

To our customers,

---

## Old Company Name in Catalogs and Other Documents

---

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

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

April 1<sup>st</sup>, 2010  
Renesas Electronics Corporation

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

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

## Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

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

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

"Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.

8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

# H8SX/1665 Group, H8SX/1665M Group

## Hardware Manual

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

H8SX/1665	R5F61665
H8SX/1662	R5F61662
H8SX/1665M	R5F61665M
H8SX/1662M	R5F61662M

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



document, please confirm the latest product information with a Renesas sales office. Also, please pay attention and careful attention to additional and different information to be disclosed by Renesas such as that 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, 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 in this document.
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 purchases made to 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 special 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 damage, injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design of hardware and software including but not limited to redundancy, fire control and malfunction prevention measures, 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 product 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 high. You should implement safety measures so that Renesas products may not be easily detached from the 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.

vicinity of LSI, an associated shoot-through current flows internally, and malfunction 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-management 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 type number, ensure that the change will not lead to problems.

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

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 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 revisions in the manual.

The following documents have been prepared for the H8SX/1665 Group and the H8SX/1665M 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.

<b>Document Type</b>	<b>Contents</b>	<b>Document Title</b>	<b>Document ID</b>
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/1665 Group, H8SX/1665M Group Hardware Manual	This manual
Software Manual	Detailed descriptions of the CPU and instruction set	H8SX Family Software Manual	REJ01B0010
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.		

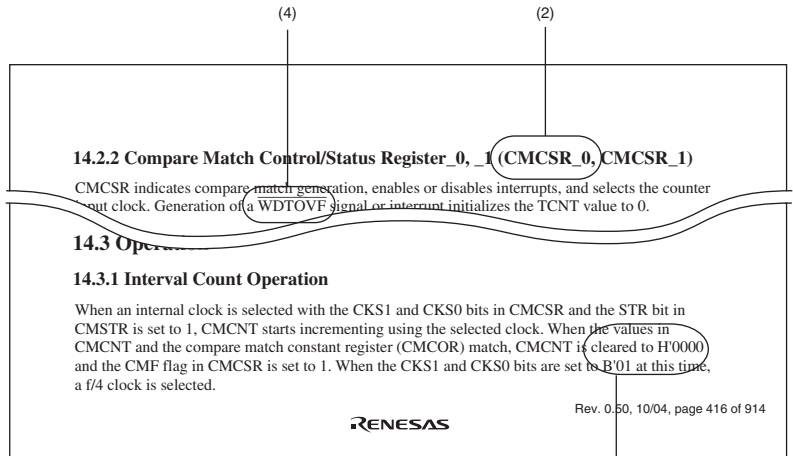
Binary numbers are given as B'nnnn (B' may be omitted if the number is obviously binary), hexadecimal numbers are given as H'nnnn or 0xnnnn, and decimal numbers are given as nnnn.

[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]  $\overline{\text{WDT0VF}}$



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



Bit	Bit Name	Initial Value	R/W	Description
15	-	0	R	Reserved
14	-	0	R	Reserved These bits are always read as 0.
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.
-	-	0	-	-

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

SCI	Serial communications interface
TMR	8-bit timer
TPU	16-bit timer pulse unit
WDT	Watchdog timer

- Abbreviations other than those listed above

<b>Abbreviation</b>	<b>Description</b>
ACIA	Asynchronous communications interface adapter
bps	Bits per second
CRC	Cyclic redundancy check
DMA	Direct memory access
DMAC	Direct memory access controller
GSM	Global System for Mobile Communications
Hi-Z	High impedance
IEBus	Inter Equipment Bus (IEBus is a trademark of NEC Electronics Corporation)
I/O	Input/output
IrDA	Infrared Data Association
LSB	Least significant bit
MSB	Most significant bit
NC	No connection
PLL	Phase-locked loop
PWM	Pulse width modulation
SFR	Special function register
SIM	Subscriber Identity Module
UART	Universal asynchronous receiver/transmitter
VCO	Voltage-controlled oscillator

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

1.4.1	Pin Assignments .....
1.4.2	Correspondence between Pin Configuration and Operating Modes .....
1.4.3	Pin Functions .....
<b>Section 2 CPU .....</b>	
2.1	Features .....
2.2	CPU Operating Modes .....
2.2.1	Normal Mode .....
2.2.2	Middle Mode .....
2.2.3	Advanced Mode .....
2.2.4	Maximum Mode .....
2.3	Instruction Fetch .....
2.4	Address Space .....
2.5	Registers .....
2.5.1	General Registers .....
2.5.2	Program Counter (PC) .....
2.5.3	Condition-Code Register (CCR) .....
2.5.4	Extended Control Register (EXR) .....
2.5.5	Vector Base Register (VBR) .....
2.5.6	Short Address Base Register (SBR) .....
2.5.7	Multiply-Accumulate Register (MAC) .....
2.5.8	Initial Values of CPU Registers .....
2.6	Data Formats .....
2.6.1	General Register Data Formats .....
2.6.2	Memory Data Formats .....
2.7	Instruction Set .....
2.7.1	Instructions and Addressing Modes .....
2.7.2	Table of Instructions Classified by Function .....
2.7.3	Basic Instruction Formats .....

2.8.7	Immediate—#xx .....
2.8.8	Program-Counter Relative—@(d:8, PC) or @(d:16, PC) .....
2.8.9	Program-Counter Relative with Index Register—@(RnL.B, PC), @(Rn.W, PC), or @(ERn.L, PC) .....
2.8.10	Memory Indirect—@@aa:8 .....
2.8.11	Extended Memory Indirect—@@vec:7 .....
2.8.12	Effective Address Calculation .....
2.8.13	MOVA Instruction .....
2.9	Processing States .....
Section 3 MCU Operating Modes .....	
3.1	Operating Mode Selection .....
3.2	Register Descriptions .....
3.2.1	Mode Control Register (MDCR) .....
3.2.2	System Control Register (SYSCR) .....
3.3	Operating Mode Descriptions .....
3.3.1	Mode 1 .....
3.3.2	Mode 2 .....
3.3.3	Mode 3 .....
3.3.4	Mode 4 .....
3.3.5	Mode 5 .....
3.3.6	Mode 6 .....
3.3.7	Mode 7 .....
3.3.8	Pin Functions .....
3.4	Address Map .....
3.4.1	Address Map .....

4.8	Watchdog Timer Reset .....
4.9	Determination of Reset Generation Source.....
Section 5 Voltage Detection Circuit (LVD) .....	
5.1	Features.....
5.2	Register Descriptions.....
5.2.1	Voltage Detection Control Register (LVDCR).....
5.2.2	Reset Status Register (RSTSR).....
5.3	Voltage Detection Circuit .....
5.3.1	Voltage Monitoring Reset.....
5.3.2	Voltage Monitoring Interrupt.....
5.3.3	Release from Deep Software Standby Mode by the Voltage-Detection Circuit .....
5.3.4	Voltage Monitor.....
Section 6 Exception Handling .....	
6.1	Exception Handling Types and Priority .....
6.2	Exception Sources and Exception Handling Vector Table .....
6.3	Reset .....
6.3.1	Reset Exception Handling.....
6.3.2	Interrupts after Reset.....
6.3.3	On-Chip Peripheral Functions after Reset Release .....
6.4	Traces.....
6.5	Address Error.....
6.5.1	Address Error Source.....
6.5.2	Address Error Exception Handling .....
6.6	Interrupts.....
6.6.1	Interrupt Sources.....
6.6.2	Interrupt Exception Handling .....

7.3.1	Interrupt Control Register (INTCR) .....
7.3.2	CPU Priority Control Register (CPUPCR) .....
7.3.3	Interrupt Priority Registers A to O, Q, and R (IPRA to IPRO, IPRQ, and IPRR) .....
7.3.4	IRQ Enable Register (IER) .....
7.3.5	IRQ Sense Control Registers H and L (ISCRH, ISCL).....
7.3.6	IRQ Status Register (ISR).....
7.3.7	Software Standby Release IRQ Enable Register (SSIER) .....
7.4	Interrupt Sources .....
7.4.1	External Interrupts .....
7.4.2	Internal Interrupts .....
7.5	Interrupt Exception Handling Vector Table.....
7.6	Interrupt Control Modes and Interrupt Operation .....
7.6.1	Interrupt Control Mode 0 .....
7.6.2	Interrupt Control Mode 2 .....
7.6.3	Interrupt Exception Handling Sequence .....
7.6.4	Interrupt Response Times .....
7.6.5	DTC and DMAC Activation by Interrupt.....
7.7	CPU Priority Control Function Over DTC, DMAC and EXDMAC .....
7.8	Usage Notes .....
7.8.1	Conflict between Interrupt Generation and Disabling .....
7.8.2	Instructions that Disable Interrupts .....
7.8.3	Times when Interrupts are Disabled .....
7.8.4	Interrupts during Execution of EEPMOV Instruction .....
7.8.5	Interrupts during Execution of MOVMD and MOVSD Instructions.....
7.8.6	Interrupts of Peripheral Modules .....

8.4.3	Condition Match Flag .....
8.5	Usage Notes .....
<b>Section 9 Bus Controller (BSC).....</b>	
9.1	Features.....
9.2	Register Descriptions.....
9.2.1	Bus Width Control Register (ABWCR).....
9.2.2	Access State Control Register (ASTCR) .....
9.2.3	Wait Control Registers A and B (WTCRA, WTCRB) .....
9.2.4	Read Strobe Timing Control Register (RDNCR) .....
9.2.5	$\overline{CS}$ Assertion Period Control Registers (CSACR) .....
9.2.6	Idle Control Register (IDLCR) .....
9.2.7	Bus Control Register 1 (BCR1) .....
9.2.8	Bus Control Register 2 (BCR2) .....
9.2.9	Endian Control Register (ENDIANCR).....
9.2.10	SRAM Mode Control Register (SRAMCR) .....
9.2.11	Burst ROM Interface Control Register (BROMCR).....
9.2.12	Address/Data Multiplexed I/O Control Register (MPXCR) .....
9.2.13	DRAM Control Register (DRAMCR) .....
9.2.14	DRAM Access Control Register (DRACCR).....
9.2.15	Synchronous DRAM Control Register (SDCR) .....
9.2.16	Refresh Control Register (REFCR) .....
9.2.17	Refresh Timer Counter (RTCNT).....
9.2.18	Refresh Time Constant Register (RTCOR) .....
9.3	Bus Configuration.....
9.4	Multi-Clock Function and Number of Access Cycles .....
9.5	External Bus.....
9.5.1	Input/Output Pins .....
9.5.2	Area Division.....

	9.6.7	$\overline{\text{DACK}}$ and $\overline{\text{EDACK}}$ Signal Output Timing.....
9.7		Byte Control SRAM Interface .....
	9.7.1	Byte Control SRAM Space Setting.....
	9.7.2	Data Bus .....
	9.7.3	I/O Pins Used for Byte Control SRAM Interface .....
	9.7.4	Basic Timing.....
	9.7.5	Wait Control .....
	9.7.6	Read Strobe ( $\overline{\text{RD}}$ ) .....
	9.7.7	Extension of Chip Select ( $\overline{\text{CS}}$ ) Assertion Period.....
	9.7.8	$\overline{\text{DACK}}$ and $\overline{\text{EDACK}}$ Signal Output Timing.....
9.8		Burst ROM Interface .....
	9.8.1	Burst ROM Space Setting.....
	9.8.2	Data Bus .....
	9.8.3	I/O Pins Used for Burst ROM Interface.....
	9.8.4	Basic Timing.....
	9.8.5	Wait Control .....
	9.8.6	Read Strobe ( $\overline{\text{RD}}$ ) Timing.....
	9.8.7	Extension of Chip Select ( $\overline{\text{CS}}$ ) Assertion Period.....
9.9		Address/Data Multiplexed I/O Interface.....
	9.9.1	Address/Data Multiplexed I/O Space Setting .....
	9.9.2	Address/Data Multiplex .....
	9.9.3	Data Bus .....
	9.9.4	I/O Pins Used for Address/Data Multiplexed I/O Interface.....
	9.9.5	Basic Timing.....
	9.9.6	Address Cycle Control.....
	9.9.7	Wait Control .....
	9.9.8	Read Strobe ( $\overline{\text{RD}}$ ) Timing.....
	9.9.9	Extension of Chip Select ( $\overline{\text{CS}}$ ) Assertion Period.....
	9.9.10	$\overline{\text{DACK}}$ and $\overline{\text{EDACK}}$ Signal Output Timing.....



	9.10.11	Burst Access Operation.....
	9.10.12	Refresh Control.....
	9.10.13	DRAM Interface and Single Address Transfer by DMAC and EXDMA.....
9.11		Synchronous DRAM Interface .....
	9.11.1	Setting SDRAM space .....
	9.11.2	Address Multiplexing.....
	9.11.3	Data Bus.....
	9.11.4	I/O Pins Used for DRAM Interface .....
	9.11.5	Basic Timing.....
	9.11.6	CAS Latency Control.....
	9.11.7	Controlling Row Address Output Cycle .....
	9.11.8	Controlling Precharge Cycle.....
	9.11.9	Controlling Clock Suspend Insertion.....
	9.11.10	Controlling Write-Precharge Delay .....
	9.11.11	Controlling Byte and Word Accesses .....
	9.11.12	Fast-Page Access Operation .....
	9.11.13	Refresh Control.....
	9.11.14	Setting SDRAM Mode Register .....
	9.11.15	SDRAM Interface and Single Address Transfer by DMAC and EXDMA.....
	9.11.16	EXDMAC Cluster Transfer .....
9.12		Idle Cycle.....
	9.12.1	Operation .....
	9.12.2	Pin States in Idle Cycle.....
9.13		Bus Release.....
	9.13.1	Operation .....
	9.13.2	Pin States in External Bus Released State.....
	9.13.3	Transition Timing .....
9.14		Internal Bus.....
	9.14.1	Access to Internal Address Space .....

10.2	Input/Output Pins .....
10.3	Register Descriptions .....
10.3.1	DMA Source Address Register (DSAR) .....
10.3.2	DMA Destination Address Register (DDAR) .....
10.3.3	DMA Offset Register (DOFR).....
10.3.4	DMA Transfer Count Register (DTCR) .....
10.3.5	DMA Block Size Register (DBSR) .....
10.3.6	DMA Mode Control Register (DMDR).....
10.3.7	DMA Address Control Register (DACR).....
10.3.8	DMA Module Request Select Register (DMRSR) .....
10.4	Transfer Modes .....
10.5	Operations.....
10.5.1	Address Modes .....
10.5.2	Transfer Modes .....
10.5.3	Activation Sources.....
10.5.4	Bus Access Modes .....
10.5.5	Extended Repeat Area Function .....
10.5.6	Address Update Function using Offset .....
10.5.7	Register during DMA Transfer .....
10.5.8	Priority of Channels .....
10.5.9	DMA Basic Bus Cycle.....
10.5.10	Bus Cycles in Dual Address Mode .....
10.5.11	Bus Cycles in Single Address Mode.....
10.6	DMA Transfer End .....
10.7	Relationship among DMAC and Other Bus Masters .....
10.7.1	CPU Priority Control Function Over DMAC .....
10.7.2	Bus Arbitration among DMAC and Other Bus Masters .....
10.8	Interrupt Sources.....
10.9	Usage Notes .....

11.3.8	Cluster Buffer Registers 0 to 7 (CLSBR0 to CLSBR7).....
11.4	Transfer Modes .....
11.4.1	Ordinary Modes .....
11.4.2	Cluster Transfer Modes.....
11.5	Mode Operation .....
11.5.1	Address Modes .....
11.5.2	Transfer Modes .....
11.5.3	Activation Sources .....
11.5.4	Bus Mode.....
11.5.5	Extended Repeat Area Function .....
11.5.6	Address Update Function Using Offset .....
11.5.7	Registers during EXDMA Transfer Operation .....
11.5.8	Channel Priority Order.....
11.5.9	Basic Bus Cycles .....
11.5.10	Bus Cycles in Dual Address Mode .....
11.5.11	Bus Cycles in Single Address Mode.....
11.5.12	Operation Timing in Each Mode .....
11.6	Operation in Cluster Transfer Mode .....
11.6.1	Address Mode .....
11.6.2	Setting of Address Update Mode .....
11.6.3	Caution for Combining with Extended Repeat Area Function .....
11.6.4	Bus Cycles in Cluster Transfer Dual Address Mode .....
11.6.5	Operation Timing in Cluster Transfer Mode .....
11.7	Ending EXDMA Transfer.....
11.8	Relationship among EXDMAC and Other Bus Masters.....
11.8.1	CPU Priority Control Function Over EXDMAC .....
11.8.2	Bus Arbitration with Another Bus Master .....
11.9	Interrupt Sources.....
11.10	Usage Notes.....

12.2.9	DTC Vector Base Register (DTCVBR).....
12.3	Activation Sources.....
12.4	Location of Transfer Information and DTC Vector Table.....
12.5	Operation .....
12.5.1	Bus Cycle Division .....
12.5.2	Transfer Information Read Skip Function .....
12.5.3	Transfer Information Writeback Skip Function.....
12.5.4	Normal Transfer Mode .....
12.5.5	Repeat Transfer Mode .....
12.5.6	Block Transfer Mode.....
12.5.7	Chain Transfer .....
12.5.8	Operation Timing.....
12.5.9	Number of DTC Execution Cycles .....
12.5.10	DTC Bus Release Timing .....
12.5.11	DTC Priority Level Control to the CPU .....
12.6	DTC Activation by Interrupt.....
12.7	Examples of Use of the DTC .....
12.7.1	Normal Transfer Mode .....
12.7.2	Chain Transfer .....
12.7.3	Chain Transfer when Counter = 0.....
12.8	Interrupt Sources.....
12.9	Usage Notes .....
12.9.1	Module Stop State Setting .....
12.9.2	On-Chip RAM .....
12.9.3	DMAC Transfer End Interrupt.....
12.9.4	DTCE Bit Setting.....
12.9.5	Chain Transfer .....
12.9.6	Transfer Information Start Address, Source Address, and Destination Address .....

13.1.5	Pull-Up MOS Control Register (PnPCR) (n = D to F, and H to K).....
13.1.6	Open-Drain Control Register (PnODR) (n = 2 and F).....
13.2	Output Buffer Control.....
13.2.1	Port 1.....
13.2.2	Port 2.....
13.2.3	Port 3.....
13.2.4	Port 5.....
13.2.5	Port 6.....
13.2.6	Port A.....
13.2.7	Port B.....
13.2.8	Port C.....
13.2.9	Port D.....
13.2.10	Port E.....
13.2.11	Port F.....
13.2.12	Port H.....
13.2.13	Port I.....
13.2.14	Port J.....
13.2.15	Port K.....
13.2.16	Port M.....
13.3	Port Function Controller.....
13.3.1	Port Function Control Register 0 (PFCR0).....
13.3.2	Port Function Control Register 1 (PFCR1).....
13.3.3	Port Function Control Register 2 (PFCR2).....
13.3.4	Port Function Control Register 4 (PFCR4).....
13.3.5	Port Function Control Register 6 (PFCR6).....
13.3.6	Port Function Control Register 7 (PFCR7).....
13.3.7	Port Function Control Register 8 (PFCR8).....
13.3.8	Port Function Control Register 9 (PFCR9).....
13.3.9	Port Function Control Register A (PFCRA).....

14.3.1	Timer Control Register (TCR).....
14.3.2	Timer Mode Register (TMDR).....
14.3.3	Timer I/O Control Register (TIOR).....
14.3.4	Timer Interrupt Enable Register (TIER).....
14.3.5	Timer Status Register (TSR).....
14.3.6	Timer Counter (TCNT).....
14.3.7	Timer General Register (TGR).....
14.3.8	Timer Start Register (TSTR).....
14.3.9	Timer Synchronous Register (TSYR).....
14.4	Operation.....
14.4.1	Basic Functions.....
14.4.2	Synchronous Operation.....
14.4.3	Buffer Operation.....
14.4.4	Cascaded Operation.....
14.4.5	PWM Modes.....
14.4.6	Phase Counting Mode.....
14.5	Interrupt Sources.....
14.6	DTC Activation.....
14.7	DMAC Activation.....
14.8	A/D Converter Activation.....
14.9	Operation Timing.....
14.9.1	Input/Output Timing.....
14.9.2	Interrupt Signal Timing.....
14.10	Usage Notes.....
14.10.1	Module Stop Function Setting.....
14.10.2	Input Clock Restrictions.....
14.10.3	Caution on Cycle Setting.....
14.10.4	Conflict between TCNT Write and Clear Operations.....
14.10.5	Conflict between TCNT Write and Increment Operations.....

Section 15	Programmable Pulse Generator (PPG)	.....
15.1	Features	.....
15.2	Input/Output Pins	.....
15.3	Register Descriptions	.....
15.3.1	Next Data Enable Registers H, L (NDERH, NDERL)	.....
15.3.2	Output Data Registers H, L (PODRH, PODRL)	.....
15.3.3	Next Data Registers H, L (NDRH, NDRL)	.....
15.3.4	PPG Output Control Register (PCR)	.....
15.3.5	PPG Output Mode Register (PMR)	.....
15.4	Operation	.....
15.4.1	Output Timing	.....
15.4.2	Sample Setup Procedure for Normal Pulse Output	.....
15.4.3	Example of Normal Pulse Output (Example of 5-Phase Pulse Output)	.....
15.4.4	Non-Overlapping Pulse Output	.....
15.4.5	Sample Setup Procedure for Non-Overlapping Pulse Output	.....
15.4.6	Example of Non-Overlapping Pulse Output (Example of 4-Phase Complementary Non-Overlapping Pulse Output)	.....
15.4.7	Inverted Pulse Output	.....
15.4.8	Pulse Output Triggered by Input Capture	.....
15.5	Usage Notes	.....
15.5.1	Module Stop State Function	.....
15.5.2	Operation of Pulse Output Pins	.....
15.5.3	TPU Setting when PPG1 is in Use	.....
Section 16	8-Bit Timers (TMR)	.....
16.1	Features	.....
16.2	Input/Output Pins	.....
16.3	Register Descriptions	.....
16.3.1	Timer Counter (TCNT)	.....

	16.5.3	Timing of Timer Output at Compare Match .....
	16.5.4	Timing of Counter Clear by Compare Match .....
	16.5.5	Timing of TCNT External Reset.....
	16.5.6	Timing of Overflow Flag (OVF) Setting .....
16.6		Operation with Cascaded Connection.....
	16.6.1	16-Bit Counter Mode .....
	16.6.2	Compare Match Count Mode.....
16.7		Interrupt Sources.....
	16.7.1	Interrupt Sources and DTC Activation .....
	16.7.2	A/D Converter Activation.....
16.8		Usage Notes .....
	16.8.1	Notes on Setting Cycle .....
	16.8.2	Conflict between TCNT Write and Counter Clear .....
	16.8.3	Conflict between TCNT Write and Increment.....
	16.8.4	Conflict between TCOR Write and Compare Match .....
	16.8.5	Conflict between Compare Matches A and B.....
	16.8.6	Switching of Internal Clocks and TCNT Operation .....
	16.8.7	Mode Setting with Cascaded Connection .....
	16.8.8	Module Stop State Setting .....
	16.8.9	Interrupts in Module Stop State .....
	Section 17	32K Timer (TM32K).....
	17.1	Features.....
	17.2	Register Descriptions.....
		17.2.1 Timer Control Register (TCR32K).....
		17.2.2 Timer Counter (TCNT32K1, TCNT32K2, TCNT32K3).....
	17.3	Operation .....
		17.3.1 Basic Operation .....
		17.3.2 EXCKSN=1 Operation .....



18.3.1	Timer Counter (TCNT).....
18.3.2	Timer Control/Status Register (TCSR).....
18.3.3	Reset Control/Status Register (RSTCSR).....
18.4	Operation .....
18.4.1	Watchdog Timer Mode .....
18.4.2	Interval Timer Mode .....
18.5	Interrupt Source .....
18.6	Usage Notes .....
18.6.1	Notes on Register Access.....
18.6.2	Conflict between Timer Counter (TCNT) Write and Increment.....
18.6.3	Changing Values of Bits CKS2 to CKS0.....
18.6.4	Switching between Watchdog Timer Mode and Interval Timer Mode.....
18.6.5	Internal Reset in Watchdog Timer Mode.....
18.6.6	System Reset by $\overline{\text{WDTOVF}}$ Signal.....
18.6.7	Transition to Watchdog Timer Mode or Software Standby Mode.....

## Section 19 Serial Communication Interface (SCI, IrDA, CRC).....

19.1	Features.....
19.2	Input/Output Pins.....
19.3	Register Descriptions.....
19.3.1	Receive Shift Register (RSR) .....
19.3.2	Receive Data Register (RDR).....
19.3.3	Transmit Data Register (TDR).....
19.3.4	Transmit Shift Register (TSR) .....
19.3.5	Serial Mode Register (SMR) .....
19.3.6	Serial Control Register (SCR).....
19.3.7	Serial Status Register (SSR) .....
19.3.8	Smart Card Mode Register (SCMR).....
19.3.9	Bit Rate Register (BRR) .....

19.5	Multiprocessor Communication Function .....
19.5.1	Multiprocessor Serial Data Transmission .....
19.5.2	Multiprocessor Serial Data Reception .....
19.6	Operation in Clocked Synchronous Mode (SCI_0, 1, 2, and 4 only).....
19.6.1	Clock.....
19.6.2	SCI Initialization (Clocked Synchronous Mode) (SCI_0, 1, 2, and 4 only).....
19.6.3	Serial Data Transmission (Clocked Synchronous Mode) (SCI_0, 1, 2, and 4 only).....
19.6.4	Serial Data Reception (Clocked Synchronous Mode) (SCI_0, 1, 2, and 4 only).....
19.6.5	Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode) (SCI_0, 1, 2, and 4 only) .....
19.7	Operation in Smart Card Interface Mode.....
19.7.1	Sample Connection .....
19.7.2	Data Format (Except in Block Transfer Mode) .....
19.7.3	Block Transfer Mode .....
19.7.4	Receive Data Sampling Timing and Reception Margin .....
19.7.5	Initialization.....
19.7.6	Data Transmission (Except in Block Transfer Mode) .....
19.7.7	Serial Data Reception (Except in Block Transfer Mode) .....
19.7.8	Clock Output Control (Only SCI_0, 1, 2, and 4) .....
19.8	IrDA Operation.....
19.9	Interrupt Sources.....
19.9.1	Interrupts in Normal Serial Communication Interface Mode .....
19.9.2	Interrupts in Smart Card Interface Mode .....
19.10	Usage Notes .....
19.10.1	Module Stop Function Setting .....
19.10.2	Break Detection and Processing .....
19.10.3	Mark State and Break Detection.....

## Section 20 USB Function Module (USB)

20.1	Features	.....
20.2	Input/Output Pins	.....
20.3	Register Descriptions	.....
20.3.1	Interrupt Flag Register 0 (IFR0)	.....
20.3.2	Interrupt Flag Register 1 (IFR1)	.....
20.3.3	Interrupt Flag Register 2 (IFR2)	.....
20.3.4	Interrupt Select Register 0 (ISR0)	.....
20.3.5	Interrupt Select Register 1 (ISR1)	.....
20.3.6	Interrupt Select Register 2 (ISR2)	.....
20.3.7	Interrupt Enable Register 0 (IER0)	.....
20.3.8	Interrupt Enable Register 1 (IER1)	.....
20.3.9	Interrupt Enable Register 2 (IER2)	.....
20.3.10	EP0i Data Register (EPDR0i)	.....
20.3.11	EP0o Data Register (EPDR0o)	.....
20.3.12	EP0s Data Register (EPDR0s)	.....
20.3.13	EP1 Data Register (EPDR1)	.....
20.3.14	EP2 Data Register (EPDR2)	.....
20.3.15	EP3 Data Register (EPDR3)	.....
20.3.16	EP0o Receive Data Size Register (EPSZ0o)	.....
20.3.17	EP1 Receive Data Size Register (EPSZ1)	.....
20.3.18	Trigger Register (TRG)	.....
20.3.19	Data Status Register (DASTS)	.....
20.3.20	FIFO Clear Register (FCLR)	.....
20.3.21	DMA Transfer Setting Register (DMA)	.....
20.3.22	Endpoint Stall Register (EPSTL)	.....
20.3.23	Configuration Value Register (CVR)	.....
20.3.24	Control Register (CTLR)	.....
20.3.25	Endpoint Information Register (EPIR)	.....

20.6	Processing of USB Standard Commands and Class/Vendor Commands .....
20.6.1	Processing of Commands Transmitted by Control Transfer .....
20.7	Stall Operations .....
20.7.1	Overview .....
20.7.2	Forcible Stall by Application .....
20.7.3	Automatic Stall by USB Function Module .....
20.8	DMA Transfer .....
20.8.1	Overview .....
20.8.2	DMA Transfer for Endpoint 1 .....
20.8.3	DMA Transfer for Endpoint 2 .....
20.9	Example of USB External Circuitry .....
20.10	Usage Notes .....
20.10.1	Receiving Setup Data.....
20.10.2	Clearing the FIFO .....
20.10.3	Overreading and Overwriting the Data Registers .....
20.10.4	Assigning Interrupt Sources to EP0 .....
20.10.5	Clearing the FIFO When DMA Transfer is Enabled .....
20.10.6	Notes on TR Interrupt .....
20.10.7	Restrictions on Peripheral Module Clock (Pφ) Operating Frequency.....
20.10.8	Notes on Deep Software Standby Mode when USB is Used .....
Section 21 I <sup>2</sup> C Bus Interface 2 (IIC2).....	
21.1	Features.....
21.2	Input/Output Pins.....
21.3	Register Descriptions.....
21.3.1	I <sup>2</sup> C Bus Control Register A (ICCRA) .....
21.3.2	I <sup>2</sup> C Bus Control Register B (ICCRB) .....
21.3.3	I <sup>2</sup> C Bus Mode Register (ICMR).....
21.3.4	I <sup>2</sup> C Bus Interrupt Enable Register (ICIER).....

21.4.6	Noise Canceler.....
21.4.7	Example of Use.....
21.5	Interrupt Request.....
21.6	Bit Synchronous Circuit.....
21.7	Usage Notes.....
<b>Section 22 A/D Converter.....</b>	
22.1	Features.....
22.2	Input/Output Pins.....
22.3	Register Descriptions.....
22.3.1	A/D Data Registers A to H (ADDRA to ADDRH) .....
22.3.2	A/D Control/Status Register_0 (ADCSR_0) for Unit 0.....
22.3.3	A/D Control/Status Register 1 (ADCSR_1) for Unit 1.....
22.3.4	A/D Control Register_0 (ADCR_0) for Unit 0.....
22.3.5	A/D Control Register_1 (ADCR_1) for Unit 1.....
22.3.6	A/D Mode Selection Register_0 (ADMOSEL_0) for Unit 0.....
22.3.7	A/D Mode Selection Register_1 (ADMOSEL_1) for Unit 1.....
22.3.8	A/D Sampling State Register_0 (ADSSTR_0) for Unit 0.....
22.3.9	A/D Sampling State Register_1 (ADSSTR_1) for Unit 1.....
22.4	Operation .....
22.4.1	Single Mode.....
22.4.2	Scan Mode .....
22.4.3	Input Sampling and A/D Conversion Time .....
22.4.4	Timing of External Trigger Input.....
22.4.5	Setting the System Clock Mode.....
22.5	Interrupt Source .....
22.6	A/D Conversion Accuracy Definitions .....
22.7	Usage Notes.....
22.7.1	Module Stop Function Setting .....

23.2	Input/Output Pins .....
23.3	Register Descriptions .....
23.3.1	D/A Data Registers 0 and 1 (DADR0, DADR1) .....
23.3.2	D/A Data Registers 0H and 1H (DADR0H and DADR1H) .....
23.3.3	D/A Data Registers 0L and 1L (DADR0L and DADR1L) .....
23.3.4	D/A Data Register 01T (DADR01T) .....
23.3.5	D/A Control Register 01 (DACR01) .....
23.3.6	Usage as an 8-Bit D/A Converter .....
23.4	Operation .....
23.5	Usage Notes .....
23.5.1	Module Stop State Setting .....
23.5.2	D/A Output Hold Function in Software Standby Mode .....
23.5.3	Notes on Deep Software Standby Mode .....
23.5.4	Limitations on Emulators .....

## Section 24 RAM .....

## Section 25 Flash Memory .....

25.1	Features .....
25.2	Mode Transition Diagram .....
25.3	Memory MAT Configuration .....
25.4	Block Structure .....
25.4.1	Block Diagram of H8SX/1662 .....
25.4.2	Block Diagram of H8SX/1665 .....
25.5	Programming/Erasing Interface .....
25.6	Input/Output Pins .....
25.7	Register Descriptions .....
25.7.1	Programming/Erasing Interface Registers .....
25.7.2	Programming/Erasing Interface Parameters .....

25.10	Flash Memory Emulation Using RAM.....
25.11	Switching between User MAT and User Boot MAT.....
25.12	Programmer Mode.....
25.13	Standard Serial Communications Interface Specifications for Boot Mode.....
25.14	Usage Notes.....
Section 26	Boundary Scan.....
26.1	Features.....
26.2	Block Diagram of Boundary Scan Function.....
26.3	Input/Output Pins.....
26.4	Register Descriptions.....
26.4.1	Instruction Register (JTIR).....
26.4.2	Bypass Register (JTBPR).....
26.4.3	Boundary Scan Register (JTBSR).....
26.4.4	IDCODE Register (JTID).....
26.5	Operations.....
26.5.1	TAP Controller.....
26.5.2	Commands.....
26.6	Usage Notes.....
Section 27	Clock Pulse Generator.....
27.1	Register Description.....
27.1.1	System Clock Control Register (SCKCR).....
27.1.2	Subclock Control Register (SUBCKCR).....
27.2	Oscillator.....
27.2.1	Connecting Crystal Resonator.....
27.2.2	External Clock Input.....
27.3	PLL Circuit.....
27.4	Frequency Divider.....

28.2.1	Standby Control Register (SBYCR) .....
28.2.2	Module Stop Control Registers A and B (MSTPCRA and MSTPCRB) .....
28.2.3	Module Stop Control Register C (MSTPCRC).....
28.2.4	Deep Standby Control Register (DPSBYCR).....
28.2.5	Deep Standby Wait Control Register (DPSWCR).....
28.2.6	Deep Standby Interrupt Enable Register (DPSIER) .....
28.2.7	Deep Standby Interrupt Flag Register (DPSIFR).....
28.2.8	Deep Standby Interrupt Edge Register (DPSIEGR) .....
28.2.9	Reset Status Register (RSTSR).....
28.2.10	Deep Standby Backup Register (DPSBKRn) .....
28.3	Multi-Clock Function .....
28.3.1	Switching of Main Clock Frequencies.....
28.3.2	Switching to Subclock .....
28.4	Module Stop State.....
28.5	Sleep Mode .....
28.5.1	Entry to Sleep Mode .....
28.5.2	Exit from Sleep Mode.....
28.6	All-Module-Clock-Stop Mode.....
28.7	Software Standby Mode.....
28.7.1	Entry to Software Standby Mode.....
28.7.2	Exit from Software Standby Mode .....
28.7.3	Setting Oscillation Settling Time after Exit from Software Standby Mode .....
28.7.4	Software Standby Mode Application Example.....
28.8	Deep Software Standby Mode .....
28.8.1	Entry to Deep Software Standby Mode .....
28.8.2	Exit from Deep Software Standby Mode .....
28.8.3	Pin State on Exit from Deep Software Standby Mode.....
28.8.4	B $\phi$ /SDRAM $\phi$ Operation after Exit from Deep Software Standby Mode.



28.12	Usage Notes .....
28.12.1	I/O Port Status.....
28.12.2	Current Consumption during Oscillation Settling Standby Period .....
28.12.3	Module Stop State of EXDMAC, DMAC, or DTC .....
28.12.4	On-Chip Peripheral Module Interrupts .....
28.12.5	Writing to MSTPCRA, MSTPCRB, and MSTPCRC .....
28.12.6	Control of Input Buffers by DIRQnE (n = 3 to 0).....
28.12.7	Conflict between a transition to deep standby mode and interrupts.....
28.12.8	B $\phi$ /SDRAM $\phi$ Output State .....

**Section 29 List of Registers .....**

29.1	Register Addresses (Address Order).....
29.2	Register Bits.....
29.3	Register States in Each Operating Mode .....

**Section 30 Electrical Characteristics .....**

30.1	Absolute Maximum Ratings .....
30.2	DC Characteristics (H8SX/1665 Group) .....
30.3	DC Characteristics (H8SX/1665M Group).....
30.4	AC Characteristics .....
30.4.1	Clock Timing .....
30.4.2	Control Signal Timing .....
30.4.3	Bus Timing .....
30.4.4	DMAC and EXDMAC Timing.....
30.4.5	Timing of On-Chip Peripheral Modules .....
30.5	USB Characteristics .....
30.6	A/D Conversion Characteristics .....
30.7	D/A Conversion Characteristics .....
30.8	Flash Memory Characteristics .....



controller which enable high-speed data transfer, and a bus-state controller, which enable connection to different kinds of memory. The LSI of the Group also includes serial communications interfaces, A/D and D/A converters, and a multi-function timer that make control easy. Together, the modules realize low-cost configurations for end systems. The consumption of these modules is kept down dynamically by an on-chip power-management function. The on-chip ROM is a flash memory (F-ZTAT™\*) with a capacity of 512 Kbytes (H8SX/1665 and H8SX/1665M), or 384 Kbytes (H8SX/1662 and H8SX/1662M).

Note: \* F-ZTAT™ is a trademark of Renesas Technology Corp.

### 1.1.1 Applications

Examples of the applications of this LSI include PC peripheral equipment, optical storage, office automation equipment, and industrial equipment.

CPU	CPU	<ul style="list-style-type: none"> <li>• 32-bit high-speed H8SX CPU (CISC type)</li> <li>Upwardly compatible for H8/300, H8/300H, and H8S CPU object level</li> <li>• General-register architecture (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 compare instructions</li> <li>• Minimum instruction execution time: 20.0 ns (for an ADD instruction while 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 + 42 \rightarrow 42</math> bits)</li> </ul>
	Operating mode	<ul style="list-style-type: none"> <li>• Advanced mode</li> </ul> <p>Normal, middle, or maximum mode is not supported.</p>

		<p>Mode 4: On-chip ROM disabled external extended mode, 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, (selected by driving the MD1 pin low and driving MD0 pins high)</p> <p>Mode 6: On-chip ROM enabled external extended mode (selected by driving the MD0 pin low and driving 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 instruction)</li> </ul>
	Power on reset (POR) *	<ul style="list-style-type: none"> <li>• At power-on or low power supply voltage, an internal reset signal is generated</li> </ul>
	Voltage detection circuit (LVD)*	<ul style="list-style-type: none"> <li>• At low power supply voltage, an internal reset and an interrupt signal are generated.</li> </ul>
Interrupt (source)	Interrupt controller (INTC)	<ul style="list-style-type: none"> <li>• 13 external interrupt pins (NMI, and <math>\overline{IRQ11}</math> to <math>\overline{IRQ0}</math>)</li> <li>• Internal interrupt sources <ul style="list-style-type: none"> <li>H8SX/1665 Group: 122 pins</li> <li>H8SX/1665M Group: 123 pins</li> </ul> </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>

		<ul style="list-style-type: none"> <li>• Extended repeat area function</li> </ul>
	DMA controller (DMAC)	<ul style="list-style-type: none"> <li>• Four-channel DMA transfer available</li> <li>• Three activation methods (auto-request, on-chip module interrupt, and external request)</li> <li>• Three transfer modes (normal, repeat, and block)</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 78 channels (number of DTC activation sources)</li> <li>• Activated by interrupt sources (chain transfer enabled)</li> <li>• Three transfer modes (normal transfer, repeat transfer, 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 internal CPU, DMAC, EXDMAC, DTC, Refresh, and external bus masters)</li> </ul>

Clock

Clock pulse generator (CPG)

- One clock generation circuit available
  - Separate clock signals are provided for each of functional modules (detailed below) and each is independently supplied (multi-clock function)
    - System-intended data transfer modules, i.e. the CPU, are in synchronization with the system clock ( $I\phi$ ): 8 to 50 MHz
    - Internal peripheral functions run in synchronization with the peripheral module clock ( $P\phi$ ): 8 to 35 MHz
    - Modules in the external space are supplied with the external bus clock ( $B\phi$ ): 8 to 50 MHz
  - Includes a PLL frequency multiplication circuit and frequency divider, so the operating frequency is selectable
  - Five low-power-consumption modes: Sleep mode, all-modules clock-stop mode, software standby mode, deep software standby mode, and hardware standby mode
-

- Three ways to start A/D conversion:
  - Unit 0: Software, timer (TPU (unit 0)/TMR (units 0 and 1) trigger, and external trigger
  - Unit 1: Software, TMR (units 2 and 3) trigger, and external trigger
- Activation of DTC and DMAC by ADI interrupt:
  - Unit 0: DTC and DMAC can be activated by an ADI0 interrupt
  - Unit 1: DMAC can be activated by an ADI1 interrupt.

D/A converter	D/A converter (DAC)	<ul style="list-style-type: none"> <li>• 10-bit resolution × two output channels</li> <li>• Output voltage: 0 V to Vref, maximum conversion time: (with 20-pF load)</li> </ul>
Timer	8-bit timer (TMR)	<ul style="list-style-type: none"> <li>• 8 bits × eight channels (can be used as 16 bits × four channels)</li> <li>• Select from among seven clock sources (six internal clocks, one external clock)</li> <li>• Allows the output of pulse trains with a desired duty cycle (PWM signals)</li> </ul>



channels), and phase counting mode (two-phase encoder input) settable for each channel

- Input capture function supported
- Output compare function (by the output of compare mode waveform) supported

Note: \* Pin function of unit 1 cannot be used in the external extended mode.

- 
- |   |   |
|---|---|
| Program-<br>mable pulse<br>generator<br>(PPG) | • 32-bit* <sup>1</sup> * <sup>2</sup> pulse output  |
|   | • Four output groups, non-overlapping mode, and invert can be set   |
|   | • Selectable output trigger signals; the PPG can operate in conjunction with the data transfer controller (DTC) and the controller (DMAC) |

Notes: 1. Pulse output pins PO31 to PO16 cannot be used for input capture.

2. Pulse of unit 1 cannot be output in external bus extended mode.

- 
- |                |                      |   |
|----------------|----------------------|---|
| Watchdog timer | Watchdog timer (WDT) | <ul style="list-style-type: none"><li>• 8 bits × one channels (selectable from eight counter input clocks)</li><li>• Switchable between watchdog timer mode and interval mode</li></ul> |
|----------------|----------------------|---|

- 
- |           |                   |  |
|-----------|-------------------|--|
| 32K timer | 32K timer (TM32K) | <ul style="list-style-type: none"><li>• Eight counter clocks which divides the 32.768 kHz clock selected</li><li>• 8 bits × 1 channel or 24 bits × 1 channel can be selected</li><li>• Interrupts can be generated when the counter overflows</li><li>• Eight overflow cycles selectable (250 msec, 500 msec, 1 sec, 30 sec, 60 sec, about 23 days, and about 46 days)</li></ul> |
|-----------|-------------------|--|
-

Smart card/On-chip		The USB module supports a smart card (On-chip) interface.
Universal serial bus interface	Universal serial bus interface (USB)	<ul style="list-style-type: none"> <li>• On-chip UDC (USB Device Controller) supporting USB transceiver</li> <li>• Transfer speed: full-speed (12 Mbps)</li> <li>• Bulk transfer by DMA</li> <li>• Self-power mode and bus power mode selectable</li> </ul>
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 SDA pins are N open drains).</li> </ul>
I/O ports		<ul style="list-style-type: none"> <li>• 9 CMOS input-only pins</li> <li>• 92 CMOS input/output pins</li> <li>• 8 large-current drive pins (port 3)</li> <li>• 40 pull-up resistors</li> <li>• 16 open drains</li> </ul>
Emulator		<p>Full-spec emulator (E6000H)*</p> <p>On-chip emulator (E10A-USB)</p> <p>Note: * E6000H of the H8SX/1668 Group can be used as a part of function of A/D and D/A though the H8SX Group is not supporting E6000H. For details on functions for which E6000H cannot be used, see section 22, A/D Converter and section 23, D/A Converter.</p>
Package		<ul style="list-style-type: none"> <li>• LQFP-144 package</li> <li>• LGA-145 package</li> </ul>

**Table 1.2 Comparison of Support Functions in the H8SX/1665 and H8SX/1665M**

Function	H8SX/1665 Group	H8SX/1665M Group
DMAC	○	○
DTC	○	○
PPG	○	○
UBC	○	○
SCI	○	○
IIC2	○	○
TMR	○	○
WDT	○	○
10-bit ADC	○	○
10-bit DAC	○	○
EXDMAC	○	○
SDRAM interface	○	○
32K timer	○	○
POR/LVD	—	○
Package	LQFP-144	○
	LGA-145	○

	R5F61662N50LGV	384 Kbytes	40 Kbytes	LGA-145	
	R5F61665D50FPV	512 Kbytes	40 Kbytes	LQFP-144	Wide ra specific
	R5F61662D50FPV	384 Kbytes	40 Kbytes	LQFP-144	
	R5F61665D50LGV	512 Kbytes	40 Kbytes	LGA-145	
	R5F61662D50LGV	384 Kbytes	40 Kbytes	LGA-145	
H8SX/1665M	R5F61665MN50FPV	512 Kbytes	40 Kbytes	LQFP-144	Regular specific
	R5F61662MN50FPV	384 Kbytes	40 Kbytes	LQFP-144	
	R5F61665MN50LGV	512 Kbytes	40 Kbytes	LGA-145	
	R5F61662MN50LGV	384 Kbytes	40 Kbytes	LGA-145	
	R5F61665MD50FPV	512 Kbytes	40 Kbytes	LQFP-144	Wide ra specific
	R5F61662MD50FPV	384 Kbytes	40 Kbytes	LQFP-144	
	R5F61665MD50LGV	512 Kbytes	40 Kbytes	LGA-145	
	R5F61662MD50LGV	384 Kbytes	40 Kbytes	LGA-145	

Product classification

Microcontroller

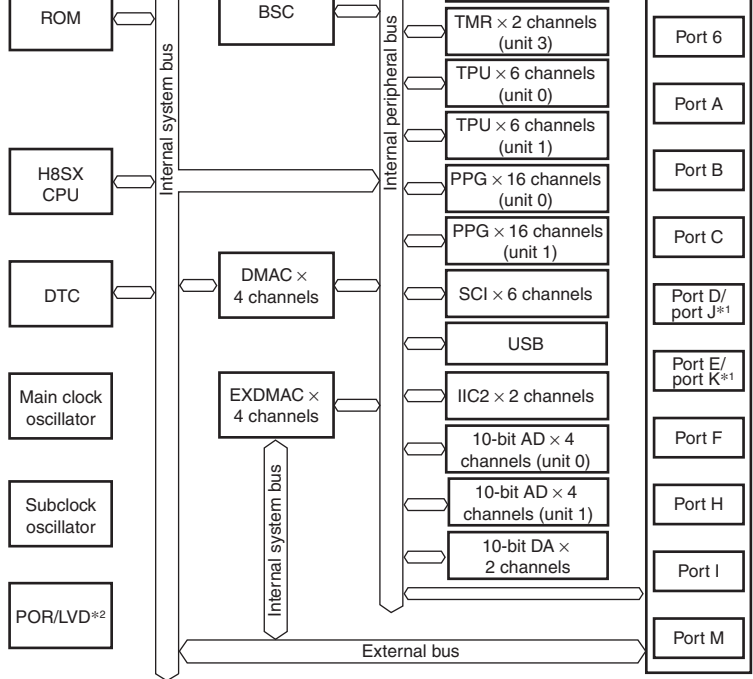
Indicates a Renesas semiconductor product

**Figure 1.1 How to Read the Product Name Code**

- Small Package

<b>Package</b>	<b>Package Code</b>	<b>Body Size</b>	<b>Pin Pitch</b>
LQFP-144	PLQP0144KA-A (FP-144LV)*	20.0 × 20.0 mm	0.50 mm
LGA-145	PTLG0145JB-A (TLP-145V)*	9.0 × 9.0 mm	0.65 mm

Note: \* Pb-free version



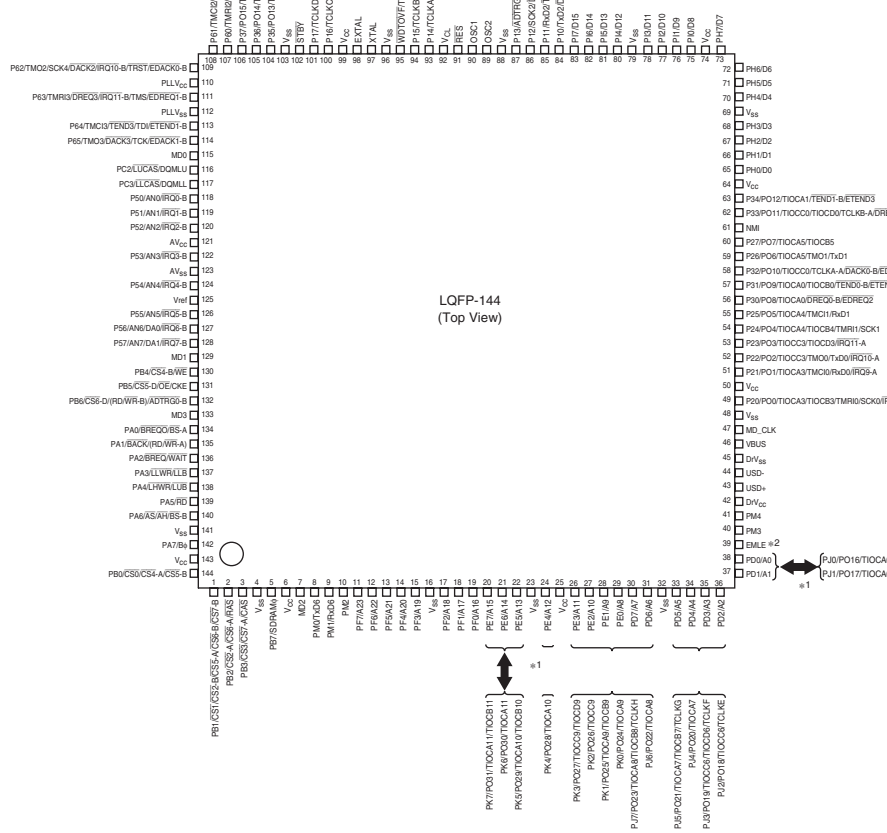
[Legend]

CPU: Central processing unit  
 DTC: Data transfer controller  
 BSC: Bus controller  
 DMAC: DMA controller  
 EXDMAC: EXDMA controller  
 TM32K: 32K timer  
 WDT: Watchdog timer

TMR: 8-bit timer  
 TPU: 16-bit timer pulse unit  
 PPG: Programmable pulse generator  
 SCI: Serial communications interface  
 USB: Universal serial bus interface  
 IIC2: IIC bus interface 2  
 POR/LVD\*2: Power-on reset / Low voltage detection circuit

- Notes: 1. In single-chip mode, the port D and port E functions can be used in the initial state. Pin functions are selectable by setting the PCJKE bit in PFCRD. Ports D and E are enabled when PCJKE = 0 (initial value) and ports J and K are enabled when PCJKE = 1. In external extended mode, only ports D and E can be used.  
 2. Supported only by the H8SX/1665M Group.

**Figure 1.2 Block Diagram**



LQFP-144 (Top View)

- Notes:
1. In single-chip mode ports D and E can be used (initial state). Pin functions are selectable by setting the PCJKE bit in PFCRD. Ports D and E are enabled when PCJKE = 0 (initial value) and ports J and K are enabled when PCJKE = 1. In external extended mode, only ports D and E can be used.
  2. 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 PB2, P63, P64, P65, and WDIOVF 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 (LQFP-144: H8SX/1665 Group and H8SX/1665M)



F	PF4	PF2	Vss	PF3					Vcc	P15	Vss	WDT0	
G	PF1	PE6 /PK6*1	PE7 /PK7*1	Vss	LGA-145 (Upper perspective view)				P14	P13	Vss	RES	
H	PE5 /PK5*1	PE4 /PK4*1	PE3 /PK3*1	PF0					Vcl	P11	OSC1	OSC	
J	Vcc	PE1 /PK1*1	PD7 /PJ7*1	PE2 /PK2*1					P17	P10	P12	P16	
K	PE0 /PK0*1	PD5 /PJ5*1	PD4 /PJ4*1	PD6 /PJ6*1					Vcc	P21	P25	P26	P34
L	PD3 /PJ3*1	Vss	EMLE*2	VBUS	MD_CLK	P24	P32	P33	NMI	PH1	PI2	PI3	PI1
M	PD2 /PJ2*1	PD1 /PJ1*1	DrVcc	USD-	Vss	P20	P23	P31	PH0	Vss	PH4	PH7	PI0
N	PD0 /PJ0*1	PM3	PM4	USD+	DrVss	P22	P30	P27	Vcc	PH3	PH5	PH6	Vcc
	1	2	3	4	5	6	7	8	9	10	11	12	13

- Notes:
1. In single-chip mode ports D and E can be used (initial state). Pin functions are selectable by setting the PCJKE bit in PFCRD. Ports D and E are enabled when PCJKE = 0 (initial value) and ports J and K are enabled when PCJKE = 1. In external extended mode, only ports D and E can be used.
  2. 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.
  3. The NC (no-connection) pin should be left open.

**Figure 1.4 Pin Assignments (LGA-145: H8SX/1665 Group and H8SX/1665M G**



3	C1	PB3/CS3/CS7-A/CAS	PB3/CS3/CS7-A/CAS	PB3/CS3/CS7-A/CAS
4	B3	Vss	Vss	Vss
5	C2	PB7/SDRAMφ	PB7/SDRAMφ	PB7/SDRAMφ
6	D4	Vcc	Vcc	Vcc
7	D1	MD2	MD2	MD2
8	D3	PM0/TxD6	PM0/TxD6	PM0/TxD6
9	D2	PM1/RxD6	PM1/RxD6	PM1/RxD6
10	E1	PM2	PM2	PM2
11	E4	PF7/A23	PF7/A23	PF7/A23
12	E3	PF6/A22	PF6/A22	PF6/A22
13	E2	PF5/A21	PF5/A21	PF5/A21
14	F1	PF4/A20	PF4/A20	A20
15	F4	PF3/A19	PF3/A19	A19
16	F3	Vss	Vss	Vss
17	F2	PF2/A18	PF2/A18	A18
18	G1	PF1/A17	PF1/A17	A17
19	H4	PF0/A16	PF0/A16	A16
20	G3	PE7/A15	PE7/A15 PK7/PO31/TIOCA11/TIOCB11* <sup>1</sup>	A15
21	G2	PE6/A14	PE6/A14 PK6/PO30/TIOCA11* <sup>1</sup>	A14
22	H1	PE5/A13	PE5/A13 PK5/PO29/TIOCA10/TIOCB10* <sup>1</sup>	A13
23	G4	Vss	Vss	Vss
24	H2	PE4/A12	PE4/A12 PK4/PO28/TIOCA10* <sup>1</sup>	A12
25	J1	Vcc	Vcc	Vcc

31	K4	PD6/A6	PD6/A6	PJ6/PO22/TIOCA6*	A6
32	L2	Vss	Vss		Vss
33	K2	PD5/A5	PD5/A5	PJ5/PO21/TIOCA7/TIOCB7/ TCLKG* <sup>1</sup>	A5
34	K3	PD4/A4	PD4/A4	PJ4/PO20/TIOCA7* <sup>1</sup>	A4
35	L1	PD3/A3	PD3/A3	PJ3/PO19/TIOCC6/TIOCD6/ TCLKF* <sup>1</sup>	A3
36	M1	PD2/A2	PD2/A2	PJ2/PO18/TIOCC6/TCLKE* <sup>1</sup>	A2
37	M2	PD1/A1	PD1/A1	PJ1/PO17/TIOCA6/TIOCB6* <sup>1</sup>	A1
38	N1	PD0/A0	PD0/A0	PJ0/PO16/TIOCA6* <sup>1</sup>	A0
39	L3	EMLE	EMLE		EMLE
40	N2	PM3	PM3		PM3
41	N3	PM4	PM4		PM4
42	M3	DrVcc	DrVcc		DrVcc
43	N4	USD+	USD+		USD+
44	M4	USD-	USD-		USD-
45	N5	DrVss	DrVss		DrVss
46	L4	VBUS	VBUS		VBUS
47	L5	MD_CLK	MD_CLK		MD_CLK
48	M5	Vss	Vss		Vss
49	M6	P20/PO0/TIOCA3/TIOCB3/ TMR10/SCK0/IRQ8-A	P20/PO0/TIOCA3/TIOCB3/TMR10/ SCK0/IRQ8-A		P20/PO0/TIOCA3/TIO SCK0/IRQ8-A
50	K5	Vcc	Vcc		Vcc
51	K6	P21/PO1/TIOCA3/TMCI0/RxD0/ IRQ9-A	P21/PO1/TIOCA3/TMCI0/RxD0/ IRQ9-A		P21/PO1/TIOCA3/TM IRQ9-A

56	N7	P30/PO8/TIOCA0/DREQ0-B/ EDREQ2	P30/PO8/TIOCA0/DREQ0-B/ EDREQ2	P30/PO8/TIOCA0/DREQ0-B/ EDREQ2
57	M8	P31/PO9/TIOCA0/TIOCB0/ TEND0-B/ETEND2	P31/PO9/TIOCA0/TIOCB0/ TEND0-B/ETEND2	P31/PO9/TIOCA0/TIOCB0/ TEND0-B/ETEND2
58	L7	P32/PO10/TIOCC0/ TCLKA-A/DACK0-B/EDACK2	P32/PO10/TIOCC0/ TCLKA-A/DACK0-B/EDACK2	P32/PO10/TIOCC0/ TCLKA-A/DACK0-B/EDACK2
59	K8	P26/PO6/TIOCA5/TMO1/TxD1	P26/PO6/TIOCA5/TMO1/TxD1	P26/PO6/TIOCA5/TMO1/TxD1
60	N8	P27/PO7/TIOCA5/TIOCB5	P27/PO7/TIOCA5/TIOCB5	P27/PO7/TIOCA5/TIOCB5
61	L9	NMI	NMI	NMI
62	L8	P33/PO11/TIOCC0/TIOCD0/ TCLKB-A/DREQ1-B/EDREQ3	P33/PO11/TIOCC0/TIOCD0/ TCLKB-A/DREQ1-B/EDREQ3	P33/PO11/TIOCC0/TIOCD0/ TCLKB-A/DREQ1-B/EDREQ3
63	K9	P34/PO12/TIOCA1/ TEND1-B/ETEND3	P34/PO12/TIOCA1/TEND1-B/ ETEND3	P34/PO12/TIOCA1/ TEND1-B/ETEND3
64	N9	Vcc	Vcc	Vcc
65	M9	PH0/D0	PH0/D0	D0
66	L10	PH1/D1	PH1/D1	D1
67	K10	PH2/D2	PH2/D2	D2
68	N10	PH3/D3	PH3/D3	D3
69	M10	Vss	Vss	Vss
70	M11	PH4/D4	PH4/D4	D4
71	N11	PH5/D5	PH5/D5	D5
72	N12	PH6/D6	PH6/D6	D6
73	M12	PH7/D7	PH7/D7	D7
74	N13	Vcc	Vcc	Vcc
75	M13	PI0/D8	PI0/D8	PI0/D8

82	J13	P16/D14	P16/D14	P16/D14
83	J10	PI7/D15	PI7/D15	PI7/D15
84	J11	P10/TxD2/ $\overline{\text{DREQ0-A}}$ / $\overline{\text{IRQ0-A}}/\overline{\text{EDREQ0-A}}$	P10/TxD2/ $\overline{\text{DREQ0-A}}$ / $\overline{\text{IRQ0-A}}$ / $\overline{\text{EDREQ0-A}}$	P10/TxD2/ $\overline{\text{DREQ0-A}}$ / $\overline{\text{IRQ0-A}}/\overline{\text{EDREQ0-A}}$
85	H11	P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}/\overline{\text{ETEND0-A}}$	P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}/\overline{\text{ETEND0-A}}$	P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}/\overline{\text{ETEND0-A}}$
86	J12	P12/SCK2/ $\overline{\text{DACK0-A}}$ / $\overline{\text{IRQ2-A}}/\overline{\text{EDACK0-A}}$	P12/SCK2/ $\overline{\text{DACK0-A}}$ / $\overline{\text{IRQ2-A}}/\overline{\text{EDACK0-A}}$	P12/SCK2/ $\overline{\text{DACK0-A}}$ / $\overline{\text{IRQ2-A}}/\overline{\text{EDACK0-A}}$
87	G11	P13/ $\overline{\text{ADTRG0-A}}$ / $\overline{\text{IRQ3-A}}/\overline{\text{EDRAK0}}$	P13/ $\overline{\text{ADTRG0-A}}$ / $\overline{\text{IRQ3-A}}$ / $\overline{\text{EDRAK0}}$	P13/ $\overline{\text{ADTRG0-A}}$ / $\overline{\text{IRQ3-A}}$ / $\overline{\text{EDRAK0}}$
88	G12	Vss	Vss	Vss
89	H13	OSC2	OSC2	OSC2
90	H12	OSC1	OSC1	OSC1
91	G13	RES	RES	RES
92	H10	V <sub>CL</sub>	V <sub>CL</sub>	V <sub>CL</sub>
93	G10	P14/TCLKA-B/TxD5/IrTxD/ SDA1/ $\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ / $\overline{\text{EDREQ1-A}}$	P14/TCLKA-B/TxD5/IrTxD/SDA1/ $\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ / $\overline{\text{EDREQ1-A}}$	P14/TCLKA-B/TxD5/IrTxD/ $\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ / $\overline{\text{EDREQ1-A}}$
94	F11	P15/TCLKB-B/RxD5/IrRxD/SCL1/ $\overline{\text{TEND1-A}}$ / $\overline{\text{IRQ5-A}}$ / $\overline{\text{ETEND1-A}}$	P15/TCLKB-B/RxD5/IrRxD/SCL1/ $\overline{\text{TEND1-A}}$ / $\overline{\text{IRQ5-A}}$ / $\overline{\text{ETEND1-A}}$	P15/TCLKB-B/RxD5/IrRxD/ $\overline{\text{TEND1-A}}$ / $\overline{\text{IRQ5-A}}$ / $\overline{\text{ETEND1-A}}$
95	F13	$\overline{\text{WDTOVF}}$	$\overline{\text{WDTOVF}}/\text{TDO}^{*2}$	$\overline{\text{WDTOVF}}$
96	F12	Vss	Vss	Vss
97	E12	XTAL	XTAL	XTAL
98	E13	EXTAL	EXTAL	EXTAL
99	F10	Vcc	Vcc	Vcc

		TCLKC-A/DACK1-B/EDACK3	TCLKC-A/DACK1-B/EDACK3	TCLKC-A/DACK1-B/EDACK3
105	C12	P36/PO14/TIOCA2/EDRAK2	P36/PO14/TIOCA2/EDRAK2	P36/PO14/TIOCA2/EDRAK2
106	C13	P37/PO15/TIOCA2/TIOCB2/ TCLKD-A/EDRAK3	P37/PO15/TIOCA2/TIOCB2/ TCLKD-A/EDRAK3	P37/PO15/TIOCA2/TIOCB2/ TCLKD-A/EDRAK3
107	B13	P60/TMRI2/TxD4/DREQ2/ IRQ8-B/EDREQ0-B	P60/TMRI2/TxD4/DREQ2/ IRQ8-B/EDREQ0-B	P60/TMRI2/TxD4/DREQ2/ IRQ8-B/EDREQ0-B
108	A13	P61/TMCI2/RxD4/TEND2/ IRQ9-B/ETEND0-B	P61/TMCI2/RxD4/TEND2/ IRQ9-B/ETEND0-B	P61/TMCI2/RxD4/TEND2/ IRQ9-B/ETEND0-B
109	B12	P62/TMO2/SCK4/DACK2/ IRQ10-B/EDACK0-B	P62/TMO2/SCK4/DACK2/ IRQ10-B/TRST*/EDACK0-B	P62/TMO2/SCK4/DACK2/ IRQ10-B/EDACK0-B
110	A12	PLLvcc	PLLvcc	PLLvcc
111	C11	P63/TMRI3/DREQ3/ IRQ11-B/EDREQ1-B	P63/TMRI3/DREQ3/ IRQ11-B/TMS*/EDREQ1-B	P63/TMRI3/DREQ3/ IRQ11-B/EDREQ1-B
112	A11	PLLvss	PLLvss	PLLvss
113	B11	P64/TMCI3/TEND3/ETEND1-B	P64/TMCI3/TEND3/TD1*/ETEND1-B	P64/TMCI3/TEND3/ETEND1-B
114	A10	P65/TMO3/DACK3/EDACK1-B	P65/TMO3/DACK3/TCK*/EDACK1-B	P65/TMO3/DACK3/EDACK1-B
115	D10	MD0	MD0	MD0
116	C10	PC2/LUCAS/DQMLU	PC2/LUCAS/DQMLU	PC2/LUCAS/DQMLU
117	B10	PC3/LLCAS/DQMLL	PC3/LLCAS/DQMLL	PC3/LLCAS/DQMLL
118	A9	P50/AN0/IRQ0-B	P50/AN0/IRQ0-B	P50/AN0/IRQ0-B
119	C9	P51/AN1/IRQ1-B	P51/AN1/IRQ1-B	P51/AN1/IRQ1-B
120	B9	P52/AN2/IRQ2-B	P52/AN2/IRQ2-B	P52/AN2/IRQ2-B
121	B8	AVcc	AVcc	AVcc
122	D9	P53/AN3/IRQ3-B	P53/AN3/IRQ3-B	P53/AN3/IRQ3-B
123	C8	AVss	AVss	AVss

130	C7	PB4/ $\overline{\text{CS4-B/WE}}$	PB4/ $\overline{\text{CS4-B/WE}}$	PB4/ $\overline{\text{CS4-B/WE}}$
131	C6	PB5/ $\overline{\text{CS5-D/OE/CKE}}$	PB5/ $\overline{\text{CS5-D/OE/CKE}}$	PB5/ $\overline{\text{CS5-D/OE/CKE}}$
132	A6	PB6/ $\overline{\text{CS6-D/}}$ (RD/ $\overline{\text{WR-B}}$ )/ $\overline{\text{ADTRG0-B}}$	PB6/ $\overline{\text{CS6-D/}}$ (RD/ $\overline{\text{WR-B}}$ )/ $\overline{\text{ADTRG0-B}}$	PB6/ $\overline{\text{CS6-D/}}$ (RD/ $\overline{\text{WR-B}}$ )/ $\overline{\text{ADTRG0-B}}$
133	B5	MD3	MD3	MD3
134	D6	PA0/ $\overline{\text{BREQO/BS-A}}$	PA0/ $\overline{\text{BREQO/BS-A}}$	PA0/ $\overline{\text{BREQO/BS-A}}$
135	A5	PA1/ $\overline{\text{BACK/}}$ (RD/ $\overline{\text{WR-A}}$ )	PA1/ $\overline{\text{BACK/}}$ (RD/ $\overline{\text{WR-A}}$ )	PA1/ $\overline{\text{BACK/}}$ (RD/ $\overline{\text{WR-A}}$ )
136	B4	PA2/ $\overline{\text{BREQ/WAIT}}$	PA2/ $\overline{\text{BREQ/WAIT}}$	PA2/ $\overline{\text{BREQ/WAIT}}$
137	D5	PA3/ $\overline{\text{LLWR/LLB}}$	PA3/ $\overline{\text{LLWR/LLB}}$	$\overline{\text{LLWR/LLB}}$
138	A4	PA4/ $\overline{\text{LHWR/LUB}}$	PA4/ $\overline{\text{LHWR/LUB}}$	PA4/ $\overline{\text{LHWR/LUB}}$
139	C5	PA5/ $\overline{\text{RD}}$	PA5/ $\overline{\text{RD}}$	$\overline{\text{RD}}$
140	C4	PA6/ $\overline{\text{AS/AH/BS-B}}$	PA6/ $\overline{\text{AS/AH/BS-B}}$	PA6/ $\overline{\text{AS/AH/BS-B}}$
141	C3	Vss	Vss	Vss
142	A2	PA7/B $\phi$	PA7/B $\phi$	PA7/B $\phi$
143	A3	Vcc	Vcc	Vcc
144	B2	PB0/ $\overline{\text{CS0/CS4-A/CS5-B}}$	PB0/ $\overline{\text{CS0/CS4-A/CS5-B}}$	PB0/ $\overline{\text{CS0/CS4-A/CS5-B}}$
—	E5	NC	NC	NC

- Notes: 1. These pins can be used when the PCJKE bit in PFCRD is set to 1 in single-chip mode.  
2. Pins TDO,  $\overline{\text{TRST}}$ , TMS, and TCK are enabled in mode 3.

	PLL <sub>V<sub>cc</sub></sub>	Input	Power supply pin for the PLL circuit. Connect the pin to the system power supply.
	PLL <sub>V<sub>ss</sub></sub>	Input	Ground pin for the PLL circuit.
	DrV <sub>cc</sub>	Input	Power supply pin for the on-chip USB transceiver. Connect this pin to the system power supply.
	DrV <sub>ss</sub>	Input	Ground pin for the on-chip USB transceiver.
Clock	XTAL	Input	Pins for a crystal resonator. An external clock signal is connected to the XTAL pin through the EXTAL pin. For an example of the connection, see section 27, Clock Pulse Generator.
	EXTAL	Input	
	OSC1	Input	The 32.768 KHz crystal resonator is connected to the OSC1 pin.
	OSC2	Input	The 32.768 KHz crystal resonator is connected to the OSC2 pin.
	B $\phi$	Output	Outputs the system clock for external devices.
	SDRAM $\phi$	Output	When connecting the synchronous DRAM, connect this pin to the CLK pin of synchronous DRAM. For details, see section 28, SDRAM Bus Controller (BSC).
Operating mode control	MD3 to MD0	Input	Pins for setting the operating mode. The signal level of these pins must not be changed during operation.
	MD_CLK	Input	This pin changes the multiplication ratio of the clock oscillator. Do not change values on this pin during operation.
System control	RES	Input	Reset signal input pin. This LSI enters the reset state when this signal goes low.
	STBY	Input	This LSI enters hardware standby mode when this signal goes low.
	EMLE	Input	Input pin for the on-chip emulator enable signal. The signal level should normally be fixed low.

Bus control	$\overline{\text{BREQ}}$	Input	External bus-master modules assert this signal to 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 released state.
	$\overline{\text{BACK}}$	Output	Bus acknowledge signal, which indicates that the bus has been released.
	$\overline{\text{BS-A}}/\overline{\text{BS-B}}$	Output	Indicates the start of a bus cycle.
	$\overline{\text{AS}}$	Output	Strobe signal which indicates that the output address bus is valid in access to the basic bus interface byte control SRAM interface space.
	$\overline{\text{AH}}$	Output	This signal is used to hold the address when accessing address-data multiplexed I/O interface space.
	$\overline{\text{RD}}$	Output	Strobe signal which indicates that reading from the interface space is in progress.
	$(\overline{\text{RD}}/\overline{\text{WR}}\text{-A})/(\overline{\text{RD}}/\overline{\text{WR}}\text{-B})$	Output	Indicates the direction (input or output) of the data.
	$\overline{\text{LHWR}}$	Output	Strobe signal which indicates that the higher-order address (A0 to D8) is valid in access to the basic bus interface space.
	$\overline{\text{LLWR}}$	Output	Strobe signal which indicates that the lower-order address (A0 to D0) is valid in access to the basic bus interface space.
	$\overline{\text{LUB}}$	Output	Strobe signal which indicates that the higher-order address (A0 to D8) is valid in access to the byte control SRAM interface space.
	$\overline{\text{LLB}}$	Output	Strobe signal which indicates that the lower-order address (A0 to D0) is valid in access to the byte control SRAM interface space.



$\overline{\text{RAS}}$	Output	<ul style="list-style-type: none"> <li>Row address strobe signal for the DRAM when area 1 is specified as the DRAM interface space</li> <li>Row address strobe signal for the synchronous DRAM when area 2 is specified as the synchronous DRAM interface space</li> </ul>
$\overline{\text{CAS}}$	Output	<ul style="list-style-type: none"> <li>Column address strobe signal for the synchronous DRAM when area 2 is specified as the synchronous DRAM interface space</li> </ul>
$\overline{\text{WE}}$	Output	<ul style="list-style-type: none"> <li>Write enable signal for the DRAM space</li> <li>Synchronous DRAM write enable signal when area 2 is specified as the synchronous DRAM interface space</li> </ul>
$\overline{\text{OE/CKE}}$	Output	<ul style="list-style-type: none"> <li>Output enable signal for the DRAM interface space</li> <li>Clock enable signal for the synchronous DRAM interface space</li> </ul>
$\overline{\text{UCAS}}$	Output	<ul style="list-style-type: none"> <li>Upper column address strobe signal for the synchronous DRAM interface space</li> </ul>
$\overline{\text{LLCAS}}$	Output	<ul style="list-style-type: none"> <li>Lower column address strobe signal for the synchronous DRAM interface space</li> <li>Column address strobe signal for the 8-bit synchronous DRAM interface space</li> </ul>
$\overline{\text{DQMLU}}$	Output	<ul style="list-style-type: none"> <li>Upper data mask enable signal for the 16-bit synchronous DRAM interface space</li> </ul>
$\overline{\text{DQMLL}}$	Output	<ul style="list-style-type: none"> <li>Lower data mask enable signal for the 16-bit synchronous DRAM interface space</li> <li>Data mask enable signal for the 8-bit synchronous DRAM interface space</li> </ul>

IRQ2-A/IRQ2-B  
IRQ1-A/IRQ1-B  
IRQ0-A/IRQ0-B

---

DMA controller (DMAC)	$\overline{\text{DREQ0-A}}/\overline{\text{DREQ0-B}}$	Input	Requests DMAC activation.
	$\overline{\text{DREQ1-A}}/\overline{\text{DREQ1-B}}$		
	DREQ2 DREQ3		
	$\overline{\text{DACK0-A}}/\overline{\text{DACK0-B}}$	Output	DMAC single address-transfer acknowledge signal
	$\overline{\text{DACK1-A}}/\overline{\text{DACK1-B}}$		
	DACK2 DACK3		
	$\overline{\text{TEND0-A}}/\overline{\text{TEND0-B}}$	Output	Indicates end of data transfer by the DMAC.
	$\overline{\text{TEND1-A}}/\overline{\text{TEND1-B}}$		
	TEND2 TEND3		

---

	EDACK3		
	ETEND0-A/ ETEND0-B ETEND1-A/ ETEND1-B ETEND2 ETEND3	Output	Indicates end of data transfer by the EXDMAC.
	EDRAK0 EDRAK1	Output	Notification to external device of EXDMAC external acceptance and start of execution.
16-bit timer pulse unit (TPU)	TCLKA-A/TCLKA-B TCLKB-A/TCLKB-B TCLKC-A/TCLKC-B TCLKD-A/TCLKD-B	Input	Input pins for the external clock signals.
	TIOCA0 TIOCB0 TIOCC0 TIOCD0	Input/ output	Signals for TGRA_0 to TGRD_0. These pins are input capture inputs, output compare outputs, or outputs.
	TIOCA1 TIOCB1	Input/ output	Signals for TGRA_1 and TGRB_1. These pins are input capture inputs, output compare outputs, or outputs.
	TIOCA2 TIOCB2	Input/ output	Signals for TGRA_2 and TGRB_2. These pins are input capture inputs, output compare outputs, or outputs.
	TIOCA3 TIOCB3 TIOCC3 TIOCD3	Input/ output	Signals for TGRA_3 to TGRD_3. These pins are input capture inputs, output compare outputs, or outputs.
	TIOCA4 TIOCB4	Input/ output	Signals for TGRA_4 and TGRB_4. These pins are input capture inputs, output compare outputs, or outputs.

	TIOCA7 TIOCB7	Input/ output	Signals for TGRA_7 and TGRB_7. These pins are input capture inputs, output compare outputs, or P outputs.
	TIOCA8 TIOCB8	Input/ output	Signals for TGRA_8 and TGRB_8. These pins are input capture inputs, output compare outputs, or P outputs.
	TIOCA9 TIOCB9 TIOCC9 TIOCD9	Input/ output	Signals for TGRA_9 to TGRB_9. These pins are input capture inputs, output compare outputs, or P outputs.
	TIOCA10 TIOCB10	Input/ output	Signals for TGRA_10 and TGRB_10. These pins are input capture inputs, output compare outputs, or P outputs.
	TIOCA11 TIOCB11	Input/ output	Signals for TGRA_11 and TGRB_11. These pins are input capture inputs, output compare outputs, or P outputs.
Programmable pulse generator (PPG)	PO31 to PO0	Output	Output pins for the pulse signals.
8-bit timer (TMR)	TMO0 to TMO3	Output	Output pins for the compare match signals.
	TMCI0 to TMCI3	Input	Input pins for the external clock signals that drive the counters.
	TMRI0 to TMRI3	Input	Input pins for the counter-reset signals.
Watchdog timer (WDT)	WDT $\overline{OVF}$	Output	Output pin for the counter-overflow signal in watch mode.

	RxD6		
	SCK0 SCK1 SCK2 SCK4	Input/ output	Input/output pins for clock signals.
SCI with IrDA (SCI)	IrTxD	Output	Output pin that outputs encoded data for IrDA.
	IrRxD	Input	Input pin that inputs encoded data for IrDA.
I <sup>2</sup> C bus interface 2 (IIC2)	SCL0 SCL1	Input/ output	Input/output pin for IIC clock. Bus can be directly the NMOS open drain output.
	SDA0 SDA1	Input/ output	Input/output pin for IIC data. Bus can be directly the NMOS open drain output.
	USD+	Input/ output	Input/output pin for USB data.
	USD-		
Universal serial interface (USB)	VBUS	Input	Input pin for detecting the connection/disconnect USB cable.
	A/D converter	AN7 to AN0	Input
	$\overline{\text{ADTRG0-A}}$ / $\overline{\text{ADTRG0-B}}$ $\overline{\text{ADTRG1}}$	Input	Input pins for the external trigger signal that starts conversion.
D/A converter	DA1, DA0	Output	Output pins for the analog signals from the D/A c

P27 to P20	Input/ output	8-bit input/output pins.
P37 to P30	Input/ output	8-bit input/output pins.
P57 to P50	Input	8-bit input-only pins.
P65 to P60	Input/ output	6-bit input/output pins.
PA7	Input	Input-only pin.
PA6 to PA0	Input/ output	7-bit input/output pins.
PB7 to PB0	Input/ output	8-bit input/output pins.
PC3 to PC2	Input/ output	2-bit input/output pins.
PD7 to PD0	Input/ output	8-bit input/output pins.
PE7 to PE0	Input/ output	8-bit input/output pins.
PF7 to PF0	Input/ output	8-bit input/output pins.
PH7 to PH0	Input/ output	8-bit input/output pins.
PI7 to PI0	Input/ output	8-bit input/output pins.
PM4 to PM0	Input/ output	5-bit input/output pins.



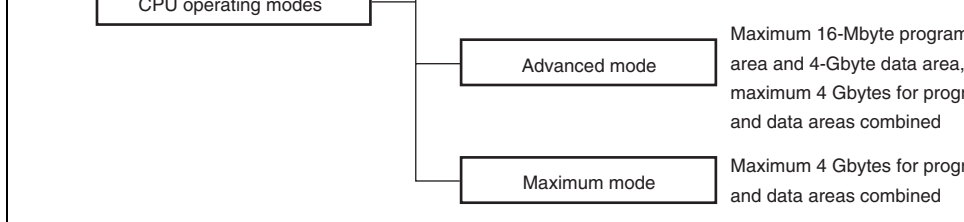




- Upward-compatible with H8/500, H8/500H, and H8S CPUs
  - Can execute object programs of these CPUs
- Sixteen 16-bit general registers
  - Also usable as sixteen 8-bit registers or eight 32-bit registers
- 87 basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide 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]

- 8 × 8-bit register-register multiply: 1 state (When the multiplier is available)
- 16 ÷ 8-bit register-register divide: 10 states (When the multiplier is available)
- 16 × 16-bit register-register multiply: 1 state (When the multiplier is available)
- 32 ÷ 16-bit register-register divide: 18 states (When the multiplier is available)
- 32 × 32-bit register-register multiply: 5 states (When the multiplier is available)
- 32 ÷ 32-bit register-register divide: 18 states (When the multiplier is available)
- 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/1665M Group and H8SX/1665M Group. Normal, middle, and maximum modes are not supported.
  2. The multiplier and divider are supported by the H8SX/1665 Group and H8SX/1665M Group.



**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, pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, 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.

## Figure 2.2 Exception Vector Table (Normal Mode)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing 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

- 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

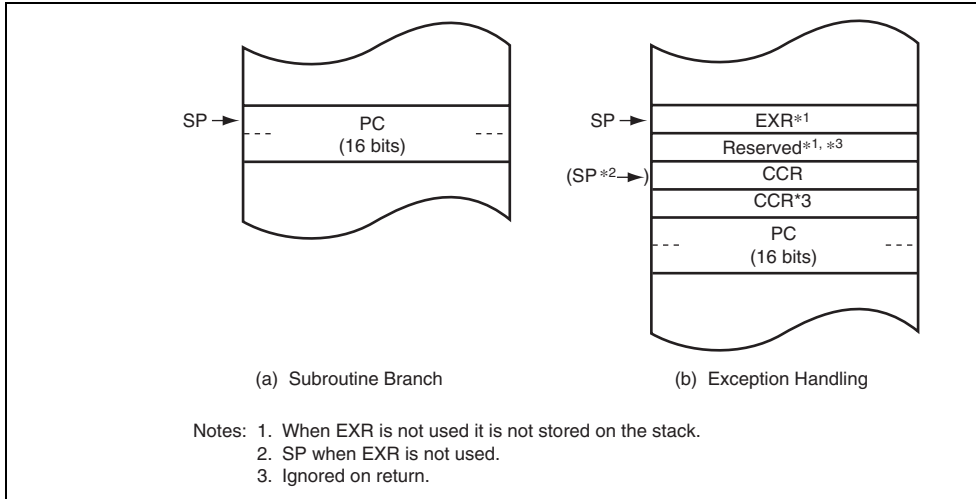


Figure 2.3 Stack Structure (Normal Mode)

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 (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 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**

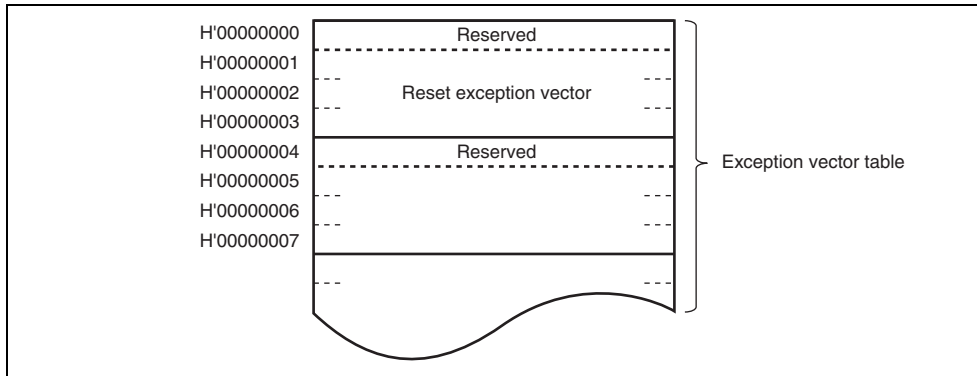
The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.

- Instruction Set

All instructions and addressing modes can be used.

- Exception Vector Table and Memory Indirect Branch Addresses

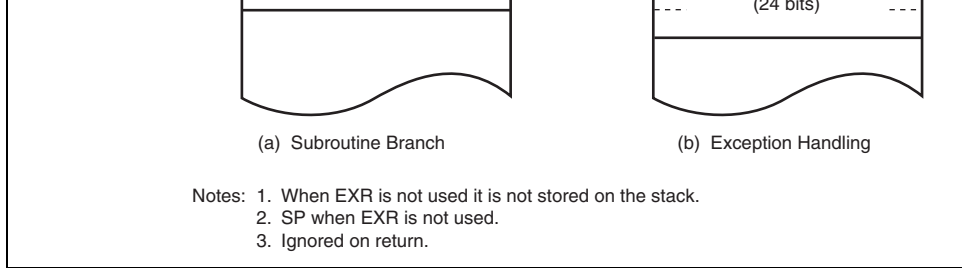
In advanced mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.



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

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing 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.



**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 table. One branch address is stored per 32 bits. The structure of the exception vector 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 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. EXR contents are saved or restored regardless of whether or not EXR is in use.

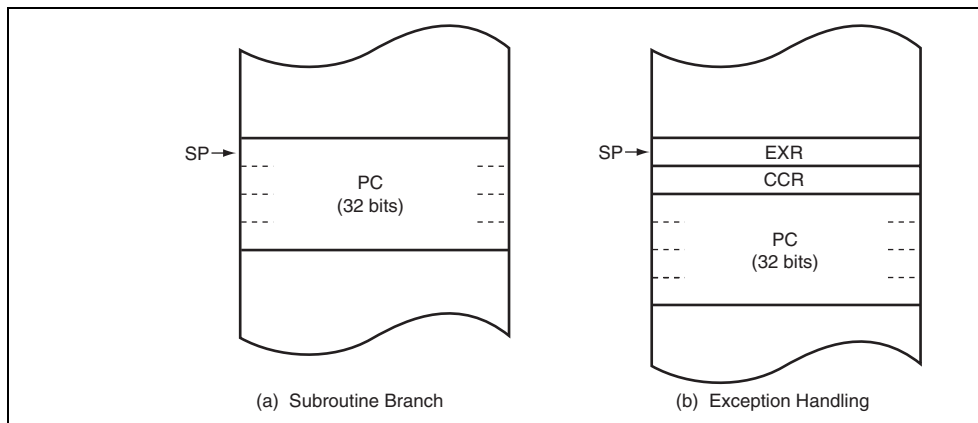
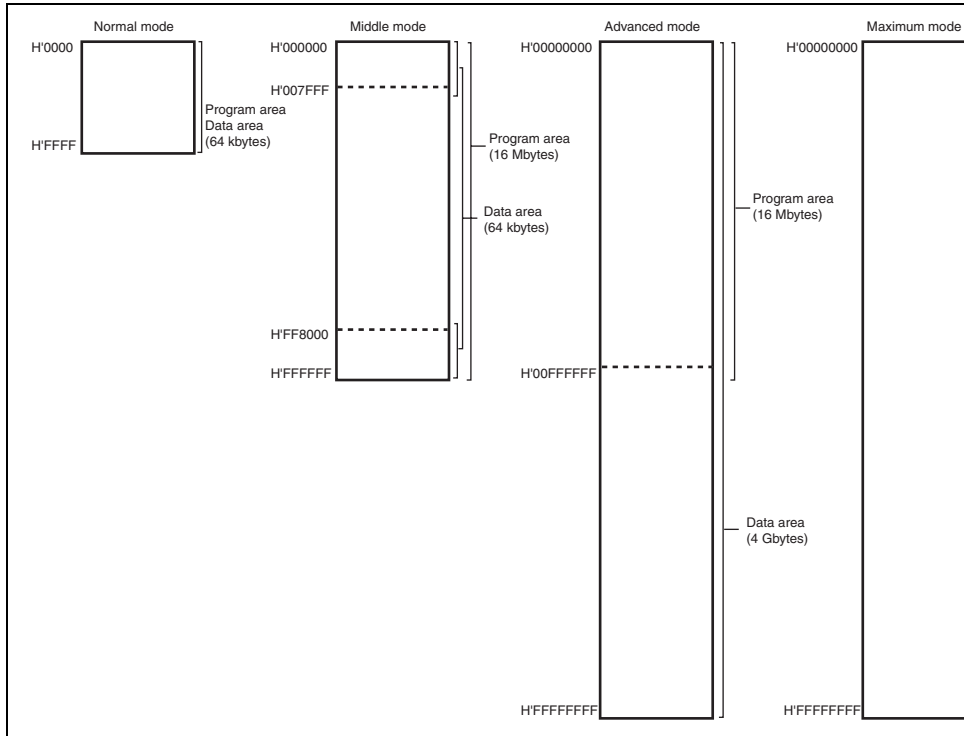


Figure 2.7 Stack Structure (Maximum Mode)



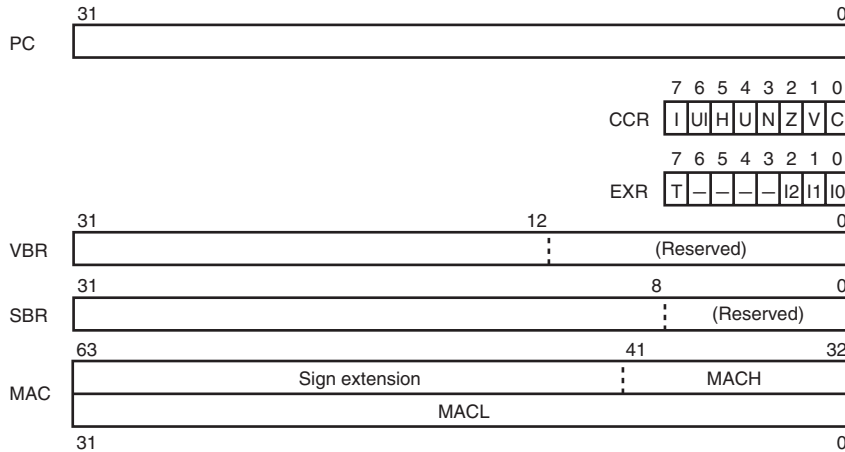
Figure 2.8 shows a memory map of the H8SX CPU. The address space differs depending on CPU operating mode.



**Figure 2.8 Memory Map**

ER0	E0	R0H	R0L
ER1	E1	R1H	R1L
ER2	E2	R2H	R2L
ER3	E3	R3H	R3L
ER4	E4	R4H	R4L
ER5	E5	R5H	R5L
ER6	E6	R6H	R6L
ER7 (SP)	E7	R7H	R7L

Control Registers



[Legend]

- |                                    |                                |                                   |
|------------------------------------|--------------------------------|-----------------------------------|
| SP: Stack pointer                  | U: User bit                    | T: Trace bit                      |
| PC: Program counter                | N: Negative flag               | I2 to I0: Interrupt mask bits     |
| CCR: Condition-code register       | Z: Zero flag                   | VBR: Vector base register         |
| I: Interrupt mask bit              | V: Overflow flag               | SBR: Short address base register  |
| UI: User bit or interrupt mask bit | C: Carry flag                  | MAC: Multiply-accumulate register |
| H: Half-carry flag                 | EXR: Extended control register |                                   |

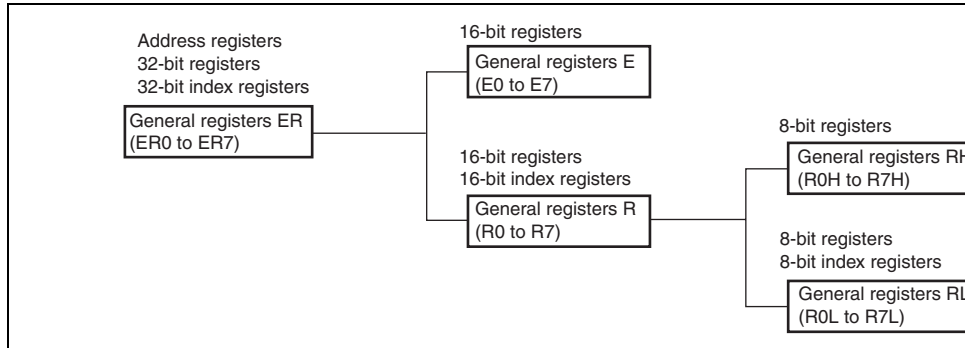
Figure 2.9 CPU Registers

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 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**



**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 a zero.)

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

- otherwise. A carry has the following types:
- Carry from the result of addition
  - Borrow from the result of subtraction
  - Carry from the result of shift or rotation
- The carry flag is also used as a bit accumulator for 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 XOR instructions.

For details, see section 6, 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 (I2, I1, I0).
0	I0	1	R/W	

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.

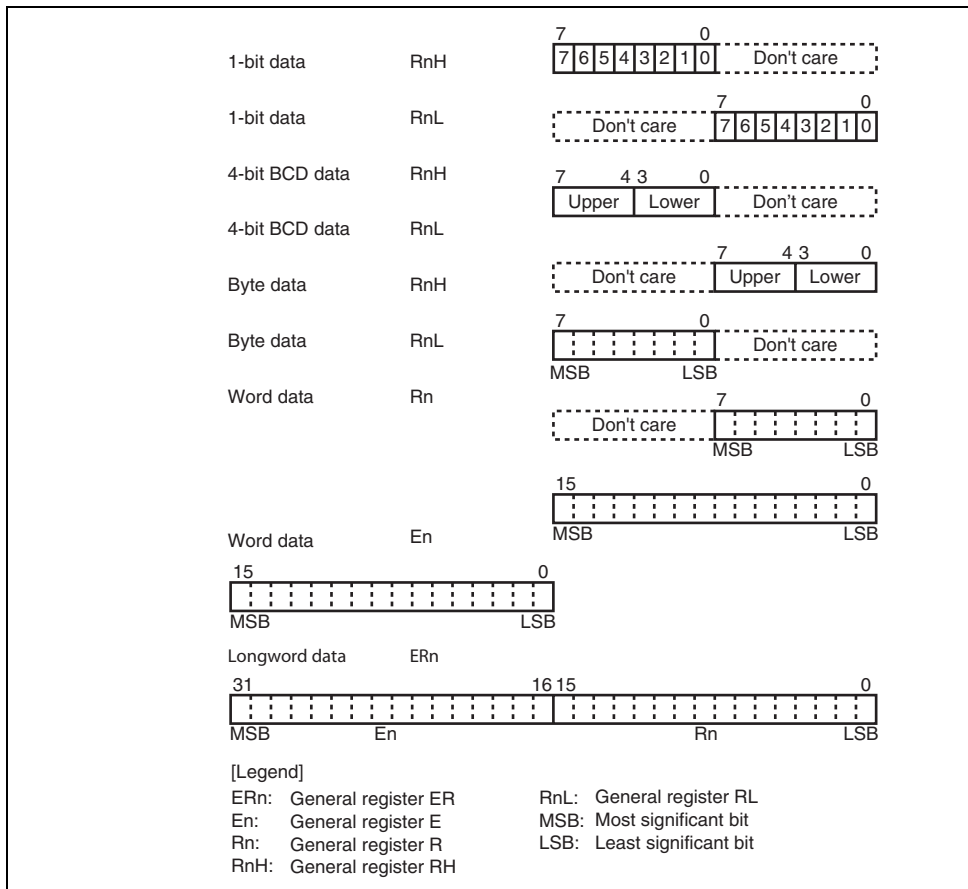
### **2.5.7 Multiply-Accumulate Register (MAC)**

MAC is a 64-bit register that stores the results of multiply-and-accumulate operations. It is composed of two 32-bit registers denoted MACH and MACL. The lower 10 bits of MACH are valid, and the upper bits are sign extended. The MAC contents are changed with the MAC, CLRMAC, 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 I bits in EXR to 0, and sets the I bits in CCR and EXR to 1. The general registers, MAC, and the stack pointer (ER7) 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.

Figure 2.12 shows the data formats in general registers.

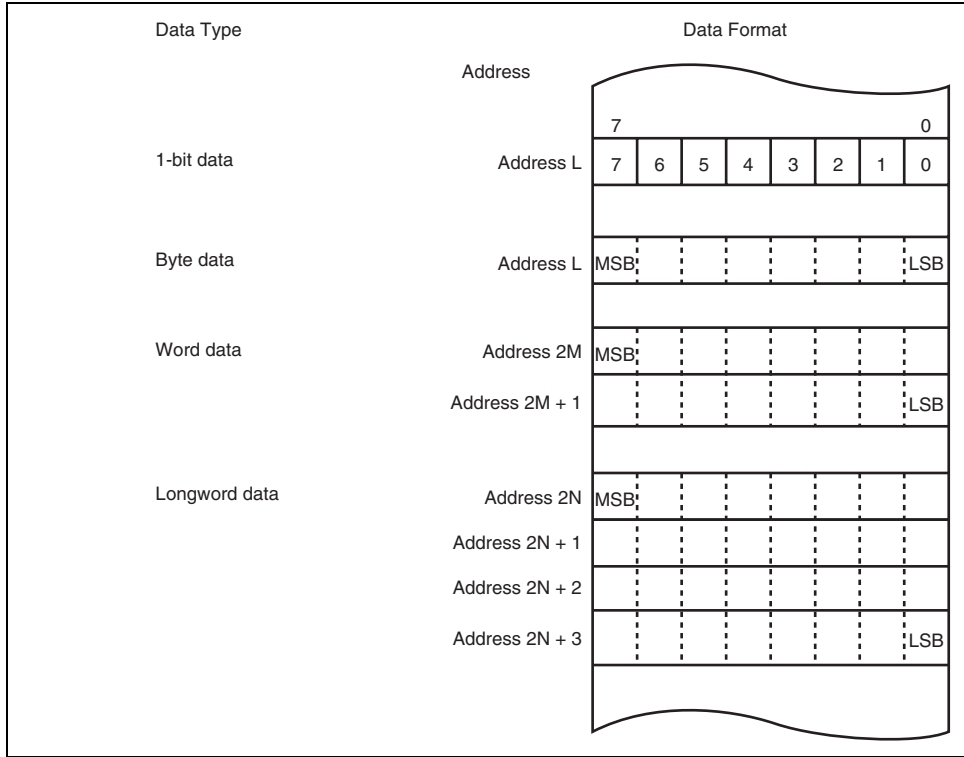


**Figure 2.12 General Register Data Formats**



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 byte size or longword size.



**Figure 2.13 Memory Data Formats**

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

[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 @-SP.  
POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L @-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. Only when the multiplier is available.

Data transfer	MOV	B/W/L	S	SD	SD	SD	SD	SD	SD
		B		S/D					S/D
	MOVFP, MOVTP	B		S/D					S/D*
	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
Block transfer	EEPMOV	B							
	MOVMD	B/W/L							
	MOVSD	B							
Arithmetic operations	ADD, CMP	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
	SUB	B	S		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					

	MAC* <sup>12</sup>	—							
	CLRMAC* <sup>12</sup>	—							
	LDMAC* <sup>12</sup>	—	S						
	STMAC* <sup>12</sup>	—	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
		W/L* <sup>5</sup>	D	D	D	D	D		D
		B/WL* <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 manipulation	BSET, BCLR, BNOT, BTST, BSET/cc, BCLR/cc	B	D	D				D	D
		BAND, BAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST, BSTZ, BISTZ	B	D	D				D

(VBR, SBR)						
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	—					

[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 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. Only when the multiplier is available.

	BRA	—		0	0				
	BRA/S	—		0*					
	JMP	—	0			0	0	0	0
	BSR	—		0					
	JSR	—	0			0	0	0	0
	RTS, RTS/L	—							
System control	TRAPA	—							
	RTE, RTE/L	—							

[Legend]

d: d:8 or d:16

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

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 (R8 to R7), and 32-bit registers (ER0 to ER7).



LDM	L	<p>@SP+ → Rn (register list)</p> <p>Restores the data from the stack to multiple general registers. Two or four general registers which have serial register numbers can be specified.</p>
STM	L	<p>Rn (register list) → @-SP</p> <p>Saves the contents of multiple general registers on the stack. Two or four general registers which have serial register numbers can be specified.</p>
MOVA	B/W	<p>EA → Rd</p> <p>Zero-extends and shifts the contents of a specified general register with memory data and adds them with a displacement. The result is stored in the specified general register.</p>

MOVMD.W	W	Transfers a data block. Transfers word data which begins at a memory location specified to a memory location specified by ER6. The number of word data 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 to a memory location specified by ER6. The number of byte data transferred is specified by R4. When zero data is detected during the transfer stops and execution branches to a specified address.

INC	B/W/L	$Rd \pm 1 \rightarrow Rd, Rd \pm 2 \rightarrow Rd$
DEC		Increments or decrements a general register by 1 or 2. (Byte operations can be incremented or decremented by 1 only.)
ADDS	L	$Rd \pm 1 \rightarrow Rd, Rd \pm 2 \rightarrow Rd, Rd \pm 4 \rightarrow Rd$
SUBS		Adds or subtracts the value 1, 2, or 4 to or from data in a general register.
DAA	B	$Rd$ (decimal adjust) $\rightarrow Rd$
DAS		Decimal-adjusts an addition or subtraction result in a general register 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: 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: 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: 8 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: 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: 8 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: 8 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 8 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.

		Compares data between immediate data, general registers, and and stores the result in CCR.
NEG	B/W/L	$0 - (\text{EAd}) \rightarrow (\text{EAd})$ Takes the two's complement (arithmetic complement) of data in a register or the contents of a memory location.
EXTU	W/L	$(\text{EAd}) (\text{zero extension}) \rightarrow (\text{EAd})$ 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, will be zero-extended.
EXTS	W/L	$(\text{EAd}) (\text{sign extension}) \rightarrow (\text{EAd})$ 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, will be sign-extended.
TAS	B	$@\text{ERd} - 0, 1 \rightarrow (<\text{bit } 7> \text{ of } @\text{EAd})$ Tests memory contents, and sets the most significant bit (bit 7) to 1 if the contents are not zero.
MAC*	—	$(\text{EAs}) \times (\text{EAd}) + \text{MAC} \rightarrow \text{MAC}$ Performs signed multiplication on memory contents and adds the result to the MAC.
CLRMAC*	—	$0 \rightarrow \text{MAC}$ Clears MAC to zero.
LDMAC*	—	$\text{Rs} \rightarrow \text{MAC}$ Loads data from a general register to MAC.
STMAC*	—	$\text{MAC} \rightarrow \text{Rd}$ Stores data from MAC to a general register.

Note: \* Only when the multiplier is available.

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) (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 1, 2, 4, 8, or 16 bits. The contents of a general register can be shifted 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) (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) (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) (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.

---

BCLR	B	$0 \rightarrow (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)$ 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, $0 \rightarrow (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)$ 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 condition status can be specified as a condition.
BNOT	B	$\sim (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle) \rightarrow (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)$ 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	$\sim (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle) \rightarrow Z$ 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	$C \wedge (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle) \rightarrow C$ 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	$C \wedge [\sim (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)] \rightarrow C$ 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	$C \vee (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle) \rightarrow C$ 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.

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	$(\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle) \rightarrow C$ 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	$\sim (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle) \rightarrow C$ 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	$C \rightarrow (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)$ 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	$Z \rightarrow (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)$ Transfers the zero 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.
BIST	B	$\sim C \rightarrow (\langle \text{bit-No.} \rangle \text{ of } \langle \text{EAd} \rangle)$ 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.

---

**Table 2.10 Branch Instructions**

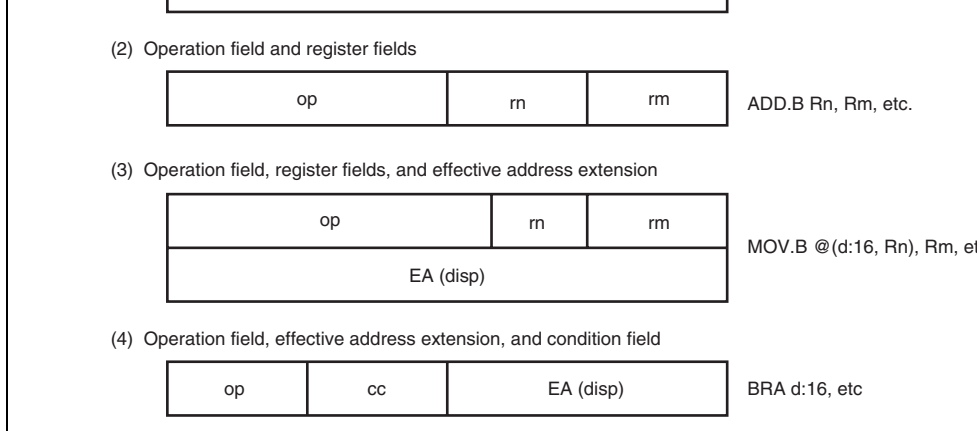
<b>Instruction</b>	<b>Size</b>	<b>Function</b>
BRA/BS BRA/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a specified address.
BSR/BS BSR/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, 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 current 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.



location to CCR or EXR.

Although CCR and EXR are 8-bit registers, word-size transfers performed between them and memory. The upper 8 bits are val

	L	$R_s \rightarrow VBR, R_s \rightarrow SBR$ Transfers the general register contents to VBR or SBR.
STC	B/W	$CCR \rightarrow (EAd), EXR \rightarrow (EAd)$ Transfers the contents of CCR or EXR to a general register or m Although CCR and EXR are 8-bit registers, word-size transfers performed between them and memory. The upper 8 bits are val
	L	$VBR \rightarrow Rd, SBR \rightarrow Rd$ Transfers the contents of VBR or SBR to a general register.
ANDC	B	$CCR \wedge \#IMM \rightarrow CCR, EXR \wedge \#IMM \rightarrow EXR$ Logically ANDs the CCR or EXR contents with immediate data.
ORC	B	$CCR \vee \#IMM \rightarrow CCR, EXR \vee \#IMM \rightarrow EXR$ Logically ORs the CCR or EXR contents with immediate data.
XORC	B	$CCR \oplus \#IMM \rightarrow CCR, EXR \oplus \#IMM \rightarrow EXR$ Logically exclusive-ORs the CCR or EXR contents with immedi
NOP	—	$PC + 2 \rightarrow PC$ Only increments the program counter.



**Figure 2.14 Instruction Formats**

- **Operation Field**  
Indicates the function of the instruction, and specifies the addressing mode and operation 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 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.

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:32, RnL.B)/@(d:32,Rn.W)/@(d:32,ERn)
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,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.

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. 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 sign-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.

The operand value is the contents of a memory location which is pointed to by the effective address. The 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 effective address. The 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 effective address. The operation result: 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. 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 multiple 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.

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	8 bits (@aa:8)	A consecutive 256-byte area (the upper address is set in SBR)		
	16 bits (@aa:16)	H'0000 to H'FFFF	H'000000 to H'007FFF, H'FF8000 to H'FFFFFF	H'00000000 to H'00007FFF, H'FFFFFF8000 to H'FFFFFF
	32 bits (@aa:32)		H'000000 to H'FFFFFF	H'00000000 to H'FFFFFF
Program area	24 bits (@aa:24)		H'000000 to H'FFFFFF	H'00000000 to H'00FFFFFF
	32 bits (@aa:32)			H'00000000 to H'00007FFF, H'00FFFFFF to H'FFFFFF

manipulation instructions contain 3-bit immediate data in the instruction code, for specifying a branch displacement. 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 operand 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).

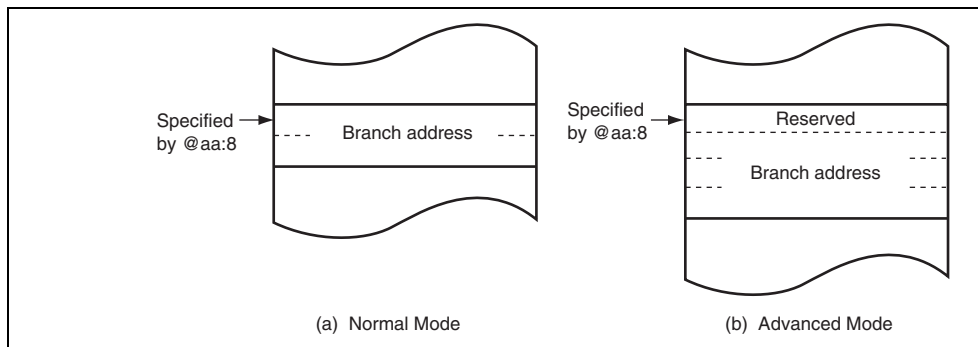
### **2.8.9 Program-Counter Relative with Index Register—@(RnL.B, PC), @(Rn.V, PC) or @(ERn.L, PC)**

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 operation result is the contents of an address register specified by the register field in the instruction code (RnL 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).

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 and the vector address of an exception handling other than a reset or a CPU address error can be specified by VBR.

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



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



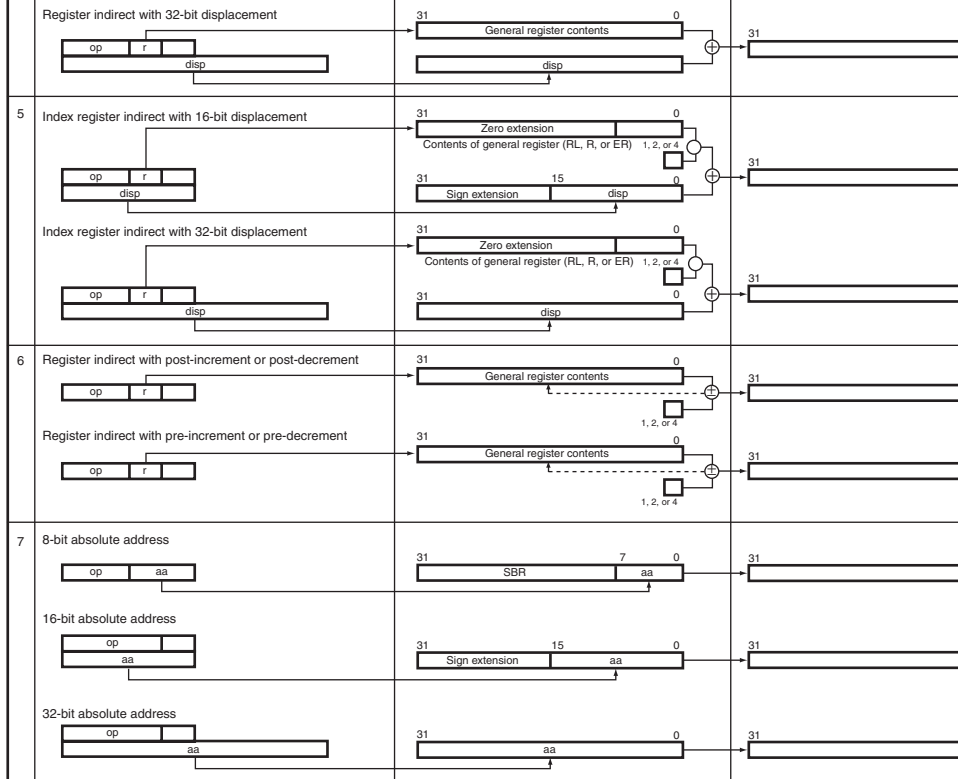
advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

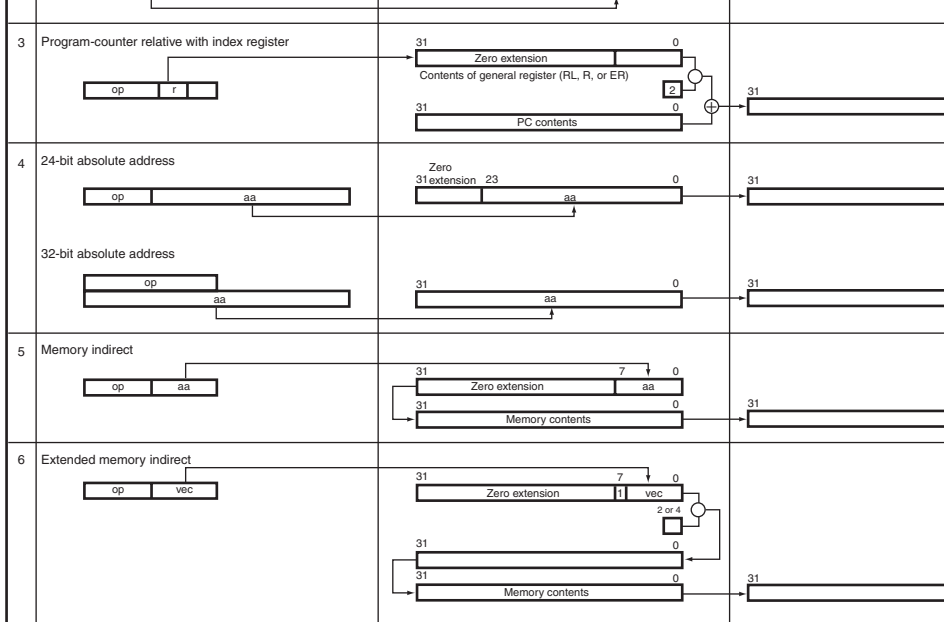
### 2.8.12 Effective Address Calculation

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 zero 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.





### 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.

The reset state can also be entered by a watchdog timer overflow when available.

- 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 table and branches to that address. For further details, see section 6, 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 further details, see section 28, Power-Down Modes.

Note: A transition to the reset state occurs whenever the STBY signal goes low.  
\* A transition to the reset state occurs when the RES signal goes low in all states except hardware standby mode. A transition can also be made to the reset state when the watchdog timer overflows.

## Figure 2.16 State Transitions



MCU Operating Mode	MD2	MD1	MD0	CPU Operating Mode	Address Space	LSI Initiation Mode	On-Chip ROM	Ext B Defa
1	0	0	1	Advanced mode	16 Mbytes	User boot mode	Enabled	—
2	0	1	0			Boot mode	Enabled	—
3	0	1	1			Boundary scan enabled single-chip mode	Enabled	—
4	1	0	0			On-chip ROM disabled extended mode	Disabled	16 bits
5	1	0	1			On-chip ROM enabled extended mode	Disabled	8 bits
6	1	1	0			On-chip ROM enabled extended mode	Enabled	8 bits
7	1	1	1			Single-chip mode	Enabled	—

**Table 3.2 SDRAM Interface Setting for each MCU Operating Mode**

MD3	SDRAM Interface
0	Disabled
1	Enabled

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, external extended mode, on-chip ROM initiation mode, or single-chip initiation mode can be selected.

as a data bus by specifying the data direction register (DDR) for each port. When the external address space is not in use, ports J and K can be used by setting the PCJKE bit in the port control register D (PFCRD) to 1.

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 a 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 width mode. If 8-bit address space is designated for all areas, it is called the 8-bit bus width mode.



MD3 to MD0 are latched. Latching is released by a reset.

Bit	15	14	13	12	11	10	9
Bit Name	MDS7	—	—	—	MDS3	MDS2	MDS1
Initial Value	Undefined*	1	0	1	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R

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

Note: \* Determined by pins MD3 to MD0.

Bit	Bit Name	Initial Value	R/W	Descriptions
15	MDS7	Undefined*	R	This pin indicates a value set with the mode (MD3) When MDCR is read, the signal levels input MD3 pin is latched into this bit. This latch is by a reset.
14	—	1	R	Reserved
13	—	0	R	These are read-only bits and cannot be mo
12	—	1	R	

4	—	1	R
3	—	Undefined*	R
2	—	Undefined*	R
1	—	Undefined*	R
0	—	Undefined*	R

Note: \* Determined by pins MD3 to MD0.

**Table 3.3 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
3	0	1	1	0	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

Bit Name	—	—	—	—	—	—	DTCMD
Initial Value	0	0	0	0	0	0	1
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 always be 1.
13	MACS	0	R/W	MAC Saturation Operation Control 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	Reserved This bit is always read as 1. The write value always be 1.
11	FETCHMD	0	R/W	Instruction Fetch Mode Select This LSI can prefetch an instruction in units of 16 or 32 bits. Select the bus width for instruction fetch depending on the used memory for the storage programs. 0: 32-bit mode 1: 16-bit mode

bit when PCJKE = 1.  
 When writing 0 to this bit after reading EXPE = 1, the 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 settings of the write data buffer function, refresh control function and EXDMAC bus right release state and other bits.  
 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 is 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. The initial value depends on the LSI initiation mode.  
 2. For details on the settings of the EXPE and PCJKE bits when the external address space is in use, see section 13.3.12, Port Function Control Register D (PFCRD).

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 25, Flash Memory.

### **3.3.3 Mode 3**

This is the boundary scan function enabled single-chip activation mode. The operation is the same as mode 7 except for the boundary scan function. For details on the boundary scan function, see section 26, Boundary Scan.

### **3.3.4 Mode 4**

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and the 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 ports A and B 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.

### 3.3.6 Mode 6

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

The initial bus width mode immediately after a reset is eight bits, with 8-bit access to all I/O ports. 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 13, I/O Ports. Port H functions as a data bus, and parts of ports A and B function as bus control signals. However, if any port is designated as a 16-bit access space by the bus controller, the bus width mode switches to 16-bit and ports H and I function as a data bus.

### 3.3.7 Mode 7

The CPU operating mode is advanced mode in which the address space is 16 Mbytes, and 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) enables the external address space. After the external address space is enabled, ports D, E, and F can be used as an address output bus and ports H and I as a data bus by specifying the data direction register (DDR) for each port. When the external address space is not in use, port K can be used by setting the PCJKE bit in the port function control register D (PFCRD) to 1. For details, see section 13, I/O Ports.

3	P*/C	P*/C	P*/C	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	P*/D
4	P/C*	P/C*	P*/C	P*/C	P/C*	P*/C	A	A	A	P*/A	D
5	P/C*	P/C*	P*/C	P*/C	P/C*	P*/C	A	A	A	P*/A	D
6	P/C*	P/C*	P*/C	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	D
7	P*/C	P*/C	P*/C	P*/C	P*/C	P*/C	P*/A	P*/A	P*/A	P*/A	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

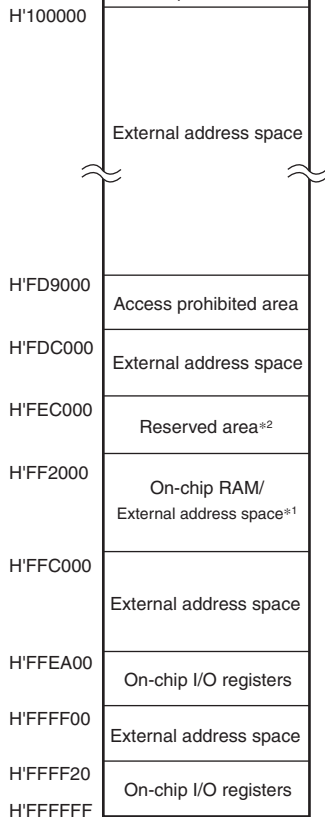
Figures 3.1 and 3.2 show the address map in each operating mode.

	External address space/ reserved area*1*3	External address space/ reserved area*1*3	External address space/ reserved area*1*3
H'FD9000	Access prohibited area	H'FD9000	Access prohibited area
H'FDC000	External address space/ reserved area*1*3	H'FDC000	External address space/ reserved area*1*3
H'FEC000	Reserved area*3	H'FEC000	Reserved area*3
H'FF2000	On-chip RAM*2	H'FF2000	On-chip RAM/ External address space*4
H'FFC000	External address space/ reserved area*1*3	H'FFC000	External address space/ reserved area*1*3
H'FFEA00	On-chip I/O registers	H'FFEA00	On-chip I/O registers
H'FFFF00	External address space/ reserved area*1*3	H'FFFF00	External address space/ reserved area*1*3
H'FFFF20	On-chip I/O registers	H'FFFF20	On-chip I/O registers
H'FFFFFF		H'FFFFFF	

Notes:1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.  
2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit in SYSCR to 0.  
3. Do not access the reserved areas.  
4. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.

**Figure 3.1 Address Map in Each Operating Mode of H8SX/1665 and H8SX/1668**





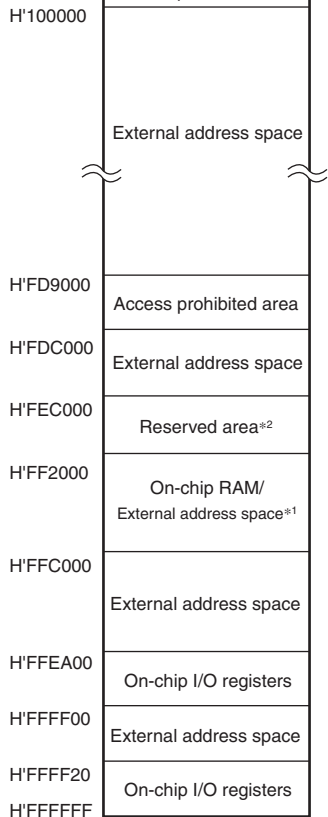
- Notes: 1. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.  
 2. Do not access the reserved area.

**Figure 3.1 Address Map in Each Operating Mode of H8SX/1665 and H8SX/1668**

	External address space/ reserved area*1*3		External address space/ reserved area*1*3		External address space/ reserved area*1*3
H'FD9000	Access prohibited area	H'FD9000	Access prohibited area	H'FD9000	Access prohibited area
H'FDC000	External address space/ reserved area*1*3	H'FDC000	External address space/ reserved area*1*3	H'FDC000	External address space/ reserved area*1*3
H'FEC000	Reserved area*3	H'FEC000	Reserved area*3	H'FEC000	Reserved area*3
H'FF2000	On-chip RAM*2	H'FF2000	On-chip RAM/ External address space*4	H'FF2000	On-chip RAM/ External address space*4
H'FFC000	External address space/ reserved area*1*3	H'FFC000	External address space/ reserved area*1*3	H'FFC000	External address space/ reserved area*1*3
H'FFEA00	On-chip I/O registers	H'FFEA00	On-chip I/O registers	H'FFEA00	On-chip I/O registers
H'FFFF00	External address space/ reserved area*1*3	H'FFFF00	External address space/ reserved area*1*3	H'FFFF00	External address space/ reserved area*1*3
H'FFFF20	On-chip I/O registers	H'FFFF20	On-chip I/O registers	H'FFFF20	On-chip I/O registers
H'FFFFFF		H'FFFFFF		H'FFFFFF	

Notes:1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.  
2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit in SYSCR to 0.  
3. Do not access the reserved areas.  
4. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.

**Figure 3.2 Address Map in Each Operating Mode of H8SX/1662 and H8SX/1661**



Notes: 1. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.  
 2. Do not access the reserved area.

**Figure 3.2 Address Map in Each Operating Mode of H8SX/1662 and H8SX/1661**

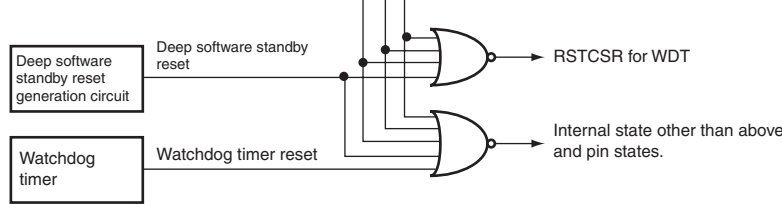


when using power-on reset\* and voltage monitoring reset\*, RES pin must be fixed high.

**Table 4.1 Reset Names And Sources**

<b>Reset Name</b>	<b>Source</b>
Pin reset	Voltage input to the $\overline{\text{RES}}$ pin is driven
Power-on reset*	Vcc rises or lowers
Voltage-monitoring reset*	Vcc falls (voltage detection: Vdet)
Deep software standby reset	Deep software standby mode is canceled by interrupt.
Watchdog timer reset	The watchdog timer overflows.

Note: \* Supported only by the H8SX/1665M Group.



Note: \* Supported only by the H8SX/1665M Group.

**Figure 4.1 Block Diagram of Reset Circuit**

When a reset is canceled, the reset exception handling is started. For the reset exception see section 6.3, Reset.

## 4.2 Input/Output Pin

Table 4.2 shows the pin related to resets.

**Table 4.2 Pin Configuration**

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

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

- Notes:
1. Only 0 can be written to clear the flag.
  2. Supported only by the H8SX/1665M Group.
  3. Initial value is undefined in the H8SX/1665M Group.
  4. Only 0 can be written to clear the flag in the H8SX/1665M Group.
  5. Only read is possible in the H8SX/1665M Group.

Bit	Bit Name	Initial Value	R/W	Description
7	DPSRSTF	0	R/(W)*1	<p>Deep Software Standby Reset Flag</p> <p>Indicates that deep software standby mode is canceled by an interrupt source specified with DPSIER or by an interrupt source specified with DPSIEGR and an internal reset is generated.</p> <p>[Setting condition]</p> <p>When deep software standby mode is canceled by an interrupt source.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When this bit is read as 1 and then written 0.</li> <li>• When a pin reset, power-on reset*2 or voltage monitoring reset*2 is generated.</li> </ul>
6 to 3	—	All 0	R/W	<p>Reserved</p> <p>These bits are always read as 0. The write value is always be 0.</p>



Bit	Bit Name	Value	R/W	Description
2	LVDF	Undefined	R/(W)* <sup>1</sup>	<p>LVD Flag</p> <p>This bit indicates that the voltage detection circuit has detected a low voltage (<math>V_{cc}</math> at or below <math>V_{det}</math>).</p> <p>[Setting condition]</p> <p><math>V_{cc}</math> falling to or below <math>V_{det}</math></p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>After <math>V_{cc}</math> has exceeded <math>V_{det}</math> and the specified stabilization period has elapsed, writing 0 to this bit after reading it as 1.</li> <li>Generation of a pin reset or power-on reset</li> </ul>
1	—	Undefined	R/W	<p>Reserved</p> <p>The write value should always be 0.</p>
0	PORF	Undefined	R	<p>Power-on Reset Flag</p> <p>This bit indicates that a power-on reset has been generated.</p> <p>[Setting condition]</p> <p>Generation of a power-on reset</p> <p>[Clearing condition]</p> <p>Generation of a pin reset</p>

- Notes:
- Only 0 can be written to clear the flag.
  - Supported only by the H8SX/1665M Group.

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 clear the flag.</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>
6	RSTE	0	R/W	<p>Reset Enable</p> <p>Selects whether or not the LSI internal state is reset when TCNT overflows in watchdog timer mode.</p> <p>0: Internal state is not reset when TCNT overflows (this LSI internal state is not reset, TCNT and TCNTWDT 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 performed by this bit.</p>
4 to 0	—	1	R	<p>Reserved</p> <p>These are read-only bits but cannot be modified.</p>

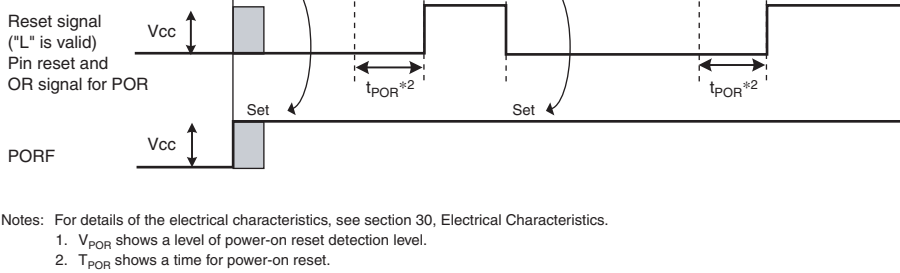
Note: \* Only 0 can be written to clear the flag.

This is an internal reset generated by the power-on reset circuit.

If  $\overline{\text{RES}}$  is in the high-level state when power is supplied, a power-on reset is generated. When  $\overline{\text{RES}}$  has exceeded  $V_{\text{por}}$  and the specified period (power-on reset time) has elapsed, the chip is released from the power-on reset state. The power-on reset time is a period for stabilization of the power supply and the LSI circuit.

If  $\overline{\text{RES}}$  is at the high-level when the power-supply voltage ( $V_{\text{cc}}$ ) falls to or below  $V_{\text{por}}$ , a power-on reset is generated. The chip is released after  $V_{\text{cc}}$  has risen above  $V_{\text{por}}$  and the power-on reset time has elapsed.

After a power-on reset has been generated, the PORF bit in RSTSR is set to 1. The PORF bit is a read-only register and is only initialized by a pin reset. Figure 4.2 shows the operation of the power-on reset.



**Figure 4.2 Operation of a Power on Reset**

## 4.6 Power Supply Monitoring Reset (H8SX/1665M Group)

This is an internal reset generated by the power-supply detection circuit.

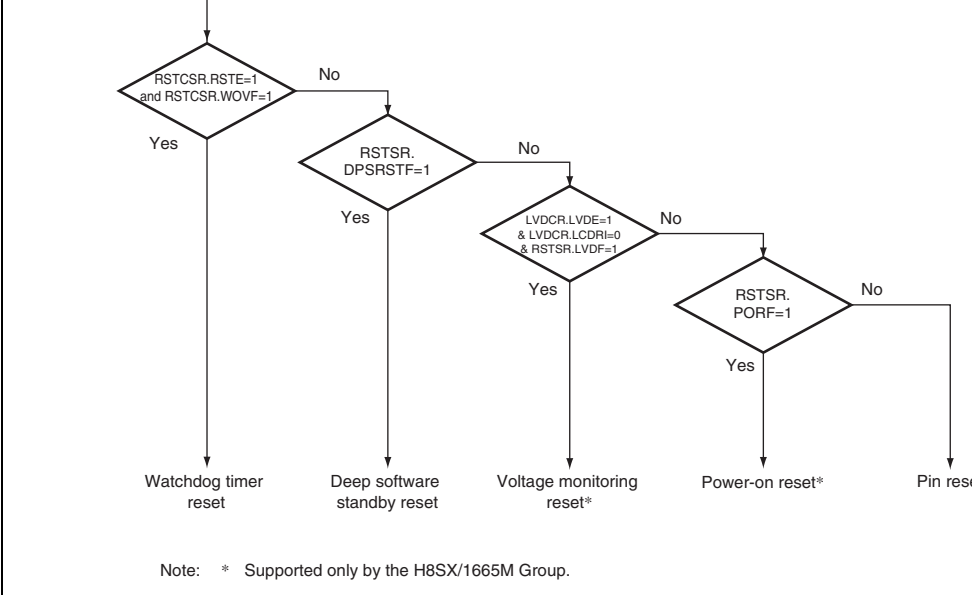
When Vcc falls below Vdet in the state where the LVDE bit in LVDCR has been set to 1 and the LVDRI bit has been cleared to 0, a voltage-monitoring reset is generated. When Vcc subsequently rises above Vdet, release from the voltage-monitoring reset proceeds after a specified time has elapsed.

For details of the voltage-monitoring reset, see section 5, Voltage Detection Circuit (LVD), section 30, Electrical Characteristics.

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 TC overflow. After a certain time, the watchdog timer reset is canceled.

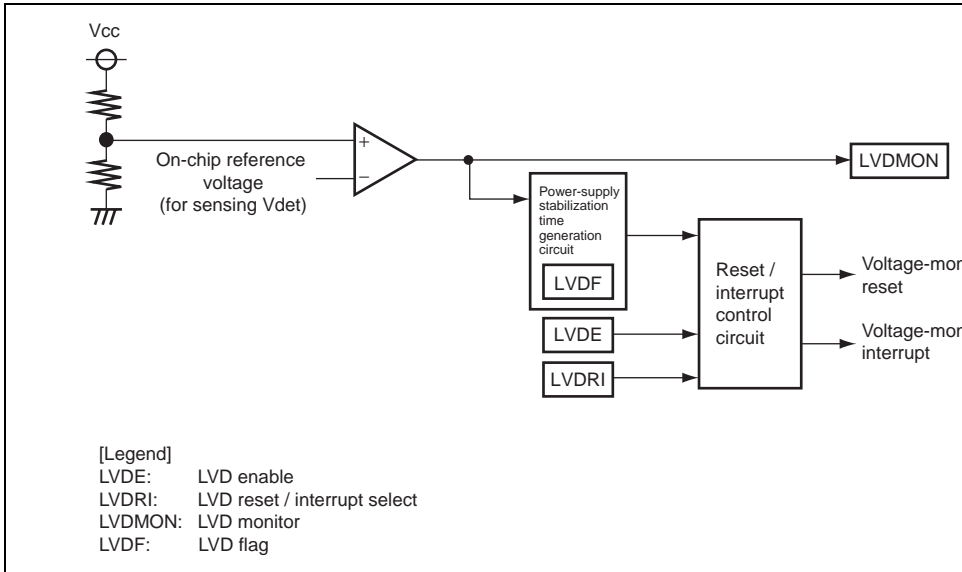
For details of the watchdog timer reset, see section 18, Watchdog Timer (WDT).



**Figure 4.3 Example of Reset Generation Source Determination Flow**

Capable of detecting the power-supply voltage (Vcc) becoming less than or equal to  
Capable of generating an internal reset or interrupt when a low voltage is detected.

A block diagram of the voltage detection circuit is shown in figure 5.1.



**Figure 5.1 Block Diagram of Voltage-Detection Circuit**

LVDE, LVDRI, and LVDMON are initialized by a pin reset or power-on reset

Bit	7	6	5	4	3	2	1
Bit name	LVDE	LVDRI	—	LVDMON	—	—	—
Initial value:	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	LVDE	0	R/W	<p>LVD Enable</p> <p>This bit enables or disables issuing of a reset interrupt by the voltage-detection circuit.</p> <p>0: Disabled 1: Enabled</p>
6	LVDRI	0	R/W	<p>LVD Reset/Interrupt Select</p> <p>This bit selects whether an internal reset interrupt is generated when the voltage detection circuit detects a low voltage. When modifying the LVDRI bit, ensure that low-voltage detection is in the disabled state (the LVDE bit is cleared).</p> <p>0: A reset is generated when a voltage is detected. 1: An interrupt is generated when a low voltage is detected.</p>
5	—	0	R/W	<p>Reserved</p> <p>This bit is always read as 0 and the write data should always be 0.</p>



## 5.2.2 Reset Status Register (RSTSR)

RSTSR indicates the source of an internal reset or voltage monitoring interrupt.

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

Note: \* To clear the flag, only 0 should be written to.

Bit	Bit Name	Initial Value	R/W	Description
7	DPSRSTF	0	R/W*	<p>Deep Software Standby Reset Flag</p> <p>This bit indicates release from deep software standby mode due to the interrupt source selected by DPSIER and DPSIEGR, and the generation of an internal reset.</p> <p>[Setting condition]</p> <p>Release from deep software standby mode due to an interrupt source.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>• Writing 0 to the bit after reading it as 1.</li> <li>• Generation of a pin reset, power-on reset, or voltage monitoring reset.</li> </ul>

- [Clearing condition]
- After Vcc has exceeded Vdet and the specified stabilization period has elapsed, writing 0 to the bit after reading it as 1
- Generation of a pin reset or power-on

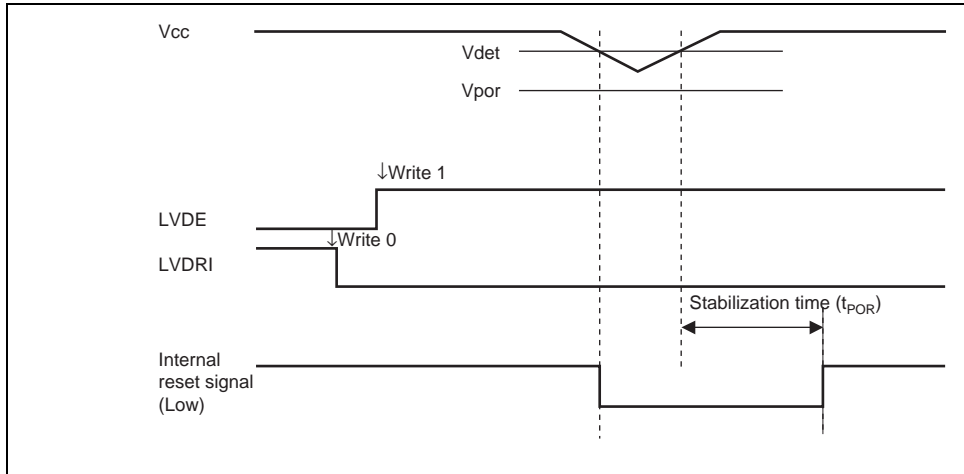
1	—	Undefined	R/W	Reserved The write value should always be 0.
0	PORF	Undefined	R	Power-on Reset Flag This bit indicates that a power-on reset has been generated. [Setting condition] Generation of a power-on reset [Clearing condition] Generation of a pin reset

Note: \* To clear the flag, only 0 should be written to.

Reset, after  $V_{CC}$  has risen above  $V_{det}$  and release from the voltage monitoring reset takes place. The period for stabilization ( $t_{por}$ ) has elapsed. The period for stabilization ( $t_{por}$ ) is a time that is set by the voltage detection circuit in order to stabilize the  $V_{CC}$  and the internal circuit of the device.

When a voltage-monitoring reset is generated, the LVDF bit is set to 1.

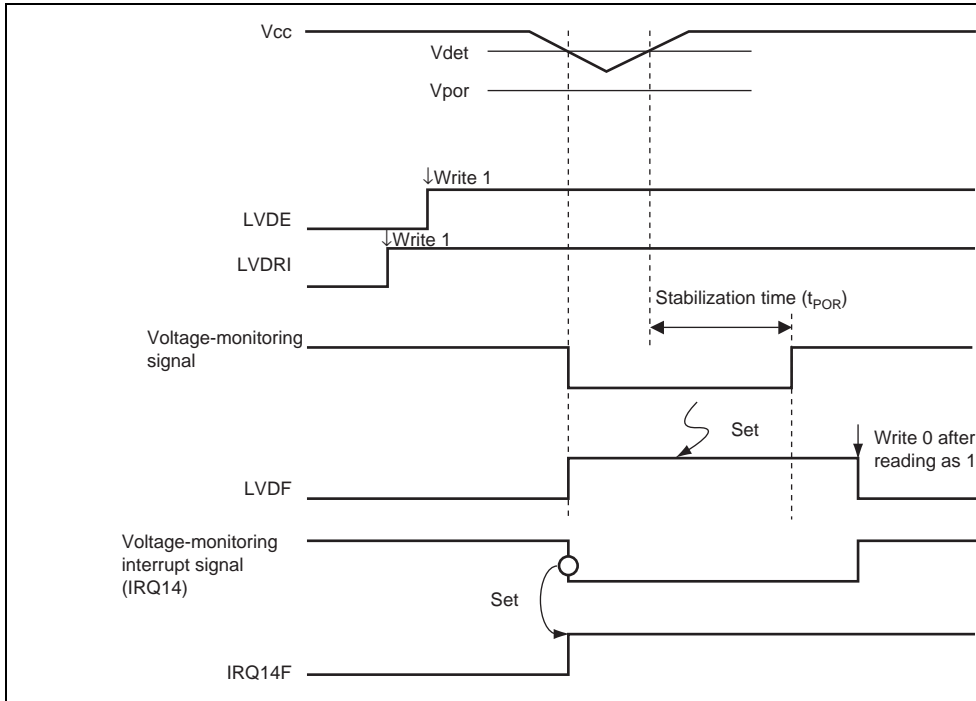
For details, see section 30, Electrical Characteristics.



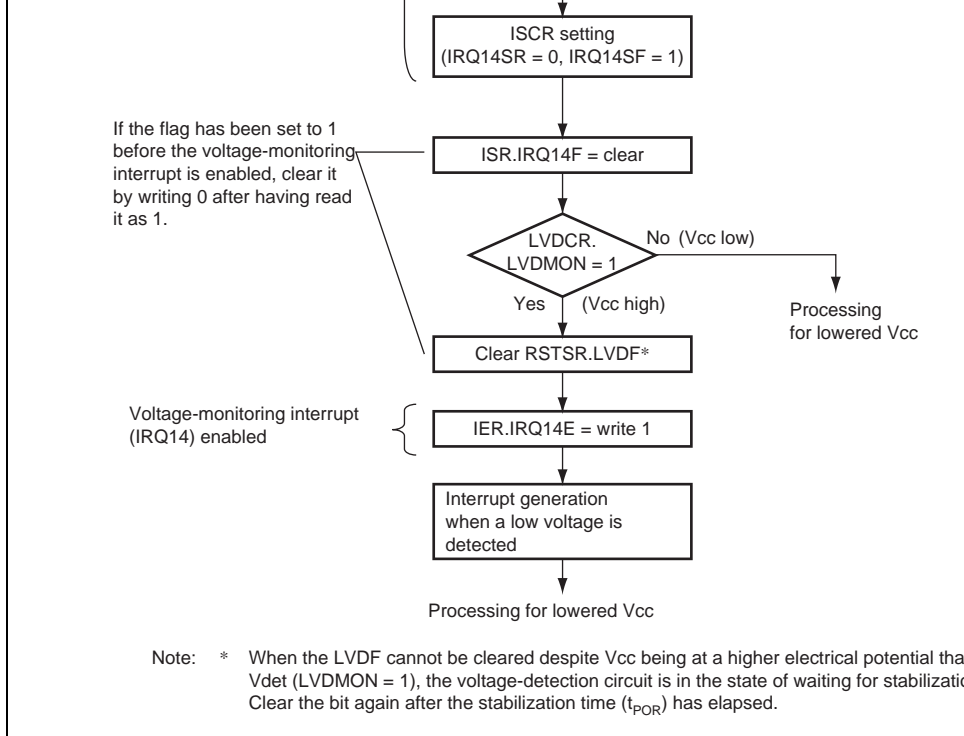
**Figure 5.2 Timing of the Voltage-Monitoring Reset**

the IRQ14SR and IRQ14SF bits in the ISCR to 01 (interrupt request on falling edge).

Figure 5.4 shows the procedure for setting the voltage-monitoring interrupt.



**Figure 5.3 Timing of the Voltage-Monitoring Interrupt**




**Figure 5.4 Example of the Procedure for Setting the Voltage-Monitoring Interrupt**

The result of voltage detection by the voltage-detection circuit can be monitored by checking the value of the LVDMON bit in LVDCR. When the LVDMON bit has been enabled by setting the LVDE bit, 0 indicates that Vcc is at or below Vdet and 1 indicates that Vcc is above Vdet. The bit should be read while the voltage-monitoring reset has been disabled by setting the LVDR to 1.

Before clearing the LVDF bit in RSTSR to 0, confirm that the LVDMON bit is set to 1 (indicating that Vcc is above Vdet). When it is impossible to clear the LVDF bit despite the LVDMON bit being 1, the voltage-detection circuit is in the state of waiting for stabilization. In such a case, read the bit again after the stabilization time ( $t_{por}$ ) has elapsed.

**Table 6.1 Exception Types and Priority**

Priority	Exception Type	Exception Handling Start Timing
High  Low	Reset	Exception handling starts at the timing of level change from low to high on the $\overline{\text{RES}}$ pin, when deep software sleep mode is canceled, or when the watchdog timer overflows. The CPU enters the reset state when the $\overline{\text{RES}}$ pin is pulled up.
	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, if the trace (T) bit in the SBYCR is set to 1.
	Address error	After an address error has occurred, exception handling starts on completion of instruction execution.
	Interrupt	Exception handling starts after execution of the current instruction or exception handling, if an interrupt request has occurred.* <sup>2</sup>
	Sleep instruction	Exception handling starts by execution of a sleep instruction (SLEEP), if the SSBY bit in SBYCR is set to 0 and the SLPIE bit in SBYCR is set to 1.
	Trap instruction* <sup>3</sup>	Exception handling starts by execution of a trap instruction (TRAPA).

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

Exception Source	Vector Number	Vector Table Address Offsets	
		Normal Mode* <sup>2</sup>	Advanced, Maximum* <sup>2</sup>
Reset	0	H'0000 to H'0001	H'0000 to H'0001
Reserved for system use	1	H'0002 to H'0003	H'0004 to H'0005
	2	H'0004 to H'0005	H'0008 to H'0009
	3	H'0006 to H'0007	H'000C to H'000D
Illegal instruction	4	H'0008 to H'0009	H'0010 to H'0011
Trace	5	H'000A to H'000B	H'0014 to H'0015
Reserved for system use	6	H'000C to H'000D	H'0018 to H'0019
Interrupt (NMI)	7	H'000E to H'000F	H'001C to H'001D
Trap instruction	(#0)	8	H'0010 to H'0011
	(#1)	9	H'0012 to H'0013
	(#2)	10	H'0014 to H'0015
	(#3)	11	H'0016 to H'0017
CPU address error	12	H'0018 to H'0019	H'0030 to H'0031
DMA address error* <sup>3</sup>	13	H'001A to H'001B	H'0034 to H'0035
UBC break interrupt	14	H'001C to H'001D	H'0038 to H'0039
Reserved for system use	15	H'001E to H'001F	H'003C to H'003D
	17	H'0022 to H'0023	H'0044 to H'0045
Sleep interrupt	18	H'0024 to H'0025	H'0048 to H'0049



	IRQ2	66	H'0084 to H'0085	H'0108 to H'0109
	IRQ3	67	H'0086 to H'0087	H'010C to H'010D
	IRQ4	68	H'0088 to H'0089	H'0110 to H'0111
	IRQ5	69	H'008A to H'008B	H'0114 to H'0115
	IRQ6	70	H'008C to H'008D	H'0118 to H'0119
	IRQ7	71	H'008E to H'008F	H'011C to H'011D
	IRQ8	72	H'0090 to H'0091	H'0120 to H'0121
	IRQ9	73	H'0092 to H'0093	H'0124 to H'0125
	IRQ10	74	H'0094 to H'0095	H'0128 to H'0129
	IRQ11	75	H'0096 to H'0097	H'012C to H'012D
Reserved for system use		76	H'0098 to H'0099	H'0130 to H'0131
		79	H'009E to H'009F	H'013C to H'013D
Internal interrupt* <sup>4</sup>		80	H'00A0 to H'00A1	H'0140 to H'0141
		255	H'01FE to H'01FF	H'03FC to H'03FD

- Notes:
1. Lower 16 bits of the address.
  2. Not available in this LSI.
  3. A DMA address error is generated by the DTC, DMAC, and EXDMAC.
  4. For details of internal interrupt vectors, see section 7.5, Interrupt Exception Handling Vector Table.

A reset has priority over any other exception. When the  $\overline{\text{RES}}$  pin goes low, all processing of this LSI enters the reset state. To ensure that this LSI is reset, hold the  $\overline{\text{RES}}$  pin low for at least 20 states 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 states.

The chip can be reset by the overflow that is generated in watchdog timer mode of the watchdog timer. For details, see section 18, Watchdog Timer (WDT).

The chip can also be reset by the exit from deep software standby mode. For details, see section 28, 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.

### 6.3.1 Reset Exception Handling

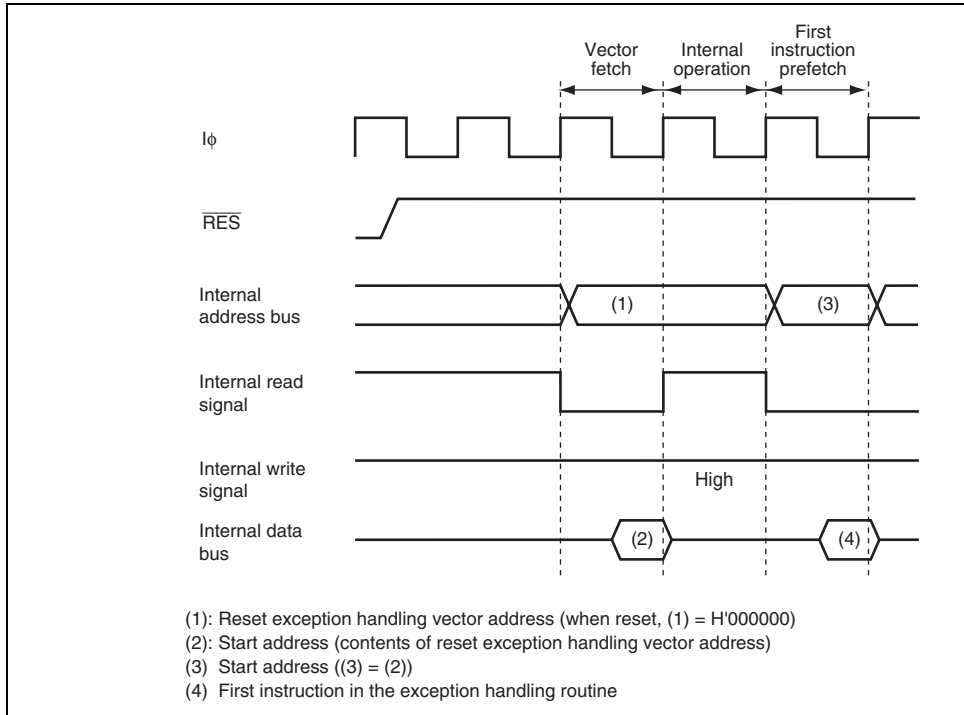
When the  $\overline{\text{RES}}$  pin goes high after being held low for the necessary time, this LSI starts reset exception handling as follows:

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 bit is 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.

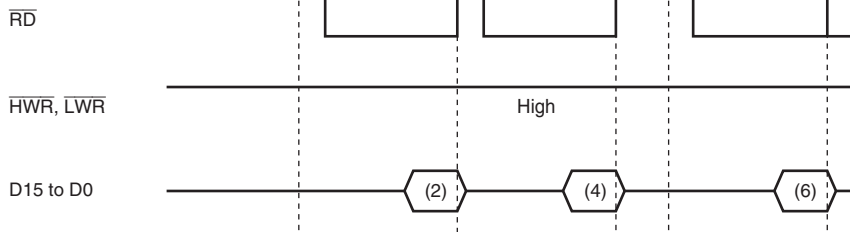
Figures 6.1 and 6.2 show examples of the reset sequence.

After the reset state is released, MSTPCRA and MSTPCRB are initialized to H'0FFF and H'0000, respectively, and all modules except the DTC, DMAC, and EXDMAC enter the module stop state.

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



**Figure 6.1 Reset Sequence (On-chip ROM Enabled Advanced Mode)**



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

Note: \* Seven program wait cycles are inserted.

**Figure 6.2 Reset Sequence  
 (16-Bit External Access in On-chip ROM Disabled Advanced Mode)**

handling routine by the RTE instruction, trace mode resumes. Trace exception handling carried out after execution of the RTE instruction.

Interrupts are accepted even within the trace exception handling routine.

**Table 6.4 Status of CCR and EXR after Trace Exception Handling**

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

[Legend]

- 1: Set to 1
- 0: Cleared to 0
- : Retains the previous value.

Type	Bus master	Description	Access
Instruction fetch	CPU	Fetches instructions from even addresses	No (no)
		Fetches instructions from odd addresses	Occurs
		Fetches instructions from areas other than on-chip peripheral module space* <sup>1</sup>	No (no)
		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 (no)
		Accesses stack when the stack pointer value is odd address	Occurs
Data read/write	CPU	Accesses word data from even addresses	No (no)
		Accesses word data from odd addresses	No (no)
		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 (no)
		Accesses word data from odd addresses	No (no)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to access prohibited area* <sup>2</sup>	Occurs

Single address transfer	DMAC or EXDMAC	Address access space is the external memory space for single address transfer	No (n
		Address access space is not the external memory space for single address transfer	Occu

- Notes:
1. For on-chip peripheral module space, see section 9, Bus Controller (BSC).
  2. For the access-prohibited area, refer to figure 3.1 in section 3.4, Address Map

### 6.5.2 Address Error Exception Handling

When an address error occurs, address error exception handling starts after the bus cycle 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 stack overflow during 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.

Table 6.6 shows the state of CCR and EXR after execution of the address error exception handling.

**Table 6.6 Status of CCR and EXR after Address Error Exception Handling**

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

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.



UBC break interrupt	UBC break controller (UBC)	1
IRQ0 to IRQ11	Pins IRQ0 to IRQ11 (external input)	12
Voltage-detection circuit*	Voltage-detection circuit (LVD)*	1
On-chip peripheral module	32K timer (TM32K)	1
	DMA controller (DMAC)	8
	EXDMA controller (EXDMAC)	8
	Watchdog timer (WDT)	1
	A/D converter	2
	16-bit timer pulse unit (TPU)	52
	8-bit timer (TMR)	16
	Serial communications interface (SCI)	24
	I <sup>2</sup> C bus interface 2 (IIC2)	2
	USB function module (USB)	5

Note: \* Supported only by the H8SX/1665M Group.

Different vector numbers and vector table offsets are assigned to different interrupt sources. For vector number and vector table offset, refer to table 7.2, Interrupt Sources, Vector Address, and Vector Table Offsets, and Interrupt Priority in section 7, Interrupt Controller.

### 6.6.2 Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to eight priority/mask levels to support multiple-interrupt control. The source to start interrupt exception handling and the vector number differ depending on the product. For details, refer to section 7, Interrupt Controller.

### 6.7.1 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 read 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 specified in the instruction code.

Table 6.8 shows the state of CCR and EXR after execution of trap instruction exception handling.

**Table 6.8 Status of CCR and EXR after Trap 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.

the SLEEP instruction is generated, the start address of the exception service routine from the vector table to PC, and program execution starts from that address.

Bus masters other than the CPU may gain the bus mastership after a sleep instruction has been executed. In such cases the sleep instruction will be started when the transactions of a bus master other than the CPU has been completed and the CPU has gained the bus mastership.

Table 6.9 shows the state of CCR and EXR after execution of sleep instruction exception handling. For details, see section 28.10, Sleep Instruction Exception Handling.

**Table 6.9 Status of CCR and EXR after Sleep Instruction Exception Handling**

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

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.

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, the PC is updated with that address, and program execution starts from that address.

Table 6.10 shows the state of CCR and EXR after execution of illegal instruction exception handling.

**Table 6.10 Status of CCR and EXR after Illegal Instruction Exception Handling**

Interrupt Control Mode	CCR		EXR	
	I	UI	T	I2 to I1
0	1	—	—	—
2	1	—	0	—

[Legend]

1: Set to 1

0: Cleared to 0

—: Retains the previous value.



Interrupt control mode 0



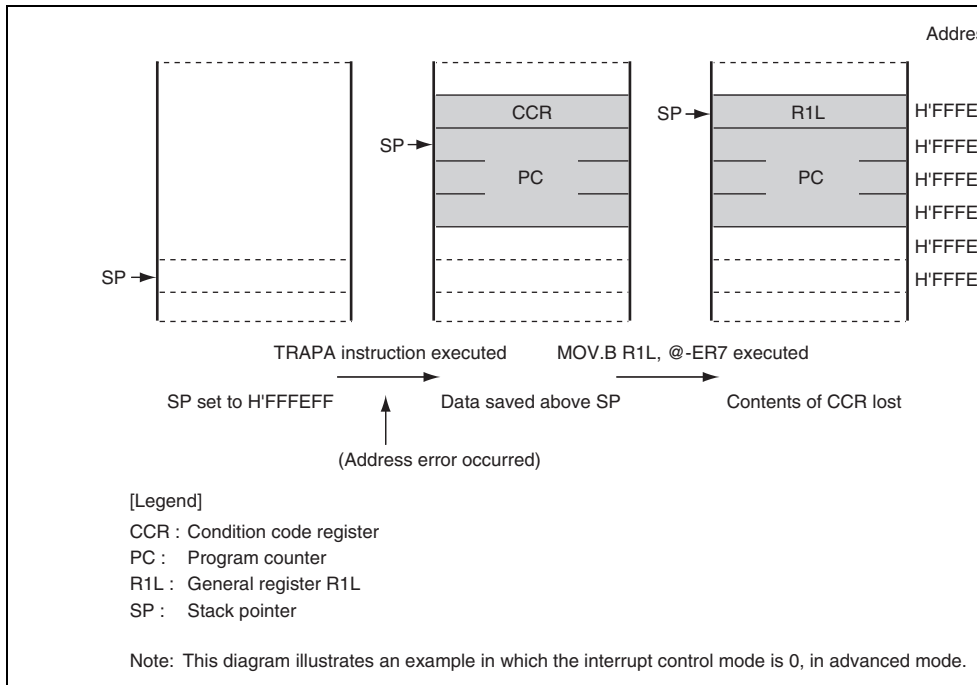
Interrupt control mode 2

Note: \* Ignored on return.

**Figure 6.3 Stack Status after Exception Handling**

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

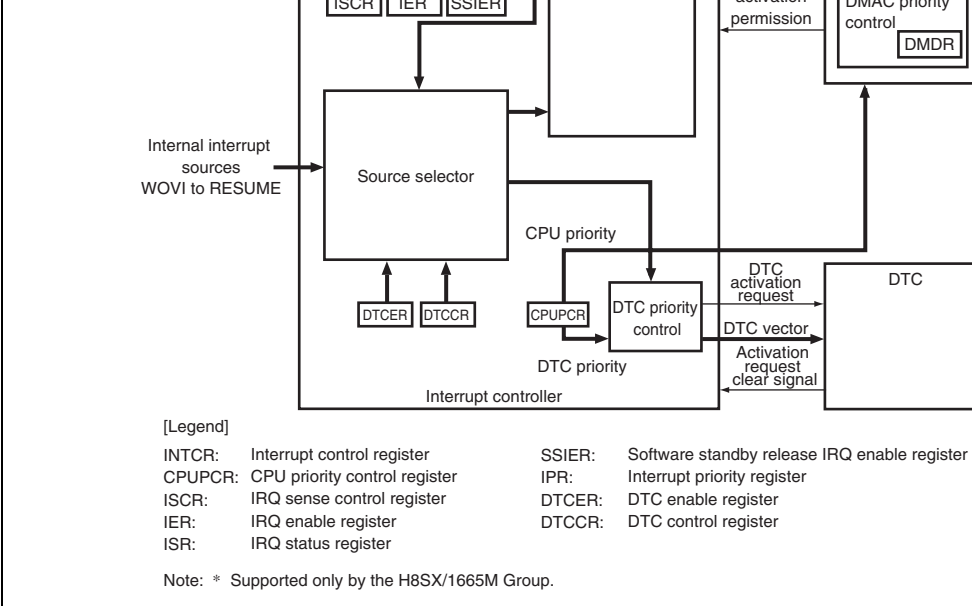
Performing stack manipulation while SP is set to an odd value leads to an address error. Figure 6.4 shows an example of operation when the SP value is odd.



**Figure 6.4 Operation when SP Value is Odd**

interrupts except for the interrupt requests listed below. The following eight interrupts are given priority of 8, therefore they are accepted at all times.

- NMI
- Illegal instructions
- Trace
- Trap instructions
- CPU address error
- DMA address error (occurred in the DTC, DMAC and EXDMAC)
- Sleep instruction
- UBC 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.
- Thirteen 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 both edge sensing, can be selected for IRQ11 to IRQ0.
- DTC, 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, DMAC and EXDMAC. The priority level of the CPU can be automatically assigned on an exception generation. Priority is given to the CPU interrupt exception handling over that of the DTC, DMAC and EXDMAC transfer.



**Figure 7.1 Block Diagram of Interrupt Controller**



## 7.3 Register Descriptions

The interrupt controller has the following registers.

- Interrupt control register (INTCR)
- CPU priority control register (CPUPCR)
- Interrupt priority registers A to O, Q, and R (IPRA to IPRO, IPRQ, and 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)

Bit	Bit Name	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 m 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 EX 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 I 1: Interrupt request generated at rising edge of M
2 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified

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

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 to 1 enables the CPU priority control over the DDMAC and EXDMAC.</p> <p>0: CPU always has the lowest priority</p> <p>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)</p> <p>001: Priority level 1</p> <p>010: Priority level 2</p> <p>011: Priority level 3</p> <p>100: Priority level 4</p> <p>101: Priority level 5</p> <p>110: Priority level 6</p> <p>111: Priority level 7 (highest)</p>

0	CPUP0	0	R/(W)*	<p>When the IPSETE bit is set to 1, the CPU priority control function over the DTC, DMAC and EXDMAC becomes the priority of CPU processing is assigned in accordance with the settings of bits CPUP2 to CPUP7.</p> <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>
---	-------	---	--------	---

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

Initial Value	0	1	1	1	0	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	—	IPR6	IPR5	IPR4	—	IPR2	IPR1
Initial Value	0	1	1	1	0	1	1
R/W	R	R/W	R/W	R/W	R	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.

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. 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)	
5	IPR5	1	R/W		
4	IPR4	1	R/W		
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. 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)
1	IPR1	1	R/W		
0	IPR0	1	R/W		

Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* Supported only by the H8SX/1665M Group.

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15E	0	R/W	IRQ15 Enable The IRQ15 interrupt request is enabled when 1. IRQ15 is internally connected to the 32KOV <sub>1</sub> in the TM32K.
14	IRQ14E*	0	R/W	IRQ14 Enable The IRQ14 interrupt request is enabled when 1. IRQ14 is internally connected to the voltage monitoring interrupt in the LVD
13, 12	—	All 0	R/W	Reserved These bits are always read as 0. The write value always be 0.
11	IRQ11E	0	R/W	IRQ11 Enable The IRQ11 interrupt request is enabled when 1.
10	IRQ10E	0	R/W	IRQ10 Enable The IRQ10 interrupt request is enabled when 1.
9	IRQ9E	0	R/W	IRQ9 Enable The IRQ9 interrupt request is enabled when t
8	IRQ8E	0	R/W	IRQ8 Enable The IRQ8 interrupt request is enabled when t

3	IRQ3E	0	R/W	IRQ3 Enable The IRQ3 interrupt request is enabled when th
2	IRQ2E	0	R/W	IRQ2 Enable The IRQ2 interrupt request is enabled when th
1	IRQ1E	0	R/W	IRQ1 Enable The IRQ1 interrupt request is enabled when th
0	IRQ0E	0	R/W	IRQ0 Enable The IRQ0 interrupt request is enabled when th

Note: \* Supported only by the H8SX/1665M Group.

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

ISCRH and ISCR L select the source that generates an interrupt request from  $\overline{\text{IRQ15}}$ ,  $\overline{\text{IRQ14}}$ ,  $\overline{\text{IRQ13}}$ ,  $\overline{\text{IRQ12}}$ ,  $\overline{\text{IRQ11}}$  to  $\overline{\text{IRQ0}}$  input.

Upon changing the setting of ISCR,  $\text{IRQnF}$  (n = 0 to 11, 14, 15) in ISR is often set to 1 accidentally through an internal operation. In this case, an interrupt exception handling is performed if an  $\text{IRQn}$  interrupt request is enabled. In order to prevent such an accidental interrupt from occurring, the setting of ISCR should be changed while the  $\text{IRQn}$  interrupt is disabled, and the  $\text{IRQnF}$  in ISR should be cleared to 0.



- ISCR1

Bit	15	14	13	12	11	10	9
Bit Name	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: \* Supported only by the H8SX/1665M Group.

13	IRQ14SR*	0	R/W	IRQ14 Sense Control Rise	11: Setting prohibited
12	IRQ14SF*	0	R/W	IRQ14 Sense Control Fall	IRQ14 is used as the voltage monitoring interrupt for LVD. Set falling edge interrupt request for use.
					00: Initial value
					01: Interrupt request generated at falling edge input
					10: Setting is prohibited
					11: Setting is prohibited
11 to 8	—	All 0	R/W	Reserved	
					These bits are always read as 0. The write value always be 0.
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 input
					01: Interrupt request generated at falling edge of input
					10: Interrupt request generated at rising edge of input
					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 input
					01: Interrupt request generated at falling edge of input
					10: Interrupt request generated at rising edge of input
					11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ10}}$

00: Interrupt request generated by low level o  
01: Interrupt request generated at falling edge  
10: Interrupt request generated at rising edge  
11: Interrupt request generated at both falling  
edges of  $\overline{\text{IRQ8}}$

---

Note: \* Supported only by the H8SX/1665M.

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 $\overline{IRQ6}$
				01: Interrupt request generated at falling edge of $\overline{IRQ6}$
				10: Interrupt request generated at rising edge of $\overline{IRQ6}$
				11: Interrupt request generated at both falling and rising edges of $\overline{IRQ6}$
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 $\overline{IRQ5}$
				01: Interrupt request generated at falling edge of $\overline{IRQ5}$
				10: Interrupt request generated at rising edge of $\overline{IRQ5}$
				11: Interrupt request generated at both falling and rising edges of $\overline{IRQ5}$
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 $\overline{IRQ4}$
				01: Interrupt request generated at falling edge of $\overline{IRQ4}$
				10: Interrupt request generated at rising edge of $\overline{IRQ4}$
				11: Interrupt request generated at both falling and rising edges of $\overline{IRQ4}$
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 $\overline{IRQ3}$
				01: Interrupt request generated at falling edge of $\overline{IRQ3}$
				10: Interrupt request generated at rising edge of $\overline{IRQ3}$
				11: Interrupt request generated at both falling and rising edges of $\overline{IRQ3}$

00: Interrupt request generated by low level of I  
 01: Interrupt request generated at falling edge of I  
 10: Interrupt request generated at rising edge of I  
 11: Interrupt request generated at both falling and rising  
 edges of  $\overline{IRQ1}$

---

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 I  
 01: Interrupt request generated at falling edge of I  
 10: Interrupt request generated at rising edge of I  
 11: Interrupt request generated at both falling and rising  
 edges of  $\overline{IRQ0}$

---

Bit Name	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1

- Notes: 1. Only 0 can be written, to clear the flag. The bit manipulation instructions or memory operation instructions can be used to clear the flag.  
2. Supported only by the H8SX/1665M Group.

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15F	0	R/(W)*1	[Setting condition] <ul style="list-style-type: none"> <li>When the interrupt selected by ISCR occurs.</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>Writing 0 after reading IRQ15F = 1</li> <li>When IRQ15 interrupt exception handling is executed while falling-edge sensing is selected.</li> </ul>
14	IRQ14F*2	0	R/(W)*1	[Setting condition] <ul style="list-style-type: none"> <li>When the interrupt selected by ISCR occurs.</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>Writing 0 after reading IRQ14 = 1</li> <li>When IRQ14 interrupt exception handling is executed while falling-edge sensing is selected.</li> </ul>
13, 12	—	All 0	R/(W)*1	Reserved These bits are always read as 0. The write value always be 0.

3	IRQ3F	0	R/(W)* <sup>1</sup>	selected
2	IRQ2F	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.</li> </ul>
1	IRQ1F	0	R/(W)* <sup>1</sup>	
0	IRQ0F	0	R/(W)* <sup>1</sup>	

- Notes: 1. Only 0 can be written, to clear the flag.  
2. Supported only by the H8SX/1665M Group.

### 7.3.7 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
Bit Name	SSI15	—	—	—	SSI11	SSI10	SSI9
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

These bits are always read as 0. The write value always be 0.

---

11	SSI11	0	R/W	Software Standby Release IRQ Setting
10	SSI10	0	R/W	These bits select the IRQn interrupt used to leave software standby mode (n = 11 to 0).
9	SSI9	0	R/W	
8	SSI8	0	R/W	0: An IRQn request is not sampled in software standby mode 1: When an IRQn request occurs in software standby mode, this LSI leaves software standby mode when the oscillation settling time has elapsed
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	

---



the NMI pin. Regardless of the interrupt control mode or the settings of the CPU interrupt mask, the NMIIEG bit in INTCR selects whether an interrupt is requested at the rising or falling edge of the NMI pin.

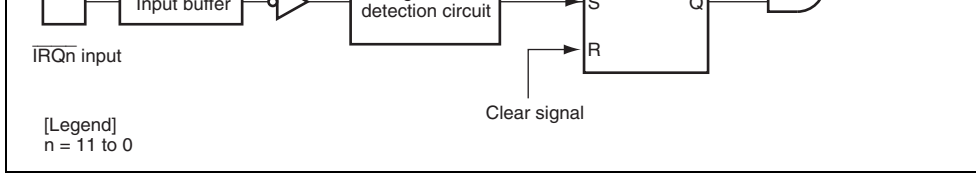
When an NMI interrupt is generated, the interrupt controller determines that an error has occurred and performs the following procedure.

- Sets the ERR bit of DTCCR in the DTC to 1.
- Sets the ERRF bit of DMDR\_0 in the DMAC to 1
- Sets the ERRF bit of EDMDR\_0 in EXDMAC to 1.
- Clears the DTE bits of DMDRs for all channels in the DMAC to 0 to forcibly terminate the data transfer.
- The DTE bits in EDMDR of all channels in EXDMAC are cleared to 0, and transfer is terminated.

## (2) IRQn Interrupts

An IRQn interrupt is requested by a signal input on pins  $\overline{\text{IRQ11}}$  to  $\overline{\text{IRQ0}}$ .  $\overline{\text{IRQn}}$  (n = 11 to 0) has 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.



**Figure 7.2 Block Diagram of Interrupts IRQn**

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

### 7.4.2 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.

Vector Address Onset*						DTC
Classification	Interrupt Source	Vector Number	Advanced Mode, Middle Mode, Maximum Mode	IPR	Priority	Activation
External pin	NMI	7	H'001C	—	High	—
UBC	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		
—	Reserved for system use	76	H'0130	—	—	
		77	H'0134	—	—	
LVD* <sup>2</sup>	Voltage-monitoring interrupt (IRQ14)	78	H'0138	IPRD6 to IPRD4	—	
TM32K	32KOV1 (IRQ15)	79	H'013C	IPRD2 to IPRD0	Low	—

—	Reserved for system use	84	H'0150	—	—
		85	H'0154		—
A/D_0	ADI0	86	H'0158	IPRF10 to IPRF8	O
—	Reserved for system use	87	H'015C	—	—
TPU_0	TGI0A	88	H'0160	IPRF6 to IPRF4	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
	TGI1B	94	H'0178		O
	TCI1V	95	H'017C		—
	TCI1U	96	H'0180		—
TPU_2	TGI2A	97	H'0184	IPRG14 to IPRG12	O
	TGI2B	98	H'0188		O
	TCI2V	99	H'018C		—
	TCI2U	100	H'0190		—
TPU_3	TGI3A	101	H'0194	IPRG10 to IPRG8	O
	TGI3B	102	H'0198		O
	TGI3C	103	H'019C		O
	TGI3D	104	H'01A0		O
	TCI3V	105	H'01A4		—

Low

	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
	DMTEND2	130	H'0208	IPRI6 to IPRI4	O
	DMTEND3	131	H'020C	IPRI2 to IPRI0	O

Low

	DMEEND2	138	H'0228		0
	DMEEND3	139	H'022C		0
EXDMAC	EXDMEEND0	140	H'0230	IPRK10 to IPRK8	0
	EXDMEEND1	141	H'0234		0
	EXDMEEND2	142	H'0238		0
	EXDMEEND3	143	H'023C		0
SCL_0	ERI0	144	H'0240	IPRK6 to IPRK4	—
	RXI0	145	H'0244		0
	TXI0	146	H'0248		0
	TEI0	147	H'024C		—
SCL_1	ERI1	148	H'0250	IPRK2 to IPRK0	—
	RXI1	149	H'0254		0
	TXI1	150	H'0258		0
	TEI1	151	H'025C		—
SCL_2	ERI2	152	H'0260	IPRL14 to IPRL12	—
	RXI2	153	H'0264		0
	TXI2	154	H'0268		0
	TEI2	155	H'026C		—
—	Reserved for system use	156	H'0270	—	—
		157	H'0274		—
		158	H'0278		—
		159	H'027C		—

Low

	TGI6C	166	H'0298		O
	TGI6D	167	H'029C		O
	TCI6V	168	H'02A0	IPRM14 to IPRM12	—
TPU_7	TGI7A	169	H'02A4	IPRM10 to IPRM8	O
	TGI7B	170	H'02A8		O
	TGI7V	171	H'02AC	IPRM6 to IPRM4	—
	TCI7U	172	H'02B0		—
TPU_8	TGI8A	173	H'02B4	IPRM2 to IPRM0	O
	TGI8B	174	H'02B8		O
	TCI8V	175	H'02BC	IPRN14 to IPRN12	—
	TCI8U	176	H'02C0		—
TPU_9	TGI9A	177	H'02C4	IPRN10 to IPRN8	O
	TGI9B	178	H'02C8		O
	TGI9C	179	H'02CC		O
	TGI9D	180	H'02D0		O
	TCI9V	181	H'02D4	IPRN6 to IPRN4	—
TPU_10	TGI10A	182	H'02D8	IPRN2 to IPRN0	O
	TGI10B	183	H'02DC		O
	Reserved for system use	184	H'02E0	—	—
	Reserved for system use	185	H'02E4		—
	TCI10V	186	H'02E8	IPRO14 to IPRO12	O
	TCI10U	187	H'02EC		—

Low

IIC2_0	IIC10	216	H'0360	IPRQ6 to IPRQ4	—
—	Reserved for system use	217	H'0364		—
IIC2_1	IIC11	218	H'0368		—
—	Reserved for system use	219	H'036C		—
SCI_5	RX15	220	H'0370	IPRQ2 to IPRQ0	—
	TX15	221	H'0374		—
	ERI5	222	H'0378		—
	TEI5	223	H'037C		—
SCI_6	RX16	224	H'0380	IPRR14 to IPRR12	—
	TX16	225	H'0384		—
	ERI6	226	H'0388		—
	TEI6	227	H'038C		—
TMR_4	CMIA4 or CMIB4	228	H'0390	IPRR10 to IPRR8	—
TMR_5	CMIA5 or CMIB5	229	H'0394		—
TMR_6	CMIA6 or CMIB6	230	H'0398		—
TMR_7	CMIA7 or CMIB7	231	H'039C		—
					Low



USB	resume	238	H'03B8		—
—	Reserved for system use	239	H'03BC	—	—
		255	H'03FC		Low —

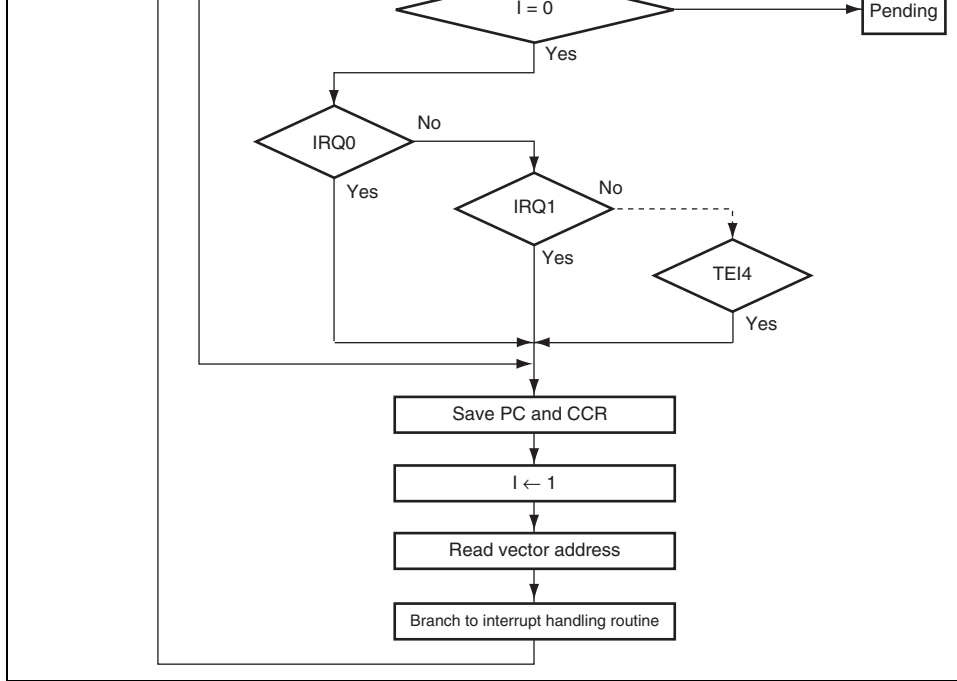
- Notes:
1. Lower 16 bits of the start address.
  2. Supported only by the H8SX/1665M Group.

Mode	Register	Mask Bit	Description
0	Default	I	The priority levels of the interrupt sources are fixed to the default settings. The interrupts except for NMI is masked by the I bit in CCR.
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 bit IPR.

### 7.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests except for NMI are masked by the I bit in CCR. Figure 7.3 shows a flowchart of the interrupt acceptance operation in this case.

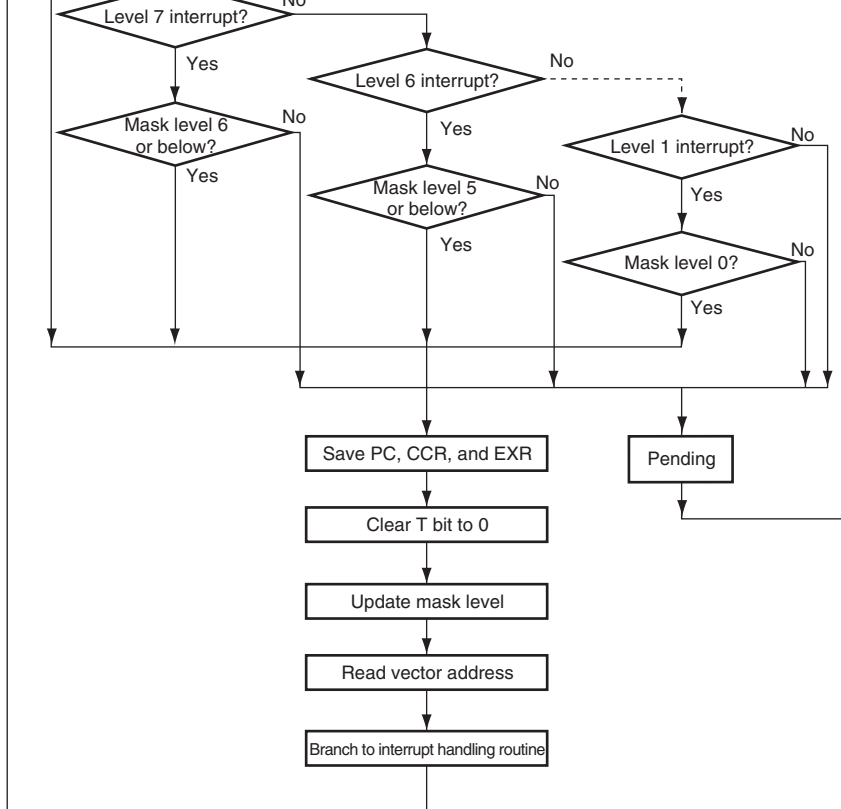
1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, the interrupt request is sent to the interrupt controller.
2. If the I bit in CCR is set to 1, NMI is accepted, and other interrupt requests are held pending. When the I bit is cleared to 0, an interrupt request 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 the 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 register in the vector table.



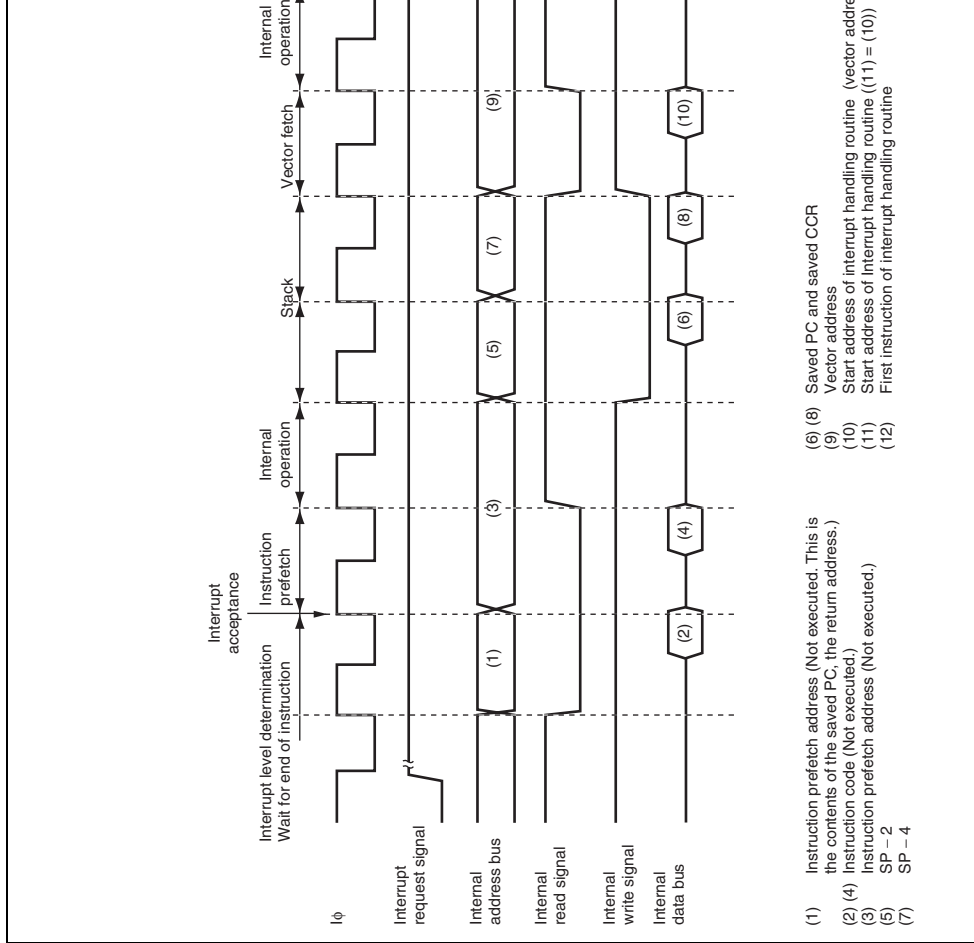
**Figure 7.3 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

multiple interrupt requests have the same priority, an interrupt request is selected according to the default setting shown in table 7.2.

3. Next, the priority of the selected interrupt request is compared with the interrupt mask level in EXR. When the interrupt request does not have priority over the mask level set, it is 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 the 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. 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 register and the vector table.



**Figure 7.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2**



**Figure 7.5 Interrupt Exception Handling**

Execution State	Normal Mode*		Advanced mode		Maximum
	Interrupt Control Mode 0	Interrupt Control Mode 2	Interrupt Control Mode 0	Interrupt Control Mode 2	Interrupt Control Mode 0
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>
Vector fetch				S <sub>h</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

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

[Legend]

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

### **7.6.5 DTC and DMAC Activation by Interrupt**

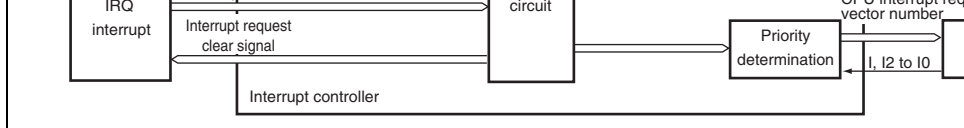
The DTC and DMAC can be activated by an interrupt. In this case, the following options are available:

- 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 10-1, section 10, DMA Controller (DMAC), and section 12, Data Transfer Controller (DTC).

Figure 7.6 shows a block diagram of the DTC, DMAC and interrupt controller.





**Figure 7.6 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 memory interrupt is enabled (DTF1 = 1, DTF0 = 0, and DTE = 1 in DMDR) and the DTA bit in DMACR 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 source or CPU interrupt sources by the DTCE bit in DTCERA to DTCERF 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 IPSET bit in CPUPCR is set to 1, the priority is determined according to the IPR setting. Therefore, the IPR setting or the IPR setting corresponding to the interrupt source must be set to lower than the DTCP and DMAP setting. If the CPU is given priority over the DTC or DMAC, the DMAC may not be activated, and the data transfer may not be performed.

operations are performed independently.

Table 7.6 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 DTCERF of the DTC, and the DISEL bit in MRB of the DTC.

**Table 7.6 Interrupt Source Selection and Clear Control**

Setting					
DMAC	DTC		Interrupt Source Selection/Clear Control		
DTA	DTCE	DISEL	DMAC	DTC	CPU
0	0	*	O	X	√
		1	O	√	X
		1	O	O	√
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 of the DTA bit in table 7.6, when the DTC or DMAC reads/writes the prescribed register.

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

EXDMAC is assigned by the bits EDMAP2 to EDMAP0 in EXDMA mode control register (EDMDR\_0 to EDMDR\_3) for each channel.

The priority control function over the DTC, DMAC and EXDMAC is enabled by setting CPUPCE bit in CPUPCR to 1. When the CPUPCE bit is 1, the DTC, DMAC and EXDMAC 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 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 level than that of the CPU are held.

The EXDMAC priority level can be assigned in each channel. The EXDMAC activation source is controlled by both the EXDMAC priority level, which is assigned by the bits EDMAP2 to EDMAP0 in the corresponding channel, and the CPU priority level. If the CPU has priority, the EXDMAC activation source for the corresponding channel is held. The activation source is re-enabled when the condition that has held the activation source is cancelled (CPUPCE = 1 and the value of bits CPUP2 to CPUP0 is greater than that of the bits EDMAP2 to EDMAP0). When different priority levels are assigned for each channel, channels having higher priority than CPU control are held.

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.

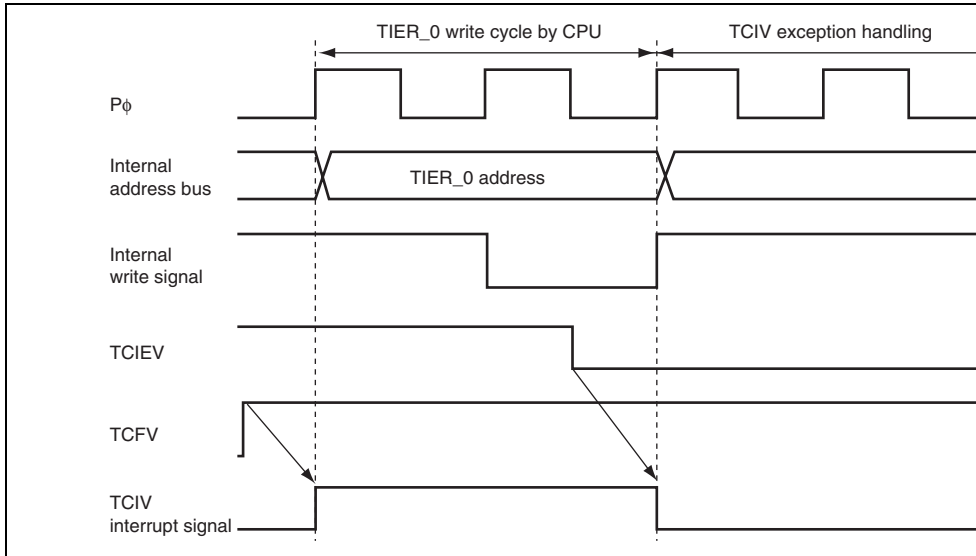
Table 7.7 shows the CPU priority control.

**Table 7.7 CPU Priority Control**

Interrupt Control Mode	Interrupt Priority	Interrupt Mask Bit	IPSETE in CPUPCR	Control Status	
				CPUP2 to CPUP0	Updating of 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

	1	B'000	B'000	B'000	B'000	Enabled	Enabled
		B'100	B'000	B'000	B'000	Masked	Masked
		B'100	B'000	B'011	B'100	Masked	Masked
		B'100	B'111	B'101	B'000	Enabled	Enabled
		B'000	B'111	B'101	B'000	Enabled	Enabled
2	0	Any	Any	Any	Any	Enabled	Enabled
	1	B'000	B'000	B'000	B'000	Enabled	Enabled
		B'000	B'011	B'101	B'110	Enabled	Enabled
		B'011	B'011	B'101	B'110	Enabled	Enabled
		B'100	B'011	B'101	B'110	Masked	Enabled
		B'101	B'011	B'101	B'110	Masked	Enabled
		B'110	B'011	B'101	B'110	Masked	Masked
		B'111	B'011	B'101	B'110	Masked	Masked
		B'101	B'011	B'101	B'011	Masked	Enabled
		B'101	B'110	B'101	B'011	Enabled	Enabled

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 7.7 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 when the interrupt is masked.



**Figure 7.7 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.

The interrupt controller disables interrupt acceptance for a 3-state period after the CPU I updated the mask level with an LDC, ANDC, ORC, or XORC instruction, and for a period of time after writing to the registers of the interrupt controller.

