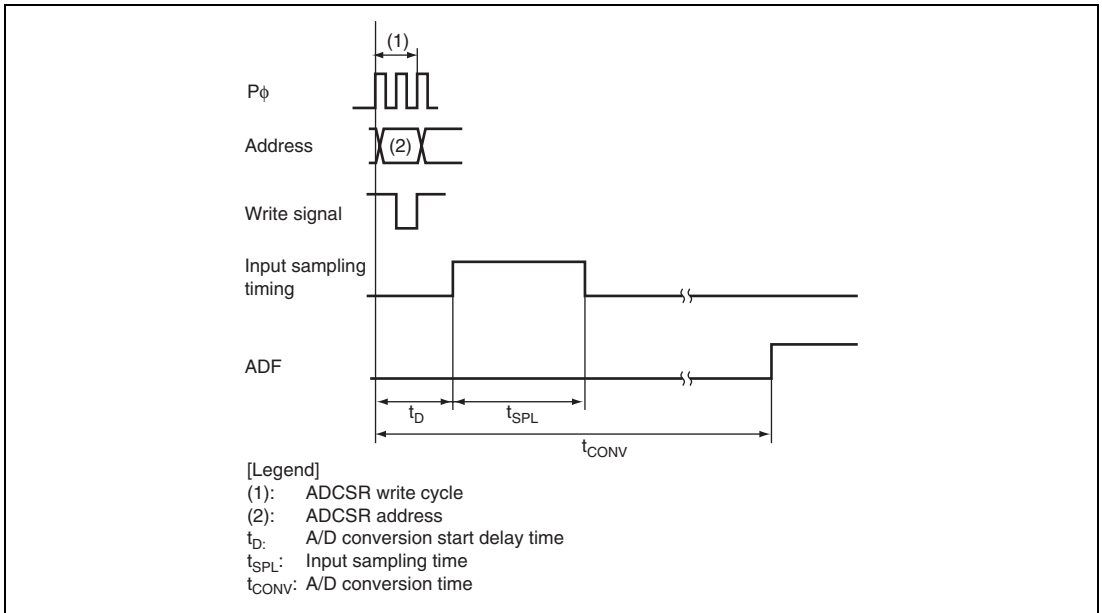
**Figure 18.4 Example of A/D Conversion  
(One-Cycle Scan Mode, Three Channels (AN0 to AN2) Selected)**

### 18.4.3 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 when the A/D conversion start delay time ( $t_d$ ) passes after the ADST bit in ADCSR is set to 1, then starts A/D conversion. Figure 18.5 shows the A/D conversion timing. Table 18.3 indicates the A/D conversion time.

As indicated in figure 18.5, the A/D conversion time ( $t_{CONV}$ ) includes  $t_d$  and the input sampling time ( $t_{SPL}$ ). 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 18.3.

In scan mode, the values given in table 18.3 apply to the first conversion time. The values given in table 18.4 apply to the second and subsequent conversions. In either case, bits CKS1 and CKS0 in ADCR should be set so that the conversion time is within the ranges indicated by the A/D conversion characteristics.



**Figure 18.5 A/D Conversion Timing**

**Table 18.3 A/D Conversion Characteristics (Single Mode)**

Item	Symbol	CKS1 = 0						CKS1 = 1					
		CKS0 = 0			CKS0 = 1			CKS0 = 0			CKS0 = 1		
		Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.
A/D conversion start delay time	t <sub>D</sub>	3	—	4	3	—	5	3	—	6	3	—	10
Input sampling time	t <sub>SPL</sub>	—	15	—	—	30	—	—	60	—	—	120	—
A/D conversion time	t <sub>CONV</sub>	45	—	46	85	—	87	165	—	168	325	—	332

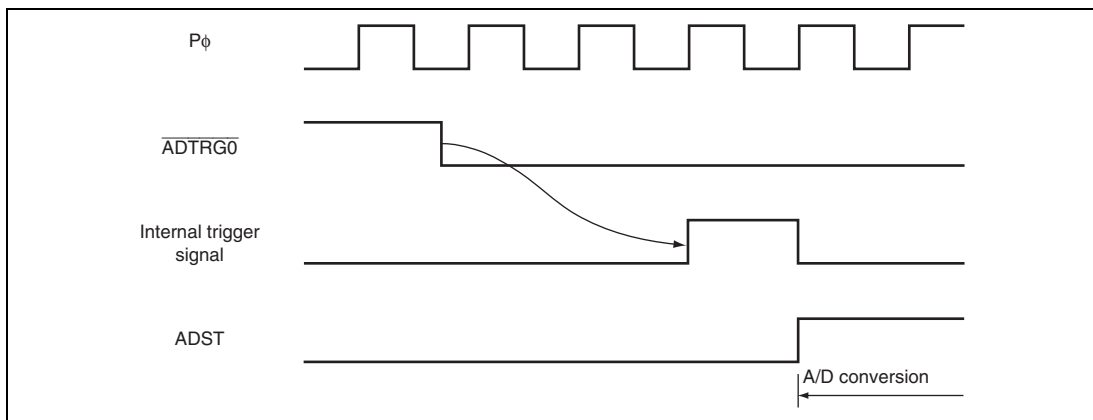
Note: Values in the table are the number of states.

**Table 18.4 A/D Conversion Characteristics (Scan Mode)**

CKS1	CKS0	Conversion Time (Number of States)
0	0	40 (Fixed)
	1	80 (Fixed)
1	0	160 (Fixed)
	1	320 (Fixed)

#### 18.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1, TRGS0, and EXTRGS bits in ADCR are set to B'110, an external trigger is input from the  $\overline{\text{ADTRG0}}$  pin. A/D conversion starts when the ADST bit in ADCSR is set to 1 on the falling edge of the  $\overline{\text{ADTRG0}}$  pin. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 18.6 shows the timing.

**Figure 18.6 External Trigger Input Timing**

## 18.5 Interrupt Source

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 when the ADF bit in ADCSR is set to 1 after A/D conversion is completed enables ADI interrupt requests. The DMA controller (DMAC) can be activated by an ADI interrupt. Having the converted data read by the DMAC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

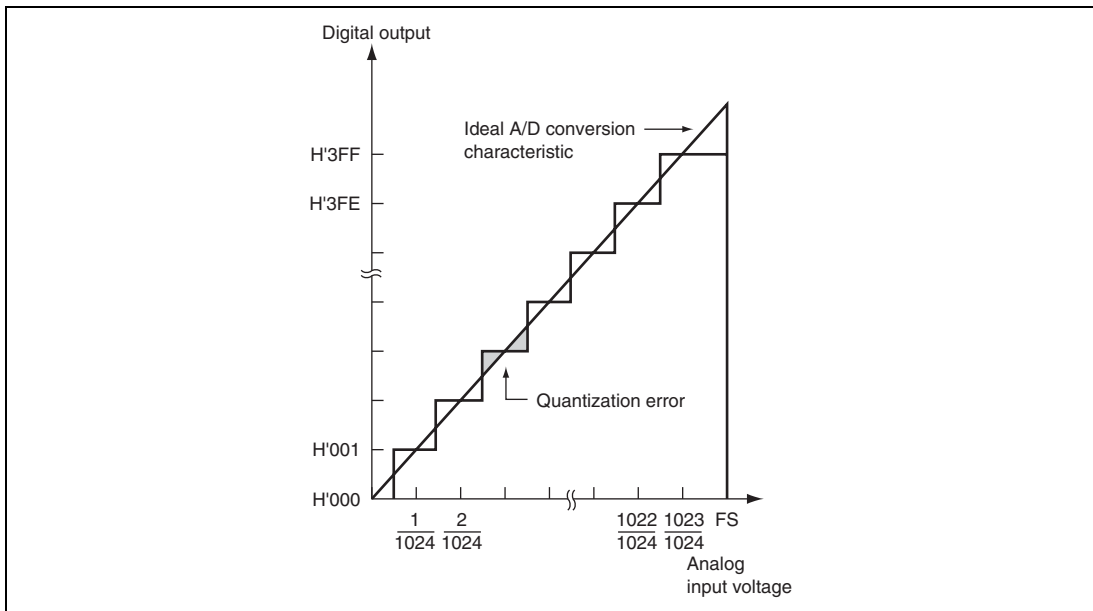
**Table 18.5 A/D Converter Interrupt Source**

Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation
ADI	A/D conversion end	ADF	Impossible	Possible

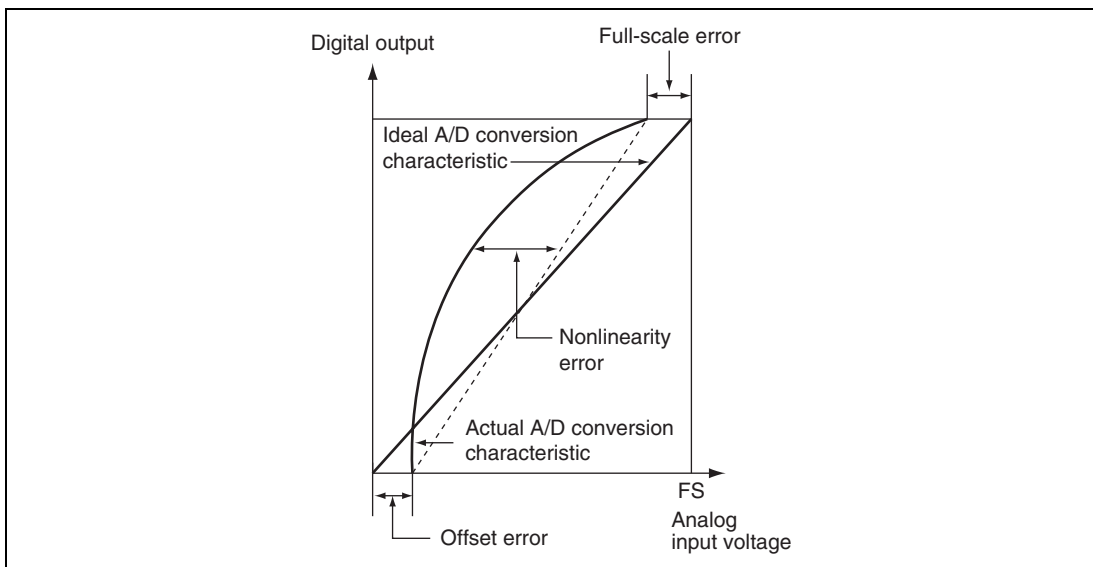
## 18.6 A/D Conversion Accuracy Definitions

This LSI's A/D conversion accuracy definitions are given below.

- Resolution  
The number of A/D converter digital output codes.
- Quantization error  
The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 18.7).
- 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'000000000 (H'000) to B'000000001 (H'001) (see figure 18.8).
- 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'111111110 (H'3FE) to B'111111111 (H'3FF) (see figure 18.8).
- 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 (see figure 18.8).
- Absolute accuracy  
The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.



**Figure 18.7 A/D Conversion Accuracy Definitions**



**Figure 18.8 A/D Conversion Accuracy Definitions**

## 18.7 Usage Notes

### 18.7.1 Module Stop Function Setting

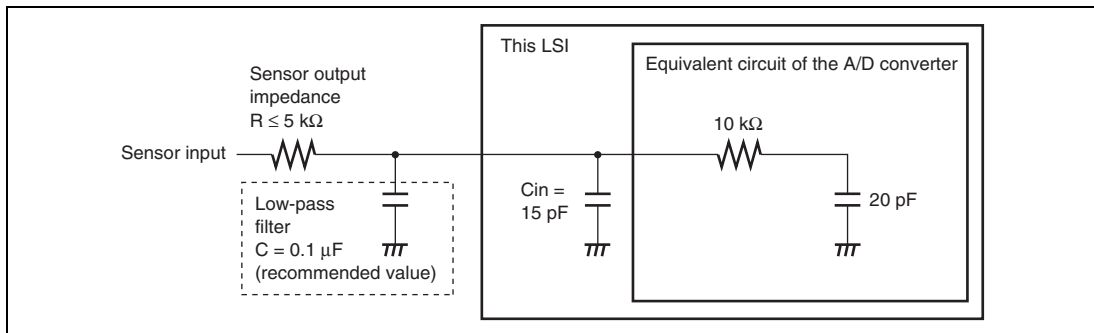
Operation of the A/D converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the A/D converter to be halted. Register access is enabled by clearing the module stop state. When placing the A/D converter in the module stop state after it performed A/D conversion, be sure to set both of the CKS1 and CKS0 bits to 1 and clear all of the ADST, TRGS1, TRGS0, and EXTRGS bits to 0 to disable A/D conversion. After that, dummy-read the ADCSR register and then set the module stop control register. For details on the module stop control register, see section 24, Power-Down Modes.

### 18.7.2 A/D Input Hold Function in Software Standby Mode

When this LSI enters software standby mode with A/D conversion enabled, the analog inputs are retained, and the analog power supply current is equal to as during A/D conversion. If the analog power supply current needs to be reduced in software standby mode, set both of the CKS1 and CKS0 bits to 1 and clear all of the ADST, TRGS1, TRGS0, and EXTRGS bits to 0 to disable A/D conversion. After that, dummy-read the ADCSR register and then enter software standby mode.

### 18.7.3 Permissible Signal Source Impedance

This LSI's analog input is designed so that the conversion accuracy is guaranteed for an input signal for which the signal source impedance is 5 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 5 k $\Omega$ , charging may be insufficient and it may not be possible to guarantee the A/D conversion accuracy. However, if a large capacitance is provided externally for conversion in single mode, 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) (see figure 18.9). When converting a high-speed analog signal or conversion in scan mode, a low-impedance buffer should be inserted.



**Figure 18.9 Example of Analog Input Circuit**

### 18.7.4 Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute accuracy. Be sure to make the connection to an electrically stable GND such as AVSS.

Care is also required to insure that filter circuits do not communicate with digital signals on the mounting board, acting as antennas.

### 18.7.5 Setting Range of Analog Power Supply and Other Pins

If the conditions shown below are not met, the reliability of the LSI may be adversely affected.

- Analog input voltage range  
The voltage applied to analog input pin ANn during A/D conversion should be in the range  $AV_{SS} \leq V_{AN} \leq V_{ref}$ .
- Relation between AVCC, AVSS and VCC, VSS  
As the relationship between AVCC, AVSS and VCC, VSS, set  $AV_{CC} = V_{CC} \pm 0.3 \text{ V}$  and  $AV_{SS} = V_{SS}$ . If the A/D converter is not used, set  $AV_{CC} = V_{CC}$  and  $AV_{SS} = V_{SS}$ .
- Vref setting range  
The reference voltage at the Vref pin should be set in the range  $V_{ref} \leq AV_{CC}$ .

### 18.7.6 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.

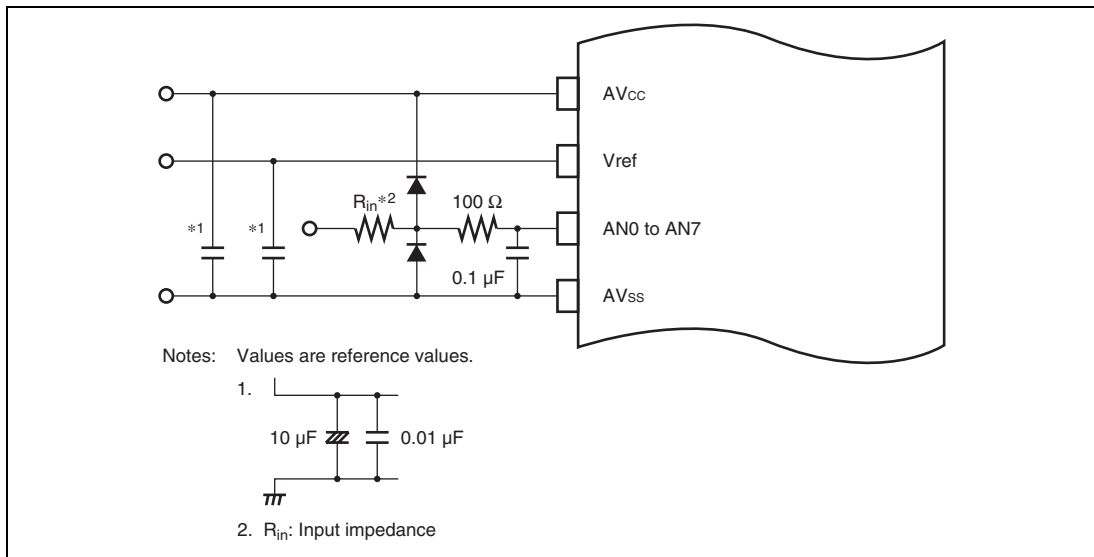
Digital circuitry must be isolated from the analog input pins (AN0 to AN7), analog reference power supply (Vref), and analog power supply (AVcc) by the analog ground (AVss). Also, the analog ground (AVss) should be connected at one point to a stable ground (Vss) on the board.

### 18.7.7 Notes on Countermeasure against Noise

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) should be connected between AVcc and AVss as shown in figure 18.10. Also, the bypass capacitors connected to AVcc and the filter capacitor connected to the AN0 to AN7 pins must be connected to AVss.

If a filter capacitor is connected, the input currents at the AN0 to AN7 pins 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.

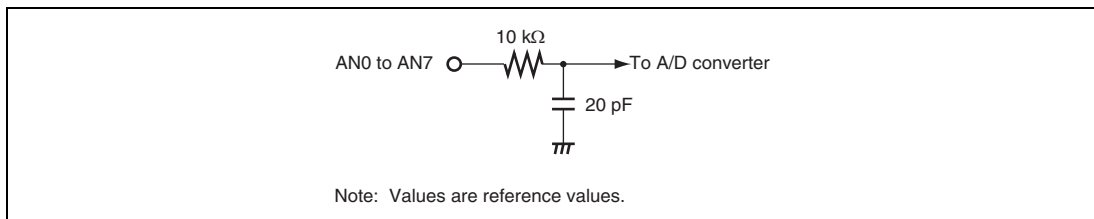




**Figure 18.10 Example of Analog Input Protection Circuit**

**Table 18.6 Analog Pin Specifications**

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Permissible signal source impedance	—	5	k $\Omega$



**Figure 18.11 Analog Input Pin Equivalent Circuit**



## Section 19 $\Delta\Sigma$ A/D Converter

This LSI includes a  $\Delta\Sigma$  modulation-type 16-bit  $\Delta\Sigma$  A/D converter. This module accepts up to six analog inputs, each of which is internally amplified eight-fold and then sequentially converted by time-division multiplexing.

A block diagram of the  $\Delta\Sigma$  A/D converter is shown in figure 19.1.

### 19.1 Features

- 16-bit resolution
- Suitable for sensor applications (cannot be applied to voice and audio applications)
- Conversion method:  $\Delta\Sigma$  modulation-based conversion
- Six input channels (time-division multiplexing)
- Two types of input channel (four single-ended channels and two differential input channels)  
Offset cancellation by 10-bit DAC on single-ended channels
- Conversion time: 91.5  $\mu$ s per channel  
(for 286-“state” conversion at  $A\phi = 25$  MHz, where 1 state =  $A\phi/8$ )
- Six data registers  
Results of A/D conversion are stored in 16-bit registers for the respective channels.
- Three ways of starting A/D conversion
  - Software
  - Trigger from the 16-bit timer pulse unit (TPU)\* or 8-bit timer (TMR)\*
  - External trigger signal
- Interrupt source  
Generates  $\Delta\Sigma$  A/D conversion end interrupt requests (DSADI).
- Can be placed in the module stop state

Note: \* Initiation of conversion by a TPU/TMR trigger is available with the on-chip emulator but not with other emulators.

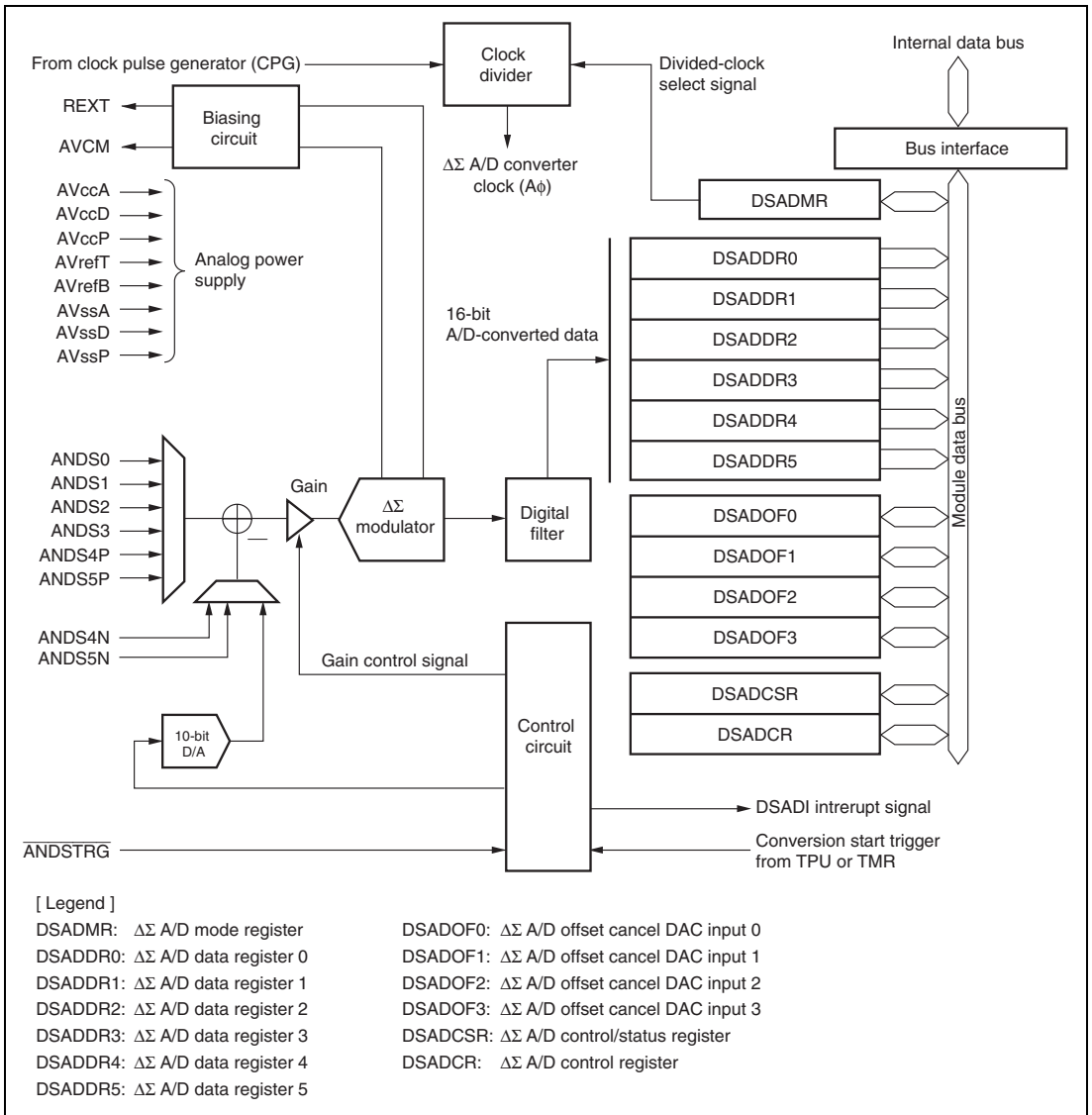


Figure 19.1 Block Diagram of the  $\Delta\Sigma$  A/D Converter

## 19.2 Input/Output Pins

Table 19.1 shows the pins used by the  $\Delta\Sigma$  A/D converter.

**Table 19.1 Pin Configuration**

Pin Name	Abbreviation	I/O	Function
Analog input pin 0	ANDS0	Input	Analog input pins: Single-ended input
Analog input pin 1	ANDS1	Input	
Analog input pin 2	ANDS2	Input	
Analog input pin 3	ANDS3	Input	Analog input pins: Differential input
Analog input pin 4-P	ANDS4P	Input	
Analog input pin 4-N	ANDS4N	Input	
Analog input pin 5-P	ANDS5P	Input	Analog input pins: Differential input
Analog input pin 5-N	ANDS5N	Input	
External trigger input pin for $\Delta\Sigma$ A/D converter	ANDSTRG	Input	External trigger input pin for starting $\Delta\Sigma$ A/D conversion
Analog power supply pin	AVccA* <sup>1</sup>	Input	Power supply pin for the analog section of the $\Delta\Sigma$ A/D converter
Analog power supply pin	AVccD* <sup>1</sup>	Input	Power supply pin for the control circuit of the $\Delta\Sigma$ A/D converter
Analog power supply pin	AVccP* <sup>1</sup>	Input	Power supply pin for the input pin control circuit of the $\Delta\Sigma$ A/D converter
Analog ground pin	AVssA	Input	Ground pin for the analog section of the $\Delta\Sigma$ A/D converter
Analog ground pin	AVssD	Input	Ground pin for the control circuit of the $\Delta\Sigma$ A/D converter
Analog ground pin	AVssP	Input	Ground pin for the input pin control circuit of the $\Delta\Sigma$ A/D converter
$\Delta\Sigma$ reference voltage (high)	AVrefT* <sup>2</sup>	Input	For connection of stabilizing capacitors (between AV <sub>ref</sub> B and AV <sub>ref</sub> T; 10 $\mu$ F + 0.1 $\mu$ F)
$\Delta\Sigma$ reference voltage (low)	AVrefB* <sup>2</sup>	Input	
Reference voltage pin	AVCM	Output	For connection of a stabilizing capacitor (0.1 $\mu$ F between AVCM and AV <sub>ss</sub> A)
Reference current pin	REXT	Output	For connection of an external resistor between REXT and AV <sub>ss</sub> A. (51 k $\Omega$ with $\pm$ 1% tolerance)

Notes: 1. AVccA = AVccD = AVccP must always hold.

2. AVccA = AVrefT, AVrefT > AVrefB, AVrefB = AVssA must always hold.

### 19.3 Register Descriptions

The  $\Delta\Sigma$  A/D converter has the following registers.

- $\Delta\Sigma$  A/D data register 0 (DSADDR0)
- $\Delta\Sigma$  A/D data register 1 (DSADDR1)
- $\Delta\Sigma$  A/D data register 2 (DSADDR2)
- $\Delta\Sigma$  A/D data register 3 (DSADDR3)
- $\Delta\Sigma$  A/D data register 4 (DSADDR4)
- $\Delta\Sigma$  A/D data register 5 (DSADDR5)
- $\Delta\Sigma$  A/D offset cancel DAC input 0 (DSADOF0)\*
- $\Delta\Sigma$  A/D offset cancel DAC input 1 (DSADOF1)\*
- $\Delta\Sigma$  A/D offset cancel DAC input 2 (DSADOF2)\*
- $\Delta\Sigma$  A/D offset cancel DAC input 3 (DSADOF3)\*
- $\Delta\Sigma$  A/D control/status register (DSADCSR)
- $\Delta\Sigma$  A/D control register (DSADCR)
- $\Delta\Sigma$  A/D mode register (DSADMR)

Note: \* Offset cancellation here means canceling DC components of the signals input to analog input pins ANDS0 to ANDS3.

### 19.3.1 $\Delta\Sigma$ A/D Mode Register (DSADMR)

DSADMR controls the biasing circuit and selects a clock for the  $\Delta\Sigma$  A/D converter. DSADMR can be read by the CPU at any time, but must be written to while the  $\Delta\Sigma$  A/D converter is in the module stop state.

Bit	7	6	5	4	3	2	1	0
Bit Name	BIASE	—	—	—	—	ACK2	ACK1	ACK0
Initial Value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	BIASE	0	R/W	Biasing Circuit Control Controls whether the biasing circuit is stopped or runs. 0: Biasing circuit is stopped. 1: Biasing circuit runs.
6 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
2	ACK2	0	R/W	$\Delta\Sigma$ A/D Converter Clock Select
1	ACK1	0	R/W	These bits select the frequency of the $\Delta\Sigma$ A/D converter clock ( $A\phi$ ). The values shown below for each setting are frequency multipliers for the input clock. Set these bits so that $A\phi$ is approximately 25 MHz. See section 23, Clock Pulse Generator, for details. 000: $\times 1/6$ 001: $\times 1/5$ 010: $\times 1/4$ 011: $\times 1/3$ 1xx: Setting prohibited
0	ACK0	0	R/W	

[Legend] x: Don't care.

### 19.3.2 $\Delta\Sigma$ A/D Data Registers 0 to 5 (DSADDR0 to DSADDR5)

DSADDR0 to DSADDR5 are 16-bit read-only registers for storing the results of A/D conversion.

One register is provided for each analog input channel, and when A/D conversion on a channel is completed, the result of conversion is stored in the corresponding register. Data stored in each register are retained until the next round of A/D conversion on that channel ends and the new result is stored.

DSADDR registers can be read by the CPU at any time, but cannot be written to.

The A/D-converted data is stored in bit 15 to bit 0 as a signed binary number (two's complement). Bit 15 holds the MSB and bit 0 the LSB.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Name																
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

### 19.3.3 $\Delta\Sigma$ A/D Control/Status Register (DSADCSR)

DSADCSR controls A/D conversion and interrupts and selects analog input channels.

When writing to the register to change the settings of bits SCANE and CH5 to CH0, bit ADST must be clear.

Bit	15	14	13	12	11	10	9	8
Bit Name	ADF	ADIE	ADST	—	SCANE	—	TRGS1	TRGS0
Initial Value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/W	R/W	R	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	CH5	CH4	CH3	CH2	CH1	CH0
Initial Value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

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



Bit	Bit Name	Initial Value	R/W	Description
15	ADF	0	R/(W)* <sup>1</sup>	<p>A/D Conversion End Flag</p> <p>Indicates whether A/D conversion has ended.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>A/D conversion on all of the selected channels has ended.</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>Writing 0 to ADF after reading it as 1</li> <li>Activation of the DMAC by the DSADI interrupt and transfer of data in DSADDRn.</li> </ul>
14	ADIE	0	R/W	<p>A/D Conversion Interrupt Enable</p> <p>Setting this bit to 1 enables generation of DSADI interrupt requests in accord with the ADF bit.</p>
13	ADST	0	R/W	<p>A/D Conversion Start</p> <p>Controls starting and stopping of A/D conversion.</p> <p>Clearing this bit to 0 stops A/D conversion, placing the converter in the wait state. Setting this bit to 1 starts A/D conversion.</p> <p>In single mode, ADST is automatically cleared at the end of A/D conversion on the selected channels. In scan mode, ADST must be cleared by software because it is not cleared automatically.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>Writing 1 to ADST by software</li> <li>Input of an A/D conversion trigger signal while starting of A/D conversion by a trigger is enabled (TRGS1, TRGS0 <math>\neq</math> B'00)</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>Writing 0 to ADST by software</li> <li>End of A/D conversion on all selected channels while SCANE = 0</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
12	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.
11	SCANE	0	R/W	Scan Mode Enable Selects the mode of A/D conversion. 0: Single mode 1: Scan mode
10	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.
9	TRGS1	0	R/W	Timer Trigger Select 1, 0
8	TRGS0	0	R/W	These bits enable starting of A/D conversion by a trigger signal. 00: Disables starting by trigger signals. 01: Enables starting by a trigger from the TPU. 10: Enables starting by a trigger from the TMR. 11: Enables starting by the $\overline{\text{ANDSTRG}}$ pin input. *2
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
5	CH5	0	R/W	A/D Conversion Channel Select
4	CH4	0	R/W	These bits select the analog input channels for A/D conversion. They are independent of each other and can be set as desired.
3	CH3	0	R/W	
2	CH2	0	R/W	0: Channel n is not selected.
1	CH1	0	R/W	1: Channel n is selected.
0	CH0	0	R/W	(n = 0 to 5)

- Notes:
1. Only 0 can be written here, to clear the flag.
  2. When selecting starting of A/D conversion by the  $\overline{\text{ANDSTRG}}$  signal, clear the DDR bit for the corresponding pin to 0 and set the ICR bit to 1. See section 11, I/O Ports, for details.

### 19.3.4 $\Delta\Sigma$ A/D Control Register (DSADCR)

DSADCR specifies the A/D conversion time and controls stopping of the  $\Delta\Sigma$  modulator.

When changing the setting of DSADCR, the ADST bit must be clear.

Bit	15	14	13	12	11	10	9	8
Bit Name	CKS	—	GAIN1	GAIN0	—	—	—	—
Initial Value:	0	0	1	1	0	0	0	0
R/W:	R/W	R	R/W	R/W	R	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DSE	—	—	—	—	—	—	—
Initial Value:	0	0	0	0	1	0	0	0
R/W:	R/W	R	R	R	R/W	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15	CKS	0	R/W	Clock Select Sets the A/D conversion time. 0: 286-state conversion 1: Setting prohibited (One "state" = $A\phi/8$ )
14	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.
13	GAIN1	1	R/W	Gain Select
12	GAIN0	1	R/W	These bits set the gain for amplifying the analog input signals. 00: $\times 1$ 01: $\times 2$ 10: $\times 4$ 11: $\times 8$
11	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
10 to 8	—	All 0	R/W	Reserved The write value should always be 0.
7	DSE	0	R/W	$\Delta\Sigma$ Modulator Control Controls whether the $\Delta\Sigma$ modulator is stopped or runs. 0: $\Delta\Sigma$ modulator is stopped ( $A\phi/8$ clock is stopped). 1: $\Delta\Sigma$ modulator runs ( $A\phi/8$ clock runs).
6 to 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
3	—	1	R/W	Reserved The write value should always be 1.
2 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.

### 19.3.5 $\Delta\Sigma$ A/D Offset Cancel DAC Inputs 0 to 3 (DSADOF0 to DSADOF3)

DSADOF0 to DSADOF3 specify the values to be input to the DAC for canceling the offsets of analog input channels 0 to 3. Offset cancellation here means cancellation of the DC components of signals input to analog input channels 0 to 3, not cancellation of the offset of the internal amplifier.

Settings of the DSADOF registers can only be changed while the ADST bit is clear.

The six higher-order bits are reserved (fixed at 0) and cannot be written to. The settable values for the analog level for offset cancellation differ depending on the gain setting. Table 19.2 shows the settable values of DSADOFn for each gain setting.

The analog level for offset cancellation that corresponds to the register setting is calculated by using formula (1). Table 19.3 shows examples of register values and calculated analog levels for offset cancellation.

$$\text{DOF} = \text{DSADOF}/2^{10} \times (\text{AVrefT} - \text{AVrefB}) \quad \dots \text{Formula (1)}$$

- DOF: Analog level for offset cancellation (V)  
 DSADOF: Register value set in DSADOFn[9:0] for the corresponding channel  
 AVrefT:  $\Delta\Sigma$  reference voltage (high) (V), AVrefT = AVccA  
 AVrefB:  $\Delta\Sigma$  reference voltage (low), AVrefB = AvssA

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—										
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/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Table 19.2 Setting Values of Gain and DSADOFn**

GAIN1, GAIN0	DSADOFn (n = 0 to 3)	
	Settable Range	Remarks
B'00	H'0200	Always set to H'0200.
B'01	H'0200	Always set to H'0200.
B'10	H'0000 to H'03FE	Bit 0 must be clear (= 0)
B'11	H'0000 to H'03FF	—

**Table 19.3 Analog Levels for Offset Cancellation and Register Settings (Calculated Examples)**

<b>DSADOF[9:0]</b>	<b>Analog Level for Offset Cancellation</b>	<b>Value Calculated for AVrefT – AVrefB = 3.0 V AVrefB = 0 V</b>
H'000	$0/1024 \times (AV_{refT} - AV_{refB})$	0.0000
H'001	$1/1024 \times (AV_{refT} - AV_{refB})$	0.0029
H'002	$2/1024 \times (AV_{refT} - AV_{refB})$	0.0059
H'100	$256/1024 \times (AV_{refT} - AV_{refB})$	0.7500
H'200	$512/1024 \times (AV_{refT} - AV_{refB})$	1.5000
H'3FF	$1023/1024 \times (AV_{refT} - AV_{refB})$	2.9971

## 19.4 Operation

The  $\Delta\Sigma$  A/D converter uses a  $\Delta\Sigma$  modulator to convert analog input voltages within the range specified by the voltages on the AVrefT and AVrefB pins to digital values with 16-bit resolution. The  $\Delta\Sigma$  A/D converter is made up of three parts: an analog block built around a  $\Delta\Sigma$  modulator, a digital filter, and a control circuit.

In the analog block, the  $\Delta\Sigma$  modulator amplifies the input signals (eight-fold when the GAIN1 and GAIN0 bits in DSADCR is set to B'11) and converts them. During this process, the DC offsets of the signals input from the single-ended input signal pins (ANDS0, ANDS1, ANDS2, ANDS3) are cancelled if offset values have been set in the DSADOF0 to DSADOF3 registers. Differential input voltages on the differential input pins (ANDS4P, ANDS4N and ANDS5P, ANDS5N) can also be converted.

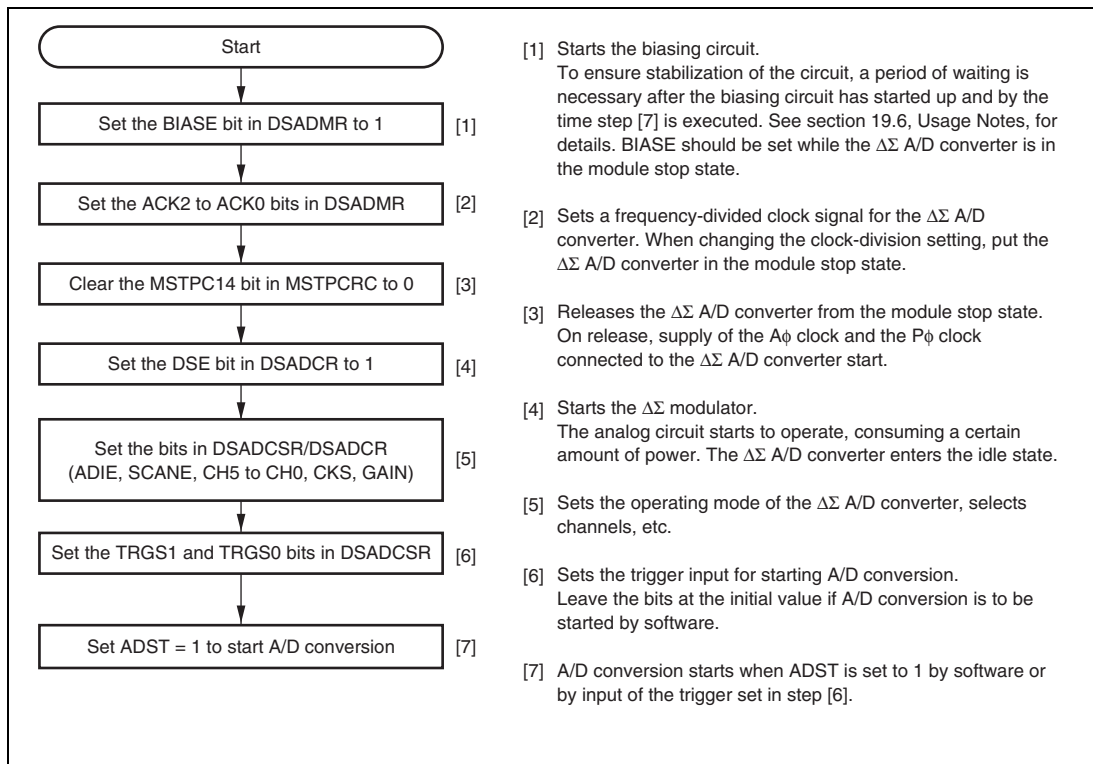
The voltage of a selected analog input signal is sampled at the  $A\phi/8$  clock frequency (oversampling frequency) and converted to a series of digital values by the second-order  $\Delta\Sigma$  modulator. The result of conversion is passed through a decimation filter (digital filter) and stored in the corresponding  $\Delta\Sigma$  A/D data register as a 16-bit signed binary number (two's complement).

The  $\Delta\Sigma$  A/D converter operates in either single mode or scan mode. Multiple channels are specified by selecting multiple A/D conversion channel-selection bits.

### 19.4.1 Procedure for Activating the $\Delta\Sigma$ A/D Converter

When the  $\Delta\Sigma$  A/D converter is to be used, register settings should be made in accord with the procedure for activation given below.

Figure 19.2 shows the procedure for activating the  $\Delta\Sigma$  A/D converter.



**Figure 19.2 Procedure for Activating the  $\Delta\Sigma$  A/D Converter**

### 19.4.2 Selecting Analog Input Channels

The  $\Delta\Sigma$  A/D converter has six analog input channels. Single-ended input signal pins ANDS0, ANDS1, ANDS2, and ANDS3 are used for channels 0 to 3. Channels 4 and 5 are capable of converting differential input signals. Pin pairs ANDS4P and ANDS4N, and ANDS5P and ANDS5N, are used for channels 4 and 5, respectively.

Channels for A/D conversion are selected by setting the corresponding CHn bit in DSADCSR to 1. A/D conversion is not performed on channels for which the CHn bit is clear. If all of the CHn bits have been cleared to 0, no A/D conversion will be performed.

Setting two or more CHn bits to 1 places the  $\Delta\Sigma$  A/D converter in multi-channel mode, where A/D conversion of the signals on the selected channels proceeds in sequence (from channel 0 to channel 5).

Values for canceling offsets of the single-ended input signals on channels 0 to 3 can be input as register settings. These values are set in  $\Delta\Sigma$  A/D offset cancel DAC inputs 0 to 3 (DSADOF0 to DSADOF3). During A/D conversion on channel n, the value set in the DSADOFn register is looked up and input to a 10-bit D/A converter that converts it to an analog signal, which provides the level for canceling the offset on the analog input channel.

Table 19.4 shows the correspondence of the analog input channel settings.

**Table 19.4 Correspondence between Settings and Analog Input Channels**

No.	Analog Input Channel	A/D Conversion Channel Select Bit	Analog Input Pin	Single-Ended/Differential Input	Offset Cancellation	Order of Execution
0	Channel 0	CH0	ANDS0	Single-ended	DSADOF0 register	↓
1	Channel 1	CH1	ANDS1	Single-ended	DSADOF1 register	
2	Channel 2	CH2	ANDS2	Single-ended	DSADOF2 register	
3	Channel 3	CH3	ANDS3	Single-ended	DSADOF3 register	
4	Channel 4	CH4	ANDS4P	Differential	ANDS4N pin	
5	Channel 5	CH5	ANDS5P	Differential	ANDS5N pin	



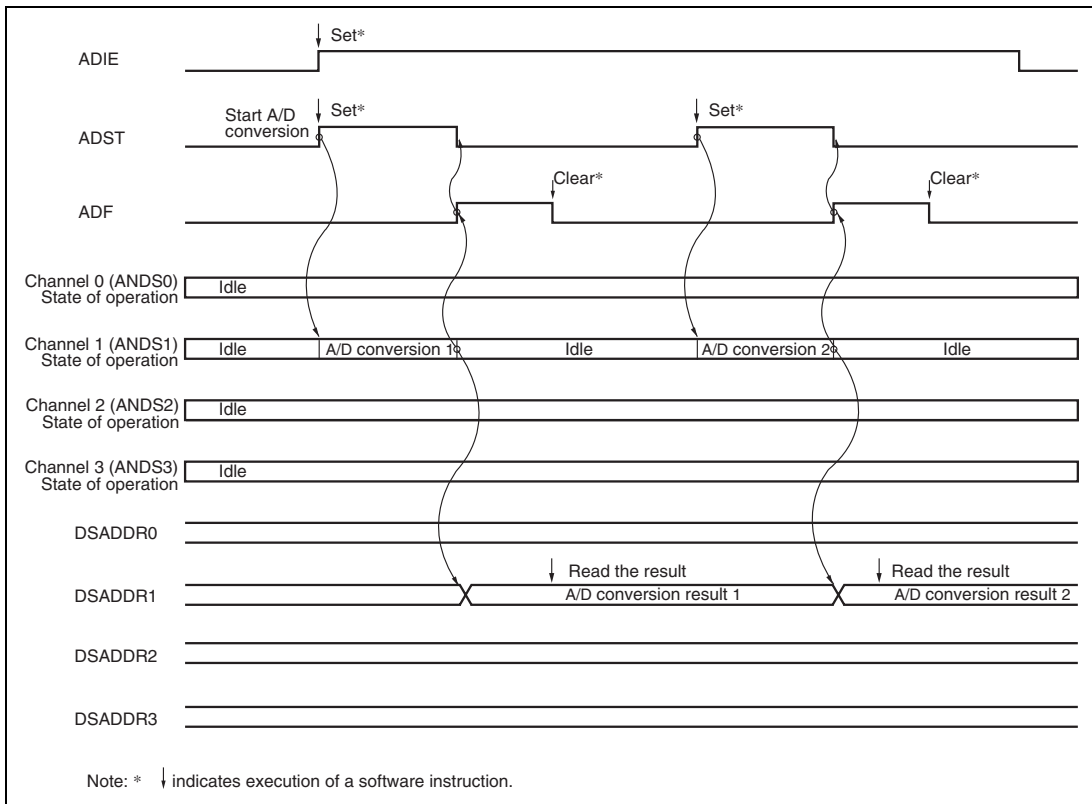
### 19.4.3 Single Mode

In single mode, either normal single mode, in which A/D conversion is executed once for a specified one analog input channel, or multi-channel mode, in which A/D conversion is executed once for each of the multiple channels in sequence, can be selected. Specifying two or more channels for A/D conversion by bits CH0 to CH5 in DSADCSR selects multi-channel mode operation.

Figure 19.3 shows an example of  $\Delta\Sigma$  A/D converter operation (in single-channel single mode with channel 1 selected).

When only one channel is selected (normal single mode), A/D conversion is performed once in the following way.

1. A/D conversion is started for the selected channel when the ADST bit in DSADCSR is set to 1 by software or by the input of trigger signal selected by the TRGS1 and TRGS0 bits in DSADCSR.
2. When A/D conversion is completed, the result is transferred to the  $\Delta\Sigma$  A/D data register for the selected channel (DSADDRn, n = 0 to 5).
3. When the result of A/D conversion is transferred to the data register and conversion by the  $\Delta\Sigma$  A/D converter is complete, the ADF bit in DSADCSR is set to 1. If the ADIE bit in DSADCSR is set to 1 at this time, a DSADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion and is automatically cleared on completion of A/D conversion. When the ADST bit is again set to 1, A/D conversion for the selected channel is started again.
5. If the ADST bit is cleared to 0 during A/D conversion, the conversion is stopped and the  $\Delta\Sigma$  A/D converter enters the idle state.



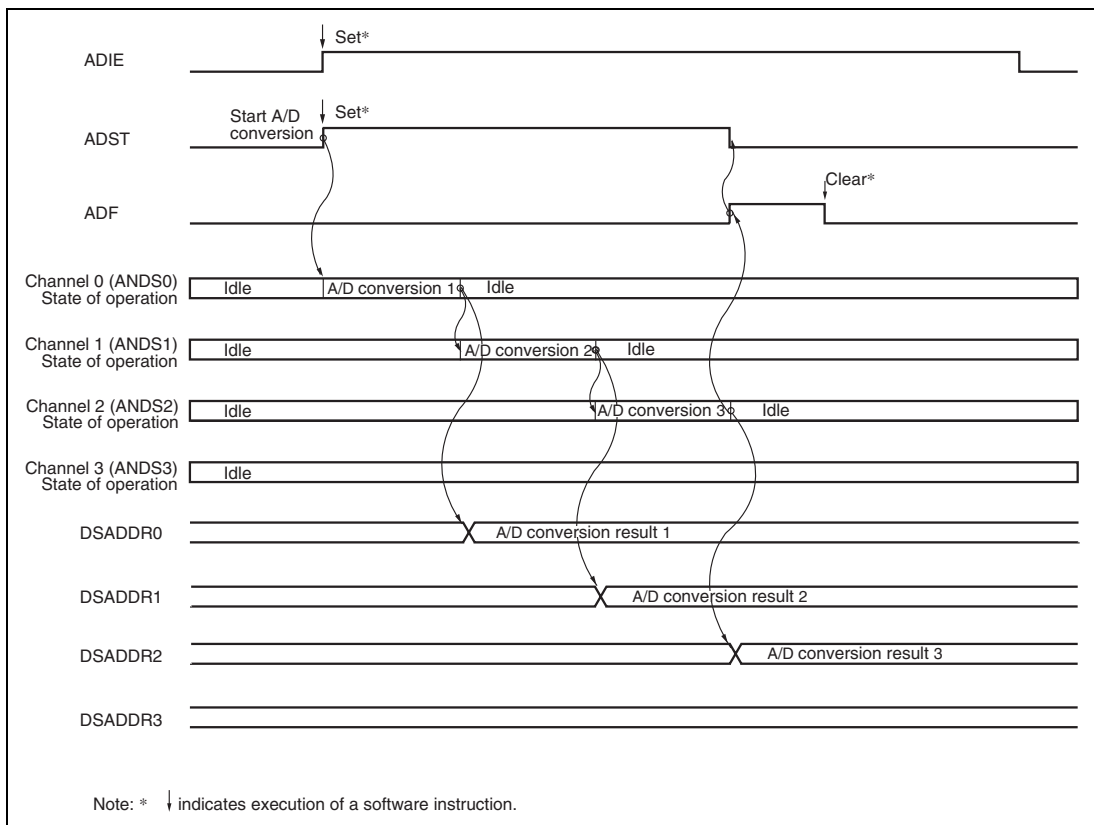
**Figure 19.3 Example of  $\Delta\Sigma$  A/D Converter Operation (Single Mode for One Channel: Channel 1)**

Figure 19.4 shows an example of  $\Delta\Sigma$  A/D converter operation (in multi-channel single mode with channels 0 to 2 selected).

When A/D conversion is performed for two or more channels (multi-channel single mode), the analog input on each of the selected channels is A/D converted once in sequence from channel 0, as described below.

1. A/D conversion is started for the selected channels when the ADST bit in DSADCSR is set to 1 by software or by the input of a trigger signal selected by the TRGS1 and TRGS0 bits in DSADCSR. Execution of A/D conversion is in order of rising channel number, so the order of precedence starts from channel 0.
2. When A/D conversion is completed for channel  $n$ , the result is transferred to the corresponding  $\Delta\Sigma$  A/D data register (DSADDR $n$ ,  $n = 0$  to 5).

3. After A/D conversion for channel n, A/D conversion for the next channel is started. Steps 2 and 3 are repeated until A/D conversion for all of the selected channels has been completed.
4. When A/D conversion for all of the selected channels has been completed, the ADF bit in DSADCSR is set to 1. If the setting of the ADIE bit in DSADCSR is 1 at this time, a DSADI interrupt request is also generated.
5. The ADST bit remains set to 1 during A/D conversion and is automatically cleared on completion of A/D conversion. When the ADST bit is subsequently set to 1, A/D conversion for the selected channels again proceeds from channel 0.
6. If the ADST bit is cleared to 0 during A/D conversion, the conversion is stopped and the  $\Delta\Sigma$  A/D converter enters the idle state.



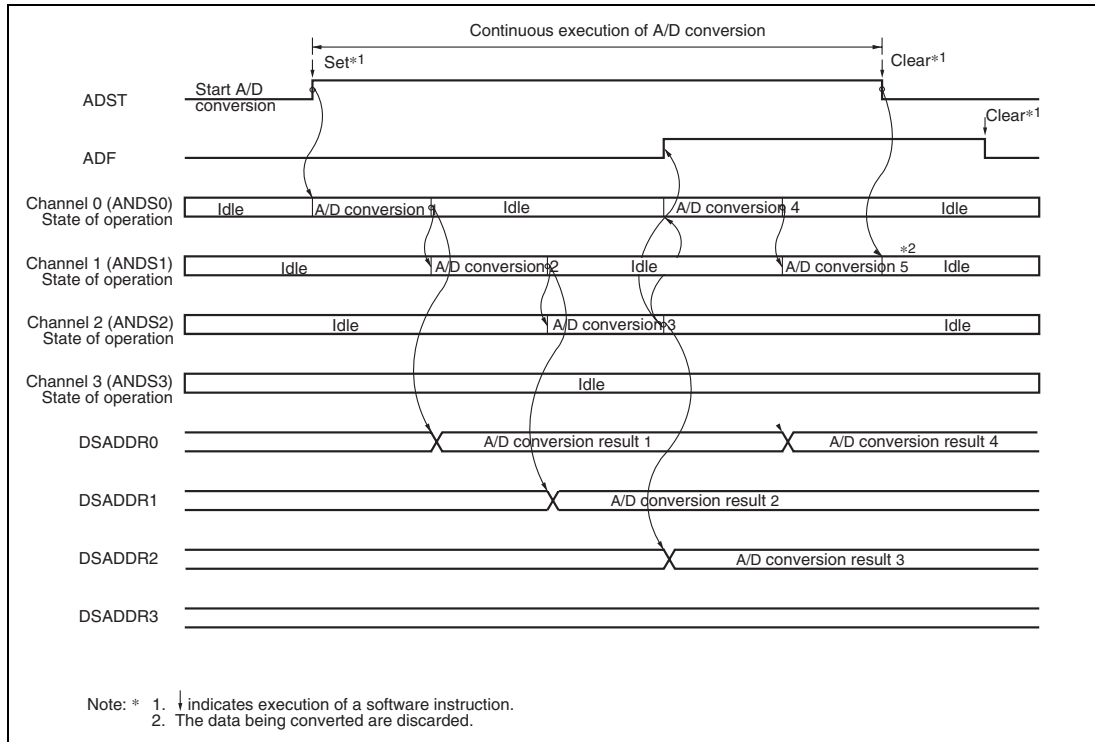
**Figure 19.4 Example of  $\Delta\Sigma$  A/D Converter Operation (Single Mode for Multiple Channels: Channels 0 to 2)**

#### 19.4.4 Scan Mode

In scan mode, A/D conversion is executed continuously for the specified analog input channels as follows. A/D conversion for up to six analog input channels can be specified by setting the CH0 to CH5 bits in DSADCSR to 1, indicating the required channels.

1. A/D conversion for the selected channels is started by a software instruction setting the ADST bit in DSADCSR to 1 or input of the trigger signal selected by the TRGS1 and TRGS0 bits in DSADCSR. When multiple channels have been selected, execution of A/D conversion is in order of rising channel number, so the order of precedence starts from channel 0.
2. When A/D conversion is completed for channel n, the result is transferred to the corresponding  $\Delta\Sigma$  A/D data register (DSADDRn, n = 0 to 5).
3. When A/D conversion for all of the selected channels has been completed, the ADF bit in DSADCSR is set to 1. If the setting of the ADIE bit in DSADCSR is 1 at this time, a DSADI interrupt request is also generated.
4. The  $\Delta\Sigma$  A/D converter starts another round of A/D conversion in order of precedence from channel 0. The ADST bit is not cleared automatically, and steps 2 to 4 are repeated as long as ADST = 1.
5. If the ADST bit is cleared to 0 during A/D conversion, the conversion is stopped and the  $\Delta\Sigma$  A/D converter enters the idle state. When the ADST bit is subsequently set to 1, A/D conversion for the selected channels again proceeds from channel 0.

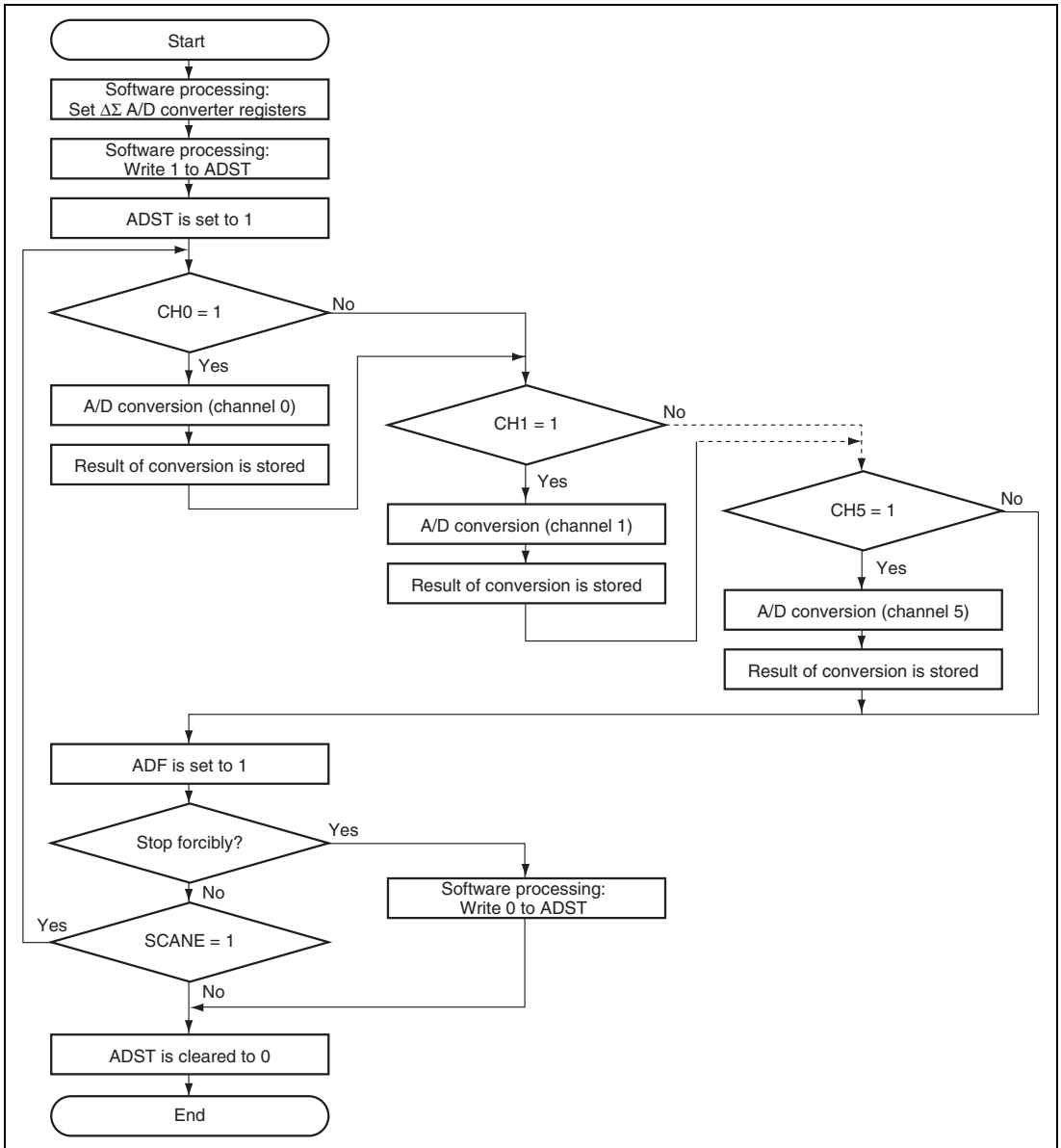
Figure 19.5 shows an example of  $\Delta\Sigma$  A/D converter operation in scan mode with channels 0 to 2 selected.



**Figure 19.5 Example of  $\Delta\Sigma$  A/D Converter Operation (Scan Mode with Channels 0 to 2 Selected)**

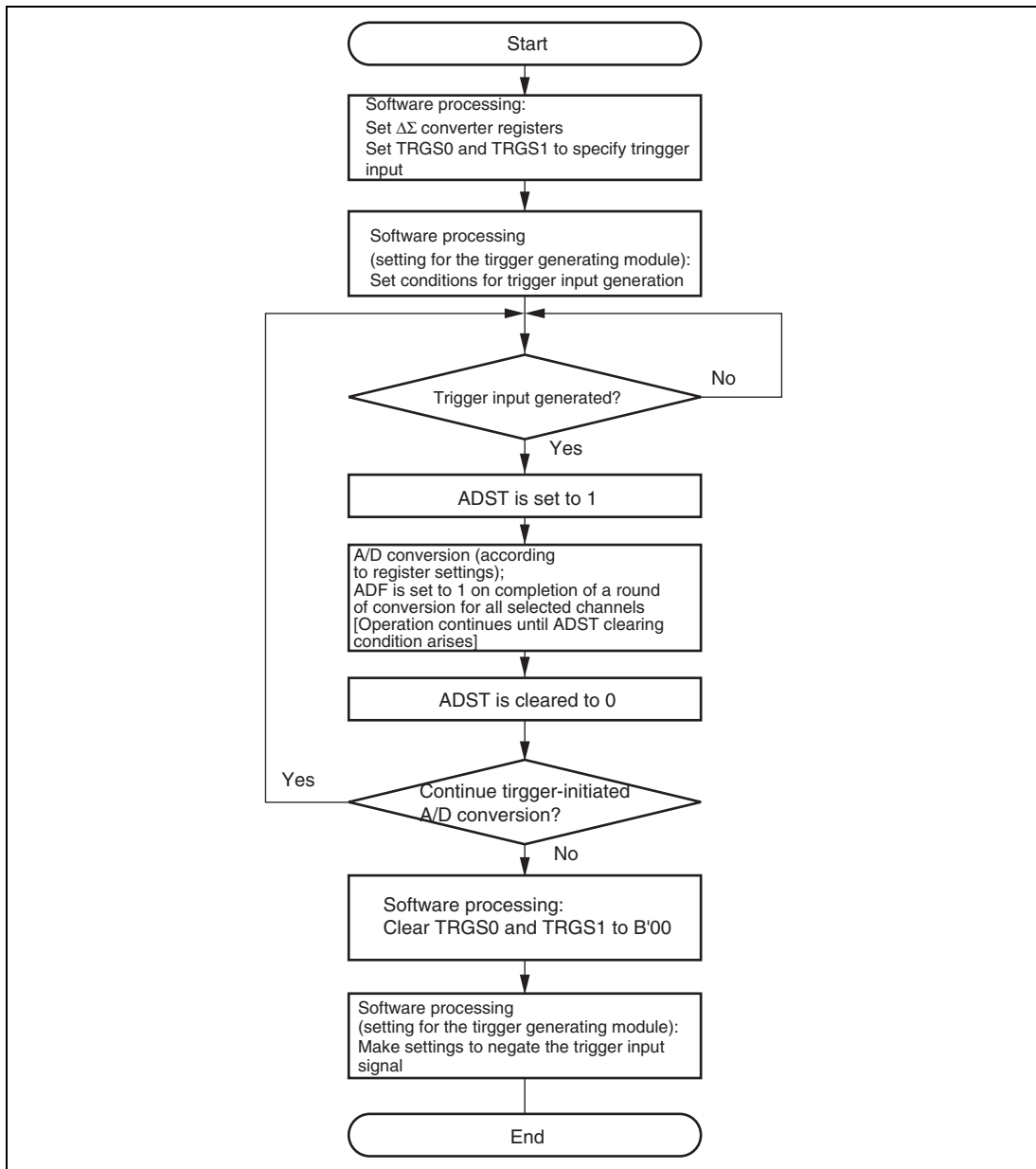
### 19.4.5 Flow of $\Delta\Sigma$ A/D Conversion Operation

Figure 19.6 shows the flow of  $\Delta\Sigma$  A/D conversion initiated by software.



**Figure 19.6** Flow of  $\Delta\Sigma$  A/D Conversion Operation (Initiated by Software)

Figure 19.7 shows the flow of  $\Delta\Sigma$  A/D conversion initiated by a trigger input.

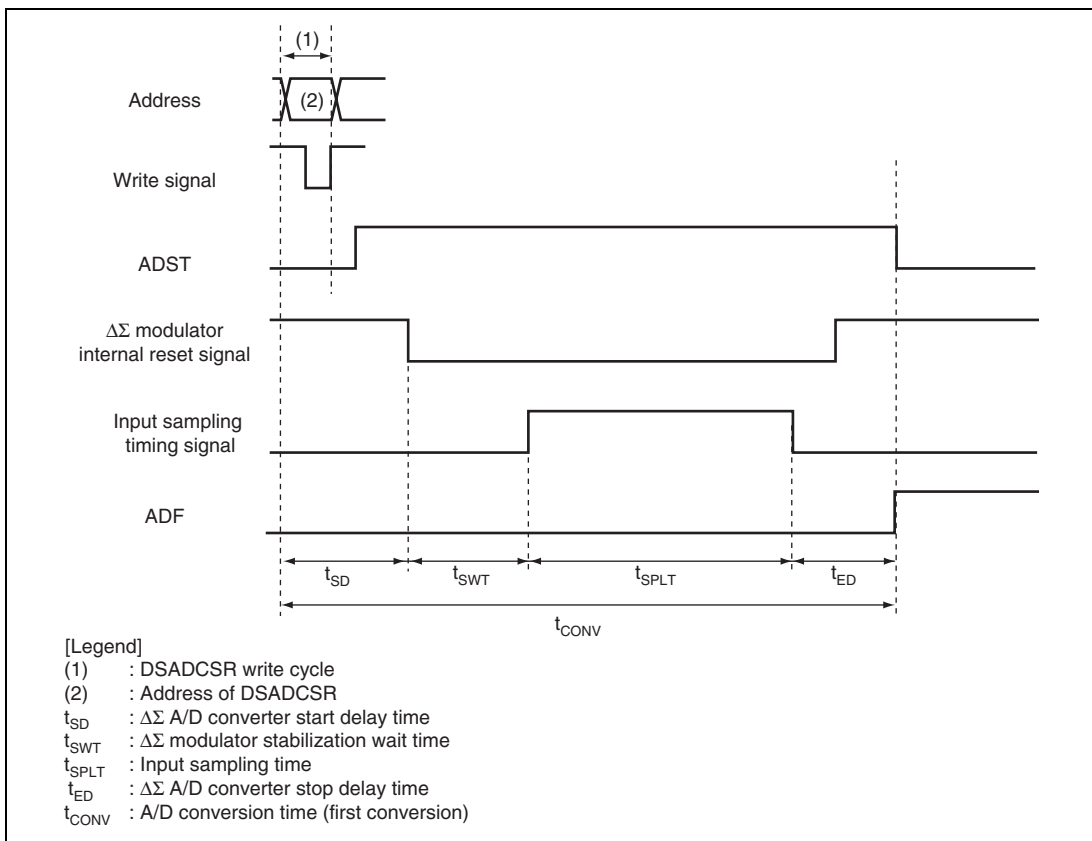


**Figure 19.7 Flow of  $\Delta\Sigma$  A/D Conversion Operation (Initiated by a Trigger Input)**

### 19.4.6 Analog Input Sampling and A/D Conversion Time

After the ADST bit in DSADCSR has been set to 1 to initiate conversion, the  $\Delta\Sigma$  A/D converter only starts sampling the analog inputs after the  $\Delta\Sigma$  A/D converter start-up delay time ( $t_{SD}$ ) and time to wait for the  $\Delta\Sigma$  modulator to be stabilized ( $t_{SWT}$ ) have elapsed. The  $\Delta\Sigma$  modulator samples the analog input and converts it to a sequence of digital values, which is then passed through a digital filter. The A/D conversion ends after the input sampling time ( $t_{SPLT}$ ) and the subsequent  $\Delta\Sigma$  modulator stop delay time ( $t_{ED}$ ) have elapsed.

Figure 19.8 shows the timing of A/D conversion, and tables 19.5 and 19.6 show A/D conversion times. As shown in figure 19.8, the A/D conversion time is a total of four periods.  $t_{SD}$  and  $t_{ED}$  can vary because they are determined by the timing of synchronization between different clock signals and the state of control of synchronization processing at the end of the previous round of A/D conversion. For this reason, conversion times vary within the range shown in table 19.5.

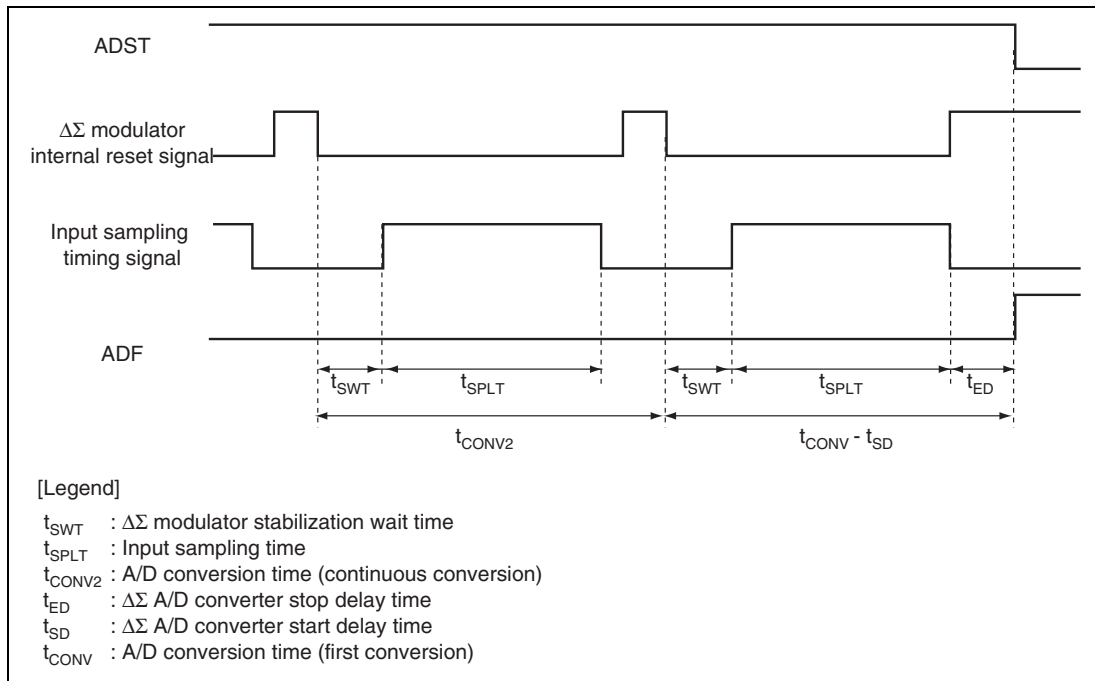


**Figure 19.8 A/D Conversion Timing (Single Mode, Once, One Channel)**



In multi-channel mode and scan mode, conversion time for the first round of conversion is as shown in table 19.5 and times for the second and subsequent rounds are as shown in table 19.6.

Figure 19.9 shows the timing of the second and subsequent rounds of A/D conversion (for successive conversion).



**Figure 19.9 Timing of Second and Subsequent Rounds of A/D Conversion (Successive Conversion)**

**Table 19.5 A/D Conversion Time (First Round)**

Item	Symbol	CKS = 0		
		min	typ	max
$\Delta\Sigma$ A/D converter start delay time	$t_{SD}$	5	—	6
$\Delta\Sigma$ modulator stabilization time at start-up	$t_{SWT}$	—	29	—
Input sampling time	$t_{SPLT}$	—	254	—
$\Delta\Sigma$ A/D converter stop delay time	$t_{ED}$	2	—	3
A/D conversion time (first round)	$t_{CONV}$	290	—	292
A/D conversion time (second and subsequent rounds)	$t_{CONV2}$		286	

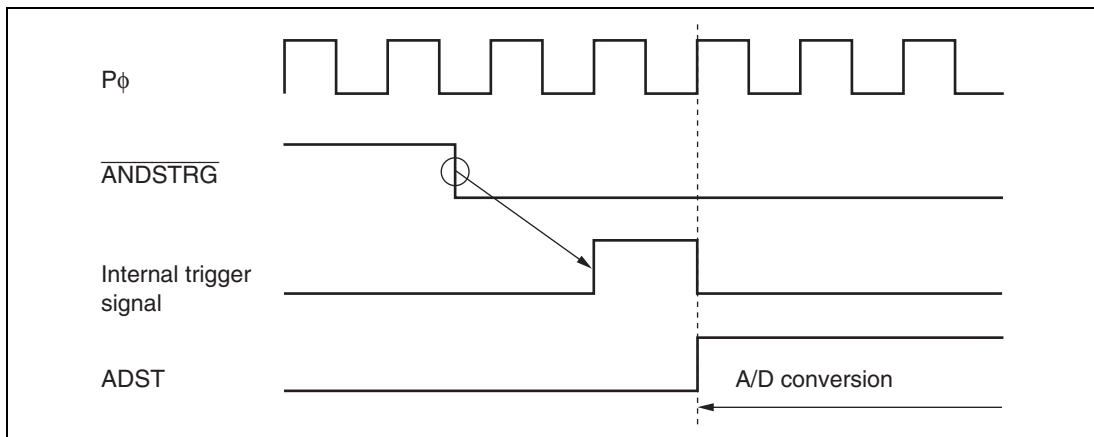
Note: The unit for values in the table is the period of  $A\phi/8$  ("state").

**Table 19.6 A/D Conversion Time (Second and Subsequent Rounds)**

CKS	Conversion Time ( $A\phi/8$ Periods)
0	286 (fixed)

### 19.4.7 External Trigger Input Timing

A/D conversion can also be started by an external trigger signal. Setting the TRGS1 and TRGS0 bits in DSADCSR to B'11 selects the signal on the  $\overline{\text{ANDSTRG}}$  pin as an external trigger. The ADST bit in DSADCSR is set to 1 on the falling edge of  $\overline{\text{ANDSTRG}}$ , initiating A/D conversion. Other operations are the same as those in the case where the ADST bit is set to 1 by software, regardless of whether the converter is in single mode or scan mode. The timing of this operation is shown in figure 19.10.



**Figure 19.10 Timing of External Trigger Input**

## 19.5 Interrupt Source

The  $\Delta\Sigma$  A/D converter can generate an A/D conversion end interrupt request (DSADI) at the end of A/D conversion. The ADF bit in DSADCSR is set to 1 on completion of A/D conversion, and if the setting of the ADIE bit is 1 at this time, a DSADI interrupt request is also generated.

The DSADI interrupt can be used to activate the DMA controller (DMAC). Using the DMAC to read the converted data in response to DSADI interrupts allows continuous conversion without software overhead.

When performing DMA transfer in response to DSADI interrupts, setting the DTA bit in the DMDR register of the DMA channel to 1 and placing the address of DSADDRn in DSAR enables clearing of the ADF bit when DMA transfer is executed.

**Table 19.7 Interrupt Source of the  $\Delta\Sigma$  A/D Converter**

Name	Interrupt Source	Interrupt Flag	Activation of DTC	Activation of DMAC
DSADI	End of A/D conversion	ADF	No	Yes

## 19.6 Usage Notes

### 19.6.1 Module Stop Function Setting

Operation of the  $\Delta\Sigma$  A/D converter can be enabled or disabled by setting the module stop control register. By default, the  $\Delta\Sigma$  A/D converter is stopped. Most registers of the  $\Delta\Sigma$  A/D converter only become accessible when it is released from the module stop state. See section 24, Power-Down Modes, for details.

Although DSADMR is accessible to the CPU at any time, writing to this register should only be performed while the converter is in the module stop state.

To stop the  $\Delta\Sigma$  A/D converter completely, place it in the module stop state and then stop the biasing circuit by clearing the BIASE bit in DSADMR to 0.

### 19.6.2 Settings for the Biasing Circuit

When the BIASE bit in DSADMR is set to enable the biasing circuit before the  $\Delta\Sigma$  A/D converter is used, a certain period must be secured for stabilization of the biasing circuit. If A/D conversion is executed without ensuring enough time for stabilization of the biasing circuit, the precision of A/D conversion is not guaranteed.

When the biasing circuit is stopped by clearing the BIASE bit in DSADMR to 0 or on entry to the hardware standby mode, the reset state, or deep software standby mode, a certain period for stabilization of the biasing circuit will be required after the BIASE bit has been set to 1 again.

A certain amount of biasing current flows while the biasing circuit is running. Since the value set in the BIASE bit is retained in software standby mode, the supply current will include the current that flows through the biasing circuit if BIASE = 1. Be sure to set the BIASE bit appropriately before initiating software standby mode.

Ensure at least 20 ms for stabilization of the biasing circuit.

### 19.6.3 State of the $\Delta\Sigma$ A/D Converter in Software Standby Mode

If the LSI enters software standby mode with A/D conversion enabled, the  $\Delta\Sigma$  A/D converter is initialized and placed in an idle state. The  $\Delta\Sigma$  A/D data registers (DSADDRn), which hold the results of conversion, are also initialized. The analog power supply current is the current that flows through the biasing circuit. If the analog power supply current in software standby mode must be reduced, clear the BIASE bit in DSDMR to 0 to stop the biasing circuit before initiating software standby mode.

### 19.6.4 Changing the Settings of $\Delta\Sigma$ A/D Converter Registers

To avoid malfunctions during A/D conversion, do not change the settings of the  $\Delta\Sigma$  A/D converter registers while the ADST bit in DSADCSR is set to 1. Always write to the registers with the ADST bit cleared to 0. The exceptions are clearing of the ADST bit and clearing of the ADF bit after reading a 1 from it.

When the TRGS1 and TRGS0 bits in DSADCSR are set to a value other than B'00, the ADST bit may be set automatically by the trigger signal. Accordingly, before setting registers of the  $\Delta\Sigma$  A/D converter, set the TRGS1 and TRGS0 bits to B'00 or take measures to ensure that no trigger signal will be input.

### 19.6.5 DSE Bit

Use the  $\Delta\Sigma$  A/D converter with the DSE bit in DSADCR set to 1.

## Section 20 D/A Converter

### 20.1 Features

- 8-bit resolution
- Two 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 state specifiable

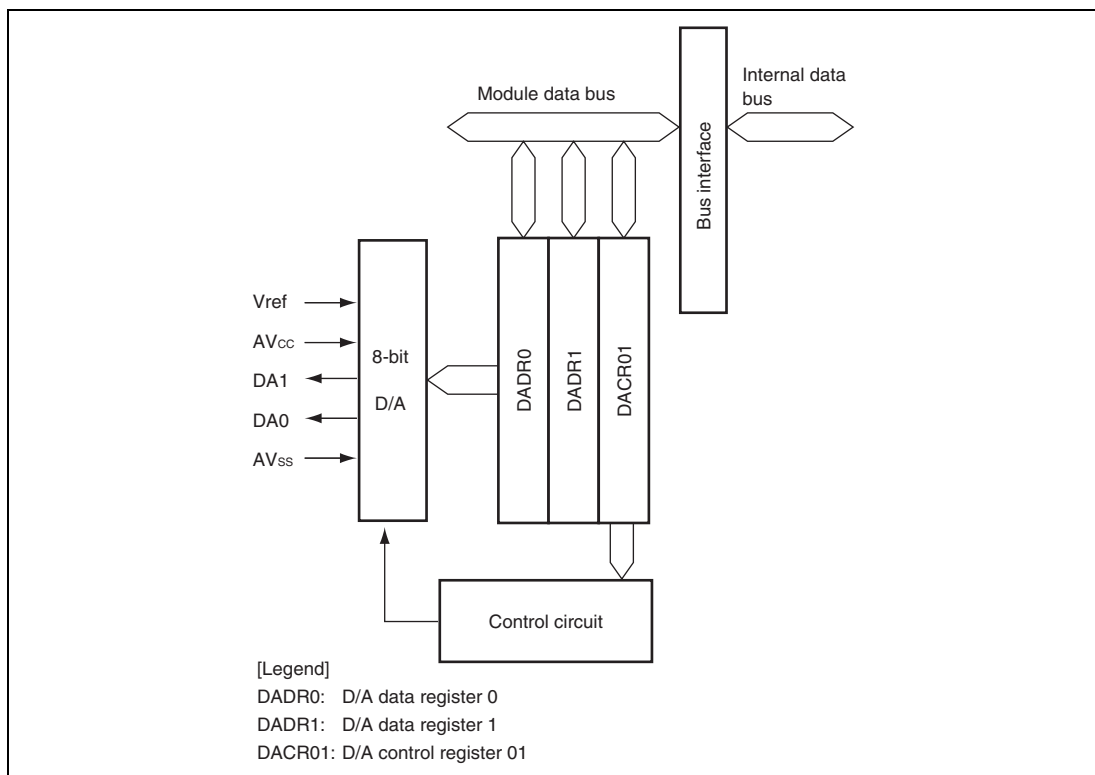


Figure 20.1 Block Diagram of D/A Converter

## 20.2 Input/Output Pins

Table 20.1 shows the pin configuration of the D/A converter.

**Table 20.1 Pin Configuration**

Pin Name	Symbol	I/O	Function
Analog power supply pin	AV <sub>CC</sub>	Input	Analog block power supply
Analog ground pin	AV <sub>SS</sub>	Input	Analog block ground
Reference voltage pin	V <sub>ref</sub>	Input	D/A conversion reference voltage
Analog output pin 0	DA0	Output	Channel 0 analog output
Analog output pin 1	DA1	Output	Channel 1 analog output

## 20.3 Register Descriptions

The D/A converter has the following registers.

- D/A data register 0 (DADR0)
- D/A data register 1 (DADR1)
- D/A control register 01 (DACR01)

### 20.3.1 D/A Data Registers 0 and 1 (DADR0 and DADR1)

DADR0 and DADR1 are 8-bit readable/writable registers that store data to which D/A conversion is to be performed. Whenever an analog output is enabled, the values in DADR are converted and output to the analog output pins.

Bit	7	6	5	4	3	2	1	0
Bit Name								
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



### 20.3.2 D/A Control Register 01 (DACR01)

DACR01 controls the operation of the D/A converter.

Bit	7	6	5	4	3	2	1	0
Bit Name	DAOE1	DAOE0	DAE	—	—	—	—	—
Initial Value	0	0	0	1	1	1	1	1
R/W	R/W	R/W	R/W	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	DAOE1	0	R/W	D/A Output Enable 1 Controls D/A conversion and analog output. 0: Analog output of channel 1 (DA1) is disabled 1: D/A conversion of channel 1 is enabled. Analog output of channel 1 (DA1) is enabled.
6	DAOE0	0	R/W	D/A Output Enable 0 Controls D/A conversion and analog output. 0: Analog output of channel 0 (DA0) is disabled 1: D/A conversion of channel 0 is enabled. Analog output of channel 0 (DA0) is enabled.
5	DAE	0	R/W	D/A Enable Used together with the DAOE0 and DAOE1 bits to control D/A conversion. When this bit is cleared to 0, D/A conversion is controlled independently for channels 0 and 1. When this bit is set to 1, D/A conversion for channels 0 and 1 is controlled together. Output of conversion results is always controlled by the DAOE0 and DAOE1 bits. For details, see table 20.2, Control of D/A Conversion.
4 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

**Table 20.2 Control of D/A Conversion**

Bit 5 DAE	Bit 7 DAOE1	Bit 6 DAOE0	Description
0	0	0	D/A conversion is disabled.
		1	D/A conversion of channel 0 is enabled and D/A conversion of channel 1 is disabled. Analog output of channel 0 (DA0) is enabled and analog output of channel 1 (DA1) is disabled.
	1	0	D/A conversion of channel 0 is disabled and D/A conversion of channel 1 is enabled. Analog output of channel 0 (DA0) is disabled and analog output of channel 1 (DA1) is enabled.
		1	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is enabled.
		0	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is disabled.
	1	0	1
0			D/A conversion of channels 0 and 1 is enabled. Analog output of channel 0 (DA0) is disabled and analog output of channel 1 (DA1) is enabled.
1		0	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is enabled.
		1	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is enabled.

## 20.4 Operation

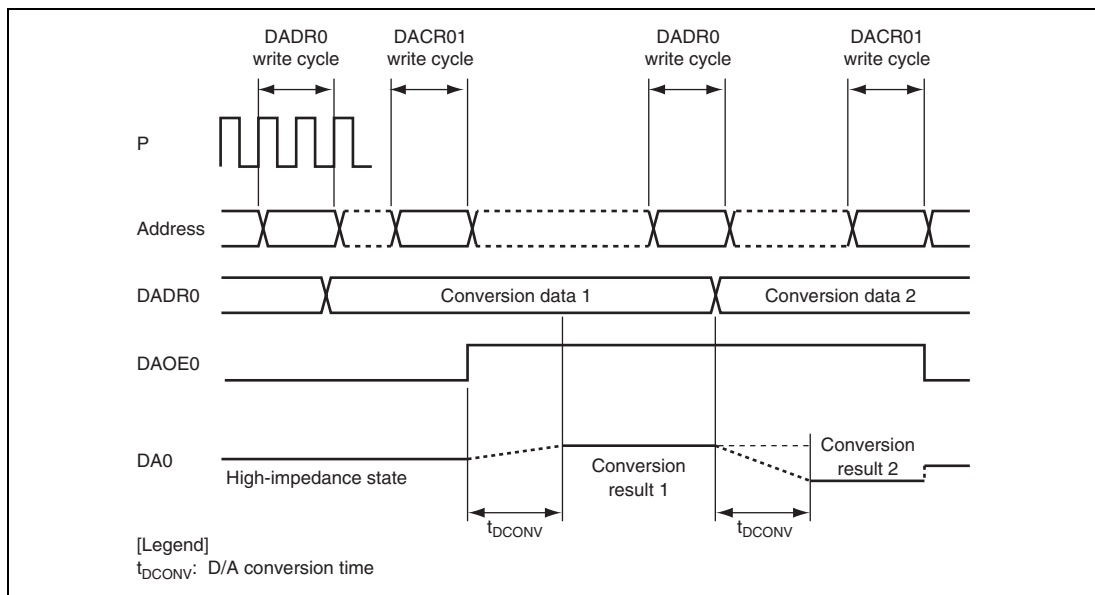
The D/A converter includes D/A conversion circuits for two channels, each of which can operate independently. When the DAOE bit in DACR01 is set to 1, D/A conversion is enabled and the conversion result is output.

An operation example of D/A conversion on channel 0 is shown below. Figure 20.2 shows the timing of this operation.

1. Write the conversion data to DADR0.
2. Set the DAOE0 bit in DACR01 to 1 to start D/A conversion. The conversion result is output from the analog output pin DA0 after the conversion time  $t_{\text{DCONV}}$  has elapsed. The conversion result continues to be output until DADR0 is written to again or the DAOE0 bit is cleared to 0. The output value is expressed by the following formula:

$$\text{Contents of DADR}/256 \times V_{\text{ref}}$$

3. If DADR0 is written to again, the conversion is immediately started. The conversion result is output after the conversion time  $t_{\text{DCONV}}$  has elapsed.
4. If the DAOE0 bit is cleared to 0, analog output is disabled.



**Figure 20.2 Example of D/A Converter Operation**

## 20.5 Usage Notes

### 20.5.1 Module Stop Function Setting

Operation of the D/A converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the D/A converter to be halted. Register access is enabled by clearing the module stop state. For details, refer to section 24, Power-Down Modes.

### 20.5.2 D/A Output Hold Function in Software Standby Mode

When this LSI enters software standby mode with D/A conversion enabled, the D/A outputs are retained, and the analog power supply current is equal to as during D/A conversion. If the analog power supply current needs to be reduced in software standby mode, clear the DAOE0, DAOE1, and DAE bits all to 0 to disable the D/A outputs.

## Section 21 RAM

This LSI has a high-speed static RAM. The RAM is connected to the CPU by a 32-bit data bus, enabling one-state access by the CPU to all byte data, word data, and longword data.

The RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on SYSCR, refer to section 3.2.2, System Control Register (SYSCR).

<b>Product Classification</b>		<b>RAM Size</b>	<b>RAM Addresses</b>
Flash memory version	H8SX/1622	24 Kbytes	H'FF6000 to H'FFBFFF



## Section 22 Flash Memory

The flash memory has the following features. Figure 22.1 is a block diagram of the flash memory.

### 22.1 Features

- ROM size

Product Classification		ROM Size	ROM Address
H8SX/1622	R5F61622	256 Kbytes	H'000000 to H'03FFFF (modes 1, 2, 6, and 7)

- Two memory MATs

The start addresses of two memory spaces (memory MATs) are allocated to the same address. The mode setting in the initiation determines which memory MAT is initiated first. The memory MATs can be switched by using the bank-switching method after initiation.

— User MAT initiated at a reset in user mode: 256 Kbytes

— User boot MAT is initiated at a reset in user boot mode: 16 Kbytes

- Programming/erasing interface by the download of on-chip program

This LSI has a programming/erasing program. After downloading this program to the on-chip RAM, programming/erasure can be performed by setting the parameters.

- Programming/erasing time

Programming time: 1 ms (typ.) for 128-byte simultaneous programming

Erasing time: 600 ms (typ.) per 1 block (64 Kbytes)

- Number of programming

The number of programming can be up to 100 times at the minimum. (1 to 100 times are guaranteed.)

- Three on-board programming modes

Boot mode: Using the on-chip SCI\_4, the user MAT and user boot MAT can be programmed/erased. In boot mode, the bit rate between the host and this LSI can be adjusted automatically.

User program mode: Using a desired interface, the user MAT can be programmed/erased.

User boot mode: Using a desired interface, the user boot program can be made and the user MAT can be programmed/erased.

- Off-board programming mode

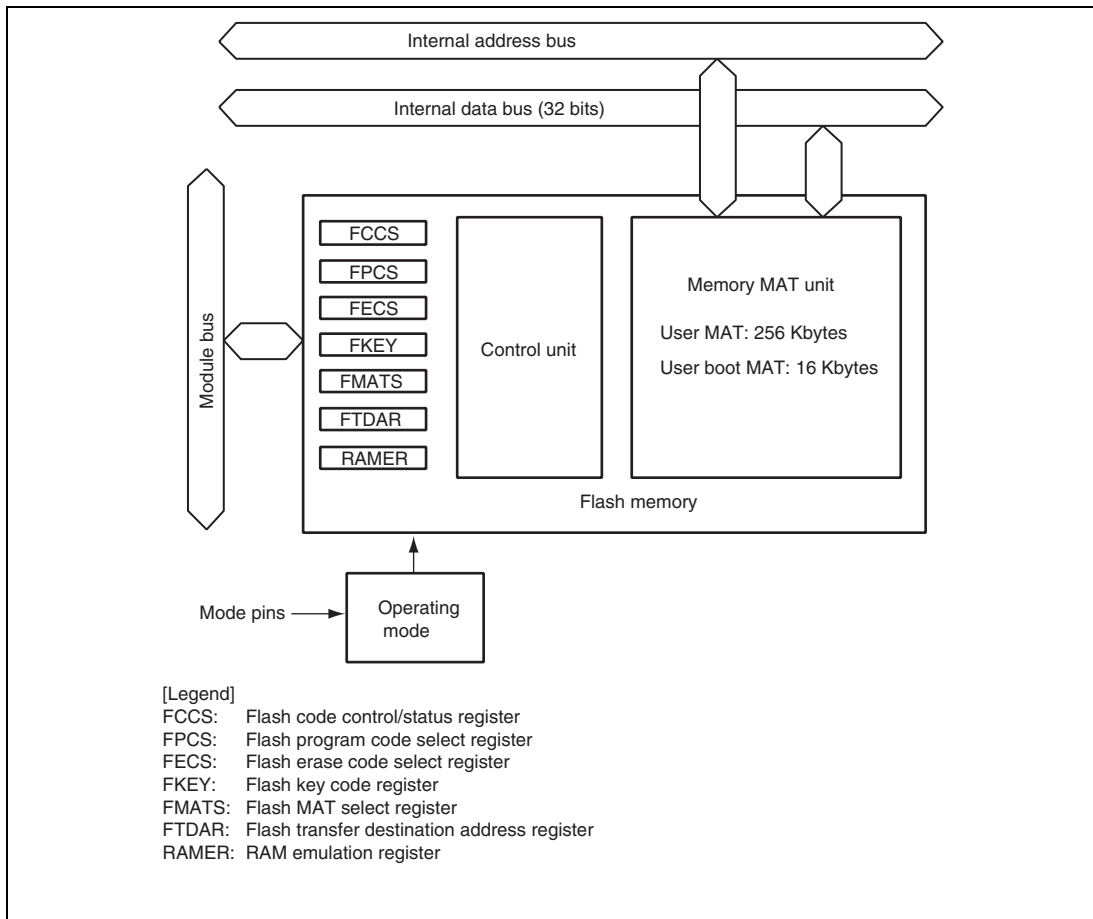
Programmer mode: Using a PROM programmer, the user MAT and user boot MAT can be programmed/erased.

- Programming/erasing protection

Protection against programming/erasure of the flash memory can be set by hardware protection, software protection, or error protection.

- Flash memory emulation function using the on-chip RAM

Realtime emulation of the flash memory programming can be performed by overlaying parts of the flash memory (user MAT) area and the on-chip RAM.

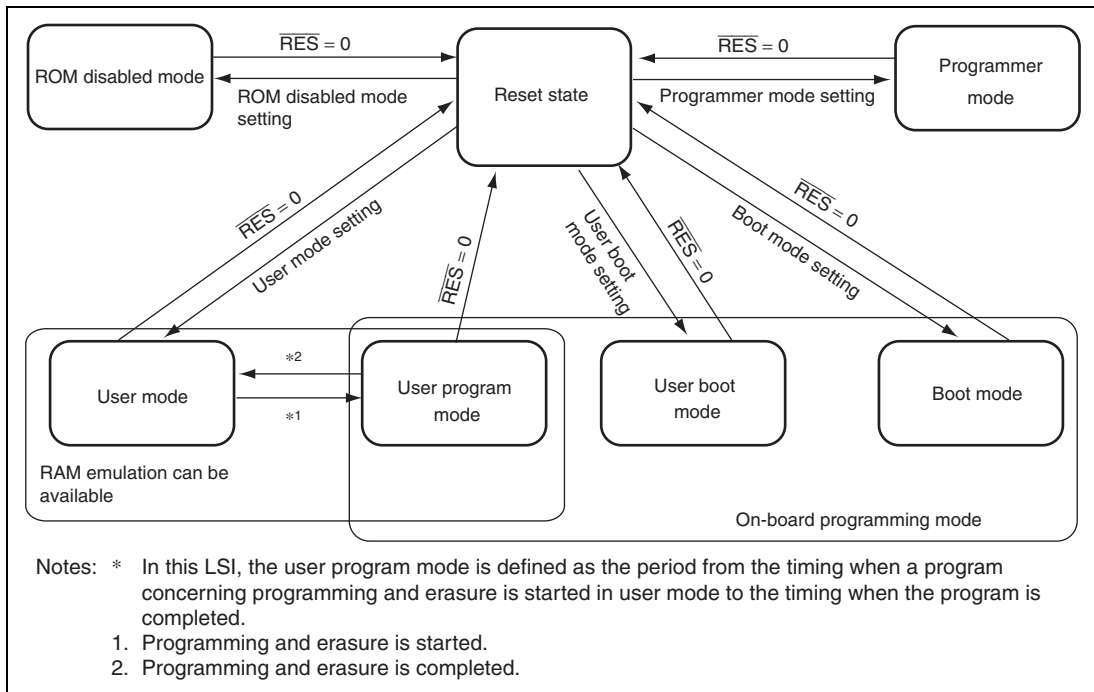


**Figure 22.1 Block Diagram of Flash Memory**



## 22.2 Mode Transition Diagram

When the mode pins are set in the reset state and reset start is performed, this LSI enters each operating mode as shown in figure 22.2. Although the flash memory can be read in user mode, it cannot be programmed or erased. The flash memory can be programmed or erased in boot mode, user program mode, user boot mode, and programmer mode. The differences between boot mode, user program mode, user boot mode, and programmer mode are shown in table 22.1.



**Figure 22.2 Mode Transition of Flash Memory**

**Table 22.1 Differences between Boot Mode, User Program Mode, User Boot Mode, and Programmer Mode**

Item	Boot Mode	User Program Mode	User Boot Mode	Programmer Mode
Programming/erasing environment	On-board programming	On-board programming	On-board programming	Off-board programming
Programming/erasing enable MAT	<ul style="list-style-type: none"> <li>• User MAT</li> <li>• User boot MAT</li> </ul>	<ul style="list-style-type: none"> <li>• User MAT</li> </ul>	<ul style="list-style-type: none"> <li>• User MAT</li> </ul>	<ul style="list-style-type: none"> <li>• User MAT</li> <li>• User boot MAT</li> </ul>
Programming/erasing control	Command	Programming/erasing interface	Programming/erasing interface	Command
All erasure	O (Automatic)	O	O	O (Automatic)
Block division erasure	O* <sup>1</sup>	O	O	×
Program data transfer	From host via SCI	From desired device via RAM	From desired device via RAM	Via programmer
RAM emulation	×	O	O	×
Reset initiation MAT	Embedded program storage area	User MAT	User boot MAT* <sup>2</sup>	—
Transition to user mode	Changing mode and reset	Completing Programming/erasure* <sup>3</sup>	Changing mode and reset	—

Notes: 1. All-erasure is performed. After that, the specified block can be erased.

2. First, the reset vector is fetched from the embedded program storage area. After the flash memory related registers are checked, the reset vector is fetched from the user boot MAT.

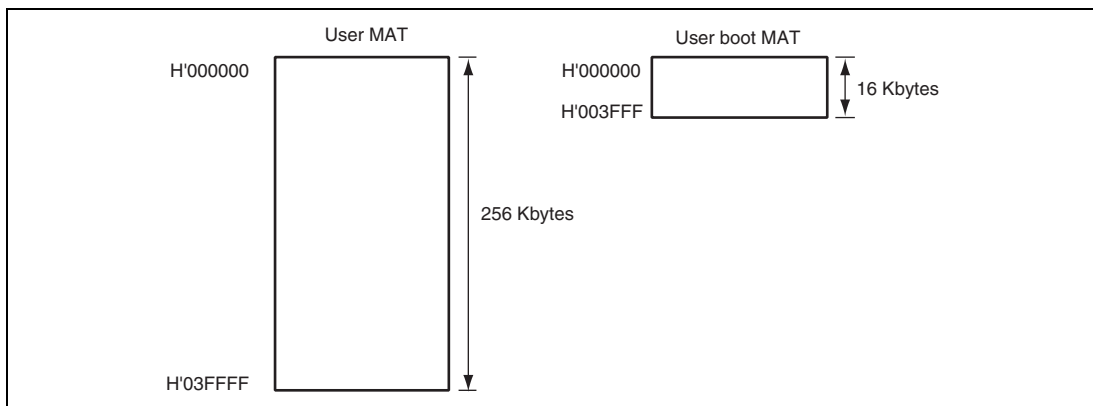
3. In this LSI, the user programming mode is defined as the period from the timing when a program concerning programming and erasure is started to the timing when the program is completed. For details on a program concerning programming and erasure, see section 22.8.2, User Program Mode.

## 22.3 Memory MAT Configuration

The memory MATs of flash memory in this LSI consists of the 256-Kbyte user MAT and 16-Kbyte user boot MAT. The start addresses of the user MAT and user boot MAT are allocated to the same address. Therefore, when the program execution or data access is performed between the two memory MATs, the memory MATs must be switched by the flash MAT select register (FMATS).

The user MAT or user boot MAT can be read in all modes. However, the user boot MAT can be programmed or erased only in boot mode and programmer mode.

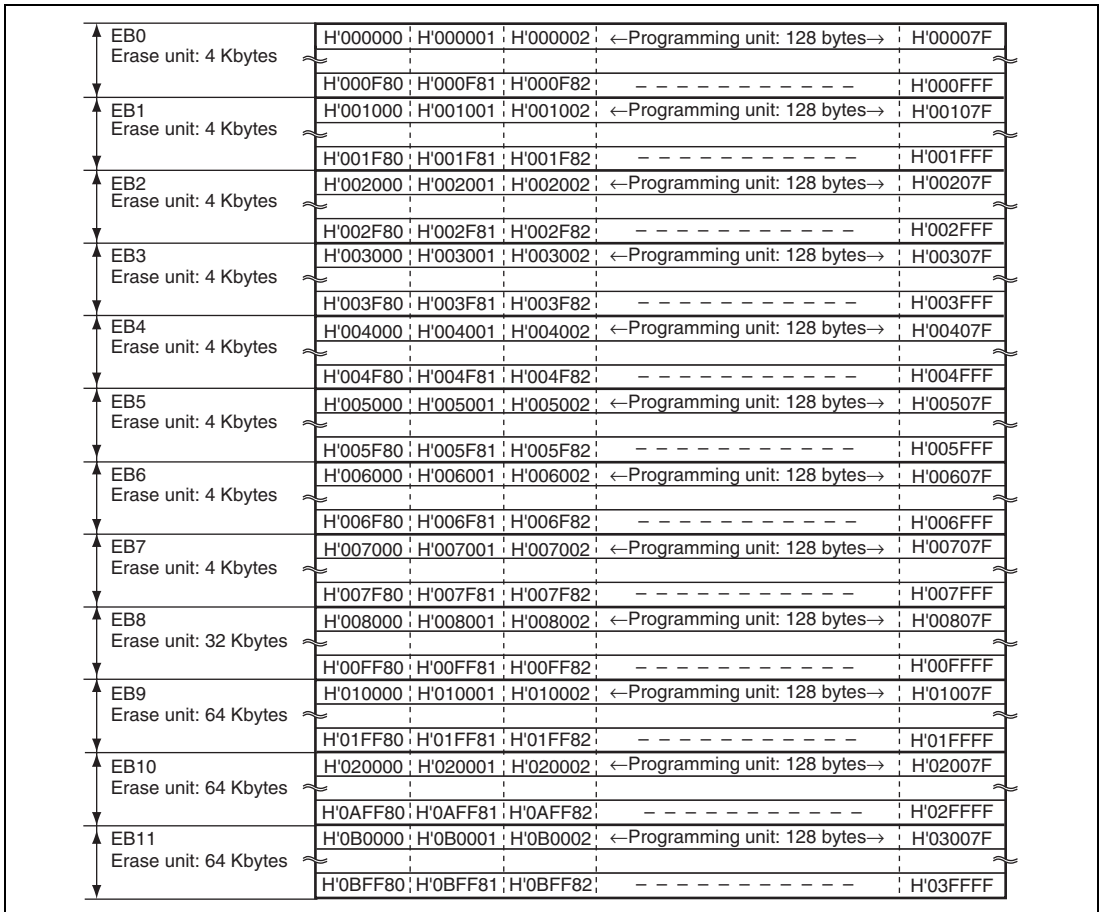
The size of the user MAT is different from that of the user boot MAT. Addresses which exceed the size of the 16-Kbyte user boot MAT should not be accessed. If an attempt is made, data is read as an undefined value.



**Figure 22.3 Memory MAT Configuration**

## 22.4 Block Structure

Figure 22.4 shows the block structure of the 256-Kbyte user MAT. The heavy-line frames indicate the erase blocks. The thin-line frames indicate the programming units and the values inside the frames stand for the addresses. The user MAT is divided into three 64-Kbyte blocks, one 32-Kbyte block, and eight 4-Kbyte blocks. The user MAT can be erased in these divided block units. Programming is done in 128-byte units starting from where the lower address is H'00 or H'80. RAM emulation can be performed in the eight 4-Kbyte blocks.

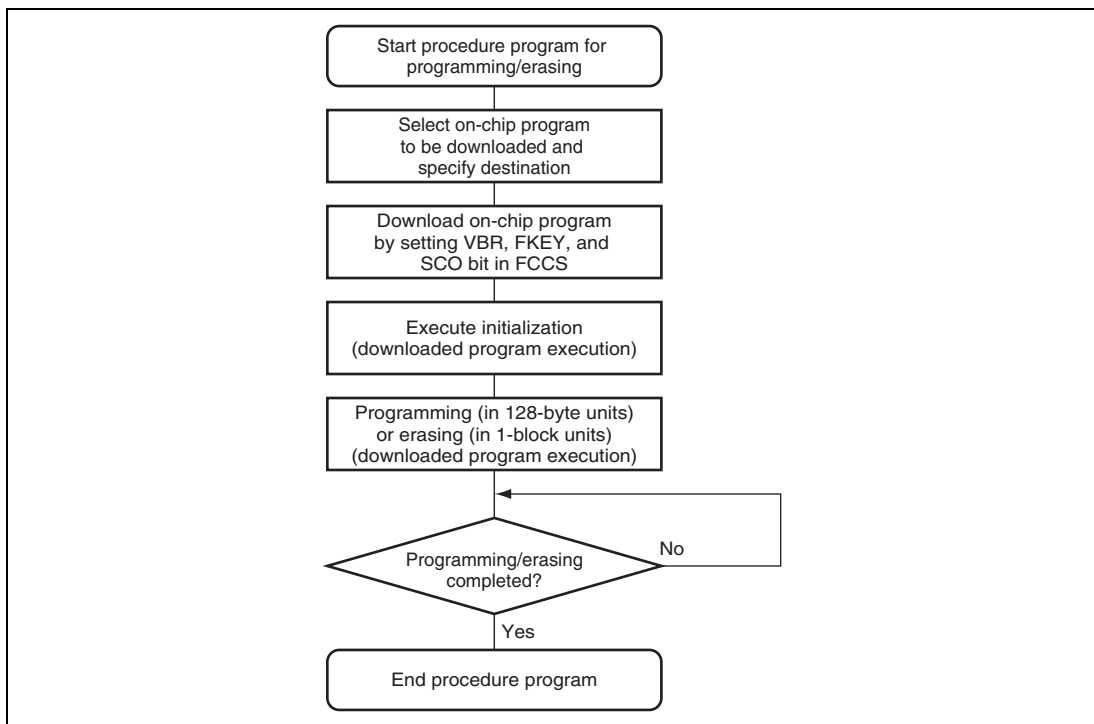


**Figure 22.4 User MAT Block Structure**

## 22.5 Programming/Erasing Interface

Programming/erasure of the flash memory is done by downloading an on-chip programming/erasing program to the on-chip RAM and specifying the start address of the programming destination, the program data, and the erase block number using the programming/erasing interface registers and programming/erasing interface parameters.

The procedure program for user program mode and user boot mode is made by the user. Figure 22.5 shows the procedure for creating the procedure program. For details, see section 22.8.2, User Program Mode.



**Figure 22.5 Procedure for Creating Procedure Program**

### (1) Selection of On-Chip Program to be Downloaded

This LSI has programming/erasing programs which can be downloaded to the on-chip RAM. The on-chip program to be downloaded is selected by the programming/erasing interface registers. The start address of the on-chip RAM where an on-chip program is downloaded is specified by the flash transfer destination address register (FTDAR).

## **(2) Download of On-Chip Program**

The on-chip program is automatically downloaded by setting the flash key code register (FKEY) and the SCO bit in the flash code control/status register (FCCS) after initializing the vector base register (VBR). The memory MAT is replaced with the embedded program storage area during download. Since the memory MAT cannot be read during programming/erasing, the procedure program must be executed in a space other than the flash memory (for example, on-chip RAM). Since the download result is returned to the programming/erasing interface parameter, whether download is normally executed or not can be confirmed. The VBR contents can be changed after completion of download.

## **(3) Initialization of Programming/Erasure**

A pulse with the specified period must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. Accordingly, the operating frequency of the CPU needs to be set before programming/erasure. The operating frequency of the CPU is set by the programming/erasing interface parameter.

## **(4) Execution of Programming/Erasure**

The start address of the programming destination and the program data are specified in 128-byte units when programming. The block to be erased is specified with the erase block number in erase-block units when erasing. Specifications of the start address of the programming destination, program data, and erase block number are performed by the programming/erasing interface parameters, and the on-chip program is initiated. The on-chip program is executed by using the JSR or BSR instruction and executing the subroutine call of the specified address in the on-chip RAM. The execution result is returned to the programming/erasing interface parameter.

The area to be programmed must be erased in advance when programming flash memory. All interrupts are disabled during programming/erasure.

## **(5) When Programming/Erasure is Executed Consecutively**

When processing does not end by 128-byte programming or 1-block erasure, consecutive programming/erasure can be realized by updating the start address of the programming destination and program data, or the erase block number. Since the downloaded on-chip program is left in the on-chip RAM even after programming/erasure completes, download and initialization are not required when the same processing is executed consecutively.

## 22.6 Input/Output Pins

The flash memory is controlled through the input/output pins shown in table 22.2.

**Table 22.2 Pin Configuration**

Abbreviation	I/O	Function
RES	Input	Reset
EMLE	Input	On-chip emulator enable pin (EMLE = 0 for flash memory programming/erasure)
MD2 to MD0	Input	Set operating mode of this LSI
TxD4	Output	Serial transmit data output (used in boot mode)
RxD4	Input	Serial receive data input (used in boot mode)

## 22.7 Register Descriptions

The flash memory has the following registers.

### Programming/Erasing Interface Registers:

- Flash code control/status register (FCCS)
- Flash program code select register (FPCS)
- Flash erase code select register (FECS)
- Flash key code register (FKEY)
- Flash MAT select register (FMATS)
- Flash transfer destination address register (FTDAR)

### Programming/Erasing Interface Parameters:

- Download pass and fail result parameter (DPFR)
- Flash pass and fail result parameter (FPFR)
- Flash program/erase frequency parameter (FPEFEQ)
- Flash multipurpose address area parameter (FMPAR)
- Flash multipurpose data destination area parameter (FMPDR)
- Flash erase block select parameter (FEBS)
- RAM emulation register (RAMER)

There are several operating modes for accessing the flash memory. Respective operating modes, registers, and parameters are assigned to the user MAT and user boot MAT. The correspondence between operating modes and registers/parameters for use is shown in table 22.3.

**Table 22.3 Registers/Parameters and Target Modes**

Register/Parameter		Down- load	Initiali- zation	Program- ming	Erase	Read	RAM Emulation
Programming/ erasing interface registers	FCCS	0	—	—	—	—	—
	FPCS	0	—	—	—	—	—
	FECS	0	—	—	—	—	—
	FKEY	0	—	0	0	—	—
	FMATS	—	—	0* <sup>1</sup>	0* <sup>1</sup>	0* <sup>2</sup>	—
	FTDAR	0	—	—	—	—	—
Programming/ erasing interface parameters	DPFR	0	—	—	—	—	—
	FPFR	—	0	0	0	—	—
	FPEFEQ	—	0	—	—	—	—
	FMPAR	—	—	0	—	—	—
	FMPDR	—	—	0	—	—	—
	FEBS	—	—	—	0	—	—
RAM emulation	RAMER	—	—	—	—	—	0

Notes: 1. The setting is required when programming or erasing the user MAT in user boot mode.  
2. The setting may be required according to the combination of initiation mode and read target memory MAT.

### 22.7.1 Programming/Erasing Interface Registers

The programming/erasing interface registers are 8-bit registers that can be accessed only in bytes. These registers are initialized by a reset.

#### (1) Flash Code Control/Status Register (FCCS)

FCCS monitors errors during programming/erasing the flash memory and requests the on-chip program to be downloaded to the on-chip RAM.



Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	FLER	—	—	—	SCO
Initial Value	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	(R)/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R	Reserved
6	—	0	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	FLER	0	R	<p>Flash Memory Error</p> <p>Indicates that an error has occurred during programming or erasing the flash memory. When this bit is set to 1, the flash memory enters the error protection state. When this bit is set to 1, high voltage is applied to the internal flash memory. To reduce the damage to the flash memory, the reset must be released after the reset input period (period of RES = 0) of at least 100 <math>\mu</math>s.</p> <p>0: Flash memory operates normally (Error protection is invalid)</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>At a reset</li> </ul> <p>1: An error occurs during programming/erasing flash memory (Error protection is valid)</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>When an interrupt, such as NMI, occurs during programming/erasure.</li> <li>When the flash memory is read during programming/erasure (including a vector read and an instruction fetch).</li> <li>When the SLEEP instruction is executed during programming/erasure (including software standby mode).</li> <li>When a bus master other than the CPU, such as the DMAC and DTC, obtains bus mastership during programming/erasure.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
3 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	SCO	0	(R)/W*	<p>Source Program Copy Operation</p> <p>Requests the on-chip programming/erasing program to be downloaded to the on-chip RAM. When this bit is set to 1, the on-chip program which is selected by FPCS or FECS is automatically downloaded in the on-chip RAM area specified by FTDAR.</p> <p>In order to set this bit to 1, the RAM emulation mode must be canceled, H'A5 must be written to FKEY, and this operation must be executed in the on-chip RAM. Dummy read of FCCS must be executed twice immediately after setting this bit to 1. All interrupts must be disabled during download. This bit is cleared to 0 when download is completed.</p> <p>During program download initiated with this bit, particular processing which accompanies bank-switching of the program storage area is executed. Before a download request, initialize the VBR contents to H'00000000. After download is completed, the VBR contents can be changed.</p> <p>0: Download of the programming/erasing program is not requested.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When download is completed</li> </ul> <p>1: Download of the programming/erasing program is requested.</p> <p>[Setting conditions] (When all of the following conditions are satisfied)</p> <ul style="list-style-type: none"> <li>Not in RAM emulation mode (the RAMS bit in RAMER is cleared to 0)</li> <li>H'A5 is written to FKEY</li> <li>Setting of this bit is executed in the on-chip RAM</li> </ul>

Note: \* This is a write-only bit. This bit is always read as 0.

**(2) Flash Program Code Select Register (FPCS)**

FPCS selects the programming program to be downloaded.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	PPVS
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	PPVS	0	R/W	Program Pulse Verify Selects the programming program to be downloaded. 0: Programming program is not selected. [Clearing condition] When transfer is completed 1: Programming program is selected.

**(3) Flash Erase Code Select Register (FECS)**

FECS selects the erasing program to be downloaded.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	EPVB
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	EPVB	0	R/W	Erase Pulse Verify Block Selects the erasing program to be downloaded. 0: Erasing program is not selected. [Clearing condition] When transfer is completed 1: Erasing program is selected.

**(4) Flash Key Code Register (FKEY)**

FKEY is a register for software protection that enables to download the on-chip program and perform programming/erasure of the flash memory.

Bit	7	6	5	4	3	2	1	0
Bit Name	K7	K6	K5	K4	K3	K2	K1	K0
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	Bit Name	Initial Value	R/W	Description
7	K7	0	R/W	Key Code
6	K6	0	R/W	When H'A5 is written to FKEY, writing to the SCO bit in FCCS is enabled. When a value other than H'A5 is written, the SCO bit cannot be set to 1. Therefore, the on-chip program cannot be downloaded to the on-chip RAM.
5	K5	0	R/W	
4	K4	0	R/W	
3	K3	0	R/W	
2	K2	0	R/W	Only when H'5A is written can programming/erasure of the flash memory be executed. When a value other than H'5A is written, even if the programming/erasing program is executed, programming/erasure cannot be performed.
1	K1	0	R/W	
0	K0	0	R/W	

H'A5: Writing to the SCO bit is enabled. (The SCO bit cannot be set to 1 when FKEY is a value other than H'A5.)

H'5A: Programming/erasure of the flash memory is enabled. (When FKEY is a value other than H'A5, the software protection state is entered.)

H'00: Initial value

**(5) Flash MAT Select Register (FMATS)**

FMATS selects the user MAT or user boot MAT. Writing to FMATS should be done when a program in the on-chip RAM is being executed.

Bit	7	6	5	4	3	2	1	0
Bit Name	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0
Initial Value	0/1*	0	0/1*	0	0/1*	0	0/1*	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* This bit is set to 1 in user boot mode, otherwise cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7	MS7	0/1*	R/W	MAT Select
6	MS6	0	R/W	The memory MATs can be switched by writing a value to FMATS.
5	MS5	0/1*	R/W	When H'AA is written to FMATS, the user boot MAT is selected. When a value other than H'AA is written, the user MAT is selected. Switch the MATs following the memory MAT switching procedure in section 22.11, Switching between User MAT and User Boot MAT. The user boot MAT cannot be selected by FMATS in user programming mode. The user boot MAT can be selected in boot mode or programmer mode.
4	MS4	0	R/W	
3	MS3	0/1*	R/W	
2	MS2	0	R/W	
1	MS1	0/1*	R/W	
0	MS0	0	R/W	

H'AA: The user boot MAT is selected. (The user MAT is selected when FMATS is a value other than H'AA.)  
(Initial value when initiated in user boot mode.)

H'00: The user MAT is selected.  
(Initial value when initiated in a mode except for user boot mode.)

Note: \* This bit is set to 1 in user boot mode, otherwise cleared to 0.

**(6) Flash Transfer Destination Address Register (FTDAR)**

FTDAR specifies the start address of the on-chip RAM at which to download an on-chip program. FTDAR must be set before setting the SCO bit in FCCS to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0
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	Bit Name	Initial Value	R/W	Description
7	TDER	0	R/W	<p>Transfer Destination Address Setting Error</p> <p>This bit is set to 1 when an error has occurred in setting the start address specified by bits TDA6 to TDA0.</p> <p>A start address error is determined by whether the value set in bits TDA6 to TDA0 is within the range of H'00 to H'02 when download is executed by setting the SCO bit in FCCS to 1. Make sure that this bit is cleared to 0 before setting the SCO bit to 1 and the value specified by bits TDA6 to TDA0 should be within the range of H'00 to H'02.</p> <p>0: The value specified by bits TDA6 to TDA0 is within the range.</p> <p>1: The value H'03 to H'FF, specified by bits TDER, and TDA6 to TDA0, stops download.</p>
6	TDA6	0	R/W	Transfer Destination Address
5	TDA5	0	R/W	Specifies the on-chip RAM start address of the download destination. By the value H'00 to H'02, and H'20, up to 4 Kbytes can be specified as the start address of the on-chip RAM.
4	TDA4	0	R/W	
3	TDA3	0	R/W	
2	TDA2	0	R/W	
1	TDA1	0	R/W	H'00: H'FF9000 is specified as the start address.
1	TDA1	0	R/W	H'01: H'FFA000 is specified as the start address.
0	TDA0	0	R/W	H'02: H'FFB000 is specified as the start address.
				H'03 to H'7F: Setting prohibited. (Specifying a value from H'03 to H'7F sets the TDER bit to 1 and stops download of the on-chip program.)

## 22.7.2 Programming/Erasing Interface Parameters

The programming/erasing interface parameters specify the operating frequency, storage place for program data, start address of programming destination, and erase block number, and exchanges the execution result. These parameters use the general registers of the CPU (ER0 and ER1) or the on-chip RAM area. The initial values of programming/erasing interface parameters are undefined at a reset or a transition to software standby mode.

Since registers of the CPU except for ER0 and ER1 are saved in the stack area during download of an on-chip program, initialization, programming, or erasing, allocate the stack area before performing these operations (the maximum stack size is 128 bytes). The return value of the processing result is written in R0L. The programming/erasing interface parameters are used in download control, initialization before programming or erasing, programming, and erasing. Table 22.4 shows the usable parameters and target modes. The meaning of the bits in the flash pass and fail result parameter (FPFR) varies in initialization, programming, and erasure.

**Table 22.4 Parameters and Target Modes**

Parameter	Download	Initialization	Programming	Erasure	R/W	Initial Value	Allocation
DPFR	○	—	—	—	R/W	Undefined	On-chip RAM*
FPFR	○	○	○	○	R/W	Undefined	R0L of CPU
FPEFEQ	—	○	—	—	R/W	Undefined	ER0 of CPU
FMPAR	—	—	○	—	R/W	Undefined	ER1 of CPU
FMPDR	—	—	○	—	R/W	Undefined	ER0 of CPU
FEBS	—	—	—	○	R/W	Undefined	ER0 of CPU

Note: \* A single byte of the start address of the on-chip RAM specified by FTDAR

**Download Control:** The on-chip program is automatically downloaded by setting the SCO bit in FCCS to 1. The on-chip RAM area to download the on-chip program is the 4-Kbyte area starting from the start address specified by FTDAR. Download is set by the programming/erasing interface registers, and the download pass and fail result parameter (DPFR) indicates the return value.

**Initialization before Programming/Erase:** The on-chip program includes the initialization program. A pulse with the specified period must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. Accordingly, the operating frequency of the CPU must be set. The initial program is set as a parameter of the programming/erasing program which has been downloaded to perform these settings.

**Programming:** When the flash memory is programmed, the start address of the programming destination on the user MAT and the program data must be passed to the programming program.

The start address of the programming destination on the user MAT must be stored in general register ER1. This parameter is called the flash multipurpose address area parameter (FMPAR).

The program data is always in 128-byte units. When the program data does not satisfy 128 bytes, 128-byte program data is prepared by filling the dummy code (H'FF). The boundary of the start address of the programming destination on the user MAT is aligned at an address where the lower eight bits (A7 to A0) are H'00 or H'80.

The program data for the user MAT must be prepared in consecutive areas. The program data must be in a consecutive space which can be accessed using the MOV.B instruction of the CPU and is not in the flash memory space.

The start address of the area that stores the data to be written in the user MAT must be set in general register ER0. This parameter is called the flash multipurpose data destination area parameter (FMPDR).

For details on the programming procedure, see section 22.8.2, User Program Mode.

**Erase:** When the flash memory is erased, the erase block number on the user MAT must be passed to the erasing program which is downloaded.

The erase block number on the user MAT must be set in general register ER0. This parameter is called the flash erase block select parameter (FEBS).

One block is selected from the block numbers of 0 to 11 as the erase block number.

For details on the erasing procedure, see section 22.8.2, User Program Mode.



### (1) Download Pass and Fail Result Parameter (DPFR: Single Byte of Start Address in On-Chip RAM Specified by FTDAR)

DPFR indicates the return value of the download result. The DPFR value is used to determine the download result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	SS	FK	SF

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused These bits return 0.
2	SS	—	R/W	Source Select Error Detect Only one type can be specified for the on-chip program which can be downloaded. When the program to be downloaded is not selected, more than two types of programs are selected, or a program which is not mapped is selected, an error occurs. 0: Download program selection is normal 1: Download program selection is abnormal
1	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'A5) and returns the result. 0: FKEY setting is normal (H'A5) 1: FKEY setting is abnormal (value other than H'A5)
0	SF	—	R/W	Success/Fail Returns the download result. Reads back the program downloaded to the on-chip RAM and determines whether it has been transferred to the on-chip RAM. 0: Download of the program has ended normally (no error) 1: Download of the program has ended abnormally (error occurs)

**(2) Flash Pass and Fail Parameter (FPFR: General Register R0L of CPU)**

FPFR indicates the return values of the initialization, programming, and erasure results. The meaning of the bits in FPFR varies depending on the processing.

**(a) Initialization before Programming/Erasure**

FPFR indicates the return value of the initialization result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	FQ	SF

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	—	—	Unused These bits return 0.
1	FQ	—	R/W	Frequency Error Detect Compares the specified CPU operating frequency with the operating frequencies supported by this LSI, and returns the result. 0: Setting of operating frequency is normal 1: Setting of operating frequency is abnormal
0	SF	—	R/W	Success/Fail Returns the initialization result. 0: Initialization has ended normally (no error) 1: Initialization has ended abnormally (error occurs)

**(b) Programming**

PPFR indicates the return value of the programming result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	—	WD	WA	SF

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Returns 0.
6	MD	—	R/W	<p>Programming Mode Related Setting Error Detect</p> <p>Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 22.9.3, Error Protection.</p> <p>0: Normal operation (FLER = 0) 1: Error protection state, and programming cannot be performed (FLER = 1)</p>
5	EE	—	R/W	<p>Programming Execution Error Detect</p> <p>Writes 1 to this bit when the specified data could not be written because the user MAT was not erased. If this bit is set to 1, there is a high possibility that the user MAT has been written to partially. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when programming is performed. In this case, both the user MAT and user boot MAT have not been written to. Programming the user boot MAT should be performed in boot mode or programmer mode.</p> <p>0: Programming has ended normally 1: Programming has ended abnormally (programming result is not guaranteed)</p>

Bit	Bit Name	Initial Value	R/W	Description
4	FK	—	R/W	<p>Flash Key Register Error Detect</p> <p>Checks the FKEY value (H'5A) before programming starts, and returns the result.</p> <p>0: FKEY setting is normal (H'5A)</p> <p>1: FKEY setting is abnormal (value other than H'5A)</p>
3	—	—	—	<p>Unused</p> <p>Returns 0.</p>
2	WD	—	R/W	<p>Write Data Address Detect</p> <p>When an address not in the flash memory area is specified as the start address of the storage destination for the program data, an error occurs.</p> <p>0: Setting of the start address of the storage destination for the program data is normal</p> <p>1: Setting of the start address of the storage destination for the program data is abnormal</p>
1	WA	—	R/W	<p>Write Address Error Detect</p> <p>When the following items are specified as the start address of the programming destination, an error occurs.</p> <ul style="list-style-type: none"> <li>• An area other than flash memory</li> <li>• The specified address is not aligned with the 128-byte boundary (lower eight bits of the address are other than H'00 and H'80)</li> </ul> <p>0: Setting of the start address of the programming destination is normal</p> <p>1: Setting of the start address of the programming destination is abnormal</p>
0	SF	—	R/W	<p>Success/Fail</p> <p>Returns the programming result.</p> <p>0: Programming has ended normally (no error)</p> <p>1: Programming has ended abnormally (error occurs)</p>

**(c) Erasure**

FPFR indicates the return value of the erasure result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	EB	—	—	SF

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Returns 0.
6	MD	—	R/W	Erasure Mode Related Setting Error Detect Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 22.9.3, Error Protection. 0: Normal operation (FLER = 0) 1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Erasure Execution Error Detect Returns 1 when the user MAT could not be erased or when the flash memory related register settings are partially changed. If this bit is set to 1, there is a high possibility that the user MAT has been erased partially. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when erasure is performed. In this case, both the user MAT and user boot MAT have not been erased. Erasing of the user boot MAT should be performed in boot mode or programmer mode. 0: Erasure has ended normally 1: Erasure has ended abnormally

Bit	Bit Name	Initial Value	R/W	Description
4	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'5A) before erasure starts, and returns the result. 0: FKEY setting is normal (H'5A) 1: FKEY setting is abnormal (value other than H'5A)
3	EB	—	R/W	Erase Block Select Error Detect Checks whether the specified erase block number is in the block range of the user MAT, and returns the result. 0: Setting of erase block number is normal 1: Setting of erase block number is abnormal
2, 1	—	—	—	Unused These bits return 0.
0	SF	—	R/W	Success/Fail Indicates the erasure result. 0: Erasure has ended normally (no error) 1: Erasure has ended abnormally (error occurs)

**(3) Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU)**

FPEFEQ sets the operating frequency of the CPU. The operating frequency available in this LSI ranges from 8 MHz to 50 MHz.

Bit	31	30	29	28	27	26	25	24
Bit Name	—	—	—	—	—	—	—	—
Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	—	—	—	—	—	—
Bit	15	14	13	12	11	10	9	8
Bit Name	F15	F14	F13	F12	F11	F10	F9	F8
Bit	7	6	5	4	3	2	1	0
Bit Name	F7	F6	F5	F4	F3	F2	F1	F0

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	—	—	Unused These bits should be cleared to 0.
15 to 0	F15 to F0	—	R/W	Frequency Set These bits set the operating frequency of the CPU. When the PLL multiplication function is used, set the multiplied frequency. The setting value must be calculated as follows: <ol style="list-style-type: none"> <li>The operating frequency shown in MHz units must be rounded in a number of three decimal places and be shown in a number of two decimal places.</li> <li>The value multiplied by 100 is converted to the binary digit and is written to FPEFEQ (general register ER0). For example, when the operating frequency of the CPU is 35.000 MHz, the value is as follows:  <ol style="list-style-type: none"> <li>The number of three decimal places of 35.000 is rounded.</li> <li>The formula of <math>35.00 \times 100 = 3500</math> is converted to the binary digit and B'0000 1101 1010 1100 (H'0DAC) is set to ER0.</li> </ol> </li> </ol>

**(4) Flash Multipurpose Address Area Parameter (FMPAR: General Register ER1 of CPU)**

FMPAR stores the start address of the programming destination on the user MAT.

When an address in an area other than the flash memory is set, or the start address of the programming destination is not aligned with the 128-byte boundary, an error occurs. The error occurrence is indicated by the WA bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOA31	MOA30	MOA29	MOA28	MOA27	MOA26	MOA25	MOA24
Bit	23	22	21	20	19	18	17	16
Bit Name	MOA23	MOA22	MOA21	MOA20	MOA19	MOA18	MOA17	MOA16
Bit	15	14	13	12	11	10	9	8
Bit Name	MOA15	MOA14	MOA13	MOA12	MOA11	MOA10	MOA9	MOA8
Bit	7	6	5	4	3	2	1	0
Bit Name	MOA7	MOA6	MOA5	MOA4	MOA3	MOA2	MOA1	MOA0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOA31 to MOA0	—	R/W	These bits store the start address of the programming destination on the user MAT. Consecutive 128-byte programming is executed starting from the specified start address of the user MAT. Therefore, the specified start address of the programming destination becomes a 128-byte boundary, and MOA6 to MOA0 are always cleared to 0.



### (5) Flash Multipurpose Data Destination Parameter (FMPDR: General Register ER0 of CPU)

FMPDR stores the start address in the area which stores the data to be programmed in the user MAT.

When the storage destination for the program data is in flash memory, an error occurs. The error occurrence is indicated by the WD bit in FPCR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOD31	MOD30	MOD29	MOD28	MOD27	MOD26	MOD25	MOD24

Bit	23	22	21	20	19	18	17	16
Bit Name	MOD23	MOD22	MOD21	MOD20	MOD19	MOD18	MOD17	MOD16

Bit	15	14	13	12	11	10	9	8
Bit Name	MOD15	MOD14	MOD13	MOD12	MOD11	MOD10	MOD9	MOD8

Bit	7	6	5	4	3	2	1	0
Bit Name	MOD7	MOD6	MOD5	MOD4	MOD3	MOD2	MOD1	MOD0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MOD0	—	R/W	These bits store the start address of the area which stores the program data for the user MAT. Consecutive 128-byte data is programmed to the user MAT starting from the specified start address.

**(6) Flash Erase Block Select Parameter (FEBS: General Register ER0 of CPU)**

FEBS specifies the erase block number. Settable values range from 0 to 11 (H'00000000 to H'0000000B). A value of 0 corresponds to block EB0 and a value of 11 corresponds to block EB11. An error occurs when a value over the range (from 0 to 11) is set.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
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
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

### 22.7.3 RAM Emulation Register (RAMER)

RAMER specifies the user MAT area overlaid with part of the on-chip RAM (H'FFFA000 to H'FFFAFFF) when performing emulation of programming the user MAT. RAMER should be set in user mode or user program mode. To ensure dependable emulation, the memory MAT to be emulated must not be accessed immediately after changing the RAMER contents. When accessed at such a timing, correct operation is not guaranteed.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	RAMS	RAM2	RAM1	RAM0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	0	R	Reserved These are read-only bits and cannot be modified.
3	RAMS	0	R/W	RAM Select Selects the function which emulates the flash memory using the on-chip RAM. 0: Disables RAM emulation function 1: Enables RAM emulation function (all blocks of the user MAT are protected against programming and erasing)
2	RAM2	0	R/W	Flash Memory Area Select
1	RAM1	0	R/W	These bits select the user MAT area overlaid with the on-chip RAM when RAMS = 1. The following areas correspond to the 4-Kbyte erase blocks. 000: H'000000 to H'000FFF (EB0) 001: H'001000 to H'001FFF (EB1) 010: H'002000 to H'002FFF (EB2) 011: H'003000 to H'003FFF (EB3) 100: H'004000 to H'004FFF (EB4) 101: H'005000 to H'005FFF (EB5) 110: H'006000 to H'006FFF (EB6) 111: H'007000 to H'007FFF (EB7)
0	RAM0	0	R/W	

## 22.8 On-Board Programming Mode

When the reset start is executed with a low level input to the EMLE pin and the mode pins (MD0, MD1, and MD2) set to on-board programming mode, a transition is made to on-board programming mode in which the on-chip flash memory can be programmed/erased. On-board programming mode has three operating modes: boot mode, user boot mode, and user program mode.

Table 22.5 shows the pin setting for each operating mode. For details on the state transition of each operating mode for flash memory, see figure 22.2.

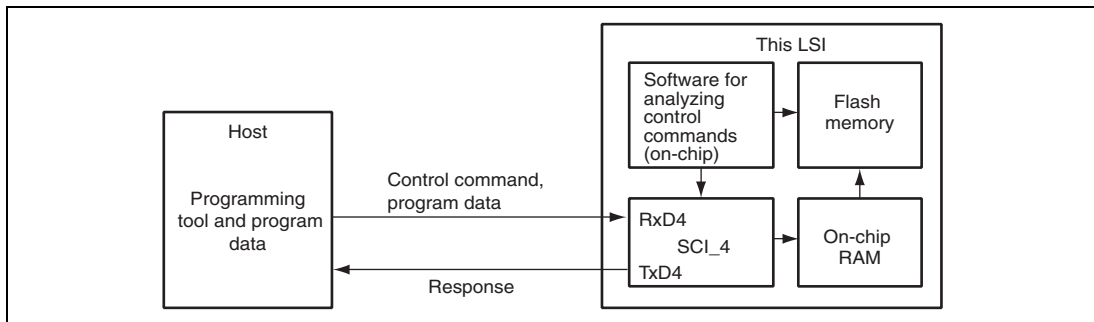
**Table 22.5 On-Board Programming Mode Setting**

Mode Setting	EMLE	MD2	MD1	MD0
User boot mode	0	0	0	1
Boot mode	0	0	1	0
User program mode	0	1	1	0
	0	1	1	1

### 22.8.1 Boot Mode

Boot mode executes programming/erasure of the user MAT or user boot MAT by means of the control command and program data transmitted from the externally connected host via the on-chip SCI\_4.

In boot mode, the tool for transmitting the control command and program data, and the program data must be prepared in the host. The serial communication mode is set to asynchronous mode. The system configuration in boot mode is shown in figure 22.6. Interrupts are ignored in boot mode. Configure the user system so that interrupts do not occur.



**Figure 22.6 System Configuration in Boot Mode**

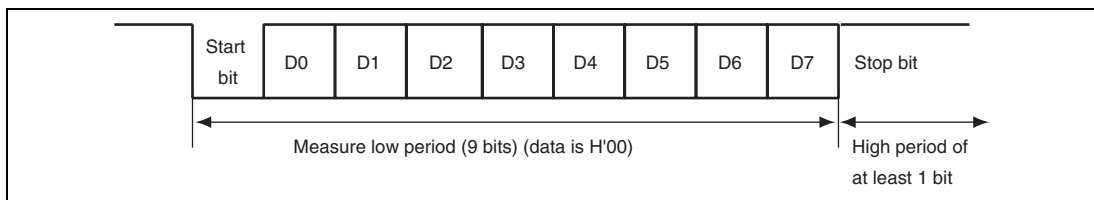
### (1) Serial Interface Setting by Host

The SCI\_4 is set to asynchronous mode, and the serial transmit/receive format is set to 8-bit data, one stop bit, and no parity.

When a transition to boot mode is made, the boot program embedded in this LSI is initiated.

When the boot program is initiated, this LSI measures the low period of asynchronous serial communication data (H'00) transmitted consecutively by the host, calculates the bit rate, and adjusts the bit rate of the SCI\_4 to match that of the host.

When bit rate adjustment is completed, this LSI transmits 1 byte of H'00 to the host as the bit adjustment end sign. When the host receives this bit adjustment end sign normally, it transmits 1 byte of H'55 to this LSI. When reception is not executed normally, initiate boot mode again. The bit rate may not be adjusted within the allowable range depending on the combination of the bit rate of the host and the system clock frequency of this LSI. Therefore, the transfer bit rate of the host and the system clock frequency of this LSI must be as shown in table 22.6.



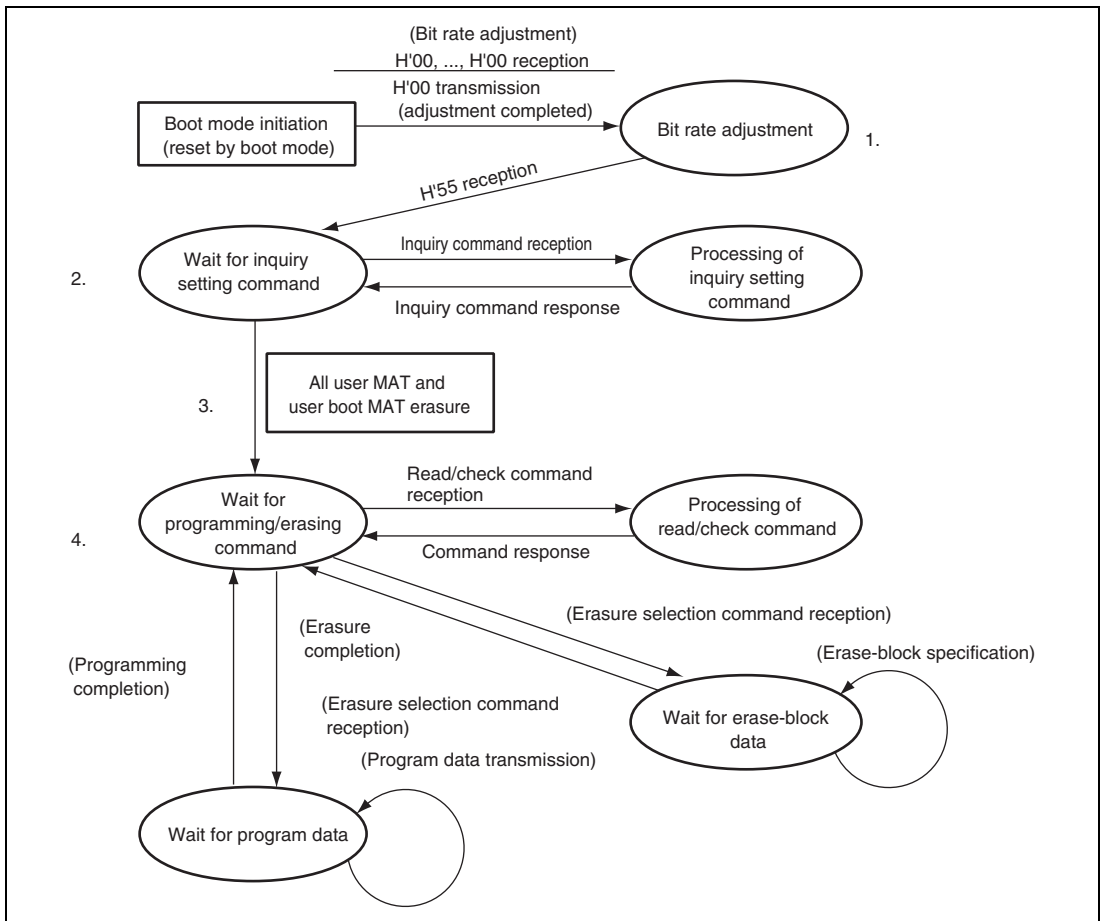
**Figure 22.7 Automatic-Bit-Rate Adjustment Operation**

**Table 22.6 System Clock Frequency for Automatic-Bit-Rate Adjustment**

Bit Rate of Host	System Clock Frequency of This LSI
9,600 bps	8 to 18 MHz
19,200 bps	8 to 18 MHz

**(2) State Transition Diagram**

The state transition after boot mode is initiated is shown in figure 22.8.

**Figure 22.8 Boot Mode State Transition Diagram**

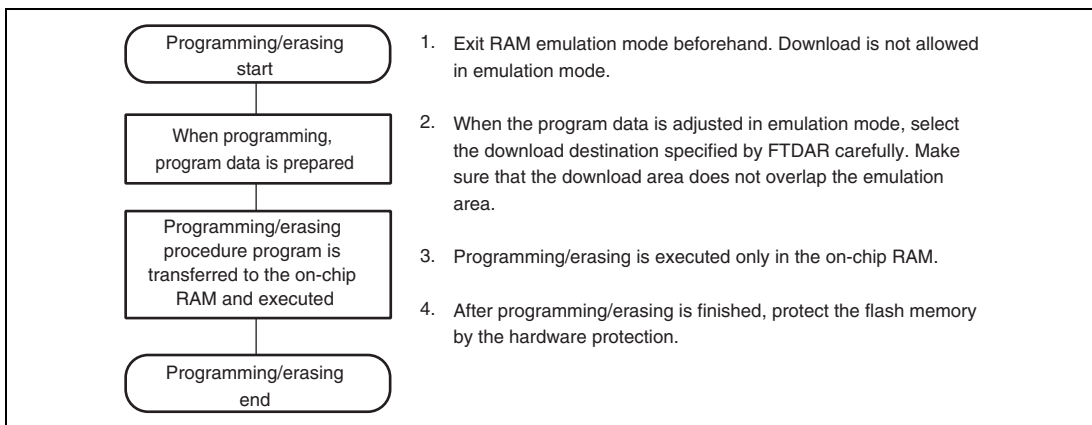
1. After boot mode is initiated, the bit rate of the SCI\_4 is adjusted with that of the host.
2. Inquiry information about the size, configuration, start address, and support status of the user MAT is transmitted to the host.
3. After inquiries have finished, all user MAT and user boot MAT are automatically erased.
4. When the program preparation notice is received, the state of waiting for program data is entered. The start address of the programming destination and program data must be transmitted after the programming command is transmitted. When programming is finished, the start address of the programming destination must be set to H'FFFFFFF and transmitted. Then the state of waiting for program data is returned to the state of waiting for programming/erasing command. When the erasure preparation notice is received, the state of waiting for erase block data is entered. The erase block number must be transmitted after the erasing command is transmitted. When the erasure is finished, the erase block number must be set to H'FF and transmitted. Then the state of waiting for erase block data is returned to the state of waiting for programming/erasing command. Erasure must be executed when the specified block is programmed without a reset start after programming is executed in boot mode. When programming can be executed by only one operation, all blocks are erased before entering the state of waiting for programming/erasing command or another command. Thus, in this case, the erasing operation is not required. The commands other than the programming/erasing command perform sum check, blank check (erasure check), and memory read of the user MAT/user boot MAT and acquisition of current status information.

Memory read of the user MAT/user boot MAT can only read the data programmed after all user MAT/user boot MAT has automatically been erased. No other data can be read.

## 22.8.2 User Program Mode

Programming/erasure of the user MAT is executed by downloading an on-chip program. The user boot MAT cannot be programmed/erased in user program mode. The programming/erasing flow is shown in figure 22.9.

Since high voltage is applied to the internal flash memory during programming/erasure, a transition to the reset state or hardware standby mode must not be made during programming/erasure. A transition to the reset state or hardware standby mode during programming/erasure may damage the flash memory. If a reset is input, the reset must be released after the reset input period (period of  $RES = 0$ ) of at least 100  $\mu s$ .

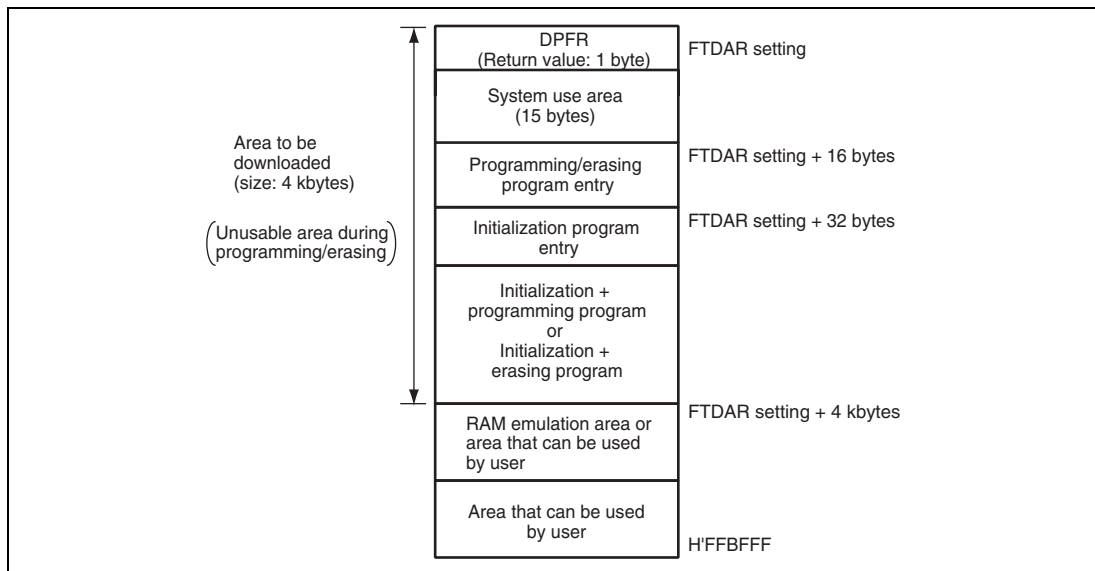


**Figure 22.9 Programming/Erasing Flow**



### (1) On-Chip RAM Address Map when Programming/Erase is Executed

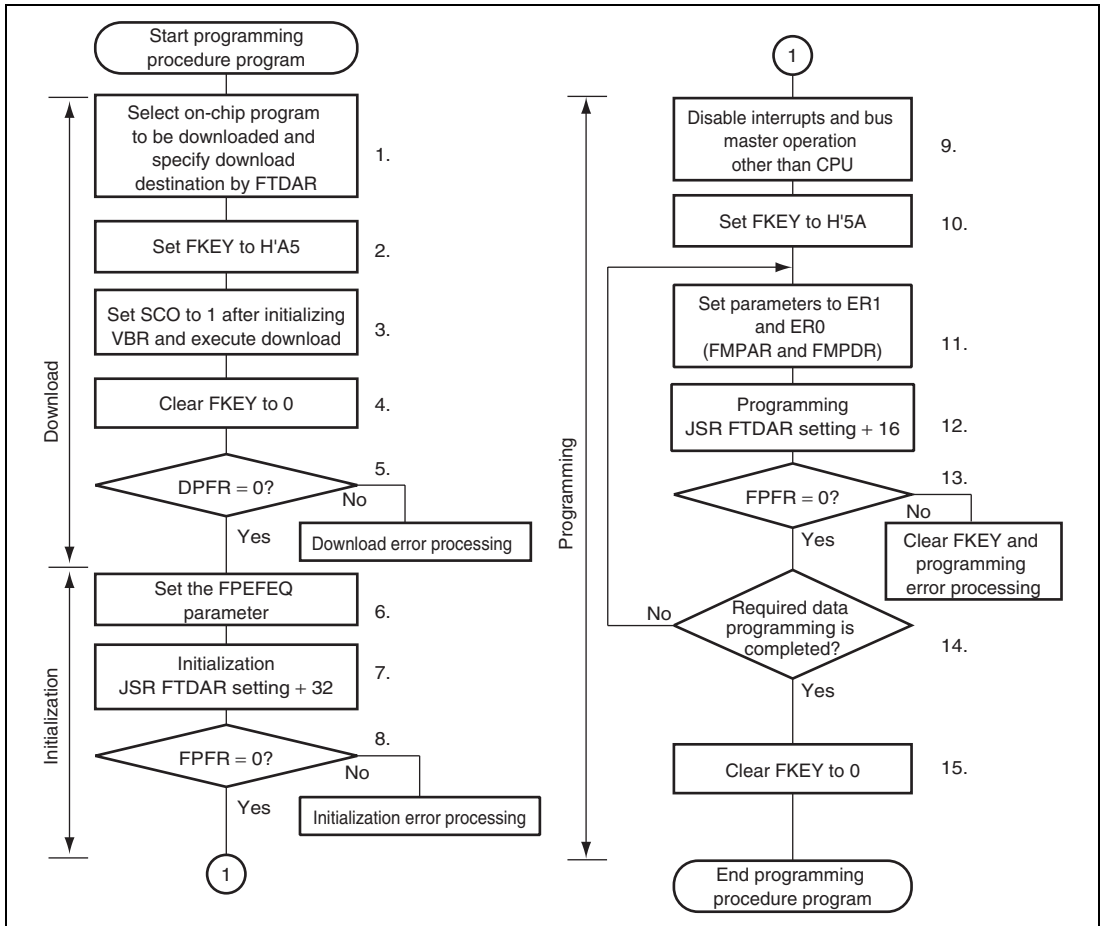
Parts of the procedure program that is made by the user, like download request, programming/erase procedure, and decision of the result, must be executed in the on-chip RAM. Since the on-chip program to be downloaded is embedded in the on-chip RAM, make sure the on-chip program and procedure program do not overlap. Figure 22.10 shows the area of the on-chip program to be downloaded.