#### 7.8.4 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
```

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

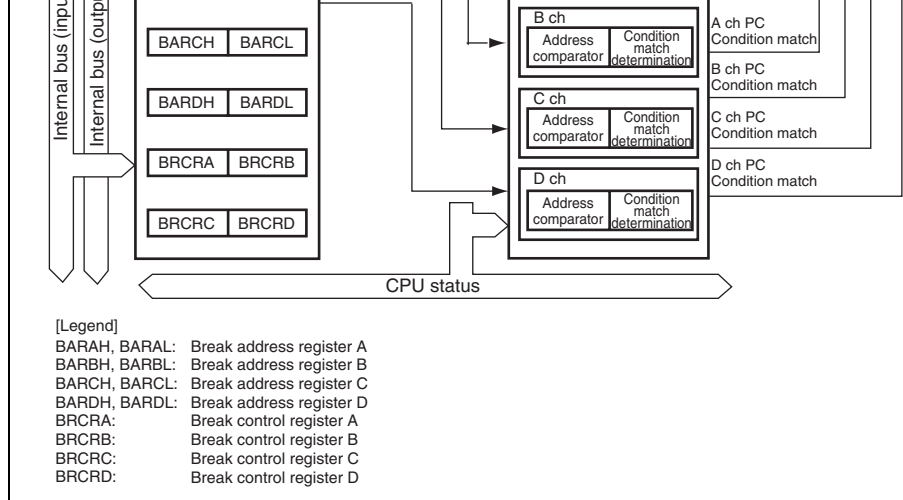
With the MOVMD or MOVSD instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual transfer cycle. The PC value saved on the stack in this case is the address of the MOVMD or MOVSD instruction. The transfer of remaining data is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





## 8.1 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))
- UBC break interrupt exception handling is executed immediately before execution of instruction fetched from the specified address (PC break).
- Module stop state can be set



**Figure 8.1 Block Diagram of UBC**

	BAMRAL	R/W	H'0000	H'FFA06
Break address register B	BARBH	R/W	H'0000	H'FFA08
	BARBL	R/W	H'0000	H'FFA0A
Break address mask register B	BAMRBH	R/W	H'0000	H'FFA0C
	BAMRBL	R/W	H'0000	H'FFA0E
Break address register C	BARCH	R/W	H'0000	H'FFA10
	BARCL	R/W	H'0000	H'FFA12
Break address mask register C	BAMRCH	R/W	H'0000	H'FFA14
	BAMRCL	R/W	H'0000	H'FFA16
Break address register D	BARDH	R/W	H'0000	H'FFA18
	BARDL	R/W	H'0000	H'FFA1A
Break address mask register D	BAMRDH	R/W	H'0000	H'FFA1C
	BAMRDL	R/W	H'0000	H'FFA1E
Break control register A	BRCRA	R/W	H'0000	H'FFA28
Break control register B	BRCRB	R/W	H'0000	H'FFA2C
Break control register C	BRCRC	R/W	H'0000	H'FFA30
Break control register D	BRCRD	R/W	H'0000	H'FFA34

BARnL

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

- 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) of 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) of the address break-condition on channel n.

[Legend]

n = Channels A to D

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 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 condition in the break control register.

[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 satisfactory 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 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 cycle 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	

condition for the given channel.

00: Break requests will not be generated.

01: The bus cycle break condition is read cycle.

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

consist of CPU cycle, PC break, and reading. Condition comparison is not performed. 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 in the corresponding channel. These flags are set when the break condition matches but are cleared when it no longer does. To confirm setting of the same flag again, read the flag from the break interrupt handling routine, and then write 0 to it (the flag is cleared by writing 1 to it after reading it as 1).

[Legend]

n = Channels A to D

### 8.4.2 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 condition never be generated.
2. The break occurs after fetching and execution of the target instruction have been completed. 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 channel.
4. The PC break condition is generated by specifying CPU cycles as the bus condition in the control register n (BRCRn.CPn0 = 1), PC break as the break condition (IDn0 = 1), and CPU cycles as the bus-cycle condition (RWn0 = 1).

[Legend]

n = Channels A to D

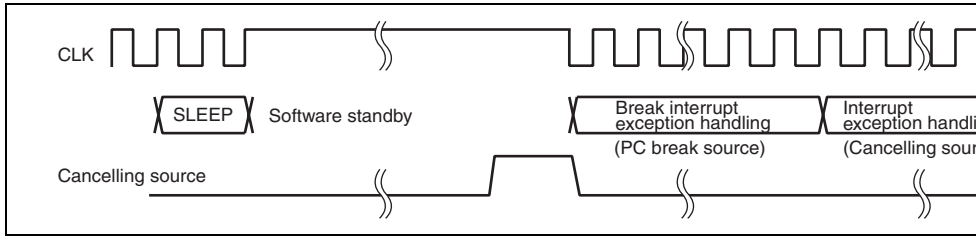


---

BRCRC	CMFCPC (bit 13)	for channel B Indicates that the condition matches in the CPU for channel C
BRCRD	CMFCPD (bit 13)	Indicates that the condition matches in the CPU for channel D

---

oscillation settling time has elapsed subsequent to the transition to software standby. When an interrupt is the canceling source, interrupt exception handling is executed after the SLEEP instruction, and the instruction following the SLEEP instruction is then executed.



**Figure 8.2 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 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.

4. After setting break conditions for the UBC, an unexpected UBC break interrupt may occur after the execution of an illegal instruction. This depends on the value of the program counter and the internal bus cycle.



- 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
  - DRAM, synchronous DRAM, 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

Synchronous DRAM interface is available as area 2  
Row/column address-multiplexed output (8, 9, 10, or 11 bits)

DQM signals control byte access for 16-bit data bus device

Auto refresh and self refresh are selectable

$\overline{\text{CAS}}$  latency can be selected from 2 to 4

High-speed data transfer is available using EXDMAC cluster transfer

- 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/EXDMAC single 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

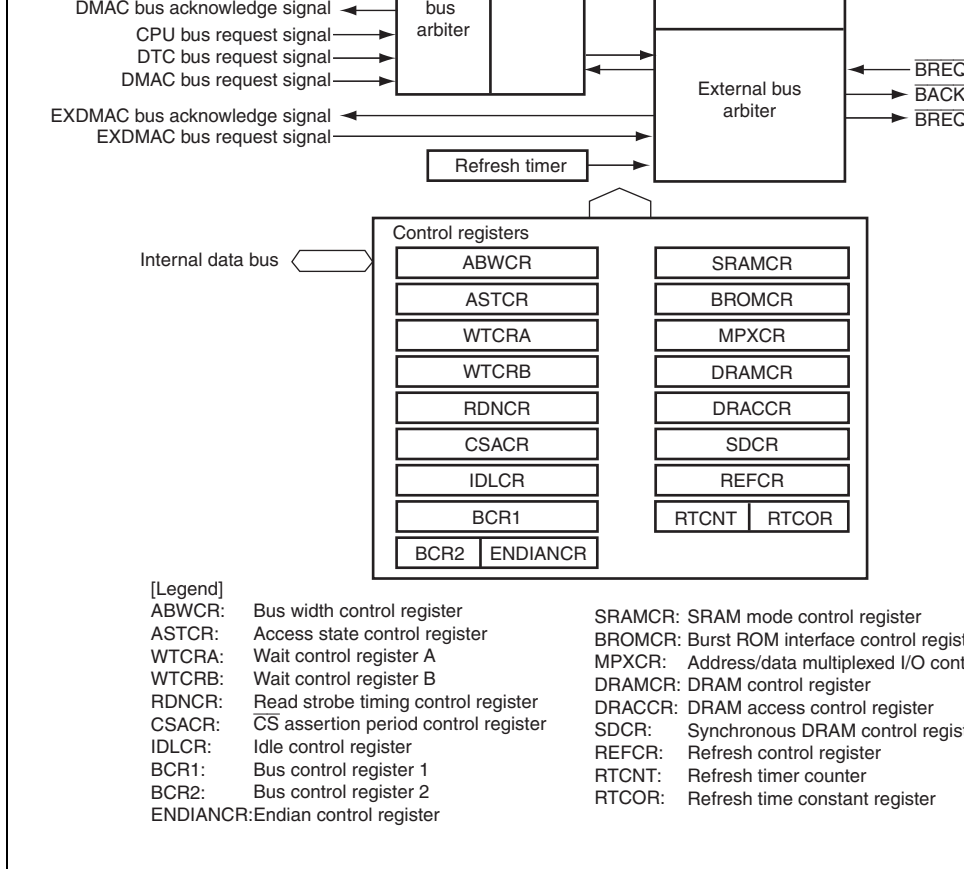
Includes a bus arbiter that arbitrates bus mastership among the CPU, DMAC, EXDMAC, DTC, refresh cycle, and external bus master

- EXDMAC external bus transfers and internal accesses can be executed in parallel.

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



**Figure 9.1 Block Diagram of Bus Controller**

- 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)
- DRAM control register (DRAMCR)
- DRAM access control register (DRACCR)
- Synchronous DRAM control register (SDCR)
- Refresh control register (REFCR)
- Refresh timer counter (RTCNT)
- Refresh time constant register (RTCOR)



Bit Name	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1
Initial Value	1	1	1	1	1	1	1
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 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	
8	ABWL0	1/0	R/W	1 1: Area n is designated as 8-bit access space* <sup>2</sup>
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 16-bit access space.

Bit Name	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
R/W	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 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
9	AST1	1	R/W	
8	AST0	1	R/W	(n = 7 to 0)
7 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified

Bit	7	6	5	4	3	2	1
Bit Name	—	W52	W51	W50	—	W42	W41
Initial Value	0	1	1	1	0	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W

• WTCRB

Bit	15	14	13	12	11	10	9
Bit Name	—	W32	W31	W30	—	W22	W21
Initial Value	0	1	1	1	0	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W

Bit	7	6	5	4	3	2	1
Bit Name	—	W12	W11	W10	—	W02	W01
Initial Value	0	1	1	1	0	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W

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

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 cy when accessing area 6 while bit AST6 in ASTCF
8	W60	1	R/W	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
7	—	0	R	Reserved This is a read-only bit and cannot be modified.

101: 5 program wait cycles inserted  
 110: 6 program wait cycles inserted  
 111: 7 program wait cycles inserted

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 inserted when accessing area 4 while bit AST4 in ASTC is set.
0	W40	1	R/W	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

- WTCRB

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This is a read-only bit and cannot be modified.

				10: 6 program wait cycles inserted 111: 7 program wait cycles inserted
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 inserted when accessing area 2 while bit AST2 in ASTCR is set to 1. When SDRAM is connected, the CAS latency is ignored. The CAS latency can be specified even if the wait cycle insertion is disabled by ASTCR.
8	W20	1	R/W	Selection of number of program wait cycles: 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 Setting of CAS latency (W22 is ignored.): 00: Setting prohibited 01: SDRAM with a CAS latency of 2 is connected 10: SDRAM with a CAS latency of 3 is connected 11: SDRAM with a CAS latency of 4 is connected

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

---

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 AST0 is set to 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	

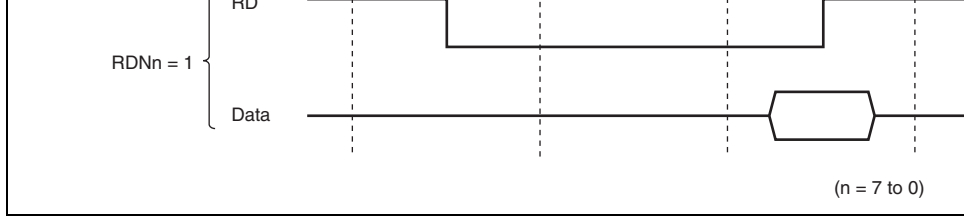
---

Bit Name	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
R/W	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
13	RDN5	0	R/W	strobe in a corresponding area read access.
12	RDN4	0	R/W	As shown in figure 9.2, the read strobe for an
11	RDN3	0	R/W	which the RDNn bit is set to 1 is negated one
10	RDN2	0	R/W	cycle earlier than that for an area for which the
9	RDN1	0	R/W	bit is cleared to 0. The read data setup and hold
8	RDN0	0	R/W	are also given one half-cycle earlier.
				0: In an area n read access, the $\overline{RD}$ signal is n
				at the end of the read cycle
				1: In an area n read access, the $\overline{RD}$ signal is n
				one half-cycle before the end of the read cy
				(n = 7 to 0)
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, RDNCr setting is ignored and the same operation when RDNn = 1 is performed.
  2. In an external address space which is specified as the burst ROM interface, the RDNCr setting is ignored and the same operation when RDNn = 0 is performed. read accesses by the CPU and EXDMAC cluster transfer.





**Figure 9.2 Read Strobe Negation Timing (Example of 3-State Access Space)**

### 9.2.5 $\overline{CS}$ Assertion Period Control Registers (CSACR)

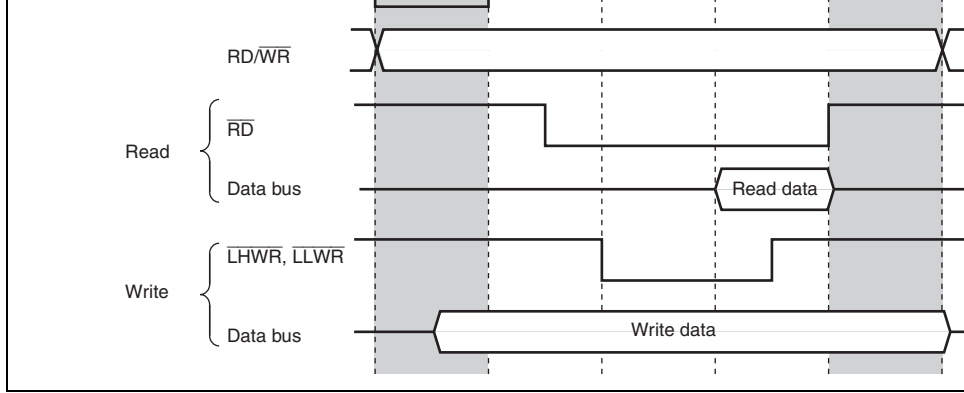
CSACR selects whether or not the assertion periods of the chip select signals ( $\overline{CSn}$ ) and signals for the basic bus, byte-control SRAM, burst ROM, and address/data multiplexed interface are to be extended. Extending the assertion period of the  $\overline{CSn}$  and address signals the setup time and hold time of read strobe ( $\overline{RD}$ ) and write strobe ( $\overline{LHWR}/\overline{LLWR}$ ) to be flexible 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
Bit Name	CSXH7	CSXH6	CSXH5	CSXH4	CSXH3	CSXH2	CSXH1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	CSXT7	CSXT6	CSXT5	CSXT4	CSXT3	CSXT2	CSXT1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

period (Th) is extended  
(n = 7 to 0)

7	CSXT7	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Contr
6	CSXT6	0	R/W	These bits specify whether or not the Tt cycle is
5	CSXT5	0	R/W	inserted (see figure 9.3). When an area for which
4	CSXT4	0	R/W	CSXTn is set to 1 is accessed, one Tt cycle, in
3	CSXT3	0	R/W	the $\overline{CSn}$ and address signals are retained, is in
2	CSXT2	0	R/W	after the normal access cycle.
1	CSXT1	0	R/W	0: In access to area n, the $\overline{CSn}$ and address as
0	CSXT0	0	R/W	period (Tt) is not extended
				1: In access to area n, the $\overline{CSn}$ and address as
				period (Tt) is extended
				(n = 7 to 0)

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



**Figure 9.3  $\overline{CS}$  and Address Assertion Period Extension**  
 (Example of Basic Bus Interface, 3-State Access Space, and RDNn = 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	<p>Idle Cycle Insertion 3</p> <p>Inserts an idle cycle between the bus cycles when DMAC or EXDMAC single address transfer (write) is followed by external access.</p> <p>0: No idle cycle is inserted 1: An idle cycle is inserted</p>
14	IDLS2	1	R/W	<p>Idle Cycle Insertion 2</p> <p>Inserts an idle cycle between the bus cycles when external write cycle is followed by external read cycle.</p> <p>0: No idle cycle is inserted 1: An idle cycle is inserted</p>
13	IDLS1	1	R/W	<p>Idle Cycle Insertion 1</p> <p>Inserts an idle cycle between the bus cycles when external read cycles of different areas continue.</p> <p>0: No idle cycle is inserted 1: An idle cycle is inserted</p>

				00: No idle cycle is inserted
				01: 2 idle cycles are inserted
				00: 3 idle cycles are inserted
				01: 4 idle cycles are inserted
9	IDLCA1	1	R/W	Idle Cycle State Number Select A
8	IDLCA0	1	R/W	Specifies the number of idle cycles to be inserted under the idle condition specified by IDLS3 to IDLS0.
				00: 1 idle cycle is inserted
				01: 2 idle cycles are inserted
				10: 3 idle cycles are inserted
				11: 4 idle cycles are inserted
7	IDLSEL7	0	R/W	Idle Cycle Number Select
6	IDLSEL6	0	R/W	Specifies the number of idle cycles to be inserted under each area for the idle insertion condition specified by IDLS1 and IDLS0.
5	IDLSEL5	0	R/W	
4	IDLSEL4	0	R/W	0: Number of idle cycles to be inserted for area specified by IDLCA1 and IDLCA0.
3	IDLSEL3	0	R/W	
2	IDLSEL2	0	R/W	1: Number of idle cycles to be inserted for area specified by IDLCB1 and IDLCB0.
1	IDLSEL1	0	R/W	
0	IDLSEL0	0	R/W	(n = 7 to 0)

Bit Name	DKC	EDKC	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	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 13, 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{BREQ}}</math>) to the external bus master in the external bus released state when an internal bus master requests 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>
13, 12	—	All 0	R	<p>Reserved</p> <p>These are read-only bits and cannot be modified.</p>
11, 10	—	All 0	R/W	<p>Reserved</p> <p>These bits are always read as 0. The write value should always be 0.</p>

Selects enabling/disabling of wait input by  $\overline{\text{WAIT}}$  pin. When area 2 is specified as the synchronous DRAM space, the setting of this bit does not affect the synchronous DRAM space access operation.

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 13, I/O Ports.

7	DKC	0	R/W	<p><math>\overline{\text{DACK}}</math> Control</p> <p>Selects the timing of DMAC transfer acknowledgment signal assertion.</p> <p>0: <math>\overline{\text{DACK}}</math> signal is asserted at the <math>B\phi</math> falling edge</p> <p>1: <math>\overline{\text{DACK}}</math> signal is asserted at the <math>B\phi</math> rising edge</p>
6	EDKC	0	R/W	<p><math>\overline{\text{EDACK}}</math> Control</p> <p>Controls the assertion timing of an acknowledgment signal for an EXDMAC transfer.</p> <p>0: <math>\overline{\text{EDACK}}</math> signal asserted at the falling edge</p> <p>1: <math>\overline{\text{EDACK}}</math> signal asserted at the rising edge</p>
5 to 0	—	All 0	R	<p>Reserved</p> <p>These are read-only bits and cannot be modified.</p>

Note: When external bus release is enabled or input by the  $\overline{\text{WAIT}}$  pin is enabled, make the ICR bit to 1. For details, see section 13, I/O Ports.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These are read-only bits and cannot be modified.
5	EBCCS	0	R/W	External Bus Cycle Control Select Selects the method for external bus arbitration. 0: Releases the bus depending on the priority 1: Executes the bus cycle alternatively when a bus request occurs between a bus request by the EXDMA external bus master or refresh bus and a request for an external space access by the CPU, DMA controller, or DTC.
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 bus mastership request conflicts with a DMA controller 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



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 end
4	LE4	0	R/W	1: Data format of area n is specified as little en
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

Initial Value	7	6	5	4	3	2	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
R/W	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
13	BCSEL5	0	R/W	When setting a bit to 1, the bus interface select
12	BCSEL4	0	R/W	BROMCR, DRAMCR and MPXCR must be clear
11	BCSEL3	0	R/W	0.
10	BCSEL2	0	R/W	0: Area n is basic bus interface
9	BCSEL1	0	R/W	1: Area n is byte control SRAM interface
8	BCSEL0	0	R/W	(n = 7 to 0)
7 to 0	—	All 0	R	Reserved

These are read-only bits and cannot be modified.

Bit Name	BSRMI	BSTS12	BSTS11	BSTS10	—	—	BSWD11
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R	R	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 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.

Specifies the area 1 bus interface as a basic interface or a burst ROM interface. To set this bit to 1, clear 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 access
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 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)

Bit Name	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

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 correspondi
13	MPXE5	0	R/W	To set this bit to 1, clear the BCSELn bit in SF
12	MPXE4	0	R/W	0.
11	MPXE3	0	R/W	0: Area n is specified as a basic interface or a control SRAM interface. 1: Area n is specified as an address/data mult I/O interface (n = 7 to 3)
10 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modifi
0	ADDEX	0	R/W	Address Output Cycle Extension Specifies whether a wait cycle is inserted for t address output cycle of address/data multiplex interface. 0: No wait cycle is inserted for the address ou 1: One wait cycle is inserted for the address o cycle

Bit Name	BE	RCDM	DDS	EDDS	—	—	MXC1	
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	

Bit	Bit Name	Initial Value	R/W	Description
15	DRAME	0	R/W	<p>Area 2 DRAM Interface Select</p> <p>Selects whether or not area 2 is specified as the DRAM/SDRAM interface. When this bit is set to 1, the type of DRAM to be used in area 2 is selected by the DTYPE bit. When this bit is set to 1, the BCSE and SRAMCR should be set to 0.</p> <p>0: Basic bus interface or byte-control SRAM interface 1: DRAM/SDRAM interface</p>
14	DTYPE	0	R/W	<p>DRAM Select</p> <p>Selects the type of DRAM to be used in area 2.</p> <p>0: DRAM is used in area 2 1: SDRAM is used in area 2</p>
13, 12	—	All 0	R	<p>Reserved</p> <p>The initial value should not be changed.</p>

Selects whether the  $\overline{RAS}$  signal is asserted at the rising edge or falling edge of the  $B\phi$  signal in the  $T_r$  cycle during a DRAM access. The relationship between  $\overline{RAS}$  bit and  $\overline{RAS}$  assertion timing is shown in figure 10-10.

When SDRAM is used, the setting of this bit does not affect operation.

0:  $\overline{RAS}$  signal is asserted at the falling edge of  $B\phi$  signal in the  $T_r$  cycle

1:  $\overline{RAS}$  signal is asserted at the rising edge of  $B\phi$  signal in the  $T_r$  cycle

9	—	0	R	Reserved The initial value should not be changed.
8	CAST	0	R/W	Column Address Output Cycle Count Select Selects whether the number of column address output cycles is two or three during a DRAM access. When SDRAM is used, the setting of this bit does not affect operation. 0: Column address is output for two cycles 1: Column address is output for three cycles
7	BE	0	R/W	Burst Access Enable Enables or disables a burst access to the DRAM/SDRAM. The DRAM/SDRAM is accessed in high-speed page mode. When DRAM with the page mode is used, connect the $\overline{OE}$ signal of the DRAM to the $\overline{OE}$ signal of DRAM. 0: DRAM/SDRAM is accessed with full access mode 1: DRAM/SDRAM is accessed in high-speed page mode

interface, the READ/WRITE command is issued. When the DRAM/SDRAM interface is accessed consecutively, the issuance of the ACTV command when the same address is accessed consecutively.

0: RAS up mode when the DRAM/SDRAM is accessed

1: RAS down mode when the DRAM/SDRAM is accessed

---

5	DDS	0	R/W	<b>DMAC Single Address Transfer Option</b> Selects whether a DMAC single address transfer through the DRAM/SDRAM interface is enabled in full access mode or is also enabled in fast-page mode.  When clearing the BE bit to 0 to disable a burst access to the DRAM/SDRAM interface, a DMAC single address transfer is performed in full access mode regardless of this bit.  This bit does not affect an external access by other masters or a DMAC dual address transfer. Setting this bit to 1 changes the $\overline{\text{DACK}}$ output timing.  0: DMAC single address transfer through the DRAM/SDRAM is enabled only in full access mode  1: DMAC single address transfer through the DRAM/SDRAM is also enabled in fast-page mode
---	-----	---	-----	---

---



This bit does not affect an external access by masters or an EXDMAC dual address transfer. Setting this bit to 1 changes the EDACK output timing.

0: EXDMAC single address transfer through t  
DRAM/SDRAM is enabled only in full access

1: EXDMAC single address transfer through t  
DRAM/SDRAM is also enabled in fast-page  
mode

---

3	—	0	R	Reserved
2	—	0	R/W	The initial value should not be changed.

---

01: Shifted by 9 bits

A23 to A9 are compared for 8-bit access space

A23 to A10 are compared for 16-bit access space

10: Shifted by 10 bits

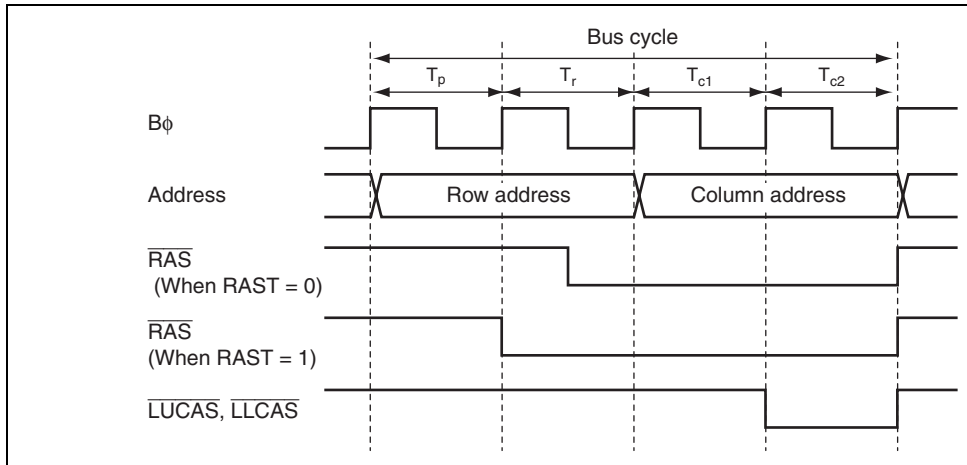
A23 to A10 are compared for 8-bit access space

A23 to A11 are compared for 16-bit access space

11: Shifted by 11 bits

A23 to A11 are compared for 8-bit access space

A23 to A12 are compared for 16-bit access space



**Figure 9.4  $\overline{\text{RAS}}$  Assertion Timing (Column Address Output for 2 states in Full Access Mode)**

Bit Name	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
15, 14	—	All 0	R	Reserved The initial value should not be changed.
13	TPC1	0	R/W	Precharge Cycle Control
12	TPC0	0	R/W	Select the number of RAS precharge cycles of normal access and a refresh cycle. 00: One cycle 01: Two cycles 10: Three cycles 11: Four cycles
11, 10	—	All 0	R	Reserved The initial value should not be changed.
9	RCD1	0	R/W	RAS-CAS Wait Control
8	RCD0	0	R/W	Select the number of wait cycles inserted between RAS and CAS cycles. 00: No wait cycle inserted 01: One wait cycle inserted 10: Two wait cycles inserted 11: Three wait cycles inserted
7 to 0	—	All 0	R	Reserved The initial value should not be changed.

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

Bit	Bit Name	Initial Value	R/W	Description
15	MRSE	0	R/W	<p>Mode Register Set Enable</p> <p>Enables the setting in the SDRAM mode register section 9.11.14, Setting SDRAM Mode Register</p> <p>0: Disables to set the SDRAM mode register</p> <p>1: Enables to set the SDRAM mode register</p>
14 to 12	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0. The initial value should not be changed.</p>
11, 10	—	0	R/W	<p>Reserved</p> <p>The initial value should not be changed.</p>
9	—	0	R	Reserved
8	—	0	R/W	The initial value should not be changed.
7	CKSPE	0	R/W	<p>Clock Suspend Enable</p> <p>Enables the clock suspend mode in which read output cycles are extended. Setting this bit to 1 extends the output cycles in which read data is output from SDRAM.</p> <p>0: Disables the clock suspend mode</p> <p>1: Enables the clock suspend mode</p>

## 9.2.16 Refresh Control Register (REFCR)

REFCR specifies the refresh type for the DRAM/SDRAM interface.

Bit	15	14	13	12	11	10	9
Bit Name	CMF	CMIE	RCW1	RCW0	—	RTCK2	RTCK1
Initial Value	0	0	0	0	0	0	0
R/W	R/(W)*	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	RFSHE	RLW2	RLW1	RLW0	SLFRF	TPCS2	TPCS1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

- When RTCNT matches RTCOR

14	CMIE	0	R/W	<p>Compare Match Interrupt Enable</p> <p>Enables or disables an interrupt request (CMI). CMF flag is set to 1.</p> <p>This bit is effective when refresh control is not performed (RFSHE = 0). When refresh control performed (RFSHE = 1), this bit is always clear. This bit cannot be modified.</p>
13 to 12	RCW1	0	R/W	CAS-RAS Wait Control
	RCW0	0	R/W	<p>Select the number of wait cycles inserted between <math>\overline{\text{CAS}}</math> asserted cycle and <math>\overline{\text{CAS}}</math> asserted cycle during DRAM refresh.</p> <p>When the SDRAM space is selected, these bits affect operations although they can be read from and written to.</p> <p>00: No wait cycle inserted  01: One wait cycle inserted  10: Two wait cycles inserted  11: Three wait cycles inserted</p>
11	—	0	R	<p>Reserved</p> <p>The initial value should not be changed.</p>

001: Counts on Pφ/128  
 001: Counts on Pφ/512  
 001: Counts on Pφ/2048  
 001: Counts on Pφ/4096

7	RFSHE	0	R/W	Refresh Control Enables or disables refresh control. When refresh control is disabled, the refresh timer can be used as an interval timer. In single-chip activation mode, the setting of this bit should be made after setting the EXPE bit in the EXPC register. 1. For SYSCR, see section 3, MCU Operating Modes. 0: Refresh control enabled 1: Refresh control disabled
6	RLW2	0	R/W	Refresh Cycle Wait Control
5	RLW1	0	R/W	Select the number of wait cycles during a CAS refresh cycle for the DRAM interface and a RAS refresh cycle for the SDRAM interface.
4	RLW0	0	R/W	Select the number of wait cycles during a RAS refresh cycle for the DRAM interface and a CAS refresh cycle for the SDRAM interface. 000: No wait cycle inserted 001: One wait cycle inserted 010: Two wait cycles inserted 010: Three wait cycles inserted 010: Four wait cycles inserted 010: Five wait cycles inserted 010: Six wait cycles inserted 010: Seven wait cycles inserted

0: Disables self-refresh

1: Enables self-refresh

---

2	TPS2	0	R/W	Precharge Cycle Control during Self-Refresh
1	TPS1	0	R/W	Selects the number of precharge cycles immediately after a self-refresh cycle. The number of actual precharge cycles is the sum of the numbers of precharge cycles selected by these bits and bits TPS1 and TPS0.
0	TPS0	0	R/W	

000: No wait cycle inserted  
001: One wait cycle inserted  
010: Two wait cycles inserted  
010: Three wait cycles inserted  
010: Four wait cycles inserted  
010: Five wait cycles inserted  
010: Six wait cycles inserted  
010: Seven wait cycles inserted

---

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



Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

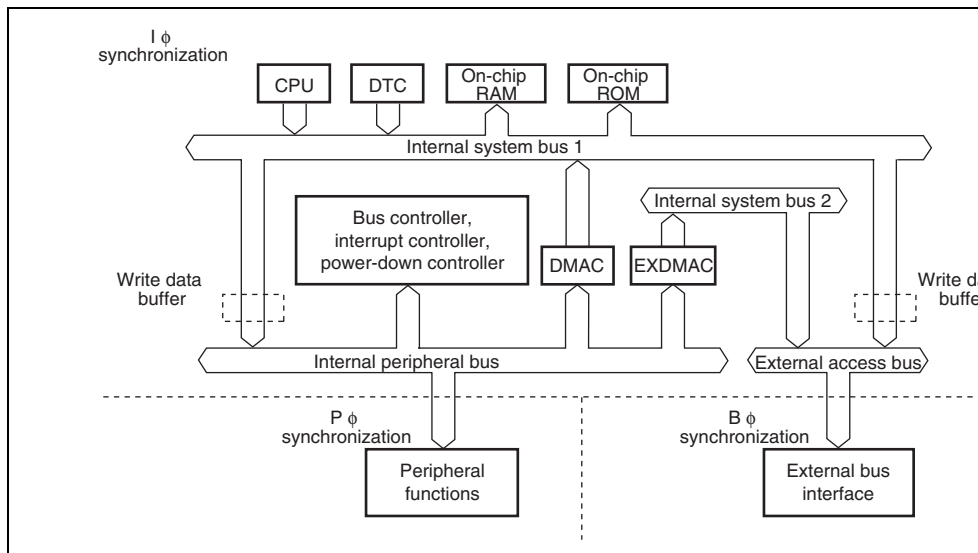
### 9.2.18 Refresh Time Constant Register (RTCOR)

RTCOR specifies intervals at which a compare match for RTCOR and RTCNT is generated.

The RTCOR value is always compared with the RTCNT value. When they match, the CTRM bit in REFCR is set to 1 and RTCNT is initialized to H'00.

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

- Internal peripheral bus  
A bus that accesses registers in the bus controller, interrupt controller, DMAC, and EXDMAC and registers of peripheral modules such as SCI and timer.
- External access bus  
A bus that accesses external devices via the external bus interface.



**Figure 9.5 Internal Bus Configuration**

	Bus controller CPU DTC DMAC EXDMAC Internal memory Clock pulse generator Power down control
P $\phi$	I/O ports TPU PPG TMR WDT SCI A/D D/A IIC2 USB
B $\phi$	External bus interface

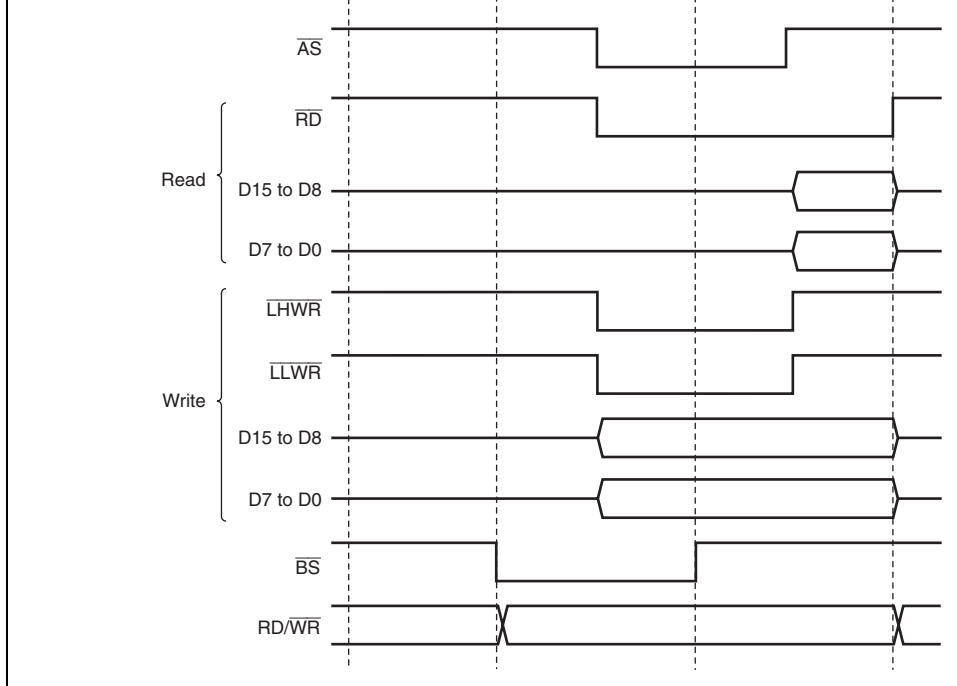
The frequency of each synchronization clock (I $\phi$ , P $\phi$ , and B $\phi$ ) is specified by the system control register (SCKCR) independently. For further details, see section 27, Clock Pulse Generator.

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 and external space is performed synchronously with P $\phi$  and B $\phi$ , respectively.

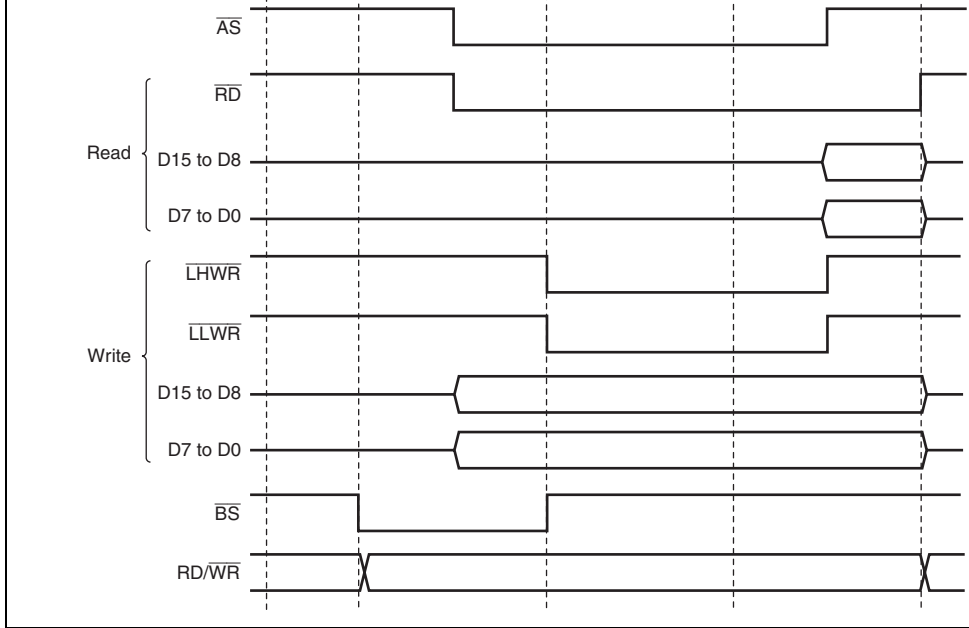
the frequency rate of  $I\phi$  and  $R\phi$  is  $m : 1$ , 0 to  $m-1$  cycles of  $T_{sy}$  may be inserted.

Figure 9.6 shows the external 2-state access timing when the frequency rate of  $I\phi$  and  $B\phi$

Figure 9.7 shows the external 3-state access timing when the frequency rate of  $I\phi$  and  $B\phi$



**Figure 9.6 System Clock: External Bus Clock = 4:1, External 2-State Access**



**Figure 9.7 System Clock: External Bus Clock = 2:1, External 3-State Access**

Bus cycle start	$\overline{CS}$	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, SRAM, or burst ROM space is accessed and output on address bus is enabled</li> <li>Signal to hold the address during access to address/data multiplexed I/O interface</li> </ul>
Read strobe	$\overline{RD}$	Output	Strobe signal indicating that the basic bus, SRAM, burst ROM, or address/data multiplexed 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 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, or address/data multiplexed I/O space is written, and the upper byte (D15 to D8) of data bus is enabled</li> <li>Strobe signal indicating that the byte control 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, or address/data multiplexed I/O space is written, and the lower byte (D7 to D0) of data bus is enabled</li> <li>Strobe signal indicating that the byte control space is accessed, and the lower byte (D7 to D0) of data bus is enabled</li> </ul>

Row address strobe	RAS	Output	<ul style="list-style-type: none"> <li>Row address strobe signal when area 2 specified as DRAM space</li> <li>Row address strobe signal when area 2 specified as SDRAM space</li> </ul>
Column address strobe	$\overline{\text{CAS}}$	Output	Column address strobe signal when area 2 specified as SDRAM space
Write enable	$\overline{\text{WE}}$	Output	<ul style="list-style-type: none"> <li>Write enable signal for DRAM</li> <li>Write enable signal when area 2 is specified as SDRAM space</li> </ul>
Lower-upper-column address strobe/lower-upper-data mask enable	$\overline{\text{LUCAS}}/\overline{\text{DQMLU}}$	Output	<ul style="list-style-type: none"> <li>Lower-upper-column address strobe signal for 8-bit DRAM</li> <li>Upper-column address strobe signal for 16-bit DRAM</li> <li>Lower-upper-data mask enable signal for SDRAM</li> <li>Upper-data mask enable signal for 16-bit SDRAM</li> </ul>
Lower-lower-column address strobe/lower-lower-data mask enable	$\overline{\text{LLCAS}}/\overline{\text{DQMLL}}$	Output	<ul style="list-style-type: none"> <li>Lower-lower-column address strobe signal for 8-bit DRAM</li> <li>Lower-column address strobe signal for 16-bit DRAM</li> <li>Column address strobe signal for 8-bit SDRAM</li> <li>Lower-lower-data mask enable signal for SDRAM</li> <li>Lower-data mask enable signal for 16-bit SDRAM</li> <li>Data mask enable signal for 8-bit SDRAM</li> </ul>



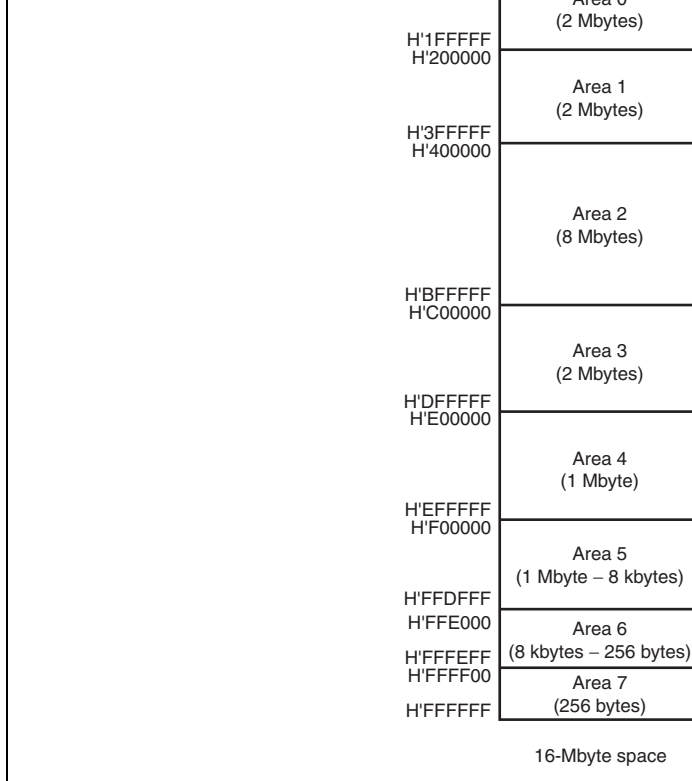
Bus request output	$\overline{\text{BREQ0}}$	Output	External bus request signal used when internal master accesses external address space in external-bus released state
Data transfer acknowledge 3 (DMAC_3)	$\overline{\text{DACK3}}$	Output	Data acknowledge signal for DMAC_3 single address transfer
Data transfer acknowledge 2 (DMAC_2)	$\overline{\text{DACK2}}$	Output	Data acknowledge signal for DMAC_2 single address transfer
Data transfer acknowledge 1 (DMAC_1)	$\overline{\text{DACK1}}$	Output	Data acknowledge signal for DMAC_1 single address transfer
Data transfer acknowledge 0 (DMAC_0)	$\overline{\text{DACK0}}$	Output	Data acknowledge signal for DMAC_0 single address transfer
Data transfer acknowledge 3 (EXDMAC_3)	$\overline{\text{EDACK3}}$	Output	Data acknowledge signal for EXDMAC_3 single address transfer
Data transfer acknowledge 2 (EXDMAC_2)	$\overline{\text{EDACK2}}$	Output	Data acknowledge signal for EXDMAC_2 single address transfer
Data transfer acknowledge 1 (EXDMAC_1)	$\overline{\text{EDACK1}}$	Output	Data acknowledge signal for EXDMAC_1 single address transfer
Data transfer acknowledge 0 (EXDMAC_0)	$\overline{\text{EDACK0}}$	Output	Data acknowledge signal for EXDMAC_0 single address transfer
External bus clock	$\text{B}\phi$	Output	External bus clock

CS6	—	—	—	0	0	0	—	—	0	0	—	—	—	—
CS7	—	—	—	0	0	0	—	—	0	0	—	—	—	—
BS	—	—	—	0	0	0	0	0	0	0	0	0	0	0
RD/WR	—	—	—	0	0	0	0	0	0	0	0	0	0	0
AS	Output	Output	—	0	0	0	0	0	—	—	—	—	—	—
AH	—	—	—	—	—	—	—	—	0	0	—	—	—	—
RD	Output	Output	—	0	0	0	0	0	0	0	0	0	—	—
LHWR/LUB	Output	Output	—	0	—	0	0	—	0	—	—	—	—	—
LLWR/LLB	Output	Output	—	0	0	0	0	0	0	0	—	—	—	—
RAS	—	—	—	—	—	—	—	—	—	—	0	0	0	0
CAS	—	—	—	—	—	—	—	—	—	—	—	—	0	0
WE	—	—	—	—	—	—	—	—	—	—	0	0	0	0
LUCAS/DQMLU	—	—	—	—	—	—	—	—	—	—	0	—	0	—
LLCAS/DQMLL	—	—	—	—	—	—	—	—	—	—	0	0	0	0
OE	—	—	—	—	—	—	—	—	—	—	0	0	—	—
CKE	—	—	—	—	—	—	—	—	—	—	—	—	0	0
WAIT	—	—	—	0	0	0	0	0	0	0	0	0	—	—

[Legend]

O: Used as bus control signal.

—: Not used as bus control signal (I/O port as initial state).



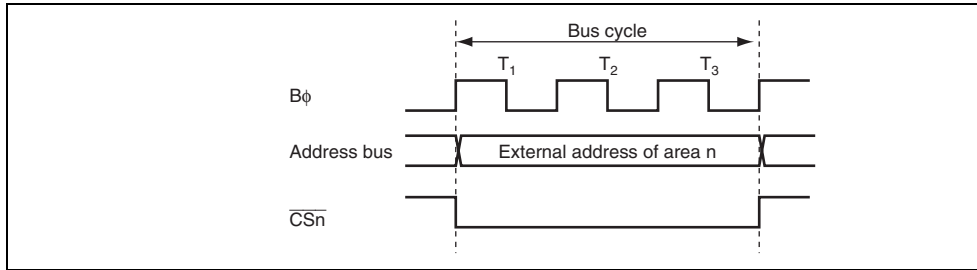
**Figure 9.8 Address Space Area Division**

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 reset and so the corresponding PFCR bits should be set to 1 when outputting signals  $\overline{CS0}$ .

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 9.10 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 9.9**  $\overline{CSn}$  Signal Output Timing ( $n = 0$  to 7)

### 9.5.4 External Bus Interface

The type of the external bus interfaces, bus width, endian format, number of access cycles, 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 not affected by the external bus settings.

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

Six types of external bus interfaces are provided and can be selected in area units. Table 9.4 shows each interface name, description, and area name to be set for each interface. Table 9.5 shows areas that can be specified for each interface. The initial state of each area is a basic bus

**Table 9.4 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 space

Byte control SRAM interface		0	0	0	0	0	0	0
Burst ROM interface	BROMCR	0	0	—	—	—	—	—
Address/data multiplexed I/O interface	MPXCR	—	—	—	0	0	0	0
DRAM interface	DRAMCR	—	—	0	—	—	—	—
Synchronous DRAM interface		—	—	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 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 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 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

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

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

**(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.

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

**(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.

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

### (e) DRAM Interface

In the DRAM interface, the numbers of precharge cycles, row address output cycles, and address output cycles can be specified.

The number of precharge cycles can be specified as one to four cycles by bits TPC1 and TPC0 in DRACCR. The number of row address output cycles can be specified as one to four cycles by bits RCD1 and RCD0 in DRACCR. The number of column address output cycles can be specified as two or three cycles by the CAST bit in DRAMCR. For the column address output cycle, program wait (0 to 7 cycles) specified by WTCRB or external wait by  $\overline{\text{WAIT}}$  can be inserted.

Number of access cycles in the DRAM interface

= number of precharge cycles (1 to 4) + number of row address output cycles (1 to 4)  
+ number of column address output cycles (2 or 3)  
+ number of program wait cycles (0 to 7)  
[+number of external wait cycles by the WAIT pin]

### (f) SDRAM Interface

In the SDRAM interface, the numbers of precharge cycles, row address output cycles, and column address output cycles, as well as clock suspend and write-precharge delay, can be specified by bits DRACCR and WTCRB.

The number of precharge cycles can be specified as one to four cycles by bits TPC1 and TPC0 in DRACCR. The number of row address output cycles can be specified as one to four cycles by bits RCD1 and RCD0 in DRACCR. The number of column address output cycles during read can be specified as two to four cycles by bits W21 and W20 in WTCRB.

The cycles for clock suspend and write-precharge delay can be inserted by bits CKSPE and TRWL in SDCR.



Byte-control SRAM interface		=	Th	+T1	+T2	[0,1]	[1]	[1]	+Tt	[0,1]				
		=	Th	+T1	+T2	[0,1]	[1]	[1]	+Tpw	+Ttw	+T3	+Tt	[0,1]	
		=	Th	+T1	+T2	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]		
Burst ROM interface		=	Th	+T1	+T2							+Tb		
		=	Th	+T1	+T2							+Tb	[(1 to 8) x m] [(2 to 11) x m]	
		=	Th	+T1	+T2	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]		[(1 to 8) x m] [(2 to 11) x m]	
Address/data multiplexed I/O interface		=	Tma	+Th	+T1	+T2						+Tt		
		=	Tma	+Th	+T1	+T2	[2,3]	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]
		=	Tma	+Th	+T1	+T2	[2,3]	[0,1]	[1]	[1]	[0 to 7]	[n]	[1]	[0,1]
DRAM interface	Full access	=	Tp	+Tr	+Trw	+TC1			+Tpw	+Ttw	+Tc2	+Tc3		
		=	[1 to 4]	[1]	[0 to 3]	[1]			[0 to 7]	[n]	[1]	[0,1]		
	Fast page	=				TC1			+Tpw	+Ttw	+Tc2	+Tc3		
		=				[1]			[0 to 7]	[n]	[1]	[0,1]		
	Refresh	=	TRp	+TRw	+TRr	+TRc1			+TRcw	+TRc2				
		=	[1 to 4]	[0 to 3]	[1]	[1]			[0 to 7]	[1]				
SDRAM interface	Setting mode register	=	Tp	+Tr	+Trw	+Tc1						+Tc2	+Trwl	
		=	[1 to 4]	[1]	[0 to 3]	[1]						[1]	[0,1]	
	Full access (read)	=	Tp	+Tr	+Trw	+Tc1				+Tcl	+Tsp	+Tc2		
		=	[1 to 4]	[1]	[0 to 3]	[1]				[1 to 3]	[0,1]	[1]		
	Full access (write)	=	Tp	+Tr	+Trw	+Tc1						+Tc2	+Trwl	
		=	[1 to 4]	[1]	[0 to 3]	[1]						[1]	[0,1]	
	Page access (read)	=				Tc1				+Tcl	+Tsp	+Tc2		
		=				[1]				[1 to 3]	[0,1]	[1]		
	Page access (write)	=				Tc1						+Tc2		
		=				[1]						[1]		
	Cluster transfer (read)	=	Tp	+Tr	+Trw	+Tc1			+Tcb	+Tcl		+Tc2		
		=	[1 to 4]	[1]	[0 to 3]	[1]			[0 to 31]	[1 to 3]		[1]		
	=				Tc1			+Tcb	+Tcl		+Tc2			
	=				[1]			[0 to 31]	[1 to 3]		[1]			
Cluster transfer (write)	=	Tp	+Tr	+Trw	+Tc1						+Tc2	+Tcb		
	=	[1 to 4]	[1]	[0 to 3]	[1]						[1]	[0 to 31]		
	=				Tc1						+Tc2	+Tcb		
	=				[1]						[1]	[0 to 31]		
Refresh	=	TRp	+TRr	+TRc1	+TRcw			+TRc2						
	=	[1 to 4]	[1]	[1]	[0 to 7]			[1]						
Self-refresh	=	TRp	+TRr	+ Software standby mode	+TRc2	+TRc3	+TRp							
	=	[1 to 4]	[1]	[1+s]	[1]	[1]	[0 to 7]							

[Legend]

Number enclosed by bracket: Number of access cycles

n: Pin wait (0 to ∞)

m: Number of burst accesses (0 to 63)

s: Time for a transition to or from software standby mode

## 9.5.5 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 9.7 shows the external interface of area 0.

**Table 9.7 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

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  $\overline{\text{CS2}}$  signal can be output.

Either the basic bus interface, byte-control SRAM interface, DRAM interface, or SDRAM interface can be selected for area 2 by the DRAME and DTYPE bits in DRAMCR and by BCSEL2 in SRAMCR. Table 9.9 shows the external interface of area 2.

**Table 9.9 Area 2 External Interface**

Interface	Register Setting		
	DRAME in DRAMCR	DTYPE in DRAMCR	BCSEL2 in SRAMCR
Basic bus interface	0	Don't care	0
Byte-control SRAM interface	0	Don't care	1
DRAM interface	1	0	0
SDRAM interface	1	1	0
Setting prohibited	1	Don't care	1

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 interface can be selected for area 4 by bit MPXE4 in MPXCR and bit BCSEL4 in SRAMCR. Table 9.11 shows the external interface of area 4.

**Table 9.11 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

interface can be selected for area 5 by the MPXE5 bit in MPXCR and the BCSEL5 bit in SRAMCR. Table 9.12 shows the external interface of area 5.

**Table 9.12 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 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 interface can be selected for area 6 by the MPXE6 bit in MPXCR and the BCSEL6 bit in SRAMCR. Table 9.13 shows the external interface of area 6.

## (8) Area 7

Area 7 includes internal I/O registers. In external extended mode, area 7 other than internal 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 interface can be selected for area 7 by the MPXE7 bit in MPXCR and the BCSEL7 bit in SRAMCR. Table 9.14 shows the external interface of area 7.

**Table 9.14 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

amount of data that can be accessed at one time is one byte: a word access is performed by four byte accesses, and a longword access, as four byte accesses.

Figures 9.11 and 9.12 illustrate data alignment control for the 8-bit access space. Figure 9.11 shows the data alignment when the data endian format is specified as big endian. Figure 9.12 shows the data alignment when the data endian format is specified as little endian.

Data Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe s
					LHWR/LUB
					Data b
					D15 D8
Byte	n	1	1st	Byte	
Word	n	2	1st	Byte	
			2nd	Byte	
Longword	n	4	1st	Byte	
			2nd	Byte	
			3rd	Byte	
			4th	Byte	

**Figure 9.11 Access Sizes and Data Alignment Control for 8-Bit Access Space (Big Endian)**

			2nd	Byte	15
			3rd	Byte	23
			4th	Byte	31

**Figure 9.12 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 word.

Figures 9.13 and 9.14 illustrate data alignment control for the 16-bit access space. Figure 9.13 shows the data alignment when the data endian format is specified as big endian. Figure 9.14 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. 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. byte access for an odd address is performed by using the third byte data bus.



Longword	Even (2n)	2	1st	Word	31   15   7   0
			2nd	Word	15   7   0
	Odd (2n+1)	3	1st	Byte	7   0
			2nd	Word	23   15   7   0
			3rd	Byte	7   0

**Figure 9.13 Access Sizes and Data Alignment Control for 16-Bit Access Space (Big Endian)**

Access Size	Access Address	Access Count	Bus Cycle	Data Size	Strobe s LHWR/LUB	Data D15 D8
Byte	Even (2n)	1	1st	Byte		
	Odd (2n+1)	1	1st	Byte	7   0	
Word	Even (2n)	1	1st	Word	15   7   0	
	Odd (2n+1)	2	1st	Byte	7   0	
			2nd	Byte		
Longword	Even (2n)	2	1st	Word	15   7   0	
			2nd	Word	31   15   7   0	
	Odd (2n+1)	3	1st	Byte	7   0	
			2nd	Word	23   15   7   0	
			3rd	Byte		

**Figure 9.14 Access Sizes and Data Alignment Control for 16-Bit Access Space (Little Endian)**

of lower byte data bus (D7 to D0) is used according to the bus specifications for the area 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 9.5.6, Endian and Data Alignment.

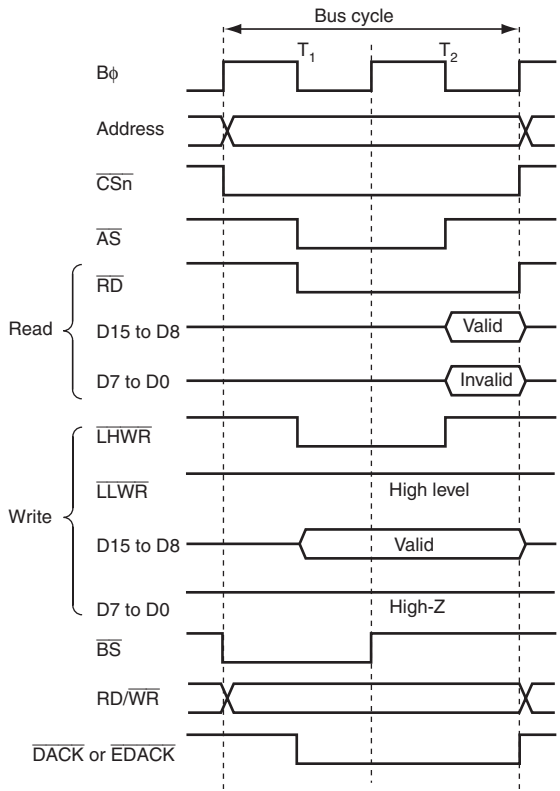
## 9.6.2 I/O Pins Used for Basic Bus Interface

Table 9.15 shows the pins used for basic bus interface.

**Table 9.15 I/O Pins for Basic Bus Interface**

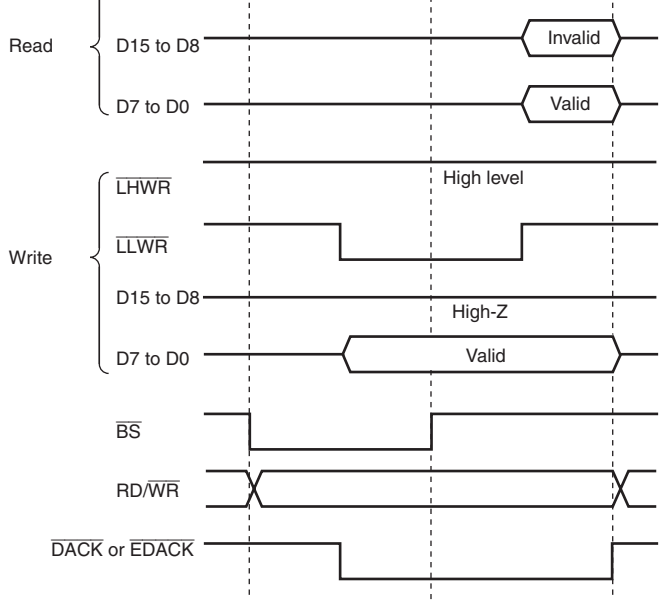
Name	Symbol	I/O	Function
Bus cycle start	$\overline{BS}$	Output	Signal indicating that the bus cycle has started
Address strobe	$\overline{AS}^*$	Output	Strobe signal indicating that an address output on the address bus is valid during access
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Read/write	$RD/\overline{WR}$	Output	Signal indicating the data bus input or output direction
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D7 to D0) 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	$CS0$ to $CS7$	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  $\overline{AS}$  output and does not function as the  $\overline{AS}$  output.



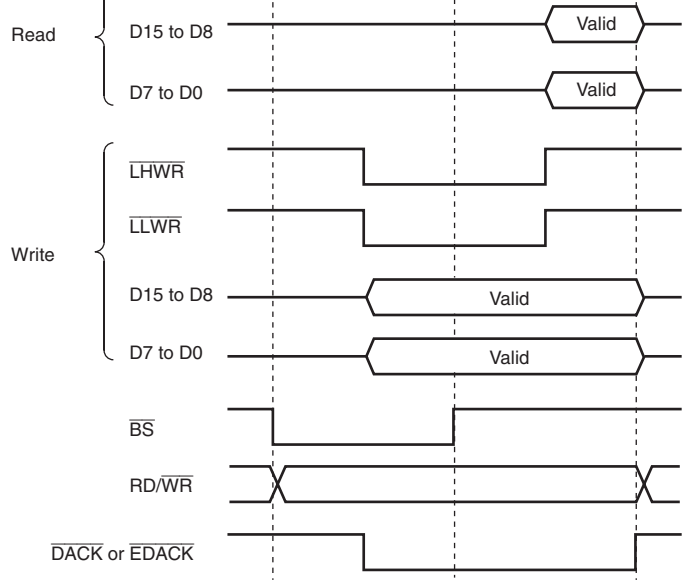
- Notes: 1.  $n = 0$  to 7  
 2. When  $RDNn = 0$   
 3. When  $DKC$  and  $EDKC = 0$

**Figure 9.15 16-Bit 2-State Access Space Bus Timing (Byte Access for Even Address)**



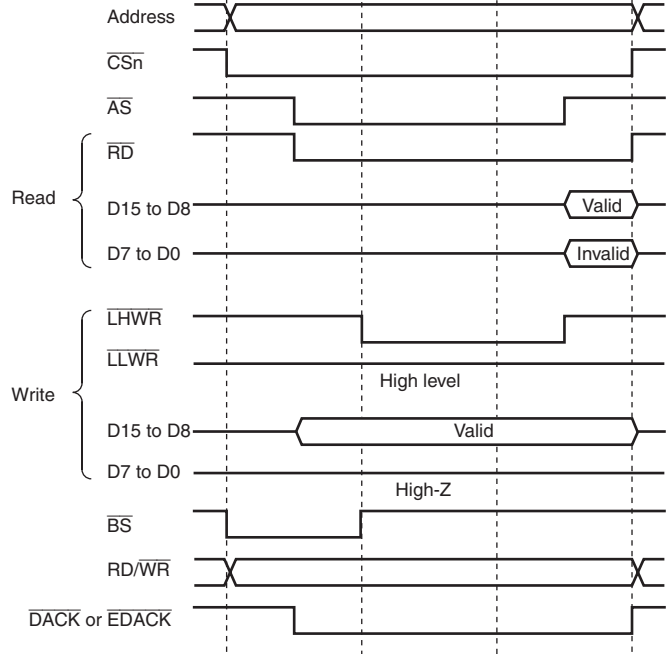
- Notes: 1. n = 0 to 7  
 2. When RDn = 0  
 3. When DKC and EDKC = 0

**Figure 9.16 16-Bit 2-State Access Space Bus Timing (Byte Access for Odd Address)**



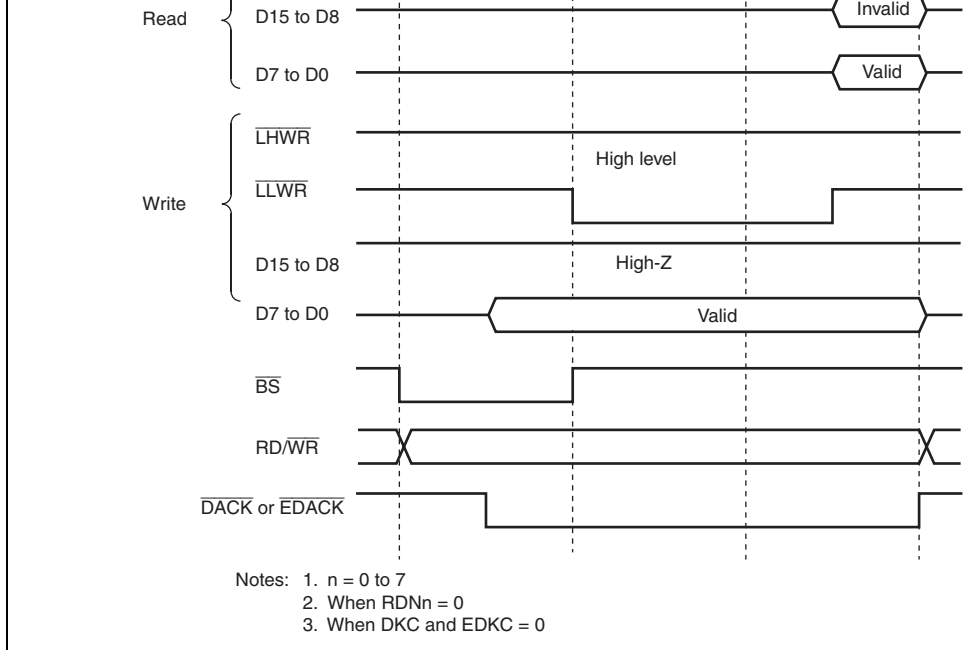
- Notes: 1.  $n = 0$  to 7  
 2. When  $RDNn = 0$   
 3. When  $DKC$  and  $EDKC = 0$

**Figure 9.17 16-Bit 2-State Access Space Bus Timing (Word Access for Even A**

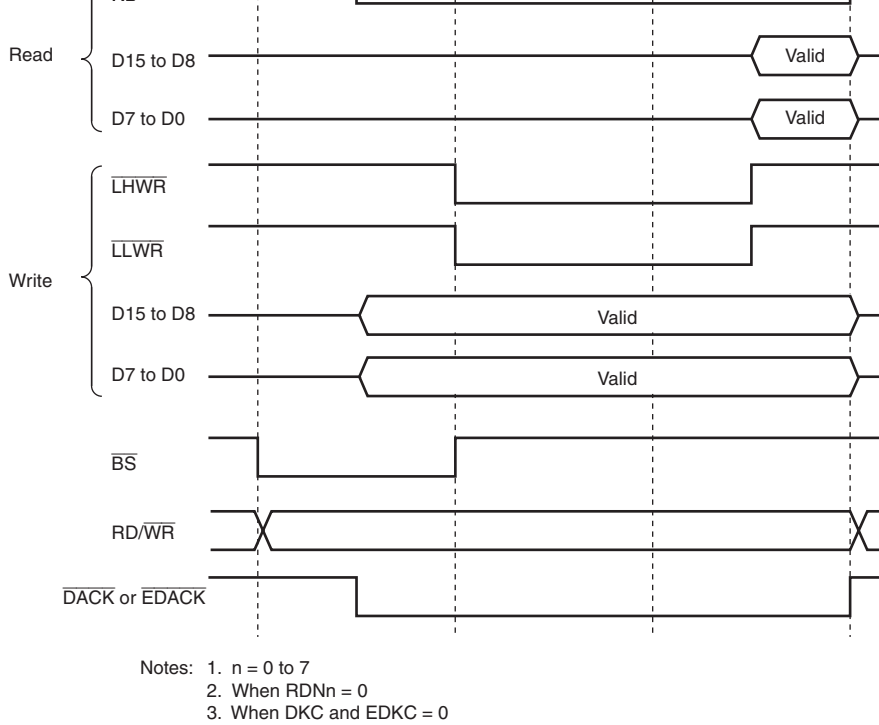


- Notes: 1. n = 0 to 7  
 2. When RDNn = 0  
 3. When DKC and EDKC = 0

**Figure 9.18 16-Bit 3-State Access Space Bus Timing (Byte Access for Even Address)**



**Figure 9.19 16-Bit 3-State Access Space Bus Timing (Word Access for Odd Address)**

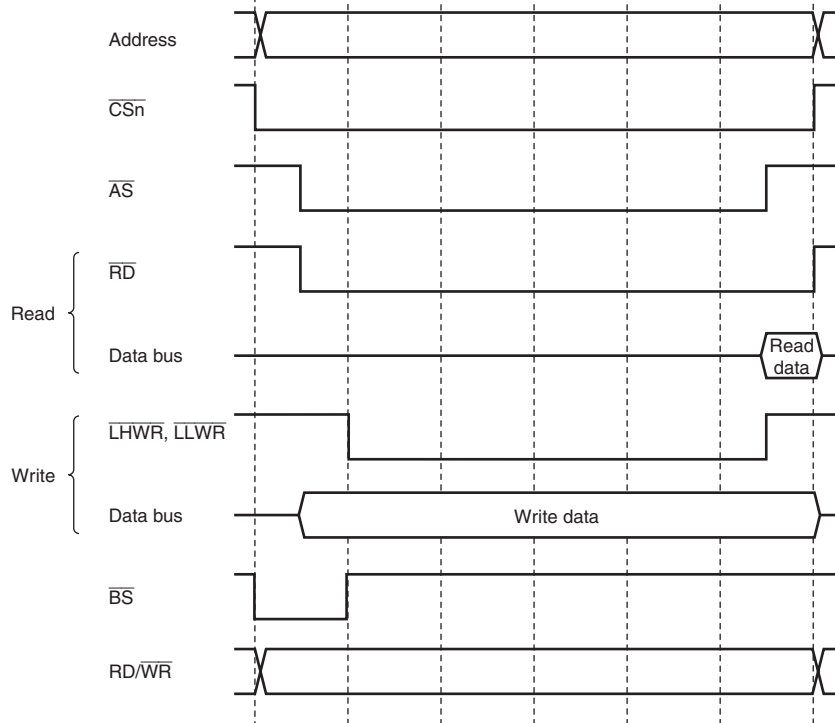


**Figure 9.20 16-Bit 3-State Access Space Bus Timing (Word Access for Even Ad**



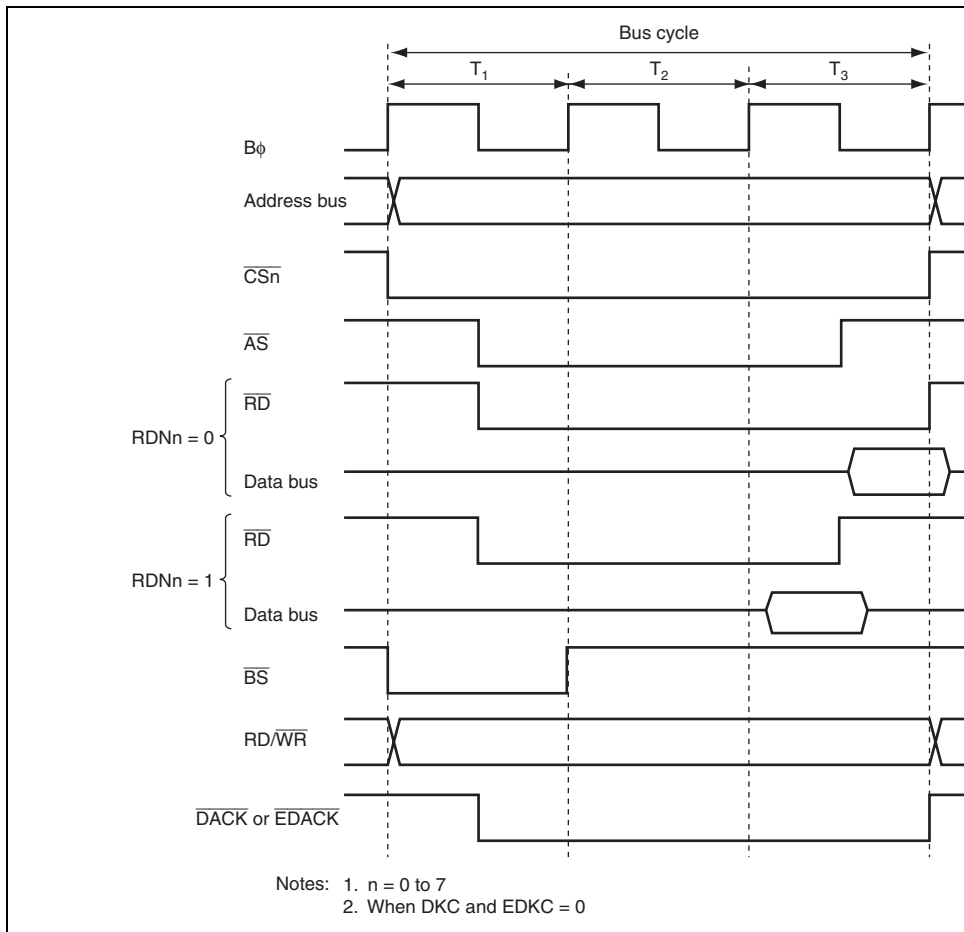
## (2) Pin Wait Insertion

For 3-state access space, when the WAITE bit in BCR1 is set to 1 and the corresponding  $\overline{\text{WAIT}}$  pin is set to 1, wait input by means of the  $\overline{\text{WAIT}}$  pin is enabled. When the external address is accessed in this state, a program wait (Tpw) is first inserted according to the WTCRA and WTCRB settings. If the  $\overline{\text{WAIT}}$  pin is low at the falling edge of  $\Phi$  in the last T2 or Tpw, another Ttw cycle is inserted until the  $\overline{\text{WAIT}}$  pin is brought high. The pin wait insertion is effective when the Tw cycles are inserted to seven cycles or more, or when the number of 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 13, I/O Ports.



- Notes: 1. Upward arrows indicate the timing of  $\overline{\text{WAIT}}$  pin sampling.  
 2.  $n = 0$  to 7  
 3. When  $\text{RD}n = 0$

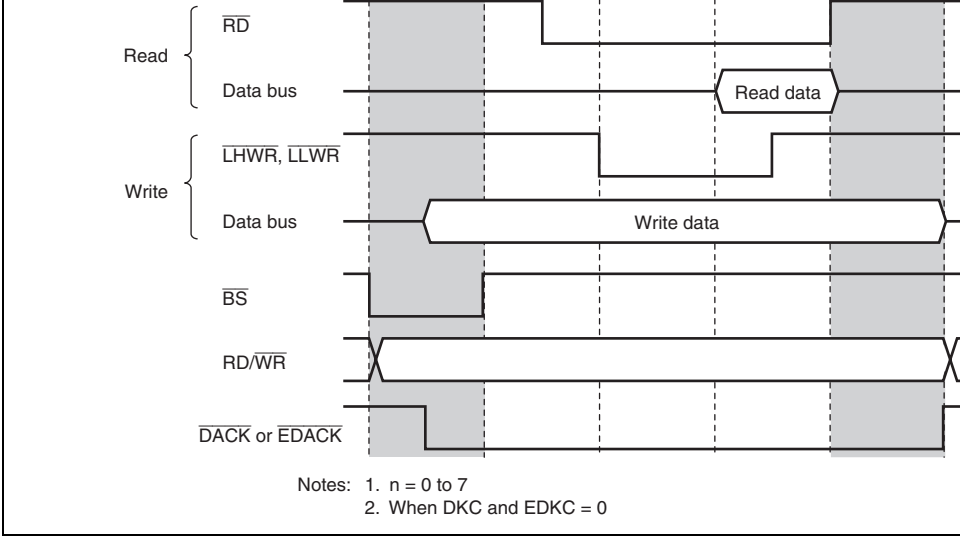
**Figure 9.21 Example of Wait Cycle Insertion Timing**



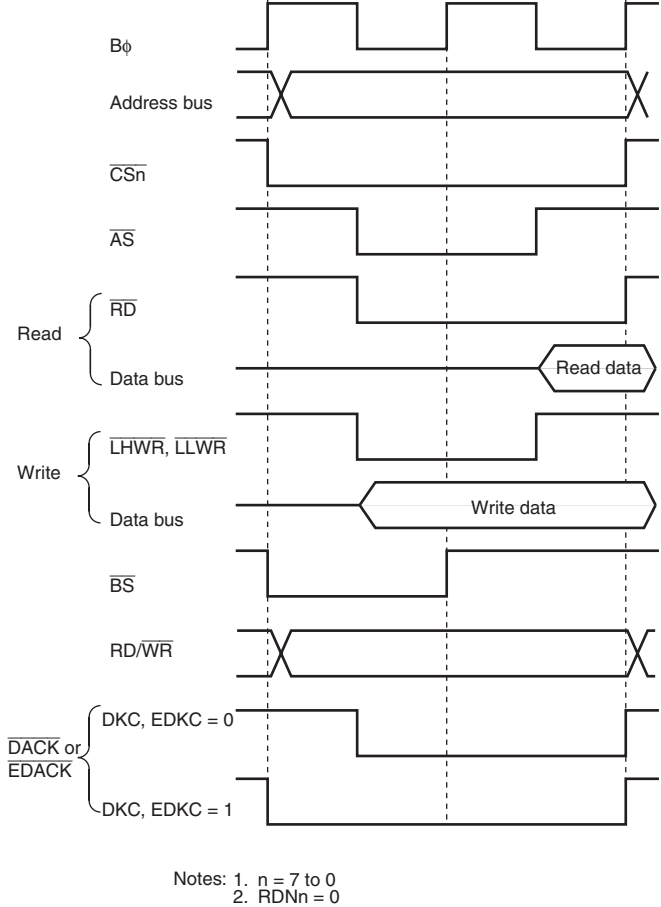
**Figure 9.22 Example of Read Strobe Timing**

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 of extension cycle  $T_h$  or  $T_t$  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 9.23 Example of Timing when Chip Select Assertion Period is Extended**



**Figure 9.24**  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  Signal Output Timing

### 9.7.1 Byte Control SRAM Space Setting

Byte control SRAM interface can be specified for areas 0 to 7. Each area can be specified as burst ROM interface or address/data multiplexed I/O interface, the SRAMCR setting and byte control SRAM interface cannot be used.

### 9.7.2 Data Bus

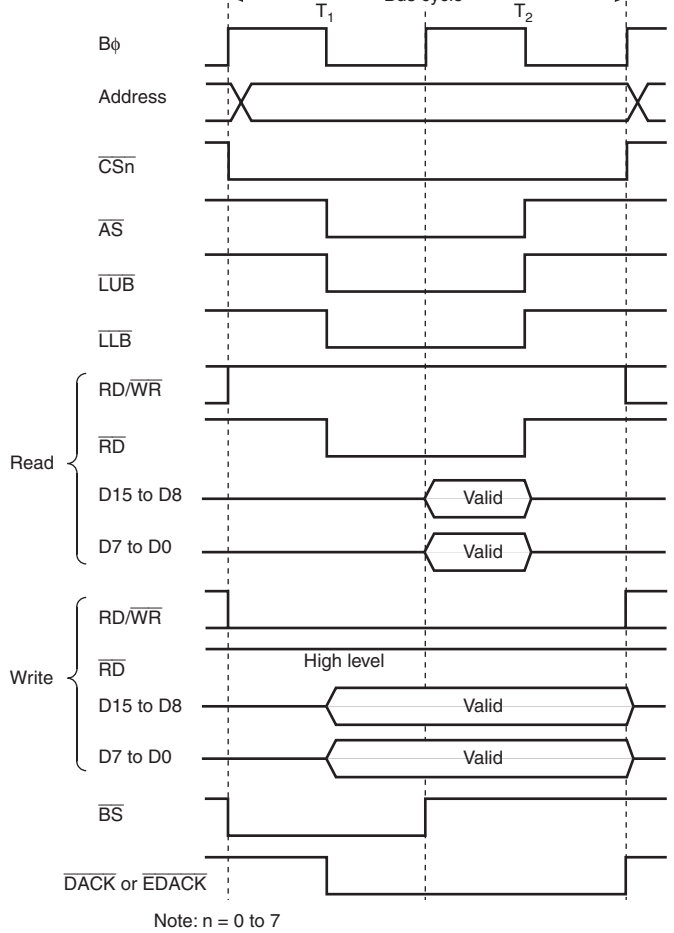
The bus width of the byte control SRAM space can be specified as 16-bit byte control SRAM space according to bits ABWH<sub>n</sub> and ABWL<sub>n</sub> (n = 0 to 7) in ABWCR. The area specified as burst ROM access space cannot be specified as the byte control SRAM space.

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

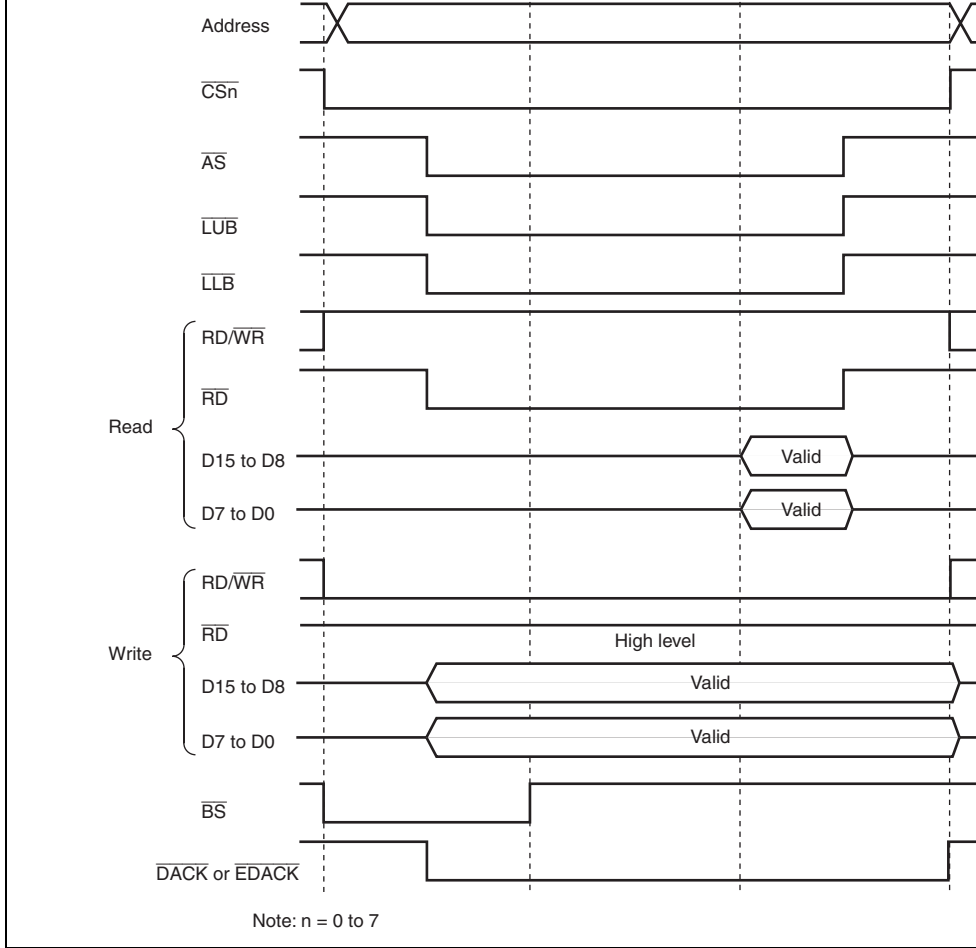
Access size and data alignment are the same as the basic bus interface. For details, see section 9.5.6, Endian and Data Alignment.

AS/AH	AS	Address strobe	Output	Strobe signal indicating that the address output on the address bus is valid when the 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 address control SRAM space is accessed
$\overline{\text{RD}}/\overline{\text{WR}}$	$\overline{\text{RD}}/\overline{\text{WR}}$	Read/write	Output	Write enable signal for the SRAM when the address byte control SRAM space is accessed
$\overline{\text{LHWR}}/\overline{\text{LUB}}$	$\overline{\text{LUB}}$	Lower-upper byte select	Output	Upper byte select when the 16-bit address control SRAM space is accessed
$\overline{\text{LLWR}}/\overline{\text{LLB}}$	$\overline{\text{LLB}}$	Lower-lower byte select	Output	Lower byte select when the 16-bit address control SRAM space is accessed
$\overline{\text{WAIT}}$	$\overline{\text{WAIT}}$	Wait	Input	Wait request signal used when an address space is accessed
A23 to A0	A23 to A0	Address pin	Output	Address output pin
D15 to D0	D15 to D0	Data pin	Input/ output	Data input/output pin





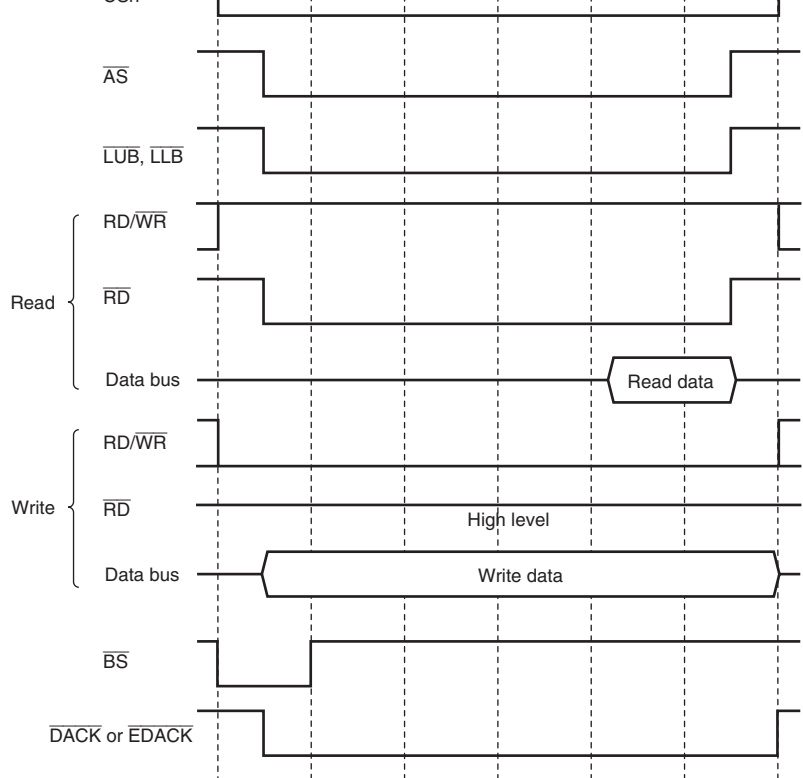
**Figure 9.25 16-Bit 2-State Access Space Bus Timing**



**Figure 9.26 16-Bit 3-State Access Space Bus Timing**

For 3-state access space, when the WAITE bit in BCR1 is set to 1, the corresponding D<sub>0</sub> 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, refer to section 13, I/O Ports.

Figure 9.27 shows an example of wait cycle insertion timing.



- Notes: 1. Upward arrows indicate the timing of  $\overline{\text{WAIT}}$  pin sampling.  
 2.  $n = 0$  to 7

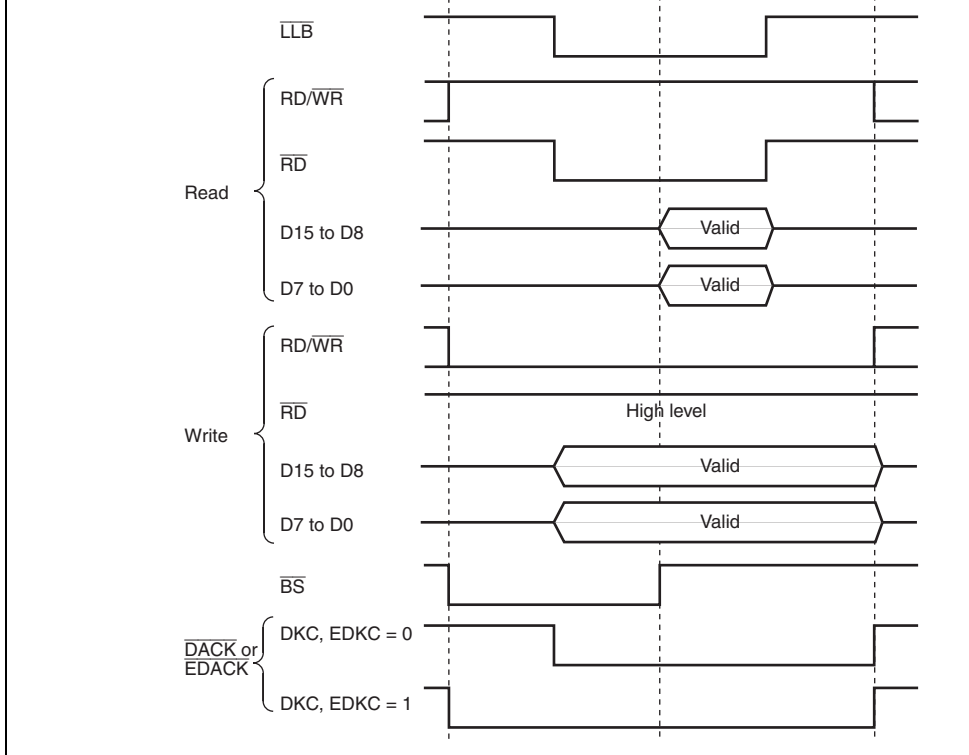
**Figure 9.27 Example of Wait Cycle Insertion Timing**

In the byte control DMA interface, the extension cycles can be inserted before and after a cycle in the same way as the basic bus interface. For details, refer to section 9.6.6, Extended Chip Select ( $\overline{CS}$ ) Assertion Period.

### 9.7.8 $\overline{DACK}$ and $\overline{EDACK}$ Signal Output Timing

For DMAC or EXDMAC single address transfers, the  $\overline{DACK}$  and  $\overline{EDACK}$  signal assertion can be modified by using the DKC and EDKC bits in BCR1.

Figure 9.28 shows the  $\overline{DACK}$  and  $\overline{EDACK}$  signal output timing. Setting the DKC bit or EDKC bit to 1 asserts the  $\overline{DACK}$  or  $\overline{EDACK}$  signal a half cycle earlier.



**Figure 9.28**  $\overline{\text{DACK}}$  Signal Output Timing

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

In the burst ROM interface, the burst access covers only read accesses by the CPU and L2 cache cluster transfer. Other accesses are performed with the similar method to the basic bus interface.

### **9.8.1 Burst ROM Space Setting**

Burst ROM interface can be specified for areas 0 and 1. Areas 0 and 1 can be specified and burst ROM space by setting bits BSRM<sub>n</sub> (n = 0, 1) in BROMCR.

### **9.8.2 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 ABWH<sub>n</sub> and ABWL<sub>n</sub> bits (n = 0, 1) in ABWCR.

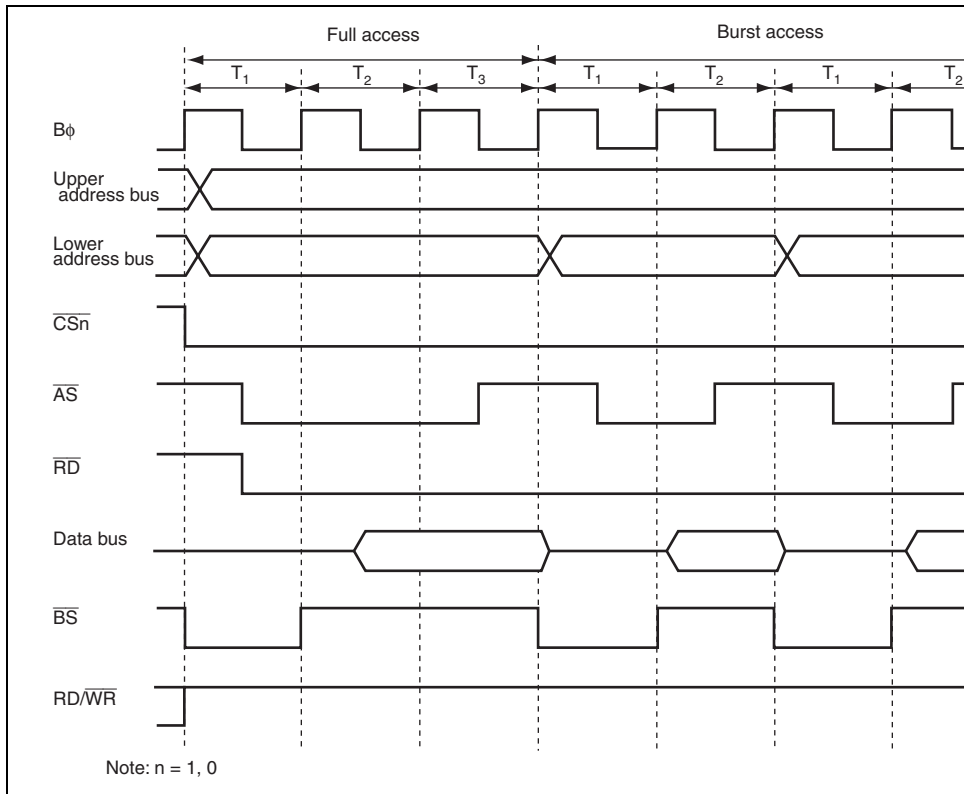
For the 8-bit bus width, data bus (D7 to D0) is valid. For the 16-bit bus width, data bus (D15 to D0) is valid.

Access size and data alignment are the same as the basic bus interface. For details, see section 9.5.6, Endian and Data Alignment.

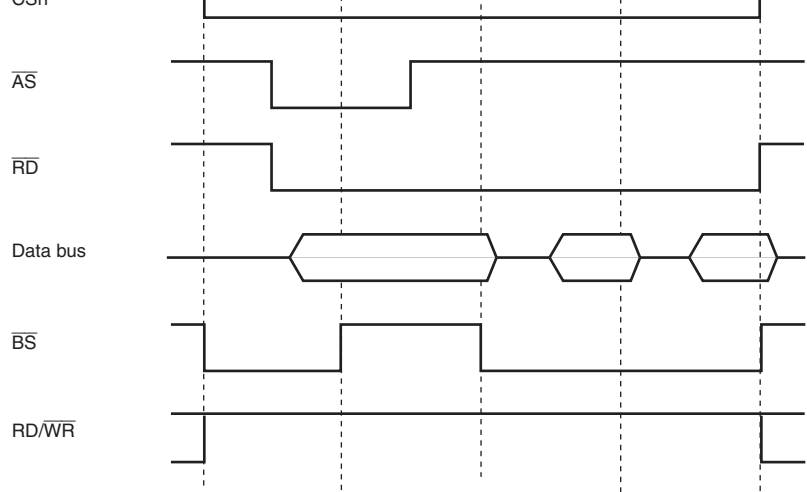
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Read/write	$RD/\overline{WR}$	Output	Signal indicating the data bus input or output direction
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D8) is valid during write access
Low-low write	$\overline{LLWR}$	Output	Strobe signal indicating that the lower byte (D0) is valid during write access
Chip select 0 and 1	$\overline{CS0}, \overline{CS1}$	Output	Strobe signal indicating that the area is selected
Wait	$\overline{WAIT}$	Input	Wait request signal used when an external address space is accessed



The basic access timing for burst ROM space is shown in figures 9.29 and 9.30.



**Figure 9.29 Example of Burst ROM Access Timing ( $ASTn = 1$ , Two Burst Cycles)**



Note:  $n = 1, 0$

**Figure 9.30 Example of Burst ROM Access Timing ( $AST_n = 0$ , One Burst Cycle)**

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

### 9.8.7 Extension of Chip Select ( $\overline{CS}$ ) 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 and EXDMAC cluster transfer. In this case, the setting of the corresponding CSXTn bit in the EXDMAC register is ignored and an extension cycle can be inserted only before the full access cycle. Note that an extension cycle can be inserted before or after the burst access cycles.

In accesses other than read accesses by the CPU and EXDMAC cluster transfer, 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.

specified as the address/data multiplexed I/O space by setting bits MPXEn (n = 3 to 7) in MPXCR.

### 9.9.2 Address/Data Multiplex

In the address/data multiplexed I/O space, data bus is multiplexed with address bus. Table 9.18 shows the relationship between the bus width and address output.

**Table 9.18 Address/Data Multiplex**

Bus Width	Cycle	Data Pins															
		PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0	PH7	PH6	PH5	PH4	PH3	PH2	PH1	
8 bits	Address	-	-	-	-	-	-	-	-	-	A7	A6	A5	A4	A3	A2	A1
	Data	-	-	-	-	-	-	-	-	-	D7	D6	D5	D4	D3	D2	D1
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

### 9.9.3 Data Bus

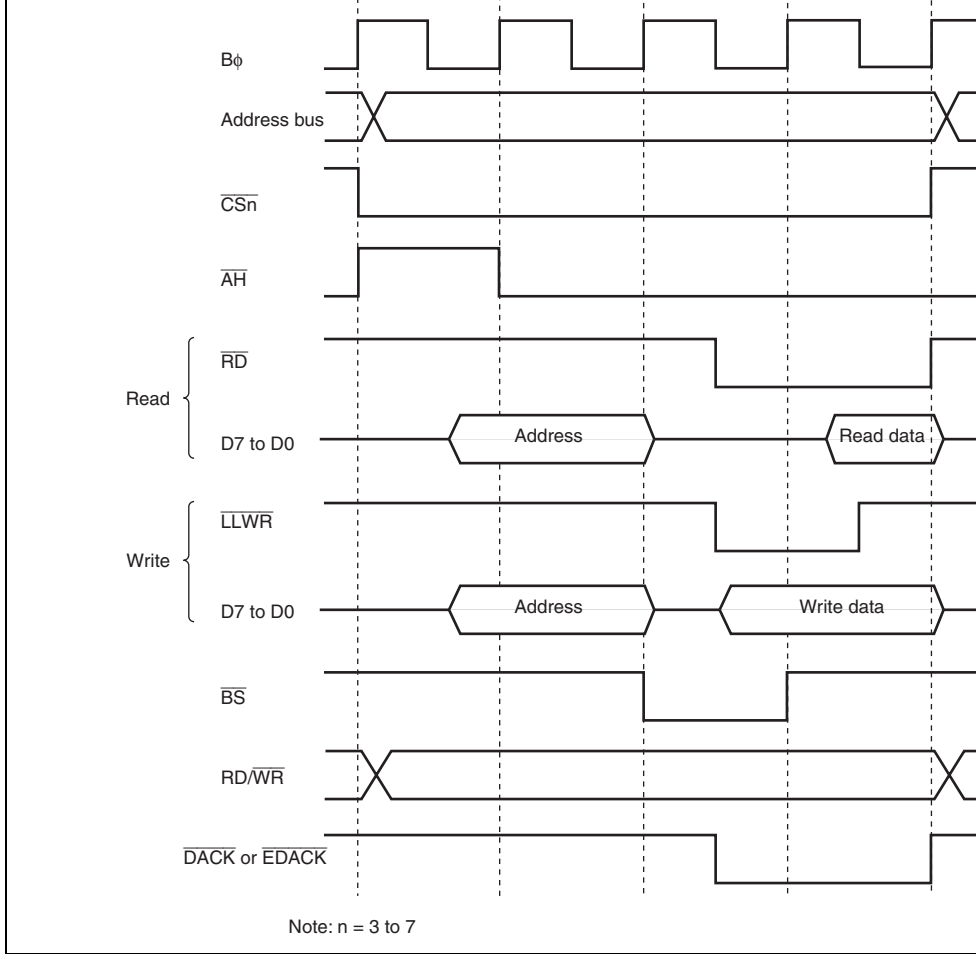
The bus width of the address/data multiplexed I/O space can be specified for either 8-bit access space or 16-bit access space by the ABWHn and ABWLn bits (n = 3 to 7) in ABWCR.

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.

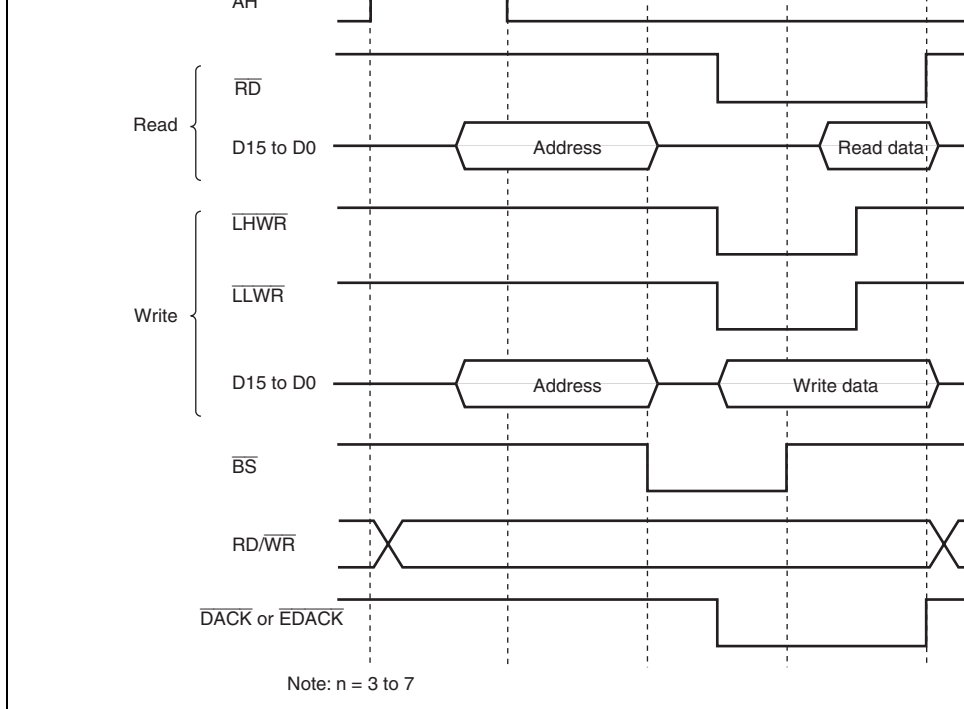
For details on access size and data alignment, see section 9.5.6, Endian and Data Alignment.

$\overline{AS}/\overline{AH}$	$\overline{AH}^*$	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
$\overline{LHWR}/\overline{LUB}$	$\overline{LHWR}$	Low-high write	Output	Strobe signal indicating that the upper byte (D8) is valid when the address/data multiplexed I/O space is written
$\overline{LLWR}/\overline{LLB}$	$\overline{LLWR}$	Low-low write	Output	Strobe signal indicating that the lower byte (D7) 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.
A23 to A0	A23 to A0	Address	Output	Address output pin
$\overline{WAIT}$	$\overline{WAIT}$	Wait	Input	Wait request signal used when the external memory 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

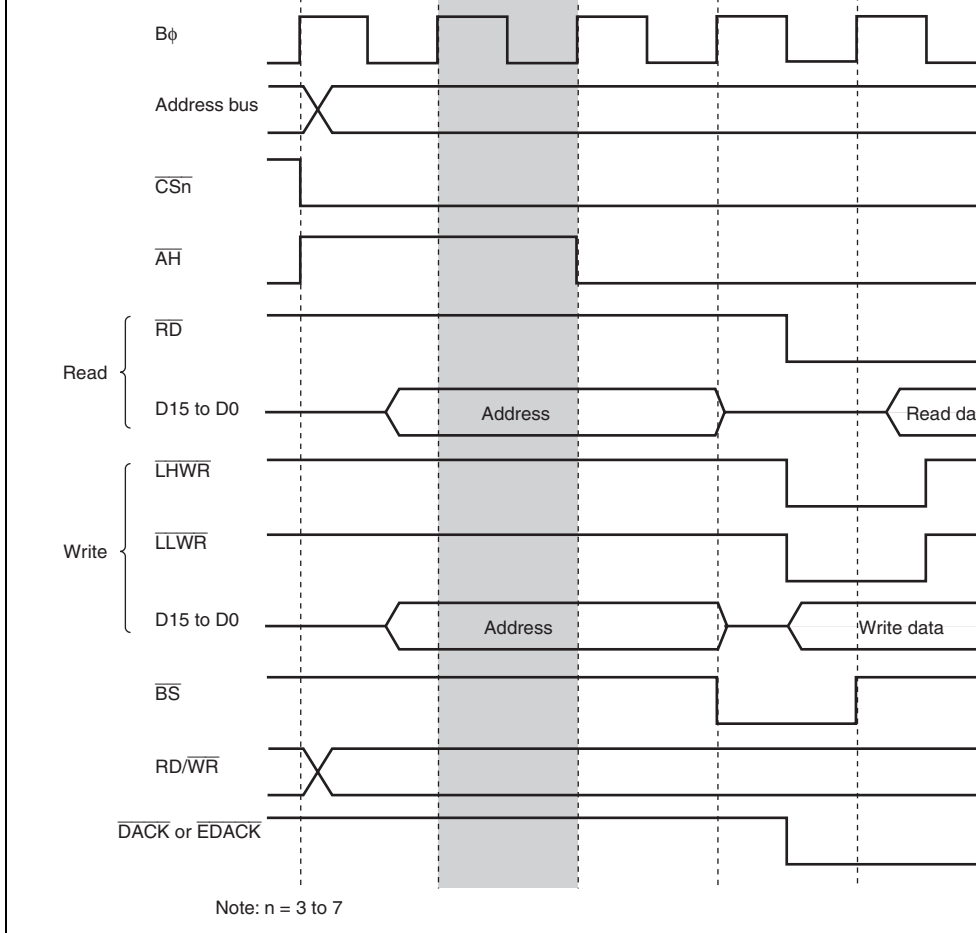
Note: \* The  $\overline{AH}$  output is multiplexed with the  $\overline{AS}$  output. At the timing that an area is accessed as address/data multiplexed I/O, this pin starts to function as the  $\overline{AH}$  output and at this time this pin cannot be used as the  $\overline{AS}$  output. At this time, when other areas are accessed through the basic bus interface is accessed, this pin does not function as the  $\overline{AS}$  output. When an area is specified as address/data multiplexed I/O, be aware that this pin functions as the  $\overline{AS}$  output.



**Figure 9.31 8-Bit Access Space Access Timing (ABWHn = 1, ABWLn = 1)**



**Figure 9.32 16-Bit Access Space Access Timing (ABWHn = 0, ABWLn =**

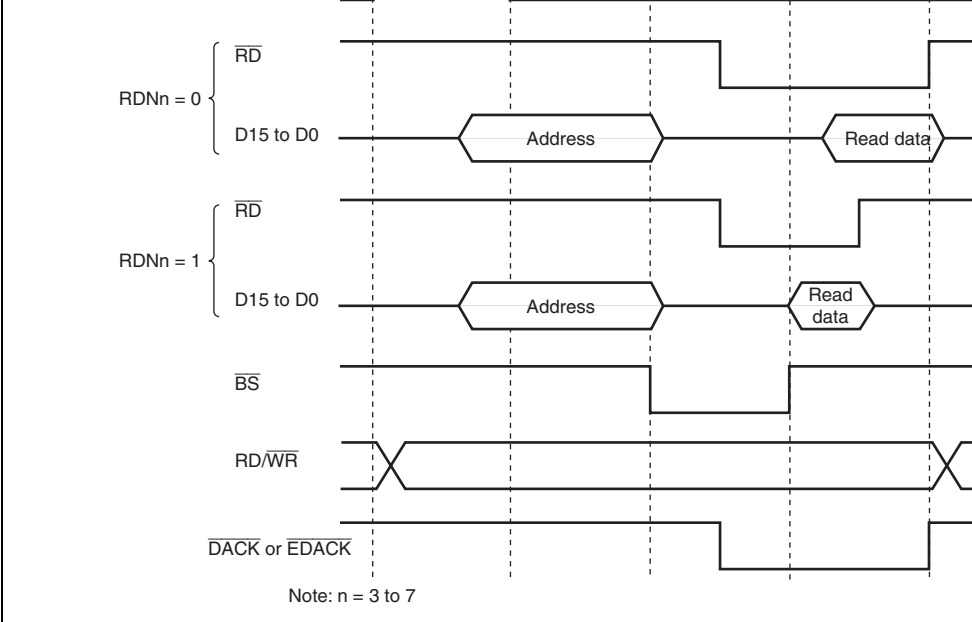


**Figure 9.33 Access Timing of 3 Address Cycles (ADDEX = 1)**

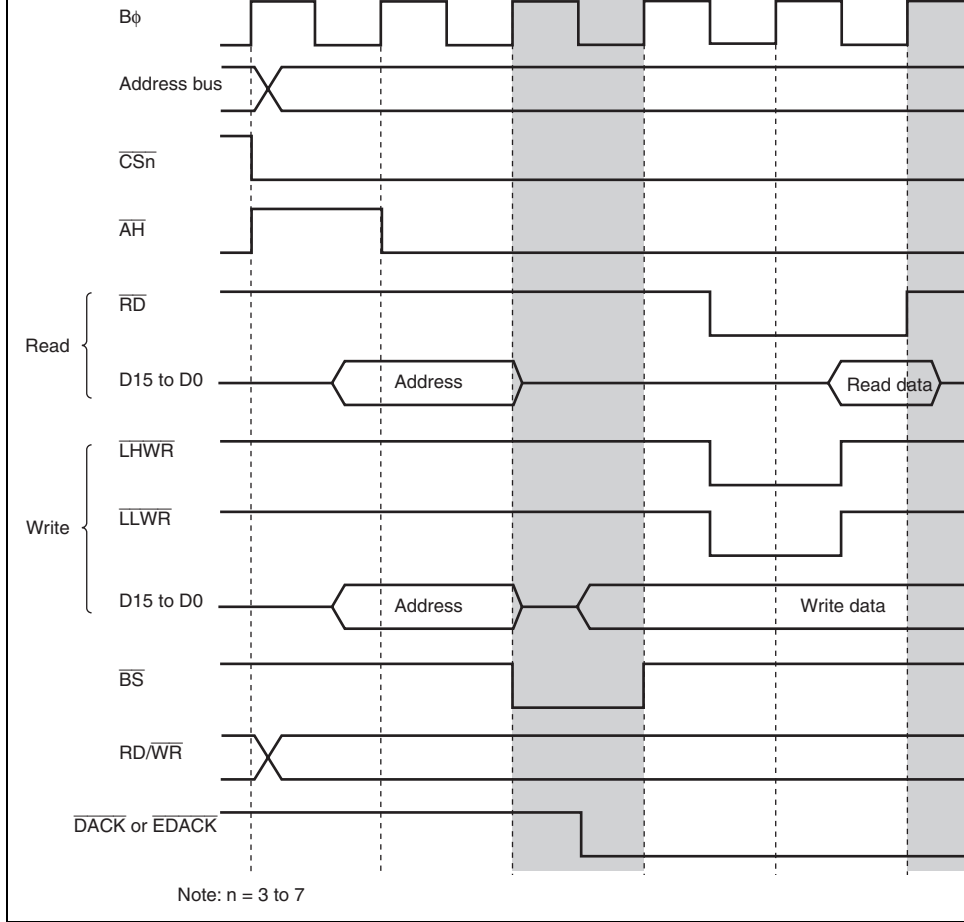


In the address/data multiplexed I/O interface, the read strobe timing of data cycles can be in the same way as in basic bus interface. For details, refer to section 9.6.5, Read Strobe Timing.

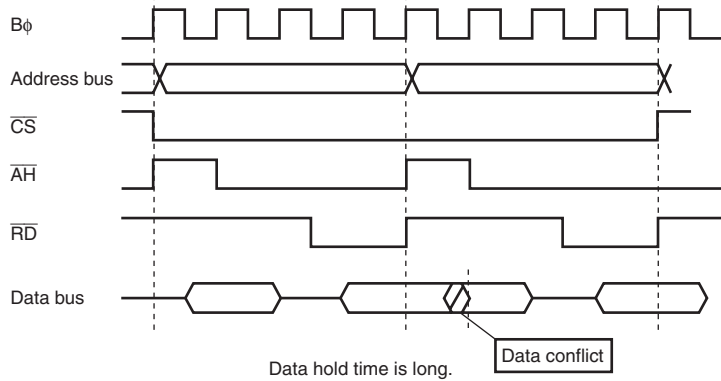
Figure 9.34 shows an example when the read strobe timing is modified.



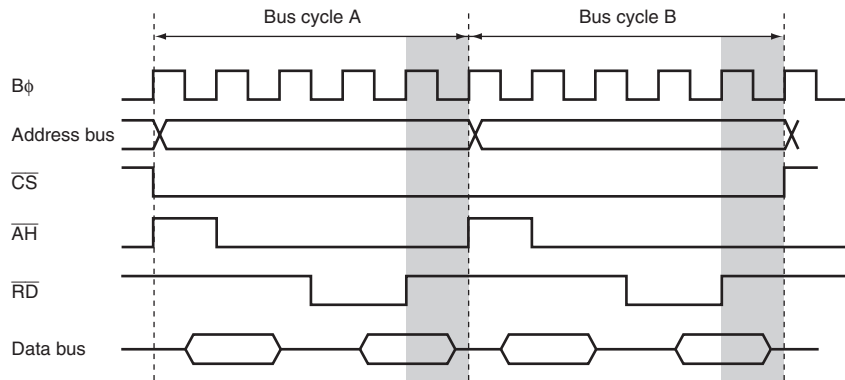
**Figure 9.34 Read Strobe Timing**



**Figure 9.35** Chip Select ( $\overline{CS}$ ) Assertion Period Extension Timing in Data C...

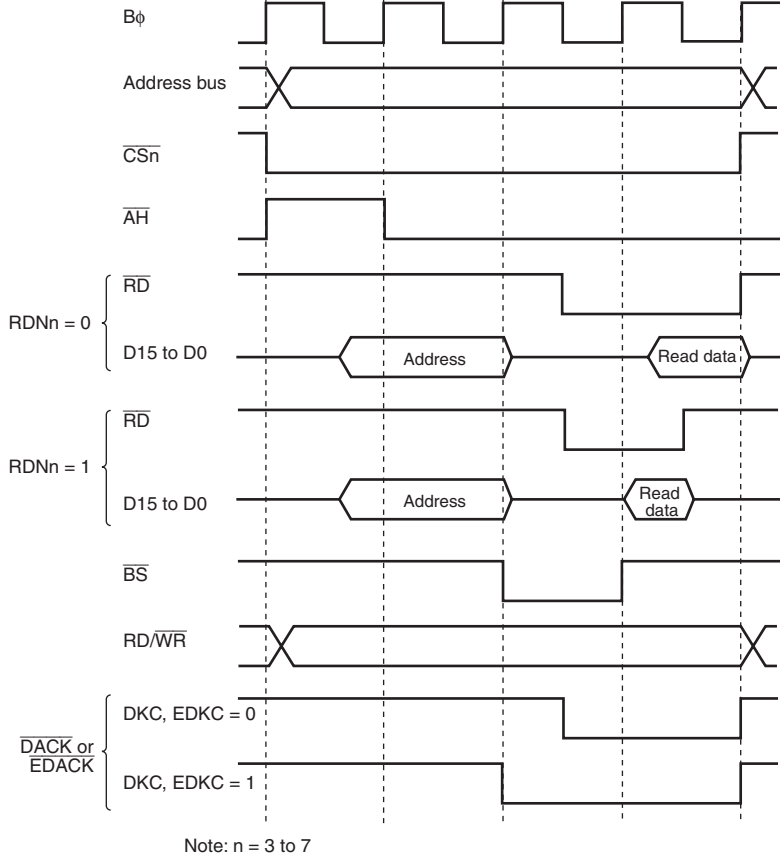


(a) Without  $\overline{CS}$  assertion period extension cycle ( $CSXTn = 0$ )



(b) With  $\overline{CS}$  assertion period extension cycle ( $CSXTn = 1$ )

**Figure 9.36 Consecutive Read Accesses to Same Area  
(Address/Data Multiplexed I/O Space)**



**Figure 9.37  $\overline{DACK}$  and  $\overline{EDACK}$  Signal Output Timing**

settings.

**Table 9.20 Relationship Among DRAME and DTYPE and Area 2 Interfaces**

DRAME	DTYPE	Area 2 Interface
0	×	Basic bus space (initial state)/byte-control SRAM space
1	0	DRAM space
1	1	SDRAM space

[Legend]

×: Don't care

### 9.10.2 Address Multiplexing

A Row address and a column address are multiplexed in the DRAM space. Select the number of row address bits to be shifted with bits MXC1 and MXC0 in DRAMCR. Table 9.21 lists the relationship among bits MXC1 and MXC0 and shifted bit number.

**Table 9.21 Relationship Among MXC1 and MXC0 and Shifted Bit Count**

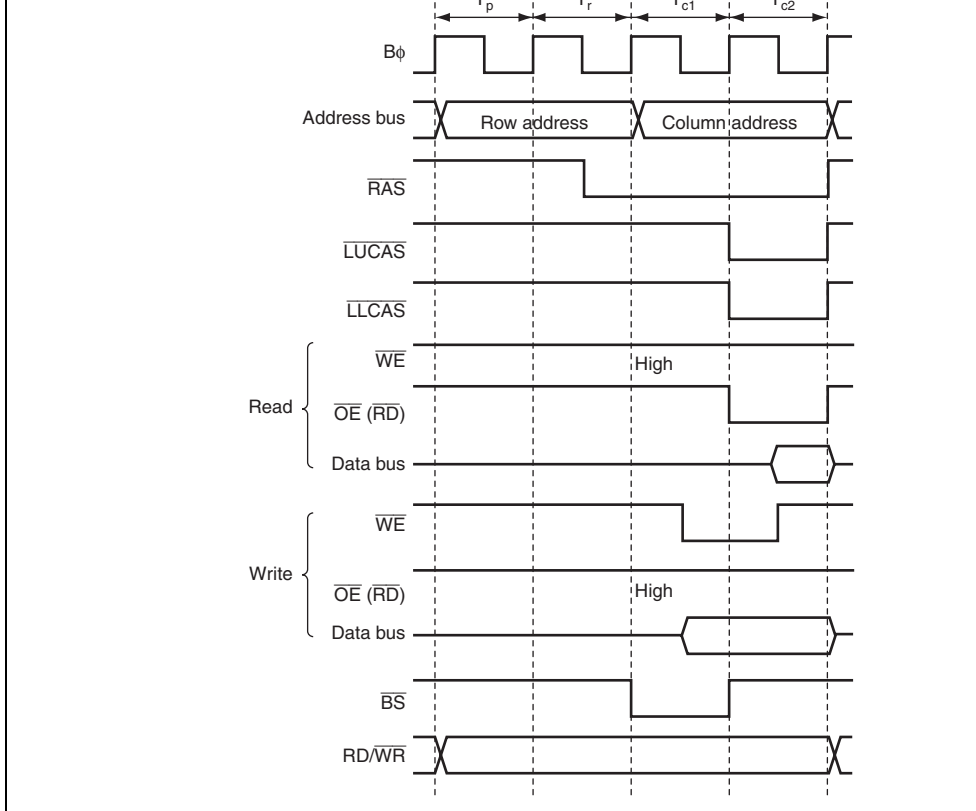
DRAMCR		Shift Bit Count	Data Bus Width	Address	External Address Pin															
MXC1	MXC0				A27 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3
0	0	8 bits	8/16 bits	Row address	A23 to A18	A17	-	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11
				Column address	A23 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3
0	1	9 bits	8/16 bits	Row address	A23 to A18	A17	-	-	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12
				Column address	A23 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3
1	0	10 bits	8/16 bits	Row address	A23 to A18	A17	-	-	-	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13
				Column address	A23 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3
1	1	11 bits	8/16 bits	Row address	A23 to A18	A17	-	-	-	-	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14
				Column address	A23 to A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3

## 9.10.4 I/O Pins Used for DRAM Interface

Table 9.22 shows the pins used for the DRAM interface.

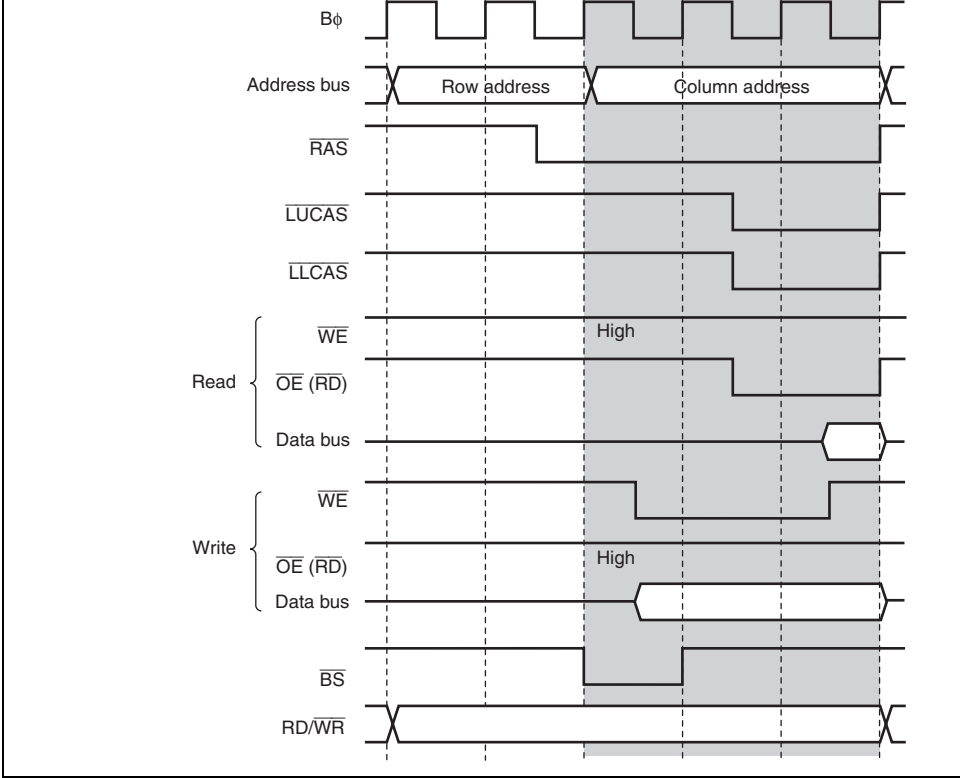
**Table 9.22 I/O Pins for DRAM Interface**

Pin	DRAM Selected	Name	I/O	Function
$\overline{\text{WE}}$	$\overline{\text{WE}}$	Write enable	Output	Write enable signal for accessing DRAM interface
$\overline{\text{RAS}}$	$\overline{\text{RAS}}$	Row address strobe	Output	Row address strobe when the DRAM address space is specified as area 2
$\overline{\text{LUCAS}}/\overline{\text{DQMLU}}$	$\overline{\text{LUCAS}}$	Lower-upper column address strobe	Output	<ul style="list-style-type: none"> <li>Lower-upper column address strobe when the 32-bit DRAM space is accessed</li> <li>Upper column address strobe when the 16-bit DRAM space is accessed</li> </ul>
$\overline{\text{LLCAS}}/\overline{\text{DQMLL}}$	$\overline{\text{LLCAS}}$	Lower-lower column address strobe	Output	<ul style="list-style-type: none"> <li>Lower-lower column address strobe when the 32-bit DRAM space is accessed</li> <li>Lower column address strobe when the 16-bit DRAM space is accessed</li> </ul>
$\overline{\text{OE}}$	$\overline{\text{OE}}$	Output enable	Output	Output enable signal when the DRAM address space is accessed
$\overline{\text{WAIT}}$	$\overline{\text{WAIT}}$	Wait	Input	Wait request signal used when a DRAM address space is accessed
A17 to A0	A17 to A0	Address pin	Output	Multiplexed address/data output
D15 to D0	D15 to D0	Data pin	Input/output	Data input/output pin

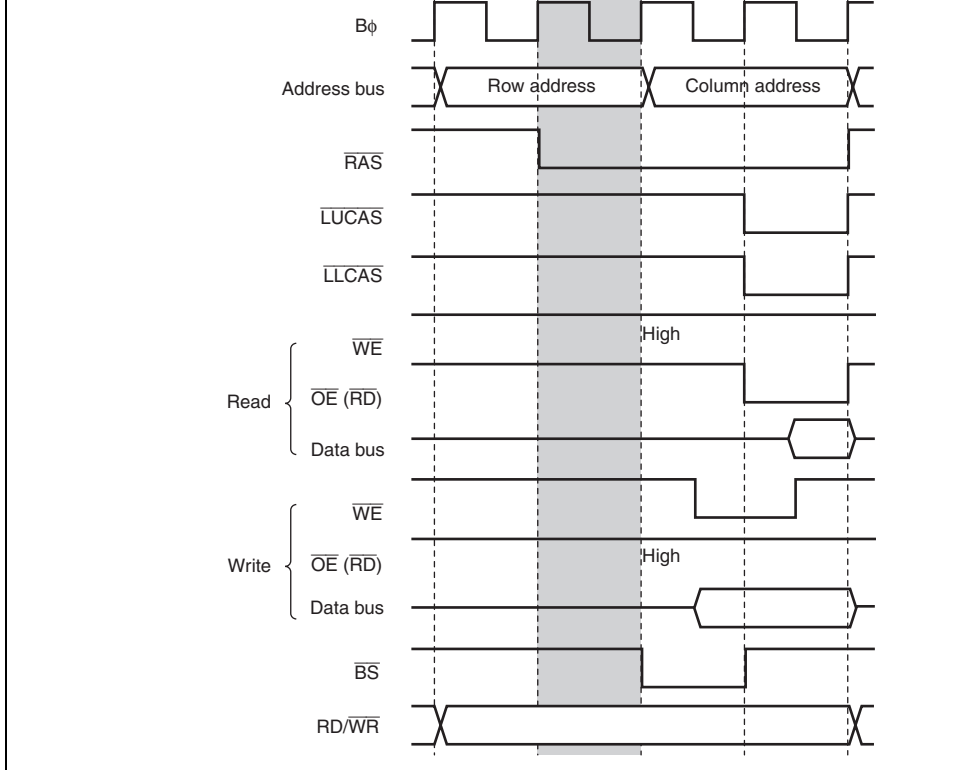


**Figure 9.38 DRAM Basic Access Timing (RAS = 0 and CAST = 0)**

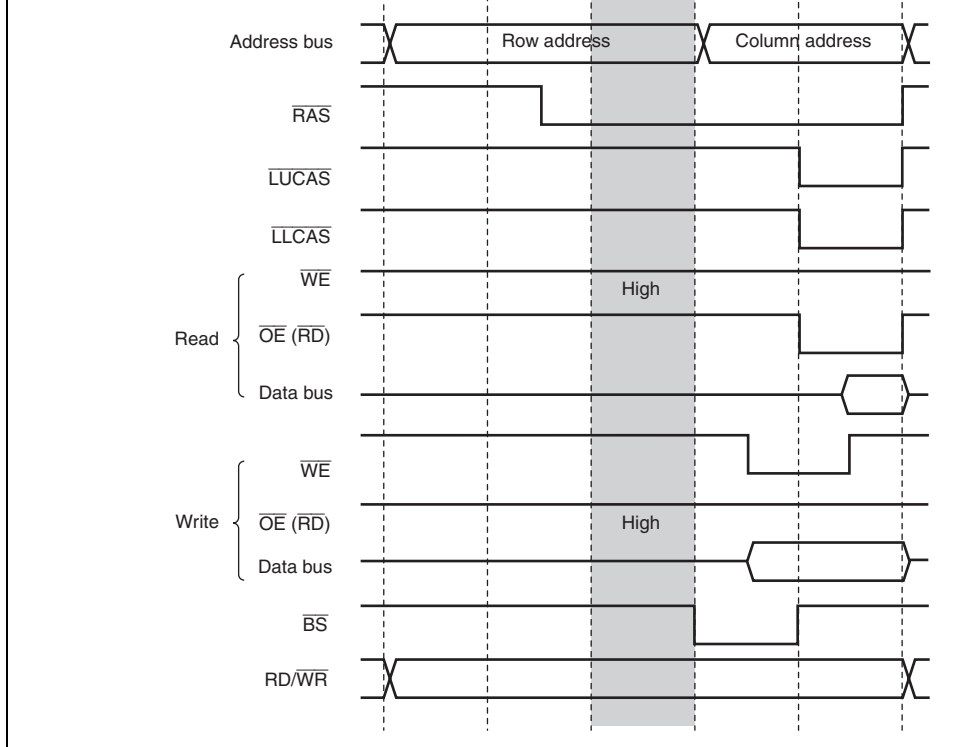




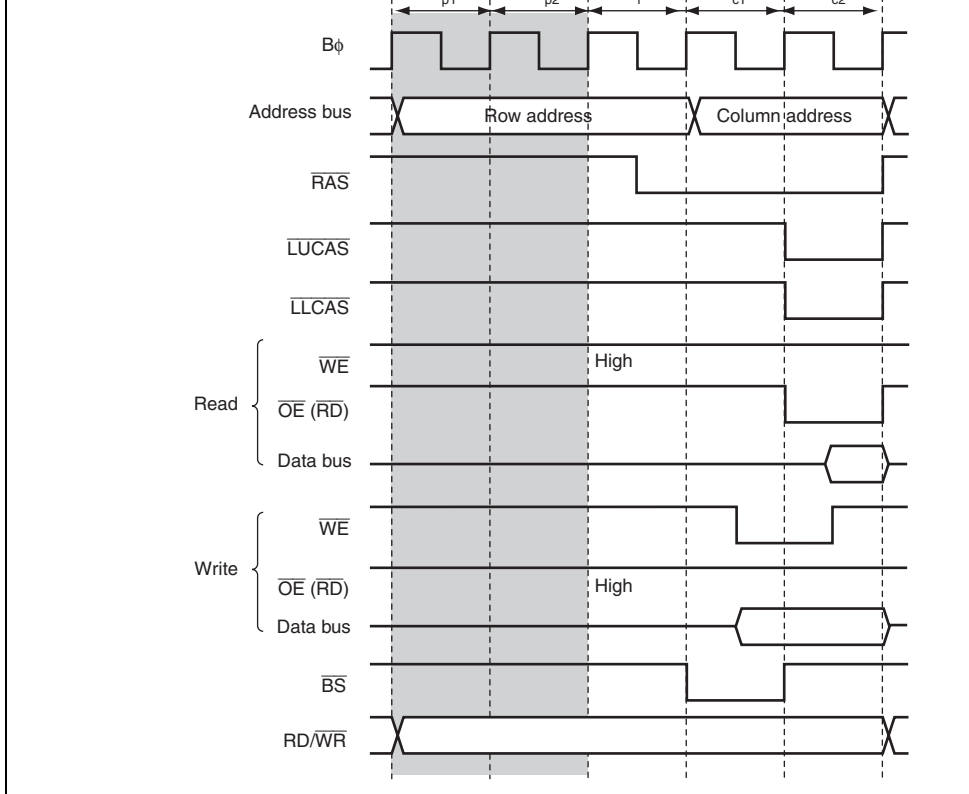
**Figure 9.39 Access Timing Example of Column Address Output Cycles for 3 Cl**  
**(RAST = 0)**



**Figure 9.40 Access Timing Example of  $\overline{\text{RAS}}$  Signal Driven Low at Start of Tr C (CAST = 0)**



**Figure 9.41 Access Timing Example when One Trw Cycle is Specified (RAST=0,**



**Figure 9.42 Access Timing Example of Two Precharge Cycles (RAST = 0 and CA**

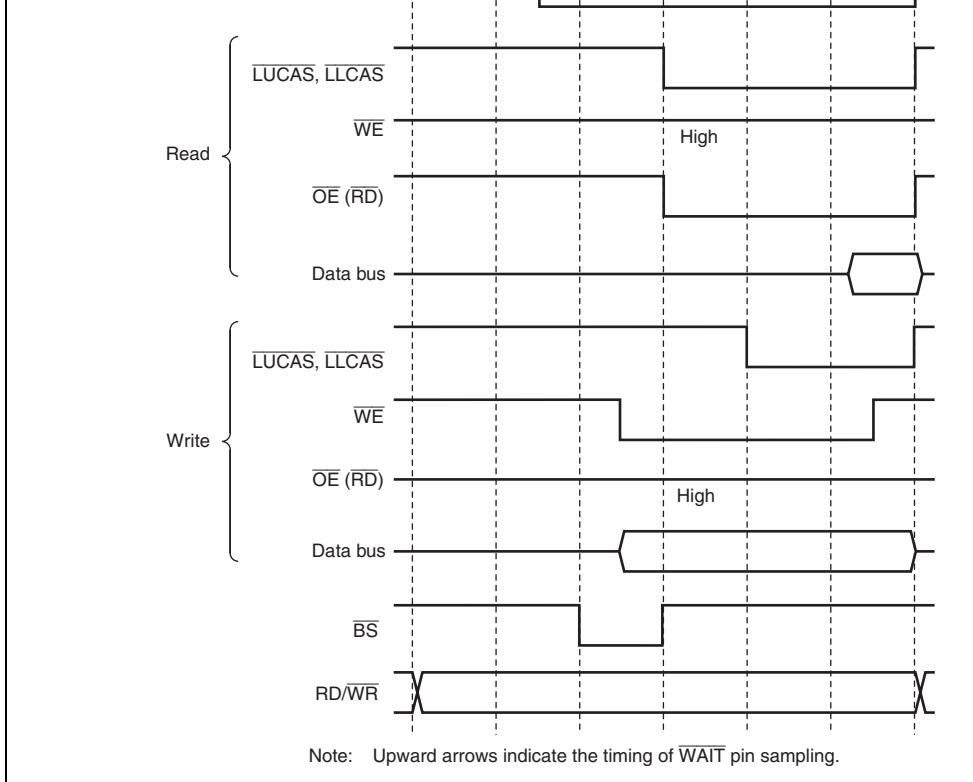
between the  $Tc1$  and  $Tc2$  cycles. The number of wait cycles is selected by bits  $W22$  to  $W20$  in the  $WTCRB$ .

## (2) Pin Wait Insertion

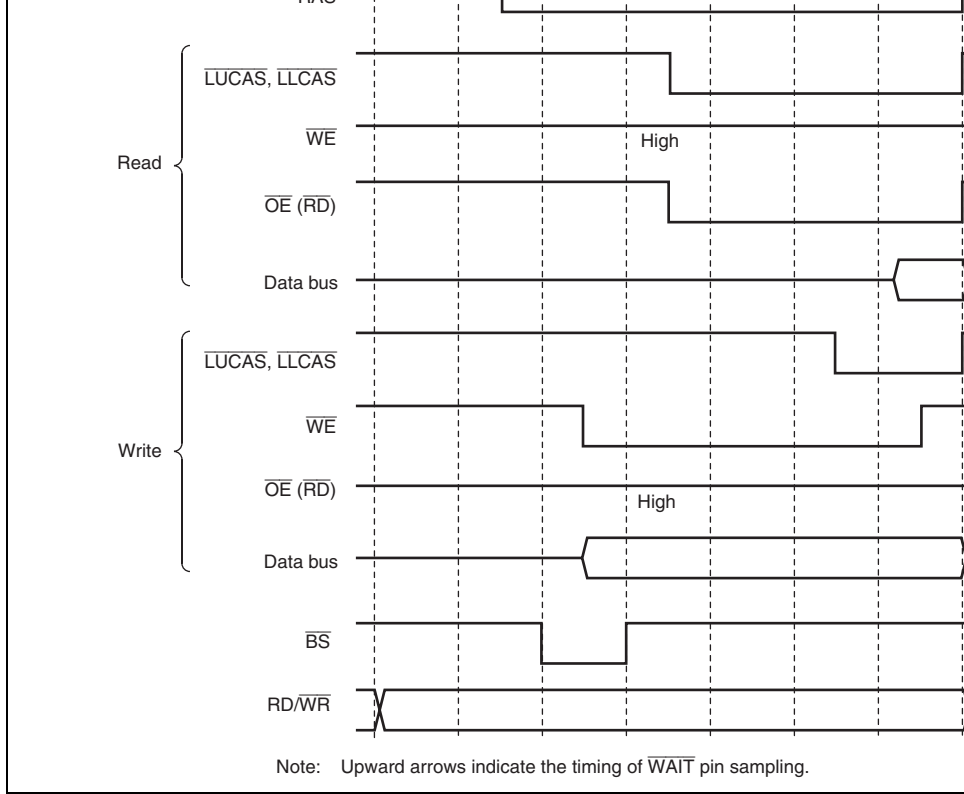
When the  $WAITE$  bit in  $BCR1$  is set to 1, and the  $AST2$  bit in  $ASTCR$  is set to 1, setting the  $WAITEN$  bit for the corresponding pin to 1 enables wait input by the  $\overline{WAIT}$  pin. When the DRAM is accessed in this state, a program wait ( $Tpw$ ) is first inserted. If the  $\overline{WAIT}$  pin is low at the falling edge of  $B\phi$  in the last  $Tc1$  or  $Tpw$  cycle, another  $Ttw$  cycle is inserted until the  $\overline{WAIT}$  pin is high. For details on ICR, see section 13, I/O Ports.

Figure 9.43 shows an example of wait cycle insertion timing for 2-cycle column address access.

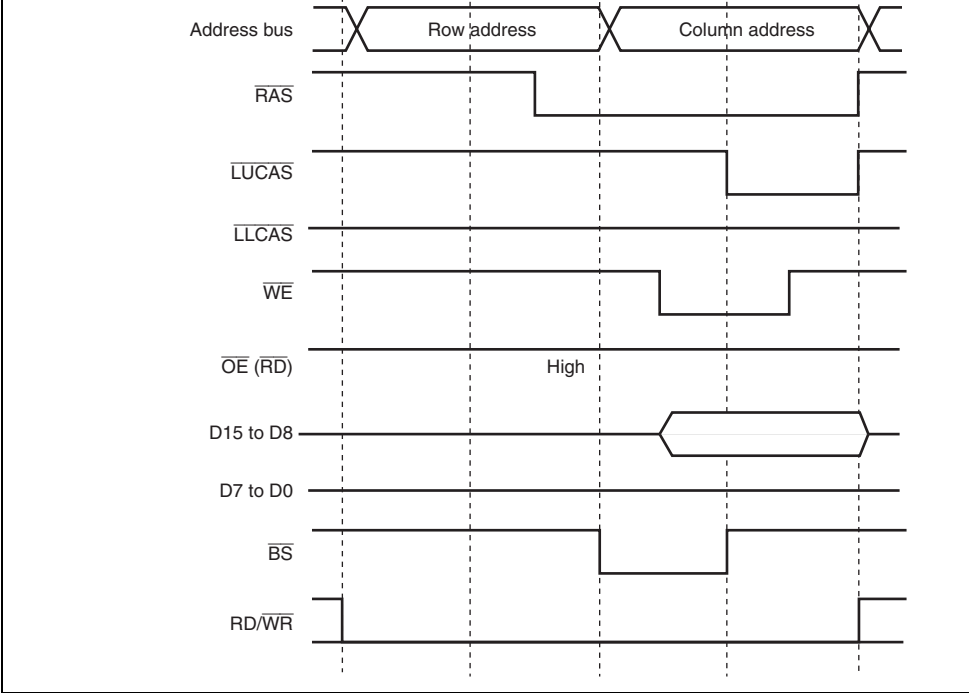
Figure 9.44 shows an example of wait cycle insertion timing for 3-cycle column address access.



**Figure 9.43 Example of Wait Cycle Insertion Timing for 2-Cycle Column Address**

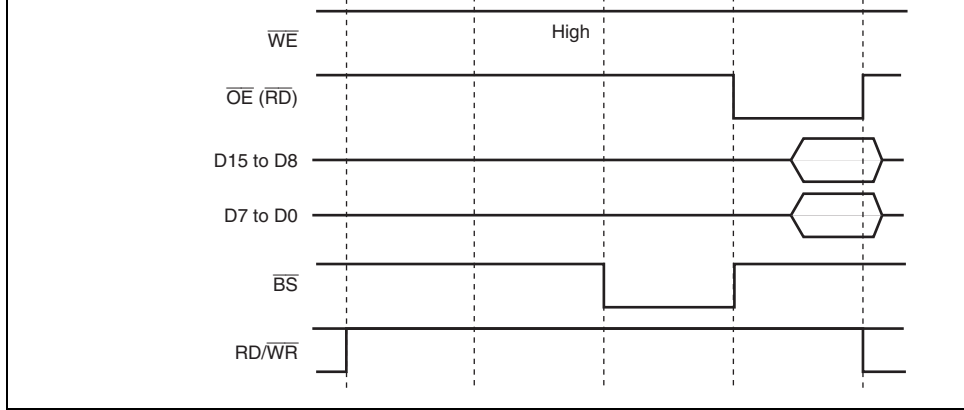


**Figure 9.44 Example of Wait Cycle Insertion Timing for 3-Cycle Column Address**

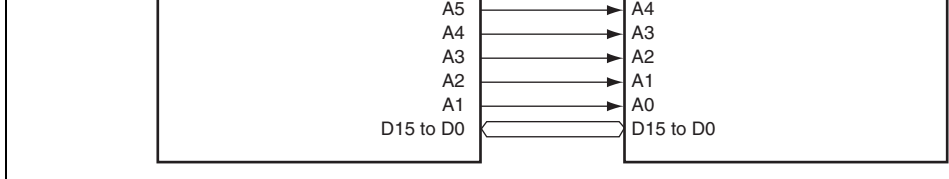


**Figure 9.45 Timing Example of Byte Control with Use of Two CAS Signals  
(Write Access with Lowest Bit of Address = B'0, RAST = 0, CAST = 0)**





**Figure 9.46 Timing Example of Word Control with Use of Two CAS Signals (Read Access with Lowest Bit of Address = B'0, RAST = 0, CAST = 0)**



**Figure 9.47 Example of Connection for Control with Two CAS Signals**

### 9.10.11 Burst Access Operation

Besides an accessing method in which this LSI outputs a row address every time it accesses DRAM (called full access or normal access), some DRAMs have a fast-page mode function which fast speed access can be achieved by modifying only a column address with the same row address output (burst access) when consecutive accesses are made to the same row address.

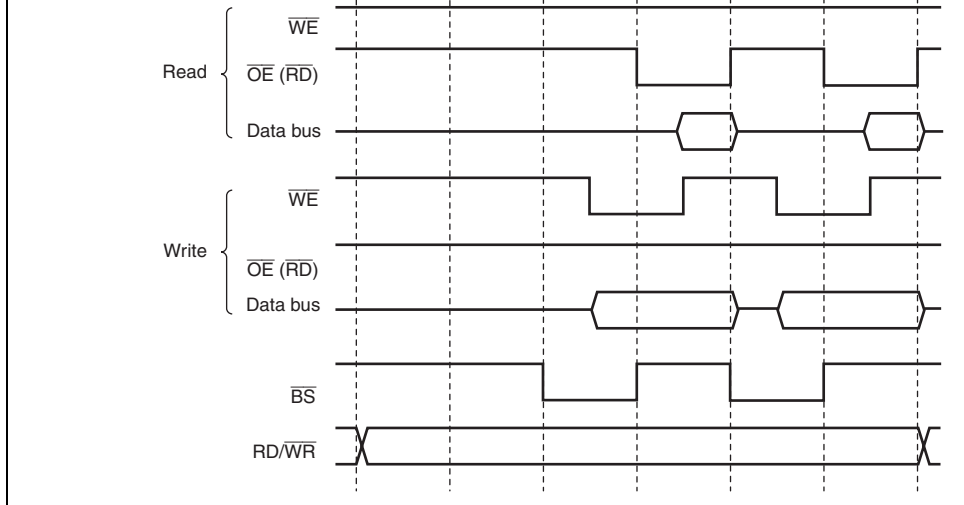
The fast-page mode (burst access) can be specified when the BE bit in DRAMCR is set to 1.

#### (1) Burst Access (Fast-Page Mode) Operation Timing

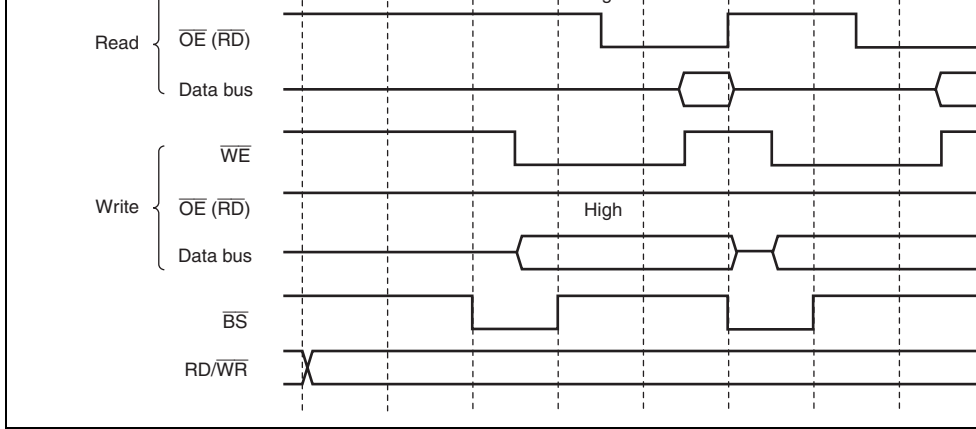
Figures 9.48 and 9.49 show operation timing of the fast-page mode.

When access cycles to the DRAM space are continued and the row addresses of the consecutive two cycles are the same, output cycles of the CAS and column address signals follow. The column address bits to be compared are decided by bits MXC1 and MXC0 in DRAMCR.

Wait cycles can be inserted during a burst access. The method and timing of the wait insertion are the same as that of full access mode. For details, see section 9.10.9, Wait Control.



**Figure 9.48 Operation Timing of Fast-Page Mode (RAST = 0, CAST = 0)**



**Figure 9.49 Operation Timing of Fast-Page Mode (RAST = 0, CAST = 1)**

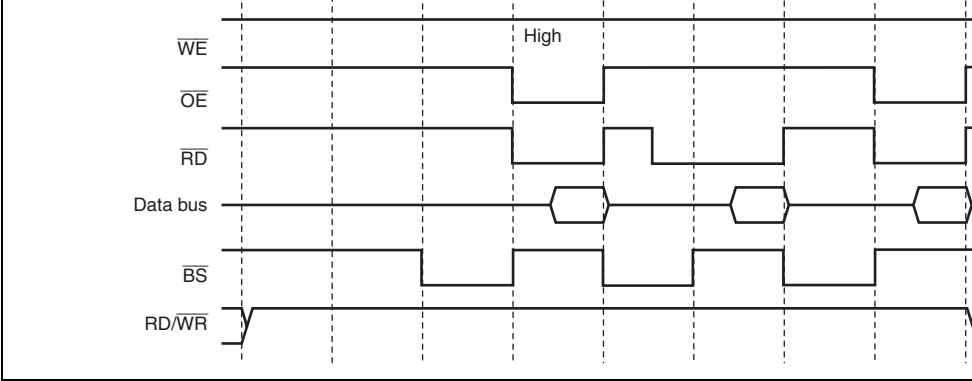
The fast-page mode access (burst access) is resumed when the row addresses of the current and previous cycle are the same. While other spaces are accessed when the DRAM space is halted, the  $\overline{\text{RAS}}$  signal must be low. Figure 9.50 shows a timing example of RAS down mode.

The  $\overline{\text{RAS}}$  signal goes high under the following conditions.

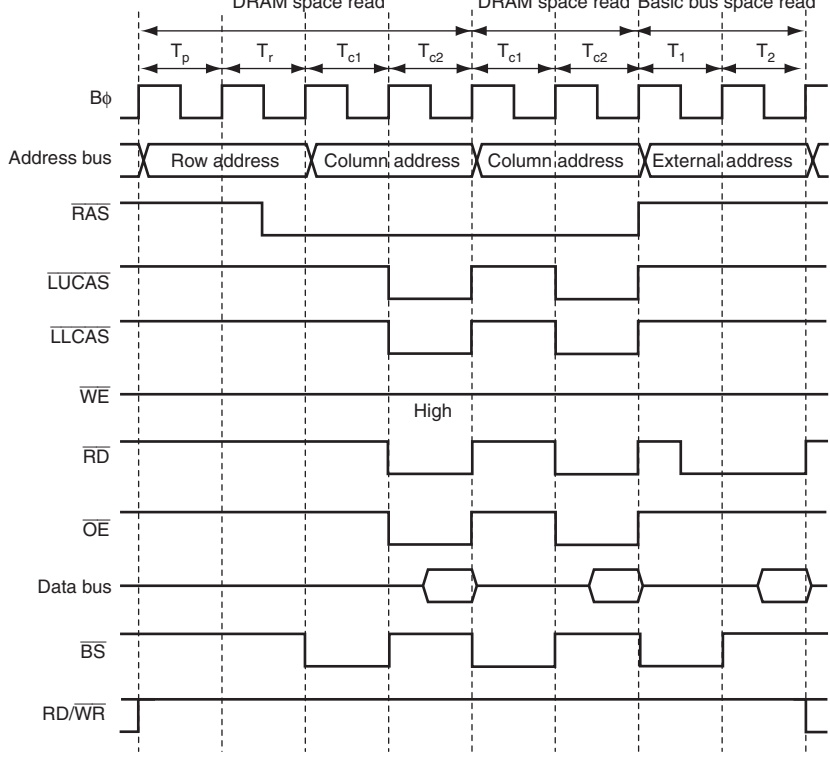
- When a refresh cycle is performed during RAS down mode
- When a self-refresh is performed
- When a transition to software standby mode is made
- When the external bus requested by the BREQ signal is released
- When either the RCDM or BE bit is cleared to 0

If a transition to the all-module clock-stop mode is made during RAS down mode, clock is stopped with the  $\overline{\text{RAS}}$  signal driven low. To make a transition with the  $\overline{\text{RAS}}$  signal driven high, clear the RCDM bit to 0 before execution of the SLEEP instruction.

Clear the RCDM bit to 0 for write access to SCKCR to set the clock frequencies. For SCKCR, see section 27, Clock Pulse Generator.



**Figure 9.50** Timing Example of RAS Down Mode (RAST = 0, CAST = 0)



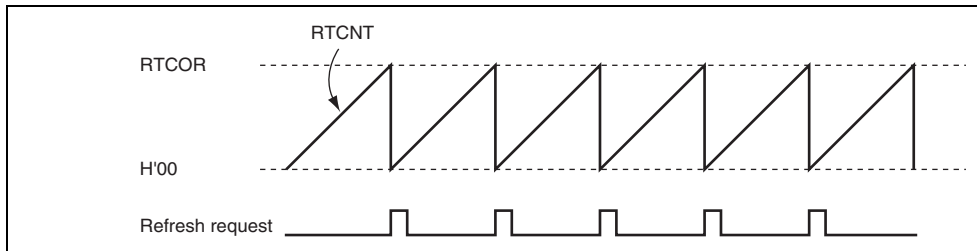
**Figure 9.51 Timing Example of RAS Up Mode (RAST = 0, CAST = 0)**

A CBR refresh cycle is performed when the value set in RTCOR matches the RTCNT value (compare match). RTCNT is an up-counter operated on the input clock specified by bits RTCR2 to RTCK0 in REFCR. RTCNT is initialized upon the compare match and restarts to count from H'00. Accordingly, a CBR refresh cycle is repeated at intervals specified by bits RTCR2 to RTCK0 in RTCOR. Set the bits so that the required refresh intervals of the DRAM must be satisfied.

Since setting bits RTCR2 to RTCK0 starts RTCNT to count up, set RTCNT and RTCOR before setting bits RTCR2 to RTCK0. When changing RTCNT and RTCOR, the counting operation should be halted. When changing bits RTCR2 to RTCK0, change them only after disabling external access and bus release by the EXDMAC, and if the write data buffer function is enabled, disabling the write data buffer function and reading the external space.

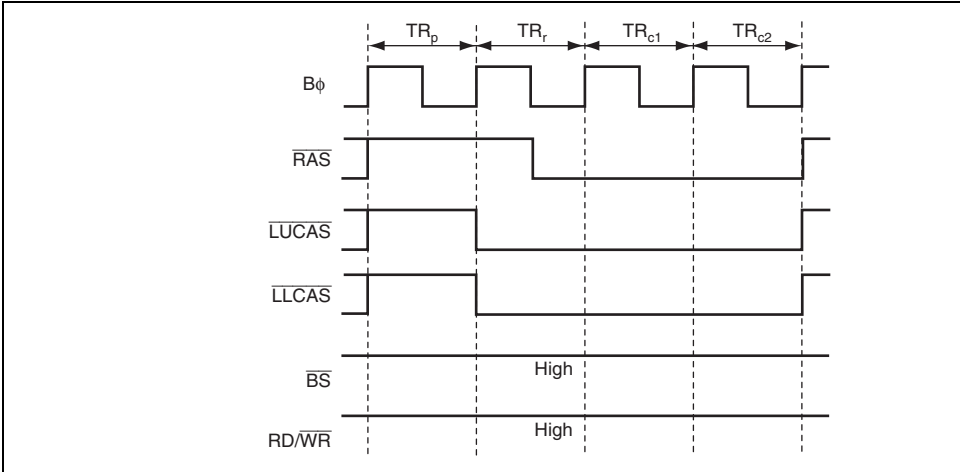
The external space cannot be accessed in CBR refresh mode.

Figure 9.52 shows RTCNT operation, figure 9.53 shows compare match timing, and figure 9.54 shows CBR refresh timing. Table 9.23 lists the pin states during a CBR refresh cycle.



**Figure 9.52 RTCNT Operation**



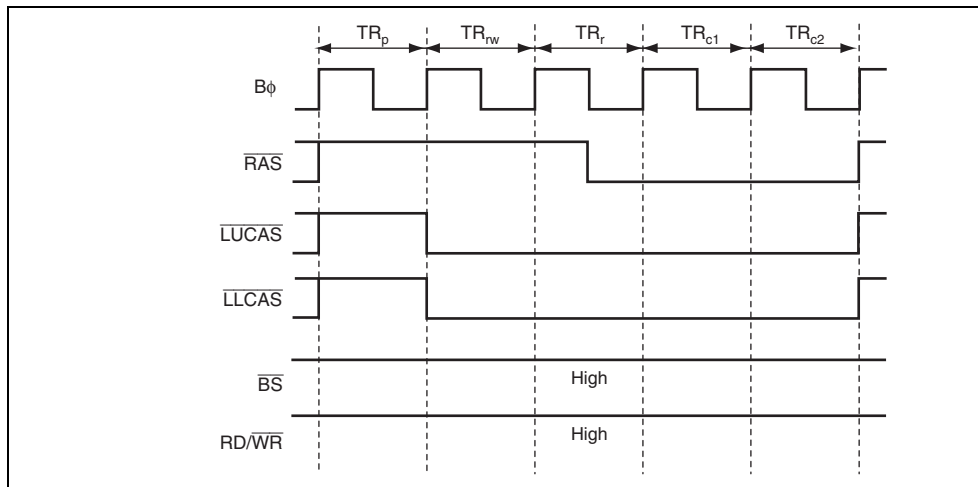


**Figure 9.54 CBR Refresh Timing**

$\overline{\text{BS}}$	High
$\text{RD}/\overline{\text{WR}}$	High

The  $\overline{\text{RAS}}$  signal can be delayed for one to three clock cycles by setting bits RCW1 and RCW0 in REFCR. The pulse width of the  $\overline{\text{RAS}}$  signal is changed by bits RLW2 to RLW0 in REFCR. Settings of bits RCW1, RCW0, and RLW2 to RLW0 are effective only for a refresh cycle precharge time set by bit TPC1 and TPC0 is effective for a refresh cycle.

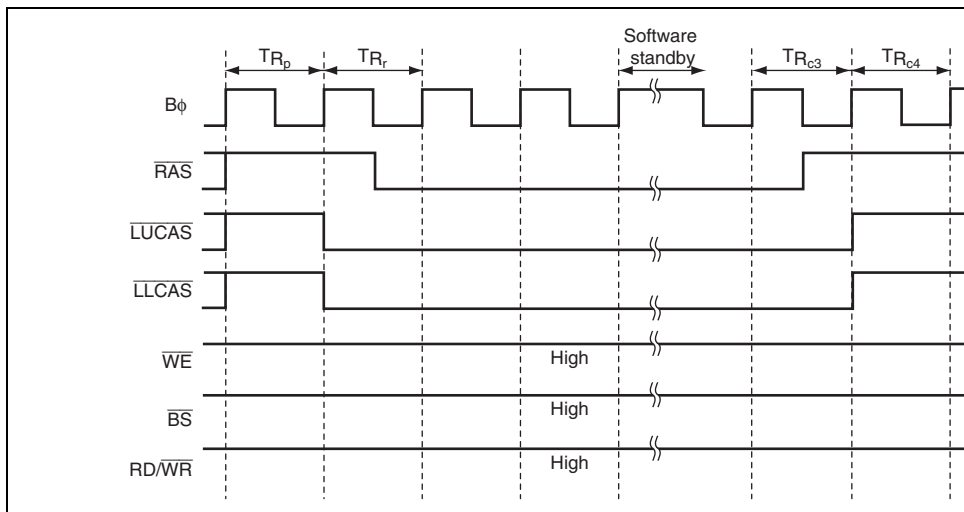
Figure 9.55 shows timing for setting bits RCW1 and RCW0



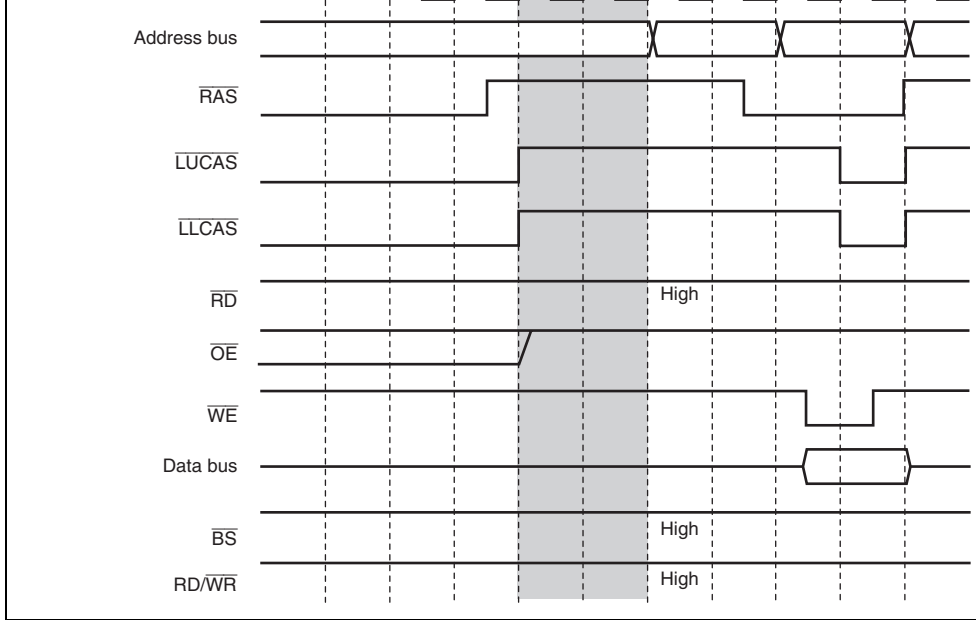
**Figure 9.55 CBR Refresh Timing**  
**(RCW1 = 0, RCW0 = 1, RLW2 = 0, RLW1 = 0, RLW0 = 0)**

When the self-refresh mode is used, do not clear the OPE bit in SBYCR to 0.

For details, see section 28.2.1, Standby Control Register (SBYCR).



**Figure 9.56 Self-Refresh Timing**



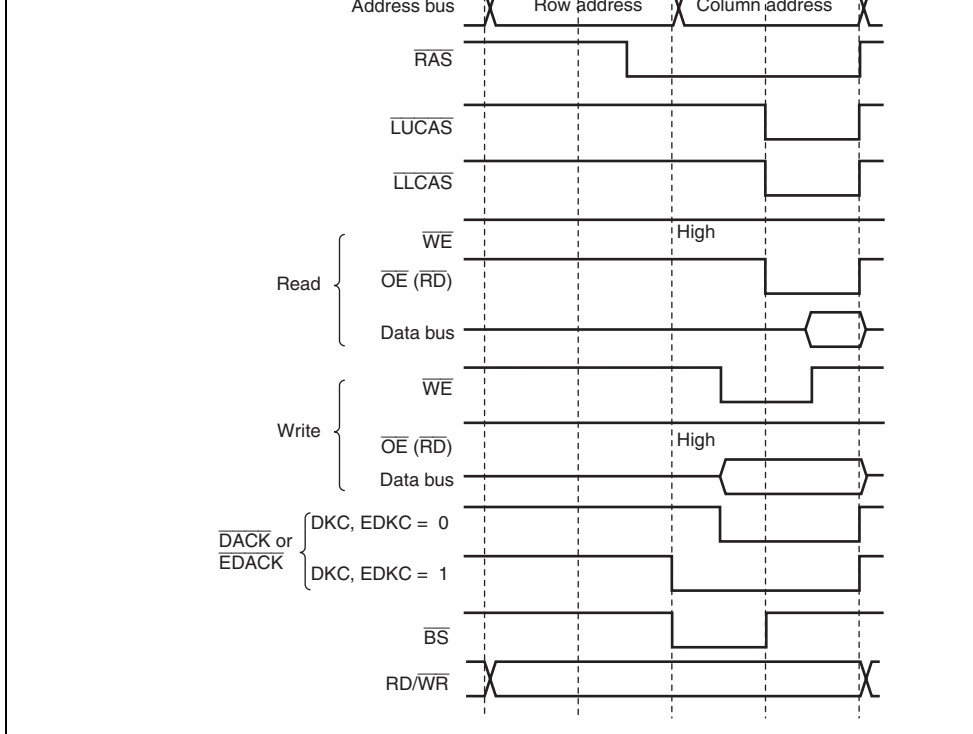
**Figure 9.57 Timing Example when 1 Precharge Cycle Added**

For details, see section 28.2.2, Module Stop Control Registers A and B (MSTPCRA and MSTPCRB).

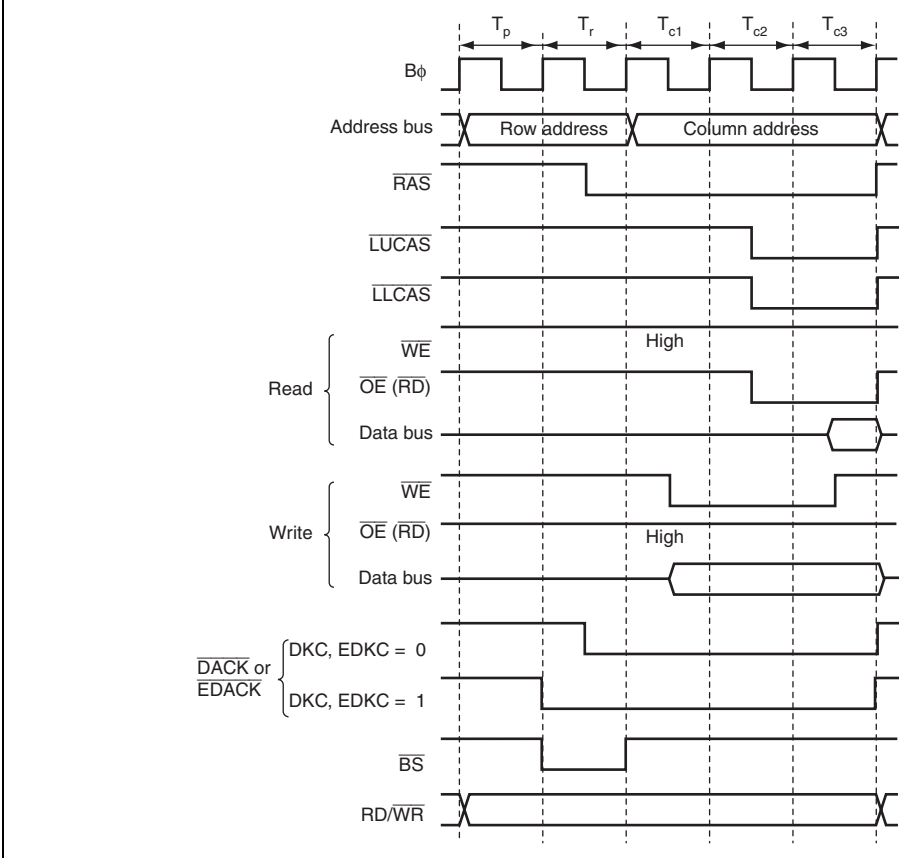
### 9.10.13 DRAM Interface and Single Address Transfer by DMAC and EXDMAC

When fast-page mode ( $BE = 1$ ) is set for the DRAM space, either fast-page access or full access can be selected, by the setting of bits DDS and EDDS in DRAMCR, for the single address transfer by the DMAC or EXDMAC where the DRAM space is specified as the transfer source or destination. At the same time, the output timings of the  $\overline{DACK}$ ,  $\overline{EDACK}$  and BS signals are changed. When  $BE = 0$ , full access to the DRAM space is performed by single address transfer regardless of the setting of bits DDS and EDDS. However, the output timing of the  $\overline{DACK}$ ,  $\overline{EDACK}$  and BS signals can be changed by the setting of bits DDS and EDDS.

The assertion timing of the  $\overline{DACK}$  and  $\overline{EDACK}$  signal can be changed by bits DKC and BCR1.



**Figure 9.58 Output Timing Example of  $\overline{DACK}$  and  $\overline{EDACK}$  when DDS = 1 or ED (RAST = 0, CAST = 0)**



**Figure 9.59 Output Timing Example of  $\overline{DACK}$  and  $\overline{EDACK}$  when  $DDS = 0$  or  $1$  ( $RAST = 0, CAST = 1$ )**

In the SDRAM space, pins PB2, PB3, and PB4 are used as the  $\overline{\text{RAS}}$ ,  $\overline{\text{CAS}}$ , and  $\overline{\text{WE}}$  signals. The PB1 pin is used as the  $\overline{\text{CS2}}$  signal by the PFCR setting, and the PB5 pin is used as the CK signal by setting the OEE bit in DRAMCR to 1. The bus settings of the SDRAM space depend on the PFCR settings. The pin wait and program wait for the SDRAM space are not available. For PFCR settings, see section 13, I/O Ports.

An SDRAM command is designated by the combination of the  $\overline{\text{RAS}}$ ,  $\overline{\text{CAS}}$ , and  $\overline{\text{WE}}$  signals. The precharge-sel command (Precharge-sel) output on the upper column address.

This LSI supports the following commands: the NOP, auto-refresh (REF), self-refresh (SRF), bank-precharge (PALL), bank active (ACTV), read (READ), write (WRIT), and mode register setting (MRS). Commands controlling a bank are not supported.

**Table 9.24 Relationship among DRAME and DTYPE and Area 2 Interfaces**

DRAME	DTYPE	Area 2 Interface
0	X	Basic bus space (initial state)/byte-control SRAM space
1	0	DRAM space
1	1	SDRAM space

[Legend]

X: Don't care



				Column address	A23 to A18	-	-	A23	A22	A21	A20	A19	P	A9	A8	A7	A6	A5	A4	A3
			16 bits	Row address	A23 to A18	-	-	A23	A22	A21	A20	P/A19*	A18	A17	A16	A15	A14	A13	A12	A11
				Column address	A23 to A18	-	-	A23	A22	A21	A20	P	A10	A9	A8	A7	A6	A5	A4	A3
0	1	9 bits	8 bits	Row address	A23 to A18	A17	-	-	A23	A22	A21	A20	P/A19	A18	A17	A16	A15	A14	A13	A12
				Column address	A23 to A18	A17	-	-	A23	A22	A21	A20	P	A9	A8	A7	A6	A5	A4	A3
			16 bits	Row address	A23 to A18	A17	-	-	A23	A22	A21	P/A20*	A19	A18	A17	A16	A15	A14	A13	A12
				Column address	A23 to A18	A17	-	-	A23	A22	A21	P	A10	A9	A8	A7	A6	A5	A4	A3
1	0	10 bits	8 bits	Row address	A23 to A18	-	-	-	-	A23	A22	A21	P/A20*	A19	A18	A17	A16	A15	A14	A13
				Column address	A23 to A18	-	-	-	-	A23	A22	A21	P	A9	A8	A7	A6	A5	A4	A3
			16 bits	Row address	A23 to A18	-	-	-	-	A23	A22	P/A21*	A20	A19	A18	A17	A16	A15	A14	A13
				Column address	A23 to A18	-	-	-	-	A23	A22	P	A10	A9	A8	A7	A6	A5	A4	A3
1	1	11 bits	8 bits	Row address	A23 to A18	A17	-	-	-	-	A23	A22	P/A21*	A20	A19	A18	A17	A16	A15	A14
				Column address	A23 to A18	A17	-	-	-	-	A23	A10	P	A9	A8	A7	A6	A5	A4	A3
			16 bits	Row address	A23 to A18	A17	-	-	-	-	A23	P/A22*	A21	A20	A19	A18	A17	A16	A15	A14
				Column address	A23 to A18	A17	-	-	-	-	A11	P	A10	A9	A8	A7	A6	A5	A4	A3

Note: \* When issuing the PALL command, precharge-sel = 1 is output and when issuing the ACTIV command, a corresponding address is output.

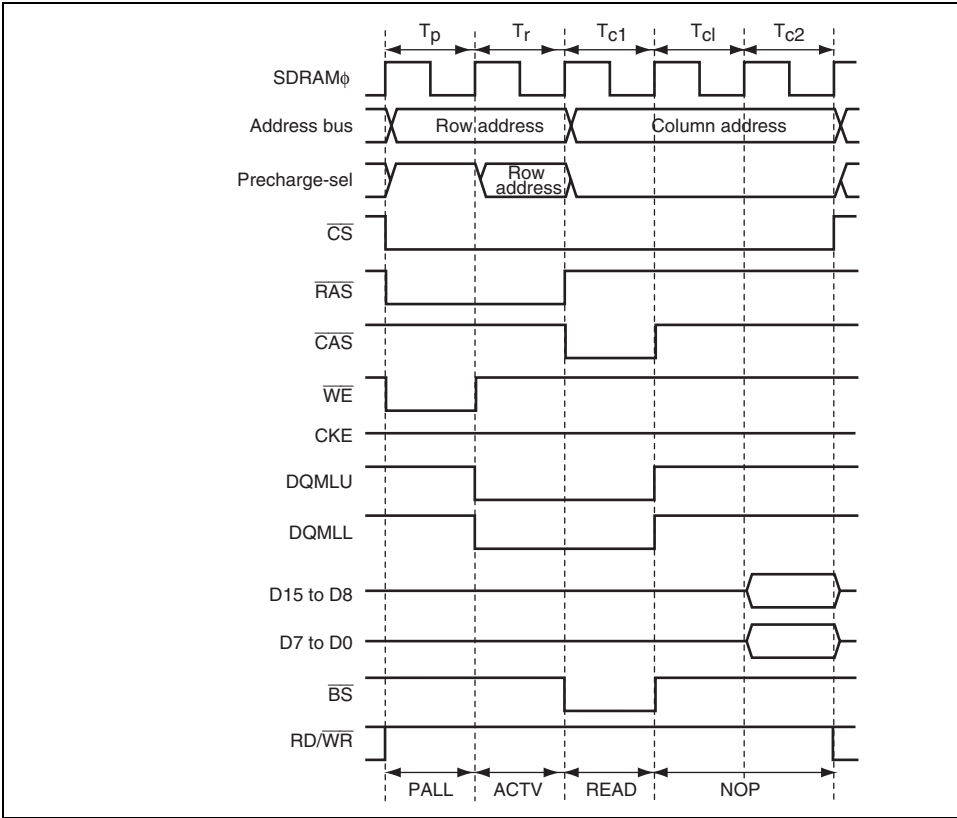
### 9.11.3 Data Bus

Either 8 or 16 bits can be selected as the data bus width of the SDRAM space by bits ABWL2 in ABWCR. SDRAM with 16-bit words can be connected directly to 16-bit bus space.

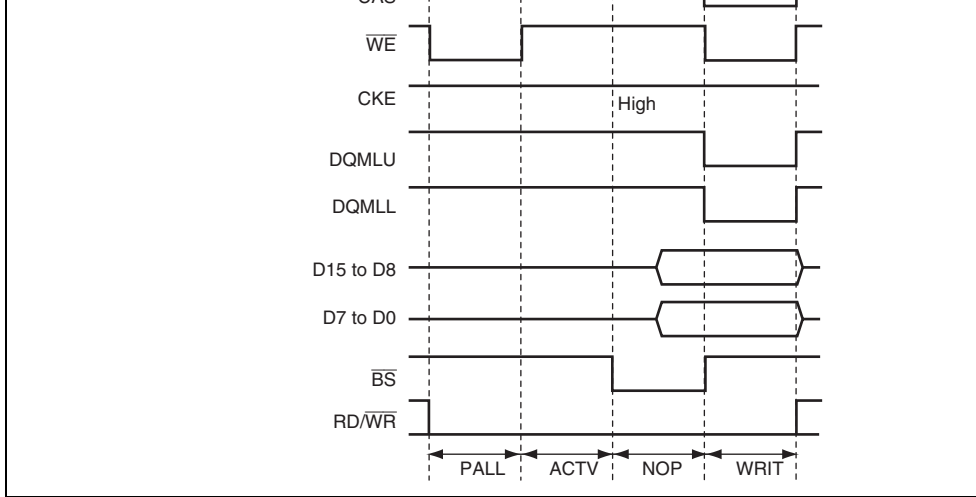
D7 to D0 are valid in 8-bit SDRAM space and D15 to D0 are valid in 16-bit SDRAM space.

The data endian format can be selected by bit LE2 in ENDIANCR. For details on the address and alignment, see section 9.5.6, Endian and Data Alignment.

Pin	DRAM Selected	Name	I/O	Function
$\overline{\text{RAS}}$	$\overline{\text{RAS}}$	Row address strobe	Output	Row address strobe when the SDRAM space is specified as area 2
$\overline{\text{CAS}}$	$\overline{\text{CAS}}$	Column address strobe	Output	Column address strobe when the SDRAM space is specified as area 2
$\overline{\text{WE}}$	$\overline{\text{WE}}$	Write enable	Output	Write enable signal for accessing SDRAM interface
$\overline{\text{OE/CKE}}$	CKE	Clock enable	Output	Clock enable signal when the SDRAM space is specified as area 2.
$\overline{\text{LLCAS}}/\text{DQMLU}$	DQMLU	Lower-upper data mask enable	Output	Upper data mask enable when the 16-bit SDRAM space is accessed
$\overline{\text{LLCAS}}/\text{DQMLL}$	DQMLL	Lower-lower data mask enable	Output	<ul style="list-style-type: none"> <li>• Lower data mask enable when the 16-bit SDRAM space is accessed</li> <li>• Data mask enable when the SDRAM is accessed</li> </ul>
A17 to A0	A17 to A0	Address pin	Output	Multiplexed row/column-address pin
D15 to D0	D15 to D0	Data pin	Input/output	Data input/output pin
PB7	SDRAM $\phi$	Clock	Output	SDRAM clock
$\overline{\text{CS2}}$	$\overline{\text{CS}}$	Chip select	Output	Strobe signal indicating that SDRAM is selected

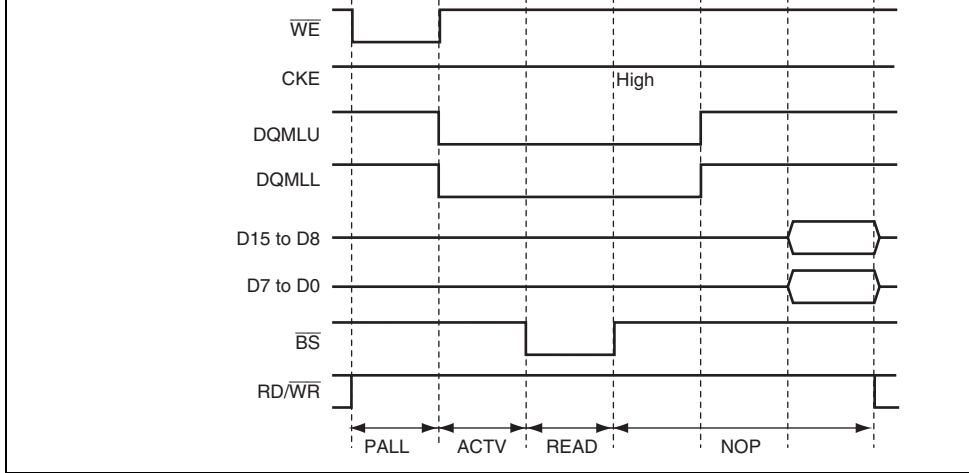


**Figure 9.60 SDRAM Basic Read Access Timing (CAS Latency = 2)**

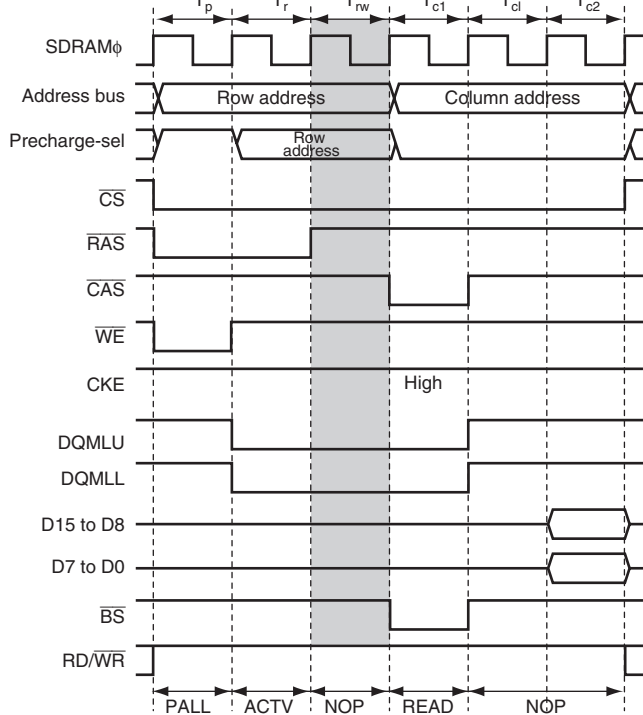


**Figure 9.61 SDRAM Basic Write Access Timing**

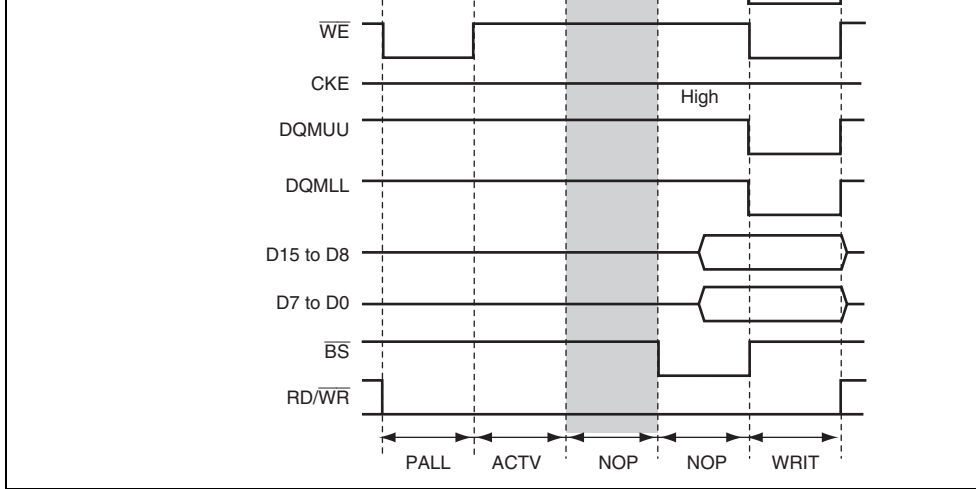
<b>W21</b>	<b>W20</b>	<b>Description</b>	<b>Number of Latency (ns)</b>
0	0	Setting prohibited	—
	1	SDRAM with CAS latency of 2 is in use	1
1	0	SDRAM with CAS latency of 3 is in use	2
	1	SDRAM with CAS latency of 4 is in use	3



**Figure 9.62 Timing Example of CAS Latency (CAS Latency = 3)**

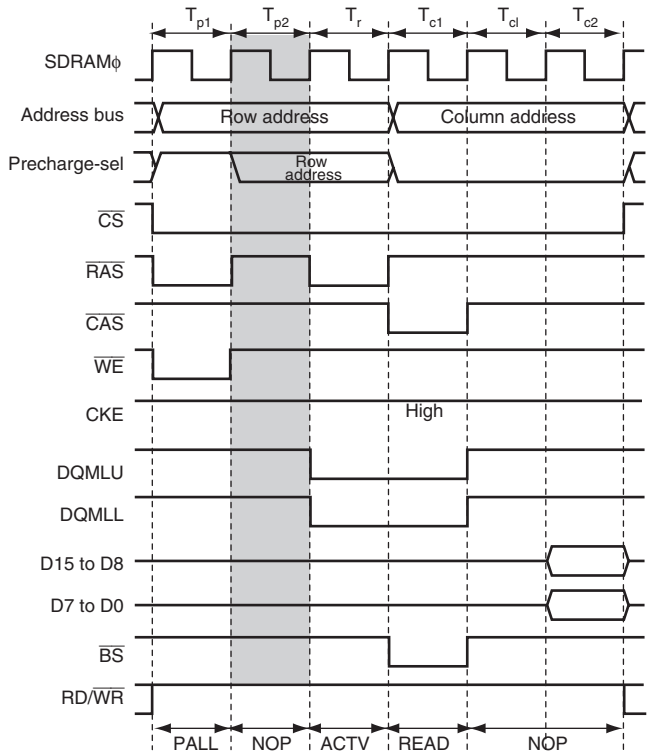


**Figure 9.63 Read Timing Example of Row Address Output Retained for 1 Clock (RCD1 = 0, RCD0 = 1, CAS Latency = 2)**

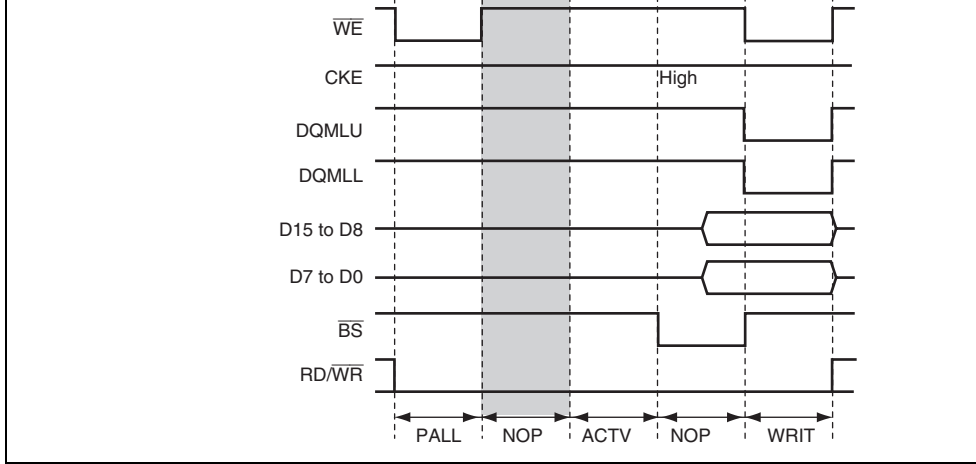


**Figure 9.64 Write Timing Example of Row Address Output Retained for 1 Clock (RCD1 = 0, RCD0 = 1)**

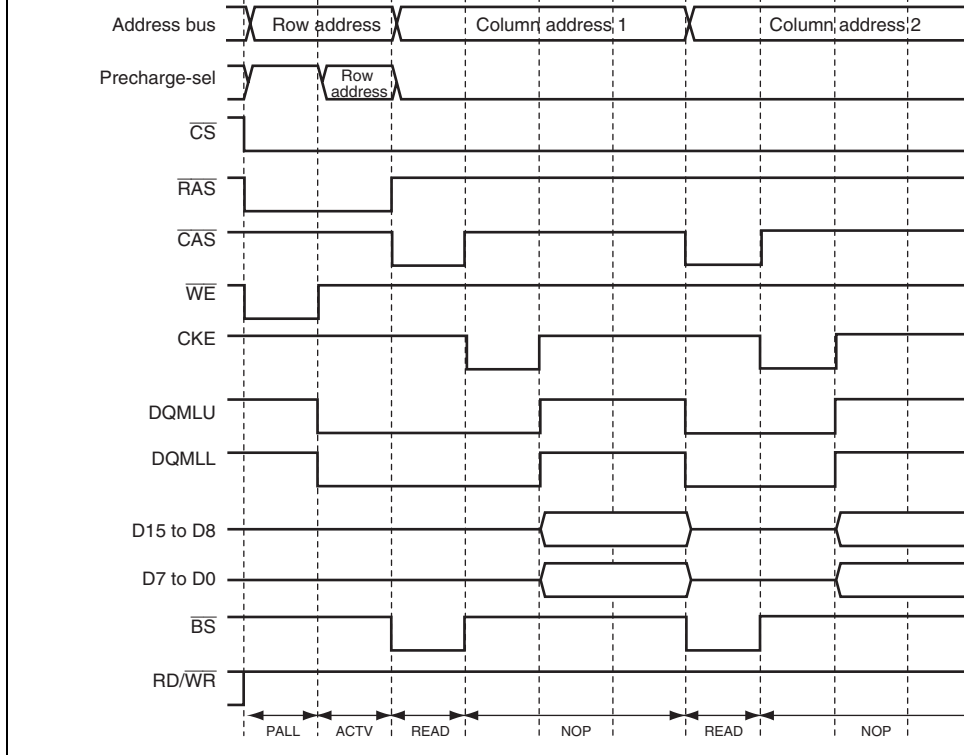




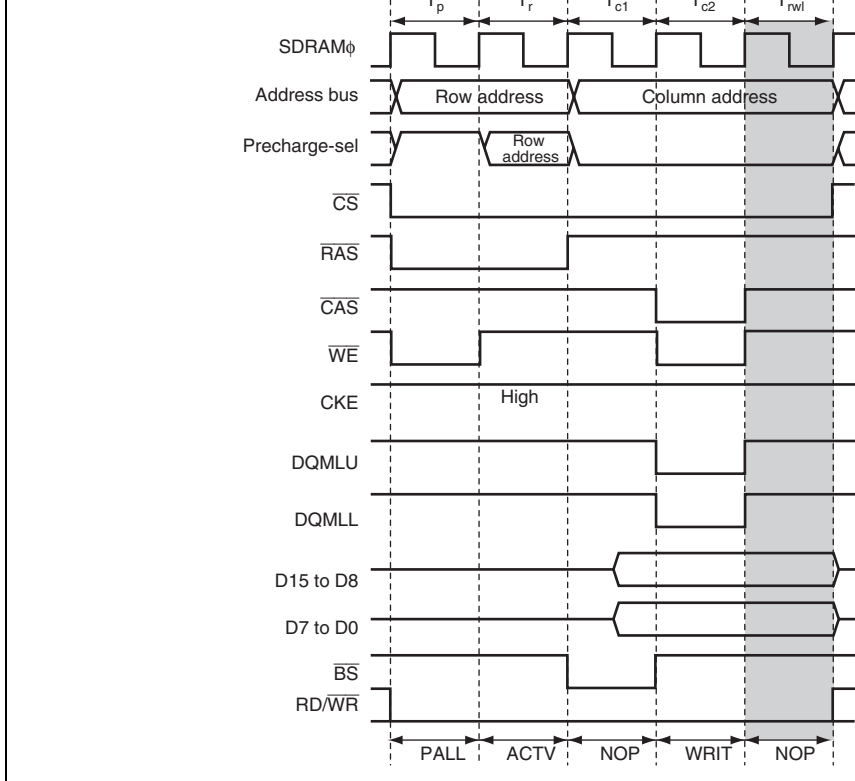
**Figure 9.65 Read Timing Example of Two Precharge Cycles (TPC1 = 0, TPC0 = 1, CAS Latency = 2)**



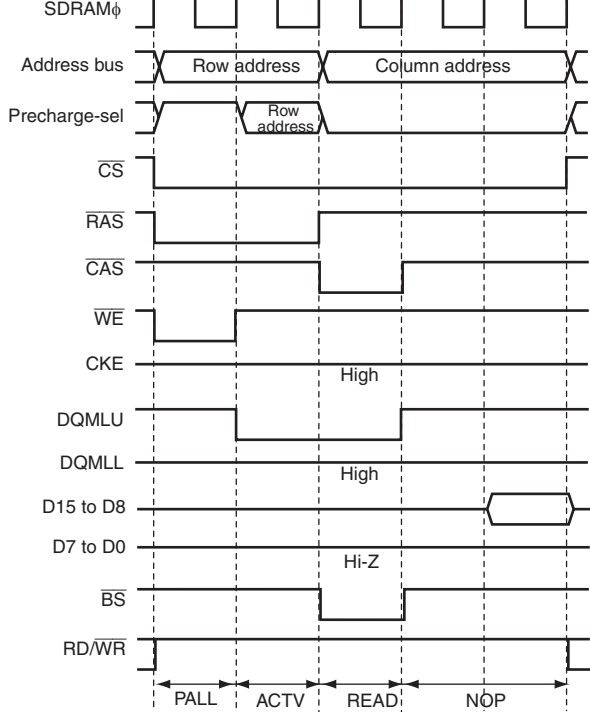
**Figure 9.66 Write Timing Example of Two Precharge Cycles (TPC1 = 0, TPC0 = 1)**



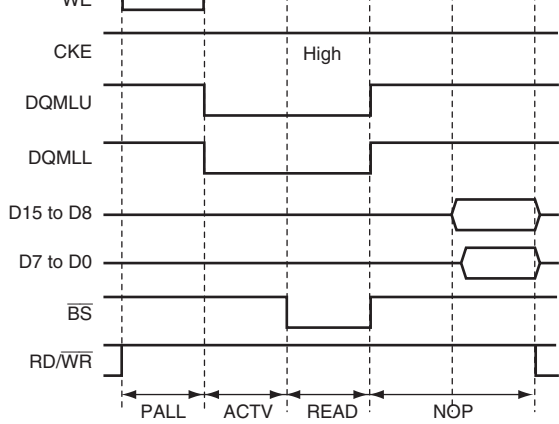
**Figure 9.67 Read Timing Example when CKSPE = 1 (CAS Latency = 2)**



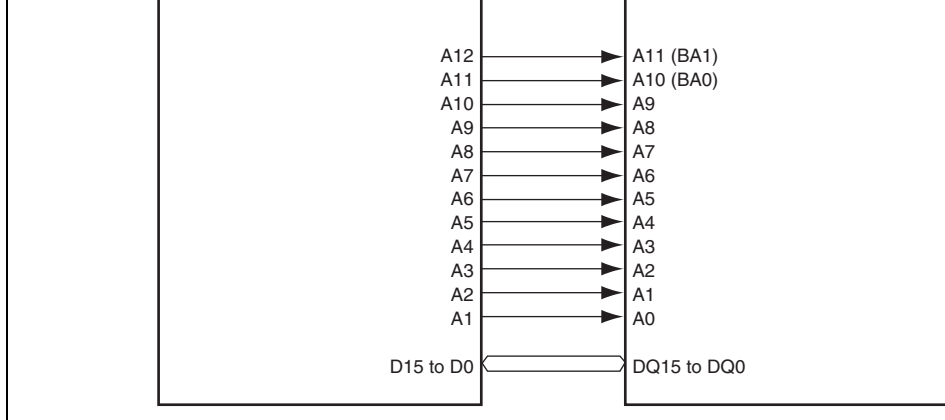
**Figure 9.68 Write Timing Example when Write-Precharge Delay Cycle Inserted (TRWL = 1)**



**Figure 9.69 Control Timing Example of Byte Control by DQM in 16-Bit Access (Read Access with Lowest Bit of Address = B'0)**



**Figure 9.70 Control Timing Example of Word Control by DQM in 16-Bit Access (Read Access with Lowest Bit of Address = B'0, CAS Latency = 2)**

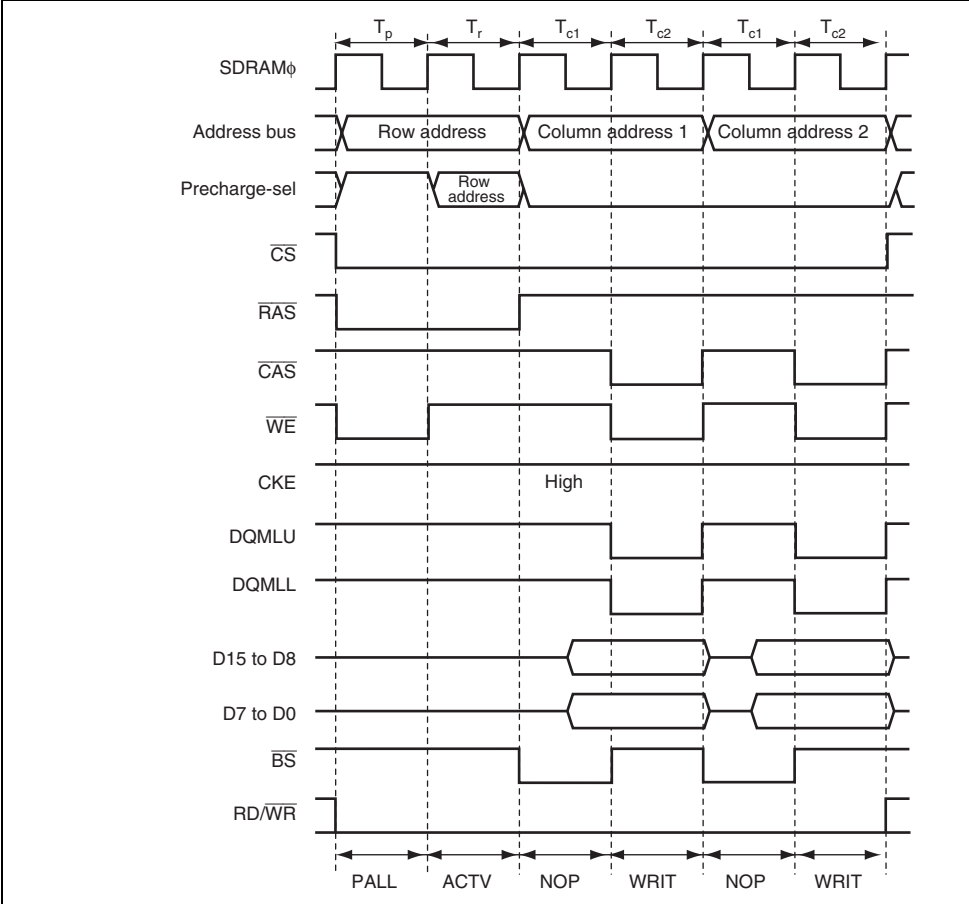


**Figure 9.71 Connection Example of DQM Byte/Word Control**

### 9.11.12 Fast-Page Access Operation

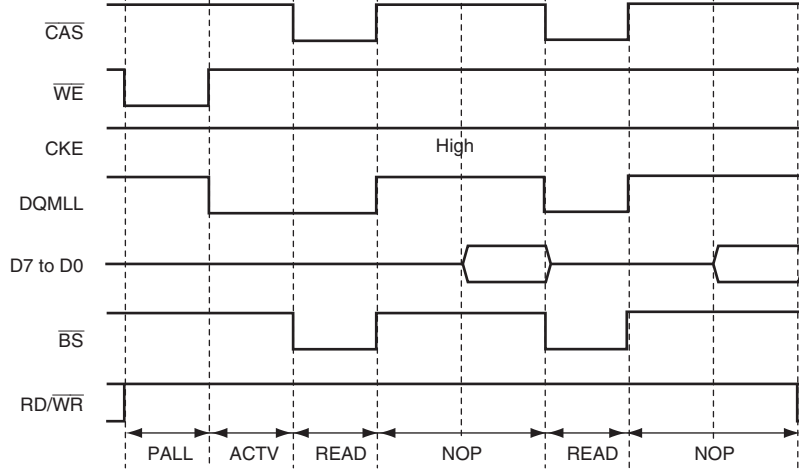
Besides an accessing method in which this LSI outputs a row address every time it accesses SDRAM (called full access or normal access), some SDRAMs have a fast-page mode function which fast speed access can be achieved by modifying only a column address with the same row address output when consecutive accesses are made to the same row address.

The fast-page mode can be used by setting the BE bit in DRAMCR to 1.



**Figure 9.72 Longword Write Timing in 16-Bit Access Space (BE = 1, RCDM)**





**Figure 9.73 Word Read Timing in 8-Bit Access Space  
(BE = 1, RCDM = 0, CAS Latency = 2)**

The next cycle after one of the following conditions is satisfied is a full access cycle.

- When a refresh cycle is performed during RAS down mode
- When a self-refresh is performed
- When a transition to software standby mode is made
- When the external bus requested by the  $\overline{\text{BREQ}}$  signal is released
- When either the RCDM or BE bit is cleared to 0
- When setting the SDRAM mode register

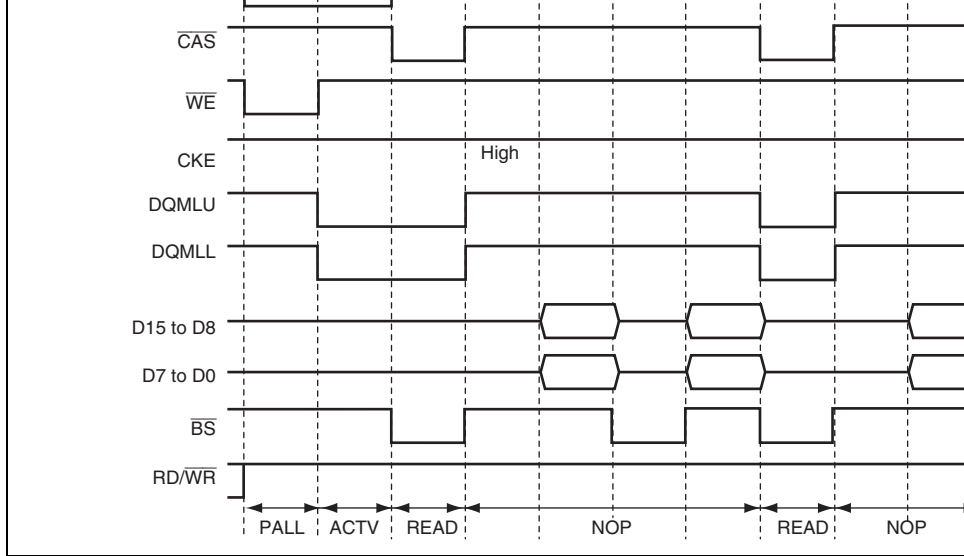
Some SDRAMs have a limitation on the time to hold each bank active. When such SDRAMs are used, if the user program cannot control the time (such as software standby or sleep mode), the auto-refresh or self-refresh so that the given specification can be satisfied. If a refresh is not used, the user program must control the time.

Clear the RCDM bit to 0 for write access to SCKCR to set the clock frequencies. For SDRAM, see section 27, Clock Pulse Generator.

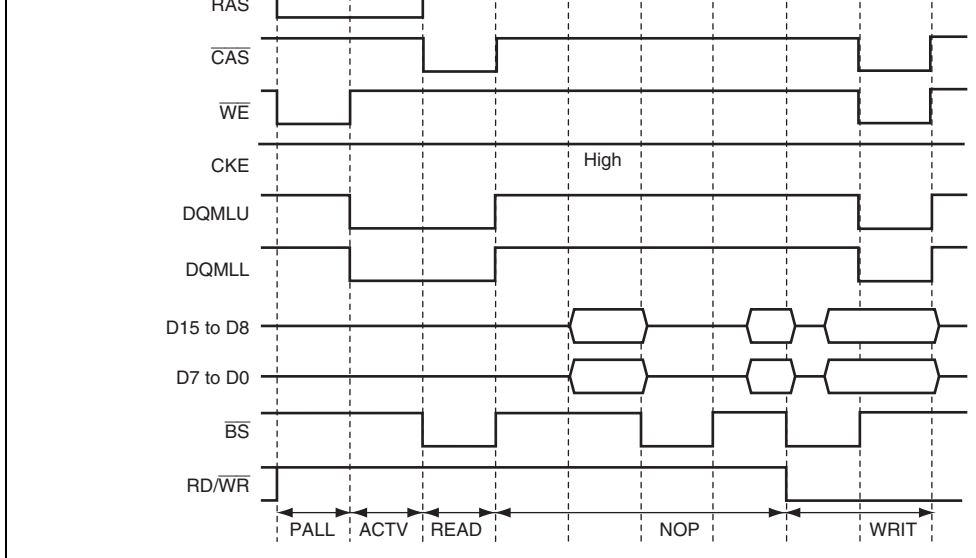
### (3) RAS Up Mode

Clear the RCDM bit in DRAMCR to 0 to set the RAS up mode.

Whenever a SDRAM space access is halted and other spaces are accessed, the next cycle is a fast-PALL command cycle. Only when the SDRAM space continues to be accessed, the fast-PALL mode access is performed.



**Figure 9.74** Timing Example of RAS Down Mode (BE = 1, RCDM = 1, CAS Latency = 1)



**Figure 9.75** Timing Example of RAS Down Mode (BE = 1, RCDM = 1, CAS Later)

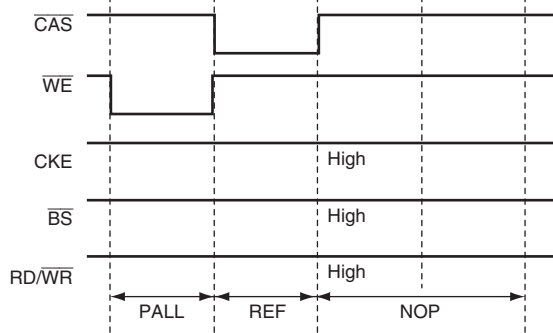
An auto-refresh cycle is performed when the value set in RTCOR matches the RTCNT value (compare match). RTCNT is an up-counter operated on the input clock specified by bits RTCK2 to RTCK0 in REFCR. RTCNT is initialized upon the compare match and restarts to count from H'00. Accordingly, an auto-refresh cycle is repeated at intervals specified by bits RTCK2 to RTCK0 in RTCOR. Set the bits so that the required refresh intervals of the DRAM must be satisfied.

Since setting bits RTCK2 to RTCK0 starts RTCNT to count up, set RTCNT and RTCOR before setting bits RTCK2 to RTCK0. When changing RTCNT and RTCOR, the count operation must be halted. When changing bits RTCK2 to RTCK0, change them only after disabling the DRAM access and external bus release by the EXDMAC, if the write data buffer function is in use. After disabling the write data buffer function and reading the external space.

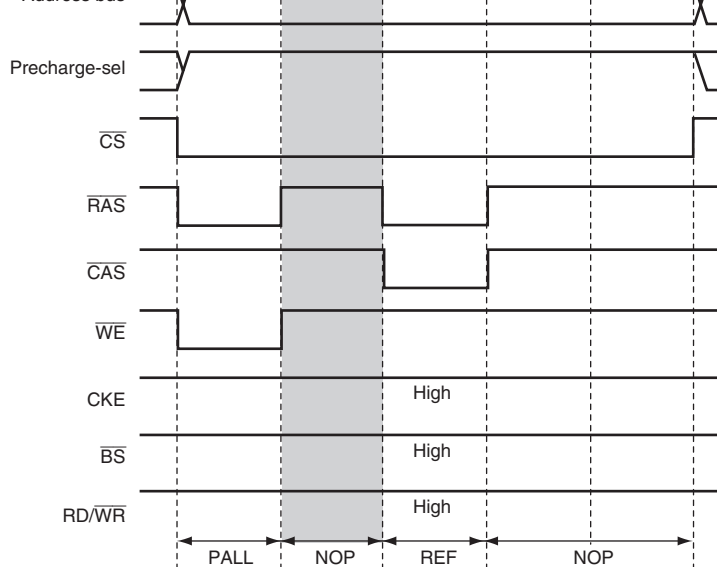
The external space cannot be accessed during auto-refresh.

Figure 9.76 shows auto-refresh cycle timing.

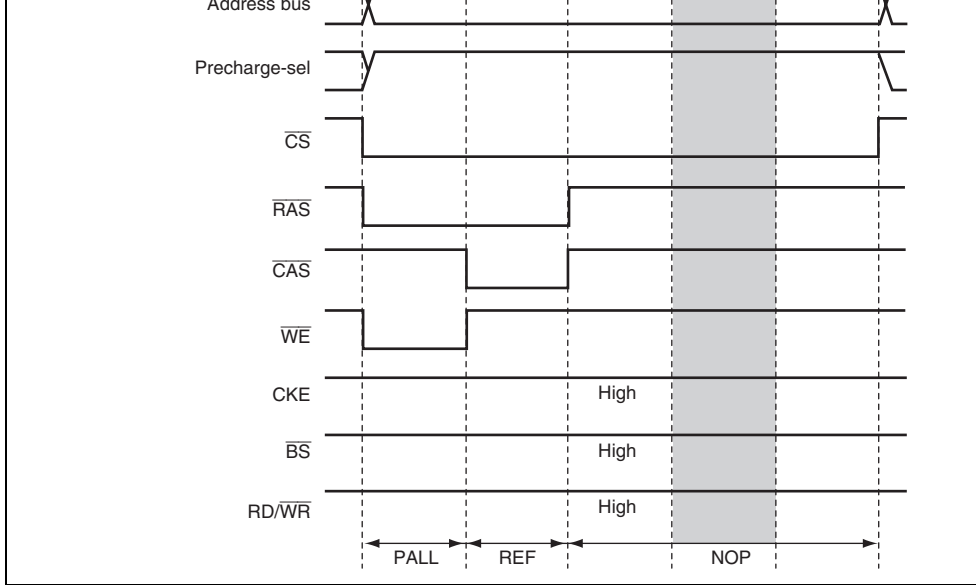
The operation of refresh counter is same as that for the DRAM interface. For details, see Section 9.10.12, Refresh Control.



**Figure 9.76 Auto-Refresh Operation**



**Figure 9.77 Auto-Refresh Timing (TPC1 = 0, TPC0 = 1)**



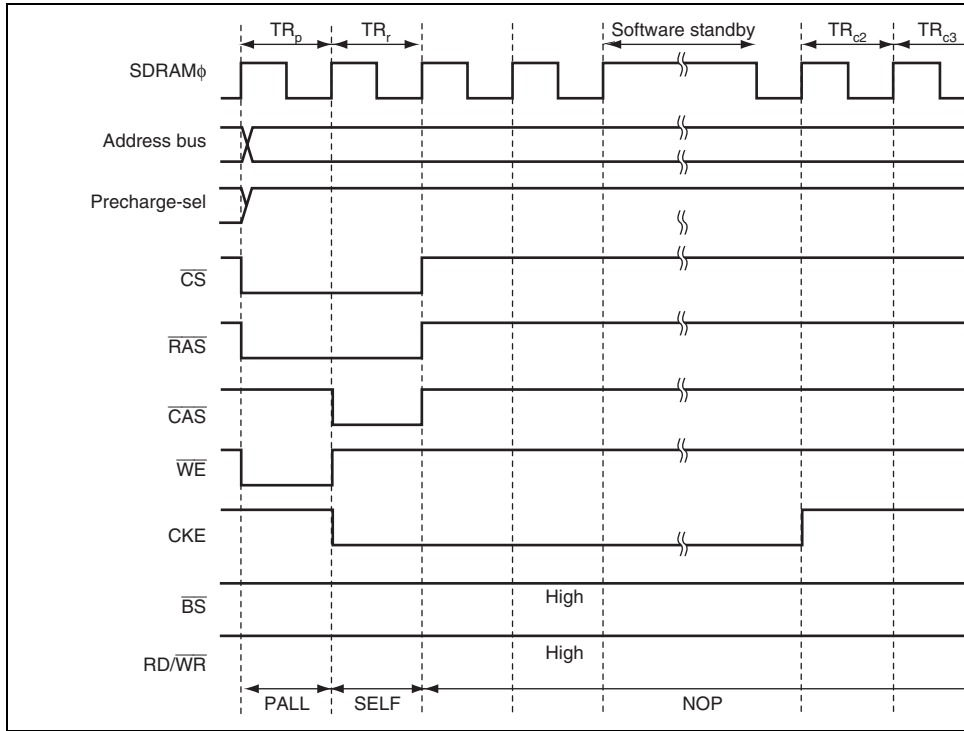
**Figure 9.78 Auto-Refresh Timing (TPC1 = 0, TPC0 = 0, RLW2 = 0, RLW1 = 0, RLW0 = 0)**



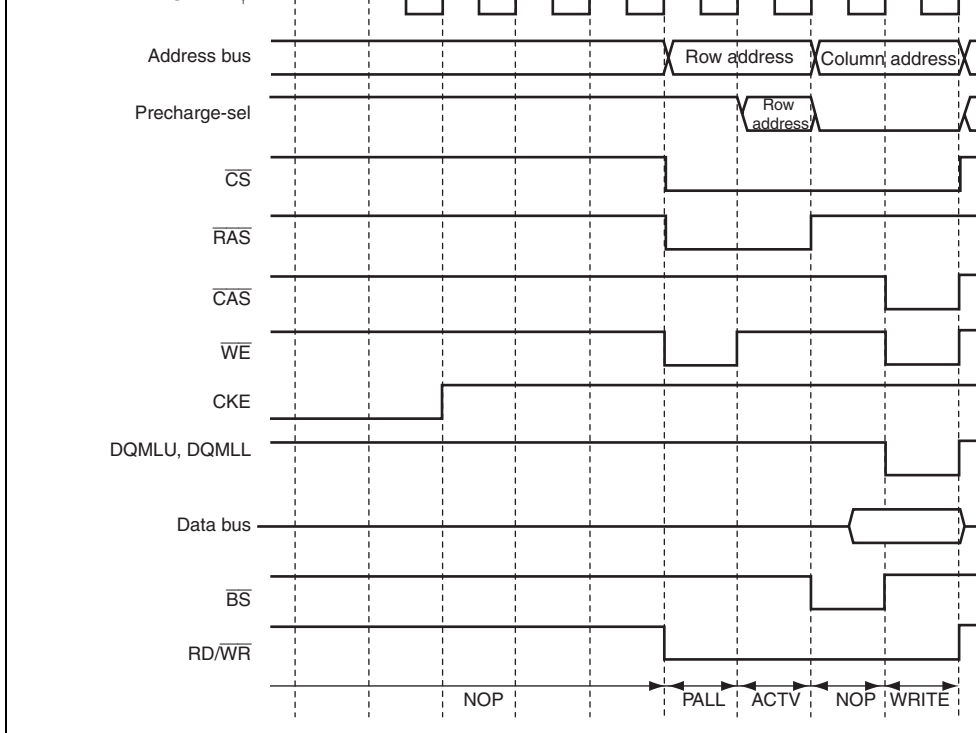
first performed and then the self refresh is entered.

When making a transition to the self-refresh, set the OEE bit in SBYCR to 1 and connect pin.

When the self-refresh is used, do not clear the OPE bit in SBYCR to 0.



**Figure 9.79 Self-Refresh Timing in software standby mode**  
**(TPC1 = 0, TPC0 = 0, RCW1 = 0, RCW0 = 0, RLW2 = 0, RLW1 = 0, RLW0 = 0)**



**Figure 9.80 Timing Example when 1 Precharge Cycle Added in the Software Standby Mode (TPCS2 to TPCS0 = H'1, TPC1 = 0, TPC0 = 0)**

1. In PBDDR/PBDR, set PB1 ( $\overline{CS2}$ ) as a high-level output and PB5 (CKE) as a low-level output.

Since the setting of the IOKEEP bit ensures retention of pin state at this time, the state of high-level output on  $\overline{CS2}$  and low-level output on is retained.

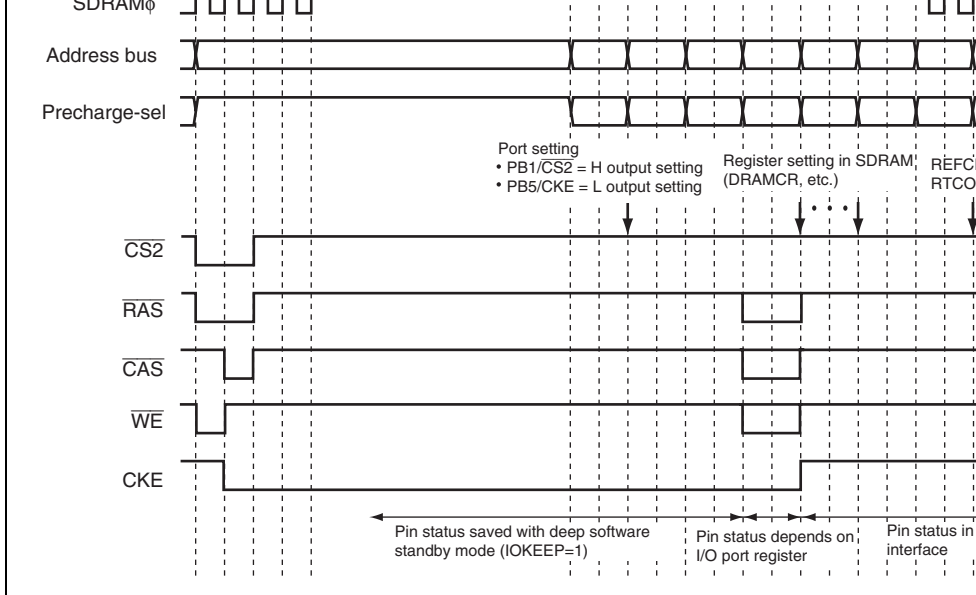
2. Set the PSTOP0 bit in SCKCR to 1 and SDRAM $\phi$  as a high level output. Since the setting of the IOKEEP bit continues to ensure retention of pin state, the existing state of low-level output on SDRAM $\phi$  is retained.
3. Clear the IOKEEP bit in DPSBYCR.

This releases pin states from retention due to the setting of the IOKEEP bit, but the state of pins  $\overline{CS2}$ , CKE, and SDRAM $\phi$  as set in steps 1 and 2 do not change.

4. In the synchronous DRAM-related control registers that were initialized by the initialization sequence, the reset that accompanied the transition to deep software standby mode, remake the settings to enable the synchronous DRAM interface. At this time, do not make settings in RTCNT, and RTCOR.

Once the synchronous DRAM interface has been enabled, the state of the CKE pin changes from low-level output to high-level output.

5. Restart output of the SDRAM $\phi$  clock signal by clearing the PSTOP0 bit in SCKCR. SDRAM $\phi$  restarts supply of SDRAM $\phi$  to the synchronous DRAM.
6. Set REFCR, RTCNT, and RTCOR and enable refreshing.  
As the state of the CKE pin has been changed in the step 4, adjust the time between the state of change of the CKE pin and the next cycle of auto-refreshing in this procedure within the stipulated refreshing interval of the synchronous DRAM.
7. Resume access to the synchronous DRAM.  
Pre-charging time after the termination of self-refresh will be secured by the timing setting in step 6.



**Figure 9.81 Self-Refresh Timing in Deep Software Standby Mode**  
 (TPC1 = 0, TPC0 = 0, RCW1 = 0, RCW0 = 0, RLW2 = 0, RLW1 = 0, RLW0 = 0)

For details, see section 28.2.2, Module Stop Control Registers A and B (MSTPCRA and MSTPCRB).

### 9.11.14 Setting SDRAM Mode Register

To use SDRAM, the mode register must be specified after a power-on reset.

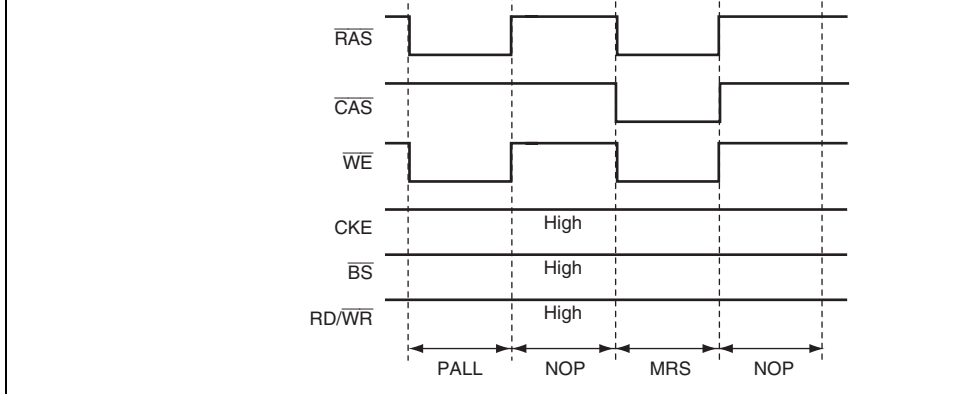
Setting the MRSE bit in SDCR to 1 enables the SDRAM mode register setting. After this, the SDRAM space in bytes.

When the value to be set in the SDRAM mode register is  $x$ , write to the following memory location (address). The value of  $x$  is written to the SDRAM mode register.

- H'4000000/H'400000 +  $x$  for 8-bit bus SDRAM
- H'4000000/H'400000 +  $2x$  for 16-bit bus SDRAM

The SDRAM mode register latches the address signals when the MRS command is issued.

This LSI does not support the burst read/burst write mode of SDRAM. When setting the mode register, use the burst read/single write mode and set the burst length to 1. Setting the SDRAM mode register must be consistent with that in the bus controller.



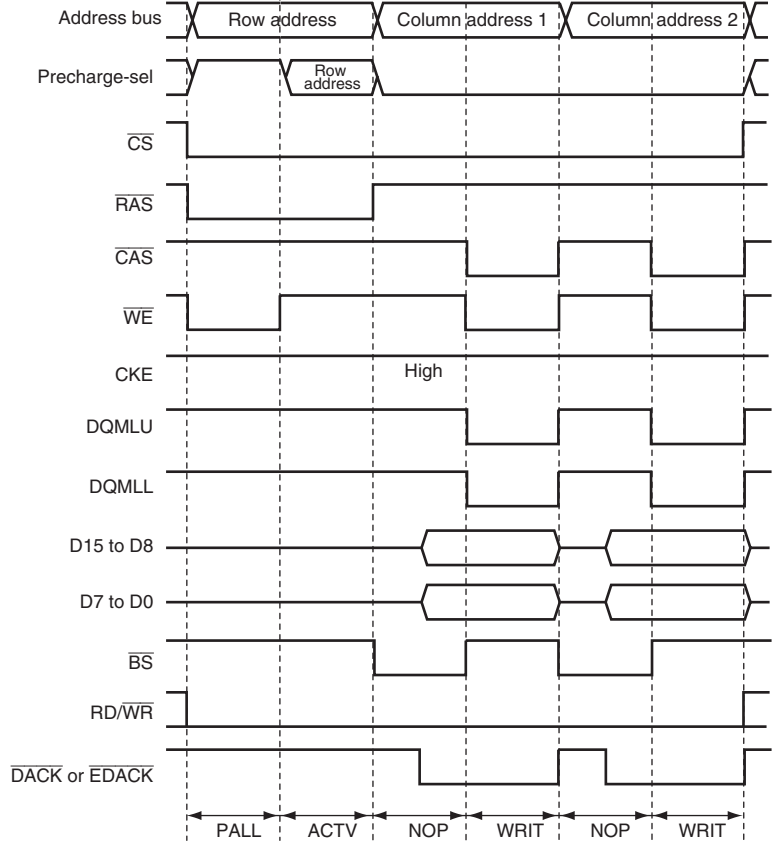
**Figure 9.82 Timing of Setting SDRAM Mode Register**

### 9.11.15 SDRAM Interface and Single Address Transfer by DMAC and EXDMAC

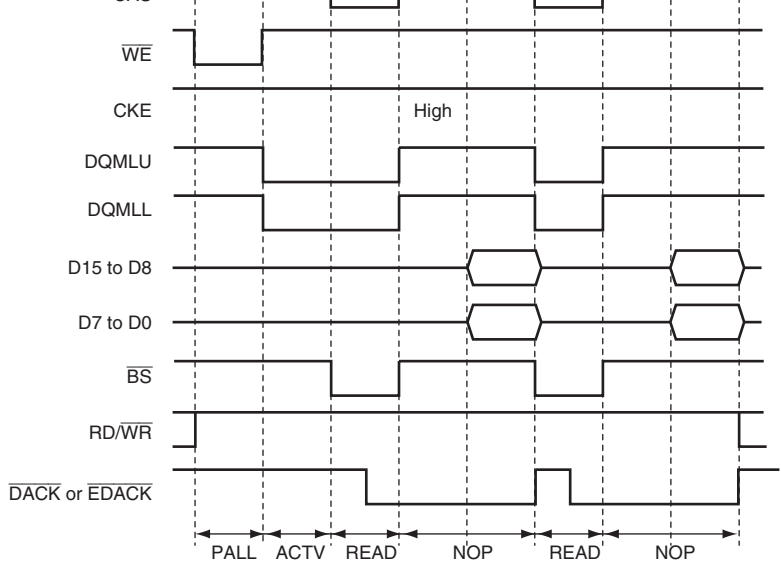
When fast-page mode ( $\text{BE} = 1$ ) is set for the SDRAM space, either fast-page access or full access can be selected, by the setting of bits DDS and EDDS in DRAMCR, for the single address transfer by the DMAC or EXDMAC where the SDRAM space is specified as the transfer source or destination. At the same time, the output timing of the  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  and  $\overline{\text{BS}}$  signals can be changed. When  $\text{BE} = 0$ , a full access to the SDRAM space is performed with a single address transfer regardless of the setting of bits DDS and EDDS. However, the output timing of the  $\overline{\text{DACK}}$ ,  $\overline{\text{EDACK}}$  and  $\overline{\text{BS}}$  signals can be changed by the setting of bits DDS and EDDS.

The assertion timing of the  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  signals can be changed by the bits DKC and EDKC in BCR1.

The output timing of the  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  signals can be independently set by the bits DKC and CKSPE in SDCR and bit DKC and EDKC in BCR1 regardless of the setting of bits DDS and EDDS.

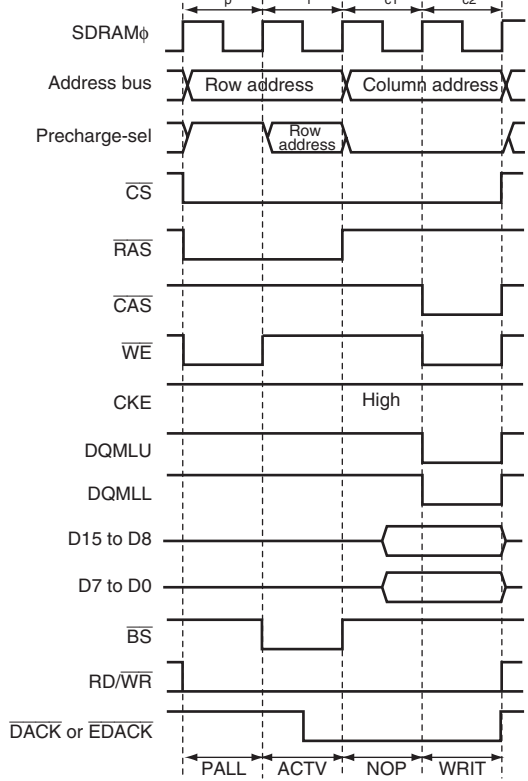


**Figure 9.83** Output Timing Example of  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  when  $\text{DDS} = 1$  or  $\text{E}$  (Write)

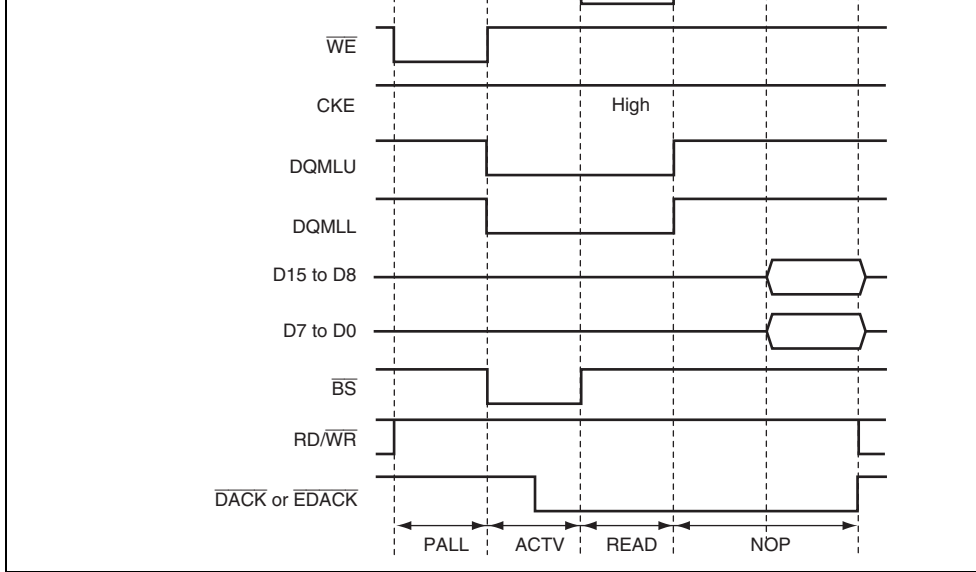


**Figure 9.84 Output Timing Example of  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  when  $\text{DDS} = 1$  or  $\text{ED}$  (Read, CAS Latency = 2)**

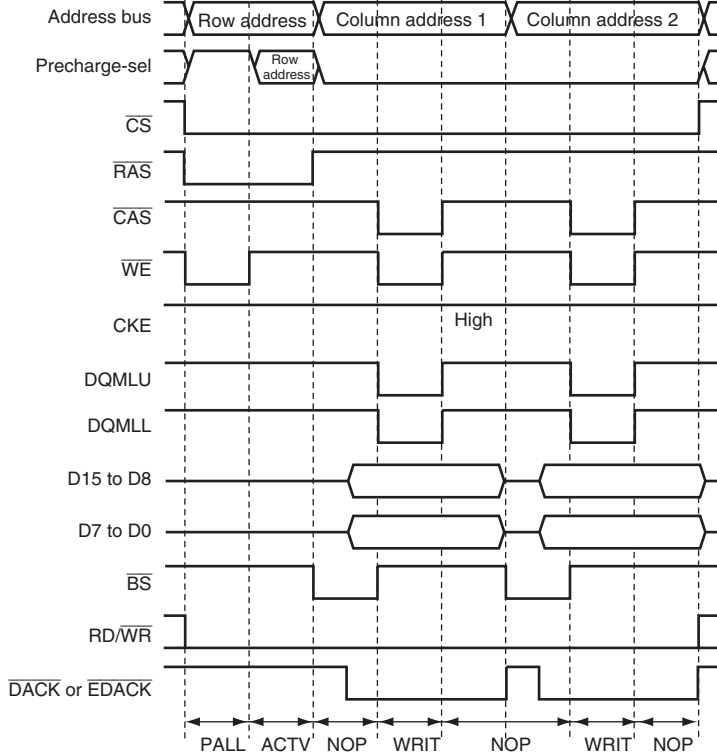




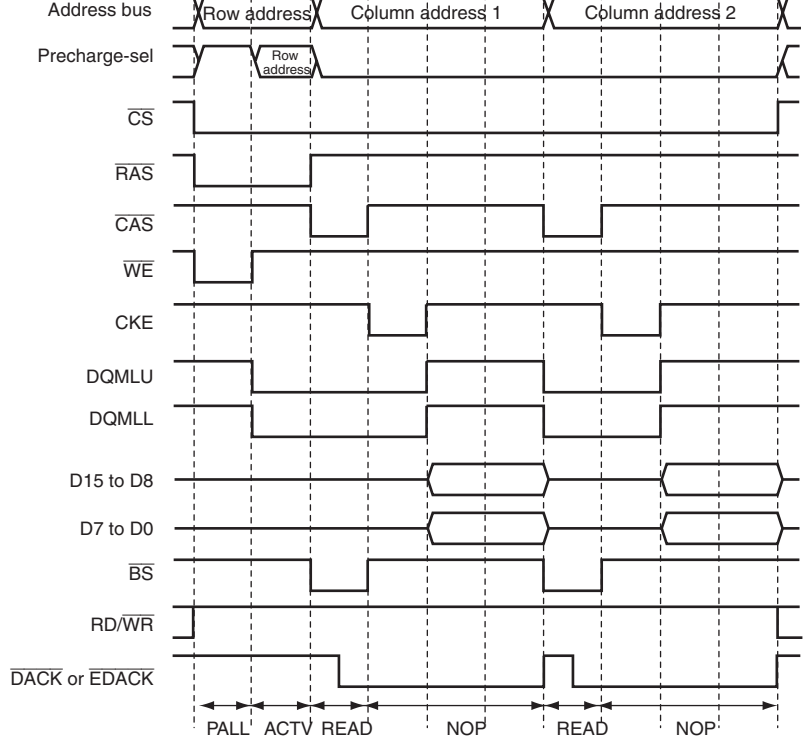
**Figure 9.85 Output Timing Example of  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  when  $\text{DDS} = 0$  or  $\text{E}$  (Write)**



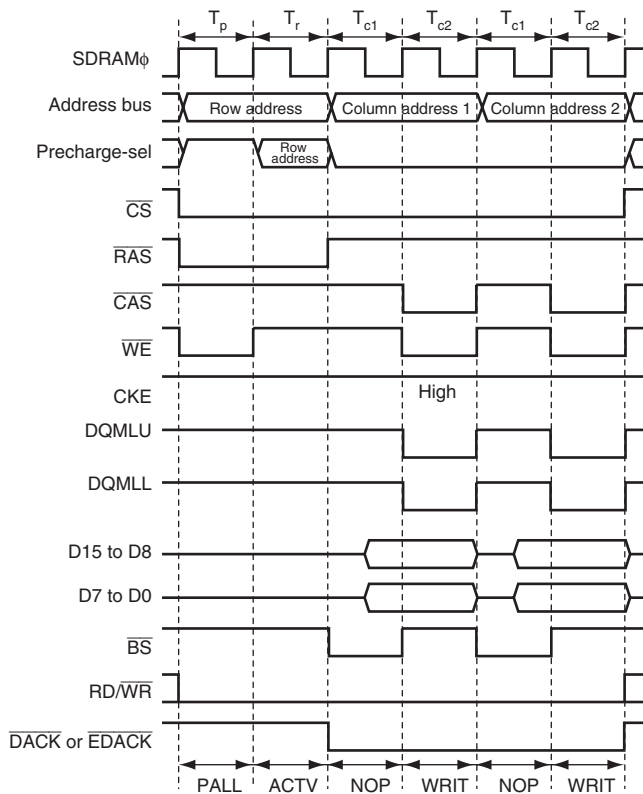
**Figure 9.86 Output Timing Example of  $\overline{DACK}$  and  $\overline{EDACK}$  when DDS = 0 or ED (Read, CAS Latency = 2)**



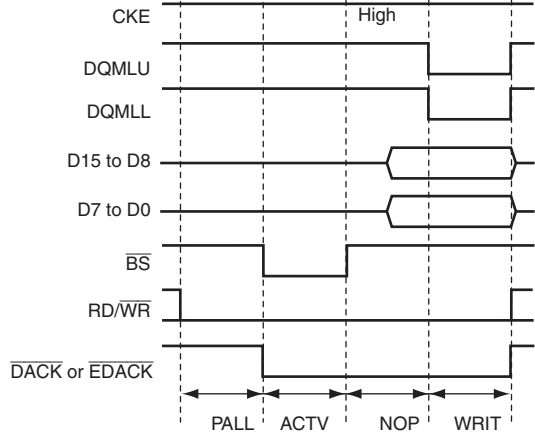
**Figure 9.87 Output Timing Example of  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  when  $\text{TRWL} = 1$  (**



**Figure 9.88 Output Timing Example of  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  when  $\text{CKSPE} = \text{Read}$ , CAS Latency = 2**



**Figure 9.89 Output Timing Example of  $\overline{\text{DACK}}$  and  $\overline{\text{EDACK}}$  when  $\text{DKC} = 1$  or  $\text{EDKC} = 1$  and  $\text{DDS} = 1$  or  $\text{EDDS} = 1$  (Write)**



**Figure 9.90 Output Timing Example of  $\overline{DACK}$  and  $\overline{EDACK}$  when  $DKC = 1$  or  $EDKC = 1$  and  $DDS = 0$  or  $EDDS = 0$  (Write)**

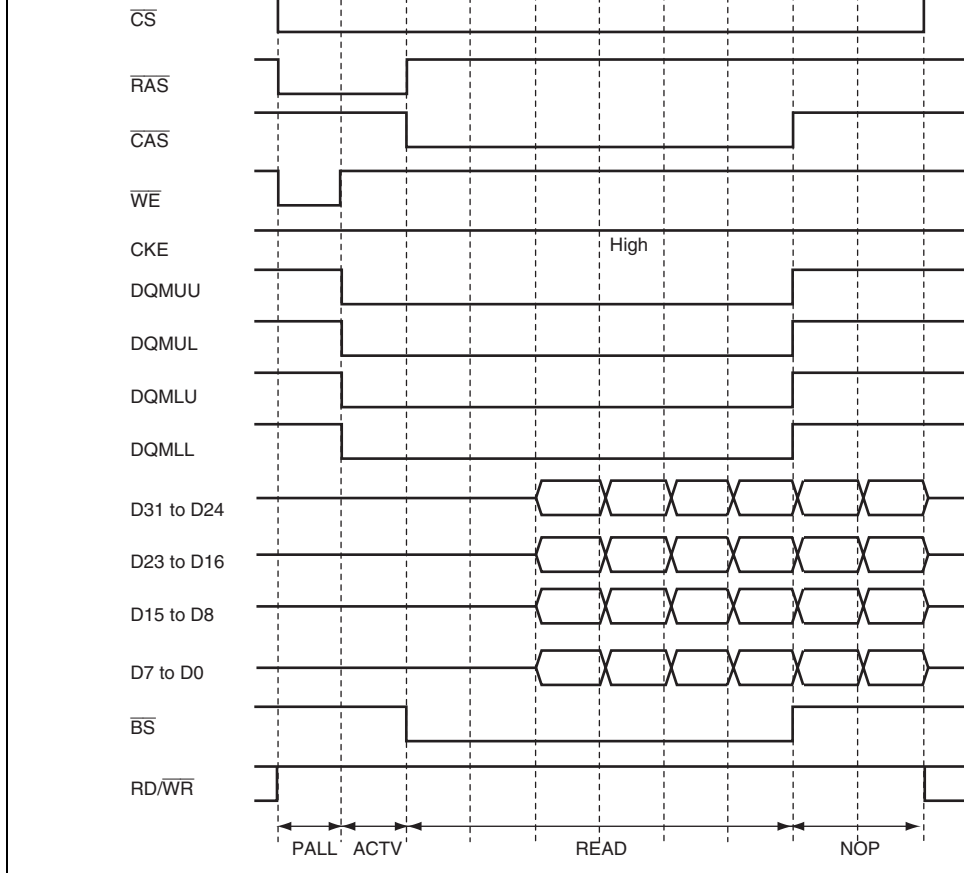
### 9.11.16 EXDMAC Cluster Transfer

Using an EXDMAC cluster transfer mode, data can be read from or written to consecutively. For details, see section 11, EXDMA Controller (EXDMAC).

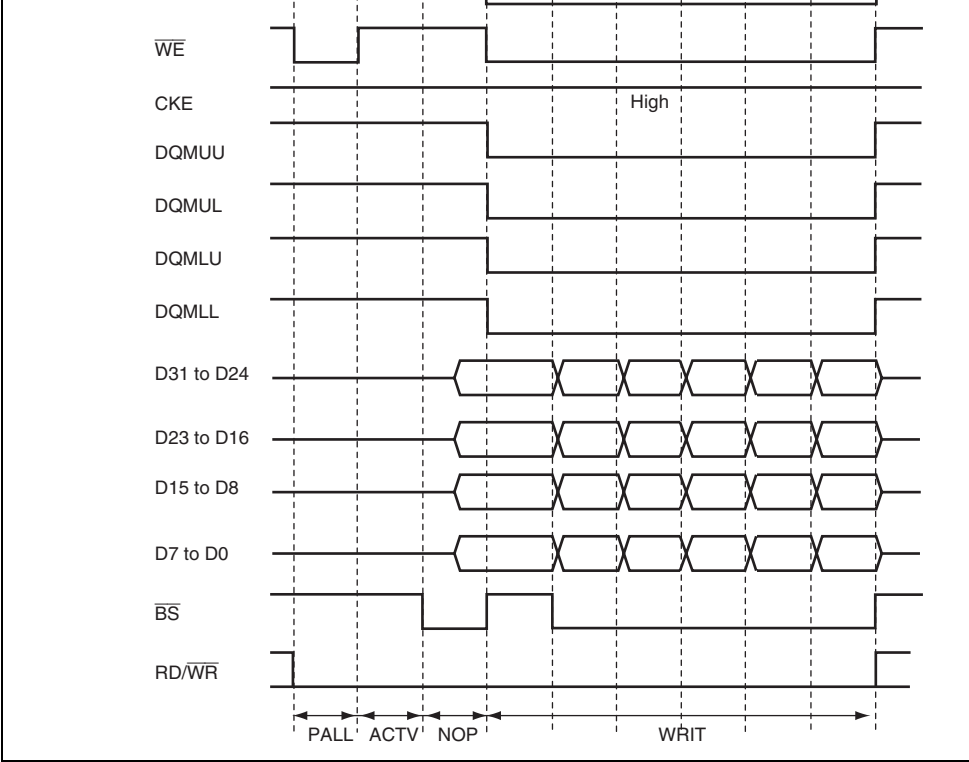
Figures 9.91 and 9.92 show a read/write timing using a cluster transfer.

For 1-cycle read or write, set the BE bit in DRAMCR to 1, clear the TRWL bit in SDCR, and set the CAS latency to 2. During a read cycle, the clock suspend mode cannot be used.

Do not change the bus controller register settings during a cluster transfer.



**Figure 9.91 Word-Size 6-Word Cluster Transfer  
(Read, BE = 1, EDDS = 1, CAS Latency = 2)**



**Figure 9.92 Word-Size 6-Word Cluster Transfer  
(Write, BE = 1, EDDS = 1)**



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 or EXDMAC single address transfer (write cycle)

Up to four idle cycles can be inserted under the conditions shown above. The number of cycles to be inserted should be specified to prevent data conflicts between the output data of 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, and 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 shown above.

Table 9.28 shows the correspondence between conditions 1 to 4 and number of idle cycles inserted for each area. Table 9.29 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.

			1	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.

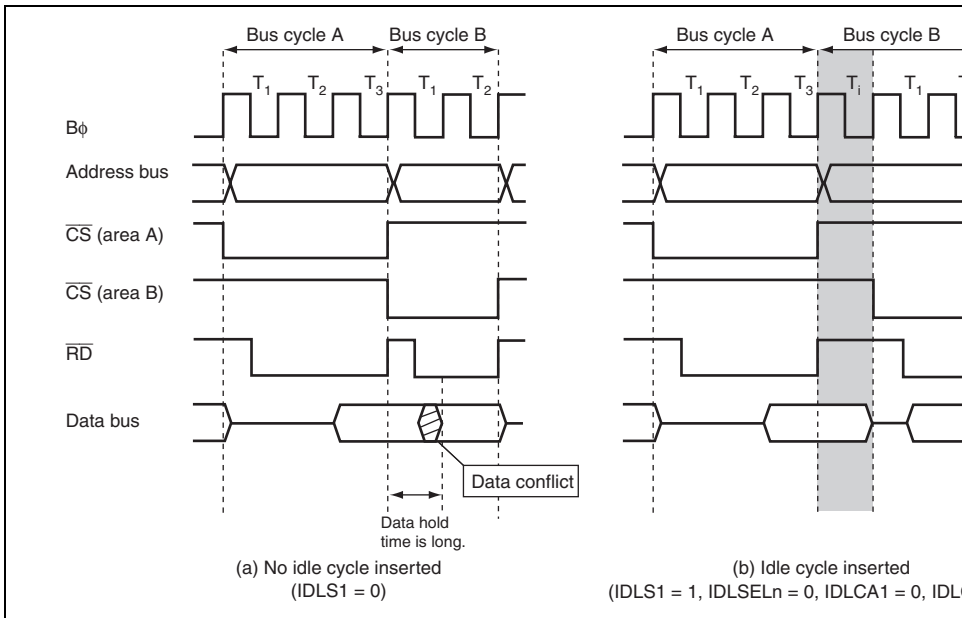
B: Number of idle cycle insertion B is selected.

Invalid: No idle cycle is inserted for the corresponding condition.

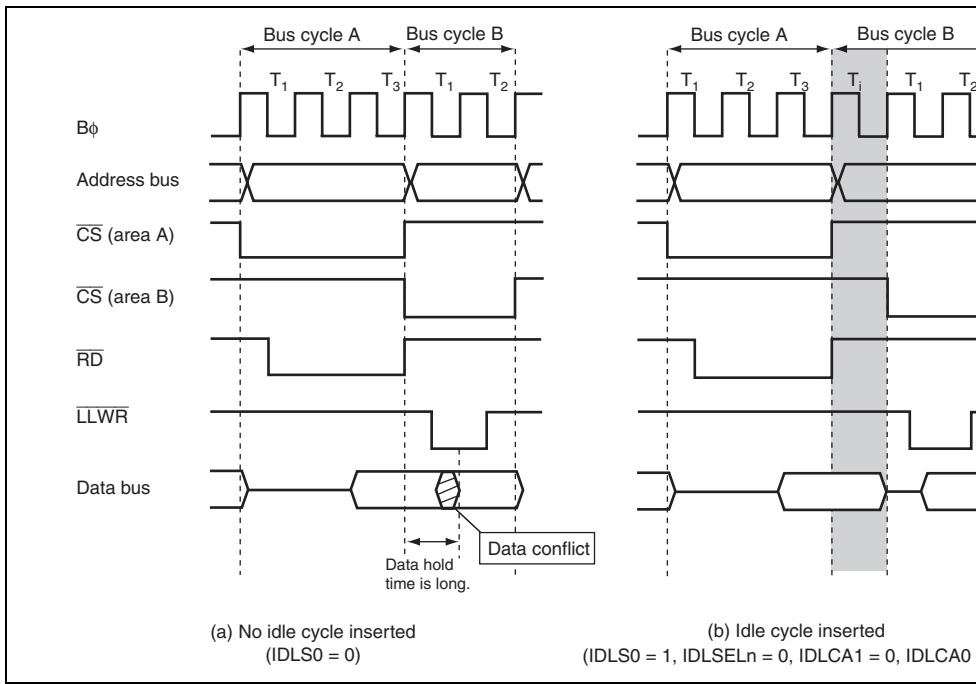
**Table 9.29 Number of Idle Cycles Inserted**

Bit Settings				
A		B		Number of Cycles
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

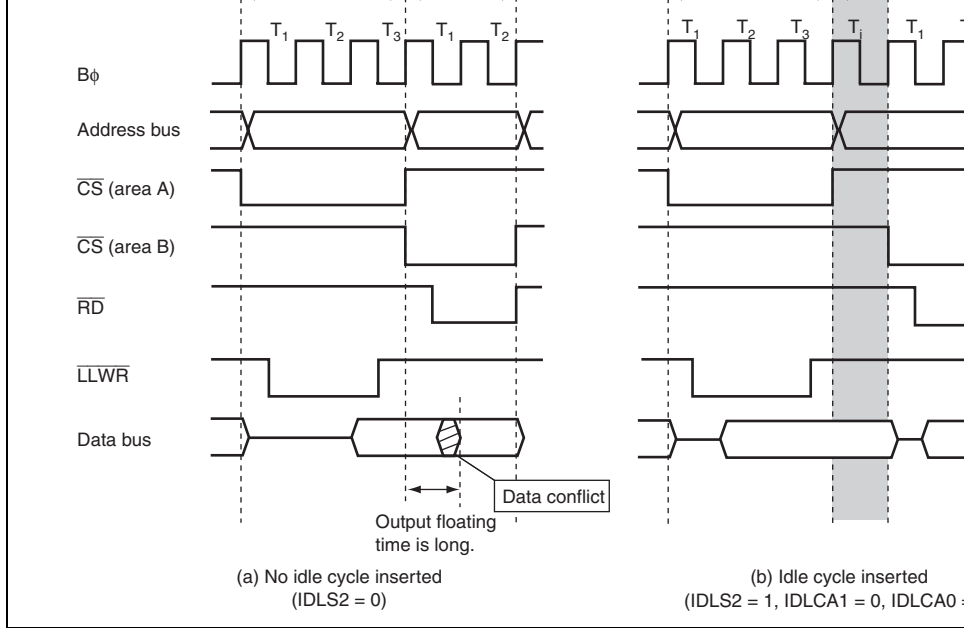
and a data conflict is prevented.



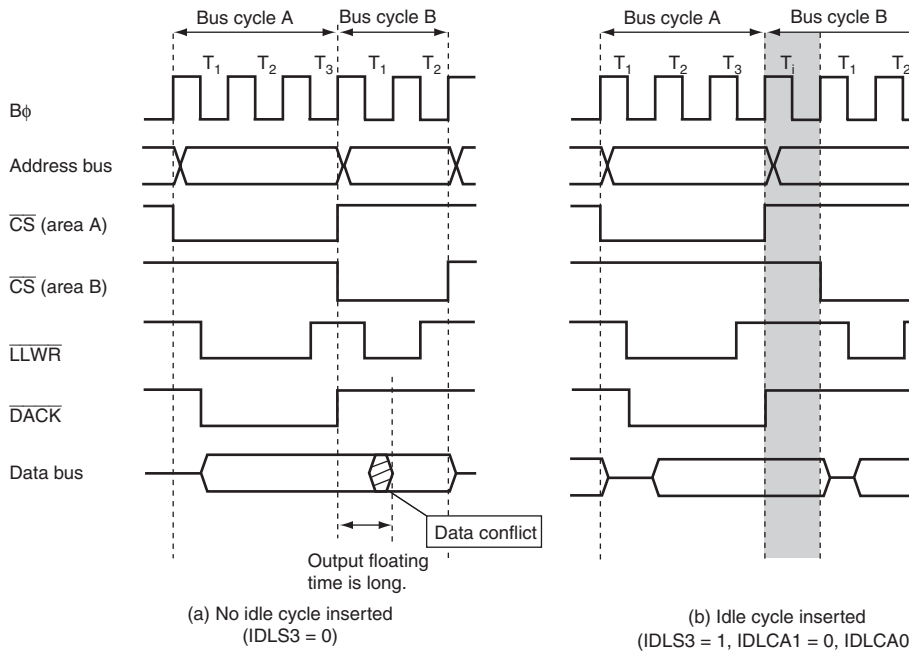
**Figure 9.93 Example of Idle Cycle Operation (Consecutive Reads in Different**



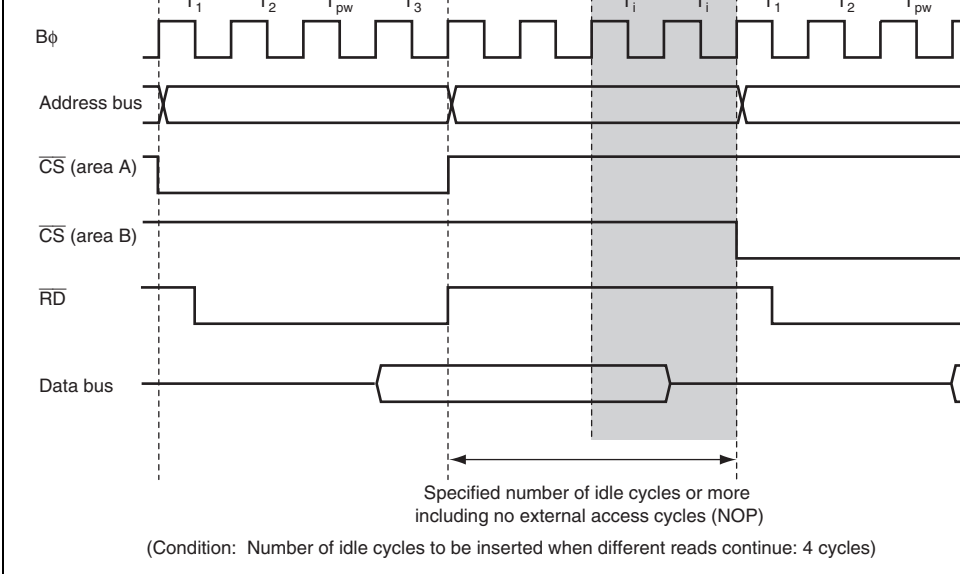
**Figure 9.94 Example of Idle Cycle Operation (Write after Read)**



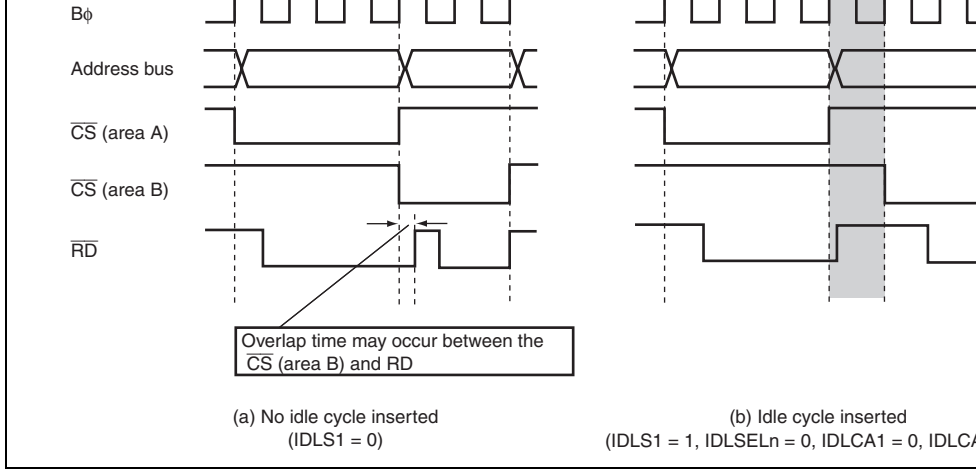
**Figure 9.95 Example of Idle Cycle Operation (Read after Write)**



**Figure 9.96 Example of Idle Cycle Operation (Write after Single Address Transfer)**



**Figure 9.97 Idle Cycle Insertion Example**

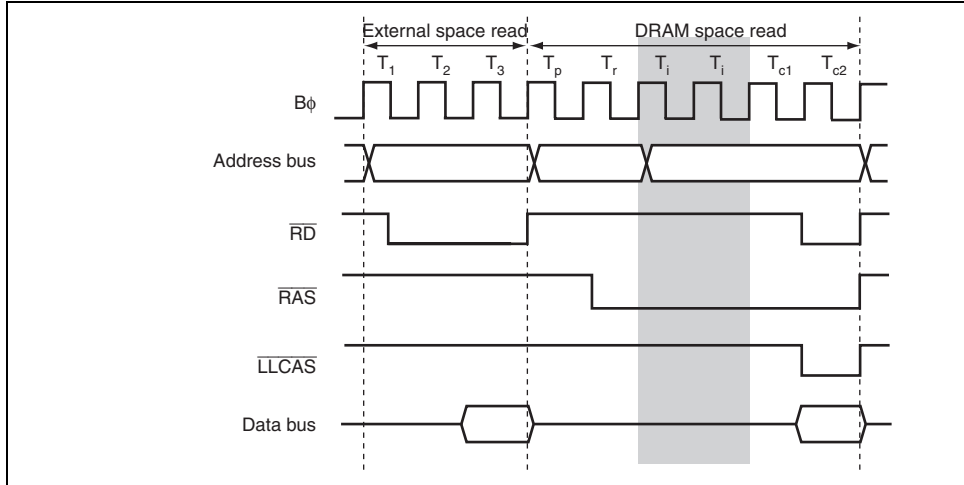


**Figure 9.98 Relationship between Chip Select ( $\overline{CS}$ ) and Read ( $\overline{RD}$ )**

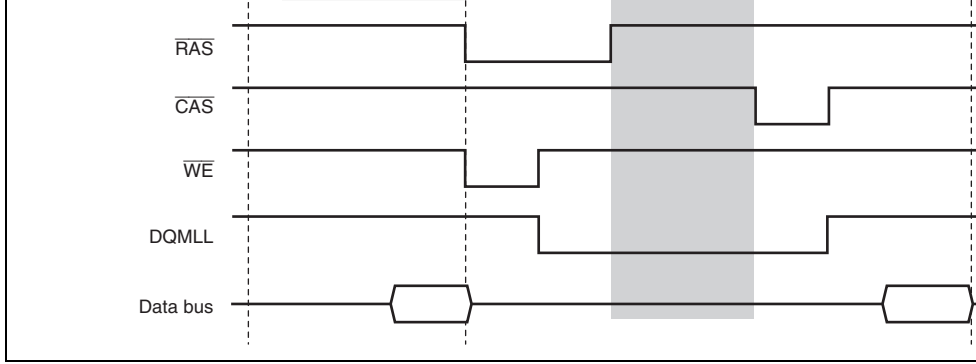


While the SDRAM space is accessed in a full access, the CS2 signal is driven low even cycle.

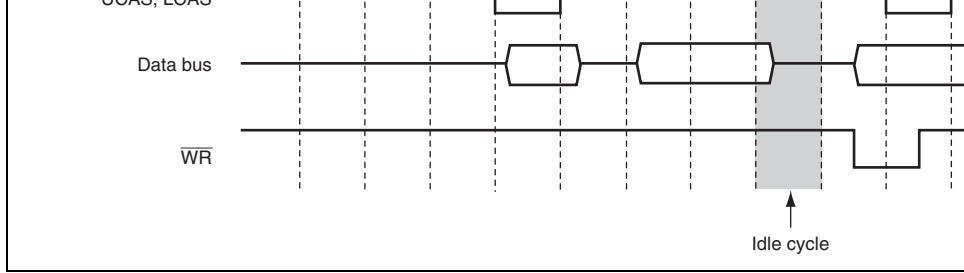
The idle cycle insertion is enabled even in a fast-page access in RAS down mode. The s number of idle cycles is inserted. Figure 9.101 shows a timing example of the idle cycle in RAS down mode.



**Figure 9.99 Example of DRAM Full Access after External Read (CAST =**



**Figure 9.100** Example of SDRAM Full Access after External Read (CAS Latency)



**Figure 9.101 Example of Idle Cycles in RAS Down Mode (Write after Read)**

										0	1	2 cycles
										1	0	3 cycles
										1	1	4 cycles
Normal/DRAM/SDRAM space read	Normal/DRAM/SDRAM space read	—	—	—	0	—	—	—	—	—	—	Disabled
		—	—	—	1	0	0	0	—	—	—	1 cycle in
							0	1				2 cycles
							1	0				3 cycles
							1	1				4 cycles
						1				0	0	0 cycle in
										0	1	2 cycles
										1	0	3 cycles
										1	1	4 cycles
Normal/DRAM/SDRAM space write	Normal/DRAM/SDRAM space read	—	0	—	—	—	—	—	—	—	—	Disabled
		—	1	—	—	—	0	0	—	—	—	1 cycle in
							0	1				2 cycles
							1	0				3 cycles
							1	1				4 cycles
Single address write	Normal/DRAM/SDRAM space write	0	—	—	—	—	—	—	—	—	—	Disabled
		1	—	—	—	—	0	0	—	—	—	1 cycle in
							0	1				2 cycles
							1	0				3 cycles
							1	1				4 cycles

$\overline{\text{LUCAS}}, \overline{\text{LLCAS}}$	High
$\overline{\text{DQMLU}}, \overline{\text{DQMLL}}$	High* <sup>2</sup>
$\overline{\text{AS}}$	High
$\overline{\text{RD}}$	High
$\overline{\text{BS}}$	High
$\overline{\text{RD/WR}}$	High* <sup>3</sup>
$\overline{\text{AH}}$	low
$\overline{\text{LHWR}}, \overline{\text{LLWR}}$	High
$\overline{\text{LUB}}, \overline{\text{LLB}}$	High
$\overline{\text{CKE}}$	High
$\overline{\text{OE}}$	High
$\overline{\text{RAS}}$	High/Low* <sup>4</sup>
$\overline{\text{CAS}}$	High
$\overline{\text{WE}}$	High
$\overline{\text{DACKn}}$ (n = 3 to 0)	High
$\overline{\text{EDACKn}}$ (n = 3 to 0)	High

- Notes:
1. Low when accessing the SDRAM in full access cycle
  2. Low when reading the SDRAM in full access cycle
  3. Low when accessing or writing to the DRAM/SDRAM in full access cycle
  4. The pin state varies depending on the DRAM space access/ area access other DRAM space, or RAS up mode/RAS down mode. For details, see figures 9.9 and 9.100.

In external extended mode, when the BRLE bit in BCR1 is set to 1, and the ICR bit for the corresponding pin is set to 1, the bus can be released to the external. Driving the  $\overline{\text{BREQ}}$  pin 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 ICR, see section 13, I/O Ports.

In the external bus released state, the CPU, DTC, DMAC can access the internal space using the internal bus. When any one of the CPU, DTC, DMAC, and EXDMAC 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.

In the external bus released state, certain operations are suspended as follows until the bus request from the external bus master is canceled:

- When a refresh is requested, refresh control is suspended.
- When the SLEEP instruction is executed to enter software standby mode or all-module clock-stop mode, control for software standby mode or all-module clock-stop mode is suspended.
- When SCKCR is written to set the clock frequencies, changing of clock frequencies is suspended. For SCKCR, see section 27, Clock Pulse Generator.

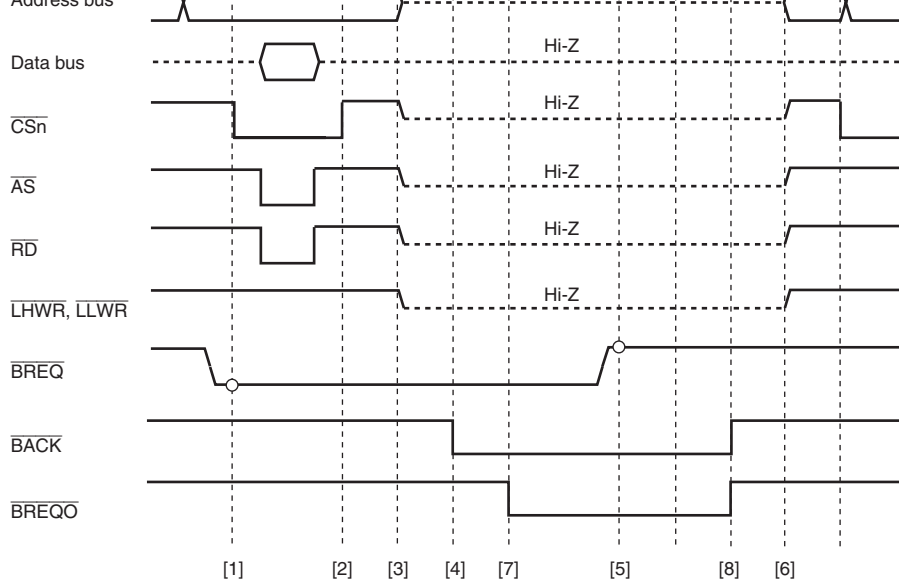
If the BREQOE bit in BCR1 is set to 1, the  $\overline{\text{BREQO}}$  pin can be driven low to request cancellation of the bus request when any of the following requests are issued.

- When any one of the CPU, DTC, DMAC, and EXDMAC attempts to access the external address space
- When a refresh is requested
- When a SLEEP instruction is executed to place the chip in software standby mode or module-clock-stop mode

Table 9.32 shows pin states in the external bus released state.

**Table 9.32 Pin States in Bus Released State**

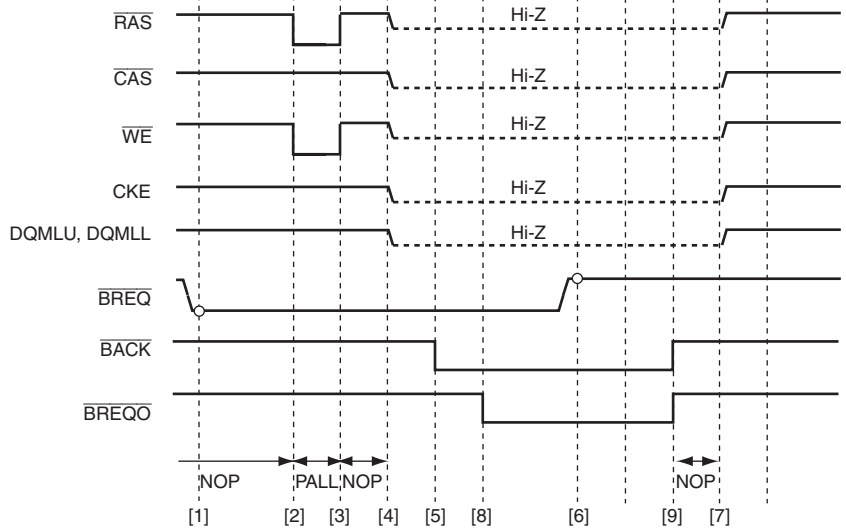
<b>Pins</b>	<b>Pin State</b>
A23 to A0	High impedance
D15 to D0	High impedance
$\overline{BS}$	High impedance
$\overline{CSn}$ (n = 7 to 0)	High impedance
$\overline{AS}$	High impedance
$\overline{AH}$	High impedance
$\overline{RD}/\overline{WR}$	High impedance
$\overline{LUCAS}$ , $\overline{LLCAS}$	High impedance
$\overline{RD}$	High impedance
$\overline{RAS}$	High impedance
$\overline{CAS}$	High impedance
$\overline{WE}$	High impedance
DQMLU, DQMLL	High impedance
CKE	High impedance
$\overline{OE}$	High impedance
$\overline{LUB}$ , $\overline{LLB}$	High impedance
$\overline{LHWR}$ , $\overline{LLWR}$	High impedance
$\overline{DACKn}$ (n = 3 to 0)	High
$\overline{EDACKn}$ (n = 3 to 0)	High



- [1] A low level of the  $\overline{\text{BREQ}}$  signal is sampled at the rising edge of the  $\text{B}\phi$  signal.
- [2] The bus control signals are driven high at the end of the external space access cycle. It takes two cycles or more after the low level of the  $\overline{\text{BREQ}}$  signal is sampled.
- [3] The  $\overline{\text{BACK}}$  signal is driven low, releasing bus to the external bus master.
- [4] The  $\overline{\text{BREQ}}$  signal state sampling is continued in the external bus released state.
- [5] A high level of the  $\overline{\text{BREQ}}$  signal is sampled.
- [6] The external bus released cycles are ended one cycle after the  $\overline{\text{BREQ}}$  signal is driven high.
- [7] When the external space is accessed by an internal bus master during external bus released while the  $\text{BR}$  bit is set to 1, the  $\overline{\text{BREQO}}$  signal goes low.
- [8] Normally the  $\overline{\text{BREQO}}$  signal goes high at the rising edge of the  $\overline{\text{BACK}}$  signal.

**Figure 9.102 Bus Released State Transition Timing (SRAM Interface is Not U**





- [1] A low level of the  $\overline{\text{BREQ}}$  signal is sampled at the rising edge of the  $\text{B}\phi$  signal.
- [2] The PALL command is issued.
- [3] The bus control signals are driven high at the end of the external access cycle. It takes two cycles or more after the low level of the  $\overline{\text{BREQ}}$  signal is sampled.
- [4] The  $\overline{\text{BACK}}$  signal is driven low, releasing bus to the external bus master.
- [5] The  $\overline{\text{BREQ}}$  signal state sampling is continued in the external bus released state.
- [6] A high level of the  $\overline{\text{BREQ}}$  signal is sampled.
- [7] The  $\overline{\text{BACK}}$  signal is driven high, ending external bus release cycle after one cycle.
- [8] When the external space is accessed by an internal bus master or a refresh cycle is requested during external bus released while the BREQOE bit is set to 1, the  $\overline{\text{BREQO}}$  signal goes low.
- [9] Normally the  $\overline{\text{BREQO}}$  signal goes high at the rising edge of the  $\overline{\text{BACK}}$  signal.

**Figure 9.103 Bus Released State Transition Timing (SRAM Interface is Us**

Access Space	Access	Number of Access Cycles
On-chip ROM space	Read	One $t_{\phi}$ cycle
	Write	Three $t_{\phi}$ cycles
On-chip RAM space	Read	One $t_{\phi}$ cycle
	Write	One $t_{\phi}$ cycle

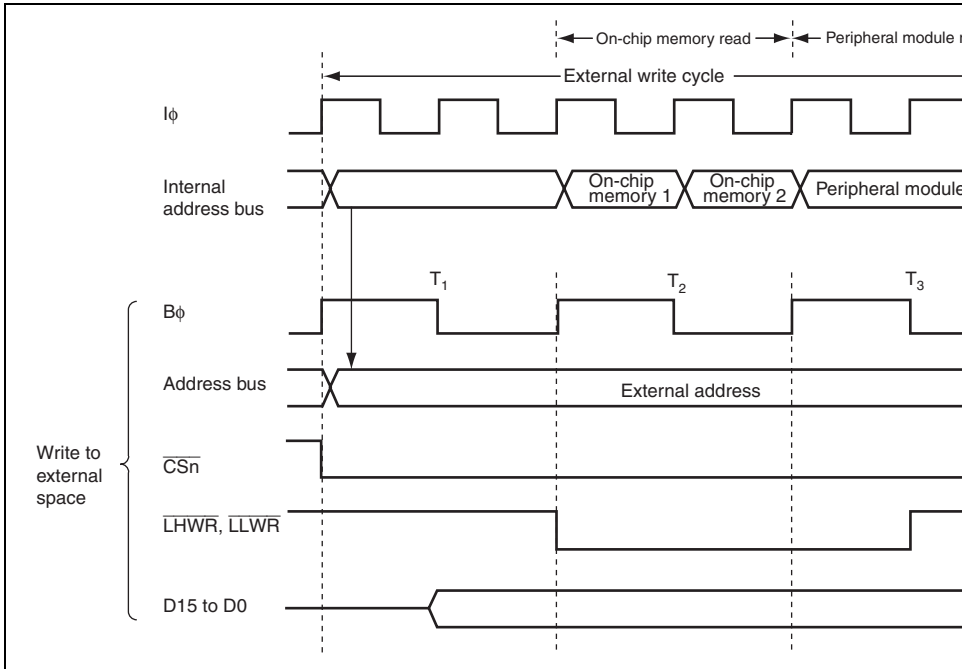
In access to the registers for on-chip peripheral modules, the number of access cycles differs according to the register to be accessed. When the dividing ratio of the operating clock of the master and that of a peripheral module is 1 : n, synchronization cycles using a clock divider of n-1 are inserted for register access in the same way as for external bus clock division.

Table 9.34 lists the number of access cycles for registers of on-chip peripheral modules.

**Table 9.34 Number of Access Cycles for Registers of On-Chip Peripheral Modules**

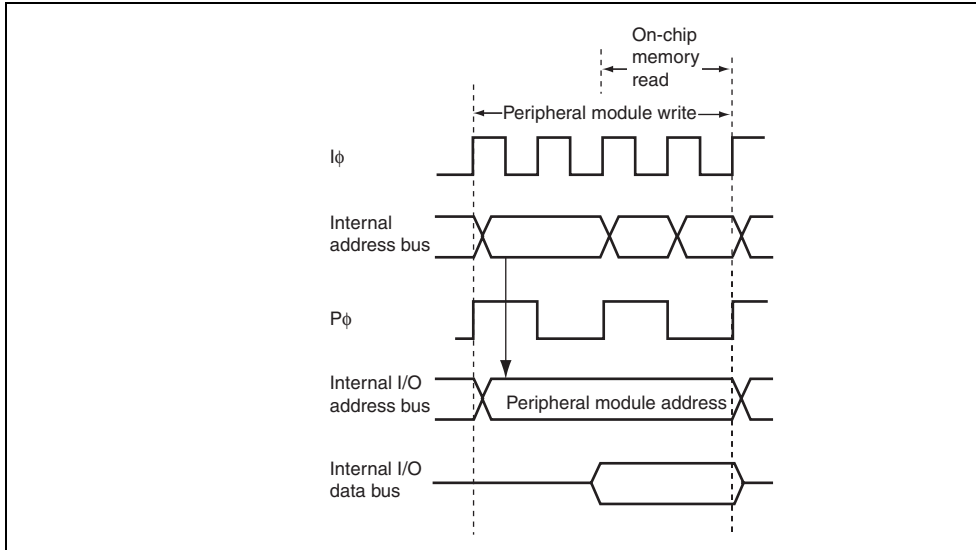
Module to be Accessed	Number of Cycles		
	Read	Write	Write Data Buffer Full
DMAC and EXDMAC registers	Two $t_{\phi}$		Disabled
MCU operating mode, clock pulse generator, power-down control registers, interrupt controller, bus controller, and DTC registers	Two $t_{\phi}$	Three $t_{\phi}$	Disabled
I/O port registers of PFCR and WDT	Two $P_{\phi}$	Three $P_{\phi}$	Disabled
I/O port registers other than PFCR and PORTM, TPU, PPG0, TMR0, TMR1, SCI0 to SCI2, SCI4, IIC2, A/D_0, and D/A registers	Two $P_{\phi}$		Enabled
I/O port registers of PORTM, TMR2, TMR3, USB, SCI5, SCI6, A/D_1, and PPG1 registers	Three $P_{\phi}$		Enabled

continues for two cycles or longer, and there is an internal access next, an external write is executed in the first two cycles. However, from the next cycle onward, internal accesses (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 9.104 Example of Timing when Write Data Buffer Function is Used**

is performed in the first two cycles. However, from the next cycle onward an internal memory read is performed in parallel with an external access and internal I/O register write are executed in parallel rather than waiting until the external access ends.



**Figure 9.105 Example of Timing when Peripheral Module Write Data Buffer Function is Used**

### 9.16.1 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a 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 the request is canceled.

The priority of the internal bus arbitration:

DMAC > DTC > CPU

The priority of the external bus arbitration:

Refresh > EXDMAC > External bus release request > External access by the CPU, DMAC

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. If an external bus release request, an EXDMAC access, and a refresh cycle request continue, an external bus access by the CPU, DTC, and DMAC can be given priority to execute the bus cycles alternatively between them by setting the EBCCS bit in BCR2. In this case, the priorities among the refresh, EXDMAC access, and external bus release request do not change.

An internal bus access by internal bus masters and an external bus access by an external bus master, a refresh cycle, or an EXDMAC access can be executed in parallel.

arbiter transfer the bus to the EXDMAC.

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 instruction 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, 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.

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 on 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 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) EXDMAC**

The EXDMAC sends the external bus arbiter a request for the bus when an activation request is generated. If an internal bus master accesses the external space, the bus is passed to the internal bus master when the bus master can release the bus. Some EXDMAC transfers are continued once the internal bus master releases control of the bus. Some EXDMAC transfers are divided and it releases the bus for each bus cycle.

- Transfers are continued without bus release between a read cycle and the subsequent write cycle in dual address mode
- Transfers are continued without bus release in cluster transfer mode

### (5) External Bus Release

When the  $\overline{\text{BREQ}}$  pin goes low and an external bus release request is issued while the BRM is set to 1, 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.

### (6) Refresh

When area 2 is specified as the DRAM space or SDRAM space with the RFSHE bit in RFSHE register set to 1, RTCNT starts to count up. When the RTCOR value matches RTCNT, a bus request is sent to the bus arbiter.

A refresh cycle is inserted on completion of the external bus cycle. A refresh cycle is not consecutively inserted. Once a refresh cycle is inserted, the bus is passed to another bus master. When the bus is passed, if there is no bus request from other bus masters, NOP cycles are inserted.



other than an instruction fetch access.

## (2) Mode Settings

The burst read-burst write mode of synchronous DRAM is not supported.

When setting the mode register of synchronous DRAM, the burst read-single write mode is selected and the burst length must be 1.

## (3) External Bus Release Function and All-Module-Clock-Stop Mode

In this LSI, if the ACSE bit in MSTPCRA is set to 1 and a SLEEP instruction is executed, the sleep state after shutting off the clocks to all peripheral modules (MSTPCRA and MSTPCRB = H'FFFFFFF) or allowing operation of the 8-bit timer module alone (MSTPCRA and MSTPCRB = H'F[C to F]FFFFFFF), the all-module-clock-stop mode is entered in which the clock for the external bus controller and I/O ports is also stopped. For details, see section 28, 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 MSTPCRA 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.

## (4) External Bus Release Function and Software Standby Mode

In this LSI, internal bus master operation does not stop even while the bus is released, 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.

#### (6) $\overline{\text{BREQO}}$ Output Timing

When the BREQOE bit is set to 1 and the  $\overline{\text{BREQO}}$  signal is output, both the  $\overline{\text{BREQO}}$  and  $\overline{\text{BREQ}}$  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.

#### (7) Refresh Settings

In single-chip activation mode, the setting of the RFSHE bit in REFCR should be made after setting the EXPE bit in SYSCR to 1. For SYSCR, see section 3, MCU Operating Modes.

#### (8) Refresh Timer Settings

The setting of bits RTCK2 to RTCK0 in REFCR should be made after RTCNT and RTCOR have been set. When changing RTCNT and RTCOR, the counter operation should be halted. When changing bits RTCK2 to RTCK0, external access and external bus release by the EXDMR should be prohibited. The write data buffer function should be used after the write data buffer function is disabled and the external space is read.

#### (9) Switching Between Refresh Timer and Interval Timer

When changing the RFSHE bit in REFCR from 1 to 0, a refresh cycle may be inserted and the change is reflected. After this, when using RTCNT as an interval timer, the compare match (CMF) may be set to 1. Therefore, confirm the state before setting the CMIE bit to 1.

to SCKCR. RAS down mode should be set again after clock frequencies are set. For SC section 27, Clock Pulse Generator.

## **(12) Cluster Transfer to SDRAM Space**

Cluster transfer mode is available for the SDRAM with CAS latency of 2. When the SD used in cluster transfer mode, the SDRAM with CAS latency of 2 should be used. In cluster transfer mode, the write-precharge output delay function by the TRWL bit is not available. TRWL bit must be cleared to 0.



- 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 all four channels
- 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  $\overline{\text{DACK}}$  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 is incremented and returns to the transfer start address
    - Up to 64K transfers (65,536 bytes/words/longwords) can be transferred at a repeat size
  - Block transfer mode: One block data is transferred at a single transfer request
    - Up to 64K transfers (65,536 bytes/words/longwords) can be transferred at a block size

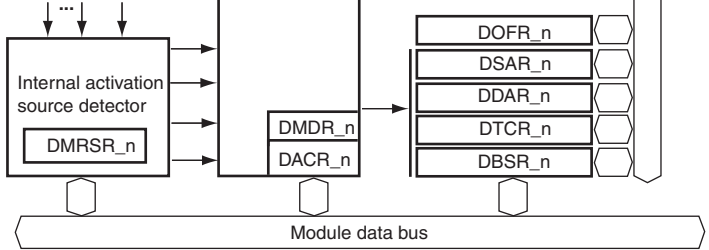
respective boundary

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 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 transfer is completed, or when the extended repeat area overflows.

- Module stop state can be set.



[Legend]

- |          |                                    |                      |                          |
|----------|------------------------------------|----------------------|--------------------------|
| DSAR_n:  | DMA source address register        | $\overline{DREQn}$ : | DMA transfer request     |
| DDAR_n:  | DMA destination address register   | $\overline{DACKn}$ : | DMA transfer acknowledge |
| DOFR_n:  | DMA offset register                | TENDn:               | DMA transfer end         |
| DTCR_n:  | DMA transfer count register        |                      | n = 0 to 3               |
| DBSR_n:  | DMA block size register            |                      |                          |
| DMDR_n:  | DMA mode control register          |                      |                          |
| DACR_n:  | DMA address control register       |                      |                          |
| DMRSR_n: | DMA module request select register |                      |                          |

**Figure 10.1 Block Diagram of DMAC**

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 acknowledge
	DMA transfer end 1	$\overline{\text{TEND1}}$	Output	Channel 1 transfer end
2	DMA transfer request 2	$\overline{\text{DREQ2}}$	Input	Channel 2 external request
	DMA transfer acknowledge 2	$\overline{\text{DACK2}}$	Output	Channel 2 single address acknowledge
	DMA transfer end 2	$\overline{\text{TEND2}}$	Output	Channel 2 transfer end
3	DMA transfer request 3	$\overline{\text{DREQ3}}$	Input	Channel 3 external request
	DMA transfer acknowledge 3	$\overline{\text{DACK3}}$	Output	Channel 3 single address acknowledge
	DMA transfer end 3	$\overline{\text{TEND3}}$	Output	Channel 3 transfer end



- 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)

#### **Channel 2:**

- DMA source address register\_2 (DSAR\_2)
- DMA destination address register\_2 (DDAR\_2)
- DMA offset register\_2 (DOFR\_2)
- DMA transfer count register\_2 (DTCR\_2)
- DMA block size register\_2 (DBSR\_2)
- DMA mode control register\_2 (DMDR\_2)
- DMA address control register\_2 (DACR\_2)
- DMA module request select register\_2 (DMRSR\_2)

### 10.3.1 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	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	23	22	21	20	19	18	17	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	15	14	13	12	11	10	9	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	7	6	5	4	3	2	1	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	23	22	21	20	19	18	17
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	15	14	13	12	11	10	9
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	15	14	13	12	11	10	9	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	7	6	5	4	3	2	1	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Although DTCR can always be read from by the CPU, it must be read from in longword must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	23	22	21	20	19	18	17	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	15	14	13	12	11	10	9	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	7	6	5	4	3	2	1	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

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	
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	
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	All 0	R/W	Specify the repeat size or block size. When H'0001 is set, the repeat or block size is one word, or one longword. When H'0000 is set, the value means the maximum value (refer to table 10.1). When the DMA is in operation, the setting is fixed.
15 to 0	BKSZ15 to BKSZ0	All 0	R/W	Indicate the remaining repeat or block size while DMA is in operation. The value is decremented every time data is transferred. When the remaining value becomes 0, the value of the BKSZH bits is loaded with the same value as the BKSZH bits.

DMDR controls the DMAC operation.

- DMDR\_0

Bit	31	30	29	28	27	26	25	
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	
Bit	23	22	21	20	19	18	17	
Bit Name	ACT	—	—	—	ERRF	—	ESIF	
Initial Value	0	0	0	0	0	0	0	
R/W	R	R	R	R	R/(W)*	R	R/(W)*	
Bit	15	14	13	12	11	10	9	
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Bit	7	6	5	4	3	2	1	
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R	R	R/W	R/W	

Note: \* Only 0 can be written to this bit after having been read as 1, to clear the flag.

Bit Name	DTF21	DTF20	WDS1	WDS0	FOE1	FOE0	DTF1	DTF0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMA2	DMA1	DMA0
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.



transfer.  
In block transfer mode, if writing 0 to this bit while data is being transferred, this bit is cleared to 0 after the next 1-block size data transfer.

If an event which stops (sustains) a transfer occurs externally, this bit is automatically cleared to 0 at the end of the transfer.

Operating modes and transfer methods must not be changed while this bit is set to 1.

0: Disables a data transfer

1: Enables a data transfer (DMA is in operation)

[Clearing conditions]

- When the specified total transfer size of transfer is completed
- When a transfer is stopped by an overflow error by a repeat size end
- When a transfer is stopped by an overflow error by an extended repeat size end
- When a transfer is stopped by a transfer size error interrupt
- When clearing this bit to 0 to stop a transfer

In block transfer mode, this bit changes after the next block transfer.

- When an address error or an NMI interrupt is requested
- In the reset state or hardware standby mode

28	—	0	R/W	Reserved Initial value should not be changed.
27	DREQS	0	R/W	DREQ Select Selects whether a low level or the falling edge of the DREQ signal used in external request mode is used. 0: Low level detection 1: Falling edge detection (the first transfer after the transfer enabled is detected on a low level)
26	NRD	0	R/W	Next Request Delay Selects the accepting timing of the next transfer request. 0: Starts accepting the next transfer request after the completion of the current transfer 1: Starts accepting the next transfer request only after completion of the current transfer
25, 24	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
23	ACT	0	R	Active State Indicates the operating state for the channel. 0: Waiting for a transfer request or a transfer data state by clearing the DTE bit to 0 1: Active state
22 to 20	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

generated

[Clearing condition]

- When clearing to 0 after reading ERRF = 1

[Setting condition]

- When an address error or an NMI interrupt generated

However, when an address error or an NMI interrupt has been generated in DMAC module stop mode, the bit is not set to 1.

---

18	—	0	R	Reserved
----	---	---	---	----------

This bit is always read as 0 and cannot be modified.

---

17	ESIF	0	R/(W)*	Transfer Escape Interrupt Flag
----	------	---	--------	--------------------------------

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 the transfer size. The bit is set to 1 when a transfer escape end interrupt is requested.

0: A transfer escape end interrupt has not been requested  
1: A transfer escape end interrupt has been requested

[Clearing conditions]

- When setting the DTE bit to 1
- When clearing to 0 before reading ESIF = 1

[Setting conditions]

- When a transfer size error interrupt is requested
  - When a repeat size end interrupt is requested
  - When a transfer end interrupt by an external interrupt or area overflow is requested
-

- When setting the DTE bit to 1
- When clearing to 0 after reading DTIF = 1 [Setting condition]
- When DTCR reaches 0 and the transfer is completed

15	DTSZ1	0	R/W	Data Access Size 1 and 0
14	DTSZ0	0	R/W	Select the data access size for a transfer. 00: Byte size (eight bits) 01: Word size (16 bits) 10: Longword size (32 bits) 11: Setting prohibited
13	MDS1	0	R/W	Transfer Mode Select 1 and 0
12	MDS0	0	R/W	Select the transfer mode. 00: Normal transfer mode 01: Block transfer mode 10: Repeat transfer mode 11: Setting prohibited

- In normal or repeat transfer mode, the total transfer size set in DTCR is less than the data access size
  - In block transfer mode, the total transfer size set in DTCR is less than the block size
- 0: Disables a transfer size error interrupt request  
1: Enables a transfer size error interrupt request

10	—	0	R	Reserved
This bit is always read as 0 and cannot be modified.				
9	ESIE	0	R/W	Transfer Escape Interrupt Enable
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.				
0: Disables a transfer escape end interrupt 1: Enables a transfer escape end interrupt				
8	DTIE	0	R/W	Data Transfer End Interrupt Enable
Enables/disables a transfer end interrupt request. 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.				
0: Disables a transfer end interrupt 1: Enables a transfer end interrupt				

				11: External request
5	DTA	0	R/W	<p>Data Transfer Acknowledge</p> <p>This bit is valid in DMA transfer by the on-chip interrupt source. This bit enables or disables to source flag selected by DMRSR.</p> <p>0: To clear the source in DMA transfer is disabled. Since the on-chip module interrupt source is cleared in DMA transfer, it should be cleared CPU or DTC transfer.</p> <p>1: To clear the source in DMA transfer is enabled. Since the on-chip module interrupt source is in DMA transfer, it does not require an interrupt CPU or DTC transfer.</p>
4, 3	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>

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.

R/W	R	R	R/W	R/W	R	R	R/W	R
Bit	15	14	13	12	11	10	9	
Bit Name	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SA
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R
Bit	7	6	5	4	3	2	1	
Bit Name	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DA
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R

Bit	Bit Name	Initial Value	R/W	Description
31	AMS	0	R/W	<p>Address Mode Select</p> <p>Selects address mode from single or dual address mode. In single address mode, the <math>\overline{DACK}</math> pin is according to the DACK bit.</p> <p>0: Dual address mode 1: Single address mode</p>
30	DIRS	0	R/W	<p>Single Address Direction Select</p> <p>Specifies the data transfer direction in single address mode. This bit is ignored in dual address mode.</p> <p>0: Specifies DSAR as source address 1: Specifies DDAR as destination address</p>
29 to 27	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>





transfer is requested after 1-block data transfer. When this bit is set to 1, the DTE bit in DMDR is cleared. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested.

0: Disables a repeat size end interrupt

1: Enables a repeat size end interrupt

25	ARS1	0	R/W	Area Select 1 and 0
24	ARS0	0	R/W	Specify the block area or repeat area in block transfer mode. 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
23, 22	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
21	SAT1	0	R/W	Source Address Update Mode 1 and 0
20	SAT0	0	R/W	Select the update method of the source address (DSAR). When DSAR is not specified as the transfer mode, this bit is ignored in single address mode, this bit is ignored. 00: Source address is fixed 01: Source address is updated by adding the data access size 10: Source address is updated by adding 1, 2, 4, or 8 according to the data access size 11: Source address is updated by subtracting 1, 2, 4, or 8 according to the data access size

10: Destination address is updated by adding  
according to the data access size

11: Destination address is updated by subtracting  
or 4 according to the data access size

---

15	SARIE	0	R/W	<p>Interrupt Enable for Source Address Extended Overflow</p> <p>Enables/disables an interrupt request for an extended repeat area overflow on the source address.</p> <p>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.</p> <p>When block transfer mode is used with the extended repeat area function, an interrupt is requested at the completion of a 1-block size transfer. When set to 0, 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.</p> <p>When the extended repeat area is not specified, this bit is ignored.</p> <p>0: Disables an interrupt request for an extended repeat area overflow on the source address</p> <p>1: Enables an interrupt request for an extended repeat area overflow on the source address</p>
14, 13	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>

---

When an overflow in the extended repeat area with the SARIE bit set to 1, an interrupt can be requested. Table 10.3 shows the settings and the extended repeat area.

7	DARIE	0	R/W	<p>Destination Address Extended Repeat Area C Interrupt Enable</p> <p>Enables/disables an interrupt request for an extended repeat area overflow on the destination address.</p> <p>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 SARIE bit in DMDR is set to 1 to indicate an interrupt request for an extended repeat area overflow on the destination address is requested.</p> <p>When block transfer mode is used with the extended repeat area function, an interrupt is requested at the completion of a 1-block size transfer. When 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.</p> <p>When the extended repeat area is not specified, this bit is ignored.</p> <p>0: Disables an interrupt request for an extended repeat area overflow on the destination address</p> <p>1: Enables an interrupt request for an extended repeat area overflow on the destination address</p>
6, 5	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>

area for address addition and subtraction, resp  
When an overflow in the extended repeat area  
with the DARIE bit set to 1, an interrupt can be  
requested. Table 10.3 shows the settings and a  
the extended repeat area.

---

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
111xx	Setting prohibited

[Legend]

x: Don't care



address	<ul style="list-style-type: none"> <li>Repeat transfer (activated by CPU)</li> <li>Block transfer</li> </ul>	<ul style="list-style-type: none"> <li>On-chip module interrupt</li> <li>External request</li> </ul>	<ul style="list-style-type: none"> <li>size: 1 to 4 Gbytes or not specified</li> <li>Offset addition</li> <li>Extended repeat area function</li> </ul>
	Repeat or block size = 1 to 65,536 bytes, 1 to 65,536 words, or 1 to 65,536 longwords		

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>	DSAR/ $\overline{DACK}$
----------------	--	----------------------------

When the auto request setting is selected as the activation source, the cycle stealing or burst can be selected. When the total transfer size is not specified (DTCR = H'00000000), the counter is stopped and the transfer is continued without the limitation of the transfer counter.

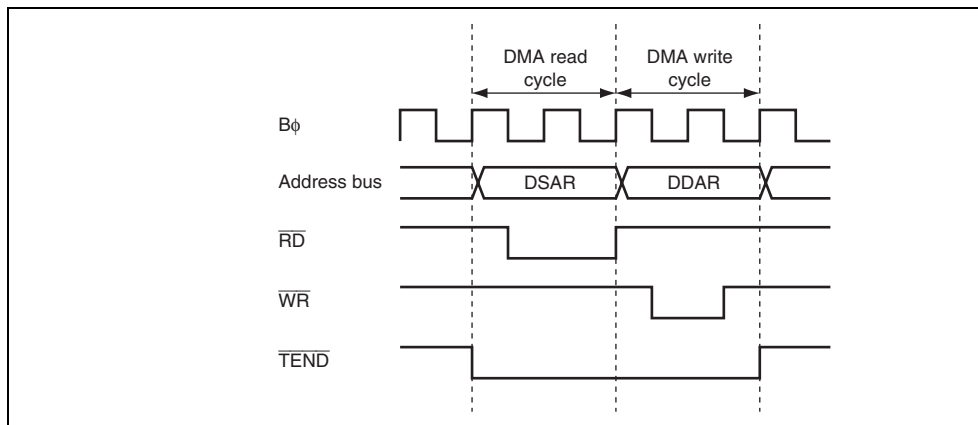
divided into multiple bus cycles).

In the first bus cycle, data at the transfer source address is read and in the next cycle, the data is written to the transfer destination address.

The read and write cycles are not separated. Other bus cycles (bus cycle by other bus master, 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 TENDE 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 10.2 shows an example of the signal timing in dual address mode and figure 10.3 shows an example of the signal timing in dual address mode.



**Figure 10.2 Example of Signal Timing in Dual Address Mode**



**(2) Single Address Mode**

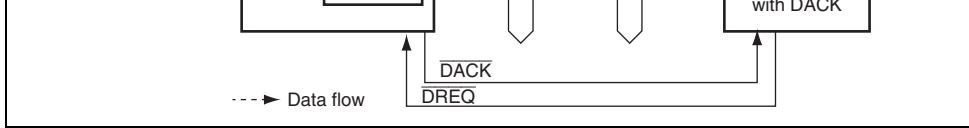
In single address mode, data between an external device and an external memory is directly transferred using the  $\overline{\text{DACK}}$  pin instead of DSAR or DDAR. A transfer at a time is performed in one bus cycle. In this mode, the data bus width must be the same as the data access size. For details on the data bus width, see section 9, Bus Controller (BSC).

The DMAC accesses an external device as the transfer source or destination by outputting a strobe signal ( $\overline{\text{DACK}}$ ) to the external device with  $\overline{\text{DACK}}$  and accesses the other transfer source or destination by outputting the address. Accordingly, the DMA transfer is performed in one bus cycle. Figure 10.3 shows an example of a transfer between an external memory and an external device with  $\overline{\text{DACK}}$  pin. In this example, the external device outputs data on the data bus and the data is transferred 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 or external memory as the transfer source or destination. When  $\text{DIRS} = 0$ , data is transferred from an external memory (DSAR) to an external device with the  $\overline{\text{DACK}}$  pin. When  $\text{DIRS} = 1$ , data is transferred from an external device with the  $\overline{\text{DACK}}$  pin to an external memory (DDAR). The settings of registers which are not used as the transfer source or destination are ignored.

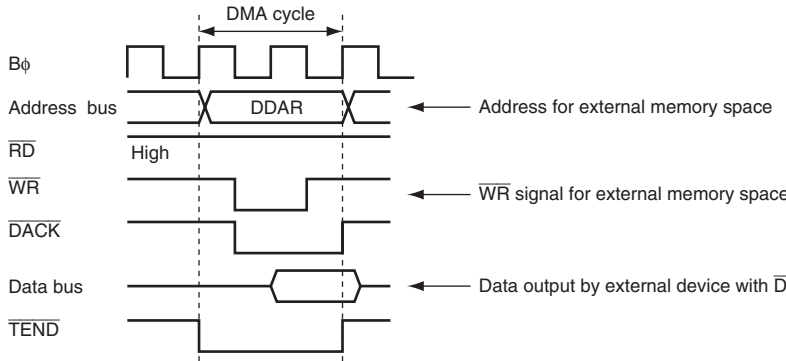
The  $\overline{\text{DACK}}$  signal output is enabled in single address mode by the DACKEN bit in DMDR. When the  $\overline{\text{DACK}}$  signal is low active.

The  $\overline{\text{TEND}}$  signal output is enabled or disabled by the TENDE bit in DMDR. The  $\overline{\text{TEND}}$  signal is output in one bus cycle. When an idle cycle is inserted before the bus cycle, the  $\overline{\text{TEND}}$  signal is also output in the idle cycle.

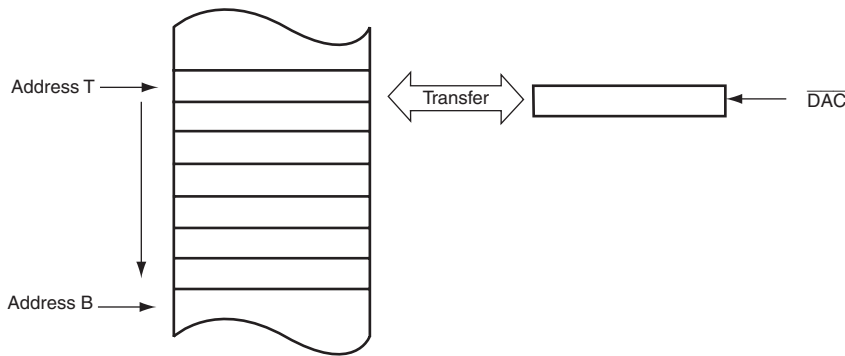


**Figure 10.4 Data Flow in Single Address Mode**

Transfer from external device with  $\overline{DACK}$  to external memory

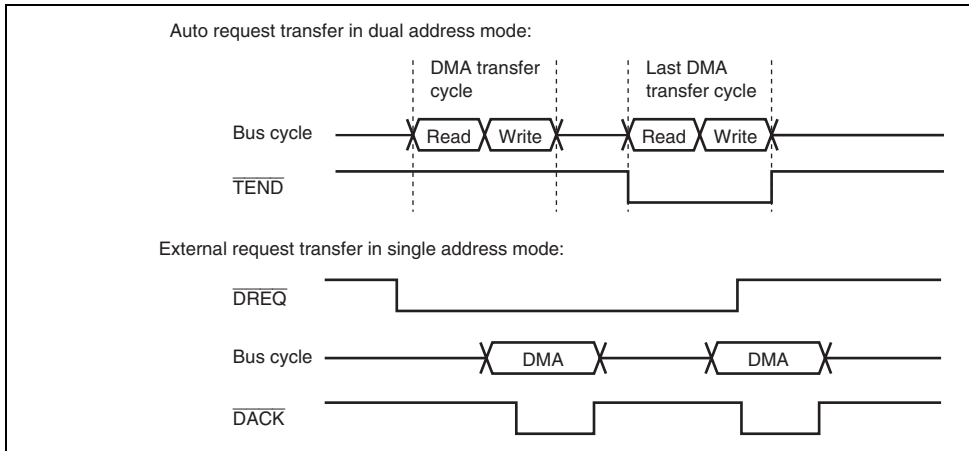


**Figure 10.5 Example of Signal Timing in Single Address Mode**

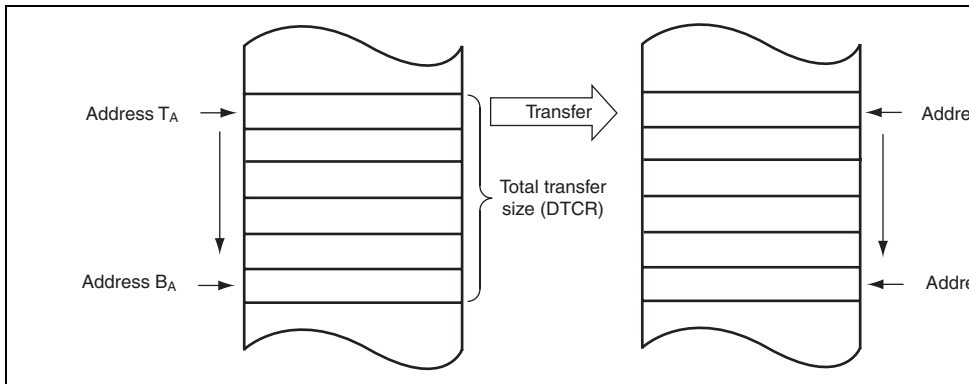


**Figure 10.6 Operations in Single Address Mode**

the operation in normal transfer mode.



**Figure 10.7 Example of Signal Timing in Normal Transfer Mode**



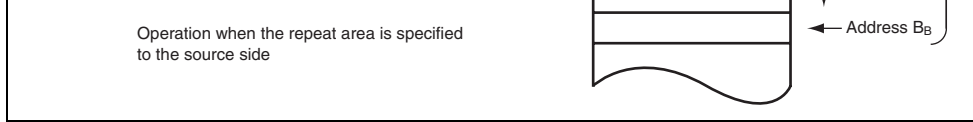
**Figure 10.8 Operations in Normal Transfer Mode**

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 timing of the  $\overline{\text{TEND}}$  signals is the same as in normal transfer mode.

Figure 10.9 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 10.8. In this case, a repeat size end interrupt can also be requested to the CPU when the repeat size of transfers is completed.



**Figure 10.9 Operations in Repeat Transfer Mode**

### (3) Block Transfer Mode

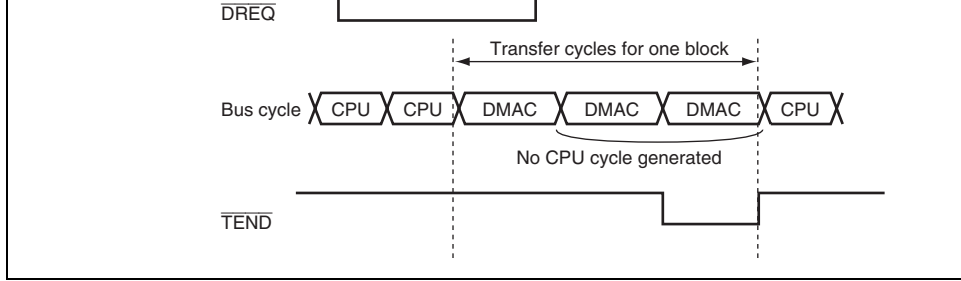
In block transfer mode, one block size of data is transferred at a single transfer request. Up to 64Kbytes can be specified as total transfer size by DTCR. The block size can be specified in units of 16bytes up to 64K × data access size.

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.

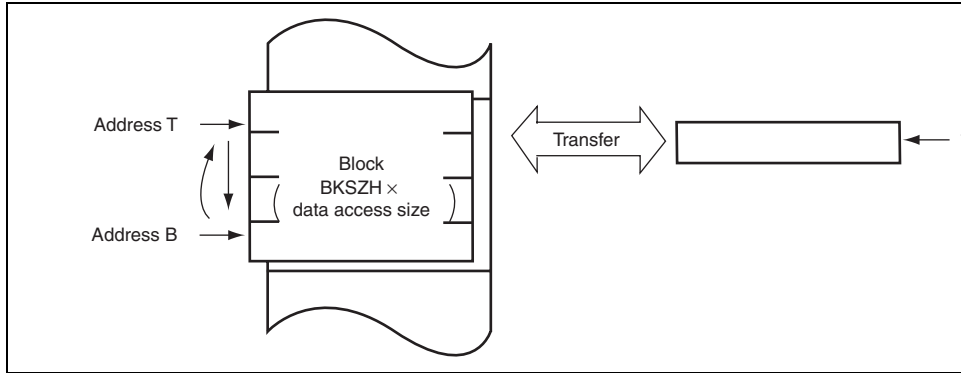
The block area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the block area returns to the transfer start address when the block size of data is completed. When the block area is specified as neither source nor destination address side, the operation continues without returning the address to the transfer start address. The repeat size end interrupt can be requested.

The  $\overline{\text{TEND}}$  signal is output every time 1-block data is transferred in the last DMA transfer.

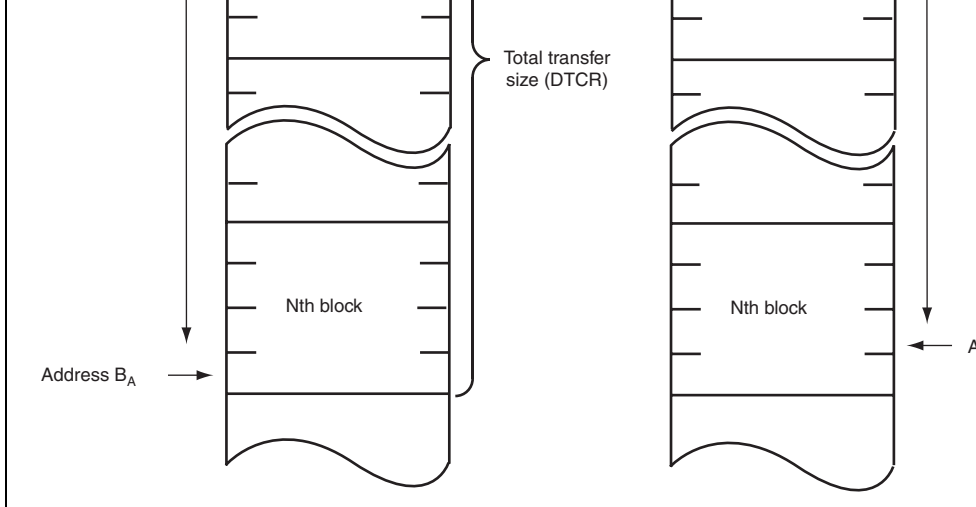
When an interrupt request by an extended repeat area overflow is used in block transfer mode, the settings should be selected carefully. For details, see section 10.5.5, Extended Repeat Area Overflow Function.



**Figure 10.10 Operations in Block Transfer Mode**



**Figure 10.11 Operation in Single Address Mode in Block Transfer Mode (Block Area Specified)**



**Figure 10.12 Operation in Dual Address Mode in Block Transfer Mode  
(Block Area Not Specified)**



DMDR starts a transfer. The bus mode can be selected from cycle stealing and burst mo

## (2) Activation by On-Chip Module Interrupt

An interrupt request from an on-chip peripheral module (on-chip peripheral module inte used as a transfer request. When a DMA transfer is enabled ( $DTE = 1$ ), the DMA transf started by an on-chip module interrupt.

The activation source of the on-chip module interrupt is selected by the DMA module re select register (DMRSR). The activation sources are specified to the individual channels 10.5 is a list of on-chip module interrupts for the DMAC. The interrupt request selected activation source can generate an interrupt request simultaneously to the CPU or DTC. I refer to section 7, Interrupt Controller.

The DMAC receives interrupt requests by on-chip peripheral modules independent of th controller. Therefore, the DMAC is not affected by priority given in the interrupt contro

When the DMAC is activated while  $DTA = 1$ , the interrupt request flag is automatically a DMA transfer. If multiple channels use a single transfer request as an activation source the channel having priority is activated, the interrupt request flag is cleared. In this case, 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 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 req 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 sour cleared to 0 before writing 1 to the DTE bit.

TGI5A (TGI5A input capture/compare match)	TPU_5	11
RXI0 (receive data full interrupt for SCI channel 0)	SCI_0	14
TXI0 (transmit data empty interrupt for SCI channel 0)	SCI_0	14
RXI1 (receive data full interrupt for SCI channel 1)	SCI_1	14
TXI1 (transmit data empty interrupt for SCI channel 1)	SCI_1	15
RXI2 (receive data full interrupt for SCI channel 2)	SCI_2	15
TXI2 (transmit data empty interrupt for SCI channel 2)	SCI_2	15
RXI4 (receive data full interrupt for SCI channel 4)	SCI_4	16
TXI4 (transmit data empty interrupt for SCI channel 4)	SCI_4	16
TGI6A (TGI6A input capture/compare match)	TPU_6	16
TGI7A (TGI7A input capture/compare match)	TPU_7	16
TGI8A (TGI8A input capture/compare match)	TPU_8	17
TGI9A (TGI9A input capture/compare match)	TPU_9	17
TGI10A (TGI10A input capture/compare match)	TPU_10	18
TGI11A (TGI11A input capture/compare match)	TPU_11	18
RXI5 (receive data full interrupt for SCI channel 5)	SCI_5	22
TXI5 (transmit data empty interrupt for SCI channel 5)	SCI_5	22
RXI6 (receive data full interrupt for SCI channel 6)	SCI_6	22
TXI6 (transmit data empty interrupt for SCI channel 6)	SCI_6	22
USBINTN0 (EP1FIFO full interrupt)	USB	23
USBINTN1 (EP2FIFO empty interrupt)	USB	23
ADI1 (conversion end interrupt for A/D converter unit 1)	A/D_1	23

When an external request is selected as an activation source, clear the DTR bit to 0 and ICR bit to 1 for the corresponding pin. For details, see section 13, I/O Ports.

#### 10.5.4 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 DTF0 in DMDR. When an activation source is the on-chip module interrupt or external interrupt, the cycle stealing mode is selected.

##### (1) Cycle Stealing Mode

In cycle stealing mode, the DMAC releases the bus every time one unit of transfers (byte, halfword, longword, or 1-block size) is completed. After that, when a transfer is requested, the DMAC obtains the bus to transfer 1-unit data and then releases the bus on completion of the transfer. The transfer operation is continued until the transfer end condition is satisfied.

When a transfer is requested to another channel during a DMA transfer, the 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 10.5.8, Priority of Channels.

### Figure 10.13 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 with higher priority, the transfer is not stopped once it is started. The DMAC releases the bus in the next bus cycle after the transfer for the channel in burst mode is completed. This is similar 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 cycle stealing 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 area overflow, or an extended repeat area overflow occurs, the DTE bit is cleared to 0 and the transfer is stopped.

Figure 10.14 shows an example of timing in burst mode.

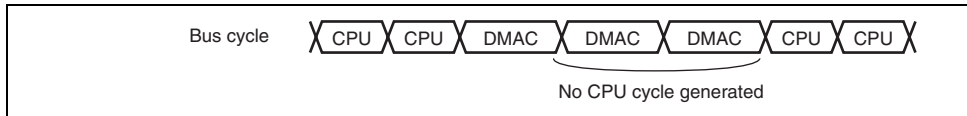
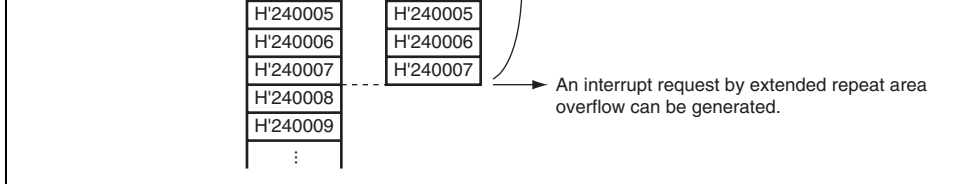


Figure 10.14 Example of Timing in Burst Mode

The extended repeat area on the source address is specified by bits SARA4 to SARA0 in DSAR. The extended repeat area on the destination address is specified by bits DARA4 to DARA0 in DDAR and DACR. The extended repeat area sizes for each side can be specified independently.

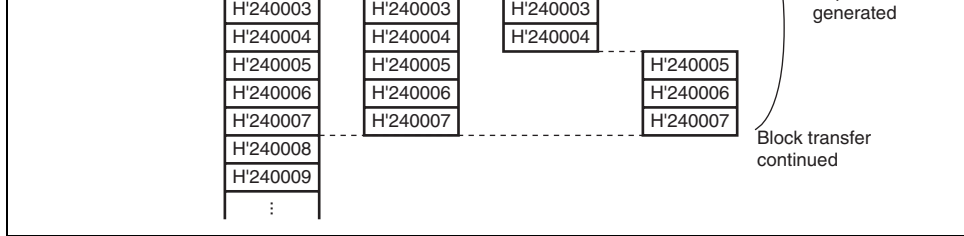
A DMA transfer is stopped and an interrupt by an extended repeat area overflow can be requested to the CPU when the contents of the address register reach the end address of the extended repeat area. When an overflow on the extended repeat area set in DSAR occurs while the SARA bit in DACR is set to 1, the ESIF bit in DMDR is set to 1 and the DTE bit in DMDR is cleared to 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 address has reached the target. During the interrupt handling, setting the DTE bit in DMDR resumes the transfer.



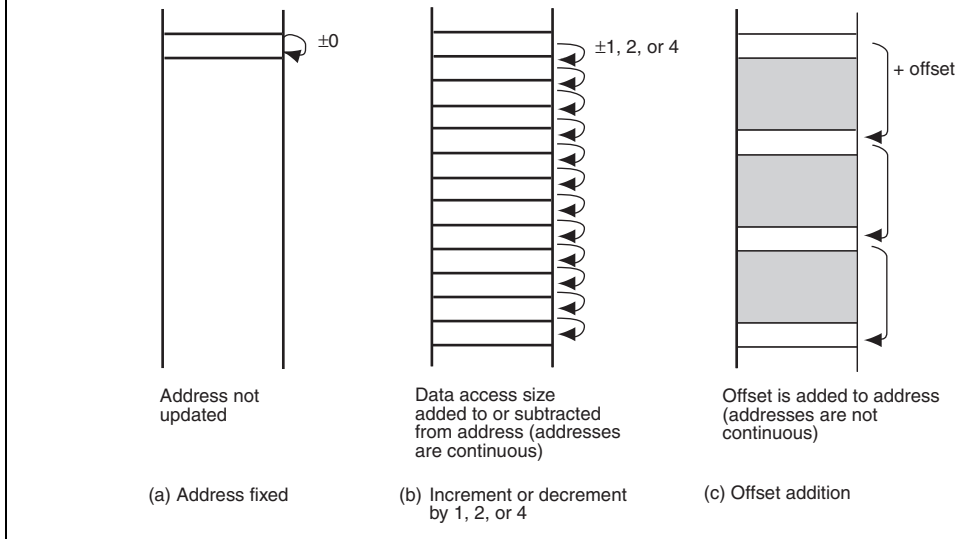
**Figure 10.15 Example of Extended Repeat Area Operation**

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 the transfer of one block, the interrupt by the overflow is suspended and the transfer overruns.



**Figure 10.16 Example of Extended Repeat Area Function in Block Transfer**



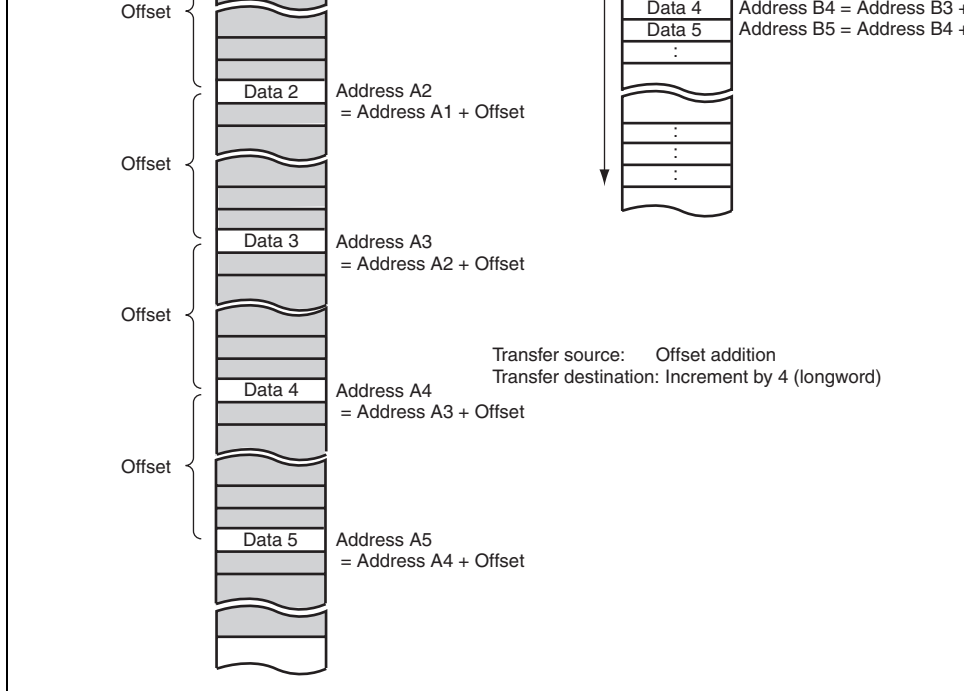
**Figure 10.17 Address Update Method**

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. A byte, word, or longword can be specified as the data access size. The value of 1 for byte, 2 for word, or 4 for longword is used for updating the address. This operation realizes the data transfer in consecutive areas.

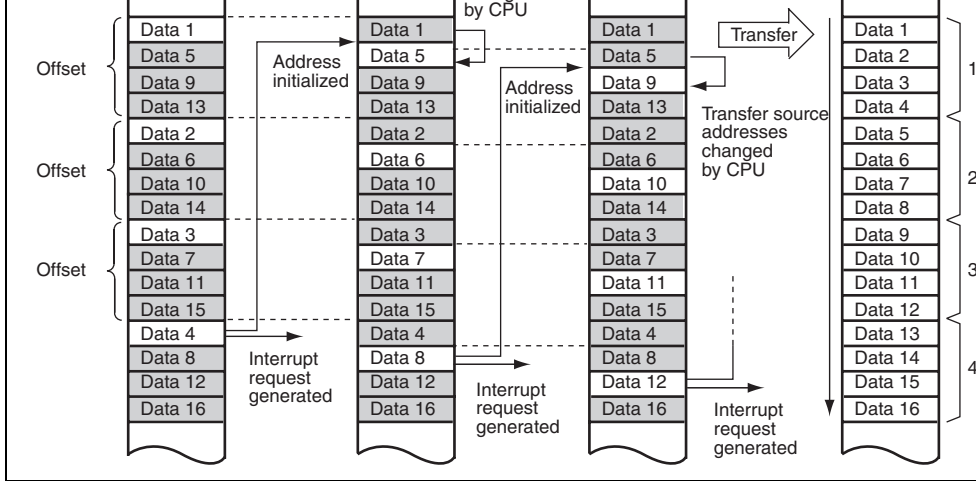
In item (c), Offset addition, the address update does not depend on the data access size. The value specified by DOFR is added to the address every time the DMAC transfers data of the data access size.





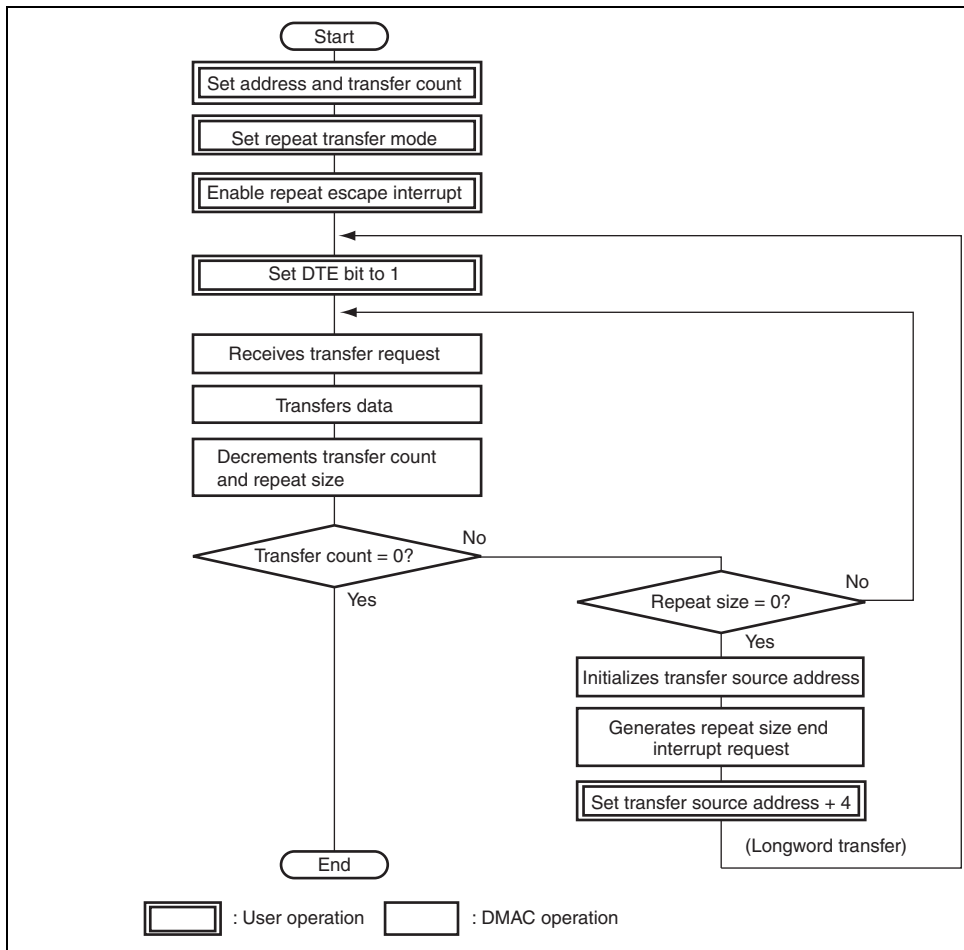
**Figure 10.18 Operation of Offset Addition**

In figure 10.18, 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 is performed on the source side. The 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 destination side in a consecutive area in the destination side.



**Figure 10.19 XY Conversion Operation Using Offset Addition in Repeat Transfer**

In figure 10.19, 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. A repeat size end interrupt is requested when the RPTIE bit in DACR is set to 1 and the size of transfers is completed.



**Figure 10.20 XY Conversion Flowchart Using Offset Addition in Repeat 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, DTCH, 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 byte, 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 source address when the read cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

## (2) DMA Destination Address Register

When the transfer destination address set in DDAR is accessed, the contents of DDAR 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 incremented by 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 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 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, the block or repeat area is specified to the destination address side, the destination address is updated 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, 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 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 DMA 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 BKSZ. Bits 15 to 0 in DBSR function as BKSZH. The BKSZH bits (16 bits) store the block 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 changed from 1 to 0 by a DMA transfer, 0 is not stored but the BKSZH value is loaded into the BKSZ bits.

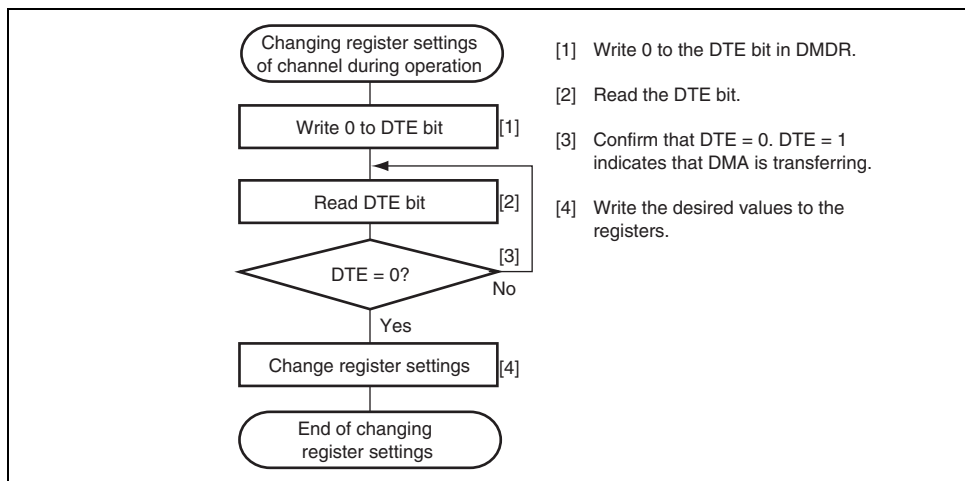
Since the upper 16 bits of DBSR are not updated, DBSR can be accessed in words.

DBSR for the channel being transferred must not be written to.

- 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 10.21 shows the procedure for changing the register settings for the channel being transferred.



**Figure 10.21 Procedure for Changing Register Setting For Channel being Transferred**

In burst mode, up to three times of DMA transfer are performed from the cycle in which the 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 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.

However, when the DMAC is in the module stop state, the ERRF bit is not set to 1 for address errors or the NMI.

### **(8) ESIF Bit in DMDR**

When an interrupt by a transfer size error, a repeat size end, or an extended repeat area 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 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 10.8, Interrupt Sources.



### 10.5.8 Priority of Channels

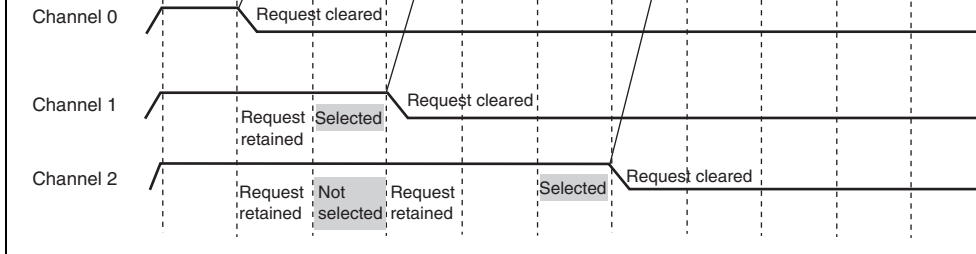
The channels of the DMAC are given following priority levels: channel 0 > channel 1 > channel 2 > channel3. Table 10.6 shows the priority levels among the DMAC channels.

**Table 10.6 Priority among DMAC Channels**

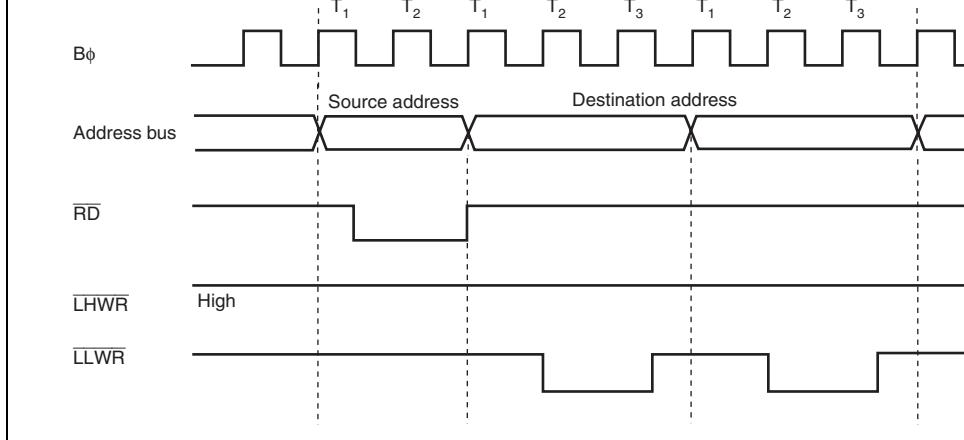
<b>Channel</b>	<b>Pr</b>
Channel 0	Hi
Channel 1	
Channel 2	
Channel 3	Lo

The channel having highest priority other than the channel being transferred is selected if 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.

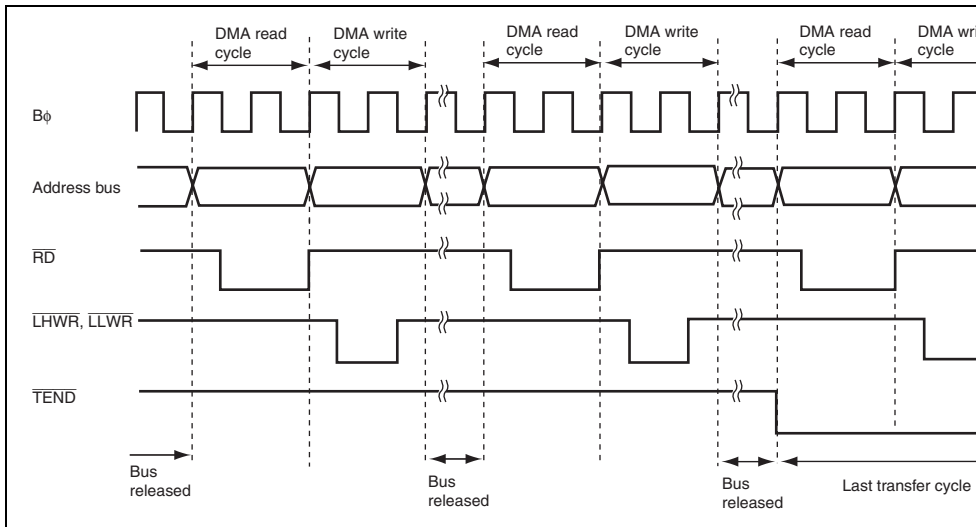
In a burst transfer or a block transfer, channels are not switched.



**Figure 10.22 Example of Timing for Channel Priority**



**Figure 10.23 Example of Bus Timing of DMA Transfer**



**Figure 10.24 Example of Transfer in Normal Transfer Mode by Cycle Stealing**

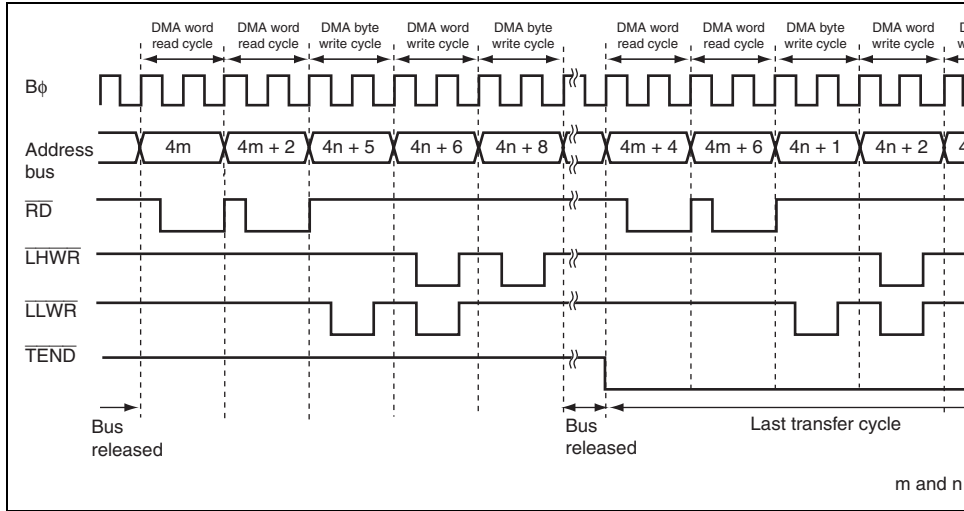
In figures 10.25 and 10.26, the  $\overline{TEND}$  signal output is enabled and data is transferred in longword mode from the external 16-bit 2-state access space to the 16-bit 2-state access space in normal transfer mode by cycle stealing.

In figure 10.25, the transfer source (DSAR) is not aligned with a longword boundary and the transfer destination (DDAR) is aligned with a longword boundary.

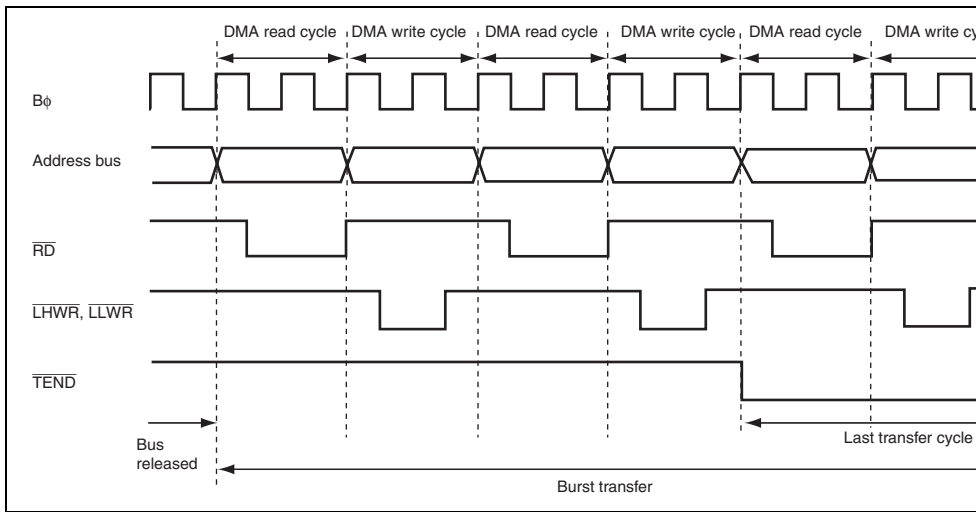
In figure 10.26, the transfer source (DSAR) is aligned with a longword boundary and the destination (DDAR) is not aligned with a longword boundary.



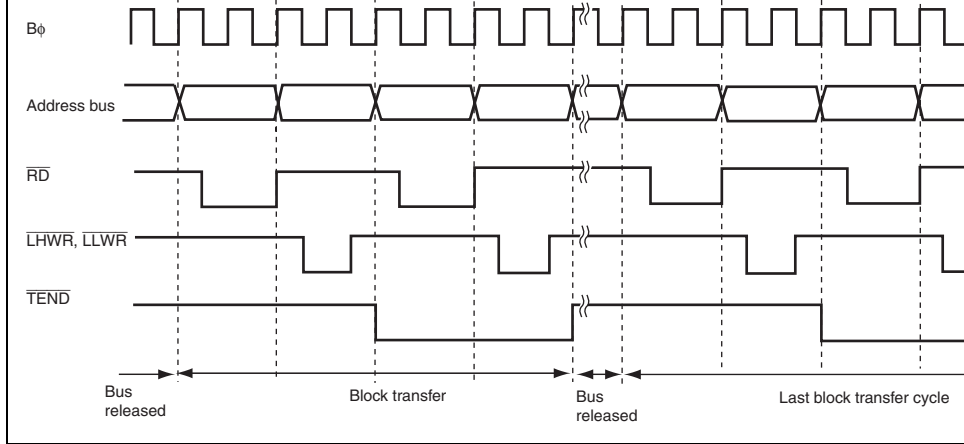
**Figure 10.25 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Source DSAR = Odd Address and Source Address Increment)**



**Figure 10.26 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Destination DDAR = Odd Address and Destination Address Decrement)**

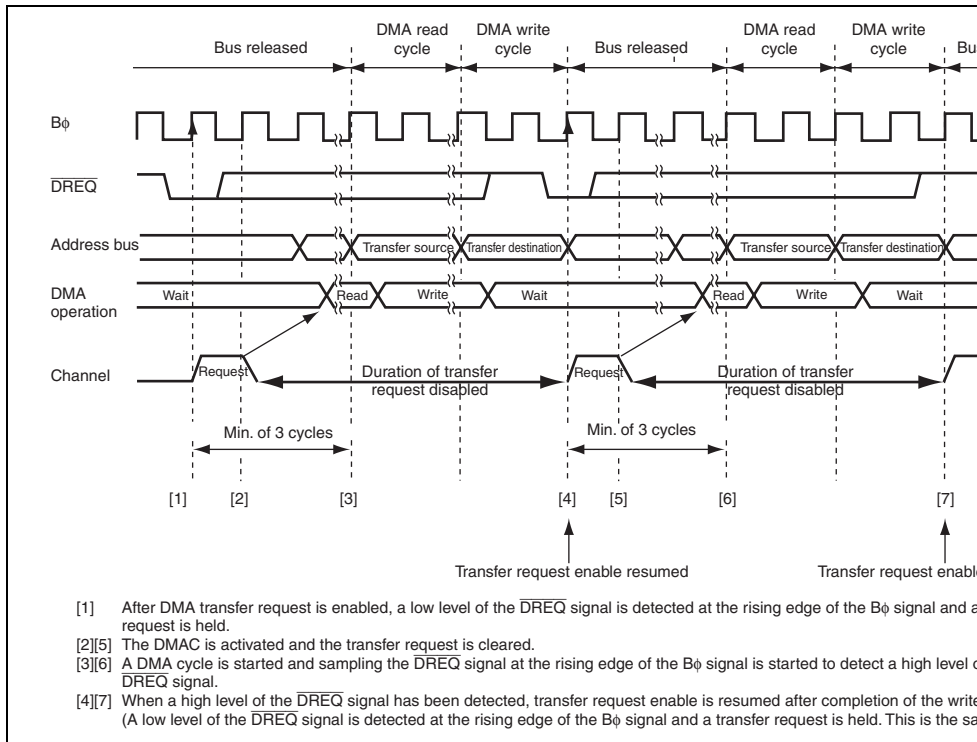


**Figure 10.27 Example of Transfer in Normal Transfer Mode by Burst Access**



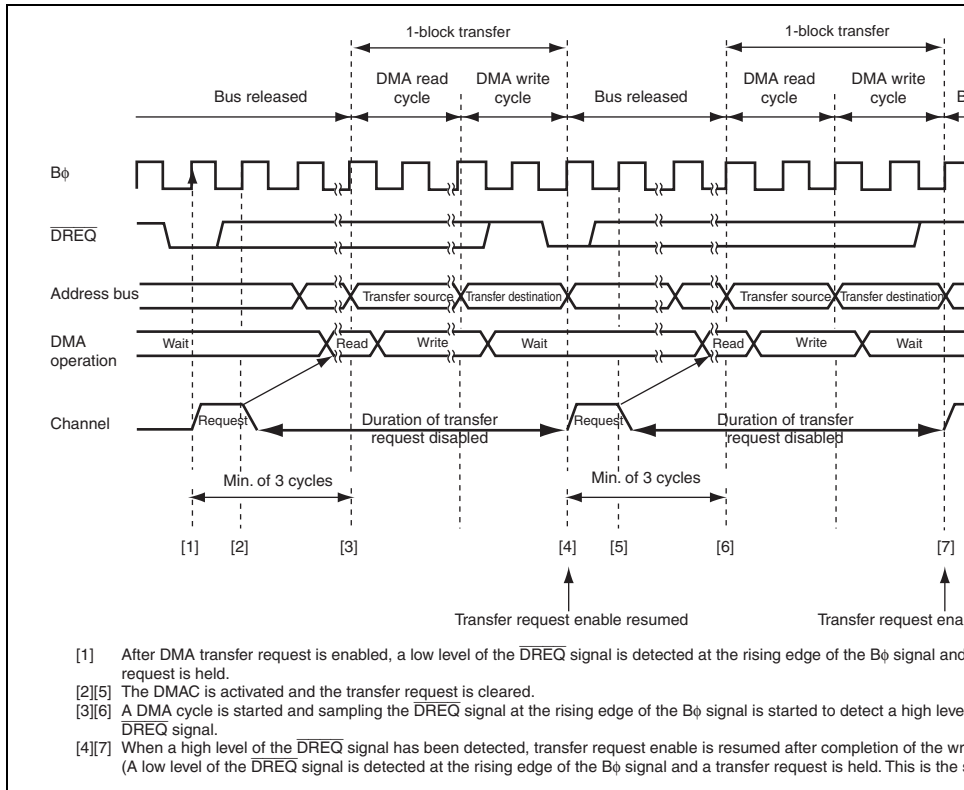
**Figure 10.28 Example of Transfer in Block Transfer Mode**

When 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.

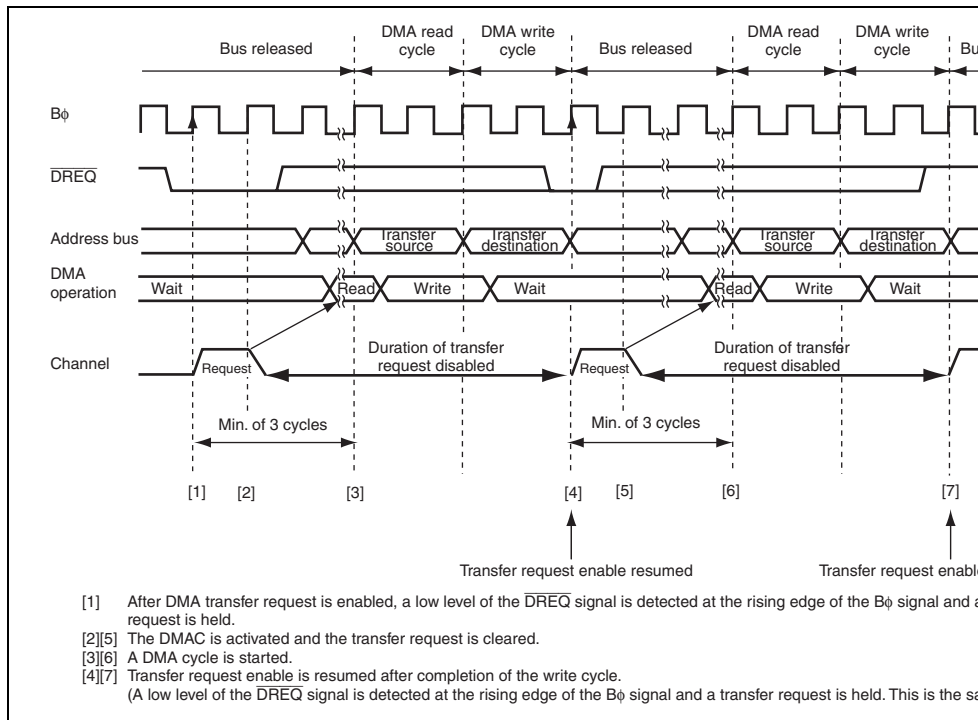


**Figure 10.29 Example of Transfer in Normal Transfer Mode Activated by  $\overline{\text{DREQ}}$  Falling Edge**

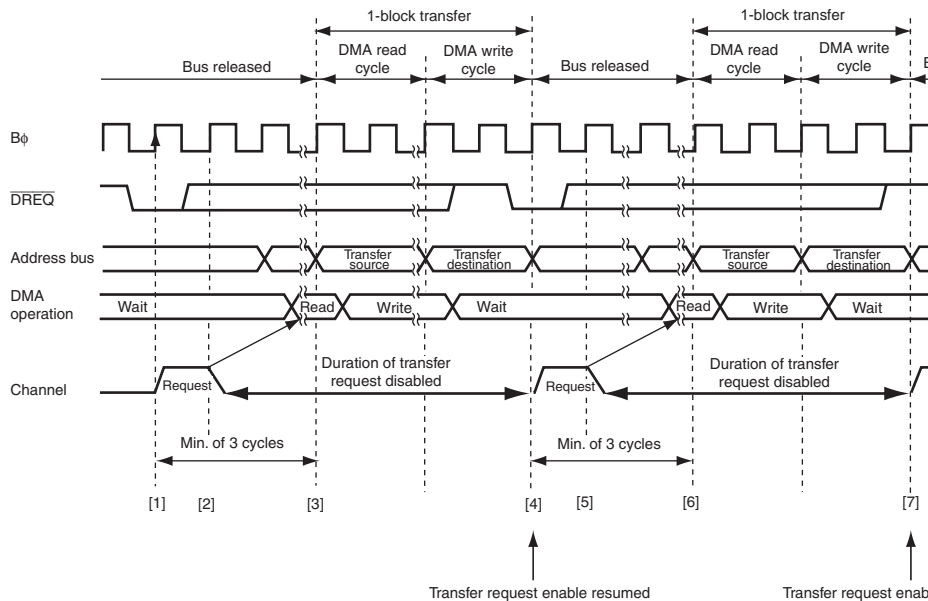




**Figure 10.30 Example of Transfer in Block Transfer Mode Activated by  $\overline{DREQ}$  Falling Edge**



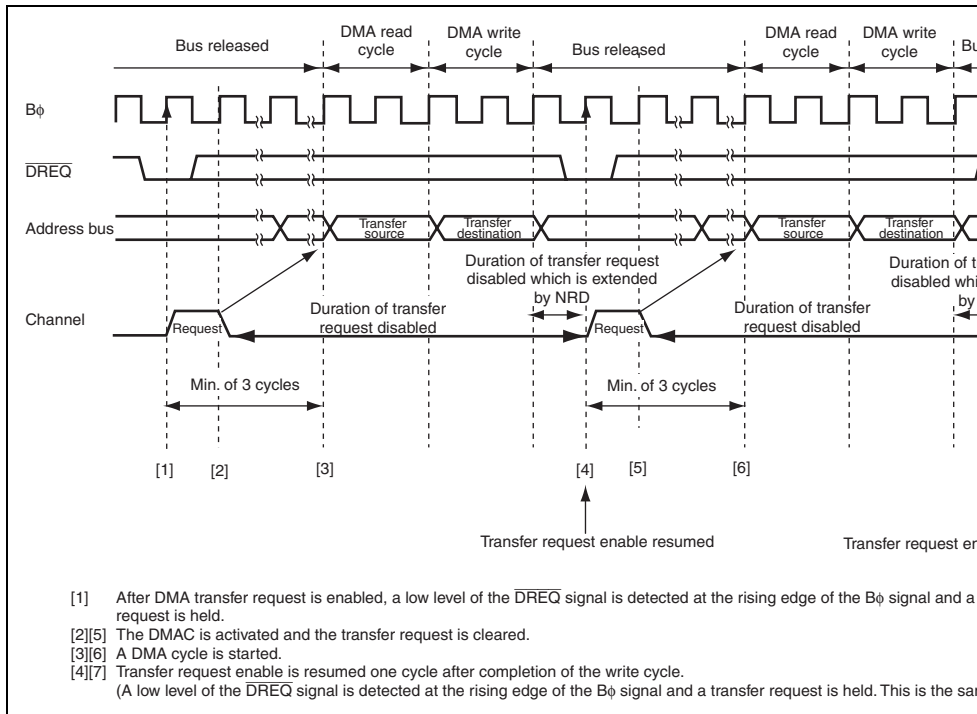
**Figure 10.31 Example of Transfer in Normal Transfer Mode Activated by  $\overline{DREQ}$  Low Level**



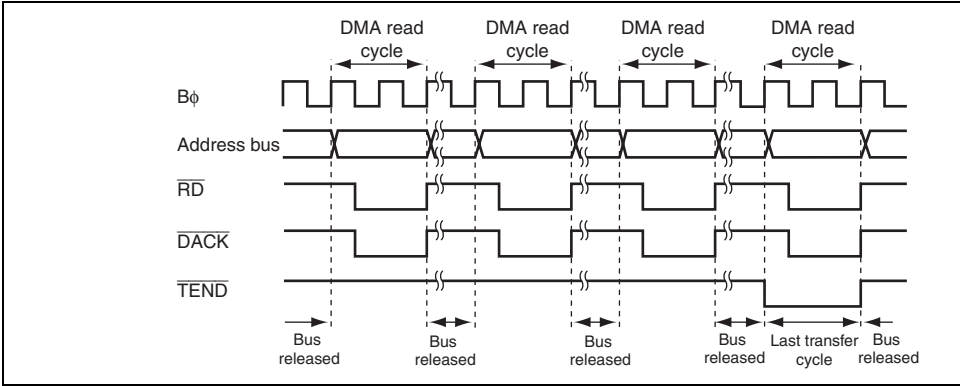
- [1] After DMA transfer request is enabled, a low level of the  $\overline{DREQ}$  signal is detected at the rising edge of the  $B\phi$  signal and a transfer request is held.
- [2][5] The DMAC is activated and the transfer request is cleared.
- [3][6] A DMA cycle is started.
- [4][7] Transfer request enable is resumed after completion of the write cycle.  
(A low level of the  $\overline{DREQ}$  signal is detected at the rising edge of the  $B\phi$  signal and a transfer request is held. This is the same as [1].)

**Figure 10.32 Example of Transfer in Block Transfer Mode Activated by  $\overline{DREQ}$  Low Level**

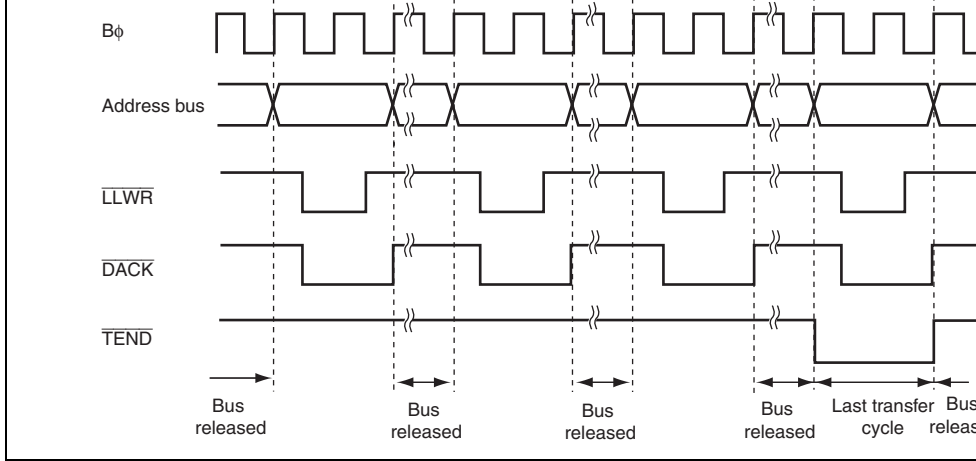
enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



**Figure 10.33 Example of Transfer in Normal Transfer Mode Activated by  $\overline{\text{DREQ}}$  Low Level with  $\text{NRD} = 1$**

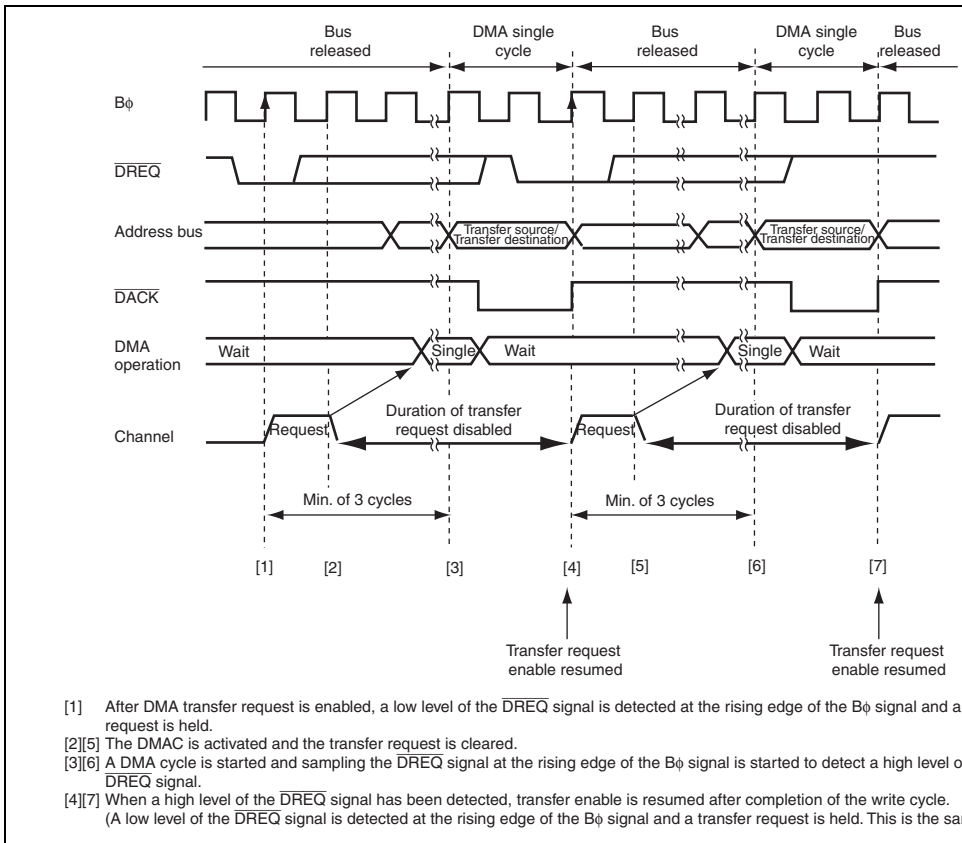


**Figure 10.34 Example of Transfer in Single Address Mode (Byte Read)**

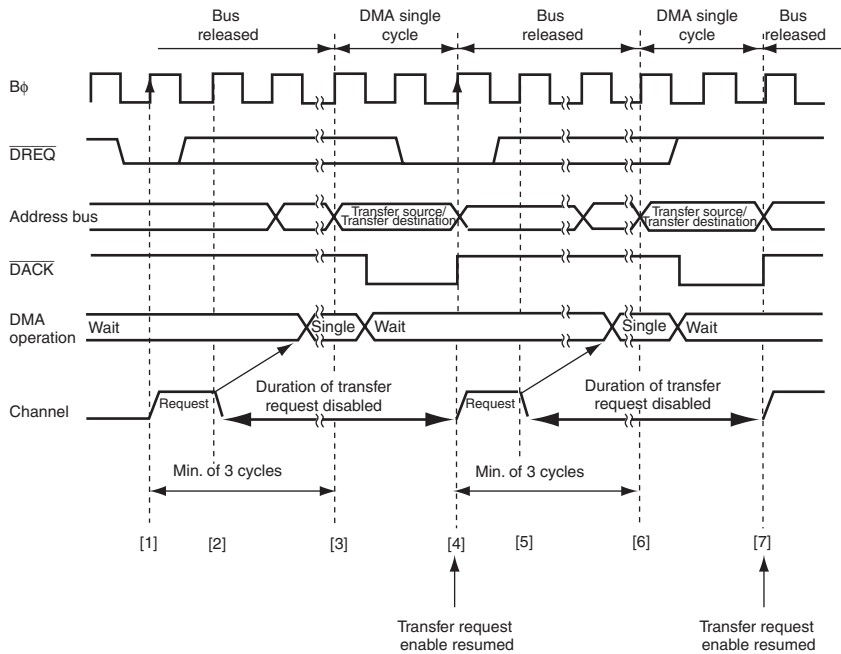


**Figure 10.35 Example of Transfer in Single Address Mode (Byte Write)**

operation is repeated until the transfer is completed.



**Figure 10.36 Example of Transfer in Single Address Mode Activated by  $\overline{DREQ}$  Falling Edge**

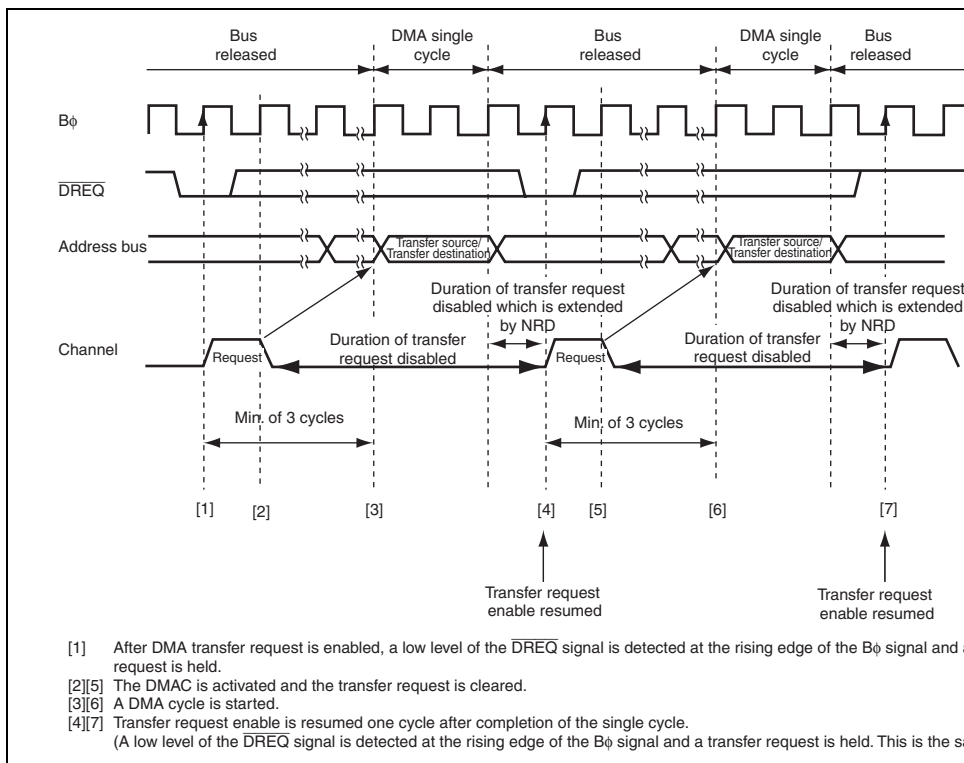


- [1] After DMA transfer request is enabled, a low level of the  $\overline{DREQ}$  signal is detected at the rising edge of the  $B\phi$  signal and a transfer request is held.
- [2][5] The DMAC is activated and the transfer request is cleared.
- [3][6] A DMA cycle is started.
- [4][7] Transfer request enable is resumed after completion of the single cycle.  
(A low level of the  $\overline{DREQ}$  signal is detected at the rising edge of the  $B\phi$  signal and a transfer request is held. This is the same as [1].)

**Figure 10.37 Example of Transfer in Single Address Mode Activated by  $\overline{DREQ}$  Low Level**



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  $\overline{\text{NRD}} = 1$  on completion of the single cycle and then a low level  $\overline{\text{DREQ}}$  signal is detected. This operation is repeated until the transfer is completed.



**Figure 10.38 Example of Transfer in Single Address Mode Activated by  $\overline{\text{DREQ}}$  Low Level with  $\overline{\text{NRD}} = 1$**

## (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 DMDR is set 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 data in DTCR is transferred instead of transferring the block size of data. The transfer is performed in bytes.

When an overflow on the extended repeat area occurs while the extended repeat area is selected 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 SARIE or DARIE bit in DACR 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 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.

transfer unit.

In single address mode, a DMA transfer is completed after completion of the bus cycle for the 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 the same as 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 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.

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 channel is given priority over the CPU by changing priority levels of the CPU or channel, a 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.

read and write cycles of a DMA transfer.

In block transfer mode and an auto request transfer by burst access, bus cycles of the DMAC transfer are consecutively performed. For this duration, since the DMAC has priority over CPU and DTC, accesses to the external space is suspended (the IBCCS bit in the bus controller 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 9, Bus Controller (BSC).

A conflict may occur between external space access of the refresh cycle and EXDMAC cycle and 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 the refresh or EXDMAC or external bus release cycle is inserted by the BSC according to the external bus priority (when the CPU external access and DTC external access do not have priority over a DMAC transfer, the transfers are not stopped 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 EXDMAC cycle, or EXDMAC cycle, or an external bus release cycle may be performed at the same time.

DMTEND2	Transfer end interrupt by channel 2 transfer counter
DMTEND3	Transfer end interrupt by channel 3 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
DMEEND2	Interrupt by channel 2 transfer size error Interrupt by channel 2 repeat size end Interrupt by channel 2 extended repeat area overflow on source address Interrupt by channel 2 extended repeat area overflow on destination address
DMEEND3	Interrupt by channel 3 transfer size error Interrupt by channel 3 repeat size end Interrupt by channel 3 extended repeat area overflow on source address Interrupt by channel 3 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 is decided by the interrupt controller and it is shown in table 10.7. For details, see section 10.7.3 Interrupt Controller.

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 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 transfer 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 completing 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 the time when a transfer end interrupt by the transfer counter is generated, the ESIF bit is set to 1.

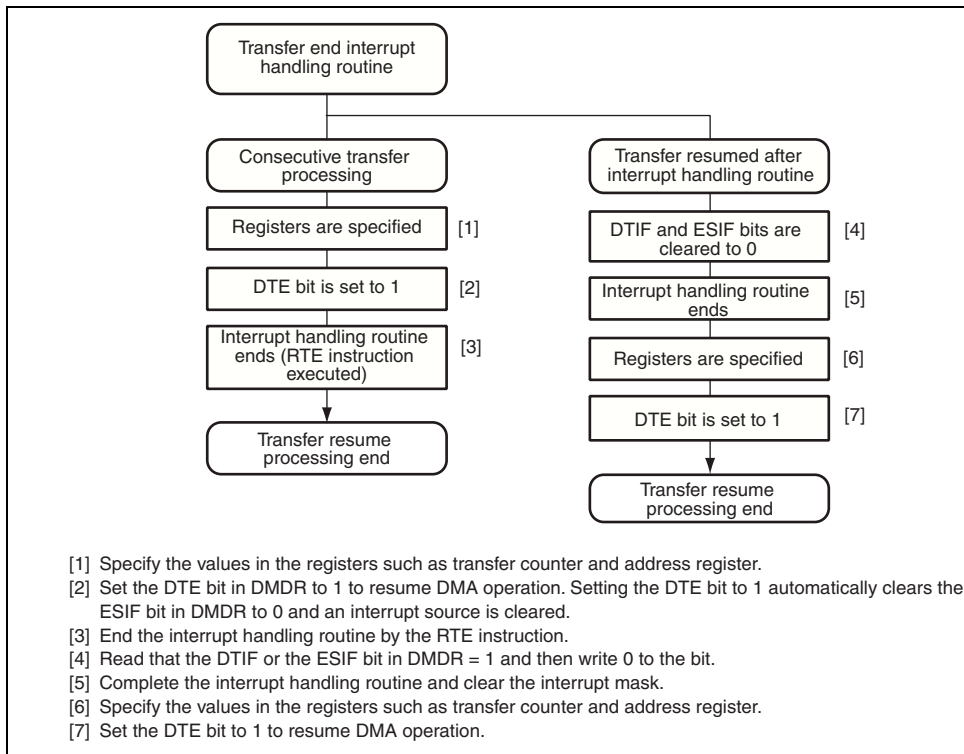
An interrupt by an extended repeat area overflow on the source and destination addresses is generated when the address exceeds the extended repeat area (overflow). At this time, when a transfer end interrupt by the transfer counter, the ESIF bit is set to 1.

Figure 10.39 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 10.40 shows procedure to restart the transfer by clearing an interrupt.



Extended repeat area overflow occurs in destination address

**Figure 10.39 Interrupt and Interrupt Sources**



**Figure 10.40 Procedure Example of Resuming Transfer by Clearing Interrupt**

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, 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 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)

### 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.

### 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 high 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.

- Up to 4-Gbyte address space accessible
- Selection of byte, word, or longword transfer data length
- Total transfer size of up to 4 Gbytes (4,294,967,295 bytes)  
Selection of free-running mode (with no total transfer size specified)
- Selection of auto-requests or external requests for activating the EXDMAC  
Auto-request: Activation from the CPU (Cycle steal mode or burst mode can be selected)  
External request: Low level sensing or falling edge sensing for the EDREQ signal can be selected.  
All of four channels can accept external requests.
- Selection of dual address mode or single address mode  
Dual address mode: Both the transfer source and destination addresses are specified to transfer data.  
Single address mode: The EDACK signal is used to access the transfer source or destination peripheral device and the address of the other device is specified to transfer data.
- Normal, repeat, block, or cluster transfer (only for the EXDMAC) 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 is specified to start the transfer  
the transfer start address  
Up to 64-Kbyte transfers can be set as repeat size (65,536 bytes/words/longwords)  
Block transfer mode: One block data is transferred at a single transfer request  
Up to 64-Kbyte data can be set as block size (65,536 bytes/words/longwords)

transfer.

- Transfer of word or longword data to addresses beyond each data boundary

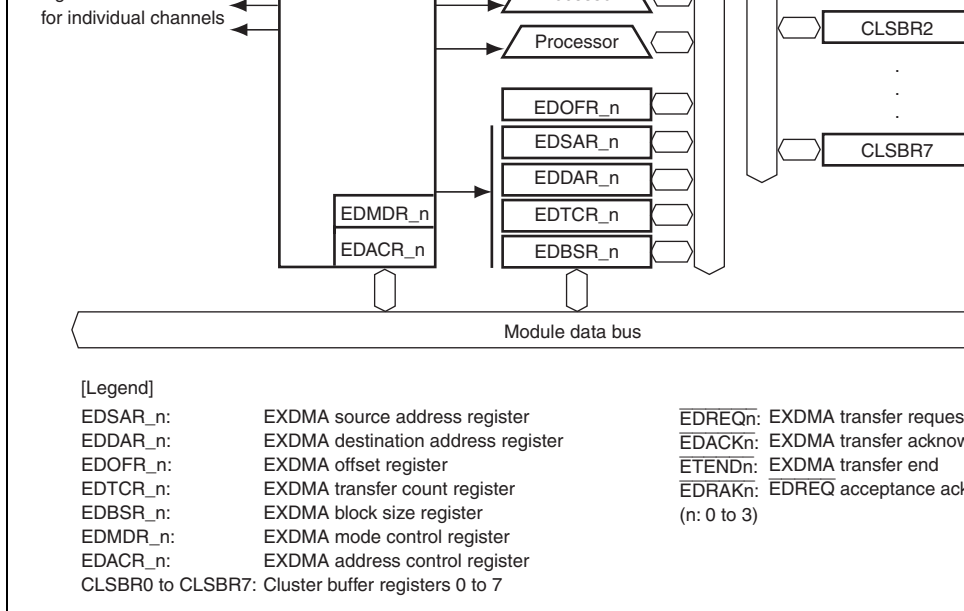
Data can be divided into an optimal data size (byte or word) according to addresses within transferring data.

- Two kinds of interrupts requested to the CPU

Transfer end interrupt: Requested after the number of data set by the transfer counter is completely transferred

Transfer escape end interrupt: Requested when the remaining transfer size is smaller than the size set for a single transfer request, after a repeat-size transfer is completed, or when an extended repeat area overflow occurs.

- Acceptance of a transfer request can be reported to an external device via the EDRACK (only for the EXDMAC).
- Operation of EXDMAC, connected to a dedicated bus, in parallel with a bus master such as CPU, DTC, or DMAC (only for the EXDMAC).
- Module stop state can be set.



**Figure 11.1 Block Diagram of EXDMAC**

	$\overline{\text{EDREQ0}}$ acceptance acknowledge	$\overline{\text{EDRAK0}}$	Output	Notification to external de channel 0 external requ acceptance and start of e
1	EXDMA transfer request 1	$\overline{\text{EDREQ1}}$	Input	Channel 1 external requ
	EXDMA transfer acknowledge 1	$\overline{\text{EDACK1}}$	Output	Channel 1 single address acknowledge
	EXDMA transfer end 1	$\overline{\text{ETEND1}}$	Output	Channel 1 transfer end
	$\overline{\text{EDREQ1}}$ acceptance acknowledge	$\overline{\text{EDRAK1}}$	Output	Notification to external de channel 1 external requ acceptance and start of e
2	EXDMA transfer request 2	$\overline{\text{EDREQ2}}$	Input	Channel 2 external requ
	EXDMA transfer acknowledge 2	$\overline{\text{EDACK2}}$	Output	Channel 2 single address acknowledge
	EXDMA transfer end 2	$\overline{\text{ETEND2}}$	Output	Channel 2 transfer end
	$\overline{\text{EDREQ2}}$ acceptance acknowledge	$\overline{\text{EDRAK2}}$	Output	Notification to external de channel 2 external requ acceptance and start of e
3	EXDMA transfer request 3	$\overline{\text{EDREQ3}}$	Input	Channel 3 external requ
	EXDMA transfer acknowledge 3	$\overline{\text{EDACK3}}$	Output	Channel 3 single address acknowledge
	EXDMA transfer end 3	$\overline{\text{ETEND3}}$	Output	Channel 3 transfer end
	$\overline{\text{EDREQ3}}$ acceptance acknowledge	$\overline{\text{EDRAK3}}$	Output	Notification to external de channel 3 external requ acceptance and start of e

- EXDMA block size register\_0 (EDBSR\_0)
- EXDMA mode control register\_0 (EDMDR\_0)
- EXDMA address control register\_0 (EDACR\_0)

#### Channel 1

- EXDMA source address register\_1 (EDSAR\_1)
- EXDMA destination address register\_1 (EDDAR\_1)
- EXDMA offset register\_1 (EDOFR\_1)
- EXDMA transfer count register\_1 (EDTCR\_1)
- EXDMA block size register\_1 (EDBSR\_1)
- EXDMA mode control register\_1 (EDMDR\_1)
- EXDMA address control register\_1 (EDACR\_1)

#### Channel 2

- EXDMA source address register\_2 (EDSAR\_2)
- EXDMA destination address register\_2 (EDDAR\_2)
- EXDMA offset register\_2 (EDOFR\_2)
- EXDMA transfer count register\_2 (EDTCR\_2)
- EXDMA block size register\_2 (EDBSR\_2)
- EXDMA mode control register\_2 (EDMDR\_2)
- EXDMA address control register\_2 (EDACR\_2)

- Cluster buffer registers 0 to 7 (CLSBR0 to CLSBR7)



Bit	31	30	29	28	27	26	25
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	23	22	21	20	19	18	17
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	15	14	13	12	11	10	9
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	31	30	29	28	27	26	25	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Bit	23	22	21	20	19	18	17	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Bit	15	14	13	12	11	10	9	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Bit	7	6	5	4	3	2	1	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R

Bit	23	22	21	20	19	18	17
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	15	14	13	12	11	10	9
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

EDTCR can be read at all times by the CPU. When reading EDTCR for a channel on which EXDMA transfer processing is in progress, a longword-size read must be executed. Do not write to EDTCR for a channel on which EXDMA transfer is in progress.

Bit	31	30	29	28	27	26	25	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	23	22	21	20	19	18	17	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	15	14	13	12	11	10	9	
Bit Name								
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	7	6	5	4	3	2	1	
Bit Name								
Initial Value	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
Bit	23	22	21	20	19	18	17
Bit Name	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	15	14	13	12	11	10	9
Bit Name	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1
Initial Value	0	0	0	0	0	0	0
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	All 0	R/W	Sets the repeat size, block size, or cluster size. When these bits are set to H'0001, one byte-, word-, or one longword-size is set. When these bits are set to H'0000, the maximum values are set (see Table 11.2). These bits are always fixed during an EXDMA operation.
15 to 0	BKSZ15 to BKSZ0	All 0	R/W	In an EXDMA operation, the remaining repeat size, block size, or cluster size is indicated. The value is decremented by one each time of a data transfer. When the remaining size becomes zero, the BKSZH16-17 are reloaded. Set the same initial value as for the BKSZH16-17 when writing.

### 11.3.6 EXDMA Mode Control Register (EDMDR)

EDMDR controls EXDMAC operations.

- EDMDR\_0

Bit	31	30	29	28	27	26	25	
Bit Name	DTE	EDACKE	ETENDE	EDRAKE	EDREQS	NRD	—	
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	
Bit	23	22	21	20	19	18	17	
Bit Name	ACT	—	—	—	ERRF	—	ESIF	D
Initial Value	0	0	0	0	0	0	0	
R/W	R	R	R	R	R/(W)*	R	R/(W)*	R
Bit	15	14	13	12	11	10	9	
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	D
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R
Bit	7	6	5	4	3	2	1	
Bit Name	DTF1	DTF0	—	—	—	DMAP2	DMAP1	DM
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R

Note: \* Only 0 can be written to this bit after having been read as 1, to clear the flag.

Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Bit	7	6	5	4	3	2	1
Bit Name	DTF1	DTF0	—	—	—	EDMAP2	EDMAP1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W

Note: \* Only 0 can be written to this bit after having been read as 1, to clear the flag.

operation, transfer is halted.  
If this bit is cleared to 0 during an EXDMA operation in block transfer mode, this bit is cleared to 0 on completion of the currently executing one-block transfer. When this bit is cleared to 0 during an operation in cluster transfer mode, this bit is cleared to 0 on completion of the currently executing one-cluster transfer.

If an external source that ends (aborts) transfer occurs, this bit is automatically cleared to 0 and transfer is terminated.

Do not change the operating mode, transfer mode, or other parameters while this bit is set to 1.

0: Data transfer disabled

1: Data transfer enabled (during an EXDMA operation)

[Clearing conditions]

- When transfer of the total transfer size specified by the transfer size register ends
- When operation is halted by a repeat size error interrupt
- When operation is halted by an extended address area overflow interrupt
- When operation is halted by a transfer size error interrupt
- When 0 is written to terminate transfer

In block transfer mode, the value written is effective after one-block transfer ends.

In cluster transfer mode, the value written is effective after one-cluster transfer ends.

- When an address error or NMI interrupt occurs
- Reset, hardware standby mode



				0: $\overline{\text{ETEND}}$ pin output disabled 1: $\overline{\text{ETEND}}$ pin output enabled
28	EDRAKE	0	R/W	$\overline{\text{EDRAK}}$ Pin Output Enable Enables or disables output from the $\overline{\text{EDRAK}}$ pin. 0: $\overline{\text{EDRAK}}$ pin output disabled 1: $\overline{\text{EDRAK}}$ pin output enabled
27	EDREQS	0	R/W	$\overline{\text{EDREQ}}$ Select Selects whether a low level or the falling edge of the $\overline{\text{EDREQ}}$ signal used in external request mode is detected. 0: Low-level detection 1: Falling edge detection (the first transfer is completed on a low level after a transfer is enabled.)
26	NRD	0	R/W	Next Request Delay Selects the timing of the next transfer request to be accepted. 0: Next transfer request starts to be accepted when the transfer of the bus cycle in progress ends. 1: Next transfer request starts to be accepted one bus cycle of $B\phi$ from the completion of the bus cycle in progress.
25, 24	—	All 0	R	Reserved They are always read as 0 and cannot be modified.

19	ERRF	0	R(W)*	System Error Flag
				<p>Flag that indicates the occurrence of an address error or NMI interrupt. This bit is only enabled in EDMA5 channels when the DTE bit for the channels is disabled. This bit is reserved in EDMA5 channels 0 to EDMDR_3. They are always read as 0 and cannot be modified.</p> <p>0: Address error or NMI interrupt is not generated  1: Address error or NMI interrupt is generated</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Writing 0 to ERRF after reading ERRF = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When an address error or NMI interrupt occurs</li> </ul> <p>However, when an address error or an NMI interrupt has been generated in EXDMAC module stop mode, this bit is not set to 1.</p>
18	—	0	R	Reserved
				They are always read as 0 and cannot be modified.

				<ul style="list-style-type: none"> <li>• Writing 1 to the DTE bit</li> </ul> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• Transfer size error interrupt request is generated</li> <li>• Repeat size end interrupt request is generated</li> <li>• Extended repeat area overflow end interrupt request is generated</li> </ul>
16	DTIF	0	R/(W)*	<p>Data Transfer Interrupt Flag</p> <p>Flag indicating that a transfer end interrupt request occurred by the transfer counter.</p> <p>0: Transfer end interrupt request is not generated by the transfer counter</p> <p>1: Transfer end interrupt request is generated by the transfer counter</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• Writing 1 to the DTE bit</li> <li>• Writing 0 to DTIF while reading DTIF = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When EDTCR becomes 0 and transfer has completed</li> </ul>
15	DTSZ1	0	R/W	Data Access Size 1 and 0
14	DTSZ0	0	R/W	<p>Selects the data access size.</p> <p>00: Byte-size (8 bits)</p> <p>01: Word-size (16 bits)</p> <p>10: Longword-size (32 bits)</p> <p>11: Setting prohibited</p>

request.

When this bit is set to 1 and the transfer count becomes smaller than the data access size for transfer request by EXDMAC transfer, the DTE is cleared to 0 by the next transfer request. At the same time, the ESIF bit is set to 1 to indicate that a transfer size error interrupt request is generated.

When cluster transfer read/write address mode is specified, this bit should be set to 1.

Transfer size error interrupt request occurs in the following conditions:

- In normal transfer and repeat transfer mode, the total transfer size set in EDTCR is smaller than the data access size
- In block transfer mode, the total transfer size set in EDTCR is smaller than the block size
- In cluster transfer mode, the total transfer size set in EDTCR is smaller than the cluster size

0: Transfer size error interrupt request disabled

1: Transfer size error interrupt request enabled

---

10	—	0	R	Reserved
----	---	---	---	----------

---

They are always read as 0 and cannot be modified.

8	DTIE	0	R/W	Data Transfer Interrupt Enable Enables or disables a transfer end interrupt request to the transfer counter. When this bit is set to 1 and the DTIF bit is set to 1, a transfer end interrupt is generated to the CPU or DTC. The transfer end interrupt is canceled by clearing this bit or the DTIF bit. 0: Transfer end interrupt request disabled 1: Transfer end interrupt request enabled
7	DTF1	0	R/W	Data Transfer Factor 1 and 0
6	DTF0	0	R/W	Selects a source to activate EXDMAC. For external requests, a sampling method is selected by the EDREQS bit. 00: Auto-request (cycle steal mode) 01: Auto-request (burst mode) 10: Setting prohibited 11: External request
5	—	0	R/W	Reserved The initial value should not be changed.

independently for each channel. This bit is enabled when the CPUPCE bit in CPUPCR is 1.

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)

---

Note: \* Only 0 can be written to these bits after 1 is read to clear the flag.

R/W	R	R	R/W	R/W	R	R	R/W
Bit	15	14	13	12	11	10	9
Bit Name	SARIE	—	—	SARA4	SARA3	SARA2	SARA1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	DARIE	—	—	DARA4	DARA3	DARA2	DARA1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
31	AMS	0	R/W	<p>Address Mode Select</p> <p>Selects single address mode or dual address mode. When single address mode is selected, <math>\overline{\text{EDACK}}</math> is not valid due to the EDACKE bit setting in EDMDR.</p> <p>0: Dual address mode</p> <p>1: Single address mode</p>
30	DIRS	0	R/W	<p>Single Address Direction Select</p> <p>Specifies the data transfer direction in single address mode. In dual address mode, the specification of this bit is ignored.</p> <p>In cluster transfer mode, the internal cluster bus can be the source or destination in place of the external device with <math>\overline{\text{DACK}}</math>.</p> <p>0: EDSAR transferred as a source address</p> <p>1: EDDAR transferred as a destination address</p>

Even if the repeat area is not specified (ARS1, B'10), the repeat size end interrupt can be requested at the end of a repeat-size transfer.

When this bit is set to 1 and the next transfer size is generated at the end of a block- or cluster-size transfer in block transfer or cluster transfer mode, the DEDMDR is cleared to 0. At the same time, the REPEAT\_SIZE bit in EDMDR is set to 1 to indicate that the repeat size end interrupt is requested.

0: Repeat size end interrupt request disabled

1: Repeat size end interrupt request enabled

25	ARS1	0	R/W	Area Select 1 and 0
24	ARS0	0	R/W	Select the block area or repeat area in block transfer, repeat transfer or cluster transfer mode. 00: Block area/repeat area on the source address side 01: Block area/repeat area on the destination address side 10: Block area/repeat area not specified 11: Setting prohibited
23, 22	—	All 0	R	Reserved They are always read as 0 and cannot be modified.



				11: Decrement (−1, −2, or −4 according to access size)
19, 18	—	All 0	R	Reserved They 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	These bits specify incrementing/decrementing transfer destination address (EDDAR). When transfer source is not specified in EDDAR in shift address mode, the specification by these bits is ignored. 00: Fixed 01: Offset added 10: Incremented (+1, +2, or +4 according to transfer access size) 11: Decrement (−1, −2, or −4 according to transfer access size)

When used together with block transfer mode, interrupt is requested at the end of a block-size transfer. If the DTE bit is set to 1 in EDMDR for the channel which transfer is terminated by an interrupt, transfer is resumed from the state in which it ended.

If a source address extended repeat area is not designated, the specification by this bit is ignored.

0: Source address extended repeat area overflow interrupt request disabled

1: Source address extended repeat area overflow interrupt request enabled

14, 13	—	All 0	R	Reserved They are always read as 0 and cannot be modified.
12	SARA4	0	R/W	Source Address Extended Repeat Area
11	SARA3	0	R/W	These bits specify the source address (EDSAR) of the extended repeat area. The extended repeat area function updates the specified lower address bits, leaving the remaining upper address bits always the same. An extended repeat area size of 4 bytes to 16 Mbytes can be specified. The setting interval is a power-of-two number of bytes.
10	SARA2	0	R/W	
9	SARA1	0	R/W	
8	SARA0	0	R/W	

When extended repeat area overflow results from incrementing or decrementing an address, the start address is the start address of the extended repeat area in the case of address incrementing, or the end address of the extended repeat area in the case of address decrementing.

If SARIE bit is set to 1, an interrupt can be requested when an extended repeat area overflow occurs.

Table 11.3 shows the settings and ranges of the extended repeat area.

When used together with block transfer mode, an interrupt is requested at the end of a block-size transfer. If DTE bit is set to 1 in EDMDR for the channel, which transfer is terminated by an interrupt, transfer can be resumed from the state in which it ended. If the destination address extended repeat area is not designated, the specification by this bit is ignored.

0: Destination address extended repeat area interrupt request disabled  
 1: Destination address extended repeat area interrupt request enabled

6, 5	—	All 0	R	Reserved They are always read as 0 and cannot be modified.
4	DARA4	0	R/W	Destination Address Extended Repeat Area Enable
3	DARA3	0	R/W	These bits specify the destination address (E) of the extended repeat area. The extended repeat area function updates the specified lower address bits, leaving the remaining upper address bits always the same. An extended repeat area size of 4 bytes to 128 Mbytes can be specified. The setting interval is a power-of-two of bytes. When extended repeat area overflow results from incrementing or decrementing an address, the start address is the start address of the extended repeat area in the case of address incrementing, or the end address of the extended repeat area in the case of address decrementing. If the DARIE bit is set to 1, an interrupt can be requested when an extended repeat area overflow occurs. Table 11.3 shows the settings and ranges of the extended repeat area.
2	DARA2	0	R/W	
1	DARA1	0	R/W	
0	DARA0	0	R/W	

00110	Lower 6 bit (64-byte area) designated as extended repeat
00111	Lower 7 bit (128-byte area) designated as extended repeat
01000	Lower 8 bit (256-byte area) designated as extended repeat
01001	Lower 9 bit (512-byte area) designated as extended repeat
01010	Lower 10 bit (1-Kbyte area) designated as extended repeat
01011	Lower 11 bit (2-Kbyte area) designated as extended repeat
01100	Lower 12 bit (4-Kbyte area) designated as extended repeat
01101	Lower 13 bit (8-Kbyte area) designated as extended repeat
01110	Lower 14 bit (16-Kbyte area) designated as extended repeat
01111	Lower 15 bit (32-Kbyte area) designated as extended repeat
10000	Lower 16 bit (64-Kbyte area) designated as extended repeat
10001	Lower 17 bit (128-Kbyte area) designated as extended repeat
10010	Lower 18 bit (256-Kbyte area) designated as extended repeat
10011	Lower 19 bit (512-Kbyte area) designated as extended repeat
10100	Lower 20 bit (1-Mbyte area) designated as extended repeat
10101	Lower 21 bit (2-Mbyte area) designated as extended repeat
10110	Lower 22 bit (4-Mbyte area) designated as extended repeat
10111	Lower 23 bit (8-Mbyte area) designated as extended repeat
11000	Lower 24 bit (16-Mbyte area) designated as extended repeat
11001	Lower 25 bit (32-Mbyte area) designated as extended repeat
11010	Lower 26 bit (64-Mbyte area) designated as extended repeat
11011	Lower 27 bit (128-Mbyte area) designated as extended repeat
111XX	Setting prohibited

[Legend] X: Don't care

In cluster transfer mode, the same CLSBR is used for all channels. When the CPU writes to CLSBR conflicts with cluster transfer, the contents of transferred data are not guaranteed. In cluster transfer read/write address mode is specified and if another channel is set for cluster transfer, the transferred data may be overwritten.

Bit	31	30	29	28	27	26	25	
Bit Name								
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
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	
Bit Name								
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
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	
Bit Name								
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
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	
Bit Name								
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Mode	Transfer mode	Activation source	Common function	Source	Destination
Dual address mode	<ul style="list-style-type: none"> <li>Normal transfer mode</li> <li>Repeat transfer mode</li> <li>Block transfer mode</li> </ul> (Repeat size/ block size = 1 to 65,536 bytes/ word/longword)	<ul style="list-style-type: none"> <li>Auto-request (activated by the CPU)</li> <li>External request</li> </ul>	<ul style="list-style-type: none"> <li>Total transfer size: 1 to 4 Gbytes, or no specification</li> <li>Offset addition</li> <li>Extended repeat area function</li> </ul>	EDSAR	EDACK
Single address mode	<ul style="list-style-type: none"> <li>Direct data transfer to/from external devices using EDACK pin instead of source or destination address register</li> <li>Above transfer mode can be specified in addition to address register setting</li> <li>One transfer possible in one bus cycle</li> </ul> (Transfer mode variations are the same as in dual address mode.)			EDSAR/ EDACK	EDACK/ EDACK

When the activation source is an auto-request, cycle steal mode or burst mode can be selected.

When the total transfer size is not specified (EDTCR = H'00000000), the transfer counter is not incremented and the transfer count is not restricted, allowing continuous transfer.

Dual address mode	(activated by the CPU)	One access size (byte/word/longword) to 32 bytes		the transfer source and written to the transfer destination
	• External request	• Total transfer size		
Cluster transfer		1 to 4 Gbytes, or no specification	EDSAR	Read from the transfer source
Read address mode (DIRS = 0)		• Offset addition		
Cluster transfer		• Extended repeat area function	—	Written to the transfer destination
Write address mode (DIRS = 1)				

In cluster transfer mode, the specified cluster size is transferred in response to a single transfer request. The cluster size can be from one access size (byte, word, or longword) to 32 bytes. With a cluster, a cluster-size transfer is performed in burst transfer mode. With a cluster-size transfer, a cluster transfer mode (dual address mode), block transfer mode (dual address mode) is used.

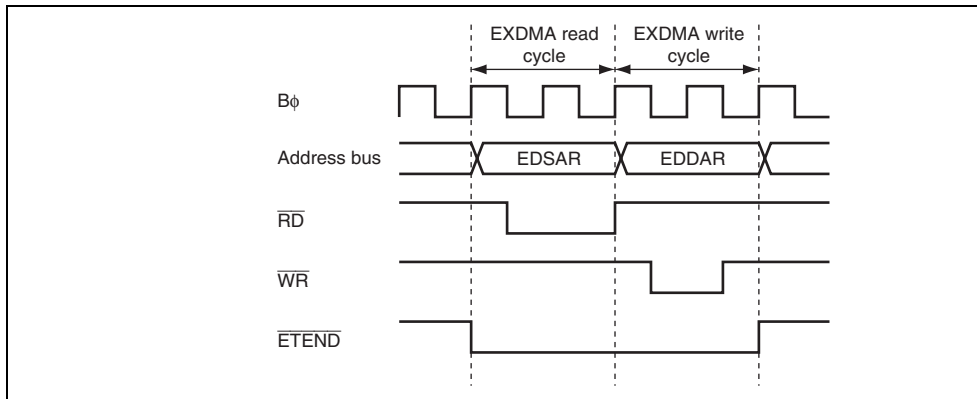
With auto-requests, cycle steal mode is set.

In a transfer operation, the data on the transfer source address is read in the first bus cycle and written to the transfer destination address in the next bus cycle.

These consecutive read and write cycles are indivisible: another bus cycle (external access by another bus master, refresh cycle, or external bus release cycle) does not occur between the two cycles.

$\overline{\text{ETEND}}$  pin output can be enabled or disabled by means of the ETENDE bit in EDMDR.  $\overline{\text{ETEND}}$  is output for two consecutive bus cycles. When an idle cycle is inserted before the bus cycle,  $\overline{\text{ETEND}}$  signal is also output in the idle cycle. The  $\overline{\text{EDACK}}$  signal is not output.

Figure 11.2 shows an example of the timing in dual address mode and figure 11.3 shows address mode operation.



**Figure 11.2 Example of Timing in Dual Address Mode**



## (2) Single Address Mode

In single address mode, the  $\overline{\text{EDACK}}$  pin is used instead of EDSAR or EDDAR to transfer data directly between an external device and external memory. One transfer operation is executed in one bus cycle.

In this mode, the data bus width must be the same as the data access size. For details on data bus width, see section 9, Bus Controller (BSC).

In this mode, the EXDMAC accesses the transfer source or transfer destination external device by outputting the strobe signal  $\overline{\text{EDACK}}$  for the external device with  $\overline{\text{DACK}}$ , and at the same time, the EXDMAC accesses the other external device in the transfer by outputting an address. In this way, one transfer can be executed in one bus cycle. In the example of transfer between external memory and an external device with  $\overline{\text{DACK}}$  shown in figure 11.4, data is output to the data bus by the external device and written to external memory in the same bus cycle.

The transfer direction, that is whether the external device with  $\overline{\text{DACK}}$  is the transfer source or transfer destination, can be specified with the DIRS bit in EDACR. Transfer is performed from external memory (EDSAR) to the external device with  $\overline{\text{DACK}}$  when DIRS = 0, and from the external device with  $\overline{\text{DACK}}$  to the external memory (EDDAR) when DIRS = 1. The setting of the source or destination address register not used in the transfer is ignored.

The  $\overline{\text{EDACK}}$  pin output is valid by the setting of EDACKE bit in EDMDR when single address mode is selected. The  $\overline{\text{EDACK}}$  pin output is active-low.

$\overline{\text{ETEND}}$  pin output can be enabled or disabled by means of the ETENDE bit in EDMDR.  $\overline{\text{ETEND}}$  is output for one bus cycle. When an idle cycle is inserted before the bus cycle, the  $\overline{\text{ETEND}}$  pin is also output in the idle cycle.

Figure 11.5 shows an example of the timing in single address mode and figure 11.6 shows an example of single address mode operation.

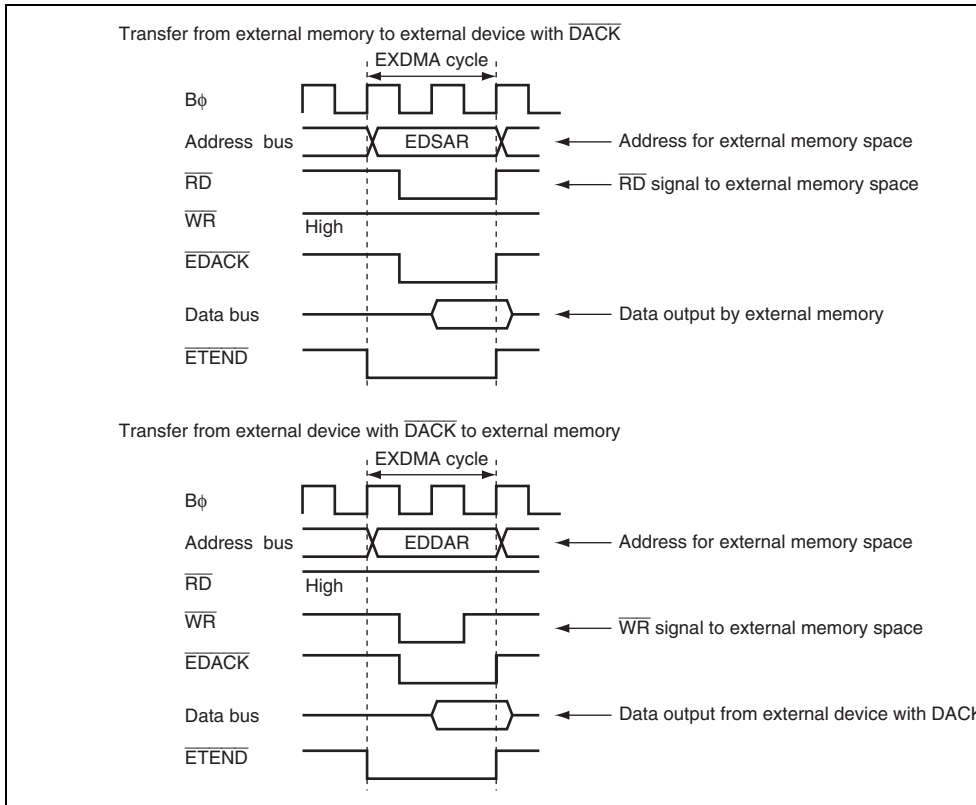


Figure 11.5 Example of Timing in Single Address Mode

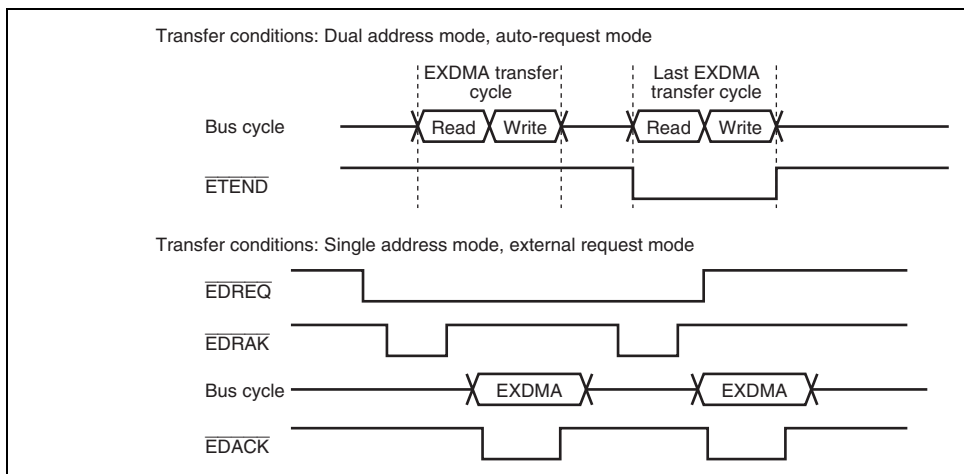
## 11.5.2 Transfer Modes

### (1) Normal Transfer Mode

In normal transfer mode, transfer of one data access size unit is processed in response to transfer request. The total transfer size of up to 4 Gbytes can be set by EDTCR. EDBSR in normal transfer mode.

The  $\overline{\text{ETEND}}$  signal is output only for the last EXDMA transfer. The  $\overline{\text{EDRAK}}$  signal is output each time a transfer request is accepted and transfer processing is started.

Figure 11.7 shows examples of transfer timing in normal transfer mode and figure 11.8 shows normal transfer mode operation in dual address mode.



**Figure 11.7 Examples of Timing in Normal Transfer Mode**

In repeat transfer mode, transfer of one data access size unit is processed in response to one transfer request. The total transfer size of up to 4 Gbytes can be set by EDTCR. The repeat size of up to 64 Kbytes  $\times$  data access size can be set by EDBSR.

The ARS1 and ARS0 bits in EDACR specify the repeat area on the source address or destination address side. The address specified for the repeat area is restored to the transfer start address at the end of a repeat-size transfer. This operation continues until transfer of total transfer size specified by EDTCR ends. EDTCR specified with H'00000000 is assumed as free-running mode and the transfer continues until the DTE bit in EDMDR is cleared to 0.

At the end of a repeat-size transfer, the EXDMA transfer is halted temporarily and a repeat-size transfer end interrupt is requested to the CPU or DTC. When the RPTIE bit in EDACR is set to 1, the next transfer request is generated at the end of a repeat-size transfer, the ESIF bit in EDMDR is set to 1 and the DTE bit in EDMDR is cleared to 0 to terminate the transfer. At this time, a repeat-size transfer end interrupt is requested to the CPU or DTC when the ESIE bit in EDMDR is set to 1.

The timing of EXDMA transfer including the  $\overline{\text{ETEND}}$  or  $\overline{\text{EDRAK}}$  output is the same as for normal transfer mode.

Figure 11.9 shows the repeat transfer mode operation in dual address mode.

The operation without specifying a repeat area on the source or destination address side is the same as for the normal transfer mode operation shown in figure 11.8. In this case, a repeat-size transfer end interrupt can also be generated at the end of a repeat-size transfer.

### (3) Block Transfer Mode

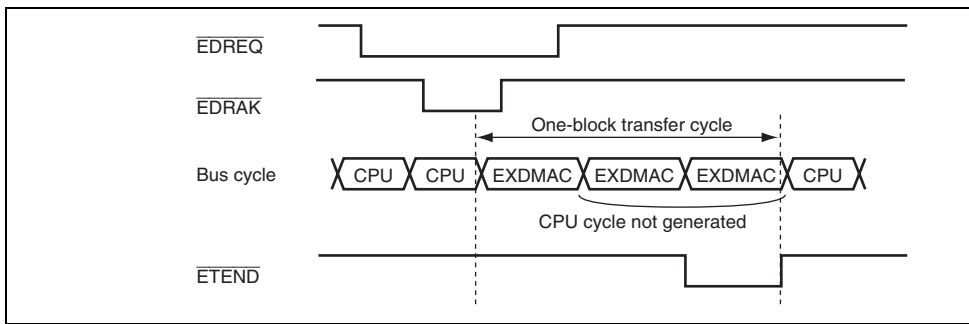
In block transfer mode, transfer of one block size unit is processed in response to one transfer request. The total transfer size of up to 4 Gbytes can be set by EDTCR. The block size of Kbytes  $\times$  data access size can be set by EDBSR.

A transfer request from another channel is held pending during one block transfer. When a block transfer is completed, the bus mastership is released for another bus master.

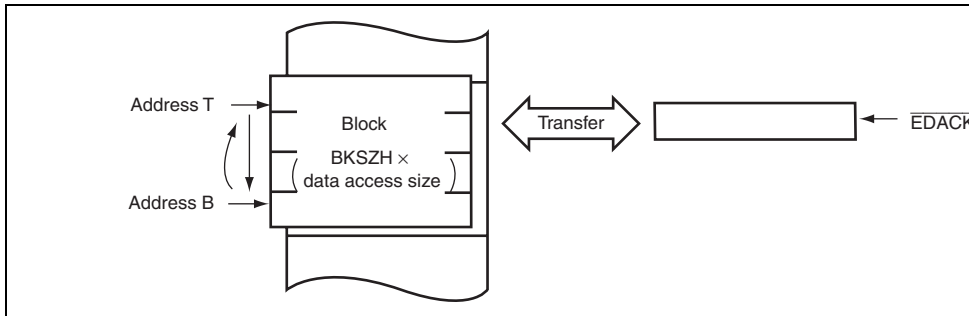
A block area can be specified by the ARS1 or ARS0 bit in EDACR on the source or destination address side. The address specified for the block area is restored to the transfer start address when one-block transfer completes. When no repeat area is specified on the source and destination address sides, the address is not restored to the transfer start address and the operation proceeds to the next sequence. A repeat size end interrupt can be generated.

The  $\overline{\text{ETEND}}$  signal is output for each block transfer in the EXDMA transfer cycle in which the block ends. The  $\overline{\text{EDRAK}}$  signal is output once for one transfer request (for transfer of one block).

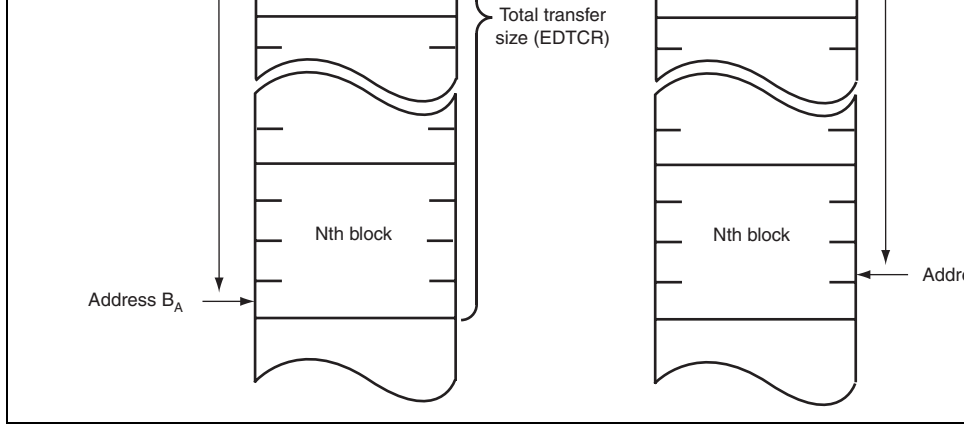
Caution is required when setting the extended repeat area overflow interrupt in block transfer mode. For details, see section 11.5.5, Extended Repeat Area Function.



**Figure 11.10 Example of Block Transfer Mode**



**Figure 11.11 Block Transfer Mode Operation in Single Address Mode (with Block Area Specified)**



**Figure 11.12 Block Transfer Mode Operation in Dual Address Mode (without BL Specified)**

request activation. The bus mode can be selected from cycle steal mode and burst mode via request activation.

## (2) Activation by External Request

Transfer is started by the transfer request signal ( $\overline{\text{EDREQ}}$ ) from the external device for activation by an external request. When the EXDMA transfer is enabled ( $\text{DTE} = 1$ ), the EXDMA transfer starts by  $\overline{\text{EDREQ}}$  input.

The transfer request signal is accepted by the  $\overline{\text{EDREQ}}$  pin. The EDREQS bit in EDMDR selects whether the  $\overline{\text{EDREQ}}$  is detected by falling edge sensing or low level sensing.

When the EDRAKE bit in EDMDR is set to 1, the signal notifying transfer request acceptance is output from the  $\overline{\text{EDRAK}}$  pin. The  $\overline{\text{EDRAK}}$  signal is accepted for one external request and output when transfer processing starts.

When specifying an external request as an activation source, set the DDR bit to 0 and the DTR to 1 on the corresponding pin in advance. For details, see section 13, I/O Ports.

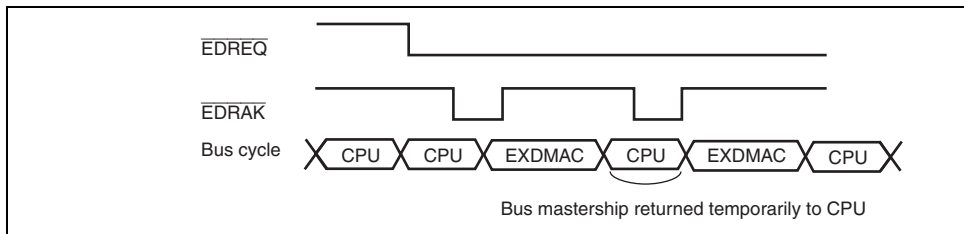


transfer request, the EXDMAC takes back the bus mastership, performs another transfer, and then releases the bus mastership again at the end of the transfer. This process is repeated until the transfer end condition is satisfied.

If a transfer request occurs in another channel during EXDMA transfer, the bus mastership is temporarily released for another bus master, then transfer is performed on the channel for which the transfer request was issued. For details on the operation when there are transfer requests in a large number of channels, see section 11.5.8, Channel Priority Order.

Figure 11.13 shows an example of the timing in cycle steal mode. The transfer condition is as follows:

- Address mode: Single address mode
- Sampling method on the  $\overline{\text{EDREQ}}$  pin: Low level sensing
- CPU internal bus master is operating in external space

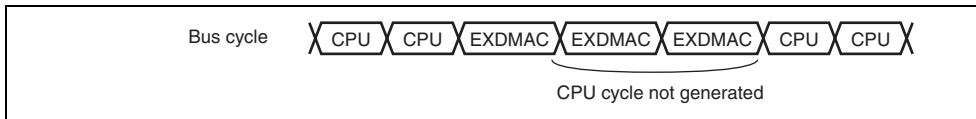


**Figure 11.13 Example of Timing in Cycle Steal Mode**

In block transfer mode and cluster transfer mode, the setting of burst mode is invalid (one-cluster transfer is processed in the same way as in burst mode). The EXDMAC always operates in cycle steal mode.

When the DTE bit is cleared to 0 in EDMDR, EXDMA transfer is halted. However, EXDMA transfer is executed for all transfer requests generated within the EXDMAC until the DTE bit is cleared to 0. If a transfer size error interrupt, a repeat size end interrupt, or extended repeat overflow interrupt is generated, the DTE bit is cleared to 0 and transfer is terminated.

Figure 11.14 shows an example of the timing in burst mode.



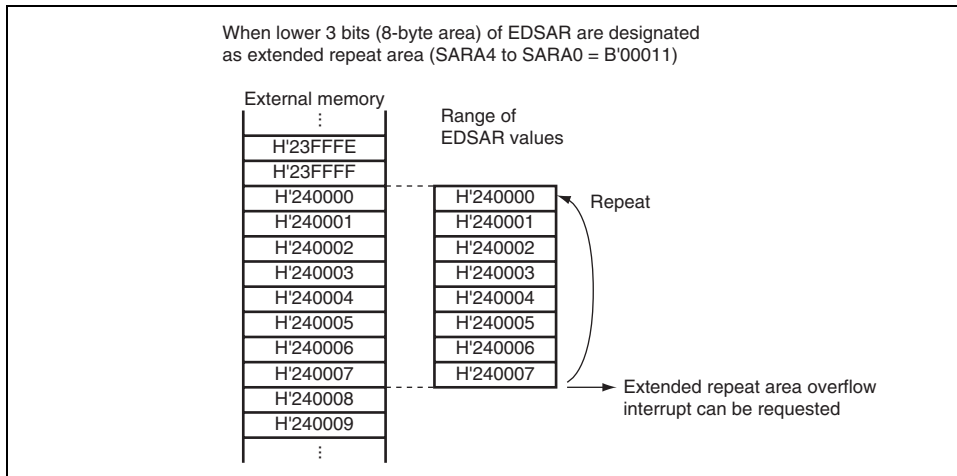
**Figure 11.14 Example of Timing in Burst Mode**

### 11.5.5 Extended Repeat Area Function

The EXDMAC has a function for designating an extended repeat area for source address and destination addresses. When an extended repeat area is designated, the address register value repeats within the range specified as the extended repeat area. Normally, when a ring buffer is involved in a transfer, an operation is required to restore the address register value to the start address each time the address register value becomes the last address in the buffer (i.e., ring buffer address overflow occurs). However, if the extended repeat area function is used, an operation that restores the address register value to the buffer start address is processed automatically within the EXDMAC.

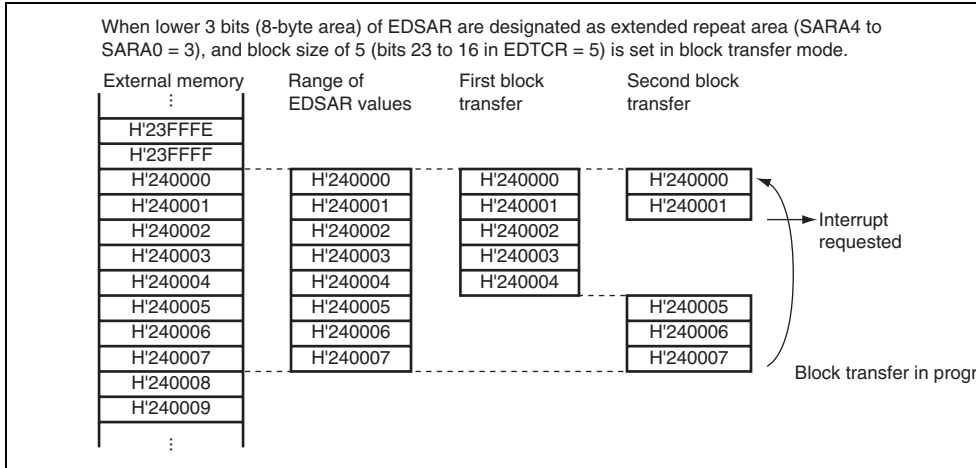
The extended repeat area function can be set independently for the source address register (EDSAR) and the destination address register (EDDAR).

Figure 11.15 illustrates the operation of the extended repeat area function.

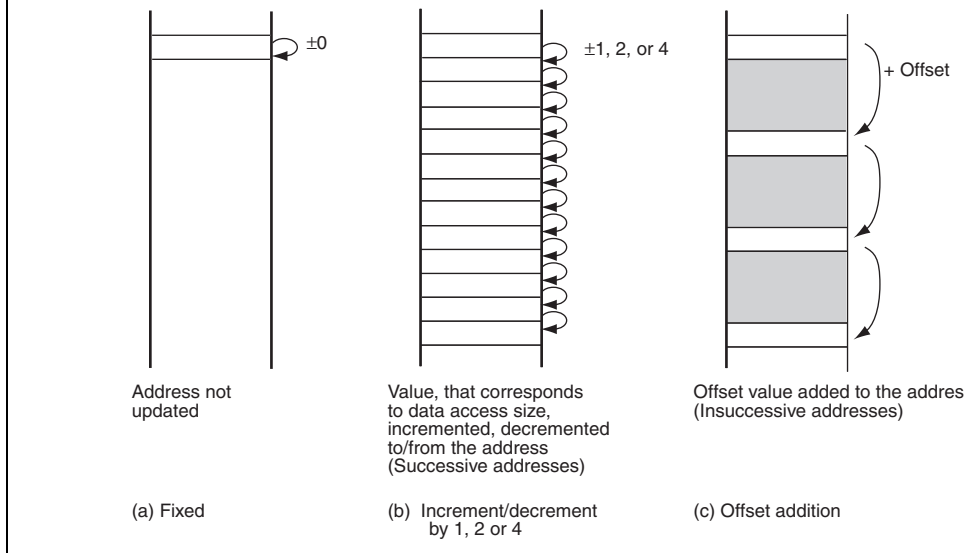


**Figure 11.15 Example of Extended Repeat Area Function Operation**

Figure 11.16 shows an example in which block transfer mode is used together with the extended repeat area function.



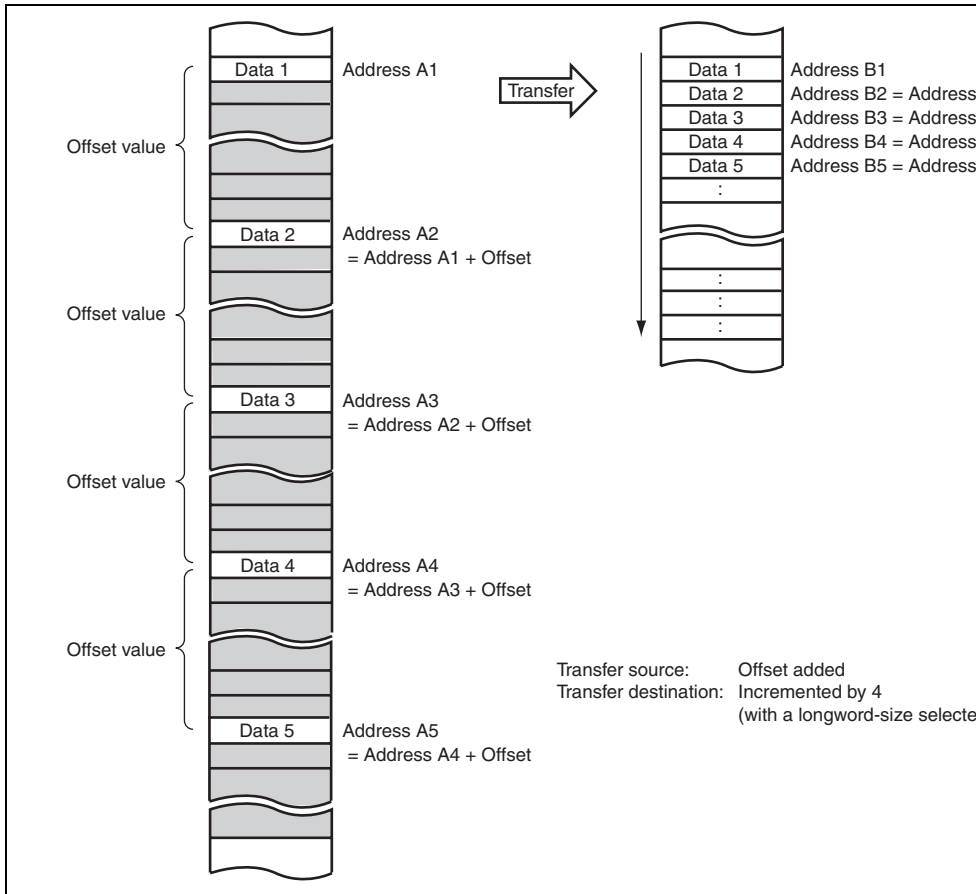
**Figure 11.16 Example of Extended Repeat Area Function Operation in Block Transfer Mode**



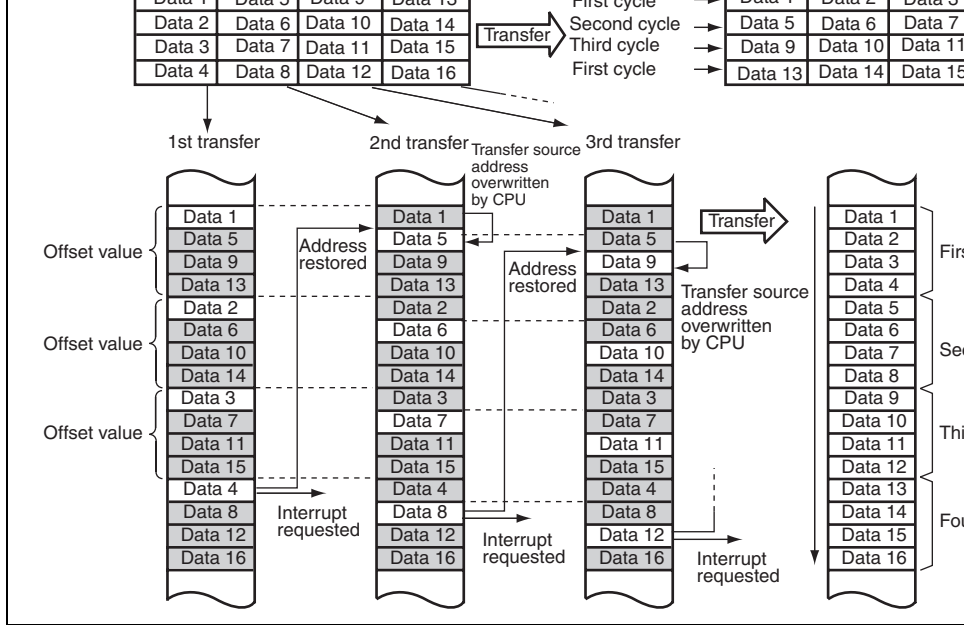
**Figure 11.17 Address Update Method**

For the fixed method (a), the same address is always indicated without the transfer destination or source address being updated.

For the method of increment/decrement by 1, 2 or 4 (b), the value corresponding to the data access size is incremented or decremented to or from the transfer destination or source address. A byte, word, or longword can be specified for the data access size. The value used for increment or decrement of an address is 1 for a byte-size, 2 for a word-size, and 4 for a longword-size transfer. This function allows continuous address transfer of EXDM.

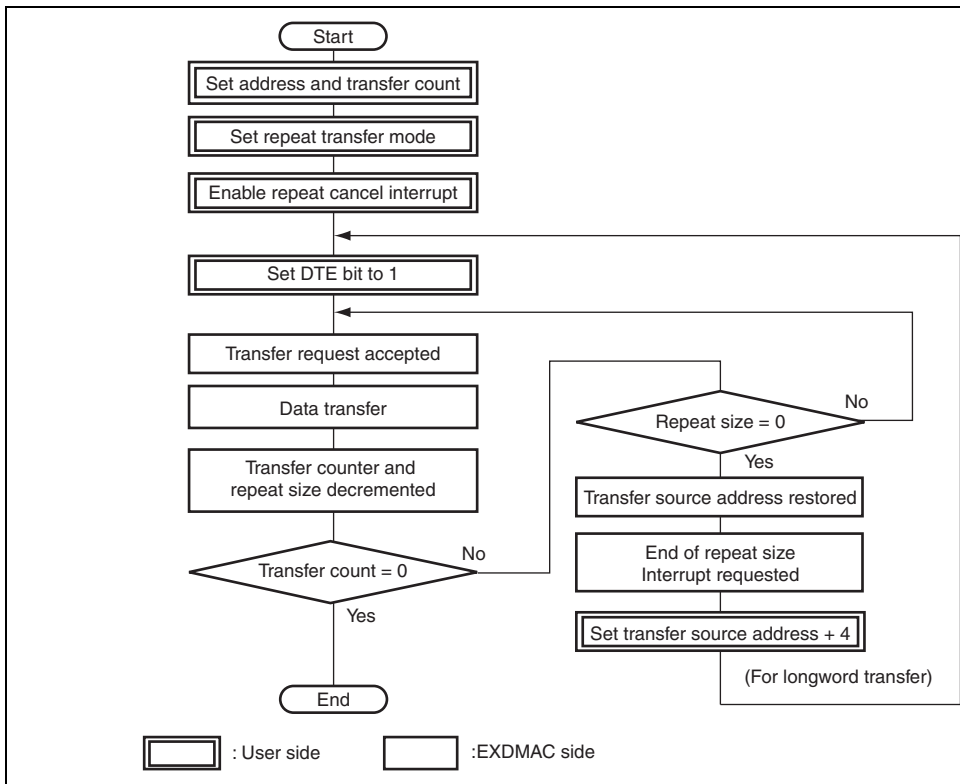


**Figure 11.18 Address Update Function Using Offset**



**Figure 11.19 XY Conversion by Combining Repeat Transfer Mode and Offset Addressing**

In figure 11.19, the source address side is set as a repeat area in EDACR and the offset address is set in EDACR. The offset value is the address that corresponds to  $4 \times$  data access size (example: for a longword-size transfer, H'00000010 is specified in EDOFR). The repeat size is  $4 \times$  data access size (example: for a longword-size transfer,  $4 \times 4 = 16$  bytes are specified as a repeat size). The increment by 1, 2 or 4 is set for the transfer destination. The RPTIE bit in EDACR is used to generate a repeat size end interrupt request at the end of a repeat-size transfer.



**Figure 11.20 Flow of XY Conversion Combining Repeat Transfer Mode and Counter Addition**



### 11.5.7 Registers during EXDMA Transfer Operation

EXDMAC register values are updated as EXDMA transfer processing is performed. The values depend on various settings and the transfer status. The following registers and bits are updated: EDSAR, EDDAR, EDTCR, bits BKSZH and BKSZ in EDBSR, and bits DTE, ERRF, ESIF and DTIF in EDMDR.

#### (1) EXDMA Source Address Register (EDSAR)

When the EDSAR address is accessed as the transfer source, the EDSAR value is output. EDSAR is updated with the address to be accessed next.

Bits SAT1 and SAT0 in EDACR specify incrementing or decrementing. The address is incremented when SAT1 and SAT0 = B'00, incremented by offset register value when SAT1 and SAT0 = B'01, incremented when SAT1 and SAT0 = B'10, and decremented when SAT1 and SAT0 = B'11. (The increment or decrement value is determined by the data access size.)

The DTSZ1 and DTSZ0 bits in EDMDR set the data access size. When DTSZ1 and DTSZ0 = B'00, the data is byte-size and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data is word-size and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data is longword-size and the address is incremented or decremented by 4. When a word-size or longword-size is specified but the source address is not aligned to the word or longword boundary, the data is divided into bytes or words for reading. When a byte-size or word-size is specified but the source address is not aligned to the byte or longword boundary, the word or longword is divided for reading, the address is incremented or decremented by 1 or 2 to an actual byte-or word-size read. After a word-size or longword-size read, the address is incremented or decremented to or from the read start address according to the setting of SAT0.

## (2) EXDMA Destination Address Register (EDDAR)

When the EDDAR address is accessed as the transfer destination, the EDDAR value is output, and then EDDAR is updated with the address to be accessed next.

Bits DAT1 and DAT0 in EDACR specify incrementing or decrementing. The address is incremented when DAT1 and DAT0 = B'00, incremented by offset register value when DAT1 and DAT0 = B'01, incremented when DAT1 and DAT0 = B'10, and decremented when DAT1 and DAT0 = B'11. (The increment or decrement value is determined by the data access size.)

The DTSZ1 and DTSZ0 bits in EDMDR set the data access size. When DTSZ1 and DTSZ0 = B'00, the data is byte-size and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data is word-size and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data is longword-size and the address is incremented or decremented by 4. When a word-size or longword-size is specified but the destination address is not at the word or longword boundary, the data is divided into bytes or words for writing. When a word or a longword is divided for writing, the address is incremented or decremented by the data access size according to an actual byte- or word-size written. After a word-size or longword-size write, the destination address is incremented or decremented to or from the write start address according to the DTSZ1 and SAT0.

When a block area (repeat area) is set for the destination address in block transfer mode (block transfer mode), the destination address is restored to the transfer start address at the end of the block transfer (repeat-size) transfer and is not affected by address updating.

When an extended repeat area is set for the destination address, the operation conforms to the block transfer mode setting. The upper addresses set for the extended repeat area is fixed, and is not affected by address updating.

EDTCR value does not change.

All of the bits of EDTCR may change, so when EDTCR is read by the CPU during EXDMA transfer, a longword access must be used. During a transfer operation, EDTCR may be updated without regard to accesses from the CPU, and the correct values may not be read if the upper and lower words are read separately. Do not write to EDTCR for a channel on which a transfer operation is in progress.

If there is conflict between an address update associated with EXDMA transfer and a write to the CPU, the CPU write has priority.

In the event of conflict between an EDTCR update from 1, 2, or 4 to 0 and a write (of a value) by the CPU, the CPU write value has priority as the EDTCR value, but transfer is terminated.

#### **(4) EXDMA Block Size Register (EDBSR)**

EDBSR is valid in block transfer or repeat transfer mode. EDBSR31 and EDBSR16 are BKSZH and EDBSR15 and EDBSR0 for BKSZ. The 16 bits of BKSZH holds a block size and repeat size and their values do not change. The 16 bits of BKSZ functions as a block size counter, the value of which is decremented by 1 when one data transfer is performed. When the BKSZ value is determined as 0 during EXDMA transfer, the EXDMAC does not store BKSZ and stores the BKSZH value.

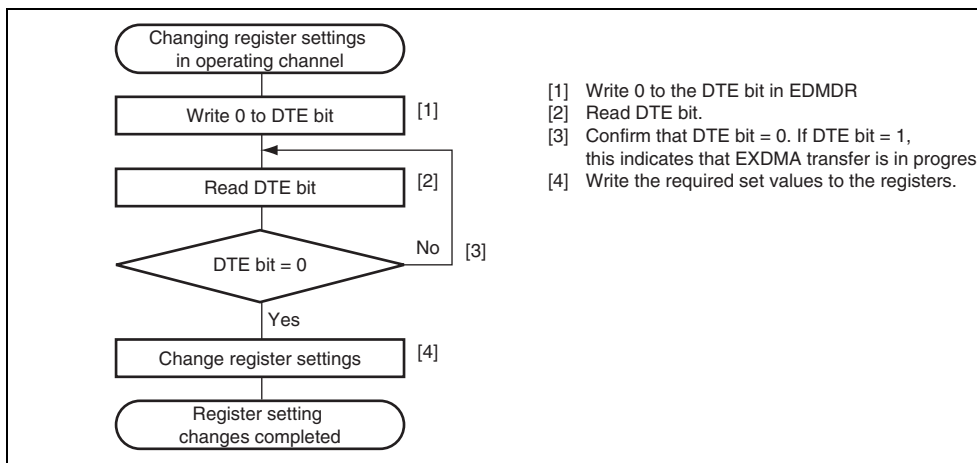
The upper 16 bits of EDBSR is never updated, allowing a word-size access.

Do not write to EDBSR for a channel on which a transfer operation is in progress.

- When an extended repeat area overflow interrupt is requested, and transfer ends
- When an NMI interrupt is generated, and transfer halts
- When an address error is generated, and transfer halts
- A reset
- Hardware standby mode
- When 0 is written to the DTE bit, and transfer halts

Writes (except to the DTE bit) are prohibited to registers of a channel for which the DTE bit is 1. When changing register settings after a 0-write to the DTE bit, it is necessary to confirm that the DTE bit has been cleared to 0.

Figure 11.21 shows the procedure for changing register settings in an operating channel.



**Figure 11.21 Procedure for Changing Register Settings in Operating Channel**

the bus cycle in which 0 is written to the DTE bit has been processed. The ACT bit is he  
between termination of the last EXDMA cycle and 0-write in the DTE bit.

### **(7) ERRF bit in EDMDR**

This bit specifies termination of transfer by EXDMAC clearing the DTE bit to 0 for all o  
an address error or NMI interrupt is generated. The EXDMAC also sets 1 to the ERRF b  
EDMDR\_0 regardless of the EXDMAC operation to indicate that an address error or NI  
interrupt is generated. However, when an address error or an NMI interrupt has been gen  
EXDMAC module stop mode, the ERRF bit is not set to 1.

### **(8) ESIF bit in EDMDR**

The ESIF bit in EDMDR is set to 1 when a transfer size interrupt, repeat size end interr  
extended repeat area overflow interrupt is requested. When the ESIF bit is set to 1 and th  
bit in EDMDR is set to 1, a transfer escape interrupt is requested to the CPU or DTC.

The timing that the ESIF bit is set to 1 is when the EXDMA transfer bus cycle (the sour  
interrupt request) terminates, the ACT bit in EDMDR is set to 0, and transfer is terminat

When the DTE bit is set to 1 to resume transfer during interrupt processing, the ESIF bit  
automatically cleared to 0 to cancel the interrupt request.

For details on interrupts, see section 11.9, Interrupt Sources.

### 11.5.8 Channel Priority Order

The priority order of the EXDMAC channels is: channel 0 > channel 1 > channel 2 > channel 3.

Table 11.6 shows the EXDMAC channel priority order.

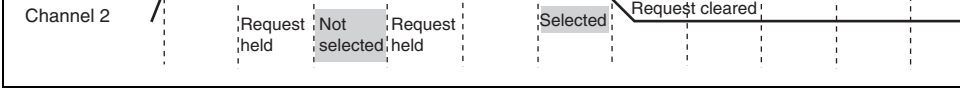
**Table 11.6 EXDMAC Channel Priority Order**

Channel	Channel P
Channel 0	High
Channel 1	↑
Channel 2	↑
Channel 3	Low

If transfer requests occur simultaneously for a number of channels, the highest-priority channel according to the priority order is selected for transfer. Transfer starts after the channel in priority releases the bus. If a bus request is issued from another bus master other than EXDMAC during transfer operation, another bus master cycle is initiated.

Channels are not switched during burst transfer, a block-size transfer in block transfer mode or cluster-size transfer in cluster transfer mode.

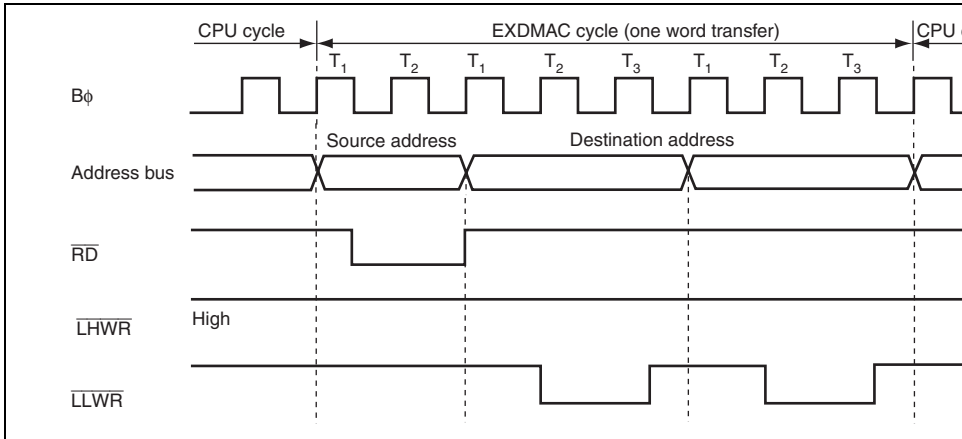
Figure 11.22 shows an example of the transfer timing when transfer requests occur simultaneously for channels 0, 1, and 2.



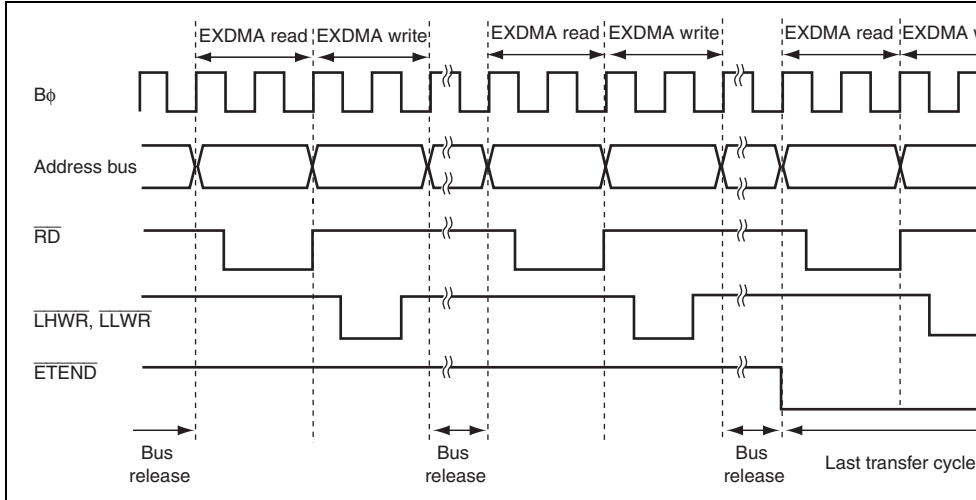
**Figure 11.22 Example of Channel Priority Timing**

### 11.5.9 Basic Bus Cycles

An example of the basic bus cycle timing is shown in figure 11.23. In this example, word transfer is performed from 16-bit, 2-state access space to 8-bit, 3-state access space. When mastership is transferred from the CPU to the EXDMAC, a source address read and destination address write are performed. The bus is not released in response to another bus request, between these read and write operations. As like CPU cycles, EXDMAC cycles conform to bus controller settings.



**Figure 11.23 Example of EXDMA Transfer Bus Timing**



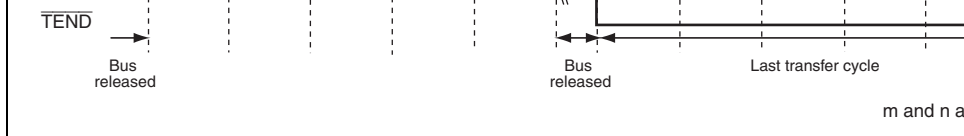
**Figure 11.24 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer**

Figures 11.25 and 11.26 show examples of transfer when  $\overline{\text{ETEND}}$  output is enabled, and longword-size, normal transfer mode (cycle steal mode) is performed from external 16-bit access space to external 16-bit, 2-state access space.

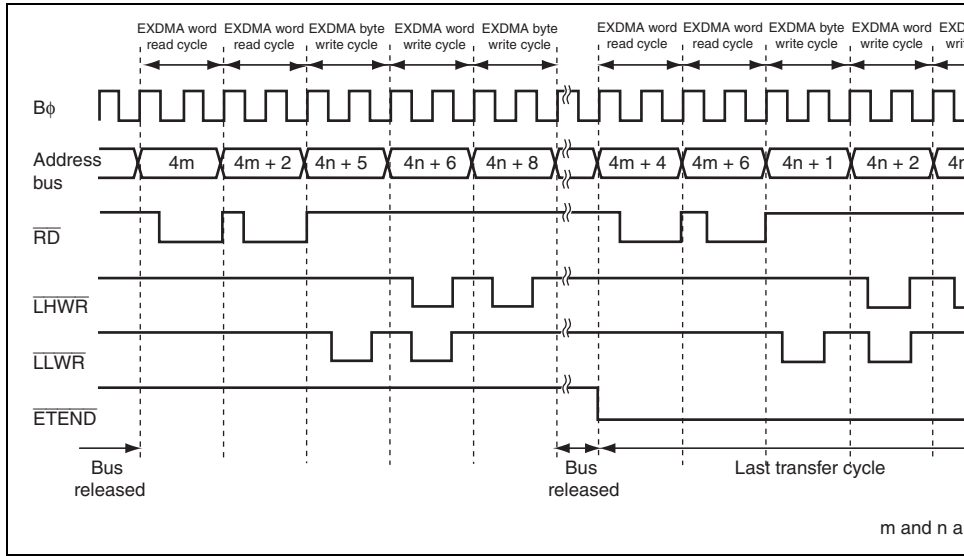
In figure 11.25, the transfer source (SAR) address is not at a longword boundary and the destination (DAR) address is at the longword boundary.

In figure 11.26, the transfer source (SAR) address is at the longword boundary and the destination (DAR) address is not at the longword boundary.

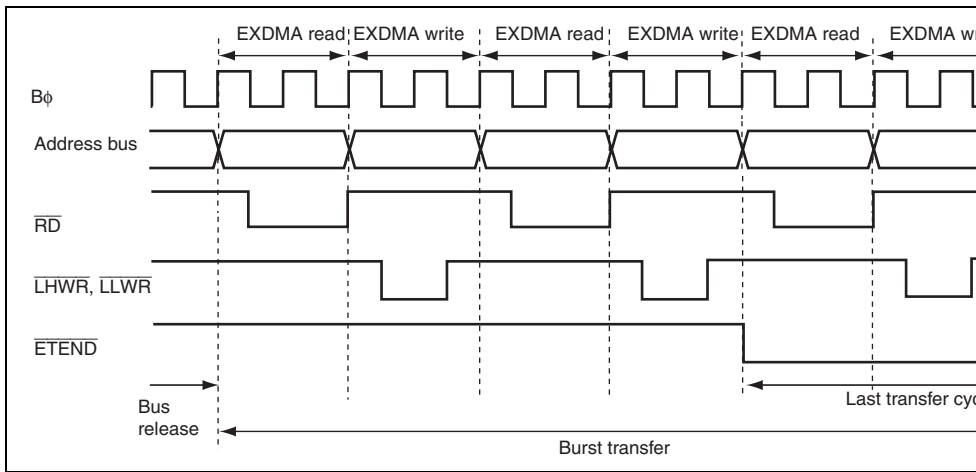




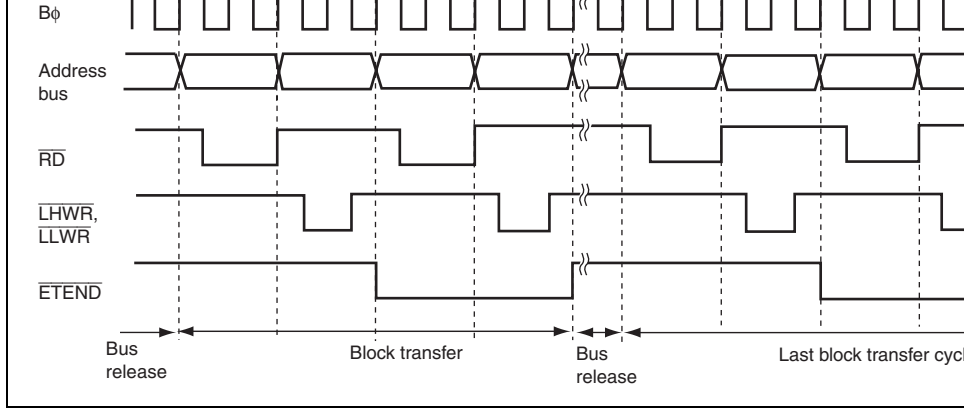
**Figure 11.25 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer (Transfer Source EDSAR = Odd Address, Source Address Incremented)**



**Figure 11.26 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer (Transfer Destination EDDAR = Odd Address, Destination Address Decremented)**

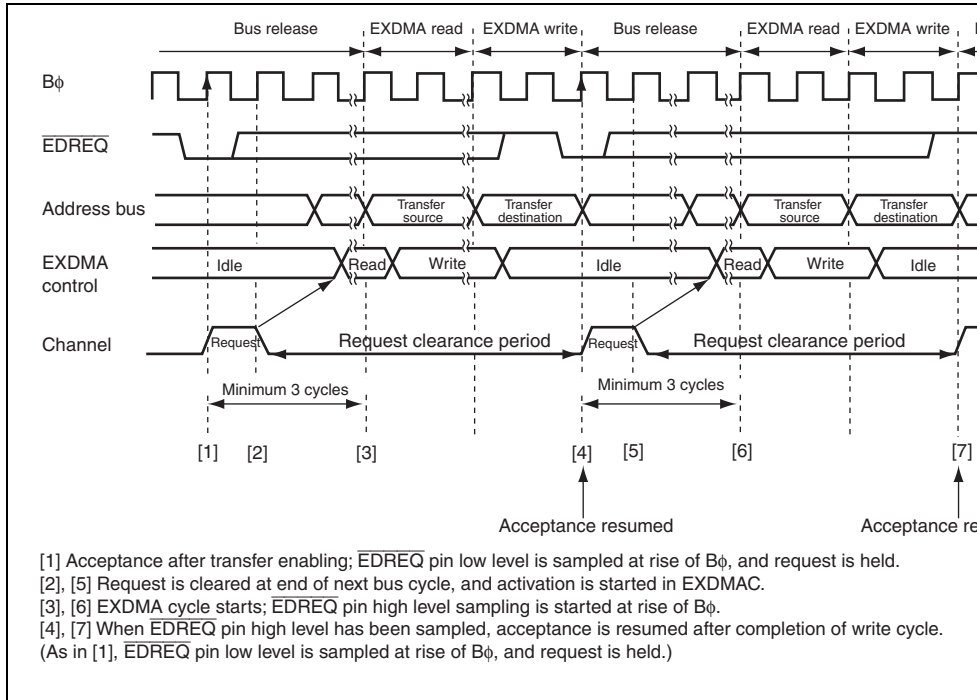


**Figure 11.27 Example of Normal Transfer Mode (Burst Mode) Transfer**



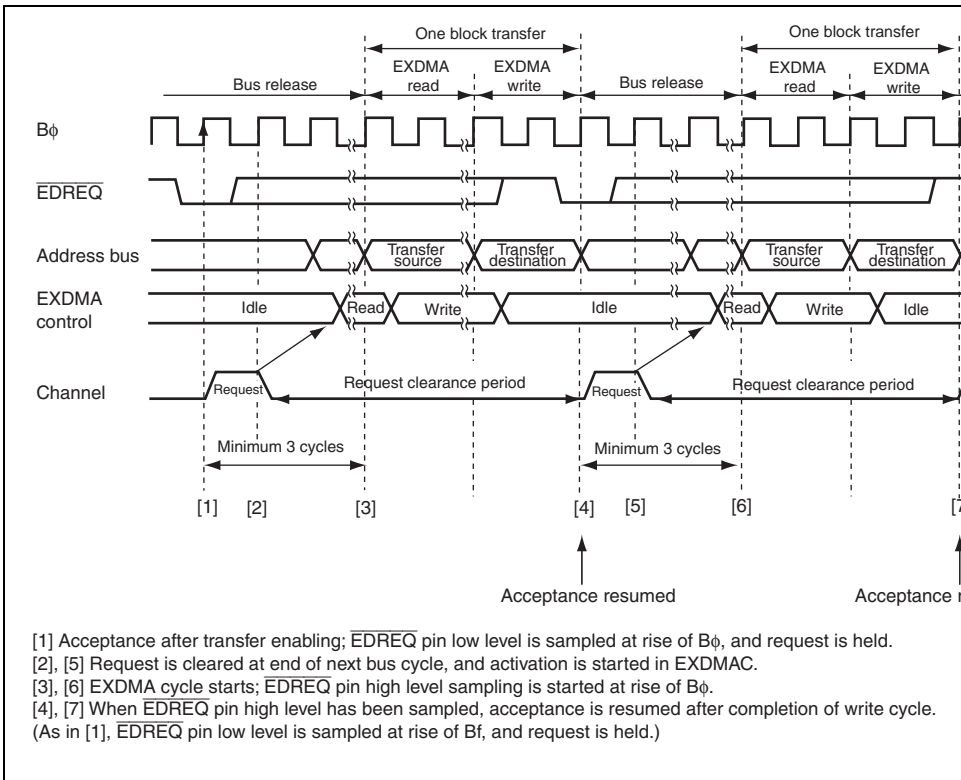
**Figure 11.28 Example of Block Transfer Mode Transfer**

is started. If  $\overline{\text{EDREQ}}$  pin high level sampling is completed by the end of the EXDMA write cycle, acceptance resumes after the end of the write cycle, and  $\overline{\text{EDREQ}}$  pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.



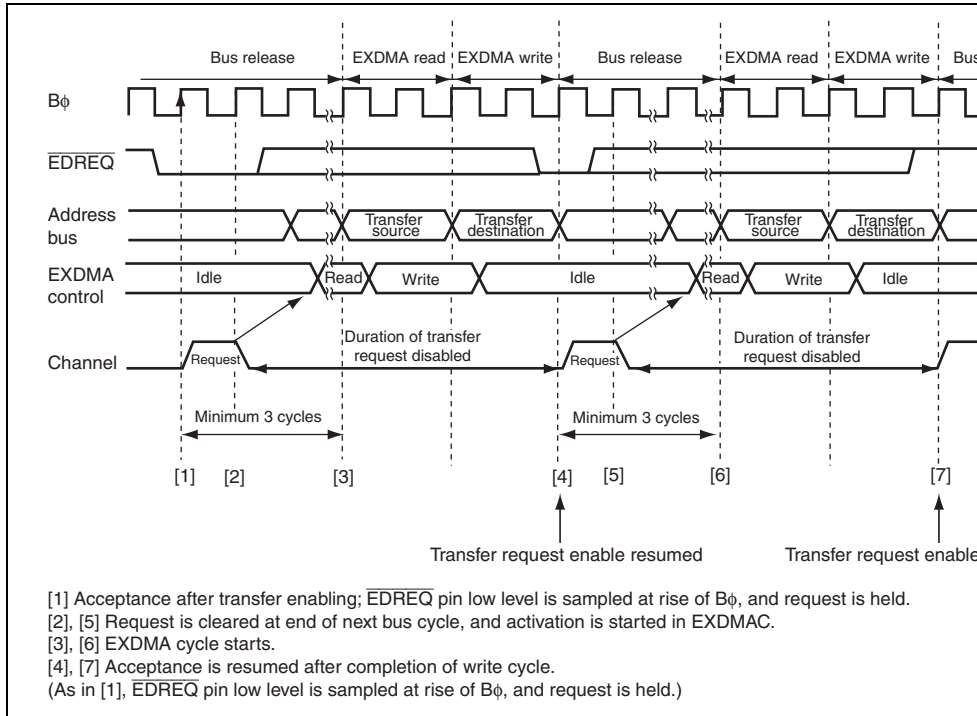
**Figure 11.29 Example of Normal Transfer Mode Transfer Activated by  $\overline{\text{EDREQ}}$  Pin Falling Edge**

performed again. This sequence of operations is repeated until the end of the transfer.

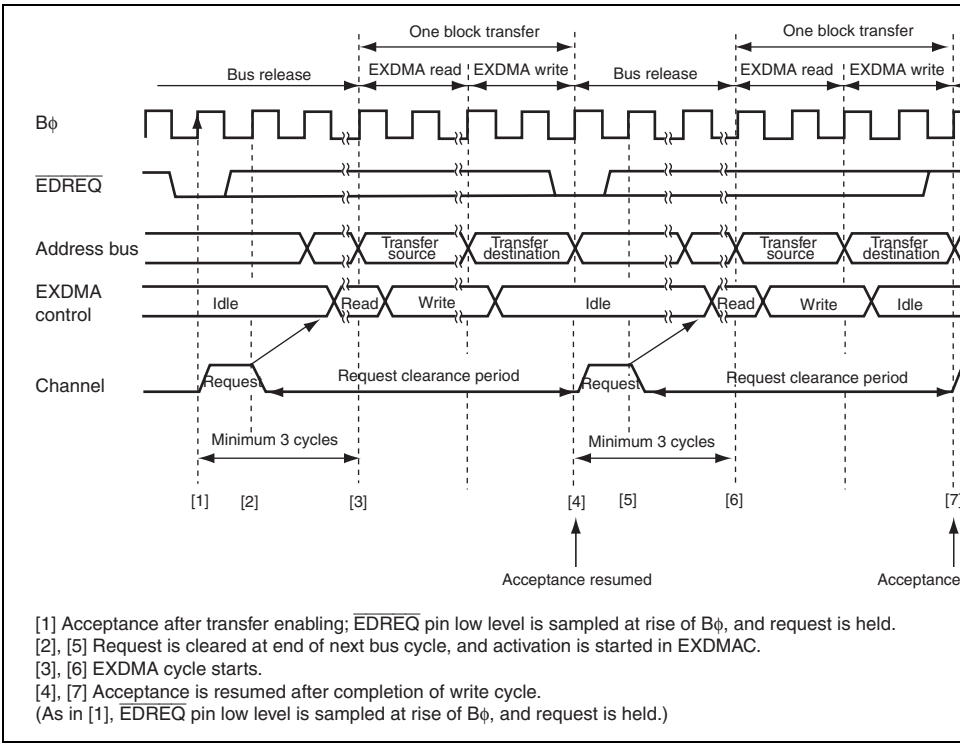


**Figure 11.30 Example of Block Transfer Mode Transfer Activated by EDREQ Pin Falling Edge**

until the end of the transfer.

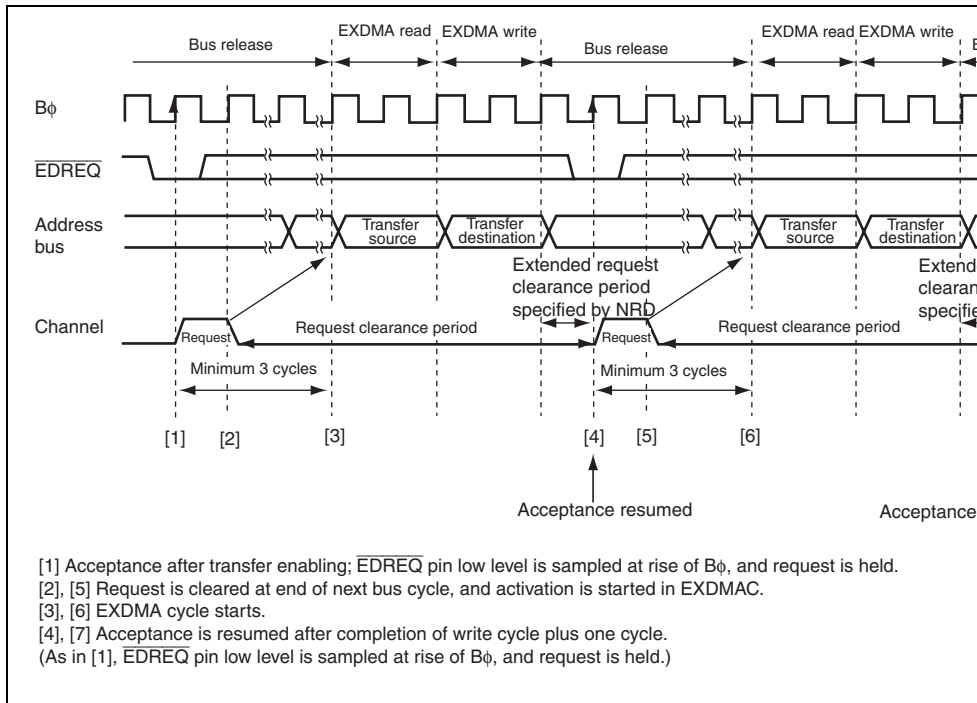


**Figure 11.31 Example of Normal Transfer Mode Transfer Activated by  $\overline{EDREQ}$  Pin Low Level**



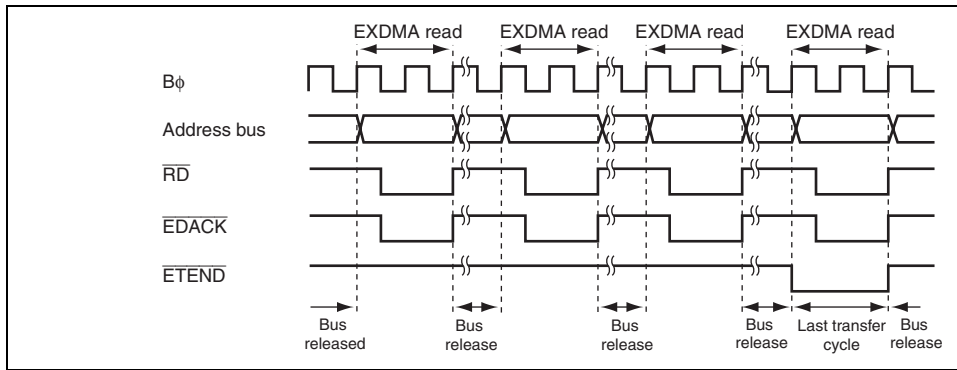
**Figure 11.32 Example of Block Transfer Mode Transfer Activated by  $\overline{EDREQ}$  Pin Low Level**

$\overline{\text{EDREQ}}$  pin is possible, the request is held within the EXDMAC. Then when activation is within the EXDMAC, the request is cleared. After the end of the write cycle, acceptance when one cycle of the request clearance period specified by  $\text{NRD} = 1$  expires and  $\overline{\text{EDREQ}}$  level sampling is performed again. This sequence of operations is repeated until the end of transfer.

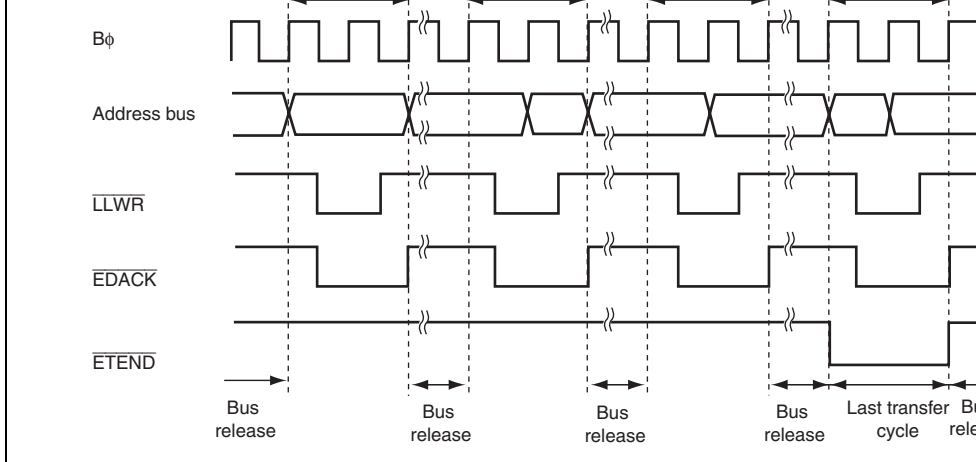


**Figure 11.33 Example of Normal Transfer Mode Transfer Activated by  $\overline{\text{EDREQ}}$  Pin Low Level with  $\text{NRD} = 1$  Specified**



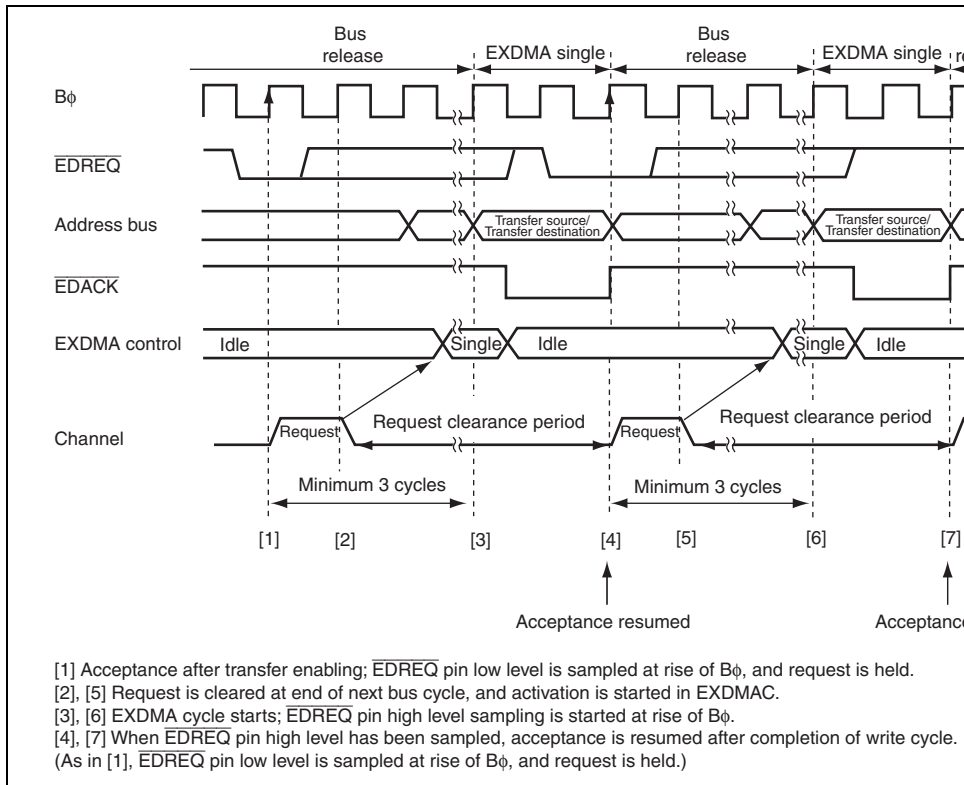


**Figure 11.34 Example of Single Address Mode (Byte Read) Transfer**



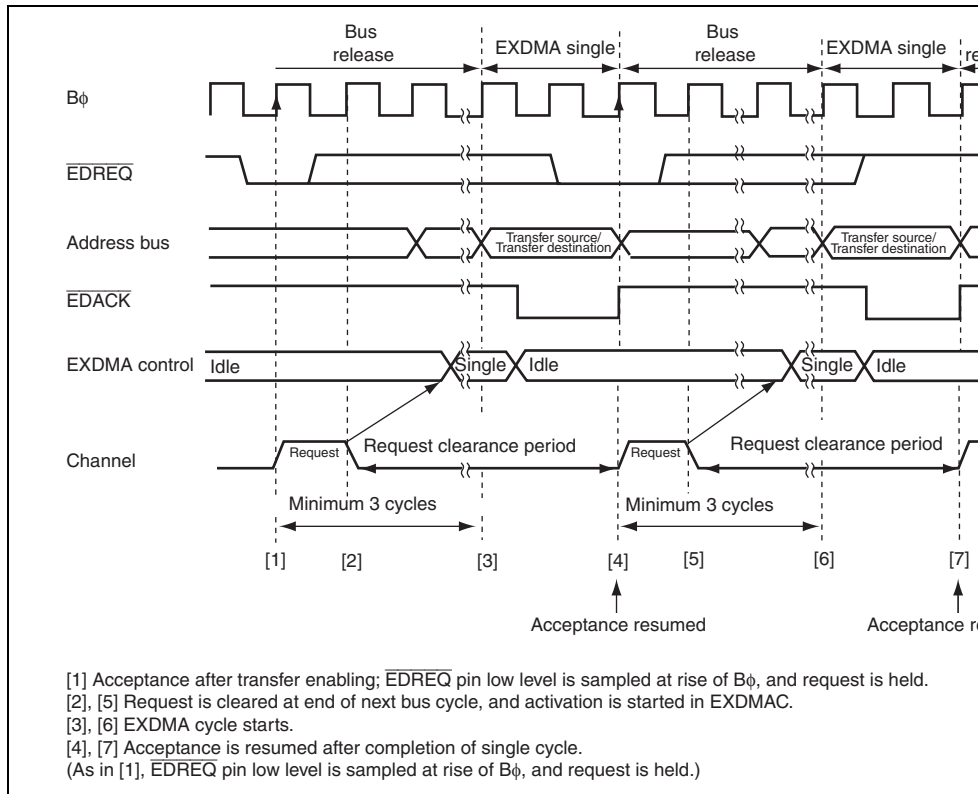
**Figure 11.35 Example of Single Address Mode (Byte Write) Transfer**

is started. If  $\overline{\text{EDREQ}}$  pin high level sampling is completed by the end of the EXDMA cycle, acceptance resumes after the end of the single cycle, and  $\overline{\text{EDREQ}}$  pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.



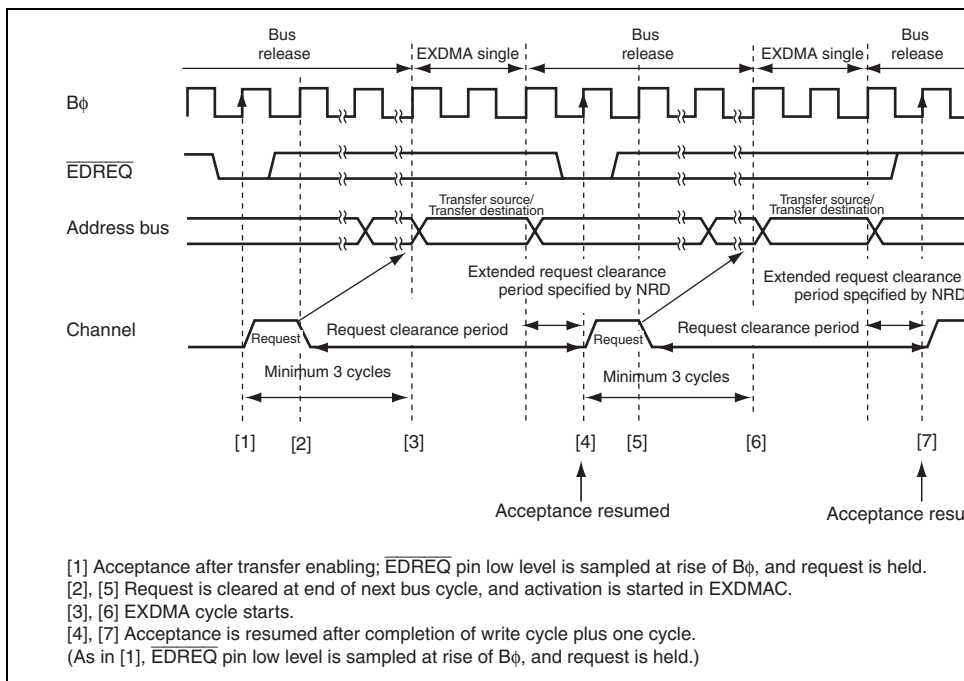
**Figure 11.36 Example of Single Address Mode Transfer Activated by  $\overline{\text{EDREQ}}$  Pin Falling Edge**

and  $\overline{\text{EDREQ}}$  pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.



**Figure 11.37 Example of Single Address Mode Transfer Activated by  $\overline{\text{EDREQ}}$  Pin Low Level**

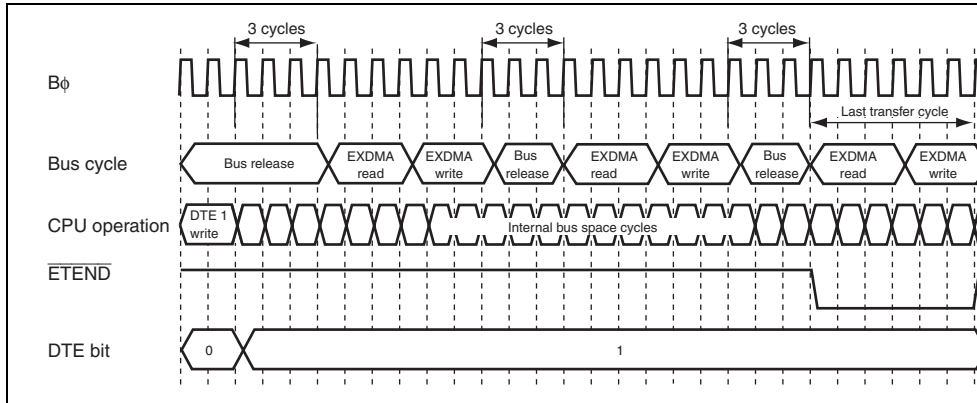
$\overline{\text{EDREQ}}$  pin is possible, the request is held within the EXDMAC. Then when activation within the EXDMAC, the request is cleared. After the end of the single cycle, acceptance when one cycle of the request clearance period specified by  $\text{NRD} = 1$  expires and  $\overline{\text{EDREQ}}$  level sampling is performed again. This sequence of operations is repeated until the end of transfer.



**Figure 11.38 Example of Single Address Mode Transfer Activated by  $\overline{\text{EDREQ}}$  Pin Low Level with  $\text{NRD} = 1$  Specified**

is resumed on completion of the higher-priority channel transfer.

Figures 11.39 and 11.40 show operation timing examples for various conditions.

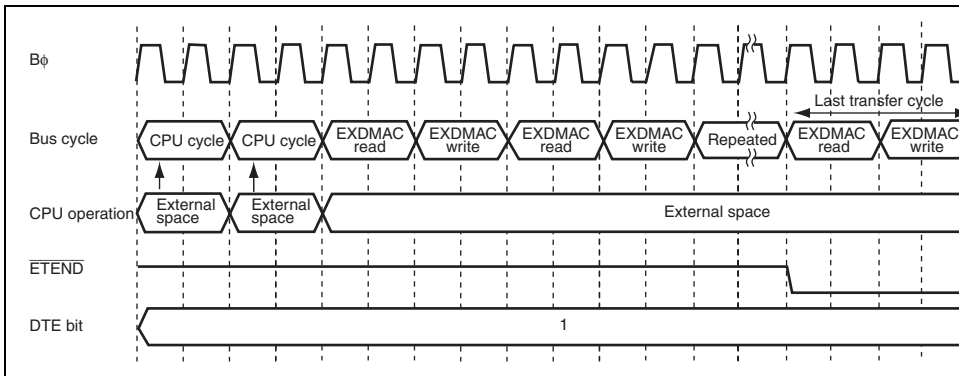


**Figure 11.39 Auto-Request/Normal Transfer Mode/Cycle Steal Mode  
(No Conflict/Dual Address Mode)**

**(2) Auto-Request/Normal Transfer Mode/Burst Mode**

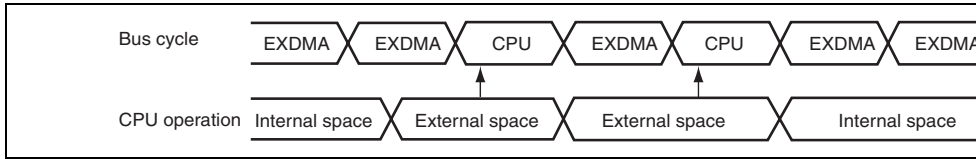
With auto-request (in burst mode), when the DTE bit is set to 1 in EDMDR, an EXDMAC cycle is started a minimum of three cycles later. Once transfer is started, it continues (as long as the transfer end condition is satisfied). Transfer requests for other channels are held off until the end of transfer on the current channel.

Figures 11.41 to 11.43 show operation timing examples for various conditions.



**Figure 11.41 Auto-Request/Normal Transfer Mode/Burst Mode  
(CPU Cycles/Dual Address Mode)**

**Figure 11.42 Auto-Request/Normal Transfer Mode/Burst Mode  
(Conflict with Another Channel/Single Address Mode)**



**Figure 11.43 External Bus Master Cycle Steal Function (Auto-Request/Normal Transfer Mode/Burst Mode with CPU Cycles/Single Address Mode/EBCCS = 1)**

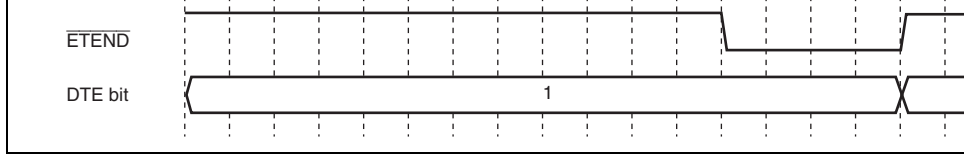
**(3) External Request/Normal Transfer Mode/Cycle Steal Mode**

In external request mode, an EXDMA transfer cycle is started a minimum of three cycles after a transfer request is accepted. The next transfer request is accepted after the end of a one-transfer unit EXDMA cycle. For external bus space CPU cycles, at least one bus cycle is generated during the next EXDMA cycle.

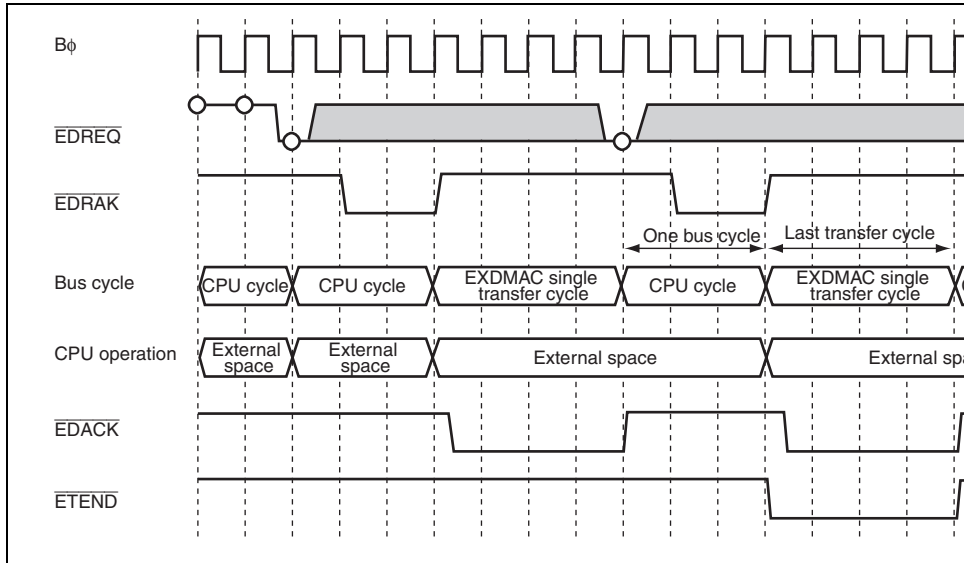
If a transfer request is generated for another channel, an EXDMA cycle for the other channel is generated before the next EXDMA cycle.

The  $\overline{\text{EDREQ}}$  pin sensing timing is different for low level sensing and falling edge sensing. The same applies to transfer request acceptance and transfer start timing.





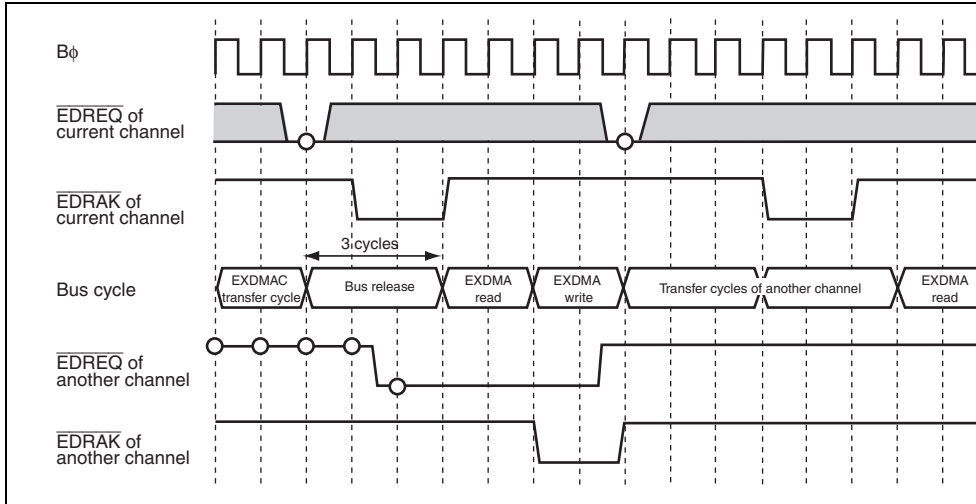
**Figure 11.44 External Request/Normal Transfer Mode/Cycle Steal Mode  
(No Conflict/Dual Address Mode/Low Level Sensing)**



**Figure 11.45 External Request/Normal Transfer Mode/Cycle Steal Mode  
(CPU Cycles/Single Address Mode/Low Level Sensing)**

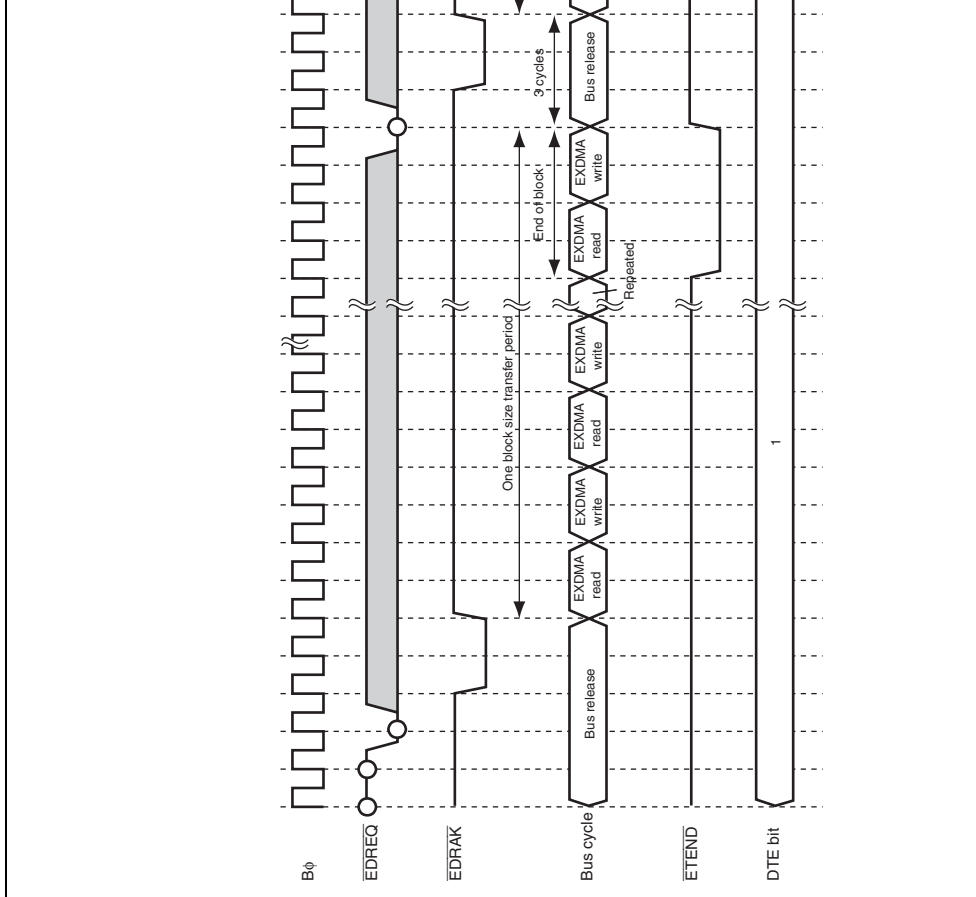


**Figure 11.46 External Request/Normal Transfer Mode/Cycle Steal Mode (Conflict/Single Address Mode/Falling Edge Sensing)**

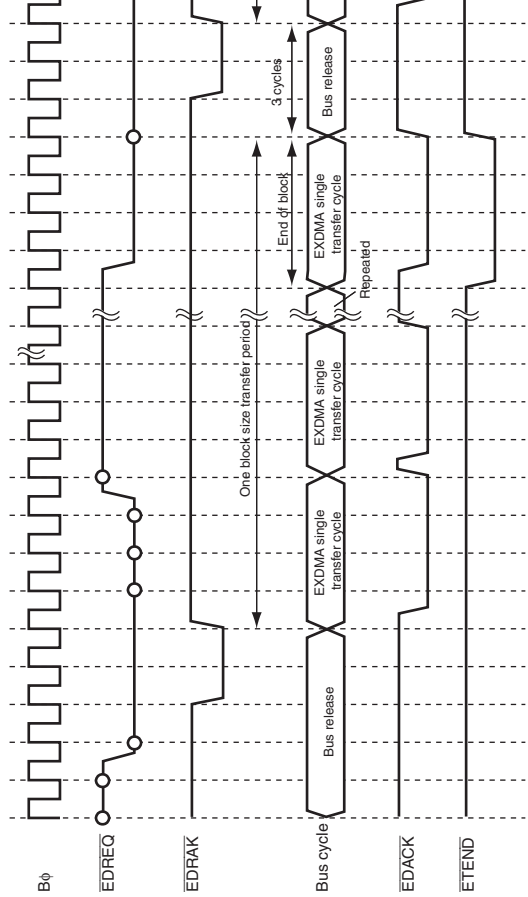


**Figure 11.47 External Request/Normal Transfer Mode/Cycle Steal Mode (Conflict with Another Channel/Dual Address Mode/Low Level Sensing)**

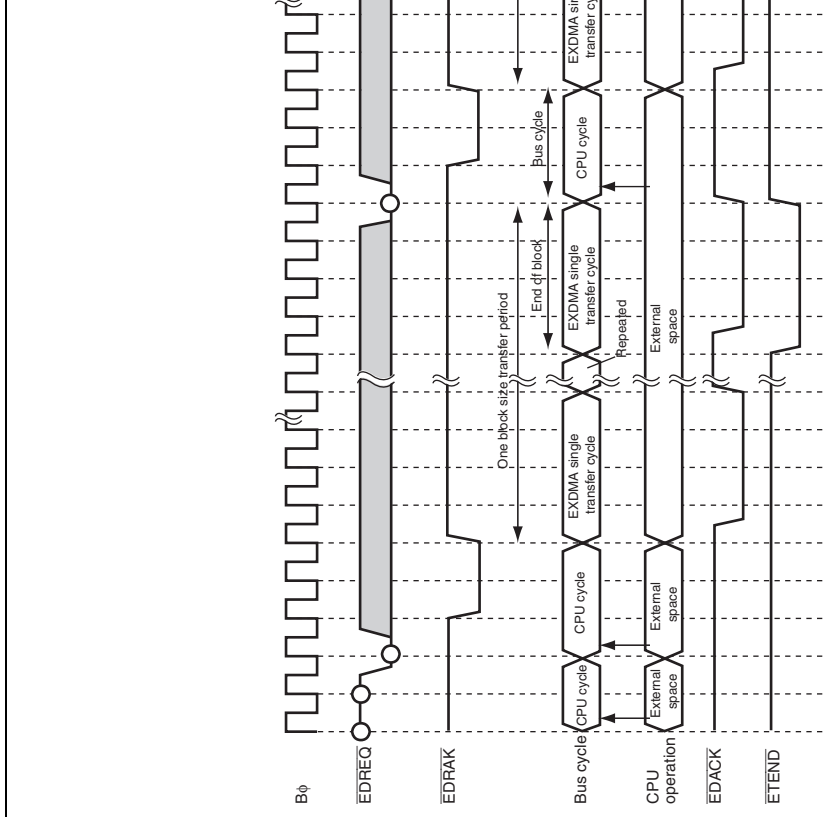
Figures 11.48 to 11.52 show operation timing examples for various conditions.



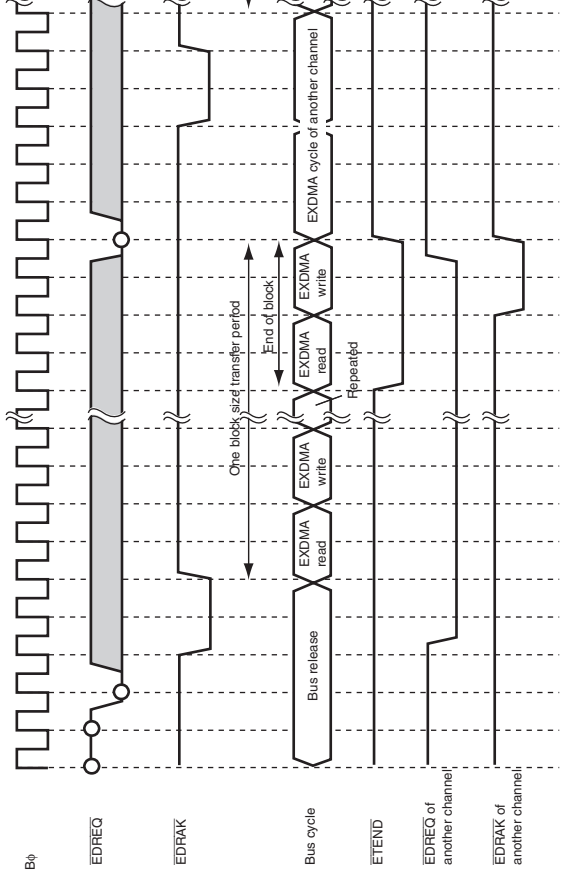
**Figure 11.48 External Request/Block Transfer Mode/Cycle Steal Mode  
(No Conflict/Dual Address Mode/Low Level Sensing)**



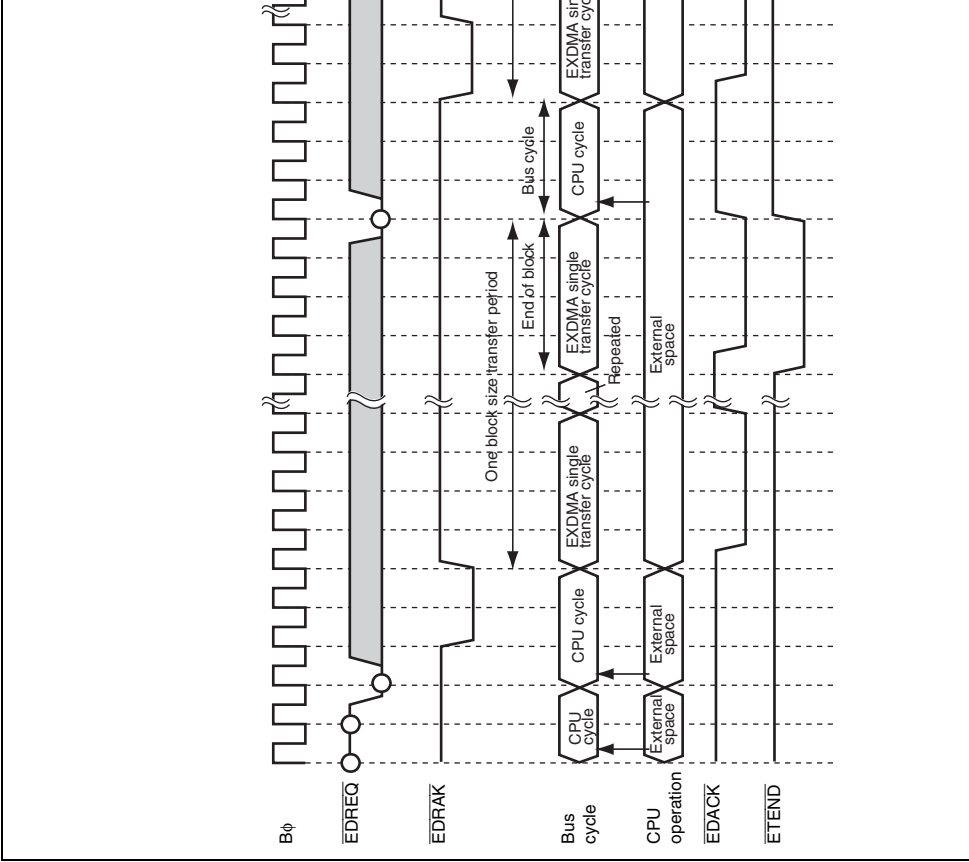
**Figure 11.49 External Request/Block Transfer Mode/Cycle Steal Mode  
(No Conflict/Single Address Mode/Falling Edge Sensing)**



**Figure 11.50 External Request/Block Transfer Mode/Cycle Steal Mode (CPU Cycles/Single Address Mode/Low Level Sensing)**



**Figure 11.51 External Request/Block Transfer Mode/Cycle Steal Mode  
(Conflict with Another Channel/Dual Address Mode/Low Level Sensing)**



**Figure 11.52 External Request/Block Transfer Mode/Cycle Steal Mode  
(CPU Cycles/EBCCS = 1/Single Address Mode/Low Level Sensing)**



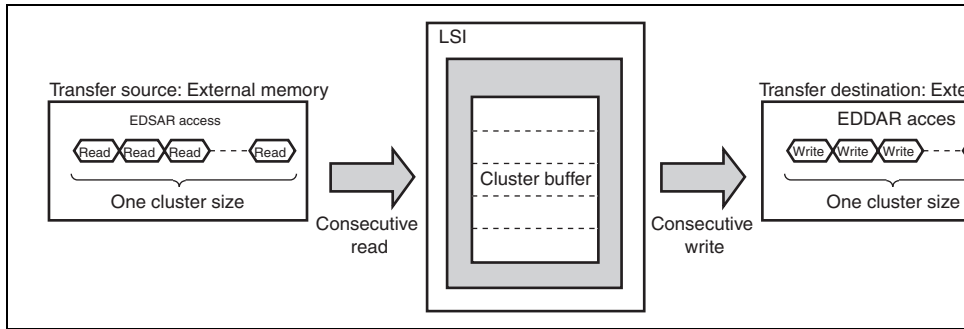
in this mode, both the transfer source and destination addresses are specified for transfer. EXDMAC internal registers. The transfer source address is set in the source address register (EDSAR), and the transfer destination address is set in the destination address register (EDDAR).

The transfer is processed by performing the consecutive read of a cluster-size from the transfer source address and then the consecutive write of that data to the transfer destination address. A data access size of 32 bytes can be specified as a cluster size. When one data access size is specified as a cluster size, block transfer mode (dual address mode) is used.

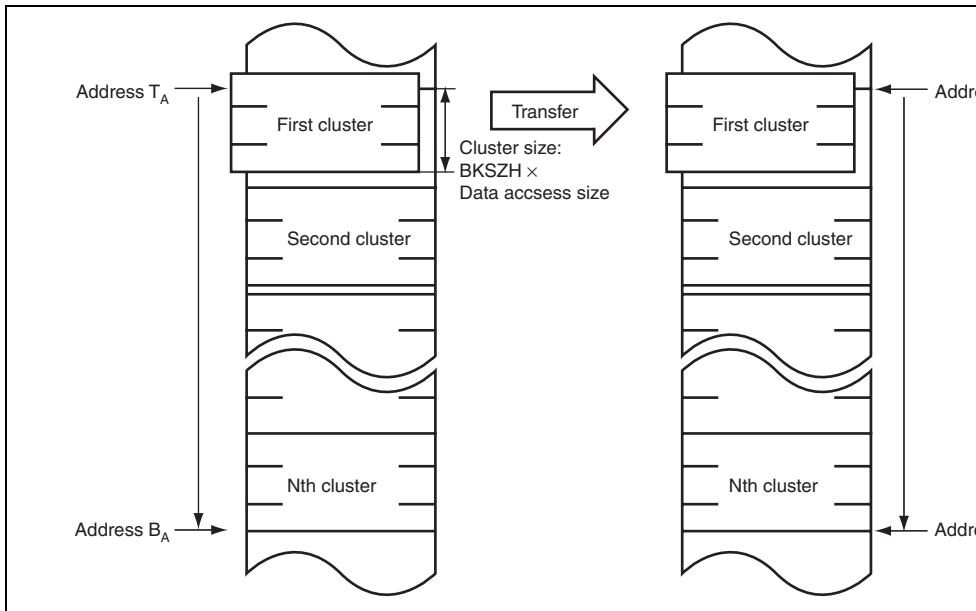
The cycles in a cluster-size transfer are indivisible: another bus cycle (external access by bus master, refresh cycle, or external bus release cycle) does not occur in a cluster-size transfer.

$\overline{\text{ETEND}}$  pin output can be enabled or disabled by means of the ETENDE bit in EDMDR.  $\overline{\text{ETEND}}$  is output for the last write cycle. The  $\overline{\text{EDACK}}$  signal is not output.

Figure 11.53 shows the data flow in the cluster transfer mode (dual address mode), figure 11.54 shows an example of the timing in cluster transfer dual address mode, and figure 11.55 shows the cluster transfer dual address mode operation.



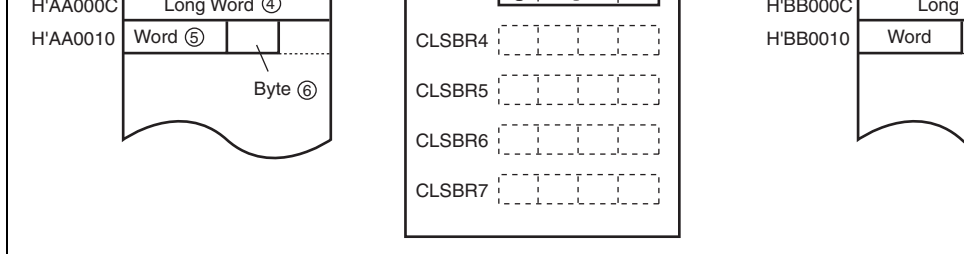
**Figure 11.53 Data Flow in Cluster Transfer Dual Address Mode**



**Figure 11.55 Cluster Transfer Dual Address Mode Operation**

When a word or longword is specified as a data access size but the source or destination address is not at the word or longword boundary, use the appropriate data access size for efficient data transfer.

In an example shown in figure 11.56, a longword-size transfer is performed with 4-longword specified as a cluster size in the cluster transfer dual address mode from the lower two bits to B'10.



**Figure 11.56 Odd Address Transfer**

**(2) Cluster Transfer Read Address Mode (AMS = 1, DIRS = 0)**

In this mode, the transfer source address is specified in the source address register (EDSR). Data is read from the transfer source and transferred to the cluster buffer. In this mode, the  $\overline{\text{ETEND}}$  bit in the mode control register (EDMDR) must be set to 1.

Two data access size to 32 bytes can be specified as a cluster size for the consecutive read operation.

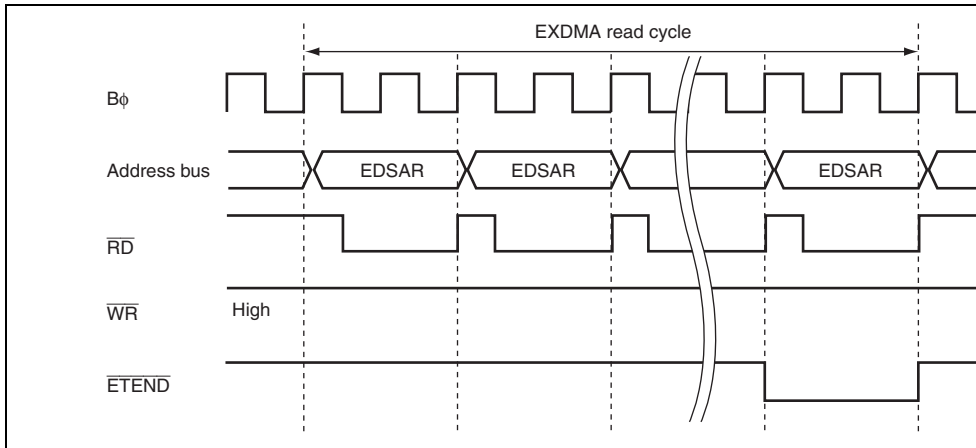
The cycles in a cluster-size transfer are indivisible: another bus cycle (external access by bus master, refresh cycle, or external bus release cycle) does not occur in a cluster-size transfer.

$\overline{\text{ETEND}}$  pin output can be enabled or disabled by means of the ETENDE bit in EDMDR.  $\overline{\text{ETEND}}$  is output for the last read cycle. When an idle cycle is inserted before the last read cycle,  $\overline{\text{ETEND}}$  signal is also output in the idle cycle.

In this mode, the EDACKE bit in EDMDR must be set to 0 to disable the  $\overline{\text{EDACK}}$  pin output.

Figure 11.57 shows the data flow in the cluster transfer read address mode (from the external memory to the cluster buffer), and figure 11.58 shows an example of the timing in cluster transfer read address mode.

**Figure 11.57 Data Flow in Cluster Transfer Read Address Mode (from External Memory to Cluster Buffer)**

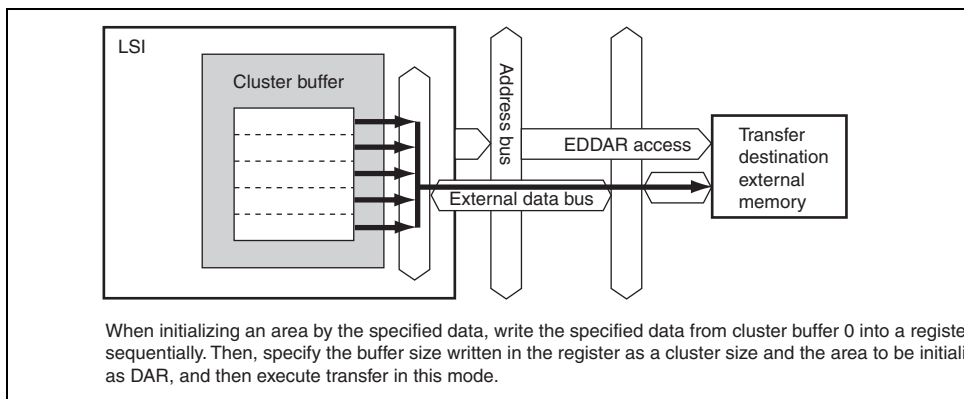


**Figure 11.58 Timing in Cluster Transfer Read Address Mode (from External Memory to Cluster Buffer)**

$\overline{\text{ETEND}}$  pin output can be enabled or disabled by means of the ETENDE bit in EDMDR.  $\overline{\text{ETEND}}$  is output for the last write cycle. When an idle cycle is inserted before the last write cycle,  $\overline{\text{ETEND}}$  signal is also output in the idle cycle.

In this mode, the EDACKE bit in EDMCR must be set to 0 to disable the  $\overline{\text{EDACK}}$  pin output.

Figure 11.59 shows the data flow in the cluster transfer write address mode (from the cluster buffer to the external memory), and figure 11.60 shows an example of the timing in cluster transfer write address mode.



**Figure 11.59 Data Flow in Cluster Transfer Write Address Mode  
(from Cluster Buffer to External Memory)**

**Figure 11.60 Timing in Cluster Transfer Write Address Mode  
(from Cluster Buffer to External Memory)**

### 11.6.2 Setting of Address Update Mode

The cluster transfer mode transfer is restricted by the address update mode function. There are the following four address update methods: increment, decrement, fixed, and offset addition.

When the address increment method is specified and if the specified address is not at the address boundary for the data access size (odd address for a word-size transfer, address beyond the address boundary for a longword-size transfer), the bus cycle is divided for transfer until the address becomes at the address boundary. When the address matches the boundary, transfer is processed in units of data access sizes. At the end of transfer, the bus cycle is divided again to transfer the remaining data in cluster transfer mode.

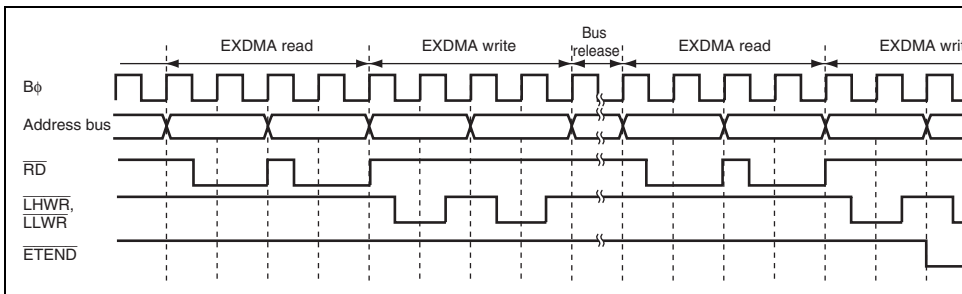
With address decrement, fixed, or offset addition method, specify the address, that matches the address boundary for the data access size, in EDSAR and EDDAR. When specifying the address that is not at the address boundary for the data access size, in EDSAR and EDDAR, fix the lower bit to 0 (lower one bit for a word-size transfer, and lower two bits for a longword-size transfer) in the address register so that the transfer is processed in units of data access sizes. The block transfer mode must be used for transfer of data by dividing the bus cycle according to the address boundary.

When the EDTCR value is smaller than the cluster size, a transfer size error occurs. In this case, when the TSEIE bit in EDMDR is cleared to 0, the cluster transfer mode is switched to the normal transfer mode to process the remaining data. With the decrement, fixed, or offset addition method, transfer is performed without fixing the lower bit to 0.

## (1) Cluster transfer mode

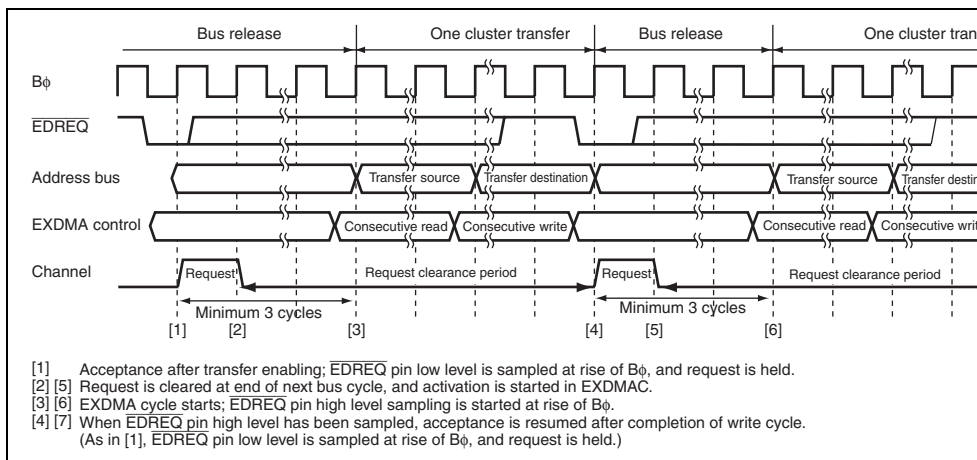
In cluster transfer mode, a cluster-size transfer is processed in response to one transfer request.

In an example shown in figure 11.61, the  $\overline{\text{ETEND}}$  pin output is enabled, and word-size transfer is performed with 4-byte cluster size in cluster transfer mode from the external 16-bit, 2-state access space to the external 16-bit, 2-state access space.



**Figure 11.61 Example of Cluster Transfer Mode Transfer**

is started. If EDREQ pin high level sampling is completed by the end of the last cluster transfer cycle, acceptance resumes after the end of the write cycle, and EDREQ pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.



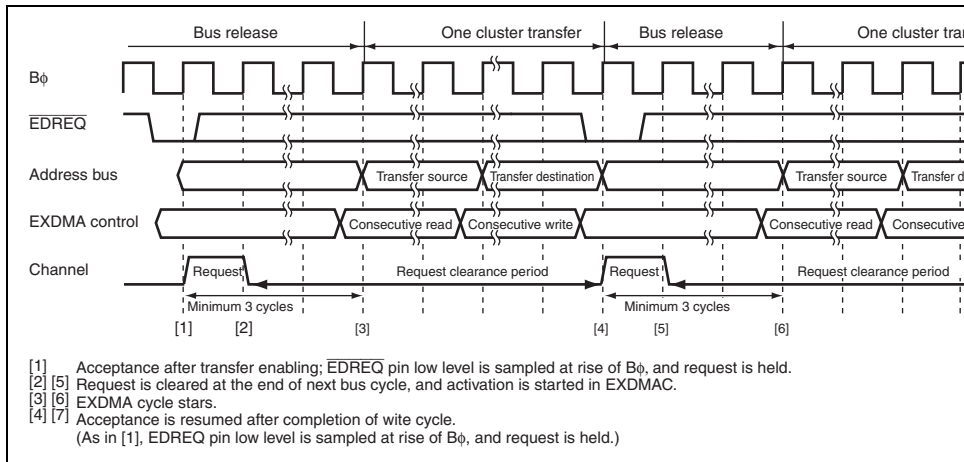
**Figure 11.62 Example of Cluster Transfer Mode Transfer Activated by EDREQ Pin Falling Edge**



resumes and  $\overline{\text{EDREQ}}$  pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.

When NRD bit = 0 in EDMDR, acceptance resumes at the end of the last cluster write cycle and  $\overline{\text{EDREQ}}$  pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.

When NRD bit = 1 in EDMDR, acceptance resumes after one cycle from the end of the last write cycle, and  $\overline{\text{EDREQ}}$  pin low level sampling is performed again. This sequence of operations is repeated until the end of the transfer.

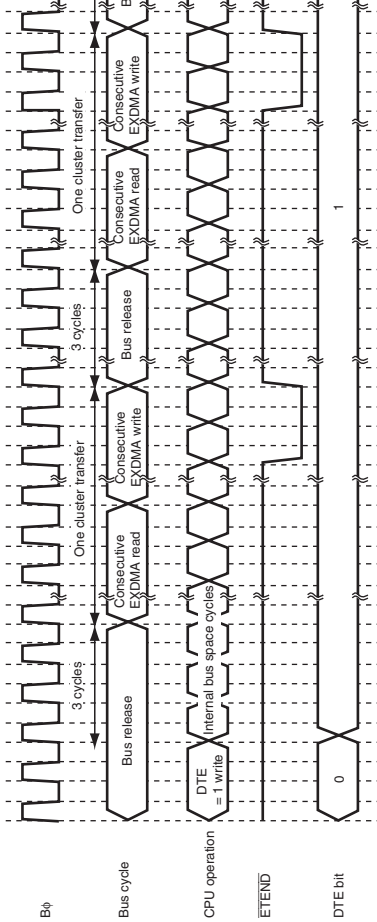


**Figure 11.63 Example of Cluster Transfer Mode Transfer Activated by  $\overline{\text{EDREQ}}$  Pin Low Level**

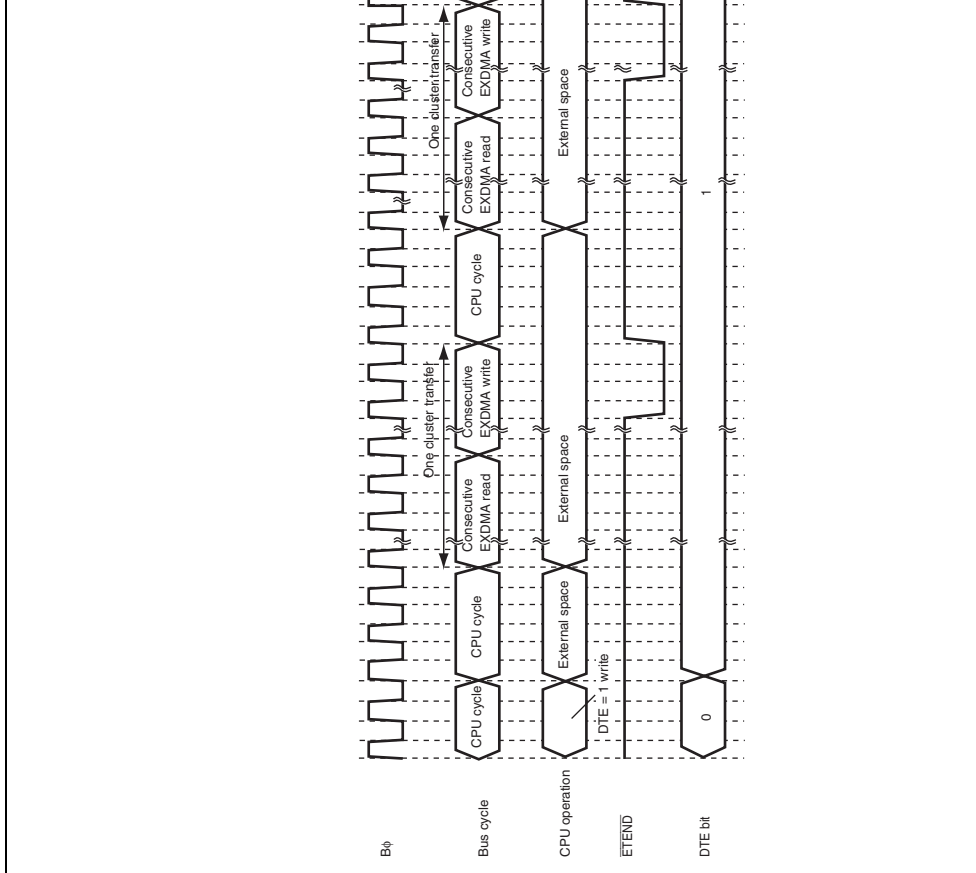
channel is resumed on completion of the higher-priority channel transfer.

The cluster transfer mode (read address mode and write address mode) can not be used with the cluster transfer mode (dual address mode) among more than one channel at the same time. When using the cluster transfer mode (read address mode and write address mode), do not set the transfer mode for another channel.

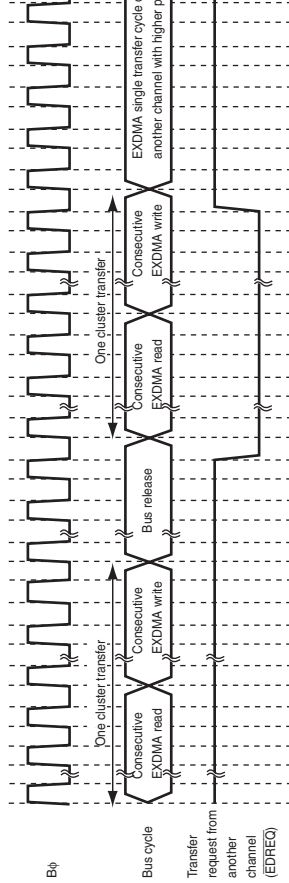
Figures 11.64 to 11.66 show operation timing examples for various conditions.



**Figure 11.64 Auto-Request/Cluster Transfer Mode/Cycle Steal Mode (No Conflict/Dual Address Mode)**



**Figure 11.65 Auto-Request/Cluster Transfer Mode/Cycle Steal Mode (CPU Cycles/Dual Address Mode)**

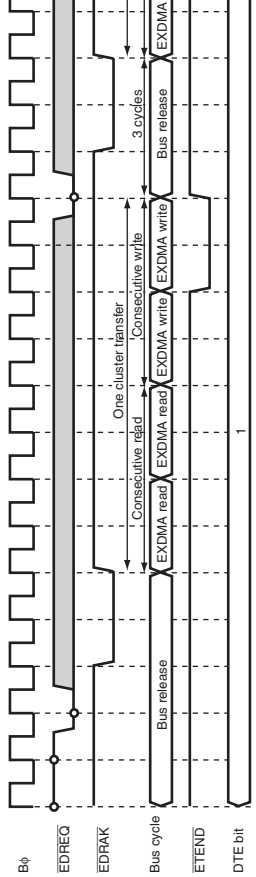


**Figure 11.66 Auto-Request/Cluster Transfer Mode/Cycle Steal Mode  
(Conflict with Another Channel/Dual Address Mode)**

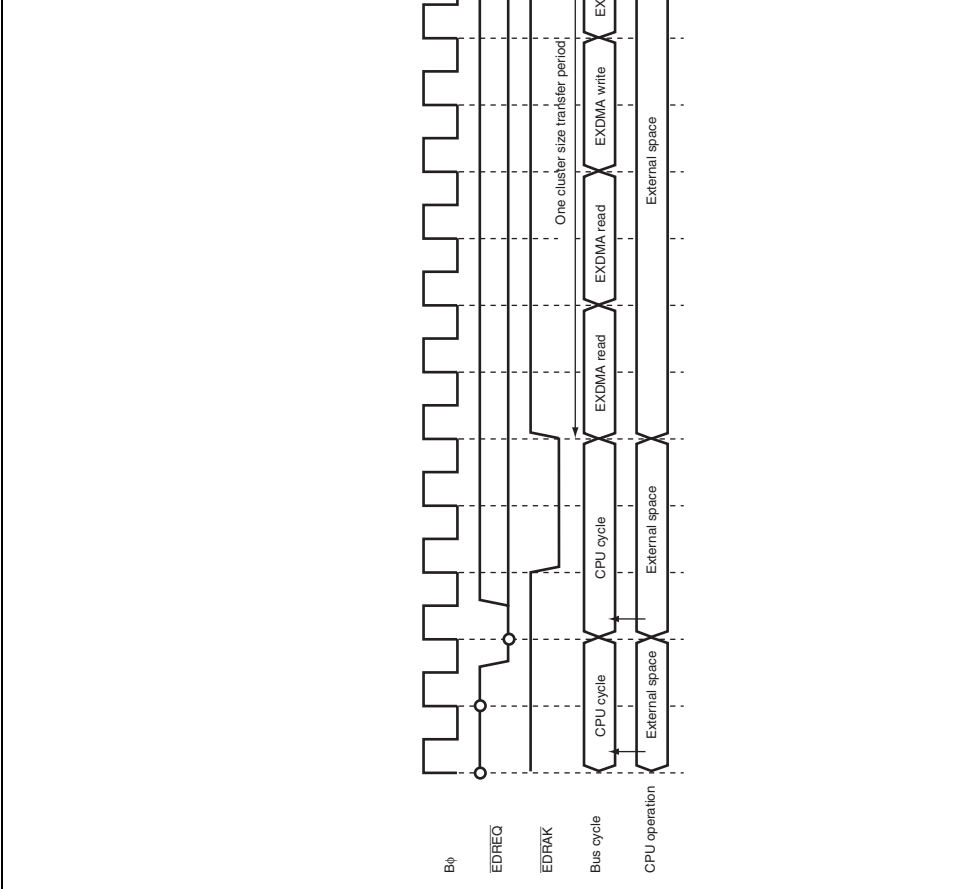
transfer mode for another channel.

The  $\overline{\text{EDREQ}}$  pin sensing timing is different for low level sensing and falling edge sensing. The same applies to transfer request acceptance and transfer start timing.

Figures 11.67 to 11.69 show operation timing examples for various conditions.

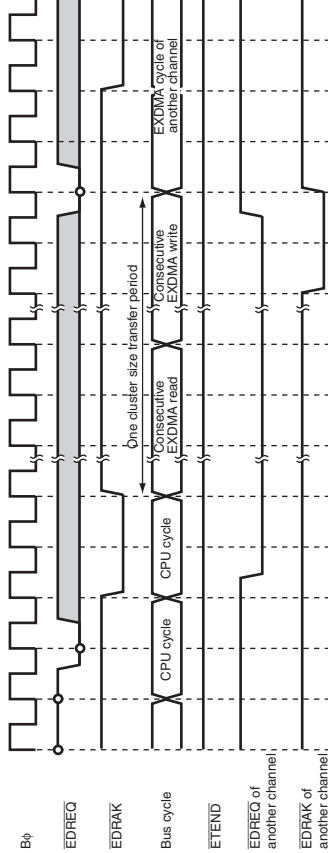


**Figure 11.67 External Request/Cluster Transfer Mode/Cycle Steal Mode  
(No Conflict/Dual Address Mode/Low Level Sensing)**



**Figure 11.68 External Request/Cluster Transfer Mode/Cycle Steal Mode (CPU Cycles/Dual Address Mode/Low Level Sensing)**





**Figure 11.69 External Request/Cluster Transfer Mode/Cycle Steal Mode  
(Conflict with Another Channel/Dual Address Mode/Low Level Sensing)**

generated by the transfer counter. EXDMA transfer does not end if the EDTCR value has since before the start of transfer.

## (2) Transfer End by Transfer Size Error Interrupt

When the following conditions are satisfied while the TSEIE bit in EDMDR is set to 1, a size error occurs and an EXDMA transfer is terminated. At this time, the DTE bit in EDMDR is cleared to 0 and the ESIF bit in EDMDR 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 EDTCR 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 EDTCR value less than the block size.
- In cluster transfer mode, when the next transfer is requested while a transfer is disabled due to the EDTCR value less than the cluster size.

When the TSEIE bit in EDMDR is cleared to 0, data is transferred until the EDTCR value reaches 0. A transfer size error is not generated. Operation in each transfer mode is described below.

- In normal transfer mode and repeat mode, when the EDTCR value is less than the data access size, data is transferred in bytes.
- In block transfer mode, when the EDTCR value is less than the block size, the specified amount of data in EDTCR is transferred instead of transferring the block size of data. When the EDTCR value is less than the data access size, data is transferred in bytes.
- In cluster transfer mode, when the EDTCR value is less than the cluster size, the specified amount of data in EDTCR is transferred instead of transferring the cluster size of data. When the EDTCR value is less than the data access size, data is transferred in bytes.

#### **(4) Transfer End by Extended Repeat Area Overflow Interrupt**

If an address overflows the extended repeat area when an extended repeat area specific interrupt has been made and the SARIE or DARIE bit in EDACR is set to 1, an extended repeat area overflow interrupt is requested. The interrupt request terminates EXDMA transfer, the DTE bit in EDMDR is cleared to 0, and the ESIF bit in EDMDR is set to 1 at the same time.

In dual address mode, if an extended repeat area overflow interrupt is requested during a write cycle, the following write cycle processing is still executed.

In block transfer mode, if an extended repeat area overflow interrupt is requested during a block transfer, transfer continues to the end of the block. Transfer end by means of an extended repeat area overflow interrupt occurs between block-size transfers.

In cluster transfer mode, if an extended repeat area overflow interrupt is requested during a cluster transfer, transfer continues to the end of the cluster. Transfer end by means of an extended repeat area overflow interrupt occurs between cluster-size transfers.

#### **(5) Transfer End by 0-Write to DTE Bit in EDMDR**

When 0 is written to the DTE bit in EDMDR by the CPU, etc., transfer ends after completion of the EXDMA cycle in which transfer is in progress or a transfer request was accepted.

In block transfer mode, EXDMA transfer ends after completion of one-block-size transfer in progress.

In cluster transfer mode, EXDMA transfer ends after completion of one-cluster-size transfer in progress.

units of transfers.

In single address mode, EXDMA transfer ends at the end of the EXDMA transfer bus cycle units of transfers.

**(b) Block transfer mode**

A block size EXDMA transfer is aborted. A block size transfer is not correctly executed, matching between the actual transfer and the transfer request is not guaranteed.

In dual address mode, a write cycle corresponding to a read cycle is executed as well as in normal transfer mode.

**(c) Cluster transfer mode**

A cluster size EXDMA transfer is aborted. If transfer is aborted in a read cycle, the read cycle is guaranteed. If transfer is aborted in a write cycle, the data not transferred is not guaranteed. Matching between the transfer counter and the address register is not guaranteed since the processing cannot be controlled.

**(7) Transfer End by Address Error**

If an address error occurs, the EXDMAC clears the DTE bit to 0 in all channels, and sets the ERRF bit in EDMDR\_0 to 1. An address error during EXDMA transfer forcibly terminates the transfer. To perform EXDMA transfer after an address error occurs, clear the ERRF bit to 0 and then set the DTE bit to 1 in each channel.

The transfer end timing after address error detection is the same as for the one when an NMI interrupt occurs.

channel register (CPUPCR). For details, see section 7.7, CPU Priority Control Function, DMAC and EXDMAC.

The EXDMAC priority level can be set independently for each channel by the EDMAPE0 to EDMAPE7 bits in EDMDR.

The CPU priority level, which corresponds to the priority level of exception handling, can be set by updating the values of the CPUP2 to CPUP0 bits in CPUPCR with the interrupt mask register (IMR) values.

When the CPUPCE bit in CPUPCR is set to 1 to enable the CPU priority level control function, if the EXDMAC priority level is lower than the CPU priority level, the transfer request of the corresponding channel is masked and the channel activation is disabled. When the priority level of another channel is the same or higher than the CPU priority level, the transfer request for that channel is accepted and transfer is enabled regardless of the priority levels of channels.

The CPU priority level control function holds pending the transfer source, which masks the transfer request. When the CPU priority level becomes lower than the channel priority level, updating one of them, the transfer request is accepted and transfer starts. The transfer request pending is cleared by writing 0 to the DTE bit.

When the CPUPCE bit is cleared to 0, the lowest CPU priority level is assumed.

These consecutive EXDMA read and write cycles are indivisible: refresh cycle, external release cycle, or external space access cycle by internal bus master (CPU, DTC, DMAC) occur between a read cycle and a write cycle.

In cluster transfer mode, the transfer cycle in one cluster is indivisible.

In block transfer mode and auto-request burst mode, the EXDMA transfer bus cycles continue during this period, the bus priority level of the internal bus master is lower than the EXDMAC so that external space access is held pending (when EBCCS = 0 in the bus control register 2 (BCR2)).

When switching to another channel, or in the auto-request cycle steal mode, the EXDMA transfer bus cycles and internal bus master cycles are alternatively executed. When the internal bus master is not issuing an external space access cycle, the EXDMA transfer bus cycles are continuously executed in the allowable range.

When the EBCCS bit in BCR2 is set to 1 to enable the arbitration function between the EXDMA and the internal bus master, the bus mastership is released, except for indivisible bus cycles transferred between the EXDMAC and the internal bus master alternatively. For details, see section 9, Bus Controller (BSC).

EXDMTEND1	Transfer end indicated by channel 1 transfer counter
EXDMTEND2	Transfer end indicated by channel 2 transfer counter
EXDMTEND3	Transfer end indicated by channel 3 transfer counter
EXDMEEND0	Channel 0 transfer size error Channel 0 repeat size end Channel 0 source address extended repeat area overflow Channel 0 destination address extended repeat area overflow
EXDMEEND1	Channel 1 transfer size error Channel 1 repeat size end Channel 1 source address extended repeat area overflow Channel 1 destination address extended repeat area overflow
EXDMEEND2	Channel 2 transfer size error Channel 2 repeat size end Channel 2 source address extended repeat area overflow Channel 2 destination address extended repeat area overflow
EXDMEEND3	Channel 3 transfer size error Channel 3 repeat size end Channel 3 source address extended repeat area overflow Channel 3 destination address extended repeat area overflow

Interrupt source can be enabled or disabled by setting the DTIE and ESIE bits in EDMDR relevant channels. The DTIE bit can be combined with the DTIF bit in EDMDR to generate an EXDMTEND interrupt. The ESIE bit can be combined with the ESIF bit in EDMDR to generate an EXDMEEND interrupt. Interrupt sources in EXDMEEND are not identified as common interrupt sources. The interrupt priority order among channels is determined by the interrupt controller. The interrupt priority order is shown in table 11.7. For details see section 7, Interrupt Controller.

corresponding interrupt enable bit is set to 1, the condition for that interrupt is satisfied, and the ESIF bit in EDMDR is set to 1.

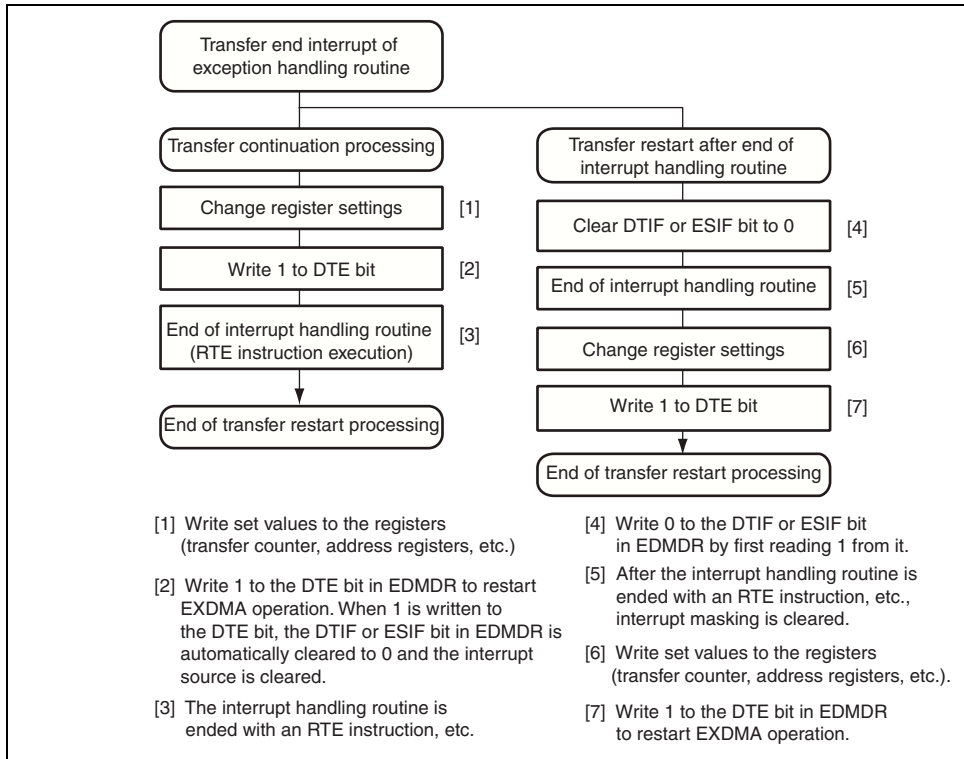
The transfer size error interrupt occurs when the EDTCR value is smaller than the data access size and a data-access-size transfer for one request cannot be performed for a transfer request. In block transfer mode, the block size is compared to the EDTCR value to determine a transfer size error. In cluster transfer mode, the cluster size is compared to the EDTCR value to determine a transfer size error.

The repeat size end interrupt occurs when the next transfer request is generated after the end of a repeat size transfer in repeat transfer mode. When the repeat area is not set in the address register, the transfer can be aborted periodically based on the set repeat size value. If the transfer end interrupt by the transfer counter occurs at the same time, the ESIF bit is set to 1.

The source/destination address extended repeat area overflow interrupt occurs when the address register overflows the specified extended repeat area. If the transfer end interrupt by the transfer counter occurs at the same time, the ESIF bit is set to 1.

Figure 11.70 shows the block diagram of various interrupts and their interrupt flags. The transfer end interrupt can be cleared either by clearing the DTIF or ESIF bit to 0 in EDMDR with an interrupt handling routine, or by re-setting the address registers and then setting the DTE bit in EDMDR to perform transfer continuation processing. An example of the procedure for clearing the transfer end interrupt and restarting transfer is shown in figure 11.71.