**Figure 22.10 RAM Map when Programming/Erase is Executed**

## (2) Programming Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and programming are shown in figure 22.11.



**Figure 22.11 Programming Procedure in User Program Mode**

The procedure program must be executed in an area other than the flash memory to be programmed. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 22.8.4, On-Chip Program and Storable Area for Program Data. The following description assumes that the area to be programmed on the user MAT is erased and that program data is prepared in the consecutive area.

The program data for one programming operation is always 128 bytes. When the program data exceeds 128 bytes, the start address of the programming destination and program data parameters are updated in 128-byte units and programming is repeated. When the program data is less than 128 bytes, invalid data is filled to prepare 128-byte program data. If the invalid data to be added is H'FF, the program processing time can be shortened.

1. Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFRR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.
2. Write H'A5 in FKEY. If H'A5 is not written to FKEY, the SCO bit in FCCS cannot be set to 1 to request download of the on-chip program.
3. After initializing VBR to H'00000000, set the SCO bit to 1 to execute download. To set the SCO bit to 1, all of the following conditions must be satisfied.
  - RAM emulation mode has been canceled.
  - H'A5 is written to FKEY.
  - Setting the SCO bit is executed in the on-chip RAM.

When the SCO bit is set to 1, download is started automatically. Since the SCO bit is cleared to 0 when the procedure program is resumed, the SCO bit cannot be confirmed to be 1 in the procedure program. The download result can be confirmed by the return value of the DPFRR parameter. To prevent incorrect decision, before setting the SCO bit to 1, set one byte of the on-chip RAM start address specified by FTDAR, which becomes the DPFRR parameter, to a value other than the return value (e.g. H'FF). Since particular processing that is accompanied by bank switching as described below is performed when download is executed, initialize the VBR contents to H'00000000. Dummy read of FCCS must be performed twice immediately after the SCO bit is set to 1.

- The user-MAT space is switched to the on-chip program storage area.
- After the program to be downloaded and the on-chip RAM start address specified by FTDAR are checked, they are transferred to the on-chip RAM.
- FPCS, FECS, and the SCO bit in FCCS are cleared to 0.

- The return value is set in the DPFR parameter.
  - After the on-chip program storage area is returned to the user-MAT space, the procedure program is resumed. After that, VBR can be set again.
  - During download, the values of general registers other than ER0 and ER1 are held.
  - During download, no interrupts can be accepted. However, since the interrupt requests are held, when the procedure program is resumed, the interrupts are requested.
  - To hold a level-detection interrupt request, the interrupt must continue to be input until the download is completed.
  - Allocate a stack area of 128 bytes at the maximum in the on-chip RAM before setting the SCO bit to 1.
  - If access to the flash memory is requested by the DMAC or DTC during download, the operation cannot be guaranteed. Make sure that an access request by the DMAC or DTC is not generated.
4. FKEY is cleared to H'00 for protection.
  5. The download result must be confirmed by the value of the DPFR parameter. Check the value of the DPFR parameter (one byte of start address of the download destination specified by FTDAR). If the value of the DPFR parameter is H'00, download has been performed normally. If the value is not H'00, the source that caused download to fail can be investigated by the description below.
    - If the value of the DPFR parameter is the same as that before downloading, the setting of the start address of the download destination in FTDAR may be abnormal. In this case, confirm the setting of the TDER bit in FTDAR.
    - If the value of the DPFR parameter is different from that before downloading, check the SS bit or FK bit in the DPFR parameter to confirm the download program selection and FKEY setting, respectively.
  6. The operating frequency of the CPU is set in the FPEFEQ parameter for initialization. The settable operating frequency of the FPEFEQ parameter ranges from 8 to 50 MHz. When the frequency is set otherwise, an error is returned to the FPFPR parameter of the initialization program and initialization is not performed. For details on setting the frequency, see section 22.7.2 (3), Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU).

7. Initialization is executed. The initialization program is downloaded together with the programming program to the on-chip RAM. The entry point of the initialization program is at the address which is 32 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute initialization by using the following steps.

```

MOV.L #DLTOP+32,ER2      ; Set entry address to ER2
JSR   @ER2              ; Call initialization routine
NOP

```

- The general registers other than ER0 and ER1 are held in the initialization program.
  - R0L is a return value of the FPFR parameter.
  - Since the stack area is used in the initialization program, a stack area of 128 bytes at the maximum must be allocated in RAM.
  - Interrupts can be accepted during execution of the initialization program. Make sure the program storage area and stack area in the on-chip RAM and register values are not overwritten.
8. The return value in the initialization program, the FPFR parameter is determined.
9. All interrupts and the use of a bus master other than the CPU are disabled during programming/erasure. The specified voltage is applied for the specified time when programming or erasing. If interrupts occur or the bus mastership is moved to other than the CPU during programming/erasure, causing a voltage exceeding the specifications to be applied, the flash memory may be damaged. Therefore, interrupts are disabled by setting bit 7 (I bit) in the condition code register (CCR) to B'1 in interrupt control mode 0 and by setting bits 2 to 0 (I2 to I0 bits) in the extend register (EXR) to B'111 in interrupt control mode 2. Accordingly, interrupts other than NMI are held and not executed. Configure the user system so that NMI interrupts do not occur. The interrupts that are held must be executed after all programming completes. When the bus mastership is moved to other than the CPU, such as to the DMAC or DTC, the error protection state is entered. Therefore, make sure the DMAC does not acquire the bus.
10. FKEY must be set to H'5A and the user MAT must be prepared for programming.

11. The parameters required for programming are set. The start address of the programming destination on the user MAT (FMPAR parameter) is set in general register ER1. The start address of the program data storage area (FMPDR parameter) is set in general register ER0.
- Example of FMPAR parameter setting: When an address other than one in the user MAT area is specified for the start address of the programming destination, even if the programming program is executed, programming is not executed and an error is returned to the FPFR parameter. Since the program data for one programming operation is 128 bytes, the lower eight bits of the address must be H'00 or H'80 to be aligned with the 128-byte boundary.
  - Example of FMPDR parameter setting: When the storage destination for the program data is flash memory, even if the programming routine is executed, programming is not executed and an error is returned to the FPFR parameter. In this case, the program data must be transferred to the on-chip RAM and then programming must be executed.
12. Programming is executed. The entry point of the programming program is at the address which is 16 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute programming by using the following steps.
- |       |               |                            |
|-------|---------------|----------------------------|
| MOV.L | #DLTOP+16,ER2 | ; Set entry address to ER2 |
| JSR   | @ER2          | ; Call programming routine |
| NOB   |               |                            |
- The general registers other than ER0 and ER1 are held in the programming program.
  - R0L is a return value of the FPFR parameter.
  - Since the stack area is used in the programming program, a stack area of 128 bytes at the maximum must be allocated in RAM.
13. The return value in the programming program, the FPFR parameter is determined.
14. Determine whether programming of the necessary data has finished. If more than 128 bytes of data are to be programmed, update the FMPAR and FMPDR parameters in 128-byte units, and repeat steps 11 to 14. Increment the programming destination address by 128 bytes and update the programming data pointer correctly. If an address which has already been programmed is written to again, not only will a programming error occur, but also flash memory will be damaged.
15. After programming finishes, clear FKEY and specify software protection. If this LSI is restarted by a reset immediately after programming has finished, secure the reset input period (period of  $\overline{\text{RES}} = 0$ ) of at least 100  $\mu\text{s}$ .

### (3) Erasing Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and erasing are shown in figure 22.12.

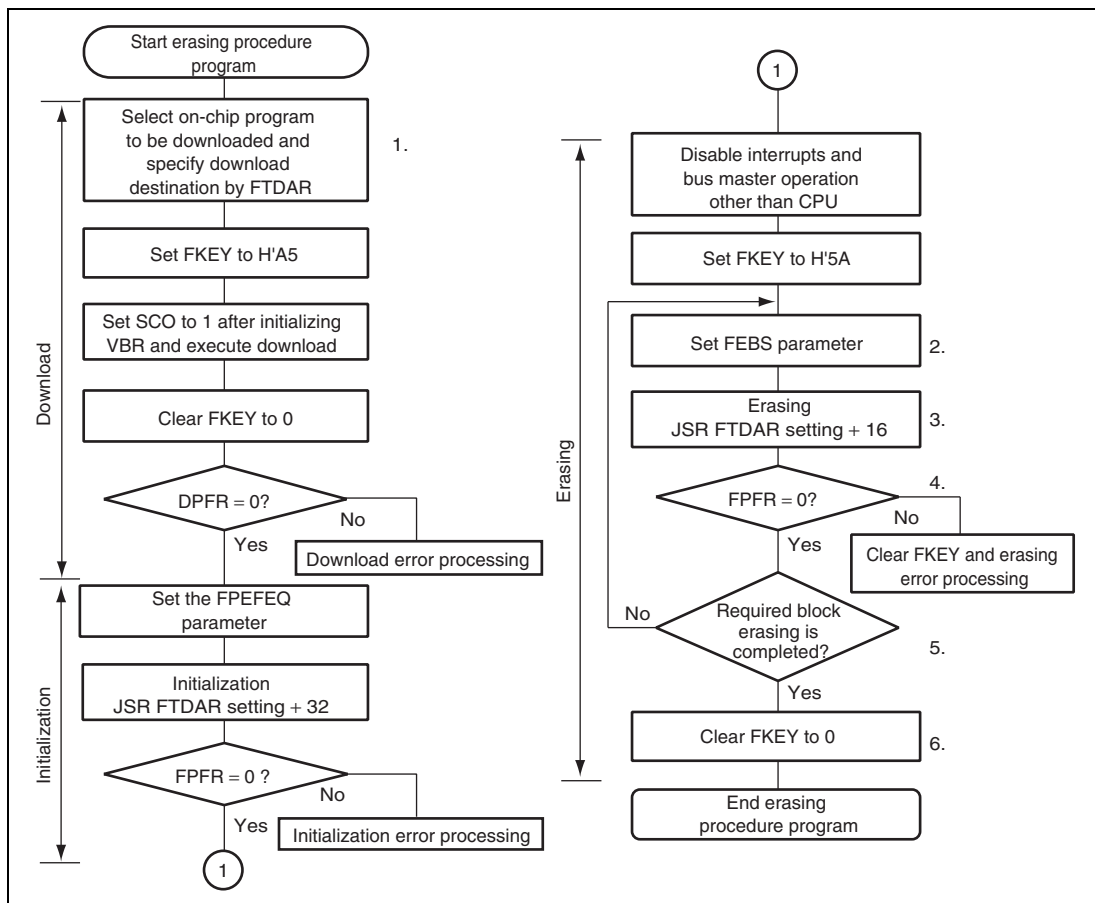


Figure 22.12 Erasing Procedure in User Program Mode

The procedure program must be executed in an area other than the user MAT to be erased. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM and user MAT) is shown in section 22.8.4, On-Chip Program and Storable Area for Program Data. For the downloaded on-chip program area, see figure 22.10.

One erasure processing erases one block. For details on block divisions, refer to figure 22.4. To erase two or more blocks, update the erase block number and repeat the erasing processing for each block.

1. Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.

For the procedures to be carried out after setting FKEY, see section 22.8.2 (2), Programming Procedure in User Program Mode.

2. Set the FEBS parameter necessary for erasure. Set the erase block number (FEBS parameter) of the user MAT in general register ER0. If a value other than an erase block number of the user MAT is set, no block is erased even though the erasing program is executed, and an error is returned to the FPFR parameter.
3. Erasure is executed. Similar to as in programming, the entry point of the erasing program is at the address which is 16 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute erasure by using the following steps.

```
MOV.L #DLTOP+16, ER2      ; Set entry address to ER2
JSR  @ER2                 ; Call erasing routine
NOP
```

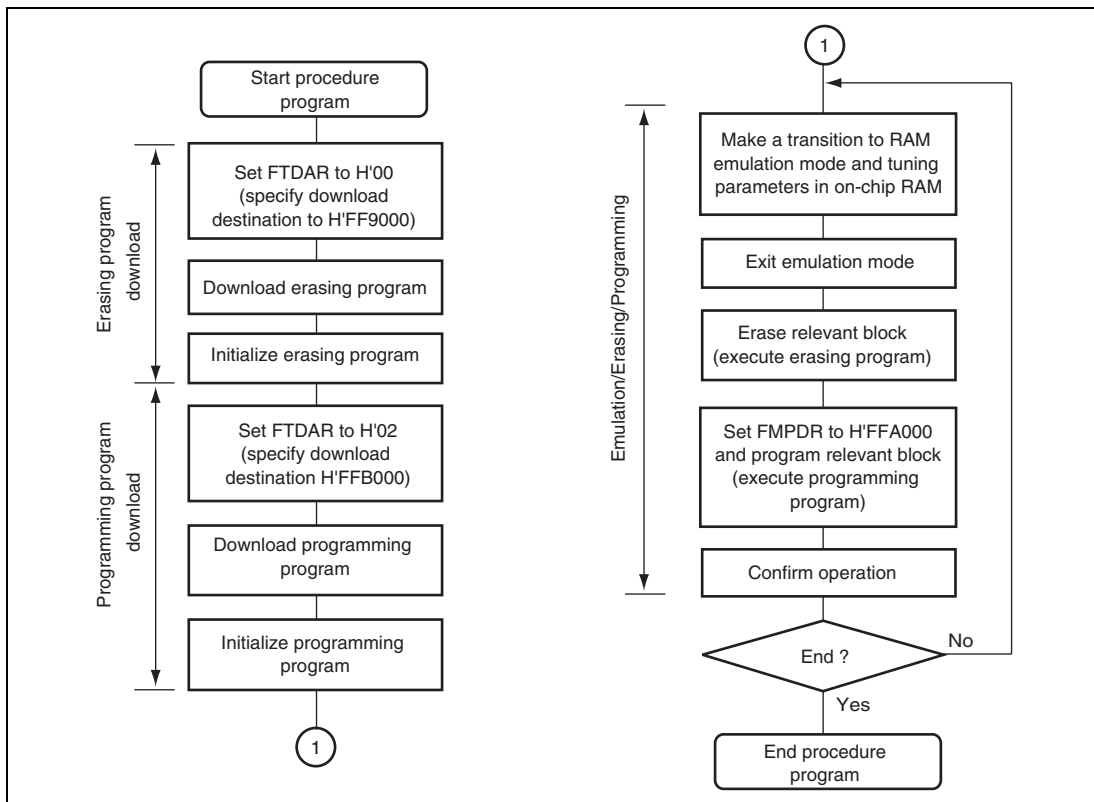
- The general registers other than ER0 and ER1 are held in the erasing program.
  - R0L is a return value of the FPFR parameter.
  - Since the stack area is used in the erasing program, a stack area of 128 bytes at the maximum must be allocated in RAM.
4. The return value in the erasing program, the FPFR parameter is determined.
  5. Determine whether erasure of the necessary blocks has finished. If more than one block is to be erased, update the FEBS parameter and repeat steps 2 to 5.
  6. After erasure completes, clear FKEY and specify software protection. If this LSI is restarted by a reset immediately after erasure has finished, secure the reset input period (period of  $\overline{RES} = 0$ ) of at least 100  $\mu\text{s}$ .



#### (4) Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

By changing the on-chip RAM start address of the download destination in FTDAR, the erasing program and programming program can be downloaded to separate on-chip RAM areas.

Figure 22.13 shows a repeating procedure of erasing, programming, and RAM emulation.



**Figure 22.13 Repeating Procedure of Erasing, Programming, and RAM Emulation in User Program Mode**

In figure 22.13, since RAM emulation is performed, the erasing/programming program is downloaded to avoid the 4-Kbyte on-chip RAM area (H'FFA000 to H'FFAFFF). Download and initialization are performed only once at the beginning. Note the following when executing the procedure program.

- Be careful not to overwrite data in the on-chip RAM with overlay settings. In addition to the programming program area, erasing program area, and RAM emulation area, areas for the procedure programs, work area, and stack area are reserved in the on-chip RAM. Do not make settings that will overwrite data in these areas.
- Be sure to initialize both the programming program and erasing program. When the FPEFEQ parameter is initialized, also initialize both the erasing program and programming program. Initialization must be executed for both entry addresses: #DLTOP (start address of download destination for erasing program) + 32 bytes, and #DLTOP (start address of download destination for programming program) + 32 bytes.

### 22.8.3 User Boot Mode

Branching to a programming/erasing program prepared by the user enables user boot mode which is a user-arbitrary boot mode to be used.

Only the user MAT can be programmed/erased in user boot mode. Programming/erasure of the user boot MAT is only enabled in boot mode or programmer mode.

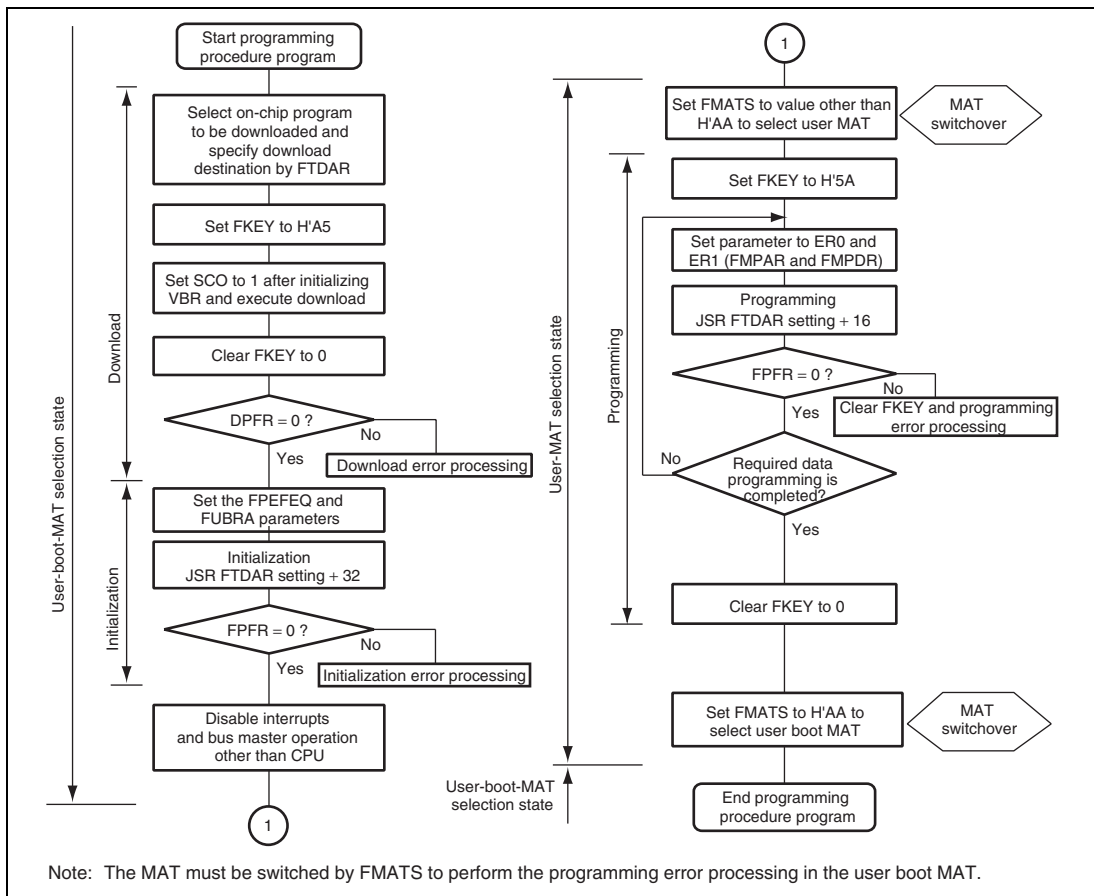
#### (1) Initiation in User Boot Mode

When the reset start is executed with the mode pins set to user boot mode, the built-in check routine runs and checks the user MAT and user boot MAT states. While the check routine is running, NMI and all other interrupts cannot be accepted. Next, processing starts from the execution start address of the reset vector in the user boot MAT. At this point, the user boot MAT is selected (FMATS = H'AA) as the execution memory MAT.

## (2) User MAT Programming in User Boot Mode

Figure 22.14 shows the procedure for programming the user MAT in user boot mode.

The difference between the programming procedures in user program mode and user boot mode is the memory MAT switching as shown in figure 22.14. For programming the user MAT in user boot mode, additional processing made by setting FMATS is required: switching from the user boot MAT to the user MAT, and switching back to the user boot MAT after programming completes.



**Figure 22.14 Procedure for Programming User MAT in User Boot Mode**

In user boot mode, though the user boot MAT can be seen in the flash memory space, the user MAT is hidden in the background. Therefore, the user MAT and user boot MAT are switched while the user MAT is being programmed. Because the user boot MAT is hidden while the user MAT is being programmed, the procedure program must be executed in an area other than flash memory. After programming completes, switch the memory MATs again to return to the first state.

Memory MAT switching is enabled by setting FMATS. However note that access to a memory MAT is not allowed until memory MAT switching is completed. During memory MAT switching, the LSI is in an unstable state, e.g. if an interrupt occurs, from which memory MAT the interrupt vector is read is undetermined. Perform memory MAT switching in accordance with the description in section 22.11, Switching between User MAT and User Boot MAT.

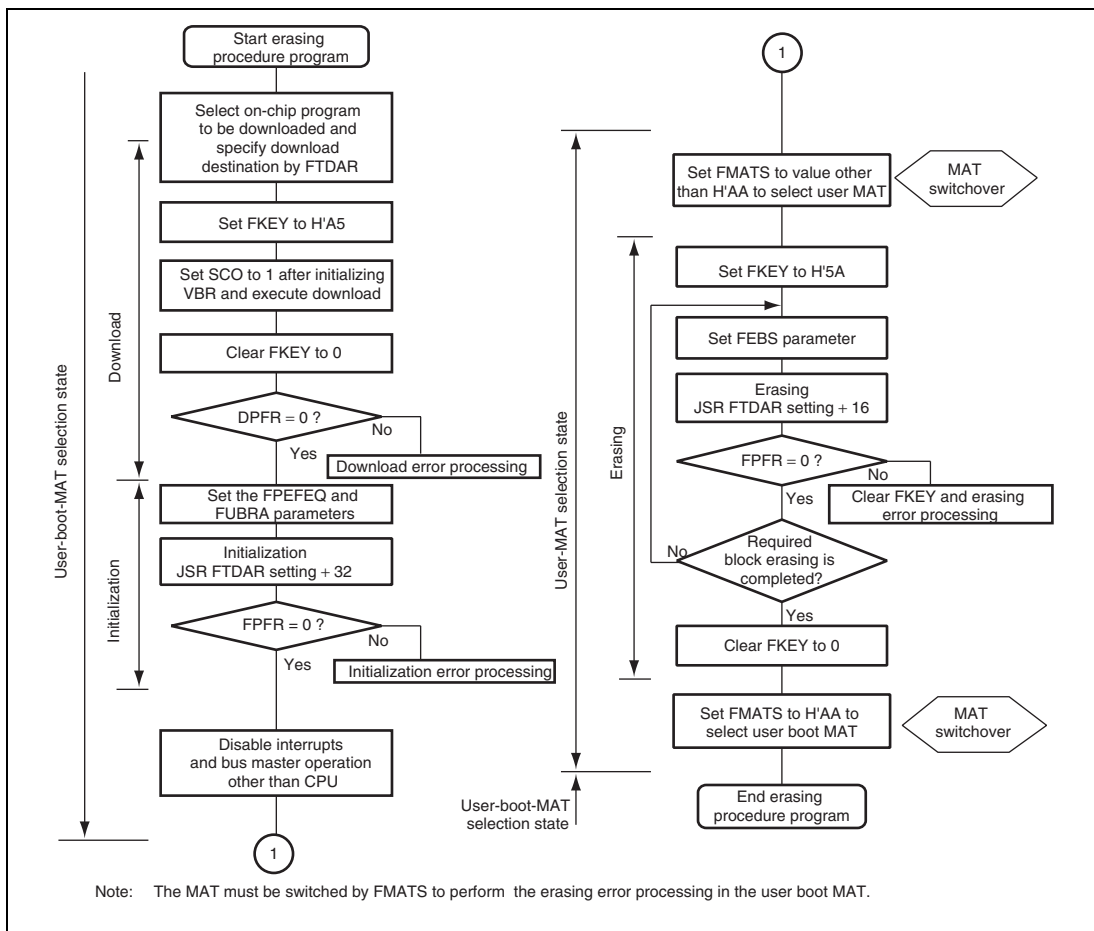
Except for memory MAT switching, the programming procedure is the same as that in user program mode.

The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 22.8.4, On-Chip Program and Storable Area for Program Data.

### **(3) User MAT Erasing in User Boot Mode**

Figure 22.15 shows the procedure for erasing the user MAT in user boot mode.

The difference between the erasing procedures in user program mode and user boot mode is the memory MAT switching as shown in figure 22.15. For erasing the user MAT in user boot mode, additional processing made by setting FMATS is required: switching from the user boot MAT to the user MAT, and switching back to the user boot MAT after erasing completes.



**Figure 22.15 Procedure for Erasing User MAT in User Boot Mode**

Memory MAT switching is enabled by setting FMATS. However note that access to a memory MAT is not allowed until memory MAT switching is completed. During memory MAT switching, the LSI is in an unstable state, e.g. if an interrupt occurs, from which memory MAT the interrupt vector is read is undetermined. Perform memory MAT switching in accordance with the description in section 22.11, Switching between User MAT and User Boot MAT.

Except for memory MAT switching, the erasing procedure is the same as that in user program mode.

The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 22.8.4, On-Chip Program and Storable Area for Program Data.

## 22.8.4 On-Chip Program and Storable Area for Program Data

In the descriptions in this manual, the on-chip programs and program data storage areas are assumed to be in the on-chip RAM. However, they can be executed from part of the flash memory which is not to be programmed or erased as long as the following conditions are satisfied.

- The on-chip program is downloaded to and executed in the on-chip RAM specified by FTDAR. Therefore, this on-chip RAM area is not available for use.
- Since the on-chip program uses a stack area, allocate 128 bytes at the maximum as a stack area.
- Download requested by setting the SCO bit in FCCS to 1 should be executed from the on-chip RAM because it will require switching of the memory MATs.
- In an operating mode in which the external address space is not accessible, such as single-chip mode, the required procedure programs, NMI handling vector table, and NMI handling routine should be transferred to the on-chip RAM before programming/erasure starts (download result is determined).
- The flash memory is not accessible during programming/erasure. Programming/erasure is executed by the program downloaded to the on-chip RAM. Therefore, the procedure program that initiates operation, the NMI handling vector table, and the NMI handling routine should be stored in the on-chip RAM other than the flash memory.
- After programming/erasure starts, access to the flash memory should be inhibited until FKEY is cleared. The reset input state (period of  $\overline{\text{RES}} = 0$ ) must be set to at least 100  $\mu\text{s}$  when the operating mode is changed and the reset start executed on completion of programming/erasure. Transitions to the reset state are inhibited during programming/erasure. When the reset signal is input, a reset input state (period of  $\text{RES} = 0$ ) of at least 100  $\mu\text{s}$  is needed before the reset signal is released.
- Switching of the memory MATs by FMATS should be needed when programming/erasure of the user MAT is operated in user boot mode. The program which switches the memory MATs should be executed from the on-chip RAM. For details, see section 22.11, Switching between User MAT and User Boot MAT. Make sure you know which memory MAT is currently selected when switching them.
- When the program data storage area is within the flash memory area, an error will occur even when the data stored is normal program data. Therefore, the data should be transferred to the on-chip RAM to place the address that the FMPDR parameter indicates in an area other than the flash memory.

In consideration of these conditions, the areas in which the program data can be stored and executed are determined by the combination of the processing contents, operating mode, and bank structure of the memory MATs, as shown in tables 22.7 to 22.11.

**Table 22.7 Executable Memory MAT**

<b>Processing Contents</b>	<b>Operating Mode</b>	
	<b>User Program Mode</b>	<b>User Boot Mode*</b>
Programming	See table 22.8	See table 22.10
Erasing	See table 22.9	See table 22.11

Note: \* Programming/Erasure is possible to the user MAT.

**Table 22.8 Usable Area for Programming in User Program Mode**

Item	Storable/Executable Area		Selected MAT	
	On-Chip RAM	User MAT	User MAT	Embedded Program Storage MAT
Storage area for program data	○	×*	—	—
Operation for selecting on-chip program to be downloaded	○	○	○	
Operation for writing H'A5 to FKEY	○	○	○	
Execution of writing 1 to SCO bit in FCCS (download)	○	×		○
Operation for clearing FKEY	○	○	○	
Decision of download result	○	○	○	
Operation for download error	○	○	○	
Operation for setting initialization parameter	○	○	○	
Execution of initialization	○	×	○	
Decision of initialization result	○	○	○	
Operation for initialization error	○	○	○	
NMI handling routine	○	×	○	
Operation for disabling interrupts	○	○	○	
Operation for writing H'5A to FKEY	○	○	○	
Operation for setting programming parameter	○	×	○	
Execution of programming	○	×	○	
Decision of programming result	○	×	○	
Operation for programming error	○	×	○	
Operation for clearing FKEY	○	×	○	

Note: \* Transferring the program data to the on-chip RAM beforehand enables this area to be used.



**Table 22.9 Usable Area for Erasure in User Program Mode**

Item	Storable/Executable Area		Selected MAT	
	On-Chip RAM	User MAT	User MAT	Embedded Program Storage MAT
Operation for selecting on-chip program to be downloaded	○	○	○	
Operation for writing H'A5 to FKEY	○	○	○	
Execution of writing 1 to SCO bit in FCCS (download)	○	×		○
Operation for clearing FKEY	○	○	○	
Decision of download result	○	○	○	
Operation for download error	○	○	○	
Operation for setting initialization parameter	○	○	○	
Execution of initialization	○	×	○	
Decision of initialization result	○	○	○	
Operation for initialization error	○	○	○	
NMI handling routine	○	×	○	
Operation for disabling interrupts	○	○	○	
Operation for writing H'5A to FKEY	○	○	○	
Operation for setting erasure parameter	○	×	○	
Execution of erasure	○	×	○	
Decision of erasure result	○	×	○	
Operation for erasure error	○	×	○	
Operation for clearing FKEY	○	×	○	

**Table 22.10 Usable Area for Programming in User Boot Mode**

Item	Storable/Executable Area			Selected MAT	
	On-Chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage MAT
Storage area for program data	O	×* <sup>1</sup>	—	—	—
Operation for selecting on-chip program to be downloaded	O	O		O	
Operation for writing H'A5 to FKEY	O	O		O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×			O
Operation for clearing FKEY	O	O		O	
Decision of download result	O	O		O	
Operation for download error	O	O		O	
Operation for setting initialization parameter	O	O		O	
Execution of initialization	O	×		O	
Decision of initialization result	O	O		O	
Operation for initialization error	O	O		O	
NMI handling routine	O	×		O	
Operation for disabling interrupts	O	O		O	
Switching memory MATs by FMATS	O	×	O		
Operation for writing H'5A to FKEY	O	×	O		
Operation for setting programming parameter	O	×	O		
Execution of programming	O	×	O		
Decision of programming result	O	×	O		
Operation for programming error	O	×* <sup>2</sup>	O		
Operation for clearing FKEY	O	×	O		
Switching memory MATs by FMATS	O	×		O	

Notes: 1. Transferring the program data to the on-chip RAM beforehand enables this area to be used.

2. Switching memory MATs by FMATS by a program in the on-chip RAM enables this area to be used.

**Table 22.11 Usable Area for Erasure in User Boot Mode**

Item	Storable/Executable Area			Selected MAT	
	On-Chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage MAT
Operation for selecting on-chip program to be downloaded	○	○		○	
Operation for writing H'A5 to FKEY	○	○		○	
Execution of writing 1 to SCO bit in FCCS (download)	○	×			○
Operation for clearing FKEY	○	○		○	
Decision of download result	○	○		○	
Operation for download error	○	○		○	
Operation for setting initialization parameter	○	○		○	
Execution of initialization	○	×		○	
Decision of initialization result	○	○		○	
Operation for initialization error	○	○		○	
NMI handling routine	○	×		○	
Operation for disabling interrupts	○	○		○	
Switching memory MATs by FMATS	○	×	○		
Operation for writing H'5A to FKEY	○	×	○		
Operation for setting erasure parameter	○	×	○		
Execution of erasure	○	×	○		
Decision of erasure result	○	×	○		
Operation for erasure error	○	×*	○		
Operation for clearing FKEY	○	×	○		
Switching memory MATs by FMATS	○	×	○		

Note: Switching memory MATs by FMATS by a program in the on-chip RAM enables this area to be used.

## 22.9 Protection

There are three types of protection against the flash memory programming/erasure: hardware protection, software protection, and error protection.

### 22.9.1 Hardware Protection

Programming and erasure of the flash memory is forcibly disabled or suspended by hardware protection. In this state, download of an on-chip program and initialization are possible. However, programming or erasure of the user MAT cannot be performed even if the programming/erasing program is initiated, and the error in programming/erasure is indicated by the FPFR parameter.

**Table 22.12 Hardware Protection**

Item	Description	Function to be Protected	
		Download	Programming/ Erasing
Reset protection	<ul style="list-style-type: none"> <li>The programming/erasing interface registers are initialized in the reset state (including a reset by the WDT) and the programming/erasing protection state is entered.</li> <li>The reset state will not be entered by a reset using the <math>\overline{\text{RES}}</math> pin unless the <math>\overline{\text{RES}}</math> pin is held low until oscillation has settled after a power is initially supplied. 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 given in the AC characteristics. If a reset is input during programming or erasure, data in the flash memory is not guaranteed. In this case, execute erasure and then execute programming again.</li> </ul>	O	O

## 22.9.2 Software Protection

The software protection protects the flash memory against programming/erasure by disabling download of the programming/erasing program, using the key code, and by the RAMER setting.

**Table 22.13 Software Protection**

Item	Description	Function to be Protected	
		Download	Programming/ Erasing
Protection by SCO bit	The programming/erasing protection state is entered when the SCO bit in FCCS is cleared to 0 to disable download of the programming/erasing programs.	○	○
Protection by FKEY	The programming/erasing protection state is entered because download and programming/erasure are disabled unless the required key code is written in FKEY.	○	○
Emulation protection	The programming/erasing protection state is entered when the RAMS bit in the RAM emulation register (RAMER) is set to 1.	○	○

## 22.9.3 Error Protection

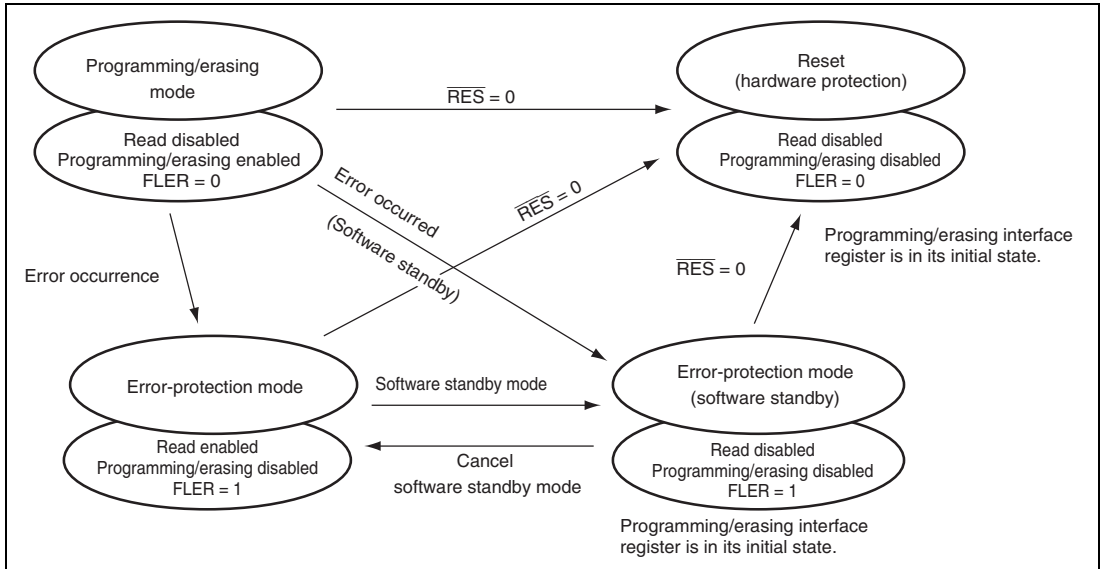
Error protection is a mechanism for aborting programming or erasure when a CPU runaway occurs or operations not according to the programming/erasing procedures are detected during programming/erasure of the flash memory. Aborting programming or erasure in such cases prevents damage to the flash memory due to excessive programming or erasing.

If an error occurs during programming/erasure of the flash memory, the FLER bit in FCCS is set to 1 and the error protection state is entered.

- When an interrupt request, such as NMI, occurs during programming/erasure.
- When the flash memory is read from during programming/erasure (including a vector read or an instruction fetch).
- When a SLEEP instruction is executed (including software-standby mode) during programming/erasure.
- When a bus master other than the CPU, such as the DMAC and DTC, obtains bus mastership during programming/erasure.

Error protection is canceled by a reset. Note that the reset should be released after the reset input period of at least 100 $\mu$ s has passed. Since high voltages are applied during programming/erasure of the flash memory, some voltage may remain after the error protection state has been entered. For this reason, it is necessary to reduce the risk of damaging the flash memory by extending the reset input period so that the charge is released.

The state-transition diagram in figure 22.16 shows transitions to and from the error protection state.



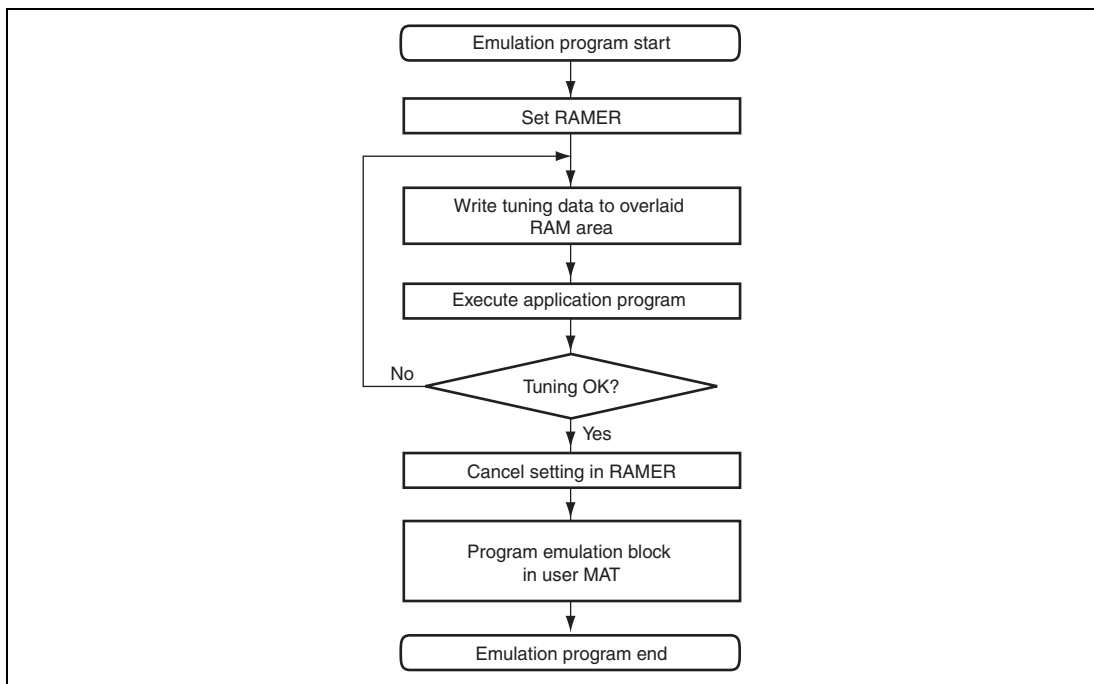
**Figure 22.16 Transitions to Error Protection State**

## 22.10 Flash Memory Emulation Using RAM

For realtime emulation of the data written to the flash memory using the on-chip RAM, the on-chip RAM area can be overlaid with several flash memory blocks (user MAT) using the RAM emulation register (RAMER).

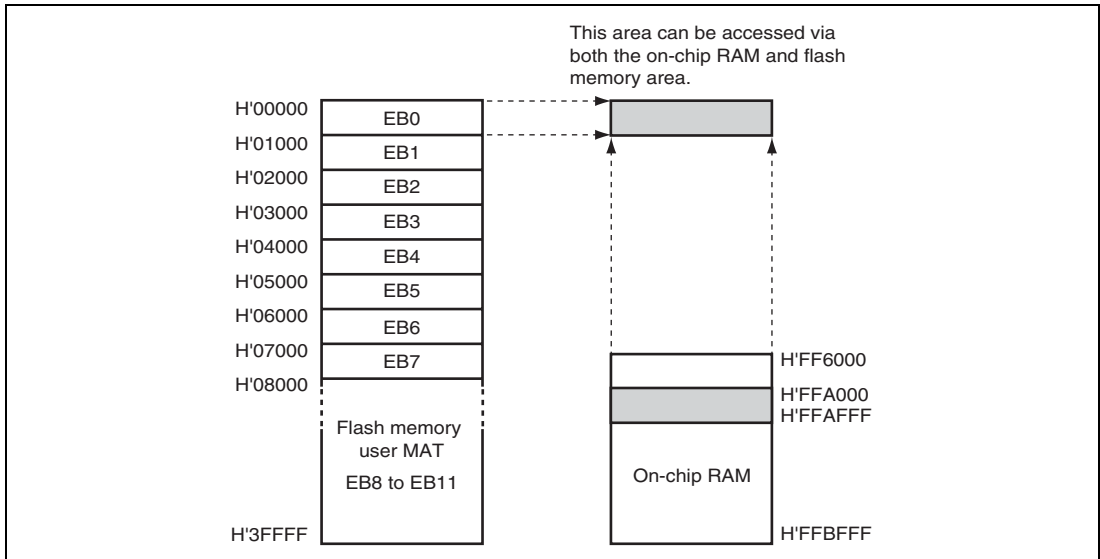
The overlaid area can be accessed from both the user MAT area specified by RAMER and the overlaid RAM area. The emulation can be performed in user mode and user program mode.

Figure 22.17 shows an example of emulating realtime programming of the user MAT.



**Figure 22.17 RAM Emulation Flow**

Figure 22.18 shows an example of overlaying flash memory block area EB0.



**Figure 22.18 Address Map of Overlaid RAM Area**

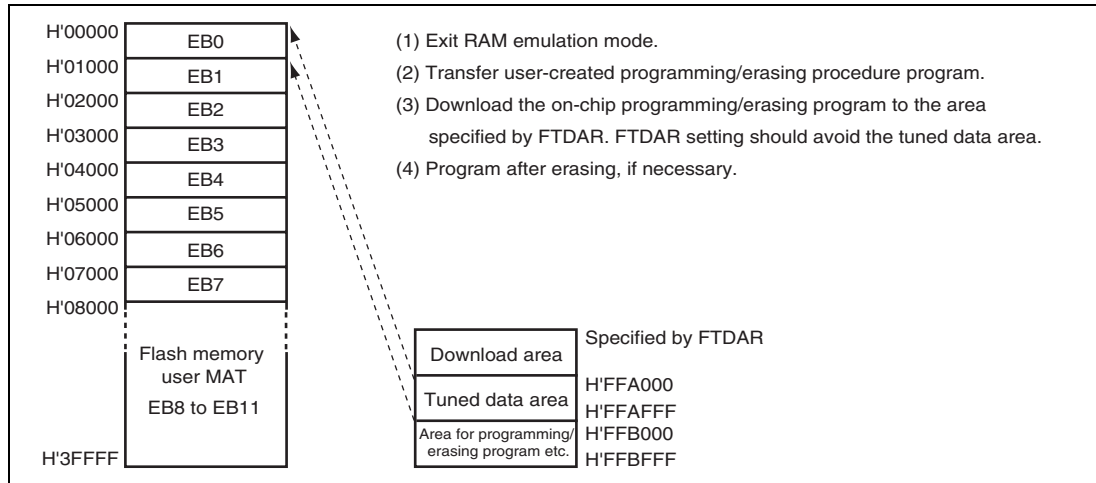
The flash memory area that can be emulated is the one area selected by bits RAM2 to RAM0 in RAMER from among the eight blocks, EB0 to EB7, of the user MAT.

To overlay a part of the on-chip RAM with block EB0 for realtime emulation, set the RAMS bit in RAMER to 1 and bits RAM2 to RAM0 to B'000.

For programming/erasing the user MAT, the procedure programs including a download program of the on-chip program must be executed. At this time, the download area should be specified so that the overlaid RAM area is not overwritten by downloading the on-chip program. Since the area in which the tuned data is stored is overlaid with the download area when FTDAR = H'01, the tuned data must be saved in an unused area beforehand.



Figure 22.19 shows an example of the procedure to program the tuned data in block EB0 of the user MAT.



**Figure 22.19 Programming Tuned Data**

1. After tuning program data is completed, clear the RAMS bit in RAMER to 0 to cancel the overlaid RAM.
2. Transfer the user-created procedure program to the on-chip RAM.
3. Start the procedure program and download the on-chip program to the on-chip RAM. The start address of the download destination should be specified by FTDAR so that the tuned data area does not overlay the download area.
4. When block EB0 of the user MAT has not been erased, the programming program must be downloaded after block EB0 is erased. Specify the tuned data saved in the FMPAR and FMPDR parameters and then execute programming.

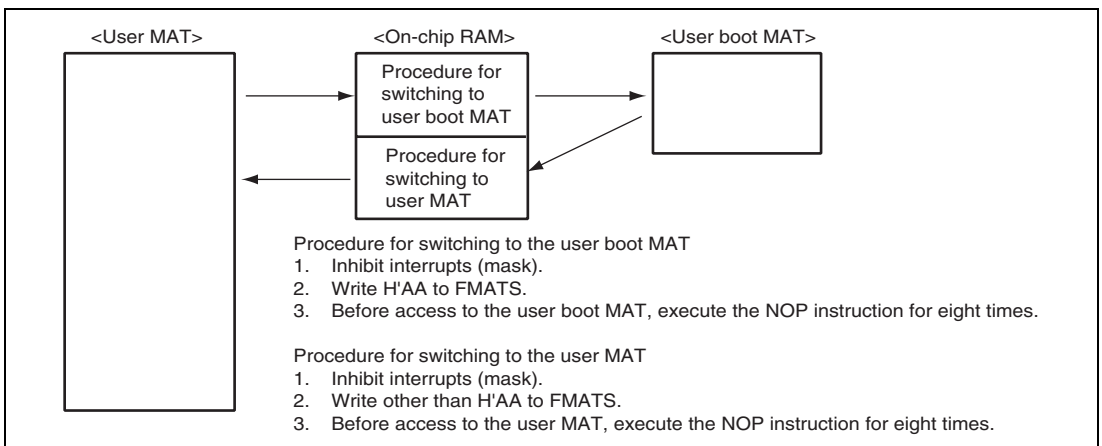
**Note:** Setting the RAMS bit to 1 makes all the blocks of the user MAT enter the programming/erasing protection state (emulation protection state) regardless of the setting of the RAM2 to RAM0 bits. Under this condition, the on-chip program cannot be downloaded. When data is to be actually programmed and erased, clear the RAMS bit to 0.

## 22.11 Switching between User MAT and User Boot MAT

It is possible to switch between the user MAT and user boot MAT. However, the following procedure is required because the start addresses of these MATs are allocated to the same address.

Switching to the user boot MAT disables programming and erasing. Programming of the user boot MAT should take place in boot mode or programmer mode.

1. Memory MAT switching by FMATS should always be executed from the on-chip RAM.
2. When accessing the memory MAT immediately after switching the memory MATs by FMATS from the on-chip RAM, similarly execute the NOP instruction in the on-chip RAM for eight times (this prevents access to the flash memory during memory MAT switching).
3. If an interrupt request has occurred during memory MAT switching, there is no guarantee of which memory MAT is accessed. Always mask the maskable interrupts before switching memory MATs. In addition, configure the system so that NMI interrupts do not occur during memory MAT switching.
4. After the memory MATs have been switched, take care because the interrupt vector table will also have been switched. If interrupt processing is to be the same before and after memory MAT switching, transfer the interrupt processing routines to the on-chip RAM and specify VBR to place the interrupt vector table in the on-chip RAM.
5. The size of the user MAT is different from that of the user boot MAT. Addresses which exceed the size of the 16-Kbyte user boot MAT should not be accessed. If an attempt is made, data is read as an undefined value.



**Figure 22.20 Switching between User MAT and User Boot MAT**

## 22.12 Programmer Mode

Along with its on-board programming mode, this LSI also has a programmer mode as a further mode for the writing and erasing of programs and data. In programmer mode, a general-purpose PROM programmer that supports the device types shown in table 22.14 can be used to write programs to the on-chip ROM without any limitation.

**Table 22.14 Device Types Supported in Programmer Mode**

Target Memory MAT	ROM Size	Device Type
User MAT	256 Kbytes	FZTAT256V3A
User boot MAT	16 Kbytes	FZTATUSBT16V3A

## 22.13 Standard Serial Communication Interface Specifications for Boot Mode

The boot program initiated in boot mode performs serial communication using the host and on-chip SCI\_4. The serial communication interface specifications are shown below.

The boot program has three states.

### 1. Bit-rate-adjustment state

In this state, the boot program adjusts the bit rate to achieve serial communication with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, the program enters the inquiry/selection state.

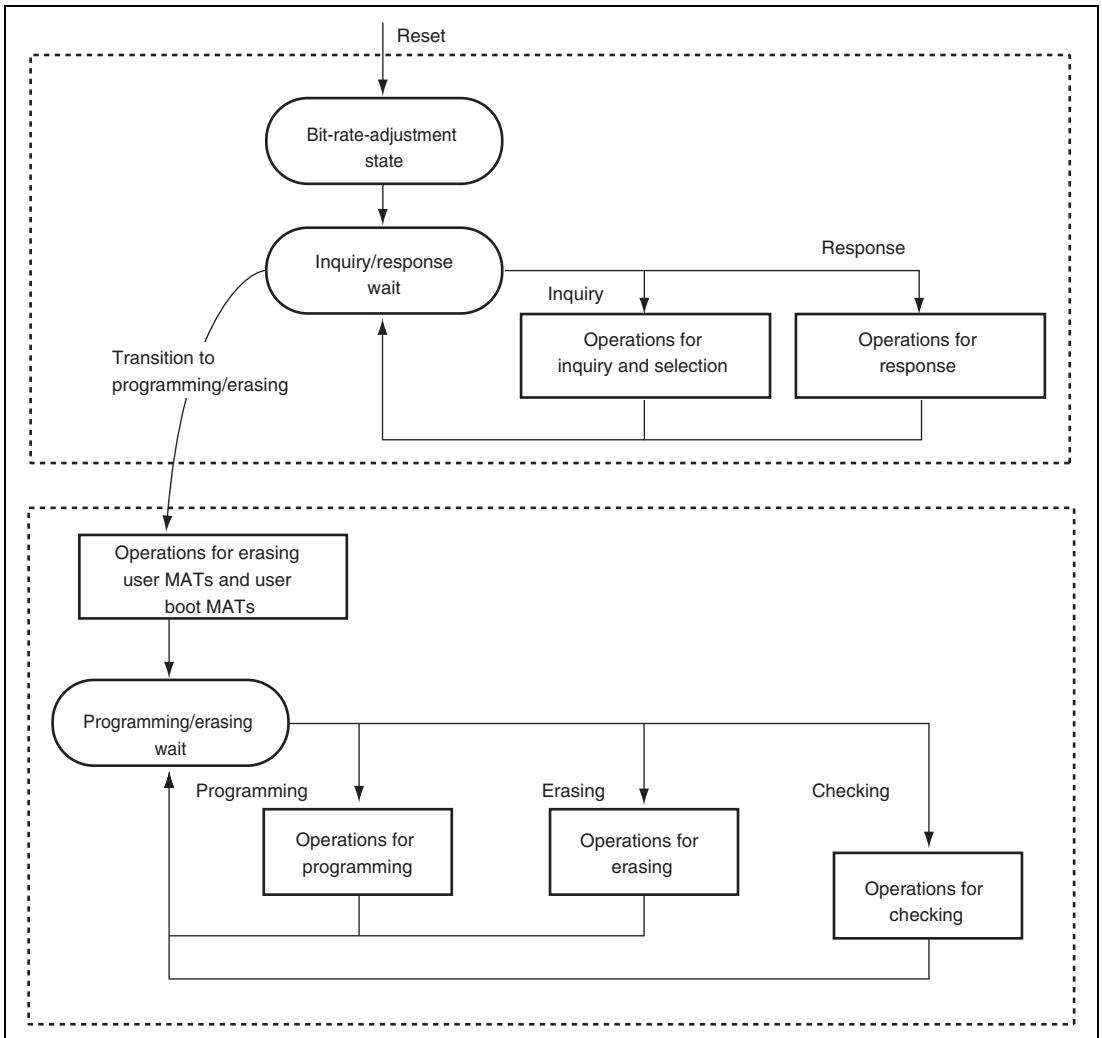
### 2. Inquiry/selection state

In this state, the boot program responds to inquiry commands from the host. The device name, clock mode, and bit rate are selected. After selection of these settings, the program is made to enter the programming/erasing state by the command for a transition to the programming/erasing state. The program transfers the libraries required for erasure to the on-chip RAM and erases the user MATs and user boot MATs before the transition.

### 3. Programming/erasing state

Programming and erasure by the boot program take place in this state. The boot program is made to transfer the programming/erasing programs to the on-chip RAM by commands from the host. Sum checks and blank checks are executed by sending these commands from the host.

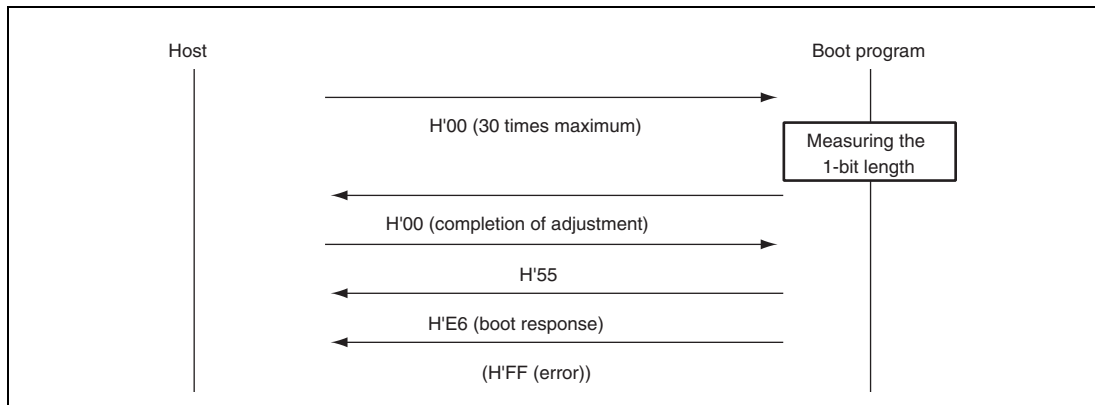
These boot program states are shown in figure 22.21.



**Figure 22.21 Boot Program States**

## (1) Bit-Rate-Adjustment State

The bit rate is calculated by measuring the period of transfer of a low-level byte (H'00) from the host. The bit rate can be changed by the command for a new bit rate selection. After the bit rate has been adjusted, the boot program enters the inquiry and selection state. The bit-rate-adjustment sequence is shown in figure 22.22.



**Figure 22.22 Bit-Rate-Adjustment Sequence**

## (2) Communications Protocol

After adjustment of the bit rate, the protocol for serial communications between the host and the boot program is as shown below.

### 1. One-byte commands and one-byte responses

These one-byte commands and one-byte responses consist of the inquiries and the ACK for successful completion.

### 2. n-byte commands or n-byte responses

These commands and responses are comprised of n bytes of data. These are selections and responses to inquiries.

The program data size is not included under this heading because it is determined in another command.

### 3. Error response

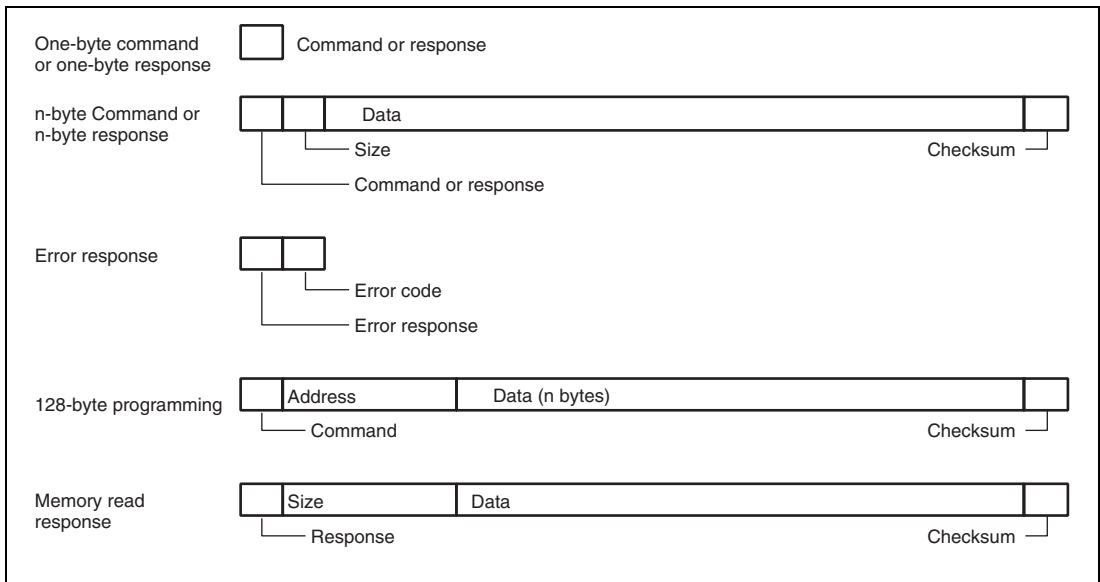
The error response is a response to inquiries. It consists of an error response and an error code and comes two bytes.

### 4. Programming of 128 bytes

The size is not specified in commands. The size of n is indicated in response to the programming unit inquiry.

## 5. Memory read response

This response consists of four bytes of data.



**Figure 22.23 Communication Protocol Format**

- **Command (one byte):** Commands including inquiries, selection, programming, erasing, and checking
- **Response (one byte):** Response to an inquiry
- **Size (one byte):** The amount of data for transmission excluding the command, amount of data, and checksum
- **Checksum (one byte):** The checksum is calculated so that the total of all values from the command byte to the SUM byte becomes H'00.
- **Data (n bytes):** Detailed data of a command or response
- **Error response (one byte):** Error response to a command
- **Error code (one byte):** Type of the error
- **Address (four bytes):** Address for programming
- **Data (n bytes):** Data to be programmed (the size is indicated in the response to the programming unit inquiry.)
- **Size (four bytes):** Four-byte response to a memory read

### (3) Inquiry and Selection States

The boot program returns information from the flash memory in response to the host's inquiry commands and sets the device code, clock mode, and bit rate in response to the host's selection command.

Table 22.15 lists the inquiry and selection commands.

**Table 22.15 Inquiry and Selection Commands**

Command	Command Name	Description
H'20	Supported device inquiry	Inquiry regarding device codes
H'10	Device selection	Selection of device code
H'21	Clock mode inquiry	Inquiry regarding numbers of clock modes and values of each mode
H'11	Clock mode selection	Indication of the selected clock mode
H'22	Multiplication ratio inquiry	Inquiry regarding the number of frequency-multiplied clock types, the number of multiplication ratios, and the values of each multiple
H'23	Operating clock frequency inquiry	Inquiry regarding the maximum and minimum values of the main clock and peripheral clocks
H'24	User boot MAT information inquiry	Inquiry regarding the number of user boot MATs and the start and last addresses of each MAT
H'25	User MAT information inquiry	Inquiry regarding the a number of user MATs and the start and last addresses of each MAT
H'26	Block for erasing information Inquiry	Inquiry regarding the number of blocks and the start and last addresses of each block
H'27	Programming unit inquiry	Inquiry regarding the unit of program data
H'3F	New bit rate selection	Selection of new bit rate
H'40	Transition to programming/erasing state	Erasing of user MAT and user boot MAT, and entry to programming/erasing state
H'4F	Boot program status inquiry	Inquiry into the operated status of the boot program

The selection commands, which are device selection (H'10), clock mode selection (H'11), and new bit rate selection (H'3F), should be sent from the host in that order. When two or more selection commands are sent at once, the last command will be valid.

All of these commands, except for the boot program status inquiry command (H'4F), will be valid until the boot program receives the programming/erasing transition (H'40). The host can choose the needed commands and make inquiries while the above commands are being transmitted. H'4F is valid even after the boot program has received H'40.

### (a) Supported Device Inquiry

The boot program will return the device codes of supported devices and the product code in response to the supported device inquiry.

Command 

H'20
------

- Command, H'20, (one byte): Inquiry regarding supported devices

Response	H'30	Size	Number of devices	
	Number of characters	Device code		Product name
	...			
	SUM			

- Response, H'30, (one byte): Response to the supported device inquiry
- Size (one byte): Number of bytes to be transmitted, excluding the command, size, and checksum, that is, the amount of data contributes by the number of devices, characters, device codes and product names
- Number of devices (one byte): The number of device types supported by the boot program
- Number of characters (one byte): The number of characters in the device codes and boot program's name
- Device code (four bytes): ASCII code of the supporting product
- Product name (n bytes): Type name of the boot program in ASCII-coded characters
- SUM (one byte): Checksum

The checksum is calculated so that the total number of all values from the command byte to the SUM byte becomes H'00.



**(b) Device Selection**

The boot program will set the supported device to the specified device code. The program will return the selected device code in response to the inquiry after this setting has been made.

Command	H'10	Size	Device code	SUM
---------	------	------	-------------	-----

- Command, H'10, (one byte): Device selection
- Size (one byte): Amount of device-code data  
This is fixed at 4.
- Device code (four bytes): Device code (ASCII code) returned in response to the supported device inquiry
- SUM (one byte): Checksum

Response	H'06
----------	------

- Response, H'06, (one byte): Response to the device selection command  
ACK will be returned when the device code matches.

Error response	H'90	ERROR
----------------	------	-------

- Error response, H'90, (one byte): Error response to the device selection command  
ERROR : (one byte): Error code  
H'11: Sum check error  
H'21: Device code error, that is, the device code does not match

**(c) Clock Mode Inquiry**

The boot program will return the supported clock modes in response to the clock mode inquiry.

Command	H'21
---------	------

- Command, H'21, (one byte): Inquiry regarding clock mode

Response	H'31	Size	Mode	...	SUM
----------	------	------	------	-----	-----

- Response, H'31, (one byte): Response to the clock-mode inquiry
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): Values of the supported clock modes (i.e. H'01 means clock mode 1.)
- SUM (one byte): Checksum

**(d) Clock Mode Selection**

The boot program will set the specified clock mode. The program will return the selected clock-mode information after this setting has been made.

The clock-mode selection command should be sent after the device-selection commands.

Command 

H'11	Size	Mode	SUM
------	------	------	-----

- Command, H'11, (one byte): Selection of clock mode
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): A clock mode returned in reply to the supported clock mode inquiry.
- SUM (one byte): Checksum

Response 

H'06
------

- Response, H'06, (one byte): Response to the clock mode selection command  
ACK will be returned when the clock mode matches.

Error Response 

H'91	ERROR
------	-------

- Error response, H'91, (one byte): Error response to the clock mode selection command
- ERROR : (one byte): Error code  
H'11: Checksum error  
H'22: Clock mode error, that is, the clock mode does not match.

Even if the clock mode numbers are H'00 and H'01 by a clock mode inquiry, the clock mode must be selected using these respective values.

**(e) Multiplication Ratio Inquiry**

The boot program will return the supported multiplication and division ratios.

Command H'22

- Command, H'22, (one byte): Inquiry regarding multiplication ratio

Response	H'32	Size	Number of multipliable operating clocks			
	Number of multiplication ratios	Multiplication ratio	...			
	...					
	SUM					

- Response, H'32, (one byte): Response to the multiplication ratio inquiry
- Size (one byte): The amount of data that represents the number of multipliable operating clocks and multiplication ratios and the multiplication ratios
- Number of multipliable operating clocks (one byte): The number of clocks that can be selected for multiplication (e.g. if the main and peripheral clock frequencies can be multiplied, the number of multipliable operating clocks will be H'02.)
- Number of multiplication ratios (one byte): The number of multiplication ratios for each type (e.g. the number of multiplication ratios to which the main clock can be set and the peripheral clock can be set.)
- Multiplication ratio (one byte)
 

Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H'04.)

Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H'FE.  $H'FE = D'-2$ )

The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are multipliable operating clocks.
- SUM (one byte): Checksum

**(f) Operating Clock Frequency Inquiry**

The boot program will return the number of operating clock frequencies, and the maximum and minimum values.

Command 

H'23
------

- Command, H'23, (one byte): Inquiry regarding operating clock frequencies

Response	H'33	Size	Number of operating clock frequencies
	Minimum value of operating clock frequency		Maximum value of operating clock frequency
	...		
	SUM		

- Response, H'33, (one byte): Response to operating clock frequency inquiry
- Size (one byte): The number of bytes that represents the minimum values, maximum values, and the number of frequencies.
- Number of operating clock frequencies (one byte): The number of supported operating clock frequency types  
(e.g. when there are two operating clock frequency types, which are the main and peripheral clocks, the number of types will be H'02.)
- Minimum value of operating clock frequency (two bytes): The minimum value of the multiplied or divided clock frequency.  
The minimum and maximum values of the operating clock frequency represent the values in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Maximum value (two bytes): Maximum value among the multiplied or divided clock frequencies.  
There are as many pairs of minimum and maximum values as there are operating clock frequencies.
- SUM (one byte): Checksum

**(g) User Boot MAT Information Inquiry**

The boot program will return the number of user boot MATs and their addresses.

Command 

H'24
------

- Command, H'24, (one byte): Inquiry regarding user boot MAT information

Response

H'34	Size	Number of areas	
Area-start address		Area-last address	
...			
SUM			

- Response, H'34, (one byte): Response to user boot MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start addresses, and area-last address
- Number of Areas (one byte): The number of consecutive user boot MAT areas  
When user boot MAT areas are consecutive, the number of areas returned is H'01.
- Area-start address (four byte): Start address of the area
- Area-last address (four byte): Last address of the area  
There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

**(h) User MAT Information Inquiry**

The boot program will return the number of user MATs and their addresses.

Command 

H'25
------

- Command, H'25, (one byte): Inquiry regarding user MAT information

Response

H'35	Size	Number of areas	
Start address area		Last address area	
...			
SUM			

- Response, H'35, (one byte): Response to the user MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start address and area-last address
- Number of areas (one byte): The number of consecutive user MAT areas  
When the user MAT areas are consecutive, the number of areas is H'01.
- Area-start address (four bytes): Start address of the area

- Area-last address (four bytes): Last address of the area  
There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

### (i) Erased Block Information Inquiry

The boot program will return the number of erased blocks and their addresses.

Command 

H'26
------

- Command, H'26, (two bytes): Inquiry regarding erased block information

Response

H'36	Size	Number of blocks	
Block start address		Block last address	
...			
SUM			

- Response, H'36, (one byte): Response to the number of erased blocks and addresses
- Size (three bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
- Number of blocks (one byte): The number of erased blocks
- Block start address (four bytes): Start address of a block
- Block last Address (four bytes): Last address of a block  
There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

### (j) Programming Unit Inquiry

The boot program will return the programming unit used to program data.

Command 

H'27
------

- Command, H'27, (one byte): Inquiry regarding programming unit

Response

H'37	Size	Programming unit	SUM
------	------	------------------	-----

- Response, H'37, (one byte): Response to programming unit inquiry
- Size (one byte): The number of bytes that indicate the programming unit, which is fixed to 2
- Programming unit (two bytes): A unit for programming  
This is the unit for reception of programming.
- SUM (one byte): Checksum

**(k) New Bit-Rate Selection**

The boot program will set a new bit rate and return the new bit rate.

This selection should be sent after sending the clock mode selection command.

Command	H'3F	Size	Bit rate	Input frequency
	Number of multipliable operating clocks	Multiplication ratio 1	Multiplication ratio 2	
	SUM			

- Command, H'3F, (one byte): Selection of new bit rate
- Size (one byte): The number of bytes that represents the bit rate, input frequency, number of multipliable operating clocks, and multiplication ratios
- Bit rate (two bytes): New bit rate  
One hundredth of the value (e.g. when the value is 19200 bps, it will be 192, which is H'00C0.)
- Input frequency (two bytes): Frequency of the clock input to the boot program  
This is valid to the hundredths place and represents the value in MHz multiplied by 100. (E.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Number of multipliable operating clocks (one byte): The number of operating clocks in the device that can be selected for multiplication.
- Multiplication ratio 1 (one byte): The value of multiplication or division ratios for the main operating frequency  
Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)  
Division ratio: The inverse of the division ratio, as a negative number (e.g. when the clock frequency is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- Multiplication ratio 2 (one byte): The value of multiplication or division ratios for the peripheral frequency  
Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)  
(Division ratio: The inverse of the division ratio, as a negative number (E.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- SUM (one byte): Checksum

Response 

H'06
------

- Response, H'06, (one byte): Response to selection of a new bit rate  
When it is possible to set the bit rate, the response will be ACK.

Error Response	H'BF	ERROR
----------------	------	-------

- Error response, H'BF, (one byte): Error response to selection of new bit rate
- ERROR: (one byte): Error code
  - H'11: Sum checking error
  - H'24: Bit-rate selection error
    - The rate is not available.
  - H'25: Error in input frequency
    - This input frequency is not within the specified range.
  - H'26: Multiplication-ratio error
    - The ratio does not match an available ratio.
  - H'27: Operating frequency error
    - The frequency is not within the specified range.

#### (4) Receive Data Check

The methods for checking of receive data are listed below.

##### 1. Input frequency

The received value of the input frequency is checked to ensure that it is within the range of minimum to maximum frequencies which matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

##### 2. Multiplication ratio

The received value of the multiplication ratio or division ratio is checked to ensure that it matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

##### 3. Operating frequency error

Operating frequency is calculated from the received value of the input frequency and the multiplication or division ratio. The input frequency is input to the LSI and the LSI is operated at the operating frequency. The expression is given below.

Operating frequency = Input frequency × Multiplication ratio, or

Operating frequency = Input frequency ÷ Division ratio

The calculated operating frequency should be checked to ensure that it is within the range of minimum to maximum frequencies which are available with the clock modes of the specified device. When it is out of this range, an operating frequency error is generated.



#### 4. Bit rate

To facilitate error checking, the value (n) of clock select (CKS) in the serial mode register (SMR), and the value (N) in the bit rate register (BRR), which are found from the peripheral operating clock frequency ( $\phi$ ) and bit rate (B), are used to calculate the error rate to ensure that it is less than 4%. If the error is more than 4%, a bit rate error is generated. The error is calculated using the following expression:

$$\text{Error (\%)} = \left\{ \left[ \frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{(2 \times n - 1)}} \right] - 1 \right\} \times 100$$

When the new bit rate is selectable, the rate will be set in the register after sending ACK in response. The host will send an ACK with the new bit rate for confirmation and the boot program will response with that rate.

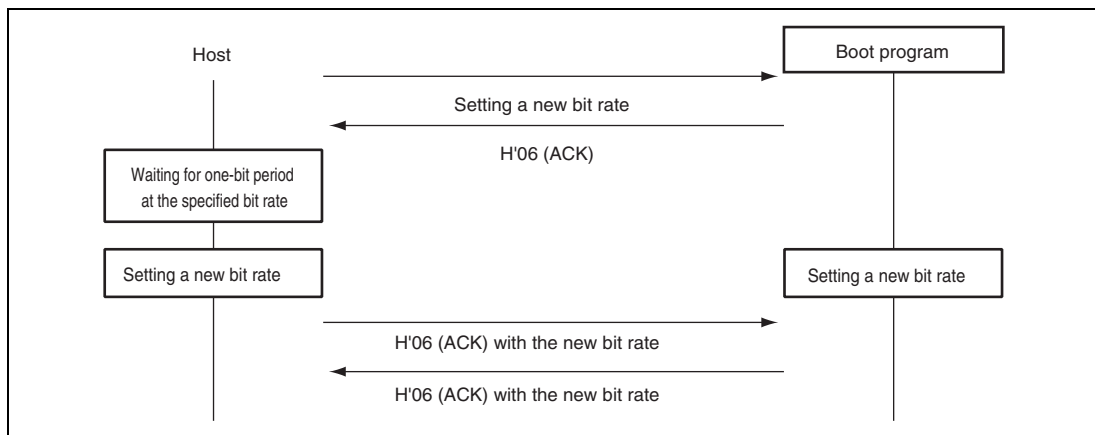
Confirmation H'06

- Confirmation, H'06, (one byte): Confirmation of a new bit rate

Response H'06

- Response, H'06, (one byte): Response to confirmation of a new bit rate

The sequence of new bit-rate selection is shown in figure 22.24.



**Figure 22.24 New Bit-Rate Selection Sequence**

## (5) Transition to Programming/Erasing State

The boot program will transfer the erasing program, and erase the user MATs and user boot MATs in that order. On completion of this erasure, ACK will be returned and will enter the programming/erasing state.

The host should select the device code, clock mode, and new bit rate with device selection, clock-mode selection, and new bit-rate selection commands, and then send the command for the transition to programming/erasing state. These procedures should be carried out before sending of the programming selection command or program data.

Command 

H'40
------

- Command, H'40, (one byte): Transition to programming/erasing state

Response 

H'06
------

- Response, H'06, (one byte): Response to transition to programming/erasing state  
The boot program will send ACK when the user MAT and user boot MAT have been erased by the transferred erasing program.

Error Response 

H'C0	H'51
------	------

- Error response, H'C0, (one byte): Error response for user boot MAT blank check
- Error code, H'51, (one byte): Erasing error  
An error occurred and erasure was not completed.

## (6) Command Error

A command error will occur when a command is undefined, the order of commands is incorrect, or a command is unacceptable. Issuing a clock-mode selection command before a device selection or an inquiry command after the transition to programming/erasing state command, are examples.

Error Response 

H'80	H'xx
------	------

- Error response, H'80, (one byte): Command error
- Command, H'xx, (one byte): Received command

## (7) Command Order

The order for commands in the inquiry selection state is shown below.

1. A supported device inquiry (H'20) should be made to inquire about the supported devices.
2. The device should be selected from among those described by the returned information and set with a device-selection (H'10) command.
3. A clock-mode inquiry (H'21) should be made to inquire about the supported clock modes.
4. The clock mode should be selected from among those described by the returned information and set.
5. After selection of the device and clock mode, inquiries for other required information should be made, such as the multiplication-ratio inquiry (H'22) or operating frequency inquiry (H'23), which are needed for a new bit-rate selection.
6. A new bit rate should be selected with the new bit-rate selection (H'3F) command, according to the returned information on multiplication ratios and operating frequencies.
7. After selection of the device and clock mode, the information of the user boot MAT and user MAT should be made to inquire about the user boot MATs information inquiry (H'24), user MATs information inquiry (H'25), erased block information inquiry (H'26), and programming unit inquiry (H'27).
8. After making inquiries and selecting a new bit rate, issue the transition to programming/erasing state command (H'40). The boot program will then enter the programming/erasing state.

**(8) Programming/Erasing State**

A programming selection command makes the boot program select the programming method, a 128-byte programming command makes it program the memory with data, and an erasing selection command and block erasing command make it erase the block. Table 22.16 lists the programming/erasing commands.

**Table 22.16 Programming/Erasing Commands**

<b>Command</b>	<b>Command Name</b>	<b>Description</b>
H'42	User boot MAT programming selection	Transfers the user boot MAT programming program
H'43	User MAT programming selection	Transfers the user MAT programming program
H'50	128-byte programming	Programs 128 bytes of data
H'48	Erasing selection	Transfers the erasing program
H'58	Block erasing	Erases a block of data
H'52	Memory read	Reads the contents of memory
H'4A	User boot MAT sum check	Checks the checksum of the user boot MAT
H'4B	User MAT sum check	Checks the checksum of the user MAT
H'4C	User boot MAT blank check	Checks the blank data of the user boot MAT
H'4D	User MAT blank check	Checks the blank data of the user MAT
H'4F	Boot program status inquiry	Inquires into the boot program's status

- Programming

Programming is executed by the programming selection and 128-byte programming commands.

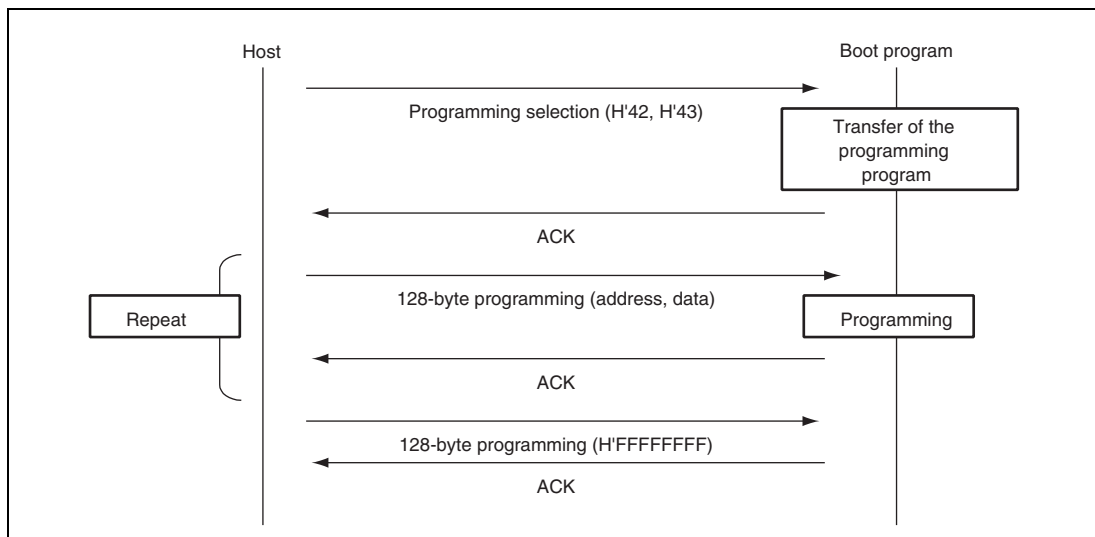
Firstly, the host should send the programming selection command and select the programming method and programming MATs. There are two programming selection commands, and selection is according to the area and method for programming.

1. User boot MAT programming selection
2. User MAT programming selection

After issuing the programming selection command, the host should send the 128-byte programming command. The 128-byte programming command that follows the selection command represents the data programmed according to the method specified by the selection command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H'FFFFFFFF as the address will stop the programming. On completion of programming, the boot program will wait for selection of programming or erasing.

Where the sequence of programming operations that is executed includes programming with another method or of another MAT, the procedure must be repeated from the programming selection command.

The sequence for the programming selection and 128-byte programming commands is shown in figure 22.25.



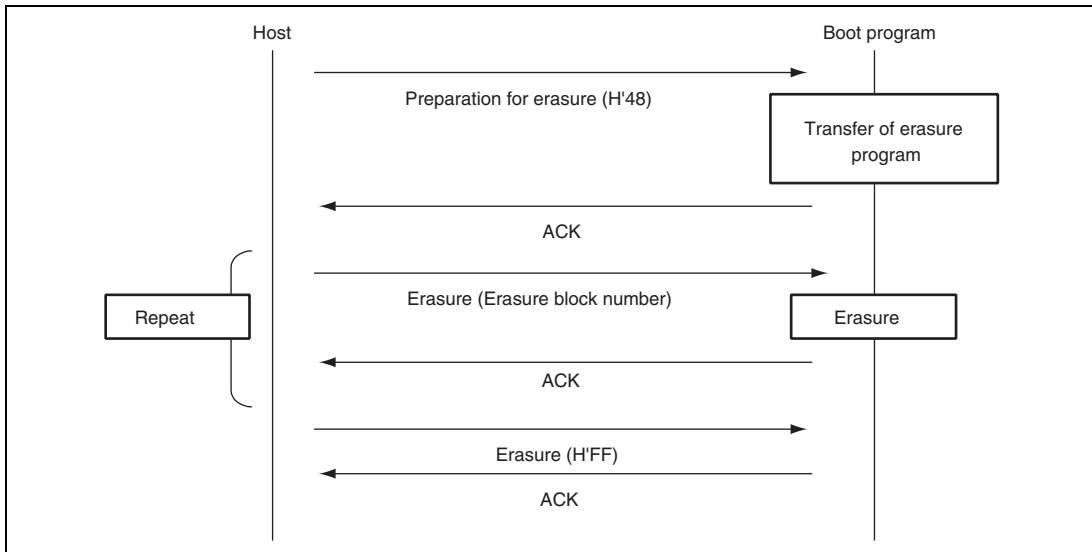
**Figure 22.25 Programming Sequence**

- Erasure

Erasure is executed by the erasure selection and block erasure commands.

Firstly, erasure is selected by the erasure selection command and the boot program then erases the specified block. The command should be repeatedly executed if two or more blocks are to be erased. Sending a block erasure command from the host with the block number H'FF will stop the erasure operating. On completion of erasing, the boot program will wait for selection of programming or erasing.

The sequence for the erasure selection and block erasure commands is shown in figure 22.26.



**Figure 22.26 Erasure Sequence**

**(a) User Boot MAT Programming Selection**

The boot program will transfer a programming program. The data is programmed to the user boot MATs by the transferred programming program.

Command 

H'42
------

- Command, H'42, (one byte): User boot-program programming selection

Response 

H'06
------

- Response, H'06, (one byte): Response to user boot-program programming selection  
When the programming program has been transferred, the boot program will return ACK.

Error Response 

H'C2	ERROR
------	-------

- Error response : H'C2 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code  
H'54: Selection processing error (transfer error occurs and processing is not completed)

**(b) User MAT Programming Selection**

The boot program will transfer a program for user MAT programming selection. The data is programmed to the user MATs by the transferred program for programming.

Command 

H'43
------

- Command, H'43, (one byte): User-program programming selection

Response 

H'06
------

- Response, H'06, (one byte): Response to user-program programming selection  
When the programming program has been transferred, the boot program will return ACK.

Error Response 

H'C3	ERROR
------	-------

- Error response : H'C3 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code  
H'54: Selection processing error (transfer error occurs and processing is not completed)

**(c) 128-Byte Programming**

The boot program will use the programming program transferred by the programming selection to program the user boot MATs or user MATs in response to 128-byte programming.

Command	H'50	Address					
	Data	...					
	...						
	SUM						

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): Start address for programming  
Multiple of the size specified in response to the programming unit inquiry (i.e. H'00, H'01, H'00, H'00 : H'01000000)
- Program data (128 bytes): Data to be programmed  
The size is specified in the response to the programming unit inquiry.
- SUM (one byte): Checksum

Response 

H'06
------

- Response, H'06, (one byte): Response to 128-byte programming  
On completion of programming, the boot program will return ACK.

Error Response 

H'D0	ERROR
------	-------

- Error response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code
  - H'11: Checksum Error
  - H'2A: Address error  
The address is not in the specified MAT.
  - H'53: Programming error  
A programming error has occurred and programming cannot be continued.

The specified address should match the unit for programming of data. For example, when the programming is in 128-byte units, the lower eight bits of the address should be H'00 or H'80. When there are less than 128 bytes of data to be programmed, the host should fill the rest with H'FF.

Sending the 128-byte programming command with the address of H'FFFFFFFF will stop the programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.



Command 

H'50	Address	SUM
------	---------	-----

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): End code is H'FF, H'FF, H'FF, H'FF.
- SUM (one byte): Checksum

Response 

H'06
------

- Response, H'06, (one byte): Response to 128-byte programming  
On completion of programming, the boot program will return ACK.

Error Response 

H'D0	ERROR
------	-------

- Error Response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code
  - H'11: Checksum error
  - H'53: Programming error

An error has occurred in programming and programming cannot be continued.

#### (d) Erasure Selection

The boot program will transfer the erasure program. User MAT data is erased by the transferred erasure program.

Command 

H'48
------

- Command, H'48, (one byte): Erasure selection

Response 

H'06
------

- Response, H'06, (one byte): Response for erasure selection  
After the erasure program has been transferred, the boot program will return ACK.

Error Response 

H'C8	ERROR
------	-------

- Error Response, H'C8, (one byte): Error response to erasure selection
- ERROR: (one byte): Error code
  - H'54: Selection processing error (transfer error occurs and processing is not completed)

**(e) Block Erasure**

The boot program will erase the contents of the specified block.

Command	H'58	Size	Block number	SUM
---------	------	------	--------------	-----

- Command, H'58, (one byte): Erasure
- Size (one byte): The number of bytes that represents the erase block number  
This is fixed to 1.
- Block number (one byte): Number of the block to be erased
- SUM (one byte): Checksum

Response	H'06
----------	------

- Response, H'06, (one byte): Response to Erasure  
After erasure has been completed, the boot program will return ACK.

Error Response	H'D8	ERROR
----------------	------	-------

- Error Response, H'D8, (one byte): Response to Erasure
- ERROR (one byte): Error code
  - H'11: Sum check error
  - H'29: Block number error  
Block number is incorrect.
  - H'51: Erasure error  
An error has occurred during erasure.

On receiving block number H'FF, the boot program will stop erasure and wait for a selection command.

Command	H'58	Size	Block number	SUM
---------	------	------	--------------	-----

- Command, H'58, (one byte): Erasure
- Size, (one byte): The number of bytes that represents the block number  
This is fixed to 1.
- Block number (one byte): H'FF  
Stop code for erasure
- SUM (one byte): Checksum

Response	H'06
----------	------

- Response, H'06, (one byte): Response to end of erasure (ACK)  
When erasure is to be performed after the block number H'FF has been sent, the procedure should be executed from the erasure selection command.

**(f) Memory Read**

The boot program will return the data in the specified address.

Command	H'52	Size	Area	Read address		
	Read size			SUM		

- Command: H'52 (1 byte): Memory read
- Size (1 byte): Amount of data that represents the area, read address, and read size (fixed at 9)
- Area (1 byte)
  - H'00: User boot MAT
  - H'01: User MAT

An address error occurs when the area setting is incorrect.

- Read address (4 bytes): Start address to be read from
- Read size (4 bytes): Size of data to be read
- SUM (1 byte): Checksum

Response	H'52	Read size						
	Data	...						
	SUM							

- Response: H'52 (1 byte): Response to memory read
- Read size (4 bytes): Size of data to be read
- Data (n bytes): Data for the read size from the read address
- SUM (1 byte): Checksum

Error Response	H'D2	ERROR
----------------	------	-------

- Error response: H'D2 (1 byte): Error response to memory read
- ERROR: (1 byte): Error code
  - H'11: Sum check error
  - H'2A: Address error
    - The read address is not in the MAT.
  - H'2B: Size error
    - The read size exceeds the MAT.

**(g) User-Boot Program Sum Check**

The boot program will return the byte-by-byte total of the contents of the bytes of the user-boot program, as a four-byte value.

Command 

H'4A
------

- Command, H'4A, (one byte): Sum check for user-boot program

Response 

H'5A	Size	Checksum of user boot program	SUM
------	------	-------------------------------	-----

- Response, H'5A, (one byte): Response to the sum check of user-boot program
- Size (one byte): The number of bytes that represents the checksum  
This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user boot MATs  
The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

**(h) User-Program Sum Check**

The boot program will return the byte-by-byte total of the contents of the bytes of the user program.

Command 

H'4B
------

- Command, H'4B, (one byte): Sum check for user program

Response 

H'5B	Size	Checksum of user program	SUM
------	------	--------------------------	-----

- Response, H'5B, (one byte): Response to the sum check of the user program
- Size (one byte): The number of bytes that represents the checksum  
This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user MATs  
The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

**(i) User Boot MAT Blank Check**

The boot program will check whether or not all user boot MATs are blank and return the result.

Command 

H'4C
------

- Command, H'4C, (one byte): Blank check for user boot MAT

Response 

H'06
------

- Response, H'06, (one byte): Response to the blank check of user boot MAT  
If all user MATs are blank (H'FF), the boot program will return ACK.

Error Response 

H'CC	H'52
------	------

- Error Response, H'CC, (one byte): Response to blank check for user boot MAT
- Error Code, H'52, (one byte): Erasure has not been completed.

**(j) User MAT Blank Check**

The boot program will check whether or not all user MATs are blank and return the result.

Command 

H'4D
------

- Command, H'4D, (one byte): Blank check for user MATs

Response 

H'06
------

- Response, H'06, (one byte): Response to the blank check for user MATs  
If the contents of all user MATs are blank (H'FF), the boot program will return ACK.

Error Response 

H'CD	H'52
------	------

- Error Response, H'CD, (one byte): Error response to the blank check of user MATs.
- Error code, H'52, (one byte): Erasure has not been completed.

**(k) Boot Program State Inquiry**

The boot program will return indications of its present state and error condition. This inquiry can be made in the inquiry/selection state or the programming/erasing state.

Command 

H'4F
------

- Command, H'4F, (one byte): Inquiry regarding boot program's state

Response 

H'5F	Size	Status	ERROR	SUM
------	------	--------	-------	-----

- Response, H'5F, (one byte): Response to boot program state inquiry
- Size (one byte): The number of bytes. This is fixed to 2.
- Status (one byte): State of the boot program
- ERROR (one byte): Error status
  - ERROR = 0 indicates normal operation.
  - ERROR = 1 indicates error has occurred.
- SUM (one byte): Sum check

**Table 22.17 Status Code**

<b>Code</b>	<b>Description</b>
H'11	Device selection wait
H'12	Clock mode selection wait
H'13	Bit rate selection wait
H'1F	Programming/erasing state transition wait (bit rate selection is completed)
H'31	Programming state for erasure
H'3F	Programming/erasing selection wait (erasure is completed)
H'4F	Program data receive wait
H'5F	Erase block specification wait (erasure is completed)

**Table 22.18 Error Code**

<b>Code</b>	<b>Description</b>
H'00	No error
H'11	Sum check error
H'12	Program size error
H'21	Device code mismatch error
H'22	Clock mode mismatch error
H'24	Bit rate selection error
H'25	Input frequency error
H'26	Multiplication ratio error
H'27	Operating frequency error
H'29	Block number error
H'2A	Address error
H'2B	Data length error
H'51	Erase error
H'52	Erase incomplete error
H'53	Programming error
H'54	Selection processing error
H'80	Command error
H'FF	Bit-rate-adjustment confirmation error

## 22.14 Usage Notes

1. The initial state of the product at its shipment is in the erased state. For the product whose revision of erasing is undefined, we recommend to execute automatic erasure for checking the initial state (erased state) and compensating.
2. For the PROM programmer suitable for programmer mode in this LSI and its program version, refer to the instruction manual of the socket adapter.
3. If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.
4. Use a PROM programmer that supports the device with 256-Kbyte on-chip flash memory and 3.3-V programming voltage. Use only the specified socket adapter.
5. Do not remove the chip from the PROM programmer nor input a reset signal during programming/erasure in which a high voltage is applied to the flash memory. Doing so may damage the flash memory permanently. If a reset is input accidentally, the reset must be released after the reset input period of at least 100 $\mu$ s.
6. The flash memory is not accessible until FKEY is cleared after programming/erasure starts. If the operating mode is changed and this LSI is restarted by a reset immediately after programming/erasure has finished, secure the reset input period (period of  $\overline{\text{RES}} = 0$ ) of at least 100 $\mu$ s. Transition to the reset state during programming/erasure is inhibited. If a reset is input accidentally, the reset must be released after the reset input period of at least 100 $\mu$ s.
7. At powering on or off the Vcc power supply, fix the  $\overline{\text{RES}}$  pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.
8. In on-board programming mode or programmer mode, programming of the 128-byte programming-unit block must be performed only once. Perform programming in the state where the programming-unit block is fully erased.
9. When the chip is to be reprogrammed with the programmer after execution of programming or erasure in on-board programming mode, it is recommended that automatic programming is performed after execution of automatic erasure.
10. To program the flash memory, the program data and program must be allocated to addresses which are higher than those of the external interrupt vector table and H'FF must be written to all the system reserved areas in the exception handling vector table.
11. The programming program that includes the initialization routine and the erasing program that includes the initialization routine are each 4 Kbytes or less. Accordingly, when the CPU clock frequency is 35 MHz, the download for each program takes approximately 60  $\mu$ s at the maximum.



12. A programming/erasing program for the flash memory used in a conventional F-ZTAT H8, H8S microcomputer which does not support download of the on-chip program by setting the SCO bit in FCCS to 1 cannot run in this LSI. Be sure to download the on-chip program to execute programming/erasure of the flash memory in this F-ZTAT H8SX microcomputer.
13. Unlike a conventional F-ZTAT H8 or H8S microcomputers, measures against a program crash are not taken by WDT while programming/erasing and downloading a programming/erasing program. When needed, measures should be taken by user. A periodic interrupt generated by the WDT can be used as the measures, as an example. In this case, the interrupt generation period should take into consideration time to program/erase the flash memory.
14. When downloading the programming/erasing program, do not clear the SCO bit in FCCS to 0 immediately after setting it to 1. Otherwise, download cannot be performed normally. Immediately after executing the instruction to set the SCO bit to 1, dummy read of the FCCS must be executed twice.
15. The contents of general registers ER0 and ER1 are not saved during download of an on-chip program, initialization, programming, end of the programming, or erasure. When needed, save the general registers before a download request or before execution of initialization, programming, or erasure using the procedure program.

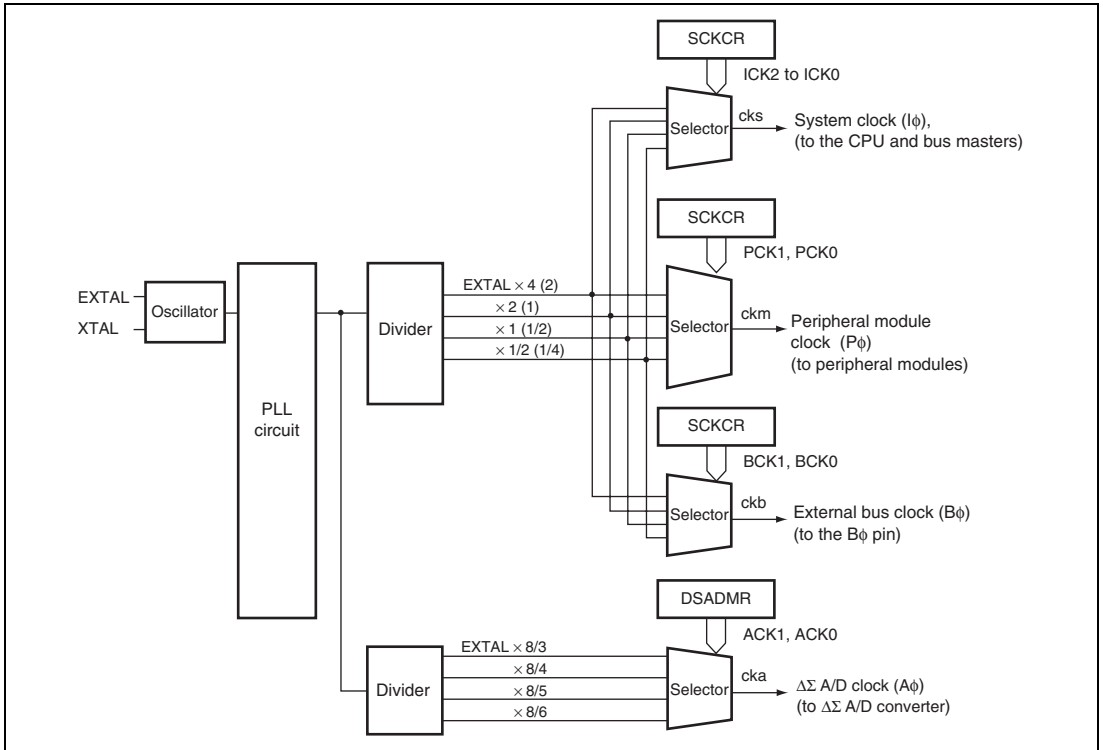


## Section 23 Clock Pulse Generator

This LSI has an on-chip clock pulse generator (CPG) that generates the system clock ( $I\phi$ ), peripheral module clock ( $P\phi$ ), external bus clock ( $B\phi$ ), and  $\Delta\Sigma$  A/D converter clock ( $A\phi$ ). The clock pulse generator comprises an oscillator, frequency dividers, PLL (phase-locked loop) circuit, and selectors. Figure 23.1 is a block diagram of the clock pulse generator.

This LSI supports four clocks: a system clock supplied to the CPU and bus masters, a peripheral module clock supplied to the peripheral modules, an external bus clock supplied to the external bus, and a  $\Delta\Sigma$  A/D clock supplied to the  $\Delta\Sigma$  A/D converter. The clock frequencies can be changed by the frequency dividers, PLL circuit, and selectors. The frequencies of the system clock, peripheral module clock and external bus clock are changed by setting the system clock control register (SCKCR) by software. The  $\Delta\Sigma$  A/D converter clock is generated from the oscillator output multiplied by 8, the frequency of which can be changed by setting the  $\Delta\Sigma$  A/D mode register (DSADMR) by software.

Frequencies of the peripheral module clock, the external bus clock, and the system clock can be set independently, although the peripheral module clock and the external bus clock only operate at frequencies lower than the system clock frequency. Since the  $\Delta\Sigma$  A/D converter has been designed to deliver the maximum precision at approximately 25 MHz, the division ratio for the  $\Delta\Sigma$  A/D converter clock should be set in DSADMR so as to make the frequency near 25 MHz.



**Figure 23.1 Block Diagram of Clock Pulse Generator**

**Table 23.1 Selection for Clock Pulse Generator**

EXTAL Input Clock Frequency	$I\phi/P\phi/B\phi$	$A\phi$ ( $\Delta\Sigma$ A/D Converter)
8 MHz to 18 MHz	EXTAL $\times 4$ , $\times 2$ , $\times 1$ , $\times 1/2$	[EXTAL $\times 8$ ] $\times 1/3$ , $\times 1/4$ , $\times 1/5$ , $\times 1/6$ (Frequency near 25 MHz is recommended)

## 23.1 Register Description

The clock pulse generator has the following register.

- System clock control register (SCKCR)
- $\Delta\Sigma$  A/D mode register (DSADMR)

### 23.1.1 System Clock Control Register (SCKCR)

SCKCR controls B $\phi$  clock output and frequencies of the system, peripheral module, and external bus clocks.

Bit	15	14	13	12	11	10	9	8
Bit Name	PSTOP1	—	POSEL1	—	—	ICK2	ICK1	ICK0
Initial Value	0	0	0	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0
Initial Value	0	0	1	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	PSTOP1	0	R/W	B $\phi$ Output Enable Enables the B $\phi$ output on PA7. <ul style="list-style-type: none"> <li>• Normal operation</li> </ul> 0: B $\phi$ output 1: Fixed high
14	—	0	R/W	Reserved This bit enables read/write operations, but the write value should always be 0.
13	POSEL1	0	R/W	Clock Output Select 1 Selects the clock signal to be output from PA7. 0: External bus clock (B $\phi$ ) 1: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description	
12, 11	—	All 0	R/W	Reserved These bits enable read/write operations, but the write value should always be 0.	
10	ICK2	0	R/W	System Clock ( $I\phi$ ) Select	
9	ICK1	1	R/W	These bits select the frequency of the system clock provided to the CPU, DTC, and DMAC. The ratio to the input clock is as follows: 000: $\times 4$ 001: $\times 2$ 010: $\times 1$ 011: $\times 1/2$ 100: Setting prohibited 101: Setting prohibited 110: Setting prohibited 111: Setting prohibited The frequencies of the peripheral module clock and external bus clock change to the same frequency as the system clock if the frequency of the system clock is lower than that of the two clocks.	
8	ICK0	0	R/W		
7	—	0	R/W		Reserved This bit enables read/write operations, but the write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description	
6	PCK2	0	R/W	Peripheral Module Clock (P $\phi$ ) Select	
5	PCK1	1	R/W	<p>These bits select the frequency of the peripheral module clock. The ratio to the input clock is as follows:</p> <p>000: <math>\times 4</math></p> <p>001: <math>\times 2</math></p> <p>010: <math>\times 1</math></p> <p>011: <math>\times 1/2</math></p> <p>100: Setting prohibited</p> <p>101: Setting prohibited</p> <p>110: Setting prohibited</p> <p>111: Setting prohibited</p> <p>The frequency of the peripheral module clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the peripheral module clock higher than that of the system clock, the clocks will have the same frequency in reality.</p>	
4	PCK0	0	R/W		
3	—	0	R/W		<p>Reserved</p> <p>This bit enables read/write operations, but the write value should always be 0.</p>
2	BCK2	0	R/W		External Bus Clock (B $\phi$ ) Select
1	BCK1	1	R/W		<p>These bits select the frequency of the external bus clock. The ratio to the input clock is as follows:</p> <p>000: <math>\times 4</math></p> <p>001: <math>\times 2</math></p> <p>010: <math>\times 1</math></p> <p>011: <math>\times 1/2</math></p> <p>100: Setting prohibited</p> <p>101: Setting prohibited</p> <p>110: Setting prohibited</p> <p>111: Setting prohibited</p> <p>The frequency of the external bus clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the external bus clock higher than that of the system clock, the clocks will have the same frequency in reality.</p>
0	BCK0	0	R/W		

[Legend]

X: Don't care

### 23.1.2 $\Delta\Sigma$ A/D Mode Register (DSADMR)

DSADMR sets the control for the bias circuit and the clock selection for the  $\Delta\Sigma$  A/D converter. Read operations are always enabled, but write operations should be performed when the  $\Delta\Sigma$  A/D module is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	BIASE	—	—	—	—	ACK2	ACK1	ACK0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	BIASE	0	R/W	Bias Circuit Control Sets whether the bias circuit is to be operated or stopped. 0: Stops the bias circuit. 1: Operates the bias circuit.
6 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
2	ACK2	0	R/W	$\Delta\Sigma$ A/D Converter Frequency Division Clock Select
1	ACK1	0	R/W	These bits select the frequency of the $\Delta\Sigma$ A/D clock ( $A\phi$ ). The ratio to the input clock is as follows. In setting, the value of $A\phi$ should be in the neighborhood of 25 MHz. 000: $\times 1/6$ 001: $\times 1/5$ 010: $\times 1/4$ 011: $\times 1/3$ 1xx: Setting prohibited
0	ACK0	0	R/W	

#### [Legend]

X: Don't care



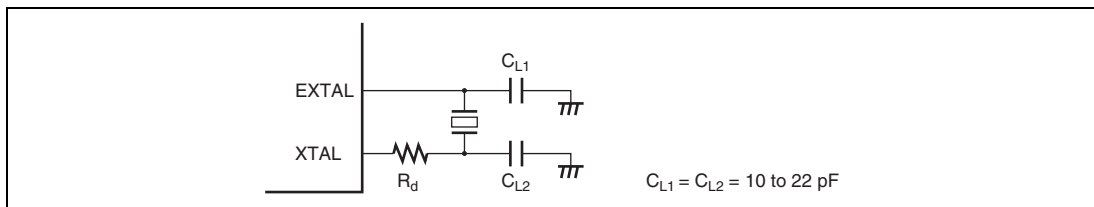
## 23.2 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

### 23.2.1 Connecting Crystal Resonator

A crystal resonator can be connected as shown in the example in figure 23.2. Select the damping resistance  $R_d$  according to table 23.1. An AT-cut parallel-resonance type should be used.

When the clock is provided by connecting a crystal resonator, a crystal resonator having a frequency of 8 to 18 MHz should be connected.

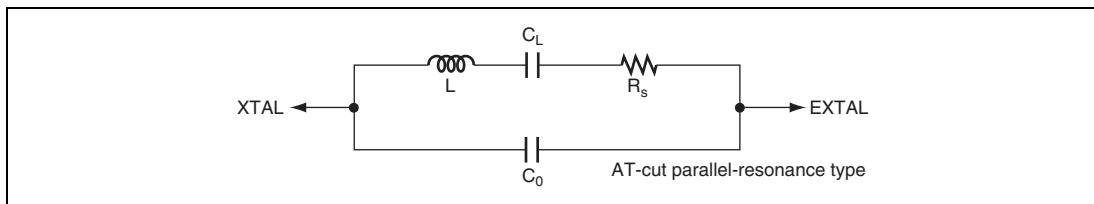


**Figure 23.2 Connection of Crystal Resonator (Example)**

**Table 23.1 Damping Resistance Value**

Frequency (MHz)	8	12	16	18
$R_d$ ( $\Omega$ )	200	0	0	0

Figure 23.3 shows an equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 23.2.



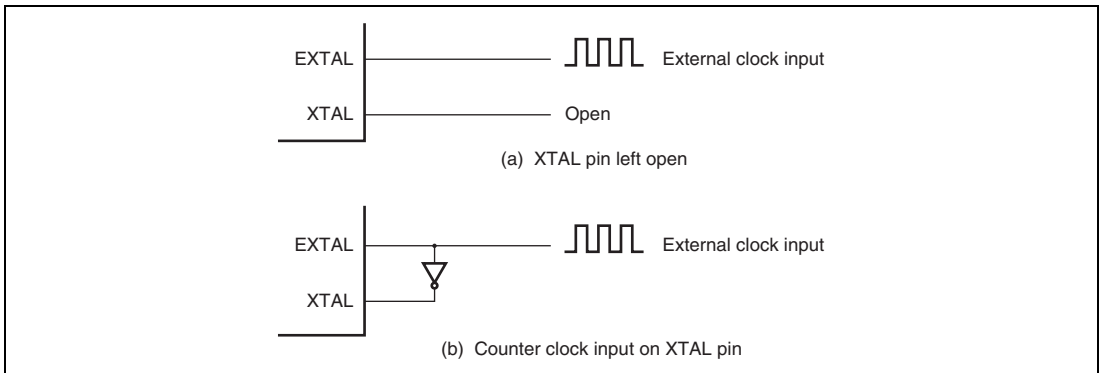
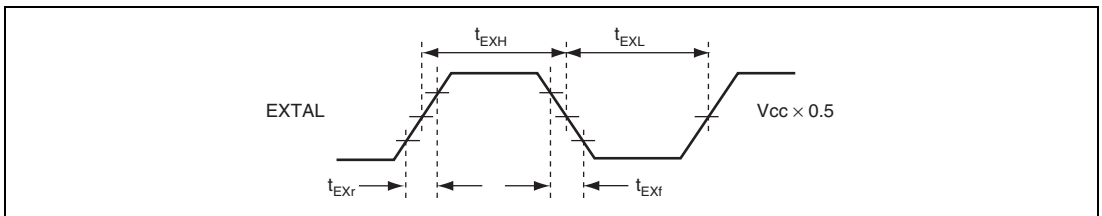
**Figure 23.3 Crystal Resonator Equivalent Circuit**

**Table 23.2 Crystal Resonator Characteristics**

Frequency (MHz)	8	12	16	18
$R_s$ Max. ( $\Omega$ )	80	60	50	40
$C_0$ Max. (pF)			7	

### 23.2.2 External Clock Input

An external clock signal can be input as shown in the examples in figure 23.4. If the XTAL pin is left open, make sure that parasitic capacitance is no more than 10 pF. When the counter clock is input to the XTAL pin, make sure that the external clock is held high in standby mode.

**Figure 23.4 External Clock Input (Examples)****Figure 23.5 External Clock Input Timing**

## 23.3 PLL Circuit

The PLL circuit has the function of multiplying the frequency of the clock from the oscillator by a factor of 4. The frequency multiplication factor is fixed. The phase difference is controlled so that the timing of the rising edge of the internal clock is the same as that of the EXTAL pin signal.

## 23.4 Frequency Divider

### 23.4.1 $1\phi$ , $B\phi$ , $P\phi$ Frequency Dividers

The frequency divider divides the PLL clock to generate a 1/2, 1/4, or 1/8 clock. After bits ICK2 to ICK0, PCK 2 to PCK0, and BCK2 to BCK0 are modified, this LSI operates at the modified frequency.

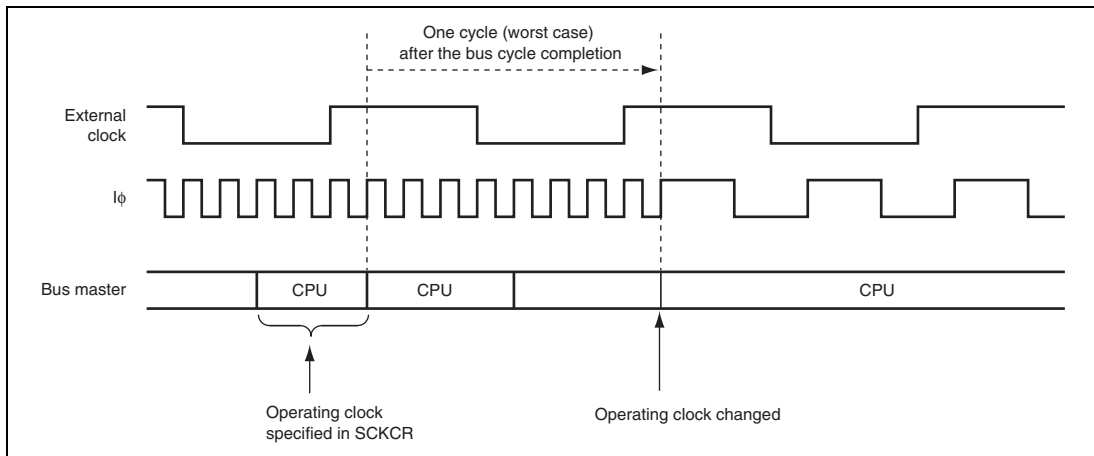
### 23.4.2 $A\phi$ Frequency Divider

The frequency divider divides the frequency of the PLL clock to create 1/3, 1/4, 1/5, and 1/6 clocks. After the ACK2, ACK1, and ACK0 bits are rewritten, the  $\Delta\Sigma$  A/D converter operates according to the frequency available after change. Before rewriting these bits, you need to set the  $\Delta\Sigma$  A/D converter's module stop bit to 1 so that the  $\Delta\Sigma$  A/D converter is stopped. Setting this frequency is recommended because of the characteristics of the  $\Delta\Sigma$  A/D converter: that is,  $A\phi$  is designed to produce maximum accuracy in the neighborhood of 25 MHz.

## 23.5 Usage Notes

### 23.5.1 Notes on Clock Pulse Generator

1. The following points should be noted since the frequency of  $\phi$  ( $I\phi$ : system clock,  $P\phi$ : peripheral module clock,  $B\phi$ : external bus clock) supplied to each module changes according to the setting of SCKCR.  
Select a clock division ratio that is within the operation guaranteed range of clock cycle time  $t_{\text{cyc}}$  shown in the AC timing of electrical characteristics.  
The setting should be within the operation guaranteed range of  $8 \text{ MHz} \leq I\phi \leq 50 \text{ MHz}$ ,  $8 \text{ MHz} \leq P\phi \leq 35 \text{ MHz}$ , and  $8 \text{ MHz} \leq B\phi \leq 50 \text{ MHz}$ .
2. All the on-chip peripheral modules (except for the DMAC and DTC) operate on the  $P\phi$ . Therefore, note that the time processing of modules such as a timer and SCI differs before and after changing the clock division ratio.  
In addition, wait time for clearing software standby mode differs by changing the clock division ratio. For details, see section 24.7.3, Setting Oscillation Settling Time after Exit from Software Standby Mode.
3. The relationship among the system clock, peripheral module clock, and external bus clock is  $I\phi \geq P\phi$  and  $I\phi \geq B\phi$ . In addition, the system clock setting has the highest priority. Accordingly,  $P\phi$  or  $B\phi$  may have the frequency set by bits ICK2 to ICK0 regardless of the settings of bits PCK2 to PCK0 or BCK2 to BCK0.
4. Figure 23.6 shows the clock modification timing. After a value is written to SCKCR, this LSI waits for the current bus cycle to complete. After the current bus cycle completes, each clock frequency will be modified within one cycle (worst case) of the external input clock  $\phi$ .



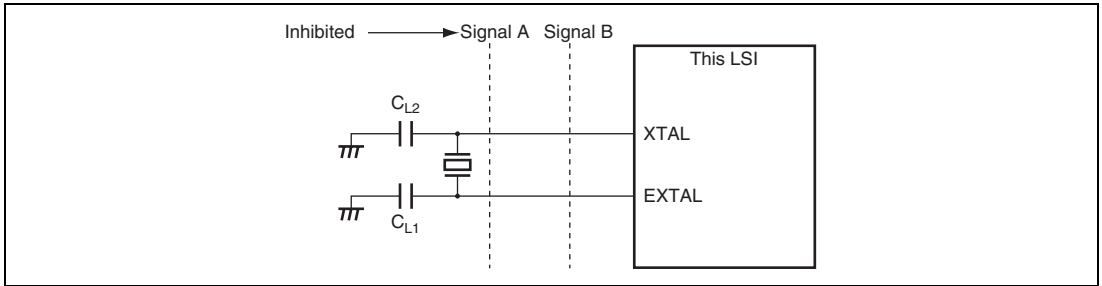
**Figure 23.6 Clock Modification Timing**

### 23.5.2 Notes on Resonator

Since various characteristics related to the resonator are closely linked to the user's board design, thorough evaluation is necessary on the user's part, using the resonator connection examples shown in this section as a reference. As the parameters for the resonator will depend on the floating capacitance of the resonator and the mounting circuit, the parameters should be determined in consultation with the resonator manufacturer. The design must ensure that a voltage exceeding the maximum rating is not applied to the resonator pin.

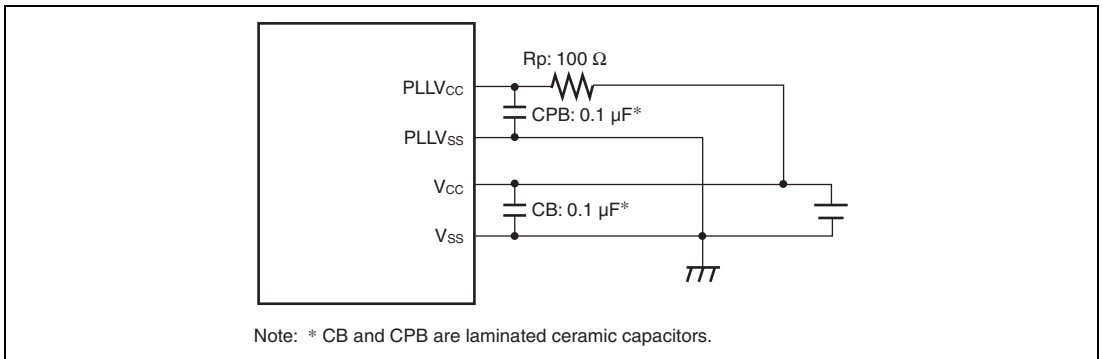
### 23.5.3 Notes on Board Design

When using the crystal resonator, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins. Other signal lines should be routed away from the oscillation circuit as shown in figure 23.7 to prevent induction from interfering with correct oscillation.



**Figure 23.7 Note on Board Design for Oscillation Circuit**

Figure 23.8 shows the external circuitry recommended for the PLL circuit. Separate PLLV<sub>cc</sub> and PLLV<sub>ss</sub> from the other V<sub>cc</sub> and V<sub>ss</sub> lines at the board power supply source, and be sure to insert bypass capacitors CPB and CB close to the pins.



**Figure 23.8 Recommended External Circuitry for PLL Circuit**

## Section 24 Power-Down Modes

Functions for reduced power consumption by this LSI include a multi-clock function, module stop function, and a function for transition to power-down mode.

### 24.1 Features

- Multi-clock function  
The frequency division ratio is settable independently for the system clock, peripheral module clock, and external bus clock.
- Module stop function  
The functions for each peripheral module can be stopped to make a transition to a power-down mode.
- Transition function to power-down mode  
Transition to a power-down mode is possible to stop the CPU, peripheral modules, and oscillator.
- Five power-down modes
  - Sleep mode
  - All-module-clock-stop mode
  - Software standby mode
  - Deep software standby mode
  - Hardware standby mode

Table 24.1 shows conditions to shift to a power-down mode, states of the CPU and peripheral modules, and clearing method for each mode. After the reset state, since this LSI operates in normal program execution state, the modules, other than the DMAC and DTC, are stopped.

**Table 24.1 States of Operation**

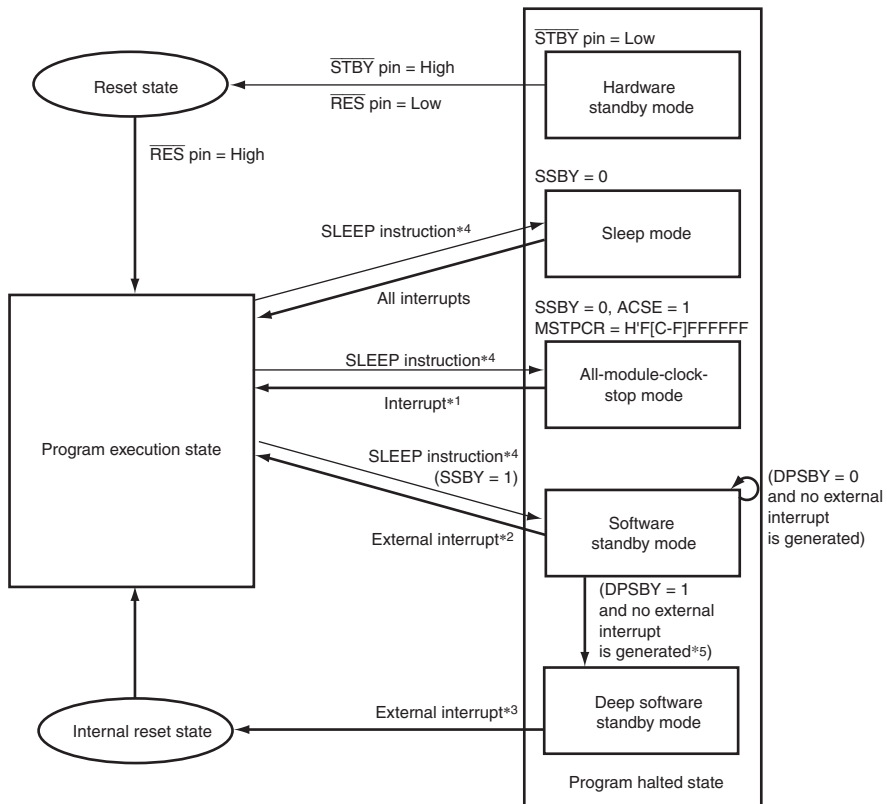
State of Operation	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode	Deep Software Standby Mode	Hardware Standby Mode
Transition condition	Control register + instruction	Control register + instruction	Control register + instruction	Control register + instruction	Pin input
Cancellation method	Interrupt	Interrupt* <sup>2</sup>	External interrupt	External interrupt	
Oscillator	Operating	Operating	Halted	Halted	Halted
CPU	Halted (retained)	Halted (retained)	Halted (retained)	Halted (undefined)	Halted (undefined)
On-chip RAM	Operating (retained)	Halted (retained)	Halted (retained)	Halted (retained/undefined)* <sup>5</sup>	Halted (undefined)
Watchdog timer	Operating	Operating	Halted (retained)	Halted (undefined)	Halted (undefined)
8-bit timer (unit 0/1)	Operating	Operating* <sup>4</sup>	Halted (retained)	Halted (undefined)	Halted (undefined)
Other peripheral modules	Operating	Halted* <sup>1</sup>	Halted* <sup>1</sup>	Halted* <sup>7</sup> (undefined)	Halted* <sup>3</sup> (undefined)
I/O ports	Operating	Retained	Retained* <sup>6</sup>	Halted* <sup>6</sup>	Hi-Z

Notes: "Halted (retained)" in the table means that the internal values are retained and internal operations are suspended.

"Halted (undefined)" in the table means that the internal values are undefined and the power supply for internal operations is turned off.

1. SCI and  $\Sigma\Delta$  A/D converter enters the reset state, and other peripheral modules retain their states.
2. External interrupt and some internal interrupts (8-bit timer and watchdog timer).
3. All peripheral modules enter the reset state.
4. "Functioning" or "Halted" is selectable through the setting of bits MSTPA11 to MSTPA8 in MSTPCRA.
5. "Retained" or "undefined" of the contents of RAM is selected by the setting of the bits RAMCUT2 to RAMCUT0 in DPSBYCR.
6. Retention or high-impedance for the address bus and bus-control signals ( $\overline{CS0}$  to  $\overline{CS7}$ ,  $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{HWR}$ , and  $\overline{LWR}$ ) is selected by the setting of the OPE bit in SBYCR.
7. Some peripheral modules enter a state where the register values are retained.





[Legend]  $\longrightarrow$  Transition after exception handling

Notes: 1. NMI,  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ15}}$ , 8-bit timer interrupts, and watchdog timer interrupts.

Note that the 8-bit timer interrupt is valid when the MSTPCRA11 to MSTPCRA8 bit is cleared to 0.

2. NMI, and  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ15}}$ . Note that IRQ is valid only when the corresponding bit in SSIER is set to 1.

3. NMI, and  $\overline{\text{IRQ0-A}}$  to  $\overline{\text{IRQ3-A}}$ . Note that IRQ is valid only when the corresponding bit in DPSIER is set to 1.

4. The SLPIE bit in SBYCR is cleared to 0.

5. If a conflict between a transition to deep software standby mode and generation of software standby mode clearing source occurs, a mode transition may be made from software standby mode to program execution state through execution of interrupt exception handling. In this case, a transition to deep software standby mode is not made. For details, refer to section 24.12, Usage Notes.

From any state, a transition to hardware standby mode occurs when  $\overline{\text{STBY}}$  is driven low.

From any state except hardware standby mode, a transition to the reset state occurs when  $\overline{\text{RES}}$  is driven low.

**Figure 24.1 Mode Transitions**

## 24.2 Register Descriptions

The registers related to the power-down modes are shown below. For details on the system clock control register (SCKCR), refer to section 23.1.1, System Clock Control Register (SCKCR).

- Standby control register (SBYCR)
- Module stop control register A (MSTPCRA)
- Module stop control register B (MSTPCRB)
- Module stop control register C (MSTPCRC)
- Deep standby control register (DPSBYCR)
- Deep standby wait control register (DPSWCR)
- Deep standby interrupt enable register (DPSIER)
- Deep standby interrupt flag register (DPSIFR)
- Deep standby interrupt edge register (DPSIEGR)
- Reset status register (RSTSR)
- Deep standby backup register (DPSBKRn)

### 24.2.1 Standby Control Register (SBYCR)

SBYCR controls software standby mode.

Bit	15	14	13	12	11	10	9	8
Bit name	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0
Initial value:	0	1	0	0	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit name	SLPIE	—	—	—	—	—	—	—
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	Bit Name	Initial Value	R/W	Description
15	SSBY	0	R/W	<p>Software Standby</p> <p>Specifies the transition mode after executing the SLEEP instruction</p> <p>0: Shifts to sleep mode after the SLEEP instruction is executed</p> <p>1: Shifts to software standby mode after the SLEEP instruction is executed</p> <p>This bit does not change when clearing the software standby mode by using interrupts and shifting to normal operation. For clearing, write 0 to this bit. When the WDT is used in watchdog timer mode, the setting of this bit is disabled. In this case, a transition is always made to sleep mode or all-module-clock-stop mode after the SLEEP instruction is executed. When the SLPIE bit is set to 1, this bit should be cleared to 0.</p>
14	OPE	1	R/W	<p>Output Port Enable</p> <p>Specifies whether the output of the address bus and bus control signals (<math>\overline{CS0}</math> to <math>\overline{CS7}</math>, <math>\overline{AS}</math>, <math>\overline{RD}</math>, <math>\overline{HWR}</math>, and <math>\overline{LWR}</math>) is retained or these lines are set to the high-Z state in software standby mode or deep software standby mode.</p> <p>0: In software standby mode or deep software standby mode, address bus and bus control signal lines are high-impedance.</p> <p>1: In software standby mode or deep software standby mode, output states of address bus and bus control signals are retained.</p>
13	•	0	R/W	<p>Reserved</p> <p>This bit is always read as 0. The write value should always be 0.</p>

Bit	Bit Name	Initial Value	R/W	Description
12	STS4	0	R/W	Standby Timer Select 4 to 0
11	STS3	1	R/W	<p>These bits select the time the MCU waits for the clock to settle when software standby mode is cleared by an external interrupt. With a crystal resonator, refer to table 24.2 and make a selection according to the operating frequency so that the standby time is at least equal to the oscillation settling time. With an external clock, a PLL circuit settling time is necessary. Refer to table 24.2 to set the standby time.</p> <p>While oscillation is being settled, the timer is counted on the P<math>\phi</math> clock frequency. Careful consideration is required in multi-clock mode.</p> <p>00000: Reserved  00001: Reserved  00010: Reserved  00011: Reserved  00100: Reserved  00101: Standby time = 64 states  00110: Standby time = 512 states  00111: Standby time = 1024 states  01000: Standby time = 2048 states  01001: Standby time = 4096 states  01010: Standby time = 16384 states  01011: Standby time = 32768 states  01100: Standby time = 65536 states  01101: Standby time = 131072 states  01110: Standby time = 262144 states  01111: Standby time = 524288 states  1xxxx: Reserved</p>
10	STS2	1	R/W	
9	STS1	1	R/W	
8	STS0	1	R/W	

Bit	Bit Name	Initial Value	R/W	Description
7	SLPIE	0	R/W	<p>Sleep Instruction Exception Handling Enable</p> <p>Selects whether a sleep interrupt is generated or a transition to power-down mode is made when a SLEEP instruction is executed.</p> <p>0: A transition to power-down mode is made when a SLEEP instruction is executed.</p> <p>1: A sleep instruction exception handling is generated when a SLEEP instruction is executed.</p> <p>Even after a sleep instruction exception handling is executed, this bit remains set to 1. For clearing, write 0 to this bit.</p>
6 to 0	—	All 0	R/W	<p>Reserved</p> <p>These bits are always read as 0. The write value should always be 0.</p>

- Notes: 1. x: Don't care  
 2. With the F-ZTAT version, the flash memory settling time must be reserved.

### 24.2.2 Module Stop Control Registers A and B (MSTPCRA and MSTPCRB)

MSTPCRA and MSTPCRB control module stop state. Setting a bit to 1 makes the corresponding module enter module stop state, while clearing the bit to 0 clears module stop state.

#### • MSTPCRA

Bit	15	14	13	12	11	10	9	8
Bit name	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8
Initial value:	0	0	0	0	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit name	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0
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

- **MSTPCRB**

Bit	15	14	13	12	11	10	9	8
Bit name	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8
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

Bit	7	6	5	4	3	2	1	0
Bit name	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0
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

- **MSTPCRA**

Bit	Bit Name	Initial Value	R/W	Module
15	ACSE	0	R/W	All-Module-Clock-Stop Mode Enable Enables/disables all-module-clock-stop state for reducing current consumption by stopping the bus controller and I/O ports operations when the CPU executes the SLEEP instruction after module stop mode has been set for all the on-chip peripheral modules controlled by MSTPCR. 0: All-module-clock-stop mode disabled 1: All-module-clock-stop mode enabled
14	MSTPA14	0	R/W	Reserved
13	MSTPA13	0	R/W	DMA controller (DMAC)
12	MSTPA12	0	R/W	Data transfer controller (DTC)
11	MSTPA11	1	R/W	8-bit timer (TMR_7 and TMR_6)
10	MSTPA10	1	R/W	8-bit timer (TMR_5 and TMR_4)
9	MSTPA9	1	R/W	8-bit timer (TMR_3 and TMR_2)
8	MSTPA8	1	R/W	8-bit timer (TMR_1 and TMR_0)
7	MSTPA7	1	R/W	Reserved
6	MSTPA6	1	R/W	These bits are always read as 1. The write value should always be 1.

Bit	Bit Name	Initial Value	R/W	Module
5	MSTPA5	1	R/W	D/A converter (channels 1 and 0)
4	MSTPA4	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
3	MSTPA3	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
2	MSTPA2	1	R/W	Reserved
1	MSTPA1	1	R/W	These bits are always read as 1. The write value should always be 1.
0	MSTPA0	1	R/W	16-bit timer pulse unit (TPU channels 5 to 0)

- MSTPCRB

Bit	Bit Name	Initial Value	R/W	Module
15	MSTPB15	1	R/W	Programmable pulse generator (PPG)
14	MSTPB14	1	R/W	Reserved
13	MSTPB13	1	R/W	These bits are always read as 1. The write value should always be 1.
12	MSTPB12	1	R/W	Serial communication interface_4 (SCI_4)
11	MSTPB11	1	R/W	Serial communication interface_3 (SCI_3)
10	MSTPB10	1	R/W	Serial communication interface_2 (SCI_2)
9	MSTPB9	1	R/W	Serial communication interface_1 (SCI_1)
8	MSTPB8	1	R/W	Serial communication interface_0 (SCI_0)
7	MSTPB7	1	R/W	I <sup>2</sup> C bus Interface 1 (IIC_1)
6	MSTPB6	1	R/W	I <sup>2</sup> C bus Interface 0 (IIC_0)
5	MSTPB5	1	R/W	User break controller (UBC)
4	MSTPB4	1	R/W	Reserved
3	MSTPB3	1	R/W	These bits are always read as 1. The write value should always be 1.
2	MSTPB2	1	R/W	
1	MSTPB1	1	R/W	
0	MSTPB0	1	R/W	

### 24.2.3 Module Stop Control Register C (MSTPCRC)

When bits MSTPC4 to MSTPC0 are set to 1, the corresponding on-chip RAM stops. Do not set the corresponding MSTPC4 to MSTPC0 bits to 1 while accessing the on-chip RAM. Do not access the on-chip RAM while bits MSTPC4 to MSTPC0 are set to 1.

MSTPC14 controls the module stop for the  $\Delta\Sigma$  A/D converter, while MSTPC13 controls the module stop for the A/D converter.

Bit	15	14	13	12	11	10	9	8
Bit name	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8
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
Bit	7	6	5	4	3	2	1	0
Bit name	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0
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	Bit Name	Initial Value	R/W	Module
15	MSTPC15	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
14	MSTPC14	1	R/W	$\Delta\Sigma$ A/D converter
13	MSTPC13	1	R/W	A/D converter
12	MSTPC12	1	R/W	Reserved
11	MSTPC11	1	R/W	These bits are always read as 1. The write value should always be 1.
10	MSTPC10	1	R/W	
9	MSTPC9	1	R/W	
8	MSTPC8	1	R/W	



Bit	Bit Name	Initial Value	R/W	Module
7	MSTPC7	0	R/W	Reserved
6	MSTPC6	0	R/W	These bits are always read as 0. The write value should always be 0.
5	MSTPC5	0	R/W	On-chip RAM Always set the MSTPC2 and MSTPC5 bits to the same value.
4	MSTPC4	0	R/W	Reserved
3	MSTPC3	0	R/W	These bits are always read as 0. The write value should always be 0.
2	MSTPC2	0	R/W	On-chip RAM_2 (H'FF6000 to H'FF7FFF) Always set the MSTPC2 and MSTPC5 bits to the same value.
1	MSTPC1	0	R/W	On-chip RAM_1, 0 (H'FF8000 to H'FFBFFF)
0	MSTPC0	0	R/W	Always set the MSTPC1 and MSTPC0 bits to the same value.

#### 24.2.4 Deep Standby Control Register (DPSBYCR)

DPSBYCR controls deep software standby mode.

DPSBYCR is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

Bit	7	6	5	4	3	2	1	0
Bit name	DPSBY	IOKEEP	RAMCUT2	RAMCUT1	—	—	—	RAMCUT0
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	R/W

Bit	Bit Name	Initial Value	R/W	Module												
7	DPSBY	0	R/W	Deep Software Standby												
<p>When the SSBY bit in SBYCR has been set to 1, executing the SLEEP instruction causes a transition to software standby mode. At this time, if there is no source to clear software standby mode and this bit is set to 1, a transition to deep software standby mode is made.</p>																
<table border="1"> <thead> <tr> <th>SSBY</th> <th>DPSBY</th> <th>Entry to</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>x</td> <td>Enters sleep mode after execution of a SLEEP instruction.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Enters software standby mode after execution of a SLEEP instruction.</td> </tr> <tr> <td>1</td> <td>1</td> <td>Enters deep software standby mode after execution of a SLEEP instruction.</td> </tr> </tbody> </table>					SSBY	DPSBY	Entry to	0	x	Enters sleep mode after execution of a SLEEP instruction.	1	0	Enters software standby mode after execution of a SLEEP instruction.	1	1	Enters deep software standby mode after execution of a SLEEP instruction.
SSBY	DPSBY	Entry to														
0	x	Enters sleep mode after execution of a SLEEP instruction.														
1	0	Enters software standby mode after execution of a SLEEP instruction.														
1	1	Enters deep software standby mode after execution of a SLEEP instruction.														

When deep software standby mode is canceled due to an external interrupt, this bit remains at 1. Write a 0 here to clear it. Setting of this bit has no effect when the WDT is used in watchdog timer mode. In this case, executing the SLEEP instruction always initiates entry to sleep mode or all-module-clock-stop mode. Be sure to clear this bit to 0 when setting the SLPIE bit to 1.

Bit	Bit Name	Initial Value	R/W	Module						
6	IOKEEP	0	R/W	I/O Port Retention  In deep software standby mode, the ports retain the states that were held in software standby mode. This bit specifies whether or not the state that has been held in deep software standby mode is retained after exit from deep software standby mode.  <table border="1"> <thead> <tr> <th>IOKEEP</th> <th>Pin State</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>The retained port states are released simultaneously with exit from deep software standby mode.</td> </tr> <tr> <td>1</td> <td>The retained port states are released when a 0 is written to this bit following exit from deep software standby mode.</td> </tr> </tbody> </table> In operation in external extended mode, however, the address bus, bus control signals ( $\overline{CS0}$ , $\overline{AS}$ , $\overline{RD}$ , $\overline{HWR}$ , and $\overline{LWR}$ ), and data bus are set to the initial state upon exit from deep software standby mode.	IOKEEP	Pin State	0	The retained port states are released simultaneously with exit from deep software standby mode.	1	The retained port states are released when a 0 is written to this bit following exit from deep software standby mode.
IOKEEP	Pin State									
0	The retained port states are released simultaneously with exit from deep software standby mode.									
1	The retained port states are released when a 0 is written to this bit following exit from deep software standby mode.									
5	RAMCUT2	0	R/W	On-chip RAM Power Off 2  Controls the internal power supply to the on-chip RAM in deep software standby mode. For details, see descriptions of the RAMCUT0 bit.						
4	RAMCUT1	0	R/W	On-chip RAM Power Off 1  Controls the internal power supply to the on-chip RAM in deep software standby mode. For details, see descriptions of the RAMCUT0 bit.						
3 to 1	—	All 0	R/W	Reserved  These bits are always read as 0. The write value should always be 0.						
0	RAMCUT0	1	R/W	On-chip RAM Power Off 0  Controls the internal power supply to the on-chip RAM in deep software standby mode, in combination with RAMCUT2 and RAMCUT 1.  000: Power is supplied to the on-chip RAM. 111: Power is not supplied to the on-chip RAM. Settings other than above are prohibited.						

### 24.2.5 Deep Standby Wait Control Register (DPSWCR)

DPSWCR selects the time for which the MCU waits until the clock settles when deep software standby mode is canceled by an external interrupt.

DPSWCR is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

Bit	7	6	5	4	3	2	1	0
Bit name	—	—	WTSTS5	WTSTS4	WTSTS3	WTSTS2	WTSTS1	WTSTS0
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	Bit Name	Initial Value	R/W	Module
7, 6	—	All 0	R/W	Reserved
These bits are always read as 0. The write value should always be 0.				

Bit	Bit Name	Initial Value	R/W	Module
5 to 0	WTSTS [5:0]	0	R/W	<p>Deep Software Standby Wait Time Setting</p> <p>These bits select the time for which the MCU waits until the clock settles when deep software standby mode is canceled by an external interrupt.</p> <p>When using a crystal resonator, see table 24.3 and select the wait time greater than the oscillation settling time for each operating frequency. When using an external clock, settling time for the PLL circuit should be considered. See table 24.3 to select the wait time.</p> <p>During the oscillation settling period, counting is performed with the clock frequency input to the EXTAL.</p> <p>000000: Reserved</p> <p>000001: Reserved</p> <p>000010: Reserved</p> <p>000011: Reserved</p> <p>000100: Reserved</p> <p>000101: Wait time = 64 states</p> <p>000110: Wait time = 512 states</p> <p>000111: Wait time = 1024 states</p> <p>001000: Wait time = 2048 states</p> <p>001001: Wait time = 4096 states</p> <p>001010: Wait time = 16384 states</p> <p>001011: Wait time = 32768 states</p> <p>001100: Wait time = 65536 states</p> <p>001101: Wait time = 131072 states</p> <p>001110: Wait time = 262144 states</p> <p>001111: Wait time = 524288 states</p> <p>01xxxx: Reserved</p>

### 24.2.6 Deep Standby Interrupt Enable Register (DPSIER)

DPSIER enables or disables interrupts to clear deep software standby mode.

DPSIER is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

Bit	7	6	5	4	3	2	1	0
Bit name	—	—	—	—	DIRQ3E	DIRQ2E	DIRQ1E	DIRQ0E
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	Bit Name	Initial Value	R/W	Module
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6 to 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
3	DIRQ3E	0	R/W	IRQ3 Interrupt Enable Enables or disables exit from deep software standby mode by IRQ3. 0: Disables exit from deep software standby mode by IRQ3. 1: Enables exit from deep software standby mode by IRQ3.
2	DIRQ2E	0	R/W	IRQ2 Interrupt Enable Enables or disables exit from deep software standby mode by IRQ2. 0: Disables exit from deep software standby mode by IRQ2. 1: Enables exit from deep software standby mode by IRQ2.

Bit	Bit Name	Initial Value	R/W	Module
1	DIRQ1E	0	R/W	IRQ1 Interrupt Enable Enables or disables exit from deep software standby mode by IRQ1. 0: Disables exit from deep software standby mode by IRQ1. 1: Enables exit from deep software standby mode by IRQ1.
0	DIRQ0E	0	R/W	IRQ0 Interrupt Enable Enables or disables exit from deep software standby mode by IRQ0. 0: Disables exit from deep software standby mode by IRQ0. 1: Enables exit from deep software standby mode by IRQ0.

### 24.2.7 Deep Standby Interrupt Flag Register (DPSIFR)

DPSIFR is used to request an exit from deep software standby mode. When the interrupt specified in DPSIEGR is generated, the applicable bit in DPSIFR is set to 1. The bit is set to 1 even when an interrupt is generated in the modes other than deep software standby. Therefore, a transition to deep software standby should be made after this register bits are cleared to 0.

DPSIFR is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

Bit	7	6	5	4	3	2	1	0
Bit name	DNMIF	—	—	—	DIRQ3F	DIRQ2F	DIRQ1F	DIRQ0F
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R	R	R	R/(W)*	R/(W)*	R/(W)*	R/(W)*

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

Bit	Bit Name	Initial Value	R/W	Module
7	DNMIF	0	R/(W)*	NMI Flag [Setting condition] NMI input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.
6 to 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
3	DIRQ3F	0	R/(W)*	IRQ3 Interrupt Flag [Setting condition] IRQ3 input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.
2	DIRQ2F	0	R/(W)*	IRQ2 Interrupt Flag [Setting condition] IRQ2 input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.
1	DIRQ1F	0	R/(W)*	IRQ1 Interrupt Flag [Setting condition]* IRQ1 input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.
0	DIRQ0F	0	R/(W)*	IRQ0 Interrupt Flag [Setting condition]* IRQ0 input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.

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



## 24.2.8 Deep Standby Interrupt Edge Register (DPSIEGR)

DPSIEGR selects the rising or falling edge to clear deep software standby mode.

DPSIEGR is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

Bit	7	6	5	4	3	2	1	0
Bit name	DNMIEG	—	—	—	DIRQ3EG	DIRQ2EG	DIRQ1EG	DIRQ0EG
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	Bit Name	Initial Value	R/W	Module
7	DNMIEG	0	R/W	<b>NMI Edge Select</b> Selects the active edge for NMI pin input. 0: The interrupt request is generated by a falling edge. 1: The interrupt request is generated by a rising edge.
6 to 4	—	All 0	R/W	<b>Reserved</b> These bits are always read as 0. The write value should always be 0.
3	DIRQ3EG	0	R/W	<b>IRQ3 Interrupt Edge Select</b> Selects the active edge for IRQ3 pin input. 0: The interrupt request is generated by a falling edge. 1: The interrupt request is generated by a rising edge.
2	DIRQ2EG	0	R/W	<b>IRQ2 Interrupt Edge Select</b> Selects the active edge for IRQ2 pin input. 0: The interrupt request is generated by a falling edge. 1: The interrupt request is generated by a rising edge.
1	DIRQ1EG	0	R/W	<b>IRQ1 Interrupt Edge Select</b> Selects the active edge for IRQ1 pin input. 0: The interrupt request is generated by a falling edge. 1: The interrupt request is generated by a rising edge.

Bit	Bit Name	Initial Value	R/W	Module
0	DIRQ0EG	0	R/W	IRQ0 Interrupt Edge Select Selects the active edge for IRQ0 pin input. 0: The interrupt request is generated by a falling edge. 1: The interrupt request is generated by a rising edge.

### 24.2.9 Reset Status Register (RSTSR)

The DPSRSTF bit in RSTSR indicates that deep software standby mode has been canceled by an interrupt.

RSTSR is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

Bit	7	6	5	4	3	2	1	0
Bit name	DPSRSTF	—	—	—	—	—	—	—
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.

Bit	Bit Name	Initial Value	R/W	Module
7	DPSRSTF	0	R/(W)*	Deep Software Standby Reset Flag Indicates that deep software standby mode has been canceled by an external interrupt source specified in DPSIER or DPSIEGR and an internal reset is generated. [Setting condition] Deep software standby mode is canceled by an external interrupt source. [Clearing condition] Writing a 0 to this bit after reading it as 1.
6 to 0	—	0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

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

### 24.2.10 Deep Standby Backup Register (DPSBKRn)

DPSBKRn (n = 15 to 0) is a 16-bit readable/writable register to store data during deep software standby mode.

Although data in on-chip RAM is not retained in deep software standby mode, data in this register is retained.

DPSBKRn is initialized by input of the reset signal on the  $\overline{\text{RES}}$  pin, but is not initialized by the internal reset signal upon exit from deep software standby mode.

## 24.3 Multi-Clock Function

When bits ICK2 to ICK0, PCK2 to PCK0, and BCK2 to BCK0 in SCKCR are set, the clock frequency is changed at the end of the bus cycle. The CPU and bus masters operate on the operating clock specified by bits ICK2 to ICK0. The peripheral modules operate on the operating clock specified by bits PCK2 to PCK0. The external bus operates on the operating clock specified by bits BCK2 to BCK0.

Even if the frequencies specified by bits PCK2 to PCK0 and BCK2 to BCK0 are higher than the frequency specified by bits ICK2 to ICK0, the specified values are not reflected in the peripheral module and external bus clocks. The peripheral module and external bus clocks are restricted to the operating clock specified by bits ICK2 to ICK0.

## 24.4 Module Stop State

Module stop functionality can be set for individual on-chip peripheral modules.

When the corresponding MSTP bit in MSTPCRA, MSTPCRB, or MSTPCRC is set to 1, module operation stops at the end of the bus cycle and a transition is made to a module stop state. The CPU continues operating independently.

When the corresponding MSTP bit is cleared to 0, a module stop state is cleared and the module starts operating at the end of the bus cycle. In a module stop state, the internal states of modules other than the SCI are retained.

After the reset state is cleared, all modules other than the DMAC, DTC, and on-chip RAM are placed in a module stop state.

The registers of the module for which the module stop state is selected cannot be read from or written to.

## 24.5 Sleep Mode

### 24.5.1 Entry to Sleep Mode

When the SLEEP instruction is executed when the SSBY bit in SBYCR is 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other peripheral functions do not stop.