**Figure 11.70 Interrupts and Interrupt Sources****Figure 11.71 Procedure for Clearing Transfer End Interrupt and Restarting T**

EXDMAC enters the module stop state. However, 1 cannot be written to the MSTPA14 bit when any of the EXDMAC's channels is enabled for transfer, or when an interrupt is requested. Before setting the MSTPA14 bit, first clear the DTE bit in EDMDR to 0, then the DTIF or DTIE bit in EDMDR to 0.

When the EXDMAC clock stops, EXDMAC registers can no longer be accessed. The following EXDMAC register settings remain valid in the module stop state, and so should be disabled, if necessary, before making the module stop transition.

- ETENDE = 1 in EDMDR ( $\overline{\text{ETEND}}$  pin enable)
- EDRAKE = 1 in EDMDR ( $\overline{\text{EDRAK}}$  pin enable)
- EDACKE = 1 in EDMDR ( $\overline{\text{EDACK}}$  pin enable)

### 3. $\overline{\text{EDREQ}}$ Pin Falling Edge Activation

Falling edge sensing on the  $\overline{\text{EDREQ}}$  pin is performed in synchronization with EXDMAC internal operations, as indicated below.

1. Activation request standby state: Waits for low level sensing on  $\overline{\text{EDREQ}}$  pin, then [2].
2. Transfer standby state: Waits for EXDMAC data transfer to become possible, then [3].
3. Activation request disabled state: Waits for high level sensing on  $\overline{\text{EDREQ}}$  pin, then [1].

After EXDMAC transfer is enabled, the EXDMAC goes to state [1], so low level sensing is used for the initial activation after transfer is enabled.

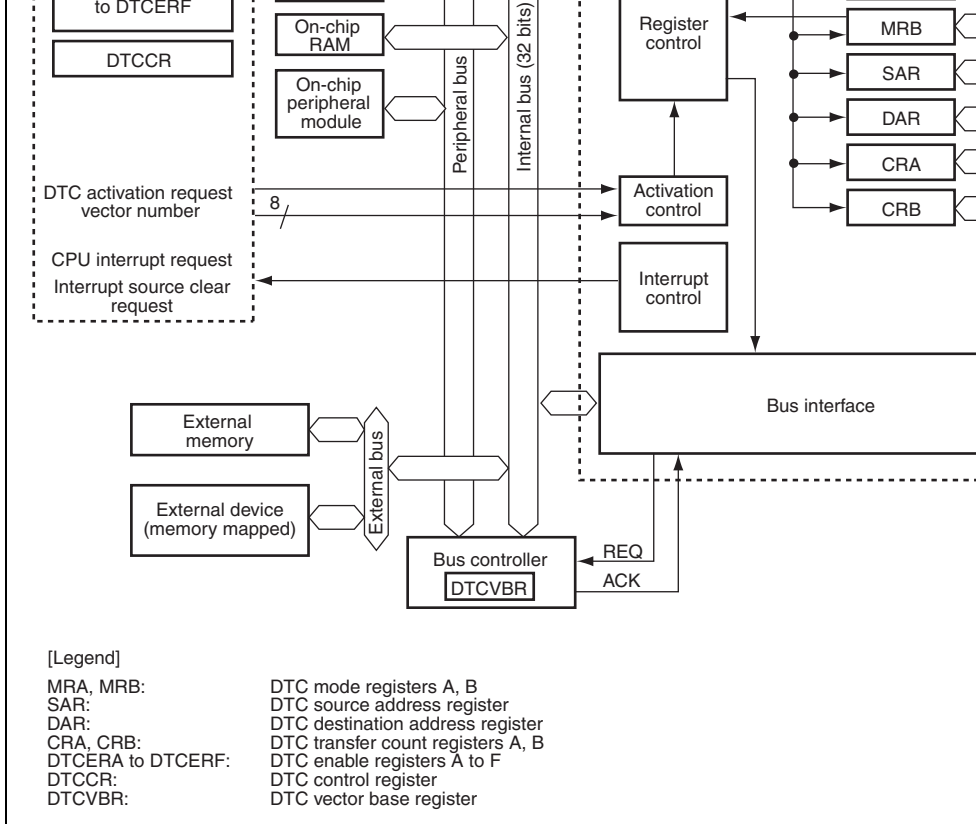
When the transfer between the transfer source/destination and the cluster buffer completes, another cluster transfer, the transferred data in the cluster buffer may be overwritten by the channel cluster transfer. Therefore, in the cluster transfer mode (single address mode), do not set the cluster transfer mode for any other channels.

#### 6. Cluster Transfer Mode and Endian

In cluster transfer mode, only a transfer to the areas in the big endian format is supported. When cluster transfer mode is specified, do not specify the areas in the little endian mode (EDSAR and EDDAR). For details on the endian, see section 9, Bus Controller (BSC).



- Three transfer modes
  - Normal/repeat/block transfer modes selectable
  - Transfer source and destination addresses can be selected from increment/decrement
- 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 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 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 12.1 Block Diagram of DTC**

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 it to the CPU. After the data transfer, it writes a set of updated transfer information back to the data area.

- DTC enable registers A to F (DTCERA to DTCERF)
- DTC control register (DTCCR)
- DTC vector base register (DTCVBR)

Bit	Bit Name	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	—	Undefined	—	Reserved The write value should always be 0.

[Legend]

x: Don't care



Bit	Bit Name	Value	R/W	Description
7	CHNE	Undefined	—	<p>DTC Chain Transfer Enable</p> <p>Specifies the chain transfer. For details, see section 12.5.7, 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 condition for chain transfer is a chain transfer, the completion check of the specified transfer count is not performed and the transfer 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 not generated when the specified number of data transfers 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>

---

1, 0 — Undefined — Reserved

The write value should always be 0.

---

[Legend]

x: Don't care

### 12.2.3 DTC Source Address Register (SAR)

SAR is a 32-bit register that designates the source address of data to be transferred by the

In full address mode, 32 bits of SAR are valid. In short address mode, the lower 24 bits of SAR are 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 12.5.1, Bus Cycle Division.

SAR cannot be accessed directly from the CPU.

into multiple cycles to transfer data. For details, see section 12.5.1, Bus Cycle Division.

DAR cannot be accessed directly from the CPU.

### 12.2.5 DTC Transfer Count Register A (CRA)

CRA is a 16-bit register that designates the number of times data is to be transferred by

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 DTCE<sub>n</sub> (n = 15 to 0) corresponding 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 when CRA = H'0000.

In repeat transfer mode, CRA is divided into two parts: the upper eight bits (CRAH) and 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 lower eight bits (CRAL). CRAH holds the block size while CRAL functions as an 8-bit block-transfer 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.

### 12.2.7 DTC enable registers A to F (DTCERA to DTCERF)

DTCER, which is comprised of eight registers, DTCERA to DTCERF, is a register that stores DTC activation interrupt sources. The correspondence between interrupt sources and DTCER is shown in table 12.1. Use bit manipulation instructions such as BSET and BCLR to read or write the DTCE bit. If all interrupts are masked, multiple activation sources can be set at one time (from 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

7	DTCE7	0	R/W
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

These bits are not cleared when the DISEL bit is set. If the specified number of transfers have not ended

### 12.2.8 DTC Control Register (DTCCR)

DTCCR specifies transfer information read skip.

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

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 always be 0.

0: Transfer read skip is not performed.

1: Transfer read skip is performed when the two 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 does not occur when CRAL is 0. If this bit is set to 1, 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
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>

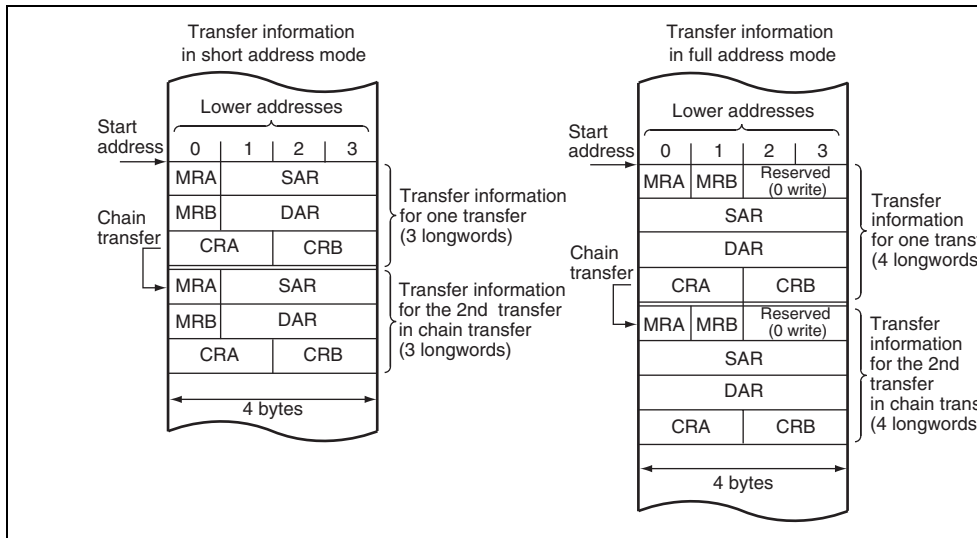
Note: \* Only 0 can be written to clear this flag.

Bit Name															
Initial Value	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	R	R	R	R	R	R	R	R	R

### 12.3 Activation Sources

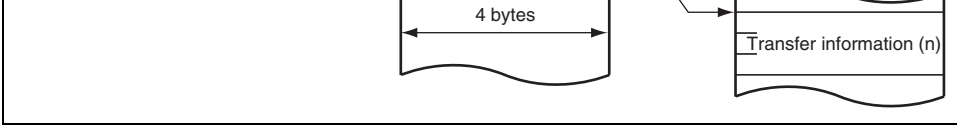
The DTC is activated by an interrupt request. The interrupt source is selected by DTCEB. The activation source can be selected by setting the corresponding bit in DTCER; the CPU interrupt source can be selected by clearing the corresponding bit in DTCER. At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the activation source interrupt corresponding DTCER bit is cleared.

activation source, and then reads the transfer information from the start address. Figure 12.1 shows the correspondences between the DTC vector address and transfer information.



**Figure 12.2 Transfer Information on Data Area**





**Figure 12.3 Correspondence between DTC Vector Address and Transfer Information**

	IRQ5	69	H'514	DTCEA10
	IRQ6	70	H'518	DTCEA9
	IRQ7	71	H'51C	DTCEA8
	IRQ8	72	H'520	DTCEA7
	IRQ9	73	H'524	DTCEA6
	IRQ10	74	H'528	DTCEA5
	IRQ11	75	H'52C	DTCEA4
A/D_0	AD10 (A/D_0 conversion end)	86	H'558	DTCEB15
TPU_0	TGI0A	88	H'560	DTCEB13
	TGI0B	89	H'564	DTCEB12
	TGI0C	90	H'568	DTCEB11
	TGI0D	91	H'56C	DTCEB10
TPU_1	TGI1A	93	H'574	DTCEB9
	TGI1B	94	H'578	DTCEB8
TPU_2	TGI2A	97	H'584	DTCEB7
	TGI2B	98	H'588	DTCEB6

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
	DMTEND2	130	H'608	DTCEC3
	DMTEND3	131	H'60C	DTCEC2
EXDMAC	EXDMTEND0	132	H'610	DTCEC1
	EXDMTEND1	133	H'614	DTCEC0
	EXDMTEND2	134	H'618	DTCEC15
	EXDMTEND3	135	H'61C	DTCEC14
DMAC	DMEEND0	136	H'620	DTCED13
	DMEEND1	137	H'624	DTCED12
	DMEEND2	138	H'628	DTCED11
	DMEEND3	139	H'62C	DTCED10
EXDMAC	EXDMEEND0	140	H'630	DTCECD9
	EXDMEEND1	141	H'634	DTCECD8
	EXDMEEND2	142	H'638	DTCED7
	EXDMEEND3	143	H'63C	DTCED6

TPU_6	TGI6A	164	H'690	DTCEE11
	TGI6B	165	H'694	DTCEE10
	TGI6C	166	H'698	DTCEE9
	TGI6D	167	H'69C	DTCEE8
TPU_7	TGI7A	169	H'6A4	DTCEE7
	TGI7B	170	H'6A8	DTCEE6
TPU_8	TGI8A	173	H'6B4	DTCEE5
	TGI8B	174	H'6B8	DTCEE4
TPU_9	TGI9A	177	H'6C4	DTCEE3
	TGI9B	178	H'6C8	DTCEE2
	TGI9C	179	H'6CC	DTCEE1
	TGI9D	180	H'6D0	DTCEE0
TPU_10	TGI10A	182	H'6D8	DTCEF15
	TGI10B	183	H'6DC	DTCEF14
	TGI10V	186	H'6E8	DTCEF11
TPU_11	TGI11A	188	H'6F0	DTCEF10
	TGI11B	189	H'6F4	DTCEF9

Note: \* The DTCE bits with no corresponding interrupt are reserved, and the write value always be 0. To leave software standby mode or all-module-clock-stop mode without interrupt, write 0 to the corresponding DTCE bit.

Table 12.2 shows the DTC transfer modes.

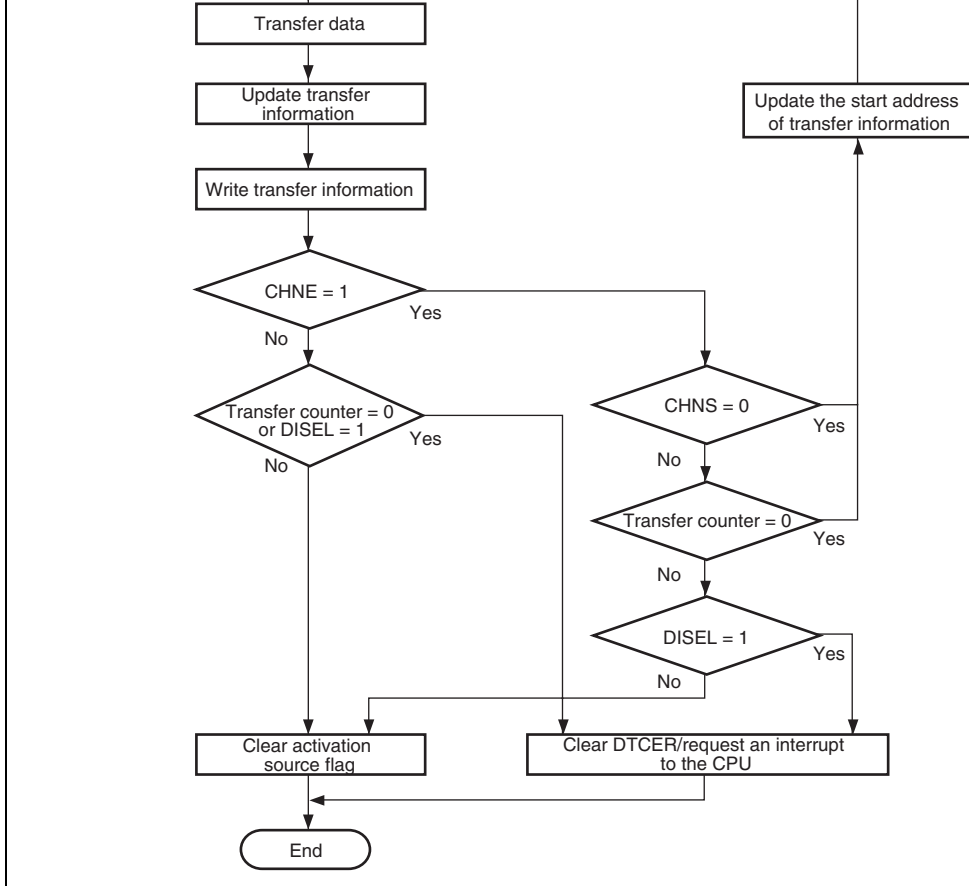
**Table 12.2 DTC Transfer Modes**

<b>Transfer Mode</b>	<b>Size of Data Transferred at One Transfer Request</b>	<b>Memory Address Increment or Decrement</b>	<b>T</b>	<b>C</b>
Normal	1 byte/word/longword	Incremented/decremented by 1, 2, or 4, or fixed		
Repeat* <sup>1</sup>	1 byte/word/longword	Incremented/decremented by 1, 2, or 4, or fixed		
Block* <sup>2</sup>	Block size specified by CRAH (1 to 256 bytes/words/longwords)	Incremented/decremented by 1, 2, or 4, or fixed		

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 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 perform a chain transfer performed only when the transfer counter value is 0.

Figure 12.4 shows a flowchart of DTC operation, and table 12.3 summarizes the chain transfer conditions (combinations for performing the second and third transfers are omitted).



**Figure 12.4 Flowchart of DTC Operation**

1	1	0	Not 0	—	—	—	—	Ends at 1st tra
1	1	—	0*2	0	—	0	Not 0	Ends at 2nd tr
				0	—	0	0*2	Ends at 2nd tr
				0	—	1		Interrupt requ
1	1	1	Not 0	—	—	—	—	Ends at 1st tra
								Interrupt requ

- 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

### 12.5.1 Bus Cycle Division

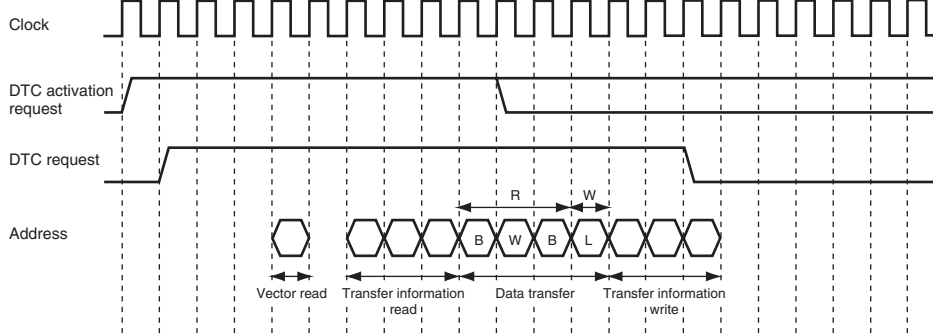
When the transfer data size is word and the SAR and DAR values are not a multiple of 2, the bus cycle is divided and the transfer data is read from or written to in bytes. Similarly, when the transfer data size is longword and the SAR and DAR values are not a multiple of 4, the bus cycle is divided and the transfer data is read from or written to in words.

Table 12.4 shows the relationship among, SAR, DAR, transfer data size, bus cycle division, and access data size. Figure 12.5 shows the bus cycle division example.

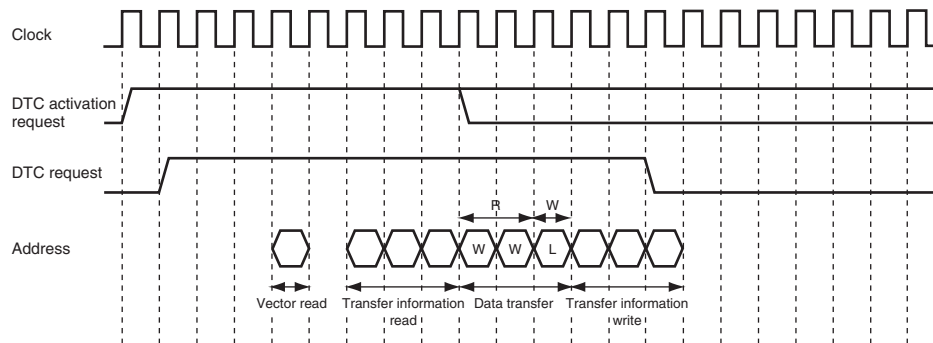
**Table 12.4 Number of Bus Cycle Divisions and Access Size**

SAR and DAR Values	Specified Data Size		
	Byte (B)	Word (W)	Longword (L)
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)

[Example 2: When an odd address and address 4n are specified in SAR and DAR, respectively, and when the data size of transfer is specified



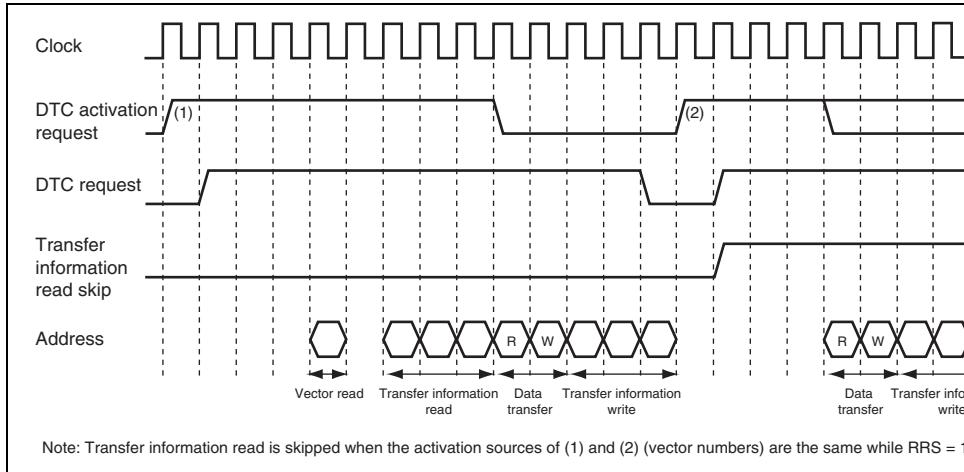
[Example 3: When address 4n + 2 and address 4n are specified in SAR and DAR, respectively, and when the data size of transfer is specified



**Figure 12.5 Bus Cycle Division Example**



cleared to 0, the stored vector number is deleted, and the updated vector table and transfer information are read at the next activation.



**Figure 12.6 Transfer Information Read Skip Timing**

SM1	DM1	SAR	DAR
0	0	Skipped	Skipped
0	1	Skipped	Written back
1	0	Written back	Skipped
1	1	Written back	Written back

#### 12.5.4 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 is reached, an interrupt can be requested to the CPU.

Table 12.6 lists the register function in normal transfer mode. Figure 12.7 shows the memory access sequence in normal transfer mode.

**Table 12.6 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 12.7 Memory Map in Normal Transfer Mode

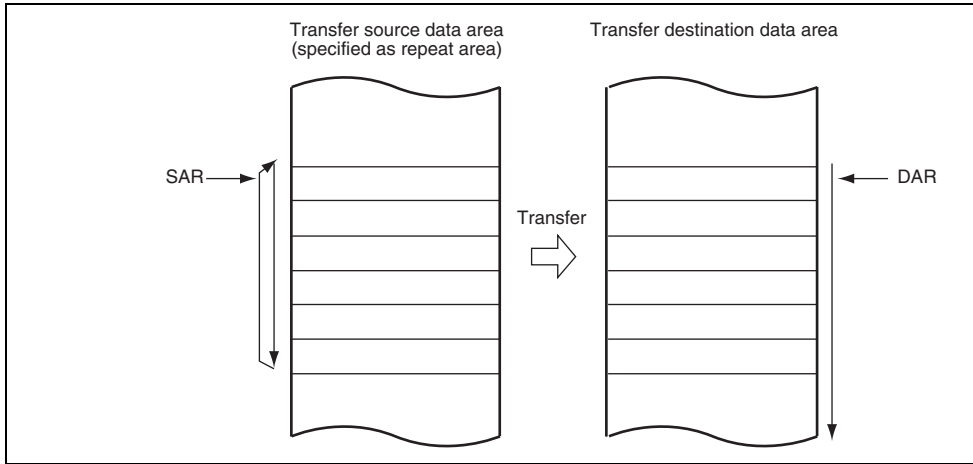
### 12.5.5 Repeat Transfer Mode

In repeat transfer mode, one operation transfers one byte, one word, or one longword of data. For the DTS bit in MRB, either the source or destination can be specified as a repeat area. For 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 interrupt cannot be requested when DISEL = 0.

Table 12.7 lists the register function in repeat transfer mode. Figure 12.8 shows the memory map in repeat transfer mode.

CRAL	Transfer count	CRAL	CRAL
	storage		
CRAL	Transfer count A	CRAL - 1	CRAH
CRB	Transfer count B	Not updated	Not updated

Note: \* Transfer information writeback is skipped.



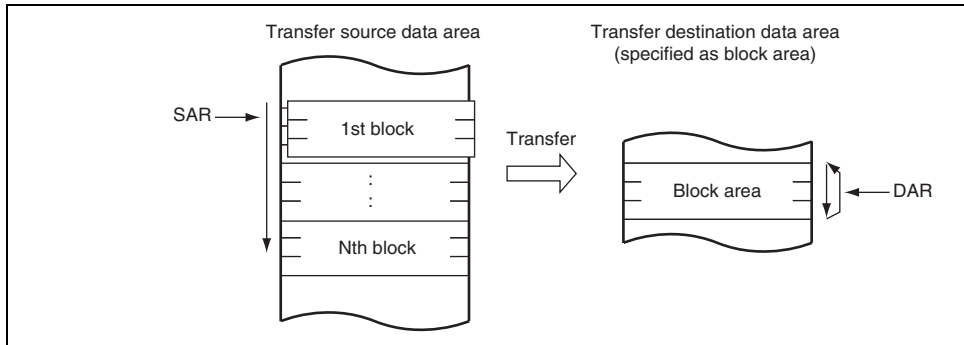
**Figure 12.8 Memory Map in Repeat Transfer Mode  
(When Transfer Source is Specified as Repeat Area)**

Table 12.8 lists the register function in block transfer mode. Figure 12.9 shows the memory map in block transfer mode.

**Table 12.8 Register Function in Block Transfer Mode**

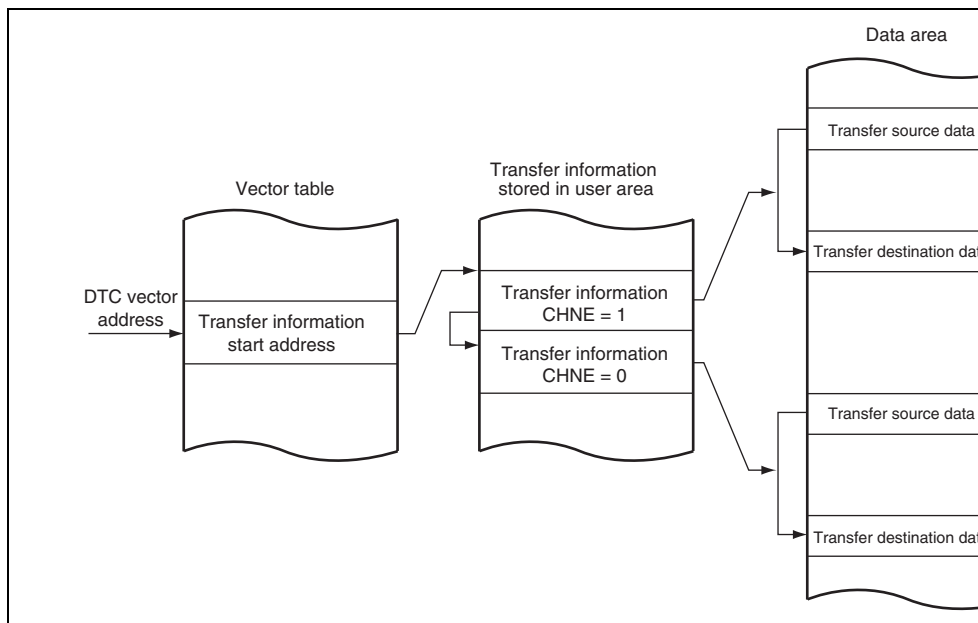
Register	Function	Written Back Value
SAR	Source address	DTS = 0: Incremented/decremented/fixed* DTS = 1: SAR initial value
DAR	Destination address	DTS = 0: DAR initial value DTS = 1: Incremented/decremented/fixed*
CRAH	Block size storage	CRAH
CRAL	Block size counter	CRAH
CRB	Block transfer counter	CRB - 1

Note: \* Transfer information writeback is skipped.

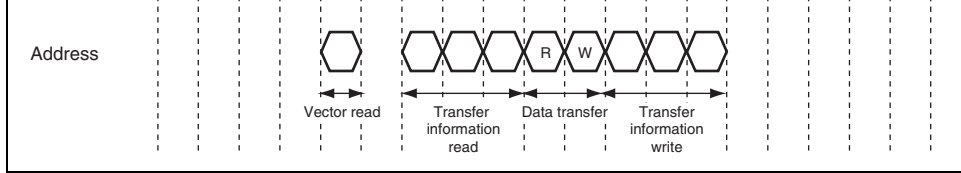


**Figure 12.9 Memory Map in Block Transfer Mode  
(When Transfer Destination is Specified as Block Area)**

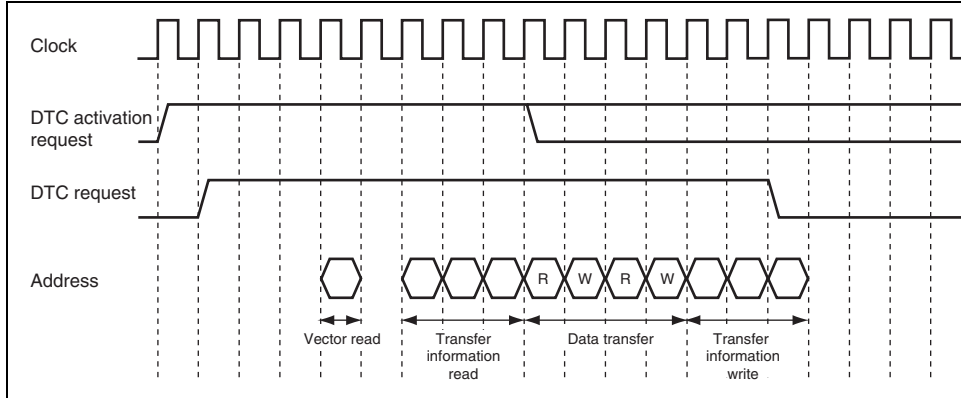
In repeat transfer mode, setting the RCHNE bit in DTCCR and the CHNE and CHNS bits to 1 enables a chain transfer after transfer with transfer counter = 1 has been completed.



**Figure 12.10 Operation of Chain Transfer**

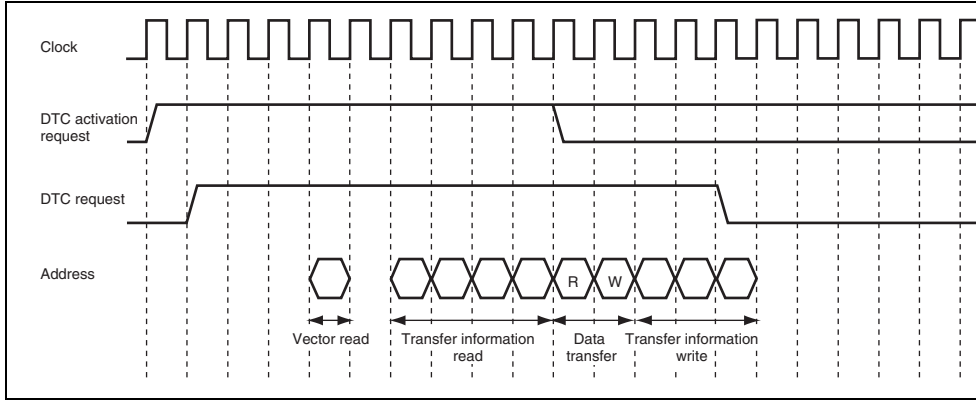


**Figure 12.11 DTC Operation Timing**  
**(Example of Short Address Mode in Normal Transfer Mode or Repeat Transfer Mode)**



**Figure 12.12 DTC Operation Timing**  
**(Example of Short Address Mode in Block Transfer Mode with Block Size of 2)**

**Figure 12.13 DTC Operation Timing (Example of Short Address Mode in Chain T**



**Figure 12.14 DTC Operation Timing (Example of Full Address Mode in Normal Transfer Mode or Repeat Transfer M**



Normal	1	0*1	4*2	3*3	0*1	3*2.3	2*4	1*5	3*6	2*7	1	3*6	2*7	1
Repeat	1	0*1	4*2	3*3	0*1	3*2.3	2*4	1*5	3*6	2*7	1	3*6	2*7	1
Block transfer	1	0*1	4*2	3*3	0*1	3*2.3	2*4	1*5	3•P*6	2•P*7	1•P	3•P*6	2•P*7	1•P

[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 when a longword is transferred while address  $4n + 2$  is specified

Word data read $S_L$	1	1	4	2	2	4	$4 + 2m$	2
Longword data read $S_L$	1	1	8	4	2	8	$12 + 4m$	4
Byte data write $S_M$	1	1	2	2	2	2	$3 + m$	2
Word data write $S_M$	1	1	4	2	2	4	$4 + 2m$	2
Longword data write $S_M$	1	1	8	4	2	8	$12 + 4m$	4
Internal operation $S_N$						1		

[Legend]

m: Number of wait cycles 0 to 7 (For details, see section 9, 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 plus 1).

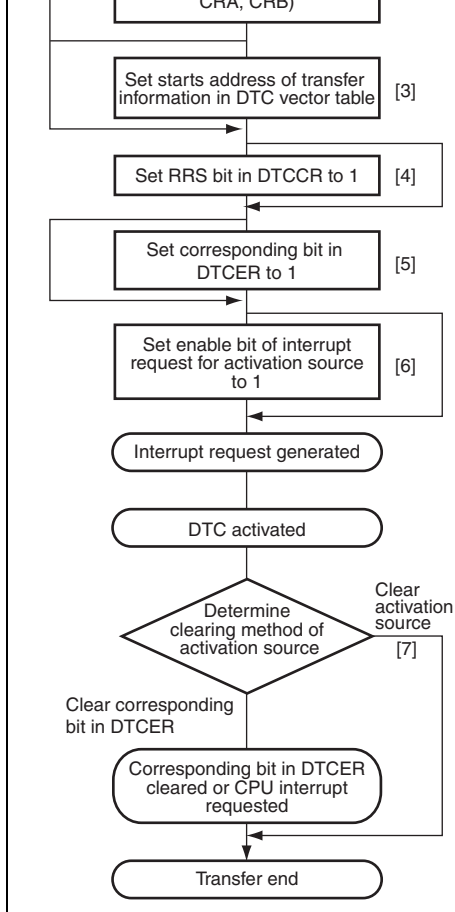
$$\text{Number of execution cycles} = I \cdot S_I + \Sigma (J \cdot S_J + K \cdot S_K + L \cdot S_L + M \cdot S_M) + N \cdot S_N$$

### 12.5.10 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, transfer information writeback. The DTC does not release the bus during transfer information read, single data transfer, or transfer information writeback.

### 12.5.11 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 7, Interrupt Controller.



- information in the data area. For details on setting the transfer information, see section 12.2, Register Descriptions. For details on location of transfer information, see section 12.4, 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 12.4, Location of Transfer Information and DTC Vector Table.
  - [4] Setting the RRS bit to 1 performs a read skip of second or later transfer information when the DTC is activated or deactivated by the same interrupt source. Setting the RRS bit to 0 always allowed. However, the value set during transfer is not valid from the next transfer.
  - [5] Set the bit in DTCER corresponding to the DTC activation source to 1. For the correspondence of interrupt sources and DTCER, refer to table 12.1. The bit in DTCER may be set during 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 activation sources to 1. The DTC is activated when an interrupt from an activation source is generated. For details on 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 source flag or clears the corresponding bit in DTCER and requests an interrupt to the CPU. The operation after that depends on the transfer information. For details, see section 12.2, Register Descriptions and figure 12.4.

**Figure 12.15 DTC with Interrupt Activation**

- the data will be received in DAR, and 128 (H0080) 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.
  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 end (RXI) interrupt. Since the generation of a receive error during the SCI reception 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. An RXI interrupt is generated, and the DTC is activated. The receive data is transferred from the SCI 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. 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.

### 12.7.2 Chain Transfer

An example of DTC chain transfer is shown in which pulse output is performed using the Chain transfer. Chain transfer can be used to perform pulse output data transfer and PPG output trigger counter 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 of the activation source and interrupt generation at the end of the specified number of transfer. Transfer is restricted to the second half of the chain transfer (transfer when CHNE = 0).

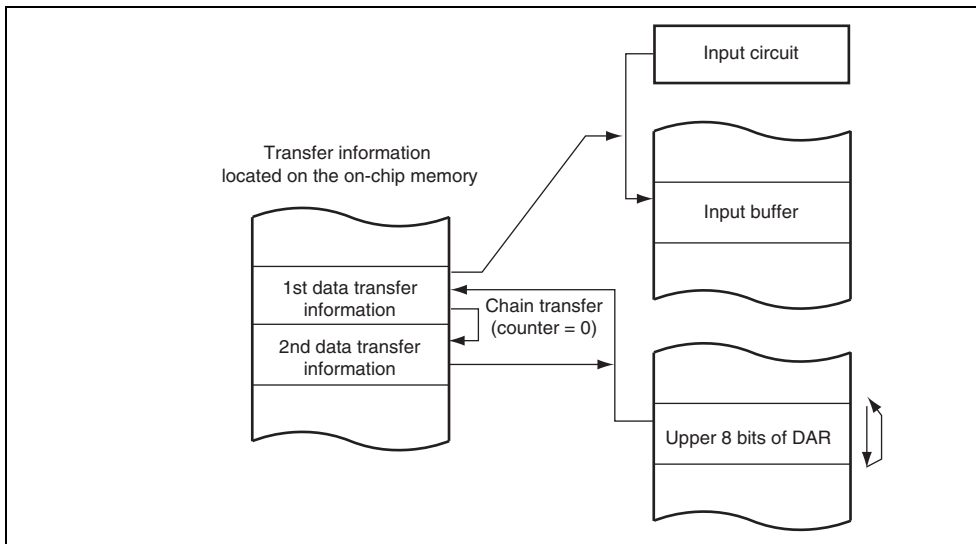
3. Locate the TPU transfer information consecutively after the NDR transfer information.
4. Set the start address of the NDR transfer information to the DTC vector address.
5. Set the bit corresponding to the TGIA interrupt in DTCER to 1.
6. Set TGRA as an output compare register (output disabled) with TIOR, and enable the interrupt with TIER.
7. Set the initial output value in PODR, and the next output value in NDR. Set bits in DTCER for which output is to be performed to 1. Using PCR, select the TPU compare register 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. The set value of the next output trigger period is transferred to TGRA. The activation source 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 issued to the CPU. Termination processing should be performed in the interrupt handling routine.

### 12.7.3 Chain Transfer when Counter = 0

By executing a second data transfer and performing re-setting of the first data transfer 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 address is assumed to have been set to start at lower address H'0000. Figure 12.16 shows the chain transfer when the counter value is 0.

- for the first data transfer reaches 0, the second data transfer is started. Set the upper 8 bits of the transfer source address for the first data transfer to H'21. The lower 16 bits of the transfer destination address of the first data transfer and the transfer counter are H'0000.
- Next, execute the first data transfer the 65536 times specified for the first data transfer 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.
  - 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 12.16 Chain Transfer when Counter = 0**

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 the module stop state. The module stop state cannot be set while the DTC is activated. For details, refer to section 28, Power-Down Modes.

### **12.9.2 On-Chip RAM**

Transfer information can be located in on-chip RAM. In this case, the RAME bit in SYSCON cannot be cleared to 0.

### **12.9.3 DMAC Transfer End Interrupt**

When the DTC is activated by a DMAC transfer end interrupt, the DTE bit of DMDR is 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.

When the DTC is activated by a DMAC transfer end interrupt, even if DISEL=0, an automatic clearing of the relevant activation source flag is not automatically cleared by the DTC. To clear the activation source flag, write 1 to the DTE bit by the DTC transfer and clear the activation source flag to 0.

### **12.9.4 DTCE Bit Setting**

For DTCE bit setting, use bit manipulation instructions such as BSET and BCLR. If all activation sources are disabled, multiple activation sources can be set at one time (only at the initial setting). After writing data after executing a dummy read on the relevant register.

The transfer information start address to be specified in the vector table should be address other than address  $4n$  is specified, the lower 2 bits of the address are regarded as 0.

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.

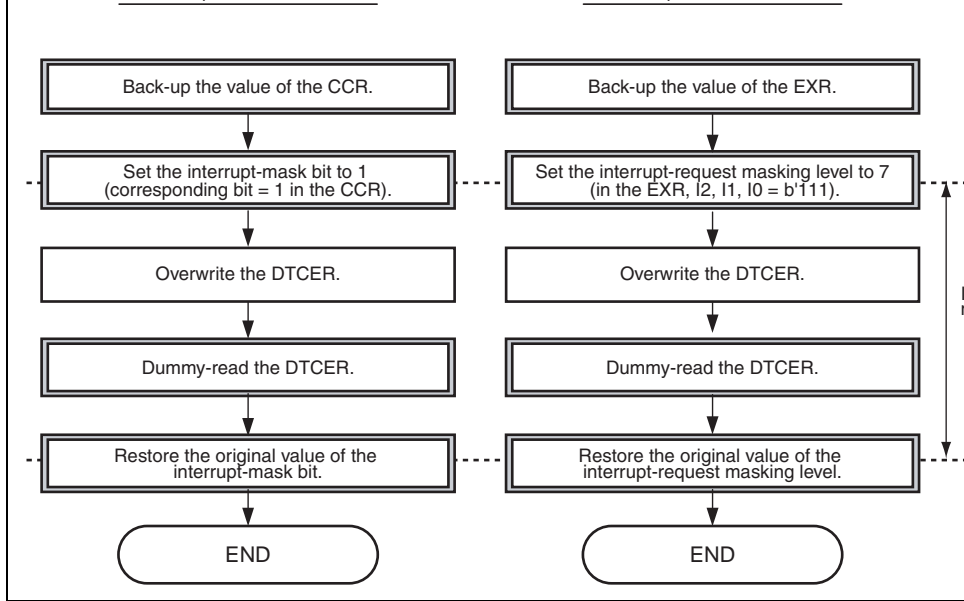
### 12.9.7 Transfer Information Modification

When  $IBCCS = 1$  and the DMAC is used, clear the IBCCS bit to 0 and then set to 1 again after modifying the DTC transfer information in the CPU exception handling routine initiated by the transfer end interrupt.

### 12.9.8 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.





**Figure 12.17 Example of Procedures for Overwriting the DTCER**



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 and capacitive loads up to 30 pF. All of the ports can drive Darlington transistors when functioning as output ports.

Port 2, 3, J and K are Schmitt-trigger input. Schmitt-trigger inputs for other ports are enabled when used as the  $\overline{\text{IRQ}}$ , TPU, TMR, or IIC2 input.

**Table 13.1 Port Functions**

Port	Description	Bit	I/O	Function		Schmitt-Trigger Input*1	Input Pull-up MOS Function
				Input	Output		
Port 1	General I/O port also functioning as interrupt inputs, SCI I/Os, DMAC I/Os, EXDMAC I/Os, A/D converter inputs, TPU inputs, and IIC2 I/Os	7	P17/SCL0	$\overline{\text{IRQ7-A}}$ / $\overline{\text{TCLKD-B}}$ / $\overline{\text{ADTRG1}}$	$\overline{\text{EDRAK1}}$	$\overline{\text{IRQ7-A}}$ , $\overline{\text{TCLKD-B}}$ , $\overline{\text{SCL0}}$	—
		6	P16/SDA0	$\overline{\text{IRQ6-A}}$ / $\overline{\text{TCLKC-B}}$	$\overline{\text{DACK1-A}}$ / $\overline{\text{EDACK1-A}}$	$\overline{\text{IRQ6-A}}$ , $\overline{\text{TCLKC-B}}$ , $\overline{\text{SDA0}}$	
		5	P15/SCL1	$\overline{\text{IRQ5-A}}$ / $\overline{\text{TCLKB-B}}$ / $\overline{\text{RxD5}}$ / $\overline{\text{IrRxD}}$	$\overline{\text{TEND1-A}}$ / $\overline{\text{ETEND1-A}}$	$\overline{\text{IRQ5-A}}$ , $\overline{\text{TCLKB-B}}$ , $\overline{\text{SCL1}}$	
		4	P14/SDA1	$\overline{\text{DREQ1-A}}$ / $\overline{\text{IRQ4-A}}$ / $\overline{\text{TCLKA-B}}$ / $\overline{\text{EDREQ1-A}}$	$\overline{\text{TxD5}}$ / $\overline{\text{IrTxD}}$	$\overline{\text{IRQ4-A}}$ , $\overline{\text{TCLKA-B}}$ , $\overline{\text{SDA1}}$	
		3	P13	$\overline{\text{ADTRG0-A}}$ / $\overline{\text{IRQ3-A}}$	$\overline{\text{EDRAK0}}$	$\overline{\text{IRQ3-A}}$	

Port 2	General I/O port also functioning as interrupt inputs, PPG outputs, TPU I/Os, TMR I/Os, and SCI I/Os	7	P27/ TIOCB5	TIOCA5	PO7	P27, TIOCB5, TIOCA5	—
		6	P26/ TIOCA5	—	PO6/TMO1/ TxD1	All input functions	
		5	P25/ TIOCA4	TMCI1/ RxD1	PO5	P25, TIOCA4, TMCI1	
		4	P24/ TIOCB4/ SCK1	TIOCA4/ TMRI1	PO4	P24, TIOCB4, TIOCA4, TMRI1	
		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, TMCI0	
		0	P20/ TIOCB3/ SCK0	TIOCA3/ TMRI0/ $\overline{\text{IRQ8-A}}$	PO0	P20, $\overline{\text{IRQ8-A}}$ , TIOCB3, TIOCA3, TMRI0	

	4	P34/ TIOCA1		P012/ TEND1-B/ ETEND3	All input functions		
	3	P33/ TIOCD0	TIOCC0/ TCLKB-A/ DREQ1-B/ EDREQ3	PO11	P33, TIOCD0, TIOCC0, TCLKB-A		
	2	P32/ TIOCC0	TCLKA-A	PO10/ DACK0-B/ EDACK2	All input functions		
	1	P31/ TIOCB0	TIOCA0	PO9/ TEND0-B/ ETEND2	All input functions		
	0	P30/ TIOCA0	DREQ0-B/ EDREQ2	PO8	P30, TIOCA0		
Port 5	General input port also functioning as interrupt inputs, A/D converter inputs, and D/A converter outputs	7	—	P57/AN7/ IRQ7-B	DA1	IRQ7-B	—
		6	—	P56/AN6/ IRQ6-B	DA0	IRQ6-B	
		5	—	P55/AN5/ IRQ5-B	—	IRQ5-B	
		4	—	P54/AN4/ IRQ4-B	—	IRQ4-B	
		3	—	P53/AN3/ IRQ3-B	—	IRQ3-B	
		2	—	P52/AN2/ IRQ2-B	—	IRQ2-B	
		1	—	P51/AN1/ IRQ1-B	—	IRQ1-B	
		0	—	P50/AN0/ IRQ0-B	—	IRQ0-B	

			$\overline{\text{DREQ3}}$ / $\overline{\text{IRQ11-B}}$ / TMS/ $\overline{\text{EDREQ1-B}}$		$\overline{\text{IRQ11-B}}$ , TMS		
		2	P62/SCK4	$\overline{\text{IRQ10-B}}$ / $\overline{\text{TRST}}$	TMO2/ $\overline{\text{DACK2}}$ / $\overline{\text{EDACK0-B}}$	$\overline{\text{IRQ10-B}}$ , $\overline{\text{TRST}}$	
		1	P61	TMC12/ RxD4/ $\overline{\text{IRQ9-B}}$	$\overline{\text{TEND2}}$ / $\overline{\text{ETEND0-B}}$	TMC12, $\overline{\text{IRQ9-B}}$	
		0	P60	TMR12/ DREQ2/ $\overline{\text{IRQ8-B}}$ / $\overline{\text{EDREQ0-B}}$	TxD4	TMR12, $\overline{\text{IRQ8-B}}$	
Port A	General I/O port also functioning as system clock output and bus control I/Os	7	—	PA7	$\text{B}\phi$	—	—
		6	PA6	—	$\overline{\text{AS/AH}}$ / $\overline{\text{BS-B}}$		
		5	PA5	—	$\overline{\text{RD}}$		
		4	PA4	—	$\overline{\text{LHWR/LUB}}$		
		3	PA3	—	$\overline{\text{LLWR/LLB}}$		
		2	PA2	—	$\overline{\text{BREQ}}$ / $\overline{\text{WAIT}}$	—	
		1	PA1	—	$\overline{\text{BACK}}$ / (RD/WR-A)		
		0	PA0	—	$\overline{\text{BREQ0}}$ / $\overline{\text{BS-A}}$		

				$\overline{\text{CS7-A}}$			
				$\overline{\text{CAS}}$			
	2	PB2	—	$\overline{\text{CS2-A}}$			
				$\overline{\text{CS6-A}}$			
				$\overline{\text{RAS}}$			
	1	PB1	—	$\overline{\text{CS1}}$			
				$\overline{\text{CS2-B}}$			
				$\overline{\text{CS5-A}}$			
				$\overline{\text{CS6-B}}$			
				$\overline{\text{CS7-B}}$			
	0	PB0	—	$\overline{\text{CS0}}$			
				$\overline{\text{CS4-A}}$			
				$\overline{\text{CS5-B}}$			
Port C	General I/O port also functioning as bus control I/Os and A/D converter inputs	7	—	—	—	—	
		6	—	—	—		
		5	—	—	—		
		4	—	—	—		
		3	PC3	—	$\overline{\text{LLCAS}}$	$\overline{\text{DQMLL}}$	
		2	PC2	—	$\overline{\text{LUCAS}}$	$\overline{\text{DQMLU}}$	
		1	—	—	—		
		0	—	—	—		

		1	PD1	—	A1		
		0	PD0	—	A0		
Port E <sup>※3</sup>	General I/O port also functioning as address outputs	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 also functioning as address outputs	7	PF7	—	A23	—	O
		6	PF6	—	A22		
		5	PF5	—	A21		
		4	PF4	—	A20		
		3	PF3	—	A19		
		2	PF2	—	A18		
		1	PF1	—	A17		
		0	PF0	—	A16		



		1	PH1/D1* <sup>2</sup>	—	—		
		0	PH0/D0* <sup>2</sup>	—	—		
Port I	General I/O port also functioning as bi-directional data bus	7	PI7/D15* <sup>2</sup>	—	—	—	O
		6	PI6/D14* <sup>2</sup>	—	—		
		5	PI5/D13* <sup>2</sup>	—	—		
		4	PI4/D12* <sup>2</sup>	—	—		
		3	PI3/D11* <sup>2</sup>	—	—		
		2	PI2/D10* <sup>2</sup>	—	—		
		1	PI1/D9* <sup>2</sup>	—	—		
		0	PI0/D8* <sup>2</sup>	—	—		
		Port J* <sup>4</sup>	General I/O port also functioning as PPG and TPU I/O	7	PJ7/TIOCB8	TIOCA8/ TCLKH	PO23
6	PJ6/TIOCA8			—	PO22		
5	PJ5/TIOCB7			TIOCA7/ TCLKG	PO21		
4	PJ4/TIOCA7			—	PO20		
3	PJ3/TIOCD6			TIOCC6/ TCLKF	PO19		
2	PJ2/TIOCC6			TCLKE	PO18		
1	PJ1/TIOCB6			TIOCA6	PO17		
0	PJ0/TIOCA6			—	PO16		

		1	PK1/TIOCB9	TIOCA9	PO25		
		0	PK0/TIOCA9	—	PO24		
Port M	General I/O port also functioning as SCI I/Os	7	—	—	—	—	—
		6	—	—	—		
		5	—	—	—		
		4	PM4	—	—		
		3	PM3	—	—		
		2	PM2	—	—		
		1	PM1	RxD6	—		
		0	PM0	—	TxD6		

- Notes:
1. Pins without Schmitt-trigger input have CMOS input functions.
  2. Addresses are also output when accessing to the address/data multiplexed I/O.
  3. Pins are disabled when PCJKE = 1.
  4. Pins are disabled when PCJKE = 0.

Port 3	8	0	0	0	0	—	—
Port 5	8	—	—	0	0	—	—
Port 6* <sup>4</sup>	6	0	0	0	0	—	—
Port A	8	0	0	0	0	—	—
Port B	8	0	0	0	0	—	—
Port C* <sup>1</sup>	2	0	0	0	0	—	—
Port D* <sup>2</sup>	8	0	0	0	0	0	0
Port E* <sup>2</sup>	8	0	0	0	0	0	0
Port F	8	0	0	0	0	0	0
Port H	8	0	0	0	0	0	0
Port I	8	0	0	0	0	0	0
Port J* <sup>3</sup>	8	0	0	0	0	0	0
Port K* <sup>3</sup>	8	0	0	0	0	0	0
Port M* <sup>5</sup>	5	0	0	0	0	—	—

[Legend]

O: Register exists

—: No register exists

- Notes:
1. Bits 2 and 3 are valid and the other bits are reserved for port C registers. The write value should be the same as the initial value.
  2. Do not access when PCJKE = 1.
  3. Do not access when PCJKE = 0.
  4. The lower six bits are valid and the upper two bits are reserved for port 6 registers. The write value should be the same as the initial value.
  5. The lower five bits are valid and the upper three bits are reserved for port M registers. The write value should be the same as the initial value.

Bit	7	6	5	4	3	2	1
Bit Name	Pn7DDR	Pn6DDR	Pn5DDR	Pn4DDR	Pn3DDR	Pn2DDR	Pn1DDR
Initial Value	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W

Notes: 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 M registers.  
Bits 2 and 3 are valid and the other bits are reserved for port C registers.  
Registers of ports J and K cannot be accessed when PCJKE = 0.  
Registers of ports D and E cannot be accessed when PCJKE = 1.

**Table 13.3 Startup Mode and Initial Value**

Port	Startup Mode	
	External Extended Mode	Single-Chip Mode
Port A	H'80	H'00
Other ports	H'00	H'00

Notes: 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 M registers.  
 Bits 2 and 3 are valid and the other bits are reserved for port C registers.  
 Registers of ports J and K cannot be accessed when PCJKE = 0.  
 Registers of ports D and E cannot be accessed when PCJKE = 1.

### 13.1.3 Port Register (PORTn) (n = 1, 2, 3, 5, 6, A to F, H to K, and M)

PORT is an 8-bit read-only register that reflects the port pin state. A write to PORT is ignored. When PORT is read, the DR bits that correspond to the respective DDR bits set to 1 are read. 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 state.

Bit	7	6	5	4	3	2	1	
Bit Name	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	U
R/W	R	R	R	R	R	R	R	

Notes: 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 M registers.  
 Bits 2 and 3 are valid and the other bits are reserved for port C registers.  
 Registers of ports J and K cannot be accessed when PCJKE = 0.  
 Registers of ports D and E cannot be accessed when PCJKE = 1.

When PORT is read, the pin state is always read regardless of the ICR value. When the ICR is cleared to 0 at this time, the read pin state is not reflected in a corresponding on-chip port module.

If ICR is modified, an internal edge may occur depending on the pin state. Accordingly, ICR should be modified when the corresponding input pins are not used. For example, an  $\overline{IRQ}$  should be modified when the corresponding input pins are not used. For example, an  $\overline{IRQ}$  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 ICR setting, the edge should be cancelled.

The initial value of ICR is H'00.

Bit	7	6	5	4	3	2	1
Bit Name	Pn7ICR	Pn6ICR	Pn5ICR	Pn4ICR	Pn3ICR	Pn2ICR	Pn1ICR
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- Notes:
- 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 M registers.
  - Bits 2 and 3 are valid and the other bits are reserved for port C registers.
  - Registers of ports J and K cannot be accessed when PCJKE = 0.
  - Registers of ports D and E cannot be accessed when PCJKE = 1.

**Table 13.4 Input Pull-Up MOS State**

Port	Pin State	Reset	Hardware Standby Mode	Software Standby Mode	Other
Port D	Address output	OFF	OFF	OFF	OFF
	Port output	OFF	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF	ON
Port E	Address output	OFF	OFF	OFF	OFF
	Port output	OFF	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF	ON
Port F	Address output	OFF	OFF	OFF	OFF
	Port output	OFF	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF	ON
Port H	Data input/output	OFF	OFF	OFF	OFF
	Port output	OFF	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF	ON
Port I	Data input/output	OFF	OFF	OFF	OFF
	Port output	OFF	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF	ON
Port J	Peripheral module output	OFF	OFF	OFF	OFF
	Port output	OFF	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF	ON

### 13.1.6 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 NMO 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
Bit Name	Pn7ODR	Pn6ODR	Pn5ODR	Pn4ODR	Pn3ODR	Pn2ODR	Pn1ODR
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



For a pin whose initial value changes according to the activation mode, "Initial value E" the initial value when the LSI is started up in external extended mode and "Initial value" indicates the initial value when the LSI is started in single-chip mode.

### 13.2.1 Port 1

#### (1) P17/ $\overline{\text{IRQ7}}$ -A/TCLKD-B/SCL0/ $\overline{\text{EDRAK1}}$ / $\overline{\text{ADTRG1}}$

The pin function is switched as shown below according to the combination of the EXDMAC IIC2 register settings and the P17DDR bit setting.

Module Name	Pin Function	Setting		
		EXDMAC	IIC2	I/O Port
		$\overline{\text{EDRAK1\_OE}}$	SCL0_OE	P17DDR
EXDMAC	$\overline{\text{EDRAK1}}$ output	1	—	—
IIC2	SCL0 input/output	0	1	—
I/O port	P17 output	0	0	1
	P17 input (initial value)	0	0	0

I/O port	P16 output	0	0	0	1
	P16 input (initial value)	0	0	0	0

(3) P15/RxD5/IrRxD/TEND1-A/ETEND1-A/IRQ5-A/TCLKB-B/SCL1

The pin function is switched as shown below according to the combination of the EXDMAC, DMAC, and IIC2 register settings and the P15DDR bit setting.

Module Name	Pin Function	Setting			
		EXDMAC	DMAC	IIC2	I/O
		ETEND1A_OE	TEND1A_OE	SCL1_OE	P15
EXDMAC	ETEND1-A output	1	—	—	—
DMAC	TEND1-A output	0	1	—	—
IIC2	SCL1 input/output	0	0	1	—
I/O port	P15 output	0	0	0	1
	P15 input (initial value)	0	0	0	0

I/O port	P14 output	0	0	0	1
	P14 input (initial value)	0	0	0	0

(5)  $\overline{\text{P13}}/\overline{\text{ADTRG0-A}}/\overline{\text{IRQ3-A}}/\overline{\text{EDRAK0}}$

The pin function is switched as shown below according to the register setting of EXDMAC the P13DDR bit setting.

Module Name	Pin Function	Setting	
		EXDMAC	I/O Port
		$\overline{\text{EDRAK0}}_{\text{OE}}$	P13DDR
EXDMAC	EDRAK0 output	1	—
I/O port	P13 output	0	1
	P13 input (initial value)	0	0

SCI	SCK2 output	0	0	1	—
I/O port	P12 output	0	0	0	1
	P12 input (initial value)	0	0	0	0

(7) P11/RxD2/ $\overline{\text{TEND0-A}}$ / $\overline{\text{IRQ1-A}}$ / $\overline{\text{ETEND0-A}}$

The pin function is switched as shown below according to the combination of the EXDMAC DMAC register settings and the P11DDR bit setting.

Module Name	Pin Function	Setting		
		EXDMAC	DMAC	I/O Port
		$\overline{\text{ETEND0A\_OE}}$	$\overline{\text{TEND0A\_OE}}$	P11DDR
EXDMAC	$\overline{\text{ETEND0-A}}$ output	1	—	—
DMAC	$\overline{\text{TEND0-A}}$ output	0	1	—
I/O port	P11 output	0	0	1
	P11 input (initial value)	0	0	0

### 13.2.2 Port 2

#### (1) P27/PO7/TIOCA5/TIOCB5

The pin function is switched as shown below according to the combination of the TPU and PPG register settings, and the 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

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/ $\overline{\text{IRQ13}}$ -A**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings, and the 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

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 the 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

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/TMCIO/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 the 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



PPG	PO0 output	0	0	1	—
I/O port	P20 output	0	0	0	1
	P20 input (initial value)	0	0	0	0

### 13.2.3 Port 3

#### (1) P37/PO15/TIOCA2/TIOCB2/TCLKD-A/ $\overline{\text{EDRAK3}}$

The pin function is switched as shown below according to the combination of the EXDMAC and PPG register settings and the P37DDR bit setting.

Module Name	Pin Function	Setting		
		EXDMAC	TPU	PPG
		$\overline{\text{EDRAK3\_OE}}$	TIOCB2_OE	PO15_OE
EXDMAC	$\overline{\text{EDRAK3}}$ output	1	—	—
TPU	TIOCB2 output	0	1	—
PPG	PO15 output	0	0	1
I/O port	P37 output	0	0	0
	P37 input (initial value)	0	0	0

PPG	PO14 output	0	0	1	—
I/O port	P36 output	0	0	0	1
	P36 input (initial value)	0	0	0	0

### (3) P35/PO13/TIOCA1/TIOCB1/TCLKC-A/ $\overline{\text{DACK1-B}}$ / $\overline{\text{EDACK3}}$

The pin function is switched as shown below according to the combination of the EXDMAC, DMAC, TPU, and PPG register settings and the P35DDR bit setting.

Module Name	Pin Function	Setting			
		EXDMAC	DMAC	TPU	PPG
		$\overline{\text{EDACK3\_OE}}$	$\overline{\text{DACK1B\_OE}}$	TIOCB1_OE	PO13_OE
EXDMAC	EDACK3 output	1	—	—	—
DMAC	$\overline{\text{DACK1-B}}$ output	0	1	—	—
TPU	TIOCB1 output	0	0	1	—
PPG	PO13 output	0	0	0	1
I/O port	P35 output	0	0	0	0
	P35 input (initial value)	0	0	0	0

TPU	TIOCA1 output	0	0	1	—
PPG	PO12 output	0	0	0	1
I/O port	P34 output	0	0	0	0
	P34 input (initial value)	0	0	0	0

(5) **P33/PO11/TIOCC0/TIOCD0/TCLKB-A/DREQ1-B/EDREQ3**

The pin function is switched as shown below according to the combination of the TPU and PPG register settings, and the P33DDR bit setting.

Module Name	Pin Function	Setting		
		TPU	PPG	I/O Port
		TIOCD0_OE	PO11_OE	P33DDR
TPU	TIOCD0 output	1	—	—
PPG	PO11 output	0	1	—
I/O port	P33 output	0	0	1
	P33 input (initial value)	0	0	0

TPU	TIOCC0 output	0	0	1	—
PPG	PO10 output	0	0	0	1
I/O port	P32 output	0	0	0	0
	P32 input (initial value)	0	0	0	0

(7) **P31/PO9/TIOCA0/TIOCB0/TEND0-B/ETEND2**

The pin function is switched as shown below according to the combination of the EXDMAC, DMAC, TPU, and PPG register settings and the P31DDR bit setting.

Module Name	Pin Function	Setting			
		EXDMAC	DMAC	TPU	PPG
		$\overline{\text{ETEND2\_OE}}$	$\overline{\text{TEND0B\_OE}}$	TIOCB0_OE	PO9_OE
EXDMAC	$\overline{\text{ETEND2}}$ output	1	—	—	—
DMAC	$\overline{\text{TEND0-B}}$ output	0	1	—	—
TPU	TIOCB0 output	0	0	1	—
PPG	PO9 output	0	0	0	1
I/O port	P31 output	0	0	0	0
	P31 input (initial value)	0	0	0	0

I/O port	P30 output	0	0	1
	P30 input (initial value)	0	0	0

### 13.2.4 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

EXDMAC	$\overline{\text{EDACK1-B}}$ output	Except for	1	—	—	—
DMAC	$\overline{\text{DACK3}}$ output	boundary scan	0	1	—	—
TMR	TMO3 output	enabled	0	0	1	—
I/O port	P65 output	mode*	0	0	0	1
	P65 input (initial value)		0	0	0	0

Note: \* These pins are boundary scan dedicated input pins during boundary scan enabled mode.

## (2) P64/TMCI3/ $\overline{\text{TEND3}}$ / $\overline{\text{ETEND1-B}}$ /TDI

The pin function is switched as shown below according to the combination of the EXDMAC DMAC register settings and P64DDR bit setting.

Module Name	Pin Function	MCU Operation Mode	Setting		
			EXDMAC $\overline{\text{ETEND1B\_OE}}$	DMAC $\overline{\text{TEND3\_OE}}$	I/O P6
EXDMAC	$\overline{\text{ETEND1-B}}$ output	Except for	1	—	—
DMAC	$\overline{\text{TEND3}}$ output	boundary scan enabled mode*	0	1	—
I/O port	P64 output		0	0	1
	P64 input (initial value)		0	0	0

Note: \* These pins are boundary scan dedicated input pins during boundary scan enabled mode.

Note: These pins are boundary scan dedicated input pins during boundary scan mode.

#### (4) P62/TMO2/SCK4/ $\overline{\text{DACK2}}$ / $\overline{\text{EDACK0-B}}$ / $\overline{\text{IRQ10-B}}$ / $\overline{\text{TRST}}$

The pin function is switched as shown below according to the combination of the EXDMAC, DMAC, TMR, and SCI register settings and P62DDR bit setting.

Module Name	Pin Function	MCU Operation Mode	Setting			
			EXDMAC	DMAC	TMR	SCI
			$\overline{\text{EDACK0B\_OE}}$	$\overline{\text{DACK2\_OE}}$	TMO2_OE	SCK4_OE
EXDMAC	$\overline{\text{EDACK0-B}}$ output	Except for boundary scan enabled mode*	1	—	—	—
DMAC	$\overline{\text{DACK2}}$ output	scan enabled mode*	0	1	—	—
TMR	TMO2 output		0	0	1	—
SCI	SCK4 output		0	0	0	1
I/O port	P62 output		0	0	0	0
	P62 input (initial value)		0	0	0	0

Note: \* These pins are boundary scan dedicated input pins during boundary scan mode.

I/O port	P61 output	0	0	1
	P61 input (initial value)	0	0	0

#### (6) P60/TMRI2/TxD4/DREQ2/EDREQ0-B/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	I/O Port
		TxD4_OE	P60DDR
SCI	TxD4 output	1	—
I/O port	P60 output	0	1
	P60 input (initial value)	0	0



PA7 input 0  
(initial value S)

[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 operating EXPE bit, bus controller register, the port function control register (PFCR), and the PA6 settings.

Module Name	Pin Function	Setting			
		Bus Controller	I/O Port		
		$\overline{AH\_OE}$	$\overline{BSB\_OE}$	$\overline{AS\_OE}$	PA6
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)

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/LHWR/LUB

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, the port function control register (PFCR), and the PA4 pin settings.

Module Name	Pin Function	Setting		
		Bus Controller	I/O Port	
		<u>LUB_OE</u> * <sup>2</sup>	<u>LHWR_OE</u> * <sup>2</sup>	PA4DDR
Bus controller	<u>LUB</u> output* <sup>1</sup>	1	—	—
	<u>LHWR</u> 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 address is not specified or while LHWR<sub>OE</sub> = 1, this pin functions as the LUB output; otherwise, it functions as the LHWR output.

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 the 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

Bus controller	BACK output *	1	—	—	—
	RD/WR-A 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/BREQO/BS-A

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, bus controller register, the port function control register (PFCR), and the PA0 pin settings.

Module Name	Pin Function	Setting		
		I/O Port BS-A_OE	Bus Controller BREQO_OE	I/O Port PA0DDR
Bus controller	BS-A output*	1	—	—
	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)

generator			
I/O port	PB7 output	0	1
	PB7 input (initial value)	0	0

Note: \* Valid in SDRAM mode

## (2) PB6/ $\overline{\text{CS6-D}}$ / $(\text{RD}/\overline{\text{WR}}\text{-B})/\overline{\text{ADTRG0-B}}$

The pin function is switched as shown below according to the combination of operating EXPE bit, bus controller register, port function control register (PFCR), and the PB6DD settings.

Module Name	Pin Function	Setting			
		I/O Port			
		Byte control SRAM Selection	$(\text{RD}/\overline{\text{WR}})\text{-B\_OE}$	$\overline{\text{CS6D\_OE}}$	PB6DD
Bus controller	$\text{RD}/\overline{\text{WR}}\text{-B output}^*$	1	—	—	—
		0	1	—	—
	$\overline{\text{CS6-D output}}^*$	0	0	1	—
I/O port	PB6 output	0	0	0	1
	PB6 input (initial value)	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

		0	1		
	$\overline{\text{CS5-D}}$ output*	0	0	1	—
I/O port	PB5 output	0	0	0	1
	PB5 input (initial value)	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

#### (4) $\text{PB4}/\overline{\text{CS4-B}}/\overline{\text{WE}}$

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, the bus controller register, the port function control register (PFCR), and the PFCR bit settings.

Module Name	Pin Function	Setting		
		Bus Controller		I/O Port
		$\overline{\text{WE}}_{\text{OE}}$	$\overline{\text{CS4B}}_{\text{OE}}$	PB4DDR
Bus controller	$\overline{\text{WE}}$ output*	1	—	—
	$\overline{\text{CS4-B}}$ output*	0	1	—
I/O port	PB4 output	0	0	1
	PB4 input (initial value)	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

	$\overline{\text{CS7-A}}$ output*	0	—	1	—
I/O port	PB3 output	0	0	0	1
	PB3 input (initial value)	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

### (6) $\overline{\text{PB2}}/\overline{\text{CS2-A}}/\overline{\text{CS6-A}}/\overline{\text{RAS}}$

The pin function is switched as shown below according to the combination of operating EXPE bit, the bus controller register, the port function control register (PFCR), and the I/O port bit settings.

Module Name	Pin Function	Setting			
		Bus Controller		I/O Port	
		$\overline{\text{RAS\_OE}}$	$\overline{\text{CS2A\_OE}}$	$\overline{\text{CS6A\_OE}}$	P
Bus controller	$\overline{\text{RAS}}$ output*	1	—	—	—
	$\overline{\text{CS2-A}}$ output*	0	1	—	—
	$\overline{\text{CS6-A}}$ output*	0	—	1	—
I/O port	PB2 output	0	0	0	1
	PB2 input (initial value)	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

	$\overline{CS5}$ -A output*	—	—	1	—	—
	$\overline{CS6}$ -B output*	—	—	—	1	—
	$\overline{CS7}$ -B output*	—	—	—	—	1
I/O port	PB1 output	0	0	0	0	0
	PB1 input (initial value)	0	0	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

### (8) PB0/ $\overline{CS0}$ / $\overline{CS4}$ -A/ $\overline{CS5}$ -B

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, the port function control register (PFCR), and the PB0DDR bit settings.

Module Name	Pin Function	Setting			
		I/O Port			
		$\overline{CS0}$ _OE	$\overline{CS4A}$ _OE	$\overline{CS5B}$ _OE	PB0
Bus controller	$\overline{CS0}$ output (initial value E)	1	—	—	—
	$\overline{CS4}$ -A output	—	1	—	—
	$\overline{CS5}$ -B output	—	—	1	—
I/O port	PB0 output	0	0	0	1
	PB0 input (initial value S)	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



	DQMLL output*	—	1	—
I/O port	PC3 output	0	0	1
	PC3 input (initial value)	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

## (2) PC2/ $\overline{\text{LUCAS}}$ /DQMLU

The pin function is switched as shown below according to the combination of operating EXPE bit, the bus controller register, and the PC2DDR bit settings.

Module Name	Pin Function	Setting		
		Bus Controller		I/O Port
		$\overline{\text{LUCAS\_OE}}$	DQMLU_OE	PC2DDR
Bus controller	$\overline{\text{LUCAS}}$ output*	1	—	—
	DQMLU output*	—	1	—
I/O port	PC2 output	0	0	1
	PC2 input (initial value)	0	0	0

Note: \* Valid in external extended mode (EXPE = 1)

Module Name	Pin Function	MCU Operating Mode	Setting
			I/O Port PDnDDR
Bus controller	Address output	On-chip ROM disabled extended mode	—
		On-chip ROM enabled extended mode	1
I/O port	PDn output	Single-chip mode*	1
	PDn 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)

### 13.2.10 Port E

The pin function of port E can be switched with that of port K according to the combination of the operating mode, the EXPE bit, and the PCJKE bit settings. The pin function of port E can be switched according to the PCJKE bit setting in the single-chip mode (EXPE = 0). However, you cannot change the setting of the PCJKE bit in external extended mode. For details, see section 13.2.10 Port Function Control Register D (PFCRD).

[Legend]

n: 0 to 7

Note: \* Address output is enabled by setting PDnDDR = 1 in external extended mode (EXPE = 1)

### 13.2.11 Port F

#### (1) PF7/A23

The pin function is switched as shown below according to the combination of operating EXPE bit, the port function control register (PFCR), and the PF7DDR bit settings.

Module Name	Pin Function	Setting	
		I/O Port	
		A23_OE	PF7DDR
Bus controller	A23 output*	1	—
I/O port	PF7 output	0	1
	PF7 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

PF6 input  
(initial value)

0

0

Note: \* Valid in external extended mode (EXPE = 1)

### (3) PF5/A21

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, the port function control register (PFCR), and the PF5DDR bit settings.

Module Name	Pin Function	Setting	
		A21_OE	PF5DDR
Bus controller	A21 output*	1	—
I/O port	PF5 output	0	1
	PF5 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

Modes other than on-chip ROM disabled extended mode	Bus controller	A20 output*	1	—
	I/O port	PF4 output	0	1
		PF4 input (initial value)	0	0

Note: \* Valid in external extended mode (EXPE = 1)

### (5) PF3/A19

The pin function is switched as shown below according to the combination of operating EXPE bit, the port function control register (PFCR), and the PF3DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			A19_OE	I/O Port 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)

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)

## (7) PF1/A17

The pin function is switched as shown below according to the combination of operating mode, EXPE bit, the port function control register (PFCR), and the PF1DDR bit settings.

MCU Operating Mode	Module Name	Pin Function	Setting	
			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)

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)

### 13.2.12 Port H

#### (1) PH7/D7, PH6/D6, PH5/D5, PH4/D4, PH3/D3, PH2/D2, PH1/D1, PH0/D0

The pin function is switched as shown below according to the combination of operating EXPE bit, and the 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)

	(initial value E)		
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 7

Note: \* Valid in external extended mode (EXPE = 1)

### 13.2.14 Port J

The pin function of port J can be switched with that of port D according to the combination of the operating mode, the EXPE bit, and the PCJKE bit settings. The pin function of port J can be switched according to the PCJKE bit setting in the single-chip mode (EXPE = 0). However, you cannot change the setting of the PCJKE bit in external extended mode. For details, see section 13.2.13. Port Function Control Register D (PFCRD).



I/O port	PJ7 output*	0	0	1
	PJ7 input*	0	0	0

Note: \* Valid when PCJKE = 1.

## (2) PJ6/TIOCA8/PO22

The pin function is switched as shown below according to the combination of register settings for PPG and TPU, setting of the port function control register (PFCR), and the PJ6DDR bit.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO22_OE	TIOCA8_OE	PJ6DDR
PPG	PO22 output*	1	—	—
TPU	TIOCA8 output*	0	1	—
I/O port	PJ6 output*	0	0	1
	PJ6 input*	0	0	0

Note: \* Valid when PCJKE = 1.

I/O port	PJ5 output*	0	0	1
	PJ5 input*	0	0	0

Note: \* Valid when PCJKE = 1.

#### (4) PJ4/TIOCA7/PO20

The pin function is switched as shown below according to the combination of register settings for PPG and TPU, setting of the port function control register (PFCR), and the PJ4DDR bit settings.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO20_OE	TIOCA7_OE	PJ4DDR
PPG	PO20 output*	1	—	—
TPU	TIOCA7 output*	0	1	—
I/O port	PJ4 output*	0	0	1
	PJ4 input*	0	0	0

Note: \* Valid when PCJKE = 1.

I/O port	PJ3 output*	0	0	1
	PJ3 input*	0	0	0

Note: \* Valid when PCJKE = 1.

### (6) PJ2/PO18/TIOCC6/TCLKE

The pin function is switched as shown below according to the combination of register settings for PPG and TPU, setting of the port function control register (PFCR), and the PJ2DDR bit.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO18_OE	TIOCC6_OE	PJ2DDR
PPG	PO18 output*	1	—	—
TPU	TIOCC6 output*	0	1	—
I/O port	PJ2 output*	0	0	1
	PJ2 input*	0	0	0

Note: \* Valid when PCJKE = 1.

I/O port	PJ1 output*	0	0	1
	PJ1 input*	0	0	0

Note: \* Valid when PCJKE = 1.

### (8) PJ0/PO16/TIOCA6

The pin function is switched as shown below according to the combination of register settings for PPG and TPU, setting of the port function control register (PFCR), and the PJ0DDR bit settings.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO16_OE	TIOCA6_OE	PJ0DDR
PPG	PO16 output*	1	—	—
TPU	TIOCA6 output*	0	1	—
I/O port	PJ0 output*	0	0	1
	PJ0 input*	0	0	0

Note: \* Valid when PCJKE = 1.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO31_OE	TIOCB11_OE	PK7DDR
PPG	PO31 output*	1	—	—
TPU	TIOCB11 output*	0	1	—
I/O port	PK7 output*	0	0	1
	PK7 input*	0	0	0

Note: \* Valid when PCJKE = 1.

## (2) PK6/PO30/TIOCA11

The pin function is switched as shown below according to the combination of register setting of PPG and TPU, setting of the port function control register (PFCR), and the PK6DDR bit.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO30_OE	TIOCA11_OE	PK6DDR
PPG	PO30 output*	1	—	—
TPU	TIOCA11 output*	0	1	—
I/O port	PK6 output*	0	0	1
	PK6 input*	0	0	0

Note: \* Valid when PCJKE = 1.

I/O port	PK5 output*	0	0	1
	PK5 input*	0	0	0

Note: \* Valid when PCJKE = 1.

#### (4) PK4/PO28/TIOCA10

The pin function is switched as shown below according to the combination of register set PPG and TPU, setting of the port function control register (PFCR), and the PK4DDR bit

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO28_OE	TIOCA10_OE	PK4DDR
PPG	PO28 output*	1	—	—
TPU	TIOCA10 output*	0	1	—
I/O port	PK4 output*	0	0	1
	PK4 input*	0	0	0

Note: \* Valid when PCJKE = 1.

I/O port	PK3 output*	0	0	1
	PK3 input*	0	0	0

Note: \* Valid when PCJKE = 1.

### (6) PK2/PO26/TIOCC9

The pin function is switched as shown below according to the combination of register settings for PPG and TPU, setting of the port function control register (PFCR), and the PK2DDR bit.

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO26_OE	TIOCC9_OE	PK2DDR
PPG	PO26 output*	1	—	—
TPU	TIOCC9 output*	0	1	—
I/O port	PK2 output*	0	0	1
	PK2 input*	0	0	0

Note: \* Valid when PCJKE = 1.

I/O port	PK1 output*	0	0	1
	PK1 input*	0	0	0

Note: \* Valid when PCJKE = 1.

### (8) PK0/PO24/TIOCA9

The pin function is switched as shown below according to the combination of register set PPG and TPU, setting of the port function control register (PFCR), and the PK0DDR bit

Module Name	Pin Function	Setting		
		PPG	TPU	I/O Port
		PO24_OE	TIOCA9_OE	PK0DDR
PPG	PO24 output*	1	—	—
TPU	TIOCA9 output*	0	1	—
I/O port	PK0 output*	0	0	1
	PK0 input*	0	0	0

Note: \* Valid when PCJKE = 1.



I/O port	PM4 output	0	1
	PM4 input (initial value)	0	0

### (2) PM3

The pin function is switched as shown below according to the combination of the PM3D setting.

Module Name	Pin Function	Setting	
		I/O Port	PM3DDR
I/O port	PM3 output	1	
	PM3 input (initial value)	0	

### (3) PM2

The pin function is switched as shown below according to the combination of the PM2D setting.

Module Name	Pin Function	Setting	
		I/O Port	PM2DDR
I/O port	PM2 output	1	
	PM2 input (initial value)	0	

### (5) PM0/TxD6

The pin function is switched as shown below according to the combination of the SCI register setting and PM0DDR bit setting.

Module Name	Pin Function	Setting	
		SCI	I/O Port
		TxD6_OE	PM0DDR
SCI	TxD6 output	1	—
I/O port	PM0 output	0	1
	PM0 input (initial value)	0	0

	SDA0_OE	SDA0		DMDR_1.DACKE = 1 ICCRA.ICE = 1
5	$\overline{\text{ETEND1A\_OE}}$	$\overline{\text{ETEND1}}$	PFCR8.EDMAS1[A,B] = 00	SYSCR.EXPE = 1, EDMDR_1.ETENDE = 1
	$\overline{\text{TEND1A\_OE}}$	$\overline{\text{TEND1}}$	PFCR7.DMAS1[A,B] = 00	DMDR_1.TENDE = 1
	SCL1_OE	SCL1		ICCRA.ICE = 1
4	TxD5_OE	TxD5		SCR.TE = 1, IrCR.IrE = 0
	IrTxD_OE	IrTxD		SCR.TE = 1, IrCR.IrE = 1
	SDA1_OE	SDA1		ICCRA.ICE = 1
3	$\overline{\text{EDRAK0\_OE}}$	$\overline{\text{EDRAK0}}$	PFCR8.EDMAS0[A,B] = 00	SYSCR.EXPE = 1, EDMDR_0.EDRAKE = 1
2	$\overline{\text{EDACK0A\_OE}}$	$\overline{\text{EDACK0}}$	PFCR8.EDMAS0[A,B] = 00	SYSCR.EXPE = 1, EDACR_0. EDMDR_0.EDACKE = 1
	$\overline{\text{DACK0A\_OE}}$	$\overline{\text{DACK0}}$	PFCR7.DMAS0[A,B] = 00	DMAC.DACR_0.AMS = 1, DMDR_0.DACKE = 1
	SCK2_OE	SCK2		When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 w SMR.GM = 0, SCR.CKE [1, 0], while SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 w SMR.C/A = 0, SCR.CKE [1, 0], while SMR.C/A = 1, SCR.CKE
1	$\overline{\text{ETEND0A\_OE}}$	$\overline{\text{ETEND0}}$	PFCR8.EDMAS0[A,B] = 00	SYSCR.EXPE = 1, EDMDR_0.ETENDE = 1
	$\overline{\text{TEND0A\_OE}}$	$\overline{\text{TEND0}}$	PFCR7.DMAS0[A,B] = 00	DMDR_0.TENDE = 1
0	TxD2_OE	TxD2		SCR.TE = 1

	PO6_OE	PO6	NDERL.NDER6 = 1
5	TIOCA4_OE	TIOCA4	TPU.TIOR_4.IOA3 = 0, TPU.TIOR_4.IOA[1,0] = 01/10/11
	PO5_OE	PO5	NDERL.NDER5 = 1
4	TIOCB4_OE	TIOCB4	TPU.TIOR_4.IOB3 = 0, TPU.TIOR_4.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] = 01/10/11 SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE [1, 0] = 01/10/11 SMR.C/A = 1, SCR.CKE 1 = 0
	PO4_OE	PO4	NDERL.NDER4 = 1
3	TIOCD3_OE	TIOCD3	TPU.TMDR.BFB = 0, TPU.TIORL_3.IOD3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
	PO3_OE	PO3	NDERL.NDER3 = 1
2	TIOCC3_OE	TIOCC3	TPU.TMDR.BFA = 0, TPU.TIORL_3.IOC3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
	TMO0_OE	TMO0	TMR.TCSR_0.OS[3,2] = 01/10/11 TMR.TCSR_0.OS[1,0] = 01/10/11
	TxD0_OE	TxD0	SCR.TE = 1
	PO2_OE	PO2	NDERL.NDER2 = 1

SMR.GM = 1  
 When SCMR.SMIF = 0:  
 SCR.TE = 1 or SCR.RE = 1 while  
 SMR.C/A = 0, SCR.CKE [1, 0] = 0  
 SMR.C/A = 1, SCR.CKE 1 = 0

		PO0_OE	PO0		NDERL.NDER0 = 1
P3	7	EDRAK3_OE	EDRAK3	PFCR8.EDMAS3[A,B] = 00	SYSCR.EXPE = 1, EDMDR_3.ED
		TIOCB2_OE	TIOCB2		TPU.TIOR_2.IOB3 = 0, TPU.TIOR_2.IOB[1,0] = 01/10/11
		PO15_OE	PO15		NDERH.NDER15 = 1
	6	EDRAK2_OE	EDRAK2	PFCR8.EDMAS2[A,B] = 00	SYSCR.EXPE = 1, EDMDR_2.ED
		TIOCA2_OE	TIOCA2		TPU.TIOR_2.IOA3 = 0, TPU.TIOR_2.IOA[1,0] = 01/10/11
		PO14_OE	PO14		NDERH.NDER14 = 1
	5	EDACK3_OE	EDACK3	PFCR8.EDMAS3[A,B] = 00	SYSCR.EXPE = 1, EDACR_3.AM EDMDR_3.EDACKE = 1
		DACK1B_OE	DACK1	PFCR7.DMAS1[A,B] = 01	DMAC.DACR.AMS = 1, DMDR_1.DACKE = 1
		TIOCB1_OE	TIOCB1		TPU.TIOR_1.IOB3 = 0, TPU.TIOR_1.IOB[1,0] = 01/10/11
		PO13_OE	PO13		NDERH.NDER13 = 1

				TPU.TIORL_0.IOD3 = 0, TPU.TIORL_0.IOD[1,0] = 01/10/11
	PO11_OE	PO11		NDERH.NDER11 = 1
2	$\overline{\text{EDACK2}}_{\text{OE}}$	$\overline{\text{EDACK2}}$	PFCR8.EDMAS2[A,B] = 00	SYSCR.EXPE = 1, EDACR_2.AMS EDMDR_2.EDACKE = 1
	$\overline{\text{DACK0B}}_{\text{OE}}$	$\overline{\text{DACK0}}$	PFCR7.DMAS0[A,B] = 01	DMAC.DACR.AMS = 1, DMDR_0.DACKE = 1
	TIOCC0_OE	TIOCC0		TPU.TMDR.BFA = 0, TPU.TIORL_0.IOC3 = 0, TPU.TIORL_0.IOD[1,0] = 01/10/11
	PO10_OE	PO10		NDERH.NDER10 = 1
1	$\overline{\text{ETEND2}}_{\text{OE}}$	$\overline{\text{ETEND2}}$	PFCR8.EDMAS2[A,B] = 00	SYSCR.EXPE = 1, EDMDR_2.ETE = 00
	$\overline{\text{TEND0B}}_{\text{OE}}$	$\overline{\text{TEND0}}$	PFCR7.DMAS0[A,B] = 01	DMDR_0.TENDE = 1
	TIOCB0_OE	TIOCB0		TPU.TIORH_0.IOB3 = 0, TPU.TIORH_0.IOB[1,0] = 01/10/11
	PO9_OE	PO9		NDERH.NDER9 = 1
0	TIOCA0_OE	TIOCA0		TPU.TIORH_0.IOA3 = 0, TPU.TIORH_0.IOA[1,0] = 01/10/11
	PO8_OE	PO8		NDERH.NDER8 = 1
P6	5	$\overline{\text{EDACK1B}}_{\text{OE}}$	$\overline{\text{EDACK1}}$	PFCR8.EDMAS1[A,B] = 01 SYSCR.EXPE = 1, EDACR_1.AMS EDMDR_1.EDACKE = 1
		$\overline{\text{DACK3}}_{\text{OE}}$	$\overline{\text{DACK3}}$	PFCR7.DMAS3[A,B] = 01 DMAC.DACR_3.AMS = 1, DMDR_3.DACKE = 1
		TMO3_OE	TMO3	TMR.TCSR_3.OS[3,2] = 01/10/11 TMR.TCSR_3.OS[1,0] = 01/10/11

	$\overline{\text{TMO2\_OE}}$	$\overline{\text{TMO2}}$		TMR.TCSR_2.OS[3,2] = 01/10/11 TMR.TCSR_2.OS[1,0] = 01/10/11
	$\overline{\text{SCK4\_OE}}$	$\overline{\text{SCK4}}$		When SCMR.SMIF = 1: SCR.TE = 1 or SCR.RE = 1 while SMR.GM = 0, SCR.CKE [1, 0] = 0 SMR.GM = 1 When SCMR.SMIF = 0: SCR.TE = 1 or SCR.RE = 1 while SMR.C/A = 0, SCR.CKE [1, 0] = 0 SMR.C/A = 1, SCR.CKE 1 = 0
1	$\overline{\text{ETEND0B\_OE}}$	$\overline{\text{ETEND0}}$	PFCR8.EDMAS0[A,B] = 01	SYSR.EXPE = 1, EDMDR_0.ET
	$\overline{\text{TEND2\_OE}}$	$\overline{\text{TEND2}}$	PFCR7.DMAS2[A,B] = 01	DMDR_2.TENDE = 1
0	$\overline{\text{TxD4\_OE}}$	$\overline{\text{TxD4}}$		SCR.TE = 1
PA	7	$\overline{\text{B}\phi\text{\_OE}}$	$\overline{\text{B}\phi}$	PADDR.PA7DDR = 1, SCKCR.PS
	6	$\overline{\text{AH\_OE}}$	$\overline{\text{AH}}$	SYSR.EXPE = 1, MPXCR.MPXEn (n = 7 to 3) = 1
		$\overline{\text{BSB\_OE}}$	$\overline{\text{BS}}$	PFCR2.BSS = 1 SYSR.EXPE = 1, PFCR2.BSE =
		$\overline{\text{AS\_OE}}$	$\overline{\text{AS}}$	SYSR.EXPE = 1, PFCR2.ASOE
	5	$\overline{\text{RD\_OE}}$	$\overline{\text{RD}}$	SYSR.EXPE = 1
	4	$\overline{\text{LUB\_OE}}$	$\overline{\text{LUB}}$	SYSR.EXPE = 1, PFCR6.LHWF SRAMCR.BCSELn = 1
		$\overline{\text{LHWR\_OE}}$	$\overline{\text{LHWR}}$	SYSR.EXPE = 1, PFCR6.LHWF
	3	$\overline{\text{LLB\_OE}}$	$\overline{\text{LLB}}$	SYSR.EXPE = 1, SRAMCR.BC
		$\overline{\text{LLWR\_OE}}$	$\overline{\text{LLWR}}$	SYSR.EXPE = 1

6	$\overline{(\text{RD}/\text{WR})\text{-B\_OE}}$	$\overline{\text{RD}/\text{WR}}$	$\text{PFCR2.RDWRS} = 1$	$\text{SYSCR.EXPE} = 1, \text{PFCR2.RDWRS} = 1$ $\text{ASRAMCR.BCSELn} = 1$
	$\overline{\text{CS6D\_OE}}$	$\overline{\text{CS6}}$	$\text{PFCR1.CS6S[A,B]} = 11$	$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS6E} = 1$
5	$\overline{\text{CKE\_OE}}$	$\overline{\text{CKE}}$		$\text{SYSCR.EXPE} = 1, \text{DRAMCR.DRAMEN} = 1$ $\text{DRAMCR.DTYPE} = 1, \text{DRAMCR.DTYPEEN} = 1$
	$\overline{\text{OE\_OE}}$	$\overline{\text{OE}}$		$\text{SYSCR.EXPE} = 1, \text{DRAMCR.DRAMEN} = 1$ $\text{DRAMCR.DTYPE} = 0, \text{DRAMCR.DTYPEEN} = 0$
	$\overline{\text{CS5D\_OE}}$	$\overline{\text{CS5}}$	$\text{PFCR1.CS5S[A,B]} = 11$	$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS5E} = 1$
4	$\overline{\text{WE\_OE}}$	$\overline{\text{WE}}$		$\text{SYSCR.EXPE} = 1, \text{DRAMCR.DRAMEN} = 1$
	$\overline{\text{CS4B\_OE}}$	$\overline{\text{CS4}}$	$\text{PFCR1.CS4S[A,B]} = 01$	$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS4E} = 1$
3	$\overline{\text{CAS\_OE}}$	$\overline{\text{CAS}}$		$\text{SYSCR.EXPE} = 1, \text{DRAMCR.DRAMEN} = 1$ $\text{DRAMCR.DTYPE} = 1$
	$\overline{\text{CS3\_OE}}$	$\overline{\text{CS3}}$		$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS3E} = 1$
	$\overline{\text{CS7A\_OE}}$	$\overline{\text{CS7}}$	$\text{PFCR1.CS7S[A,B]} = 00$	$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS7E} = 1$
2	$\overline{\text{RAS\_OE}}$	$\overline{\text{RAS}}$		$\text{SYSCR.EXPE} = 1, \text{DRAMCR.DRAMEN} = 1$
	$\overline{\text{CS2A\_OE}}$	$\overline{\text{CS2}}$	$\text{PFCR2.CS2S} = 0$	$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS2E} = 1$
	$\overline{\text{CS6A\_OE}}$	$\overline{\text{CS6}}$	$\text{PFCR1.CS6S[A,B]} = 00$	$\text{SYSCR.EXPE} = 1, \text{PFCR0.CS6E} = 1$



	0	$\overline{\text{CS0\_OE}}$	$\overline{\text{CS0}}$		SYSCR.EXPE = 1, PFCR0.CS0E
		$\overline{\text{CS4A\_OE}}$	$\overline{\text{CS4}}$	PFCR1.CS4S[A,B] = 00	SYSCR.EXPE = 1, PFCR0.CS4E
		$\overline{\text{CS5B\_OE}}$	$\overline{\text{CS5}}$	PFCR1.CS5S[A,B] = 01	SYSCR.EXPE = 1, PFCR0.CS5E
PC	3	$\overline{\text{LLCAS\_OE}}$	LLCAS		SYSCR.EXPE = 1, DRAMCR.DR DRAMCR.DTYPE = 0
		$\overline{\text{DQMLL\_OE}}$	DQMLL		SYSCR.EXPE = 1, DRAMCR.DR DRAMCR.DTYPE = 1
	2	$\overline{\text{LUCAS\_OE}}$	LUCAS		SYSCR.EXPE = 1, ABWCR.[ABWH2,ABWL2] = x0/0 DRAMCR.DRAME = 1, DRAMCR.DTYPE = 0
		$\overline{\text{DQMLU\_OE}}$	DQMLU		SYSCR.EXPE = 1, ABWCR.[ABWH2,ABWL2] = x0/0 DRAMCR.DRAME = 1, DRAMCR.DTYPE = 1
PD	7	A7_OE	A7		SYSCR.EXPE = 1, PDDDR.PD7D
	6	A6_OE	A6		SYSCR.EXPE = 1, PDDDR.PD6D
	5	A5_OE	A5		SYSCR.EXPE = 1, PDDDR.PD5D
	4	A4_OE	A4		SYSCR.EXPE = 1, PDDDR.PD4D
	3	A3_OE	A3		SYSCR.EXPE = 1, PDDDR.PD3D
	2	A2_OE	A2		SYSCR.EXPE = 1, PDDDR.PD2D
	1	A1_OE	A1		SYSCR.EXPE = 1, PDDDR.PD1D
	0	A0_OE	A0		SYSCR.EXPE = 1, PDDDR.PD0D

	0	A8_OE	A8	SYSCR.EXPE = 1, PEDDR.PE0DD
PF	7	A23_OE	A23	SYSCR.EXPE = 1, PFCR4.A23E =
	6	A22_OE	A22	SYSCR.EXPE = 1, PFCR4.A22E =
	5	A21_OE	A21	SYSCR.EXPE = 1, PFCR4.A21E =
	4	A20_OE	A20	SYSCR.EXPE = 1, PFCR4.A20E =
	3	A19_OE	A19	SYSCR.EXPE = 1, PFCR4.A19E =
	2	A18_OE	A18	SYSCR.EXPE = 1, PFCR4.A18E =
	1	A17_OE	A17	SYSCR.EXPE = 1, PFCR4.A17E =
	0	A16_OE	A16	SYSCR.EXPE = 1, PFCR4.A16E =
PH	7	D7_E	D7	SYSCR.EXPE = 1
	6	D6_E	D6	SYSCR.EXPE = 1
	5	D5_E	D5	SYSCR.EXPE = 1
	4	D4_E	D4	SYSCR.EXPE = 1
	3	D3_E	D3	SYSCR.EXPE = 1
	2	D2_E	D2	SYSCR.EXPE = 1
	1	D1_E	D1	SYSCR.EXPE = 1
	0	D0_E	D0	SYSCR.EXPE = 1
PI	7	D15_E	D15	SYSCR.EXPE = 1, ABWCR.ABW[H
	6	D14_E	D14	SYSCR.EXPE = 1, ABWCR.ABW[H
	5	D13_E	D13	SYSCR.EXPE = 1, ABWCR.ABW[H
	4	D12_E	D12	SYSCR.EXPE = 1, ABWCR.ABW[H
	3	D11_E	D11	SYSCR.EXPE = 1, ABWCR.ABW[H
	2	D10_E	D10	SYSCR.EXPE = 1, ABWCR.ABW[H
	1	D9_E	D9	SYSCR.EXPE = 1, ABWCR.ABW[H
	0	D8_E	D8	SYSCR.EXPE = 1, ABWCR.ABW[H

	PO 21_OE	PO21	NDERL_1.NDER21 = 1
4	TIOCA7_OE	TIOCA7	TPU.TIOR_7.IOA3 = 0, TPU.TIOR_7.IOA[1,0] = 01/10/11
	PO 20_OE	PO20	NDERL_1.NDER20 = 1
3	TIOCD6_OE	TIOCD6	TPU.TMDR_6.BFB = 0, TPU.TIORL_6.IOD3 = 0 TPU.TIORL_6.IOD[1,0] = 01/10/11
	PO 19_OE	PO19	NDERL_1.NDER19 = 1
2	TIOCC6_OE	TIOCC6	TPU.TMDR_6.BFA = 0, TPU.TIORL_6.IOC3 = 0 TPU.TIORL_6.IOC[1,0] = 01/10/11
	PO 18_OE	PO18	NDERL_1.NDER18 = 1
1	TIOCB6_OE	TIOCB6	TPU.TIORH_6.IOB3 = 0, TPU.TIORH_6.IOB[1,0] = 01/10/11
	PO 17_OE	PO17	NDERL_1.NDER17 = 1
0	TIOCA6_OE	TIOCA6	TPU.TIORH_6.IOA3 = 0, TPU.TIORH_6.IOA[1,0] = 01/10/11
	PO 16_OE	PO16	NDERL_1.NDER16 = 1

		PO29_OE	PO29		NDERH_1.NDER29 = 1
4		TIOCA10_OE	TIOCA10		TPU.TIOR_10.IOA3 = 0, TPU.TIOR_10.IOA[1,0] = 01/10/11
		PO28_OE	PO28		NDERH_1.NDER28 = 1
3		TIOCD9_OE	TIOCD9		TPU.TMDR_9.BFB = 0, TPU.TIORL_9.IOD3 = 0  TPU.TIORL_9.IOD[1,0] = 01/10/11
		PO27_OE	PO27		NDERH_1.NDER27 = 1
2		TIOCC9_OE	TIOCC9		TPU.TMDR_9.BFA = 0, TPU.TIORL_9.IOC3 = 0  TPU.TIORL_9.IOC[1,0] = 01/10/11
		PO26_OE	PO26		NDERH_1.NDER26 = 1
1		TIOCB9_OE	TIOCB9		TPU.TIORH_9.IOB3 = 0, TPU.TIORH_9.IOB[1,0] = 01/10/11
		PO25_OE	PO25		NDERH_1.NDER25 = 1
0		TIOCA9_OE	TIOCA9		TPU.TIORH_9.IOA3 = 0, TPU.TIORH_9.IOA[1,0] = 01/10/11
		PO24_OE	PO24		NDERH_1.NDER24 = 1
PM	4	—	—	—	—
	3	—	—	—	—
	2	—	—	—	—
	1	—	—	—	—
	0	TxD6_OE	TxD6		SCR.TE = 1

- Port function control register 6 (PFCR6)
- Port function control register 7 (PFCR7)
- Port function control register 8 (PFCR8)
- Port function control register 9 (PFCR9)
- Port function control register A (PFCRA)
- Port function control register B (PFCRB)
- Port function control register C (PFCRC)
- Port function control register D (PFCRD)

### 13.3.1 Port Function Control Register 0 (PFCR0)

PFCR0 enables/disables the  $\overline{CS}$  output.

Bit	7	6	5	4	3	2	1	
Bit Name	CS7E	CS6E	CS5E	CS4E	CS3E	CS2E	CS1E	
Initial Value	0	0	0	0	0	0	0	U
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.

### 13.3.2 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: Specifies pin PB6 as $\overline{CS6}$ -D output

00: Specifies pin PB0 as  $\overline{CS4}$ -A output  
 01: Specifies pin PB4 as  $\overline{CS4}$ -B output  
 10: Setting prohibited  
 11: Setting prohibited

Note: \* If multiple  $\overline{CS}$  outputs are specified to a single pin according to the  $\overline{CSn}$  output select bits (n = 4 to 7), multiple  $\overline{CS}$  signals are output from the pin. For details see section 9.5.3, Chip Select Signals.

### 13.3.3 Port Function Control Register 2 (PFCR2)

PFCR2 selects the  $\overline{CS}$  output pin, enables/disables bus control I/O, and selects the bus control pins.

Bit	7	6	5	4	3	2	1
Bit Name	—	CS2S	BSS	BSE	RDWRS	RDWRE	ASOE
Initial Value	0	0	0	0	0	0	1
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.

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	RDWRS* <sup>2</sup>	0	R/W	$RD/\overline{WR}$ Output Pin Select Selects the output pin for $RD/\overline{WR}$ 0: Specifies pin PA1 as $RD/\overline{WR}$ -A output pin 1: Specifies pin PB6 as $RD/\overline{WR}$ -B output pin
2	RDWRE* <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	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  $\overline{CS}_n$  output select bit ( $n = 2$ ), multiple  $\overline{CS}$  signals are output from the pin. For details, see section 9.5.3, Chip Select Signals.
  2. If an area is specified as a byte control SDRAM space, the pin functions as  $RD/\overline{WR}$  output regardless of the RDWRE bit value.



Bit	Bit Name	Value	R/W	Description
7	A23E	0	R/W	Address A23 Enable Enables/disables the address output (A23) 0: Disables the A23 output 1: Enables the A23 output
6	A22E	0	R/W	Address A22 Enable Enables/disables the address output (A22) 0: Disables the A22 output 1: Enables the A22 output
5	A21E	0	R/W	Address A21 Enable Enables/disables the address output (A21) 0: Disables the A21 output 1: Enables the A21 output
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

---

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 changes depending on the operating mode. 1 in on-chip ROM mode and 0 in on-chip ROM enabled mode.

Bit	Bit Name	Value	R/W	Description
7	—	1	R/W	Reserved This bit is always read as 1. The write value s always be 1.
6	LHWROE	1	R/W	$\overline{\text{LHWR}}$ Output Enable Enables/disables $\overline{\text{LHWR}}$ output (valid in exten 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 s 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 clock input pins. 1: Specifies pins P14 to P17 as external clock pins.
2, 1	—	All 0	R/W	Reserved These bits are always read as 0. The write va always be 0.
0	ADTRG0S	0	R/W	$\overline{\text{ADTRG0S}}$ Input Pin Select Selects the external trigger input pins of A/D C 0: Specifies pin P13 as $\overline{\text{ADTRG0}}$ -A input pin. 1: Specifies pin PB6 as $\overline{\text{ADTRG0}}$ -B input pin.

7	DMAS3A	0	R/W	DMAC control pin select
6	DMAS3B	0	R/W	Selects the I/O port to control DMAC_3. 00: DMAC_3 control pins are disabled. 01: Specifies pins P63 to P65 as DMAC control pins. 10: Setting prohibited 11: Setting prohibited
5	DMAS2A	0	R/W	DMAC control pin select
4	DMAS2B	0	R/W	Selects the I/O port to control DMAC_2. 00: DMAC_2 control pins are disabled. 01: Specifies pins P60 to P62 as DMAC control pins. 10: Setting prohibited 11: Setting prohibited
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

Bit	Bit Name	Value	R/W	Description
7	EDMAS3A	0	R/W	EXDMAC Control Pin Select
6	EDMAS3B	0	R/W	Select the I/O port to control EXDMAC_3. 00: Specify pins P33 to P35 and P37 as EXDMAC pin 01: Setting prohibited 10: Setting prohibited 11: Setting prohibited
5	EDMAS2A	0	R/W	EXDMAC Control Pin Select
4	EDMAS2B	0	R/W	Select the I/O port to control EXDMAC_2. 00: Specify pins P30 to P32 and P36 as EXDMAC pin 01: Setting prohibited 10: Setting prohibited 11: Setting prohibited
3	EDMAS1A	0	R/W	EXDMAC Control Pin Select
2	EDMAS1B	0	R/W	Select the I/O port to control EXDMAC_1. 00: Specify pins P14 to P17 as EXDMAC control pin 01: Specify pins P63 to P65 as EXDMAC control pin 10: Setting prohibited 11: Setting prohibited

### 13.3.8 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	
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 capture 1: Specifies P27 as input capture input and P26 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 capture 1: Specifies P24 as input capture input and P25 output compare

				capture 1: Specifies P23 as input capture input and P2 output compare
3	TPUMS2	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA2 function 0: Specifies P36 as output compare output and capture 1: Specifies P37 as input capture input and P3 output compare
2	TPUMS1	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA1 function 0: Specifies P34 as output compare output and capture 1: Specifies P35 as input capture input and P3 output compare
1	TPUMS0A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA0 function 0: Specifies P30 as output compare output and capture 1: Specifies P31 as input capture input and P3 output compare
0	TPUMS0B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC0 function 0: Specifies P32 as output compare output and capture 1: Specifies P33 as input capture input and P3 output compare

Bit	Bit Name	Value	R/W	Description
7	TPUMS11	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCA11 function.</p> <p>0: Specifies pin PK6 as output compare output and input capture</p> <p>1: Specifies PK7 as input capture input and PK6 output compare</p>
6	TPUMS10	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCA10 function.</p> <p>0: Specifies PK4 as output compare output and input capture</p> <p>1: Specifies PK5 as input capture input and PK4 output compare</p>
5	TPUMS9A	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCA9 function.</p> <p>0: Specifies PK0 as output compare output and input capture</p> <p>1: Specifies PK1 as input capture input and PK0 output compare</p>
4	TPUMS9B	0	R/W	<p>TPU I/O Pin Multiplex Function Select</p> <p>Selects TIOCC9 function.</p> <p>0: Specifies PK2 as output compare output and input capture</p> <p>1: Specifies PK3 as input capture input and PK2 output compare</p>



				0: Specifies PJ4 as output compare output and capture 1: Specifies PJ5 as input capture input and PJ4 compare
1	TPUMS6A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA6 function. 0: Specifies PJ0 as output compare output and capture 1: Specifies PJ1 as input capture input and PJ0 compare
0	TPUMS6B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC6 function. 0: Specifies PJ2 as output compare output and capture 1: Specifies PJ3 as input capture input and PJ2 compare

- H8SX/1665 Group

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 always be 0.

- H8SX/1665M Group

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	ITS14	0	R/W	LVD Interrupt Select This bit allows or prohibits LVD interrupt. 0: Prohibits LVD interrupt 1: Allows LVD interrupt
5, 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value always be 0.
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

---

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

---

Bit	Bit Name	Value	R/W	Description
7	ITS7	0	R/W	<p><math>\overline{\text{IRQ7}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ7}}</math>.</p> <p>0: Selects pin P17 as <math>\overline{\text{IRQ7}}</math>-A input</p> <p>1: Selects pin P57 as <math>\overline{\text{IRQ7}}</math>-B input</p>
6	ITS6	0	R/W	<p><math>\overline{\text{IRQ6}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ6}}</math>.</p> <p>0: Selects pin P16 as <math>\overline{\text{IRQ6}}</math>-A input</p> <p>1: Selects pin P56 as <math>\overline{\text{IRQ6}}</math>-B input</p>
5	ITS5	0	R/W	<p><math>\overline{\text{IRQ5}}</math> Pin Select</p> <p>Selects an input pin for <math>\overline{\text{IRQ5}}</math>.</p> <p>0: Selects pin P15 as <math>\overline{\text{IRQ5}}</math>-A input</p> <p>1: Selects pin P55 as <math>\overline{\text{IRQ5}}</math>-B input</p>
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 input</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 input</p>

0	ITSO	0	R/W	$\overline{\text{IRQ0}}$ Pin Select
				Selects an input pin for $\overline{\text{IRQ0}}$ .
				0: Selects pin P10 as $\overline{\text{IRQ0}}$ -A input
				1: Selects pin P50 as $\overline{\text{IRQ0}}$ -B input

### 13.3.12 Port Function Control Register D (PFCRD)

PFCRD enables/disables the pin functions of ports J and K.

Bit	7	6	5	4	3	2	1
Bit Name	PCJKE*	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0
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	PCJKE*	0	R/W	Ports J and K Enable Enables/disables ports J and K. 0: Ports J and K are disabled 1: Ports J and K are enabled
6 to 0	—	0	R/W	Reserved  These bits are always read as 0 and cannot be modified. The initial values should not be changed.

Note: \* This bit is valid during single-chip mode. The initial value should not be changed for the single-chip mode.

3. When a pin is used as an output, data to be output from the pin will be latched as the pin level if the input function corresponding to the pin is enabled. To use the pin as an output, disable the input function for the pin by setting ICR.

### 13.4.2 Notes on Port Function Control Register (PFCR) Settings

1. Port function controller controls the I/O port.  
Before enabling a port function, select the input/output destination.
2. When changing input pins, this LSI may malfunction due to the internal edge generated by the pin level difference before and after the change.
  - To change input pins, the following procedure must be performed.
    - A. Disable the input function by the corresponding on-chip peripheral module settings
    - B. Select another input pin by PFCR
    - C. Enable its input function by the corresponding on-chip peripheral module settings
3. If a pin function has both a select bit that modifies the input/output destination and an enable bit that enables the pin function, first specify the input/output destination by the select bit and then enable the pin function by the enable bit.
4. Modifying of the PCJKE bit should be done in the initial setting right after activation. Do not change the PCJKE bit setting once it is set.
5. Do not change the PCJKE bit setting once it is set.

## 14.1 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 can be generated (unit 0 only)
- Module stop state can be set

(TGR)	TGRB_0	TGRB_1	TGRB_2	TGRB_3	TGRB_4	TGRB_5
General registers/ buffer registers	TGRC_0 TGRD_0	—	—	TGRC_3 TGRD_3	—	—
I/O pins	TIOCA0 TIOCB0 TIOCC0 TIOCD0	TIOCA1 TIOCB1	TIOCA2 TIOCB2	TIOCA3 TIOCB3 TIOCC3 TIOCD3	TIOCA4 TIOCB4	TIOCA5 TIOCB5
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	—	—
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



Interrupt sources	TGRB_0 compare match or input capture	TGRB_1 compare match or input capture	TGRB_2 compare match or input capture	TGRB_3 compare match or input capture	TGRB_4 compare match or input capture	TGRB_5 compare match or input capture
Interrupt sources	5 sources	4 sources	4 sources	5 sources	4 sources	4 sources
	Compare match or input capture 0A	Compare match or input capture 1A	Compare match or input capture 2A	Compare match or input capture 3A	Compare match or input capture 4A	Compare match or input capture 5A
	Compare match or input capture 0B	Compare match or input capture 1B	Compare match or input capture 2B	Compare match or input capture 3B	Compare match or input capture 4B	Compare match or input capture 5B
	Compare match or input capture 0C			Compare match or input capture 3C		
	Compare match or input capture 0D			Compare match or input capture 3D		
	Overflow	Overflow	Overflow	Overflow	Overflow	Overflow
		Underflow	Underflow		Underflow	Underflow

[Legend]

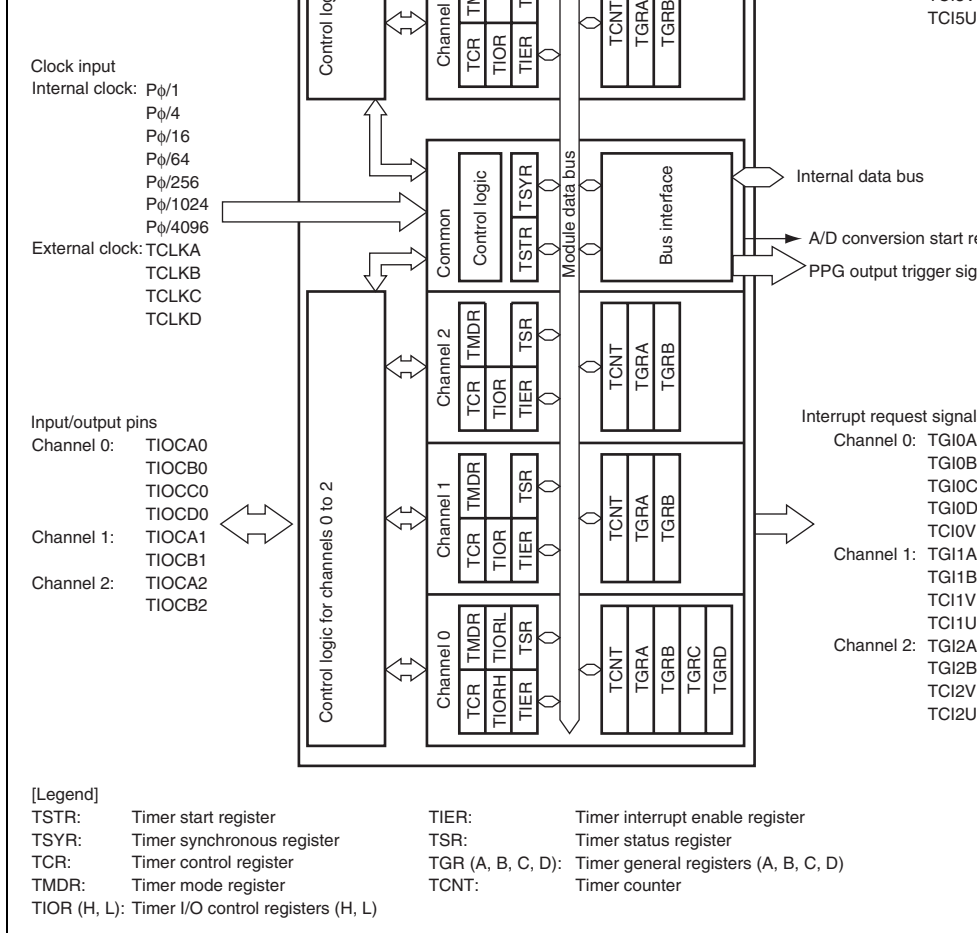
- : Possible
- : Not possible

(TGR)	TGRB_6	TGRB_7	TGRB_8	TGRB_9	TGRB_10	TGRB_11
General registers/ buffer registers	TGRC_6 TGRD_6	—	—	TGRC_9 TGRD_9	—	—
I/O pins	TIOCA6 TIOCB6 TIOCC6 TIOCD6	TIOCA7 TIOCB7	TIOCA8 TIOCB8	TIOCA9 TIOCB9 TIOCC9 TIOCD9	TIOCA10 TIOCB10	TIOCA11 TIOCB11
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	—	—
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

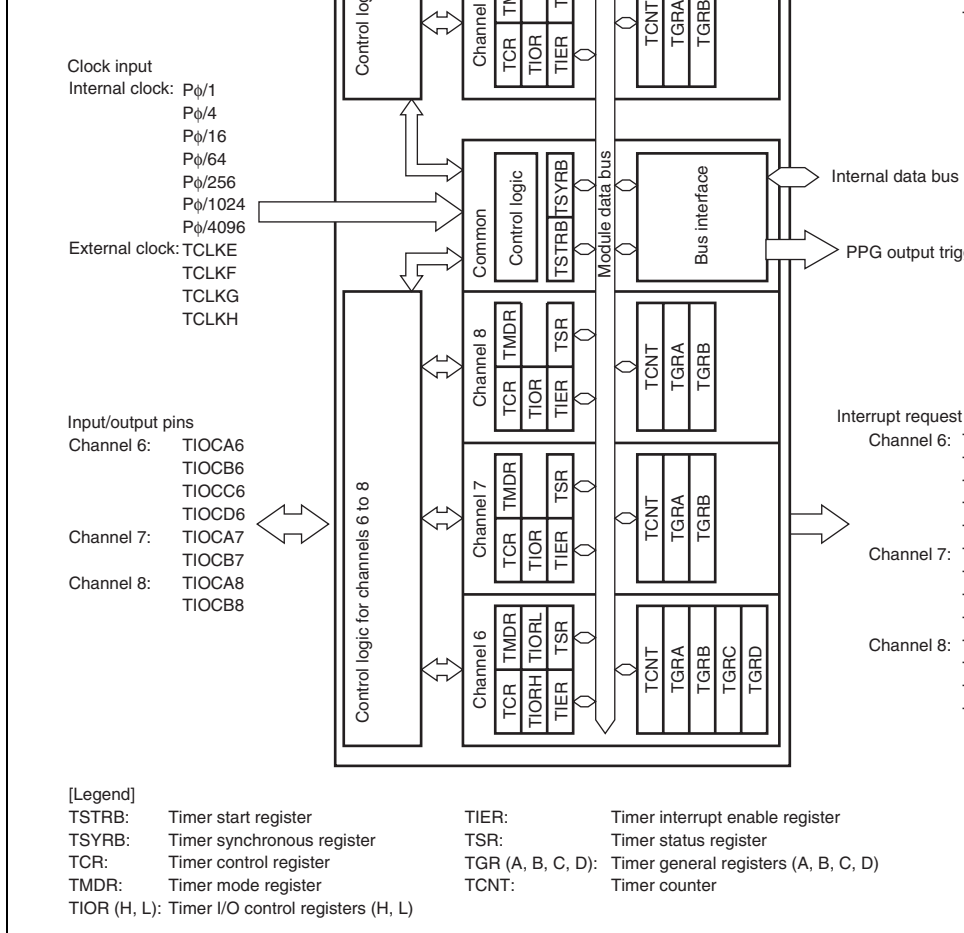
Interrupt sources	5 sources	4 sources	4 sources	5 sources	4 sources	4
	Compare match or input capture 6A	Compare match or input capture 7A	Compare match or input capture 8A	Compare match or input capture 9A	Compare match or input capture 10A	C
	Compare match or input capture 6B	Compare match or input capture 7B	Compare match or input capture 8B	Compare match or input capture 9B	Compare match or input capture 10B	C
	Compare match or input capture 6C			Compare match or input capture 9C		C
	Compare match or input capture 6D			Compare match or input capture 9D		C
	Overflow	Overflow	Overflow	Overflow	Overflow	C
		Underflow	Underflow		Underflow	U

[Legend]

- : Possible
- : Not possible



**Figure 14.1 Block Diagram of TPU (Unit 0)**



**Figure 14.2 Block Diagram of TPU (Unit 1)**

	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 o
	TIOCB0	I/O	TGRB_0 input capture input/output compare output/PWM o
	TIOCC0	I/O	TGRC_0 input capture input/output compare output/PWM o
	TIOCD0	I/O	TGRD_0 input capture input/output compare output/PWM o
1	TIOCA1	I/O	TGRA_1 input capture input/output compare output/PWM o
	TIOCB1	I/O	TGRB_1 input capture input/output compare output/PWM o
2	TIOCA2	I/O	TGRA_2 input capture input/output compare output/PWM o
	TIOCB2	I/O	TGRB_2 input capture input/output compare output/PWM o
3	TIOCA3	I/O	TGRA_3 input capture input/output compare output/PWM o
	TIOCB3	I/O	TGRB_3 input capture input/output compare output/PWM o
	TIOCC3	I/O	TGRC_3 input capture input/output compare output/PWM o
	TIOCD3	I/O	TGRD_3 input capture input/output compare output/PWM o
4	TIOCA4	I/O	TGRA_4 input capture input/output compare output/PWM o
	TIOCB4	I/O	TGRB_4 input capture input/output compare output/PWM o
5	TIOCA5	I/O	TGRA_5 input capture input/output compare output/PWM o
	TIOCB5	I/O	TGRB_5 input capture input/output compare output/PWM o

6	TIOCA6	I/O	TGRA_6 input capture input/output compare output/PWM
	TIOCB6	I/O	TGRB_6 input capture input/output compare output/PWM
	TIOCC6	I/O	TGRC_6 input capture input/output compare output/PWM
7	TIOCD6	I/O	TGRD_6 input capture input/output compare output/PWM
	TIOCA7	I/O	TGRA_7 input capture input/output compare output/PWM
	TIOCB7	I/O	TGRB_7 input capture input/output compare output/PWM
8	TIOCA8	I/O	TGRA_8 input capture input/output compare output/PWM
	TIOCB8	I/O	TGRB_8 input capture input/output compare output/PWM
9	TIOCA9	I/O	TGRA_9 input capture input/output compare output/PWM
	TIOCB9	I/O	TGRB_9 input capture input/output compare output/PWM
	TIOCC9	I/O	TGRC_9 input capture input/output compare output/PWM
	TIOCD9	I/O	TGRD_9 input capture input/output compare output/PWM
10	TIOCA10	I/O	TGRA_10 input capture input/output compare output/PWM
	TIOCB10	I/O	TGRB_10 input capture input/output compare output/PWM
11	TIOCA11	I/O	TGRA_11 input capture input/output compare output/PWM
	TIOCB11	I/O	TGRB_11 input capture input/output compare output/PWM

- 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 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)

- Common Registers:
  - Timer start register (TSTR)
  - Timer synchronous register (TSYR)

Unit 1:

- Channel 6:
  - Timer control register\_6 (TCR\_6)
  - Timer mode register\_6 (TMDR\_6)
  - Timer I/O control register H\_6 (TIORH\_6)
  - Timer I/O control register L\_6 (TIORL\_6)
  - Timer interrupt enable register\_6 (TIER\_6)
  - Timer status register\_6 (TSR\_6)
  - Timer counter\_6 (TCNT\_6)
  - Timer general register A\_6 (TGRA\_6)
  - Timer general register B\_6 (TGRB\_6)
  - Timer general register C\_6 (TGRC\_6)
  - Timer general register D\_6 (TGRD\_6)

- Channel 8:
  - Timer control register\_8 (TCR\_8)
  - Timer mode register\_8 (TMDR\_8)
  - Timer I/O control register\_8 (TIOR\_8)
  - Timer interrupt enable register\_8 (TIER\_8)
  - Timer status register\_8 (TSR\_8)
  - Timer counter\_8 (TCNT\_8)
  - Timer general register A\_8 (TGRA\_8)
  - Timer general register B\_8 (TGRB\_8)
  
- Channel 9:
  - Timer control register\_9 (TCR\_9)
  - Timer mode register\_9 (TMDR\_9)
  - Timer I/O control register H\_9 (TIORH\_9)
  - Timer I/O control register L\_9 (TIORL\_9)
  - Timer interrupt enable register\_9 (TIER\_9)
  - Timer status register\_9 (TSR\_9)
  - Timer counter\_9 (TCNT\_9)
  - Timer general register A\_9 (TGRA\_9)
  - Timer general register B\_9 (TGRB\_9)
  - Timer general register C\_9 (TGRC\_9)
  - Timer general register D\_9 (TGRD\_9)

- Channel 11:
  - Timer control register\_11 (TCR\_11)
  - Timer mode register\_11 (TMDR\_11)
  - Timer I/O control register\_11 (TIOR\_11)
  - Timer interrupt enable register\_11 (TIER\_11)
  - Timer status register\_11 (TSR\_11)
  - Timer counter\_11 (TCNT\_11)
  - Timer general register A\_11 (TGRA\_11)
  - Timer general register B\_11 (TGRB\_11)
  
- Common Registers:
  - Timer start register (TSTRB)
  - Timer synchronous register (TSYRB)

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 sources. See tables 14.4 and 14.5 for details.
5	CCLR0	0	R/W	
4	CKEG1	0	R/W	Clock Edge 1 and 0 These bits select the input clock edge. For details, see table 14.6. When the input clock is counted using both rising and falling edges, the input clock period is halved (e.g. $P_{\phi} = 2 \times P_{\phi}/2$ 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. Clock edge selection is valid when the input clock frequency is greater than or equal to the input clock period or slower. This setting is ignored if the input clock is selected or when overflow/underflow of another channel is selected.
3	CKEG0	0	R/W	
2	TPSC2	0	R/W	Timer Prescaler 2 to 0
1	TPSC1	0	R/W	These bits select the TCNT counter clock. The clock source can be selected independently for each channel. See tables 14.7 to 14.12 for details. To select the input clock as the clock source, the DDR bit and ICR1 bit of the corresponding pin should be set to 0 and 1, respectively. For details, see section 13, I/O Ports.
0	TPSC0	0	R/W	

			synchronous operation* <sup>1</sup>
1	0	0	TCNT clearing disabled
1	0	1	TCNT cleared by TGRC compare match capture* <sup>2</sup>
1	1	0	TCNT cleared by TGRD compare match capture* <sup>2</sup>
1	1	1	TCNT cleared by counter clearing for a channel performing synchronous clearing in 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 14.5 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 capture
	0	1	0	TCNT cleared by TGRB compare match capture
	0	1	1	TCNT cleared by counter clearing for a channel performing synchronous clearing in 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 14.7 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 14.8 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.

1	1	0	External clock: counts on TCLKC pin in
1	1	1	Internal clock: counts on P $\phi$ /1024

Note: This setting is ignored when channel 2 is in phase counting mode.

**Table 14.10 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 in
	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



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 14.12 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 i
	1	0	1	External clock: counts on TCLKC pin i
	1	1	0	Internal clock: counts on P $\phi$ /256
	1	1	1	External clock: counts on TCLKD pin i

Note: This setting is ignored when channel 5 is in phase counting mode.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
5	BFB	0	R/W	Buffer Operation B Specifies whether TGRB is to normally operate, and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD capture/output compare is not generated. In channels 1, 2, 4, and 5, which have no TGRD 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, and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC capture/output compare is not generated. In channels 1, 2, 4, and 5, which have no TGRC 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.
0	MD0	0	R/W	0. See table 14.13 for details.

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 be written to MD2.

To designate the input capture pin in TIOR, the DDR bit and ICR bit for the corresponding should be set to 0 and 1, respectively. For details, see section 13, I/O Ports.

- TIORH\_0, TIOR\_1, TIOR\_2, TIORH\_3, TIOR\_4, TIOR\_5

Bit	7	6	5	4	3	2	1	
Bit Name	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	
Initial Value	0	0	0	0	0	0	0	
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	
Bit Name	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	
Initial Value	0	0	0	0	0	0	0	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

1	IOA1	0	R/W	For details, see tables 14.22, 14.24 to 14.26, 14.29.
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 14.15, and 14.19.
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 14.23, and 14.27.
0	IOC0	0	R/W	

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
					Input capture at TCNT_1 count-up/count

[Legend]

x: Don't care

- Notes:
1. When bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and Pφ/1 is used as the TCNT\_1 count clock, this setting is invalid and input capture is not generated.
  2. In PWM mode 1, the IOB3 to IOB0 settings control output of the TIOCA0 pin function.

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-up/clock Input capture at TCNT_1 count-up/clock

[Legend]

x: Don't care

- Notes:
1. 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.
  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.
  3. In PWM mode 1, the IOD3 to IOD0 settings control output of the TIOCC0 pin.

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

Note: In PWM mode 1, the IOB3 to IOB0 settings control output of the TIOCA1 pin function.



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

Note: In PWM mode 1, the IOB3 to IOB0 settings control output of the TIOCA2 pin function.

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
					Input capture at TCNT_4 count-up/count

[Legend]

x: Don't care

- Notes:
1. When bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and Pφ/1 is used as the TCNT\_4 count clock, this setting is invalid and input capture is not generated.
  2. In PWM mode 1, the IOB3 to IOB0 settings control output of the TIOCA3 pin function.

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
					Input capture at TCNT_4 count-up/count

[Legend]

x: Don't care

- Notes:
1. 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.
  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.
  3. In PWM mode 1, the IOD3 to IOD0 settings control output of the TIOCC3 pin.

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

Note: \* In PWM mode 1, the IOB3 to IOB0 settings control output of the TIOCA4 pin function.

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

Note: \* In PWM mode 1, the IOB3 to IOB0 settings control output of the TIOCA5 pin

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
					Input capture at TCNT_1 count-up/count

[Legend]

x: Don't care

Note: \* When the bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and P $\phi$ /1 is used a count clock of TCNT\_1, this setting is invalid and input capture is not generate

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
					Input capture at TCNT_1 count-up/count

[Legend]

x: Don't care

- Note:
1. When the bits TPSC2 to TPSC0 in TCR\_1 are set to B'000 and P $\phi$ /1 is used as count clock of TCNT\_1, 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 setting is invalid and input capture/output compare is not generated.

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 compare match/input capture

[Legend]

x: Don't care



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

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
					Input capture at TCNT_4 count-up/count

[Legend]

x: Don't care

Note: \* When the bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and P $\phi$ /1 is used a count clock of TCNT\_4, this setting is invalid and input capture is not generate

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-up/clock Input capture at TCNT_4 count-up/clock

[Legend]

x: Don't care

- Note:
1. When the bits TPSC2 to TPSC0 in TCR\_4 are set to B'000 and P $\phi$ /1 is used as a count clock of TCNT\_4, 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.

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

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

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 start requests by TGRA input capture/compare match.</p> <p>0: A/D conversion start request generation disabled</p> <p>1: A/D conversion start request generation enabled</p>
6	—	1	—	<p>Reserved</p> <p>This bit is always read as 1 and cannot be modified.</p>
5	TCIEU	0	R/W	<p>Underflow Interrupt Enable</p> <p>Enables/disables interrupt requests (TCIU) by TCFU flag when the TCFU flag in TSR is set to 1 in channels 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 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>

Enables/disables interrupt requests (TGIC) by the TGIC bit in TSR is set to 1 in channels 0 and 1.

In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.

0: Interrupt requests (TGIC) by TGFC bit disabled

1: Interrupt requests (TGIC) by TGFC bit enabled

1	TGIEB	0	R/W	TGR Interrupt Enable B Enables/disables interrupt requests (TGIB) by the TGIB bit in TSR is set to 1. 0: Interrupt requests (TGIB) by TGFB bit disabled 1: Interrupt requests (TGIB) by TGFB bit enabled
0	TGIEA	0	R/W	TGR Interrupt Enable A Enables/disables interrupt requests (TGIA) by the TGIA bit in TSR is set to 1. 0: Interrupt requests (TGIA) by TGFA bit disabled 1: Interrupt requests (TGIA) by TGFA bit enabled

Note: \* The bit 7 in TIER of unit 1 is a reserved bit. This bit is always read as 0 and the value should not be changed.

### 14.3.5 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
Bit Name	TCFD	—	TCFU	TCFV	TGFD	TGFC	TGFB
Initial Value	1	1	0	0	0	0	0
R/W	R	—	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: \* Only 0 can be written to bits 5 to 0, to clear flags.

5	TCFU	0	R/(W)*	<p>Underflow Flag</p> <p>Status flag that indicates that a TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting. In channels 0 and 3, bit 5 is reserved. It is always read as 1 and cannot be modified.</p> <p>[Setting condition]</p> <p>When the TCNT value underflows (changes from H'0000 to H'FFFF)</p> <p>[Clearing condition]</p> <p>When a 0 is written to TCFU after reading TCFU = 1</p> <p>(When the CPU is used to clear this flag by writing 0 to it, the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
4	TCFV	0	R/(W)*	<p>Overflow Flag</p> <p>Status flag that indicates that a TCNT overflow has occurred. It is always read as 1 and cannot be modified.</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</p> <p>(When the CPU is used to clear this flag by writing 0 to it, the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>



- When TCNT value is transferred to TGRD while TGRC is functioning as capture register

[Clearing conditions]

- When DTC is activated by a TGID interrupt  
DISEL bit in MRB of DTC is 0
- When 0 is written to TGFD after reading TGFD  
(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled to read the flag after writing 0 to it.)

---

2	TGFC	0	R/(W)*	<p>Input Capture/Output Compare Flag C</p> <p>Status flag that indicates the occurrence of TGRC capture or compare match in channels 0 and 3. In channels 1, 2, 4, and 5, bit 2 is reserved. It is 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 TGRC capture signal while TGRC is functioning as capture register</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When DTC is activated by a TGIC interrupt DISEL bit in MRB of DTC is 0</li> <li>• When 0 is written to TGFC after reading TGFC (When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled to read the flag after writing 0 to it.)</li> </ul>
---	------	---	--------	---

---

capture register

[Clearing conditions]

- When DTC is activated by a TGIB interrupt while the DISEL bit in MRB of DTC is 0
- When 0 is written to TGFB after reading TGFB (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, to read the flag after writing 0 to it.)

---

0	TGFA	0	R/(W)*	Input Capture/Output Compare Flag A
---	------	---	--------	-------------------------------------

Status flag that indicates the occurrence of TGRA capture or compare match.

[Setting conditions]

- When TCNT = TGRA while TGRA is functioning as output compare register
- When TCNT value is transferred to TGRA by capture signal while TGRA is functioning as input capture register

[Clearing conditions]

- When DTC is activated by a TGIA interrupt while the DISEL bit in MRB of DTC is 0
- When DMAC is activated by a TGIA interrupt while the DTA bit in DMDR of DMAC is 1
- When 0 is written to TGFA after reading TGFA (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, to read the flag after writing 0 to it.)

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

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

### 14.3.7 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 input capture 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 operation are TGRA–TGRC and TGRB–TGRD.

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

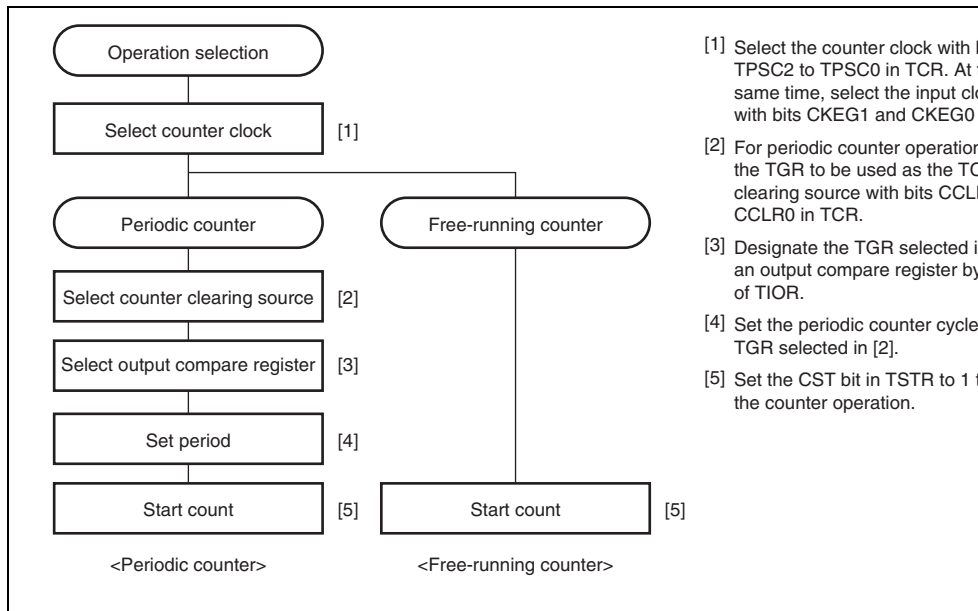
Bit	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	—	Reserved The write value should always be 0.
5	CST5	0	R/W	Counter Start 5 to 0
4	CST4	0	R/W	These bits select operation or stoppage for TCNT
3	CST3	0	R/W	If 0 is written to the CST bit during operation with
2	CST2	0	R/W	TIOC pin designated for output, the counter stop
1	CST1	0	R/W	TIOC pin output compare output level is retained
0	CST0	0	R/W	is written to when the CST bit is cleared to 0, the output level will be changed to the set initial outp 0: TCNT_5 to TCNT_0 count operation is stoppe 1: TCNT_5 to TCNT_0 performs count operation

Bit	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	R/W	Reserved 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 or synchronized with other channels.
3	SYNC3	0	R/W	
2	SYNC2	0	R/W	When synchronous operation is selected, synchronous presetting of multiple channels, and synchronous clearing through counter clearing on another channel are possible.
1	SYNC1	0	R/W	
0	SYNC0	0	R/W	To set synchronous operation, the SYNC bits for both channels must be set to 1. To set synchronous clearing, in addition to the SYNC bit, the TCNT source must also be set by means of bits CCLF0 and CCLR0 in TCR. 0: TCNT_5 to TCNT_0 operate independently (synchronous presetting/clearing is unrelated to other channels) 1: TCNT_5 to TCNT_0 perform synchronous operation (TCNT synchronous presetting/synchronous clearing is possible)

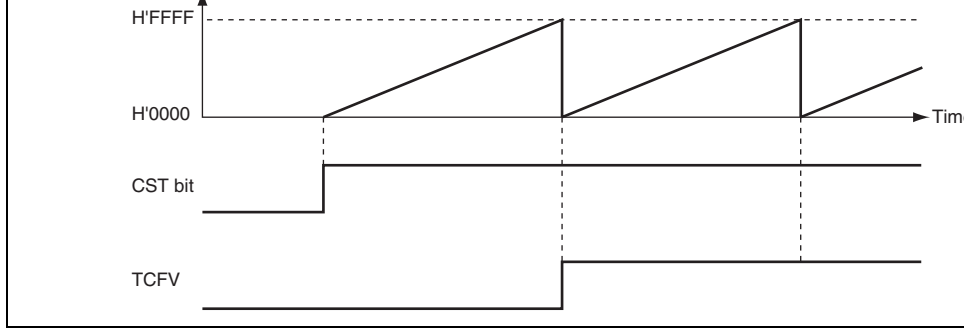
When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and

**(a) Example of count operation setting procedure**

Figure 14.3 shows an example of the count operation setting procedure.



**Figure 14.3 Example of Counter Operation Setting Procedure**



**Figure 14.4 Free-Running Counter Operation**

When compare match is selected as the TCNT clearing source, the TCNT counter for the 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 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 reaches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

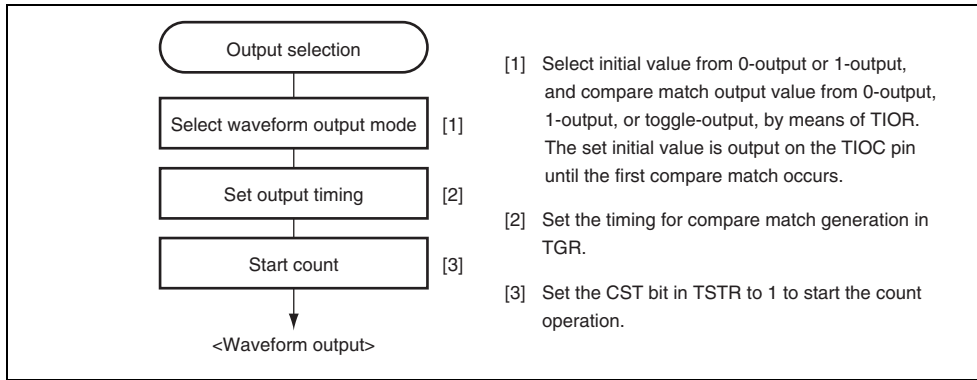
If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests a compare match interrupt. After a compare match, TCNT starts counting up again from H'0000.

**Figure 14.5 Periodic Counter Operation****(2) Waveform Output by Compare Match**

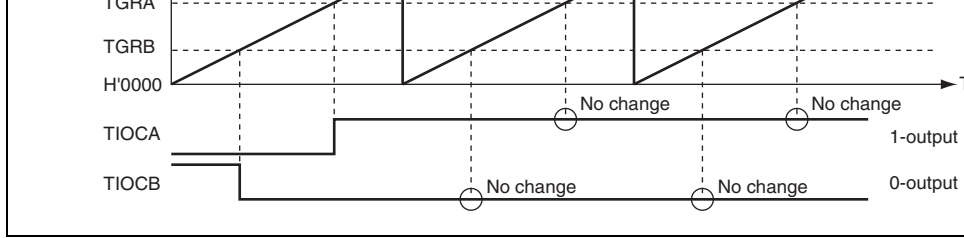
The TPU can perform 0, 1, or toggle output from the corresponding output pin using a compare match.

**(a) Example of setting procedure for waveform output by compare match**

Figure 14.6 shows an example of the setting procedure for waveform output by a compare match.

**Figure 14.6 Example of Setting Procedure for Waveform Output by Compare Match**

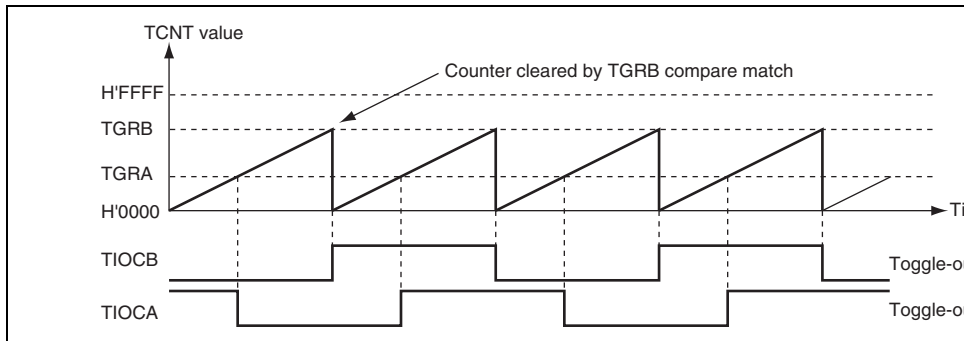




**Figure 14.7 Example of 0-Output/1-Output Operation**

Figure 14.8 shows an example of toggle output.

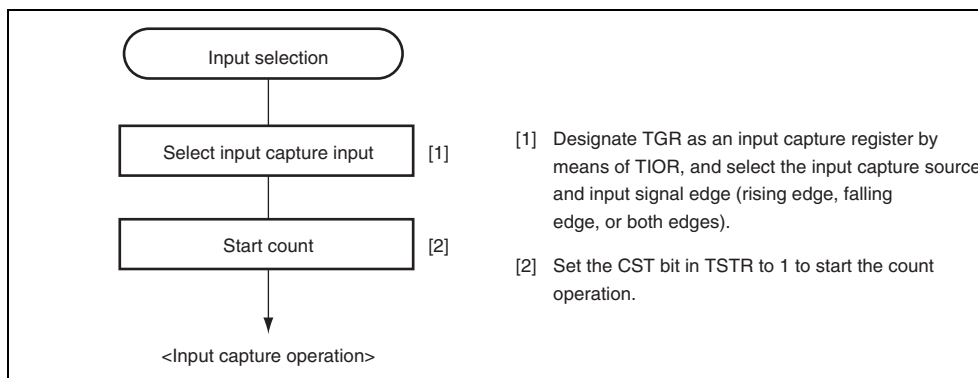
In this example, TCNT has been designated as a periodic counter (with counter clearing by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.



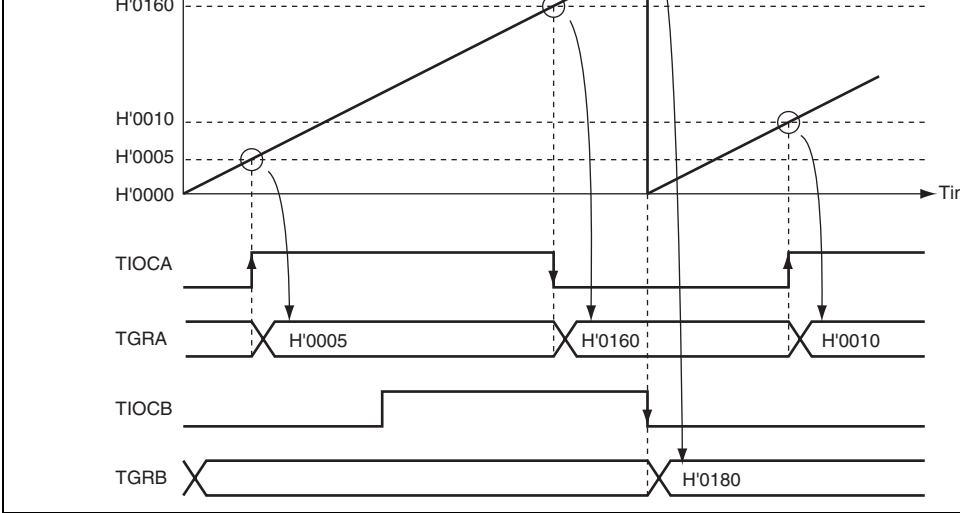
**Figure 14.8 Example of Toggle Output Operation**

**(a) Example of setting procedure for input capture operation**

Figure 14.9 shows an example of the setting procedure for input capture operation.

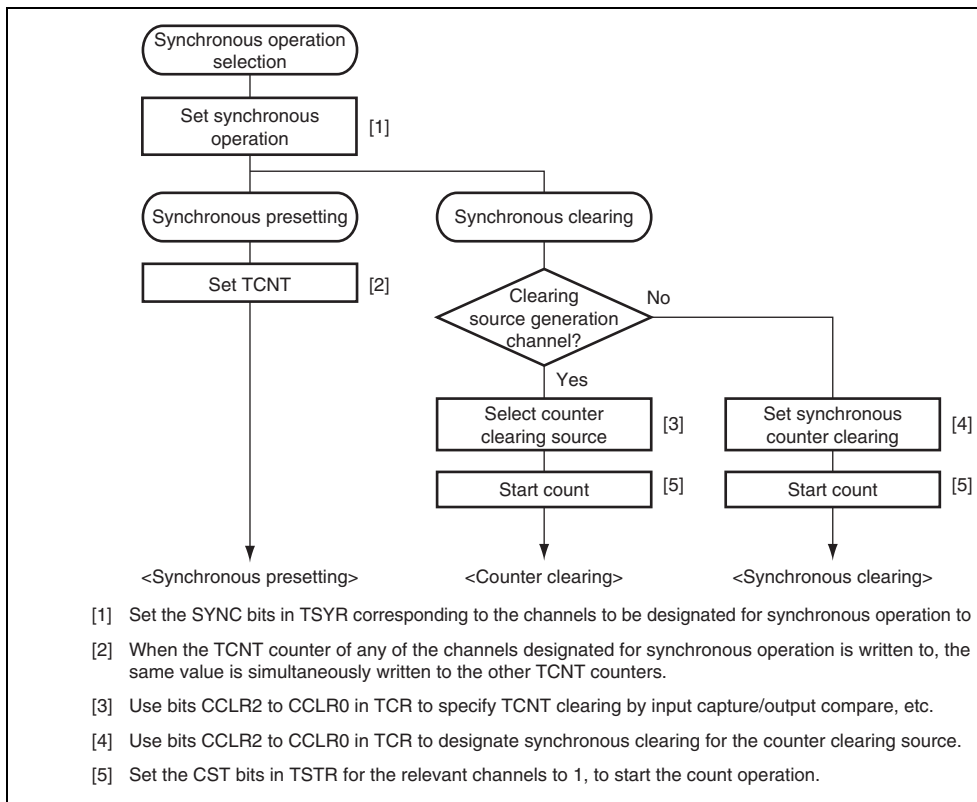


**Figure 14.9 Example of Setting Procedure for Input Capture Operation**

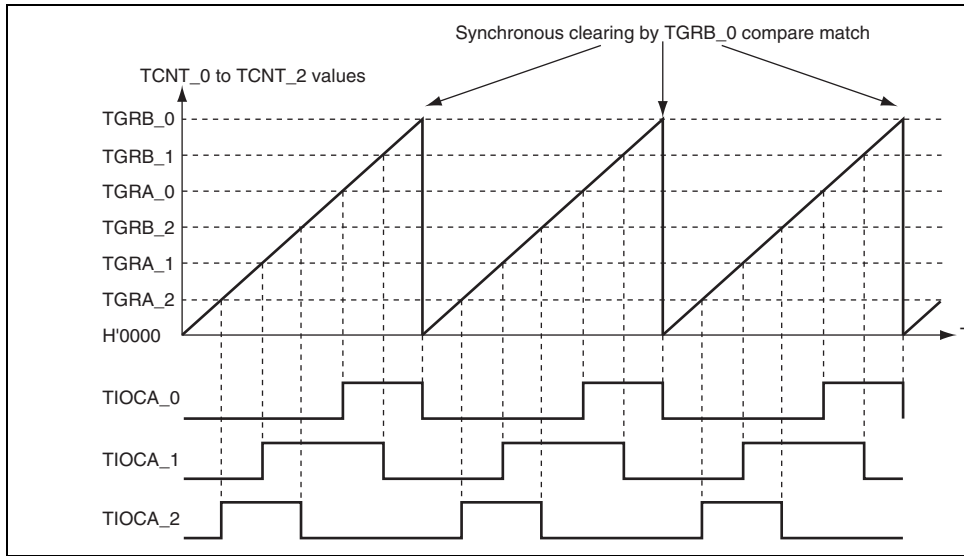


**Figure 14.10 Example of Input Capture Operation**

Figure 14.11 shows an example of the synchronous operation setting procedure.



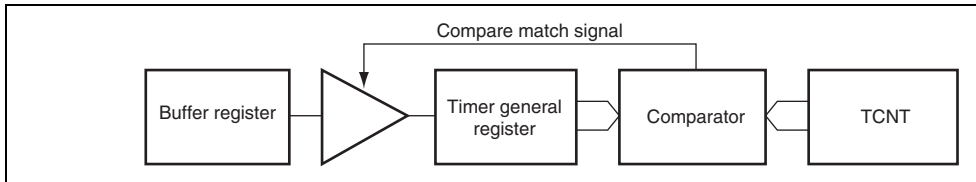
**Figure 14.11 Example of Synchronous Operation Setting Procedure**



**Figure 14.12 Example of Synchronous 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

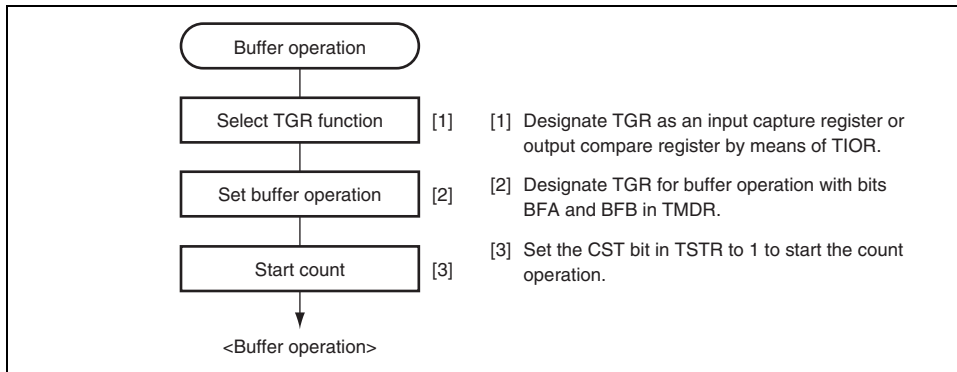
- When TGR is an output compare register  
When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.  
This operation is illustrated in figure 14.13.



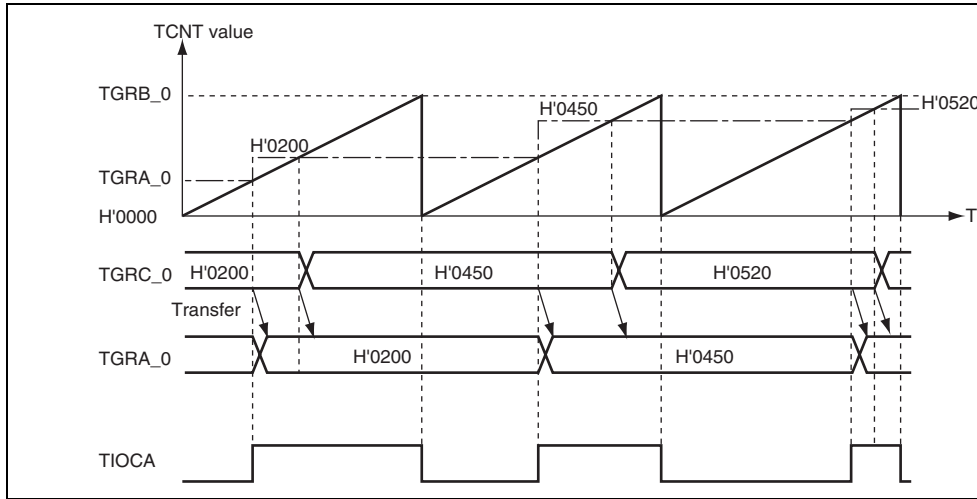
**Figure 14.13 Compare Match Buffer Operation**

## (1) Example of Buffer Operation Setting Procedure

Figure 14.15 shows an example of the buffer operation setting procedure.

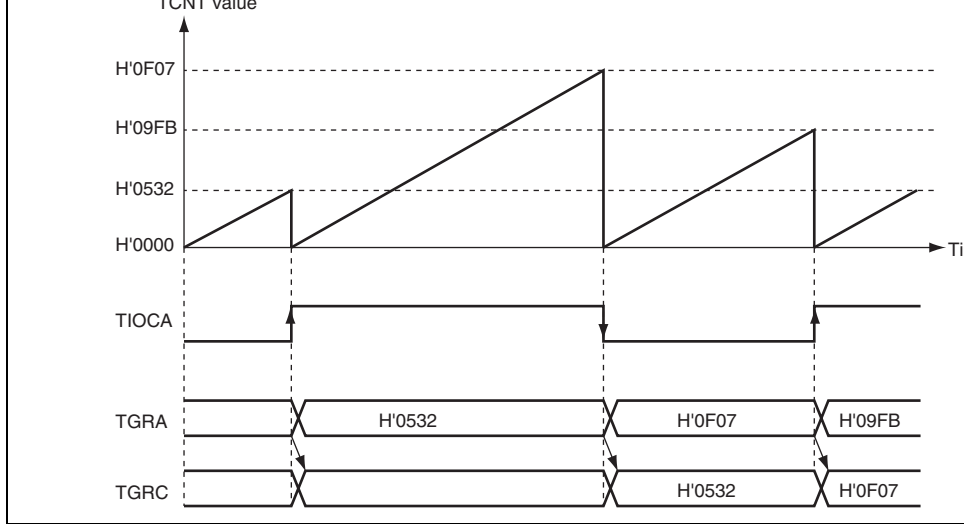


**Figure 14.15 Example of Buffer Operation Setting Procedure**



**Figure 14.16 Example of Buffer Operation (1)**





**Figure 14.17 Example of Buffer Operation (2)**

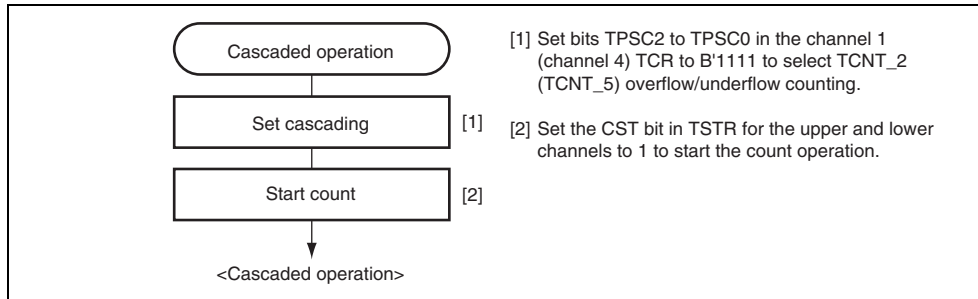
Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is i and the counter operates independently in phase counting mode.

**Table 14.31 Cascaded Combinations**

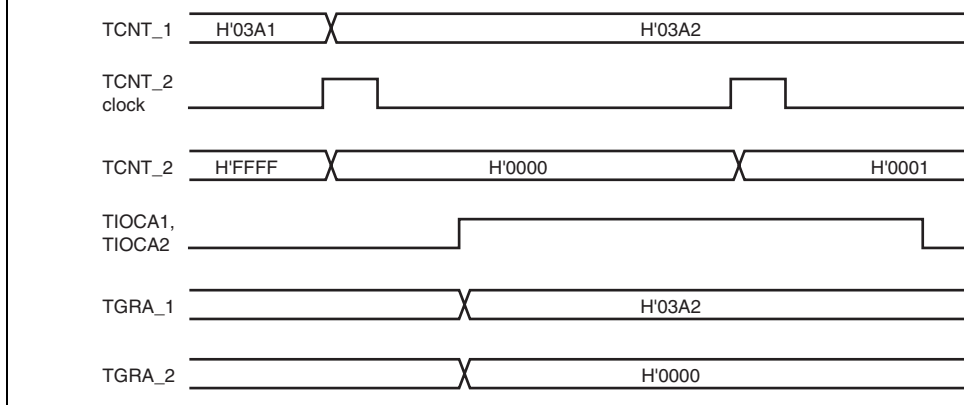
Combination	Upper 16 Bits	Lower 16 Bits
Channels 1 and 2	TCNT_1	TCNT_2
Channels 4 and 5	TCNT_4	TCNT_5

**(1) Example of Cascaded Operation Setting Procedure**

Figure 14.18 shows an example of the setting procedure for cascaded operation.



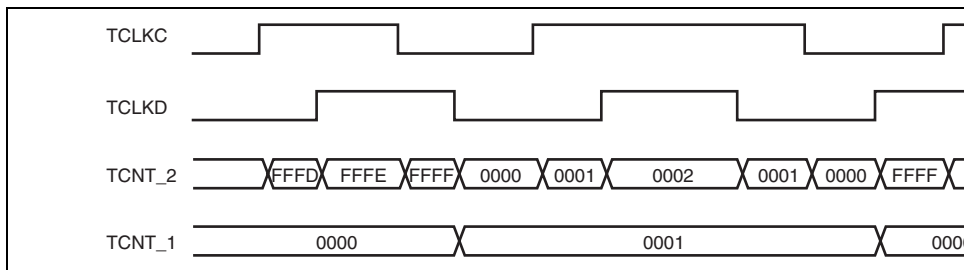
**Figure 14.18 Cascaded Operation Setting Procedure**



**Figure 14.19 Example of Cascaded Operation (1)**

Figure 14.20 illustrates the operation when counting upon TCNT\_2 overflow/underflow 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 14.20 Example of Cascaded Operation (2)**

There are two PWM modes, as described below.

### 1. PWM mode 1

PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRC and TGRB 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 TGRB. 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.

### 2. PWM mode 2

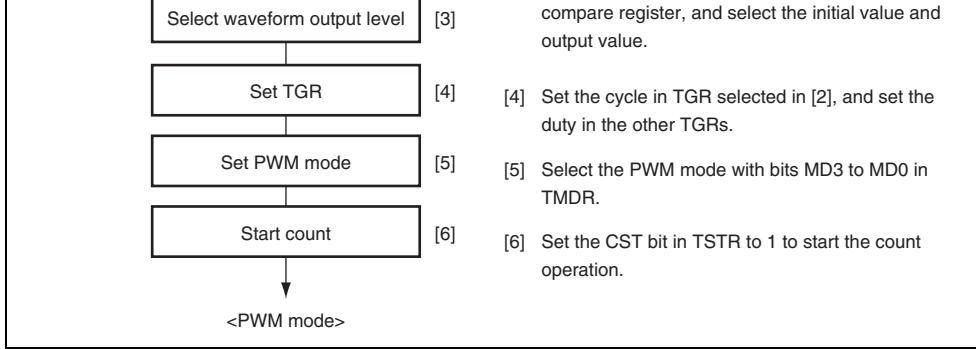
PWM output is generated using one TGR as the cycle register and the others as duty cycle registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronous register compare match, the output value of each pin is set to the initial value set in TIOR. 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 14.32.

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 cy



**Figure 14.21 Example of PWM Mode Setting Procedure**

## (2) Examples of PWM Mode Operation

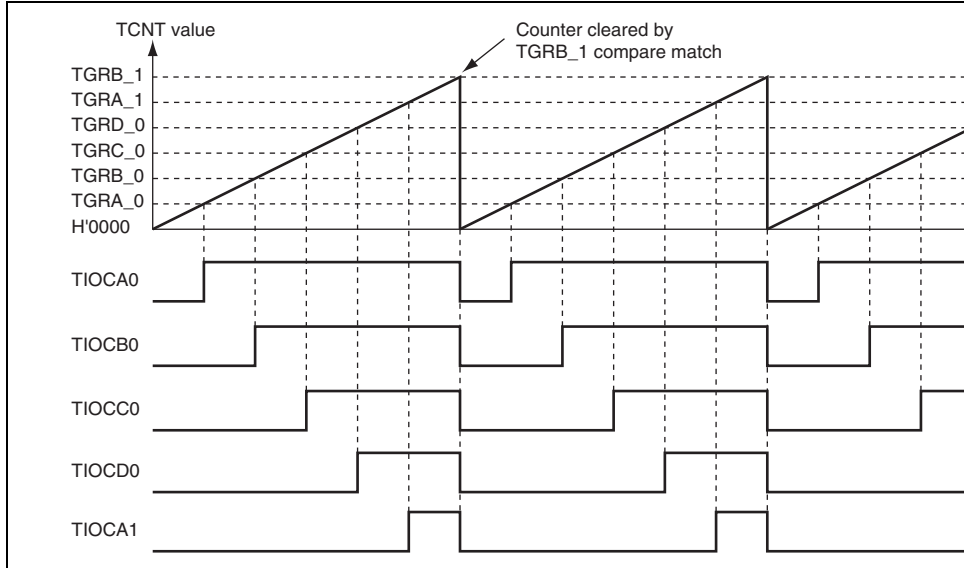
Figure 14.22 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the initial output value and output value, and 1 is set as the TGRB output value.

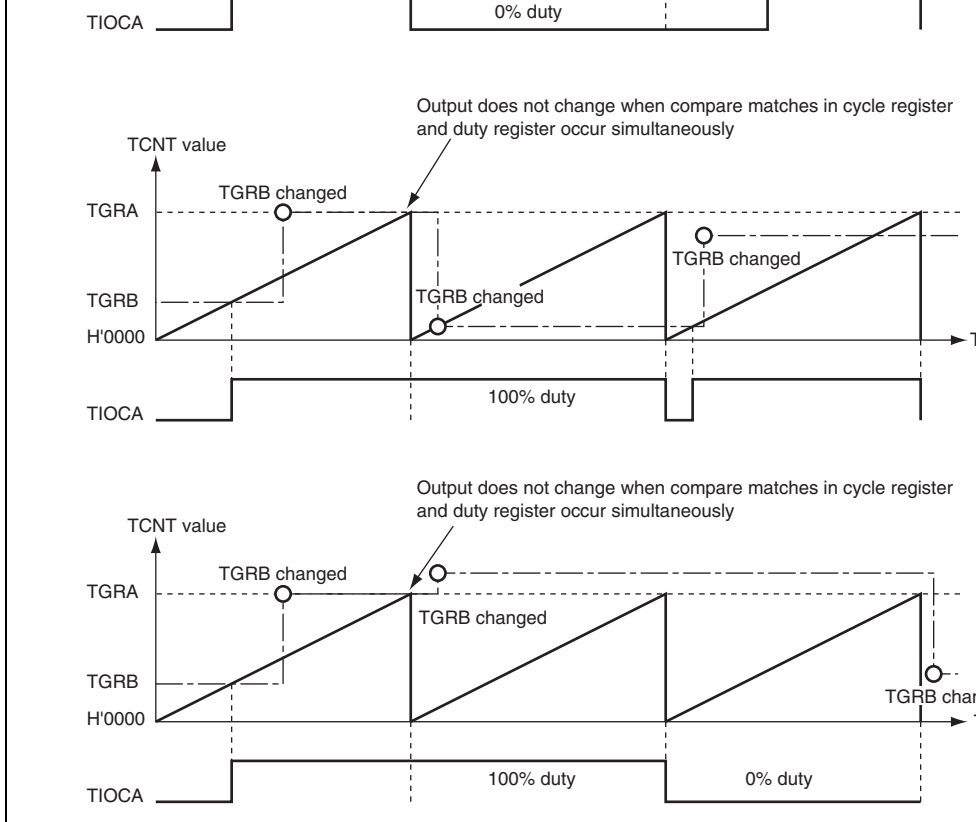
In this case, the value set in TGRA is used as the cycle, and the value set in TGRB registers the duty cycle.

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 50% duty cycle PWM waveform.

In this case, the value set in TGRB\_1 is used as the cycle, and the values set in the other TGR registers are used to set the duty cycle.



**Figure 14.23 Example of PWM Mode Operation (2)**



**Figure 14.24 Example of PWM Mode Operation (3)**



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.

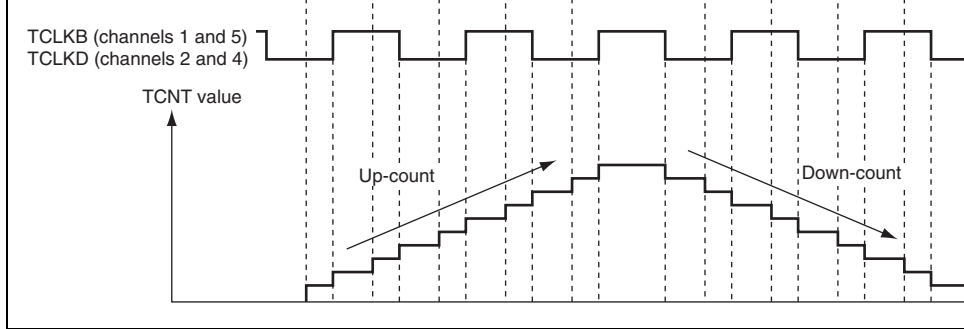
The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 14.33 shows the correspondence between external clock pins and channels.

**Table 14.33 Clock Input Pins in Phase Counting Mode**

<b>Channels</b>	<b>External Clock Pins</b>	
	<b>A-Phase</b>	<b>B-Phase</b>
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

## Figure 14.25 Example of Phase Counting Mode Setting Procedure



**Figure 14.26 Example of Phase Counting Mode 1 Operation**

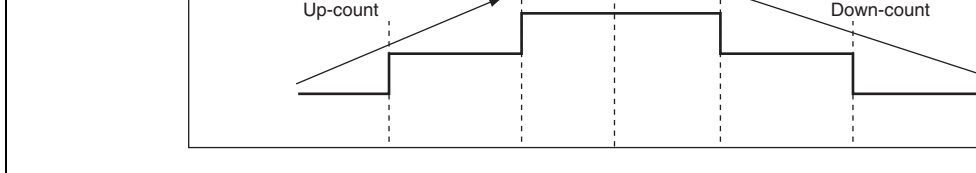
**Table 14.34 Up/Down-Count Conditions in Phase Counting Mode 1**

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		
Low level		
	High level	
	Low level	

[Legend]

: Rising edge

: Falling edge



**Figure 14.27 Example of Phase Counting Mode 2 Operation**

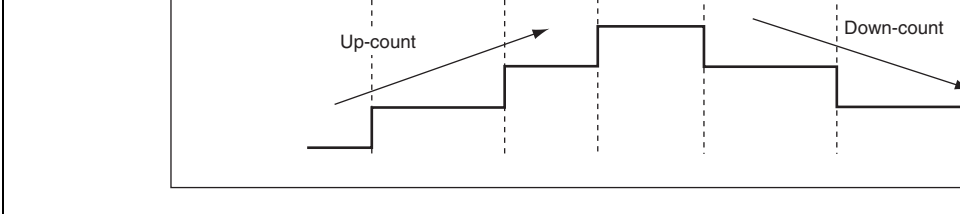
**Table 14.35 Up/Down-Count Conditions in Phase Counting Mode 2**

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	$\uparrow$	Don't care
Low level	$\downarrow$	Don't care
$\uparrow$	Low level	Don't care
$\downarrow$	High level	Up-count
High level	$\downarrow$	Don't care
Low level	$\uparrow$	Don't care
$\uparrow$	High level	Don't care
$\downarrow$	Low level	Down-count

[Legend]

$\uparrow$ : Rising edge

$\downarrow$ : Falling edge



**Figure 14.28 Example of Phase Counting Mode 3 Operation**

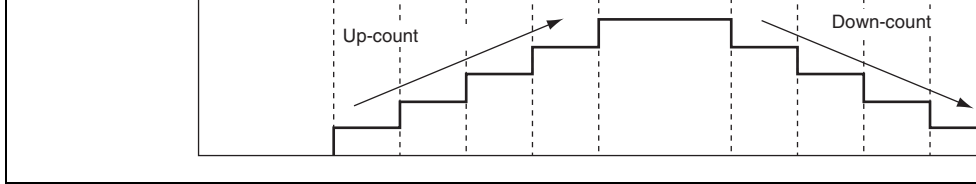
**Table 14.36 Up/Down-Count Conditions in Phase Counting Mode 3**

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



**Figure 14.29 Example of Phase Counting Mode 4 Operation**

**Table 14.37 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		
	Low level	Don't care
	High level	
High level		Down-count
Low level		
	High level	Don't care
	Low level	

[Legend]

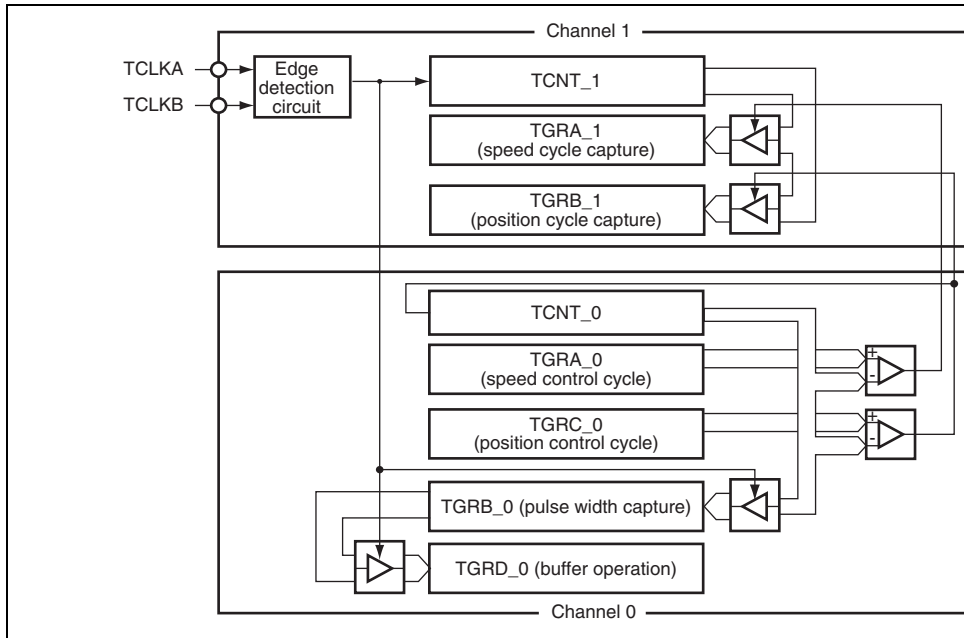
: Rising edge

: Falling edge

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, TGRC\_0 compare matches are selected as the input capture source, and the up/down-count values for the control cycles are stored.

This procedure enables accurate position/speed detection to be achieved.



**Figure 14.30 Phase Counting Mode Application Example**

channel is fixed. For details, see section 7, Interrupt Controller.

Table 14.38 lists the TPU interrupt sources.

**Table 14.38 TPU Interrupts**

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	DMA 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
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



	TCI4V	TCNT_4 overflow	TCFV_4	Not possible	Not possible
5	TCI4U	TCNT_4 underflow	TCFU_4	Not possible	Not possible
	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 priorities 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 0, 3, 4, and 5.

The A/D converter can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 10, DMA Controller (DMAC).

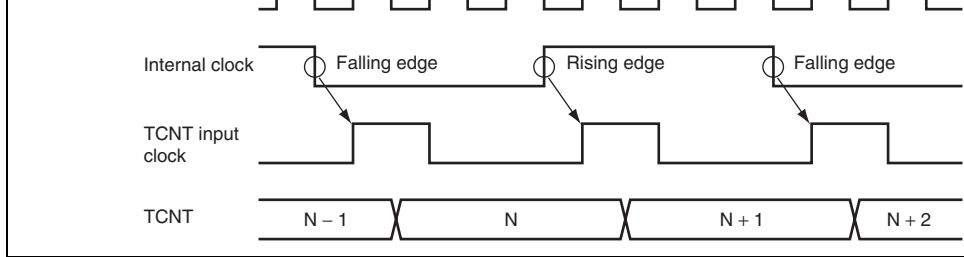
In TPU, one in each channel, totally six TGRA input capture/compare match interrupts can be used as DMAC activation sources.

## 14.8 A/D Converter Activation

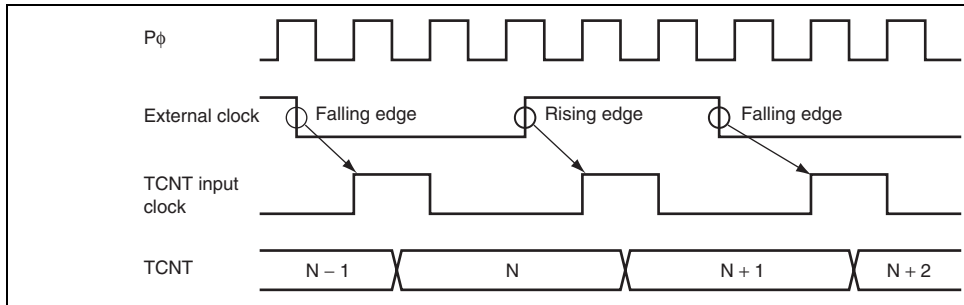
Concerning the unit 0 in TPU, the TGRA input capture/compare match for each channel can be used to activate the A/D converter. (However, the A/D converter cannot be activated in unit 1.)

If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

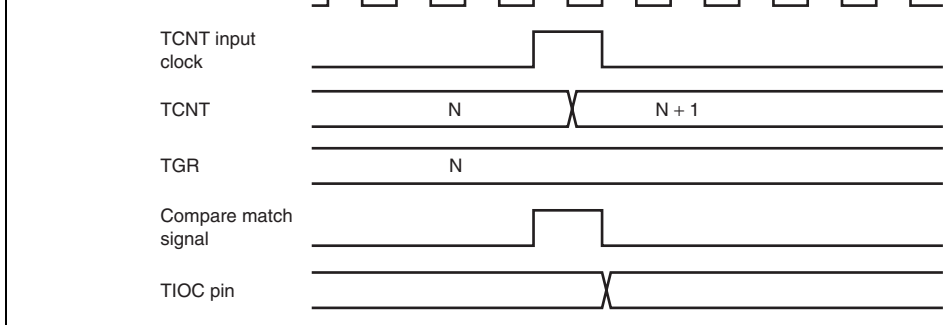
In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.



**Figure 14.31 Count Timing in Internal Clock Operation**



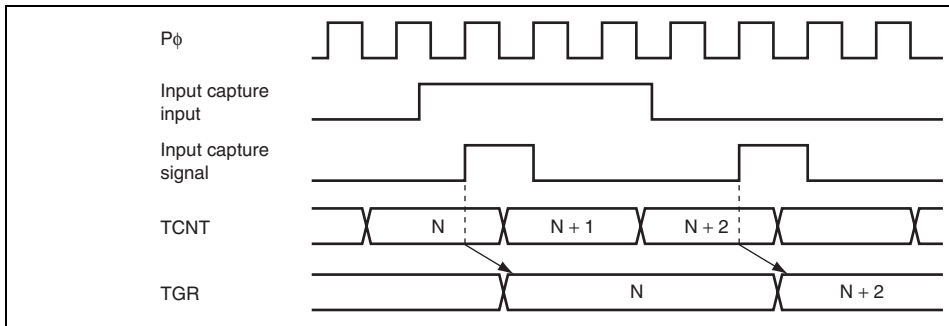
**Figure 14.32 Count Timing in External Clock Operation**



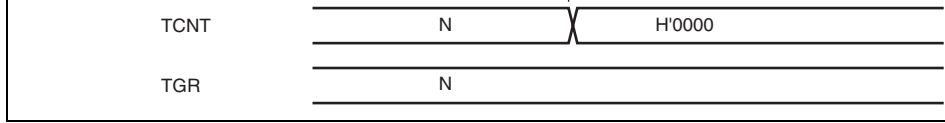
**Figure 14.33 Output Compare Output Timing**

### (3) Input Capture Signal Timing

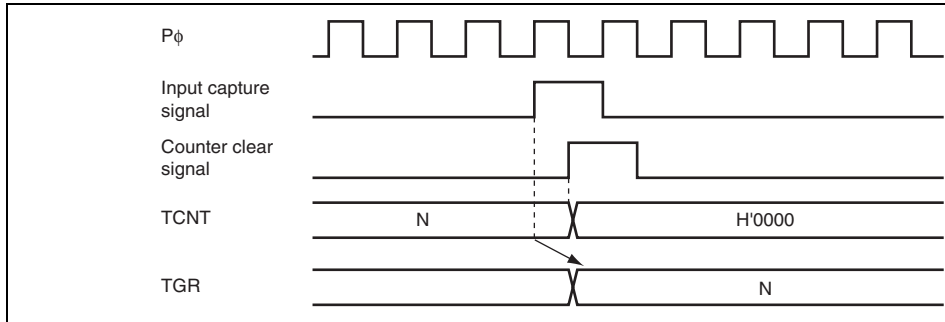
Figure 14.34 shows input capture signal timing.



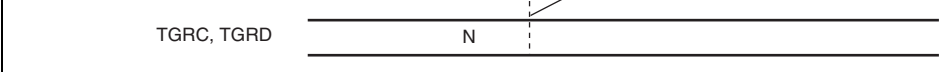
**Figure 14.34 Input Capture Input Signal Timing**



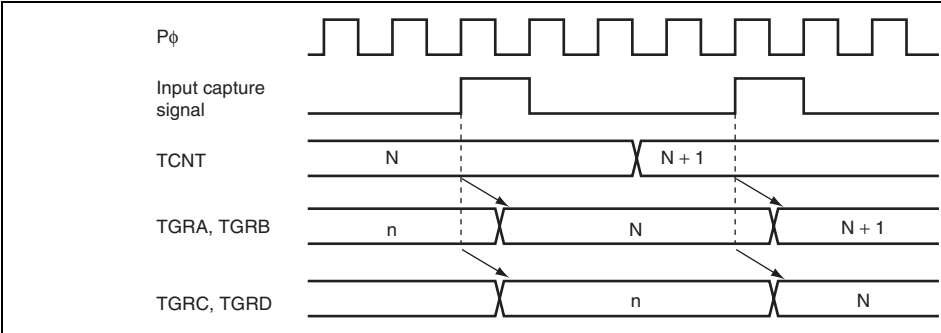
**Figure 14.35 Counter Clear Timing (Compare Match)**



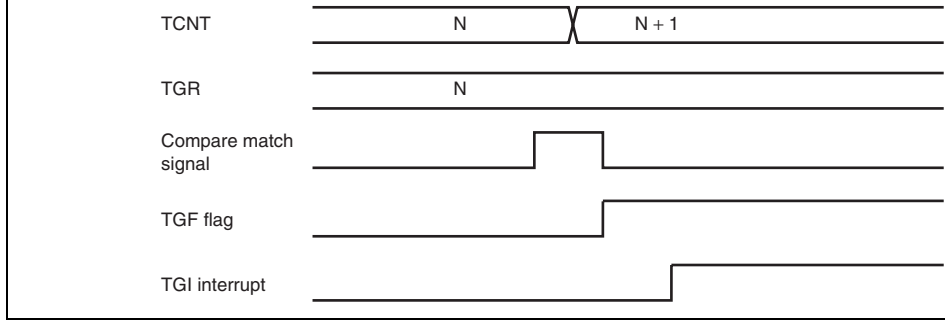
**Figure 14.36 Counter Clear Timing (Input Capture)**



**Figure 14.37 Buffer Operation Timing (Compare Match)**



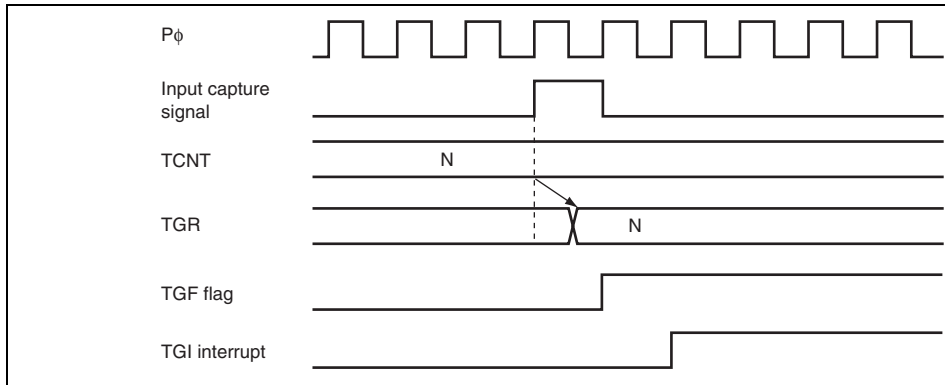