### 24.5.2 Exit from Sleep Mode

Sleep mode is exited by any interrupt, signals on the  $\overline{\text{RES}}$  or  $\overline{\text{STBY}}$  pin, and a reset caused by a watchdog timer overflow.

- Exit from sleep mode by interrupt

When an interrupt occurs, sleep mode is exited and interrupt exception processing starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.

- Exit from sleep mode by  $\overline{\text{RES}}$  pin

Setting the  $\overline{\text{RES}}$  pin level low selects the reset state. After the stipulated reset input duration, driving the  $\overline{\text{RES}}$  pin high makes the CPU start the reset exception processing.

- Exit from sleep mode by  $\overline{\text{STBY}}$  pin

When the  $\overline{\text{STBY}}$  pin level is driven low, a transition is made to hardware standby mode.

- Exit from sleep mode by reset caused by watchdog timer overflow

Sleep mode is exited by an internal reset caused by a watchdog timer overflow.

## 24.6 All-Module-Clock-Stop Mode

When the ACSE bit is set to 1 and all modules controlled by MSTPCRA and MSTPCRB are stopped (MSTPCRA, MSTPCRB = H'FFFFFFF), or all modules except for the 8-bit timer (units 0 and 1) are stopped (MSTPCRA, MSTPCRB = H'F[C to F]FFFFFF), executing a SLEEP instruction with the SSBY bit in SBYCR cleared to 0 will cause all modules (except for the 8-bit timer\* and watchdog timer), the bus controller, and the I/O ports to stop operating, and to make a transition to all-module-clock-stop mode at the end of the bus cycle.

When power consumption should be reduced ever more in all-module-clock-stop mode, stop modules controlled by MSTPCRC (MSTPCRC[15:8] = H'FFFF).

All-module-clock-stop mode is cleared by an external interrupt (NMI or  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ15}}$  pins),  $\overline{\text{RES}}$  pin input, or an internal interrupt (8-bit timer\* or watchdog timer), and the CPU returns to the normal program execution state via the exception handling state. All-module-clock-stop mode is not cleared if interrupts are disabled, if interrupts other than NMI are masked on the CPU side, or if the relevant interrupt is designated as a DTC activation source.

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

Note: \* Operation or halting of the 8-bit timer can be selected by bits MSTPA11 to MSTPA8 in MSTPCRA.

## 24.7 Software Standby Mode

### 24.7.1 Entry to Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1 and the DPSBY bit in DPSBYCR is cleared to 0, software standby mode is entered. In this mode, the CPU, on-chip peripheral functions, and oscillator all stop. However, the contents of the CPU's internal registers, on-chip RAM data, and the states of on-chip peripheral functions other than the SCI, and the states of the 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, allowing power consumption to be significantly reduced.

If the WDT is used in watchdog timer mode, it is impossible to make a transition to software standby mode. The WDT should be stopped before the SLEEP instruction execution.

### 24.7.2 Exit from Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI, or  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ15}}$ \*) or by means of the  $\overline{\text{RES}}$  pin or  $\overline{\text{STBY}}$  pin.

#### 1. Exit from software standby mode by interrupt

When an NMI, or  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ15}}$ \* interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS4 to STS0 in SBYCR, stable clocks are supplied to the entire LSI, software standby mode is cleared, and interrupt exception handling is started.

When clearing software standby mode with an  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ11}}$ \* interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ11}}$ \* 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: \* By setting the SSIn bit in SSIER to 1,  $\overline{\text{IRQ0}}$  to  $\overline{\text{IRQ15}}$  can be used as a software standby mode clearing source.

#### 2. Exit from software standby mode by $\overline{\text{RES}}$ 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 LSI. Note that the  $\overline{\text{RES}}$  pin must be held low until clock oscillation settles. When the  $\overline{\text{RES}}$  pin goes high, the CPU begins reset exception handling.

#### 3. Exit from software standby mode by $\overline{\text{STBY}}$ pin

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

### 24.7.3 Setting Oscillation Settling Time after Exit from Software Standby Mode

Bits STS4 to STS0 in SBYCR should be set as described below.

1. Using a crystal resonator

Set bits STS4 to STS0 so that the standby time is at least equal to the oscillation settling time. Table 24.2 shows the standby times for operating frequencies and settings of bits STS4 to STS0.


2. Using an external clock


A PLL circuit settling time is necessary. Refer to table 24.2 to set the standby time.

**Table 24.2 Oscillation Settling Time Setting**

STS4	STS3	STS2	STS1	STS0	Standby Time	P $\phi$ * (MHz)						Unit
						35	25	20	13	10	8	
0	0	0	0	0	Reserved	—	—	—	—	—	—	$\mu$ s
				1	Reserved	—	—	—	—	—	—	
			1	0	Reserved	—	—	—	—	—	—	
			1	1	Reserved	—	—	—	—	—	—	
	1	0	0	0	Reserved	—	—	—	—	—	—	
				1	64	1.8	2.6	3.2	4.9	6.4	8.0	
		1	0	0	512	14.6	20.5	25.6	39.4	51.2	64.0	
				1	1024	29.3	41.0	51.2	78.8	102.4	128.0	
1	0	0	0	0	2048	58.5	81.9	102.4	157.5	204.8	256.0	
				1	4096	0.12	0.16	0.20	0.32	0.41	0.51	ms
		1	0	0	16384	0.47	0.66	0.82	1.26	1.64	2.05	
				1	32768	0.94	1.31	1.64	2.52	3.28	4.10	
	1	0	0	0	65536	1.87	2.62	3.28	5.04	6.55	8.19	
				1	131072	3.74	5.24	6.55	10.08	13.11	16.38	
		1	0	0	262144	7.49	10.49	13.11	20.16	26.21	32.77	
				1	524288	14.98	20.97	26.21	40.33	52.43	65.54	
1	0	0	0	0	Reserved	—	—	—	—	—	—	

**[Legend]**

 : Recommended setting when external clock is in use

 : Recommended setting when crystal oscillator is in use

Note: \* P $\phi$  is the output from the peripheral module frequency divider. The oscillation settling time, which includes a period where the oscillation by an oscillator is not stable, depends on the resonator characteristics. The above figures are for reference.

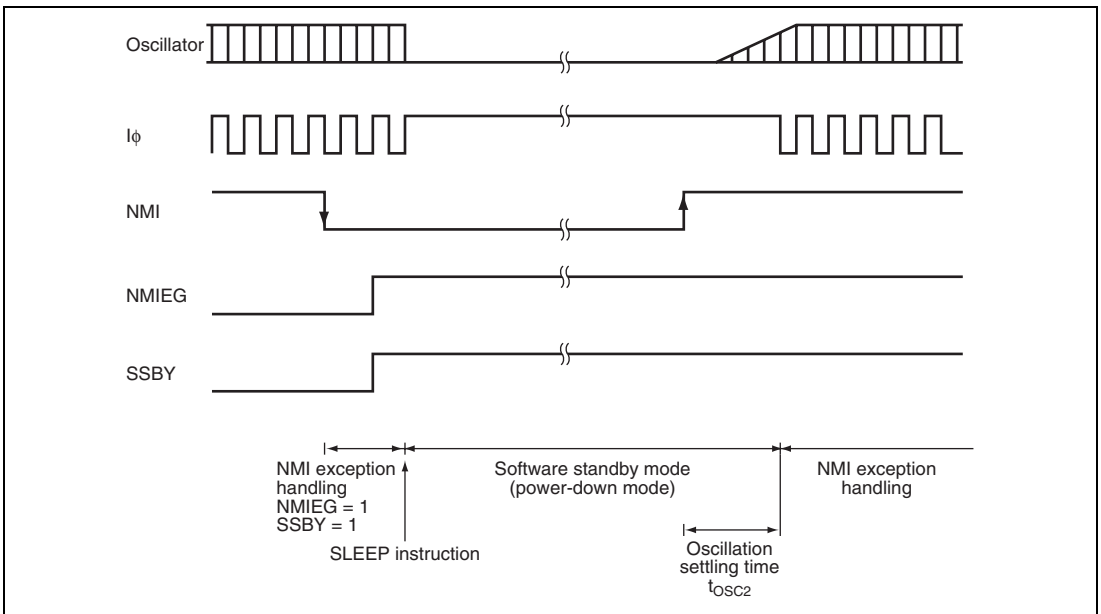


### 24.7.4 Software Standby Mode Application Example

Figure 24.2 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 INTCR 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 24.2 Software Standby Mode Application Example**

## 24.8 Deep Software Standby Mode

### 24.8.1 Entry to Deep Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR has been set to 1, a transition to software standby mode is made. In this state, if the DPSBY bit in DPSBYCR is set to 1, a transition to deep software standby mode is made.

If a software standby mode clearing source (an NMI, or IRQ0 to IRQ15) occurs when a transition to software standby mode is made, software standby mode will be cleared regardless of the DPSBY bit setting, and the interrupt exception handling starts after the oscillation settling time for software standby mode specified by the bits STS4 to STS0 in SBYCR has elapsed.

When both of the SSBY bit in SBYCR and the DPSBY bit in DPSBYCR are set to 1 and no software standby mode clearing source event occurs, a transition to deep software standby mode will be made immediately after software standby mode is entered.

In deep software standby mode, the CPU, on-chip peripheral functions, on-chip RAM, and oscillator functionality are all halted. In addition, the internal power supply to these modules stops, resulting in a significant reduction in power consumption. At this time, the contents of all the registers of the CPU, on-chip peripheral functions, and on-chip RAM become undefined.

Contents of the on-chip RAM can be retained when all the bits RAMCUT2 to RAMCUT0 in DPSBYCR have been cleared to 0. If these bits are set to all 1, the internal power supply to the on-chip RAM stops and the power consumption is further reduced. At this time, the contents of the on-chip RAM become undefined.

The I/O ports can be retained in the same state as in software standby mode.

## 24.8.2 Exit from Deep Software Standby Mode

Exit from deep software standby mode is initiated by signals on the external interrupt pins (NMI and  $\overline{\text{IRQ0-A}}$  to  $\overline{\text{IRQ3-A}}$ ),  $\overline{\text{RES}}$  pin, or  $\overline{\text{STBY}}$  pin.

### 1. Exit from deep software standby mode by external interrupt pins

Deep software standby mode is canceled when any of the DNMI $\overline{\text{F}}$  and DIRQn $\overline{\text{F}}$  (n = 3 to 0) bits in DPSIFR is set to 1. The DNMI $\overline{\text{F}}$  or DIRQn $\overline{\text{F}}$  (n = 3 to 0) bit is set to 1 when a specified edge is generated in the NMI or  $\overline{\text{IRQ0-A}}$  to  $\overline{\text{IRQ3-A}}$  pins, that has been enabled by the DIRQn $\overline{\text{E}}$  (n = 3 to 0) bit in DPSIER. The rising or falling edge of the signals can be specified with DPSIEGR.

When deep software standby mode clearing source is generated, internal power supply starts simultaneously with the start of clock oscillation, and internal reset signal is generated for the entire LSI. Once the time specified by the WTSTS5 to WTSTS0 bits in DPSWCR has elapsed, a stable clock signal is being supplied throughout the LSI and the internal reset is cleared.

Deep software standby mode is canceled on clearing of the internal reset, and then the reset exception handling starts.

When deep software standby mode is canceled by an external interrupt pin, the DPSRSTF bit in RSTSR is set to 1.

### 2. Exit from deep software standby mode by the signal on the $\overline{\text{RES}}$ pin

Clock oscillation and internal power supply start as soon as the signal on the  $\overline{\text{RES}}$  pin is driven low. At the same time, clock signals are supplied to the LSI. In this case, the  $\overline{\text{RES}}$  pin has to be held low until the clock oscillation has become stable. Once the signal on the  $\overline{\text{RES}}$  pin is driven high, the CPU starts reset exception handling.

### 3. Exit from deep software standby mode by the signal on the $\overline{\text{STBY}}$ pin

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

### 24.8.3 Pin State on Exit from Deep Software Standby Mode

In deep software standby mode, the ports retain the states that were held during software standby mode. The internal of the LSI is initialized by an internal reset caused by deep software standby mode, and the reset exception handling starts as soon as deep software standby mode is canceled. The following shows the port states at this time.

#### (1) Pins for address bus, bus control and data bus

Pins for the address bus, bus control signals ( $\overline{\text{CS0}}$ ,  $\overline{\text{AS}}$ ,  $\overline{\text{HWR}}$  and  $\overline{\text{LWR}}$ ), and data bus operate depending on the CPU.

#### (2) Pins other than address bus, bus control and data bus pins

Whether the ports are initialized or retain the states that were held during software standby mode can be selected by the IOKEEP bit.

- When IOKEEP = 0  
Ports are initialized by an internal reset caused by deep software standby mode.
- When IOKEEP = 1  
The port states that were held in deep software standby mode are retained regardless of the LSI internal state though the internal of the LSI is initialized by an internal reset caused by deep software standby mode. At this time, the port states that were held in software standby mode are retained even if settings of I/O ports or peripheral modules are set. Subsequently, the retained port states are released when the IOKEEP bit is cleared to 0 and operation is performed according to the internal settings.

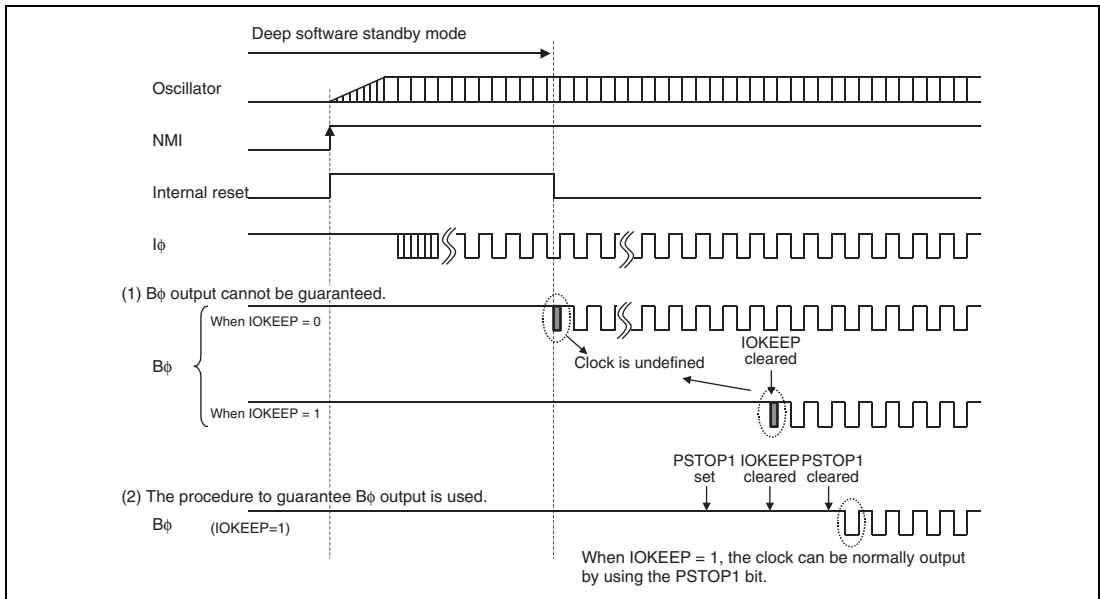
### 24.8.4 B $\phi$ Operation after Exit from Deep Software Standby Mode

When the IOKEEP bit is 0, B $\phi$  output is undefined for a maximum of one cycle immediately after exit from deep software standby mode. At this time, the output state cannot be guaranteed. Even when the IOKEEP bit is set to 1, B $\phi$  output is undefined for a maximum of one cycle immediately after the IOKEEP bit is cleared to 0 after deep software standby mode was canceled, and the output state cannot be guaranteed. (See figure 24.3)

However, clock can be normally output by canceling deep software standby mode with the IOKEEP bit set to 1 and then controlling the B $\phi$  output with the IOKEEP and PSTOP1 bits. Use the following procedure.

1. Change the value of the PSTOP1 bit from 0 to 1 to fix the B $\phi$  output at the high level (given that the B $\phi$  output was already fixed high).
2. Clear the IOKEEP bit to 0 to end retention of the B $\phi$  state.
3. Clear the PSTOP1 bit to 0 to enable B $\phi$  output.

For the port state when the IOKEEP bit is set to 1, see section 24.8.3, Pin State on Exit from Deep Software Standby Mode.



**Figure 24.3 B $\phi$  Operation after Exit from Deep Software Standby Mode**

### 24.8.5 Setting Oscillation Settling Time after Exit from Deep Software Standby Mode

The WTSTS5 to WTSTS0 bits in DPSWCR should be set as follows:

1. Using a crystal resonator

Specify the WTSTS5 to WTSTS0 bits so that the standby time is at least equal to the oscillation settling time. Table 24.3 shows EXTAL input clock frequencies and the standby time according to WTSTS5 to WTSTS0 settings.

2. Using an external clock

The PLL circuit settling time should be considered. See table 24.3 to set the standby time.

**Table 24.3 Oscillation Settling Time Settings**

WT STS5	WT STS4	WT STS3	WT STS2	WT STS1	WT STS0	Standby Time	EXTAL Input Clock Frequency* (MHz)						Unit													
							18	16	14	12	10	8														
0	0	0	0	0	0	0	Reserved	—	—	—	—	—	—	μs												
						1	Reserved	—	—	—	—	—	—													
						1	0	0	0	0	0	Reserved	—	—	—	—	—									
											1	Reserved	—	—	—	—	—		—							
						1	0	0	0	0	0	0	Reserved	—	—	—	—	—								
												1	64	3.6	4.0	4.6	5.3	6.4		8.0						
												1	0	0	0	0	0	512		28.4	32.0	36.6	42.7	51.2	64.0	
																	1	1024		56.9	64.0	73.1	85.3	102.4	128.0	
												1	0	0	0	0	0	0		2048	113.8	128.0	146.3	170.7	204.8	256.0
																		1		4096	0.23	0.26	0.29	0.34	0.41	0.51
						1	0	0	0	0	0	0	16384	0.91	1.02	1.17	1.37	1.64	2.05							
												1	32768	1.82	2.05	2.34	2.73	3.28	4.10							
1	0	0	0	0	0							0	65536	3.64	4.10	4.68	5.46	6.55	8.19							
												1	131072	7.28	8.19	9.36	10.92	13.11	16.38							
1	0	0	0	0	0							0	262144	14.56	16.38	18.72	21.85	26.21	32.77							
												1	524288	29.13	32.77	37.45	43.69	52.43	65.54							
1	0	0	0	0	0	Reserved	—	—	—	—	—															

**[Legend]**

: Recommended setting when external clock is in use



: Recommended setting when crystal oscillator is in use

Note: \* The oscillation settling time, which includes a period where the oscillation by an oscillator is not stable, depends on the resonator characteristics.  
The above figures are for reference.

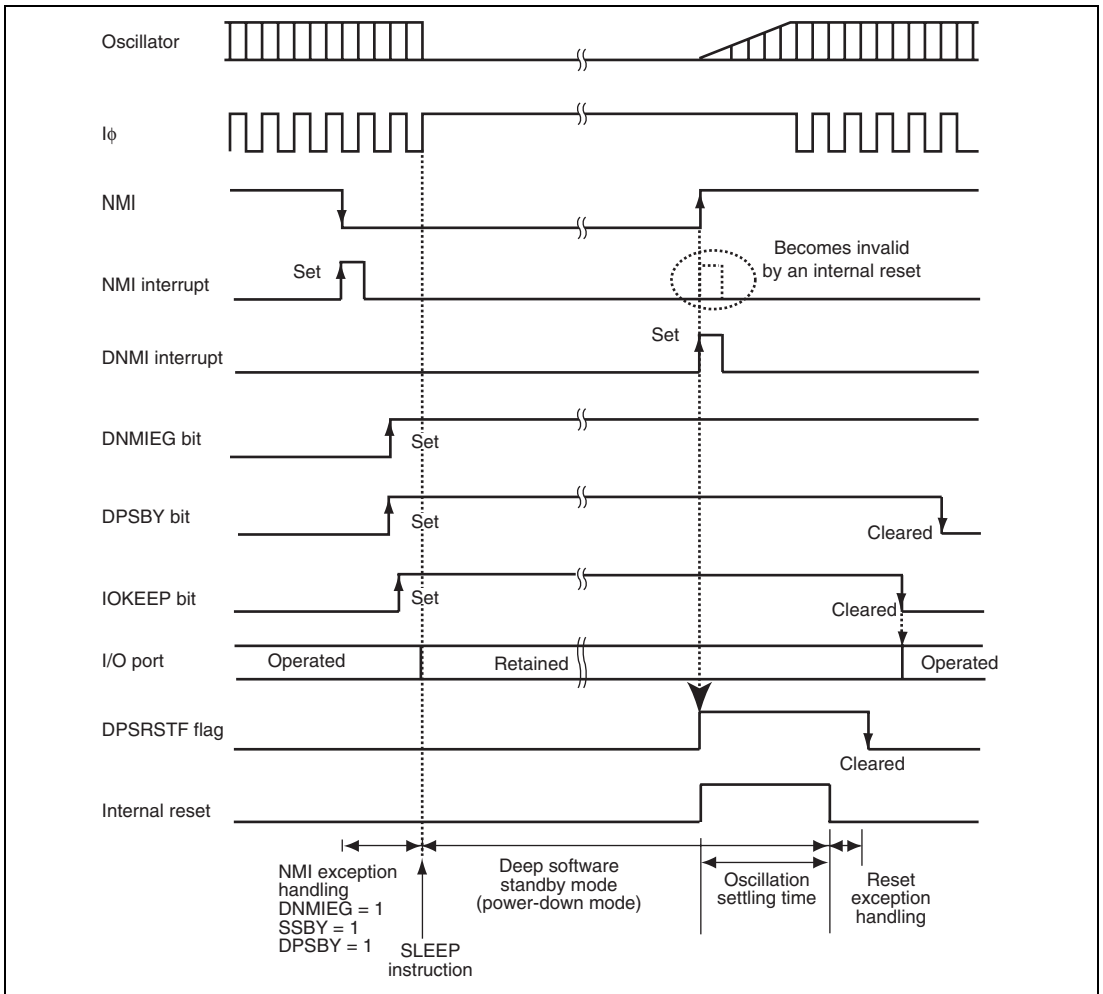
## 24.8.6 Deep Software Standby Mode Application Example

### (1) Transition to and Exit from Deep Software Standby Mode

Figure 24.4 shows an example where the transition to deep software standby mode is initiated by a falling edge on the NMI pin and exit from deep software standby mode is initiated by a rising edge on the NMI pin.

In this example, falling-edge sensing of NMI interrupts has been specified by clearing the NMIEG bit in INTCR to 0 (not shown). After an NMI interrupt has been sensed, rising-edge sensing is specified by setting the DNMIEG bit to 1, the SSBY and DPSBY bits are set to 1, and the transition to deep software standby mode is triggered by execution of a SLEEP instruction.

After that, deep software standby mode is canceled at the rising edge on the NMI pin.



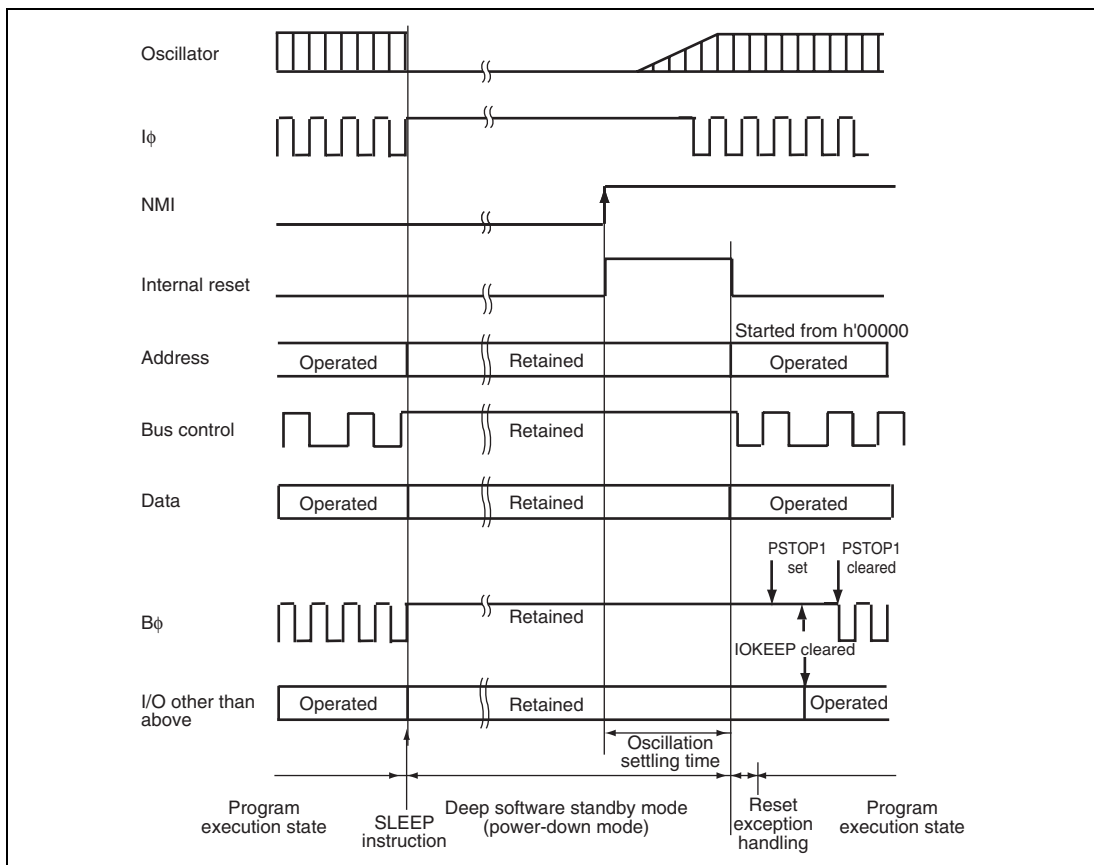
**Figure 24.4 Deep Software Standby Mode Application Example (IOKEEP = 1)**



## (2) Deep Software Standby Mode in External Extended Mode (IOKEEP = 1)

Figure 24.5 shows an example of operations in deep standby mode when the IOKEEP and OPE bits are both set to 1 in external extended mode.

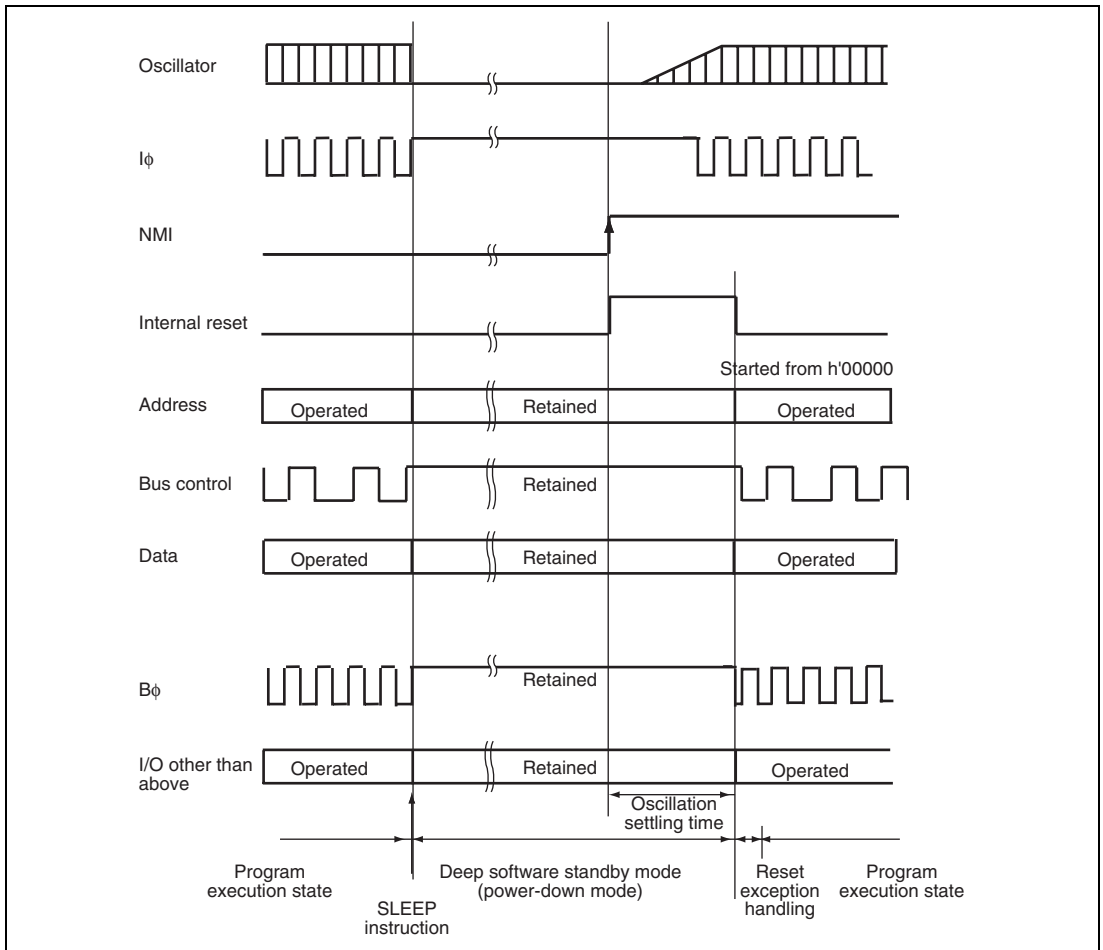
In this example, deep software standby mode is entered with the IOKEEP and OPE bits set to 1, and then exited at the rising edge of the NMI pin. In external extended mode, while the IOKEEP bit is set to 1, retention of the states of pins for the address bus, bus-control signals ( $\overline{CS0}$ ,  $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{HWR}$ , and  $\overline{LWR}$ ), data bus is released after the oscillation settling time has elapsed. For other pins, including the  $B\phi$  output pin, retention is released when the IOKEEP bit is cleared to 0, and then they are set according to the I/O port or peripheral module settings.



**Figure 24.5 Example of Deep Software Standby Mode Operation in External Extended Mode (IOKEEP = OPE = 1)**

### (3) Deep Software Standby Mode in External Extended Mode (IOKEEP = 0)

Figure 24.6 shows an example of operations in deep software standby mode with the IOKEEP bit cleared to 0 and the OPE bit is set to 1 in external extended mode. When the IOKEEP bit is cleared to 0, retention of the states of pins including the address bus, bus-control signals ( $\overline{CS0}$ ,  $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{HWR}$ , and  $\overline{LWR}$ ), data bus, and other pins including  $B\phi$  output is released after the oscillation settling time has elapsed.



**Figure 24.6 Example of Deep Software Standby Mode Operation in External Extended Mode (IOKEEP = 0, OPE = 1)**

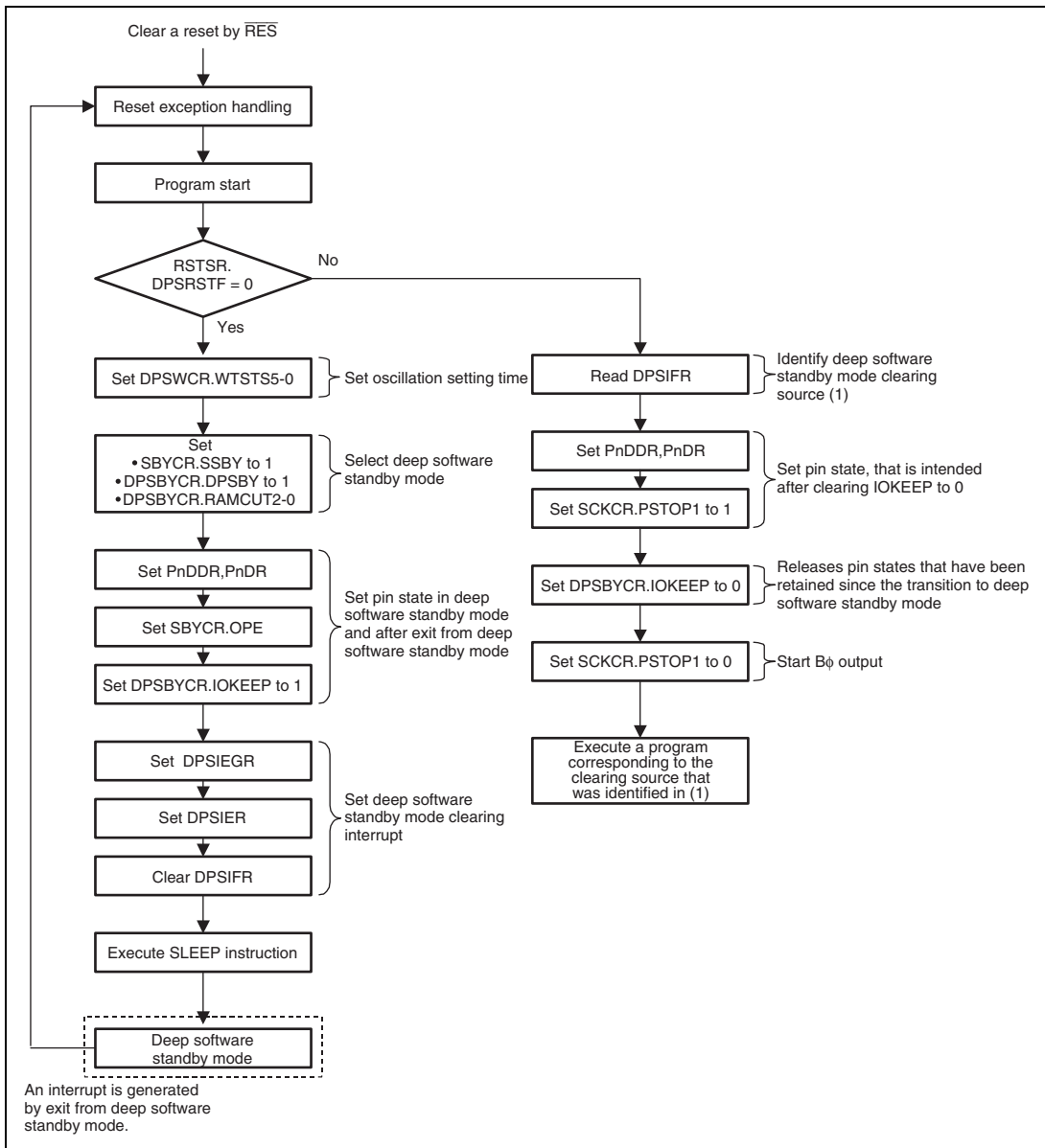
## 24.8.7 Flowchart of Deep Software Standby Mode Operation

Figure 24.7 shows an example of flowchart of deep software standby mode operation. In this example, reading the DPSRSTF bit determines whether a reset was generated by the  $\overline{\text{RES}}$  pin or exit from deep software standby mode, after the reset exception handling was performed.

When a reset was caused by the  $\overline{\text{RES}}$  pin, deep software standby mode is entered after required register settings.

When a reset was caused by exit from deep software standby mode, the IOKEEP bit is cleared after the I/O ports setting. When the IOKEEP bit is cleared, the setting to avoid an undefined state in  $\text{B}\phi$  output is also set.

In this flowchart, an interrupt source is checked by reading DPSIFR before the I/O ports setting. If DPSIFR is read after the I/O ports setting, a source flag may be set without intention by the I/O ports setting.



**Figure 24.7 Flowchart of Deep Software Standby Mode Operation**

## 24.9 Hardware Standby Mode

### 24.9.1 Transition to 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 consumption. Data in the on-chip RAM is not retained because the internal power supply to the on-chip RAM stops. I/O ports are set to the high-impedance state.

Do not change the states of mode pins (MD2 to MD0) while this LSI is in hardware standby mode.

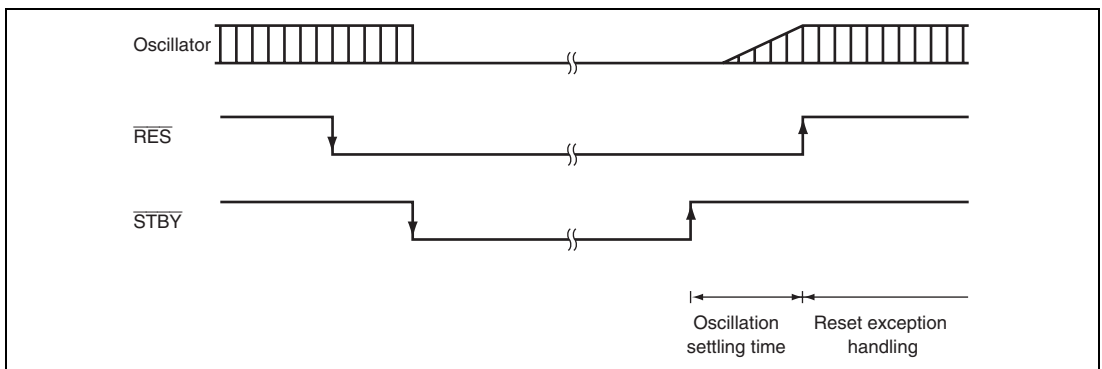
### 24.9.2 Clearing Hardware Standby Mode

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 entered and clock oscillation is started. Ensure that the  $\overline{\text{RES}}$  pin is held low until clock oscillation settles (for details on the oscillation settling time, refer to table 24.2). 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.

### 24.9.3 Hardware Standby Mode Timing

Figure 24.8 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 settling time, then changing the  $\overline{\text{RES}}$  pin from low to high.



**Figure 24.8 Hardware Standby Mode Timing**

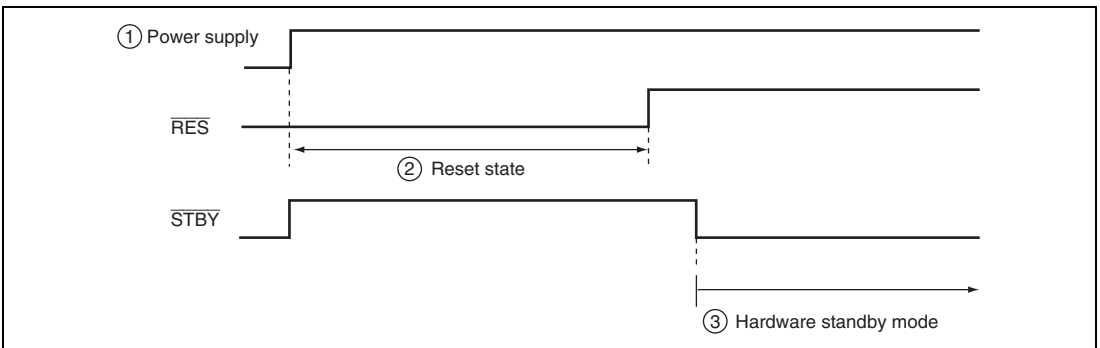
#### 24.9.4 Timing Sequence at Power-On

Figure 24.9 shows the timing sequence at power-on.

At power-on, the  $\overline{\text{RES}}$  pin must be driven low with the  $\overline{\text{STBY}}$  pin driven high for a given time in order to clear the reset state.

To enter hardware standby mode immediately after power-on, drive the  $\overline{\text{STBY}}$  pin low after exiting the reset state.

For details on clearing hardware standby mode, see section 24.9.3, Hardware Standby Mode Timing.



**Figure 24.9 Timing Sequence at Power-On**

## 24.10 Sleep Instruction Exception Handling

Sleep instruction exception handling is the exception handling initiated by the execution of a SLEEP instruction. Sleep instruction exception handling is always accepted while the program is in execution.

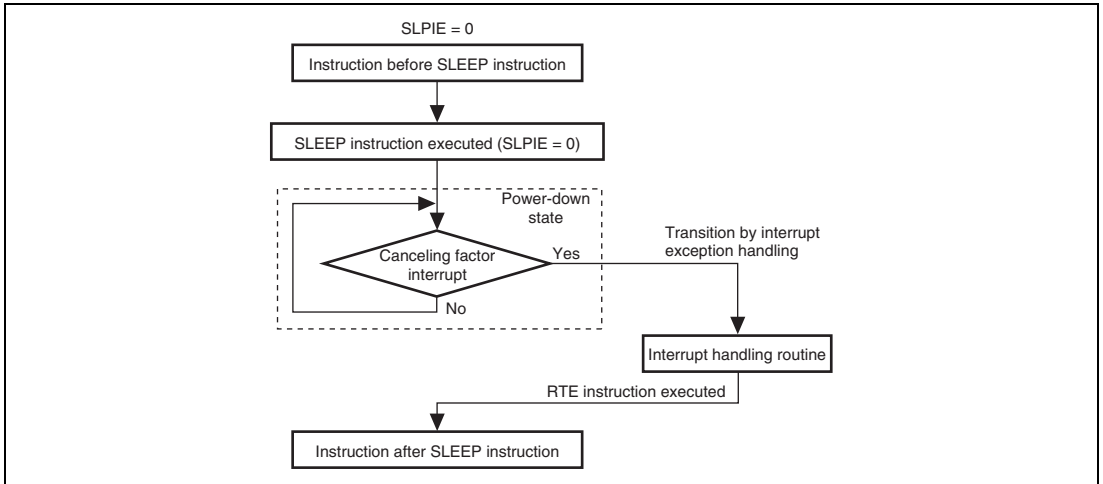
When the SLPIE bit is set to 0, the execution of a SLEEP instruction does not initiate sleep instruction exception handling. Instead, the CPU enters the power-down state. After this, generation of an exception handling request that cancels the power-down state causes the powerdown state to be canceled, after which the CPU starts to handle the exception. When the SLPIE bit is set to 1, sleep instruction exception handling starts after the execution of a SLEEP instruction. Transitions to the power-down state are inhibited when sleep instruction exception handling is initiated, and the CPU immediately starts sleep instruction exception handling.

When a SLEEP instruction is executed while the SLPIE bit is cleared to 0, a transition is made to the power-down state. The power-down state is canceled by a canceling factor interrupt (see figure 24.10).

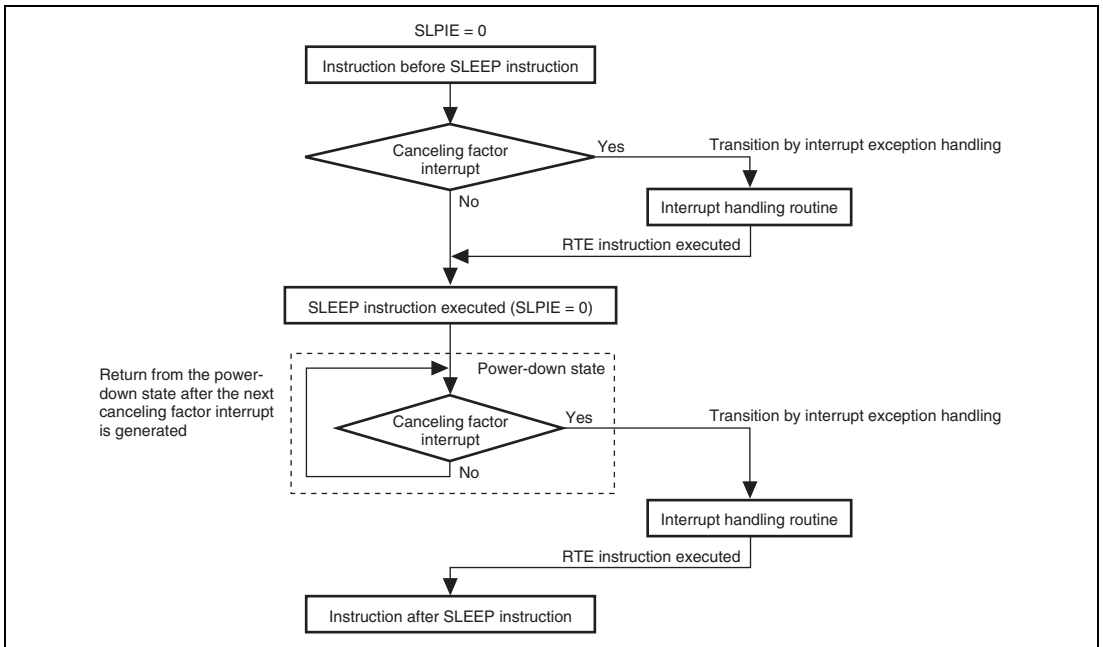
When a canceling factor interrupt is generated immediately before the execution of a SLEEP instruction, exception handling for the interrupt starts. When execution returns from the exception service routine, the SLEEP instruction is executed to enter the power-down state. In this case, the power-down state is not canceled until the next canceling factor interrupt is generated (see figure 24.11).

When the SLPIE bit is set to 1 in the service routine for a canceling factor interrupt so that the execution of a SLEEP instruction will produce sleep instruction exception handling, the operation of the system is as shown in figure 24.12. Even if a canceling factor interrupt is generated immediately before the SLEEP instruction is executed, sleep instruction exception handling is initiated by execution of the SLEEP instruction. Therefore, the CPU executes the instruction that follows the SLEEP instruction after sleep instruction exception and exception service routine without shifting to the power-down state.

When the SLPIE bit is set to 1 to start sleep exception handling, clear the SSBY bit in SBYCR to 0.

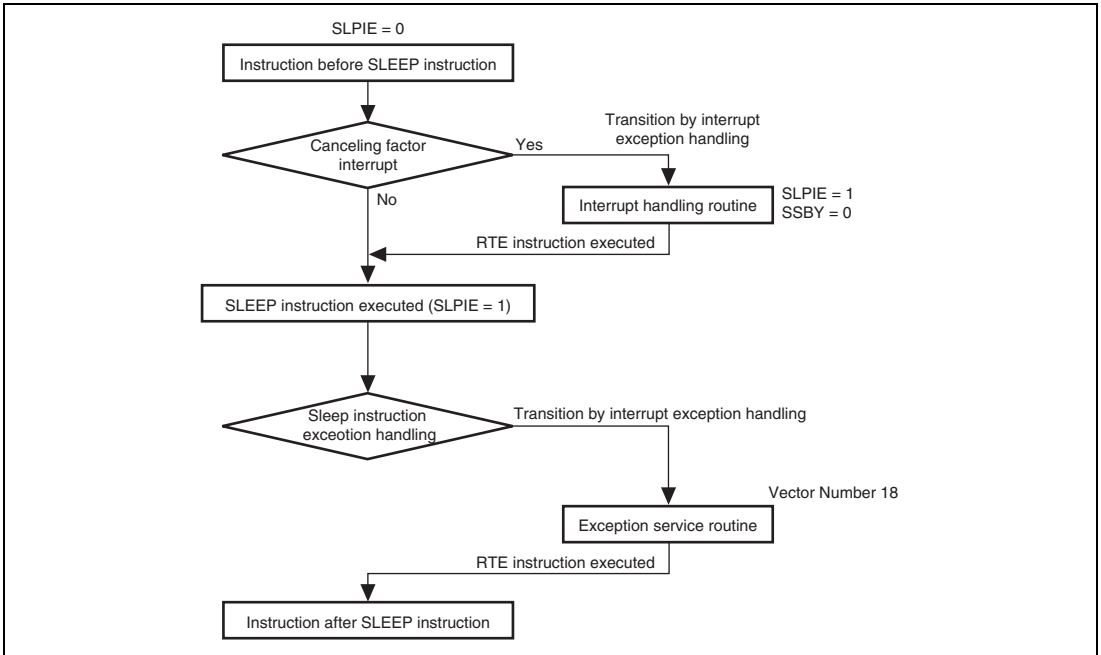


**Figure 24.10 When Canceling Factor Interrupt is Generated after SLEEP Instruction Execution**



**Figure 24.11 When Canceling Factor Interrupt is Generated before SLEEP Instruction Execution (Sleep Instruction Exception Handling Not Initiated)**





**Figure 24.12 When Canceling Factor Interrupt is Generated before SLEEP Instruction Execution (Sleep Instruction Exception Handling Initiated)**

## 24.11 B $\phi$ Clock Output Control

Output of the B $\phi$  clock can be controlled by the PSTOP1 bit in SCKCR, and DDR for the corresponding PA7 pin.

Clearing the PSTOP1 bit to 0 enables the B $\phi$  clock output on the PA7 pin. When bit PSTOP1 is set to 1, the B $\phi$  clock output stops at the end of the bus cycle, and the B $\phi$  clock output goes high. When DDR for the PA7 pin is cleared to 0, the B $\phi$  clock output is disabled and the pin becomes an input port. Tables 24.4 shows the states of the B $\phi$  pin in each processing state.

**Table 24.4  $\phi$  Pin (PA7) State in Each Processing State**

Register Setting Value		Normal Operating Mode	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode		Deep Software Standby Mode		Hardware Standby Mode
DDR	PSTOP1				OPE = 0	OPE = 1	IOKEEP = 0	IOKEEP = 1	
0	x	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
1	0	B $\phi$ output	B $\phi$ output	B $\phi$ output	High	High	High	High	Hi-Z
1	1	High	High	High	High	High	High	High	Hi-Z

[Legend]

x = Don't care

## 24.12 Usage Notes

### 24.12.1 I/O Port Status

In software standby mode or deep software standby mode, the I/O port states are retained. Therefore, there is no reduction in current drawn due to output currents when high-level signals are being output.

### 24.12.2 Current Consumption during Oscillation Settling Standby Period

Current consumption increases during the oscillation settling standby period.

### 24.12.3 Module Stop State of DMAC or DTC

Depending on the operating state of the DMAC and DTC, bits MSTPA13 and MSTPA12 may not be set to 1, respectively. The module stop state setting for the DMAC or DTC should be carried out only when the DMAC or DTC is not activated.

For details, refer to section 9, DMA Controller (DMAC), and section 10, Data Transfer Controller (DTC).

### 24.12.4 On-Chip Peripheral Module Interrupts

Relevant interrupt operations cannot be performed in a module stop state. Consequently, if module stop state 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 a module stop state.

### 24.12.5 Writing to MSTPCRA, MSTPCRB, and MSTPCRC

MSTPCRA, MSTPCRB, and MSTPCRC should only be written to by the CPU.

### 24.12.6 Control of Input Buffers by DIRQnE (n = 3 to 0)

When the input buffers for the P10/ $\overline{\text{IRQ0}}$ -A to P13/ $\overline{\text{IRQ3}}$ -A pins are enabled by setting the DIRQnE bits (n = 3 to 0) in DSPIER to 1, the PnICR settings corresponding to these pins are invalid. Therefore, note that external inputs to these pins, of which states are reflected on the DIRQnF bits, are also input to the interrupt controller, peripheral modules and I/O ports, after the DIRQnE bits (n = 3 to 0) are set to 1

### 24.12.7 Input Buffer Control by DIRQnE (n = 3 to 0)

If a conflict between a transition to deep software standby mode and generation of software standby mode clearing source occurs, a transition to deep software standby mode is not made but the software standby mode clearing sequence is executed. In this case, an interrupt exception handling for the input interrupt starts after the oscillation settling time for software standby mode (set by the STS4 to STS0 bits in SBYCR) has elapsed.

Note that if a conflict between a deep software standby mode transition and NMI interrupt occurs, the NMI interrupt exception handling routine is required.

If a conflict between a deep software standby mode transition and IRQ0 to IRQ15 interrupts occurs, a transition to deep software standby mode can be made without executing the interrupt execution handling by clearing the SSIn bits in SSIER to 0 beforehand.

### 24.12.8 B $\phi$ Output State

B $\phi$  output is undefined for a maximum of one cycle immediately after deep software standby mode is canceled with the IOKEEP bit cleared to 0 or immediately after the IOKEEP bit is cleared after cancellation of deep software standby mode with the IOKEEP bit set to 1.

However, B $\phi$  can be normally output by setting the IOKEEP and PSTOP1 bits. For details, see section 24.8.4, B $\phi$  Operation after Exit from Deep Software Standby Mode.

## Section 25 List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register addresses (address order)
  - Registers are listed from the lower allocation addresses.
  - Registers are classified according to functional modules.
  - The number of Access Cycles indicates the number of states based on the specified reference clock. For details, refer to section 8.5.4, External Bus Interface.
  - Among the internal I/O register area, addresses not listed in the list of registers are undefined or reserved addresses. Undefined and reserved addresses cannot be accessed. Do not access these addresses; otherwise, the operation when accessing these bits and subsequent operations cannot be guaranteed.
2. Register bits
  - Bit configurations of the registers are listed in the same order as the register addresses.
  - Reserved bits are indicated by — in the bit name column.
  - Space in the bit name field indicates that the entire register is allocated to either the counter or data.
  - For the registers of 16 or 32 bits, the MSB is listed first.
  - Byte configuration description order is subject to big endian.
3. Register states in each operating mode
  - Register states are listed in the same order as the register addresses.
  - For the initialized state of each bit, refer to the register description in the corresponding section.
  - The register states shown here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

## 25.1 Register Addresses (Address Order)

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
A/D data register A	ADDRA	16	H'FEA60	A/D	16	3P $\phi$ /3P $\phi$
A/D data register B	ADDRB	16	H'FEA62	A/D	16	3P $\phi$ /3P $\phi$
A/D data register C	ADDRC	16	H'FEA64	A/D	16	3P $\phi$ /3P $\phi$
A/D data register D	ADDRD	16	H'FEA66	A/D	16	3P $\phi$ /3P $\phi$
A/D data register E	ADDRE	16	H'FEA68	A/D	16	3P $\phi$ /3P $\phi$
A/D data register F	ADDRF	16	H'FEA6A	A/D	16	3P $\phi$ /3P $\phi$
A/D data register G	ADDRG	16	H'FEA6C	A/D	16	3P $\phi$ /3P $\phi$
A/D data register H	ADDRH	16	H'FEA6E	A/D	16	3P $\phi$ /3P $\phi$
A/D control/status register	ADCSR	8	H'FEA70	A/D	16	3P $\phi$ /3P $\phi$
A/D control register	ADCR	8	H'FEA71	A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D data register 0	DSADDR0	16	H'FEC00	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D data register 1	DSADDR1	16	H'FEC02	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D data register 2	DSADDR2	16	H'FEC04	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D data register 3	DSADDR3	16	H'FEC06	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D data register 4	DSADDR4	16	H'FEC08	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D data register 5	DSADDR5	16	H'FEC0A	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D offset cancel DAC input 0	DSADOF0	16	H'FEC10	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D offset cancel DAC input 1	DSADOF1	16	H'FEC12	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D offset cancel DAC input 2	DSADOF2	16	H'FEC14	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D offset cancel DAC input 3	DSADOF3	16	H'FEC16	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D control/status register	DSADCSR	16	H'FEC18	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D control register	DSADCR	16	H'FEC1A	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
$\Delta\Sigma$ A/D mode register	DSADMR	8	H'FEC24	$\Delta\Sigma$ A/D	16	3P $\phi$ /3P $\phi$
Break address register AH	BARAH	16	H'FFA00	UBC	16	2I $\phi$ /2I $\phi$
Break address register AL	BARAL	16	H'FFA02	UBC	16	2I $\phi$ /2I $\phi$
Break address mask register AH	BAMRAH	16	H'FFA04	UBC	16	2I $\phi$ /2I $\phi$
Break address mask register AL	BAMRAL	16	H'FFA06	UBC	16	2I $\phi$ /2I $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Break address register BH	BARBH	16	H'FFA08	UBC	16	21 $\phi$ /21 $\phi$
Break address register BL	BARBL	16	H'FFA0A	UBC	16	21 $\phi$ /21 $\phi$
Break address mask register BH	BAMRBH	16	H'FFA0C	UBC	16	21 $\phi$ /21 $\phi$
Break address mask register BL	BAMRBL	16	H'FFA0E	UBC	16	21 $\phi$ /21 $\phi$
Break address register CH	BARCH	16	H'FFA10	UBC	16	21 $\phi$ /21 $\phi$
Break address register CL	BARCL	16	H'FFA12	UBC	16	21 $\phi$ /21 $\phi$
Break address mask register CH	BAMRCH	16	H'FFA14	UBC	16	21 $\phi$ /21 $\phi$
Break address mask register CL	BAMRCL	16	H'FFA16	UBC	16	21 $\phi$ /21 $\phi$
Break address register DH	BARDH	16	H'FFA18	UBC	16	21 $\phi$ /21 $\phi$
Break address register DL	BARDL	16	H'FFA1A	UBC	16	21 $\phi$ /21 $\phi$
Break address mask register DH	BAMRDH	16	H'FFA1C	UBC	16	21 $\phi$ /21 $\phi$
Break address mask register DL	BAMRDL	16	H'FFA1E	UBC	16	21 $\phi$ /21 $\phi$
Break control register A	BRCRA	16	H'FFA28	UBC	16	21 $\phi$ /21 $\phi$
Break control register B	BRCRB	16	H'FFA2C	UBC	16	21 $\phi$ /21 $\phi$
Break control register C	BRCRC	16	H'FFA30	UBC	16	21 $\phi$ /21 $\phi$
Break control register D	BRCRD	16	H'FFA34	UBC	16	21 $\phi$ /21 $\phi$
Timer control register_6	TCR_6	8	H'FFAB0	TMR_6	16	2P $\phi$ /2P $\phi$
Timer control register_7	TCR_7	8	H'FFAB1	TMR_7	16	2P $\phi$ /2P $\phi$
Timer control/status register_6	TCSR_6	8	H'FFAB2	TMR_6	16	2P $\phi$ /2P $\phi$
Timer control/status register_7	TCSR_7	8	H'FFAB3	TMR_7	16	2P $\phi$ /2P $\phi$
Time constant register A_6	TCORA_6	8	H'FFAB4	TMR_6	16	2P $\phi$ /2P $\phi$
Time constant register A_7	TCORA_7	8	H'FFAB5	TMR_7	16	2P $\phi$ /2P $\phi$
Time constant register B_6	TCORB_6	8	H'FFAB6	TMR_6	16	2P $\phi$ /2P $\phi$
Time constant register B_7	TCORB_7	8	H'FFAB7	TMR_7	16	2P $\phi$ /2P $\phi$
Timer counter_6	TCNT_6	8	H'FFAB8	TMR_6	16	2P $\phi$ /2P $\phi$
Timer counter_7	TCNT_7	8	H'FFAB9	TMR_7	16	2P $\phi$ /2P $\phi$
Timer counter control register_6	TCCR_6	8	H'FFABA	TMR_6	16	2P $\phi$ /2P $\phi$
Timer counter control register_7	TCCR_7	8	H'FFABB	TMR_7	16	2P $\phi$ /2P $\phi$
Port 1 data direction register	P1DDR	8	H'FFB80	I/O port	8	2P $\phi$ /2P $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Port 2 data direction register	P2DDR	8	H'FFB81	I/O port	8	2P $\phi$ /2P $\phi$
Port 3 data direction register	P3DDR	8	H'FFB82	I/O port	8	2P $\phi$ /2P $\phi$
Port 6 data direction register	P6DDR	8	H'FFB85	I/O port	8	2P $\phi$ /2P $\phi$
Port A data direction register	PADDR	8	H'FFB89	I/O port	8	2P $\phi$ /2P $\phi$
Port D data direction register	PDDDR	8	H'FFB8C	I/O port	8	2P $\phi$ /2P $\phi$
Port E data direction register	PEDDR	8	H'FFB8D	I/O port	8	2P $\phi$ /2P $\phi$
Port F data direction register	PFDDR	8	H'FFB8E	I/O port	8	2P $\phi$ /2P $\phi$
Port 1 input buffer control register	P1ICR	8	H'FFB90	I/O port	8	2P $\phi$ /2P $\phi$
Port 2 input buffer control register	P2ICR	8	H'FFB91	I/O port	8	2P $\phi$ /2P $\phi$
Port 3 input buffer control register	P3ICR	8	H'FFB92	I/O port	8	2P $\phi$ /2P $\phi$
Port 4 input buffer control register	P4ICR	8	H'FFB93	I/O port	8	2P $\phi$ /2P $\phi$
Port 5 input buffer control register	P5ICR	8	H'FFB94	I/O port	8	2P $\phi$ /2P $\phi$
Port 6 input buffer control register	P6ICR	8	H'FFB95	I/O port	8	2P $\phi$ /2P $\phi$
Port A input buffer control register	PAICR	8	H'FFB99	I/O port	8	2P $\phi$ /2P $\phi$
Port D input buffer control register	PDICR	8	H'FFB9C	I/O port	8	2P $\phi$ /2P $\phi$
Port E input buffer control register	PEICR	8	H'FFB9D	I/O port	8	2P $\phi$ /2P $\phi$
Port F input buffer control register	PFICR	8	H'FFB9E	I/O port	8	2P $\phi$ /2P $\phi$
Port H register	PORTH	8	H'FFBA0	I/O port	8	2P $\phi$ /2P $\phi$
Port I register	PORTI	8	H'FFBA1	I/O port	8	2P $\phi$ /2P $\phi$
Port H data register	PHDR	8	H'FFBA4	I/O port	8	2P $\phi$ /2P $\phi$
Port I data register	PIDR	8	H'FFBA5	I/O port	8	2P $\phi$ /2P $\phi$
Port H data direction register	PHDDR	8	H'FFBA8	I/O port	8	2P $\phi$ /2P $\phi$
Port I data direction register	PIDDR	8	H'FFBA9	I/O port	8	2P $\phi$ /2P $\phi$
Port H input buffer control register	PHICR	8	H'FFBAC	I/O port	8	2P $\phi$ /2P $\phi$
Port I input buffer control register	PIICR	8	H'FFBAD	I/O port	8	2P $\phi$ /2P $\phi$
Port D pull-Up MOS control register	PDPCR	8	H'FFBB4	I/O port	8	2P $\phi$ /2P $\phi$
Port E pull-Up MOS control register	PEPCR	8	H'FFBB5	I/O port	8	2P $\phi$ /2P $\phi$
Port F pull-Up MOS control register	PFPCR	8	H'FFBB6	I/O port	8	2P $\phi$ /2P $\phi$
Port H pull-Up MOS control register	PHPCR	8	H'FFBB8	I/O port	8	2P $\phi$ /2P $\phi$



Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Port I pull-Up MOS control register	PIPCR	8	H'FFBB9	I/O port	8	2P $\phi$ /2P $\phi$
Port 2 open-drain control register	P2ODR	8	H'FFBBC	I/O port	8	2P $\phi$ /2P $\phi$
Port F open-drain control register	PFODR	8	H'FFBBD	I/O port	8	2P $\phi$ /2P $\phi$
Port function control register 0	PFCR0	8	H'FFBC0	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register 1	PFCR1	8	H'FFBC1	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register 2	PFCR2	8	H'FFBC2	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register 4	PFCR4	8	H'FFBC4	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register 6	PFCR6	8	H'FFBC6	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register 7	PFCR7	8	H'FFBC7	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register 9	PFCR9	8	H'FFBC9	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register B	PFCRB	8	H'FFBCB	I/O port	8	2P $\phi$ /3P $\phi$
Port function control register C	PFCRC	8	H'FFBCC	I/O port	8	2P $\phi$ /3P $\phi$
Software standby release IRQ enable register	SSIER	16	H'FFBCE	INTC	8	2P $\phi$ /3P $\phi$
Deep standby backup register 0	DPSBKR0	8	H'FFBF0	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 1	DPSBKR1	8	H'FFBF1	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 2	DPSBKR2	8	H'FFBF2	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 3	DPSBKR3	8	H'FFBF3	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 4	DPSBKR4	8	H'FFBF4	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 5	DPSBKR5	8	H'FFBF5	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 6	DPSBKR6	8	H'FFBF6	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 7	DPSBKR7	8	H'FFBF7	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 8	DPSBKR8	8	H'FFBF8	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 9	DPSBKR9	8	H'FFBF9	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 10	DPSBKR10	8	H'FFBFA	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 11	DPSBKR11	8	H'FFBFB	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 12	DPSBKR12	8	H'FFBFC	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 13	DPSBKR13	8	H'FFBFD	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 14	DPSBKR14	8	H'FFBFE	SYSTEM	8	2I $\phi$ /3I $\phi$
Deep standby backup register 15	DPSBKR15	8	H'FFBFF	SYSTEM	8	2I $\phi$ /3I $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
DMA source address register_0	DSAR_0	32	H'FFC00	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA destination address register_0	DDAR_0	32	H'FFC04	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA offset register_0	DOFR_0	32	H'FFC08	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA transfer count register_0	DTCR_0	32	H'FFC0C	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA block size register_0	DBSR_0	32	H'FFC10	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA mode control register_0	DMDR_0	32	H'FFC14	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA address control register_0	DACR_0	32	H'FFC18	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA source address register_1	DSAR_1	32	H'FFC20	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA destination address register_1	DDAR_1	32	H'FFC24	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA offset register_1	DOFR_1	32	H'FFC28	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA transfer count register_1	DTCR_1	32	H'FFC2C	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA block size register_1	DBSR_1	32	H'FFC30	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA mode control register_1	DMDR_1	32	H'FFC34	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA address control register_1	DACR_1	32	H'FFC38	DMAC_1	16	21 $\phi$ /21 $\phi$
DMA module request select register_0	DMRSR_0	8	H'FFD20	DMAC_0	16	21 $\phi$ /21 $\phi$
DMA module request select register_1	DMRSR_1	8	H'FFD21	DMAC_1	16	21 $\phi$ /21 $\phi$
Interrupt priority register A	IPRA	16	H'FFD40	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register B	IPRB	16	H'FFD42	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register C	IPRC	16	H'FFD44	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register D	IPRD	16	H'FFD46	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register E	IPRE	16	H'FFD48	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register F	IPRF	16	H'FFD4A	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register G	IPRG	16	H'FFD4C	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register H	IPRH	16	H'FFD4E	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register I	IPRI	16	H'FFD50	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register K	IPRK	16	H'FFD54	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register L	IPRL	16	H'FFD56	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register P	IPRP	16	H'FFD5E	INTC	16	21 $\phi$ /31 $\phi$
Interrupt priority register Q	IPRQ	16	H'FFD60	INTC	16	21 $\phi$ /31 $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Interrupt priority register R	IPRR	16	H'FFD62	INTC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
IRQ sense control register H	ISCRH	16	H'FFD68	INTC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
IRQ sense control register L	ISCR L	16	H'FFD6A	INTC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
DTC vector base register	DTCVBR	32	H'FFD80	DTC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Bus width control register	ABWCR	16	H'FFD84	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Access state control register	ASTCR	16	H'FFD86	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Wait control register A	WTCRA	16	H'FFD88	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Wait control register B	WTCRB	16	H'FFD8A	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Read strobe timing control register	RDNCR	16	H'FFD8C	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
$\overline{\text{CS}}$ assertion period control register	CSACR	16	H'FFD8E	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Idle control register	IDLCR	16	H'FFD90	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Bus control register 1	BCR1	16	H'FFD92	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Bus control register 2	BCR2	8	H'FFD94	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Endian control register	ENDIANCR	8	H'FFD95	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
SRAM mode control register	SRAMCR	16	H'FFD98	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Burst ROM interface control register	BROMCR	16	H'FFD9A	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Address/data multiplexed I/O control register	MPXCR	16	H'FFD9C	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
RAM emulation register	RAMER	8	H'FFD9E	BSC	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Mode control register	MDCR	16	H'FFDC0	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
System control register	SYSCR	16	H'FFDC2	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
System clock control register	SCKCR	16	H'FFDC4	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Standby control register	SBYCR	16	H'FFDC6	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Module stop control register A	MSTPCRA	16	H'FFDC8	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Module stop control register B	MSTPCRB	16	H'FFDCA	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Module stop control register C	MSTPCRC	16	H'FFDCC	SYSTEM	16	2t <sub>φ</sub> /3t <sub>φ</sub>
Flash code control/status register	FCCS	8	H'FFDE8	FLASH	16	2t <sub>φ</sub> /2t <sub>φ</sub>
Flash program code select register	FPCS	8	H'FFDE9	FLASH	16	2t <sub>φ</sub> /2t <sub>φ</sub>
Flash erase code select register	FECS	8	H'FFDEA	FLASH	16	2t <sub>φ</sub> /2t <sub>φ</sub>
Flash key code register	FKEY	8	H'FFDEC	FLASH	16	2t <sub>φ</sub> /2t <sub>φ</sub>

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Flash MAT select register	FMATS	8	H'FFDED	FLASH	16	21 $\phi$ /21 $\phi$
Flash transfer destination address register	FTDAR	8	H'FFDEE	FLASH	16	21 $\phi$ /21 $\phi$
Deep standby control register	DPSBYCR	8	H'FFE70	SYSTEM	8	21 $\phi$ /31 $\phi$
Deep standby wait control register	DPSWCR	8	H'FFE71	SYSTEM	8	21 $\phi$ /31 $\phi$
Deep standby interrupt enable register	DPSIER	8	H'FFE72	SYSTEM	8	21 $\phi$ /31 $\phi$
Deep standby interrupt flag register	DPSIFR	8	H'FFE73	SYSTEM	8	21 $\phi$ /31 $\phi$
Deep standby interrupt edge register	DPSIEGR	8	H'FFE74	SYSTEM	8	21 $\phi$ /31 $\phi$
Reset status register	RSTSR	8	H'FFE75	SYSTEM	8	21 $\phi$ /31 $\phi$
Serial extended mode register_2	SEMR_2	8	H'FFE84	SCI_2	8	2P $\phi$ /2P $\phi$
Serial mode register_3	SMR_3	8	H'FFE88	SCI_3	8	2P $\phi$ /2P $\phi$
Bit rate register_3	BRR_3	8	H'FFE89	SCI_3	8	2P $\phi$ /2P $\phi$
Serial control register_3	SCR_3	8	H'FFE8A	SCI_3	8	2P $\phi$ /2P $\phi$
Transmit data register_3	TDR_3	8	H'FFE8B	SCI_3	8	2P $\phi$ /2P $\phi$
Serial status register_3	SSR_3	8	H'FFE8C	SCI_3	8	2P $\phi$ /2P $\phi$
Receive data register_3	RDR_3	8	H'FFE8D	SCI_3	8	2P $\phi$ /2P $\phi$
Smart card mode register_3	SCMR_3	8	H'FFE8E	SCI_3	8	2P $\phi$ /2P $\phi$
Serial mode register_4	SMR_4	8	H'FFE90	SCI_4	8	2P $\phi$ /2P $\phi$
Bit rate register_4	BRR_4	8	H'FFE91	SCI_4	8	2P $\phi$ /2P $\phi$
Serial control register_4	SCR_4	8	H'FFE92	SCI_4	8	2P $\phi$ /2P $\phi$
Transmit data register_4	TDR_4	8	H'FFE93	SCI_4	8	2P $\phi$ /2P $\phi$
Serial status register_4	SSR_4	8	H'FFE94	SCI_4	8	2P $\phi$ /2P $\phi$
Receive data register_4	RDR_4	8	H'FFE95	SCI_4	8	2P $\phi$ /2P $\phi$
Smart card mode register_4	SCMR_4	8	H'FFE96	SCI_4	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus control register A_0	ICCRA_0	8	H'FFEB0	IIC2_0	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus control register B_0	ICCRB_0	8	H'FFEB1	IIC2_0	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus mode register_0	ICMR_0	8	H'FFEB2	IIC2_0	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus interrupt enable register_0	ICIER_0	8	H'FFEB3	IIC2_0	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus status register_0	ICSR_0	8	H'FFEB4	IIC2_0	8	2P $\phi$ /2P $\phi$
Slave address register_0	SAR_0	8	H'FFEB5	IIC2_0	8	2P $\phi$ /2P $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
I <sup>2</sup> C bus transmit data register_0	ICDRT_0	8	H'FFEB6	IIC2_0	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus receive data register_0	ICDRR_0	8	H'FFEB7	IIC2_0	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus control register A_1	ICCRA_1	8	H'FFEB8	IIC2_1	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus control register B_1	ICCRB_1	8	H'FFEB9	IIC2_1	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus mode register_1	ICMR_1	8	H'FFEBA	IIC2_1	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus interrupt enable register_1	ICIER_1	8	H'FFEBB	IIC2_1	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus status register_1	ICSR_1	8	H'FFEBC	IIC2_1	8	2P $\phi$ /2P $\phi$
Slave address register_1	SAR_1	8	H'FFEBD	IIC2_1	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus transmit data register_1	ICDRT_1	8	H'FFEBE	IIC2_1	8	2P $\phi$ /2P $\phi$
I <sup>2</sup> C bus receive data register_1	ICDRR_1	8	H'FFEBF	IIC2_1	8	2P $\phi$ /2P $\phi$
Timer control register_2	TCR_2	8	H'FFEC0	TMR_2	16	2P $\phi$ /2P $\phi$
Timer control register_3	TCR_3	8	H'FFEC1	TMR_3	16	2P $\phi$ /2P $\phi$
Timer control/status register_2	TCSR_2	8	H'FFEC2	TMR_2	16	2P $\phi$ /2P $\phi$
Timer control/status register_3	TCSR_3	8	H'FFEC3	TMR_3	16	2P $\phi$ /2P $\phi$
Time constant register A_2	TCORA_2	8	H'FFEC4	TMR_2	16	2P $\phi$ /2P $\phi$
Time constant register A_3	TCORA_3	8	H'FFEC5	TMR_3	16	2P $\phi$ /2P $\phi$
Time constant register B_2	TCORB_2	8	H'FFEC6	TMR_2	16	2P $\phi$ /2P $\phi$
Time constant register B_3	TCORB_3	8	H'FFEC7	TMR_3	16	2P $\phi$ /2P $\phi$
Timer counter_2	TCNT_2	8	H'FFEC8	TMR_2	16	2P $\phi$ /2P $\phi$
Timer counter_3	TCNT_3	8	H'FFEC9	TMR_3	16	2P $\phi$ /2P $\phi$
Timer counter control register_2	TCCR_2	8	H'FFECA	TMR_2	16	2P $\phi$ /2P $\phi$
Timer counter control register_3	TCCR_3	8	H'FFECB	TMR_3	16	2P $\phi$ /2P $\phi$
Timer control register_4	TCR_4	8	H'FFED0	TMR_4	16	2P $\phi$ /2P $\phi$
Timer control register_5	TCR_5	8	H'FFED1	TMR_5	16	2P $\phi$ /2P $\phi$
Timer control/status register_4	TCSR_4	8	H'FFED2	TMR_4	16	2P $\phi$ /2P $\phi$
Timer control/status register_5	TCSR_5	8	H'FFED3	TMR_5	16	2P $\phi$ /2P $\phi$
Time constant register A_4	TCORA_4	8	H'FFED4	TMR_4	16	2P $\phi$ /2P $\phi$
Time constant register A_5	TCORA_5	8	H'FFED5	TMR_5	16	2P $\phi$ /2P $\phi$
Time constant register B_4	TCORB_4	8	H'FFED6	TMR_4	16	2P $\phi$ /2P $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Time constant register B_5	TCORB_5	8	H'FFED7	TMR_5	16	2P $\phi$ /2P $\phi$
Timer counter_4	TCNT_4	8	H'FFED8	TMR_4	16	2P $\phi$ /2P $\phi$
Timer counter_5	TCNT_5	8	H'FFED9	TMR_5	16	2P $\phi$ /2P $\phi$
Timer counter control register_4	TCCR_4	8	H'FFEDA	TMR_4	16	2P $\phi$ /2P $\phi$
Timer counter control register_5	TCCR_5	8	H'FFEDB	TMR_5	16	2P $\phi$ /2P $\phi$
Timer control register_4	TCR_4	8	H'FFEE0	TPU_4	16	2P $\phi$ /2P $\phi$
Timer mode register_4	TMDR_4	8	H'FFEE1	TPU_4	16	2P $\phi$ /2P $\phi$
Timer I/O control register_4	TIOR_4	8	H'FFEE2	TPU_4	16	2P $\phi$ /2P $\phi$
Timer interrupt enable register_4	TIER_4	8	H'FFEE4	TPU_4	16	2P $\phi$ /2P $\phi$
Timer status register_4	TSR_4	8	H'FFEE5	TPU_4	16	2P $\phi$ /2P $\phi$
Timer counter_4	TCNT_4	16	H'FFEE6	TPU_4	16	2P $\phi$ /2P $\phi$
Timer general register A_4	TGRA_4	16	H'FFEE8	TPU_4	16	2P $\phi$ /2P $\phi$
Timer general register B_4	TGRB_4	16	H'FFEEA	TPU_4	16	2P $\phi$ /2P $\phi$
Timer control register_5	TCR_5	8	H'FFEF0	TPU_5	16	2P $\phi$ /2P $\phi$
Timer mode register_5	TMDR_5	8	H'FFEF1	TPU_5	16	2P $\phi$ /2P $\phi$
Timer I/O control register_5	TIOR_5	8	H'FFEF2	TPU_5	16	2P $\phi$ /2P $\phi$
Timer interrupt enable register_5	TIER_5	8	H'FFEF4	TPU_5	16	2P $\phi$ /2P $\phi$
Timer status register_5	TSR_5	8	H'FFEF5	TPU_5	16	2P $\phi$ /2P $\phi$
Timer counter_5	TCNT_5	16	H'FFEF6	TPU_5	16	2P $\phi$ /2P $\phi$
Timer general register A_5	TGRA_5	16	H'FFEF8	TPU_5	16	2P $\phi$ /2P $\phi$
Timer general register B_5	TGRB_5	16	H'FFEFA	TPU_5	16	2P $\phi$ /2P $\phi$
DTC enable register A	DTCERA	16	H'FFF20	DTC	16	2I $\phi$ /3I $\phi$
DTC enable register B	DTCERB	16	H'FFF22	DTC	16	2I $\phi$ /3I $\phi$
DTC enable register C	DTCERC	16	H'FFF24	DTC	16	2I $\phi$ /3I $\phi$
DTC enable register D	DTCERD	16	H'FFF26	DTC	16	2I $\phi$ /3I $\phi$
DTC enable register E	DTCERE	16	H'FFF28	DTC	16	2I $\phi$ /3I $\phi$
DTC enable register F	DTCERF	16	H'FFF2A	DTC	16	2I $\phi$ /3I $\phi$
DTC enable register G	DTCERG	16	H'FFF2C	DTC	16	2I $\phi$ /3I $\phi$
DTC control register	DTCCR	8	H'FFF30	DTC	16	2I $\phi$ /3I $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Interrupt control register	INTCR	8	H'FFF32	INTC	16	21 $\phi$ /31 $\phi$
CPU priority control register	CPUPCR	8	H'FFF33	INTC	16	21 $\phi$ /31 $\phi$
IRQ enable register	IER	16	H'FFF34	INTC	16	21 $\phi$ /31 $\phi$
IRQ status register	ISR	16	H'FFF36	INTC	16	21 $\phi$ /31 $\phi$
Port 1 register	PORT1	8	H'FFF40	I/O port	8	2P $\phi$ /-
Port 2 register	PORT2	8	H'FFF41	I/O port	8	2P $\phi$ /-
Port 3 register	PORT3	8	H'FFF42	I/O port	8	2P $\phi$ /-
Port 4 register	PORT4	8	H'FFF43	I/O port	8	2P $\phi$ /-
Port 5 register	PORT5	8	H'FFF44	I/O port	8	2P $\phi$ /-
Port 6 register	PORT6	8	H'FFF45	I/O port	8	2P $\phi$ /-
Port A register	PORTA	8	H'FFF49	I/O port	8	2P $\phi$ /-
Port D register	PORTD	8	H'FFF4C	I/O port	8	2P $\phi$ /-
Port E register	PORTE	8	H'FFF4D	I/O port	8	2P $\phi$ /-
Port F register	PORTF	8	H'FFF4E	I/O port	8	2P $\phi$ /-
Port 1 data register	P1DR	8	H'FFF50	I/O port	8	2P $\phi$ /2P $\phi$
Port 2 data register	P2DR	8	H'FFF51	I/O port	8	2P $\phi$ /2P $\phi$
Port 3 data register	P3DR	8	H'FFF52	I/O port	8	2P $\phi$ /2P $\phi$
Port 6 data register	P6DR	8	H'FFF55	I/O port	8	2P $\phi$ /2P $\phi$
Port A data register	PADR	8	H'FFF59	I/O port	8	2P $\phi$ /2P $\phi$
Port D data register	PDDR	8	H'FFF5C	I/O port	8	2P $\phi$ /2P $\phi$
Port E data register	PEDR	8	H'FFF5D	I/O port	8	2P $\phi$ /2P $\phi$
Port F data register	PFDR	8	H'FFF5E	I/O port	8	2P $\phi$ /2P $\phi$
Serial mode register_2	SMR_2	8	H'FFF60	SCI_2	8	2P $\phi$ /2P $\phi$
Bit rate register_2	BRR_2	8	H'FFF61	SCI_2	8	2P $\phi$ /2P $\phi$
Serial control register_2	SCR_2	8	H'FFF62	SCI_2	8	2P $\phi$ /2P $\phi$
Transmit data register_2	TDR_2	8	H'FFF63	SCI_2	8	2P $\phi$ /2P $\phi$
Serial status register_2	SSR_2	8	H'FFF64	SCI_2	8	2P $\phi$ /2P $\phi$
Receive data register_2	RDR_2	8	H'FFF65	SCI_2	8	2P $\phi$ /2P $\phi$
Smart card mode register_2	SCMR_2	8	H'FFF66	SCI_2	8	2P $\phi$ /2P $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
D/A data register 0	DADR0	8	H'FFF68	D/A	8	2P $\phi$ /2P $\phi$
D/A data register 1	DADR1	8	H'FFF69	D/A	8	2P $\phi$ /2P $\phi$
D/A control register 01	DACR01	8	H'FFF6A	D/A	8	2P $\phi$ /2P $\phi$
PPG output control register	PCR	8	H'FFF76	PPG	8	2P $\phi$ /2P $\phi$
PPG output mode register	PMR	8	H'FFF77	PPG	8	2P $\phi$ /2P $\phi$
Next data enable register H	NDERH	8	H'FFF78	PPG	8	2P $\phi$ /2P $\phi$
Next data enable register L	NDERL	8	H'FFF79	PPG	8	2P $\phi$ /2P $\phi$
Output data register H	PODRH	8	H'FFF7A	PPG	8	2P $\phi$ /2P $\phi$
Output data register L	PODRL	8	H'FFF7B	PPG	8	2P $\phi$ /2P $\phi$
Next data register H*	NDRH	8	H'FFF7C	PPG	8	2P $\phi$ /2P $\phi$
Next data register L*	NDRL	8	H'FFF7D	PPG	8	2P $\phi$ /2P $\phi$
Next data register H*	NDRH	8	H'FFF7E	PPG	8	2P $\phi$ /2P $\phi$
Next data register L*	NDRL	8	H'FFF7F	PPG	8	2P $\phi$ /2P $\phi$
Serial mode register_0	SMR_0	8	H'FFF80	SCI_0	8	2P $\phi$ /2P $\phi$
Bit rate register_0	BRR_0	8	H'FFF81	SCI_0	8	2P $\phi$ /2P $\phi$
Serial control register_0	SCR_0	8	H'FFF82	SCI_0	8	2P $\phi$ /2P $\phi$
Transmit data register_0	TDR_0	8	H'FFF83	SCI_0	8	2P $\phi$ /2P $\phi$
Serial status register_0	SSR_0	8	H'FFF84	SCI_0	8	2P $\phi$ /2P $\phi$
Receive data register_0	RDR_0	8	H'FFF85	SCI_0	8	2P $\phi$ /2P $\phi$
Smart card mode register_0	SCMR_0	8	H'FFF86	SCI_0	8	2P $\phi$ /2P $\phi$
Serial mode register_1	SMR_1	8	H'FFF88	SCI_1	8	2P $\phi$ /2P $\phi$
Bit rate register_1	BRR_1	8	H'FFF89	SCI_1	8	2P $\phi$ /2P $\phi$
Serial control register_1	SCR_1	8	H'FFF8A	SCI_1	8	2P $\phi$ /2P $\phi$
Transmit data register_1	TDR_1	8	H'FFF8B	SCI_1	8	2P $\phi$ /2P $\phi$
Serial status register_1	SSR_1	8	H'FFF8C	SCI_1	8	2P $\phi$ /2P $\phi$
Receive data register_1	RDR_1	8	H'FFF8D	SCI_1	8	2P $\phi$ /2P $\phi$
Smart card mode register_1	SCMR_1	8	H'FFF8E	SCI_1	8	2P $\phi$ /2P $\phi$
Timer control/status register	TCSR	8	H'FFFA4	WDT		2P $\phi$ /3P $\phi$
Timer counter	TCNT	8	H'FFFA5	WDT		2P $\phi$ /3P $\phi$



Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Reset control/status register	RSTCSR	8	H'FFFA7	WDT		2P $\phi$ /3P $\phi$
Timer control register_0	TCR_0	8	H'FFFB0	TMR_0	16	2P $\phi$ /2P $\phi$
Timer control register_1	TCR_1	8	H'FFFB1	TMR_1	16	2P $\phi$ /2P $\phi$
Timer control/status register_0	TCSR_0	8	H'FFFB2	TMR_0	16	2P $\phi$ /2P $\phi$
Timer control/status register_1	TCSR_1	8	H'FFFB3	TMR_1	16	2P $\phi$ /2P $\phi$
Time constant register A_0	TCORA_0	8	H'FFFB4	TMR_0	16	2P $\phi$ /2P $\phi$
Time constant register A_1	TCORA_1	8	H'FFFB5	TMR_1	16	2P $\phi$ /2P $\phi$
Time constant register B_0	TCORB_0	8	H'FFFB6	TMR_0	16	2P $\phi$ /2P $\phi$
Time constant register B_1	TCORB_1	8	H'FFFB7	TMR_1	16	2P $\phi$ /2P $\phi$
Timer counter_0	TCNT_0	8	H'FFFB8	TMR_0	16	2P $\phi$ /2P $\phi$
Timer counter_1	TCNT_1	8	H'FFFB9	TMR_1	16	2P $\phi$ /2P $\phi$
Timer counter control register_0	TCCR_0	8	H'FFFB8A	TMR_0	16	2P $\phi$ /2P $\phi$
Timer counter control register_1	TCCR_1	8	H'FFFB8B	TMR_1	16	2P $\phi$ /2P $\phi$
Timer start register	TSTR	8	H'FFFB8C	TPU	16	2P $\phi$ /2P $\phi$
Timer synchronous register	TSYR	8	H'FFFB8D	TPU	16	2P $\phi$ /2P $\phi$
Timer control register_0	TCR_0	8	H'FFFC0	TPU_0	16	2P $\phi$ /2P $\phi$
Timer mode register_0	TMDR_0	8	H'FFFC1	TPU_0	16	2P $\phi$ /2P $\phi$
Timer I/O control register H_0	TIORH_0	8	H'FFFC2	TPU_0	16	2P $\phi$ /2P $\phi$
Timer I/O control register L_0	TIORL_0	8	H'FFFC3	TPU_0	16	2P $\phi$ /2P $\phi$
Timer interrupt enable register_0	TIER_0	8	H'FFFC4	TPU_0	16	2P $\phi$ /2P $\phi$
Timer status register_0	TSR_0	8	H'FFFC5	TPU_0	16	2P $\phi$ /2P $\phi$
Timer counter_0	TCNT_0	16	H'FFFC6	TPU_0	16	2P $\phi$ /2P $\phi$
Timer general register A_0	TGRA_0	16	H'FFFC8	TPU_0	16	2P $\phi$ /2P $\phi$
Timer general register B_0	TGRB_0	16	H'FFFC9	TPU_0	16	2P $\phi$ /2P $\phi$
Timer general register C_0	TGRC_0	16	H'FFFCB	TPU_0	16	2P $\phi$ /2P $\phi$
Timer general register D_0	TGRD_0	16	H'FFFCB	TPU_0	16	2P $\phi$ /2P $\phi$
Timer control register_1	TCR_1	8	H'FFFD0	TPU_1	16	2P $\phi$ /2P $\phi$
Timer mode register_1	TMDR_1	8	H'FFFD1	TPU_1	16	2P $\phi$ /2P $\phi$
Timer I/O control register_1	TIOR_1	8	H'FFFD2	TPU_1	16	2P $\phi$ /2P $\phi$

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles (Read/Write)
Timer interrupt enable register_1	TIER_1	8	H'FFFD4	TPU_1	16	2P $\phi$ /2P $\phi$
Timer status register_1	TSR_1	8	H'FFFD5	TPU_1	16	2P $\phi$ /2P $\phi$
Timer counter_1	TCNT_1	16	H'FFFD6	TPU_1	16	2P $\phi$ /2P $\phi$
Timer general register A_1	TGRA_1	16	H'FFFD8	TPU_1	16	2P $\phi$ /2P $\phi$
Timer general register B_1	TGRB_1	16	H'FFFDA	TPU_1	16	2P $\phi$ /2P $\phi$
Timer control register_2	TCR_2	8	H'FFFE0	TPU_2	16	2P $\phi$ /2P $\phi$
Timer mode register_2	TMDR_2	8	H'FFFE1	TPU_2	16	2P $\phi$ /2P $\phi$
Timer I/O control register_2	TIOR_2	8	H'FFFE2	TPU_2	16	2P $\phi$ /2P $\phi$
Timer interrupt enable register_2	TIER_2	8	H'FFFE4	TPU_2	16	2P $\phi$ /2P $\phi$
Timer status register_2	TSR_2	8	H'FFFE5	TPU_2	16	2P $\phi$ /2P $\phi$
Timer counter_2	TCNT_2	16	H'FFFE6	TPU_2	16	2P $\phi$ /2P $\phi$
Timer general register A_2	TGRA_2	16	H'FFFE8	TPU_2	16	2P $\phi$ /2P $\phi$
Timer general register B_2	TGRB_2	16	H'FFFEA	TPU_2	16	2P $\phi$ /2P $\phi$
Timer control register_3	TCR_3	8	H'FFFF0	TPU_3	16	2P $\phi$ /2P $\phi$
Timer mode register_3	TMDR_3	8	H'FFFF1	TPU_3	16	2P $\phi$ /2P $\phi$
Timer I/O control register H_3	TIORH_3	8	H'FFFF2	TPU_3	16	2P $\phi$ /2P $\phi$
Timer I/O control register L_3	TIORL_3	8	H'FFFF3	TPU_3	16	2P $\phi$ /2P $\phi$
Timer interrupt enable register_3	TIER_3	8	H'FFFF4	TPU_3	16	2P $\phi$ /2P $\phi$
Timer status register_3	TSR_3	8	H'FFFF5	TPU_3	16	2P $\phi$ /2P $\phi$
Timer counter_3	TCNT_3	16	H'FFFF6	TPU_3	16	2P $\phi$ /2P $\phi$
Timer general register A_3	TGRA_3	16	H'FFFF8	TPU_3	16	2P $\phi$ /2P $\phi$
Timer general register B_3	TGRB_3	16	H'FFFFA	TPU_3	16	2P $\phi$ /2P $\phi$
Timer general register C_3	TGRC_3	16	H'FFFFC	TPU_3	16	2P $\phi$ /2P $\phi$
Timer general register D_3	TGRD_3	16	H'FFFFE	TPU_3	16	2P $\phi$ /2P $\phi$

Note: \* When the same output trigger is specified for pulse output groups 2 and 3 by the PCR setting, the NDRH address is H'FFF7C. When different output triggers are specified, the NDRH addresses for pulse output groups 2 and 3 are H'FFF7E and H'FFF7C, respectively. Similarly, when the same output trigger is specified for pulse output groups 0 and 1 by the PCR setting, the NDRL address is H'FFF7D. When different output triggers are specified, the NDRL addresses for pulse output groups 0 and 1 are H'FFF7F and H'FFF7D, respectively.

## 25.2 Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, and 16-bit and 32-bit registers are shown as 2 or 4 lines, respectively.

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
ADDRA									A/D
ADDRB									
ADDRC									
ADDRD									
ADDRE									
ADDRF									
ADDRG									
ADDRH									
ADCSR	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0	
ADCR	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	ADSTCLR	EXTRGS	
DSADDR0									ΔΣA/D
DSADDR1									
DSADDR2									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DSADDR3									ΔΣA/D
DSADDR4									
DSADDR5									
DSADOF0									
DSADOF1									
DSADOF2									
DSADOF3									
DSADCSR	ADF	ADIE	ADST	—	SCANE	—	TRGS1	TRGS0	
	—	—	CH5	CH4	CH3	CH2	CH1	CH0	
DSADCR	CKS	—	GAIN1	GAIN0	—	—	—	—	
	DSE	—	—	—	—	—	—	—	
DSADMR	BIASE	—	—	—	—	ACK2	ACK1	ACK0	
BARAH	BARA31	BARA30	BARA29	BARA28	BARA27	BARA26	BARA25	BARA24	UBC
	BARA23	BARA22	BARA21	BARA20	BARA19	BARA18	BARA17	BARA16	
BARAL	BARA15	BARA14	BARA13	BARA12	BARA11	BARA10	BARA9	BARA8	
	BARA7	BARA6	BARA5	BARA4	BARA3	BARA2	BARA1	BARA0	
BAMRAH	BAMRA31	BAMRA30	BAMRA29	BAMRA28	BAMRA27	BAMRA26	BAMRA25	BAMRA24	
	BAMRA23	BAMRA22	BAMRA21	BAMRA20	BAMRA19	BAMRA18	BAMRA17	BAMRA16	
BAMRAL	BAMRA15	BAMRA14	BAMRA13	BAMRA12	BAMRA11	BAMRA10	BAMRA9	BAMRA8	
	BAMRA7	BAMRA6	BAMRA5	BAMRA4	BAMRA3	BAMRA2	BAMRA1	BAMRA0	
BARBH	BARB31	BARB30	BARB29	BARB28	BARB27	BARB26	BARB25	BARB24	
	BARB23	BARB22	BARB21	BARB20	BARB19	BARB18	BARB17	BARB16	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
BARBL	BARB15	BARB14	BARB13	BARB12	BARB11	BARB10	BARB9	BARB8	UBC
	BARB7	BARB6	BARB5	BARB4	BARB3	BARB2	BARB1	BARB0	
BAMRBH	BAMRB31	BAMRB30	BAMRB29	BAMRB28	BAMRB27	BAMRB26	BAMRB25	BAMRB24	
	BAMRB23	BAMRB22	BAMRB21	BAMRB20	BAMRB19	BAMRB18	BAMRB17	BAMRB16	
BAMRBL	BAMRB15	BAMRB14	BAMRB13	BAMRB12	BAMRB11	BAMRB10	BAMRB9	BAMRB8	
	BAMRB7	BAMRB6	BAMRB5	BAMRB4	BAMRB3	BAMRB2	BAMRB1	BAMRB0	
BARCH	BARC31	BARC30	BARC29	BARC28	BARC27	BARC26	BARC25	BARC24	
	BARC23	BARC22	BARC21	BARC20	BARC19	BARC18	BARC17	BARC16	
BARCL	BARC15	BARC14	BARC13	BARC12	BARC11	BARC10	BARC9	BARC8	
	BARC7	BARC6	BARC5	BARC4	BARC3	BARC2	BARC1	BARC0	
BAMRCH	BAMRC31	BAMRC30	BAMRC29	BAMRC28	BAMRC27	BAMRC26	BAMRC25	BAMRC24	
	BAMRC23	BAMRC22	BAMRC21	BAMRC20	BAMRC19	BAMRC18	BAMRC17	BAMRC16	
BAMRCL	BAMRC15	BAMRC14	BAMRC13	BAMRC12	BAMRC11	BAMRC10	BAMRC9	BAMRC8	
	BAMRC7	BAMRC6	BAMRC5	BAMRC4	BAMRC3	BAMRC2	BAMRC1	BAMRC0	
BARDH	BARD31	BARD30	BARD29	BARD28	BARD27	BARD26	BARD25	BARD24	
	BARD23	BARD22	BARD21	BARD20	BARD19	BARD18	BARD17	BARD16	
BARDL	BARD15	BARD14	BARD13	BARD12	BARD11	BARD10	BARD9	BARD8	
	BARD7	BARD6	BARD5	BARD4	BARD3	BARD2	BARD1	BARD0	
BAMRDH	BAMRD31	BAMRD30	BAMRD29	BAMRD28	BAMRD27	BAMRD26	BAMRD25	BAMRD24	
	BAMRD23	BAMRD22	BAMRD21	BAMRD20	BAMRD19	BAMRD18	BAMRD17	BAMRD16	
BAMRDL	BAMRD15	BAMRD14	BAMRD13	BAMRD12	BAMRD11	BAMRD10	BAMRD9	BAMRD8	
	BAMRD7	BAMRD6	BAMRD5	BAMRD4	BAMRD3	BAMRD2	BAMRD1	BAMRD0	
BRCRA	—	—	CMFCPA	—	CPA2	CPA1	CPA0	—	
	—	—	IDA1	IDA0	RWA1	RWA0	—	—	
BRCRB	—	—	CMFCPB	—	CPB2	CPB1	CPB0	—	
	—	—	IDB1	IDB0	RWB1	RWB0	—	—	
BRCRC	—	—	CMFCPC	—	CPC2	CPC1	CPC0	—	
	—	—	IDC1	IDC0	RWC1	RWC0	—	—	
BRCRD	—	—	DMFCPD	—	CPD2	CPD1	CPD0	—	
	—	—	IDD1	IDD0	RWD1	RWD0	—	—	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_6	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_6
TCR_7	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_7
TCSR_6	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0	TMR_6
TCSR_7	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_7
TCORA_6									TMR_6
TCORA_7									TMR_7
TCORB_6									TMR_6
TCORB_7									TMR_7
TCNT_6									TMR_6
TCNT_7									TMR_7
TCCR_6	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_6
TCCR_7	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_7
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	I/O port
P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P6DDR	—	—	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	
PADDR	PA7DDR	PA6DDR	PA5DDR	PA4DDR	PA3DDR	PA2DDR	PA1DDR	PA0DDR	
PDDDR	PD7DDR	PD6DDR	PD5DDR	PD4DDR	PD3DDR	PD2DDR	PD1DDR	PD0DDR	
PEDDR	PE7DDR	PE6DDR	PE5DDR	PE4DDR	PE3DDR	PE2DDR	PE1DDR	PE0DDR	
PFDDR	—	—	—	PF4DDR	PF3DDR	PF2DDR	PF1DDR	PF0DDR	
P1ICR	P17ICR	P16ICR	P15ICR	P14ICR	P13ICR	P12ICR	P11ICR	P10ICR	
P2ICR	P27ICR	P26ICR	P25ICR	P24ICR	P23ICR	P22ICR	P21ICR	P20ICR	
P3ICR	P37ICR	P36ICR	P35ICR	P34ICR	P33ICR	P32ICR	P31ICR	P30ICR	
P4ICR	P47ICR	P46ICR	P45ICR	P44ICR	P43ICR	P42ICR	P41ICR	P40ICR	
P5ICR	P57ICR	P56ICR	P55ICR	P54ICR	P53ICR	P52ICR	P51ICR	P50ICR	
P6ICR	—	—	P65ICR	P64ICR	P63ICR	P62ICR	P61ICR	P60ICR	
PAICR	PA7ICR	PA6ICR	PA5ICR	PA4ICR	PA3ICR	PA2ICR	PA1ICR	PA0ICR	
PDICR	PD7ICR	PD6ICR	PD5ICR	PD4ICR	PD3ICR	PD2ICR	PD1ICR	PD0ICR	
PEICR	PE7ICR	PE6ICR	PE5ICR	PE4ICR	PE3ICR	PE2ICR	PE1ICR	PE0ICR	
PFICR	—	—	—	PF4ICR	PF3ICR	PF2ICR	PF1ICR	PF0ICR	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PORTH	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0	I/O port
PORTI	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0	
PHDR	PH7DR	PH6DR	PH5DR	PH4DR	PH3DR	PH2DR	PH1DR	PH0DR	
PIDR	PI7DR	PI6DR	PI5DR	PI4DR	PI3DR	PI2DR	PI1DR	PI0DR	
PHDDR	PH7DDR	PH6DDR	PH5DDR	PH4DDR	PH3DDR	PH2DDR	PH1DDR	PH0DDR	
PIDDR	PI7DDR	PI6DDR	PI5DDR	PI4DDR	PI3DDR	PI2DDR	PI1DDR	PI0DDR	
PHICR	PH7ICR	PH6ICR	PH5ICR	PH4ICR	PH3ICR	PH2ICR	PH1ICR	PH0ICR	
PIICR	PI7ICR	PI6ICR	PI5ICR	PI4ICR	PI3ICR	PI2ICR	PI1ICR	PI0ICR	
PDPCR	PD7PCR	PD6PCR	PD5PCR	PD4PCR	PD3PCR	PD2PCR	PD1PCR	PD0PCR	
PEPCR	PE7PCR	PE6PCR	PE5PCR	PE4PCR	PE3PCR	PE2PCR	PE1PCR	PE0PCR	
PFPCR	—	—	—	PF4PCR	PF3PCR	PF2PCR	PF1PCR	PF0PCR	
PHPCR	PH7PCR	PH6PCR	PH5PCR	PH4PCR	PH3PCR	PH2PCR	PH1PCR	PH0PCR	
PIPCR	PI7PCR	PI6PCR	PI5PCR	PI4PCR	PI3PCR	PI2PCR	PI1PCR	PI0PCR	
P2ODR	P27ODR	P26ODR	P25ODR	P24ODR	P23ODR	P22ODR	P21ODR	P20ODR	
PFODR	—	—	—	PF4ODR	PF3ODR	PF2ODR	PF1ODR	PF0ODR	
PF0CR0	CS7E	CS6E	CS5E	CS4E	CS3E	CS2E	CS1E	CS0E	
PF0CR1	CS7SA	CS7SB	CS6SA	CS6SB	CS5SA	CS5SB	CS4SA	CS4SB	
PF0CR2	—	CS2S	BSS	BSE	—	RDWRE	ASOE	—	
PF0CR4	—	—	—	A20E	A19E	A18E	A17E	A16E	
PF0CR6	—	LHWROE	—	—	TCLKS	—	—	—	
PF0CR7	—	—	—	—	DMAS1A	DMAS1B	DMAS0A	DMAS0B	
PF0CR9	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B	
PF0CRB	—	—	ITS13	ITS12	ITS11	ITS10	ITS9	ITS8	
PF0CRC	ITS7	ITS6	ITS5	ITS4	ITS3	ITS2	ITS1	ITS0	
SSIER	SSI15	SSI14	SSI13	SSI12	SSI11	SSI10	SSI9	SSI8	INTC
	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0	
DPSBKR0									SYSTEM
DPSBKR1									
DPSBKR2									
DPSBKR3									

Section 25 List of Registers

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DPSBKR4									SYSTEM
DPSBKR5									
DPSBKR6									
DPSBKR7									
DPSBKR8									
DPSBKR9									
DPSBKR10									
DPSBKR11									
DPSBKR12									
DPSBKR13									
DPSBKR14									
DPSBKR15									
DSAR_0									DMAC_0
DDAR_0									
DOFR_0									
DTCR_0									



Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DBSR_0	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	DMAC_0
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_0	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_0	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_1								DMAC_1	
DDAR_1									
DOFR_1									
DTCR_1									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DBSR_1	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	DMAC_1
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_1	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_1	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DMRSR_0								DMAC_0	
DMRSR_1								DMAC_1	
IPRA	—	IPRA14	IPRA13	IPRA12	—	IPRA10	IPRA9	IPRA8	INTC
	—	IPRA6	IPRA5	IPRA4	—	IPRA2	IPRA1	IPRA0	
IPRB	—	IPRB14	IPRB13	IPRB12	—	IPRB10	IPRB9	IPRB8	
	—	IPRB6	IPRB5	IPRB4	—	IPRB2	IPRB1	IPRB0	
IPRC	—	IPRC14	IPRC13	IPRC12	—	IPRC10	IPRC9	IPRC8	
	—	IPRC6	IPRC5	IPRC4	—	IPRC2	IPRC1	IPRC0	
IPRD	—	IPRD14	IPRD13	IPRD12	—	IPRD10	IPRD9	IPRD8	
	—	IPRD6	IPRD5	IPRD4	—	IPRD2	IPRD1	IPRD0	
IPRE	—	—	—	—	—	IPRE10	IPRE9	IPRE8	
	—	—	—	—	—	—	—	—	
IPRF	—	—	—	—	—	—	—	—	
	—	IPRF6	IPRF5	IPRF4	—	IPRF2	IPRF1	IPRF0	
IPRG	—	IPRG14	IPRG13	IPRG12	—	IPRG10	IPRG9	IPRG8	
	—	IPRG6	IPRG5	IPRG4	—	IPRG2	IPRG1	IPRG0	
IPRH	—	IPRH14	IPRH13	IPRH12	—	IPRH10	IPRH9	IPRH8	
	—	IPRH6	IPRH5	IPRH4	—	IPRH2	IPRH1	IPRH0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
IPRI	—	IPRI14	IPRI13	IPRI12	—	IPRI10	IPRI9	IPRI8	INTC
	—	—	—	—	—	—	—	—	
IPRK	—	IPRK14	IPRK13	IPRK12	—	—	—	—	
	—	IPRK6	IPRK5	IPRK4	—	IPRK2	IPRK1	IPRK0	
IPRL	—	IPRL14	IPRL13	IPRL12	—	IPRL10	IPRL9	IPRL8	
	—	IPRL6	IPRL5	IPRL4	—	—	—	—	
IPRP	—	—	—	—	—	IPRP10	IPRP9	IPRP8	
	—	IPRP6	IPRP5	IPRP4	—	IPRP2	IPRP1	IPRP0	
IPRQ	—	IPRQ14	IPRQ13	IPRQ12	—	—	—	—	
	—	IPRQ6	IPRQ5	IPRQ4	—	IPRQ2	IPRQ1	IPRQ0	
IPRR	—	IPRR14	IPRR13	IPRR12	—	—	—	—	
	—	—	—	—	—	—	—	—	
ISCRH	IRQ15SR	IRQ15SF	IRQ14SR	IRQ14SF	IRQ13SR	IRQ13SF	IRQ12SR	IRQ12SF	
	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF	
ISCLR	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF	
	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF	
DTCVBR									DTC
ABWCR	ABWH7	ABWH6	ABWH5	ABWH4	ABWH3	ABWH2	ABWH1	ABWH0	BSC
	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1	ABWL0	
ASTCR	AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0	
WTCRA	—	W72	W71	W70	—	W62	W61	W60	
	—	W52	W51	W50	—	W42	W41	W40	
WTCRB	—	W32	W31	W30	—	W22	W21	W20	
	—	W12	W11	W10	—	W02	W01	W00	
RDNCR	RDN7	RDN6	RDN5	RDN4	RDN3	RDN2	RDN1	RDN0	
	—	—	—	—	—	—	—	—	

## Section 25 List of Registers

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
CSACR	CSXH7	CSXH6	CSXH5	CSXH4	CSXH3	CSXH2	CSXH1	CSXH0	BSC
	CSXT7	CSXT6	CSXT5	CSXT4	CSXT3	CSXT2	CSXT1	CSXT0	
IDLCR	IDLS3	IDLS2	IDLS1	IDLS0	IDLCB1	IDLCB0	IDLCA1	IDLCA0	
	IDLSEL7	IDLSEL6	IDLSEL5	IDLSEL4	IDLSEL3	IDLSEL2	IDLSEL1	IDLSEL0	
BCR1	BRLE	BREQOE	—	—	—	—	WDBE	WAITE	
	DKC	—	—	—	—	—	—	—	
BCR2	—	—	—	IBCCS	—	—	—	PWDBE	
ENDIANCR	LE7	LE6	LE5	LE4	LE3	LE2	—	—	
SRAMCR	BCSEL7	BCSEL6	BCSEL5	BCSEL4	BCSEL3	BCSEL2	BCSEL1	BCSEL0	
	—	—	—	—	—	—	—	—	
BROMCR	BSRM0	BSTS02	BSTS01	BSTS00	—	—	BSWD01	BSWD00	
	BSRM1	BSTS12	BSTS11	BSTS10	—	—	BSWD11	BSWD10	
MPXCR	MPXE7	MPXE6	MPXE5	MPXE4	MPXE3	—	—	—	
	—	—	—	—	—	—	—	ADDEX	
RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0	
MDCR	—	—	—	—	MDS3	MDS2	MDS1	MDS0	SYSTEM
	—	—	—	—	—	—	—	—	
SYSCR	—	—	MACS	—	FETCHMD	—	EXPE	RAME	
	—	—	—	—	—	—	DTCMD	—	
SCKCR	PSTOP1	—	POSEL1	—	—	ICK2	ICK1	ICK0	
	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0	
SBYCR	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0	
	SLPIE	—	—	—	—	—	—	—	
MSTPCRA	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8	
	—	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	
MSTPCRB	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8	
	—	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	
MSTPCRC	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8	
	—	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
FCCS	—	—	—	FLER	—	—	—	SCO	FLASH
FPCS	—	—	—	—	—	—	—	PPVS	
FECS	—	—	—	—	—	—	—	EPVB	
FKEY	K7	K6	K5	K4	K3	K2	K1	K0	
FMATS	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0	
FTDAR	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0	
DPSBY	DPSBY	IOKEEP	RAMCUT2	RAMCUT1	—	—	—	RAMCUT0	SYSTEM
DPSWCR	—	—	WTSTS5	WTSTS4	WTSTS3	WTSTS2	WTSTS1	WTSTS0	
DPSIER	—	—	—	—	DIRQ3E	DIRQ2E	DIRQ1E	DIRQ0E	
DPSIFR	DNMIF	—	—	—	DIRQ3F	DIRQ2F	DIRQ1F	DIRQ0F	
DPSIEGR	DNMIEG	—	—	—	DIRQ3EG	DIRQ2EG	DIRQ1EG	DIRQ0EG	
RSTSR	DPSRSTF	—	—	—	—	—	—	—	
SEMR_2	—	—	—	—	ABCS	ACS2	ACS1	ACS0	SCI_2
SMR_3*1	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP0)	MP (BCP0)	CKS1	CKS0	SCI_3
BRR_3	—	—	—	—	—	—	—	—	
SCR_3*1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_3	—	—	—	—	—	—	—	—	
SSR_3*1	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_3	—	—	—	—	—	—	—	—	
SCMR_3	—	—	—	—	SDIR	SINV	—	SMIF	
SMR_4*1	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_4
BRR_4	—	—	—	—	—	—	—	—	
SCR_4*1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_4	—	—	—	—	—	—	—	—	
SSR_4*1	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_4	—	—	—	—	—	—	—	—	
SCMR_4	—	—	—	—	SDIR	SINV	—	SMIF	

## Section 25 List of Registers

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
ICCRA_0	ICE	RCVD	MST	TRS	CKS3	CKS2	CKS1	CKS0	IIC2_0
ICCRB_0	BBSY	SCP	SDAO	—	SCLO	—	IICRST	—	
ICMR_0	—	WAIT	—	—	BCWP	BC2	BC1	BC0	
ICIER_0	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR	ACKBT	
ICSR_0	TDRE	TEND	RDRF	NACKF	STOP	AL	AAS	ADZ	
SAR_0	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	—	
ICDRT_0									
ICDRR_0									
ICCRA_1	ICE	RCVD	MST	TRS	CKS3	CKS2	CKS1	CKS0	IIC2_1
ICCRB_1	BBSY	SCP	SDAO	—	SCLO	—	IICRST	—	
ICMR_1	—	WAIT	—	—	BCWP	BC2	BC1	BC0	
ICIER_1	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR	ACKBT	
ICSR_1	TDRE	TEND	RDRF	NACKF	STOP	AL	AAS	ADZ	
SAR_1	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	—	
ICDRT_1									
ICDRR_1									
TCR_2	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_2
TCR_3	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_3
TCSR_2	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0	TMR_2
TCSR_3	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_3
TCORA_2									TMR_2
TCORA_3									TMR_3
TCORB_2									TMR_2
TCORB_3									TMR_3
TCNT_2									TMR_2
TCNT_3									TMR_3
TCCR_2	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_2
TCCR_3	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_3
TCR_4	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_4
TCR_5	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_5

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCSR_4	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0	TMR_4
TCSR_5	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_5
TCORA_4									TMR_4
TCORA_5									TMR_5
TCORB_4									TMR_4
TCORB_5									TMR_5
TCNT_4									TMR_4
TCNT_5									TMR_5
TCCR_4	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_4
TCCR_5	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_5
TCR_4	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_4
TMDR_4	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_4	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_4	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_4	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_4									
TGRA_4									
TGRB_4									
TCR_5	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_5
TMDR_5	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_5	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_5	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_5	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_5									
TGRA_5									

## Section 25 List of Registers

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TGRB_5									TPU_5
DTCERA	DTCEA15	DTCEA14	DTCEA13	DTCEA12	DTCEA11	DTCEA10	DTCEA9	DTCEA8	DTC
	DTCEA7	DTCEA6	DTCEA5	DTCEA4	DTCEA3	DTCEA2	DTCEA1	DTCEA0	
DTCERB	—	—	DTCEB13	DTCEB12	DTCEB11	DTCEB10	DTCEB9	DTCEB8	
	DTCEB7	DTCEB6	DTCEB5	DTCEB4	DTCEB3	DTCEB2	DTCEB1	DTCEB0	
DTCERC	DTCEC15	DTCEC14	DTCEC13	DTCEC12	DTCEC11	DTCEC10	DTCEC9	DTCEC8	
	DTCEC7	DTCEC6	DTCEC5	DTCEC4	—	—	—	—	
DTCERD	—	—	DTCED13	DTCED12	—	—	—	—	
	—	—	DTCED5	DTCED4	DTCED3	DTCED2	DTCED1	DTCED0	
DTCERE	DTCEE15	DTCEE14	DTCEE13	DTCEE12	—	—	—	—	
	—	—	—	—	—	—	—	—	
DTCERF	—	—	—	—	—	—	—	—	
	—	—	—	—	DTCEF3	DTCEF2	DTCEF1	DTCEF0	
DTCERG	DTCEG15	DTCEG14	DTCEG13	DTCEG12	—	—	—	—	
	—	—	—	—	—	—	—	—	
DTCCR	—	—	—	RRS	RCHNE	—	—	ERR	
INTCR	—	—	INTM1	INTM0	NMIEG	—	—	—	INTC
CPUPCR	CPUPCE	DTCP2	DTCP1	DTCP0	IPSETE	CPUP2	CPUP1	CPUP0	
IER	IRQ15E	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E	
	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	
ISR	IRQ15F	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F	
	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
PORT1	P17	P16	P15	P14	P13	P12	P11	P10	I/O port
PORT2	P27	P26	P25	P24	P23	P22	P21	P20	
PORT3	P37	P36	P35	P34	P33	P32	P31	P30	
PORT4	P47	P46	P45	P44	P43	P42	P41	P40	
PORT5	P57	P56	P55	P54	P53	P52	P51	P50	
PORT6	—	—	P65	P64	P63	P62	P61	P60	



Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PORTA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	I/O port
PORTD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	
PORTE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	
PORTF	—	—	—	PF4	PF3	PF2	PF1	PF0	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	P27DR	P26DR	P25DR	P24DR	P23DR	P22DR	P21DR	P20DR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P6DR	—	—	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
PADR	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR	
PDDR	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR	
PEDR	PE7DR	PE6DR	PE5DR	PE4DR	PE3DR	PE2DR	PE1DR	PE0DR	
PFDR	—	—	—	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR	
SMR_2* <sup>1</sup>	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_2
BRR_2									
SCR_2* <sup>1</sup>	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_2									
SSR_2* <sup>1</sup>	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_2									
SCMR_2	—	—	—	—	SDIR	SINV	—	SMIF	
DADR0									D/A
DADR1									
DACR01	DAOE1	DAOE0	DAE	—	—	—	—	—	
PCR	G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0	PPG
PMR	G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV	
NDERH	NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8	
NDERL	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0	
PODRH	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	
PODRL	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0	

## Section 25 List of Registers

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
NDRH* <sup>2</sup>	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8	PPG
NDRL* <sup>2</sup>	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0	
NDRH* <sup>2</sup>	—	—	—	—	NDR11	NDR10	NDR9	NDR8	
NDRL* <sup>2</sup>	—	—	—	—	NDR3	NDR2	NDR1	NDR0	
SMR_0* <sup>1</sup>	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_0
BRR_0									
SCR_0* <sup>1</sup>	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_0									
SSR_0* <sup>1</sup>	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_0									
SCMR_0	—	—	—	—	SDIR	SINV	—	SMIF	
SMR_1* <sup>1</sup>	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_1
BRR_1									
SCR_1* <sup>1</sup>	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_1									
SSR_1* <sup>1</sup>	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_1									
SCMR_1	—	—	—	—	SDIR	SINV	—	SMIF	
TCSR	OVF	WT/ $\bar{IT}$	TME	—	—	CKS2	CKS1	CKS0	WDT
TCNT									
RSTCSR	WOVF	RSTE	—	—	—	—	—	—	
TCR_0	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_0
TCR_1	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_1
TCSR_0	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0	TMR_0
TCSR_1	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_1
TCORA_0									TMR_0
TCORA_1									TMR_1
TCORB_0									TMR_0

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCORB_1									TMR_1
TCNT_0									TMR_0
TCNT_1									TMR_1
TCCR_0	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_0
TCCR_1	—	—	—	—	TMRIS	—	ICKS1	ICKS0	TMR_1
TSTR	—	—	CST5	CST4	CST3	CST2	CST1	CST0	TPU
TSYR	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0	
TCR_0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_0
TMDR_0	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_0	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_0	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_0	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_0	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_0									
TGRA_0									
TGRB_0									
TGRC_0									
TGRD_0									
TCR_1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_1
TMDR_1	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_1	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_1	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_1	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_1									

Section 25 List of Registers

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TGRA_1									TPU_1
TGRB_1									
TCR_2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_2
TMDR_2	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_2	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_2	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_2	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_2									
TGRA_2									
TGRB_2									
TCR_3	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_3
TMDR_3	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_3	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_3	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_3	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_3	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_3									
TGRA_3									
TGRB_3									
TGRC_3									

Register	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Module
Abbreviation	31/23/15/7	30/22/14/6	29/21/13/5	28/20/12/4	27/19/11/3	26/18/10/2	25/17/9/1	24/16/8/0	
TGRD_3									TPU_3

- Notes:
- Parts of the bit functions differ in normal mode and the smart card interface.
  - When the same output trigger is specified for pulse output groups 2 and 3 by the PCR setting, the NDRH address is H'FFF7C. When different output triggers are specified, the NDRH addresses for pulse output groups 2 and 3 are H'FFF7E and H'FFF7C, respectively. Similarly, when the same output trigger is specified for pulse output groups 0 and 1 by the PCR setting, the NDRL address is H'FFF7D. When different output triggers are specified, the NDRL addresses for pulse output groups 0 and 1 are H'FFF7F and H'FFF7D, respectively.

## 25.3 Register States in Each Operating Mode

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module-Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
ADDRA	Initialized	—	—	—	—	Initialized*	Initialized	A/D
ADDRB	Initialized	—	—	—	—	Initialized*	Initialized	
ADDRC	Initialized	—	—	—	—	Initialized*	Initialized	
ADDRD	Initialized	—	—	—	—	Initialized*	Initialized	
ADDRE	Initialized	—	—	—	—	Initialized*	Initialized	
ADDRF	Initialized	—	—	—	—	Initialized*	Initialized	
ADDRG	Initialized	—	—	—	—	Initialized*	Initialized	
ADDRH	Initialized	—	—	—	—	Initialized*	Initialized	
ADCSR	Initialized	—	—	—	—	Initialized*	Initialized	
ADCR	Initialized	—	—	—	—	Initialized*	Initialized	
DSADDR0	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	$\Delta\Sigma$ /A/D
DSADDR1	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADDR2	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADDR3	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADDR4	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADDR5	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADOF0	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADOF1	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADOF2	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADOF3	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADCSR	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADCR	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
DSADMR	Initialized	—	—	—	—	Initialized*	Initialized	
BARAH	Initialized	—	—	—	—	Initialized*	Initialized	UBC
BARAL	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRAH	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRAL	Initialized	—	—	—	—	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
BARBH	Initialized	—	—	—	—	Initialized*	Initialized	UBC
BARBL	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRBH	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRBL	Initialized	—	—	—	—	Initialized*	Initialized	
BARCH	Initialized	—	—	—	—	Initialized*	Initialized	
BARCL	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRCH	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRCL	Initialized	—	—	—	—	Initialized*	Initialized	
BARDH	Initialized	—	—	—	—	Initialized*	Initialized	
BARDL	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRDH	Initialized	—	—	—	—	Initialized*	Initialized	
BAMRDL	Initialized	—	—	—	—	Initialized*	Initialized	
BRCRA	Initialized	—	—	—	—	Initialized*	Initialized	
BRCRB	Initialized	—	—	—	—	Initialized*	Initialized	
BRCRC	Initialized	—	—	—	—	Initialized*	Initialized	
BRCRD	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_6	Initialized	—	—	—	—	Initialized*	Initialized	TMR_6
TCR_7	Initialized	—	—	—	—	Initialized*	Initialized	TMR_7
TCSR_6	Initialized	—	—	—	—	Initialized*	Initialized	TMR_6
TCSR_7	Initialized	—	—	—	—	Initialized*	Initialized	TMR_7
TCORA_6	Initialized	—	—	—	—	Initialized*	Initialized	TMR_6
TCORA_7	Initialized	—	—	—	—	Initialized*	Initialized	TMR_7
TCORB_6	Initialized	—	—	—	—	Initialized*	Initialized	TMR_6
TCORB_7	Initialized	—	—	—	—	Initialized*	Initialized	TMR_7
TCNT_6	Initialized	—	—	—	—	Initialized*	Initialized	TMR_6
TCNT_7	Initialized	—	—	—	—	Initialized*	Initialized	TMR_7
TCCR_6	Initialized	—	—	—	—	Initialized*	Initialized	TMR_6
TCCR_7	Initialized	—	—	—	—	Initialized*	Initialized	TMR_7

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
P1DDR	Initialized	—	—	—	—	Initialized*	Initialized	I/O port
P2DDR	Initialized	—	—	—	—	Initialized*	Initialized	
P3DDR	Initialized	—	—	—	—	Initialized*	Initialized	
P6DDR	Initialized	—	—	—	—	Initialized*	Initialized	
PADDR	Initialized	—	—	—	—	Initialized*	Initialized	
PDDDR	Initialized	—	—	—	—	Initialized*	Initialized	
PEDDR	Initialized	—	—	—	—	Initialized*	Initialized	
PFDDR	Initialized	—	—	—	—	Initialized*	Initialized	
P1ICR	Initialized	—	—	—	—	Initialized*	Initialized	
P2ICR	Initialized	—	—	—	—	Initialized*	Initialized	
P3ICR	Initialized	—	—	—	—	Initialized*	Initialized	
P4ICR	Initialized	—	—	—	—	Initialized*	Initialized	
P5ICR	Initialized	—	—	—	—	Initialized*	Initialized	
P6ICR	Initialized	—	—	—	—	Initialized*	Initialized	
PAICR	Initialized	—	—	—	—	Initialized*	Initialized	
PDICR	Initialized	—	—	—	—	Initialized*	Initialized	
PEICR	Initialized	—	—	—	—	Initialized*	Initialized	
PFICR	Initialized	—	—	—	—	Initialized*	Initialized	
PORTH	Initialized	—	—	—	—	Initialized*	Initialized	
PORTI	Initialized	—	—	—	—	Initialized*	Initialized	
PHDR	Initialized	—	—	—	—	Initialized*	Initialized	
PIDR	Initialized	—	—	—	—	Initialized*	Initialized	
PHDDR	Initialized	—	—	—	—	Initialized*	Initialized	
PIDDR	Initialized	—	—	—	—	Initialized*	Initialized	
PHICR	Initialized	—	—	—	—	Initialized*	Initialized	
PIICR	Initialized	—	—	—	—	Initialized*	Initialized	
PDPCR	Initialized	—	—	—	—	Initialized*	Initialized	
PEPCR	Initialized	—	—	—	—	Initialized*	Initialized	
PFPCR	Initialized	—	—	—	—	Initialized*	Initialized	



Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
PHPCR	Initialized	—	—	—	—	Initialized*	Initialized	I/O port
PIPCR	Initialized	—	—	—	—	Initialized*	Initialized	
P2ODR	Initialized	—	—	—	—	Initialized*	Initialized	
PFODR	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR0	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR1	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR2	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR4	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR6	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR7	Initialized	—	—	—	—	Initialized*	Initialized	
PFCR9	Initialized	—	—	—	—	Initialized*	Initialized	
PFCRB	Initialized	—	—	—	—	Initialized*	Initialized	
PFCRC	Initialized	—	—	—	—	Initialized*	Initialized	
SSIER	Initialized	—	—	—	—	Initialized*	Initialized	INTC
DPSBKR0	Initialized	—	—	—	—	Retained	Initialized	SYSTEM
DPSBKR1	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR2	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR3	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR4	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR5	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR6	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR7	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR8	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR9	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR10	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR11	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR12	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR13	Initialized	—	—	—	—	Retained	Initialized	
DPSBKR14	Initialized	—	—	—	—	Retained	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
DPSBKR15	Initialized	—	—	—	—	Retained	Initialized	SYSTEM
DSAR_0	Initialized	—	—	—	—	Initialized*	Initialized	DMAC_0
DDAR_0	Initialized	—	—	—	—	Initialized*	Initialized	
DOFR_0	Initialized	—	—	—	—	Initialized*	Initialized	
DTCR_0	Initialized	—	—	—	—	Initialized*	Initialized	
DBSR_0	Initialized	—	—	—	—	Initialized*	Initialized	
DMDR_0	Initialized	—	—	—	—	Initialized*	Initialized	
DACR_0	Initialized	—	—	—	—	Initialized*	Initialized	
DSAR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DDAR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DOFR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DTCR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DBSR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DMDR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DACR_1	Initialized	—	—	—	—	Initialized*	Initialized	
DMRSR_0	Initialized	—	—	—	—	Initialized*	Initialized	DMAC_0
DMRSR_1	Initialized	—	—	—	—	Initialized*	Initialized	DMAC_1
IPRA	Initialized	—	—	—	—	Initialized*	Initialized	INTC
IPRB	Initialized	—	—	—	—	Initialized*	Initialized	
IPRC	Initialized	—	—	—	—	Initialized*	Initialized	
IPRD	Initialized	—	—	—	—	Initialized*	Initialized	
IPRE	Initialized	—	—	—	—	Initialized*	Initialized	
IPRF	Initialized	—	—	—	—	Initialized*	Initialized	
IPRG	Initialized	—	—	—	—	Initialized*	Initialized	
IPRH	Initialized	—	—	—	—	Initialized*	Initialized	
IPRI	Initialized	—	—	—	—	Initialized*	Initialized	
IPRK	Initialized	—	—	—	—	Initialized*	Initialized	
IPRL	Initialized	—	—	—	—	Initialized*	Initialized	
IPRP	Initialized	—	—	—	—	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
IPRQ	Initialized	—	—	—	—	Initialized*	Initialized	INTC
IPRR	Initialized	—	—	—	—	Initialized*	Initialized	
ISCRH	Initialized	—	—	—	—	Initialized*	Initialized	
ISCLR	Initialized	—	—	—	—	Initialized*	Initialized	
DTCVBR	Initialized	—	—	—	—	Initialized*	Initialized	DTC
ABWCR	Initialized	—	—	—	—	Initialized*	Initialized	BSC
ASTCR	Initialized	—	—	—	—	Initialized*	Initialized	
WTCRA	Initialized	—	—	—	—	Initialized*	Initialized	
WTCRB	Initialized	—	—	—	—	Initialized*	Initialized	
RDNCR	Initialized	—	—	—	—	Initialized*	Initialized	
CSACR	Initialized	—	—	—	—	Initialized*	Initialized	
IDLCR	Initialized	—	—	—	—	Initialized*	Initialized	
BCR1	Initialized	—	—	—	—	Initialized*	Initialized	
BCR2	Initialized	—	—	—	—	Initialized*	Initialized	
ENDIANCR	Initialized	—	—	—	—	Initialized*	Initialized	
SRAMCR	Initialized	—	—	—	—	Initialized*	Initialized	
BROMCR	Initialized	—	—	—	—	Initialized*	Initialized	
MPXCR	Initialized	—	—	—	—	Initialized*	Initialized	
RAMER	Initialized	—	—	—	—	Initialized*	Initialized	
MDCR	Initialized	—	—	—	—	Initialized*	Initialized	
SYSCR	Initialized	—	—	—	—	Initialized*	Initialized	
SCKCR	Initialized	—	—	—	—	Initialized*	Initialized	
SBYCR	Initialized	—	—	—	—	Initialized*	Initialized	
MSTPCRA	Initialized	—	—	—	—	Initialized*	Initialized	
MSTPCRB	Initialized	—	—	—	—	Initialized*	Initialized	
MSTPCRC	Initialized	—	—	—	—	Initialized*	Initialized	
FCCS	Initialized	—	—	—	—	Initialized*	Initialized	FLASH
FPCS	Initialized	—	—	—	—	Initialized*	Initialized	
FECS	Initialized	—	—	—	—	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
FKEY	Initialized	—	—	—	—	Initialized*	Initialized	FLASH
FMATS	Initialized	—	—	—	—	Initialized*	Initialized	
FTDAR	Initialized	—	—	—	—	Initialized*	Initialized	
DPSBYCR	Initialized	—	—	—	—	Retained	Initialized	SYSTEM
DPSWCR	Initialized	—	—	—	—	Retained	Initialized	
DPSIER	Initialized	—	—	—	—	Retained	Initialized	
DPSIFR	Initialized	—	—	—	—	Retained	Initialized	
DPSIEGR	Initialized	—	—	—	—	Retained	Initialized	
RSTSR	Initialized	—	—	—	—	Retained	Initialized	
SEMR_2	Initialized	—	—	—	—	Initialized*	Initialized	SCI_2
SMR_3	Initialized	—	—	—	—	Initialized*	Initialized	SCI_3
BRR_3	Initialized	—	—	—	—	Initialized*	Initialized	
SCR_3	Initialized	—	—	—	—	Initialized*	Initialized	
TDR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SSR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
RDR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SCMR_3	Initialized	—	—	—	—	Initialized*	Initialized	
SMR_4	Initialized	—	—	—	—	Initialized*	Initialized	SCI_4
BRR_4	Initialized	—	—	—	—	Initialized*	Initialized	
SCR_4	Initialized	—	—	—	—	Initialized*	Initialized	
TDR_4	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SSR_4	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
RDR_4	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SCMR_4	Initialized	—	—	—	—	Initialized*	Initialized	
ICCRA_0	Initialized	—	—	—	—	Initialized*	Initialized	IIC2_0
ICCRB_0	Initialized	—	—	—	—	Initialized*	Initialized	
ICMR_0	Initialized	—	—	—	—	Initialized*	Initialized	
ICIER_0	Initialized	—	—	—	—	Initialized*	Initialized	
ICSR_0	Initialized	—	—	—	—	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
SAR_0	Initialized	—	—	—	—	Initialized*	Initialized	IIC2_0
ICDRT_0	Initialized	—	—	—	—	Initialized*	Initialized	
ICDRR_0	Initialized	—	—	—	—	Initialized*	Initialized	
ICCRA_1	Initialized	—	—	—	—	Initialized*	Initialized	IIC2_1
ICCRB_1	Initialized	—	—	—	—	Initialized*	Initialized	
ICMR_1	Initialized	—	—	—	—	Initialized*	Initialized	
ICIER_1	Initialized	—	—	—	—	Initialized*	Initialized	
ICSR_1	Initialized	—	—	—	—	Initialized*	Initialized	
SAR_1	Initialized	—	—	—	—	Initialized*	Initialized	
ICDRT_1	Initialized	—	—	—	—	Initialized*	Initialized	
ICDRR_1	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_2	Initialized	—	—	—	—	Initialized*	Initialized	TMR_2
TCR_3	Initialized	—	—	—	—	Initialized*	Initialized	TMR_3
TCSR_2	Initialized	—	—	—	—	Initialized*	Initialized	TMR_2
TCSR_3	Initialized	—	—	—	—	Initialized*	Initialized	TMR_3
TCORA_2	Initialized	—	—	—	—	Initialized*	Initialized	TMR_2
TCORA_3	Initialized	—	—	—	—	Initialized*	Initialized	TMR_3
TCORB_2	Initialized	—	—	—	—	Initialized*	Initialized	TMR_2
TCORB_3	Initialized	—	—	—	—	Initialized*	Initialized	TMR_3
TCNT_2	Initialized	—	—	—	—	Initialized*	Initialized	TMR_2
TCNT_3	Initialized	—	—	—	—	Initialized*	Initialized	TMR_3
TCCR_2	Initialized	—	—	—	—	Initialized*	Initialized	TMR_2
TCCR_3	Initialized	—	—	—	—	Initialized*	Initialized	TMR_3
TCR_4	Initialized	—	—	—	—	Initialized*	Initialized	TMR_4
TCR_5	Initialized	—	—	—	—	Initialized*	Initialized	TMR_5
TCSR_4	Initialized	—	—	—	—	Initialized*	Initialized	TMR_4
TCSR_5	Initialized	—	—	—	—	Initialized*	Initialized	TMR_5
TCORA_4	Initialized	—	—	—	—	Initialized*	Initialized	TMR_4
TCORA_5	Initialized	—	—	—	—	Initialized*	Initialized	TMR_5

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
TCORB_4	Initialized	—	—	—	—	Initialized*	Initialized	TMR_4
TCORB_5	Initialized	—	—	—	—	Initialized*	Initialized	TMR_5
TCNT_4	Initialized	—	—	—	—	Initialized*	Initialized	TMR_4
TCNT_5	Initialized	—	—	—	—	Initialized*	Initialized	TMR_5
TCCR_4	Initialized	—	—	—	—	Initialized*	Initialized	TMR_4
TCCR_5	Initialized	—	—	—	—	Initialized*	Initialized	TMR_5
TCR_4	Initialized	—	—	—	—	Initialized*	Initialized	TPU_4
TMDR_4	Initialized	—	—	—	—	Initialized*	Initialized	
TIOR_4	Initialized	—	—	—	—	Initialized*	Initialized	
TIER_4	Initialized	—	—	—	—	Initialized*	Initialized	
TSR_4	Initialized	—	—	—	—	Initialized*	Initialized	
TCNT_4	Initialized	—	—	—	—	Initialized*	Initialized	
TGRA_4	Initialized	—	—	—	—	Initialized*	Initialized	
TGRB_4	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_5	Initialized	—	—	—	—	Initialized*	Initialized	TPU_5
TMDR_5	Initialized	—	—	—	—	Initialized*	Initialized	
TIOR_5	Initialized	—	—	—	—	Initialized*	Initialized	
TIER_5	Initialized	—	—	—	—	Initialized*	Initialized	
TSR_5	Initialized	—	—	—	—	Initialized*	Initialized	
TCNT_5	Initialized	—	—	—	—	Initialized*	Initialized	
TGRA_5	Initialized	—	—	—	—	Initialized*	Initialized	
TGRB_5	Initialized	—	—	—	—	Initialized*	Initialized	
DTCERA	Initialized	—	—	—	—	Initialized*	Initialized	DTC
DTCERB	Initialized	—	—	—	—	Initialized*	Initialized	
DTCERC	Initialized	—	—	—	—	Initialized*	Initialized	
DTCERD	Initialized	—	—	—	—	Initialized*	Initialized	
DTCERE	Initialized	—	—	—	—	Initialized*	Initialized	
DTCERF	Initialized	—	—	—	—	Initialized*	Initialized	
DTCERG	Initialized	—	—	—	—	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
DTCCR	Initialized	—	—	—	—	Initialized*	Initialized	DTC
INTCR	Initialized	—	—	—	—	Initialized*	Initialized	INTC
CPUPCR	Initialized	—	—	—	—	Initialized*	Initialized	
IER	Initialized	—	—	—	—	Initialized*	Initialized	
ISR	Initialized	—	—	—	—	Initialized*	Initialized	
PORT1	—	—	—	—	—	—	—	I/O port
PORT2	—	—	—	—	—	—	—	
PORT3	—	—	—	—	—	—	—	
PORT4	—	—	—	—	—	—	—	
PORT5	—	—	—	—	—	—	—	
PORT6	—	—	—	—	—	—	—	
PORTA	—	—	—	—	—	—	—	
PORTD	—	—	—	—	—	—	—	
PORTE	—	—	—	—	—	—	—	
PORTF	—	—	—	—	—	—	—	
P1DR	Initialized	—	—	—	—	Initialized*	Initialized	
P2DR	Initialized	—	—	—	—	Initialized*	Initialized	
P3DR	Initialized	—	—	—	—	Initialized*	Initialized	
P6DR	Initialized	—	—	—	—	Initialized*	Initialized	
PADR	Initialized	—	—	—	—	Initialized*	Initialized	
PDDR	Initialized	—	—	—	—	Initialized*	Initialized	
PEDR	Initialized	—	—	—	—	Initialized*	Initialized	
PFDR	Initialized	—	—	—	—	Initialized*	Initialized	
SMR_2	Initialized	—	—	—	—	Initialized*	Initialized	SCI_2
BRR_2	Initialized	—	—	—	—	Initialized*	Initialized	
SCR_2	Initialized	—	—	—	—	Initialized*	Initialized	
TDR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SSR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
RDR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
SCMR_2	Initialized	—	—	—	—	Initialized*	Initialized	SCI_2
DADR0	Initialized	—	—	—	—	Initialized*	Initialized	D/A
DADR1	Initialized	—	—	—	—	Initialized*	Initialized	
DACR01	Initialized	—	—	—	—	Initialized*	Initialized	PPG
PCR	Initialized	—	—	—	—	Initialized*	Initialized	
PMR	Initialized	—	—	—	—	Initialized*	Initialized	
NDERH	Initialized	—	—	—	—	Initialized*	Initialized	
NDERL	Initialized	—	—	—	—	Initialized*	Initialized	
PODRH	Initialized	—	—	—	—	Initialized*	Initialized	
PODRL	Initialized	—	—	—	—	Initialized*	Initialized	
NDRH	Initialized	—	—	—	—	Initialized*	Initialized	
NDRL	Initialized	—	—	—	—	Initialized*	Initialized	
SMR_0	Initialized	—	—	—	—	Initialized*	Initialized	
BRR_0	Initialized	—	—	—	—	Initialized*	Initialized	
SCR_0	Initialized	—	—	—	—	Initialized*	Initialized	
TDR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SSR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
RDR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SCMR_0	Initialized	—	—	—	—	Initialized*	Initialized	SCI_1
SMR_1	Initialized	—	—	—	—	Initialized*	Initialized	
BRR_1	Initialized	—	—	—	—	Initialized*	Initialized	
SCR_1	Initialized	—	—	—	—	Initialized*	Initialized	
TDR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
SSR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	
RDR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized*	Initialized	WDT
SCMR_1	Initialized	—	—	—	—	Initialized*	Initialized	
TCSR	Initialized	—	—	—	—	Initialized*	Initialized	
TCNT	Initialized	—	—	—	—	Initialized*	Initialized	
RSTCSR	Initialized	—	—	—	—	Initialized*	Initialized	



Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
TCR_0	Initialized	—	—	—	—	Initialized*	Initialized	TMR_0
TCR_1	Initialized	—	—	—	—	Initialized*	Initialized	TMR_1
TCSR_0	Initialized	—	—	—	—	Initialized*	Initialized	TMR_0
TCSR_1	Initialized	—	—	—	—	Initialized*	Initialized	TMR_1
TCORA_0	Initialized	—	—	—	—	Initialized*	Initialized	TMR_0
TCORA_1	Initialized	—	—	—	—	Initialized*	Initialized	TMR_1
TCORB_0	Initialized	—	—	—	—	Initialized*	Initialized	TMR_0
TCORB_1	Initialized	—	—	—	—	Initialized*	Initialized	TMR_1
TCNT_0	Initialized	—	—	—	—	Initialized*	Initialized	TMR_0
TCNT_1	Initialized	—	—	—	—	Initialized*	Initialized	TMR_1
TCCR_0	Initialized	—	—	—	—	Initialized*	Initialized	TMR_0
TCCR_1	Initialized	—	—	—	—	Initialized*	Initialized	TMR_1
TSTR	Initialized	—	—	—	—	Initialized*	Initialized	TPU
TSYR	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_0	Initialized	—	—	—	—	Initialized*	Initialized	TPU_0
TMDR_0	Initialized	—	—	—	—	Initialized*	Initialized	
TIORH_0	Initialized	—	—	—	—	Initialized*	Initialized	
TIORL_0	Initialized	—	—	—	—	Initialized*	Initialized	
TIER_0	Initialized	—	—	—	—	Initialized*	Initialized	
TSR_0	Initialized	—	—	—	—	Initialized*	Initialized	
TCNT_0	Initialized	—	—	—	—	Initialized*	Initialized	
TGRA_0	Initialized	—	—	—	—	Initialized*	Initialized	
TGRB_0	Initialized	—	—	—	—	Initialized*	Initialized	
TGRC_0	Initialized	—	—	—	—	Initialized*	Initialized	
TGRD_0	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_1	Initialized	—	—	—	—	Initialized*	Initialized	TPU_1
TMDR_1	Initialized	—	—	—	—	Initialized*	Initialized	
TIOR_1	Initialized	—	—	—	—	Initialized*	Initialized	
TIER_1	Initialized	—	—	—	—	Initialized*	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop State	All-Module- Clock-Stop	Software Standby	Deep Software Standby	Hardware Standby	Module
TSR_1	Initialized	—	—	—	—	Initialized*	Initialized	TPU_1
TCNT_1	Initialized	—	—	—	—	Initialized*	Initialized	
TGRA_1	Initialized	—	—	—	—	Initialized*	Initialized	
TGRB_1	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_2	Initialized	—	—	—	—	Initialized*	Initialized	TPU_2
TMDR_2	Initialized	—	—	—	—	Initialized*	Initialized	
TIOR_2	Initialized	—	—	—	—	Initialized*	Initialized	
TIER_2	Initialized	—	—	—	—	Initialized*	Initialized	
TSR_2	Initialized	—	—	—	—	Initialized*	Initialized	
TCNT_2	Initialized	—	—	—	—	Initialized*	Initialized	
TGRA_2	Initialized	—	—	—	—	Initialized*	Initialized	
TGRB_2	Initialized	—	—	—	—	Initialized*	Initialized	
TCR_3	Initialized	—	—	—	—	Initialized*	Initialized	TPU_3
TMDR_3	Initialized	—	—	—	—	Initialized*	Initialized	
TIORH_3	Initialized	—	—	—	—	Initialized*	Initialized	
TIORL_3	Initialized	—	—	—	—	Initialized*	Initialized	
TIER_3	Initialized	—	—	—	—	Initialized*	Initialized	
TSR_3	Initialized	—	—	—	—	Initialized*	Initialized	
TCNT_3	Initialized	—	—	—	—	Initialized*	Initialized	
TGRA_3	Initialized	—	—	—	—	Initialized*	Initialized	
TGRB_3	Initialized	—	—	—	—	Initialized*	Initialized	
TGRC_3	Initialized	—	—	—	—	Initialized*	Initialized	
TGRD_3	Initialized	—	—	—	—	Initialized*	Initialized	

Note: \* This register is not initialized in deep software standby mode, though, initialized after clearing the deep software standby mode. It is because a reset exception handling is carried out by an internal reset when the deep software standby mode is cleared.

## Section 26 Electrical Characteristics

### 26.1 Electrical Characteristics

#### 26.1.1 Absolute Maximum Ratings

**Table 26.1 Absolute Maximum Ratings**

Item	Symbol	Value	Unit
Power supply voltage	$V_{CC, PLLV_{cc}}$	-0.3 to +4.6	V
Input voltage (except ports 4 and 5)	$V_{in}$	-0.3 to $V_{CC} + 0.3$	V
Input voltage (port 4)	$V_{in}$	-0.3 to $AV_{CC}P + 0.3$	V
Input voltage (port 5)	$V_{in}$	-0.3 to $AV_{CC} + 0.3$	V
Reference power supply voltage ( $V_{ref}$ )	$V_{ref}$	-0.3 to $AV_{CC} + 0.3$	V
Reference power supply voltage ( $AV_{ref}T$ )	$AV_{ref}T$	-0.3 to $AV_{CC}A + 0.3$	V
Analog power supply voltage ( $AV_{cc}$ )	$AV_{CC}$	-0.3 to +4.6	V
Analog power supply voltage ( $AV_{cc}P$ , $AV_{cc}A$ , $AV_{cc}D$ )	$AV_{CC}P =$ $AV_{CC}A =$ $AV_{CC}D$	-0.3 to +4.6	V
Analog input voltage (AN0 to 7)	$V_{AN}$	-0.3 to $AV_{CC} + 0.3$	V
Analog input voltage (ANDS0 to 3, ANDS4P/4N, ANDS5P/5N)	$V_{AN}$	-0.3 to $AV_{CC}P + 0.3$	V
Operating temperature	$T_{opr}$	Regular specifications: -20 to +75* <hr/> Wide-range specifications: -40 to +85*	°C
Storage temperature	$T_{stg}$	-55 to +125	°C

Caution: Permanent damage to the LSI may result if absolute maximum ratings are exceeded.