**Figure 14.38 Buffer Operation Timing (Input Capture)**



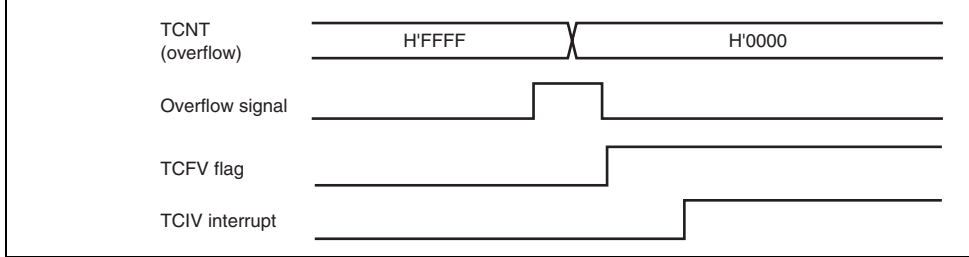
**Figure 14.39 TGI Interrupt Timing (Compare Match)**

**(2) TGF Flag Setting Timing in Case of Input Capture**

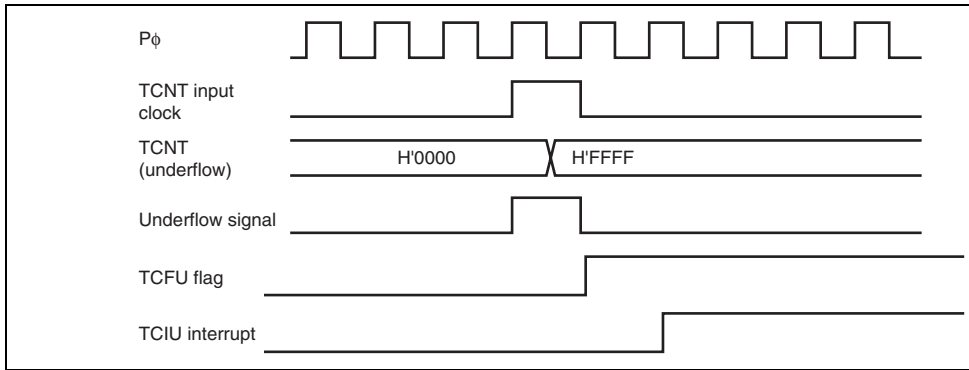
Figure 14.40 shows the timing for setting of the TGF flag in TSR by input capture occurring at the TGI interrupt request signal timing.



**Figure 14.40 TGI Interrupt Timing (Input Capture)**

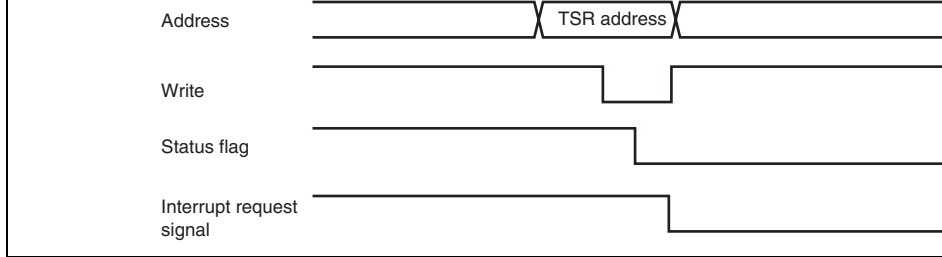


**Figure 14.41 TCIV Interrupt Setting Timing**



**Figure 14.42 TCIU Interrupt Setting Timing**

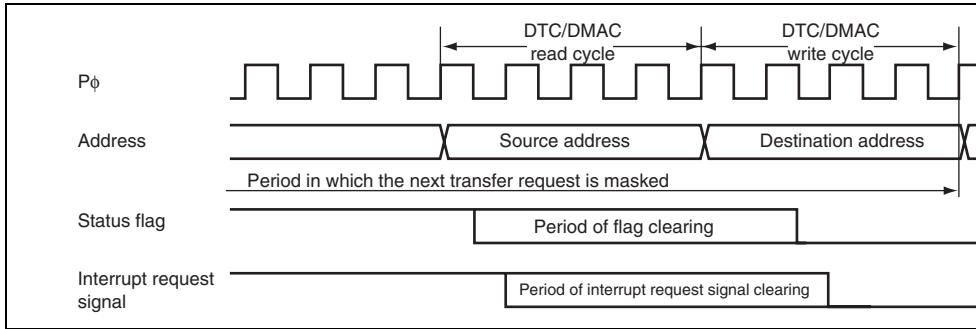




**Figure 14.43 Timing for Status Flag Clearing by CPU**

The status flag and interrupt request signal are cleared in synchronization with  $P\phi$  after the DMAC transfer has started, as shown in figure 14.44. If conflict occurs for clearing the status flag and interrupt request signal due to activation of multiple DTC or DMAC transfers, it will be masked for five clock cycles ( $P\phi$ ) for clearing them, as shown in figure 14.45. The next transfer request signal 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.

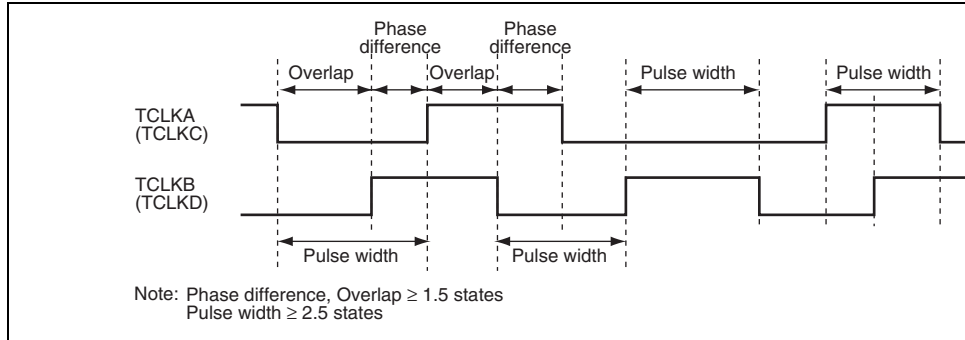
**Figure 14.44 Timing for Status Flag Clearing by DTC/DMAC Activation (1)**



**Figure 14.45 Timing for Status Flag Clearing by DTC/DMAC Activation (2)**

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.

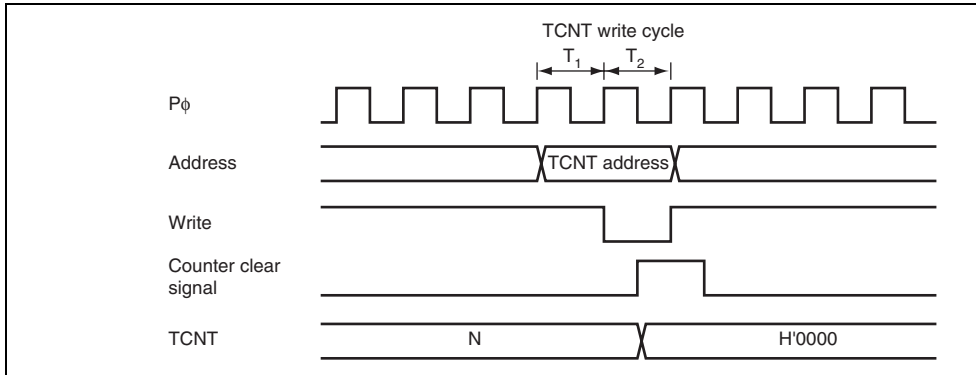
In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 14.46 shows the input conditions in phase counting mode.



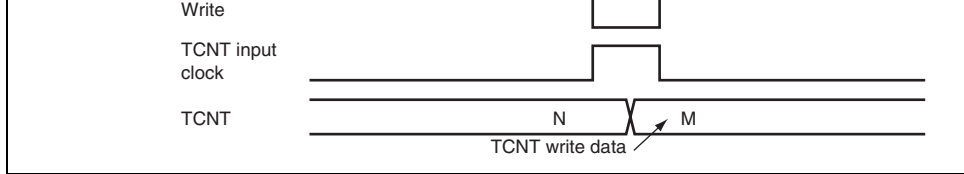
**Figure 14.46 Phase Difference, Overlap, and Pulse Width in Phase Counting**

### 14.10.4 Conflict between TCNT Write and Clear Operations

If the counter clearing signal is generated in the T2 state of a TCNT write cycle, TCNT c takes precedence and the TCNT write is not performed. Figure 14.47 shows the timing in case.



**Figure 14.47 Conflict between TCNT Write and Clear Operations**

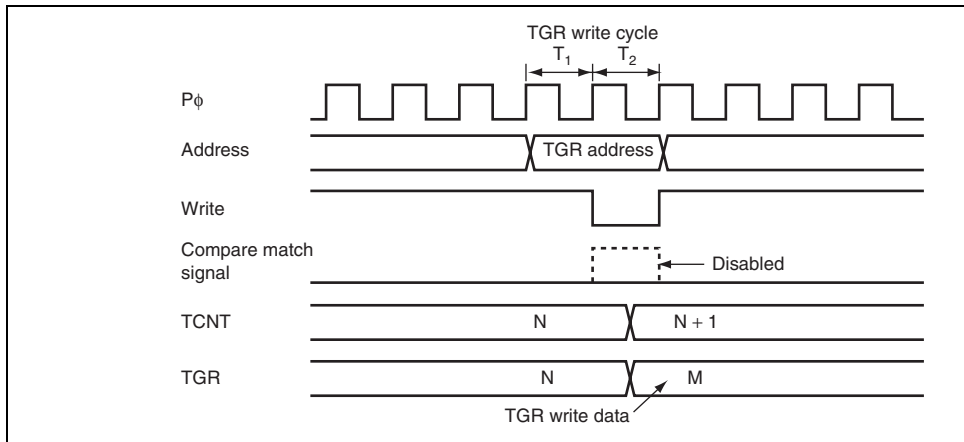


**Figure 14.48 Conflict between TCNT Write and Increment Operations**

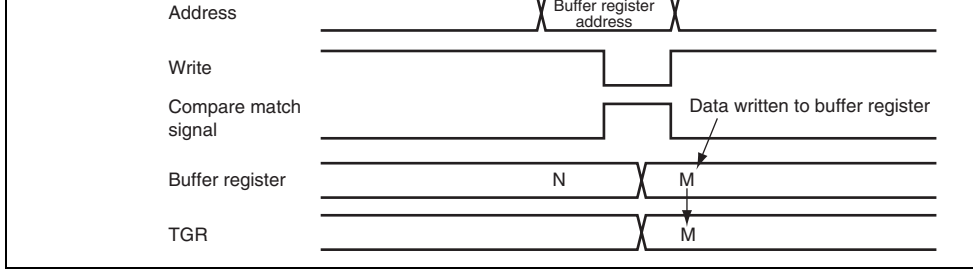
### 14.10.6 Conflict between TGR Write and Compare Match

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 value as before is written.

Figure 14.49 shows the timing in this case.



**Figure 14.49 Conflict between TGR Write and Compare Match**

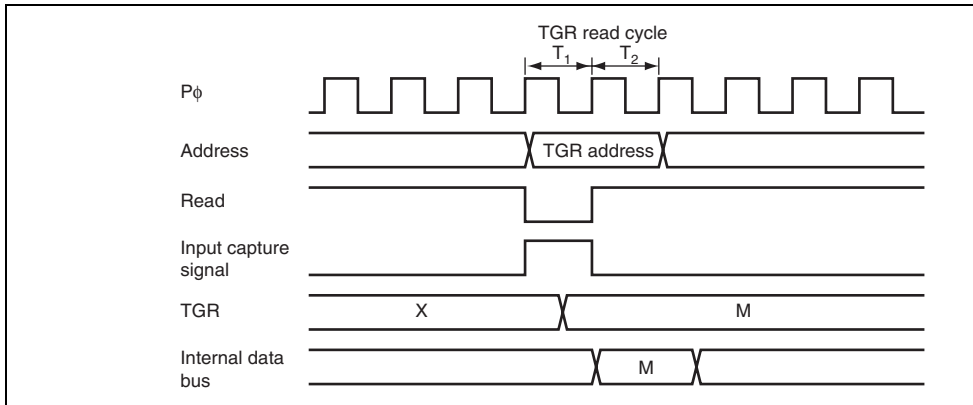


**Figure 14.50 Conflict between Buffer Register Write and Compare Match**

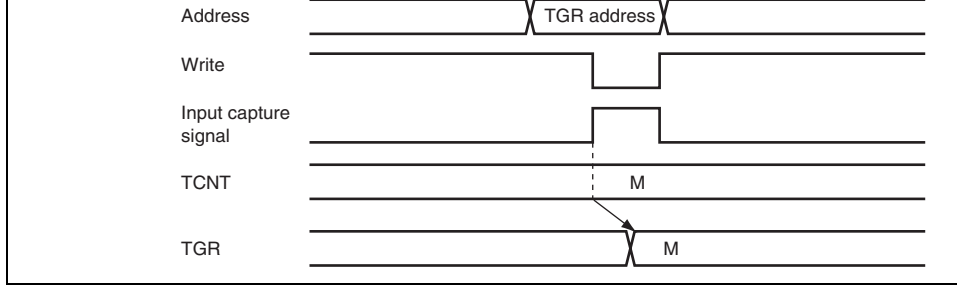
### 14.10.8 Conflict between TGR Read and Input Capture

If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is captured will be the data after input capture transfer.

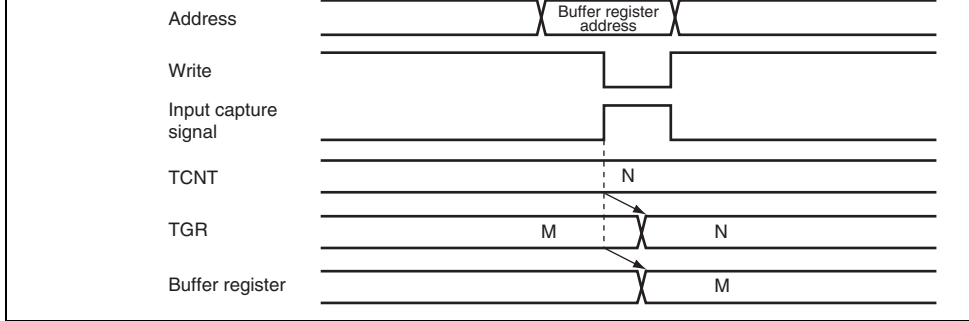
Figure 14.51 shows the timing in this case.



**Figure 14.51 Conflict between TGR Read and Input Capture**

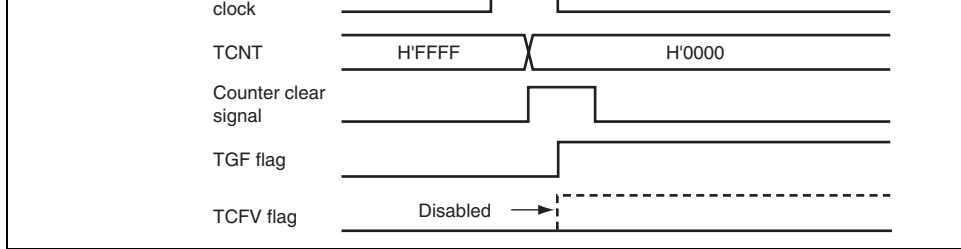


**Figure 14.52 Conflict between TGR Write and Input Capture**



**Figure 14.53 Conflict between Buffer Register Write and Input Capture**



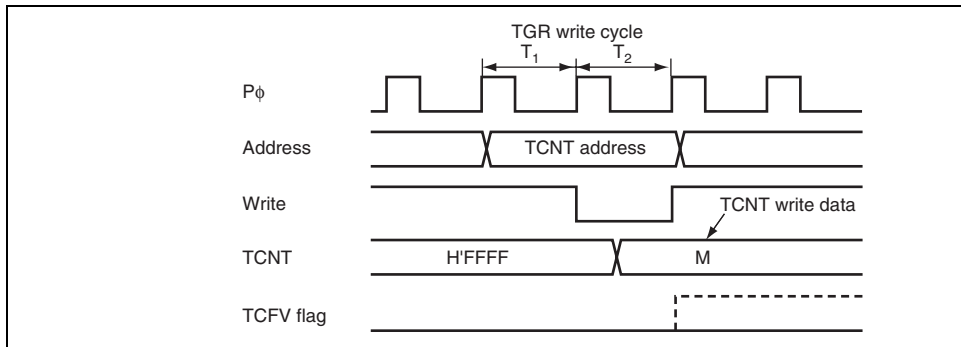


**Figure 14.54 Conflict between Overflow and Counter Clearing**

#### 14.10.12 Conflict between TCNT Write and Overflow/Underflow

If an overflow/underflow occurs due to increment/decrement in the T2 state of a TCNT cycle, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 14.55 shows the operation timing when there is conflict between TCNT write and overflow.



**Figure 14.55 Conflict between TCNT Write and Overflow**

be cleared to halt the output. For details, see section 15, I/O Ports.

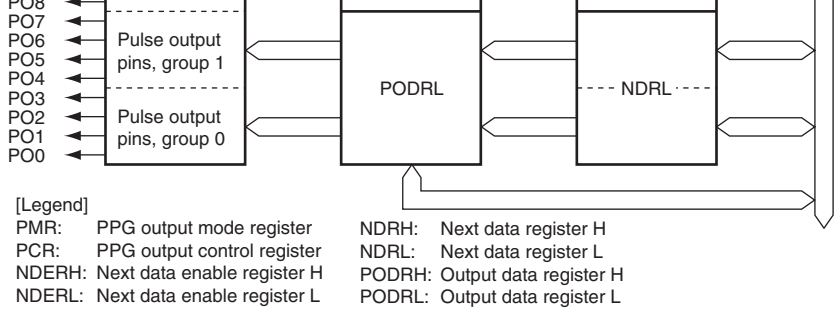
### **14.10.15 Interrupts in the Module Stop State**

If module stop state is entered when an interrupt has been requested, it will not be possible to clear the interrupt source, or the DTC or DMAC activation sources. Interrupts should therefore be disabled before entering module stop state.

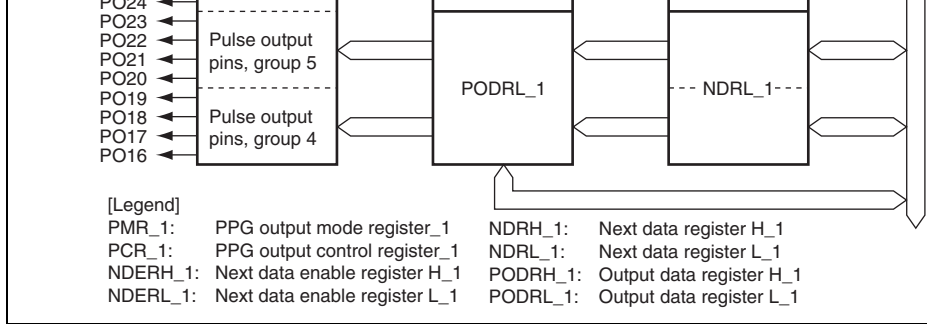
- Four output groups
- Selectable output trigger signals
- Non-overlapping mode
- Can operate together with the data transfer controller (DTC) and DMA controller (D)
- Inverted output can be set
- Module stop state specifiable

**Table 15.1 List of PPG Functions**

	<b>Function</b>	<b>PPG0</b>	<b>PPG1</b>	
PPG output trigger	TPU0	Compare match	Possible	Not possible
		Input capture	Possible	Not possible
	TPU1	Compare match	Not possible	Possible
		Input capture	Not possible	Not possible
Non-overlapping mode		Possible	Possible	
Output data transfer	DTC	Possible	Possible	
	DMAC	Possible	Possible	
Inverted output		Possible	Possible	



**Figure 15.1 Block Diagram of PPG (Unit 0)**



**Figure 15.2 Block Diagram of PPG (Unit 1)**

PO3	Output	
PO4	Output	Group 1 pulse output
PO5	Output	
PO6	Output	
PO7	Output	
PO8	Output	Group 2 pulse output
PO9	Output	
PO10	Output	
PO11	Output	
PO12	Output	Group 3 pulse output
PO13	Output	
PO14	Output	
PO15	Output	

PO24	Output	Group 6 pulse output
PO25	Output	
PO26	Output	
PO27	Output	
PO28	Output	Group 7 pulse output
PO29	Output	
PO30	Output	
PO31	Output	

- Next data register H (NDRH)
- Next data register L (NDRL)
- PPG output control register (PCR)
- PPG output mode register (PMR)

Unit 1:

- Next data enable register H\_1 (NDERH\_1)
- Next data enable register L\_1 (NDERL\_1)
- Output data register H\_1 (PODRH\_1)
- Output data register L\_1 (PODRL\_1)
- Next data register H\_1 (NDRH\_1)
- Next data register L\_1 (NDRL\_1)
- PPG output control register\_1 (PCR\_1)
- PPG output mode register\_1 (PMR\_1)



- NDERL

Bit	7	6	5	4	3	2	1
Bit Name	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	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 output trigger. Values are not transferred from NPODRL 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	

1	NDER25	0	R/W
0	NDER24	0	R/W

---

- NDERL\_1

Bit	Bit Name	Initial Value	R/W	Description
7	NDER23	0	R/W	Next Data Enable 23 to 16
6	NDER22	0	R/W	When a bit is set to 1, the value in the corresponding NDRL_1 bit is transferred to the PODRL_1 bit to clear the selected output trigger. Values are not transferred from NDRL_1 to PODRL_1 for cleared bits.
5	NDER21	0	R/W	
4	NDER20	0	R/W	
3	NDER19	0	R/W	
2	NDER18	0	R/W	
1	NDER17	0	R/W	
0	NDER16	0	R/W	

---

- **PODRL**

Bit	7	6	5	4	3	2	1
Bit Name	POD7	POD6	POD5	POD4	POD3	POD2	POD1
Initial Value	0	0	0	0	0	0	0
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 the output trigger transfers NDRH values to this register during PPG operation. While NDERH is set to 1, the output trigger transfers NDRH values to this register. While NDERH is set to 0, the output trigger transfers NDRL values to this register.
5	POD13	0	R/W	
4	POD12	0	R/W	cannot write to this register. While NDERH is set to 1, the initial output value of the pulse can be set.
3	POD11	0	R/W	
2	POD10	0	R/W	
1	POD9	0	R/W	
0	POD8	0	R/W	

1	POD1	0	R/W
0	POD0	0	R/W

- PODRH\_1

Bit	Bit Name	Initial Value	R/W	Description
7	POD31	0	R/W	Output Data Register 31 to 24
6	POD30	0	R/W	For bits which have been set to pulse output by NDERH_1, the output trigger transfers NDRH_ this register during PPG operation. While NDER set to 1, the CPU cannot write to this register. V NDERH_1 is cleared, the initial output value of can be set.
5	POD29	0	R/W	
4	POD28	0	R/W	
3	POD27	0	R/W	
2	POD26	0	R/W	
1	POD25	0	R/W	
0	POD24	0	R/W	

1	POD17	0	R/W
0	POD16	0	R/W

### 15.3.3 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
Bit Name	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9
Initial Value	0	0	0	0	0	0	0
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
Bit Name	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	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 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 with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be
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 with PCR.
1	NDR9	0	R/W	
0	NDR8	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 with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.

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 with PCR.
1	NDR1	0	R/W	
0	NDR0	0	R/W	



3	NDR27	0	R/W
2	NDR26	0	R/W
1	NDR25	0	R/W
0	NDR24	0	R/W

If pulse output groups 6 and 7 have different output triggers, the upper four bits and bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR31	0	R/W	Next Data Register 31 to 28
6	NDR30	0	R/W	The register contents are transferred to the corresponding PODRH_1 bits by the output trigger specified with PCR_1.
5	NDR29	0	R/W	
4	NDR28	0	R/W	
3 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1 and cannot be
3	NDR27	0	R/W	Next Data Register 27 to 24
2	NDR26	0	R/W	The register contents are transferred to the corresponding PODRH_1 bits by the output trigger specified with PCR_1.
1	NDR25	0	R/W	
0	NDR24	0	R/W	

3	NDR19	0	R/W
2	NDR18	0	R/W
1	NDR17	0	R/W
0	NDR16	0	R/W

If pulse output groups 4 and 5 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	NDR23	0	R/W	Next Data Register 23 to 20
6	NDR22	0	R/W	The register contents are transferred to the corresponding PODRL_1 bits by the output trigger specified with PCR_1.
5	NDR21	0	R/W	
4	NDR20	0	R/W	
3 to 0	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.

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	NDR19	0	R/W	Next Data Register 19 to 16
2	NDR18	0	R/W	The register contents are transferred to the corresponding PODRL_1 bits by the output trigger specified with PCR_1.
1	NDR17	0	R/W	
0	NDR16	0	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 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 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 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 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

4	G2CMS0	1	R/W	These bits select output trigger of pulse output g 00: Compare match in TPU channel 6 01: Compare match in TPU channel 7 10: Compare match in TPU channel 8 11: Compare match in TPU channel 9
3	G1CMS1	1	R/W	Group 5 Compare Match Select 1 and 0
2	G1CMS0	1	R/W	These bits select output trigger of pulse output g 00: Compare match in TPU channel 6 01: Compare match in TPU channel 7 10: Compare match in TPU channel 8 11: Compare match in TPU channel 9
1	G0CMS1	1	R/W	Group 4 Compare Match Select 1 and 0
0	G0CMS0	1	R/W	These bits select output trigger of pulse output g 00: Compare match in TPU channel 6 01: Compare match in TPU channel 7 10: Compare match in TPU channel 8 11: Compare match in TPU channel 9

Bit	Bit Name	Initial Value	R/W	Description
7	G3INV	1	R/W	Group 3 Inversion Selects direct output or inverted output for pulse group 3. 0: Inverted output 1: Direct output
6	G2INV	1	R/W	Group 2 Inversion Selects direct output or inverted output for pulse group 2. 0: Inverted output 1: Direct output
5	G1INV	1	R/W	Group 1 Inversion Selects direct output or inverted output for pulse group 1. 0: Inverted output 1: Direct output
4	G0INV	1	R/W	Group 0 Inversion Selects direct output or inverted output for pulse group 0. 0: Inverted output 1: Direct output

Selects normal or non-overlapping operation for output group 2.

0: Normal operation (output values updated at compare match A in the selected TPU channel)

1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)

---

1	G1NOV	0	R/W	Group 1 Non-Overlap Selects normal or non-overlapping operation for output group 1. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)
0	G0NOV	0	R/W	Group 0 Non-Overlap Selects normal or non-overlapping operation for output group 0. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)

---

Selects direct output or inverted output for pulse group 6.

0: Inverted output

1: Direct output

---

5	G1INV	1	R/W	Group 5 Inversion
---	-------	---	-----	-------------------

Selects direct output or inverted output for pulse group 5.

0: Inverted output

1: Direct output

---

4	G0INV	1	R/W	Group 4 Inversion
---	-------	---	-----	-------------------

Selects direct output or inverted output for pulse group 4.

0: Inverted output

1: Direct output

Selects normal or non-overlapping operation for output group 6.

0: Normal operation (output values updated by compare match A on the selected TPU channel)

1: Non-overlapping operation (output values updated by compare match A or B on the selected TPU channel)

---

1	G1NOV	0	R/W	Group 5 Non-Overlap
---	-------	---	-----	---------------------

Selects normal or non-overlapping operation for output group 5.

0: Normal operation (output values updated by compare match A on the selected TPU channel)

1: Non-overlapping operation (output values updated by compare match A or B on the selected TPU channel)

---

0	G0NOV	0	R/W	Group 4 Non-Overlap
---	-------	---	-----	---------------------

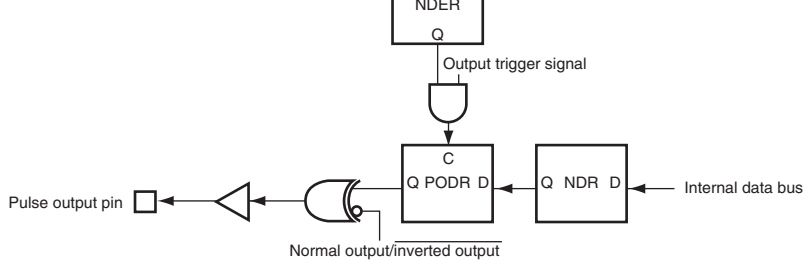
Selects normal or non-overlapping operation for output group 4.

0: Normal operation (output values updated by compare match A on the selected TPU channel)

1: Non-overlapping operation (output values updated by compare match A or B on the selected TPU channel)

---

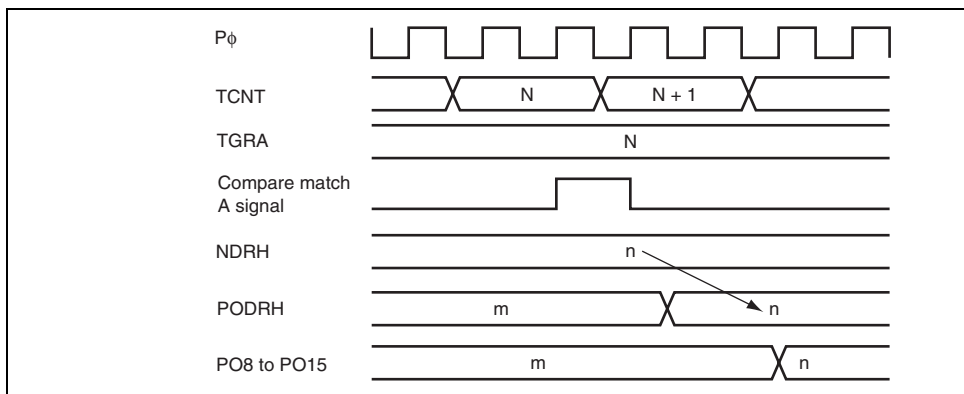




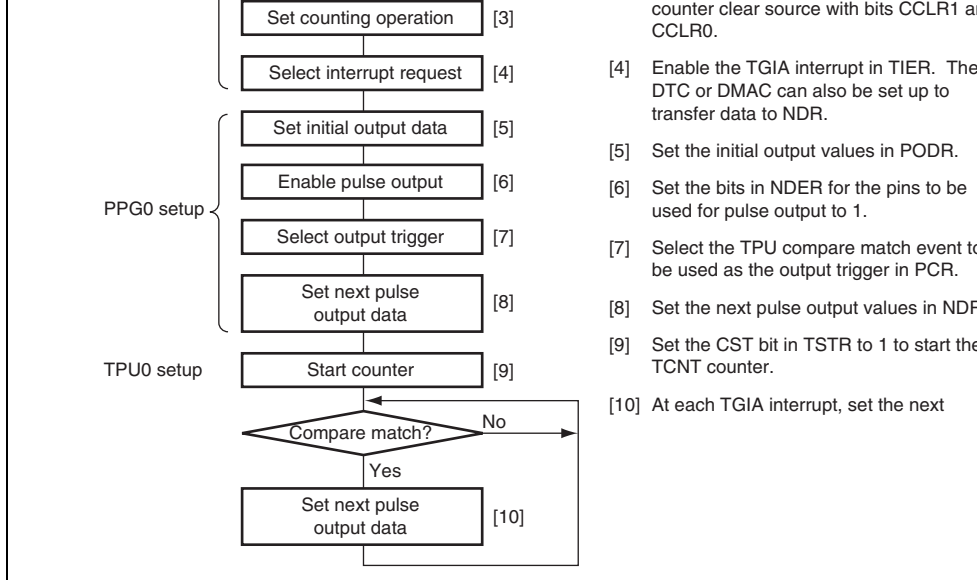
**Figure 15.3 Schematic Diagram of PPG**

### 15.4.1 Output Timing

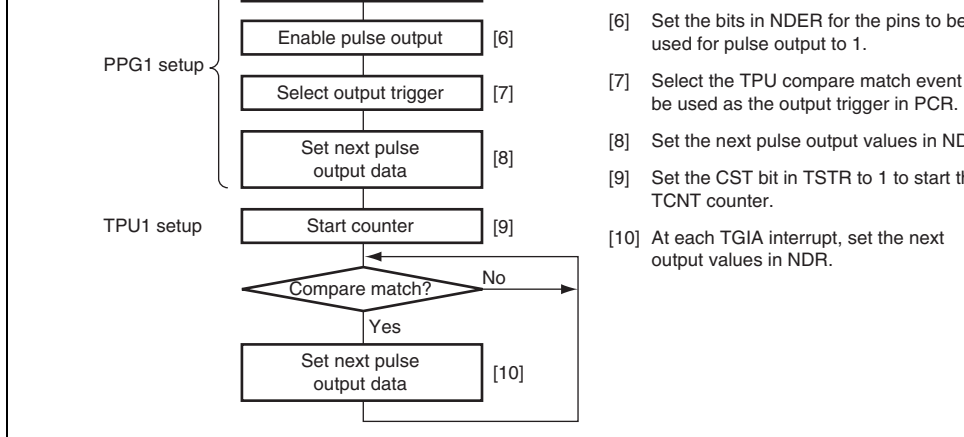
If pulse output is enabled, the NDR contents are transferred to PODR and output when the specified compare match event occurs. Figure 15.4 shows the timing of these operations in the case of normal output in groups 2 and 3, triggered by compare match A.



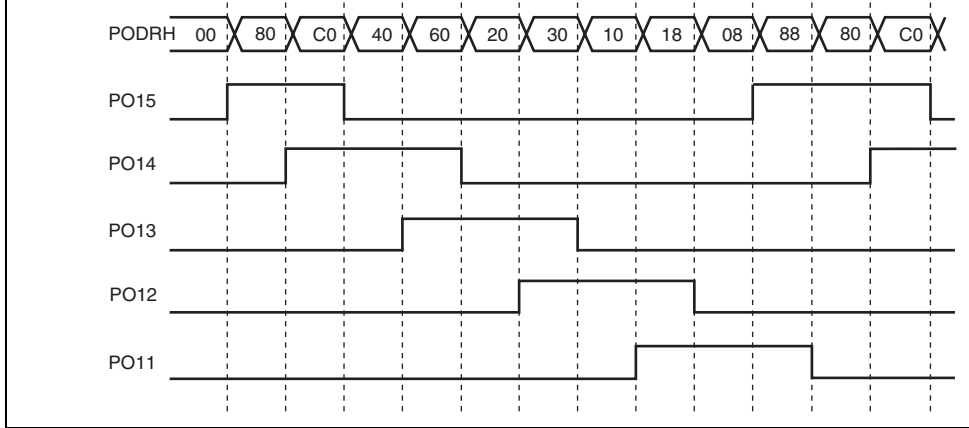
**Figure 15.4 Timing of Transfer and Output of NDR Contents (Example)**



**Figure 15.5 Setup Procedure for Normal Pulse Output (PPG0)**

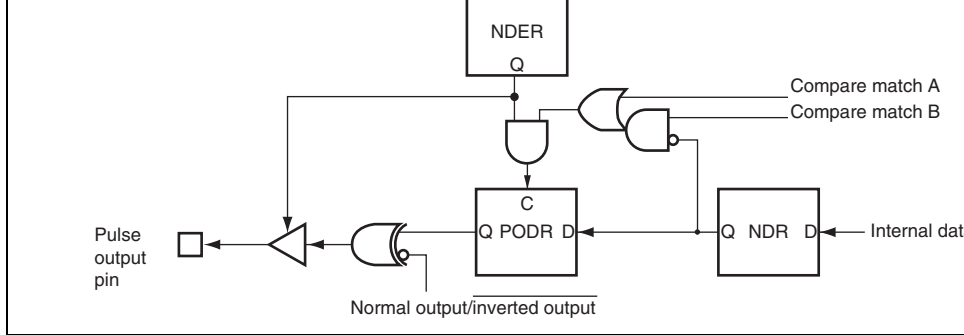


**Figure 15.6 Setup Procedure for Normal Pulse Output (PPG1)**



**Figure 15.7 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 the period of a cycle in TGRA so the counter will be cleared by compare match A. Set the TGIEA and 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 NDERL 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. If the DTC or DMAC is set for activation by the TGIA interrupt, pulse output can be achieved without imposing a load on the CPU.

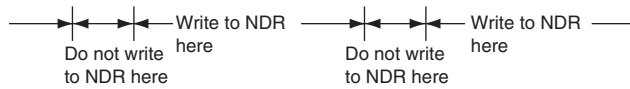


**Figure 15.8 Non-Overlapping Pulse Output**

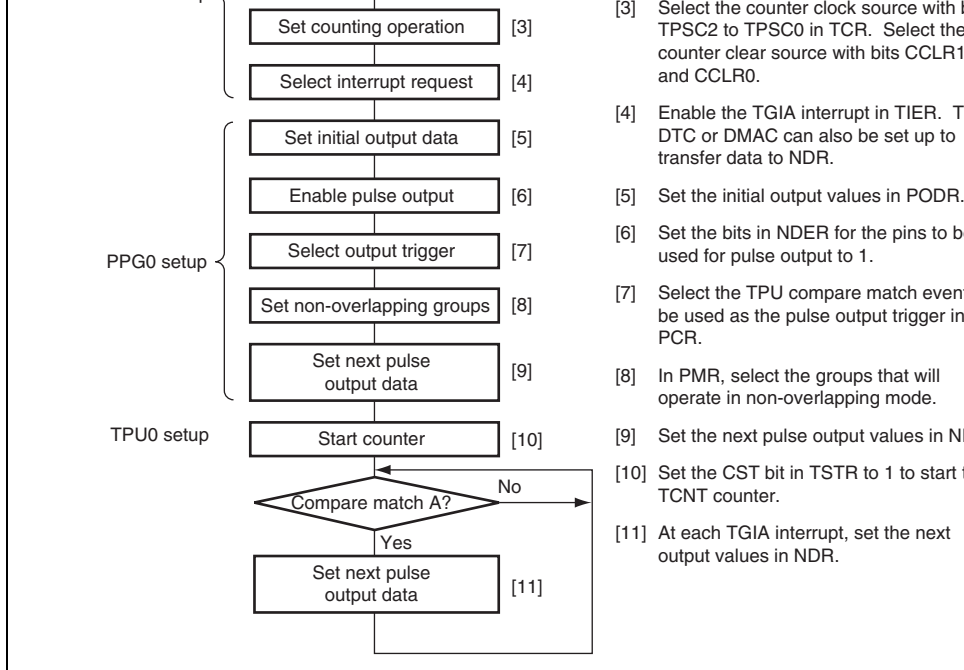
Therefore, 0 data can be transferred ahead of 1 data by making compare match B occur before compare match A.

The NDR contents should not be altered during the interval from compare match B to compare match A (the non-overlapping margin).

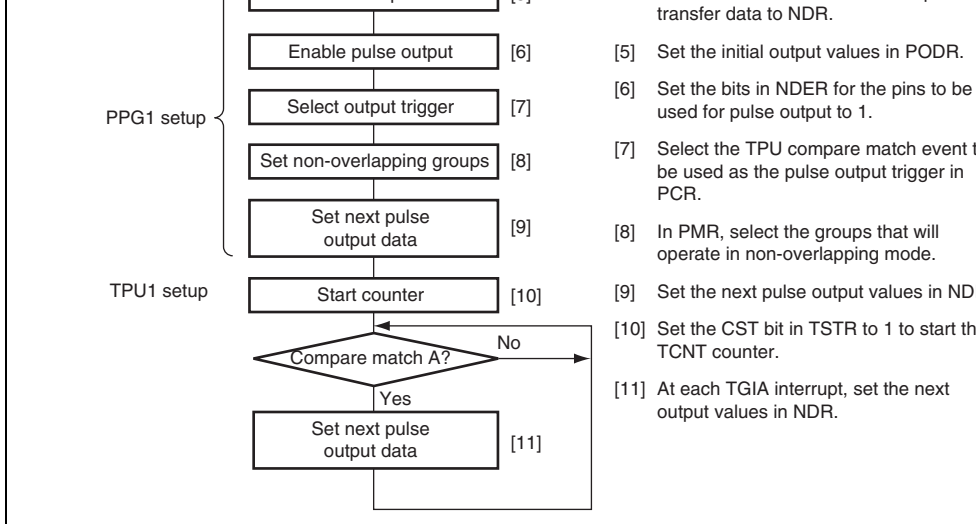
This can be accomplished by having the TGIA interrupt handling routine write the next NDR, or by having the TGIA interrupt activate the DTC or DMAC. Note, however, that data must be written before the next compare match B occurs.



**Figure 15.9 Non-Overlapping Operation and NDR Write Timing**

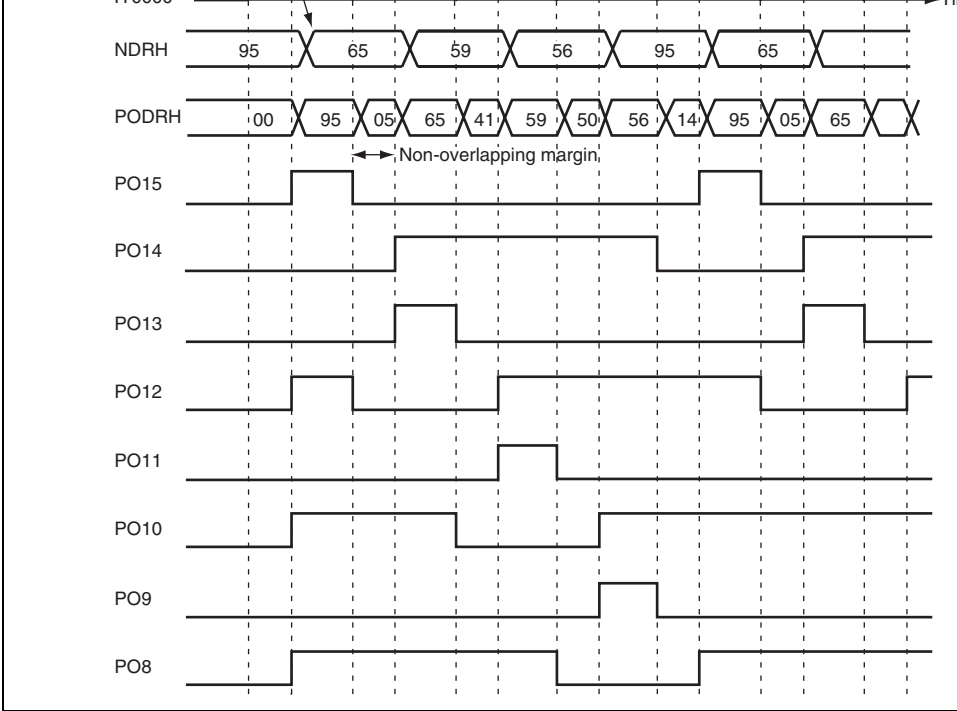


**Figure 15.10 Setup Procedure for Non-Overlapping Pulse Output (PPG0)**



**Figure 15.11 Setup Procedure for Non-Overlapping Pulse Output (PPG1)**





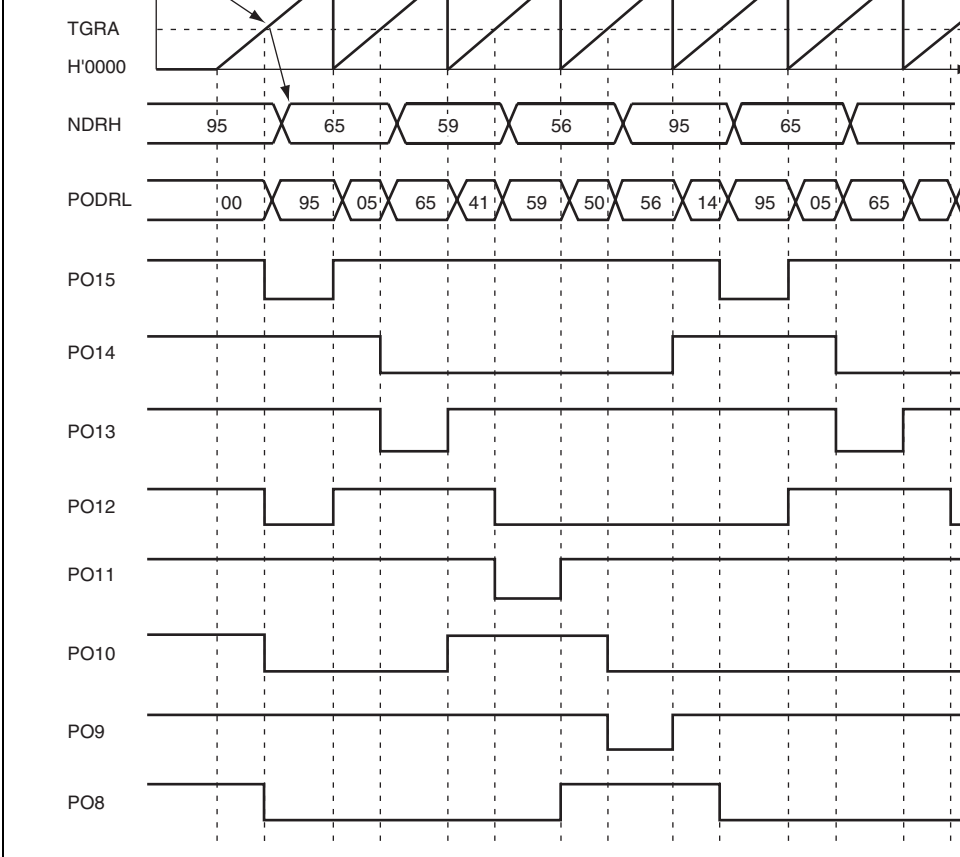
**Figure 15.12 Non-Overlapping Pulse Output Example (4-Phase Complemen**

to 1 (the change from 0 to 1 is delayed by the value set in 1GRA).

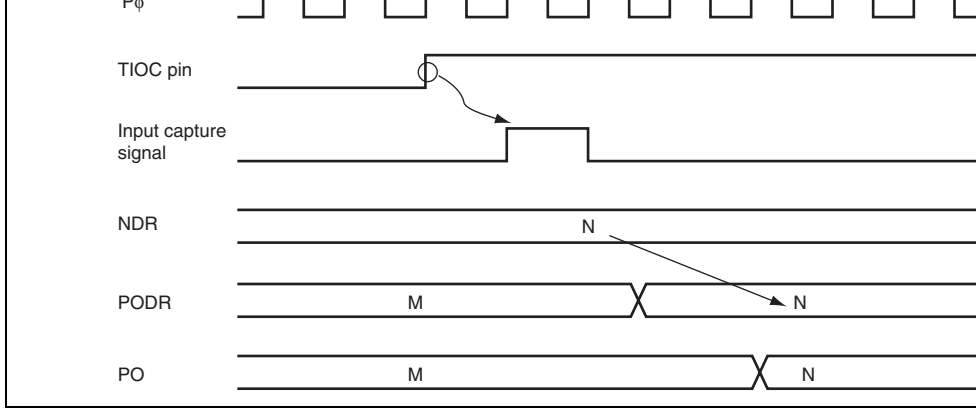
The TGIA interrupt handling routine writes the next output data (H'65) to NDRH.

4. 4-phase complementary non-overlapping pulse output can be obtained subsequently by 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.



**Figure 15.13 Inverted Pulse Output (Example)**



**Figure 15.14 Pulse Output Triggered by Input Capture (Example)**

Pins PO0 to PO15 are also used for other peripheral functions such as the TPU. When 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 pins.

Pin functions should be changed only under conditions in which the output trigger event occurs.

### **15.5.3 TPU Setting when PPG1 is in Use**

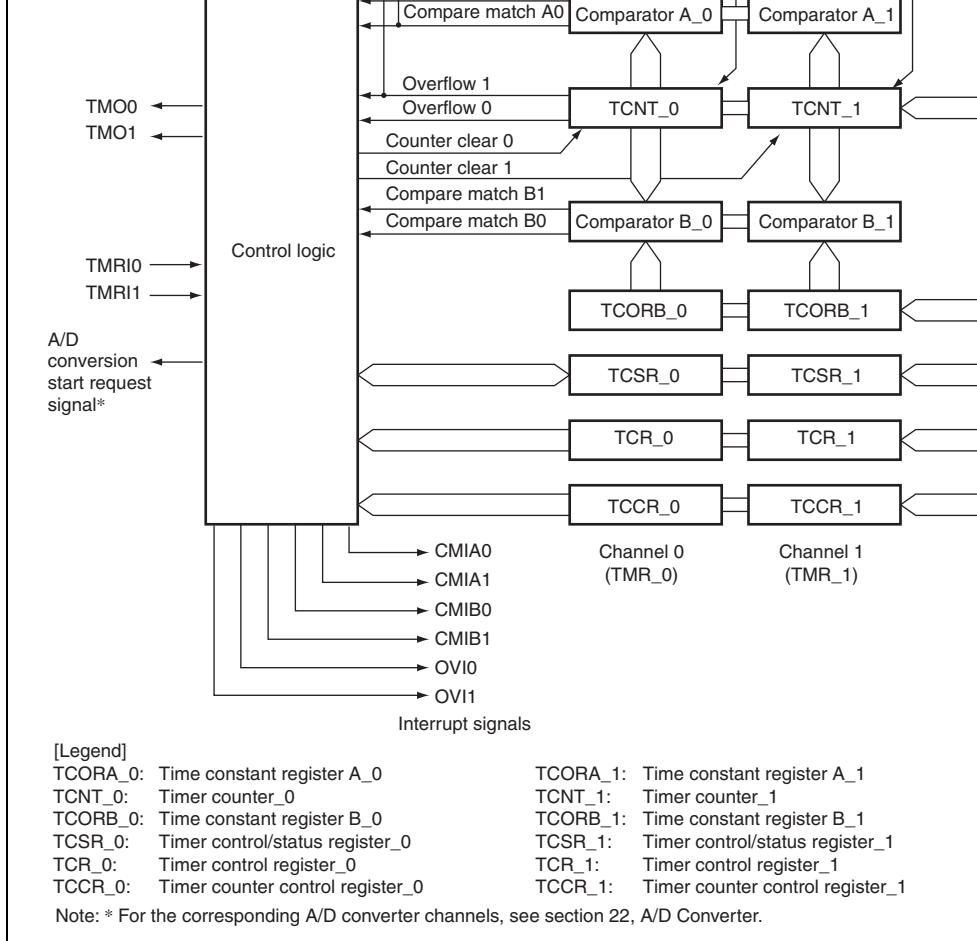
When using PPG1, output toggling on compare-matches must be specified in the TIOR register. The TPU that acts as the activation source and output must be selected as the PPG1 function.



the same functions. Unit 2 and unit 3 can generate baud rate clock for SCI and have the functions.

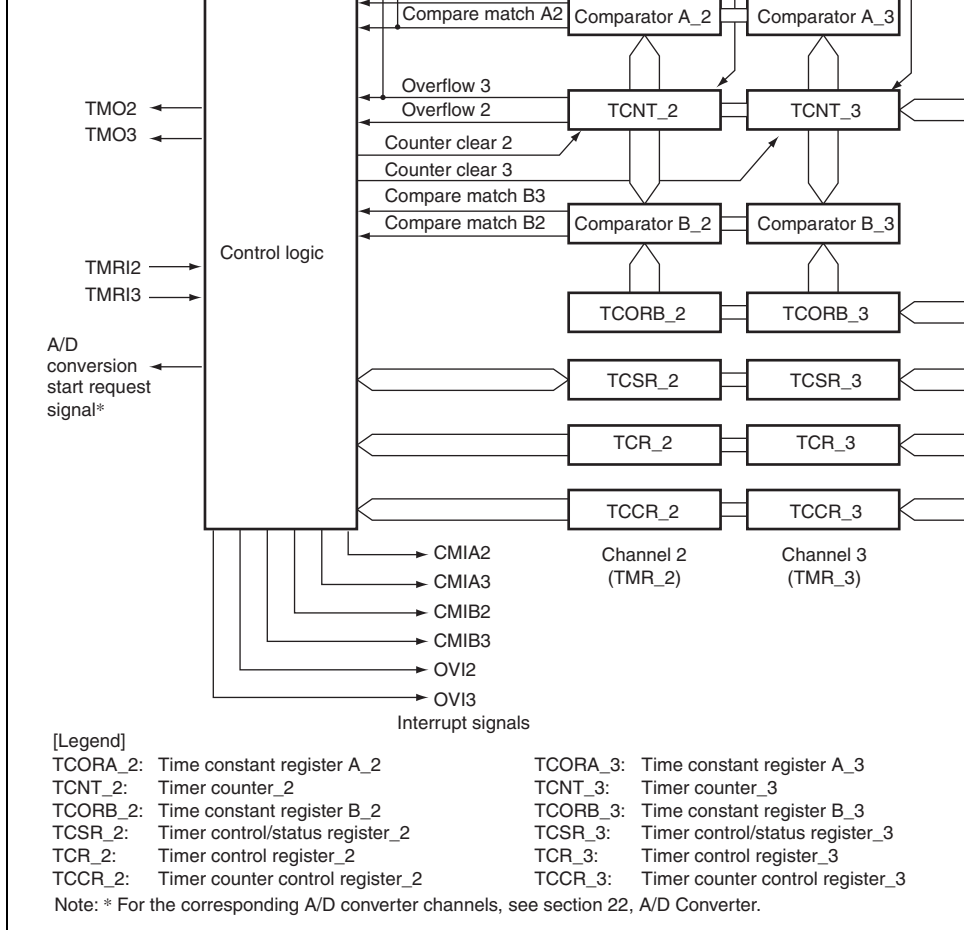
## 16.1 Features

- 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 unit 0 and unit 1).
- 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 (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 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 (This is available only in unit 0 and unit 1.)
- Generation of trigger to start A/D converter conversion (available in unit 0 to unit 3)
- Capable of generating baud rate clock for SCI\_5 and SCI\_6. (This is available only in unit 2 and unit 3). For details, see section 19, Serial Communication Interface (SCI, IrDA, IrDA).
- Module stop state specifiable

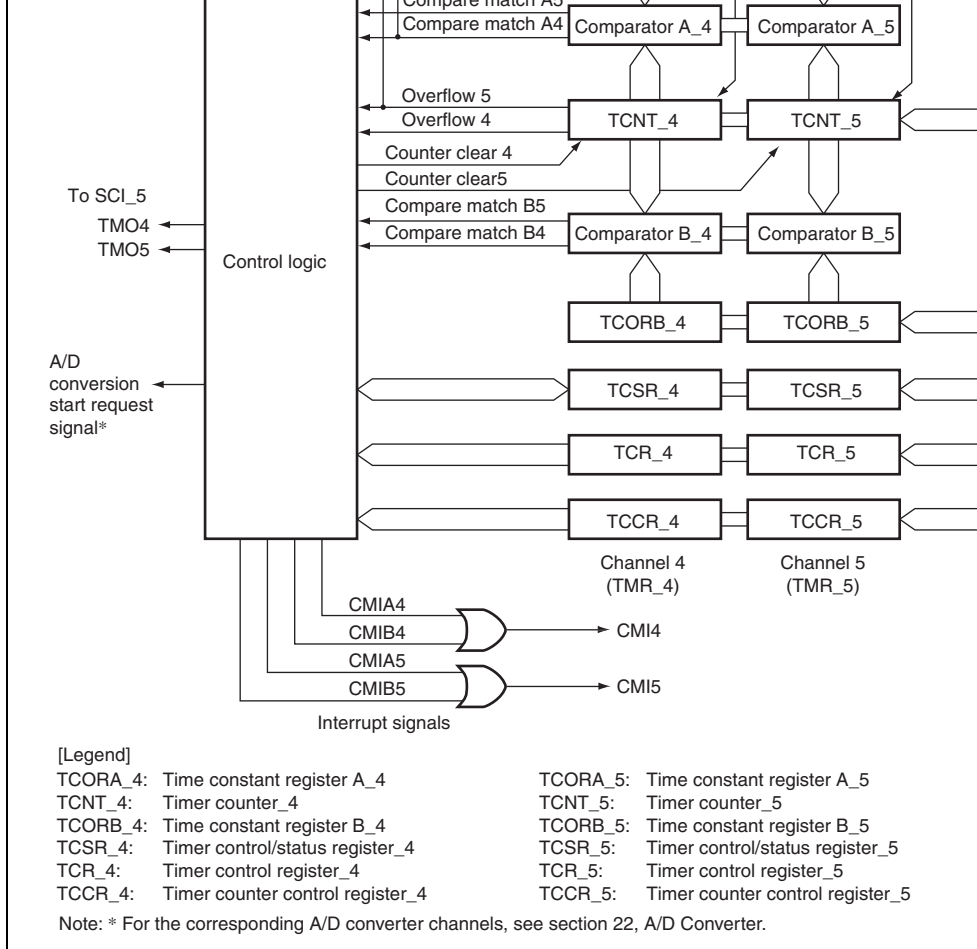


**Figure 16.1 Block Diagram of 8-Bit Timer Module (Unit 0)**

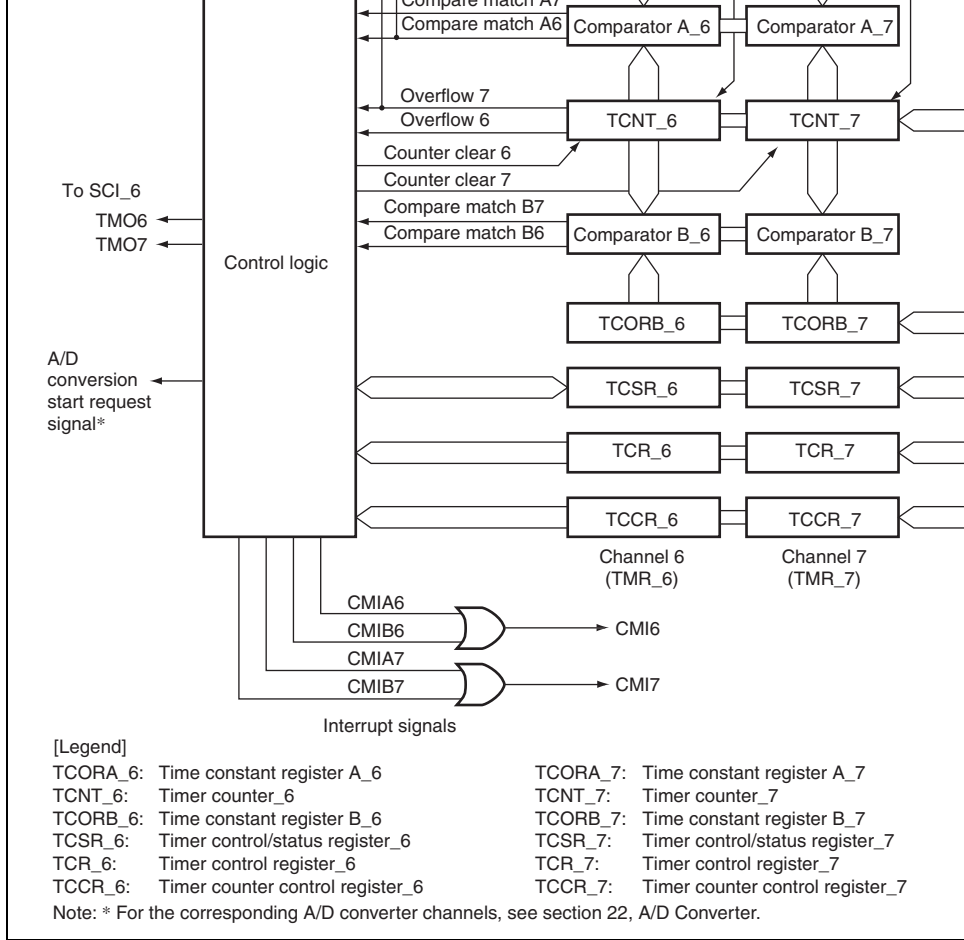




**Figure 16.2 Block Diagram of 8-Bit Timer Module (Unit 1)**



**Figure 16.3 Block Diagram of 8-Bit Timer Module (Unit 2)**



**Figure 16.4 Block Diagram of 8-Bit Timer Module (Unit 3)**

	1	Timer output pin	TMO1	Output	Outputs compare match
		Timer clock input pin	TMCI1	Input	Inputs external clock for co
		Timer reset input pin	TMRI1	Input	Inputs external reset to cou
1	2	Timer output pin	TMO2	Output	Outputs compare match
		Timer clock input pin	TMCI2	Input	Inputs external clock for co
		Timer reset input pin	TMRI2	Input	Inputs external reset to cou
	3	Timer output pin	TMO3	Output	Outputs compare match
		Timer clock input pin	TMCI3	Input	Inputs external clock for co
		Timer reset input pin	TMRI3	Input	Inputs external reset to cou
2	4	—	—	—	—
	5				
3	6				
	7				

- 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)

- 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)

Bit Name															
Initial Value	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

### 16.3.2 Time Constant Register A (TCORA)

TCORA is an 8-bit readable/writable register. TCORA\_0 and TCORA\_1 comprise a single 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 TCORA write cycle. The timer output from the TMO pin can be freely controlled by this match signal (compare match A) and the settings of bits OS1 and OS0 in TCSR. TCORA is initialized to H'FF.

	TCORA_0								TCORA_1						
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	
Bit Name															
Initial Value	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	

Bit Name																	
Initial Value	1	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	R/W

### 16.3.4 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	
Bit Name	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	
Initial Value	0	0	0	0	0	0	0	
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 (CMIEB) are enabled or disabled when the CMFB flag in TCSR is set to 1. *2 0: CMFB interrupt requests (CMIEB) are disabled 1: CMFB interrupt requests (CMIEB) are enabled



enabled or disabled when the OVI flag in TCCR 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 set to 1 or 0) of the external reset input or when the external reset input is high (TMRIS in TCCR is set to 1)
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 counter condition. See table 16.2.
0	CKS0	0	R/W	

---

- Notes:
1. To use an external reset or external clock, the DDR and ICR bits in the corresponding I/O Port Register should be set to 0 and 1, respectively. For details, see section 13, I/O Port Registers.
  2. In unit 2 and unit 3, one interrupt signal is used for CMIEB or CMIEA. For details, see section 16.7, Interrupt Sources.
  3. Available only in unit 0 and unit 1.

Bit	Bit Name	Value	R/W	Description
7 to 4	—	All 0	R	Reserved These bits are always read as 0. It should not be set
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
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 select the internal clock. See table 16.2.

Note: \* Available only in unit 0 and unit 1. The write value should always be 0 in unit 2 and 3.

			1	0	Uses internal clock. Counts at falling edge of
			1	1	Uses internal clock. Counts at rising edge of
	0	1	1	0	Uses internal clock. Counts at rising edge of
			0	1	Uses internal clock. Counts at rising edge of
			1	0	Uses internal clock. Counts at falling edge of
			1	1	Uses internal clock. Counts at falling edge of
	1	0	0	—	Counts at TCNT_1 overflow signal* <sup>1</sup> .
TMR_1	0	0	0	—	Clock input prohibited
	0	0	1	0	Uses internal clock. Counts at rising edge of
			0	1	Uses internal clock. Counts at rising edge of
			1	0	Uses internal clock. Counts at falling edge of
			1	1	Uses internal clock. Counts at falling edge of
	0	1	0	0	Uses internal clock. Counts at rising edge of
			0	1	Uses internal clock. Counts at rising edge of
			1	0	Uses internal clock. Counts at falling edge of
			1	1	Uses internal clock. Counts at falling edge of
	0	1	1	0	Uses internal clock. Counts at rising edge of
			0	1	Uses internal clock. Counts at rising edge of
			1	0	Uses internal clock. Counts at falling edge of
			1	1	Uses internal clock. Counts at falling edge of
	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 the 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 13, I/O Ports.

			1	0	Uses internal clock. Counts at falling edge of P
			1	1	Uses internal clock. Counts at falling edge of P
	0	1	1	0	Uses internal clock. Counts at rising edge of P
			0	1	Uses internal clock. Counts at rising edge of P
			1	0	Uses internal clock. Counts at falling edge of P
			1	1	Uses internal clock. Counts at falling edge of P
	1	0	0	—	Counts at TCNT_3 overflow signal* <sup>1</sup> .
TMR_3	0	0	0	—	Clock input prohibited
	0	0	1	0	Uses internal clock. Counts at rising edge of P
			0	1	Uses internal clock. Counts at rising edge of P
			1	0	Uses internal clock. Counts at falling edge of P
			1	1	Uses internal clock. Counts at falling edge of P
	0	1	0	0	Uses internal clock. Counts at rising edge of P
			0	1	Uses internal clock. Counts at rising edge of P
			1	0	Uses internal clock. Counts at falling edge of P
			1	1	Uses internal clock. Counts at falling edge of P
	0	1	1	0	Uses internal clock. Counts at rising edge of P
			0	1	Uses internal clock. Counts at rising edge of P
			1	0	Uses internal clock. Counts at falling edge of P
			1	1	Uses internal clock. Counts at falling edge of P
	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 1 is the TCNT\_2 compare match signal, no incrementing clock is generated. Do not use the ICR bit 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 13, I/O Ports.

			1	0	Uses internal clock. Counts at rising edge of	
			1	1	Uses internal clock. Counts at falling edge of	
	0	1	1	0	0	Uses internal clock. Counts at rising edge of
				0	1	Uses internal clock. Counts at rising edge of
				1	0	Uses internal clock. Counts at rising edge of
				1	1	Uses internal clock. Counts at falling edge of
	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
				0	1	Uses internal clock. Counts at rising edge of
				1	0	Uses internal clock. Counts at falling edge of
				1	1	Uses internal clock. Counts at falling edge of
	0	1	0	0	0	Uses internal clock. Counts at rising edge of
				0	1	Uses internal clock. Counts at rising edge of
				1	0	Uses internal clock. Counts at falling edge of
				1	1	Uses internal clock. Counts at falling edge of
	0	1	1	0	0	Uses internal clock. Counts at rising edge of
				0	1	Uses internal clock. Counts at rising edge of
				1	0	Uses internal clock. Counts at rising edge of
				1	1	Uses internal clock. Counts at falling edge of
	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 set the setting.

			1	0	Uses internal clock. Counts at falling edge of P	
			1	1	Uses internal clock. Counts at falling edge of P	
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P
				0	1	Uses internal clock. Counts at rising edge of P
				1	0	Uses internal clock. Counts at rising edge of P
				1	1	Uses internal clock. Counts at falling edge of P
	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
				0	1	Uses internal clock. Counts at rising edge of P
				1	0	Uses internal clock. Counts at falling edge of P
				1	1	Uses internal clock. Counts at falling edge of P
	0	1	0	0	0	Uses internal clock. Counts at rising edge of P
				0	1	Uses internal clock. Counts at rising edge of P
				1	0	Uses internal clock. Counts at falling edge of P
				1	1	Uses internal clock. Counts at falling edge of P
	0	1	1	0	0	Uses internal clock. Counts at rising edge of P
				0	1	Uses internal clock. Counts at rising edge of P
				1	0	Uses internal clock. Counts at rising edge of P
				1	1	Uses internal clock. Counts at falling edge of P
	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.

• TCSR\_1

Bit	7	6	5	4	3	2	1
Bit Name	CMFB	CMFA	OVF	—	OS3	OS2	OS1
Initial Value	0	0	0	1	0	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R	R/W	R/W	R/W

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

• TCSR\_0, TCSR4

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 while the corresponding interrupt is enabled to read the flag after writing 0 to it.)</li> <li>• When the DTC is activated by a CMIB interrupt, the DISEL bit in MRB of the DTC is 0*<sup>3</sup></li> </ul>

- When the DTC is activated by a CMIA interrupt, the DISEL bit in MRB in the DTC is 0\*<sup>3</sup>

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</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	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 disabled</p> <p>1: A/D converter start requests by compare match A 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>



- 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 when a compare match occurs after a reset.
  3. For the corresponding A/D converter channels, see section 22, A/D Converter.

- TCSR\_1, TCSR\_5

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 to it while the corresponding interrupt is enabled to read the flag after writing 0 to it.)</li> <li>• When the DTC is activated by a CMIB interrupt and the DISEL bit in MRB of the DTC is 0<sup>*3</sup></li> </ul>

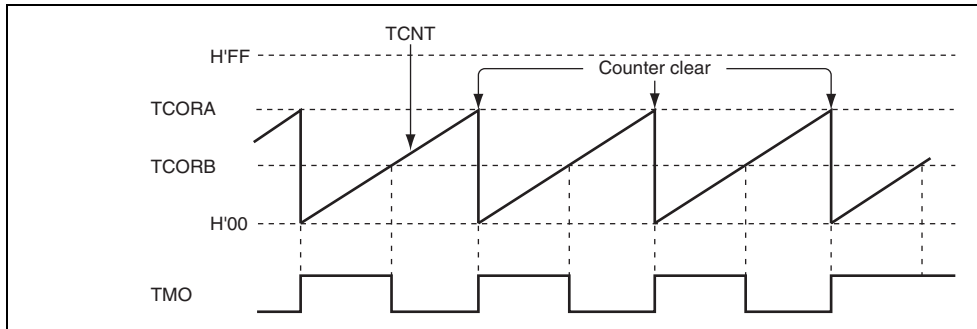
- When the DTC is activated by a CMIA interrupt, the DISEL bit in MRB of the DTC is 0\*<sup>3</sup>

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>Cleared by reading OVF when OVF = 1, then writing 0 to OVF</p> <p>(When the CPU is used to clear this flag by writing 0 to OVF while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
4	—	1	R	<p>Reserved</p> <p>This bit is always read as 1 and cannot be modified.</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 a 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>

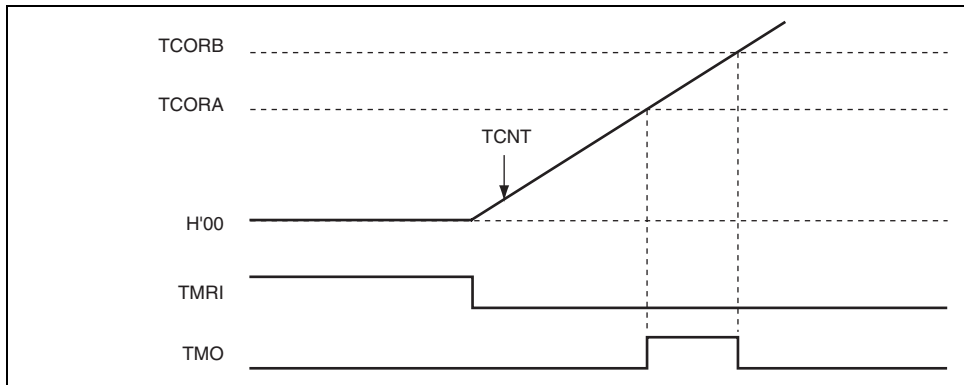
- Notes:
1. Only 0 can be written to bits 7 to 5, to clear these flags.
  2. Timer output is disabled when bits OS3 to OS0 are all 0. Timer output is 0 u compare match occurs after a reset.
  3. Available only in unit 0 and unit 1.

compare match and to 0 at a TCORB compare match.

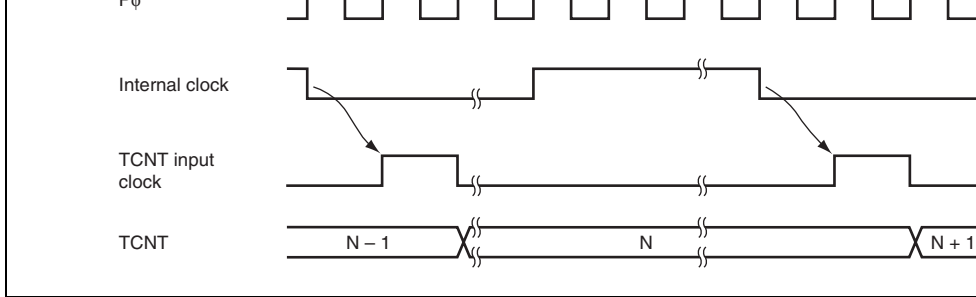
With these settings, the 8-bit timer provides pulses output at a cycle determined by TCORA pulse width determined by TCORB. No software intervention is required. The timer outputs until the first compare match occurs after a reset.



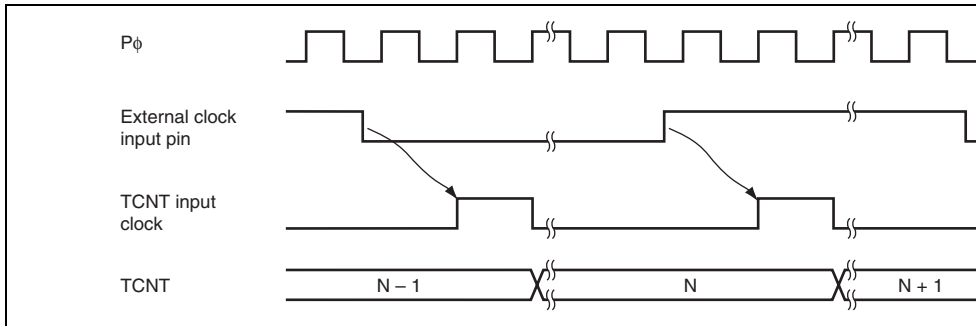
**Figure 16.5 Example of Pulse Output**



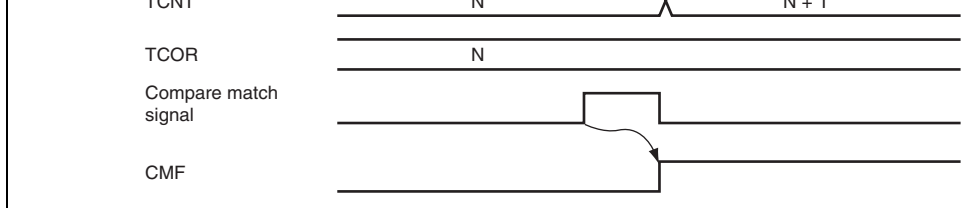
**Figure 16.6 Example of Reset Input**



**Figure 16.7 Count Timing for Internal Clock Input**



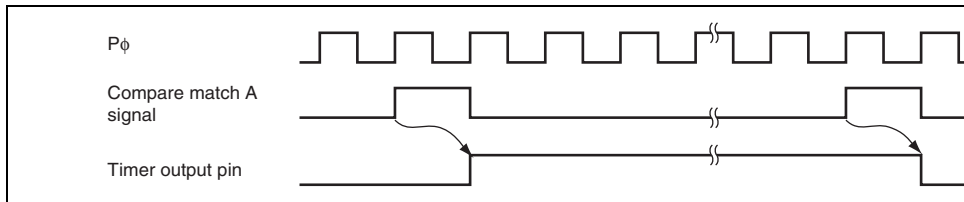
**Figure 16.8 Count Timing for External Clock Input**



**Figure 16.9 Timing of CMF Setting at Compare Match**

### 16.5.3 Timing of Timer Output at Compare Match

When a compare match signal is generated, the timer output changes as specified by the OS0 bit in TCSR. Figure 16.10 shows the timing when the timer output is toggled by the match A signal.

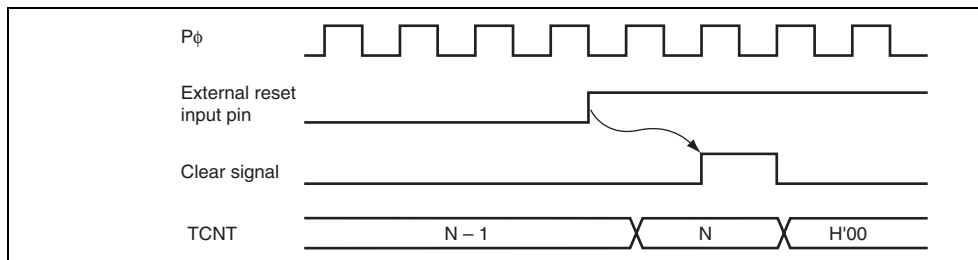


**Figure 16.10 Timing of Toggled Timer Output at Compare Match A**

### 16.5.5 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 16.12 and figure 16.13 shows the timing of this operation.

Note: \* Clearing by an external reset is available only in units 0 and 1.

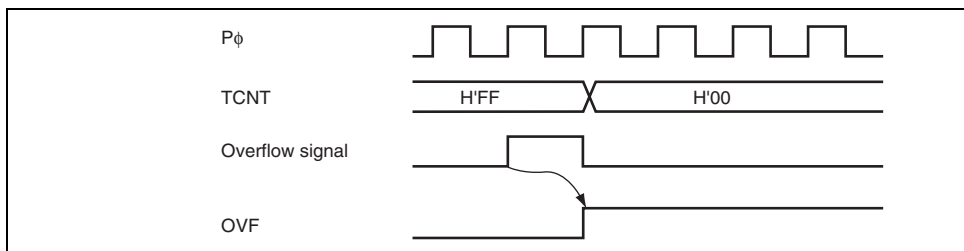


**Figure 16.12 Timing of Clearance by External Reset (Rising Edge)**



### 16.5.6 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 16.14 shows the timing of this operation.



**Figure 16.14 Timing of OVF Setting**

### (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 is not set if the TMRI0 pin has been set.
- The settings of the CCLR1 and CCLR0 bits in TCR\_1 are ignored. The lower 8 bits of the counter are 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.

## 16.6.2 Compare Match Count Mode

When the bits CKS2 to CKS0 in TCR\_1 are set to B'100, TCNT\_1 counts compare match events in 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.

**Table 16.6 8-Bit Timer (TMR\_0 or TMR\_1) Interrupt Sources (in Unit 0 and Unit 1)**

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

- Interrupt in unit 2 and unit 3

There are two interrupt sources for the 8-bit timer (TMR\_4 or TMR\_5): CMIA, CMIB. The interrupt signal is CMI only. The interrupt sources are shown in table 16.7. When enabling the interrupt, the interrupt enabling bit in TCR or TCSR, and when either CMIA or CMIB interrupt source is generated, CMI is sent to the interrupt controller. To verify which interrupt source is generated, confirm by checking each flag in TCSR. No overflow-related interrupt exists. DTC cannot be activated by this interrupt.

The A/D converter can be activated by a compare match A for the even channels of each unit. \*

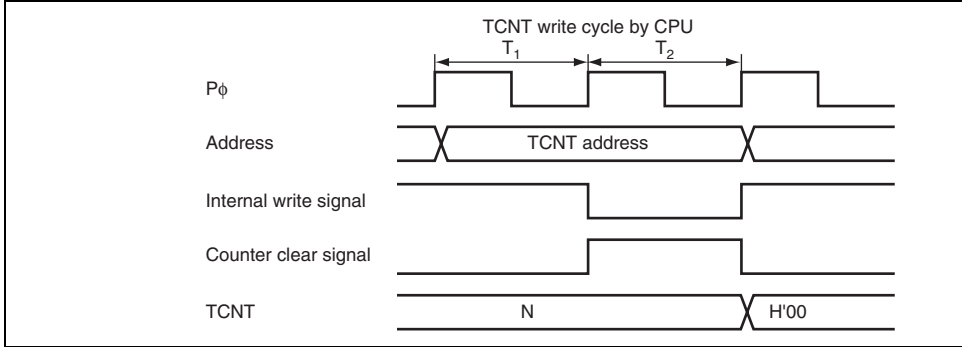
If the ADTE bit in TCSR is set to 1 when the CMFA flag in TCSR is set to 1 by the occurrence of a compare match A, a request to start A/D conversion is sent to the A/D converter. If the timer conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

Note: \* For the corresponding A/D converter channels, see section 22, A/D Converter

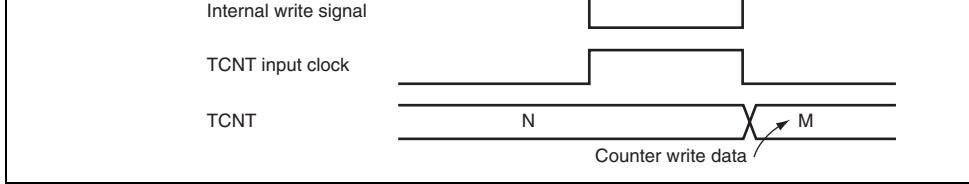
f: Counter frequency  
 $\phi$ : Operating frequency  
N: TCOR value

### 16.8.2 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 has priority and the write is not performed as shown in figure 16.15.



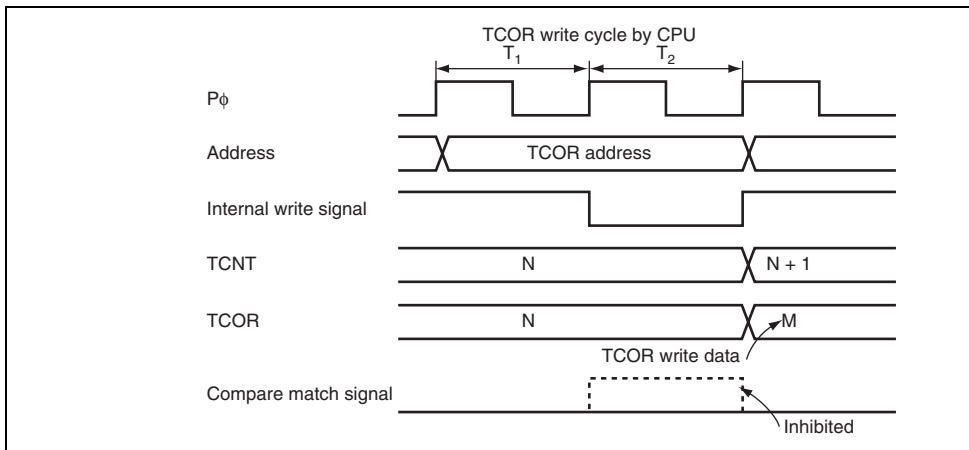
**Figure 16.15 Conflict between TCNT Write and Clear**



**Figure 16.16 Conflict between TCNT Write and Increment**

#### 16.8.4 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 priority and the compare match signal is inhibited as shown in figure 16.17.



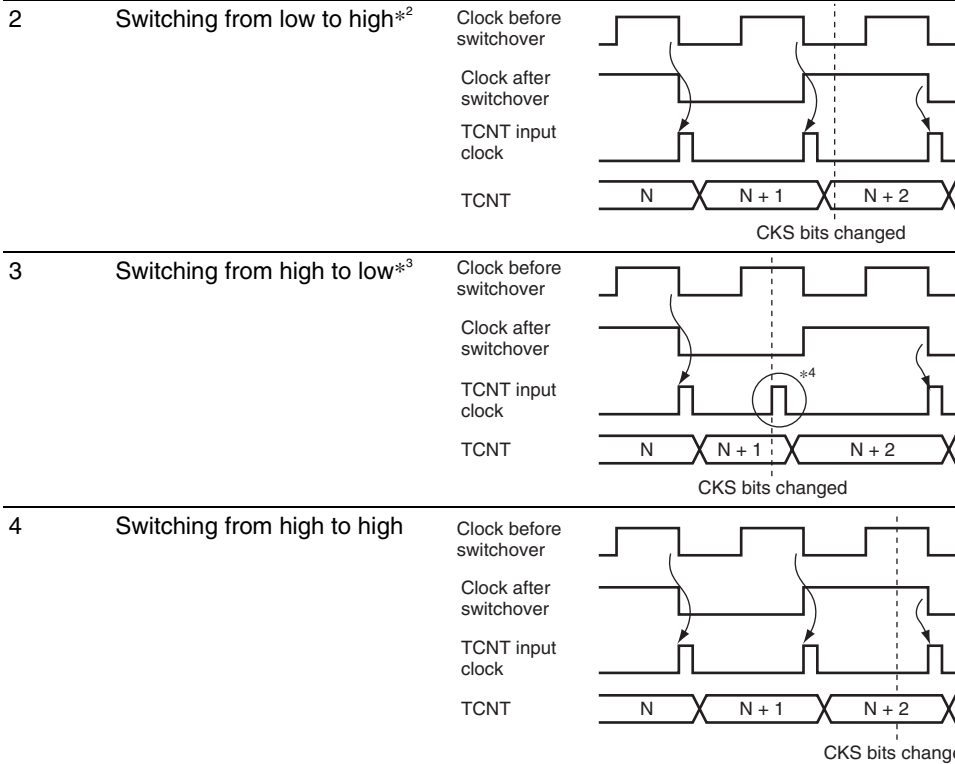
**Figure 16.17 Conflict between TCOR Write and Compare Match**

### 16.8.6 Switching of Internal Clocks and TCNT Operation

TCNT may be incremented erroneously depending on when the internal clock is switched. Figure 16.9 shows the relationship between the timing at which the internal clock is switched (from CKS0 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 clock pulse are always monitored. Table 16.9 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 Figure 16.9, 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.



- 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.



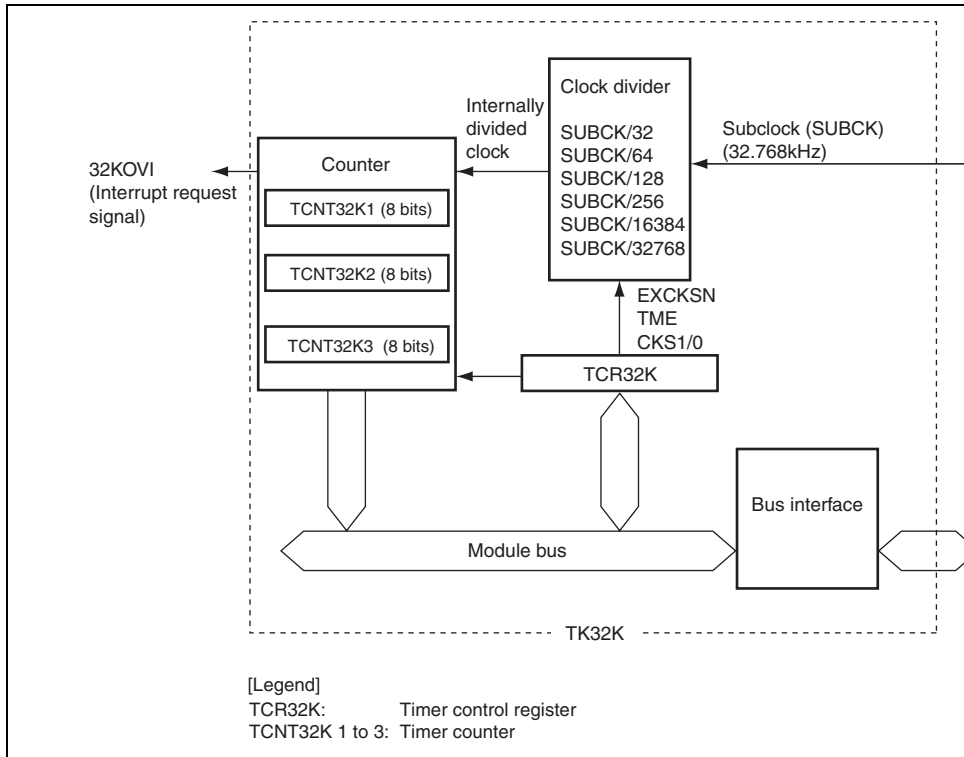
module stop state. For details, see section 28, Power-Down Modes.

### **16.8.9 Interrupts in Module Stop State**

If the module stop state is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DTC activation source. Interrupts should therefore be disabled before entering the module stop state.



- Counter operational except in hardware standby mode or the reset state.
- Canceling software standby mode and deep software standby mode is possible.



**Figure 17.1 Block Diagram of TM32K**

to TCNT32K. Change values of the bits when TME=0.

Bit	7	6	5	4	3	2	1
Bit Name	EXCKSN	—	TME	—	—	OSC32STP	CKS1
Initial Value	1	1	0	1	1	0	0
R/W	R/W	R	R/W	R	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	EXCKSN	1	R/W	Extended Clock Select and Clock Select 1, 0
1	CKS1	0	R/W	These bits select the internally divided clock source input to TCNT32K. The interrupt cycle for SUBCK 32.768 kHz is indicated in parentheses. When EXCKSN = 1: 00: Clock SUBCK/32 (cycle: 250 ms) 01: Clock SUBCK/64 (cycle: 500 ms) 10: Clock SUBCK/128 (cycle: 1 s) 11: Clock SUBCK/256 (cycle: 2 s) When EXCKSN = 0: 00: Clock SUBCK/16384 (cycle: 30 s) 01: Clock SUBCK/32768 (cycle: 60 s) 10: Clock SUBCK/16384 (cycle: approx. 22.7 da 11: Clock SUBCK/32768 (cycle: approx. 45.5 da
0	CKS0	0	R/W	
6	—	1	R	Reserved This bit is always read as 1 and cannot be modified.

## 17.2.2 Timer Counter (TCNT32K1, TCNT32K2, TCNT32K3)

The timer counter is a 24-bit counter for which either 8- or 24-bit operation is selectable. Allocation to the registers of the timer counter differs according to the setting of the EX in TCR32K.

### (1) EXCKSN = 1 (Initial State)

The timer counter operates as an 8-bit counter. Only TCNT32K1 is used and TCNT32K2, TCNT32K3 become reserved registers. Clearing the TME bit in the timer control register (TCR32K) initializes TCNT32K1 to H'00.

- TCNT32K1

Bit	7	6	5	4	3	2	1
Bit Name	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1
Initial value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

- TCNT32K2

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

- TCNT32K1

Bit	7	6	5	4	3	2	1
Bit Name	TCNT23	TCNT22	TCNT21	TCNT20	TCNT19	TCNT18	TCNT17
Initial value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

- TCNT32K2

Bit	7	6	5	4	3	2	1
Bit Name	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9
Initial value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

- TCNT32K3

Bit	7	6	5	4	3	2	1
Bit Name	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1
Initial value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

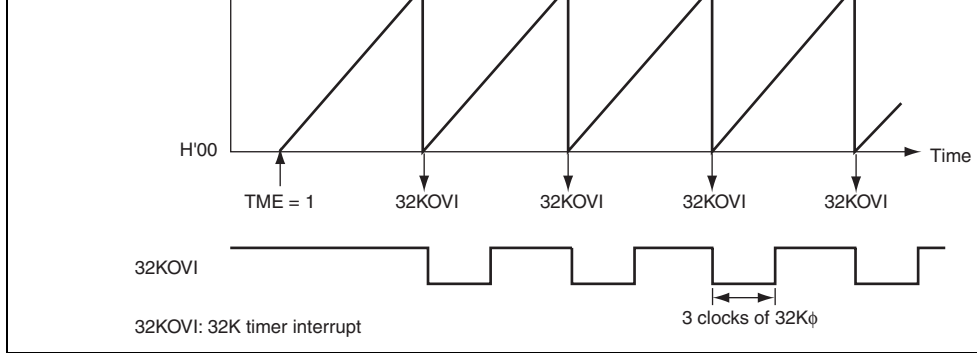
Note: A correct value cannot be read if the counter is read while the subclock oscillator operation (OSC32STP = 1).

CKS1, and CKS2 in TCNT2H and the interrupt cycles.

**Table 17.1 Relationships between Counter Operations and 32KOV1 Cycles**

Setting			Division Ratio	Selected Counter			Interrupt Cycles	
EXCKSN	CKS1	CKS2	Internally Divided CLK	TCNT 32K1	TCNT 32K2	TCNT 32K3	Counter value when interrupt generates	32KOV1
1	0	0	SUBCK/32	√	–	–	TCNT32K1=H'FF	250 m
1	0	1	SUBCK/64	√	–	–		500 m
1	1	0	SUBCK/128	√	–	–		1 s
1	1	1	SUBCK/256	√	–	–		2 s
0	0	0	SUBCK/16384	√	√	√	TCNT32K3=H'3B	30 s
0	0	1	SUBCK/32768	√	√	√		60 s
0	1	0	SUBCK/16384	√	√	√	TCNT32K1 to 3 =H'FFF3B	Approx. days
0	1	1	SUBCK/32768	√	√	√		Approx. days

√ : in use      – : not in use



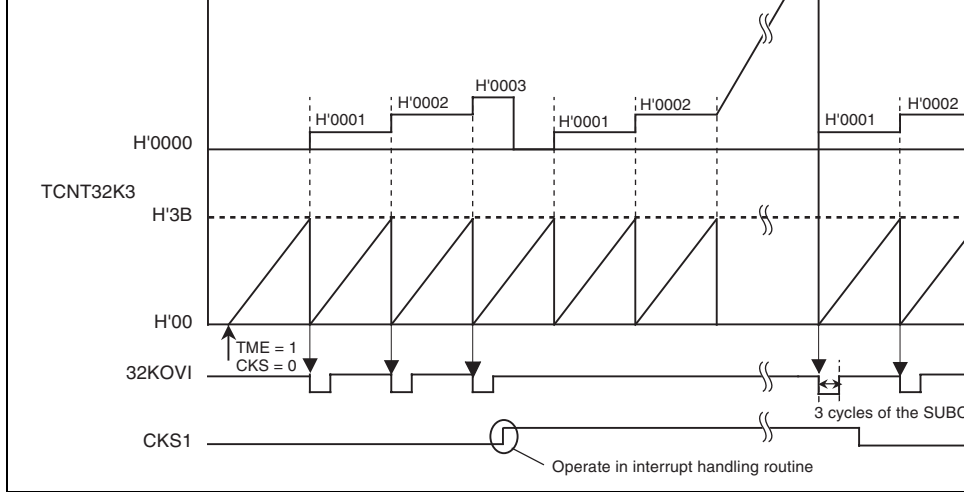
**Figure 17.2 Operation of 32K Timer when EXCKSN = 1**

### 17.3.3 EXCKSN=0 Operation

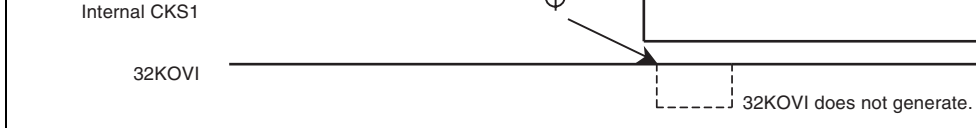
When the EXCKSN bit in TCR32K is 0, the timer counter operates as a 24-bit counter. The lower-order 8-bits are TCNT32K3, which operates as a free-running up-counter that counts from H'3B. The higher-order 16-bits in TCNT32K1 and TCNT32K2 act as an up-counter that counts from H'00\_00 to H'FFFF3B the number of times TCNT32K3=H'3B. Generation of the 32KOV1 interrupt either for all times TCNT32K3=H'3B or when TCNT32K1 to TCNT32K3=H'FFF3B can be selected by the setting of the CKS1 bit. This setting in combination with the setting of the CKS0 bit can be used to set the interrupt interval to 30s, 60s, approx. 22.7 days, or approx. 45.5 days.

When the EXCKSN bit is 0, the CKS1 bit can be changed while the timer counter is operating because the TME bit is 1. This allows variation of the 32KOV1 interrupt interval while the counter is in operation. When the value of the CKS1 bit is from 0 to 1, the higher-order 16-bit counter (TCNT32K1, TCNT32K2) resumes counting after being initialized to H'00\_00.





**Figure 17.3 Operation of Changing the CKS1 bit when EXCKSN = 0**



**Figure 17.4 Conflict between the CKS1 bit being Changed from 1 to 0 and the 32KOVI**

## 17.4 Interrupt Source

At the interrupt interval, the 32K timer generates a 32K timer-overflow interrupt (32KOV). The interrupt signal lasts for three cycles of the subclock. Since 32KOVI is internally connected to IRQ15, the CKS1 bit is set to 1 when an interrupt is generated. For the IRQ15 setting, select an interrupt receiver that is generated at the falling edge with ISCR. For details, see section 7, Interrupt Controller.

The 32K timer operates in both software-standby and deep-software standby modes. Through the generation of the 32K timer interrupt, the timer can provide the trigger for release from software-standby and deep-software-standby modes. For details, see section 28, Power-Down Mode.

**Table 17.2 TM32K Interrupt Source**

Name	Interrupt Source	Interrupt Flag	DTC Activation	Software Standby	Deep S
				Mode	Standb
32KOVI	TCNT32K interrupt	IRQ15F	Impossible	Possible	Impossible
		DT32KIF	Impossible	Impossible	Possible

## 17.5.2 Note on Register Initialization

TCR32K, TCNT32K1, TCNT32K2, and TCNT32K3 of the 32K timer are initialized in standby mode or in the pin reset state. A reset from the watchdog timer or deep-software reset does not initialize these registers.

## 17.5.3 Usage Notes on 32K Timer

- The 32K timer does not operate when the OSC32STP bit is set to 1. Always set the OSC32STP bit to 0 when starting the 32K timer.
- When the OSC32STP bit has been changed from 1 to 0, allow enough time to ensure of the oscillation by the subclock oscillator.
- Before stopping the TM32K, clear the TME bit to 0 for one clock (30  $\mu$ s) or longer to subclock.
- When switching between subclock and main clock operation, wait for 500  $\mu$ s or more (the timer counter is updated next time) before reading the timer counter.



## 18.1 Features

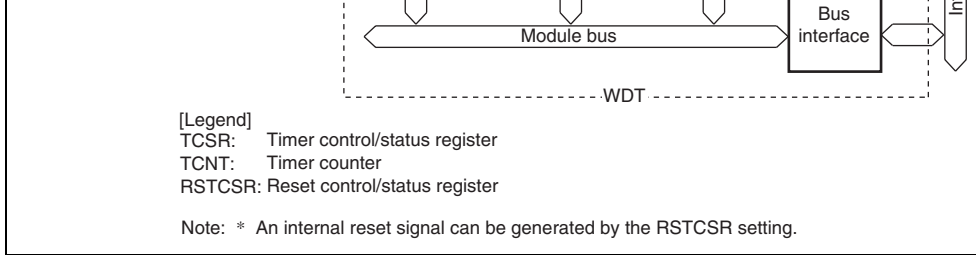
- Selectable from eight counter input clocks
- Switchable between watchdog timer mode and interval timer mode

- 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 18.1 Block Diagram of WDT**

## 18.2 Input/Output Pin

Table 18.1 shows the WDT pin configuration.

**Table 18.1 Pin Configuration**

Name	Symbol	I/O	Function
Watchdog timer overflow*	$\overline{\text{WDTOVF}}$	Output	Outputs a counter overflow signal in watchdog timer mode

Note: \* In boundary scan valid mode, counter overflow signal output cannot be used.

### 18.3.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 when the TMSCSR is cleared to 0.

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

### 18.3.2 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
Bit Name	OVF	WT/ $\bar{T}$	TME	—	—	CKS2	CKS1
Initial Value	0	0	0	1	1	0	0
R/W	R/(W)*	R/W	R/W	R	R	R/W	R/W

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

Cleared by reading TCSR when OVF = 1, then writing 0 to OVF

(When the CPU is used to clear this flag while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

6	WT/IT	0	R/W	<p>Timer Mode Select</p> <p>Selects whether the WDT is used as a watchdog interval timer.</p> <p>0: Interval timer mode When TCNT overflows, an interval timer interrupt (WOVI) is requested.</p> <p>1: Watchdog timer mode When TCNT overflows, the <math>\overline{\text{WDTOVF}}</math> signal is asserted.</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 clock 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.



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. 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 LSI is not reset, TCNT and TCSR in WDT are reset.)</p> <p>1: LSI is reset if TCNT overflows</p>
5	—	0	R/W	<p>Reserved</p> <p>Although this bit is readable/writable, reading from or writing to this bit does not affect operation.</p>
4 to 0	—	All 1	R	<p>Reserved</p> <p>These are read-only bits and cannot be modified.</p>

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

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  $\overline{\text{WDTOVF}}$  signal. If a reset caused by a signal to the  $\overline{\text{RES}}$  pin occurs at the same time as a reset caused by a WDT overflow, the  $\overline{\text{RES}}$  pin has priority and the WOVF bit in RSTCSR is cleared to 0.

The  $\overline{\text{WDTOVF}}$  signal is output for 133 cycles of  $P\phi$  when  $\text{RSTE} = 1$  in RSTCSR, and for 519 cycles of  $P\phi$  when  $\text{RSTE} = 0$  in RSTCSR. The internal reset signal is output for 519 cycles of  $P\phi$ .

When  $\text{RSTE} = 1$ , 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.

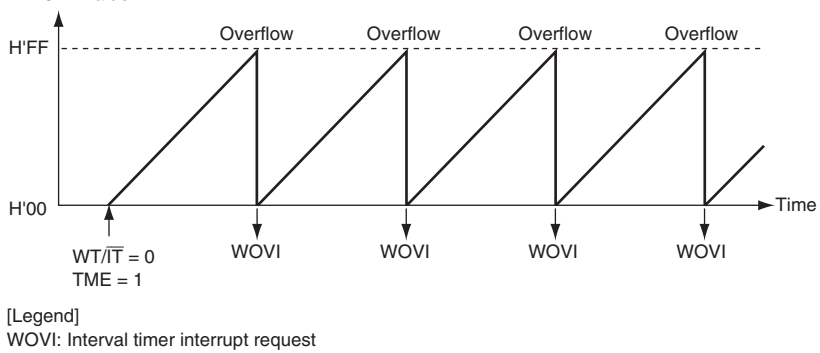
When  $\text{RSTE} = 0$ , an internal reset signal is not generated. Neither SCKCR nor the multiplication ratio of  $P\phi$  is changed.

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 and the entire LSI is reset.



- Notes: 1. If TCNT overflows when the RSTE bit is set to 1, an internal reset is generated.  
 2. 130 states when the RSTE bit is cleared to 0.

**Figure 18.2 Operation in Watchdog Timer Mode**



**Figure 18.3 Operation in Interval Timer Mode**

## 18.5 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 18.2 WDT Interrupt Source**

Name	Interrupt Source	Interrupt Flag	DTC Activation
WOVI	TCNT overflow	OVF	Impossible

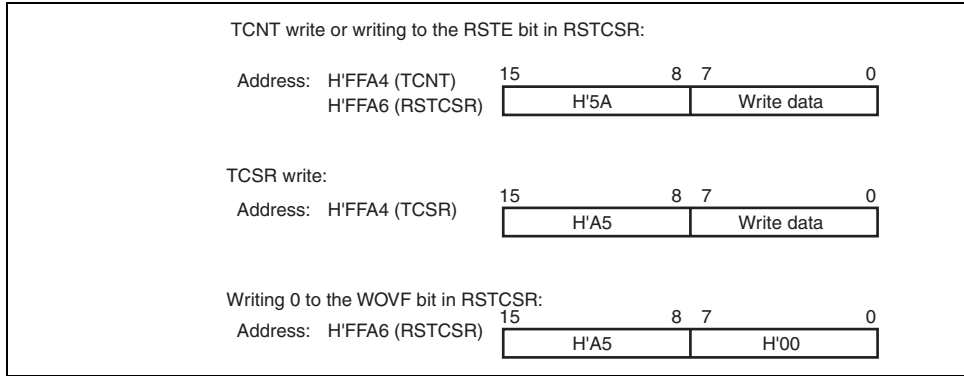
TCNT and TCSR must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

For writing, TCNT and TCSR are assigned to the same address. Accordingly, perform data transfer as shown in figure 18.4. The transfer instruction writes the lower byte data to TCNT and TCSR.

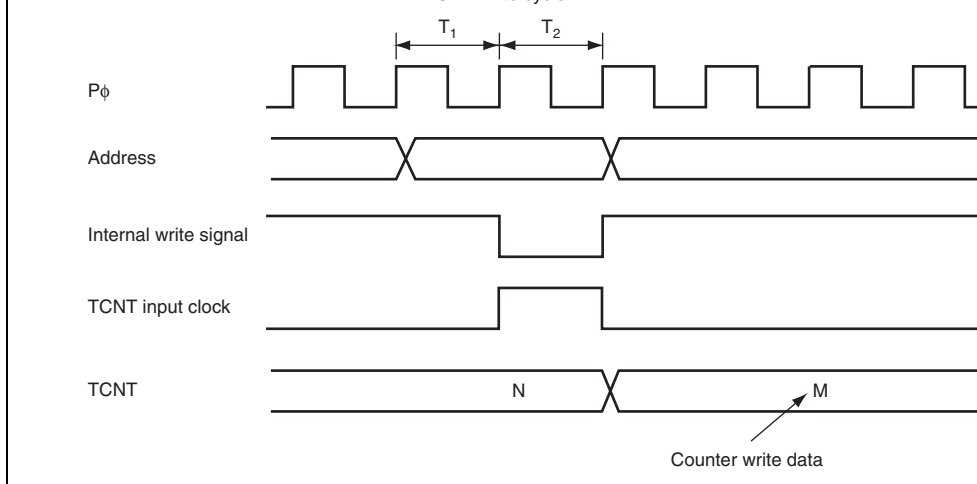
To write to RSTCSR, execute a word transfer instruction for address H'FFA6. A byte transfer instruction cannot be used to write to RSTCSR.

The method of writing 0 to the WOVF bit in RSTCSR differs from that of writing to the WOVF bit in RSTCSR. Perform data transfer as shown in figure 18.4.

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 18.4. 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 18.4 Writing to TCNT, TCSR, and RSTCSR**



**Figure 18.5 Conflict between TCNT Write and Increment**

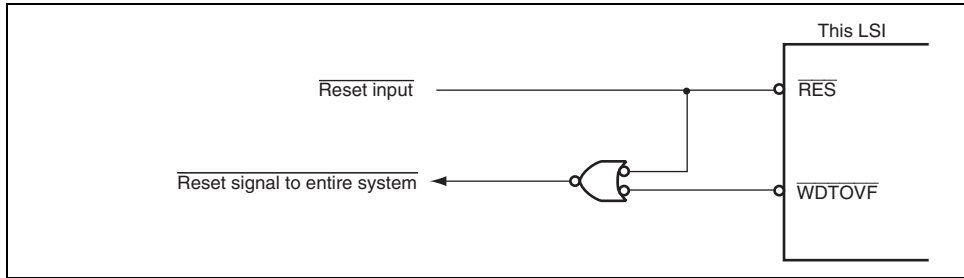
### 18.6.3 Changing Values of Bits CKS2 to CKS0

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.

### 18.6.4 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 timer 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.

If the  $\overline{\text{WDTOVF}}$  signal is input to the  $\overline{\text{RES}}$  pin, this LSI will not be initialized correctly. To ensure 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 18.6.



**Figure 18.6** Circuit for System Reset by  $\overline{\text{WDTOVF}}$  Signal (Example)

### 18.6.7 Transition to Watchdog Timer Mode or Software Standby Mode

When the WDT operates in watchdog timer mode, a transition to software standby mode is 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.





communication mode. SCI\_5 enables transmitting and receiving IrDA communication v based on the IrDA Specifications version 1.0. This LSI incorporates the on-chip CRC (C Redundancy Check) computing unit that realizes high reliability of high-speed data tran the CRC computing unit is not connected to SCI, operation is executed by writing data t registers.

Figure 19.1 shows a block diagram of the SCI\_0 to SCI\_4. Figure 19.2 shows a block di the SCI\_5 and SCI\_6.

## 19.1 Features

- Choice of asynchronous or clocked synchronous serial communication mode
- Full-duplex communication capability  
The transmitter and receiver are mutually independent, enabling transmission and re be executed simultaneously. Double-buffering is used in both the transmitter and the 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 ca interface).
- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode)
- Four interrupt sources  
The interrupt sources are transmit-end, transmit-data-empty, receive-data-full, and re error. The transmit-data-empty and receive-data-full interrupt sources can activate th DMAC.
- Module stop state specifiable

16-MHz operation: 115.192 kbps, 460.784 kbps, or 720 kbps can be selected

32-MHz operation: 720 kbps

- Average transfer rate generator (SCI\_5, SCI\_6)

8-MHz operation: 460.784 kbps can be selected

10.667-MHz operation: 115.152 kbps or 460.606 kbps can be selected

12-MHz operation: 230.263 kbps or 460.526 kbps can be selected

16-MHz operation: 115.196 kbps, 460.784 kbps, 720 kbps, or 921.569 kbps can be selected

24-MHz operation: 115.132 kbps, 460.526 kbps, 720 kbps, or 921.053 kbps can be selected

32-MHz operation: 720 kbps can be selected

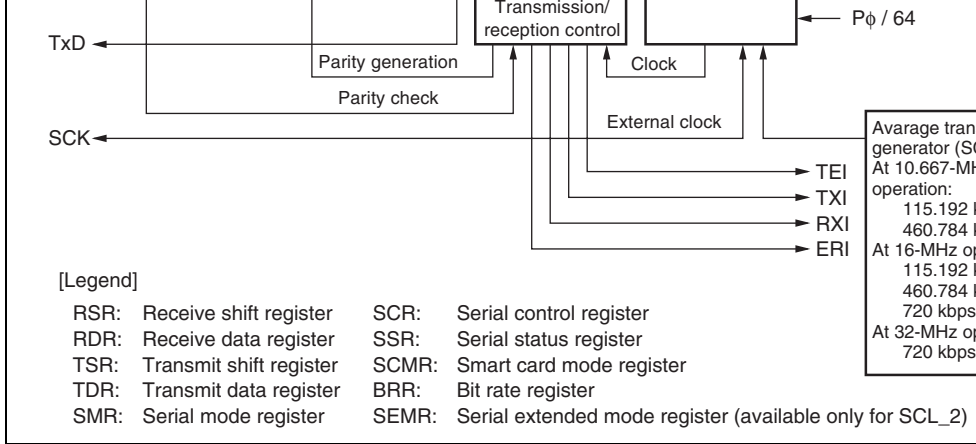
### **Clocked Synchronous Mode (SCI\_0, 1, 2, and 4):**

- Data length: 8 bits
- Receive error detection: Overrun errors

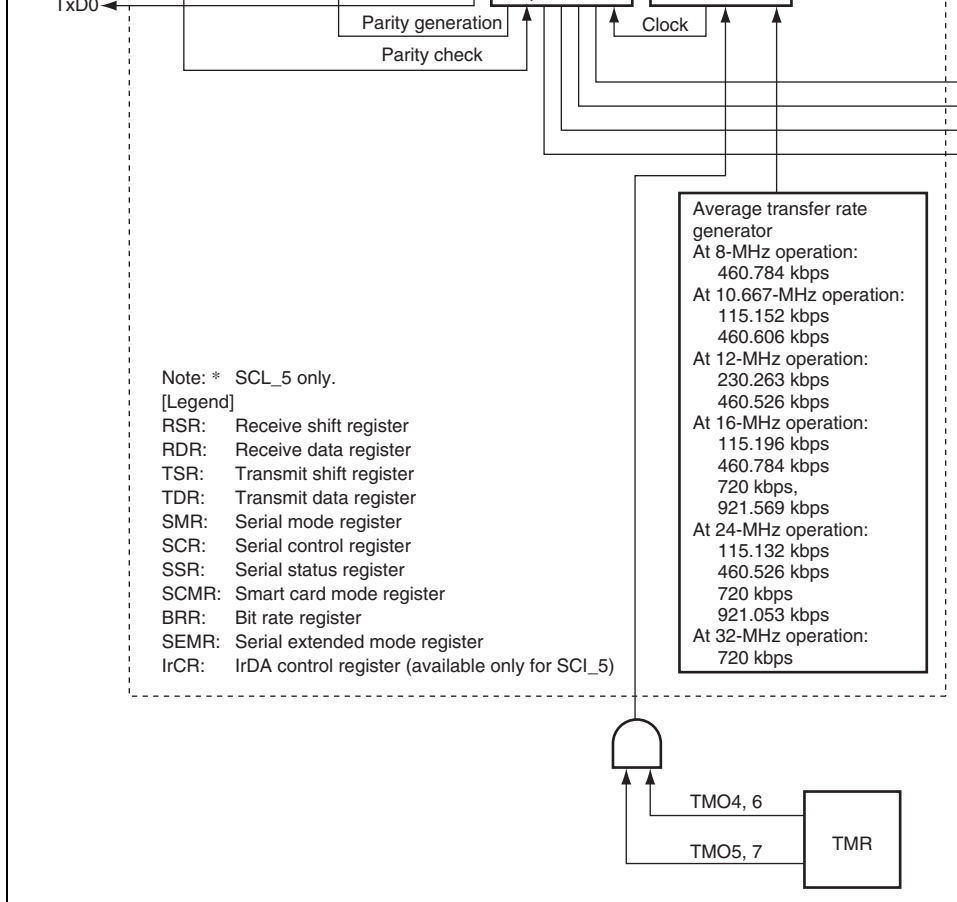
### **Smart Card Interface:**

- An error signal can be automatically transmitted on detection of a parity error during transmission
- Data can be automatically re-transmitted on receiving an error signal during transmission
- Both direct convention and inverse convention are supported

			115.192 kbps	115.192
$P\phi = 12$ MHz	—	—	460.52	230.26
$P\phi = 16$ MHz	—	720 kbps	921.56	720 kbps
		460.784 kbps	460.784	460.784
		115.192 kbps	115.192	115.192
$P\phi = 24$ MHz	—	—	921.05	720 kbps
			460.52	460.52
			115.13	115.13
$P\phi = 32$ MHz	—	720 kbps	720 kbps	720 kbps



**Figure 19.1 Block Diagram of SCI\_0, 1, 2, and 4**



**Figure 19.2 Block Diagram of SCI\_5 and SCI\_6**

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
4	SCK4	I/O	Channel 4 clock input/output
	RxD4	Input	Channel 4 receive data input
	TxD4	Output	Channel 4 transmit data output
5	RxD5/IrRxD	Input	Channel 5 receive data input
	TxD5/IrTxD	Output	Channel 5 transmit data output
6	RxD6	Input	Channel 6 receive data input
	TxD6	Output	Channel 6 transmit data output

Note: \* Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

- 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)

- Serial extended mode register\_2 (SEMR\_2)

#### **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)

#### **Channel 5:**

- Receive shift register\_5 (RSR\_5)
- Transmit shift register\_5 (TSR\_5)
- Receive data register\_5 (RDR\_5)
- Transmit data register\_5 (TDR\_5)
- Serial mode register\_5 (SMR\_5)
- Serial control register\_5 (SCR\_5)
- Serial status register\_5 (SSR\_5)
- Smart card mode register\_5 (SCMR\_5)
- Bit rate register\_5 (BRR\_5)
- Serial extended mode register\_5 (SEMR\_5)
- IrDA control register\_5 (IrCR)



- Bit rate register\_0 (BRK\_0)
- Serial extended mode register\_6 (SEMR\_6)

### 19.3.1 Receive Shift Register (RSR)

RSR is a shift register which is used to receive serial data input from the RxD pin and convert 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.

### 19.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one frame of data, it transfers the received serial data from RSR to RDR where it is stored. This allows the CPU 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, the CPU can read RDR only once. RDR cannot be written to by the CPU.

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

Bit Name							
Initial Value	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

### 19.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI automatically transfers transmit data from TDR to TSR, and then sends the data to the Tx pin. The TSR cannot be directly accessed by the CPU.

### 19.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock. 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
Bit Name	C/ $\bar{A}$	CHR	PE	O/ $\bar{E}$	STOP	MP	CKS1
Initial Value	0	0	0	0	0	0	0
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
Bit Name	GM	BLK	PE	O/ $\bar{E}$	BCP1	BCP0	CKS1
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

the MSB (bit 7) in 1DR is not transmitted in transmission.

In clocked synchronous mode, a fixed data length in bits is used.

---

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 the data before transmission, and the parity bit is checked on reception. For a multiprocessor format, parity bit generation and checking are not performed regardless of the setting.
4	O/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 to 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/E bit settings are invalid in multiprocessor mode.

---

baud rate, see section 19.3.9, Bit Rate Register (BRR).  
 is the decimal display of the value of n in BRR (see section 19.3.9, Bit Rate Register (BRR)).

Note: \* Available in SCI\_0, 1, 2, and 4 only. Setting is prohibited in SCI\_5 and SCI\_6.

### 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	<p>GSM Mode</p> <p>Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 μs from the start and the clock output control function is not appended. For details, see sections 19.7.6, Data Format (Except in Block Transfer Mode) and 19.7.8, Clock Output Control (Only SCI_0, 1, 2, and 4).</p>
6	BLK	0	R/W	<p>Setting this bit to 1 allows block transfer mode operation. For details, see section 19.7.3, Block Transfer Mode.</p>
5	PE	0	R/W	<p>Parity Enable (valid only in asynchronous mode)</p> <p>When this bit is set to 1, the parity bit is added to the data before transmission, and the parity bit is checked during reception. Set this bit to 1 in smart card interface mode.</p>
4	O/E	0	R/W	<p>Parity Mode (valid only when the PE bit is 1 in asynchronous mode)</p> <p>0: Selects even parity            1: Selects odd parity</p> <p>For details on the usage of this bit in smart card interface mode, see section 19.7.2, Data Format (Except in Block Transfer Mode).</p>

---

1	CKS1	0	R/W	Clock Select 1, 0
0	CKS0	0	R/W	These bits select the clock source for the baud 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 19.3.9, Bit Rate Register (BRR). The value of $n$ in BRR is the decimal display of the value of $n$ in BRR (see section 19.3.9, Bit Rate Register (BRR)).

---

Note:  $t_{\text{etu}}$  (Elementary Time Unit): 1-bit transfer time

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
Bit Name	TIE	RIE	TE	RE	MPIE	TEIE	CKE1
Initial Value	0	0	0	0	0	0	0
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 = 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>

When this bit is set to 1, reception is enabled. Under the normal condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous condition bit in clocked synchronous mode. Note that SMR must be set prior to setting the RE bit to 1 in order to determine the reception format.

Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected. When the bit is set to the previous value is retained.

3	MPIE	0	R/W	<p>Multiprocessor Interrupt Enable (valid only when the multiprocessor 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 not performed. On receiving data in which the multiprocessor bit is 1, the bit is automatically cleared and normal reception is resumed. For details, see section 19.5, Multiprocessor Communication Function.</p> <p>When receive data including MPB = 0 in SSR is received, transfer of the received data from RS to CPU is performed. In the case of detection of reception errors, and the settings of the FER, and ORER flags in SSR are not performed. When receive data including MPB = 1 is received, the MIE 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 the FER and ORER flags are enabled.</p>
---	------	---	-----	--

00: On-chip baud rate generator  
The SCK pin functions as I/O port.

01: On-chip baud rate generator  
The clock with the same frequency as the bit rate  
output from the SCK pin.

1X: External clock  
The clock with a frequency 16 times the bit rate  
should be input from the SCK pin.

- Clocked synchronous mode

0X: Internal clock  
The SCK pin functions as the clock output pin.

1X: External clock  
The SCK pin functions as the clock input pin.

---



1X: External clock or average transfer rate generator

When an external clock is used, the clock waveform frequency 16 times the bit rate should be input to the SCK pin.

When an average transfer rate generator is used,

- Clocked synchronous mode

0X: Internal clock

The SCK pin functions as the clock output pin.

1X: External clock

The SCK pin functions as the clock input pin.

---

1	CKE1	0	R/W	Clock Enable 1, 0 (for SCI_5 and SCI_6)
0	CKE0	0	R/W	These bits select the clock source.

- Asynchronous mode

00: On-chip baud rate generator

1X: TMR clock input or average transfer rate generator

When an average transfer rate generator is used,

When TMR clock input is used.

- Clocked synchronous mode

Not available

---

[Legend]

X: Don't care

When this bit is set to 1, RXI and ERI interrupt requests are enabled.

RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag, then clearing the flag to 0, or by clearing the RIE bit.

5	TE	0	R/W	Transmit Enable  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 to 0. Note that SMR should be set prior to setting this bit to 1 in order to designate the transmission format.  If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.
4	RE	0	R/W	Receive Enable  When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting a start bit in asynchronous mode or the synchronous clock 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.  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.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the bit in SMR is 1 in asynchronous mode)  Write 0 to this bit in smart card interface mode.
2	TEIE	0	R/W	Transmit End Interrupt Enable  Write 0 to this bit in smart card interface mode.

1X: Reserved

- When GM in SMR = 1  
00: Output fixed low  
01: Clock output  
10: Output fixed high  
11: Clock output

---

Note: \* No SCK pins exist in SCI\_5 and SCI\_6.

### 19.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. TDRF, 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
Bit Name	TDRE	RDRF	ORER	FER	PER	TEND	MPB
Initial Value	1	0	0	0	0	1	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R

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

Bit	Bit Name	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 TDR (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, 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>

while the corresponding interrupt is enabled, to read the flag after writing 0 to it.)

- When an RXI interrupt request is issued all DMAC or DTC to read data from RDR

The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.

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.

---

5	ORER	0	R/(W)*	Overrun Error
---	------	---	--------	---------------

Indicates that an overrun error has occurred during reception and the reception ends abnormally.

[Setting condition]

- When the next serial reception is completed RDRF = 1

In RDR, receive data prior to an overrun error occurrence is retained, but data received after overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous serial transmission also cannot continue.

[Clearing condition]

- When 0 is written to ORER after reading ORER = 1 (When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, to read the flag after writing 0 to it.)

Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.

---

is transferred to RDR, however, the RDRF flag is set. In addition, when the FER flag is being set, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.

[Clearing condition]

- When 0 is written to FER after reading FER = 1.  
(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, the CPU must read the flag after writing 0 to it.)  
Even when the RE bit in SCR is cleared, the FER flag is not affected and retains its previous value.
-

subsequent serial reception cannot be performed in clocked synchronous mode, serial transmission cannot continue.

[Clearing condition]

- When 0 is written to PER after reading PER (When the CPU is used to clear this flag by while the corresponding interrupt is enabled to read the flag after writing 0 to it.)  
Even when the RE bit in SCR is cleared, the is not affected and retains its previous value.

2	TEND	1	R	Transmit End [Setting conditions] <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 transmit character</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE</li> <li>• When a TXI interrupt request is issued allowing DMAC or DTC to write data to TDR</li> </ul>
1	MPB	0	R	Multiprocessor Bit Stores the multiprocessor bit value in the receiver. When the RE bit in SCR is cleared to 0 its previous value is retained.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Sets the multiprocessor bit value to be added to the transmit frame.

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

- When 0 is written to TDRE after reading TDR (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, to read the flag after writing 0 to it.)
- When a TXI interrupt request is issued allowing DMAC or DTC to write data to TDR

---

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 received 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 (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, 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>
---	------	---	--------	--

---



is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous serial transmission also cannot continue.

[Clearing condition]

- When 0 is written to OREER after reading OREER (When the CPU is used to clear this flag by software while the corresponding interrupt is enabled, the CPU must read the flag after writing 0 to it.)

Even when the RE bit in SCR is cleared, the OREER flag is not affected and retains its previous value.

---

4	ERS	0	R/(W)*	Error Signal Status
---	-----	---	--------	---------------------

[Setting condition]

- When a low error signal is sampled

[Clearing condition]

- When 0 is written to ERS after reading ERS

---

subsequent serial reception cannot be performed. In  
clocked synchronous mode, serial transmission  
cannot continue.

[Clearing condition]

- When 0 is written to PER after reading PER = 1.  
(When the CPU is used to clear this flag by writing 0 to it,  
while the corresponding interrupt is enabled, the CPU  
must read the flag after writing 0 to it.)

Even when the RE bit in SCR is cleared, the RE bit  
is not affected and retains its previous value.

---

Set timing depends on the register setting.

When GM = 0 and BLK = 0, 2.5 etu after tra start

When GM = 0 and BLK = 1, 1.5 etu after tra start

When GM = 1 and BLK = 0, 1.0 etu after tra start

When GM = 1 and BLK = 1, 1.0 etu after tra start

[Clearing conditions]

- When 0 is written to TEND after reading TE
- When a TXI interrupt request is issued allow DMAC or DTC to write the next data to TDF

1	MPB	0	R	Multiprocessor Bit Not used in smart card interface mode.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Write 0 to this bit in smart card interface mode.

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

Bit	Bit Name	Value	R/W	Description
7 to 4	—	All 1	—	Reserved These bits are always read as 1.
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 L
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	—	Reserved This bit is always read as 1.
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

Asynchronous mode	0	$N = \frac{P\phi \times 10^6}{64 \times 2^{2n-1} \times B} - 1$	$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} \right\}$
	1	$N = \frac{P\phi \times 10^6}{32 \times 2^{2n-1} \times B} - 1$	$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{B \times 32 \times 2^{2n-1} \times (N+1)} \right\}$
Clocked synchronous mode		$N = \frac{P\phi \times 10^6}{8 \times 2^{2n-1} \times B} - 1$	
Smart card interface mode		$N = \frac{P\phi \times 10^6}{S \times 2^{2n+1} \times B} - 1$	$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N+1)} \right\}$

[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	3
0	1	1	0	1	6
1	0	2	1	0	3
1	1	3	1	1	2

**Table 19.4 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode)**

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
110	2	141	0.03	2	174	-0.26	2	177	-0.25	2	212
150	2	103	0.16	2	127	0.00	2	129	0.16	2	155
300	1	207	0.16	1	255	0.00	2	64	0.16	2	77
600	1	103	0.16	1	127	0.00	1	129	0.16	1	155
1200	0	207	0.16	0	255	0.00	1	64	0.16	1	77
2400	0	103	0.16	0	127	0.00	0	129	0.16	0	155
4800	0	51	0.16	0	63	0.00	0	64	0.16	0	77
9600	0	25	0.16	0	31	0.00	0	32	-1.36	0	38
19200	0	12	0.16	0	15	0.00	0	15	1.73	0	19
31250	0	7	0.00	0	9	-1.70	0	9	0.00	0	11
38400	—	—	—	0	7	0.00	0	7	1.73	0	9

4800	0	79	0.00	0	90	0.16	0	95	0.00	0	103
9600	0	39	0.00	0	45	-0.93	0	47	0.00	0	51
19200	0	19	0.00	0	22	-0.93	0	23	0.00	0	25
31250	0	11	2.40	0	13	0.00	0	14	-1.70	0	15
38400	0	9	0.00	—	—	—	0	11	0.00	0	12

Note: In SCI\_2, 5, and 6, this is an example when the ABCS bit in SEMR\_2, 5, and 6 is set to 1.  
When the ABCS bit is set to 1, the bit rate is two times.

**Table 19.4 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode)**

Bit Rate (bit/s)	Operating Frequency P <sub>φ</sub> (MHz)											
	17.2032			18			19.6608			20		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	
110	3	75	0.48	3	79	-0.12	3	86	0.31	3	88	
150	2	223	0.00	2	233	0.16	2	255	0.00	3	64	
300	2	111	0.00	2	116	0.16	2	127	0.00	2	129	
600	1	223	0.00	1	233	0.16	1	255	0.00	2	64	
1200	1	111	0.00	1	116	0.16	1	127	0.00	1	129	
2400	0	223	0.00	0	233	0.16	0	255	0.00	1	64	
4800	0	111	0.00	0	116	0.16	0	127	0.00	0	129	
9600	0	55	0.00	0	58	-0.69	0	63	0.00	0	64	
19200	0	27	0.00	0	28	1.02	0	31	0.00	0	32	
31250	0	16	1.20	0	17	0.00	0	19	-1.70	0	19	
38400	0	13	0.00	0	14	-2.34	0	15	0.00	0	15	

2400	1	80	0.47	1	97	-0.35	1	100	0.39	1	113
4800	0	162	-0.15	0	194	0.16	0	214	-0.07	0	227
9600	0	80	0.47	0	97	-0.35	0	106	0.39	0	113
19200	0	40	-0.76	0	48	-0.35	0	53	-0.54	0	56
31250	0	24	0.00	0	29	0	0	32	0	0	34
38400	0	19	1.73	0	23	1.73	0	26	-0.54	0	27

Note: In SCI\_2, 5, and 6, this is an example when the ABCS bit in SEMR\_2, 5, and 6 is 0.  
When the ABCS bit is set to 1, the bit rate is two times.

**Table 19.5 Maximum Bit Rate for Each Operating Frequency (Asynchronous Mode)**

$P\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N	$P\phi$ (MHz)	Maximum Bit Rate (bit/s)	n
8	250000	0	0	17.2032	537600	0
9.8304	307200	0	0	18	562500	0
10	312500	0	0	19.6608	614400	0
12	375000	0	0	20	625000	0
12.288	384000	0	0	25	781250	0
14	437500	0	0	30	937500	0
14.7456	460800	0	0	33	1031250	0
16	500000	0	0	35	1093750	0



14.7456	3.6864	230400	33	8.2500	515
16	4.0000	250000	35	8.7500	546

Note: In SCI\_2, this is an example when the ABCS bit in SEMR\_2 is 0.  
When the ABCS bit is set to 1, the bit rate is two times.

5k	1	99	1	124	1	199	1	249	2	77	2	93	2	102	1
10k	0	199	0	249	1	99	1	124	1	155	1	187	1	205	1
25k	0	79	0	99	0	159	0	199	0	249	1	74	1	82	1
50k	0	39	0	49	0	79	0	99	0	124	0	149	0	164	0
100k	0	19	0	24	0	39	0	49	0	62	0	74	0	82	0
250k	0	7	0	9	0	15	0	19	0	24	0	29	0	32	0
500k	0	3	0	4	0	7	0	9	—	—	0	14	—	—	—
1M	0	1		0	3	0	4	—	—	—	—	—	—	—	—
2.5M			0	0* <sup>1</sup>		0	1	—	—	0	2	—	—	—	—
5M						0	0* <sup>1</sup>	—	—	—	—	—	—	—	—

[Legend]

Space: Setting prohibited.

—: Can be set, but there will be error.

Notes: 1. Continuous transmission or reception is not possible.

2. No clocked synchronous mode exists in SCI\_5 and SCI\_6.

**Table 19.8 Maximum Bit Rate with External Clock Input (Clocked Synchronous M**

$P\phi$ (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			

Note \* No clocked synchronous mode exists in SCI\_5 and SCI\_6.

(bit/sec)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N
9600	0	1	0.00	0	1	12.01	0	2	15.99	0	2

Bit Rate (bit/sec)	Operating Frequency P $\phi$ (MHz)											
	25.00			30.00			33.00			35.00		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	
9600	0	3	12.49	0	3	5.01	0	4	7.59	0	4	

**Table 19.10 Maximum Bit Rate for Each Operating Frequency (Smart Card Inter Mode, S = 372)**

P $\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N	P $\phi$ (MHz)	Maximum Bit Rate (bit/s)	n
7.1424	9600	0	0	18.00	24194	0
10.00	13441	0	0	20.00	26882	0
10.7136	14400	0	0	25.00	33602	0
13.00	17473	0	0	30.00	40323	0
14.2848	19200	0	0	33.00	44355	0
16.00	21505	0	0	35.00	47043	0

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	Undefined	R	Reserved These bits are always read as undefined and cannot be modified.
3	ABCS	0	R/W	Asynchronous Mode Base clock Select (valid only in asynchronous mode) Selects the base clock for a 1-bit period. 0: The base clock has a frequency 16 times the rate 1: The base clock has a frequency 8 times the rate

base clock with a frequency 16 times the transfer rate)

010: 460.784 kbps of average transfer rate specified  
 $P\phi = 10.667$  MHz is selected (operated using the base clock with a frequency 8 times the transfer rate)

011: 720 kbps of average transfer rate specified  
32 MHz is selected (operated using the base clock with a frequency 16 times the transfer rate)

100: Setting prohibited

101: 115.192 kbps of average transfer rate specified  
 $P\phi = 16$  MHz is selected (operated using the base clock with a frequency 16 times the transfer rate)

110: 460.784 kbps of average transfer rate specified  
 $P\phi = 16$  MHz is selected (operated using the base clock with a frequency 16 times the transfer rate)

111: 720 kbps of average transfer rate specified  
16 MHz is selected (operated using the base clock with a frequency 8 times the transfer rate)

The average transfer rate only supports operation at clock frequencies of 10.667 MHz, 16 MHz, and 32 MHz.

---

Bit Name	—	—	—	ABCS	ACS3	ACS2	ACS1
Initial Value	Undefined	Undefined	Undefined	0	0	0	0
R/W	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	Undefined	R	Reserved These bits are always read as undefined and cannot be modified.
4	ABCS	0	R/W	Asynchronous Mode Base Clock Select (valid only in asynchronous mode) Selects the base clock for a 1-bit period. 0: The base clock has a frequency 16 times the average transfer rate 1: The base clock has a frequency 8 times the average transfer rate
3	ACS3	0	R/W	Asynchronous Mode Clock Source Select
2	ACS2	0	R/W	These bits select the clock source for the average transfer rate function in the asynchronous mode. If the average transfer rate function is enabled, the clock is automatically specified regardless of the bit value. The average transfer rate only corresponds to 8MHz, 10.667MHz, 12MHz, 16MHz, 24MHz, and 32MHz. No other clock is available. Setting of ACS0 must be done in the asynchronous mode (the C/A bit in SMR = 0) and the external clock input must be selected (the CKE bit I SCR = 1). The setting examples are shown in figures 19.3 and 19.4.  (Each number in the four-digit number below corresponds to the value in the bits ACS3 to ACS0 left to right respectively.)
1	ACS1	0	R/W	
0	ACS0	0	R/W	

average transfer rate specific to  $P\phi = 8M$   
selected (operated using the base clock  
frequency 8 times the transfer rate)

0100: TMR clock input  
This setting allows the TMR compare ma  
to be used as the base clock. The table  
shows the correspondence between the  
channels and the compare match output

SCI Channel	TMR Unit	Compare M Output
SCI_5	Unit 2	TMO4, TMC
SCI_6	Unit 3	TMO6, TMC

0101: 115.196 kbps of average transfer rate sp  
 $P\phi = 16$  MHz is selected (operated using  
clock with a frequency 16 times the trans

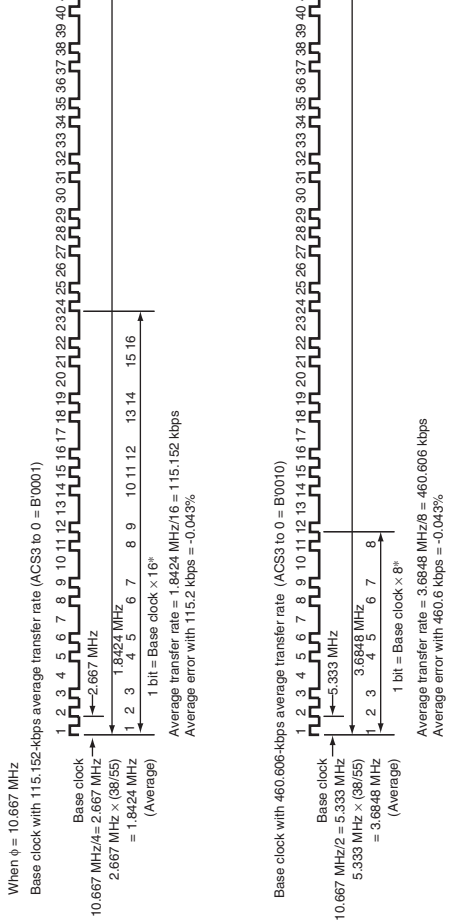
0110: 460.784 kbps of average transfer rate sp  
 $P\phi = 16$  MHz is selected (operated using  
clock with a frequency 16 times the trans

0111: 720 kbps of average transfer rate specifi  
16 MHz is selected (operated using the b  
with a frequency 8 times the transfer rate

---

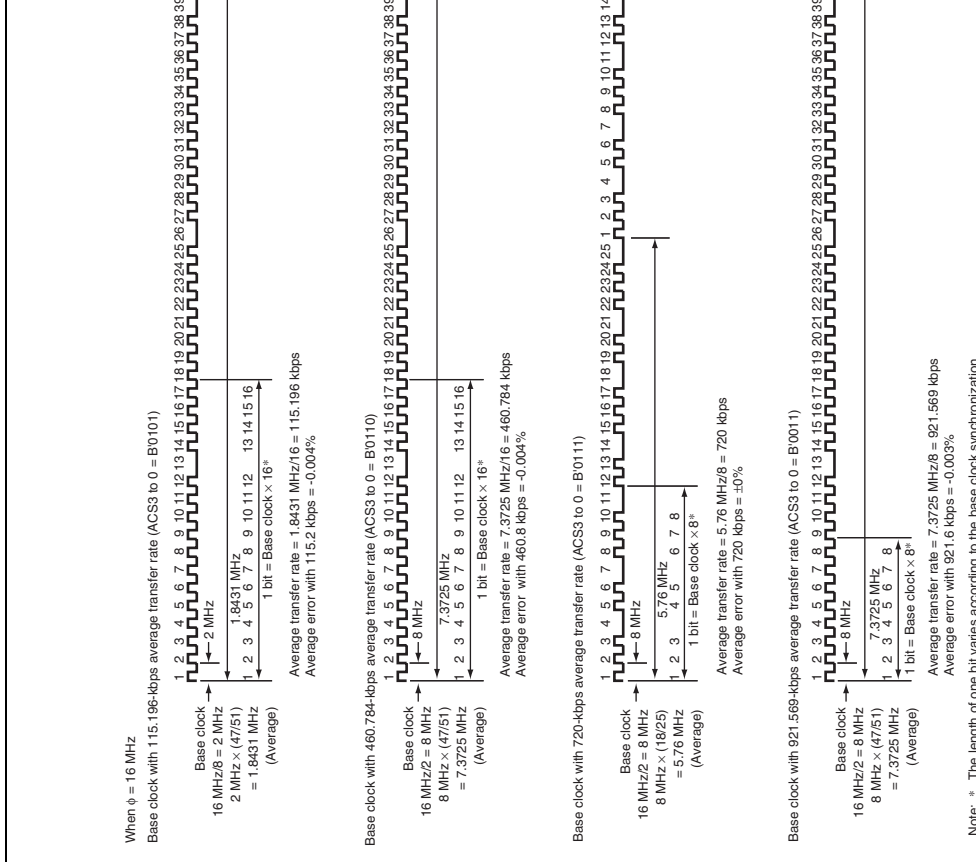
- 1011: 921.053 kbps of average transfer rate specific to  $P\phi = 24$  MHz is selected or 460.526 kbps of average transfer rate specific to  $P\phi = 12$  MHz is selected (operated using the base clock with a frequency 8 times the transfer rate)
  - 1100: 720 kbps of average transfer rate specific to  $P\phi = 32$  MHz is selected (operated using the base clock with a frequency 16 times the transfer rate)
  - 1101: Reserved (setting prohibited)
  - 111x: Reserved (setting prohibited)
-



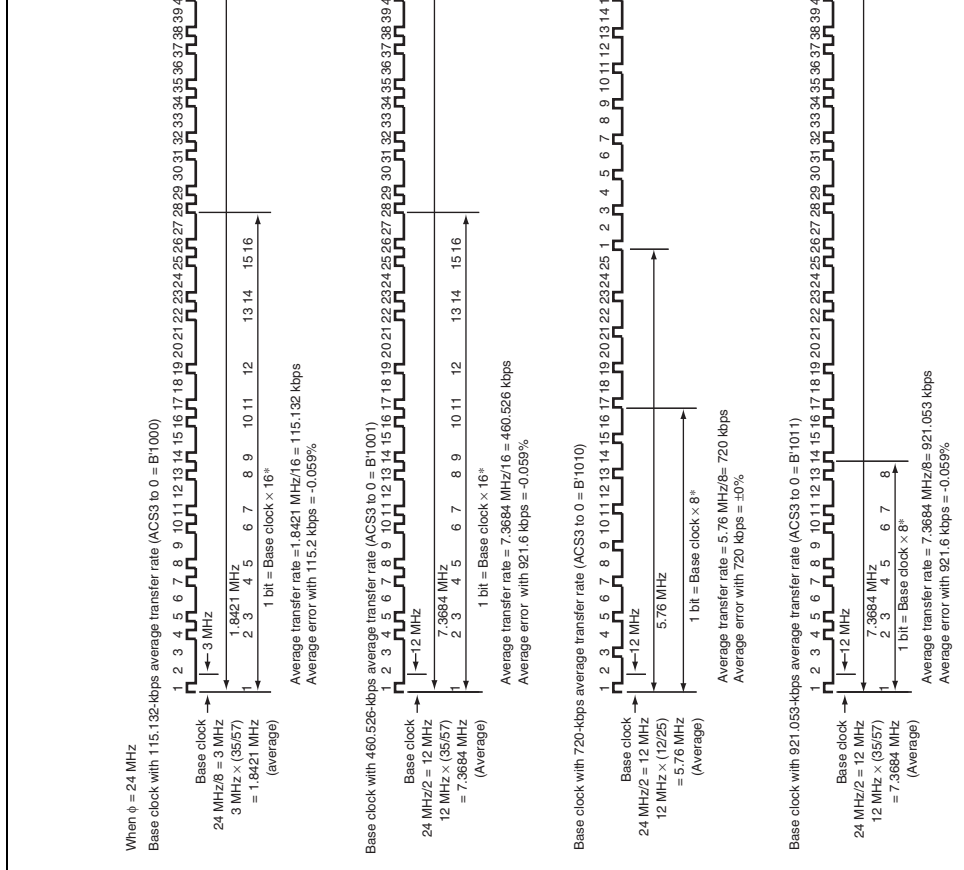


Note: \* The length of one bit varies according to the base clock synchronization.

Figure 19.3 Examples of Base Clock when Average Transfer Rate Is Selected



**Figure 19.3 Examples of Base Clock when Average Transfer Rate Is Selected**



**Figure 19.3 Examples of Base Clock when Average Transfer Rate Is Selected**

Example when TMR clock input is used in SCI\_5  
 187.5-kbps average transfer rate is generated by TMR when  $\phi = 32$  MHz  
 (1) TMO4 is set as a base clock and generates 4 MHz.  
 (2) TMO5 is set as TCNT\_4 compare match count and generates a clock enable multiplied by 3/4.  
 The average transfer rate will be  $3 \text{ MHz}/16 = 187.5 \text{ kbps}$ .

TMR and SCI Settings:

- TCR\_4 = H'09 (TCNT4 cleared by TCORA\_4 compare match, TCNT4 incremented at rising edge of P0(2))
  - TCCR\_4 = H'01
  - TCR\_5 = H'0C (TCNT5 cleared by TCORA\_5 compare match, TCNT5 incremented by TCNT\_4 compare match A)
  - TCCR\_5 = H'00
  - TCSR\_4 = H'09 (0 output on TCORA\_4 compare match, 1 output on TCORB\_4 compare match)
  - TCSR\_5 = H'09 (0 output on TCORA\_5 compare match, 1 output on TCORB\_5 compare match)
  - TCNT\_4 = TCNT\_5 = 0
  - TCORA\_4 = H'03, TCORB\_4 = H'01
  - TCORA\_5 = H'03, TCORB\_5 = H'00
  - SEMR\_5 = H'04
- When SCI\_6 is used, set TMO6 as a base clock and TMO7 as a clock enable.

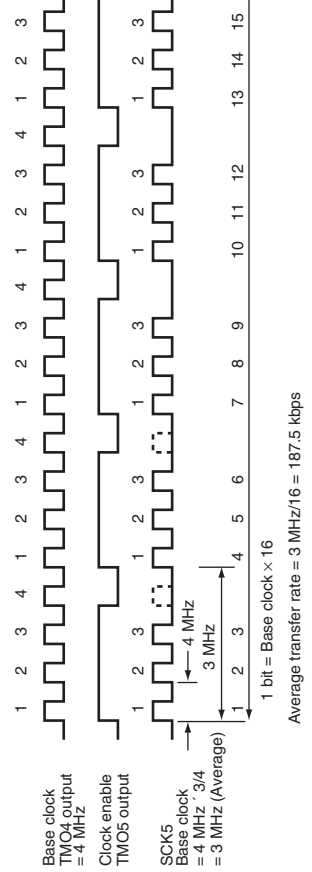


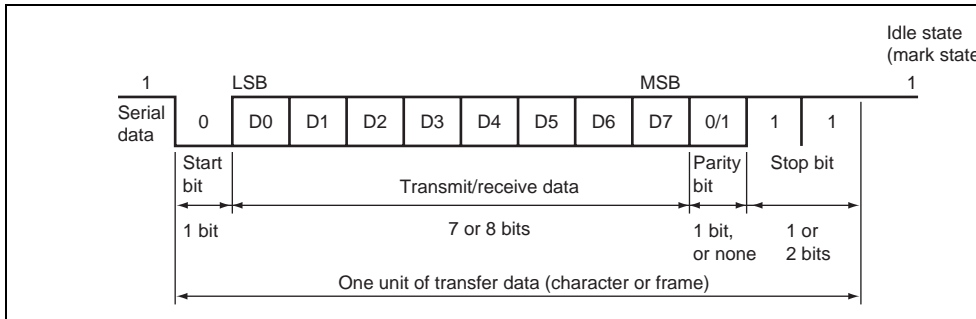
Figure 19.4 Example of Average Transfer Rate Setting when TMR Clock Is I

Bit	Bit Name	Value	R/W	Description
7	IrE	0	R/W	IrDA Enable* Sets the SCI_5 I/O to normal SCI or IrDA. 0: TxD5/IrTxD and RxD5/IrRxD pins operate as TxD5 and RxD5. 1: TxD5/IrTxD and RxD5/IrRxD pins are operated as IrTxD and IrRxD.
6	IrCK2	0	R/W	IrDA Clock Select 2 to 0
5	IrCK1	0	R/W	Sets the pulse width of high state at encoding
4	IrCK0	0	R/W	output pulse when the IrDA function is enabled 000: Pulse-width = $B \times 3/16$ (Bit rate $\times 3/16$ ) 001: Pulse-width = $P\phi/2$ 010: Pulse-width = $P\phi/4$ 011: Pulse-width = $P\phi/8$ 100: Pulse-width = $P\phi/16$ 101: Pulse-width = $P\phi/32$ 110: Pulse-width = $P\phi/64$ 111: Pulse-width = $P\phi/128$
3	IrTxINV	0	R/W	IrTx Data Invert This bit specifies the inversion of the logic level of the IrTx output. When inversion is done, the pulse width of the high state specified by the bits 6 to 4 becomes the pulse width in low state. 0: Outputs the transmission data as it is as IrTx output 1: Outputs the inverted transmission data as IrTx output

Note: \* The IrDA function should be used when the ABCS bit in SEMR\_5 is set to 0 and ACS3 to ACS0 bits in SEMR\_5 are set to B'0000.

## 19.4 Operation in Asynchronous Mode

Figure 19.5 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), transmit/receive data, a parity bit, and 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 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 19.5 Data Format in Asynchronous Communication  
(Example with 8-Bit Data, Parity, Two Stop Bits)**

0	0	0	0	S	8-bit data	STOP
0	0	0	1	S	8-bit data	STOP ST
0	1	0	0	S	8-bit data	P ST
0	1	0	1	S	8-bit data	P ST
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 ST
0	-	1	0	S	8-bit data	MPB ST
0	-	1	1	S	8-bit data	MPB ST
1	-	1	0	S	7-bit data	MPB STOP
1	-	1	1	S	7-bit data	MPB STOP ST

[Legend]

S: Start bit

STOP: Stop bit

P: Parity bit

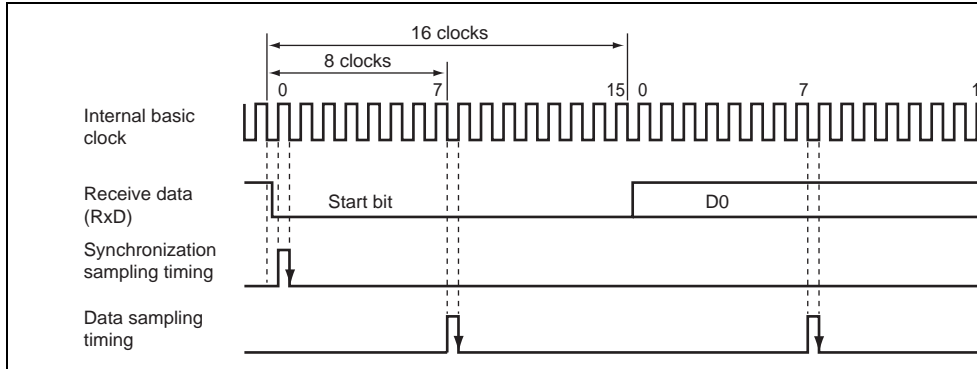
MPB: Multiprocessor bit

M: Reception margin  
 N: Ratio of bit rate to clock (When ABCS = 0, N = 16. When ABCS = 1, N = 8.)  
 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.

$$M = \left( 0.5 - \frac{1}{2 \times 16} \right) \times 100 \quad [\%] = 46.875\%$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

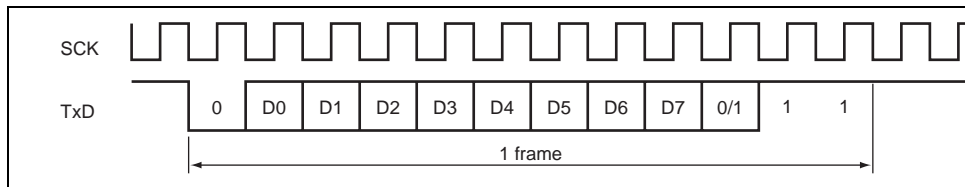


**Figure 19.6 Receive Data Sampling Timing in Asynchronous Mode**

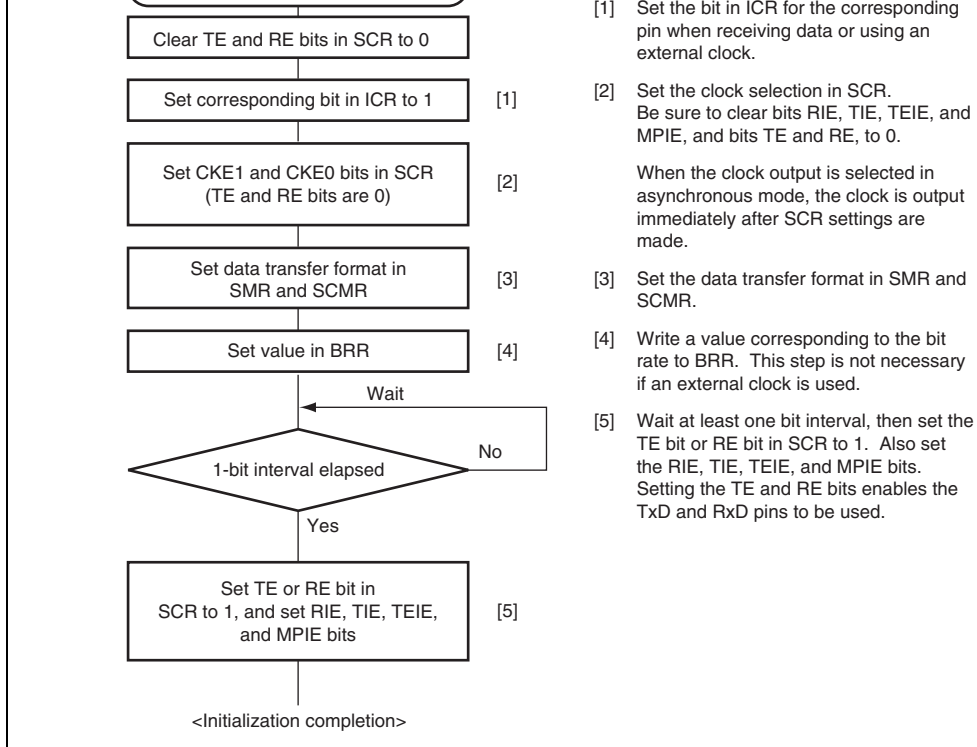
Note: \* This is an example when the ABCS bit in SEMR\_2, 5, and 6 is 0. When the ABCS bit is 1, a frequency of 8 times the bit rate is used as a base clock and receive data is sampled at the rising edge of the 4th pulse of the base clock.



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 19.7.



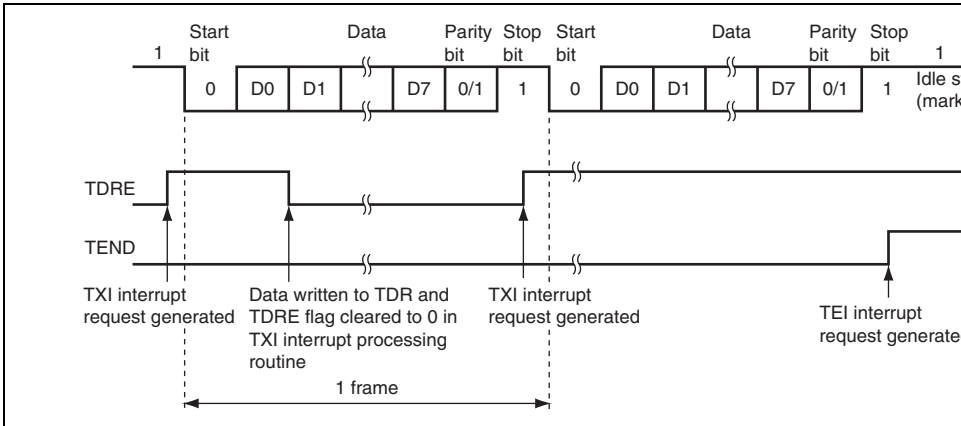
**Figure 19.7 Phase Relation between Output Clock and Transmit Data (Asynchronous Mode)**



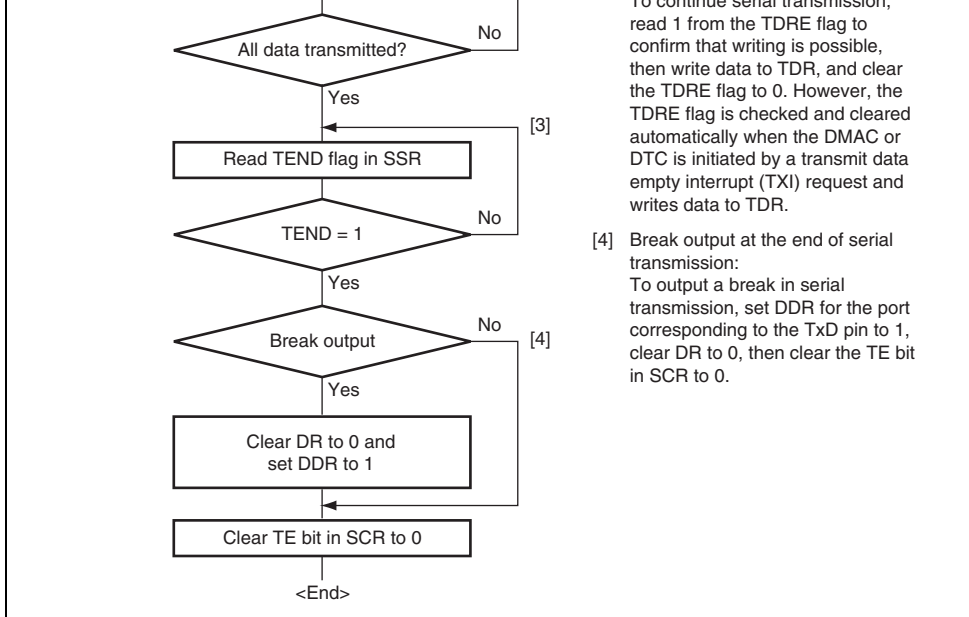
**Figure 19.8 Sample SCI Initialization Flowchart**

3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit, 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 state is entered in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TXE interrupt request is generated.

Figure 19.10 shows a sample flowchart for transmission in asynchronous mode.

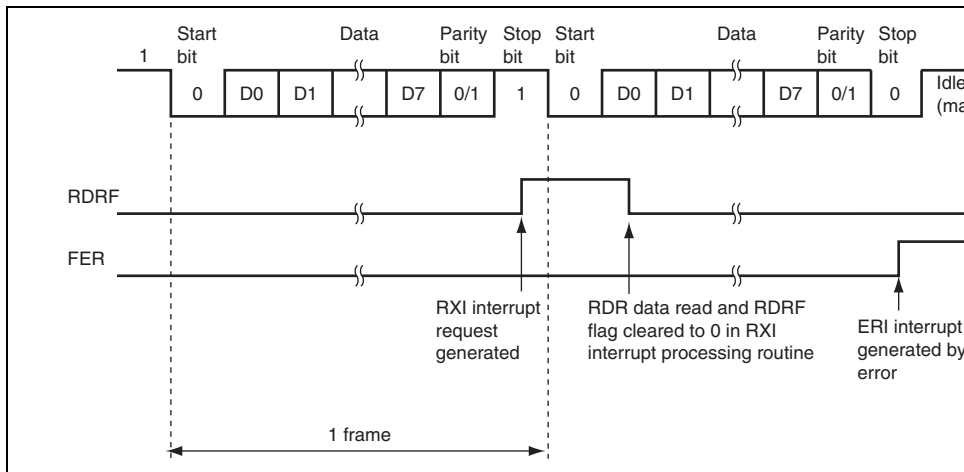


**Figure 19.9 Example of Operation for Transmission in Asynchronous Mode  
(Example with 8-Bit Data, Parity, One Stop Bit)**



**Figure 19.10 Example of Serial Transmission Flowchart**

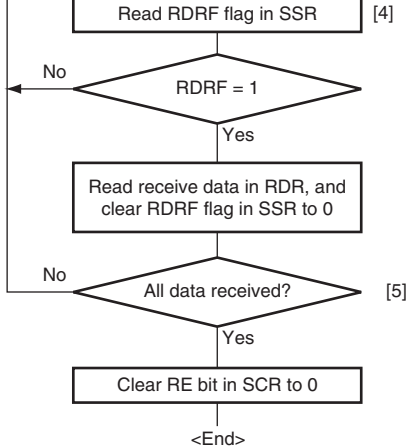
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is 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 19.11 Example of SCI Operation for Reception  
(Example with 8-Bit Data, Parity, One Stop Bit)**

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.

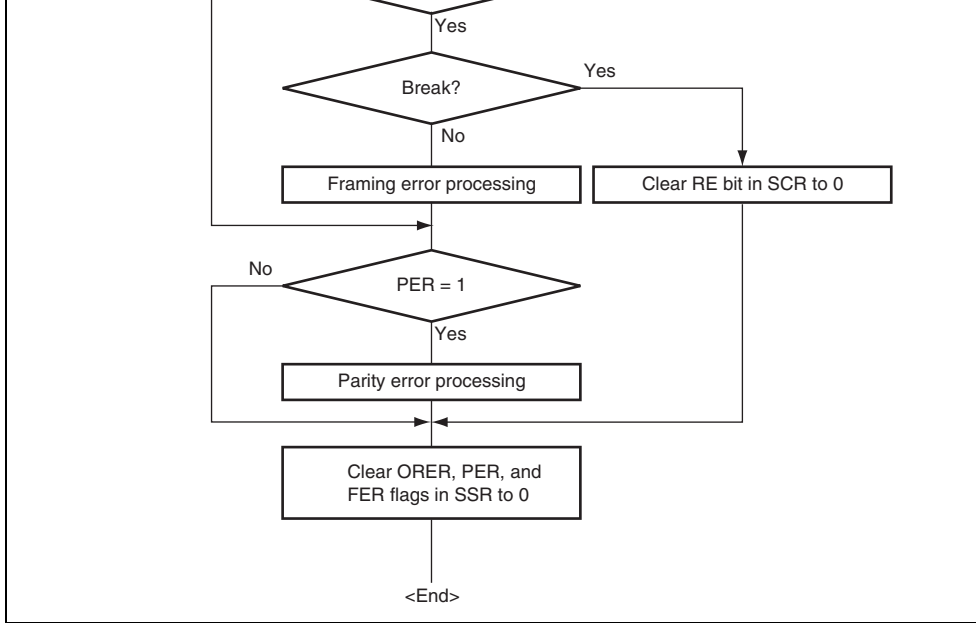


value of the input port corresponding to the RxID pin.

[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 DMAC or DTC is initiated by an RXI interrupt and reads data from RDR.

**Figure 19.12 Sample Serial Reception Flowchart (1)**



**Figure 19.12 Sample Serial Reception Flowchart (2)**



19.13 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 data added with a multiprocessor bit set to 0. The receiving station skips data until data with a 1 multiprocessor bit is sent. When a 1 multiprocessor bit is received, the receiving station compares that data with its own ID code. 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.

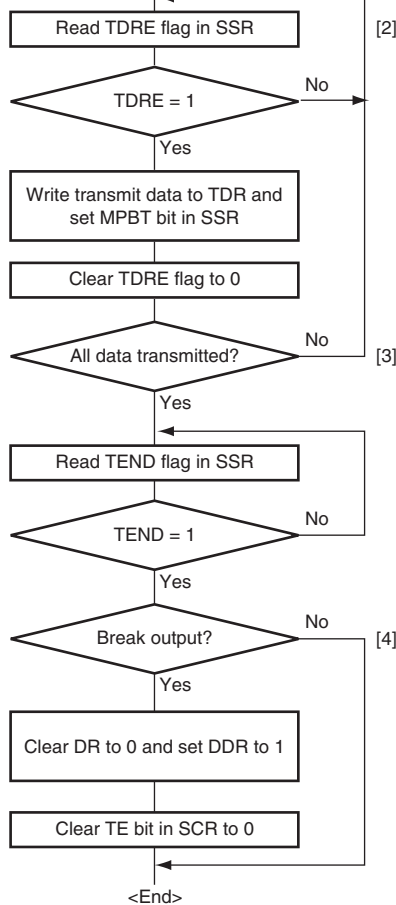
The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set, the transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status bits RDRF, FER, and ORER in SSR to 1 are prohibited until data with a 1 multiprocessor bit is received. On reception of a receive character with a 1 multiprocessor bit, the MPBR bit is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RXI 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 bits are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.

ID transmission cycle = Data transmission cycle =  
receiving station      Data transmission to  
specification          receiving station specified by ID

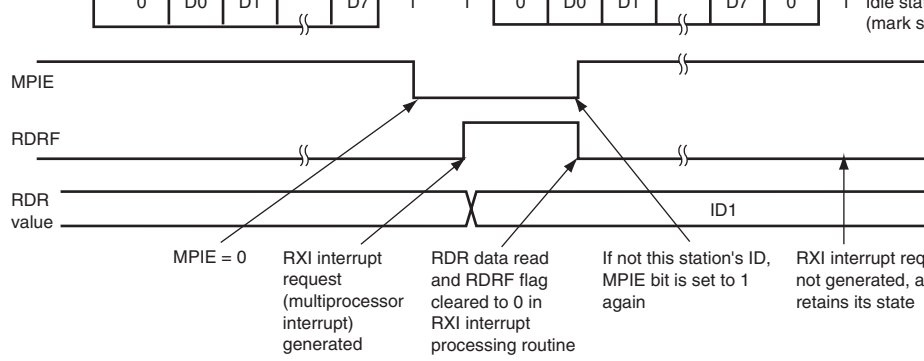
[Legend]  
MPB: Multiprocessor bit

**Figure 19.13 Example of Communication Using Multiprocessor Format  
(Transmission of Data H'AA to Receiving Station A)**

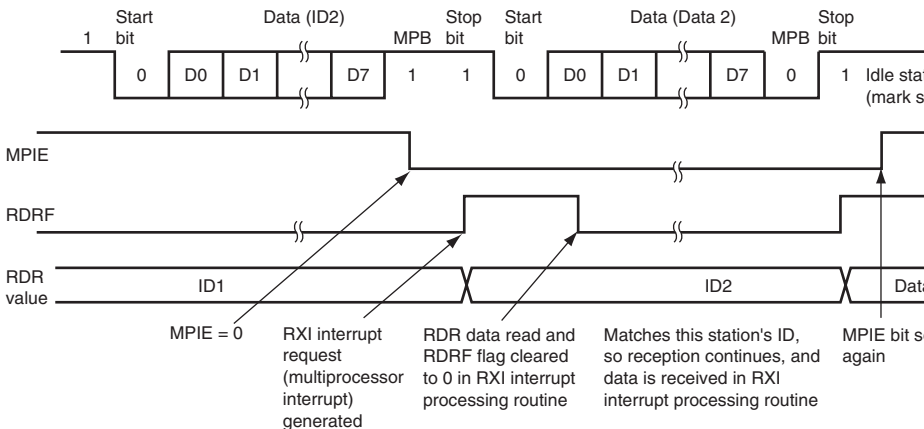


- to 1, a 1 is output for one frame, and transmission is enabled.
- [2] SCI status check and transmit data write:  
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. Set the MPBT bit in SSR to 0 or 1. Finally, clear the TDRE flag to 0.
- [3] Serial transmission continuation procedure:  
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DMAC or DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.
- [4] Break output at the end of serial transmission:  
To output a break in serial transmission, set DDR for the port to 1, clear DR to 0, and then clear the TE bit in SCR to 0.

**Figure 19.14 Sample Multiprocessor Serial Transmission Flowchart**

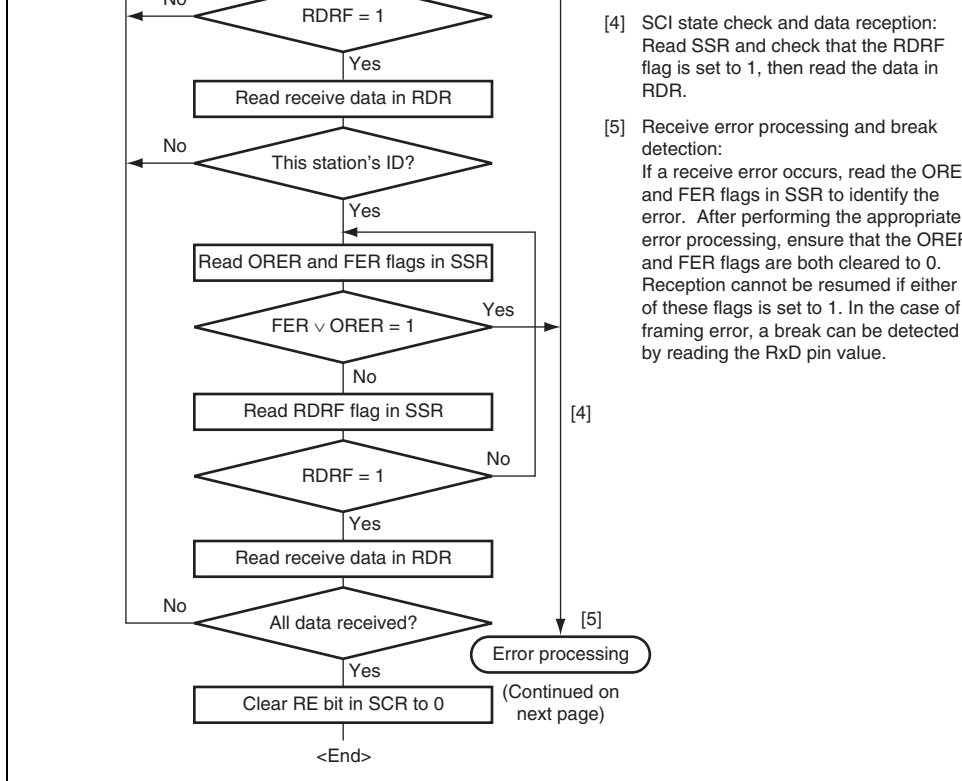


(a) Data does not match station's ID

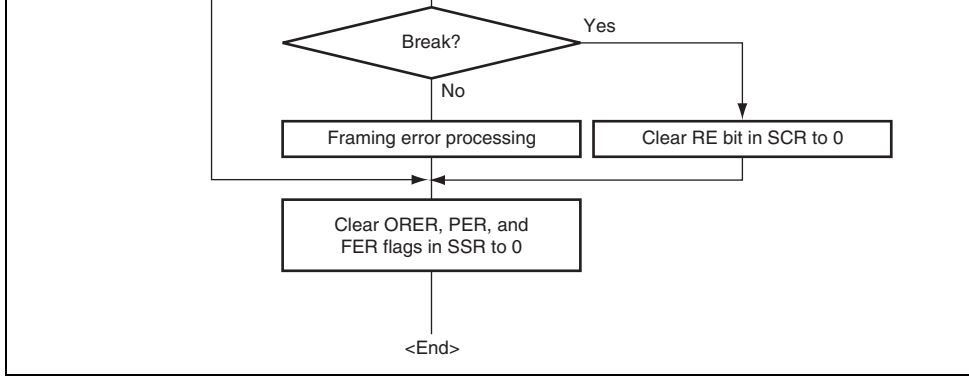


(b) Data matches station's ID

**Figure 19.15 Example of SCI Operation for Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

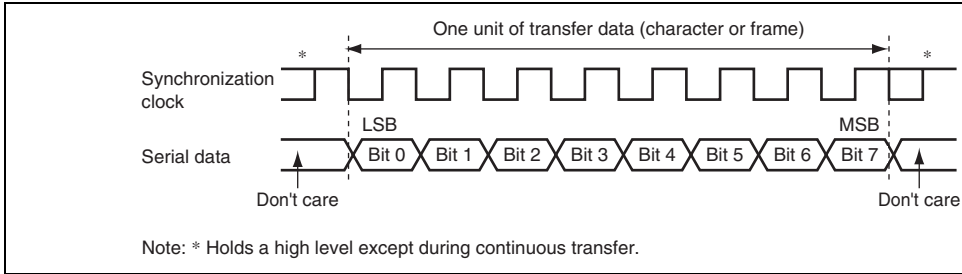


**Figure 19.16 Sample Multiprocessor Serial Reception Flowchart (1)**



**Figure 19.16 Sample Multiprocessor Serial Reception Flowchart (2)**

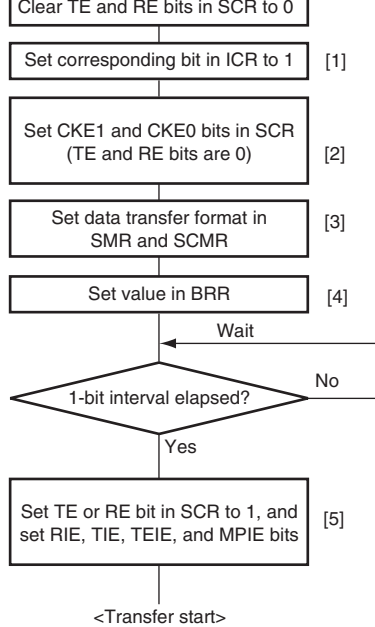
receiver also have a double buffered structure, so that the next transmit data can be written during the previous transmission or the previous receive data can be read during reception, enabling continuous transfer. (Setting is prohibited in SCI\_5 and SCI\_6.)



**Figure 19.17 Data Format in Clocked Synchronous Communication (LSB-First)**

### 19.6.1 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 SCKE 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 overrun error flag is cleared to 0. (Setting is prohibited in SCI\_5 and SCI\_6.)



- [1] pin when receiving data or using an external clock.
- [2] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.
- [3] Set the data transfer format in SMR and SCMR.
- [4] Write a value corresponding to the bit rate to BRR. This step is not necessary if an external clock is used.
- [5] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits. Setting the TE and RE bits enables the TxD and RxD pins to be used.

Note: In simultaneous transmit and receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

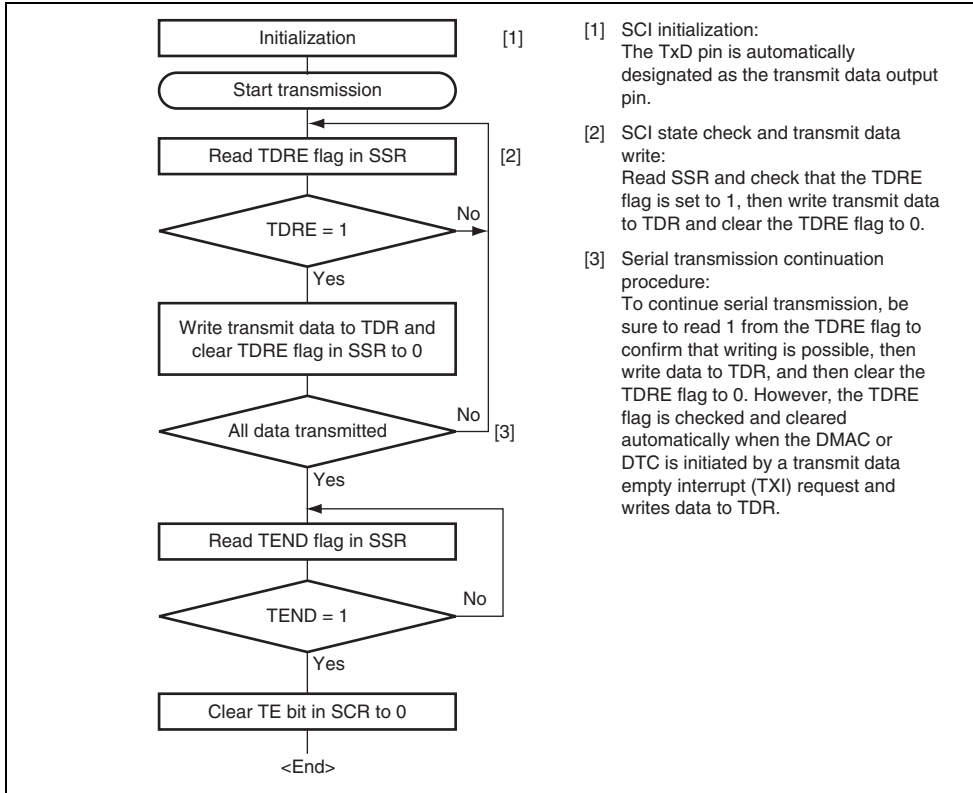
**Figure 19.18 Sample SCI Initialization Flowchart**



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 TDRH 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 is generated. The SCK pin is fixed high.

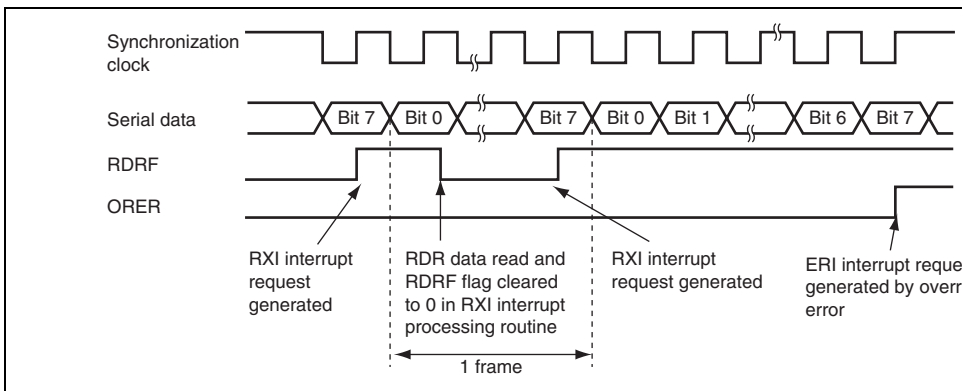
Figure 19.20 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.

**Figure 19.19 Example of Operation for Transmission in Clocked Synchronous**



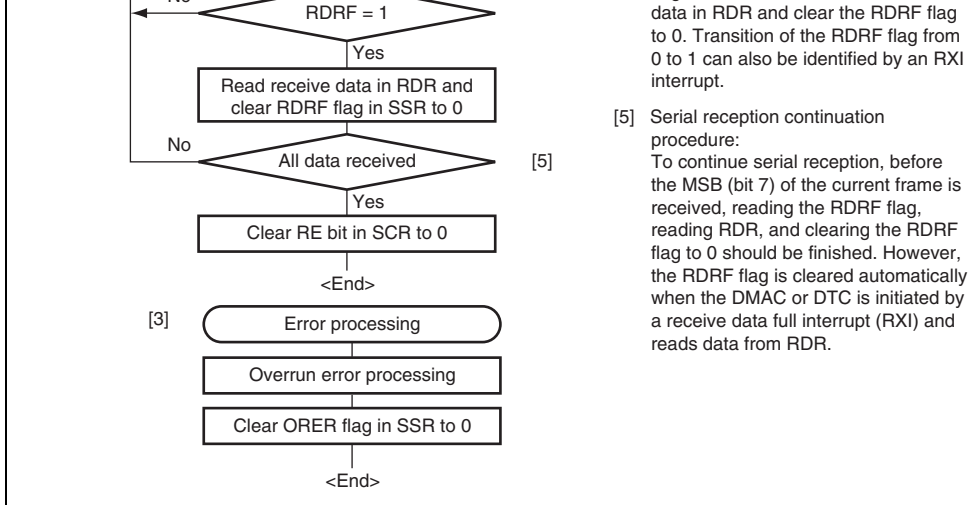
**Figure 19.20 Sample Serial Transmission Flowchart**

- 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 19.21 Example of Operation for Reception in Clocked Synchronous Mode**

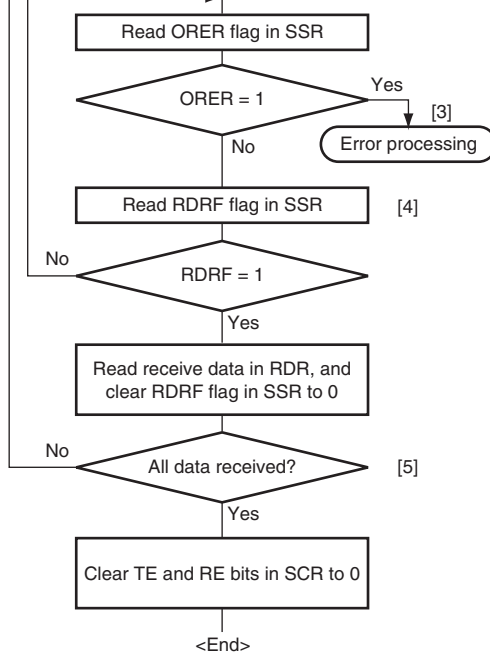
Transfer cannot be resumed while a receive error flag is set to 1. Accordingly, clear the FER, PER, and RDRF bits to 0 before resuming reception. Figure 19.22 shows a sample timing diagram for serial data reception.



**Figure 19.22 Sample Serial Reception Flowchart**

### 19.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode) (SCI\_0, 1, 2, and 4 only)

Figure 19.23 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial 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 TE 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.

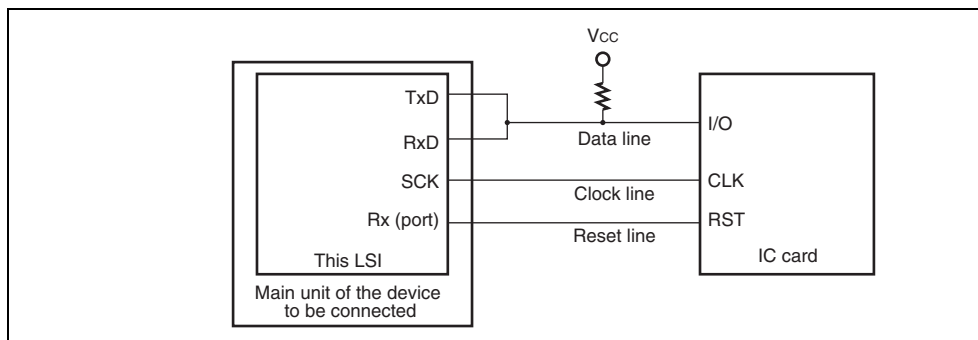


Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

- ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Reception cannot be resumed if the ORER flag is set to 1.
- [4] SCI state check and receive data read:  
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
- [5] Serial transmission/reception continuation procedure:  
To continue serial transmission/reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DMAC or DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR. Similarly, the RDRF flag is cleared automatically when the DMAC or DTC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

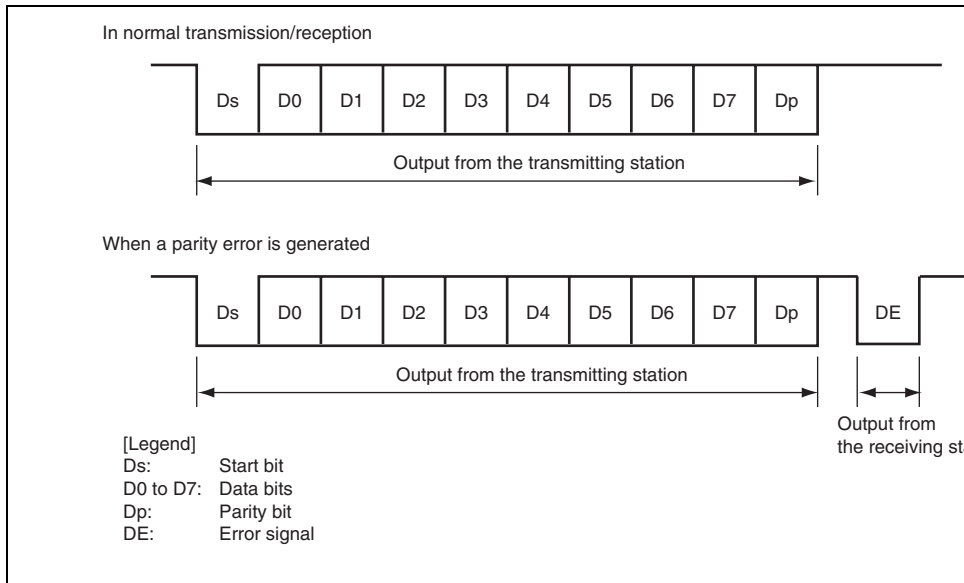
**Figure 19.23 Sample Flowchart of Simultaneous Serial Transmission and Reception**

TxD and RxD pins and pull up the data transmission line to  $V_{CC}$  using a resistor. Setting TE bits to 1 with the smart card not connected enables closed transmission/reception self diagnosis. To supply the smart card with the clock pulses generated by the SCI, input pin output to the CLK pin of the smart card. A reset signal can be supplied via the output pin of this LSI. (In SCI\_5 and SCI-6, the clock generated in SCI cannot be provided to smart card)



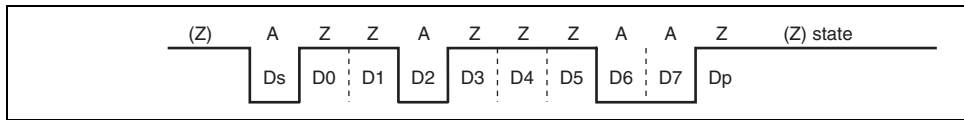
**Figure 19.24 Pin Connection for Smart Card Interface**

after at least 2 etu.



**Figure 19.25 Data Formats in Normal Smart Card Interface Mode**

For communication with the smart cards of the direct convention and inverse convention follow the procedure below.



**Figure 19.26 Direct Convention (SDIR = SINV =  $\overline{O/E} = 0$ )**

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 19.27. The 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 SNINV bit in this LSI only inverts data bits D7 to D0, write 1 to the O/E bit in SMR to invert the parity bit for both transmission and reception.

### 19.7.3 Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following:

- 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 1 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 transmitted.



$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\%$$

[Legend]

M: Reception margin (%)

N: Ratio of bit rate to clock (N = 32, 64, 372, 256)

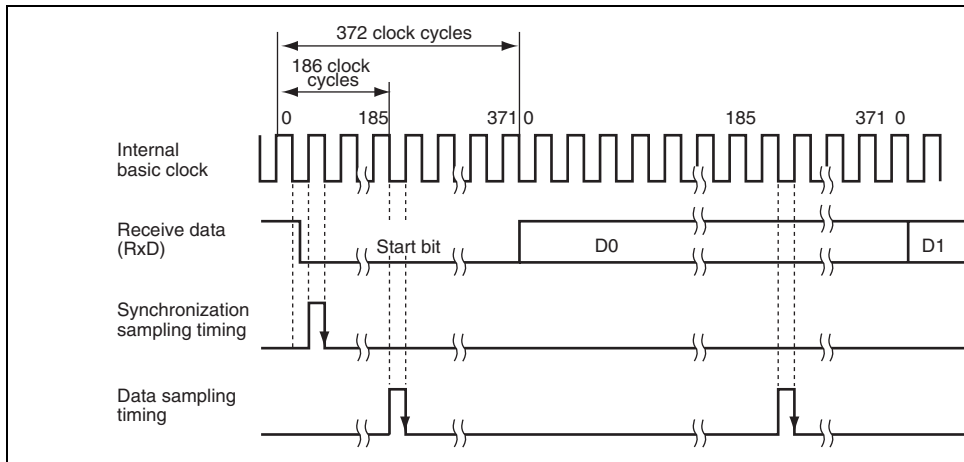
D: Duty cycle of clock (D = 0 to 1.0)

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 19.28 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency is 372 Times the Bit Rate)**

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 data reception.

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.

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-transmission. 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 19.31 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 as 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 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 12, Data Transfer Controller (DTC) and section 10, DMA Controller (DMAC).

Note that the TEND flag is set in different timings depending on the GM bit setting in SM. Figure 19.30 shows the TEND flag set timing.

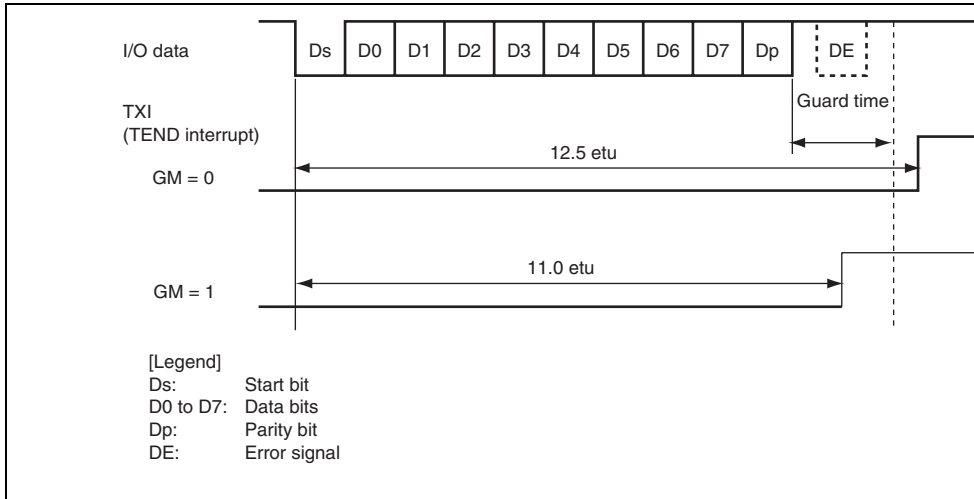
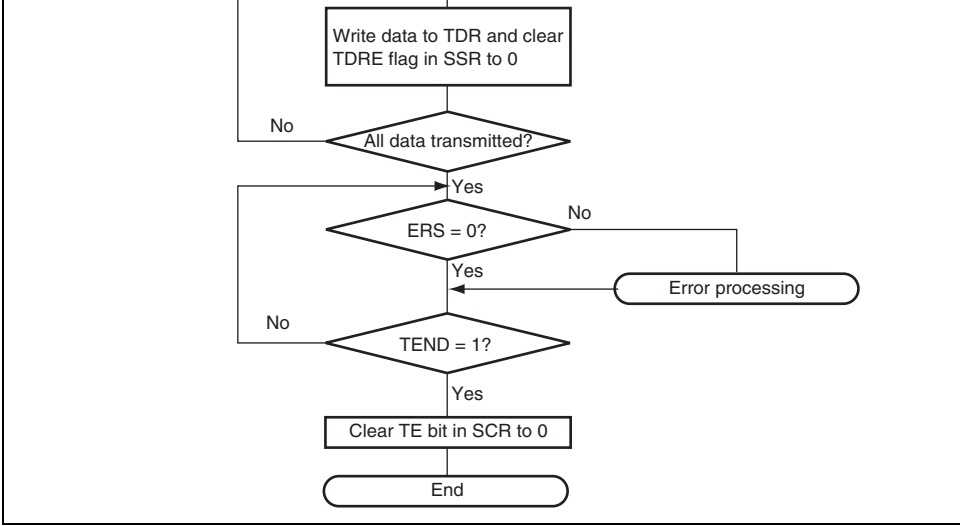


Figure 19.30 TEND Flag Set Timing during Transmission

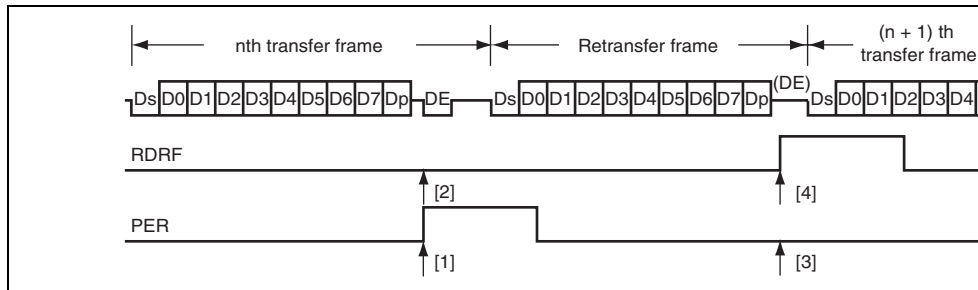


**Figure 19.31 Sample Transmission Flowchart**

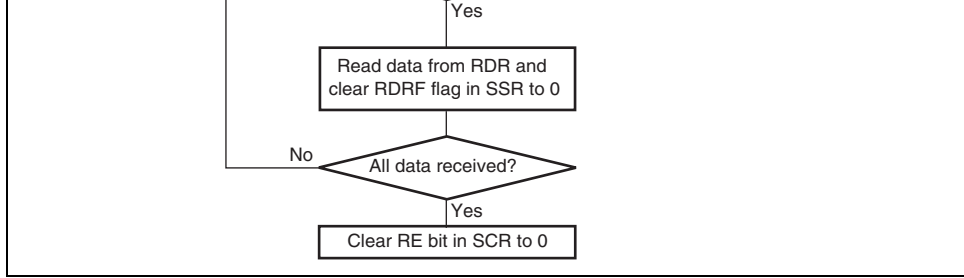
4. In this case, data is determined to have been received successfully, and the RDRF bit is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set to 1.

Figure 19.33 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 interrupt request activates the DTC or DMAC by an RXI request, thus allowing transfer of receive data if the interrupt request is specified as a source of DTC or DMAC activation beforehand. The RDRF bit is automatically cleared to 0 at data transfer by the DTC or DMAC. If an error occurs during reception, i.e., either the ORE or PER flag is set to 1, a transmit/receive error interrupt (TXR/REI) interrupt 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 during reception, receive data is transferred to RDR, thus allowing the data to be read.

Note: For operations in block transfer mode, see section 19.4, Operation in Asynchronous Mode.



**Figure 19.32 Data Re-Transfer Operation in SCI Reception Mode**

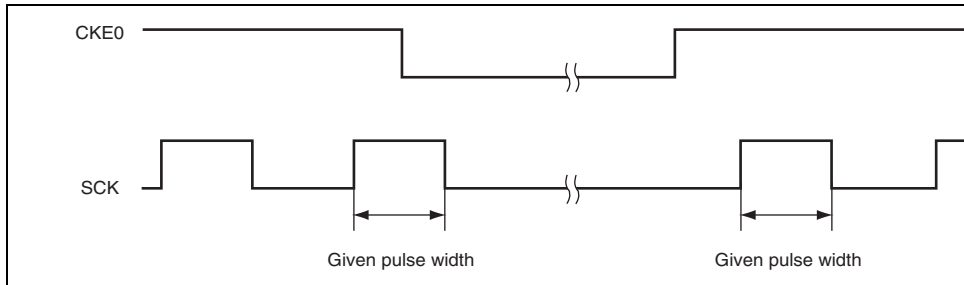


**Figure 19.33 Sample Reception Flowchart**

### 19.7.8 Clock Output Control (Only SCI\_0, 1, 2, and 4)

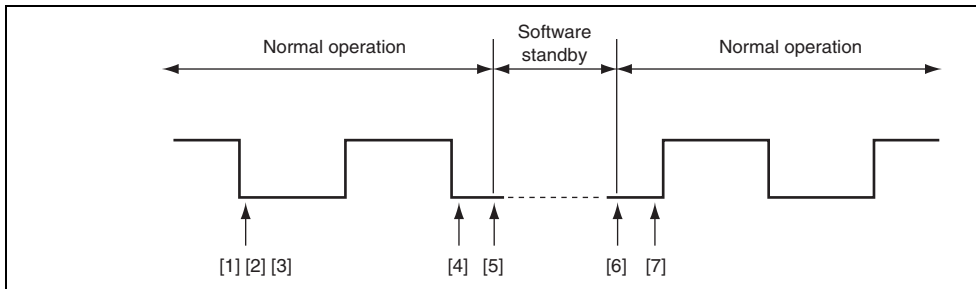
Clock output can be fixed using the CKE1 and CKE0 bits in SCR when the GM bit in SPCR is set to 1. Specifically, the minimum width of a clock pulse can be specified.

Figure 19.34 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 0.



**Figure 19.34 Clock Output Fixing Timing**

- Set the CKE0 bit in SCR to 1 to start clock output.
- At mode switching
  - a) 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 pin to the values for the output fixed state in software standby mode. (SCI\_0, 1, 4 only)
    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 at the specified level with the duty cycle retained.
    5. Make the transition to software standby mode.
  - b) At transition from software standby mode to smart card interface mode
    1. Clear software standby mode.
    2. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty cycle is then generated.

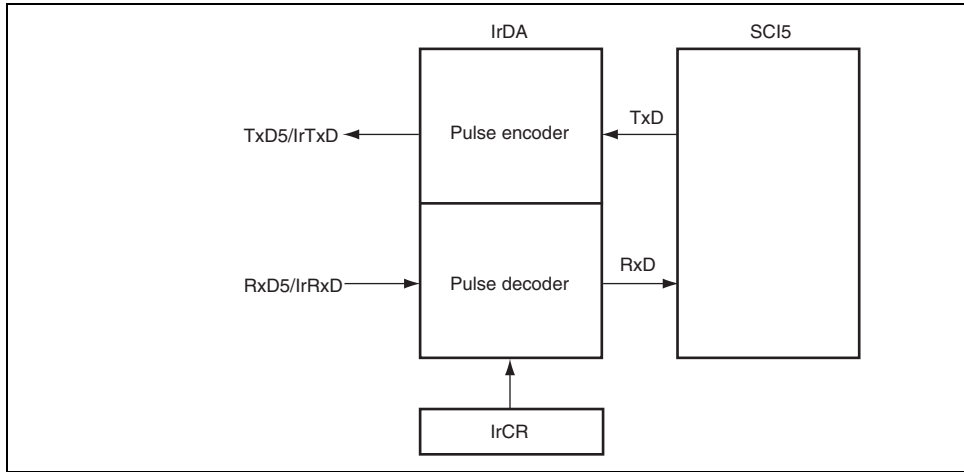


**Figure 19.35 Clock Stop and Restart Procedure**



rate, the transfer rate must be modified through programming.

Figure 19.36 is the IrDA block diagram.

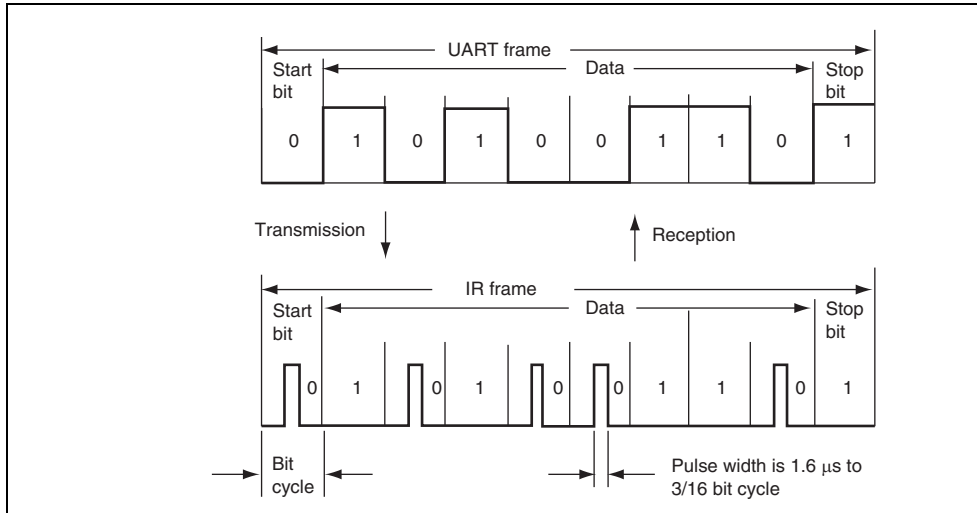


**Figure 19.36 IrDA Block Diagram**

Note: \* The IrDA function should be used when the ABCS bit in SEMR\_5 is set to 0. ACS3 to ACS0 bits in SEMR\_5 are set to B'0000.

time, a high-level pulse width of 1.6  $\mu$ s can be specified because it is the smallest value in the range greater than 1.41  $\mu$ s.

For serial data of level 1, no pulses are output.



**Figure 19.37 IrDA Transmission and Reception**

## (2) Reception

During reception, IR frames are converted to UART frames using the IrDA interface before inputting to SCI. 0 is output when the high level pulse is detected while 1 is output when a low level pulse is detected during one bit period. Note that a pulse shorter than the minimum pulse width of 1.6  $\mu$ s is also regarded as a 0 signal.

7.3728	100	100	100	100	100	100
8	100	100	100	100	100	100
9.8304	100	100	100	100	100	100
10	100	100	100	100	100	100
12	101	101	101	101	101	101
12.288	101	101	101	101	101	101
14	101	101	101	101	101	101
14.7456	101	101	101	101	101	101
16	101	101	101	101	101	101
17.2032	101	101	101	101	101	101
18	101	101	101	101	101	101
19.6608	101	101	101	101	101	101
20	101	101	101	101	101	101
25	110	110	110	110	110	110
30	110	110	110	110	110	110
33	110	110	110	110	110	110
35	110	110	110	110	110	110

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 request activates 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 TXI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority over the TEI interrupt. However, note that if the TDRE and TEND flags are cleared to 0 simultaneously, the TXI interrupt processing routine, the SCI cannot branch to the TEI interrupt processing routine later.

Note that the priority order for interrupts is different between the group of SCI\_0, 1, 2, and 4 and the group of SCI\_5 and SCI\_6.

**Table 19.14 SCI Interrupt Sources (SCI\_0, 1, 2, and 4)**

Name	Interrupt Source	Interrupt Flag	DTC Activation	DMAC Activation
ERI	Receive error	ORER, FER, or PER	Not possible	Not possible
RXI	Receive data full	RDRF	Possible	Possible
TXI	Transmit data empty	TDRE	Possible	Possible
TEI	Transmit end	TEND	Not possible	Not possible

Table 19.16 shows the interrupt sources in smart card interface mode. A transmit end (TX) interrupt request cannot be used in this mode.

Note that the priority order for interrupts is different between the group of SCI\_0, 1, 2, and 3 and the group of SCI\_5 and SCI\_6.

**Table 19.16 SCI Interrupt Sources (SCI\_0, 1, 2, and 4)**

<b>Name</b>	<b>Interrupt Source</b>	<b>Interrupt Flag</b>	<b>DTC Activation</b>	<b>DMAC Activation</b>
ERI	Receive error or error signal detection	ORER, PER, or ERS	Not possible	Not possible
RXI	Receive data full	RDRF	Possible	Possible
TXI	Transmit data empty	TEND	Possible	Possible

**Table 19.17 SCI Interrupt Sources (SCI\_5 and SCI\_6)**

<b>Name</b>	<b>Interrupt Source</b>	<b>Interrupt Flag</b>	<b>DTC Activation</b>	<b>DMAC Activation</b>
RXI	Receive data full	RDRF	Not possible	Possible
TXI	Transmit data empty	TDRE	Not possible	Possible
ERI	Receive error or error signal detection	ORER, PER, or ERS	Not possible	Not possible

error occurrence.

When transmitting/receiving data using the DTC or DMAC, be sure to set and enable the DMAC prior to making SCI settings. For DTC or DMAC settings, see section 12, Data Transfer Controller (DTC) and section 10, DMA Controller (DMAC).

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 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, the RDRF flag is not set but the error flag is set. Therefore, the DTC or DMAC is not activated and no interrupt request is issued to the CPU instead; the error flag must be cleared.

When framing error detection is performed, a break can be detected by reading the RxD pin directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set. The PER flag may also be set. Note that, since the SCI continues the receive operation even when receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

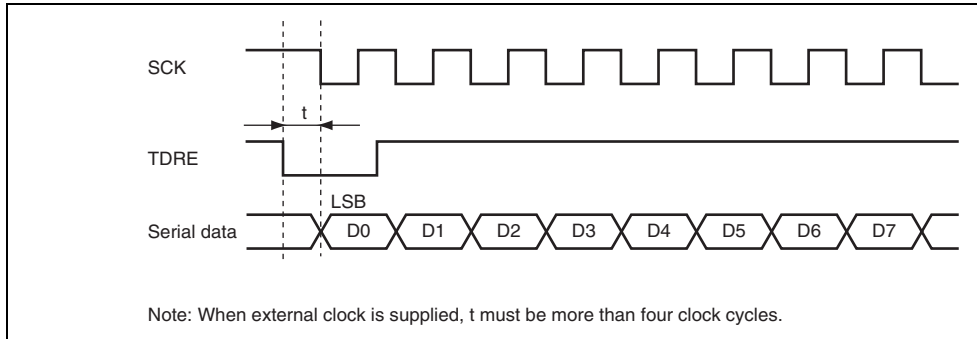
### 19.10.3 Mark State and Break Detection

When the TE bit is 0, the TxD pin is used as an I/O port whose direction (input or output) and data level are determined by DR and DDR. This can be used to set the TxD pin to mark state (input level) or send a break during serial data transmission. To maintain the communication line in a 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. When the TE bit is set to 1, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

### 19.10.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) is set to 1. When 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 REIE bit is cleared to 0.

- When the external clock source is used as a synchronization clock, update TDR by the TXIE or DTC 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 transmitter may malfunction (see figure 19.38).
- When using the DMAC or DTC to read RDR, be sure to set the receive end interrupt to the DTC or DMAC activation source.



**Figure 19.38 Sample Transmission using DTC in Clocked Synchronous Mode**

- The DTC is not activated by the RXI or TXI request by SCI\_5 or SCI6.



Figure 19.39 shows a sample flowchart for transition to software standby mode during transmission. Figures 19.40 and 19.41 show the port pin states during transition to software standby mode.

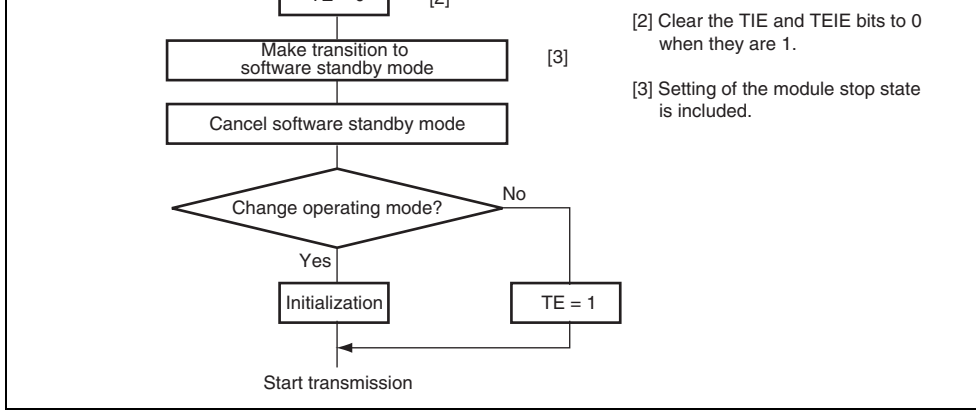
Before specifying the module stop state or making a transition to software standby mode during transmission mode using DTC transfer, stop all transmit operations ( $TE = TIE = TEIE = 0$ ). Setting the TE and TIE bits to 1 after cancellation sets the TXI flag to start transmission. DTC.

**Reception:** Before specifying the module stop state or making a transition to software standby mode, stop the receive operations ( $RE = 0$ ). RSR, RDR, and SSR are reset. If transition to software standby mode occurs during data reception, the data being received will be invalid.

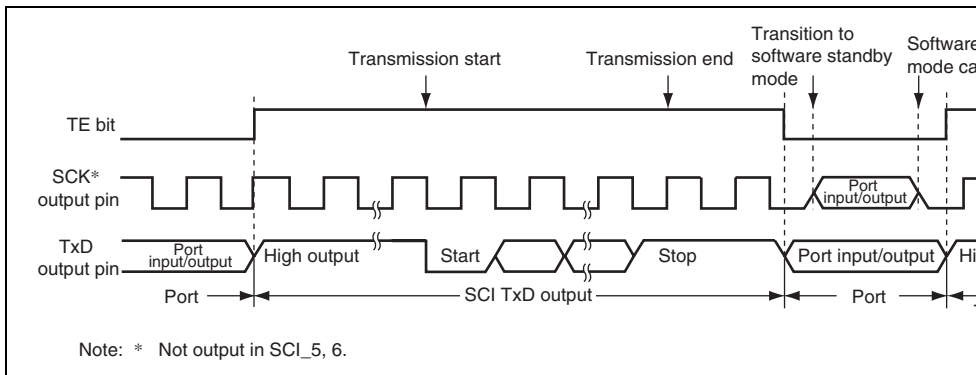
To receive data in the same reception mode after cancellation of the power-down state, set the RE bit to 1, and then start reception. To receive data in a different reception mode, initialize the mode first.

For using the IrDA function, set the IrE bit in addition to setting the RE bit.

Figure 19.42 shows a sample flowchart for mode transition during reception.

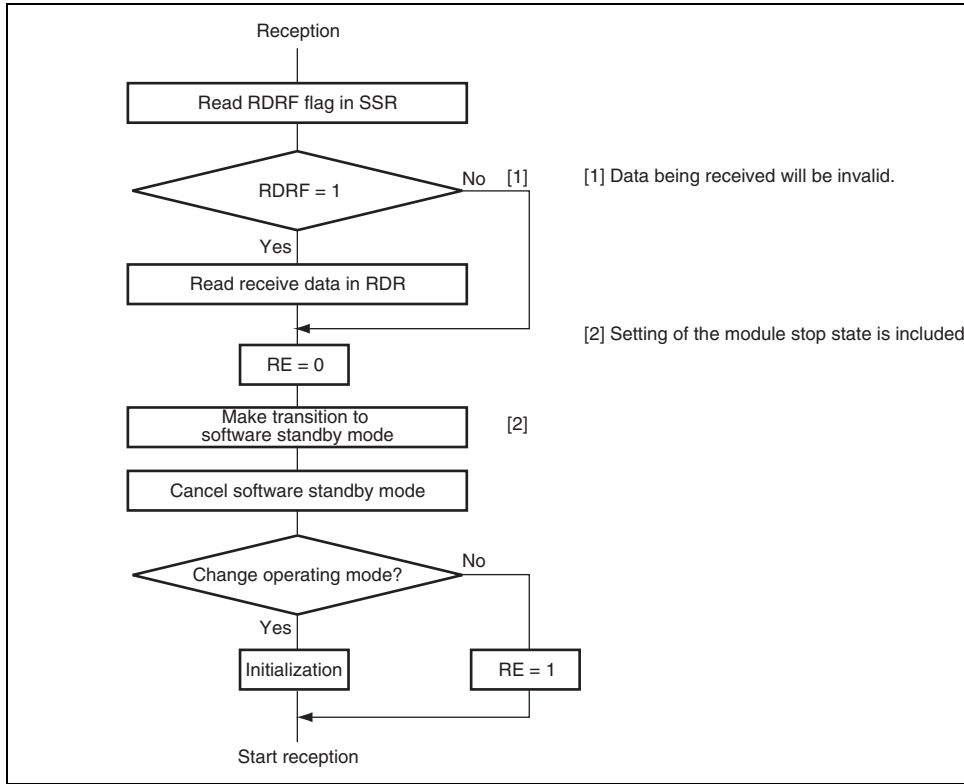


**Figure 19.39 Sample Flowchart for Software Standby Mode Transition during Transmission**



**Figure 19.40 Port Pin States during Software Standby Mode Transition (Internal Clock, Asynchronous Transmission)**

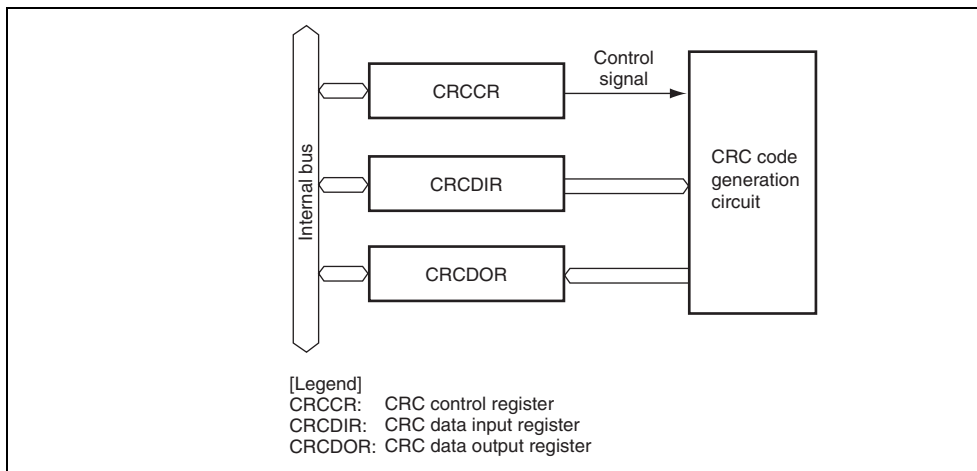
**Figure 19.41 Port Pin States during Software Standby Mode Transition  
(Internal Clock, Clocked Synchronous Transmission)  
(Setting is Prohibited in SCI\_5 and SCI\_6)**



**Figure 19.42 Sample Flowchart for Software Standby Mode Transition during Reception**

- One of three generating polynomials selectable
- CRC code generation for LSB-first or MSB-first communication selectable

Figure 19.43 shows a block diagram of the CRC operation circuit.



**Figure 19.43 Block Diagram of CRC Operation Circuit**

generating polynomial.

Bit	7	6	5	4	3	2	1
Bit Name	DORCLR	—	—	—	—	LMS	G1
Initial Value	0	0	0	0	0	0	0
R/W	W	R	R	R	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	DORCLR	0	W	CRCDOR Clear Setting this bit to 1 clears CRCDOR to H'0000.
6 to 3	—	All 0	R	Reserved The initial value should not be changed.
2	LMS	0	R/W	CRC Operation Switch Selects CRC code generation for LSB-first or MSB-first communication. 0: Performs CRC operation for LSB-first communication. The lower byte (bits 7 to 0) of the CRC is transmitted when CRCDOR contents (CRCDOR) are divided into two bytes to be transmitted parts. 1: Performs CRC operation for MSB-first communication. The upper byte (bits 15 to 8) of the CRC is transmitted when CRCDOR contents (CRCDOR) are divided into two bytes to be transmitted parts.

CRCDIR is an 8-bit readable/writable register, to which the bytes to be CRC-operated are written. The result is obtained in CRCDOR.

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

### (3) CRC Data Output Register (CRCDOR)

CRCDOR is a 16-bit readable/writable register that contains the result of CRC operation. The bytes to be CRC-operated are written to CRCDIR after CRCDOR is cleared. When the CRC operation result is additionally written to the bytes to which CRC operation is to be performed, the CRC operation result will be H'0000 if the data contains no CRC error. When bits 1 and 0 of CRCCR (G1 and G0 bits) are set to 0 and 1, respectively, the lower byte of this register contains the result.

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

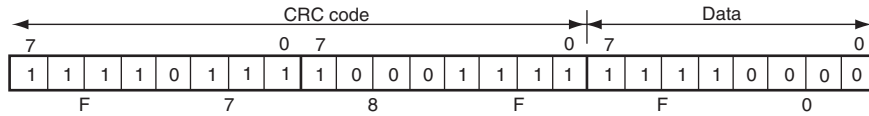
Bit	7	6	5	4	3	2	1
Bit Name							
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	7							0
CRCDORH	0	0	0	0	0	0	0	0
CRCDORL	0	0	0	0	0	0	0	0

	7							0
CRCDORH	1	1	1	1	0	1	1	1
CRCDORL	1	0	0	0	1	1	1	1

3. Read from CRCDOR  
CRC code = H'F78F

4. Serial transmission (LSB first)



**Figure 19.44 LSB-First Data Transmission**

1. Write H'87 to CRCCLR

	7							0
CRCCLR	1	0	0	0	0	1	1	1

↓ CRCDOR clearing

	7							0
CRCDORH	0	0	0	0	0	0	0	0
CRCDORL	0	0	0	0	0	0	0	0

2. Write H'F0 to CRCDIR

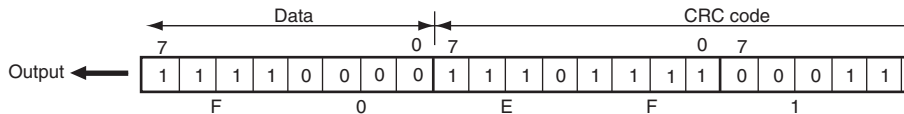
	7							0
CRCDIR	1	1	1	1	0	0	0	0

↓ CRC code generation

	7							0
CRCDORH	1	1	1	0	1	1	1	1
CRCDORL	0	0	0	1	1	1	1	1

3. Read from CRCDOR  
CRC code = H'EF1F

4. Serial transmission (MSB first)




**Figure 19.45 MSB-First Data Transmission**

CRCDORH	7	0	0	0	0	0	0	0	0	0
CRCDORL	0	0	0	0	0	0	0	0	0	0

CRCDORH	7	1	1	1	1	0	1	1	1	0
CRCDORL	0	1	0	0	0	1	1	1	1	1

4. Write H'8F to CRCDIR


CRCDIR	7	1	0	0	0	1	1	1	1	0
--------	---	---	---	---	---	---	---	---	---	---


 CRC code generation

CRCDORH	7	0	0	0	0	0	0	0	0	0
CRCDORL	0	1	1	1	1	0	1	1	1	1

5. Write H'F7 to CRCDIR

CRCDIR	7	1	1	1	1	0	1	1	1	0
--------	---	---	---	---	---	---	---	---	---	---


 CRC code genera

CRCDORH	7	0	0	0	0	0	0	0	0	0
CRCDORL	0	0	0	0	0	0	0	0	0	0

6. Read from CRCDOR

CRC code = H'0000 → No error

**Figure 19.46 LSB-First Data Reception**




	7	6	5	4	3	2	1	0
CRCDORH	0	0	0	0	0	0	0	0
CRCDORL	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
CRCDORH	1	1	1	0	1	1	1	1
CRCDORL	0	0	0	1	1	1	1	1

4. Write H'EF to CRCDIR


	7	6	5	4	3	2	1	0
CRCDIR	1	1	1	0	1	1	1	1


 CRC code generation

	7	6	5	4	3	2	1	0
CRCDORH	0	0	0	1	1	1	1	1
CRCDORL	0	0	0	0	0	0	0	0

5. Write H'1F to CRCDIR

	7	6	5	4	3	2	1	0
CRCDIR	0	0	0	1	1	1	1	1


 CRC code generation

	7	6	5	4	3	2	1	0
CRCDORH	0	0	0	0	0	0	0	0
CRCDORL	0	0	0	0	0	0	0	0

6. Read from CRCDOR

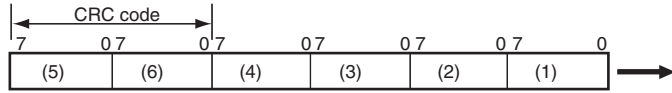
CRC code = H'0000 → No error

**Figure 19.47 MSB-First Data Reception**

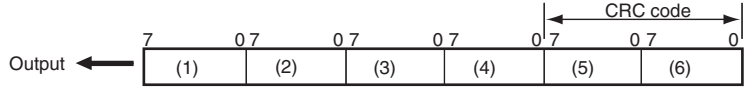
CRCDORH	(5)
CRCDORL	(6)

2. Transmission data

(i) LSB-first transmission



(ii) MSB-first transmission



**Figure 19.48 LSB-First and MSB-First Transmit Data**

- Transfer speed: Supports full-speed (12 Mbps)
- Endpoint configuration:

Endpoint Name	Abbreviation	Transfer Type	Maximum Packet Size	FIFO Buffer Capacity (Byte)	DMA Tr
Endpoint 0	EP0s	Setup	8	8	—
	EP0i	Control-in	8	8	—
	EP0o	Control-out	8	8	—
Endpoint 1	EP1	Bulk-out	64	128	Possible
Endpoint 2	EP2	Bulk-in	64	128	Possible
Endpoint 3	EP3	Interrupt-in	8	8	—

Configuration1-Interface0-AlternateSetting0

- └ EndPoint1
- └ EndPoint2
- └ EndPoint3

- Interrupt requests: Generates various interrupt signals necessary for USB transmission/reception
- Power mode: Self power mode or bus power mode can be selected by the power mode (PWMD) in the control register (CTLR).

[Legend]

UDC: USB device controller

**Figure 20.1 Block Diagram of USB**

## 20.2 Input/Output Pins

Table 20.1 shows the USB pin configuration.

**Table 20.1 Pin Configuration**

Pin Name	I/O	Function
VBUS	Input	USB cable connection monitor pin
USD+	I/O	USB data I/O pin
USD-	I/O	USB data I/O pin
DrVcc	Input	Power supply pin for USB on-chip transceiver
DrVss	Input	Ground pin for USB on-chip transceiver

- Interrupt select register 2 (ISR2)
- Interrupt enable register 0 (IER0)
- Interrupt enable register 1 (IER1)
- Interrupt enable register 2 (IER2)
- EP0i data register (EPDR0i)
- EP0o data register (EPDR0o)
- EP0s data register (EPDR0s)
- EP1 data register (EPDR1)
- EP2 data register (EPDR2)
- EP3 data register (EPDR3)
- EP0o receive data size register (EPSZ0o)
- EP1 receive data size register (EPSZ1)
- Trigger register (TRG)
- Data status register (DASTS)
- FIFO clear register (FCLR)
- DMA transfer setting register (DMA)
- Endpoint stall register (EPSTL)
- Configuration value register (CVR)
- Control register (CTLR)
- Endpoint information register (EPIR)
- Transceiver test register 0 (TRNTREG0)
- Transceiver test register 1 (TRNTREG1)

Bit	Bit Name	Initial Value	R/W	Description
7	BRST	0	R/W	<p><b>Bus Reset</b></p> <p>This bit is set to 1 when a bus reset signal is detected on the USB bus.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
6	EP1 FULL	0	R	<p><b>EP1 FIFO Full</b></p> <p>This bit is set when endpoint 1 receives one packet of data successfully from the host, and holds a valid packet as long as there is valid data in the FIFO buffer.</p> <p>This is a status bit, and cannot be cleared.</p>
5	EP2 TR	0	R/W	<p><b>EP2 Transfer Request</b></p> <p>This bit is set if there is no valid transmit data in the FIFO buffer when an IN token for endpoint 2 is received from the host. A NACK handshake is returned to the host. A NACK handshake is returned to the host if the data is written to the FIFO buffer and packet transmission is enabled.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>

(When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

---

2	EP0o TS	0	R/W	EP0o Receive Complete  This bit is set to 1 when endpoint 0 receives data from the host successfully, stores the data in the FIFO buffer, and returns an ACK handshake to the host.  (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	EP0i TR	0	R/W	EP0i Transfer Request  This bit is set if there is no valid transmit data in the FIFO buffer when an IN token for endpoint 0 is received from the host. A NACK handshake is returned to the host until data is written to the FIFO buffer and transmission is enabled.  (When the CPU is used to clear this flag by writing 0 to it, while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
0	EP0i TS	0	R/W	EP0i Transmit Complete  This bit is set when data is transmitted to the host from endpoint 0 and an ACK handshake is returned to the host.  (When the CPU is used to clear this flag by writing 0 to it, 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
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	VBUS MN	0	R	<p>This is a status bit which monitors the state of the VBUS pin.</p> <p>This bit reflects the state of the VBUS pin and generates no interrupt request. This bit is always 0 when the PULLUP_E bit in DMA is 0.</p>
2	EP3 TR	0	R/W	<p>EP3 Transfer Request</p> <p>This bit is set if there is no valid transmit data in the EP3 FIFO buffer when an IN token for endpoint 3 is received from the host. A NACK handshake is sent to the host until data is written to the FIFO buffer and packet transmission is enabled.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
1	EP3 TS	0	R/W	<p>EP3 Transmit Complete</p> <p>This bit is set when data is transmitted to the host through endpoint 3 and an ACK handshake is returned from the host.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>



### 20.3.3 Interrupt Flag Register 2 (IFR2)

IFR2, together with interrupt flag registers 0 and 1 (IFR0 and IFR1), indicates interrupt information required by the application. When an interrupt source is generated, the corresponding bit is set to 1. And then this bit, in combination with interrupt enable register 2 (IER2), generates an interrupt request to the CPU. To clear, write 0 to the bit to be cleared and 1 to the other bits.

Bit	7	6	5	4	3	2	1
Bit Name	—	—	SURSS	SURSF	CFDN	—	SETC
Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R/W	R/W	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	SURSS	0	R	Suspend/Resume Status This is a status bit that describes bus state. 0: Normal state 1: Suspended state This bit is a status bit and generates no interrupt request.

This bit is set to 1 when writing data in the endpoint information register to the EPIR register ends (end). This module starts the USB operation after the endpoint information is completely set.

(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

2	—	0	R	Reserved	<p>This bit is always read as 0. The write value should always be 0.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
1	SETC	0	R/W	Set_Configuration Command Detection	<p>When the Set_Configuration command is detected, this bit is set to 1.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
0	SETI	0	R/W	Set_Interface Command Detection	<p>When the Set_Interface command is detected, this bit is set to 1.</p> <p>(When the CPU is used to clear this flag by writing 0 to it while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>

R/W

R/W

R/W

R/W

R/W

R/W

R/W

R/W

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	BRST	0	R/W	Bus Reset
6	EP1 FULL	0	R/W	EP1 FIFO Full
5	EP2 TR	0	R/W	EP2 Transfer Request
4	EP2 EMPTY	0	R/W	EP2 FIFO Empty
3	SETUP TS	0	R/W	Setup Command Receive Complete
2	EP0o TS	0	R/W	EP0o Receive Complete
1	EP0i TR	0	R/W	EP0i Transfer Request
0	EP0i TS	0	R/W	EP0i Transmission Complete

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	—	0	R	
2	EP3 TR	1	R/W	EP3 Transfer Request
1	EP3 TS	1	R/W	EP3 Transmission Complete
0	VBUSF	1	R/W	USB Bus Connect

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	SURSE	1	R/W	Suspend/Resume Detection
3	CFDN	1	R/W	End Point Information Load End
2	—	1	R	Reserved This bit is always read as 1. The write value should always be 1.
1	SETCE	1	R/W	Set_Configuration Command Detection
0	SETIE	1	R/W	Set_Interface Command Detection



R/W

R/W

R/W

R/W

R/W

R/W

R/W

R/W

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	BRST	0	R/W	Bus Reset
6	EP1 FULL	0	R/W	EP1 FIFO Full
5	EP2 TR	0	R/W	EP2 Transfer Request
4	EP2 EMPTY	0	R/W	EP2 FIFO Empty
3	SETUP TS	0	R/W	Setup Command Receive Complete
2	EP0o TS	0	R/W	EP0o Receive Complete
1	EP0i TR	0	R/W	EP0i Transfer Request
0	EP0i TS	0	R/W	EP0i Transmission Complete

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	—	0	R	
2	EP3 TR	0	R/W	EP3 Transfer Request
1	EP3 TS	0	R/W	EP3 Transmission Complete
0	VBUSF	0	R/W	USB Bus Connect

**20.3.9 Interrupt Enable Register 2 (IER2)**

IER2 enables the interrupt requests of interrupt flag register 2 (IFR2). When an interrupt request bit is set to 1 while the corresponding bit of each interrupt is set to 1, an interrupt request is sent to the CPU. The interrupt vector number is determined by the contents of interrupt select register 2 (ISR2).

Bit	7	6	5	4	3	2	1
Bit Name	SSRSME	—	—	SURSE	CFDN	—	SETCE
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R	R/W

3	CFDN	0	R/W	End Point Information Load End
2	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.
1	SETCE	0	R/W	Set_Configuration Command Detection
0	SETIE	0	R/W	Set_Interface Command Detection

### 20.3.10 EP0i Data Register (EPDR0i)

EPDR0i is an 8-byte transmit FIFO buffer for endpoint 0. EPDR0i holds one packet of transmit data for control-in. Transmit data is fixed by writing one packet of data and setting EP0iP in the trigger register. When an ACK handshake is returned from the host after the data has been transmitted, EP0iTS in interrupt flag register 0 is set. This FIFO buffer can be initialized by writing 0 to EP0iCLR in the FCLR register.

Bit	7	6	5	4	3	2	1	
Bit Name	D7	D6	D5	D4	D3	D2	D1	
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	W	W	W	W	W	W	W	W

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for control-in transfer



Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	R	Data register for control-out transfer

### 20.3.12 EP0s Data Register (EPDR0s)

EPDR0s is an 8-byte FIFO buffer specifically for receiving endpoint 0 setup commands. A setup command to be processed by the application is received. When command data is received successfully, the SETUPTS bit in interrupt flag register 0 is set.

As a latest setup command must be received in high priority, if data is left in this buffer, it is overwritten with new data. If reception of the next command is started while the current command is being read, command reception has priority, the read by the application is forcibly stopped, and the read data is invalid.

Bit	7	6	5	4	3	2	1
Bit Name	D7	D6	D5	D4	D3	D2	D1
Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	R	Data register for storing the setup command data for control-out transfer

Bit Name	D7	D6	D5	D4	D3	D2	D1	
Initial Value	0	0	0	0	0	0	0	
R/W	R	R	R	R	R	R	R	

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	R	Data register for endpoint 1 transfer

### 20.3.14 EP2 Data Register (EPDR2)

EPDR2 is a 128-byte transmit FIFO buffer for endpoint 2. EPDR2 has a dual-buffer configuration and has a capacity of twice the maximum packet size. When transmit data is written to the buffer and EP2PKTE in the trigger register is set, one packet of transmit data is fixed, and the dual-FIFO buffer is switched over. The transmit data for this FIFO buffer can be transferred by DMA. This FIFO buffer can be initialized by means of EP2CLR in the FCLR register.

Bit	7	6	5	4	3	2	1	
Bit Name	D7	D6	D5	D4	D3	D2	D1	
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	W	W	W	W	W	W	W	

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 2 transfer

Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	W	W	W	W	W	W	W

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 3 transfer

### 20.3.16 EP0o Receive Data Size Register (EPSZ0o)

EPSZ0o indicates the number of bytes received at endpoint 0 from the host.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R	Number of receive data for endpoint 0

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R	Number of received bytes for endpoint 1

### 20.3.18 Trigger Register (TRG)

TRG generates one-shot triggers to control the transfer sequence for each endpoint.

Bit	7	6	5	4	3	2	1	
Bit Name	—	EP3 PKTE	EP1 RDFN	EP2 PKTE	—	EP0s RDFN	EP0o RDFN	EP
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Un
R/W	—	W	W	W	—	W	W	

Bit	Bit Name	Initial Value	R/W	Description
7	—	Undefined	—	Reserved The write value should always be 0.
6	EP3 PKTE	Undefined	W	EP3 Packet Enable After one packet of data has been written to the endpoint 3 transmit FIFO buffer, the transmit is fixed by writing 1 to this bit.

3	—	Undefined	—	Reserved The write value should always be 0.
2	EP0s RDFN	Undefined	W	EP0s Read Complete Write 1 to this bit after data for the EP0s control FIFO has been read. Writing 1 to this bit enables transfer of data in the following data stage. A handshake is returned in response to transfer requests from the host in the data stage until 1 is written to this bit.
1	EP0o RDFN	Undefined	W	EP0o Read Complete Writing 1 to this bit after one packet of data has been read from the endpoint 0 transmit FIFO buffer initializes the FIFO buffer, enabling the next packet to be received.
0	EP0i PKTE	Undefined	W	EP0i Packet Enable After one packet of data has been written to the endpoint 0 transmit FIFO buffer, the transmit FIFO is fixed by writing 1 to this bit.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	EP3 DE	0	R	EP3 Data Present This bit is set when the endpoint 3 FIFO buffer contains valid data.
4	EP2 DE	0	R	EP2 Data Present This bit is set when the endpoint 2 FIFO buffer contains valid data.
3	—	0	R	Reserved
2	—	0	R	These bits are always read as 0.
1	—	0	R	
0	EP0i DE	0	R	EP0i Data Present This bit is set when the endpoint 0 FIFO buffer contains valid data.

Bit	Bit Name	Initial Value	R/W	Description
7	—	Undefined	—	Reserved The write value should always be 0.
6	EP3 CLR	Undefined	W	EP3 Clear Writing 1 to this bit initializes the endpoint 3 FIFO buffer.
5	EP1 CLR	Undefined	W	EP1 Clear Writing 1 to this bit initializes both sides of the endpoint 1 receive FIFO buffer.
4	EP2 CLR	Undefined	W	EP2 Clear Writing 1 to this bit initializes both sides of the endpoint 2 transmit FIFO buffer.
3	—	Undefined	—	Reserved
2	—	—	—	The write value should always be 0.
1	EP0o CLR	Undefined	W	EP0o Clear Writing 1 to this bit initializes the endpoint 0 FIFO buffer.
0	EP0i CLR	Undefined	W	EP0i Clear Writing 1 to this bit initializes the endpoint 0 FIFO buffer.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	—	0	R	
2	PULLUP_E	0	R/W	



(USBINTN1) is asserted again. However, if the data packet to be transmitted is less than 255 bytes, the EP2 packet enable bit is not set automatically, and so should be set by the CPU. The DMA transfer end interrupt.

As EP2-related interrupt requests to the CPU are automatically masked, interrupt requests should be masked as necessary in the interrupt enable register.

- Operating procedure
  1. Write of 1 to the EP2 DMAE bit in DMAR
  2. Set the DMAC to activate through USBIN
  3. Transfer count setting in the DMAC
  4. DMAC activation
  5. DMA transfer
  6. DMA transfer end interrupt generated

See section 20.8.3, DMA Transfer for Endp

---

- Endpoint interrupt requests to the CPU are automatically masked.
- Operating procedure:
    1. Write of 1 to the EP1 DMAE bit in DMA
    2. Set the DMAC to activate through USBIN
    3. Transfer count setting in the DMAC
    4. DMAC activation
    5. DMA transfer
    6. DMA transfer end interrupt generated
- See section 20.8.2, DMA Transfer for Endpoint
-

Bit Name	—	—	—	—	EP3STL	EP2STL	EP1STL
Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	EP3STL	0	R/W	EP3 Stall When this bit is set to 1, endpoint 3 is placed in stall state.
2	EP2STL	0	R/W	EP2 Stall When this bit is set to 1, endpoint 2 is placed in stall state.
1	EP1STL	0	R/W	EP1 Stall When this bit is set to 1, endpoint 1 is placed in stall state.
0	EP0STL	0	R/W	EP0 Stall When this bit is set to 1, endpoint 0 is placed in stall state.

Bit	Bit Name	Initial Value	R/W	Description
7	CNFV1	All 0	R	These bits store Configuration Setting value when they receive Set Configuration command. CNFV is updated when the SETC bit in IFR2 is set to 1.
6	CNFV0			
5	INTV1	All 0	R	These bits store Interface Setting value when receive Set Interface command. INTV is updated when the SETI bit in IFR2 is set to 1.
4	INTV0			
3	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.
2	ALTV2	0	R	These bits store Alternate Setting value when receive Set Interface command. ALTV2 to ALTV0 is updated when the SETI bit in IFR2 is set to 1.
1	ALTV1	0	R	
0	ALTV0	0	R	

### 20.3.24 Control Register (CTLR)

This register sets functions for bits ASCE, PWMD, RSME, and, PWUPS.

Bit	7	6	5	4	3	2	1
Bit Name	—	—	—	RWUPS	RSME	PWMD	ASCE
Initial Value	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W

				Feature request. This bit is set to 1 when remote wakeup command is enabled.
3	RSME	0	R/W	<p>Resume Enable</p> <p>This bit releases the suspend state (or execution from remote wakeup). When RSME is set to 1, remote wakeup request starts. If RSME is once set to 1, cleared to 0 again afterwards. In this case, the value of RSME must be kept for at least one clock period of the 12-MHz clock.</p>
2	PWMD	0	R/W	<p>Bus Power Mode</p> <p>This bit specifies the USB power mode. When this bit is set to 0, the self-power mode is selected for the USB module. When set to 1, the bus-power mode is selected.</p>
1	ASCE	0	R/W	<p>Automatic Stall Clear Enable</p> <p>Setting the ASCE bit to 1 automatically clears the stall setting bit (the EPxSTL (x = 1, 2, or 3) bit in EPSTR0 or EPSTR1) of the end point that has returned the stall handshake to the host. The automatic stall clear enable is common to the all end points. Thus, the individual control of the end point is not possible.</p> <p>When the ASCE bit is set to 0, the stall setting bit is not automatically cleared. This bit must be reset by the users. To enable this bit, make sure that the ASCE bit should be set to 1 before the EPxSTL (x = 1, 2, or 3) bit in EPSTR is set to 1.</p>
0	—	0	R	<p>Reserved</p> <p>This bit is always read as 0. The write value is always be 0.</p>

Bit	7	6	5	4	3	2	1	
Bit Name	D7	D6	D5	D4	D3	D2	D1	
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Un
R/W	W	W	W	W	W	W	W	

- EPIR00

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	D7 to D4	Undefined	W	Endpoint Number [Enable setting range] 0 to 3
3, 2	D3, D2	Undefined	W	Endpoint Configuration Number [Enable setting range] 0 or 1
1, 0	D1, D0	Undefined	W	Endpoint Interface Number [Enable setting range] 0 to 3

				2: Bulk 3: Interrupt
3	D3	Undefined	W	Endpoint Transmission Direction [Possible setting range] 0: Out 1: In
2 to 0	D2 to D0	Undefined	W	Reserved [Possible setting range] Fixed to 0.

- EPIR02

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	D7 to D1	Undefined	W	Endpoint Maximum Packet Size [Possible setting range] 0 to 64
0	D0	Undefined	W	Reserved [Possible setting range] Fixed to 0.

- EPIR03

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Reserved [Possible setting range] Fixed to 0.

described below.

Since each endpoint FIFO number is optimized by the exclusive software that corresponds to the transfer system, direction, and the maximum packet size, make sure to set the endpoint FIFO number to the data described in table 20.2.

1. The endpoint FIFO number 1 cannot designate other than the maximum packet size of 64 bytes, bulk transfer method, and out transfer direction.
2. endpoint number 0 and the endpoint FIFO number must have one-on-one relationship.
3. The maximum packet size for the endpoint FIFO number 0 is 8 bytes only.
4. The endpoint FIFO number 0 can specify only the maximum packet size and the data transfer direction. The rest should be all 0.
5. The maximum packet size for the endpoint FIFO numbers 1 and 2 is limited to 64 bytes only.
6. The maximum packet size for the endpoint FIFO numbers 3 is limited to 8 bytes.
7. The maximum number of endpoint information setting is ten.
8. Up to ten endpoint information setting should be made.
9. Write 0 to the endpoints not in use.

Table 20.2 shows the example of limitations for the maximum packet size, the transfer method, and the transfer direction.

**Table 20.2 Example of Limitations for Setting Values**

Endpoint FIFO Number	Maximum Packet Size	Transfer Method	Transfer Direction
0	8 bytes	Control	—
1	64 bytes	Bulk	Out
2	64 bytes	Bulk	In
3	8 bytes	Interrupt	In



N	EPIR[N]0	EPIR[N]1	EPIR[N]2	EPIR[N]3	EPIR[N]4
0	00	00	10	00	00
1	14	20	80	00	01
2	24	28	80	00	02
3	34	38	10	00	03
4	00	00	00	00	00
5	00	00	00	00	00
6	00	00	00	00	00
7	00	00	00	00	00
8	00	00	00	00	00
9	00	00	00	00	00

Configuration	Interface	Alternate Setting	Endpoint Number	Endpoint FIFO Number	...
—	—	—	0	0	C
1	0	0	1	1	B
			2	2	E
			3	3	Ir

Bit	Bit Name	Initial Value	R/W	Description
7	PTSTE	0	R/W	Pin Test Enable Enables the test control for the on-chip transceiver output pins (USD+ and USD-).
6 to 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
3	SUSPEND	0	R/W	On-Chip Transceiver Output Signal Setting
2	txenl	0	R/W	SUSPEND: Sets the (SUSPEND) signal of the transceiver.
1	txse0	0	R/W	
0	txdata	0	R/W	txenl: Sets the output enable (txenl) signal of the on-chip transceiver. txse0: Sets the Signal-ended 0 (txse0) signal of the on-chip transceiver. txdata: Sets the (txdata) signal of the on-chip transceiver.

1	1	1	X	X	Hi-Z	Hi-Z
---	---	---	---	---	------	------

[Legend]

X: Don't care.

—: Cannot be controlled. Indicates state in normal operation according to the USB operation and port settings.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value always be 0.
2	xver_data	—*	R	On-Chip Transceiver Input Signal Monitor
1	dpls	—*	R	xver_data: Monitors the differential input level (xver_data) signal of the on-chip transceiver.
0	dmns	—*	R	dpls: Monitors the USD+ (dpls) signal of the on-chip transceiver. dmns: Monitors the USD- (dmns) signal of the on-chip transceiver.

Note: \* Determined by the state of pins, VBUS, USD+, and USD-

1	0	1	1	0	1	1	0
1	0	1	1	1	X	1	1
1	1	1	0	0	0	0	0
1	1	1	0	1	0	0	1
1	1	1	1	0	0	1	0
1	1	1	1	1	0	1	1
1	X	0	X	X	0	1	1

Can be m  
when VBU

[Legend]

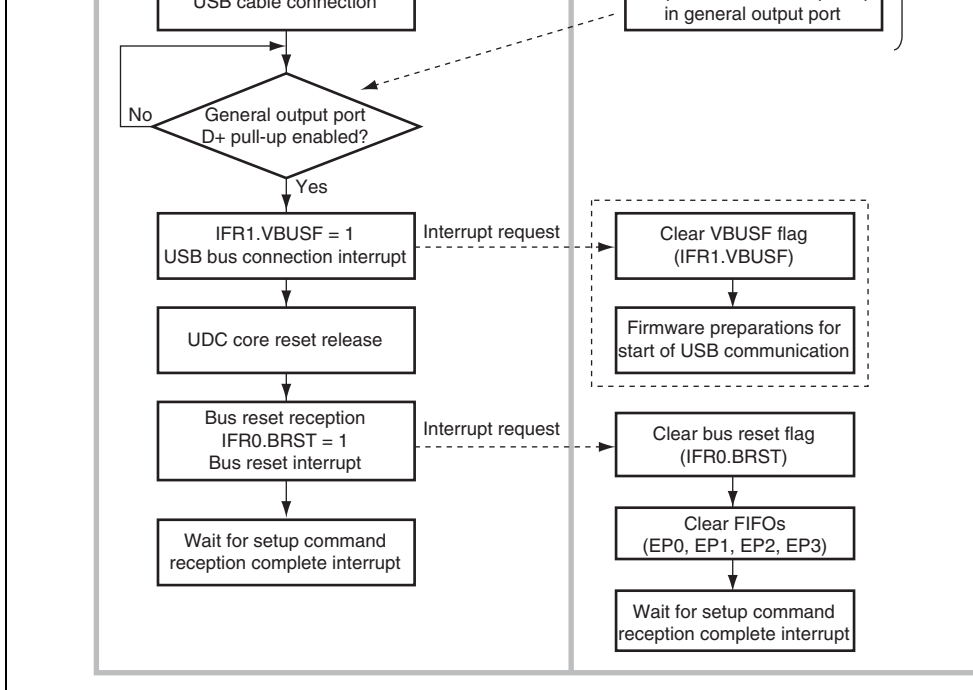
X: Don't care.

		transfer (EP0)		complete	USBINTN3			
	1		EP0i_TR*	EP0i transfer request	USBINTN2 or USBINTN3	x		x
	2		EP0o_TS*	EP0o receive complete	USBINTN2 or USBINTN3	x		x
	3		SETUP_TS*	Setup command receive complete	USBINTN2 or USBINTN3	x		x
	4	Bulk_in transfer (EP2)	EP2_EMPTY	EP2 FIFO empty	USBINTN2 or USBINTN3	x		US
	5		EP2_TR	EP2 transfer request	USBINTN2 or USBINTN3	x		x
	6	Bulk_out transfer (EP1)	EP1_FULL	EP1 FIFO Full	USBINTN2 or USBINTN3	x		US
	7	Status	BRST	Bus reset	USBINTN2 or USBINTN3	x		x
IFR1	0	Status	VBUSF	USB disconnection detection	USBINTN2 or USBINTN3	x		x
	1	Interrupt_in transfer (EP3)	EP3_TS	EP3 transfer complete	USBINTN2 or USBINTN3	x		x
	2		EP3_TR	EP3 transfer request	USBINTN2 or USBINTN3	x		x
	3	Status	VBUSMN	VBUS connection status	—	x		x
	4	—	Reserved	—	—	—		—
	5							
	6							
	7							

			USBINTN3, or RESUME			
5		SURSS	Suspend/resume status	—	×	×
6	—	Reserved	—	—	—	—
7						

Note: \* EP0 interrupts must be assigned to the same interrupt request signal.

- USBINTN0 signal  
DMAC start interrupt signal only EP1. See section 20.8, DMA Transfer.
- USBINTN1 signal  
DMAC start interrupt signal only EP2. See section 20.8, DMA Transfer.
- USBINTN2 signal  
The USBINTN2 signal requests interrupt sources for which the corresponding bits in select registers 0 to 2 (ISR0 to ISR2) are cleared to 0. The USBINTN2 is driven low when the corresponding bit in the interrupt flag register is set to 1.
- USBINTN3 signal  
The USBINTN3 signal requests interrupt sources for which the corresponding bits in select registers 0 to 2 (ISR0 to ISR2) are cleared to 0. The USBINTN3 is driven low when the corresponding bit in the interrupt flag register is set to 1.
- RESUME signal  
The RESUME signal is a resume interrupt signal for canceling software standby mode and deep software standby mode. The RESUME signal is driven low at the transition to the suspend state for canceling software standby mode and deep software standby mode.

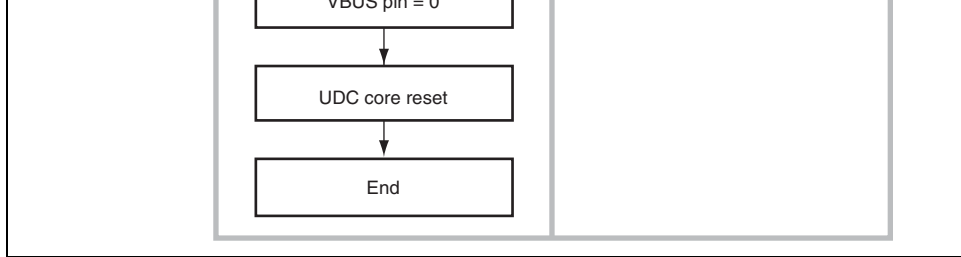


**Figure 20.2 Cable Connection Operation**

The above flowchart shows the operation in the case of in section 20.9, Example of USB Circuitry.

In applications that do not require USB cable connection to be detected, processing by the bus connection interrupt is not necessary. Preparations should be made with the bus-reset interrupt.





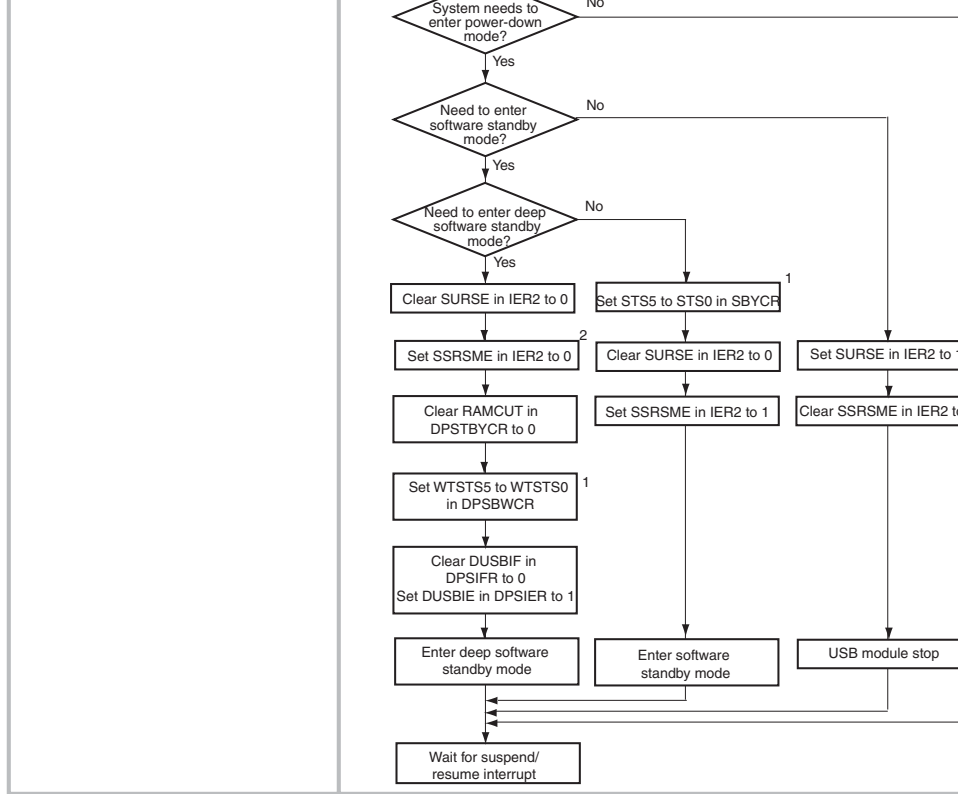
**Figure 20.3 Cable Disconnection Operation**

The above flowchart shows the operation in section 20.9, Example of USB External Circuit.

### **20.5.3 Suspend and Resume Operations**

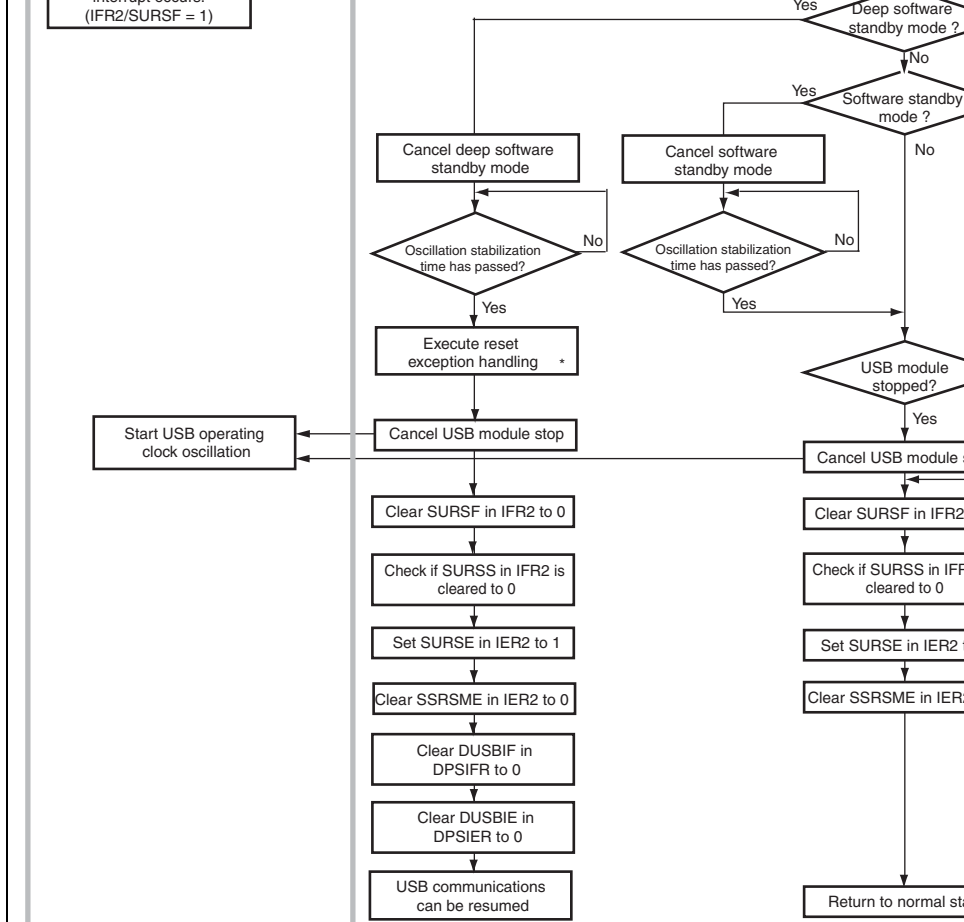
#### **(1) Suspend Operation**

If the USB bus enters the suspend state from the non-suspend state, perform the operation shown in figure 20.4.

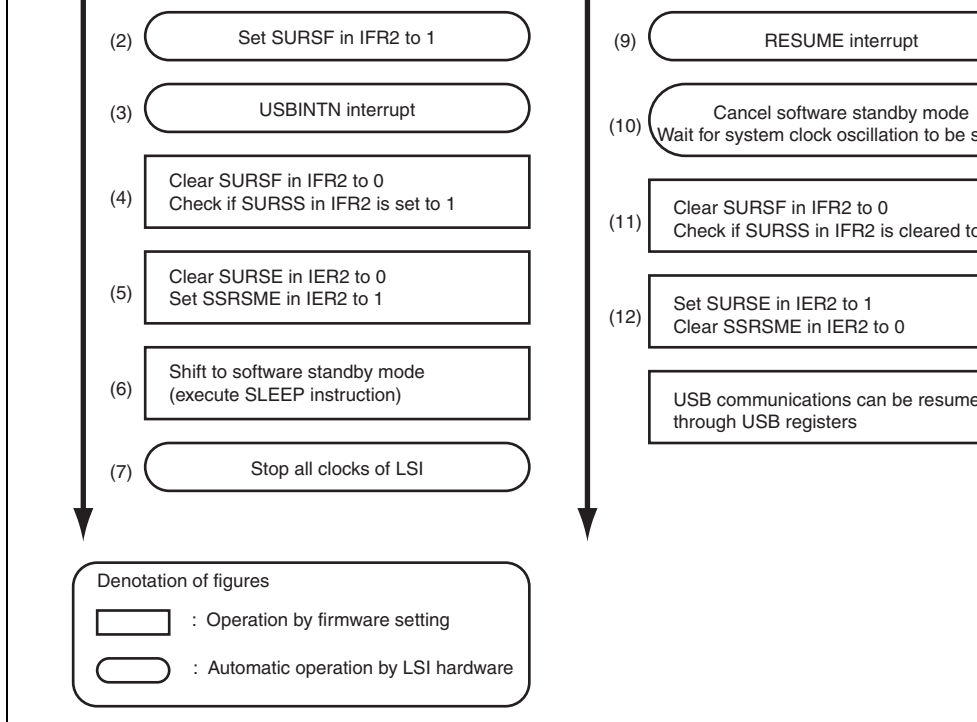


- Notes:
1. For details, see section 28, Power-Down Modes.
  2. When the USB enters deep software standby mode, the sources to cancel software standby mode may be conflicted. In this figure, the operation to cancel software standby mode is not performed. For details, see section 28.12, Usage Notes.

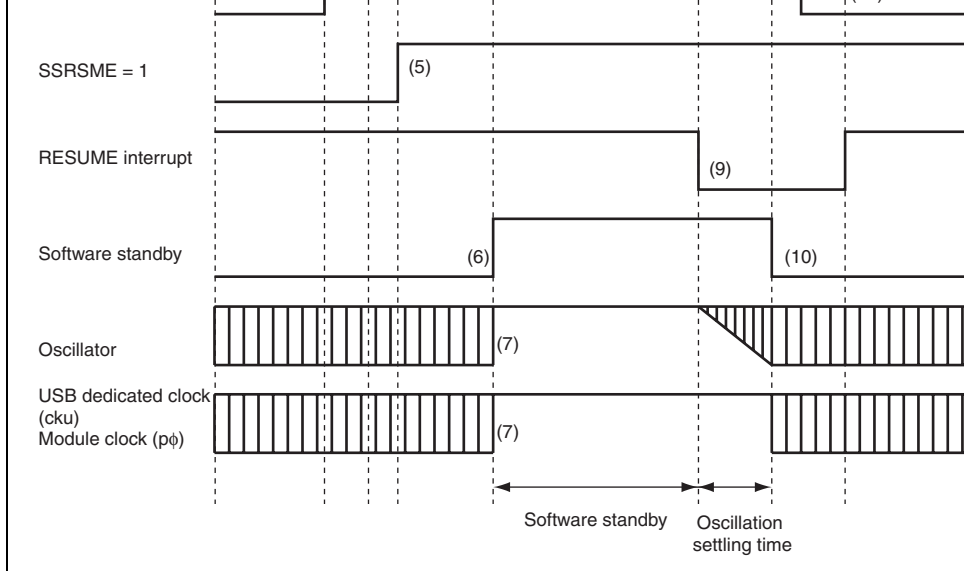
**Figure 20.4 Suspend Operation**



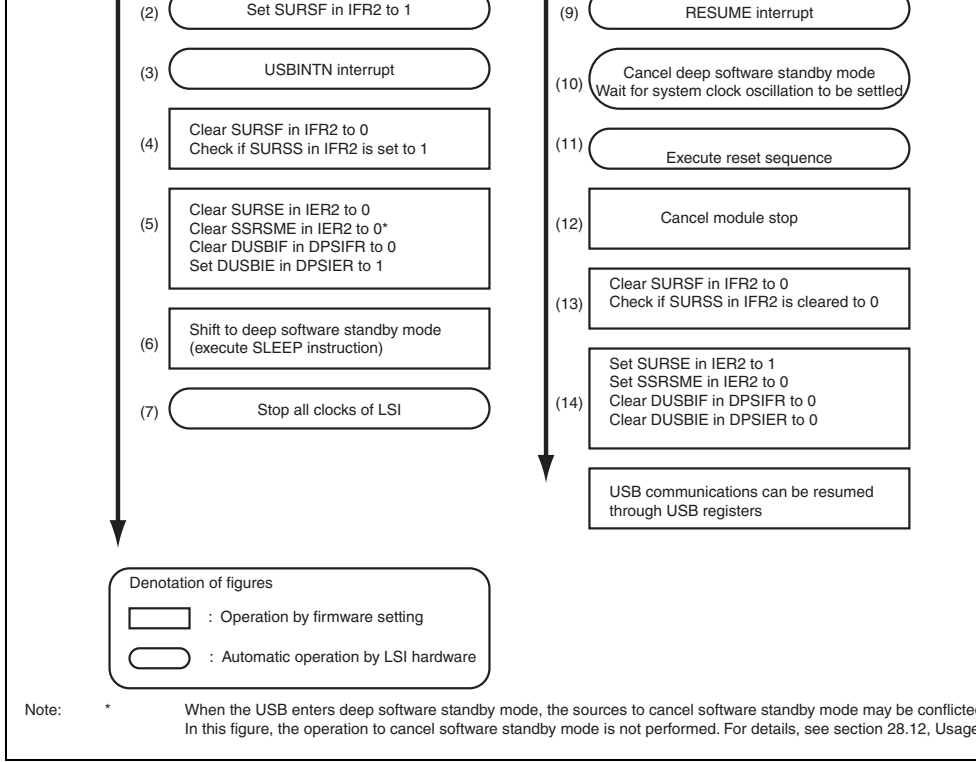
**Figure 20.5 Resume Operation from Up-Stream**



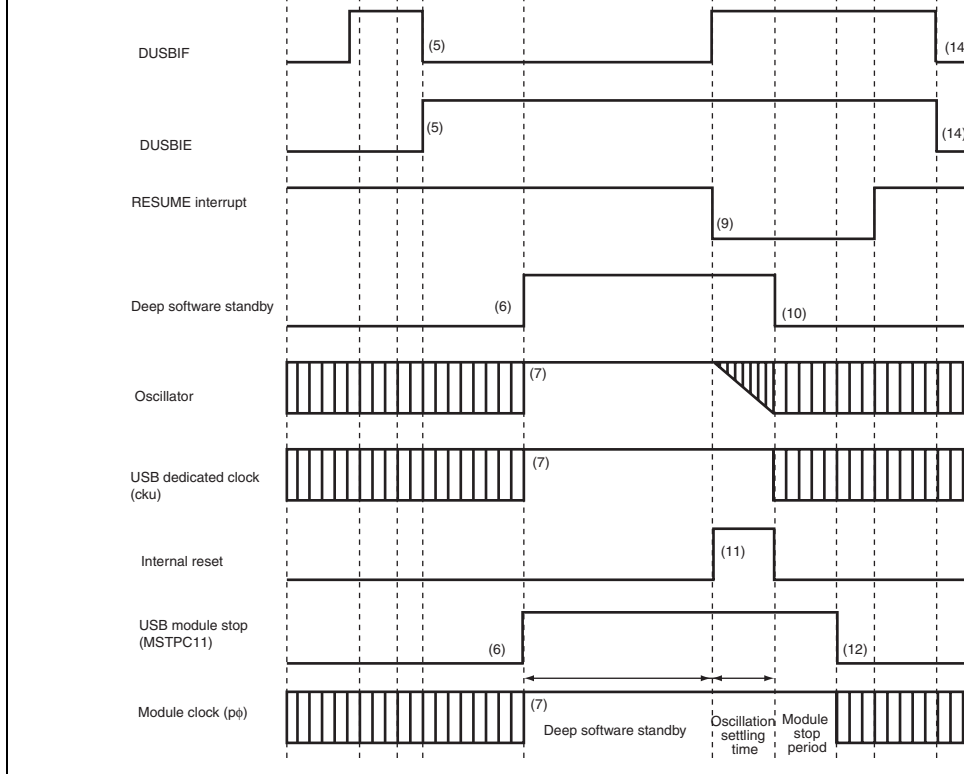
**Figure 20.6 Flow of Transition to and Canceling Software Standby Mode**



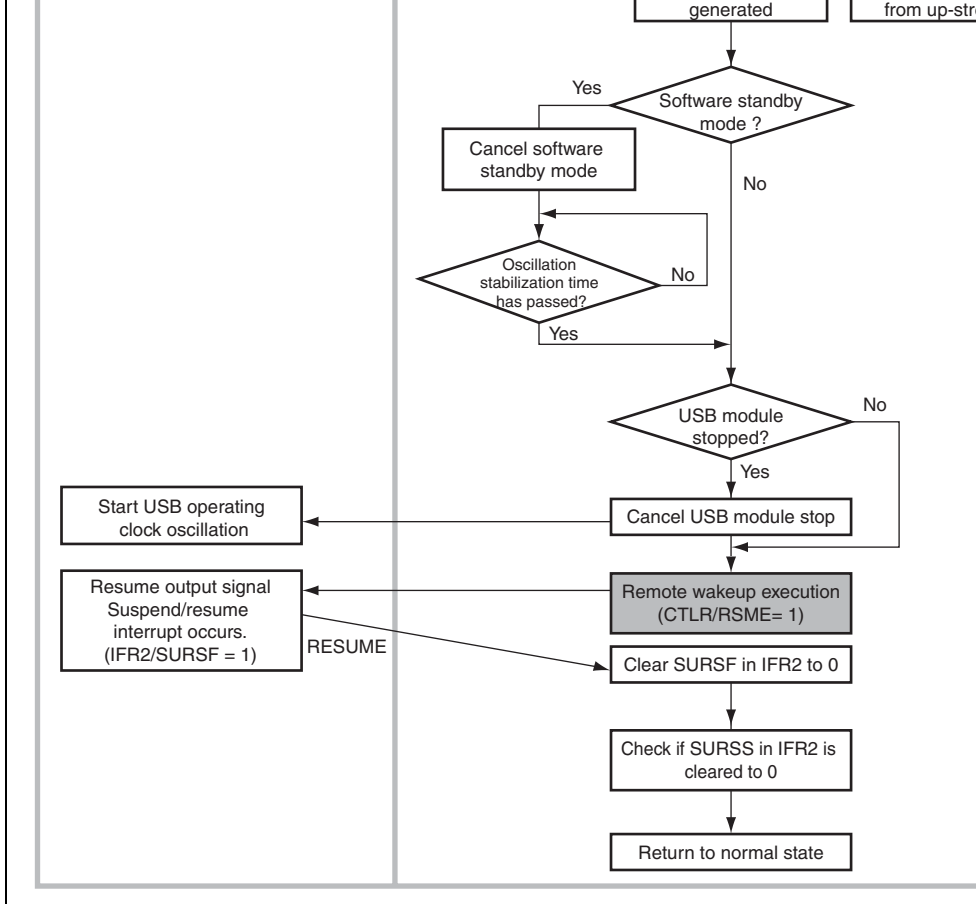
**Figure 20.7 Timing of Transition to and Canceling Software Standby Mode**



**Figure 20.8 Flow of Transition to and Canceling Deep Software Standby Mode**

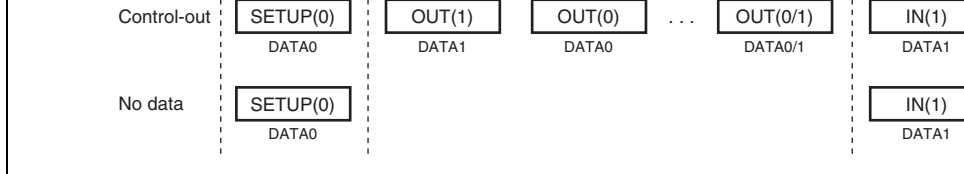


**Figure 20.9 Timing of Transition to and Canceling Deep Software Standby M**

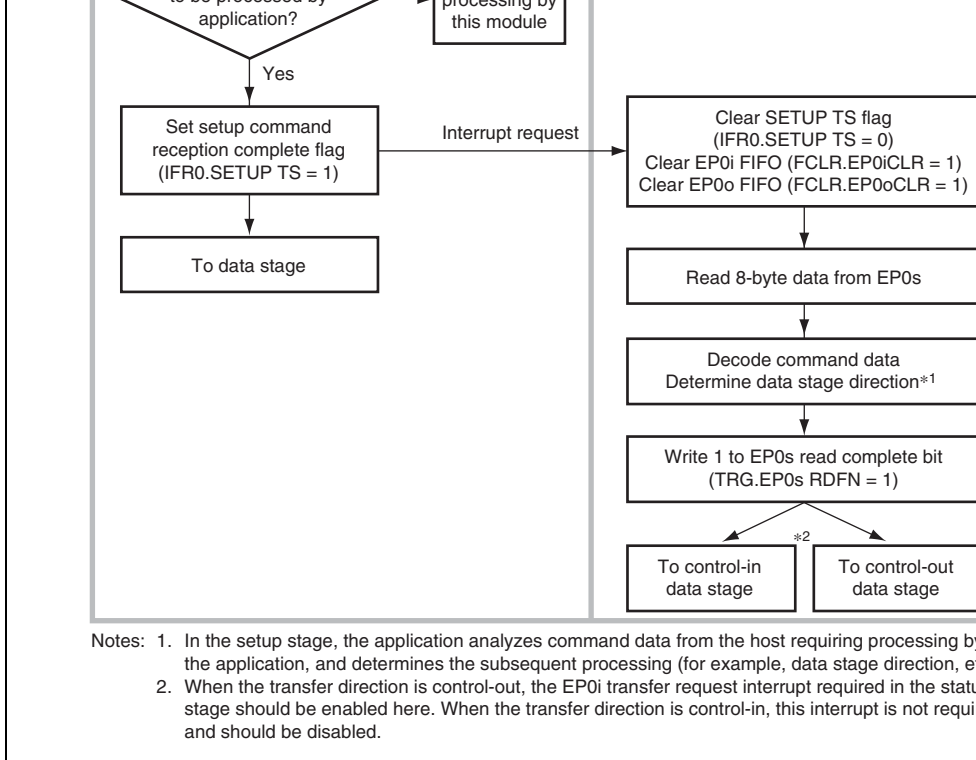


**Figure 20.10 Remote-Wakeup**

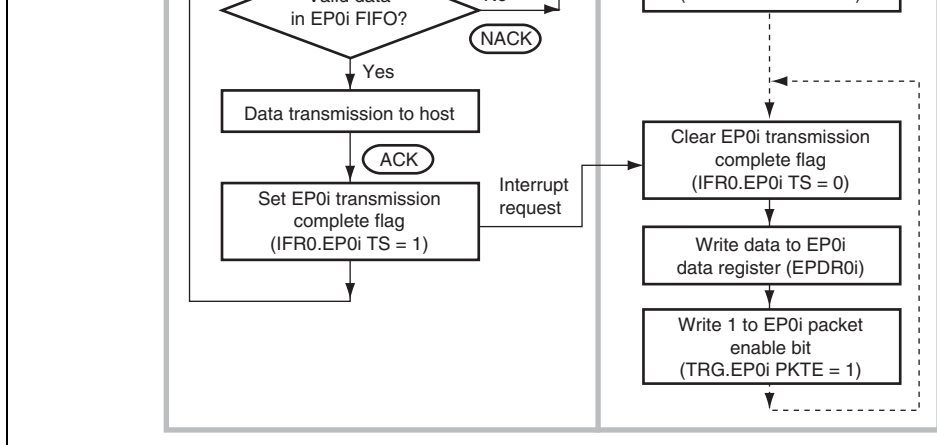




**Figure 20.11 Transfer Stages in Control Transfer**



**Figure 20.12 Setup Stage Operation**

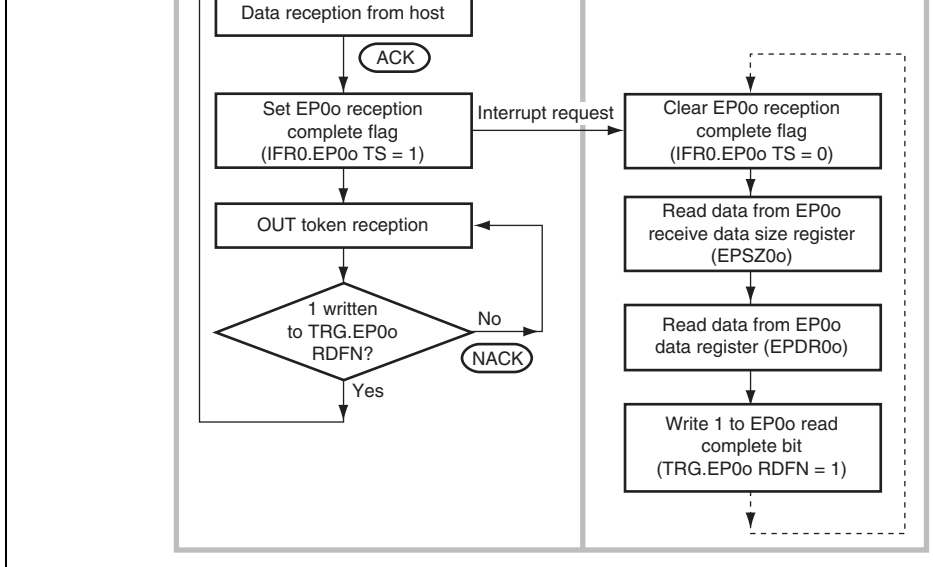


**Figure 20.13 Data Stage (Control-In) Operation**

The application first analyzes command data from the host in the setup stage, and determines the subsequent data stage direction. If the result of command data analysis is that the data stage is for data transfer, one packet of data to be sent to the host is written to the FIFO. If there is more data to be sent, this data is written to the FIFO after the data written first has been sent to the host (the bit in IFR0 = 1).

The end of the data stage is identified when the host transmits an OUT token and the status flag is entered.

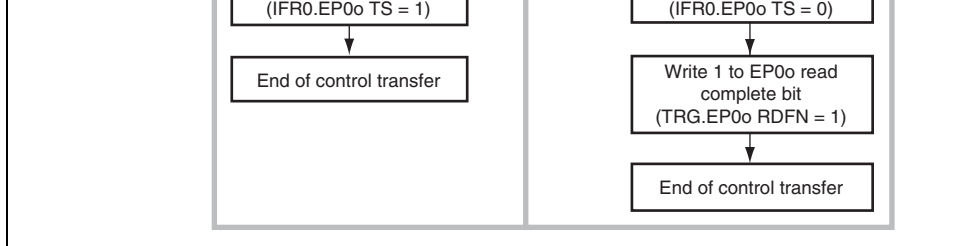
**Note:** If the size of the data transmitted by the function is smaller than the data size received from the host, the function indicates the end of the data stage by returning to the host a packet shorter than the maximum packet size. If the size of the data transmitted by the function is an integral multiple of the maximum packet size, the function indicates the end of the data stage by transmitting a zero-length packet.



**Figure 20.14 Data Stage (Control-Out) Operation**

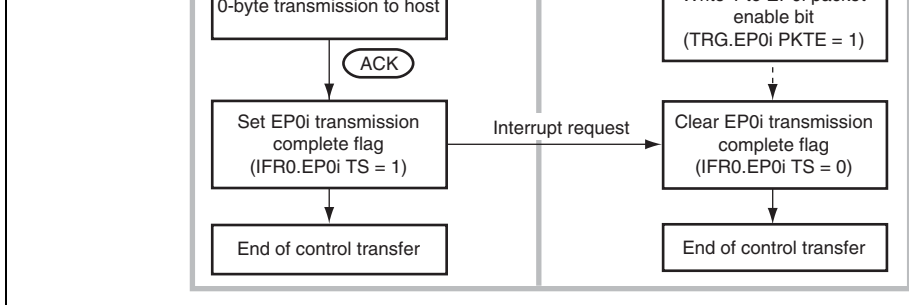
The application first analyzes command data from the host in the setup stage, and determines subsequent data stage direction. If the result of command data analysis is that the data stage is for data transfer, the application waits for data from the host, and after data is received (EP0oTS bit IFR0 = 1), reads data from the FIFO. Next, the application writes 1 to the EP0o read complete bit, empties the receive FIFO, and waits for reception of the next data.

The end of the data stage is identified when the host transmits an IN token and the status is entered.



**Figure 20.15 Status Stage (Control-In) Operation**

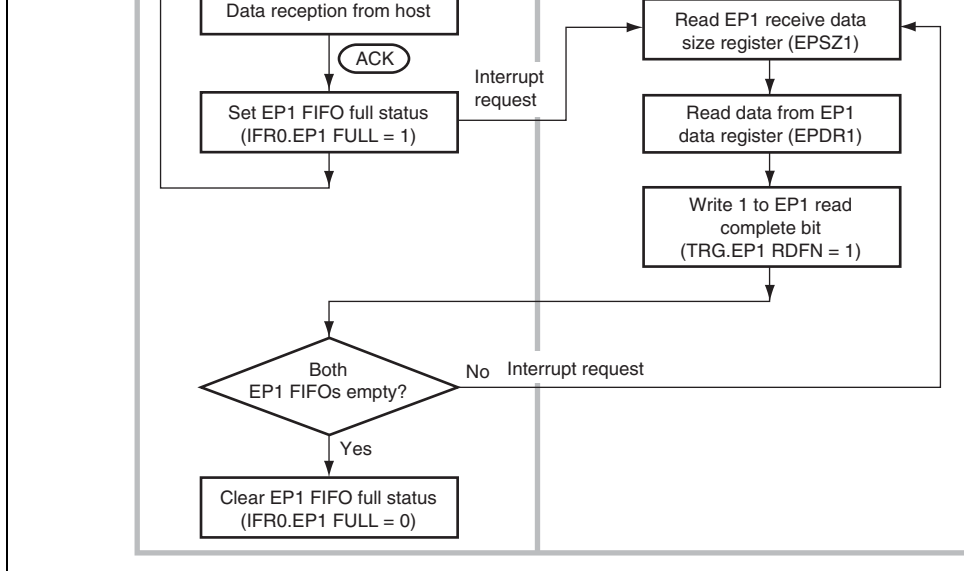
The control-in status stage starts with an OUT token from the host. The application receives byte data from the host, and ends control transfer.



**Figure 20.16 Status Stage (Control-Out) Operation**

The control-out status stage starts with an IN token from the host. When an IN-token is received at the start of the status stage, there is not yet any data in the EP0i FIFO, and so an EP0i transmission request interrupt is generated. The application recognizes from this interrupt that the status stage has started. Next, in order to transmit 0-byte data to the host, 1 is written to the EP0i packet enable bit but no data is written to the EP0i FIFO. As a result, the next IN token causes 0-byte data to be transmitted to the host, and control transfer ends.

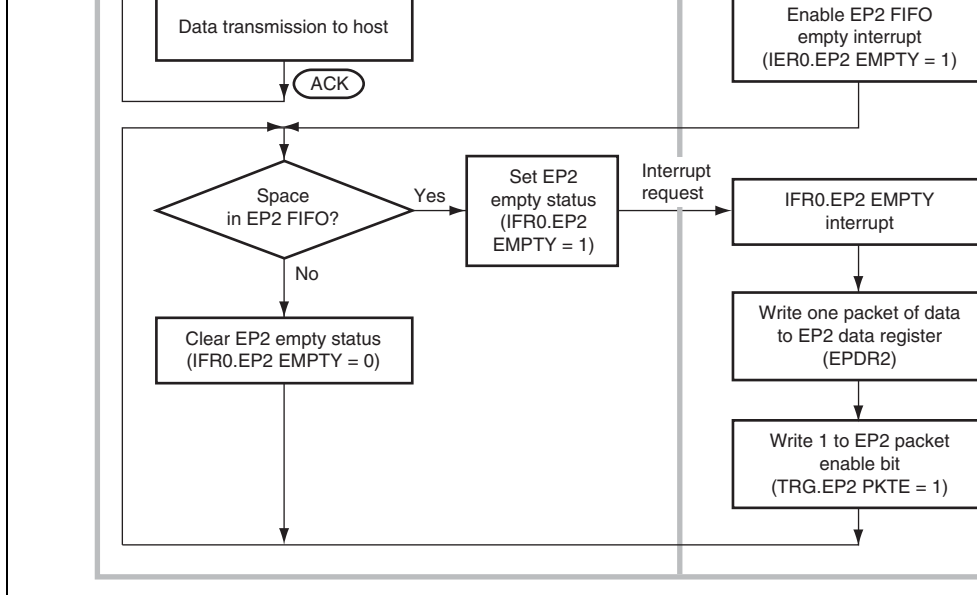
After the application has finished all processing relating to the data stage, 1 should be written to the EP0i packet enable bit.



**Figure 20.17 EP1 Bulk-Out Transfer Operation**

EP1 has two 64-byte FIFOs, but the user can receive data and read receive data without aware of this dual-FIFO configuration.

When one FIFO is full after reception is completed, the EP1FULL bit in IFR0 is set. After receive operation into one of the FIFOs when both FIFOs are empty, the other FIFO is empty so the next packet can be received immediately. When both FIFOs are full, NACK is returned to the host automatically. When reading of the receive data is completed following data received is written to the EP1RDFN bit in TRG. This operation empties the FIFO that has just been read and makes it ready to receive the next packet.

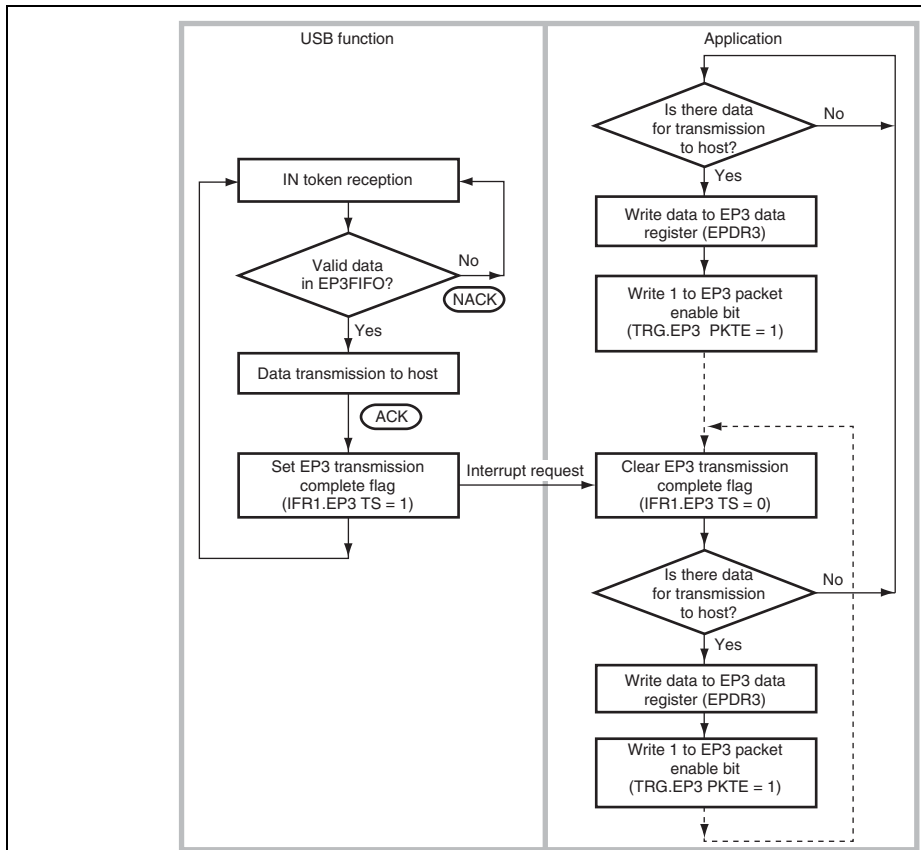


**Figure 20.18 EP2 Bulk-In Transfer Operation**

EP2 has two 64-byte FIFOs, but the user can transmit data and write transmit data without being aware of this dual-FIFO configuration. However, one data write is performed for one FIFO. For example, even if both FIFOs are empty, it is not possible to perform EP2PKTE at one time. To write 128 bytes of data consecutively, EP2PKTE must be performed for each 64-byte write.

When performing bulk-in transfer, as there is no valid data in the FIFOs on reception of the IN token, an EP2TR bit interrupt in IFR0 is requested. With this interrupt, 1 is written to the EP2EMPTY bit in IER0, and the EP2 FIFO empty interrupt is enabled. At first, both EP2 FIFOs are empty, and so an EP2 FIFO empty interrupt is generated immediately.





Note: This flowchart shows just one example of interrupt transfer processing. Other possibilities include an operation flow in which, if there is data to be transferred, the EP3 DE bit in the data status register is referenced to confirm that the FIFO is empty, and then data is written to the FIFO.

**Figure 20.19 Operation of EP3 Interrupt-In Transfer**

<b>Decoding not Necessary on Application Side</b>	<b>Decoding Necessary on Application Side</b>
Clear Feature	Get Descriptor
Get Configuration	Class/Vendor command
Get Interface	Set Descriptor
Get Status	Sync Frame
Set Address	
Set Configuration	
Set Feature	
Set Interface	

If decoding is not necessary on the application side, command decoding and data stage and status stage processing are performed automatically. No processing is necessary by the user. An interrupt is not generated in this case.

If decoding is necessary on the application side, this module stores the command in the EPDRO FIFO. After reception is completed successfully, the IFR0/SETUP TS flag is set and an interrupt request is generated. In the interrupt routine, eight bytes of data must be read from the EPDRO register (EPDROs) and decoded by firmware. The necessary data stage and status stage processing should then be carried out according to the result of the decoding operation.

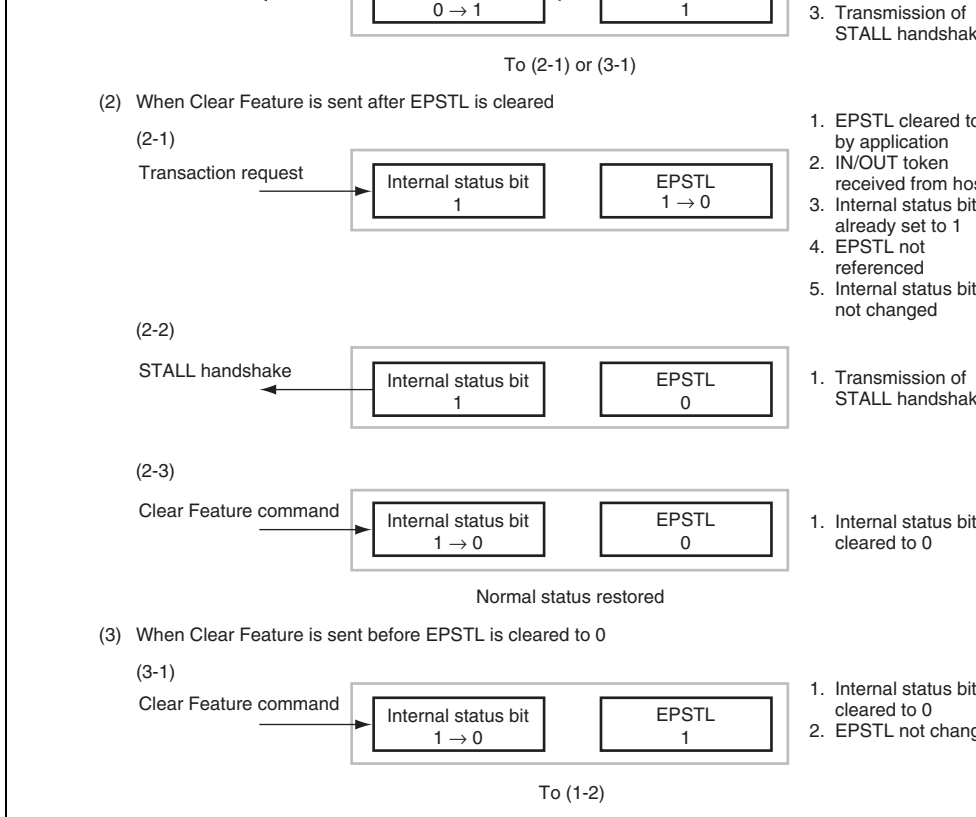
The USB function module has internal status bits that hold the status (stall or non-stall) of each endpoint. When a transaction is sent from the host, the module references these internal status bits and determines whether to return a stall to the host. These bits cannot be cleared by the application; they must be cleared with a Clear Feature command from the host.

However, the internal status bit for EP0 is automatically cleared only when the setup command is received.

## 20.7.2 Forcible Stall by Application

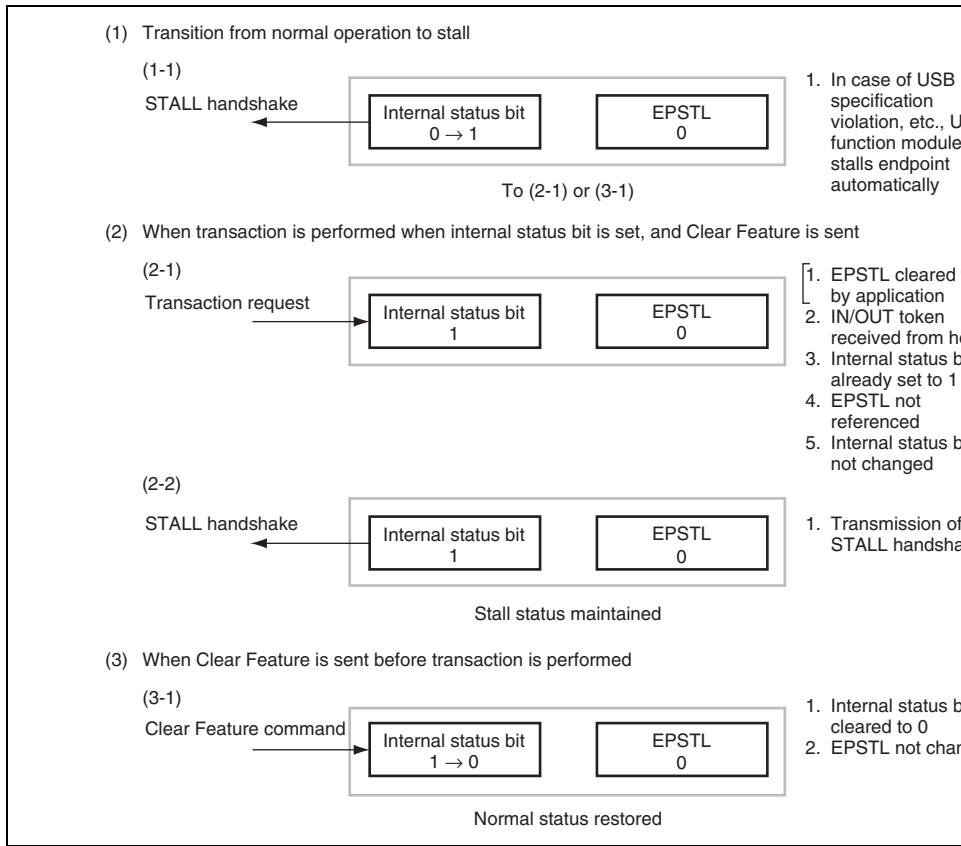
The application uses the EPSTL register to issue a stall request for the USB function module. When the application wishes to stall a specific endpoint, it sets the corresponding bit in the EPSTL register (1-1 in figure 20.20). The internal status bits are not changed at this time. When a transaction is received from the host for the endpoint for which the EPSTL bit was set, the USB function module references the internal status bit, and if this is not set, references the corresponding bit in the EPSTL register (1-2 in figure 20.20). If the corresponding bit in EPSTL is set, the USB function module sets the internal status bit and returns a stall handshake to the host (1-3 in figure 20.20). If the corresponding bit in EPSTL is not set, the internal status bit is not changed and the transaction is accepted.

Once an internal status bit is set, it remains set until cleared by a Clear Feature command from the host, without regard to the EPSTL register. Even after a bit is cleared by the Clear Feature command (3-1 in figure 20.20), the USB function module continues to return a stall handshake while the bit in EPSTL is set, since the internal status bit is set each time a transaction is received for the corresponding endpoint (1-2 in figure 20.20). To clear a stall, therefore, it is necessary for the corresponding bit in EPSTL to be cleared by the application, and also for the internal status bit to be cleared with a Clear Feature command (2-1, 2-2, and 2-3 in figure 20.20).



**Figure 20.20 Forcible Stall by Application**

the internal status bit must be cleared with a Clear Feature command (3-1 in figure 20.21) by the application, EPSTL should also be cleared (2-1 in figure 20.21).



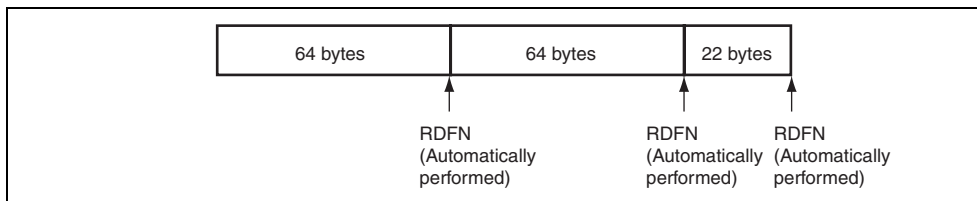
**Figure 20.21 Automatic Stall by USB Function Module**

If the DMA transfer is enabled by setting the DFEN bit in the DMA transfer setting register to 1, zero-length data reception at endpoint 1 is ignored. When the DMA transfer is enabled, the RDFN bit for EP1 and PKTE bit for EP2 do not need to be set to 1 in TRG (note that the PKTE bit must be set to 1 when the transfer data is less than the maximum number of bytes). When the data received at EP1 is read, the FIFO automatically enters the EMPTY state. When the maximum number of bytes (64 bytes) are written to the EP2 FIFO, the FIFO automatically enters the FULL state, and the data in the FIFO can be transmitted (see figures 20.22 and 20.23).

### 20.8.2 DMA Transfer for Endpoint 1

When the data received at EP1 is transferred by the DMA, the USB function module automatically performs the same processing as writing 1 to the RDFN bit in TRG if the currently selected data FIFO becomes empty. Accordingly, in DMA transfer, do not write 1 to the RDFN bit for EPI in TRG. If the user writes 1 to the RDFN bit in DMA transfer, correct operation cannot be guaranteed.

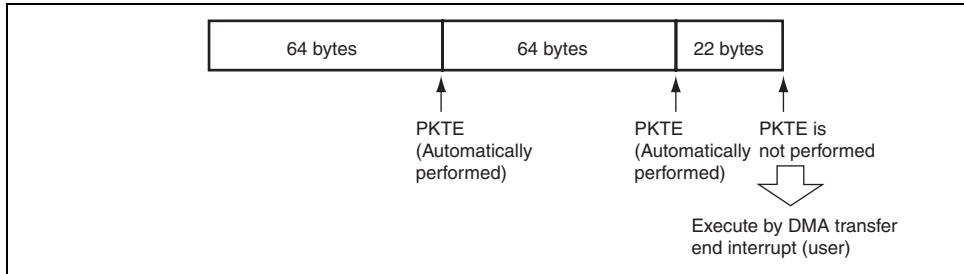
Figure 20.22 shows an example of receiving 150 bytes of data from the host. In this case, the internal processing which is the same as writing 1 to the RDFN bit in TRG is automatically performed three times. This internal processing is performed when the currently selected data FIFO becomes empty. Accordingly, this processing is automatically performed both when 64-byte data is received and when data less than 64 bytes is sent.



**Figure 20.22 RDFN Bit Operation for EP1**

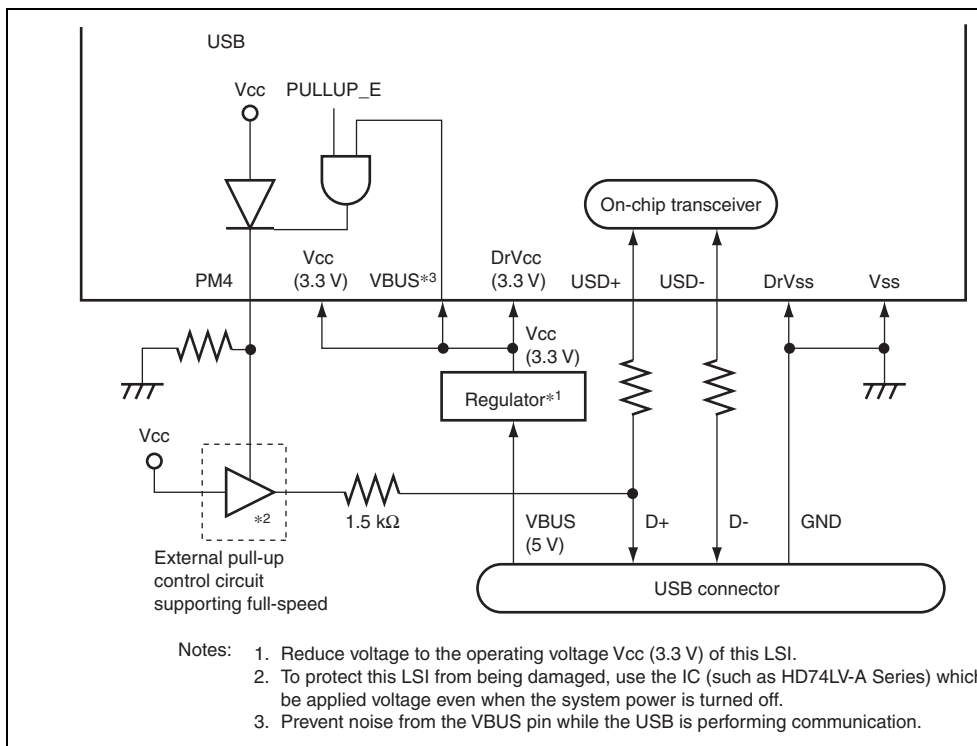
processing which is the same as writing 1 to the PKTE bit in TRG is automatically performed twice. This internal processing is performed when the currently selected data FIFO becomes empty. Accordingly, this processing is automatically performed only when 64-byte data is sent.

When the last 22 bytes are sent, the internal processing for writing 1 to the PKTE bit is not performed, and the user must write 1 to the PKTE bit by software. In this case, the application must write no more data to transfer but the USB function module continues to output DMA requests as long as the FIFO has an empty space. When all data has been transferred, write 0 to the EP2DMAE bit in DMAR to cancel DMA requests for EP2.



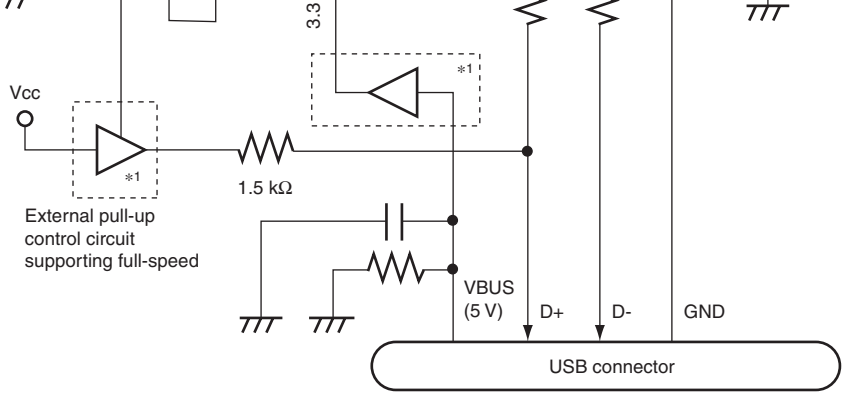
**Figure 20.23 PKTE Bit Operation for EP2**

connection/disconnection is necessary. The power supply signal (VBUS) in the USB is used for this purpose. However, if the cable is connected to the USB host/hub when the function (system installing this LSI) power is off, a voltage (5 V) will be applied from host/hub. Therefore, an IC (such as an HD74LV1G08A or 2G08A) that allows voltage application when the system power is off should be connected externally.



**Figure 20.24 Example of Circuitry in Bus Power Mode**





- Notes:
1. To protect this LSI from being damaged, use the IC (such as HD74LV-A Series) which can be applied voltage even when the system power is turned off.
  2. Prevent noise from the VBUS pin while the USB is performing communication.

**Figure 20.25 Example of Circuitry in Self Power Mode**

2. EPDR0s must always be read in 8-byte units. If the read is terminated at a midpoint, the data received at the next setup cannot be read correctly.

### **20.10.2 Clearing the FIFO**

If a USB cable is disconnected during data transfer, the data being received or transmitted remain in the FIFO. When disconnecting a USB cable, clear the FIFO.

While a FIFO is transferring data, it must not be cleared.

### **20.10.3 Overreading and Overwriting the Data Registers**

Note the following when reading or writing to a data register of this module.

#### **(1) Receive data registers**

The receive data registers must not be read exceeding the valid amount of receive data, the number of bytes indicated by the receive data size register. Even for EPDR1 which has double FIFO buffers, the maximum data to be read at one time is 64 bytes. After the data is read from the current valid FIFO buffer, be sure to write 1 to EP1RDFN in TRG, which switches the valid FIFO buffer, updates the receive data size to the new number of bytes, and enables the next data to be received.

#### **(2) Transmit data registers**

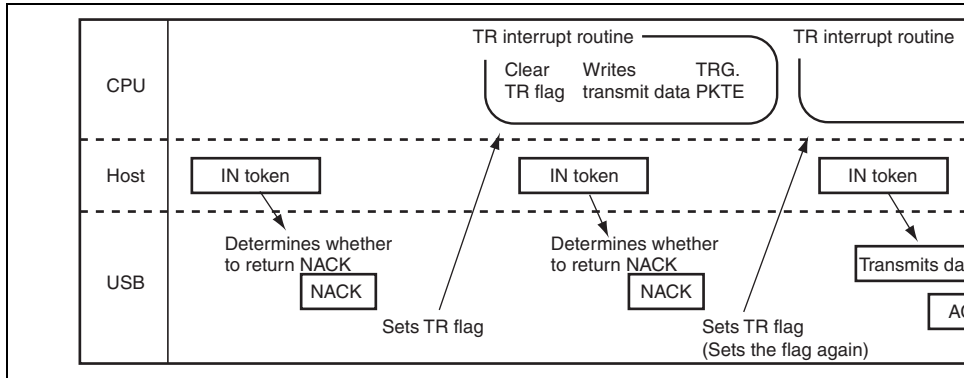
The transmit data registers must not be written to exceeding the maximum packet size. Even for EPDR2 which has double FIFO buffers, write data within the maximum packet size at one time. After the data is written, write 1 to PKTE in TRG to switch the valid buffer and enable the next data to be written. Data must not be continuously written to the two FIFO buffers.

## 20.10.6 Notes on TR Interrupt

Note the following when using the transfer request interrupt (TR interrupt) for IN transfer on EP2, or EP3.

The TR interrupt flag is set if the FIFO for the target EP has no data when the IN token is received from the USB host. However, at the timing shown in figure 20.26, multiple TR interrupts occur successively. Take appropriate measures against malfunction in such a case.

Note: This module determines whether to return NAKC if the FIFO of the target EP has no data when receiving the IN token, but the TR interrupt flag is set after a NAKC handshake is completed. If the next IN token is sent before PKTEND of TRG is written to, the TR interrupt flag is set again.



**Figure 20.26 TR Interrupt Flag Set Timing**

---

### 20.10.8 Notes on Deep Software Standby Mode when USB is Used

1. Unlike software standby mode, deep standby software mode is canceled from the reset. For details, see section 28.8, Deep Software Standby Mode.
2. If the RAMCUT bit is set to 1 when the USB enters deep software standby mode, the states of the USB cannot be retained. When USB is used, set the RAMCUT bit to 1, and then make the USB enter deep software standby mode.
3. Set the USB module stop (MSTPC11) bit to 0 after canceling deep software standby mode.
4. If the DUSBIE bit is set to 0 when the USB enters deep software standby mode, software standby mode cannot be canceled through USB RESUME interrupt. Set the DUSBIE bit to 1 and then, make the USB enter deep software standby mode.

## 21.1 Features

- Continuous transmission/reception

Since the shift register, transmit data register, and receive data register are independent of 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 the preparations are completed

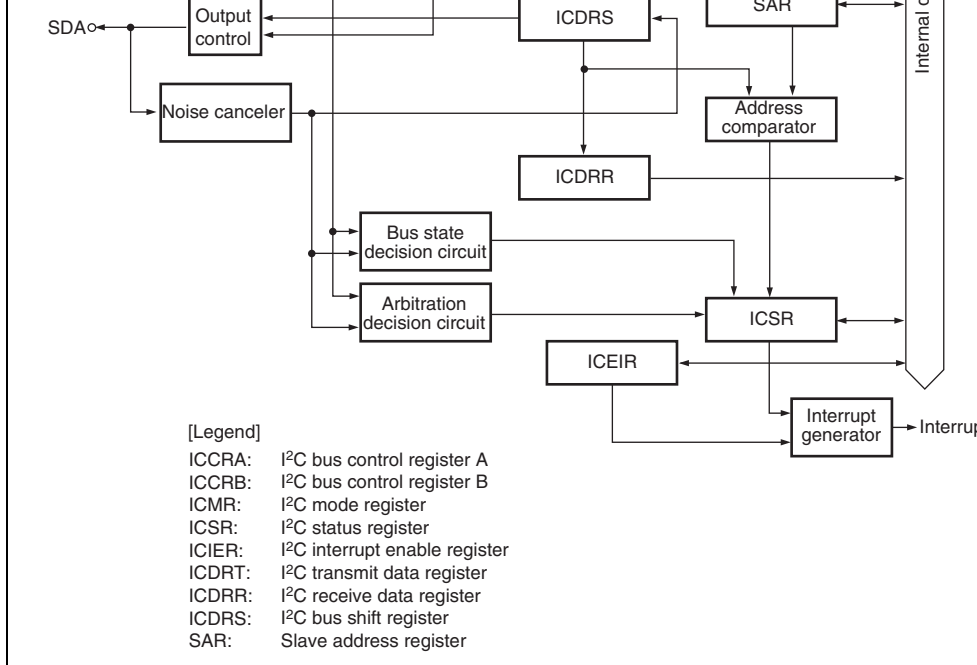
- Six interrupt sources

Transmit-data-empty (including slave-address match), transmit-end, receive-data-full (including slave-address match), arbitration lost, NACK detection, and stop condition detection

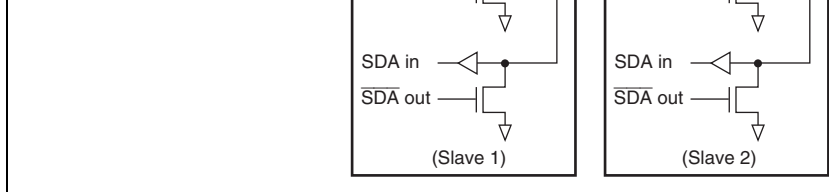
- Direct bus drive

Two pins, the SCL and SDA pins function as NMOS open-drain outputs.

- Module stop state can be set.



**Figure 21.1 Block Diagram of I<sup>2</sup>C Bus Interface 2**



**Figure 21.2 Connections to the External Circuit by the I/O Pins**

## 21.2 Input/Output Pins

Table 21.1 shows the pin configuration of the I<sup>2</sup>C bus interface 2.

**Table 21.1 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 the manual.

- 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)



Bit	Bit Name	Initial Value	R/W	Description
7	ICE	0	R/W	I <sup>2</sup> C Bus Interface Enable 0: This module is halted 1: This bit is enabled for transfer operations (SDA pins are bus drive state)
6	RCVD	0	R/W	Reception Disable This bit enables or disables the next operation. TRS is 0 and ICDRR is read. 0: Enables next reception 1: Disables next reception
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	Transmit/Receive Select 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 should be made between transfer frames. Operating modes are described below according to MST and TRS combination. 00: Slave receive mode 01: Slave transmit mode 10: Master receive mode 11: Master transmit mode
3	CKS3	0	R/W	Transfer Clock Select 3 to 0
2	CKS2	0	R/W	These bits are valid only in master mode. Mode setting according to the required transfer rate. For details on the transfer rate, see table 21.2.
1	CKS1	0	R/W	
0	CKS0	0	R/W	

			1	P $\phi$ /100	80.0 kHz	100 kHz	200 kHz	250 kHz	330 kHz	
		1	0	P $\phi$ /112	71.4 kHz	89.3 kHz	179 kHz	223 kHz	295 kHz	
			1	P $\phi$ /128	62.5 kHz	78.1 kHz	156 kHz	195 kHz	258 kHz	
1	0	0	0	P $\phi$ /56	143 kHz	179 kHz	357 kHz	446 kHz	589 kHz	
			1	P $\phi$ /80	100 kHz	125 kHz	250 kHz	313 kHz	413 kHz	
			1	0	P $\phi$ /96	83.3 kHz	104 kHz	208 kHz	260 kHz	344 kHz
			1	P $\phi$ /128	62.5 kHz	78.1 kHz	156 kHz	195 kHz	258 kHz	
	1	0	0	P $\phi$ /336	23.8 kHz	29.8 kHz	59.5 kHz	74.4 kHz	98.2 kHz	
			1	P $\phi$ /200	40.0 kHz	50.0 kHz	100 kHz	125 kHz	165 kHz	
			1	0	P $\phi$ /224	35.7 kHz	44.6 kHz	89.3 kHz	112 kHz	147 kHz
			1	P $\phi$ /256	31.3 kHz	39.1 kHz	78.1 kHz	97.7 kHz	129 kHz	

### 21.3.2 I<sup>2</sup>C Bus Control Register B (ICCRB)

ICCRB issues start/stop condition, manipulates the SDA pin, monitors the SCL pin, and resets in the I<sup>2</sup>C control module.

Bit	7	6	5	4	3	2	1
Bit Name	BBSY	SCP	SDAO	—	SCLO	—	IICRST
Initial Value	0	1	1	1	1	1	0
R/W	R/W	R/W	R	R/W	R	—	R/W

				start or stop condition, use the MOV instruction.
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 BBSY. A re-transmit start condition is issued in the same way. To issue a stop condition, write 0 to BBSY and 0 to BBSY. This bit is always read as 1. If 1 is written, the data is 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.</p> <p>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 1.</p>
1	IICRST	0	R/W	<p>IIC Control Module Reset</p> <p>This bit reset the IIC control module except the IIC registers. If hang-up occurs because of communication failure during I<sup>2</sup>C operation, by setting this bit to 1, the IIC control module can be reset without setting port I/O registers.</p>
0	—	1	—	<p>Reserved</p> <p>This bit is always read as 1.</p>

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 set to 1, after the falling of the clock for the last data transfer, the low period is extended for two transfer clock periods. When this bit is cleared to 0, data and the acknowledge bit are transferred consecutively with no wait insertion. 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 and the MOV instruction should be used. 0: When writing, the values of BC2 to BC0 are ignored. 1: When reading, 1 is always read. When writing, the settings of BC2 to BC0 are ignored.

001: 2  
 010: 3  
 011: 4  
 100: 5  
 101: 6  
 110: 7  
 111: 8

### 21.3.4 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
Bit Name	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R

This bit enables or disables the transmit end interrupt (TEI) request at the rising of the ninth clock when the TDRE bit in ICSR is set to 1. The TEI request can be canceled by clearing the TEND bit or the TEIE bit.

0: Transmit end interrupt (TEI) request is disabled  
 1: Transmit end interrupt (TEI) request is enabled

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          1: Receive data full interrupt (RXI) request is enabled</p>
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.</p> <p>0: NACK receive interrupt (NAKI) request is disabled          1: NACK receive interrupt (NAKI) request is enabled</p>

suspended.

---

1	ACKBR	0	R	Receive Acknowledge In transmit mode, this bit stores the acknowledge that are returned by the receive device. This bit can be modified. 0: Receive acknowledge = 0 1: Receive acknowledge = 1
0	ACKBT	0	R/W	Transmit Acknowledge In receive mode, this bit specifies the bit to be sent at the acknowledge timing. 0: 0 is sent at the acknowledge timing. 1: 1 is sent at the acknowledge timing.

---

Bit	Bit Name	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 IC and ICDRT becomes empty</li> <li>• When the TRS bits are set</li> <li>• When the start (re-transmit included) condition has been issued</li> <li>• When switched from reception to transmission in slave mode</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>• When 0 is written to this bit after reading TDRE (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 TEND flag is 1</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>• When 0 is written to this bit after reading TEND (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>



				<ul style="list-style-type: none"> <li>When data is read from ICDRR</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 device in transmission while the ACKE bit 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 1</li> </ul> <p>(When the CPU is used to clear this flag bit 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>
3	STOP	0	R/W	<p>Stop Condition Detection Flag</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a stop condition is detected after frame transfer</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to this bit after reading 1</li> </ul> <p>(When the CPU is used to clear this flag bit 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</p>

- When the internal SDA and the SDA pin level disagree at the rising of SCL in master transmit mode
- When the SDA pin outputs a high level in master transmit mode while a start condition is detected

[Clearing condition]

- When 0 is written to this bit after reading AAS (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.)

---

1	AAS	0	R/W	<p>Slave Address Recognition Flag</p> <p>In slave receive mode, this flag is set to 1 when a data frame following a start condition matches bits 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</li> </ul>
---	-----	---	-----	---

---

### 21.3.6 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
Bit Name	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0
Initial Value	0	0	0	0	0	0	0
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	Slave Address 6 to 0 These bits set a unique address differing from the addresses of other slave devices connected to the bus.
0	—	0	R/W	Reserved Although this bit is readable/writable, only 0 should be written to.

### 21.3.8 I<sup>2</sup>C Bus Receive Data Register (ICDRR)

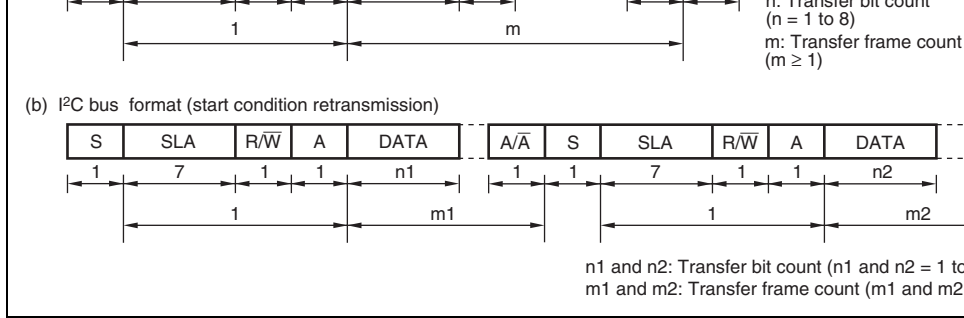
ICDRR is an 8-bit read-only register that stores the receive data. When one byte of data has been received, ICDRR transfers the receive data from ICDRS to ICDRR and the next data can be received. ICDRR is a receive-only register; therefore, this register cannot be written to by CPU.

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

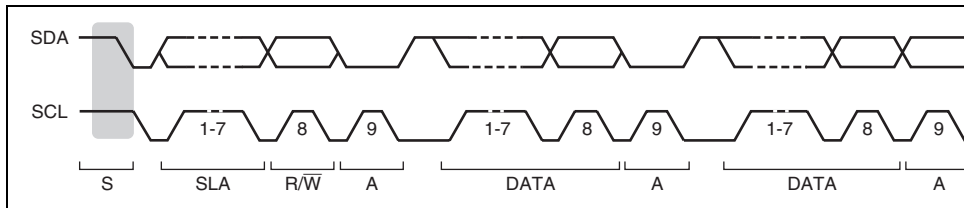
### 21.3.9 I<sup>2</sup>C Bus Shift Register (ICDRS)

ICDRS is an 8-bit 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 or written to by the CPU.

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



**Figure 21.3 I<sup>2</sup>C Bus Formats**

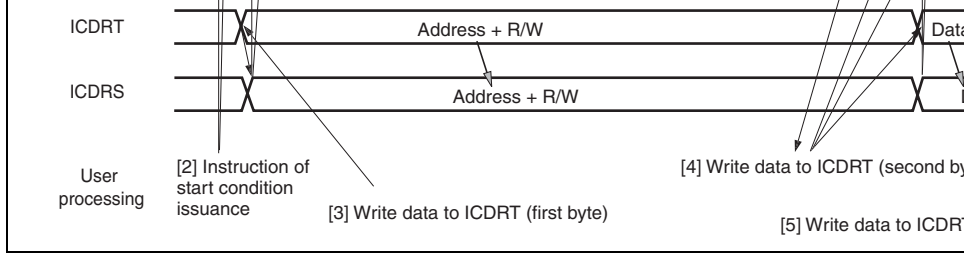


**Figure 21.4 I<sup>2</sup>C Bus Timing**

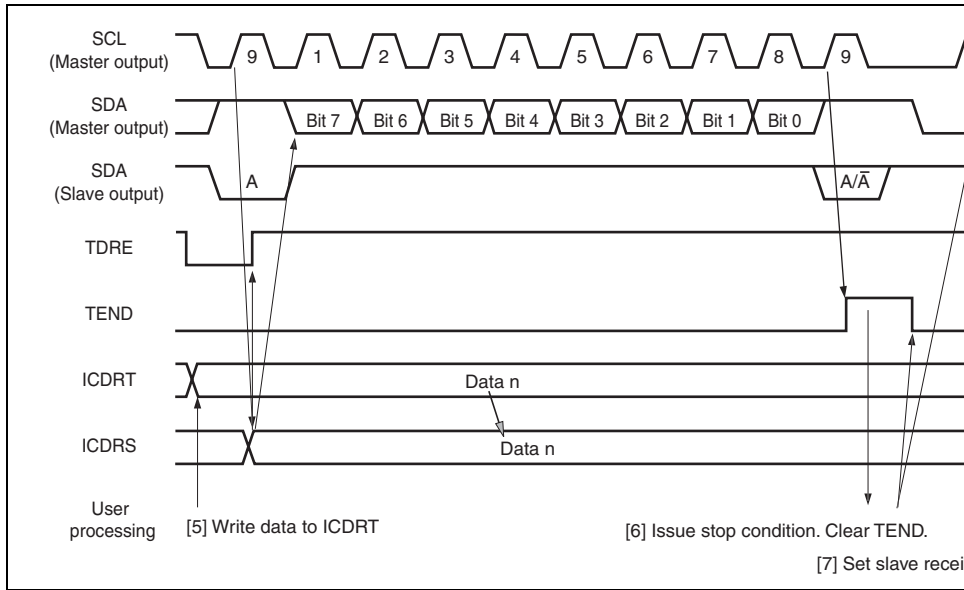
[Legend]

- 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.

- ICDR1 to select master transmit mode. Then, write 1 to BBS1 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 of the slave address and R/W) to ICDRT. After this, when TDRE is automatically cleared, the 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 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 TDRE 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 the byte data transmission) while TDRE is 1, or wait for NACK (NACKF in ICSR is 1) from the receive device while ACKE in ICIER is 1. Then, issue the stop condition to clear TEND and NACKF.
  7. When the STOP bit in ICSR is set to 1, the operation returns to the slave receive mode.



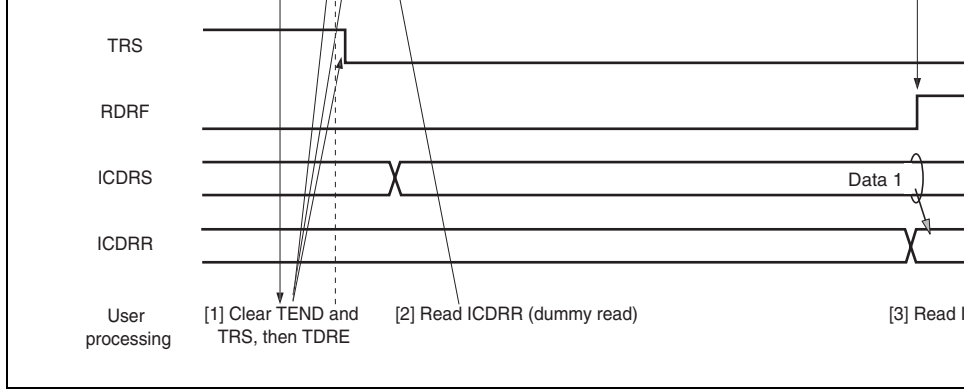
**Figure 21.5 Master Transmit Mode Operation Timing 1**



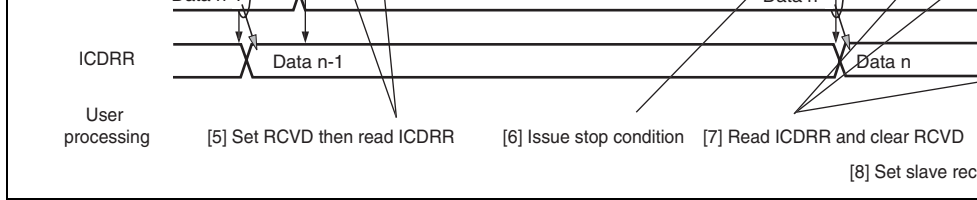
**Figure 21.6 Master Transmit Mode Operation Timing 2**

- data is received, in synchronization with the internal clock. The master mode output 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. If RDRF is set. If the eighth receive clock pulse falls after reading ICDRR by other process 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 ICCRA 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 21.7 Master Receive Mode Operation Timing 1**

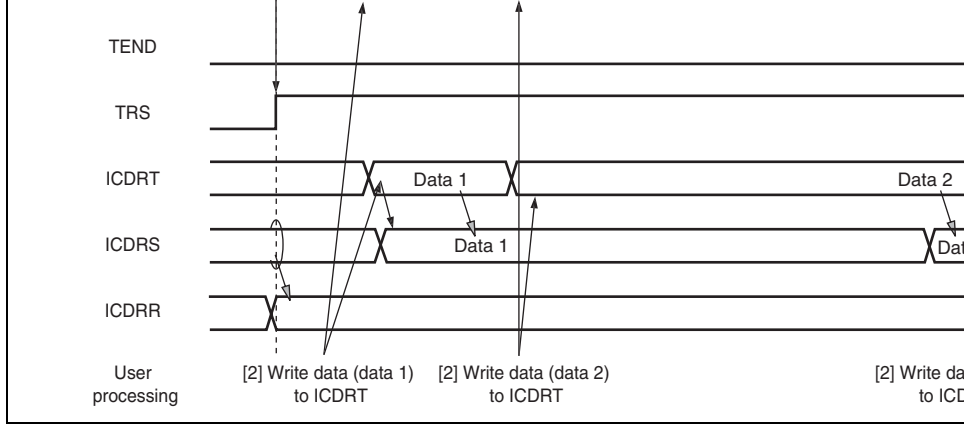


**Figure 21.8 Master Receive Mode Operation Timing 2**

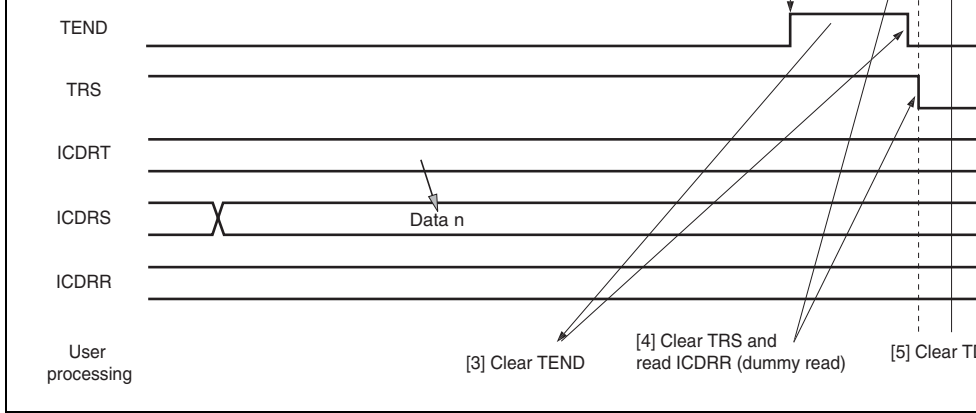
#### 21.4.4 Slave Transmit Operation

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 21.9 and 21.10 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 WAIT in ICMR and CKS3 to CKS0 in ICCRA (initial setting). Set the MST and TRS 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/W) is 1, TRS in ICMR 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 even if 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 release SCL.
5. Clear TDRE.

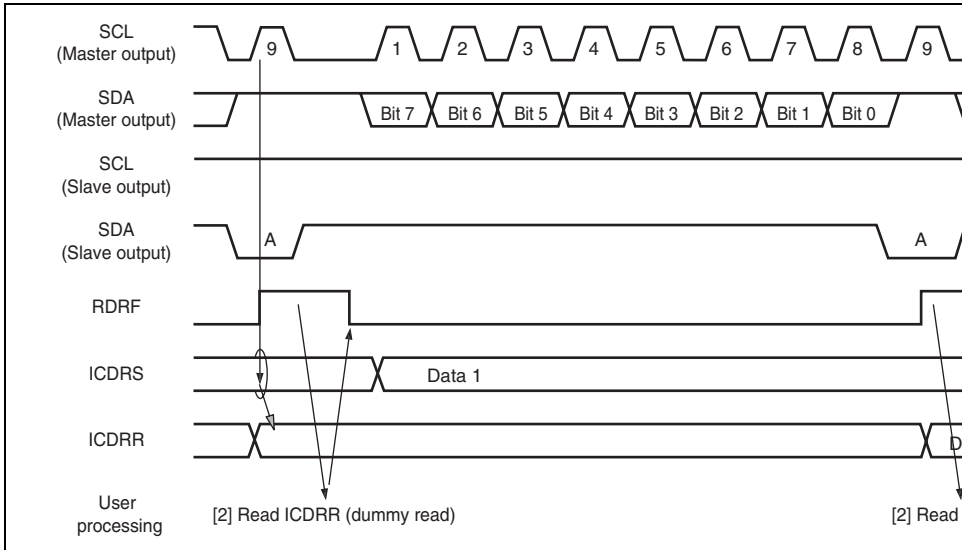


**Figure 21.9 Slave Transmit Mode Operation Timing 1**

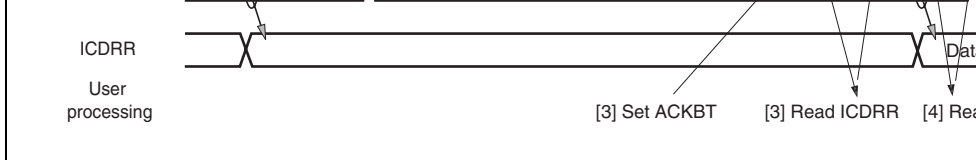


**Figure 21.10 Slave Transmit Mode Operation Timing 2**

2. When the slave address matches in the first frame following detection of the start pulse, the slave address outputs the level specified by ACKBT in ICIER to SDA, at the rising edge 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  $R/\overline{W}$ , it is not used).
3. Read ICDRR every time RDRF is set. If the eighth clock pulse falls while RDRF is fixed to a low level until ICDRR is read. The change of the acknowledge (ACKBT) before reading ICDRR to be returned to the master device is reflected in the next transfer frame.
4. The last byte data is read by reading ICDRR.



**Figure 21.11 Slave Receive Mode Operation Timing 1**

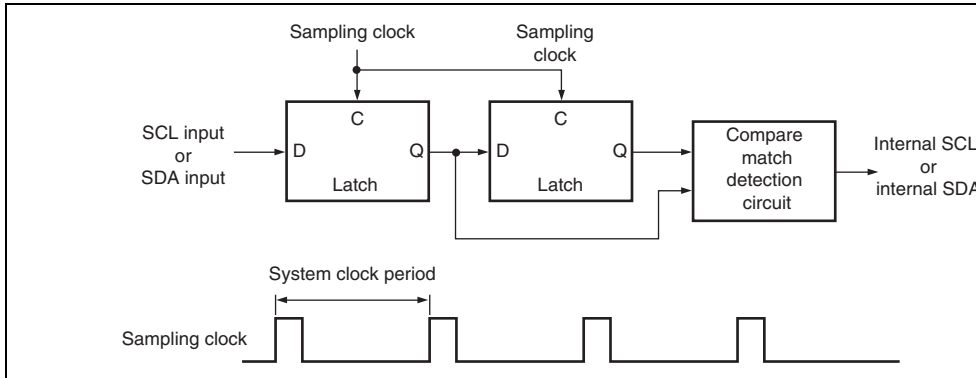


**Figure 21.12 Slave Receive Mode Operation Timing 2**

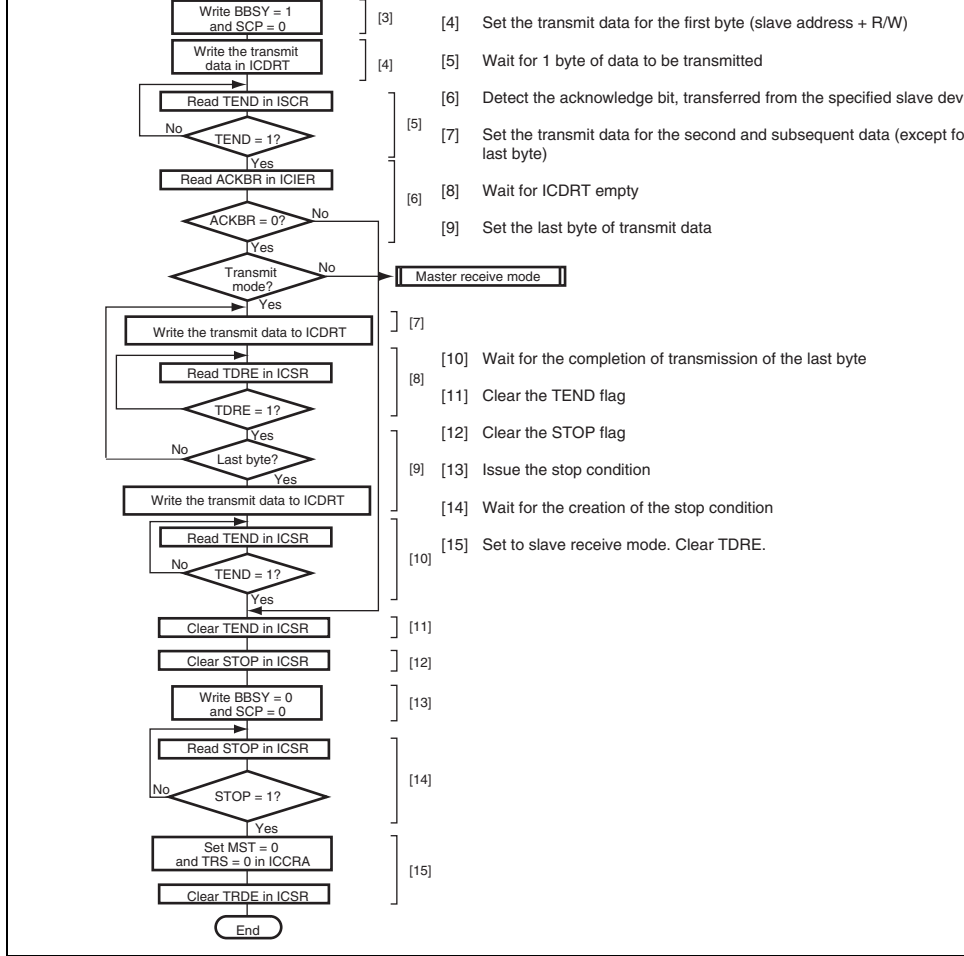
### 21.4.6 Noise Canceler

The logic levels at the SCL and SDA pins are routed through the noise cancelers before being latched internally. Figure 21.13 shows a block diagram of the noise canceler circuit.

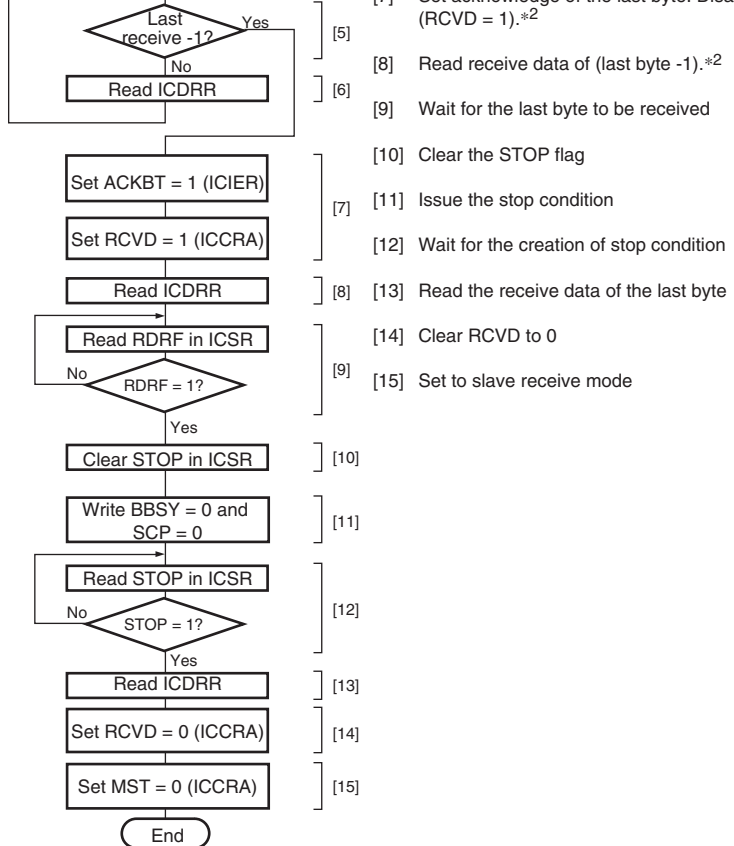
The noise canceler consists of two cascaded latches and a match detector. The signal input (or SDA) is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.



**Figure 21.13 Block Diagram of Noise Canceler**



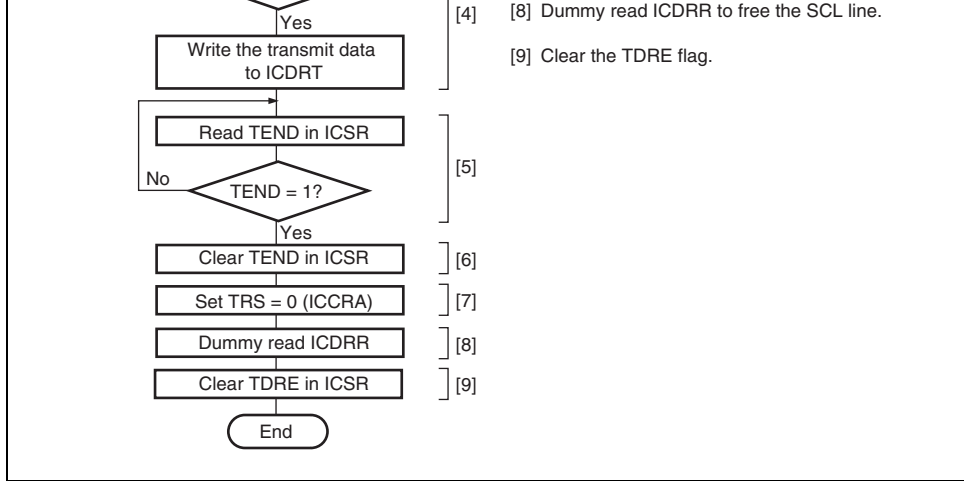
**Figure 21.14 Sample Flowchart of Master Transmit Mode**



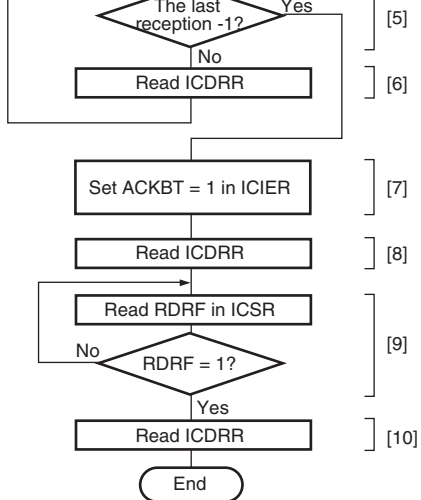
- Note: 1. Do not generate an interrupt during steps [1] to [3].  
 2. For one-byte reception, steps [2] to [6] do not need to be executed. After step [1], execute step [7]. In step [8], read ICDRR (dummy read).

**Figure 21.15 Sample Flowchart for Master Receive Mode**





**Figure 21.16 Sample Flowchart for Slave Transmit Mode**



[5] Read the receive data of (last byte -1).\*

[6] Wait for the reception of the last byte to be complete.

[7] Set ACKBT = 1 in ICIER.

[8] Read the receive data of (last byte -1).\*

[9] Wait for the reception of the last byte to be complete.

[10] Read the last byte of receive data.

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 ICRRR (dummy read).

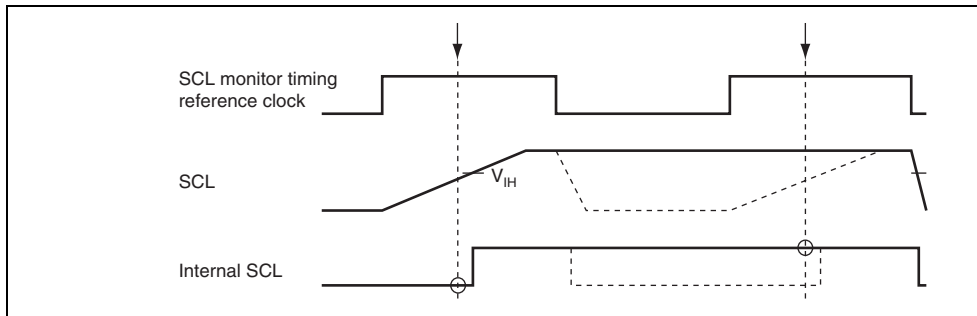
**Figure 21.17 Sample Flowchart for Slave Receive Mode**

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		

When one of the interrupt conditions in table 21.3 is 1 and the I bit in CCR is 0, the CPU interrupt exception handling. Clear the interrupt sources during interrupt exception handling that the TDRE and TEND bits are automatically cleared to 0 by writing data to ICDRT, RDRF bit is cleared to 0 by reading ICDRR. In particular, the TDRE bit can be set again at the same time as data are for transmission written to ICDRT, and 1 extra byte can be transmitted. TDRE is again cleared to 0.

Therefore, this module monitors SCL and communicates bit by bit in synchronization.

Figure 21.18 shows the timing of the bit synchronous circuit, and table 21.4 shows the time for monitoring SCL output changes from low to Hi-Z and the period which SCL is monitored.



**Figure 21.18 Timing of the Bit Synchronous Circuit**

**Table 21.4 Time for Monitoring SCL**

<b>CKS3</b>	<b>CKS2</b>	<b>Time for Monitoring SCL</b>
0	0	7.5 tcyc
	1	19.5 tcyc
1	0	17.5 tcyc
	1	41.5 tcyc

during the eighth clock.

2. 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 is shorter than a given period.

3. Restriction in transfer rate setting value in multi-master mode

When the transfer rate of I<sup>2</sup>C transfer of this LSI is slower than that of other master, signal the width of which is unexpected may be output. To avoid this phenomenon, set the transfer rate of 1/1.8 or more of the fastest rate of other master to the transfer rate of this LSI. For example, if the fastest rate of other masters is 400 kbps, the I<sup>2</sup>C transfer rate of this LSI should be 223 kbps (= 400/1.8) or more.

4. Restriction in bit manipulation when the MST and TRS bits are set in multi-master mode

When the MST and TRS bits are set to master slave mode by manipulating these bits sequentially, the conflict state occurs as follows according to the timing that arbitration is lost. The AL bit in ICSR is set to 0, and set to master mode (MST = 1, TRS = 1). There are the following methods to avoid this phenomenon.

- In multi-master mode, set the MST and TRS bits by MOV instruction.
- When arbitration is lost, confirm that the MST and TRS bits are set to 0. If these bits are set to other than 0, set these bits to 0.

5. Notes on master receive mode

In master receive mode, the RDRF bit is set to 0 at the eighth rising clock, the SCL signal is pulled to “Low” state. When ICDRR is read near at the eighth falling clock, the SCL signal level is released and the ninth clock is output by fixing the eighth clock of receive data to “Low” state. Reading ICDRR is not required. As a result, the failure to receive data may occur. There are the following methods to avoid this phenomenon.

- In master receive mode, read ICDRR by the eighth rising clock.



- 10-bit resolution
- Eight or four input channels (total eight input channels for the two units)
  - Four channels x two units (for unit 0 and unit 1)
  - Eight channels x one unit (for unit 0)
- Conversion time: 2.7  $\mu$ s per channel (in peripheral clock mode)
  - 1.0  $\mu$ s per channel (in system clock mode\*<sup>3</sup>)
- 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\*<sup>1</sup>
- Eight data registers for the A/D converter unit 0 and four data registers for unit 1 (total 12 data registers for the two units)
 

Results of A/D conversion are held in a 16-bit data register for each channel.
- Sample and hold functionality
- Three types of conversion start
 

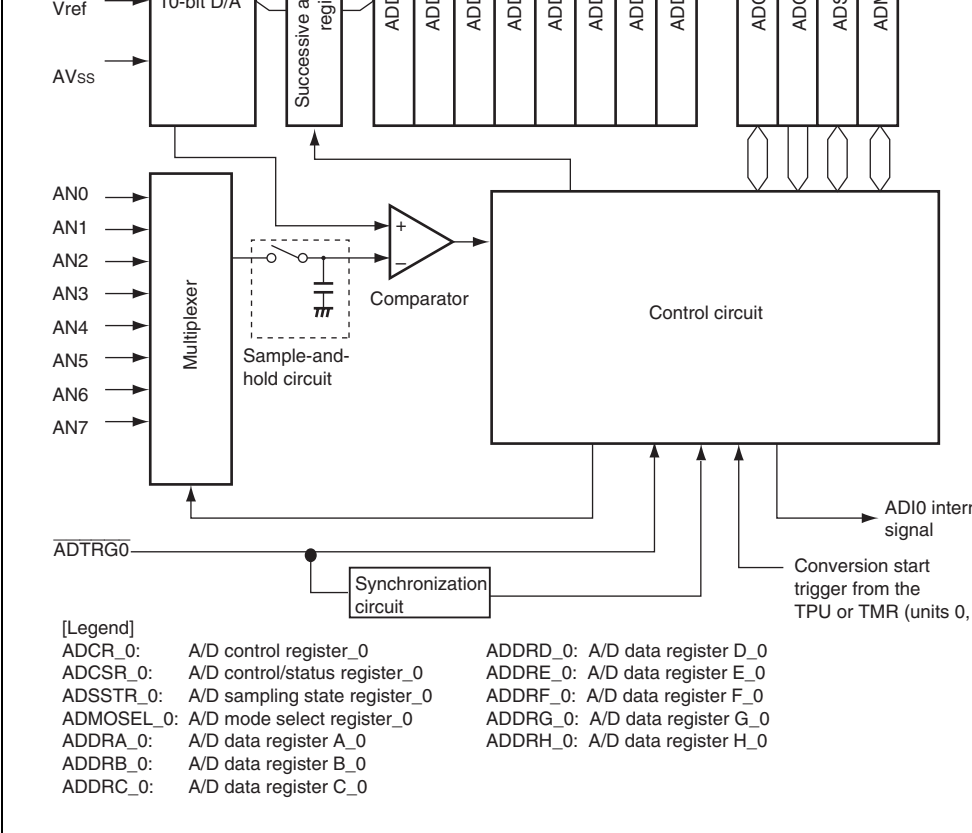
Conversion can be started by software, a conversion start trigger by the 16-bit timer (TPU)\*<sup>1</sup> or 8-bit timer (TMR)\*<sup>2</sup>, or an external trigger signal.
- Function of starting units simultaneously
 

A/D conversion for multiple units can be started by external trigger ( $\overline{\text{ADTRG0}}$ ).
- Interrupt source
 

A/D conversion end interrupt (ADI) request can be generated.
- Module stop state specifiable

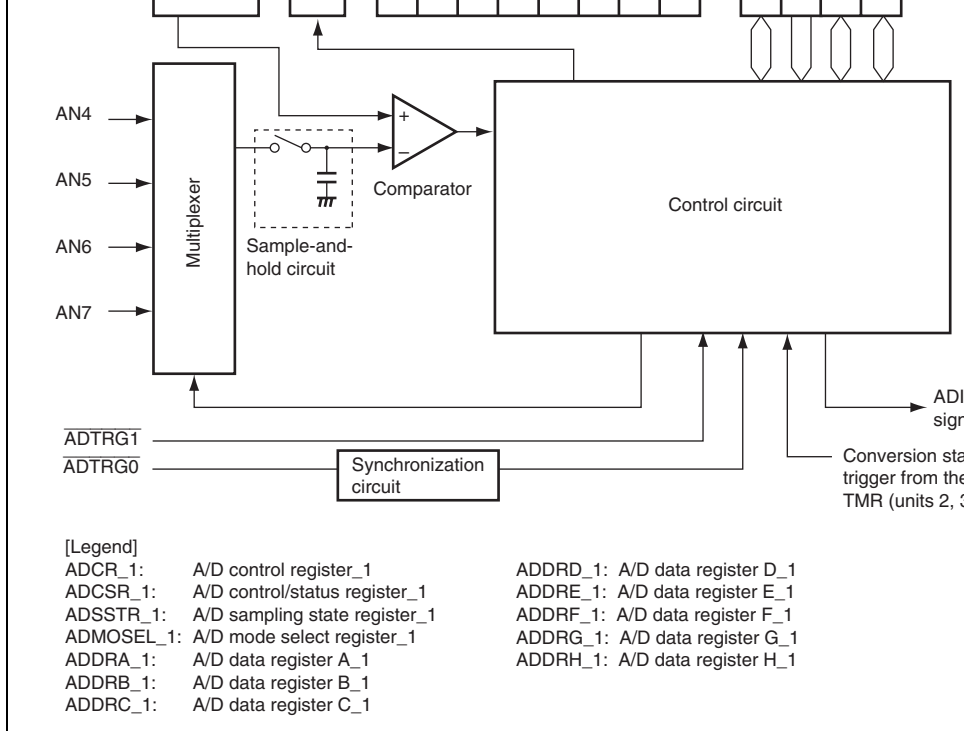
Notes: 1. Only supported in the A/D converter unit 0.

2. For unit 0, A/D conversion can be started by a conversion start trigger by the units 0 and 1, whereas for unit 1, A/D conversion can be started by a conversion start trigger by the TMR units 2 and 3.



**Figure 22.1 Block Diagram of A/D Converter Unit 0 (AD\_0)**





**Figure 22.2 Block Diagram of A/D Converter Unit 1 (AD\_1)**

		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 0	ADTRG0	Input	External trigger input starting A/D conversion
1	AD_1	Analog input pin 4	AN4	Input	Analog inputs
		Analog input pin 5	AN5	Input	
		Analog input pin 6	AN6	Input	
		Analog input pin 7	AN7	Input	
		A/D external trigger input pin 0	ADTRG0	Input	External trigger input starting A/D conversion
		A/D external trigger input pin 1	ADTRG1	Input	External trigger input starting A/D conversion
Common		Analog power supply pin	$AV_{CC}$	Input	Analog block power supply
		Analog ground pin	$AV_{SS}$	Input	Analog block ground
		Reference voltage pin	Vref	Input	A/D conversion reference voltage

Note: \* Selectable by setting of the TRGS1, TRGS0, and EXTRGS bits in ADCR.

- A/D data register E\_0 (ADDRE\_0)
- A/D data register F\_0 (ADDRF\_0)
- A/D data register G\_0 (ADDRG\_0)
- A/D data register H\_0 (ADDRH\_0)
- A/D control/status register\_0 (ADCSR\_0)
- A/D control register\_0 (ADCR\_0)
- A/D mode selection register\_0 (ADMOSEL\_0)
- A/D sampling state register\_0 (ADSSTR\_0)

**Unit 1 (A/D\_1) registers:**

- A/D data register A\_1 (ADDRA\_1)
- A/D data register B\_1 (ADDRB\_1)
- A/D data register C\_1 (ADDRC\_1)
- A/D data register D\_1 (ADDRD\_1)
- A/D data register E\_1 (ADDRE\_1)
- A/D data register F\_1 (ADDRF\_1)
- A/D data register G\_1 (ADDRG\_1)
- A/D data register H\_1 (ADDRH\_1)
- A/D control/status register\_1 (ADCSR\_1)
- A/D control register\_1 (ADCR\_1)
- A/D mode selection register\_1 (ADMOSEL\_1)
- A/D sampling state register\_1 (ADSSTR\_1)

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

**Table 22.2 Analog Input Channels and Corresponding ADDR Registers**

Analog Input Channel	A/D Data Register Storing Conversion Result	
	Unit 0	Unit 1* <sup>2</sup>
AN0	ADDRA_0 (Unit 0)	—
AN1	ADDRB_0 (Unit 0)	—
AN2	ADDRC_0 (Unit 0)	—
AN3	ADDRD_0 (Unit 0)	—
AN4	ADDRE_0 (Unit 0)* <sup>1</sup>	ADDRE_1 (Unit 1)* <sup>1</sup>
AN5	ADDRF_0 (Unit 0)* <sup>1</sup>	ADDRF_1 (Unit 1)* <sup>1</sup>
AN6	ADDRG_0 (Unit 0)* <sup>1</sup>	ADDRG_1 (Unit 1)* <sup>1</sup>
AN7	ADDRH_0 (Unit 0)* <sup>1</sup>	ADDRH_1 (Unit 1)* <sup>1</sup>

Notes: 1. A/D conversion should not be performed on the same channel by multiple units.  
2. The ADDRA\_1 to ADDRD\_1 registers for unit 1 are not used.

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W) <sup>*1</sup>	<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>Completion of A/D conversion in single mode</li> <li>Completion of A/D conversion on all specified channels in scan mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>Writing of 0 after reading ADF = 1 (When the CPU is used to clear this flag by software while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>Reading from ADDR after activation of the DTC by an ADI interrupt</li> </ul>
6	ADIE	0	R/W	<p>A/D Interrupt Enable</p> <p>Setting this bit to 1 enables ADI interrupts by A/D conversion.</p>

cleared to 0 upon completion of A/D conversion of the selected channels to stop A/D conversion.

The ADST bit is automatically cleared at a different time from that of setting the ADF bit. The ADST bit is cleared before setting the ADF bit.

---

4	EXCKS <sup>*2</sup>	0	R/W	Extended Clock Selection
---	---------------------	---	-----	--------------------------

Sets the A/D conversion time in accord with bits CKS1 and CKS0 in ADCR and the ICKSEL bit in ADMCR. For details, see section 22.3.4, A/D Control Register (ADCR\_0) for Unit 0. Write to the EXCKS bit at the same time as bits CKS1 and CKS0 in ADCR.

---

0101: AN3

0110: AN6

0111: AN7

1xxx: Setting prohibited

- When SCANE = 1 and SCANS = 0

0000: AN0

0001: AN0 and AN1

0010: AN0 to AN2

0011: AN0 to AN3

0100: AN4

0101: AN4 and AN5

0110: AN4 to AN6

0111: AN4 to AN7

1xxx: Setting prohibited

- When SCANE = 1 and SCANS = 1

0000: AN0

0001: AN0 and AN1

0010: AN0 to AN2

0011: AN0 to AN3

0100: AN0 to AN4

0101: AN0 to AN5

0110: AN0 to AN6

0111: AN0 to AN7

1xxx: Setting prohibited

---

[Legend]

x: Don't care

Notes: 1. Only 0 can be written to this bit, to clear the flag.

2. The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E6000H) (via USB) is usable.

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>• Completion of A/D conversion in single mode</li> <li>• Completion of A/D conversion on all specified channels in scan mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• Writing of 0 after reading ADF = 1 (When the CPU is used to clear this flag by writing 0 to it, the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)</li> <li>• Reading from ADDR after activation of the DMAC or after an ADI interrupt</li> </ul>
6	ADIE	0	R/W	<p>A/D Interrupt Enable</p> <p>Setting this bit to 1 enables ADI interrupts by ADF.</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 the bit is cleared to 0 by software, a reset, or hardware standby mode. In addition, when the ADSTCLR bit in ADCR is 1, the ADST bit is automatically cleared to 0 upon completion of A/D conversion on all of the selected channels to stop A/D conversion.</p> <p>The ADST bit is automatically cleared at a different time than that of setting the ADF bit. The ADST bit is cleared before setting the ADF bit.</p>



00xx: Setting prohibited

0100: AN4

0101: AN5

0110: AN6

0111: AN7

1xxx: Setting prohibited

- When SCANE = 1 and SCANS = 0

00xx: Setting prohibited

0100: AN4

0101: AN4 and AN5

0110: AN4 to AN6

0111: AN4 to AN7

1xxx: Setting prohibited

- When SCANE = 1 and SCANS = 1

xxxx: Setting prohibited

---

[Legend]

x: Don't care

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

Bit	Bit Name	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: Disables starting of A/D conversion by external trigger</p> <p>010: A/D conversion is started by conversion trigger from T (unit 0)</p> <p>100: A/D conversion is started by conversion trigger from T (units 0 and 1)</p> <p>110: A/D conversion is started by the <math>\overline{\text{ADTRG0}}</math> signal*<sup>1</sup></p> <p>001: External trigger is invalid</p> <p>011: Setting prohibited</p> <p>101: Setting prohibited</p> <p>111: A/D conversion is started by the <math>\overline{\text{ADTRG0}}</math> signal*<sup>1</sup> (start simultaneously)</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 channels 1 to 4.</p> <p>11: Scan mode. A/D conversion is performed continuously channels 1 to 8.</p>

0001: A/D conversion time = 268 states\*<sup>4</sup> (max.)  
 0010: A/D conversion time = 138 states\*<sup>4</sup> (max.)  
 0011: A/D conversion time = 73 states\*<sup>4</sup> (max.)  
 01xx: Prohibited setting  
 1000: A/D conversion time = 336 states\*<sup>4</sup> (max.)  
 1001: A/D conversion time = 172 states\*<sup>4</sup> (max.)  
 1010: A/D conversion time = 90 states\*<sup>4</sup> (max.)  
 1011: A/D conversion time = 49 states\*<sup>4</sup> (max.)  
 11xx: A/D conversion time = 34 states\*<sup>4</sup>\*<sup>5</sup> (max.)  
 This setting applies when  $P\phi = I\phi/2$ \*<sup>6</sup>.

---

1	ADSTCLR* <sup>2</sup>	0	R/W	A/D Start Clear
---	-----------------------	---	-----	-----------------

Enables or disables automatic clearing of the ADST bit in scan mode.

0: The ADST bit is not automatically cleared to 0 in scan mode.  
 1: The ADST bit is cleared to 0 upon completion of A/D conversion for all of the selected channels in scan mode.

---

[Legend]

x: Don't care

- Notes:
1. To set A/D conversion to start by the  $\overline{ADTRG}$  pin, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 10.1.1.1 Ports.
  2. The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
  3. ICKSEL = 1: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
  4. Cycles of  $P\phi$
  5. Set the number of "states" (clock cycles) for sampling to 25 ( $ADSSTR_0 = D$ ).
  6. When  $P\phi = I\phi, I\phi/4,$  or  $I\phi/8,$  settings of the form 11xx are prohibited.

Bit	Bit Name	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	000: Disables starting of A/D conversion by external trigger 010: Setting prohibited 100: Setting prohibited 110: A/D conversion is started by the $\overline{ADTRG1}$ signal* <sup>1</sup> 001: Setting prohibited 011: External trigger is invalid 101: A/D conversion is started by conversion trigger from TM2 and 3) 111: A/D conversion is started by the $\overline{ADTRG0}$ signal* <sup>1</sup> (start simultaneously)
5	SCANE	0	R/W	Scan Mode
4	SCANS	0	R/W	These bits select the A/D conversion operating mode. 0x: Single mode 10: Scan mode. A/D conversion is performed continuously for channels 1 to 4. 11: Setting prohibited

0001: A/D conversion time = 268 states\*<sup>3</sup> (max.)  
0010: A/D conversion time = 138 states\*<sup>3</sup> (max.)  
0011: A/D conversion time = 73 states\*<sup>3</sup> (max.)  
01xx: Prohibited setting  
1000: A/D conversion time = 336 states\*<sup>3</sup> (max.)  
1001: A/D conversion time = 172 states\*<sup>3</sup> (max.)  
1010: A/D conversion time = 90 states\*<sup>3</sup> (max.)  
1011: A/D conversion time = 49 states\*<sup>3</sup> (max.)  
11xx: A/D conversion time = 34 states\*<sup>3</sup>\*<sup>4</sup> (max.)  
This setting applies when  $P\phi = I\phi/2$ \*<sup>5</sup>.

---

1	ADSTCLR 0	R/W	A/D Start Clear	Enables or disables automatic clearing of the ADST bit in scan mode. 0: The ADST bit is not automatically cleared to 0 in scan mode. 1: The ADST bit is cleared to 0 upon completion of A/D conversion of all of the selected channels in scan mode.
---	-----------	-----	-----------------	--

---

[Legend]

x: Don't care

- Notes:
1. To set A/D conversion to start by the ADTRG pin, the DDR bit and ICR bit for corresponding pin should be set to 0 and 1, respectively. For details, see section 10.1.1 Ports.
  2. Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (USB) is usable.
  3. Cycles of  $P\phi$
  4. Set the number of "states" (clock cycles) for sampling to 25 ( $ADSSTR_1 = D'$ ).
  5. When  $P\phi = I\phi$ ,  $I\phi/4$ , or  $I\phi/8$ , settings of the form 11xx are prohibited.

Bit	Bit Name	Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	—	0	R	
2	—	0	R	
1	ICKSEL* <sup>1</sup>	0	R/W	System Clock Mode Selection This bit is used to select the system clock mode. 0: Peripheral clock mode 1: System clock mode* <sup>2</sup> For details, see section 22.4.5, Setting the System Clock Mode.
0	—	0	R/W	Reserved This bit is always read as 0. The write value should be 0.

- Notes: 1. The full-spec emulator (E6000H) should not be used, but the on-chip emulator (USB) is usable.
2. In system clock mode, operate all units with the system clock.

Bit	Bit Name	Value	R/W	Description
7	—	0	R	Reserved
6	—	0	R	These bits are always read as 0. The write value should always be 0.
5	—	0	R	
4	—	0	R	
3	—	0	R	
2	—	0	R	
1	ICKSEL* <sup>1</sup>	0	R/W	System Clock Mode Selection This bit is used to select the system clock mode. 0: Peripheral clock mode 1: System clock mode* <sup>2</sup> For details, see section 22.4.5, Setting the System Mode.
0	—	0	R/W	Reserved This bit is always read as 0. The write value should be 0.

- Notes:
1. Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (E6000H) (USB) is usable.
  2. In system clock mode, operate all units with the system clock.

Bit	Bit Name	Initial Value	R/W	Description
7	SMP7	0	R/W	<ul style="list-style-type: none"> <li>When EXCKS = 0</li> </ul>
6	SMP6	0	R/W	Set these bits to H'0F.
5	SMP5	0	R/W	<ul style="list-style-type: none"> <li>When EXCKS = 1</li> </ul>
4	SMP4	0	R/W	If ICKSEL = 0, set these bits to H'0F.
3	SMP3	1	R/W	If ICKSEL = 1, set these bits to H'19.
2	SMP2	1	R/W	
1	SMP1	1	R/W	
0	SMP0	1	R/W	

Note: \* The full-spec emulator (E6000H) should not be used, but the on-chip emulator (USB) is usable.



Bit	Bit Name	Initial Value	R/W	Description
7	SMP7	0	R/W	<ul style="list-style-type: none"> <li>When EXCKS = 0</li> </ul>
6	SMP6	0	R/W	Set these bits to H'0F.
5	SMP5	0	R/W	<ul style="list-style-type: none"> <li>When EXCKS = 1</li> </ul>
4	SMP4	0	R/W	If ICKSEL = 0, set these bits to H'0F.
3	SMP3	1	R/W	If ICKSEL = 1, set these bits to H'19.
2	SMP2	1	R/W	
1	SMP1	1	R/W	
0	SMP0	1	R/W	

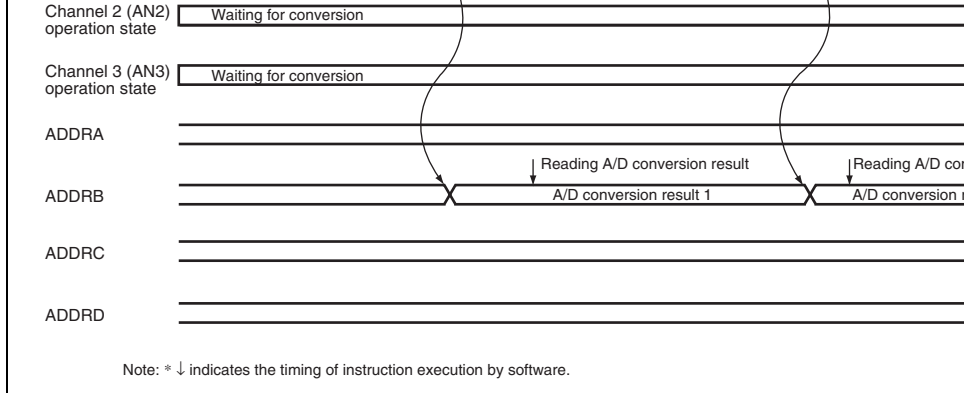
Note: \* Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (USB) is usable.

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\*<sup>1</sup>, TMR\*<sup>2</sup>, 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 at 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 during A/D conversion, A/D conversion stops and the A/D converter enters a wait state.

Notes: 1. Only possible in unit 0.

2. As conversion start trigger, units 0 and 1 of TMR, and units 2 and 3 of TMR are available in unit 0, and unit 1, respectively.



**Figure 22.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)**

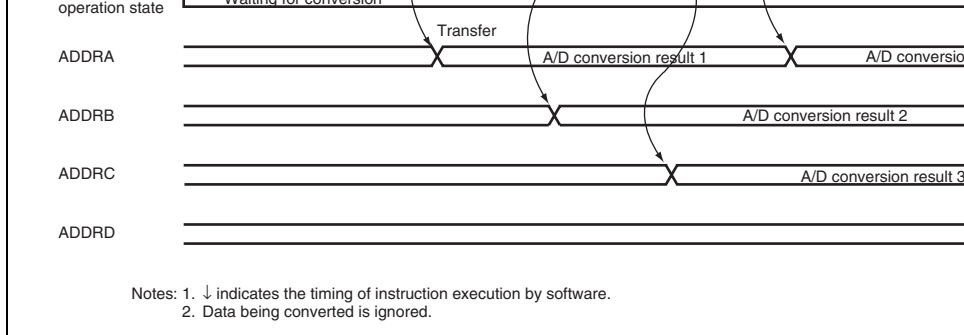
### 22.4.2 Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the analog inputs of the channels up to four or eight\*<sup>1</sup> 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\*<sup>3</sup> where conversion is performed for the specified channels for one cycle.

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 cleared. 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 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.

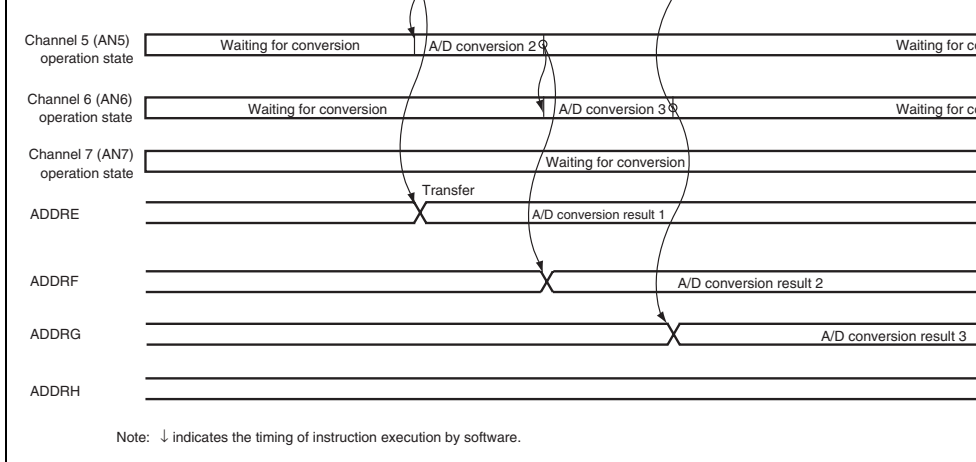
- Notes:
1. Only possible in unit 0.
  2. As conversion start trigger, units 0 and 1 of TMR, and units 2 and 3 of TMR are available in unit 0, and unit 1, respectively.
  3. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.



**Figure 22.4 Example of A/D Conversion  
(Continuous Scan Mode, Three Channels (AN0 to AN2) Selected)**

**(2) One-Cycle Scan Mode\***

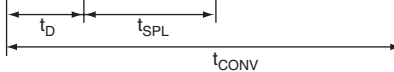
1. Set the ADSTCLR bit in ADCR to 1.
2. When the ADST bit in ADCSR is set to 1 by software, TPU, TMR (units 2 and 3), or external trigger input, A/D conversion starts on the first channel in the specified channel. Consecutive A/D conversion on a maximum of four channels (SCANE and SCANS can be selected. For unit 1, A/D conversion starts on AN4 when CH3 and CH2 = B'00).
3. When A/D conversion for each channel is completed, the A/D conversion result is sent 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 the waiting for conversion state.



**Figure 22.5 Example of A/D Conversion  
(One-Cycle Scan Mode, Three Channels (AN4 to AN6) Selected)**

In scan mode, the values given in tables 22.3 to 22.5 apply to the first conversion time. The values given in table 22.6 apply to the second and subsequent rounds of conversion. In either case, the CKS1 and CKS0 bits in ADCR, the ICKSEL\*<sup>1</sup> bit in ADMOSEL, and the EXCKS\*<sup>2</sup> bit in ADCSR should be set so that the conversion time is within the ranges indicated by the conversion characteristics.

- Notes:
1. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.  
Unit 1: Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (E10A-USB) is usable.
  2. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.



- [Legend]  
 (1): ADCSR write cycle  
 (2): ADCSR address  
 $t_D$ : A/D conversion start delay time  
 $t_{SPL}$ : Input sampling time  
 $t_{CONV}$ : A/D conversion time

**Figure 22.6 Periods of A/D Conversion**

**Table 22.3 Characteristics of A/D Conversion (Unit 0: when EXCK<sup>\*</sup> = 0, ICKSEL = 0 and ADSSTR<sup>\*</sup> = H'0F) (1)**

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.
A/D conversion start delay time	$t_D$	3	—	14	3	—	10	3	—	8	3	—
Input sampling time	$t_{SPL}$	—	312	—	—	156	—	—	78	—	—	3
A/D conversion time	$t_{CONV}$	517	—	528	261	—	268	133	—	138	69	—

Notes: Values in the table are numbers of states.

\* The full-spec emulator (E6000H) should not be used, but the on-chip emulator (USB) is usable.



Notes: Values in the table are numbers of states.

- \* The full-spec emulator (E6000H) should not be used, but the on-chip emulator (USB) is usable.

**Table 22.4 Characteristics of A/D Conversion (Unit 1: when EXCKS = 0, ICKSEL = 0, ADSSTR\* = H'0F) (1)**

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	$t_d$	4	—	14	4	—	10	4	—	8	4	—	10
Input sampling time	$t_{SPL}$	—	312	—	—	156	—	—	78	—	—	—	—
A/D conversion time	$t_{CONV}$	518	—	528	262	—	268	134	—	138	70	—	—

Notes: Values in the table are numbers of states.

- \* Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (USB) is usable.

Notes: Values in the table are numbers of states.

- \* Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (E10A-USB) is usable.

**Table 22.5 Characteristics of A/D Conversion (When EXCKSP<sup>\*1</sup> = 1, ICKSEL<sup>\*1</sup> = 0, ADSSTR<sup>\*2</sup> = H'19)**

Item	Symbol	$\phi:P\phi = 1:1/2$				
		Unit 0			Unit 1	
		Min.	Typ.	Max.	Min.	Typ.
A/D conversion start delay time	$t_{D}$	2.5	—	6.5	3.5	—
Input sampling time	$t_{SPL}$	—	12.5	—	—	12.5
A/D conversion time	$t_{CONV}$	30	—	34	31	—

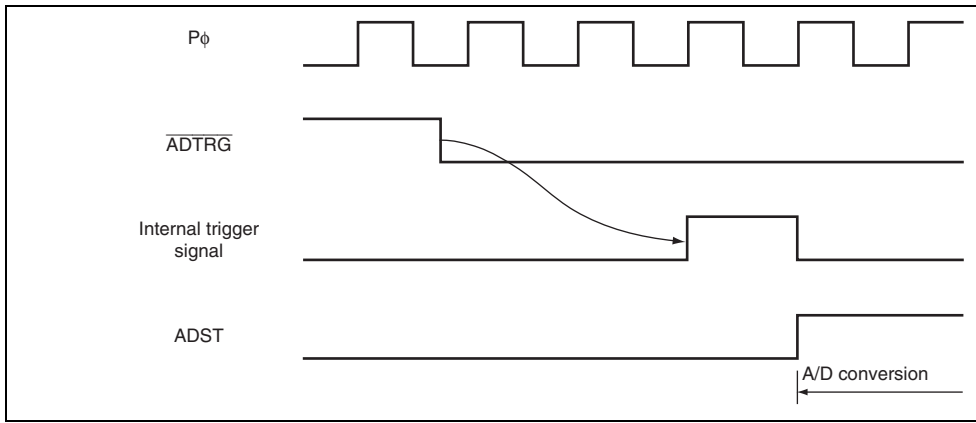
Notes: Values in the table are numbers of states (cycles of  $P\phi$ ). Make the sampling setting (ADSSRT = D'25).

1. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
2. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.  
Unit 1: Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (E10A-USB) is usable.

	1	0	80* <sup>1</sup>
		1	40* <sup>1</sup>
1* <sup>3</sup> * <sup>4</sup>	—	—	25* <sup>2</sup>

- Notes:
1. Make the sampling setting 15 (ADSSRT = D'15).
  2. When  $P\phi = I\phi/2$ , make the sampling setting 25 (ADSSRT = D'25).
  3. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
  4. Unit 1: Access to the full-spec emulator (E6000H) is prohibited but the on-chip emulator (E10A-USB) is usable.

TRGS1, TRGS0, and EXTRGS bits are set to B'111 in ADCR\_0 and ADCR\_1. A/D conversion starts when the ADST bit in ADCSR is set to 1 on the falling edge of the  $\overline{\text{ADTRG0}}$  pin. The timing is different from the one when multiple units do not start simultaneously. Figure 22.7 shows the timing.



**Figure 22.7 External Trigger Input Timing (TRGS1, TRGS0, and EXTRGS  $\neq$  I)**

**Figure 22.8 External Trigger Input Timing when Multiple Units Start Simultaneously (TRSG1, TRGS0, and EXTRGS = B'111)**

### 22.4.5 Setting the System Clock Mode

In system clock mode, set  $I\phi = 50$  MHz,  $P\phi = I\phi/2$ , and make the sampling setting 25. A conversion\*<sup>1</sup> with a conversion time of 1  $\mu$ s per channel is possible. For information on the frequency of the system clock relative to the input clock, see section 27, Clock Pulse Generator.

When the ADST bit is cleared to 0, start A/D conversion following the procedures shown below.

1. Set  $P\phi = I\phi/2$ .
2. Release the A/D converter from the module-stopped state.
3. Set the EXCKS\*<sup>2</sup> bit in ADCSR to 1 (making setting of the number of states for sampling to 25 and ADSSTR\*<sup>2\*3</sup> effective).
4. Set\*<sup>2\*3</sup> the ICKSEL bit in ADMODSEL\*<sup>2\*3</sup> to 1 (selecting system clock mode).
5. Write H'19 to ADSSTR\*<sup>2\*3</sup> (setting the number of states for sampling to 25).
6. Start A/D conversion (set the ADST bit to 1 or have the trigger signal initiate conversion).

- Notes:
1. The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
  2. For unit 0, the full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
  3. For unit 1, access to the full-spec emulator (E6000H) is prohibited, but the on-chip emulator (E10A-USB) is usable.

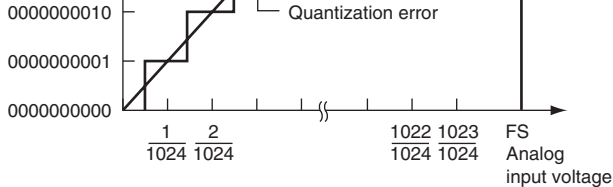
**Table 22.7 A/D Converter Interrupt Source**

<b>Name</b>	<b>Interrupt Source</b>	<b>Interrupt Flag</b>	<b>DTC Activation</b>	<b>DMAC Activation</b>
ADI	A/D conversion end	ADF	Possible*	Possible

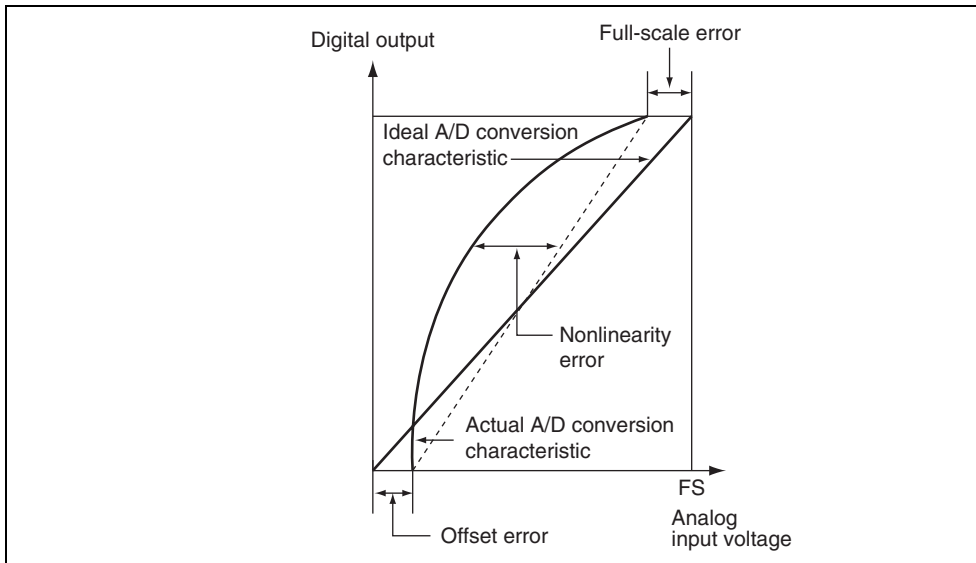
Note: \* Only possible in unit 0.

when the digital output changes from the minimum voltage value B'0000000000 (H'000) to B'0000000001 (H'001) (see figure 22.10).

- Full-scale error  
The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 (H'3FE) to B'1111111111 (H'3FF) (see figure 22.10).
- 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 22.10).
- 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 22.9 A/D Conversion Accuracy Definitions**



**Figure 22.10 A/D Conversion Accuracy Definitions**



### 22.7.2 A/D Input Hold Function in Software Standby Mode

When this LSI enters software standby mode with A/D conversion enabled, the analog input is retained, and the analog power supply current is equal to as during A/D conversion. If the power supply current needs to be reduced in software standby mode, set the CKS1 and CKS0 to 1 and clear the ADST, TRGS1, TRGS0, and EXTRGS bits all to 0 to disable A/D conversion. After that, enter software standby mode after executing a dummy read from ADCSR.

### 22.7.3 Notes on Stopping the A/D Converter

When the A/D start bit (ADST) is cleared during A/D conversion by software, A/D conversion results may be stored incorrectly (ADDR), or when A/D conversion restarts, the interrupt may be missed.

To avoid these events, follow the steps below.

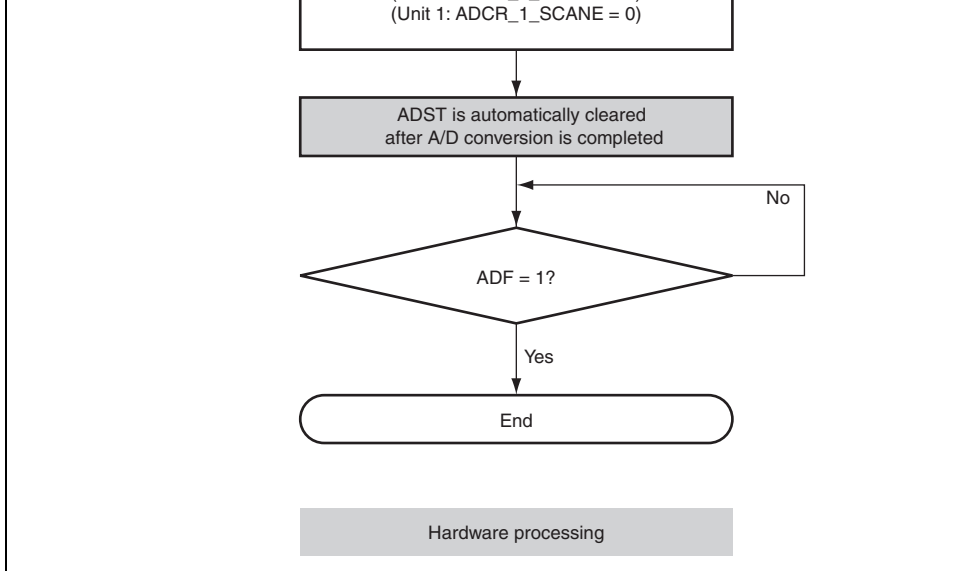
#### (1) In Single Mode or Scan Mode (One-Cycle Scan Mode)

As the ADST bit is automatically cleared when A/D conversion is completed, do not clear the ADST bit during A/D conversion.

#### (2) In Scan Mode (Continuous Scan Mode)

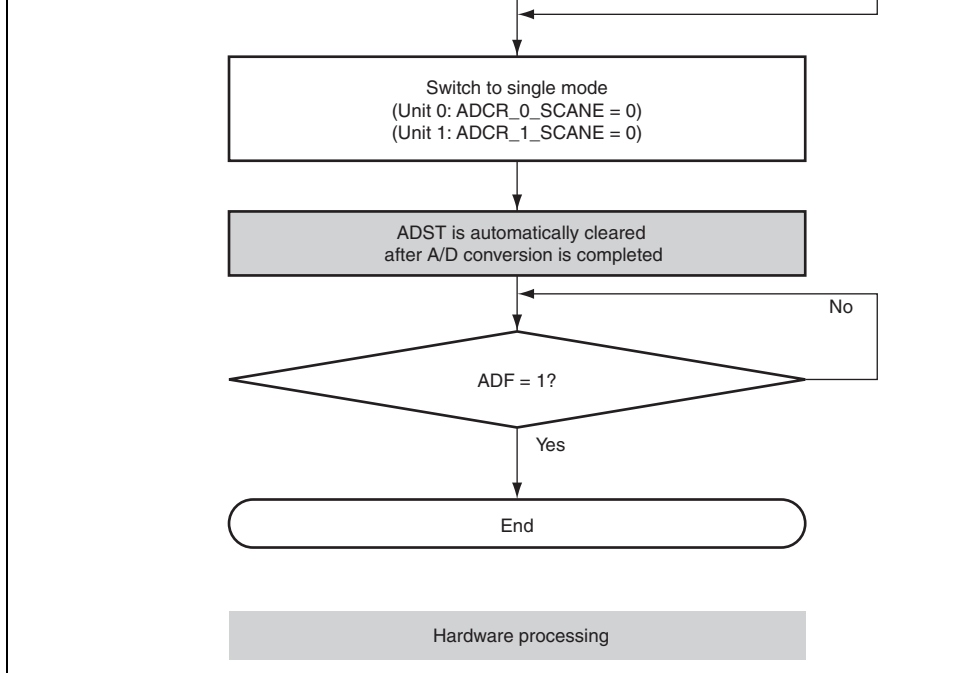
- When the A/D Converter is Activated by Software

Do not clear the ADST bit during A/D conversion. To stop A/D conversion, rewrite the SCANE bit to change modes from scan mode to single mode. By rewriting the SCANE bit, the A/D converter is stopped without clearing the ADST bit by software.



**Figure 22.11 Stopping Continuous Scan Mode Activated by Software**

- When the A/D Converter is Activated by an External Trigger  
 Do not clear the ADST bit during A/D conversion. To stop A/D conversion, disable external triggers and then rewrite the SCANE bit to change modes from scan mode to single mode. This stops A/D conversion without clearing the ADST bit by software.  
 However, after rewriting the SCANE bit, it may take up to 1.5-channel A/D conversions to stop A/D conversion and set the A/D end flag (ADF) to 1. Moreover, the ADDR value after A/D conversion is completed should not be used.  
 For details of settings, see figure 22.12.



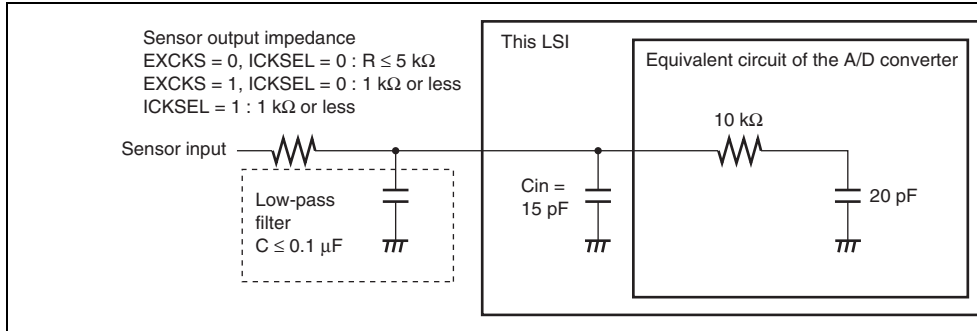
**Figure 22.12 Stopping Continuous Scan Mode Activated by External Triggers**

#### 22.7.4 Notes in System Clock Mode

1. For board design, see section 22.7.8, Notes on Board Design.
2. In system clock mode, operate all units with the system clock ( $I\phi$ ). The system clock peripheral module clock should not be used together.

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/μs or greater) (see figure 22.13). When converting a high-speed analog signal to digital conversion in scan mode, a low-impedance buffer should be inserted.

- Notes:
1. Unit 0: The full-spec emulator (E6000H) should not be used, but the on-chip emulator (E10A-USB) is usable.
  2. Unit 1: Access to the full-spec emulator (E6000H) is prohibited, but the on-chip emulator (E10A-USB) is usable.



**Figure 22.13 Example of Analog Input Circuit**

If the conditions shown below are not met, the reliability of the LSI may be adversely affected.

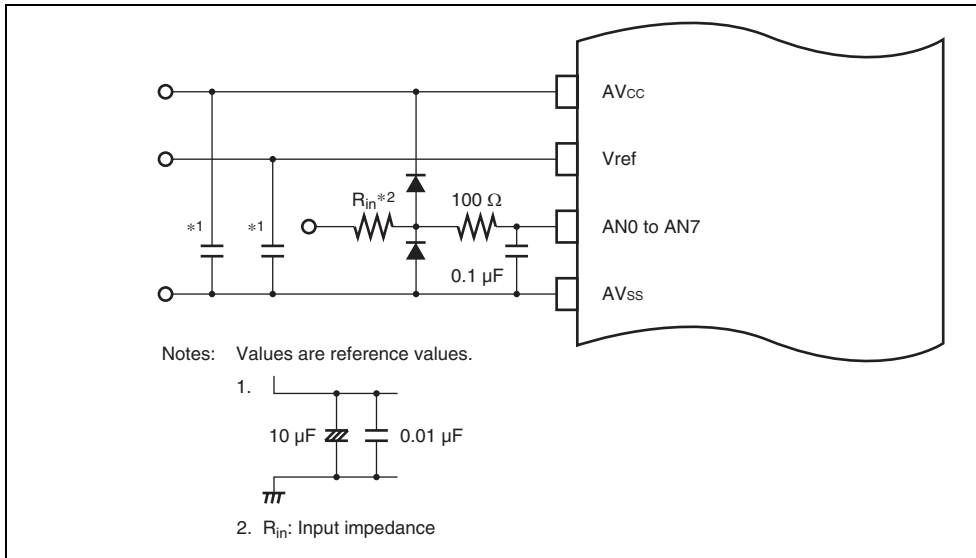
- Analog input voltage range  
The voltage applied to analog input pin ANn during A/D conversion should be in the range  $AV_{SS} \leq V_{AN} \leq V_{ref}$ .
- Relation between  $AV_{CC}$ ,  $AV_{SS}$  and  $V_{CC}$ ,  $V_{SS}$   
As the relationship between  $AV_{CC}$ ,  $AV_{SS}$  and  $V_{CC}$ ,  $V_{SS}$ , set  $AV_{CC} = V_{CC} \pm 0.3 \text{ V}$  and  $AV_{SS} = V_{SS} \pm 0.3 \text{ V}$ .  
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  $V_{ref} \leq AV_{CC}$ .

### 22.7.8 Notes on Board Design

In board design, follow the notes below.

1. Digital circuitry and analog circuitry should be as mutually isolated as possible, and the crossing of 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. Moreover, the digital circuitry must be isolated from the analog reference power supply pin ( $V_{ref}$ ), analog power supply pin ( $AV_{CC}$ ), and analog ground pin ( $AV_{SS}$ ) by shielding the analog input pins (AN0 to AN7) with the analog ground pin ( $AV_{SS}$ ).
2. Lines should be connected with the analog reference power supply pin ( $AV_{CC}$ ), analog power supply pin ( $V_{ref}$ ), and analog ground pin ( $AV_{SS}$ ) with low impedance as possible.
3. The analog ground pin ( $AV_{SS}$ ) should be connected at one point to a stable ground (VSS) on the board.

converter exceeds the current input via the input impedance ( $R_{in}$ ), an error will arise in the input pin voltage. Careful consideration is therefore required when deciding the circuit co



**Figure 22.14 Example of Analog Input Protection Circuit**



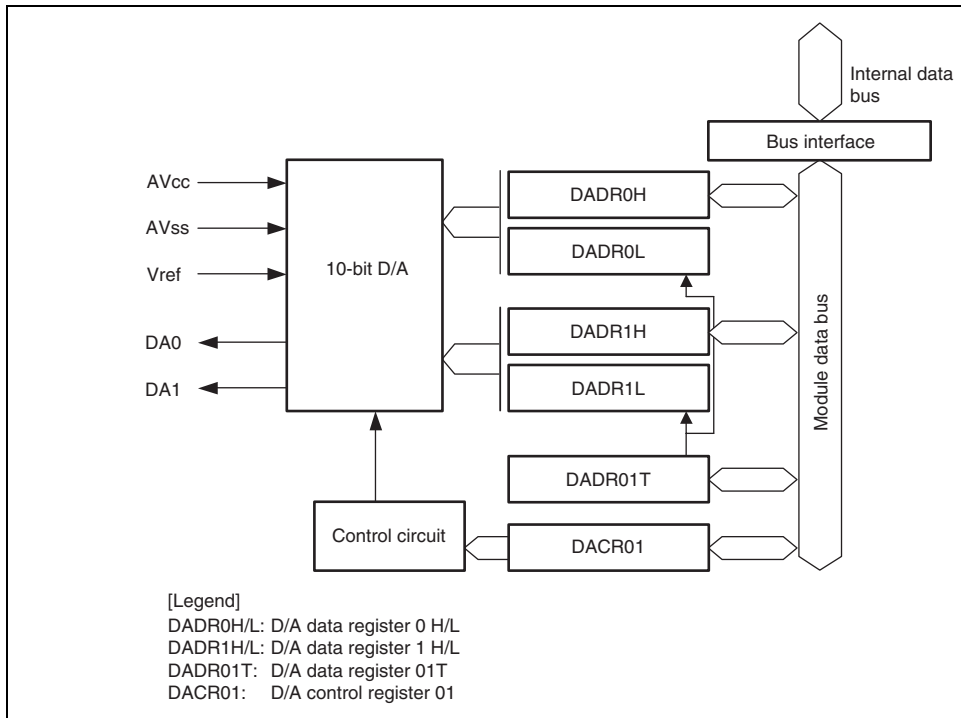
Note: Values are reference values.

**Figure 22.15 Analog Input Pin Equivalent Circuit**





- Module stop state specifiable



**Figure 23.1 Block Diagram of the 10-Bit D/A Converter**

Analog output pin 0	DA0	Output	Channel 0 analog output
Analog output pin 1	DA1	Output	Channel 1 analog output

### 23.3 Register Descriptions

The D/A converter has the following registers.

- D/A data register 0H (DADR0H)
- D/A data register 0L (DADR0L)
- D/A data register 1H (DADR1H)
- D/A data register 1L (DADR1L)
- D/A data register 01T (DADR01T)
- D/A control register 01 (DACR01)

DADR1L are non-readable registers. Writing is accomplished by transferring values from temporary register, DADR01T.

When the value in a DADR is to be updated, the new lower-order two bits of the value previously have been written to DADR01T. The corresponding DADR is actually updated at the same time as a new value is written to DADR0H or DADR1H. The eight higher-order bits are reflected as written in DADR0H or DADR1H, while the two lower-order bits are updated by transfer from DADR01T to DADR0L or DADR1L.

### 23.3.2 D/A Data Registers 0H and 1H (DADR0H and DADR1H)

DADR0H and DADR1H are 8-bit readable/writable registers. When a value is written to DADR0H or DADR1H, the corresponding bits of DADR01T are simultaneously transferred to DADR0L or DADR1L.

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

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

### 23.3.4 D/A Data Register 01T (DADR01T)

DADR01T is an 8-bit temporary register that is used to transfer data to DADR0L and DADR0H. Writing is only valid for bits 7 and 6. Values written to bits 5 to 0 are ignored. In reading, the recent setting (of bits 7 and 6) and the values for the two lower-order bits held in DADR0L and DADR0H can be read. The value for the two lower-order bits of DADR1 can be read in bits 3 and 2. The value for the two lower-order bits of DADR0 can be read in bits 1 and 0.

Bits 1 and 0 are reserved. In reading, these bits are read as 1.

Bits 7 and 6 of DADR01T are used to store the two lower-order bits of a 10-bit value for conversion.

Bit	7	6	5	4	3	2	1
Bit Name	DADT1	DADT0	DAD1L1	DAD1L0	DAD0L1	DAD0L0	—
Initial Value	0	0	0	0	0	0	1
R/W	R/W	R/W	R	R	R	R	R

Bit	Bit Name	Value	R/W	Description
7	DAOE1	0	R/W	<p>D/A Output Enable 1</p> <p>Controls D/A conversion and analog output.</p> <p>0: Analog output of channel 1 (DA1) is disabled.</p> <p>1: D/A conversion of channel 1 is enabled. Analog output of channel 1 (DA1) is enabled.</p>
6	DAOE0	0	R/W	<p>D/A Output Enable 0</p> <p>Controls D/A conversion and analog output.</p> <p>0: Analog output of channel 0 (DA0) is disabled.</p> <p>1: D/A conversion of channel 0 is enabled. Analog output of channel 0 (DA0) is enabled.</p>
5	DAE	0	R/W	<p>D/A Enable</p> <p>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.</p> <p>Output of conversion results is always controlled by the DAOE0 and DAOE1 bits. For details, see table 10-1: Control of D/A Conversion.</p>
4 to 0	—	All 1	R	<p>Reserved</p> <p>These are read-only bits and cannot be modified.</p>

			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.
1	0	0	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is disabled.
		1	D/A conversion of channels 0 and 1 is enabled. Analog output of channel 0 (DA0) is enabled and analog output of channel 1 (DA1) is disabled.
	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	D/A conversion of channels 0 and 1 is enabled. Analog output of channels 0 and 1 (DA0 and DA1) is enabled.

### 23.3.6 Usage as an 8-Bit D/A Converter

In advance of usage as an 8-bit D/A converter, fix the lower-order two bits to 0 by clearing DADT1 and DADT0 bits in DADR01T to 0. By clearing these bits in DADR01T to 0 in the D/A converter is made to function by simply setting DADR0H or DADR1H. That is, conversion of the eight higher-order bits proceeds when the two lower-order bits remain 0.

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. Output values are expressed by the following formulae.

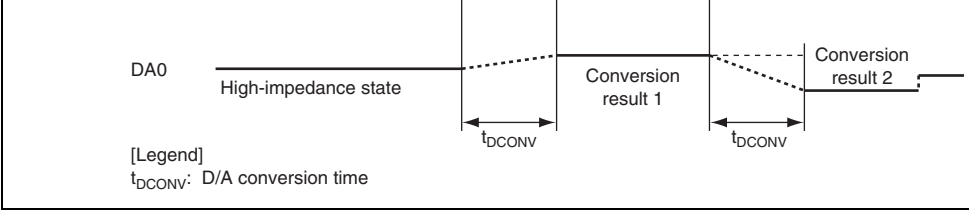
- Formula for 8-bit conversion

$$\frac{\text{Contents of DADR}}{256} \times V_{\text{ref}}$$

- Formula for 10-bit conversion

$$\frac{\text{Contents of DADR}}{1024} \times V_{\text{ref}}$$

3. If DADR0 is written to again, the conversion is immediately started. The conversion output after the conversion time  $t_{\text{DCONV}}$  has elapsed.
4. If the DAOE0 bit is cleared to 0, analog output is disabled.



**Figure 23.2 Example of D/A Converter Operation**



When this LSI makes a transition to software standby mode with D/A conversion enabled, the D/A outputs are retained, and the flow of current from the analog power supply remains as during D/A conversion. If the analog power-supply current has to be reduced in software standby mode, clear the DAOE0, DAOE1, and DAE bits to 0 to disable D/A conversion.

### 23.5.3 Notes on Deep Software Standby Mode

When this LSI makes a transition to deep software standby mode with D/A conversion enabled, the D/A outputs enter high-impedance state.

### 23.5.4 Limitations on Emulators

The limitations described below apply to emulation of the 10-bit D/A converter.

In emulation with the full-spec emulator (E6000H), the resolution of the analog output is only eight bits, and the precision of D/A conversion is not guaranteed. Furthermore, when D/A conversion is read, the values in emulation differ from those for the actual product. Thus, as a precondition for emulation with the full-spec emulator (E6000H), do not read DADR01T.

No particular limitations apply to emulation with the on-chip emulator (E10A-USB).

The above notes are summarized in table 23.3 below.

Analog output: precision

D/A precision is  
guaranteed

As at left

D/A precision is  
guaranteed

---

Flash memory version	H8SX/1662	40 Kbytes	H'FF2000 to H'
	H8SX/1665		
	H8SX/1662M		
	H8SX/1665M		

---



- 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 two memory MATs can be switched by using the bank-switching method after initiation.

— User MAT initiated at a reset in user mode: 384 Kbytes/512 Kbytes

— User boot MAT is initiated at 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 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 guaranteed.)

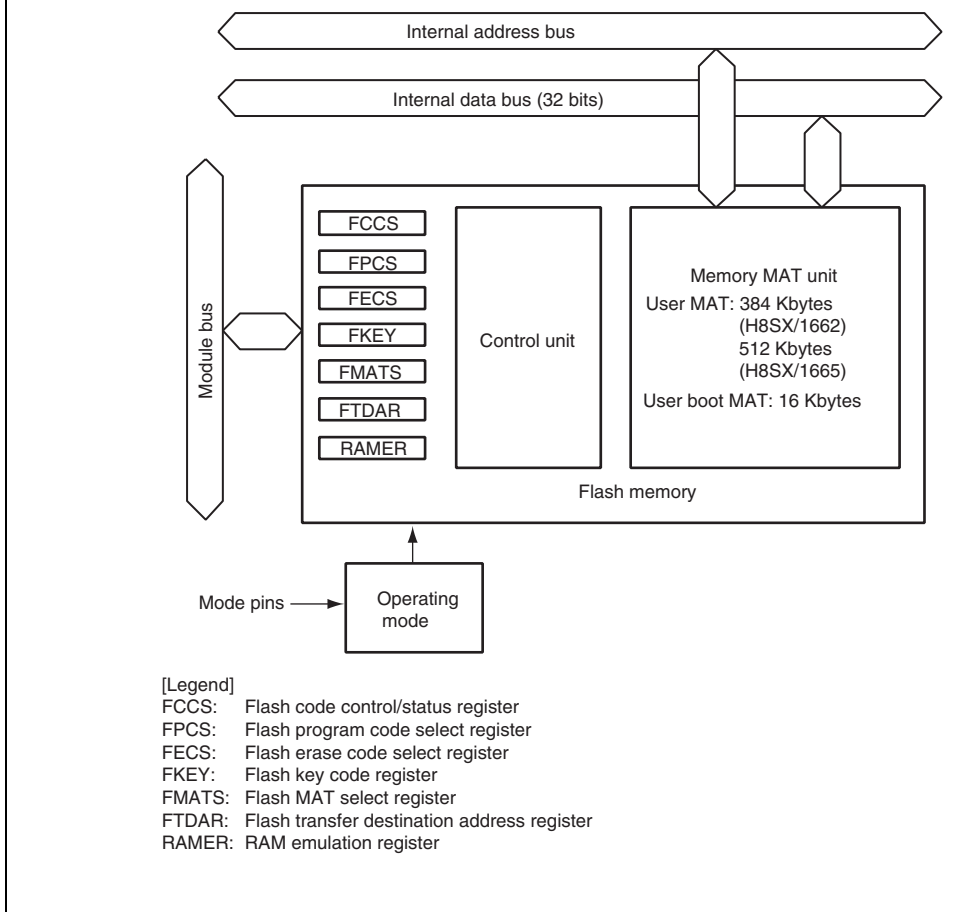
- Three on-board programming modes

SCI Boot mode: Using the on-chip SCI\_4, the user MAT and user boot MAT can be programmed/erased. In SCI boot mode, the bit rate between the host and this LSI can be adjusted automatically.

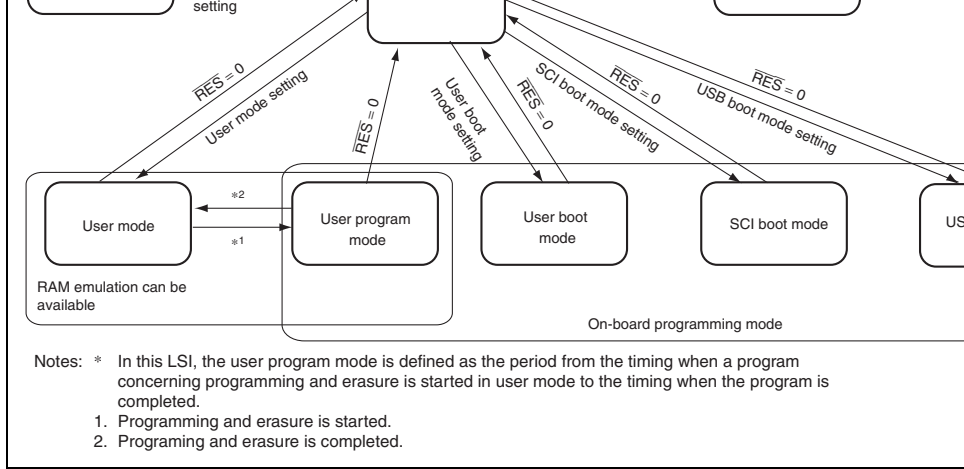
USB boot mode: Using the on-chip USB, the user MAT can be programmed/erased.

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 boot MAT can be programmed/erased.



**Figure 25.1 Block Diagram of Flash Memory**



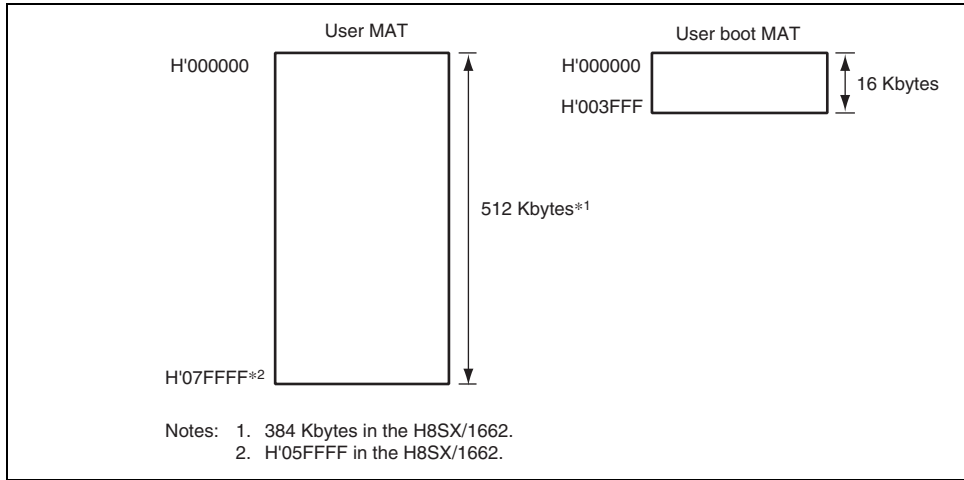
**Figure 25.2 Mode Transition of Flash Memory**

Programming/ erasing control	Command	Command	Programming/ erasing interface	Programming/ erasing interface	Com
All erasure	O (Automatic)	O (Automatic)	O	O	O (Au
Block division erasure	O* <sup>1</sup>	O* <sup>1</sup>	O	O	×
Program data transfer	From host via SCI	From host via USB	From desired device via RAM	From desired device via RAM	Via progra
RAM emulation	×	×	O	O	×
Reset initiation MAT	Embedded program storage area	Embedded program storage area	User MAT	User boot MAT* <sup>2</sup>	—
Transition to user mode	Changing mode and reset	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 that, the flash memory related registers are checked, the reset vector is fetched from the boot MAT.
3. In this LSI, the user programming mode is defined as the period from the timing when the user program concerning programming and erasure is started to the timing when the user program is completed. For details on a program concerning programming and erasure, see section 25.8.3, User Program Mode.



The size of the user MAT is different from that of the user boot MAT. Addresses which the size of the 16-Kbyte user boot MAT should not be accessed. If an attempt is made, it is treated as an undefined value.



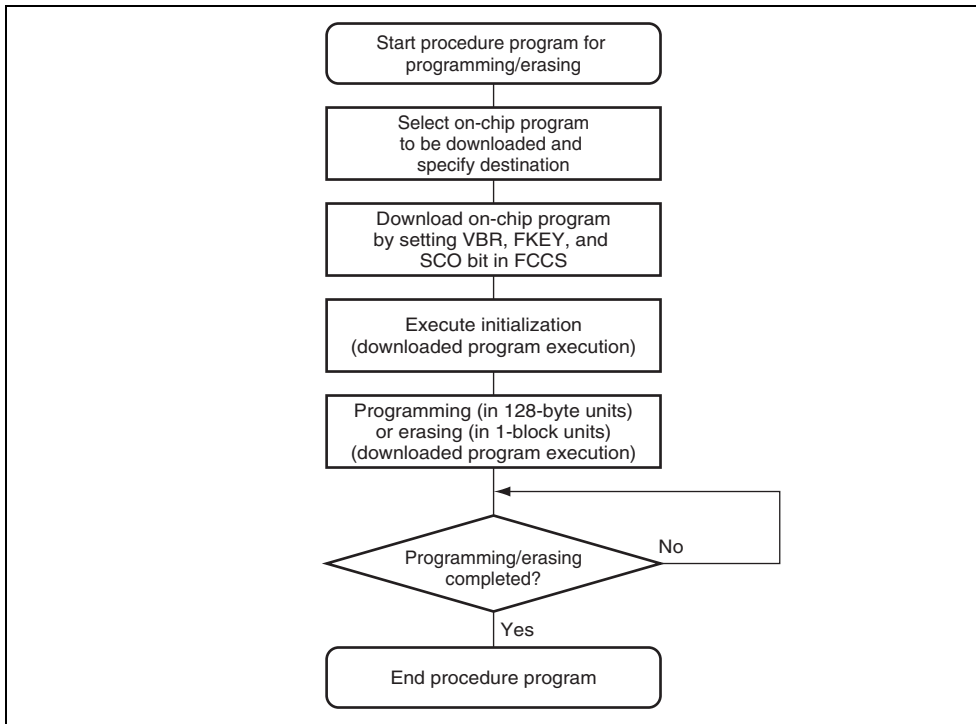
**Figure 25.3 Memory MAT Configuration (H8SX/1665)**

↑ EB0 Erase unit: 4 Kbytes	H'000000	H'000001	H'000002	←Programming unit: 128 bytes→	H'000003
	H'000F80	H'000F81	H'000F82	-----	H'000F83
↓ EB1 Erase unit: 4 Kbytes	H'001000	H'001001	H'001002	←Programming unit: 128 bytes→	H'001003
	H'001F80	H'001F81	H'001F82	-----	H'001F83
↑ EB2 Erase unit: 4 Kbytes	H'002000	H'002001	H'002002	←Programming unit: 128 bytes→	H'002003
	H'002F80	H'002F81	H'002F82	-----	H'002F83
↑ EB3 Erase unit: 4 Kbytes	H'003000	H'003001	H'003002	←Programming unit: 128 bytes→	H'003003
	H'003F80	H'003F81	H'003F82	-----	H'003F83
↑ EB4 Erase unit: 4 Kbytes	H'004000	H'004001	H'004002	←Programming unit: 128 bytes→	H'004003
	H'004F80	H'004F81	H'004F82	-----	H'004F83
↑ EB5 Erase unit: 4 Kbytes	H'005000	H'005001	H'005002	←Programming unit: 128 bytes→	H'005003
	H'005F80	H'005F81	H'005F82	-----	H'005F83
↑ EB6 Erase unit: 4 Kbytes	H'006000	H'006001	H'006002	←Programming unit: 128 bytes→	H'006003
	H'006F80	H'006F81	H'006F82	-----	H'006F83
↑ EB7 Erase unit: 4 Kbytes	H'007000	H'007001	H'007002	←Programming unit: 128 bytes→	H'007003
	H'007F80	H'007F81	H'007F82	-----	H'007F83
↑ EB8 Erase unit: 32 Kbytes	H'008000	H'008001	H'008002	←Programming unit: 128 bytes→	H'008003
	H'00FF80	H'00FF81	H'00FF82	-----	H'00FF83
↑ EB9 Erase unit: 64 Kbytes	H'010000	H'010001	H'010002	←Programming unit: 128 bytes→	H'010003
	H'01FF80	H'01FF81	H'01FF82	-----	H'01FF83
↑ EB10	H'020000	H'020001	H'020002	←Programming unit: 128 bytes→	H'020003
	H'04FF80	H'04FF81	H'04FF82	-----	H'04FF83
↑ EB13 Erase unit: 64 Kbytes	H'050000	H'050001	H'050002	←Programming unit: 128 bytes→	H'050003
	H'05FF80	H'05FF81	H'05FF82	-----	H'05FF83

**Figure 25.4 (1) User MAT Block Structure of H8SX/1662**

EB1 Erase unit: 4 Kbytes	H'000F80	H'000F81	H'000F82	←Programming unit: 128 bytes→	H'000F83
	H'001000	H'001001	H'001002		H'001003
EB2 Erase unit: 4 Kbytes	H'001F80	H'001F81	H'001F82	←Programming unit: 128 bytes→	H'001F83
	H'002000	H'002001	H'002002		H'002003
EB3 Erase unit: 4 Kbytes	H'002F80	H'002F81	H'002F82	←Programming unit: 128 bytes→	H'002F83
	H'003000	H'003001	H'003002		H'003003
EB4 Erase unit: 4 Kbytes	H'003F80	H'003F81	H'003F82	←Programming unit: 128 bytes→	H'003F83
	H'004000	H'004001	H'004002		H'004003
EB5 Erase unit: 4 Kbytes	H'004F80	H'004F81	H'004F82	←Programming unit: 128 bytes→	H'004F83
	H'005000	H'005001	H'005002		H'005003
EB6 Erase unit: 4 Kbytes	H'005F80	H'005F81	H'005F82	←Programming unit: 128 bytes→	H'005F83
	H'006000	H'006001	H'006002		H'006003
EB7 Erase unit: 4 Kbytes	H'006F80	H'006F81	H'006F82	←Programming unit: 128 bytes→	H'006F83
	H'007000	H'007001	H'007002		H'007003
EB8 Erase unit: 4 Kbytes	H'007F80	H'007F81	H'007F82	←Programming unit: 128 bytes→	H'007F83
	H'008000	H'008001	H'008002		H'008003
EB9 Erase unit: 32 Kbytes	H'008F80	H'008F81	H'008F82	←Programming unit: 128 bytes→	H'008F83
	H'00FF80	H'00FF81	H'00FF82		H'00FF83
EB10 Erase unit: 64 Kbytes	H'010000	H'010001	H'010002	←Programming unit: 128 bytes→	H'010003
	H'01FF80	H'01FF81	H'01FF82		H'01FF83
EB15 Erase unit: 64 Kbytes	H'020000	H'020001	H'020002	←Programming unit: 128 bytes→	H'020003
	H'06FF80	H'06FF81	H'06FF82		H'06FF83
	H'070000	H'070001	H'070002	←Programming unit: 128 bytes→	H'070003
	H'07FF80	H'07FF81	H'07FF82		H'07FF83

**Figure 25.4 (2) User MAT Block Structure of H8SX/1665**



**Figure 25.5 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 register. 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).

### **(3) Initialization of Programming/Erase**

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/erasing. The operating frequency of the CPU is set by the programming/erasing interface parameter.

### **(4) Execution of Programming/Erase**

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 128-byte 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. Interrupts are disabled during programming/erasure.

### **(5) When Programming/Erase 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 stored in on-chip RAM even after programming/erasure completes, download and initialization are required when the same processing is executed consecutively.

PM2	Input	SCI boot mode/USB boot mode setting (for boot mode setting by MD3 to MD0)
TxD4	Output	Serial transmit data output (used in SCI boot mode)
RxD4	Input	Serial receive data input (used in SCI boot mode)
USD+, USD-	I/O	USB data I/O (used in USB boot mode)
VBUS	Input	USB cable connection/disconnection detect (used in USB boot mode)
PM3	Input	USB bus power mode/self power mode setting (used in USB boot mode)
PM4	Output	D+ pull-up control (used in USB boot mode)

- 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 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 25.3.

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 bo
  2. The setting may be required according to the combination of initiation mode and target memory MAT.



Bit Name	—	—	—	FLER	—	—	—
Initial Value	1	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

Note: \* This is a write-only bit. This bit is always read as 0.

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	

[Clearing condition]

- At a reset

1: An error occurs during programming/erasing memory (Error protection is valid)

[Setting conditions]

- When an interrupt, such as NMI, occurs during programming/erasure.
- When the flash memory is read during programming/erasure (including a vector read and an instruction fetch).
- When the SLEEP instruction is executed during programming/erasure (including software step mode).
- When a bus master other than the CPU, such as DMAC and DTC, obtains bus mastership during programming/erasure.

---

3 to 1	—	All 0	R	Reserved
--------	---	-------	---	----------

---

These are read-only bits and cannot be modified.

---

immediately after setting this bit to 1. All interrupts be disabled during download. This bit is cleared when download is completed.

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 to H'00000000. After download is completed, contents can be changed.

0: Download of the programming/erasing program requested.

[Clearing condition]

- When download is completed

1: Download of the programming/erasing program requested.

[Setting conditions] (When all of the following are satisfied)

- Not in RAM emulation mode (the RAMS bit RAMER is cleared to 0)
- H'A5 is written to FKEY
- Setting of this bit is executed in the on-chip

---

Note: \* This is a write-only bit. This bit is always read as 0.

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	—	—	—	—	—	—	—	—
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
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.

Bit	Bit Name	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 S
5	K5	0	R/W	FCCS is enabled. When a value other than H'
4	K4	0	R/W	written, the SCO bit cannot be set to 1. Theref
3	K3	0	R/W	on-chip program cannot be downloaded to the
2	K2	0	R/W	RAM.
1	K1	0	R/W	Only when H'5A is written can programming/e
0	K0	0	R/W	the flash memory be executed. When a value
				H'5A is written, even if the programming/erasi
				program is executed, programming/erasure ca
				performed.
				H'A5: Writing to the SCO bit is enabled. (The
				cannot be set to 1 when FKEY is a value
				than H'A5.)
				H'5A: Programming/erasure of the flash memo
				enabled. (When FKEY is a value other t
				the software protection state is entered.
				H'00: Initial value

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 to FMATS.
5	MS5	0/1*	R/W	
4	MS4	0	R/W	When H'AA is written to FMATS, the user boot MAT is selected. When a value other than H'AA is written to FMATS, the user MAT is selected. Switch the MATs following the memory MAT switching procedure in section 23.
3	MS3	0/1*	R/W	
2	MS2	0	R/W	Switching between User MAT and User Boot MAT is not possible. The user boot MAT cannot be selected by FMATS in programming mode. The user boot MAT can be selected in boot mode or programmer mode.
1	MS1	0/1*	R/W	
0	MS0	0	R/W	

H'AA: The user boot MAT is selected. (The user boot 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 user boot mode.)

---

Note: \* This bit is set to 1 in user boot mode, otherwise cleared to 0.

Bit	Bit Name	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 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'FF. If the value is not within the range, the bit is set to 1 when download is executed by setting the bit in FCCS to 1. Make sure that this bit is cleared 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 specified by bits TDA6 to TDA0 is not within the range of H'03 and H'FF and download has stopped.</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. A value between H'00 and H'02 and 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'01: H'FFA000 is specified as the start address.
0	TDA0	0	R/W	<p>H'02: H'FFB000 is specified as the start address.</p> <p>H'03 to H'7F: Setting prohibited.</p> <p>(Specifying a value from H'03 to H'7F, setting the TDER bit to 1 and stops download of the on-chip program.)</p>

processing result is written in R0. The programming/erasing interface parameters are used for download control, initialization before programming or erasing, programming, and erasing. Table 25.4 shows the usable parameters and target modes. The meaning of the bits in the flash programming/erasing fail result parameter (FPFR) varies in initialization, programming, and erasure.

**Table 25.4 Parameters and Target Modes**

Parameter	Download	Initialization	Programming	Erasure	R/W	Initial Value	Allo
DPFR	0	—	—	—	R/W	Undefined	On-chip
FPFR	0	0	0	0	R/W	Undefined	R0L
FPEFEQ	—	0	—	—	R/W	Undefined	ER0
FMPAR	—	—	0	—	R/W	Undefined	ER1
FMPDR	—	—	0	—	R/W	Undefined	ER0
FEBS	—	—	—	0	R/W	Undefined	ER0

Note: \* A single byte of the start address of the on-chip RAM specified by FTDAR

**(a) 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 register. The download pass and fail result parameter (DPFR) indicates the return value.



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 (FMPAA).

The program data is always in 128-byte units. When the program data does not satisfy 128-byte program data is prepared by filling the dummy code (H'FF). The boundary of the programming destination on the user MAT is aligned at an address where 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 user MAT 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 stored in general register ER0. This parameter is called the flash multipurpose data destination address parameter (FMPDR).

For details on the programming procedure, see section 25.8.3, User Program Mode.

#### **(d) Erasure**

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 19 as the erase block number.

For details on the erasing procedure, see section 25.8.3, User Program Mode.

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 type downloaded is not selected, more than two type 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 value. 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 Flash memory. 0: Download of the program has ended normally (no error) 1: Download of the program has ended abnormally (error occurs)

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 L3 cache 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 occurred)

6	MD	—	R/W	<p>Programming Mode Related Setting Error Detect</p> <p>Detects the error protection state and returns the error protection state. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered can be confirmed with the FLER bit in FCCS. For the conditions to enter the error protection state see Section 25.9.3, Error Protection.</p> <p>0: Normal operation (FLER = 0)</p> <p>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 programming ends, 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, before programming the user MAT and user boot MAT have not been written. Programming the user boot MAT should be performed in boot mode or programmer mode.</p> <p>0: Programming has ended normally</p> <p>1: Programming has ended abnormally (programming result is not guaranteed)</p>

When an address not in the flash memory are specified as the start address of the storage destination for the program data, an error occurs.

0: Setting of the start address of the storage destination for the program data is normal

1: Setting of the start address of the storage destination for the program data is abnormal

---

1	WA	—	R/W	Write Address Error Detect
---	----	---	-----	----------------------------

When the following items are specified as the start address of the programming destination, an error occurs.

- An area other than flash memory
- The specified address is not aligned with the byte boundary (lower eight bits of the address other than H'00 and H'80)

0: Setting of the start address of the programming destination is normal

1: Setting of the start address of the programming destination is abnormal

---

0	SF	—	R/W	Success/Fail
---	----	---	-----	--------------

Returns the programming result.

0: Programming has ended normally (no error)

1: Programming has ended abnormally (error)

---

6	MD	—	R/W	<p>Erase Mode Related Setting Error Detect</p> <p>Detects the error protection state and returns the error protection state. When the error protection state is entered, this bit returns to 1. Whether the error protection state is entered can be confirmed with the FLER bit in FCCS. For the conditions to enter the error protection state, see 25.9.3, Error Protection.</p> <p>0: Normal operation (FLER = 0)</p> <p>1: Error protection state, and programming cannot be performed (FLER = 1)</p>
5	EE	—	R/W	<p>Erase Execution Error Detect</p> <p>Returns 1 when the user MAT could not be erased normally when the flash memory related register settings are partially changed. If this bit is set to 1, there is a 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 MAT is selected, an error occurs when erasure is performed. In this case, both the user MAT and the boot MAT have not been erased. Erasing of the boot MAT should be performed in boot mode of the programmer mode.</p> <p>0: Erasure has ended normally</p> <p>1: Erasure has ended abnormally</p>

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 occurred)

#### (4) Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER)

FPEFEQ sets the operating frequency of the CPU. The operating frequency available in ranges from 8 MHz to 50 MHz.

Bit	31	30	29	28	27	26	25	
Bit Name	—	—	—	—	—	—	—	
Bit	23	22	21	20	19	18	17	
Bit Name	—	—	—	—	—	—	—	
Bit	15	14	13	12	11	10	9	
Bit Name	F15	F14	F13	F12	F11	F10	F9	
Bit	7	6	5	4	3	2	1	
Bit Name	F7	F6	F5	F4	F3	F2	F1	

be shown in a number of two decimal places.

2. The value multiplied by 100 is converted to a binary digit and is written to FPEFEQ (general purpose register ERO).

For example, when the operating frequency of the microcontroller is 35.000 MHz, the value is as follows:

1. The number of three decimal places of 35.000 is rounded.
  2. The formula of  $35.00 \times 100 = 3500$  is converted to the binary digit and B'0000 1101 1010 1100 (H'0DAC) is set to ERO.
-



Bit	23	22	21	20	19	18	17	
Bit Name	MOA23	MOA22	MOA21	MOA20	MOA19	MOA18	MOA17	
Bit	15	14	13	12	11	10	9	
Bit Name	MOA15	MOA14	MOA13	MOA12	MOA11	MOA10	MOA9	
Bit	7	6	5	4	3	2	1	
Bit Name	MOA7	MOA6	MOA5	MOA4	MOA3	MOA2	MOA1	

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 start address of the programming destination must be a 128-byte boundary, and MOA6 to MOA0 are always cleared to 0.

Bit	23	22	21	20	19	18	17	
Bit Name	MOD23	MOD22	MOD21	MOD20	MOD19	MOD18	MOD17	M
Bit	15	14	13	12	11	10	9	
Bit Name	MOD15	MOD14	MOD13	MOD12	MOD11	MOD10	MOD9	M
Bit	7	6	5	4	3	2	1	
Bit Name	MOD7	MOD6	MOD5	MOD4	MOD3	MOD2	MOD1	M

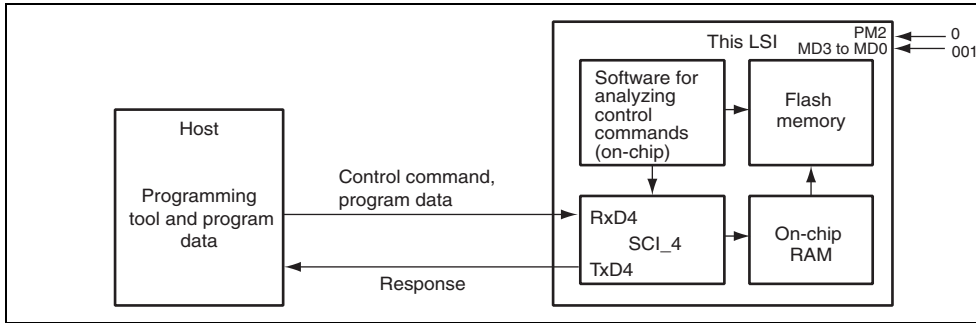
Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MODA0	—	R/W	These bits store the start address of the area w stores the program data for the user MAT. Con 128-byte data is programmed to the user MAT from the specified start address.

Bit	31	30	29	28	27	26	25	
Bit Name								
Initial Value	—	—	—	—	—	—	—	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	23	22	21	20	19	18	17	
Bit Name								
Initial Value	—	—	—	—	—	—	—	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	15	14	13	12	11	10	9	
Bit Name								
Initial Value	—	—	—	—	—	—	—	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Bit	7	6	5	4	3	2	1	
Bit Name								
Initial Value	—	—	—	—	—	—	—	
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	—	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 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 on-chip RAM when RAMS = 1. The following addresses correspond to the 4-Kbyte erase blocks.
0	RAM0	0	R/W	
				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)

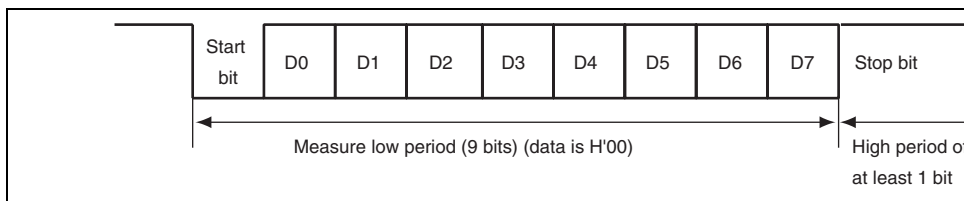
**Table 25.5 On-Board Programming Mode Setting**

<b>Mode Setting</b>	<b>EMLE</b>	<b>MD3</b>	<b>MD2</b>	<b>MD1</b>	<b>MD0</b>	<b>P</b>
User boot mode	0	0	0	0	1	—
SCI boot mode	0	0	0	1	0	0
USB boot mode	0	0	0	1	0	1
User program mode	0	0	1	1	0	—
	0	0	1	1	1	—



**Figure 25.6 System Configuration in Boot Mode**

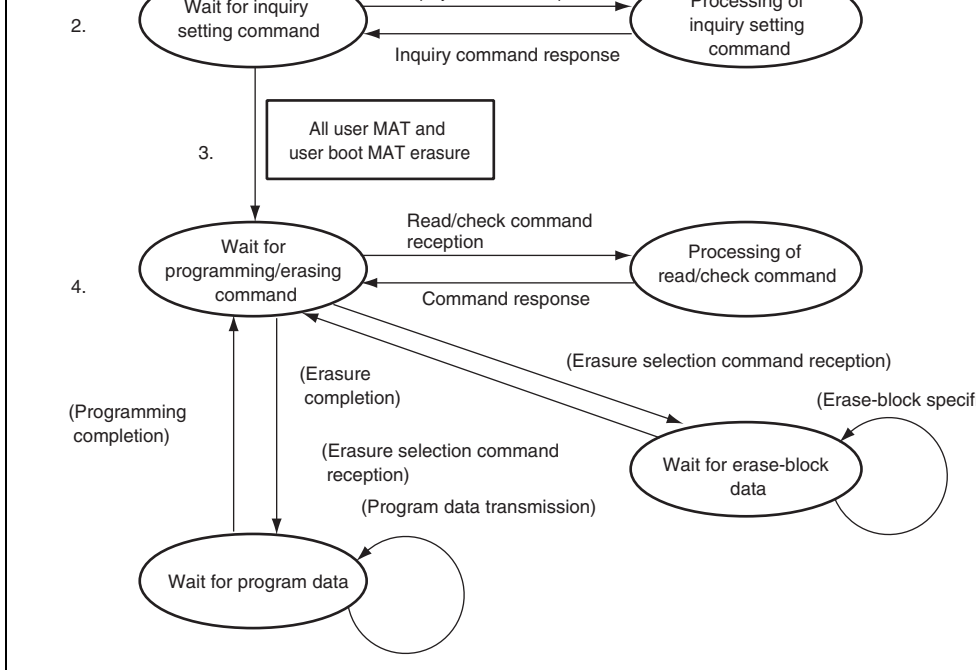
adjustment end sign. When the host receives this bit adjustment end sign normally, it transmits the next byte of H'55 to this LSI. When reception is not executed normally, initiate boot mode again. The transfer bit rate may not be adjusted within the allowable range depending on the combination of the transfer 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 25.6.



**Figure 25.7 Automatic-Bit-Rate Adjustment Operation**

**Table 25.6 System Clock Frequency for Automatic-Bit-Rate Adjustment**

<b>Bit Rate of Host</b>	<b>System Clock Frequency of This LSI</b>
9,600 bps	8 to 18 MHz
19,200 bps	8 to 18 MHz

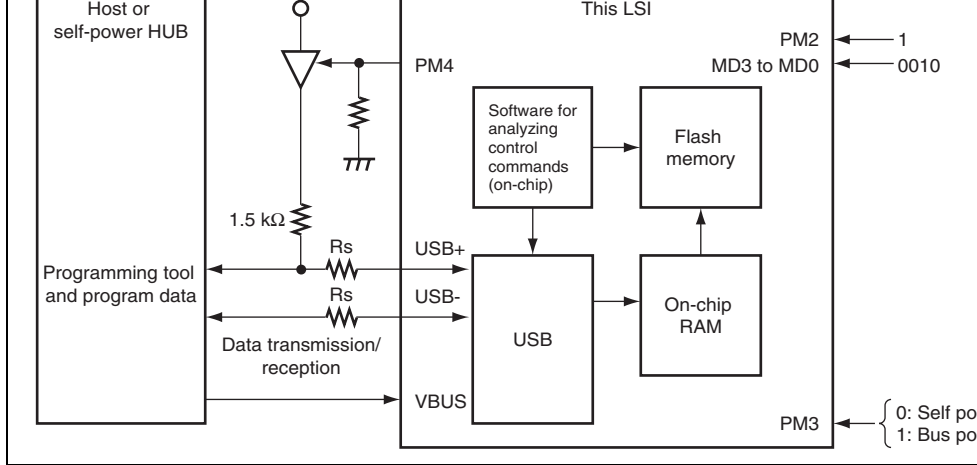


**Figure 25.8 SCI Boot Mode State Transition Diagram**



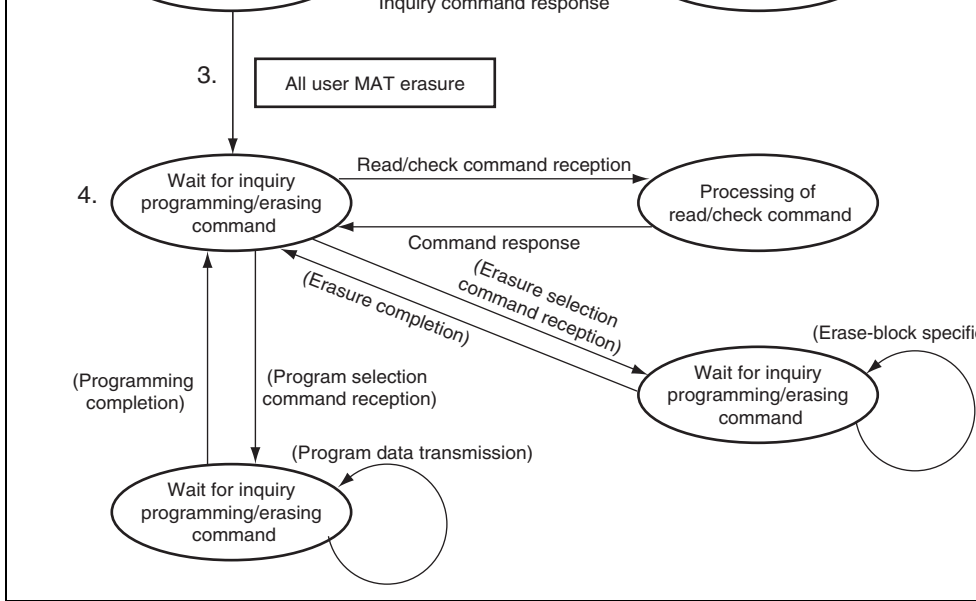
waiting for erase block data is entered. The erase block number must be transmitted when the erasing command is transmitted. When the erasure is finished, the erase block number is set to H'FF and transmitted. Then the state of waiting for erase block data is returned to the state of waiting for programming/erasing command. Erasure must be executed when the specified block is programmed without a reset start after programming is executed in programming mode. When programming can be executed by only one operation, all blocks are erased after entering the state of waiting for programming/erasing command or another command. 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 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 the user MAT/user boot MAT has automatically been erased. No other data can be read.



**Figure 25.9 System Configuration in USB Boot Mode**

	For bus power mode (PM3 = 1)
Endpoint configuration	EP0 Control (in out) 8 bytes Configuration 1 <ul style="list-style-type: none"><li>└─ InterfaceNumber0<ul style="list-style-type: none"><li>└─ AlternateSetting0<ul style="list-style-type: none"><li>└─ EP1 Bulk (out) 64 bytes</li><li>└─ EP2 Bulk (in) 64 bytes</li></ul></li></ul></li></ul>

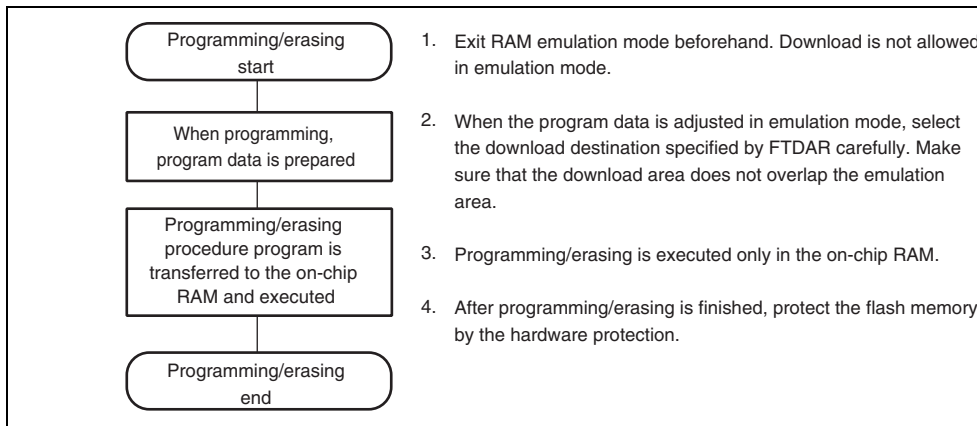


**Figure 25.10 USB Boot Mode State Transition Diagram**

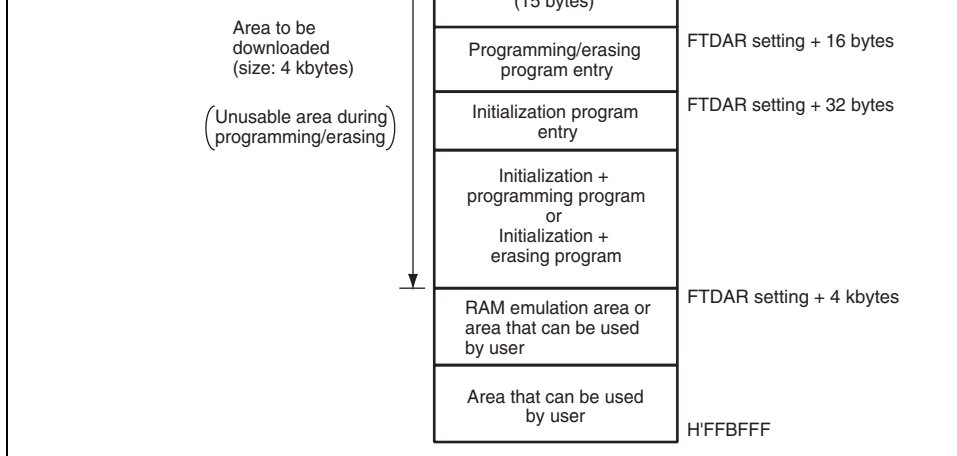
1. After a transition to the USB boot mode is made, the boot program embedded in this is initialized. This LSI performs enumeration to the host after the USB boot program is initialized.
2. Inquiry information about the size, configuration, start address, and support status of the MAT is transmitted to the host.
3. After inquiries have finished, all user MAT are automatically erased.

For details, refer to section 27, Clock Pulse Generator.

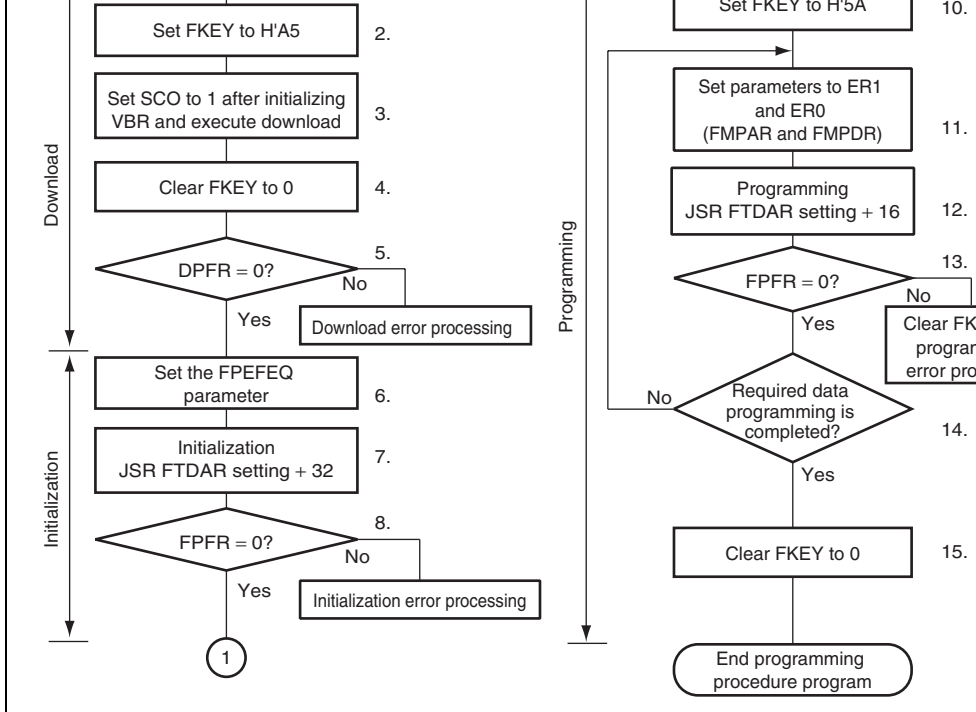
- Use the PM4 pin for the D+ pull-up control connection.
- For the stable supply of the power during the flash memory programming and erasing, the USB cable should not be connected via the bus powered HUB.
- If the bus powered HUB is disconnected during the flash memory programming and erasing, permanent damage to the LSI may result.
- If the USB bus in the bus power mode enters the suspend mode, this does not make a transition to the software standby mode in the power-down state.



**Figure 25.11 Programming/Erasing Flow**



**Figure 25.12 RAM Map when Programming/Erasure is Executed**



**Figure 25.13 Programming Procedure in User Program Mode**



H'FF, the program processing time can be shortened.

1. Select the on-chip program to be downloaded and the download destination. When the FPCS bit in FPCS is set to 1, the programming program is selected. Several programming/programs cannot be selected at one time. If several programs are selected, a download is returned to the SS bit in the DPF0 parameter. The on-chip RAM start address of the 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.

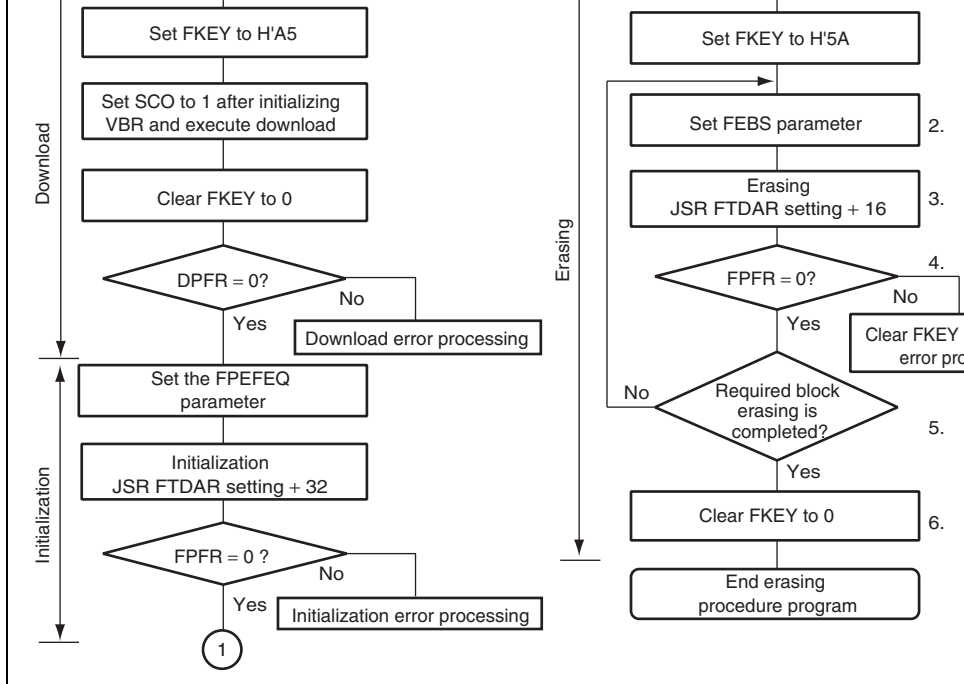
When the SCO bit is set to 1, download is started automatically. Since the SCO bit is cleared to 0 when the procedure program is resumed, the SCO bit cannot be confirmed to be set to 1. To confirm the SCO bit is set to 1, the return value of the FKEY parameter is checked. To prevent incorrect decision, before setting the SCO bit to 1, set one byte to the on-chip RAM start address specified by FTDAR, which becomes the DPF0 parameter. The return value is checked. The return value is a value other than the return value (e.g. H'FF). Since particular processing that is accomplished by bank switching as described below is performed when download is executed, initialize 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.

- If access to the flash memory is requested by the DMAC or DTC during download operation cannot be guaranteed. Make sure that an access request by the DMAC 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 in 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 TDER bit or FK bit in the DPFR parameter to confirm the download program selection and the 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 operating frequency is set otherwise, an error is returned to the FPFRE parameter of the initialization program and initialization is not performed. For details on setting the frequency, see section 25.7.2 (3), Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER00000000 CPU).

- Since the stack area is used in the initialization program, a stack area of 128 bytes maximum must be allocated in RAM.
  - Interrupts can be accepted during execution of the initialization program. Make sure that 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 FPCR 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 1 (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 0. Accordingly, interrupts other than NMI are held and not executed. Configure the user MAT so that NMI interrupts do not occur. The interrupts that are held must be executed after programming completes. When the bus mastership is moved to other than the CPU, 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 address of the program data storage area (FMPDR parameter) is set in general register ER2.
    - 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 in the FPCR 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.

- 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 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 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 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 of  $\overline{\text{RES}} = 0$ ) of at least 100  $\mu\text{s}$ .



**Figure 25.14 Erasing Procedure in User Program Mode**

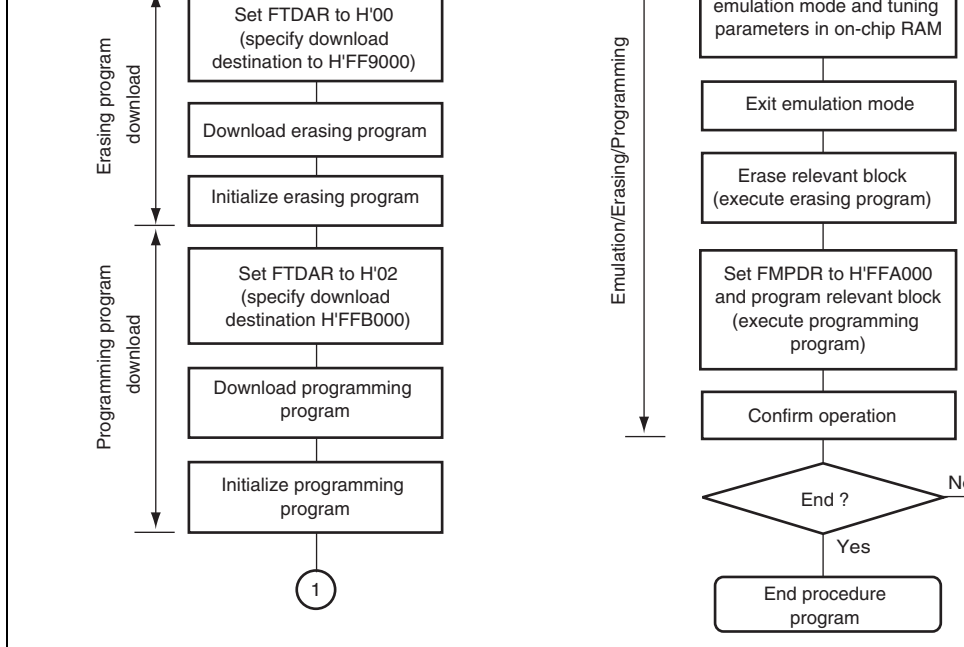
bit in FECS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download program is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.

For the procedures to be carried out after setting FKEY, see section 25.8.3 (2), Programming/Erasing 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 the return value is returned to the FPFPR parameter.
3. Erasure is executed. Similar to as in programming, the entry point of the erasing program is 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 sequence.

```
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.
  - ROL is a return value of the FPFPR 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 FPFPR 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 reset immediately after erasure has finished, secure the reset input period (period of time from the start of reset to the start of program execution) of at least 100  $\mu$ s.



**Figure 25.15 Repeating Procedure of Erasing, Programming, and RAM Emulation in User Program Mode**

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.

## 25.8.4 User Boot Mode

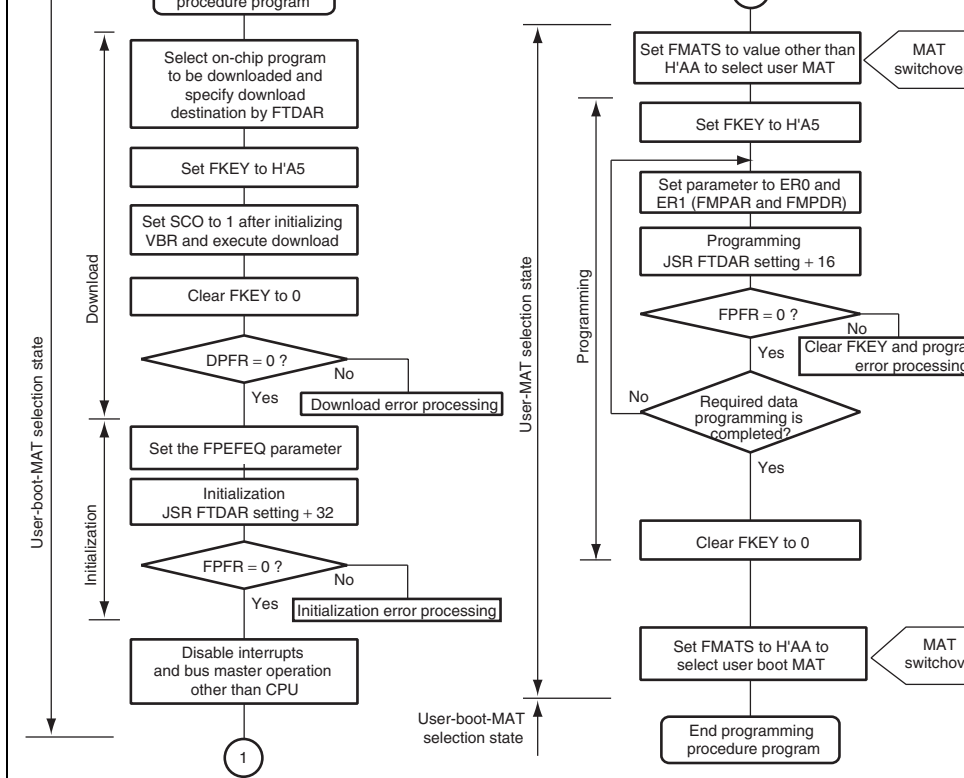
Branching to a programming/erasing program prepared by the user enables user boot mode. This is a user-arbitrary boot mode to be used.

Only the user MAT can be programmed/erased in user boot mode. Programming/erasure of 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.





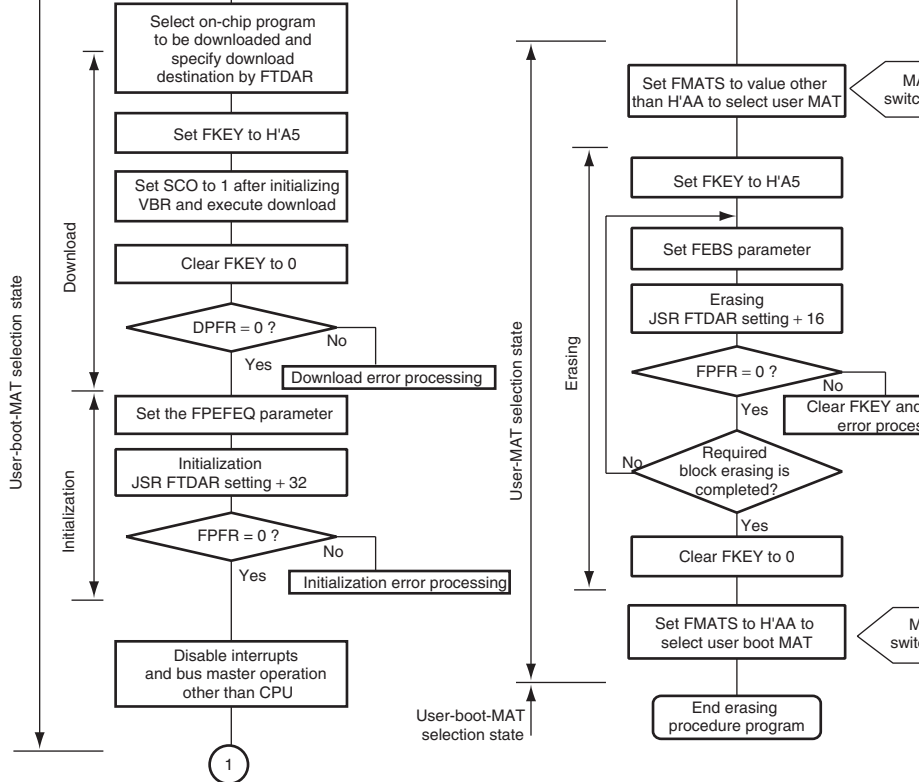
Note: The MAT must be switched by FMATS to perform the programming error processing in the user boot MAT.

**Figure 25.16 Procedure for Programming User MAT in User Boot Mode**

description in section 25.11, 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 program and external space) is shown in section 25.8.5, On-Chip Program and Storable Area for Program Data.



Note: The MAT must be switched by FMATS to perform the erasing error processing in the user boot MAT.

**Figure 25.17 Procedure for Erasing User MAT in User Boot Mode**

### 25.8.5 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.

1. The on-chip program is downloaded to and executed in the on-chip RAM specified by the FTDAR. Therefore, this on-chip RAM area is not available for use.
2. Since the on-chip program uses a stack area, allocate 128 bytes at the maximum as a stack area.
3. 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.
4. 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 is determined).
5. 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 programs 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.
6. After programming/erasure starts, access to the flash memory should be inhibited until the reset signal is cleared. The reset input state (period of  $\overline{RES} = 0$ ) must be set to at least 100  $\mu$ 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  $\overline{RES} = 0$ ) of at least 100  $\mu$ s is needed before the reset signal is released.

executed are determined by the combination of the processing contents, operating mode structure of the memory MATs, as shown in tables 25.8 to 25.12.

**Table 25.8 Executable Memory MAT**

<b>Processing Contents</b>	<b>Operating Mode</b>	
	<b>User Program Mode</b>	<b>User Boot Mode*</b>
Programming	See table 25.9	See table 25.11
Erasing	See table 25.10	See table 25.12

Note: \* Programming/Erasure is possible to the user MAT.

## 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.

Operation for clearing FKEY	0	0	0
Decision of download result	0	0	0
Operation for download error	0	0	0
Operation for setting initialization parameter	0	0	0
Execution of initialization	0	×	0
Decision of initialization result	0	0	0
Operation for initialization error	0	0	0
NMI handling routine	0	×	0
Operation for disabling interrupts	0	0	0
Operation for writing H'5A to FKEY	0	0	0
Operation for setting erasure parameter	0	×	0
Execution of erasure	0	×	0
Decision of erasure result	0	×	0
Operation for erasure error	0	×	0
Operation for clearing FKEY	0	×	0

## 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 programming parameter	○	×	○
Execution of programming	○	×	○
Decision of programming result	○	×	○
Operation for programming error	○	×* <sup>2</sup>	○
Operation for clearing FKEY	○	×	○
Switching memory MATs by FMATS	○	×	○

- 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.



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 the MATs to be used.

**Table 25.13 Hardware Protection**

Item	Description	Function to be Pro	
		Download	Progra 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

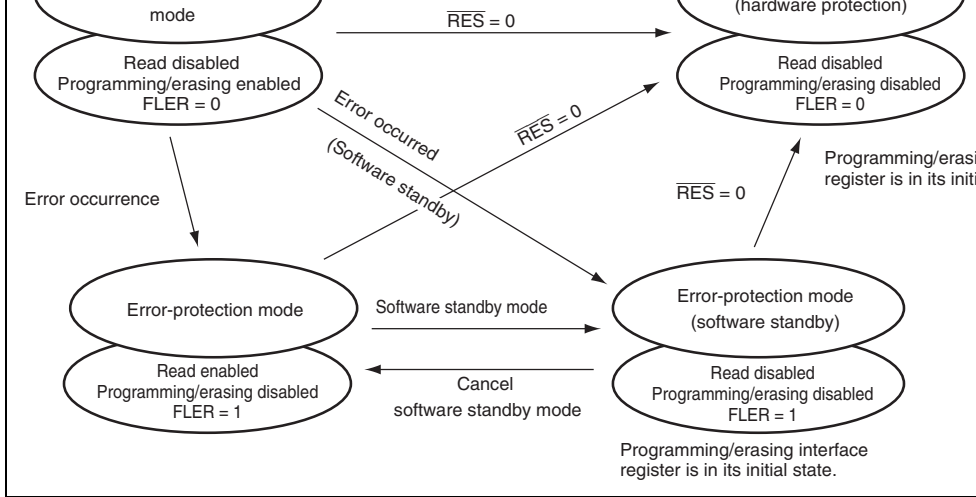
by SCO bit	entered when the SCO bit in FCCS is cleared to 0 to disable download of the programming/erasing programs.		
Protection by FKEY	The programming/erasing protection state is entered because download and programming/erasure are disabled unless the required key code is written in FKEY.	○	○
Emulation protection	The programming/erasing protection state is entered when the RAMS bit in the RAM emulation register (RAMER) is set to 1.	○	○

### 25.9.3 Error Protection

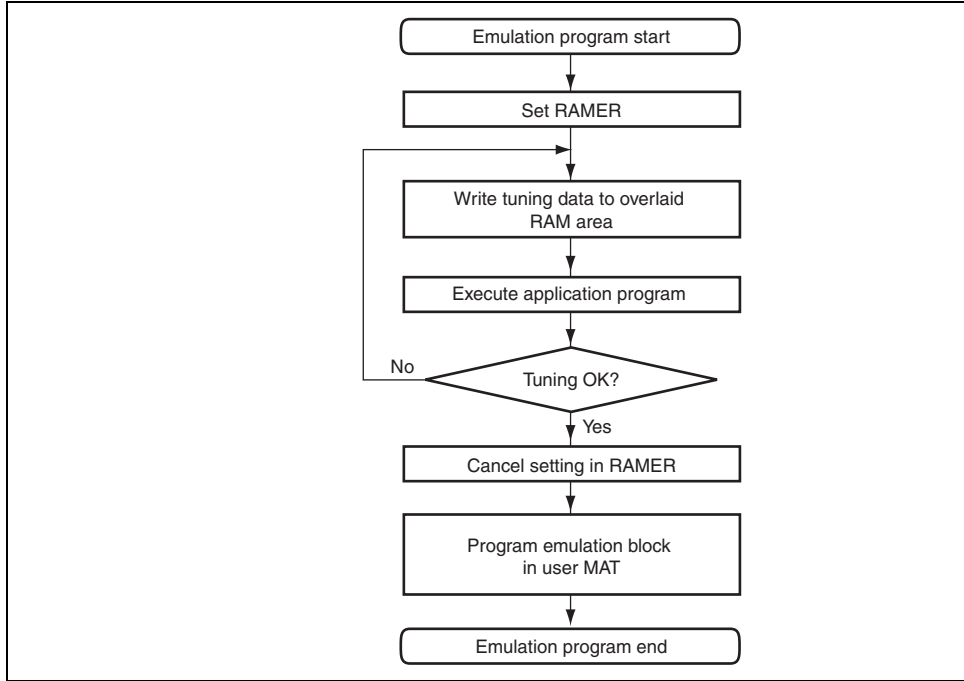
Error protection is a mechanism for aborting programming or erasure when a CPU runa occurs or operations not according to the programming/erasing procedures are detected programming/erasure of the flash memory. Aborting programming or erasure in such ca prevents damage to the flash memory due to excessive programming or erasing.

If an error occurs during programming/erasure of the flash memory, the FLER bit in FC to 1 and the error protection state is entered.

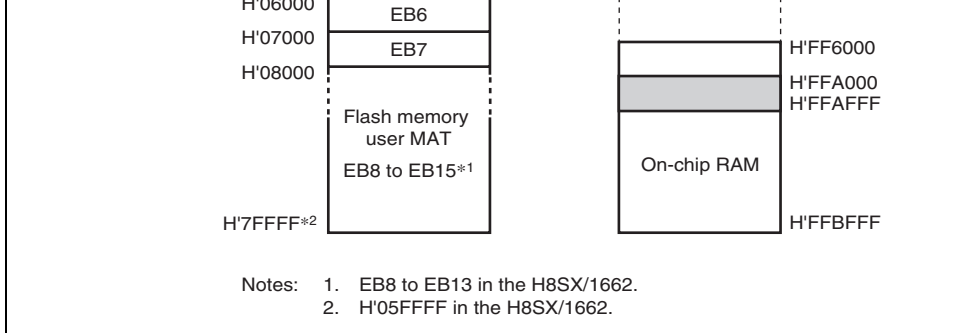
- When an interrupt request, such as NMI, occurs during programming/erasure.
- When the flash memory is read from during programming/erasure (including a vecto an instruction fetch).
- When a SLEEP instruction is executed (including software-standby mode) during programming/erasure.
- When a bus master other than the CPU, such as the DMAC and DTC, obtains bus m during programming/erasure.



**Figure 25.18 Transitions to Error Protection State**



**Figure 25.19 RAM Emulation Flow**

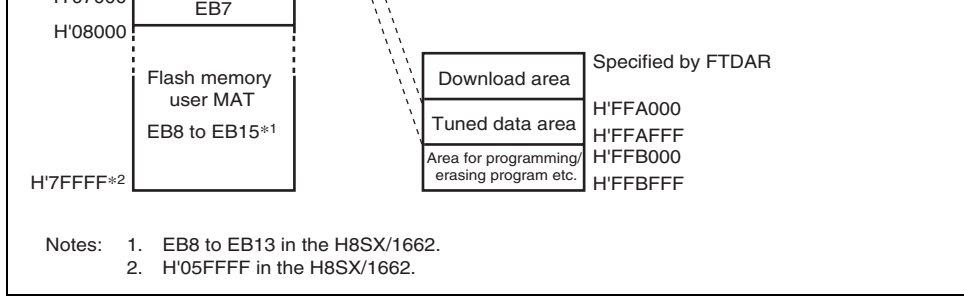


**Figure 25.20 Address Map of Overlaid RAM Area (H8SX/1665)**

The flash memory area that can be emulated is the one area selected by bits RAM2 to 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 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.

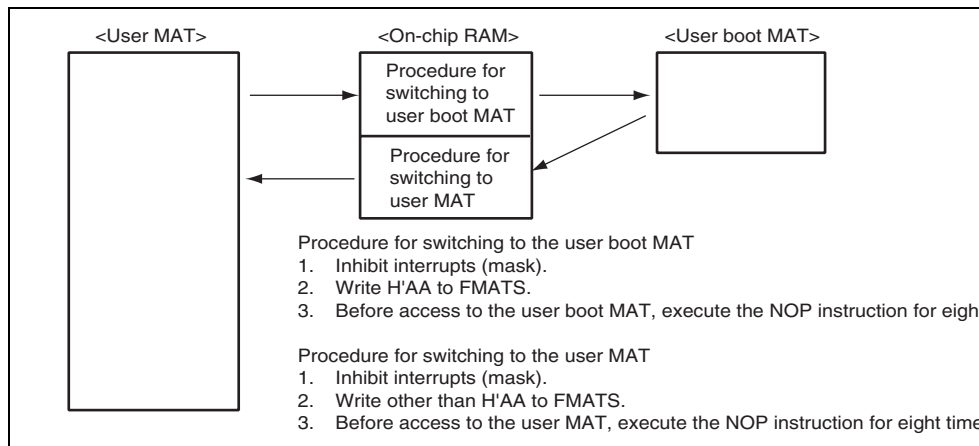


**Figure 25.21 Programming Tuned Data (H8SX/1665)**

1. After tuning program data is completed, clear the RAMS bit in RAMER to 0 to cancel 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 address of the download destination should be specified by FTDAR so that the tuned data 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 values 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.

- 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 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 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 within the size of the 16-Kbyte user boot MAT should not be accessed. If an attempt is made to read as an undefined value.



**Figure 25.22 Switching between User MAT and User Boot MAT**



	H8SX/1665	512 Kbytes	
User boot MAT	H8SX/1662	16 Kbytes	FZTATUSBT
	H8SX/1665		

## 25.13 Standard Serial Communications Interface Specifications for Mode

The boot program initiated in boot mode performs serial communications using the host chip SCI\_4. The serial communications interface specifications are shown below.

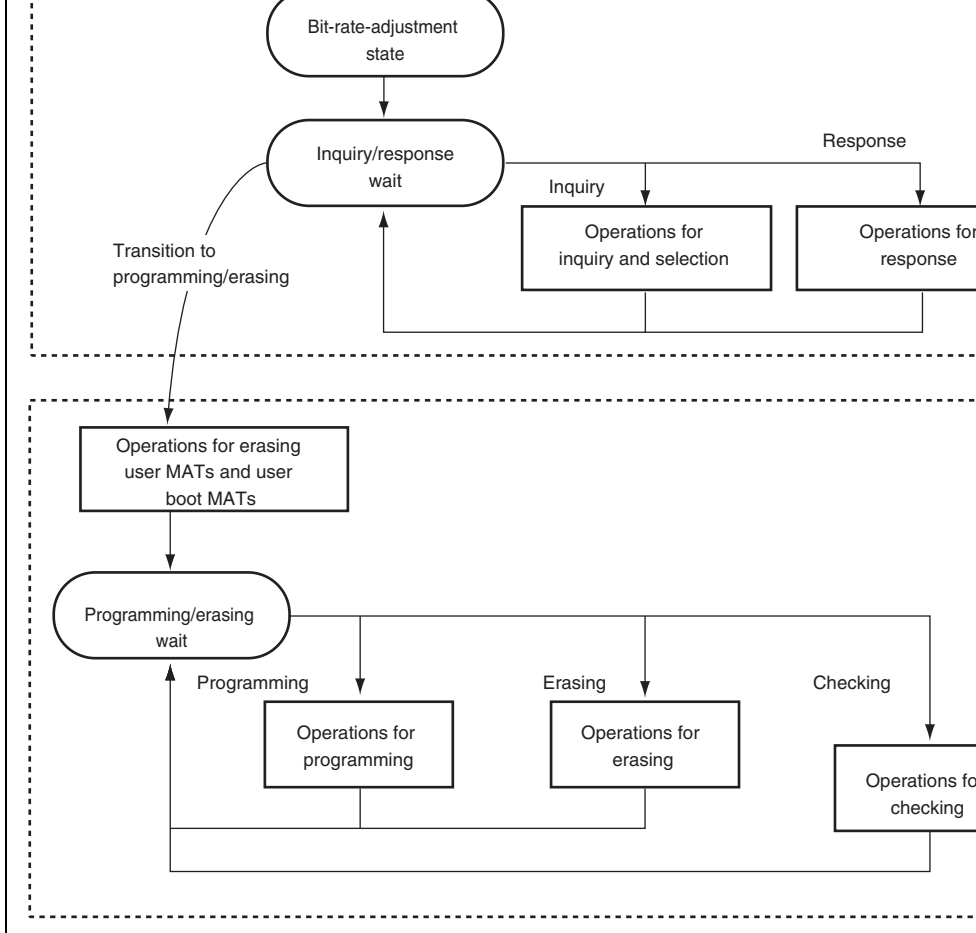
The boot program has three states.

### 1. Bit-rate-adjustment state

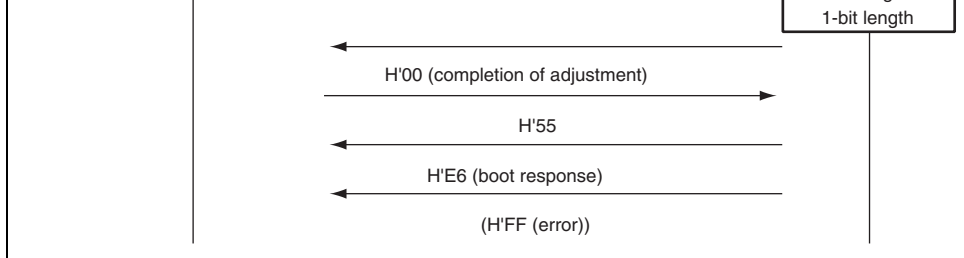
In this state, the boot program adjusts the bit rate to achieve serial communications with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, 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 clock mode, and bit rate are selected. After selection of these settings, the program is instructed 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 chip RAM and erases the user MATs and user boot MATs before the transition.



**Figure 25.23 Boot Program States**



**Figure 25.24 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 A responses to successful completion.

### 2. n-byte commands or n-byte responses

These commands and responses are comprised of n bytes of data. These are selection responses to inquiries.

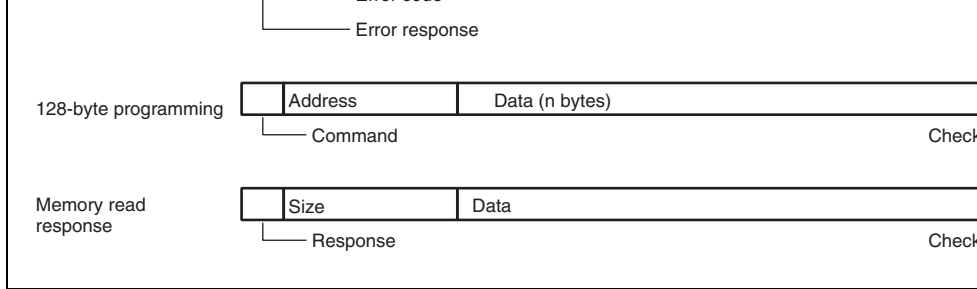
The program data size is not included under this heading because it is determined in response to the command.

### 3. Error response

The error response is a response to inquiries. It consists of an error response and an error response 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.



**Figure 25.25 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

H'10	Device selection	Selection of device code
H'21	Clock mode inquiry	Inquiry regarding numbers of clock 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 multiple
H'23	Operating clock frequency inquiry	Inquiry regarding the maximum and minimum values of the main clock and peripheral clock
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 programming
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 entry to programming/erasing state
H'4F	Boot program status inquiry	Inquiry into the operated status of the boot program

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 contributed by the number of devices, characters, 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 the 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 and response, excluding the SUM byte becomes H'00.

- 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 (two bytes): Values of the supported clock modes (i.e. H'01 means clock mode 1)
  - H'00: MD\_CLK = 0 (8 to 18 MHz input)
  - H'01: MD\_CLK = 1 (16 MHz input)
- SUM (one byte): Checksum

- 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.



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 types of multiplication ratios, and the multiplication ratios
- Number of types of multiplication (one byte): The number of types of multiplication the device can be set  
(e.g. when there are two multiplied clock types, which are the main and peripheral clock, the number of types will be H'02.)
- Number of multiplication ratios (one byte): The number of types 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-frequency is divided by two, the value of division ratio will be H'FE. H'FE = [-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 types of multiplication.
- SUM (one byte): Checksum

...	
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 value in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 17.00 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

- 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 address 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

Response	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
- 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
- Programming unit (two bytes): A unit for programming  
This is the unit for reception of programming.
- SUM (one byte): Checksum

- Size (one byte): The number of bytes that represents the bit rate, input frequency, number of types of multiplication, and multiplication ratio
- 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'0C0.)
- 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. When the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Number of types of multiplication (one byte): The number of multiplication to which the device can be set.
- Multiplication ratio 1 (one byte): The value of multiplication or division ratios for the core operating frequency  
Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the core 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 core 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 peripheral 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 peripheral frequency 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.

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, a multiplication or division frequency error is generated.

##### 3. Operating frequency error

Operating frequency is calculated from the received value of the input frequency and 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  $\times$  Multiplication ratio, or

Operating frequency = Input frequency  $\div$  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.

When the new bit rate is selectable, the rate will be set in the register after sending ACK response. The host will send an ACK with the new bit rate for confirmation and the boot will response with that rate.

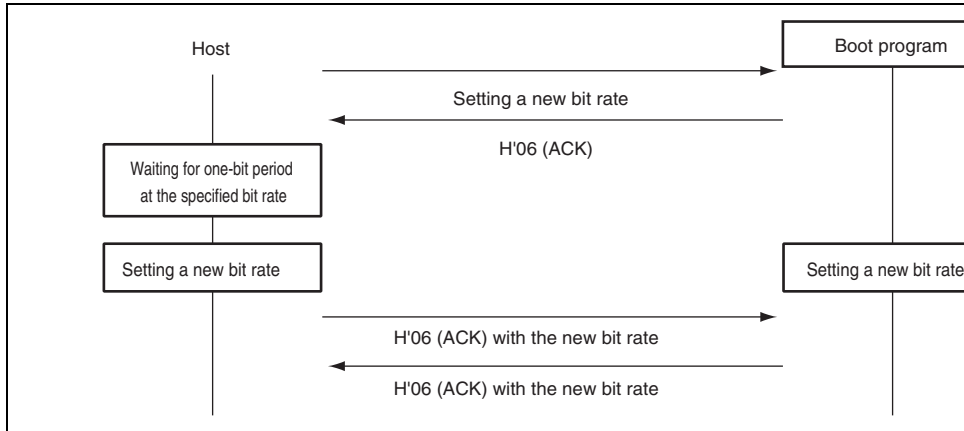
Confirmation H'06

- 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 25.26.



**Figure 25.26 New Bit-Rate Selection Sequence**

- 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 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 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



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.

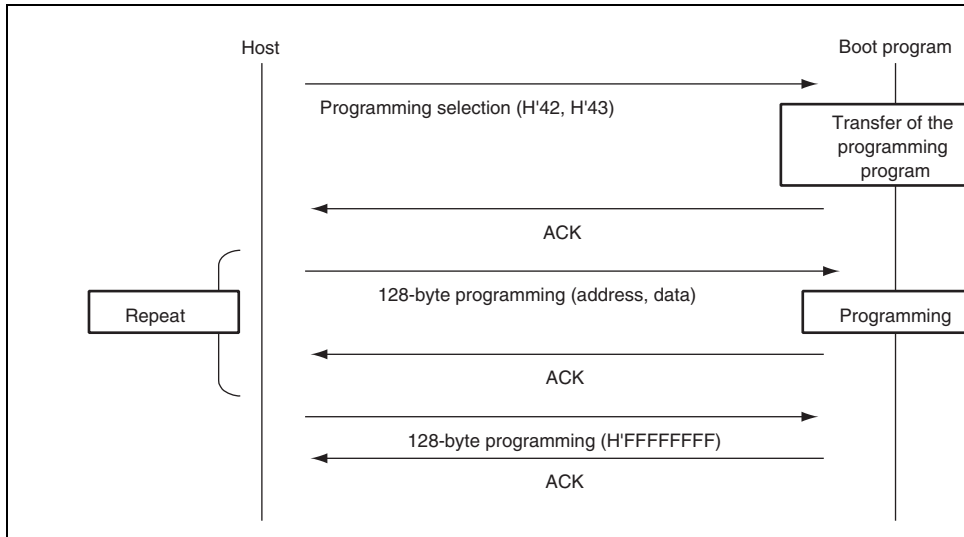
6. A new bit rate should be selected with the new bit-rate selection (H'3F) command, and the returned information on multiplication ratios and operating frequencies.
7. After selection of the device and clock mode, the information of the user boot MATs should be made to inquire about the user boot MATs information inquiry (H'24), MATs information inquiry (H'25), erased block information inquiry (H'26), and program 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.

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

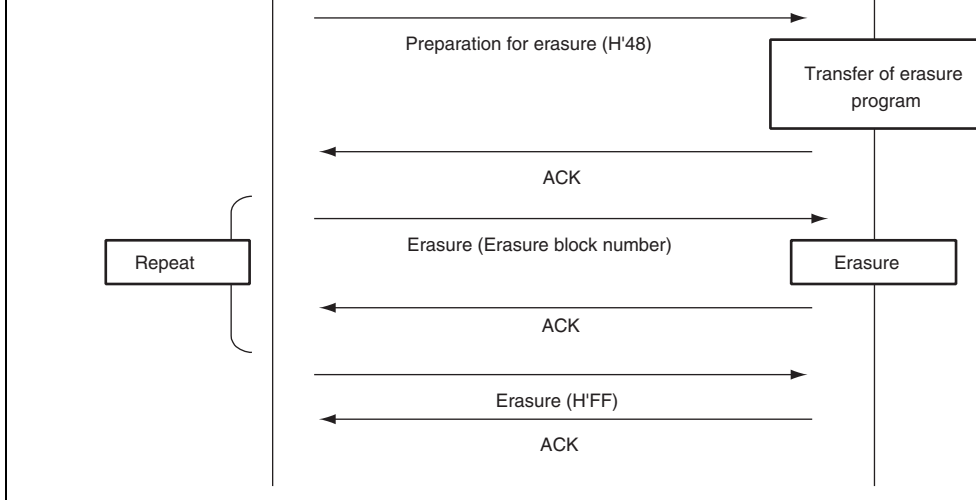
command represents the data programmed according to the method specified by the command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H'FFFFFFF address will stop the programming. On completion of programming, the boot program wait for selection of programming or erasing.

Where the sequence of programming operations that is executed includes programming 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 in figure 25.27.



**Figure 25.27 Programming Sequence**



**Figure 25.28 Erasure Sequence**

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 MAT programming selection

Response 

H'06
------

- Response, H'06, (one byte): Response to user MAT programming selection  
When the programming program has been transferred, the boot program will return a response.

Error Response 

H'C3	ERROR
------	-------

- Error response : H'C3 (1 byte): Error response to user MAT programming selection
- ERROR : (1 byte): Error code  
H'54: Selection processing error (transfer error occurs and processing is not completed)

- 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 programming is in 128-byte units, the lower eight bits of the address should be H'00 or H'01. 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 programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.

- 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 completed.

**(d) Erasure Selection**

The boot program will transfer the erasure program. User MAT data is erased by the transfer of the 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)

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 select 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.



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.

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 MAT 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

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.

- 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 25.18 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)

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	Erasure error
H'52	Erasure incomplete error
H'53	Programming error
H'54	Selection processing error
H'80	Command error
H'FF	Bit-rate-adjustment confirmation error

3.3-V programming voltage. Use only the specified socket adapter.

5. Do not turn off the Vcc power supply nor remove the chip from the PROM programmer during programming/erasure in which a high voltage is applied to the flash memory. Doing so will permanently damage the flash memory. If a reset is input, the reset must be released after a reset input period of at least 100ms.
6. The flash memory is not accessible until FKEY is cleared after programming/erasure. After 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  $\overline{\text{RES}} = 0$ ) at 100 $\mu$ s. Transition to the reset state during programming/erasure is inhibited. If a reset is input during the reset, the reset must be released after the reset input period of at least 100 $\mu$ s.
7. At powering on the Vcc power supply, fix the  $\overline{\text{RES}}$  pin to low and set the flash memory to hardware protection state. This power-on procedure must also be satisfied at a 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/erasure in on-board programming mode, it is recommended that automatic programming be 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 clock frequency is 35 MHz, the download for each program takes approximately 60  $\mu$ s at the maximum.

Immediately after setting it to 1. Otherwise, download cannot be performed normally.  
must be executed twice.

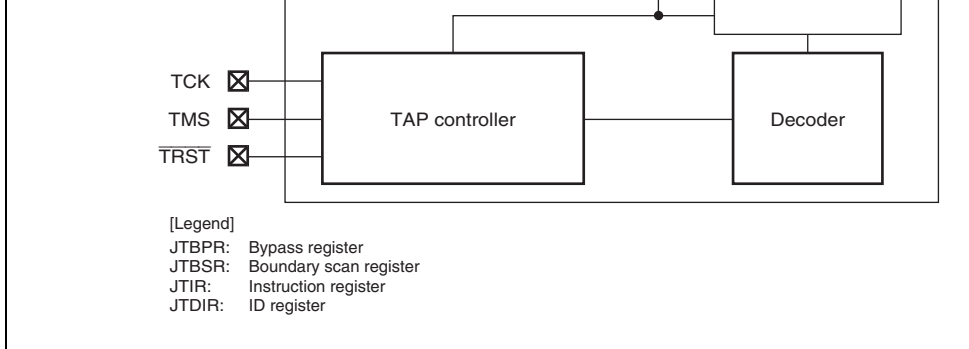
15. The contents of general registers ER0 and ER1 are not saved during download of a program, initialization, programming, or erasure. When needed, save the general registers before a download request or before execution of initialization, programming, or erasure of the procedure program.





valid

- Six test modes:  
BYPASS mode  
EXTEST mode  
SAMPLE/PRELOAD mode  
CLAMP mode  
HIGHZ mode  
IDCODE mode



**Figure 26.1 Block Diagram of Boundary Scan Function**

## 26.3 Input/Output Pins

Table 26.1 shows the I/O pins used in the boundary scan function.

**Table 26.1 Pin Configuration**

Pin Name	I/O	Description
TCK	Input	Test clock input pin Clock signal for boundary scan. Input the clock the duty cycle of which is 50 percent when boundary scan function is used.
TMS	Input	Test mode select pin
TDI	Input	Test data input pin
TDO	Output	Test data output pin
$\overline{\text{TRST}}$	Input	Test reset input pin

TDI and TDO pins in BYPASS mode. The boundary scan register (JTBSR), which is a . . . register (see table 26.4), is connected between the TDI and TDO pins when test data are shifted in. None of the registers is accessible from the CPU.

Table 26.2 shows the availability of serial transfer for the registers.

**Table 26.2 Serial Transfers for Registers**

<b>Register Abbreviation</b>	<b>Serial Input</b>	<b>Serial Output</b>
JTIR	Available	Not available
JTBPR	Available	Available
JTBSR	Available	Available
JTID	Not available	Available

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

Bit	Bit Name	Initial Value	R/W	Descriptions
15 to 12	TS[3:0]	All 0	R/W	Test Bit Set Specify an instruction as shown in table 26.3.
11 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value is always 0.

0	1	1	1	Reserved
1	0	0	0	Reserved
1	0	0	1	Reserved
1	0	1	0	Reserved
1	0	1	1	Reserved
1	1	0	0	Reserved
1	1	0	1	Reserved
1	1	1	0	Reserved
1	1	1	1	BYPASS

### 26.4.2 Bypass Register (JTBPR)

JTBPR is a 1-bit register and is connected between the TDI and TDO pins when JTIR is in BYPASS mode. JTBPR cannot be read from or written to by the CPU.

### 26.4.3 Boundary Scan Register (JTBSR)

JTBSR is a shift register to control the external input and output pins of this LSI and is connected across the pads. The initial values are undefined. JTBSR cannot be accessed by the CPU. EXTEST, SAMPLE/PRELOAD, CLAMP, and HIGHZ instructions are issued to apply boundary-scan testing conformant to the JTAG standard.

Table 26.4 shows the correspondence between the JTBSR bits and the pins of this LSI.

			Output	290
7	D1	MD2	Input	289
8	D3	PM0	Input	288
			Output enable	287
			Output	286
9	D2	PM1	Input	284
			Output enable	283
			Output	282
10	E1	PM2	Input	280
			Output enable	279
			Output	278
11	E4	PF7	Input	274
			Output enable	273
			Output	272
12	E3	PF6	Input	271
			Output enable	270
			Output	269
13	E2	PF5	Input	268
			Output enable	267
			Output	266
14	F1	PF4	Input	265
			Output enable	264
			Output	263

			Output
18	G1	PF1	Input
			Output enable
			Output
19	H4	PF0	Input
			Output enable
			Output
20	G3	PE7	Input
			Output enable
			Output
21	G2	PE6	Input
			Output enable
			Output
22	H1	PE5	Input
			Output enable
			Output
24	H2	PE4	Input
			Output enable
			Output
26	H3	PE3	Input
			Output enable
			Output

			Output enable	228				Output enable
			Output	227				Output
30	J3	PD7	Input	226	46	L4	VBUS	Input
			Output enable	225	47	L5	MD_CLK	Input
			Output	224	49	M6	P20	Input
31	K4	PD6	Input	223				Output enable
			Output enable	222				Output
			Output	221	51	K6	P21	Input
33	K2	PD5	Input	220				Output enable
			Output enable	219				Output
			Output	218	52	N6	P22	Input
34	K3	PD4	Input	217				Output enable
			Output enable	216				Output
			Output	215	53	M7	P23	Input
35	L1	PD3	Input	214				Output enable
			Output enable	213				Output
			Output	212	54	L6	P24	Input
36	M1	PD2	Input	211				Output enable
			Output enable	210				Output
			Output	209	55	K7	P25	Input
37	M2	PD1	Input	208				Output enable
			Output enable	207				Output
			Output	206				

			Output enable	158				Output enable
			Output	157				Output
59	K8	P26	Input	156	71	N11	PH5	Input
			Output enable	155				Output enable
			Output	154				Output
60	N8	P27	Input	153	72	N12	PH6	Input
			Output enable	152				Output enable
			Output	151				Output
61	L9	NMI	Input	150	76	L13	PI1	Input
62	L8	P33	Input	149				Output enable
			Output enable	148				Output
			Output	147	77	L11	PI2	Input
63	K9	P34	Input	146				Output enable
			Output enable	145				Output
			Output	144	78	L12	PI3	Input
65	M9	PH0	Input	143				Output enable
			Output enable	142				Output
			Output	141	75	M13	PI0	Input
66	L10	PH1	Input	140				Output enable
			Output enable	139				Output
			Output	138	80	K11	PI4	Input
67	K10	PH2	Input	137				Output enable
			Output enable	136				Output
			Output	135				

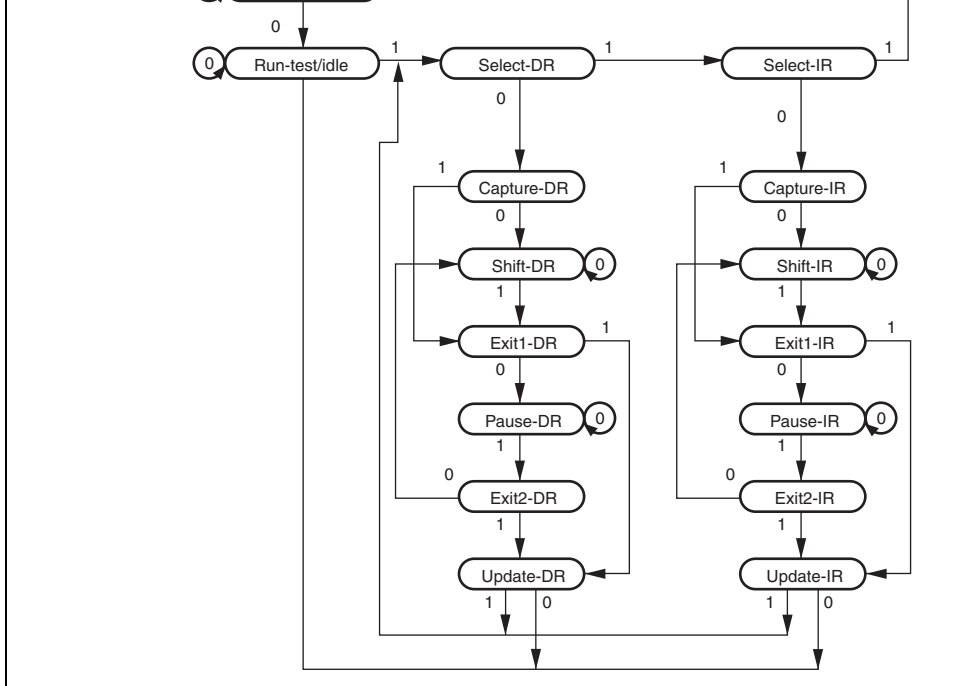


			Output enable	97				Output enable
			Output	96				Output
84	J11	P10	Input	95	107	B13	P60	Input
			Output enable	94				Output enable
			Output	93				Output
85	H11	P11	Input	92	106	C13	P37	Input
			Output enable	91				Output enable
			Output	90				Output
86	J12	P12	Input	89	108	A13	P61	Input
			Output enable	88				Output enable
			Output	87				Output
87	G11	P13	Input	86	115	D10	MD0	Input
			Output enable	85	116	C10	PC2	Input
			Output	84				Output enable
93	G10	P14	Input	77				Output
			Output enable	76	117	B10	PC3	Input
			Output	75				Output enable
94	F11	P15	Input	74				Output
			Output enable	73	129	B6	MD1	Input
			Output	72	130	C7	PB4	Input
100	E11	P16	Input	71				Output enable
			Output enable	70				Output
			Output	69				

134	D6	PA0	Input	32				Output enable
			Output enable	31				Output
			Output	30	144	B2	PB0	Input
135	A5	PA1	Input	29				Output enable
			Output enable	28				Output
			Output	27	1	A1	PB1	Input
136	B4	PA2	Input	26				Output enable
			Output enable	25				Output
			Output	24	2	B1	PB2	Input
137	D5	PA3	Input	23				Output enable
			Output enable	22				Output
			Output	21	to TDO			
138	A4	PA4	Input	20				
			Output enable	19				
			Output	18				

Initial Value —  
 R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W R/W

Bit	Bit Name	Initial Value	R/W	Descriptions
31 to 0	DID31 to DID0	H'0807F447	—	JTID is a register the value showing the decision is fixed.



**Figure 26.2 State Transitions of the TAP Controller**

**EXTEST (Instruction Code: B'0000):** The EXTEST instruction is used to test external pins when this LSI is installed on the printed circuit board. If this instruction is executed, output pins are used to output test data (specified by the SAMPLE/PRELOAD instruction) from the scan register to the print circuit board, and input pins are used to input test result.

**SAMPLE/PRELOAD (Instruction Code: B'0100):** The SAMPLE/PRELOAD instruction is used to input data from the LSI internal circuits to the boundary scan register, output data from the scan path, and reload the data to the scan path. While this instruction is executed, input signals are directly input to the LSI and output signals are also directly output to the external circuit. The system circuit is not affected by this function.

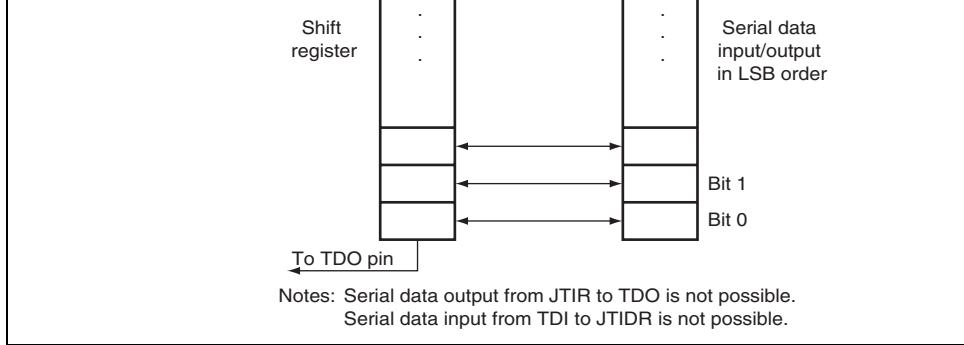
In SAMPLE operation, the boundary scan register latches the snap shot of data transferred from input pins to internal circuit or data transferred from internal circuit to output pins. The test data is read from the scan path. The scan register latches the snap data at the rising edge of TCK in Capture-DR state. The scan register latches snap shot without affecting the LSI normal operation.

In PRELOAD operation, initial value is written from the scan path to the parallel output pins of the boundary scan register prior to the EXTEST instruction execution. If the EXTEST instruction is executed without executing this PRELOAD operation, undefined values are output from the beginning to the end (transfer to the output latch) of the EXTEST sequence. (In EXTEST instruction, parallel latches are always output to the output pins.)

**IDCODE (Instruction Code: B'0001):** When the IDCODE instruction is selected, IDCODE register value is output to the TDO in Shift-DR state of the TAP controller. In this case, IDCODE register value is output from the LSB. During this instruction execution, test circuit does not affect the system circuit. INSTR is initialized by the IDCODE instruction in Test-Logic-Reset state of the TAP controller.

register is maintained regardless of the state of the TAP controller.

BYPASS is connected between TDI and TDO pins, leading to the same operation as when the BYPASS instruction has been selected.



**Figure 26.3 Serial Data Input/Output**

2. If a pin with open-drain function is SAMPLEed while its open-drain function is enabled while the corresponding OUT register is set to 1, the corresponding Control register is cleared to 0 (the pin status is Hi-Z). If the pin is SAMPLEed while the corresponding OUT register is cleared to 0, the corresponding Control register is 1 (the pin status is 0)
3. Pins of the boundary scan (TCK, TDI, TMS, and  $\overline{\text{TRST}}$ ) have to be pulled up by pull-up resistors.
4. Power supply pins (Vcc, VCL, Vss, AVcc, AVss, Vref, PLLVcc, PLLVss, DrVcc, and DrVss) cannot be boundary-scanned.
5. Clock pins (EXTAL and XTAL) cannot be boundary-scanned.
6. Reset and standby signals ( $\overline{\text{RES}}$  and  $\overline{\text{STBY}}$ ) cannot be boundary-scanned.
7. Boundary scan pins (TCK, TMS,  $\overline{\text{TRST}}$ , TDI, and TDO) cannot be boundary-scanned.
8. The boundary scan function is not available when this LSI is in the following states.
  - (1) Reset state
  - (2) Hardware standby mode, software standby mode, and deep software standby mode





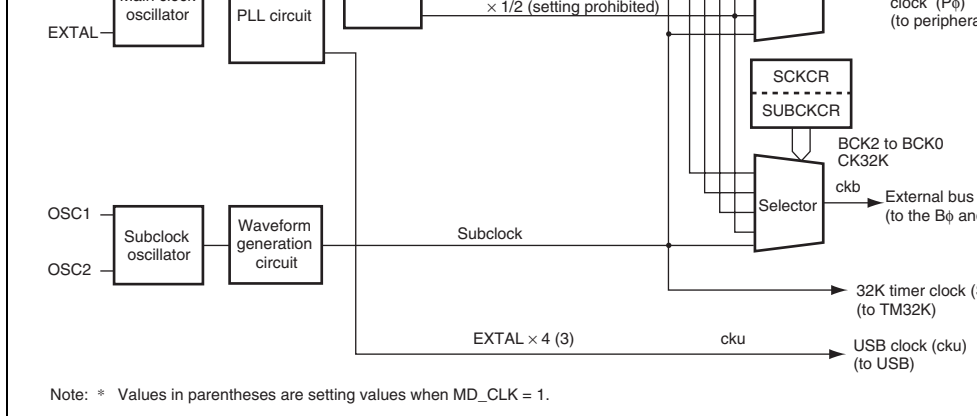
changes the frequency through the setting of the system clock control register (SCKCR) subclock control register (SUBCKCR).

This LSI supports five clocks: a system clock provided to the CPU and bus masters, a peripheral module clock provided to the peripheral modules, an external bus clock provided to the external bus, a 32K timer clock, and a USB clock provided to the USB module. 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 operate with the frequency less than the system clock frequency.

The system clock, peripheral module clock, and external bus clock can be uniformly set to a 32.768 kHz subclock.

The USB module requires the 48-MHz clock. Set the external clock frequency and the MD\_CLK pin so that the USB clock (cku) frequency becomes 48 MHz.

Note that the MD\_CLK pin setting also changes the frequencies of the peripheral module clock, the external bus clock, and the system clock.



**Figure 27.1 Block Diagram of Clock Pulse Generator**

**Table 27.1 Selection of Clock Pulse Generator**

MD_CLK	EXTAL Input Clock Frequencies	I $\phi$ /P $\phi$ /B $\phi$	USB Clock (cku)
0	8 MHz to 18 MHz	EXTAL $\times 4$ , $\times 2$ , $\times 1$ , $\times 1/2$	EXTAL $\times 4$
1	16 MHz	EXTAL $\times 2$ , $\times 1$ , $\times 1/2$	EXTAL $\times 3$

Bus CLOCKS.

Bit	15	14	13	12	11	10	9
Bit Name	PSTOP1	PSTOP0	—	—	—	ICK2	ICK1
Initial Value	0	0	0	0	0	0	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit Name	—	PCK2	PCK1	PCK0	—	BCK2	BCK1
Initial Value	0	0	1	0	0	0	1
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	<p>B<math>\phi</math> Clock Output Enable Controls <math>\phi</math> output on PA7.</p> <ul style="list-style-type: none"> <li>Normal operation</li> </ul> <p>0: <math>\phi</math> output 1: Fixed high</p>
14	PSTOP0	0	R/W	<p><math>\phi</math> Clock Output Enable Controls <math>\phi</math> output (SD<math>\phi</math>) on PB7.</p> <ul style="list-style-type: none"> <li>Normal operation</li> </ul> <p>0: <math>\phi</math> output 1: Fixed high</p>

000: × 4 × 2  
 001: × 2 × 1  
 010: × 1 × 1/2  
 011: × 1/2 Setting prohibited  
 1XX: 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.

7	—	0	R/W	Reserved
Although this bit is readable/writable, only 0 should be written to.				
6	PCK2	0	R/W	Peripheral Module Clock (P $\phi$ ) Select
5	PCK1	1	R/W	These bits select the frequency of the peripheral module clock. The ratio to the input clock is as follows.
4	PCK0	0	R/W	
PCK (2:0) MD_CLK = 0      MD_CLK = 1				
000: × 4 × 2				
001: × 2 × 1				
010: × 1 × 1/2				
011: × 1/2 Setting prohibited				
1XX: Setting prohibited				
The frequency of the peripheral module clock should be set so as to be lower than that of the system clock. Though the ratio 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 if the system clock is lower than that of the two clocks.				

001: × 2 × 1  
 010: × 1 × 1/2  
 011: × 1/2 Setting prohibited  
 1XX: Setting prohibited

The frequency of the external bus clock should be set so as to make the frequency of the external bus clock higher than that of the system clock, the external bus clock will have the same frequency in reality.

Note: X: Don't care

### 27.1.2 Subclock Control Register (SUBCKCR)

SUBCKCR stops the main clock oscillator, selects the operating clock of the system clock, and selects the operating clock after a transition from software standby mode.

Bit	7	6	5	4	3	2	1
Bit Name	—	—	—	—	—	EXSTP	WAKE32K
Initial Value	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

standby mode.

1: The main clock oscillator and PLL are stopped during subclock operation.

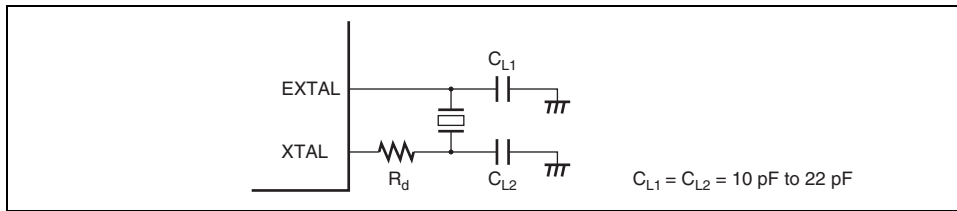
---

1	WAKE32K	0	R/W	<b>Wakeup Clock Select</b>  Selects the operating clock for use as the system clock after the transition from the subclock operation to software standby mode has been initiated by a hardware interrupt.  0: On leaving software standby mode, the main clock is selected as the operating clock.  1: On leaving software standby mode, the subclock is selected as the operating clock. This setting is valid when the WAKE32K bit (CK32K) is set to 1.
---	---------	---	-----	---

---

0	CK32K	0	R/W	<b>Subclock Select</b>  0: The system clock ( $I\phi$ ), peripheral module clock ( $P\phi$ ) and external bus clock ( $B\phi$ ) operate on the main clock.  1: The system clock ( $I\phi$ ), peripheral module clock ( $P\phi$ ) and external bus clock ( $B\phi$ ) operate on the subclock.  When the OSC32STP bit in TCR32K is 1, 1 can be written to this bit. This bit is cleared to 0 when the device enters software standby mode while the value of WAKE32K is 0. Dummy read of this bit must be performed immediately after this bit is written to.
---	-------	---	-----	---

---

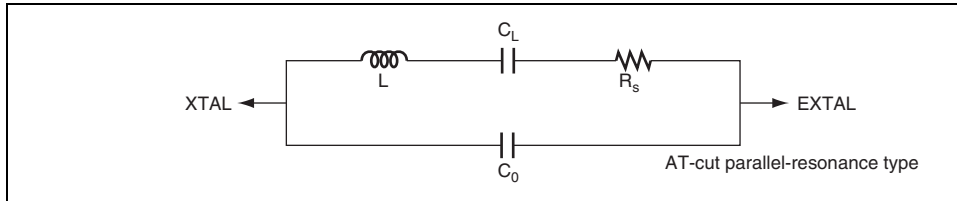


**Figure 27.2 Connection of Crystal Resonator (Example)**

**Table 27.2 Damping Resistance Value**

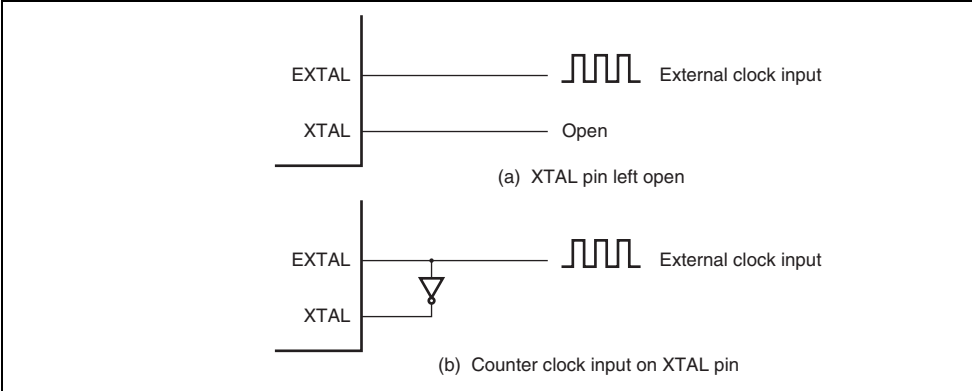
Frequency (MHz)	8	12	16	18
$R_d (\Omega)$	200	0	0	0

Figure 27.3 shows an equivalent circuit of the crystal resonator. Use a crystal resonator with the characteristics shown in table 27.3.

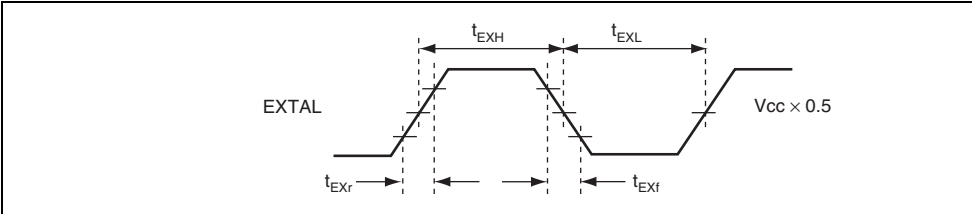


**Figure 27.3 Crystal Resonator Equivalent Circuit**

pin, put the external clock in high level during standby mode.



**Figure 27.4 External Clock Input (Examples)**



**Figure 27.5 External Clock Input Timing**

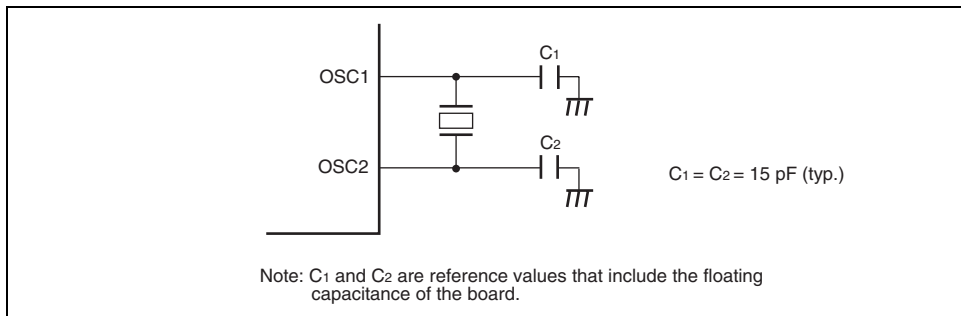


updated frequency.

## 27.5 Subclock Oscillator

### 27.5.1 Connecting 32.768 kHz Crystal Resonator

To supply a clock to the subclock oscillator, connect a 32.768-kHz crystal resonator, as shown in figure 27.6. The usage notes given in section 27.6.3, Notes on Board Design, apply to the connection of this crystal resonator.



**Figure 27.6 Connection Example of 32.768-kHz Crystal Resonator**

### 27.5.2 Handling of Pins when the Subclock is Not to be Used

If the subclock is not required, connect the OSC1 pin to V<sub>SS</sub> and leave the OSC2 pin open as shown in figure 27.8.

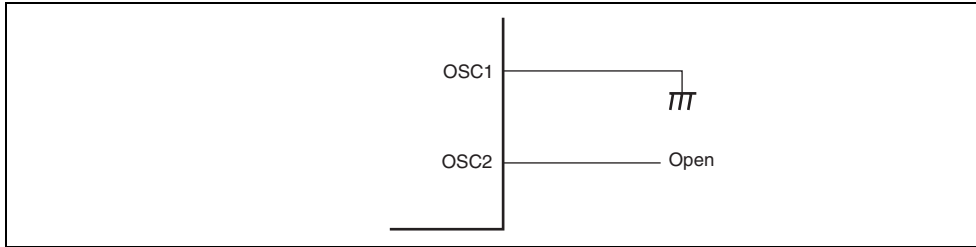


Figure 27.8 Pin Handling when Subclock is not Used

$I f_{\max} = 50 \text{ MHz}$ ,  $P f_{\max} = 35 \text{ MHz}$ , and  $B f_{\max} = 50 \text{ MHz}$ ,

the frequencies should satisfy the conditions  $8 \text{ MHz} \leq I f \leq 50 \text{ MHz}$ ,  $8 \text{ MHz} \leq P f \leq 35 \text{ MHz}$ , and  $8 \text{ MHz} \leq B f \leq 50 \text{ MHz}$ .

2. All the on-chip peripheral modules (except for the EXDMAC, DMAC, and DTC) operate at the  $P\phi$ . Note therefore that the time processing of modules such as a timer and SCI depends on the clock 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 28.7.3, Setting Oscillation Settling Time after Entering 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 PCK2 to PCK0 or BCK2 to BCK0.
4. Note that the frequency of  $\phi$  will be changed in the middle of a bus cycle when setting SCKCR or SUBCKCR while executing the external bus cycle with the write-data-buffer function of EXDMAC.
5. Figure 27.9 shows the clock modification timing. After a value is written to SCKCR, the system clock waits for the current bus cycle to complete. After the current bus cycle completes, each frequency will be modified within one cycle (worst case) of the external input clock.

### 27.6.2 Notes on Resonator

Since various characteristics related to the resonator are closely linked to the user's board, thorough evaluation is necessary on the user's part, using the resonator connection example 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 exceeding the maximum rating is not applied to the resonator pin.

### 27.6.3 Notes on Board Design

When using the crystal resonator, place the crystal resonator and its load capacitors as close to the XTAL and EXTAL pins as possible. Other signal lines should be routed away from the oscillation circuit as shown in figure 27.10 to prevent induction from interfering with correct oscillation.

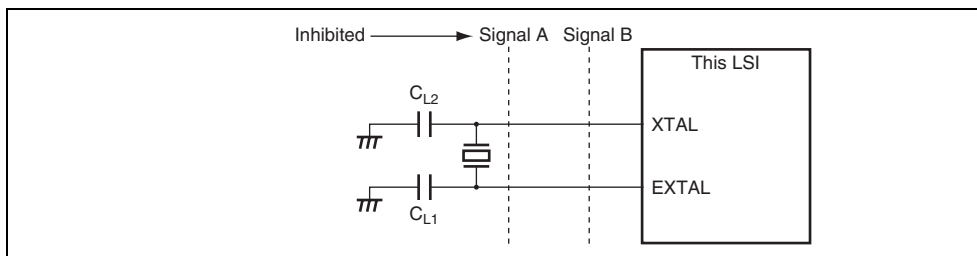


Figure 27.10 Note on Board Design for Oscillation Circuit

Note: \* CB and CPB are laminated ceramic capacitors.

## Figure 27.11 Recommended External Circuitry for PLL Circuit



All of the system clock, peripheral module clock, and external bus clock can be switched to a 32.768-kHz subclock.

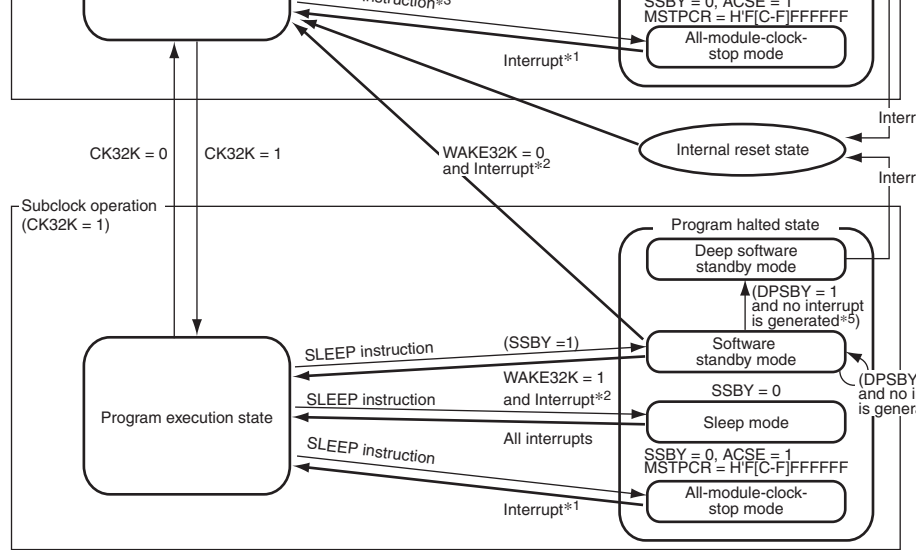
- **Module stop function**  
The functions for each peripheral module can be stopped to make a transition to a power-down mode.
- **Transition function to power-down mode**  
Transition to a power-down mode is possible to stop the CPU, peripheral modules, and oscillator.
- **Five power-down modes**
  - Sleep mode
  - All-module-clock-stop mode
  - Software standby mode
  - Deep software standby mode
  - Hardware standby mode

Table 28.1 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 a normal program execution state, the modules, other than the DMAC, DTC, and EXDMA, are stopped.

Subclock oscillator	Operating* <sup>9</sup>	Operating* <sup>9</sup>	Operating* <sup>9</sup>	Operating* <sup>9</sup>	Stopped
CPU	Stopped (retained)	Stopped (retained)	Stopped (retained)	Stopped (undefined)	Stopped (undefined)
On-chip RAMs 4 (H'FF2000 to H'FF3FFF)	Operating (retained)	Stopped (retained)	Stopped (retained)	Stopped (undefined)	Stopped (undefined)
On-chip RAMs 3 to 0 (H'FF4000 to H'FFBFFF)	Operating (retained)	Stopped (retained)	Stopped (retained)	Stopped (retained/undefined)* <sup>5</sup>	Stopped (undefined)
Universal Serial Bus interface	Operating	Stopped (retained)	Stopped (retained)	Stopped (retained/undefined)* <sup>5</sup>	Stopped (undefined)
Watchdog timer	Operating	Operating	Stopped (retained)	Stopped (undefined)	Stopped (undefined)
8-bit timer (unit 0/1)	Operating	Operating* <sup>4</sup>	Stopped (retained)	Stopped (undefined)	Stopped (undefined)
32K timer	Operating	Operating	Operating	Operating	Stopped
Voltage detection circuit* <sup>10</sup>	Operating	Operating	Operating	Operating	Stopped
Power-on reset circuit* <sup>10</sup>	Operating	Operating	Operating	Operating	Stopped
Other peripheral modules	Operating	Stopped* <sup>1</sup>	Stopped* <sup>1</sup>	Stopped* <sup>7</sup> (undefined)	Stopped (undefined)
I/O ports	Operating	Retained	Retained* <sup>6</sup>	Stopped* <sup>6</sup> (undefined)	Hi-Z



6. Retention or high-impedance for the address bus and bus-control signals ( $\overline{CS}$ ,  $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{HWR}$ , 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.
8. An external interrupt, 32K timer interrupt, or USB suspend/resume interrupt.
9. Start/stop can be selected by setting the OSC32STP bit in TCR32K.
10. External interrupt and voltage monitoring interrupt\*<sup>11</sup>.
11. Supported only by the H8SX/1665M Group.



[Legend]  $\longrightarrow$  Transition after exception handling

- Notes:
1. NMI, IRQ0 to IRQ11, 8-bit timer interrupt, watchdog timer interrupt, 32K timer interrupt, and voltage monitoring interrupt\*6. Note that the 8-bit timer interrupt is valid when the MSTPCRA9 or MSTPCRA8 bit is cleared to 0.
  2. NMI, IRQ0 to IRQ11, 32K timer interrupt, USB suspend/resume interrupt, and voltage monitoring interrupt\*6. Note that IRQ is valid only when the corresponding bit in SSIER is set to 1.
  3. The SLPIE bit is cleared to 0.
  4. NMI, IRQ0-A to IRQ3-A, 32K timer interrupts, USB suspend/resume interrupts, and voltage monitoring interrupt\*6. Note that IRQ, 32K timer, and voltage monitoring interrupt\*6 are valid only when the corresponding bit in DPSIER is set to 1.
  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 28.12, Usage Notes.
  6. Supported only by the H8SX/1665M Group.

Note: From any state, a transition to hardware standby mode occurs when  $\overline{STBY}$  is driven low.  
From any state except hardware standby mode, a transition to the reset state occurs when  $\overline{RES}$  is driven low.

**Figure 28.1 Mode Transitions**

- 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 n (DPSBKRn) (n = 15 to 0)

Bit name	SLPIE						
Initial value:	0	0	0	0	0	0	0
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 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 operation. For clearing, write 0 to this bit. When this bit 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{CS}</math>) are 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 set to 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>

according to the operating frequency so that the  
time is at least equal to the oscillation settling time  
an external clock, a PLL circuit settling time is  
Refer to table 28.2 to set the standby time.

While oscillation is being settled, the timer is  
the P $\phi$  clock frequency. Careful consideration is  
in multi-clock mode.

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

---

executed, this bit remains set to 1. For clearing, this bit.

6 to 0 — All 0 R/W Reserved

These bits are always read as 0. The write value always be 0.

[Legend]

x: Don't care

Note: With the F-ZTAT version, the flash memory settling time must be reserved.

### 28.2.2 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
Bit name	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9
Initial value:	0	0	0	0	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1
Bit name	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1
Initial value:	1	1	1	1	1	1	1
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 current consumption by stopping the bus controller I/O ports operations when the CPU executes the instruction after module stop state has been seen on-chip peripheral modules controlled by MSTPCRA.  0: All-module-clock-stop mode disabled 1: All-module-clock-stop mode enabled
14	MSTPA14	0	R/W	EXDMA controller (EXDMAC)
13	MSTPA13	0	R/W	DMA controller (DMAC)
12	MSTPA12	0	R/W	Data transfer controller (DTC)
11	MSTPA11	1	R/W	Reserved
10	MSTPA10	1	R/W	These bits are always read as 1. The write value always be 1.
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 always be 1.

1	MSTPA1	1	R/W	16-bit timer pulse unit (TPU channels 11 to 6)
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_0: PO15 t
14	MSTPB14	1	R/W	Reserved
13	MSTPB13	1	R/W	These bits are always read as 1. The write value always be 1.
12	MSTPB12	1	R/W	Serial communications interface_4 (SCI_4)
11	MSTPB11	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
10	MSTPB10	1	R/W	Serial communications interface_2 (SCI_2)
9	MSTPB9	1	R/W	Serial communications interface_1 (SCI_1)
8	MSTPB8	1	R/W	Serial communications interface_0 (SCI_0)
7	MSTPB7	1	R/W	I <sup>2</sup> C bus interface 2_1 (IIC2_1)
6	MSTPB6	1	R/W	I <sup>2</sup> C bus interface 2_0 (IIC2_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 always be 1.
2	MSTPB2	1	R/W	
1	MSTPB1	1	R/W	
0	MSTPB0	1	R/W	



Bit	15	14	13	12	11	10	9
Bit name	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9
Initial value:	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1
Bit name	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1
Initial value:	0	0	0	0	0	0	0
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	Serial communications interface_5 (SCI_5), (IrDA)
14	MSTPC14	1	R/W	Serial communications interface_6 (SCI_6)
13	MSTPC13	1	R/W	8-bit timer (TMR_4, TMR_5)
12	MSTPC12	1	R/W	8-bit timer (TMR_6, TMR_7)
11	MSTPC11	1	R/W	Universal Serial Bus interface (USB)
10	MSTPC10	1	R/W	Cyclic redundancy check calculator
9	MSTPC9	1	R/W	A/D converter (unit 1)
8	MSTPC8	1	R/W	Programmable pulse generator (PPG_1: PO31 to P

value.

## 28.2.4 Deep Standby Control Register (DPSBYCR)

DPSBYCR controls deep software standby mode.

DPSBYCR 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	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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 interrupt, this bit remains at 1. Write a 0 here to clear this bit. Setting of this bit has no effect when the WDT is in watchdog timer mode. In this case, executing the SLEEP instruction always initiates entry to sleep mode or module-clock-stop mode. Be sure to clear this bit by setting the SLPIE bit to 1.

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 determines whether or not the state that has been held in deep software standby mode is retained after 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.

In operation in external extended mode, however, the address bus, bus control signals ( $\overline{CS0}$ ,  $\overline{AS}$ ,  $\overline{RD}$ ,  $\overline{H}$ ,  $\overline{LWR}$ ), and data bus are set to the initial state upon exit from deep software standby mode.



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 RAMCUT 2, 1, and 0 control the internal power supply to the on-chip RAM and USB in deep software standby mode. RAMCUT 2 to 0 000: Power is supplied to the on-chip RAM and USB. 111: Power is not supplied to the on-chip RAM and USB. Settings other than above are prohibited.
---	---------	---	-----	---

### 28.2.5 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 interrupt.

DPSWCR 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

time for each operating frequency. When using external clock, settling time for the PLL circuit is considered. See table 28.3 to select the wait time.

During the oscillation settling period, counting is performed with the clock frequency input to the

000000: Reserved

000001: Reserved

000010: Reserved

000011: Reserved

000100: Reserved

000101: Wait time = 64 states

000110: Wait time = 512 states

000111: Wait time = 1024 states

001000: Wait time = 2048 states

001001: Wait time = 4096 states

001010: Wait time = 16384 states

001011: Wait time = 32768 states

001100: Wait time = 65536 states

001101: Wait time = 131072 states

001110: Wait time = 262144 states

001111: Wait time = 524288 states

01xxxx: Reserved

---

[Legend]

x: Don't care

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	DUSBIE	0	R/W	USB Suspend/Resume Interrupt Enable Enables/disables exit from deep software standby mode by USB suspend/resume interrupt signal. 0: Disables exit from deep software standby mode by USB suspend/resume interrupt signal. 1: Enables exit from deep software standby mode by USB suspend/resume interrupt signal.
5	DT32KIE	0	R/W	32K Timer Interrupt Enable Enables/disables exit from deep software standby mode by 32K timer interrupt signal. 0: Disables exit from deep software standby mode by 32K timer interrupt signal. 1: Enables exit from deep software standby mode by 32K timer interrupt signal.

				Enables or disables exit from deep software standby mode by $\overline{\text{IRQ3-A}}$ .
				0: Disables exit from deep software standby mode by $\overline{\text{IRQ3-A}}$ .
				1: Enables exit from deep software standby mode by $\overline{\text{IRQ3-A}}$ .
2	DIRQ2E	0	R/W	IRQ2 Interrupt Enable Enables or disables exit from deep software standby mode by $\overline{\text{IRQ2-A}}$ .
				0: Disables exit from deep software standby mode by $\overline{\text{IRQ2-A}}$ .
				1: Enables exit from deep software standby mode by $\overline{\text{IRQ2-A}}$ .
1	DIRQ1E	0	R/W	IRQ1 Interrupt Enable Enables or disables exit from deep software standby mode by $\overline{\text{IRQ1-A}}$ .
				0: Disables exit from deep software standby mode by $\overline{\text{IRQ1-A}}$ .
				1: Enables exit from deep software standby mode by $\overline{\text{IRQ1-A}}$ .
0	DIRQ0E	0	R/W	IRQ0 Interrupt Enable Enables or disables exit from deep software standby mode by $\overline{\text{IRQ0-A}}$ .
				0: Disables exit from deep software standby mode by $\overline{\text{IRQ0-A}}$ .
				1: Enables exit from deep software standby mode by $\overline{\text{IRQ0-A}}$ .

Note: \* Supported only by the H8SX/1665M Group.

Initial value:	0	0	0	0	0	0	0
R/W:	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1	R/(W)*1

- Notes: 1. Only 0 can be written to clear the flag.  
 2. Supported only by the H8SX/1665M Group.

Bit	Bit Name	Initial Value	R/W	Module
7	DNMIF	0	R/(W)*1	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	DUSBIF	0	R/(W)*1	USB Suspend/Resume Interrupt Flag [Setting condition] When the USB suspend/resume interrupt occurs. [Clearing condition] Writing a 0 to this bit after reading it as 1.
5	DT32KIF	0	R/(W)*1	32K Timer Interrupt Flag [Setting condition] When the 32K timer interrupt occurs. [Clearing condition] Writing a 0 to this bit after reading it as 1.



				[Clearing condition] Writing a 0 to this bit after reading it as 1.
2	DIRQ2F	0	R/(W)* <sup>1</sup>	IRQ2 Interrupt Flag [Setting condition] $\overline{\text{IRQ2}}$ -A input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.
1	DIRQ1F	0	R/(W)* <sup>1</sup>	IRQ1 Interrupt Flag [Setting condition] $\overline{\text{IRQ1}}$ -A input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.
0	DIRQ0F	0	R/(W)* <sup>1</sup>	IRQ0 Interrupt Flag [Setting condition] $\overline{\text{IRQ0}}$ -A input specified in DPSIEGR is generated. [Clearing condition] Writing a 0 to this bit after reading it as 1.

- Notes: 1. Only 0 can be written to clear the flag.  
2. Supported only by the H8SX/1665M Group.

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 always be 0.
3	DIRQ3EG	0	R/W	<b>IRQ3 Interrupt Edge Select</b> Selects the active edge for $\overline{\text{IRQ3}}$ -A 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 $\overline{\text{IRQ2}}$ -A 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 $\overline{\text{IRQ1}}$ -A pin input. 0: The interrupt request is generated by a falling edge. 1: The interrupt request is generated by a rising edge.



The DPSRSTF bit in RSTSR indicates that deep software standby mode has been canceled by an interrupt.

RSTSR is not initialized by the internal reset signal upon exit from deep software standby mode.

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

- Notes:
1. Only 0 can be written to clear the flag.
  2. Supported only by the H8SX/1665M Group.
  3. Initial value is undefined in the H8SX/1665M Group.
  4. Only 0 can be written to clear the flag in the H8SX/1665M Group.
  5. Readable only in the H8SX/1665M Group.

Bit	Bit Name	Initial Value	R/W	Module
7	DPSRSTF	0	R/(W)*	<p>Deep Software Standby Reset Flag</p> <p>Indicates that deep software standby mode has been canceled by an interrupt source specified in DPSIEGR and an internal reset is generated.</p> <p>[Setting condition]</p> <p>Deep software standby mode is canceled by an interrupt source.</p> <p>[Clearing condition]</p> <p>Writing a 0 to this bit after reading it as 1.</p>

- H8SX/1665M Group

Bit	Bit Name	Initial Value	R/W	Module
2	LVDF	Undefined	R/(W)*	LVD Flag This bit indicates that the voltage-detection circuit detected a low voltage (Vcc at or below Vdet). For details, see section 5, Voltage Detection Circuit (LVD).
1	—	Undefined	R/W	Reserved The write value should always be 0.
0	PORF	Undefined	R	Power-on reset flag Indicates the Power-on reset is generated. For details, see section 4, Resets.

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

Bit	7	6	5	4	3	2	1
Bit name	BKUPn7	BKUPn6	BKUPn5	BKUPn4	BKUPn3	BKUPn2	BKUPn1
Initial value:	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W

n: 15 to 0

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.

### 28.3.2 Switching to Subclock

When the CK32K bit in SUBCKCR is set to 1, a transition from the main clock operation to subclock operation is made at the end of the bus cycle regardless of the SCKCR setting. In subclock operation, the CPU, bus masters, peripheral modules, and all external buses operate at the 32.768-kHz subclock.

When the CK32K bit in SUBCKCR is set to 0 in the subclock operation, a transition to the main clock operation is made at the end of the bus cycle. Since a transition from the subclock operation to the main clock operation is made via software standby mode, the oscillation settling time of the main clock must elapse. Set the oscillation settling time of the main clock with bits STS4 to STS0 in SBYCR.

The main clock oscillator can be operated or stopped by the EXSTP bit in SUBCKCR in the subclock operation. When a transition is made from the subclock operation to the main clock operation with the main clock oscillator operating, the wait for the oscillation settling time of the main clock oscillator is not necessary. A transition to the main clock operation can be made with the minimum setting time with the setting of bits STS4 to STS0 in SBYCR.

In the same way as in the main clock operation, if a SLEEP instruction is executed in the subclock operation while the SSBY bit in SBYCR is set to 1, this LSI enters software standby mode. When a transition is made to software standby mode in the subclock operation, the operating clock after clearing of software standby mode can be selected with the WAKE32K bit in SUBCKCR. This LSI is placed in the subclock operation if the WAKE32K bit is 1, or placed in the main clock operation if the WAKE32K bit is 0.

After the reset state is cleared, all modules other than the EXDMAC, DMAC, and DTC 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.

Sleep mode is exited by any interrupt, signals on the RES or STBY pin, and a reset caused by a watchdog timer overflow, a voltage monitoring reset\*, or a power-on reset\*.

- 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 on 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.
- Exit from voltage monitoring reset\*  
Sleep mode is exited by a voltage monitoring reset of the voltage detection circuit.
- Exit from power-on reset\*  
Sleep mode is exited by a power-on reset.

Note: \* Supported only by the H8SX/1665M Group.



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  $\overline{\text{IRQ11}}$ ,  $\overline{\text{RES}}$  pin input, or an internal interrupt (8-bit timer\*<sup>1</sup>, watchdog timer, 32K timer, or voltage detector\*<sup>2</sup>), 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 non-activation source.

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

- Notes:
1. Operation or halting of the 8-bit timer can be selected by bits MSTPA9 and MSTPA10 in MSTPCRA.
  2. Supported only by the H8SX/1665M Group.

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.

### 28.7.2 Exit from Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI, or IRQ0 to IRQ11\*<sup>1</sup>) or an interrupt (32K timer, voltage detection interrupt\*<sup>2</sup>, or USB suspend/resume), voltage-detection reset\*<sup>2</sup>, power-on reset\*<sup>2</sup>, 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, IRQ0 to IRQ11\*<sup>1</sup>, 32K-timer, or USB suspend/resume interrupt request is input, clock oscillation starts, and after the elapse of the time set in bits STS4 to STS7 of the 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 IRQ0 to IRQ11\*<sup>1</sup> interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than the IRQ0 to IRQ11\*<sup>1</sup> 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.

2. Exit from voltage monitoring reset\*<sup>2</sup>

When a voltage monitoring reset is generated by the fall of power-voltage, software standby mode is cleared and a clock oscillation starts. At the same time, a clock signal is supplied throughout the LSI. After that, if power voltage rises, the voltage detection reset is released. Thereafter, CPU starts the reset exception handling.

5. 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.


- Notes:
1. By setting the SSIn bit in SSIER to 1, IRQ0 to IRQ11 can be used as a software standby mode clearing source.
  2. Supported only by the H8SX/1665M Group.

**Table 28.2 Oscillation Settling Time Setting**

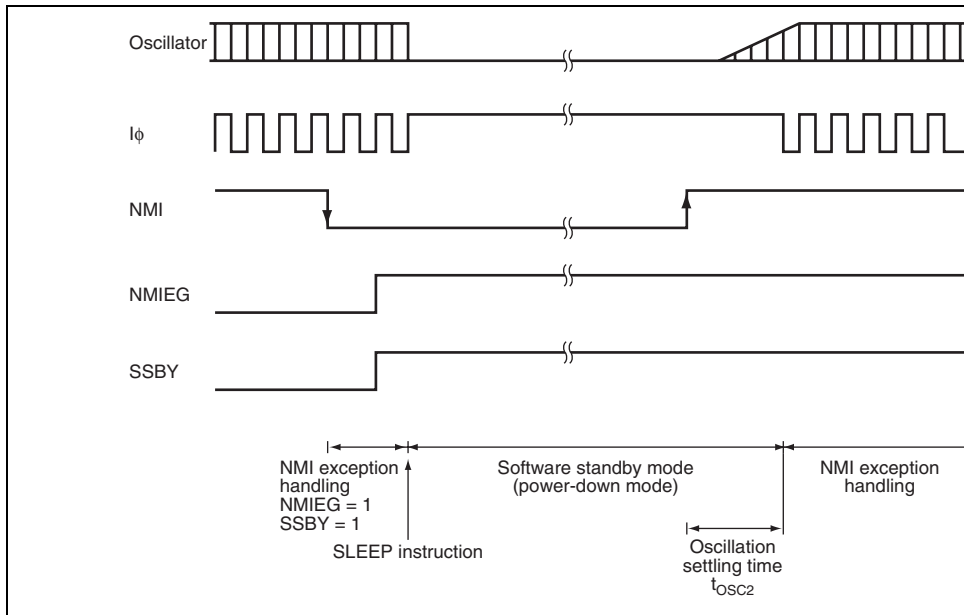
STS4	STS3	STS2	STS1	STS0	Standby Time	P $\phi$ * (MHz)											
						35	25	20	13	10	8						
0	0	0	0	0	Reserved	—	—	—	—	—	—						
					1	Reserved	—	—	—	—	—	—					
					1	0	Reserved	—	—	—	—	—	—				
						1	Reserved	—	—	—	—	—	—				
					1	0	0	Reserved	—	—	—	—	—	—			
								1	64	1.8	2.6	3.2	4.9	6.4	8.0		
								1	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	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
					1	0	0	1	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				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			

[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.



**Figure 28.2 Software Standby Mode Application Example**

standby mode is made, software standby mode will be cleared regardless of the DPSBYT1 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 CPSBY bit in DPSBYCR are set to 1 and software standby mode clearing source occurs, a transition to deep software standby mode is made immediately after software standby mode is entered.

In deep software standby mode, the CPU, on-chip peripheral functions (except for the USB and 32K timer), on-chip RAM 4, and oscillator functionality are all stopped. 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 (except for the USB and 32K timer), and on-chip RAM 4 become undefined.

Contents of the on-chip RAMs 3 to 0 and USB registers can be retained when all the bits RAMCUT2 to RAMCUT0 in DPSBYCR have been cleared to 0. If these bits are set to 1, the internal power supply to the on-chip RAMs 3 to 0 and USB stops and the power consumption is further reduced. At this time, the contents of the on-chip RAMs 3 to 0 and USB registers become undefined.

The 32K timer, voltage detection circuit\*, and power-on reset circuit\* can be operated in software standby mode.

The I/O ports can be retained in the same state as in software standby mode.

Note: \* Supported only by the H8SX/1665M Group.

0) bit is set to 1 when a specified edge is generated in the LSI. The DIRQnE (n = 3 to 0) bit in DPSIER. The rising or falling edge of the signals can be specified with DPSIEGR. The DT32KIF bit is set to 1 when a 32KHz interrupt occurs. The DLVDIF bit is set to 1 when a voltage-monitoring interrupt occurs. The DUSBIF bit is set to 1 when a USB suspend/resume interrupt occurs.

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 to 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 or internal interrupt signal, the DPSRSTF bit in RSTSR is set to 1.

2. Exit from deep software standby mode by a voltage-monitoring reset\*

When a voltage monitoring reset is generated by the power-supply voltage falling, the LSI is released from deep software standby mode and internal power supply starts simultaneously with the start of clock oscillation. At the same time, a clock signal is supplied throughout the LSI. When the power-supply voltage has risen sufficiently, the LSI is released from the voltage-detection reset state after the clock oscillation stabilization time has been secured. The CPU then starts reset-exception handling.

3. Exit from power-on reset\*

When a power-on reset is generated by the power-supply voltage falling, the LSI is released from deep software standby mode. If the power-supply voltage then rises sufficiently, clock oscillation starts and the LSI is released from the power-on reset state after the clock oscillation stabilization time has been secured. As soon as the clock oscillation starts, a clock signal is provided to the LSI. After that, the CPU starts reset-exception handling.

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 ( $\overline{CS0}$ ,  $\overline{AS}$ ,  $\overline{HWR}$ , and  $\overline{LWR}$ ), and data bus output pins are initialized 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

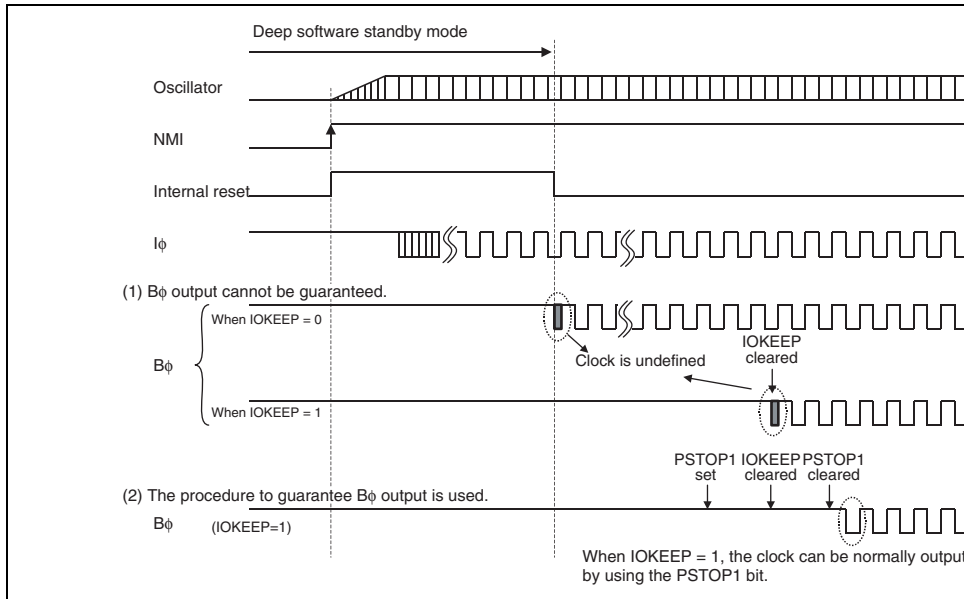
The port states that were held in deep software standby mode are retained regardless of the internal state though the internal of the LSI is initialized by an internal reset caused by deep software standby mode. At this time, the port states that were held in software standby mode are retained even if 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.

The IOKEEP bit is not initialized by an internal reset caused by canceling deep software standby mode.



1. Change the value of the PSTOP1 bit from 0 to 1 to fix the B $\phi$  output at the high level (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.

In case of the SDRAM $\phi$ , clock can be normally output by controlling the PSTOP0 bit in the same way as the procedure above mentioned. For the port state with IOKEEP bit is set to 1, see section 28.8.3, Pin State on Exit from Deep Software Standby Mode.



**Figure 28.3 B $\phi$  Operation after Exit from Deep Software Standby Mode**

**Table 28.3 Oscillation Settling Time Settings**

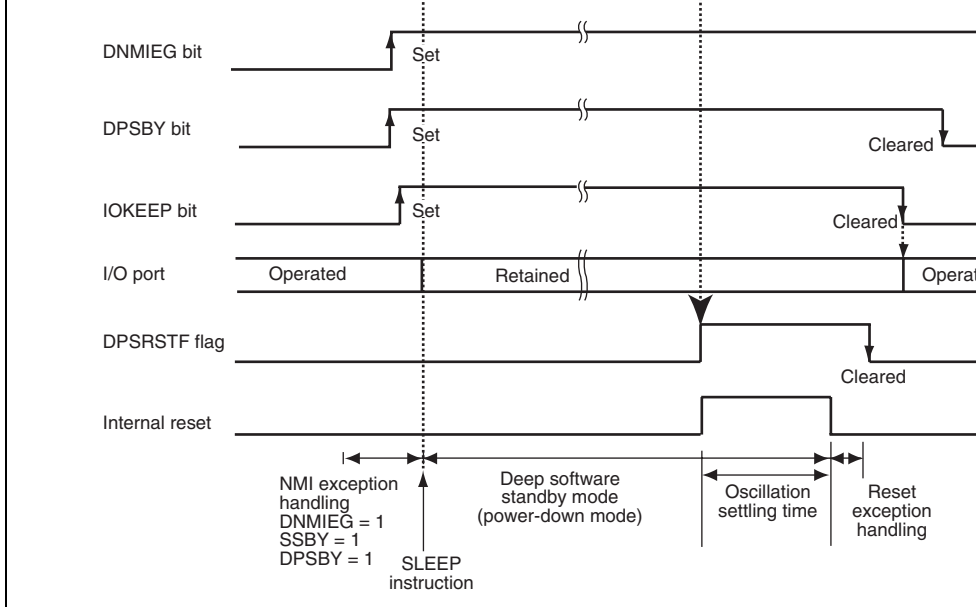
							EXTAL Input Clock Frequency* (MHz)					
WT STS5	WT STS4	WT STS3	WT STS2	WT STS1	WT STS0	Standby Time	18	16	14	12	10	8
0	0	0	0	0	0	Reserved	—	—	—	—	—	—
						1	Reserved	—	—	—	—	—
				1	0	Reserved	—	—	—	—	—	—
						1	Reserved	—	—	—	—	—
			1	0	0	Reserved	—	—	—	—	—	—
						1	64	3.6	4.0	4.6	5.3	6.4
				1	0	512	28.4	32.0	36.6	42.7	51.2	64
						1	1024	56.9	64.0	73.1	85.3	102.4
			1	0	0	2048	113.8	128.0	146.3	170.7	204.8	256
						1	4096	0.23	0.26	0.29	0.34	0.41
				1	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
			1	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
				1	0	262144	14.56	16.38	18.72	21.85	26.21	32.22
						1	524288	29.13	32.77	37.45	43.69	52.43
	1	0	0	0	0	Reserved	—	—	—	—	—	—

[Legend]

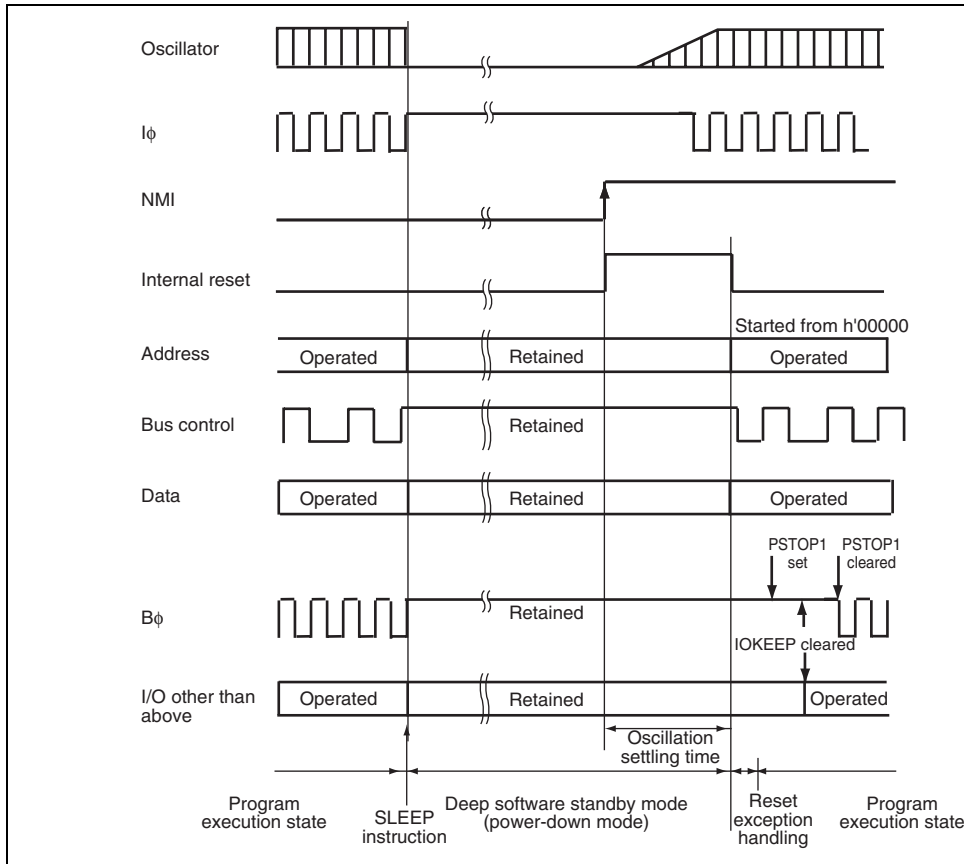
- : Recommended setting when external clock is in use
- : Recommended setting when crystal oscillator is in use

In this example, falling-edge sensing of NMI interrupts has been specified by clearing the 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.

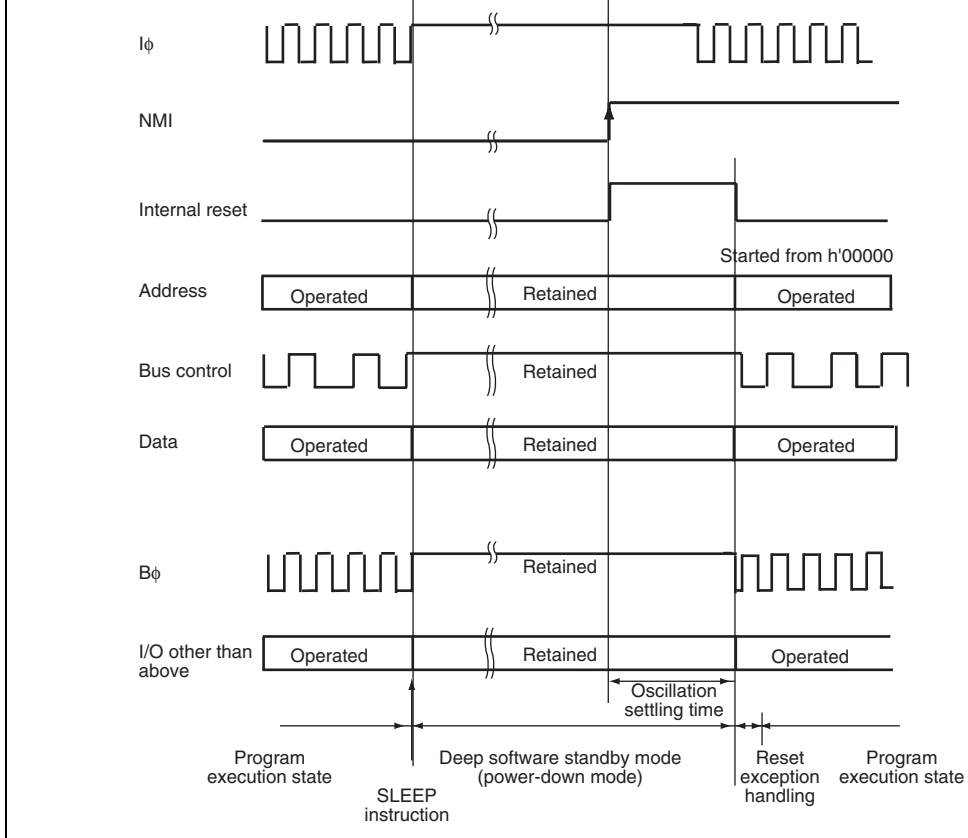
After that, deep software standby mode is canceled at the rising edge on the NMI pin.



**Figure 28.4 Deep Software Standby Mode Application Example (IOKEEP = 1)**

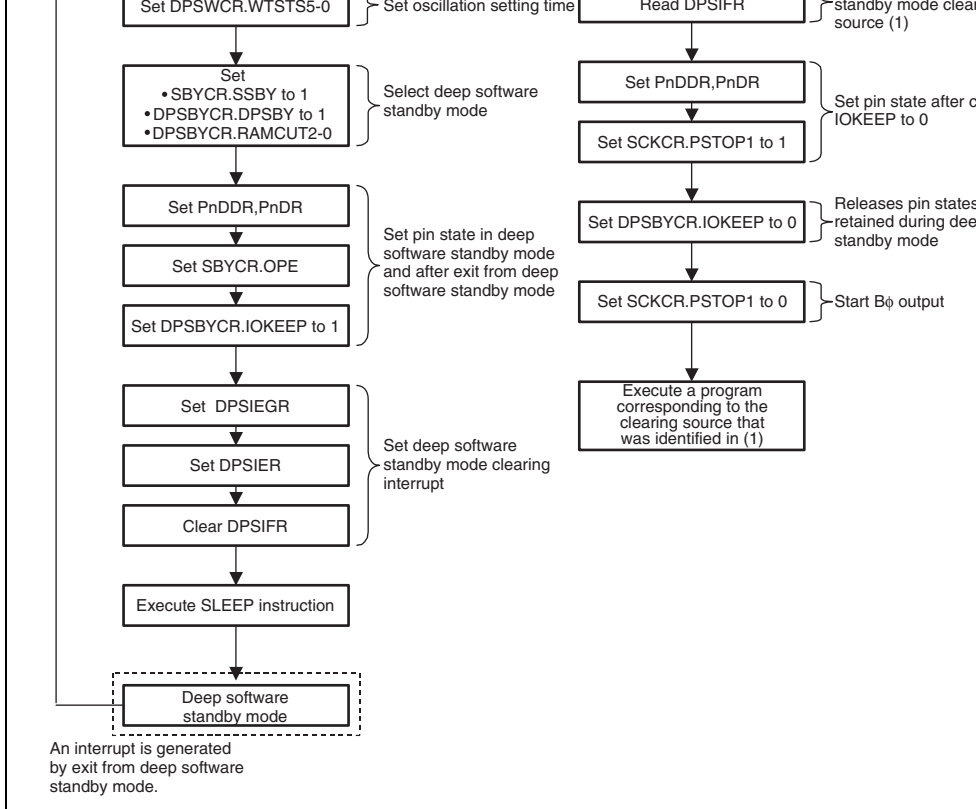


**Figure 28.5 Example of Deep Software Standby Mode Operation in External Extended Mode (IOKEEP = OPE = 1)**



**Figure 28.6 Example of Deep Software Standby Mode Operation in External Extended Mode (IOKEEP = OPE = 0)**

Output is also set.



**Figure 28.7 Flowchart of Deep Software Standby Mode Operation**



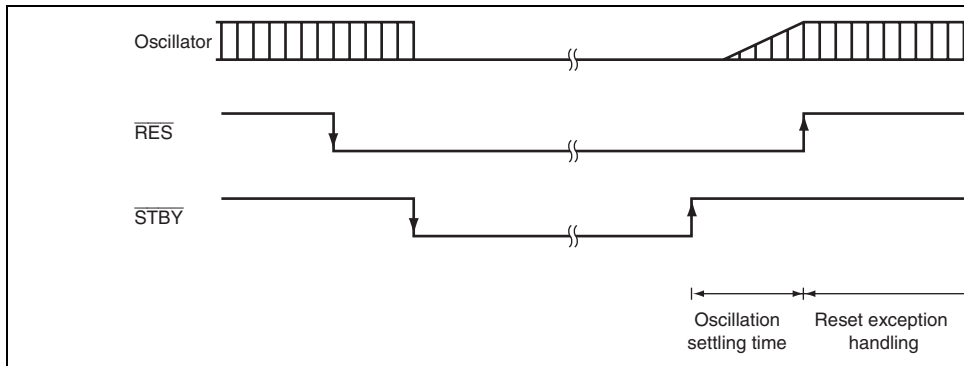
## 28.9.2 Clearing Hardware Standby Mode

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 started. Ensure that the  $\overline{\text{RES}}$  pin is held low until clock oscillation settles (for details on oscillation settling time, refer to table 28.2). 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.

## 28.9.3 Hardware Standby Mode Timing

Figure 28.8 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.

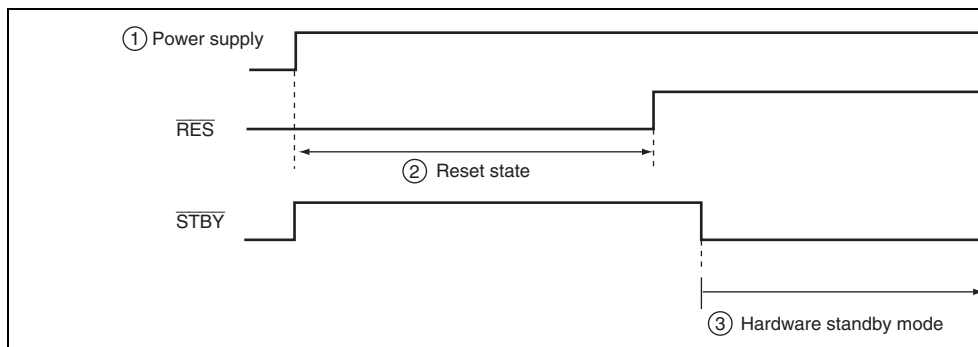


**Figure 28.8 Hardware Standby Mode Timing**

Timing.

In a power-on reset\*, power on while driving the  $\overline{\text{STBY}}$  or  $\overline{\text{RES}}$  pin to a high-level.

Note: \* Supported only by the H8SX/1665M Group.



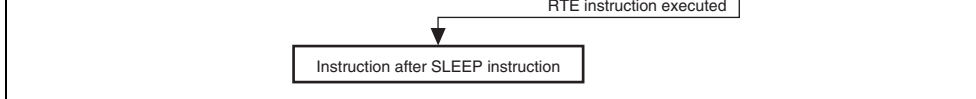
**Figure 28.9 Timing Sequence at Power-On**

When a SLEEP instruction is executed while the SLPIE bit is cleared to 0, a transition to the power-down state. Exit from the power-down state is initiated by an exit-initiating interrupt source (see figure 28.10).

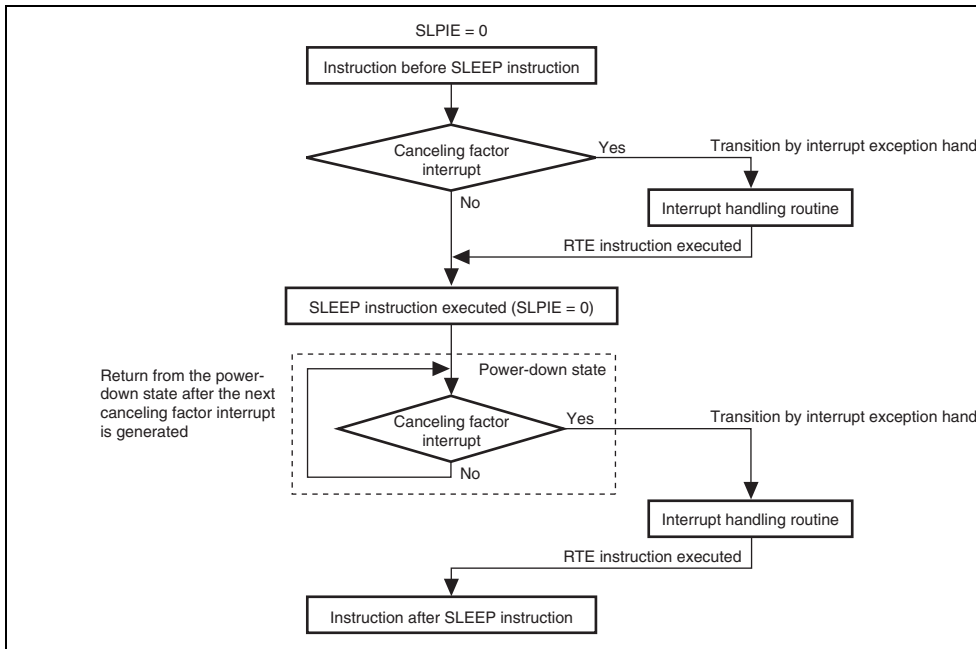
When an interrupt that causes exit from the power-down state is generated immediately after the execution of a SLEEP instruction, exception handling for the interrupt starts. On return from the exception service routine, the SLEEP instruction is executed to enter the power-down state. In this case, exit from the power-down state will not take place until the next time an exit-initiating interrupt is generated (see figure 28.11).

As stated above, setting the SLPIE bit to 1 causes sleep-instruction exception handling to start at the execution of the SLEEP instruction. If this setting is made in the exception service routine, when an interrupt that initiates exit from the power-down state, handling of the sleep-instruction exception due to the execution of a SLEEP instruction will proceed even if the interrupt was generated immediately beforehand (see figure 28.12). Consequently, the CPU will execute the instruction that follows the SLEEP instruction, after handling of the sleep-instruction exception and exception service routine, and will not enter the power-down state.

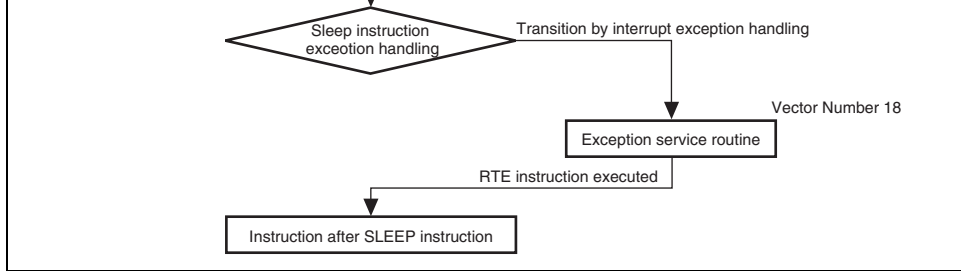
Thus, when the SLPIE bit is set to 1 to enable the sleep exception handling, clear the SSBYCR to 0.



**Figure 28.10 When an Interrupt that Initiates Exit from the Power-Down State is Generated after SLEEP Instruction Execution**



**Figure 28.11 When an Interrupt that Initiates Exit from the Power-Down State is Generated before SLEEP Instruction Execution (Sleep-Instruction Exception Handling does not Proceed)**



**Figure 28.12 When an Interrupt that Initiates Exit from the Power-Down State is Generated before SLEEP Instruction Execution (Sleep Instruction Exception Handling Proceeds)**

**Table 28.4  $\phi$  Pin (PA7) State in Each Processing State**

Register Setting Value		Normal		All-Module-	Software Standby Mode		Deep Software Standby Mode	
DDR	PSTOP1	Operating Mode	Sleep Mode	Clock-Stop Mode	OPE = 0	OPE = 1	IOKEEP = 0	IOKEEP = 1
0	x	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
1	1	High	High	High	High	High	High	High

[Legend]

x = Don't care

**Table 28.5  $\phi$  Pin (PB7) State in Each Processing State (When SDRAM Interface is Enabled)**

Register Setting Value		Normal		All-Module-	Software Standby Mode		Deep Software Standby Mode	
PSTOP0		Operating Mode	Sleep Mode	Clock-Stop Mode	OPE = 0	OPE = 1	IOKEEP = 0	IOKEEP = 1
0		SD $\phi$ output	SD $\phi$ output	SD $\phi$ output	High	High	High	High
1		High	High	High	High	High	High	High

Current consumption increases during the oscillation settling standby period.

### **28.12.3 Module Stop State of EXDMAC, DMAC, or DTC**

Depending on the operating state of the EXDMAC, DMAC, and DTC, bits MSTPA14, MSTPA13, and MSTPA12 may not be set to 1, respectively. The module stop state setting of EXDMAC, DMAC, or DTC should be carried out only when the EXDMAC, DMAC, or DTC is not activated.

For details, refer to section 10, DMA Controller (DMAC), section 11, EXDMA Controller (EXDMAC), and section 12, Data Transfer Controller (DTC).

### **28.12.4 On-Chip Peripheral Module Interrupts**

Relevant interrupt operations cannot be performed in a module stop state. Consequently, when a 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.

### **28.12.5 Writing to MSTPCRA, MSTPCRB, and MSTPCRC**

MSTPCRA, MSTPCRB, and MSTPCRC should only be written to by the CPU.

If a conflict among the 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 until 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 among a deep software standby mode transition, the IRQ0 to IRQ11 interrupt timer interrupt, and voltage-monitoring interrupt\* occurs, a transition to deep software standby mode can be made without executing the interrupt execution handling by clearing the SSIER to 0 beforehand.

Note: \* Supported only by the H8SX/1665M Group

### **28.12.8 B $\phi$ /SDRAM $\phi$ Output State**

B $\phi$ /SDRAM $\phi$  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 $\phi$ /SDRAM $\phi$  can be normally output by setting the IOKEEP, PSTOP1, and PSTOP2 bits to 1. For details, see section 28.8.4, B $\phi$ /SDRAM $\phi$  Operation after Exit from Deep Software Standby Mode.



clock. For details, refer to section 9.5.4, 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 output or input 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 mode for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

Time constant registerA_5	TCORA_5	8	H'FEA45	TMR_5	16	3F
Time constant registerB_4	TCORB_4	8	H'FEA46	TMR_4	16	3F
Time constant registerB_5	TCORB_5	8	H'FEA47	TMR_5	16	3F
Timer counter_4	TCNT_4	8	H'FEA48	TMR_4	16	3F
Timer counter_5	TCNT_5	8	H'FEA49	TMR_5	16	3F
Timer counter control register_4	TCCR_4	8	H'FEA4A	TMR_4	16	3F
Timer counter control register_5	TCCR_5	8	H'FEA4B	TMR_5	16	3F
CRC control register	CRCCR	8	H'FEA4C	CRC	16	3F
CRC data input register	CRCDIR	8	H'FEA4D	CRC	16	3F
CRC data output register	CRCDOR	16	H'FEA4E	CRC	16	3F
Timer control register_6	TCR_6	8	H'FEA50	TMR_6	16	3F
Timer control register_7	TCR_7	8	H'FEA51	TMR_7	16	3F
Timer control/status register_6	TCSR_6	8	H'FEA52	TMR_6	16	3F
Timer control/status register_7	TCSR_7	8	H'FEA53	TMR_7	16	3F
Time constant registerA_6	TCORA_6	8	H'FEA54	TMR_6	16	3F
Time constant registerA_7	TCORA_7	8	H'FEA55	TMR_7	16	3F
Time constant registerB_6	TCORB_6	8	H'FEA56	TMR_6	16	3F
Time constant registerB_7	TCORB_7	8	H'FEA57	TMR_7	16	3F
Timer counter_6	TCNT_6	8	H'FEA58	TMR_6	16	3F
Timer counter_7	TCNT_7	8	H'FEA59	TMR_7	16	3F
Timer counter control register_6	TCCR_6	8	H'FEA5A	TMR_6	16	3F
Timer counter control register_7	TCCR_7	8	H'FEA5B	TMR_7	16	3F
A/D data register A_1	ADDRA_1	16	H'FEA80	A/D_1	16	3F
A/D data register B_1	ADDRB_1	16	H'FEA82	A/D_1	16	3F

A/D control register_1	ADCR_1	8	H'FEAA1	A/D_1	16
A/D mode selection register_1	ADMOSEL_1	8	H'FEAA2	A/D_1	16
A/D sampling state register_1	ADSSTR_1	8	H'FEAAB	A/D_1	16
Interrupt flag register 0	IFR0	8	H'FEE00	USB	8
Interrupt flag register 1	IFR1	8	H'FEE01	USB	8
Interrupt flag register 2	IFR2	8	H'FEE02	USB	8
Interrupt enable register 0	IER0	8	H'FEE04	USB	8
Interrupt enable register 1	IER1	8	H'FEE05	USB	8
Interrupt enable register 2	IER2	8	H'FEE06	USB	8
Interrupt select register 0	ISR0	8	H'FEE08	USB	8
Interrupt select register 1	ISR1	8	H'FEE09	USB	8
Interrupt select register 2	ISR2	8	H'FEE0A	USB	8
EP0i data register	EPDR0i	8	H'FEE0C	USB	8
EP0o data register	EPDR0o	8	H'FEE0D	USB	8
EP0s data register	EPDR0s	8	H'FEE0E	USB	8
EP1 data register	EPDR1	8	H'FEE10	USB	8
EP2 data register	EPDR2	8	H'FEE14	USB	8
EP3 data register	EPDR3	8	H'FEE18	USB	8
EP0o receive data size register	EPSZ0o	8	H'FEE24	USB	8
EP1 receive data size register	EPSZ1	8	H'FEE25	USB	8
Data status register	DASTS	8	H'FEE27	USB	8
FIFO clear register	FCLR	8	H'FEE28	USB	8
End point store register	EPSTL	8	H'FEE2A	USB	8
Trigger register	TRG	8	H'FEE2C	USB	8

Port M data register	PMDR	8	H'FEE51	I/O port	8	3F
Port M register	PORTM	8	H'FEE52	I/O port	8	3F
Port M input buffer control register	PMICR	8	H'FEE53	I/O port	8	3F
Serial mode register_5	SMR_5	8	H'FF600	SCI_5	8	3F
Bit rate register_5	BRR_5	8	H'FF601	SCI_5	8	3F
Serial control register_5	SCR_5	8	H'FF602	SCI_5	8	3F
Transmit data register_5	TDR_5	8	H'FF603	SCI_5	8	3F
Serial status register_5	SSR_5	8	H'FF604	SCI_5	8	3F
Receive data register_5	RDR_5	8	H'FF605	SCI_5	8	3F
Smart card mode register_5	SCMR_5	8	H'FF606	SCI_5	8	3F
Serial extended mode register_5	SEMR_5	8	H'FF608	SCI_5	8	3F
IrDA control register	IrCR	8	H'FF60C	SCI_5	8	3F
Serial mode register_6	SMR_6	8	H'FF610	SCI_6	8	3F
Bit rate register_6	BRR_6	8	H'FF611	SCI_6	8	3F
Serial control register_6	SCR_6	8	H'FF612	SCI_6	8	3F
Transmit data register_6	TDR_6	8	H'FF613	SCI_6	8	3F
Serial status register_6	SSR_6	8	H'FF614	SCI_6	8	3F
Receive data register_6	RDR_6	8	H'FF615	SCI_6	8	3F
Smart card mode register_6	SCMR_6	8	H'FF616	SCI_6	8	3F
Serial extended mode register_6	SEMR_6	8	H'FF618	SCI_6	8	3F
PPG output control register_1	PCR_1	8	H'FF636	PPG_1	8	3F
PPG output mode register_1	PMR_1	8	H'FF637	PPG_1	8	3F
Next data enable register H_1	NDERH_1	8	H'FF638	PPG_1	8	3F
Next data enable register L_1	NDERL_1	8	H'FF639	PPG_1	8	3F

Break address register AL	BARAL	16	H'FFA02	UBC	16
Break address mask register AH	BAMRAH	16	H'FFA04	UBC	16
Break address mask register AL	BAMRAL	16	H'FFA06	UBC	16
Break address register BH	BARBH	16	H'FFA08	UBC	16
Break address register BL	BARBL	16	H'FFA0A	UBC	16
Break address mask register BH	BAMRBH	16	H'FFA0C	UBC	16
Break address mask register BL	BAMRBL	16	H'FFA0E	UBC	16
Break address register CH	BARCH	16	H'FFA10	UBC	16
Break address register CL	BARCL	16	H'FFA12	UBC	16
Break address mask register CH	BAMRCH	16	H'FFA14	UBC	16
Break address mask register CL	BAMRCL	16	H'FFA16	UBC	16
Break address register DH	BARDH	16	H'FFA18	UBC	16
Break address register DL	BARDL	16	H'FFA1A	UBC	16
Break address mask register DH	BAMRDH	16	H'FFA1C	UBC	16
Break address mask register DL	BAMRDL	16	H'FFA1E	UBC	16
Break control register A	BRCRA	16	H'FFA28	UBC	16
Break control register B	BRCRB	16	H'FFA2C	UBC	16
Break control register C	BRCRC	16	H'FFA30	UBC	16
Break control register D	BRCRD	16	H'FFA34	UBC	16
Timer control register	TCR32K	8	H'FFABC	TM32K	8
Timer counter 1	TCNT32K1	8	H'FFABD	TM32K	8
Timer counter 2	TCNT32K2	8	H'FFABE	TM32K	8
Timer counter 3	TCNT32K3	8	H'FFABF	TM32K	8
A/D sampling state register_0	ADSSTR_0	8	H'FFADB	A/D_0	16

Timer status register_6	TSR_6	8	H'FFB15	TPU_6	16	2F
Timer counter_6	TCNT_6	16	H'FFB16	TPU_6	16	2F
Timer general register A_6	TGRA_6	16	H'FFB18	TPU_6	16	2F
Timer general register B_6	TGRB_6	16	H'FFB1A	TPU_6	16	2F
Timer general register C_6	TGRC_6	16	H'FFB1C	TPU_6	16	2F
Timer general register D_6	TGRD_6	16	H'FFB1E	TPU_6	16	2F
Timer control register_7	TCR_7	8	H'FFB20	TPU_7	16	2F
Timer mode register_7	TMDR_7	8	H'FFB21	TPU_7	16	2F
Timer I/O control register_7	TIOR_7	8	H'FFB22	TPU_7	16	2F
Timer interrupt enable register_7	TIER_7	8	H'FFB24	TPU_7	16	2F
Timer status register_7	TSR_7	8	H'FFB25	TPU_7	16	2F
Timer counter_7	TCNT_7	16	H'FFB26	TPU_7	16	2F
Timer general register A_7	TGRA_7	16	H'FFB28	TPU_7	16	2F
Timer general register B_7	TGRB_7	16	H'FFB2A	TPU_7	16	2F
Timer control register_8	TCR_8	8	H'FFB30	TPU_8	16	2F
Timer mode register_8	TMDR_8	8	H'FFB31	TPU_8	16	2F
Timer I/O control register_8	TIOR_8	8	H'FFB32	TPU_8	16	2F
Timer interrupt enable register_8	TIER_8	8	H'FFB34	TPU_8	16	2F
Timer status register_8	TSR_8	8	H'FFB35	TPU_8	16	2F
Timer counter_8	TCNT_8	16	H'FFB36	TPU_8	16	2F
Timer general register A_8	TGRA_8	16	H'FFB38	TPU_8	16	2F
Timer general register B_8	TGRB_8	16	H'FFB3A	TPU_8	16	2F
Timer control register_9	TCR_9	8	H'FFB40	TPU_9	16	2F
Timer mode register_9	TMDR_9	8	H'FFB41	TPU_9	16	2F

Timer general register C_9	TGRC_9	16	H'FFB4C	TPU_9	16
Timer general register D_9	TGRD_9	16	H'FFB4E	TPU_9	16
Timer control register_10	TCR_10	8	H'FFB50	TPU_10	16
Timer mode register_10	TMDR_10	8	H'FFB51	TPU_10	16
Timer I/O control register_10	TIOR_10	8	H'FFB52	TPU_10	16
Timer interrupt enable register_10	TIER_10	8	H'FFB54	TPU_10	16
Timer status register_10	TSR_10	8	H'FFB55	TPU_10	16
Timer counter_10	TCNT_10	16	H'FFB56	TPU_10	16
Timer general register A_10	TGRA_10	16	H'FFB58	TPU_10	16
Timer general register B_10	TGRB_10	16	H'FFB5A	TPU_10	16
Timer control register_11	TCR_11	8	H'FFB60	TPU_11	16
Timer mode register_11	TMDR_11	8	H'FFB61	TPU_11	16
Timer I/O control register_11	TIOR_11	8	H'FFB62	TPU_11	16
Timer interrupt enable register_11	TIER_11	8	H'FFB64	TPU_11	16
Timer status register_11	TSR_11	8	H'FFB65	TPU_11	16
Timer counter_11	TCNT_11	16	H'FFB66	TPU_11	16
Timer general register A_11	TGRA_11	16	H'FFB68	TPU_11	16
Timer general register B_11	TGRB_11	16	H'FFB6A	TPU_11	16
Port 1 data direction register	P1DDR	8	H'FFB80	I/O port	8
Port 2 data direction register	P2DDR	8	H'FFB81	I/O port	8
Port 3 data direction register	P3DDR	8	H'FFB82	I/O port	8
Port 6 data direction register	P6DDR	8	H'FFB85	I/O port	8
Port A data direction register	PADDR	8	H'FFB89	I/O port	8
Port B data direction register	PBDDR	8	H'FFB8A	I/O port	8

Port 5 input buffer control register	P5ICR	8	H'FFB94	I/O port	8	2F
Port 6 input buffer control register	P6ICR	8	H'FFB95	I/O port	8	2F
Port A input buffer control register	PAICR	8	H'FFB99	I/O port	8	2F
Port B input buffer control register	PBICR	8	H'FFB9A	I/O port	8	2F
Port C input buffer control register	PCICR	8	H'FFB9B	I/O port	8	2F
Port D input buffer control register	PDICR	8	H'FFB9C	I/O port	8	2F
Port E input buffer control register	PEICR	8	H'FFB9D	I/O port	8	2F
Port F input buffer control register	PFICR	8	H'FFB9E	I/O port	8	2F
Port H register	PORTH	8	H'FFBA0	I/O port	8	2F
Port I register	PORTI	8	H'FFBA1	I/O port	8	2F
Port J register	PORTJ	8	H'FFBA2	I/O port	8	2F
Port K register	PORTK	8	H'FFBA3	I/O port	8	2F
Port H data register	PHDR	8	H'FFBA4	I/O port	8	2F
Port I data register	PIDR	8	H'FFBA5	I/O port	8	2F
Port J data register	PJDR	8	H'FFBA6	I/O port	8	2F
Port K data register	PKDR	8	H'FFBA7	I/O port	8	2F
Port H data direction register	PHDDR	8	H'FFBA8	I/O port	8	2F
Port I data direction register	PIDDR	8	H'FFBA9	I/O port	8	2F
Port J data direction register	PJDDR	8	H'FFBAA	I/O port	8	2F
Port K data direction register	PKDDR	8	H'FFBAB	I/O port	8	2F
Port H input buffer control register	PHICR	8	H'FFBAC	I/O port	8	2F
Port I input buffer control register	PIICR	8	H'FFBAD	I/O port	8	2F
Port J input buffer control register	PJICR	8	H'FFBAE	I/O port	8	2F
Port K input buffer control register	PKICR	8	H'FFBAF	I/O port	8	2F



Port 2 open-drain control register	P2ODR	8	H'FFBBC	I/O port	8
Port F open-drain control register	PFODR	8	H'FFBBD	I/O port	8
Port function control register 0	PFCR0	8	H'FFBC0	I/O port	8
Port function control register 1	PFCR1	8	H'FFBC1	I/O port	8
Port function control register 2	PFCR2	8	H'FFBC2	I/O port	8
Port function control register 4	PFCR4	8	H'FFBC4	I/O port	8
Port function control register 6	PFCR6	8	H'FFBC6	I/O port	8
Port function control register 7	PFCR7	8	H'FFBC7	I/O port	8
Port function control register 8	PFCR8	8	H'FFBC8	I/O port	8
Port function control register 9	PFCR9	8	H'FFBC9	I/O port	8
Port function control register A	PFCRA	8	H'FFBCA	I/O port	8
Port function control register B	PFCRB	8	H'FFBCB	I/O port	8
Port function control register C	PFCRC	8	H'FFBCC	I/O port	8
Port function control register D	PFCRD	8	H'FFBCD	I/O port	8
Software standby release IRQ enable register	SSIER	16	H'FFBCE	INTC	8
Deep standby backup register 0	DPSBKR0	8	H'FFBF0	SYSTEM	8
Deep standby backup register 1	DPSBKR1	8	H'FFBF1	SYSTEM	8
Deep standby backup register 2	DPSBKR2	8	H'FFBF2	SYSTEM	8
Deep standby backup register 3	DPSBKR3	8	H'FFBF3	SYSTEM	8
Deep standby backup register 4	DPSBKR4	8	H'FFBF4	SYSTEM	8
Deep standby backup register 5	DPSBKR5	8	H'FFBF5	SYSTEM	8
Deep standby backup register 6	DPSBKR6	8	H'FFBF6	SYSTEM	8
Deep standby backup register 7	DPSBKR7	8	H'FFBF7	SYSTEM	8
Deep standby backup register 8	DPSBKR8	8	H'FFBF8	SYSTEM	8

DMA source address register_0	DSAR_0	32	H'FFC00	DMAC_0	16	21
DMA destination address register_0	DDAR_0	32	H'FFC04	DMAC_0	16	21
DMA offset register_0	DOFR_0	32	H'FFC08	DMAC_0	16	21
DMA transfer count register_0	DTCR_0	32	H'FFC0C	DMAC_0	16	21
DMA block size register_0	DBSR_0	32	H'FFC10	DMAC_0	16	21
DMA mode control register_0	DMDR_0	32	H'FFC14	DMAC_0	16	21
DMA address control register_0	DACR_0	32	H'FFC18	DMAC_0	16	21
DMA source address register_1	DSAR_1	32	H'FFC20	DMAC_1	16	21
DMA destination address register_1	DDAR_1	32	H'FFC24	DMAC_1	16	21
DMA offset register_1	DOFR_1	32	H'FFC28	DMAC_1	16	21
DMA transfer count register_1	DTCR_1	32	H'FFC2C	DMAC_1	16	21
DMA block size register_1	DBSR_1	32	H'FFC30	DMAC_1	16	21
DMA mode control register_1	DMDR_1	32	H'FFC34	DMAC_1	16	21
DMA address control register_1	DACR_1	32	H'FFC38	DMAC_1	16	21
DMA source address register_2	DSAR_2	32	H'FFC40	DMAC_2	16	21
DMA destination address register_2	DDAR_2	32	H'FFC44	DMAC_2	16	21
DMA offset register_2	DOFR_2	32	H'FFC48	DMAC_2	16	21
DMA transfer count register_2	DTCR_2	32	H'FFC4C	DMAC_2	16	21
DMA block size register_2	DBSR_2	32	H'FFC50	DMAC_2	16	21
DMA mode control register_2	DMDR_2	32	H'FFC54	DMAC_2	16	21
DMA address control register_2	DACR_2	32	H'FFC58	DMAC_2	16	21
DMA source address register_3	DSAR_3	32	H'FFC60	DMAC_3	16	21
DMA destination address register_3	DDAR_3	32	H'FFC64	DMAC_3	16	21
DMA offset register_3	DOFR_3	32	H'FFC68	DMAC_3	16	21

EXDMA transfer count register_0	EDTCR_0	32	H'FFC8C	EXDMAC_0	16
EXDMA block size register_0	EDBSR_0	32	H'FFC90	EXDMAC_0	16
EXDMA mode control register_0	EDMDR_0	32	H'FFC94	EXDMAC_0	16
EXDMA address control register_0	EDACR_0	32	H'FFC98	EXDMAC_0	16
EXDMA source address register_1	EDSAR_1	32	H'FFCA0	EXDMAC_1	16
EXDMA destination address register_1	EDDAR_1	32	H'FFCA4	EXDMAC_1	16
EXDMA offset register_1	EDOFR_1	32	H'FFCA8	EXDMAC_1	16
EXDMA transfer count register_1	EDTCR_1	32	H'FFCAC	EXDMAC_1	16
EXDMA block size register_1	EDBSR_1	32	H'FFCB0	EXDMAC_1	16
EXDMA mode control register_1	EDMDR_1	32	H'FFCB4	EXDMAC_1	16
EXDMA address control register_1	EDACR_1	32	H'FFCB8	EXDMAC_1	16
EXDMA source address register_2	EDSAR_2	32	H'FFCC0	EXDMAC_2	16
EXDMA destination address register_2	EDDAR_2	32	H'FFCC4	EXDMAC_2	16
EXDMA offset register_2	EDOFR_2	32	H'FFCC8	EXDMAC_2	16
EXDMA transfer count register_2	EDTCR_2	32	H'FFCCC	EXDMAC_2	16
EXDMA block size register_2	EDBSR_2	32	H'FFCD0	EXDMAC_2	16
EXDMA mode control register_2	EDMDR_2	32	H'FFCD4	EXDMAC_2	16
EXDMA address control register_2	EDACR_2	32	H'FFCD8	EXDMAC_2	16
EXDMA source address register_3	EDSAR_3	32	H'FFCE0	EXDMAC_3	16
EXDMA destination address register_3	EDDAR_3	32	H'FFCE4	EXDMAC_3	16
EXDMA offset register_3	EDOFR_3	32	H'FFCE8	EXDMAC_3	16
EXDMA transfer count register_3	EDTCR_3	32	H'FFCEC	EXDMAC_3	16
EXDMA block size register_3	EDBSR_3	32	H'FFCF0	EXDMAC_3	16
EXDMA mode control register_3	EDMDR_3	32	H'FFCF4	EXDMAC_3	16

Cluster buffer register 6	CLSBR6	32	H'FFD18	EXDMAC	16	21
Cluster buffer register 7	CLSBR7	32	H'FFD1C	EXDMAC	16	21
DMA module request select register_0	DMRSR_0	8	H'FFD20	DMAC_0	16	21
DMA module request select register_1	DMRSR_1	8	H'FFD21	DMAC_1	16	21
DMA module request select register_2	DMRSR_2	8	H'FFD22	DMAC_2	16	21
DMA module request select register_3	DMRSR_3	8	H'FFD23	DMAC_3	16	21
Interrupt priority register A	IPRA	16	H'FFD40	INTC	16	21
Interrupt priority register B	IPRB	16	H'FFD42	INTC	16	21
Interrupt priority register C	IPRC	16	H'FFD44	INTC	16	21
Interrupt priority register D	IPRD	16	H'FFD46	INTC	16	21
Interrupt priority register E	IPRE	16	H'FFD48	INTC	16	21
Interrupt priority register F	IPRF	16	H'FFD4A	INTC	16	21
Interrupt priority register G	IPRG	16	H'FFD4C	INTC	16	21
Interrupt priority register H	IPRH	16	H'FFD4E	INTC	16	21
Interrupt priority register I	IPRI	16	H'FFD50	INTC	16	21
Interrupt priority register J	IPRJ	16	H'FFD52	INTC	16	21
Interrupt priority register K	IPRK	16	H'FFD54	INTC	16	21
Interrupt priority register L	IPRL	16	H'FFD56	INTC	16	21
Interrupt priority register M	IPRM	16	H'FFD58	INTC	16	21
Interrupt priority register N	IPRN	16	H'FFD5A	INTC	16	21
Interrupt priority register O	IPRO	16	H'FFD5C	INTC	16	21
Interrupt priority register Q	IPRQ	16	H'FFD60	INTC	16	21
Interrupt priority register R	IPRR	16	H'FFD62	INTC	16	21
IRQ sense control register H	ISCRH	16	H'FFD68	INTC	16	21

CS assertion period control register	CSACR	16	H'FFD8E	BSC	16
Idle control register	IDLCR	16	H'FFD90	BSC	16
Bus control register 1	BCR1	16	H'FFD92	BSC	16
Bus control register 2	BCR2	8	H'FFD94	BSC	16
Endian control register	ENDIANCR	8	H'FFD95	BSC	16
SRAM mode control register	SRAMCR	16	H'FFD98	BSC	16
Burst ROM interface control register	BROMCR	16	H'FFD9A	BSC	16
Address/data multiplexed I/O control register	MPXCR	16	H'FFD9C	BSC	16
DRAM control register	DRAMCR	16	H'FFDA0	BSC	16
DRAM access control register	DRACCR	16	H'FFDA2	BSC	16
Synchronous DRAM control register	SDCR	16	H'FFDA4	BSC	16
Refresh control register	REFCR	16	H'FFDA6	BSC	16
Refresh timer counter	RTCNT	8	H'FFDA8	BSC	16
Refresh time constant register	RTCOR	8	H'FFDA9	BSC	16
RAM emulation register	RAMER	8	H'FFD9E	BSC	16
Mode control register	MDCR	16	H'FFDC0	SYSTEM	16
System control register	SYSCR	16	H'FFDC2	SYSTEM	16
System clock control register	SCKCR	16	H'FFDC4	SYSTEM	16
Standby control register	SBYCR	16	H'FFDC6	SYSTEM	16
Module stop control register A	MSTPCRA	16	H'FFDC8	SYSTEM	16
Module stop control register B	MSTPCRB	16	H'FFDCA	SYSTEM	16
Module stop control register C	MSTPCRC	16	H'FFDCC	SYSTEM	16
Subclock control register	SUBCKCR	8	H'FFDCF	SYSTEM	16
Flash code control/status register	FCCS	8	H'FFDE8	FLASH	16

Deep standby interrupt enable register	DPSIER	8	H'FFE72	SYSTEM	8	2F
Deep standby interrupt flag register	DPSIFR	8	H'FFE73	SYSTEM	8	2F
Deep standby interrupt edge register	DPSIEGR	8	H'FFE74	SYSTEM	8	2F
Reset status register	RSTSR	8	H'FFE75	SYSTEM	8	2F
Low voltage detection control register* <sup>2</sup>	LVDCCR	8	H'FFE78	SYSTEM	8	2F
Serial extended mode register_2	SEMR_2	8	H'FFE84	SCI_2	8	2F
Serial mode register_4	SMR_4	8	H'FFE90	SCI_4	8	2F
Bit rate register_4	BRR_4	8	H'FFE91	SCI_4	8	2F
Serial control register_4	SCR_4	8	H'FFE92	SCI_4	8	2F
Transmit data register_4	TDR_4	8	H'FFE93	SCI_4	8	2F
Serial status register_4	SSR_4	8	H'FFE94	SCI_4	8	2F
Receive data register_4	RDR_4	8	H'FFE95	SCI_4	8	2F
Smart card mode register_4	SCMR_4	8	H'FFE96	SCI_4	8	2F
I <sup>2</sup> C bus control register A_0	ICCRA_0	8	H'FFEB0	IIC2_0	8	2F
I <sup>2</sup> C bus control register B_0	ICCRB_0	8	H'FFEB1	IIC2_0	8	2F
I <sup>2</sup> C bus mode register_0	ICMR_0	8	H'FFEB2	IIC2_0	8	2F
I <sup>2</sup> C bus interrupt enable register_0	ICIER_0	8	H'FFEB3	IIC2_0	8	2F
I <sup>2</sup> C bus status register_0	ICSR_0	8	H'FFEB4	IIC2_0	8	2F
Slave address register_0	SAR_0	8	H'FFEB5	IIC2_0	8	2F
I <sup>2</sup> C bus transmit data register_0	ICDRT_0	8	H'FFEB6	IIC2_0	8	2F
I <sup>2</sup> C bus receive data register_0	ICDRR_0	8	H'FFEB7	IIC2_0	8	2F
I <sup>2</sup> C bus control register A_1	ICCRA_1	8	H'FFEB8	IIC2_1	8	2F
I <sup>2</sup> C bus control register B_1	ICCRB_1	8	H'FFEB9	IIC2_1	8	2F
I <sup>2</sup> C bus mode register_1	ICMR_1	8	H'FFEBA	IIC2_1	8	2F

Timer control/status register_2	TCSR_2	8	H'FFEC2	TMR_2	16
Timer control/status register_3	TCSR_3	8	H'FFEC3	TMR_3	16
Time constant register A_2	TCORA_2	8	H'FFEC4	TMR_2	16
Time constant register A_3	TCORA_3	8	H'FFEC5	TMR_3	16
Time constant register B_2	TCORB_2	8	H'FFEC6	TMR_2	16
Time constant register B_3	TCORB_3	8	H'FFEC7	TMR_3	16
Timer counter_2	TCNT_2	8	H'FFEC8	TMR_2	16
Timer counter_3	TCNT_3	8	H'FFEC9	TMR_3	16
Timer counter control register_2	TCCR_2	8	H'FFECA	TMR_2	16
Timer counter control register_3	TCCR_3	8	H'FFECB	TMR_3	16
Timer control register_4	TCR_4	8	H'FFEE0	TPU_4	16
Timer mode register_4	TMDR_4	8	H'FFEE1	TPU_4	16
Timer I/O control register_4	TIOR_4	8	H'FFEE2	TPU_4	16
Timer interrupt enable register_4	TIER_4	8	H'FFEE4	TPU_4	16
Timer status register_4	TSR_4	8	H'FFEE5	TPU_4	16
Timer counter_4	TCNT_4	16	H'FFEE6	TPU_4	16
Timer general register A_4	TGRA_4	16	H'FFEE8	TPU_4	16
Timer general register B_4	TGRB_4	16	H'FFEEA	TPU_4	16
Timer control register_5	TCR_5	8	H'FFEF0	TPU_5	16
Timer mode register_5	TMDR_5	8	H'FFEF1	TPU_5	16
Timer I/O control register_5	TIOR_5	8	H'FFEF2	TPU_5	16
Timer interrupt enable register_5	TIER_5	8	H'FFEF4	TPU_5	16
Timer status register_5	TSR_5	8	H'FFEF5	TPU_5	16
Timer counter_5	TCNT_5	16	H'FFEF6	TPU_5	16

DTC enable register F	DTCERF	16	H'FFF2A	INTC	16	2F
DTC control register	DTCCR	8	H'FFF30	INTC	16	2F
Interrupt control register	INTCR	8	H'FFF32	INTC	16	2F
CPU priority control register	CPUPCR	8	H'FFF33	INTC	16	2F
IRQ enable register	IER	16	H'FFF34	INTC	16	2F
IRQ status register	ISR	16	H'FFF36	INTC	16	2F
Port 1 register	PORT1	8	H'FFF40	I/O port	8	2F
Port 2 register	PORT2	8	H'FFF41	I/O port	8	2F
Port 3 register	PORT3	8	H'FFF42	I/O port	8	2F
Port 5 register	PORT5	8	H'FFF44	I/O port	8	2F
Port 6 register	PORT6	8	H'FFF45	I/O port	8	2F
Port A register	PORTA	8	H'FFF49	I/O port	8	2F
Port B register	PORTB	8	H'FFF4A	I/O port	8	2F
Port C register	PORTC	8	H'FFF4B	I/O port	8	2F
Port D register	PORTD	8	H'FFF4C	I/O port	8	2F
Port E register	PORTE	8	H'FFF4D	I/O port	8	2F
Port F register	PORTF	8	H'FFF4E	I/O port	8	2F
Port 1 data register	P1DR	8	H'FFF50	I/O port	8	2F
Port 2 data register	P2DR	8	H'FFF51	I/O port	8	2F
Port 3 data register	P3DR	8	H'FFF52	I/O port	8	2F
Port 6 data register	P6DR	8	H'FFF55	I/O port	8	2F
Port A data register	PADR	8	H'FFF59	I/O port	8	2F
Port B data register	PBDR	8	H'FFF5A	I/O port	8	2F
Port C data register	PCDR	8	H'FFF5B	I/O port	8	2F



Serial status register_2	SSR_2	8	H'FFF64	SCI_2	8
Receive data register_2	RDR_2	8	H'FFF65	SCI_2	8
Smart card mode register_2	SCMR_2	8	H'FFF66	SCI_2	8
D/A data register 0H	DADR0H	8	H'FFF68	D/A	8
D/A data register 1H	DADR1H	8	H'FFF69	D/A	8
D/A control register 01	DACR01	8	H'FFF6A	D/A	8
D/A data register 01T	DADR01T	8	H'FFF6B	D/A	8
PPG output control register	PCR	8	H'FFF76	PPG_0	8
PPG output mode register	PMR	8	H'FFF77	PPG_0	8
Next data enable register H	NDERH	8	H'FFF78	PPG_0	8
Next data enable register L	NDERL	8	H'FFF79	PPG_0	8
Output data register H	PODRH	8	H'FFF7A	PPG_0	8
Output data register L	PODRL	8	H'FFF7B	PPG_0	8
Next data register H <sup>*1</sup>	NDRH	8	H'FFF7C	PPG_0	8
Next data register L <sup>*1</sup>	NDRL	8	H'FFF7D	PPG_0	8
Next data register H <sup>*1</sup>	NDRH	8	H'FFF7E	PPG_0	8
Next data register L <sup>*1</sup>	NDRL	8	H'FFF7F	PPG_0	8
Serial mode register_0	SMR_0	8	H'FFF80	SCI_0	8
Bit rate register_0	BRR_0	8	H'FFF81	SCI_0	8
Serial control register_0	SCR_0	8	H'FFF82	SCI_0	8
Transmit data register_0	TDR_0	8	H'FFF83	SCI_0	8
Serial status register_0	SSR_0	8	H'FFF84	SCI_0	8
Receive data register_0	RDR_0	8	H'FFF85	SCI_0	8
Smart card mode register_0	SCMR_0	8	H'FFF86	SCI_0	8

A/D data register A_0	ADDRA_0	16	H'FFF90	A/D_0	16	2F
A/D data register B_0	ADDRB_0	16	H'FFF92	A/D_0	16	2F
A/D data register C_0	ADDRC_0	16	H'FFF94	A/D_0	16	2F
A/D data register D_0	ADDRD_0	16	H'FFF96	A/D_0	16	2F
A/D data register E_0	ADDRE_0	16	H'FFF98	A/D_0	16	2F
A/D data register F_0	ADDRF_0	16	H'FFF9A	A/D_0	16	2F
A/D data register G_0	ADDRG_0	16	H'FFF9C	A/D_0	16	2F
A/D data register H_0	ADDRH_0	16	H'FFF9E	A/D_0	16	2F
A/D control/status register_0	ADCSR_0	8	H'FFFA0	A/D_0	16	2F
A/D control register_0	ADCR_0	8	H'FFFA1	A/D_0	16	2F
A/D mode selection register_0	ADMOSEL_0	8	H'FFFA2	A/D_0	16	2F
Timer control/status register	TCSR	8	H'FFFA4	WDT	16	2F
Timer counter	TCNT	8	H'FFFA5	WDT	16	2F
Reset control/status register	RSTCSR	8	H'FFFA7	WDT	16	2F
Timer control register_0	TCR_0	8	H'FFFB0	TMR_0	16	2F
Timer control register_1	TCR_1	8	H'FFFB1	TMR_1	16	2F
Timer control/status register_0	TCSR_0	8	H'FFFB2	TMR_0	16	2F
Timer control/status register_1	TCSR_1	8	H'FFFB3	TMR_1	16	2F
Time constant register A_0	TCORA_0	8	H'FFFB4	TMR_0	16	2F
Time constant register A_1	TCORA_1	8	H'FFFB5	TMR_1	16	2F
Time constant register B_0	TCORB_0	8	H'FFFB6	TMR_0	16	2F
Time constant register B_1	TCORB_1	8	H'FFFB7	TMR_1	16	2F
Timer counter_0	TCNT_0	8	H'FFFB8	TMR_0	16	2F
Timer counter_1	TCNT_1	8	H'FFFB9	TMR_1	16	2F

Timer I/O control register L_0	TIORL_0	8	H'FFFC3	TPU_0	16
Timer interrupt enable register_0	TIER_0	8	H'FFFC4	TPU_0	16
Timer status register_0	TSR_0	8	H'FFFC5	TPU_0	16
Timer counter_0	TCNT_0	16	H'FFFC6	TPU_0	16
Timer general register A_0	TGRA_0	16	H'FFFC8	TPU_0	16
Timer general register B_0	TGRB_0	16	H'FFFC9	TPU_0	16
Timer general register C_0	TGRC_0	16	H'FFFC	TPU_0	16
Timer general register D_0	TGRD_0	16	H'FFFC	TPU_0	16
Timer control register_1	TCR_1	8	H'FFFD0	TPU_1	16
Timer mode register_1	TMDR_1	8	H'FFFD1	TPU_1	16
Timer I/O control register_1	TIOR_1	8	H'FFFD2	TPU_1	16
Timer interrupt enable register_1	TIER_1	8	H'FFFD4	TPU_1	16
Timer status register_1	TSR_1	8	H'FFFD5	TPU_1	16
Timer counter_1	TCNT_1	16	H'FFFD6	TPU_1	16
Timer general register A_1	TGRA_1	16	H'FFFD8	TPU_1	16
Timer general register B_1	TGRB_1	16	H'FFFDA	TPU_1	16
Timer control register_2	TCR_2	8	H'FFFE0	TPU_2	16
Timer mode register_2	TMDR_2	8	H'FFFE1	TPU_2	16
Timer I/O control register_2	TIOR_2	8	H'FFFE2	TPU_2	16
Timer interrupt enable register_2	TIER_2	8	H'FFFE4	TPU_2	16
Timer status register_2	TSR_2	8	H'FFFE5	TPU_2	16
Timer counter_2	TCNT_2	16	H'FFFE6	TPU_2	16
Timer general register A_2	TGRA_2	16	H'FFFE8	TPU_2	16
Timer general register B_2	TGRB_2	16	H'FFFEA	TPU_2	16

Timer general register A_3	TGRA_3	16	H'FFFF8	TPU_3	16	2F
Timer general register B_3	TGRB_3	16	H'FFFFA	TPU_3	16	2F
Timer general register C_3	TGRC_3	16	H'FFFFC	TPU_3	16	2F
Timer general register D_3	TGRD_3	16	H'FFFFE	TPU_3	16	2F

Notes: 1. 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.

When the same output trigger is specified for pulse output groups 6 and 7 by the PCR setting, the NDRH address is H'FF63C. When different output triggers are specified, the NDRH addresses for pulse output groups 6 and 7 are H'FF63E and H'FF63C, respectively.

When the same output trigger is specified for pulse output groups 4 and 5 by the PCR setting, the NDRL address is H'FF63D. When different output triggers are specified, the NDRL addresses for pulse output groups 4 and 5 are H'FF63F and H'FF63D, respectively.

2. Supported only by the H8SX/1665M Group.

TCSR_4	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0
TCORA_4								
TCORA_5								
TCORB_4								
TCORB_5								
TCNT_4								
TCNT_5								
TCCR_4	—	—	—	—	TMRIS	—	ICKS1	ICKS0
TCCR_5	—	—	—	—	TMRIS	—	ICKS1	ICKS0
CRCCR	DORCLR	—	—	—	—	LMS	G1	G0
CRCDIR								
CRCDOR								
TCR_6	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
TCR_7	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
TCSR_6	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0
TCSR_7	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0
TCORA_6								
TCORA_7								
TCORB_6								
TCORB_7								
TCNT_6								
TCNT_7								

---

ADDRD\_1

---

ADDRE\_1

---

ADDRF\_1

---

ADDRG\_1

---

ADDRH\_1

---

---

ADCSR_1	ADF	ADIE	ADST	EXCKS	CH3	CH2	CH1	CH0
ADCR_1	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	ADSTCLR	EXTRGS
ADMOSEL_1	—	—	—	—	—	—	ICKSEL	—
ADSSTR_1	SMP7	SMP6	SMP5	SMP4	SMP3	SMP2	SMP1	SMP0
IFR0	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS
IFR1	—	—	—	—	VBUS MN	EP3 TR	EP3 TS	VBUSF
IFR2	—	—	SURSS	SURSF	CFDN	—	SETC	SETI
IER0	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS
IER1	—	—	—	—	—	EP3 TR	EP3 TS	VBUSF
IER2	SSRSME	—	—	SURSE	CFDN	—	SETCE	SETIE
ISR0	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS

---

EPDR3	D7	D6	D5	D4	D3	D2	D1	D0
EPSZ0o	—	—	—	—	—	—	—	—
EPSZ1	—	—	—	—	—	—	—	—
DASTS	—	—	EP3 DE	EP2 DE	—	—	—	EP0i DE
FCLR	—	EP3 CLR	EP1 CLR	EP2 CLR	—	—	EP0o CLR	EP0i CLR
EPSTL	—	—	—	—	EP3STL	EP2STL	EP1STL	EP0STL
TRG	—	EP3 PKTE	EP1 RDFN	EP2 PKTE	—	EP0s RDFN	EP0o RDFN	EP0i PKTE
DMA	—	—	—	—	—	PULLUP_E	EP2DMAE	EP1DMAE
CVR	CNFV1	CNFV0	INTV1	INTV0	—	ALTV2	ALTV1	ALTV0
CTLR	—	—	—	RWUPS	RSME	RWMD	ASCE	—
EPIR	D7	D6	D5	D4	D3	D2	D1	D0
TRNTREG0	PTSTE	—	—	—	SUSPEND	txenI	txse0	txdata
TRNTREG1	—	—	—	—	—	xver_data	dpls	dmns
PMDDR	—	—	—	PM4DDR	PM3DDR	PM2DDR	PM1DDR	PM0DDR
PMDR	—	—	—	PM4DR	PM3DR	PM2DR	PM1DR	PM0DR
PORTM	—	—	—	PM4	PM3	PM2	PM1	PM0
PMICR	—	—	—	PM4ICR	PM3ICR	PM2ICR	PM1ICR	PM0ICR
SMR_5* <sup>1</sup>	C/ $\bar{A}$	CHR	PE	O/ $\bar{E}$	STOP	MP	CKS1	CKS0
	(GM)	(BLK)	(PE)	(O/ $\bar{E}$ )	(BCP1)	(BCP0)		
BRR_5								
SCR_5* <sup>1</sup>	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
TDR_5								

	(GM)	(BLK)	(PE)	(O/Ē)	(BCP1)	(BCP0)		
BRR_6								
SCR_6* <sup>1</sup>	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
TDR_6								
SSR_6* <sup>1</sup>	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
				(ERS)				
RDR_6								
SCMR_6	—	—	—	—	SDIR	SINV	—	SMIF
SEMR_6	—	—	—	ABCS	ACS3	ACS2	ACS1	ACS0
PCR_1	G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0
PMR_1	G3INV	G2INV	G1INV	G0INV	G3NOV	G2NOV	G1NOV	G0NOV
NDERH_1	NDER31	NDER30	NDER29	NDER28	NDER27	NDER26	NDER25	NDER24
NDERL_1	NDER23	NDER22	NDER21	NDER20	NDER19	NDER18	NDER17	NDER16
PODRH_1	POD31	POD30	POD29	POD28	POD27	POD26	POD25	POD24
PODRL_1	POD23	POD22	POD21	POD20	POD19	POD18	POD17	POD16
NDRH_1* <sup>2</sup>	NDR31	NDR30	NDR29	NDR28	NDR27	NDR26	NDR25	NDR24
NDRL_1* <sup>2</sup>	NDR23	NDR22	NDR21	NDR20	NDR19	NDR18	NDR17	NDR16
NDRH_1* <sup>2</sup>	—	—	—	—	NDR27	NDR26	NDR25	NDR24
NDRL_1* <sup>2</sup>	—	—	—	—	NDR19	NDR18	NDR17	NDR16



	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
BARBL	BARB15	BARB14	BARB13	BARB12	BARB11	BARB10	BARB9	BARB8
	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
BRCRA	—	—	CMFCPA	—	CPA2	CPA1	CPA0	—
	—	—	IDA1	IDA0	RWA1	RWA0	—	—

TCNT32K1

TCNT32K2

TCNT32K3

ADSSSTR_0	SMP7	SMP6	SMP5	SMP4	SMP3	SMP2	SMP1	SMP0
TSTRB	—	—	CST5	CST4	CST3	CST2	CST1	CST0
TSYRB	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
TCR_6	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
TMDR_6	—	—	BFB	BFA	MD3	MD2	MD1	MD0
TIORH_6	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
TIORL_6	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
TIER_6	—	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA
TSR_6	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA

TCNT\_6

TGRA\_6

TGRB\_6

TGRC\_6

TGRD\_6

TGRA\_7

---

TGRB\_7

---

TCR_8	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
TMDR_8	—	—	—	—	MD3	MD2	MD1	MD0
TIOR_8	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
TIER_8	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA
TSR_8	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA
TCNT_8								

TGRA\_8

---

TGRB\_8

---

TCR_9	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
TMDR_9	—	—	BFB	BFA	MD3	MD2	MD1	MD0
TIORH_9	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
TIORL_9	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
TIER_9	—	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA
TSR_9	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA
TCNT_9								

TCR_10	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
TMDR_10	—	—	—	—	MD3	MD2	MD1	MD0
TIOR_10	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
TIER_10	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA
TSR_10	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA

TCNT\_10

---

TGRA\_10

---

TGRB\_10

---

TCR_11	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
TMDR_11	—	—	—	—	MD3	MD2	MD1	MD0
TIOR_11	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
TIER_11	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA
TSR_11	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA

TCNT\_11

---

TGRA\_11

---

TGRB\_11

---

PDDDR	PD7DDR	PD6DDR	PD5DDR	PD4DDR	PD3DDR	PD2DDR	PD1DDR	PD0DDR
PEDDR	PE7DDR	PE6DDR	PE5DDR	PE4DDR	PE3DDR	PE2DDR	PE1DDR	PE0DDR
PFDDR	PF7DDR	PF6DDR	PF5DDR	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
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
PBICR	PB7ICR	PB6ICR	PB5ICR	PB4ICR	PB3ICR	PB2ICR	PB1ICR	PB0ICR
PCICR	—	—	—	—	PC3ICR	PC2ICR	—	—
PDICR	PD7ICR	PD6ICR	PD5ICR	PD4ICR	PD3ICR	PD2ICR	PD1ICR	PD0ICR
PEICR	PE7ICR	PE6ICR	PE5ICR	PE4ICR	PE3ICR	PE2ICR	PE1ICR	PE0ICR
PFICR	PF7ICR	PF6ICR	PF5ICR	PF4ICR	PF3ICR	PF2ICR	PF1ICR	PF0ICR
PORTH	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0
PORTI	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0
PORTJ	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0
PORTK	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
PHDR	PH7DR	PH6DR	PH5DR	PH4DR	PH3DR	PH2DR	PH1DR	PH0DR
PIDR	PI7DR	PI6DR	PI5DR	PI4DR	PI3DR	PI2DR	PI1DR	PI0DR
PJDR	PJ7DR	PJ6DR	PJ5DR	PJ4DR	PJ3DR	PJ2DR	PJ1DR	PJ0DR
PKDR	PK7DR	PK6DR	PK5DR	PK4DR	PK3DR	PK2DR	PK1DR	PK0DR
PHDDR	PH7DDR	PH6DDR	PH5DDR	PH4DDR	PH3DDR	PH2DDR	PH1DDR	PH0DDR

PDPCR	PD7PCR	PD6PCR	PD5PCR	PD4PCR	PD3PCR	PD2PCR	PD1PCR	PD0PCR
PEPCR	PE7PCR	PE6PCR	PE5PCR	PE4PCR	PE3PCR	PE2PCR	PE1PCR	PE0PCR
PFPCR	PF7PCR	PF6PCR	PF5PCR	PF4PCR	PF3PCR	PF2PCR	PF1PCR	PF0PCR
PHPCR	PH7PCR	PH6PCR	PH5PCR	PH4PCR	PH3PCR	PH2PCR	PH1PCR	PH0PCR
PIPCR	PI7PCR	PI6PCR	PI5PCR	PI4PCR	PI3PCR	PI2PCR	PI1PCR	PI0PCR
PJPCR	PJ7PCR	PJ6PCR	PJ5PCR	PJ4PCR	PJ3PCR	PJ2PCR	PJ1PCR	PJ0PCR
PKPCR	PK7PCR	PK6PCR	PK5PCR	PK4PCR	PK3PCR	PK2PCR	PK1PCR	PK0PCR
P2ODR	P27ODR	P26ODR	P25ODR	P24ODR	P23ODR	P22ODR	P21ODR	P20ODR
PFODR	PF7ODR	PF6ODR	PF5ODR	PF4ODR	PF3ODR	PF2ODR	PF1ODR	PF0ODR
PFCR0	CS7E	CS6E	CS5E	CS4E	CS3E	CS2E	CS1E	CS0E
PFCR1	CS7SA	CS7SB	CS6SA	CS6SB	CS5SA	CS5SB	CS4SA	CS4SB
PFCR2	—	CS2S	BSS	BSE	RDWRS	RDWRE	ASOE	—
PFCR4	A23E	A22E	A21E	A20E	A19E	A18E	A17E	A16E
PFCR6	—	LHWROE	—	—	TCLKS	—	—	—
PFCR7	DMAS3A	DMAS3B	DMAS2A	DMAS2B	DMAS1A	DMAS1B	DMAS0A	DMAS0B
PFCR8	—	—	—	—	EDMAS1A	EDMAS1B	EDMAS0A	EDMAS0B
PFCR9	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B
PFCRA	TPUMS11	TPUMS10	TPUMS9A	TPUMS9B	TPUMS8	TPUMS7	TPUMS6A	TPUMS6B
PFCRB	—	—	—	—	ITS11	ITS10	ITS9	ITS8
PFCRC	ITS7	ITS6	ITS5	ITS4	ITS3	ITS2	ITS1	ITS0
PFCRD	PCJKE	—	—	—	—	—	—	—
SSIER	SSI15	—	—	—	SSI11	SSI10	SSI9	SSI8
	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0

DPSBKR7	DKUP77	DKUP76	DKUP75	DKUP74	DKUP73	DKUP72	DKUP71	DKUP70
DPSBKR8	DKUP87	DKUP86	DKUP85	DKUP84	DKUP83	DKUP82	DKUP81	DKUP80
DPSBKR9	DKUP97	DKUP96	DKUP95	DKUP94	DKUP93	DKUP92	DKUP91	DKUP90
DPSBKR10	DKUP107	DKUP106	DKUP105	DKUP104	DKUP103	DKUP102	DKUP101	DKUP100
DPSBKR11	DKUP117	DKUP116	DKUP115	DKUP114	DKUP113	DKUP112	DKUP111	DKUP110
DPSBKR12	DKUP127	DKUP126	DKUP125	DKUP124	DKUP123	DKUP122	DKUP121	DKUP120
DPSBKR13	DKUP137	DKUP136	DKUP135	DKUP134	DKUP133	DKUP132	DKUP131	DKUP130
DPSBKR14	DKUP147	DKUP146	DKUP145	DKUP144	DKUP143	DKUP142	DKUP141	DKUP140
DPSBKR15	DKUP157	DKUP156	DKUP155	DKUP154	DKUP153	DKUP152	DKUP151	DKUP150

DSAR\_0

---



---



---

DDAR\_0

---



---



---

DOFR\_0

---



---



---



---

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
DMDR_0	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
	ACT	—	—	—	—	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAPO
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	_____							
	_____							
	_____							
DDAR_1	_____							
	_____							
	_____							
DOFR_1	_____							
	_____							
	_____							



	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	DMAPO
DACR_1	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0

DSAR\_2

---



---



---

DDAR\_2

---



---



---

DOFR\_2

---



---



---



---

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
DMDR_2	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
	ACT	—	—	—	—	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAPO
DACR_2	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
DSAR_3	_____							
	_____							
	_____							
DDAR_3	_____							
	_____							
	_____							
DOFR_3	_____							
	_____							
	_____							

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
DMDR_3	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
	ACT	—	—	—	—	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAPO
DACR_3	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0

EDSAR\_0

---



---



---

EDDAR\_0

---



---



---

EDOFR\_0

---



---



---



---

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
EDMDR_0	DTE	EDACKE	ETENDE	EDRAKE	EDREQS	NRD	—	—
	ACT	—	—	—	ERRF	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	—	—	—	EDMAP2	DEMAP1	EDMAP0
EDACR_0	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
EDSAR_1	_____							
	_____							
	_____							
EDDAR_1	_____							
	_____							
	_____							
EDOFR_1	_____							
	_____							
	_____							

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
EDMDR_1	DTE	EDACKE	ETENDE	EDRAKE	EDREQS	NRD	—	—
	ACT	—	—	—	—	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	—	—	—	EDMAP2	DEMAP1	EDMAP0
EDACR_1	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0

EDSAR\_2

---



---



---

EDDAR\_2

---



---



---

EDOFR\_2

---



---



---



---



---



---

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
EDMDR_2	DTE	EDACKE	ETENDE	EDRAKE	EDREQS	NRD	—	—
	ACT	—	—	—	—	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	—	—	—	EDMAP2	DEMAP1	EDMAP0
EDACR_2	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
EDSAR_3	_____							
	_____							
	_____							
EDDAR_3	_____							
	_____							
	_____							
EDOFR_3	_____							
	_____							
	_____							

	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
EDMDR_3	DTE	EDACKE	ETENDE	EDRAKE	EDREQS	NRD	—	—
	ACT	—	—	—	—	—	ESIF	DTIF
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
	DTF1	DTF0	—	—	—	EDMAP2	DEMAP1	EDMAP0
EDACR_3	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0

CLSBR0

---



---



---

CLSBR1

---



---



---

CLSBR2

---



---



---



---

---

CLSBR5

---

---

---

---

CLSBR6

---

---

---

---

CLSBR7

---

---

---

---

DMRSR\_0

---

---

DMRSR\_1

---

---

DMRSR\_2

---

---

DMRSR\_3

---

IPRA	—	IPRA14	IPRA13	IPRA12	—	IPRA10	IPRA9	IPRA8
	—	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



	—	IPRG6	IPRG5	IPRG4	—	IPRG2	IPRG1	IPRG0
IPRH	—	IPRH14	IPRH13	IPRH12	—	IPRH10	IPRH9	IPRH8
	—	IPRH6	IPRH5	IPRH4	—	IPRH2	IPRH1	IPRH0
IPRI	—	IPRI14	IPRI13	IPRI12	—	IPRI10	IPRI9	IPRI8
	—	IPRI6	IPRI5	IPRI4	—	IPRI2	IPRI1	IPRI0
IPRJ	—	IPRJ4	IPRJ13	IPRJ12	—	IPRJ10	IPRJ9	IPRJ8
	—	IPRJ6	IPRJ5	IPRJ4	—	IPRJ2	IPRJ1	IPRJ0
IPRK	—	IPRK14	IPRK13	IPRK12	—	IPRK10	IPRK9	IPRK8
	—	IPRK6	IPRK5	IPRK4	—	IPRK2	IPRK1	IPRK0
IPRL	—	IPRL14	IPRL13	IPRL12	—	—	—	—
	—	IPRL6	IPRL5	IPRL4	—	IPRL2	IPRL1	IPRL0
IPRM	—	IPRM14	IPRM13	IPRM12	—	IPRM10	IPRM9	IPRM8
	—	IPRM6	IPRM5	IPRM4	—	IPRM2	IPRM1	IPRM0
IPRN	—	IPRN14	IPRN13	IPRN12	—	IPRN10	IPRN9	IPRN8
	—	IPRN6	IPRN5	IPRN4	—	IPRN2	IPRN1	IPRN0
IPRO	—	IPRO14	IPRO13	IPRO12	—	IPRO10	IPRO9	IPRO8
	—	IPRO6	IPRO5	IPRO4	—	—	—	—
IPRQ	—	—	—	—	—	—	—	—
	—	IPRQ6	IPRQ5	IPRQ4	—	IPRQ2	IPRQ1	IPRQ0
IPRR	—	IPRR14	IPRR13	IPRR12	—	IPRR10	IPRR9	IPRR8
	—	IPRR6	IPRR5	IPRR4	—	IPRR2	IPRR1	IPRR0
ISCRH	IRQ15SR	IRQ15SF	—	—	—	—	—	—
	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF

	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
	—	—	—	—	—	—	—	—
CSACR	CSXH7	CSXH6	CSXH5	CSXH4	CSXH3	CSXH2	CSXH1	CSXH0
	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	—	—	EBCCS	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

	RFSHE	RLW2	RLW1	RLW0	SLFRF	TPCS2	TPCS1	TPCS0
RTCNT								
RTCOR								
RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0
MDCR	MDS7	—	—	—	MDS3	MDS2	MDS1	MDS0
	—	—	—	—	—	—	—	—
SYSCR	—	—	MACS	—	FETCHMD	—	EXPE	RAME
	—	—	—	—	—	—	DTCMD	—
SCKCR	PSTOP1	PSTOP0	—	—	—	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	MSTPA0
MSTPCRB	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8
	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0
MSTPCRC	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8
	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0
SUBCKCR	—	—	—	—	—	EXSTP	WAKE32K	CK32K

DPSWCR	—	—	WTSTS5	WTSTS4	WTSTS3	WTSTS2	WTSTS1	WTSTS0
DPSIER	—	DUSBIE	DT32KIE	DLVDIE	DIRQ3E	DIRQ2E	DIRQ1E	DIRQ0E
DPSIFR	DNMIF	DUSBIF	DT32KIF	DLVDIF	DIRQ3F	DIRQ2F	DIRQ1F	DIRQ0F
DPSIEGR	DNMIEG	—	—	—	DIRQ3EG	DIRQ2EG	DIRQ1EG	DIRQ0EG
RSTSR	DPSRSTF	—	—	—	—	LVDF	—	PORF
LVDCR*3	LVDE	LVDRI	—	LVDMON	—	—	—	—
SEMR_2	—	—	—	—	ABCS	ACS2	ACS1	ACS0
SMR_4*1	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1	CKS0
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
ICCRA_0	ICE	RCVD	MST	TRS	CKS3	CKS2	CKS1	CKS0
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								

ICDRR_1								
TCR_2	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
TCR_3	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
TCSR_2	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0
TCSR_3	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0
TCORA_2								
TCORA_3								
TCORB_2								
TCORB_3								
TCNT_2								
TCNT_3								
TCCR_2	—	—	—	—	TMRIS	—	ICKS1	ICKS0
TCCR_3	—	—	—	—	TMRIS	—	ICKS1	ICKS0
TCR_4	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
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								

TCNT\_5

---

TGRA\_5

---

TGRB\_5

---

DTCERA	DTCEA15	DTCEA14	DTCEA13	DTCEA12	DTCEA11	DTCEA10	DTCEA9	DTCEA8
	DTCEA7	DTCEA6	DTCEA5	DTCEA4	—	—	—	—
DTCERB	DTCEB15	—	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	DTCEC3	DTCEC2	DTCEC1	DTCEC0
DTCERD	DTCED15	DTCED14	DTCED13	DTCED12	DTCED11	DTCED10	DTCED9	DTCED8
	DTCED7	DTCED6	DTCED5	DTCED4	DTCED3	DTCED2	DTCED1	DTCED0
DTCERE	—	—	DTCEE13	DTCEE12	DTCEE11	DTCEE10	DTCEE9	DTCEE8
	DTCEE7	DTCEE6	DTCEE5	DTCEE4	DTCEE3	DTCEE2	DTCEE1	DTCEE0
DTCERF	DTCEF15	DTCEF14	—	—	DTCEF11	DTCEF10	DTCEF9	—
	—	—	—	—	—	—	—	—
DTCCR	—	—	—	RRS	RCHNE	—	—	ERR
INTCR	—	—	INTM1	INTM0	NMIEG	—	—	—
CPUPCR	CPUPCE	DTCP2	DTCP1	DTCP0	IPSETE	CPUP2	CPUP1	CPUP0
IER	IRQ15E	—	—	—	IRQ11E	IRQ10E	IRQ9E	IRQ8E
	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E

PORTA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PORTB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PORTC	—	—	—	—	PC3	PC2	—	—
PORTD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PORTE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PORTF	PF7	PF6	PF5	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
PBDR	PB7DR	PB6DR	PB5DR	PB4DR	PB3DR	PB2DR	PB1DR	PB0DR
PCDR	—	—	—	—	PC3DR	PC2DR	—	—
PDDR	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR
PEDR	PE7DR	PE6DR	PE5DR	PE4DR	PE3DR	PE2DR	PE1DR	PE0DR
PFDR	PF7DR	PF6DR	PF5DR	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR

SCMR_2	—	—	—	—	SDIR	SINV	—	SMIF
DADRH								
DADR1H								
DACR01	DAOE1	DAOE0	DAE	—	—	—	—	—
DADR01T	DADT1	DADT0	DAD1L1	DAD1L0	DAD0L1	DAD0L0	—	—
PCR	G3CMS1	G3CMS0	G2CMS1	G2CMS0	G1CMS1	G1CMS0	G0CMS1	G0CMS0
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
NDRH* <sup>2</sup>	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8
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
BRR_0								
SCR_0* <sup>1</sup>	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
TDR_0								



TDR\_1

SSR_1*1	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT
---------	------	------	------	--------------	-----	------	-----	------

RDR\_1

SCMR_1	—	—	—	—	SDIR	SINV	—	SMIF
--------	---	---	---	---	------	------	---	------

ADDRA\_0

ADDRB\_0

ADDRC\_0

ADDRD\_0

ADDRE\_0

ADDRF\_0

ADDRG\_0

ADDRH\_0

TCR_1	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0
TCSR_0	CMFB	CMFA	OVF	ADTE	OS3	OS2	OS1	OS0
TCSR_1	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0
TCORA_0								
TCORA_1								
TCORB_0								
TCORB_1								
TCNT_0								
TCNT_1								
TCCR_0	—	—	—	—	TMRIS	—	ICKS1	ICKS0
TCCR_1	—	—	—	—	TMRIS	—	ICKS1	ICKS0
TSTR	—	—	CST5	CST4	CST3	CST2	CST1	CST0
TSYR	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
TCR_0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
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

TGRD\_0

---

TCR_1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
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								

TGRA\_1

---

TGRB\_1

---

TCR_2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
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

---

TCNT\_3

---

TGRA\_3

---

TGRB\_3

---

TGRC\_3

---

TGRD\_3

---

- Notes:
1. Parts of the bit functions differ in normal mode and the smart card interface.
  2. 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.  
 When the same output trigger is specified for pulse output groups 6 and 7 by the PCR setting, the NDRH address is H'FF63C. When different output triggers are specified, the NDRH addresses for pulse output groups 6 and 7 are H'FF63E and H'FF63C, respectively.  
 When the same output trigger is specified for pulse output groups 4 and 5 by the PCR setting, the NDRL address is H'FF63D. When different output triggers are specified, the NDRL addresses for pulse output groups 4 and 5 are H'FF63F and H'FF63D, respectively.
  3. Supported only by the H8SX/1665M Group.

TCORA_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORB_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORB_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCCR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCCR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
CRCCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
CRCDIR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
CRCDOR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCSR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCSR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORA_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORA_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORB_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORB_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCCR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCCR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

ADDRH_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADCSR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADCR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADMOSEL_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADSSTR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IFR0	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
IFR1	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
IFR2	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
IER0	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
IER1	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
IER2	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
ISR0	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
ISR1	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
ISR2	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPDR0i	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPDR0o	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPDR0s	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPDR1	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPDR2	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPDR3	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPSZ0o	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
EPSZ1	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
DASTS	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized

TRNTREG0	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
TRNTREG1	Initialized	—	—	—	—	Initialized* <sup>2</sup>	Initialized
PMDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PMDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PORTM	—	—	—	—	—	—	—
PMICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SMR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SCR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TDR_5	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SSR_5	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
RDR_5	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SCMR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SEMR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IrCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SMR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SCR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TDR_6	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SSR_6	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
RDR_6	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SCMR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SEMR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

NDRL_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARAH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARAL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRAH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRAL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARBH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARBL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRBH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRBL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARCH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARCL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRCH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRCL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARDH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BARDL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRDH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BAMRDL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRCRA	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRCRB	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRCRC	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRCRD	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized



TCR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIORH_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIORL_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRC_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRD_6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_8	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_8	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_8	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_8	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

TIORL_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRC_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRD_9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_10	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_11	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_11	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_11	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_11	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_11	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_11	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

PBDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PCDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PDDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PEDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P1ICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P2ICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P3ICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P5ICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P6ICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PAICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PBICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PCICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PDICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PEICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PORTH	—	—	—	—	—	—	—
PORTI	—	—	—	—	—	—	—
PORTJ	—	—	—	—	—	—	—
PORTK	—	—	—	—	—	—	—
PHDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PIDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PJDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

PJICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PKICR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PDPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PEPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PHPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PIPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PJPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PKPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P2ODR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFODR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR6	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR7	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR8	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PFCR9	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

DPSBKR2	Initialized	—	—	—	—	—	Initialized
DPSBKR3	Initialized	—	—	—	—	—	Initialized
DPSBKR4	Initialized	—	—	—	—	—	Initialized
DPSBKR5	Initialized	—	—	—	—	—	Initialized
DPSBKR6	Initialized	—	—	—	—	—	Initialized
DPSBKR7	Initialized	—	—	—	—	—	Initialized
DPSBKR8	Initialized	—	—	—	—	—	Initialized
DPSBKR9	Initialized	—	—	—	—	—	Initialized
DPSBKR10	Initialized	—	—	—	—	—	Initialized
DPSBKR11	Initialized	—	—	—	—	—	Initialized
DPSBKR12	Initialized	—	—	—	—	—	Initialized
DPSBKR13	Initialized	—	—	—	—	—	Initialized
DPSBKR14	Initialized	—	—	—	—	—	Initialized
DPSBKR15	Initialized	—	—	—	—	—	Initialized
DSAR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DDAR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DOFR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DTCR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DBSR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DMDR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DACR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

DSAR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DDAR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DOFR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DTCR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DBSR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DMDR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DACR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DSAR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DDAR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DOFR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DTCR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DBSR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DMDR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DACR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDSAR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDDAR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDOFR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDTCR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDBSR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDMDR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDACR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

EDSAR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDDAR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDOFR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDTCR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDBSR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDMDR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDACR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDSAR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDDAR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDOFR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDTCR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDBSR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDMDR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
EDACR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
CLSBR0	—	—	—	—	—	—	—
CLSBR1	—	—	—	—	—	—	—
CLSBR2	—	—	—	—	—	—	—
CLSBR3	—	—	—	—	—	—	—
CLSBR4	—	—	—	—	—	—	—
CLSBR5	—	—	—	—	—	—	—
CLSBR6	—	—	—	—	—	—	—
CLSBR7	—	—	—	—	—	—	—
DMRSR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

IPRE	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRF	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRG	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRI	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRJ	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRK	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRM	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRN	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRO	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRQ	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IPRR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ISCRH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ISURL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DTCVBR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ABWCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ASTCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
WTCRA	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
WTCRB	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
RDNCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
CSACR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IDLCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized



DRACCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SDCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
REFCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
RTCNT	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
RTCOR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
RAMER	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
MDCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SYSCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SCKCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SBYCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
MSTPCRA	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
MSTPCRB	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
MSTPCRC	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SUBCKCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
FCCS	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
FPCS	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
FECS	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
FKEY	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
FMATS	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
FTDAR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

SEMR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SMR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SCR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TDR_4	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SSR_4	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
RDR_4	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SCMR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICCRA_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICCRB_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICMR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICIER_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICSR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SAR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICDRT_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICDRR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICCRA_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICCRB_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICMR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICIER_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICSR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SAR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ICDRT_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

TCORB_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORB_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCCR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCCR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_4	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_5	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

INTCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
CPUPCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
IER	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ISR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PORT1	—	—	—	—	—	—	—
PORT2	—	—	—	—	—	—	—
PORT3	—	—	—	—	—	—	—
PORT5	—	—	—	—	—	—	—
PORT6	—	—	—	—	—	—	—
PORTA	—	—	—	—	—	—	—
PORTB	—	—	—	—	—	—	—
PORTC	—	—	—	—	—	—	—
PORTD	—	—	—	—	—	—	—
PORTE	—	—	—	—	—	—	—
PORTF	—	—	—	—	—	—	—
P1DR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P2DR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P3DR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
P6DR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PADR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PBDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PCDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PDDR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

RDR_2	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SCMR_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DADR0H	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DADR1H	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DACR01	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
DADR01T	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PCR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PMR	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
NDERH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
NDERL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PODRH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
PODRL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
NDRH	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
NDRL	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SMR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
BRR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
SCR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TDR_0	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SSR_0	Initialized	Initialized	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized

RDR_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized* <sup>1</sup>	Initialized
SCMR_1	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDR_A_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDRB_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDR_C_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDRD_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDRE_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDRF_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDRG_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADDRH_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADCSR_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADCR_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
ADMOSEL_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCSR	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
RSTCSR	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_1	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCSR_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCSR_1	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORA_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORA_1	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCORB_0	Initialized	—	—	—	—	—	Initialized* <sup>1</sup>	Initialized

TCR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIORH_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIORL_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRC_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRD_0	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIOR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_1	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

TGRB_2	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TMDR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIORH_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIORL_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TIER_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TSR_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TCNT_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRA_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRB_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRC_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized
TGRD_3	Initialized	—	—	—	—	Initialized* <sup>1</sup>	Initialized

- Notes:
1. Not initialized in deep software standby mode but initialized when deep software standby mode is released by the internal reset.
  2. These registers are initialized when all the RAMCUT2 to RAMCUT0 bits in DP are set to 1, and not initialized when these bits are set to 0.
  3. Supported only by the H8SX/1665M Group.
  4. LVDCCR is initialized by a pin reset or power-on reset not by a voltage-monitoring deep software standby reset, or watchdog timer reset.



Input voltage (except for port 5)	$V_{in}$	-0.3 to $V_{CC} + 0.3$
Input voltage (port 5)	$V_{in}$	-0.3 to $AV_{CC} + 0.3$
Reference power supply voltage	$V_{ref}$	-0.3 to $AV_{CC} + 0.3$
Analog power supply voltage	$AV_{CC}$	-0.3 to +4.6
Analog input voltage	$V_{AN}$	-0.3 to $AV_{CC} + 0.3$
Operating temperature	$T_{opr}$	Regular specifications: -20 to +75*  Wide-range specifications: -40 to +85*
Storage temperature	$T_{stg}$	-55 to +125

Caution: Permanent damage to the LSI may result if absolute maximum ratings are exceeded.

Note: \* The operating temperature range during programming/erasing of the flash memory is 0°C to +75°C for regular specifications and 0°C to +85°C for wide-range specifications.

Schmitt trigger input voltage	IRQ input pin, TPU input pin, TMR input pin, port 2, port 3 port J, port K	$V_T$	$V_{CC} \times 0.2$	—	—	V	
		$V_T^+$	—	—	$V_{CC} \times 0.7$	V	
		$V_T^- - V_T$	$V_{CC} \times 0.06$	—	—	V	
		$V_T^-$	$AV_{CC} \times 0.2$	—	—	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		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
	Other input pins Port 5		$AV_{CC} \times 0.7$	—	$AV_{CC} + 0.3$		
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	—	$V_{CC} \times 0.2$		
	Other input pins		-0.3	—	$V_{CC} \times 0.2$		
Output high voltage	All output pins	$V_{OH}$	$V_{CC} - 0.5$	—	—	V	$I_{OH} =$
			$V_{CC} - 1.0$	—	—		$I_{OH} =$
Output low voltage	All output pins	$V_{OL}$	—	—	0.4	V	$I_{OL} =$
	Port 3		—	—	1.0		$I_{OL} =$
Input leakage current	$\overline{RES}$	$ I_{in} $	—	—	10.0	$\mu A$	$V_{in} =$ $V_{CC} -$
	MD, $\overline{STBY}$ , EMLE, NMI		—	—	1.0		
	Port 5		—	—	1.0		$V_{in} =$ $AV_{CC}$

Supply current <sup>*2</sup>	Normal operation	$I_{CC}^{*3}$	—	50	85	mA	f =	
	Sleep mode		—	48	60			
	Subclock operation		—	5	10		32. cry: res: use:	
Standby mode	Software standby mode		—	0.15	1.1		$T_a \leq$ 50°	
	Deep software standby mode	RAM ,USB retained <sup>*6</sup>	—	20	60	$\mu$ A	$T_a \leq$ 50°	
		RAM, TM32K USB halted power supply halted operation		—	—	200		$T_a \leq$ 50°
				—	3	8		$T_a \leq$ 50°
	Hardware standby mode			—	—	26		$T_a \leq$ 50°
				—	9	16		$T_a \leq$ 50°
			—	—	41		$T_a \leq$ 50°	
	All-module-clock-stop mode <sup>*4</sup>		—	23	30	mA	$T_a \leq$ 50°	
Analog power supply current	During A/D and D/A conversion	$I_{CC}$	—	2.0	3.5	mA		
	Standby for A/D and D/A conversion		—	0.5	1.5	$\mu$ A		
Reference power supply current	During A/D and D/A conversion	$I_{CC}$	—	0.8	1.5	mA		
	Standby for A/D and D/A conversion		—	0.5	1.5	$\mu$ A		
RAM standby voltage		$V_{RAM}$	2.5	—	—	V		
Vcc start voltage <sup>*5</sup>		$V_{CC}^{START}$	—	—	0.8	V		

5. This can be applied at power-on.
6. The value when TM 32K halted.

**Table 30.3 Permissible Output Currents**

Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  
 $V_{ref} = 3.0\text{ V to }AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0\text{ V}^*$ ,  
 $T_a = -20^\circ\text{C to }+75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to }+85^\circ\text{C}$  (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.
Permissible output low current (per pin)	Output pins except port 3	$I_{OL}$	—	—	2.0
Permissible output low current (per pin)	Port 3	$I_{OL}$	—	—	10
Permissible output low current (total)	Total of all output pins	$\Sigma I_{OL}$	—	—	80
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2.0
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	40

Caution: To protect the LSI's reliability, do not exceed the output current values in table

Note: \* When the A/D and D/A converters are not used, the  $AV_{CC}$ ,  $V_{ref}$ , and  $AV_{SS}$  pins should be open. Connect the  $AV_{CC}$  and  $V_{ref}$  pins to  $V_{CC}$ , and the  $AV_{SS}$  pin to  $V_{SS}$ .

Schmitt trigger input voltage	IRQ input pin,	$V_I$	$V_{CC} \times 0.2$	—	—	V	
	TPU input pin,	$VT^+$	—	—	$V_{CC} \times 0.7$	V	
	TMR input pin, port 2, port 3 port J, port K	$VT^+ - VT^-$	$V_{CC} \times 0.06$	—	—	V	
IRQ0-B to IRQ7-B input pin	$\overline{IRQ0-B}$ to	$VT^-$	$AV_{CC} \times 0.2$	—	—	V	
	IRQ7-B input pin	$VT^+$	—	—	$AV_{CC} \times 0.7$	V	
		$VT^+ - VT^-$	$AV_{CC} \times 0.06$	—	—	V	
Input high voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$ , STBY, EMLE, NMI	$V_{IH}$	$V_{CC} \times 0.9$	—	$V_{CC} + 0.3$	V	
	EXTAL		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
	Other input pins Port 5		$AV_{CC} \times 0.7$	—	$AV_{CC} + 0.3$		
Input low voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$ , STBY, EMLE	$V_{IL}$	-0.3	—	$V_{CC} \times 0.1$	V	
	EXTAL, NMI		-0.3	—	$V_{CC} \times 0.2$		
	Other input pins		-0.3	—	$V_{CC} \times 0.2$		
Output high voltage	All output pins	$V_{OH}$	$V_{CC} - 0.5$	—	—	V	$I_{OH} =$
			$V_{CC} - 1.0$	—	—		$I_{OH} =$
Output low voltage	All output pins	$V_{OL}$	—	—	0.4	V	$I_{OL} =$
	Port 3		—	—	1.0		$I_{OL} =$
Input leakage current	$\overline{RES}$	$ I_{in} $	—	—	10.0	$\mu A$	$V_{in} =$
	MD, $\overline{STBY}$ , EMLE, NMI		—	—	1.0		$V_{CC}$
	Port 5		—	—	1.0		$V_{in} =$ $AV_{CC}$

Supply current <sup>*2</sup>	Normal operation	$I_{cc}^{*3}$	—	50	85	mA	f = 5	
	Sleep mode		—	48	60			
	Subclock operation		—	5	10			
Standby mode	Software standby mode		—	0.15	1.1	mA	$T_a \leq$	
			—	—	3.5		50°C	
	Deep software standby mode		RAM, USB retained* <sup>6</sup>	—	24	67	$\mu$ A	$T_a \leq$
				—	—	200		50°C
			RAM, TM32K USB halted power supply halted operation	—	23	35		$T_a \leq$
				—	—	60		
				—	29	43		50°C
				—	—	75		
	Hardware standby mode		—	2	7		$T_a \leq$	
			—	—	25		50°C	
All-module-clock-stop mode* <sup>4</sup>			—	23	30	mA		
Analog power supply current	During A/D and D/A conversion	$I_{cc}$	—	2.0	3.5	mA		
	Standby for A/D and D/A conversion		—	0.5	1.5		$\mu$ A	
Reference power supply current	During A/D and D/A conversion	$I_{cc}$	—	0.8	1.5	mA		
	Standby for A/D and D/A conversion		—	0.5	1.5		$\mu$ A	
RAM standby voltage		$V_{RAM}$	2.5	—	—	V		
Vcc start voltage* <sup>5</sup>		$V_{cc}START$	—	—	0.8	V		

5. This can be applied at power-on.
6. The value when TM 32K halted.

**Table 30.5 Permissible Output Currents**

Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95 \text{ V to } 3.6 \text{ V}$ ,  $AV_{CC} = 3.0 \text{ V to } 3.6 \text{ V}$ ,  
 $V_{ref} = 3.0 \text{ V to } AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0 \text{ V}^*$ ,  
 $T_a = -20^\circ\text{C to } +75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to } +85^\circ\text{C}$  (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.
Permissible output low current (per pin)	Output pins except port 3	$I_{OL}$	—	—	2.0
Permissible output low current (per pin)	Port 3	$I_{OL}$	—	—	10
Permissible output low current (total)	Total of all output pins	$\Sigma I_{OL}$	—	—	80
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2.0
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	40

Caution: To protect the LSI's reliability, do not exceed the output current values in table.

Note: \* When the A/D and D/A converters are not used, the  $AV_{CC}$ ,  $V_{ref}$ , and  $AV_{SS}$  pins should be open. Connect the  $AV_{CC}$  and  $V_{ref}$  pins to  $V_{CC}$ , and the  $AV_{SS}$  pin to  $V_{SS}$ .

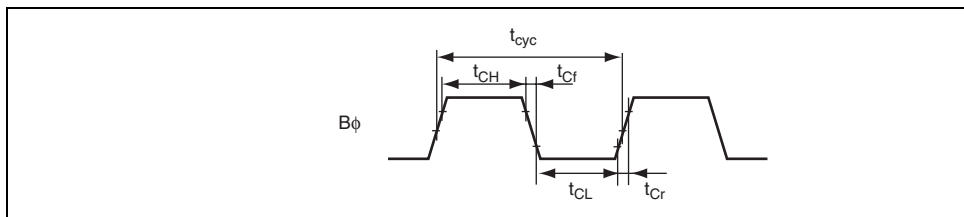
**Figure 30.1 Output Load Circuit****30.4.1 Clock Timing****Table 30.6 Clock Timing**

Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 3.0$  V to 3.6 V\*,  $AV_{CC} = 3.0$  V to 3.6 V,  
 $V_{ref} = 3.0$  V to  $AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0$  V,  $I\phi = 8$  MHz to 50  
 MHz,  $B\phi = 8$  MHz to 50 MHz,  $P\phi = 8$  MHz to 35 MHz,  
 $T_a = -20^\circ\text{C}$  to  $+75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C}$  to  $+85^\circ\text{C}$  (wide-range specifications)

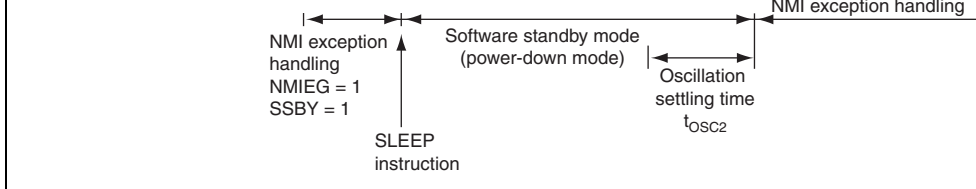
Item	Symbol	Min.	Max.	Unit.	Test Co
Clock cycle time	$t_{cyc}$	20	125	ns	Figure 3
Clock high pulse width	$t_{CH}$	5	—	ns	
Clock low pulse width	$t_{CL}$	5	—	ns	
Clock rising time	$t_{Cr}$	—	5	ns	
Clock falling time	$t_{Cf}$	—	5	ns	
Oscillation settling time after reset (crystal)	$t_{OSC1}$	10	—	ms	Figure 3
Oscillation settling time after leaving software standby mode (crystal)	$t_{OSC2}$	10	—	ms	Figure 3



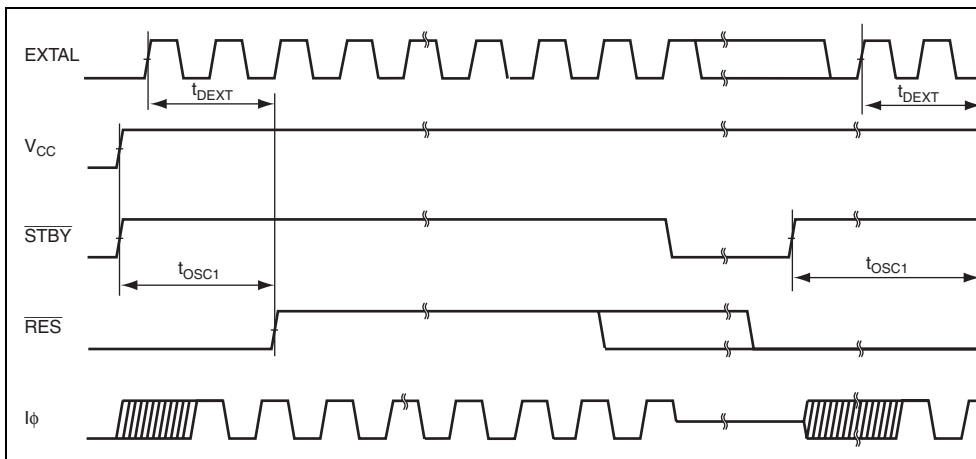
Note: \*  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95\text{ V to }3.60\text{ V}$  in the H8SX/1665M Group.



**Figure 30.2 External Bus Clock Timing**



**Figure 30.3 Oscillation Settling Timing after Software Standby Mode**

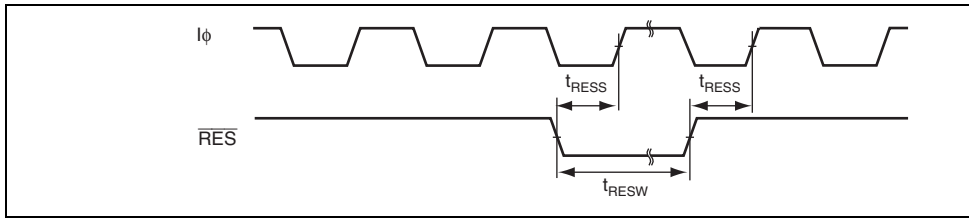


**Figure 30.4 Oscillation Settling Timing**



$\overline{\text{RES}}$ pulse width	$t_{\text{RESW}}$	20	—	$t_{\text{cyc}}$	Figure 30
NMI setup time	$t_{\text{NMIS}}$	150	—	ns	
NMI hold time	$t_{\text{NMIH}}$	10	—	ns	
NMI pulse width (after leaving software standby mode)	$t_{\text{NMIW}}$	200	—	ns	
$\overline{\text{IRQ}}$ setup time	$t_{\text{IRQS}}$	150	—	ns	
$\overline{\text{IRQ}}$ hold time	$t_{\text{IRQH}}$	10	—	ns	
$\overline{\text{IRQ}}$ pulse width (after leaving software standby mode)	$t_{\text{IRQW}}$	200	—	ns	

Note: \*  $V_{\text{CC}} = \text{PLL}V_{\text{CC}} = \text{Dr}V_{\text{CC}} = 2.95 \text{ V to } 3.60 \text{ V}$  in the H8SX/1665M Group.



**Figure 30.6 Reset Input Timing**

Note: \* SSIER must be set to cancel software standby mode.

**Figure 30.7 Interrupt Input Timing**

**30.4.3 Bus Timing**

**Table 30.8 Bus Timing (1)**

Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 3.0\text{ V to }3.6\text{ V}^*$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{ref} = 3.0\text{ V}$ ,  
 $AV_{CC}, V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0\text{ V}$ ,  $B\phi = 8\text{ MHz to }50\text{ MHz}$ ,  
 $T_a = -20^\circ\text{C to }+75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to }+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Cond
Address delay time	$t_{AD}$	—	15	ns	Figure 30.36
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	

Read data hold time 2	$t_{RDH2}$	—	—	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
Read data access time (from address) 1	$t_{AA1}$	—	$1.0 \times t_{cyc} - 20$	ns
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

Note: \*  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95 \text{ V to } 3.6 \text{ V}$  in the H8SX/1665M Group.

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	Figure 30.14
Byte control pulse width 1	$t_{UBW1}$	—	$1.0 \times t_{cyc} - 15$	ns	Figure 30.14
Byte control pulse width 2	$t_{UBW2}$	—	$2.0 \times t_{cyc} - 15$	ns	Figure 30.14
Multiplexed address delay time 1	$t_{MAD1}$	—	15	ns	Figure 30.18
Multiplexed address hold time	$t_{MAH}$	$1.0 \times t_{cyc} - 15$	—	ns	Figure 30.18
Multiplexed address setup time 1	$t_{MAS1}$	$0.5 \times t_{cyc} - 15$	—	ns	Figure 30.18
Multiplexed address setup time 2	$t_{MAS2}$	$1.5 \times t_{cyc} - 15$	—	ns	Figure 30.18
Address hold delay time	$t_{AHD}$	—	15	ns	Figure 30.18
Address hold pulse width 1	$t_{AHW1}$	$1.0 \times t_{cyc} - 15$	—	ns	Figure 30.18
Address hold pulse width 2	$t_{AHW2}$	$2.0 \times t_{cyc} - 15$	—	ns	Figure 30.18
$\overline{WAIT}$ setup time	$t_{WTS}$	15	—	ns	Figure 30.18
$\overline{WAIT}$ hold time	$t_{WTH}$	5.0	—	ns	Figure 30.18
$\overline{BREQ}$ setup time	$t_{BREQS}$	20	—	ns	Figure 30.18
$\overline{BACK}$ delay time	$t_{BACD}$	—	15	ns	Figure 30.18
Bus floating time	$t_{BZD}$	—	30	ns	Figure 30.18
$\overline{BREQO}$ delay time	$t_{BRQOD}$	—	15	ns	Figure 30.18
$\overline{BS}$ delay time	$t_{BSD}$	1.0	15	ns	Figure 30.9, 30.14
$RD/\overline{WR}$ delay time	$t_{RWD}$	—	15	ns	Figure 30.9, 30.14

Note: \*  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95\text{ V to }3.6\text{ V}$  in the H8SX/1665M Group.

Read data access time 3	$t_{AC3}^{AC1}$	—	$2.0 \times t_{cyc} - 20$	ns
Read data access time 7	$t_{AC7}$	—	$4.0 \times t_{cyc} - 20$	ns
Read data access time 8	$t_{AC8}$	—	$3.0 \times t_{cyc} - 20$	ns
Write data hold time 2	$t_{WDH2}$	$1.0 \times t_{cyc} - 8$	—	ns
Read command setup time 1	$t_{RCS1}$	$1.5 \times t_{cyc} - 10$	—	ns
Read command setup time 2	$t_{RCS2}$	$2.0 \times t_{cyc} - 10$	—	ns
Read command hold time	$t_{RCH}$	$0.5 \times t_{cyc} - 10$	—	ns
Write command setup time 1	$t_{WCS1}$	$0.5 \times t_{cyc} - 10$	—	ns
Write command setup time 2	$t_{WCS2}$	$1.0 \times t_{cyc} - 10$	—	ns
Write command hold time 1	$t_{WCH1}$	$0.5 \times t_{cyc} - 10$	—	ns
Write command hold time 2	$t_{WCH2}$	$1.0 \times t_{cyc} - 10$	—	ns
$\overline{CAS}$ delay time 1	$t_{CASD1}$	—	15	ns
$\overline{CAS}$ delay time 2	$t_{CASD2}$	—	15	ns
$\overline{CAS}$ setup time 1	$t_{CSR1}$	$0.5 \times t_{cyc} - 10$	—	ns
$\overline{CAS}$ setup time 2	$t_{CSR2}$	$1.5 \times t_{cyc} - 10$	—	ns
$\overline{CAS}$ pulse width 1	$t_{CASW1}$	$1.0 \times t_{cyc} - 15$	—	ns
$\overline{CAS}$ pulse width 2	$t_{CASW2}$	$1.5 \times t_{cyc} - 15$	—	ns
$\overline{CAS}$ precharge time 1	$t_{CPW1}$	$1.0 \times t_{cyc} - 15$	—	ns
$\overline{CAS}$ precharge time 2	$t_{CPW2}$	$1.5 \times t_{cyc} - 15$	—	ns
$\overline{OE}$ delay time 1	$t_{OED1}$	—	15	ns
$\overline{OE}$ delay time 2	$t_{OED2}$	—	15	ns
Precharge time 1	$t_{PCH1}$	$1.0 \times t_{cyc} - 20$	—	ns
Precharge time 2	$t_{PCH2}$	$1.5 \times t_{cyc} - 20$	—	ns

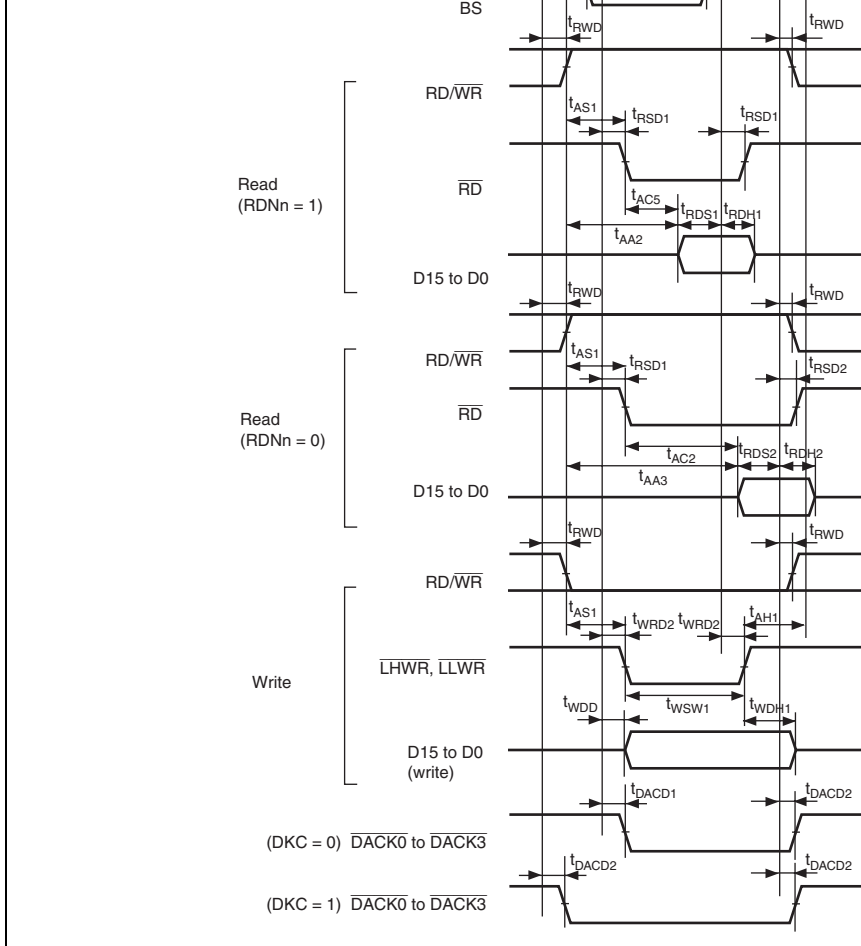


**Table 30.8 Bus Timing (4)**

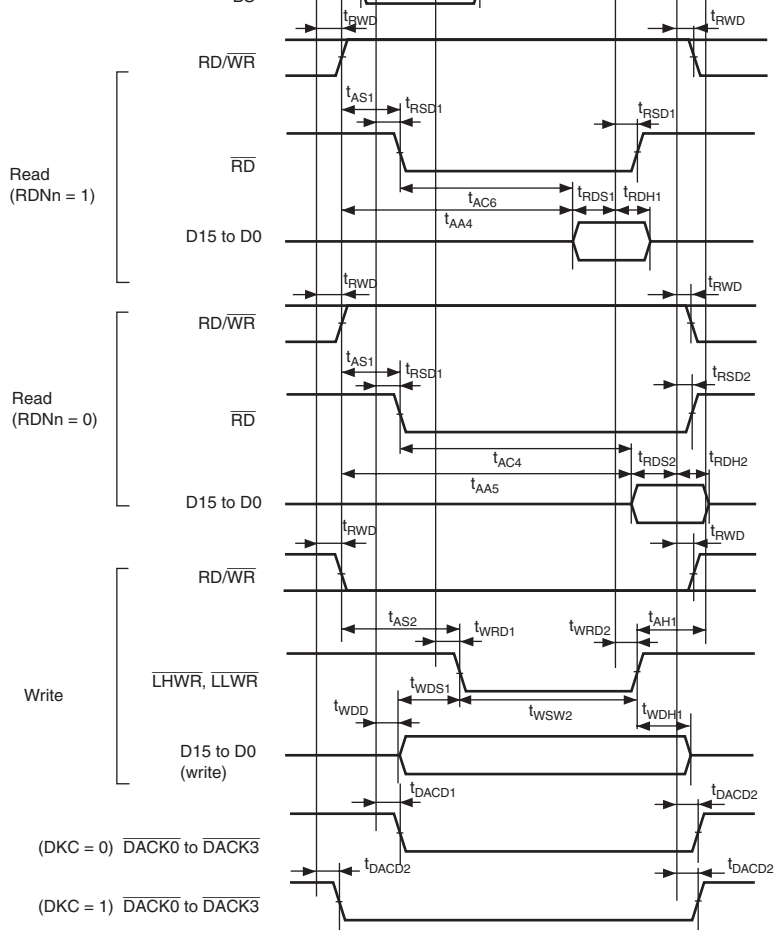
Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 3.0\text{ V to }3.6\text{ V}^*$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  
 $V_{ref} = 3.0\text{ V to }AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0\text{ V}$ ,  $B\phi = 8\text{ MHz to }16\text{ MHz}$ ,  
 $T_a = -20^\circ\text{C to }+75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to }+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Cond.
Read data setup time 3	$t_{RDS3}$	12	—	ns	Figure 30-10 to 30-12
Read data hold time 3	$t_{RDH3}$	0	—	ns	
Read data setup time 4	$t_{RDS4}$	12	—	ns	
Read data hold time 4	$t_{RDH4}$	0	—	ns	
Write data delay time 2	$t_{WDD2}$	—	15	ns	
Write data hold time 4	$t_{WDH4}$	1	—	ns	

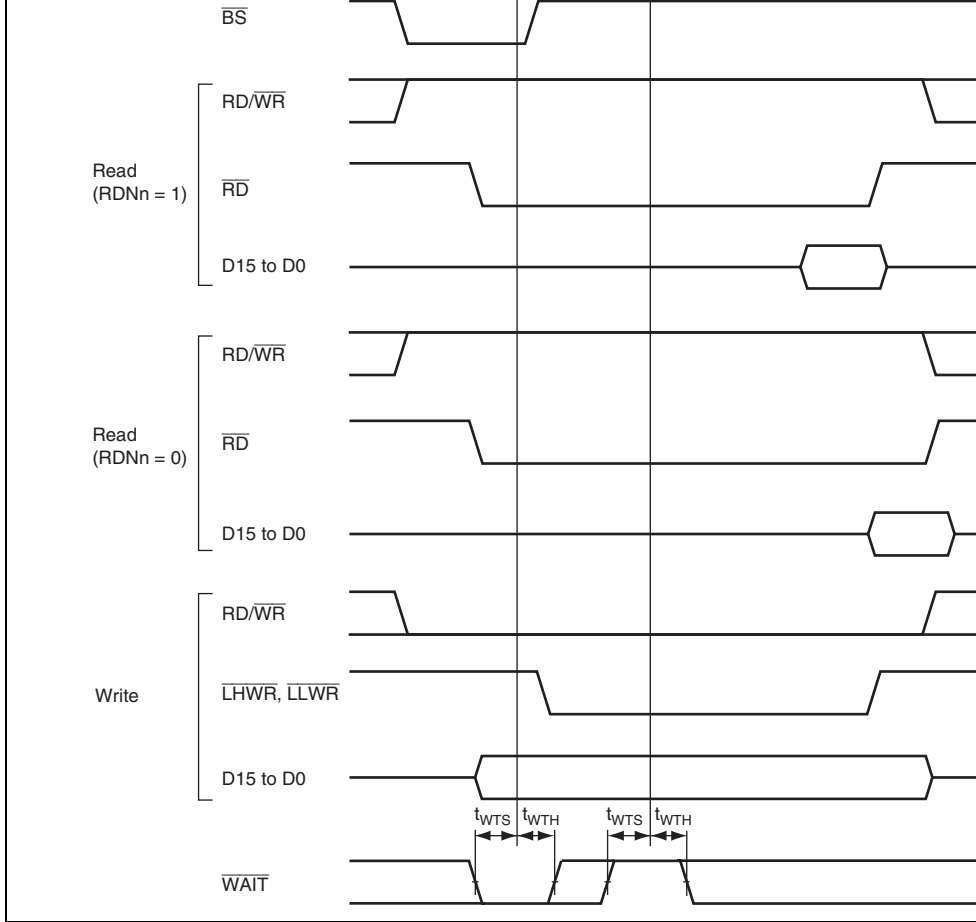
Note: \*  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95\text{ V to }3.6\text{ V}$  in the H8SX/1665M Group.



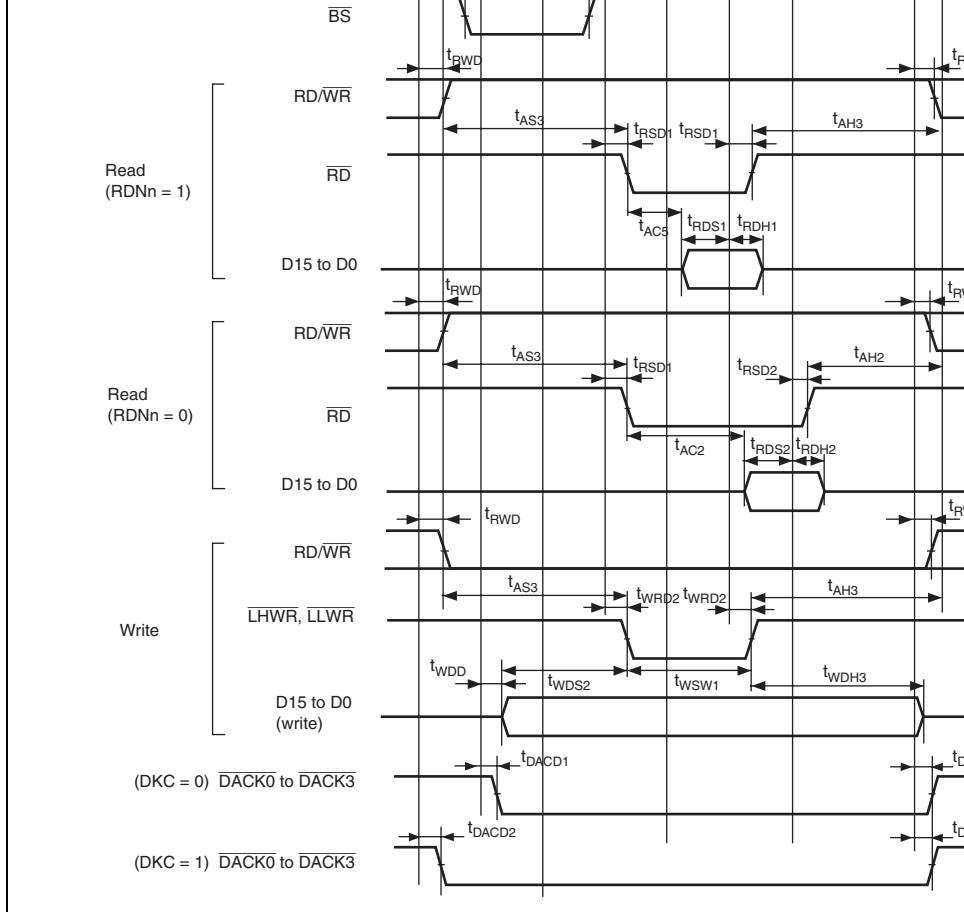
**Figure 30.8 Basic Bus Timing: Two-State Access**



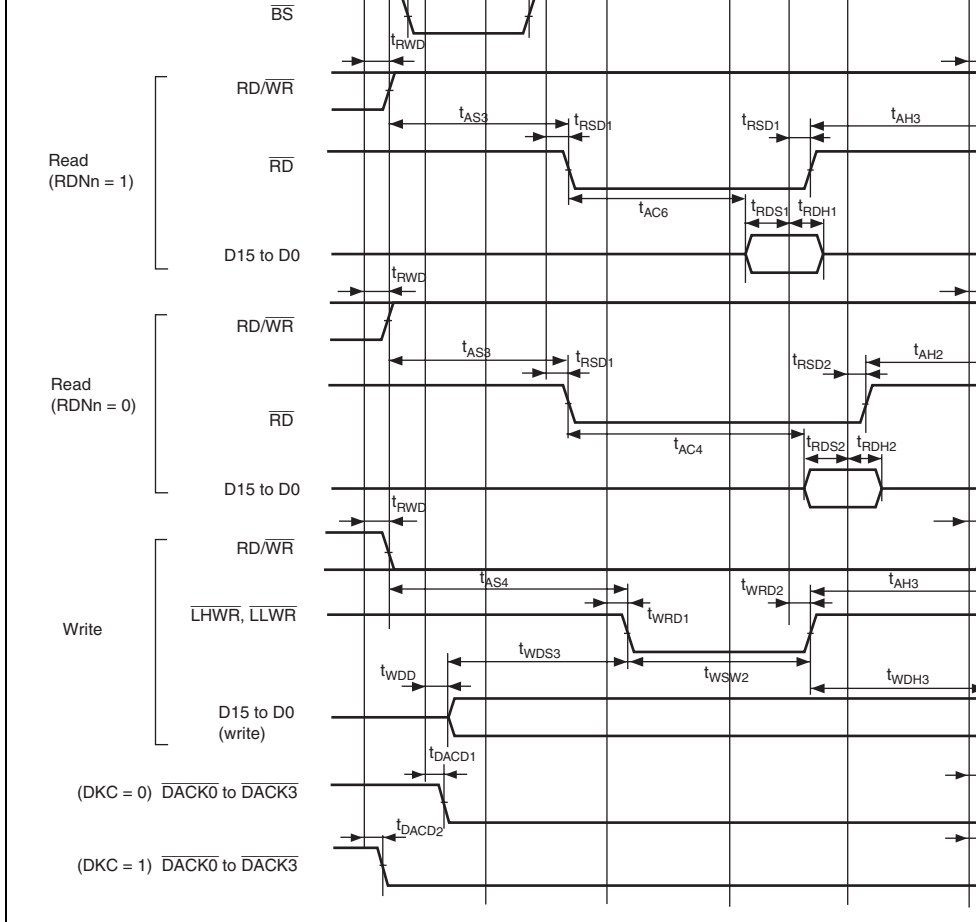
**Figure 30.9 Basic Bus Timing: Three-State Access**



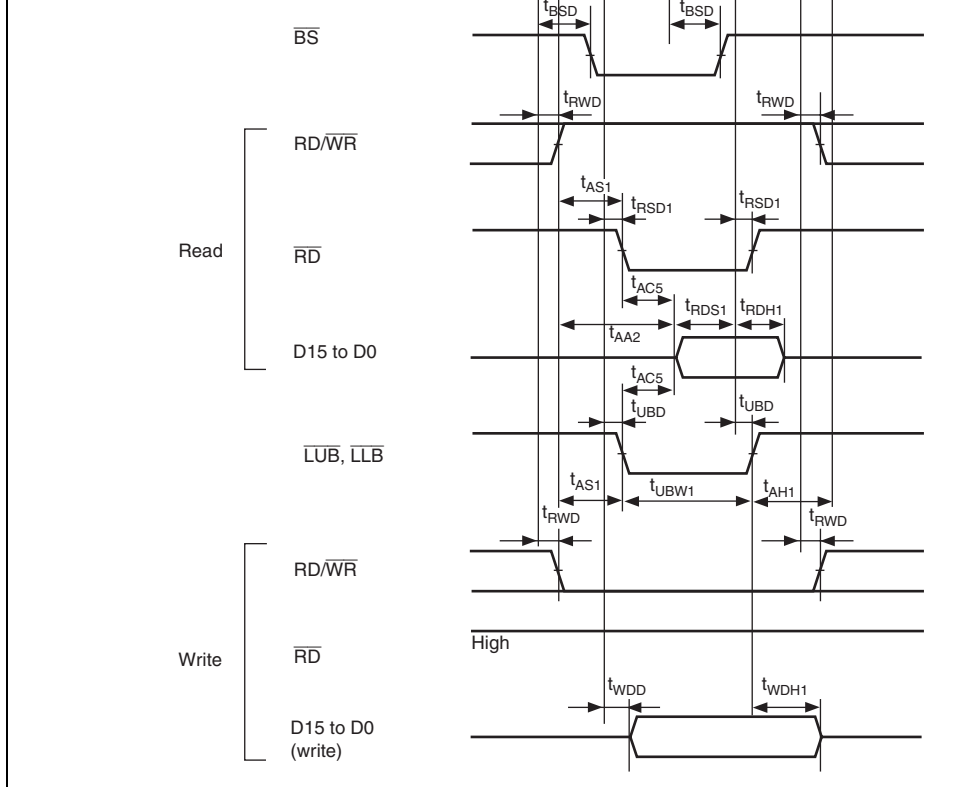
**Figure 30.10 Basic Bus Timing: Three-State Access, One Wait**



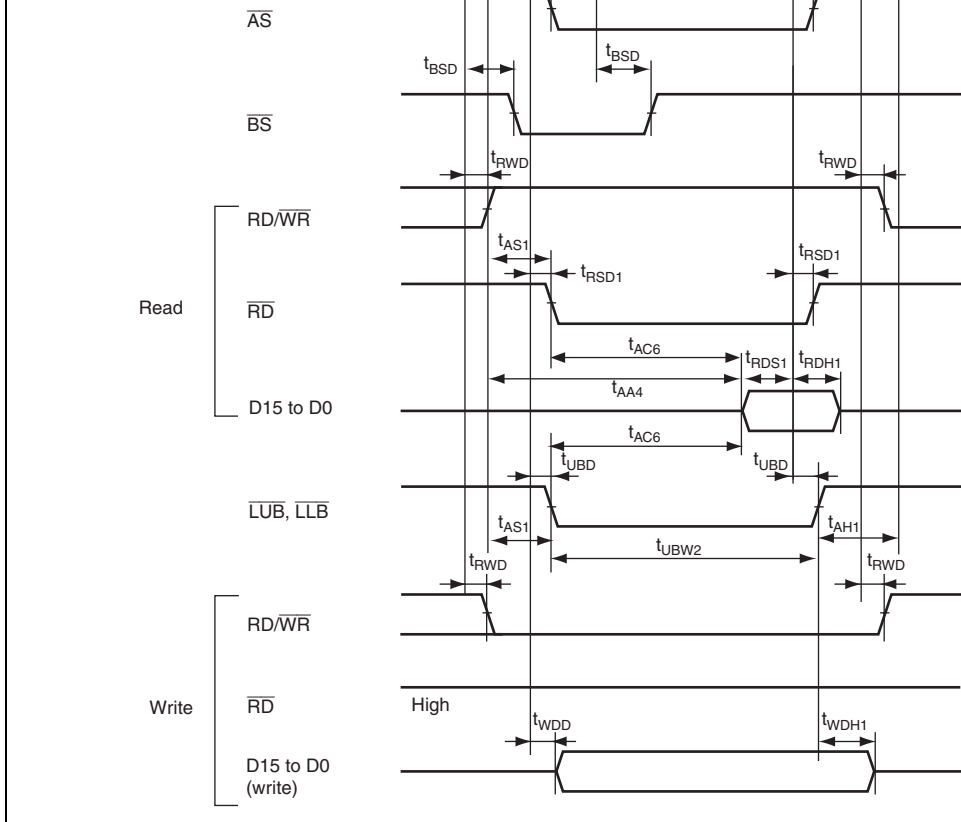
**Figure 30.11 Basic Bus Timing: Two-State Access ( $\overline{CS}$  Assertion Period Extended)**



**Figure 30.12 Basic Bus Timing: Three-State Access ( $\overline{CS}$  Assertion Period Extension)**

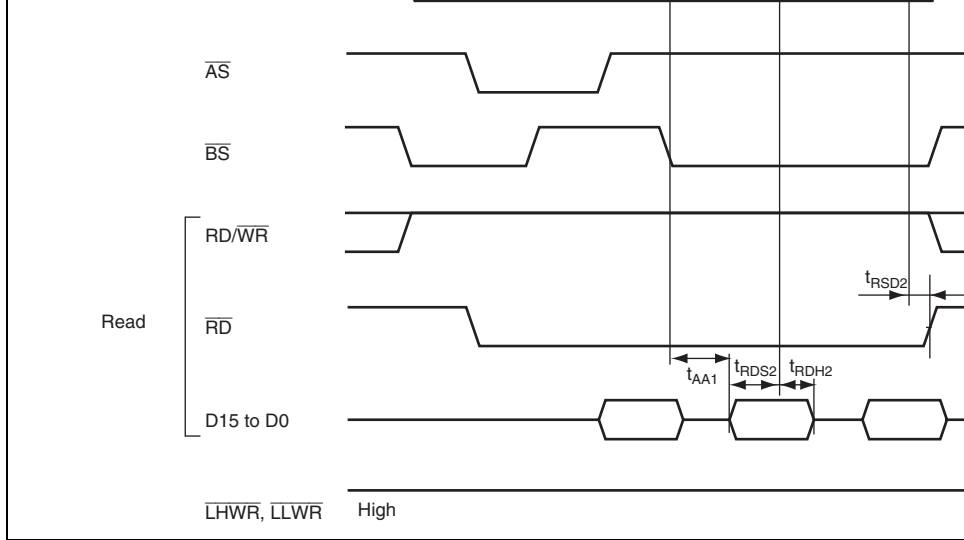


**Figure 30.13 Byte Control SRAM: Two-State Read/Write Access**

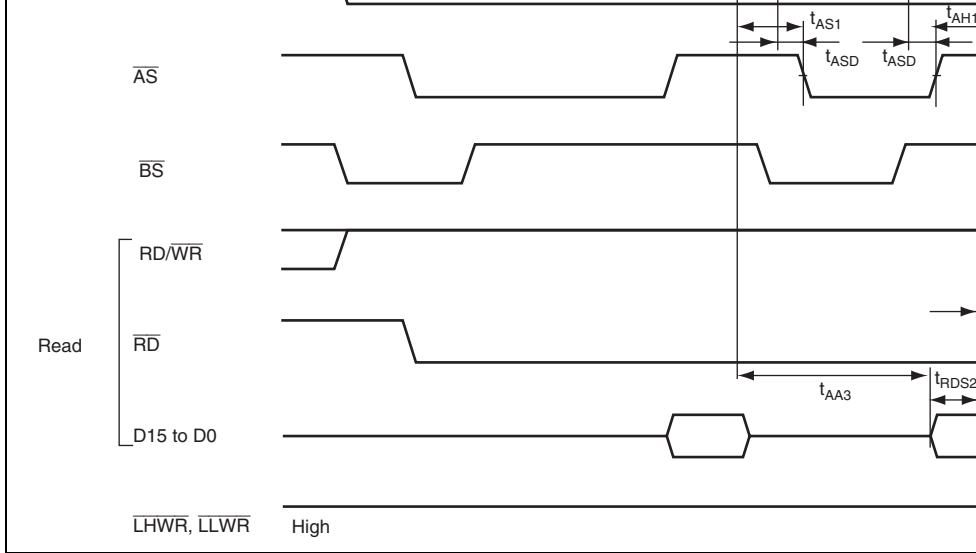


**Figure 30.14 Byte Control SRAM: Three-State Read/Write Access**

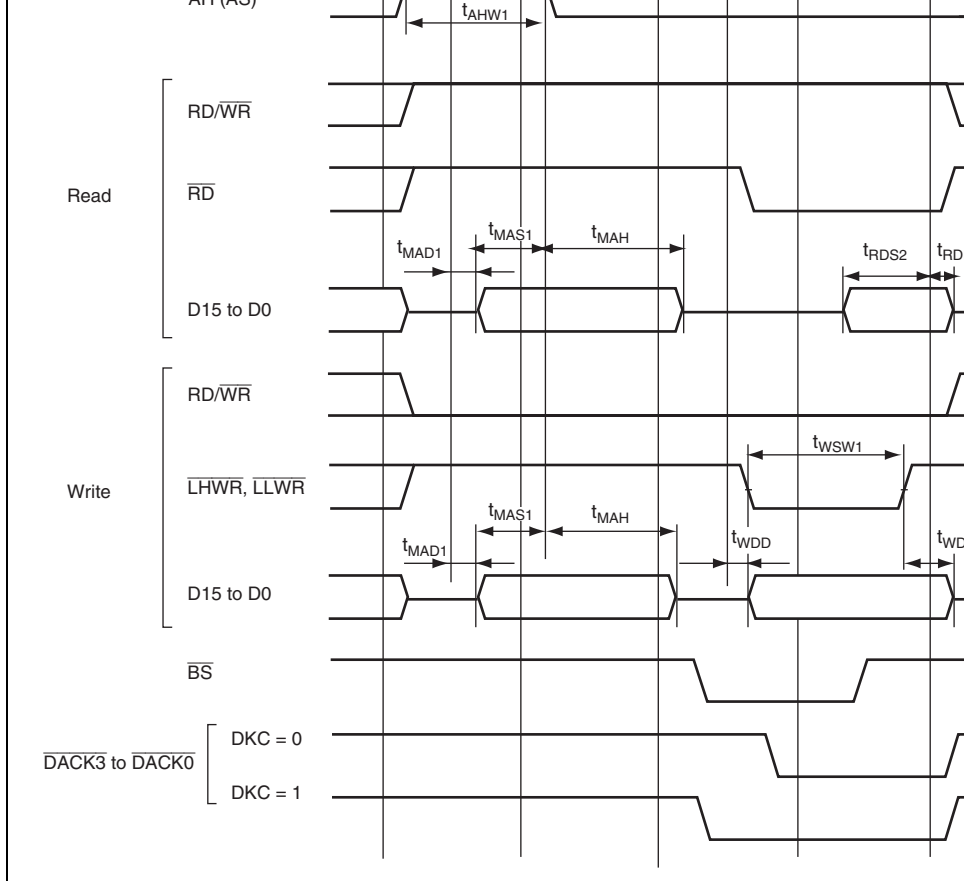




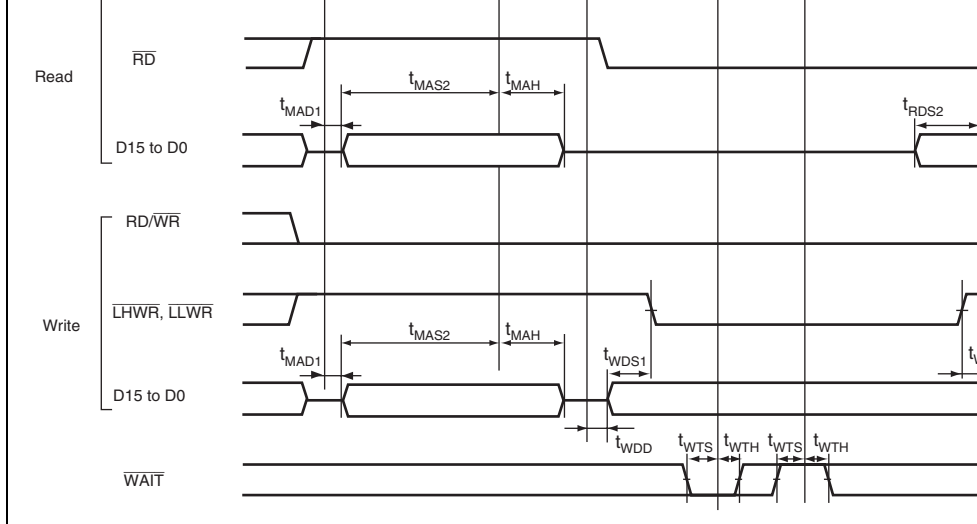
**Figure 30.15 Burst ROM Access Timing: One-State Burst Access**



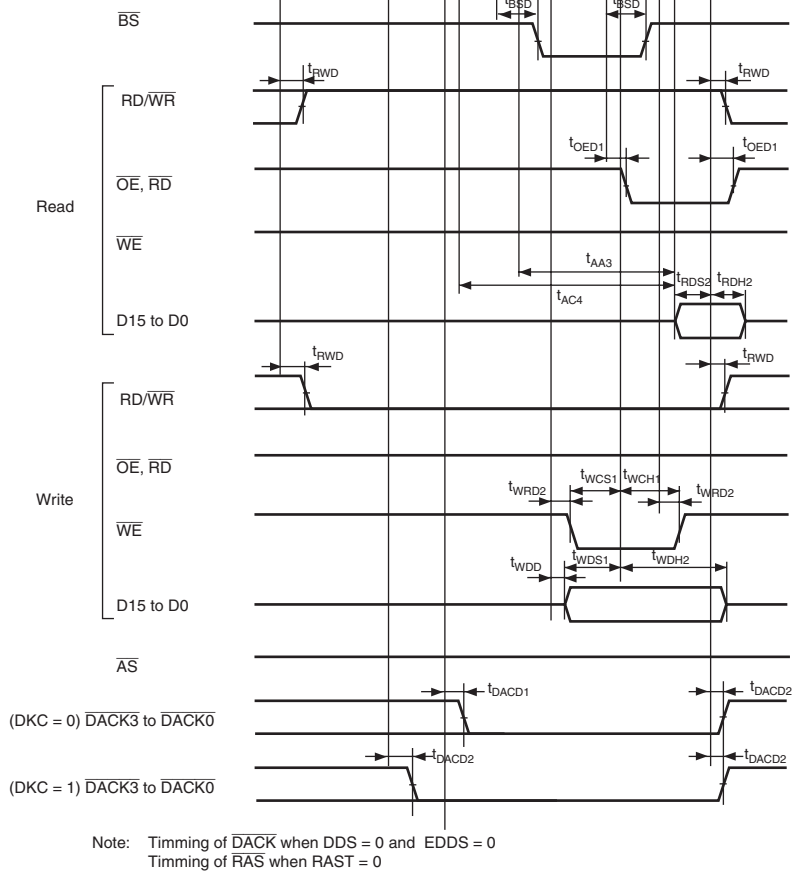
**Figure 30.16 Burst ROM Access Timing: Two-State Burst Access**



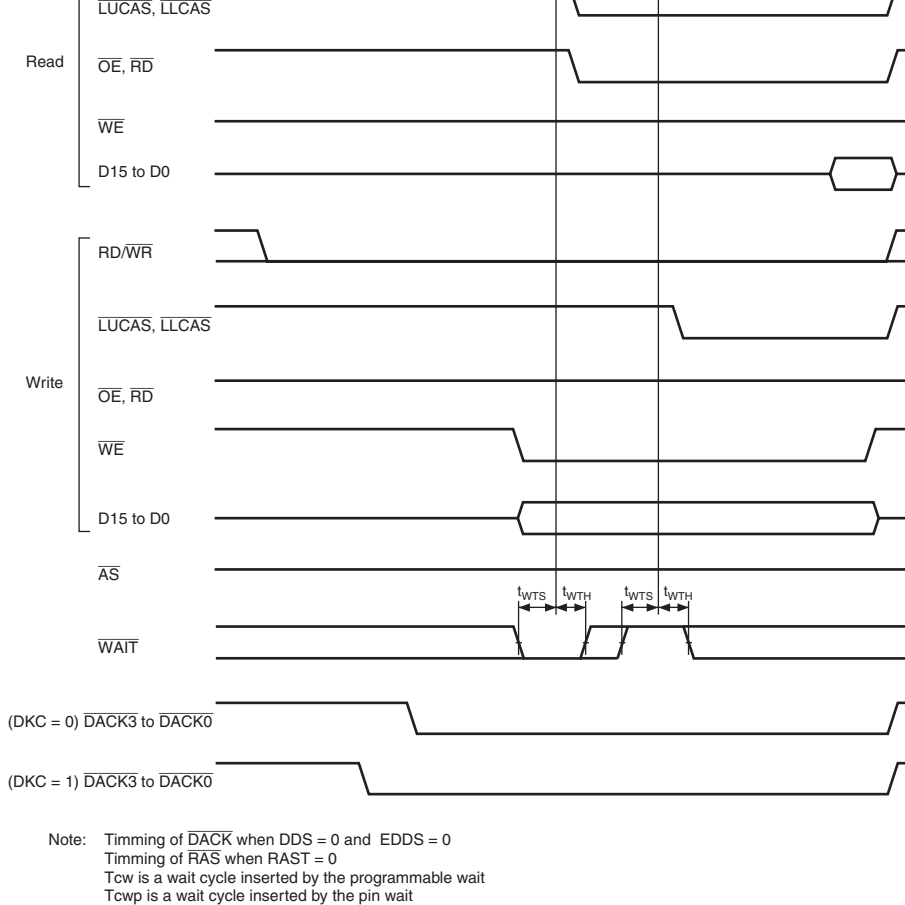
**Figure 30.17 Address/Data Multiplexed Access Timing (No Wait)  
(Basic, Four-State Access)**



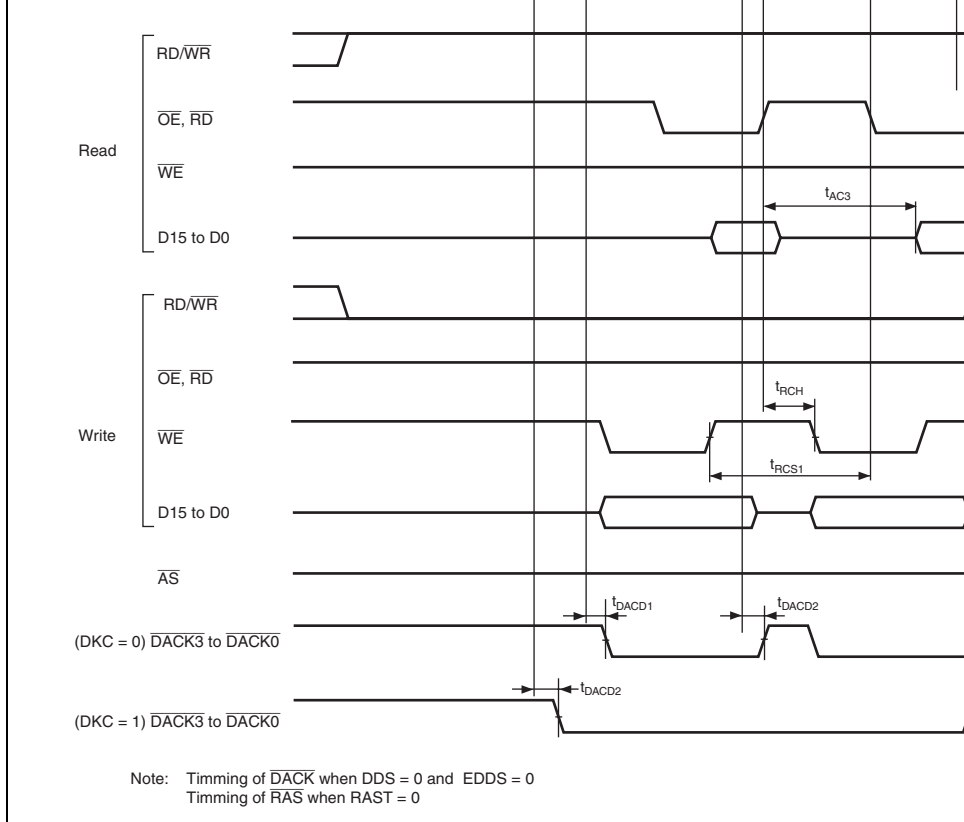
**Figure 30.18 Address/Data Multiplexed Access Timing (Wait Control)  
 (Address Cycle Program Wait × 1 + Data Cycle Program Wait × 1 +  
 Data Cycle Pin Wait × 1)**



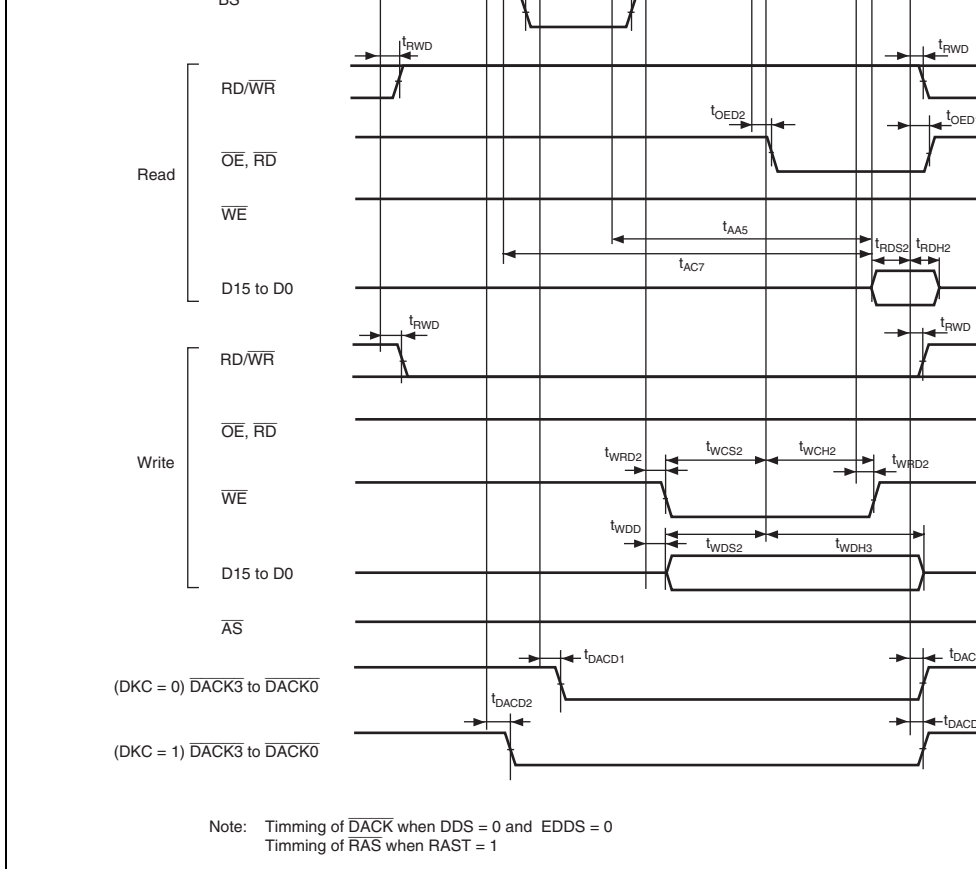
**Figure 30.19 DRAM Access Timing: Two-State Access**



**Figure 30.20 DRAM Access Timing: Two-State Access, One Wait**

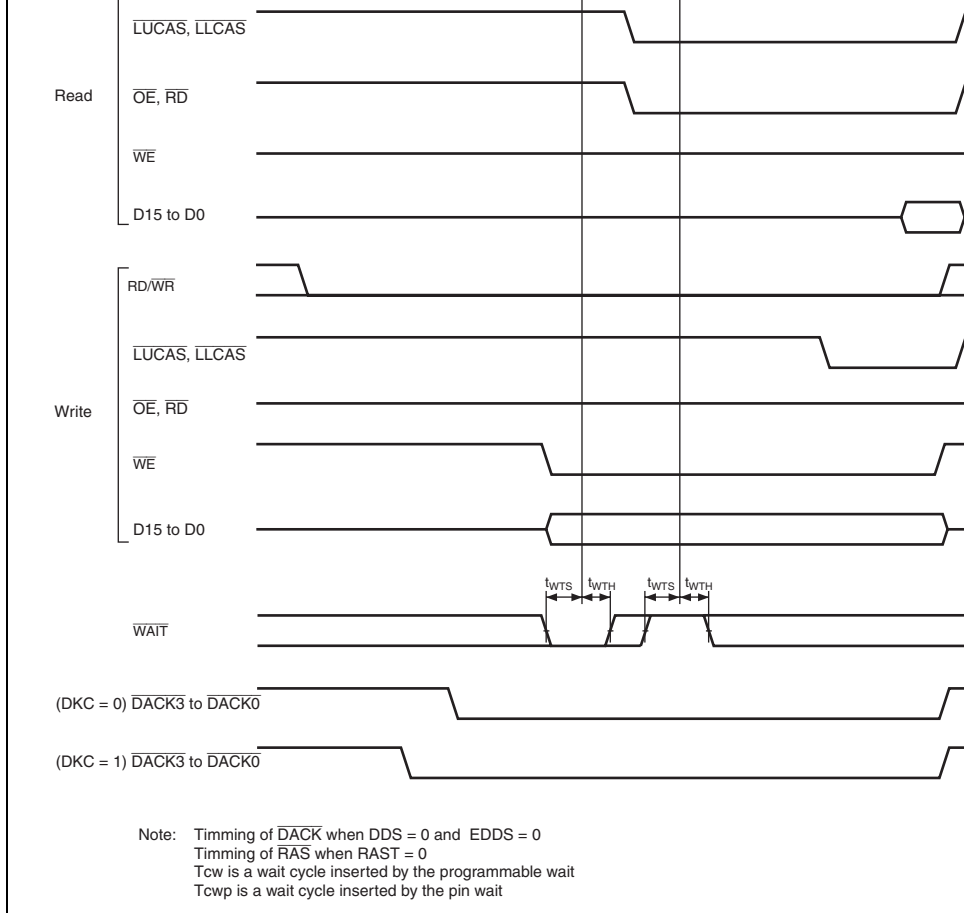


**Figure 30.21 DRAM Access Timing: Two-State Burst Access**

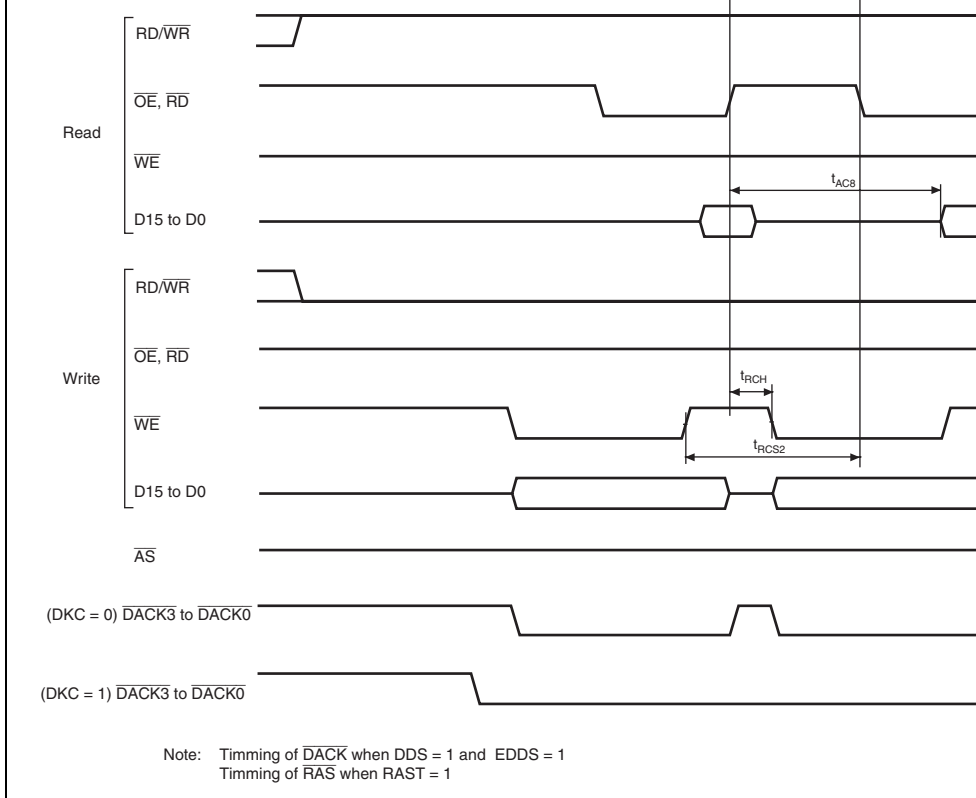


**Figure 30.22 DRAM Access Timing: Three-State Access (RAST = 1)**



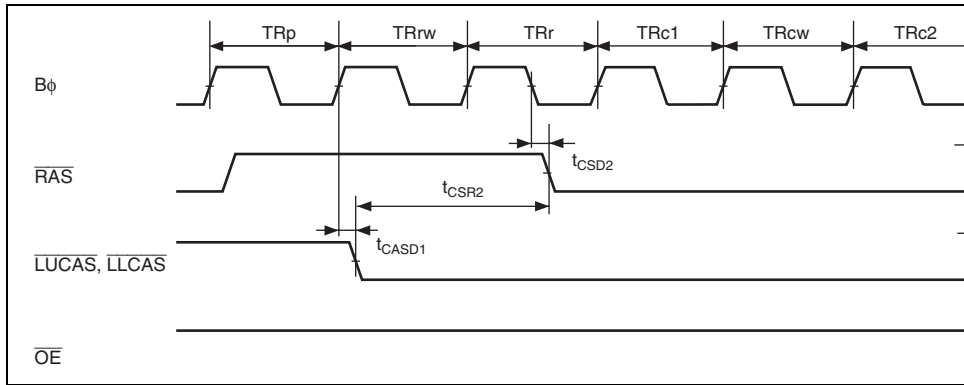


**Figure 30.23 DRAM Access Timing: Three-State Access, One Wait**

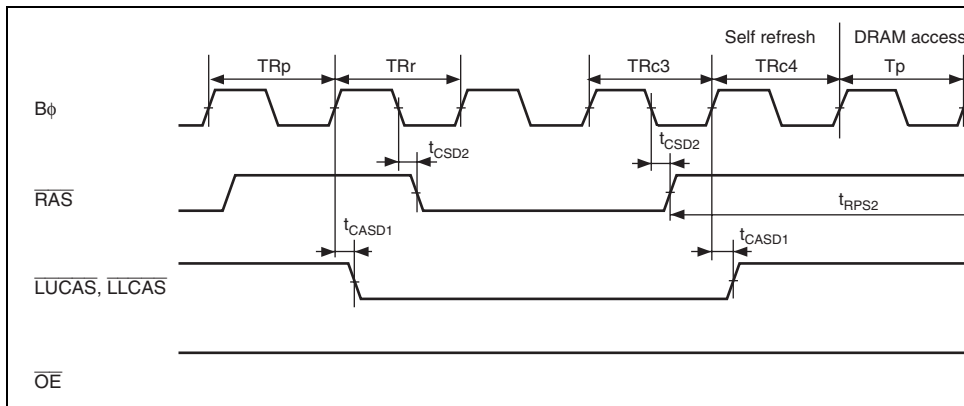


**Figure 30.24 DRAM Access Timing: Three-State Burst Access**

**Figure 30.25 CAS Before RAS Refresh Timing**

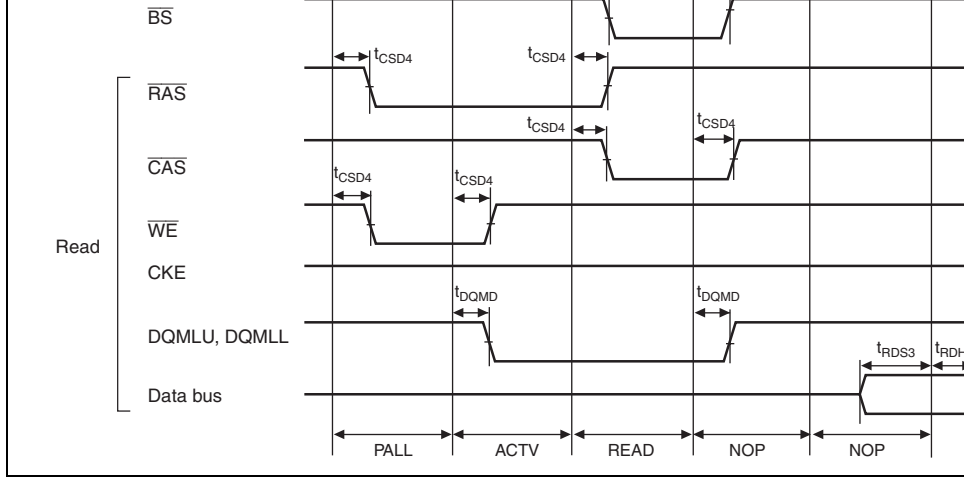


**Figure 30.26 CAS Before RAS Refresh Timing (Wait Cycle Inserted)**

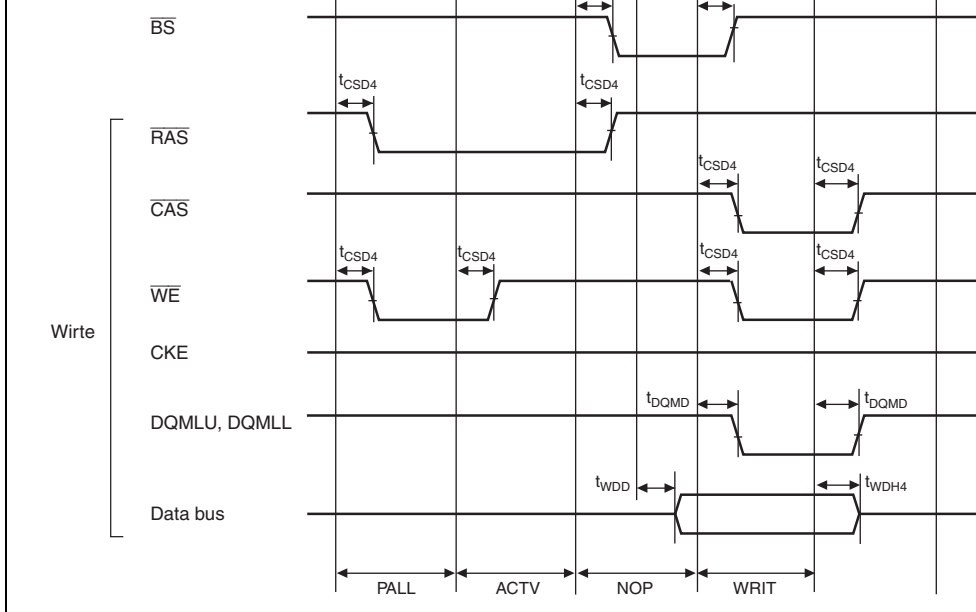


**Figure 30.27 Self-Refresh Timing (After Leaving Software Standby: RAST)**

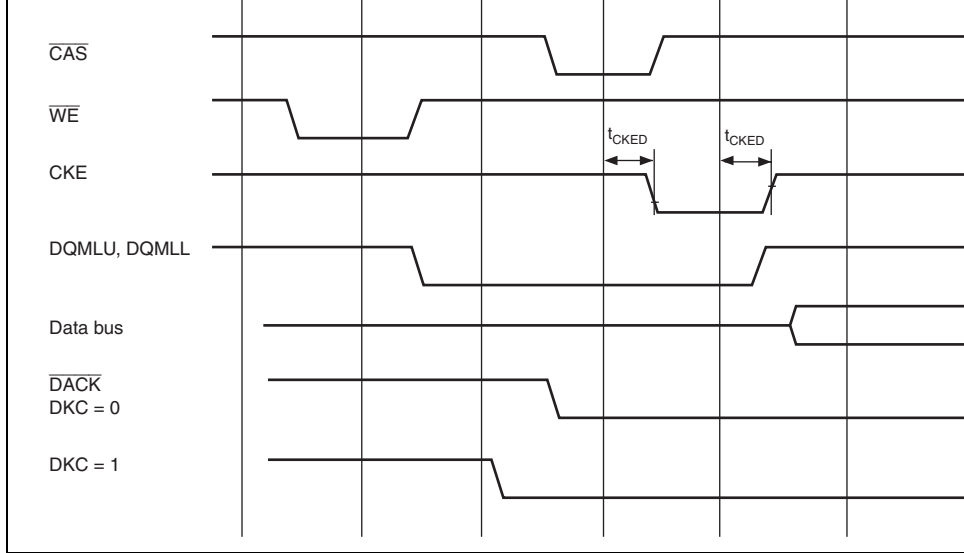




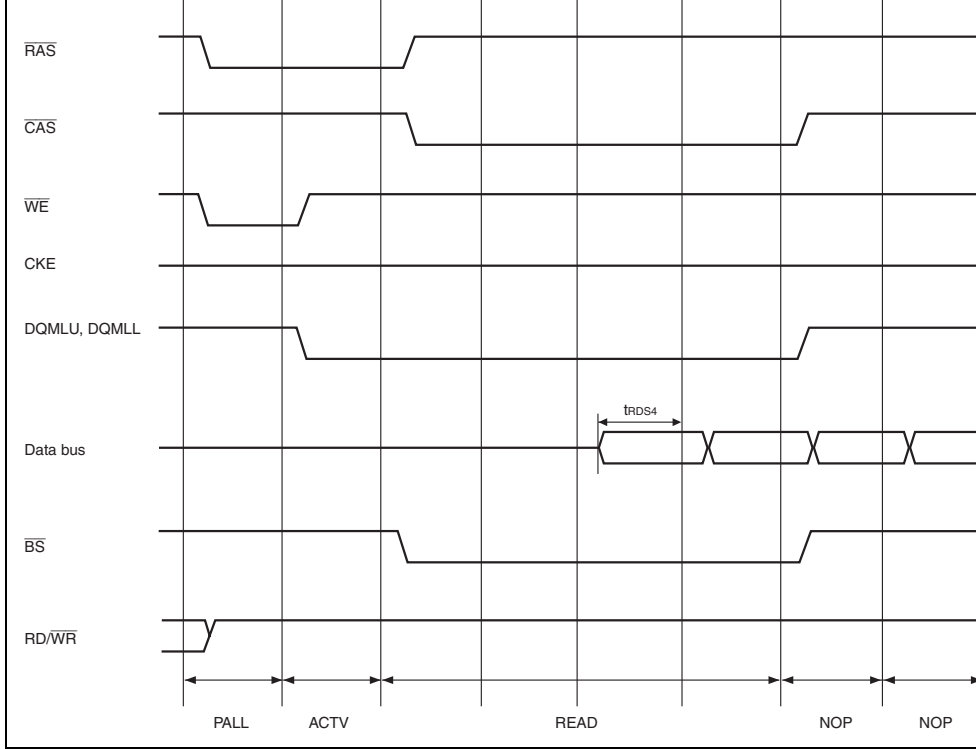
**Figure 30.29 Synchronous DRAM Basic Read Access Timing (CAS Latency)**



**Figure 30.30 Synchronous DRAM Basic Write Access Timing (CAS Latency)**

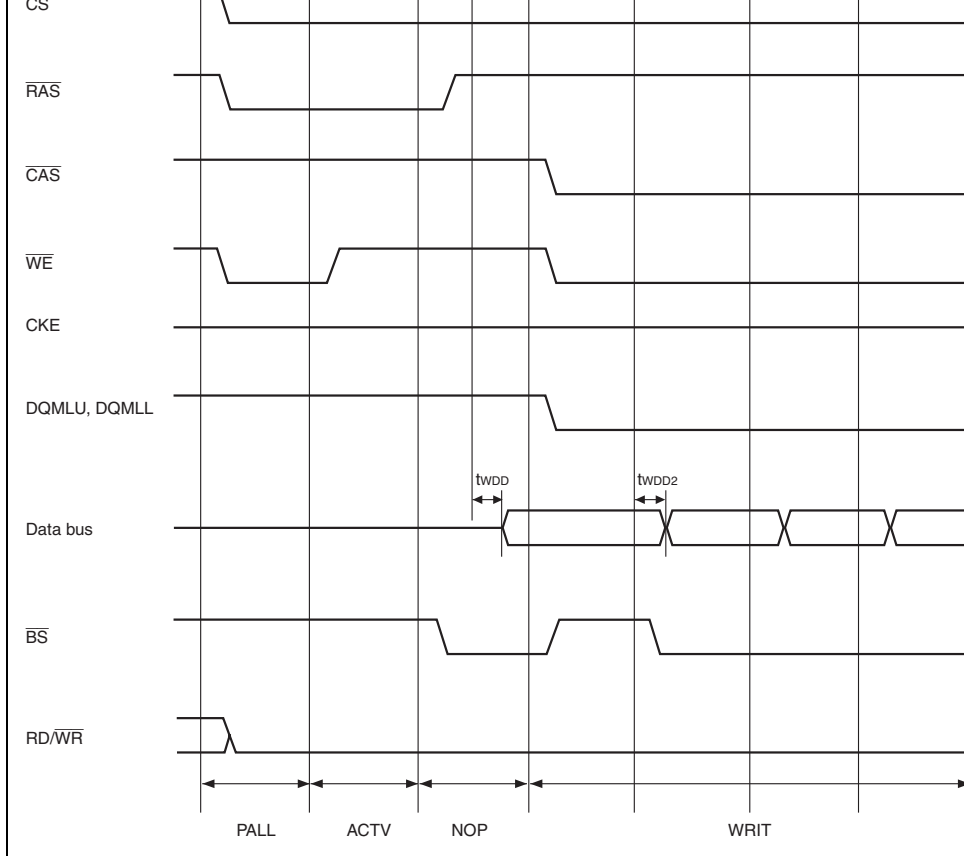


**Figure 30.31 Extended Read Data Cycle (CAS Latency 2)**

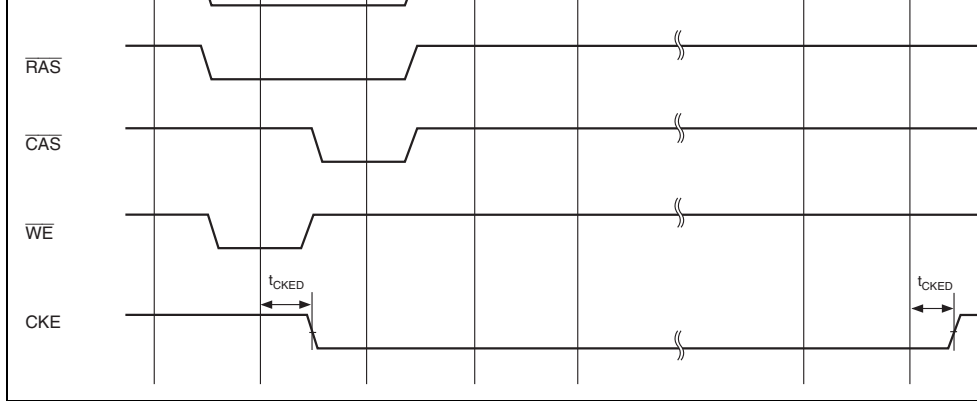


**Figure 30.32 Synchronous DRAM Cluster Transfer Access Timing (Read)**

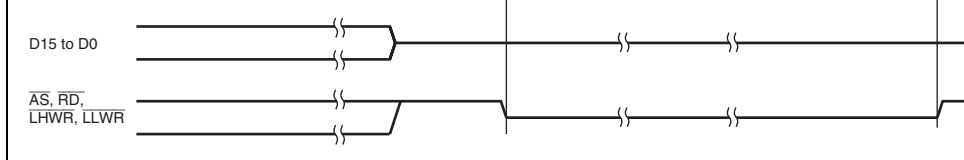




**Figure 30.33 Synchronous DRAM Cluster Transfer Access Timing (Write)**



**Figure 30.34 Synchronous DRAM Self-Refresh Timing**



**Figure 30.35 External Bus Release Timing**

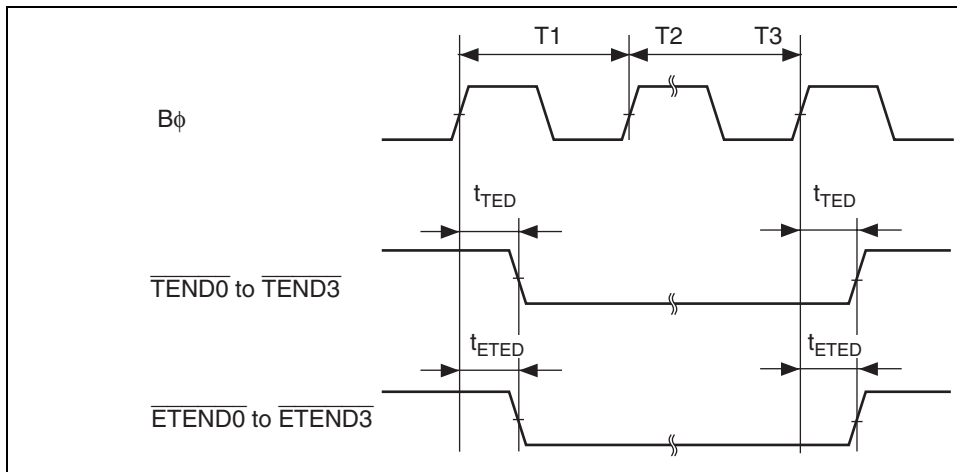
### 30.4.4 DMAC and EXDMAC Timing

**Table 30.9 DMAC and EXDMAC Timing**

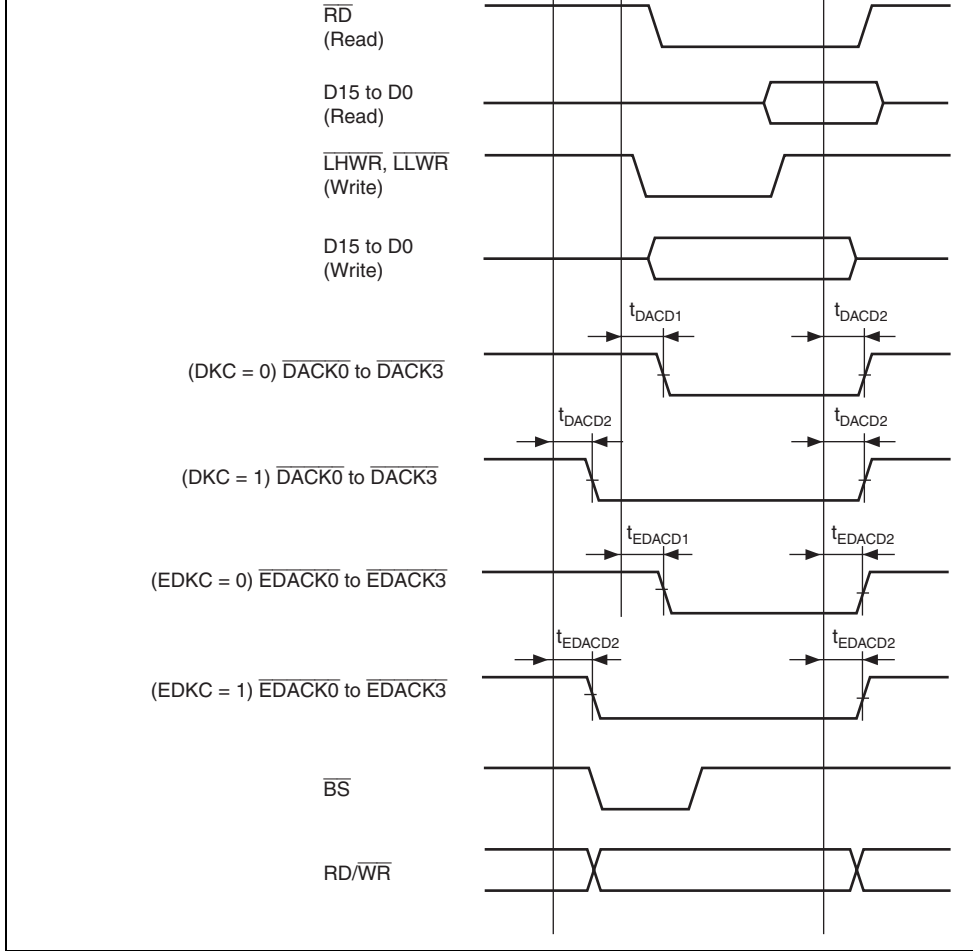
Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 3.0\text{ V to }3.6\text{ V}^*$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  
 $V_{ref} = 3.0\text{ V to }AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0\text{ V}$ ,  $B\phi = 8\text{ MHz to }50\text{ MHz}$ ,  
 $T_a = -20^\circ\text{C to }+75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to }+85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Con
$\overline{DREQ}$ setup time	$t_{DRQS}$	20	—	ns	Figure 30
$\overline{DREQ}$ hold time	$t_{DRQH}$	5	—	ns	
$\overline{TEND}$ delay time	$t_{TED}$	—	15	ns	Figure 30
$\overline{DACK}$ delay time 1	$t_{DADC1}$	—	15	ns	Figure 30
$\overline{DACK}$ delay time 2	$t_{DADC2}$	—	15	ns	Figure 30
$\overline{EDREQ}$ setup time	$t_{DRQS}$	20	—	ns	Figure 30
$\overline{EDREQ}$ hold time	$t_{DRQH}$	5	—	ns	
$\overline{ETEND}$ delay time	$t_{TED}$	—	15	ns	Figure 30
$\overline{EDACK}$ delay time 1	$t_{DADC1}$	—	15	ns	Figure 30
$\overline{EDACK}$ delay time 2	$t_{DADC2}$	—	15	ns	Figure 30
$\overline{EDRAK}$ delay time	$t_{EDRKD}$	—	15	ns	Figure 30

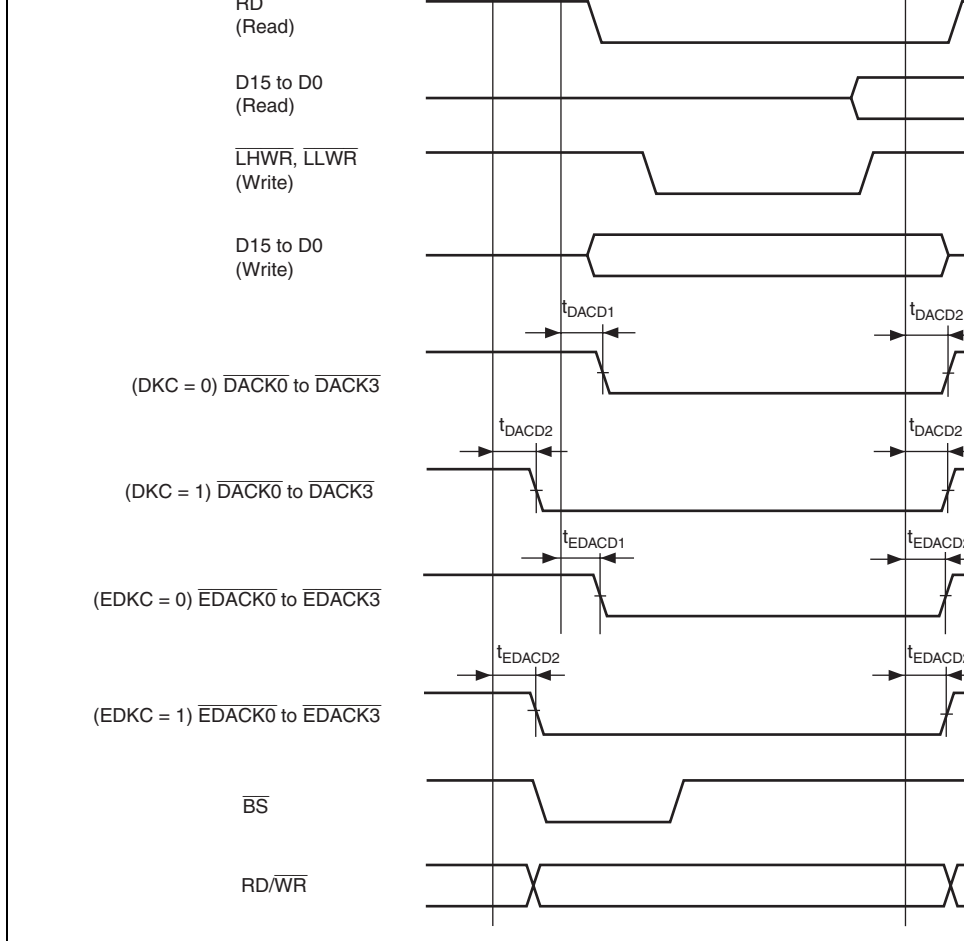
Note: \*  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95\text{ V to }3.60\text{ V}$  in the H8SX/1665M Group.



**Figure 30.38 DMAC/EXDMAC,  $\overline{TEND}$  and  $\overline{ETEND}$  Output Timing**



**Figure 30.39 DMAC/EXDMAC Single Address Transfer Timing (2-state access)**



**Figure 30.40 DMAC/EXDMAC Single Address Transfer Timing (3-state access)**

### 30.4.5 Timing of On-Chip Peripheral Modules

**Table 30.10 Timing of On-Chip Peripheral Modules**

Conditions:  $V_{CC} = PLLV_{CC} = DrV_{CC} = 3.0\text{ V to }3.6\text{ V}^{*2}$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  
 $V_{ref} = 3.0\text{ V to }AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0\text{ V}$ ,  $P\phi = 8\text{ MHz to }30\text{ MHz}$ ,  
 $T_a = -20^{\circ}\text{C to }+75^{\circ}\text{C}$  (regular specifications),  
 $T_a = -40^{\circ}\text{C to }+85^{\circ}\text{C}$  (wide-range specifications)

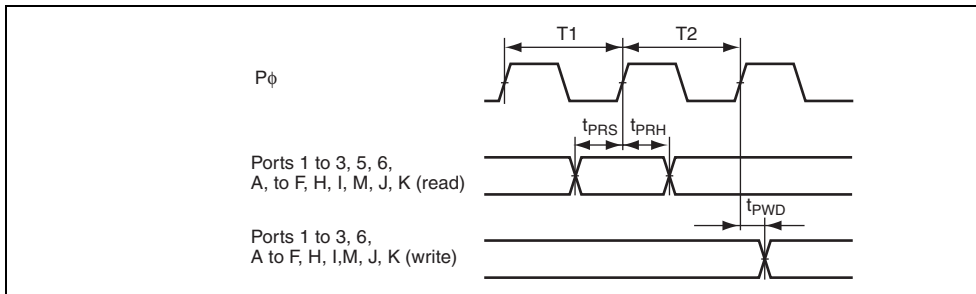
	Item	Symbol	Min.	Max.	Unit	Test Co	
I/O ports	Output data delay time	$t_{PWD}$	—	40	ns	Figure 3	
	Input data setup time	$t_{PRS}$	25	—	ns		
	Input data hold time	$t_{PRH}$	25	—	ns		
TPU	Timer output delay time	$t_{TOCD}$	—	40	ns	Figure 3	
	Timer input setup time	$t_{TICS}$	25	—	ns		
	Timer clock input setup time	$t_{TCKS}$	25	—	ns	Figure 3	
	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 3	



	cycle	Clocked synchronous	6	—		
	Input clock pulse width	$t_{SCKW}$	0.4	0.6	$t_{Scyc}$	
	Input clock rise time	$t_{SCKr}$	—	1.5	$t_{cyc}$	
	Input clock fall time	$t_{SCKf}$	—	1.5	$t_{cyc}$	
	Transmit data delay time	$t_{TXD}$	—	40	ns	Figure
	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
IIC2	SCL input cycle time	$t_{SCL}$	$12 t_{cyc} + 600$	—	ns	Figure
	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	

TCK clock low level pulse width	$t_{TCKL}$	20	—	ns	
TCK clock rising time	$t_{TCKr}$	—	5	ns	
TCK clock falling time	$t_{TCKf}$	—	5	ns	
TRST pulse width	$t_{TRSTW}$	20	—	T <sub>cyc</sub>	Figure 3
TMS setup time	$t_{TMSS}$	20	—	ns	Figure 3
TMS hold time	$t_{TMSh}$	20	—	ns	
TDI setup time	$t_{TDis}$	20	—	ns	
TDI hold time	$t_{TDIH}$	20	—	ns	
TDI data delay time	$t_{TDOD}$	—	23	ns	

- Notes: 1.  $t_{TCKcyc} \geq t_{TCKcyc}$   
2.  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95\text{ V to }3.60\text{ V}$  in the H8SX/1665M Group.



**Figure 30.42 I/O Port Input/Output Timing**

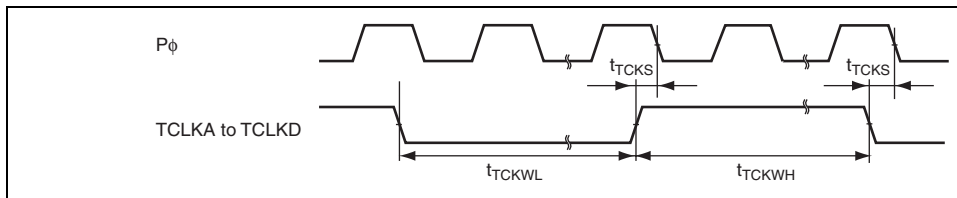


Figure 30.44 TPU Clock Input Timing

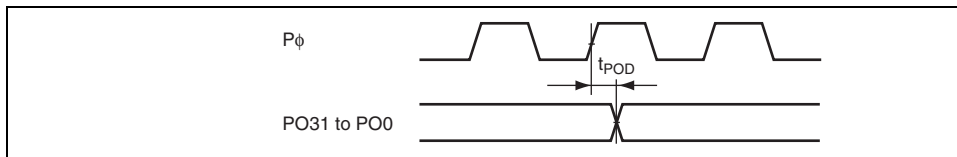


Figure 30.45 PPG Output Timing

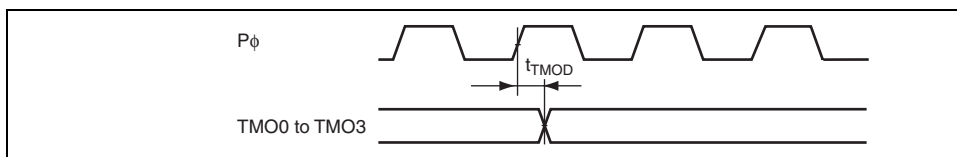
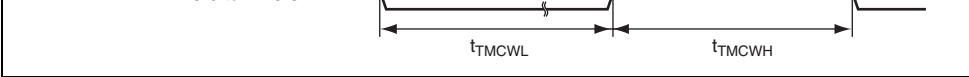
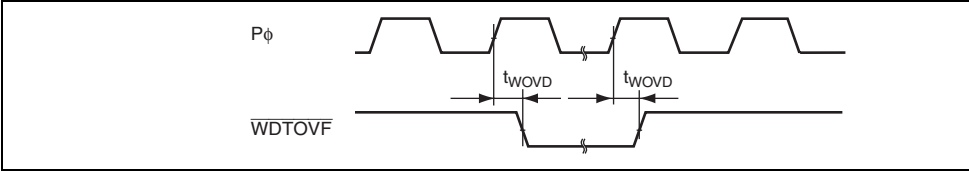


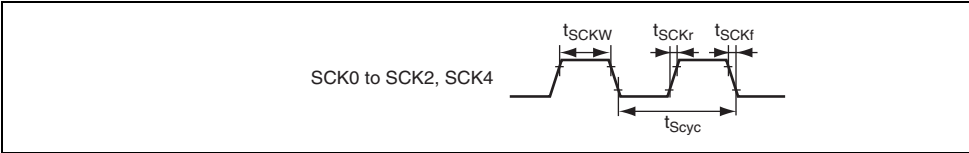
Figure 30.46 8-Bit Timer Output Timing



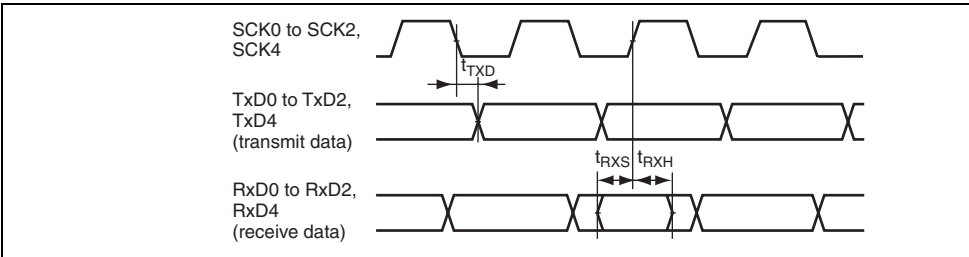
**Figure 30.48 8-Bit Timer Clock Input Timing**



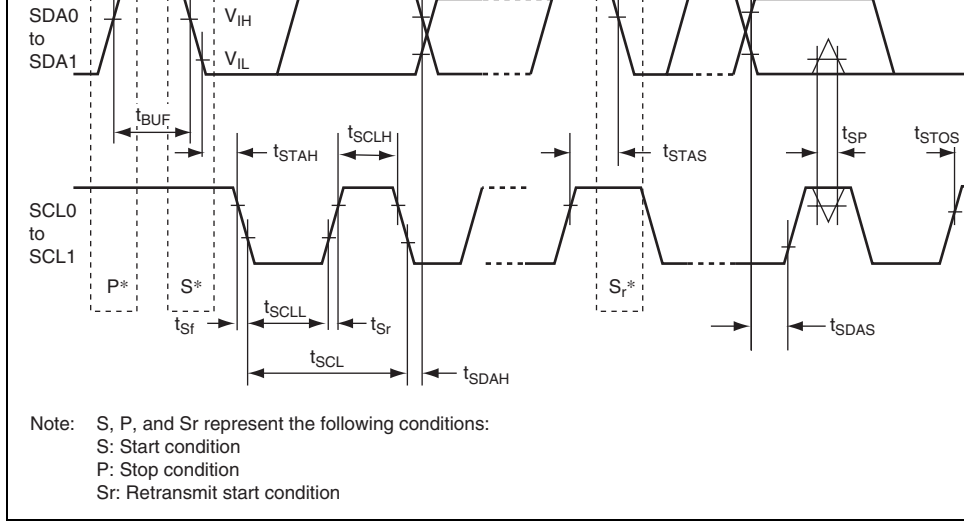
**Figure 30.49 WDT Output Timing**



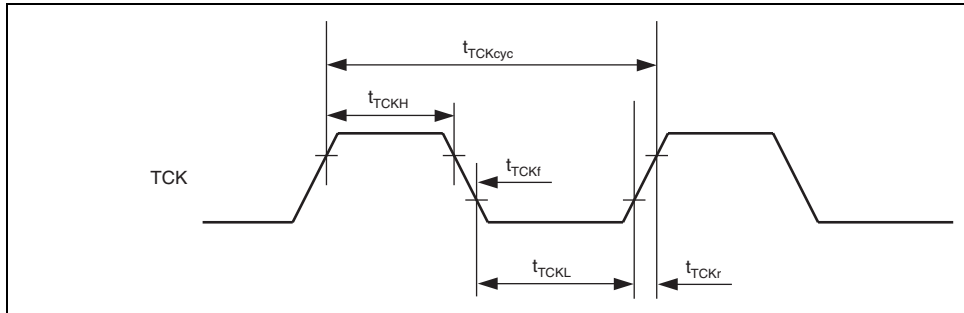
**Figure 30.50 SCK Clock Input Timing**



**Figure 30.51 SCI Input/Output Timing: Clocked Synchronous Mode**



**Figure 30.53 I<sup>2</sup>C Bus Interface2 Input/Output Timing (Option)**



**Figure 30.54 Boundary Scan TCK Timing**

Figure 30.55 Boundary Scan TRST Timing

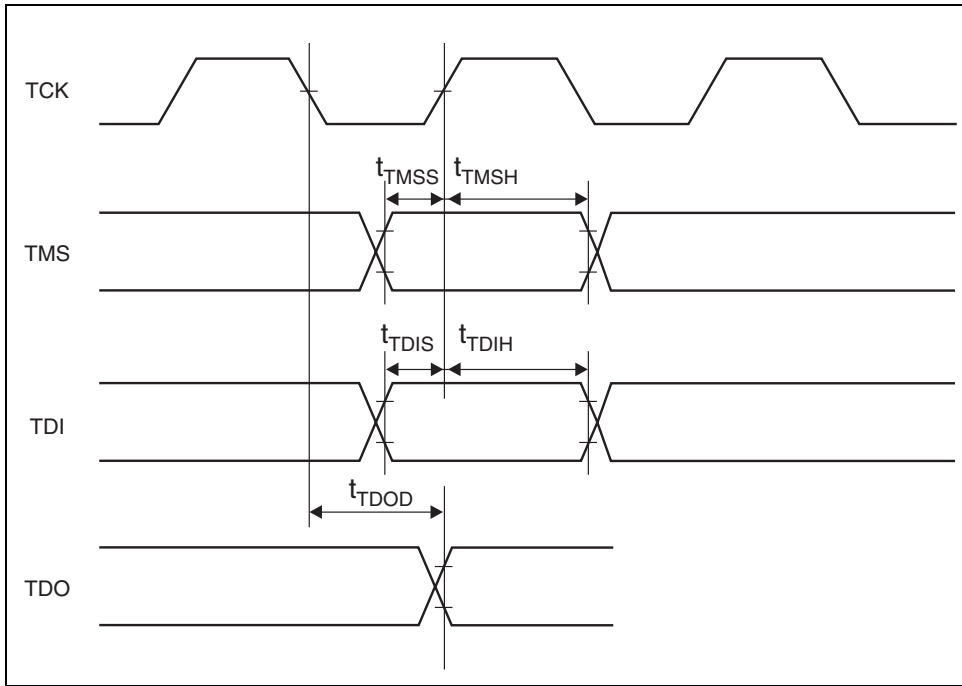
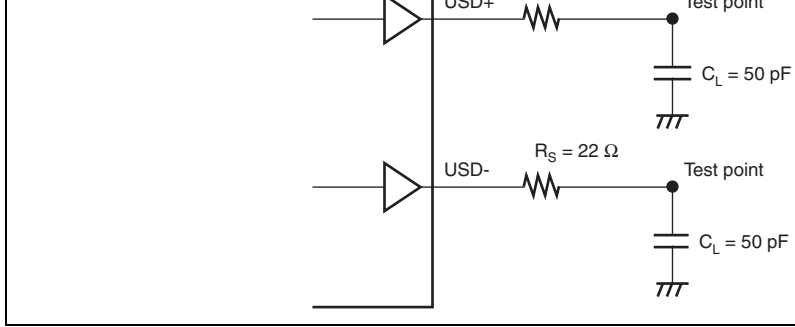


Figure 30.56 Boundary Scan Input/Output Timing

	Input low voltage	$V_{IL}$	—	0.8	V	
	Differential input sensitivity	$V_{DI}$	0.2	—	V	$ (D+) - (D-) $
	Differential common mode range	$V_{CM}$	0.8	2.5	V	
Output	Output high voltage	$V_{OH}$	2.8	—	V	$I_{OH} = -200 \mu A$
	Output low voltage	$V_{OL}$	—	0.3	V	$I_{OL} = 2 \text{ mA}$
	Crossover voltage	$V_{CRS}$	1.3	2.0	V	
	Rising time	$t_R$	4	20	ns	
	Falling time	$t_F$	4	20	ns	
	Ratio of rising time to falling time	$t_{RFM}$	90	111.11	%	$(T_R/T_F)$
	Output resistance	$Z_{DRV}$	28	44	$\Omega$	Including $R_s = 22\Omega$

Note: \*  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95 \text{ V to } 3.60 \text{ V}$  in the H8SX/1665M Group.



**Figure 30.58 Load Condition**



Resolution		10	10	10	Bit
Conversion time		2.7	—	—	$\mu$ s
Analog input capacitance		—	—	20	pF
Permissible signal source impedance	EXCKS = 0	—	—	5	k $\Omega$
	EXCKS = 1	—	—	1	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

Note: \*  $V_{cc} = PLLV_{cc} = DrV_{cc} = 2.95$  V to 3.60 V in the H8SX/1665M Group.

Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	1	k $\Omega$
Nonlinearity error	—	—	$\pm 7.5$	LSE
Offset error	—	—	$\pm 7.5$	LSE
Full-scale error	—	—	$\pm 7.5$	LSE
Quantization error	—	$\pm 0.5$	—	LSE
Absolute accuracy	—	—	$\pm 8.0$	LSE

Note: \*  $V_{cc} = PLLV_{cc} = DrV_{cc} = 2.95 \text{ V to } 3.60 \text{ V}$  in the H8SX/1665M Group.

Resolution	8	8	8	Bit	
Conversion time	—	—	10	μs	20-pF capac
Absolute accuracy	—	±2.0	±3.0	LSB	2-MΩ resist
	—	—	±2.0	LSB	4-MΩ resist

Note: \*  $V_{cc} = PLLV_{cc} = DrV_{cc} = 2.95\text{ V to }3.60\text{ V}$  in the H8SX/1665M Group.

**Table 30.15 10-Bit D/A Conversion Characteristics**

Conditions:  $V_{cc} = PLLV_{cc} = DrV_{cc} = 3.0\text{ V to }3.6\text{ V}^*$ ,  $AV_{cc} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{ref} = 3.0\text{ V}$ ,  
 $AV_{cc}, V_{ss} = PLLV_{ss} = DrV_{ss} = AV_{ss} = 0\text{ V}$ ,  $P\phi = 8\text{ MHz to }35\text{ MHz}$ ,  
DADT[1:0] = Don't care (used as 10-bit D/A converter),  
 $T_a = -20^\circ\text{C to }+75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to }+85^\circ\text{C}$  (wide-range specifications)

Item	Min.	Typ.	Max.	Unit	Test Condi
Resolution	10	10	10	Bit	
Conversion time	—	—	10	μs	20-pF capac
Absolute accuracy	—	±2.0	±3.0	LSB	16-MΩ resist

Note: \*  $V_{cc} = PLLV_{cc} = DrV_{cc} = 2.95\text{ V to }3.60\text{ V}$  in the H8SX/1665M Group.

Operating voltage range:  $V_{CC} = PLLV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  
 $V_{ref} = 3.0\text{ V to }AV_{CC}$ ,  $V_{SS} = PLLV_{SS} = DrV_{SS} = AV_{SS} = 0\text{ V}$

Item	Symbol	Min.	Typ.	Max.	Unit	Test Cond
Programming time* <sup>1</sup> , * <sup>2</sup> , * <sup>4</sup>	$t_p$	—	1	10	ms/128 bytes	
Erasure time* <sup>1</sup> , * <sup>2</sup> , * <sup>4</sup>	$t_E$	—	40	130	ms/4-Kbyte block	
		—	300	800	ms/32-Kbyte block	
		—	600	1500	ms/64-Kbyte block	
Programming time (total)* <sup>1</sup> , * <sup>2</sup> , * <sup>4</sup>	$\Sigma_{tp}$	—	3.4	9	H8SX/1662, H8SX/1662Ms/384 Kbytes	$T_a = 25^\circ\text{C}$ for all
		—	4.5	12	H8SX/1665, H8SX/1665Ms/512 Kbytes	
Erasure time (total)* <sup>1</sup> , * <sup>2</sup> , * <sup>4</sup>	$\Sigma_{tE}$	—	3.4	9	H8SX/1662, H8SX/1662Ms/384 Kbytes	$T_a = 25^\circ\text{C}$
		—	4.5	12	H8SX/1665, H8SX/1665Ms/512 Kbytes	

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
5.  $V_{CC} = PLLV_{CC} = DrV_{CC} = 2.95 \text{ V to } 3.60 \text{ V}$  in the H8SX/1665M Group.

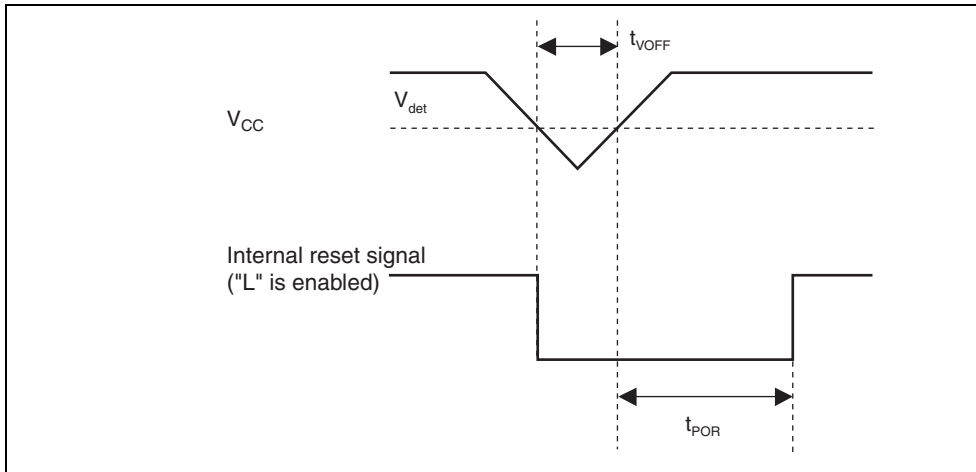
### 30.9 Power-On Reset Circuit and Voltage-Detection Circuit Characteristics (H8SX/1665M Group)

**Table 30.17 Power-On Reset Circuit and Voltage-Detection Circuit Characteristics**

Conditions:  $V_{CC} = PLLV_{CC}$ ,  $AV_{CC} = 3.0 \text{ V to } 3.6 \text{ V}$ ,  $V_{ref} = 3.0 \text{ V to } AV_{CC}$ ,  
 $V_{SS} = PLLV_{SS} = AV_{SS} = 0 \text{ V}$ ,  
 $T_a = -20^\circ\text{C to } +75^\circ\text{C}$  (regular specifications),  
 $T_a = -40^\circ\text{C to } +85^\circ\text{C}$  (wide-range specifications)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Condition
Voltage detection level	Voltage detection circuit (LVD) $V_{det}$	3.00	3.10	3.20	V	Fig. 30.17
	Power-on reset (POR) $V_{POR}$	2.48	2.58	2.68		Fig. 30.18
Internal reset time	$t_{POR}$	20	35	50	ms	Fig. 30.19
Power-off time*	$t_{VOFF}$	200	—	—	$\mu\text{s}$	Fig. 30.20

Note: \* Power-off time ( $t_{VOFF}$ ) is the time over which  $V_{CC}$  is lower than minimum value voltage-detection level of the POR and LVD.

**Figure 30.59 Power-On Reset Timing****Figure 30.60 Voltage Detection Circuit Timing**

Port 2	All	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep
Port 3	All	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep
P55 to P50	All	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
P56/ AN6/ DA0/ $\overline{\text{IRQ6}}$ -B	All	Hi-Z	Hi-Z	Hi-Z	Hi-Z	[DAOE0 = 1]	[DAOE0 = 1]
						Keep	Keep
						[DAOE0 = 0]	[DAOE0 = 0]
P57/ AN7/ DA1/ $\overline{\text{IRQ7}}$ -B	All	Hi-Z	Hi-Z	Hi-Z	Hi-Z	[DAOE1 = 1]	[DAOE1 = 1]
						Keep	Keep
						[DAOE1 = 0]	[DAOE1 = 0]
P65 to P60	All	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep
PA0/ $\overline{\text{BREQO}}$ / BS-A	All	Hi-Z	Hi-Z	[BREQO output]	[BREQO output]	[BREQO output]	[BREQO output]
				Hi-Z	Hi-Z	Hi-Z	Hi-Z
				$\overline{\text{BS}}$ output]	$\overline{\text{BS}}$ output]	$\overline{\text{BS}}$ output]	$\overline{\text{BS}}$ output]
				Keep	Hi-Z	Keep	Hi-Z
				[Other than above]	[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep	Keep
PA1/ $\overline{\text{BACK}}$ / (RD/WR-A)	All	Hi-Z	Hi-Z	[ $\overline{\text{BACK}}$ output]	[ $\overline{\text{BACK}}$ output]	[ $\overline{\text{BACK}}$ output]	[ $\overline{\text{BACK}}$ output]
				Hi-Z	Hi-Z	Hi-Z	Hi-Z
				[RD/WR output]	[RD/WR output]	[RD/WR output]	[RD/WR output]
				Keep	Hi-Z	Keep	Hi-Z
				[Other than above]	[Other than above]	[Other than above]	[Other than above]
				Keep	Keep	Keep	Keep

LLB	(EXPE = 0)								
	External extended mode (EXPE = 1)	H	Hi-Z	H	Hi-Z	H	Hi-Z	H	Hi-Z
PA4/ LHWR/ LUB	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep	Keep	Keep
	External extended mode (EXPE = 1)	H	Hi-Z	[LHWR, LUB output] H [Other than above] Keep	[LHWR, LUB output] Hi-Z [Other than above] Keep	[LHWR, LUB output] H [Other than above] Keep	[LHWR, LUB output] Hi-Z [Other than above] Keep	[LHWR, LUB output] H [Other than above] Keep	[LHWR, LUB output] Hi-Z [Other than above] Keep
PA5/RD	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep	Keep	Keep
	External extended mode (EXPE = 1)	H	Hi-Z	H	Hi-Z	H	Hi-Z	H	Hi-Z
PA6/ AS/ AH/ BS-B	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	[AS, BS output] H	[AS, AH, BS output] Hi-Z	[AS, BS output] H	[AS, AH, BS output] Hi-Z	[AS, BS output] H	[AS, AH, BS output] Hi-Z
	External extended mode (EXPE = 1)	H	Hi-Z	[AH output] L [Other than above] Keep	[Other than above] Keep	[AH output] L [Other than above] Keep	[Other than above] Keep	[AH output] L [Other than above] Keep	[Other than above] Keep



CS5-B	External extended mode (EXPE = 1)	H	Hi-Z	[Other than above]	[Other than above]	[Other than above]	[Other than above]
PB1/ $\overline{\text{CS1}}$ / $\overline{\text{CS2-B}}$ / $\overline{\text{CS5-A}}$ / $\overline{\text{CS6-B}}$ / $\overline{\text{CS7-B}}$	All	Hi-Z	Hi-Z	$\overline{\text{CS}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ output Hi-Z [Other than above] Keep	$\overline{\text{CS}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ output Hi-Z [Other than above] Keep
PB2/ $\overline{\text{CS2-A}}$ / $\overline{\text{CS6-A}}$ / $\overline{\text{RAS}}$	All	Hi-Z	Hi-Z	$\overline{\text{CS}}$ , $\overline{\text{RAS}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{RAS}}$ output Hi-Z [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{RAS}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{RAS}}$ output Hi-Z [Other than above] Keep
PB3/ $\overline{\text{CS3}}$ / $\overline{\text{CS7-A}}$ / $\overline{\text{CAS}}$	All	Hi-Z	Hi-Z	$\overline{\text{CS}}$ , $\overline{\text{CAS}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{CAS}}$ output Hi-Z [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{CAS}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{CAS}}$ output Hi-Z [Other than above] Keep
PB4/ $\overline{\text{CS4-B}}$ / $\overline{\text{WE}}$	All	Hi-Z	Hi-Z	$\overline{\text{CS}}$ , $\overline{\text{WE}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{WE}}$ output Hi-Z [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{WE}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{WE}}$ output Hi-Z [Other than above] Keep
PB5/ $\overline{\text{CS5-B}}$ / $\overline{\text{OE}}$	All	Hi-Z	Hi-Z	$\overline{\text{CS}}$ , $\overline{\text{OE}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{OE}}$ output Hi-Z [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{OE}}$ output H [Other than above] Keep	$\overline{\text{CS}}$ , $\overline{\text{OE}}$ output Hi-Z [Other than above] Keep

	mode	output		H	H	H	H	[Other than above]	[Other than above]	[Other than above]	[Other than above]
	Other than SDRAM mode	H	Hi-Z	[Other than above]	[Other than above]	[Other than above]	[Other than above]	Keep	Keep	Keep	Keep
PC2 to PC3	All	Hi-Z	Hi-Z	Keep				Keep			

	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep
Port E	External extended mode (EXPE = 1)	L	Hi-Z	Keep	Hi-Z	Keep	Hi-Z
	ROM enabled extended mode	Hi-Z	Hi-Z	Keep	[Address output] Hi-Z [Other than above] Keep	Keep	[Address output] Hi-Z [Other than above] Keep
	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep
PF3 to PF0	External extended mode (EXPE = 1)	L	Hi-Z	Keep	Hi-Z	Keep	Hi-Z
	ROM enabled extended mode	Hi-Z	Hi-Z	Keep	[Address output] Hi-Z [Other than above] Keep	Keep	[Address output] Hi-Z [Other than above] Keep
	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep
PF7 to PF4	External extended mode (EXPE = 1)	Hi-Z	Hi-Z	Keep	[Address output] Hi-Z [Other than above] Keep	Keep	[Address output] Hi-Z [Other than above] Keep
	Single-chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	Keep	Keep	Keep

	mode (EXPE = 1)	mode 16-bit bus mode	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	H
Port J	Hi-Z		Hi-Z	Hi-Z	Keep	Keep	Keep	Keep	K
Port K	Hi-Z		Hi-Z	Hi-Z	Keep	Keep	Keep	Keep	K
Port M	Hi-Z		Hi-Z	Hi-Z	Keep	Keep	Keep	Keep	K

[Legend]

H: High-level output

L: Low-level output

Keep: Input pins become high-impedance, output pins retain their state.

Hi-Z: High impedance

H8SX/1665M

R5F61665M

R5F61665MFPV

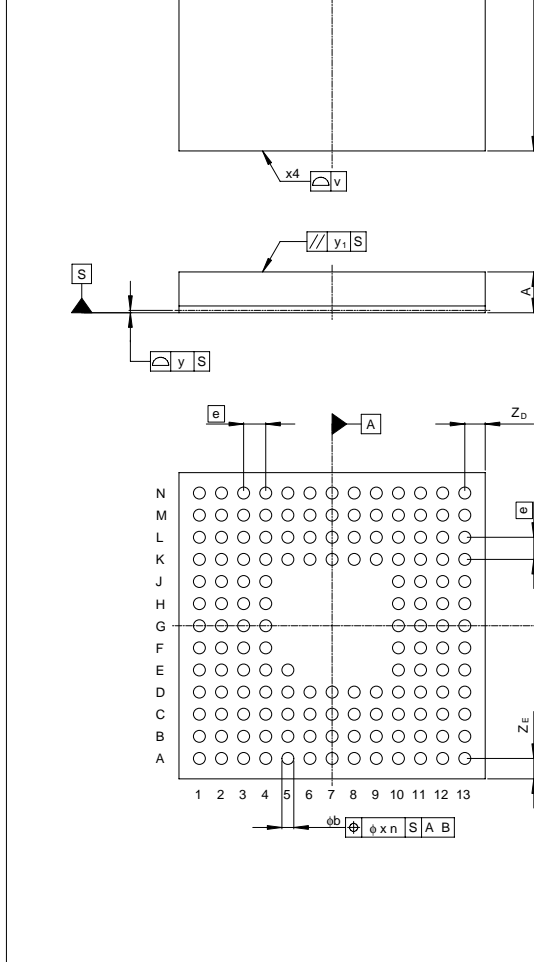
PLQP0144KA-A (FP-

R5F61665MLGV

PTLG0145JB-A (TLP-

Note: \* Pb-free version





Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	—	9.0	—
E	—	9.0	—
v	—	—	0.1
w	—	—	0.1
A	—	—	—
A <sub>1</sub>	—	—	—
e	—	0.65	—
b	0.30	0.35	0.4
x	—	—	0.1
y	—	—	—
y <sub>1</sub>	—	—	0.1
S <sub>D</sub>	—	—	—
S <sub>E</sub>	—	—	—
Z <sub>D</sub>	—	0.6	—
Z <sub>E</sub>	—	0.6	—

Figure C.2 Package Dimensions (TLP-145V)

MD_CLK	(Always used as mode pins)
MD3 to MD0	(Always used as mode pins)
NMI	<ul style="list-style-type: none"> <li>• Connect this pin to Vcc via a pull-up resistor</li> </ul>
EXTAL	(Always used as a clock pin)
XTAL	<ul style="list-style-type: none"> <li>• Leave this pin open</li> </ul>
OSC1	<ul style="list-style-type: none"> <li>• Connect this pin to Vss via a pull-down resistor</li> </ul>
OSC2	<ul style="list-style-type: none"> <li>• Leave this pin open</li> </ul>
WDTOVF	<ul style="list-style-type: none"> <li>• Leave this pin open</li> </ul>
USD+	<ul style="list-style-type: none"> <li>• Leave this pin open</li> </ul>
USD-	<ul style="list-style-type: none"> <li>• Leave this pin open</li> </ul>
VBUS	<ul style="list-style-type: none"> <li>• Leave this pin open</li> </ul>
Port 1	<ul style="list-style-type: none"> <li>• Connect these pins to Vcc via a pull-up resistor or to Vss via a pull-down resistor, respectively</li> </ul>
Port 2	
Port 3	
Port 6	
PA2 to PA0	
PB7 to PB1	
Port C	
PF7 to PF5	
Port J	
Port K	
Port M	
Port 5	<ul style="list-style-type: none"> <li>• Connect these pins to AVcc via a pull-up resistor or to AVss via a pull-down resistor, respectively</li> </ul>



Port D	<ul style="list-style-type: none"> <li>These pins are left open in the initial state for the address output.</li> </ul>	
Port E		
PF4 to PF0		
Port H	(Used as a data bus)	
Port I	(Used as a data bus)	<ul style="list-style-type: none"> <li>Connect these pins to VCC via a pull-up resistor or to VSS via a pull-down resistor, respectively, in the initial state for the general input.</li> </ul>
Vref	<ul style="list-style-type: none"> <li>Connect this pin to AVcc</li> </ul>	

- Notes:
- Do not change the initial value (input-buffer disabled) of PnICR, where n corresponds to an unused pin.
  - When the pin function is changed from its initial state, use a pull-up or pull-down resistor as needed.





6. Exception Handling
7. Interrupt Controller
8. User Break Controller (UBC)
9. Bus Controller (BSC)
10. DMA Controller (DMAC)
11. EXDMA Controller (EXDMAC)
12. Data Transfer Controller (DTC)
13. I/O Ports
14. 16-Bit Timer Pulse Unit (TPU)
15. Programmable Pulse Generator (PPG)
16. 8-Bit Timers (TMR)
17. 32K Timer (TM32K)
18. Watchdog Timer (WDT)
19. Serial Communication Interface (SCI, IrDA, CRC)
20. USB Function Module (USB)
21. I <sup>2</sup> C Bus Interface 2 (IIC2)
22. A/D Converter
23. D/A Converter
24. RAM
25. Flash Memory
26. Boundary Scan
27. Clock Pulse Generator
28. Power-Down Modes
29. List of Registers
30. Electrical Characteristics
Appendix

Section 7 Interrupt Controller 145 Amended

7.5 Interrupt Exception Handling Vector Table

Table 7.2 Interrupt Sources, Vector Address Offsets, and Interrupt Priority

Classification	Interrupt Source	Vector Number	Vector Address Offset*1		IPR	Priority	DTC Activation
			Mode, Middle	Mode, Maximum			
—	Reserved for system use	76	H'0130	—	—	High	—
		77	H'0134	—			
LVD*2	Voltage-monitoring interrupt (IRQ14)	78	H'0138	—	IPRD6 to IPRD4	↑	—
TM32K	32KOV1 (IRQ15)	79	H'013C	—	IPRD2 to IPRD0		Low

Section 13 I/O Ports 601 Notes added

13.1 Register Descriptions

Table 13.2 Register Configuration in Each Port

- Notes:
1. Bits 2 and 3 are valid and the other bits are reserved for port C registers. The write value should be the same as the initial value.
  2. Do not access when PCJKE = 1.
  3. Do not access when PCJKE = 0.
  4. The lower six bits are valid and the upper two bits are reserved for port 6 registers. The write value should be the same as the initial value.
  5. The lower five bits are valid and the upper three bits are reserved for port M registers. The write value should be the same as the initial value.

22.4.3 Input Sampling  
and A/D Conversion  
Time

1086  
to  
1088

The following tables replaced  
Table 22.3 Characteristics of A/D Conversion (Unit 0: wh  
EXCKS\* = 0, ICKSEL = 0, and ADSSTR\* = H'0F) (1)  
Table 22.3 Characteristics of A/D Conversion (Unit 0: wh  
EXCKS\* = 1, ICKSEL = 0, and ADSSTR\* = H'0F) (2)  
Table 22.4 Characteristics of A/D Conversion (Unit 1: wh  
EXCKS = 0, ICKSEL = 0, and ADSSTR\* = H'0F) (1)  
Table 22.4 Characteristics of A/D Conversion (Unit 1: wh  
EXCKS = 1, ICKSEL = 0, and ADSSTR\* = H'0F) (2)  
Table 22.5 Characteristics of A/D Conversion (When EX  
1, ICKSEL\*<sup>1</sup> = 0, and ADSSTR\*<sup>2</sup> = H'19)

Table 22.6 Period of A/D  
Conversion (Scan Mode)  
(Units 0 and 1)

1089

Note deleted  
Notes: 1. Make the sampling setting15 (ADSSRT = D'1  
2. When  $P\phi = I\phi/2$ , make the sampling setting 25  
(ADSSRT = D'25).  
3. Unit 0: The full-spec emulator (E6000H) shou  
used, but the on-chip emulator (E10A-USB) is  
4. ~~Unit 0: The full-spec emulator (E6000H) shou  
used, but the on-chip emulator (E10A-USB) is~~  
Unit 1: Access to the full-spec emulator (E600  
prohibited but the on-chip emulator (E10A-US  
usable.

29.2 Register Bits

Register	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit
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	23/15/7/0
DPSIER	—	DUSBIE	DT32KIE	DLVDIE	DIRQ3E	DIRQ2E	DIRQ1E	DIRQ0E	DIRQ0E
DPSIFR	DNMIF	DUSBIF	DT32KIF	DLVDIF	DIRQ3F	DIRQ2F	DIRQ1F	DIRQ0F	DIRQ0F
DPSIEGR	DNMIEG	—	—	—	DIRQ3EG	DIRQ2EG	DIRQ1EG	DIRQ0EG	DIRQ0EG
RSTSR	DPSRSTF	—	—	—	—	LVDF	—	—	PO
LVDCR <sup>3</sup>	LVDE	LVDRI	—	LVDMON	—	—	—	—	—

Appendix

1438 Amended

D. Treatment of Unused Pins

Table D.1 Treatment of Unused Pins

Pin Name	Mode 4	Mode 5	Mode 6	Mode 7
OSC1	• Connect this pin to Vss via a pull-down resistor.			





8-bit timers (TMK) ..... 805

## A

A/D conversion accuracy.....	1093
Absolute accuracy.....	1093
Acknowledge.....	1043
Address error.....	116
Address map.....	85
Address mode.....	527
Address modes.....	398, 478
Address/data multiplexed I/O interface.....	230, 266
All-module-clock-stop mode.....	1246, 1271
Area 0.....	232
Area 1.....	233
Area 2.....	233
Area 3.....	234
Area 4.....	234
Area 5.....	235
Area 6.....	235
Area 7.....	236
Area division.....	225
Asynchronous mode.....	908
AT-cut parallel-resonance type.....	1237
Available output signal and settings in each port.....	649
Average transfer rate generator.....	864

Boot mode.....	
Boundary scan commands.....	
Buffer operation.....	
Bulk-in transfer.....	
Bulk-out transfer.....	
Burst access mode.....	
Burst mode.....	
Burst ROM interface.....	
Bus access modes.....	
Bus arbitration.....	
Bus configuration.....	
Bus controller (BSC).....	
Bus cycle division.....	
Bus mode.....	
Bus release.....	
Bus width.....	
Bus-released state.....	
Byte control SRAM interface.....	

## C

Cascaded connection.....	
Cascaded operation.....	
Chain transfer.....	
Chip select signals.....	
Clock pulse generator.....	
Clock synchronization cycle (Tsy.....	
Clocked synchronous mode.....	

CPU priority control function over DTC and DMAC .....	161
CRC operation circuit.....	954
Crystal resonator.....	1237
Cycle steal mode.....	487
Cycle stealing mode.....	409

## D

D/A converter .....	1103
Data direction register .....	602
Data register.....	603
Data stage .....	1009
Data transfer controller (DTC) .....	555
Direct convention .....	933
DMA controller (DMAC).....	371
Double-buffered structure.....	908
Download pass/fail result parameter.....	1136
DRAM interface .....	276
DRAM Interface.....	230
DTC vector address .....	568
DTC vector address offset .....	568, 569, 570
Dual address mode.....	398, 478

## E

Endian and data alignment .....	237
Endian format .....	228
Error protection .....	1177

External bus interface .....	
External clock .....	
External interrupts.....	

## F

Flash erase block select parameter .....	
Flash memory .....	
Flash multipurpose address area parameter .....	
Flash multipurpose data destination parameter .....	
Flash pass and fail parameter .....	
Flash program/erase frequency parameter .....	1
Free-running count operation.....	
Frequency divider .....	1
Full address mode .....	
Full-scale error .....	

## G

General illegal instructions .....	
------------------------------------	--

## H

Hardware protection .....	
Hardware standby mode .....	1

Internal system bus .....	216
Interrupt .....	119
Interrupt control mode 0 .....	152
Interrupt control mode 2 .....	154
Interrupt controller .....	125
Interrupt exception handling sequence ...	156
Interrupt exception handling vector table .....	145
Interrupt response times .....	157
Interrupt sources .....	143
Interrupt sources and vector address offsets .....	145
Interrupt-in transfer .....	1015
Interval timer .....	858
Interval timer mode .....	858
Inverse convention .....	934
IRQn interrupts .....	143

## J

JTAG interface .....	1061
----------------------	------

## L

Little endian .....	228
---------------------	-----

## M

Mark state .....	908, 949
------------------	----------

Multiprocessor bit .....	
Multiprocessor communication function .....	

## N

NMI interrupt .....	
Noise canceler .....	
Nonlinearity error .....	
Non-overlapping pulse output .....	
Normal transfer mode .....	
Normal transfer mode .....	
Number of access cycles .....	

## O

Offset addition .....	
Offset addition method .....	
Offset error .....	
On-board programming .....	
On-board programming mode .....	
On-chip baud rate generator .....	
On-chip ROM disabled extended r .....	
On-chip ROM enabled extended r .....	
Open-drain control register .....	
Oscillator .....	
Output buffer control .....	
Output trigger .....	
Overflow .....	

Power-down modes .....	1245
Procedure program .....	1170
Processing states .....	74
Product lineup.....	1435
Program execution state.....	74
Program stop state .....	74
Programmable pulse generator (PPG) ....	769
Programmer mode .....	1117, 1183
Programming/erasing interface.....	1122
Programming/erasing interface parameters.....	1134
Programming/erasing interface register .....	1127
Protection.....	1176
Pull-up MOS control register.....	605
PWM modes .....	738

## Q

Quantization error.....	1093
-------------------------	------

## R

RAM .....	1113
Read strobe ( $\overline{RD}$ ) timing.....	249
Register addresses .....	1296
Register Bits .....	1315
Register configuration in each port.....	601

CRA .....	
CRB .....	
CRCCR.....	
CRCDIR .....	
CRCDOR.....	
CSACR .....	
CTLR .....	
CVR .....	
DACR .....	
DACR01 .....	
DADR01T.....	
DADR0H .....	
DADR0L.....	
DADR1H .....	
DADR1L.....	
DAR.....	
DASTS.....	
DBSR.....	
DDAR.....	
DDR.....	
DMA .....	
DMDR .....	
DMRSR .....	
DOFR.....	
DPFR .....	
DR.....	
DRACCR.....	
DRAMCR .....	
DSAR.....	

ENDIANCR.....	199	ISCRL.....	
EPDR.....	976	ISR.....	
EPDR0i.....	974	ISR (USB).....	
EPDR0o.....	975	LVDCR.....	
EPDR0s.....	975	MAC.....	
EPIR.....	988	MDCR.....	
EPSTL.....	985	MPXCR.....	
EPSZ0o.....	977	MRA.....	
EPSZ1.....	978	MRB.....	
EXR.....	44	MSTPCRA.....	
FCCS.....	1127	MSTPCRB.....	
FCLR.....	981	MSTPCRC.....	
FEBS.....	1145	NDERH.....	
FECS.....	1130	NDERL.....	
FKEY.....	1131	NDRH.....	
FMATS.....	1132	NDRL.....	
FMPAR.....	1143	ODR.....	
FMPDR.....	1144	PC.....	
FPCS.....	1130	PCR.....	
FPEFEQ.....	1141, 1160	PCR (I/O port).....	
FPFR.....	1137	PFCR0.....	
FTDAR.....	1133	PFCR1.....	
General registers.....	41	PFCR2.....	
ICCRA.....	1031	PFCR4.....	
ICCRB.....	1032	PFCR6.....	
ICDRR.....	1042	PFCR7.....	
ICDRS.....	1042	PFCR9.....	
ICDRT.....	1042	PFCRB.....	
ICIER.....	1035	PFCRC.....	

RTCNT .....	215
RTCOR .....	215
SAR .....	560, 1041
SBR .....	45
SBYCR .....	1250
SCKCR .....	1233
SCMR .....	890
SCR .....	876
SDBPR .....	1219
SDBSR .....	1219
SDCR .....	210
SDID .....	1225
SEMR .....	898
SMR .....	872
SRAMCR .....	200
SSIER .....	141
SSR .....	881
SYSCR .....	81
TCCR .....	816
TCNT .....	721
TCNT (TMR) .....	813
TCNT (WDT) .....	853
TCORA .....	813
TCORB .....	814
TCR .....	691
TCR (TMR) .....	814
TCR32K .....	842
TCSR (TMR) .....	821
TCSR (WDT) .....	853

WTCRA .....	
WTCRB .....	
Repeat transfer mode .....	403
Reset .....	
Reset state .....	
Resolution .....	

## S

Sample-and-hold circuit .....	
Scan mode .....	
SDRAM interface .....	
Serial communication interface (SC) .....	
Setup stage .....	
Short address mode .....	
Single address mode .....	
Single mode .....	
Slave receive mode .....	
Slave transmit mode .....	
Sleep instruction exception handling .....	
Sleep mode .....	12
Slot illegal instructions .....	
Smart card interface .....	
Software protection .....	
Software standby mode .....	12
Space state .....	
Stack status after exception handling .....	
Stall operations .....	

Synchronous presetting..... 730  
System clock (I $\phi$ )..... 217, 1231

Vector table address.....  
Vector table address offset.....  
Voltage Detection Circuit (LVD).

## T

TAP controller ..... 1226  
Toggle output..... 727  
Trace exception handling..... 115  
Transfer information..... 566  
Transfer information read skip  
function..... 575  
Transfer information writeback skip  
function..... 576  
Transfer modes ..... 402, 481  
Transmit/receive data..... 908  
Trap instruction exception handling ..... 120

## W

Wait control .....  
Watchdog timer (WDT).....  
Watchdog timer mode.....  
Waveform output by compare ma  
Write data buffer function.....  
Write data buffer function for exte  
data bus .....  
Write data buffer function for peri  
modules.....





---

**Renesas 32-Bit CISC Microcomputer  
Hardware Manual  
H8SX/1665 Group, H8SX/1665M Group**

Publication Date: Rev.1.00, Mar. 25, 2009  
Rev.2.00, Oct. 21, 2009  
Published by: Sales Strategic Planning Div.  
Renesas Technology Corp.  
Edited by: Customer Support Department  
Global Strategic Communication Div.  
Renesas Solutions Corp.

---

© 2009. Renesas Technology Corp., All rights reserved. Printed in Japan.



**RENESAS SALES OFFICES**

<http://www.ren>

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, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120  
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7858/7898

**Renesas Technology Hong Kong Ltd.**  
7th Floor, North Tower, World Finance Centre, Harbour City, Canton Road, Tsimshatsui, Kowloon, Hong Kong  
Tel: <852> 2265-6688, Fax: <852> 2377-3473

**Renesas Technology Taiwan Co., Ltd.**  
10th Floor, No.99, Fushing North Road, Taipei, Taiwan  
Tel: <886> (2) 2715-2888, Fax: <886> (2) 3518-3399

**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, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, M  
Tel: <603> 7955-9390, Fax: <603> 7955-9510





# H8SX/1665 Group, H8SX/1665M Group Hardware Manual



Renesas Electronics Corporation

1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ09B0498-0200

## X-ON Electronics

Largest Supplier of Electrical and Electronic Components

*Click to view similar products for [32-bit Microcontrollers - MCU category](#):*

*Click to view products by [Renesas manufacturer](#):*

Other Similar products are found below :

[MB91F575BHSPMC-GSE1](#) [MB91F594BSPMC-GSE1](#) [PIC32MX120F032B-50I/ML](#) [MB91F464AAPMC-GSE2](#) [MB91F577BHSPMC-GSE1](#)  
[SPC5604EEF2MLH](#) [MB91F528USCPMC-GSE2](#) [MB91F248PFV-GE1](#) [MB91F594BPMC-GSE1](#) [MB91243PFV-GS-136E1](#)  
[MB91F577BHSPMC1-GSE1](#) [PIC32MM0032GPL020-E/ML](#) [PIC32MM0032GPL020-E/SS](#) [MEC1632X-AUE](#) [PIC32MM0016GPL020-E/ML](#)  
[PIC32MM0016GPL020-E/SS](#) [PIC32MM0016GPL028-E/SS](#) [PIC32MM0016GPL028-E/SO](#) [PIC32MM0016GPL028-E/ML](#)  
[PIC32MM0032GPL028-E/SS](#) [PIC32MM0032GPL028-E/SO](#) [PIC32MM0032GPL028-E/ML](#) [PIC32MM0032GPL036-E/MV](#)  
[PIC32MM0064GPL028-E/M6](#) [PIC32MM0016GPL028-E/M6](#) [PIC32MM0032GPL028-E/M6](#) [MB91F526KSEPMC-GSE1](#)  
[PIC32MM0064GPL028-E/SP](#) [PIC32MM0032GPL036-E/M2](#) [TLE9872QTW40XUMA1](#) [FT902L-T](#) [R5F564MLCDFB#31](#)  
[R5F523E5ADFL#30](#) [R5F524TAADFF#31](#) [MCF51AC256ACPUE](#) [PIC32MM0064GPL028-I/ML](#) [PIC32MM0064GPL028-I/SP](#)  
[PIC32MM0064GPL028-I/SO](#) [PIC32MX120F032D-I/TL](#) [PIC32MX130F064D-I/ML](#) [PIC32MZ2064DAB169-I/HF](#) [PIC32MZ2064DAB288-](#)  
[I/4J](#) [ATUC256L4U-AUT](#) [R5F56318CDBG#U0](#) [PIC32MX150F128C-I/TL](#) [PIC32MX130F064C-ITL](#) [PIC32MX230F064D-IML](#)  
[PIC32MM0032GPL028-I/ML](#) [PIC32MX174F256B-I/SO](#) [PIC32MX154F128D-I/PT](#)