Note: \* The operating temperature when programming or erasing the flash memory is:

Regular specifications: 0 to +75°C

Wide-range specifications: 0 to +85°C

## 26.1.2 DC Characteristics

Table 26.2 DC Characteristics (1)

Conditions:  $V_{cc} = PLLV_{cc} = 3.0$  to  $3.6$  V,  $AV_{cc} = 3.0$  to  $3.6$  V,  $AV_{ccP} = AV_{ccA} = AV_{ccD} = 3.0$  to  $3.6$  V,  $V_{ref} = 3.0$  V to  $AV_{cc}$ ,  $AV_{refT} = AV_{ccA}$ ,  $V_{ss} = PLLV_{ss} = AV_{ss} = AV_{ssP} = AV_{ssA} = AV_{ssD} = AV_{refB} = 0$  V\*<sup>1</sup>  
 $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Schmitt trigger input voltage	$\overline{IRQ}$ input pin,	$VT^-$	$V_{cc} \times 0.2$	—	—	V	
	TPU input pin,	$VT^+$	—	—	$V_{cc} \times 0.7$	V	
	TMR input pin, port 2, port 3	$VT^+ - VT^-$	$V_{cc} \times 0.06$	—	—	V	
	Port 5* <sup>2</sup>	$VT^-$	$AV_{cc} \times 0.2$	—	—	V	
		$VT^+$	—	—	$AV_{cc} \times 0.7$	V	
		$VT^+ - VT^-$	$AV_{cc} \times 0.06$	—	—	V	
Input high voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$ , $\overline{STBY}$ , EMLE, NMI	$V_{IH}$	$V_{cc} \times 0.9$	—	$V_{cc} + 0.3$	V	
	EXTAL		$V_{cc} \times 0.7$	—	$V_{cc} + 0.3$	V	
	Port 4		$AV_{ccP} \times 0.7$	—	$AV_{ccP} + 0.3$	V	
	Port 5		$AV_{cc} \times 0.7$	—	$AV_{cc} + 0.3$	V	
	Other input pins		$V_{cc} \times 0.7$	—	$V_{cc} + 0.3$	V	
Input low voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$ , $\overline{STBY}$ , EMLE	$V_{IL}$	-0.3	—	$V_{cc} \times 0.1$	V	
	EXTAL, NMI		-0.3	—	$V_{cc} \times 0.2$	V	
	Port 4		-0.3	—	$AV_{ccP} \times 0.2$	V	
	Port 5		-0.3	—	$AV_{cc} \times 0.2$	V	
	Other input pins		-0.3	—	$V_{cc} \times 0.2$	V	
Output high voltage	All output pins	$V_{OH}$	$V_{cc} - 0.5$	—	—	V	$I_{OH} = -200 \mu A$
			$V_{cc} - 1.0$	—	—	V	$I_{OH} = -1 \text{ mA}$
Output low voltage	All output pins	$V_{OL}$	—	—	0.4	V	$I_{OL} = 1.6 \text{ mA}$
	Port 3		—	—	1.0	V	$I_{OL} = 10 \text{ mA}$

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Input leakage current	$\overline{\text{RES}}$	$ I_{in} $	—	—	10.0	$\mu\text{A}$	$V_{in} = 0.5$ to $V_{cc} - 0.5$ V
	MD, $\overline{\text{STBY}}$ , EMLE, NMI		—	—	1.0		
	Port 4		—	—	1.0		$V_{in} = 0.5$ to $AV_{cc}P - 0.5$ V
	Port 5		—	—	1.0		$V_{in} = 0.5$ to $AV_{cc} - 0.5$ V

**Table 26.2 DC Characteristics (2)**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CCP} = AV_{CCA} = AV_{CCD} = 3.0$  to  $3.6$  V,  $V_{REF} = 3.0$  V to  $AV_{CC}$ ,  $AV_{REFT} = AV_{CCA}$ ,  $V_{SS} = PLLV_{SS} = AV_{SS} = AV_{SSP} = AV_{SSA} = AV_{SSD} = AV_{REFB} = 0$  V\*<sup>1</sup>

$T_a = -20$  to  $+75$  °C (regular specifications),

$T_a = -40$  to  $+85$  °C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Three-state leakage current (off state)	Ports 1 to 3, 6, A, D to F, H, I	$ I_{TSI} $	—	—	1.0	μA	$V_{in} = 0.5$ to $V_{CC} - 0.5$ V
Input pull-up MOS current	Ports D to F, H, I	$-I_p$	10	—	300	μA	$V_{CC} = 3.0$ to $3.6$ V $V_{in} = 0$ V
Input capacitance	ANDS[3:0], ANDS4P,4N ANDS5P,5N	$C_{in}$	—	—	30	pF	$V_{in} = 0$ V $f = 1$ MHz $T_a = 25$ °C
	Other input pins		—	—	15	pF	$V_{in} = 0$ V $f = 1$ MHz $T_a = 25$ °C
Supply current* <sup>3</sup>	Normal operation	$I_{CC}^{*5}$	—	48	76	mA	$I\phi = B\phi = 50$ MHz $P\phi = 25$ MHz
			—	46	66		$I\phi = B\phi = P\phi = 35$ MHz* <sup>6</sup>
	Sleep mode		—	39	45		$I\phi = B\phi = 50$ MHz $P\phi = 25$ MHz
			—	38	43		$I\phi = B\phi = P\phi = 35$ MHz* <sup>6</sup>
	Software standby mode		—	0.5	1	mA	$T_a \leq 50$ °C
			—	—	3.2		$T_a > 50$ °C
	Deep software standby mode* <sup>4</sup> (RAM retained)		—	19	55	μA	$T_a \leq 50$ °C
			—	—	190		$T_a > 50$ °C
	Deep software standby mode* <sup>4</sup> (RAM power supply stopped)		—	4	7		$T_a \leq 50$ °C
			—	—	16		$T_a > 50$ °C
	Hardware standby mode* <sup>4</sup>		—	3	5		$T_a \leq 50$ °C
			—	—	15		$T_a > 50$ °C
All-module-clock-stop mode* <sup>6</sup>		—	22	29	mA		

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Analog power supply current	During A/D and D/A conversion	$I_{CC}$	—	1.0	2.0	mA	
	Standby for A/D and D/A conversion		—	0.5	2.0	$\mu$ A	
Reference power supply current	During A/D and D/A conversion	$I_{CC}$	—	1.0	2.0	mA	
	Standby for A/D and D/A conversion		—	0.1	1.0	$\mu$ A	
RAM standby voltage		$V_{RAM}$	2.5	—	—	V	
$V_{CC}$ start voltage* <sup>7</sup>		$V_{CCSTART}$	—	—	0.8	V	
$V_{CC}$ rising gradient* <sup>7</sup>		$SV_{CC}$	—	—	20	ms/V	

Notes: 1. When the A/D and D/A converters are not used, the  $AV_{CC}$ ,  $V_{ref}$  and  $AV_{SS}$  pins should not be open. Connect the  $AV_{CC}$  and  $V_{ref}$  pins to  $V_{CC}$ , and the  $AV_{SS}$  pin to  $V_{SS}$ .

When the  $\Delta\Sigma$  A/D converter is not used, the  $AV_{CCP}$ ,  $AV_{CCA}$ ,  $AV_{CCD}$ ,  $AV_{refT}$ ,  $AV_{SSP}$ ,  $AV_{SSA}$ ,  $AV_{SSD}$  and  $AV_{refB}$  pins should not be open.

Connect the  $AV_{CCP}$ ,  $AV_{CCA}$ ,  $AV_{CCD}$  and  $AV_{refT}$  pins to  $V_{CC}$ , and the  $AV_{SSP}$ ,  $AV_{SSA}$ ,  $AV_{SSD}$  and  $AV_{refB}$  pins to  $V_{SS}$ .

- The case where port 5 is used as  $\overline{IRQ0}$  to  $\overline{IRQ7}$ .
- Supply current values are for  $V_{IHmin} = V_{CC} - 0.5$  V and  $V_{ILmax} = 0.5$  V with all output pins unloaded and all input pull-up MOSs in the off state.
- The values are for  $V_{IHmin} = V_{CC} \times 0.9$  and  $V_{ILmax} = 0.3$  V.
- $I_{CC}$  depends on f as follows.

Normal operation:

$$I_{CCmax} = 16 \text{ (mA)} + 1.20 \text{ (mA/MHz)} \times f \quad (I\phi = B\phi, P\phi = 1/2I\phi)$$

$$I_{CCmax} = 16 \text{ (mA)} + 1.44 \text{ (mA/MHz)} \times f \quad (I\phi = B\phi = P\phi)$$

Sleep mode:

$$I_{CCmax} = 16 \text{ (mA)} + 0.57 \text{ (mA/MHz)} \times f \quad (I\phi = B\phi, P\phi = 1/2I\phi)$$

$$I_{CCmax} = 16 \text{ (mA)} + 0.78 \text{ (mA/MHz)} \times f \quad (I\phi = B\phi = P\phi)$$

- The values are for reference.
- This can be applied when the  $\overline{RES}$  pin is held low at power-on.

**Table 26.3 Permissible Output Currents**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CCP} = AV_{CCA} = AV_{CCD} = 3.0$  to  $3.6$  V,  $V_{ref} = 3.0$  V to  $AV_{CC}$ ,  $AV_{refT} = AV_{CCA}$ ,  $V_{SS} = PLLV_{SS} = AV_{SS} = AV_{SSP} = AV_{SSA} = AV_{SSD} = AV_{refB} = 0$  V\*

$T_a = -20$  to  $+75$  °C (regular specifications),

$T_a = -40$  to  $+85$  °C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit
Permissible output low current (per pin)	Output pins except port 3	$I_{OL}$	—	—	2.0	mA
Permissible output low current (per pin)	Port 3	$I_{OL}$	—	—	10	mA
Permissible output low current (total)	Total of all output pins	$\Sigma I_{OL}$	—	—	80	mA
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2.0	mA
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	40	mA

Caution: To protect the LSI's reliability, do not exceed the output current values in table 26.3.

Note: \* When the A/D and D/A converters are not used, the  $AV_{CC}$ ,  $V_{ref}$ , and  $AV_{SS}$  pins should not be open. Connect the  $AV_{CC}$  and  $V_{ref}$  pins to  $V_{CC}$ , and the  $AV_{SS}$  pin to  $V_{SS}$ .

When the  $\Delta\Sigma$  A/D converter is not used, the  $AV_{CCP}$ ,  $AV_{CCA}$ ,  $AV_{CCD}$ ,  $AV_{refT}$ ,  $AV_{SSP}$ ,  $AV_{SSA}$ ,  $AV_{SSD}$  and  $AV_{refB}$  pins should not be open.

Connect the  $AV_{CCP}$ ,  $AV_{CCA}$ ,  $AV_{CCD}$  and  $AV_{refT}$  pins to  $V_{CC}$ , and the  $AV_{SSP}$ ,  $AV_{SSA}$ ,  $AV_{SSD}$  and  $AV_{refB}$  pins to  $V_{SS}$ .



## 26.1.3 AC Characteristics

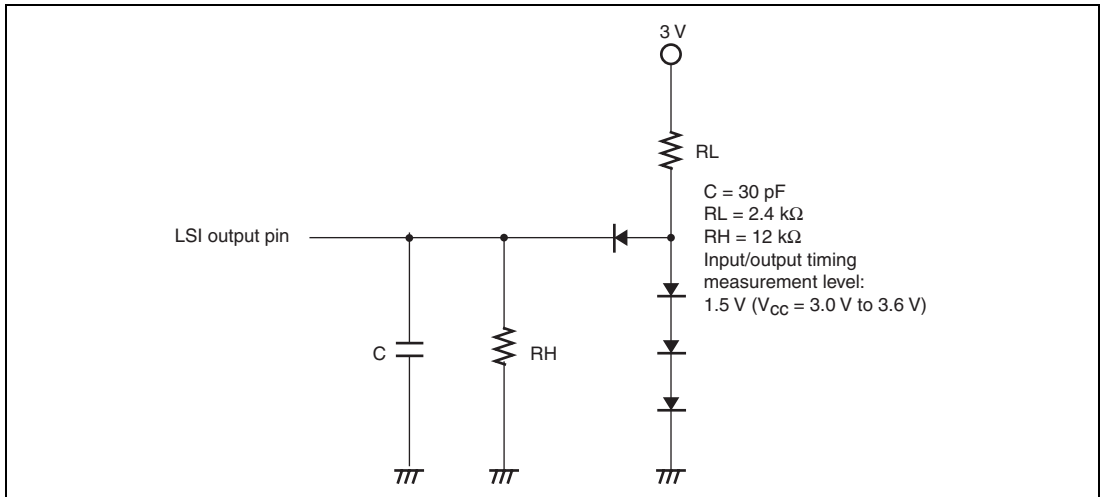


Figure 26.1 Output Load Circuit

## (1) Clock Timing

Table 26.4 Clock Timing

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CCP} = AV_{CCA} = AV_{CCD} = 3.0$  to  $3.6$  V,  $V_{ref} = 3.0$  V to  $AV_{CC}$ ,  $AV_{refT} = AV_{CCA}$ ,  $V_{SS} = PLLV_{SS} = AV_{SS} = AV_{SSP} = AV_{SSA} = AV_{SSD} = AV_{refB} = 0$  V  
 $I\phi = 8$  to  $50$  MHz,  $B\phi = 8$  to  $50$  MHz,  $P\phi = 8$  to  $35$  MHz,  
 $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit.	Test Conditions
Clock cycle time	$t_{cyc}$	20.0	125	ns	Figure 26.2
Clock high pulse width	$t_{CH}$	5	—	ns	
Clock low pulse width	$t_{CL}$	5	—	ns	
Clock rising time	$t_{Cr}$	—	5	ns	
Clock falling time	$t_{Cf}$	—	5	ns	
Oscillation settling time after reset (crystal)	$t_{OSC1}$	10	—	ms	Figure 26.5

Item	Symbol	Min.	Max.	Unit	Test Conditions
Oscillation settling time after leaving software standby mode (crystal)	$t_{OSC2}$	10	—	ms	Figure 26.3
Oscillation settling time after leaving deep software standby (crystal)	$t_{OSC2}$	10	—	ms	Figure 26.4
External clock output delay settling time	$t_{DEXT}$	1	—	ms	Figure 26.5
External clock input low pulse width	$t_{EXL}$	27.7	—	ns	Figure 26.6
External clock input high pulse width	$t_{EXH}$	27.7	—	ns	
External clock rising time	$t_{EXr}$	—	5	ns	
External clock falling time	$t_{EXf}$	—	5	ns	

## (2) Control Signal Timing

**Table 26.5 Control Signal Timing**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CCP} = AV_{CCA} = AV_{CCD} = 3.0$  to  $3.6$  V,  $V_{REF} = 3.0$  V to  $AV_{CC}$ ,  $AV_{REFT} = AV_{CCA}$ ,  $V_{SS} = PLLV_{SS} = AV_{SS} = AV_{SSP} = AV_{SSA} = AV_{SSD} = AV_{REFB} = 0$  V

$I_{\Phi} = 8$  to  $50$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),

$T_a = -40$  to  $+85$  °C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{RES}$ setup time	$t_{RESS}$	200	—	ns	Figure 26.7
$\overline{RES}$ pulse width	$t_{RESW}$	20	—	$t_{cyc}$	
NMI setup time	$t_{NMIS}$	150	—	ns	Figure 26.8
NMI hold time	$t_{NMIH}$	10	—	ns	
NMI pulse width (after leaving software standby mode or deep software standby mode)	$t_{NMIW}$	200	—	ns	
$\overline{IRQ}$ setup time	$t_{IRQS}$	150	—	ns	
$\overline{IRQ}$ hold time	$t_{IRQH}$	10	—	ns	
$\overline{IRQ}$ pulse width (after leaving software standby mode or deep software standby mode)	$t_{IRQW}$	200	—	ns	

**(3) Bus Timing****Table 26.6 Bus Timing**

Conditions:  $V_{cc} = PLLV_{cc} = 3.0$  to  $3.6$  V,  $AV_{cc} = 3.0$  to  $3.6$  V,  $AV_{ccP} = AV_{ccA} = AV_{ccD} = 3.0$  to  $3.6$  V,  $V_{ref} = 3.0$  V to  $AV_{cc}$ ,  $AV_{refT} = AV_{ccA}$ ,  $V_{ss} = PLLV_{ss} = AV_{ss} = AV_{ssP} = AV_{ssA} = AV_{ssD} = AV_{refB} = 0$  V

$B\phi = 8$  to  $50$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),

$T_a = -40$  to  $+85$  °C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
Address delay time	$t_{AD}$	—	15	ns	Figures 26.9 to 26.21
Address setup time 1	$t_{AS1}$	$0.5 \times t_{cyc} - 8$	—	ns	
Address setup time 2	$t_{AS2}$	$1.0 \times t_{cyc} - 8$	—	ns	
Address setup time 3	$t_{AS3}$	$1.5 \times t_{cyc} - 8$	—	ns	
Address setup time 4	$t_{AS4}$	$2.0 \times t_{cyc} - 8$	—	ns	
Address hold time 1	$t_{AH1}$	$0.5 \times t_{cyc} - 8$	—	ns	
Address hold time 2	$t_{AH2}$	$1.0 \times t_{cyc} - 8$	—	ns	
Address hold time 3	$t_{AH3}$	$1.5 \times t_{cyc} - 8$	—	ns	
$\overline{CS}$ delay time 1	$t_{CSD1}$	—	15	ns	
$\overline{AS}$ delay time	$t_{ASD}$	—	15	ns	
$\overline{RD}$ delay time 1	$t_{RSD1}$	—	15	ns	
$\overline{RD}$ delay time 2	$t_{RSD2}$	—	15	ns	
Read data setup time 1	$t_{RDS1}$	15	—	ns	
Read data setup time 2	$t_{RDS2}$	15	—	ns	
Read data hold time 1	$t_{RDH1}$	0	—	ns	
Read data hold time 2	$t_{RDH2}$	0	—	ns	
Read data access time 2	$t_{AC2}$	—	$1.5 \times t_{cyc} - 20$	ns	
Read data access time 4	$t_{AC4}$	—	$2.5 \times t_{cyc} - 20$	ns	
Read data access time 5	$t_{AC5}$	—	$1.0 \times t_{cyc} - 20$	ns	
Read data access time 6	$t_{AC6}$	—	$2.0 \times t_{cyc} - 20$	ns	

Item	Symbol	Min.	Max.	Unit	Test Conditions
Read data access time (from address) 1	$t_{AA1}$	—	$1.0 \times t_{cyc} - 20$	ns	Figures 26.9 to 26.21
Read data access time (from address) 2	$t_{AA2}$	—	$1.5 \times t_{cyc} - 20$	ns	
Read data access time (from address) 3	$t_{AA3}$	—	$2.0 \times t_{cyc} - 20$	ns	
Read data access time (from address) 4	$t_{AA4}$	—	$2.5 \times t_{cyc} - 20$	ns	
Read data access time (from address) 5	$t_{AA5}$	—	$3.0 \times t_{cyc} - 20$	ns	
$\overline{WR}$ delay time 1	$t_{WRD1}$	—	15	ns	Figures 26.9 to 26.21
$\overline{WR}$ delay time 2	$t_{WRD2}$	—	15	ns	
$\overline{WR}$ pulse width 1	$t_{WSW1}$	$1.0 \times t_{cyc} - 13$	—	ns	
$\overline{WR}$ pulse width 2	$t_{WSW2}$	$1.5 \times t_{cyc} - 13$	—	ns	
Write data delay time	$t_{WDD}$	—	20	ns	
Write data setup time 1	$t_{WDS1}$	$0.5 \times t_{cyc} - 13$	—	ns	
Write data setup time 2	$t_{WDS2}$	$1.0 \times t_{cyc} - 13$	—	ns	
Write data setup time 3	$t_{WDS3}$	$1.5 \times t_{cyc} - 13$	—	ns	
Write data hold time 1	$t_{WDH1}$	$0.5 \times t_{cyc} - 8$	—	ns	
Write data hold time 3	$t_{WDH3}$	$1.5 \times t_{cyc} - 8$	—	ns	
Byte control delay time	$t_{UBD}$	—	15	ns	Figures 26.14, 26.15
Byte control pulse width 1	$t_{UBW1}$	—	$1.0 \times t_{cyc} - 15$	ns	Figure 26.14
Byte control pulse width 2	$t_{UBW2}$	—	$2.0 \times t_{cyc} - 15$	ns	Figure 26.15
Multiplexed address delay time 1	$t_{MAD1}$	—	15	ns	Figures 26.18, 26.19
Multiplexed address hold time	$t_{MAH}$	$1.0 \times t_{cyc} - 15$	—	ns	
Multiplexed address setup time 1	$t_{MAS1}$	$0.5 \times t_{cyc} - 15$	—	ns	
Multiplexed address setup time 2	$t_{MAS2}$	$1.5 \times t_{cyc} - 15$	—	ns	
Address hold delay time	$t_{AHD}$	—	15	ns	
Address hold pulse width 1	$t_{AHW1}$	$1.0 \times t_{cyc} - 15$	—	ns	
Address hold pulse width 2	$t_{AHW2}$	$2.0 \times t_{cyc} - 15$	—	ns	

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{\text{WAIT}}$ setup time	$t_{\text{WTS}}$	15	—	ns	Figures 26.11, 26.19
$\overline{\text{WAIT}}$ hold time	$t_{\text{WTH}}$	5.0	—	ns	
$\overline{\text{BREQ}}$ setup time	$t_{\text{BREQS}}$	20	—	ns	Figure 26.20
$\overline{\text{BACK}}$ delay time	$t_{\text{BACD}}$	—	15	ns	
Bus floating time	$t_{\text{BZD}}$	—	30	ns	
$\overline{\text{BREQO}}$ delay time	$t_{\text{BRQOD}}$	—	15	ns	Figure 26.21
$\overline{\text{BS}}$ delay time	$t_{\text{BSD}}$	1.0	15	ns	Figures 26.9, 26.10, 26.12 to 26.15
$\overline{\text{RD}}/\overline{\text{WR}}$ delay time	$t_{\text{RWD}}$	—	15	ns	

#### (4) DMAC Timing

**Table 26.7 DMAC Timing**

Conditions:  $V_{\text{cc}} = \text{PLL}V_{\text{cc}} = 3.0$  to  $3.6$  V,  $AV_{\text{cc}} = 3.0$  to  $3.6$  V,  $AV_{\text{ccP}} = AV_{\text{ccA}} = AV_{\text{ccD}} = 3.0$  to  $3.6$  V,  $V_{\text{ref}} = 3.0$  V to  $AV_{\text{cc}}$ ,  $AV_{\text{refT}} = AV_{\text{ccA}}$ ,  $V_{\text{ss}} = \text{PLL}V_{\text{ss}} = AV_{\text{ss}} = AV_{\text{ssP}} = AV_{\text{ssA}} = AV_{\text{ssD}} = AV_{\text{refB}} = 0$  V  
 $B\phi = 8$  to  $50$  MHz,  $T_{\text{a}} = -20$  to  $+75$  °C (regular specifications),  
 $T_{\text{a}} = -40$  to  $+85$  °C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{\text{DREQ}}$ setup time	$t_{\text{DRQS}}$	20	—	ns	Figure 26.22
$\overline{\text{DREQ}}$ hold time	$t_{\text{DRQH}}$	5	—	ns	
$\overline{\text{TEND}}$ delay time	$t_{\text{TED}}$	—	20	ns	Figure 26.23
$\overline{\text{DACK}}$ delay time 1	$t_{\text{DACD1}}$	—	20	ns	Figures 26.24, 26.25
$\overline{\text{DACK}}$ delay time 2	$t_{\text{DACD2}}$	—	20	ns	

**(5) On-Chip Peripheral Modules****Table 26.8 Timing of On-Chip Peripheral Modules**

Conditions:  $V_{cc} = PLLV_{cc} = 3.0$  to  $3.6$  V,  $AV_{cc} = 3.0$  to  $3.6$  V,  $AV_{ccP} = AV_{ccA} = AV_{ccD} = 3.0$  to  $3.6$  V,  $V_{ref} = 3.0$  V to  $AV_{cc}$ ,  $AV_{refT} = AV_{ccA}$ ,  $V_{ss} = PLLV_{ss} = AV_{ss} = AV_{ssP} = AV_{ssA} = AV_{ssD} = AV_{refB} = 0$  V  
 $B\phi = 8$  to  $50$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications)

	Item	Symbol	Min.	Max.	Unit	Test Conditions	
I/O ports	Output data delay time	$t_{PWD}$	—	40	ns	Figure 26.26	
	Input data setup time	$t_{PRS}$	25	—	ns		
	Input data hold time	$t_{PRH}$	25	—	ns		
TPU	Timer output delay time	$t_{TOCD}$	—	40	ns	Figure 26.26	
	Timer input setup time	$t_{TICS}$	25	—	ns		
	Timer clock input setup time	$t_{TCKS}$	25	—	ns	Figure 26.28	
	Timer clock pulse width	Single-edge setting	$t_{TCKWH}$	1.5	—	$t_{cyc}$	
		Both-edge setting	$t_{TCKWL}$	2.5	—	$t_{cyc}$	
PPG	Pulse output delay time	$t_{POD}$	—	40	ns	Figure 26.29	
8-bit timer	Timer output delay time	$t_{TMOD}$	—	40	ns	Figure 26.30	
	Timer reset input setup time	$t_{TMRS}$	25	—	ns	Figure 26.31	
	Timer clock input setup time	$t_{TMCS}$	25	—	ns	Figure 26.32	
	Timer clock pulse width	Single-edge setting	$t_{TMCWH}$	1.5	—	$t_{cyc}$	
		Both-edge setting	$t_{TMCWL}$	2.5	—	$t_{cyc}$	
WDT	Overflow output delay time	$t_{WOVD}$	—	40	ns	Figure 26.33	

	Item	Symbol	Min.	Max.	Unit	Test Conditions	
SCI	Input clock cycle	Asynchronous	$t_{S_{cyc}}$	4	—	$t_{cyc}$	Figure 26.34
		Clocked synchronous		6	—		
	Input clock pulse width	$t_{S_{CKW}}$	0.4	0.6	$t_{S_{cyc}}$		
	Input clock rise time	$t_{S_{CKr}}$	—	1.5	$t_{cyc}$		
	Input clock fall time	$t_{S_{CKf}}$	—	1.5	$t_{cyc}$		
	Transmit data delay time	$t_{TXD}$	—	40	ns	Figure 26.35	
	Receive data setup time (clocked synchronous)	$t_{RXS}$	40	—	ns		
Receive data hold time (clocked synchronous)	$t_{RXH}$	40	—	ns			
A/D converter	Trigger input setup time	$t_{TRGS}$	30	—	ns	Figure 26.36	
$\Delta\Sigma$ A/D converter	Trigger input setup time	$t_{DSTRS}$	30	—	ns	Figure 26.37	
IIC2	SCL input cycle time	$t_{SCL}$	$12 t_{cyc} + 600$	—	ns	Figure 26.38	
	SCL input high pulse width	$t_{SCLH}$	$3 t_{cyc} + 300$	—	ns		
	SCL input low pulse width	$t_{SCLL}$	$5 t_{cyc} + 300$	—	ns		
	SCL, SDA input falling time	$t_{Sf}$	—	300	ns		
	SCL, SDA input spike pulse removal time	$t_{SP}$	—	$1 t_{cyc}$	ns		
	SDA input bus free time	$t_{BUF}$	$5 t_{cyc}$	—	ns		
	Start condition input hold time	$t_{STAH}$	$3 t_{cyc}$	—	ns		
	Retransmit start condition input setup time	$t_{STAS}$	$3 t_{cyc}$	—	ns		
	Stop condition input setup time	$t_{STOS}$	$1 t_{cyc} + 20$	—	ns		
	Data input setup time	$t_{SDAS}$	0	—	ns		
	Data input hold time	$t_{SDAH}$	0	—	ns		
	SCL, SDA capacitive load	$C_b$	—	400	pF		
	SCL, SDA falling time	$t_{Sf}$	—	300	ns		

### 26.1.4 A/D Conversion Characteristics

**Table 26.9 A/D Conversion Characteristics**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CC} = 3.0$  to  $3.6$  V,  $V_{REF} = 3.0$  V to  $AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = AV_{SS} = 0$  V,  $P\phi = 8$  to  $35$  MHz,  
 $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit
Resolution	10	10	10	Bit
Conversion time	5.33	—	—	$\mu$ s
Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	5	k $\Omega$
Nonlinearity error	—	—	$\pm 3.5$	LSB
Offset error	—	—	$\pm 3.5$	LSB
Full-scale error	—	—	$\pm 3.5$	LSB
Quantization error	—	$\pm 0.5$	—	LSB
Absolute accuracy	—	—	$\pm 4.0$	LSB

### 26.1.5 D/A Conversion Characteristics

**Table 26.10 D/A Conversion Characteristics**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CC} = 3.0$  to  $3.6$  V,  $V_{REF} = 3.0$  V to  $AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = AV_{SS} = 0$  V,  $P\phi = 8$  to  $35$  MHz,  
 $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit	Test Conditions
Resolution	8	8	8	Bit	
Conversion time	—	—	10	$\mu$ s	20 pF capacitive load
Absolute accuracy	—	$\pm 2.0$	$\pm 3.0$	LSB	2 M $\Omega$ resistive load
	—	—	$\pm 2.0$	LSB	4 M $\Omega$ resistive load



## 26.1.6 $\Delta\Sigma$ A/D Conversion Characteristics

**Table 26.11  $\Delta\Sigma$  A/D Conversion Characteristics (Reference Value) (1)**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CCP} = AV_{CCA} = AV_{CCD} = 3.0$  to  $3.6$  V,  
 $AV_{REFT} = AV_{CCA}$ ,  $V_{SS} = PLLV_{SS} = AV_{SSP} = AV_{SSA} = AV_{SSD} = AV_{REFB} = 0$  V,  $P\phi = 8$  to  $35$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications),  $REXT = 51K \Omega^*$

Item	Condition	min	typ	max	Unit	
Output word length		—	16	—	bit	
Number of states for conversion (in fos)		—	286	—	cyc	
Oversampling frequency (fos)		2.5	—	3.3	MHz	
Conversion time		86.67	—	114.40	$\mu$ s	
Input frequency		—	—	1.0	KHz	
Input voltage range (with respect to input offset voltage)	Single-ended	$\times 8$ gain mode	—	—	$\pm 1/12 \times AV_{REFT}$	V
		$\times 4$ gain mode	—	—	$\pm 1/6 \times AV_{REFT}$	V
		$\times 2$ gain mode	—	—	$\pm 1/3 \times AV_{REFT}$	V
		$\times 1$ gain mode	—	—	$\pm 1/2 \times AV_{REFT}$	V
	Differential	$\times 8$ gain mode	—	—	$\pm 1/24 \times AV_{REFT}$	V
		$\times 4$ gain mode	—	—	$\pm 1/12 \times AV_{REFT}$	V
		$\times 2$ gain mode	—	—	$\pm 1/6 \times AV_{REFT}$	V
		$\times 1$ gain mode	—	—	$\pm 1/3 \times AV_{REFT}$	V
Input offset voltage	$\times 8$ gain mode	$1/4 \times AV_{REFT}$	—	$3/4 \times AV_{REFT}$	V	
	$\times 4$ gain mode	$1/4 \times AV_{REFT}$	—	$3/4 \times AV_{REFT}$	V	
	$\times 2$ gain mode	—	$1/2 \times AV_{REFT}$	—	V	
	$\times 1$ gain mode	—	$1/2 \times AV_{REFT}$	—	V	

Note: \* It is recommended to use a 1%-error resistor as the external biasing resistor connected on the REXT pin.

**Table 26.11  $\Delta\Sigma$  A/D Conversion Characteristics (Reference Value) (2)**

Conditions:  $V_{CC} = PLLV_{CC} = 3.0$  to  $3.6$  V,  $AV_{CCP} = AV_{CCA} = AV_{CCD} = 3.0$  to  $3.6$  V,  
 $AV_{refT} = AV_{CCA}$ ,  $V_{SS} = PLLV_{SS} = AV_{SSP} = AV_{SSA} = AV_{SSD} = AV_{refB} =$   
 $0$  V,  $P\phi = 8$  to  $35$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications),  $REXT = 51K \Omega^*$

Conditions									
Item	Channel	Gain Mode	Input Voltage	Input Offset Voltage	Other Conditions	min	typ	max	Unit
Gain error	Single-ended	×8	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Sine wave input= up to 1 kHz	—	±2	—	%
		×4	$\pm 1/6 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	fos= 3.125 MHz	—	±2	—	%
		×2	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±2	—	%
		×1	$\pm 1/2 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±2	—	%
	Differential	×8	$\pm 1/24 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Sine wave input= up to 1 kHz	—	±2	—	%
		×4	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	fos= 3.125 MHz	—	±2	—	%
		×2	$\pm 1/6 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±2	—	%
		×1	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±2	—	%
SNDR	Single-ended	×8	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Sine wave input= up to 1 kHz	—	83	—	dB
		×4	$\pm 1/6 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	fos= 3.125 MHz	—	84	—	dB
		×2	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	87	—	dB
		×1	$\pm 1/2 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	84	—	dB
	Differential	×8	$\pm 1/24 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Sine wave input= up to 1 kHz	—	85	—	dB
		×4	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	fos= 3.125 MHz	—	86	—	dB
		×2	$\pm 1/6 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	88	—	dB
		×1	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	88	—	dB

Conditions									
Item	Channel	Gain Mode	Input Voltage	Input Offset Voltage	Other Conditions	min	typ	max	Unit
DNL	Single-ended	×8	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Ramp wave input fos = 3.125 MHz	—	±1.3	—	LSB
		×4	$\pm 1/6 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$		—	±1.2	—	LSB
		×2	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±0.9	—	LSB
		×1	$\pm 1/2 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±0.9	—	LSB
	Differential	×8	$\pm 1/24 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Ramp wave input fos = 3.125 MHz	—	±1.1	—	LSB
		×4	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$		—	±1.0	—	LSB
		×2	$\pm 1/6 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±0.8	—	LSB
		×1	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±0.8	—	LSB
INL	Single-ended	×8	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Ramp wave input fos = 3.125 MHz	—	±5.0	—	LSB
		×4	$\pm 1/6 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$		—	±4.5	—	LSB
		×2	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±3.0	—	LSB
		×1	$\pm 1/2 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±3.0	—	LSB
	Differential	×8	$\pm 1/24 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	Ramp wave input fos = 3.125 MHz	—	±4.0	—	LSB
		×4	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$		—	±3.5	—	LSB
		×2	$\pm 1/6 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±2.5	—	LSB
		×1	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		—	±2.5	—	LSB

Item	Channel	Conditions				Other Conditions	min	typ	max	Unit
		Gain Mode	Input Voltage	Input Offset Voltage						
Input impedance	Single-ended	×8	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	fos = 3.125 MHz	20	—	—	k $\Omega$	
		×4	$\pm 1/6 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$		40	—	—	k $\Omega$	
		×2	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		80	—	—	k $\Omega$	
		×1	$\pm 1/2 \times AV_{refT}$	$1/2 \times AV_{refT}$		160	—	—	k $\Omega$	
	Differential	×8	$\pm 1/24 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$	fos = 3.125 MHz	20	—	—	k $\Omega$	
		×4	$\pm 1/12 \times AV_{refT}$	$1/4 \times AV_{refT}$ to $3/4 \times AV_{refT}$		40	—	—	k $\Omega$	
		×2	$\pm 1/6 \times AV_{refT}$	$1/2 \times AV_{refT}$		80	—	—	k $\Omega$	
		×1	$\pm 1/3 \times AV_{refT}$	$1/2 \times AV_{refT}$		160	—	—	k $\Omega$	

Note: \* It is recommended to use a 1%-error resistor as the external biasing resistor connected on the REXT pin.

**Table 26.11  $\Delta\Sigma$  A/D Conversion Characteristics (Reference Value) (3)**

Conditions:  $V_{cc} = PLLV_{cc} = 3.0$  to  $3.6$  V,  $AV_{ccP} = AV_{ccA} = AV_{ccD} = 3.0$  to  $3.6$  V,  
 $AV_{refT} = AV_{ccA}$ ,  $V_{ss} = PLLV_{ss} = AV_{ssP} = AV_{ssA} = AV_{ssD} = AV_{refB} = 0$  V,  $P\phi = 8$  to  $35$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications),  $R_{EXT} = 51K \Omega^*$

Item	Conditions	min	typ	max	Unit
Offset cancellation resolution	×8 gain mode	—	10	—	bit
Offset cancellation absolute accuracy	×8 gain mode	—	±2.0	—	LSB

Note: \* It is recommended to use a 1%-error resistor as the external biasing resistor connected on the REXT pin.

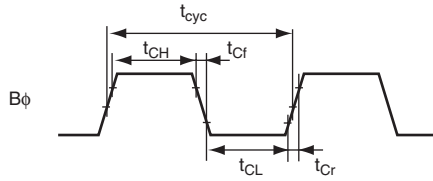
**Table 26.11  $\Delta\Sigma$  A/D Conversion Characteristics (Reference Value) (4)**

Conditions:  $V_{cc} = PLLV_{cc} = 3.0$  to  $3.6$  V,  $AV_{ccP} = AV_{ccA} = AV_{ccD} = 3.0$  to  $3.6$  V,  
 $AV_{refT} = AV_{ccA}$ ,  $V_{ss} = PLLV_{ss} = AV_{ssP} = AV_{ssA} = AV_{ssD} = AV_{refB} =$   
 $0$  V,  $P\phi = 8$  to  $35$  MHz,  $T_a = -20$  to  $+75$  °C (regular specifications),  
 $T_a = -40$  to  $+85$  °C (wide-range specifications),  $REXT = 51K \Omega^*$

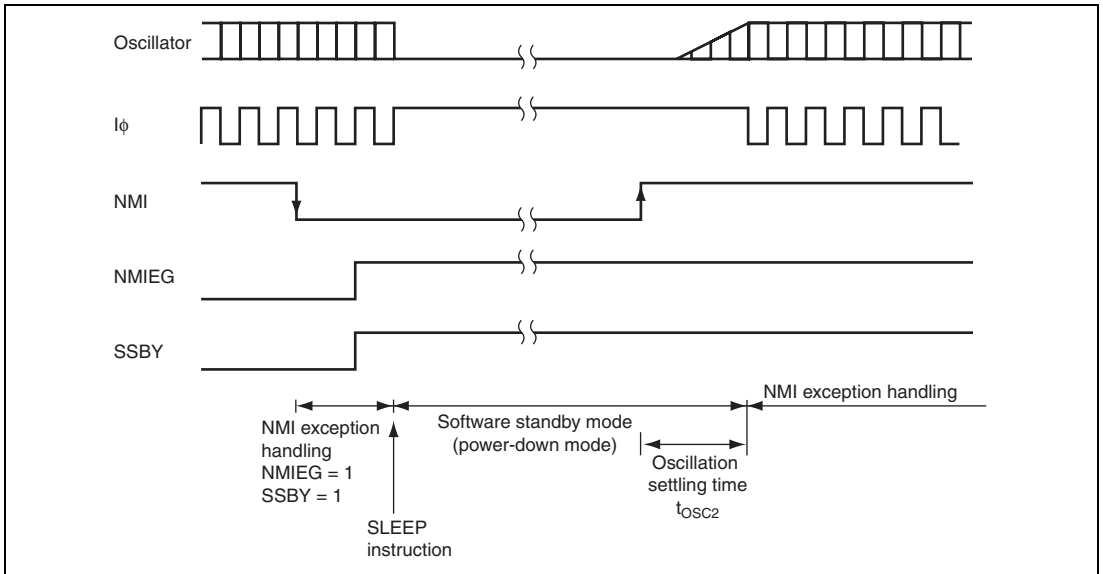
Item	Conditions		min	typ	max	Unit
Supply current (during conversion)	$AV_{ccA} + AV_{ccD} + AV_{ccP}$	$f_{os} = 3.125$ MHz	—	3.5	5	mA
Supply current (standby state)	$AV_{ccA} + AV_{ccD} + AV_{ccP}$	—	—	0.5	5	$\mu$ A
Stabilization time (AVCM)	Stabilization time from the point the BIASE bit is set	$AV_{CM} = 0.1 \mu$ F	20	—	—	mS

Note: \* It is recommended to use a 1%-error resistor as the external biasing resistor connected on the REXT pin.

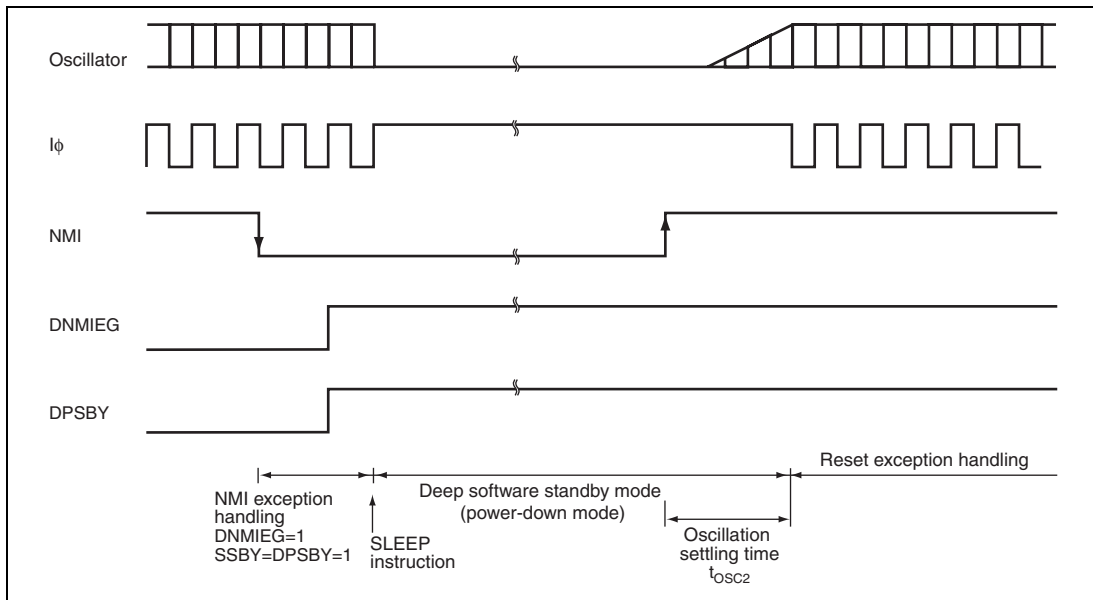
## 26.2 Timing Charts



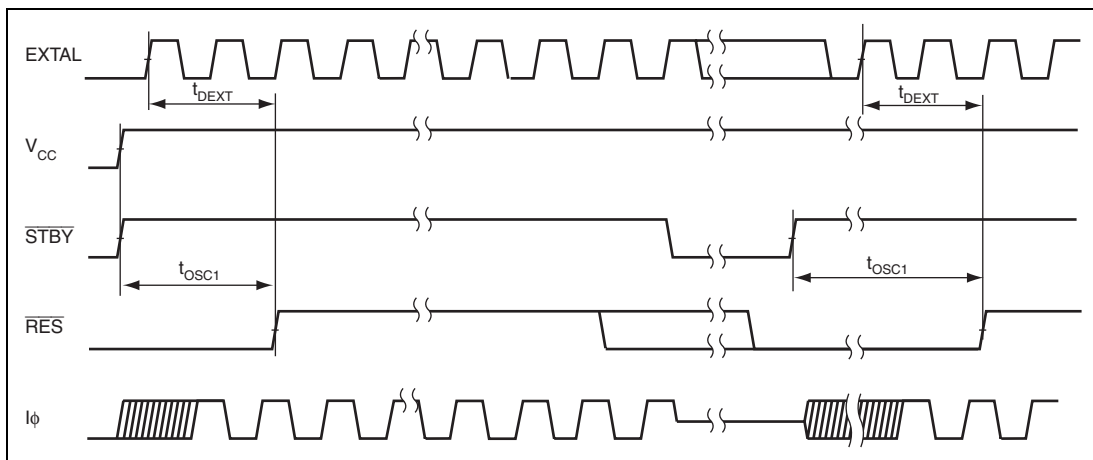
**Figure 26.2 External Bus Clock Timing**



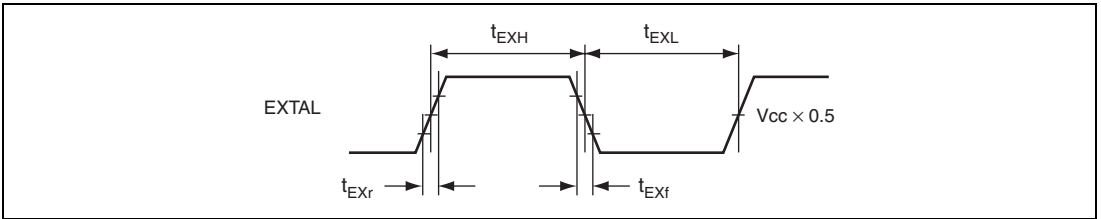
**Figure 26.3 Oscillation Settling Timing after Software Standby Mode**



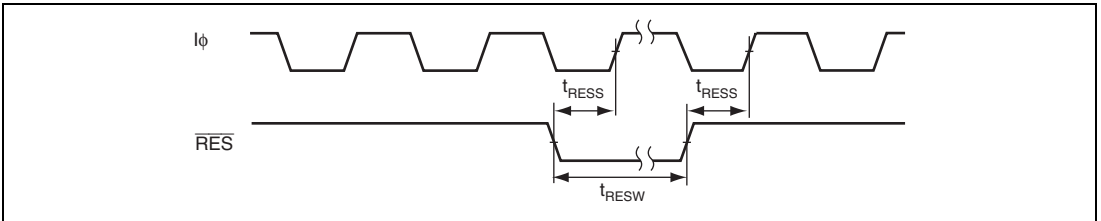
**Figure 26.4 Oscillation Settling Timing after Deep Software Standby Mode**



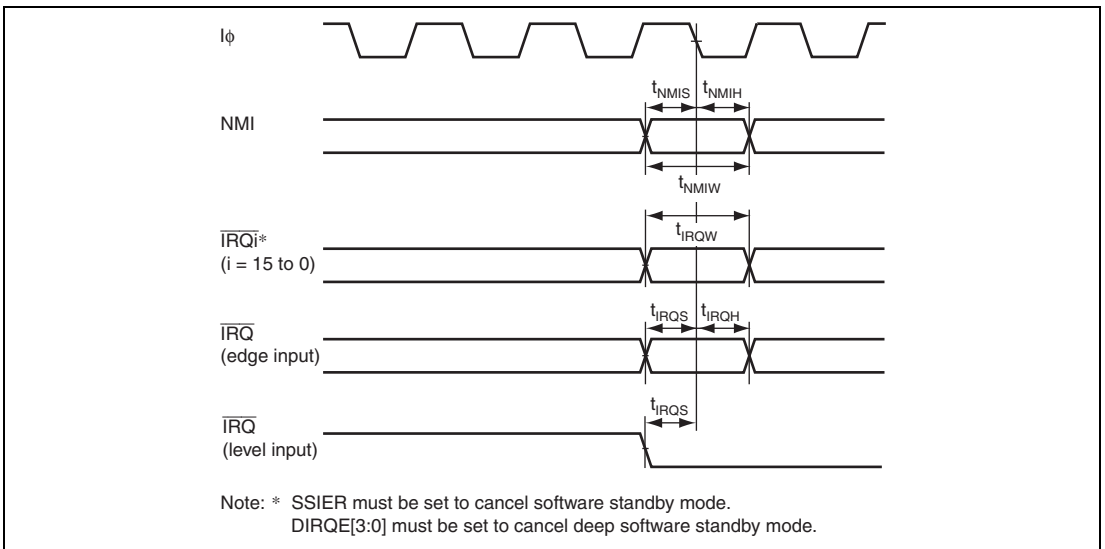
**Figure 26.5 Oscillation Settling Timing**



**Figure 26.6 External Input Clock Timing**



**Figure 26.7 Reset Input Timing**



**Figure 26.8 Interrupt Input Timing**



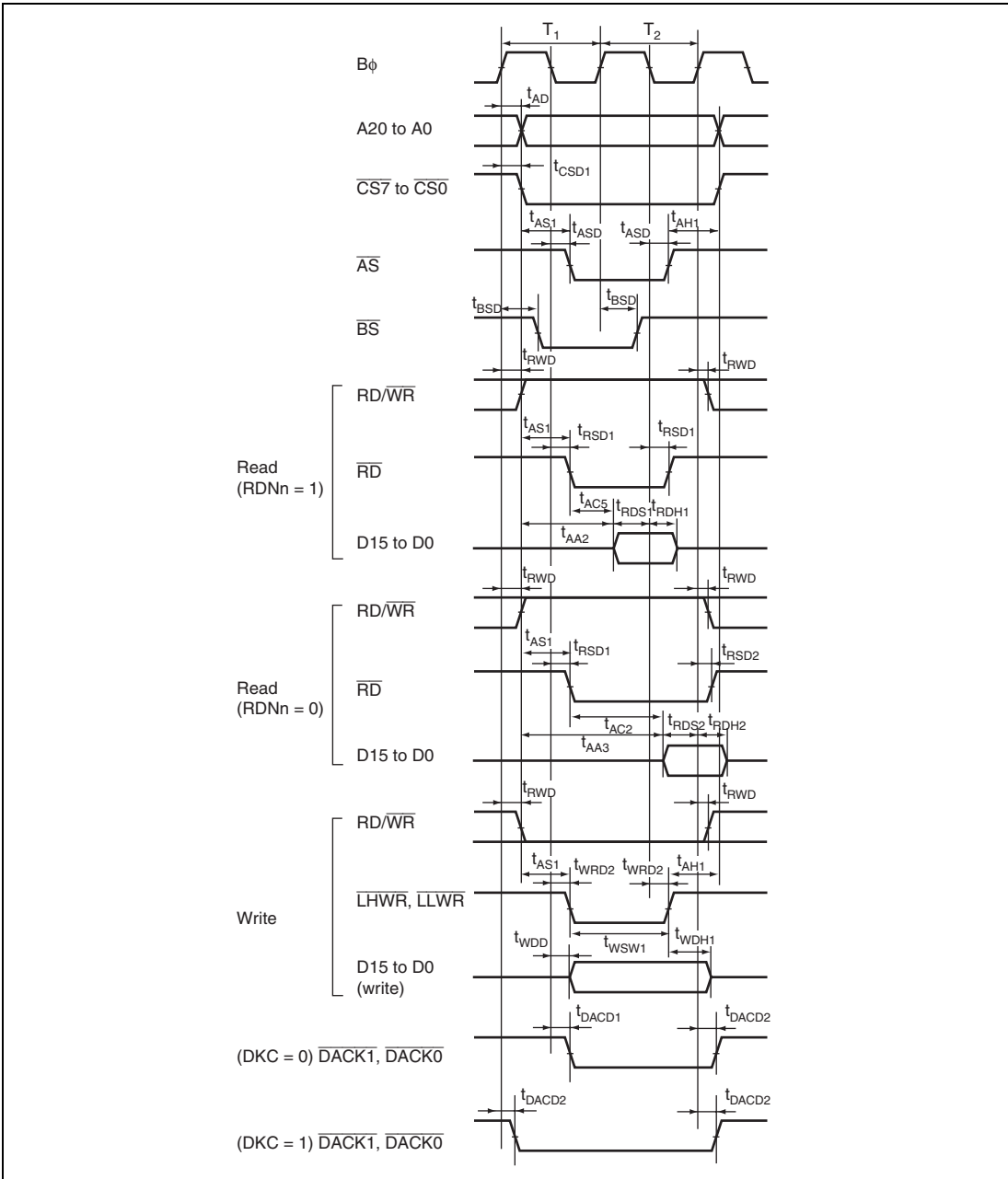


Figure 26.9 Basic Bus Timing: 2-State Access

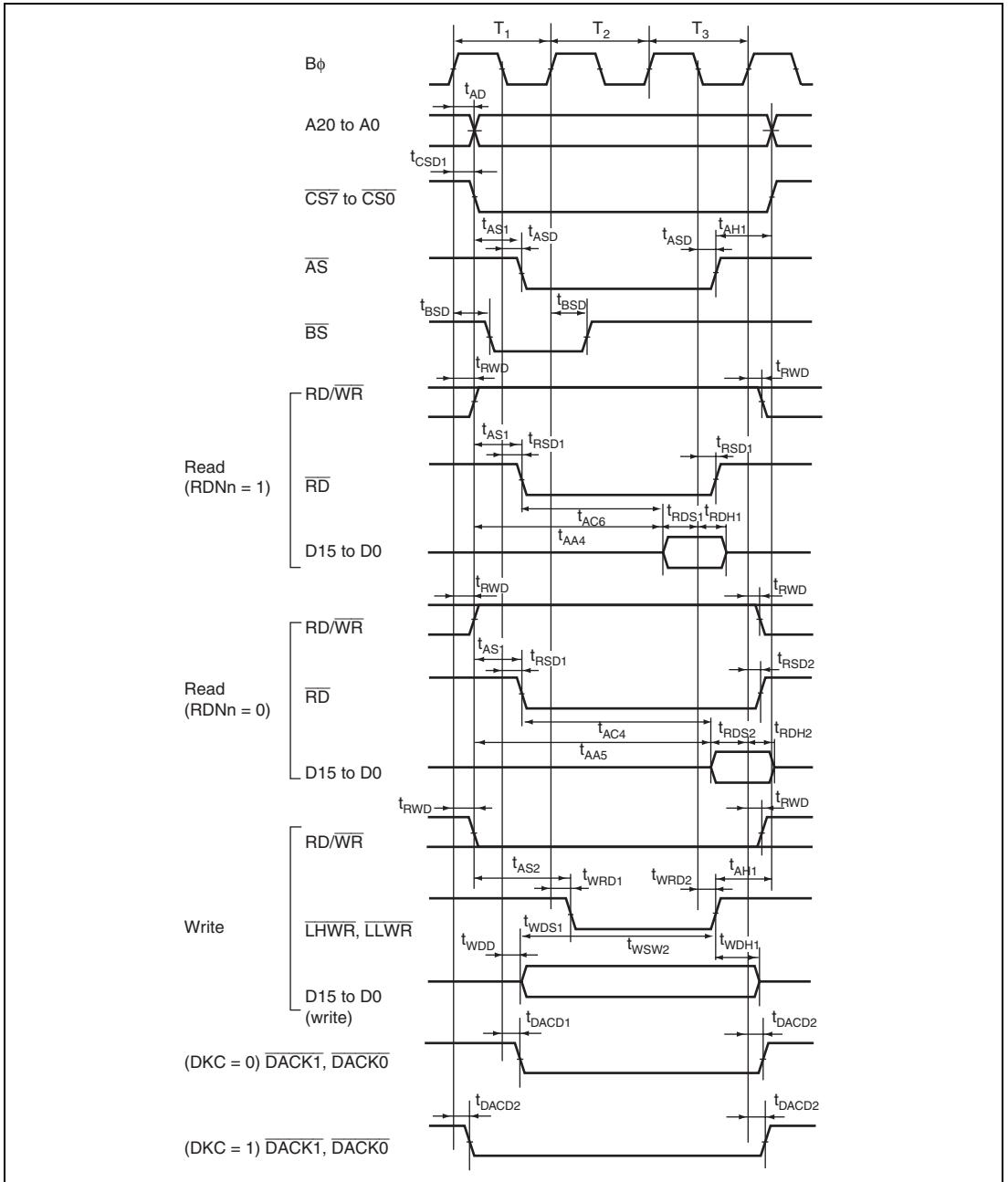
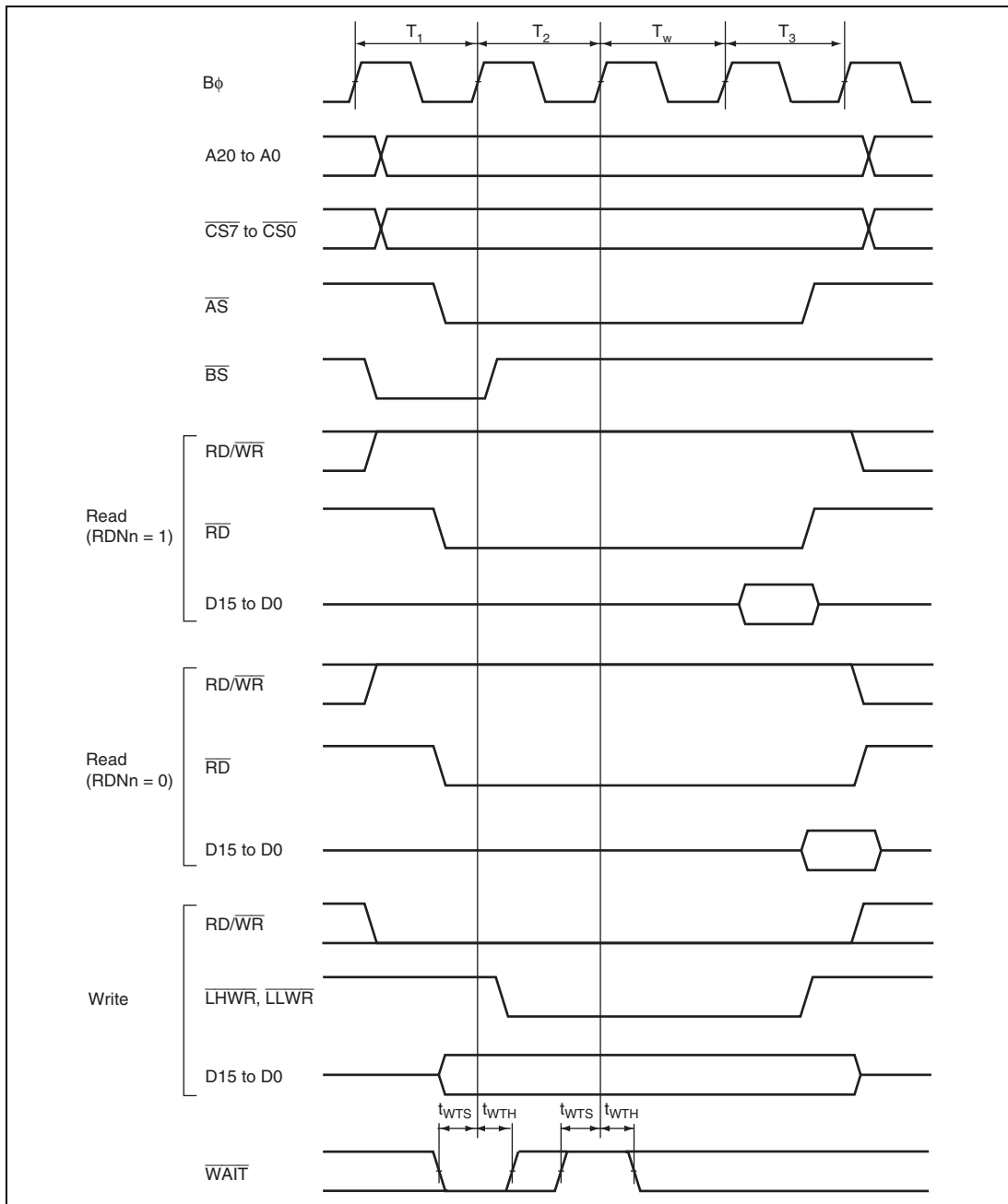


Figure 26.10 Basic Bus Timing: 3-State Access



**Figure 26.11 Basic Bus Timing: Three-State Access, One Wait**

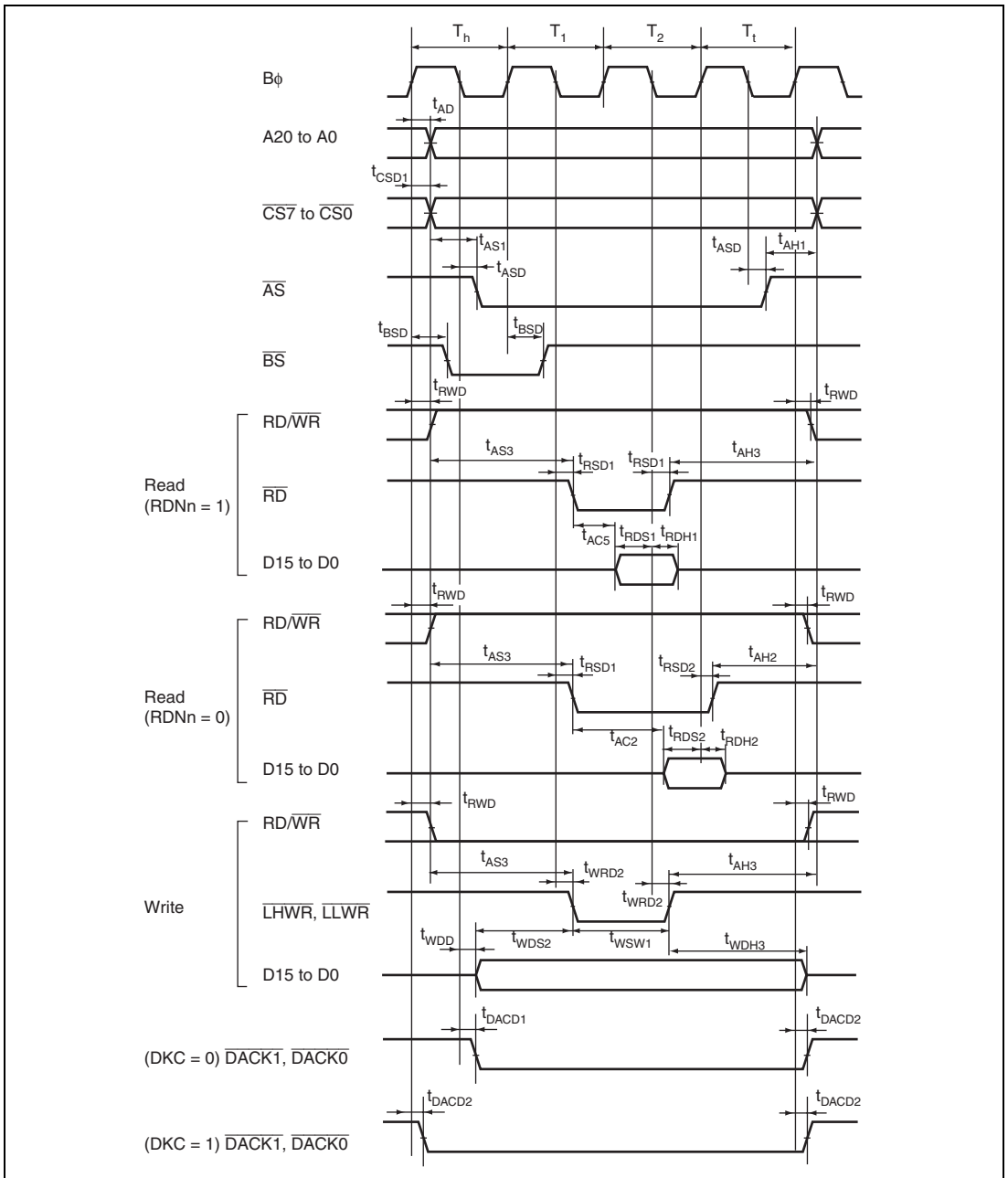


Figure 26.12 Basic Bus Timing: 2-State Access ( $\overline{CS}$  Assertion Period Extended)

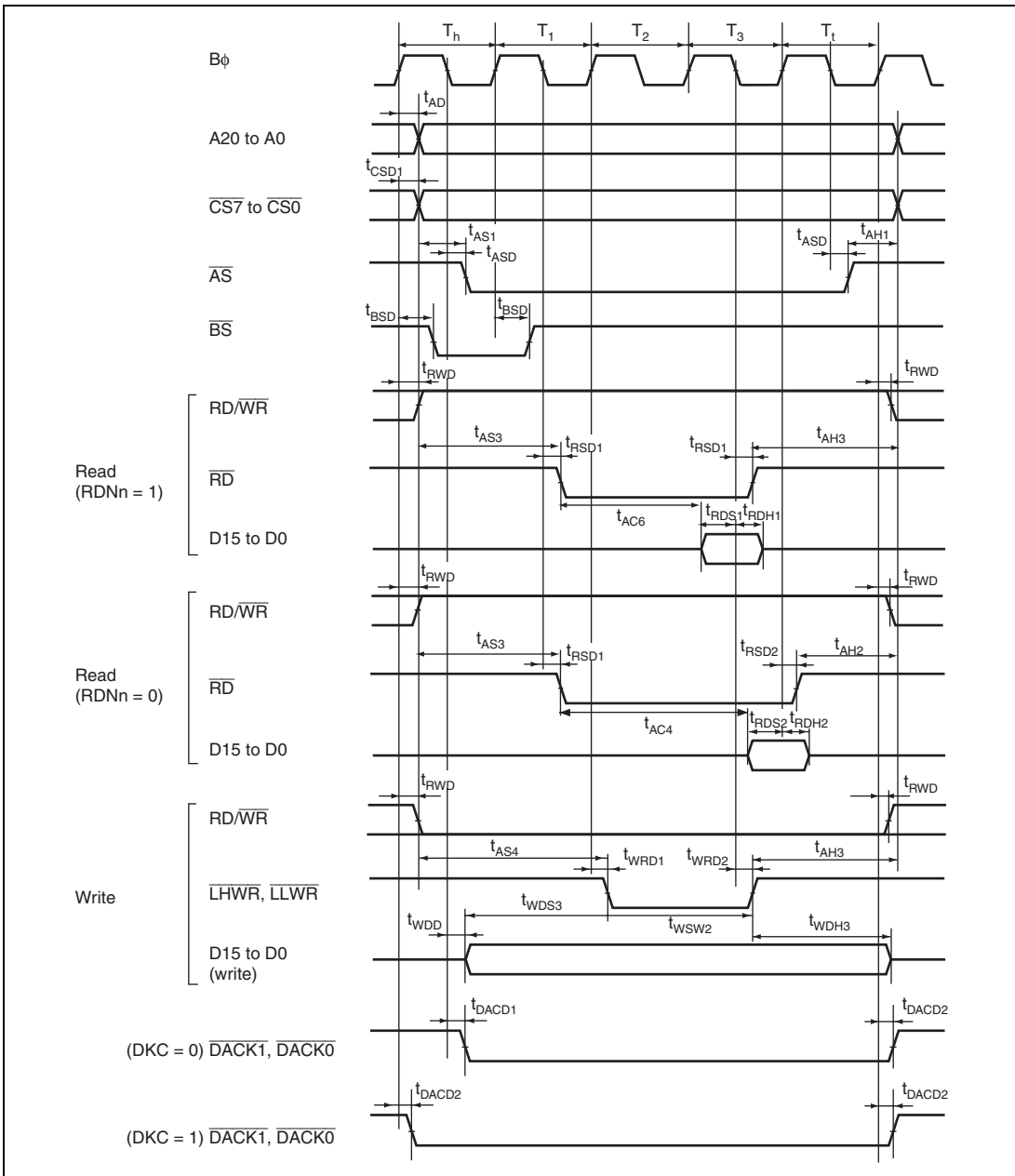


Figure 26.13 Basic Bus Timing: 3-State Access ( $\overline{CS}$  Assertion Period Extended)

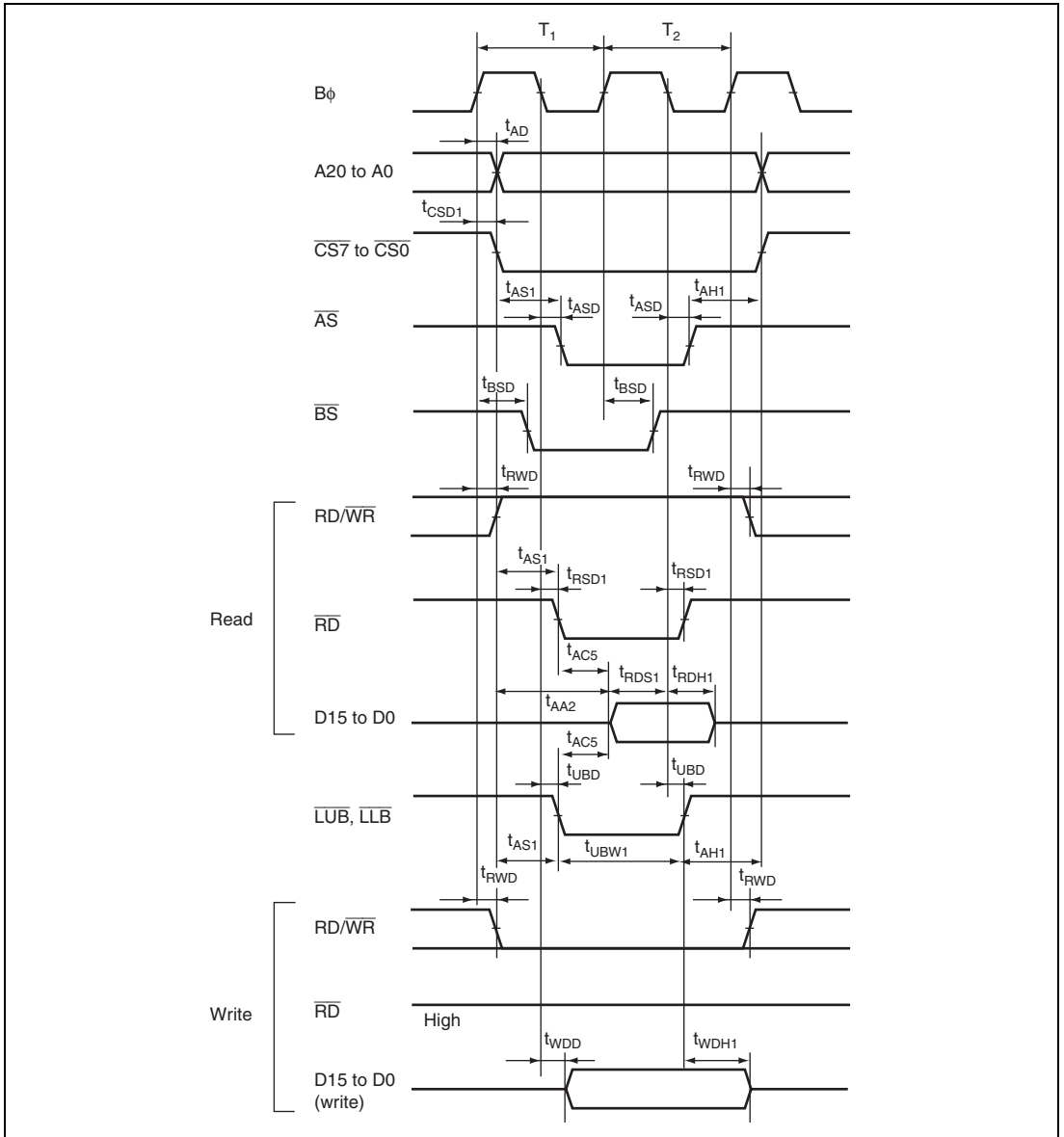


Figure 26.14 Byte Control SRAM: 2-State Read/Write Access

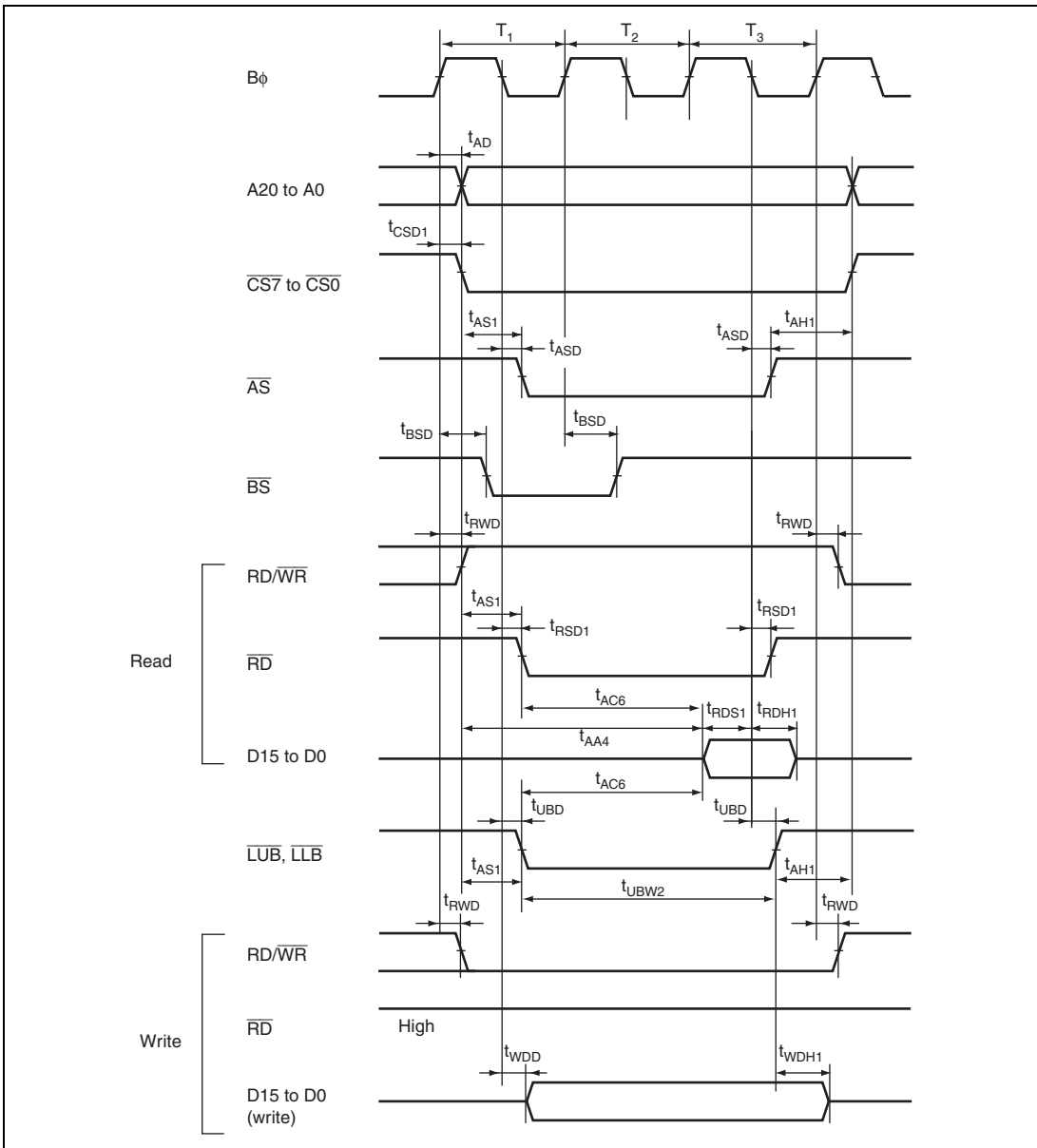
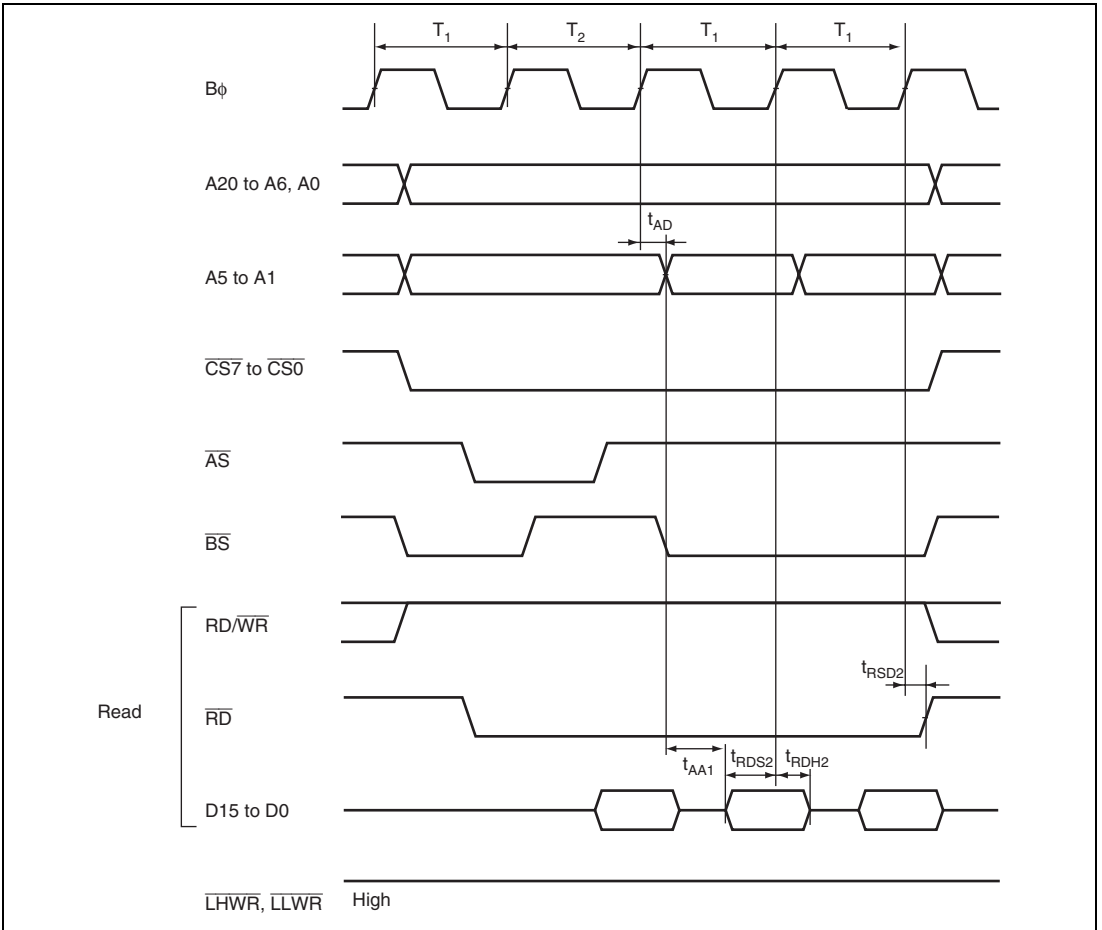
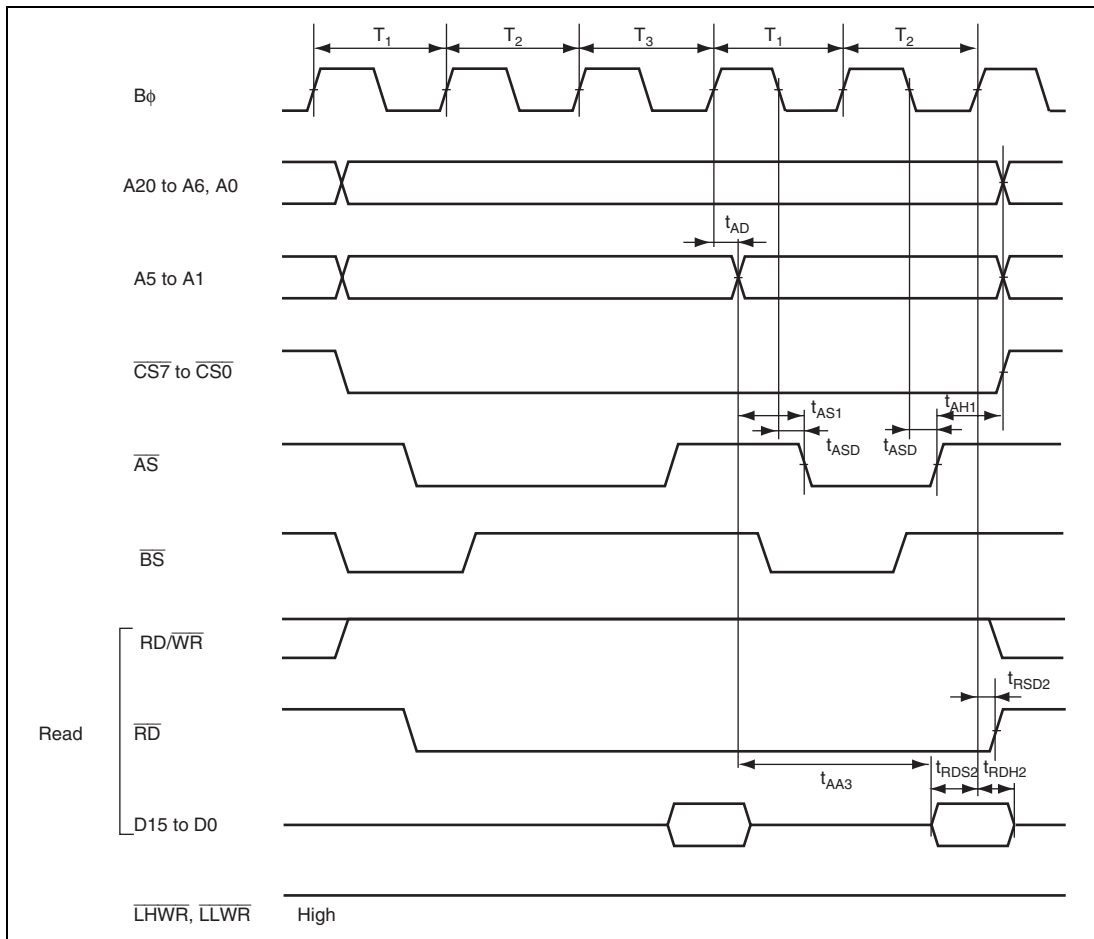


Figure 26.15 Byte Control SRAM: 3-State Read/Write Access

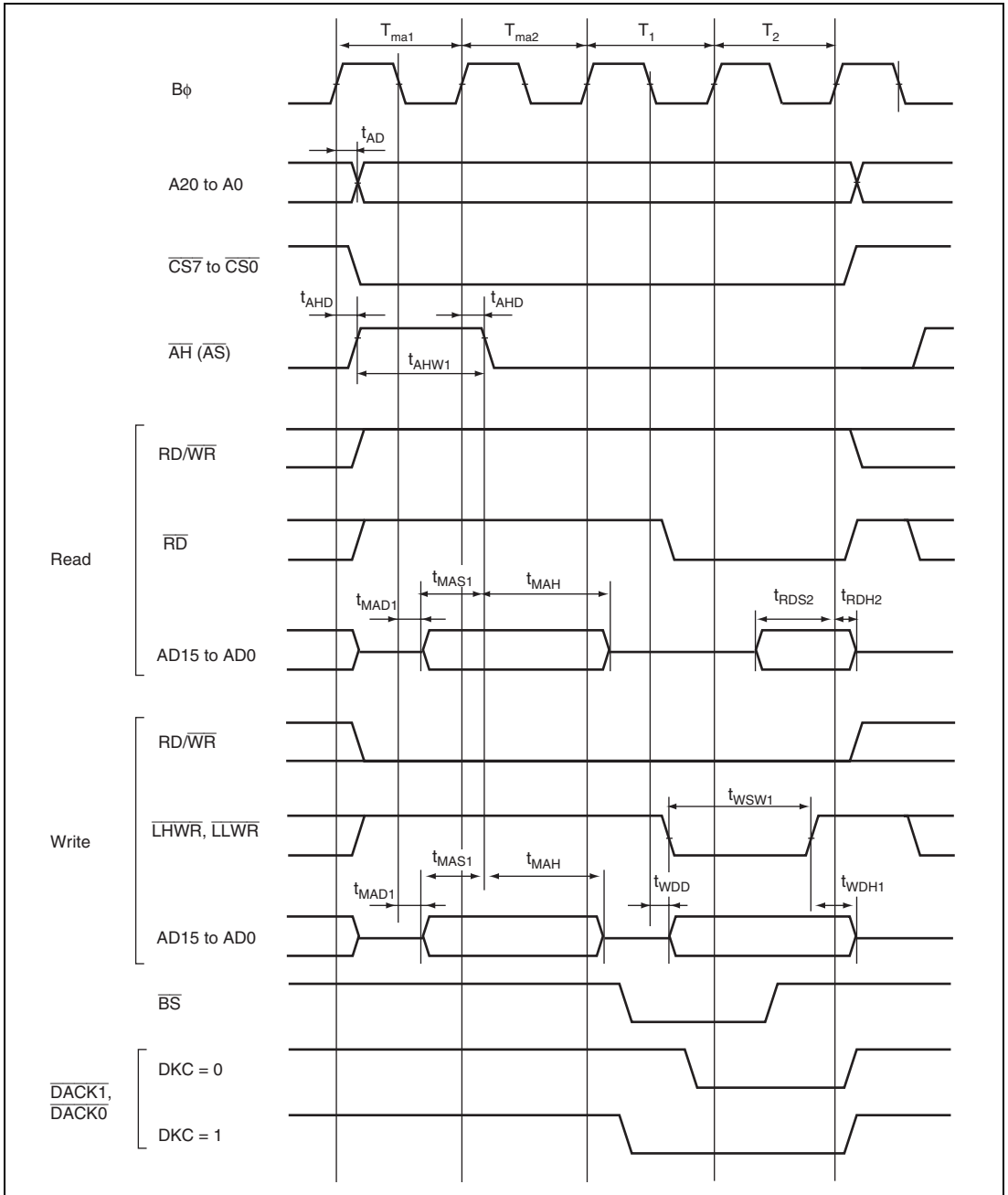


**Figure 26.16 Burst ROM Access Timing: 1-State Burst Access**

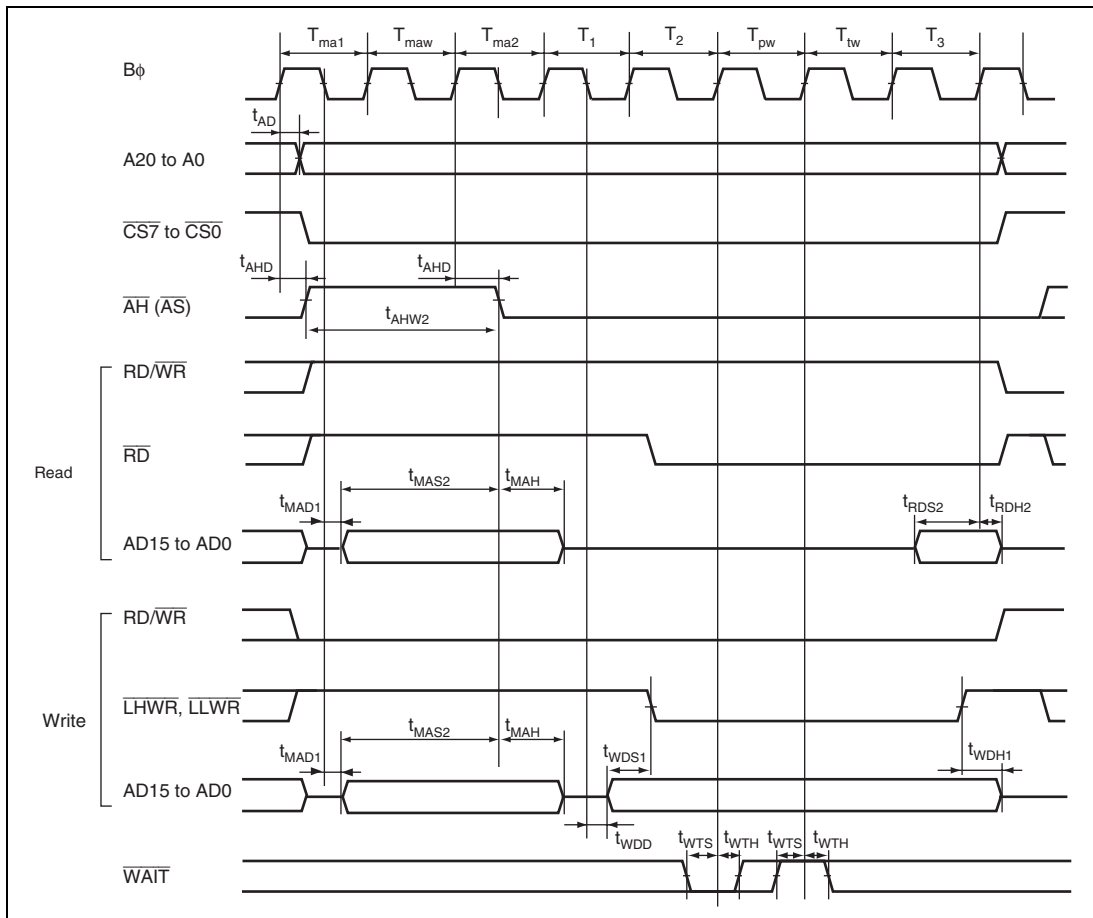




**Figure 26.17 Burst ROM Access Timing: 2-State Burst Access**



**Figure 26.18 Address/Data Multiplexed Access Timing (No Wait) (Basic, 4-State Access)**



**Figure 26.19 Address/Data Multiplexed Access Timing (Wait Control)**  
 (Address Cycle Program Wait × 1 + Data Cycle Program Wait × 1 +  
 Data Cycle Pin Wait × 1)

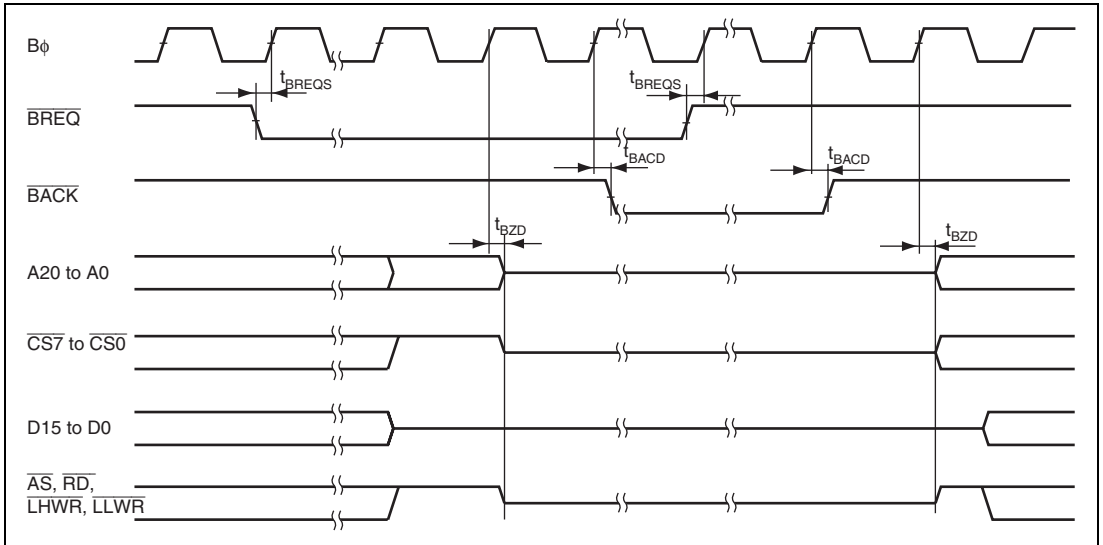


Figure 26.20 External Bus Release Timing

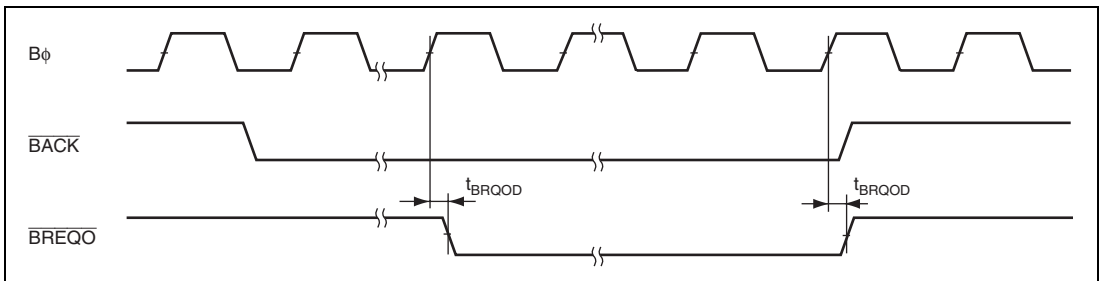


Figure 26.21 External Bus Request Output Timing

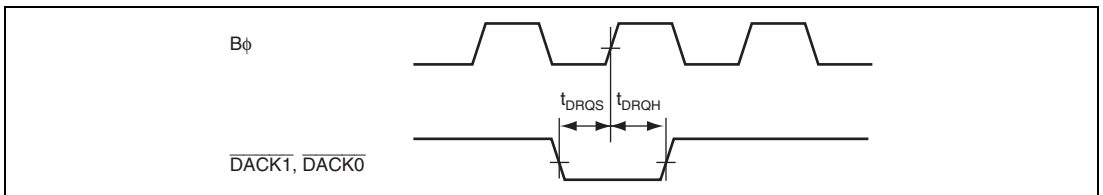
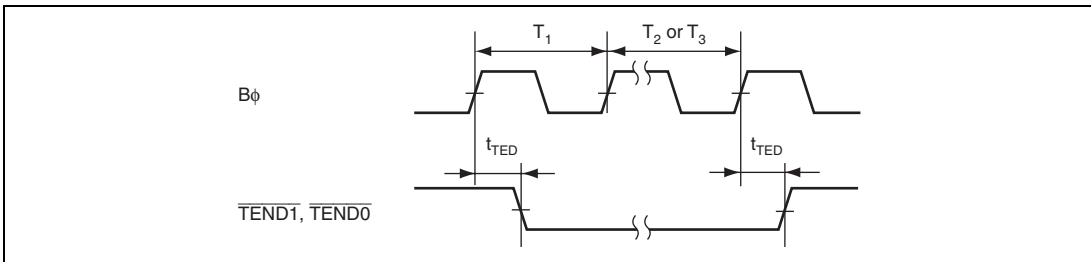
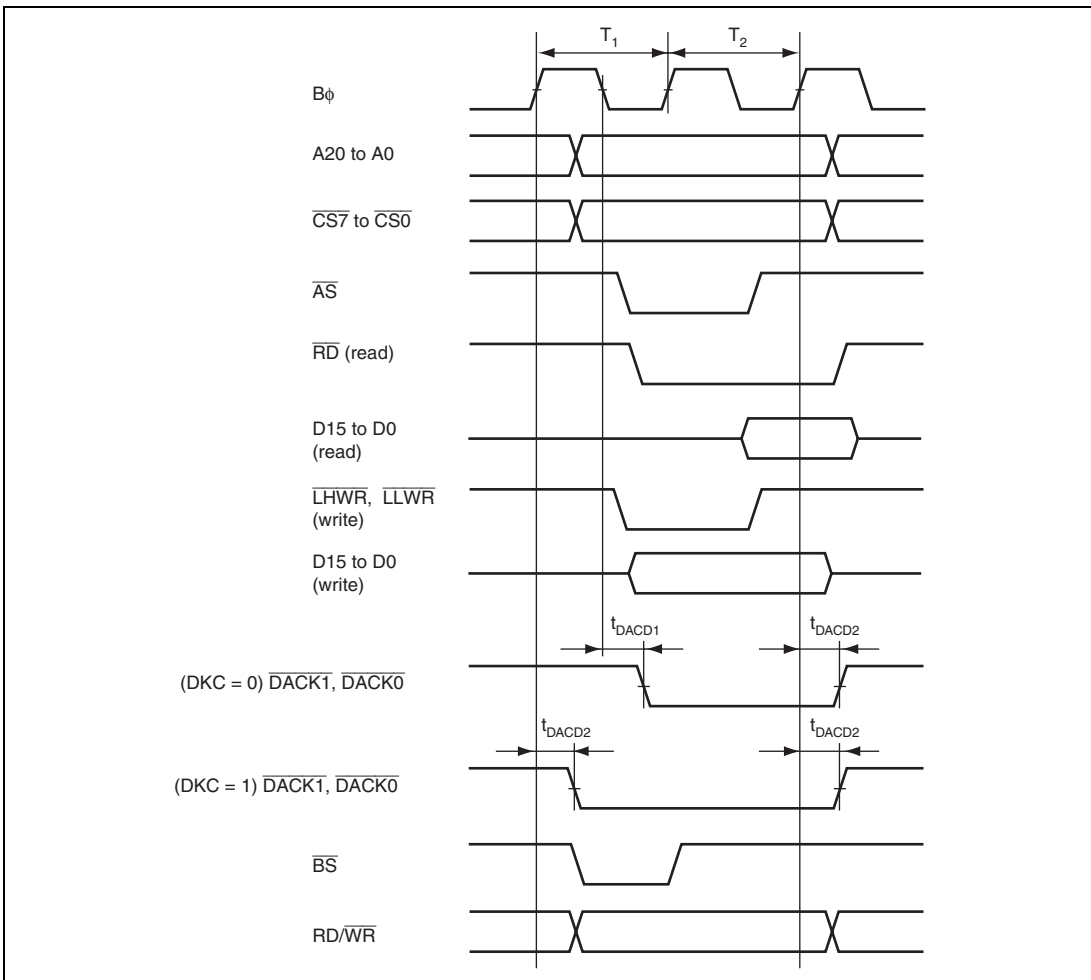


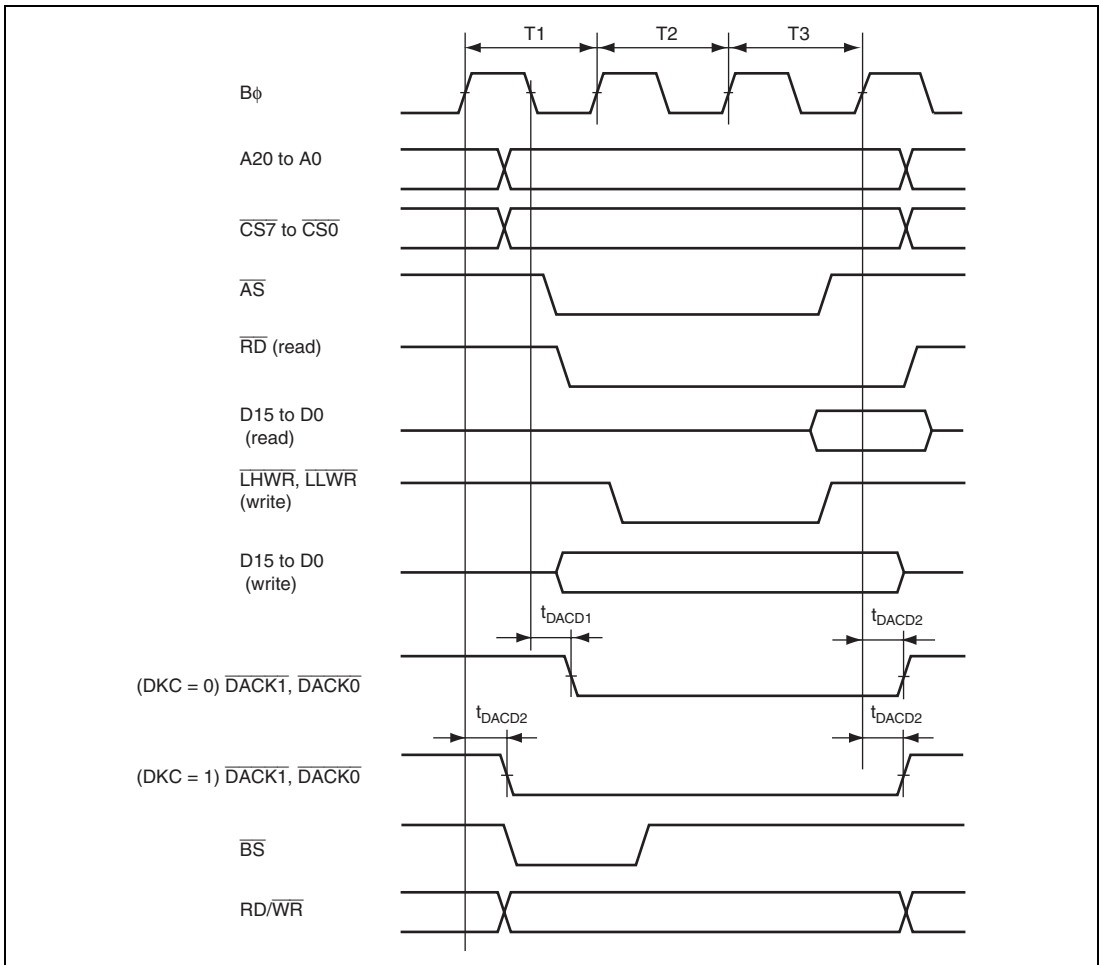
Figure 26.22 DMAC,  $\overline{DREQ}$  Input Timing



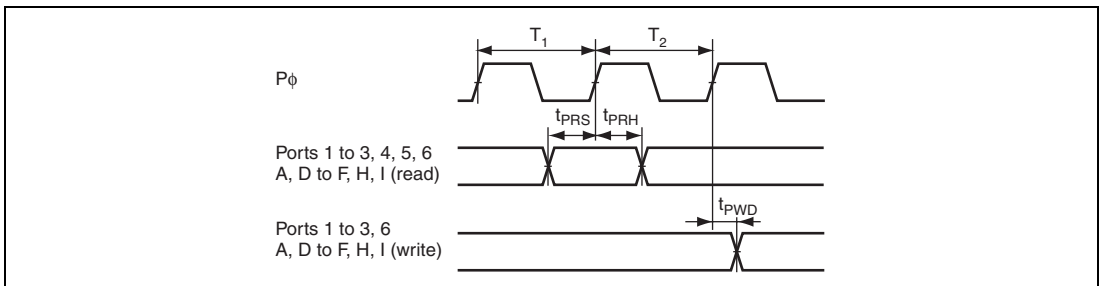
**Figure 26.23 DMAC,  $\overline{TEND}$  Output Timing**



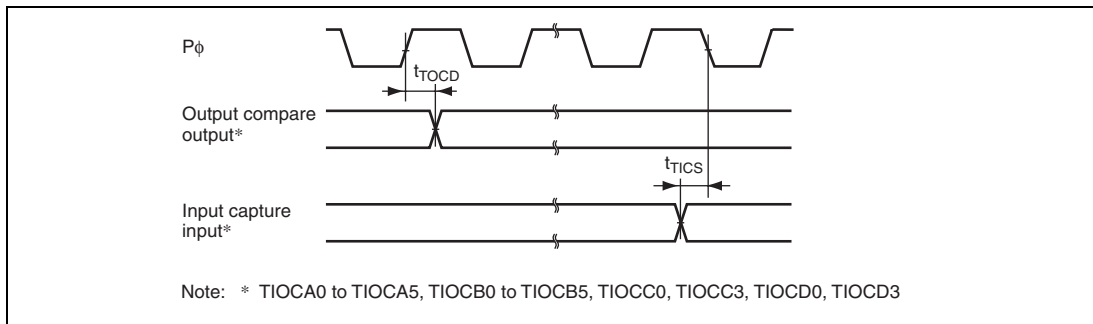
**Figure 26.24 DMAC Single Address Transfer Timing: 2-State Access**



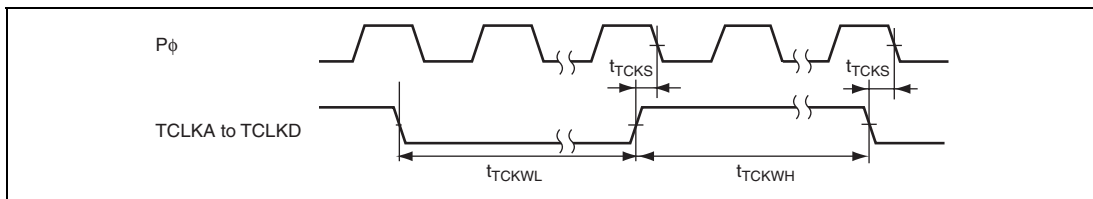
**Figure 26.25 DMAC Single Address Transfer Timing: 3-State Access**



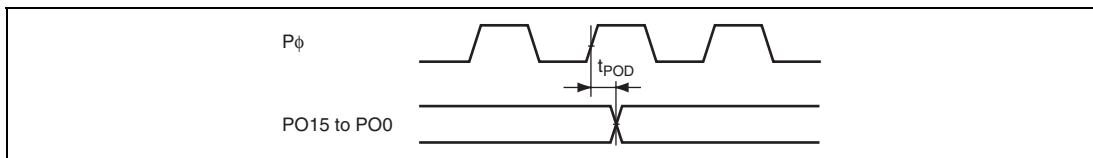
**Figure 26.26 I/O Port Input/Output Timing**



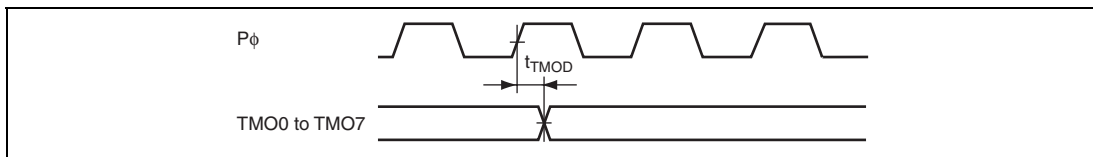
**Figure 26.27 TPU Input/Output Timing**



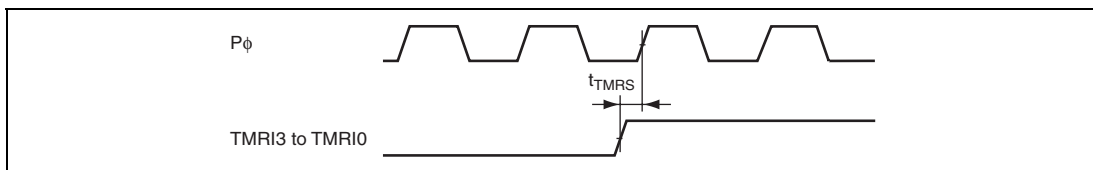
**Figure 26.28 TPU Clock Input Timing**



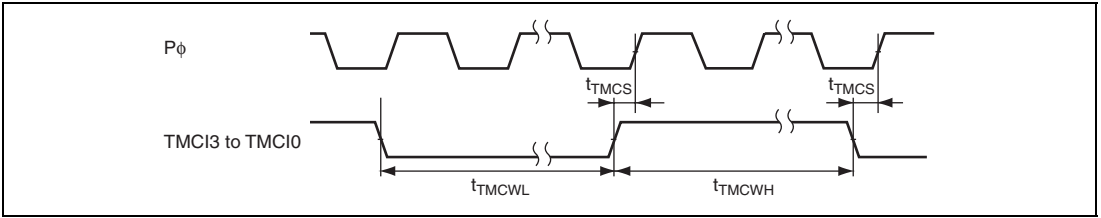
**Figure 26.29 PPG Output Timing**



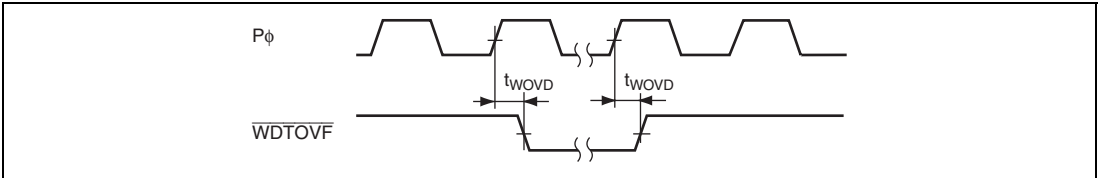
**Figure 26.30 8-Bit Timer Output Timing**



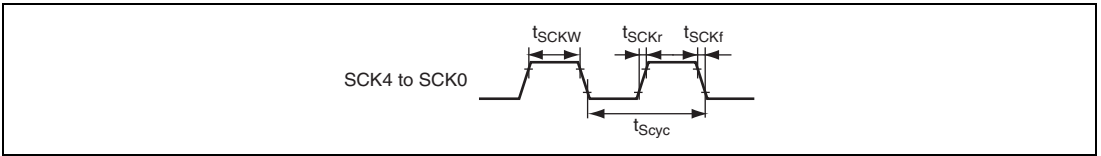
**Figure 26.31 8-Bit Timer Reset Input Timing**



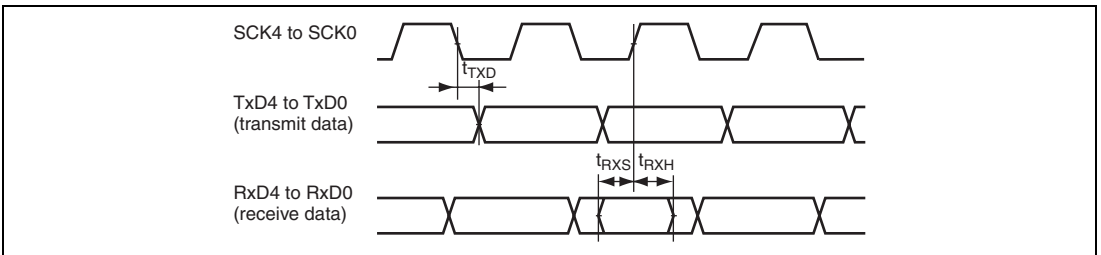
**Figure 26.32 8-Bit Timer Clock Input Timing**



**Figure 26.33 WDT Output Timing**

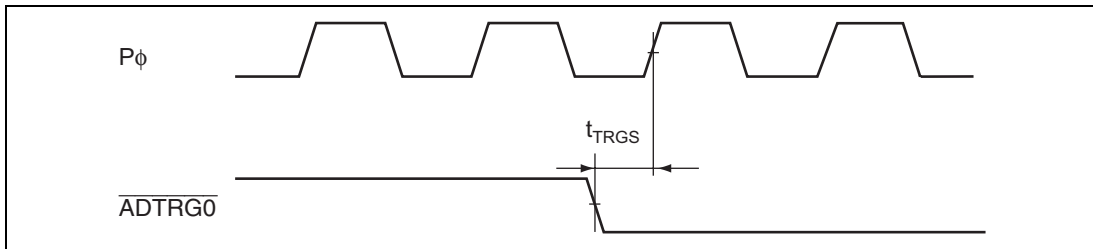


**Figure 26.34 SCK Clock Input Timing**

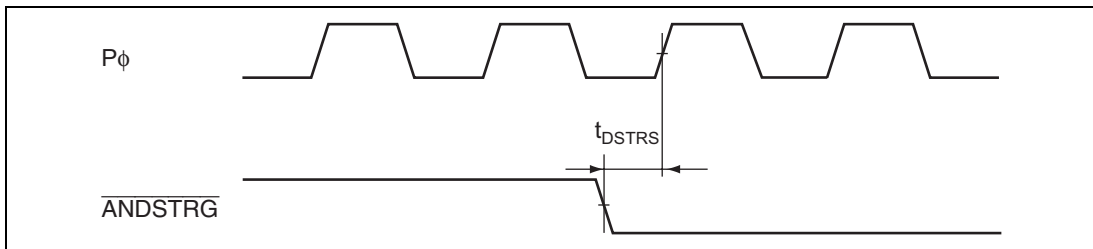


**Figure 26.35 SCI Input/Output Timing: Clocked Synchronous Mode**

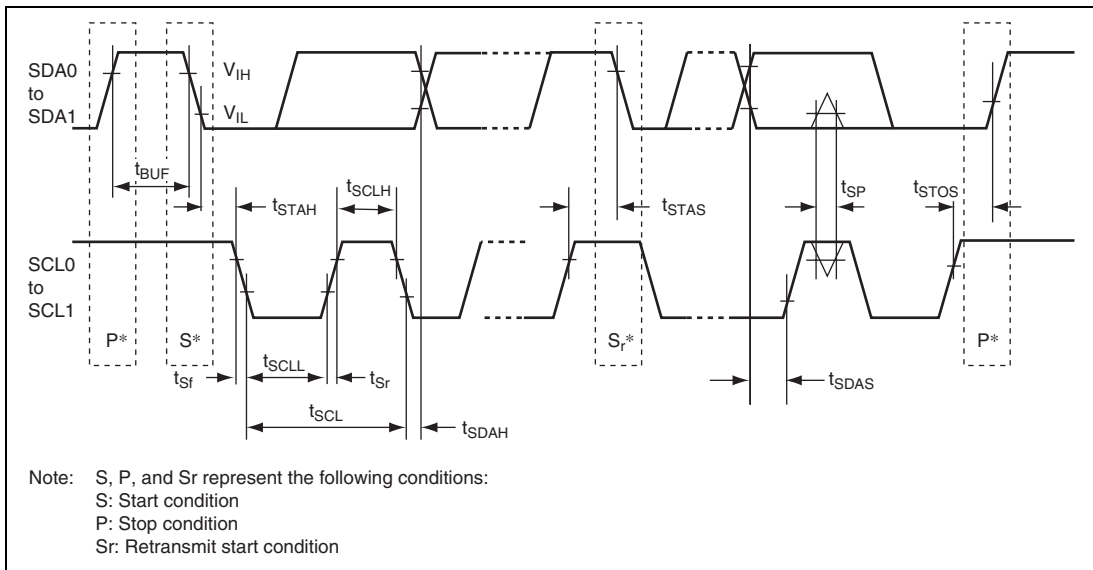




**Figure 26.36 A/D Converter External Trigger Input Timing**



**Figure 26.37 ΔΣ A/D Converter External Trigger Input Timing**



**Figure 26.38 I²C Bus Interface 2 Input/Output Timing**

## 26.3 Flash Memory Characteristics

**Table 26.12 Flash Memory Characteristics**

Conditions:  $V_{cc} = PLLV_{cc} = 3.0$  to  $3.6$  V,  $AV_{cc} = 3.0$  to  $3.6$  V,  $AV_{ccP} = AV_{ccA} = AV_{ccD} = 3.0$  to  $3.6$  V,  $V_{ref} = 3.0$  V to  $AV_{cc}$ ,  $AV_{refT} = AV_{ccA}$ ,  $V_{ss} = PLLV_{ss} = AV_{ss} = AV_{ssP} = AV_{ssA} = AV_{ssD} = AV_{refB} = 0$  V,  
 $I\phi = 8$  to  $50$  MHz,  $P\phi = 8$  to  $35$  MHz  
 Operating temperature range during programming/erasing:  
 $T_a = 0$  to  $+75$  °C (regular specifications),  
 $T_a = 0$  to  $+85$  °C (wide-range specifications)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time* <sup>1, *2, *4</sup>	$t_p$	—	1	10	ms/128 bytes	
Erasure time* <sup>1, *2, *4</sup>	$t_E$	—	40	130	ms/4k byte block	
		—	300	800	ms/32k byte block	
		—	600	1500	ms/64k byte block	
Programming time (total)* <sup>1, *2, *4</sup>	$\Sigma_{IP}$	—	2.3	6	s/256k bytes	$T_a = 25^\circ\text{C}$ , for all 0s
Erasure time (total)* <sup>1, *2, *4</sup>	$\Sigma_{IE}$	—	2.3	6	s/256k bytes	$T_a = 25^\circ\text{C}$
Programming, Erasure time (total)* <sup>1, *2, *4</sup>	$\Sigma_{IPE}$	—	4.6	12	s/256k bytes	$T_a = 25^\circ\text{C}$
Overwrite count	$N_{WEC}$	100* <sup>3</sup>	—	—	times	
Data save time* <sup>4</sup>	$T_{DRP}$	10	—	—	Year	

- Notes: 1. Programming time and erase time depend on data in the flash memory.  
 2. Programming time and erase time do not include time for data transfer.  
 3. All the characteristics after programming are guaranteed within this value (guaranteed value is from 1 to Min. value).  
 4. Characteristics when programming is performed within the Min. value

# Appendix

## A. Port States in Each Pin State

**Table A.1 Port States in Each Pin State**

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Deep Software Standby Mode IOKEEP = 1/0		Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	OPE = 1	OPE = 0	
Port 1	All	HiZ	HiZ	Keep		Keep		
Port 2	All	HiZ	HiZ	Keep		Keep		
P30/PO8/TIOCA0/ DREQ0-B/CS0/ CS4-A/CS5-B	Single-chip mode (EXPE = 0)	HiZ	HiZ	[CS output] H	[CS output] HiZ	[CS output] H	[CS output] HiZ	
	External extended mode (EXPE = 1)	H	HiZ	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	
P31/PO9/TIOCA0/ TIOCB0/TEND0-B/ CS1/CS2-B/CS5-A/ CS6-B/CS7-B	Single-chip mode (EXPE = 0)	HiZ	HiZ	[CS output] H	[CS output] HiZ	[CS output] H	[CS output] HiZ	
	External extended mode (EXPE = 1)	HiZ	HiZ	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	
P32/PO10/TIOCC0/ TCLKA-A/DACK0-B/ CS2-A/CS6-A	Single-chip mode (EXPE = 0)	HiZ	HiZ	[CS output] H	[CS output] HiZ	[CS output] H	[CS output] HiZ	
	External extended mode (EXPE = 1)	HiZ	HiZ	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	
P33/PO11/TIOCC0/ TIOCD0/TCLKB-A/ DREQ1-B/CS3/ CS7-A	Single-chip mode (EXPE = 0)	HiZ	HiZ	[CS output] H	[CS output] HiZ	[CS output] H	[CS output] HiZ	
	External extended mode (EXPE = 1)	HiZ	HiZ	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	[Other than above] Keep	
P34 to P37	All	HiZ	HiZ	Keep		Keep		
P40 to P47	All	HiZ	HiZ	HiZ		HiZ		Keep
P50 to P55	All	HiZ	HiZ	HiZ		HiZ		Keep

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Deep Software Standby Mode IOKEEP = 1/0		Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	OPE = 1	OPE = 0	
P56/AN6/DA0/IRQ6-B	All	HiZ	HiZ	HiZ		[DAOE0 = 1] Keep [DAOE0 = 0] HiZ		Keep
P57/AN7/DA1/IRQ7-B	All	HiZ	HiZ	HiZ		[DAOE1=1] Keep [DAOE1=0] HiZ		Keep
P60 to P65	All	HiZ	HiZ	Keep		Keep		
PA0/BREQO/BS-A	All	HiZ	HiZ	[BREQO output] HiZ		[BREQO output] HiZ		[BREQO output] BREQO [BS output] HiZ [Other than above] Keep
				[BS output] Keep	[BS output] HiZ	[BS output] Keep	[BS output] HiZ	
				[Other than above] Keep		[Other than above] Keep		
PA1/BACK/(RD/WR)	All	HiZ	HiZ	[BACK output] HiZ		[BACK output] HiZ		[BACK output] BACK
				[RD/WR output] Keep	[RD/WR output] HiZ	[RD/WR output] Keep	[RD/WR output] HiZ	
				[Other than above] Keep		[Other than above] Keep		
PA2/BREQ/WAIT	All	HiZ	HiZ	[BREQ input] HiZ [WAIT input] HiZ [Other than above] Keep	[BREQ input] HiZ [WAIT input] HiZ [Other than above] Keep	[BREQ input] HiZ (BREQ) [WAIT input] HiZ (WAIT)		
PA3/LLWR/LLB	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep		Keep
	External extended mode (EXPE = 1)	H	HiZ	H	HiZ	H	HiZ	

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Deep Software Standby Mode IOKEEP = 1/0		Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	OPE = 1	OPE = 0	
PA4/LHWR/LUB	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep		Keep
	External extended mode (EXPE = 1)	H	HiZ	[LHWR, LUB output] H [Other than above] Keep	[LHWR, LUB output] HiZ [Other than above] Keep	[LHWR, LUB output] H [Other than above] Keep	[LHWR, LUB output] HiZ [Other than above] Keep	
PA5/RD	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep		
	External extended mode (EXPE = 1)	H	HiZ	H	HiZ	H	HiZ	
PA6/AS/AH/BS-B	Single-chip mode (EXPE = 0)	HiZ	HiZ	[AS, BS output] H	[AS, AH, BS output] HiZ	[AS, BS output] H	[AS, AH, BS output] HiZ	
	External extended mode (EXPE = 1)	H	HiZ	[AH output] L [Other than above] Keep	[Other than above] Keep	[AH output] L [Other than above] Keep	[Other than above] Keep	
PA7/B $\phi$	Single-chip mode (EXPE = 0)	HiZ	HiZ	[Clock output] H		[Clock output] H		[Clock output] Clock output
	External extended mode (EXPE = 1)	Clock output	HiZ	[Other than above] Keep		[Other than above] Keep		[Other than above] Keep

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Deep Software Standby Mode IOKEEP = 1/0		Software Standby Mode		Bus Released State
				OPE = 1	OPE = 0	OPE = 1	OPE = 0	
Port D	External extended mode (EXPE = 1)	L	HiZ	Keep	HiZ	Keep	HiZ	
	ROM enabled extended mode	HiZ	HiZ	Keep	[Address output] HiZ [Other than above] Keep	Keep	[Address output] HiZ [Other than above] Keep	
	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep		
Port E	External extended mode (EXPE = 1)	L	HiZ	Keep	HiZ	Keep	HiZ	
	ROM enabled extended mode	HiZ	HiZ	Keep	[Address output] HiZ [Other than above] Keep	Keep	[Address output] HiZ [Other than above] Keep	
	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep		
PF0 to PF4	External extended mode (EXPE = 1)	L	HiZ	Keep	HiZ	Keep	HiZ	
	ROM enabled extended mode	HiZ	HiZ	Keep	[Address output] HiZ [Other than above] Keep	Keep	[Address output] HiZ [Other than above] Keep	
	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep		

Port Name	MCU Operating Mode	Reset	Hardware Standby Mode	Deep Software Standby Mode IOKEEP = 1/0		Software Standby Mode		Bus Released State	
				OPE = 1	OPE = 0	OPE = 1	OPE = 0		
Port H	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep			
	External extended mode (EXPE = 1)	HiZ	HiZ	HiZ		HiZ			
Port I	Single-chip mode (EXPE = 0)	HiZ	HiZ	Keep		Keep			
	External extended mode (EXPE = 1)	8-bit bus mode	HiZ	HiZ	Keep		Keep		
		16-bit bus mode	HiZ	HiZ	HiZ		HiZ		
		32-bit bus mode	HiZ	HiZ	HiZ		HiZ		

## B. Product Lineup

<b>Product Classification</b>	<b>Product Model</b>	<b>Marking</b>	<b>Package (Package Code)</b>
H8SX/1622	R5F61622LGV	R5F61622	PTLG0145JB-A (TLP-145V)*
H8SX/1622	R5F61622FPV	R5F61622	PLQP0144KA-A (FP-144LV)*

Note: \* Pb-free version



## C. Package Dimensions

For the package dimensions, data in the Renesas IC Package General Catalog has priority.

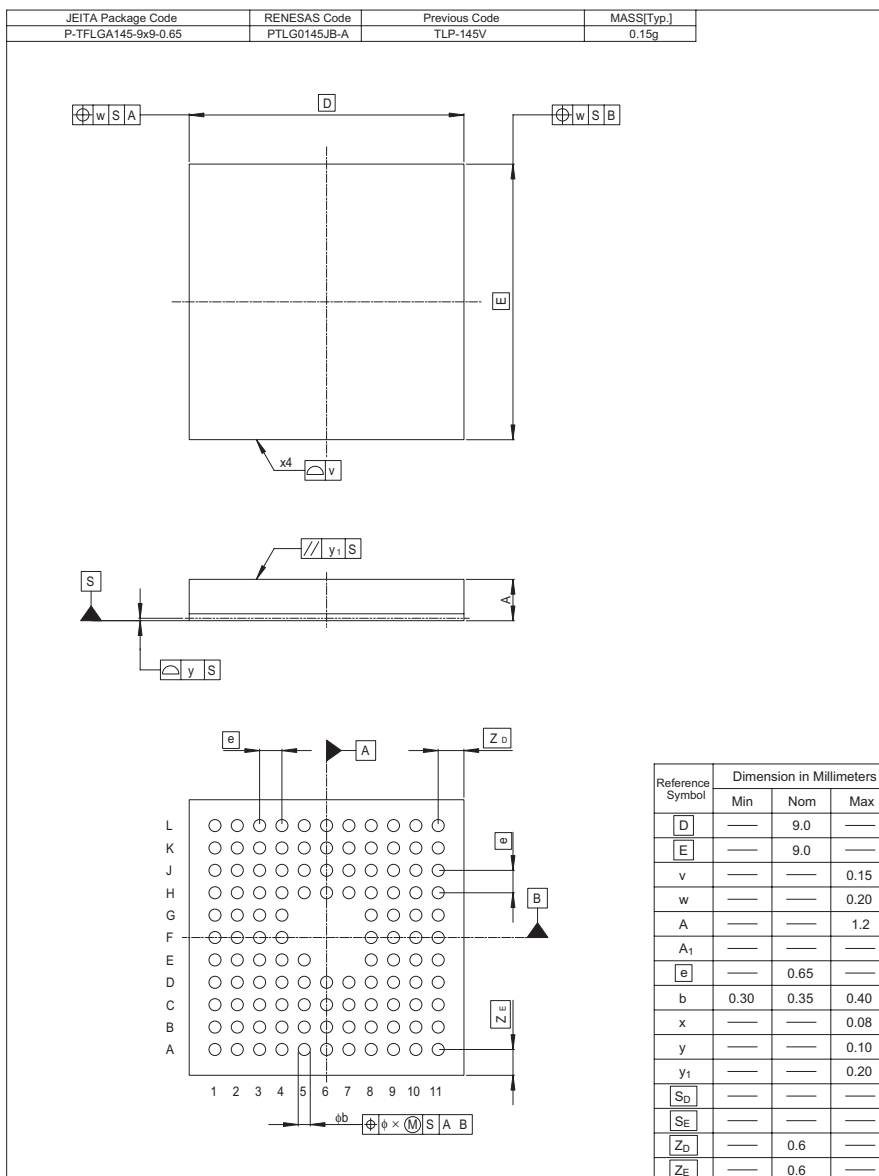


Figure C.1 Package Dimensions (TLP-145V)



## D. Treatment of Unused Pins

The treatments of unused pins are listed in table D.1

**Table D.1 Treatment of Unused Pins**

Pin Name	Mode 4	Mode 5	Mode 6	Modes 1, 2, and 7
RES	(Always used as a reset pin)			
STBY	Connect this pin to VCC via a pull-up resistor			
EMLE	Connect this pin to VSS via a pull-down resistor			
MD2, MD1, MDO	(Always used as mode pins)			
NMI	Connect this pin to VCC via a pull-up resistor			
EXTAL	(Always used as a clock pin)			
XTAL	Leave this pin open			
WDTOVF	Leave this pin open			
Port 1, port 2, port 3, ports 37 to 31, port 6, PA2 to PA0, PF7 to PF5	Connect these pins to VCC via a pull-up resistor or to VSS via a pull-down resistor, respectively			
Port 4	Connect these pins to AVccP via a pull-up resistor or to AVssP via a pull-down resistor, respectively			
Port 5	Connect these pins to AVcc via a pull-up resistor or to AVss via a pull-down resistor, respectively			
PA7	Since this is the B $\phi$ output in its initial state, leave this pin unconnected.		Since this is a general-purpose input port in its initial state, connect each pin to V <sub>CC</sub> via a pull-up resistor or connect each pin to V <sub>SS</sub> via a pull-down resistor.	

Pin Name	Mode 4	Mode 5	Mode 6	Modes 1, 2, and 7
PA6	Since this is the AS output in its initial state, leave this pin unconnected.			Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.
PA5	Since this is the RD output in its initial state, leave this pin unconnected.			Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.
PA4	Since this is the LHWR output in its initial state, leave this pin unconnected.			Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.
PA3	Since this is the LLWR output in its initial state, leave this pin unconnected.			Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.
P30	Since this is the CS0 output in its initial state, leave this pin unconnected		Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.	

Pin Name	Mode 4	Mode 5	Mode 6	Modes 1, 2, and 7
Port D Port E PF4 to PF0	Since this is the address output in its initial state, leave this pin unconnected.		Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.	
Port H	(Used as a data bus)		Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.	
Port I	(Used as a data bus)	Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pulldown resistor.	Since this is a general-purpose input port in its initial state, connect each pin to $V_{CC}$ via a pull-up resistor or connect each pin to $V_{SS}$ via a pull-down resistor.	
Vref	Connect this pin to AVcc			
ANDS0, ANDS1, ANDS2, ANDS3, ANDS4P, ANDS4N, ANDS5P, ANDS5N	After the BIASE bit in DSADMR of the $\Delta\Sigma$ A/D converter is cleared to 0 and the module stop bit of the $\Delta\Sigma$ A/D converter is set to 1, connect each pin to AVccP via a pull-up resistor or connect each pin to AVssP via a pull-down resistor.			
REXT	After the BIASE bit in DSADMR of the $\Delta\Sigma$ A/D converter is cleared to 0 and the module stop bit of the $\Delta\Sigma$ A/D converter is set to 1, this pin is left open.			
AVCM	After the BIASE bit in DSADMR of the $\Delta\Sigma$ A/D converter is cleared to 0 and the module stop bit of the $\Delta\Sigma$ A/D converter is set to 1, this pin is left open.			
AVrefT	Connect to AV <sub>CC</sub> A.			
AVrefB	Connect to AVssA.			
NC	Open			

- Notes:
1. Do not change the function of an unused pin from its initial state.
  2. Do not change the initial value (input-buffer disabled) of PnICR, where n corresponds to an unused pin.

## E. Example of an External Circuit of $\Delta\Sigma$ A/D Converter

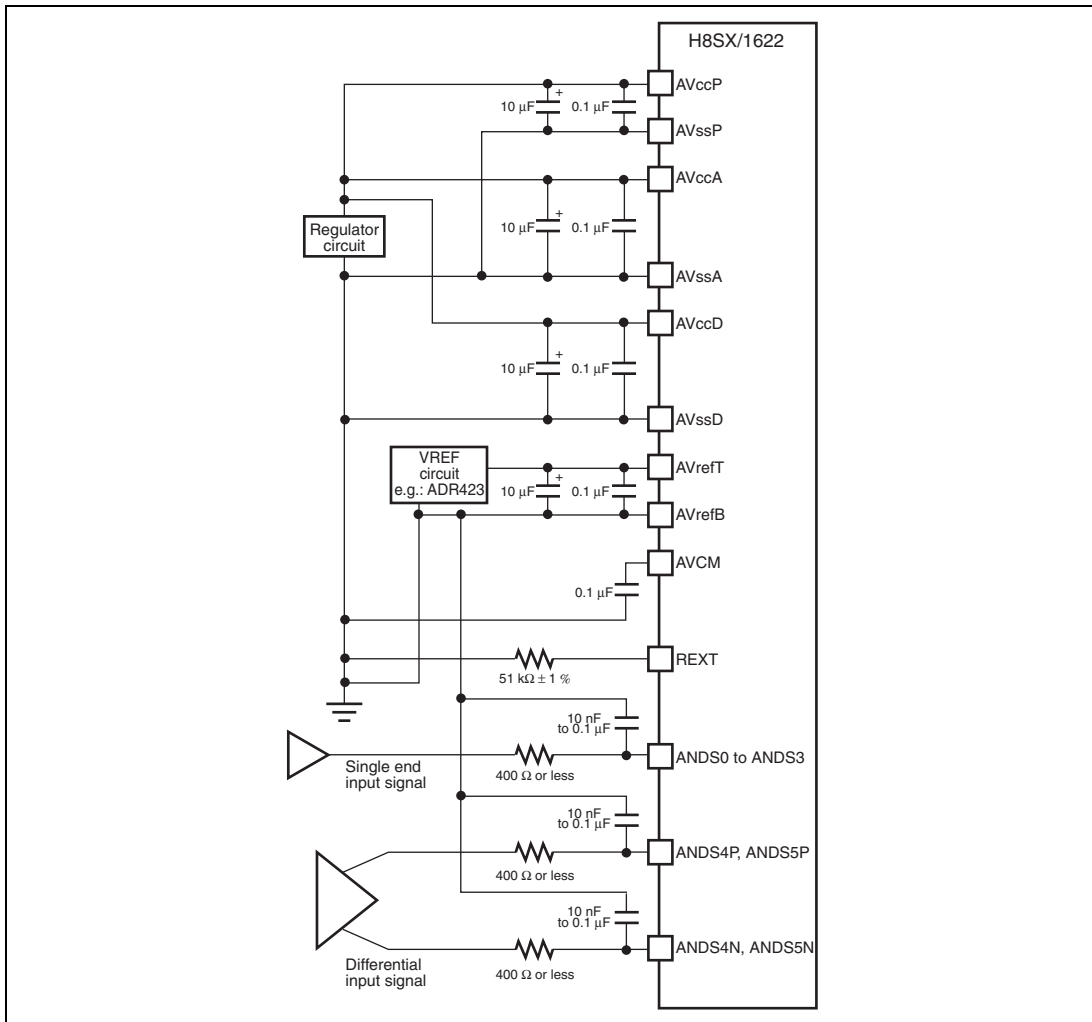


Figure E.1 Example of an External Circuit of  $\Delta\Sigma$  A/D Converter

# Index

## Numerics

$\Delta\Sigma$ A/D converter .....	735
0-output/1-output .....	481
16-bit access space .....	200
16-bit counter mode .....	577
16-bit timer pulse unit (TPU) .....	437
8-bit access space .....	199
8-bit timers (TMR) .....	551

## A

A/D conversion accuracy .....	728
A/D converter .....	713
Absolute accuracy .....	728
Acknowledge .....	695
Address error .....	93
Address map .....	76
Address modes .....	286
Address/data multiplexed I/O interface .....	193, 228
All-module-clock-stop mode .....	876, 897
Area 0 .....	194
Area 1 .....	195
Area 2 .....	195
Area 3 .....	196
Area 4 .....	196
Area 5 .....	197
Area 6 .....	198
Area 7 .....	198
Area division .....	188
Asynchronous mode .....	635
AT-cut parallel-resonance type .....	869
Average transfer rate generator .....	600

## B

B $\phi$ clock output control .....	918
-------------------------------------	-----

Basic bus interface .....	192, 202
Big endian .....	191
Bit rate .....	623
Bit synchronous circuit .....	710
Block structure .....	776
Block transfer mode .....	292, 363
Boot mode .....	773, 800
Buffer operation .....	486
Burst access mode .....	298
Burst ROM interface .....	192, 223
Bus access modes .....	297
Bus arbitration .....	254
Bus configuration .....	180
Bus controller (BSC) .....	155
Bus cycle division .....	357
Bus width .....	191
Bus-released state .....	68
Byte control SRAM interface .....	192, 215

## C

Cascaded connection .....	577
Cascaded operation .....	490
Chain transfer .....	364
Chip select signals .....	189
Clock pulse generator .....	863
Clock synchronization cycle (T <sub>sy</sub> ) .....	182
Clocked synchronous mode .....	652
Communications protocol .....	833
Compare match A .....	575
Compare match B .....	576
Compare match count mode .....	578
Compare match signal .....	575
Counter operation .....	478
CPU priority control function over DTC  and DMAC .....	136
Crystal resonator .....	869

Cycle stealing mode..... 297

## D

D/A converter ..... 763  
Data direction register ..... 383  
Data register..... 384  
Data transfer controller (DTC) ..... 339  
Direct convention ..... 661  
DMA controller (DMAC)..... 259  
Double-buffered structure..... 635  
Download pass/fail result parameter..... 789  
DTC vector address ..... 352  
DTC vector address offset ..... 352  
Dual address mode..... 286

## E

Endian and data alignment ..... 199  
Endian format ..... 191  
Error protection ..... 825  
Error signal ..... 661  
Exception handling ..... 85  
Exception handling vector table ..... 86  
Exception-handling state ..... 68  
Extended repeat area..... 283  
Extended repeat area function ..... 299  
Extension of chip select ( $\overline{CS}$ ) assertion  
period..... 212  
External access bus..... 180  
External bus ..... 185  
External bus clock ( $B\phi$ ) ..... 181, 863  
External bus interface ..... 190  
External clock ..... 870  
External interrupts ..... 120

## F

Flash erase block select parameter ..... 798

Flash memory ..... 771  
Flash multipurpose address area  
parameter ..... 796  
Flash multipurpose data destination  
parameter ..... 797  
Flash pass and fail parameter ..... 790  
Flash program/erase frequency  
parameter ..... 795, 808  
Free-running count operation..... 479  
Frequency divider ..... 863, 871  
Full address mode ..... 350  
Full-scale error..... 728

## H

Hardware protection ..... 824  
Hardware standby mode ..... 876, 913

## I

I/O ports ..... 375  
I<sup>2</sup>C bus format..... 695  
I<sup>2</sup>C bus interface2 (IIC2)..... 679  
ID code..... 646  
Idle cycle..... 238  
Illegal instruction ..... 97, 99  
Input buffer control register ..... 385  
Input Capture Function ..... 482  
Internal interrupts..... 121  
Internal peripheral bus ..... 180  
Internal system bus ..... 180  
Interrupt ..... 95  
Interrupt control mode 0 ..... 127  
Interrupt control mode 2 ..... 129  
Interrupt controller..... 103  
Interrupt exception handling sequence ... 131  
Interrupt exception handling vector  
table ..... 122  
Interrupt response times..... 132



Interrupt sources .....	120
Interrupt sources and vector address offsets .....	122
Interval timer .....	594
Interval timer mode.....	594
Inverse convention.....	662
IRQn interrupts.....	120

## L

Little endian.....	191
--------------------	-----

## M

Mark state.....	635, 673
Master receive mode.....	698
Master transmit mode .....	696
MCU operating modes.....	69
Memory MAT configuration .....	775
Mode 2.....	74
Mode 4.....	74
Mode 5.....	74
Mode 6.....	75
Mode 7.....	75
Mode pin.....	69
Multi-clock mode .....	895
Multiprocessor bit.....	646
Multiprocessor communication function.....	646

## N

NMI interrupt.....	120
Noise canceler.....	704
Nonlinearity error .....	728
Non-overlapping pulse output .....	543
Normal transfer mode.....	360
Normal transfer mode.....	290
Number of Access Cycles.....	192

## O

Offset addition .....	301
Offset error.....	728
On-board programming .....	800
On-chip baud rate generator.....	638
On-chip ROM disabled extended mode....	69
On-chip ROM enabled extended mode....	69
Open-drain control register .....	388
Oscillator.....	869
Output buffer control .....	389
Output trigger.....	543
Overflow .....	577, 592

## P

Package dimensions .....	1013, 1015, 1018
Parity bit.....	635
Periodic count operation .....	479
Peripheral module clock (P $\phi$ ).....	181, 863
Phase counting mode .....	498
Pin assignments.....	10
Pin functions .....	18
PLL circuit .....	863, 871
Port function controller .....	424
Port register.....	384
Power-down modes.....	875
Procedure program.....	818
Processing states .....	68
Product lineup.....	1012
Program execution state .....	68
Program stop state.....	68
Programmable pulse generator (PPG)....	527
Programmer mode.....	773, 831
Programming/erasing interface .....	777
Programming/erasing interface parameters.....	787
Programming/erasing interface register .....	780
Protection .....	824
Pull-up MOS control register.....	386

PWM modes ..... 492

## Q

Quantization error ..... 728

## R

RAM ..... 769

Read strobe ( $\overline{RD}$ ) timing ..... 211

Register addresses ..... 922

Register Bits ..... 935

Register configuration in each port ..... 382

### Registers

ABWCR ..... 159, 927, 943, 959

ADCR ..... 719, 935, 954

ADCSR ..... 717, 922, 935, 954

ADDR ..... 716, 922, 935, 954

ASTCR ..... 160, 927, 943, 959

BCR1 ..... 172, 927, 944, 959

BCR2 ..... 174, 927, 944, 959

BROMCR ..... 177, 927, 944, 959

BRR ..... 623, 932, 950, 964

CCR ..... 37

CPUPCR ..... 107, 931, 948, 963

CRA ..... 346

CRB ..... 346

CSACR ..... 167, 927, 944, 959

DACR ..... 278, 926, 941, 958

DACR01 ..... 765, 932, 949, 964

DADR0 ..... 764, 932, 949, 964

DADR1 ..... 764, 932, 949, 964

DAR ..... 345

DBSR ..... 268, 926, 941, 958

DDAR ..... 265, 926, 940, 958

DDR ..... 383, 923, 938, 956

DMDR ..... 269, 926, 941, 958

DMRSR ..... 284

DOFR ..... 266, 926, 940, 958

DPFR ..... 789

DR ..... 384, 931, 949, 963

DSAR ..... 264, 926, 940, 958

DTCCR ..... 348, 930, 948, 963

DTCER ..... 347, 930, 948, 962

DTCR ..... 267, 926, 940, 958

DTCVBR ..... 349, 927, 943, 959

ENDIANCR ..... 175, 927, 944, 959

EXR ..... 38

FCCS ..... 780, 927, 945, 959

FEBS ..... 798

FECS ..... 783, 927, 945, 959

FKEY ..... 784, 927, 928, 945, 960

FMATS ..... 785

FMPAR ..... 796

FMPDR ..... 797

FPCS ..... 783, 927, 945, 959

FPEFEQ ..... 795, 808

FPFR ..... 790

FTDAR ..... 786, 928, 945, 960

General registers ..... 35

ICCRA ..... 683, 928, 960

ICCRB ..... 684, 928, 960

ICDRR ..... 694, 929, 946, 961

ICDRS ..... 694

ICDRT ..... 694, 929, 946, 961

ICIER ..... 688, 928, 946, 960

ICMR ..... 686, 928, 946, 960

ICR ..... 385, 924, 938, 956

ICSR ..... 690, 928, 946, 960

IDLCR ..... 170, 927, 944, 959

IER ..... 111, 931, 948, 963

INTCR ..... 106, 931, 948, 963

IPR ..... 926, 942, 958

ISCRH ..... 113, 927, 943, 959

ISCR L ..... 113, 927, 943, 959

ISR ..... 118, 931, 948, 963

MAC ..... 39

MDCR ..... 70, 927, 944, 959

MPXCR ..... 179, 927, 944, 959

MRA ..... 342

MRB	343	SEMR	633, 928, 945, 960
MSTPCRA	881, 927, 944, 959	SMR	605, 932, 950, 964
MSTPCRB	881, 927, 944, 959	SRAMCR	176, 927, 944, 959
MSTPCRC	884, 927, 944, 959	SSIER	119, 925, 939, 957
NDER	530	SSR	614, 932, 950, 964
NDERH	932, 949, 964	SYSCR	72, 927, 944, 959
NDERL	932, 949, 964	TCCR	562, 933, 951, 965
NDR	533	TCNT	475
NDRH	932, 950, 964	TCNT (TMR)	559
NDRL	932, 950, 964	TCNT (WDT)	589
ODR	388, 925, 939, 957	TCNT(TMR)	933, 951, 965
PC	36	TCNT(TPU)	933, 951, 965
PCR	386, 536	TCNT(WDT)	932, 950, 964
PCR(I/O ports)	924, 939, 956	TCORA	559, 933, 950, 965
PCR(PPG)	932, 949, 964	TCORB	560, 933, 950, 965
PFCR0	424, 925, 939, 957	TCR	445
PFCR1	425, 925, 939, 957	TCR (TMR)	560
PFCR2	426, 925, 939, 957	TCR(TMR)	933, 950, 965
PFCR4	428, 925, 939, 957	TCR(TPU)	933, 951, 965
PFCR6	429, 925, 939, 957	TCSR (TMR)	567
PFCR7	430, 925, 939, 957	TCSR (WDT)	589
PFCR9	431, 925, 939, 957	TCSR(TMR)	933, 950, 965
PFCRB	433, 925, 939, 957	TCSR(WDT)	932, 950, 964
PFCRC	434, 925, 939, 957	TDR	604, 932, 950, 964
PMR	537, 932, 949, 964	TGR	475, 933, 951, 965
PODR	532	TIER	469, 933, 951, 965
PODRH	932, 949, 964	TIOR	451, 933, 951, 965
PODRL	932, 949, 964	TMDR	450, 933, 951, 965
PORT	384, 931, 948, 963	TSR	471
RAMER	799, 927, 944, 959	TSR(TPU)	933, 951, 965
RDNCR	166, 927, 943, 959	TSTR	476, 933, 951, 965
RDR	604, 932, 950, 964	TSYR	477, 933, 951, 965
RSR	604	VBR	39
RSTCSR	591, 933, 950, 964	WTCRA	161, 927, 943, 959
SAR	345, 693, 928, 946, 961	WTCRB	161, 927, 943, 959
SBR	39	Repeat transfer mode	291, 361
SBYCR	878, 927, 944, 959	Reset	88
SCKCR	865, 927, 944, 959	Reset state	68
SCMR	622, 932, 950, 964	Resolution	728
SCR	608, 932, 950, 964		

<b>S</b>	
Sample-and-hold circuit .....	725
Scan mode .....	723
Serial communication interface (SCI) ....	599
Short address mode.....	350
Single address mode .....	287
Single mode.....	721
Slave receive mode.....	703
Slave transmit mode .....	700
Sleep instruction .....	98
Sleep mode .....	876, 896
Smart card interface.....	660
Software protection.....	825
Software standby mode .....	876, 898
Space state .....	635
Stack status after exception handling.....	100
Standard serial communication interface specifications for boot mode.....	831
Start bit .....	635
State transitions .....	68
Stop bit .....	635
Strobe assert/negate timing.....	193
Synchronous clearing .....	484
Synchronous operation .....	484
Synchronous presetting.....	484
System clock (I $\phi$ ).....	181, 863
<b>T</b>	
Toggle output.....	481
Trace exception handling.....	92
Transfer information.....	350
Transfer information read skip function .....	359
Transfer information writeback skip function.....	360
Transfer modes .....	290
Transmit/receive data.....	635
Trap instruction exception handling .....	97
<b>U</b>	
User boot MAT .....	775
User boot mode.....	773, 814
User break controller (UBC).....	143
User MAT.....	775
User program mode .....	773, 804
<b>V</b>	
Vector table address.....	86
Vector table address offset.....	86
<b>W</b>	
Wait control .....	209
Watchdog timer (WDT).....	587
Watchdog timer mode.....	592
Waveform output by compare match.....	480
Write data buffer function.....	252
Write data buffer function for external data bus .....	252
Write data buffer function for peripheral modules.....	253

---

**Renesas 32-Bit CISC Microcomputer  
Hardware Manual  
H8SX/1622 Group**

Publication Date: Rev.1.00, Nov. 01, 2007  
Published by: Sales Strategic Planning Div.  
Renesas Technology Corp.  
Edited by: Customer Support Department  
Global Strategic Communication Div.  
Renesas Solutions Corp.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

---



## RENESAS SALES OFFICES

<http://www.renesas.com>

Refer to "<http://www.renesas.com/en/network>" for the latest and detailed information.

### **Renesas Technology America, Inc.**

450 Holger Way, San Jose, CA 95134-1368, U.S.A  
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

### **Renesas Technology Europe Limited**

Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.  
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

### **Renesas Technology (Shanghai) Co., Ltd.**

Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120  
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

### **Renesas Technology Hong Kong Ltd.**

7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong  
Tel: <852> 2265-6688, Fax: <852> 2730-6071

### **Renesas Technology Taiwan Co., Ltd.**

10th Floor, No.99, Fushing North Road, Taipei, Taiwan  
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

### **Renesas Technology Singapore Pte. Ltd.**

1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632  
Tel: <65> 6213-0200, Fax: <65> 6278-8001

### **Renesas Technology Korea Co., Ltd.**

Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea  
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

### **Renesas Technology Malaysia Sdn. Bhd**

Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia  
Tel: <603> 7955-9390, Fax: <603> 7955-9510





# H8SX/1622 Group Hardware Manual



Renesas Technology Corp.  
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan