

16

H8S/2158 Group, H8S/2158 F-ZTAT™  
Hardware Manual

Renesas 16-Bit Single-Chip Microcomputer  
H8S Family/H8S/2100 Series

H8S/2158      HD64F2158

Hardware Manual

Rev. 3.00  
Revision Date: Jan 25, 2006

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

## Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.  
Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

## Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.  
The information described here may contain technical inaccuracies or typographical errors.  
Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.  
Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (<http://www.renesas.com>).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.  
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.

## General Precautions on Handling of Product

### 1. Treatment of NC Pins

Note: Do not connect anything to the NC pins.

The NC (not connected) pins are either not connected to any of the internal circuitry or are used as test pins or to reduce noise. If something is connected to the NC pins, the operation of the LSI is not guaranteed.

### 2. Treatment of Unused Input Pins

Note: Fix all unused input pins to high or low level.

Generally, the input pins of CMOS products are high-impedance input pins. If unused pins are in their open states, intermediate levels are induced by noise in the vicinity, a pass-through current flows internally, and a malfunction may occur.

### 3. Processing before Initialization

Note: When power is first supplied, the product's state is undefined.

The states of internal circuits are undefined until full power is supplied throughout the chip and a low level is input on the reset pin. During the period where the states are undefined, the register settings and the output state of each pin are also undefined. Design your system so that it does not malfunction because of processing while it is in this undefined state. For those products which have a reset function, reset the LSI immediately after the power supply has been turned on.

### 4. Prohibition of Access to Undefined or Reserved Addresses

Note: Access to undefined or reserved addresses is prohibited.

The undefined or reserved addresses may be used to expand functions, or test registers may have been allocated to these addresses. Do not access these registers; the system's operation is not guaranteed if they are accessed.

# Configuration of This Manual

This manual comprises the following items:

1. General Precautions on Handling of Product
2. Configuration of This Manual
3. Preface
4. Main Revisions in This Edition

The list of revisions is a summary of points that have been revised or added to earlier versions. This does not include all of the revised contents. For details, see the actual locations in this manual.

5. Contents
6. Overview
7. Description of Functional Modules
  - CPU and System-Control Modules
  - On-Chip Peripheral Modules

The configuration of the functional description of each module differs according to the module. However, the generic style includes the following items:

- i) Feature
- ii) Input/Output Pin
- iii) Register Description
- iv) Operation
- v) Usage Note

When designing an application system that includes this LSI, take notes into account. Each section includes notes in relation to the descriptions given, and usage notes are given, as required, as the final part of each section.

8. List of Registers
9. Electrical Characteristics
10. Appendix
11. Index

# Preface

The H8S/2158 is a microcomputer made up of the H8S/2000 CPU employing Renesas Technology's original architecture as its core, and the peripheral functions required to configure a system, such as a notebook PC and portable information appliance products.

The H8S/2000 CPU has an internal 32-bit configuration, sixteen 16-bit general registers, and a simple and optimized instruction set for high-speed operation. The H8S/2600 CPU can handle a 16-Mbyte linear address space. The instruction set of the H8S/2600 CPU maintains upward compatibility at the object level with the H8/300 CPU and H8/300H CPU. This allows the H8/300, H8/300L, or H8/300H user to easily utilize the H8S/2600 CPU.

This LSI is equipped with ROM, RAM, two PWM timers (PWM and PWMX), a 16-bit free-running timer (FRT), an 8-bit timer (TMR), a watchdog timer (WDT), a serial communication interface (SCI), an I<sup>2</sup>C bus interface (IIC), a D/A converter, an A/D converter, and I/O ports as on-chip peripheral modules required for system configuration.

In particular, this LSI incorporates a universal serial bus interface (USB) and a multimedia card (MultiMediaCard™\*<sup>1</sup>) interface (MCIF) for system configuration using a flash memory card as the recording media. In addition, the serial communication interface (SCI) has a smart card interface function. Auxiliary hardware for encryption operation (DES, GF) conforming to the "Keitaide-Music\*<sup>2</sup>" standard is necessary to protect music copyright. Thus, it is easy to interface to the secure multimedia card (Secure-MultiMediaCard™\*<sup>1</sup>).

Further, a data transfer controller (DTC), and a RAM-FIFO unit (RFU) that can operate FIFOs such as the USB and MCIF together are incorporated as a bus master.

The on-chip ROM is flash memory (F-ZTAT™\*<sup>3</sup>) with a capacity of 256 kbytes. ROM is connected to the CPU via a 16-bit data bus, enabling both byte and word data to be accessed in one state. Instruction fetching has been speeded up, and processing speed increased.

Two operating modes, modes 2 and 3, are provided, and there is a choice of address spaces and modes, single-chip mode and external extended mode. Other unique operating modes, such as writing the boot program to the flash memory, on-chip emulation, and boundary scan, are also available.

Notes: 1. MultiMediaCard™ is a trademark of Infineon Technologies AG.

Secure-MultiMediaCard™ is a multimedia card with a content protection function.

2. Technology standards for systems that deliver digital contents, such as music over mobile phones. These standards were put together by five companies: SANYO Electric Co., Ltd., Fujitsu Limited, Nippon Columbia Co., Ltd., PFU Limited, and Hitachi, Ltd. These standards consist of a security guideline, a protocol standard, a secure

multimedia card standard, and a download and playback system standard.

URL: <http://www.keitaide-music.org/>

3. F-ZTAT™ is a trademark of Renesas Technology Corp.

**Target Users:** This manual was written for users who will be using the H8S/2158 in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logical circuits, and microcomputers.

**Objective:** This manual was written to explain the hardware functions and electrical characteristics of the H8S/2158 to the target users.  
Refer to the H8S/2600 Series, H8S/2000 Series Programming Manual for a detailed description of the instruction set.

Notes on reading this manual:

- In order to understand the overall functions of the chip  
Read the manual according to the contents. This manual can be roughly categorized into parts on the CPU, system control functions, peripheral functions and electrical characteristics.
- In order to understand the details of the CPU's functions  
Read the H8S/2600 Series, H8S/2000 Series Programming Manual.
- In order to understand the details of a register when its name is known  
Read the index that is the final part of the manual to find the page number of the entry on the register. The addresses, bits, and initial values of the registers are summarized in section 28, List of Registers.

**Rules:**            **Register name:**            The following notation is used for cases when the same or a similar function, e.g. serial communication, is implemented on more than one channel:

XXX\_N (XXX is the register name and N is the channel number)

**Bit order:**                            The MSB is on the left and the LSB is on the right.

**Number notation:**            Binary is B'xxxx, hexadecimal is H'xxxx, decimal is xxxx.

**Signal notation:**            An overbar is added to a low-active signal:  $\overline{\text{xxxx}}$

**Related Manuals:**    The latest versions of all related manuals are available from our web site. Please ensure you have the latest versions of all documents you require.  
<http://www.renesas.com/>

H8S/2158 manuals:

<b>Document Title</b>	<b>Document No.</b>
H8S/2158 Hardware Manual	This manual
H8S/2600 Series, H8S/2000 Series Programming Manual	REJ09B0139

User's manuals for development tools:

<b>Document Title</b>	<b>Document No.</b>
H8S, H8/300 Series C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual	REJ10B0058
H8S, H8/300 Series Simulator/Debugger User's Manual	ADE-702-037
H8S, H8/300 Series High-performance Embedded Workshop, High-performance Debugging Interface Tutorial	ADE-702-231
High-performance Embedded Workshop User's Manual	ADE-702-201

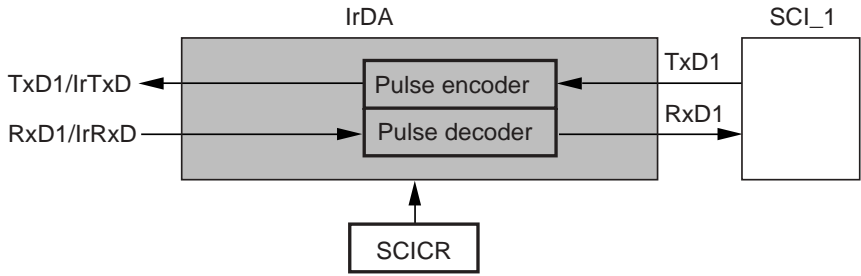




# Main Revisions in This Edition

Item	Page	Revision (See Manual for Details)															
All	—	<p>All references to Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names changed to Renesas Technology Corp. Designation for categories changed from “series” to “group”</p> <hr/> <p>Package</p> <p>TQFP (TFP-100B) deleted</p> <p>(Before) TFBGA (TBP-112) → (After) TFBGA (TBP-112A)</p>															
5.3.4 IRQ Sense Control Registers (ISCR16H, ISCR16L, ISCRH, ISCR)	79	<p>Description added</p> <p>The ISCR registers ... or pins <math>\overline{\text{ExIRQ15}}</math> to <math>\overline{\text{ExIRQ2}}</math>. Switching between pins <math>\overline{\text{IRQ15}}</math> to <math>\overline{\text{IRQ2}}</math> and pins <math>\overline{\text{ExIRQ15}}</math> to <math>\overline{\text{ExIRQ2}}</math> is performed by means of IRQ sense port select register 16 (ISSR16) and the IRQ sense port select register (ISSR).</p>															
6.3.1 Bus Control Register (BCR)	107	<p>Bit table amended</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>Bit Name</th> <th>Initial Value</th> <th>R/W</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>IOS1</td> <td>1</td> <td>R/W</td> <td>IOS Select 1, 0</td> </tr> <tr> <td>0</td> <td>IOS0</td> <td>1</td> <td>R/W</td> <td>Select the address range where the <math>\overline{\text{IOS}}</math> signal is output. For details, refer to table 6.8.</td> </tr> </tbody> </table>	Bit	Bit Name	Initial Value	R/W	Description	1	IOS1	1	R/W	IOS Select 1, 0	0	IOS0	1	R/W	Select the address range where the $\overline{\text{IOS}}$ signal is output. For details, refer to table 6.8.
Bit	Bit Name	Initial Value	R/W	Description													
1	IOS1	1	R/W	IOS Select 1, 0													
0	IOS0	1	R/W	Select the address range where the $\overline{\text{IOS}}$ signal is output. For details, refer to table 6.8.													
8.2.15 Data Transfer ID Read/Write Select Register B (DTIDSRB)	179	<p>Bit table amended</p> <p>0: RAM → Peripheral modules (write)</p> <p>1: Peripheral modules (read) → RAM</p>															
9.9.2 Port 9 Data Register (P9DR)	251	<p>Table amended</p> <p>(Before) R/W → (After) R</p>															
13.1 Features	315	<p>Description amended</p> <ul style="list-style-type: none"> <li>Cascading of two channels <ul style="list-style-type: none"> <li>Cascading of TMR_0 and TMR_1</li> </ul> </li> </ul> <p>... TMR_1 can be used to count TMR_0 compare-match occurrences (compare-match count mode).</p> <ul style="list-style-type: none"> <li>Multiple interrupt sources for each channel ...</li> </ul>															

Item	Page	Revision (See Manual for Details)
13.3.4 Time Control Register (TCR) Table 13.2 Clock Input to TCNT and Count Condition	322	Table 13.2 amended TMR_Y when (CKS2, CKS1, CKS0) = (1, 0, 0) (Before) Increments at overflow signal from TCNT_X* → (After) Setting prohibited TMR_X when (CKS2, CKS1, CKS0) = (1, 0, 0) (Before) Increments at compare-match A from TCNT_Y → (After) Setting prohibited Note * amended Note: * If the TMR_0 clock input is ... , a count-up clock cannot be generated. Simultaneous setting of this condition should therefore be avoided.
—	—	Description of “TMR_Y and TMR_X Cascaded Connection” deleted
13.7 Input Capture Operation	336	Section number amended
13.9.6 Mode Setting with Cascaded Connection	344	Description amended If the 16-bit count mode ... , the input clock pulses for TCNT_0 and TCNT_1 are not generated,
15.3 Register Descriptions	378	• TCSR_1 Notes amended R/(W)*1 [Setting conditions] ... • When TCSR is read when OVF = 1*2, then 0 is written to OVF ...
16.3.7 Serial Status Register (SSR)	402	Description amended Bit Functions in Smart card Interface Mode (when SMIF in SCMR = 1) Bit 6 [Clearing conditions] ... • When RFU is activated by RDRF = 1 allowing data to be read from RDR (only for SCI_0 and SCI_2)

Item	Page	Revision (See Manual for Details)															
16.3.9 Bit Rate Register (BRR) Table 16.2 Relationship between N Setting in BRR and Bit Rate B	405	Table 16.2 amended <table border="1"> <thead> <tr> <th>Mode</th> <th>Bit Rate</th> <th>Error</th> </tr> </thead> <tbody> <tr> <td>Smart card interface mode</td> <td><math>B = \frac{\phi \times 10^6}{S \times 2^{2n+1} \times (N + 1)}</math></td> <td><math>\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N + 1)} - 1 \right\} \times 100</math></td> </tr> </tbody> </table>	Mode	Bit Rate	Error	Smart card interface mode	$B = \frac{\phi \times 10^6}{S \times 2^{2n+1} \times (N + 1)}$	$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N + 1)} - 1 \right\} \times 100$									
Mode	Bit Rate	Error															
Smart card interface mode	$B = \frac{\phi \times 10^6}{S \times 2^{2n+1} \times (N + 1)}$	$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N + 1)} - 1 \right\} \times 100$															
16.3.10 Serial Interface Control Register (SCICR)	412	Table amended <table border="1"> <thead> <tr> <th>Bit</th> <th>Bit Name</th> <th>Initial Value</th> <th>R/W</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>3, 2</td> <td>—</td> <td>All 0</td> <td>R/W</td> <td>Reserved The initial value should not be changed.</td> </tr> <tr> <td>1, 0</td> <td>—</td> <td>All 0</td> <td>R</td> <td>Reserved These bits are always read as 0 and cannot be modified.</td> </tr> </tbody> </table>	Bit	Bit Name	Initial Value	R/W	Description	3, 2	—	All 0	R/W	Reserved The initial value should not be changed.	1, 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
Bit	Bit Name	Initial Value	R/W	Description													
3, 2	—	All 0	R/W	Reserved The initial value should not be changed.													
1, 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.													
16.7.8 Clock Output Control	455	Description amended At Transition from Smart Card Interface Mode to Software Standby Mode: <ol style="list-style-type: none"> <li>Set the port data register (DR) ...</li> </ol> At Transition from Software Standby Mode to Smart Card Interface Mode: <ol style="list-style-type: none"> <li>Cancel software standby mode. ...</li> </ol>															
16.8 IrDA Operation Figure 16.36 IrDA Block Diagram	456	Figure 16.36 amended  <p>The diagram shows an IrDA block containing a Pulse encoder and a Pulse decoder. The Pulse encoder receives data from TxD1 and outputs IrTxD. The Pulse decoder receives IrRxD and outputs RxD1. The IrDA block is connected to SCI_1, which has TxD1 and RxD1 pins. The SCICR register is connected to the IrDA block via an arrow pointing to the Pulse encoder.</p>															
	457	Description amended Transmission: ... The high-level pulse can be selected using the IrCKS2 to IrCKS0 bits in SCICR. Reception: ... IR frames are converted to UART frames using the IrDA interface before inputting to SCI_1. Data of level 0 is ...															

Item	Page	Revision (See Manual for Details)
17.3.8 IIC Operation Reservation Adapter Status Register A (ICSRA)	498	Bit table amended ACKXE (Before) R/W → (After) R
17.3.10 IIC Operation Reservation Adapter Status Register C (ICSRC)	506	Description amended Bit 0 [Clearing conditions] • When ICDRX is read from with no receive data in the shift register (SDRF = 0) in receive mode ...
Figure 17.3 State Transitions of TDRE, SDRF, and RDRF Bits	507	Figure 17.3 amended (b) Receive mode (Before) TDRE → (After) SDRE (Before) SDRF → (After) RDRF
17.5.3 Master Receive Operation	520	Description amended 9. Clear the IRIC flag in ICCR to cancel wait state. The master device outputs the 9th clock and drives SDA low at 9th receive clock pulse ...

IICX1, t <sub>cy</sub> IICX0 Indication	IICX0	Indication	Time Indication[ns]					
			I <sup>2</sup> C Bus Specification (Max.)	φ = 5 MHz	φ = 8 MHz	φ = 10 MHz	φ = 16 MHz	φ = 20 MHz
1	17.5 t <sub>cy</sub>	Standard mode	1000	1000	1000	1000	875	700

Description added

15. Notes on WAIT function

(a) Conditions to cause this phenomenon

When both of the following conditions are satisfied, the clock pulse of the 9th clock could be outputted continuously in master mode using the WAIT function due to the failure of the WAIT insertion after the 8th clock fall.

(1) Setting the WAIT bit of the ICMR register to 1 and operating WAIT, in master mode

(2) If the IRIC bit of interrupt flag is cleared from 1 to 0 between the fall of the 7th clock and the fall of the 8th clock.

(b) Error phenomenon

Normally, WAIT State will be cancelled by clearing the IRIC flag bit from 1 to 0 after the fall of the 8th clock in WAIT State. In this case, if the IRIC flag bit is cleared between the 7th clock fall and the 8th clock fall, the IRIC flag clear- data will be retained internally. Therefore, the WAIT State will be cancelled right after WAIT insertion on 8th clock fall.

(c) Restrictions

Please clear the IRIC flag before the rise of the 7th clock (the counter value of BC2 through BC0 should be 2 or greater), after the IRIC flag is set to 1 on the rise of the 9th clock.

If the IRIC flag-clear is delayed due to the interrupt or other processes and the value of BC counter is turned to 1 or 0, please confirm the SCL pins are in L' state after the counter value of BC2 through BC0 is turned to 0, and clear the IRIC flag. (See figure 17.28.)

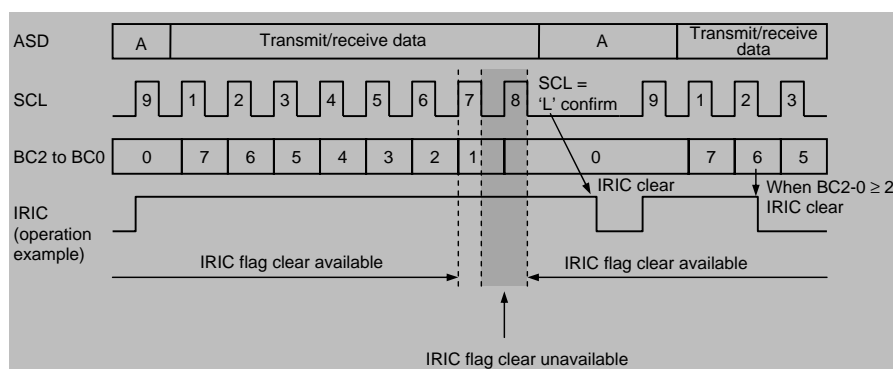


Figure 17.28 IRIC Flag Clear Timing on WAIT Operation

## 16. Notes on Arbitration Lost

The I<sup>2</sup>C bus interface recognizes the data in transmit/receive frame as an address when arbitration is lost in master mode and a transition to slave receive mode is automatically carried out.

When arbitration is lost not in the first frame but in the second frame or subsequent frame, transmit/receive data that is not an address is compared with the value set in the SAR or SARX register as an address. If the receive data matches with the address in the SAR or SARX register, the I<sup>2</sup>C bus interface erroneously recognizes that the address call has occurred. (See figure 17.29.)

In multi-master mode, a bus conflict could happen. When The I<sup>2</sup>C bus interface is operated in master mode, check the state of the AL bit in the ICSR register every time after one frame of data has been transmitted or received.

When arbitration is lost during transmitting the second frame or subsequent frame, take avoidance measures.

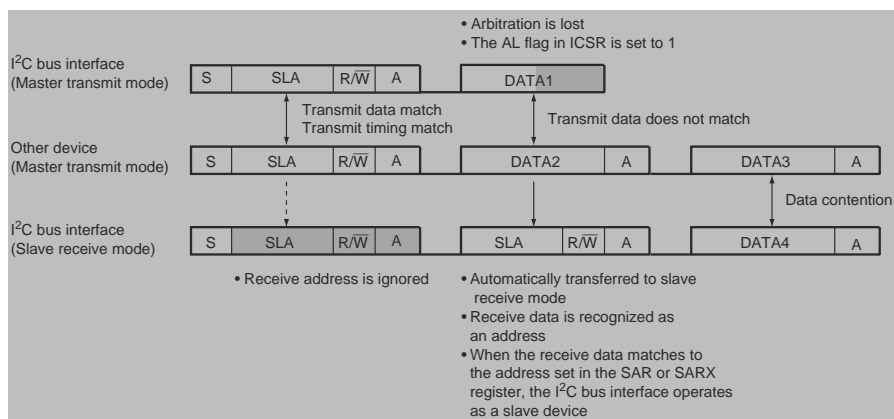


Figure 17.29 Diagram of Erroneous Operation when Arbitration is Lost

Though it is prohibited in the normal I<sup>2</sup>C protocol, the same problem may occur when the MST bit is erroneously set to 1 and a transition to master mode is occurred during data transmission or reception in slave mode. In multi-master mode, pay attention to the setting of the MST bit when a bus conflict may occur. In this case, the MST bit in the ICCR register should be set to 1 according to the order below.

(a) Make sure that the BBSY flag in the ICCR register is 0 and the bus is free before setting the MST bit.

Item	Page	Revision (See Manual for Details)																																								
17.7 Usage Notes	551	<p>(b) Set the MST bit to 1.</p> <p>(c) To confirm that the bus was not entered to the busy state while the MST bit is being set, check that the BBSY flag in the ICCR register is 0 immediately after the MST bit has been set.</p> <p>Note: Above restriction can be cleared by setting bits FNC1 and FNC0 in the ICXR register.</p>																																								
18.3.1 USB Data FIFO	557	Table 18.2 amended																																								
Table 18.2 FIFO Configuration		<table border="1"> <thead> <tr> <th>Endpoint</th> <th>Transfer Direction</th> <th>FIFO Size</th> <th>Configuration</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Endpoint 4 EP4</td> <td>IN</td> <td>Max. 2048 bytes</td> <td>Max. 64 bytes x 32</td> <td>RAM-FIFO (RFU)</td> </tr> <tr> <td>Endpoint 5 EP5</td> <td>OUT</td> <td>Max. 2048 bytes</td> <td>Max. 64 bytes x 32</td> <td></td> </tr> </tbody> </table>	Endpoint	Transfer Direction	FIFO Size	Configuration	Description	Endpoint 4 EP4	IN	Max. 2048 bytes	Max. 64 bytes x 32	RAM-FIFO (RFU)	Endpoint 5 EP5	OUT	Max. 2048 bytes	Max. 64 bytes x 32																										
Endpoint	Transfer Direction	FIFO Size	Configuration	Description																																						
Endpoint 4 EP4	IN	Max. 2048 bytes	Max. 64 bytes x 32	RAM-FIFO (RFU)																																						
Endpoint 5 EP5	OUT	Max. 2048 bytes	Max. 64 bytes x 32																																							
25.4.1 TAP Controller State Transitions	755	Figure 25.2 replaced																																								
Figure 25.2 TAP Controller State Transitions																																										
28.1 Register Addresses (Address Order)	792	Table amended																																								
		<table border="1"> <thead> <tr> <th>Register Name</th> <th>Abbreviation</th> <th>Number of Bits</th> <th>Address</th> <th>Module</th> <th>Data Bus Width</th> <th>Number of Access States</th> </tr> </thead> <tbody> <tr> <td>Command register 5</td> <td>CMDR5</td> <td>8</td> <td>H'FBC5</td> <td>MCIF</td> <td>8</td> <td>3</td> </tr> <tr> <td>Command start register</td> <td>CMDSTRT</td> <td>8</td> <td>H'FBC6</td> <td>MCIF</td> <td>8</td> <td>3</td> </tr> <tr> <td>Operation control register</td> <td>OPCR</td> <td>8</td> <td>H'FBCA</td> <td>MCIF</td> <td>8</td> <td>3</td> </tr> </tbody> </table>	Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States	Command register 5	CMDR5	8	H'FBC5	MCIF	8	3	Command start register	CMDSTRT	8	H'FBC6	MCIF	8	3	Operation control register	OPCR	8	H'FBCA	MCIF	8	3												
Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States																																				
Command register 5	CMDR5	8	H'FBC5	MCIF	8	3																																				
Command start register	CMDSTRT	8	H'FBC6	MCIF	8	3																																				
Operation control register	OPCR	8	H'FBCA	MCIF	8	3																																				
	793	Table amended																																								
		<table border="1"> <thead> <tr> <th>Register Name</th> <th>Abbreviation</th> <th>Number of Bits</th> <th>Address</th> <th>Module</th> <th>Data Bus Width</th> <th>Number of Access States</th> </tr> </thead> <tbody> <tr> <td>Response register 16</td> <td>RSPR16</td> <td>8</td> <td>H'FBF0</td> <td>MCIF</td> <td>8</td> <td>3</td> </tr> <tr> <td>Response register D</td> <td>RSPRD</td> <td>8</td> <td>H'FBF1</td> <td>MCIF</td> <td>8</td> <td>3</td> </tr> <tr> <td>Data timeout register H</td> <td>DTOUTRH</td> <td>8</td> <td>H'FBF2</td> <td>MCIF</td> <td>8</td> <td>3</td> </tr> </tbody> </table>	Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States	Response register 16	RSPR16	8	H'FBF0	MCIF	8	3	Response register D	RSPRD	8	H'FBF1	MCIF	8	3	Data timeout register H	DTOUTRH	8	H'FBF2	MCIF	8	3												
Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States																																				
Response register 16	RSPR16	8	H'FBF0	MCIF	8	3																																				
Response register D	RSPRD	8	H'FBF1	MCIF	8	3																																				
Data timeout register H	DTOUTRH	8	H'FBF2	MCIF	8	3																																				
28.1 Register Bits	804	Table amended																																								
		<table border="1"> <thead> <tr> <th>Register Abbreviation</th> <th>Bit 7</th> <th>Bit 6</th> <th>Bit 5</th> <th>Bit 4</th> <th>Bit 3</th> <th>Bit 2</th> <th>Bit 1</th> <th>Bit 0</th> <th>Module</th> </tr> </thead> <tbody> <tr> <td>CMDR5</td> <td>CRC</td> <td>CRC</td> <td>CRC</td> <td>CRC</td> <td>CRC</td> <td>CRC</td> <td>CRC</td> <td>End</td> <td>MCIF</td> </tr> <tr> <td>CMDSTRT</td> <td>—</td> <td>—</td> <td>—</td> <td>—</td> <td>—</td> <td>—</td> <td>—</td> <td>START</td> <td></td> </tr> <tr> <td>OPCR</td> <td>CMDOFF</td> <td>—</td> <td>RD_CONTI</td> <td>DATAEN</td> <td>—</td> <td>—</td> <td>—</td> <td>—</td> <td></td> </tr> </tbody> </table>	Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	CMDR5	CRC	CRC	CRC	CRC	CRC	CRC	CRC	End	MCIF	CMDSTRT	—	—	—	—	—	—	—	START		OPCR	CMDOFF	—	RD_CONTI	DATAEN	—	—	—	—	
Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module																																	
CMDR5	CRC	CRC	CRC	CRC	CRC	CRC	CRC	End	MCIF																																	
CMDSTRT	—	—	—	—	—	—	—	START																																		
OPCR	CMDOFF	—	RD_CONTI	DATAEN	—	—	—	—																																		
	805	Table amended																																								
		<table border="1"> <thead> <tr> <th>Register Abbreviation</th> <th>Bit 7</th> <th>Bit 6</th> <th>Bit 5</th> <th>Bit 4</th> <th>Bit 3</th> <th>Bit 2</th> <th>Bit 1</th> <th>Bit 0</th> <th>Module</th> </tr> </thead> <tbody> <tr> <td>RSPR16</td> <td>Bit 7</td> <td>Bit 6</td> <td>Bit 5</td> <td>Bit 4</td> <td>Bit 3</td> <td>Bit 2</td> <td>Bit 1</td> <td>Bit 0</td> <td>MCIF</td> </tr> <tr> <td>PSPRD</td> <td>Bit 7</td> <td>Bit 6</td> <td>Bit 5</td> <td>Bit 4</td> <td>Bit 3</td> <td>Bit 2</td> <td>Bit 1</td> <td>Bit 0</td> <td></td> </tr> <tr> <td>DTOUTRH</td> <td>DTOUT15</td> <td>DTOUT14</td> <td>DTOUT13</td> <td>DTOUT12</td> <td>DTOUT11</td> <td>DTOUT10</td> <td>DTOUT9</td> <td>DTOUT8</td> <td></td> </tr> </tbody> </table>	Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	RSPR16	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	MCIF	PSPRD	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		DTOUTRH	DTOUT15	DTOUT14	DTOUT13	DTOUT12	DTOUT11	DTOUT10	DTOUT9	DTOUT8	
Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module																																	
RSPR16	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	MCIF																																	
PSPRD	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0																																		
DTOUTRH	DTOUT15	DTOUT14	DTOUT13	DTOUT12	DTOUT11	DTOUT10	DTOUT9	DTOUT8																																		

Item	Page	Revision (See Manual for Details)																																												
28.1 Register Bits	808	Table amended																																												
<table border="1"> <thead> <tr> <th>Register Abbreviation</th> <th>Bit 7</th> <th>Bit 6</th> <th>Bit 5</th> <th>Bit 4</th> <th>Bit 3</th> <th>Bit 2</th> <th>Bit 1</th> <th>Bit 0</th> <th>Module</th> </tr> </thead> <tbody> <tr> <td>KBCOMP</td> <td>—</td> <td>—</td> <td>—</td> <td>SCANE</td> <td>KBADE</td> <td>KBCH2</td> <td>KBCH1</td> <td>KBCH0</td> <td>A/D converter</td> </tr> <tr> <td>SCICR</td> <td>IrE</td> <td>IrCKS2</td> <td>IrCKS1</td> <td>IrCKS0</td> <td>—</td> <td>—</td> <td>—</td> <td>—</td> <td>SCI_1</td> </tr> </tbody> </table>			Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	KBCOMP	—	—	—	SCANE	KBADE	KBCH2	KBCH1	KBCH0	A/D converter	SCICR	IrE	IrCKS2	IrCKS1	IrCKS0	—	—	—	—	SCI_1														
Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module																																					
KBCOMP	—	—	—	SCANE	KBADE	KBCH2	KBCH1	KBCH0	A/D converter																																					
SCICR	IrE	IrCKS2	IrCKS1	IrCKS0	—	—	—	—	SCI_1																																					
28.2 Register Bits	808	Bits 2 and 3 in SCICR description amended (Before) IrTxINV → (After) — (Before) IrRxINV → (After) —																																												
28.3 Register States in Each Operating Mode	815	Table amended																																												
<table border="1"> <thead> <tr> <th>Register Abbreviation</th> <th>Reset</th> <th>High-Speed/ Medium-Speed</th> <th>Watch</th> <th>Sleep</th> <th>Sub-Active</th> <th>Sub-Sleep</th> <th>Module Stop</th> <th>Software Standby</th> <th>Hardware Standby</th> <th>Module</th> </tr> </thead> <tbody> <tr> <td>CMDR5</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>MCIF</td> </tr> <tr> <td>CMDSTRT</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td></td> </tr> <tr> <td>OPCR</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td></td> </tr> </tbody> </table>			Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module	CMDR5	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	MCIF	CMDSTRT	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized		OPCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module																																				
CMDR5	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	MCIF																																				
CMDSTRT	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized																																					
OPCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized																																					
	816	Table amended																																												
<table border="1"> <thead> <tr> <th>Register Abbreviation</th> <th>Reset</th> <th>High-Speed/ Medium-Speed</th> <th>Watch</th> <th>Sleep</th> <th>Sub-Active</th> <th>Sub-Sleep</th> <th>Module Stop</th> <th>Software Standby</th> <th>Hardware Standby</th> <th>Module</th> </tr> </thead> <tbody> <tr> <td>RSPR16</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>MCIF</td> </tr> <tr> <td>PSPRD</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td></td> </tr> <tr> <td>DTOUTRH</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>—</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td>Initialized</td> <td></td> </tr> </tbody> </table>			Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module	RSPR16	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	MCIF	PSPRD	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized		DTOUTRH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module																																				
RSPR16	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	MCIF																																				
PSPRD	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized																																					
DTOUTRH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized																																					
29.6 Flash Memory Characteristics	859	Table 29.18 amended																																												
Table 29.18 Flash Memory Characteristics		<table border="1"> <thead> <tr> <th>Item</th> <th>Symbol</th> <th>Min</th> <th>Typ</th> <th>Max</th> <th>Unit</th> <th>Test Conditions</th> </tr> </thead> <tbody> <tr> <td>Reprogramming count</td> <td>N<sub>WEC</sub></td> <td>100<sup>*8</sup></td> <td>10,000<sup>*9</sup></td> <td>—</td> <td>Times</td> <td></td> </tr> <tr> <td>Data retention time<sup>*10</sup></td> <td>t<sub>DRP</sub></td> <td>10</td> <td>—</td> <td>—</td> <td>Years</td> <td></td> </tr> </tbody> </table>	Item	Symbol	Min	Typ	Max	Unit	Test Conditions	Reprogramming count	N <sub>WEC</sub>	100 <sup>*8</sup>	10,000 <sup>*9</sup>	—	Times		Data retention time <sup>*10</sup>	t <sub>DRP</sub>	10	—	—	Years																								
Item	Symbol	Min	Typ	Max	Unit	Test Conditions																																								
Reprogramming count	N <sub>WEC</sub>	100 <sup>*8</sup>	10,000 <sup>*9</sup>	—	Times																																									
Data retention time <sup>*10</sup>	t <sub>DRP</sub>	10	—	—	Years																																									
	860	Notes *8*9*10 added																																												
<p>Notes: 8. Minimum number of times for which all characteristics are guaranteed after rewriting. (Guarantee range is 1 to minimum value.)</p> <p>9. Reference value for 25°C (as a guide line, rewriting should normally function up to this value).</p> <p>10. Data retention characteristics when rewriting is performed within the specification range, including the minimum value.</p>																																														
B. Product Lineup	863	Product type amended (Before) F2158VBP25 → (After) F2158VBQ25																																												
C. Package Dimensions	—	Figure of TFP-100B deleted																																												
Figure C.1 Package Dimensions (TBP-112A)	864	Figure C.1 replaced																																												



# Contents

Section 1	Overview .....	1
1.1	Features .....	1
1.2	Internal Block Diagram .....	2
1.3	Pin Description .....	3
1.3.1	Pin Arrangement .....	3
1.3.2	Pin Arrangement in Each Operating Mode .....	4
1.3.3	Pin Functions .....	8
Section 2	CPU .....	17
2.1	Features .....	17
2.1.1	Differences between H8S/2600 CPU and H8S/2000 CPU .....	18
2.1.2	Differences from H8/300 CPU .....	19
2.1.3	Differences from H8/300H CPU .....	19
2.2	CPU Operating Modes .....	20
2.2.1	Normal Mode .....	20
2.2.2	Advanced Mode .....	22
2.3	Address Space .....	24
2.4	Register Configuration .....	25
2.4.1	General Registers .....	26
2.4.2	Program Counter (PC) .....	27
2.4.3	Extended Control Register (EXR) .....	27
2.4.4	Condition-Code Register (CCR) .....	28
2.4.5	Initial Register Values .....	29
2.5	Data Formats .....	30
2.5.1	General Register Data Formats .....	30
2.5.2	Memory Data Formats .....	32
2.6	Instruction Set .....	33
2.6.1	Table of Instructions Classified by Function .....	34
2.6.2	Basic Instruction Formats .....	43
2.7	Addressing Modes and Effective Address Calculation .....	44
2.7.1	Register Direct—Rn .....	45
2.7.2	Register Indirect—@ERn .....	45
2.7.3	Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn) .....	45
2.7.4	Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn ..	45
2.7.5	Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32 .....	46
2.7.6	Immediate—#xx:8, #xx:16, or #xx:32 .....	46
2.7.7	Program-Counter Relative—@(d:8, PC) or @(d:16, PC) .....	47
2.7.8	Memory Indirect—@@aa:8 .....	47

2.7.9	Effective Address Calculation.....	48
2.8	Processing States.....	50
2.9	Usage Notes .....	52
2.9.1	Note on TAS Instruction Usage .....	52
2.9.2	Note on Bit Manipulation Instructions.....	52
2.9.3	EPMOV Instruction.....	54
<b>Section 3 MCU Operating Modes .....</b>		<b>55</b>
3.1	Operating Mode Selection.....	55
3.2	Register Descriptions .....	55
3.2.1	Mode Control Register (MDCR) .....	56
3.2.2	System Control Register (SYSCR).....	57
3.2.3	Serial Timer Control Register (STCR) .....	59
3.3	Operating Mode Descriptions .....	60
3.3.1	Mode 2.....	60
3.3.2	Mode 3.....	61
3.3.3	Pin Functions .....	61
3.4	Address Map in Each Operating Mode.....	62
<b>Section 4 Exception Handling .....</b>		<b>65</b>
4.1	Exception Handling Types and Priority .....	65
4.2	Exception Sources and Exception Vector Table .....	66
4.3	Reset.....	67
4.3.1	Reset Exception Handling.....	68
4.3.2	Interrupts after Reset.....	69
4.3.3	On-Chip Peripheral Modules after Reset Is Cancelled .....	69
4.4	Interrupt Exception Handling.....	69
4.5	Trap Instruction Exception Handling.....	69
4.6	Stack Status after Exception Handling.....	70
4.7	Usage Note.....	71
<b>Section 5 Interrupt Controller .....</b>		<b>73</b>
5.1	Features.....	73
5.2	Input/Output Pins .....	75
5.3	Register Descriptions .....	75
5.3.1	Interrupt Control Registers A to D (ICRA to ICRD).....	76
5.3.2	Address Break Control Register (ABRKCR).....	77
5.3.3	Break Address Registers A to C (BARA to BARC).....	78
5.3.4	IRQ Sense Control Registers (ISCR16H, ISCR16L, ISCRH, ISCRL).....	79
5.3.5	IRQ Enable Registers (IER16, IER) .....	81
5.3.6	IRQ Status Registers (ISR16, ISR).....	82

5.3.7	Keyboard Matrix Interrupt Mask Registers (KMIMRA, KMIMR6)	
	Wake-Up Event Interrupt Mask Register (WUEMR3)	83
5.4	Interrupt Sources	84
5.4.1	External Interrupts	84
5.4.2	Internal Interrupts	86
5.5	Interrupt Exception Handling Vector Table	86
5.6	Interrupt Control Modes and Interrupt Operation	90
5.6.1	Interrupt Control Mode 0	92
5.6.2	Interrupt Control Mode 1	94
5.6.3	Interrupt Exception Handling Sequence	96
5.6.4	Interrupt Response Times	98
5.6.5	DTC Activation by Interrupt	99
5.7	Usage Notes	101
5.7.1	Conflict between Interrupt Generation and Disabling	101
5.7.2	Instructions that Disable Interrupts	102
5.7.3	Interrupts during Execution of EEPMOV Instruction	102
<b>Section 6 Bus Controller</b>		<b>103</b>
6.1	Features	103
6.2	Input/Output Pins	105
6.3	Register Descriptions	106
6.3.1	Bus Control Register (BCR)	106
6.3.2	Bus Control Register 2 (BCR2)	108
6.3.3	Wait State Control Register (WSCR)	110
6.3.4	Wait State Control Register 2 (WSCR2)	112
6.4	Bus Control	113
6.4.1	Bus Specifications	113
6.4.2	Advanced Mode	121
6.4.3	Normal Mode	122
6.4.4	I/O Select Signals	122
6.5	Basic Bus Interface	123
6.5.1	Data Size and Data Alignment	123
6.5.2	Valid Strobes	124
6.5.3	Basic Operation Timing	125
6.5.4	Wait Control	133
6.6	Burst ROM Interface	134
6.6.1	Basic Operation Timing	135
6.6.2	Wait Control	136
6.7	Memory Card Interface	137
6.7.1	Data Size and Data Alignment	137
6.7.2	Valid Strobes	138

6.7.3	Basic Operation Timing .....	138
6.7.4	Wait Control .....	140
6.8	Idle Cycle .....	141
6.9	Bus Arbitration.....	142
6.9.1	Bus Master Priority .....	142
6.9.2	Bus Transfer Timing.....	143
<b>Section 7 Data Transfer Controller (DTC).....</b>		<b>145</b>
7.1	Features.....	145
7.2	Register Descriptions .....	146
7.2.1	DTC Mode Register A (MRA) .....	147
7.2.2	DTC Mode Register B (MRB).....	148
7.2.3	DTC Source Address Register (SAR).....	149
7.2.4	DTC Destination Address Register (DAR).....	149
7.2.5	DTC Transfer Count Register A (CRA) .....	149
7.2.6	DTC Transfer Count Register B (CRB).....	149
7.2.7	DTC Enable Registers (DTCER).....	150
7.2.8	DTC Vector Register (DTVECR).....	151
7.3	Activation Sources .....	152
7.4	Location of Register Information and DTC Vector Table .....	153
7.5	Operation.....	156
7.5.1	Normal Mode .....	157
7.5.2	Repeat Mode .....	158
7.5.3	Block Transfer Mode .....	159
7.5.4	Chain Transfer .....	160
7.5.5	Interrupts .....	161
7.5.6	Operation Timing.....	161
7.5.7	Number of DTC Execution States.....	163
7.6	Procedures for Using DTC.....	164
7.6.1	Activation by Interrupt.....	164
7.6.2	Activation by Software .....	164
7.7	Examples of Use of the DTC .....	164
7.7.1	Normal Mode .....	164
7.7.2	Software Activation .....	165
7.8	Usage Notes .....	166
7.8.1	Module Stop Mode Setting .....	166
7.8.2	On-Chip RAM .....	166
7.8.3	DTCE Bit Setting.....	166
7.8.4	Setting Required on Entering Subactive Mode or Watch Mode .....	166
7.8.5	DTC Activation by Interrupt Sources of SCI, IIC, or A/D Converter.....	166
7.8.6	DTC Activation by Interrupt Sources of USB or MCIF .....	166

Section 8	RAM-FIFO Unit (RFU).....	167
8.1	Features.....	167
8.2	Register Descriptions .....	169
8.2.1	FIFO Status/Register/Pointer (FSTR).....	169
8.2.2	Base Address Register (BAR).....	170
8.2.3	Read Address Pointer (RAR).....	170
8.2.4	Write Address Pointer (WAR).....	171
8.2.5	Temporary Pointer (TMP) .....	171
8.2.6	Valid Data Byte Number (DATAN).....	172
8.2.7	Free Area Byte Number (FREEN).....	172
8.2.8	Read Start Address (NRA).....	172
8.2.9	Write Start Address (NWA).....	173
8.2.10	Data Transfer Control Register A (DTCRA) .....	173
8.2.11	Data Transfer Control Register B (DTCRB) .....	175
8.2.12	Data Transfer Status Register C (DTSTRC).....	176
8.2.13	Data Transfer ID Register (DTIDR) .....	178
8.2.14	Data Transfer ID Read/Write Select Register A (DTIDSRA) .....	178
8.2.15	Data Transfer ID Read/Write Select Register B (DTIDSRB).....	179
8.2.16	Data Transfer Status Register A (DTSTRA).....	179
8.2.17	Data Transfer Status Register B (DTSTRB).....	180
8.2.18	Data Transfer Control Register C (DTCRC).....	180
8.2.19	Data Transfer Control Register D (DTCRD) .....	181
8.2.20	Data Transfer Interrupt Enable Register (DTIER).....	181
8.2.21	Data Transfer Register Select Register (DTRSR).....	181
8.3	Activation Source and Priority.....	183
8.4	RAM-FIFO Location .....	184
8.5	RAM-FIFO Pointer .....	184
8.6	RAM-FIFO Manipulation and RFU Bus Cycles.....	184
8.7	RFU Bus Cycle .....	188
8.7.1	Clock Division .....	188
8.7.2	RFU Bus Cycle Insertion .....	189
8.7.3	RFU Response Time .....	189
8.8	Operation.....	191
8.8.1	Transmission/Reception of Single Data Block .....	191
8.8.2	Transmission/Reception of Consecutive Data Blocks .....	192
8.8.3	RFU Manipulation by USB.....	193
8.8.4	RFU Manipulation by SCI.....	197
8.8.5	RFU Manipulation by MCIF.....	200
8.9	Interrupt Sources .....	202
8.10	RFU Initialization .....	203
8.11	Usage Notes .....	204

Section 9	I/O Ports .....	205
9.1	Port 1 .....	209
9.1.1	Port 1 Data Direction Register (P1DDR) .....	209
9.1.2	Port 1 Data Register (P1DR) .....	210
9.1.3	Port 1 Pull-Up MOS Control Register (P1PCR) .....	210
9.1.4	Pin Functions .....	211
9.1.5	Port 1 Input Pull-Up MOS .....	211
9.2	Port 2 .....	212
9.2.1	Port 2 Data Direction Register (P2DDR) .....	212
9.2.2	Port 2 Data Register (P2DR) .....	213
9.2.3	Port 2 Pull-Up MOS Control Register (P2PCR) .....	213
9.2.4	Pin Functions .....	214
9.2.5	Port 2 Input Pull-Up MOS .....	215
9.3	Port 3 .....	215
9.3.1	Port 3 Data Direction Register (P3DDR) .....	216
9.3.2	Port 3 Data Register (P3DR) .....	216
9.3.3	Port 3 Pull-Up MOS Control Register (P3PCR) .....	217
9.3.4	Pin Functions .....	217
9.3.5	Port 3 Input Pull-Up MOS .....	222
9.4	Port 4 .....	222
9.4.1	Port 4 Data Direction Register (P4DDR) .....	223
9.4.2	Port 4 Data Register (P4DR) .....	223
9.4.3	Pin Functions .....	224
9.5	Port 5 .....	228
9.5.1	Port 5 Data Direction Register (P5DDR) .....	228
9.5.2	Port 5 Data Register (P5DR) .....	229
9.5.3	Pin Functions .....	229
9.6	Port 6 .....	232
9.6.1	Port 6 Data Direction Register (P6DDR) .....	233
9.6.2	Port 6 Data Register (P6DR) .....	233
9.6.3	Port 6 Pull-Up MOS Control Register (KMPCR6) .....	234
9.6.4	System Control Register 2 (SYSCR2) .....	235
9.6.5	Pin Functions .....	236
9.6.6	Port 6 Input Pull-Up MOS .....	244
9.7	Port 7 .....	244
9.7.1	Port 7 Input Data Register (P7PIN) .....	244
9.8	Port 8 .....	245
9.8.1	Port 8 Data Direction Register (P8DDR) .....	245
9.8.2	Port 8 Data Register (P8DR) .....	246
9.8.3	Pin Functions .....	246
9.9	Port 9 .....	250

9.9.1	Port 9 Data Direction Register (P9DDR)	250
9.9.2	Port 9 Data Register (P9DR)	251
9.9.3	Pin Functions	251
9.10	Port A	254
9.10.1	Port A Data Direction Register (PADDR)	254
9.10.2	Port A Output Data Register (PAODR)	254
9.10.3	Port A Input Data Register (PAPIN)	255
9.10.4	Pin Functions	255
9.10.5	Input Pull-Up MOS	257
9.11	Change of Peripheral Function Pins	258
9.11.1	IRQ Sense Port Select Register 16 (ISSR16), IRQ Sense Port Select Register (ISSR)	258
9.11.2	Port Control Register 0 (PTCNT0)	260
<b>Section 10 8-Bit PWM Timer (PWM)</b>		<b>261</b>
10.1	Features	261
10.2	Input/Output Pins	263
10.3	Register Descriptions	263
10.3.1	PWM Register Select (PWSL)	264
10.3.2	PWM Data Registers (PWDR0 to PWDR15)	266
10.3.3	PWM Data Polarity Registers A and B (PWDPRA and PWDPRB)	266
10.3.4	PWM Output Enable Registers A and B (PWOERA and PWOERB)	267
10.3.5	Peripheral Clock Select Register (PCSR)	269
10.4	Operation	270
<b>Section 11 14-Bit PWM Timer (PWMX)</b>		<b>273</b>
11.1	Features	273
11.2	Input/Output Pins	274
11.3	Register Descriptions	274
11.3.1	PWM D/A Counter H, L (DACNTH, DACNTL)	275
11.3.2	PWM (D/A) Data Registers A and B (DADRA and DADRB)	276
11.3.3	PWM (D/A) Control Register (DACR)	277
11.3.4	Peripheral Clock Select Register (PCSR)	279
11.4	Bus Master Interface	280
11.5	Operation	281
<b>Section 12 16-Bit Free-Running Timer (FRT)</b>		<b>287</b>
12.1	Features	287
12.2	Input/Output Pins	289
12.3	Register Descriptions	289
12.3.1	Free-Running Counter (FRC)	290

12.3.2	Output Compare Registers A and B (OCRA and OCRB).....	290
12.3.3	Input Capture Registers A to D (ICRA to ICRD) .....	290
12.3.4	Output Compare Registers AR and AF (OCRAR and OCRAF) .....	291
12.3.5	Output Compare Register DM (OCRDM).....	291
12.3.6	Timer Interrupt Enable Register (TIER).....	292
12.3.7	Timer Control/Status Register (TCSR).....	293
12.3.8	Timer Control Register (TCR).....	296
12.3.9	Timer Output Compare Control Register (TOCR) .....	297
12.4	Operation.....	299
12.4.1	Pulse Output.....	299
12.5	Operation Timing.....	300
12.5.1	FRC Increment Timing.....	300
12.5.2	Output Compare Output Timing .....	301
12.5.3	FRC Clear Timing.....	301
12.5.4	Input Capture Input Timing .....	302
12.5.5	Buffered Input Capture Input Timing .....	303
12.5.6	Timing of Input Capture Flag (ICF) Setting .....	304
12.5.7	Timing of Output Compare Flag (OCF) setting.....	305
12.5.8	Timing of FRC Overflow Flag Setting .....	305
12.5.9	Automatic Addition Timing.....	306
12.5.10	Mask Signal Generation Timing .....	307
12.6	Interrupt Sources .....	308
12.7	Usage Notes .....	309
12.7.1	Conflict between FRC Write and Clear .....	309
12.7.2	Conflict between FRC Write and Increment.....	310
12.7.3	Conflict between OCR Write and Compare-Match .....	311
12.7.4	Switching of Internal Clock and FRC Operation .....	312
<b>Section 13</b>	<b>8-Bit Timer (TMR).....</b>	<b>315</b>
13.1	Features.....	315
13.2	Input/Output Pins .....	318
13.3	Register Descriptions .....	318
13.3.1	Timer Counter (TCNT).....	320
13.3.2	Time Constant Register A (TCORA).....	320
13.3.3	Time Constant Register B (TCORB) .....	320
13.3.4	Timer Control Register (TCR).....	321
13.3.5	Timer Control/Status Register (TCSR).....	323
13.3.6	Input Capture Register (TICR) .....	328
13.3.7	Time Constant Register (TCORC).....	328
13.3.8	Input Capture Registers R and F (TICRR and TICRF).....	329
13.3.9	Timer Input Select Register (TISR).....	329



13.4	Operation.....	330
13.4.1	Pulse Output.....	330
13.5	Operation Timing.....	331
13.5.1	TCNT Count Timing.....	331
13.5.2	Timing of CMFA and CMFB Setting at Compare-Match .....	332
13.5.3	Timing of Timer Output at Compare-Match.....	332
13.5.4	Timing of Counter Clear at Compare-Match .....	333
13.5.5	TCNT External Reset Timing .....	333
13.5.6	Timing of Overflow Flag (OVF) Setting .....	334
13.6	TMR_0 and TMR_1 Cascaded Connection.....	335
13.6.1	16-Bit Count Mode .....	335
13.6.2	Compare-Match Count Mode .....	335
13.7	Input Capture Operation.....	336
13.8	Interrupt Sources .....	338
13.9	Usage Notes .....	339
13.9.1	Conflict between TCNT Write and Clear .....	339
13.9.2	Conflict between TCNT Write and Increment.....	340
13.9.3	Conflict between TCOR Write and Compare-Match.....	341
13.9.4	Conflict between Compare-Matches A and B.....	342
13.9.5	Switching of Internal Clocks and TCNT Operation.....	342
13.9.6	Mode Setting with Cascaded Connection .....	344
<b>Section 14 Timer Connection.....</b>		<b>345</b>
14.1	Features .....	345
14.2	Input/Output Pins .....	347
14.3	Register Descriptions .....	347
14.3.1	Timer Connection Register I (TCONRI) .....	348
14.3.2	Timer Connection Register O (TCONRO) .....	352
14.3.3	Timer Connection Register S (TCONRS).....	354
14.3.4	Edge Sense Register (SEDGR).....	356
14.4	Operation.....	358
14.4.1	PWM Decoding (PDC Signal Generation) .....	358
14.4.2	Clamp Waveform Generation (CL1/CL2/CL3 Signal Generation) .....	359
14.4.3	8-Bit Timer Divided Waveform Period Measurement.....	361
14.4.4	IHI Signal and 2fH Modification .....	363
14.4.5	IVI Signal Fall Modification and IHI Synchronization.....	365
14.4.6	Internal Synchronization Signal Generation (IHG/IVG/CL4 Signal Generation) .....	367
14.4.7	HSYNCO Output.....	370
14.4.8	VSYNCO Output.....	371
14.4.9	CBLANK Output.....	372

Section 15	Watchdog Timer (WDT)	373
15.1	Features	373
15.2	Input/Output Pins	375
15.3	Register Descriptions	375
15.3.1	Timer Counter (TCNT)	375
15.3.2	Timer Control/Status Register (TCSR)	376
15.4	Operation	379
15.4.1	Watchdog Timer Mode	379
15.4.2	Interval Timer Mode	381
15.4.3	RESO Signal Output Timing	382
15.5	Interrupt Sources	382
15.6	Usage Notes	383
15.6.1	Notes on Register Access	383
15.6.2	Conflict between Timer Counter (TCNT) Write and Increment	384
15.6.3	Changing Values of CKS2 to CKS0 Bits	384
15.6.4	Switching between Watchdog Timer Mode and Interval Timer Mode	384
15.6.5	System Reset by RESO Signal	385
15.6.6	Counter Values during Transitions between High-Speed, Sub-Active, and Watch Modes	385
Section 16	Serial Communication Interface (SCI, IrDA, and CRC)	387
16.1	Features	387
16.2	Input/Output Pins	391
16.3	Register Descriptions	391
16.3.1	Receive Shift Register (RSR)	392
16.3.2	Receive Data Register (RDR)	392
16.3.3	Transmit Data Register (TDR)	392
16.3.4	Transmit Shift Register (TSR)	392
16.3.5	Serial Mode Register (SMR)	393
16.3.6	Serial Control Register (SCR)	396
16.3.7	Serial Status Register (SSR)	398
16.3.8	Smart Card Mode Register (SCMR)	404
16.3.9	Bit Rate Register (BRR)	405
16.3.10	Serial Interface Control Register (SCICR)	412
16.3.11	Serial Enhanced Mode Register_0 and 2 (SEMR_0 and SEMR_2)	412
16.3.12	Serial RFU Enable Register_0 and 2 (SCIDTER_0 and SCIDTER_2)	416
16.4	Operation in Asynchronous Mode	417
16.4.1	Data Transfer Format	418
16.4.2	Receive Data Sampling Timing and Reception Margin in Asynchronous Mode	419
16.4.3	Clock	420
16.4.4	Serial Enhanced Mode Clock	421

16.4.5	SCI Initialization (Asynchronous Mode).....	424
16.4.6	Serial Data Transmission (Asynchronous Mode) .....	425
16.4.7	Serial Data Reception (Asynchronous Mode).....	427
16.5	Multiprocessor Communication Function.....	431
16.5.1	Multiprocessor Serial Data Transmission .....	432
16.5.2	Multiprocessor Serial Data Reception .....	433
16.6	Operation in Clocked Synchronous Mode .....	437
16.6.1	Clock.....	437
16.6.2	SCI Initialization (Synchronous).....	437
16.6.3	Serial Data Transmission (Clocked Synchronous Mode) .....	438
16.6.4	Serial Data Reception (Clocked Synchronous Mode).....	441
16.6.5	Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode) .....	443
16.6.6	SCI Selection in Serial Enhanced Mode .....	443
16.7	Smart Card Interface Description.....	445
16.7.1	Sample Connection .....	445
16.7.2	Data Format (Except in Block Transfer Mode) .....	446
16.7.3	Block Transfer Mode .....	447
16.7.4	Receive Data Sampling Timing and Reception Margin.....	447
16.7.5	Initialization .....	449
16.7.6	Serial Data Transmission (Except in Block Transfer Mode) .....	450
16.7.7	Serial Data Reception (Except in Block Transfer Mode).....	453
16.7.8	Clock Output Control.....	454
16.8	IrDA Operation .....	456
16.9	Interrupt Sources .....	459
16.9.1	Interrupts in Normal Serial Communication Interface Mode.....	459
16.9.2	Interrupts in Smart Card Interface Mode .....	460
16.10	Usage Notes .....	461
16.10.1	Module Stop Mode Setting .....	461
16.10.2	Break Detection and Processing.....	461
16.10.3	Mark State and Break Detection .....	462
16.10.4	Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only).....	462
16.10.5	Relation between Writing to TDR and TDRE Flag .....	462
16.10.6	Restrictions on Using DTC or RFU .....	462
16.10.7	SCI Operations during Mode Transitions .....	463
16.10.8	Notes on Switching from SCK Pins to Port Pins .....	466
16.11	CRC Operation Circuit.....	467
16.11.1	Features .....	467
16.11.2	Register Descriptions .....	467
16.11.3	CRC Operation Circuit Operation.....	469

16.11.4 Note on CRC Operation Circuit.....	472
<b>Section 17 I<sup>2</sup>C Bus Interface (IIC).....</b>	<b>473</b>
17.1 Features.....	473
17.2 Input/Output Pins.....	476
17.3 Register Descriptions.....	476
17.3.1 I <sup>2</sup> C Bus Data Register (ICDR).....	477
17.3.2 Slave Address Register (SAR).....	480
17.3.3 Second Slave Address Register (SARX).....	481
17.3.4 I <sup>2</sup> C Bus Mode Register (ICMR).....	482
17.3.5 I <sup>2</sup> C Bus Control Register (ICCR).....	485
17.3.6 I <sup>2</sup> C Bus Status Register (ICSR).....	492
17.3.7 IIC Operation Reservation Adapter Control Register (ICCRX).....	496
17.3.8 IIC Operation Reservation Adapter Status Register A (ICSRA).....	497
17.3.9 IIC Operation Reservation Adapter Status Register B (ICSRB).....	499
17.3.10 IIC Operation Reservation Adapter Status Register C (ICSRC).....	502
17.3.11 IIC Operation Reservation Adapter Data Register (ICDRX).....	507
17.3.12 IIC Data Shift Register (ICDRS).....	508
17.3.13 IIC Operation Reservation Adapter Count Register (ICCNT).....	508
17.3.14 IIC Operation Reservation Adapter Command Register (ICCMD).....	510
17.4 IIC Operation Reservation Adapter.....	510
17.4.1 Restrictions on Accessing IIC Registers.....	510
17.4.2 Operation Reservation Commands.....	512
17.5 Operation.....	516
17.5.1 I <sup>2</sup> C Bus Data Format.....	516
17.5.2 Master Transmit Operation.....	517
17.5.3 Master Receive Operation.....	519
17.5.4 Slave Receive Operation.....	522
17.5.5 Slave Transmit Operation.....	525
17.5.6 IRIC Setting Timing and SCL Control.....	527
17.5.7 Operation Using DTC.....	531
17.5.8 Noise Canceler.....	533
17.5.9 Initialization of Internal State.....	534
17.5.10 Sample Flowcharts.....	535
17.6 Interrupt Sources.....	539
17.7 Usage Notes.....	541
<b>Section 18 Universal Serial Bus Interface (USB).....</b>	<b>553</b>
18.1 Features.....	553
18.2 Input/Output Pins.....	555
18.3 Register Descriptions.....	555

18.3.1	USB Data FIFO.....	557
18.3.2	Endpoint Size Register 1 (EPSZR1) .....	558
18.3.3	Endpoint Data Registers 0S, 0O, 0I, 1, 2, and 3 (EPDR0S, EPDR0O, EPDR0I, EPDR1, EPDR2, and EPDR3).....	559
18.3.4	Endpoint Valid Size Registers 0S, 0O, 0I, 1, 2, and 3 (FVSR0S, FVSR0O, FVSR0I, FVSR1, FVSR 2, and FVSR3).....	560
18.3.5	Endpoint Direction Register 0 (EPDIR0) .....	563
18.3.6	Packet Transfer Enable Register 0 (PTTER0) .....	564
18.3.7	USB Interrupt Enable Registers 0 and 1 (USBIER0, USBIER1) .....	565
18.3.8	USB Interrupt Flag Registers 0 and 1 (USBIFR0, USBIFR1).....	567
18.3.9	Transfer Normal Completion Interrupt Flag Register 0 (TSFR0).....	571
18.3.10	Transfer Abnormal Completion Interrupt Flag Register 0 (TFFR0).....	576
18.3.11	USB Control /Status Register 0 (USBCSR0).....	581
18.3.12	Endpoint Stall Register 0 (EPSTLR0) .....	584
18.3.13	Endpoint Reset Register 0 (EPRSTR0).....	587
18.3.14	Device Resume Register (DEVRSMR) .....	589
18.3.15	Interrupt Source Select Register 0 (INTSELR0).....	590
18.3.16	USB Control Registers 0 and 1 (USBCR0, USBCR1) .....	592
18.3.17	USB PLL Control Register (UPLLCR) .....	595
18.3.18	Configuration Value Register (CONFV) .....	597
18.3.19	Endpoint 4 Packet Size Register (EP4PKTSZR) .....	598
18.3.20	RFU/FIFO Read Request Flag Register (UDTRFR) .....	599
18.3.21	USB Mode Control Register (USBMDCR).....	600
18.3.22	USB Port Control Register (UPRTCR), and USB Test Registers 0 and 1 (UTESTR0 and UTESTR1).....	601
18.4	Operation.....	602
18.4.1	USB Function Core Functions .....	602
18.4.2	Operation on Receiving a SETUP Token (Endpoint 0) .....	604
18.4.3	Operation on Receiving an OUT Token (Endpoints 0, 2, and 5).....	610
18.4.4	Operation on Receiving an IN Token (Endpoints 0, 1, 2, 3 and 4).....	614
18.4.5	Suspend/Resume Operation .....	618
18.4.6	USB Module Reset and Operation Stop Modes.....	618
18.4.7	USB Module Startup Sequence.....	621
18.5	Interrupt Sources .....	625
18.6	Usage Notes .....	626
Section 19 Multimedia Card Interface (MCIF).....		627
19.1	Features .....	627
19.2	Input/Output Pins .....	629
19.3	Register Descriptions .....	630
19.3.1	Mode Register (MODER).....	631

19.3.2	Command Type Register (CMDTYR).....	632
19.3.3	Response Type Register (RSPTYR).....	633
19.3.4	Transfer Byte Number Count Register (TBCR) .....	636
19.3.5	Transfer Block Number Counter (TBNCR).....	636
19.3.6	Command Registers 0 to 5 (CMDR0 to CMDR5).....	637
19.3.7	Response Registers 0 to 16, and D (RSPR0 to RSPR16, and RSPRD) .....	638
19.3.8	Command Start Register (CMDSTRT).....	640
19.3.9	Operation Control Register (OPCR).....	641
19.3.10	Command Timeout Control Register (CTOCR).....	643
19.3.11	Data Timeout Register (DTOUTR) .....	644
19.3.12	Card Status Register (CSTR).....	645
19.3.13	Interrupt Control Registers 0, 1 (INTCR0, INTCR1) .....	647
19.3.14	Interrupt Status Registers 0, 1 (INTSTR0, INTSTR1).....	649
19.3.15	Pin Mode Control Register (IOMCR).....	653
19.3.16	Transfer Clock Control Register (CLKON).....	654
19.4	MCIF Activation.....	655
19.4.1	Initial Status .....	655
19.4.2	Activation Procedure .....	655
19.5	Operations in MMC Mode.....	656
19.5.1	Operation of Broadcast Commands .....	656
19.5.2	Operation of Relative Address Commands.....	657
19.5.3	Operation of Commands Not Requiring Command Response.....	657
19.5.4	Operation of Commands without Data Transfer .....	659
19.5.5	Commands with Read Data.....	663
19.5.6	Commands with Write Data.....	669
19.6	Operations in SPI Mode.....	675
19.6.1	Operation of Commands without Data Transfer.....	675
19.6.2	Commands with Read Data.....	679
19.6.3	Commands with Write Data.....	683
19.7	Interrupt Sources .....	687
19.8	Usage Notes .....	687
 <b>Section 20 Encryption Operation Circuit (DES and GF).....</b>		<b>689</b>
 <b>Section 21 D/A Converter.....</b>		<b>691</b>
21.1	Features .....	691
21.2	Input/Output Pins .....	692
21.3	Register Descriptions .....	692
21.3.1	D/A Data Registers 0 and 1 (DADR0, DADR1) .....	692
21.3.2	D/A Control Register (DACR) .....	692
21.4	Operation.....	694

21.5	Usage Notes .....	695
Section 22	A/D Converter .....	697
22.1	Features .....	697
22.2	Input/Output Pins .....	699
22.3	Register Descriptions .....	700
22.3.1	A/D Data Registers A to D (ADDRA to ADDR D).....	700
22.3.2	A/D Control/Status Register (ADCSR) .....	701
22.3.3	A/D Control Register (ADCR) .....	702
22.3.4	Keyboard Comparator Control Register (KBCOMP).....	703
22.4	DTC Comparator Scan.....	704
22.5	Operation.....	705
22.5.1	Single Mode .....	705
22.5.2	Scan Mode .....	706
22.5.3	Input Sampling and A/D Conversion Time.....	706
22.5.4	External Trigger Input Timing.....	708
22.6	Interrupt Source.....	708
22.7	A/D Conversion Accuracy Definitions .....	709
22.8	Usage Notes .....	711
22.8.1	Permissible Signal Source Impedance .....	711
22.8.2	Influences on Absolute Accuracy .....	711
22.8.3	Setting Range of Analog Power Supply and Other Pins .....	712
22.8.4	Notes on Board Design .....	712
22.8.5	Notes on Noise Countermeasures .....	712
Section 23	RAM .....	715
Section 24	ROM .....	717
24.1	Features .....	717
24.2	Mode Transition Diagrams .....	718
24.3	Block Configuration.....	722
24.4	Input/Output Pins .....	723
24.5	Register Descriptions .....	723
24.5.1	Flash Memory Control Register 1 (FLMCR1).....	723
24.5.2	Flash Memory Control Register 2 (FLMCR2).....	725
24.5.3	Erase Block Registers 1 and 2 (EBR1, EBR2) .....	725
24.6	Operating Modes.....	727
24.7	On-Board Programming Modes.....	727
24.7.1	Boot Mode .....	728
24.7.2	User Program Mode.....	732
24.8	Flash Memory Programming/Erasing.....	732

24.8.1	Program/Program-Verify .....	733
24.8.2	Erase/Erase-Verify .....	735
24.9	Program/Erase Protection.....	737
24.9.1	Hardware Protection .....	737
24.9.2	Software Protection.....	737
24.9.3	Error Protection.....	737
24.10	Interrupts during Flash Memory Programming/Erasing .....	738
24.11	Programmer Mode .....	738
24.12	Usage Notes .....	739
<b>Section 25</b>	<b>User Debug Interface (H-UDI).....</b>	<b>741</b>
25.1	Features .....	741
25.2	Input/Output Pins .....	743
25.3	Register Descriptions .....	744
25.3.1	Instruction Register (SDIR) .....	744
25.3.2	Bypass Register (SDBPR) .....	746
25.3.3	Boundary Scan Register (SDBSR).....	746
25.3.4	ID Code Register (SDIDR).....	754
25.4	Operation.....	755
25.4.1	TAP Controller State Transitions.....	755
25.4.2	H-UDI Reset .....	755
25.5	Boundary Scan .....	756
25.5.1	Supported Instructions .....	756
25.5.2	Notes .....	757
25.6	Usage Notes .....	758
<b>Section 26</b>	<b>Clock Pulse Generator .....</b>	<b>761</b>
26.1	Oscillator.....	762
26.1.1	Connecting a Crystal Oscillator .....	762
26.1.2	External Clock Input Method.....	763
26.2	Duty Correction Circuit.....	765
26.3	Medium-Speed Clock Divider .....	766
26.4	Bus Master Clock Select Circuit .....	766
26.5	Subclock Input Circuit .....	766
26.6	Waveform Forming Circuit.....	767
26.7	Clock Select Circuit .....	767
26.8	PLL Circuit .....	767
26.9	Usage Notes .....	768
26.9.1	Note on Resonator.....	768
26.9.2	Notes on Board Design .....	768
26.9.3	Processing for X1 and X2 Pins .....	769



Section 27	Power-Down Modes.....	771
27.1	Register Descriptions .....	772
27.1.1	Standby Control Register (SBYCR) .....	772
27.1.2	Low-Power Control Register (LPWRCR) .....	774
27.1.3	System Control Register 2 (SYSCR2) .....	775
27.1.4	Module Stop Control Registers H and L (MSTPCRH, MSTPCRL) Sub-Chip Module Stop Control Registers BH and BL (SUBMSTPBH, SUBMSTPBL) ....	777
27.2	Mode Transitions and LSI States .....	778
27.3	Medium-Speed Mode.....	781
27.4	Sleep Mode .....	782
27.5	Software Standby Mode.....	782
27.6	Hardware Standby Mode .....	784
27.7	Watch Mode.....	785
27.8	Subsleep Mode.....	786
27.9	Subactive Mode .....	787
27.10	Module Stop Mode .....	787
27.11	Direct Transitions.....	788
27.12	Usage Notes .....	789
27.12.1	I/O Port Status.....	789
27.12.2	Current Consumption when Waiting for Oscillation Stabilization .....	789
27.12.3	DTC Module Stop Mode .....	789
Section 28	List of Registers.....	791
28.1	Register Addresses (Address Order).....	791
28.2	Register Bits.....	804
28.3	Register States in Each Operating Mode.....	815
Section 29	Electrical Characteristics.....	825
29.1	Absolute Maximum Ratings .....	825
29.2	DC Characteristics .....	826
29.3	AC Characteristics .....	834
29.3.1	Clock Timing .....	835
29.3.2	Control Signal Timing .....	837
29.3.3	Bus Timing .....	839
29.3.4	Timing of On-Chip Peripheral Modules .....	845
29.4	A/D Conversion Characteristics.....	856
29.5	D/A Conversion Characteristics.....	858
29.6	Flash Memory Characteristics.....	859
Appendix	.....	861
A.	I/O Port States in Each Pin State.....	861

B. Product Lineup.....	863
C. Package Dimensions .....	864
Index .....	865

# Figures

## Section 1 Overview

Figure 1.1	Internal Block Diagram .....	2
Figure 1.2	Pin Arrangement (TBP-112A: Top View).....	3

## Section 2 CPU

Figure 2.1	Exception Vector Table (Normal Mode) .....	21
Figure 2.2	Stack Structure in Normal Mode .....	21
Figure 2.3	Exception Vector Table (Advanced Mode) .....	22
Figure 2.4	Stack Structure in Advanced Mode .....	23
Figure 2.5	Memory Map .....	24
Figure 2.6	CPU Internal Registers .....	25
Figure 2.7	Usage of General Registers.....	26
Figure 2.8	Stack .....	27
Figure 2.9	General Register Data Formats (1) .....	30
Figure 2.9	General Register Data Formats (2) .....	31
Figure 2.10	Memory Data Formats .....	32
Figure 2.11	Instruction Formats (Examples).....	44
Figure 2.12	Branch Address Specification in Memory Indirect Addressing Mode .....	47
Figure 2.13	State Transitions .....	51

## Section 3 MCU Operating Modes

Figure 3.1	Address Map (Mode 2).....	62
Figure 3.2	Address Map (Mode 3).....	63

## Section 4 Exception Handling

Figure 4.1	Reset Sequence (Mode 3) .....	68
Figure 4.2	Stack Status after Exception Handling.....	70
Figure 4.3	Operation when SP Value Is Odd .....	71

## Section 5 Interrupt Controller

Figure 5.1	Block Diagram of Interrupt Controller .....	74
Figure 5.2	Block Diagram of Interrupts IRQ15 to IRQ0 .....	85
Figure 5.3	Block Diagram of Interrupts KIN9 to KIN0 and WUE15 to WUE8 (Example of KIN9 to KIN0).....	86
Figure 5.4	Block Diagram of Interrupt Control Operation.....	90
Figure 5.5	Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0 .....	93
Figure 5.6	State Transition in Interrupt Control Mode 1.....	94

Figure 5.7	Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1 .....	96
Figure 5.8	Interrupt Exception Handling .....	97
Figure 5.9	Interrupt Control for DTC.....	99
Figure 5.10	Conflict between Interrupt Generation and Disabling .....	101

## Section 6 Bus Controller

Figure 6.1	Block Diagram of Bus Controller .....	104
Figure 6.2	$\overline{IOS}$ Signal Output Timing.....	122
Figure 6.3	Access Sizes and Data Alignment Control (8-Bit Access Space).....	123
Figure 6.4	Access Sizes and Data Alignment Control (16-bit Access Space) .....	124
Figure 6.5	Bus Timing for 8-Bit, 2-State Access Space.....	125
Figure 6.6	Bus Timing for 8-Bit, 3-State Access Space.....	126
Figure 6.7	Bus Timing for 16-Bit, 2-State Access Space (Even Byte Access) .....	127
Figure 6.8	Bus Timing for 16-Bit, 2-State Access Space (Odd Byte Access) .....	128
Figure 6.9	Bus Timing for 16-Bit, 2-State Access Space (Word Access).....	129
Figure 6.10	Bus Timing for 16-Bit, 3-State Access Space (Even Byte Access) .....	130
Figure 6.11	Bus Timing for 16-Bit, 3-State Access Space (Odd Byte Access) .....	131
Figure 6.12	Bus Timing for 16-Bit, 3-State Access Space (Word Access).....	132
Figure 6.13	Example of Wait State Insertion Timing (Pin Wait Mode) .....	134
Figure 6.14	Access Timing Example in Burst ROM Space (AST = BRSTS1 = 1) .....	135
Figure 6.15	Access Timing Example in Burst ROM Space (AST = BRSTS1 = 0) .....	136
Figure 6.16	Access Sizes and Data Alignment Control .....	137
Figure 6.17	Access Timing in Memory Card Mode (Basic Cycle).....	139
Figure 6.18	Access Timing in Memory Card Mode (OWEAC = OWENC = 1 with Wait State Insertion) .....	139
Figure 6.19	Access Timing Example in Memory Card Mode (Wait State Insertion by Program Wait and $\overline{CPWAIT}$ Pin) .....	140
Figure 6.20	Examples of Idle Cycle Operation.....	141

## Section 7 Data Transfer Controller (DTC)

Figure 7.1	Block Diagram of DTC.....	146
Figure 7.2	Block Diagram of DTC Activation Source Control.....	152
Figure 7.3	DTC Register Information Location in Address Space .....	153
Figure 7.4	DTC Operation Flowchart .....	156
Figure 7.5	Memory Mapping in Normal Mode.....	157
Figure 7.6	Memory Mapping in Repeat Mode.....	158
Figure 7.7	Memory Mapping in Block Transfer Mode.....	159
Figure 7.8	Chain Transfer Operation .....	160
Figure 7.9	DTC Operation Timing (Example in Normal Mode or Repeat Mode).....	161

Figure 7.10	DTC Operation Timing (Example of Block Transfer Mode, with Block Size of 2) .....	162
Figure 7.11	DTC Operation Timing (Example of Chain Transfer).....	162
 <b>Section 8 RAM-FIFO Unit (RFU)</b>		
Figure 8.1	Block Diagram of RFU.....	168
Figure 8.2	Examples of Temporary Cancellation of Medium-Speed Mode .....	188
Figure 8.3	Example of RFU Response Time .....	190
Figure 8.4	RFU Interface of USB .....	194
Figure 8.5	Operation Flow of USB IN Transfer .....	195
Figure 8.6	Operation Flow of USB OUT Transfer.....	196
Figure 8.7	RFU Interface of SCI.....	197
Figure 8.8	Operation Flow of SCI Transmission .....	198
Figure 8.9	Operation Flow of SCI Reception.....	199
Figure 8.10	RFU Interface of MCIF .....	200
Figure 8.11	Operation Flow of MCIF Transmission.....	201
Figure 8.12	Operation Flow of MCIF Reception .....	202
Figure 8.13	RFU Initialization Flow .....	203
 <b>Section 10 8-Bit PWM Timer (PWM)</b>		
Figure 10.1	Block Diagram of PWM Timer .....	262
Figure 10.2	Example of Additional Pulse Timing (When Upper 4 Bits of PWDR = B'1000) .....	271
 <b>Section 11 14-Bit PWM Timer (PWMX)</b>		
Figure 11.1	PWM (D/A) Block Diagram.....	273
Figure 11.2	PWM (D/A) Operation .....	281
Figure 11.3	Output Waveform (OS = 0, DADR corresponds to $T_L$ ).....	283
Figure 11.4	Output Waveform (OS = 1, DADR corresponds to $T_H$ ) .....	284
Figure 11.5	D/A Data Register Configuration when CFS = 1 .....	284
Figure 11.6	Output Waveform when DADR = H'0207 (OS = 1).....	285
 <b>Section 12 16-Bit Free-Running Timer (FRT)</b>		
Figure 12.1	Block Diagram of 16-Bit Free-Running Timer.....	288
Figure 12.2	Example of Pulse Output .....	299
Figure 12.3	Increment Timing with Internal Clock Source.....	300
Figure 12.4	Increment Timing with External Clock Source .....	300
Figure 12.5	Timing of Output Compare A Output.....	301
Figure 12.6	Clearing of FRC by Compare-Match A Signal.....	301
Figure 12.7	Input Capture Input Signal Timing (Usual Case) .....	302
Figure 12.8	Input Capture Input Signal Timing (When ICRA to ICRD Is Read).....	302

Figure 12.9	Buffered Input Capture Timing .....	303
Figure 12.10	Buffered Input Capture Timing (BUFEA = 1).....	304
Figure 12.11	Timing of Input Capture Flag (ICFA, ICFB, ICFC, or ICFD) Setting .....	304
Figure 12.12	Timing of Output Compare Flag (OCFA or OCFB) Setting .....	305
Figure 12.13	Timing of Overflow Flag (OVF) Setting .....	306
Figure 12.14	OCRA Automatic Addition Timing.....	306
Figure 12.15	Timing of Input Capture Mask Signal Setting.....	307
Figure 12.16	Timing of Input Capture Mask Signal Clearing.....	307
Figure 12.17	FRC Write-Clear Conflict.....	309
Figure 12.18	FRC Write-Increment Conflict.....	310
Figure 12.19	Conflict between OCR Write and Compare-Match (When Automatic Addition Function Is Not Used) .....	311
Figure 12.20	Conflict between OCR Write and Compare-Match (When Automatic Addition Function Is Used) .....	312

### Section 13 8-Bit Timer (TMR)

Figure 13.1	Block Diagram of 8-Bit Timer (TMR_0 and TMR_1).....	316
Figure 13.2	Block Diagram of 8-Bit Timer (TMR_Y and TMR_X) .....	317
Figure 13.3	Pulse Output Example .....	330
Figure 13.4	Count Timing for Internal Clock Input.....	331
Figure 13.5	Count Timing for External Clock Input.....	331
Figure 13.6	Timing of CMF Setting at Compare-Match.....	332
Figure 13.7	Timing of Toggled Timer Output by Compare-Match A Signal .....	332
Figure 13.8	Timing of Counter Clear by Compare-Match.....	333
Figure 13.9	Timing of Counter Clear by External Reset Input .....	333
Figure 13.10	Timing of OVF Flag Setting.....	334
Figure 13.11	Timing of Input Capture Operation .....	336
Figure 13.12	Timing of Input Capture Signal (Input Capture Signal Is Input during TICRR and TICRF Read).....	337
Figure 13.13	Input Capture Signal Selection .....	337
Figure 13.14	Conflict between TCNT Write and Clear .....	340
Figure 13.15	Conflict between TCNT Write and Increment.....	340
Figure 13.16	Conflict between TCOR Write and Compare-Match.....	341

### Section 14 Timer Connection

Figure 14.1	Block Diagram of Timer Connection.....	346
Figure 14.2	Timing Chart for PWM Decoding.....	359
Figure 14.3	Timing Chart for Clamp Waveform Generation (CL1 and CL2 Signals).....	360
Figure 14.4	Timing Chart for Clamp Waveform Generation (CL3 Signal).....	360
Figure 14.5	Timing Chart for Measurement of IVI Signal and IHI Signal Divided Waveform Periods .....	363

Figure 14.6	2fH Modification Timing Chart.....	364
Figure 14.7	Fall Modification and IHI Synchronization Timing Chart.....	366
Figure 14.8	IVG Signal/IHG Signal/CL4 Signal Timing Chart.....	369
Figure 14.9	CBLANK Output Waveform Generation .....	372

## Section 15 Watchdog Timer (WDT)

Figure 15.1	Block Diagram of WDT .....	374
Figure 15.2	Watchdog Timer Mode ( $RST/\overline{NMI} = 1$ ) Operation .....	380
Figure 15.3	Interval Timer Mode Operation .....	381
Figure 15.4	OVF Flag Set Timing .....	381
Figure 15.5	Output Timing of $\overline{RESO}$ Signal.....	382
Figure 15.6	Writing to TCNT and TCSR (WDT_0).....	383
Figure 15.7	Conflict between TCNT Write and Increment.....	384
Figure 15.8	Sample Circuit for Resetting the System by the $\overline{RESO}$ Signal .....	385

## Section 16 Serial Communication Interface (SCI, IrDA, and CRC)

Figure 16.1	Block Diagram of SCI_1 .....	389
Figure 16.2	Block Diagram of SCI_0 and SCI_2 .....	390
Figure 16.3	Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits) .....	417
Figure 16.4	Receive Data Sampling Timing in Asynchronous Mode.....	419
Figure 16.5	Relation between Output Clock and Transmit Data Phase (Asynchronous Mode) .....	420
Figure 16.6	Basic Clock Examples When Average Transfer Rate Is Selected (1).....	422
Figure 16.7	Basic Clock Examples When Average Transfer Rate Is Selected (2).....	423
Figure 16.8	Sample SCI Initialization Flowchart.....	424
Figure 16.9	Example of Operation in Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit) .....	425
Figure 16.10	Sample Serial Transmission Flowchart.....	426
Figure 16.11	Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit) .....	427
Figure 16.12	Sample Serial Reception Flowchart (1) .....	429
Figure 16.12	Sample Serial Reception Flowchart (2) .....	430
Figure 16.13	Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A).....	432
Figure 16.14	Sample Multiprocessor Serial Transmission Flowchart.....	433
Figure 16.15	Example of SCI Operation in Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit) .....	434
Figure 16.16	Sample Multiprocessor Serial Reception Flowchart (1) .....	435
Figure 16.16	Sample Multiprocessor Serial Reception Flowchart (2) .....	436
Figure 16.17	Data Format in Synchronous Communication (LSB-First) .....	437

Figure 16.18	Sample SCI Initialization Flowchart.....	438
Figure 16.19	Sample SCI Transmission Operation in Clocked Synchronous Mode.....	439
Figure 16.20	Sample Serial Transmission Flowchart.....	440
Figure 16.21	Example of SCI Receive Operation in Clocked Synchronous Mode.....	441
Figure 16.22	Sample Serial Reception Flowchart.....	442
Figure 16.23	Sample Flowchart of Simultaneous Serial Transmission and Reception.....	444
Figure 16.24	Pin Connection for Smart Card Interface.....	445
Figure 16.25	Data Formats in Normal Smart Card Interface Mode.....	446
Figure 16.26	Direct Convention ( $SDIR = SINV = O/\bar{E} = 0$ ).....	446
Figure 16.27	Inverse Convention ( $SDIR = SINV = O/\bar{E} = 1$ ).....	447
Figure 16.28	Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency Is 372 Times the Bit Rate) .....	448
Figure 16.29	Data Re-transfer Operation in SCI Transmission Mode .....	451
Figure 16.30	TEND Flag Set Timings during Transmission .....	451
Figure 16.31	Sample Transmission Flowchart.....	452
Figure 16.32	Data Re-transfer Operation in SCI Reception Mode .....	453
Figure 16.33	Sample Reception Flowchart.....	454
Figure 16.34	Clock Output Fixing Timing.....	455
Figure 16.35	Clock Stop and Restart Procedure .....	456
Figure 16.36	IrDA Block Diagram .....	456
Figure 16.37	IrDA Transmission and Reception.....	457
Figure 16.38	Sample Transmission using DTC in Clocked Synchronous Mode .....	463
Figure 16.39	Sample Flowchart for Mode Transition during Transmission .....	464
Figure 16.40	Pin States during Transmission in Asynchronous Mode (Internal Clock).....	464
Figure 16.41	Pin States during Transmission in Clocked Synchronous Mode (Internal Clock).....	465
Figure 16.42	Sample Flowchart for Mode Transition during Reception.....	465
Figure 16.43	Switching from SCK Pins to Port Pins .....	466
Figure 16.44	Prevention of Low Pulse Output at Switching from SCK Pins to Port Pins .....	466
Figure 16.45	Block Diagram of CRC Operation Circuit.....	467
Figure 16.46	LSB-First Data Transmission .....	469
Figure 16.47	MSB-First Data Transmission .....	469
Figure 16.48	LSB-First Data Reception.....	470
Figure 16.49	MSB-First Data Reception.....	471
Figure 16.50	LSB-First and MSB-First Transmit Data.....	472

## Section 17 I<sup>2</sup>C Bus Interface (IIC)

Figure 17.1	Block Diagram of I <sup>2</sup> C Bus Interface .....	475
Figure 17.2	I <sup>2</sup> C Bus Interface Connections (Example: This LSI as Master).....	476
Figure 17.3	State Transitions of TDRE, SDRF, and RDRF Bits .....	507
Figure 17.4	I <sup>2</sup> C Bus Data Formats (I <sup>2</sup> C Bus Formats).....	516



Figure 17.5	I <sup>2</sup> C Bus Formats (Serial Formats).....	516
Figure 17.6	I <sup>2</sup> C Bus Timing.....	517
Figure 17.7	Master Transmit Mode Operation Timing Example (MLS = WAIT = 0) .....	519
Figure 17.8	Master Receive Mode Operation Timing Example (1) (MLS = ACKB = 0, WAIT = 1) .....	521
Figure 17.9	Master Receive Mode Operation Timing Example (2) (MLS = ACKB = 0, WAIT = 1) .....	522
Figure 17.10	Slave Receive Mode Operation Timing Example (1) (MLS = ACKB = 0).....	523
Figure 17.11	Slave Receive Mode Operation Timing Example (2) (MLS = ACKB = 0).....	524
Figure 17.12	Slave Transmit Mode Operation Timing Example (MLS = 0).....	526
Figure 17.13	IRIC Flag Timing and SCL Control (1).....	527
Figure 17.14	IRIC Flag Timing and SCL Control (2).....	528
Figure 17.15	IRIC Flag Timing and SCL Control (3).....	529
Figure 17.16	Example of Interrupt Flag Timing of Operation Reservation Adapter .....	530
Figure 17.17	Block Diagram of Noise Canceler .....	534
Figure 17.18	Sample Flowchart for Master Transmit Mode.....	536
Figure 17.19	Sample Flowchart for Master Receive Mode .....	537
Figure 17.20	Sample Flowchart for Slave Receive Mode.....	538
Figure 17.21	Sample Flowchart for Slave Transmit Mode .....	539
Figure 17.22	Notes on Reading Master Receive Data .....	544
Figure 17.23	Flowchart and Timing of Start Condition Issuance for Retransmission .....	545
Figure 17.24	Stop Condition Issuance Timing.....	546
Figure 17.25	IRIC Flag Clearing Timing When WAIT = 1 .....	546
Figure 17.26	ICDR Read and ICCR Access Timing in Slave Transmit Mode .....	547
Figure 17.27	TRS Bit Set Timing in Slave Mode .....	548
Figure 17.28	IRIC Flag Clear Timing on WAIT Operation.....	550
Figure 17.29	Diagram of Erroneous Operation when Arbitration Is Lost.....	551

## Section 18 Universal Serial Bus Interface (USB)

Figure 18.1	Block Diagram of USB.....	554
Figure 18.2	Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is not Required and When SETICNT = 0) .....	606
Figure 18.3	Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is Required and When SETICNT = 0) .....	607
Figure 18.4	Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is Not Required and When SETICNT = 1) ....	608
Figure 18.5	Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is Required and When SETICNT = 1) .....	609
Figure 18.6	Operation on Receiving an OUT Token (EP2-OUT: Initial FIFO Is Empty).....	610
Figure 18.7	Operation on Receiving an OUT Token (EP2-OUT: Initial FIFO Is Full).....	611
Figure 18.8	Operation on Receiving an OUT Token (EP5-OUT: Initial FIFO Is Empty).....	612

Figure 18.9	Operation on Receiving an OUT Token (EP5-OUT: Initial FIFO Is Full) .....	613
Figure 18.10	Operation on Receiving an IN Token (EP2-IN: Initial FIFO Is Full) .....	614
Figure 18.11	Operation on Receiving an IN Token (EP2-IN: Initial FIFO Is Empty) .....	615
Figure 18.12	Operation on Receiving an IN Token (EP4-IN: Initial FIFO Is Full) .....	616
Figure 18.13	Operation on Receiving an IN Token (EP4-IN: Initial FIFO Is Empty) .....	617
Figure 18.14	Operation Procedure for Initializing USB Module .....	624

## **Section 19 Multimedia Card Interface (MCIF)**

Figure 19.1	Block Diagram of MCIF .....	628
Figure 19.2	Example of Command Sequence for Commands that Do Not Require Command Response .....	658
Figure 19.3	Operational Flow for Commands that Do Not Require Command Response.....	658
Figure 19.4	Example of Command Sequence for Commands without Data Transfer (No Data Busy State) .....	660
Figure 19.5	Example of Command Sequence for Commands without Data Transfer (with Data Busy State).....	661
Figure 19.6	Operational Flow for Commands without Data Transfer.....	662
Figure 19.7	Example of Command Sequence for Commands with Read Data (1).....	664
Figure 19.8	Example of Command Sequence for Commands with Read Data (2).....	665
Figure 19.9	Example of Command Sequence for Commands with Read Data (3).....	666
Figure 19.10	Example of Command Sequence for Commands with Read Data (4).....	667
Figure 19.11	Operational Flow for Commands with Read Data.....	668
Figure 19.12	Example of Command Sequence for Commands with Write Data (1).....	670
Figure 19.13	Example of Command Sequence for Commands with Write Data (2) .....	671
Figure 19.14	Example of Command Sequence for Commands with Write Data (3) .....	672
Figure 19.15	Example of Command Sequence for Commands with Write Data (4) .....	673
Figure 19.16	Operational Flow for Commands with Write Data .....	674
Figure 19.17	Example of Command Sequence for Commands without Data Transfer (No Data Busy State) .....	676
Figure 19.18	Example of Command Sequence for Commands without Data Transfer (with Data Busy State).....	677
Figure 19.19	Operational Flow for Commands without Data Transfer.....	678
Figure 19.20	Example of Command Sequence for Commands with Read Data (1).....	680
Figure 19.21	Example of Command Sequence for Commands with Read Data (2).....	681
Figure 19.22	Operational Flow for Commands with Read Data.....	682
Figure 19.23	Example of Command Sequence for Commands with Write Data (1).....	684
Figure 19.24	Example of Command Sequence for Commands with Write Data (2).....	685
Figure 19.25	Operational Flow for Commands with Write Data .....	686

## **Section 21 D/A Converter**

Figure 21.1	Block Diagram of D/A Converter.....	691
-------------	-------------------------------------	-----

Figure 21.2	D/A Converter Operation Example .....	694
<b>Section 22 A/D Converter</b>		
Figure 22.1	Block Diagram of A/D Converter.....	698
Figure 22.2	A/D Conversion Timing .....	707
Figure 22.3	External Trigger Input Timing.....	708
Figure 22.4	A/D Conversion Accuracy Definitions.....	710
Figure 22.5	A/D Conversion Accuracy Definitions.....	710
Figure 22.6	Example of Analog Input Circuit.....	711
Figure 22.7	Example of Analog Input Protection Circuit .....	713
Figure 22.8	Analog Input Pin Equivalent Circuit.....	713
<b>Section 24 ROM</b>		
Figure 24.1	Block Diagram of Flash Memory .....	718
Figure 24.2	Flash Memory State Transitions .....	719
Figure 24.3	Boot Mode .....	720
Figure 24.4	User Program Mode (Example).....	721
Figure 24.5	Flash Memory Block Configuration .....	722
Figure 24.6	On-Chip RAM Area in Boot Mode .....	731
Figure 24.7	ID Code Area.....	731
Figure 24.8	Programming/Erasing Flowchart Example in User Program Mode .....	732
Figure 24.9	Program/Program-Verify Flowchart.....	734
Figure 24.10	Erase/Erase-Verify Flowchart.....	736
Figure 24.11	Memory Map in Programmer Mode .....	739
<b>Section 25 User Debug Interface (H-UDI)</b>		
Figure 25.1	Block Diagram of H-UDI .....	742
Figure 25.2	TAP Controller State Transitions.....	755
Figure 25.3	Reset Signal Circuit Without Reset Signal Interference .....	758
Figure 25.4	Serial Data Input/Output (1) .....	759
Figure 25.4	Serial Data Input/Output (2) .....	760
<b>Section 26 Clock Pulse Generator</b>		
Figure 26.1	Block Diagram of Clock Pulse Generator.....	761
Figure 26.2	Typical Connection to Crystal Resonator .....	762
Figure 26.3	Equivalent Circuit of Crystal Resonator .....	762
Figure 26.4	Example of External Clock Input .....	763
Figure 26.5	External Clock Input Timing .....	764
Figure 26.6	Timing of External Clock Output Stabilization Delay Time .....	765
Figure 26.7	Subclock Input Timing .....	767
Figure 26.8	Note on Board Design of Oscillation Circuit Section.....	768

Figure 26.9	Processing for X1 and X2 Pins .....	769
<b>Section 27 Power-Down Modes</b>		
Figure 27.1	Mode Transition Diagram.....	779
Figure 27.2	Medium-Speed Mode Timing.....	782
Figure 27.3	Software Standby Mode Application Example.....	784
Figure 27.4	Hardware Standby Mode Timing.....	785
<b>Section 29 Electrical Characteristics</b>		
Figure 29.1	Darlington Transistor Drive Circuit (Example).....	833
Figure 29.2	LED Drive Circuit (Example).....	833
Figure 29.3	Output Load Circuit.....	834
Figure 29.4	System Clock Timing .....	835
Figure 29.5	Oscillation Stabilization Timing.....	836
Figure 29.6	Oscillation Stabilization Timing (Exiting Software Standby Mode).....	836
Figure 29.7	Reset Input Timing .....	838
Figure 29.8	Interrupt Input Timing .....	838
Figure 29.9	Basic Bus Timing/2-State Access.....	840
Figure 29.10	Basic Bus Timing/3-State Access.....	841
Figure 29.11	Basic Bus Timing/3-State Access with One Wait State.....	842
Figure 29.12	CF Interface Basic Timing/3-State Access .....	843
Figure 29.13	Burst ROM Access Timing/2-State Access .....	844
Figure 29.14	Burst ROM Access Timing/1-State Access .....	845
Figure 29.15	I/O Port Input/Output Timing.....	847
Figure 29.16	FRT Input/Output Timing.....	847
Figure 29.17	FRT Clock Input Timing .....	847
Figure 29.18	8-Bit Timer Output Timing.....	848
Figure 29.19	8-Bit Timer Clock Input Timing.....	848
Figure 29.20	8-Bit Timer Reset Input Timing .....	848
Figure 29.21	PWM, PWMX Output Timing.....	848
Figure 29.22	SCK Clock Input Timing.....	849
Figure 29.23	SCI Input/Output Timing (Clock Synchronous Mode).....	849
Figure 29.24	A/D Converter External Trigger Input Timing .....	849
Figure 29.25	WDT Output Timing ( $\overline{\text{RESO}}$ ).....	849
Figure 29.26	I <sup>2</sup> C Bus Interface Input/Output Timing .....	851
Figure 29.27	USB Driver/Receiver Output Timing .....	852
Figure 29.28	Multimedia Card Interface Timing .....	853
Figure 29.29	H-UDI ETCK Timing.....	854
Figure 29.30	Reset Hold Timing.....	855
Figure 29.31	H-UDI Input/Output Timing.....	855

**Appendix**

Figure C.1 Package Dimensions (TBP-112A) ..... 864



# Tables

## Section 1 Overview

Table 1.1	Pin Arrangement in Each Operating Mode .....	4
Table 1.2	Pin Functions .....	8

## Section 2 CPU

Table 2.1	Instruction Classification .....	33
Table 2.2	Operation Notation .....	34
Table 2.3	Data Transfer Instructions .....	35
Table 2.4	Arithmetic Operations Instructions (1) .....	36
Table 2.4	Arithmetic Operations Instructions (2) .....	37
Table 2.5	Logic Operations Instructions .....	38
Table 2.6	Shift Instructions .....	38
Table 2.7	Bit Manipulation Instructions (1) .....	39
Table 2.7	Bit Manipulation Instructions (2) .....	40
Table 2.8	Branch Instructions .....	41
Table 2.9	System Control Instructions .....	42
Table 2.10	Block Data Transfer Instructions .....	43
Table 2.11	Addressing Modes .....	45
Table 2.12	Absolute Address Access Ranges .....	46
Table 2.13	Effective Address Calculation (1) .....	48
Table 2.13	Effective Address Calculation (2) .....	49

## Section 3 MCU Operating Modes

Table 3.1	MCU Operating Mode Selection .....	55
Table 3.2	Pin Functions in Each Operating Mode .....	61

## Section 4 Exception Handling

Table 4.1	Exception Types and Priority .....	65
Table 4.2	Exception Handling Vector Table .....	66
Table 4.3	Status of CCR after Trap Instruction Exception Handling .....	70

## Section 5 Interrupt Controller

Table 5.1	Pin Configuration .....	75
Table 5.2	Correspondence between Interrupt Source and ICR .....	76
Table 5.3	Interrupt Sources, Vector Addresses, and Interrupt Priorities .....	87
Table 5.4	Interrupt Control Modes .....	90
Table 5.5	Interrupts Acceptable in Each Interrupt Control Mode .....	91
Table 5.6	Operations and Control Signal Functions in Each Interrupt Control Mode .....	92

Table 5.7	Interrupt Response Times.....	98
Table 5.8	Number of States in Interrupt Handling Routine Execution Status.....	98
Table 5.9	Interrupt Source Selection and Clearing Control.....	100
<b>Section 6</b>	<b>Bus Controller</b>	
Table 6.1	Pin Configuration.....	105
Table 6.2	Address Ranges and External Address Spaces.....	115
Table 6.3	Bit Settings and Bus Specifications of Basic Bus Interface.....	117
Table 6.4	Bus Specifications for Basic Expansion Area/Basic Bus Interface.....	118
Table 6.5	Bus Specifications for 256-kbyte Expansion Area/Basic Bus Interface.....	119
Table 6.6	Bus Specifications for CP Expansion Area (Basic Mode)/Basic Bus Interface.....	120
Table 6.7	Bus Specifications for CF Expansion Area (Memory Card Mode)/ Basic Bus Interface.....	121
Table 6.8	Address Range for $\overline{\text{IOS}}$ Signal Output.....	122
Table 6.9	Data Buses Used and Valid Strobes.....	124
Table 6.10	Data Buses Used and Valid Strobes.....	138
Table 6.11	Pin States in Idle Cycle.....	142
<b>Section 7</b>	<b>Data Transfer Controller (DTC)</b>	
Table 7.1	Correspondence between Interrupt Sources and DTCER.....	150
Table 7.2	Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs.....	154
Table 7.3	Register Functions in Normal Mode.....	157
Table 7.4	Register Functions in Repeat Mode.....	158
Table 7.5	Register Functions in Block Transfer Mode.....	159
Table 7.6	DTC Execution Status.....	163
Table 7.7	Number of States Required for Each Execution Status.....	163
<b>Section 8</b>	<b>RAM-FIFO Unit (RFU)</b>	
Table 8.1	Valid Bits in BAR, RAR, WAR, and TMP.....	174
Table 8.2	Correspondence between Activation Sources and ID Numbers.....	183
Table 8.3	RFU Bus Cycle Types.....	185
Table 8.4	Requests from Peripheral Modules and RFU Bus Cycle.....	186
Table 8.5	Bus Cycle Insertion.....	189
Table 8.6	Settings when Using Boundary Overflow (Transmission/Reception of Single Data Block).....	192
Table 8.7	DATAN/FREEN Read Value.....	204
<b>Section 9</b>	<b>I/O Ports</b>	
Table 9.1	Port Functions.....	205
Table 9.2	Port 1 Input Pull-Up MOS States.....	211
Table 9.3	Port 2 Input Pull-Up MOS States.....	215

Table 9.4	Port 3 Input Pull-Up MOS States .....	222
Table 9.5	Port 6 Input Pull-Up MOS States .....	244
Table 9.6	Port A Input Pull-Up MOS States .....	257
<b>Section 10</b>	<b>8-Bit PWM Timer (PWM)</b>	
Table 10.1	Pin Configuration .....	263
Table 10.2	Internal Clock Selection .....	265
Table 10.3	Resolution, PWM Conversion Period, and Carrier Frequency when $\phi = 20$ MHz.....	266
Table 10.4	Duty Cycle of Basic Pulse.....	270
Table 10.5	Position of Pulses Added to Basic Pulses.....	271
<b>Section 11</b>	<b>14-Bit PWM Timer (PWMX)</b>	
Table 11.1	Pin Configuration .....	274
Table 11.2	Read and Write Access Methods for 16-Bit Registers .....	280
Table 11.3	Settings and Operation (Examples when $\phi = 25$ MHz) .....	282
Table 11.4	Locations of Additional Pulses Added to Base Pulse (When CFS = 1) .....	286
<b>Section 12</b>	<b>16-Bit Free-Running Timer (FRT)</b>	
Table 12.1	Pin Configuration .....	289
Table 12.2	FRT Interrupt Sources .....	308
Table 12.3	Switching of Internal Clock and FRC Operation .....	313
<b>Section 13</b>	<b>8-Bit Timer (TMR)</b>	
Table 13.1	Pin Configuration .....	318
Table 13.2	Clock Input to TCNT and Count Condition .....	322
Table 13.3	Input Capture Signal Selection.....	338
Table 13.4	Interrupt Sources of 8-Bit Timers TMR_0, TMR_1, TMR_Y, and TMR_X.....	339
Table 13.5	Timer Output Priorities .....	342
Table 13.6	Switching of Internal Clocks and TCNT Operation.....	343
<b>Section 14</b>	<b>Timer Connection</b>	
Table 14.1	Pin Configuration .....	347
Table 14.2	Synchronization Signal Connection Enable .....	351
Table 14.3	Registers Accessible by TMR_X/TMR_Y.....	355
Table 14.4	Examples of TCR Settings .....	358
Table 14.5	Examples of TCORB (Pulse Width Threshold) Settings.....	358
Table 14.6	Examples of TCR and TCSR Settings.....	362
Table 14.7	Examples of TCR, TCSR, TOCR, and OCRDM Settings.....	364
Table 14.8	Examples of TCR, TCSR, and TCORB Settings .....	366



Table 14.9	Examples of TCR, TCSR, TCORA, TCORB, OCRAR, OCRAF, and TOCR Settings.....	368
Table 14.10	HSYNCO Output Modes.....	370
Table 14.11	VSYNCO Output Modes.....	371
<b>Section 15</b>	<b>Watchdog Timer (WDT)</b>	
Table 15.1	Pin Configuration.....	375
Table 15.2	WDT Interrupt Source.....	382
<b>Section 16</b>	<b>Serial Communication Interface (SCI, IrDA, and CRC)</b>	
Table 16.1	Pin Configuration.....	391
Table 16.2	Relationships between N Setting in BRR and Bit Rate B.....	405
Table 16.3	BRR Settings for Various Bit Rates (Asynchronous Mode).....	406
Table 16.4	Maximum Bit Rate for Each Frequency (Asynchronous Mode).....	409
Table 16.5	Maximum Bit Rate with External Clock Input (Asynchronous Mode).....	409
Table 16.6	BRR Settings for Various Bit Rates (Clocked Synchronous Mode).....	410
Table 16.7	Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode).....	410
Table 16.8	BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, s = 372).....	411
Table 16.9	Maximum Bit Rate for Each Frequency (Smart Card Interface Mode, S = 372) ...	411
Table 16.10	Serial Transfer Formats (Asynchronous Mode).....	418
Table 16.11	SSR Status Flags and Receive Data Handling.....	428
Table 16.12	IrCKS2 to IrCKS0 Bit Settings.....	458
Table 16.13	SCI Interrupt Sources.....	460
Table 16.14	SCI Interrupt Sources.....	460
<b>Section 17</b>	<b>I<sup>2</sup>C Bus Interface (IIC)</b>	
Table 17.1	Pin Configuration.....	476
Table 17.2	Communication Format.....	481
Table 17.3	I <sup>2</sup> C Transfer Rate.....	484
Table 17.4	Flags and Transfer States.....	490
Table 17.5	Restrictions on Accessing IIC Registers.....	511
Table 17.6	Operation Reservation Commands.....	513
Table 17.7	Operation When the Operation Reservation Command Is Completed.....	514
Table 17.8	Examples of Operation Using DTC.....	532
Table 17.9	Examples of Operation Reservation Adapter Operation Using DTC.....	533
Table 17.10	IIC Interrupt Sources.....	540
Table 17.11	I <sup>2</sup> C Bus Timing (SCL and SDA Outputs).....	541
Table 17.12	Permissible SCL Rise Time (t <sub>sr</sub> ) Values.....	542
Table 17.13	I <sup>2</sup> C Bus Timing (with Maximum Influence of t <sub>sr</sub> /t <sub>sf</sub> ).....	543

<b>Section 18</b>	<b>Universal Serial Bus Interface (USB)</b>	
Table 18.1	Pin Configuration .....	555
Table 18.2	FIFO Configuration .....	557
Table 18.3	Port 6 Functions .....	595
Table 18.4	USB Function Core and Slave CPU Functions .....	603
Table 18.5	Packets Included in Each Transaction .....	605
Table 18.6	Registers Initialized by Bit UIFRST or FSRST .....	619
Table 18.7	Endpoint Information .....	622
Table 18.8	USB Interrupt Sources (When SETICNT of USBMDCR Is 0).....	625
Table 18.9	USB Interrupt Sources (When SETICNT of USBMDCR Is 1).....	625
<b>Section 19</b>	<b>Multimedia Card Interface (MCIF)</b>	
Table 19.1	Pin Configuration .....	629
Table 19.2	Correspondence between Commands and Settings of CMDTYR and RSPTYR ...	634
Table 19.3	CMDR Configuration.....	637
Table 19.4	Correspondence between Number of Command Response Bytes and RSPR Register .....	639
Table 19.5	Card States in which Command Sequence Is Halted.....	642
Table 19.6	MCIF Interrupt Sources.....	687
<b>Section 21</b>	<b>D/A Converter</b>	
Table 21.1	Pin Configuration .....	692
Table 21.2	D/A Channel Enable.....	693
<b>Section 22</b>	<b>A/D Converter</b>	
Table 22.1	Pin Configuration .....	699
Table 22.2	Analog Input Channels and Corresponding ADDR Registers.....	700
Table 22.3	CIN7 to CIN0 Scan by DTC Comparator Scan Function.....	704
Table 22.4	A/D Conversion Time (Single Mode) .....	707
Table 22.5	A/D Converter Interrupt Source .....	708
<b>Section 24</b>	<b>ROM</b>	
Table 24.1	Differences between Boot Mode and User Program Mode.....	719
Table 24.2	Pin Configuration .....	723
Table 24.3	Operating Modes and ROM .....	727
Table 24.4	On-Board Programming Mode Settings .....	727
Table 24.5	Boot Mode Operation.....	730
Table 24.6	System Clock Frequencies for which Automatic Adjustment of LSI Bit Rate Is Possible.....	731

<b>Section 25</b>	<b>User Debug Interface (H-UDI)</b>	
Table 25.1	Pin Configuration .....	743
Table 25.2	H-UDI Register Serial Transfer.....	744
Table 25.3	Correspondence between Pins and Boundary Scan Register.....	747
<b>Section 26</b>	<b>Clock Pulse Generator</b>	
Table 26.1	Damping Resistance Values .....	762
Table 26.2	Crystal Resonator Parameters.....	763
Table 26.3	External Clock Input Conditions .....	764
Table 26.4	External Clock Output Stabilization Delay Time .....	765
Table 26.5	Subclock Input Conditions .....	766
<b>Section 27</b>	<b>Power-Down Modes</b>	
Table 27.1	Operating Frequency and Wait Time .....	774
Table 27.2	LSI Internal States in Each Operating Mode.....	780
<b>Section 29</b>	<b>Electrical Characteristics</b>	
Table 29.1	Absolute Maximum Ratings.....	825
Table 29.2	DC Characteristics (1).....	826
Table 29.2	DC Characteristics (2).....	828
Table 29.2	DC Characteristics (3).....	829
Table 29.3	Permissible Output Currents.....	830
Table 29.4	I <sup>2</sup> C Bus Drive Characteristics.....	831
Table 29.5	USB Pin Characteristics .....	832
Table 29.6	Multimedia Card Interface Pin Characteristics.....	833
Table 29.7	Clock Timing.....	835
Table 29.8	Control Signal Timing.....	837
Table 29.9	Bus Timing (1) (Normal Mode and Advanced Mode) .....	839
Table 29.10	Timing of On-Chip Peripheral Modules (1) .....	846
Table 29.11	I <sup>2</sup> C Bus Timing.....	850
Table 29.12	USB Timing .....	851
Table 29.13	Multimedia Card Interface .....	852
Table 29.14	H-UDI Timing.....	854
Table 29.15	A/D Conversion Characteristics (AN7 to AN2 Input: 134/266-State Conversion).....	856
Table 29.16	A/D Conversion Characteristics (CIN7 to CIN0 Input: 134/266-State Conversion) .....	857
Table 29.17	D/A Conversion Characteristics .....	858
Table 29.18	Flash Memory Characteristics.....	859



# Section 1 Overview

## 1.1 Features

- High-speed H8S/2000 CPU with an internal 16-bit architecture
  - Upward-compatible with H8/300 CPU and H8/300H CPU on an object level
  - Sixteen 16-bit general registers
  - 65 basic instructions
- Various peripheral functions
  - Data transfer controller (DTC)
  - RAM-FIFO unit (RFU)
  - 8-bit PWM timer (PWM)
  - 14-bit PWM timer (PWMX)
  - 16-bit free-running timer (FRT)
  - 8-bit timer (TMR)
  - Timer connection
  - Watchdog timer (WDT)
  - Asynchronous or clocked synchronous serial communication interface (SCI)
  - CRC operator (CRC)
  - I<sup>2</sup>C bus interface (IIC)
  - Universal serial bus interface (USB)
  - Multimedia card interface (MCIF)
  - Encryption operation circuit (DES, GF)
  - 8-bit D/A converter
  - 10-bit A/D converter
  - User debug interface (H-UDI)
  - Clock pulse generator
- On-chip memory
 

ROM	Model	ROM	RAM
F-ZTAT Version	HD64F2158	256 kbytes	10 kbytes
- General I/O ports
  - I/O pins: 65
  - Input-only pins: 7
- Supports various power-down modes

- Compact package

Package	Code	Body Size	Pin Pitch
TFBGA-112	TBP-112A	10.0 × 10.0 mm	0.8 mm

## 1.2 Internal Block Diagram

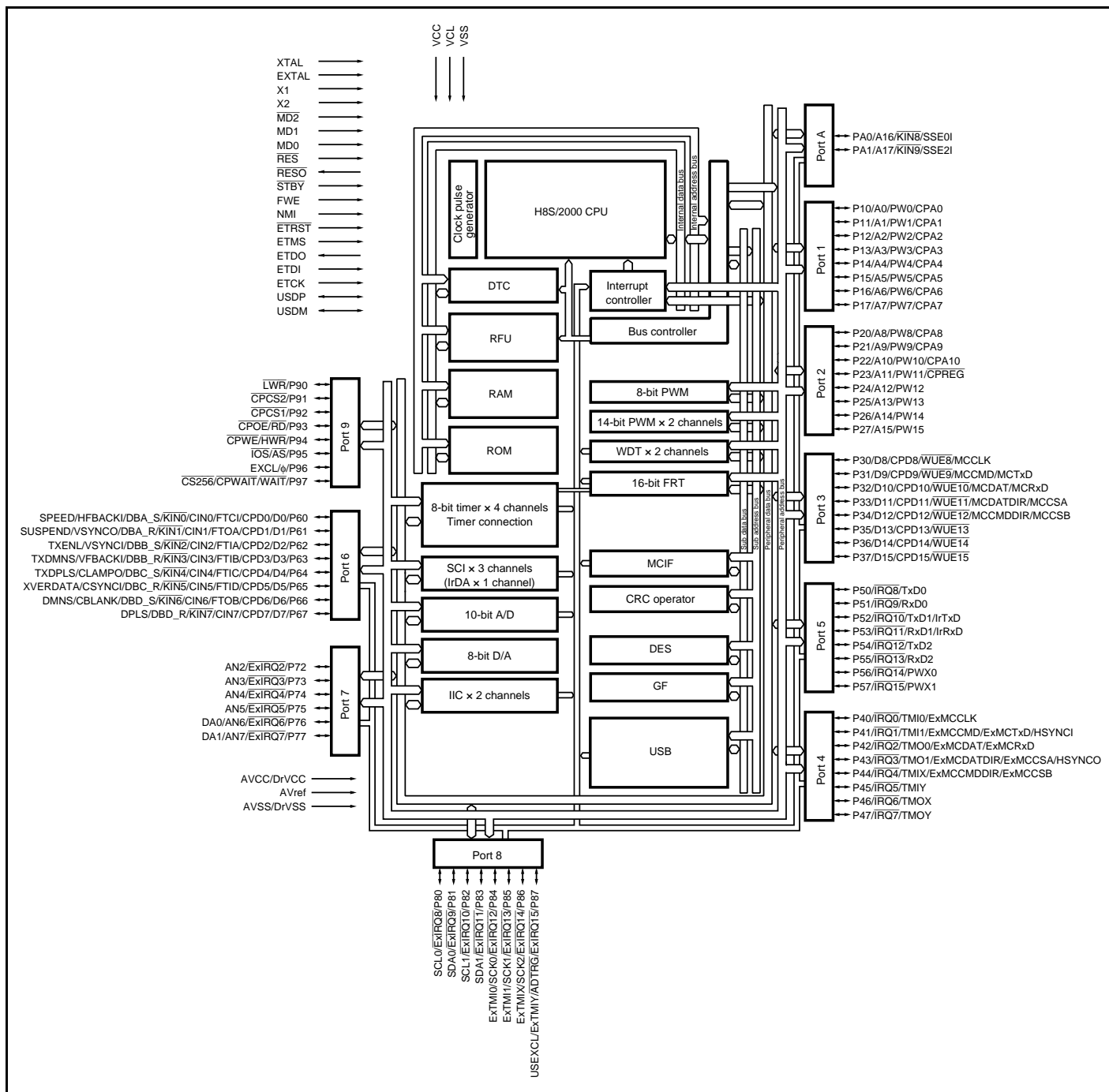


Figure 1.1 Internal Block Diagram

## 1.3 Pin Description

### 1.3.1 Pin Arrangement

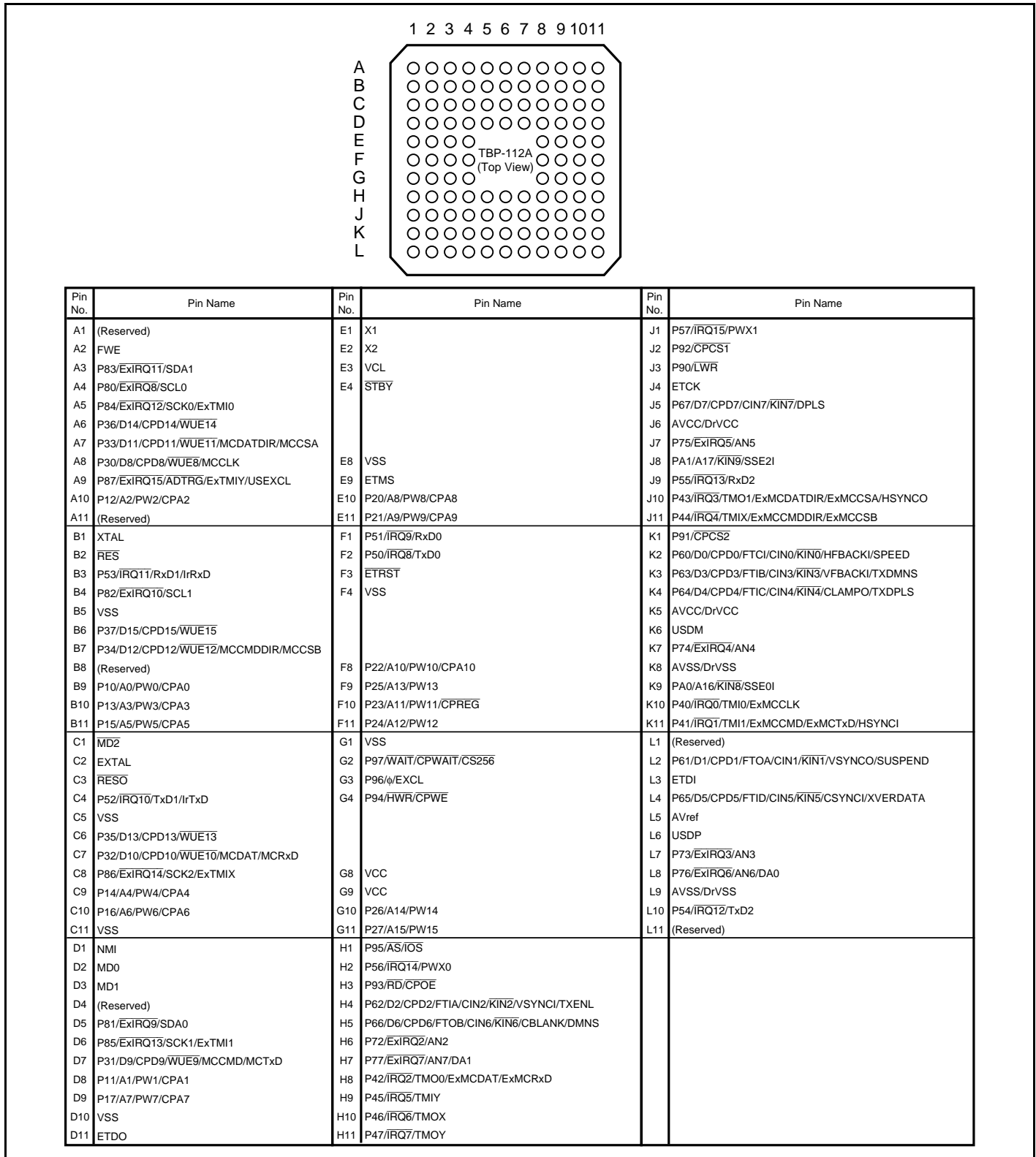


Figure 1.2 Pin Arrangement (TBP-112A: Top View)

### 1.3.2 Pin Arrangement in Each Operating Mode

**Table 1.1 Pin Arrangement in Each Operating Mode**

Pin No.	Pin Name		Flash Memory Programmer Mode
	Extended Mode Modes 2 and 3 (EXPE = 1)	Single-Chip Mode Modes 2 and 3 (EXPE = 0)	
TBP-112A			
B2	$\overline{\text{RES}}$	$\overline{\text{RES}}$	$\overline{\text{RES}}$
B1	XTAL	XTAL	XTAL
C2	EXTAL	EXTAL	EXTAL
C1	$\overline{\text{MD2}}$	$\overline{\text{MD2}}$	VCC
D3	MD1	MD1	VSS
D2	MD0	MD0	VSS
D1	NMI	NMI	FA9
E4	$\overline{\text{STBY}}$	$\overline{\text{STBY}}$	VCC
E3	VCL	VCL	VCL
E1	X1	X1	NC
E2	X2	X2	NC
F3	$\overline{\text{ETRST}}$	$\overline{\text{ETRST}}$	VSS
F1	P51/ $\overline{\text{IRQ9}}$ /RxD0	P51/ $\overline{\text{IRQ9}}$ /RxD0	FA17
F2	P50/ $\overline{\text{IRQ8}}$ /TxD0	P50/ $\overline{\text{IRQ8}}$ /TxD0	NC
F4, G1	VSS	VSS	VSS
G2	P97/ $\overline{\text{WAIT}}$ / $\overline{\text{CPWAIT}}$ / $\overline{\text{CS256}}$	P97	VCC
G3	P96/ $\phi$ /EXCL	P96/ $\phi$ /EXCL	NC
H1	$\overline{\text{AS}}$ / $\overline{\text{IOS}}$	P95	FA16
G4	$\overline{\text{HWR}}$ / $\overline{\text{CPWE}}$	P94	FA15
H2	P56/ $\overline{\text{IRQ14}}$ /PWX0	P56/ $\overline{\text{IRQ14}}$ /PWX0	NC
J1	P57/ $\overline{\text{IRQ15}}$ /PWX1	P57/ $\overline{\text{IRQ15}}$ /PWX1	NC
H3	$\overline{\text{RD}}$ / $\overline{\text{CPOE}}$	P93	$\overline{\text{WE}}$
J2	P92/ $\overline{\text{CPCS1}}$	P92	VSS
K1	P91/ $\overline{\text{CPCS2}}$	P91	VCC
J3	P90/ $\overline{\text{LWR}}$	P90	VCC
K2	P60/FTCI/CIN0/ $\overline{\text{KIN0}}$ /HFBACKI*1	D0/CPD0*2 P60/FTCI/CIN0/ $\overline{\text{KIN0}}$ /HFBACKI/SPEED	NC



Pin No.	Pin Name		Flash Memory Programmer Mode	
	Extended Mode Modes 2 and 3 (EXPE = 1)	Single-Chip Mode Modes 2 and 3 (EXPE = 0)		
L2	P61/FTOA/CIN1/ $\overline{\text{KIN1}}$ /VSYNCO* <sup>1</sup>	D1/CPD1* <sup>2</sup>	P61/FTOA/CIN1/ $\overline{\text{KIN1}}$ / VSYNCO/SUSPEND	NC
H4	P62/FTIA/CIN2/ $\overline{\text{KIN2}}$ /VSYNCI* <sup>1</sup>	D2/CPD2* <sup>2</sup>	P62/FTIA/CIN2/ $\overline{\text{KIN2}}$ / VSYNCI/TXENL	NC
K3	P63/FTIB/CIN3/ $\overline{\text{KIN3}}$ /VFBACKI* <sup>1</sup>	D3/CPD3* <sup>2</sup>	P63/FTIB/CIN3/ $\overline{\text{KIN3}}$ / VFBACKI/TXDMNS	NC
L3	ETDI		ETDI	NC
J4	ETCK		ETCK	NC
K4	P64/FTIC/CINk4/ $\overline{\text{KIN4}}$ /CLAMPO* <sup>1</sup>	D4/CPD4* <sup>2</sup>	P64/FTIC/CIN4/ $\overline{\text{KIN4}}$ / CLAMPO/TXDPLS	NC
L4	P65/FTID/CIN5/ $\overline{\text{KIN5}}$ /CSYNCI* <sup>1</sup>	D5/CPD5* <sup>2</sup>	P65/FTID/CIN5/ $\overline{\text{KIN5}}$ / CSYNCI/XVERDATA	NC
H5	P66/FTOB/CIN6/ $\overline{\text{KIN6}}$ /CBLANK* <sup>1</sup>	D6/CPD6* <sup>2</sup>	P66/FTOB/CIN6/ $\overline{\text{KIN6}}$ / CBLANK/DMNS	NC
J5	P67/CIN7/ $\overline{\text{KIN7}}$ * <sup>1</sup>	D7/CPD7* <sup>2</sup>	P67/CIN7/ $\overline{\text{KIN7}}$ /DPLS	VSS
L5	AVref		AVref	VCC
K5, J6	AVCC/DrVCC		AVCC/DrVCC	VCC
L6	USDP		USDP	NC
K6	USDM		USDM	NC
H6	P72/ $\overline{\text{ExIRQ2}}$ /AN2		P72/ $\overline{\text{ExIRQ2}}$ /AN2	NC
L7	P73/ $\overline{\text{ExIRQ3}}$ /AN3		P73/ $\overline{\text{ExIRQ3}}$ /AN3	NC
K7	P74/ $\overline{\text{ExIRQ4}}$ /AN4		P74/ $\overline{\text{ExIRQ4}}$ /AN4	NC
J7	P75/ $\overline{\text{ExIRQ5}}$ /AN5		P75/ $\overline{\text{ExIRQ5}}$ /AN5	NC
L8	P76/ $\overline{\text{ExIRQ6}}$ /AN6/DA0		P76/ $\overline{\text{ExIRQ6}}$ /AN6/DA0	NC
H7	P77/ $\overline{\text{ExIRQ7}}$ /AN7/DA1		P77/ $\overline{\text{ExIRQ7}}$ /AN7/DA1	NC
K8, L9	AVSS/DrVSS		AVSS/DrVSS	VSS
J8	PA1/A17/ $\overline{\text{KIN9}}$ / SSE2I* <sup>3</sup>	PA1/ $\overline{\text{KIN9}}$ / SSE2I* <sup>4</sup>	PA1/ $\overline{\text{KIN9}}$ /SSE2I	NC
K9	PA0/A16/ $\overline{\text{KIN8}}$ / SSE0I* <sup>3</sup>	PA0/ $\overline{\text{KIN8}}$ / SSE0I* <sup>4</sup>	PA0/ $\overline{\text{KIN8}}$ /SSE0I	NC
L10	P54/ $\overline{\text{IRQ12}}$ /TxD2		P54/ $\overline{\text{IRQ12}}$ /TxD2	NC
J9	P55/ $\overline{\text{IRQ13}}$ /RxD2		P55/ $\overline{\text{IRQ13}}$ /RxD2	NC

Pin No.	Pin Name		Flash Memory Programmer Mode
	Extended Mode Modes 2 and 3 (EXPE = 1)	Single-Chip Mode Modes 2 and 3 (EXPE = 0)	
K10	P40/ $\overline{\text{IRQ0}}$ /TMI0/ExMCCLK	P40/ $\overline{\text{IRQ0}}$ /TMI0/ExMCCLK	NC
K11	P41/ $\overline{\text{IRQ1}}$ /TMI1/ExMCCMD/ ExMCTxD/HSYNCl	P41/ $\overline{\text{IRQ1}}$ /TMI1/ExMCCMD/ ExMCTxD/HSYNCl	NC
H8	P42/ $\overline{\text{IRQ2}}$ /TMO0/ExMCDAT/ ExMCRxD	P42/ $\overline{\text{IRQ2}}$ /TMO0/ExMCDAT/ ExMCRxD	NC
J10	P43/ $\overline{\text{IRQ3}}$ /TMO1/ ExMCDATDIR/ExMCCSA/ HSYNCO	P43/ $\overline{\text{IRQ3}}$ /TMO1/ ExMCDATDIR/ExMCCSA/ HSYNCO	NC
J11	P44/ $\overline{\text{IRQ4}}$ /TMIX/ ExMCCMDDIR/ExMCCSB	P44/ $\overline{\text{IRQ4}}$ /TMIX/ ExMCCMDDIR/ExMCCSB	NC
H9	P45/ $\overline{\text{IRQ5}}$ /TMIY	P45/ $\overline{\text{IRQ5}}$ /TMIY	NC
H10	P46/ $\overline{\text{IRQ6}}$ /TMOX	P46/ $\overline{\text{IRQ6}}$ /TMOX	NC
H11	P47/ $\overline{\text{IRQ7}}$ /TMOY	P47/ $\overline{\text{IRQ7}}$ /TMOY	NC
G8, G9	VCC	VCC	VCC
G11	P27/A15	P27/PW15	$\overline{\text{CE}}$
G10	P26/A14	P26/PW14	FA14
F9	P25/A13	P25/PW13	FA13
F11	P24/A12	P24/PW12	FA12
F10	P23/A11/ $\overline{\text{CPREG}}$	P23/PW11	FA11
F8	P22/A10/CPA10	P22/PW10	FA10
E11	P21/A9/CPA9	P21/PW9	$\overline{\text{OE}}$
E10	P20/A8/CPA8	P20/PW8	FA8
E9	ETMS	ETMS	NC
D11	ETDO	ETDO	NC
E8, D10	VSS	VSS	VSS
C11	VSS	VSS	VSS
D9	P17/A7/CPA7	P17/PW7	FA7
C10	P16/A6/CPA6	P16/PW6	FA6
B11	P15/A5/CPA5	P15/PW5	FA5
C9	P14/A4/CPA4	P14/PW4	FA4
B10	P13/A3/CPA3	P13/PW3	FA3

Pin No.	Pin Name		Flash Memory Programmer Mode
	Extended Mode Modes 2 and 3 (EXPE = 1)	Single-Chip Mode Modes 2 and 3 (EXPE = 0)	
TBP-112A			
A10	P12/A2/CPA2	P12/PW2	FA2
D8	P11/A1/CPA1	P11/PW1	FA1
B9	P10/A0/CPA0	P10/PW0	FA0
A9	P87/ $\overline{\text{ExIRQ15}}$ / $\overline{\text{ADTRG}}$ / $\overline{\text{ExTMIY}}$ / $\overline{\text{USEXCL}}$	P87/ $\overline{\text{ExIRQ15}}$ / $\overline{\text{ADTRG}}$ / $\overline{\text{ExTMIY}}$ / $\overline{\text{USEXCL}}$	NC
C8	P86/ $\overline{\text{ExIRQ14}}$ /SCK2/ $\overline{\text{ExTMIX}}$	P86/ $\overline{\text{ExIRQ14}}$ /SCK2/ $\overline{\text{ExTMIX}}$	NC
A8	D8/CPD8	P30/ $\overline{\text{WUE8}}$ /MCCLK	FO0
D7	D9/CPD9	P31/ $\overline{\text{WUE9}}$ /MCCMD/MCTxD	FO1
C7	D10/CPD10	P32/ $\overline{\text{WUE10}}$ /MCDAT/MCRxD	FO2
A7	D11/CPD11	P33/ $\overline{\text{WUE11}}$ /MCDATDIR/ MCCSA	FO3
B7	D12/CPD12	P34/ $\overline{\text{WUE12}}$ /MCCMDDIR/ MCCSB	FO4
C6	D13/CPD13	P35/ $\overline{\text{WUE13}}$	FO5
A6	D14/CPD14	P36/ $\overline{\text{WUE14}}$	FO6
B6	D15/CPD15	P37/ $\overline{\text{WUE15}}$	FO7
D6	P85/ $\overline{\text{ExIRQ13}}$ /SCK1/ $\overline{\text{ExTMI1}}$	P85/ $\overline{\text{ExIRQ13}}$ /SCK1/ $\overline{\text{ExTMI1}}$	NC
A5	P84/ $\overline{\text{ExIRQ12}}$ /SCK0/ $\overline{\text{ExTMI0}}$	P84/ $\overline{\text{ExIRQ12}}$ /SCK0/ $\overline{\text{ExTMI0}}$	NC
B5, C5	VSS	VSS	VSS
A4	P80/ $\overline{\text{ExIRQ8}}$ /SCL0	P80/ $\overline{\text{ExIRQ8}}$ /SCL0	NC
D5	P81/ $\overline{\text{ExIRQ9}}$ /SDA0	P81/ $\overline{\text{ExIRQ9}}$ /SDA0	NC
B4	P82/ $\overline{\text{ExIRQ10}}$ /SCL1	P82/ $\overline{\text{ExIRQ10}}$ /SCL1	NC
A3	P83/ $\overline{\text{ExIRQ11}}$ /SDA1	P83/ $\overline{\text{ExIRQ11}}$ /SDA1	NC
C4	P52/ $\overline{\text{IRQ10}}$ /TxD1/IrTxD	P52/ $\overline{\text{IRQ10}}$ /TxD1/IrTxD	FA18
B3	P53/ $\overline{\text{IRQ11}}$ /RxD1/IrRxD	P53/ $\overline{\text{IRQ11}}$ /RxD1/IrRxD	NC
A2	FWE	FWE	FWE
C3	$\overline{\text{RESO}}$	$\overline{\text{RESO}}$	NC

- Notes:
1. 8-bit data bus
  2. 16-bit data bus
  3. Extended mode (mode 2)
  4. Extended mode (mode 3)

### 1.3.3 Pin Functions

**Table 1.2 Pin Functions**

Type	Symbol	Pin No.		Name and Function
		TBP-112A	I/O	
Power supply	VCC	G8, G9	Input	Power supply pins. Connect all these pins to the system power supply.
	VCL	E3	Input	Power supply pin. Connect this pin to VCC.
	VSS	F4, G1 E8, D10 C11, B5 C5	Input	Ground pins. Connect all these pins to the system power supply (0 V).
Clock	XTAL	B1	Input	For connection to a crystal resonator. An external clock can be supplied from the EXTAL pin. For an example of crystal resonator connection, see section 26, Clock Pulse Generator.
	EXTAL	C2	Input	
	$\phi$	G3	Output	Supplies the system clock to external devices.
	EXCL	G3	Input	32.768-kHz external clock for subclock should be supplied.
	X1	E1	Input	For connection to a crystal resonator. An external clock can be supplied from the X2 pin. For an example of crystal resonator connection, see section 26, Clock Pulse Generator.
	X2	E2	Input	
Operating mode control	$\overline{\text{MD2}}$	C1	Input	These pins set the operating mode. Inputs at these pins should not be changed during operation.
	MD1	D3		
	MD0	D2		
System control	$\overline{\text{RES}}$	B2	Input	Reset pin. When this pin is low, the chip is reset.
	$\overline{\text{RESO}}$	C3	Output	Outputs a reset signal to an external device.
	$\overline{\text{STBY}}$	E4	Input	When this pin is low, a transition is made to hardware standby mode.
	FWE	A2	Input	Pin for use by flash memory.

Type	Symbol	Pin No.		Name and Function	
		TBP-112A	I/O		
Address bus	A17 to A0	J8, K9 G11, G10 F9, F11 F10, F8 E11, E10 D9, C10 B11, C9 B10, A10 D8, B9	Output	Address output pins	
Data bus	D15 to D8	B6, A6 C6, B7 A7, C7 D7, A8	Input/ Output	Upper bidirectional data bus	
	D7 to D0	J5, H5 L4, K4 K3, H4 L2, K2		Lower bidirectional data bus	
Compact-Flash control	$\overline{\text{CPREG}}$ CPA10 to CPA0	F10 F8, E11 E10, D9 C10, B11 C9, B10 A10, D8 B9	Output	CompactFlash address output pins	
	CPD15 to CPD0	B6, A6 C6, B7 A7, C7 D7, A8 J5, H5 L4, K4 K3, H4 L2, K2		Input/ Output	CompactFlash bidirectional data bus
	$\overline{\text{CPCS2}}$ $\overline{\text{CPCS1}}$	K1 J2		Output	CompactFlash chip select output pins
	$\overline{\text{CPWAIT}}$	G2		Input	CompactFlash wait input pin
	$\overline{\text{CPOE}}$	H3		Output	CompactFlash output enable output pin
	$\overline{\text{CPWE}}$	G4		Output	CompactFlash write enable output pin

Type	Symbol	Pin No.		Name and Function
		TBP-112A	I/O	
Bus control	$\overline{\text{WAIT}}$	G2	Input	Requests insertion of a wait state in the bus cycle when accessing an external 3-state address space.
	$\overline{\text{RD}}$	H3	Output	This pin is low when the external address space is being read from.
	$\overline{\text{HWR}}$	G4	Output	This pin is low when the external address space is being written to, and the upper half of the data bus is enabled.
	$\overline{\text{LWR}}$	J3	Output	This pin is low when the external space is being written to, and the lower half of the data bus is enabled.
	$\overline{\text{AS/IOS}}$	H1	Output	This pin is low when address output on the address bus is valid.
	$\overline{\text{CS256}}$	G2	Output	Indicates that the 256-kbyte area from H'F80000 to H'FBFFFF is accessed.
Interrupts	NMI	D1	Input	Nonmaskable interrupt request input pin
	$\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$	J1, H2 J9, L10 B3, C4 F1, F2 H11, H10 H9, J11 J10, H8 K11, K10	Input	These pins request a maskable interrupt. Selectable to which pin of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ to insert IRQ15 to IRQ0 interrupts.
	$\overline{\text{ExIRQ15}}$ to $\overline{\text{ExIRQ2}}$	A9, C8 D6, A5 A3, B4 D5, A4 H7, L8 J7, K7 L7, H6		
On-chip emulator	$\overline{\text{ETRST}}$	F3	Input	On-chip emulator interface pins
	ETMS	E9	Input	
	ETDO	D11	Output	
	ETDI	L3	Input	
	ETCK	J4	Input	

Type	Symbol	Pin No.	I/O	Name and Function	
		TBP-112A			
PWM timer (PWM)	PW15 to PW0	G11, G10 F9, F11 F10, F8 E11, E10 D9, C10 B11, C9 B10, A10 D8, B9	Output	PWM timer pulse output pins	
14-bit PWM timer (PWMX)	PWX1 PWX0	J1 H2	Output	PWMX (D/A) pulse output pins	
16-bit free running timer (FRT)	FTCI	K2	Input	External event input pin	
	FTOA FTOB	L2 H5	Output Output	Output compare output pins	
	FTIA to FTID	H4, K3 K4, L4	Input	Input capture input pins	
8-bit timer (TMR_0, TMR_1, TMR_X, TMR_Y)	TMO0 TMO1 TMOX TMOY	H8 J10 H10 H11	Output	Waveform output pins with output compare function	
	TMI0 TMI1 TMIX TMIY ExTMI0 ExTMI1 ExTMIX ExTMIY	K10 K11 J11 H9 A5 D6 C8 A9	Input	External event input pins and counter reset input pins. Selectable to which pin of TMI <sub>n</sub> or ExTMI <sub>n</sub> to insert external event and counter reset.	
	Timer connection	VSYNCI HSYNCI CSYNCI VFBACKI HFBACKI	H4 K11 L4 K3 K2	Input	Timer connection synchronization signal input pins
		VSYNCO HSYNCO CLAMPO CBLANK	L2 J10 K4 H5	Output	Timer connection synchronization signal output pins

Type	Symbol	Pin No.		Name and Function
		TBP-112A	I/O	
Serial communication Interface (SCI_0, SCI_1, SCI_2)	TxD0 to TxD2	F2, C4 L10	Output	Transmit data output pins
	RxD0 to RxD2	F1, B3 J9	Input	Receive data input pins
	SCK0 to SCK2	A5, D6 C8	Input/ Output	Clock input/output pins. Output format is NMOS push-pull output.
	SSE0I	K9	Input	Input pin to halt SCI_0
	SSE2I	J8	Input	Input pin to halt SCI_2
SCI with IrDA (SCI)	IrTxD	C4	Output	Encoded data output pin for IrDA
	IrRxD	B3	Input	Encoded data input pin for IrDA
I <sup>2</sup> C bus interface (IIC)	SCL0	A4	Input/ Output	IIC clock input/output pins. These pins can drive a bus directly with the NMOS open drain output.
	SCL1	B4		
	SDA0 SDA1	D5 A3	Input/ Output	
Keyboard control	$\overline{\text{KIN9}}$ to $\overline{\text{KIN0}}$	J8, K9 J5, H5 L4, K4 K3, H4 L2, K2	Input	Keyboard matrix input pins. All pins have a wake-up function. Normally, $\overline{\text{KIN9}}$ to $\overline{\text{KIN0}}$ function as key scan inputs, and P17 to P10 and P27 to P20 function as key scan outputs. Thus, at a maximum of 16 outputs x 8 inputs, 128-key matrix can be configured.
	$\overline{\text{WUE15}}$ to $\overline{\text{WUE8}}$	B6, A6 C6, B7 A7, C7 D7, A8	Input	Wake-up event input pins. Same wake up as key wake up can be performed with various sources.
A/D converter	AN7 to AN2	H7, L8 J7, K7 L7, H6	Input	Analog input pins
	CIN7 to CIN0	J5, H5 L4, K4 K3, H4 L2, K2	Input	Extended A/D conversion input pins
	$\overline{\text{ADTRG}}$	A9	Input	External trigger input pin to start A/D conversion
D/A converter	DA1	H7	Output	Analog output pins
	DA0	L8		



Type	Symbol	Pin No.		Name and Function
		TBP-112A	I/O	
A/D converter  D/A converter	AVCC	K5, J6	Input	Analog power supply pins for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, these pins should be connected to the system power supply. These pins are used with the USB internal driver/receiver power supply, and should therefore be connected to a power supply of $3.3\text{ V} \pm 0.3\text{ V}$ whenever the USB is used.
	AVref	L5	Input	Reference voltage input pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply.
	AVSS	K8, L9	Input	Ground pins for the A/D converter and D/A converter. These pins should be connected to the system power supply (0 V).
Universal serial bus (USB)	USDP	L6	Input/	USB serial data I/O pins
	USDM	K6	Output	
	USEXCL	A9	Input	USB external clock input pin
	DrVCC	K5, J6	Input	These pins should be connected to the internal driver/receiver power supply ( $3.3\text{ V} \pm 0.3\text{ V}$ ).
	DrVSS	K8, L9	Input	These pins should be connected to the internal driver/receiver power supply (0 V).
	SPEED	K2	Output	External driver/receiver connection signals.
	SUSPEND	L2	Output	
	TXENL	H4	Output	These pins are for connection to a driver/receiver compatible with PDIUSBP11A manufactured by Philips Electronics.
	TXDMNS	K3	Output	
	TXDPLS	K4	Output	
	XVERDATA	L4	Input	
DMNS	H5	Input		
DPLS	J5	Input		

Type	Symbol	Pin No.		Name and Function	
		TBP-112A	I/O		
Multimedia card interface (MCIF)	ExMCCLK MCCLK	K10 A8	Output	Common clock output pins for MMC mode*/SPI mode	
	ExMCTxD MCTxD	K11 D7	Output	Command/data output pins in SPI mode	
	ExMCRxD MCRxD	H8 C7	Input	Response/data input pins in SPI mode	
	ExMCCSA ExMCCSB MCCSA MCCSB	J10 J11 A7 B7	Output	Chip select output pins to select multimedia card in SPI mode	
	ExMCCMD MCCMD	K11 D7	Input/ Output	Command output/response input pins in MMC mode*	
	ExMCDAT MCDAT	H8 C7	Input/ Output	Data I/O pins in MMC mode*	
	ExMCDATDIR ExMCCMDDIR MCDATDIR MCCMDDIR	J10 J11 A7 B7	Output	Output pins indicating I/O direction of MCCMD and MCDAT pins	
	I/O ports	P17 to P10	D9, C10 B11, C9 B10, A10 D8, B9	Input/ Output	Eight input/output pins
		P27 to P20	G11, G10 F9, F11 F10, F8 E11, E10	Input/ Output	Eight input/output pins
		P37 to P30	B6, A6 C6, B7 A7, C7 D7, A8	Input/ Output	Eight input/output pins
P47 to P40		H11, H10 H9, J11 J10, H8 K11, K10	Input/ Output	Eight input/output pins	
P57 to P50		J1, H2 J9, L10 B3, C4 F1, F2	Input/ Output	Eight input/output pins	

Type	Symbol	Pin No.		Name and Function
		TBP-112A	I/O	
I/O ports	P67 to P60	J5, H5 L4, K4 K3, H4 L2, K2	Input/ Output	Eight input/output pins
	P77 to P72	H7, L8 J7, K7 L7, H6	Input	Six input pins
	P87 to P80	A9, C8 D6, A5 A3, B4 D5, A4	Input/ Output	Eight input/output pins
	P97 to P90	G2, G3 H1, G4 H3, J2 K1, J3	Input/ Output	Eight input/output pins. Note that pin P96 cannot be used as a general output port.
	PA1, PA0	J8, K9	Input/ Output	Two input/output pins

Note: \* MMC mode is MultiMediaCard mode.



## Section 2 CPU

The H8S/2000 CPU is a high-speed central processing unit with an internal 32-bit architecture that is upward-compatible with the H8/300 and H8/300H CPUs. The H8S/2000 CPU has sixteen 16-bit general registers, can address a 16-Mbyte linear address space, and is ideal for realtime control.

This section describes the H8S/2000 CPU. The usable modes and address spaces differ depending on the product. For details on each product, see section 3, MCU Operating Modes.

### 2.1 Features

- Upward-compatibility with H8/300 and H8/300H CPUs
  - Can execute H8/300 CPU and H8/300H CPU object programs
- General-register architecture
  - Sixteen 16-bit general registers also usable as sixteen 8-bit registers or eight 32-bit registers
- Sixty-five basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
- Eight addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
  - Register indirect with post-increment or pre-decrement [@ERn+ or @-ERn]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Memory indirect [@@aa:8]
- 16-Mbyte address space
  - Program: 16 Mbytes
  - Data: 16 Mbytes
- High-speed operation
  - All frequently-used instructions are executed in one or two states
  - 8/16/32-bit register-register add/subtract: 1 state
  - $8 \times 8$ -bit register-register multiply: 12 states (MULXU.B), 13 states (MULXS.B)
  - $16 \div 8$ -bit register-register divide: 12 states (DIVXU.B)

- 16 × 16-bit register-register multiply: 20 states (MULXU.W), 21 states (MULXS.W)
- 32 ÷ 16-bit register-register divide: 20 states (DIVXU.W)
- Two CPU operating modes
  - Normal mode
  - Advanced mode
- Power-down state
  - Transition to power-down state by SLEEP instruction
  - Selectable CPU clock speed

### 2.1.1 Differences between H8S/2600 CPU and H8S/2000 CPU

The differences between the H8S/2600 CPU and the H8S/2000 CPU are as shown below.

- Register configuration
  - The MAC register is supported only by the H8S/2600 CPU.
- Basic instructions
  - The four instructions MAC, CLRMAC, LDMAC, and STMAC are supported only by the H8S/2600 CPU.
- The number of execution states of the MULXU and MULXS instructions

Instruction	Mnemonic	Execution States	
		H8S/2600	H8S/2000
MULXU	MULXU.B Rs, Rd	3	12
	MULXU.W Rs, ERd	4	20
MULXS	MULXS.B Rs, Rd	4	13
	MULXS.W Rs, ERd	5	21

In addition, there are differences in address space, CCR and EXR register functions, power-down modes, etc., depending on the model.

### 2.1.2 Differences from H8/300 CPU

In comparison to the H8/300 CPU, the H8S/2000 CPU has the following enhancements.

- More general registers and control registers
  - Eight 16-bit extended registers and one 8-bit control register have been added.
- Extended address space
  - Normal mode supports the same 64-kbyte address space as the H8/300 CPU.
  - Advanced mode supports a maximum 16-Mbyte address space.
- Enhanced addressing
  - The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Signed multiply and divide instructions have been added.
  - Two-bit shift and two-bit rotate instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions are executed twice as fast.

### 2.1.3 Differences from H8/300H CPU

In comparison to the H8/300H CPU, the H8S/2000 CPU has the following enhancements.

- Additional control register
  - One 8-bit control register has been added.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Two-bit shift and two-bit rotate instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions are executed twice as fast.

## 2.2 CPU Operating Modes

The H8S/2000 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte address space. The mode is selected by the LSI's mode pins.

### 2.2.1 Normal Mode

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

- Address space

Linear access to a maximum address space of 64 kbytes is possible.

- 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 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 general register Rn is referenced in the register indirect addressing mode with pre-decrement (@-Rn) or post-increment (@Rn+) and a carry or borrow occurs, the value in the corresponding extended register (En) will be affected.)

- Instruction set

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

- Exception vector table and memory indirect branch addresses

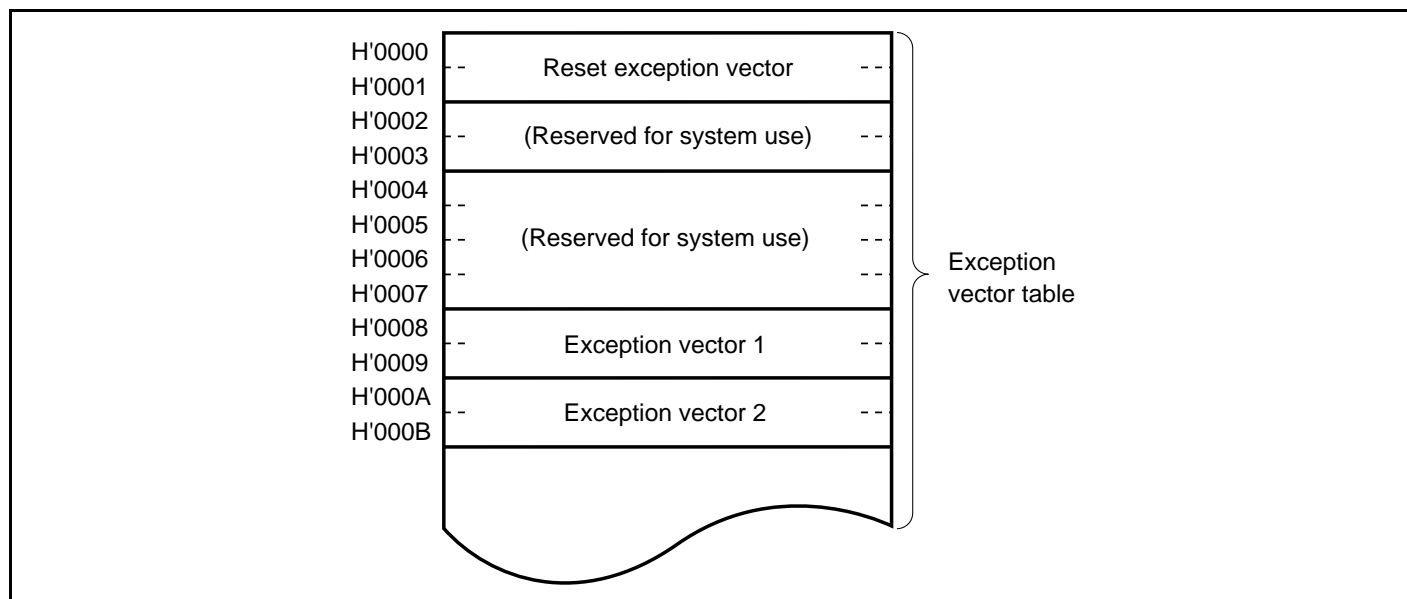
In normal mode, the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The exception vector table in normal mode is shown in figure 2.1. For details of the exception vector table, see section 4, Exception Handling.

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In normal mode, the operand is a 16-bit (word) operand, providing a 16-bit branch address. Branch addresses can be stored in the top area from H'0000 to H'00FF. Note that this area is also used for the exception vector table.

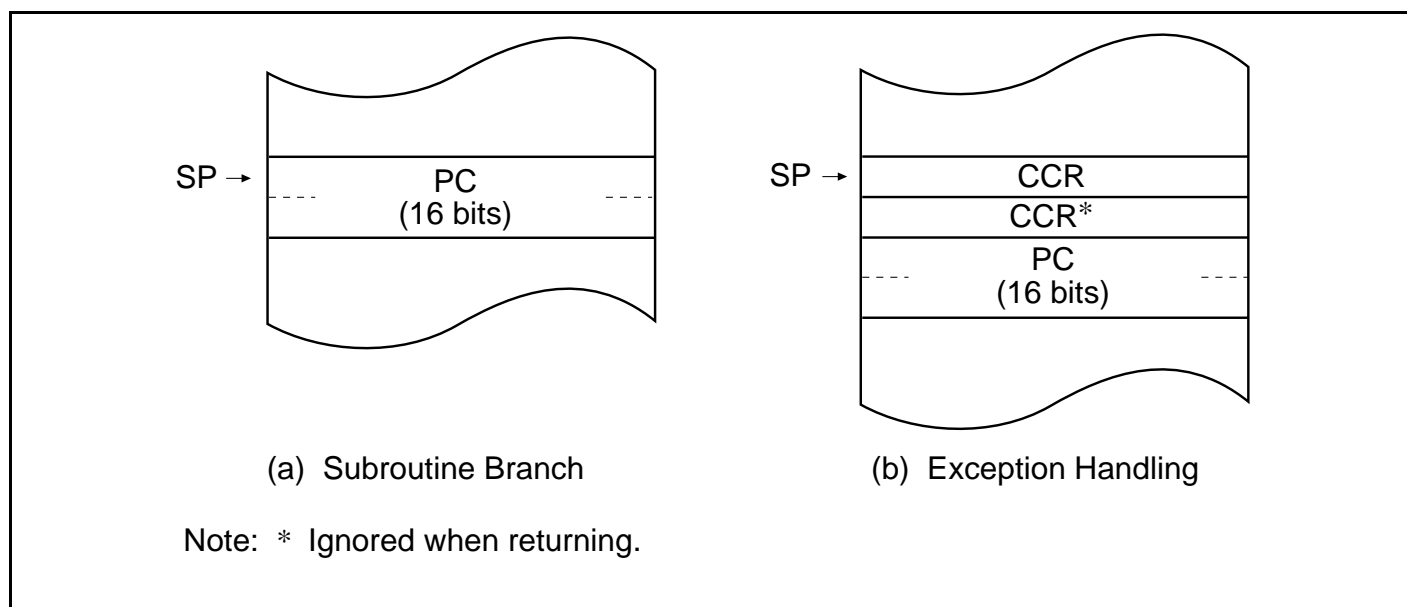
- Stack structure

In normal mode, when the program counter (PC) is pushed onto the stack in a subroutine call in normal mode, and the PC and condition-code register (CCR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.2. The extended control register (EXR) is not pushed onto the stack. For details, see section 4, Exception Handling.





**Figure 2.1 Exception Vector Table (Normal Mode)**



**Figure 2.2 Stack Structure in Normal Mode**

## 2.2.2 Advanced Mode

- Address space

Linear access to a maximum address space of 16 Mbytes is possible.

- Extended registers (En)

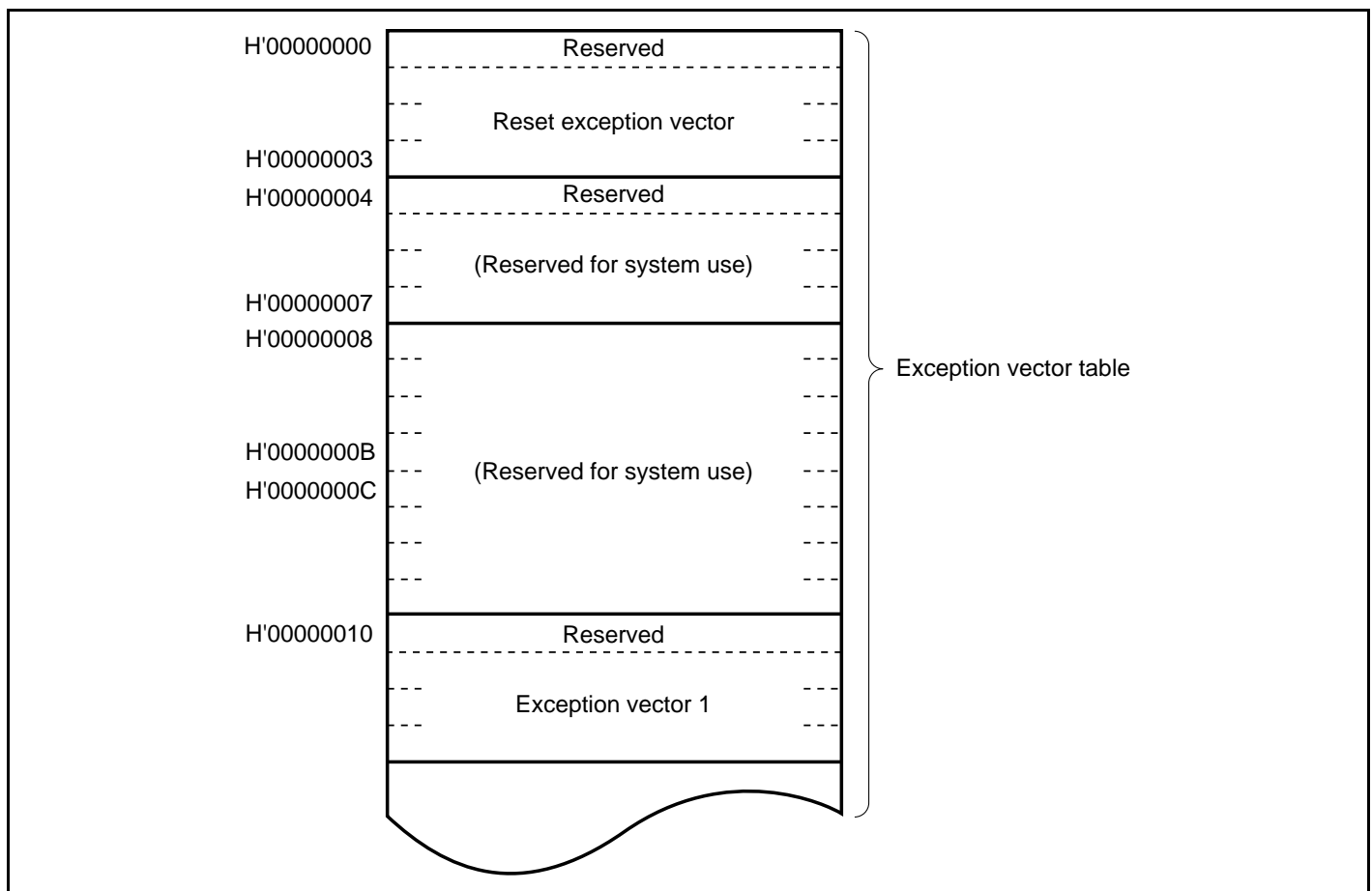
The extended registers (E0 to E7) can be used as 16-bit registers. They can also be used as the upper 16-bit segments of 32-bit registers or address registers.

- Instruction set

All instructions and addressing modes can be used.

- Exception vector table and memory indirect branch addresses

In advanced mode, the top area starting at H'00000000 is allocated to the exception vector table in 32-bit units. In each 32 bits, the upper 8 bits are ignored and a branch address is stored in the lower 24 bits (see figure 2.3). For details of the exception vector table, see section 4, Exception Handling.

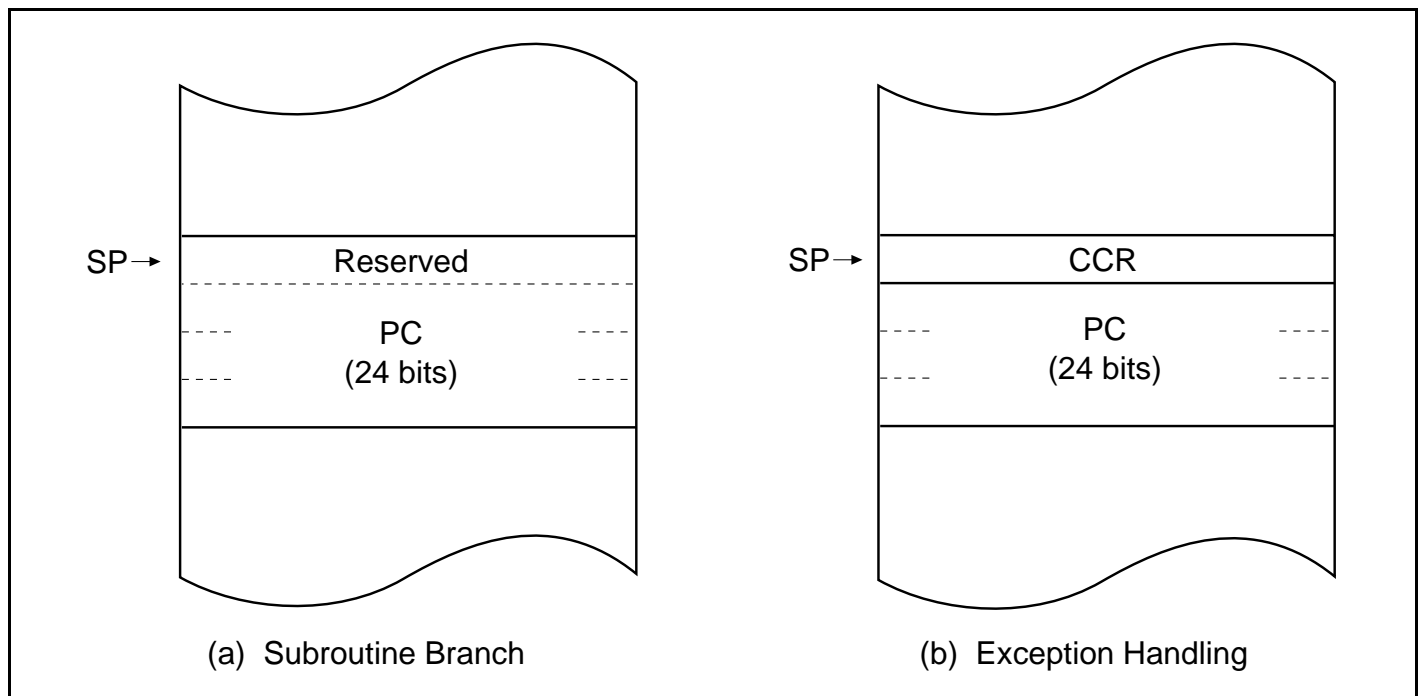


**Figure 2.3 Exception Vector Table (Advanced Mode)**

The memory indirect addressing mode ( $@@aa:8$ ) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In advanced mode, the operand is a 32-bit longword operand, providing a 32-bit branch address. The upper 8 bits of these 32 bits are a reserved area that is regarded as H'00. Branch addresses can be stored in the area from H'00000000 to H'000000FF. Note that the top area of this range is also used for the exception vector table.

- Stack structure

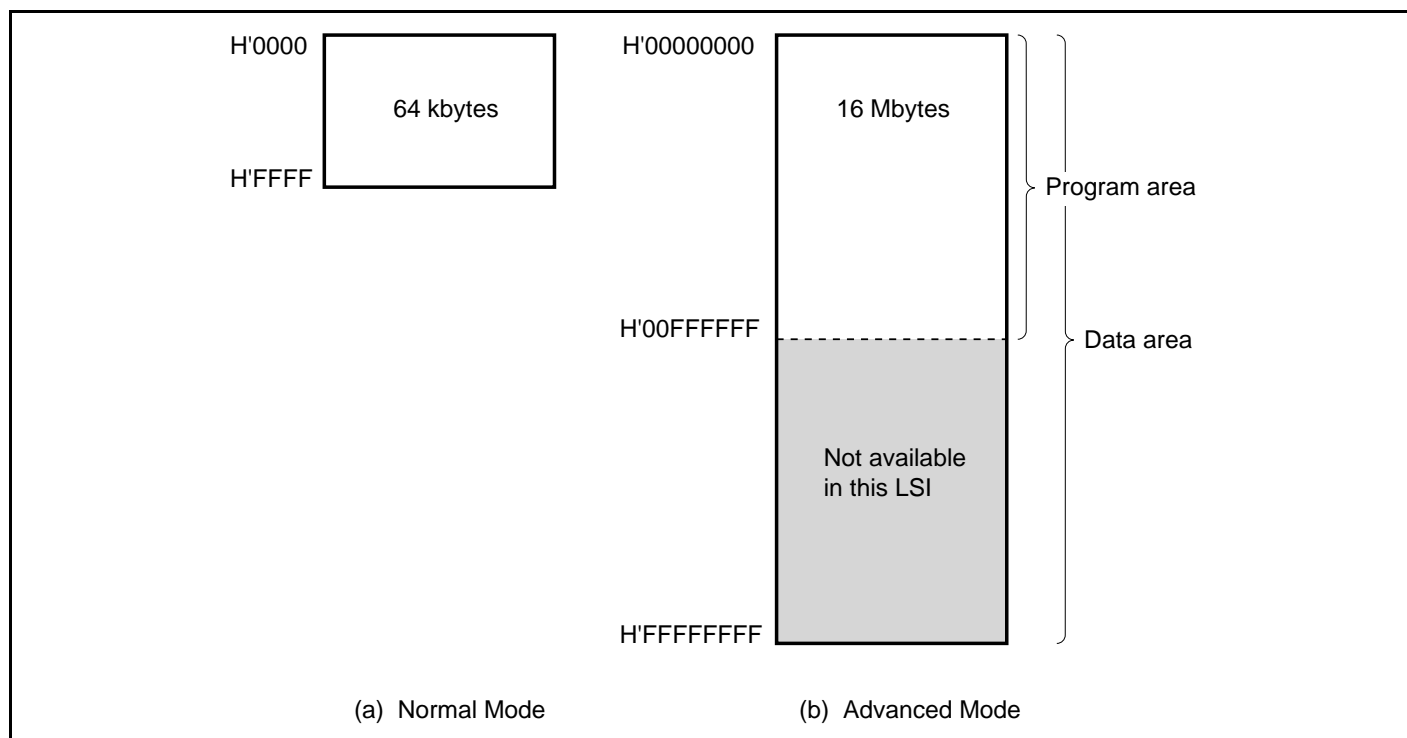
In advanced mode, when the program counter (PC) is pushed onto the stack in a subroutine call, and the PC and condition-code register (CCR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.4. The extended control register (EXR) is not pushed onto the stack. For details, see section 4, Exception Handling.



**Figure 2.4 Stack Structure in Advanced Mode**

## 2.3 Address Space

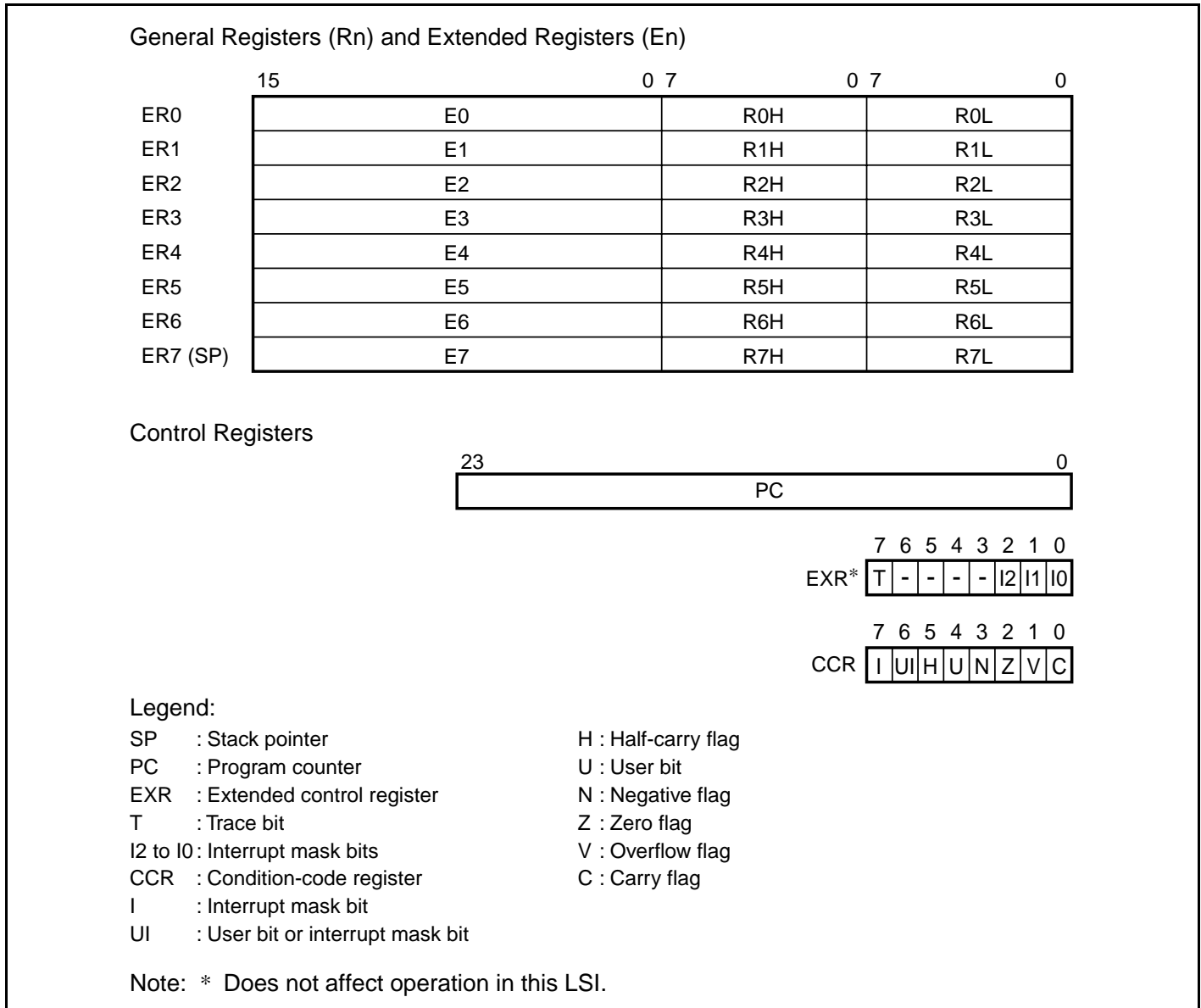
Figure 2.5 shows a memory map of the H8S/2000 CPU. The H8S/2000 CPU provides linear access to a maximum 64-kbyte address space in normal mode, and a maximum 16-Mbyte (architecturally 4-Gbyte) address space in advanced mode. The usable modes and address spaces differ depending on the product. For details on each product, see section 3, MCU Operating Modes.



**Figure 2.5 Memory Map**

## 2.4 Register Configuration

The H8S/2000 CPU has the internal registers shown in figure 2.6. There are two types of registers: general registers and control registers. Control registers are a 24-bit program counter (PC), an 8-bit extended control register (EXR), and an 8-bit condition code register (CCR).



**Figure 2.6 CPU Internal Registers**

### 2.4.1 General Registers

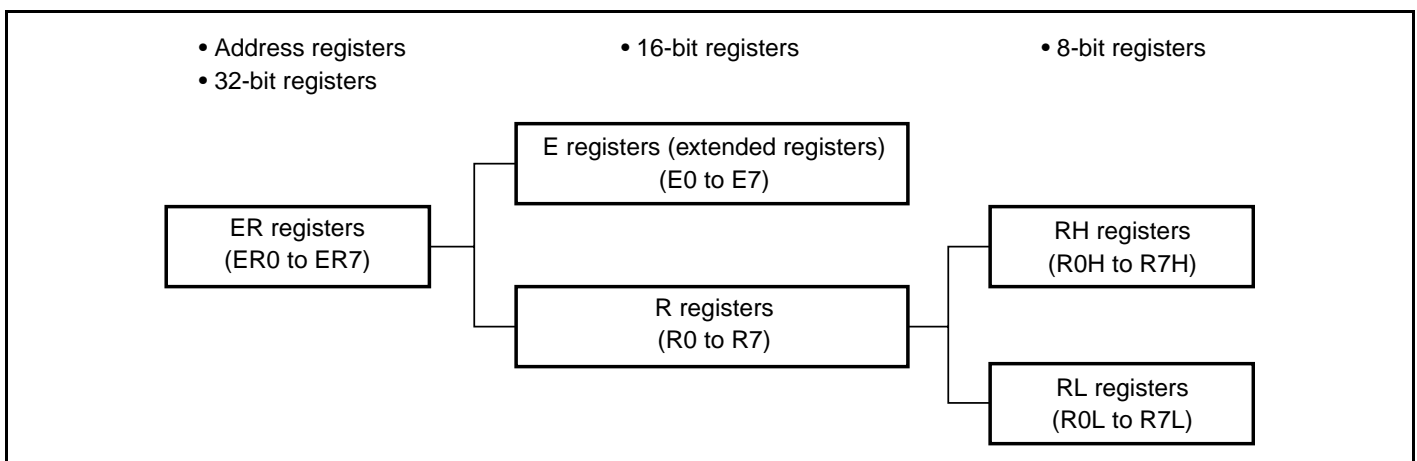
The H8S/2000 CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.7 illustrates the usage of the general registers. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

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

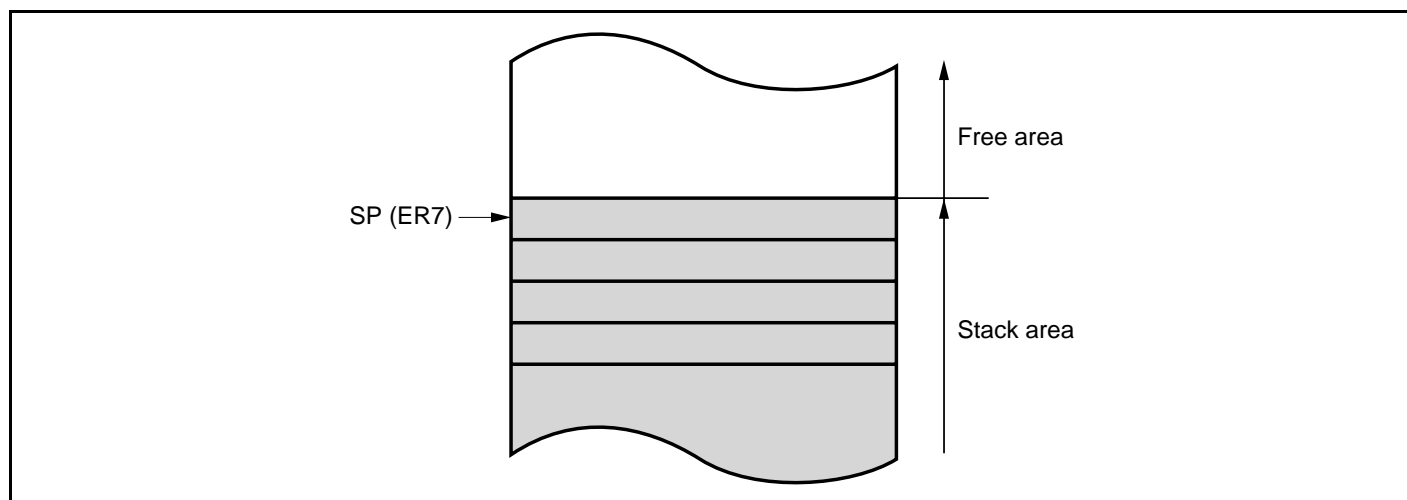
When the general registers are used as 8-bit registers, the R registers are divided into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

The usage of each register can be selected independently.

General register ER7 has the function of the stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2.8 shows the stack.



**Figure 2.7 Usage of General Registers**



**Figure 2.8 Stack**

### 2.4.2 Program Counter (PC)

This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 2 bytes (one word), so the least significant PC bit is ignored. (When an instruction is fetched for read, the least significant PC bit is regarded as 0.)

### 2.4.3 Extended Control Register (EXR)

EXR does not affect operation in this LSI.

Bit	Bit Name	Initial Value	R/W	Description
7	T	0	R/W	Trace Bit Does not affect operation in this LSI.
6 to 3	—	All 1	R	Reserved These bits are always read as 1.
2	I2	1	R/W	Interrupt Mask Bits 2 to 0
1	I1	1		Do not affect operation in this LSI.
0	I0	1		

## 2.4.4 Condition-Code Register (CCR)

This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

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

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	<p>Interrupt Mask Bit</p> <p>Masks interrupts other than NMI when set to 1. NMI is accepted regardless of the I bit setting. The I bit is set to 1 at the start of an exception-handling sequence. For details, see section 5, Interrupt Controller.</p>
6	UI	Undefined	R/W	<p>User Bit or Interrupt Mask Bit</p> <p>Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.</p>
5	H	Undefined	R/W	<p>Half-Carry Flag</p> <p>When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.</p>
4	U	Undefined	R/W	<p>User Bit</p> <p>Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.</p>
3	N	Undefined	R/W	<p>Negative Flag</p> <p>Stores the value of the most significant bit of data as a sign bit.</p>
2	Z	Undefined	R/W	<p>Zero Flag</p> <p>Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.</p>



Bit	Bit Name	Initial Value	R/W	Description
1	V	Undefined	R/W	<p>Overflow Flag</p> <p>Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.</p>
0	C	Undefined	R/W	<p>Carry Flag</p> <p>Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:</p> <ul style="list-style-type: none"> <li>• Add instructions, to indicate a carry</li> <li>• Subtract instructions, to indicate a borrow</li> <li>• Shift and rotate instructions, to indicate a carry</li> </ul> <p>The carry flag is also used as a bit accumulator by bit manipulation instructions.</p>

### 2.4.5 Initial Register Values

Reset exception handling loads the CPU's program counter (PC) from the vector table, clears the trace (T) bit in EXR to 0, and sets the interrupt mask (I) bits in CCR and EXR to 1. The other CCR bits and the general registers are not initialized. Note that the stack pointer (ER7) is undefined. The stack pointer should therefore be initialized by an MOV.L instruction executed immediately after a reset.

## 2.5 Data Formats

The H8S/2000 CPU can process 1-bit, 4-bit BCD, 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit  $n$  ( $n = 0, 1, 2, \dots, 7$ ) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

### 2.5.1 General Register Data Formats

Figure 2.9 shows the data formats of general registers.

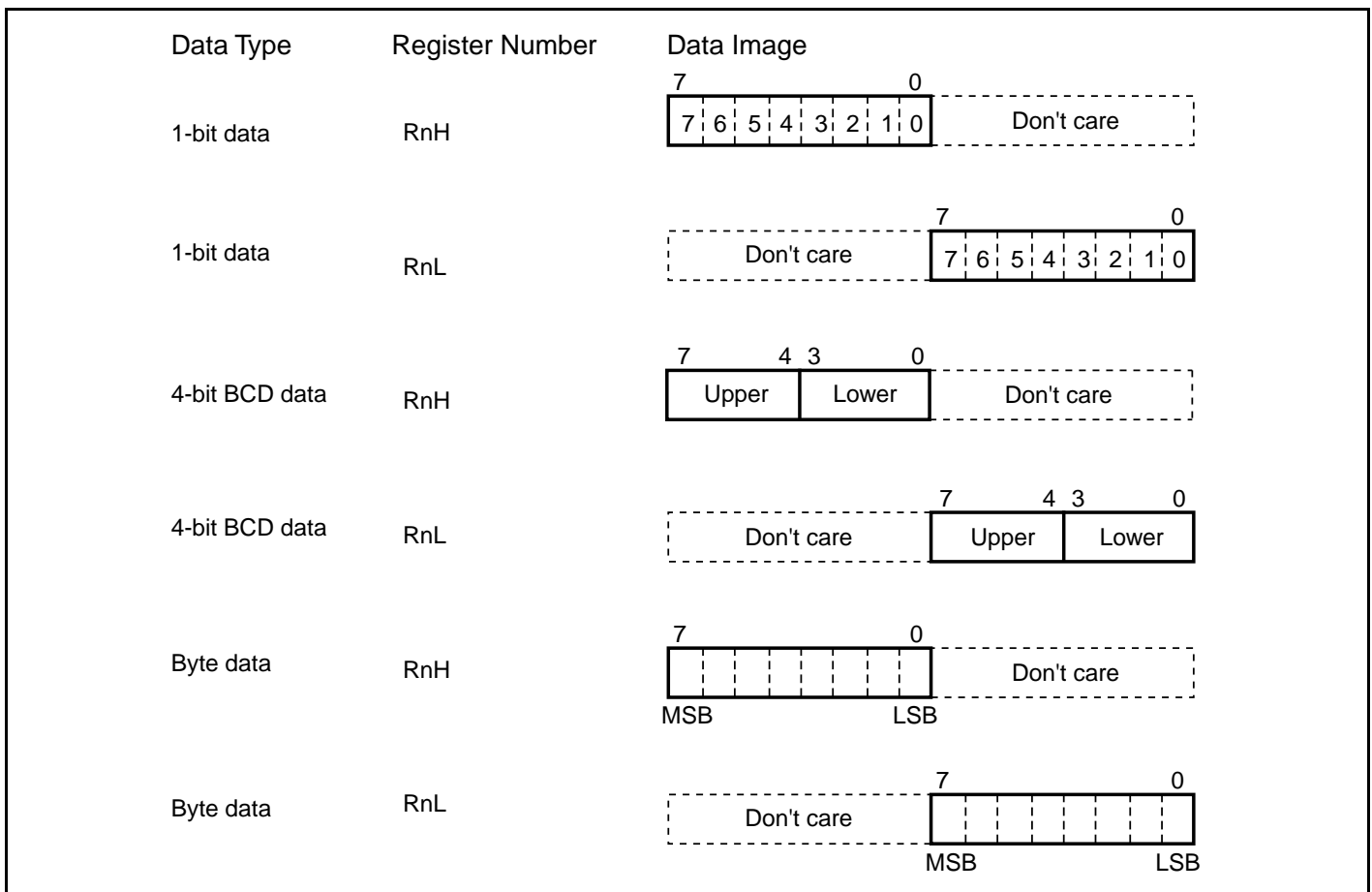
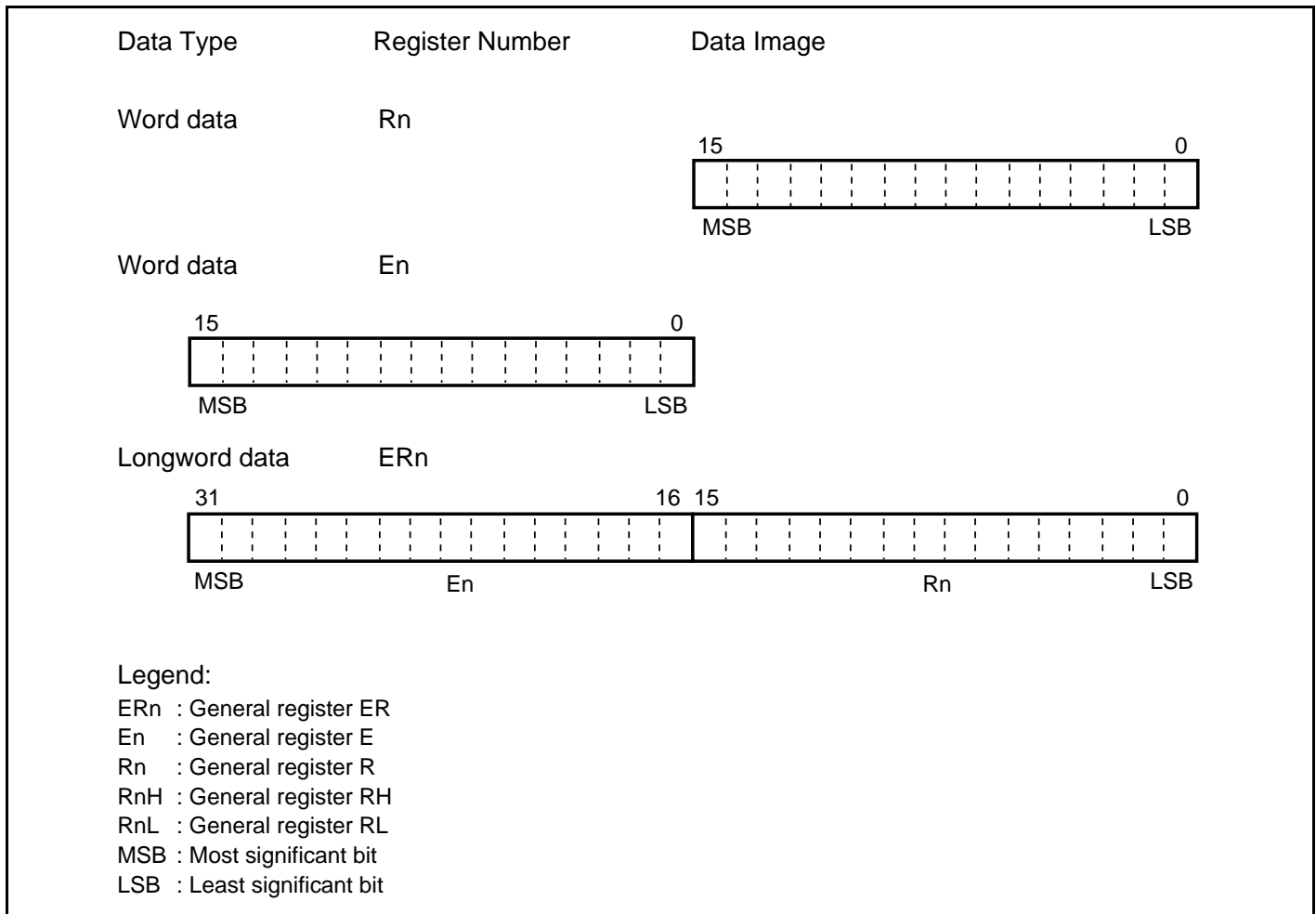


Figure 2.9 General Register Data Formats (1)

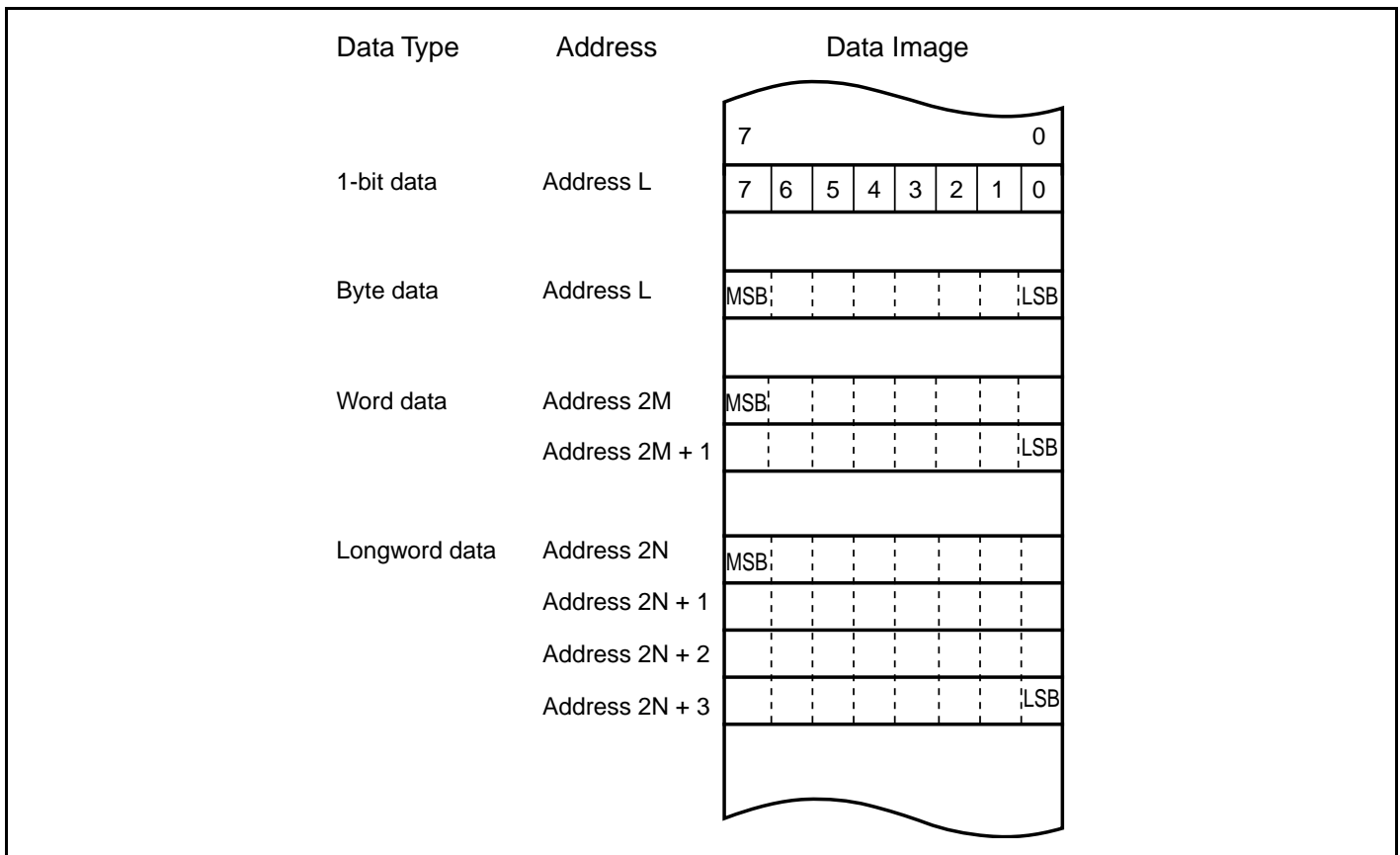


**Figure 2.9 General Register Data Formats (2)**

## 2.5.2 Memory Data Formats

Figure 2.10 shows the data formats in memory. The H8S/2000 CPU can access word data and longword data in memory, but word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, no address error occurs but the least significant bit of the address is regarded as 0, so the access starts at the preceding address. This also applies to instruction fetches.

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



**Figure 2.10 Memory Data Formats**

## 2.6 Instruction Set

The H8S/2000 CPU has 65 types of instructions. The instructions are classified by function as shown in table 2.1.

**Table 2.1 Instruction Classification**

Function	Instructions	Size	Types
Data transfer	MOV	B/W/L	5
	POP <sup>*1</sup> , PUSH <sup>*1</sup>	W/L	
	LDM, STM <sup>*2</sup>	L	
	MOVFPPE <sup>*3</sup> , MOVTPPE <sup>*3</sup>	B	
Arithmetic operations	ADD, SUB, CMP, NEG	B/W/L	19
	ADDX, SUBX, DAA, DAS	B	
	INC, DEC	B/W/L	
	ADDS, SUBS	L	
	MULXU, DIVXU, MULXS, DIVXS	B/W	
	EXTU, EXTS	W/L	
	TAS	B	
Logic operations	AND, OR, XOR, NOT	B/W/L	4
Shift	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	B/W/L	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BLD, BILD, BST, BIST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR	B	14
Branch	B <sub>CC</sub> <sup>*4</sup> , JMP, BSR, JSR, RTS	—	5
System control	TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	—	9
Block data transfer	EPEMOV	—	1

Total: 65

Legend: B: Byte size  
W: Word size  
L: Longword size

- Notes: 1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @-SP. POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @-SP.
2. Since register ER7 functions as the stack pointer in an STM/LDM instruction, it cannot be used as an STM/LDM register.
3. Cannot be used in this LSI.
4. B<sub>CC</sub> is the general name for conditional branch instructions.

## 2.6.1 Table of Instructions Classified by Function

Tables 2.3 to 2.10 summarize the instructions in each functional category. The notation used in tables 2.3 to 2.10 is defined below.

**Table 2.2 Operation Notation**

Symbol	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extended control register
CCR	Condition-code register
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
~	NOT (logical complement)
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

Note: \* General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

**Table 2.3 Data Transfer Instructions**

<b>Instruction</b>	<b>Size<sup>*1</sup></b>	<b>Function</b>
MOV	B/W/L	(EAs) → Rd, Rs → (EAd) Moves data between two general registers or between a general register and memory, or moves immediate data to a general register.
MOVFPPE	B	Cannot be used in this LSI.
MOVTPE	B	Cannot be used in this LSI.
POP	W/L	@SP+ → Rn Pops a general register from the stack. POP.W Rn is identical to MOV.W @SP+, Rn. POP.L ERn is identical to MOV.L @SP+, ERn
PUSH	W/L	Rn → @-SP Pushes a general register onto the stack. PUSH.W Rn is identical to MOV.W Rn, @-SP. PUSH.L ERn is identical to MOV.L ERn, @-SP.
LDM <sup>*2</sup>	L	@SP+ → Rn (register list) Pops two or more general registers from the stack.
STM <sup>*2</sup>	L	Rn (register list) → @-SP Pushes two or more general registers onto the stack.

Notes: 1. Size refers to the operand size.

B: Byte

W: Word

L: Longword

2. Since register ER7 functions as the stack pointer in an STM/LDM instruction, it cannot be used as an STM/LDM register.

**Table 2.4 Arithmetic Operations Instructions (1)**

<b>Instruction</b>	<b>Size*</b>	<b>Function</b>
ADD	B/W/L	$Rd \pm Rs \rightarrow Rd, Rd \pm \#IMM \rightarrow Rd$
SUB		Performs addition or subtraction on data in two general registers, or on immediate data and data in a general register. (Subtraction on immediate data and data in a general register cannot be performed in bytes. Use the SUBX or ADD instruction.)
ADDX	B	$Rd \pm Rs \pm C \rightarrow Rd, Rd \pm \#IMM \pm C \rightarrow Rd$
SUBX		Performs addition or subtraction with carry on data in two general registers, or on immediate data and data in a general register.
INC	B/W/L	$Rd \pm 1 \rightarrow Rd, Rd \pm 2 \rightarrow Rd$
DEC		Adds or subtracts the value 1 or 2 to or from data in a general register. (Only the value 1 can be added to or subtracted from byte operands.)
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 32-bit register.
DAA	B	$Rd$ (decimal adjust) $\rightarrow Rd$
DAS		Decimal-adjusts an addition or subtraction result in a general register by referring to CCR to produce 4-bit BCD data.
MULXU	B/W	$Rd \times Rs \rightarrow Rd$ Performs unsigned multiplication on data in two general registers: either 8 bits $\times$ 8 bits $\rightarrow$ 16 bits or 16 bits $\times$ 16 bits $\rightarrow$ 32 bits.
MULXS	B/W	$Rd \times Rs \rightarrow Rd$ Performs signed multiplication on data in two general registers: either 8 bits $\times$ 8 bits $\rightarrow$ 16 bits or 16 bits $\times$ 16 bits $\rightarrow$ 32 bits.
DIVXU	B/W	$Rd \div Rs \rightarrow Rd$ Performs unsigned division on data in two general registers: either 16 bits $\div$ 8 bits $\rightarrow$ 8-bit quotient and 8-bit remainder or 32 bits $\div$ 16 bits $\rightarrow$ 16-bit quotient and 16-bit remainder.

Note: \* Size refers to the operand size.

B: Byte

W: Word

L: Longword



**Table 2.4 Arithmetic Operations Instructions (2)**

<b>Instruction</b>	<b>Size*</b>	<b>Function</b>
DIVXS	B/W	$Rd \div Rs \rightarrow Rd$ Performs signed division on data in two general registers: either 16 bits $\div$ 8 bits $\rightarrow$ 8-bit quotient and 8-bit remainder or 32 bits $\div$ 16 bits $\rightarrow$ 16-bit quotient and 16-bit remainder.
CMP	B/W/L	$Rd - Rs, Rd - \#IMM$ Compares data in a general register with data in another general register or with immediate data, and sets the CCR bits according to the result.
NEG	B/W/L	$0 - Rd \rightarrow Rd$ Takes the two's complement (arithmetic complement) of data in a general register.
EXTU	W/L	$Rd$ (zero extension) $\rightarrow Rd$ Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by padding with zeros on the left.
EXTS	W/L	$Rd$ (sign extension) $\rightarrow Rd$ Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by extending the sign bit.
TAS	B	$@ERd - 0, 1 \rightarrow (<bit\ 7> \text{ of } @ERd)$ Tests memory contents, and sets the most significant bit (bit 7) to 1.

Note: \* Size refers to the operand size.

B: Byte

W: Word

L: Longword

**Table 2.5 Logic Operations Instructions**

<b>Instruction</b>	<b>Size*</b>	<b>Function</b>
AND	B/W/L	$Rd \wedge Rs \rightarrow Rd, Rd \wedge \#IMM \rightarrow Rd$ Performs a logical AND operation on a general register and another general register or immediate data.
OR	B/W/L	$Rd \vee Rs \rightarrow Rd, Rd \vee \#IMM \rightarrow Rd$ Performs a logical OR operation on a general register and another general register or immediate data.
XOR	B/W/L	$Rd \oplus Rs \rightarrow Rd, Rd \oplus \#IMM \rightarrow Rd$ Performs a logical exclusive OR operation on a general register and another general register or immediate data.
NOT	B/W/L	$\sim Rd \rightarrow Rd$ Takes the one's complement (logical complement) of data in a general register.

Note: \* Size refers to the operand size.

B: Byte

W: Word

L: Longword

**Table 2.6 Shift Instructions**

<b>Instruction</b>	<b>Size*</b>	<b>Function</b>
SHAL	B/W/L	$Rd \text{ (shift)} \rightarrow Rd$
SHAR		Performs an arithmetic shift on data in a general register. 1-bit or 2 bit shift is possible.
SHLL	B/W/L	$Rd \text{ (shift)} \rightarrow Rd$
SHLR		Performs a logical shift on data in a general register. 1-bit or 2 bit shift is possible.
ROTL	B/W/L	$Rd \text{ (rotate)} \rightarrow Rd$
ROTR		Rotates data in a general register. 1-bit or 2 bit rotation is possible.
ROTXL	B/W/L	$Rd \text{ (rotate)} \rightarrow Rd$
ROTXR		Rotates data including the carry flag in a general register. 1-bit or 2 bit rotation is possible.

Note: \* Size refers to the operand size.

B: Byte

W: Word

L: Longword

**Table 2.7 Bit Manipulation Instructions (1)**

<b>Instruction</b>	<b>Size*</b>	<b>Function</b>
BSET	B	$1 \rightarrow (\text{<bit-No.> of <EAd>})$ Sets a specified bit in a general register or memory operand to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BCLR	B	$0 \rightarrow (\text{<bit-No.> of <EAd>})$ Clears a specified bit in a general register or memory operand to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BNOT	B	$\sim (\text{<bit-No.> of <EAd>}) \rightarrow (\text{<bit-No.> of <EAd>})$ Inverts a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BTST	B	$\sim (\text{<bit-No.> of <EAd>}) \rightarrow Z$ Tests a specified bit in a general register or memory operand and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BAND	B	$C \wedge (\text{<bit-No.> of <EAd>}) \rightarrow C$ Logically ANDs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIAND	B	$C \wedge (\text{<bit-No.> of <EAd>}) \rightarrow C$ Logically ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BOR	B	$C \vee (\text{<bit-No.> of <EAd>}) \rightarrow C$ Logically ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIOR	B	$C \vee (\sim \text{<bit-No.> of <EAd>}) \rightarrow C$ Logically ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.

Note: \* Size refers to the operand size.

B: Byte

**Table 2.7 Bit Manipulation Instructions (2)**

<b>Instruction</b>	<b>Size*</b>	<b>Function</b>
BXOR	B	$C \oplus (\text{<bit-No.> of <EAd>}) \rightarrow C$ Logically exclusive-ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIXOR	B	$C \oplus \sim (\text{<bit-No.> of <EAd>}) \rightarrow C$ Logically exclusive-ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BLD	B	$(\text{<bit-No.> of <EAd>}) \rightarrow C$ Transfers a specified bit in a general register or memory operand to the carry flag.
BILD	B	$\sim (\text{<bit-No.> of <EAd>}) \rightarrow C$ Transfers the inverse of a specified bit in a general register or memory operand to the carry flag. The bit number is specified by 3-bit immediate data.
BST	B	$C \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the carry flag value to a specified bit in a general register or memory operand.
BIST	B	$\sim C \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data.

Note: \* Size refers to the operand size.

B: Byte

**Table 2.8 Branch Instructions**

<b>Instruction</b>	<b>Size</b>	<b>Function</b>																																																			
Bcc	—	Branches to a specified address if a specified condition is true. The branching conditions are listed below.																																																			
		<table border="1"> <thead> <tr> <th><b>Mnemonic</b></th> <th><b>Description</b></th> <th><b>Condition</b></th> </tr> </thead> <tbody> <tr> <td>BRA (BT)</td> <td>Always (true)</td> <td>Always</td> </tr> <tr> <td>BRN (BF)</td> <td>Never (false)</td> <td>Never</td> </tr> <tr> <td>BHI</td> <td>High</td> <td><math>C \vee Z = 0</math></td> </tr> <tr> <td>BLS</td> <td>Low or same</td> <td><math>C \vee Z = 1</math></td> </tr> <tr> <td>BCC (BHS)</td> <td>Carry clear (high or same)</td> <td><math>C = 0</math></td> </tr> <tr> <td>BCS (BLO)</td> <td>Carry set (low)</td> <td><math>C = 1</math></td> </tr> <tr> <td>BNE</td> <td>Not equal</td> <td><math>Z = 0</math></td> </tr> <tr> <td>BEQ</td> <td>Equal</td> <td><math>Z = 1</math></td> </tr> <tr> <td>BVC</td> <td>Overflow clear</td> <td><math>V = 0</math></td> </tr> <tr> <td>BVS</td> <td>Overflow set</td> <td><math>V = 1</math></td> </tr> <tr> <td>BPL</td> <td>Plus</td> <td><math>N = 0</math></td> </tr> <tr> <td>BMI</td> <td>Minus</td> <td><math>N = 1</math></td> </tr> <tr> <td>BGE</td> <td>Greater or equal</td> <td><math>N \oplus V = 0</math></td> </tr> <tr> <td>BLT</td> <td>Less than</td> <td><math>N \oplus V = 1</math></td> </tr> <tr> <td>BGT</td> <td>Greater than</td> <td><math>Z \vee (N \oplus V) = 0</math></td> </tr> <tr> <td>BLE</td> <td>Less or equal</td> <td><math>Z \vee (N \oplus V) = 1</math></td> </tr> </tbody> </table>	<b>Mnemonic</b>	<b>Description</b>	<b>Condition</b>	BRA (BT)	Always (true)	Always	BRN (BF)	Never (false)	Never	BHI	High	$C \vee Z = 0$	BLS	Low or same	$C \vee Z = 1$	BCC (BHS)	Carry clear (high or same)	$C = 0$	BCS (BLO)	Carry set (low)	$C = 1$	BNE	Not equal	$Z = 0$	BEQ	Equal	$Z = 1$	BVC	Overflow clear	$V = 0$	BVS	Overflow set	$V = 1$	BPL	Plus	$N = 0$	BMI	Minus	$N = 1$	BGE	Greater or equal	$N \oplus V = 0$	BLT	Less than	$N \oplus V = 1$	BGT	Greater than	$Z \vee (N \oplus V) = 0$	BLE	Less or equal	$Z \vee (N \oplus V) = 1$
<b>Mnemonic</b>	<b>Description</b>	<b>Condition</b>																																																			
BRA (BT)	Always (true)	Always																																																			
BRN (BF)	Never (false)	Never																																																			
BHI	High	$C \vee Z = 0$																																																			
BLS	Low or same	$C \vee Z = 1$																																																			
BCC (BHS)	Carry clear (high or same)	$C = 0$																																																			
BCS (BLO)	Carry set (low)	$C = 1$																																																			
BNE	Not equal	$Z = 0$																																																			
BEQ	Equal	$Z = 1$																																																			
BVC	Overflow clear	$V = 0$																																																			
BVS	Overflow set	$V = 1$																																																			
BPL	Plus	$N = 0$																																																			
BMI	Minus	$N = 1$																																																			
BGE	Greater or equal	$N \oplus V = 0$																																																			
BLT	Less than	$N \oplus V = 1$																																																			
BGT	Greater than	$Z \vee (N \oplus V) = 0$																																																			
BLE	Less or equal	$Z \vee (N \oplus V) = 1$																																																			
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																																																			

**Table 2.9 System Control Instructions**

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

Note: \* Size refers to the operand size.

B: Byte

W: Word

**Table 2.10 Block Data Transfer Instructions**

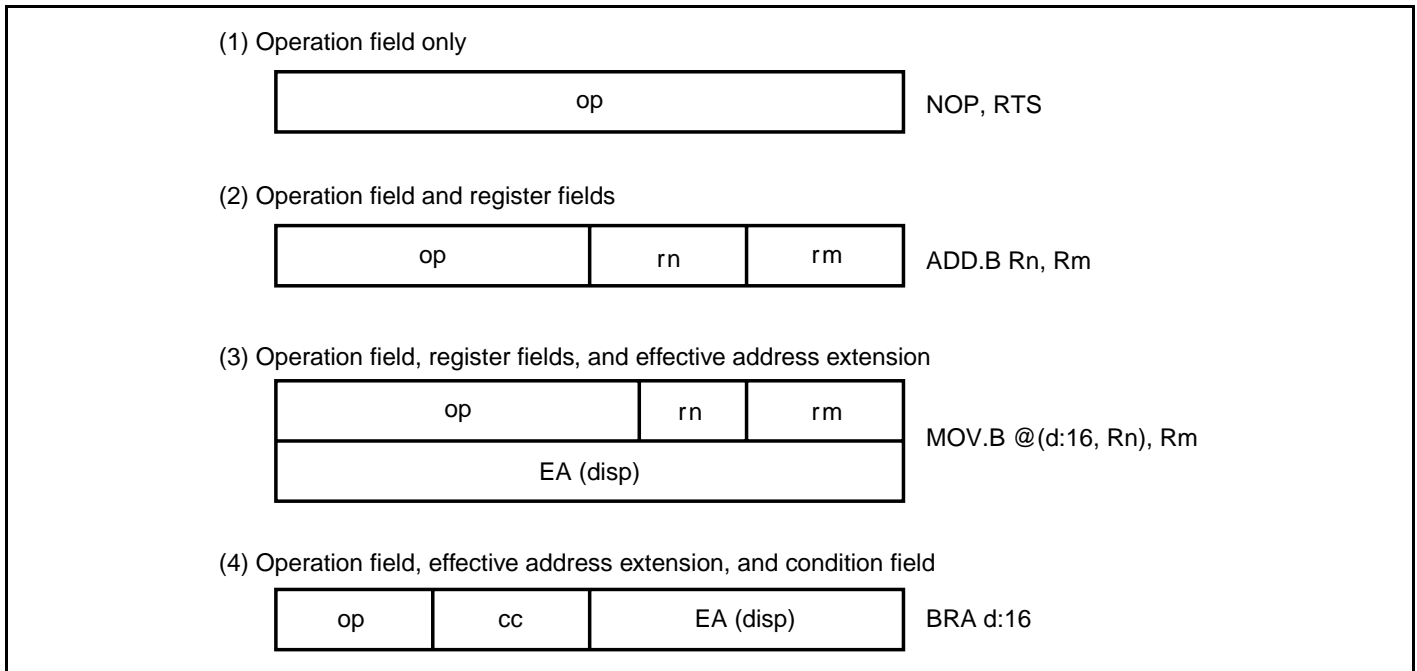
<b>Instruction</b>	<b>Size</b>	<b>Function</b>
EEPMOV.B	—	if R4L $\neq$ 0 then Repeat @ER5+ $\rightarrow$ @ER6+ R4L-1 $\rightarrow$ R4L Until R4L = 0 else next:
EEPMOV.W	—	if R4 $\neq$ 0 then Repeat @ER5+ $\rightarrow$ @ER6+ R4-1 $\rightarrow$ R4 Until R4 = 0 else next:  Transfers a data block. Starting from the address set in ER5, transfers data for the number of bytes set in R4L or R4 to the address location set in ER6.  Execution of the next instruction begins as soon as the transfer is completed.

## 2.6.2 Basic Instruction Formats

The H8S/2000 CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op), a register field (r), an effective address extension (EA), and a condition field (cc).

Figure 2.11 shows examples of instruction formats.

- **Operation field**  
Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.
- **Register field**  
Specifies a general register. Address registers are specified by 3 bits, and data registers by 3 bits or 4 bits. Some instructions have two register fields, and 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 branching condition of Bcc instructions.



**Figure 2.11 Instruction Formats (Examples)**

## 2.7 Addressing Modes and Effective Address Calculation

The H8S/2000 CPU supports the eight addressing modes listed in table 2.11. Each instruction uses a subset of these addressing modes.

Arithmetic and logic operations instructions can use the register direct and immediate addressing modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit manipulation instructions can use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.



**Table 2.11 Addressing Modes**

No.	Addressing Mode	Symbol
1	Register direct	Rn
2	Register indirect	@ERn
3	Register indirect with displacement	@(d:16,ERn)/@(d:32,ERn)
4	Register indirect with post-increment	@ERn+
	Register indirect with pre-decrement	@-ERn
5	Absolute address	@aa:8/@aa:16/@aa:24/@aa:32
6	Immediate	#xx:8/#xx:16/#xx:32
7	Program-counter relative	@(d:8,PC)/@(d:16,PC)
8	Memory indirect	@@aa:8

### 2.7.1 Register Direct—Rn

The register field of the instruction code specifies an 8-, 16-, or 32-bit general register which contains the operand. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

### 2.7.2 Register Indirect—@ERn

The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. If the address is a program instruction address, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

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

A 16-bit or 32-bit displacement contained in the instruction code is added to an address register (ERn) specified by the register field of the instruction, and the sum gives the address of a memory operand. A 16-bit displacement is sign-extended when added.

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

**Register Indirect with Post-Increment—@ERn+:** The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, and 4 for longword access. For word or longword transfer instructions, the register value should be even.

**Register Indirect with Pre-Decrement—@-ERn:** The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the result becomes the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word access, and 4 for longword access. For word or longword transfer instructions, the register value should be even.

### 2.7.5 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32

The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24), or 32 bits long (@aa:32). Table 2.12 indicates the accessible absolute address ranges.

To access data, the absolute address should be 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) long. For an 8-bit absolute address, the upper 16 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address, the upper 16 bits are a sign extension. For a 32-bit absolute address, the entire address space is accessed.

A 24-bit absolute address (@aa:24) indicates the address of a program instruction. The upper 8 bits are all assumed to be 0 (H'00).

**Table 2.12 Absolute Address Access Ranges**

Absolute Address		Normal Mode	Advanced Mode
Data address	8 bits (@aa:8)	H'FF00 to H'FFFF	H'FFFF00 to H'FFFFFF
	16 bits (@aa:16)	H'0000 to H'FFFF	H'000000 to H'007FFF, H'FF8000 to H'FFFFFF
	32 bits (@aa:32)		H'000000 to H'FFFFFF
Program instruction address	24 bits (@aa:24)		

### 2.7.6 Immediate—#xx:8, #xx:16, or #xx:32

The 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data contained in a instruction code can be used directly as an operand.

The ADDS, SUBS, INC, and DEC instructions implicitly contain immediate data in their instruction codes. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, specifying a bit number. The TRAPA instruction contains 2-bit immediate data in its instruction code, specifying a vector address.

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

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

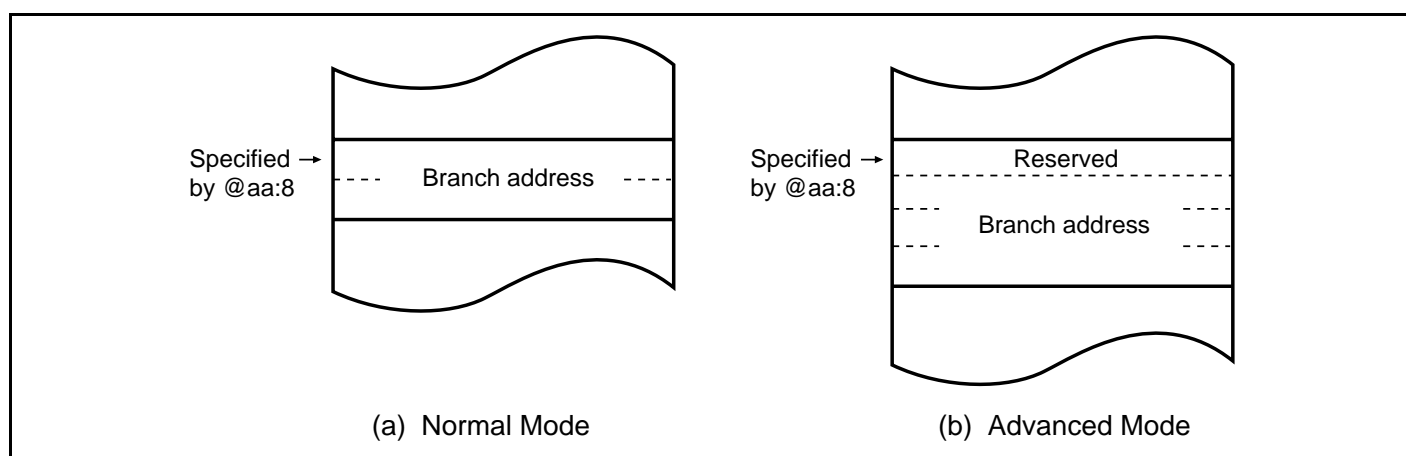
### 2.7.8 Memory Indirect— $@@aa:8$

This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand which contains a branch address. The upper bits of the 8-bit absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in advanced mode).

In normal mode, the memory operand is a word operand and the branch address is 16 bits long. In advanced mode, the memory operand is a longword operand, the first byte of which is assumed to be 0 (H'00).

Note that the top area of the address range in which the branch address is stored is also used for the exception vector area. For further details, see section 4, Exception Handling.

If an odd address is specified in word or longword memory access, or as a branch address, the least significant bit is regarded as 0, causing data to be accessed or the instruction code to be fetched at the address preceding the specified address. (For further information, see section 2.5.2, Memory Data Formats.)



**Figure 2.12 Branch Address Specification in Memory Indirect Addressing Mode**

## 2.7.9 Effective Address Calculation

Table 2.13 indicates how effective addresses are calculated in each addressing mode. In normal mode, the upper 8 bits of the effective address are ignored in order to generate a 16-bit address.

**Table 2.13 Effective Address Calculation (1)**

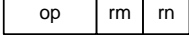
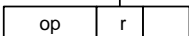

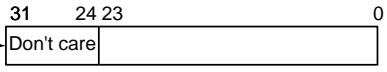
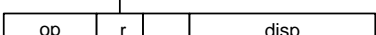
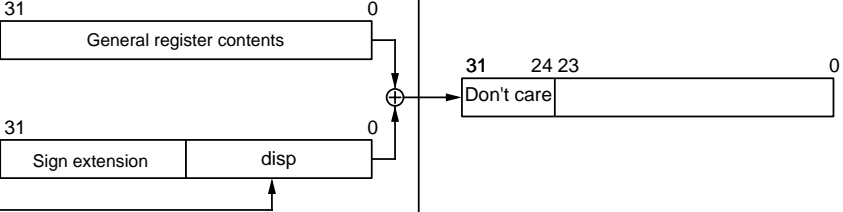
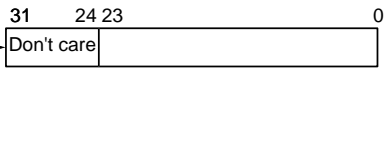
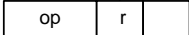

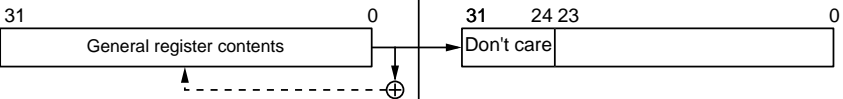
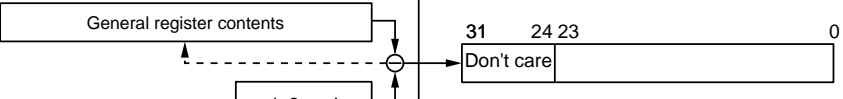
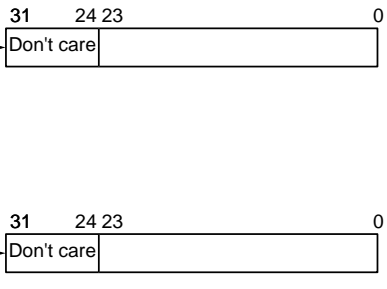
No	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)								
1	Register direct (Rn) 		Operand is general register contents.								
2	Register indirect (@ERn) 										
3	Register indirect with displacement @d:(d:16,ERn) or @:(d:32,ERn) 										
4	Register indirect with post-increment or pre-decrement • Register indirect with post-increment @ERn+  • Register indirect with pre-decrement @-ERn 	  <table border="1" data-bbox="630 1289 943 1394"> <thead> <tr> <th>Operand Size</th> <th>Offset</th> </tr> </thead> <tbody> <tr> <td>Byte</td> <td>1</td> </tr> <tr> <td>Word</td> <td>2</td> </tr> <tr> <td>Longword</td> <td>4</td> </tr> </tbody> </table>	Operand Size	Offset	Byte	1	Word	2	Longword	4	
Operand Size	Offset										
Byte	1										
Word	2										
Longword	4										

Table 2.13 Effective Address Calculation (2)

No	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
5	Absolute address @aa:8 		
	@aa:16 		
	@aa:24 		
	@aa:32 		
6	Immediate #xx:8/#xx:16/#xx:32 		Operand is immediate data.
7	Program-counter relative @(d:8,PC)/@(d:16,PC) 		
8	Memory indirect @aa:8 • Normal mode 		
	• Advanced mode 		

## 2.8 Processing States

The H8S/2000 CPU has five main processing states: the reset state, exception handling state, program execution state, bus-released state, and program stop state. Figure 2.13 indicates the state transitions.

- Reset state

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

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

- Exception-handling state

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

- Program execution state

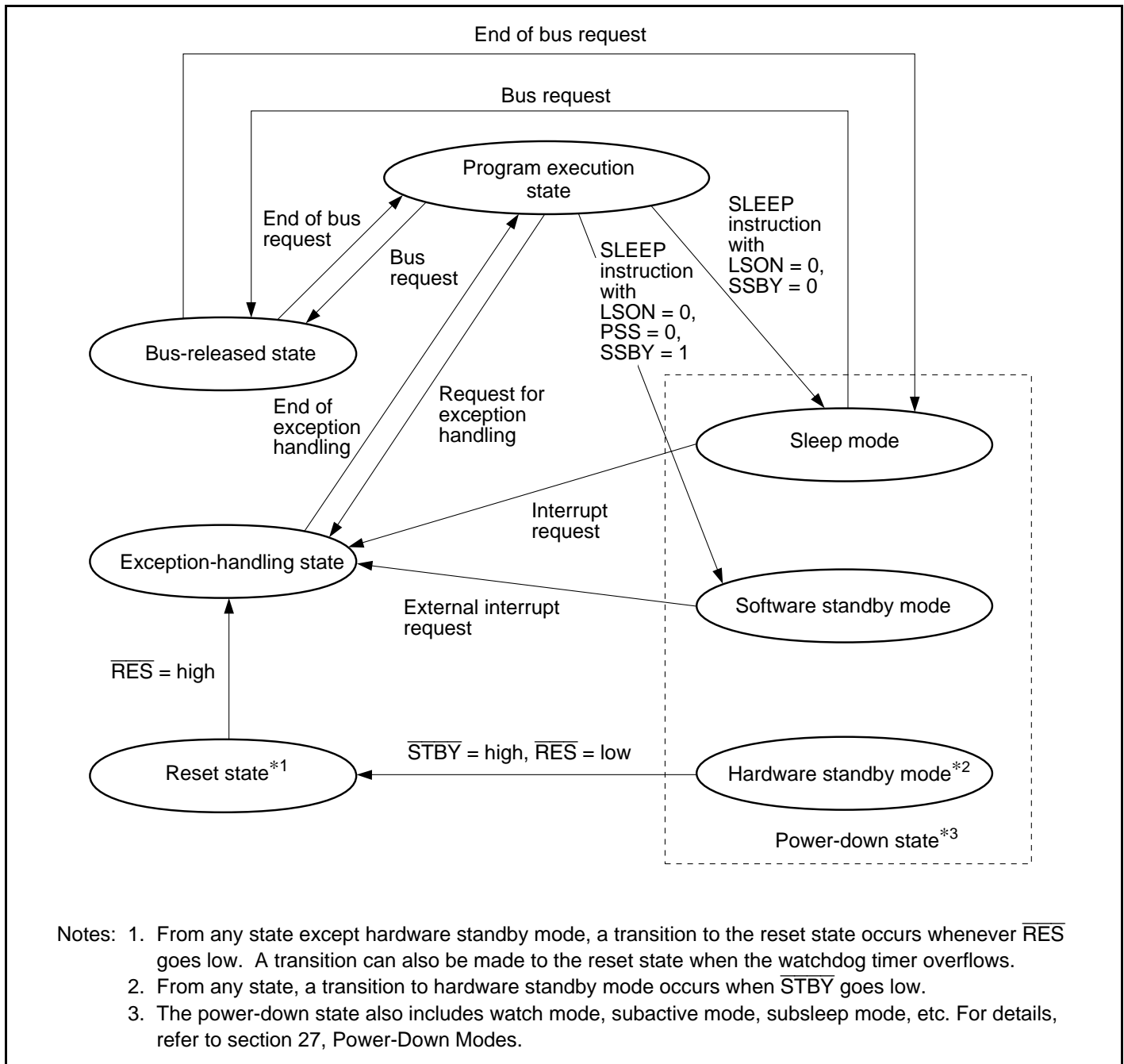
In this state the CPU executes program instructions in sequence.

- Bus-released state

In a product which has a bus master other than the CPU, such as a data transfer controller (DTC) and a RAM-FIFO unit (RFU), the bus-released state occurs when the bus has been released in response to a bus request from a bus master other than the CPU. While the bus is released, the CPU halts operations.

- Program stop state

This is a power-down state in which the CPU stops operating. The program stop state occurs when a SLEEP instruction is executed or the CPU enters hardware standby mode. For details, see section 27, Power-Down Modes.



**Figure 2.13 State Transitions**

## 2.9 Usage Notes

### 2.9.1 Note on TAS Instruction Usage

The TAS instruction is not generated by the Renesas H8S and H8/300 series C/C++ compilers. The TAS instruction can be used as a user-defined intrinsic function.

### 2.9.2 Note on Bit Manipulation Instructions

The BSET, BCLR, BNOT, BST, and BIST instructions read data from the specified address in byte units, manipulate the data of the target bit, and write data to the same address again in byte units. Special care is required when using these instructions in cases where a register containing a write-only bit is used or a bit is directly manipulated for a port, because this may rewrite data of a bit other than the bit to be manipulated.

**Example:** The BCLR instruction is executed for DDR in port 4.

P47 and P46 are input pins, with a low-level signal input at P47 and a high-level signal input at P46. P45 to P40 are output pins and output low-level signals. The following shows an example in which P40 is set to be an input pin with the BCLR instruction.

Prior to executing BCLR

	<b>P47</b>	<b>P46</b>	<b>P45</b>	<b>P44</b>	<b>P43</b>	<b>P42</b>	<b>P41</b>	<b>P40</b>
Input/output	Input	Input	Output	Output	Output	Output	Output	Output
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	Low level
DDR	0	0	1	1	1	1	1	1
DR	1	0	0	0	0	0	0	0



## BCLR instruction executed

BCLR	#0,	@P4DDR
------	-----	--------

The BCLR instruction is executed for DDR in port 4

## After executing BCLR

	<b>P47</b>	<b>P46</b>	<b>P45</b>	<b>P44</b>	<b>P43</b>	<b>P42</b>	<b>P41</b>	<b>P40</b>
Input/output	Output	Output	Output	Output	Output	Output	Output	Input
Pin state	Low level	High level	Low level	Low level	Low level	Low level	Low level	High level
DDR	1	1	1	1	1	1	1	0
DR	1	0	0	0	0	0	0	0

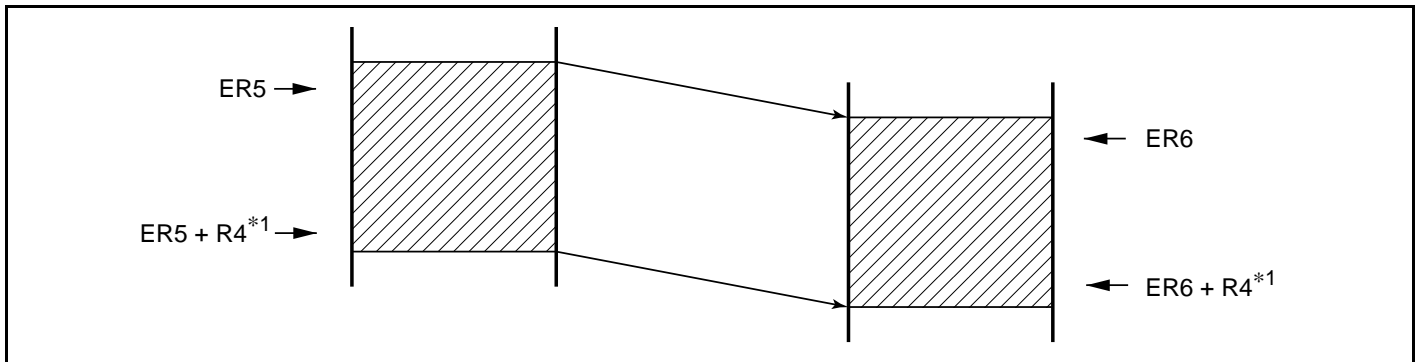
## [Description on Operation]

1. When the BCLR instruction is executed, first the CPU reads P4DDR.  
Since P4DDR is a write-only register, so the CPU reads H'FF. In this example P4DDR has a value of H'3F, but the value read by the CPU is H'FF.
2. The CPU clears bit 0 of the read data to 0, changing data to H'FE.
3. The CPU writes H'FE to DDR, completing execution of BCLR.

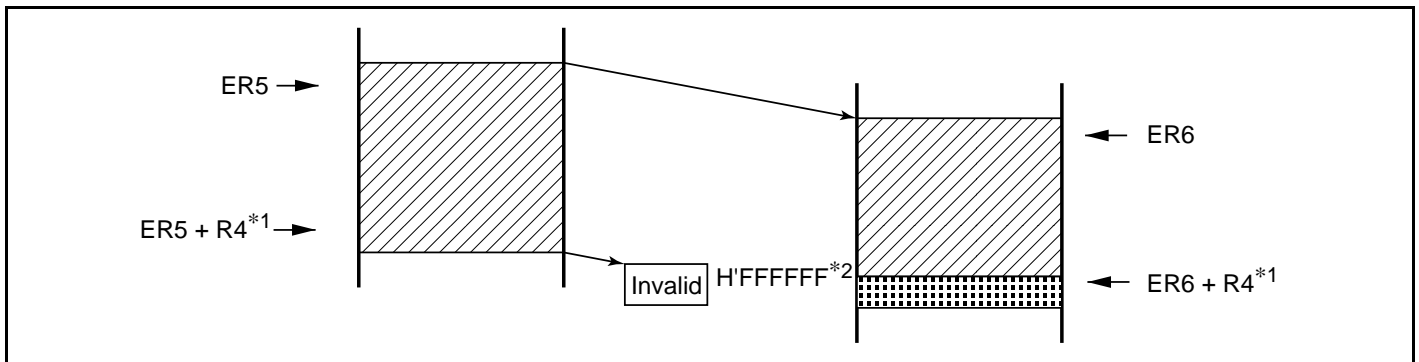
As a result of the BCLR instruction, bit 0 in DDR is set to 0, and P40 becomes an input pin. However, bits 7 and 6 of DDR are modified to 1, therefore P47 and P46 become output pins.

### 2.9.3 EEPMOV Instruction

1. EEPMOV is a block-transfer instruction that transfers the byte size of data indicated by  $R4^{*1}$ , which starts from the address indicated by ER5, to the address indicated by ER6.



2. Set  $R4^{*1}$  and ER6 so that the end address of the destination address (value of ER6 +  $R4^{*1}$ ) does not exceed  $H'00FFFFFF^{*2}$  (the value of ER6 must not change from  $H'00FFFFFF$  to  $H'01000000^{*2}$  during execution).



- Notes:
1. In normal mode, it becomes R4L.
  2. In normal mode, it should not exceed  $H'0000FFFF$ .

## Section 3 MCU Operating Modes

### 3.1 Operating Mode Selection

This LSI supports two operating modes (modes 2 and 3). The operating mode is determined by the setting of the mode pins ( $\overline{\text{MD2}}$ , MD1, and MD0). Table 3.1 shows the MCU operating mode selection.

**Table 3.1 MCU Operating Mode Selection**

MCU Operating Mode	$\overline{\text{MD2}}$	MD1	MD0	CPU Operating Mode	Description	On-Chip ROM
2	1	1	0	Advanced mode	Extended mode with on-chip ROM Single-chip mode	Enabled
3	1	1	1	Normal mode	Extended mode with on-chip ROM Single-chip mode	Enabled

Modes 2 and 3 are single-chip mode after a reset. The CPU can switch to extended mode by setting bit EXPE in MDCR to 1.

Modes 0, 1, and 5 cannot be used in this LSI. Modes 4, 6, and 7 are specific modes. Thus, mode pins should be set to enable mode 2 or 3 in normal program execution state. Mode pins should not be changed during operation.

Mode 4 is a boot mode to write/erase the flash memory. For details, see section 24, ROM.

Modes 6 and 7 are on-chip emulation modes. These modes are controlled by the on-chip emulator (E10A) via the JTAG interface, and on-chip emulation can be performed.

### 3.2 Register Descriptions

The following registers are related to the operating mode. For details on the bus control register (BCR), see section 6.3.1, Bus Control Register (BCR), and for details on bus control register 2 (BCR2), see section 6.3.2, Bus Control Register 2 (BCR2).

- Mode control register (MDCR)
- System control register (SYSCR)
- Serial timer control register (STCR)

### 3.2.1 Mode Control Register (MDCR)

MDCR is used to set an operating mode and to monitor the current operating mode.

Bit	Bit Name	Initial Value	R/W	Description
7	EXPE	0	R/W	Extended Mode Enable Specifies extended mode. 0: Single-chip mode 1: Extended mode
6 to 3	—	All 0	R	Reserved
2	MDS2	—*	R	Mode Select 2 to 0 These bits indicate the input levels at mode pins ( $\overline{MD2}$ , MD1, and MD0) (the current operating mode). Bits MDS2, MDS1, and MDS0 correspond to $\overline{MD2}$ , MD1, and MD0, respectively. MDS2 to MDS0 are read-only bits and they cannot be written to. The mode pin ( $\overline{MD2}$ , MD1, and MD0) input levels are latched into these bits when MDCR is read. These latches are canceled by a reset.
1	MDS1	—*	R	
0	MDS0	—*	R	

Note: \* The initial values are determined by the settings of the  $\overline{MD2}$ , MD1, and MD0 pins.

### 3.2.2 System Control Register (SYSCR)

SYSCR selects a system pin function, monitors a reset source, selects the interrupt control mode and the detection edge for NMI, enables or disables register access to the on-chip peripheral modules, and enables or disables on-chip RAM address space.

Bit	Bit Name	Initial Value	R/W	Description
7	CS256E	0	R/W	<p>Chip Select 256 Enable</p> <p>Enables or disables P97/<math>\overline{\text{WAIT}}</math>/<math>\overline{\text{CPWAIT}}</math>/<math>\overline{\text{CS256}}</math> pin function in extended mode.</p> <p>0: P97/<math>\overline{\text{WAIT}}</math>/<math>\overline{\text{CPWAIT}}</math> pin  <math>\overline{\text{WAIT}}</math>/<math>\overline{\text{CPWAIT}}</math> pin function is selected by the settings of WSCR and WSCR2.</p> <p>1: <math>\overline{\text{CS256}}</math> pin            Outputs low when a specified address of addresses H'F80000 to H'FBFFFF is accessed.</p>
6	IOSE	0	R/W	<p>IOS Enable</p> <p>Enables or disables <math>\overline{\text{AS}}</math>/<math>\overline{\text{IOS}}</math> pin function in extended mode.</p> <p>0: <math>\overline{\text{AS}}</math> pin            Outputs low when an external area is accessed.</p> <p>1: <math>\overline{\text{IOS}}</math> pin            Outputs low when a specified address of addresses H'(FF)F000 to H'(FF)F7FF is accessed.</p>
5	INTM1	0	R	<p>These bits select the control mode of the interrupt controller. For details on the interrupt control modes, see section 5.6, Interrupt Control Modes and Interrupt Operation.</p> <p>00: Interrupt control mode 0</p> <p>01: Interrupt control mode 1</p> <p>10: Setting prohibited</p> <p>11: Setting prohibited</p>
4	INTM0	0	R/W	
3	XRST	1	R	<p>External Reset</p> <p>This bit indicates the reset source. A reset is caused by an external reset input, or when the watchdog timer overflows.</p> <p>0: A reset is caused when the watchdog timer overflows.</p> <p>1: A reset is caused by an external reset.</p>

Bit	Bit Name	Initial Value	R/W	Description
2	NMIEG	0	R/W	<p>NMI Edge Select</p> <p>Selects the valid edge of the NMI interrupt input.</p> <p>0: An interrupt is requested at the falling edge of NMI input</p> <p>1: An interrupt is requested at the rising edge of NMI input</p>
1	KINWUE	0	R/W	<p>Keyboard Control Register Access Enable</p> <p>Enables or disables CPU access for input control registers (KMIMRA, KMIMR6, WUEMR3) of <math>\overline{KINn}</math> and <math>\overline{WUEn}</math> pins, input pull-up MOS control register (KMPCR6) of the <math>\overline{KINn}</math> pin, registers (TCR_X/TCR_Y, TCSR_X/TCSR_Y, TICRR/TCORA_Y, TICRF/TCORB_Y, TCNT_X/TCNT_Y, TCORC/TISR, TCORA_X, TCORB_X) of 8-bit timers (TMR_X, TMR_Y), and timer connection registers (TCONRI, TCONRO, TCONRS, SEDGR).</p> <p>0: Enables CPU access for registers of TMR_X and TMR_Y and timer connection registers in an area from H'(FF)FFF0 to H'(FF)FFF7 and from H'(FF)FFFC to H'(FF)FFFF.</p> <p>1: Enables CPU access for input control registers of the <math>\overline{KINn}</math> and <math>\overline{WUEn}</math> pins and the input pull-up MOS control register of the <math>\overline{KINn}</math> pin in an area from H'(FF)FFF0 to H'(FF)FFF7 and from H'(FF)FFFC to H'(FF)FFFF.</p>
0	RAME	1	R/W	<p>RAM Enable</p> <p>Enables or disables on-chip RAM. The RAME bit is initialized when the reset state is released.</p> <p>0: On-chip RAM is disabled</p> <p>1: On-chip RAM is enabled</p>

### 3.2.3 Serial Timer Control Register (STCR)

STCR enables or disables register access, IIC operating mode, and on-chip flash memory, and selects the input clock of the timer counter.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/(W)	Reserved The initial value should not be changed.
6	IICX1	0	R/W	IIC Transfer Rate Select 1 and 0
5	IICX0	0	R/W	These bits control the IIC operation. These bits select a transfer rate in master mode together with bits CKS2 to CKS0 in the I <sup>2</sup> C bus mode register (ICMR). For details on the transfer rate, see table 17.3. The IICX0 bit controls IIC_0 and the IICX1 bit controls IIC_1.
4	IICE	0	R/W	IIC Master Enable Enables or disables CPU access for IIC registers (ICCR, ICSR, ICDR/SARX, ICMR/SAR), PWMX registers (DADRAH/DACR, DADRAL, DADRBH/DACNTH, DADRBL/DACNTL), and SCI registers (SMR, BRR, SCMR). 0: SCI_1 registers are accessed in an area from H'(FF)FF88 to H'(FF)FF89 and from H'(FF)FF8E to H'(FF)FF8F. SCI_2 registers are accessed in an area from H'(FF)FFA0 to H'(FF)FFA1 and from H'(FF)FFA6 to H'(FF)FFA7. SCI_0 registers are accessed in an area from H'(FF)FFD8 to H'(FF)FFD9 and from H'(FF)FFDE to H'(FF)FFDF. 1: IIC_1 registers are accessed in an area from H'(FF)FF88 to H'(FF)FF89 and from H'(FF)FF8E to H'(FF)FF8F. PWMX registers are accessed in an area from H'(FF)FFA0 to H'(FF)FFA1 and from H'(FF)FFA6 to H'(FF)FFA7. IIC_0 registers are accessed in an area from H'(FF)FFD8 to H'(FF)FFD9 and from H'(FF)FFDE to H'(FF)FFDF.

Bit	Bit Name	Initial Value	R/W	Description
3	FLSHE	0	R/W	Flash Memory Control Register Enable  Enables or disables CPU access for flash memory registers (FLMCR1, FLMCR2, EBR1, EBR2), control registers of power-down states (SBYCR, LPWRCR, MSTPCRH, MSTPCRL), and control registers of on-chip peripheral modules (BCR2, WSCR2, PCSR, SYSCR2).  0: Control registers of power-down states and on-chip peripheral modules are accessed in an area from H'(FF)FF80 to H'(FF)FF87.  1: Control registers of flash memory are accessed in an area from H'(FF)FF80 to H'(FF)FF87.
2	—	0	R/(W)	Reserved  The initial value should not be changed.
1	ICKS1	0	R/W	Internal Clock Source Select 1, 0  These bits select a clock to be input to the timer counter (TCNT) and a count condition together with bits CKS2 to CKS0 in the timer control register (TCR). For details, see section 13.3.4, Timer Control Register (TCR).
0	ICKS0	0		

### 3.3 Operating Mode Descriptions

#### 3.3.1 Mode 2

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled.

After a reset, the LSI is set to single-chip mode. To access an external address space, bit EXPE in MDCR should be set to 1. However, because this LSI has a maximum of 18 address output pins, an external address space can be accessed only when the I/O strobe function of the  $\overline{AS}/\overline{IOS}$  pin, the CP/CF extension function, and the  $\overline{CS256}$  function are used.

In extended mode, ports 1 and 2 function as input ports after a reset. Ports 1 and 2 function as an address bus by setting 1 to the corresponding port data direction register (DDR). Port 3 functions as a data bus, and parts of port 9 carry bus control signals. Port 6 functions as a data bus when the ABW bit in WSCR is cleared to 0.



### 3.3.2 Mode 3

The CPU can access a 64-kbyte address space in normal mode. The on-chip ROM is enabled. The CPU can access a 56-kbyte address space in mode 3.

After a reset, the LSI is set to single-chip mode. To access an external address space, bit EXPE in MDCR should be set to 1.

In extended mode, ports 1 and 2 function as input ports after a reset. Ports 1 and 2 function as an address bus by setting 1 to the corresponding port data direction register (DDR). Port 3 functions as a data bus, and parts of port 9 carry bus control signals. Port 6 functions as a data bus when the ABW bit in WSCR is cleared to 0.

### 3.3.3 Pin Functions

Pin functions of ports 1 to 3, 6, 9, and A depend on the operating mode. Table 3.2 shows pin functions in each operating mode.

**Table 3.2 Pin Functions in Each Operating Mode**

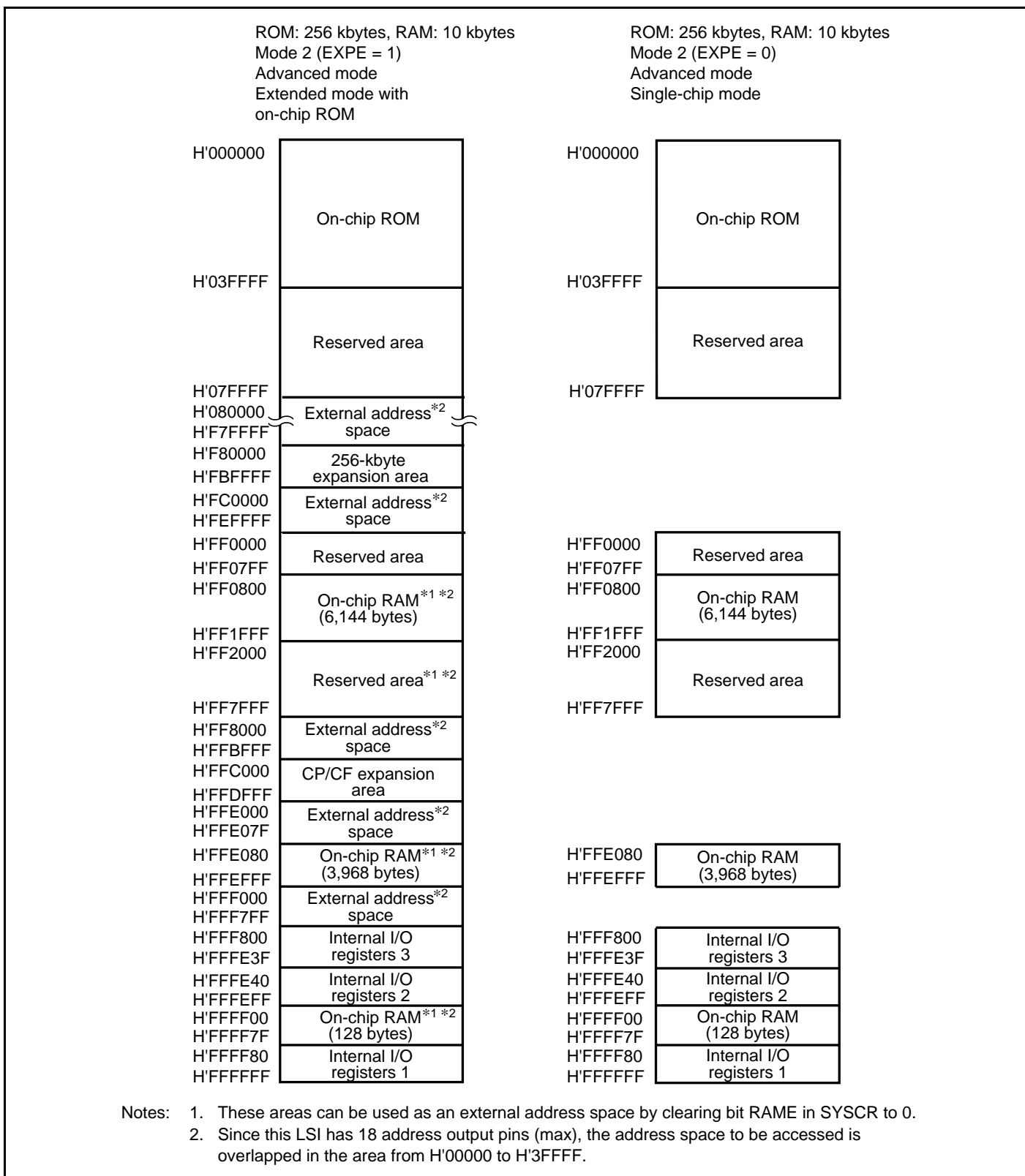
Port	Mode 2	Mode 3
Port 1	I/O port*/Address bus output	I/O port*/Address bus output
Port 2	I/O port*/Address bus output	I/O port*/Address bus output
Port 3	I/O port*/Data bus I/O	I/O port*/Data bus output
Port 6	I/O port*/Data bus I/O	I/O port*/Data bus output
Port 9	P97	I/O port*/Control signal output
	P96	Input port*/Clock I/O
	P95 to P93	I/O port*/Control signal output
	P90	I/O port*/Control signal output
Port A	I/O port*/Address bus output	I/O port*

Legend:

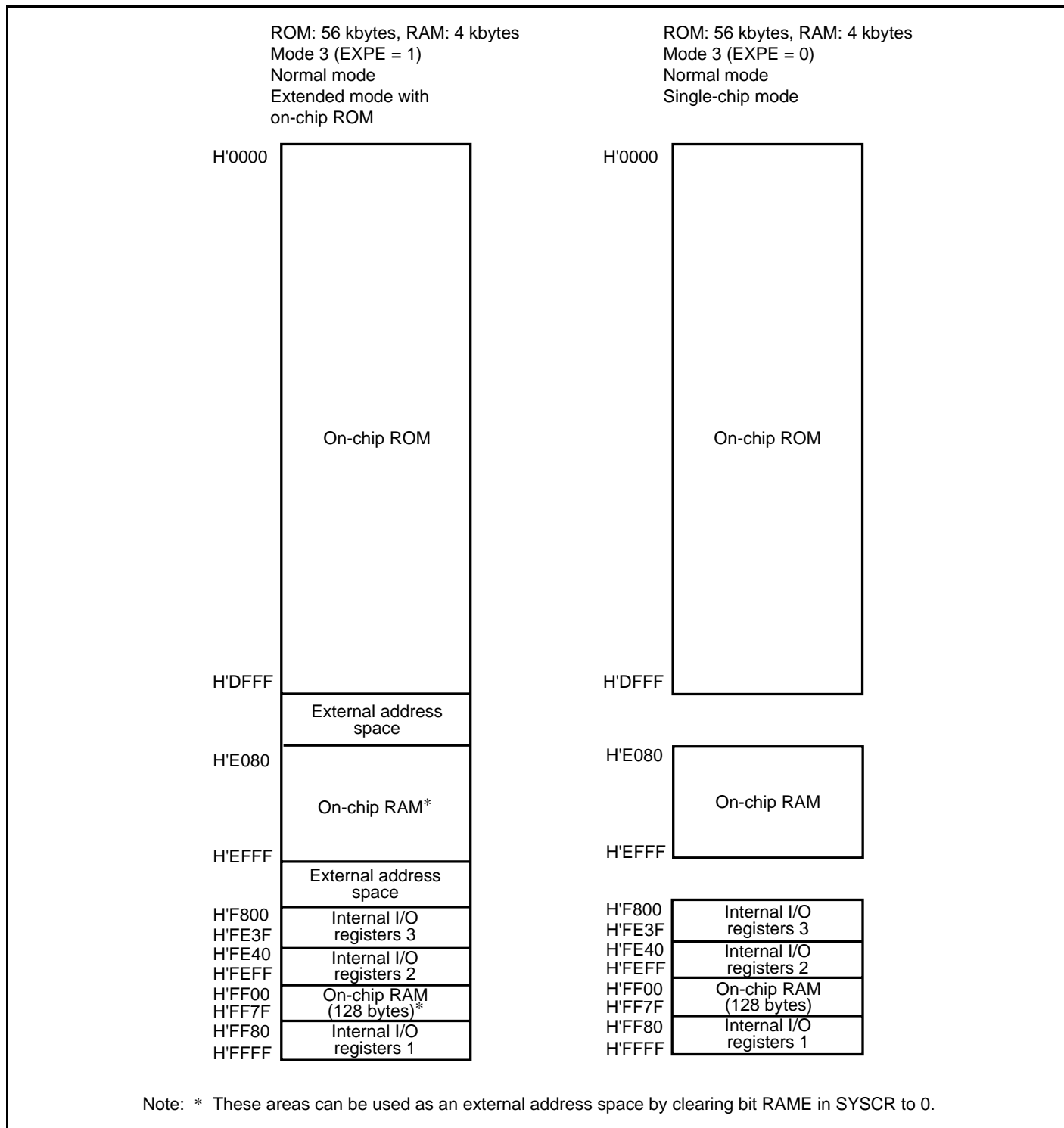
\*: After reset

## 3.4 Address Map in Each Operating Mode

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



**Figure 3.1 Address Map (Mode 2)**



**Figure 3.2 Address Map (Mode 3)**

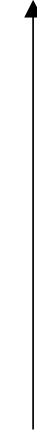


## Section 4 Exception Handling

### 4.1 Exception Handling Types and Priority

As table 4.1 indicates, exception handling may be caused by a reset, interrupt, direct transition, or trap instruction. Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority.

**Table 4.1 Exception Types and Priority**

Priority	Exception Type	Start of Exception Handling
High  Low	Reset	Starts immediately after a low-to-high transition of the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows.
	Interrupt	Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
	Direct transition	Starts when a direction transition occurs as the result of SLEEP instruction execution.
	Trap instruction	Started by execution of a trap (TRAPA) instruction. Trap instruction exception handling requests are accepted at all times in program execution state.

## 4.2 Exception Sources and Exception Vector Table

Different vector addresses are assigned to different exception sources. Table 4.2 lists the exception sources and their vector addresses.

**Table 4.2 Exception Handling Vector Table**

Exception Source	Vector Number	Vector Address		
		Normal Mode	Advanced Mode	
Reset	0	H'0000 to H'0001	H'000000 to H'000003	
Reserved for system use	1	H'0002 to H'0003	H'000004 to H'000007	
	5	H'000A to H'000B	H'000014 to H'000017	
Direct transition	6	H'000C to H'000D	H'000018 to H'00001B	
External interrupt (NMI)	7	H'000E to H'000F	H'00001C to H'00001F	
Trap instruction (four sources)	8	H'0010 to H'0011	H'000020 to H'000023	
	9	H'0012 to H'0013	H'000024 to H'000027	
	10	H'0014 to H'0015	H'000028 to H'00002B	
	11	H'0016 to H'0017	H'00002C to H'00002F	
Direct transition (clock switchover)	12	H'0018 to H'0019	H'000030 to H'000033	
Reserved for system use	13	H'001A to H'001B	H'000034 to H'000037	
	15	H'001E to H'001F	H'00003C to H'00003F	
External interrupt	IRQ0	16	H'0020 to H'0021	H'000040 to H'000043
External interrupt	IRQ1	17	H'0022 to H'0023	H'000044 to H'000047
External interrupt	IRQ2	18	H'0024 to H'0025	H'000048 to H'00004B
External interrupt	IRQ3	19	H'0026 to H'0027	H'00004C to H'00004F
External interrupt	IRQ4	20	H'0028 to H'0029	H'000050 to H'000053
External interrupt	IRQ5	21	H'002A to H'002B	H'000054 to H'000057
External interrupt	IRQ6	22	H'002C to H'002D	H'000058 to H'00005B
External interrupt	IRQ7	23	H'002E to H'002F	H'00005C to H'00005F
Internal interrupt*	24	H'0030 to H'0031	H'000060 to H'000063	
	29	H'003A to H'003B	H'000074 to H'000077	

Exception Source	Vector Number	Vector Address	
		Normal Mode	Advanced Mode
External interrupt KIN7 to KIN0	30	H'003C to H'003D	H'000078 to H'00007B
External interrupt KIN9, KIN8	31	H'003E to H'003F	H'00007C to H'00007F
Reserved for system use	32	H'0040 to H'0041	H'000080 to H'000083
External interrupt WUE15 to WUE8	33	H'0042 to H'0043	H'000084 to H'000087
Internal interrupt*	34	H'0044 to H'0045	H'000088 to H'00008B
	55	H'006E to H'006F	H'0000DC to H'0000DF
External interrupt IRQ8	56	H'0070 to H'0071	H'0000E0 to H'0000E3
External interrupt IRQ9	57	H'0072 to H'0073	H'0000E4 to H'0000E7
External interrupt IRQ10	58	H'0074 to H'0075	H'0000E8 to H'0000EB
External interrupt IRQ11	59	H'0076 to H'0077	H'0000EC to H'0000EF
External interrupt IRQ12	60	H'0078 to H'0079	H'0000F0 to H'0000F3
External interrupt IRQ13	61	H'007A to H'007B	H'0000F4 to H'0000F7
External interrupt IRQ14	62	H'007C to H'007D	H'0000F8 to H'0000FB
External interrupt IRQ15	63	H'007E to H'007F	H'0000FC to H'0000FF
Internal interrupt*	64	H'0080 to H'0081	H'000100 to H'000103
	114	H'00E4 to H'00E5	H'0001C8 to H'0001CB

Note: \* For details on the internal interrupt vector table, see section 5.5, Interrupt Exception Handling Vector Table.

### 4.3 Reset

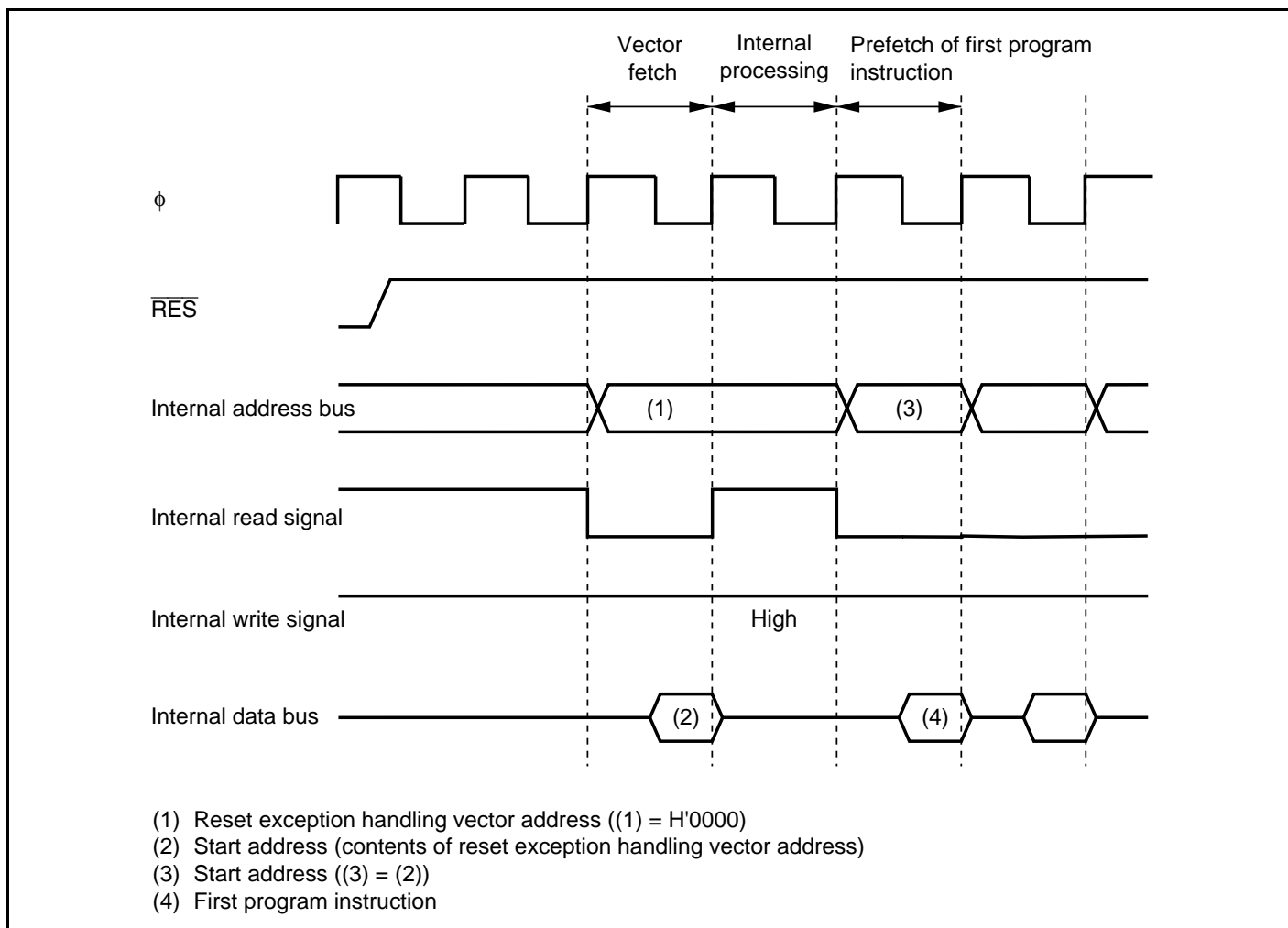
A reset has the highest exception priority. When the  $\overline{\text{RES}}$  pin goes low, all processing halts and this LSI enters the reset. To ensure that this LSI is reset, hold the  $\overline{\text{RES}}$  pin low for at least 20 ms at power-on. To reset the chip during operation, hold the  $\overline{\text{RES}}$  pin low for at least 20 states. A reset initializes the internal state of the CPU and the registers of on-chip peripheral modules. The chip can also be reset by overflow of the watchdog timer. For details, see section 15, Watchdog Timer (WDT).

### 4.3.1 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 and the I bit is set to 1 in 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.

Figure 4.1 shows an example of the reset sequence.



**Figure 4.1 Reset Sequence (Mode 3)**



### 4.3.2 Interrupts after Reset

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

### 4.3.3 On-Chip Peripheral Modules after Reset Is Cancelled

After a reset is cancelled, the module stop control registers (MSTPCR, SUBMSTPA, and SUBMSTPB) are initialized, and all modules except the DTC operate in module stop mode. Therefore, the registers of on-chip peripheral modules cannot be read from or written to. To read from and write to these registers, clear module stop mode.

## 4.4 Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The sources to start interrupt exception handling are external interrupt sources (NMI, IRQ15 to IRQ0, KIN9 to KIN0, and WUE15 to WUE8) and internal interrupt sources from the on-chip peripheral modules. NMI is an interrupt with the highest priority. For details, see section 5, Interrupt Controller.

Interrupt exception handling is conducted as follows:

1. The values in the program counter (PC) and condition code register (CCR) are saved to the stack.
2. A vector address corresponding to the interrupt source is generated, the start address is loaded from the vector table to the PC, and program execution begins from that address.

## 4.5 Trap Instruction Exception Handling

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.

Trap instruction exception handling is conducted as follows:

1. The values in the program counter (PC) and condition code register (CCR) are saved to the stack.
2. A vector address corresponding to the interrupt source is generated, the start address is loaded from the vector table to the PC, and program execution starts from that address.

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

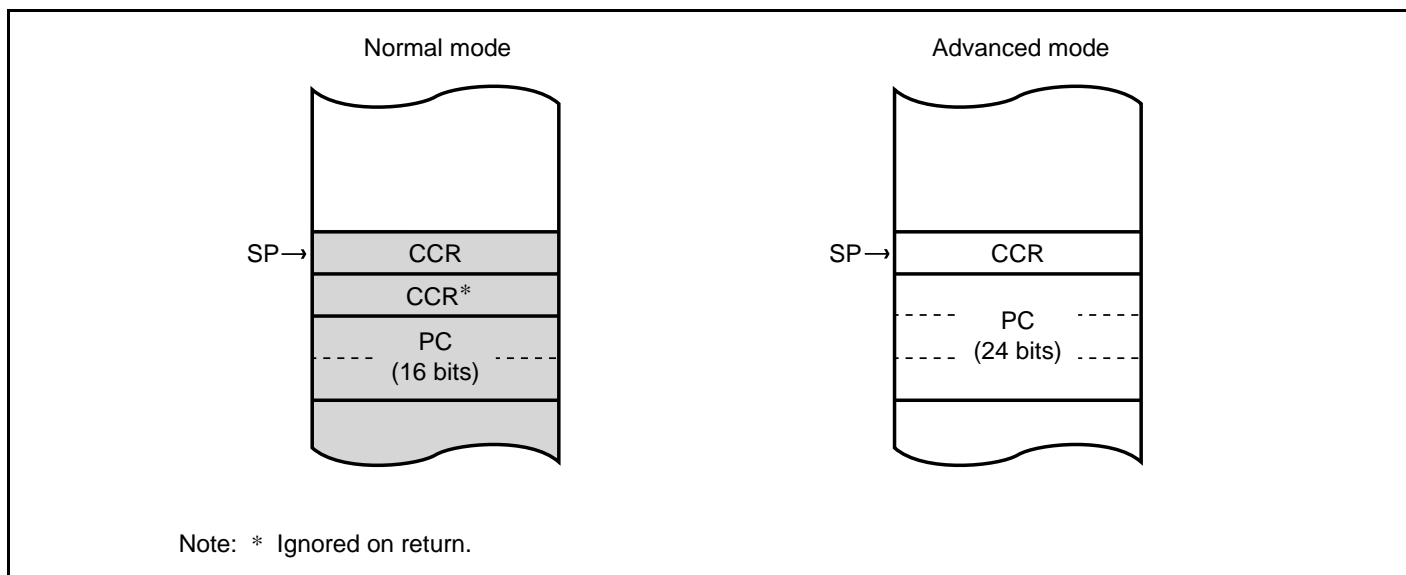
Table 4.3 shows the status of CCR after execution of trap instruction exception handling.

**Table 4.3 Status of CCR after Trap Instruction Exception Handling**

Interrupt Control Mode	CCR	
	I	UI
0	Set to 1	Retains value prior to execution
1	Set to 1	Set to 1

## 4.6 Stack Status after Exception Handling

Figure 4.2 shows the stack after completion of trap instruction exception handling and interrupt exception handling.



**Figure 4.2 Stack Status after Exception Handling**

## 4.7 Usage Note

When accessing word data or longword data, this LSI assumes that the lowest address bit is 0. The stack should always be accessed in words or longwords, and the value of the stack pointer (SP: ER7) should always be kept even.

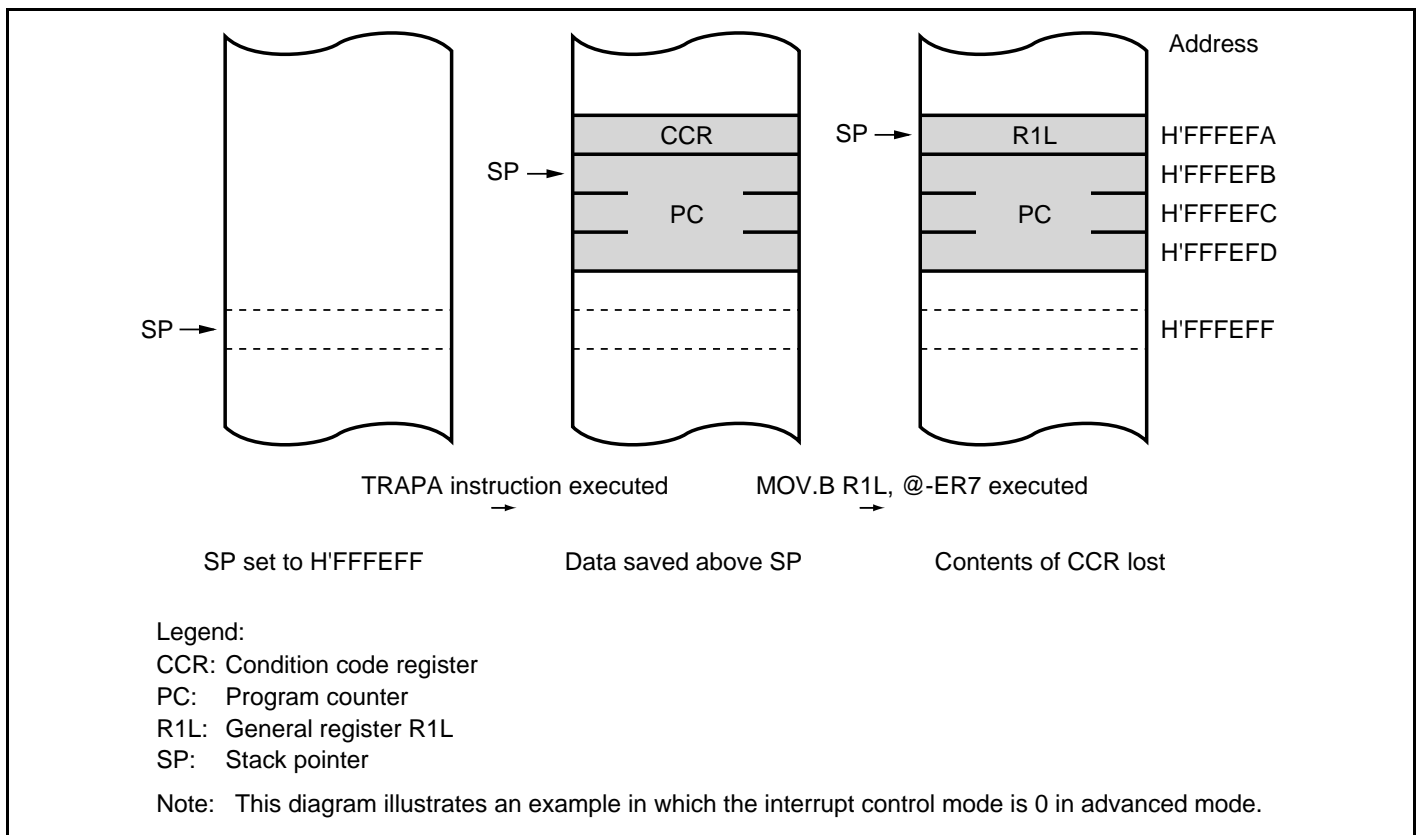
Use the following instructions to save registers:

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

Use the following instructions to restore registers:

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

Setting SP to an odd value may lead to a malfunction. Figure 4.3 shows an example of what happens when the SP value is odd.



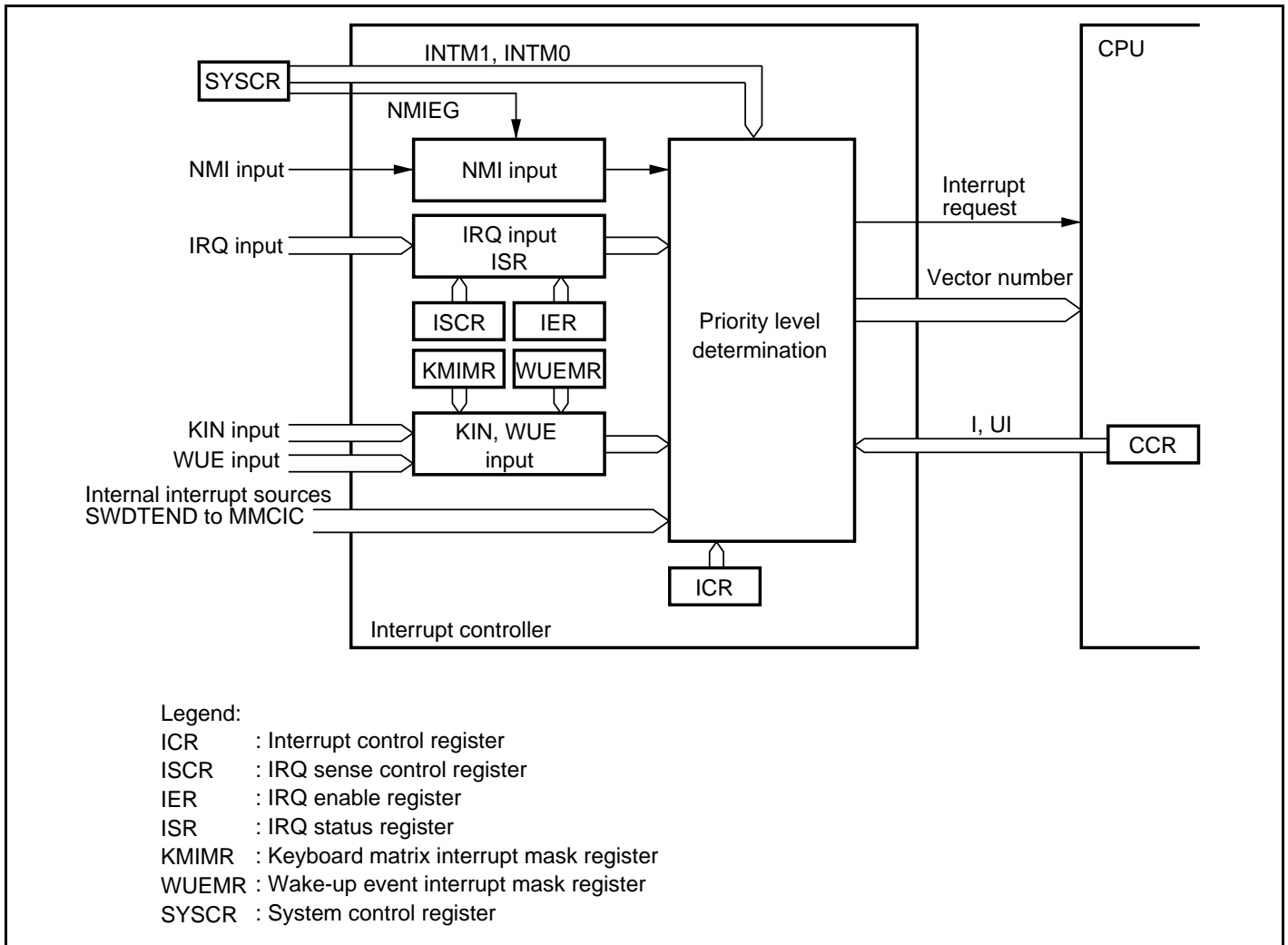
**Figure 4.3 Operation when SP Value Is Odd**



## Section 5 Interrupt Controller

### 5.1 Features

- Two interrupt control modes  
Any of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).
- Priorities settable with ICR  
An interrupt control register (ICR) is provided for setting interrupt priorities. Three priority levels can be set for each module for all interrupts except NMI, KIN, and WUE.
- 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.
- Thirty-five external interrupts  
NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge detection can be selected for NMI. Falling-edge, rising-edge, or both-edge detection, or level sensing, can be selected for  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ0}}$ . An interrupt is requested at the falling edge for  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$ .
- DTC control  
The DTC can be activated by an interrupt request.



**Figure 5.1 Block Diagram of Interrupt Controller**

## 5.2 Input/Output Pins

Table 5.1 summarizes the pins of the interrupt controller.

**Table 5.1 Pin Configuration**

Symbol	I/O	Function
NMI	Input	Nonmaskable external interrupt Rising edge or falling edge can be selected
$\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$ $\overline{\text{ExIRQ15}}$ to $\overline{\text{ExIRQ2}}$	Input	Maskable external interrupts Rising edge, falling edge, or both edges, or level sensing, can be selected individually for each pin. Pin of IRQn or ExIRQn to input IRQ15 to IRQ2 interrupts can be selected.
$\overline{\text{KIN9}}$ to $\overline{\text{KIN0}}$	Input	Maskable external interrupts An interrupt is requested at falling edge.
$\overline{\text{WUE15}}$ to $\overline{\text{WUE8}}$	Input	Maskable external interrupts An interrupt is requested at falling edge.

## 5.3 Register Descriptions

The interrupt controller has the following registers. For details on the system control register (SYSCR), see section 3.2.2, System Control Register (SYSCR), and for details on the IRQ sense port select registers (ISSR16, ISSR), see section 9.11.1, IRQ Sense Port Select Register 16 (ISSR16), IRQ Sense Port Select Register (ISSR).

- Interrupt control registers A to D (ICRA to ICRD)
- Address break control register (ABRKCR)
- Break address registers A to C (BARA to BARC)
- IRQ sense control registers (ISCR16H, ISCR16L, ISCRH, ISCRL)
- IRQ enable registers (IER16, IER)
- IRQ status registers (ISR16, ISR)
- Keyboard matrix interrupt mask registers (KMIMRA, KMIMR6)
- Wake-up event interrupt mask registers (WUEMR3)

### 5.3.1 Interrupt Control Registers A to D (ICRA to ICRD)

The ICR registers set interrupt control levels for interrupts other than NMI.

The correspondence between interrupt sources and ICRA to ICRD settings is shown in table 5.2.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	ICRn7 to ICRn0	All 0	R/W	Interrupt Control Level 0: Corresponding interrupt source is interrupt control level 0 (no priority) 1: Corresponding interrupt source is interrupt control level 1 (priority)

Note: n: A to D

**Table 5.2 Correspondence between Interrupt Source and ICR**

Bit	Bit Name	Register			
		ICRA	ICRB	ICRC	ICRD
7	ICRn7	IRQ0	A/D converter	SCI_0	IRQ8 to IRQ11
6	ICRn6	IRQ1	FRT	SCI_1	IRQ12 to IRQ15
5	ICRn5	IRQ2, IRQ3	—	SCI_2	—
4	ICRn4	IRQ4, IRQ5	TMR_X	IIC_0	—
3	ICRn3	IRQ6, IRQ7	TMR_0	IIC_1	—
2	ICRn2	DTC	TMR_1	—	—
1	ICRn1	WDT_0	TMR_Y	—	—
0	ICRn0	WDT_1	—	USB	MCIF

Notes: n: A to D

—: Reserved. The write value should always be 0.



### 5.3.2 Address Break Control Register (ABRKCR)

ABRKCR controls the address breaks. When both the CMF flag and BIE flag are set to 1, an address break is requested.

Bit	Bit Name	Initial Value	R/W	Description
7	CMIF	Undefined	R/W	Condition Match Flag Address break source flag. Indicates that an address specified by BARA to BARC is prefetched. [Clearing condition] When an exception handling is executed for an address break interrupt. [Setting condition] When an address specified by BARA to BARC is prefetched while the BIE flag is set to 1.
6 to 1	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
0	BIE	0	R/W	Break Interrupt Enable Enables or disables address break. 0: Disabled 1: Enabled

### 5.3.3 Break Address Registers A to C (BARA to BARC)

The BAR registers specify an address that is to be a break address. An address in which the first byte of an instruction exists should be set as a break address. In normal mode, addresses A23 to A16 are not compared.

#### BARA

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	A23 to A16	All 0	R/W	Addresses 23 to 16 The A23 to A16 bits are compared with A23 to A16 in the internal address bus.

#### BARB

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	A15 to A8	All 0	R/W	Addresses 15 to 8 The A15 to A8 bits are compared with A15 to A8 in the internal address bus.

#### BARC

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	A7 to A1	All 0	R/W	Addresses 7 to 1 The A7 to A1 bits are compared with A7 to A1 in the internal address bus.
0	—	All 0	R	Reserved This bit is always read as 0 and cannot be modified.

### 5.3.4 IRQ Sense Control Registers (ISCR16H, ISCR16L, ISCRH, ISCRL)

The ISCR registers select the source that generates an interrupt request at pins  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ0}}$  or pins  $\overline{\text{ExIRQ15}}$  to  $\overline{\text{ExIRQ2}}$ . Switching between pins  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ2}}$  and pins  $\overline{\text{ExIRQ15}}$  to  $\overline{\text{ExIRQ2}}$  is performed by means of IRQ sense port select register 16 (ISSR16) and the IRQ sense port select register (ISSR).

#### ISCR16H

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ15SCB	0	R/W	IRQn Sense Control B
6	IRQ15SCA	0	R/W	IRQn Sense Control A
5	IRQ14SCB	0	R/W	00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
4	IRQ14SCA	0	R/W	
3	IRQ13SCB	0	R/W	01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
2	IRQ13SCA	0	R/W	
1	IRQ12SCB	0	R/W	10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
0	IRQ12SCA	0	R/W	
				11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input (n = 15 to 12)

#### ISCR16L

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ11SCB	0	R/W	IRQn Sense Control B
6	IRQ11SCA	0	R/W	IRQn Sense Control A
5	IRQ10SCB	0	R/W	00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
4	IRQ10SCA	0	R/W	
3	IRQ9SCB	0	R/W	01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
2	IRQ9SCA	0	R/W	
1	IRQ8SCB	0	R/W	10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
0	IRQ8SCA	0	R/W	
				11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input (n = 11 to 8)

**ISCRH**

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7SCB	0	R/W	IRQn Sense Control B
6	IRQ7SCA	0	R/W	IRQn Sense Control A
5	IRQ6SCB	0	R/W	00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
4	IRQ6SCA	0	R/W	
3	IRQ5SCB	0	R/W	01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
2	IRQ5SCA	0	R/W	
1	IRQ4SCB	0	R/W	10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input
0	IRQ4SCA	0	R/W	
				11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input (n = 7 to 4)

**ISCRL**

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ3SCB	0	R/W	IRQn Sense Control B
6	IRQ3SCA	0	R/W	IRQn Sense Control A
5	IRQ2SCB	0	R/W	00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}^*$ input
4	IRQ2SCA	0	R/W	
3	IRQ1SCB	0	R/W	01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}^*$ input
2	IRQ1SCA	0	R/W	
1	IRQ0SCB	0	R/W	10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}^*$ input
0	IRQ0SCA	0	R/W	
				11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}^*$ input (n = 3 to 0)

Note: \*  $\overline{\text{ExIRQn}}$  stands for  $\overline{\text{ExIRQ3}}$  or  $\overline{\text{ExIRQ2}}$ .

### 5.3.5 IRQ Enable Registers (IER16, IER)

IERs control the enabling and disabling of interrupt requests IRQ15 to IRQ0.

#### IER16

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15E	0	R/W	IRQn Enable (n = 15 to 8)
14	IRQ14E	0	R/W	The IRQn interrupt request is enabled when this bit is 1.
13	IRQ13E	0	R/W	
12	IRQ12E	0	R/W	
11	IRQ11E	0	R/W	
10	IRQ10E	0	R/W	
9	IRQ9E	0	R/W	
8	IRQ8E	0	R/W	

#### IER

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7E	0	R/W	IRQn Enable (n = 7 to 0)
6	IRQ6E	0	R/W	The IRQn interrupt request is enabled when this bit is 1.
5	IRQ5E	0	R/W	
4	IRQ4E	0	R/W	
3	IRQ3E	0	R/W	
2	IRQ2E	0	R/W	
1	IRQ1E	0	R/W	
0	IRQ0E	0	R/W	

### 5.3.6 IRQ Status Registers (ISR16, ISR)

The ISR registers are flag registers that indicate the status of IRQ15 to IRQ0 interrupt requests.

#### ISR16

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ15F	0	R/W	[Setting condition]
6	IRQ14F	0	R/W	When the interrupt source selected by the ISCR registers occurs
5	IRQ13F	0	R/W	
4	IRQ12F	0	R/W	[Clearing conditions]
3	IRQ11F	0	R/W	
2	IRQ10F	0	R/W	<ul style="list-style-type: none"> <li>When reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag</li> <li>When interrupt exception handling is executed when low-level detection is set and <math>\overline{\text{IRQn}}</math> or <math>\overline{\text{ExIRQn}}</math> input is high (n = 15 to 8)</li> <li>When IRQn interrupt exception handling is executed when falling-edge, rising-edge, or both-edge detection is set</li> </ul>
1	IRQ9F	0	R/W	
0	IRQ8F	0	R/W	

#### ISR

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7F	0	R/W	[Setting condition]
6	IRQ6F	0	R/W	When the interrupt source selected by the ISCR registers occurs
5	IRQ5F	0	R/W	
4	IRQ4F	0	R/W	[Clearing conditions]
3	IRQ3F	0	R/W	
2	IRQ2F	0	R/W	<ul style="list-style-type: none"> <li>When reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag</li> <li>When interrupt exception handling is executed when low-level detection is set and <math>\overline{\text{IRQn}}</math> or <math>\overline{\text{ExIRQn}}^*</math> input is high (n = 7 to 0)</li> <li>When IRQn interrupt exception handling is executed when falling-edge, rising-edge, or both-edge detection is set (n = 7 to 0)</li> </ul>
1	IRQ1F	0	R/W	
0	IRQ0F	0	R/W	

Note: \*  $\overline{\text{ExIRQn}}$  stands for  $\overline{\text{ExIRQ7}}$  to  $\overline{\text{ExIRQ2}}$ .

### 5.3.7 Keyboard Matrix Interrupt Mask Registers (KMIMRA, KMIMR6) Wake-Up Event Interrupt Mask Register (WUEMR3)

The KMIMR and WUEMR registers enable or disable wake-up key-sensing interrupt inputs ( $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$ ), and wake-up event interrupt inputs ( $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$ ).

#### KMIMRA

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 1	R/W	Reserved These bits should not be cleared to 0.
1	KMIM9	1	R/W	Keyboard Matrix Interrupt Mask
0	KMIM8	1	R/W	These bits enable or disable a key-sensing input interrupt request (KIN9 and KIN8). 0: Enables a key-sensing input interrupt request 1: Disables a key-sensing input interrupt request

#### KMIMR6

Bit	Bit Name	Initial Value	R/W	Description
7	KMIM7	1	R/W	Keyboard Matrix Interrupt Mask
6	KMIM6	1	R/W	These bits enable or disable a wake-up key-in input interrupt request (KIN7 to KIN0).
5	KMIM5	1	R/W	
4	KMIM4	1	R/W	0: Enables a key-sensing input interrupt request 1: Disables a key-sensing input interrupt request
3	KMIM3	1	R/W	
2	KMIM2	1	R/W	1: Disables a key-sensing input interrupt request
1	KMIM1	1	R/W	
0	KMIM0	1	R/W	

#### WUEMR3

Bit	Bit Name	Initial Value	R/W	Description
7	WUEM15	1	R/W	Wake-Up Event Interrupt Mask
6	WUEM14	1	R/W	These bits enable or disable a wake-up event input interrupt request (WUE15 to WUE8).
5	WUEM13	1	R/W	
4	WUEM12	1	R/W	0: Enables a wake-up event input interrupt request 1: Disables a wake-up event input interrupt request
3	WUEM11	1	R/W	
2	WUEM10	1	R/W	1: Disables a wake-up event input interrupt request
1	WUEM9	1	R/W	
0	WUEM8	1	R/W	

## 5.4 Interrupt Sources

### 5.4.1 External Interrupts

There are four external interrupts: NMI, IRQ15 to IRQ0, KIN9 to KIN0 and WUE15 to WUE8. These interrupts can be used to restore this LSI from software standby mode.

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

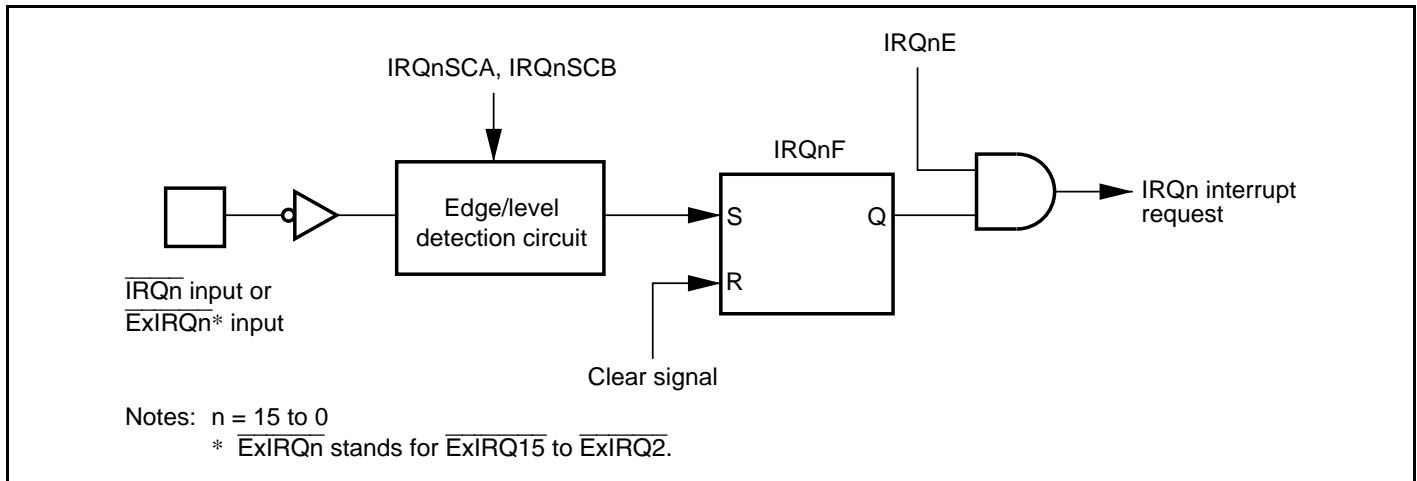
**IRQ15 to IRQ0 Interrupts:** Interrupts IRQ15 to IRQ0 are requested by an input signal at pins  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ0}}$  or pins  $\overline{\text{ExIRQ15}}$  to  $\overline{\text{ExIRQ2}}$ . Interrupts IRQ15 to IRQ0 have the following features:

- The interrupt exception handling for interrupt requests IRQ15 to IRQ0 can be started at an independent vector address.
- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins  $\overline{\text{IRQ15}}$  to  $\overline{\text{IRQ0}}$  or pins  $\overline{\text{ExIRQ15}}$  to  $\overline{\text{ExIRQ2}}$ .
- Enabling or disabling of interrupt requests IRQ15 to IRQ0 can be selected with IER.
- The status of interrupt requests IRQ15 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

The detection of IRQ15 to IRQ0 interrupts does not depend on whether the relevant pin has been set for input or output. However, when a pin is used as an external interrupt input pin, do not clear the corresponding port DDR to 0 to use the pin as an I/O pin for another function.

A block diagram of interrupts IRQ15 to IRQ0 is shown in figure 5.2.





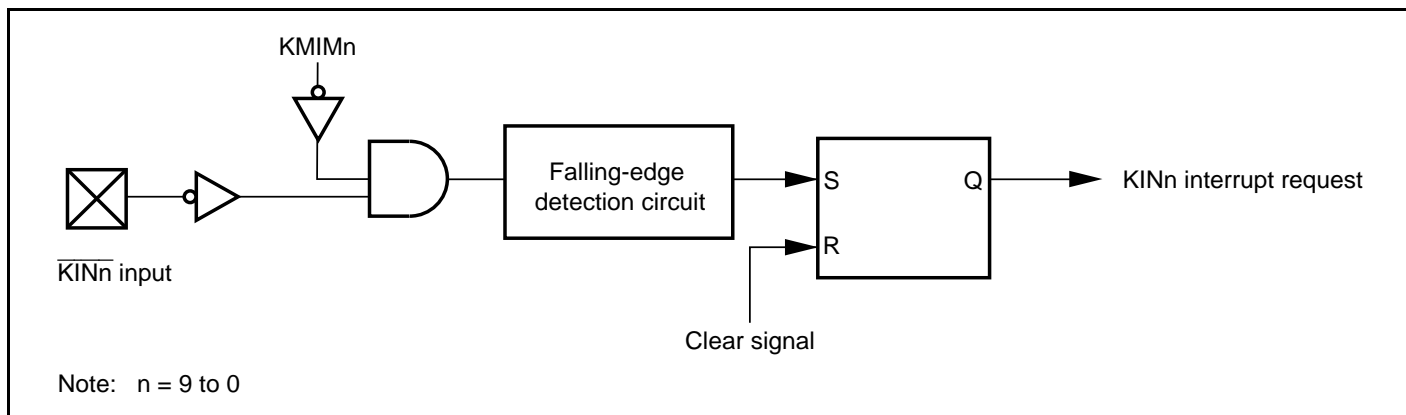
**Figure 5.2 Block Diagram of Interrupts IRQ15 to IRQ0**

**KIN9 to KIN0 Interrupts, WUE15 to WUE8 Interrupts:** Interrupts  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  are requested by an input signal at pins  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$ . Interrupts  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  have the following features:

- Interrupts  $\overline{\text{KIN9}}$  and  $\overline{\text{KIN8}}$ ,  $\overline{\text{KIN7}}$  to  $\overline{\text{KIN0}}$ , and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  each form a group. The interrupt exception handling for an interrupt request from the same group is started at the same vector address.
- Enabling or disabling of interrupt requests can be selected with the I bit in CCR.
- An interrupt is generated by a falling edge at pins  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$ .
- Enabling or disabling of interrupt requests  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  can be selected using KMIMRA, KMIMR6, and WUEMR3.
- The status of interrupt requests  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  are not indicated.

The detection of  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  interrupts does not depend on whether the relevant pin has been set for input or output. However, when a pin is used as an external interrupt input pin, do not clear the corresponding port DDR to 0 to use the pin as an I/O pin for another function.

A block diagram of interrupts  $\overline{\text{KIN9}}$  to  $\overline{\text{KIN0}}$  and  $\overline{\text{WUE15}}$  to  $\overline{\text{WUE8}}$  is shown in figure 5.3.



**Figure 5.3 Block Diagram of Interrupts KIN9 to KIN0 and WUE15 to WUE8  
(Example of KIN9 to KIN0)**

### 5.4.2 Internal Interrupts

Internal interrupts issued from the 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 individually select enabling or disabling of these interrupts. When the enable bit for a particular interrupt source is set to 1, an interrupt request is sent to the interrupt controller.
- The control level for each interrupt can be set by ICR.
- The DTC can be activated by an interrupt request from an on-chip peripheral module.
- An interrupt request that activates the DTC is not affected by the interrupt control mode or the status of the CPU interrupt mask bits.

## 5.5 Interrupt Exception Handling Vector Table

Table 5.3 lists interrupt exception handling sources, vector addresses, and interrupt priorities. For default priorities, the lower the vector number, the higher the priority. Modules set at the same priority will conform to their default priorities. Priorities within a module are fixed.

An interrupt control level can be specified for a module to which an ICR bit is assigned. Interrupt requests from modules that are set to interrupt control level 1 (priority) by the ICR bit setting and the I and UI bits in CCR are given priority and processed before interrupt requests from modules that are set to interrupt control level 0 (no priority).

**Table 5.3 Interrupt Sources, Vector Addresses, and Interrupt Priorities**

Origin of Interrupt Source	Name	Vector Number	Vector Address		ICR	Priority
			Normal Mode	Advanced Mode		
External pin	NMI	7	H'000E	H'00001C	—	High
	IRQ0	16	H'0020	H'000040	ICRA7	↑
	IRQ1	17	H'0022	H'000044	ICRA6	
	IRQ2	18	H'0024	H'000048	ICRA5	
	IRQ3	19	H'0026	H'00004C		
	IRQ4	20	H'0028	H'000050	ICRA4	
	IRQ5	21	H'002A	H'000054		
	IRQ6	22	H'002C	H'000058	ICRA3	
IRQ7	23	H'002E	H'00005C			
DTC	SWDTEND (Software activation data transfer end)	24	H'0030	H'000060	ICRA2	
WDT_0	WOVI0 (Interval timer)	25	H'0032	H'000064	ICRA1	
WDT_1	WOVI1 (Interval timer)	26	H'0034	H'000068	ICRA0	
—	Address break	27	H'0036	H'00006C	—	
A/D converter	ADI (A/D conversion end)	28	H'0038	H'000070	ICRB7	
—	Reserved for system use	29	H'003A	H'000074	—	
External pin	KIN7 to KIN0	30	H'003C	H'000078	—	
	KIN9 and KIN8	31	H'003E	H'00007C		
	Reserved for system use	32	H'0040	H'000080		
	WUE15 to WUE8	33	H'0042	H'000084		
RFU	DTI0	34	H'0044	H'000088	—	
	DTI1	35	H'0046	H'00008C		
	DTI2	36	H'0048	H'000090		
	DTI3	37	H'004A	H'000094		
	Reserved for system use	38	H'004C	H'000098		
	Reserved for system use	39	H'004E	H'00009C		
	Reserved for system use	40	H'0050	H'0000A0		
	Reserved for system use	41	H'0052	H'0000A4		
	DTIE	42	H'0054	H'0000A8		
	Reserved for system use	43	H'0056	H'0000AC		
TMR_X	CMIAX (Compare match A)	44	H'0058	H'0000B0	ICRB4	
	CMIBX (Compare match B)	45	H'005A	H'0000B4		
	OVIX (Overflow)	46	H'005C	H'0000B8		
	ICIX (Input capture)	47	H'005E	H'0000BC		Low

Origin of Interrupt Source	Name	Vector Number	Vector Address		ICR	Priority
			Normal Mode	Advanced Mode		
FRT	ICIA (Input capture A)	48	H'0060	H'0000C0	ICRB6	High ↑
	ICIB (Input capture B)	49	H'0062	H'0000C4		
	ICIC (Input capture C)	50	H'0064	H'0000C8		
	ICID (Input capture D)	51	H'0066	H'0000CC		
	OCIA (Output compare A)	52	H'0068	H'0000D0		
	OCIB (Output compare B)	53	H'006A	H'0000D4		
	FOVI (Overflow)	54	H'006C	H'0000D8		
	Reserved for system use	55	H'006E	H'0000DC		
External pin	IRQ8	56	H'0070	H'0000E0	ICRD7	
	IRQ9	57	H'0072	H'0000E4		
	IRQ10	58	H'0074	H'0000E8		
	IRQ11	59	H'0076	H'0000EC		
	IRQ12	60	H'0078	H'0000F0	ICRD6	
	IRQ13	61	H'007A	H'0000F4		
	IRQ14	62	H'007C	H'0000F8		
	IRQ15	63	H'007E	H'0000FC		
TMR_0	CMIA0 (Compare match A)	64	H'0080	H'000100	ICRB3	
	CMIB0 (Compare match A)	65	H'0082	H'000104		
	OVI0 (Overflow)	66	H'0084	H'000108		
	Reserved for system use	67	H'0086	H'00010C		
TMR_1	CMIA1 (Compare match A)	68	H'0088	H'000110	ICRB2	
	CMIB1 (Compare match B)	69	H'008A	H'000114		
	OVI1 (Overflow)	70	H'008C	H'000118		
	Reserved for system use	71	H'008E	H'00011C		
TMR_Y	CMIA Y (Compare match A)	72	H'0090	H'000120	ICRB1	
	CMIB Y (Compare match B)	73	H'0092	H'000124		
	OVI Y (Overflow)	74	H'0094	H'000128		
	Reserved for system use	75	H'0096	H'00012C		
—	Reserved for system use	76	H'0098	H'000130	ICRC2	
		77	H'009A	H'000134		
		78	H'009C	H'000138		
		79	H'009E	H'00013C		
SCI_0	ERI0 (Reception error 0)	80	H'00A0	H'000140	ICRC7	
	RXI0 (Reception completion 0)	81	H'00A2	H'000144		
	TXI0 (Transmission data empty 0)	82	H'00A4	H'000148		
	TEI0 (Transmission end 0)	83	H'00A6	H'00014C		
SCI_1	ERI1 (Reception error 1)	84	H'00A8	H'000150	ICRC6	Low
	RXI1 (Reception completion 1)	85	H'00AA	H'000154		
	TXI1 (Transmission data empty 1)	86	H'00AC	H'000158		
	TEI1 (Transmission end 1)	87	H'00AE	H'00015C		

Origin of Interrupt Source	Name	Vector Number	Vector Address		ICR	Priority
			Normal Mode	Advanced Mode		
SCI_2	ERI2 (Reception error 2)	88	H'00B0	H'000160	ICRC5	High ↑
	RXI2 (Reception completion 2)	89	H'00B2	H'000164		
	TXI2 (Transmission data empty 2)	90	H'00B4	H'000168		
	TEI2 (Transmission end 2)	91	H'00B6	H'00016C		
IIC_0	IICC0	92	H'00B8	H'000170	ICRC4	
	IICM0	93	H'00BA	H'000174		
	IICR0	94	H'00BC	H'000178		
	IICT0	95	H'00BE	H'00017C		
IIC_1	IICC1	96	H'00C0	H'000180	ICRC3	
	IICM1	97	H'00C2	H'000184		
	IICR1	98	H'00C4	H'000188		
	IICT1	99	H'00C6	H'00018C		
—	Reserved for system use	100	H'00C8	H'000190	ICRB0	
		101	H'00CA	H'000194		
		102	H'00CC	H'000198		
		103	H'00CE	H'00019C		
—	Reserved for system use	104	H'00D0	H'0001A0	ICRC1	
		105	H'00D2	H'0001A4		
		106	H'00D4	H'0001A8		
		107	H'00D6	H'0001AC		
USB	USBI0	108	H'00D8	H'0001B0	ICRC0	
	USBI1	109	H'00DA	H'0001B4		
	USBI2	110	H'00DC	H'0001B8		
	USBI3	111	H'00DE	H'0001BC		
MCIF	MMCIA	112	H'00E0	H'0001C0	ICRD0	Low
	MMCIB	113	H'00E2	H'0001C4		
	MMCIC	114	H'00E4	H'0001C8		

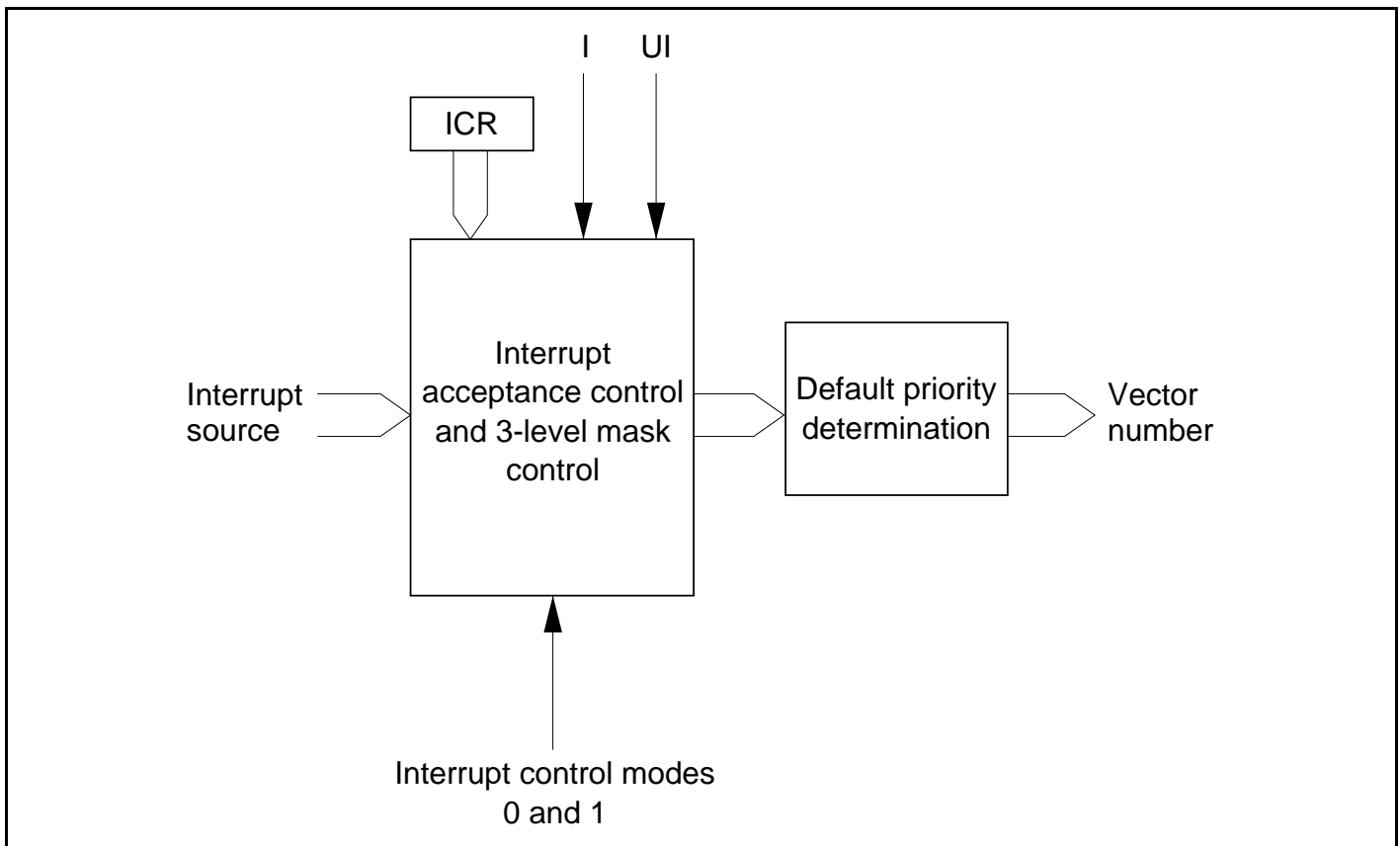
## 5.6 Interrupt Control Modes and Interrupt Operation

The interrupt controller has two modes: Interrupt control mode 0 and interrupt control mode 1. Interrupt operations differ depending on the interrupt control mode. NMI interrupts and address break interrupts are always accepted except for in reset state or in hardware standby mode. The interrupt control mode is selected by SYSCR. Table 5.4 shows the interrupt control modes.

**Table 5.4 Interrupt Control Modes**

Interrupt Control Mode	SYSCR		Priority Setting Registers	Interrupt Mask Bits	Description
	INTM1	INTM0			
0	0	0	ICR	I	Interrupt mask control is performed by the I bit. Priority levels can be set with ICR.
1		1	ICR	I, UI	3-level interrupt mask control is performed by the I bit. Priority levels can be set with ICR.

Figure 5.4 shows a block diagram of the priority decision circuit.



**Figure 5.4 Block Diagram of Interrupt Control Operation**

**Interrupt Acceptance Control and 3-Level Control:** In interrupt control modes 0 and 1, interrupt acceptance control and 3-level mask control is performed by means of the I and UI bits in CCR and ICR (control level).

Table 5.5 shows the interrupts that can be accepted in each interrupt control mode.

**Table 5.5 Interrupts Acceptable in Each Interrupt Control Mode**

Interrupt Control Mode	I Bit	UI Bit	NMI, Address Break	KIN, WUE, DTI	Peripheral Module Interrupt
0	0	—	O	O	O (All interrupts)
	1	—	O	X	X
1	0	—	O	O	O (All interrupts)
	1	0	O	X	O (Interrupts with ICR = 1)
		1	O	X	X

Legend:

—: Don't care

Note: \* Interrupt control level 1 has priority.

**Default Priority Determination:** The priority is determined for the selected interrupt, and a vector number is generated.

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

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

Table 5.6 shows operations and control signal functions in each interrupt control mode.

**Table 5.6 Operations and Control Signal Functions in Each Interrupt Control Mode**

Interrupt Control Mode	Setting		Interrupt Acceptance Control 3-Level Control			Default Priority Determination	T (Trace)	
	INTM1	INTM0	I	UI	ICR			
0	0	0	O	IM	—	PR	O	—
1		1	O	IM	IM	PR	O	—

Legend:

O: Interrupt operation control performed

IM: Used as an interrupt mask bit

PR: Sets priority

—: Not used

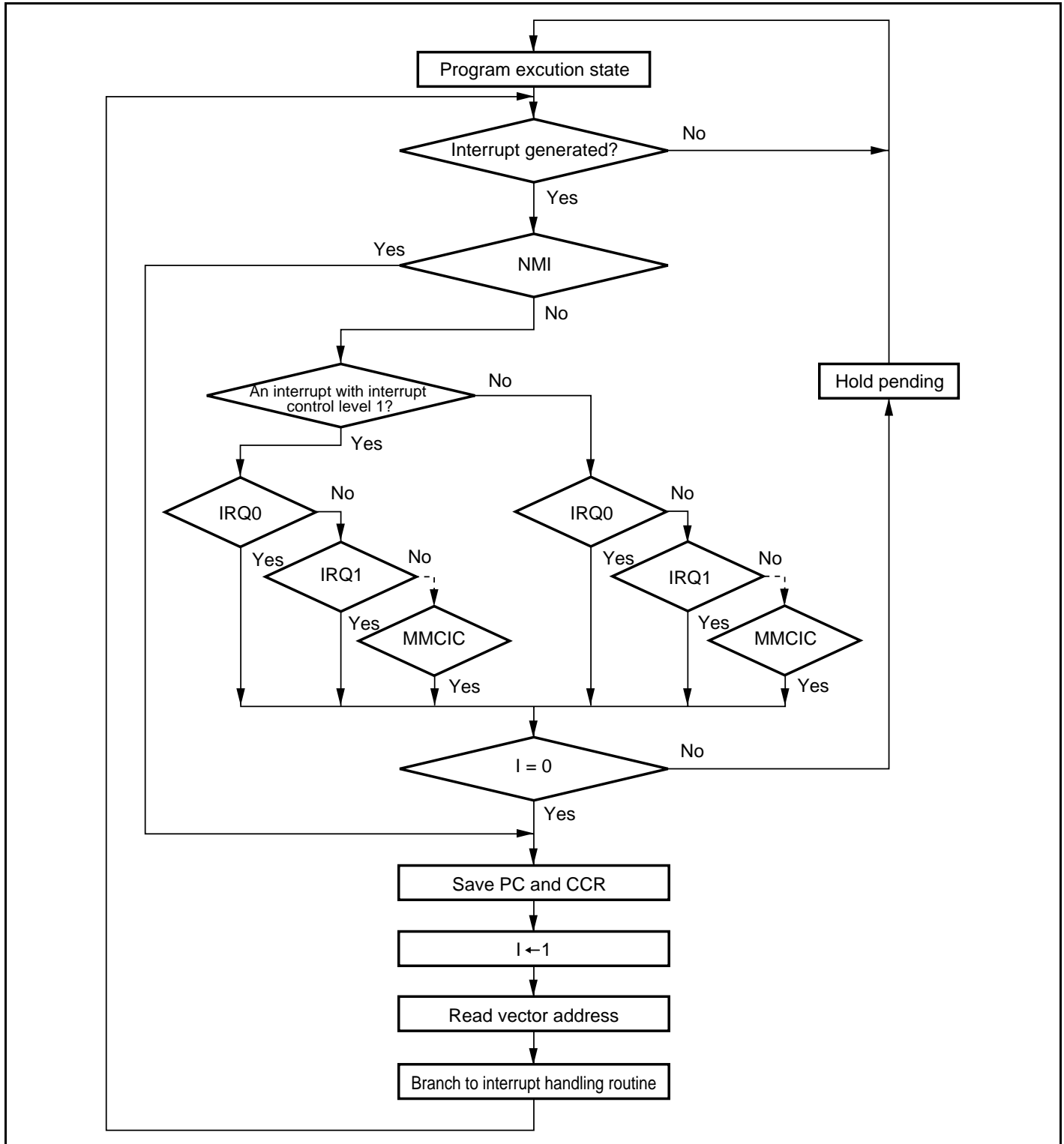
### 5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests other than NMI are masked by ICR and the I bit of the CCR in the CPU. Note however that the KIN, WUE, and DTI interrupt requests can be accepted when the I bit is cleared to 0 and are held pending when the I bit is set to 1. Figure 5.5 shows a flowchart of the interrupt acceptance operation.

1. If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. According to the interrupt control level specified in ICR, the interrupt controller only accepts an interrupt request with interrupt control level 1 (priority), and holds pending an interrupt request with interrupt control level 0 (no priority). If several interrupt requests are issued, an interrupt request with the highest priority is accepted according to the priority order, an interrupt handling is requested to the CPU, and other interrupt requests are held pending.
3. If the I bit in CCR is set to 1, only NMI and address break interrupts are accepted by the interrupt controller, and other interrupt requests are held pending. If the I bit is cleared to 0, any interrupt request is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
6. Next, the I bit in CCR is set to 1. This masks all interrupts except for NMI and address break interrupts.



7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.



**Figure 5.5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0**

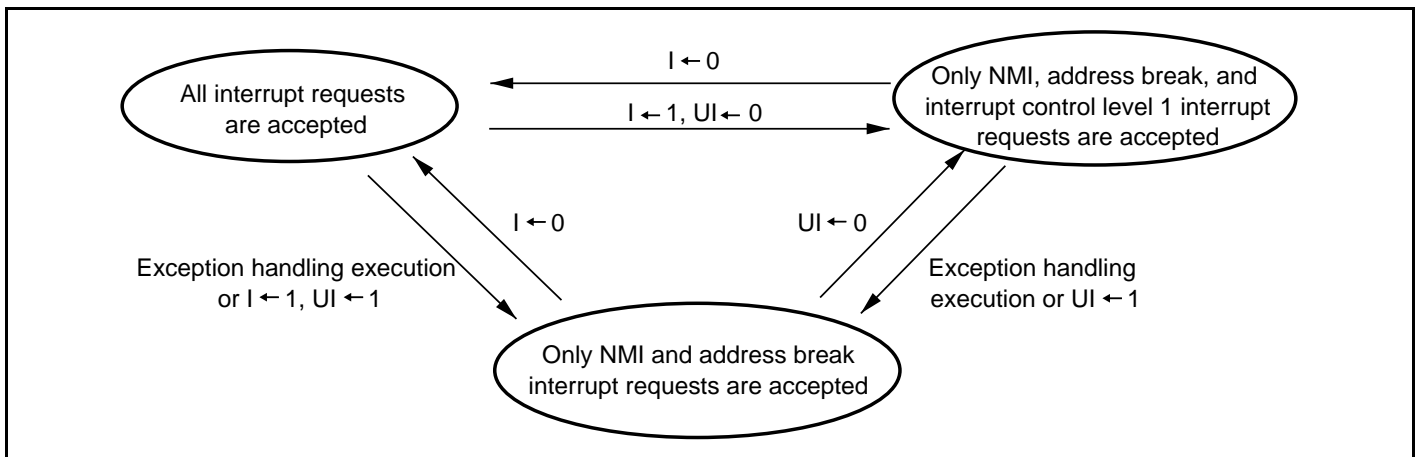
## 5.6.2 Interrupt Control Mode 1

In interrupt control mode 1, mask control is applied to three levels for IRQ and on-chip peripheral module interrupt requests by comparing the I and UI bits in CCR in the CPU, and the ICR setting. Note however that the KIN, WUE, and DTI interrupt requests can be accepted when the I bit is cleared to 0 and are held pending when the I bit is set to 1.

1. An interrupt request with interrupt control level 0 is accepted when the I bit in CCR is cleared to 0. When the I bit is set to 1, the interrupt request is held pending.
2. An interrupt request with interrupt control level 1 is accepted when the I bit or UI bit in CCR is cleared to 0. When both I and UI bits are set to 1, the interrupt request is held pending.

For instance, the state transition when the interrupt enable bit corresponding to each interrupt is set to 1, and ICRA to ICRD are set to H'20, H'00, and H'00, respectively (IRQ2 and IRQ3 interrupts are set to interrupt control level 1, and other interrupts are set to interrupt control level 0) is shown below. Figure 5.6 shows a state transition diagram.

1. All interrupt requests are accepted when  $I = 0$ . (Priority order: NMI > IRQ2 > IRQ3 > IRQ0 > IRQ1 > address break ...)
2. Only NMI, IRQ2, IRQ3, and address break interrupt requests are accepted when  $I = 1$  and  $UI = 0$ .
3. Only NMI and address break interrupt requests are accepted when  $I = 1$  and  $UI = 1$ .



**Figure 5.6 State Transition in Interrupt Control Mode 1**

Figure 5.7 shows a flowchart of the interrupt acceptance operation.

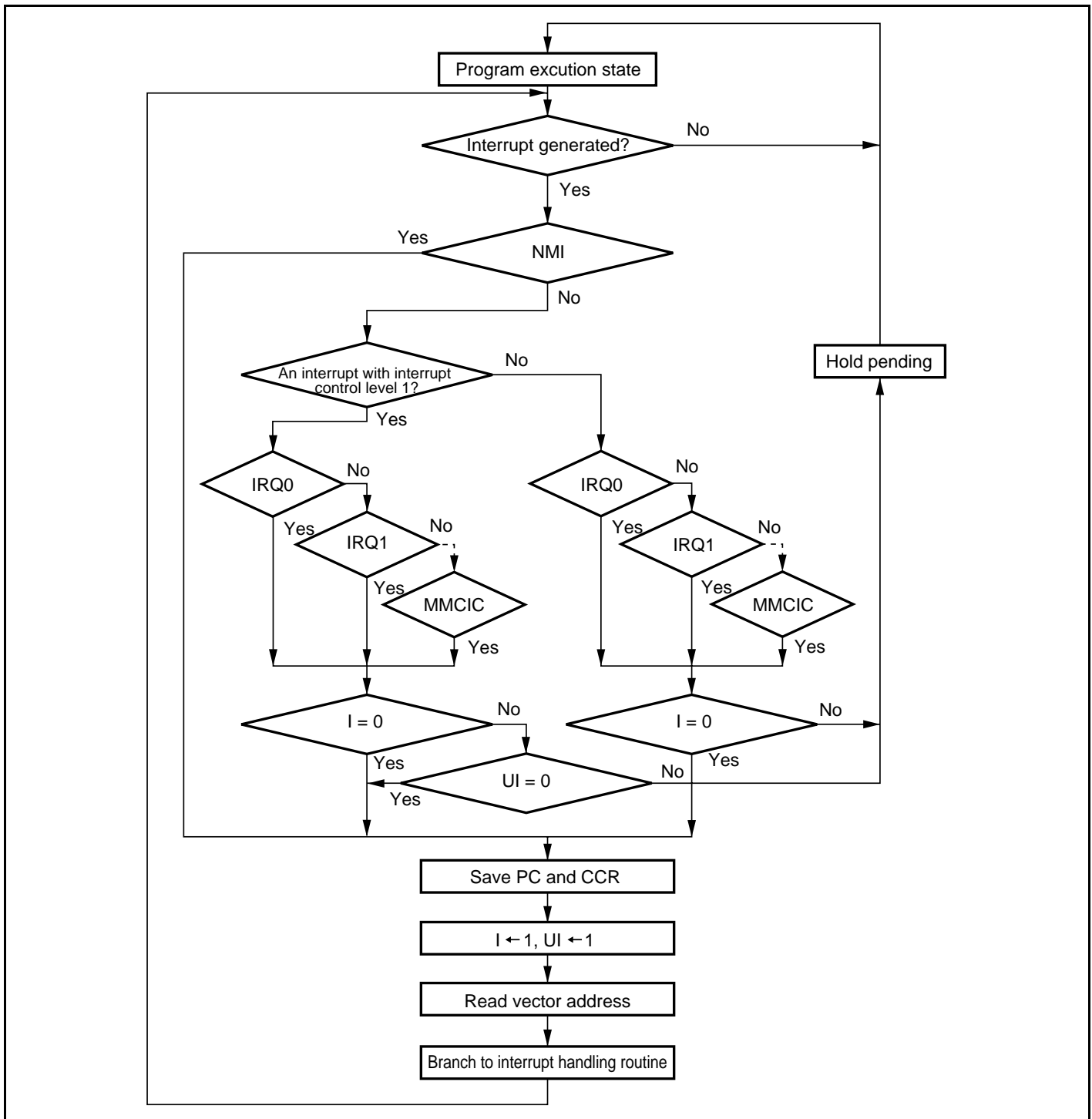
1. If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. According to the interrupt control level specified in ICR, the interrupt controller only accepts an interrupt request with interrupt control level 1 (priority), and holds pending an interrupt request with interrupt control level 0 (no priority). If several interrupt requests are issued, an interrupt request with the highest priority is accepted according to the priority order, an interrupt handling is requested to the CPU, and other interrupt requests are held pending.
3. An interrupt request with interrupt control level 1 is accepted when the I bit is cleared to 0, or when the I bit is set to 1 while the UI bit is cleared to 0.

An interrupt request with interrupt control level 0 is accepted when the I bit is cleared to 0. When the I bit is set to 1, only an NMI or address break interrupt request is accepted, and other interrupts are held pending.

When both the I and UI bits are set to 1, only an NMI or address break interrupt request is accepted, and other interrupts are held pending.

When the I bit is cleared to 0, the UI bit is not affected.

4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
6. The I and UI bits in CCR are set to 1. This masks all interrupts except for an NMI or address break interrupt.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.



**Figure 5.7 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1**

### 5.6.3 Interrupt Exception Handling Sequence

Figure 5.8 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.

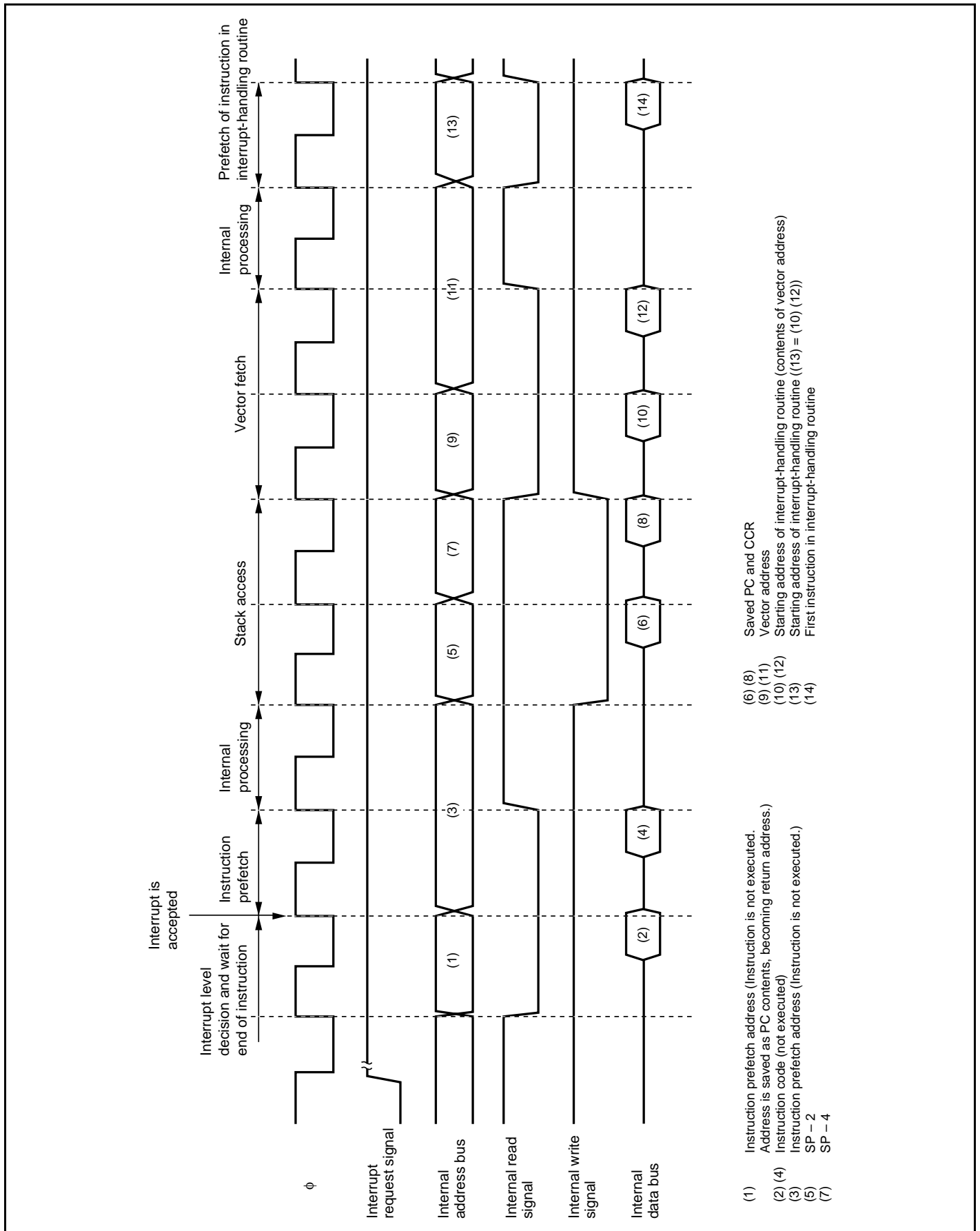


Figure 5.8 Interrupt Exception Handling

## 5.6.4 Interrupt Response Times

Table 5.7 shows interrupt response times—the intervals between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The execution status symbols used in table 5.7 are explained in table 5.8.

**Table 5.7 Interrupt Response Times**

No.	Execution Status	Normal Mode	Advanced Mode
1	Interrupt priority determination* <sup>1</sup>	3	3
2	Number of wait states until executing instruction 1 to ends* <sup>2</sup>	$(19 + 2 \cdot S_i)$	1 to $(19 + 2 \cdot S_i)$
3	PC, CCR stack save	$2 \cdot S_k$	$2 \cdot S_k$
4	Vector fetch	$S_i$	$2 \cdot S_i$
5	Instruction fetch* <sup>3</sup>	$2 \cdot S_i$	$2 \cdot S_i$
6	Internal processing* <sup>4</sup>	2	2
Total (using on-chip memory)		11 to 31	12 to 32

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

**Table 5.8 Number of States in Interrupt Handling Routine Execution Status**

Symbol		Object of Access				
		Internal Memory	External Device		3-State Access	
			2-State Access	3-State Access		2-State Access
Instruction fetch	$S_i$	1	4	$6 + 2m$	2	$3 + m$
Branch address read	$S_j$					
Stack manipulation	$S_k$					

Legend:

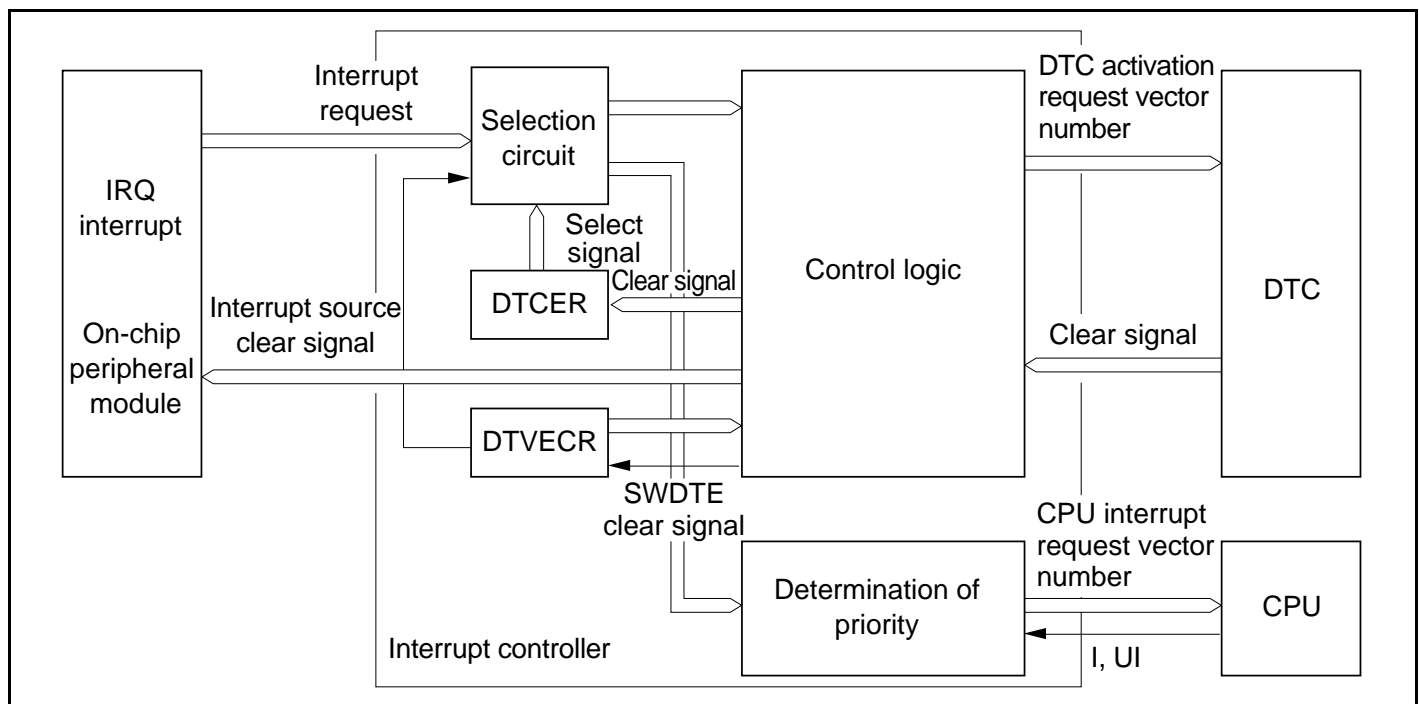
m: Number of wait states in external device access.

### 5.6.5 DTC Activation by Interrupt

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

- Interrupt request to CPU
- Activation request to DTC
- Both of the above

For details of interrupt requests that can be used to activate the DTC, see section 7, Data Transfer Controller (DTC). Figure 5.9 shows a block diagram of the DTC and interrupt controller.



**Figure 5.9 Interrupt Control for DTC**

The interrupt controller has three main functions in DTC control.

**Selection of Interrupt Source:** It is possible to select DTC activation request or CPU interrupt request with the DTCE bit of DTCERA to DTCERE in the DTC. After a DTC data transfer, the DTCE bit can be cleared to 0 and an interrupt request sent to the CPU in accordance with the specification of the DISEL bit of MRB in the DTC. When the DTC performs the specified number of data transfers and the transfer counter reaches 0, following the DTC data transfer the DTCE bit is cleared to 0 and an interrupt request is sent to the CPU.

**Determination of Priority:** The DTC activation source is selected in accordance with the default priority order, and is not affected by mask or priority levels. See section 7.4, Location of Register Information and DTC Vector Table, for the respective priorities.

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

Table 5.9 summarizes interrupt source selection and interrupt source clearance control according to the settings of the DTCE bit of DTCERA to DTCERE in the DTC and the DISEL bit of MRB in the DTC.

**Table 5.9 Interrupt Source Selection and Clearing Control**

Settings		Interrupt Source Selection/Clearing Control	
DTC		DTC	CPU
DTCE	DISEL		
0	*	×	Δ
1	0	Δ	×
	1	○	Δ

Legend:

Δ: The relevant interrupt is used. Interrupt source clearing is performed.  
(The CPU should clear the source flag in the interrupt handling routine.)

○: The relevant interrupt is used. The interrupt source is not cleared.

×: The relevant bit cannot be used.

\*: Don't care

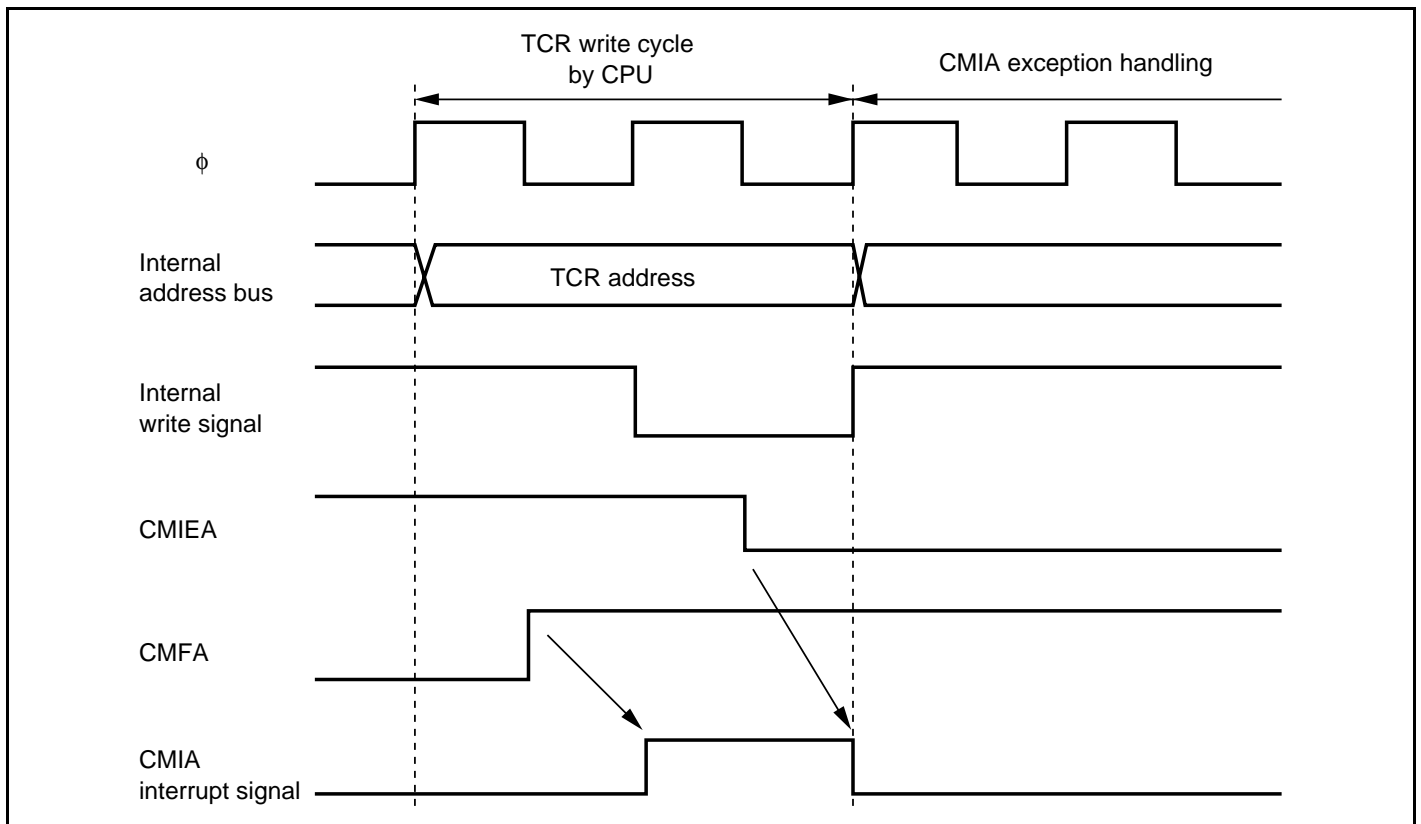


## 5.7 Usage Notes

### 5.7.1 Conflict between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupt requests, the disabling becomes effective after execution of the instruction. When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, and if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored. The same rule is also applied when an interrupt source flag is cleared to 0. Figure 5.10 shows an example in which the CMIEA bit in the TMR's TCR register is cleared to 0.

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



**Figure 5.10 Conflict between Interrupt Generation and Disabling**

### 5.7.2 Instructions that Disable Interrupts

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

### 5.7.3 Interrupts during Execution of EEPMOV Instruction

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

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

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

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

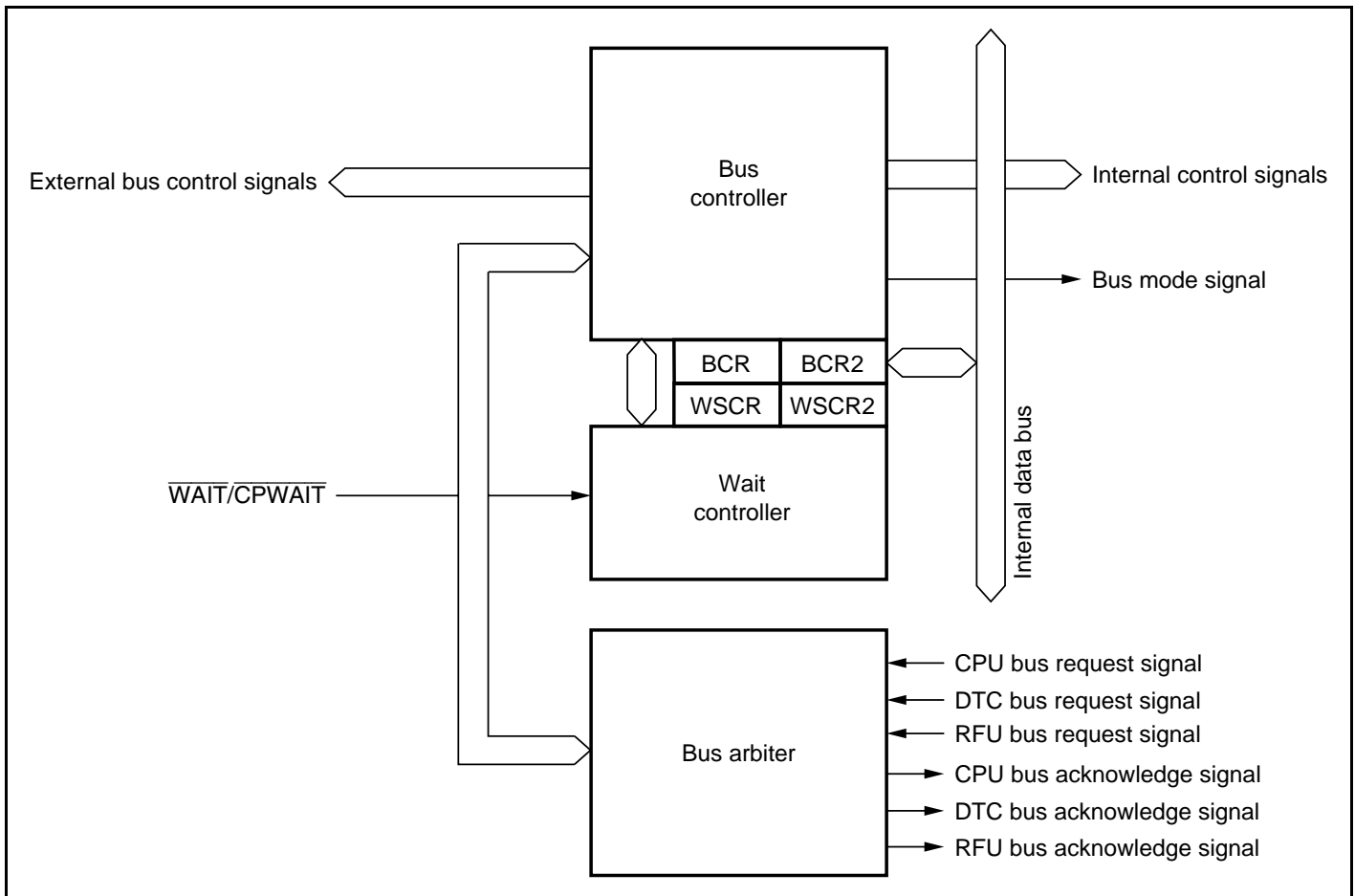
## Section 6 Bus Controller

This LSI has an on-chip bus controller (BSC) that manages the bus width and the number of access states of the external address space. The BSC also has a bus arbitration function, and controls the operation of the internal bus masters—CPU, data transfer controller (DTC), and RAM FIFO unit (RFU).

### 6.1 Features

- Expansion area division
  - The external address space can be accessed as basic expansion areas
  - A 256-kbyte expansion area can be set and controlled independently of basic expansion areas in mode 2 (advanced mode)
  - A CP expansion area can be set and controlled independently of basic expansion areas in mode 2 (advanced mode)
- Address pin reduction
  - A 256-kbyte expansion area from H'F80000 to H'FBFFFF can be selected using 18 address pins and the  $\overline{\text{CS}}_{256}$  signal
  - A CP expansion area (8 kbytes, basic mode) from H'FFC000 to H'FFDFFF can be selected using 13 address pins and the  $\overline{\text{CPCS}}_1$  signal
  - A 2-kbyte area from H'(FF)F000 to H'(FF)F7FF can be selected using six to eleven address pins and the  $\overline{\text{IOS}}$  signal
- Basic bus interface
  - 2-state access or 3-state access can be selected for each area
  - Program wait states can be inserted for each area
- Memory card interface
  - A CompactFlash\* interface can be supported for the CF expansion area (4 kbytes, memory card mode) in the CP expansion area
- Burst ROM interface
  - A burst ROM interface can be set for basic expansion areas
  - 1-state access or 2-state access can be selected for burst access
- Idle cycle insertion
  - An idle cycle can be inserted for external write cycles immediately after external read cycles
- Bus arbitration function
  - Includes a bus arbiter that arbitrates bus mastership between the CPU, DTC, and RFU

Note: \* CompactFlash (CompactFlash™) is a trademark of SanDisk Corporation in the United States, licensed through CFA (CompactFlash™ Association).



**Figure 6.1 Block Diagram of Bus Controller**

## 6.2 Input/Output Pins

Table 6.1 summarizes the pins of the bus controller.

**Table 6.1 Pin Configuration**

Symbol	I/O	Function
$\overline{AS}$	Output	Strobe signal indicating that address output on the address bus is enabled (when the IOSE bit in SYSCR is cleared to 0). Note that this signal is not output (the 256-kbyte expansion area is accessed while the CS256E bit in SYSCR is 1) or when the CP/CF expansion area is accessed (the CPCSE bit in BCR2 is 1).
$\overline{IOS}$	Output	I/O select signal (when the IOSE bit in SYSCR is set to 1).
$\overline{CPCS1}, \overline{CPCS2}$	Output	Chip select signal indicating that the CP/CF expansion area is being accessed (in mode 2 or when the CPCSE bit in BCR2 is set to 1).
$\overline{CS256}$	Output	Chip select signal indicating that the 256-kbyte expansion area is being accessed (in mode 2 or when the CS256E bit in SYSCR is set to 1).
$\overline{RD/CPOE}$	Output	Strobe signal indicating that the external address space is being read.
$\overline{HWR/CPWE}$	Output	Strobe signal indicating that the external address space is being written to, and the upper half (D15 to D8) of the data bus is enabled.  (Note however that the effective data bus must be specified by the $\overline{CPCS1}$ and $\overline{CPCS2}$ signals when the CP/CF expansion area is being accessed.)
$\overline{LWR}$	Output	Strobe signal indicating that the external address space is being written to, and the lower half (D7 to D0) of the data bus is enabled.
$\overline{WAIT/CPWAIT}$	Input	Wait request signal when accessing the external 3-state access space or CP/CF expansion area.

## 6.3 Register Descriptions

The bus controller has the following registers. For details on the system control register, see section 3.2.2, System Control Register (SYSCR).

- Bus control register (BCR)
- Bus control register 2 (BCR2)
- Wait state control register (WSCR)
- Wait state control register 2 (WSCR2)

### 6.3.1 Bus Control Register (BCR)

BCR is used to specify the access mode for the external address space or the I/O area range when the  $\overline{AS}/\overline{IOS}$  pin is specified as an I/O strobe pin.

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R/(W)	Reserved The initial value should not be changed.
6	ICIS	1	R/W	Idle Cycle Insertion Selects whether or not to insert 1-state of the idle cycle between successive external read and external write cycles. 0: Idle cycle not inserted 1: 1-state idle cycle inserted
5	BRSTRM	0	R/W	Burst ROM Enable Selects the bus interface for the external address space. 0: Basic bus interface 1: Burst ROM interface When the CS256E bit in SYSCR and the CPCSE bit in BCR2 are set to 1, burst ROM interface cannot be selected for the 256-kbyte expansion area and CP/CF expansion area.

Bit	Bit Name	Initial Value	R/W	Description
4	BRSTS1	1	R/W	<p>Burst Cycle Select 1</p> <p>Selects the number of states in the burst cycle of the burst ROM interface.</p> <p>0: 1 state 1: 2 states</p>
3	BRSTS0	0	R/W	<p>Burst Cycle Select 0</p> <p>Selects the number of words that can be accessed by burst access via the burst ROM interface.</p> <p>0: Max. 4 words 1: Max. 8 words</p>
2	CFE	0	R/W	<p>CF Expansion Area Enable</p> <p>Selects the CP/CF expansion area to be accessed when the CPCSE bit in BCR2 is set to 1. For details, see table 6.2.</p> <p>0: CP expansion area (basic mode) 1: CF expansion area (memory card mode)</p>
1	IOS1	1	R/W	IOS Select 1, 0
0	IOS0	1	R/W	Select the address range where the $\overline{\text{IOS}}$ signal is output. For details, refer to table 6.8.

### 6.3.2 Bus Control Register 2 (BCR2)

BCR2 is used to specify the access mode for the CP expansion area (basic mode) and CF expansion area (memory card mode).

Bit	Bit Name	Initial Value	R/W	Description
7	OWEAC	0	R/W	<p>OE/WE Assert Control</p> <p>Specifies the number of cycles from address output to the <math>\overline{CPOE}</math> and <math>\overline{CPWE}</math> signal assertion when the CF expansion area is specified as the CP expansion area.</p> <p>0: 0.5 cycles 1: 1.5 cycles</p> <p>If the ASTCP bit is cleared to 0, this bit must not be set to 1.</p>
6	OWENC	0	R/W	<p>OE/WE Negate Control</p> <p>Specifies the number of delay cycles from <math>\overline{CPOE}</math> and <math>\overline{CPWE}</math> signal negation to address hold when the CF expansion area is specified as the CP expansion area.</p> <p>0: 0.5 cycles 1: 1.5 cycles</p> <p>If the ASTCP bit is cleared to 0, this bit must not be set to 1.</p>
5	ABWCP	1	R/W	<p>CP Expansion Area Bus Width Control</p> <p>Selects the bus width for access to the CP expansion area when the CPCSE bit in BCR2 is set to 1 while the CFE bit in BCR is cleared to 0. When the CPCSE bit in BCR2 is set to 1 while the CFE bit in BCR is set to 1, the bus width for access to the CF expansion area is fixed at 16 bits.</p> <p>0: 16-bit bus 1: 8-bit bus</p>



Bit	Bit Name	Initial Value	R/W	Description
4	ASTCP	1	R/W	<p>CP/CF Expansion Area Access State Control</p> <p>Selects the number of states for access to the CP/CF expansion area when the CPCSE bit in BCR2 is set to 1. This bit also enables or disables wait-state insertion.</p> <p>0: 2-state access space. Wait state insertion disabled in CP/CF expansion area access</p> <p>1: 3-state access space. Wait state insertion enabled in CP/CF expansion area access</p>
3	ADFULLE	0	R/W	<p>Address Output Full Enable</p> <p>Controls the <math>\overline{IOS}</math> signal output and address output in access to the 256-kbyte expansion area and CP/CF expansion area. For details, refer to section 9, I/O Ports.</p>
2	EXCKS	0	R/W	<p>External Expansion Clock Select</p> <p>Selects the operating clock used in external expansion area access.</p> <p>0: Medium-speed clock is selected as the operating clock</p> <p>1: System clock (<math>\phi</math>) is selected as the operating clock. The operating clock is switched in the bus cycle prior to external expansion area access.</p>
1	BUSDIVE	1	R/W	<p>Bus Division Arbitration Enable</p> <p>Controls the bus arbitration timing for the divided bus cycles in the RFU operation. For details, refer to section 8, RAM FIFO Unit (RFU).</p>
0	CPCSE	0	R/W	<p>CP/CF Expansion Area Enable</p> <p>Selects the expansion area to be accessed.</p> <p>0: External address space (basic expansion area)</p> <p>1: CP/CF expansion area (basic mode when CFE bit in BCR is 0, memory card mode when CFE bit in BCR is 1)</p>

### 6.3.3 Wait State Control Register (WSCR)

WSCR is used to specify the data bus width for external address space access, the number of access states, the wait mode, and the number of wait states for access to external address spaces (basic expansion area and 256-kbyte expansion area). The bus width and the number of access states for internal memory and internal I/O registers are fixed regardless of the WSCR settings.

Bit	Bit Name	Initial Value	R/W	Description
7	ABW256	1	R/W	<p>256-kbyte Expansion Area Bus Width Control</p> <p>Selects the bus width for access to the 256-kbyte expansion area when the CS256E bit in SYSCR is set to 1.</p> <p>0: 16-bit bus 1: 8-bit bus</p>
6	AST256	1	R/W	<p>256-kbyte Expansion Area Access State Control</p> <p>Selects the number of states for access to the 256-kbyte expansion area when the CS256E bit in SYSCR is set to 1. This bit also enables or disables wait-state insertion.</p> <p>0: 2-state access space. Wait state insertion disabled in 256-kbyte expansion area access 1: 3-state access space. Wait state insertion enabled in 256-kbyte expansion area access</p>
5	ABW	1	R/W	<p>Bus Width Control</p> <p>Selects the bus width for access to the basic expansion area.</p> <p>0: 16-bit bus 1: 8-bit bus</p> <p>When the CS256E bit in SYSCR and the CPCSE bit in BCR2 are set to 1, this bit setting is ignored in 256-kbyte expansion area access and CP/CF expansion area access.</p>

Bit	Bit Name	Initial Value	R/W	Description
4	AST	1	R/W	<p>Access State Control</p> <p>Selects the number of states for access to the basic expansion area. This bit also enables or disables wait-state insertion.</p> <p>0: 2-state access space. Wait state insertion disabled in basic expansion area access</p> <p>1: 3-state access space. Wait state insertion enabled in basic expansion area access</p> <p>When the CS256E bit in SYSCR and the CPCSE bit in BCR2 are set to 1, this bit setting is ignored in 256-kbyte expansion area access and CP/CF expansion area access.</p>
3	WMS21	0	R/W	Wait Mode Select 1, 0
2	WMS20	0	R/W	<p>Select the wait mode for access to the basic expansion area when the AST bit is set to 1.</p> <p>00: Program wait mode</p> <p>01: Wait disabled mode</p> <p>10: Pin wait mode</p> <p>11: Pin auto-wait mode</p> <p>When the CS256E bit in SYSCR and the CPCSE bit in BCR2 are set to 1, this bit setting is ignored in 256-kbyte expansion area access and CP/CF expansion area access.</p>
1	WC1	1	R/W	Wait Count 1, 0
0	WC0	1	R/W	<p>Select the number of program wait states to be inserted when the basic expansion area is accessed while the AST bit is set to 1.</p> <p>00: Program wait state is not inserted</p> <p>01: 1 program wait state is inserted</p> <p>10: 2 program wait states are inserted</p> <p>11: 3 program wait states are inserted</p> <p>When the CS256E bit in SYSCR and the CPCSE bit in BCR2 are set to 1, this bit setting is ignored in 256-kbyte expansion area access and CP/CF expansion area access.</p>

### 6.3.4 Wait State Control Register 2 (WSCR2)

WSCR2 is used to specify the wait mode and number of wait states in access to the 256-kbyte expansion area and CP/CF expansion area.

Bit	Bit Name	Initial Value	R/W	Description
7	WMS10	0	R/W	<p>256-kbyte Expansion Area Wait Mode Select 0</p> <p>Selects the wait mode for access to the 256-kbyte expansion area when the CS256E bit in SYSCR and the AST256 bit in WSCR are set to 1.</p> <p>0: Program wait mode 1: Wait disabled mode</p>
6	WC11	1	R/W	256-kbyte Expansion Area Wait Count 1, 0
5	WC10	1	R/W	<p>Select the number of program wait states to be inserted for access to the 256-kbyte expansion area when the CS256E bit in SYSCR and the AST256 bit in WSCR are set to 1.</p> <p>00: Program wait state is not inserted 01: 1 program wait state is inserted 10: 2 program wait states are inserted 11: 3 program wait states are inserted</p>
4	WMS21	0	R/W	CP/CF Expansion Area Wait Mode Select 1, 0
3	WMS20	0	R/W	<p>Select the wait mode for access to the CP/CF expansion area when the CPCSE and ASTCP bits in BCR2 are set to 1.</p> <p>00: Program wait mode 01: Wait disabled mode 10: Pin wait mode 11: Pin auto-wait mode (only in CP expansion area access)</p>

Bit	Bit Name	Initial Value	R/W	Description
2	WC22	1	R/W	CP/CF Expansion Area Wait Count 2–0
1	WC21	1	R/W	Select the number of program wait states to be inserted for access to the CP/CF expansion area when the CPCSE and ASTCP bits in BCR2 are set to 1.  If the CP expansion area is selected, the WC22 bit must be cleared to 0.  000: Program wait state is not inserted 001: 1 program wait state is inserted 010: 2 program wait states are inserted 011: 3 program wait states are inserted 100: 4 program wait states are inserted (only for CF expansion area) 101: 6 program wait states are inserted (only for CF expansion area) 110: 8 program wait states are inserted (only for CF expansion area) 111: 10 program wait states are inserted (only for CF expansion area)
0	WC20	1	R/W	

## 6.4 Bus Control

### 6.4.1 Bus Specifications

The external address space bus specifications consist of three elements: Bus width, the number of access states, and the wait mode and the number of program wait states. The bus width and the number of access states for on-chip memory and internal I/O registers are fixed, and are not affected by the bus controller settings.

**Bus Width:** A bus width of 8 or 16 bits can be selected via the ABW and ABW256 bits in WSCR, and the ABWCP bit in BCR2. If memory card mode is selected when the CFE bit in BCR is set to 1, a 16-bit bus is automatically selected for CP expansion area access.

**Number of Access States:** Two or three access states can be selected via the AST and AST256 bits in WSCR, and the ASTCP bit in BCR2. When the 2-state access space is designated, wait-state insertion is disabled.

In the burst ROM interface, the number of access states for the basic expansion area is determined regardless of the AST bit setting.

**Wait Mode and Number of Program Wait States:** When a 3-state access space is designated by the AST bit in WSCR, the wait mode and the number of program wait states to be inserted automatically is selected by the WMS1, WMS0, WC1, and WC0 bits in WSCR. From 0 to 3 program wait states can be selected.

When the 256-kbyte expansion area is specified as a 3-state access space by the AST256 bit in WSCR, the wait mode and the number of program wait states to be inserted automatically is selected by the WMS10, WC11, and WC10 bits in WSCR2. From 0 to 3 program wait states can be selected.

When the CP/CF expansion area is specified as a 3-state access space by the ASTCP bit in BCR2, the wait mode and the number of program wait states to be inserted automatically is selected by the WMS21, WMS20, WC21, and WC20 bits in WSCR2. From 0 to 3 program wait states can be selected.

When the CP expansion area is set to the CF expansion area (memory card mode) by the CFE bit in BCR, the wait mode and the number of program wait states to be inserted automatically is selected by the WMS21, WMS20, WC22, WC21, and WC20 bits in WSCR2. From 0 to 4, 6, 8, or 10 program wait states can be selected.

The wait function for external expansion is effective for connecting low-speed devices to the external address space. However, this wait function may cause some problems when bus masters other than the CPU, such as the DTC and RFU are to be delayed. The RFU is mostly used for data transmission and reception of interface peripheral modules. In this case, the wait function for external expansion may cause difficulty in making data transfer satisfy the communication rate of transmit/receive data of the interface peripheral modules. To prevent such a problem, it is recommended that when using the RFT, the external address space should be designated as a 2-state or 3-state access space with no wait state.

Tables 6.2 to 6.7 show each bit setting and external address space division in the address ranges of the external address space, and the bus specifications for the basic bus interface of each area.

**Table 6.2 Address Ranges and External Address Spaces**

Address Range	Basic Expansion Area	Areas
		256-kbyte Expansion Area, CP Expansion Area (Basic Mode), CF Expansion Area (Memory Card Mode)
H'080000–H'F7FFFF (15 Mbytes)	○ No condition	—
H'F80000–H'FBFFFF (256 kbytes) 256-kbyte expansion area	△ When CS256E = 0, used as basic expansion area.	When $\overline{\text{WAIT}}/\overline{\text{CPWAIT}}$ pin function is not selected while CS256E = 1, $\overline{\text{CS256}}$ is output and address pins A17 to A0 are used.
H'FC0000–H'FEFFFF (192 kbytes)	○ No condition	—
H'FF0800–H'FF7FFF	△ When RAME = 0, used as basic expansion area.	—
H'FF8000–H'FFBFFF (16 kbytes)	○ No condition	—
H'FFC000–H'FFCFFF (4 kbytes) CP expansion area (1), CF expansion area	△ When CPCSE = 0, used as basic expansion area.	When CPCSE = 1 while CFE = 0, $\overline{\text{CPCS1}}$ is output in the CP expansion area and address pins A12 to A0 are used.  When CPCSE = 1 while CFE = 1, $\overline{\text{CPCS1}}$ and $\overline{\text{CPCS2}}$ are output in the CF expansion area and pins $\overline{\text{CPREG}}$ and CAP10 to CAP0 are used.
H'FFD000–H'FFDFFF (4 kbytes) CP expansion area (2)	△ When CPCSE = 0, used as basic expansion area.	When CPCSE = 1 while CFE = 0, $\overline{\text{CPCS1}}$ is output in the CP expansion area and address pins A12 to A0 are used.  When CPCSE = 1 while CFE = 1, this address area cannot be used.
H'(FF)E000–H'(FF)E07F (128 bytes)	○ No condition	—
H'(FF)E080–H'(FF)EFFF (3968 bytes)	△ When RAME = 0, used as basic expansion area.	—

Address Range	Basic Expansion Area	Areas
		256-kbyte Expansion Area, CP Expansion Area (Basic Mode), CF Expansion Area (Memory Card Mode)
H'(FF)F000–H'(FF)F7FF (2 kbytes)	○ No condition When IOSE = 1, $\overline{\text{IOS}}$ is output and address pins A10 to A0 are used.	—
H'(FF)FF00–H'(FF)FF7F (128 bytes)	△ When RAME = 0, used as basic expansion area.	—

## Legend:

○: This address range unconditionally becomes the basic expansion area when it is accessed.

△: Condition for making this address range included in the basic expansion area when it is accessed.

—: Irrelevant address range.



**Table 6.3 Bit Settings and Bus Specifications of Basic Bus Interface**

				Areas		
BRSTRM	CS256E	CPCSE	CFE	Basic Expansion Area	256-kbyte Expansion Area	CP Expansion Area (Basic Mode) and CF Expansion Area (Memory Card Mode)
0	0	0	—	Basic expansion area ABW, AST, WMS1, WMS0, WC1, WC0	Used as basic expansion area	Used as basic expansion area
			1			0
			1			Memory card mode WMS21, WMS20, WC22, WC21, WC20
	1	0	—		ABW256, AST256, WMS10, WC11, WC10	Same as when CS256E = 0
			1	0		
				1		
1	0	0	—	Burst ROM interface* ABW, AST, WMS0, WC1, WC0, BRSTS1, BRSTS0	Used as burst ROM interface	Used as burst ROM interface
			1			0
			1			Memory card mode WMS21, WMS20, WC22, WC21, WC20
	1	0	—		ABW256, AST256, WMS10, WC11, WC10	Same as when CS256E = 0
			1	0		
				1		

Legend:

—: Don't care

Note: \* In the burst ROM interface, the bus width is specified by the ABW bit in WSCR, the number of full access states (wait can be inserted) is specified by the AST bit in WSCR, and the number of access cycles in burst access is specified regardless of the AST bit setting.

**Table 6.4 Bus Specifications for Basic Expansion Area/Basic Bus Interface**

ABW	AST	WMS1	WMS0	WC1	WC0	Bus Specifications		
						Bus Width	Number of Access States	Number of Program Wait States
0	0	—	—	—	—	16	2	0
	1	0	1	—	—	16	3	0
		Other than WMS1 = 0 and WMS0 = 1		0	0		3	0
					1			1
				1	0			2
			1			3		
1	0	—	—	—	—	8	2	0
	1	0	1	—	—	8	3	0
		Other than WMS1 = 0 and WMS0 = 1		0	0		3	0
					1			1
				1	0			2
			1			3		

Legend:

—: Don't care

**Table 6.5 Bus Specifications for 256-kbyte Expansion Area/Basic Bus Interface**

					Bus Specifications		
ABW256	AST256	WMS10	WC11	WC10	Bus Width	Number of Access States	Number of Program Wait States
0	0	—	—	—	16	2	0
	1	1	—	—	16	3	0
		0	0	0		3	0
				1			1
				1		0	2
			1	3			
1	0	—	—	—	8	2	0
	1	1	—	—	8	3	0
		0	0	0		3	0
				1			1
				1		0	2
			1	3			

Legend:

—: Don't care

**Table 6.6 Bus Specifications for CP Expansion Area (Basic Mode)/Basic Bus Interface**

						Bus Specifications		
ABWCP	ASTCP	WMS21	WMS20	WC21	WC20	Bus Width	Number of Access States	Number of Program Wait States
0	0	—	—	—	—	16	2	0
	1	0	1	—	—	16	3	0
		Other than WMS21 = 0 and WMS20 = 1		0	0		3	0
					1			1
				1	0			2
							1	3
1	0	—	—	—	—	8	2	0
	1	0	1	—	—	8	3	0
		Other than WMS21 = 0 and WMS20 = 1		0	0		3	0
					1			1
				1	0			2
							1	3

Legend:

—: Don't care

**Table 6.7 Bus Specifications for CF Expansion Area (Memory Card Mode)/Basic Bus Interface**

ASTCP	WMS21	WMS20	WC22	WC21	WC20	Bus Specifications			
						Bus Width	Number of Access States	Number of Program Wait States	
0	—	—	—	—	—	16	2	0	
1	0	1	—	—	—	16	3	0	
	Other than		0	0	0		3	0	
	WMS21 = 0 and						1	1	
	WMS20 = 1 or						1	0	2
	WMS21 = 1 and							1	3
	WMS20 = 1						1	0	4
								1	6
		1				0	8		
			1	10					

Legend:

—: Don't care

### 6.4.2 Advanced Mode

This LSI cannot output upper addresses (A23 to A18) in mode 2 (advanced mode) because this LSI has 18 address output pins (A17 to A0). Therefore, the external address space (H'FFF000 to H'FFF7FF) can be accessed by specifying the  $\overline{AS}/\overline{IOS}$  pin as an I/O strobe pin. The 256-kbyte expansion area (H'F80000 to H'FBFFFF) and CP expansion area (H'FFC000 to H'FFDFFF) can be accessed by the  $\overline{CS256}$  pin and  $\overline{CPCS1}$  pin functions, respectively.

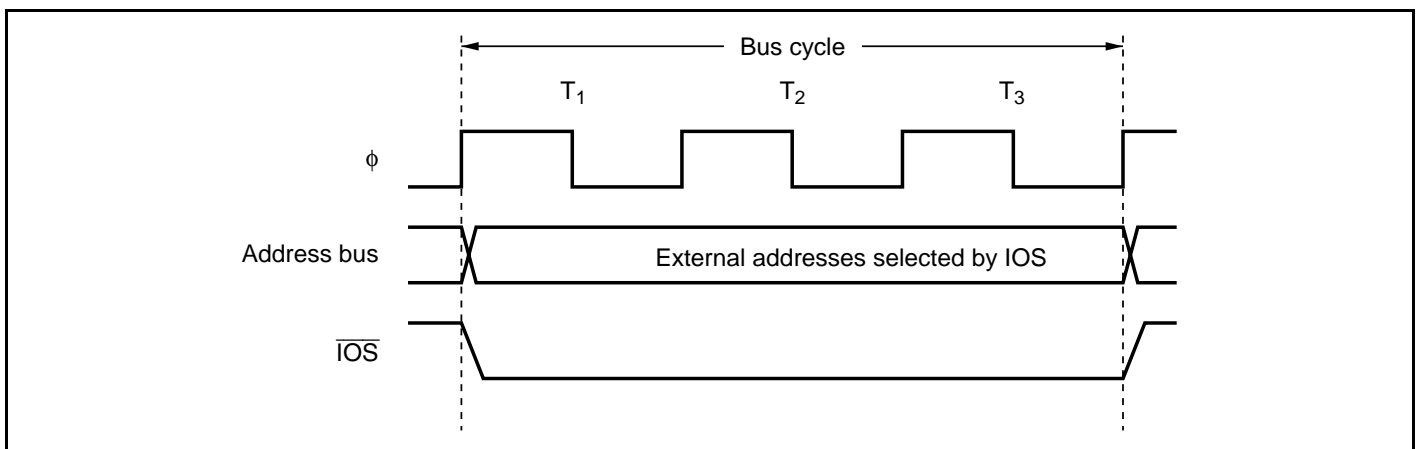
The external address space is initialized as the basic bus interface and a 3-state access space. In mode 2, the address space other than on-chip ROM, on-chip RAM, internal I/O registers, and their reserved areas is specified as the external address space. The on-chip RAM and its reserved area are enabled when the RAME bit in SYSCR is set to 1, and disabled when the RAME bit is cleared to 0. Addresses H'FF0800 to H'FF7FFF, H'FFE080 to H'FFEFFF, and H'FFFF00 to H'FFFF7F in the on-chip RAM area and its reserved area are always specified as the external address space.

### 6.4.3 Normal Mode

The external address space is initialized as the basic bus interface and a 3-state access space. In mode 3 (normal mode), the address space other than on-chip ROM, on-chip RAM, internal I/O registers, and their reserved areas is specified as the external address space. The on-chip RAM area is enabled when the RAME bit in SYSCR is set to 1, and disabled and specified as the external address space when the RAME bit is cleared to 0.

### 6.4.4 I/O Select Signals

The LSI can output I/O select signals ( $\overline{\text{IOS}}$ ); the signal is driven low when the corresponding external address space is accessed. Figure 6.2 shows an example of  $\overline{\text{IOS}}$  signal output timing.



**Figure 6.2  $\overline{\text{IOS}}$  Signal Output Timing**

Enabling or disabling  $\overline{\text{IOS}}$  signal output is performed by the IOSE bit in SYSCR. In extended mode, the  $\overline{\text{IOS}}$  pin functions as an  $\overline{\text{AS}}$  pin by a reset. To use this pin as an  $\overline{\text{IOS}}$  pin, set the IOSE bit to 1. For details, refer to section 9, I/O Ports.

The address ranges of the  $\overline{\text{IOS}}$  signal output can be specified by the IOS1 and IOS0 bits in BCR, as shown in table 6.8.

**Table 6.8 Address Range for  $\overline{\text{IOS}}$  Signal Output**

IOS1	IOS0	$\overline{\text{IOS}}$ Signal Output Range
0	0	H'(FF)F000 to H'(FF)F03F
	1	H'(FF)F000 to H'(FF)FOFF
1	0	H'(FF)F000 to H'(FF)F3FF
	1	H'(FF)F000 to H'(FF)F7FF (Initial value)

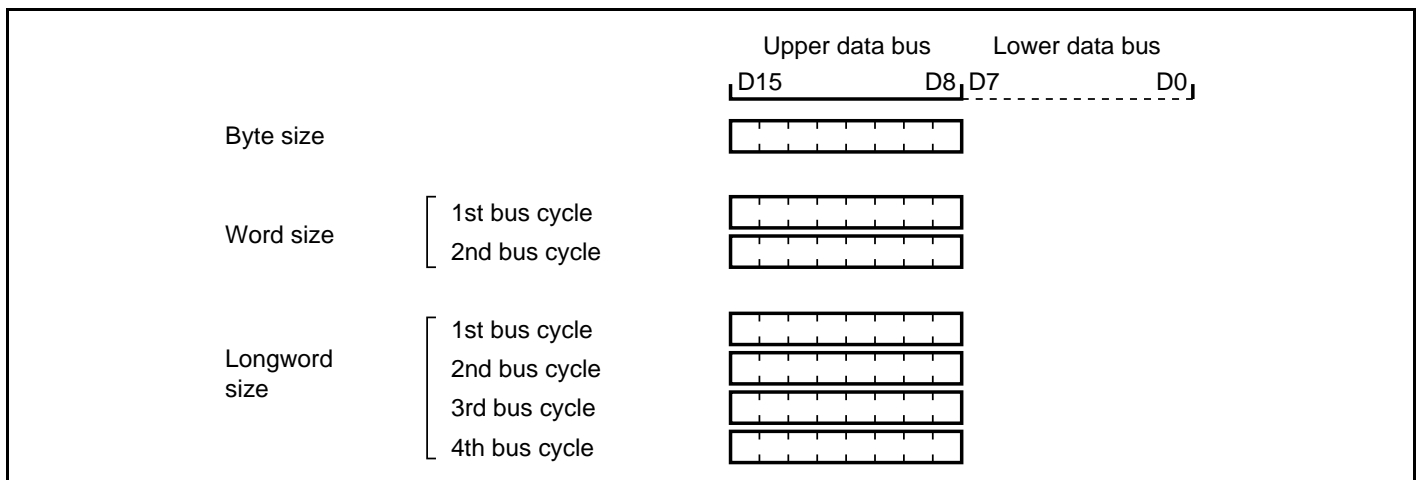
## 6.5 Basic Bus Interface

The basic bus interface enables direct connection to ROM and SRAM. For details on selection of the bus specifications for the basic expansion area, 256-kbyte expansion area, and CP/CF expansion area when using the basic bus interface, see tables 6.4 to 6.6.

### 6.5.1 Data Size and Data Alignment

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

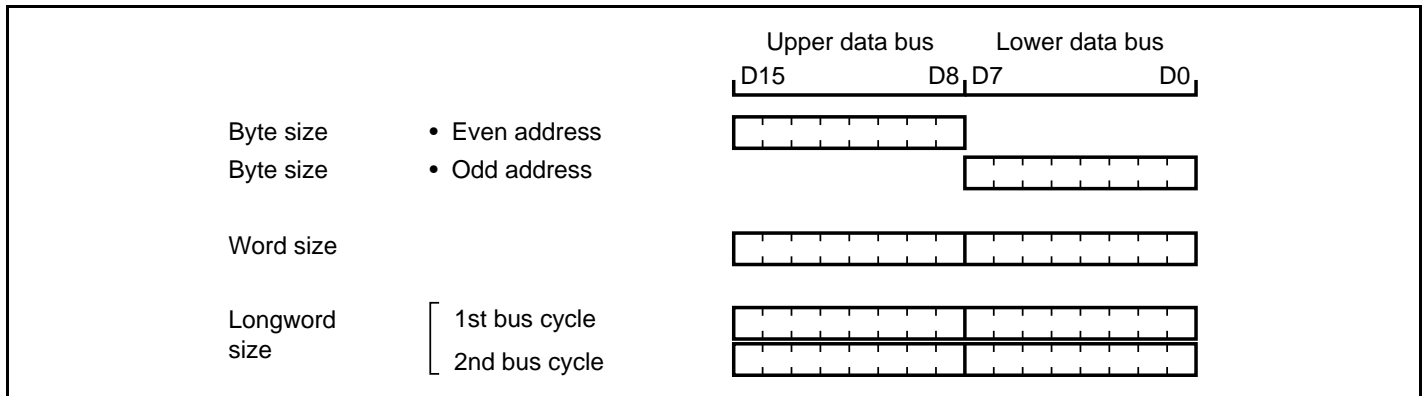
**8-Bit Access Space:** Figure 6.3 illustrates data alignment control for the 8-bit access space. With the 8-bit access space, the upper data bus (D15 to D8) is always used for accesses. The amount of data that can be accessed at one time is one byte: a word access is performed as two byte accesses, and a longword access, as four byte accesses.



**Figure 6.3 Access Sizes and Data Alignment Control (8-Bit Access Space)**

**16-Bit Access Space:** Figure 6.4 illustrates data alignment control for the 16-bit access space. With the 16-bit access space, the upper data bus (D15 to D8) and lower data bus (D7 to D0) are used for accesses. The amount of data that can be accessed at one time is one byte or one word, and a longword access is executed as two word accesses.

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



**Figure 6.4 Access Sizes and Data Alignment Control (16-bit Access Space)**

### 6.5.2 Valid Strobes

Table 6.9 shows the data buses used and valid strobes for each access space.

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

**Table 6.9 Data Buses Used and Valid Strobes**

Area	Access Size	Read/Write	Address	Valid Strobe	Upper Data Bus (D15 to D8)	Lower Data Bus (D7 to D0)
8-bit access space	Byte	Read	—	$\overline{RD}$	Valid	Ports or others
		Write	—	$\overline{HWR}$		Ports or others
16-bit access space	Byte	Read	Even	$\overline{RD}$	Valid	Invalid
			Odd		Invalid	Valid
	Write	Even	$\overline{HWR}$	Valid	Undefined	
		Odd	$\overline{LWR}$	Undefined	Valid	
Word	Read	—	$\overline{RD}$	Valid	Valid	
	Write	—		$\overline{HWR}$ , $\overline{LWR}$	Valid	Valid

Notes: Undefined : Undefined data is output.

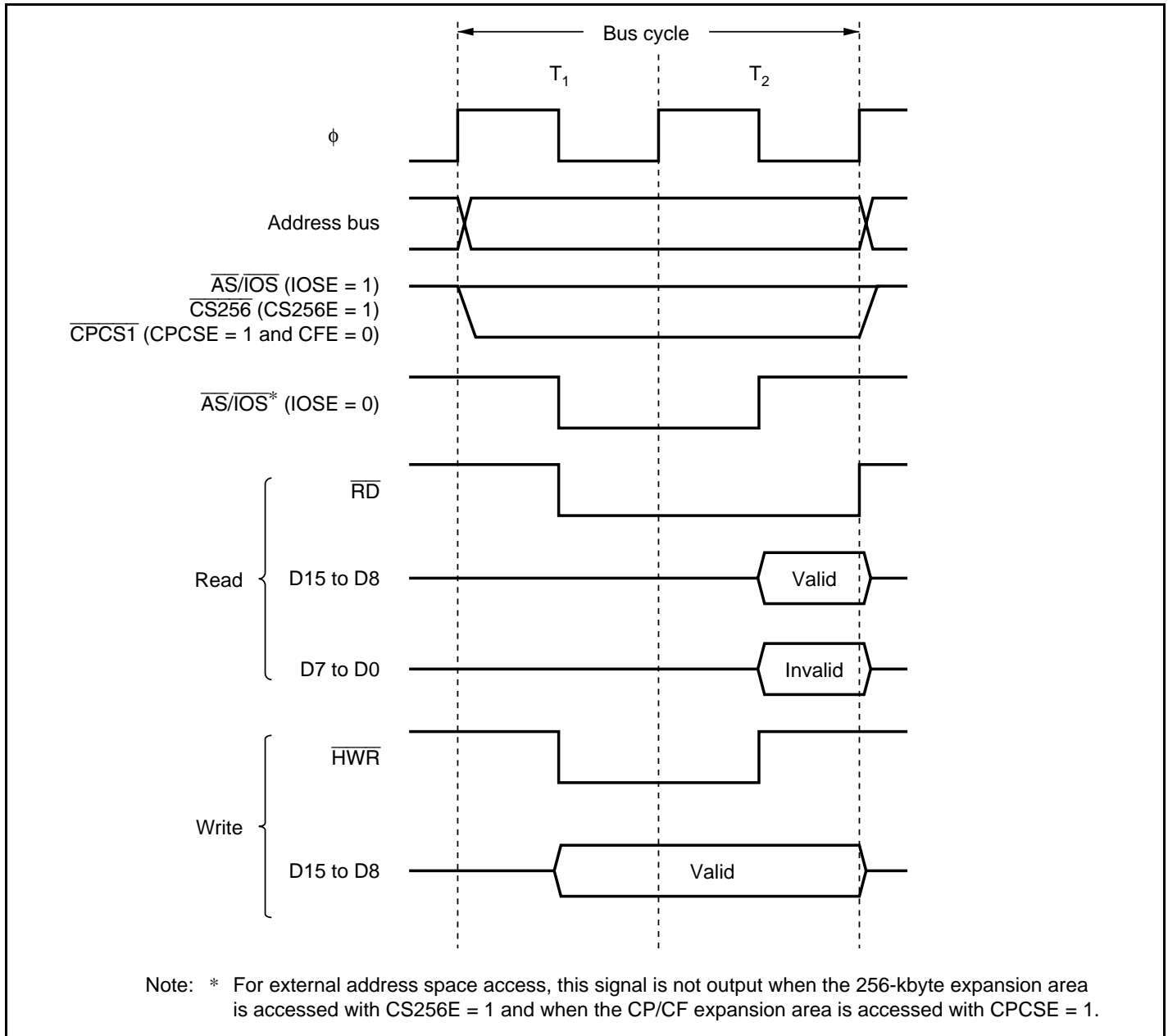
Invalid : Input state with the input value ignored.

Ports or others : Used as ports or I/O pins for on-chip peripheral modules, and are not used as the data bus.



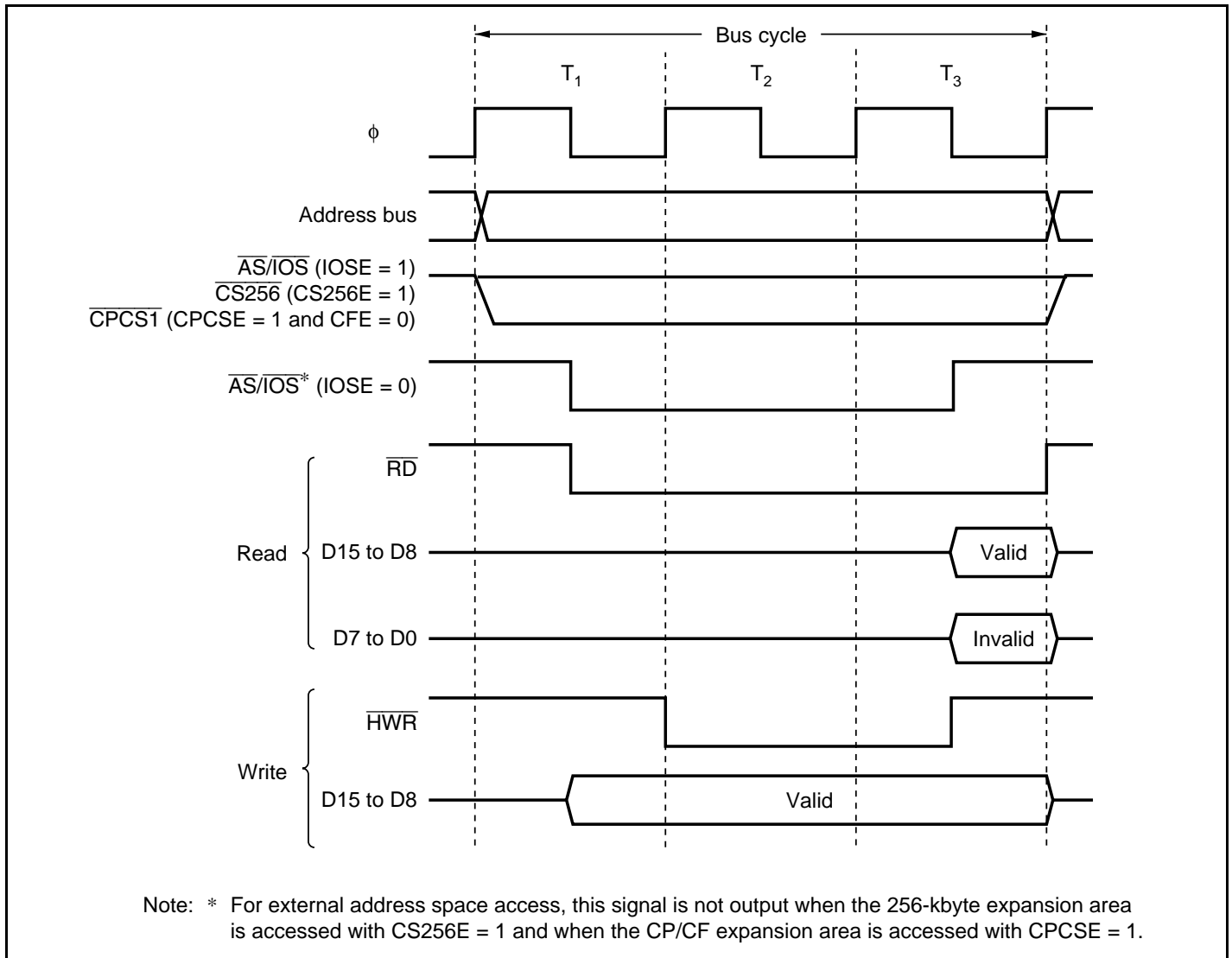
### 6.5.3 Basic Operation Timing

**8-Bit, 2-State Access Space:** Figure 6.5 shows the bus timing for an 8-bit, 2-state access space. When an 8-bit access space is accessed, the upper half (D15 to D8) of the data bus is used. Wait states cannot be inserted.



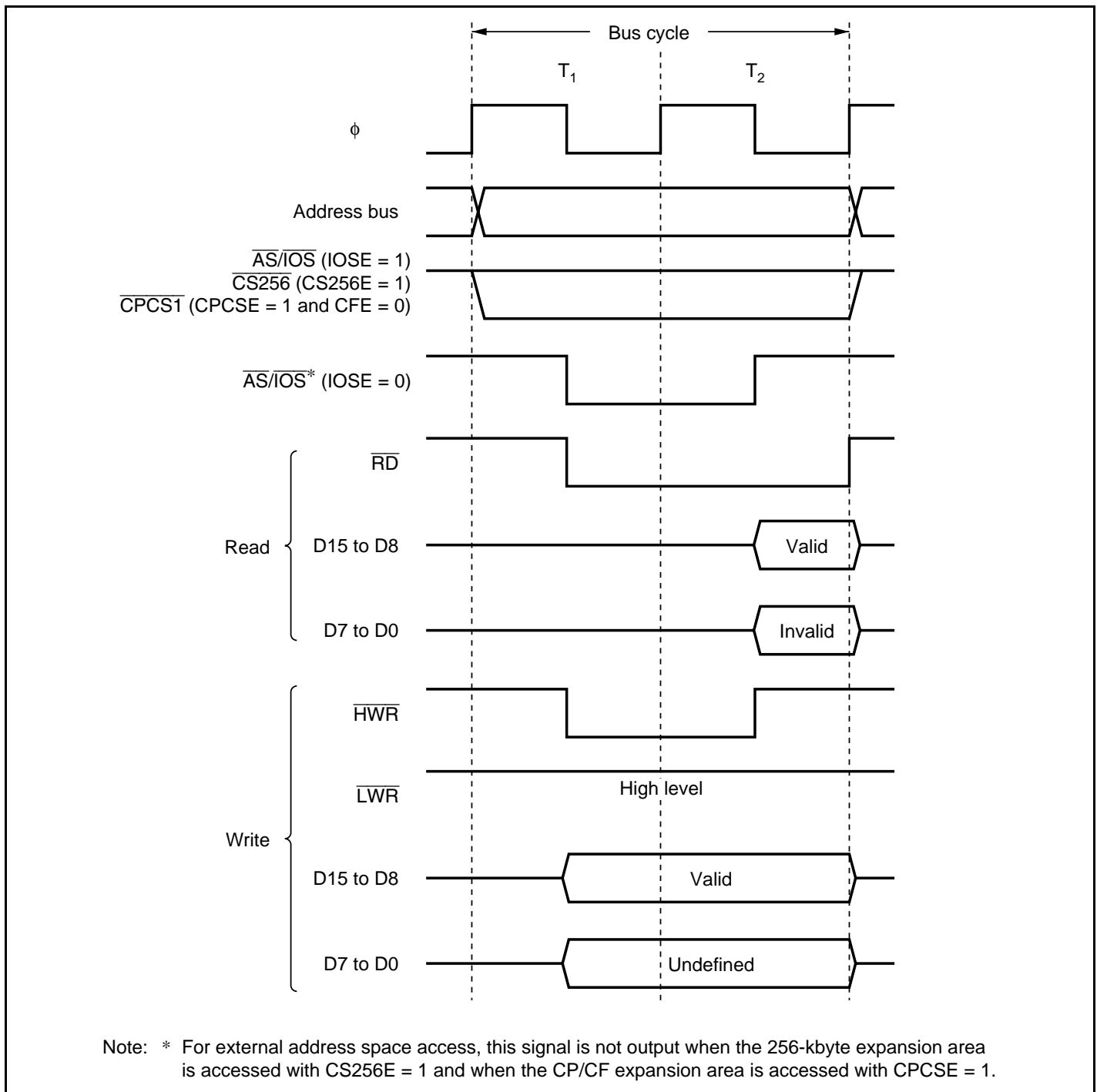
**Figure 6.5 Bus Timing for 8-Bit, 2-State Access Space**

**8-Bit, 3-State Access Space:** Figure 6.6 shows the bus timing for an 8-bit, 3-state access space. When an 8-bit access space is accessed, the upper half (D15 to D8) of the data bus is used. Wait states can be inserted.

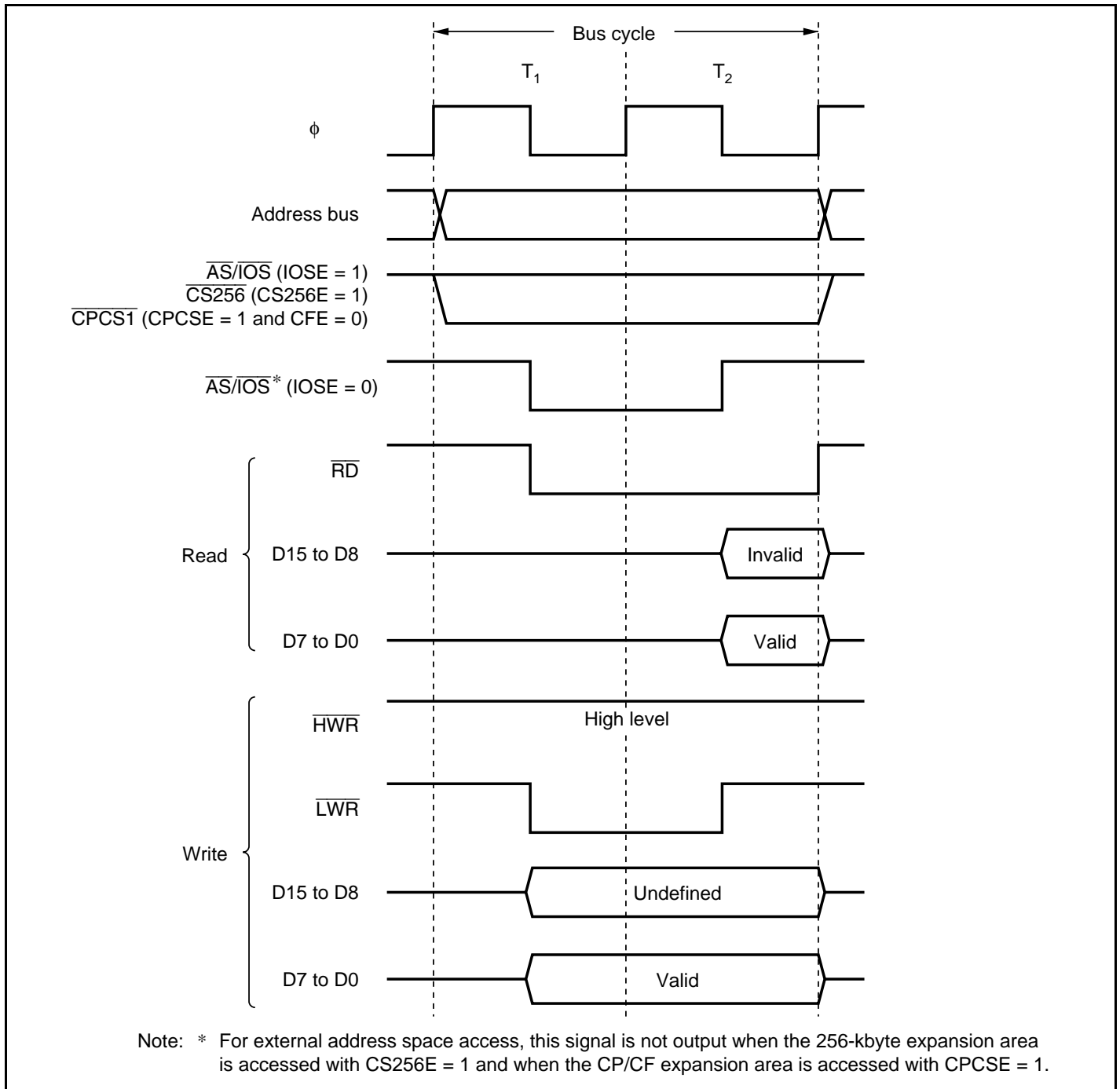


**Figure 6.6 Bus Timing for 8-Bit, 3-State Access Space**

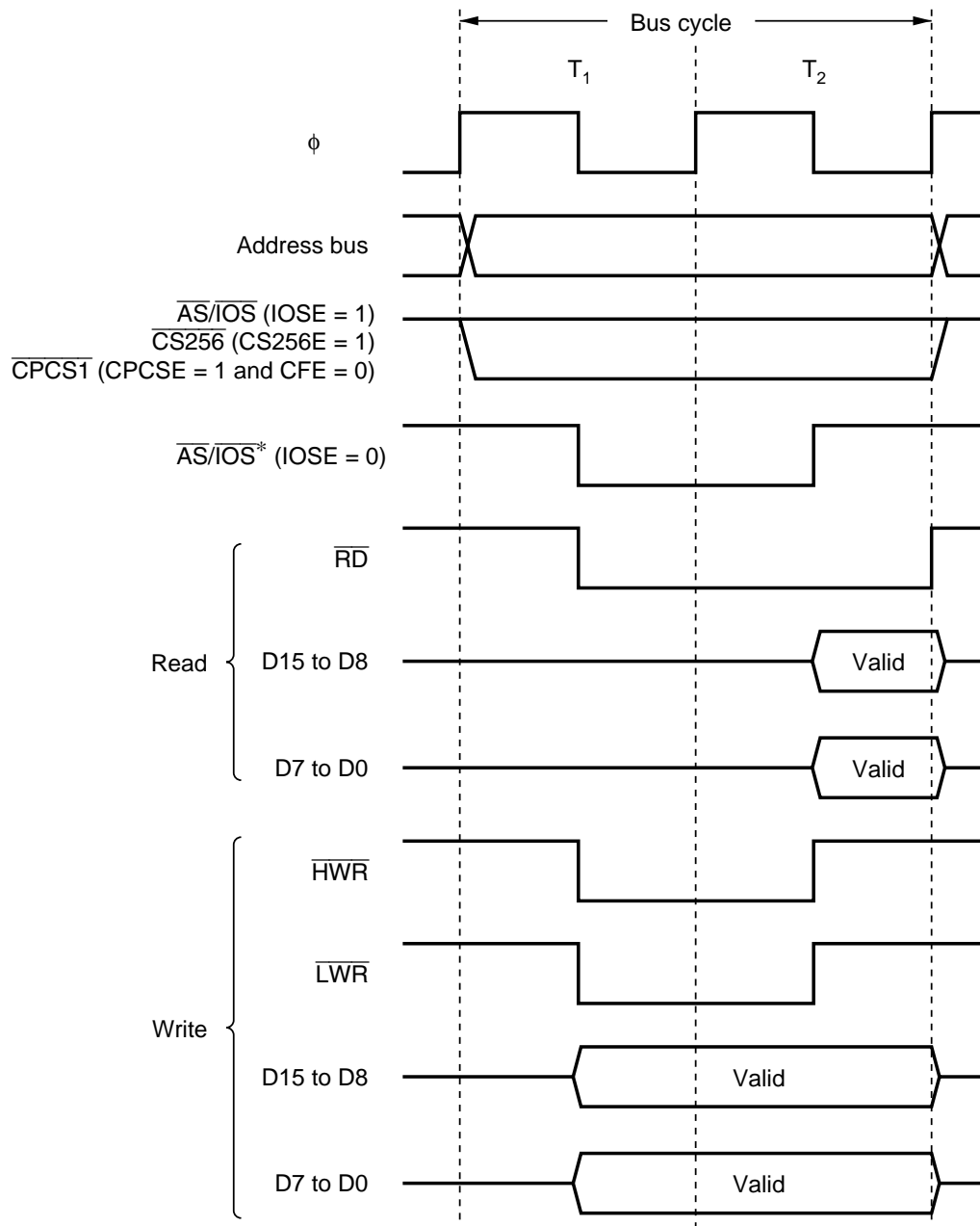
**16-Bit, 2-State Access Space:** Figures 6.7 to 6.9 show bus timings for a 16-bit, 2-state access space. When a 16-bit access space is accessed, the upper half (D15 to D8) of the data bus is used for even addresses, and the lower half (D7 to D0) for odd addresses. Wait states cannot be inserted.



**Figure 6.7 Bus Timing for 16-Bit, 2-State Access Space (Even Byte Access)**



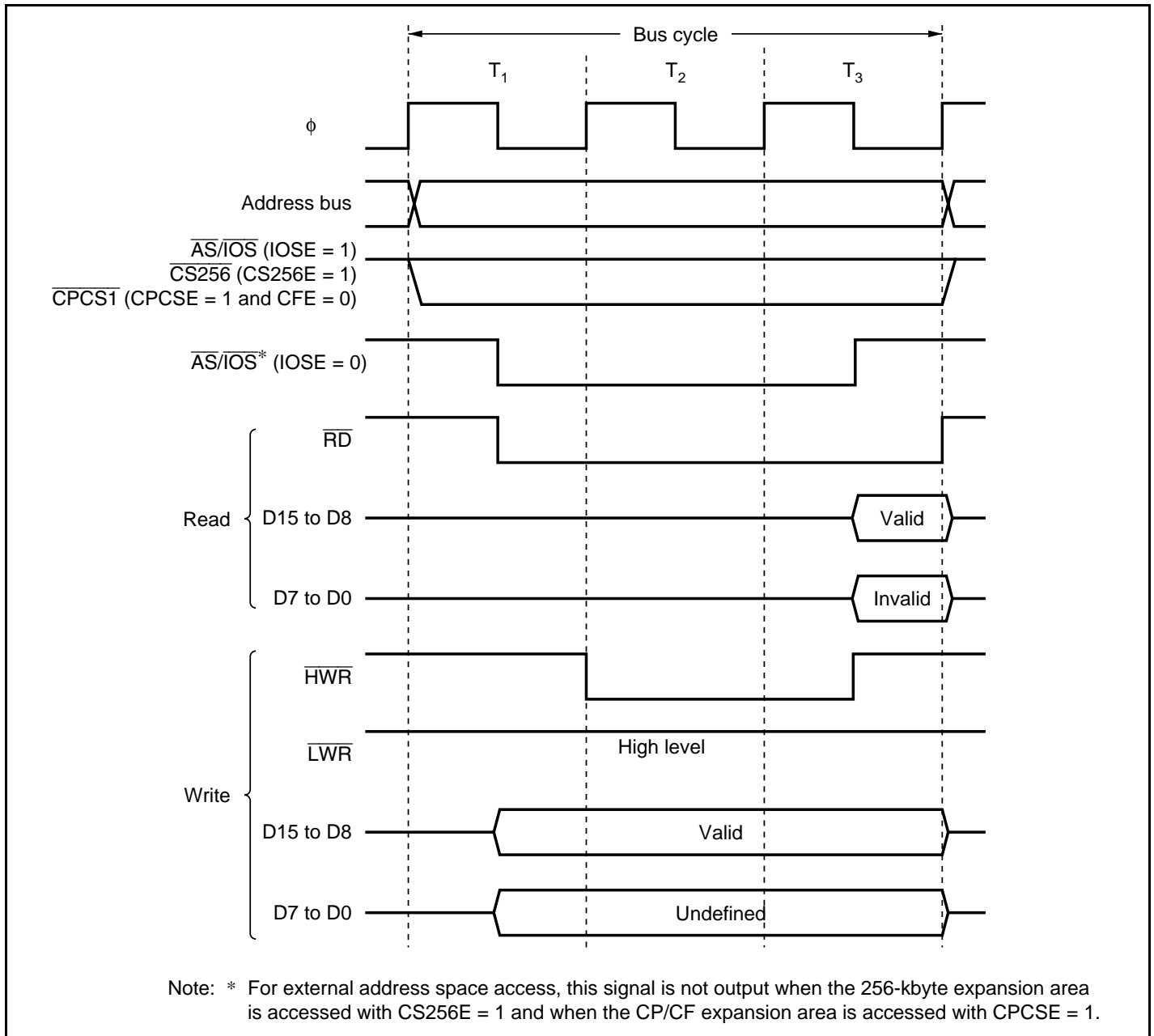
**Figure 6.8 Bus Timing for 16-Bit, 2-State Access Space (Odd Byte Access)**



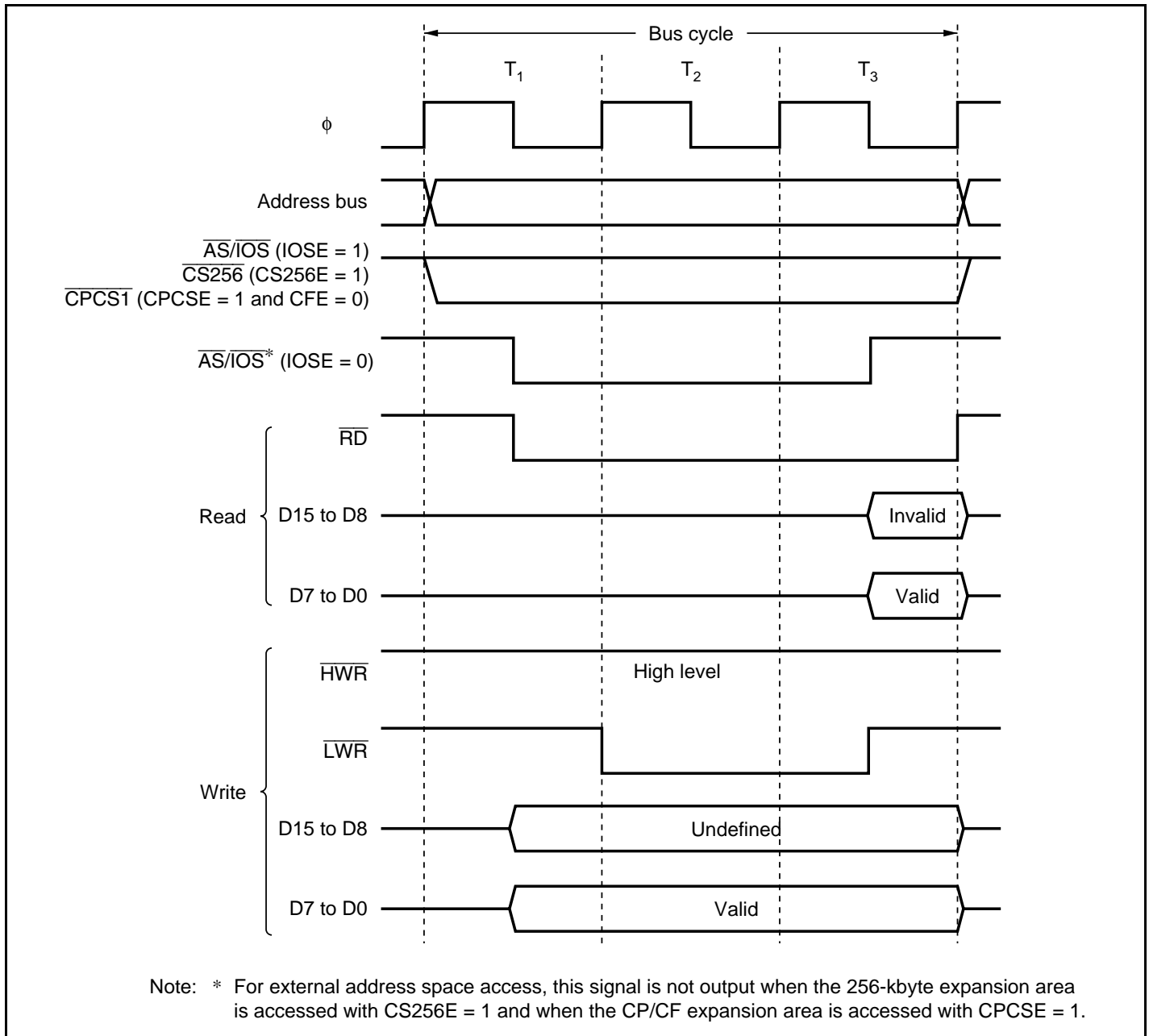
Note: \* For external address space access, this signal is not output when the 256-kbyte expansion area is accessed with CS256E = 1 and when the CP/CF expansion area is accessed with CPCSE = 1.

**Figure 6.9 Bus Timing for 16-Bit, 2-State Access Space (Word Access)**

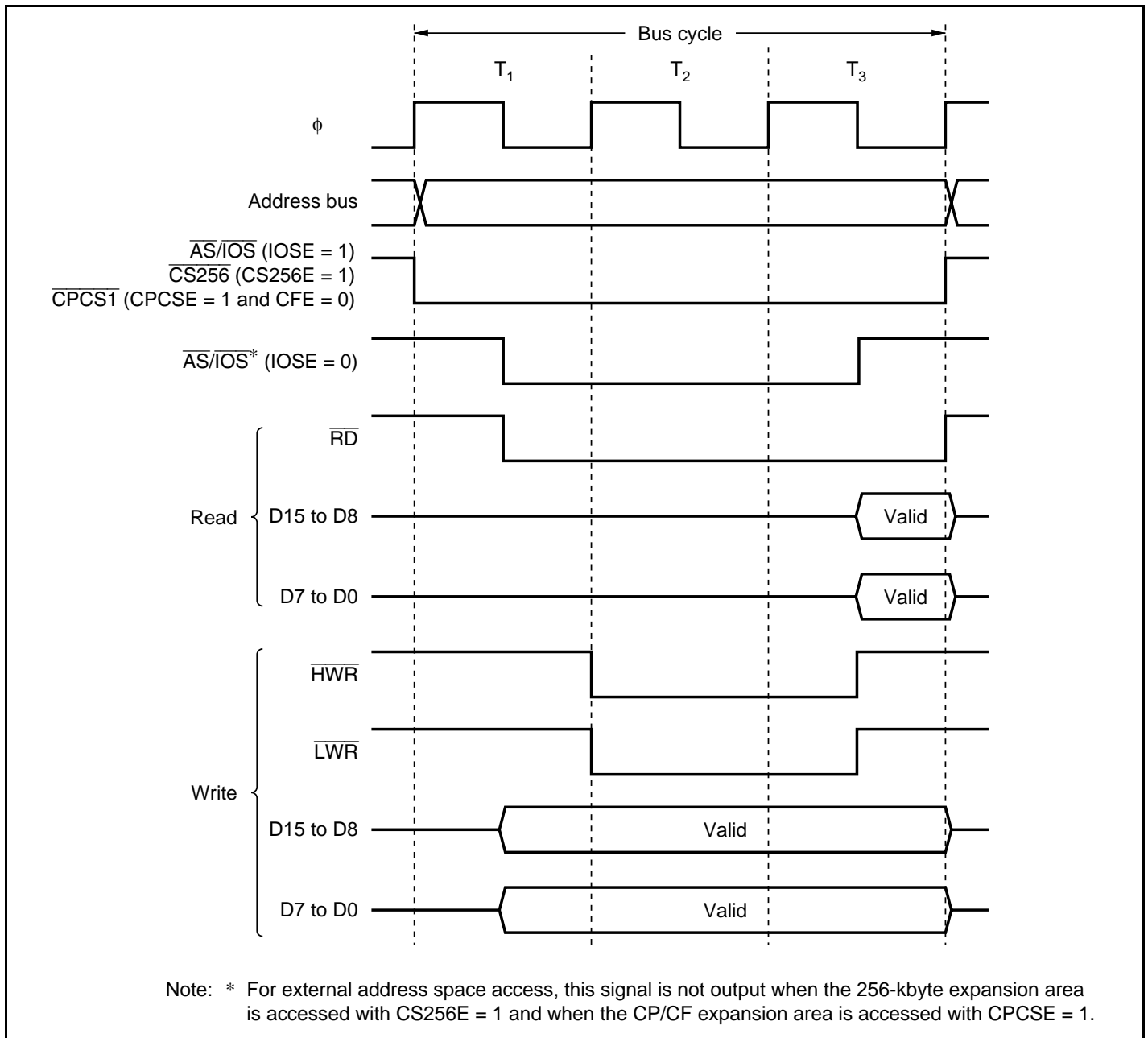
**16-Bit, 3-State Access Space:** Figures 6.10 to 6.12 show bus timings for a 16-bit, 3-state access space. When a 16-bit access space is accessed, the upper half (D15 to D8) of the data bus is used for even addresses, and the lower half (D7 to D0) for odd addresses. Wait states can be inserted.



**Figure 6.10 Bus Timing for 16-Bit, 3-State Access Space (Even Byte Access)**



**Figure 6.11 Bus Timing for 16-Bit, 3-State Access Space (Odd Byte Access)**



**Figure 6.12 Bus Timing for 16-Bit, 3-State Access Space (Word Access)**



### 6.5.4 Wait Control

When accessing the external address space, this LSI can extend the bus cycle by inserting one or more wait states ( $T_W$ ). There are three ways of inserting wait states: Program wait insertion, pin wait insertion using the  $\overline{\text{WAIT/CPWAIT}}$  pin, and the combination of program wait and the  $\overline{\text{WAIT/CPWAIT}}$  pin.

**Program Wait Mode:** A specified number of wait states  $T_W$  can be inserted automatically between the  $T_2$  state and  $T_3$  state when accessing the external address space always according to the settings of the WC1 and WC0 bits in WSCR (the WC11 and WC10 bits in WSCR2 for the 256-kbyte expansion area, and the WC21 and WC20 bits in WSCR2 for the CP expansion area).

**Pin Wait Mode:** A specified number of wait states  $T_W$  can be inserted automatically between the  $T_2$  state and  $T_3$  state when accessing the external address space always according to the settings of the WC1 and WC0 bits (the WC21 and WC20 bits for the CP expansion area). If the  $\overline{\text{WAIT/CPWAIT}}$  pin is low at the falling edge of  $\phi$  in the last  $T_2$  or  $T_W$  state, another  $T_W$  state is inserted. If the  $\overline{\text{WAIT/CPWAIT}}$  pin is held low,  $T_W$  states are inserted until it goes high.

This is useful when inserting four or more  $T_W$  states, or when changing the number of  $T_W$  states to be inserted for each external device.

**Pin Auto-Wait Mode:** A specified number of wait states  $T_W$  can be inserted automatically between the  $T_2$  state and  $T_3$  state when accessing the external address space according to the settings of the WC1 and WC0 bits (the WC21 and WC20 bits for the CP expansion area) if the  $\overline{\text{WAIT/CPWAIT}}$  pin is low at the falling edge of  $\phi$  in the last  $T_2$  state. Even if the  $\overline{\text{WAIT/CPWAIT}}$  pin is held low,  $T_W$  states can be inserted only up to the specified number of states.

This function enables the low-speed memory interface only by inputting the chip select signal to the  $\overline{\text{WAIT/CPWAIT}}$  pin.

Figure 6.13 shows an example of wait state insertion timing in pin wait mode.

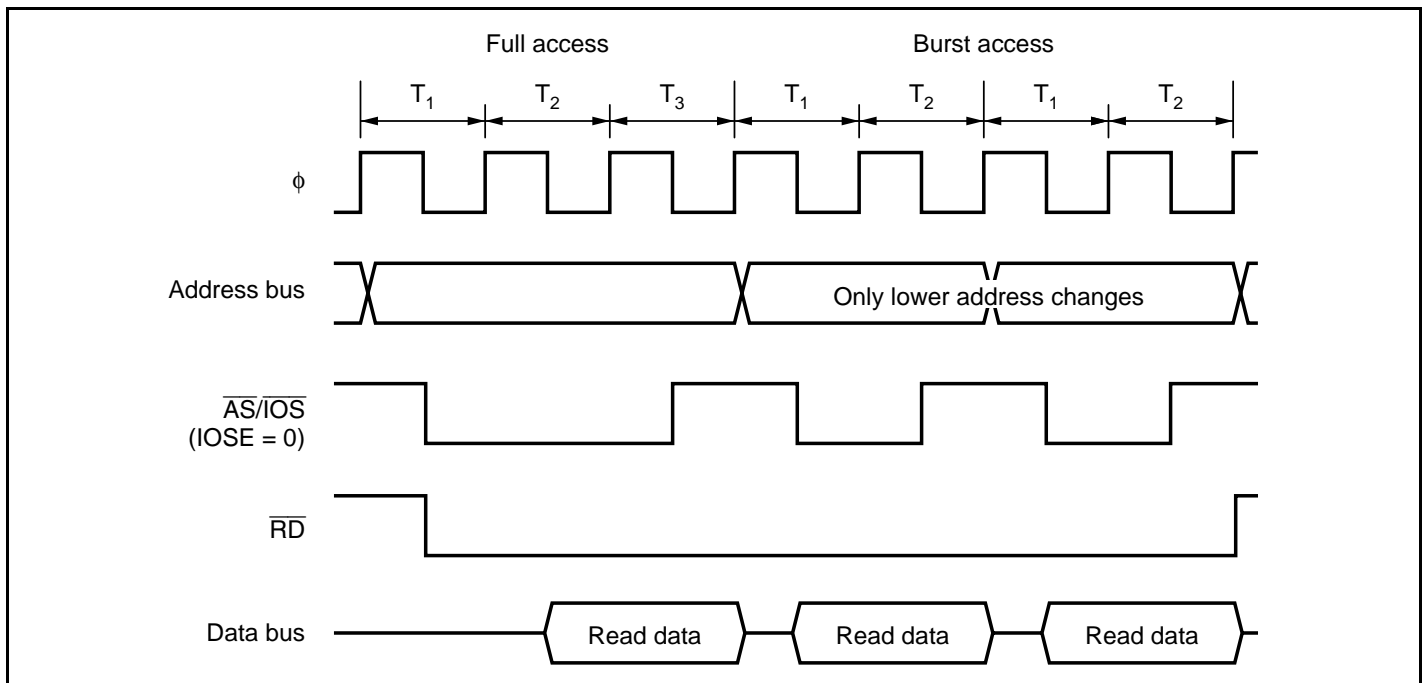
The settings after a reset are: 3-state access, 3 program wait insertion, and  $\overline{\text{WAIT/CPWAIT}}$  pin input disabled.



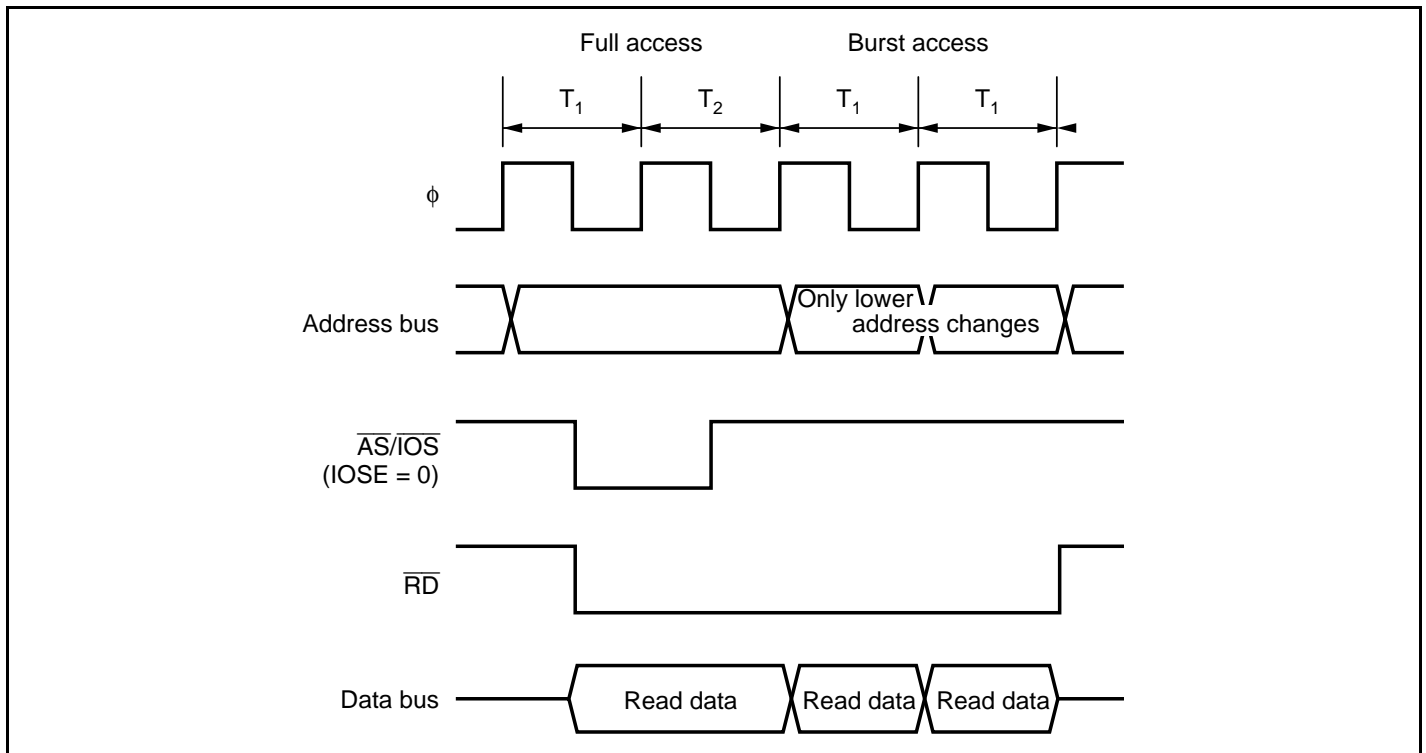
### 6.6.1 Basic Operation Timing

The number of access states in the initial cycle (full access) of the burst ROM interface is determined by the AST bit in WSCR. When the AST bit is set to 1, wait states can be inserted. 1 or 2 states can be selected for burst access according to the setting of the BRSTS1 bit in BCR. Wait states cannot be inserted in a burst cycle. Burst accesses of a maximum four words is performed when the BRSTS0 bit in BCR is cleared to 0, and burst accesses of a maximum eight words is performed when the BRSTS0 bit in BCR is set to 1.

The basic access timing for the burst ROM space is shown in figures 6.14 and 6.15.



**Figure 6.14 Access Timing Example in Burst ROM Space (AST = BRSTS1 = 1)**



**Figure 6.15 Access Timing Example in Burst ROM Space (AST = BRSTS1 = 0)**

## 6.6.2 Wait Control

As with the basic bus interface, program wait insertion or pin wait insertion using the  $\overline{WAIT}$  pin can be used in the initial cycle (full access) of the burst ROM interface. For details, see section 6.5.4, Wait Control. Wait states cannot be inserted in a burst cycle.

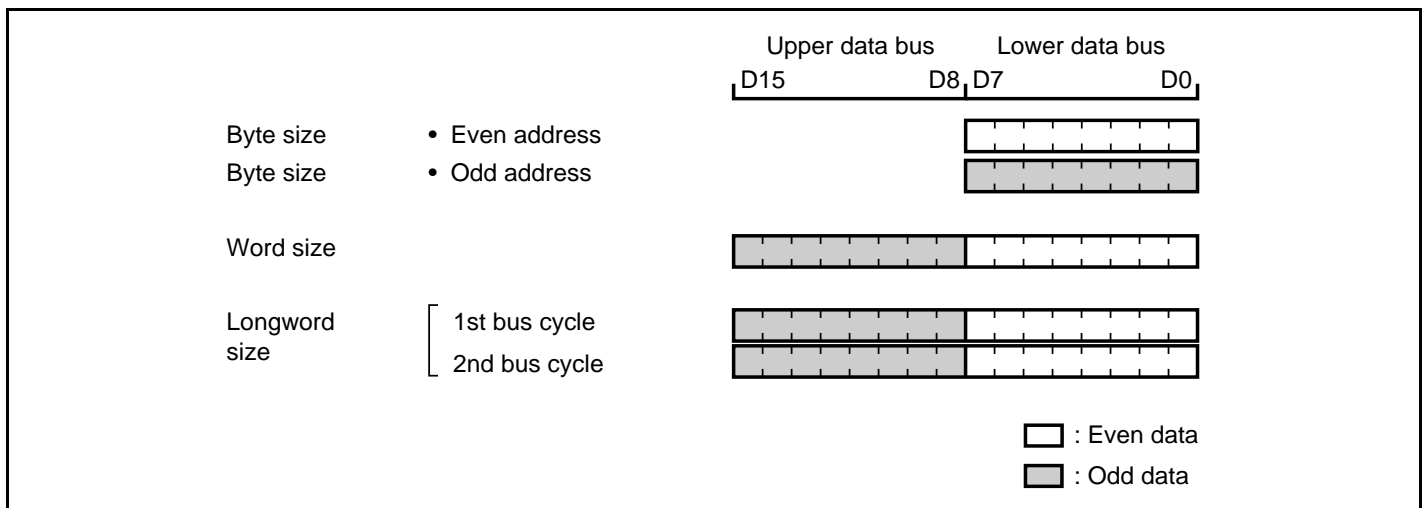
## 6.7 Memory Card Interface

A CP expansion area can be set to the CF expansion area (memory card mode) by setting both the CPCSE bit in BCR2 to 1 and the CFE bit in BCR to 1. In memory card mode, the bus width is fixed to 16 bits. In this mode, signal pins other than  $\overline{\text{CPCS1}}$  and  $\overline{\text{CPCS2}}$  are common to the basic bus interface, but their signal waveforms differ. The number of access states and waveforms of the strobe signals ( $\overline{\text{CPOE}}$  and  $\overline{\text{CPWE}}$ ) can be controlled by the WMS21, WMS20, WC22, WC21, and WC20 bits in WSCR2 and the OWEAC and OWENC bits in BCR2.

### 6.7.1 Data Size and Data Alignment

The data sizes for the CPU and other internal bus masters are byte, word, and longword. The BSC has a data alignment function, and controls whether the upper data bus (D15 to D8) or lower data bus (D7 to D0) is used when the CF expansion area is accessed in memory card mode according to the accessed data size.

Figure 6.16 illustrates the data alignment control. In CF expansion area access, the upper data bus (D15 to D8) and lower data bus (D7 to D0) are used. The amount of data that can be accessed at one time is one byte or one word: a longword access is performed as two word accesses.



**Figure 6.16 Access Sizes and Data Alignment Control**

## 6.7.2 Valid Strobes

Table 6.10 shows the data buses used and valid strobes.

**Table 6.10 Data Buses Used and Valid Strobes**

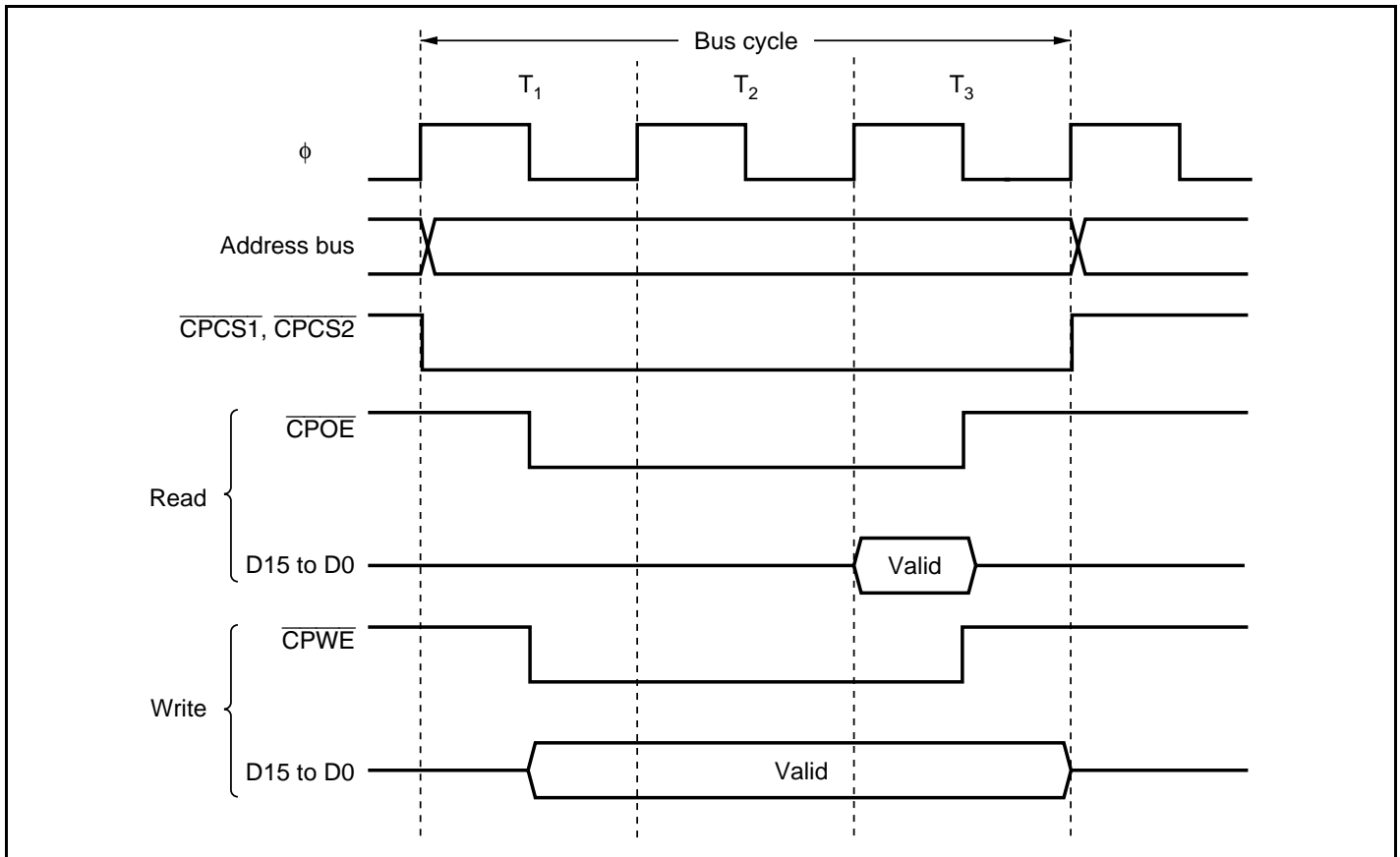
Access Size	Read/Write	Address	$\overline{\text{CPCS1}}$ Pin	$\overline{\text{CPCS2}}$ Pin	Valid Strobe	Upper Data Bus (D15 to D8)	Lower Data Bus (D7 to D0)
Byte	Read	Even	L	H	CPOE	Invalid	Valid (even data)
		Odd	L	H		Invalid	Valid (odd data)
	Write	Even	L	H	CPWE	Undefined	Valid (even data)
		Odd	L	H		Undefined	Valid (even data)
Word	Read	—	L	L	CPOE	Valid (odd data)	Valid (even data)
	Write	—	L	L	CPWE	Valid (odd data)	Valid (even data)

Notes: Undefined: Undefined data is output.

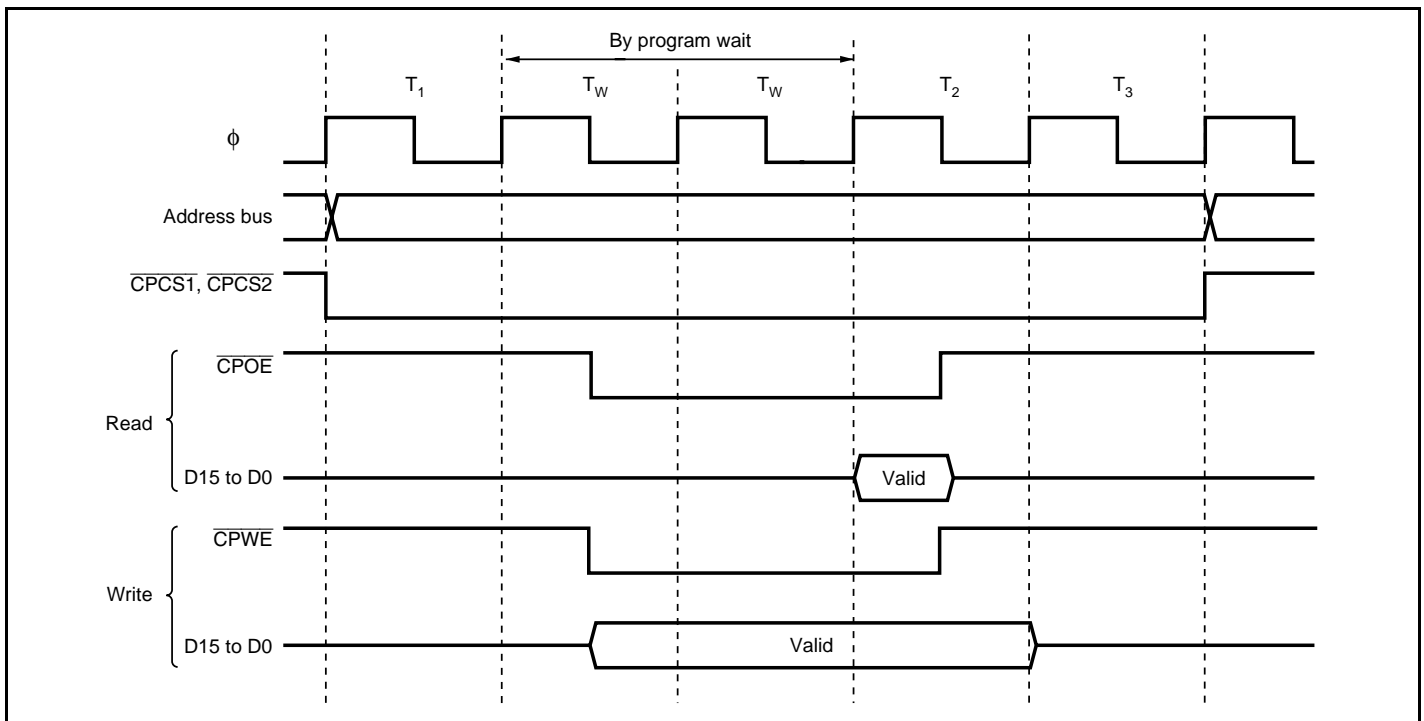
Invalid: Input state with the input value ignored.

## 6.7.3 Basic Operation Timing

The memory card interface is basically specified as having 3 access states. Figure 6.17 shows the access timing in memory card mode. The strobe signal waveform for the rising edge and falling edge at address output can be moved one state by setting the OWEAC and OWENC bits in BCR, respectively. In this case, wait states must be inserted. For 2-state access, clear both the OWEAC and OWENC bits to 0. In addition, note that in 3-state access, set the OWEAC and OWENC bits to B'01 or B'10. Figure 6.18 shows the access timing in memory card mode when the OWEAC and OWENC bits are set to 1 simultaneously.



**Figure 6.17 Access Timing in Memory Card Mode (Basic Cycle)**



**Figure 6.18 Access Timing in Memory Card Mode ( $OWEAC = OWENC = 1$  with Wait State Insertion)**

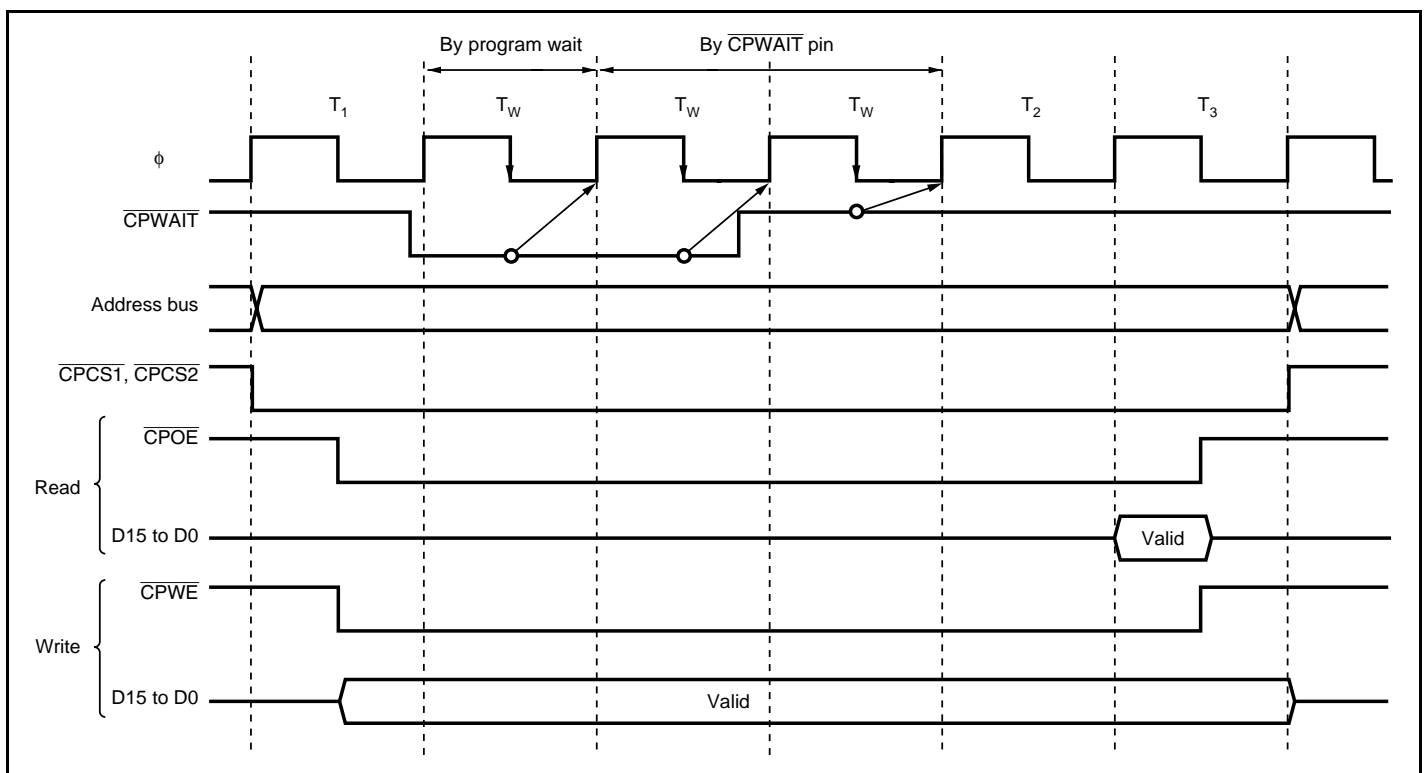
### 6.7.4 Wait Control

With memory card interface, there are two ways of inserting wait states: Program wait insertion and pin wait insertion using the  $\overline{\text{CPWAIT}}$  pin.

**Program Wait Mode:** A specified number of wait states  $T_W$  can be inserted between the  $T_2$  state and  $T_3$  state when accessing the CF expansion area according to the settings in the WC22, WC21, and WC20 bits in WSCR2.

**Pin Wait Mode:** A specified number of wait states  $T_W$  can be inserted between the  $T_2$  state and  $T_3$  state when accessing the CF expansion area according to the settings in the WC22, WC21, and WC20 bits. If the  $\overline{\text{CPWAIT}}$  pin is low at the falling edge of  $\phi$  in the last  $T_2$  or  $T_W$  state, another  $T_W$  state is inserted. If the  $\overline{\text{CPWAIT}}$  pin is held low,  $T_W$  states are inserted until it goes high. Note, however, that  $T_W$  state insertion by the  $\overline{\text{CPWAIT}}$  pin is not performed unless program wait mode is specified.

Figure 6.19 shows an example of wait state insertion timing.



**Figure 6.19 Access Timing Example in Memory Card Mode (Wait State Insertion by Program Wait and  $\overline{\text{CPWAIT}}$  Pin)**

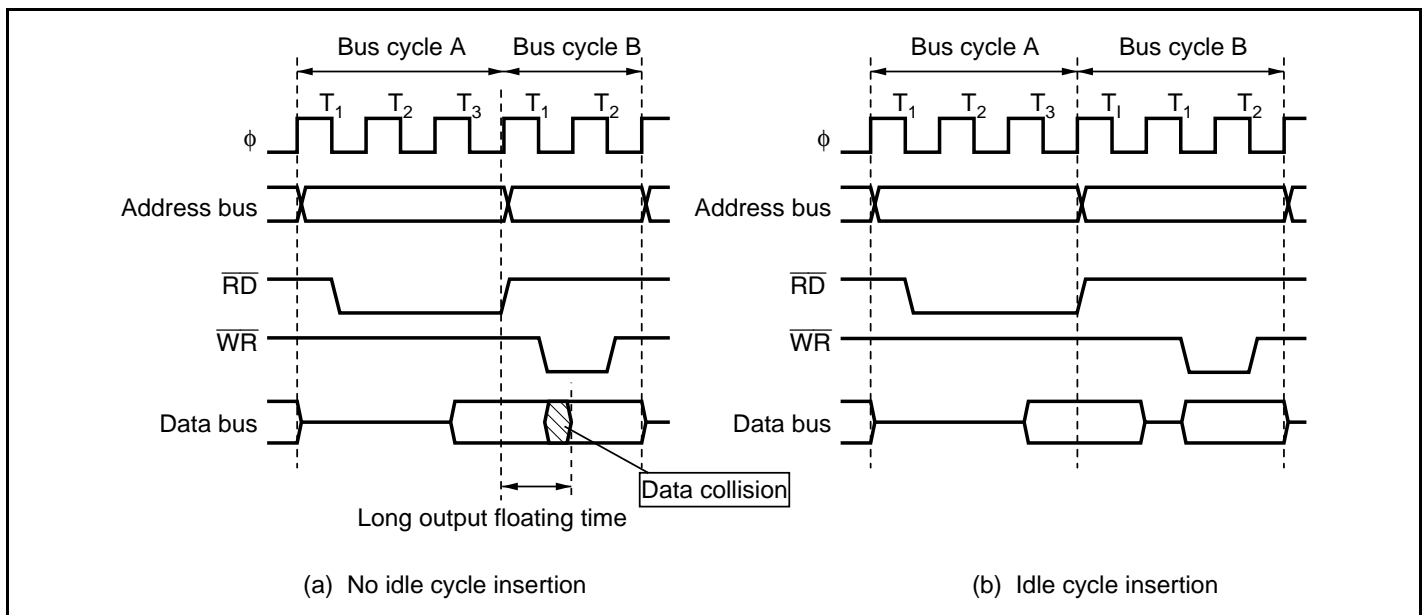


## 6.8 Idle Cycle

When this LSI accesses the external address space, it can insert a 1-state idle cycle ( $T_1$ ) between bus cycles when a write cycle occurs immediately after a read cycle. By inserting an idle cycle it is possible, for example, to avoid data collisions between ROM with a long output floating time, and high-speed memory and I/O interfaces.

If an external write occurs after an external read while the ICIS bit is set to 1 in BCR, an idle cycle is inserted at the start of the write cycle.

Figure 6.20 shows examples of idle cycle operation. In these examples, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a CPU write cycle. In figure 6.20 (a), with no idle cycle inserted, a collision occurs in bus cycle B between the read data from ROM and the CPU write data. In figure 6.20 (b), an idle cycle is inserted, thus preventing data collision.



**Figure 6.20 Examples of Idle Cycle Operation**

Table 6.11 shows the pin states in an idle cycle.

**Table 6.11 Pin States in Idle Cycle**

<b>Pins</b>	<b>Pin State</b>
A17 to A0	Contents of immediately following bus cycle
D15 to D0	High impedance
$\overline{AS}$ , $\overline{IOS}$ , $\overline{CS256}$ , $\overline{CPCS1}$ , $\overline{CPCS2}$	High
$\overline{RD}$ , $\overline{CPOE}$	High
$\overline{HWR}$ , $\overline{LWR}$ , $\overline{CPWE}$	High

## 6.9 Bus Arbitration

The BSC has a bus arbiter that arbitrates bus master operations. There are three bus masters—the CPU, DTC, and RFU—that perform read/write operations when they have possession of the bus.

### 6.9.1 Bus Master Priority

Each bus master requests the bus by means of a bus request signal. The bus arbiter detects the bus masters' bus request signals, and if a bus request occurs, it sends a bus request acknowledge signal to the bus master making the request at the designated timing. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled. The order of priority of the bus masters is as follows:

(High) RFU > DTC > CPU (Low)

## 6.9.2 Bus Transfer Timing

When a bus request is received from a bus master with a higher priority than that of the bus master that has acquired the bus and is currently operating, the bus is not necessarily transferred immediately. Each bus master can relinquish the bus at the timings given below.

**CPU:** The CPU is the lowest-priority bus master, and if a bus request is received from the DTC or RFU, the bus arbiter transfers the bus to the DTC.

- DTC bus transfer timing
  - The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the component operations. For details, refer to section 2.7, Bus States During Instruction Execution in the H8S/2600 Series, H8S/2000 Series Programming Manual.
  - If the CPU is in sleep mode, the bus is transferred immediately.
- RFU bus transfer timing
  - The bus is transferred at a break between bus cycles. Even in discrete operations, as in the case of a longword-size access, the bus can be transferred between the component operations. For details, refer to section 8, RAM-FIFO Unit (RFU).
  - If the CPU is in sleep mode, the bus is transferred immediately.

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

Since the bus master priority of the DTC is lower than the RFU, the bus arbiter transfers the bus mastership from the DTC to the RFU if the RFU requests the bus.

- RFU bus transfer timing

The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the component operations. In addition, in 32-bit access by the DTC, the bus is not transferred at a break between longword access cycles. For details, refer to section 8, RAM-FIFO Unit (RFU).

**RFU:** The RFU has the highest bus master priority. The RFU sends the bus arbiter a request for the bus when an activation request is generated. The RFU does not release the bus until it completes its operation. For details, refer to section 8, RAM-FIFO Unit (RFU).



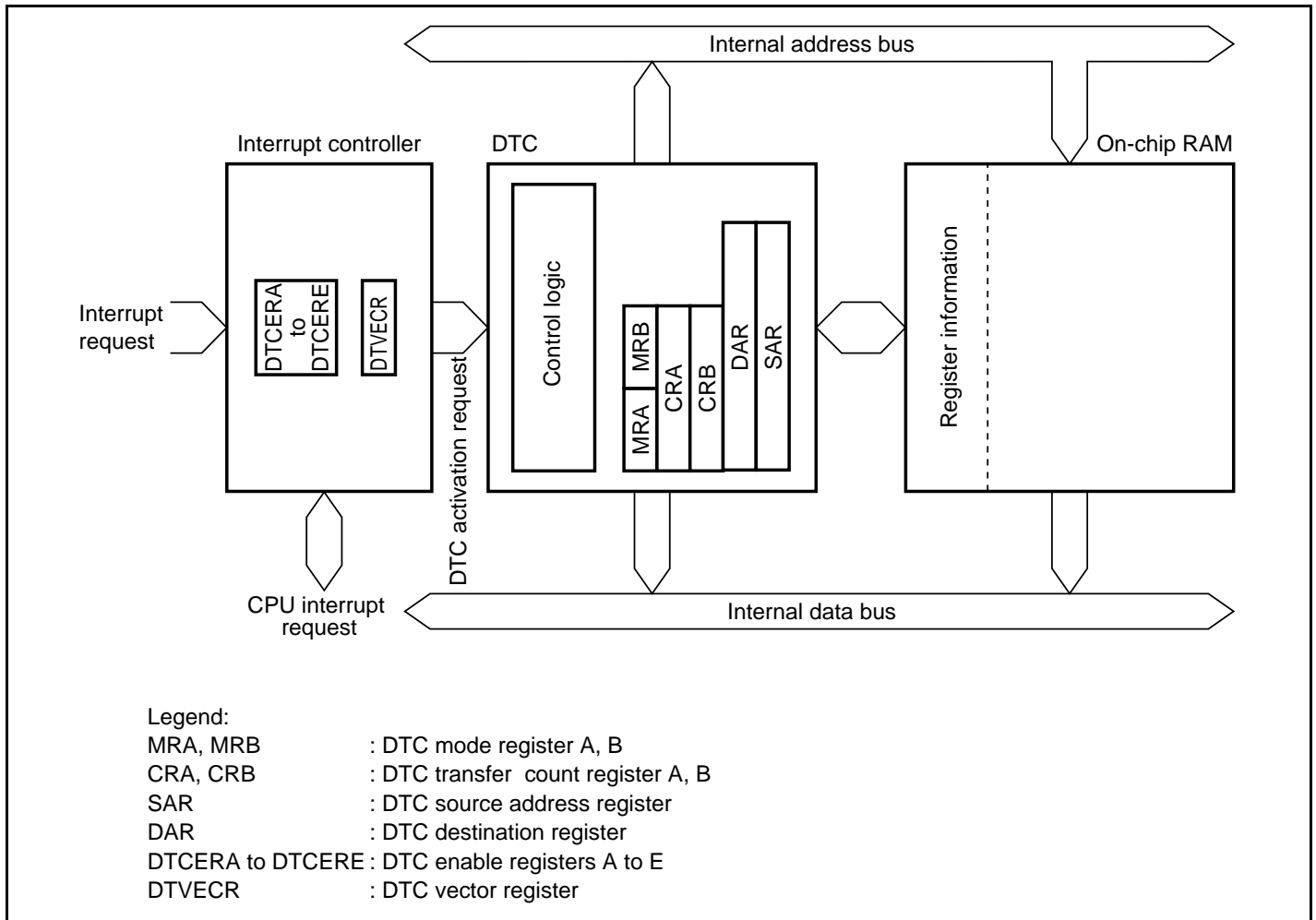
## Section 7 Data Transfer Controller (DTC)

This LSI includes a data transfer controller (DTC). The DTC can be activated by an interrupt or software, to transfer data.

Figure 7.1 shows a block diagram of the DTC. The DTC's register information is stored in the on-chip RAM. When the DTC is used, the RAME bit in SYSCR must be set to 1. A 32-bit bus connects the DTC to addresses H'(FF)EC00 to H'(FF)EFFF in on-chip RAM (1 kbyte), enabling 32-bit/1-state reading and writing of the DTC register information.

### 7.1 Features

- Transfer is possible over any number of channels
- Three transfer modes:  
Normal, repeat, and block transfer modes are available
- One activation source can trigger a number of data transfers (chain transfer)
- Direct specification of 16-Mbyte address space is possible
- Activation by software is possible
- Transfer can be set in byte or word units
- A CPU interrupt can be requested for the interrupt that activated the DTC
- Module stop mode can be set
- Usable for scan operations of CIN7 to CIN0
- DTC operates in high-speed mode even when the LSI is in medium-speed mode



**Figure 7.1 Block Diagram of DTC**

## 7.2 Register Descriptions

The DTC has the following registers.

- DTC mode register A (MRA)
- DTC mode register B (MRB)
- DTC source address register (SAR)
- DTC destination address register (DAR)
- DTC transfer count register A (CRA)
- DTC transfer count register B (CRB)

These six registers cannot be directly accessed from the CPU. When a DTC activation interrupt source occurs, the DTC reads a set of register information that is stored in on-chip RAM to the corresponding DTC registers and transfers data. After the data transfer, it writes a set of updated register information back to on-chip RAM.

- DTC enable registers (DTCER)
- DTC vector register (DTVECR)

### 7.2.1 DTC Mode Register A (MRA)

MRA selects the DTC operating mode.

Bit	Bit Name	Initial Value	R/W	Description
7	SM1	Undefined	—	Source Address Mode 1, 0
6	SM0	Undefined	—	These bits specify an SAR operation after a data transfer. 0X: SAR is fixed 10: SAR is incremented after a transfer (by +1 when Sz = 0, by +2 when Sz = 1) 11: SAR is decremented after a transfer (by –1 when Sz = 0, by –2 when Sz = 1)
5	DM1	Undefined	—	Destination Address Mode 1, 0
4	DM0	Undefined	—	These bits specify a DAR operation after a data transfer. 0X: DAR is fixed 10: DAR is incremented after a transfer (by +1 when Sz = 0, by +2 when Sz = 1) 11: DAR is decremented after a transfer (by –1 when Sz = 0, by –2 when Sz = 1)
3	MD1	Undefined	—	DTC Mode
2	MD0	Undefined	—	These bits specify the DTC transfer mode. 00: Normal mode 01: Repeat mode 10: Block transfer mode 11: Setting prohibited
1	DTS	Undefined	—	DTC Transfer Mode Select Specifies whether the source side or the destination side is set to be a repeat area or block area in repeat mode or block transfer mode. 0: Destination side is repeat area or block area 1: Source side is repeat area or block area

Bit	Bit Name	Initial Value	R/W	Description
0	Sz	Undefined	—	DTC Data Transfer Size Specifies the size of data to be transferred. 0: Byte-size transfer 1: Word-size transfer

Legend:

X: Don't care

### 7.2.2 DTC Mode Register B (MRB)

MRB selects the DTC operating mode.

Bit	Bit Name	Initial Value	R/W	Description
7	CHNE	Undefined	—	DTC Chain Transfer Enable When this bit is set to 1, a chain transfer will be performed. For details, see section 7.5.4, Chain Transfer. In data transfer with CHNE set to 1, determination of the end of the specified number of data transfers, clearing of the interrupt source flag, and clearing of DTCER are not performed.
6	DISEL	Undefined	—	DTC Interrupt Select When this bit is set to 1, a CPU interrupt request is generated every time data transfer ends. When this bit is cleared to 0, a CPU interrupt request is generated only when the specified number of data transfer ends. Note however that when the DTC is activated by a USB or MCIF interrupt source and this bit is cleared to 0, this LSI does not operate correctly. In such a case, be sure to set this bit to 1.
5 to 0	—	Undefined	—	Reserved These bits have no effect on DTC operation. The write value should always be 0.



### 7.2.3 DTC Source Address Register (SAR)

SAR is a 24-bit register that designates the source address of data to be transferred by the DTC. For word-size transfer, specify an even source address.

### 7.2.4 DTC Destination Address Register (DAR)

DAR is a 24-bit register that designates the destination address of data to be transferred by the DTC. For word-size transfer, specify an even destination address.

### 7.2.5 DTC Transfer Count Register A (CRA)

CRA is a 16-bit register that designates the number of times data is to be transferred by the DTC.

In normal mode, the entire CRA functions as a 16-bit transfer counter (1 to 65536). It is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

In repeat mode or block transfer mode, the CRA is divided into two parts; the upper 8 bits (CRAH) and the lower 8 bits (CRAL). CRAH holds the number of transfers while CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are sent when the count reaches H'00.

### 7.2.6 DTC Transfer Count Register B (CRB)

CRB is a 16-bit register that designates the number of times data is to be transferred by the DTC in block transfer mode. It functions as a 16-bit transfer counter (1 to 65536) that is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000.

### 7.2.7 DTC Enable Registers (DTCER)

DTCER specifies DTC activation interrupt sources. DTCER is comprised of five registers: DTCERA to DTCERE. The correspondence between interrupt sources and DTCE bits is shown in tables 7.1 to 7.3. For DTCE bit setting, use bit manipulation instructions such as BSET and BCLR. Multiple DTC activation sources can be set at one time (only at the initial setting) by masking all interrupts and writing data after executing a dummy read on the relevant register.

Bit	Bit Name	Initial Value	R/W	Description
7	DTCE7	0	R/W	DTC Activation Enable
6	DTCE6	0	R/W	Setting this bit to 1 specifies a relevant interrupt source as a DTC activation source.
5	DTCE5	0	R/W	
4	DTCE4	0	R/W	[Clearing conditions]
3	DTCE3	0	R/W	
2	DTCE2	0	R/W	<ul style="list-style-type: none"> <li>When data transfer has ended with the DISEL bit in MRB set to 1</li> </ul>
1	DTCE1	0	R/W	
0	DTCE0	0	R/W	<ul style="list-style-type: none"> <li>When the specified number of transfers have ended</li> </ul>

These bits are not cleared when the DISEL bit is 0 and the specified number of transfers have not been completed

**Table 7.1 Correspondence between Interrupt Sources and DTCER**

Bit	Bit Name	Register				
		DTCERA	DTCERB	DTCERC	DTCERD	DTCERE
7	DTCE <sub>n</sub> 7	(16)IRQ0	(53)OCIB	(69)CMIB1	(86)TXI1	(108)USBI0
6	DTCE <sub>n</sub> 6	(17)IRQ1	(93)IICM0	(72)CMIAY	(89)RXI2	(109)USBI1
5	DTCE <sub>n</sub> 5	(18)IRQ2	(94)IICR0	(73)CMIBY	(90)TXI2	(110)USBI2
4	DTCE <sub>n</sub> 4	(19)IRQ3	(95)IICT0	—	(97)IICM1	(111)USBI3
3	DTCE <sub>n</sub> 3	(28)ADI	—	(44)CMIAX	(98)IICR1	—
2	DTCE <sub>n</sub> 2	(48)ICIA	(64)CMIA0	(81)RXI0	(99)IICT1	—
1	DTCE <sub>n</sub> 1	(49)ICIB	(65)CMIB0	(82)TXI0	(112)MMCIA	—
0	DTCE <sub>n</sub> 0	(52)OCIA	(68)CMIA1	(85)RXI1	(45)CMIBX	—

Notes: n: A to E

( ): Vector number

—: Reserved. The write value should always be 0.

### 7.2.8 DTC Vector Register (DTVECR)

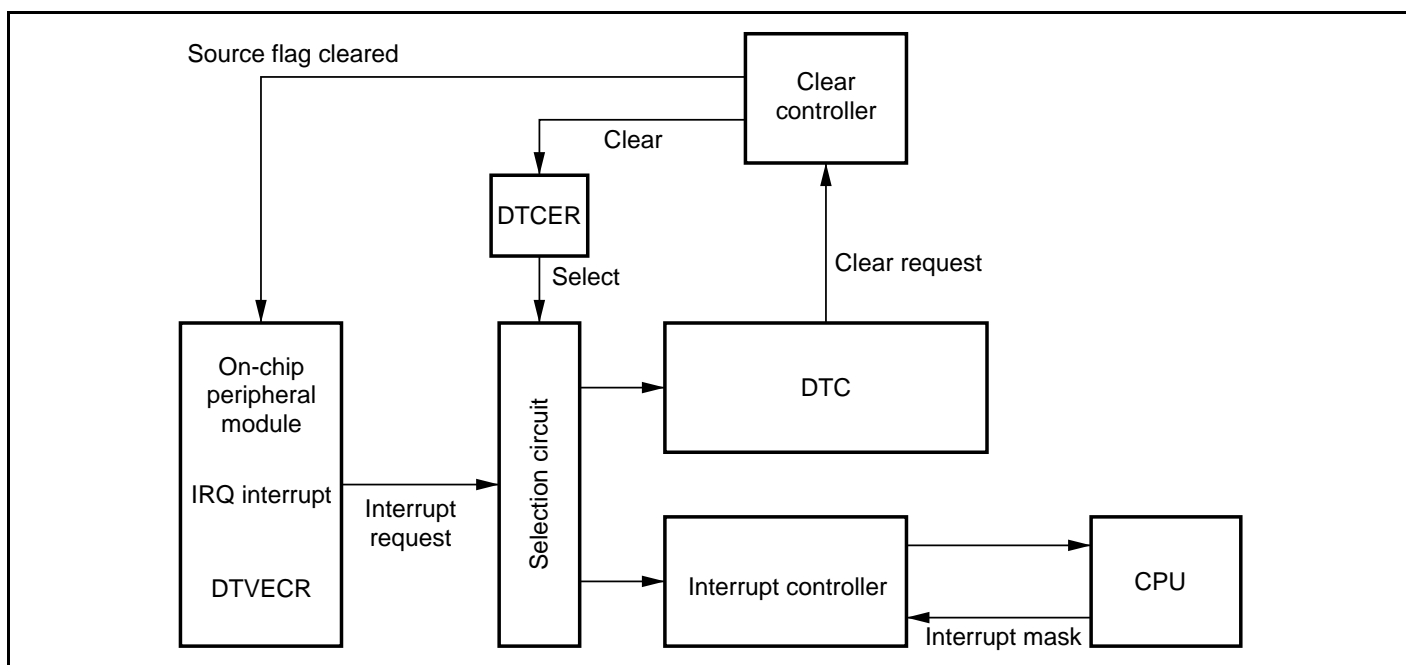
DTVECR enables or disables DTC activation by software, and sets a vector number for the software activation interrupt.

Bit	Bit Name	Initial Value	R/W	Description
7	SWDTE	0	R/W	<p>DTC Software Activation Enable</p> <p>Setting this bit to 1 activates DTC. Only 1 can be written to this bit.</p> <p>[Clearing conditions]</p> <ol style="list-style-type: none"> <li>1. When the DISEL bit is 0 and the specified number of transfers have not ended</li> <li>2. When 0 is written to the DISEL bit after a software-activated data transfer end interrupt (SWDTEND) request has been sent to the CPU.</li> </ol> <p>This bit will not be cleared when the DISEL bit is 1 and data transfer has ended or when the specified number of transfers have ended.</p>
6	DTVEC6	0	R/W	DTC Software Activation Vectors 6 to 0
5	DTVEC5	0	R/W	These bits specify a vector number for DTC software activation.
4	DTVEC4	0	R/W	
3	DTVEC3	0	R/W	
2	DTVEC2	0	R/W	
1	DTVEC1	0	R/W	
0	DTVEC0	0	R/W	

### 7.3 Activation Sources

The DTC is activated by an interrupt request or by a write to DTVECR by software. The interrupt request source to activate the DTC is selected by DTCER. At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the interrupt flag that became the activation source or the corresponding DTCER bit is cleared. The activation source flag, in the case of RXI0, for example, is the RDRF flag in SCI\_0.

When an interrupt has been designated as a DTC activation source, the existing CPU mask level and interrupt controller priorities have no effect. If there is more than one activation source at the same time, the DTC operates in accordance with the default priorities. Figure 7.2 shows a block diagram of DTC activation source control. For details on the interrupt controller, see section 5, Interrupt Controller.



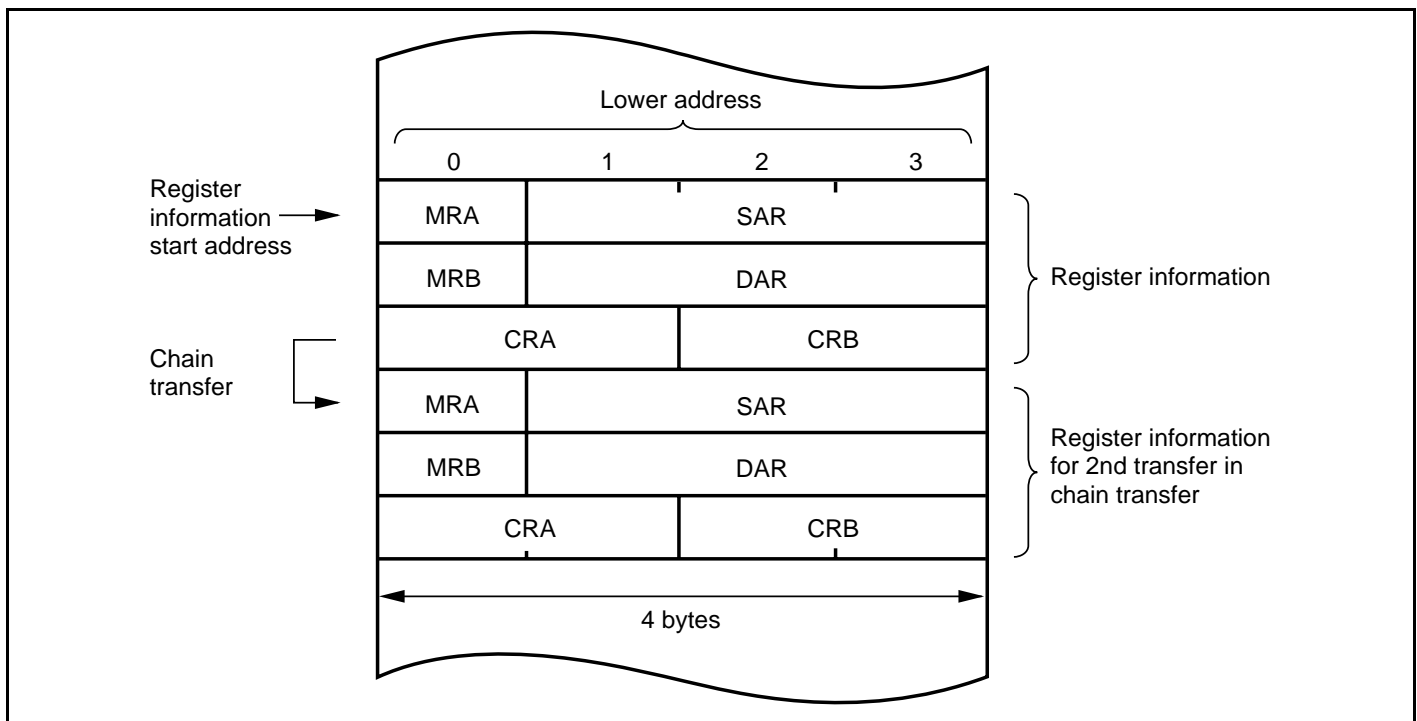
**Figure 7.2 Block Diagram of DTC Activation Source Control**

## 7.4 Location of Register Information and DTC Vector Table

Locate the register information in the on-chip RAM (addresses: H'(FF)EC00 to H'(FF)EFFF). Register information should be located at an address that is a multiple of four within the range. The method for locating the register information in address space is shown in figure 7.3. Locate MRA, SAR, MRB, DAR, CRA, and CRB, in that order, from the start address of the register information. In the case of chain transfer, register information should be located in consecutive areas as shown in figure 7.3, and the register information start address should be located at the vector address corresponding to the interrupt source in the DTC vector table. The DTC reads the start address of the register information from the vector table set for each activation source, and then reads the register information from that start address.

When the DTC is activated by software, the vector address is obtained from:  $H'0400 + (DTVECR[6:0] \times 2)$ . For example, if DTVECR is H'10, the vector address is H'0420.

The configuration of the vector address is the same in both normal and advanced modes; a 2-byte unit is used in both cases. Specify the lower two bits of the register information start address.



**Figure 7.3 DTC Register Information Location in Address Space**

**Table 7.2 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs**

Activation Source Origin	Activation Source	Vector Number	DTC Vector Address	DTCE*	Priority
Software	Write to DTVECR	DTVECR	H'0400 + (vector number × 2)	—	High
External pins	IRQ0	16	H'0420	DTCEA7	↑
	IRQ1	17	H'0422	DTCEA6	
	IRQ2	18	H'0424	DTCEA5	
	IRQ3	19	H'0426	DTCEA4	
A/D converter	ADI	28	H'0438	DTCEA3	
TMR_X	CMIAx	44	H'0458	DTCEC3	
	CMIBx	45	H'045A	DTCED0	
FRT	ICIA	48	H'0460	DTCEA2	
	ICIB	49	H'0462	DTCEA1	
	OCIA	52	H'0468	DTCEA0	
	OCIB	53	H'046A	DTCEB7	
TMR_0	CMIA0	64	H'0480	DTCEB2	
	CMIB0	65	H'0482	DTCEB1	
TMR_1	CMIA1	68	H'0488	DTCEB0	
	CMIB1	69	H'048A	DTCEC7	
TMR_Y	CMIAy	72	H'0490	DTCEC6	
	CMIBy	73	H'0492	DTCEC5	
—	Reserved for system use	76	H'0498	DTCED2	
	Reserved for system use	78	H'049C	DTCED2	
	Reserved for system use	79	H'049E	DTCEB4	
SCI_0	RXI0	81	H'04A2	DTCEC2	
	TXI0	82	H'04A4	DTCEC1	
SCI_1	RXI1	85	H'04AA	DTCEC0	
	TXI1	86	H'04AC	DTCED7	
SCI_2	RXI2	89	H'04B2	DTCED6	
	TXI2	90	H'04B4	DTCED5	
IIC_0	IICM0	93	H'04BA	DTCEB6	
	IICR0	94	H'04BC	DTCEB5	
	IICT0	95	H'04BE	DTCEB4	Low

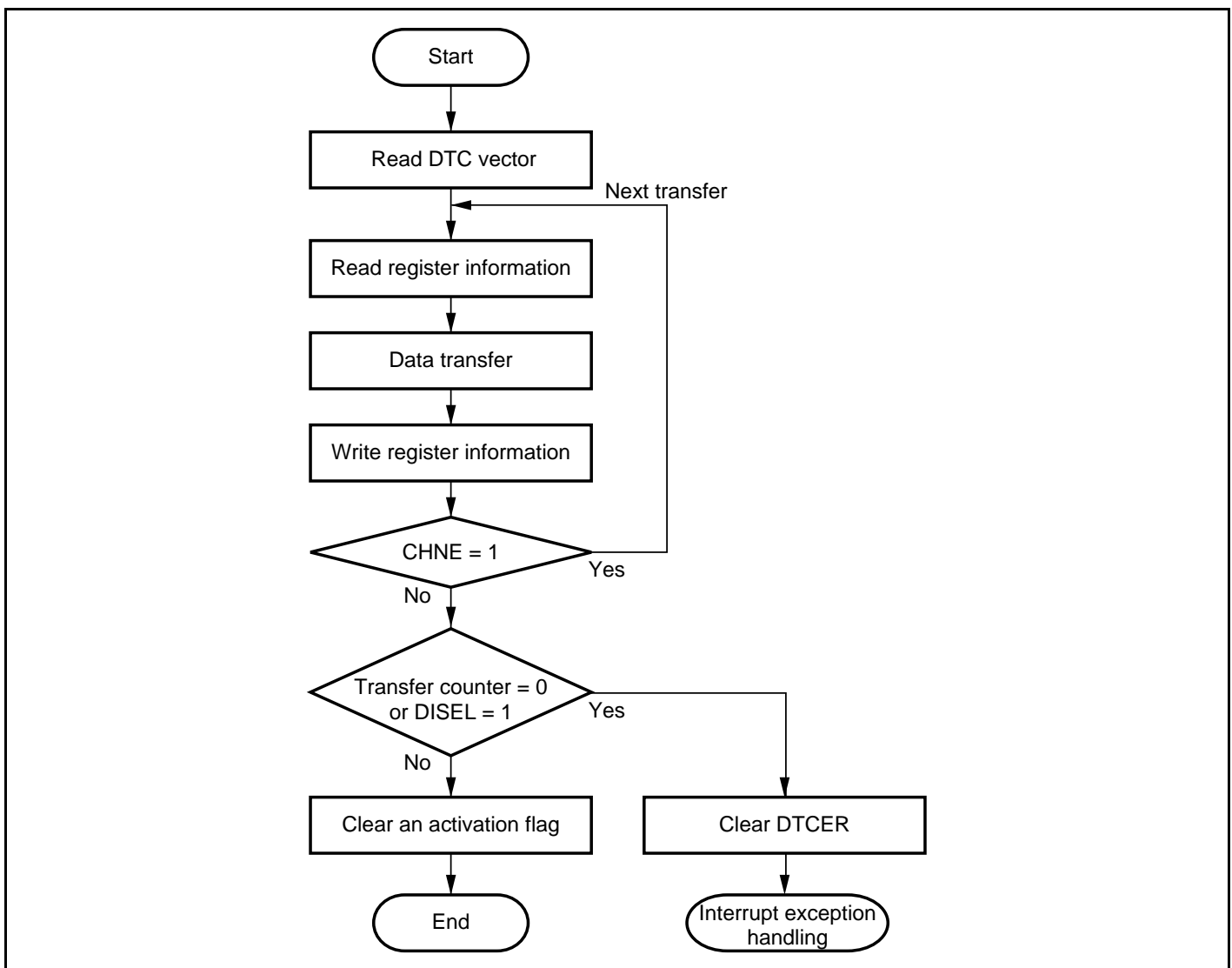
Activation Source Origin	Activation Source	Vector Number	DTC Vector Address	DTCE*	Priority
IIC_1	IICM1	97	H'04C2	DTCED4	High ↑
	IICR1	98	H'04C4	DTCED3	
	IICT1	99	H'04C6	DTCED2	
—	Reserved for system use	104	H'04D0	DTCEE3	
	Reserved for system use	105	H'04D2	DTCEE2	
	Reserved for system use	106	H'04D4	DTCEE1	
	Reserved for system use	107	H'04D6	DTCEE0	
USB	USBI0	108	H'04D8	DTCEE7	
	USBI1	109	H'04DA	DTCEE6	
	USBI2	110	H'04DC	DTCEE5	
	USBI3	111	H'04DE	DTCEE4	
MCIF	MMCIA	112	H'04E0	DTCED1	Low

Note: \* DTCE bits with no corresponding interrupt are reserved, and the write value should always be 0.

## 7.5 Operation

The DTC stores register information in on-chip RAM. When activated, the DTC reads register information in on-chip RAM and transfers data. After the data transfer, the DTC writes updated register information back to on-chip RAM. The pre-storage of register information in memory makes it possible to transfer data over any required number of channels. The transfer mode can be specified as normal, repeat, or block transfer mode. Setting the CHNE bit in MRB to 1 makes it possible to perform a number of transfers with a single activation source (chain transfer).

The 24-bit SAR designates the DTC transfer source address, and the 24-bit DAR designates the transfer destination address. After each transfer, SAR and DAR are independently incremented, decremented, or left fixed depending on its register information.



**Figure 7.4 DTC Operation Flowchart**

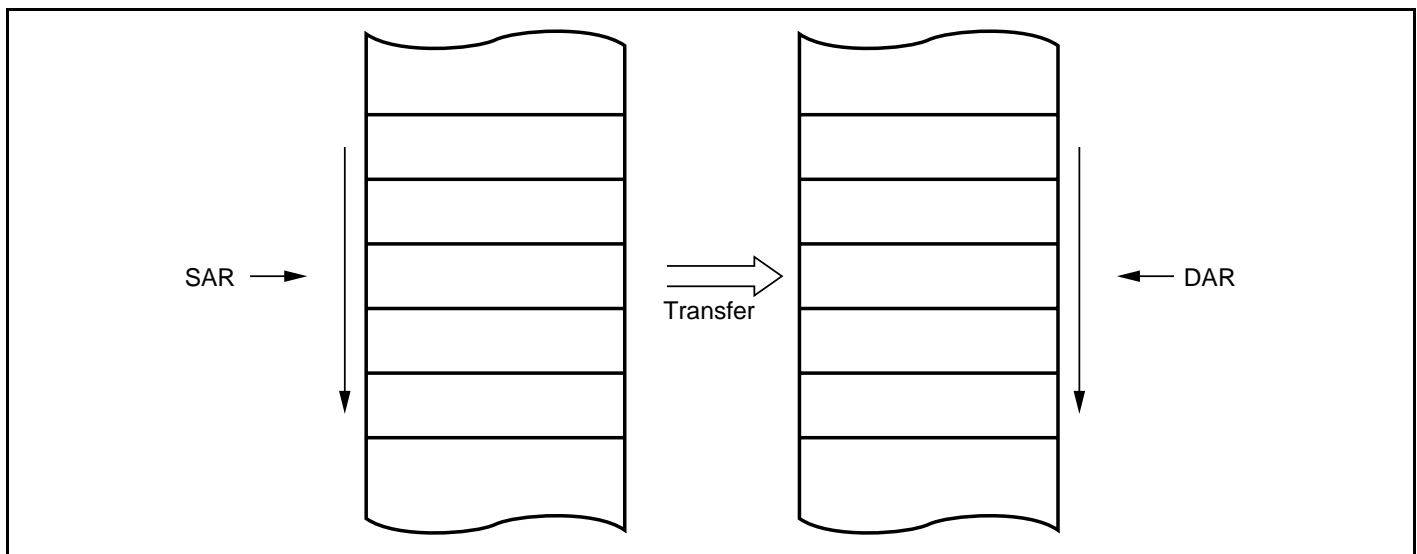


### 7.5.1 Normal Mode

In normal mode, one activation source transfers one byte or one word of data. Table 7.3 lists the register functions in normal mode. From 1 to 65,536 transfers can be specified. Once the specified number of transfers have been completed, a CPU interrupt can be requested.

**Table 7.3 Register Functions in Normal Mode**

Name	Abbreviation	Function
DTC source address register	SAR	Transfer source address
DTC destination address register	DAR	Transfer destination address
DTC transfer count register A	CRA	Transfer counter
DTC transfer count register B	CRB	Not used



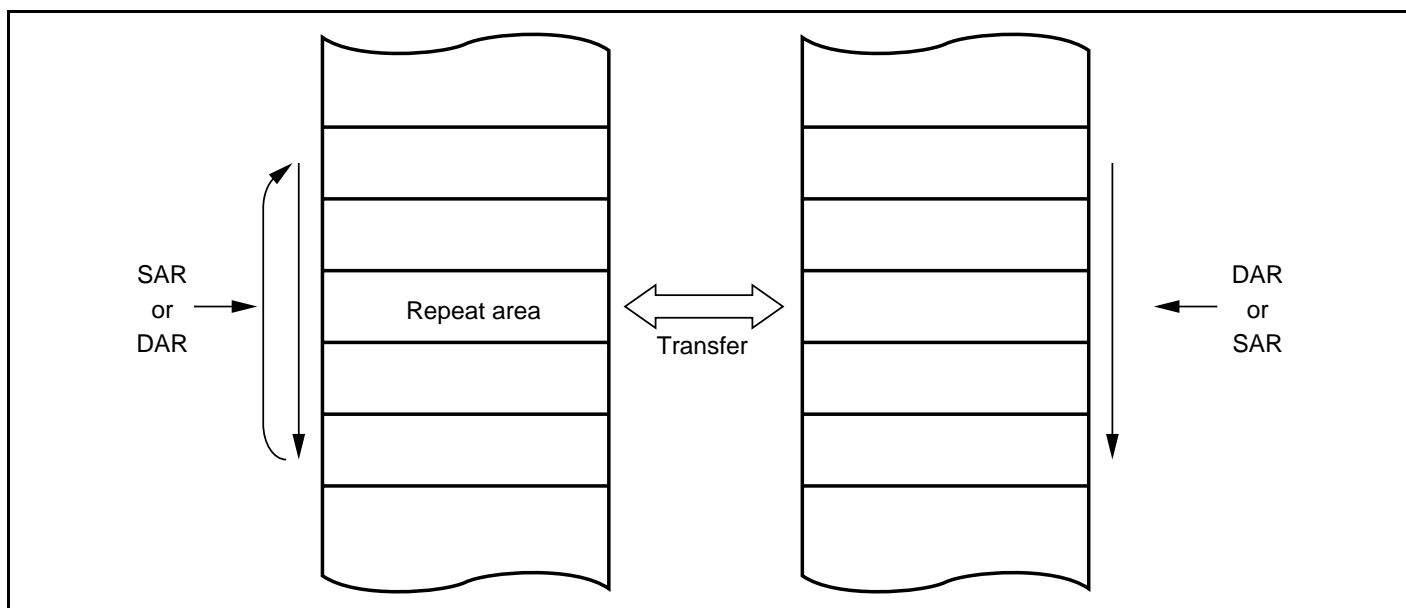
**Figure 7.5 Memory Mapping in Normal Mode**

## 7.5.2 Repeat Mode

In repeat mode, one activation source transfers one byte or one word of data. Table 7.4 lists the register functions in repeat mode. From 1 to 256 transfers can be specified. Once the specified number of transfers have completed, the initial states of the transfer counter and the address register that is specified as the repeat area is restored, and transfer is repeated. In repeat mode, the transfer counter value does not reach H'00, and therefore CPU interrupts cannot be requested when the DISEL bit in MRB is cleared to 0.

**Table 7.4 Register Functions in Repeat Mode**

Name	Abbreviation	Function
DTC source address register	SAR	Transfer source address
DTC destination address register	DAR	Transfer destination address
DTC transfer count register AH	CRAH	Holds number of transfers
DTC transfer count register AL	CRAL	Transfer counter
DTC transfer count register B	CRB	Not used



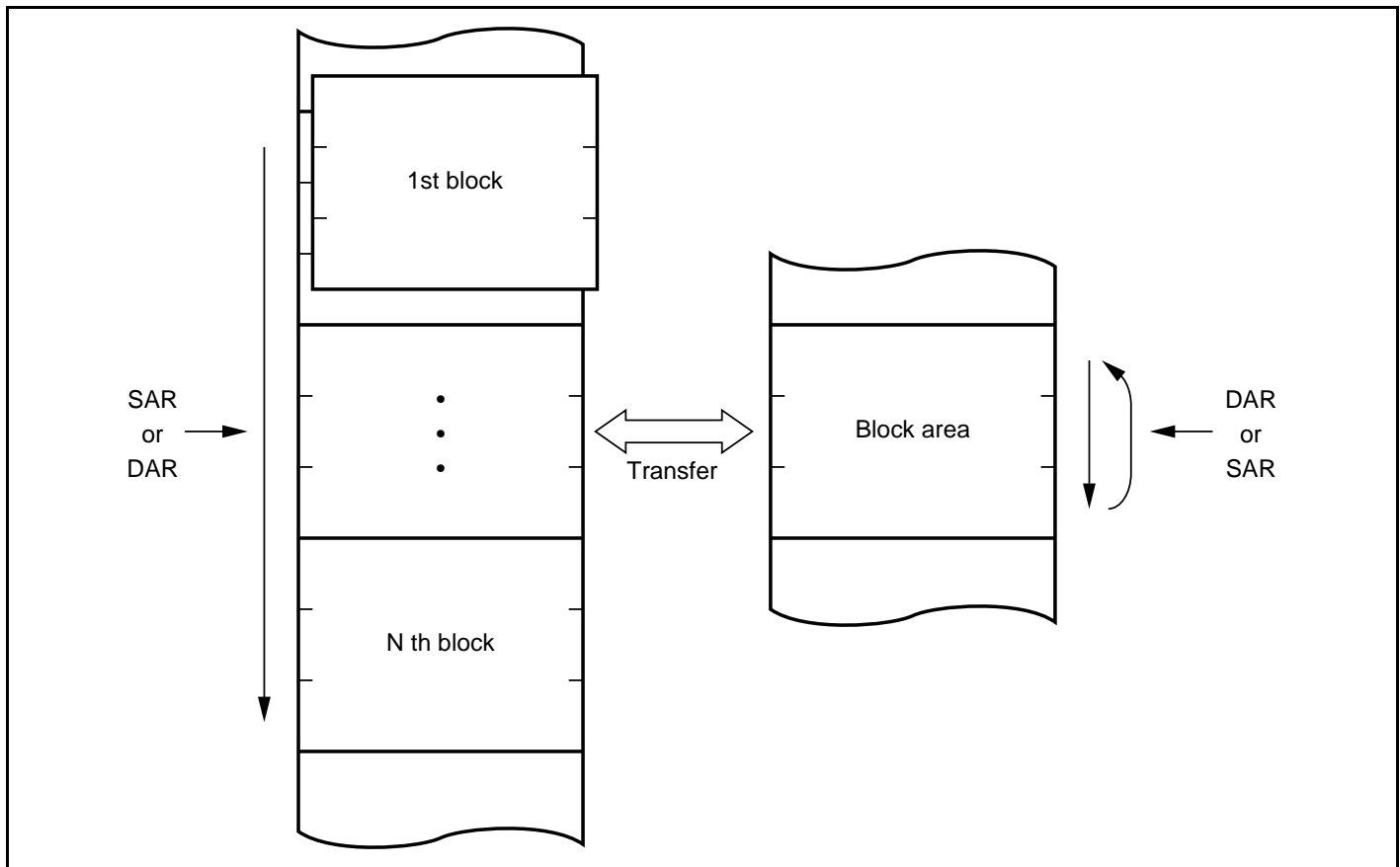
**Figure 7.6 Memory Mapping in Repeat Mode**

### 7.5.3 Block Transfer Mode

In block transfer mode, one activation source transfers one block of data. Either the transfer source or the transfer destination is designated as a block area. Table 7.5 lists the register functions in block transfer mode. The block size can be between 1 and 256. When the transfer of one block ends, the initial state of the block size counter and the address register that is specified as the block area is restored. The other address register is then incremented, decremented, or left fixed according to the register information. From 1 to 65,536 transfers can be specified. Once the specified number of transfers have been completed, a CPU interrupt is requested.

**Table 7.5 Register Functions in Block Transfer Mode**

Name	Abbreviation	Function
DTC source address register	SAR	Transfer source address
DTC destination address register	DAR	Transfer destination address
DTC transfer count register AH	CRAH	Holds block size
DTC transfer count register AL	CRAL	Block size counter
DTC transfer count register B	CRB	Transfer counter



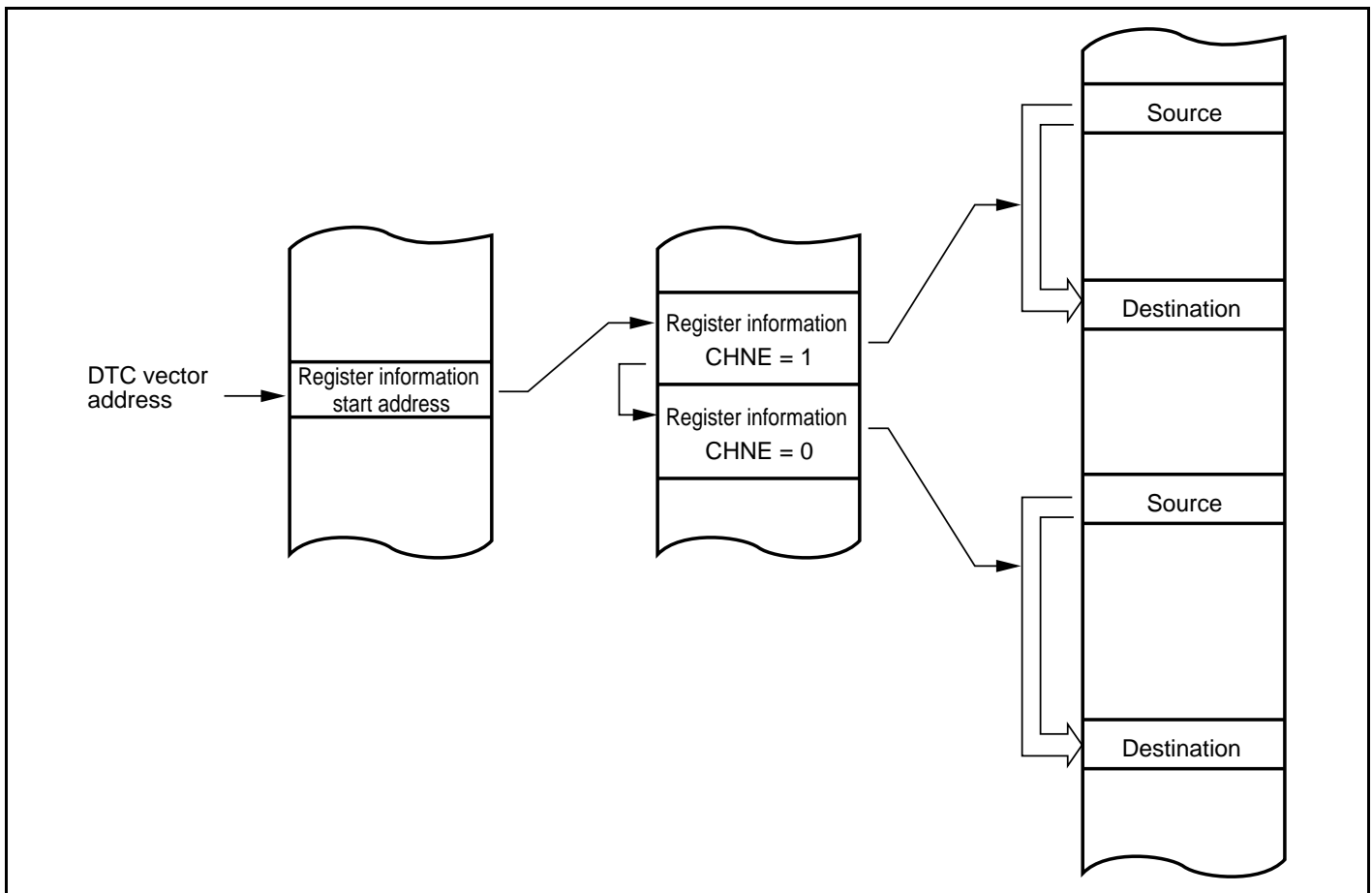
**Figure 7.7 Memory Mapping in Block Transfer Mode**

### 7.5.4 Chain Transfer

Setting the CHNE bit in MRB to 1 enables a number of data transfers to be performed consecutively in response to a single transfer request. SAR, DAR, CRA, CRB, MRA, and MRB, which define data transfers, can be set independently.

Figure 7.8 shows the overview of chain transfer operation. When activated, the DTC reads the register information start address stored at the DTC vector address, and then reads the first register information at that start address. After the data transfer, the CHNE bit will be tested. When it has been set to 1, DTC reads the next register information located in a consecutive area and performs the data transfer. These sequences are repeated until the CHNE bit is cleared to 0.

In the case of transfer with the CHNE bit set to 1, an interrupt request to the CPU is not generated at the end of the specified number of transfers or by setting of the DISEL bit to 1, and the interrupt source flag for the activation source is not affected.



**Figure 7.8 Chain Transfer Operation**

### 7.5.5 Interrupts

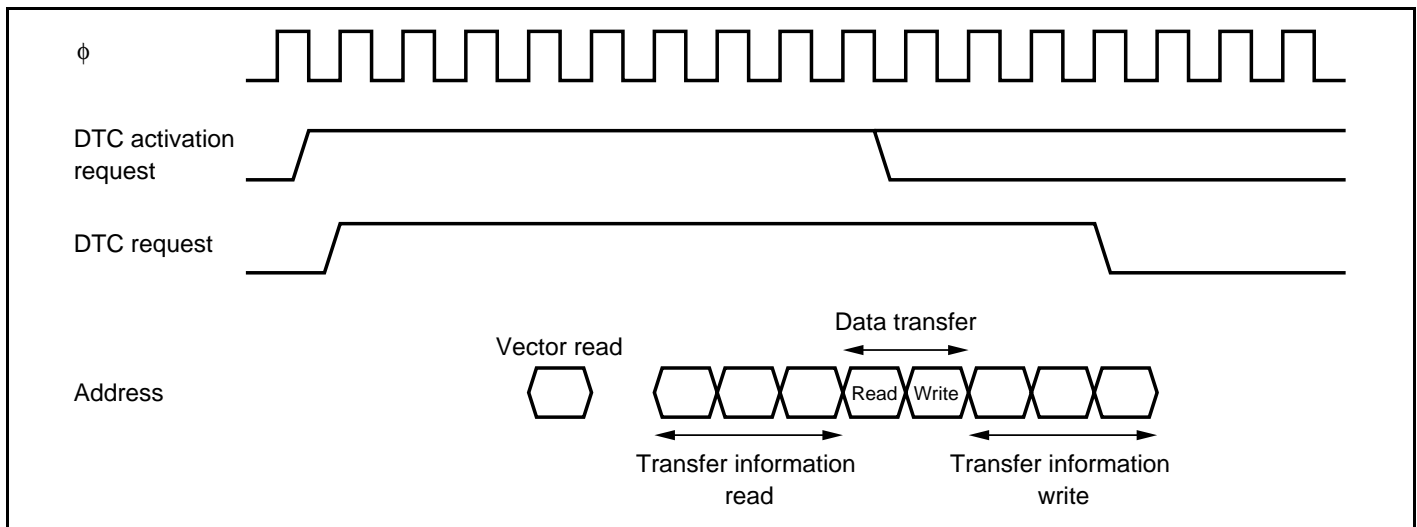
An interrupt request is issued to the CPU when the DTC has completed the specified number of data transfers, or a data transfer for which the DISEL bit was set to 1. In the case of interrupt activation, the interrupt set as the activation source is generated. These interrupts to the CPU are subject to CPU mask level and priority level control by the interrupt controller.

In the case of software activation, a software-activated data transfer end interrupt (SWDTEND) is generated.

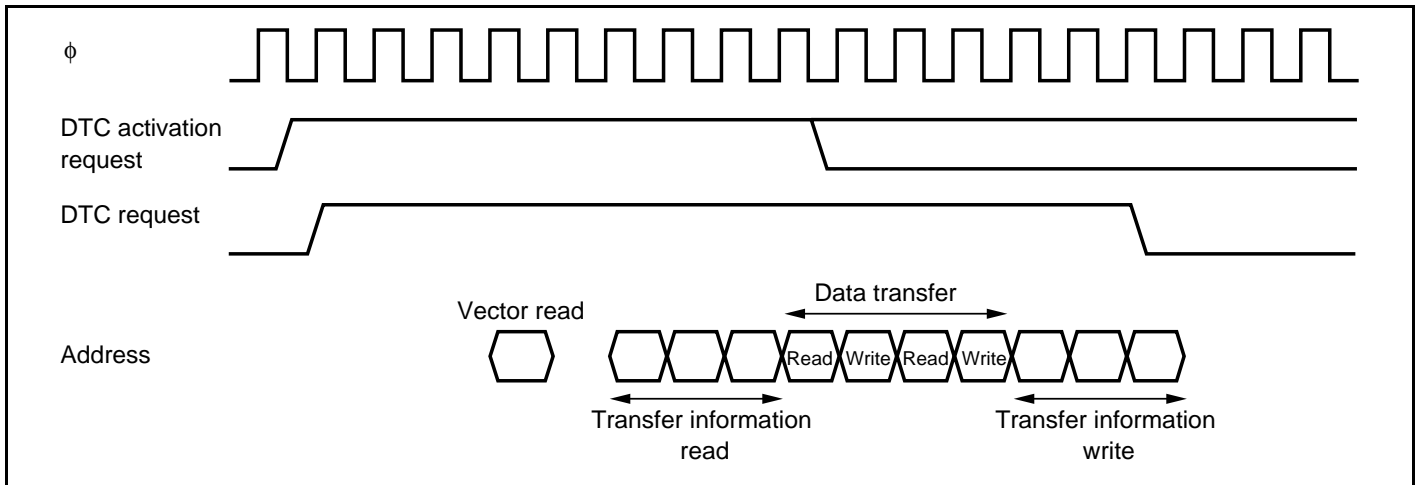
When the DISEL bit is 1 and one data transfer has been completed, or the specified number of transfers have been completed, after data transfer ends, the SWDTE bit is held at 1 and an SWDTEND interrupt is generated. The interrupt handling routine will then clear the SWDTE bit to 0.

When the DTC is activated by software, an SWDTEND interrupt is not generated during a data transfer wait or during data transfer even if the SWDTE bit is set to 1.

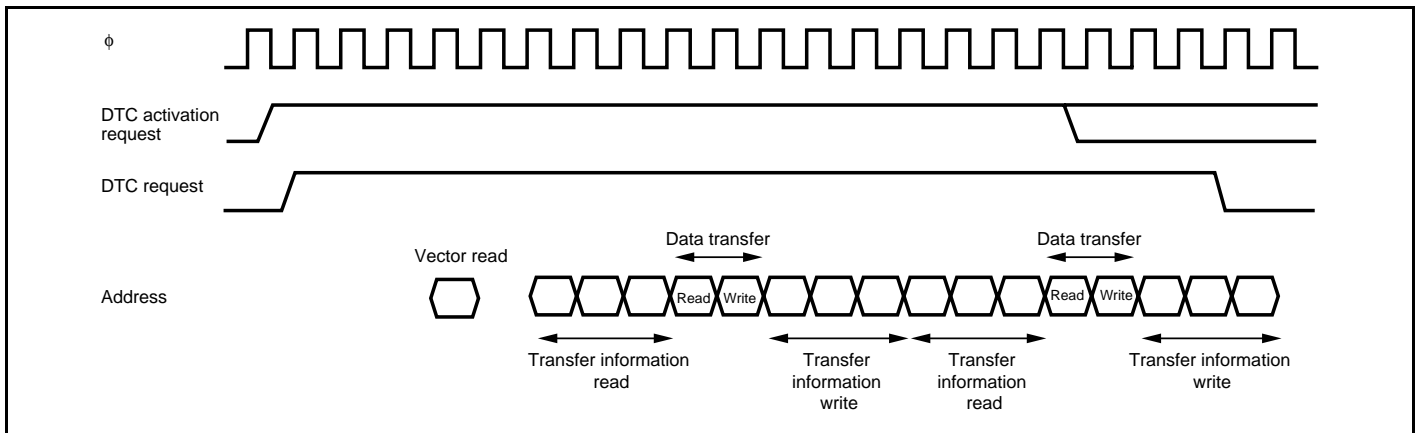
### 7.5.6 Operation Timing



**Figure 7.9 DTC Operation Timing (Example in Normal Mode or Repeat Mode)**



**Figure 7.10 DTC Operation Timing (Example of Block Transfer Mode, with Block Size of 2)**



**Figure 7.11 DTC Operation Timing (Example of Chain Transfer)**

### 7.5.7 Number of DTC Execution States

Table 7.6 lists the execution status for a single DTC data transfer, and table 7.7 shows the number of states required for each execution status.

**Table 7.6 DTC Execution Status**

Mode	Register Information				
	Vector Read I	Read/Write J	Data Read K	Data Write L	Internal Operations M
Normal	1	6	1	1	3
Repeat	1	6	1	1	3
Block transfer	1	6	N	N	3

N: Block size (initial setting of CRAH and CRAL)

**Table 7.7 Number of States Required for Each Execution Status**

Object to be Accessed		On-Chip RAM (H'(FF)EC00 to H'(FF)EFFF)		On-Chip RAM (On-Chip RAM other than left)		On- Chip ROM		On-Chip I/O Registers		External Devices				
Bus width			32		16		16		8	16	8	8	16	16
Access states			1		1		1		2	2	2	3	2	3
Execution status	Vector read	$S_I$	—		—		1		—	—	4	$6 + 2m$	2	$3 + m$
	Register information read/write	$S_J$	1		—		—		—	—	—	—	—	—
	Byte data read	$S_K$	1		1		1		2	2	2	$3 + m$	2	$3 + m$
	Word data read	$S_K$	1		1		1		4	2	4	$6 + 2m$	2	$3 + m$
	Byte data write	$S_L$	1		1		1		2	2	2	$3 + m$	2	$3 + m$
	Word data write	$S_L$	1		1		1		4	2	4	$6 + 2m$	2	$3 + m$
	Internal operation	$S_M$	1		1		1		1	1	1	1	1	1

The number of execution states is calculated from using the formula below. Note that  $\Sigma$  is the sum of all transfers activated by one activation source (the number in which the CHNE bit is set to 1, plus 1).

$$\text{Number of execution states} = I \cdot S_I + \Sigma (J \cdot S_J + K \cdot S_K + L \cdot S_L) + M \cdot S_M$$

For example, when the DTC vector address table is located in on-chip ROM, normal mode is set, and data is transferred from on-chip ROM to an internal I/O register, then the time required for the DTC operation is 13 states. The time from activation to the end of data write is 10 states.

## 7.6 Procedures for Using DTC

### 7.6.1 Activation by Interrupt

The procedure for using the DTC with interrupt activation is as follows:

1. Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in on-chip RAM.
2. Set the start address of the register information in the DTC vector address.
3. Set the corresponding bit in DTCER to 1.
4. Set the enable bits for the interrupt sources to be used as the activation sources to 1. The DTC is activated when an interrupt used as an activation source is generated.
5. After one data transfer has been completed, or after the specified number of data transfers have been completed, the DTCE bit is cleared to 0 and a CPU interrupt is requested. If the DTC is to continue transferring data, set the DTCE bit to 1.

### 7.6.2 Activation by Software

The procedure for using the DTC with software activation is as follows:

1. Set the MRA, MRB, SAR, DAR, CRA, and CRB register information in on-chip RAM.
2. Set the start address of the register information in the DTC vector address.
3. Check that the SWDTE bit is 0.
4. Write 1 to the SWDTE bit and the vector number to DTVECR.
5. Check the vector number written to DTVECR.
6. After one data transfer has been completed, if the DISEL bit is 0 and a CPU interrupt is not requested, the SWDTE bit is cleared to 0. If the DTC is to continue transferring data, set the SWDTE bit to 1. When the DISEL bit is 1 or after the specified number of data transfers have been completed, the SWDTE bit is held at 1 and a CPU interrupt is requested.

## 7.7 Examples of Use of the DTC

### 7.7.1 Normal Mode

An example is shown in which the DTC is used to receive 128 bytes of data via the SCI.

1. Set MRA to a fixed source address ( $SM1 = SM0 = 0$ ), incrementing destination address ( $DM1 = 1$ ,  $DM0 = 0$ ), normal mode ( $MD1 = MD0 = 0$ ), and byte size ( $Sz = 0$ ). The DTS bit can have any value. Set MRB for one data transfer by one interrupt ( $CHNE = 0$ ,  $DISEL = 0$ ). Set the



SCI's RDR address in SAR, the start address of the RAM area where the data will be received in DAR, and 128 (H'0080) in CRA. CRB can be set to any value.

2. Set the start address of the register information at the DTC vector address.
3. Set the corresponding bit in DTCER to 1.
4. Set the SCI to the appropriate receive mode. Set the RIE bit in SCR to 1 to enable the reception complete (RXI) interrupt. Since the generation of a receive error during the SCI reception operation will disable subsequent reception, the CPU should be enabled to accept receive error interrupts.
5. Each time the reception of one byte of data has been completed on the SCI, the RDRF flag in SSR is set to 1, an RXI interrupt is generated, and the DTC is activated. The receive data is transferred from RDR to RAM by the DTC. DAR is incremented and CRA is decremented. The RDRF flag is automatically cleared to 0.
6. When CRA becomes 0 after 128 data transfers have been completed, the RDRF flag is held at 1, the DTCE bit is cleared to 0, and an RXI interrupt request is sent to the CPU. The interrupt handling routine will perform wrap-up processing.

### 7.7.2 Software Activation

An example is shown in which the DTC is used to transfer a block of 128 bytes of data by means of software activation. The transfer source address is H'1000 and the transfer destination address is H'2000. The vector number is H'60, so the vector address is H'04C0.

1. Set MRA to incrementing source address (SM1 = 1, SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), block transfer mode (MD1 = 1, MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one block transfer by one interrupt (CHNE = 0). Set the transfer source address (H'1000) in SAR, the transfer destination address (H'2000) in DAR, and 128 (H'8080) in CRA. Set 1 (H'0001) in CRB.
2. Set the start address of the register information at the DTC vector address (H'04C0).
3. Check that the SWDTE bit in DTVECR is 0. Check that there is currently no transfer activated by software.
4. Write 1 to the SWDTE bit and the vector number (H'60) to DTVECR. The write data is H'E0.
5. Read DTVECR again and check that it is set to the vector number (H'60). If it is not, this indicates that the write failed. This is presumably because an interrupt occurred between steps 3 and 4 and led to a different software activation. To activate this transfer, go back to step 3.
6. If the write was successful, the DTC is activated and a block of 128 bytes of data is transferred.
7. After the transfer, an SWDTEND interrupt occurs. The interrupt handling routine should clear the SWDTE bit to 0 and perform wrap-up processing.

## 7.8 Usage Notes

### 7.8.1 Module Stop Mode Setting

DTC operation can be enabled or disabled by the module stop control register (MSTPCR). In the initial state, DTC operation is enabled. Access to DTC registers are disabled when module stop mode is set. Note that when the DTC is being activated, module stop mode can be specified. For details, refer to section 27, Power-Down Modes.

### 7.8.2 On-Chip RAM

MRA, MRB, SAR, DAR, CRA, and CRB are all located in on-chip RAM. When the DTC is used, the RAME bit in SYSCR should not be cleared to 0.

### 7.8.3 DTCE Bit Setting

For DTCE bit setting, use bit manipulation instructions such as BSET and BCLR, for reading and writing. Multiple DTC activation sources can be set at one time (only at the initial setting) by masking all interrupts and writing data after executing a dummy read on the relevant register.

### 7.8.4 Setting Required on Entering Subactive Mode or Watch Mode

Set the MSTP14 bit in MSTPCRH to 1 to make the DTC enter module stop mode, then confirm that is set to 1 before making a transition to subactive mode or watch mode.

### 7.8.5 DTC Activation by Interrupt Sources of SCI, IIC, or A/D Converter

Interrupt sources of the SCI, IIC, or A/D converter which activate the DTC are cleared when DTC reads from or writes to the respective registers, and they cannot be cleared by the DISEL bit in MRB.

### 7.8.6 DTC Activation by Interrupt Sources of USB or MCIF

When activating the DTC by a USB or MCIF interrupt source, correct operation is not guaranteed if the DISEL bit in MRB is cleared to 0. Be sure to set the DISEL bit to 1 before DTC activation.

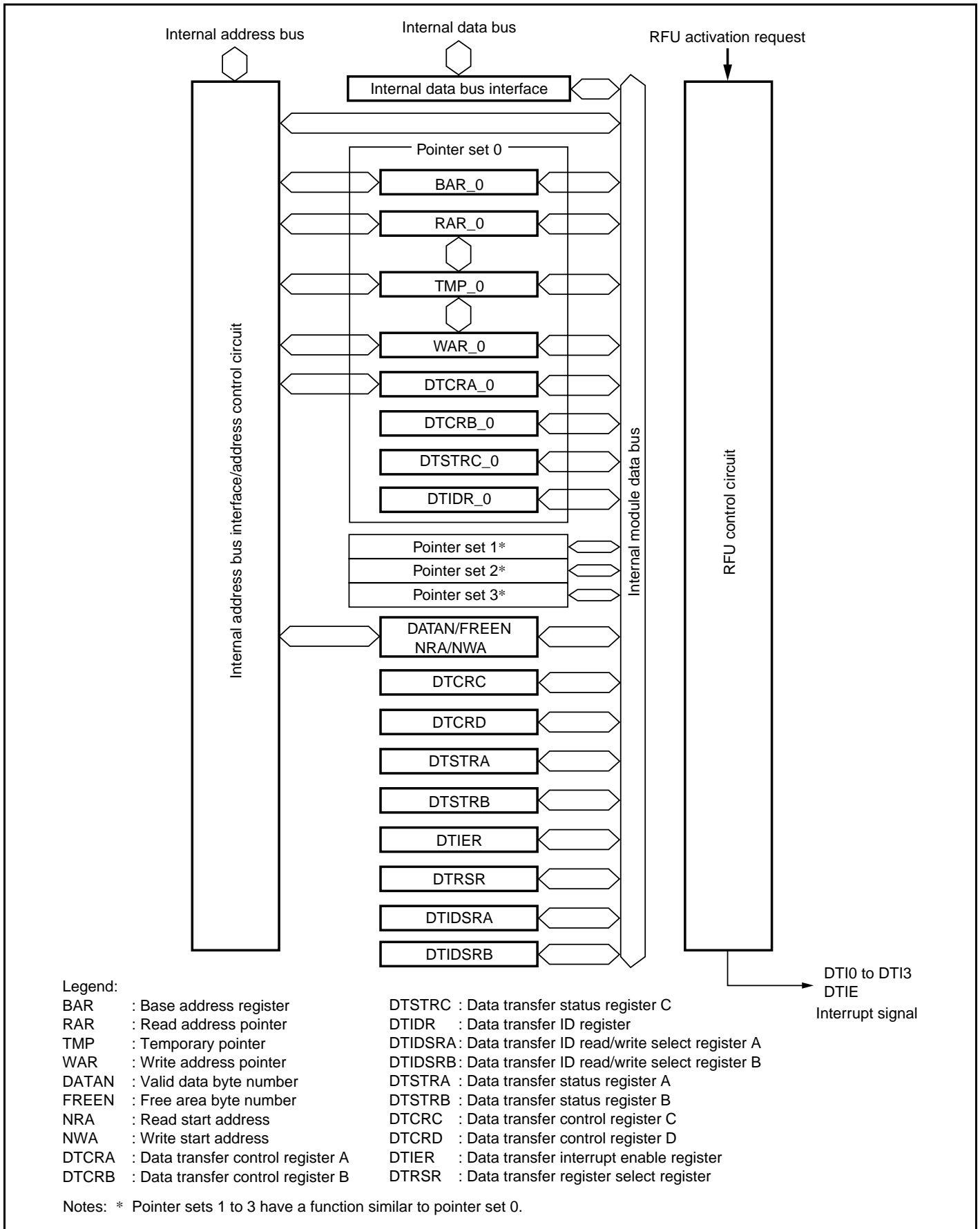
## Section 8 RAM-FIFO Unit (RFU)

This LSI incorporates a RAM-FIFO unit (RFU). The RFU is activated by a request from the peripheral modules and can transfer data between the peripheral modules and on-chip RAM. As the RFU can specify the RAM address to be transferred by using a pointer that is updated for every data transfer execution, the RAM specified area can be regarded as an FIFO. If an FIFO full/empty or overrun error occurs according to pointer update, the RFU can acknowledge this error to the peripheral modules. The peripheral modules request pointer reset and manipulation of the temporary pointer in addition to data transfer.

A block diagram of the RFU is shown in figure 8.1.

### 8.1 Features

- Bus master with priority higher than that of the CPU and DTC
- Provides the RFU-ID to specific peripheral modules (SCI, USB, and MCIF) to specify the peripheral modules to be manipulated by the RFU with ID numbers
- RFU bus cycle accesses the peripheral modules and on-chip RAM simultaneously
- During an RFU bus cycle, the address bus outputs a RAM address for data transfer
- RAM address for data transfer is specified by the on-chip pointer set of the RFU
- Four pointer sets
- The contents of the pointer set are updated for every data transfer, and a specific RAM area can be manipulated as the FIFO (RAM-FIFO)
- RAM-FIFO size: 32/64/128/256/512/1024/2048 bytes selectable
- An interrupt can be generated by a RAM-FIFO full/empty or overrun error
- RFU operates in high-speed mode even when the LSI is in medium-speed mode



**Figure 8.1 Block Diagram of RFU**

## 8.2 Register Descriptions

The RFU has the following registers.

- FIFO status/register/pointer (FSTR)
  - Base address register (BAR)
  - Read address pointer (RAR)
  - Write address pointer (WAR)
  - Temporary pointer (TMP)
  - Valid data byte number (DATAN)
  - Free area byte number (FREEN)
  - Read start address (NRA)
  - Write start address (NWA)
- Data transfer control register A (DTCRA)
- Data transfer control register B (DTCRB)
- Data transfer status register C (DTSTRC)
- Data transfer ID register (DTIDR)
- Data transfer ID read/write select register A (DTIDSRA)
- Data transfer ID read/write select register B (DTIDSRB)
- Data transfer status register A (DTSTRA)
- Data transfer status register B (DTSTRB)
- Data transfer control register C (DTCRC)
- Data transfer control register D (DTCRD)
- Data transfer interrupt enable register (DTIER)
- Data transfer select register (DTRSR)

### 8.2.1 FIFO Status/Register/Pointer (FSTR)

FSTR is a 32-bit readable/writable register used to access the FIFO status/register/pointer of the pointer sets (0 to 3). Any one of four registers/pointers (BAR, RAR, WAR, and TMP) and four states (DATAN, FREEN, NRA, and NWA) can be accessed using FSTR.

The status is updated by writing (selecting pointer or status) to DTRSR. The new status should be referenced after writing to DTRSR, even when referencing the same pointer or same status.

BAR in pointer set 0 is selected via the initial value of DTRSR, however, an undefined value is read from FSTR until data is written to DTRSR.

### 8.2.2 Base Address Register (BAR)

BAR is a 16-bit register provided in each pointer set, and allocated to bits 19 to 4 in FSTR. The base address should be set at the boundary specified by the BUD2 to BUD0 bits in DTCRA. Otherwise, address specification by the pointer and status display by calculation inter-pointers may not be performed correctly.

Bit	Bit Name	Initial Value	R/W	Description
31 to 20	—	All 1	R	Base Addresses 31 to 20 These bits are always read as 1 and cannot be modified.
19 to 4	BA19 to BA4	Undefined	R/W	Base Addresses 19 to 4 These bits specify a RAM base address that can be used as the FIFO.
3 to 0	—	All 0	R	Base Addresses 3 to 0 These bits are always read as 0 and cannot be modified.

### 8.2.3 Read Address Pointer (RAR)

RAR is an 11-bit pointer provided in each pointer set, and allocated to bits 10 to 0 in FSTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 11	—	All 0	R	Read Addresses 31 to 11 These bits are always read as 0 and cannot be modified.
10 to 0	RA10 to RA0	All 0	R/W	Read Addresses 10 to 0 These bits are pointers to specify the RAM address to be read from in a RAM read cycle of the RFU. The RAM address is calculated by $BAR + RAR$ . These can be read as NRA. These bits are incremented for the number of bytes to be read for each RAM read cycle. However, these bits are not incremented and cleared to 0 when exceeding the selected FIFO size.

### 8.2.4 Write Address Pointer (WAR)

WAR is an 11-bit pointer provided in each pointer set, and allocated to bits 10 to 0 in FSTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 11	—	All 0	R	Write Addresses 31 to 11 These bits are always read as 0 and cannot be modified.
10 to 0	WA10 to WA0	All 0	R/W	Write Addresses 10 to 0 These bits are pointers to specify the RAM address to be written to in a RAM write cycle of the RFU. The RAM address is calculated by BAR + WAR. These can be read as NWA. These bits are incremented for the number of bytes to be written for each RAM write cycle. However, these bits are not incremented and cleared to 0 when exceeding the selected FIFO size.

### 8.2.5 Temporary Pointer (TMP)

TMP is an 11-bit pointer provided in each pointer set, and allocated to bits 10 to 0 in FSTR. TMP can select the pointer set operation by setting the PMD1 and PMD0 bits in DTCRA.

Bit	Bit Name	Initial Value	R/W	Description
31 to 11	—	All 0	R	Temporary Addresses 31 to 11 These bits are always read as 0 and cannot be modified.
10 to 0	TMP10 to TMP0	All 0	R/W	Temporary Addresses 10 to 0 [When TMP is used as a read temporary pointer] These bits can copy the contents of RAR to TMP or copy the contents of TMP to RAR by pointer update manipulation. [When TMP is used as a write temporary pointer] These bits can copy the contents of WAR to TMP or copy the contents of TMP to WAR by pointer update manipulation.

### 8.2.6 Valid Data Byte Number (DATAN)

DATAN is 11-bit status data allocated to bits 10 to 0 in FSTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 11	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
10 to 0	—	All 0	R	Valid Data Byte Number These bits indicate the number of bytes of valid data that can be read by FIFO in each pointer set.

### 8.2.7 Free Area Byte Number (FREEN)

FREEN is 11-bit status data allocated to bits 10 to 0 in FSTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 11	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
10 to 0	—	All 0	R	Free Area Byte Number These bits indicate the number of bytes of the free area that can be written to FIFO in each pointer set.

### 8.2.8 Read Start Address (NRA)

NRA is 32-bit status data allocated to bits 31 to 0 in FSTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	—	Undefined	R	Read Start Addresses 31 to 0 The RAM address is calculated by BAR + RAR



### 8.2.9 Write Start Address (NWA)

NWA is 32-bit status data allocated to bits 31 to 0 in FSTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	—	Undefined	R	Write Start Addresses 31 to 0 The RAM address is calculated by BAR + WAR

### 8.2.10 Data Transfer Control Register A (DTCRA)

DTCRA is a register provided in each pointer set that controls the operation of each pointer set.

Bit	Bit Name	Initial Value	R/W	Description
7	IDE-A	0	R/W	ID-A Enable Enables/disables ID-A selected by DTIDR. 0: Disables ID-A 1: Enables ID-A
6	IDE-B	0	R/W	ID-B Enable Enables/disables ID-B selected by DTIDR. 0: Disables ID-B 1: Enables ID-B
5	PMD1	0	R/W	Pointer mode 1, 0
4	PMD0	0	R/W	These bits select operation mode of the TMP. 0X: TMP is not used 10: TMP is used as the write temporary pointer 11: TMP is used as the read temporary pointer
3	Sz	0	R/W	Transfer Size Selects data size of the bus cycle that is activated by the peripheral modules. 0: Byte transfer 1: Word transfer

Bit	Bit Name	Initial Value	R/W	Description
2	BUD2	1	R/W	Boundary 2 to 0
1	BUD1	1	R/W	These bits select the FIFO size and the existence of boundary overflow. 000: 32 bytes 001: 64 bytes 010: 128 bytes 011: 256 bytes 100: 512 bytes 101: 1024 bytes 110: 2048 bytes 111: Boundary is not set
0	BUD0	0	R/W	

Legend:

X: Don't care

Table 8.1 shows how to make settings for BAR, RAR, WAR, and TMP.

In BAR, the bits upper than the boundary become valid depending on the number of FIFO bytes set in the BUD2 to BUD0 bits. Since the lower bits become invalid, 0 should be written to these bits. In RAR, WAR, and TMP, the bits lower than the boundary become valid. Since the upper bits become invalid, 0 should be written to these bits.

**Table 8.1 Valid Bits in BAR, RAR, WAR, and TMP**

Number of FIFO Bytes	BAR Valid Section	BAR Invalid Section (Should be set to 0)	Invalid Sections in RAR, WAR, and TMP (Should be set to 0, no carry)	Valid Sections in RAR, WAR, and TMP
32 bytes	A19 to A5	A4	A10 to A5	A4 to A0
64 bytes	A19 to A6	A5 and A4	A10 to A6	A5 to A0
128 bytes	A19 to A7	A6 to A4	A10 to A7	A6 to A0
256 bytes	A19 to A8	A7 to A4	A10 to A8	A7 to A0
512 bytes	A19 to A9	A8 to A4	A10 and A9	A8 to A0
1024 bytes	A19 to A10	A9 to A4	A10	A9 to A0
2048 bytes	A19 to A11	A10 to A4	—	A10 to A0

### 8.2.11 Data Transfer Control Register B (DTCRB)

DTCRB is a register provided in each pointer set that controls the operation of the interrupt flag in each pointer set and the data transfer between pointers.

Bit	Bit Name	Initial Value	R/W	Description
7	BOVF_RE	0	R/W	<p>Boundary Overflow Enable (at reading)</p> <p>Selects whether to reflect RAR boundary overflow to the BOVF_R flag in DTSTRC.</p> <p>0: Boundary overflow (at reading) is not reflected to the BOVF_R flag</p> <p>1: Boundary overflow (at reading) is reflected to the BOVF_R flag</p>
6	BOVF_WE	0	R/W	<p>Boundary Overflow Enable (at writing)</p> <p>Selects whether to reflect WAR boundary overflow to the BOVF_W flag in DTSTRC.</p> <p>0: Boundary overflow (at writing) is not reflected to the BOVF_W flag</p> <p>1: Boundary overflow (at writing) is reflected to the BOVF_W flag</p>
5	FULLE	0	R/W	<p>FIFO Full Enable</p> <p>Selects whether to reflect detection of FIFO full (generation of WAR = RAR or TMP = WAR (when the read temporary pointer is selected) according to the write bus cycle) to the FULL flag in DTSTRC.</p> <p>0: FIFO full detection is not reflected to the FULL flag</p> <p>1: FIFO full detection is reflected to the FULL flag</p>
4	EMPTYE	0	R/W	<p>FIFO Empty Enable</p> <p>Selects whether to reflect detection of FIFO empty (generation of RAR = WAR or TMP = RAR (when the write temporary pointer is selected) according to the read bus cycle) to the EMPTY flag in DTSTRC.</p> <p>0: FIFO empty detection is not reflected to the EMPTY flag</p> <p>1: FIFO empty detection is reflected to the EMPTY flag</p>

Bit	Bit Name	Initial Value	R/W	Description
3	LOAD	0	W	<p>Pointer Reload</p> <p>If this bit is set to 1 when the TMP settings are made, this bit copies the contents of TMP to RAR or WAR.</p> <p>When the read temporary pointer is used, the contents of TMP are copied to RAR.</p> <p>When the write temporary pointer is used, the contents of TMP are copied to WAR.</p>
2	MARK	0	W	<p>Pointer Mark</p> <p>If this bit is set to 1 when the TMP settings are made, this bit copies the contents of RAR or WAR to TMP.</p> <p>When the read temporary pointer is used, the contents of RAR are copied to TMP.</p> <p>When the write temporary pointer is used, the contents of WAR are copied to TMP.</p>
1	REST	0	W	<p>Pointer Reset</p> <p>When this bit is set to 1, this bit initializes RAR, WAR, and TMP, and also returns the status to FIFO empty state.</p>
0	STCLR	0	W	<p>Status Clear</p> <p>Clears information of FIFO full and FIFO empty.</p>

Note: Do not set bits 1 and 0 to 1 simultaneously.

### 8.2.12 Data Transfer Status Register C (DTSTRC)

DTSTRC is a register provided in each pointer set. DTSTRC includes the interrupt flags for each pointer set.

When any one bit of bits 7 to 4 is set to 1, a flag corresponding to the pointer set number in DTSTRA is set. When any one bit of bits 3 and 2 is set to 1, a flag corresponding to the pointer set number in DTSTRB is set.

Bit	Bit Name	Initial Value	R/W	Description
7	BOVF_R	0	R/(W)	Boundary Overflow (at reading) Indicates detection of boundary overflow in RAR or TMP (when the read temporary pointer is selected). This flag can be masked by the BOVF_RE bit in DTCRB.
6	BOVF_W	0	R/(W)	Boundary Overflow (at writing) Indicates detection of boundary overflow in WAR or TMP (when the write temporary pointer is selected). This flag can be masked by the BOVF_WE bit in DTCRB.
5	FULL	0	R/(W)	FIFO Full Indicates detection of FIFO full (generation of WAR = RAR or TMP = WAR (when the read temporary pointer is selected) according to the write bus cycle). This flag can be masked by the FULLE bit in DTCRB.
4	EMPTY	0	R/(W)	FIFO Empty Indicates detection of FIFO empty (generation of WAR = RAR or TMP = RAR (when the write temporary pointer is selected) according to the read bus cycle). This flag can be masked by the EMPTYE bit in DTCRB.
3	OVER_R	0	R/(W)	FIFO Over Read Indicates detection of FIFO empty (when a read request is generated while RAR = WAR or TMP = RAR (when the write temporary pointer is selected)).
2	OVER_W	0	R/(W)	FIFO Over Write Indicates detection of FIFO full (when a write request is generated while WAR = RAR or TMP = WAR (when the read temporary pointer is selected)).
1, 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

### 8.2.13 Data Transfer ID Register (DTIDR)

DTIDR is a register provided in each pointer set. DTIDR selects the peripheral module, which is an activation source of each pointer set.

A 4-bit ID has been assigned to the peripheral modules. DTIDR selects two IDs. The ID selected by DTIDR is enabled by setting the IDE-A and IDE-B bits in DTCRA to 1.

When selecting an ID, the following two points should be noted:

- To select two IDs, the IDs should be combined such that the data transfer direction is read and write.
- The same IDs should not be selected over several pointer sets.

Bit	Bit Name	Initial Value	R/W	Description
7	ID-A3	0	R/W	ID-A Select
6	ID-A2	0	R/W	These bits write the ID number to be selected by the IDE-A bit.
5	ID-A1	0	R/W	
4	ID-A0	0	R/W	
3	ID-B3	0	R/W	
2	ID-B2	0	R/W	These bits write the ID number to be selected by the IDE-B bit.
1	ID-B1	0	R/W	
0	ID-B0	0	R/W	

### 8.2.14 Data Transfer ID Read/Write Select Register A (DTIDSRA)

DTIDSRA selects the direction for transferring ID15 to ID8. As IDs have already been assigned for the peripheral modules, the transfer direction is fixed. For details, refer to section 8.8, Operation.

Bit	Bit Name	Initial Value	R/W	Description
7	IDRW15	0	R/W	ID15 R/W to ID8 R/W
6	IDRW14	0	R/W	These bits select the direction for transferring peripheral modules with ID numbers 15 to 8.
5	IDRW13	0	R/W	
4	IDRW12	0	R/W	
3	IDRW11	0	R/W	
2	IDRW10	0	R/W	0: RAM → Peripheral modules (write) 1: Peripheral modules (read) → RAM
1	IDRW9	0	R/W	
0	IDRW8	0	R/W	

### 8.2.15 Data Transfer ID Read/Write Select Register B (DTIDSRB)

DTIDSRB selects the direction for transferring ID7 to ID0. As IDs have already been assigned for the peripheral modules, the transfer direction is fixed. For details, refer to section 8.8, Operation.

Bit	Bit Name	Initial Value	R/W	Description
7	IDRW7	0	R/W	ID7 R/W to ID0 R/W
6	IDRW6	0	R/W	These bits select the direction for transferring peripheral modules with ID numbers 7 to 0. 0: RAM → Peripheral modules (write) 1: Peripheral modules (read) → RAM
5	IDRW5	0	R/W	
4	IDRW4	0	R/W	
3	IDRW3	0	R/W	
2	IDRW2	0	R/W	
1	IDRW1	0	R/W	
0	IDRW0	0	R/W	

### 8.2.16 Data Transfer Status Register A (DTSTRA)

DTSTRA includes interrupt flags for each pointer set.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/(W)	Reserved The initial value should not be changed.
3	DTF3	0	R/(W)*	Data Transfer Interrupt Flags 3 to 0
2	DTF2	0	R/(W)*	These are interrupt flags for pointer set numbers 3 to 0. 0: No interrupt source 1: Any one flag of BOVF_R, BOVF_W, FULL, or EMPTY is set to 1
1	DTF1	0	R/(W)*	
0	DTF0	0	R/(W)*	

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

### 8.2.17 Data Transfer Status Register B (DTSTRB)

DTSTRB includes error interrupt flags for each pointer set.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/(W)	Reserved The initial value should not be changed.
3	DTEF3	0	R/(W)*	Data Transfer Error Interrupt Flags 3 to 1 These are error interrupt flags for pointer set numbers 3 to 1. 0: No interrupt source 1: Either flag of OVER_R or OVER_W is set to 1
2	DTEF2	0	R/(W)*	
1	DTEF1	0	R/(W)*	
0	DTEIE	0	R/W	Data Transfer Error Interrupt Enable 0: Disables an interrupt generated by DTEF3 to DTEF1. 1: Enables an interrupt generated by DTEF3 to DTEF 1.

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

### 8.2.18 Data Transfer Control Register C (DTCRC)

DTCRC includes a bit to assign the ID-6 and ID-7 functions.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R/(W)	Reserved The initial value should not be changed.
5 to 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.



### 8.2.19 Data Transfer Control Register D (DTCRD)

DTCRD includes enable bits for each pointer set. When the DTE bit is cleared to 0 and then reset to 1, the empty information is restored.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/(W)	Reserved The initial value should not be changed.
3	DTE3	0	R/W	Data Transfer Enable 3 to 0
2	DTE2	0	R/W	These are enable bits for pointer set numbers 3 to 0. 0: Disables pointer set 1: Enables pointer set
1	DTE1	0	R/W	
0	DTE0	0	R/W	

### 8.2.20 Data Transfer Interrupt Enable Register (DTIER)

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/(W)	Reserved The initial value should not be changed.
3	DTIE3	0	R/W	Data Transfer Interrupt Enable 3 to 0
2	DTIE2	0	R/W	These are interrupt enable bits for pointer set numbers 3 to 0. 0: Disables an interrupt generated by DTF 1: Enables an interrupt generated by DTF
1	DTIE1	0	R/W	
0	DTIE0	0	R/W	

### 8.2.21 Data Transfer Register Select Register (DTRSR)

DTRSR specifies the pointer set number or FIFO status/register/pointer accessed by FSTR. The resource of the specified pointer number can be accessed from FSTR, DTCRA, DTCRB, DTCRC, DTIDR, and DTSTRC by specifying the pointer set number.

Bit	Bit Name	Initial Value	R/W	Description
7	CHS2	0	R/W	Pointer Number Select
6	CHS1	0	R/W	These bits represent the pointer set number to be accessed from FSTR, DTCRA, DTCRB, DTIDR, and DTSTRC.
5	CHS0	0	R/W	
4	RS	0	R/W	Register Select Selects whether to access register/pointer or FIFO status by FSTR. 0: FSTR accesses register/pointer 1: FSTR accesses FIFO status
3	POS1	0	R/W	Register/Pointer Select 1, 0
2	POS0	0	R/W	These bits select the register/pointer to be accessed by FSTR while the RS bit is 0. 00: The base address register (BAR) is accessed by FSTR 01: The read address pointer (RAR) is accessed by FSTR 10: The write address pointer (WAR) is accessed by FSTR 11: The temporary pointer (TMP) is accessed by FSTR
1	STS1	0	R/W	FIFO Status Selects 1, 0
0	STS0	0	R/W	These bits select the FIFO status to be accessed by FSTR while the RS bit is 1. 00: The valid data byte number (DATATN) is accessed by FSTR 01: The free area byte number (FREEN) is accessed by FSTR 10: The read start address (NRA) is accessed by FSTR 11: The write start address (NWA) is accessed by FSTR To access NWA, NRA, FREEN, or DATATN while the RS bit is 0, write 1 to the RS bit twice.

### 8.3 Activation Source and Priority

The RFU operates upon a request from the peripheral modules regarding it as an activation source. In SCI, the request from the peripheral modules is an event that indicates 1-byte data transfer completion, such as setting the TDRE and RDRF bits. However, in USB, the request from the peripheral modules is an internal event that cannot be referenced by a CPU instruction. A total of 14 ID numbers have been assigned to the request from the peripheral modules. A peripheral module to which an ID number has not been assigned cannot use the RFU. Table 8.2 summarizes the correspondence between the ID numbers and activation sources.

When the RFU accepts multiple activation sources simultaneously, the RFU selects the ID with the highest priority. The ID is classified into two groups, and the priority inter-groups is fixed. The initial priority in the group is as shown in table 8.2. However, the priority is changed whenever the RFU performs processing. The priority in the group is like a loop, that is, the priority of the processed ID number becomes the lowest, and the priority of the next ID number becomes the highest.

**Table 8.2 Correspondence between Activation Sources and ID Numbers**

ID Number	Peripheral Module	Activation Source	Transfer Direction	Transfer Size	Priority in Group (Initial Value)	Group	Priority Inter-Groups (Fixed)
0	USB	EP4	RAM → USB	Byte	Highest	A	Highest
1	USB	EP5	USB → RAM	Byte			
2	USB	EP9	RAM → USB	Byte			
3	USB	EP10	USB → RAM	Byte			
4	USB	EP11	RAM → USB	Byte			
5	USB	EP12	USB → RAM	Byte			
6	USB	EP6I	RAM → USB	Byte			
7	USB	EP6O	USB → RAM	Byte	Lowest		
8	SCI_0	RDRF	SCI_0 → RAM	Byte	Highest	B	
9	SCI_0	TDRE	RAM → SCI_0	Byte			
10	SCI_2	RDRF	SCI_2 → RAM	Byte			
11	SCI_2	TDRE	RAM → SCI_2	Byte			
12	MCIF	MCIF → Multimedia card	RAM → Multimedia card	Byte			
13	MCIF	Multimedia card → MCIF	Multimedia card → RAM	Byte	Lowest		Lowest

## 8.4 RAM-FIFO Location

The RAM-FIFO should be allocated at the addresses H'(FF)E080 to H'(FF)EFFF and H'FF0800 to H'FF1FFF in on-chip RAM. Do not allocate the RAM-FIFO at the external address space.

## 8.5 RAM-FIFO Pointer

The RAM-FIFO specifies the start address by BAR, and the size by the BUD2 to BUD0 bits in DTCRA. BAR and the BUD2 to BUD0 bits in DTCRA should be set so that the RAM-FIFO areas of all pointer sets are stored in the on-chip RAM area, and the RAM-FIFO areas do not overlap inter-pointer sets.

The RAM-FIFO can be accessed when the value of the sum of the contents of BAR and either the contents of RAR, WAR, TMP, is output to the address bus.

## 8.6 RAM-FIFO Manipulation and RFU Bus Cycles

Table 8.4 summarizes the requests from the peripheral modules to the RFU and manipulations of the RFU bus cycle and pointer.

The RFU returns an acknowledge signal for a request from the peripheral modules to clear the request. All RFU bus cycles are executed in two states. In the RFU bus cycle, data transfer is executed or the error status is notified in addition to clearing a request and the RFU pointer is manipulated simultaneously. In data transfer in the RFU bus cycle, the RAM address is output to the address bus, the ID of the peripheral module is specified, and data transfer from RAM to the peripheral module or from peripheral module to RAM is executed in one bus cycle, simultaneously. An RFU bus cycle other than the read/write cycle is two states.

There are four types of requests from the peripheral modules: Data transfer, pointer mark (RAR/WAR → TMP), pointer reload (TMP → RAR/WAR), and pointer reset (0 → RAR, WAR, TMP).

When data transfer is executed, either RAR or WAR is added for the number of transfer bytes, according to the settings of the transfer direction and the byte/word transfer. As one TMP is provided in a pointer set, TMP is used as either the read temporary pointer or write temporary pointer.

If the added contents exceed the FIFO size specified by bits BUD2 to BUD0 in DTCRA, the pointer is set to a value from which the FIFO size is decremented (remainder of the FIFO size).

All peripheral modules perform a handshake following data transfer to update the pointer, depending on approval/refusal of the handshake. In this case, TMP is used during data transfer until a handshake. When the handshake is approved, the contents of RAR/WAR are regarded as the formal contents of the pointer, and are sent to TMP (mark operation). When the handshake is refused, the contents of RAR/WAR are restored to the previous contents, so the contents of TMP are sent to RAR or WAR (reload operation).

In the pointer reset operation, RAR, WAR, and TMP are all cleared to 0.

**Table 8.3 RFU Bus Cycle Types**

<b>Peripheral Module Request</b>	<b>USB</b>	<b>SCI</b>	<b>MCIF</b>
Data transfer	O	O	O
Pointer mark	O	—	—
Pointer reload	O	—	—
Reset	O	—	—

**Table 8.4 Requests from Peripheral Modules and RFU Bus Cycle**

<b>Requests from Peripheral Modules</b>	<b>Transfer Condition and FIFO Pointer Status</b>	<b>RFU Bus Cycle Contents</b>	<b>Pointer Manipulation</b>	
Data transfer	RAM → peripheral modules	Other than the following	RAM read, peripheral module write cycle	Adds RAR
	TMP is not used	RAR = WAR after RAR addition	RAM read, peripheral module write cycle Notification of FIFO empty state	Adds RAR
		RAR = WAR	Notification of FIFO overread state	No pointer manipulations
		RAM → peripheral modules	Other than the following	RAM read, peripheral module write cycle
	TMP is used as a read temporary pointer	RAR = WAR after RAR addition	RAM read, peripheral module write cycle Notification of FIFO empty state	Adds RAR
		RAR = WAR	Notification of FIFO overread state	No pointer manipulations
Peripheral modules → RAM		Other than the following	RAM read, peripheral module write cycle	Adds WAR
TMP is not used	WAR = RAR after WAR addition	Peripheral module read, RAM write cycle Notification of FIFO full state	Adds WAR	
	WAR = RAR	Notification of FIFO overwrite state	No pointer manipulations	
	Peripheral modules → RAM	Other than the following	RAM read, peripheral module write cycle	Adds WAR
TMP is used as a write temporary pointer	WAR = RAR after WAR addition	Peripheral module read, RAM write cycle Notification of FIFO full state	Adds WAR	
	WAR = RAR	Notification of FIFO overwrite state	No pointer manipulations	

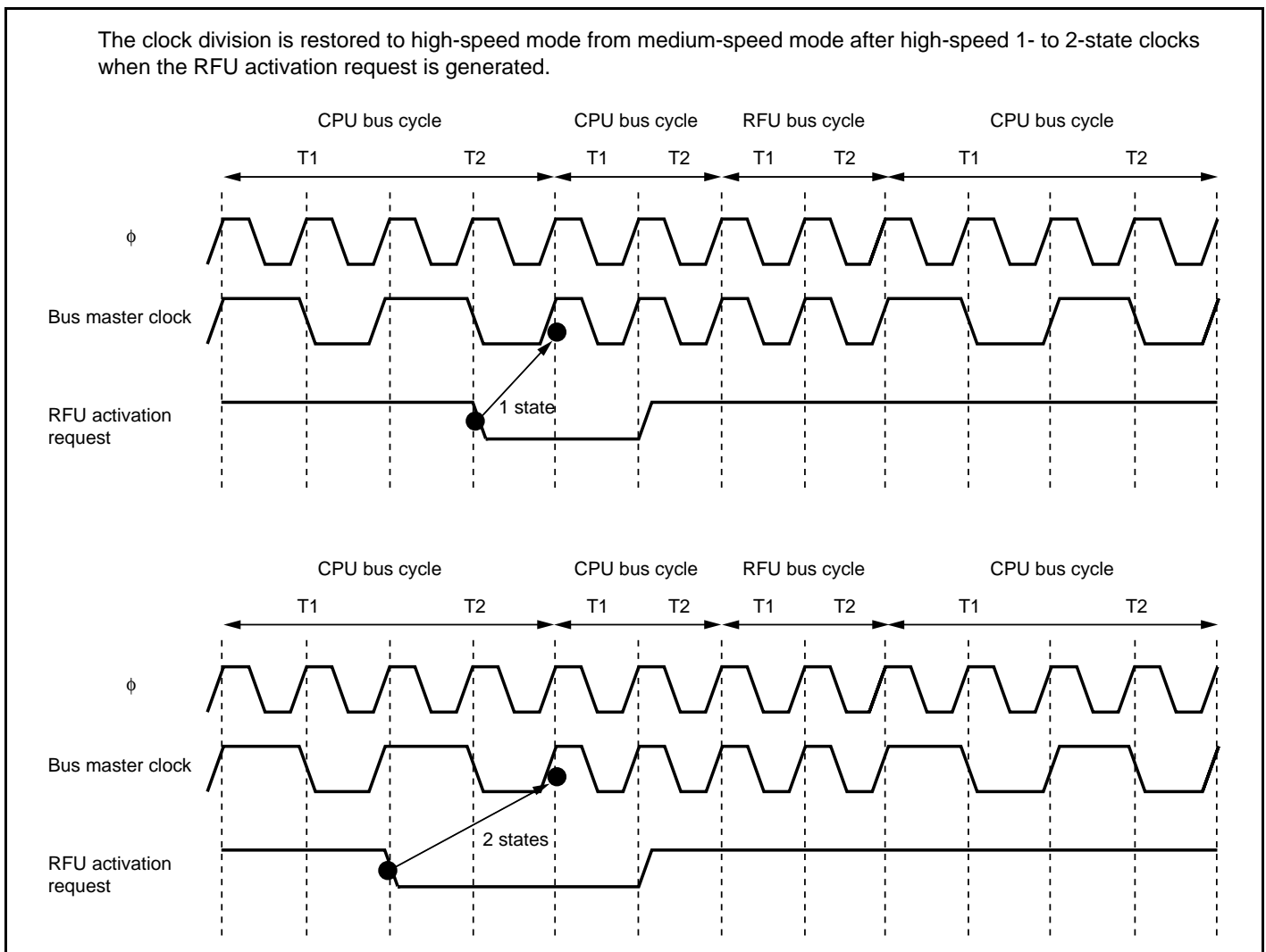
<b>Requests from Peripheral Modules</b>	<b>Transfer Condition and FIFO Pointer Status</b>	<b>RFU Bus Cycle Contents</b>	<b>Pointer Manipulation</b>
Pointer mark	TMP is not used	Acknowledge only	No pointer manipulations
	TMP is used as a read temporary pointer	Acknowledge only	RAR → TMP
	TMP is used as a write temporary pointer	Acknowledge only	WAR → TMP
Pointer reload	TMP is not used	Acknowledge only	No pointer manipulations
	TMP is used as a read temporary pointer	Acknowledge only	TMP → RAR
	TMP is used as a write temporary pointer	Acknowledge only	TMP → WAR
Pointer reset	—	Acknowledge only	No pointer manipulations

## 8.7 RFU Bus Cycle

### 8.7.1 Clock Division

As this LSI supports medium-speed mode, current consumption can be reduced by dividing the operating clock of the bus master. On the other hand, high-speed response may be requested of the RFU, which is one of the bus masters. In particular, if the RFU is used as a slave of a host interface, such as the USB, transfer data should be supplied with a sufficient transfer rate.

The RFU does not support medium-speed mode. The clock division in the medium-speed mode should be temporarily suspended to switch the clock to high-speed mode by setting the DTSPEED bit in SBYCR to 1 during DTC and RFU operations (and during CPU operation when transfer request is generated).



**Figure 8.2 Examples of Temporary Cancellation of Medium-Speed Mode**



### 8.7.2 RFU Bus Cycle Insertion

The RFU bus cycle can be inserted at a break in the bus cycle under almost all conditions.

Table 8.5 summarizes a comparison of enabling/disabling bus cycle insertion for the DTC and RFU. The RFU bus cycle can be inserted with the same condition as the DTC when the BUSDIVE bit in BCR2 is cleared to 0.

**Table 8.5 Bus Cycle Insertion**

<b>Bus Cycle Condition</b>	<b>RFU</b>	<b>DTC</b>
CPU internal operation cycle	Enabled	Enabled
The end of CPU bus cycle	Enabled	Enabled
The end of upper-byte CPU bus cycle for divided word access	Disabled	Disabled
The end of upper-word CPU bus cycle for divided long word access	Enabled	Disabled
The end of each CPU bus cycle for EEPMOV instruction	Enabled	Disabled
The end of each CPU bus cycle for STM/LDM instruction	Enabled	Disabled
The end of each CPU bus cycle for bit manipulation instruction	Enabled	Disabled
The end of CPU bus cycle for CCR manipulation instruction	Enabled	Disabled
The end of each CPU bus cycle for stacking/unstacking processing	Enabled	Disabled
The end of each CPU bus cycle for vector address read	Enabled	Disabled
The end of each DTC bus cycle for vector address read	Disabled	—
The end of accessing DTC control register (32 bits × 3)	Enabled	—
The end of the first and second long word bus cycles for accessing DTC control register (32 bits × 3)	Disabled	—
The end of DTC bus cycle	Enabled	—
The end of upper-byte DTC bus cycle for divided word access	Disabled	—

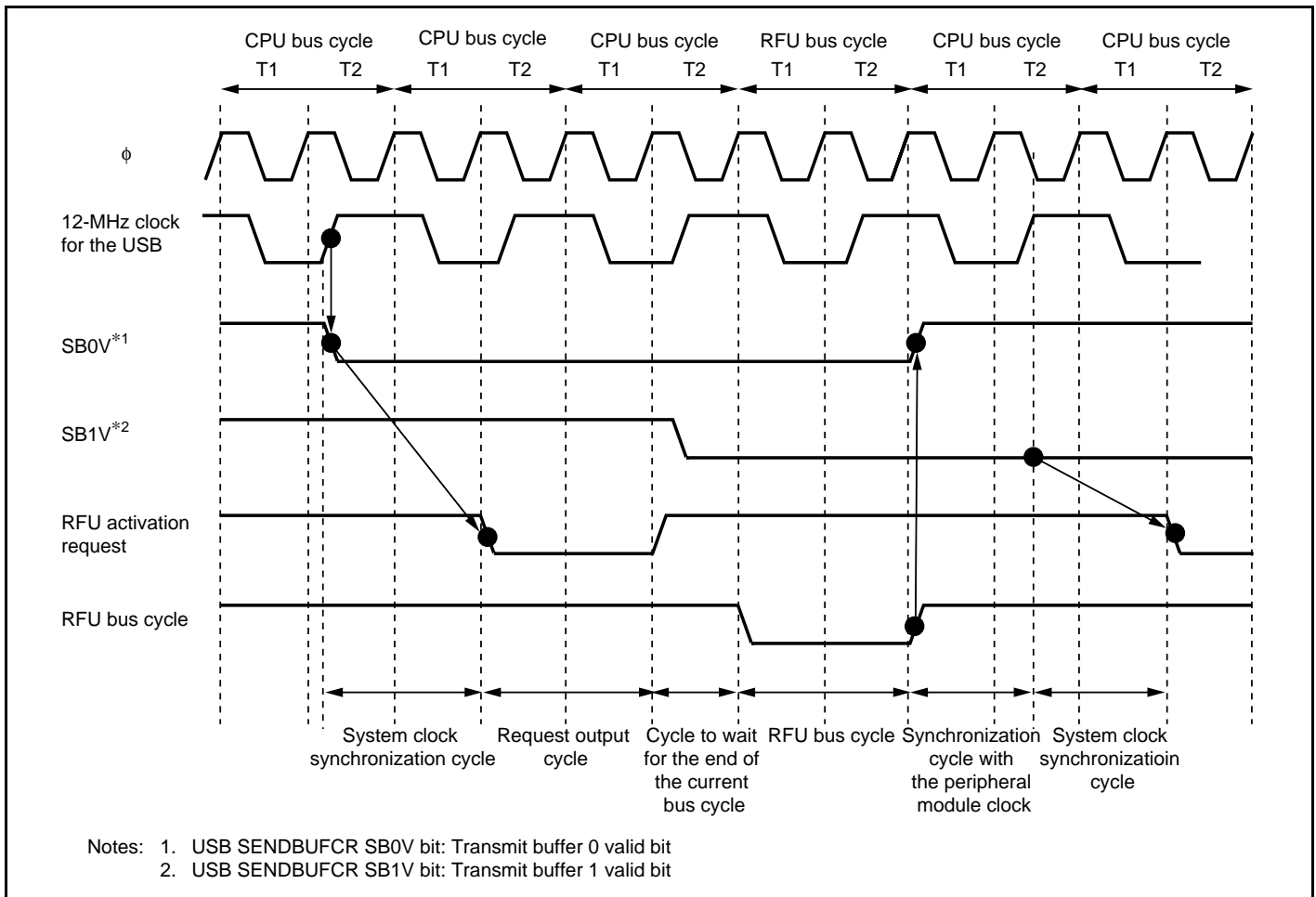
### 8.7.3 RFU Response Time

Figure 8.3 shows an example of the RFU response time. The RFU response time consists of the following:

- Cycle to synchronize peripheral module signals with the system clock: 1 and 2 states
- Cycle to output a request from the peripheral modules: 1 to 5 states
- Cycle to wait for the end of the current bus cycle: 0 to 2 states (except for wait states for external extension)

- RFU bus cycle: 2 states
- Cycle to synchronize the RFU signal with the peripheral module clock: 1 and 2 states (depending on the peripheral module clock)

The total of the above cycles is 5 to 13 states. Cycles for synchronization of the peripheral module clock with the system clock are needed even when the hardware FIFO is used instead of the RFU. Therefore, RAM-FIFO overhead by the RFU is 3 to 10 states (except for wait states for external extension). To reduce the RFU response time, it is recommended to set the external extension area access to 3 states/no waits.



**Figure 8.3 Example of RFU Response Time**

## 8.8 Operation

The RFU is presupposed to operate with the following procedure:

- Enable ID to be written to the RFU, and store data received by the peripheral module into FIFO.
- Enable ID to be read from the RAM-FIFO, and supply data transmitted by the peripheral module from FIFO.

Two IDs can be enabled simultaneously. However, when OVER-READ or OVER-WRITE error status occurs, sufficient error processing may not be made. Thus, the processing should be made with a sufficient number of valid data bytes and free area bytes.

When the CPU reads and uses the received data, the read start address of the pointer status and the number of valid data bytes are used. When the CPU generates data to be transmitted, data is written to on-chip RAM, and then the RAM address where data is stored in the pointer is written.

### 8.8.1 Transmission/Reception of Single Data Block

To transmit a single data block or to receive a data block with a known number of bytes, boundary overflow is convenient. Even if the peripheral module does not include a function to generate an interrupt at the completion of the specified number of bytes of transfer data, the completion of data block transmission/reception can be acknowledged by an interrupt from the RFU side. This interrupt is generated when the RFU pointer reaches the boundary of the FIFO size specified by bits BUD2 to BUD0 in DTCRA.

Table 8.6 summarizes the settings when boundary overflow is used.

**Table 8.6 Settings when Using Boundary Overflow (Transmission/Reception of Single Data Block)**

Transfer Condition		RAM → Peripheral Modules	Peripheral Modules → RAM
Number of transfer data bytes	—	N_R	N_W
FIFO size	BUD2 to BUD0	Sz	Sz
Base address	BAR	Clear the bits lower than the boundary to 0 according to the FIFO size.	Clear the bits lower than the boundary to 0 according to the FIFO size.
Read pointer	RAR	Sz – N_R	Sz – N_W
Write pointer	WAR	0	Sz – N_W

When the number of transfer data bytes has been transmitted from RAM to the peripheral modules the read pointer becomes 0, and boundary overflow occurs. At this time, the BOVF\_R flag in DTSTRC is set to 1.

When the number of transfer data bytes has been received from the peripheral modules to RAM, the write pointer becomes 0, and boundary overflow occurs. At this time, the BOVF\_W flag in DTSTRC is set to 1.

### 8.8.2 Transmission/Reception of Consecutive Data Blocks

If the peripheral module includes a function to generate an interrupt request at the completion of the specified number of bytes of transfer data, data blocks can be processed consecutively according to the following procedure. An example in which the ID to be written to the RFU is enabled and receive data is processed by the CPU is shown below.

1. The pointer set is initialized.
2. ID of RFU write is enabled.
3. Data transfer of the corresponding peripheral module is initiated.
4. Data block receive end interrupt (peripheral module).
5. RAR and DATAN are read from, and receive data block processing is started by the CPU.

When FIFO has sufficient free area after starting the CPU processing at the 5th step, the next block transfer can be started, returning to the 3rd step.

At this time, the RFU is the FIFO-size ring buffer, specified by bits BUD2 to BUD0 in DTCRA. If the contents of the RFU pointer exceed the FIFO size, it automatically becomes the remainder of

the FIFO size. Programming should be such that the CPU access does not deviate from the FIFO area.

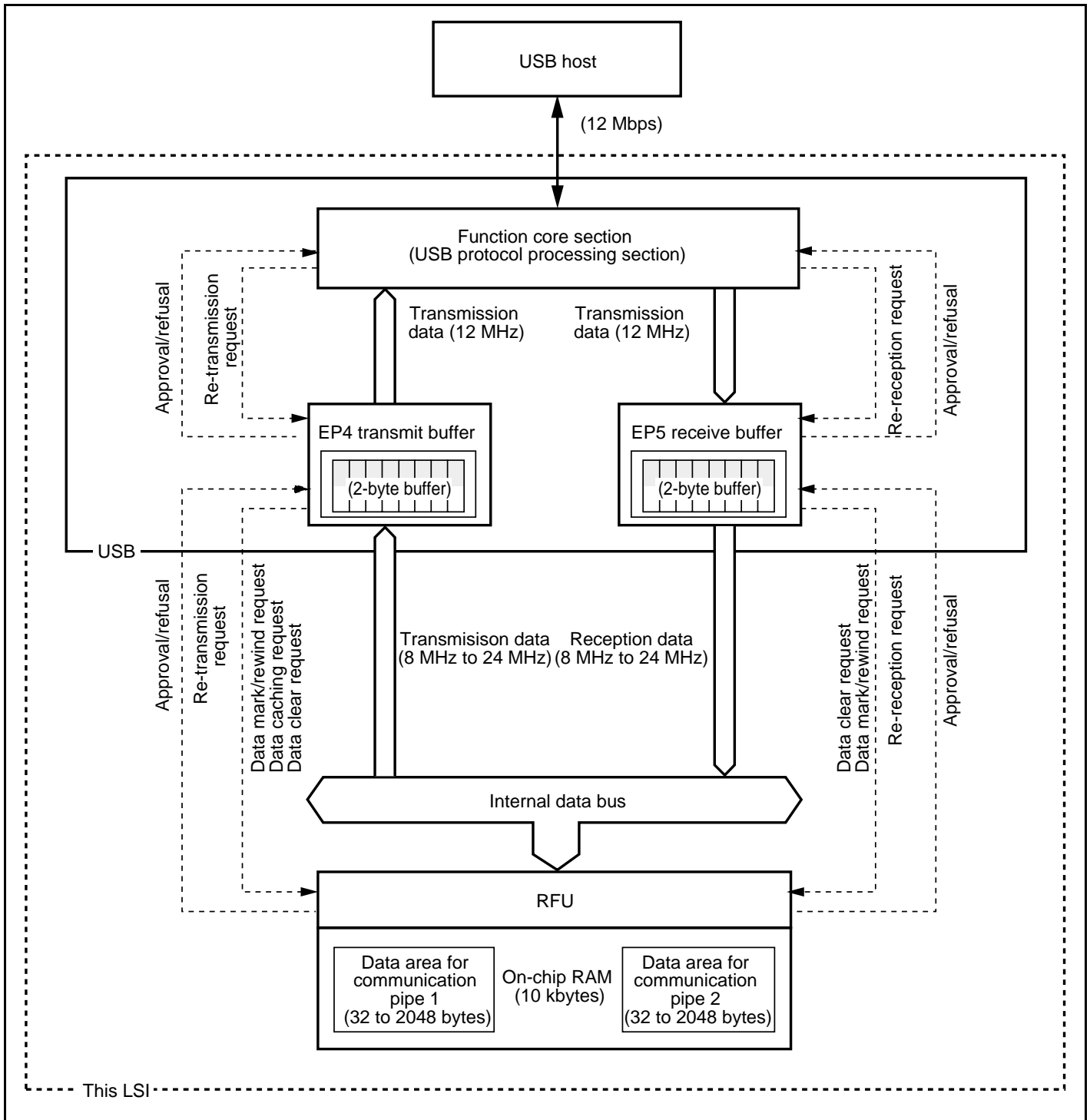
### 8.8.3 RFU Manipulation by USB

Figure 8.4 is a block diagram of the RFU interface in the USB.

The USB can use the RFU for data transfer with end point 4 (EP4) and end point 5 (EP5). The USB has a 2-byte transmit buffer in end point 4 dedicated for IN transfer. The USB also has a 2-byte receive buffer in end point 5 dedicated for OUT transfer.

Figure 8.5 shows the operational flow for IN transfer. EP4 is bulk IN transfer. When the transmit data is written to the FIFO, and start of transmission is triggered (the PTTE bit is set to 1), the USB issues a data transfer request to the RFU, and the 2-byte transmit buffer is filled and enters an output enable state. When the host issues an IN transfer request, the USB transmits data in the transmit buffer. The USB issues data transfer requests until data of MAX\_PACKET\_SIZE bytes is transferred to the transmit buffer, and operates such that the transmit buffer is always filled. When the transmission for MAX\_PACKET\_SIZE bytes is completed, the USB issues a mark/reload (rewind) request to the RFU according to the ACK/NACK handshake received from the host. If the FIFO underruns (OVER-R) during transmission, transmission ends correctly by regarding the data packet as a short packet. If the transmit buffer underruns, the USB transmits abnormal data to lead to the NACK handshake from the host.

Figure 8.6 shows the operational flow for OUT transfer. EP5 is bulk OUT transfer. The USB writes the received data to the receive buffer. When data is stored in the receive buffer, the USB issues the data transfer request, and operates such that the receive buffer is always empty. When the reception for MAX\_PACKET\_SIZE bytes completes, and all data in the receive buffer is transferred to the FIFO, the USB transmits the ACK handshake to the host, and requests a mark to the RFU. If an error is detected from the received data or all received data cannot be transferred to the FIFO, the USB transmits the NACK handshake to the host, and requests a reload (rewind) to the RFU. FIFO overrun (OVER-W) and receive buffer overrun can be regarded as the error status.



**Figure 8.4 RFU Interface of USB**

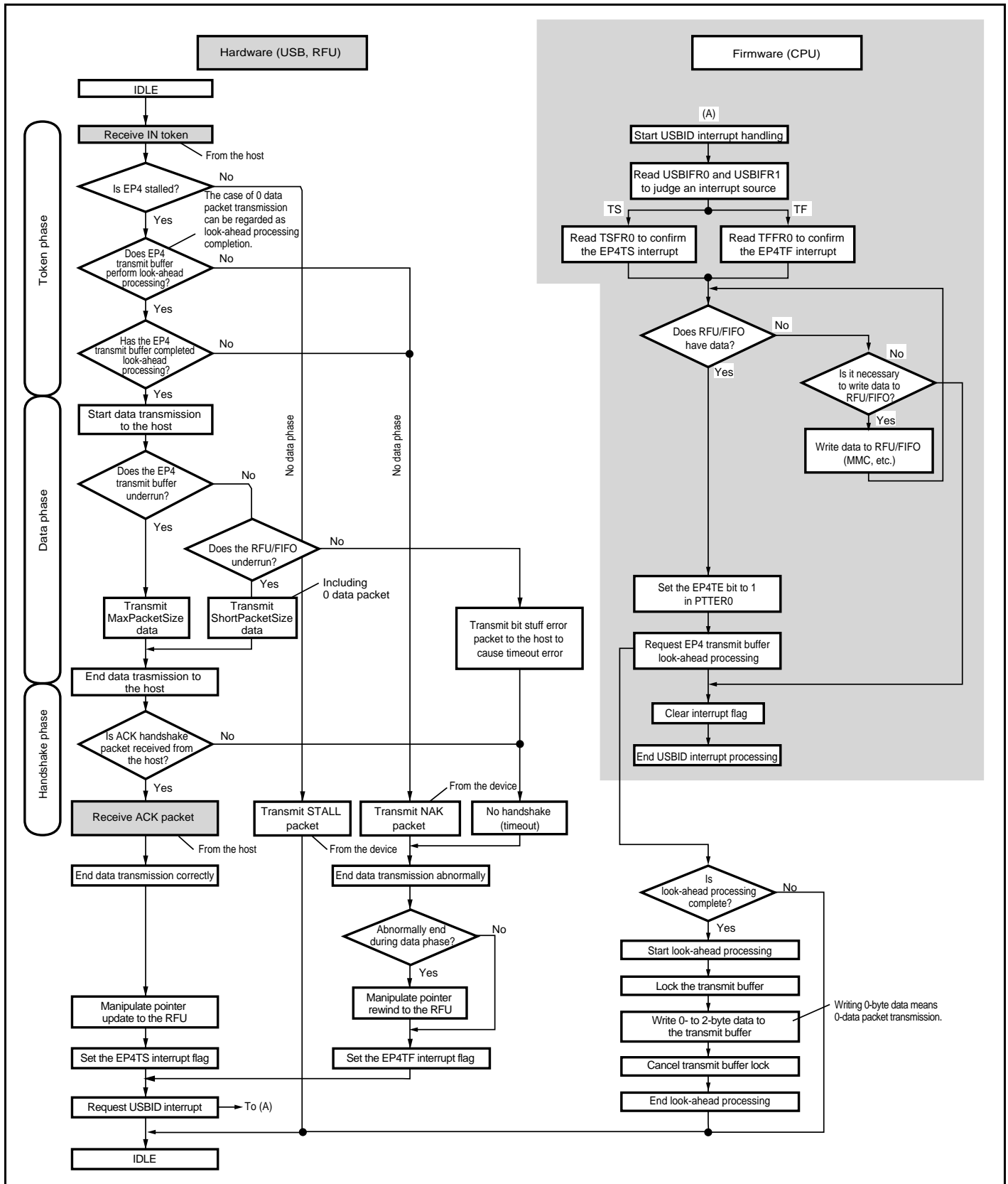


Figure 8.5 Operation Flow of USB IN Transfer

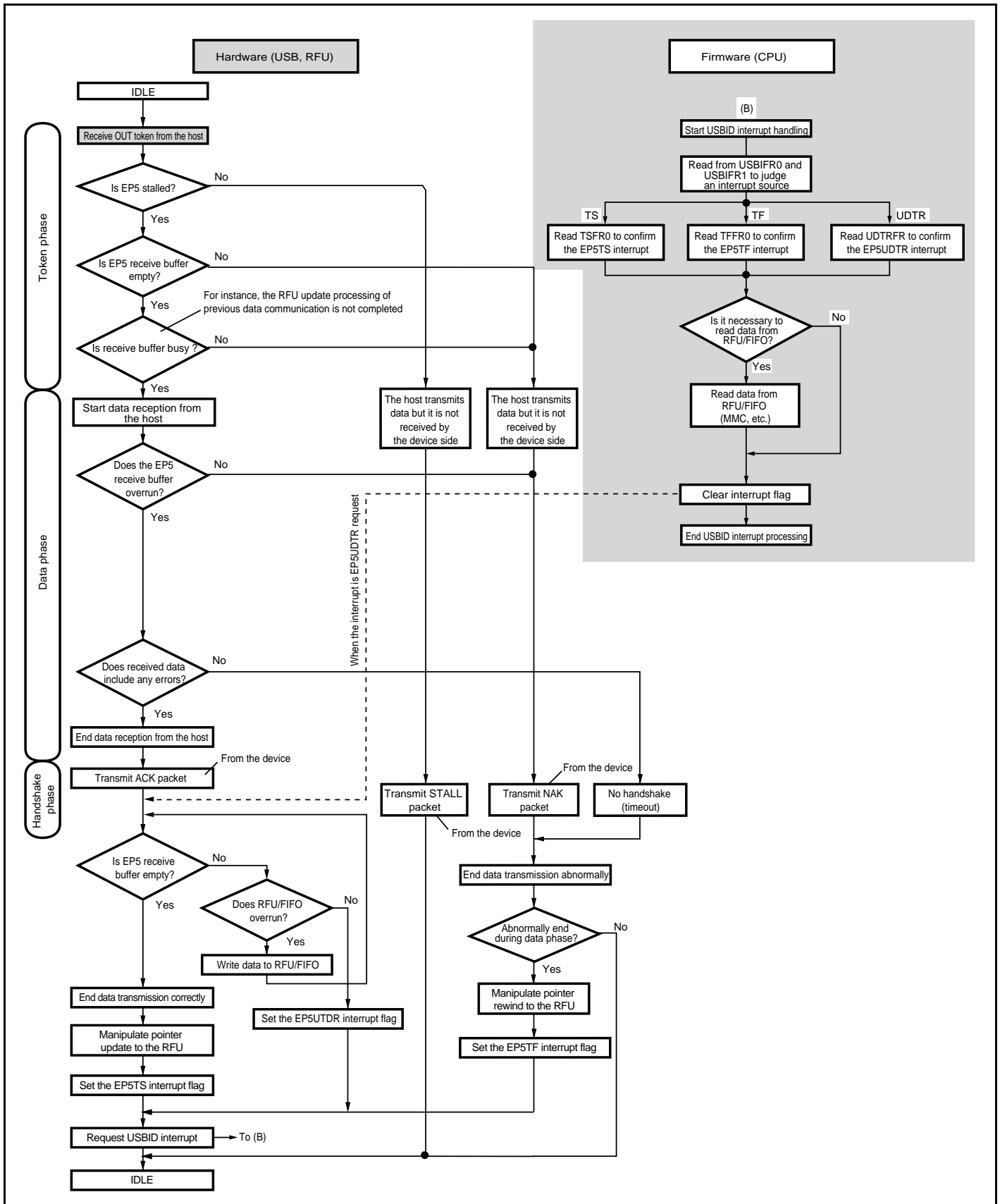


Figure 8.6 Operation Flow of USB OUT Transfer

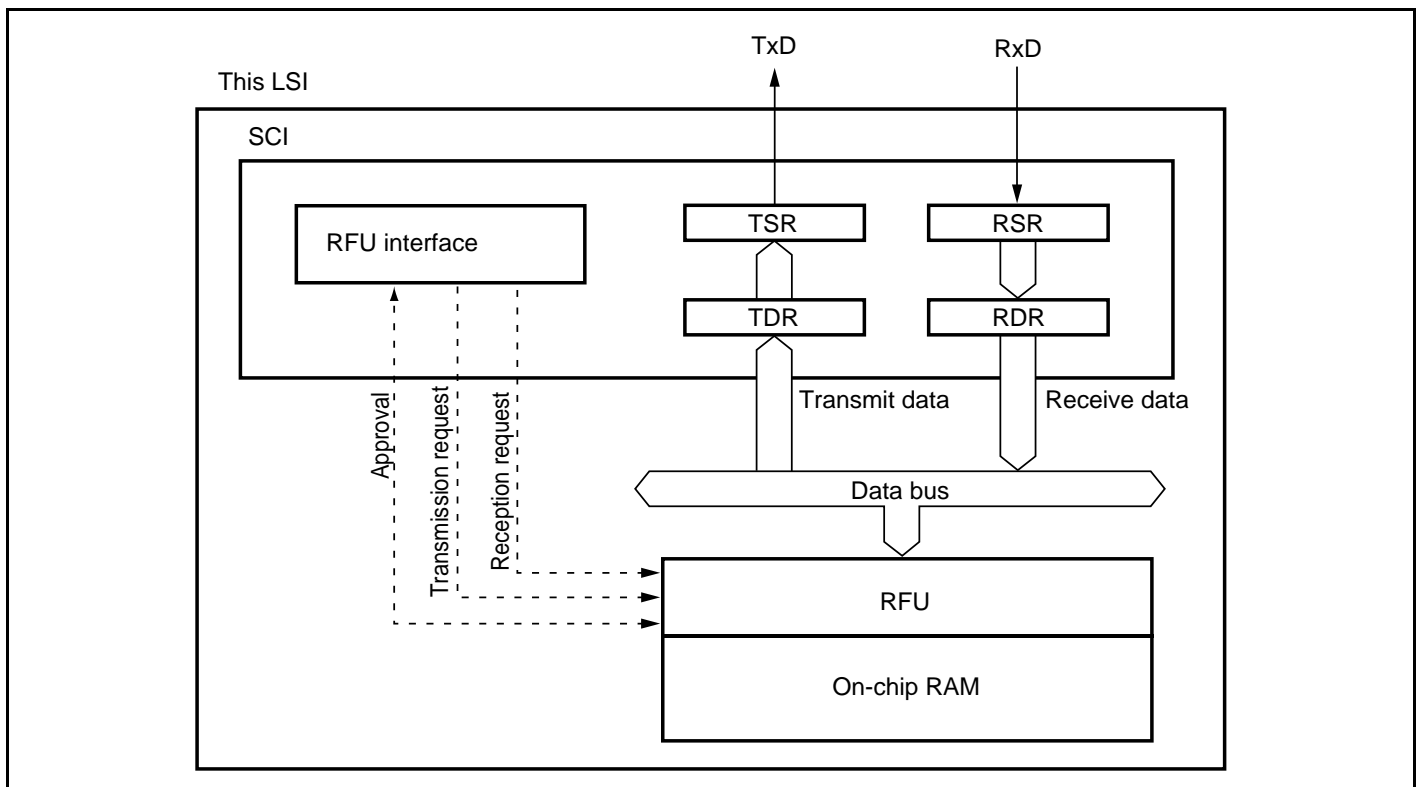


### 8.8.4 RFU Manipulation by SCI

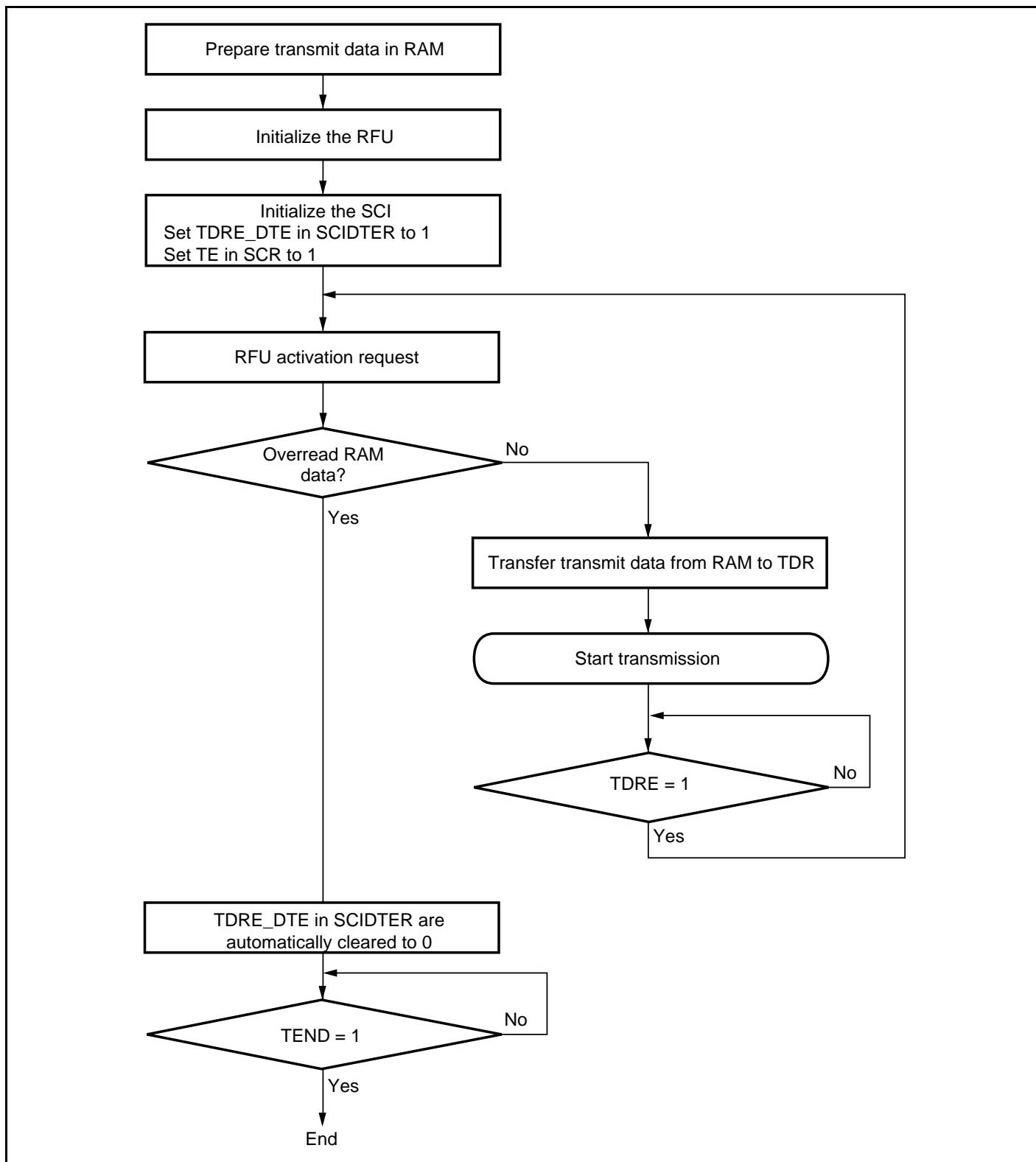
Figure 8.7 is a block diagram of the RFU interface in the SCI. The SCI can use the RFU for data transmission and reception. The RFU is activated by setting the TDRE flag and the RDRF flag.

Figure 8.8 shows the operational flow for transmission. When the transmitted data is written to the FIFO, and start of transmission is triggered (the TDRE and RDRF flags in SSR are set to 1), the SCI issues a data transfer request to the RFU.

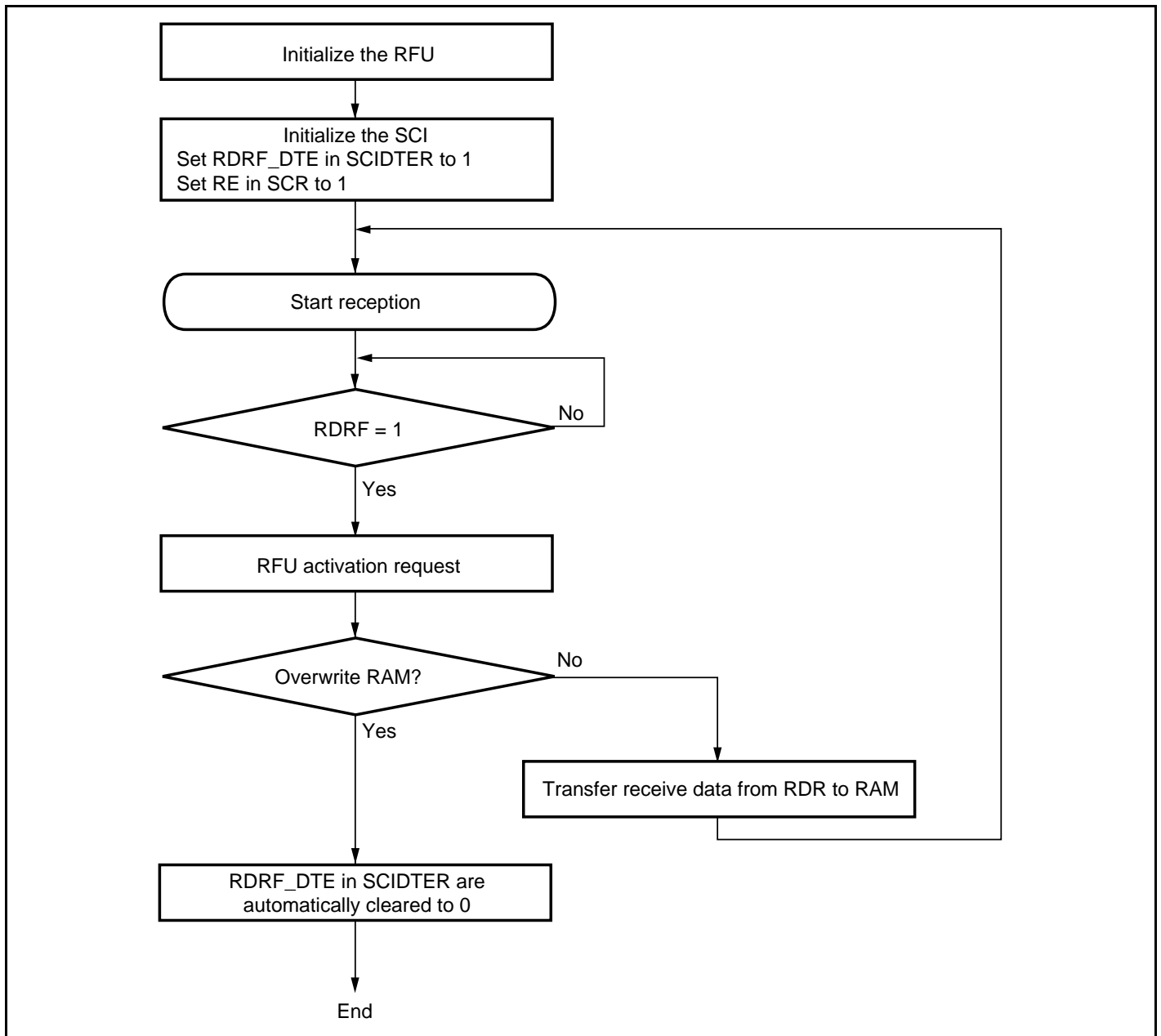
Figure 8.9 shows the operational flow for reception.



**Figure 8.7 RFU Interface of SCI**



**Figure 8.8 Operation Flow of SCI Transmission**

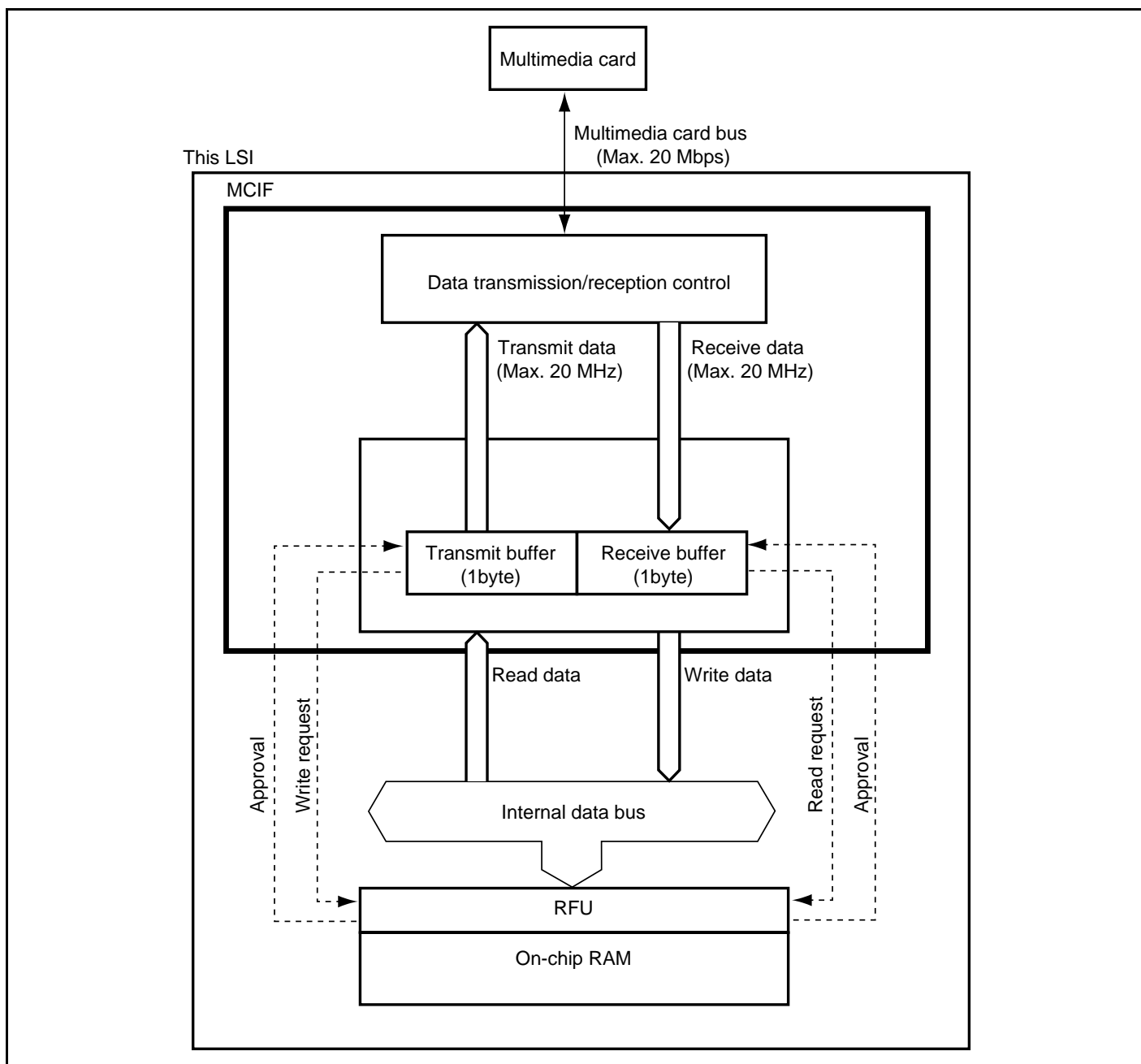
**Figure 8.9 Operation Flow of SCI Reception**

### 8.8.5 RFU Manipulation by MCIF

Figure 8.10 is a block diagram of the RFU interface in the MCIF. The MCIF can use the RFU for data transmission and reception.

Figure 8.11 shows the operational flow for transmission. When the transmitted data is written to the RFU, and start of transmission is triggered (the DATAEN bit in OPCR is set), the MCIF issues a data transfer request to the RFU.

Figure 8.12 shows the operational flow for reception.



**Figure 8.10 RFU Interface of MCIF**

When the RFU is emptied during data transmission, the transmission resume trigger (the DATAEN bit in OPCR is set) is set once the RFU empty is cancelled (after the necessary data is written), and data transmission is resumed.

When data reception is started, data is automatically written to the RFU. When the RFU is filled during data reception, the reception resume trigger (the RD\_CONTI bit in OPCR is set) is set once the RFU full is cancelled (after the necessary data is read from the RFU), and data reception is resumed.

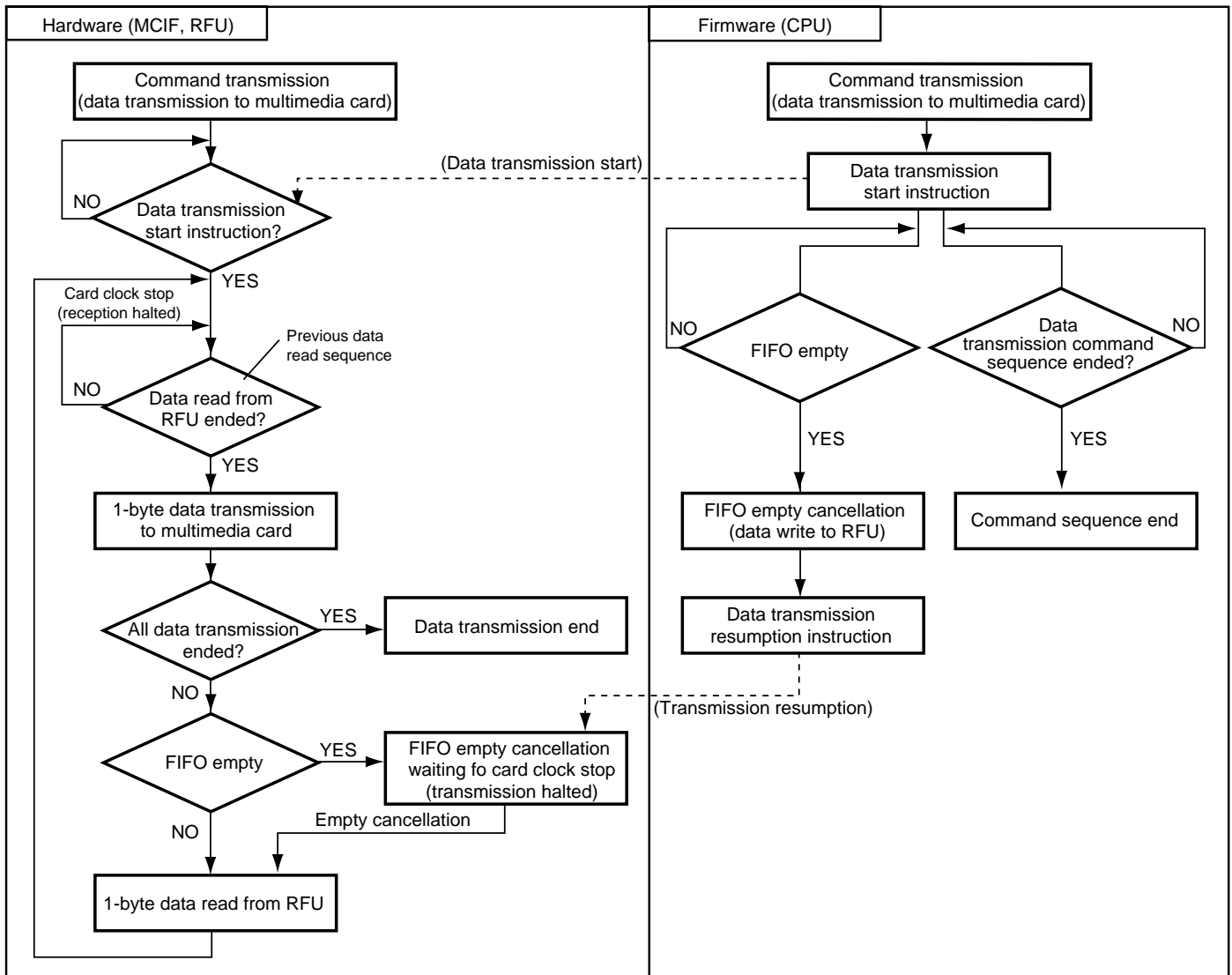


Figure 8.11 Operation Flow of MCIF Transmission

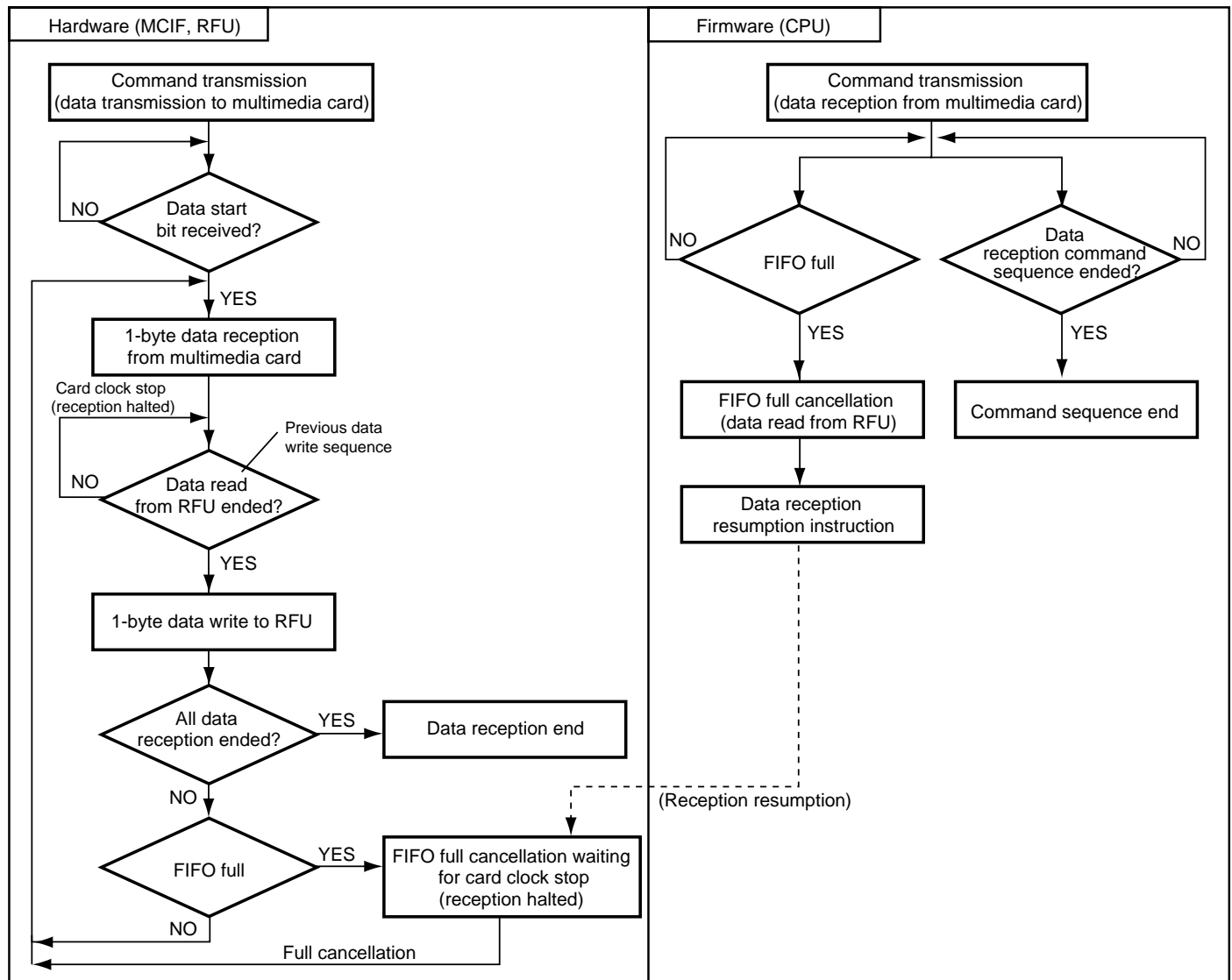


Figure 8.12 Operation Flow of MCIF Reception

## 8.9 Interrupt Sources

The RFU supports six interrupts: Boundary overflow (at reading), boundary overflow (at writing), FIFO full, FIFO empty, FIFO overread, and FIFO overwrite.

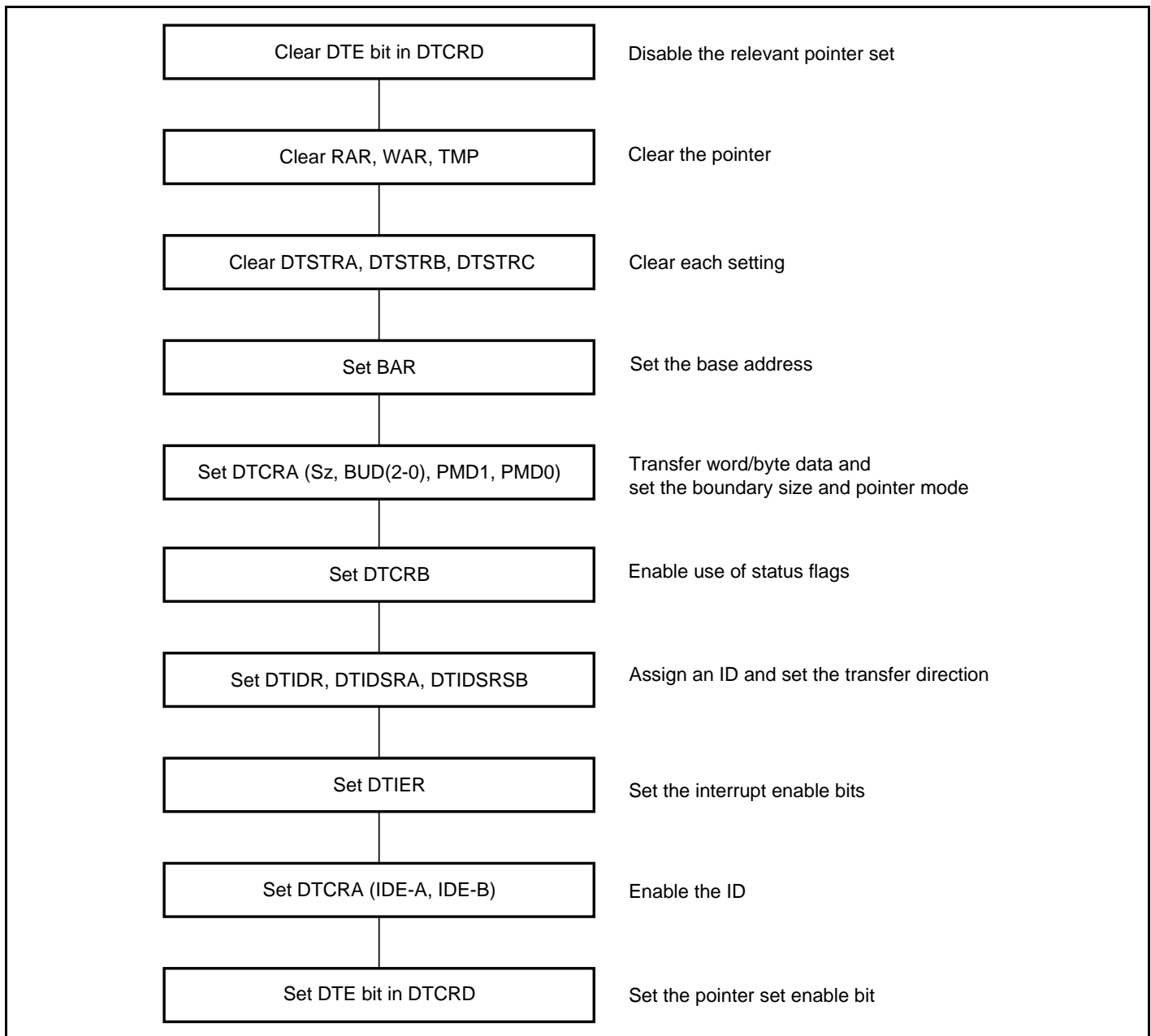
Identical interrupt vector addresses are assigned to boundary overflow (at reading), boundary overflow (at writing), FIFO full, and FIFO empty.

When an interrupt is generated, the interrupt source can be judged by reading the interrupt flag in DTSTRC.

In addition, FIFO overread and FIFO overwrite have common interrupt vector addresses. When an interrupt is generated, the interrupt source can be judged by reading the interrupt flag in DTSTRC.

## 8.10 RFU Initialization

Figure 8.13 shows the initialization flow of the RFU.



**Figure 8.13 RFU Initialization Flow**

1. The initial state of the FIFO is the FIFO empty state. When a single data block is transferred from RAM to the peripheral module, first initialize the RFU and then clear the FIFO empty state by writing 1 to the STCLR bit in DTCRB.
2. If the DTE bit is cleared to 0, the FIFO full state is automatically canceled and then the FIFO empty state is entered.
3. In medium-speed mode, the DTSPEED bit in SBYCR should be set to 1.

## 8.11 Usage Notes

### 1. Conflict between CPU write to DTRSR and RFU activation request

If conflict occurs between a CPU write to the RS bit in DTRSR while it is 0 and an RFU activation request, data transfer by the RFU cannot be performed correctly.

In order to access the FIFO state, the RS bit must be set to 1 at RFU initialization in advance.

When the RS bit is cleared to 0, the CPU should not write to DTRSR during RFU data processing.

### 2. FIFO full and FIFO empty states

When using a temporary pointer, the FIFO full and FIFO empty states during RFU data processing cannot be retained successfully, and overread or overwrite of data may occur.

If making use of a temporary pointer, be sure to check on the number of bytes of valid data and free area for the FIFO, and the EMPTY and FULL flag states before data processing.

### 3. DATAN/FREEN read value

If DATAN or FREEN is read in the FIFO full or FIFO empty state when the conditions listed in table 8.7 are satisfied, the read values may not be correct. When DATAN or FREEN is read as 0, whether the FIFO is full or empty needs to be checked by using the FULL and EMPTY bits in DTSTRC.

**Table 8.7 DATAN/FREEN Read Value**

Pointer Mode	Condition	DATAN/FREEN Read Value
Write temporary pointer	Write until the FIFO is full and request a mark (when TMP = RAR)	DATAN = 0 even though the FIFO is full
	Read until the FIFO is empty (when WAR = RAR)	FREEN = 0 even though the FIFO is empty
Read temporary pointer	Write until the FIFO is full (when WAR = RAR)	DATAN = 0 even though the FIFO is full
	Read until the FIFO is empty and request a mark (when TMP = WAR)	FREEN = 0 even though the FIFO is empty
Temporary pointer not used	Write until the FIFO is full	DATAN = 0 even though the FIFO is full
	Read until the FIFO is empty	FREEN = 0 even though the FIFO is empty

### 4. RFU operation in medium-speed mode

This LSI does not support medium-speed mode operation of the RFU. When using the RFU in medium-speed mode, the DTSPEED bit in SBYCR must be set to 1. If the RFU is activated with the DTSPEED bit cleared to 0, the program may get out of control.



## Section 9 I/O Ports

Table 9.1 is a summary of the port functions. The pins of each port also function as input/output pins of peripheral modules and interrupt input pins. Each input/output port includes a data direction register (DDR) that controls input/output and a data register (DR) that stores output data. DDR and DR are not provided for an input-only port.

Ports 1 to 3, 6, and A have built-in input pull-up MOSs. For port A, the on/off status of the input pull-up MOS is controlled by DDR and ODR. Ports 1 to 3 and 6 have an input pull-up MOS control register (PCR), in addition to DDR and DR, to control the on/off status of the input pull-up MOSs.

Ports 1 to 6, 9, and A can drive a single TTL load and 30 pF capacitive load. All the I/O ports can drive a Darlington transistor when in output mode. Port 8 is an NMOS push-pull output.

**Table 9.1 Port Functions**

Port	Description	Extended Mode	Single-Chip Mode	I/O Status
		Mode 2, Mode 3 (EXPE = 1)	Mode 2, Mode 3 (EXPE = 0)	
Port 1	General I/O port also functioning as PWM output, CompactFlash address output, and address output	P17/A7/CPA7 P16/A6/CPA6 P15/A5/CPA5 P14/A4/CPA4 P13/A3/CPA3 P12/A2/CPA2 P11/A1/CPA1 P10/A0/CPA0	P17/PW7 P16/PW6 P15/PW5 P14/PW4 P13/PW3 P12/PW2 P11/PW1 P10/PW0	Built-in input pull-up MOSs LED drive capability (sink current 5 mA)
Port 2	General I/O port also functioning as PWM output, CompactFlash address output, and address output	P27/A15 P26/A14 P25/A13 P24/A12 P23/A11/CPREG P22/A10/CPA10 P21/A9/CPA9 P20/A8/CPA8	P27/PW15 P26/PW14 P25/PW13 P24/PW12 P23/PW11 P22/PW10 P21/PW9 P20/PW8	Built-in input pull-up MOSs LED drive capability (sink current 5 mA)

Port	Description	Extended Mode	Single-Chip Mode	I/O Status
		Mode 2, Mode 3 (EXPE = 1)	Mode 2, Mode 3 (EXPE = 0)	
Port 3	General I/O port also functioning as bidirectional data bus, CompactFlash bidirectional data bus, wake-up event input, and MCIF input/output	D15/CPD15	P37/WUE15	Built-in input pull-up MOSs LED drive capability (sink current 5 mA)
		D14/CPD14	P36/WUE14	
		D13/CPD13	P35/WUE13	
		D12/CPD12	P34/WUE12/MCCMDDIR/MCCSB	
		D11/CPD11	P33/WUE11/MCDATDIR/MCCSA	
		D10/CPD10	P32/WUE10/MCDAT/MCRxD	
		D9/CPD9	P31/WUE9/MCCMD/MCTxD	
		D8/CPD8	P30/WUE8/MCCLK	
Port 4	General I/O port also functioning as interrupt input, TMR_0, TMR_1, TMR_X, TMR_Y, timer connection input/output, and MCIF inputs/outputs	P47/IRQ7/TMOY P46/IRQ6/TMOX P45/IRQ5/TMIY P44/IRQ4/TMIX/ExMCCMDDIR/ExMCCSB P43/IRQ3/TMO1/ExMCDATDIR/ExMCCSA/HSYNCO P42/IRQ2/TMO0/ExMCDAT/ExMCRxD P41/IRQ1/TMI1/ExMCCMD/ExMCTxD/HSYNCI P40/IRQ0/TMI0/ExMCCLK		
Port 5	General I/O port also functioning as interrupt input, PWMX output, and SCI_0, SCI_1, and SCI_2 inputs/outputs	P57/IRQ15/PWX1 P56/IRQ14/PWX0 P55/IRQ13/RxD2 P54/IRQ12/TxD2 P53/IRQ11/RxD1/IrRxD P52/IRQ10/TxD1/IrTxD P51/IRQ9/RxD0 P50/IRQ8/TxD0		

Port	Description	Extended Mode		Single-Chip Mode	I/O Status
		Mode 2, Mode 3 (EXPE = 1)		Mode 2, Mode 3 (EXPE = 0)	
Port 6	General I/O port also functioning as bidirectional data bus, CompactFlash bidirectional data bus, FRT input/output, enhanced A/D conversion input, keyboard input, timer connection input/output, and external USB driver/receiver input/output	P67/CIN7/ $\overline{\text{KIN7}}^{*1}$	D7/CPD7 <sup>*2</sup>	P67/CIN7/ $\overline{\text{KIN7}}$ /DPLS	Built-in input pull-up MOSs
		P66/FTOB/CIN6/ $\overline{\text{KIN6}}$ /CBLANK <sup>*1</sup>	D6/CPD6 <sup>*2</sup>	P66/FTOB/CIN6/ $\overline{\text{KIN6}}$ / CBLANK/DMNS	
		P65/FTID/CIN5/ $\overline{\text{KIN5}}$ /CSYNCl <sup>*1</sup>	D5/CPD5 <sup>*2</sup>	P65/FTID/CIN5/ $\overline{\text{KIN5}}$ / CSYNCl/XVERDATA	
		P64/FTIC/CIN4/ $\overline{\text{KIN4}}$ /CLAMPO <sup>*1</sup>	D4/CPD4 <sup>*2</sup>	P64/FTIC/CIN4/ $\overline{\text{KIN4}}$ / CLAMPO/TXDPLS	
		P63/FTIB/CIN3/ $\overline{\text{KIN3}}$ /VFBACKI <sup>*1</sup>	D3/CPD3 <sup>*2</sup>	P63/FTIB/CIN3/ $\overline{\text{KIN3}}$ / VFBACKI/TXDMNS	
		P62/FTIA/CIN2/ $\overline{\text{KIN2}}$ /VSYNCl <sup>*1</sup>	D2/CPD2 <sup>*2</sup>	P62/FTIA/CIN2/ $\overline{\text{KIN2}}$ / VSYNCl/TXENL	
		P61/FTOA/CIN1/ $\overline{\text{KIN1}}$ /VSYNCO <sup>*1</sup>	D1/CPD1 <sup>*2</sup>	P61/FTOA/CIN1/ $\overline{\text{KIN1}}$ / VSYNCO/SUSPEND	
		P60/FTCI/CIN0/ $\overline{\text{KIN0}}$ /HFBACKI <sup>*1</sup>	D0/CPD0 <sup>*2</sup>	P60/FTCI/CIN0/ $\overline{\text{KIN0}}$ / HFBACKI/SPEED	
Port 7	General input port also functioning as A/D converter analog input, D/A converter analog output, and interrupt input	P77/ $\overline{\text{ExIRQ7}}$ /AN7/DA1 P76/ $\overline{\text{ExIRQ6}}$ /AN6/DA0 P75/ $\overline{\text{ExIRQ5}}$ /AN5 P74/ $\overline{\text{ExIRQ4}}$ /AN4 P73/ $\overline{\text{ExIRQ3}}$ /AN3 P72/ $\overline{\text{ExIRQ2}}$ /AN2			

Port	Description	Extended Mode		Single-Chip Mode	I/O Status
		Mode 2, Mode 3 (EXPE = 1)		Mode 2, Mode 3 (EXPE = 0)	
Port 8	General I/O port also functioning as A/D converter external trigger input pin, SCI_0, SCI_1, and SCI_2 clock inputs/outputs, IIC_0 and IIC_1 inputs/outputs, ExTMR_0, ExTMR_1, ExTMR_X, and ExTMR_Y inputs, USB external clock input, and interrupt input	P87/ $\overline{\text{ExIRQ15}}$ /ADTRG/ExTMIY/USEXCL P86/ $\overline{\text{ExIRQ14}}$ /SCK2/ExTMIX P85/ $\overline{\text{ExIRQ13}}$ /SCK1/ExTMI1 P84/ $\overline{\text{ExIRQ12}}$ /SCK0/ExTMI0 P83/ $\overline{\text{ExIRQ11}}$ /SDA1 P82/ $\overline{\text{ExIRQ10}}$ /SCL1 P81/ $\overline{\text{ExIRQ9}}$ /SDA0 P80/ $\overline{\text{ExIRQ8}}$ /SCL0			P87 to P80 are NMOS push-pull outputs. SDA1, SCL1, SDA0, and SCL0 are NMOS open drain outputs.
Port 9	General I/O port also functioning as bus control input/output, CompactFlash control input/output, system clock output, and external sub-clock input	P97/ $\overline{\text{WAIT}}$ / $\overline{\text{CPWAIT}}$ / $\overline{\text{CS256}}$		P97	
		P96/ $\phi$ /EXCL			
		AS/ $\overline{\text{IOS}}$	P95		
		HWR/ $\overline{\text{CPWE}}$	P94		
		RD/ $\overline{\text{CPOE}}$	P93		
		P92/ $\overline{\text{CPCS1}}$	P92		
		P91/ $\overline{\text{CPCS2}}$	P91		
	P90/ $\overline{\text{LWR}}$	P90			
Port A	General I/O port also functioning as address output, keyboard input, and SCI_0 and SCI_2 external control pins	PA1/A17/ $\overline{\text{KIN9}}$ /SSE2I <sup>*3</sup> PA0/A16/ $\overline{\text{KIN8}}$ /SSE0I <sup>*3</sup>	PA1/ $\overline{\text{KIN9}}$ /SSE2I <sup>*4</sup> PA0/ $\overline{\text{KIN8}}$ /SSE0I <sup>*4</sup>	PA1/ $\overline{\text{KIN9}}$ /SSE2I PA0/ $\overline{\text{KIN8}}$ /SSE0I	Built-in input pull-up MOSs

- Notes:
1. 8-bit data bus is selected.
  2. 16-bit data bus is selected.
  3. Extended mode (mode 2)
  4. Extended mode (mode 3)

## 9.1 Port 1

Port 1 is an 8-bit I/O port. Port 1 pins also function as an address bus, PWM output pins, and CompactFlash address output pins. Port 1 functions change according to the operating mode. Port 1 has the following registers.

- Port 1 data direction register (P1DDR)
- Port 1 data register (P1DR)
- Port 1 pull-up MOS control register (P1PCR)

### 9.1.1 Port 1 Data Direction Register (P1DDR)

The individual bits of P1DDR specify input or output for the pins of port 1.

Bit	Bit Name	Initial Value	R/W	Description
7	P17DDR	0	W	In extended mode:
6	P16DDR	0	W	The corresponding port 1 pins are address output or CompactFlash address output ports when P1DDR bits are set to 1, and input ports when cleared to 0.
5	P15DDR	0	W	
4	P14DDR	0	W	
3	P13DDR	0	W	In single-chip mode:
2	P12DDR	0	W	The corresponding port 1 pins are output ports or PWM outputs when the P1DDR bits are set to 1, and input ports when cleared to 0.
1	P11DDR	0	W	
0	P10DDR	0	W	

### 9.1.2 Port 1 Data Register (P1DR)

P1DR stores output data for the port 1 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P17DR	0	R/W	P1DR stores output data for the port 1 pins that are used as the general output port. If a port 1 read is performed while the P1DDR bits are set to 1, the P1DR values are read. If a port 1 read is performed while the P1DDR bits are cleared to 0, the pin states are read.
6	P16DR	0	R/W	
5	P15DR	0	R/W	
4	P14DR	0	R/W	
3	P13DR	0	R/W	
2	P12DR	0	R/W	
1	P11DR	0	R/W	
0	P10DR	0	R/W	

### 9.1.3 Port 1 Pull-Up MOS Control Register (P1PCR)

P1PCR controls the port 1 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P17PCR	0	R/W	When the pins are in input state, the corresponding input pull-up MOS is turned on when a P1PCR bit is set to 1.
6	P16PCR	0	R/W	
5	P15PCR	0	R/W	
4	P14PCR	0	R/W	
3	P13PCR	0	R/W	
2	P12PCR	0	R/W	
1	P11PCR	0	R/W	
0	P10PCR	0	R/W	

### 9.1.4 Pin Functions

The relationship between register setting values and pin functions are as follows in each operating mode. In the tables, the symbol “—” stands for Don't care.

#### Extended Mode:

The function of port 1 pins is switched as shown below according to the P1nDDR bit.

P1nDDR	0	1
Pin function	P17 to P10 input pins	A7 to A0 output pins, CPA7 to CPA0 output pins

Note: n = 7 to 0

#### Single-Chip Mode:

The function of port 1 pins is switched as shown below according to the combination of the OEn bit in PWOERA of PWM and the P1nDDR bit.

P1nDDR	0		1	
OEn	—		0	1
Pin function	P17 to P10 input pins	P17 to P10 output pins	PW7 to PW0 output pins	

Note: n = 7 to 0

### 9.1.5 Port 1 Input Pull-Up MOS

Port 1 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used regardless of the operating mode. Table 9.2 summarizes the input pull-up MOS states.

**Table 9.2 Port 1 Input Pull-Up MOS States**

Reset	Hardware Standby Mode	Software Standby Mode	In Other Operations
Off	Off	On/Off	On/Off

Legend:

Off: Always off.

On/Off: On when P1DDR = 0 and P1PCR = 1; otherwise off.

## 9.2 Port 2

Port 2 is an 8-bit I/O port. Port 2 pins also function as an address bus, PWM output pins, and CompactFlash address output pins. Port 2 functions change according to the operating mode. Port 2 has the following registers.

- Port 2 data direction register (P2DDR)
- Port 2 data register (P2DR)
- Port 2 pull-up MOS control register (P2PCR)

### 9.2.1 Port 2 Data Direction Register (P2DDR)

The individual bits of P2DDR specify input or output for the pins of port 2.

Bit	Bit Name	Initial Value	R/W	Description
7	P27DDR	0	W	In extended mode:
6	P26DDR	0	W	The corresponding port 2 pins are address output or CompactFlash address output ports when the P2DDR bits are set to 1, and input ports when cleared to 0.
5	P25DDR	0	W	
4	P24DDR	0	W	
3	P23DDR	0	W	Pins function as the address output port depending on the setting of bits IOSE and CS256E in SYSCR.
2	P22DDR	0	W	
1	P21DDR	0	W	In single-chip mode: The corresponding port 2 pins are output ports or PWM outputs when the P2DDR bits are set to 1, and input ports when cleared to 0.
0	P20DDR	0	W	



### 9.2.2 Port 2 Data Register (P2DR)

P2DR stores output data for the port 2 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P27DR	0	R/W	P2DR stores output data for the port 2 pins that are used as the general output port. If a port 2 read is performed while the P2DDR bits are set to 1, the P2DR values are read. If a port 2 read is performed while the P2DDR bits are cleared to 0, the pin states are read.
6	P26DR	0	R/W	
5	P25DR	0	R/W	
4	P24DR	0	R/W	
3	P23DR	0	R/W	
2	P22DR	0	R/W	
1	P21DR	0	R/W	
0	P20DR	0	R/W	

### 9.2.3 Port 2 Pull-Up MOS Control Register (P2PCR)

P2PCR controls the port 2 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P27PCR	0	R/W	When the pins are in input state, the corresponding input pull-up MOS is turned on when a P2PCR bit is set to 1.
6	P26PCR	0	R/W	
5	P25PCR	0	R/W	
4	P24PCR	0	R/W	
3	P23PCR	0	R/W	
2	P22PCR	0	R/W	
1	P21PCR	0	R/W	
0	P20PCR	0	R/W	

## 9.2.4 Pin Functions

The relationship between register setting values and pin functions are as follows in each operating mode. In the tables, the symbol “—” stands for Don't care.

### Extended Mode:

The function of port 2 pins is switched as shown below according to the combination of the CS256E and IOSE bits in SYSCR, the ADFULLE and CPCSE bits in BCR2 of BSC, and the P2nDDR bit.

Addresses 13 and 11 in the following table are expressed by the following logical expressions:

$$\text{Address 13} = 1 : \overline{\text{ADFULLE}} \cdot \overline{\text{CS256E}} \cdot \text{IOSE}$$

$$\text{Address 11} = 1 : \overline{\text{ADFULLE}} \cdot \overline{\text{CS256E}} \cdot \overline{\text{CPCSE}} \cdot \text{IOSE}$$

P2nDDR	0	1	
Address 13	—	0	1
Pin function	P27 to P25 input pins	A15 to A13 output pins	P27 to P25 output pins

Notes: n = 7 to 5

Even if P2nDDR = 1 and CPCSE = 1, pins P27 to P25 can be used as output pins only when IOSE = 1.

P24DDR	0	1	
Address 11	—	0	1
Pin function	P24 input pin	A12 output pin	P24 output pin

P23DDR	0	1	
Address 11	—	0	1
Pin function	P23 input pin	A11 output pin, $\overline{\text{CPREG}}$ output pin	P23 output pin

P2nDDR	0	1
Pin function	P22 to P20 input pins	A10 to A8 output pins, CPA10 to CPA8 output pins

Note: n = 2 to 0

### Single-Chip Mode:

The function of port 2 pins is switched as shown below according to the combination of the OEm bit in PWOERB of PWM and the P2nDDR bit.

P2nDDR	0	1	
OEm	—	0	1
Pin function	P27 to P20 input pins	P27 to P20 input pins	PW15 to PW8 output pins

Notes: n = 7 to 0  
m = 15 to 8

#### 9.2.5 Port 2 Input Pull-Up MOS

Port 2 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used regardless of the operating mode. Table 9.3 summarizes the input pull-up MOS states.

**Table 9.3 Port 2 Input Pull-Up MOS States**

Reset	Hardware Standby Mode	Software Standby Mode	In Other Operations
Off	Off	On/Off	On/Off

Legend:

Off: Always off.

On/Off: On when P2DDR = 0 and P2PCR = 1; otherwise off.

### 9.3 Port 3

Port 3 is an 8-bit I/O port. Port 3 pins also function as a bidirectional data bus, CompactFlash bidirectional data bus, wake-up event input, and MCIF input/output pins. Port 3 functions change according to the operating mode. Port 3 has the following registers.

- Port 3 data direction register (P3DDR)
- Port 3 data register (P3DR)
- Port 3 pull-up MOS control register (P3PCR)

### 9.3.1 Port 3 Data Direction Register (P3DDR)

The individual bits of P3DDR specify input or output for the pins of port 3.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DDR	0	W	In extended mode:
6	P36DDR	0	W	The port functions as the data bus regardless of the values in these bits.
5	P35DDR	0	W	
4	P34DDR	0	W	In single-chip mode:
3	P33DDR	0	W	The corresponding port 3 pins are output ports when the P3DDR bits are set to 1, and input ports when cleared to 0.
2	P32DDR	0	W	
1	P31DDR	0	W	
0	P30DDR	0	W	

### 9.3.2 Port 3 Data Register (P3DR)

P3DR stores output data for the port 3 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DR	0	R/W	In extended mode:
6	P36DR	0	R/W	If a port 3 read is performed while the P3DDR bits are set to 1, the P3DR values are read. When the P3DDR bits are cleared to 0, 1 is read.
5	P35DR	0	R/W	
4	P34DR	0	R/W	In single-chip mode:
3	P33DR	0	R/W	P3DR stores output data for the port 3 pins that are used as the general output port.
2	P32DR	0	R/W	
1	P31DR	0	R/W	If a port 3 read is performed while the P3DDR bits are set to 1, the P3DR values are read. If a port 3 read is performed while the P3DDR bits are cleared to 0, the pin states are read.
0	P30DR	0	R/W	

### 9.3.3 Port 3 Pull-Up MOS Control Register (P3PCR)

P3PCR controls the port 3 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P37PCR	0	R/W	In extended mode:
6	P36PCR	0	R/W	Operation is not affected.
5	P35PCR	0	R/W	In single-chip mode:
4	P34PCR	0	R/W	When the pins are in input state, the corresponding input pull-up MOS is turned on when a P3PCR bit is set to 1.
3	P33PCR	0	R/W	
2	P32PCR	0	R/W	
1	P31PCR	0	R/W	
0	P30PCR	0	R/W	

### 9.3.4 Pin Functions

The relationship between register setting values and pin functions are as follows in each operating mode. Note that MMC mode stands for MultiMediaCard mode, and the symbol “—” stands for Don't care in the tables.

#### Extended Mode:

Port 3 pins automatically function as the data bus.

#### Single-Chip Mode:

- $P37/\overline{WUE15}$

The pin function is switched as shown below according to the P37DDR bit.

When the WUEM15 bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{WUE15}$  input pin. To use this pin as the  $\overline{WUE15}$  input pin, clear the P37DDR bit to 0.

P37DDR	0	1
Pin function	P37 input pin	P37 output pin
	$\overline{WUE15}$ input pin	

- P36/ $\overline{\text{WUE14}}$

The pin function is switched as shown below according to the P36DDR bit.

When the WUEM14 bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{WUE14}}$  input pin. To use this pin as the  $\overline{\text{WUE14}}$  input pin, clear the P36DDR bit to 0.

P36DDR	0	1
Pin function	P36 input pin	P36 output pin
	$\overline{\text{WUE14}}$ input pin	

- P35/ $\overline{\text{WUE13}}$

The pin function is switched as shown below according to the P35DDR bit.

When the WUEM13 bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{WUE13}}$  input pin. To use this pin as the  $\overline{\text{WUE13}}$  input pin, clear the P35DDR bit to 0.

P35DDR	0	1
Pin function	P35 input pin	P35 output pin
	$\overline{\text{WUE13}}$ input pin	

- P34/ $\overline{WUE12}$ /MCCMDDIR/MCCSB

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P34DDR bit.

When the WUEM12 bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{WUE12}$  input pin. To use this pin as the  $\overline{WUE12}$  input pin, clear the P34DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or MMCS in PTCNT0 is 1		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0	
	MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0 (MMC mode (SPI in MODER is 0) DIRME in IOMCR is 0) or (SPI mode (SPI in MODER is 1) SPCNUM in IOMCR is 0)		MMC mode (SPI in MODER is 0) DIRME in IOMCR is 1	SPI mode (SPI in MODER is 1) SPCNUM in IOMCR is 1
P34DDR	0	1	—	—
Pin function	P34 input pin	P34 output pin	MCCMDDIR output pin	MCCSB output pin
	$\overline{WUE12}$ input pin			

- P33/ $\overline{\text{WUE11}}$ /MCDATDIR/MCCSA

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P33DDR bit.

When the  $\overline{\text{WUEM11}}$  bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{WUE11}}$  input pin. To use this pin as the  $\overline{\text{WUE11}}$  input pin, clear the P33DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or MMCS in PTCNT0 is 1		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0	
	MCIF Enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0 MMC mode (SPI in MODER is 0) DIRME in IOMCR is 0		MMC mode (SPI in MODER is 0) DIRME in IOMCR is 1	SPI mode (SPI in MODER is 1)
P33DDR	0	1	—	—
Pin function	P33 input pin	P33 output pin	MCDATDIR output pin	MCCSA output pin
	$\overline{\text{WUE11}}$ input pin			

- P32/ $\overline{\text{WUE10}}$ /MCDAT/MCRxD

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P32DDR bit.

When the  $\overline{\text{WUEM10}}$  bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{WUE10}}$  input pin. To use this pin as the  $\overline{\text{WUE10}}$  input pin, clear the P32DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or MMCS in PTCNT0 is 1		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0	
			MMC mode (SPI in MODER is 0)	SPI mode (SPI in MODER is 1)
P32DDR	0	1	—	—
Pin function	P32 input pin	P32 output pin	MCDAT input/output pin	MCRxD input pin
	$\overline{\text{WUE10}}$ input pin			



- P31/ $\overline{\text{WUE9}}$ /MCCMD/MCTxD

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P31DDR bit.

When the WUEM9 bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{WUE9}}$  input pin. To use this pin as the  $\overline{\text{WUE9}}$  input pin, clear the P31DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or MMCS in PTCNT0 is 1		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0	
			MMC mode (SPI in MODER is 0)	SPI mode (SPI in MODER is 1)
P31DDR	0	1	—	—
Pin function	P31 input pin	P31 output pin	MCCMD input/output pin	MCTxD output pin
	$\overline{\text{WUE9}}$ input pin			

- P30/ $\overline{\text{WUE8}}$ /MCCLK

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P30DDR bit.

When the WUEM8 bit in WUEMR3 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{WUE8}}$  input pin. To use this pin as the  $\overline{\text{WUE8}}$  input pin, clear the P30DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or MMCS in PTCNT0 is 1		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 0
			—
P30DDR	0	1	—
Pin function	P30 input pin	P30 output pin	MCCLK output pin
	$\overline{\text{WUE8}}$ input pin		

### 9.3.5 Port 3 Input Pull-Up MOS

Port 3 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used in single-chip mode. Table 9.4 summarizes the input pull-up MOS states.

**Table 9.4 Port 3 Input Pull-Up MOS States**

Mode	Reset	Hardware Standby Mode	Software Standby Mode	In Other Operations
Extended mode (EXPE = 1)	Off	Off	Off	Off
Single-chip mode (EXPE = 0)	Off	Off	On/Off	On/Off

Legend:

Off: Always off.

On/Off: On when input state and P3PCR = 1; otherwise off.

## 9.4 Port 4

Port 4 is an 8-bit I/O port. Port 4 pins also function as interrupt input pins, TMR\_0, TMR\_1, TMR\_X, and TMR\_Y input/output pins, timer connection input/output pins, and MCIF input/output pins. Port 4 has the following registers.

- Port 4 data direction register (P4DDR)
- Port 4 data register (P4DR)

### 9.4.1 Port 4 Data Direction Register (P4DDR)

The individual bits of P4DDR specify input or output for the pins of port 4.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DDR	0	W	If port 4 pins are specified for use as the general I/O port, the corresponding port 4 pins are output ports when the P4DDR bits are set to 1, and input ports when cleared to 0.
6	P46DDR	0	W	
5	P45DDR	0	W	
4	P44DDR	0	W	
3	P43DDR	0	W	
2	P42DDR	0	W	
1	P41DDR	0	W	
0	P40DDR	0	W	

### 9.4.2 Port 4 Data Register (P4DR)

P4DR stores output data for the port 4 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DR	0	R/W	P4DR stores output data for the port 4 pins that are used as the general output port.
6	P46DR	0	R/W	
5	P45DR	0	R/W	If a port 4 read is performed while the P4DDR bits are set to 1, the P4DR values are read. If a port 4 read is performed while the P4DDR bits are cleared to 0, the pin states are read.
4	P44DR	0	R/W	
3	P43DR	0	R/W	
2	P42DR	0	R/W	
1	P41DR	0	R/W	
0	P40DR	0	R/W	

### 9.4.3 Pin Functions

The relationship between register setting values and pin functions are as follows. Note that MMC mode stands for MultiMediaCard mode, and the symbol “—” stands for Don't care in the tables.

- P47/ $\overline{\text{IRQ7}}$ /TMOY

The pin function is switched as shown below according to the combination of the OS3 to OS0 bits in TCSR of TMR\_Y and the P47DDR bit.

When the ISS7 bit in ISSR is cleared to 0 and the IRQ7E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ7}}$  input pin. To use this pin as the  $\overline{\text{IRQ7}}$  input pin, clear the P47DDR bit to 0.

OS3 to OS0	All bits are set as 0		One bit is set as 1
P47DDR	0	1	—
Pin function	P47 input pin	P47 output pin	TMOY output pin
	$\overline{\text{IRQ7}}$ input pin		

- P46/ $\overline{\text{IRQ6}}$ /TMOX

The pin function is switched as shown below according to the combination of the OS3 to OS0 bits in TCSR of TMR\_X and the P46DDR bit.

When the ISS6 bit in ISSR is cleared to 0 and the IRQ6E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ6}}$  input pin. To use this pin as the  $\overline{\text{IRQ6}}$  input pin, clear the P46DDR bit to 0.

OS3 to OS0	All bits are set as 0		One bit is set as 1
P46DDR	0	1	—
Pin function	P46 input pin	P46 output pin	TMOX output pin
	$\overline{\text{IRQ6}}$ input pin		

- P45/ $\overline{\text{IRQ5}}$ /TMIY

The pin function is switched as shown below according to the P45DDR bit.

When the TMIYS bit in PTCNT0 is cleared to 0 and the external clock is selected by the CKS2 to CKS0 bits in TCR of TMR\_Y, this pin is used as the TMCIY input pin. When the CCLR1 and CCLR0 bits in TCR of TMR\_Y are set to 1, this pin is used as the TMRIY input pin.

When the ISS5 bit in ISSR is cleared to 0 and the IRQ5E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ5}}$  input pin. To use this pin as the  $\overline{\text{IRQ5}}$  input pin, clear the P45DDR bit to 0.

P45DDR	0	1
Pin function	P45 input pin	P45 output pin
	TMIY (TMCIY/TMRIY) input pin/ $\overline{\text{IRQ5}}$ input pin	

- P44/ $\overline{\text{IRQ4}}$ /TMIX/ExMCCMDDIR/ExMCCSB

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P44DDR bit.

When the TMIXS bit in PTCNT0 is cleared to 0 and the external clock is selected by the CKS2 to CKS0 bits in TCR of TMR\_X, this pin is used as the TMCIX input pin. When the CCLR1 and CCLR0 bits in TCR of TMR\_X are set to 1, this pin is used as the TMRIX input pin.

When the ISS4 bit in ISSR is cleared to 0 and the IRQ4E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ4}}$  input pin. To use this pin as the  $\overline{\text{IRQ4}}$  input pin, clear the P44DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or (Single-chip mode (EXPE = 0) MMCS in PTCNT0 is 0)		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 or Extended mode (EXPE = 1)	
	MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 (MMC mode (SPI in MODER is 0) DIRME in IOMCR is 0) or (SPI mode (SPI in MODER is 1) SPCNUM in IOMCR is 0)		MMC mode (SPI in MODER is 0) DIRME in IOMCR is 1	SPI mode (SPI in MODER is 1) SPCNUM in IOMCR is 1
P44DDR	0	1	—	—
Pin function	P44 input pin	P44 output pin	ExMCCMDDIR output pin	ExMCCSB output pin
	TMIX (TMCIX/TMRIX) input pin/ $\overline{\text{IRQ4}}$ input pin			

- P43/ $\overline{\text{IRQ3}}$ /TMO1/ExMCDATDIR/ExMCCSA/HSYNCO

The pin function is switched as shown below according to the combination of the MCIF operating mode, the HOE bit in TCONRO of the timer connection, the OS3 to OS0 bits in TCSR of TMR\_1, and the P43DDR bit. When the ISS3 bit in ISSR is cleared to 0 and the IRQ3E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ3}}$  input pin. To use this pin as the  $\overline{\text{IRQ3}}$  input pin, clear the P43DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or (Single-chip mode (EXPE = 0) MMCS in PTCNT0 is 0)			MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 or Extended mode (EXPE = 1)		
	MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 MMC mode (SPI in MODER is 0) DIRME in IOMCR is 0)			MMC mode (SPI in MODER is 0) DIRME in IOMCR is 1	SPI mode (SPI in MODER is 1)	
HOE	0		1	—		
OS3 to OS0	All bits are set as 0		One bit is set as 1	—		
P43DDR	0	1	—	—		
Pin function	P43 input pin	P43 output pin	TMO1 output pin	HSYNC O output pin	ExMCDATDIR output pin	ExMCCSA output pin
	IRQ3 input pin					

- P42/ $\overline{\text{IRQ2}}$ /TMO0/ExMCDAT/ExMCRxD

The pin function is switched as shown below according to the combination of the MCIF operating mode, the OS3 to OS0 bits in TCSR of TMR\_1, and the P42DDR bit.

When the ISS2 bit in ISSR is cleared to 0 and the IRQ2E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ2}}$  input pin. To use this pin as the  $\overline{\text{IRQ2}}$  input pin, clear the P42DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or (Single-chip mode (EXPE = 0) MMCS in PTCNT0 is 0)			MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 or Extended mode (EXPE = 1)		
				MMC mode (SPI in MODER is 0)	SPI mode (SPI in MODER is 1)	
OS3 to OS0	All bits are set as 0		One bit is set as 1	—		
P42DDR	0	1	—	—		
Pin function	P42 input pin	P42 output pin	TMO0 output pin	ExMCDAT input/output pin	ExMCRxD input pin	
	IRQ2 input pin					

- P41/ $\overline{\text{IRQ1}}$ /TMI1/ExMCCMD/ExMCTxD/HSYNCl

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P41DDR bit.

When the TMI1S bit in PTCNT0 is cleared to 0 and the external clock is selected by the CKS2 to CKS0 bits in TCR of TMR\_1, this pin is used as the TMC11 input pin. When the CCLR1 and CCLR0 bits in TCR of TMR\_1 are set to 1, this pin is used as the TMRI1 input pin. When the ISS1 bit in ISSR is cleared to 0 and the IRQ1E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ1}}$  input pin. To use this pin as the  $\overline{\text{IRQ1}}$  input pin, clear the P41DDR bit to 0. When the SIMOD1 bit (IHI signal) in TCONRI of the timer connection module is set to 1, this pin is used as the HSYNCl input pin.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or (Single-chip mode (EXPE = 0) MMCS in PTCNT0 is 0)		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 or Extended mode (EXPE = 1)	
			MMC mode (SPI in MODER is 0)	SPI mode (SPI in MODER is 1)
P41DDR	0	1	—	
Pin function	P41 input pin	P41 output pin	ExMCCMD input/output pin	ExMCTxD output pin
	TMI1 (TMC11/TMRI1) input pin/ $\overline{\text{IRQ1}}$ input pin/HSYNCl input pin			

- P40/ $\overline{\text{IRQ0}}$ /TMI0/ExMCCLK

The pin function is switched as shown below according to the combination of the MCIF operating mode and the P40DDR bit.

When the TMI0S bit in PTCNT0 is cleared to 0 and the external clock is selected by the CKS2 to CKS0 bits in TCR of TMR\_0, this pin is used as the TMC10 input pin. When the CCLR1 and CCLR0 bits in TCR of TMR\_0 are set to 1, this pin is used as the TMRI0 input pin. When the ISS0 bit in ISSR is cleared to 0 and the IRQ0E bit in IER of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{IRQ0}}$  input pin. To use this pin as the  $\overline{\text{IRQ0}}$  input pin, clear the P40DDR bit to 0.

MCIF operating mode	MCIF disable (MMCPE in IOMCR is 0) or (Single-chip mode (EXPE = 0) MMCS in PTCNT0 is 0)		MCIF enable (MMCPE in IOMCR is 1) MMCS in PTCNT0 is 1 or Extended mode (EXPE = 1)
P40DDR	0	1	—
Pin function	P40 input pin	P40 output pin	ExMCCLK output pin
	TMI0 (TMCIO/TMRI0) input pin/ $\overline{\text{IRQ0}}$ input pin		

## 9.5 Port 5

Port 5 is an 8-bit I/O port. Port 5 pins also function as interrupt input pins, the PWMX output pin, SCI\_0, SCI\_1, and SCI\_2 input/output pins. Port 5 has the following registers.

- Port 5 data direction register (P5DDR)
- Port 5 data register (P5DR)

### 9.5.1 Port 5 Data Direction Register (P5DDR)

The individual bits of P5DDR specify input or output for the pins of port 5.

Bit	Bit Name	Initial Value	R/W	Description
7	P57DDR	0	W	If port 5 pins are specified for use as the general I/O port, the corresponding port 5 pins are output ports when the P5DDR bits are set to 1, and input ports when cleared to 0.
6	P56DDR	0	W	
5	P55DDR	0	W	
4	P54DDR	0	W	
3	P53DDR	0	W	
2	P52DDR	0	W	
1	P51DDR	0	W	
0	P50DDR	0	W	



### 9.5.2 Port 5 Data Register (P5DR)

P5DR stores output data for the port 5 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P57DR	0	R/W	P5DR stores output data for the port 5 pins that are used as the general output port. If a port 5 read is performed while the P5DDR bits are set to 1, the P5DR values are read. If a port 5 read is performed while the P5DDR bits are cleared to 0, the pin states are read.
6	P56DR	0	R/W	
5	P55DR	0	R/W	
4	P54DR	0	R/W	
3	P53DR	0	R/W	
2	P52DR	0	R/W	
1	P51DR	0	R/W	
0	P50DR	0	R/W	

### 9.5.3 Pin Functions

The relationship between register setting values and pin functions are as follows. In the tables, the symbol “—” stands for Don't care.

- P57/ $\overline{\text{IRQ15}}$ /PWX1

The pin function is switched as shown below according to the combination of the OEB bit in DACR of PWMX and the P57DDR bit.

When the IRQ15E bit in IER16 of the interrupt controller is set to 1 or the ISS15 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ15}}$  input pin. To use this pin as the  $\overline{\text{IRQ15}}$  input pin, clear the P57DDR bit to 0.

OEB	0		1
P57DDR	0	1	—
Pin function	P57 input pin	P57 output pin	PWX1 output pin
	$\overline{\text{IRQ15}}$ input pin		

- P56/ $\overline{\text{IRQ14}}$ /PWX0

The pin function is switched as shown below according to the combination of the OEA bit in DACR of PWMX and the P56DDR bit.

When the IRQ14E bit in IER16 of the interrupt controller is set to 1 or the ISS14 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ14}}$  input pin. To use this pin as the  $\overline{\text{IRQ14}}$  input pin, clear the P56DDR bit to 0.

OEA	0		1
P56DDR	0	1	—
Pin function	P56 input pin	P56 output pin	PWX0 output pin
	$\overline{\text{IRQ14}}$ input pin		

- P55/ $\overline{\text{IRQ13}}$ /RxD2

The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI\_2 and the P55DDR bit.

When the IRQ13E bit in IER16 of the interrupt controller is set to 1 or the ISS13 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ13}}$  input pin. To use this pin as the  $\overline{\text{IRQ13}}$  input pin, clear the P55DDR bit to 0.

RE	0		1
P55DDR	0	1	—
Pin function	P55 input pin	P55 output pin	RxD2 input pin
	$\overline{\text{IRQ13}}$ input pin		

- P54/ $\overline{\text{IRQ12}}$ /TxD2

The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI\_2 and the P54DDR bit.

When the IRQ12E bit in IER16 of the interrupt controller is set to 1 or the ISS12 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ12}}$  input pin. To use this pin as the  $\overline{\text{IRQ12}}$  input pin, clear the P54DDR bit to 0.

TE	0		1
P54DDR	0	1	—
Pin function	P54 input pin	P54 output pin	TxD2 output pin
	$\overline{\text{IRQ12}}$ input pin		

- P53/ $\overline{\text{IRQ11}}$ /RxD1/IrRxD

The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI\_1 and the P53DDR bit.

When the IRQ11E bit in IER16 of the interrupt controller is set to 1 or the ISS11 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ11}}$  input pin. To use this pin as the  $\overline{\text{IRQ11}}$  input pin, clear the P53DDR bit to 0.

RE	0		1
P53DDR	0	1	—
Pin function	P53 input pin	P53 output pin	RxD1/IrRxD input pin
	$\overline{\text{IRQ11}}$ input pin		

- P52/ $\overline{\text{IRQ10}}$ /TxD1/IrTxD

The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI\_1 and the P52DDR bit.

When the IRQ10E bit in IER16 of the interrupt controller is set to 1 or the ISS10 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ10}}$  input pin. To use this pin as the  $\overline{\text{IRQ10}}$  input pin, clear the P52DDR bit to 0.

TE	0		1
P52DDR	0	1	—
Pin function	P52 input pin	P52 output pin	TxD1/IrTxD output pin
	$\overline{\text{IRQ10}}$ input pin		

- P51/ $\overline{\text{IRQ9}}$ /RxD0

The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI\_0 and the P51DDR bit.

When the IRQ9E bit in IER16 of the interrupt controller is set to 1 or the ISS9 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ9}}$  input pin. To use this pin as the  $\overline{\text{IRQ9}}$  input pin, clear the P51DDR bit to 0.

RE	0		1
P51DDR	0	1	—
Pin function	P51 input pin	P51 output pin	RxD0 input pin
	$\overline{\text{IRQ9}}$ input pin		

- P50/ $\overline{\text{IRQ8}}$ /TxD0

The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI\_0 and the P50DDR bit.

When the IRQ8E bit in IER16 of the interrupt controller is set to 1 or the ISS8 bit in ISSR16 is cleared to 0, this pin can be used as the  $\overline{\text{IRQ8}}$  input pin. To use this pin as the  $\overline{\text{IRQ8}}$  input pin, clear the P50DDR bit to 0.

TE	0		1
P50DDR	0	1	—
Pin function	P50 input pin	P50 output pin	TxD0 output pin
	$\overline{\text{IRQ8}}$ input pin		

## 9.6 Port 6

Port 6 is an 8-bit I/O port. Port 6 pins also function as the FRT input/output pin, enhanced A/D conversion input pin, keyboard input pin, timer connection input/output pins, and external USB driver/receiver input/output pin. Port 6 functions change according to the operating mode. The port can be used as the extended data bus (lower eight bits). Port 6 has the following registers.

- Port 6 data direction register (P6DDR)
- Port 6 data register (P6DR)
- Port 6 pull-up MOS control register (KMPCR6)
- System control register 2 (SYSCR2)

### 9.6.1 Port 6 Data Direction Register (P6DDR)

The individual bits of P6DDR specify input or output for the pins of port 6.

Bit	Bit Name	Initial Value	R/W	Description
7	P67DDR	0	W	Extended mode (16-bit data bus):
6	P66DDR	0	W	The port functions as the data bus regardless of the values in these bits.
5	P65DDR	0	W	Single-chip mode/extended mode (8-bit data bus): If port 6 pins are specified for use as the general I/O port, the corresponding port 6 pins are output ports when the P6DDR bits are set to 1, and input ports when cleared to 0.
4	P64DDR	0	W	
3	P63DDR	0	W	
2	P62DDR	0	W	
1	P61DDR	0	W	
0	P60DDR	0	W	

### 9.6.2 Port 6 Data Register (P6DR)

P6DR stores output data for the port 6 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P67DR	0	R/W	Extended mode (16-bit data bus):
6	P66DR	0	R/W	If a port 6 read is performed while the P6DDR bits are set to 1, the P6DR values are read. If a port 6 read is performed while the P6DDR bits are cleared to 0, 1 is read.
5	P65DR	0	R/W	
4	P64DR	0	R/W	Single-chip mode/extended mode (8-bit data bus): P6DR stores output data for the port 6 pins that are used as the general output port.
3	P63DR	0	R/W	
2	P62DR	0	R/W	
1	P61DR	0	R/W	
0	P60DR	0	R/W	If a port 6 read is performed while the P6DDR bits are set to 1, the P6DR values are read. If a port 6 read is performed while the P6DDR bits are cleared to 0, the pin states are read.

### 9.6.3 Port 6 Pull-Up MOS Control Register (KMPCR6)

KMPCR6 controls the port 6 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	KM7PCR	0	R/W	Extended mode (16-bit data bus):
6	KM6PCR	0	R/W	Operation is not affected.
5	KM5PCR	0	R/W	Single-chip mode/extended mode (8-bit data bus):
4	KM4PCR	0	R/W	When the pins are in input state, the
3	KM3PCR	0	R/W	corresponding input pull-up MOS is turned on
2	KM2PCR	0	R/W	when a KMPCR6 bit is set to 1.
1	KM1PCR	0	R/W	
0	KM0PCR	0	R/W	

---

### 9.6.4 System Control Register 2 (SYSCR2)

SYSCR2 selects the port 6 input level and current specification for the input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	KWUL1	0	R/W	Key Wakeup Level 1, 0:
6	KWUL0	0	R/W	Select the port 6 input level. 00: Standard input level is selected 01: Input level 1 is selected 10: Input level 2 is selected 11: Input level 3 is selected
5	P6PUE	0	R/W	Port 6 Input Pull-Up Extra Selects the current specification for the input pull-up MOS connected by means of KMPCR settings. 0: Standard current specification is selected 1: Current-limit specification is selected
4, 3	—	All 0	R/(W)	Reserved The initial value should not be changed.
2	CKCHGE	0	R/W	For details, refer to section 27.1.3, System Control Register 2 (SYSCR2).
1	—	0	R/(W)	Reserved The initial value should not be changed.
0	PLCKS	0	R/W	For details, refer to section 27.1.3, System Control Register 2 (SYSCR2).

### 9.6.5 Pin Functions

The relationship between the operating modes, register setting values, and pin functions are as follows. In the tables, the symbol “—” stands for Don't care.

In extended mode, port 6 pins also function as the bidirectional data bus, CompactFlash bidirectional data bus, FRT input/output pin, enhanced A/D conversion input pin, keyboard input pin, timer connection input/output pins, or I/O port. The extended data bus width can be specified by bits ABW and ABW256 in WSCR of BSC and bit ABWCP in BCR2. When the 16-bit bus interface is specified, port 6 pins function as the bidirectional data bus (D7 to D0/CPD7 to CPD0). When the 8-bit bus interface is specified, port 6 pins function as the FRT input/output pin, enhanced A/D conversion input pin, keyboard input pin, timer connection input/output pins, or I/O port.

In single-chip mode, port 6 pins function as the FRT input/output pin, enhanced A/D conversion input pin, keyboard input pin, timer connection input/output pins, or external USB driver/receiver input/output pin, or I/O port

**Extended Mode with 16-Bit Data Bus Specified:** Port 6 pins function as the lower 8 bits of the data bus.

Note: When the 16-bit data bus interface is specified in extended mode, USB should not be enabled. If USB is enabled, port 6 cannot operate as a data bus.

**Extended Mode with 8-Bit Data Bus Specified:**

- P67/CIN7/ $\overline{\text{KIN7}}$

The function of port 6 pins is switched as shown below according to the P67DDR bit.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'111, this pin can be used as the CIN7 input pin. When the  $\overline{\text{KMIM7}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN7}}$  input pin. To use this pin as the  $\overline{\text{KIN7}}$  input pin, clear the P67DDR bit to 0.

P67DDR	0	1
Pin function	P67 input pin	P67 output pin
	$\overline{\text{KIN7}}$ input pin/ $\overline{\text{KIN7}}$ input pin	



- P66/FTOB/CIN6/ $\overline{\text{KIN6}}$ /CBLANK

The function of port 6 pins is switched as shown below according to the combination of the CBOE bit in TCONRO of the timer connection, the OEB bit in TOCR of FRT, and the P66DDR bit.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'110, this pin can be used as the CIN6 input pin. When the KMIM6 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN6}}$  input pin. To use this pin as the  $\overline{\text{KIN6}}$  input pin, clear the P66DDR bit to 0.

CBOE	0			1
OEB	0		1	—
P66DDR	0	1	—	—
Pin function	P66 input pin	P66 output pin	FTOB output pin	CBLANK output pin
	$\overline{\text{KIN6}}$ input pin			

- P65/FTID/CIN5/ $\overline{\text{KIN5}}$ /CSYNCI

The function of port 6 pins is switched as shown below according to the P65DDR bit.

When the ICIDE bit in TIER of FRT is set to 1, this pin can be used as the FTID input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'101, this pin can be used as the CIN5 input pin. When the KMIM5 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN5}}$  input pin. To use this pin as the  $\overline{\text{KIN5}}$  input pin, clear the P65DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IHI signal) in TCONRI of the timer connection are set to B'01, this pin can be used as the CSYNCI input pin.

P65DDR	0	1
Pin function	P65 input pin	P65 output pin
	$\overline{\text{KIN5}}$ input pin	

- P64/FTIC/CIN4/ $\overline{\text{KIN4}}$ /CLAMPO

The function of port 6 pins is switched as shown below according to the combination of the CLOE bit in TCONRO of the timer connection and the P64DDR bit.

When the ICICE bit in TIER of FRT is set to 1, this pin can be used as the FTIC input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'100, this pin can be used as the CIN4 input pin. When the KMIM4 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN4}}$  input pin. To use this pin as the  $\overline{\text{KIN4}}$  input pin, clear the P64DDR bit to 0.

CLOE	0	0	1
P64DDR	0	1	—
Pin function	P64 input pin	P64 output pin	CLAMPO output pin
	FTIC input pin/CIN4 input pin/ $\overline{KIN4}$ input pin		

- P63/FTIB/CIN3/ $\overline{KIN3}$ /VFBACKI

The function of port 6 pins is switched as shown below according to the P63DDR bit.

When the ICIBE bit in TIER of FRT is set to 1, this pin can be used as the FTIB input pin.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'011, this pin can be used as the CIN3 input pin. When the KMIM3 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{KIN3}$  input pin. To use this pin as the  $\overline{KIN3}$  input pin, clear the P63DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IVI signal) in TCONRI of the timer connection are cleared to B'00, this pin can be used as the VFBACKI input pin.

P63DDR	0	1
Pin function	P63 input pin	P63 output pin
	FTIB input pin/CIN3 input pin/ $\overline{KIN3}$ input pin/VFBACKI input pin	

- P62/FTIA/CIN2/ $\overline{KIN2}$ /VSYNCI

The function of port 6 pins is switched as shown below according to the P62DDR bit.

When the ICIAE bit in TIER of FRT is set to 1, this pin can be used as the FTIA input pin.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'010, this pin can be used as the CIN2 input pin. When the KMIM2 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{KIN2}$  input pin. To use this pin as the  $\overline{KIN2}$  input pin, clear the P62DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IVI signal) in TCONRI of the timer connection are set to B'11, this pin can be used as the VSYNCI input pin.

P62DDR	0	1
Pin function	P62 input pin	P62 output pin
	FTIA input pin/CIN2 input pin/ $\overline{KIN2}$ input pin/VSYNCI input pin	

- P61/FTOA/CIN1/ $\overline{\text{KIN1}}$ /VSYNCO

The function of port 6 pins is switched as shown below according to the combination of the VOE bit in TCONRO of the timer connection, the OEA bit in TOCR of FRT, and the P61DDR bit.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'001, this pin can be used as the CIN1 input pin. When the  $\overline{\text{KMIM1}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN1}}$  input pin. To use this pin as the  $\overline{\text{KIN1}}$  input pin, clear the P61DDR bit to 0.

VOE	0			1
OEA	0		1	—
P61DDR	0	1	—	—
Pin function	P61 input pin	P61 output pin	FTOA output pin	VSYNCO output pin
	$\overline{\text{KIN1}}$ input pin			

- P60/FTCI/CIN0/ $\overline{\text{KIN0}}$ /HFBACKI

The function of port 6 pins is switched as shown below according to the P60DDR bit.

When the CKS1 and CKS0 bits in TCR of FRT are both set to 1, this pin can be used as the FTCI input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are cleared to B'000, this pin can be used as the CIN0 input pin.

When the  $\overline{\text{KMIM0}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN0}}$  input pin. To use this pin as the  $\overline{\text{KIN0}}$  input pin, clear the P60DDR bit to 0.

When the SIMOD1 and SIMOD0 bits (IHI signal) in TCONRI of the timer connection are cleared to B'00, this pin can be used as the HFBACKI input pin.

P60DDR	0	1
Pin function	P60 input pin	P60 output pin
	$\overline{\text{KIN0}}$ input pin/HFBACKI input pin	

**Single-Chip Mode:**

- P67/CIN7/ $\overline{\text{KIN7}}$ /DPLS

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB and the P67DDR bit.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'111, this pin can be used as the CIN7 input pin. When the KMIM7 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN7}}$  input pin. To use this pin as the  $\overline{\text{KIN7}}$  input pin, clear the P67DDR bit to 0.

FADSEL	0		1
P67DDR	0	1	—
Pin function	P67 input pin		DPLS input pin
	CIN7 input pin/ $\overline{\text{KIN7}}$ input pin		

- P66/FTOB/CIN6/ $\overline{\text{KIN6}}$ /CBLANK/DMNS

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB, the CBOE bit in TCONRO of the timer connection, the OEB bit in TOCR of FRT, and the P66DDR bit.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'110, this pin can be used as the CIN6 input pin. When the KMIM6 bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN6}}$  input pin. To use this pin as the  $\overline{\text{KIN6}}$  input pin, clear the P66DDR bit to 0.

FADSEL	0			1
CBOE	0		1	—
OEB	0		1	—
P66DDR	0	1	—	—
Pin function	P66 input pin	P66 output pin	FTOB output pin	CBLANK output pin
	CIN6 input pin/ $\overline{\text{KIN6}}$ input pin			

- P65/FTID/CIN5/ $\overline{\text{KIN5}}$ /CSYNCI/XVERDATA

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB and the P65DDR bit.

When the ICIDE bit in TIER of FRT is set to 1, this pin can be used as the FTID input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'101, this pin can be used as the CIN5 input pin. When the  $\overline{\text{KMIM5}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN5}}$  input pin. To use this pin as the  $\overline{\text{KIN5}}$  input pin, clear the P65DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IHI signal) in TCONRI of the timer connection are set to B'01, this pin can be used as the CSYNCI input pin.

FADSEL	0		1
P65DDR	0	1	—
Pin function	P65 input pin	P65 output pin	XVERDATA input pin
	FTID input pin/CIN5 input pin/ $\overline{\text{KIN5}}$ input pin/CSYNCI input pin		

- P64/FTIC/CIN4/ $\overline{\text{KIN4}}$ /CLAMPO/TXDPLS

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB, the CLOE bit in TCONRO of the timer connection, and the P64DDR bit.

When the ICICE bit in TIER of FRT is set to 1, this pin can be used as the FTIC input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'100, this pin can be used as the CIN4 input pin. When the  $\overline{\text{KMIM4}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN4}}$  input pin. To use this pin as the  $\overline{\text{KIN4}}$  input pin, clear the P64DDR bit to 0.

FADSEL	0			1
CLOE	0		1	—
P64DDR	0	1	—	—
Pin function	P64 input pin	P64 output pin	CLAMPO output pin	TXDPLS output pin
	FTIC input pin/CIN4 input pin/ $\overline{\text{KIN4}}$ input pin			

- P63/FTIB/CIN3/ $\overline{\text{KIN3}}$ /VFBACKI/TXDMNS

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB and the P63DDR bit.

When the ICIBE bit in TIER of FRT is set to 1, this pin can be used as the FTIB input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'011, this pin can be used as the CIN3 input pin. When the  $\overline{\text{KMIM3}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN3}}$  input pin. To use this pin as the  $\overline{\text{KIN3}}$  input pin, clear the P63DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IVI signal) in TCONRI of the timer connection are cleared to B'00, this pin can be used as the VFBACKI input pin.

FADSEL	0		1
P63DDR	0	1	—
Pin function	P63 input pin	P63 output pin	TXDMNS output pin
	FTIB input pin/CIN3 input pin/ $\overline{\text{KIN3}}$ input pin/VFBACKI input pin		

- P62/FTIA/CIN2/ $\overline{\text{KIN2}}$ /VSYNCI/TXENL

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB and the P62DDR bit.

When the ICIAE bit in TIER of FRT is set to 1, this pin can be used as the FTIA input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'010, this pin can be used as the CIN2 input pin. When the  $\overline{\text{KMIM2}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN2}}$  input pin. To use this pin as the  $\overline{\text{KIN2}}$  input pin, clear the P62DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IVI signal) in TCONRI of the timer connection are set to B'11, this pin can be used as the VSYNCI input pin.

FADSEL	0		1
P62DDR	0	1	—
Pin function	P62 input pin	P62 output pin	TXENL output pin
	FTIA input pin/CIN2 input pin/ $\overline{\text{KIN2}}$ input pin/VSYNCI input pin		

- P61/FTOA/CIN1/ $\overline{\text{KIN1}}$ /VSYNCO/SUSPEND

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB, the VOE bit in TCONRO of the timer connection, the OEA bit in TOCR of FRT, and the P61DDR bit.

When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are set to B'001, this pin can be used as the CIN1 input pin. When the  $\overline{\text{KMIM1}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN1}}$  input pin. To use this pin as the  $\overline{\text{KIN1}}$  input pin, clear the P61DDR bit to 0.

FADSEL	0			1	
VOE	0		1	—	
OEA	0		1	—	—
P61DDR	0	1	—	—	—
Pin function	P61 input pin	P61 output pin	FTOA output pin	VSYNCO output pin	SUSPEND output pin
	CIN1 input pin/ $\overline{\text{KIN1}}$ input pin				

- P60/FTCI/CIN0/ $\overline{\text{KIN0}}$ /HFBACKI/SPEED

The function of port 6 pins is switched as shown below according to the combination of the FADSEL bit in USBCR0 of USB and the P60DDR bit.

When the CKS1 and CKS0 bits in TCR of FRT are both set to 1, this pin can be used as the FTCI input pin. When the KBADE bit in KBCOMP of the A/D converter is set to 1 while the KBCH2 to KBCH0 bits are cleared to B'000, this pin can be used as the CIN0 input pin. When the  $\overline{\text{KMIM0}}$  bit in KMIMR6 of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN0}}$  input pin. To use this pin as the  $\overline{\text{KIN0}}$  input pin, clear the P60DDR bit to 0. When the SIMOD1 and SIMOD0 bits (IHI signal) in TCONRI of the timer connection are cleared to B'00, this pin can be used as the HFBACKI input pin.

FADSEL	0		1
P60DDR	0	1	—
Pin function	P60 input pin	P60 output pin	SPEED output pin
	FTCI input pin/CIN0 input pin/ $\overline{\text{KIN0}}$ input pin/HFBACKI input pin		

## 9.6.6 Port 6 Input Pull-Up MOS

Port 6 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used regardless of the operating mode. Table 9.5 summarizes the input pull-up MOS states.

**Table 9.5 Port 6 Input Pull-Up MOS States**

<b>Reset</b>	<b>Hardware Standby Mode</b>	<b>Software Standby Mode</b>	<b>In Other Operations</b>
Off	Off	On/Off	On/Off

Legend:

Off: Always off.

On/Off: On when input state and KMPCR = 1; otherwise off.

## 9.7 Port 7

Port 7 is a 6-bit input port. Port 7 pins also function as the A/D converter analog input pins, D/A converter analog output pins, and interrupt input pins. When the ISS bit in ISSR is set to 1, these pins can be used as the interrupt input pins. Port 7 has the following register.

- Port 7 input data register (P7PIN)

### 9.7.1 Port 7 Input Data Register (P7PIN)

P7PIN indicates the pin states.

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	P77PIN	Undefined*	R	When a P7PIN read is performed, the pin states are always read.
6	P76PIN	Undefined*	R	
5	P75PIN	Undefined*	R	
4	P74PIN	Undefined*	R	
3	P73PIN	Undefined*	R	
2	P72PIN	Undefined*	R	
1, 0	—	All 1	R	

Note: \* The initial values of bits 7 to 2 are determined according to the pin states of P77 to P72.



## 9.8 Port 8

Port 8 is an 8-bit I/O port. Port 8 pins also function as the A/D converter external trigger input pin, SCI\_0, SCI\_1, and SCI\_2 clock input/output pins, IIC\_0 and IIC\_1 input/output pins, ExTMR\_0, ExTMR\_1, ExTMR\_X, and ExTMR\_Y input pins, USB external clock input pin, and interrupt input pins. Port 8 is an NMOS push-pull output. Port 8 has the following registers.

- Port 8 data direction register (P8DDR)
- Port 8 data register (P8DR)

### 9.8.1 Port 8 Data Direction Register (P8DDR)

The individual bits of P8DDR specify input or output for the pins of port 8.

Bit	Bit Name	Initial Value	R/W	Description
7	P87DDR	0	W	If port 8 pins are specified for use as the general I/O port, the corresponding port 8 pins are output ports when the P8DDR bits are set to 1, and input ports when cleared to 0.
6	P86DDR	0	W	
5	P85DDR	0	W	
4	P84DDR	0	W	
3	P83DDR	0	W	
2	P82DDR	0	W	
1	P81DDR	0	W	
0	P80DDR	0	W	

### 9.8.2 Port 8 Data Register (P8DR)

P8DR stores output data for the port 8 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P87DR	0	R/W	P8DR stores output data for the port 8 pins that are used as the general output port. If a port 8 read is performed while the P8DDR bits are set to 1, the P8DR values are read. If a port 8 read is performed while the P8DDR bits are cleared to 0, the pin states are read.
6	P86DR	0	R/W	
5	P85DR	0	R/W	
4	P84DR	0	R/W	
3	P83DR	0	R/W	
2	P82DR	0	R/W	
1	P81DR	0	R/W	
0	P80DR	0	R/W	

### 9.8.3 Pin Functions

The relationship between register setting values and pin functions are as follows. In the tables, the symbol “—” stands for Don't care.

- $\overline{\text{P87}}/\overline{\text{ExIRQ15}}/\overline{\text{ADTRG}}/\overline{\text{ExTMIY}}/\text{USEXCL}$

The pin function is switched as shown below according to the P87DDR bit.

When the TRGS1 and TRGS0 bits in ADCR of the A/D converter are both set to 1, this pin can be used as the  $\overline{\text{ADTRG}}$  input pin. When the CKSEL2 to CKSEL0 bits in UPLLCR of the USB are all set to 1, this pin can be used as the USEXCL input pin.

When the ISS15 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ15}}$  input pin. When the TMIYS bit in PTCNT0 is set to 1, this pin can be used as the TMIY (TMCIX/TMRIY) input pin. To use this pin as the  $\overline{\text{ExIRQ15}}$  input pin, clear the P87DDR bit to 0.

When this pin is used as the P87 output pin, the output format is NMOS push-pull output.

P87DDR	0	1
Pin function	P87 input pin	P87 output pin
	$\overline{\text{ExIRQ15}}$ input pin/ $\overline{\text{ADTRG}}$ input pin/ $\overline{\text{ExTMIY}}$ input pin/USEXCL input pin	

- P86/ $\overline{\text{ExIRQ14}}$ /SCK2/ExTMIX

The pin function is switched as shown below according to the combination of the  $C/\overline{A}$  bit in SMR of SCI\_2, the CKE1 and CKE0 bits in SCR, and the P86DDR bit.

When the ISS14 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ14}}$  input pin. When the TMIXS bit in PTCNT0 is set to 1, this pin can be used as the TMIX (TMCIX/TMRIX) input pin. To use this pin as the  $\overline{\text{ExIRQ14}}$  input pin, clear the P86DDR bit to 0.

When this pin is used as the P86 output pin, the output format is NMOS push-pull output.

CKE1	0				1
$C/\overline{A}$	0			1	—
CKE0	0		1	—	—
P86DDR	0	1	—	—	—
Pin function	P86 input pin	P86 output pin	SCK2 output pin	SCK2 output pin	SCK2 input pin
	$\overline{\text{ExIRQ14}}$ input pin/ExTMIX input pin				

- P85/ $\overline{\text{ExIRQ13}}$ /SCK1/ExTMI1

The pin function is switched as shown below according to the combination of the  $C/\overline{A}$  bit in SMR of SCI\_1, the CKE1 and CKE0 bits in SCR, and the P85DDR bit.

When the ISS13 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ13}}$  input pin. When the TMI1S bit in PTCNT0 is set to 1, this pin can be used as the TMI1 (TMI1/TMRI1) input pin. To use this pin as the  $\overline{\text{ExIRQ13}}$  input pin, clear the P85DDR bit to 0.

When this pin is used as the P85 output pin, the output format is NMOS push-pull output.

CKE1	0				1
$C/\overline{A}$	0			1	—
CKE0	0		1	—	—
P85DDR	0	1	—	—	—
Pin function	P85 input pin	P85 output pin	SCK1 output pin	SCK1 output pin	SCK1 input pin
	$\overline{\text{ExIRQ13}}$ input pin/ExTMI1 input pin				

- P84/ $\overline{\text{ExIRQ12}}$ /SCK0/ExTMI0

The pin function is switched as shown below according to the combination of the  $C/\overline{A}$  bit in SMR of SCI\_0, the CKE1 and CKE0 bits in SCR, and the P84DDR bit.

When the ISS12 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ12}}$  input pin. When the TMI0S bit in PTCNT0 is set to 1, this pin can be used as the TMI0 (TMI0/TMRI0) input pin. To use this pin as the  $\overline{\text{ExIRQ12}}$  input pin, clear the P84DDR bit to 0.

When this pin is used as the P84 output pin, the output format is NMOS push-pull output.

CKE1	0				1
$C/\overline{A}$	0			1	—
CKE0	0		1	—	—
P84DDR	0	1	—	—	—
Pin function	P84 input pin	P84 output pin	SCK0 output pin	SCK0 output pin	SCK0 input pin
	$\overline{\text{ExIRQ12}}$ input pin/ExTMI0 input pin				

- P83/ $\overline{\text{ExIRQ11}}$ /SDA1

The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC\_1 and the P83DDR bit.

When the ISS11 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ11}}$  input pin. To use this pin as the  $\overline{\text{ExIRQ11}}$  input pin, clear the P83DDR bit to 0.

When this pin is used as the P83 output pin, the output format is NMOS push-pull output. The output format for SDA1 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P83DDR	0	1	—
Pin function	P83 input pin	P83 output pin	SDA1 input/output pin
	$\overline{\text{ExIRQ11}}$ input pin		

- P82/ $\overline{\text{ExIRQ10}}$ /SCL1

The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC\_1 and the P82DDR bit.

When the ISS10 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ10}}$  input pin. To use this pin as the  $\overline{\text{ExIRQ10}}$  input pin, clear the P82DDR bit to 0.

When this pin is used as the P82 output pin, the output format is NMOS push-pull output. The output format for SCL1 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P82DDR	0	1	—
Pin function	P82 input pin	P82 output pin	SCL1 input/output pin
	$\overline{\text{ExIRQ10}}$ input pin		

- P81/ $\overline{\text{ExIRQ9}}$ /SDA0

The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC\_0 and the P81DDR bit.

When the ISS9 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ9}}$  input pin. To use this pin as the  $\overline{\text{ExIRQ9}}$  input pin, clear the P81DDR bit to 0.

When this pin is used as the P81 output pin, the output format is NMOS push-pull output. The output format for SDA0 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P81DDR	0	1	—
Pin function	P81 input pin	P81 output pin	SDA0 input/output pin
	$\overline{\text{ExIRQ9}}$ input pin		

- P80/ $\overline{\text{ExIRQ8}}$ /SCL0

The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC\_0 and the P80DDR bit.

When the ISS8 bit in ISSR16 of the interrupt controller is set to 1, this pin can be used as the  $\overline{\text{ExIRQ8}}$  input pin. To use this pin as the  $\overline{\text{ExIRQ8}}$  input pin, clear the P80DDR bit to 0.

When this pin is used as the P80 output pin, the output format is NMOS push-pull output. The output format for SCL0 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P80DDR	0	1	—
Pin function	P80 input pin	P80 output pin	SCL0 input/output pin
	$\overline{\text{ExIRQ8}}$ input pin		

## 9.9 Port 9

Port 9 is an 8-bit I/O port. However note that pin P96 cannot be used as a general output port. Port 9 pins also function as the bus control I/O pins, CompactFlash control I/O pins, the system clock output pin, and the external subclock input pin. Pin functions are switched depending on the operating mode. Port 9 has the following registers.

- Port 9 data direction register (P9DDR)
- Port 9 data register (P9DR)

### 9.9.1 Port 9 Data Direction Register (P9DDR)

The individual bits of P9DDR specify input or output for the pins of port 9.

Bit	Bit Name	Initial Value	R/W	Description
7	P97DDR	0	W	If port 9 pins are specified for use as the general I/O port, the corresponding port 9 pins are output ports when the P9DDR bits are set to 1, and input ports when cleared to 0.
6	P96DDR	0	W	When this bit is set to 1, the corresponding port 9 pin is the system clock output pin ( $\phi$ ), and as a general input port when cleared to 0.
5	P95DDR	0	W	If port 9 pins are specified for use as the general I/O port, the corresponding port 9 pins are output ports when the P9DDR bits are set to 1, and input ports when cleared to 0.
4	P94DDR	0	W	
3	P93DDR	0	W	
2	P92DDR	0	W	
1	P91DDR	0	W	
0	P90DDR	0	W	

### 9.9.2 Port 9 Data Register (P9DR)

P9DR stores output data for the port 9 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P97DR	0	R/W	P9DR stores output data for the port 9 pins that are used as the general output port except for bit 6.
6	P96DR	Undefined*	R	
5	P95DR	0	R/W	If a port 9 read is performed while the P9DDR bits are set to 1, the P9DR values are read. If a port 9 read is performed while the P9DDR bits are cleared to 0, the pin states are read.
4	P94DR	0	R/W	
3	P93DR	0	R/W	
2	P92DR	0	R/W	
1	P91DR	0	R/W	
0	P90DR	0	R/W	

Note: \* The initial value of bit 6 is determined according to the P96 pin state.

### 9.9.3 Pin Functions

The relationship between the operating mode, register setting values, and pin functions are as follows. In the tables, the symbol “—” stands for Don't care.

- $\overline{P97}/\overline{WAIT}/\overline{CPWAIT}/\overline{CS256}$

The pin function is switched as shown below according to the combination of the operating mode, the CS256E bit in SYSCR, the WMS1 bit in WSCR, the WMS21 bit in WSCR2, and the P97DDR bit.

Operating Mode	Extended Mode			Single-Chip Mode		
	All bits are set as 0		One bit is set as 1	—		
WMS1, WMS21	0		1	—		
CS256E	0		1	—		
P97DDR	0	1	—	0	1	
Pin function	P97 input pin	P97 output pin	$\overline{CS256}$ output pin	$\overline{WAIT}$ (WMS1 = 1) and $\overline{CPWAIT}$ (WMS21 = 1) input pins	P97 input pin	P97 output pin

- P96/ $\phi$ /EXCL

The pin function is switched as shown below according to the combination of the EXCLE bit in LPWRCR and the P96DDR bit.

P96DDR	0		1
EXCLE	0	1	0
Pin function	P96 input pin	EXCL input pin	$\phi$ output pin

- P95/ $\overline{AS}$ / $\overline{IOS}$

The pin function is switched as shown below according to the combination of the operating mode, the IOSE bit in SYSCR, and the P95DDR bit.

Operating Mode	Extended Mode		Single-Chip Mode	
P95DDR	—		0	1
IOSE	0	1	—	
Pin function	$\overline{AS}$ output pin	$\overline{IOS}$ output pin	P95 input pin	P95 output pin

- P94/ $\overline{HWR}$ / $\overline{CPWE}$

The pin function is switched as shown below according to the combination of the operating mode and the P94DDR bit.

Operating Mode	Extended Mode	Single-Chip Mode	
P94DDR	—	0	1
Pin function	$\overline{HWR}/\overline{CPWR}$ output pin	P94 input pin	P94 output pin

- P93/ $\overline{RD}$ / $\overline{CPOE}$

The pin function is switched as shown below according to the combination of the operating mode and the P93DDR bit.

Operating Mode	Extended Mode	Single-Chip Mode	
P93DDR	—	0	1
Pin function	$\overline{RD}/\overline{CPOE}$ output pin	P93 input pin	P93 output pin



- P92/ $\overline{\text{CPCS1}}$

The pin function is switched as shown below according to the combination of the operating mode, the CPCSE bit in BCR2 of BSC, and the P92DDR bit.

Operating Mode	Extended Mode			Single-Chip Mode	
	CPCSE	0		1	—
P92DDR	0	1	—	0	1
Pin function	P92 input pin	P92 output pin	$\overline{\text{CPCS1}}$ output pin	P92 input pin	P92 output pin

- P91/ $\overline{\text{CPCS2}}$

The pin function is switched as shown below according to the combination of the operating mode, the CPCSE bit in BCR2 of BSC, the CFE bit in BCR, and the P91DDR bit.

Operating Mode	Extended Mode			Single-Chip Mode	
	CPCSE and CFE	Either bit is 0		Both bits are 1	—
P91DDR	0	1	—	0	1
Pin function	P91 input pin	P91 output pin	$\overline{\text{CPCS2}}$ output pin	P91 input pin	P91 output pin

- P90/ $\overline{\text{LWR}}$

The pin function is switched as shown below according to the combination of the operating mode, the ABW and ABW256 bits in WSCR, the ABWCP bit in BCR2, and the P90DDR bit.

Operating Mode	Extended Mode			Single-Chip Mode	
	ABW, ABW256, ABWCP	All bits are 1		One bit is 0	—
P90DDR	0	1	—	0	1
Pin function	P90 input pin	P90 output pin	$\overline{\text{LWR}}$ output pin	P90 input pin	P90 output pin

## 9.10 Port A

Port A is a 2-bit I/O port. Port A pins also function as the address output, keyboard input, SCI\_0 and SCI\_2 external control pins. Pin functions are switched depending on the operating mode. Port A has the following registers. PADDR and PAPIN have the same address.

- Port A data direction register (PADDR)
- Port A output data register (PAODR)
- Port A input data register (PAPIN)

### 9.10.1 Port A Data Direction Register (PADDR)

The individual bits of PADDR specify input or output for the pins of port A.

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 0	(W)	Reserved These bits cannot be modified.
1	PA1DDR	0	W	In extended mode (mode 2):
0	PA0DDR	0	W	The corresponding port A pins are address output ports when the PADDR bits are set to 1, and input ports when cleared to 0. Pins function as the address output port depending on the setting of bits IOSE and CS256E in SYSCR.  In single-chip mode or extended mode (mode 3): The corresponding port A pins are output ports when the PADDR bits are set to 1, and input ports when cleared to 0.

### 9.10.2 Port A Output Data Register (PAODR)

PAODR stores output data for the port A pins.

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 1	R/(W)	Reserved These bits are always read as 1. The initial value should not be changed.
1	PA1ODR	0	R/W	PAODR stores output data for the port A pins that are used as the general output port.
0	PA0ODR	0	R/W	

### 9.10.3 Port A Input Data Register (PAPIN)

PAPIN indicates the pin states.

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 1	R	Reserved These bits are always read as 1.
1	PA1PIN	Undefined*	R	When a PAPIN read is performed, the pin states are always read.
0	PA0PIN	Undefined*	R	

Note: \* The initial values of bits 1 and 0 are determined according to the pin states of PA1 and PA0.

### 9.10.4 Pin Functions

The relationship between the operating mode, register setting values, and pin functions are as follows. In the tables, the symbol “—” stands for Don't care.

#### Extended Mode (Mode 2):

- PA1/A17/ $\overline{\text{KIN9}}$ /SSE2I

The function of port A pins is switched as shown below according to the combination of the SSE bit in SEMR of SCI\_2, the C/ $\overline{\text{A}}$  bit in SMR, the CKE1 bit in SCR, the CS256E and IOSE bits in SYSCR, the ADFULLE bit in BCR2 of BSC, and the PA1DDR bit.

Address 13 in the following table is expressed by the following logical expression:

$$\text{Address 13} = 1 : \overline{\text{ADFULLE}} \cdot \overline{\text{CS256E}} \cdot \text{IOSE}$$

When the KMIM9 bit in KMIMRA of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN9}}$  input pin. To use this pin as the  $\overline{\text{KIN9}}$  input pin, clear the PA1DDR bit to 0.

SSE	0			1
C/ $\overline{\text{A}}$	—			1
CKE1	—			1
PA1DDR	0	1	1	—
Address 13	1		0	—
Pin function	PA1 input pin	PA1 output pin	A17 output pin	SSE2I
	$\overline{\text{KIN9}}$ input pin			

Note: Even if SSE = 0, PA1DDR = 1 and CPCSE = 1, pin PA1 can be used as an output pin only when IOSE = 1.

- PA0/A16/ $\overline{\text{KIN8}}$ /SSE0I

The function of port A pins is switched as shown below according to the combination of the SSE bit in SEMR of SCI\_0, the C/ $\overline{\text{A}}$  bit in SMR, the CKE1 bit in SCR, the CS256E and IOSE bits in SYSCR, the ADFULLE bit in BCR2 of BSC, and the PA0DDR bit.

Address 13 in the following table is expressed by the following logical expression:

$$\text{Address 13} = 1:\overline{\text{ADFULLE}} \cdot \overline{\text{CS256E}} \cdot \text{IOSE}$$

When the KMIM8 bit in KMIMRA of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN8}}$  input pin. To use this pin as the  $\overline{\text{KIN8}}$  input pin, clear the PA0DDR bit to 0.

SSE	0			1
C/ $\overline{\text{A}}$	—			1
CKE1	—			1
PA0DDR	0	1	1	—
Address 13	1		0	—
Pin function	PA0 input pin	PA0 output pin	A16 output pin	SSE0I
	$\overline{\text{KIN8}}$ input pin			

Note: Even if SSE = 0, PA0DDR = 1 and CPCSE = 1, pin PA0 can be used as an output pin only when IOSE = 1.

### Single-Chip Mode and Extended Mode (Mode 3):

- PA1/ $\overline{\text{KIN9}}$ /SSE2I

The function of port A pins is switched as shown below according to the combination of the SSE bit in SEMR of SCI\_2, the C/ $\overline{\text{A}}$  bit in SMR, the CKE1 bit in SCR, and the PA1DDR bit.

When the KMIM9 bit in KMIMRA of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN9}}$  input pin. To use this pin as the  $\overline{\text{KIN9}}$  input pin, clear the PA1DDR bit to 0.

SSE	0		1
C/ $\overline{\text{A}}$	—		1
CKE1	—		1
PA1DDR	0	1	—
Pin function	PA1 input pin	PA1 output pin	SSE2I
	$\overline{\text{KIN9}}$ input pin		

- PA0/ $\overline{\text{KIN8}}$ /SSE0I

The function of port A pins is switched as shown below according to the combination of the SSE bit in SEMR of SCI\_0, the C/ $\overline{\text{A}}$  bit in SMR, the CKE1 bit in SCR, and the PA0DDR bit.

When the KMIM8 bit in KMIMRA of the interrupt controller is cleared to 0, this pin can be used as the  $\overline{\text{KIN8}}$  input pin. To use this pin as the  $\overline{\text{KIN8}}$  input pin, clear the PA0DDR bit to 0.

SSE	0		1
C/ $\overline{\text{A}}$	—		1
CKE1	—		1
PA0DDR	0	1	—
Pin function	PA0 input pin	PA0 output pin	SSE0I
	$\overline{\text{KIN8}}$ input pin		

### 9.10.5 Input Pull-Up MOS

Port A has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used in any operating mode, and can be specified as on or off on a bit-by-bit basis.

When a PADDR bit is cleared to 0, setting the corresponding PAODR bit to 1 turns on the input pull-up MOS.

The input pull-up MOS is in the off state after a reset and in hardware standby mode. The prior state is retained in software standby mode.

Table 9.6 summarizes the input pull-up MOS states.

**Table 9.6 Port A Input Pull-Up MOS States**

Reset	Hardware Standby Mode	Software Standby Mode	In Other Operations
Off	Off	On/Off	On/Off

Legend:

Off: Always off.

On/Off: On when PADDR = 0 and PAODR = 1; otherwise off.

## 9.11 Change of Peripheral Function Pins

I/O ports that also function as peripheral modules, such as the external interrupts, 8-bit timers, and MCIF, can be changed.

I/O ports that also function as the external interrupt pins are changed according to the setting of ISSR16 and ISSR. I/O ports that also function as the 8-bit timer input pins and MCIF I/O pins are changed according to the setting of PTCNT0. The pin name of the peripheral function is indicated by adding 'Ex' at the head of the original pin name. In each peripheral function description, the original pin name is used.

### 9.11.1 IRQ Sense Port Select Register 16 (ISSR16), IRQ Sense Port Select Register (ISSR)

ISSR16 and ISSR select ports that also function as  $\overline{\text{IRQ}}_{15}$  to  $\overline{\text{IRQ}}_0$  input pins.

#### ISSR16

Bit	Bit Name	Initial Value	R/W	Description
15	ISS15	0	R/W	0: P57/ $\overline{\text{IRQ}}_{15}$ is selected 1: P87/ $\overline{\text{Ex}}\overline{\text{IRQ}}_{15}$ is selected
14	ISS14	0	R/W	0: P56/ $\overline{\text{IRQ}}_{14}$ is selected 1: P86/ $\overline{\text{Ex}}\overline{\text{IRQ}}_{14}$ is selected
13	ISS13	0	R/W	0: P55/ $\overline{\text{IRQ}}_{13}$ is selected 1: P85/ $\overline{\text{Ex}}\overline{\text{IRQ}}_{13}$ is selected
12	ISS12	0	R/W	0: P54/ $\overline{\text{IRQ}}_{12}$ is selected 1: P84/ $\overline{\text{Ex}}\overline{\text{IRQ}}_{12}$ is selected
11	ISS11	0	R/W	0: P53/ $\overline{\text{IRQ}}_{11}$ is selected 1: P83/ $\overline{\text{Ex}}\overline{\text{IRQ}}_{11}$ is selected
10	ISS10	0	R/W	0: P52/ $\overline{\text{IRQ}}_{10}$ is selected 1: P82/ $\overline{\text{Ex}}\overline{\text{IRQ}}_{10}$ is selected
9	ISS9	0	R/W	0: P51/ $\overline{\text{IRQ}}_9$ is selected 1: P81/ $\overline{\text{Ex}}\overline{\text{IRQ}}_9$ is selected
8	ISS8	0	R/W	0: P50/ $\overline{\text{IRQ}}_8$ is selected 1: P80/ $\overline{\text{Ex}}\overline{\text{IRQ}}_8$ is selected

**ISSR**

Bit	Bit Name	Initial Value	R/W	Description
7	ISS7	0	R/W	0: P47/ $\overline{\text{IRQ7}}$ is selected 1: P77/ $\overline{\text{ExIRQ7}}$ is selected
6	ISS6	0	R/W	0: P46/ $\overline{\text{IRQ6}}$ is selected 1: P76/ $\overline{\text{ExIRQ6}}$ is selected
5	ISS5	0	R/W	0: P45/ $\overline{\text{IRQ5}}$ is selected 1: P75/ $\overline{\text{ExIRQ5}}$ is selected
4	ISS4	0	R/W	0: P44/ $\overline{\text{IRQ4}}$ is selected 1: P74/ $\overline{\text{ExIRQ4}}$ is selected
3	ISS3	0	R/W	0: P43/ $\overline{\text{IRQ3}}$ is selected 1: P73/ $\overline{\text{ExIRQ3}}$ is selected
2	ISS2	0	R/W	0: P42/ $\overline{\text{IRQ2}}$ is selected 1: P72/ $\overline{\text{ExIRQ2}}$ is selected
1	ISS1	0	R/W	P41/ $\overline{\text{IRQ1}}$ is always selected
0	ISS0	0	R/W	P40/ $\overline{\text{IRQ0}}$ is always selected

### 9.11.2 Port Control Register 0 (PTCNT0)

PTCNT0 selects ports that also function as 8-bit timer input pins and MCIF I/O pins.

Bit	Bit Name	Initial Value	R/W	Description
7	TMI0S	0	R/W	0: P40/TMI0 is selected 1: P84/ExTMI0 is selected
6	TMI1S	0	R/W	0: P41/TMI1 is selected 1: P85/ExTMI1 is selected
5	TMIXS	0	R/W	0: P44/TMIX is selected 1: P86/ExTMIX is selected
4	TMIYS	0	R/W	0: P45/TMIY is selected 1: P87/ExTMIY is selected
3	MMCS	0	R/W	0: P30/MCCLK, P31/MCCMD/MCTxD, P32/MCDAT/MCRxD, P33/MCDATDIR/MCCSA, P34/MCCMDDIR/MCCSB are selected 1: P40/ExMCCLK, P41/ExMCCMD/ExMCTxD, P42/ExMCDAT/ExMCRxD, P43/ExMCDATDIR/ExMCCSA, P44/ExMCCMDDIR/ExMCCSB are selected  In extended mode, the MCIF pin function is multiplexed with port 4 regardless of the MMCS setting.
2 to 0	—	All 0	R(W)	Reserved  The initial value should not be changed.



## Section 10 8-Bit PWM Timer (PWM)

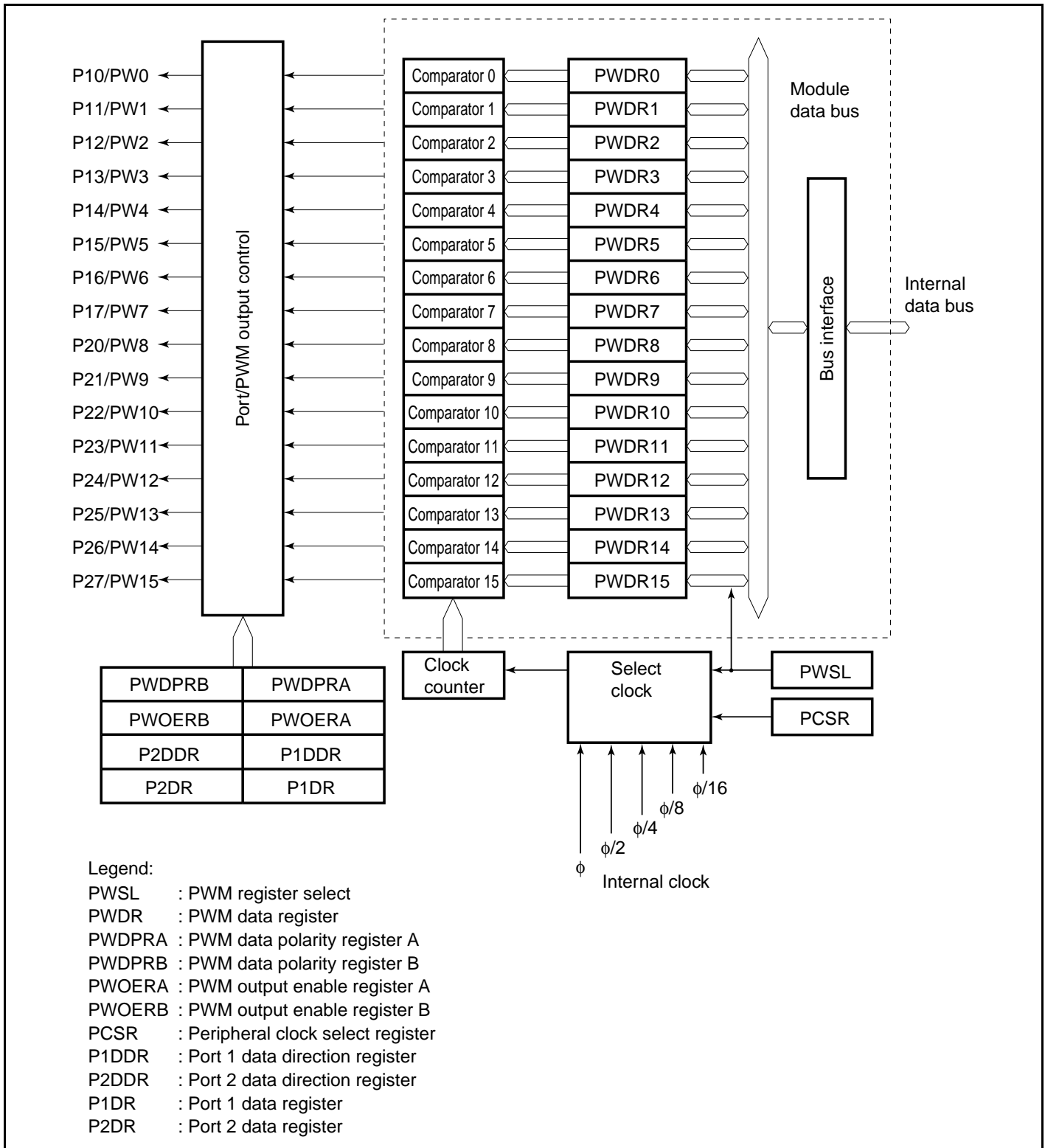
This LSI has an on-chip pulse width modulation (PWM) timer with sixteen outputs. Sixteen output waveforms are generated from a common time base, enabling PWM output with a high carrier frequency to be produced using pulse division.

### 10.1 Features

Operable at a maximum carrier frequency of 1.25 MHz using pulse division (at 20- MHz operation)

- Duty cycles from 0 to 100% with 1/256 resolution (100% duty realized by port output)
- Direct or inverted PWM output, and PWM output enable/disable control

Figure 10.1 shows a block diagram of the PWM timer.



**Figure 10.1 Block Diagram of PWM Timer**

## 10.2 Input/Output Pins

Table 10.1 shows the PWM output pins.

**Table 10.1 Pin Configuration**

<b>Name</b>	<b>Abbreviation</b>	<b>I/O</b>	<b>Function</b>
PWM output 15 to 0	PW15 to PW0	Output	PWM timer pulse output 15 to 0

## 10.3 Register Descriptions

The PWM has the following registers. To access PCSR, the FLSHE bit in the serial timer control register (STCR) must be cleared to 0. For details on the serial timer control register (STCR), see section 3.2.3, Serial Timer Control Register (STCR).

- PWM register select (PWSL)
- PWM data registers 0 to 15 (PWDR0 to PWDR15)
- PWM data polarity register A (PWPRA)
- PWM data polarity register B (PWPRB)
- PWM output enable register A (PWOERA)
- PWM output enable register B (PWOERB)
- Peripheral clock select register (PCSR)

### 10.3.1 PWM Register Select (PWSL)

PWSL is used to select the input clock and the PWM data register.

Bit	Bit Name	Initial Value	R/W	Description
7	PWCKE	0	R/W	PWM Clock Enable
6	PWCKS	0	R/W	PWM Clock Select
<p>These bits, together with bits PWCKB and PWCKA in PCSR, select the internal clock input to TCNT in the PWM. For details, see table 10.2.</p> <p>The resolution, PWM conversion period, and carrier frequency depend on the selected internal clock, and can be obtained from the following equations.</p> <p>Resolution (minimum pulse width) = <math>1/\text{internal clock frequency}</math></p> <p>PWM conversion period = resolution <math>\times</math> 256</p> <p>Carrier frequency = <math>16/\text{PWM conversion period}</math></p> <p>With a 20-MHz system clock (<math>\phi</math>), the resolution, PWM conversion period, and carrier frequency are as shown in table 10.3.</p>				
5	—	1	R	Reserved
<p>This bit is always read as 1 and cannot be modified.</p>				
4	—	0	R	Reserved
<p>This bit is always read as 0 and cannot be modified.</p>				

Bit	Bit Name	Initial Value	R/W	Description	
3	RS3	0	R/W	Register Select	
2	RS2	0	R/W	These bits select the PWM data register.	
1	RS1	0	R/W		
0	RS0	0	R/W		
					0000: PWDR0 selected
					0001: PWDR1 selected
					0010: PWDR2 selected
					0011: PWDR3 selected
					0100: PWDR4 selected
					0101: PWDR5 selected
					0110: PWDR6 selected
					0111: PWDR7 selected
					1000: PWDR8 selected
					1001: PWDR9 selected
					1010: PWDR10 selected
					1011: PWDR11 selected
					1100: PWDR12 selected
				1101: PWDR13 selected	
				1110: PWDR14 selected	
				1111: PWDR15 selected	

Table 10.2 Internal Clock Selection

PWSL		PCSR		Description	
PWCKE	PWCKS	PWCKB	PWCKA		
0	—	—	—	Clock input is disabled	(Initial value)
1	0	—	—	—	$\phi$ (system clock) is selected
			1	0	$\phi/2$ is selected
				1	$\phi/4$ is selected
	1		0	$\phi/8$ is selected	
			1	$\phi/16$ is selected	

**Table 10.3 Resolution, PWM Conversion Period, and Carrier Frequency when  $\phi = 20$  MHz**

Internal Clock Frequency	Resolution	PWM Conversion Period	Carrier Frequency
$\phi$	50 ns	12.8 $\mu$ s	1250 kHz
$\phi/2$	100 ns	25.6 $\mu$ s	625 kHz
$\phi/4$	200 ns	51.2 $\mu$ s	312.5 kHz
$\phi/8$	400 ns	102.4 $\mu$ s	156.3 kHz
$\phi/16$	800 ns	204.8 $\mu$ s	78.1 kHz

### 10.3.2 PWM Data Registers (PWDR0 to PWDR15)

PWDR are 8-bit readable/writable registers. The PWM has sixteen PWM data registers. Each PWDR specifies the duty cycle of the basic pulse to be output, and the number of additional pulses. The value set in PWDR corresponds to a 0 or 1 ratio in the conversion period. The upper four bits specify the duty cycle of the basic pulse as 0/16 to 15/16 with a resolution of 1/16. The lower four bits specify how many extra pulses are to be added within the conversion period comprising 16 basic pulses. Thus, a specification of 0/256 to 255/256 is possible for 0/1 ratios within the conversion period. For 256/256 (100%) output, port output should be used.

### 10.3.3 PWM Data Polarity Registers A and B (PWPRA and PWPBR)

Each PWPDR selects the PWM output phase.

#### PWPRA

Bit	Bit Name	Initial Value	R/W	Description
7	OS7	0	R/W	Output Select 7 to 0
6	OS6	0	R/W	These bits select the PWM output phase. Bits OS7 to OS0 correspond to outputs PW7 to PW0.
5	OS5	0	R/W	
4	OS4	0	R/W	0: PWM direct output (PWDR value corresponds to high width of output)
3	OS3	0	R/W	
2	OS2	0	R/W	1: PWM inverted output (PWDR value corresponds to low width of output)
1	OS1	0	R/W	
0	OS0	0	R/W	

**PWDPRB**

Bit	Bit Name	Initial Value	R/W	Description
7	OS15	0	R/W	Output Select 15 to 8
6	OS14	0	R/W	These bits select the PWM output phase. Bits OS15 to OS8 correspond to outputs PW15 to PW8.
5	OS13	0	R/W	
4	OS12	0	R/W	0: PWM direct output (PWDR value corresponds to high width of output)
3	OS11	0	R/W	
2	OS10	0	R/W	1: PWM inverted output (PWDR value corresponds to low width of output)
1	OS9	0	R/W	
0	OS8	0	R/W	

**10.3.4 PWM Output Enable Registers A and B (PWOERA and PWOERB)**

Each PWOER switches between PWM output and port output.

**PWOERA**

Bit	Bit Name	Initial Value	R/W	Description
7	OE7	0	R/W	Output Enable 7 to 0
6	OE6	0	R/W	These bits, together with P1DDR, specify the P1n/PWn pin state. Bits OE7 to OE0 correspond to outputs PW7 to PW0.
5	OE5	0	R/W	
4	OE4	0	R/W	P1nDDR OEn: Pin state
3	OE3	0	R/W	
2	OE2	0	R/W	0X: Port input
1	OE1	0	R/W	
0	OE0	0	R/W	10: Port output or PWM 256/256 output
				11: PWM output (0 to 255/256 output)

Legend:

X: Don't care

**PWOERB**

Bit	Bit Name	Initial Value	R/W	Description
7	OE15	0	R/W	Output Enable 15 to 8
6	OE14	0	R/W	These bits, together with P2DDR, specify the P2n/PWn pin state. Bits OE15 to OE8 correspond to outputs PW15 to PW8.
5	OE13	0	R/W	
4	OE12	0	R/W	P2nDDR OEn: Pin state
3	OE11	0	R/W	
2	OE10	0	R/W	0X: Port input
1	OE9	0	R/W	
0	OE8	0	R/W	10: Port output or PWM 256/256 output
				11: PWM output (0 to 255/256 output)

Legend:

X: Don't care

To perform PWM 256/256 output when DDR = 1 and OE = 0, the corresponding pin should be set to port output. The corresponding pin can be set as port output in single-chip mode or when IOSE = 1 and CS256E = 0 in SYSCR in extended mode with on-chip ROM. Otherwise, it should be noted that an address bus is output to the corresponding pin.

DR data is output when the corresponding pin is used as port output. A value corresponding to PWM 256/256 output is determined by the OS bit, so the value should have been set to DR beforehand.



### 10.3.5 Peripheral Clock Select Register (PCSR)

PCSR selects the PWM input clock.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R/(W)	Reserved The initial value should not be changed.
5	PWCKXB	0	R/W	See section 11.3.4, Peripheral Clock Select Register (PCSR).
4	PWCKXA	0	R/W	
3	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
2	PWCKB	0	R/W	PWM Clock Select B, A Together with bits PWCKE and PWCKS in PWSL, these bits select the internal clock input to the clock counter. For details, see table 10.2.
1	PWCKA	0	R/W	
0	—	0	R/(W)	Reserved The initial value should not be changed.

## 10.4 Operation

The upper four bits of PWDR specify the duty cycle of the basic pulse as 0/16 to 15/16 with a resolution of 1/16. Table 10.4 shows the duty cycles of the basic pulse.

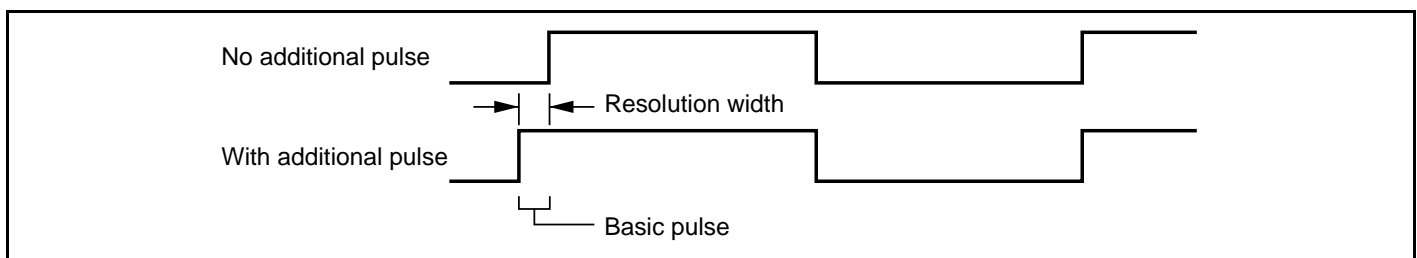
**Table 10.4 Duty Cycle of Basic Pulse**

Upper 4 Bits	Basic Pulse Waveform (Internal)
B'0000	0 1 2 3 4 5 6 7 8 9 A B C D E F 0
B'0001	
B'0010	
B'0011	
B'0100	
B'0101	
B'0110	
B'0111	
B'1000	
B'1001	
B'1010	
B'1011	
B'1100	
B'1101	
B'1110	
B'1111	

The lower four bits of PWDR specify the position of pulses added to the 16 basic pulses. An additional pulse adds a high period (when OS = 0) with a width equal to the resolution before the rising edge of a basic pulse. When the upper four bits of PWDR are B'0000, there is no rising edge of the basic pulse, but the timing for adding pulses is the same. Table 10.5 shows the positions of the additional pulses added to the basic pulses, and figure 10.2 shows an example of additional pulse timing.

**Table 10.5 Position of Pulses Added to Basic Pulses**

Lower 4 Bits	Basic Pulse No.															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
B'0000																
B'0001																Yes
B'0010								Yes								Yes
B'0011								Yes				Yes				Yes
B'0100				Yes				Yes				Yes				Yes
B'0101				Yes				Yes				Yes		Yes		Yes
B'0110				Yes		Yes		Yes				Yes		Yes		Yes
B'0111				Yes		Yes		Yes		Yes		Yes		Yes		Yes
B'1000		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes
B'1001		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes
B'1010		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes
B'1011		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
B'1100		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
B'1101		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
B'1110		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
B'1111		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes



**Figure 10.2 Example of Additional Pulse Timing (When Upper 4 Bits of PWDR = B'1000)**



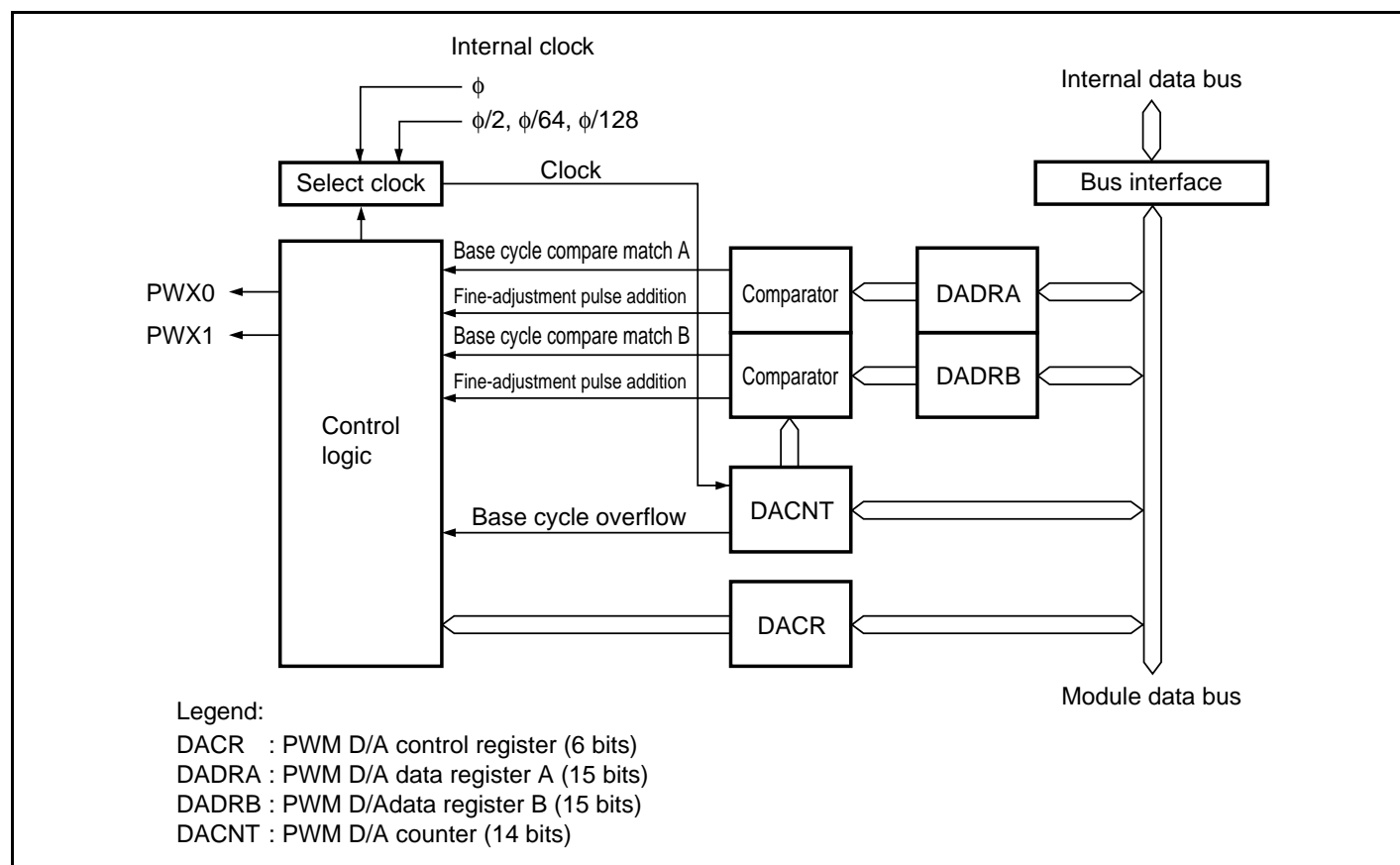
## Section 11 14-Bit PWM Timer (PWMX)

This LSI has an on-chip 14-bit pulse-width modulator (PWM) timer with two output channels. It can be connected to an external low-pass filter to operate as a 14-bit D/A converter.

### 11.1 Features

- Division of pulse into multiple base cycles to reduce ripple
- Four resolution settings  
The resolution can be set equal to one, two, 64, or 128 system clock cycles.
- Two base cycle settings  
The base cycle can be set equal to  $T \times 64$  or  $T \times 256$ , where  $T$  is the resolution.
- Eight operating speeds
- Eight operation clocks (by combination of four resolution settings and two base cycle settings)

Figure 11.1 shows a block diagram of the PWM (D/A) module.



**Figure 11.1 PWM (D/A) Block Diagram**

## 11.2 Input/Output Pins

Table 11.1 lists the PWM (D/A) module input and output pins.

**Table 11.1 Pin Configuration**

<b>Name</b>	<b>Abbreviation</b>	<b>I/O</b>	<b>Function</b>
PWM output pin X0	PWX0	Output	PWM output of PWMX channel A
PWM output pin X1	PWX1	Output	PWM output of PWMX channel B

## 11.3 Register Descriptions

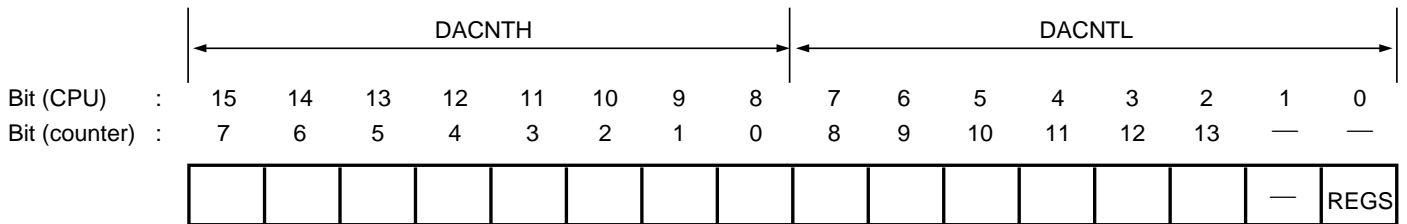
The PWM (D/A) module has the following registers. The PWM (D/A) registers are assigned to the same addresses with other registers. The registers are selected by the IICE bit in the serial timer control register (STCR). To access PCSR, the FLSHE bit in STCR must be cleared to 0. For details on STCR, see section 3.2.3, Serial Timer Control Register (STCR).

- PWM (D/A) counter H (DACNTH)
- PWM (D/A) counter L (DACNTL)
- PWM (D/A) data register AH (DADRAH)
- PWM (D/A) data register AL (DADRAL)
- PWM (D/A) data register BH (DADRBH)
- PWM (D/A) data register BL (DADRBL)
- PWM (D/A) control register (DACR)
- Peripheral clock select register (PCSR)

Note: The same addresses are shared by DADRA and DACR, and by DADRB and DACNT. Switching is performed by the REGS bit in DACNT or DADRB.

### 11.3.1 PWM D/A Counter H, L (DACNTH, DACNTL)

DACNT is a 14-bit readable/writable up-counter. The input clock is selected by the clock select bit (CKS) in DACR. DACNT functions as the time base for both PWM (D/A) channels. When a channel operates with 14-bit precision, it uses all DACNT bits. When a channel operates with 12-bit precision, it uses the lower 12 bits and ignores the upper two bits. DACNT cannot be accessed in 8-bit units. DACNT should always be accessed in 16-bit units. For details, see section 11.4, Bus Master Interface.



#### DACNTH

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	UC7 to UC0	All 0	R/W	Upper Up-Counter

#### DACNTL

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	UC8 to UC13	All 0	R/W	Lower Up-Counter
1	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
0	REGS	1	R/W	Register Select DADRA and DACR, and DADR B and DACNT, are located at the same addresses. The REGS bit specifies which registers can be accessed. 0: DADRA and DADR B can be accessed 1: DACR and DACNT can be accessed

### 11.3.2 PWM (D/A) Data Registers A and B (DADRA and DADRB)

DADRA corresponds to PWM (D/A) channel A, and DADRB to PWM (D/A) channel B. The DADR registers cannot be accessed in 8-bit units. The DADR registers should always be accessed in 16-bit units. For details, see section 11.4, Bus Master Interface.

#### DADRA

Bit	Bit Name	Initial Value	R/W	Description
15	DA13	1	R/W	D/A Data 13 to 0
14	DA12	1	R/W	These bits set a digital value to be converted to an analog value.
13	DA11	1	R/W	
12	DA10	1	R/W	In each base cycle, the DACNT value is continually compared with the DADR value to determine the duty cycle of the output waveform, and to decide whether to output a fine-adjustment pulse equal in width to the resolution. To enable this operation, this register must be set within a range that depends on the CFS bit. If the DADR value is outside this range, the PWM output is held constant.
11	DA9	1	R/W	
10	DA8	1	R/W	
9	DA7	1	R/W	
8	DA6	1	R/W	
7	DA5	1	R/W	
6	DA4	1	R/W	
5	DA3	1	R/W	
4	DA2	1	R/W	
3	DA1	1	R/W	
2	DA0	1	R/W	A channel can be operated with 12-bit precision by keeping the two lowest data bits (DA1 and DA0) cleared to 0. The two lowest data bits correspond to the two highest bits in DACNT.
1	CFS	1	R/W	Carrier Frequency Select 0: Base cycle = resolution (T) × 64 DADR range = H'0401 to H'FFFD 1: Base cycle = resolution (T) × 256 DADR range = H'0103 to H'FFFF
0	—	1	R	Reserved This bit is always read as 1 and cannot be modified.



**DADRB**

Bit	Bit Name	Initial Value	R/W	Description
15	DA13	1	R/W	D/A Data 13 to 0
14	DA12	1	R/W	These bits set a digital value to be converted to an analog value.
13	DA11	1	R/W	
12	DA10	1	R/W	In each base cycle, the DACNT value is continually compared with the DADR value to determine the duty cycle of the output waveform, and to decide whether to output a fine-adjustment pulse equal in width to the resolution. To enable this operation, this register must be set within a range that depends on the CFS bit. If the DADR value is outside this range, the PWM output is held constant.
11	DA9	1	R/W	
10	DA8	1	R/W	
9	DA7	1	R/W	
8	DA6	1	R/W	
7	DA5	1	R/W	
6	DA4	1	R/W	
5	DA3	1	R/W	
4	DA2	1	R/W	
3	DA1	1	R/W	
2	DA0	1	R/W	A channel can be operated with 12-bit precision by keeping both the DA0 and DA1 bits cleared to 0. This 2-bit data is not compared with the UC12 and UC13 bits in DACNT.
1	CFS	1	R/W	Carrier Frequency Select 0: Base cycle = Resolution (T) × 64 Range of values in DA13 to DA0 = H'0100 to H'3FFF 1: Base cycle = Resolution (T) × 256 Range of values in DA13 to DA0 = H'0040 to H'3FFF
0	REGS	1	R/W	Register Select DADRA and DACR, and DADRB and DACNT, are located at the same addresses. This bit specifies which registers can be accessed. Make this bit setting before changing the address register. 0: DADRA and DADRB can be accessed 1: DACR and DACNT can be accessed

**11.3.3 PWM (D/A) Control Register (DACR)**

DACR selects test mode, enables the PWM outputs, and selects the output phase and operating speed.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/(W)	Reserved The initial value should not be changed.
6	PWME	0	R/W	PWM Enable Starts or stops the PWM D/A counter (DACNT). 0: DACNT operates as a 14-bit up-counter 1: DACNT halts at H'0003
5, 4	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.
3	OEB	0	R/W	Output Enable B Enables or disables output on PWM (D/A) channel B. 0: PWM (D/A) channel B output (at the PWX1 pin) is disabled 1: PWM (D/A) channel B output (at the PWX1 pin) is enabled
2	OEA	0	R/W	Output Enable A Enables or disables output on PWM (D/A) channel A. 0: PWM (D/A) channel A output (at the PWX0 pin) is disabled 1: PWM (D/A) channel A output (at the PWX0 pin) is enabled
1	OS	0	R/W	Output Select Selects the phase of the PWM (D/A) output. 0: Direct PWM (D/A) output 1: Inverted PWM (D/A) output
0	CKS	0	R/W	Clock Select Selects the PWM (D/A) resolution. If the system clock ( $\phi$ ) frequency is 10 MHz, resolutions of 100 ns, 200 ns, 6,400 ns, and 12,800 ns can be selected. 0: Operates at resolution (T) = system clock cycle time ( $t_{cyc}$ ) 1: Operates at resolution (T) = system clock cycle time ( $t_{cyc}$ ) $\times 2$ , $\times 64$ , and $\times 128$

### 11.3.4 Peripheral Clock Select Register (PCSR)

PCSR selects the operating speed of DACR.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R/(W)	Reserved The initial value should not be changed.
5	PWCKXB	0	R/W	PWMX Clock Select
4	PWCKXA	0	R/W	Select the clock when the CKS bit in DACR of PWMX is set to 1. 00: Operates at resolution (T) = system clock cycle time ( $t_{cyc}$ ) × 2 01: Operates at resolution (T) = system clock cycle time ( $t_{cyc}$ ) × 64 10: Operates at resolution (T) = system clock cycle time ( $t_{cyc}$ ) × 128 11: Setting prohibited
3	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
2	PWCKB	0	R/W	See section 10.3.5, Peripheral Clock Select Register (PCSR).
1	PWCKA	0	R/W	
0	—	0	R/(W)	Reserved The initial value should not be changed.

## 11.4 Bus Master Interface

DACNT, DADRA, and DADRB are 16-bit registers. The data bus linking the bus master and the on-chip peripheral modules, however, is only 8 bits wide. When the bus master accesses these registers, it therefore uses an 8-bit temporary register (TEMP).

These registers are written to and read from as follows.

- **Write**  
When the upper byte is written to, the upper-byte write data is stored in TEMP. Next, when the lower byte is written to, the lower-byte write data and TEMP value are combined, and the combined 16-bit value is written in the register.
- **Read**  
When the upper byte is read from, the upper-byte value is transferred to the CPU and the lower-byte value is transferred to TEMP. Next, when the lower byte is read from, the lower-byte value in TEMP is transferred to the CPU.

These registers should always be accessed 16 bits at a time with a MOV instruction, and the upper byte should always be accessed before the lower byte. Correct data will not be transferred if only the upper byte or only the lower byte is accessed. Also note that a bit manipulation instruction cannot be used to access these registers.

Example 1: Write to DACNT

```
MOV.W R0, @DACNT ; Write R0 contents to DACNT
```

Example 2: Read DADRA

```
MOV.W @DADRA, R0 ; Copy contents of DADRA to R0
```

**Table 11.2 Read and Write Access Methods for 16-Bit Registers**

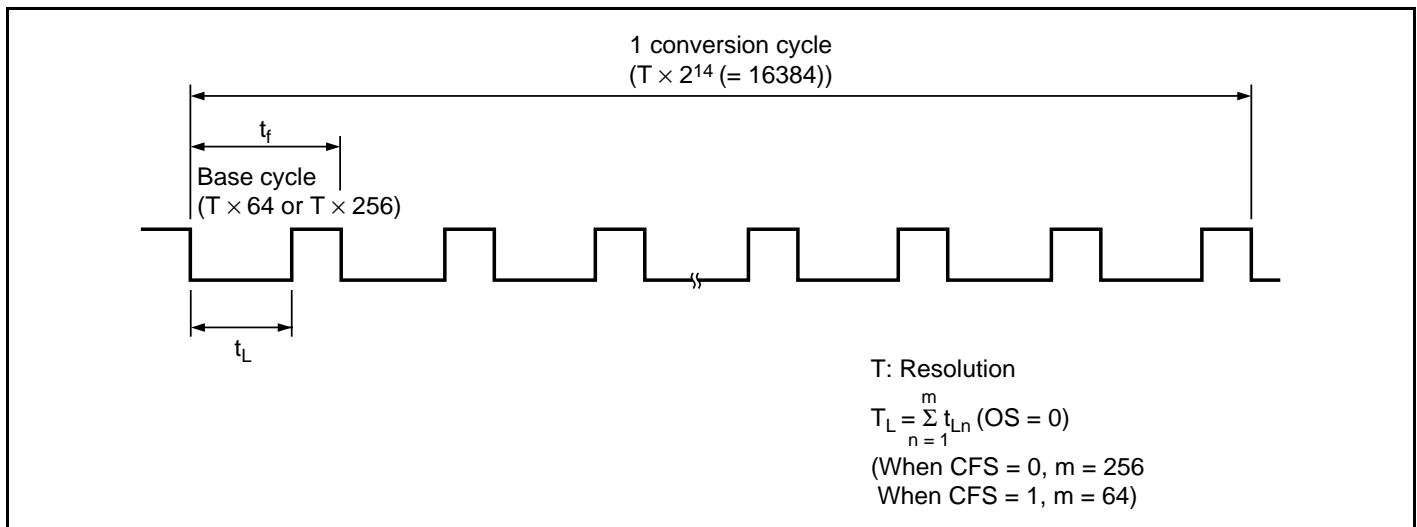
Register Name	Read		Write	
	Word	Byte	Word	Byte
DADRA and DADRB	Yes	Yes	Yes	×
DACNT	Yes	×	Yes	×

Legend: Yes: Permitted type of access. Word access includes successive byte accesses to the upper byte (first) and lower byte (second).

×: This type of access may give incorrect results.

## 11.5 Operation

A PWM waveform like the one shown in figure 11.2 is output from the PWMX pin. The value in DADR corresponds to the total width ( $T_L$ ) of the low (0) pulses output in one conversion cycle (256 pulses when CFS = 0, 64 pulses when CFS = 1). When OS = 0, this waveform is directly output. When OS = 1, the output waveform is inverted, and the DADR value corresponds to the total width ( $T_H$ ) of the high (1) output pulses. Figures 11.3 and 11.4 show the types of waveform output available.



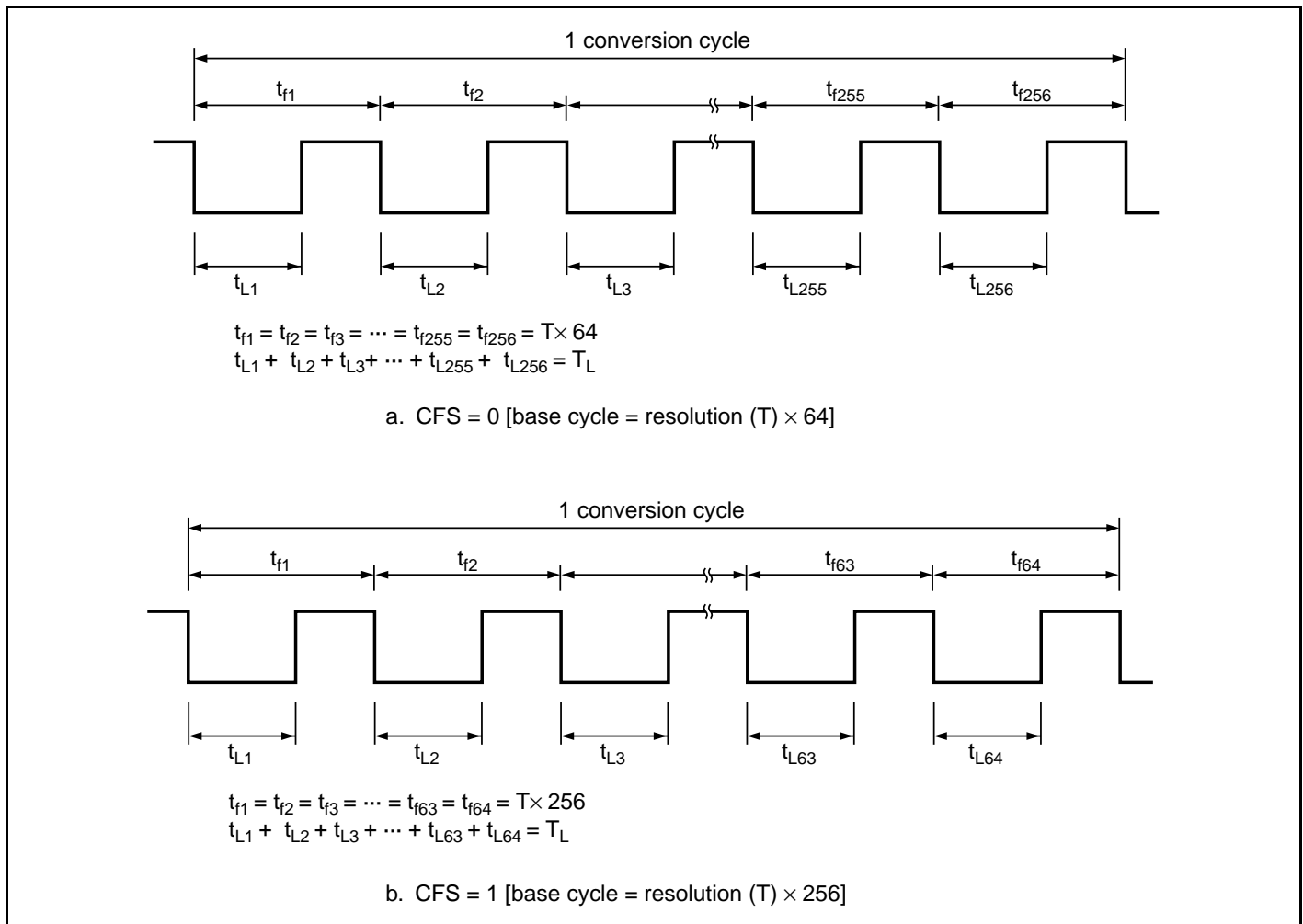
**Figure 11.2 PWM (D/A) Operation**

Table 11.3 summarizes the relationships between the CKS, CFS, and OS bit settings and the resolution, base cycle, and conversion cycle. The PWM output remains fixed unless DADR contains at least a certain minimum value.

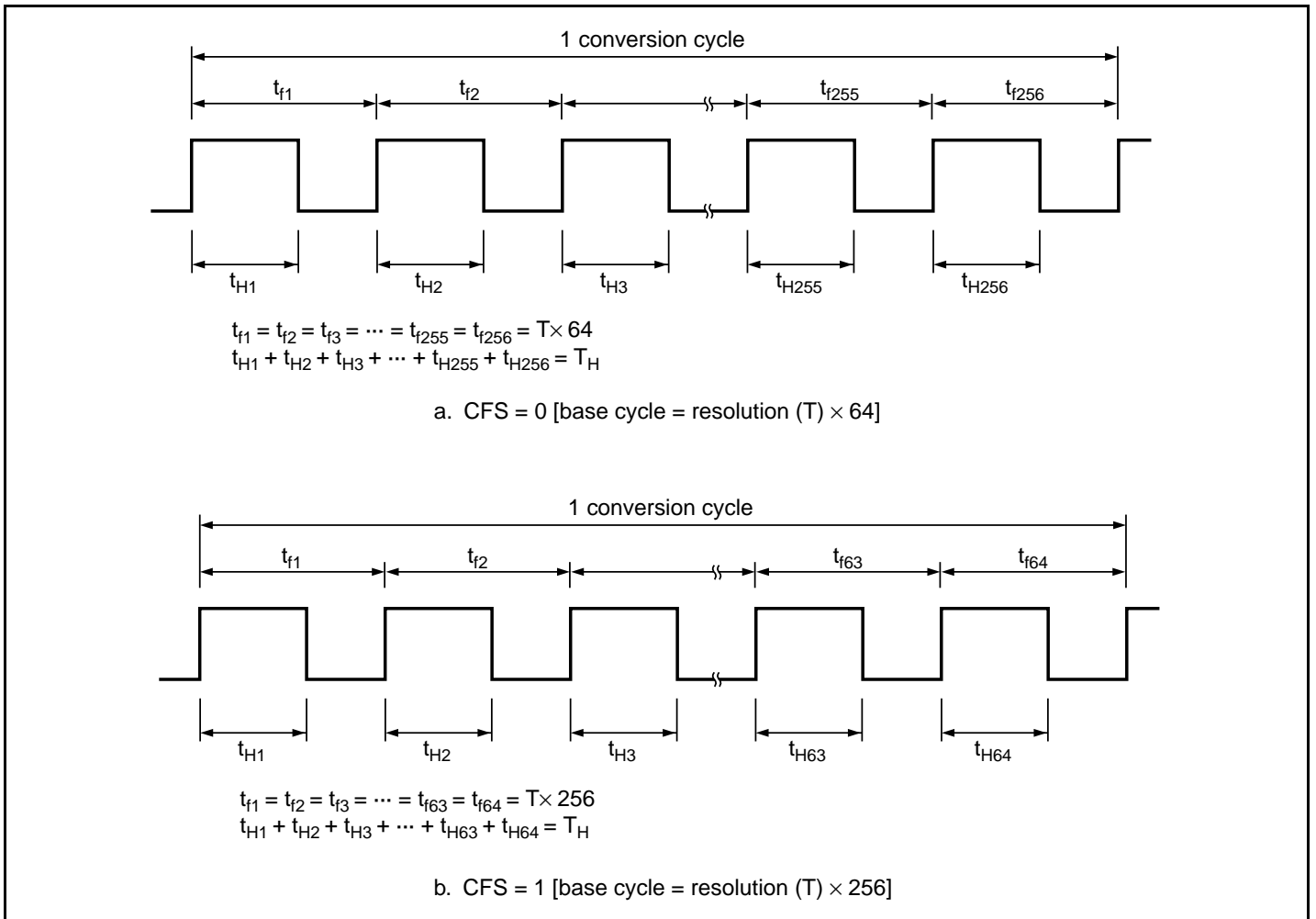
Table 11.3 Settings and Operation (Examples when  $\phi = 25$  MHz)

PCSR		CKS	Resolution T ( $\mu$ s)	CFS	Base Cycle ( $\mu$ s)/ Frequency	Conversion Cycle ( $\mu$ s)/ Frequency	TL/TH (OS = 0/OS = 1)	Fixed DADR Bits				Conversion Cycle* ( $\mu$ s)	
PWCKXB	PWCKXA							Conversion Accuracy (Bits)	Bit Data				
									3	2	1		0
—	—	0	0.04	0	2.56/ 391 kHz	655.36/ 1.53 kHz	1. Always low/high output (DADR = H'0001 to H'03FD) 2. (Data value) $\times$ T (DADR = H'0401 to H'FFFD)	14					655.36
								12			0	0	163.84
								10	0	0	0	0	40.96
								14					655.36
								12			0	0	163.84
								10	0	0	0	0	40.96
0	0	1	0.08	0	5.12/ 195 kHz	1310.72/ 763 Hz	1. Always low/high output (DADR = H'0001 to H'03FD) 2. (Data value) $\times$ T (DADR = H'0401 to H'FFFD)	14					1310.72
								12			0	0	327.68
								10	0	0	0	0	81.92
								14					1310.72
								12			0	0	327.68
								10	0	0	0	0	81.92
0	1	1	2.56	0	163.84/ 6.10 kHz	41.943/ 23.84 Hz	1. Always low/high output (DADR = H'0001 to H'03FD) 2. (Data value) $\times$ T (DADR = H'0401 to H'FFFD)	14					41.943
								12			0	0	10.486
								10	0	0	0	0	2.621
								14					41.943
								12			0	0	10.486
								10	0	0	0	0	2.621
1	0	1	5.12	0	327.68/ 3.05 kHz	83.886/ 11.92 Hz	1. Always low/high output (DADR = H'0001 to H'03FD) 2. (Data value) $\times$ T (DADR = H'0401 to H'FFFD)	14					83.886
								12			0	0	20.972
								10	0	0	0	0	5.243
								14					83.886
								12			0	0	20.972
								10	0	0	0	0	5.243

Note: \* Indicates the conversion cycle when specific DADR bits are fixed.



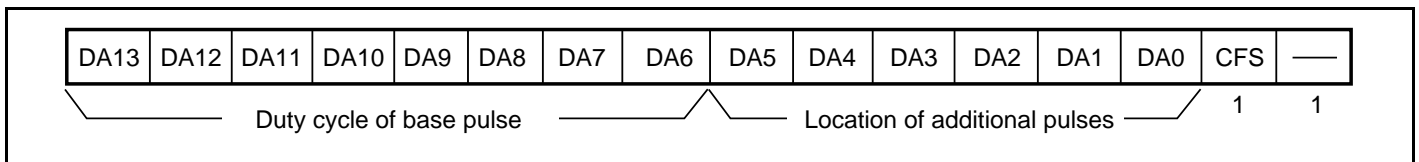
**Figure 11.3 Output Waveform (OS = 0, DADR corresponds to  $T_L$ )**



**Figure 11.4 Output Waveform (OS = 1, DADR corresponds to TH)**

An example of the additional pulses when CFS = 1 (base cycle = resolution (T) × 256) and OS = 1 (inverted PWMX output) is described below. When CFS = 1, the upper eight bits (DA13 to DA6) in DADR determine the duty cycle of the base pulse while the subsequent six bits (DA5 to DA0) determine the locations of the additional pulses as shown in figure 11.5.

Table 11.4 lists the locations of the additional pulses.

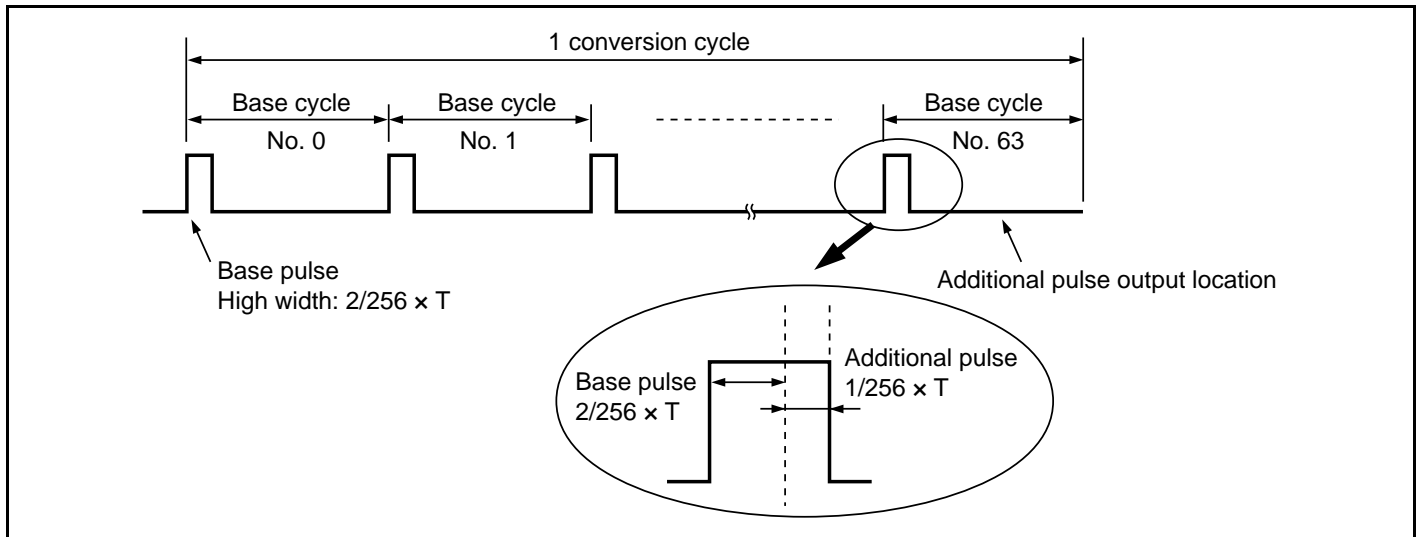


**Figure 11.5 D/A Data Register Configuration when CFS = 1**



In this example, DADR = H'0207 (B'0000 0010 0000 0111). The output waveform is shown in figure 11.6. Since CFS = 1 and the value of the upper eight bits is B'0000 0010, the high width of the base pulse duty cycle is  $2/256 \times T$ .

Since the value of the subsequent six bits is B'0000 01, an additional pulse is output only at the location of base pulse No. 63 according to table 11.4. Thus, an additional pulse of  $1/256 \times T$  is to be added to the base pulse.



**Figure 11.6 Output Waveform when DADR = H'0207 (OS = 1)**

However, when CFS = 0 (base cycle = resolution (T)  $\times$  64), the duty cycle of the base pulse is determined by the upper six bits and the locations of the additional pulses by the subsequent eight bits with a method similar to as above.



## Section 12 16-Bit Free-Running Timer (FRT)

This LSI has an on-chip 16-bit free-running timer (FRT). The FRT operates on the basis of the 16-bit free-running counter (FRC), and outputs two independent waveforms, and measures the input pulse width and external clock periods.

### 12.1 Features

- Selection of four clock sources
  - One of the three internal clocks ( $\phi/2$ ,  $\phi/8$ , or  $\phi/32$ ), or an external clock input can be selected (enabling use as an external event counter).
- Two independent comparators
  - Two independent waveforms can be output.
- Four independent input capture channels
  - The rising or falling edge can be selected.
  - Buffer modes can be specified.
- Counter clearing
  - The free-running counters can be cleared on compare-match A.
- Seven independent interrupts
  - Two compare-match interrupts, four input capture interrupts, and one overflow interrupt can be requested independently.
- Special functions provided by automatic addition function
  - The contents of OCRAR and OCRAF can be added to the contents of OCRA automatically, enabling a periodic waveform to be generated without software intervention. The contents of ICRD can be added automatically to the contents of OCRDM  $\times 2$ , enabling input capture operations in this interval to be restricted.

Figure 12.1 shows a block diagram of the FRT.

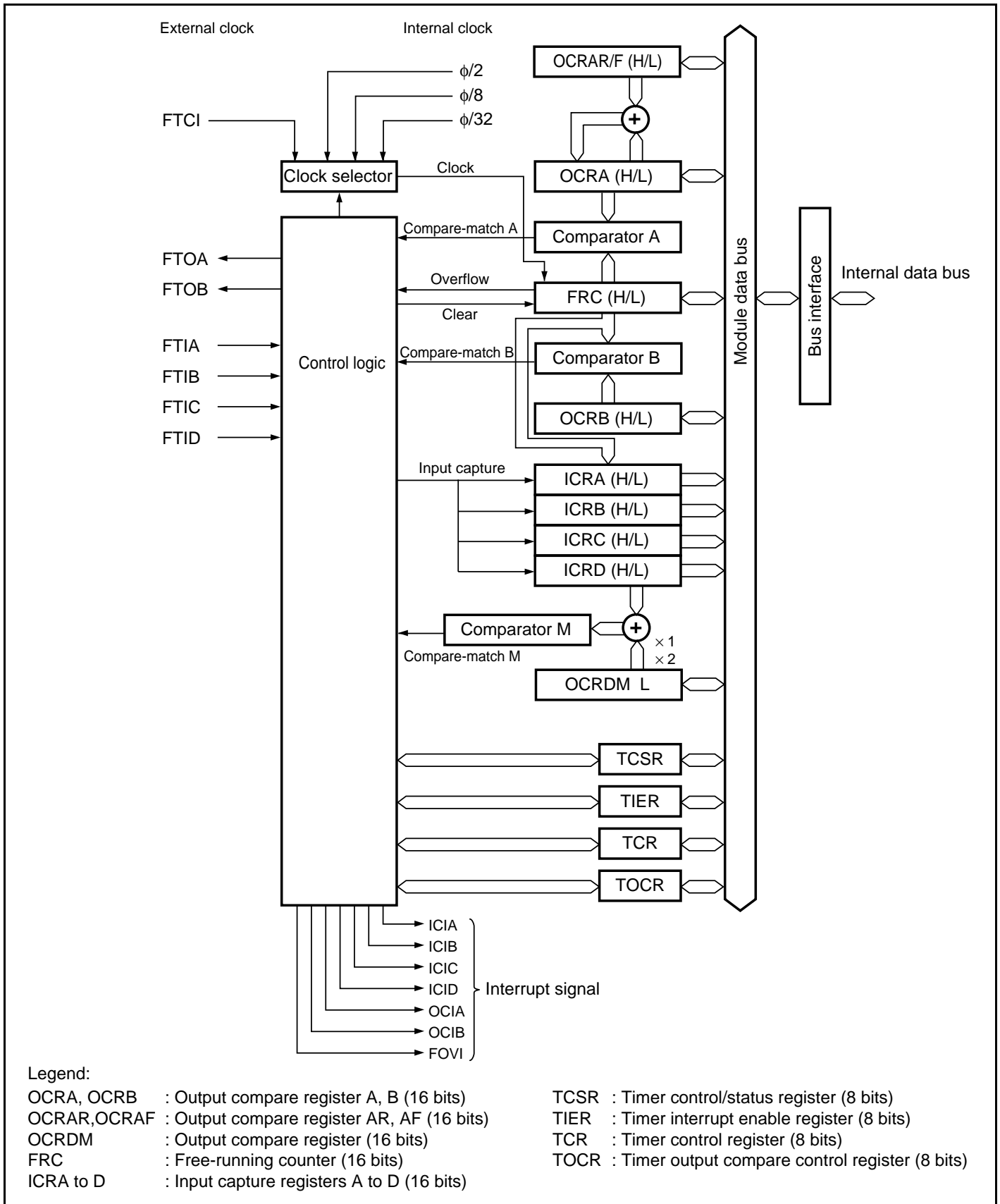


Figure 12.1 Block Diagram of 16-Bit Free-Running Timer

## 12.2 Input/Output Pins

Table 12.1 lists the FRT input and output pins.

**Table 12.1 Pin Configuration**

Name	Abbreviation	I/O	Function
Counter clock input pin	FTCI	Input	FRC counter clock input
Output compare A output pin	FTOA	Output	Output compare A output
Output compare B output pin	FTOB	Output	Output compare B output
Input capture A input pin	FTIA	Input	Input capture A input
Input capture B input pin	FTIB	Input	Input capture B input
Input capture C input pin	FTIC	Input	Input capture C input
Input capture D input pin	FTID	Input	Input capture D input

## 12.3 Register Descriptions

The FRT has the following registers.

- Free-running counter (FRC)
- Output compare register A (OCRA)
- Output compare register B (OCRB)
- Input capture register A (ICRA)
- Input capture register B (ICRB)
- Input capture register C (ICRC)
- Input capture register D (ICRD)
- Output compare register AR (OCRAR)
- Output compare register AF (OCRAF)
- Output compare register DM (OCRDM)
- Timer interrupt enable register (TIER)
- Timer control/status register (TCSR)
- Timer control register (TCR)
- Timer output compare control register (TOCR)

Note: OCRA and OCRB share the same address. Register selection is controlled by the OCRS bit in TOCR. ICRA, ICRB, and ICRC share the same addresses with OCRAR, OCRAF, and OCRDM. Register selection is controlled by the ICRS bit in TOCR.

### 12.3.1 Free-Running Counter (FRC)

FRC is a 16-bit readable/writable up-counter. The clock source is selected by bits CKS1 and CKS0 in TCR. FRC can be cleared by compare-match A. When FRC overflows from H'FFFF to H'0000, the overflow flag bit (OVF) in TCSR is set to 1. FRC should always be accessed in 16-bit units; cannot be accessed in 8-bit units. FRC is initialized to H'0000.

### 12.3.2 Output Compare Registers A and B (OCRA and OCRB)

The FRT has two output compare registers, OCRA and OCRB, each of which is a 16-bit readable/writable register whose contents are continually compared with the value in FRC. When a match is detected (compare-match), the corresponding output compare flag (OCFA or OCFB) is set to 1 in TCSR. If the OEA or OEB bit in TOCR is set to 1, when the OCR and FRC values match, the output level selected by the OLVLA or OLVLB bit in TOCR is output at the output compare output pin (FTOA or FTOB). Following a reset, the FTOA and FTOB output levels are 0 until the first compare-match. OCR should always be accessed in 16-bit units; cannot be accessed in 8-bit units. OCR is initialized to H'FFFF.

### 12.3.3 Input Capture Registers A to D (ICRA to ICRD)

The FRT has four input capture registers, ICRA to ICRD, each of which is a 16-bit read-only register. When the rising or falling edge of the signal at an input capture input pin (FTIA to FTID) is detected, the current FRC value is transferred to the corresponding input capture register (ICRA to ICRD). At the same time, the corresponding input capture flag (ICFA to ICFD) in TCSR is set to 1. The FRC contents are transferred to ICR regardless of the value of ICF. The input capture edge is selected by the input edge select bits (IEDGA to IEDGD) in TCR.

ICRC and ICRD can be used as ICRA and ICRB buffer registers, respectively, by means of buffer enable bits A and B (BUFEA and BUFEB) in TCR. For example, if an input capture occurs when ICRC is specified as the ICRA buffer register, the FRC contents are transferred to ICRA, and then transferred to the buffer register ICRC.

To ensure input capture, the input capture pulse width should be at least 1.5 system clocks ( $\phi$ ) for a single edge. When triggering is enabled on both edges, the input capture pulse width should be at least 2.5 system clocks ( $\phi$ ).

ICRA to ICRD should always be accessed in 16-bit units; cannot be accessed in 8-bit units. ICR is initialized to H'0000.

### 12.3.4 Output Compare Registers AR and AF (OCRAR and OCRAF)

OCRAR and OCRAF are 16-bit readable/writable registers. When the OCRAMS bit in TOCR is set to 1, the operation of OCRA is changed to include the use of OCRAR and OCRAF. The contents of OCRAR and OCRAF are automatically added alternately to OCRA, and the result is written to OCRA. The write operation is performed on the occurrence of compare-match A. In the 1st compare-match A after setting the OCRAMS bit to 1, OCRAF is added. The operation due to compare-match A varies according to whether the compare-match follows addition of OCRAR or OCRAF. The value of the OLVLA bit in TOCR is ignored, and 1 is output on a compare-match A following addition of OCRAF, while 0 is output on a compare-match A following addition of OCRAR.

When using the OCRA automatic addition function, do not select internal clock  $\phi/2$  as the FRC input clock together with a set value of H'0001 or less for OCRAR (or OCRAF).

OCRAR and OCRAF should always be accessed in 16-bit units; cannot be accessed in 8-bit units. OCRAR and OCRAF are initialized to H'FFFF.

### 12.3.5 Output Compare Register DM (OCRDM)

OCRDM is a 16-bit readable/writable register in which the upper 8 bits are fixed at H'00. When the ICRDMS bit in TOCR is set to 1 and the contents of OCRDM are other than H'0000, the operation of ICRD is changed to include the use of OCRDM. The point at which input capture D occurs is taken as the start of a mask interval. Next, twice the contents of OCRDM is added to the contents of ICRD, and the result is compared with the FRC value. The point at which the values match is taken as the end of the mask interval. New input capture D events are disabled during the mask interval. A mask interval is not generated when the contents of OCRDM are H'0000 while the ICRDMS bit is set to 1.

OCRDM should always be accessed in 16-bit units; cannot be accessed in 8-bit units. OCRDM is initialized to H'0000.

### 12.3.6 Timer Interrupt Enable Register (TIER)

TIER enables and disables interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7	ICIAE	0	R/W	<p>Input Capture Interrupt A Enable</p> <p>Selects whether to enable input capture interrupt A request (ICIA) when input capture flag A (ICFA) in TCSR is set to 1.</p> <p>0: ICIA requested by ICFA is disabled</p> <p>1: ICIA requested by ICFA is enabled</p>
6	ICIBE	0	R/W	<p>Input Capture Interrupt B Enable</p> <p>Selects whether to enable input capture interrupt B request (ICIB) when input capture flag B (ICFB) in TCSR is set to 1.</p> <p>0: ICIB requested by ICFB is disabled</p> <p>1: ICIB requested by ICFB is enabled</p>
5	ICICE	0	R/W	<p>Input Capture Interrupt C Enable</p> <p>Selects whether to enable input capture interrupt C request (ICIC) when input capture flag C (ICFC) in TCSR is set to 1.</p> <p>0: ICIC requested by ICFC is disabled</p> <p>1: ICIC requested by ICFC is enabled</p>
4	ICIDE	0	R/W	<p>Input Capture Interrupt D Enable</p> <p>Selects whether to enable input capture interrupt D request (ICID) when input capture flag D (ICFD) in TCSR is set to 1.</p> <p>0: ICID requested by ICFD is disabled</p> <p>1: ICID requested by ICFD is enabled</p>
3	OCIAE	0	R/W	<p>Output Compare Interrupt A Enable</p> <p>Selects whether to enable output compare interrupt A request (OCIA) when output compare flag A (OCFA) in TCSR is set to 1.</p> <p>0: OCIA requested by OCFA is disabled</p> <p>1: OCIA requested by OCFA is enabled</p>



Bit	Bit Name	Initial Value	R/W	Description
2	OCIBE	0	R/W	Output Compare Interrupt B Enable Selects whether to enable output compare interrupt B request (OCIB) when output compare flag B (OCFB) in TCSR is set to 1. 0: OCIB requested by OCFB is disabled 1: OCIB requested by OCFB is enabled
1	OVIE	0	R/W	Timer Overflow Interrupt Enable Selects whether to enable a free-running timer overflow request interrupt (FOVI) when the timer overflow flag (OVF) in TCSR is set to 1. 0: FOVI requested by OVF is disabled 1: FOVI requested by OVF is enabled
0	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

### 12.3.7 Timer Control/Status Register (TCSR)

TCSR is used for counter clear selection and control of interrupt request signals.

Bit	Bit Name	Initial Value	R/W	Description
7	ICFA	0	R/(W)*	Input Capture Flag A This status flag indicates that the FRC value has been transferred to ICRA by means of an input capture signal. When BUFEA = 1, ICFA indicates that the old ICRA value has been moved into ICRC and the new FRC value has been transferred to ICRA. [Setting condition] When an input capture signal causes the FRC value to be transferred to ICRA [Clearing condition] Read ICFA when ICFA = 1, then write 0 to ICFA

Bit	Bit Name	Initial Value	R/W	Description
6	ICFB	0	R/(W)*	<p>Input Capture Flag B</p> <p>This status flag indicates that the FRC value has been transferred to ICRB by means of an input capture signal. When BUFEB = 1, ICFB indicates that the old ICRB value has been moved into ICRD and the new FRC value has been transferred to ICRB.</p> <p>[Setting condition]</p> <p>When an input capture signal causes the FRC value to be transferred to ICRB</p> <p>[Clearing condition]</p> <p>Read ICFB when ICFB = 1, then write 0 to ICFB</p>
5	ICFC	0	R/(W)*	<p>Input Capture Flag C</p> <p>This status flag indicates that the FRC value has been transferred to ICRC by means of an input capture signal. When BUFEA = 1, on occurrence of an input capture signal specified by the IEDGC bit at the FTIC input pin, ICFC is set but data is not transferred to ICRC. In buffer operation, ICFC can be used as an external interrupt signal by setting the ICICE bit to 1.</p> <p>[Setting condition]</p> <p>When an input capture signal is received</p> <p>[Clearing condition]</p> <p>Read ICFC when ICFC = 1, then write 0 to ICFC</p>
4	ICFD	0	R/(W)*	<p>Input Capture Flag D</p> <p>This status flag indicates that the FRC value has been transferred to ICRD by means of an input capture signal. When BUFEB = 1, on occurrence of an input capture signal specified by the IEDGD bit at the FTID input pin, ICFD is set but data is not transferred to ICRD. In buffer operation, ICFD can be used as an external interrupt signal by setting the ICIDE bit to 1.</p> <p>[Setting condition]</p> <p>When an input capture signal is received</p> <p>[Clearing condition]</p> <p>Read ICFD when ICFD = 1, then write 0 to ICFD</p>

Bit	Bit Name	Initial Value	R/W	Description
3	OCFA	0	R/(W)*	<p>Output Compare Flag A</p> <p>This status flag indicates that the FRC value matches the OCRA value.</p> <p>[Setting condition] When FRC = OCRA</p> <p>[Clearing condition] Read OCFA when OCFA = 1, then write 0 to OCFA</p>
2	OCFB	0	R/(W)*	<p>Output Compare Flag B</p> <p>This status flag indicates that the FRC value matches the OCRB value.</p> <p>[Setting condition] When FRC = OCRB</p> <p>[Clearing condition] Read OCFB when OCFB = 1, then write 0 to OCFB</p>
1	OVF	0	R/(W)*	<p>Overflow Flag</p> <p>This status flag indicates that the FRC has overflowed.</p> <p>[Setting condition] When FRC overflows (changes from H'FFFF to H'0000)</p> <p>[Clearing condition] Read OVF when OVF = 1, then write 0 to OVF</p>
0	CCLRA	0	R/W	<p>Counter Clear A</p> <p>This bit selects whether the FRC is to be cleared at compare-match A (when the FRC and OCRA values match).</p> <p>0: FRC clearing is disabled 1: FRC is cleared at compare-match A</p>

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

### 12.3.8 Timer Control Register (TCR)

TCR selects the rising or falling edge of the input capture signals, enables the input capture buffer mode, and selects the FRC clock source.

Bit	Bit Name	Initial Value	R/W	Description
7	IEDGA	0	R/W	Input Edge Select A Selects the rising or falling edge of the input capture A signal (FTIA). 0: Capture on the falling edge of FTIA 1: Capture on the rising edge of FTIA
6	IEDGB	0	R/W	Input Edge Select B Selects the rising or falling edge of the input capture B signal (FTIB). 0: Capture on the falling edge of FTIB 1: Capture on the rising edge of FTIB
5	IEDGC	0	R/W	Input Edge Select C Selects the rising or falling edge of the input capture C signal (FTIC). 0: Capture on the falling edge of FTIC 1: Capture on the rising edge of FTIC
4	IEDGD	0	R/W	Input Edge Select D Selects the rising or falling edge of the input capture D signal (FTID). 0: Capture on the falling edge of FTID 1: Capture on the rising edge of FTID
3	BUFEA	0	R/W	Buffer Enable A Selects whether ICRC is to be used as a buffer register for ICRA. 0: ICRC is not used as a buffer register for ICRA 1: ICRC is used as a buffer register for ICRA
2	BUFEB	0	R/W	Buffer Enable B Selects whether ICRD is to be used as a buffer register for ICRB. 0: ICRD is not used as a buffer register for ICRB 1: ICRD is used as a buffer register for ICRB

Bit	Bit Name	Initial Value	R/W	Description
1	CKS1	0	R/W	Clock Select 1, 0
0	CKS0	0	R/W	Select clock source for FRC. 00: $\phi/2$ internal clock source 01: $\phi/8$ internal clock source 10: $\phi/32$ internal clock source 11: External clock source (counting at FTCl rising edge)

### 12.3.9 Timer Output Compare Control Register (TOCR)

TOCR enables output from the output compare pins, selects the output levels, switches access between output compare registers A and B, controls the ICRD and OCRA operating modes, and switches access to input capture registers A, B, and C.

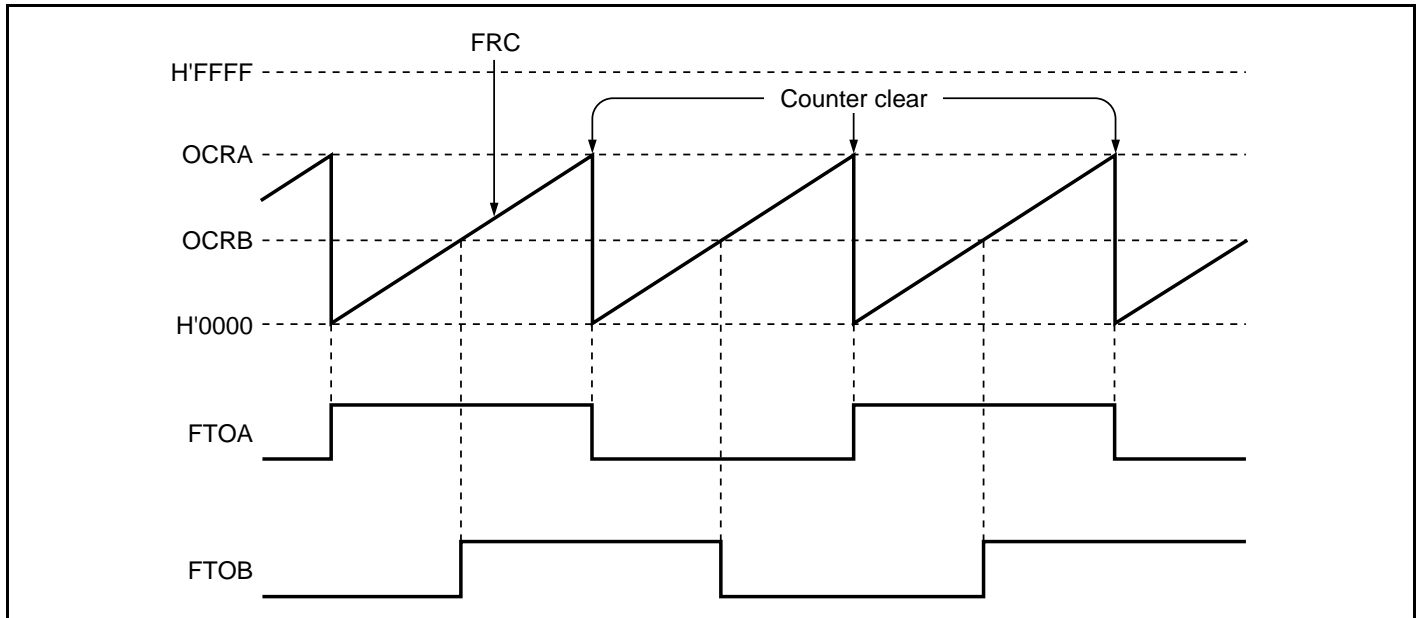
Bit	Bit Name	Initial Value	R/W	Description
7	ICRDMS	0	R/W	Input Capture D Mode Select Specifies whether ICRD is used in the normal operating mode or in the operating mode using OCRDM. 0: The normal operating mode is specified for ICRD 1: The operating mode using OCRDM is specified for ICRD
6	OCRAMS	0	R/W	Output Compare A Mode Select Specifies whether OCRA is used in the normal operating mode or in the operating mode using OCRAR and OCRAF. 0: The normal operating mode is specified for OCRA 1: The operating mode using OCRAR and OCRAF is specified for OCRA
5	ICRS	0	R/W	Input Capture Register Select The same addresses are shared by ICRA and OCRAR, by ICRB and OCRAF, and by ICRC and OCRDM. The ICRS bit determines which registers are selected when the shared addresses are read from or written to. The operation of ICRA, ICRB, and ICRC is not affected. 0: ICRA, ICRB, and ICRC are selected 1: OCRAR, OCRAF, and OCRDM are selected

Bit	Bit Name	Initial Value	R/W	Description
4	OCRS	0	R/W	<p>Output Compare Register Select</p> <p>OCRA and OCRB share the same address. When this address is accessed, the OCRS bit selects which register is accessed. The operation of OCRA or OCRB is not affected.</p> <p>0: OCRA is selected 1: OCRB is selected</p>
3	OEA	0	R/W	<p>Output Enable A</p> <p>Enables or disables output of the output compare A output pin (FTOA).</p> <p>0: Output compare A output is disabled 1: Output compare A output is enabled</p>
2	OEB	0	R/W	<p>Output Enable B</p> <p>Enables or disables output of the output compare B output pin (FTOB).</p> <p>0: Output compare B output is disabled 1: Output compare B output is enabled</p>
1	OLVLA	0	R/W	<p>Output Level A</p> <p>Selects the level to be output at the output compare A output pin (FTOA) in response to compare-match A (signal indicating a match between the FRC and OCRA values). When the OCRAMS bit is 1, this bit is ignored.</p> <p>0: 0 is output at compare-match A 1: 1 is output at compare-match A</p>
0	OLVLB	0	R/W	<p>Output Level B</p> <p>Selects the level to be output at the output compare B output pin (FTOB) in response to compare-match B (signal indicating a match between the FRC and OCRB values).</p> <p>0: 0 is output at compare-match B 1: 1 is output at compare-match B</p>

## 12.4 Operation

### 12.4.1 Pulse Output

Figure 12.2 shows an example of 50%-duty pulses output with an arbitrary phase difference. When a compare match occurs while the CCLRA bit in TCSR is set to 1, the OLVLA and OLVLB bits are inverted by software.

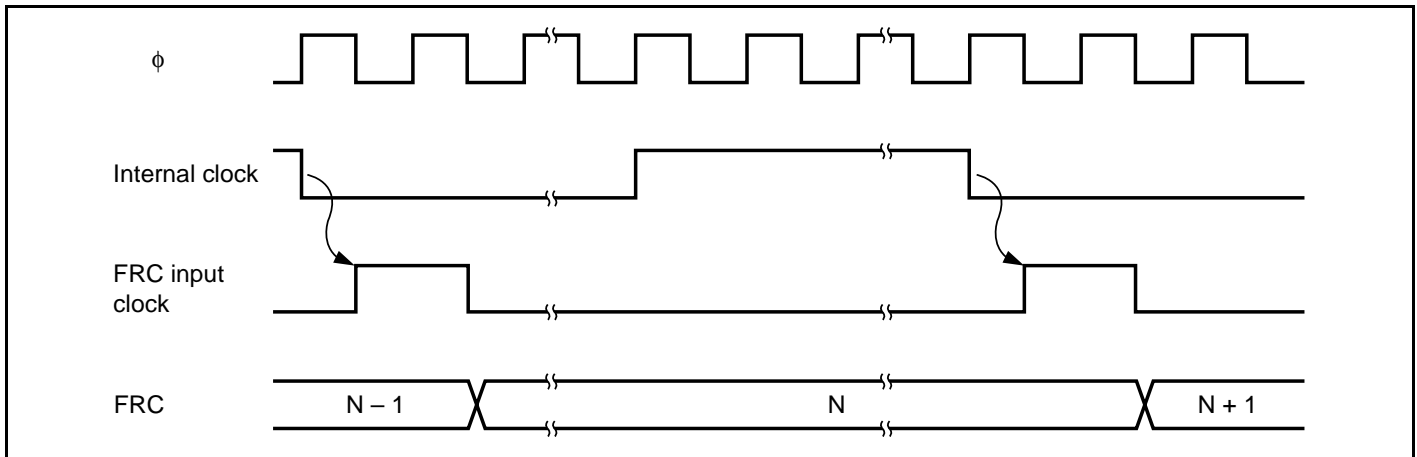


**Figure 12.2 Example of Pulse Output**

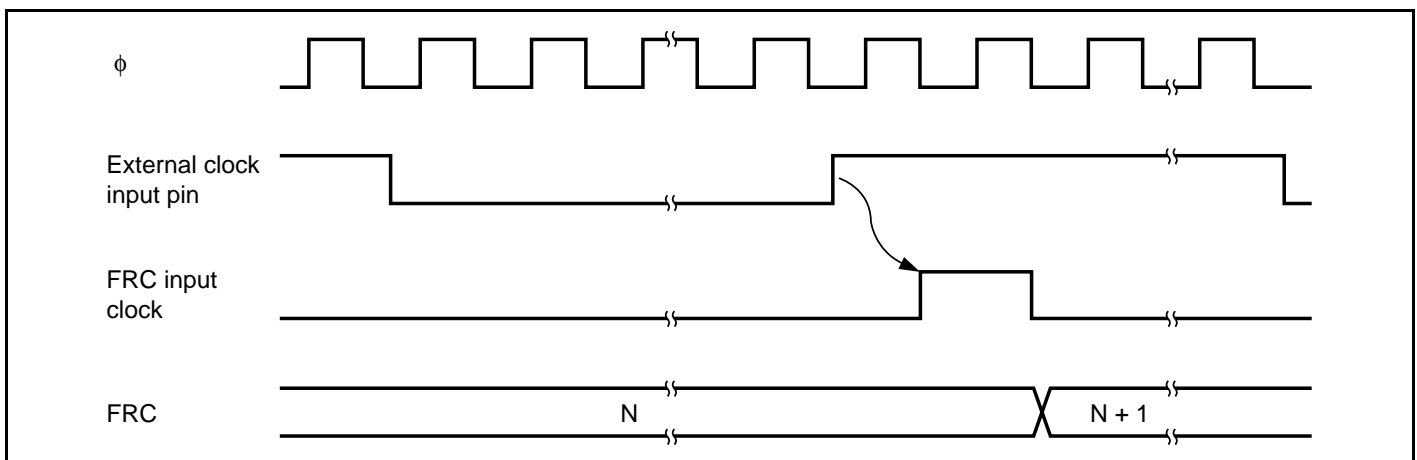
## 12.5 Operation Timing

### 12.5.1 FRC Increment Timing

Figure 12.3 shows the FRC increment timing with an internal clock source. Figure 12.4 shows the increment timing with an external clock source. The pulse width of the external clock signal must be at least 1.5 system clocks ( $\phi$ ). The counter will not increment correctly if the pulse width is shorter than 1.5 system clocks ( $\phi$ ).



**Figure 12.3 Increment Timing with Internal Clock Source**

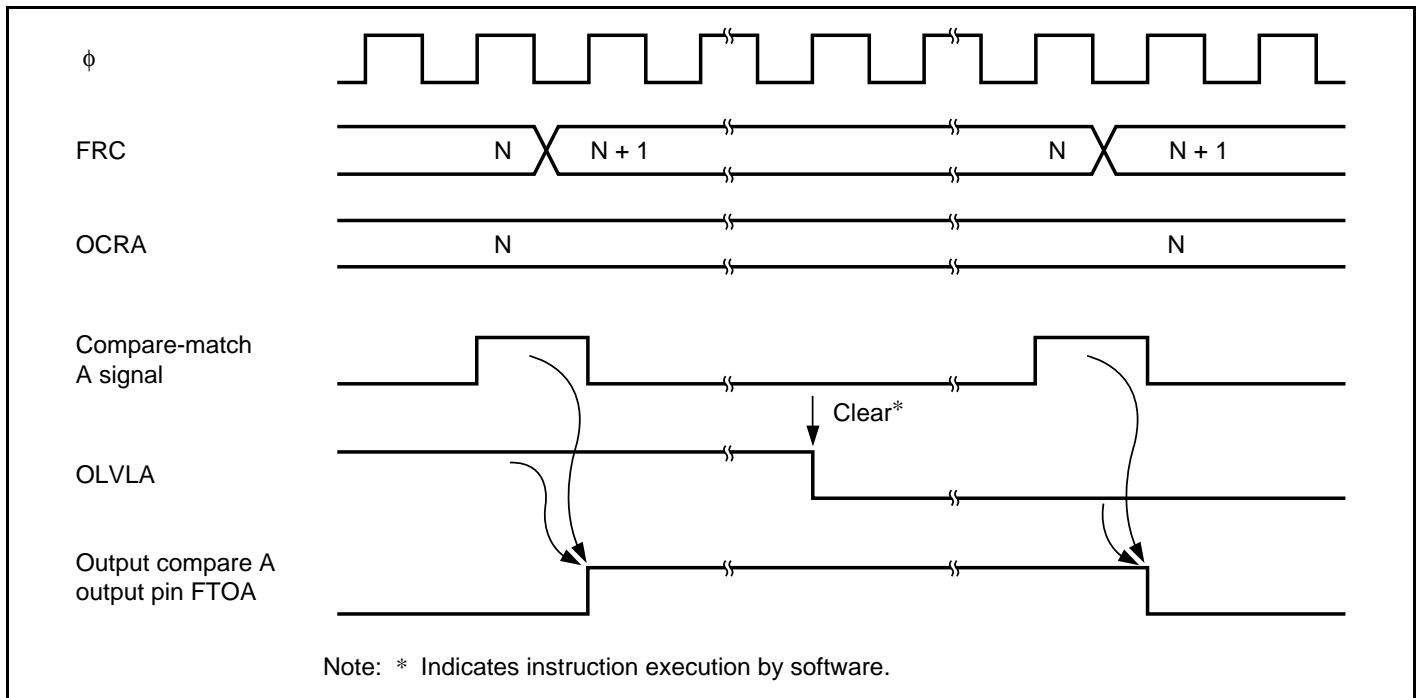


**Figure 12.4 Increment Timing with External Clock Source**



### 12.5.2 Output Compare Output Timing

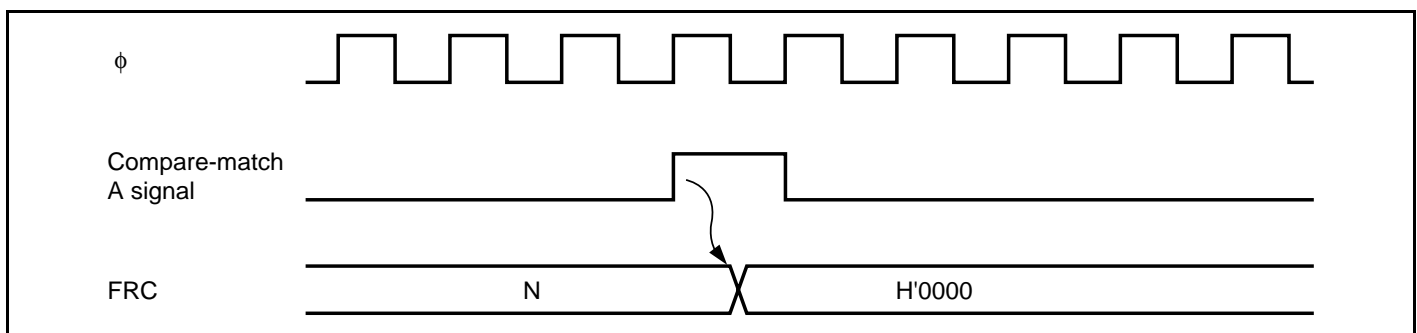
A compare-match signal occurs at the last state when the FRC and OCR values match (at the timing when the FRC updates the counter value). When a compare-match signal occurs, the level selected by the OLVL bit in TOCR is output at the output compare pin (FTOA or FTOB). Figure 12.5 shows the timing of this operation for compare-match A.



**Figure 12.5 Timing of Output Compare A Output**

### 12.5.3 FRC Clear Timing

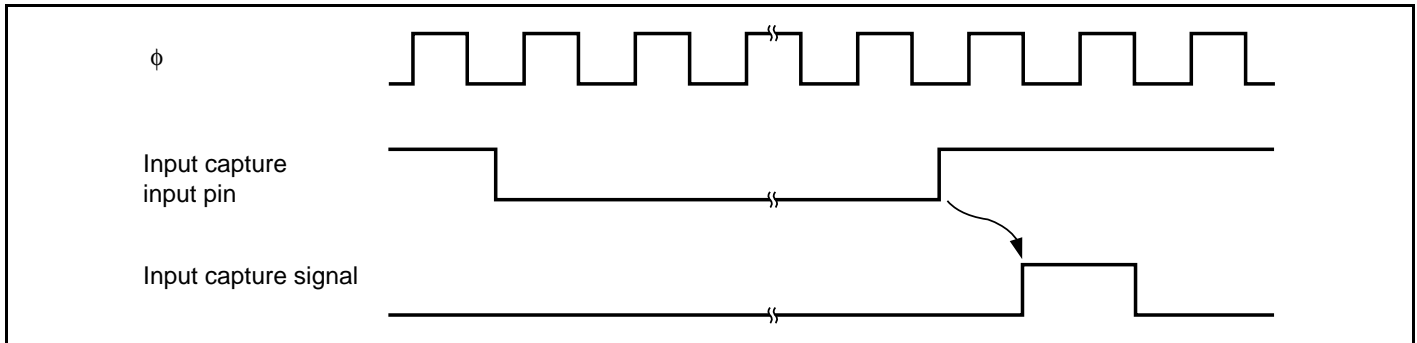
FRC can be cleared when compare-match A occurs. Figure 12.6 shows the timing of this operation.



**Figure 12.6 Clearing of FRC by Compare-Match A Signal**

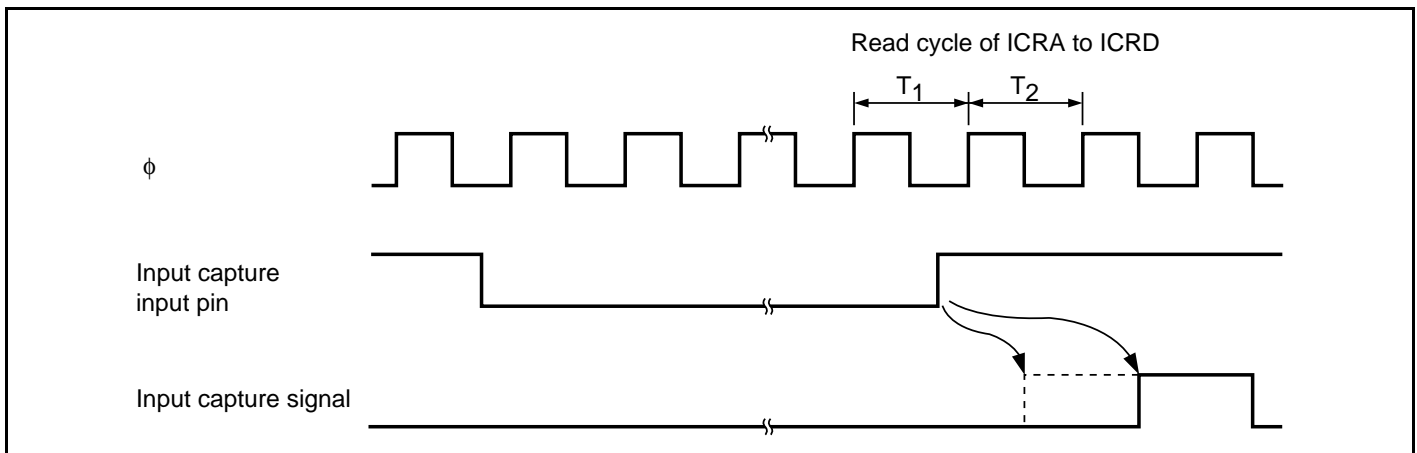
### 12.5.4 Input Capture Input Timing

The rising or falling edge can be selected for the input capture input timing by the IEDGA to IEDGD bits in TCR. Figure 12.7 shows the usual input capture timing when the rising edge is selected.



**Figure 12.7 Input Capture Input Signal Timing (Usual Case)**

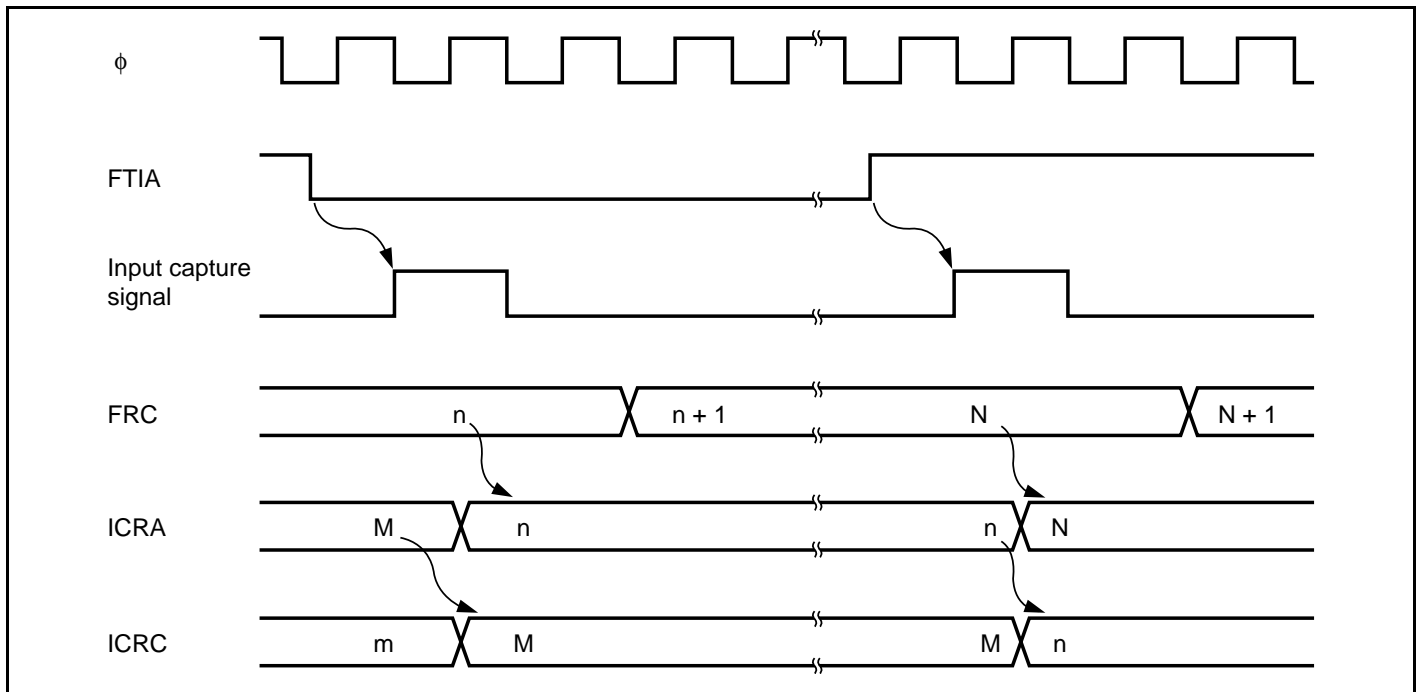
If ICRA to ICRAD are read when the corresponding input capture signal arrives, the internal input capture signal is delayed by one system clock ( $\phi$ ). Figure 12.8 shows the timing for this case.



**Figure 12.8 Input Capture Input Signal Timing (When ICRA to ICRD Is Read)**

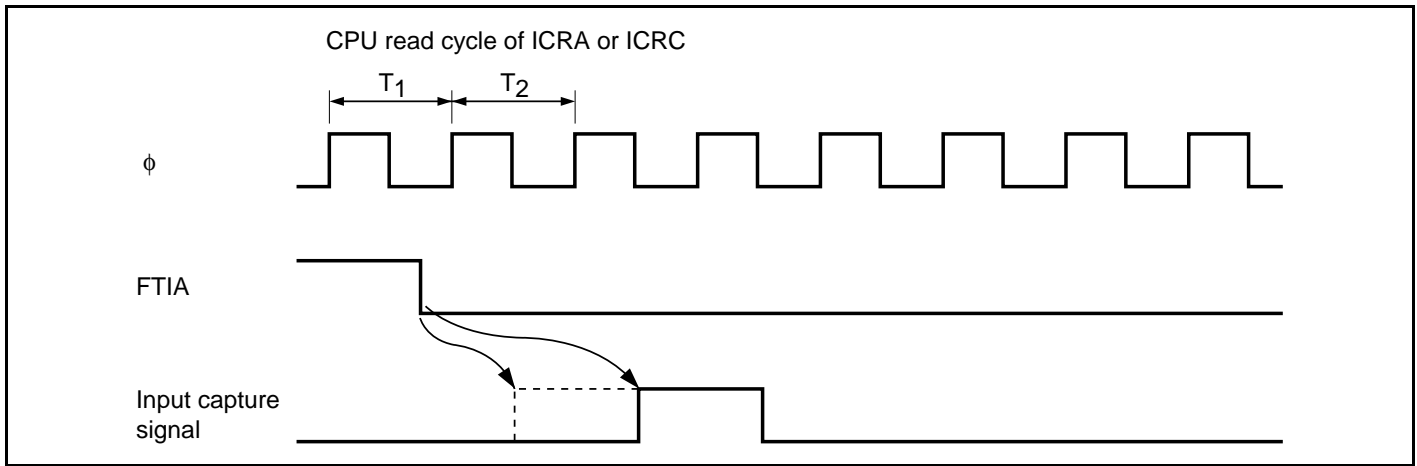
### 12.5.5 Buffered Input Capture Input Timing

ICRC and ICRD can operate as buffers for ICRA and ICRB, respectively. Figure 12.9 shows how input capture operates when ICRC is used as ICRA's buffer register (BUFEA = 1) and IEDGA and IEDGC are set to different values (IEDGA = 0 and IEDGC = 1, or IEDGA = 1 and IEDGC = 0), so that input capture is performed on both the rising and falling edges of FTIA.



**Figure 12.9 Buffered Input Capture Timing**

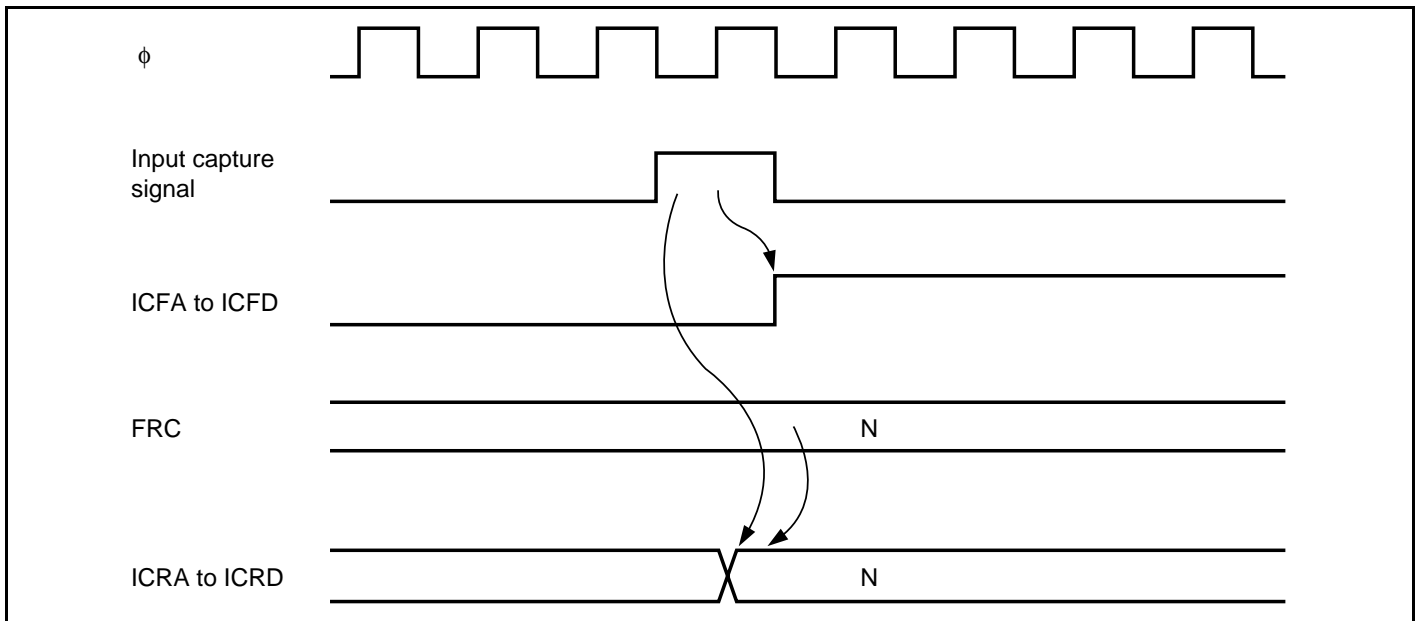
Even when ICRC or ICRD is used as a buffer register, its input capture flag is set by the selected transition of its input capture signal. For example, if ICRC is used to buffer ICRA, when the edge transition selected by the IEDGC bit occurs on the FTIC input capture line, ICFC will be set, and if the ICICE bit is set at this time, an interrupt will be requested. The FRC value will not be transferred to ICRC, however. In buffered input capture, if either set of two registers to which data will be transferred (ICRA and ICRC, or ICRB and ICRD) is being read when the input capture input signal arrives, input capture is delayed by one system clock ( $\phi$ ). Figure 12.10 shows the timing when BUFEA = 1.



**Figure 12.10 Buffered Input Capture Timing (BUFEA = 1)**

### 12.5.6 Timing of Input Capture Flag (ICF) Setting

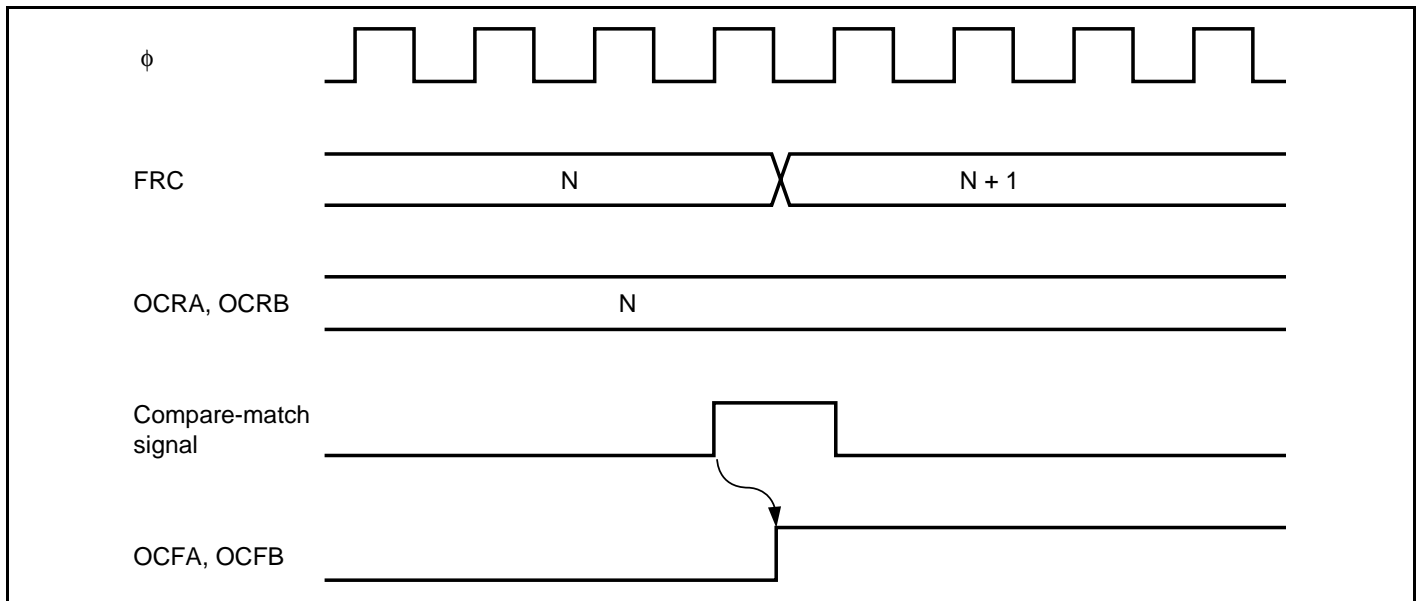
The input capture flag, ICFA, ICFB, ICFC, or ICFD, is set to 1 by the input capture signal. The FRC value is simultaneously transferred to the corresponding input capture register (ICRA, ICRB, ICRC, or ICRD). Figure 12.11 shows the timing of setting the ICFA to ICFD flag.



**Figure 12.11 Timing of Input Capture Flag (ICFA, ICFB, ICFC, or ICFD) Setting**

### 12.5.7 Timing of Output Compare Flag (OCF) setting

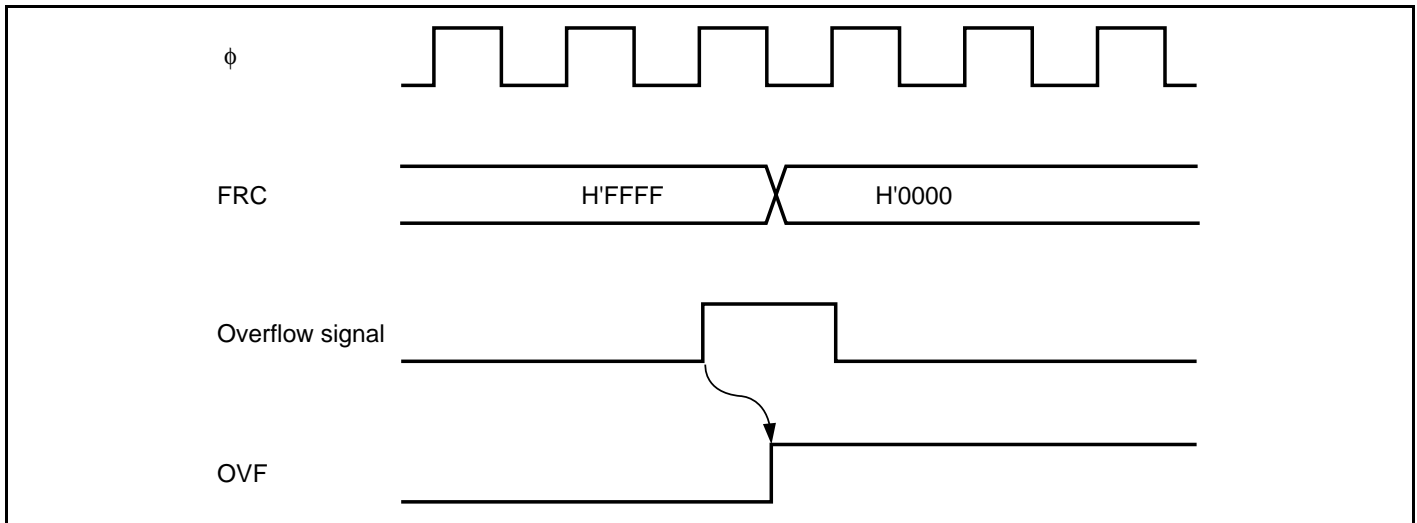
The output compare flag, OCFA or OCFB, is set to 1 by a compare-match signal generated when the FRC value matches the OCRA or OCRB value. This compare-match signal is generated at the last state in which the two values match, just before FRC increments to a new value. When the FRC and OCRA or OCRB value match, the compare-match signal is not generated until the next cycle of the clock source. Figure 12.12 shows the timing of setting the OCFA or OCFB flag.



**Figure 12.12 Timing of Output Compare Flag (OCFA or OCFB) Setting**

### 12.5.8 Timing of FRC Overflow Flag Setting

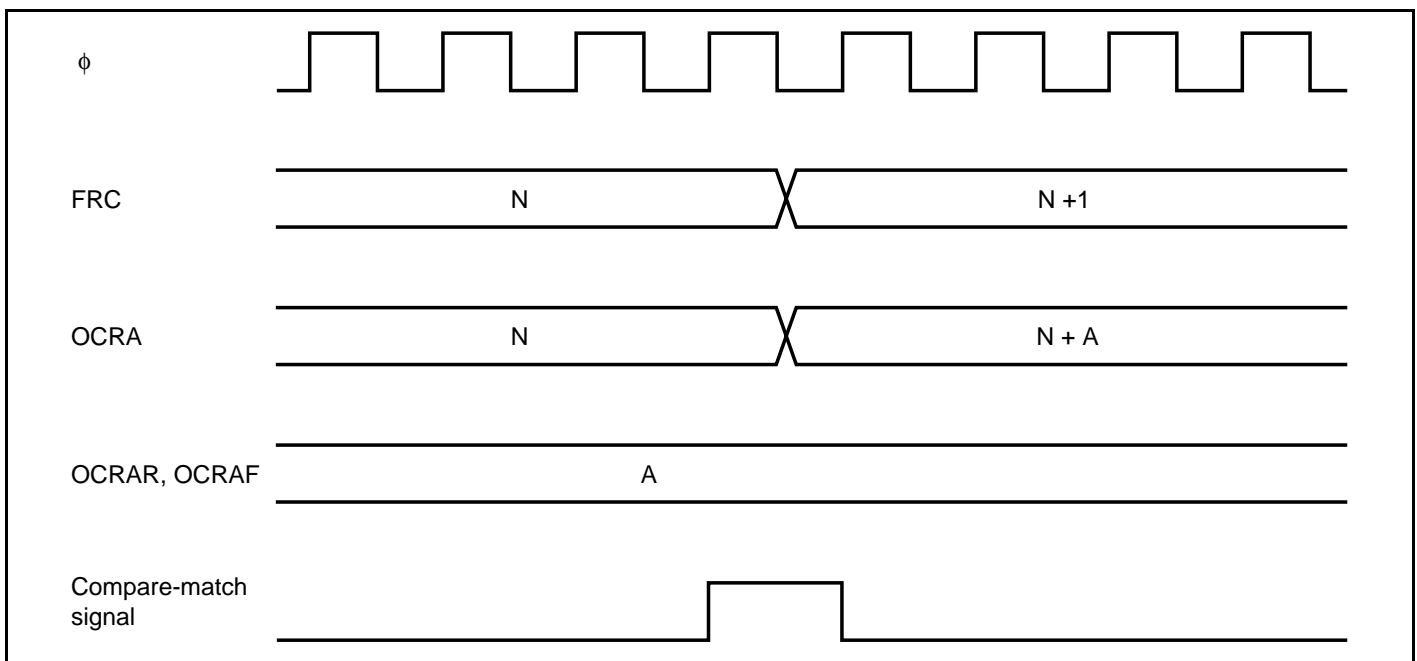
The FRC overflow flag (OVF) is set to 1 when FRC overflows (changes from H'FFFF to H'0000). Figure 12.13 shows the timing of setting the OVF flag.



**Figure 12.13 Timing of Overflow Flag (OVF) Setting**

### 12.5.9 Automatic Addition Timing

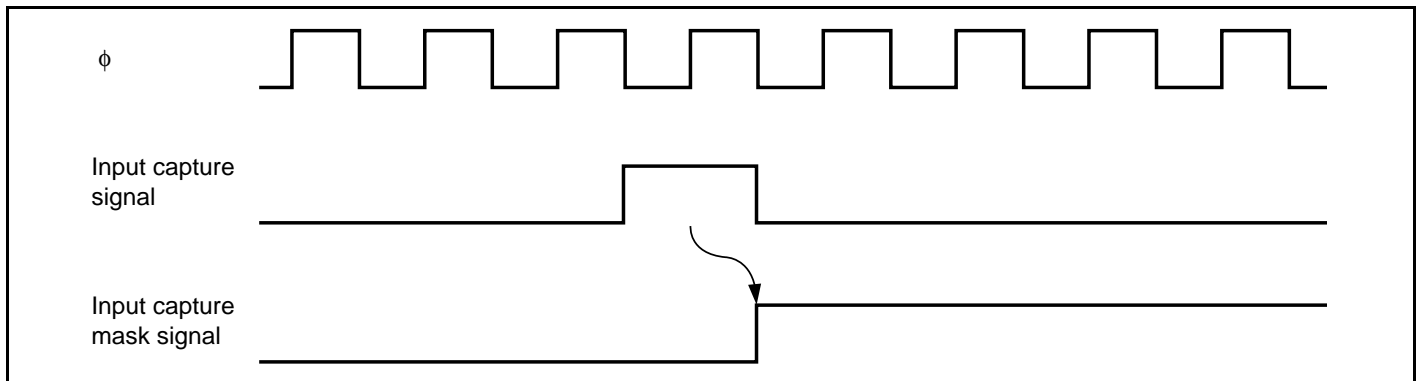
When the OCRAMS bit in TOCR is set to 1, the contents of OCRAR and OCRAF are automatically added to OCRA alternately, and when an OCRA compare-match occurs a write to OCRA is performed. Figure 12.14 shows the OCRA write timing.



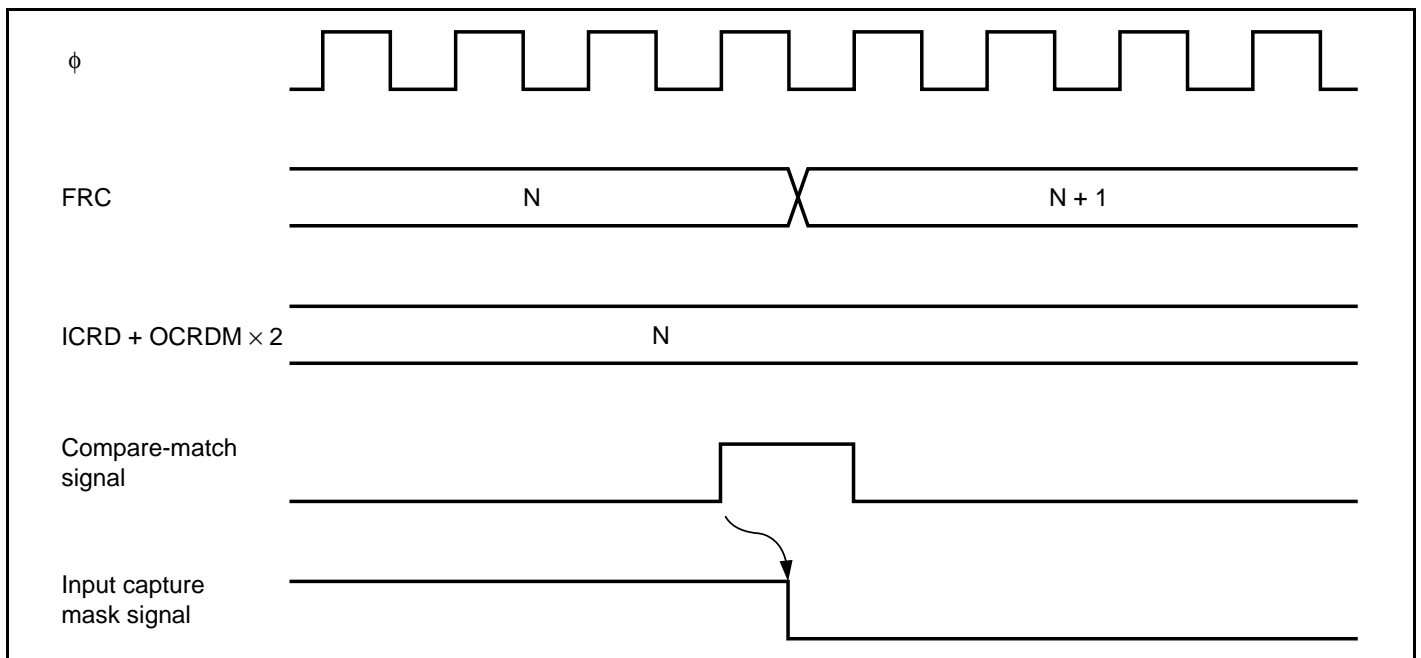
**Figure 12.14 OCRA Automatic Addition Timing**

### 12.5.10 Mask Signal Generation Timing

When the ICRDMS bit in TOCR is set to 1 and the contents of OCRDM are other than H'0000, a signal that masks the ICRD input capture signal is generated. The mask signal is set by the input capture signal. The mask signal is cleared by the sum of the ICRD contents and twice the OCRDM contents, and an FRC compare-match. Figure 12.15 shows the timing of setting the mask signal. Figure 12.16 shows the timing of clearing the mask signal.



**Figure 12.15 Timing of Input Capture Mask Signal Setting**



**Figure 12.16 Timing of Input Capture Mask Signal Clearing**

## 12.6 Interrupt Sources

The free-running timer can request seven interrupts: ICIA to ICID, OCIA, OCIB, and FOVI. Each interrupt can be enabled or disabled by an enable bit in TIER. Independent signals are sent to the interrupt controller for each interrupt. Table 12.2 lists the sources and priorities of these interrupts.

The ICIA, ICIB, OCIA, and OCIB interrupts can be used as the on-chip DTC activation sources.

**Table 12.2 FRT Interrupt Sources**

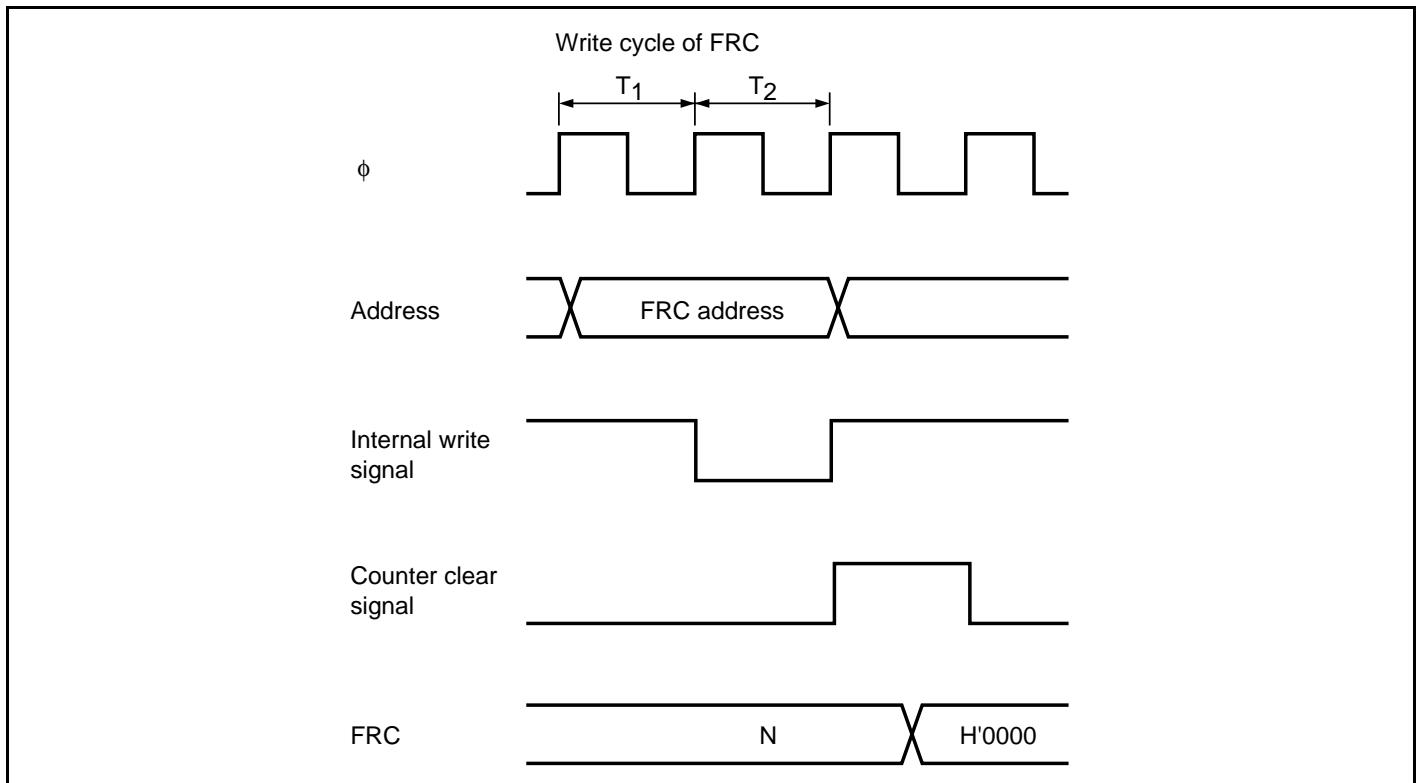
Interrupt	Interrupt Source	Interrupt Flag	DTC Activation	Priority
ICIA	Input capture of ICRA	ICFA	Possible	High
ICIB	Input capture of ICRB	ICFB	Possible	↑ Low
ICIC	Input capture of ICRC	ICFC	Not possible	
ICID	Input capture of ICRD	ICFD	Not possible	
OCIA	Compare match of OCRA	OCFA	Possible	
OCIB	Compare match of OCRB	OCFB	Possible	
FOVI	Overflow of FRC	OVF	Not possible	



## 12.7 Usage Notes

### 12.7.1 Conflict between FRC Write and Clear

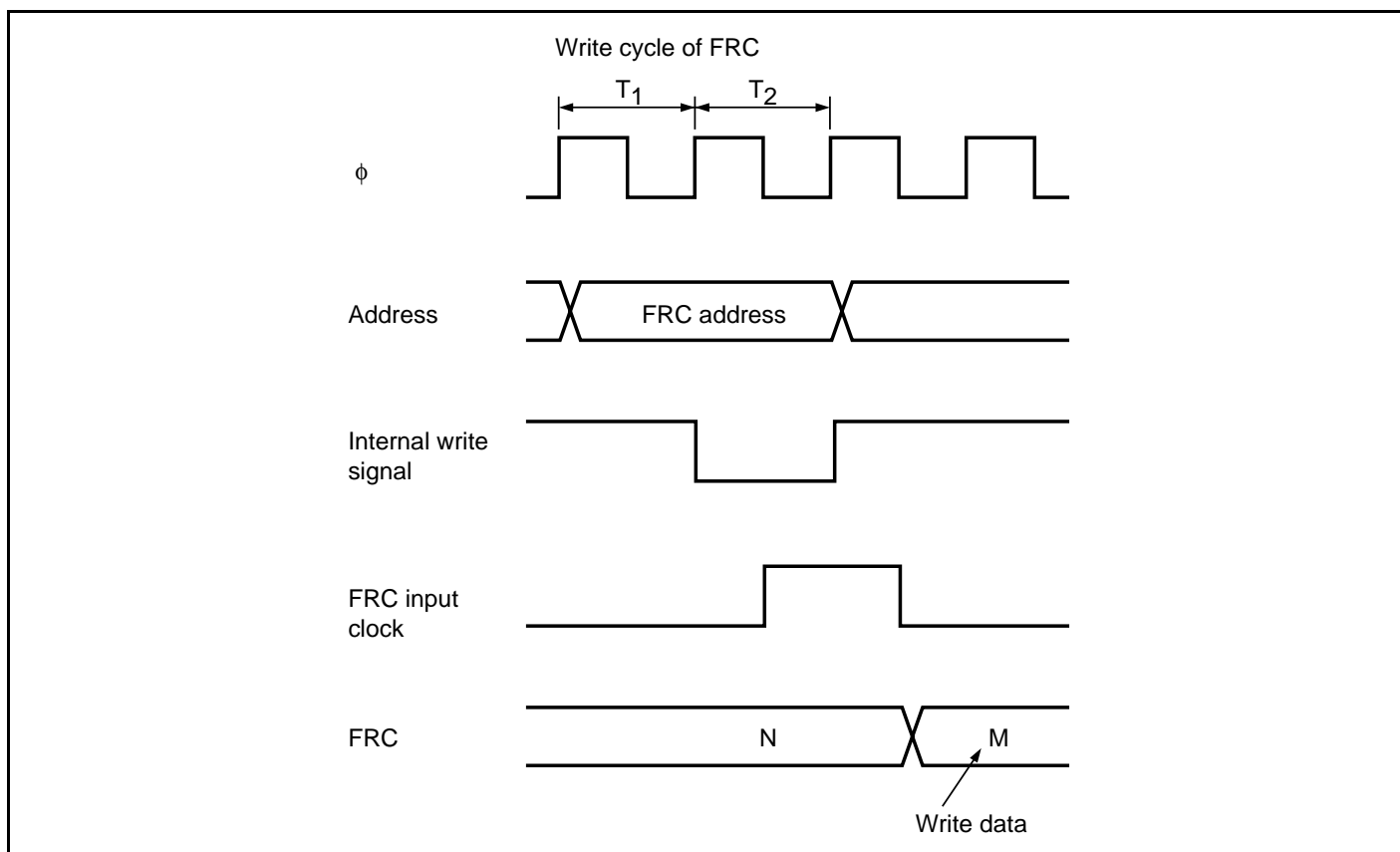
If an internal counter clear signal is generated during the state after an FRC write cycle, the clear signal takes priority and the write is not performed. Figure 12.17 shows the timing for this type of conflict.



**Figure 12.17 FRC Write-Clear Conflict**

### 12.7.2 Conflict between FRC Write and Increment

If an FRC increment pulse is generated during the state after an FRC write cycle, the write takes priority and FRC is not incremented. Figure 12.18 shows the timing for this type of conflict.

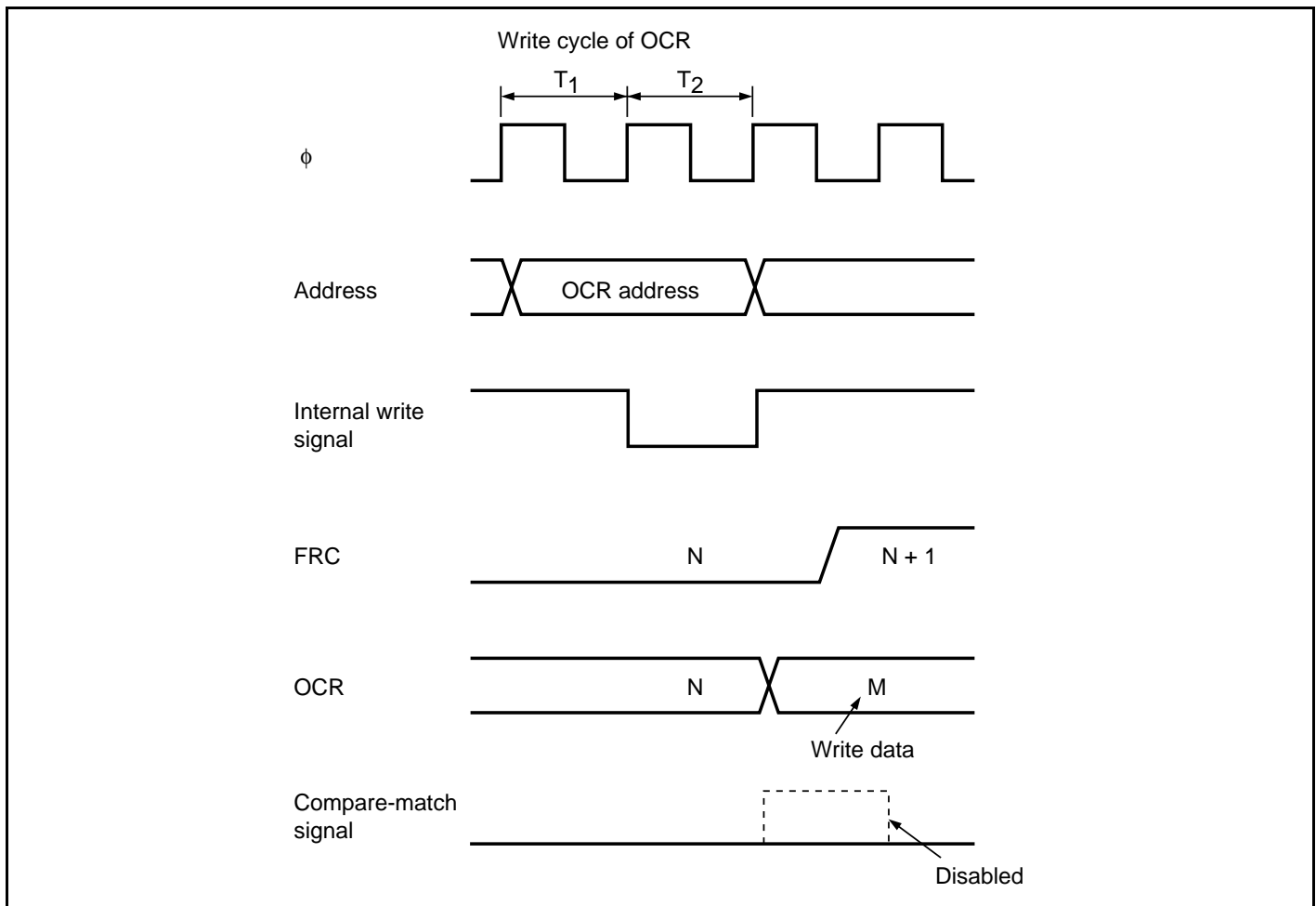


**Figure 12.18 FRC Write-Increment Conflict**

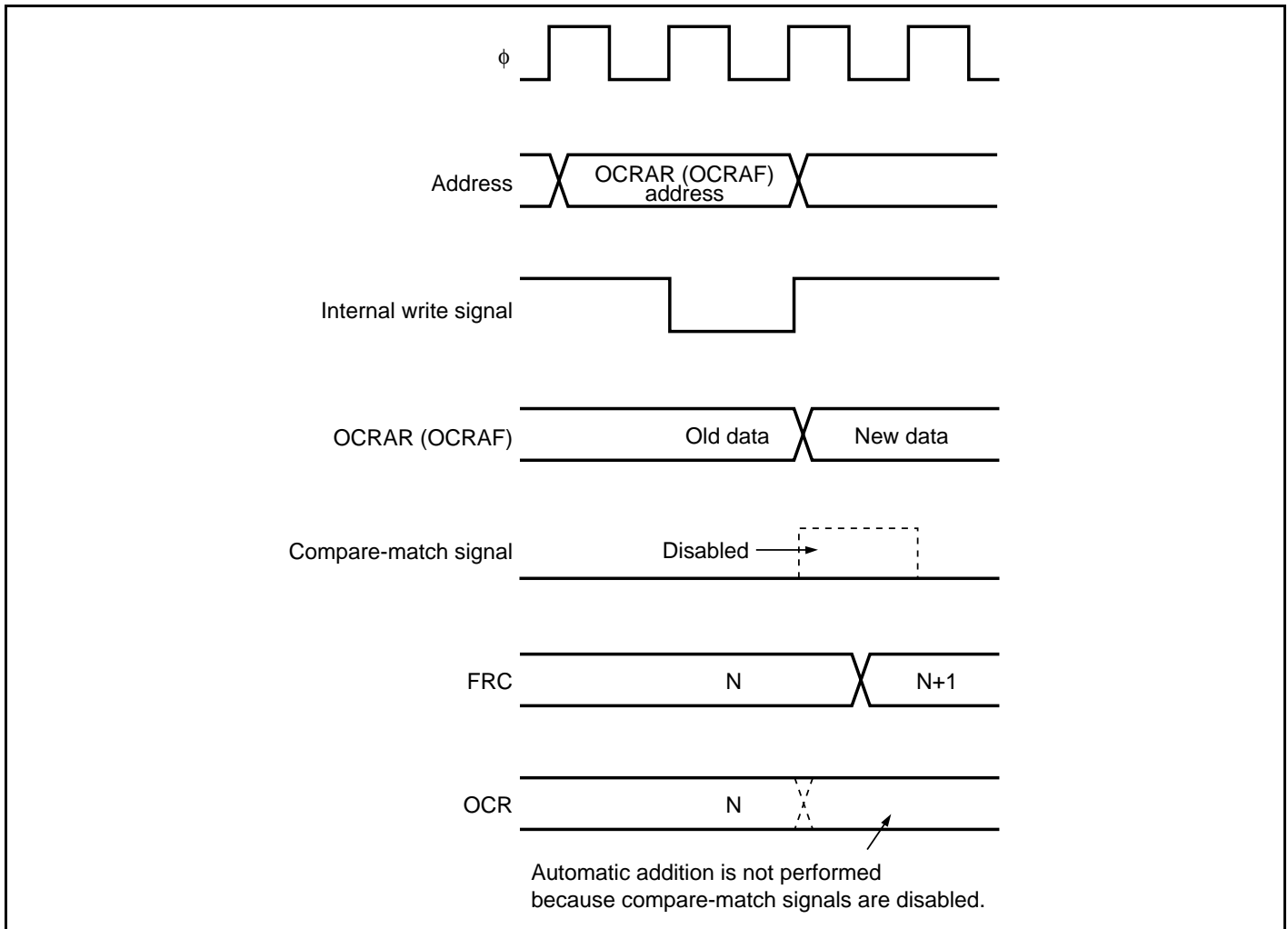
### 12.7.3 Conflict between OCR Write and Compare-Match

If a compare-match occurs during the state after an OCRA or OCRB write cycle, the write takes priority and the compare-match signal is disabled. Figure 12.19 shows the timing for this type of conflict.

If automatic addition of OCRAR and OCRAF to OCRA is selected, and a compare-match occurs in the cycle following the OCRA, OCRAR, and OCRAF write cycle, the OCRA, OCRAR and OCRAF write takes priority and the compare-match signal is disabled. Consequently, the result of the automatic addition is not written to OCRA. Figure 12.20 shows the timing for this type of conflict.



**Figure 12.19 Conflict between OCR Write and Compare-Match  
(When Automatic Addition Function Is Not Used)**



**Figure 12.20 Conflict between OCR Write and Compare-Match  
(When Automatic Addition Function Is Used)**

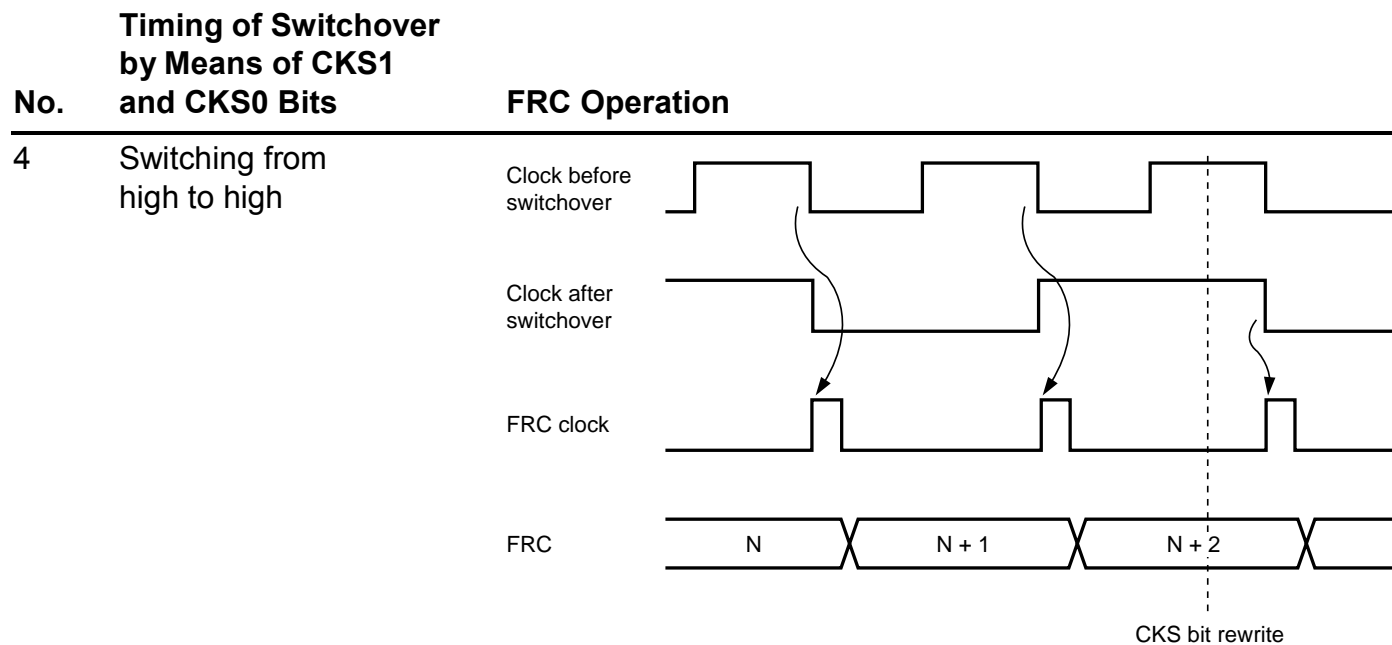
#### 12.7.4 Switching of Internal Clock and FRC Operation

When the internal clock is changed, the changeover may cause FRC to increment. This depends on the time at which the clock is switched (bits CKS1 and CKS0 are rewritten), as shown in table 12.3.

When an internal clock is used, the FRC clock is generated on detection of the falling edge of the internal clock scaled from the system clock ( $\phi$ ). If the clock is changed when the old source is high and the new source is low, as in case no. 3 in table 12.3, the changeover is regarded as a falling edge that triggers the FRC clock, and FRC is incremented. Switching between an internal clock and external clock can also cause FRC to increment.

Table 12.3 Switching of Internal Clock and FRC Operation

No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	FRC Operation
1	Switching from low to low	<p data-bbox="618 323 743 373">Clock before switchover</p> <p data-bbox="618 443 743 493">Clock after switchover</p> <p data-bbox="618 573 724 598">FRC clock</p> <p data-bbox="618 688 667 714">FRC</p> <p data-bbox="854 688 878 714">N</p> <p data-bbox="1166 688 1214 714">N + 1</p> <p data-bbox="906 779 1052 804">CKS bit rewrite</p>
2	Switching from low to high	<p data-bbox="618 858 743 909">Clock before switchover</p> <p data-bbox="618 978 743 1029">Clock after switchover</p> <p data-bbox="618 1108 724 1134">FRC clock</p> <p data-bbox="618 1224 667 1249">FRC</p> <p data-bbox="854 1224 878 1249">N</p> <p data-bbox="1044 1224 1092 1249">N + 1</p> <p data-bbox="1263 1224 1312 1249">N + 2</p> <p data-bbox="1130 1314 1276 1339">CKS bit rewrite</p>
3	Switching from high to low	<p data-bbox="618 1394 743 1444">Clock before switchover</p> <p data-bbox="618 1514 743 1564">Clock after switchover</p> <p data-bbox="618 1644 724 1669">FRC clock</p> <p data-bbox="618 1759 667 1785">FRC</p> <p data-bbox="854 1759 878 1785">N</p> <p data-bbox="1016 1759 1065 1785">N + 1</p> <p data-bbox="1235 1759 1284 1785">N + 2</p> <p data-bbox="1016 1850 1162 1875">CKS bit rewrite</p>



Note: \* Generated on the assumption that the switchover is a falling edge; FRC is incremented.

## Section 13 8-Bit Timer (TMR)

This LSI has an on-chip 8-bit timer module (TMR\_0 and TMR\_1) with two channels operating on the basis of an 8-bit counter. The 8-bit timer module can be used as a multifunction timer in a variety of applications, such as generation of counter reset, interrupt requests, and pulse output with an arbitrary duty cycle using a compare-match signal with two registers.

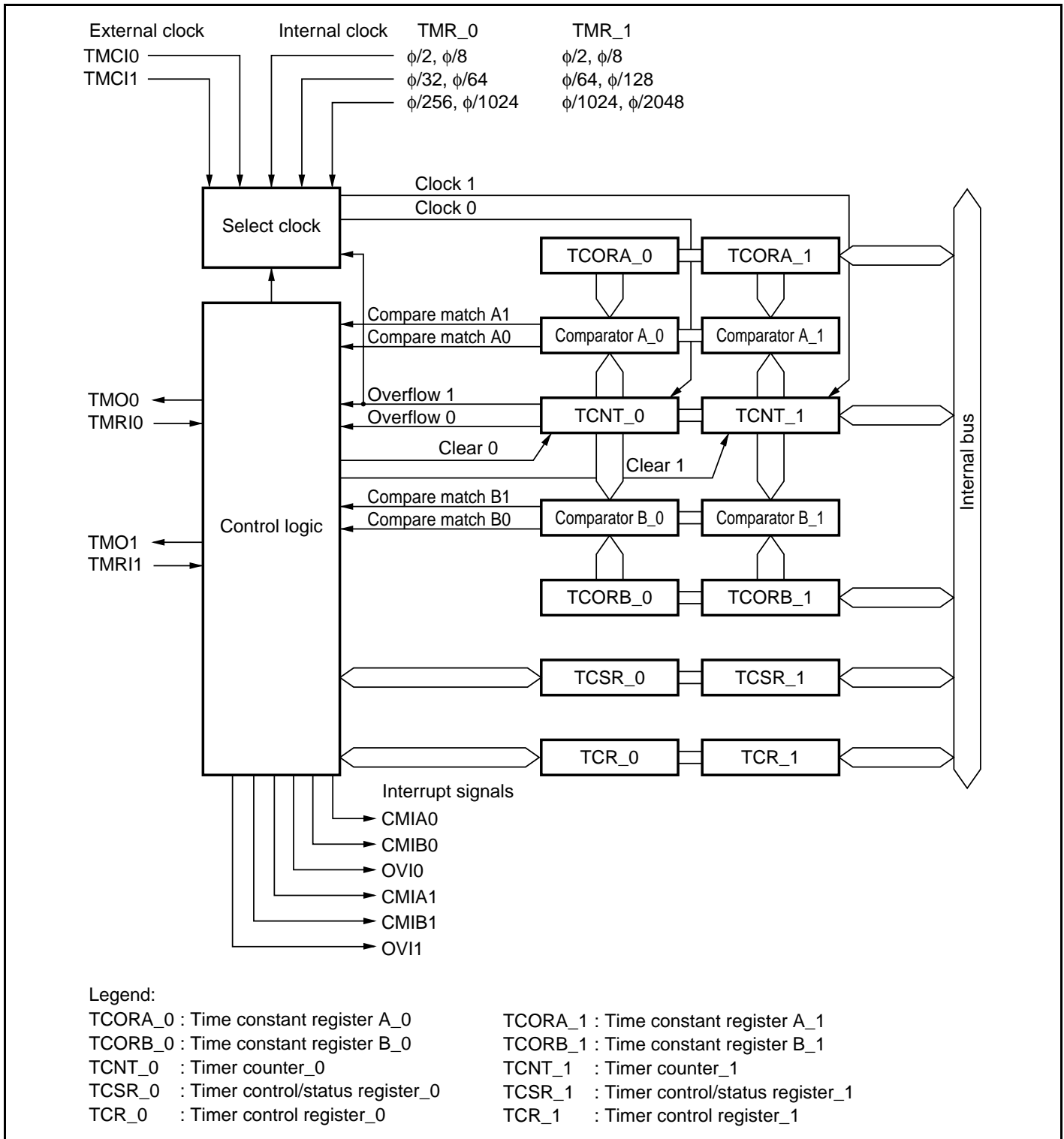
This LSI also has a similar on-chip 8-bit timer module (TMR\_Y and TMR\_X) with two channels, which can be used through connection to the timer connection.

### 13.1 Features

- Selection of clock sources
  - TMR\_0, TMR\_1: The counter input clock can be selected from six internal clocks and an external clock
  - TMR\_Y, TMR\_X: The counter input clock can be selected from three internal clocks and an external clock
- Selection of three ways to clear the counters
  - The counters can be cleared on compare-match A or compare-match B, or by an external reset signal.
- Timer output controlled by two compare-match signals
  - The timer output signal in each channel is controlled by two independent compare-match signals, enabling the timer to be used for various applications, such as the generation of pulse output or PWM output with an arbitrary duty cycle.
- Cascading of two channels
  - Cascading of TMR\_0 and TMR\_1
    - Operation as a 16-bit timer can be performed using TMR\_0 as the upper half and TMR\_1 as the lower half (16-bit count mode).
    - TMR\_1 can be used to count TMR\_0 compare-match occurrences (compare-match count mode).
- Multiple interrupt sources for each channel
  - TMR\_0, TMR\_1, and TMR\_Y: Three types of interrupts: Compare-match A, compare-match B, and overflow
  - TMR\_X: Four types of interrupts: Compare-match A, compare-match B, overflow, and input capture

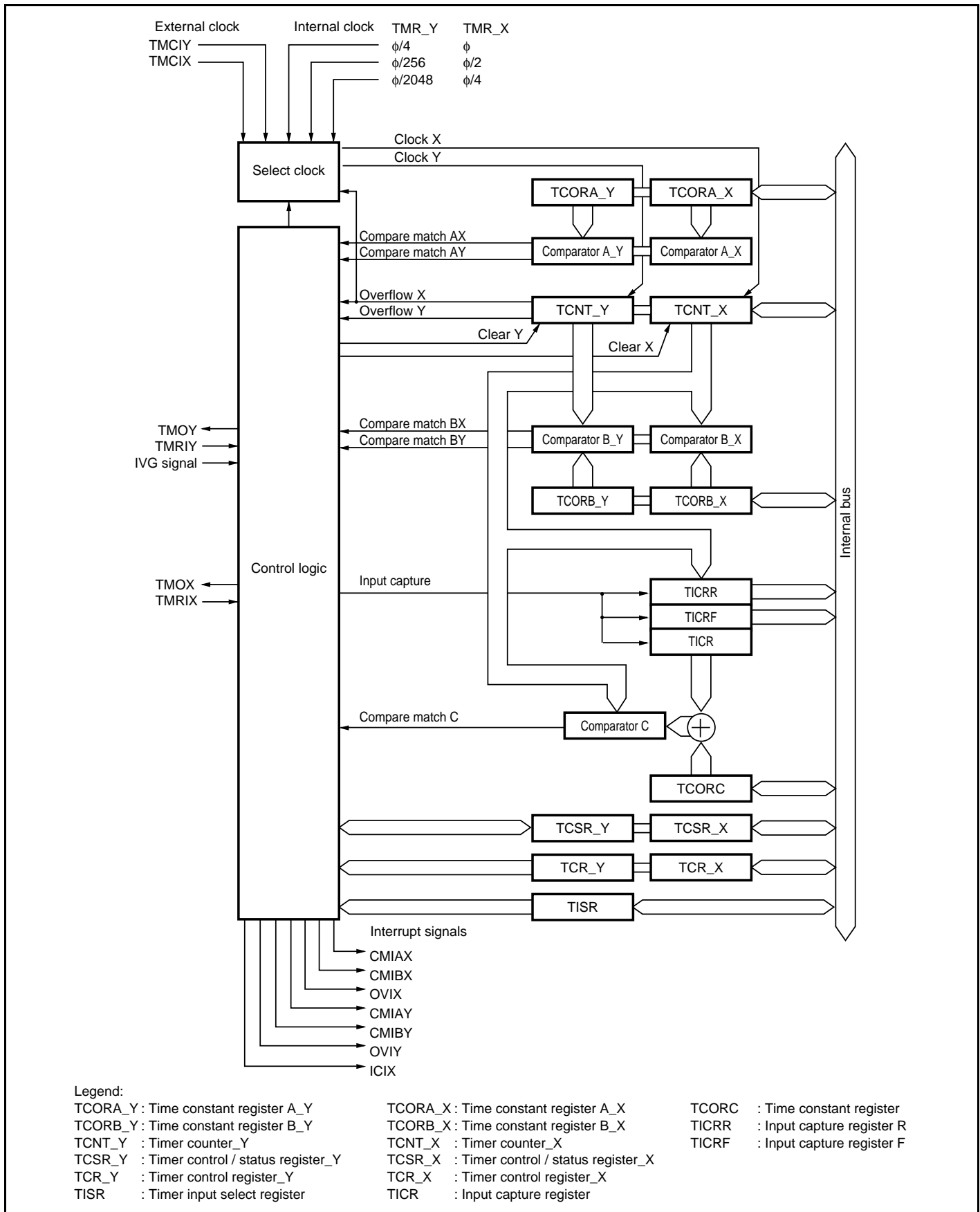
Figures 13.1 and 13.2 show block diagrams of 8-bit timers.

An input capture function is added to TMR\_X. For details, see section 14, Timer Connection.



**Figure 13.1 Block Diagram of 8-Bit Timer (TMR\_0 and TMR\_1)**





**Figure 13.2 Block Diagram of 8-Bit Timer (TMR\_Y and TMR\_X)**

## 13.2 Input/Output Pins

Table 13.1 summarizes the input and output pins of the TMR.

**Table 13.1 Pin Configuration**

Channel	Name	Symbol	I/O	Function
TMR_0	Timer output	TMO0	Output	Output controlled by compare-match
	Timer clock/reset input	TMI0/ExTMI0	Input	External clock input (TMCI0)/external reset input (TMRI0) for the counter
TMR_1	Timer output	TMO1	Output	Output controlled by compare-match
	Timer clock/reset input	TMI1/ExTMI1	Input	External clock input (TMCI1)/external reset input (TMRI1) for the counter
TMR_Y	Timer output	TMOY	Output	Output controlled by compare-match
	Timer clock/reset input	TMIY/ExTMIY	Input	External clock input (TMCIY)/external reset input (TMRIY) for the counter
TMR_X	Timer output	TMOX	Output	Output controlled by compare-match
	Timer clock/reset input	TMIX/ExTMIX	Input	External clock input (TMCIX)/external reset input (TMRIX) for the counter

## 13.3 Register Descriptions

The TMR has the following registers. For details on the serial timer control register, see section 3.2.3, Serial Timer Control Register (STCR). For details on timer connection register S, see section 14.3.3, Timer Connection Register S (TCONRS).

### TMR\_0

- Timer counter\_0 (TCNT\_0)
- Time constant register A\_0 (TCORA\_0)
- Time constant register B\_0 (TCORB\_0)
- Timer control register\_0 (TCR\_0)
- Timer control/status register\_0 (TCSR\_0)

## 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 control/status register\_1 (TCSR\_1)

## TMR\_Y

- Timer counter\_Y (TCNT\_Y)
- Time constant register A\_Y (TCORA\_Y)
- Time constant register B\_Y (TCORB\_Y)
- Timer control register\_Y (TCR\_Y)
- Timer control/status register\_Y (TCSR\_Y)
- Timer input select register (TISR)

## TMR\_X

- Timer counter\_X (TCNT\_X)
- Time constant register A\_X (TCORA\_X)
- Time constant register B\_X (TCORB\_X)
- Timer control register\_X (TCR\_X)
- Timer control/status register\_X (TCSR\_X)
- Input capture register (TICR)
- Time constant register (TCORC)
- Input capture register R (TICRR)
- Input capture register F (TICRF)

### 13.3.1 Timer Counter (TCNT)

Each TCNT is an 8-bit readable/writable up-counter. TCNT\_0 and TCNT\_1 (or TCNT\_Y and TCNT\_X) comprise a single 16-bit register, so they can be accessed together by word access. The clock source is selected by the CKS2 to CKS0 bits in TCR. TCNT can be cleared by an external reset input signal, compare-match A signal or compare-match B signal. The method of clearing can be selected by the CCLR1 and CCLR0 bits in TCR. When TCNT overflows (changes from H'FF to H'00), the OVF bit in TCSR is set to 1. TCNT is initialized to H'00.

### 13.3.2 Time Constant Register A (TCORA)

TCORA is an 8-bit readable/writable register. TCORA\_0 and TCORA\_1 (or TCORA\_Y and TCORA\_X) comprise a single 16-bit register, so they can be accessed together by word access. TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding compare-match flag A (CMFA) in TCSR is set to 1. Note however that comparison is disabled during the T2 state of a TCORA write cycle. The timer output from the TMO pin can be freely controlled by these compare-match A signals and the settings of output select bits OS1 and OS0 in TCSR. TCORA is initialized to H'FF.

### 13.3.3 Time Constant Register B (TCORB)

TCORB is an 8-bit readable/writable register. TCORB\_0 and TCORB\_1 (or TCORB\_Y and TCORB\_X) comprise a single 16-bit register, so they can be accessed together by word access. TCORB is continually compared with the value in TCNT. When a match is detected, the corresponding compare-match flag B (CMFB) in TCSR is set to 1. Note however that comparison is disabled during the T2 state of a TCORB write cycle. The timer output from the TMO pin can be freely controlled by these compare-match B signals and the settings of output select bits OS3 and OS2 in TCSR. TCORB is initialized to H'FF.

### 13.3.4 Timer Control Register (TCR)

TCR selects the TCNT clock source and the condition by which TCNT is cleared, and enables/disables interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7	CMIEB	0	R/W	<p>Compare-Match Interrupt Enable B</p> <p>Selects whether the CMFB interrupt request (CMIB) is enabled or disabled when the CMFB flag in TCSR is set to 1.</p> <p>0: CMFB interrupt request (CMIB) is disabled 1: CMFB interrupt request (CMIB) is enabled</p>
6	CMIEA	0	R/W	<p>Compare-Match Interrupt Enable A</p> <p>Selects whether the CMFA interrupt request (CMIA) is enabled or disabled when the CMFA flag in TCSR is set to 1.</p> <p>0: CMFA interrupt request (CMIA) is disabled 1: CMFA interrupt request (CMIA) is enabled</p>
5	OVIE	0	R/W	<p>Timer Overflow Interrupt Enable</p> <p>Selects whether the OVF interrupt request (OVI) is enabled or disabled when the OVF flag in TCSR is set to 1.</p> <p>0: OVF interrupt request (OVI) is disabled 1: OVF interrupt request (OVI) is enabled</p>
4	CCLR1	0	R/W	Counter Clear 1, 0
3	CCLR0	0	R/W	<p>These bits select the method by which the timer counter is cleared.</p> <p>00: Clearing is disabled 01: Cleared on compare-match A 10: Cleared on compare-match B 11: Cleared on rising edge of external reset input</p>
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	These bits select the clock input to TCNT and count condition, together with the ICKS1 and ICKS0 bits in STCR. For details, see table 13.2.
0	CKS0	0	R/W	

**Table 13.2 Clock Input to TCNT and Count Condition**

Channel	TCR			STCR		Description
	CKS2	CKS1	CKS0	ICKS1	ICKS0	
TMR_0	0	0	0	—	—	Disables clock input
	0	0	1	—	0	Increments at falling edge of internal clock $\phi/8$
	0	0	1	—	1	Increments at falling edge of internal clock $\phi/2$
	0	1	0	—	0	Increments at falling edge of internal clock $\phi/64$
	0	1	0	—	1	Increments at falling edge of internal clock $\phi/32$
	0	1	1	—	0	Increments at falling edge of internal clock $\phi/1024$
	0	1	1	—	1	Increments at falling edge of internal clock $\phi/256$
	1	0	0	—	—	Increments at overflow signal from TCNT_1*
TMR_1	0	0	0	—	—	Disables clock input
	0	0	1	0	—	Increments at falling edge of internal clock $\phi/8$
	0	0	1	1	—	Increments at falling edge of internal clock $\phi/2$
	0	1	0	0	—	Increments at falling edge of internal clock $\phi/64$
	0	1	0	1	—	Increments at falling edge of internal clock $\phi/128$
	0	1	1	0	—	Increments at falling edge of internal clock $\phi/1024$
	0	1	1	1	—	Increments at falling edge of internal clock $\phi/2048$
	1	0	0	—	—	Increments at compare-match A from TCNT_0*
TMR_Y	0	0	0	—	—	Disables clock input
	0	0	1	—	—	Increments at falling edge of internal clock $\phi/4$
	0	1	0	—	—	Increments at falling edge of internal clock $\phi/256$
	0	1	1	—	—	Increments at falling edge of internal clock $\phi/2048$
	1	0	0	—	—	Setting prohibited
TMR_X	0	0	0	—	—	Disables clock input
	0	0	1	—	—	Increments at falling edge of internal clock $\phi$
	0	1	0	—	—	Increments at falling edge of internal clock $\phi/2$
	0	1	1	—	—	Increments at falling edge of internal clock $\phi/4$
	1	0	0	—	—	Setting prohibited
Common	1	0	1	—	—	Increments at rising edge of external clock
	1	1	0	—	—	Increments at falling edge of external clock
	1	1	1	—	—	Increments at both rising and falling edges of external clock.

Legend: —: Don't care

Note: \* If the TMR\_0 clock input is set as the TCNT\_1 overflow signal and the TMR\_1 clock input is set as the TCNT\_0 compare-match signal simultaneously, a count-up clock cannot be generated. Simultaneous setting of this condition should therefore be avoided.

### 13.3.5 Timer Control/Status Register (TCSR)

TCSR indicates the status flags and controls compare-match output.

#### TCSR\_0

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*	Compare-Match Flag B [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_0 and TCORB_0 match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFB when CMFB = 1, then write 0 in CMFB</li> </ul>
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_0 and TCORA_0 match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFA when CMFA = 1, then write 0 in CMFA</li> </ul>
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] <ul style="list-style-type: none"> <li>When TCNT_0 overflows from H'FF to H'00</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read OVF when OVF = 1, then write 0 in OVF</li> </ul>
4	ADTE	0	R/W	A/D Trigger Enable Enables or disables A/D converter start requests by compare-match A. 0: A/D converter start requests by compare-match A are disabled 1: A/D converter start requests by compare-match A are enabled

Bit	Bit Name	Initial Value	R/W	Description
3	OS3	0	R/W	Output Select 3, 2
2	OS2	0	R/W	These bits specify how the TMO0 pin output level is to be changed by compare-match B of TCORB_0 and TCNT_0.  00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)
1	OS1	0	R/W	Output Select 1, 0
0	OS0	0	R/W	These bits specify how the TMO0 pin output level is to be changed by compare-match A of TCORA_0 and TCNT_0.  00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)

Note: \* Only 0 can be written, for flag clearing.

### TCSR\_1

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*	Compare-Match Flag B [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_1 and TCORB_1 match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFB when CMFB = 1, then write 0 in CMFB</li> </ul>
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_1 and TCORA_1 match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFA when CMFA = 1, then write 0 in CMFA</li> </ul>



Bit	Bit Name	Initial Value	R/W	Description
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] <ul style="list-style-type: none"> <li>When TCNT_1 overflows from H'FF to H'00</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read OVF when OVF = 1, then write 0 in OVF</li> </ul>
4	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
3	OS3	0	R/W	Output Select 3, 2
2	OS2	0	R/W	These bits specify how the TMO1 pin output level is to be changed by compare-match B of TCORB_1 and TCNT_1. 00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)
1	OS1	0	R/W	Output Select 1, 0
0	OS0	0	R/W	These bits specify how the TMO1 pin output level is to be changed by compare-match A of TCORA_1 and TCNT_1. 00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)

Note: \* Only 0 can be written, for flag clearing.

**TCSR\_X**

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	CMFB	0	R/(W)*	Compare-Match Flag B [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_X and TCORB_X match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFB when CMFB = 1, then write 0 in CMFB</li> </ul>
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_X and TCORA_X match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFA when CMFA = 1, then write 0 in CMFA</li> </ul>
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] <ul style="list-style-type: none"> <li>When TCNT_X overflows from H'FF to H'00</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read OVF when OVF = 1, then write 0 in OVF</li> </ul>
4	ICF	0	R/(W)*	Input Capture Flag [Setting condition] <ul style="list-style-type: none"> <li>When a rising edge and falling edge is detected in the external reset signal in that order, after the ICST bit in TCONRI of the timer connection is set to 1</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read ICF when ICF = 1, then write 0 in ICF</li> </ul>
3	OS3	0	R/W	Output Select 3, 2
2	OS2	0	R/W	These bits specify how the TMOX pin output level is to be changed by compare-match B of TCORB_X and TCNT_X. 00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)

Bit	Bit Name	Initial Value	R/W	Description
1	OS1	0	R/W	Output Select 1, 0
0	OS0	0	R/W	These bits specify how the TMOX pin output level is to be changed by compare-match A of TCORA_X and TCNT_X.  00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)

Note: \* Only 0 can be written, for flag clearing.

### TCSR\_Y

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*	Compare-Match Flag B [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_Y and TCORB_Y match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFB when CMFB = 1, then write 0 in CMFB</li> </ul>
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] <ul style="list-style-type: none"> <li>When the values of TCNT_Y and TCORA_Y match</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read CMFA when CMFA = 1, then write 0 in CMFA</li> </ul>
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] <ul style="list-style-type: none"> <li>When TCNT_Y overflows from H'FF to H'00</li> </ul> [Clearing condition] <ul style="list-style-type: none"> <li>Read OVF when OVF = 1, then write 0 in OVF</li> </ul>
4	ICIE	1	R/(W)*	Input Capture Interrupt Enable Enables or disables the ICF interrupt request (ICIX) when the ICF bit in TCSR_X is set to 1. 0: ICF interrupt request (ICIX) is disabled 1: ICF interrupt request (ICIX) is enabled

Bit	Bit Name	Initial Value	R/W	Description
3	OS3	0	R/W	Output Select 3, 2
2	OS2	0	R/W	These bits specify how the TMOY pin output level is to be changed by compare-match B of TCORB_Y and TCNT_Y. 00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)
1	OS1	0	R/W	Output Select 1, 0
0	OS0	0	R/W	These bits specify how the TMOY pin output level is to be changed by compare-match A of TCORA_Y and TCNT_Y. 00: No change 01: 0 is output 10: 1 is output 11: Output is inverted (toggle output)

Note: \* Only 0 can be written, for flag clearing.

### 13.3.6 Input Capture Register (TICR)

TICR is an 8-bit register. The contents of TCNT are transferred to TICR at the rising edge of the external reset input. TICR cannot be directly accessed by the CPU. The TICR function is used for the timer connection. For details, see section 14, Timer Connection.

### 13.3.7 Time Constant Register (TCORC)

TCORC is an 8-bit readable/writable register. The sum of contents of TCORC and TICR is always compared with TCNT. When a match is detected, a compare-match C signal is generated. However, comparison at the T2 state in the write cycle to TCORC and at the input capture cycle of TICR is disabled. TCORC is initialized to H'FF. The TCORC function is used for the timer connection. For details, see section 14, Timer Connection.

### 13.3.8 Input Capture Registers R and F (TICRR and TICRF)

TICRR and TICRF are 8-bit read-only registers. The contents of TCNT are transferred at the rising edge and falling edge of the external reset input in that order, when the ICST bit in TCONRI of the timer connection is set to 1. The ICST bit is cleared to 0 when one capture operation ends. TICRR and TICRF are initialized to H'00. The TICRR and TICRF functions are used for timer connection. For details, see section 14, Timer Connection.

### 13.3.9 Timer Input Select Register (TISR)

TISR selects a signal source of external clock/reset input for the counter.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 1	R/(W)	Reserved The initial value should not be changed.
0	IS	0	R/W	Input Select Selects an internal synchronization signal (IVG signal) or timer clock/reset input pin (TMIY or ExTMIY) as the signal source of external clock/reset input for the TMR counter. 0: IVG signal is selected 1: TMIY or ExTMIY (TMCIY/TMRIY) is selected

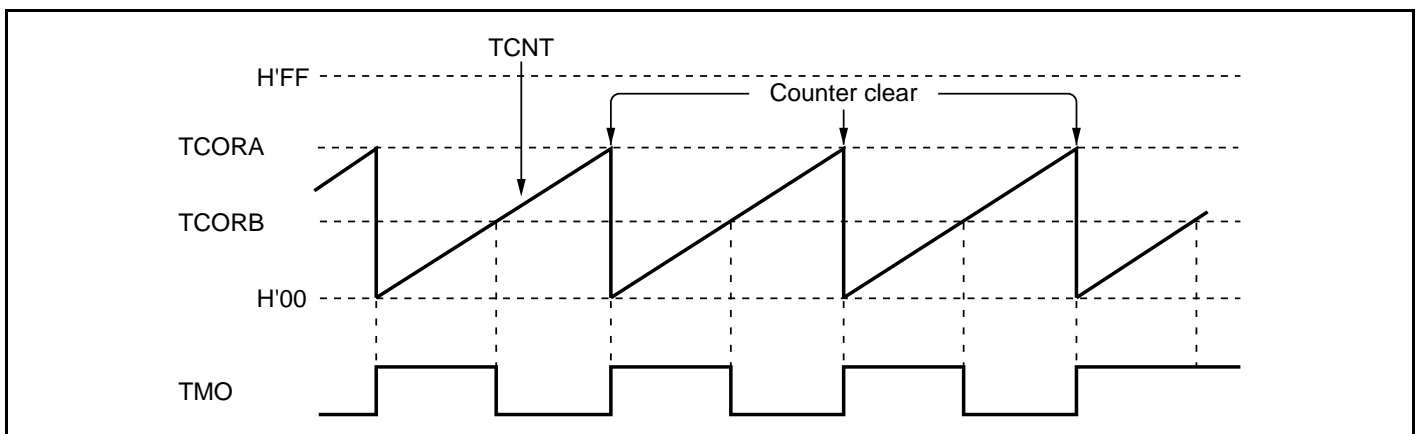
## 13.4 Operation

### 13.4.1 Pulse Output

Figure 13.3 shows an example for outputting an arbitrary duty pulse.

1. Clear the CCLR1 bit in TCR to 0 so that TCNT is cleared according to the compare match of TCORA, and then set the CCLR0 bit to 1.
2. Set the OS3 to OS0 bits in TCSR to B'0110 so that 1 is output according to the compare match of TCORA and 0 is output according to the compare match of TCORB.

According to the above settings, the waveforms with the TCORA cycle and TCORB pulse width can be output without the intervention of software.

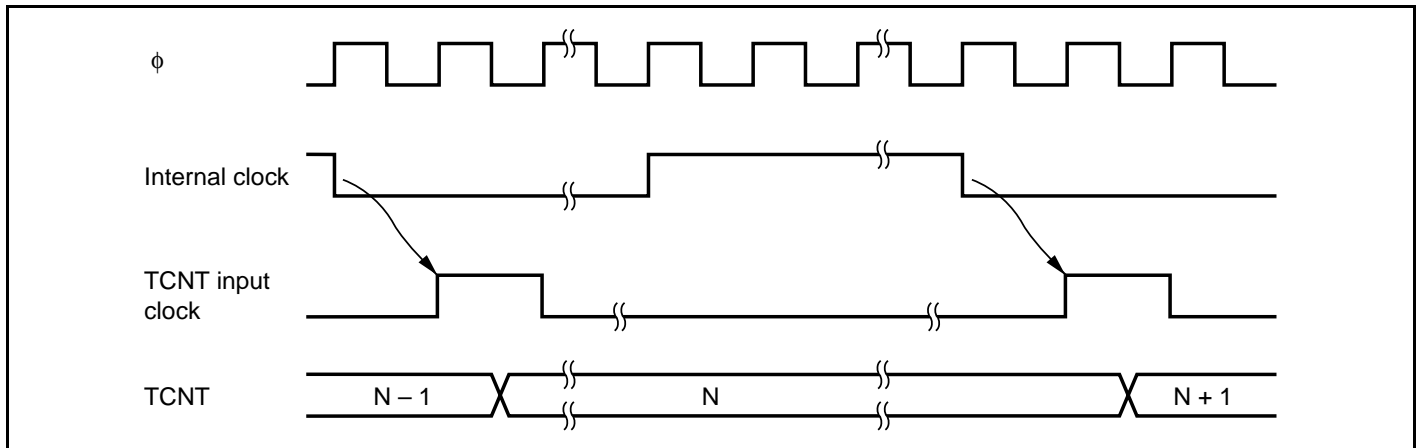


**Figure 13.3 Pulse Output Example**

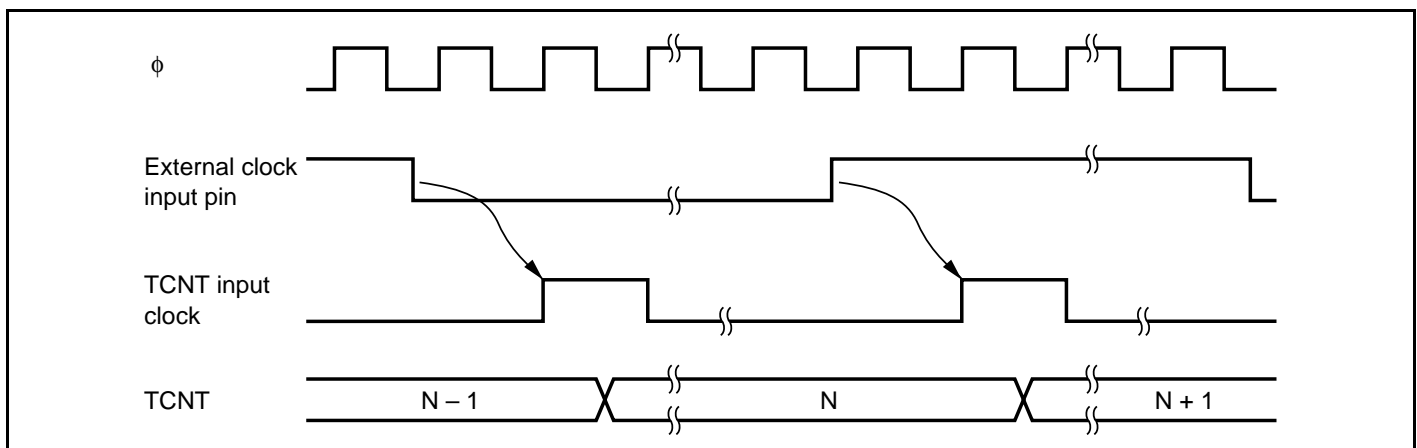
## 13.5 Operation Timing

### 13.5.1 TCNT Count Timing

Figure 13.4 shows the TCNT count timing with an internal clock source. Figure 13.5 shows the TCNT count timing with an external clock source. The pulse width of the external clock signal must be at least 1.5 system clocks ( $\phi$ ) for a single edge and at least 2.5 system clocks ( $\phi$ ) for both edges. The counter will not increment correctly if the pulse width is less than these values.



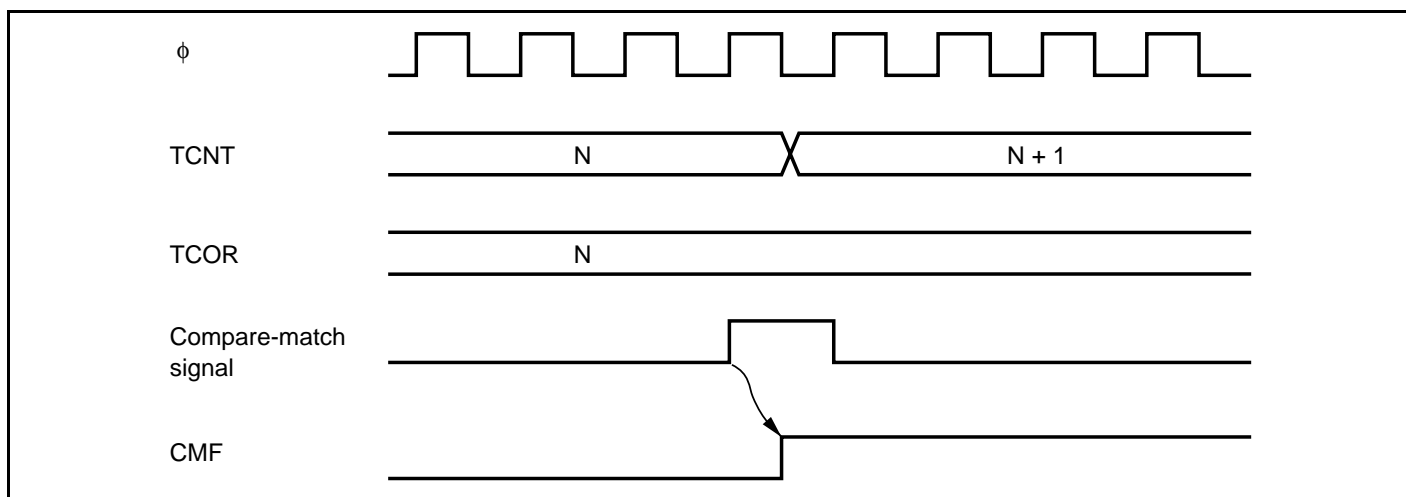
**Figure 13.4 Count Timing for Internal Clock Input**



**Figure 13.5 Count Timing for External Clock Input**

### 13.5.2 Timing of CMFA and CMFB Setting at Compare-Match

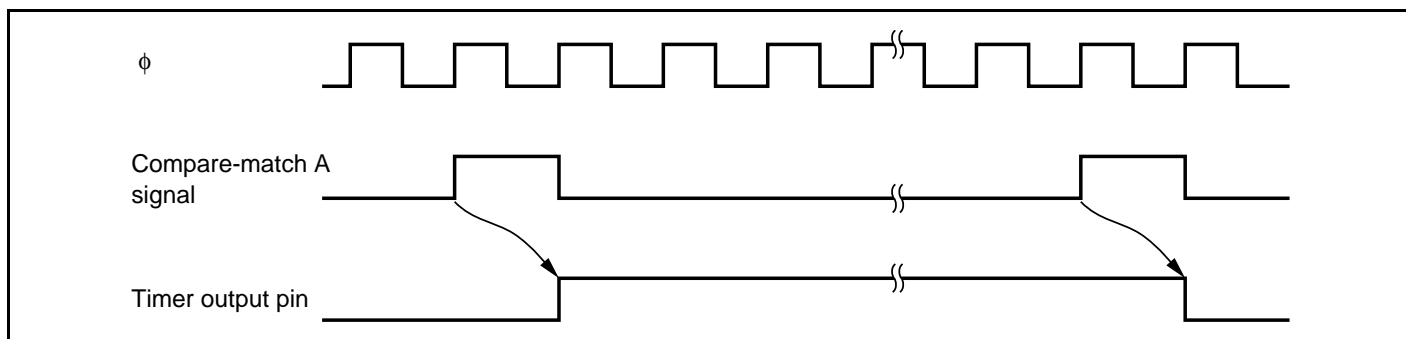
The CMFA and CMFB flags in TCSR are set to 1 by a compare-match signal generated when the TCNT and TCOR values match. The compare-match signal is generated at the last state in which the match is true, just when the timer counter is updated. Therefore, when TCNT and TCOR match, the compare-match signal is not generated until the next TCNT input clock. Figure 13.6 shows the timing of CMF flag setting.



**Figure 13.6 Timing of CMF Setting at Compare-Match**

### 13.5.3 Timing of Timer Output at Compare-Match

When a compare-match signal occurs, the timer output changes as specified by the OS3 to OS0 bits in TCSR. Figure 13.7 shows the timing of timer output when the output is set to toggle by a compare-match A signal.

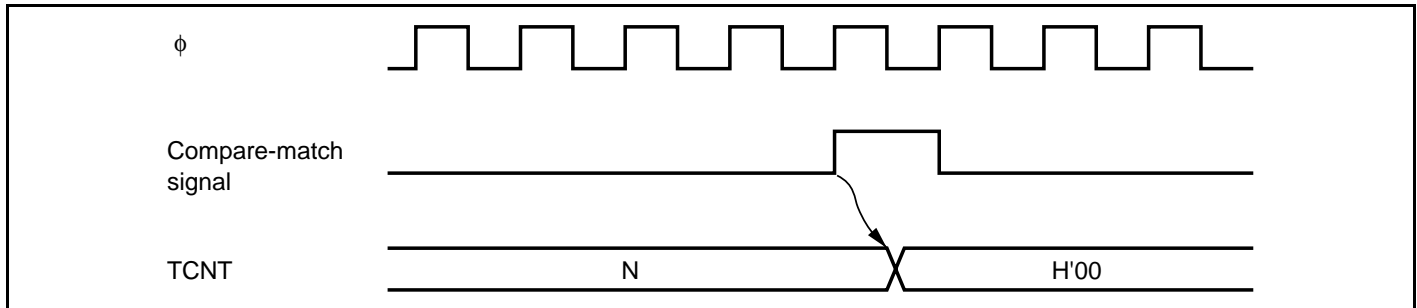


**Figure 13.7 Timing of Toggled Timer Output by Compare-Match A Signal**



### 13.5.4 Timing of Counter Clear at Compare-Match

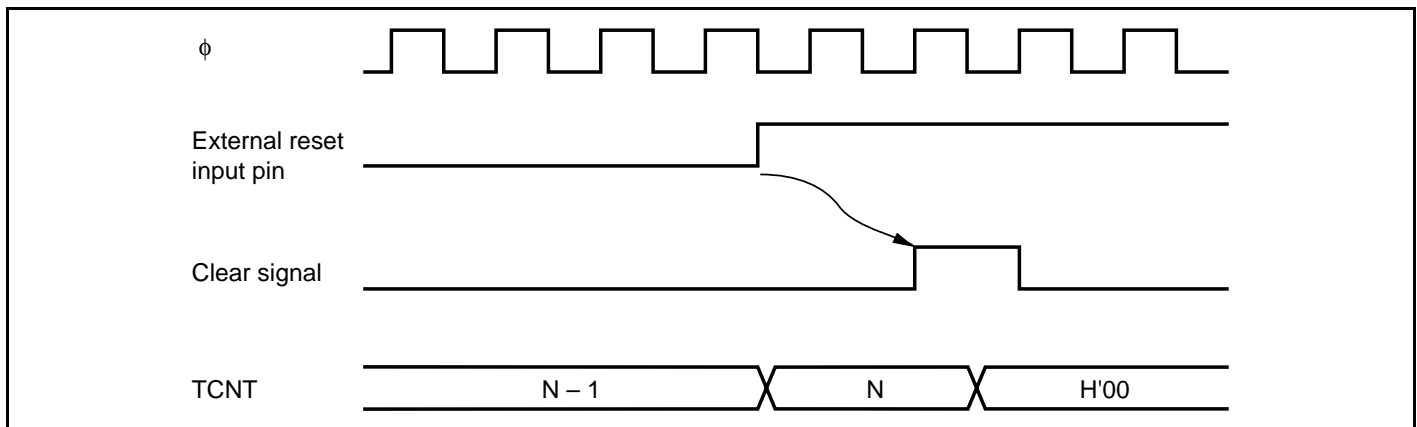
TCNT is cleared when compare-match A or compare-match B occurs, depending on the setting of the CCLR1 and CCLR0 bits in TCR. Figure 13.8 shows the timing of clearing the counter by a compare-match.



**Figure 13.8 Timing of Counter Clear by Compare-Match**

### 13.5.5 TCNT External Reset Timing

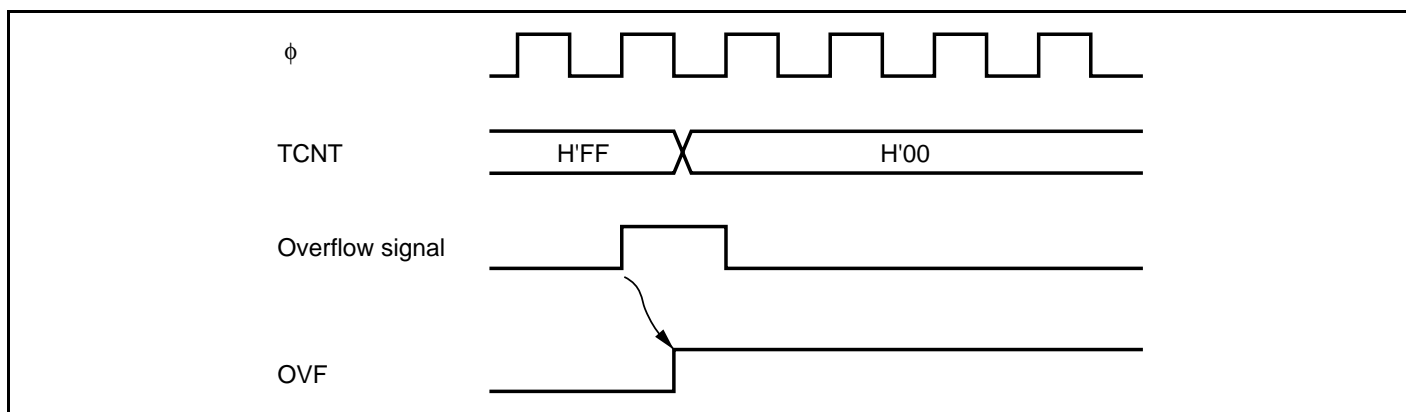
TCNT is cleared at the rising edge of an external reset input, depending on the settings of the CCLR1 and CCLR0 bits in TCR. The width of the clearing pulse must be at least 1.5 states. Figure 13.9 shows the timing of clearing the counter by an external reset input.



**Figure 13.9 Timing of Counter Clear by External Reset Input**

### 13.5.6 Timing of Overflow Flag (OVF) Setting

The OVF bit in TCSR is set to 1 when the TCNT overflows (changes from H'FF to H'00). Figure 13.10 shows the timing of OVF flag setting.



**Figure 13.10 Timing of OVF Flag Setting**

## 13.6 TMR\_0 and TMR\_1 Cascaded Connection

If bits CKS2 to CKS0 in either TCR\_0 or TCR\_1 are set to B'100, the 8-bit timers of the two channels are cascaded. With this configuration, a single 16-bit timer can be used (16-bit count mode) or the compare-matches of the 8-bit timer of channel 0 can be counted by the 8-bit timer of channel 1 (compare-match count mode).

### 13.6.1 16-Bit Count Mode

When bits CKS2 to CKS0 in TCR\_0 are set to B'100, the timer functions as a single 16-bit timer with channel 0 occupying the upper 8 bits and channel 1 occupying the lower 8 bits.

- Setting of compare-match flags
  - The CMF flag in TCSR\_0 is set to 1 when a 16-bit compare-match occurs.
  - The CMF flag in TCSR\_1 is set to 1 when a lower 8-bit compare-match occurs.
- 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 occurs. The 16-bit counter (TCNT\_0 and TCNT\_1 together) is also cleared when counter clear by the TMI0 pin has been set.
  - The settings of the CCLR1 and CCLR0 bits in TCR\_1 are ignored. The lower 8 bits cannot be cleared independently.
- Pin output
  - Control of output from the TMO0 pin by 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 bits OS3 to OS0 in TCSR\_1 is in accordance with the lower 8-bit compare-match conditions.

### 13.6.2 Compare-Match Count Mode

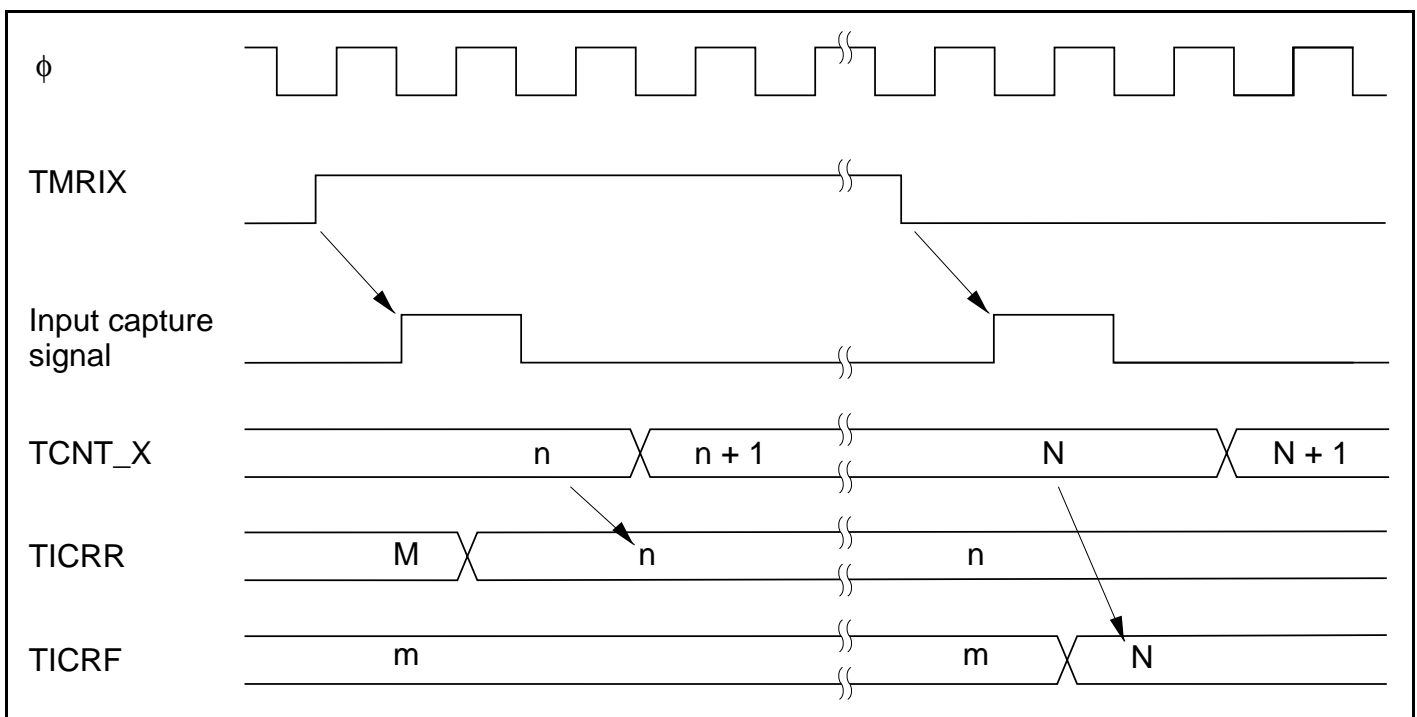
When bits CKS2 to CKS0 in TCR\_1 are B'100, TCNT\_1 counts the occurrence of compare-match A for channel 0. Channels 0 and 1 are controlled independently. Conditions such as setting of the CMF flag, generation of interrupts, output from the TMO pin, and counter clearing are in accordance with the settings for each channel.

## 13.7 Input Capture Operation

TMR\_X has input capture registers (TICR, TICRR and TICRF). A narrow pulse width can be measured with TICRR and TICRF, using a single capture operation controlled by the ICST bit in TCONRI of the timer connection. If the falling edge of TMRIX (TMR\_X input capture input signal) is detected after its rising edge has been detected while the ICST bit is set to 1, the value of TCNT\_X at that time is transferred to both TICRR and TICRF, and the ICST bit is cleared to 0.

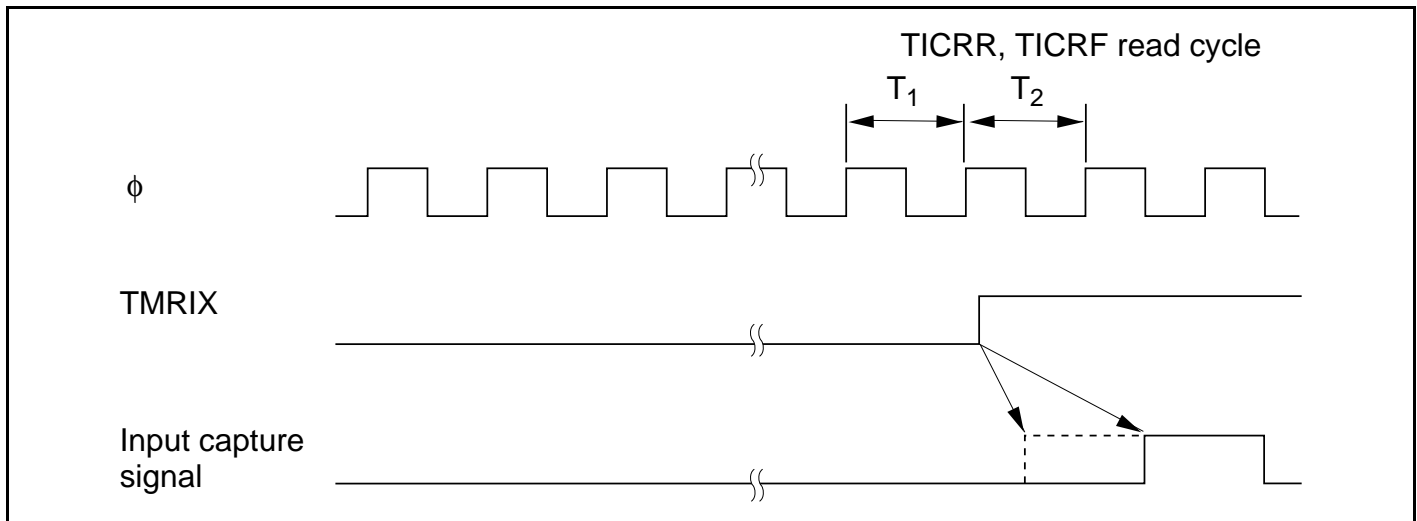
The TMRIX input signal can be switched by the setting of the other bits in TCONRI.

**Input Capture Signal Input Timing:** Figure 13.11 shows the timing of the input capture operation.



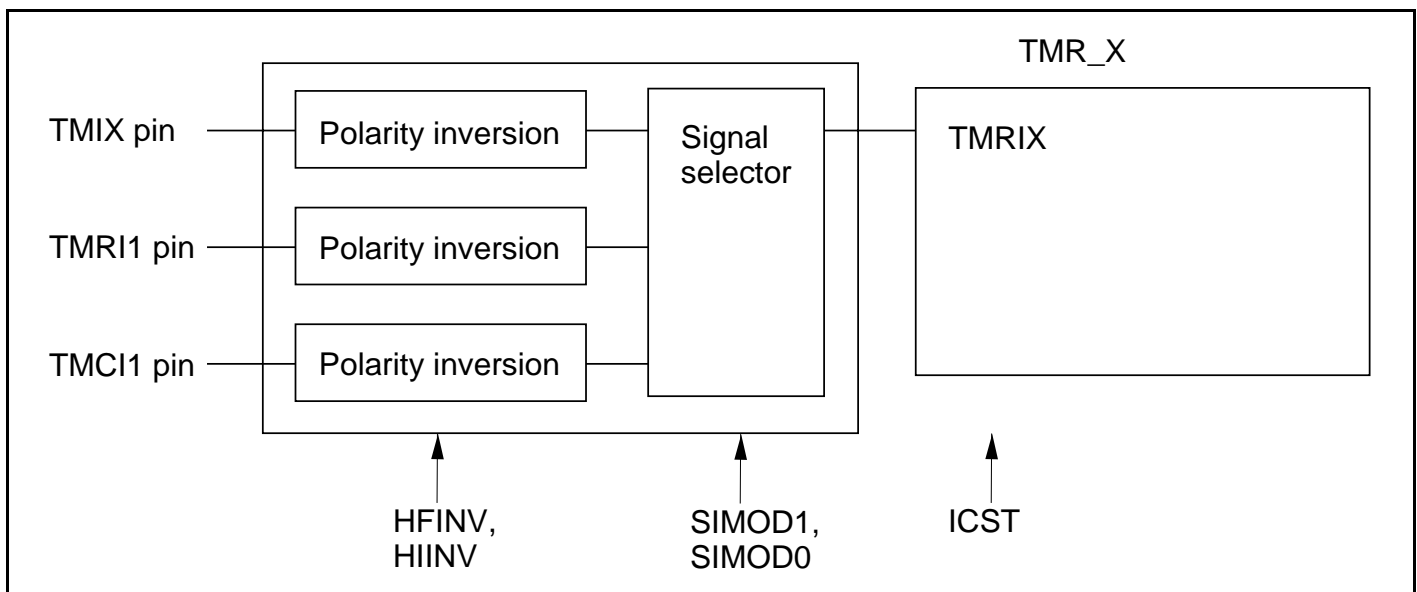
**Figure 13.11 Timing of Input Capture Operation**

If the input capture signal is input while TICRR and TICRF are being read, the input capture signal is delayed by one system clock ( $\phi$ ) cycle. Figure 13.12 shows the timing of this operation.



**Figure 13.12 Timing of Input Capture Signal  
(Input Capture Signal Is Input during TICRR and TICRF Read)**

**Selection of Input Capture Signal Input:** TMRX (input capture input signal of TMR\_X) is switched according to the setting of the bits in TCONRI of the timer connection. Input capture signal selections are shown in figure 13.13 and table 13.3. For details, see section 14.3.1, Timer Connection Register I (TCONRI).



**Figure 13.13 Input Capture Signal Selection**

**Table 13.3 Input Capture Signal Selection**

TCONRI					Description	
Bit 4	Bit 7	Bit 6	Bit 3	Bit 1		
ICST	SIMOD1	SIMOD0	HFINV	HIINV		
0	—	—	—	—	Input capture function not used	
1	0	0	0	—	TMIX pin input selection	
			1	—	Inverted TMIX pin input selection	
		1	—	0	TMRI1 pin input selection	
			—	1	Inverted TMRI1 pin input selection	
		1	1	—	0	TMC11 pin input selection
				—	1	Inverted TMC11 pin input selection

Legend:

—: Don't care

## 13.8 Interrupt Sources

TMR\_0, TMR\_1, and TMR\_Y can generate three types of interrupts: CMIA, CMIB, and OVI. TMR\_X can generate four types of interrupts: CMIA, CMIB, OVI, and ICIX. Table 13.4 shows the interrupt sources and priorities. Each interrupt source can be enabled or disabled independently by interrupt enable bits in TCR or TCSR. Independent signals are sent to the interrupt controller for each interrupt.

The CMIA and CMIB interrupts can be used as DTC activation interrupt sources.

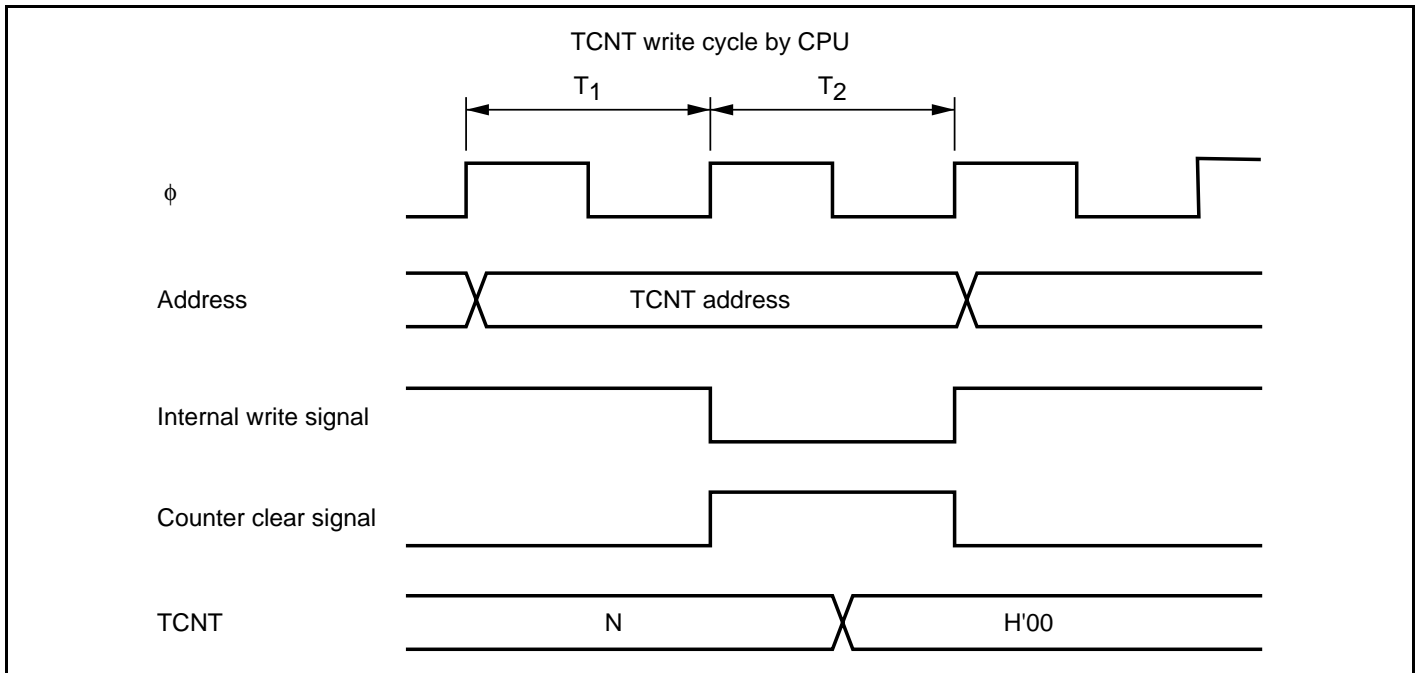
**Table 13.4 Interrupt Sources of 8-Bit Timers TMR\_0, TMR\_1, TMR\_Y, and TMR\_X**

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	Interrupt Priority
TMR_X	CMIA_X	TCORA_X compare-match	CMFA	Possible	High ↑
	CMIB_X	TCORB_X compare-match	CMFB	Possible	
	OVI_X	TCNT_X overflow	OVF	Not possible	
	ICIX	Input capture	ICF	Not possible	
TMR_0	CMIA0	TCORA_0 compare-match	CMFA	Possible	
	CMIB0	TCORB_0 compare-match	CMFB	Possible	
	OVI0	TCNT_0 overflow	OVF	Not possible	
TMR_1	CMIA1	TCORA_1 compare-match	CMFA	Possible	
	CMIB1	TCORB_1 compare-match	CMFB	Possible	
	OVI1	TCNT_1 overflow	OVF	Not possible	
TMR_Y	CMIA_Y	TCORA_Y compare-match	CMFA	Possible	Low
	CMIB_Y	TCORB_Y compare-match	CMFB	Possible	
	OVI_Y	TCNT_Y overflow	OVF	Not possible	

## 13.9 Usage Notes

### 13.9.1 Conflict between TCNT Write and Clear

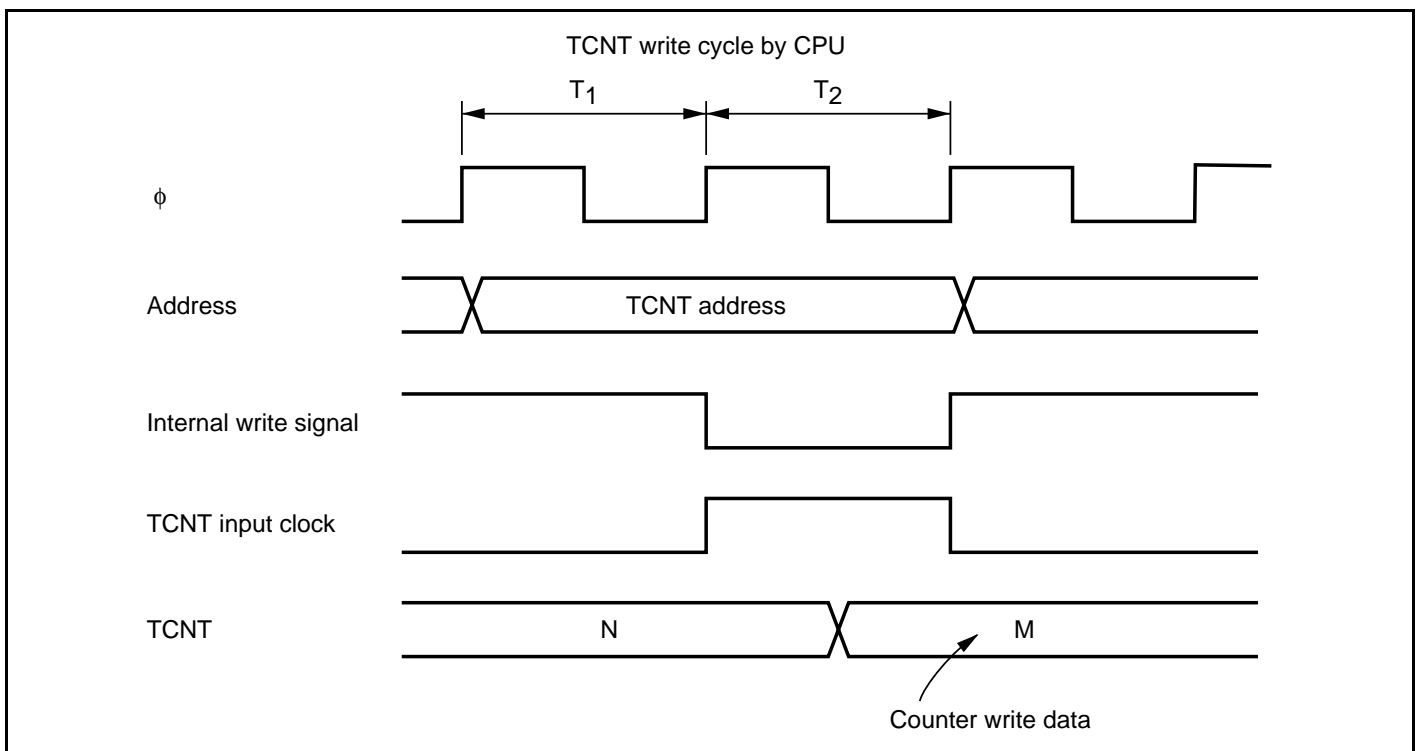
If a counter clear signal is generated during the T2 state of a TCNT write cycle as shown in figure 13.14, clearing takes priority, so that the counter is cleared and the write is not performed.



**Figure 13.14 Conflict between TCNT Write and Clear**

### 13.9.2 Conflict between TCNT Write and Increment

If a TCNT input clock is generated during the  $T_2$  state of a TCNT write cycle as shown in figure 13.15, the write takes priority and the counter is not incremented.

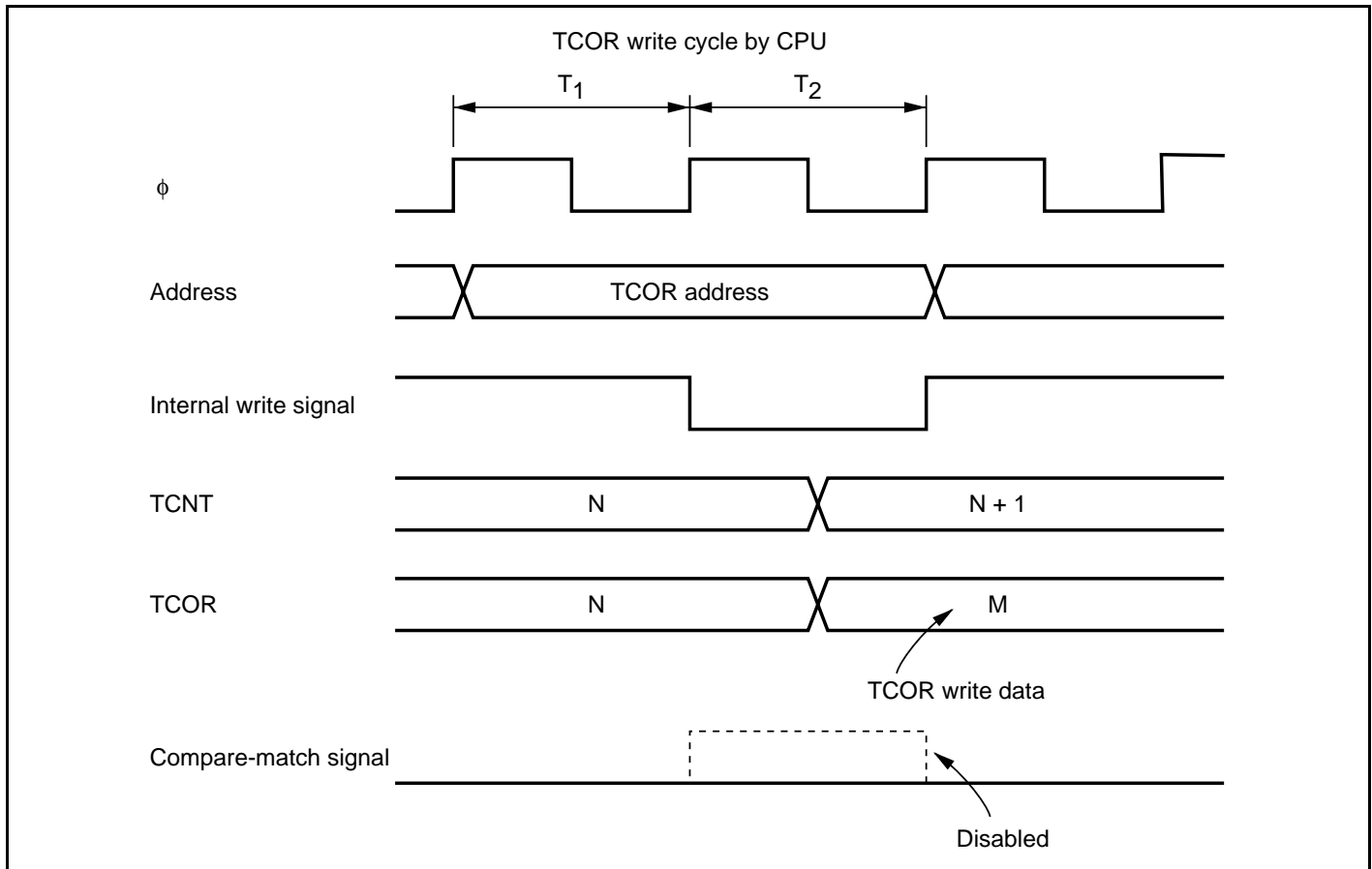


**Figure 13.15 Conflict between TCNT Write and Increment**



### 13.9.3 Conflict between TCOR Write and Compare-Match

If a compare-match occurs during the T2 state of a TCOR write cycle as shown in figure 13.16, the TCOR write takes priority and the compare-match signal is disabled. With TMR\_X, a TCCR input capture conflicts with a compare-match in the same way as with a write to TCORC. In this case also, the input capture takes priority and the compare-match signal is disabled.



**Figure 13.16 Conflict between TCOR Write and Compare-Match**

### 13.9.4 Conflict between Compare-Matches A and B

If compare-matches A and B occur at the same time, the 8-bit timer operates in accordance with the priorities for the output states set for compare-match A and compare-match B, as shown in table 13.5.

**Table 13.5 Timer Output Priorities**

Output Setting	Priority
Toggle output	High
1 output	↑
0 output	
No change	Low

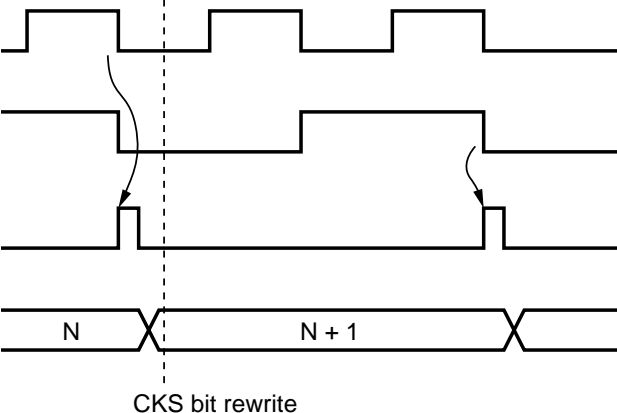
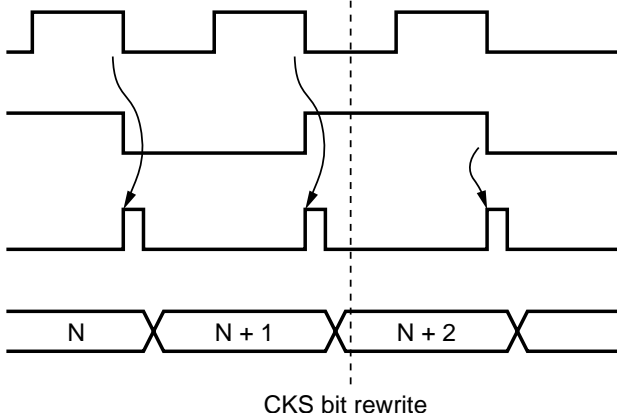
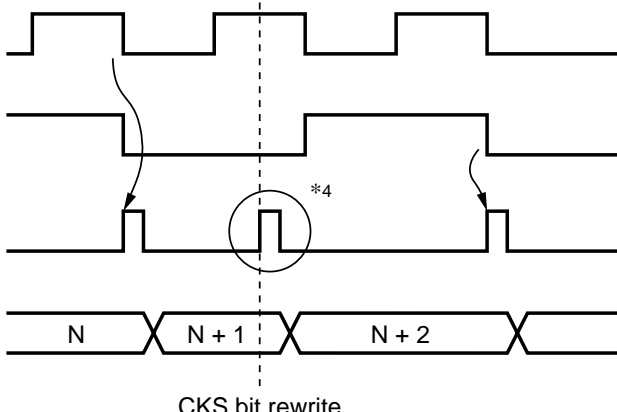
### 13.9.5 Switching of Internal Clocks and TCNT Operation

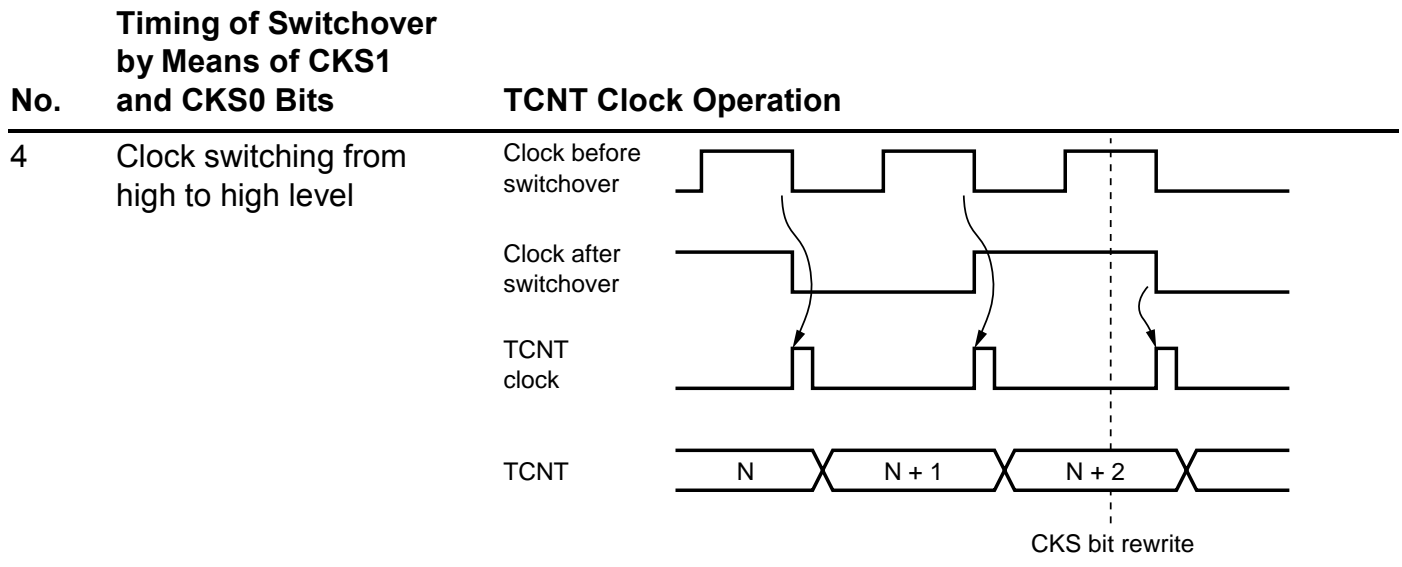
TCNT may increment erroneously when the internal clock is switched over. Table 13.6 shows the relationship between the timing at which the internal clock is switched (by writing to the CKS1 and CKS0 bits) and the TCNT operation.

When the TCNT clock is generated from an internal clock, the falling edge of the internal clock pulse is detected. If clock switching causes a change from high to low level, as shown in no. 3 in table 13.6, a TCNT clock pulse is generated on the assumption that the switchover is a falling edge, and TCNT is incremented.

Erroneous incrementation can also happen when switching between internal and external clocks.

Table 13.6 Switching of Internal Clocks and TCNT Operation

No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	TCNT Clock Operation
1	Clock switching from low to low level* <sup>1</sup>	<p data-bbox="618 321 760 373">Clock before switchover</p> <p data-bbox="618 422 737 474">Clock after switchover</p> <p data-bbox="618 516 683 569">TCNT clock</p> <p data-bbox="618 636 683 663">TCNT</p> <p data-bbox="915 709 1081 737">CKS bit rewrite</p> 
2	Clock switching from low to high level* <sup>2</sup>	<p data-bbox="618 783 760 835">Clock before switchover</p> <p data-bbox="618 884 737 936">Clock after switchover</p> <p data-bbox="618 978 683 1031">TCNT clock</p> <p data-bbox="618 1098 683 1125">TCNT</p> <p data-bbox="1045 1167 1211 1194">CKS bit rewrite</p> 
3	Clock switching from high to low level* <sup>3</sup>	<p data-bbox="618 1245 760 1297">Clock before switchover</p> <p data-bbox="618 1346 737 1398">Clock after switchover</p> <p data-bbox="618 1440 683 1493">TCNT clock</p> <p data-bbox="618 1560 683 1587">TCNT</p> <p data-bbox="964 1629 1130 1656">CKS bit rewrite</p> <p data-bbox="1084 1423 1117 1451">*<sup>4</sup></p> 



- Notes:
1. Includes switching from low to stop, and from stop to low.
  2. Includes switching from stop to high.
  3. Includes switching from high to stop.
  4. Generated on the assumption that the switchover is a falling edge; TCNT is incremented.

### 13.9.6 Mode Setting with Cascaded Connection

If the 16-bit count mode and compare-match count mode are set simultaneously, the input clock pulses for TCNT\_0 and TCNT\_1 are not generated, and thus the counters will stop operating. Simultaneous setting of these two modes should therefore be avoided.

## Section 14 Timer Connection

This LSI allows interconnection between a 16-bit free-running timer (FRT) and three 8-bit timer channels (TMR\_1, TMR\_X, and TMR\_Y). This capability can be used to implement complex functions such as PWM decoding and clamp waveform output.

### 14.1 Features

- Five input pins and four output pins, all of which can be designated for phase inversion.
- Positive logic is assumed for all signals used within the timer connection facility.
- An edge-detection circuit is connected to the input pins, simplifying signal input detection.
- TMR\_X can be used for PWM input signal decoding.
- TMR\_X can be used for clamp waveform generation.
- An external clock signal divided by TMR\_1 can be used as the FRT capture input signal.
- An internal synchronization signal can be generated using the FRT and TMR\_Y.
- A signal generated/modified using an input signal and timer connection can be selected and output.

Figure 14.1 shows a block diagram of the timer connection facility.

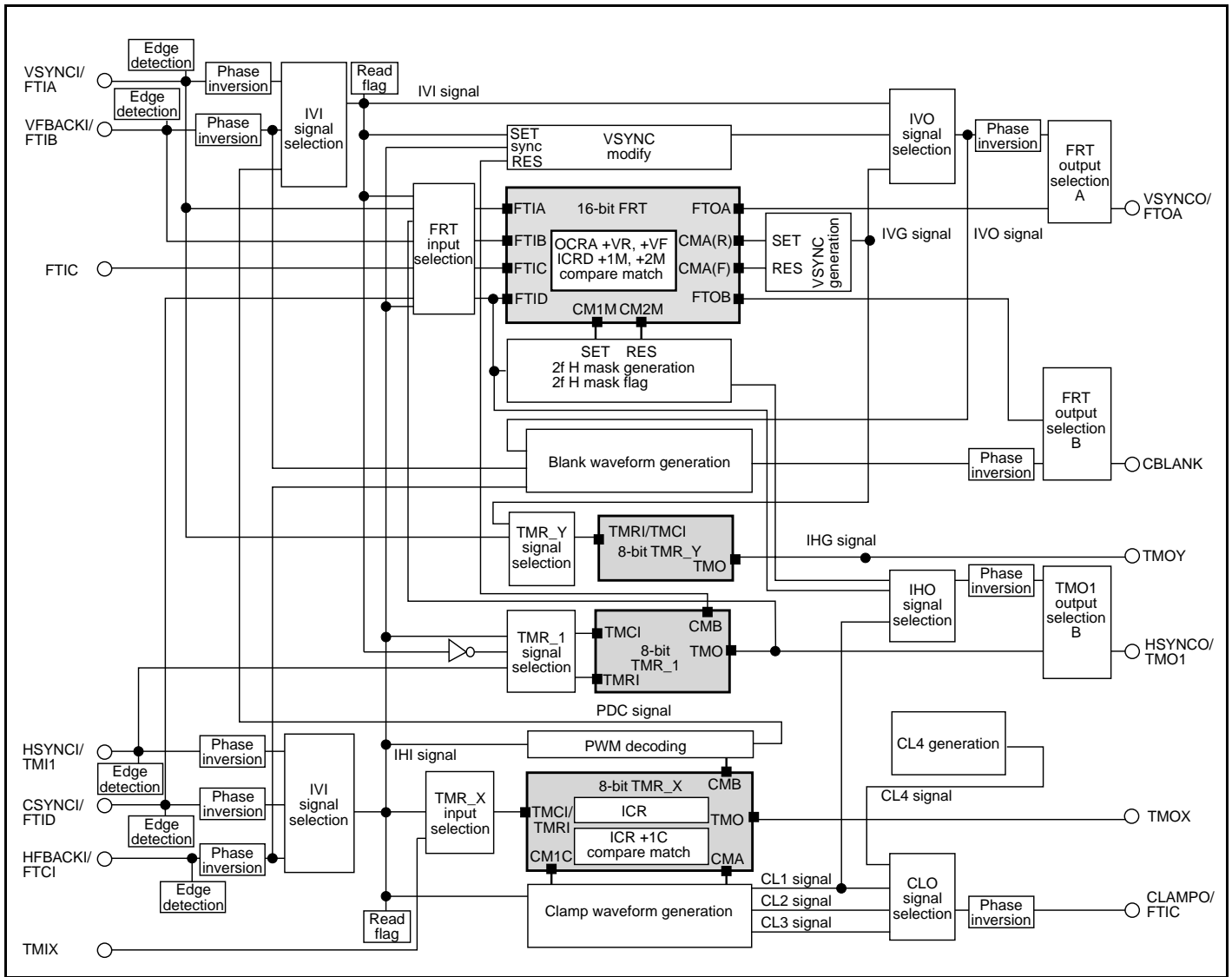


Figure 14.1 Block Diagram of Timer Connection

## 14.2 Input/Output Pins

Table 14.1 lists the timer connection input and output pins.

**Table 14.1 Pin Configuration**

Name	Abbreviation	Input/ Output	Function
Vertical synchronization signal input pin	VSYNCI	Input	Vertical synchronization signal input pin or FTIA input pin
Horizontal synchronization signal input pin	HSYNCI	Input	Horizontal synchronization signal input pin or TMI1 input pin
Composite synchronization signal input pin	CSYNCI	Input	Composite synchronization signal input pin or FTID input pin
Spare vertical synchronization signal input pin	VFBACKI	Input	Spare vertical synchronization signal input pin or FTIB input pin
Spare horizontal synchronization signal input pin	HFBACKI	Input	Spare horizontal synchronization signal input pin or FTIC input pin
Vertical synchronization signal output pin	VSYNCO	Output	Vertical synchronization signal output pin or FTOA output pin
Horizontal synchronization signal output pin	HSYNCO	Output	Horizontal synchronization signal output pin or TMO1 output pin
Clamp waveform output pin	CLAMPO	Output	Clamp waveform output pin or FTIC input pin
Blanking waveform output pin	CBLANK	Output	Blanking waveform output pin or FTOB output pin

## 14.3 Register Descriptions

The timer connection has the following registers.

- Timer connection register I (TCONRI)
- Timer connection register O (TCONRO)
- Timer connection register S (TCONRS)
- Edge sense register (SEDGR)

### 14.3.1 Timer Connection Register I (TCONRI)

TCONRI controls connection between timers, the signal source for synchronization signal input, phase inversion, etc.

Bit	Bit Name	Initial Value	R/W	Description
7	SIMOD1	0	R/W	Input Synchronization Mode Select 1, 0
6	SIMOD0	0	R/W	These bits select the signal source of the IHI and IVI signals. <ul style="list-style-type: none"> <li>• Mode               <ul style="list-style-type: none"> <li>00: No signal</li> <li>01: S-on-G mode</li> <li>10: Composite mode</li> <li>11: Separate mode</li> </ul> </li> <li>• IHI Signal               <ul style="list-style-type: none"> <li>00: HFBACKI input</li> <li>01: CSYNCI input</li> <li>1X: HSYNCI input</li> </ul> </li> <li>• IVI Signal               <ul style="list-style-type: none"> <li>00: VFBACKI input</li> <li>01: PDC input</li> <li>10: PDC input</li> <li>11: VSYNCI input</li> </ul> </li> </ul>
5	SCONE	0	R/W	Synchronization Signal Connection Enable Selects the signal source of the FIT input of the FRT, the TMI1 input of TMR_1, the TMIX input of TMR_X, and the TMIY input of TMR_Y. For details, see table 14.2.



Bit	Bit Name	Initial Value	R/W	Description
4	ICST	0	R/W	<p>Input Capture Start Bit</p> <p>The TMR_X external reset input (TMRX) is connected to the IHI signal. TMR_X has input capture registers (TICR, TICRR, and TICRF). TICRR and TICRF can measure the width of a pulse by means of a single capture operation under the control of the ICST bit. When a rising edge followed by a falling edge is detected on TMRX after the ICST bit is set to 1, the contents of TCNT at those points are captured into TICRR and TICRF, respectively, and the ICST bit is cleared to 0.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>• When a rising edge followed by a falling edge is detected on TMRX</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When 1 is written in ICST after reading ICST = 0</li> </ul>

Legend:

X: Don't care

Bit	Bit Name	Initial Value	R/W	Description
3	HFINV	0	R/W	Input Synchronization Signal Inversion
2	VFINV	0	R/W	These bits select inversion of the input phase of the spare horizontal synchronization signal (HFBACKI), the spare vertical synchronization signal (VFBACKI), the horizontal synchronization signal (HSYNCI), composite synchronization signal (CSYNCI), and the vertical synchronization signal (VSYNCI).
1	HIINV	0	R/W	
0	VIINV	0	R/W	
				<ul style="list-style-type: none"> <li>• HFINV           <ul style="list-style-type: none"> <li>0: The HFBACKI pin state is used directly as the HFBACKI input</li> <li>1: The HFBACKI pin state is inverted before use as the HFBACKI input</li> </ul> </li> <li>• VFINV           <ul style="list-style-type: none"> <li>0: The VFBACKI pin state is used directly as the VFBACKI input</li> <li>1: The VFBACKI pin state is inverted before use as the VFBACKI input</li> </ul> </li> <li>• HIINV           <ul style="list-style-type: none"> <li>0: The HSYNCI and CSYNCI pin states are used directly as the HSYNCI and CSYNCI inputs</li> <li>1: The HSYNCI and CSYNCI pin states are inverted before use as the HSYNCI and CSYNCI inputs</li> </ul> </li> <li>• VIINV           <ul style="list-style-type: none"> <li>0: The VSYNCI pin state is used directly as the VSYNCI input</li> <li>1: The VSYNCI pin state is inverted before use as the VSYNCI input</li> </ul> </li> </ul>

**Table 14.2 Synchronization Signal Connection Enable**

<b>Bit 5</b>		<b>Description</b>					
<b>SCONE</b>	<b>Mode</b>	<b>FTIA</b>	<b>FTIB</b>	<b>FTIC</b>	<b>FTID</b>	<b>TMC11</b>	<b>TMRI1</b>
0	Normal connection (Initial value)	FTIA input	FTIB input	FTIC input	FTID input	TMI1 input	TMI1 input
1	Synchronization signal connection mode	IVI signal	TMO1 signal	VFBACKI input	IHI signal	IHI signal	IVI inverse signal

<b>Bit 5</b>		<b>Description</b>			
<b>SCONE</b>	<b>Mode</b>	<b>TMCIX</b>	<b>TMRIX</b>	<b>TMCY</b>	<b>TMRIY</b>
0	Normal connection (Initial value)	TMIX input	TMIX input	TMIY input	TMIY input
1	Synchronization signal connection mode	IHI signal	IHI signal	IVG signal	IVG signal

### 14.3.2 Timer Connection Register O (TCONRO)

TCONRO controls output signal output, phase inversion, etc.

Bit	Bit Name	Initial Value	R/W	Description
7	HOE	0	R/W	Output Enable
6	VOE	0	R/W	These bits control enabling/disabling of output of horizontal synchronization signal (HSYNCO), vertical synchronization signal (VSYNCO), clamp waveform (CLAMPO), and blanking waveform (CBLANK) output. When output is disabled, the state of the relevant pin is determined by port DR and DDR, FRT, TMR, and PWM settings.  Output enabling/disabling control does not affect the port, FRT, or TMR input functions, but some FRT and TMR input signal sources are determined by the SCONE bit in TCONRI.  HOE: 0: The P43/TMO1/HSYNCO pin functions as the P43/TMO1 pin 1: The P43/TMO1/HSYNCO pin functions as the HSYNCO pin  VOE: 0: The P61/FTOA/VSYNCO pin functions as the P61/FTOA pin 1: The P61/FTOA/VSYNCO pin functions as the VSYNCO pin  CLOE: 0: The P64/FTIC/CLAMPO pin functions as the P64/FTIC pin 1: The P64/FTIC/CLAMPO pin functions as the CLAMPO pin  CBOE: 0: The P66/FTOB/CBLANK pin functions as the P66/FTOB pin 1: The P66/FTOB/CBLANK pin functions as the CBLANK pin
5	CLOE	0	R/W	
4	CBOE	0	R/W	

Bit	Bit Name	Initial Value	R/W	Description
3	HOINV	0	R/W	Output Synchronization Signal Inversion
2	VOINV	0	R/W	These bits select inversion of the output phase of the horizontal synchronization signal (HSYNCO), the vertical synchronization signal (VSYNCO), the clamp waveform (CLAMPO), and the blanking waveform (CBLANK).
1	CLOINV	0	R/W	
0	CBOINV	0	R/W	
				HOINV:
				0: The IHO signal is used directly as the HSYNCO output
				1: The IHO signal is inverted before use as the HSYNCO output
				VOINV:
				0: The IVO signal is used directly as the VSYNCO output
				1: The IVO signal is inverted before use as the VSYNCO output
				CLOINV:
				0: The CLO signal (CL1, CL2, CL3, or CL4 signal) is used directly as the CLAMPO output
				1: The CLO signal (CL1, CL2, CL3, or CL4 signal) is inverted before use as the CLAMPO output
				CBOINV:
				0: The CBLANK signal is used directly as the CBLANK output
				1: The CBLANK signal is inverted before use as the CBLANK output

### 14.3.3 Timer Connection Register S (TCONRS)

TCONRS selects whether to access TMR\_X or TMR\_Y registers, and the synchronization signal output signal source and generation method.

Bit	Bit Name	Initial Value	R/W	Description
7	TMR_X/Y	0	R/W	<p>TMR_X/TMR_Y Access Select</p> <p>For details, see table 14.3.</p> <p>0: The TMR_X registers are accessed at addresses H'(FF)FFF0 to H'(FF)FFF5</p> <p>1: The TMR_Y registers are accessed at addresses H'(FF)FFF0 to H'(FF)FFF5</p>
6	ISGENE	0	R/W	<p>Internal Synchronization Signal</p> <p>Selects internal synchronization signals (IHG, IVG, and CL4 signals) as the signal sources for the IHO, IVO, and CLO signals together with the HOMOD1, HOMOD0, VOMOD1, VOMOD0, CLMOD1, and CLMOD0 bits.</p>
5	HOMOD1	0	R/W	Horizontal Synchronization Output Mode Select 1, 0
4	HOMOD0	0	R/W	<p>These bits select the signal source and generation method for the IHO signal.</p> <ul style="list-style-type: none"> <li>• ISGENE = 0 <ul style="list-style-type: none"> <li>00: The IHI signal (without 2fH modification) is selected</li> <li>01: The IHI signal (with 2fH modification) is selected</li> <li>1X: The CL1 signal is selected</li> </ul> </li> <li>• ISGENE = 1 <ul style="list-style-type: none"> <li>XX: The IHG signal is selected</li> </ul> </li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
3	VOMOD1	0	R/W	Vertical Synchronization Output Mode Select 1, 0
2	VOMOD0	0	R/W	These bits select the signal source and generation method for the IVO signal. <ul style="list-style-type: none"> <li>• ISGENE = 0 <ul style="list-style-type: none"> <li>00: The IVI signal (without fall modification or IHI synchronization) is selected</li> <li>01: The IVI signal (without fall modification, with IHI synchronization) is selected</li> <li>10: The IVI signal (with fall modification, without IHI synchronization) is selected</li> <li>11: The IVI signal (with fall modification and IHI synchronization) is selected</li> </ul> </li> <li>• ISGENE = 1 <ul style="list-style-type: none"> <li>XX: The IVG signal is selected</li> </ul> </li> </ul>
1	CLMOD1	0	R/W	Clamp Waveform Mode Select 1, 0
0	CLMOD0	0	R/W	These bits select the signal source for the CLO signal (clamp waveform). <ul style="list-style-type: none"> <li>• ISGENE = 0 <ul style="list-style-type: none"> <li>00: The CL1 signal is selected</li> <li>01: The CL2 signal is selected</li> <li>1X: The CL3 signal is selected</li> </ul> </li> <li>• ISGENE = 1 <ul style="list-style-type: none"> <li>XX: The CL4 signal is selected</li> </ul> </li> </ul>

Legend:

X: Don't care

**Table 14.3 Registers Accessible by TMR\_X/TMR\_Y**

TMRX/Y	H'FFF0	H'FFF1	H'FFF2	H'FFF3	H'FFF4	H'FFF5	H'FFF6	H'FFF7
0	TMR_X	TMR_X	TMR_X	TMR_X	TMR_X	TMR_X	TMR_X	TMR_X
	TCR_X	TCSR_X	TICRR	TICRF	TCNT	TCORC	TCORA_X	TCORB_X
1	TMR_Y	TMR_Y	TMR_Y	TMR_Y	TMR_Y	TMR_Y		
	TCR_Y	TCSR_Y	TCORA_Y	TCORB_Y	TCNT_Y	TISR		

### 14.3.4 Edge Sense Register (SEDGR)

SEDGR detects a rising edge on the timer connection input pins and the occurrence of 2fH modification, and determines the phase of the IVI and IHI signals.

Bit	Bit Name	Initial Value	R/W	Description
7	VEDG	0	R/(W) <sup>*1</sup>	<p>VSYNCI Edge</p> <p>Detects a rising edge on the VSYNCI pin.</p> <p>[Clearing condition]</p> <p>When 0 is written in VEDG after reading VEDG = 1</p> <p>[Setting condition]</p> <p>When a rising edge is detected on the VSYNCI pin</p>
6	HEDG	0	R/(W) <sup>*1</sup>	<p>HSYNCI Edge</p> <p>Detects a rising edge on the HSYNCI pin.</p> <p>[Clearing condition]</p> <p>When 0 is written in HEDG after reading HEDG = 1</p> <p>[Setting condition]</p> <p>When a rising edge is detected on the HSYNCI pin</p>
5	CEDG	0	R/(W) <sup>*1</sup>	<p>CSYNCI Edge</p> <p>Detects a rising edge on the CSYNCI pin.</p> <p>[Clearing condition]</p> <p>When 0 is written in CEDG after reading CEDG = 1</p> <p>[Setting condition]</p> <p>When a rising edge is detected on the CSYNCI pin</p>
4	HFEDG	0	R/(W) <sup>*1</sup>	<p>HFBACKI Edge</p> <p>Detects a rising edge on the HFBACKI pin.</p> <p>[Clearing condition]</p> <p>When 0 is written in HFEDG after reading HFEDG = 1</p> <p>[Setting condition]</p> <p>When a rising edge is detected on the HFBACKI pin</p>



Bit	Bit Name	Initial Value	R/W	Description
3	VFEDG	0	R/(W) *1	<p>VFBACKI Edge</p> <p>Detects a rising edge on the VFBACKI pin.</p> <p>[Clearing condition]</p> <p>When 0 is written in VFEDG after reading VFEDG = 1</p> <p>[Setting condition]</p> <p>When a rising edge is detected on the VFBACKI pin</p>
2	PREDG	0	R/(W) *1	<p>Pre-Equalization Flag</p> <p>Detects the occurrence of an IHI signal 2fH modification condition. The generation of a falling/rising edge in the IHI signal during a mask interval is expressed as the occurrence of a 2fH modification condition. For details, see section 14.4.4, IHI Signal and 2fH Modification.</p> <p>[Clearing condition]</p> <p>When 0 is written in PREQF after reading PREQF = 1</p> <p>[Setting condition]</p> <p>When an IHI signal 2fH modification condition is detected</p>
1	IHI	Undefined*2	R	<p>IHI Signal Level</p> <p>Indicates the current level of the IHI signal. Signal source and phase inversion selection for the IHI signal depends on the contents of TCONRI. Read this bit to determine whether the input signal is positive or negative, then maintain the IHI signal at positive phase by modifying TCONRI.</p> <p>0: The IHI signal is low</p> <p>1: The IHI signal is high</p>
0	IVI	Undefined*2	R	<p>IVI Signal Level</p> <p>Indicates the current level of the IVI signal. Signal source and phase inversion selection for the IVI signal depends on the contents of TCONRI. Read this bit to determine whether the input signal is positive or negative, then maintain the IVI signal at positive phase by modifying TCONRI.</p> <p>0: The IVI signal is low</p> <p>1: The IVI signal is high</p>

Notes: 1. Only 0 can be written, to clear the flag.  
2. The initial value is undefined since it depends on the pin state.

## 14.4 Operation

### 14.4.1 PWM Decoding (PDC Signal Generation)

The timer connection facility and TMR\_X can be used to decode a PWM signal in which 0 and 1 are represented by the pulse width. To do this, a signal in which a rising edge is generated at regular intervals must be selected as the IHI signal.

The timer counter (TCNT) in TMR\_X is set to count the internal clock pulses and to be cleared on the rising edge of the external reset signal (IHI signal). The value to be used as the threshold for deciding the pulse width is written in TCORB. The PWM decoder contains a delay latch which uses the IHI signal as data and compare-match signal B (CMB) as a clock, and the state of the IHI signal (the result of the pulse width decision) at the first compare-match signal B timing after TCNT is reset by the rise of the IHI signal is output as the PDC signal.

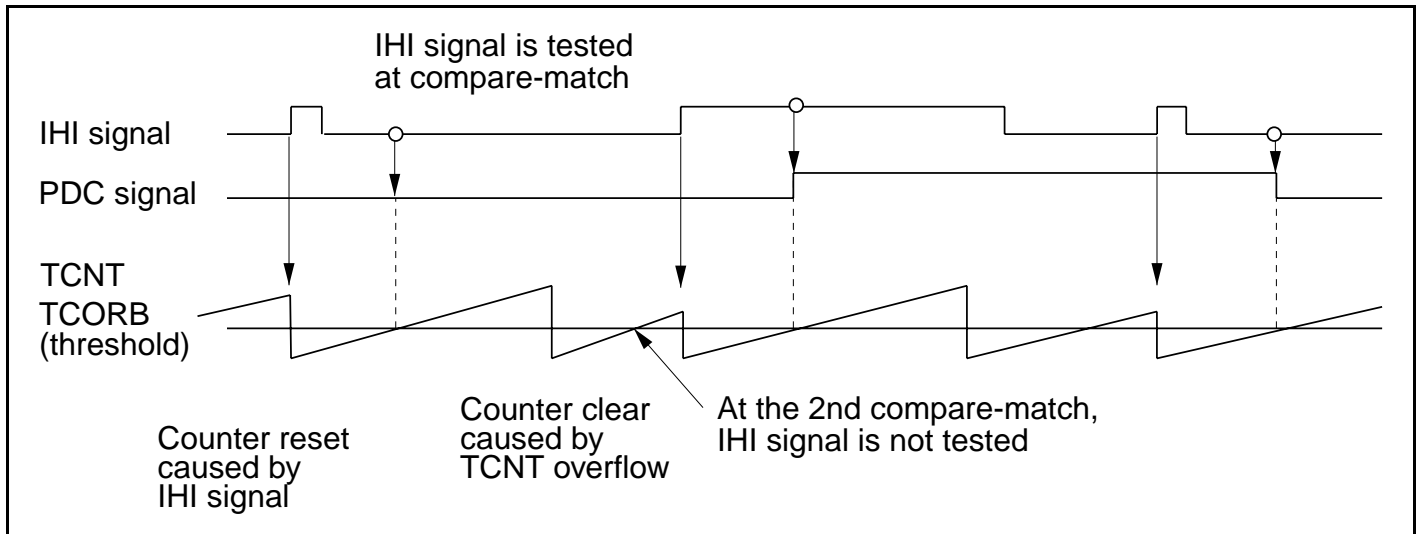
The pulse width setting using TICRR and TICRF of TMR\_X can be used to determine the pulse width decision threshold. Examples of TCR and TCORB settings of TMR\_X are shown in tables 14.4 and 14.5, and the PWM decoding timing chart is shown in figure 14.2.

**Table 14.4 Examples of TCR Settings**

Bit	Abbreviation	Contents	Description
7	CMIEB	0	Interrupts due to compare-match and overflow are disabled
6	CMIEA	0	
5	OVIE	0	
4, 3	CCLR1, CCLR0	11	TCNT is cleared by the rising edge of the external reset signal (IHI signal)
2 to 0	CKS2 to CKS0	001	Incremented on internal clock ( $\phi$ )

**Table 14.5 Examples of TCORB (Pulse Width Threshold) Settings**

	$\phi$ : 10 MHz	$\phi$ : 12 MHz	$\phi$ : 16 MHz	$\phi$ : 20 MHz
H'07	0.8 $\mu$ s	0.67 $\mu$ s	0.5 $\mu$ s	0.4 $\mu$ s
H'0F	1.6 $\mu$ s	1.33 $\mu$ s	1.0 $\mu$ s	0.8 $\mu$ s
H'1F	3.2 $\mu$ s	2.67 $\mu$ s	2.0 $\mu$ s	1.6 $\mu$ s
H'3F	6.4 $\mu$ s	5.33 $\mu$ s	4.0 $\mu$ s	3.2 $\mu$ s
H'7F	12.8 $\mu$ s	10.67 $\mu$ s	8.0 $\mu$ s	6.4 $\mu$ s



**Figure 14.2 Timing Chart for PWM Decoding**

#### 14.4.2 Clamp Waveform Generation (CL1/CL2/CL3 Signal Generation)

The timer connection facility and TMR\_X can be used to generate signals with different duty cycles and rising/falling edges (clamp waveforms) in synchronization with the input signal (IHI signal). Three clamp waveforms can be generated: the CL1, CL2, and CL3 signals. In addition, the CL4 signal can be generated using TMR\_Y.

The CL1 signal rises simultaneously with the rise of the IHI signal, and when the CL1 signal is high, the CL2 signal rises simultaneously with the fall of the IHI signal. The fall of both the CL1 and CL2 signals can be specified by TCORA.

The rise of the CL3 signal can be specified as simultaneous with the sampling of the fall of the IHI signal using the system clock, and the fall of the CL3 signal can be specified by TCORC. The CL3 signal can also fall when the IHI signal rises.

TCNT in TMR\_X is set to count internal clock pulses and to be cleared on the rising edge of the external reset signal (IHI signal).

The value to be used as the CL1 signal pulse width is written in TCORA. Write a value of H'02 or more in TCORA when internal clock  $\phi$  is selected as the TMR\_X counter clock, and a value of H'01 or more when  $\phi/2$  is selected. When internal clock  $\phi$  is selected, the CL1 signal pulse width is (TCORA set value + 3  $\pm$  0.5). When the CL2 signal is used, the setting must be made so that this pulse width is greater than the IHI signal pulse width.

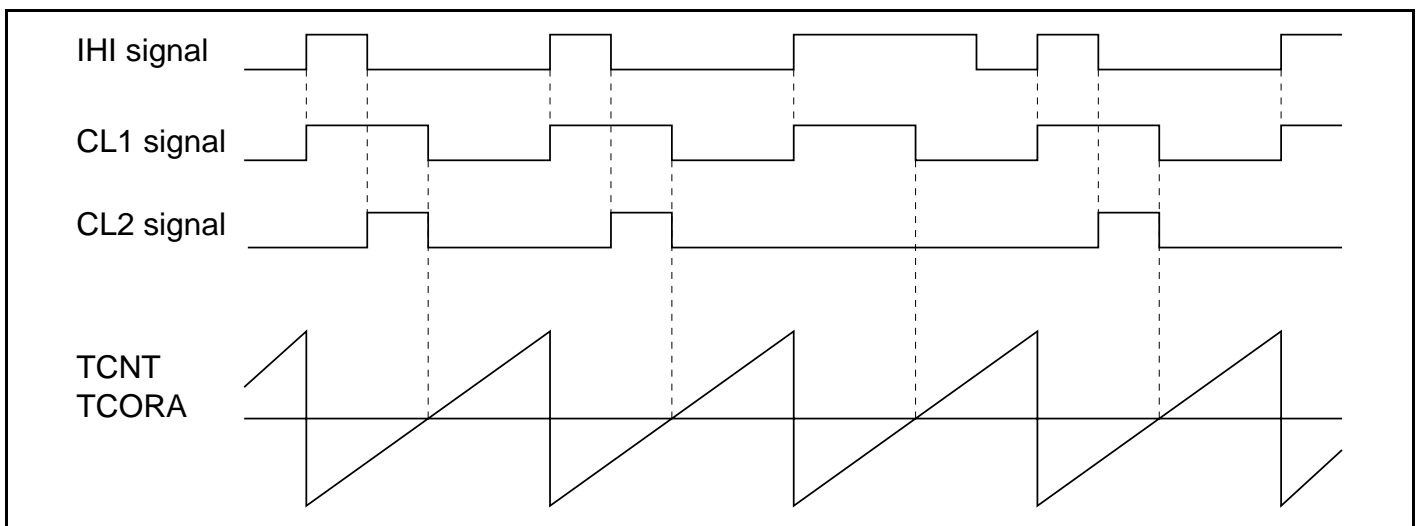
The value to be used as the CL3 signal pulse width is written in TCORC. TCCR in TMR\_X captures the value of TCNT at the inverse of the external reset signal edge (in this case, the falling edge of the IHI signal). The timing of the fall of the CL3 signal is determined by the sum of the

contents of TICR and TCORC. Caution is required if the rising edge of the IHI signal precedes the fall timing set by the contents of TCORC, since the IHI signal will cause the CL3 signal to fall.

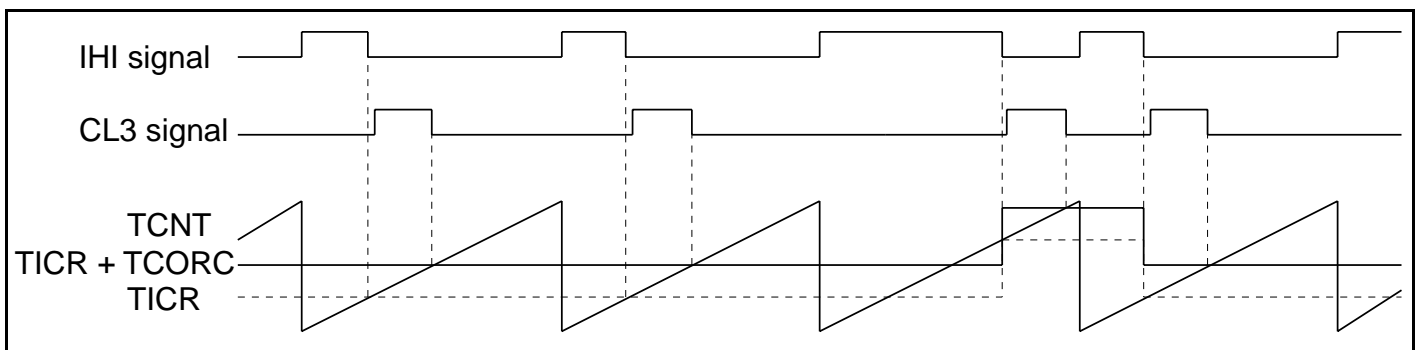
Examples of TCR settings of TMR\_X are the same as those in table 14.4. The clamp waveform timing charts are shown in figures 14.3 and 14.4.

Since the rise of the CL1 and CL2 signals is synchronized with the edge of the IHI signal, and their fall is synchronized with the system clock, the pulse width variation is equivalent to the resolution of the system clock.

Both the rise and the fall of the CL3 signal are synchronized with the system clock and the pulse width is fixed, but there is a variation in the phase relationship with the IHI signal equivalent to the resolution of the system clock.



**Figure 14.3 Timing Chart for Clamp Waveform Generation (CL1 and CL2 Signals)**



**Figure 14.4 Timing Chart for Clamp Waveform Generation (CL3 Signal)**

### 14.4.3 8-Bit Timer Divided Waveform Period Measurement

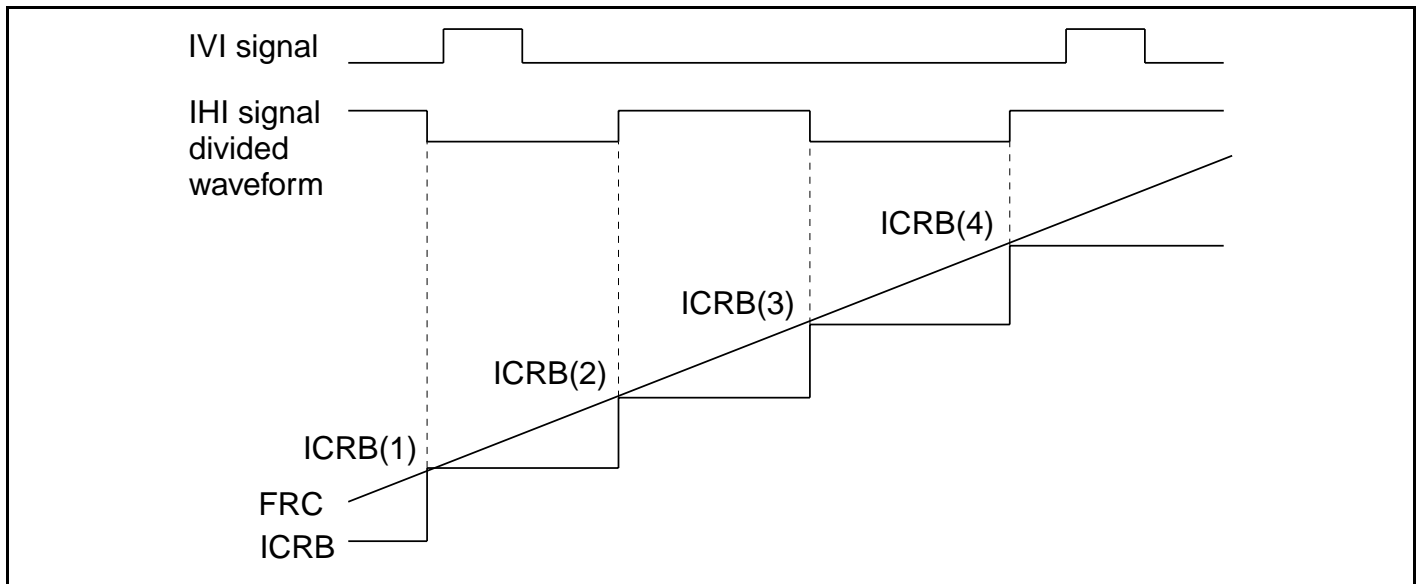
The timer connection facility, TMR\_1, and the free-running timer (FRT) can be used to measure the period of an IHI signal divided waveform. Since TMR\_1 can be cleared by a rising edge of the inverted IVI signal, the rise and fall of the IHI signal divided waveform can be synchronized with the IVI signal. This enables period measurement to be carried out efficiently.

To measure the period of an IHI signal divided waveform, TCNT in TMR\_1 is set to count the external clock (IHI signal) pulses and to be cleared on the rising edge of the external reset signal (inverse of the IVI signal). The value to be used as the division factor is written in TCORA, and the TMO output method is specified by the OS bits in TCSR.

Examples of TCR and TCSR settings in TMR\_1, and TCR and TCSR settings in the FRT are shown in table 14.6, and the timing chart for measurement of the IVI signal and IHI signal divided waveform periods is shown in figure 14.5. The period of the IHI signal divided waveform is given by  $(ICRD(3) - ICRD(2)) \times \text{resolution}$ .

**Table 14.6 Examples of TCR and TCSR Settings**

Register	Bit	Abbreviation	Contents	Description
TCR in TMR_1	7	CMIEB	0	Interrupts due to compare-match and overflow are disabled
	6	CMIEA	0	
	5	OVIE	0	
	4, 3	CCLR1, CCLR0	11	TCNT is cleared by the rising edge of the external reset signal (inverse of the IVI signal)
	2 to 0	CKS2 to CKS0	101	TCNT is incremented on the rising edge of the external clock (IHI signal)
TCSR in TMR_1	3 to 0	OS3 to OS0	0011	Not changed by compare-match B; output inverted by compare-match A (toggle output): Division by 512
			1001	When TCORB < TCORA, 1 output on compare-match B, and 0 output on compare-match A: Division by 256
TCR in FRT	6	IEDGB	0/1	0: FRC value is transferred to ICRB on falling edge of input capture input B (IHI divided signal waveform)
				1: FRC value is transferred to ICRB on rising edge of input capture input B (IHI divided signal waveform)
	1, 0	CKS1, CKS0	01	FRC is incremented on internal clock: $\phi/8$
TCSR in FRT	0	CCLRA	0	FRC clearing is disabled



**Figure 14.5 Timing Chart for Measurement of IVI Signal and IHI Signal Divided Waveform Periods**

#### 14.4.4 IHI Signal and 2fH Modification

By using the timer connection facility and FRT, even if there is a part of the IHI signal with twice the frequency, this can be eliminated. In order for this function to operate properly, the duty cycle of the IHI signal must be approximately 30% or less, or approximately 70% or above.

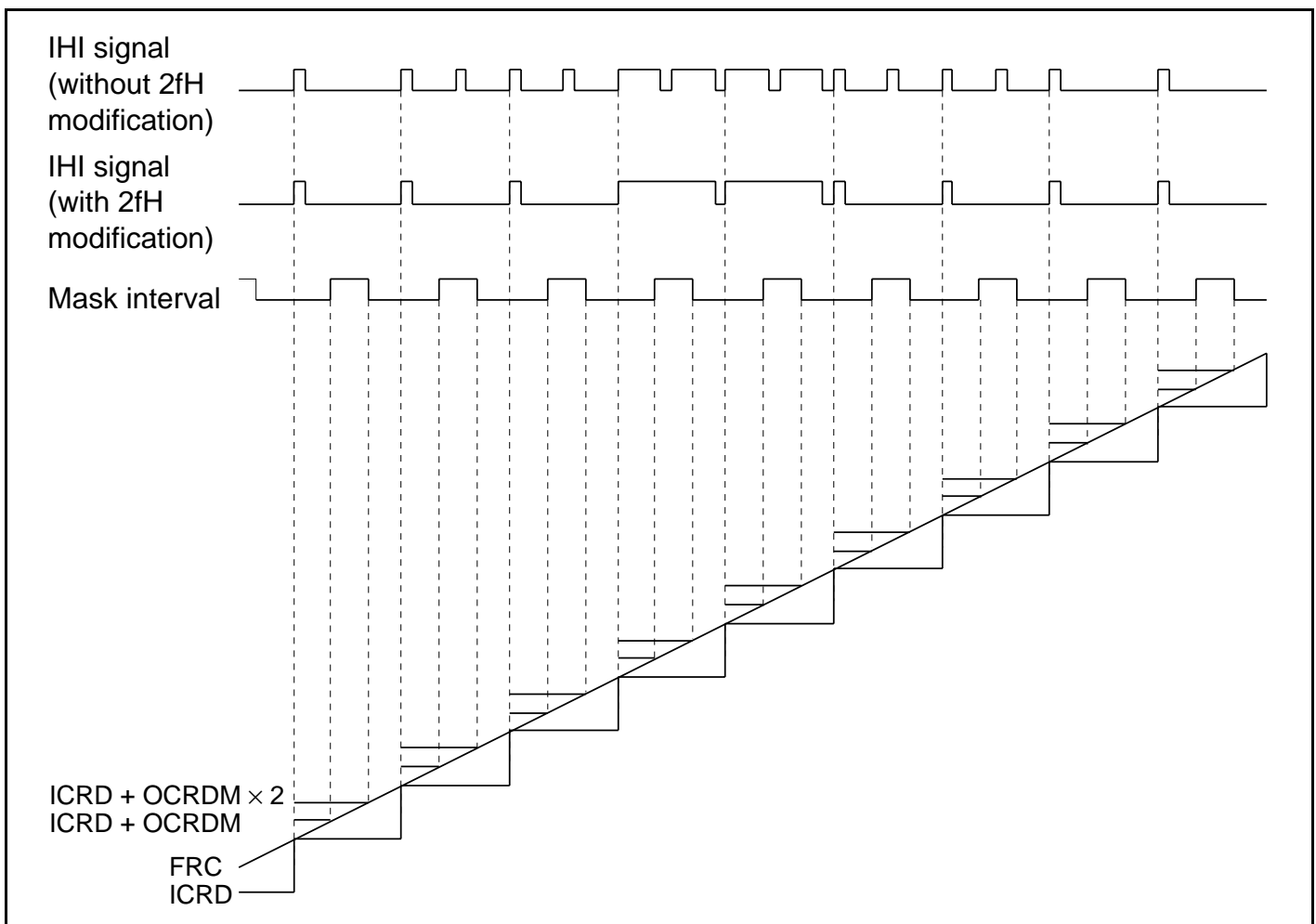
The 8-bit OCRDM contents or twice the OCRDM contents can be added automatically to the data captured in ICRD in the FRT, and compare-matches generated at these points. The interval between the two compare-matches is called a mask interval. A value equivalent to approximately 1/3 the IHI signal period is written in OCRDM. ICRD is set so that capture is performed on the rise of the IHI signal.

Since the IHI signal supplied to the IHO signal selection circuit is normally set on the rise of the IHI signal and reset on the fall, its waveform is the same as that of the original IHI signal. When 2fH modification is selected, IHI signal edge detection is disabled during mask intervals. Capture is also disabled during these intervals.

Examples of TCR, TCSR, TOCR, and OCRDM settings in the FRT are shown in table 14.7, and the 2fH modification timing chart is shown in figure 14.6.

**Table 14.7 Examples of TCR, TCSR, TOCR, and OCRDM Settings**

Register	Bit	Abbreviation	Contents	Description
TCR in FRT	4	IEDGD	1	FRC value is transferred to ICRD on the rising edge of input capture input D (IHI signal)
	1, 0	CKS1, CKS0	01	FRC is incremented on internal clock: $\phi/8$
TCSR in FRT	0	CCLRA	0	FRC clearing is disabled
TOCR in FRT	7	ICRDMS	1	ICRD is set to the operating mode in which OCRDM is used
OCRDM in FRT	7 to 0	OCRDM7 to OCRDM0	H'01 to H'FF	Specifies the period during which ICRD operation is masked

**Figure 14.6 2fH Modification Timing Chart**



### 14.4.5 IVI Signal Fall Modification and IHI Synchronization

By using the timer connection facility and TMR\_1, the fall of the IVI signal can be shifted backward by the specified number of IHI signal waveforms. Also, the fall of the IVI signal can be synchronized with the rise of the IHI signal.

To perform 8-bit timer divided waveform period measurement, TCNT in TMR\_1 is set to count external clock (IHI signal) pulses, and to be cleared on the rising edge of the external reset signal (inverse of the IVI signal). The number of IHI signal pulses until the fall of the IVI signal is written in TCORB.

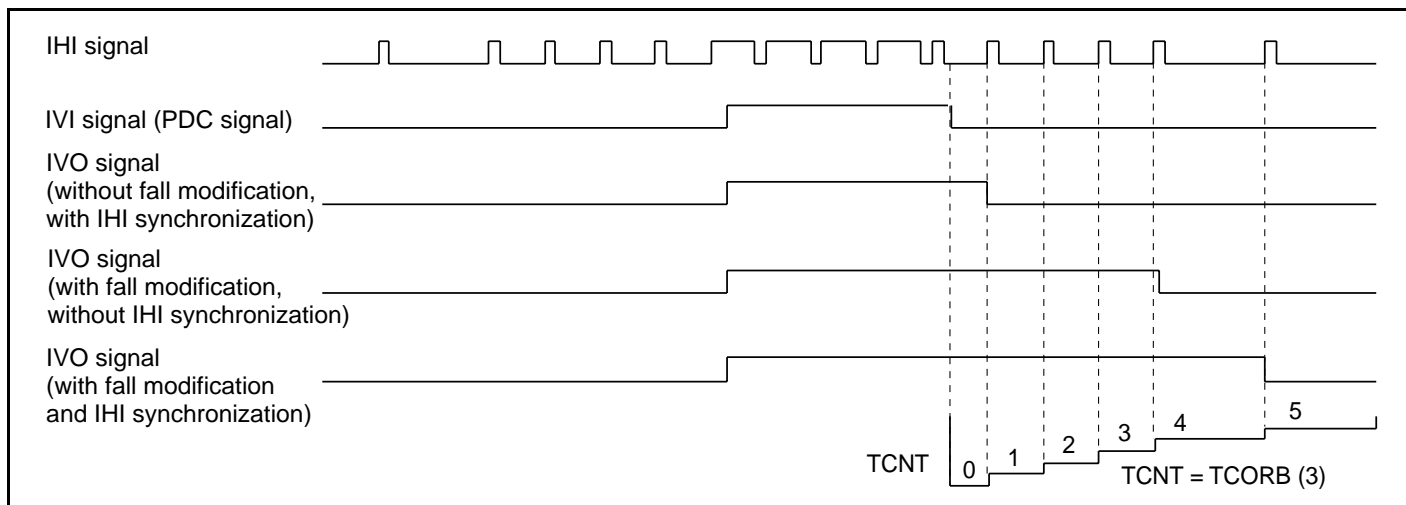
Since the IVI signal supplied to the IVO signal selection circuit is normally set on the rise of the IVI signal and reset on the fall, its waveform is the same as that of the original IVI signal. When fall modification is selected, a reset is performed on a TMR\_1 TCORB compare-match in TMR\_1.

The fall of the waveform generated in this way can be synchronized with the rise of the IHI signal, regardless of whether or not fall modification is selected.

Examples of TCR, TCSR, and TCORB settings in TMR\_1 are shown in table 14.8, and the fall modification/IHI synchronization timing chart is shown in figure 14.7.

**Table 14.8 Examples of TCR, TCSR, and TCORB Settings**

Register	Bit	Abbreviation	Contents	Description	
TCR in TMR_1	7	CMIEB	0	Interrupts due to compare-match and overflow are disabled	
	6	CMIEA	0		
	5	OVIE	0		
	4, 3	CCLR1, CCLR0	11		TCNT is cleared by the rising edge of the external reset signal (inverse of the IVI signal)
	2 to 0	CKS2 to CKS0	101		TCNT is incremented on the rising edge of the external clock (IHI signal)
TCSR in TMR_1	3 to 0	OS3 to OS0	0011	Not changed by compare-match B; output inverted by compare-match A (toggle output)	
			1001	When TCORB < TCORA, 1 output on compare-match B, 0 output on compare-match A	
TCORB in TMR_1			H'03 (example)	Compare-match on the 4th (example) rise of the IHI signal after the rise of the inverse of the IVI signal	

**Figure 14.7 Fall Modification and IHI Synchronization Timing Chart**

#### 14.4.6 Internal Synchronization Signal Generation (IHG/IVG/CL4 Signal Generation)

By using the timer connection facility, FRT, and TMR\_Y, it is possible to automatically generate internal signals (IHG and IVG signals) corresponding to the IHI and IVI signals. As the IHG signal is synchronized with the rise of the IVG signal, the IHG signal period must be made a divisor of the IVG signal period in order to keep it constant. In addition, the CL4 signal can be generated in synchronization with the IHG signal.

The contents of OCRA in the FRT are updated by the automatic addition of the contents of OCRAR or OCRAF, alternately, each time a compare-match occurs. A value corresponding to the 0 interval of the IVG signal is written in OCRAR, and a value corresponding to the 1 interval of the IVG signal is written in OCRAF. The IVG signal is set by a compare-match after an OCRAR addition, and reset by a compare-match after an OCRAF addition.

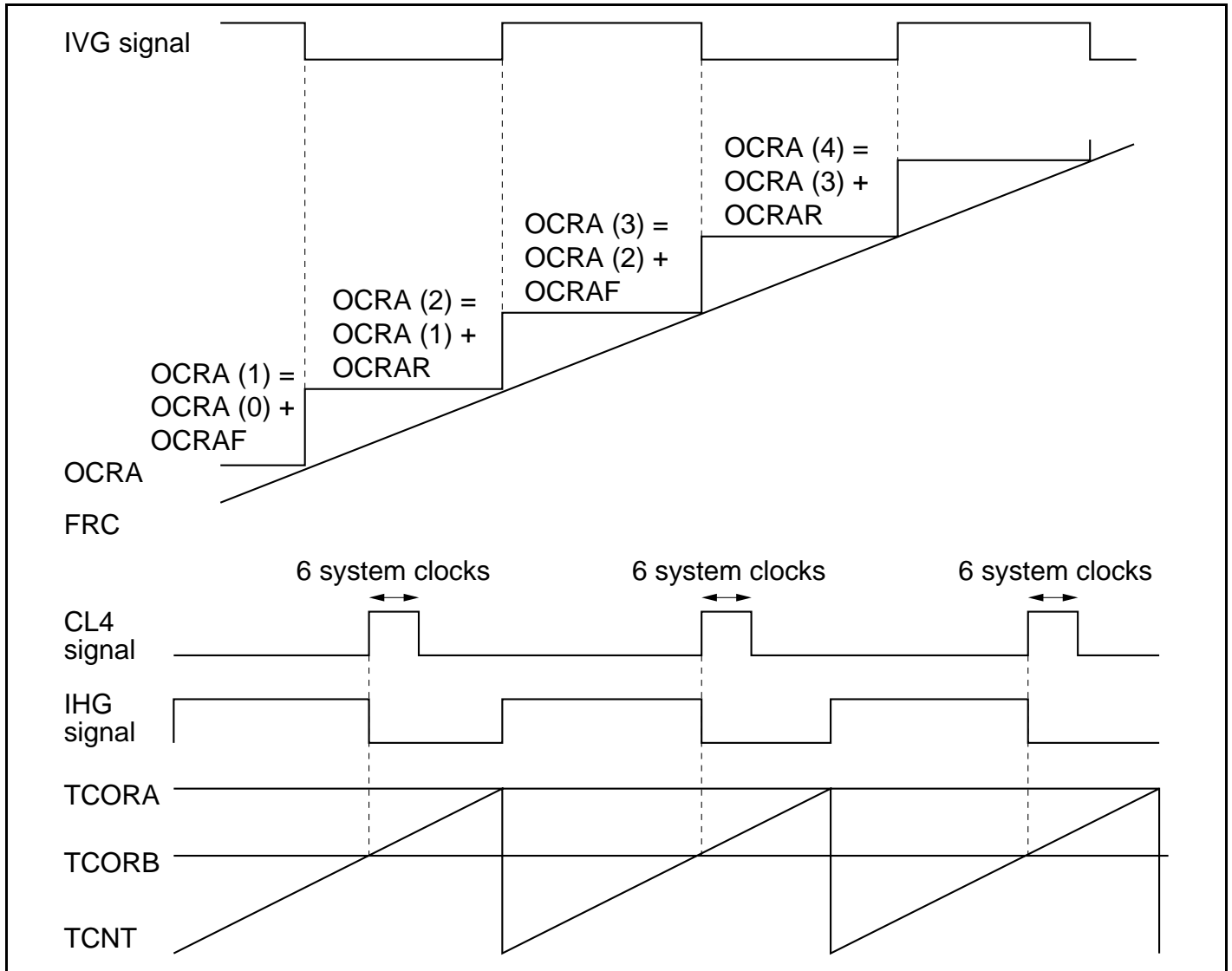
The IHG signal is the TMR\_Y timer output. TMR\_Y is set to count internal clock pulses, and to be cleared on a TCORA compare-match, to fix the period and set the timer output. TCORB is set so as to reset the timer output. The IVG signal is connected as the TMR\_Y reset input (TMRI), and the rise of the IVG signal can be treated in the same way as a TCORA compare-match.

The CL4 signal is a waveform that rises within one system clock period after the fall of the IHG signal, and has an interval of 1 for 6 system clock periods.

Examples of TCR, TCSR, TCORA, and TCORB settings in TMR\_Y, and TCR, OCRAR, OCRAF, and TOCR settings in the FRT are shown in table 14.9, and the IHG signal/IVG signal timing chart is shown in figure 14.8.

**Table 14.9 Examples of TCR, TCSR, TCORA, TCORB, OCRAR, OCRAF, and TOCR Settings**

Register	Bit	Abbreviation	Contents	Description	
TCR in TMR_Y	7	CMIEB	0	Interrupts due to compare-match and overflow are disabled	
	6	CMIEA	0		
	5	OVIE	0		
	4, 3	CCLR1, CCLR0	01	TCNT is cleared by compare-match A	
	2 to 0	CKS2 to CKS0	001	TCNT is incremented on internal clock: $\phi/4$	
TCSR in TMR_Y	3 to 0	OS3 to OS0	0110	0 output on compare-match B 1 output on compare-match A	
TCORA in TMR_Y			H'3F (example)	IHG signal period = $\phi \times 256$	
TCORB in TMR_Y			H'03 (example)	IHG signal 1 interval = $\phi \times 16$	
TCR in FRT	1, 0	CKS1, CKS0	01	FRC is incremented on internal clock: $\phi/8$	
OCRAR in FRT			H'7FEF (example)	IVG signal 0 interval = $\phi \times 262016$	IVG signal period = $\phi \times 262144$ (1024 times IHG signal)
OCRAF in FRT			H'000F (example)	IVG signal 1 interval = $\phi \times 128$	
TOCR in FRT	6	OCRAMS	1	OCRA is set to the operating mode in which OCRAR and OCRAF are used	



**Figure 14.8 IVG Signal/IHG Signal/CL4 Signal Timing Chart**

### 14.4.7 HSYNCO Output

With the HSYNCO output, the meaning of the signal source to be selected and use or non-use of modification varies according to the IHI signal source and the waveform required by external circuitry. The HSYNCO output modes are shown in table 14.10.

**Table 14.10 HSYNCO Output Modes**

Mode	IHI Signal	IHO Signal	Meaning of IHO Signal
No signal	HFBACKI input	IHI signal (without 2fH modification)	HFBACKI input is output directly
		IHI signal (with 2fH modification)	Meaningless unless there is a double-frequency part in the HFBACKI input
		CL1 signal	HFBACKI input 1 interval is changed before output
		IHG signal	Internal synchronization signal is output
S-on-G mode	CSYNCI input	IHI signal (without 2fH modification)	CSYNCI input (composite synchronization signal) is output directly
		IHI signal (with 2fH modification)	Double-frequency part of CSYNCI input (composite synchronization signal) is eliminated before output
		CL1 signal	CSYNCI input (composite synchronization signal) horizontal synchronization signal part is separated before output
		IHG signal	Internal synchronization signal is output
Composite mode	HSYNCI input	IHI signal (without 2fH modification)	HSYNCI input (composite synchronization signal) is output directly
		IHI signal (with 2fH modification)	Double-frequency part of HSYNCI input (composite synchronization signal) is eliminated before output
		CL1 signal	HSYNCI input (composite synchronization signal) horizontal synchronization signal part is separated before output
		IHG signal	Internal synchronization signal is output
Separate mode	HSYNCI input	IHI signal (without 2fH modification)	HSYNCI input (horizontal synchronization signal) is output directly
		IHI signal (with 2fH modification)	Meaningless unless there is a double-frequency part in the HSYNCI input (horizontal synchronization signal)
		CL1 signal	HSYNCI input (horizontal synchronization signal) 1 interval is changed before output
		IHG signal	Internal synchronization signal is output

### 14.4.8 VSYNCO Output

With the VSYNCO output, the meaning of the signal source to be selected and use or non-use of modification varies according to the IVO signal source and the waveform required by external circuitry. The VSYNCO output modes are shown in table 14.11.

**Table 14.11 VSYNCO Output Modes**

Mode	IVI Signal	IVO Signal	Meaning of IVO Signal
No signal	VFBACKI input	IVI signal (without fall modification or IHI synchronization)	VFBACKI input is output directly
		IVI signal (without fall modification, with IHI synchronization)	Meaningless unless VFBACKI input is synchronized with HFBACKI input
		IVI signal (with fall modification, without IHI synchronization)	VFBACKI input fall is modified before output
		IVI signal (with fall modification and IHI synchronization)	VFBACKI input fall is modified and signal is synchronized with HFBACKI input before output
		IVG signal	Internal synchronization signal is output
S-on-G mode or composite mode	PDC signal	IVI signal (without fall modification or IHI synchronization)	CSYNCI/HSYNCI input (composite synchronization signal) vertical synchronization signal part is separated before output
		IVI signal (without fall modification, with IHI synchronization)	CSYNCI/HSYNCI input (composite synchronization signal) vertical synchronization signal part is separated, and signal is synchronized with CSYNCI/HSYNCI input before output
		IVI signal (with fall modification, without IHI synchronization)	CSYNCI/HSYNCI input (composite synchronization signal) vertical synchronization signal part is separated, and fall is modified before output
		IVI signal (with fall modification and IHI synchronization)	CSYNCI/HSYNCI input (composite synchronization signal) vertical synchronization signal part is separated, fall is modified, and signal is synchronized with CSYNCI/HSYNCI input before output
		IVG signal	Internal synchronization signal is output

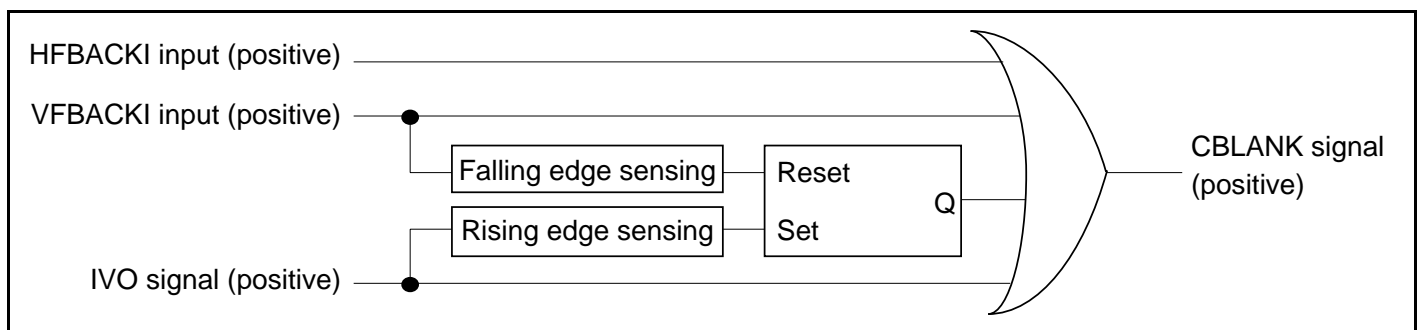
Mode	IVI Signal	I/O Signal	Meaning of I/O Signal
Separate mode	VSYNCl input	IVI signal (without fall modification or IHI synchronization)	VSYNCl input (vertical synchronization signal) is output directly
		IVI signal (without fall modification, with IHI synchronization)	Meaningless unless VSYNCl input (vertical synchronization signal) is synchronized with HSYNCl input (horizontal synchronization signal)
		IVI signal (with fall modification, without IHI synchronization)	VSYNCl input (vertical synchronization signal) fall is modified before output
		IVI signal (with fall modification and IHI synchronization)	VSYNCl input (vertical synchronization signal) fall is modified and signal is synchronized with HSYNCl input (horizontal synchronization signal) before output
		IVG signal	Internal synchronization signal is output

#### 14.4.9 CBLANK Output

Using the signals generated/selected with timer connection, it is possible to generate a waveform based on the composite synchronization signal (blanking waveform).

One kind of blanking waveform is generated by combining HFBACKI and VFBACKI inputs, with the phase polarity made positive by means of bits HFINV and VFINV in TCONRI, with the IVO signal.

The logic of CBLANK output waveform generation is shown in figure 14.9.



**Figure 14.9 CBLANK Output Waveform Generation**



## Section 15 Watchdog Timer (WDT)

This LSI incorporates two watchdog timer channels (WDT\_0 and WDT\_1). The watchdog timer can output an overflow signal ( $\overline{\text{RESO}}$ ) externally if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow. Simultaneously, it can generate an internal reset signal or an internal NMI interrupt signal.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows. A block diagram of the WDT is shown in figure 15.1.

### 15.1 Features

- Selectable from eight (WDT\_0) or 16 (WDT\_1) counter input clocks.
- Switchable between watchdog timer mode and interval timer mode

Watchdog Timer Mode:

- If the counter overflows, an internal reset or an internal NMI interrupt is generated.
- When the LSI is selected to be internally reset at counter overflow, a low level signal is output from the  $\overline{\text{RESO}}$  pin if the counter overflows.

Internal Timer Mode:

- If the counter overflows, an internal timer interrupt (WOVI) is generated.

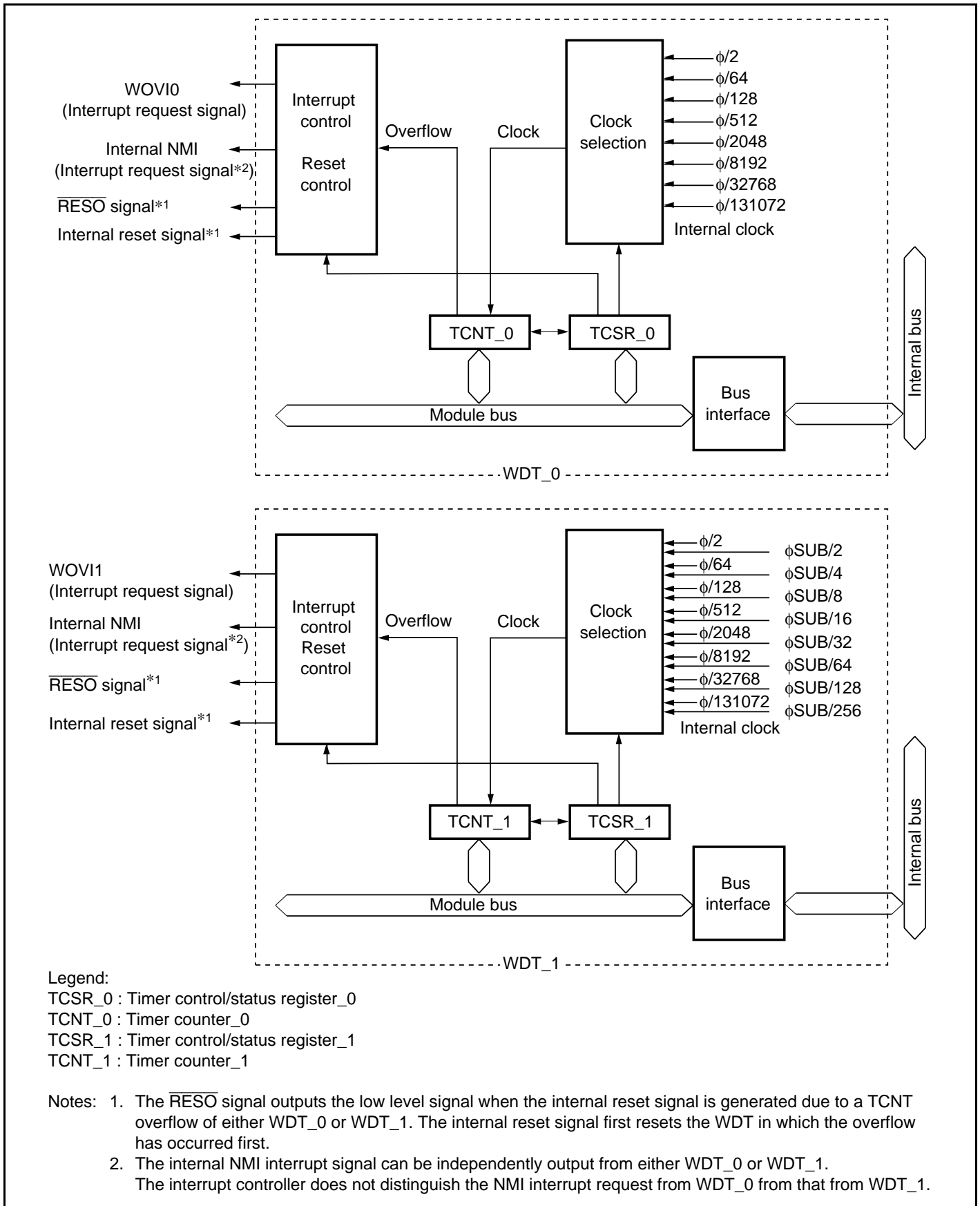


Figure 15.1 Block Diagram of WDT

## 15.2 Input/Output Pins

The WDT has the pins listed in table 15.1.

**Table 15.1 Pin Configuration**

Name	Symbol	I/O	Function
Reset output pin	$\overline{\text{RESO}}$	Output	Outputs the counter overflow signal in watchdog timer mode
External sub-clock input pin	EXCL	Input	Inputs the clock pulses to the WDT_1 prescaler counter

## 15.3 Register Descriptions

The WDT has the following registers. To prevent accidental overwriting, TCSR and TCNT have to be written to in a method different from normal registers. For details, see section 15.6.1, Notes on Register Access. For details on the system control register, see section 3.2.2, System Control Register (SYSCR).

- Timer counter\_0 (TCNT\_0)
- Timer control/status register\_0 (TCSR\_0)
- Timer counter\_1 (TCNT\_1)
- Timer control/status register\_1 (TCSR\_1)

### 15.3.1 Timer Counter (TCNT)

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

TCNT is initialized to H'00 when the TME bit in timer control/status register (TCSR) is cleared to 0.

### 15.3.2 Timer Control/Status Register (TCSR)

TCSR selects the clock source to be input to TCNT, and the timer mode.

- TCSR\_0

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*	<p>Overflow Flag</p> <p>Indicates that TCNT has overflowed (changes from H'FF to H'00).</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When TCNT overflows (changes from H'FF to H'00)</li> </ul> <p>When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When TCSR is read when OVF = 1, then 0 is written to OVF</li> <li>• When 0 is written to TME</li> </ul>
6	WT/ $\overline{\text{IT}}$	0	R/W	<p>Timer Mode Select</p> <p>Selects whether the WDT is used as a watchdog timer or interval timer.</p> <p>0: Interval timer mode 1: Watchdog timer mode</p>
5	TME	0	R/W	<p>Timer Enable</p> <p>When this bit is set to 1, TCNT starts counting.</p> <p>When this bit is cleared, TCNT stops counting and is initialized to H'00.</p>
4	—	0	R/(W)	<p>Reserved</p> <p>The initial value should not be changed.</p>
3	RST/ $\overline{\text{NMI}}$	0	R/W	<p>Reset or NMI</p> <p>Selects to request an internal reset or an NMI interrupt when TCNT has overflowed.</p> <p>0: An NMI interrupt is requested 1: An internal reset is requested</p>

Bit	Bit Name	Initial Value	R/W	Description
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	Selects the clock source to be input to. The overflow frequency for $\phi = 25$ MHz is enclosed in parentheses. 000: $\phi/2$ (frequency: 20.4 $\mu$ s) 001: $\phi/64$ (frequency: 655.3 $\mu$ s) 010: $\phi/128$ (frequency: 1.3 ms) 011: $\phi/512$ (frequency: 5.2 ms) 100: $\phi/2048$ (frequency: 20.9 ms) 101: $\phi/8192$ (frequency: 83.8 ms) 110: $\phi/32768$ (frequency: 335.5 ms) 111: $\phi/131072$ (frequency: 1.34 s)
0	CKS0	0	R/W	

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

## • TCSR\_1

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)* <sup>1</sup>	<p>Overflow Flag</p> <p>Indicates that TCNT has overflowed (changes from H'FF to H'00).</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When TCNT overflows (changes from H'FF to H'00)</li> </ul> <p>When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When TCSR is read when <math>OVF = 1^{*2}</math>, then 0 is written to OVF</li> <li>When 0 is written to TME</li> </ul>
6	WT/ $\overline{IT}$	0	R/W	<p>Timer Mode Select</p> <p>Selects whether the WDT is used as a watchdog timer or interval timer.</p> <p>0: Interval timer mode 1: Watchdog timer mode</p>
5	TME	0	R/W	<p>Timer Enable</p> <p>When this bit is set to 1, TCNT starts counting.</p> <p>When this bit is cleared, TCNT stops counting and is initialized to H'00.</p>
4	PSS	0	R/W	<p>Prescaler Select</p> <p>Selects the clock source to be input to TCNT.</p> <p>0: Counts the divided cycle of <math>\phi</math>-based prescaler (PSM) 1: Counts the divided cycle of <math>\phi</math>SUB-based prescaler (PSS)</p>
3	RST/ $\overline{NMI}$	0	R/W	<p>Reset or NMI</p> <p>Selects to request an internal reset or an NMI interrupt when TCNT has overflowed.</p> <p>0: An NMI interrupt is requested 1: An internal reset is requested</p>

Bit	Bit Name	Initial Value	R/W	Description
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	Selects the clock source to be input to TCNT. The overflow cycle for $\phi = 25$ MHz and $\phi\text{SUB} = 32.768$ kHz is enclosed in parentheses.
0	CKS0	0	R/W	
When PSS = 0:				
000: $\phi/2$ (frequency: 20.4 $\mu\text{s}$ )				
001: $\phi/64$ (frequency: 655.3 $\mu\text{s}$ )				
010: $\phi/128$ (frequency: 1.3 ms)				
011: $\phi/512$ (frequency: 5.2 ms)				
100: $\phi/2048$ (frequency: 20.9 ms)				
101: $\phi/8192$ (frequency: 83.8 ms)				
110: $\phi/32768$ (frequency: 335.5 ms)				
111: $\phi/131072$ (frequency: 1.34 s)				
When PSS = 1:				
000: $\phi\text{SUB}/2$ (cycle: 15.6 ms)				
001: $\phi\text{SUB}/4$ (cycle: 31.3 ms)				
010: $\phi\text{SUB}/8$ (cycle: 62.5 ms)				
011: $\phi\text{SUB}/16$ (cycle: 125 ms)				
100: $\phi\text{SUB}/32$ (cycle: 250 ms)				
101: $\phi\text{SUB}/64$ (cycle: 500 ms)				
110: $\phi\text{SUB}/128$ (cycle: 1 s)				
111: $\phi/256$ (cycle: 2 s)				

- Notes:
1. Only 0 can be written, to clear the flag.
  2. When OVF is polled with the interval timer interrupt disabled, OVF = 1 must be read at least twice.

## 15.4 Operation

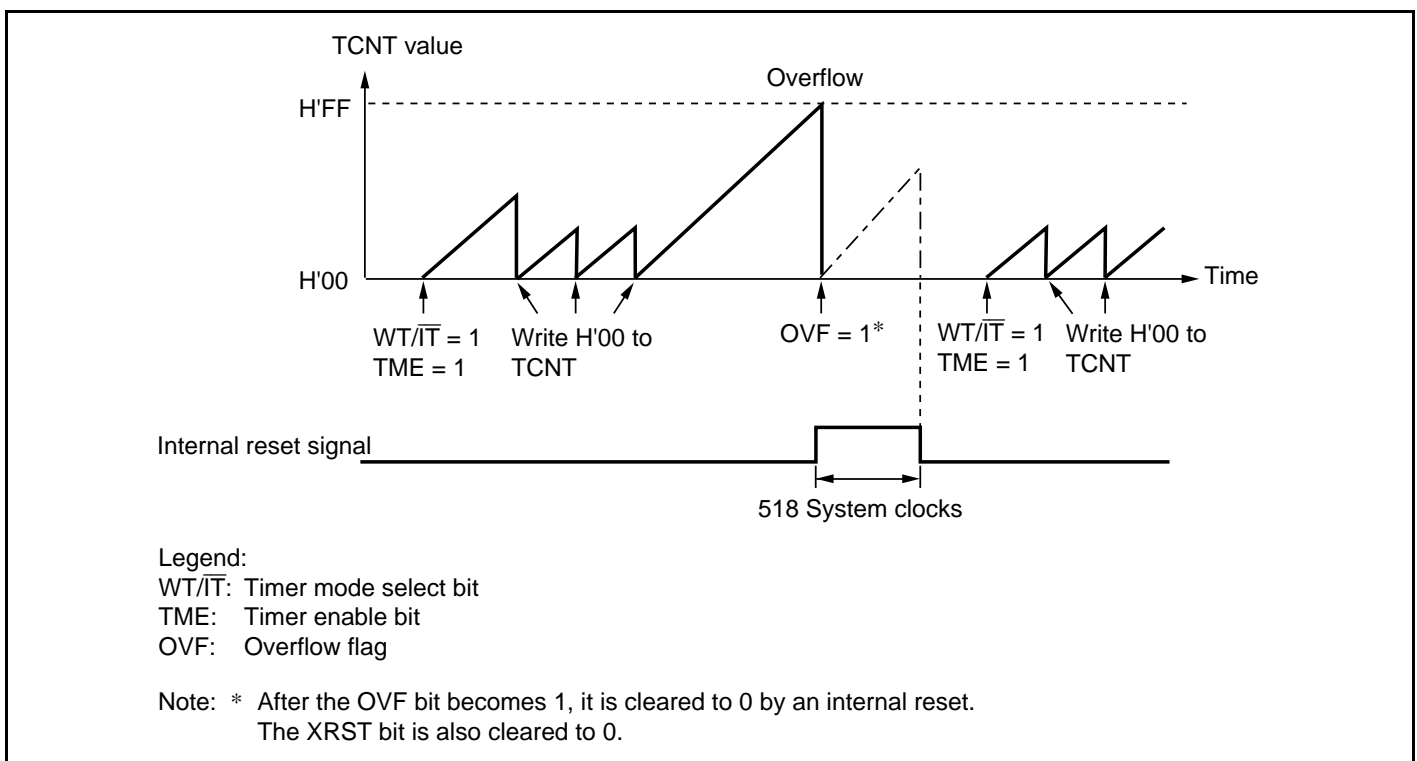
### 15.4.1 Watchdog Timer Mode

To use the WDT as a watchdog timer, set the  $\overline{\text{WT/IT}}$  bit and the TME bit in TCSR to 1. While the WDT is used as a watchdog timer, if TCNT overflows without being rewritten because of a system malfunction or another error, an internal reset or NMI interrupt request is generated. TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally by writing H'00) before overflows occurs.

If the  $\overline{\text{RST/NMI}}$  bit of TCSR is set to 1, when the TCNT overflows, an internal reset signal for this LSI is issued for 518 system clocks, and the low level signal is simultaneously output from the  $\overline{\text{RESO}}$  pin for 132 states, as shown in figure 15.2. If the  $\overline{\text{RST/NMI}}$  bit is cleared to 0, when the TCNT overflows, an NMI interrupt request is generated. Here, the output from the  $\overline{\text{RESO}}$  pin remains high.

An internal reset request from the watchdog timer and a reset input from the  $\overline{\text{RES}}$  pin are processed in the same vector. Reset source can be identified by the XRST bit status in SYSCR. If a reset caused by a signal input to the  $\overline{\text{RES}}$  pin occurs at the same time as a reset caused by a WDT overflow, the  $\overline{\text{RES}}$  pin reset has priority and the XRST bit in SYSCR is set to 1.

An NMI interrupt request from the watchdog timer and an interrupt request from the NMI pin are processed in the same vector. Do not handle an NMI interrupt request from the watchdog timer and an interrupt request from the NMI pin at the same time.

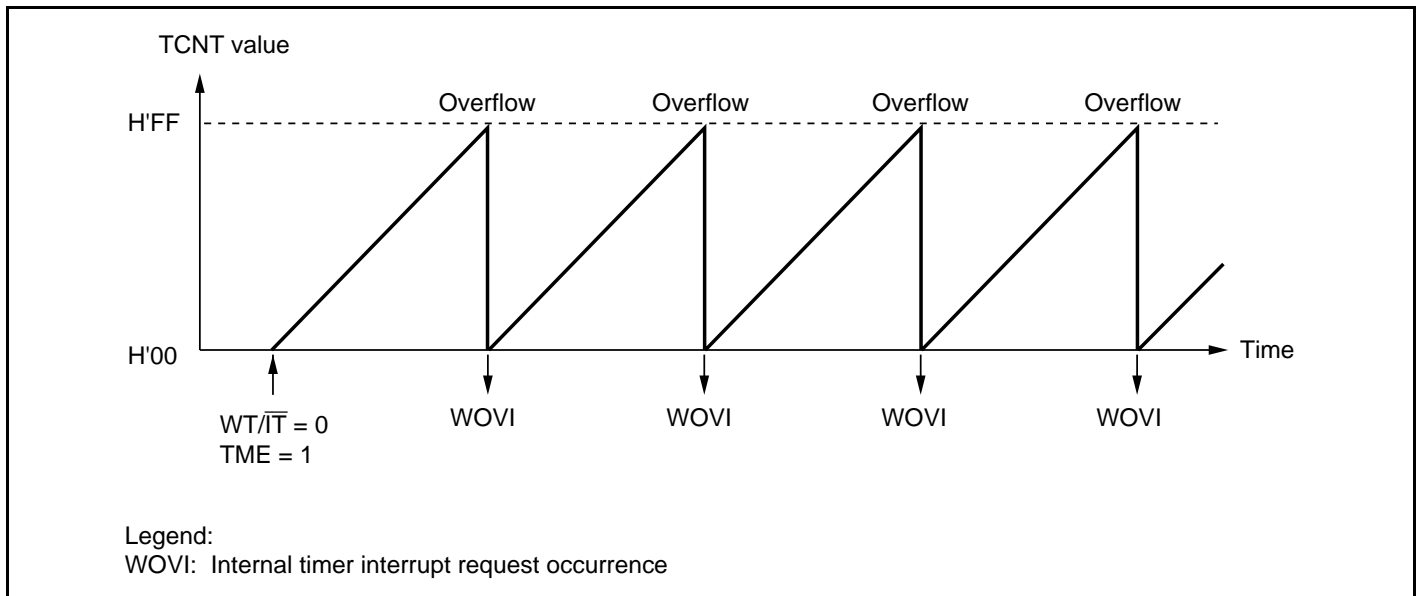


**Figure 15.2 Watchdog Timer Mode ( $\overline{\text{RST/NMI}} = 1$ ) Operation**

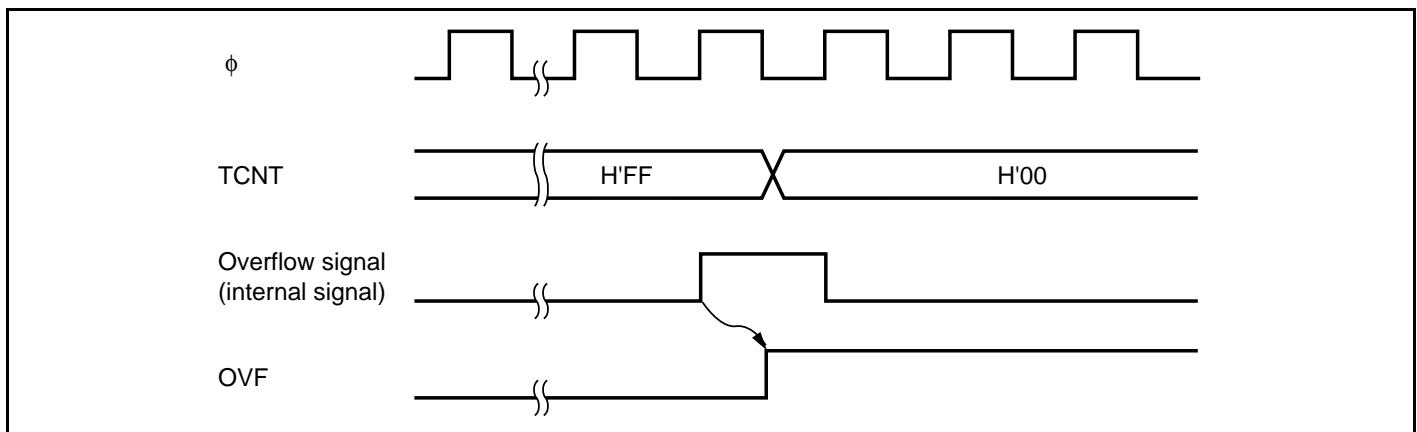


### 15.4.2 Interval Timer Mode

When the WDT is used as an interval timer, an interval timer interrupt (WOVI) is generated each time the TCNT overflows, as shown in figure 15.3. Therefore, an interrupt can be generated at intervals. When the TCNT overflows in interval timer mode, an interval timer interrupt (WOVI) is requested at the same time the OVF bit of TCSR is set to 1. The timing is shown figure 15.4.



**Figure 15.3 Interval Timer Mode Operation**



**Figure 15.4 OVF Flag Set Timing**

### 15.4.3 $\overline{\text{RESO}}$ Signal Output Timing

When TCNT overflows in watchdog timer mode, the OVF bit in TCSR is set to 1. When the RST/ $\overline{\text{NMI}}$  bit is 1 here, the internal reset signal is generated for the entire LSI. At the same time, the low level signal is output from the  $\overline{\text{RESO}}$  pin. The timing is shown in figure 15.5.

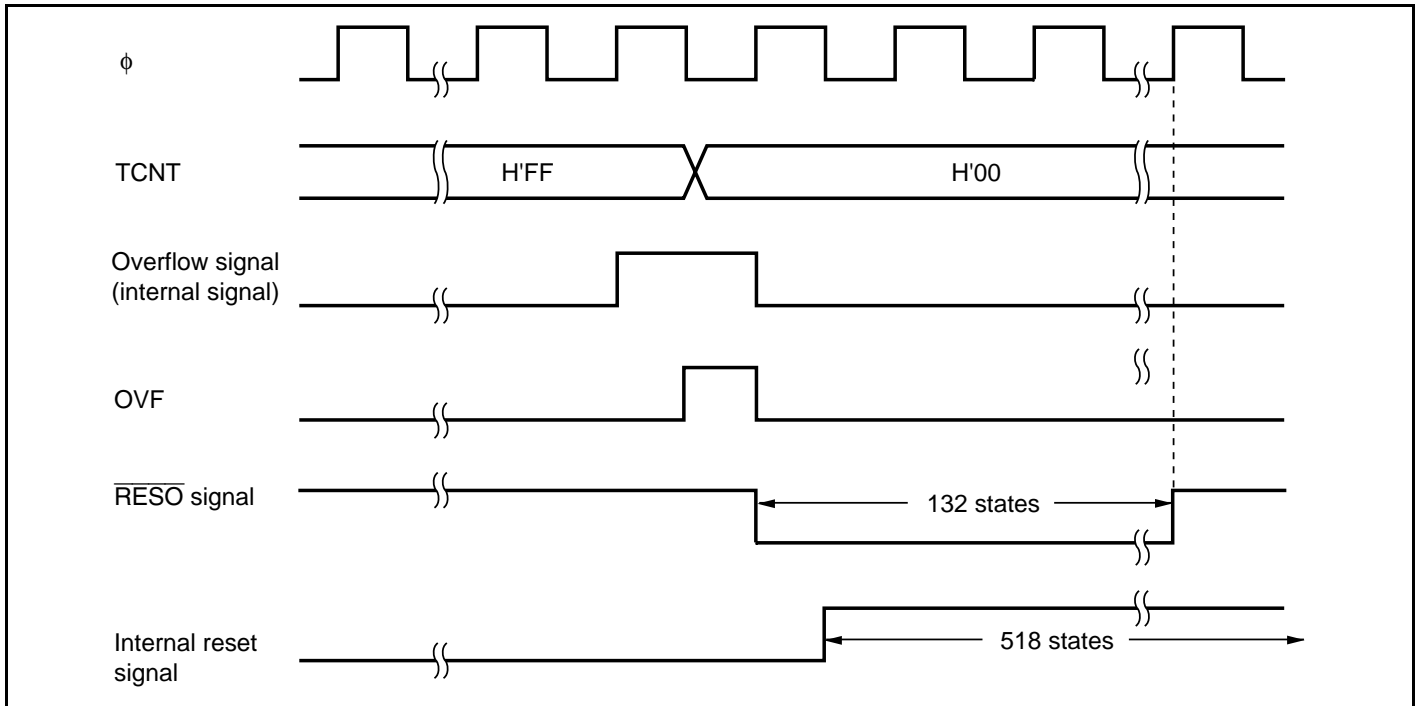


Figure 15.5 Output Timing of  $\overline{\text{RESO}}$  Signal

## 15.5 Interrupt Sources

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. OVF must be cleared to 0 in the interrupt handling routine.

When the NMI interrupt request is selected in watchdog timer mode, an NMI interrupt request is generated by an overflow.

Table 15.2 WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag	DTC Activation
WOVI	TCNT overflow	OVF	Not possible

## 15.6 Usage Notes

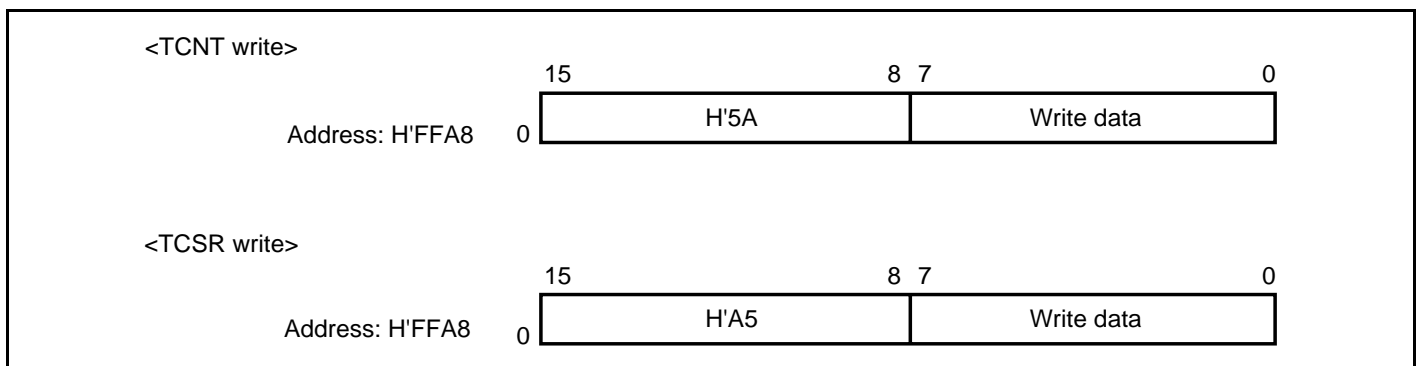
### 15.6.1 Notes on Register Access

The watchdog timer's registers, TCNT and TCSR differ from other registers in being more difficult to write to. The procedures for writing to and reading from these registers are given below.

#### Writing to TCNT and TCSR (Example of WDT\_0):

These registers must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

TCNT and TCSR both have the same write address. Therefore, satisfy the relative condition shown in figure 15.6 to write to TCNT or TCSR. To write to TCNT, the higher bytes must contain the value H'5A and the lower bytes must contain the write data before the transfer instruction execution. To write to TCSR, the higher bytes must contain the value H'A5 and the lower bytes must contain the write data.



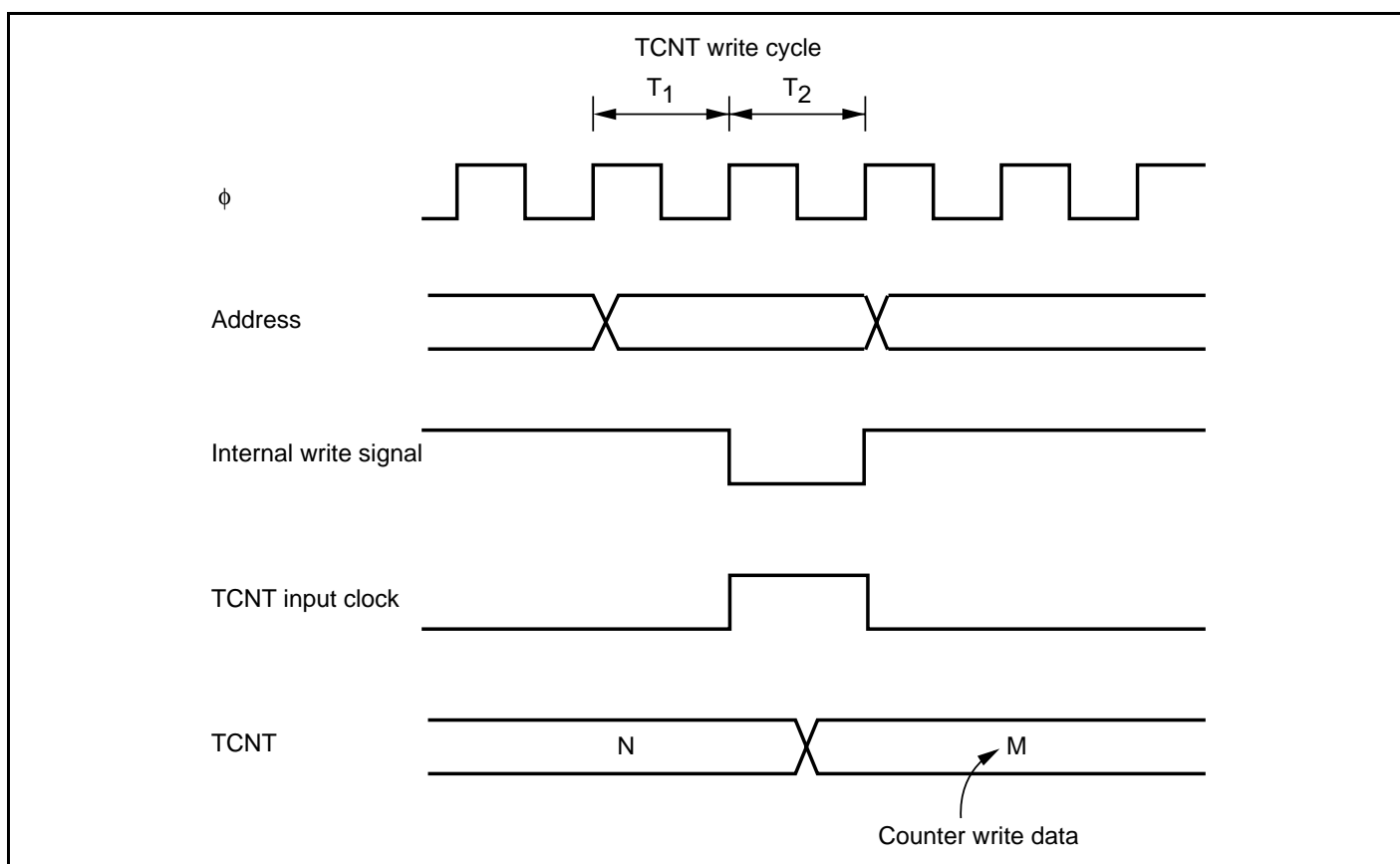
**Figure 15.6 Writing to TCNT and TCSR (WDT\_0)**

#### Reading from TCNT and TCSR (Example of WDT\_0):

These registers are read in the same way as other registers. The read address is H'FFA8 for TCSR and H'FFA9 for TCNT.

### 15.6.2 Conflict between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the T2 state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 15.7 shows this operation.



**Figure 15.7 Conflict between TCNT Write and Increment**

### 15.6.3 Changing Values of CKS2 to CKS0 Bits

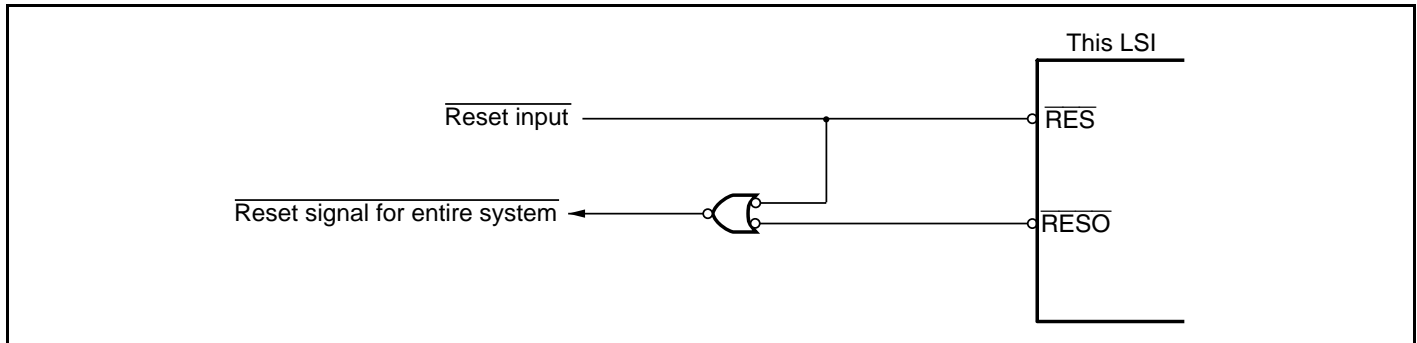
If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before changing the values of bits CKS2 to CKS0.

### 15.6.4 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer, while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

### 15.6.5 System Reset by $\overline{\text{RESO}}$ Signal

Inputting the  $\overline{\text{RESO}}$  output signal to the  $\overline{\text{RESO}}$  pin of this LSI prevents the LSI from being initialized correctly; the  $\overline{\text{RESO}}$  signal must not be logically connected to the  $\overline{\text{RES}}$  pin of the LSI. To reset the entire system by the  $\overline{\text{RESO}}$  signal, use the circuit as shown in figure 15.8.



**Figure 15.8 Sample Circuit for Resetting the System by the  $\overline{\text{RESO}}$  Signal**

### 15.6.6 Counter Values during Transitions between High-Speed, Sub-Active, and Watch Modes

When WDT\_1 is used as a clock counter and is allowed to transit between high-speed mode and sub-active or watch mode, the counter does not display the correct value due to internal clock switching.

Specifically, when transiting from high-speed mode to sub-active or watch mode, that is, when the control clock for WDT\_1 switches from the main clock to the sub-clock, the counter incrementing timing is delayed for approximately two to three clock cycles.

Similarly, when transiting from sub-active or watch mode to high-speed mode, the clock is not supplied until stabilized internal oscillation is available because the main clock pulse generator is halted in sub-clock mode. The counter is therefore prevented from incrementing for the time specified by the STS2 to STS0 bits in SBYCR after internal oscillation starts, thus producing counter value differences for this time.

Special care must be taken when using WDT\_1 as a clock counter. Note that no counter value difference is produced while operated in the same mode.



## Section 16 Serial Communication Interface (SCI, IrDA, and CRC)

This LSI has three independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clocked synchronous serial communication. Asynchronous serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface based on ISO/IEC 7816-3 (Identification Card) as an enhanced asynchronous communication function.

SCI\_1 can handle communication using the waveform based on the Infrared Data Association (IrDA) standard version 1.0. SCI\_0 and SCI\_2 provide high-speed communication at an average transfer rate of a specific system clock frequency. Reliable fast data transfers are secured using the internal cyclic redundancy check (CRC) operation circuit. Since the CRC operation circuit is not connected to the SCI, data is transferred to the circuit using the MOV instruction to be operated there.

### 16.1 Features

- Choice of asynchronous or clocked synchronous serial communication mode
- Full-duplex communication capability

The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected

The External clock can be selected as a transfer clock source (except for the smart card interface).
- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode 7-bit data)
- Four interrupt sources

Four interrupt sources — transmit-end, transmit-data-empty, receive-data-full, and receive error — that can issue requests.

The transmit-data-empty and receive-data-full interrupt sources can activate DTC. SCI\_0 and SCI\_2 can activate the RFU using the transmit-data-empty and receive-data-full interrupt sources.
- Module stop mode availability

**Asynchronous Mode:**

- Data length: 7 or 8 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the RxD pin level directly in case of a framing error
- Average transfer rate generator (SCI\_0 and SCI\_2): 460.606 kbps or 115.152 kbps selectable at 10.667-MHz operation; 720 kbps, 460.784 kbps, 230.392 kbps, or 115.196 kbps selectable at 16- or 24-MHz operation; and 230.392 kbps or 115.196 kbps selectable at 20-MHz operation

**Clocked Synchronous Mode:**

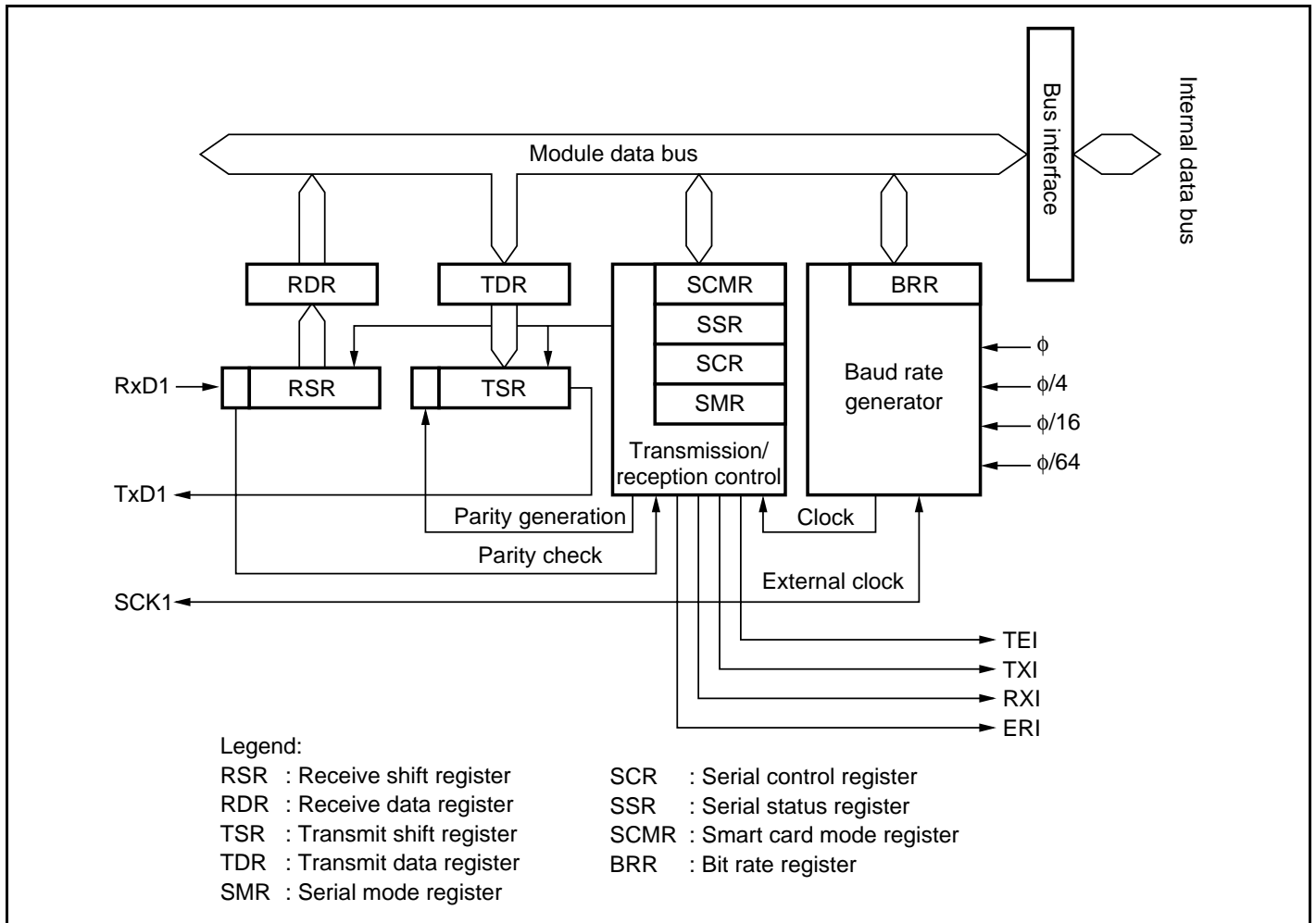
- Data length: 8 bits
- Receive error detection: Overrun errors
- SCI channel selectable (SCI\_0 and SCI\_2): When SSE0I = 1, TxD0 = high-impedance state and SCK0 = fixed to high input; when SSE2I = 1, TxD2 = high-impedance state and SCK2 = fixed to high input

**Smart Card Interface:**

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on detection of a error signal during transmission
- Both direct convention and inverse convention are supported

Figure 16.1 shows a block diagram of SCI\_1, and figure 16.2 shows a block diagram of SCI\_0 and SCI\_2.





**Figure 16.1 Block Diagram of SCI\_1**

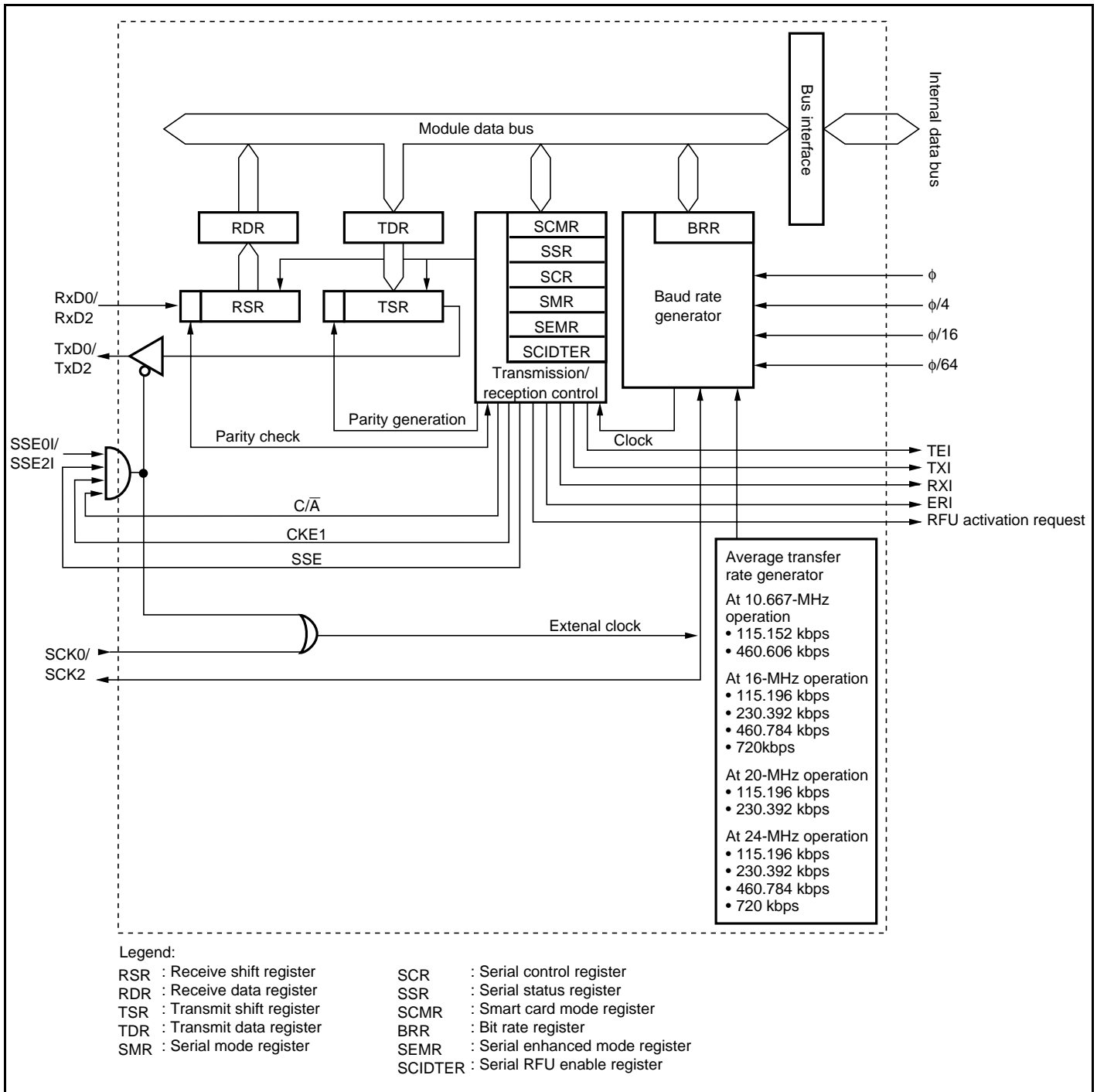


Figure 16.2 Block Diagram of SCI\_0 and SCI\_2

## 16.2 Input/Output Pins

Table 16.1 shows the input/output pins for each SCI channel.

**Table 16.1 Pin Configuration**

Channel	Symbol*	Input/Output	Function
0	SCK0	Input/Output	Channel 0 clock input/output
	RxD0	Input	Channel 0 receive data input
	TxD0	Output	Channel 0 transmit data output
	SSE0I	Input	Channel 0 stop input
1	SCK1	Input/Output	Channel 1 clock input/output
	RxD1/IrRxD	Input	Channel 1 receive data input (normal/IrDA)
	TxD1/IrTxD	Output	Channel 1 transmit data output (normal/IrDA)
2	SCK2	Input/Output	Channel 2 clock input/output
	RxD2	Input	Channel 2 receive data input
	TxD2	Output	Channel 2 transmit data output
	SSE2I	Input	Channel 2 stop input

Note: \* Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

## 16.3 Register Descriptions

The SCI has the following registers for each channel. Some bits in the serial mode register (SMR), serial status register (SSR), and serial control register (SCR) have different functions in different modes—normal serial communication interface mode and smart card interface mode; therefore, the bits are described separately for each mode in the corresponding register sections.

- Receive shift register (RSR)
- Receive data register (RDR)
- Transmit data register (TDR)
- Transmit shift register (TSR)
- Serial mode register (SMR)
- Serial control register (SCR)
- Serial status register (SSR)
- Smart card mode register (SCMR)
- Bit rate register (BRR)

- Serial interface control register (SCICR)\*<sup>1</sup>
- Serial enhanced mode register (SEMR)\*<sup>2</sup>
- Serial RFU enable register (SCIDTER)\*<sup>2</sup>

Notes: 1. SCICR is not available in SCI\_0 or SCI\_2.  
2. SEMR and SCIDTER are not available in SCI\_1.

### 16.3.1 Receive Shift Register (RSR)

RSR is a shift register used to receive serial data that converts it into parallel data. When one frame of data has been received, it is transferred to RDR automatically. RSR cannot be directly accessed by the CPU.

### 16.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one frame of serial data, it transfers the received serial data from RSR to RDR where it is stored. After this, RSR can receive the next data. Since RSR and RDR function as a double buffer in this way, continuous receive operations be performed. After confirming that the RDRF bit in SSR is set to 1, read RDR for only once. RDR cannot be written to by the CPU.

### 16.3.3 Transmit Data Register (TDR)

TDR is an 8-bit register that stores transmit data. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structures of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR when one frame of data is transmitted, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read from or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR for only once after confirming that the TDRE bit in SSR is set to 1.

### 16.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first transfers transmit data from TDR to TSR, then sends the data to the TxD pin. TSR cannot be directly accessed by the CPU.

### 16.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock source. Some bits in SMR have different functions in normal mode and smart card interface mode.

#### Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0)

Bit	Bit Name	Initial Value	R/W	Description
7	$C/\bar{A}$	0	R/W	Communication Mode 0: Asynchronous mode 1: Clocked synchronous mode
6	CHR	0	R/W	Character Length (enabled only in asynchronous mode) 0: Selects 8 bits as the data length. 1: Selects 7 bits as the data length. LSB-first is fixed and the MSB of TDR is not transmitted in transmission.  In clocked synchronous mode, a fixed data length of 8 bits is used.
5	PE	0	R/W	Parity Enable (enabled only in asynchronous mode)  When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. For a multiprocessor format, parity bit addition and checking are not performed regardless of the PE bit setting.
4	$O/\bar{E}$	0	R/W	Parity Mode (enabled 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 (enabled only in asynchronous mode)  Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits  In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.

Bit	Bit Name	Initial Value	R/W	Description
2	MP	0	R/W	<p>Multiprocessor Mode (enabled only in asynchronous mode)</p> <p>When this bit is set to 1, the multiprocessor communication function is enabled. The PE bit and O/<math>\bar{E}</math> bit settings are invalid in multiprocessor mode.</p>
1	CKS1	0	R/W	Clock Select 1,0
0	CKS0	0	R/W	<p>These bits select the clock source for the baud rate generator.</p> <p>00: <math>\phi</math> clock (n = 0)</p> <p>01: <math>\phi/4</math> clock (n = 1)</p> <p>10: <math>\phi/16</math> clock (n = 2)</p> <p>11: <math>\phi/64</math> clock (n = 3)</p> <p>For the relation between the bit rate register setting and the baud rate, see section 16.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 16.3.9, Bit Rate Register (BRR)).</p>

### 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>GS 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 etu from the start and the clock output control function is appended. For details, see section 16.7.8, Clock Output Control.</p>
6	BLK	0	R/W	<p>Setting this bit to 1 allows block transfer mode operation. For details, see section 16.7.3, 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 transmit data before transmission, and the parity bit is checked in reception. Set this bit to 1 in smart card interface mode.</p>

Bit	Bit Name	Initial Value	R/W	Description
4	O $\bar{E}$	0	R/W	<p>Parity Mode (valid only when the PE bit is 1 in asynchronous mode)</p> <p>0: Selects even parity</p> <p>1: Selects odd parity</p> <p>For details on the usage of this bit in smart card interface mode, see section 16.7.2, Data Format (Except in Block Transfer Mode).</p>
3	BCP1	0	R/W	Basic Clock Pulse 1,0
2	BCP0	0	R/W	<p>These bits select the number of basic clock cycles in a 1-bit data transfer time in smart card interface mode.</p> <p>00: 32 clock cycles (S = 32)</p> <p>01: 64 clock cycles (S = 64)</p> <p>10: 372 clock cycles (S = 372)</p> <p>11: 256 clock cycles (S = 256)</p> <p>For details, see section 16.7.4, Receive Data Sampling Timing and Reception Margin. S is described in section 16.3.9, Bit Rate Register (BRR).</p>
1	CKS1	0	R/W	Clock Select 1,0
0	CKS0	0	R/W	<p>These bits select the clock source for the baud rate generator.</p> <p>00: <math>\phi</math> clock (n = 0)</p> <p>01: <math>\phi/4</math> clock (n = 1)</p> <p>10: <math>\phi/16</math> clock (n = 2)</p> <p>11: <math>\phi/64</math> clock (n = 3)</p> <p>For the relation between the bit rate register setting and the baud rate, see section 16.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 16.3.9, Bit Rate Register (BRR)).</p>

### 16.3.6 Serial Control Register (SCR)

SCR is a register that performs enabling or disabling of SCI transfer operations and interrupt requests, and selection of the transfer clock source. For details on interrupt requests, see section 16.9, Interrupt Sources. Some bits in SCR have different functions in normal mode and smart card interface mode.

#### Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0)

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, a TXI interrupt request is enabled.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (enabled only when the MP bit in SMR is 1 in asynchronous mode) When this bit is set to 1, receive data in which the multiprocessor bit is 0 is skipped, and setting of the RDRF, FER, and ORER status flags in SSR is disabled. On receiving data in which the multiprocessor bit is 1, this bit is automatically cleared and normal reception is resumed. For details, see section 16.5, Multiprocessor Communication Function.
2	TEIE	0	R/W	Transmit End Interrupt Enable When this bit is set to 1, a TEI interrupt request is enabled.



Bit	Bit Name	Initial Value	R/W	Description
1	CKE1	0	R/W	Clock Enable 1,0
0	CKE0	0	R/W	<p>These bits select the clock source and SCK pin function.</p> <p>Asynchronous mode</p> <p>00: Internal clock (SCK pin functions as I/O port.)</p> <p>01: Internal clock (Outputs a clock of the same frequency as the bit rate from the SCK pin.)</p> <p>1X: External clock (Inputs a clock with a frequency 16 times the bit rate from the SCK pin.)</p> <p>Clocked synchronous mode</p> <p>0X: Internal clock (SCK pin functions as clock output.)</p> <p>1X: External clock (SCK pin functions as clock input.)</p>

Legend:

X: Don't care

**Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1)**

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, a TXI interrupt request is enabled.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (enabled only when the MP 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.
1	CKE1	0	R/W	Clock Enable 1,0
0	CKE0	0	R/W	Controls the clock output from the SCK pin. In GSM mode, clock output can be dynamically switched. For details, see section 16.7.8, Clock Output Control. When GM in SMR = 0 00: Output disabled (SCK pin functions as I/O port.) 01: Clock output 1X: Reserved When GM in SMR = 1 00: Output fixed to low 01: Clock output 10: Output fixed to high 11: Clock output

Legend:

X: Don't care

**16.3.7 Serial Status Register (SSR)**

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. TDRE, RDRF, ORER, PER, and FER can only be cleared. Some bits in SSR have different functions in normal mode and smart card interface mode.

**Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0)**

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	<p>Transmit Data Register Empty</p> <p>Indicates whether TDR contains transmit data.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When the TE bit in SCR is 0</li> <li>• When data is transferred from TDR to TSR and TDR is ready for data write</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE = 1</li> <li>• When a TXI interrupt request is issued allowing DTC to write data to TDR</li> <li>• When RFU is activated by TDRE = 1 allowing data to be written to TDR (only for SCI_0 and SCI_2)</li> </ul>
6	RDRF	0	R/(W)*	<p>Receive Data Register Full</p> <p>Indicates that receive data is stored in RDR.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When serial reception ends normally and receive data is transferred from RSR to RDR</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to RDRF after reading RDRF = 1</li> <li>• When an RXI interrupt request is issued allowing DTC to read data from RDR</li> <li>• When RFU is activated by RDRE = 1 allowing data to be read from RDR (only for SCI_0 and SCI_2)</li> </ul> <p>The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.</p>

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	<p>Overrun Error</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the next serial reception is completed while RDRF = 1</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to ORER after reading ORER = 1</li> </ul>
4	FER	0	R/(W)*	<p>Framing Error</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the stop bit is 0</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to FER after reading FER = 1</li> </ul> <p>In 2-stop-bit mode, only the first stop bit is checked.</p>
3	PER	0	R/(W)*	<p>Parity Error</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a parity error is detected during reception</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to PER after reading PER = 1</li> </ul>
2	TEND	1	R	<p>Transmit End</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>When the TE bit in SCR is 0</li> <li>When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written to TDRE after reading TDRE = 1</li> <li>When a TXI interrupt request is issued allowing DTC to write data to TDR</li> <li>When RFU is activated by TDRE = 1 allowing data to be written to TDR (only for SCI_0 and SCI_2)</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
1	MPB	0	R	Multiprocessor Bit MPB stores the multiprocessor bit in the receive frame. When the RE bit in SCR is cleared to 0 its previous state is retained.
0	MPBT	0	R/W	Multiprocessor Bit Transfer MPBT stores the multiprocessor bit to be added to the transmit frame.

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

### Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1)

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates whether TDR contains transmit data. [Setting conditions] <ul style="list-style-type: none"> <li>• When the TE bit in SCR is 0</li> <li>• When data is transferred from TDR to TSR, and TDR can be written to.</li> </ul> [Clearing conditions] <ul style="list-style-type: none"> <li>• When 0 is written to TDRE after reading TDRE = 1</li> <li>• When a TXI interrupt request is issued allowing DTC to write data to TDR</li> <li>• When BLK in SMR is 0, and RFU is activated by TEND = 1 allowing data to be written to TDR (only for SCI_0 and SCI_2)</li> <li>• When BLK in SMR is 1 and RFU is activated by TDRE = 1 allowing data to be written to TDR (only for SCI_0 and SCI_2)</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
6	RDRF	0	R/(W)*	<p>Receive Data Register Full</p> <p>Indicates that receive data is stored in RDR.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When serial reception ends normally and receive data is transferred from RSR to RDR</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written to RDRF after reading RDRF = 1</li> <li>When an RXI interrupt request is issued allowing DTC to read data from RDR</li> <li>When RFU is activated by RDRF = 1 allowing data to be read from RDR (only for SCI_0 and SCI_2)</li> </ul> <p>The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.</p>
5	ORER	0	R/(W)*	<p>Overrun Error</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the next serial reception is completed while RDRF = 1</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to ORER after reading ORER = 1</li> </ul>
4	ERS	0	R/(W)*	<p>Error Signal Status</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a low error signal is sampled</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to ERS after reading ERS = 1</li> </ul>
3	PER	0	R/(W)*	<p>Parity Error</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a parity error is detected during reception</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>When 0 is written to PER after reading PER = 1</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	TEND	1	R	<p>Transmit End</p> <p>TEND is set to 1 when the receiving end acknowledges no error signal and the next transmit data is ready to be transferred to TDR.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When both TE and EPS in SCR are 0</li> <li>• When ERS = 0 and TDRE = 1 after a specified time passed after the start of 1-byte data transfer. The set timing depends on the register setting as follows.</li> </ul> <p>When GM = 0 and BLK = 0, 2.5 etu after transmission start</p> <p>When GM = 0 and BLK = 1, 1.5 etu after transmission start</p> <p>When GM = 1 and BLK = 0, 1.0 etu after transmission start</p> <p>When GM = 1 and BLK = 1, 1.0 etu after transmission start</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written to TEND after reading TEND = 1</li> <li>• When a TXI interrupt request is issued allowing DTC to write the next data to TDR</li> <li>• When BLK in SMR is 0 and RFU is activated by TEND = 1 allowing data to be written to TDR (only for SCI_0 and SCI_2)</li> <li>• When BLK in SMR is 1 and RFU is activated by TDRE = 1 allowing data to be written to TDR (only for SCI_0 and SCI_2)</li> </ul>
1	MPB	0	R	<p>Multiprocessor Bit</p> <p>Not used in smart card interface mode.</p>
0	MPBT	0	R/W	<p>Multiprocessor Bit Transfer</p> <p>Write 0 to this bit in smart card interface mode.</p>

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

### 16.3.8 Smart Card Mode Register (SCMR)

SCMR selects smart card interface mode and its format.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.
3	SDIR	0	R/W	Smart Card Data Transfer Direction Selects the serial/parallel conversion format. 0: TDR contents are transmitted with LSB-first. Stores receive data as LSB first in RDR. 1: TDR contents are transmitted with MSB-first. Stores receive data as MSB first in RDR. The SDIR bit is valid only when the 8-bit data format is used for transmission/reception; when the 7-bit data format is used, data is always transmitted/received with LSB-first.
2	SINV	0	R/W	Smart Card Data Invert Specifies inversion of the data logic level. The SINV bit does not affect the logic level of the parity bit. When the parity bit is inverted, invert the O/ $\bar{E}$ bit in SMR. 0: TDR contents are transmitted as they are. Receive data is stored as it is in RDR. 1: TDR contents are inverted before being transmitted. Receive data is stored in inverted form in RDR.
1	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
0	SMIF	0	R/W	Smart Card Interface Mode Select: When this bit is set to 1, smart card interface mode is selected. 0: Normal asynchronous or clocked synchronous mode 1: Smart card interface mode



### 16.3.9 Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 16.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clocked synchronous mode, and smart card interface mode. The initial value of BRR is H'FF, and it can be read from or written to by the CPU at all times.

**Table 16.2 Relationships between N Setting in BRR and Bit Rate B**

Mode	Bit Rate	Error
Asynchronous mode	$B = \frac{\phi \times 10^6}{64 \times 2^{2n-1} \times (N + 1)}$	$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N + 1)} - 1 \right\} \times 100$
Clocked synchronous mode	$B = \frac{\phi \times 10^6}{8 \times 2^{2n-1} \times (N + 1)}$	—
Smart card interface mode	$B = \frac{\phi \times 10^6}{S \times 2^{2n+1} \times (N + 1)}$	$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N + 1)} - 1 \right\} \times 100$

Notes: B: Bit rate (bit/s)

N: BRR setting for baud rate generator ( $0 \leq N \leq 255$ )

$\phi$ : Operating frequency (MHz)

n and S: Determined by the SMR settings shown in the following table.

SMR Setting			SMR Setting		
CKS1	CKS0	n	BCP1	BCP0	S
0	0	0	0	0	32
0	1	1	0	1	64
1	0	2	1	0	372
1	1	3	1	1	256

Table 16.3 shows sample N settings in BRR in normal asynchronous mode. Table 16.4 shows the maximum bit rate settable for each frequency. Table 16.6 shows sample N settings in BRR in clocked synchronous mode, and table 16.8 shows sample N settings in BRR in smart card interface mode. In smart card interface mode, the number of basic clock cycles S in a 1-bit data transfer time can be selected. For details, see section 16.7.4, Receive Data Sampling Timing and Reception Margin. Tables 16.5 and 16.7 show the maximum bit rates with external clock input.

**Table 16.3 BRR Settings for Various Bit Rates (Asynchronous Mode)**

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	2			2.097152			2.4576			3		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	141	0.03	1	148	-0.04	1	174	-0.26	1	212	0.03
150	1	103	0.16	1	108	0.21	1	127	0.00	1	155	0.16
300	0	207	0.16	0	217	0.21	0	255	0.00	1	77	0.16
600	0	103	0.16	0	108	0.21	0	127	0.00	0	155	0.16
1200	0	51	0.16	0	54	-0.70	0	63	0.00	0	77	0.16
2400	0	25	0.16	0	26	1.14	0	31	0.00	0	38	0.16
4800	0	12	0.16	0	13	-2.48	0	15	0.00	0	19	-2.34
9600	—	—	—	0	6	-2.48	0	7	0.00	0	9	-2.34
19200	—	—	—	—	—	—	0	3	0.00	0	4	-2.34
31250	0	1	0.00	—	—	—	—	—	—	0	2	0.00
38400	—	—	—	—	—	—	0	1	0.00	—	—	—

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	3.6864			4			4.9152			5		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	64	0.70	2	70	0.03	2	86	0.31	2	88	-0.25
150	1	191	0.00	1	207	0.16	1	255	0.00	2	64	0.16
300	1	95	0.00	1	103	0.16	1	127	0.00	1	129	0.16
600	0	191	0.00	0	207	0.16	0	255	0.00	1	64	0.16
1200	0	95	0.00	0	103	0.16	0	127	0.00	0	129	0.16
2400	0	47	0.00	0	51	0.16	0	63	0.00	0	64	0.16
4800	0	23	0.00	0	25	0.16	0	31	0.00	0	32	-1.36
9600	0	11	0.00	0	12	0.16	0	15	0.00	0	15	1.73
19200	0	5	0.00	—	—	—	0	7	0.00	0	7	1.73
31250	—	—	—	0	3	0.00	0	4	-1.70	0	4	0.00
38400	0	2	0.00	—	—	—	0	3	0.00	0	3	1.73

Legend:

—: Can be set, but there will be a degree of error.

Note: Make the settings so that the error does not exceed 1%.

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	6			6.144			7.3728			8		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	106	-0.44	2	108	0.08	2	130	-0.07	2	141	0.03
150	2	77	0.16	2	79	0.00	2	95	0.00	2	103	0.16
300	1	155	0.16	1	159	0.00	1	191	0.00	1	207	0.16
600	1	77	0.16	1	79	0.00	1	95	0.00	1	103	0.16
1200	0	155	0.16	0	159	0.00	0	191	0.00	0	207	0.16
2400	0	77	0.16	0	79	0.00	0	95	0.00	0	103	0.16
4800	0	38	0.16	0	39	0.00	0	47	0.00	0	51	0.16
9600	0	19	-2.34	0	19	0.00	0	23	0.00	0	25	0.16
19200	0	9	-2.34	0	9	0.00	0	11	0.00	0	12	0.16
31250	0	5	0.00	0	5	2.40	—	—	—	0	7	0.00
38400	0	4	-2.34	0	4	0.00	0	5	0.00	—	—	—

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	9.8304			10			12			12.288		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	174	-0.26	2	177	-0.25	2	212	0.03	2	217	0.08
150	2	127	0.00	2	129	0.16	2	155	0.16	2	159	0.00
300	1	255	0.00	2	64	0.16	2	77	0.16	2	79	0.00
600	1	127	0.00	1	129	0.16	1	155	0.16	1	159	0.00
1200	0	255	0.00	1	64	0.16	1	77	0.16	1	79	0.00
2400	0	127	0.00	0	129	0.16	0	155	0.16	0	159	0.00
4800	0	63	0.00	0	64	0.16	0	77	0.16	0	79	0.00
9600	0	31	0.00	0	32	-1.36	0	38	0.16	0	39	0.00
19200	0	15	0.00	0	15	1.73	0	19	-2.34	0	19	0.00
31250	0	9	-1.70	0	9	0.00	0	11	0.00	0	11	2.40
38400	0	7	0.00	0	7	1.73	0	9	-2.34	0	9	0.00

Legend:

—: Can be set, but there will be a degree of error.

Note: Make the settings so that the error does not exceed 1%.

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	14			14.7456			16			17.2032		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	248	-0.17	3	64	0.70	3	70	0.03	3	75	0.48
150	2	181	0.16	2	191	0.00	2	207	0.16	2	223	0.00
300	2	90	0.16	2	95	0.00	2	103	0.16	2	111	0.00
600	1	181	0.16	1	191	0.00	1	207	0.16	1	223	0.00
1200	1	90	0.16	1	95	0.00	1	103	0.16	1	111	0.00
2400	0	181	0.16	0	191	0.00	0	207	0.16	0	223	0.00
4800	0	90	0.16	0	95	0.00	0	103	0.16	0	111	0.00
9600	0	45	-0.93	0	47	0.00	0	51	0.16	0	55	0.00
19200	0	22	-0.93	0	23	0.00	0	25	0.16	0	27	0.00
31250	0	13	0.00	0	14	-1.70	0	15	0.00	0	16	1.20
38400	—	—	—	0	11	0.00	0	12	0.16	0	16	0.00

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	18			19.6608			20			25		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	79	-0.12	3	86	0.31	3	88	-0.25	3	110	-0.02
150	2	233	0.16	2	255	0.00	3	64	0.16	3	80	-0.47
300	2	116	0.16	2	127	0.00	2	129	0.16	2	162	0.15
600	1	233	0.16	1	255	0.00	2	64	0.16	2	80	-0.47
1200	1	166	0.16	1	127	0.00	1	129	0.16	1	162	0.15
2400	0	233	0.16	0	255	0.00	1	64	0.16	1	80	-0.47
4800	0	166	0.16	0	127	0.00	0	129	0.16	0	162	0.15
9600	0	58	-0.69	0	63	0.00	0	64	0.16	0	80	-0.47
19200	0	28	1.02	0	31	0.00	0	32	-1.36	0	40	-0.76
31250	0	17	0.00	0	19	-1.70	0	19	0.00	0	24	0.00
38400	0	14	-2.34	0	15	0.00	0	15	1.73	0	19	1.73

Legend:

—: Can be set, but there will be a degree of error.

Note: Make the settings so that the error does not exceed 1%.

**Table 16.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode)**

$\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N	$\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N
2	62500	0	0	9.8304	307200	0	0
2.097152	65536	0	0	10	312500	0	0
2.4576	76800	0	0	12	375000	0	0
3	93750	0	0	12.288	384000	0	0
3.6864	115200	0	0	14	437500	0	0
4	125000	0	0	14.7456	460800	0	0
4.9152	153600	0	0	16	500000	0	0
5	156250	0	0	17.2032	537600	0	0
6	187500	0	0	18	562500	0	0
6.144	192000	0	0	19.6608	614400	0	0
7.3728	230400	0	0	20	625000	0	0
8	250000	0	0	25	781250	0	0

**Table 16.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)**

$\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	$\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
2	0.5000	31250	9.8304	2.4576	153600
2.097152	0.5243	32768	10	2.5000	156250
2.4576	0.6144	38400	12	3.0000	187500
3	0.7500	46875	12.288	3.0720	192000
3.6864	0.9216	57600	14	3.5000	218750
4	1.0000	62500	14.7456	3.6864	230400
4.9152	1.2288	76800	16	4.0000	250000
5	1.2500	78125	17.2032	4.3008	268800
6	15.000	93750	18	4.5000	281250
6.144	1.5360	96000	19.6608	4.9152	307200
7.3728	1.8432	115200	20	5.0000	312500
8	2.0000	125000	25	6.2500	390625

**Table 16.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)**

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)													
	2		4		8		10		16		20		25	
	n	N	n	N	n	N	n	N	n	N	n	N	n	N
110	3	70	—	—										
250	2	124	2	249	3	124	—	—	3	249				
500	1	249	2	124	2	249	—	—	3	124	—	—		
1k	1	124	1	249	2	124	—	—	2	249	—	—	3	97
2.5k	0	199	1	99	1	199	1	249	2	99	2	124	2	155
5k	0	99	0	199	1	99	1	124	1	199	1	249	2	77
10k	0	49	0	99	0	199	0	249	1	99	1	124	1	155
25k	0	19	0	39	0	79	0	99	0	159	0	199	0	249
50k	0	9	0	19	0	39	0	49	0	79	0	99	0	124
100k	0	4	0	9	0	19	0	24	0	39	0	49	0	62
250k	0	1	0	3	0	7	0	9	0	15	0	19	0	24
500k	0	0*	0	1*	0	3	0	4	0	7	0	9	—	—
1M			0	0	0	1			0	3	0	4	—	—
2.5M							0	0*			0	1	—	—
5M											0	0*	—	—

Legend:

Blank: Cannot be set.

—: Can be set, but there will be a degree of error.

\*: Continuous transfer or reception is not possible.

**Table 16.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)**

$\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	$\phi$ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
2	0.3333	333333.3	12	2.0000	2000000.0
4	0.6667	666666.7	14	2.3333	2333333.3
6	1.0000	1000000.0	16	2.6667	2666666.7
8	1.3333	1333333.3	18	3.0000	3000000.0
10	1.6667	1666666.7	20	3.3333	3333333.3
			25	4.1667	4166666.7

**Table 16.8 BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, s = 372)**

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)											
	7.1424			10.0000			10.7136			13.0000		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	0	0.00	0	1	30	0	1	25	0	1	8.99

Bit Rate (bit/s)	Operating Frequency $\phi$ (MHz)														
	14.2848			16.0000			18.0000			20.0000			25.0000		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	1	0.00	0	1	12.01	0	2	15.99	0	2	6.60	0	3	12.49

**Table 16.9 Maximum Bit Rate for Each Frequency (Smart Card Interface Mode, S = 372)**

$\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N	$\phi$ (MHz)	Maximum Bit Rate (bit/s)	n	N
7.1424	9600	0	0	14.2848	19200	0	0
10.0000	13441	0	0	16.0000	21505	0	0
10.7136	14400	0	0	18.0000	24194	0	0
13.0000	17473	0	0	20.0000	26882	0	0
				25.0000	33602	0	0

### 16.3.10 Serial Interface Control Register (SCICR)

SCICR controls IrDA operation of SCI\_1.

Bit	Bit Name	Initial Value	R/W	Description
7	IrE	0	R/W	IrDA Enable Specifies SCI_1 I/O pins for either normal SCI or IrDA. 0: TxD1/IrTxD and RxD1/IrRxD pins function as TxD1 and RxD1 pins, respectively 1: TxD1/IrTxD and RxD1/IrRxD pins function as IrTxD and IrRxD pins, respectively
6	IrCKS2	0	R/W	IrDA Clock Select 2 to 0
5	IrCKS1	0	R/W	Specifies the high-level width of the clock pulse during IrTxD output pulse encoding when the IrDA function is enabled. 000: B x 3/16 (three sixteenths of the bit rate) 001: $\phi/2$ 010: $\phi/4$ 011: $\phi/8$ 100: $\phi/16$ 101: $\phi/32$ 110: $\phi/64$ 111: $\phi/128$
4	IrCKS0	0	R/W	
3, 2	—	All 0	R/W	Reserved The initial value should not be changed.
1, 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

### 16.3.11 Serial Enhanced Mode Register\_0 and 2 (SEMR\_0 and SEMR\_2)

SEMR\_0 and SEMR\_2 select the SCI\_0 and SCI\_2 functions, respectively, and the clock source in asynchronous mode. The basic clock is automatically specified when the average transfer rate operation is selected.



Bit	Bit Name	Initial Value	R/W	Description
7	SSE	0	R/W	<p>SCI Select Enable</p> <p>Enables/disables the external pins to select the SCI functions when the external clock is supplied in clocked synchronous mode.</p> <p>0: Disables the external pins to select the SCI functions (normal)</p> <p>1: Enables the external pins to select the SCI functions</p> <ul style="list-style-type: none"> <li>• SCI_0           <p>SSE0I pin input = 0 (selected state): SCI_0 operates normally</p> <p>SSE0I pin input = 1 (non-selected state): SCI_0 halts operation</p> <p>(TxD0 = high-impedance state, SCK0 = fixed to high)</p> </li> <li>• SCI_2           <p>SSE2I pin input = 0 (selected state): SCI_2 operates normally</p> <p>SSE2I pin input = 1 (non-selected state): SCI_2 halts operation</p> <p>(TxD2 = high-impedance state, SCK2 = fixed to high)</p> </li> </ul>
6, 5	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>
4	ABCS	0	R/W	<p>Asynchronous Mode Basic Clock Select</p> <p>Specifies the basic clock for a 1-bit cycle in asynchronous mode.</p> <p>This bit is valid only in asynchronous mode (<math>C/\bar{A}</math> bit in SMR is 0).</p> <p>0: The basic clock has a frequency 16 times the transfer clock frequency (normal operation)</p> <p>1: The basic clock has a frequency 8 times the transfer clock frequency (double-speed operation)</p>

Bit	Bit Name	Initial Value	R/W	Description
3	ACS4	0	R/W	Asynchronous Mode Clock Source Select
2	ACS2	0	R/W	These bits specify the clock source and the average transfer rate in asynchronous mode.  These bits are valid only when external clock is supplied in asynchronous mode.
1	ACS1	0	R/W	
0	ACS0	0	R/W	
0000: Normal operation with external clock input and average transfer rate operation not used (operated using the basic clock with a frequency 16 or 8 times the transfer clock frequency)				
0001: Average transfer rate operation at 115.152 kbps when the system clock frequency is 10.667 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)				
0010: Average transfer rate operation at 460.606 kbps when the system clock frequency is 10.667 MHz (operated using the basic clock with a frequency 8 times the transfer clock frequency)				
0011: Reserved				
0100: Reserved				
0101: Average transfer rate operation at 115.196 kbps when the system clock frequency is 16 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)				
0110: Average transfer rate operation at 460.784 kbps when the system clock frequency is 16 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)				
0111: Average transfer rate operation at 720 kbps when the system clock frequency is 16 MHz (operated using the basic clock with a frequency 8 times the transfer clock frequency)				

Bit	Bit Name	Initial Value	R/W	Description
3	ACS4	0	R/W	1000: Average transfer rate operation at 115.196
2	ACS2	0	R/W	kbps when the system clock frequency is 16
1	ACS1	0	R/W	MHz (operated using the basic clock with a
0	ACS0	0	R/W	frequency 16 times the transfer clock frequency)
				1001: Average transfer rate operation at 230.392 kbps when the system clock frequency is 16 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)
				1011: Average transfer rate operation at 115.196 kbps when the system clock frequency is 20 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)
				1011: Average transfer rate operation at 230.392 kbps when the system clock frequency is 20 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)
				1100: Average transfer rate operation at 115.196 kbps when the system clock frequency is 24 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)
				1101: Average transfer rate operation at 230.392 kbps when the system clock frequency is 24 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)
				1110: Average transfer rate operation at 460.784 kbps when the system clock frequency is 24 MHz (operated using the basic clock with a frequency 16 times the transfer clock frequency)
				1111: Average transfer rate operation at 720 kbps when the system clock frequency is 24 MHz (operated using the basic clock with a frequency 8 times the transfer clock frequency)

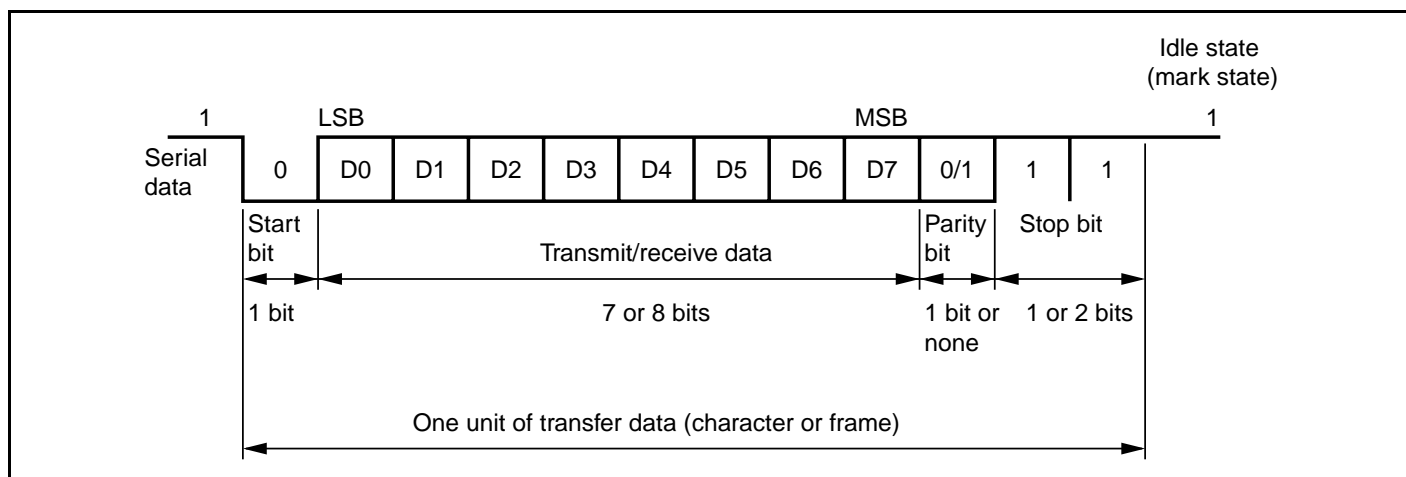
### 16.3.12 Serial RFU Enable Register\_0 and 2 (SCIDTER\_0 and SCIDTER\_2)

SCIDTER\_0 and SCIDTER\_2 enable or disable the RFU activation requests by SCI\_0 and SCI\_2, respectively.

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE_DTE	0	R/W	<p>TERE Data Transfer Enable</p> <p>Enables/disables the RFU to be activated by TDRE = 1.</p> <ul style="list-style-type: none"> <li>SMIF = 0 in SCMR, or SMIF = 1 and BLK = 1 in SMR           <ul style="list-style-type: none"> <li>0: Disables activation of the RFU by TDRE = 1 in SSR, and does not mask the TXI interrupt request</li> <li>1: Enables activation of the RFU by TDRE = 1 in SSR, and masks the TXI interrupt request</li> </ul> </li> </ul> <p>[Clearing condition]</p> <p>When data transfer has been completed by activation of the RFU by TDRE = 1 (FIFO EMPTY)</p> <ul style="list-style-type: none"> <li>SMIF = 1 in SCMR and BLK = 1 in SMR           <ul style="list-style-type: none"> <li>0: Disables activation of the RFU by TEND = 1 in SSR, and does not mask the TXI interrupt request</li> <li>1: Enables activation of the RFU by TEND = 1 in SSR, and masks the TXI interrupt request</li> </ul> </li> </ul> <p>[Clearing condition]</p> <p>When data transfer has been completed by the RFU activation by TEND = 1 in SSR</p>
6	RDRF_DTE	0	R/W	<p>RDRF Data Transfer Enable</p> <p>Enables/disables activation of the RFU by RDRF = 1 in SSR.</p> <ul style="list-style-type: none"> <li>0: Disables activation of the RFU, and does not mask the RXI interrupt request</li> <li>1: Enables activation of the RFU, and masks the RXI interrupt request</li> </ul> <p>[Clearing condition]</p> <p>When data transfer has been completed by the RFU activated by RDRF = 1 in SSR (FIFO FULL)</p>
5 to 0	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>

## 16.4 Operation in Asynchronous Mode

Figure 16.3 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transmit/receive data, a parity bit, and finally stop bits (high level). In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer and reception.



**Figure 16.3 Data Format in Asynchronous Communication  
(Example with 8-Bit Data, Parity, Two Stop Bits)**

### 16.4.1 Data Transfer Format

Table 16.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting. For details on the multiprocessor bit, see section 16.5, Multiprocessor Communication Function.

**Table 16.10 Serial Transfer Formats (Asynchronous Mode)**

SMR Settings				Serial Transmit/Receive Format and Frame Length												
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12	
0	0	0	0	S	8-bit data								STOP			
0	0	0	1	S	8-bit data								STOP	STOP		
0	1	0	0	S	8-bit data								P	STOP		
0	1	0	1	S	8-bit data								P	STOP	STOP	
1	0	0	0	S	7-bit data							STOP				
1	0	0	1	S	7-bit data							STOP	STOP			
1	1	0	0	S	7-bit data							P	STOP			
1	1	0	1	S	7-bit data							P	STOP	STOP		
0	—	1	0	S	8-bit data								MPB	STOP		
0	—	1	1	S	8-bit data								MPB	STOP	STOP	
1	—	1	0	S	7-bit data							MPB	STOP			
1	—	1	1	S	7-bit data							MPB	STOP	STOP		

Legend:

S : Start bit

STOP : Stop bit

P : Parity bit

MPB : Multiprocessor bit

— : Don't care

### 16.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the bit rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Since receive data is latched internally at the rising edge of the 8th pulse of the basic clock, data is latched at the middle of each bit, as shown in figure 16.4. Thus the reception margin in asynchronous mode is determined by formula (1) below.

$$M = \left\{ \left( 0.5 - \frac{1}{2N} \right) - \frac{D - 0.5}{N} - (L - 0.5) F \right\} \times 100 [\%] \quad \dots \text{Formula (1)}$$

M: Reception margin (%)

N: Ratio of bit rate to clock ( $N = 16$ )

D: Clock duty ( $D = 0.5$  to  $1.0$ )

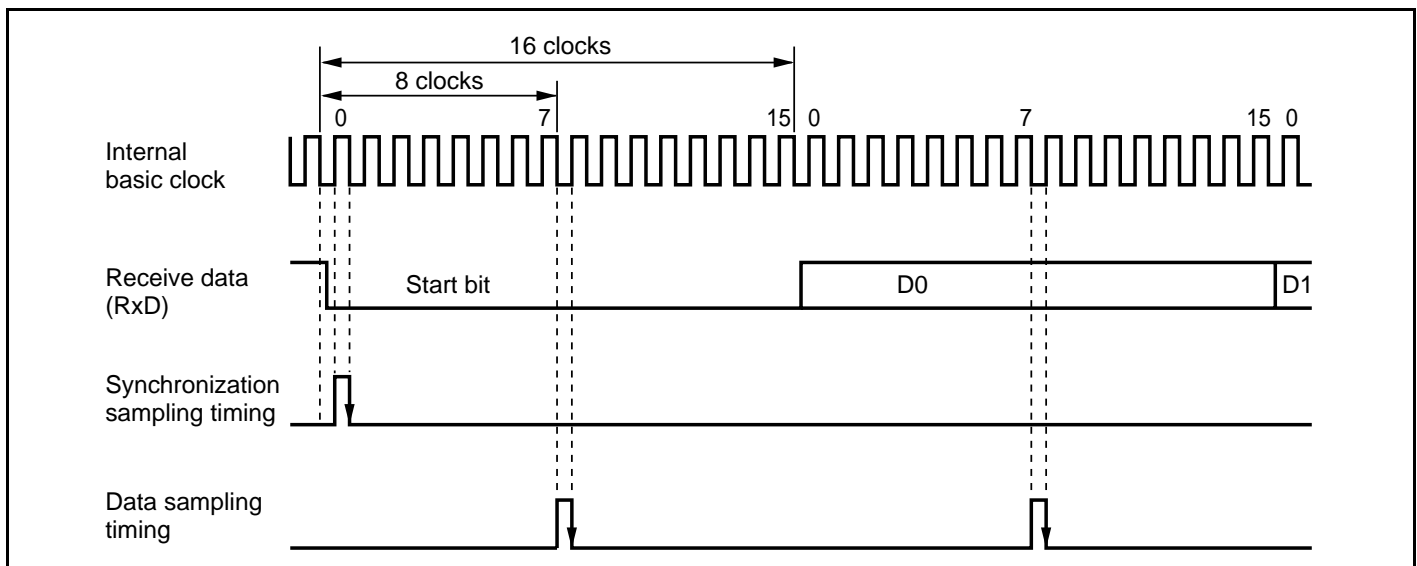
L: Frame length ( $L = 9$  to  $12$ )

F: Absolute value of clock rate deviation

Assuming values of  $F = 0$  and  $D = 0.5$  in formula (1), the reception margin is determined by the formula below.

$$M = \{ 0.5 - 1/(2 \times 16) \} \times 100[\%] = 46.875\%$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

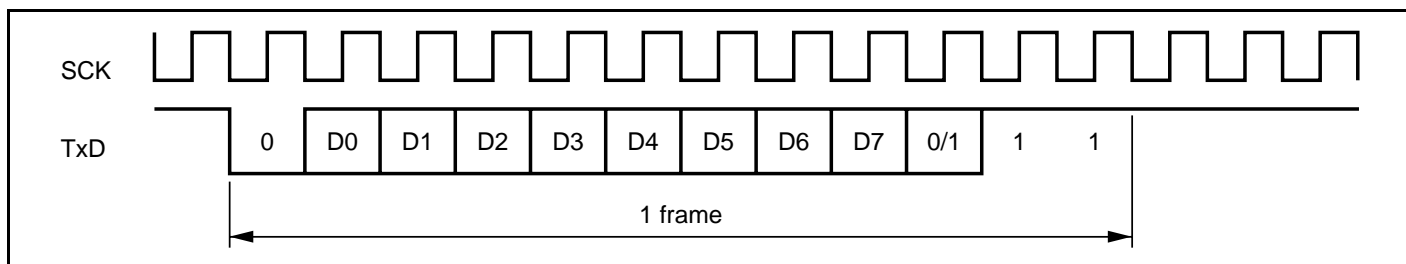


**Figure 16.4 Receive Data Sampling Timing in Asynchronous Mode**

### 16.4.3 Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK pin can be selected as the SCI's transfer clock, according to the setting of the  $C/\overline{A}$  bit in SMR and the CKE1 and CKE0 bits in SCR. When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 16.5.



**Figure 16.5 Relation between Output Clock and Transmit Data Phase (Asynchronous Mode)**



#### 16.4.4 Serial Enhanced Mode Clock

SCI\_0 and SCI\_2 can be operated not only based on the clocks described in section 16.4.3, Clock, but based on the following clocks, which are specified by the serial enhanced mode registers, SEMR\_0 and SEMR\_2.

**Double-Speed Operation:** Operations that are usually achieved using the clock with frequency 16 times the normal bit rate can be achieved using the clock with frequency 8 times the bit rate in this mode. That is, double transfer rate can be achieved using a single basic clock.

Double-speed operation can be specified by the ABCS bit in SEMR and is available for both clock sources of an internal clock generated by the on-chip baud rate generator and an external clock input at the SCK pin. However, double-speed operation cannot be specified when the average transfer rate operation is selected.

**Average Transfer Rate Operation:** The SCI can be operated based on the clock with an average transfer rate generated from the system clock instead of the external clock input at the SCK pin. In this case, the SCK pin is fixed to input.

Average transfer rate operation can be specified by the ACS4 and ACS2 to ACS0 bits in SEMR. Double-speed operation may be selected by clearing the ACS4 and ACS2 to ACS0 bits to 0.

Figures 16.6 and 16.7 show some examples of internal basic clock operations when average transfer rate operation is selected.

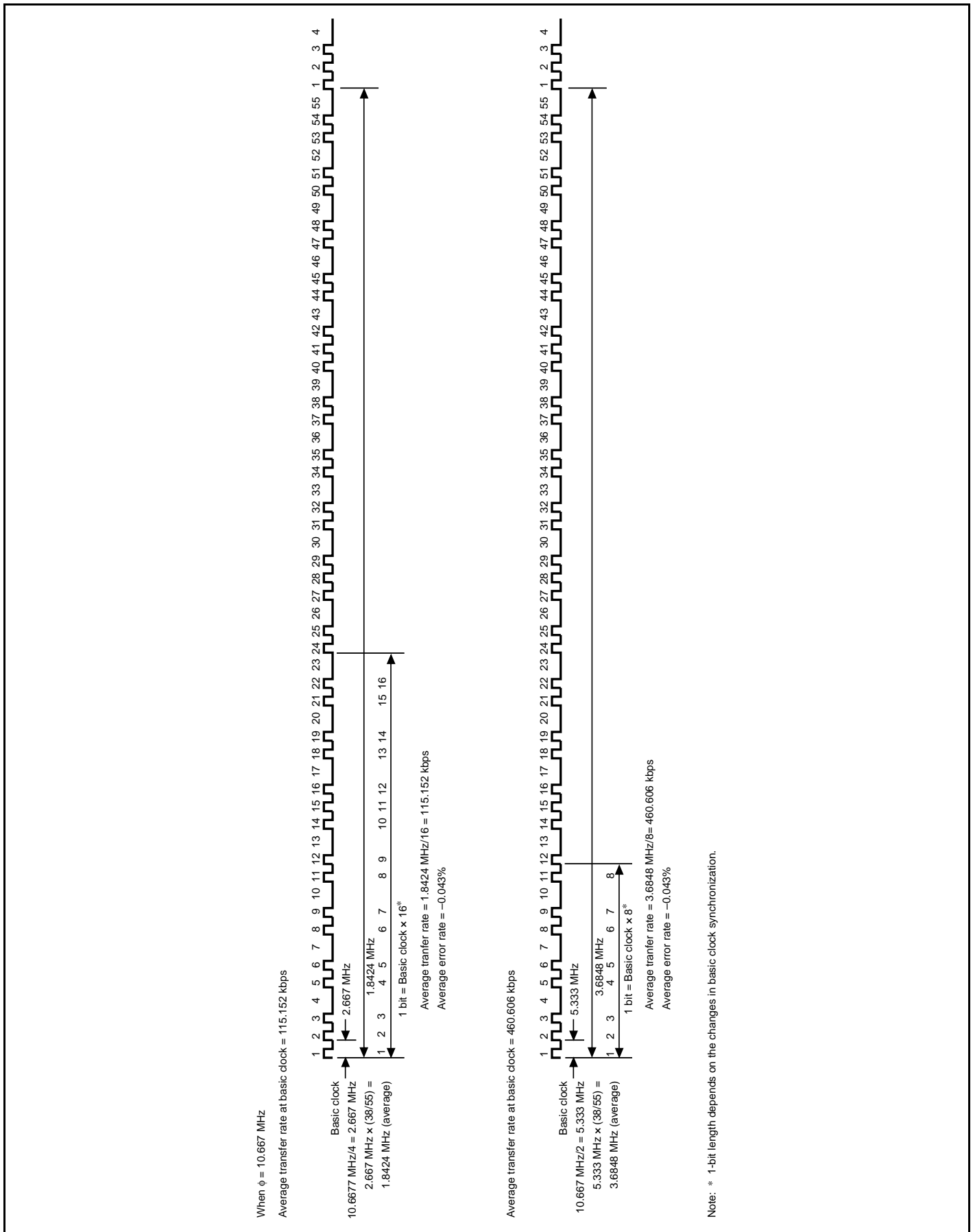


Figure 16.6 Basic Clock Examples When Average Transfer Rate Is Selected (1)

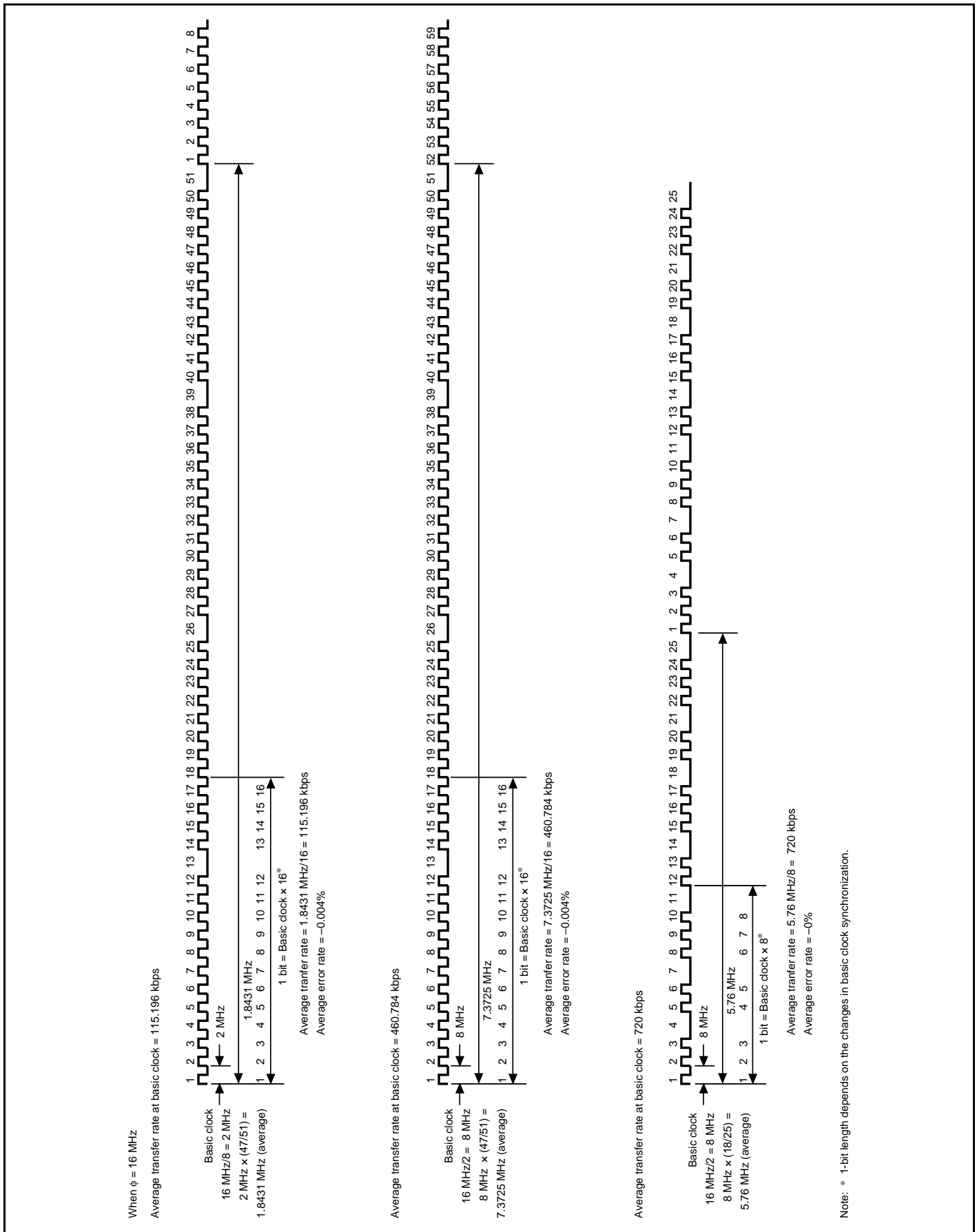
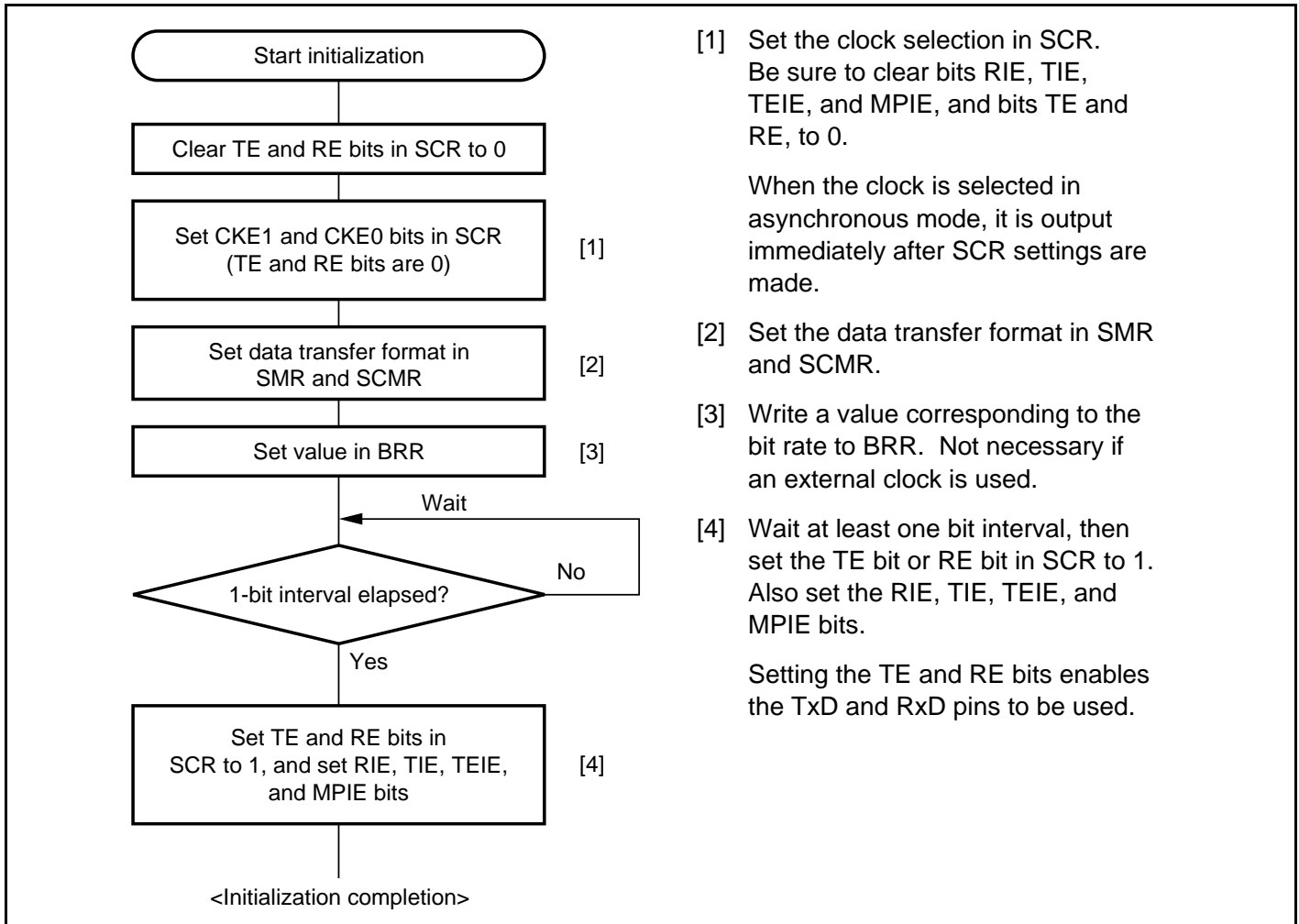


Figure 16.7 Basic Clock Examples When Average Transfer Rate Is Selected (2)

### 16.4.5 SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as shown in figure 16.8. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag in SSR is set to 1. Note that clearing the RE bit to 0 does not initialize the contents of the RDRF, PER, FER, and ORER flags in SSR, or the contents of RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization.



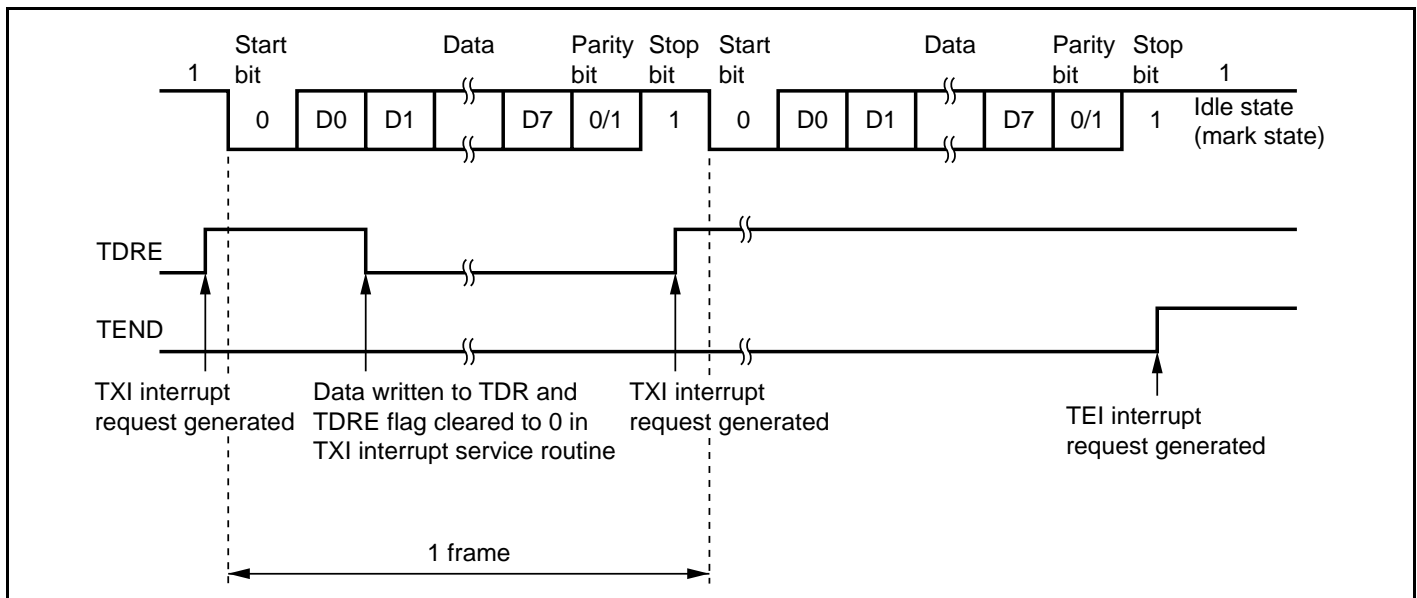
**Figure 16.8 Sample SCI Initialization Flowchart**

### 16.4.6 Serial Data Transmission (Asynchronous Mode)

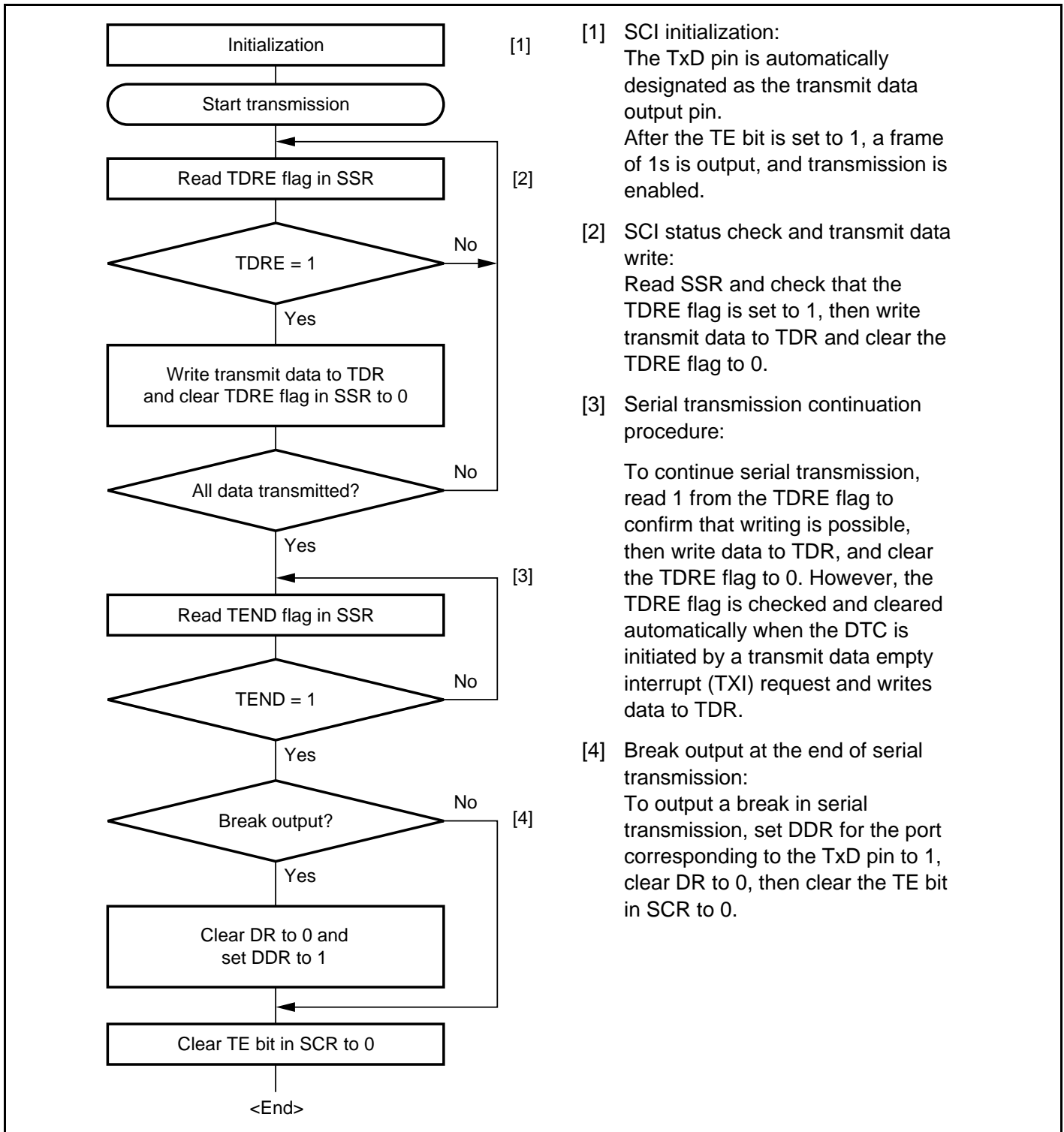
Figure 16.9 shows an example of the operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is cleared to 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a transmit data empty interrupt request (TXI) is generated. Because the TXI interrupt routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit or multiprocessor bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the “mark state” is entered in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 16.10 shows a sample flowchart for transmission in asynchronous mode.



**Figure 16.9 Example of Operation in Transmission in Asynchronous Mode  
(Example with 8-Bit Data, Parity, One Stop Bit)**

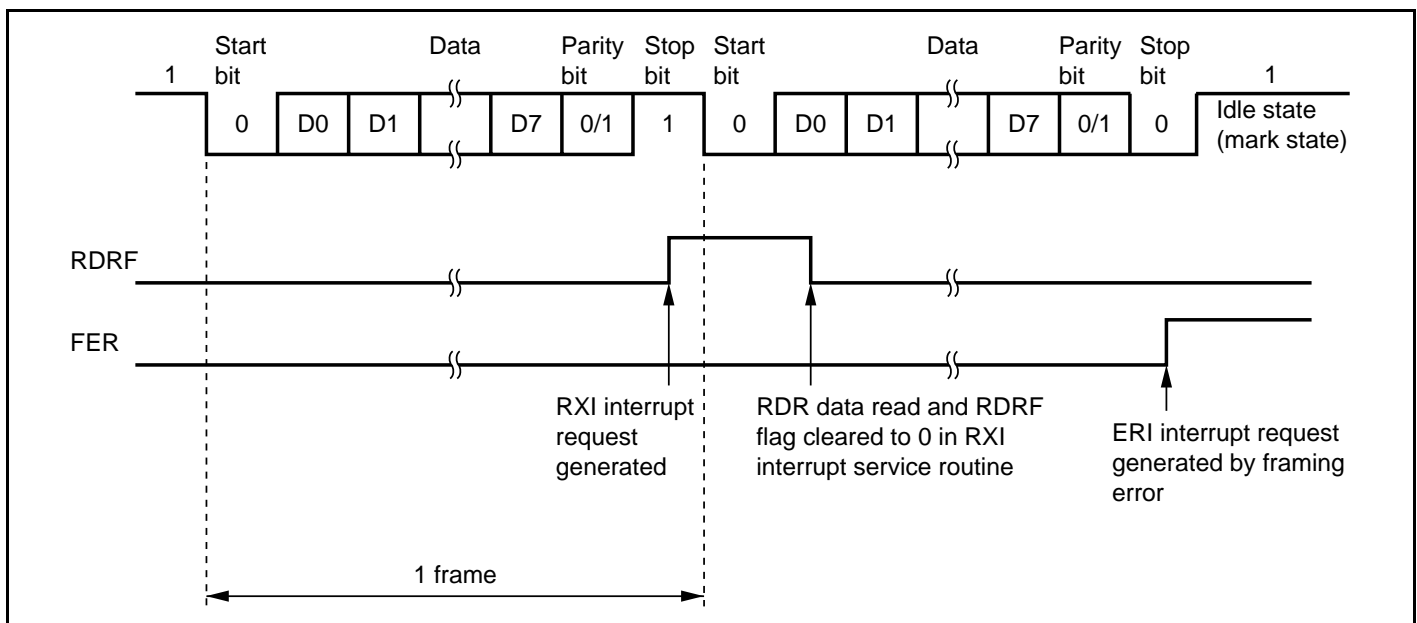


**Figure 16.10 Sample Serial Transmission Flowchart**

### 16.4.7 Serial Data Reception (Asynchronous Mode)

Figure 16.11 shows an example of the operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, receives receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.



**Figure 16.11 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit)**

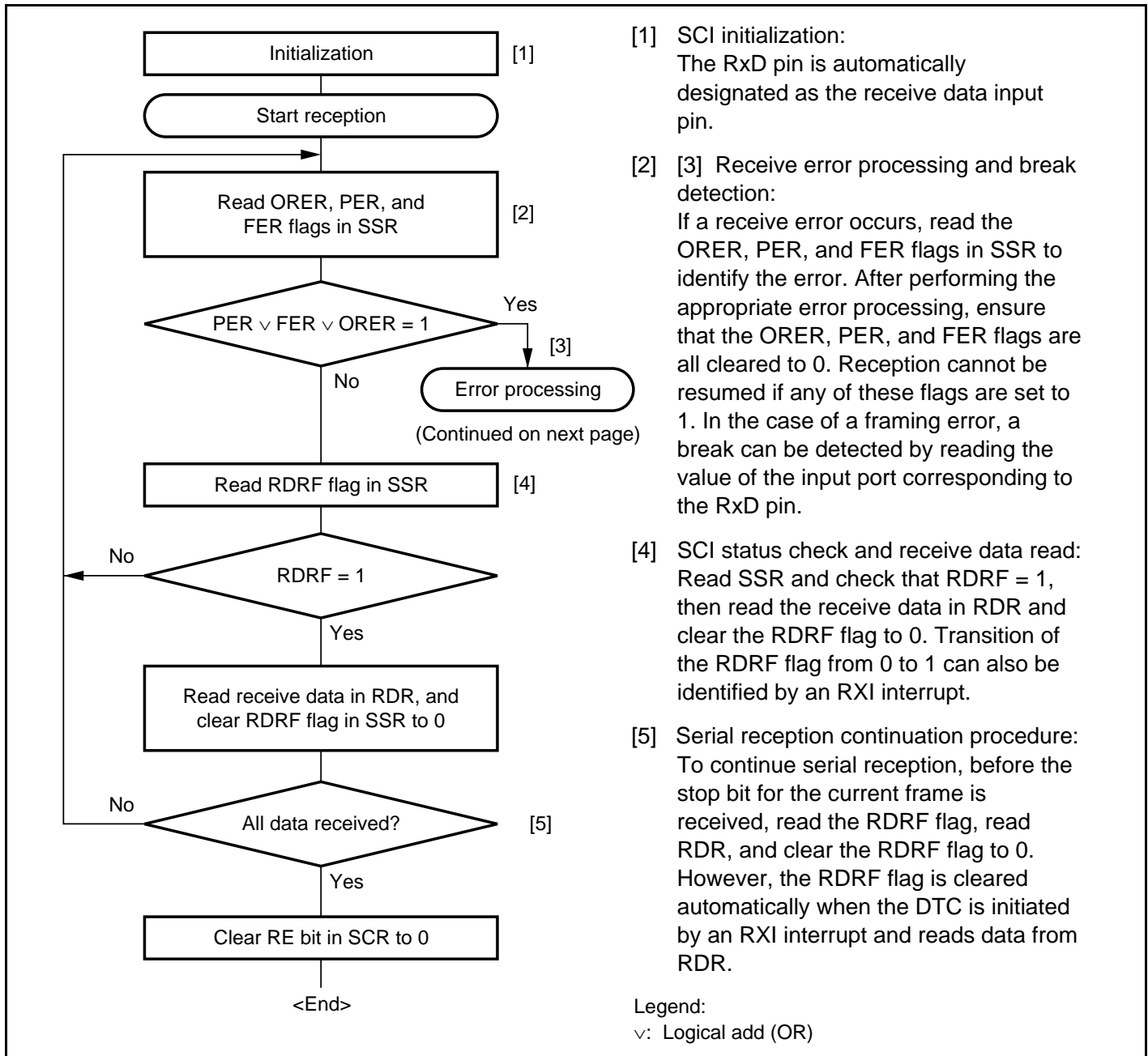
Table 16.11 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 16.12 shows a sample flow chart for serial data reception.

**Table 16.11 SSR Status Flags and Receive Data Handling**

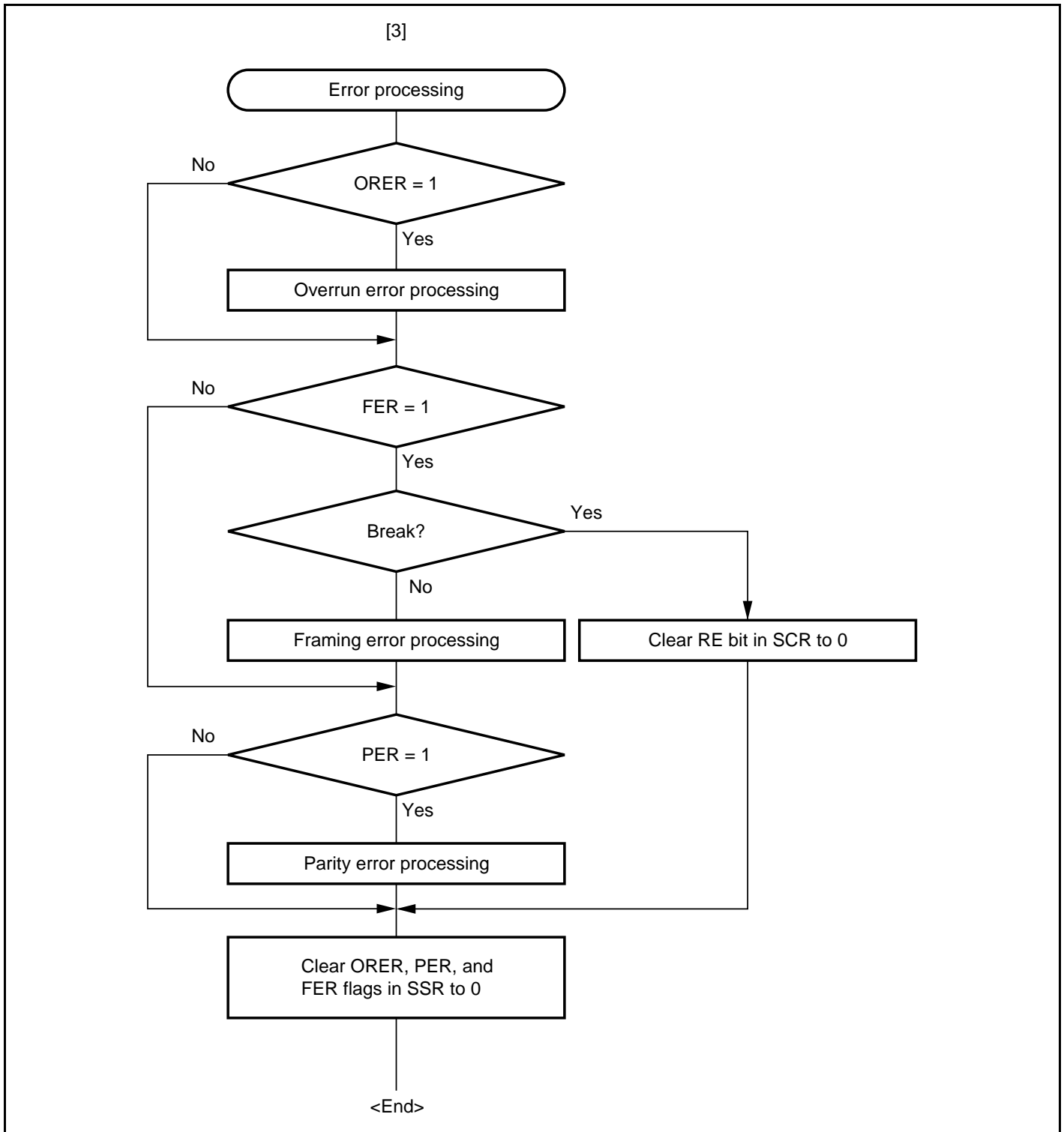
SSR Status Flag				Receive Data	Receive Error Type
RDRF*	ORER	FER	PER		
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

Note: \* The RDRF flag retains the state it had before data reception.





**Figure 16.12 Sample Serial Reception Flowchart (1)**

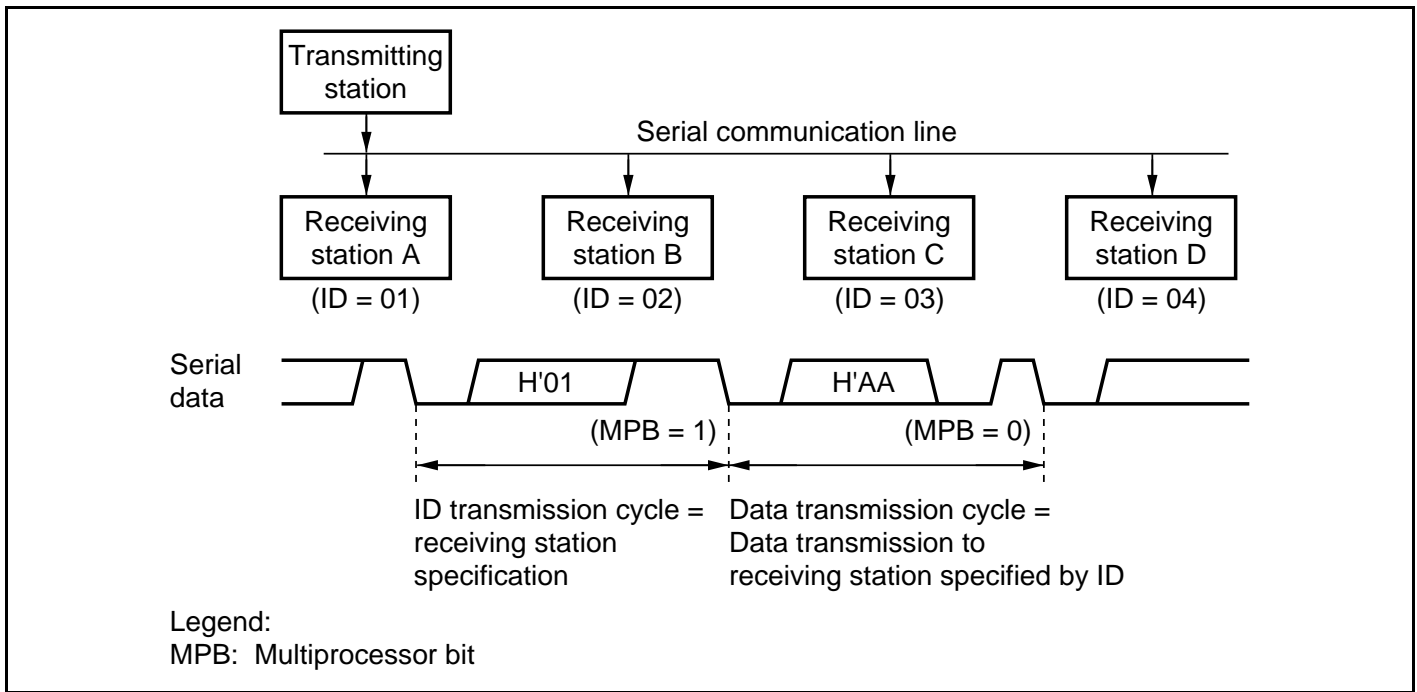
**Figure 16.12 Sample Serial Reception Flowchart (2)**

## 16.5 Multiprocessor Communication Function

Use of the multiprocessor communication function enables data transfer to be performed among a number of processors sharing communication lines by means of asynchronous serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data. When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code. The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle for the specified receiving station. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle. If the multiprocessor bit is 1, the cycle is an ID transmission cycle, and if the multiprocessor bit is 0, the cycle is a data transmission cycle. Figure 16.13 shows an example of inter-processor communication using the multiprocessor format. The transmitting station first sends the ID code of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added. The receiving station skips data until data with a 1 multiprocessor bit is sent. When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip data until data with a 1 multiprocessor bit is again received.

The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set to 1, transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status flags, RDRF, FER, and ORER in SSR to 1 are prohibited until data with a 1 multiprocessor bit is received. On reception of a receive character with a 1 multiprocessor bit, the MPB bit in SSR is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt is generated.

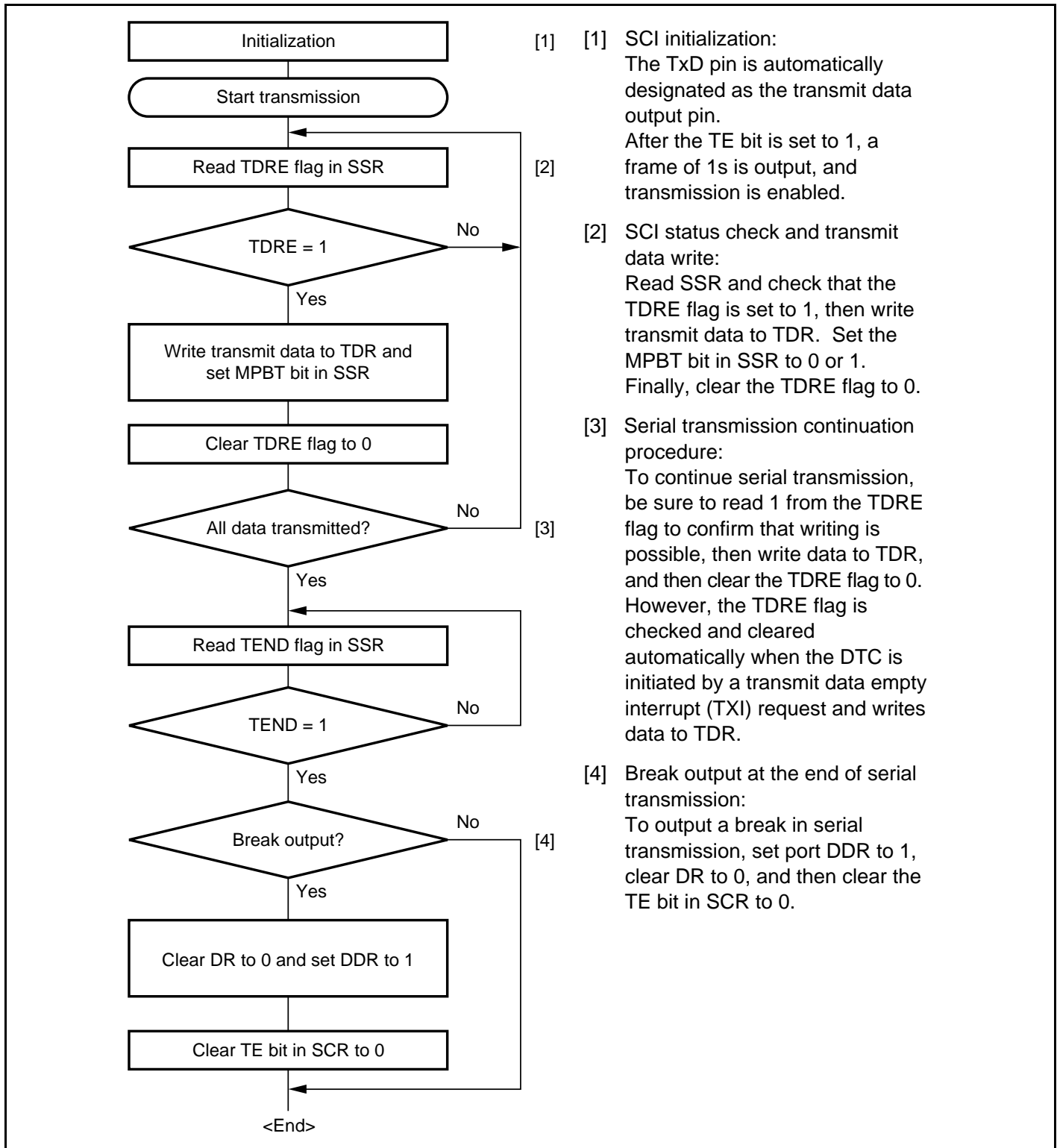
When the multiprocessor format is selected, the parity bit setting is invalid. All other bit settings are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.



**Figure 16.13 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)**

### 16.5.1 Multiprocessor Serial Data Transmission

Figure 16.14 shows a sample flowchart for multiprocessor serial data transmission. For an ID transmission cycle, set the MPBT bit in SSR to 1 before transmission. For a data transmission cycle, clear the MPBT bit in SSR to 0 before transmission. All other SCI operations are the same as those in asynchronous mode.

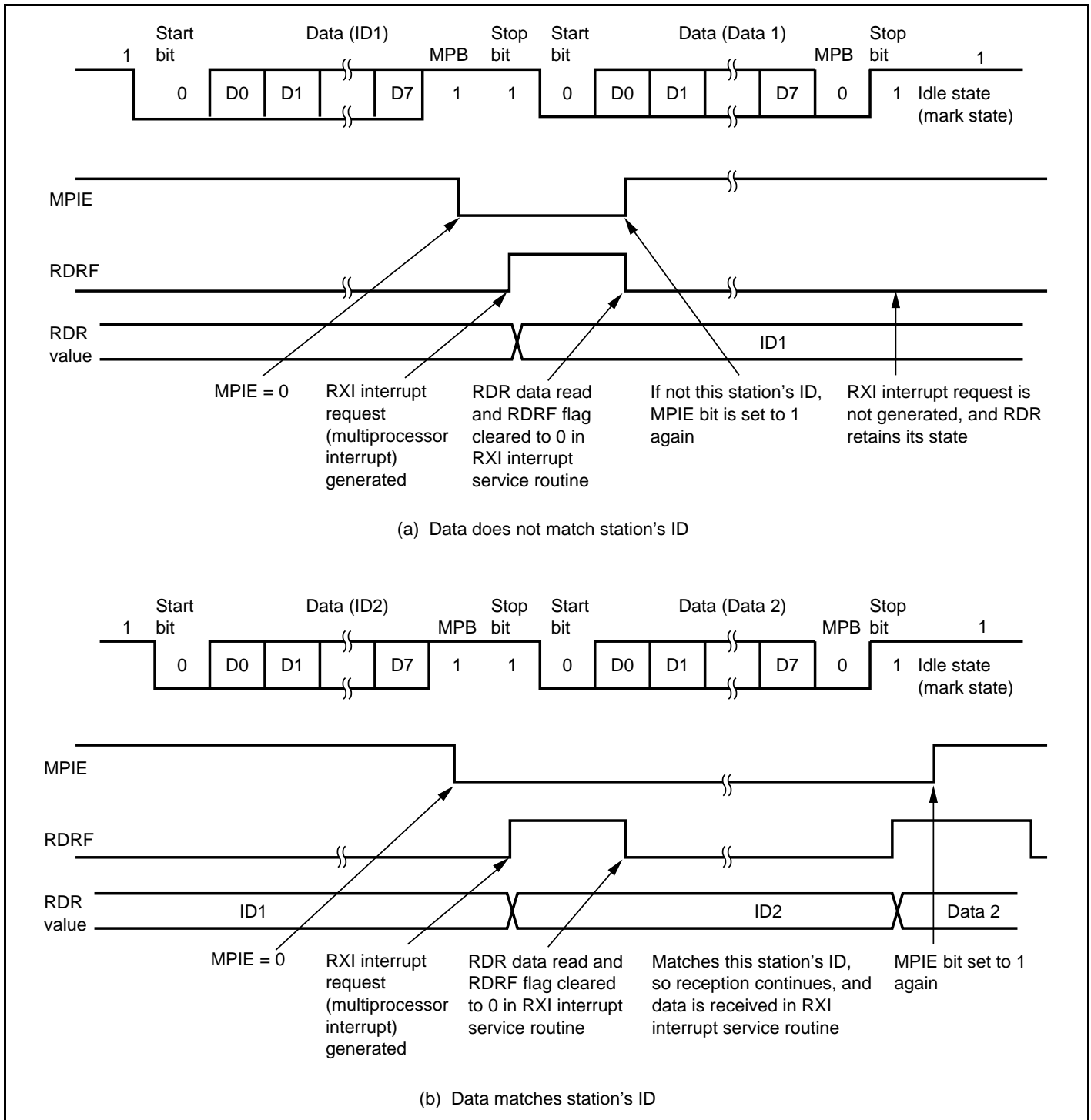


**Figure 16.14 Sample Multiprocessor Serial Transmission Flowchart**

## 16.5.2 Multiprocessor Serial Data Reception

Figure 16.16 shows a sample flowchart for multiprocessor serial data reception. If the MPIE bit in SCR is set to 1, data is skipped until data with a 1 multiprocessor bit is sent. On receiving data

with a 1 multiprocessor bit, the receive data is transferred to RDR. An RXI interrupt request is generated at this time. All other SCI operations are the same as in asynchronous mode. Figure 16.15 shows an example of SCI operation for multiprocessor format reception.



**Figure 16.15 Example of SCI Operation in Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

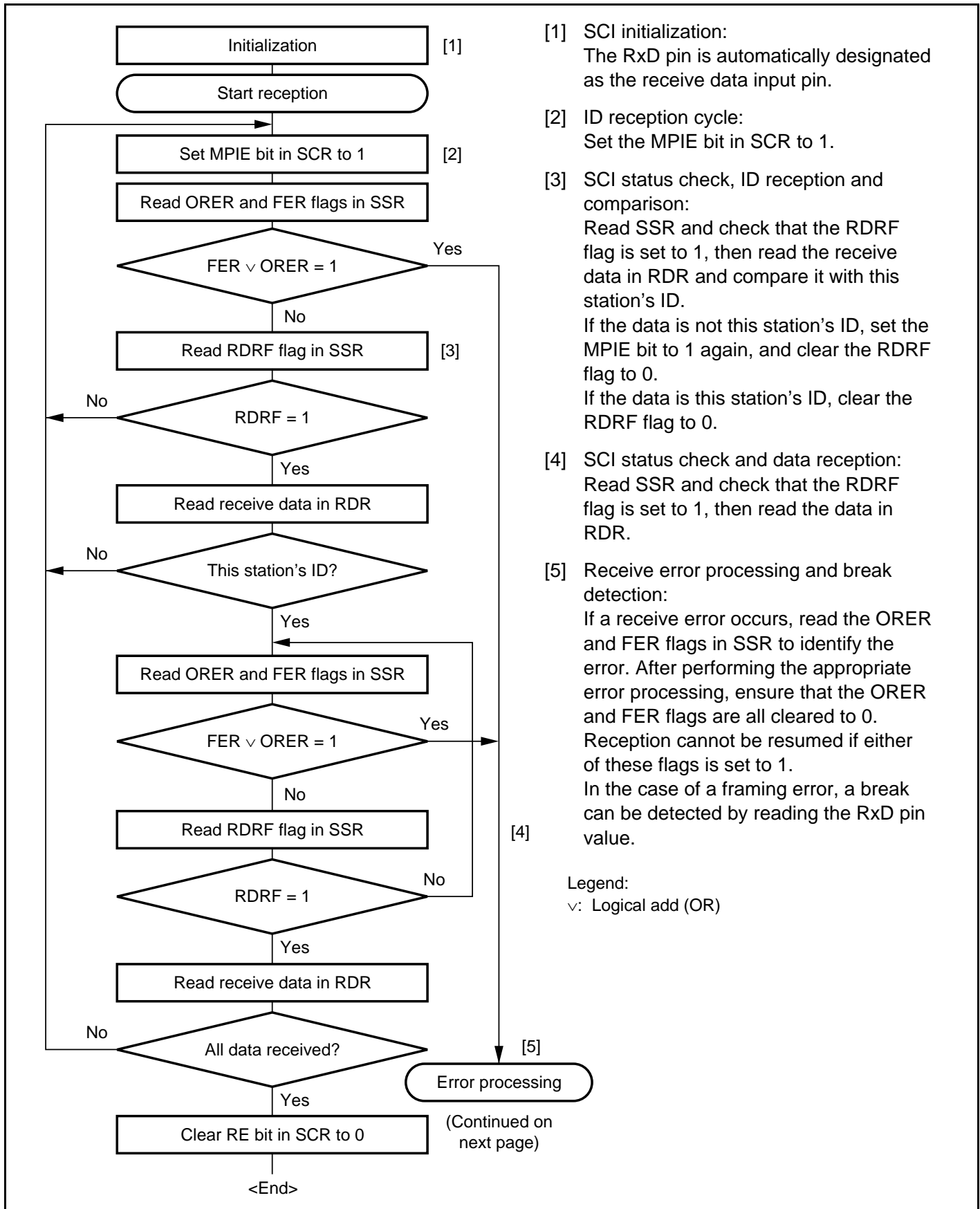
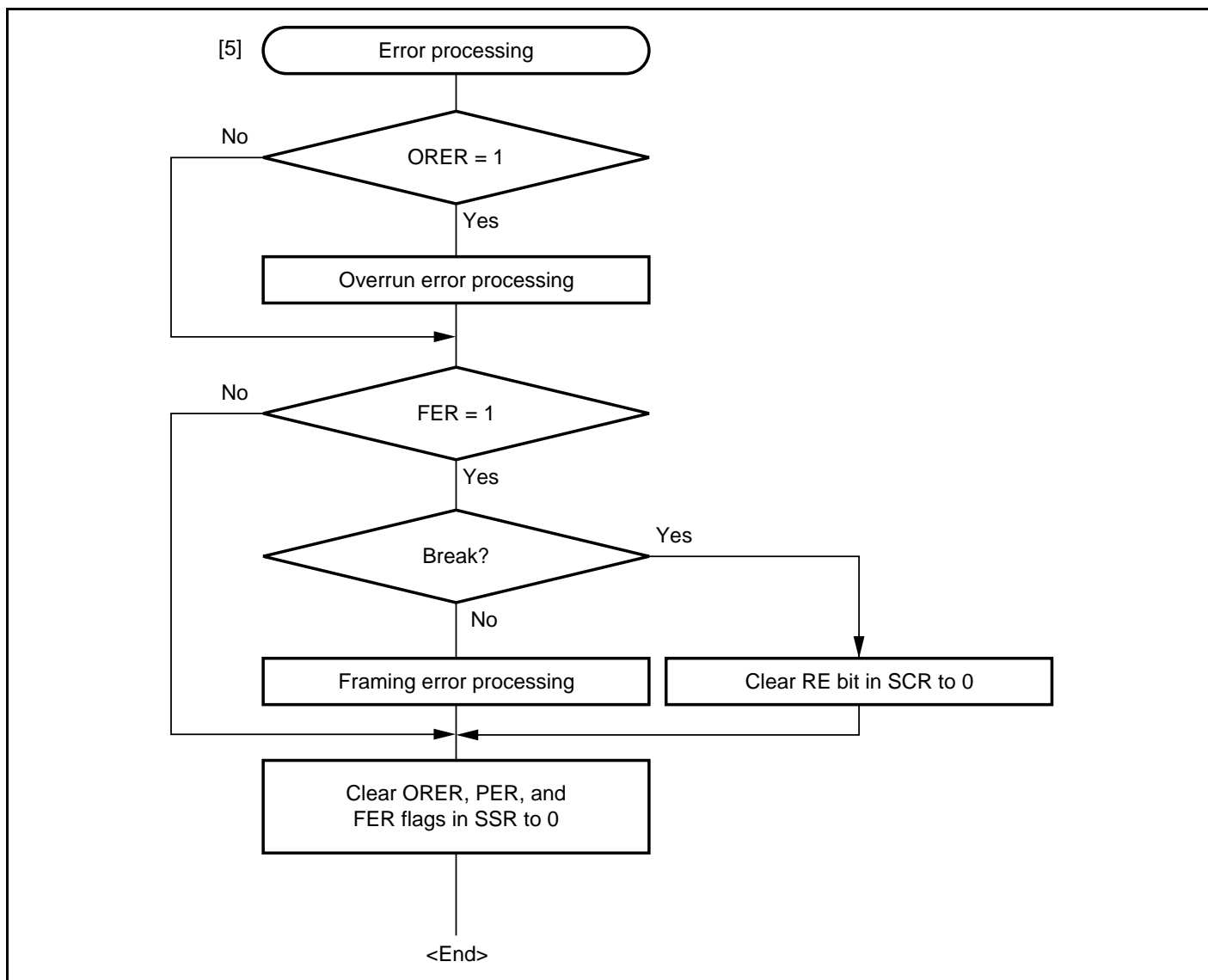


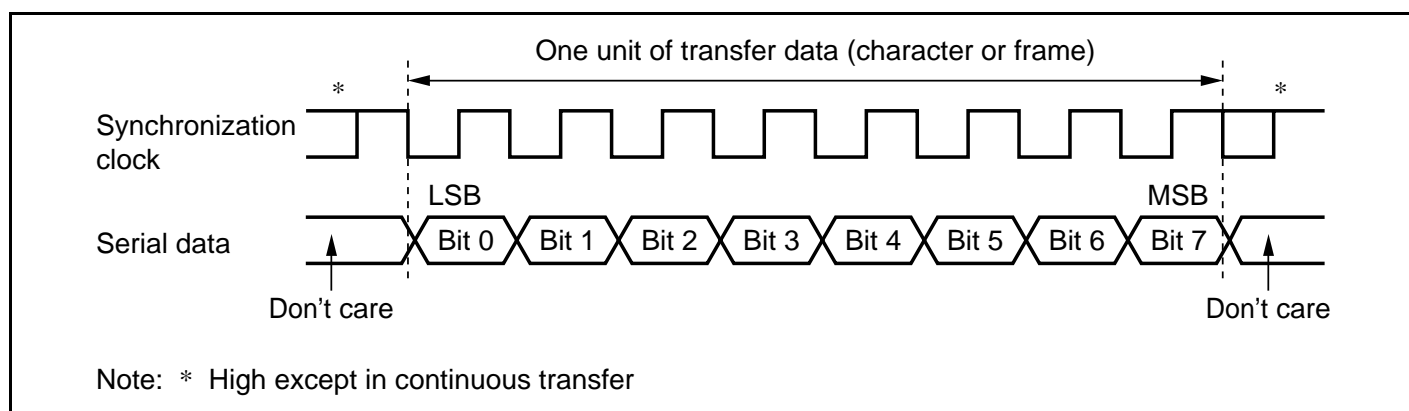
Figure 16.16 Sample Multiprocessor Serial Reception Flowchart (1)

**Figure 16.16 Sample Multiprocessor Serial Reception Flowchart (2)**



## 16.6 Operation in Clocked Synchronous Mode

Figure 16.17 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses. One character in transfer data consists of 8-bit data. In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the MSB state. In clocked synchronous mode, no parity or multiprocessor bit is added. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that the next transmit data can be written during transmission or the previous receive data can be read during reception, enabling continuous data transfer.



**Figure 16.17 Data Format in Synchronous Communication (LSB-First)**

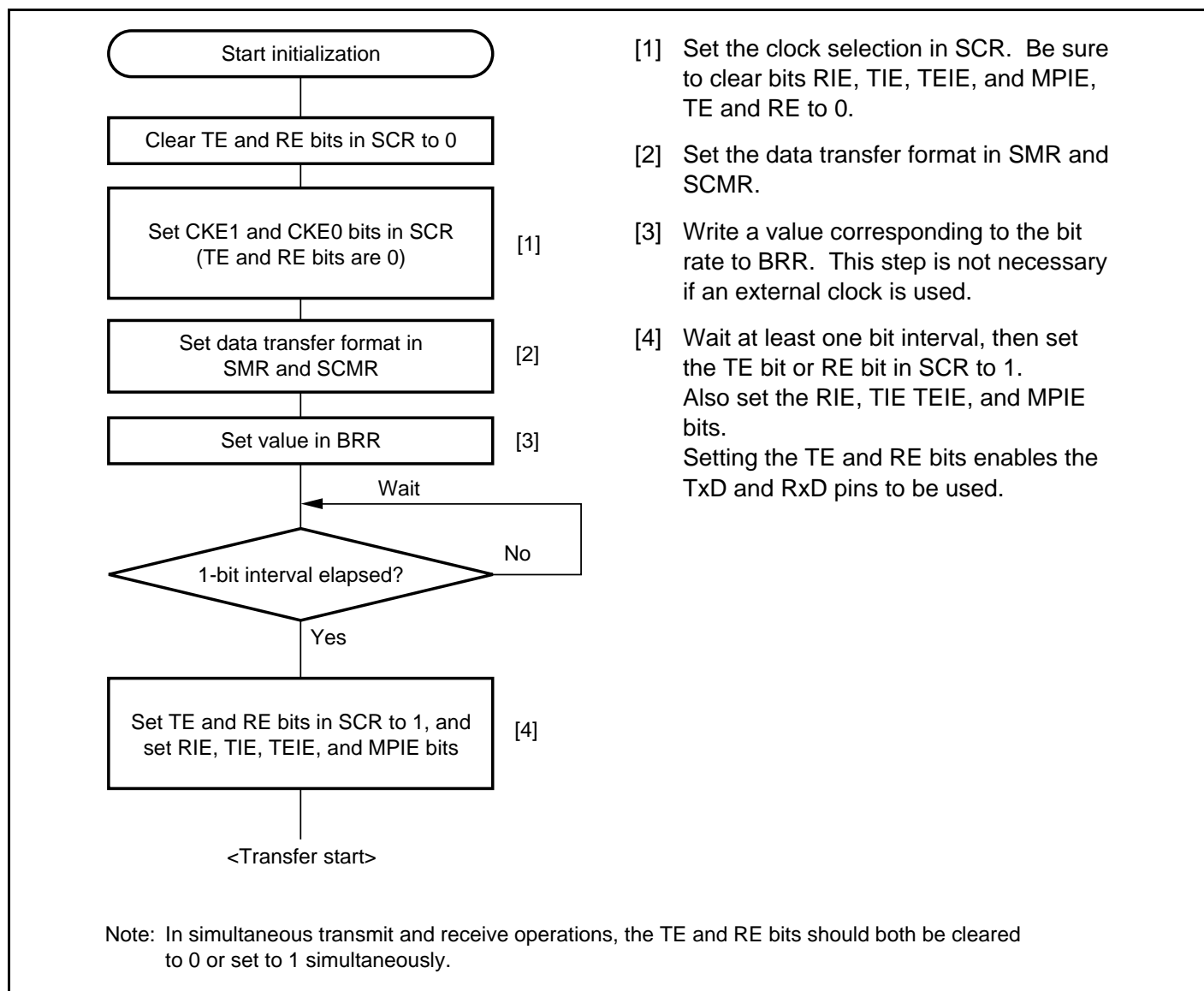
### 16.6.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCR. When the SCI is operated on an internal clock, the synchronization clock is output from the SCK pin. Eight synchronization clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high.

### 16.6.2 SCI Initialization (Synchronous)

Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 16.18. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag in SSR is

set to 1. However, clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags in SSR, or RDR.



**Figure 16.18 Sample SCI Initialization Flowchart**

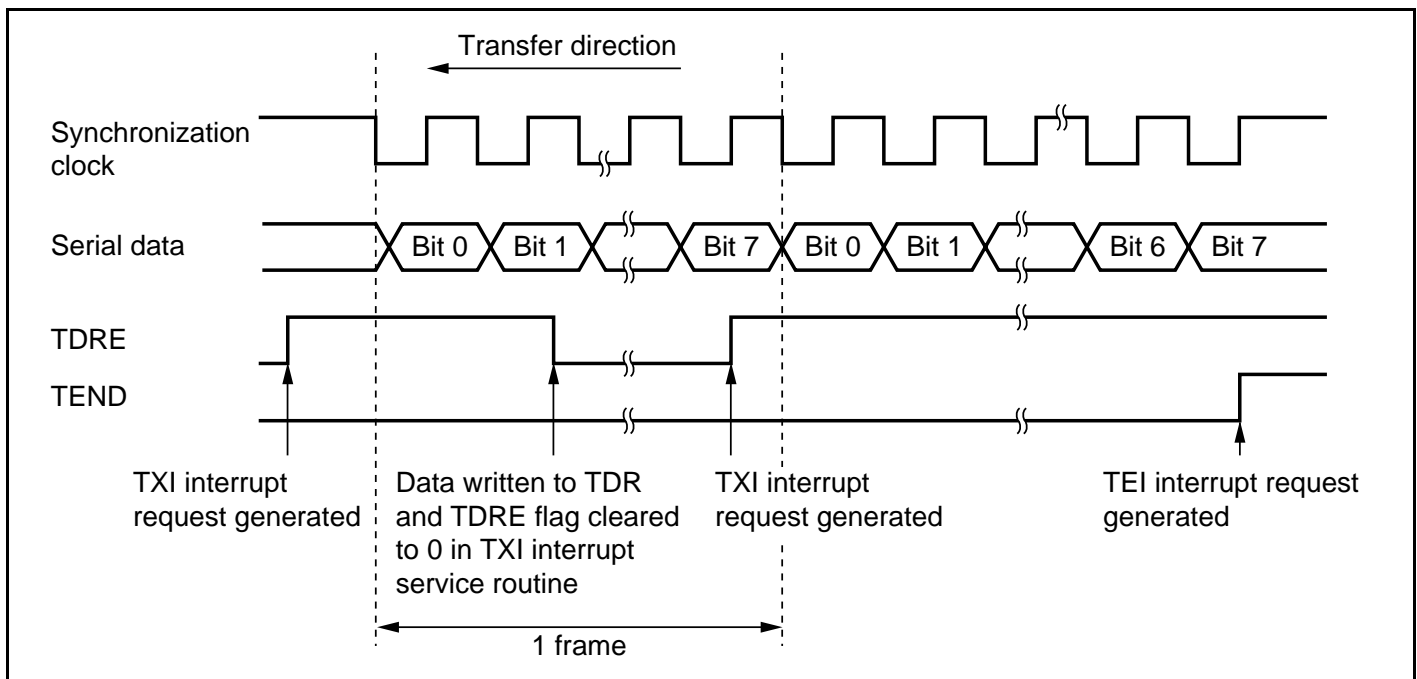
### 16.6.3 Serial Data Transmission (Clocked Synchronous Mode)

Figure 16.19 shows an example of SCI operation for transmission in clocked synchronous mode. In serial transmission, the SCI operates as described below.

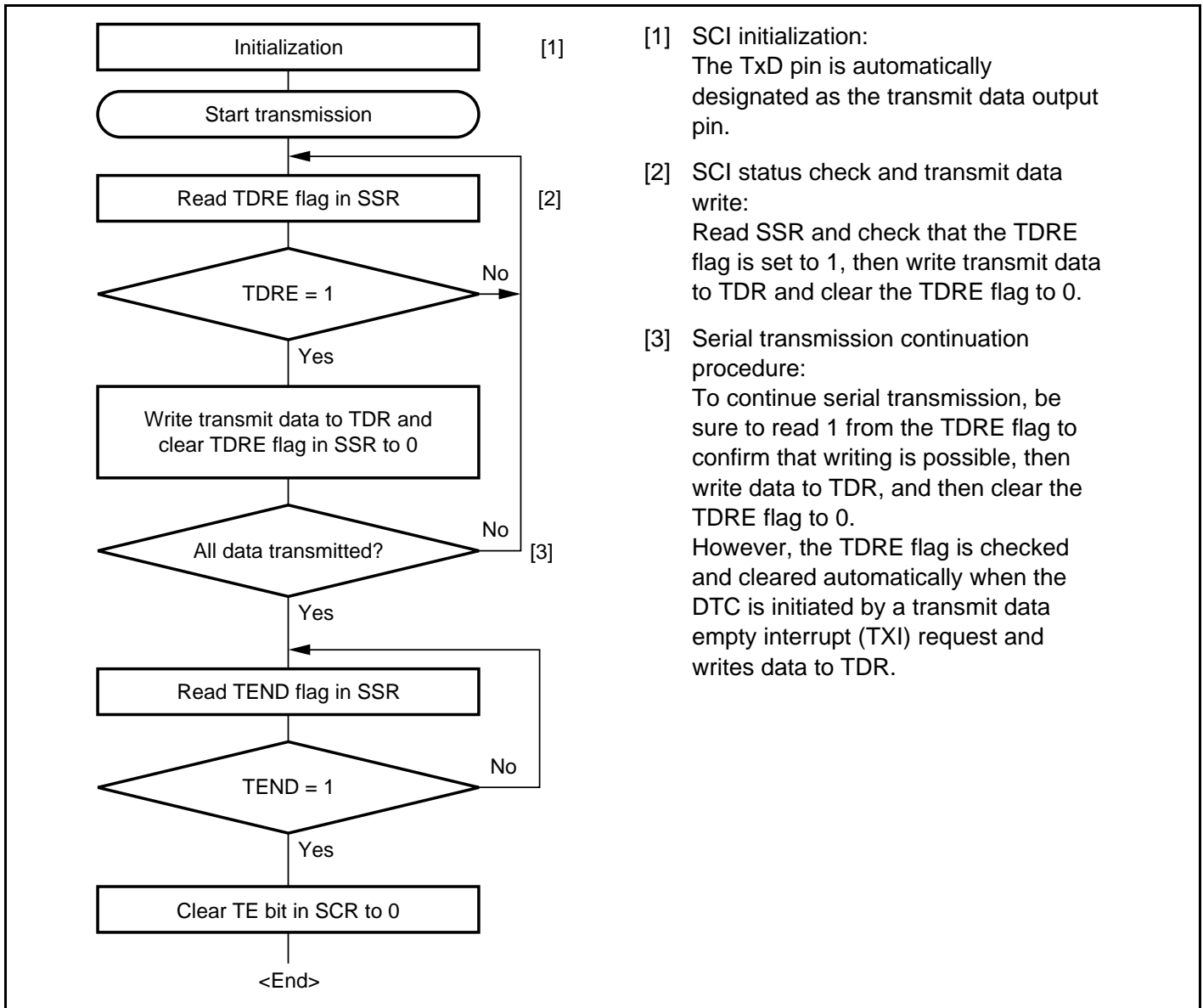
1. The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.

2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. 8-bit data is sent from the TxD pin synchronized with the output clock when output clock 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, data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TxD pin maintains the output state of the last bit. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated. The SCK pin is fixed high.

Figure 16.20 shows a sample flow chart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.



**Figure 16.19 Sample SCI Transmission Operation in Clocked Synchronous Mode**

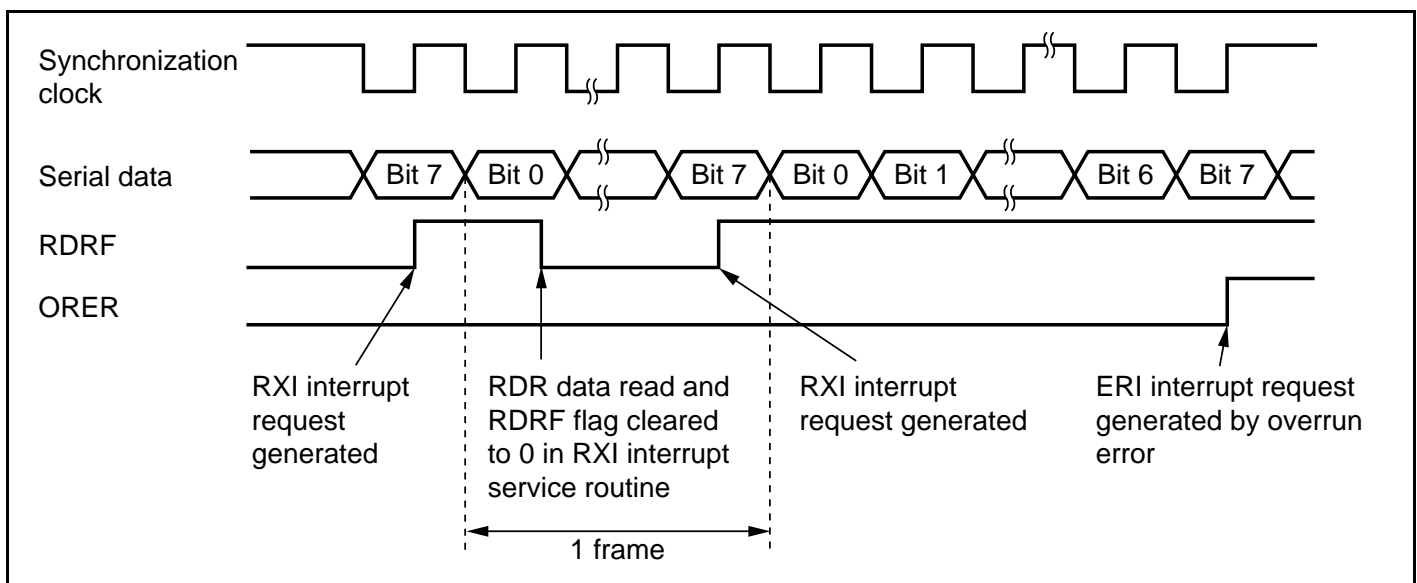


**Figure 16.20 Sample Serial Transmission Flowchart**

### 16.6.4 Serial Data Reception (Clocked Synchronous Mode)

Figure 16.21 shows an example of SCI operation for reception in clocked synchronous mode. In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with a synchronization clock input or output, starts receiving data, and stores the receive data in RSR.
2. If an overrun error (when reception of the next data is completed while the RDRF flag is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.



**Figure 16.21 Example of SCI Receive Operation in Clocked Synchronous Mode**

Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 16.22 shows a sample flowchart for serial data reception.

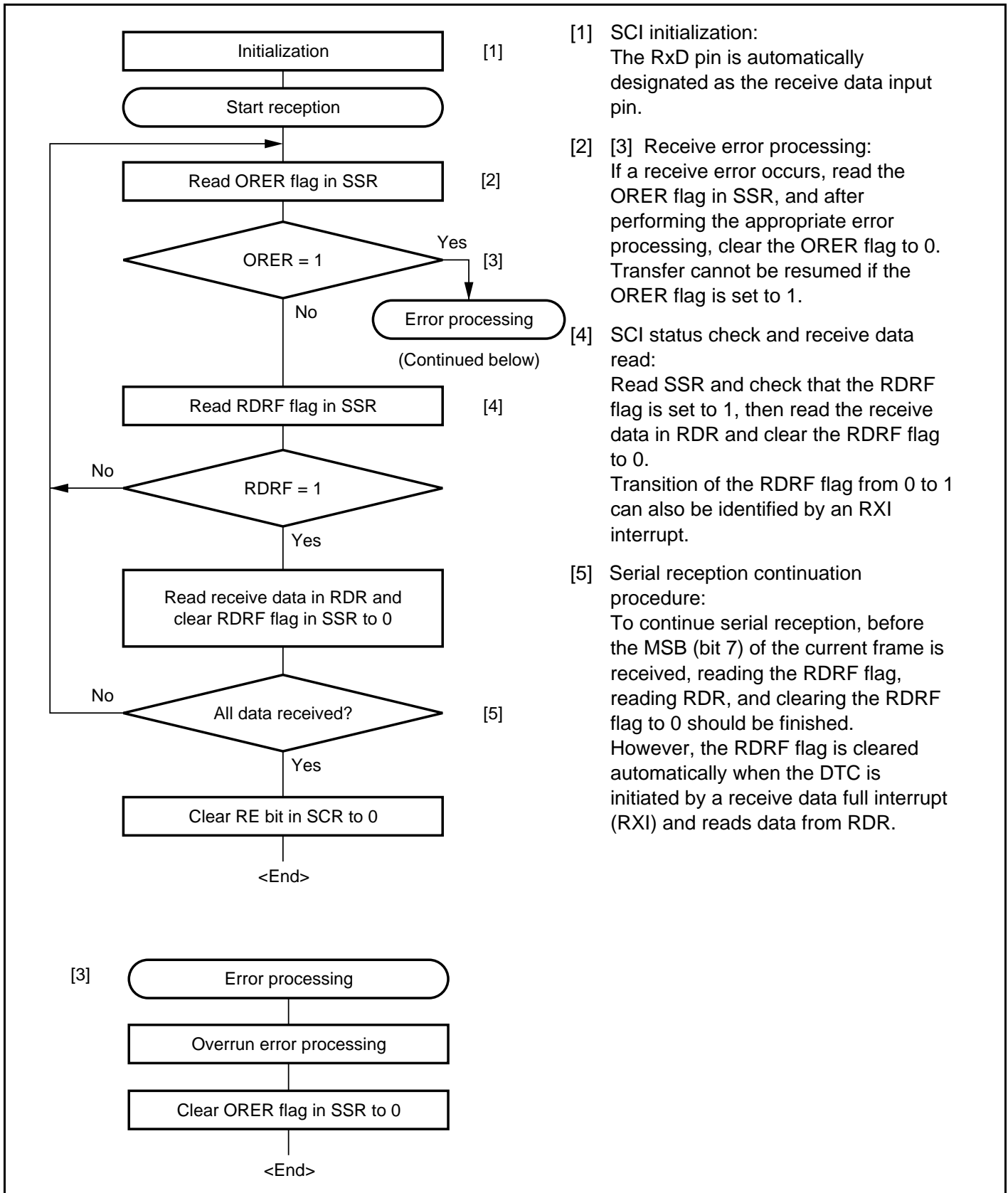


Figure 16.22 Sample Serial Reception Flowchart

### 16.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode)

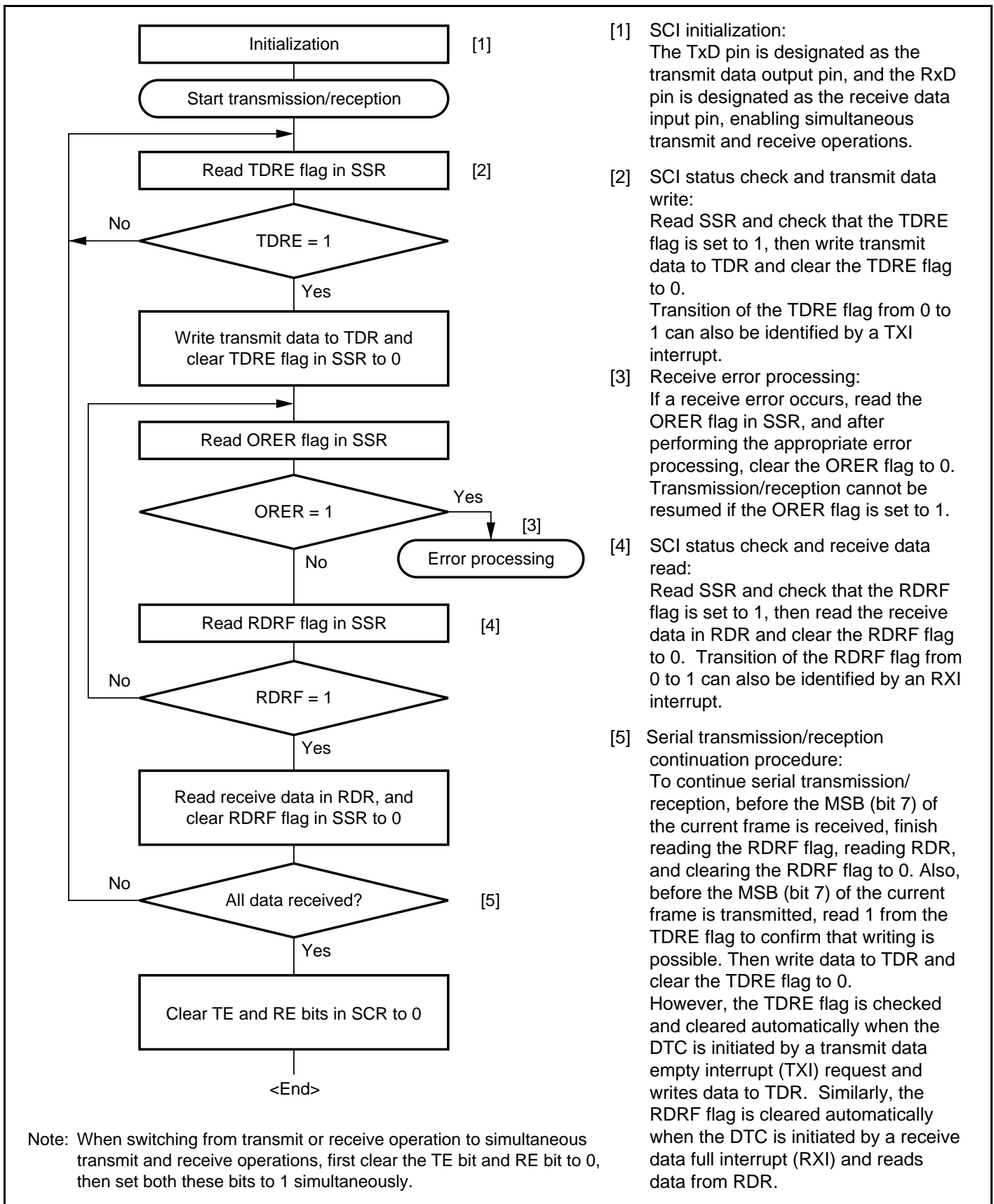
Figure 16.23 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags in SSR are set to 1, clear the TE bit in SCR to 0. Then simultaneously set 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 in SSR and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set the TE and RE bits to 1 with a single instruction.

### 16.6.6 SCI Selection in Serial Enhanced Mode

SCI\_0 and SCI\_2 provides the following capability according to the serial enhanced mode registers (SEMR\_0 and SEMR\_2) settings.

If the SCI is used in clocked synchronous mode with clock input, the SCI channel can be enabled/disabled using the input at the external pins. The external pins include PA0/SSE0I (SCI\_0) and PA1/SSE2I (SCI\_2); therefore, this capability is not available in modes where the PA0 and PA1 pins are automatically set for address output.

When the SCI operation is disabled (not selected) by input at the external pins, TxD output is fixed to the high-impedance state and SCK input is internally fixed to high. One-to-multipoint communication is possible if the master device, which outputs SCK, controls these external pins for chip selection. SCI selection capability is selected using the SSE bits in SEMR.



**Figure 16.23 Sample Flowchart of Simultaneous Serial Transmission and Reception**

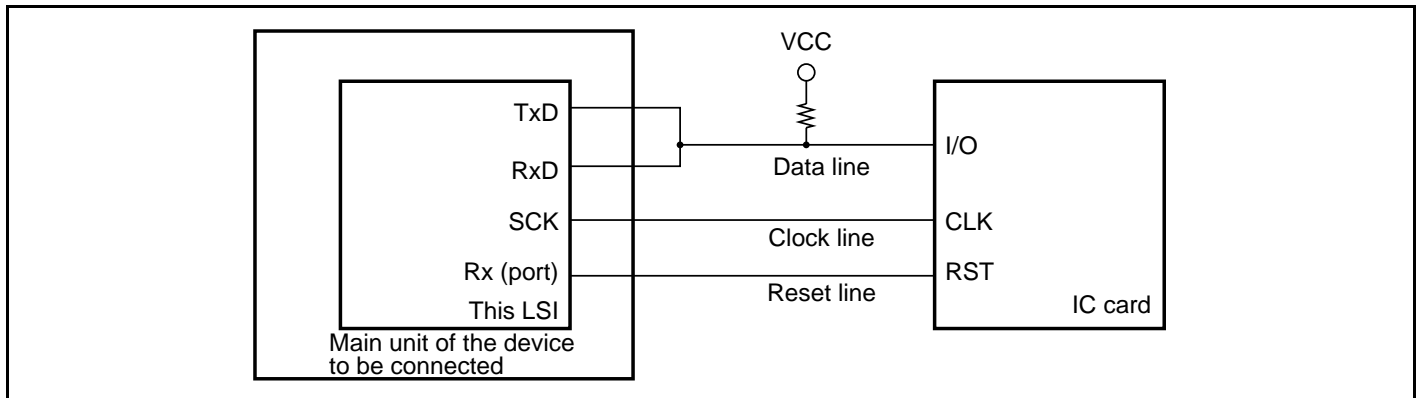


## 16.7 Smart Card Interface Description

The SCI supports the IC card (smart card) interface based on the ISO/IEC 7816-3 (Identification Card) standard as an enhanced serial communication interface function. Smart card interface mode can be selected using the appropriate register.

### 16.7.1 Sample Connection

Figure 16.24 shows a sample connection between the smart card and this LSI. As in the figure, since this LSI communicates with the IC card using a single transmission line, interconnect the TxD and RxD pins and pull up the data transmission line to VCC using a resistor. Setting the RE and TE bits in SCR to 1 with the IC card not connected enables closed transmission/reception allowing self diagnosis. To supply the IC card with the clock pulses generated by the SCI, input the SCK pin output to the CLK pin of the IC card. A reset signal can be supplied via the output port of this LSI.

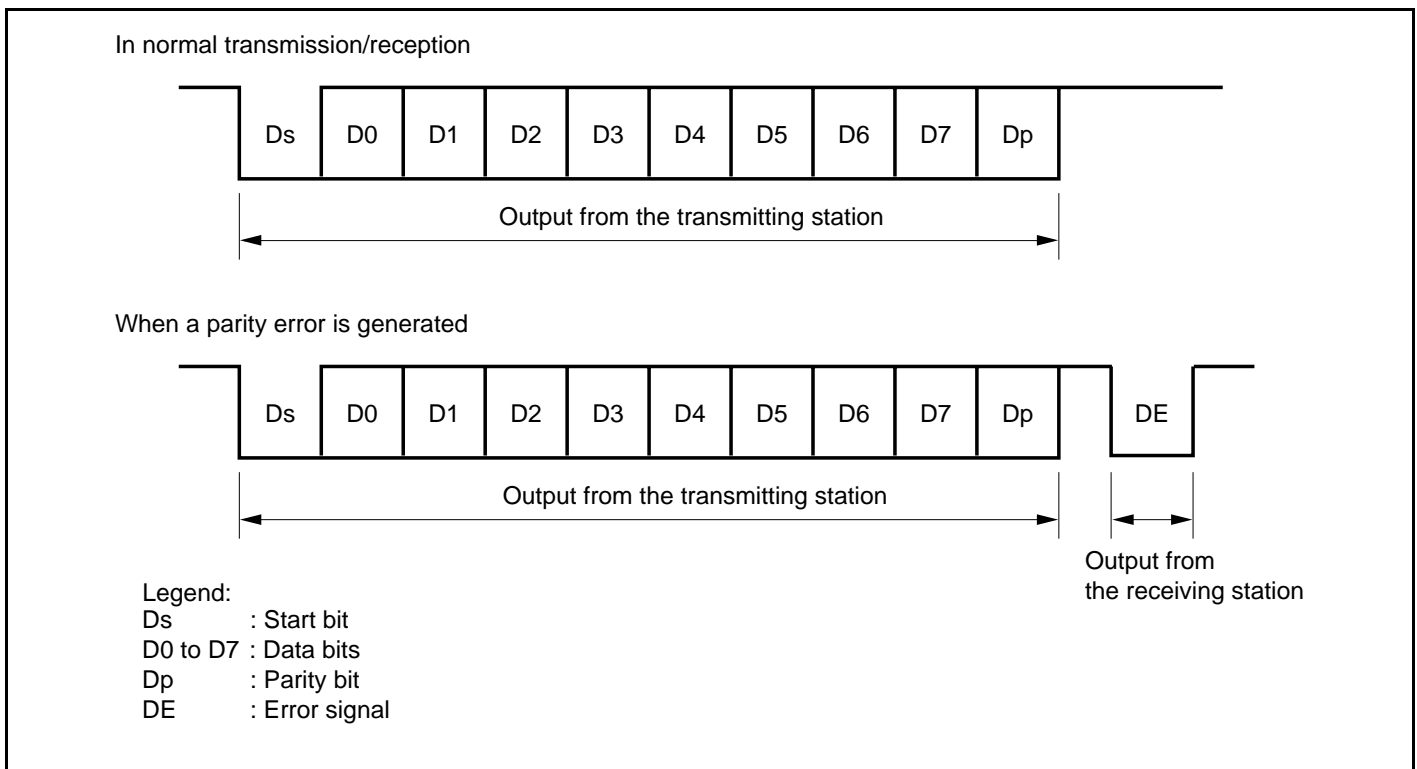


**Figure 16.24 Pin Connection for Smart Card Interface**

### 16.7.2 Data Format (Except in Block Transfer Mode)

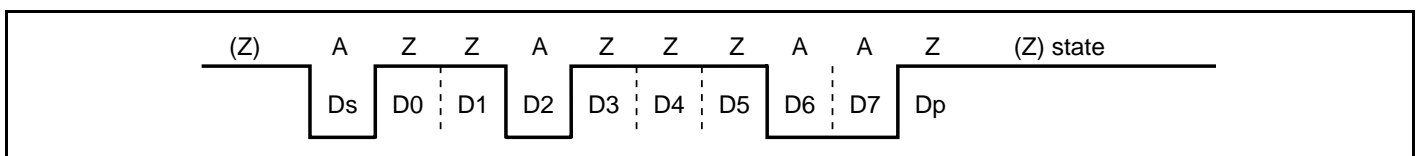
Figure 16.25 shows the data transfer formats in smart card interface mode.

- One frame contains 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etu (elementary time unit: time required for transferring one bit) is secured as a guard time after the end of the parity bit before the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etu has passed from the start bit.
- If an error signal is sampled during transmission, the same data is automatically re-transmitted after two or more etu.



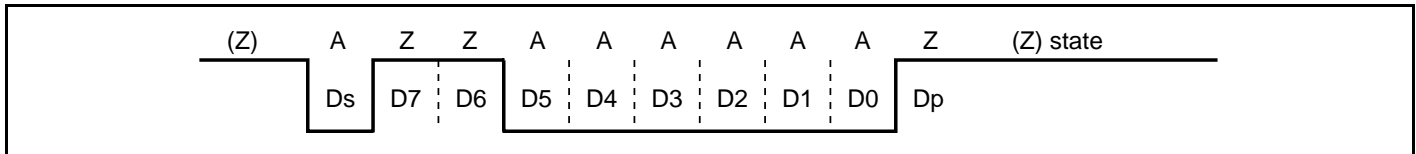
**Figure 16.25 Data Formats in Normal Smart Card Interface Mode**

For communication with the IC cards of the direct convention and inverse convention types, follow the procedure below.



**Figure 16.26 Direct Convention (SDIR = SINV =  $\overline{O/E} = 0$ )**

For the direct convention type, logic levels 1 and 0 correspond to states Z and A, respectively, and data is transferred with LSB-first as the start character, as shown in figure 16.26. Therefore, data in the start character in the figure is H'3B. When using the direct convention type, write 0 to both the SDIR and SINV bits in SCMR. Write 0 to the  $O/\bar{E}$  bit in SMR in order to use even parity, which is prescribed by the smart card standard.



**Figure 16.27 Inverse Convention (SDIR = SINV =  $O/\bar{E}$  = 1)**

For the inverse convention type, logic levels 1 and 0 correspond to states A and Z, respectively and data is transferred with MSB-first as the start character, as shown in figure 16.27. Therefore, data in the start character in the figure is H'3F. When using the inverse convention type, write 1 to both the SDIR and SINV bits in SCMR. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to state Z. Since the SNIV bit of this LSI only inverts data bits D7 to D0, write 1 to the  $O/\bar{E}$  bit in SMR to invert the parity bit in both transmission and reception.

### 16.7.3 Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following respects.

- 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 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 in SSR is set 11.5 etu after transmission start.
- Although the ERS flag in block transfer mode displays the error signal status as in normal smart card interface mode, the flag is always read as 0 because no error signal is transferred.

### 16.7.4 Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the internal baud rate generator can be used as a communication clock in smart card interface mode. In this mode, the SCI can operate using a basic clock with a frequency of 32, 64, 372, or 256 times the bit rate according to the BCP1 and BCP0 settings (the frequency is always 16 times the bit rate in normal asynchronous mode). At reception, the falling edge of the start bit is sampled using the internal basic clock in order to perform internal synchronization. Receive data is sampled at the 16th, 32nd, 186th and 128th

rising edges of the basic clock pulses so that it can be latched at the center of each bit as shown in figure 16.28. The reception margin here is determined by the following formula.

$$M = \left| \left( 0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100 [\%] \quad \dots \text{Formula (1)}$$

M: Reception margin (%)

N: Ratio of bit rate to clock ( $N = 32, 64, 372, 256$ )

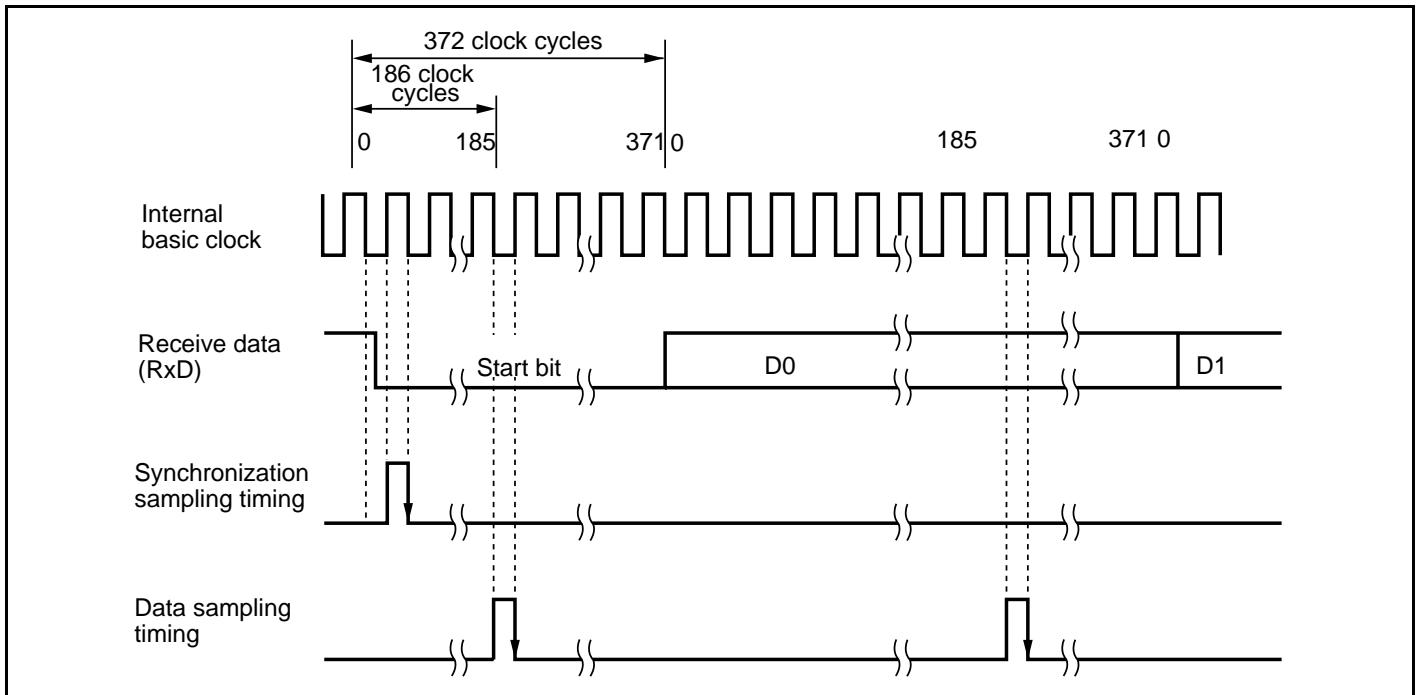
D: Clock duty ( $D = 0$  to  $1.0$ )

L: Frame length ( $L = 10$ )

F: Absolute value of clock rate deviation

Assuming values of  $F = 0$ ,  $D = 0.5$ , and  $N = 372$  in formula (1), the reception margin is determined by the formula below.

$$M = \left( 0.5 - \frac{1}{2 \times 372} \right) \times 100 [\%] = 49.866\%$$



**Figure 16.28 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency Is 372 Times the Bit Rate)**

### 16.7.5 Initialization

Before starting transmitting and receiving data, initialize the SCI using the following procedure. Initialization is also necessary before switching from transmission to reception and vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Clear the error flags ORER, ERS, and PER in SSR to 0.
3. Set the GM, BLK,  $O/\bar{E}$ , BCP1, BCP0, CKS1, and CKS0 bits in SMR appropriately. Also set the PE bit to 1.
4. Set the SMIF, SDIR, and SINV bits in SCMR appropriately. When the SMIF bit is set to 1, the TxD and RxD pins are changed from port pins to SCI pins, placing the pins into high impedance state.
5. Set the value corresponding to the bit rate in BRR.
6. 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.
7. Set the TIE, RIE, TE, and RE bits in SCR appropriately after waiting for at least 1 bit interval. Do not set the TE and RE bits to 1 simultaneously except for self diagnosis.

To switch from reception to transmission, first verify that reception has completed, and 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 flag or PER and ORER flags. To switch from transmission to reception, first verify that transmission has completed, and initialize the SCI. At the end of initialization, TE and RE should be set to 0 and 1, respectively. Transmission completion can be verified by reading the TEND flag.

### 16.7.6 Serial Data Transmission (Except in Block Transfer Mode)

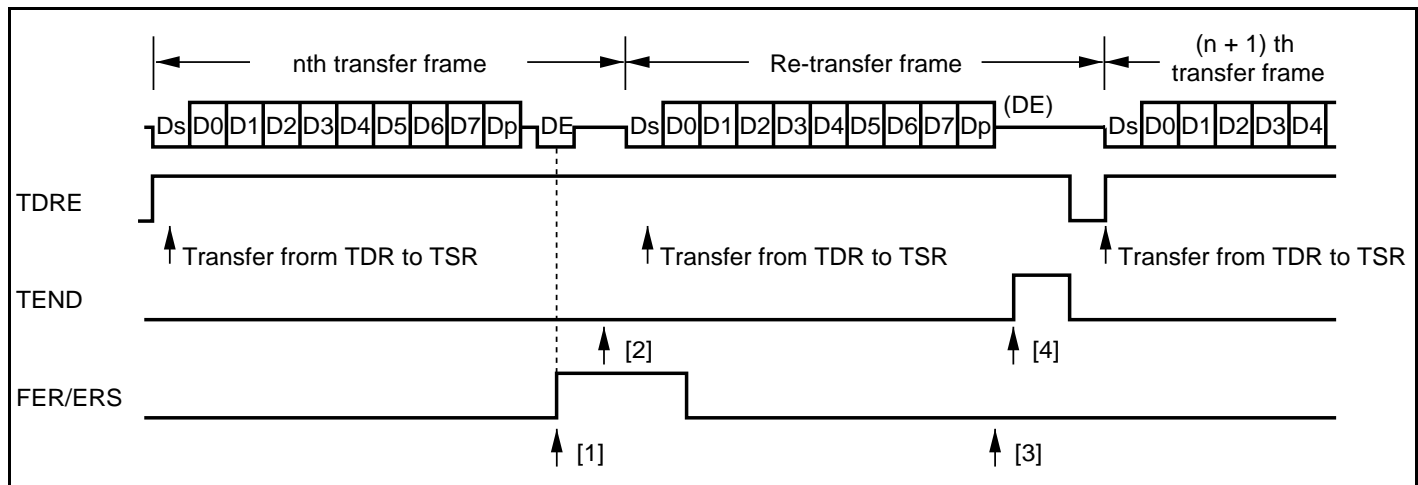
Data transmission in smart card interface mode (except in block transfer mode) is different from that in normal serial communication interface mode in that an error signal is sampled and data is re-transmitted. Figure 16.29 shows the data re-transfer operation during transmission.

1. If an error signal from the receiving end is sampled after one frame of data has been transmitted, the ERS bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the ERS bit to 0 before the next parity bit is sampled.
2. For the frame in which an error signal is received, the TEND bit in SSR is not set to 1. Data is re-transferred from TDR to TSR allowing automatic data retransmission.
3. If no error signal is returned from the receiving end, the ERS bit in SSR is not set to 1. In this case, one frame of data is determined to have been transmitted including re-transfer, and the TEND bit in SSR is set to 1. Here, a TXI interrupt request is generated if the TIE bit in SCR is set to 1. Writing transmit data to TDR starts transmission of the next data.

Figure 16.31 shows a sample flowchart for transmission. All the processing steps are automatically performed using a TXI interrupt request to activate the DTC. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request when TIE in SCR is set. This activates the DTC by a TXI request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DTC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DTC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, TEND remains as 0, thus not activating the DTC. Therefore, the SCI and DTC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit to 1 to enable an ERI interrupt request to be generated at error occurrence.

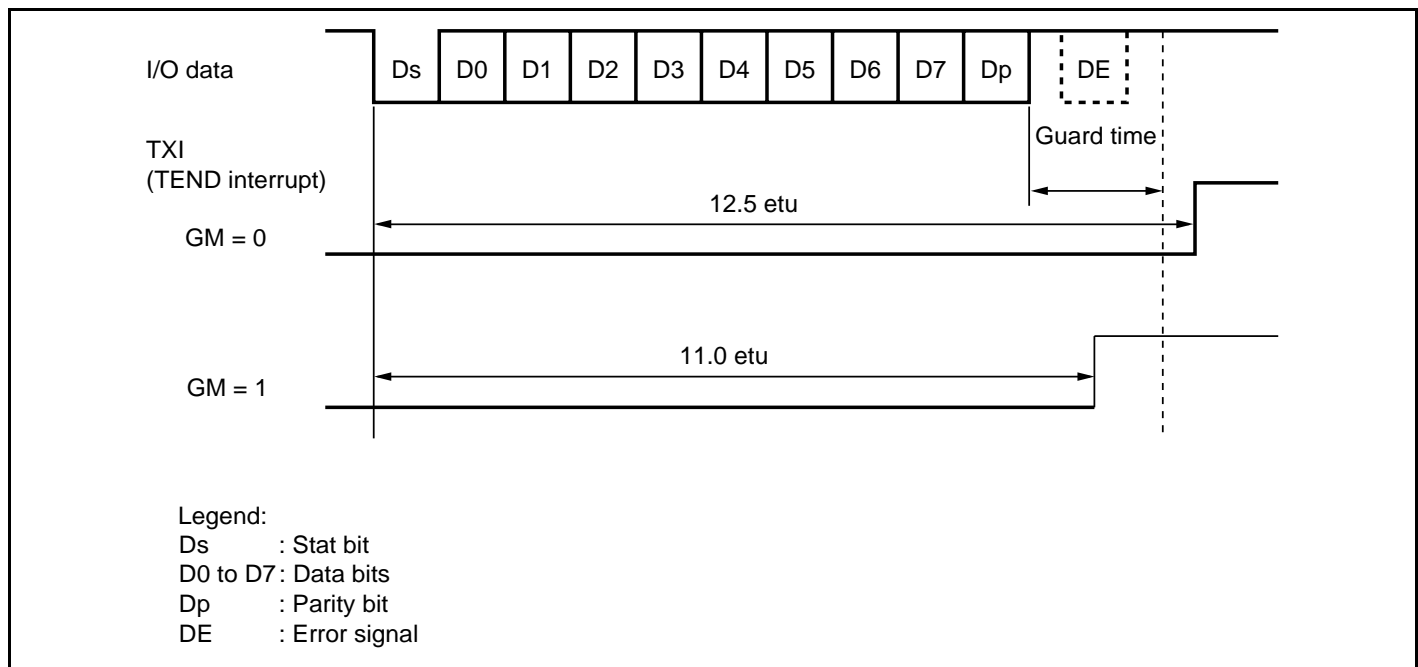
The above procedure also applies to the case in which RFU is activated by TEND in SCI\_0 and SCI\_2.

When transmitting/receiving data using the DTC or RFU, be sure to set and enable them prior to making SCI settings. For DTC and RFU settings, see section 7, Data Transfer Controller (DTC) and section 8, RAM-FIFO Unit (RFU).

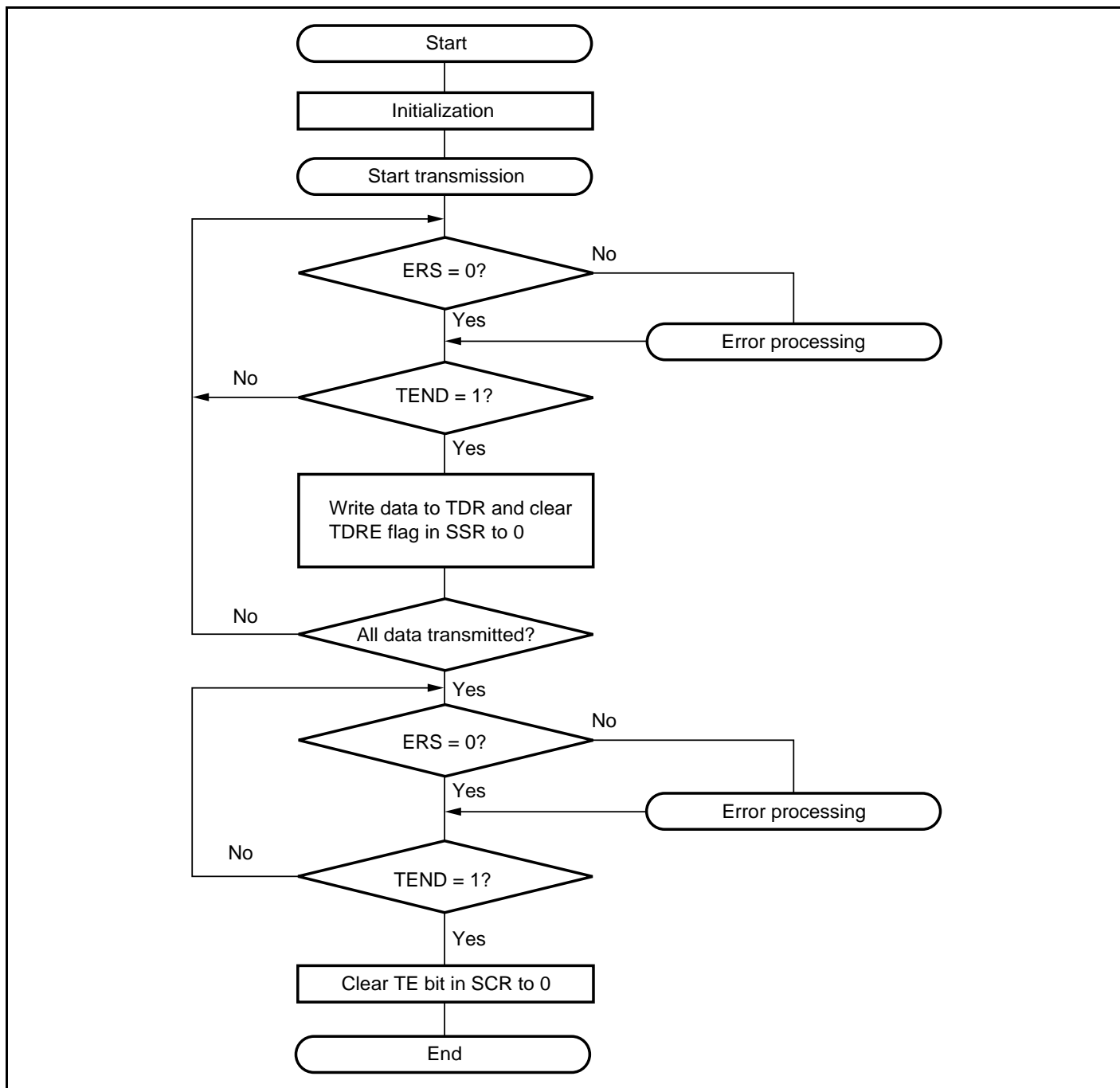


**Figure 16.29 Data Re-transfer Operation in SCI Transmission Mode**

Note that the TEND flag is set in different timings depending on the GM bit setting in SMR, which is shown in figure 16.30.



**Figure 16.30 TEND Flag Set Timings during Transmission**



**Figure 16.31 Sample Transmission Flowchart**



### 16.7.7 Serial Data Reception (Except in Block Transfer Mode)

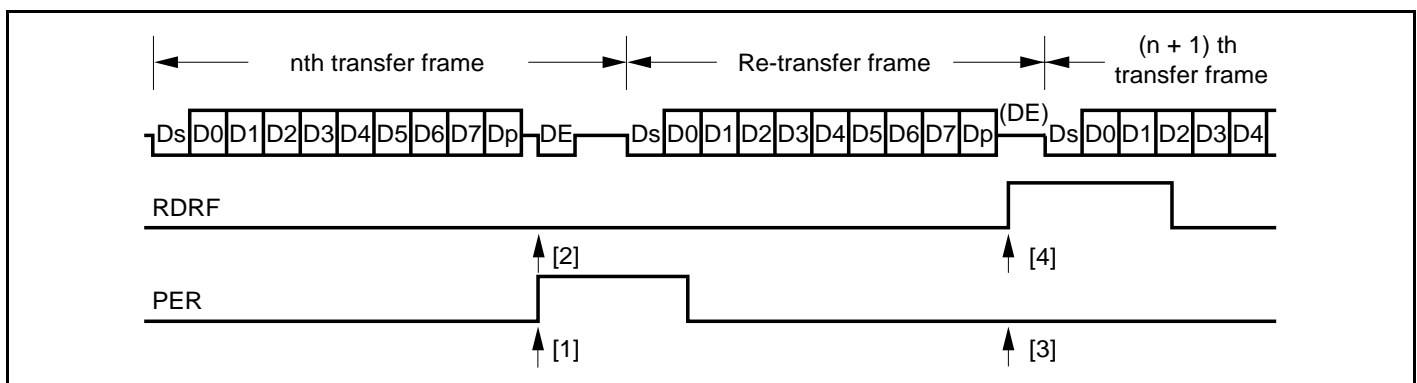
Data reception in smart card interface mode is identical to that in normal serial communication interface mode. Figure 16.32 shows the data re-transfer operation during reception.

1. If a parity error is detected in receive data, the PER bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the PER bit to 0 before the next parity bit is sampled.
2. For the frame in which a parity error is detected, the RDRF bit in SSR is not set to 1.
3. If no parity error is detected, the PER bit in SSR is not set to 1. In this case, data is determined to have been received successfully, and the RDRF bit in SSR is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set.

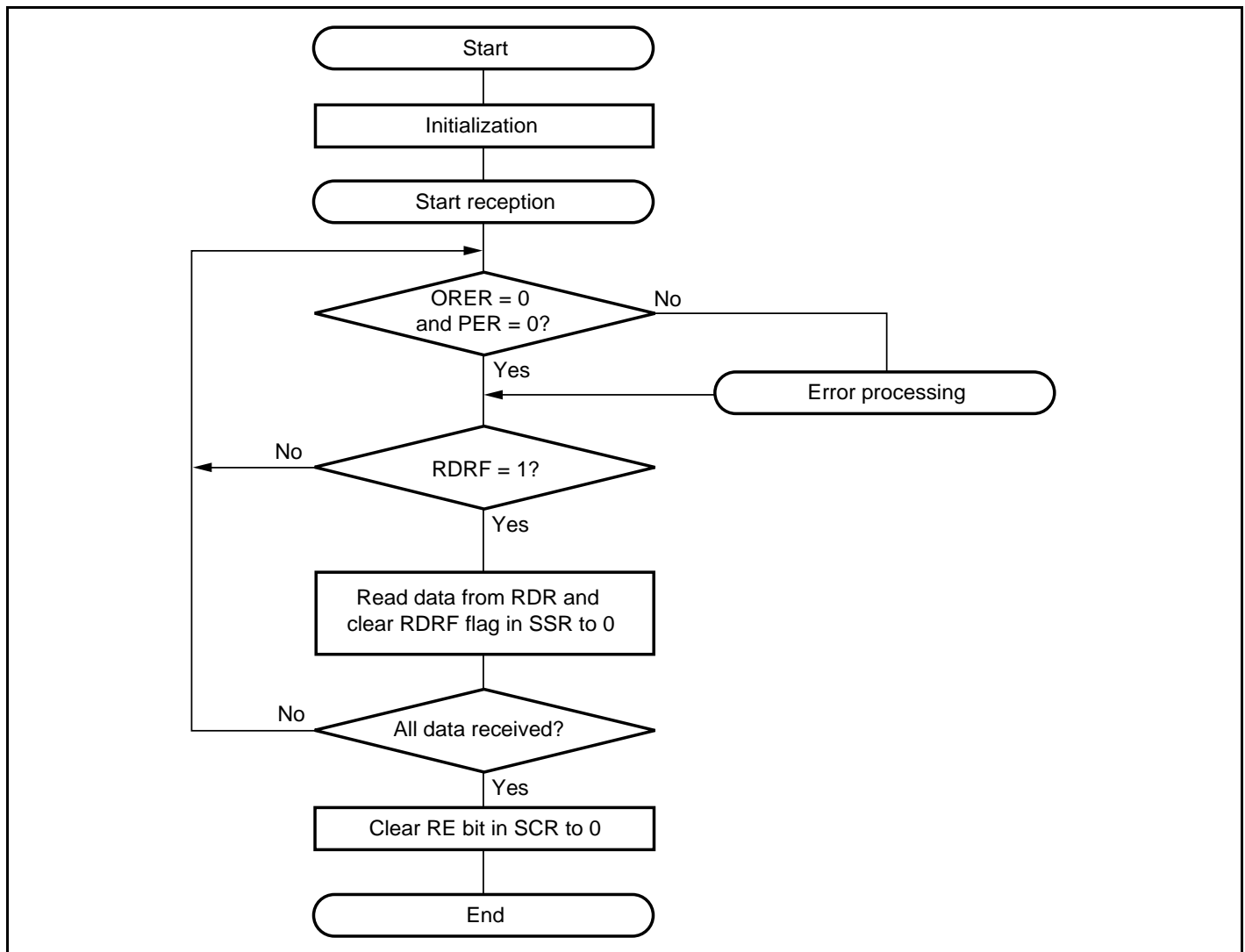
Figure 16.33 shows a sample flowchart for reception. All the processing steps are automatically performed using an RXI interrupt request to activate the DTC. In reception, setting the RIE bit to 1 allows an RXI interrupt request to be generated when the RDRF flag is set to 1. This activates DTC by an RXI request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DTC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by DTC. If an error occurs during reception, i.e., either the ORER or PER flag is set to 1, a transmit/receive error interrupt (ERI) request is generated and the error flag must be cleared. If an error occurs, DTC is not activated and receive data is skipped, therefore, the number of bytes of receive data specified in DTC are transferred. Even if a parity error occurs and PER is set to 1 in reception, receive data is transferred to RDR, thus allowing the data to be read.

The above flow also applies to the case in which RFU is activated by RDRF in SCI\_0 and SCI\_2.

Note: For operations in block transfer mode, see section 16.4, Operation in Asynchronous Mode.



**Figure 16.32 Data Re-transfer Operation in SCI Reception Mode**

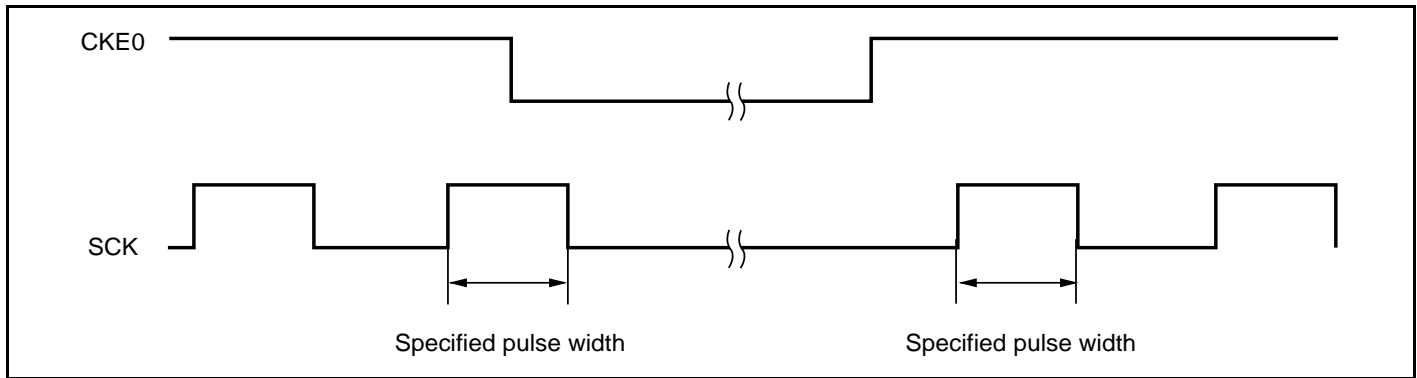


**Figure 16.33 Sample Reception Flowchart**

### 16.7.8 Clock Output Control

Clock output can be fixed using the CKE1 and CKE0 bits in SCR when the GM bit in SMR is set to 1. Specifically, the minimum width of a clock pulse can be specified.

Figure 16.34 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 1.



**Figure 16.34 Clock Output Fixing Timing**

At power-on and transitions to/from software standby mode, use the following procedure to secure the appropriate clock duty ratio.

#### **At Power-On:**

To secure the appropriate clock duty ratio simultaneously with power-on, use the following procedure.

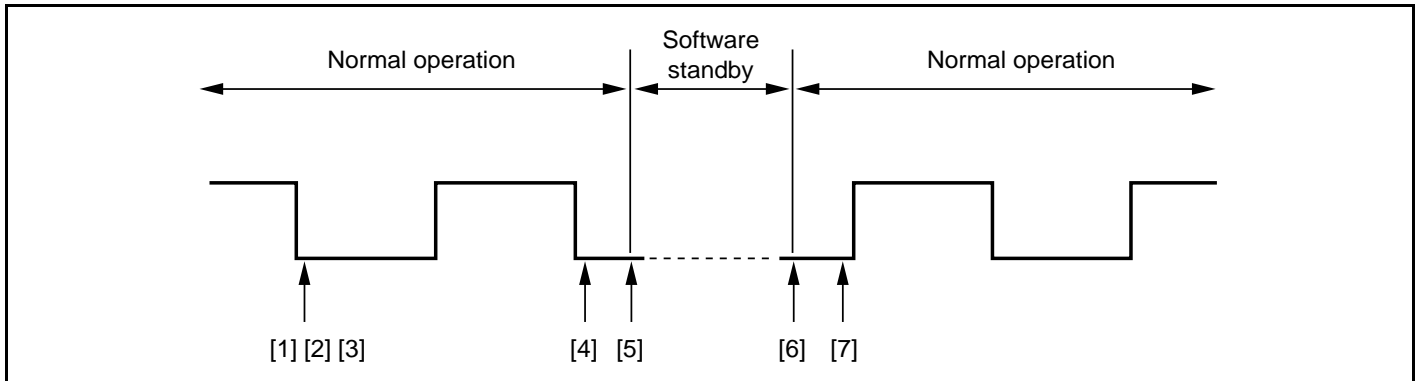
1. Initially, port input is enabled in the high-impedance state. To fix the potential level, use a pull-up or pull-down resistor.
2. Fix the SCK pin to the specified output using the CKE1 bit in SCR.
3. Set SMR and SCMR to enable smart card interface mode.
4. Set the CKE0 bit in SCR to 1 to start clock output.

#### **At Transition from Smart Card Interface Mode to Software Standby Mode:**

1. Set the port data register (DR) and data direction register (DDR) corresponding to the SCK pins to the values for the output fixed state in software standby mode.
2. Write 0 to the TE and RE bits in SCR to stop transmission/reception. Simultaneously, set the CKE1 bit to the value for the output fixed state in software standby mode.
3. Write 0 to the CKE0 bit in SCR to stop the clock.
4. Wait for one cycle of the serial clock. In the mean time, the clock output is fixed to the specified level with the duty ratio retained.
5. Make the transition to software standby mode.

#### **At Transition from Software Standby Mode to Smart Card Interface Mode:**

1. Cancel software standby mode.
2. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty ratio is then generated.



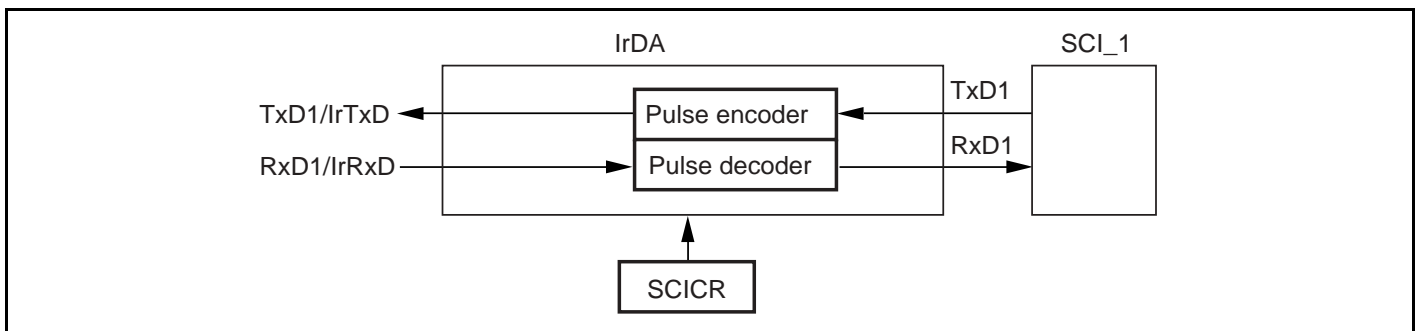
**Figure 16.35 Clock Stop and Restart Procedure**

## 16.8 IrDA Operation

IrDA operation can be used with SCI\_1. Figure 16.36 shows an IrDA block diagram.

If the IrDA function is enabled using the IrE bit in SCICR, the TxD1 and RxD1 pins in SCI\_1 are allowed to encode and decode the waveform based on the IrDA standard version 1.0 (function as the IrTxD and IrRxD pins). Connecting these pins to the infrared data transceiver achieves infrared data communication based on the system defined by the IrDA standard version 1.0.

In the system defined by the IrDA standard version 1.0, communication is started at a transfer rate of 9600 bps, which can be modified as required. The IrDA interface provided by this LSI does not incorporate the capability of automatic modification of the transfer rate; the transfer rate must be modified through programming.



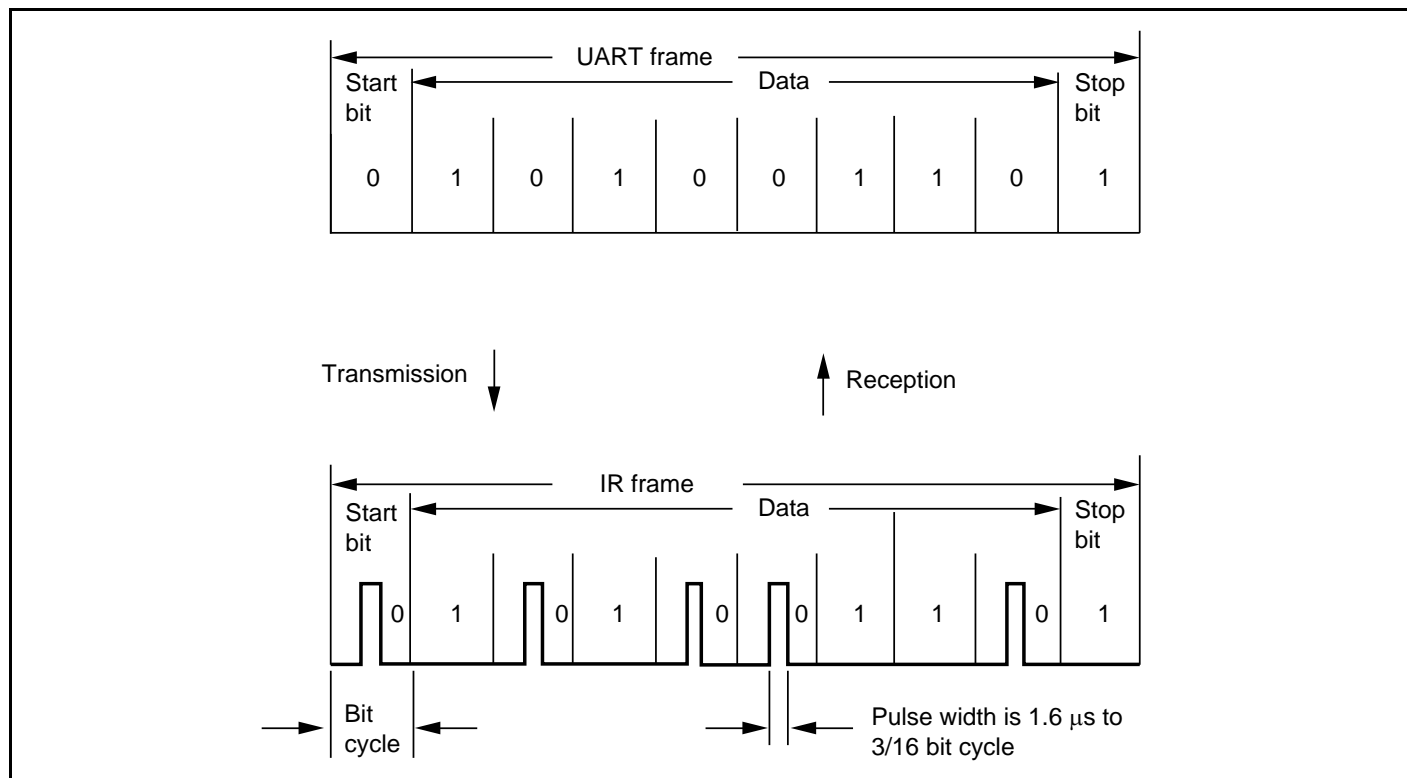
**Figure 16.36 IrDA Block Diagram**

**Transmission:** During transmission, the output signals from the SCI (UART frames) are converted to IR frames using the IrDA interface (see figure 16.37).

For serial data of level 0, a high-level pulse having a width of 3/16 of the bit rate (1-bit interval) is output (initial setting). The high-level pulse can be selected using the IrCKS2 to IrCKS0 bits in SCICR.

The high-level pulse width is defined to be  $1.41 \mu\text{s}$  at minimum and  $(3/16 + 2.5\%) \times \text{bit rate}$  or  $(3/16 \times \text{bit rate}) + 1.08 \mu\text{s}$  at maximum. For example, when the frequency of system clock  $\phi$  is 20 MHz, a high-level pulse width of at least  $1.4 \mu\text{s}$  to  $1.6 \mu\text{s}$  can be specified.

For serial data of level 1, no pulses are output.



**Figure 16.37 IrDA Transmission and Reception**

**Reception:** During reception, IR frames are converted to UART frames using the IrDA interface before inputting to SCI\_1.

Data of level 0 is output each time a high-level pulse is detected and data of level 1 is output when no pulse is detected in a bit cycle. If a pulse has a high-level width of less than  $1.41 \mu\text{s}$ , the minimum width allowed, the pulse is recognized as level 0.

**High-Level Pulse Width Selection:** Table 16.12 shows possible settings for bits IrCKS2 to IrCKS0 (minimum pulse width), and this LSI's operating frequencies and bit rates, for making the pulse width shorter than  $3/16$  times the bit rate in transmission.

**Table 16.12 IrCKS2 to IrCKS0 Bit Settings**

Operating Frequency $\phi$ (MHz)	Bit Rate (bps) (Upper Row)/Bit Interval $\times$ 3/16 ( $\mu$ s) (Lower Row)					
	<b>2400</b>	<b>9600</b>	<b>19200</b>	<b>38400</b>	<b>57600</b>	<b>115200</b>
	<b>78.13</b>	<b>19.53</b>	<b>9.77</b>	<b>4.88</b>	<b>3.26</b>	<b>1.63</b>
2	010	010	010	010	010	—
2.097152	010	010	010	010	010	—
2.4576	010	010	010	010	010	—
3	011	011	011	011	011	—
3.6864	011	011	011	011	011	011
4.9152	011	011	011	011	011	011
5	011	011	011	011	011	011
6	100	100	100	100	100	100
6.144	100	100	100	100	100	100
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
16.9344	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

Legend:

—: An SCI bit rate setting cannot be made.

## 16.9 Interrupt Sources

### 16.9.1 Interrupts in Normal Serial Communication Interface Mode

Table 16.13 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt can activate the DTC to allow data transfer. The TDRE flag is automatically cleared to 0 at data transfer by the DTC.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DTC to allow data transfer. The RDRF flag is automatically cleared to 0 at data transfer by the DTC.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared simultaneously by the TXI interrupt routine, the SCI cannot branch to the TEI interrupt routine later.

When the DTE bit in TDRE in SCIDTER\_0 and SCIDTER\_2 is 1, TXI0 and TXI2 interrupts are masked, and when the DTE bit in RDRF is 1, RXI0 and RXI2 interrupts are masked.

**Table 16.13 SCI Interrupt Sources**

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority
0	ERI0	Receive error	ORER, FER, PER	Not possible	High ↑ Low
	RXI0	Receive data full	RDRF	Possible	
	TXI0	Transmit data empty	TDRE	Possible	
	TEI0	Transmit end	TEND	Not possible	
1	ERI1	Receive error	ORER, FER, PER	Not possible	High ↑ Low
	RXI1	Receive data full	RDRF	Possible	
	TXI1	Transmit data empty	TDRE	Possible	
	TEI1	Transmit end	TEND	Not possible	
2	ERI2	Receive error	ORER, FER, PER	Not possible	High ↑ Low
	RXI2	Receive data full	RDRF	Possible	
	TXI2	Transmit data empty	TDRE	Possible	
	TEI2	Transmit end	TEND	Not possible	

### 16.9.2 Interrupts in Smart Card Interface Mode

Table 16.14 shows the interrupt sources in smart card interface mode. A TEI interrupt request cannot be used in this mode.

**Table 16.14 SCI Interrupt Sources**

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority
0	ERI0	Receive error, error signal detection	ORER, PER, ERS	Not possible	High ↑ Low
	RXI0	Receive data full	RDRF	Possible	
	TXI0	Transmit data empty	TEND	Possible	
1	ERI1	Receive error, error signal detection	ORER, PER, ERS	Not possible	High ↑ Low
	RXI1	Receive data full	RDRF	Possible	
	TXI1	Transmit data empty	TEND	Possible	
2	ERI2	Receive error, error signal detection	ORER, PER, ERS	Not possible	High ↑ Low
	RXI2	Receive data full	RDRF	Possible	
	TXI2	Transmit data empty	TEND	Possible	



Data transmission/reception using the DTC is also possible in smart card interface mode, similar to in the normal SCI mode. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request. This activates the DTC by a TXI interrupt request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DTC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DTC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DTC. Therefore, the SCI and DTC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag in SSR, which is set at error occurrence, is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit in SCR to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DTC, be sure to set and enable the DTC prior to making SCI settings. For DTC settings, see section 7, Data Transfer Controller (DTC).

In reception, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. This activates the DTC by an RXI interrupt request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DTC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DTC. If an error occurs, the RDRF flag is not set but the error flag is set. Therefore, the DTC is not activated and an ERI interrupt request is issued to the CPU instead; the error flag must be cleared.

When the DTE bit in TDRE in SCIDTER\_0 and SCIDTER\_2 is 1, TXI\_0 and TXI\_2 interrupt requests are masked, and when the DTE bit in RDRF is 1, RXI0 and RXI2 interrupt requests are masked.

## 16.10 Usage Notes

### 16.10.1 Module Stop Mode Setting

SCI operation can be disabled or enabled using the module stop control register. The initial setting is for SCI operation to be halted. Register access is enabled by clearing module stop mode. For details, see section 27, Power-Down Modes.

### 16.10.2 Break Detection and Processing

When framing error detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag in SSR is set, and the PER flag may also be set. Note that, since the SCI continues the receive operation even after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

### 16.10.3 Mark State and Break Detection

When the TE bit in SCR is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR of the port. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line at mark state until TE is set to 1, set both DDR and DR to 1. Since the TE bit is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set DDR to 1 and DR to 0, and then clear the TE bit to 0. When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

### 16.10.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) in SSR is set to 1, even if the TDRE flag in SSR is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that the receive error flags cannot be cleared to 0 even if the RE bit in SCR is cleared to 0.

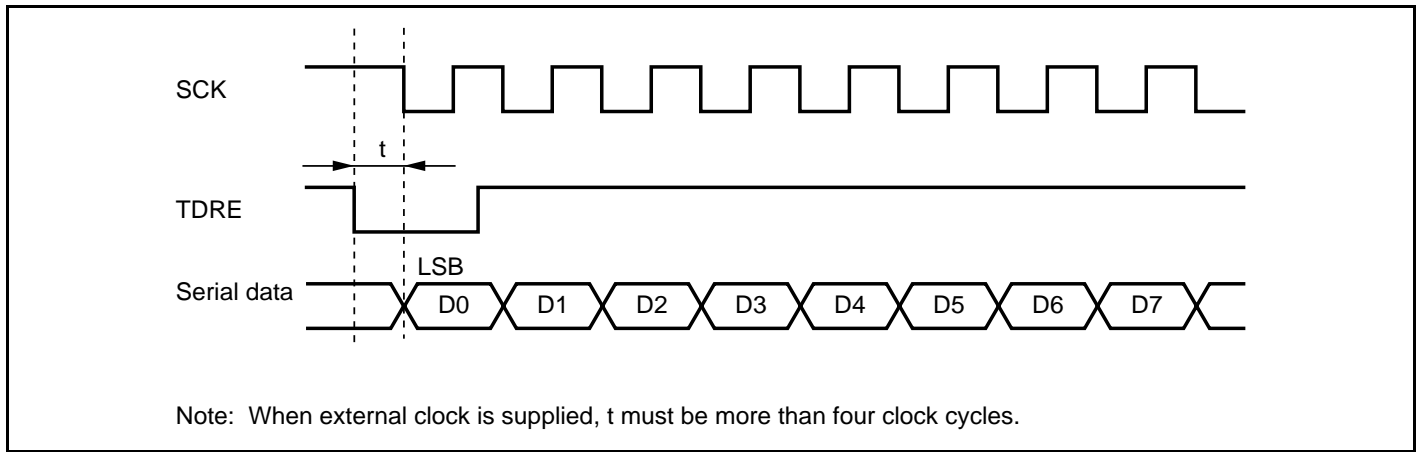
### 16.10.5 Relation between Writing to TDR and TDRE Flag

Data can be written to TDR irrespective of the TDRE flag status in SSR. However, if the new data is written to TDR when the TDRE flag is 0, that is, when the previous data has not been transferred to TSR yet, the previous data in TDR is lost. Be sure to write transmit data to TDR after verifying that the TDRE flag is set to 1.

### 16.10.6 Restrictions on Using DTC or RFU

When the external clock source is used as a synchronization clock, update TDR by the DTC or RFU and wait for at least five  $\phi$  clock cycles before allowing the transmit clock to be input. If the transmit clock is input within four clock cycles after TDR modification, the SCI may malfunction (figure 16.38).

When using the DTC or RFU to read RDR, be sure to set the receive end interrupt source (RXI) as a DTC or RFU activation source.



**Figure 16.38 Sample Transmission using DTC in Clocked Synchronous Mode**

### 16.10.7 SCI Operations during Mode Transitions

**Transmission:** Before making the transition to module stop, software standby, or sub-sleep mode, stop all transmit operations ( $TE = TIE = TEIE = 0$ ). TSR, TDR, and SSR are reset. The states of the output pins during each mode depend on the port settings, and the pins output a high-level signal after mode cancellation. If the transition is made during data transmission, the data being transmitted will be undefined.

To transmit data in the same transmission mode after mode cancellation, set TE to 1, read SSR, write to TDR, clear TDRE in this order, and then start transmission. To transmit data in a different transmission mode, initialize the SCI first.

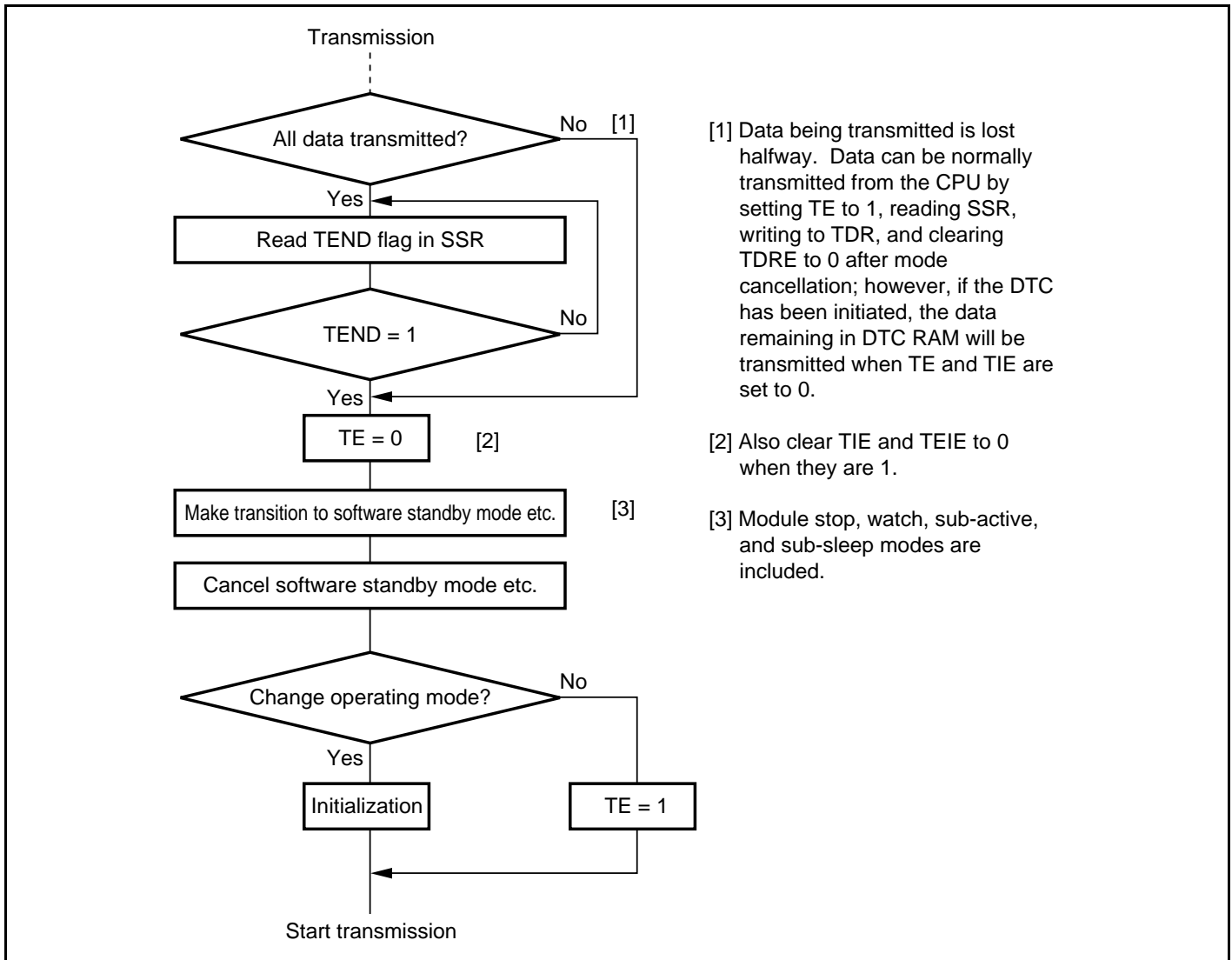
Figure 16.39 shows a sample flowchart for mode transition during transmission. Figures 16.40 and 16.41 show the pin states during transmission.

Before making the transition from the transmission mode using DTC transfer to module stop, software standby, or sub-sleep mode, stop all transmit operations ( $TE = TIE = TEIE = 0$ ). Setting TE and TIE to 1 after mode cancellation generates a TXI interrupt request to start transmission using the DTC.

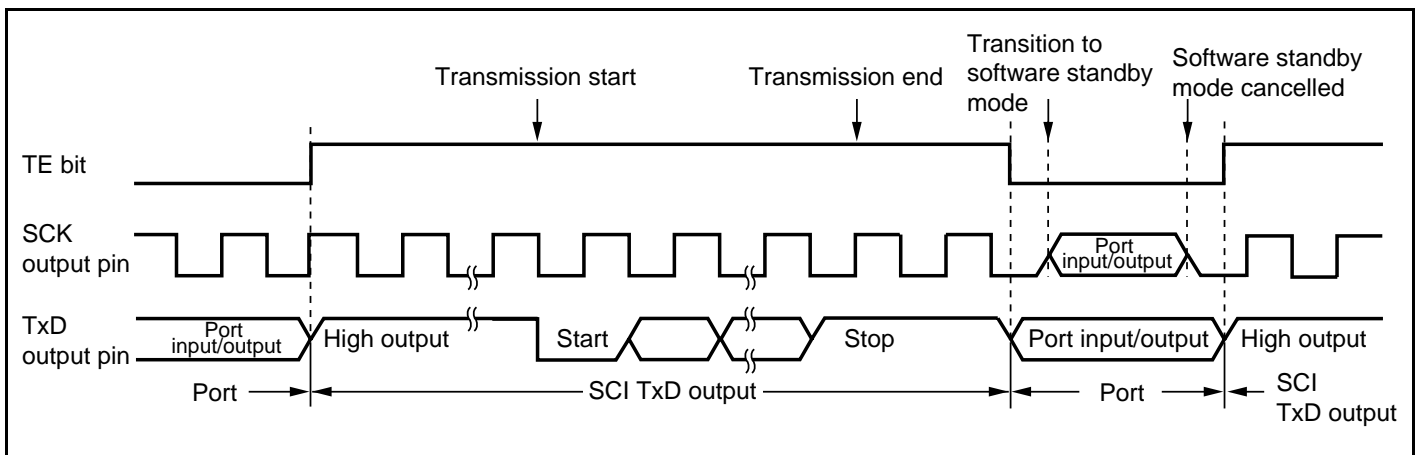
**Reception:** Before making the transition to module stop, software standby, watch, sub-active, or sub-sleep mode, stop reception ( $RE = 0$ ). RSR, RDR, and SSR are reset. If transition is made during data reception, the data being received will be invalid.

To receive data in the same reception mode after mode cancellation, set RE to 1, and then start reception. To receive data in a different reception mode, initialize the SCI first.

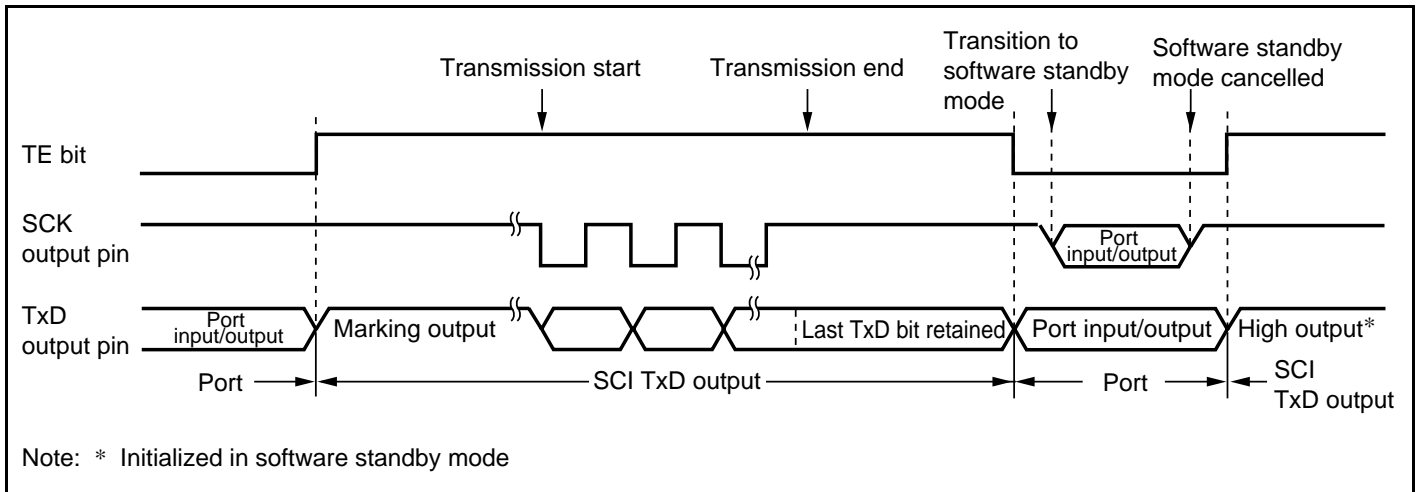
Figure 16.42 shows a sample flowchart for mode transition during reception.



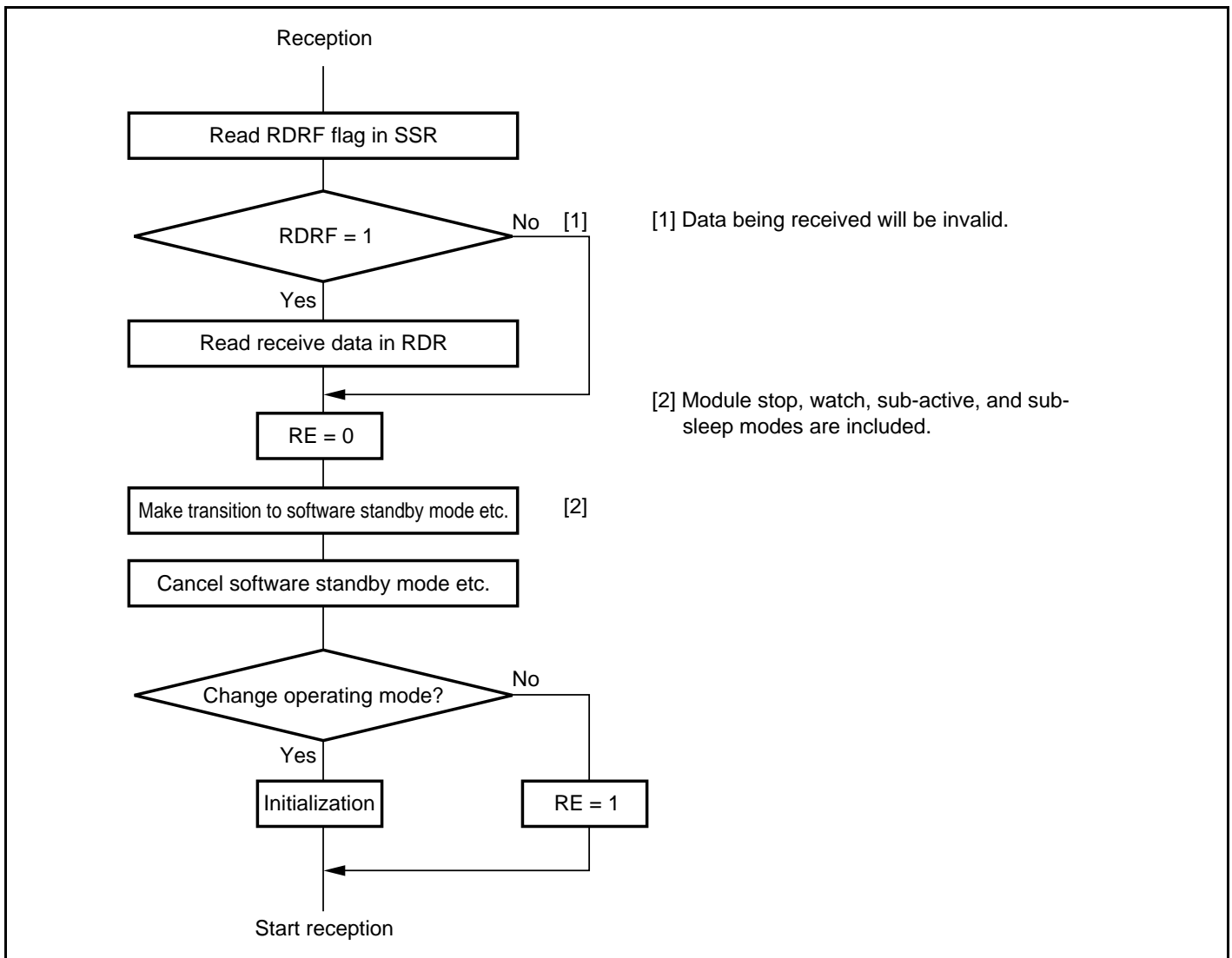
**Figure 16.39 Sample Flowchart for Mode Transition during Transmission**



**Figure 16.40 Pin States during Transmission in Asynchronous Mode (Internal Clock)**



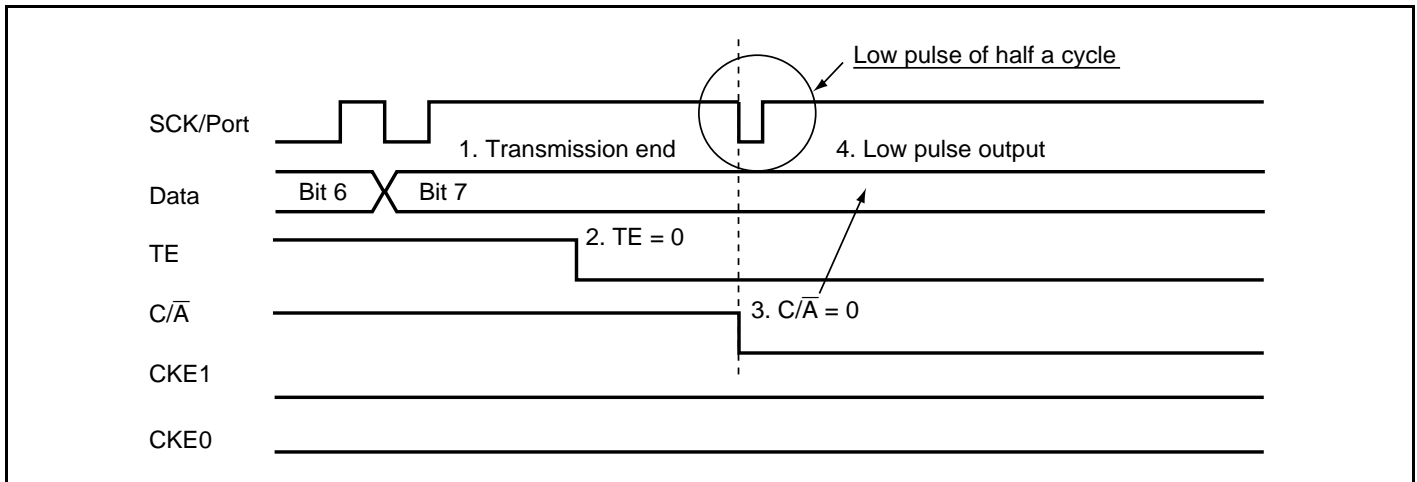
**Figure 16.41 Pin States during Transmission in Clocked Synchronous Mode (Internal Clock)**



**Figure 16.42 Sample Flowchart for Mode Transition during Reception**

### 16.10.8 Notes on Switching from SCK Pins to Port Pins

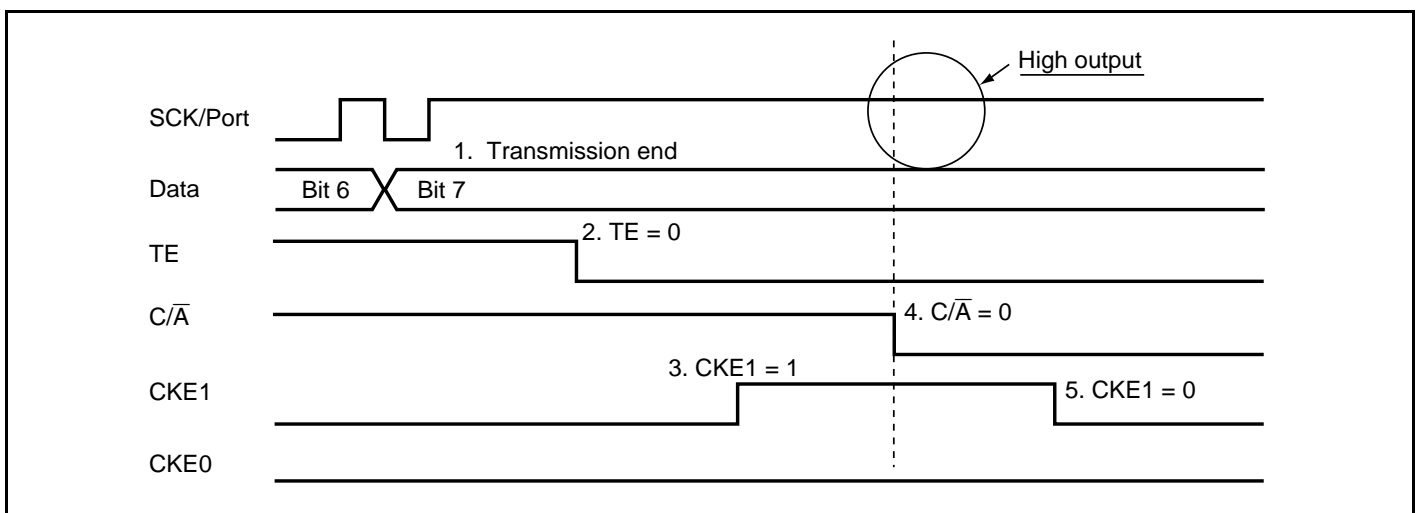
When SCK pins are switched to port pins after transmission has completed, pins are enabled for port output after outputting a low pulse of half a cycle as shown in figure 16.43.



**Figure 16.43 Switching from SCK Pins to Port Pins**

To prevent the low pulse output that is generated when switching the SCK pins to the port pins, specify the SCK pins for input (pull up the SCK/port pins externally), and follow the procedure below with  $DDR = 1$ ,  $DR = 1$ ,  $C/\bar{A} = 1$ ,  $CKE1 = 0$ ,  $CKE1 = 0$ , and  $TE = 1$ .

1. End serial data transmission
2. TE bit = 0
3. CKE1 bit = 1
4. C/A bit = 0 (switch to port output)
5. CKE1 bit = 0



**Figure 16.44 Prevention of Low Pulse Output at Switching from SCK Pins to Port Pins**

## 16.11 CRC Operation Circuit

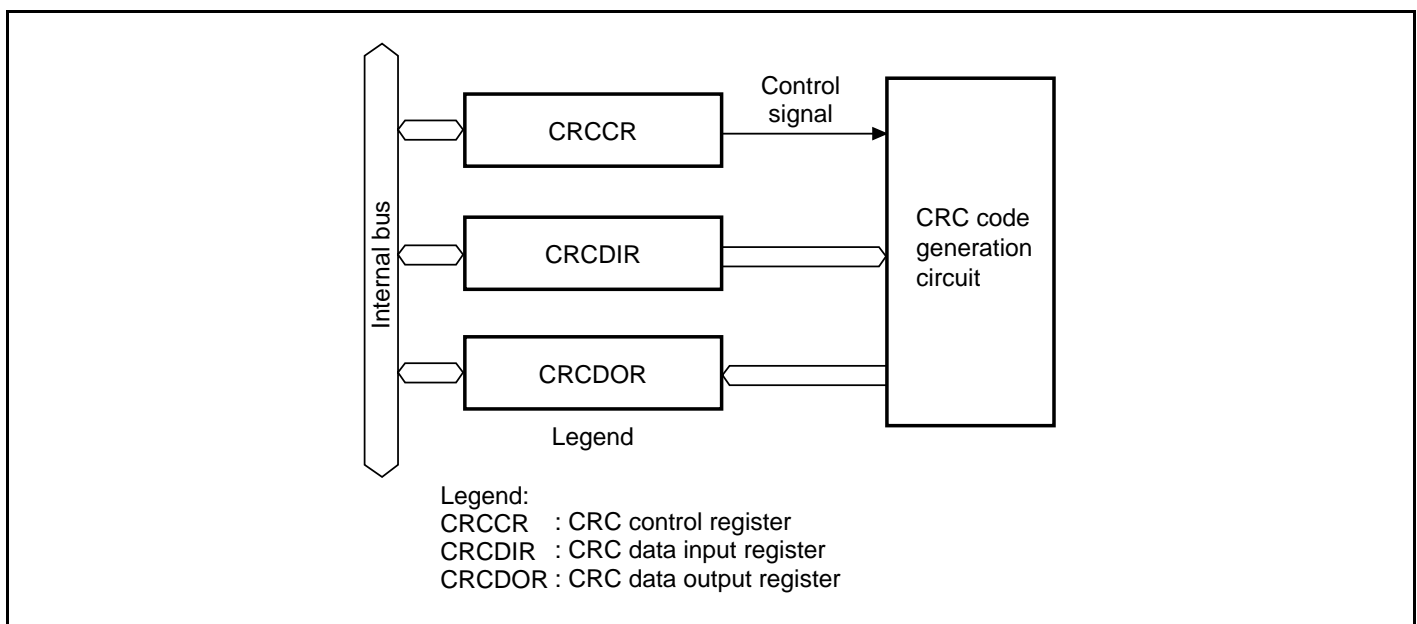
The cyclic redundancy check (CRC) operation circuit detects errors in data blocks.

### 16.11.1 Features

The features of the CRC operation circuit are listed below.

- CRC code generated for any desired data length in an 8-bit unit
- CRC operation executed on eight bits in parallel
- One of three generating polynomials selectable
- CRC code generation for LSB-first or MSB-first communication selectable

Figure 16.45 shows a block diagram of the CRC operation circuit.



**Figure 16.45 Block Diagram of CRC Operation Circuit**

### 16.11.2 Register Descriptions

The CRC operation circuit has the following registers.

- CRC control register (CRCCR)
- CRC data input register (CRCDIR)
- CRC data output register (CRCDOR)

**CRC Control Register (CRCCR):** CRCCR initializes the CRC operation circuit, switches the operation mode, and selects the generating polynomial.

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) is first transmitted when CRCDOR contents (CRC code) are divided into two bytes to be transmitted in two parts. 1: Performs CRC operation for MSB-first communication. The upper byte (bits 15 to 8) is first transmitted when CRCDOR contents (CRC code) are divided into two bytes to be transmitted in two parts.
1	G1	0	R/W	CRC Generating Polynomial Select: Selects the polynomial. 00: Reserved 01: $X^8 + X^2 + X + 1$ 10: $X^{16} + X^{15} + X^2 + 1$ 11: $X^{16} + X^{12} + X^5 + 1$
0	G0	0	R/W	

**CRC Data Input Register (CRCDIR):** CRCDIR is an 8-bit readable/writable register, to which the bytes to be CRC-operated are written. The result is obtained in CRCDOR.

**CRC Data Output Register (CRCDOR):** CRCDOR is a 16-bit readable/writable register that contains the result of CRC operation when 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 in CRCCR (G1 and G0 bits) are set to 0 and 1, respectively, the lower byte of this register contains the result.



### 16.11.3 CRC Operation Circuit Operation

The CRC operation circuit generates a CRC code for LSB-first/MSB-first communications. An example in which a CRC code for hexadecimal data H'F0 is generated using the  $X^{16} + X^{12} + X^5 + 1$  polynomial with the G1 and G0 bits in CRCCR set to B'11 is shown below.

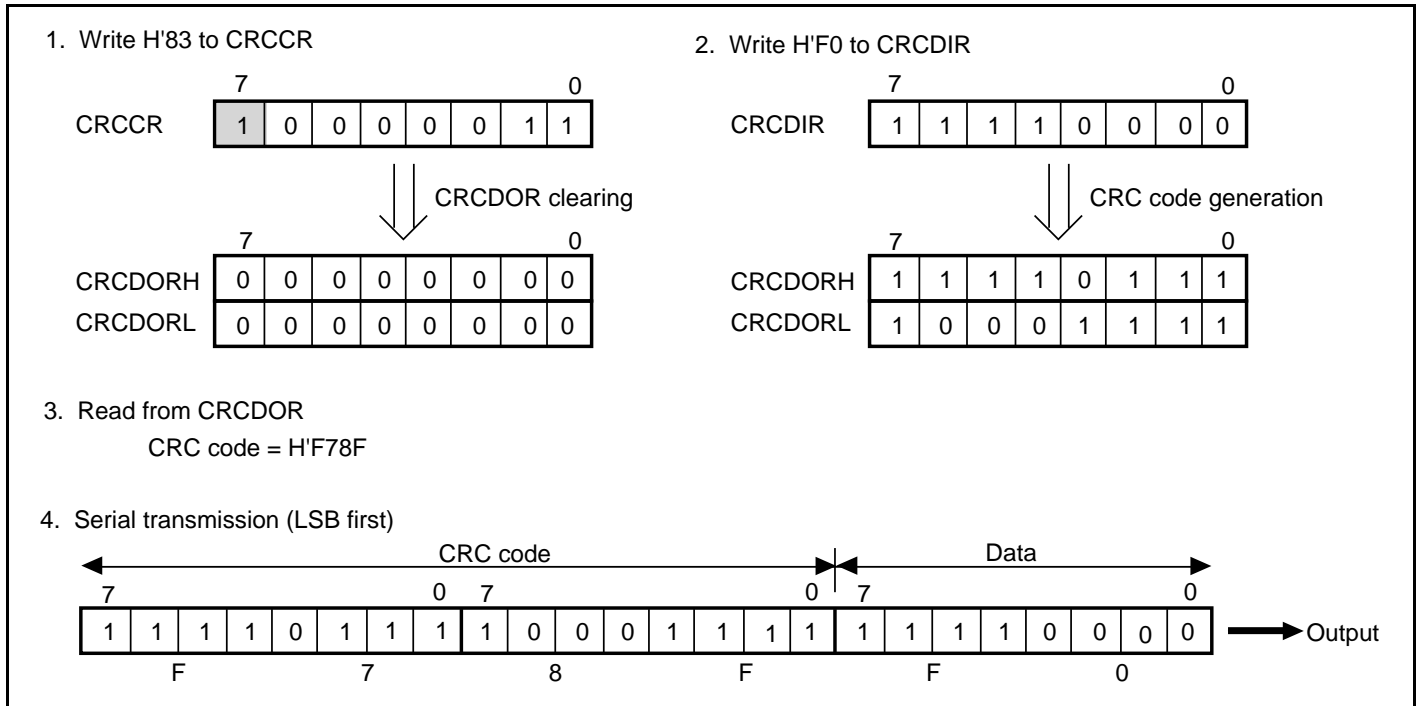


Figure 16.46 LSB-First Data Transmission

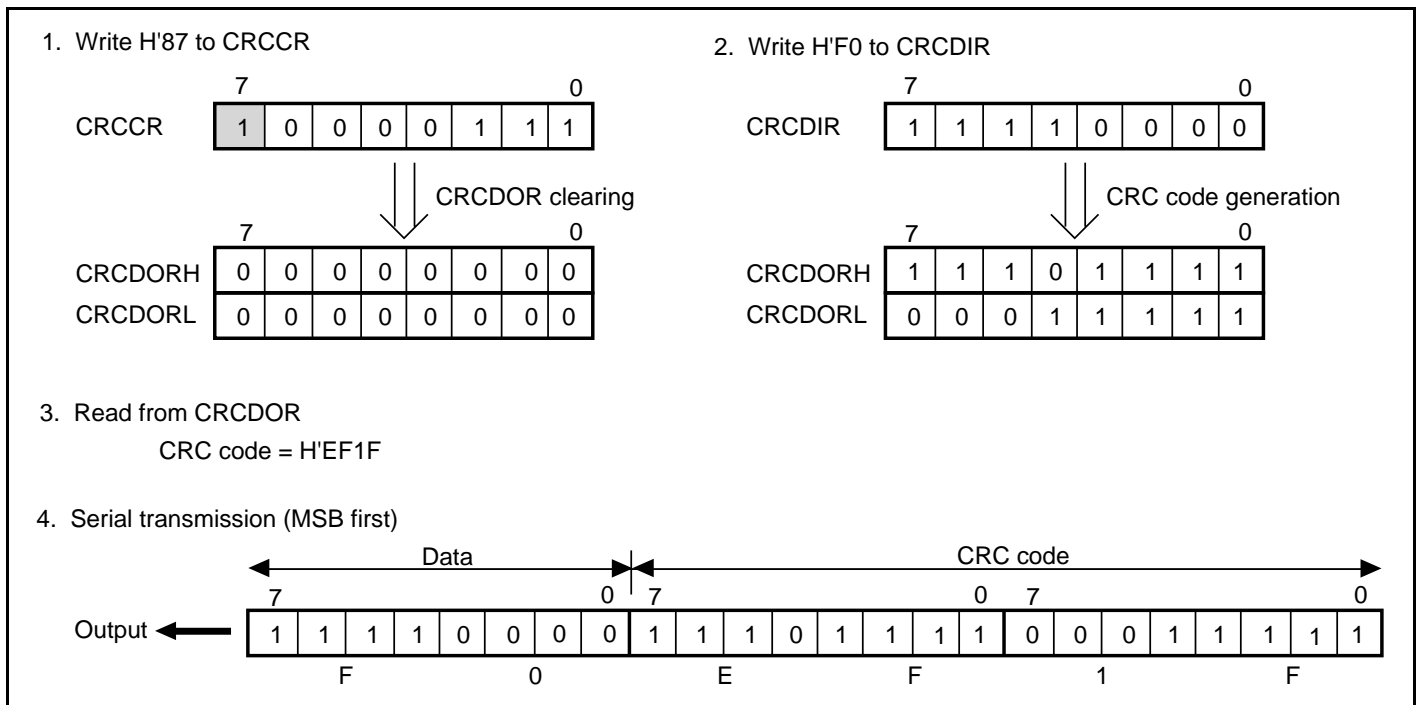


Figure 16.47 MSB-First Data Transmission

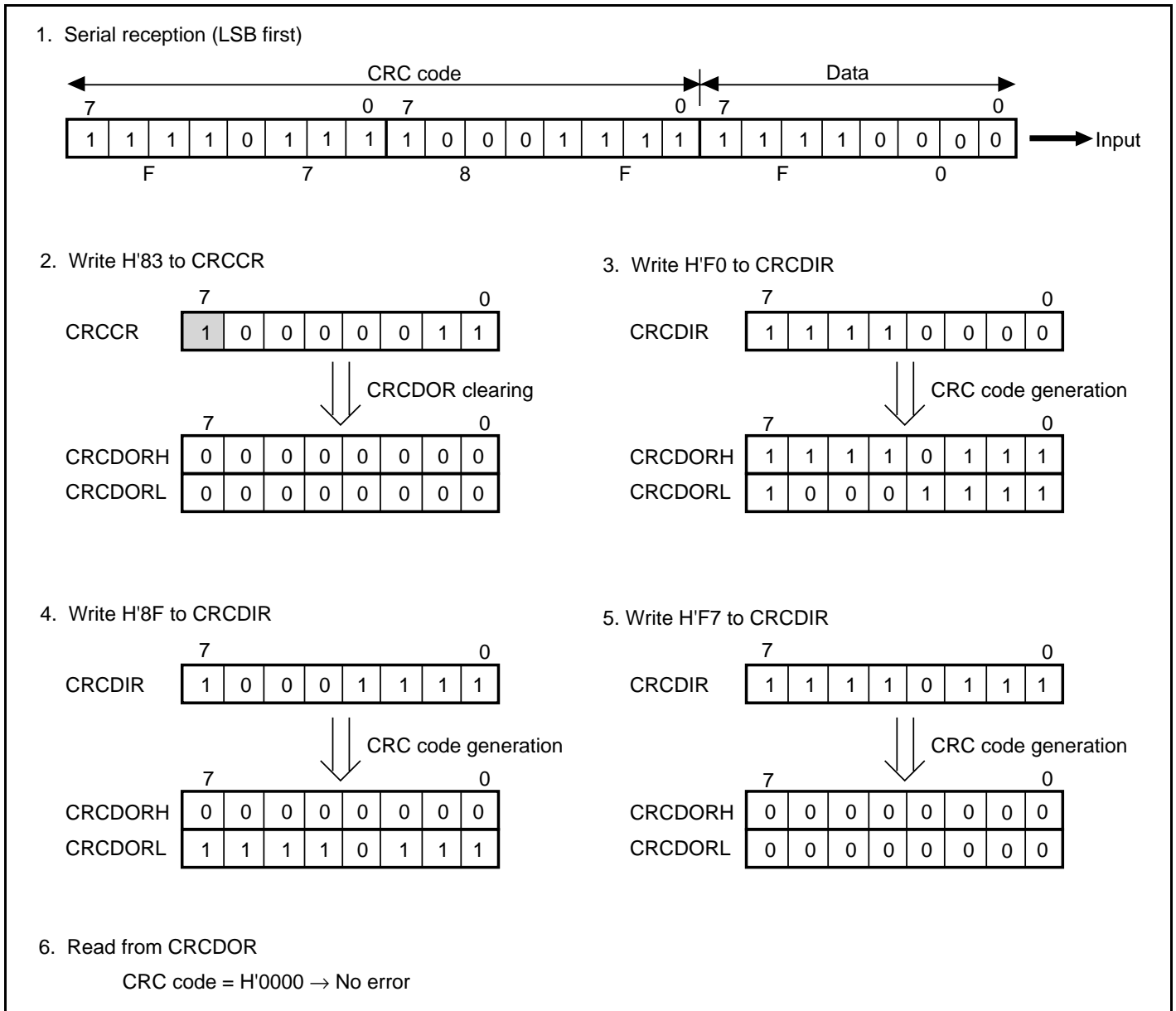


Figure 16.48 LSB-First Data Reception

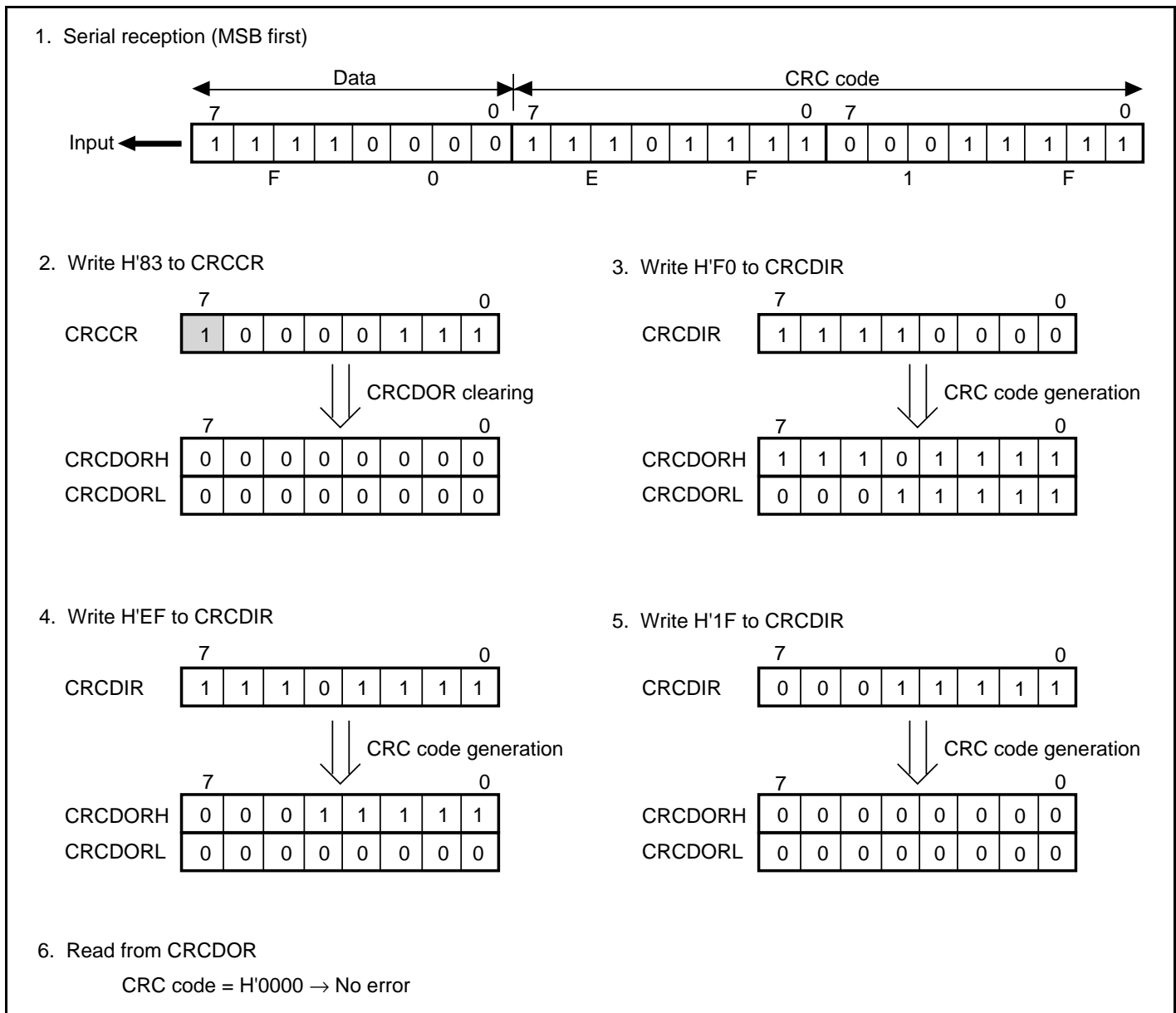
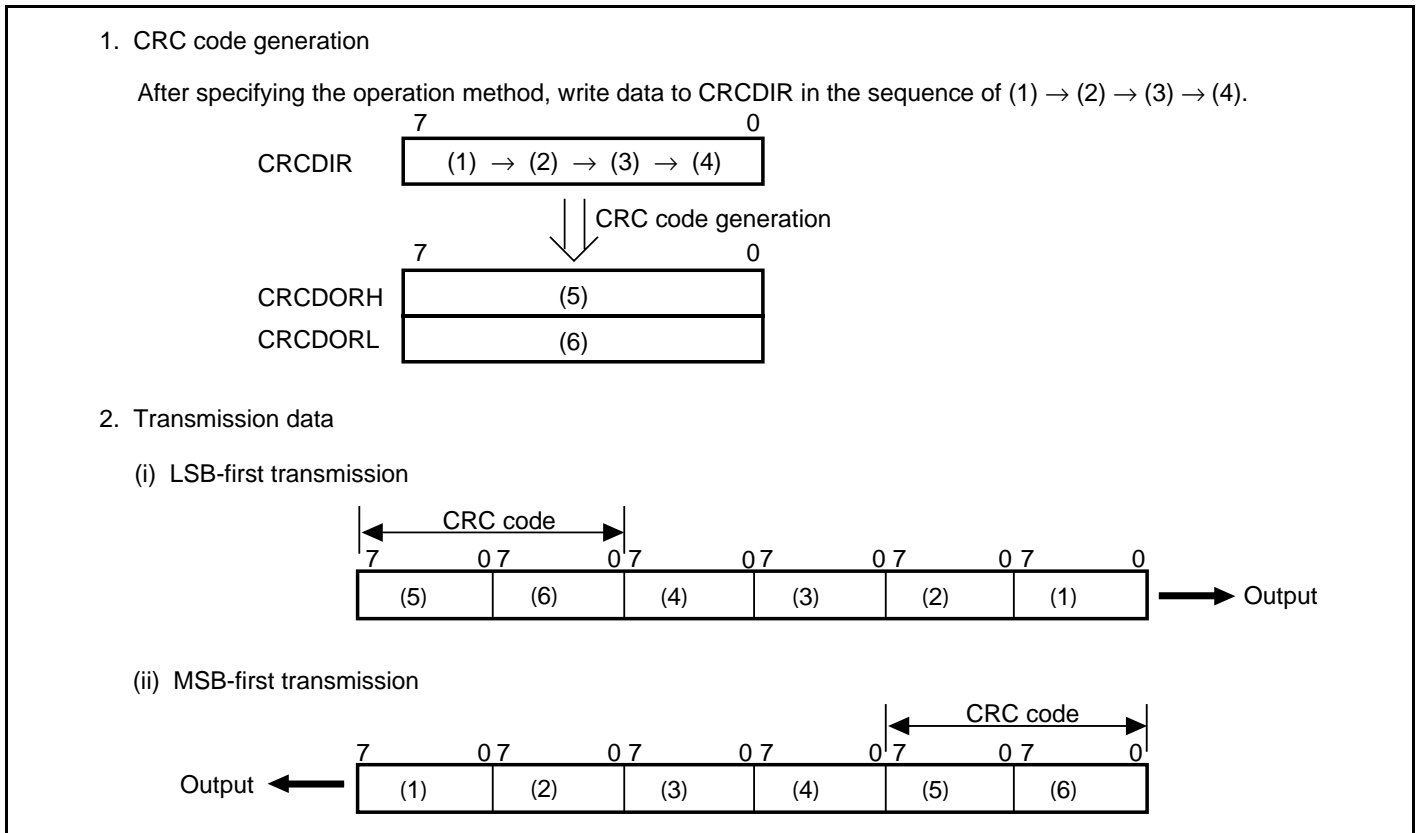


Figure 16.49 MSB-First Data Reception

### 16.11.4 Note on CRC Operation Circuit

Note that the sequence to transmit the CRC code differs between LSB-first transmission and MSB-first transmission.



**Figure 16.50 LSB-First and MSB-First Transmit Data**

## Section 17 I<sup>2</sup>C Bus Interface (IIC)

This LSI has an I<sup>2</sup>C bus interface (IIC) of two channels.

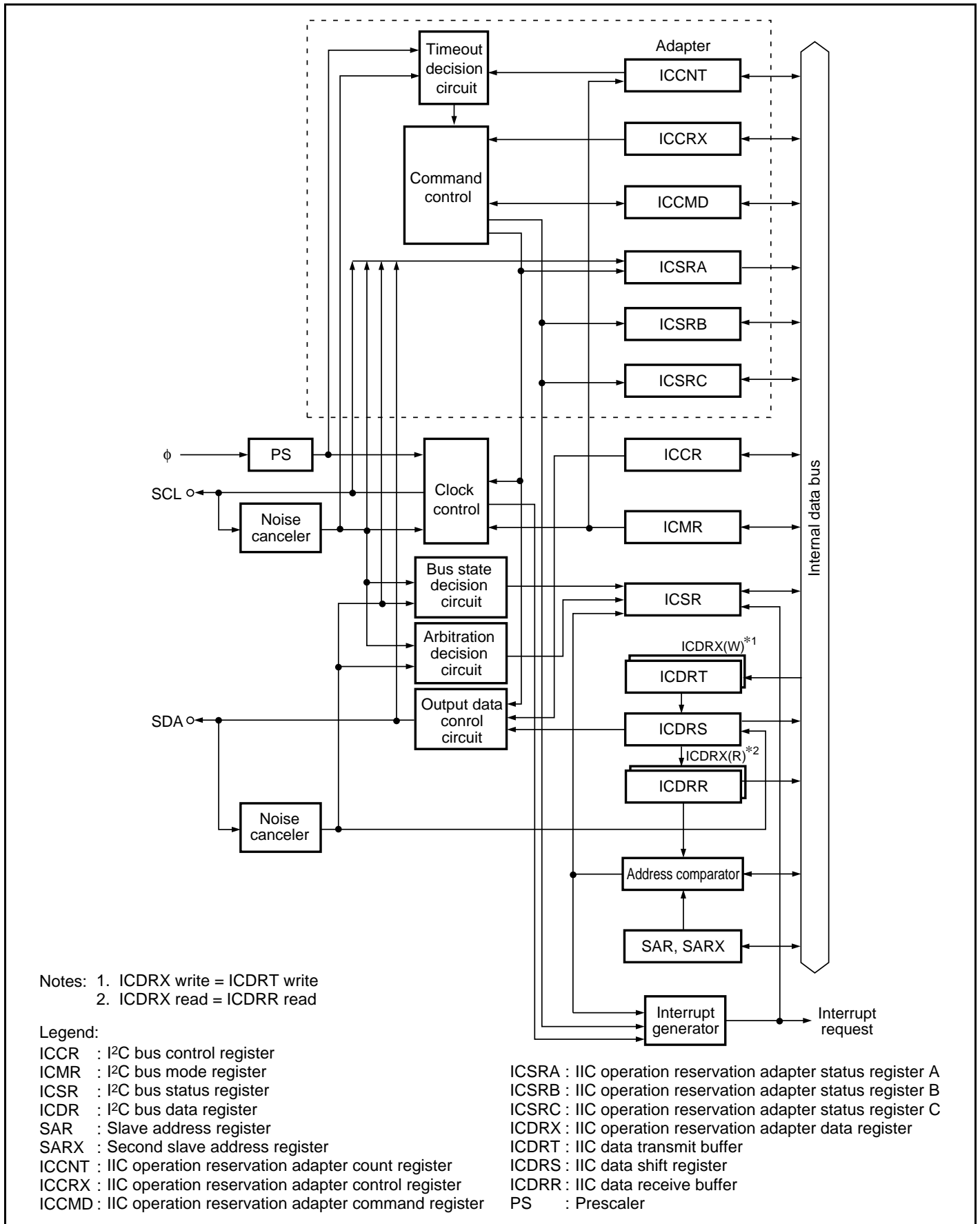
The I<sup>2</sup>C bus interface conforms to and provides a subset of the Philips I<sup>2</sup>C bus (inter-IC bus) interface functions. Note however that the register configuration that controls the I<sup>2</sup>C bus differs partly from the Philips configuration.

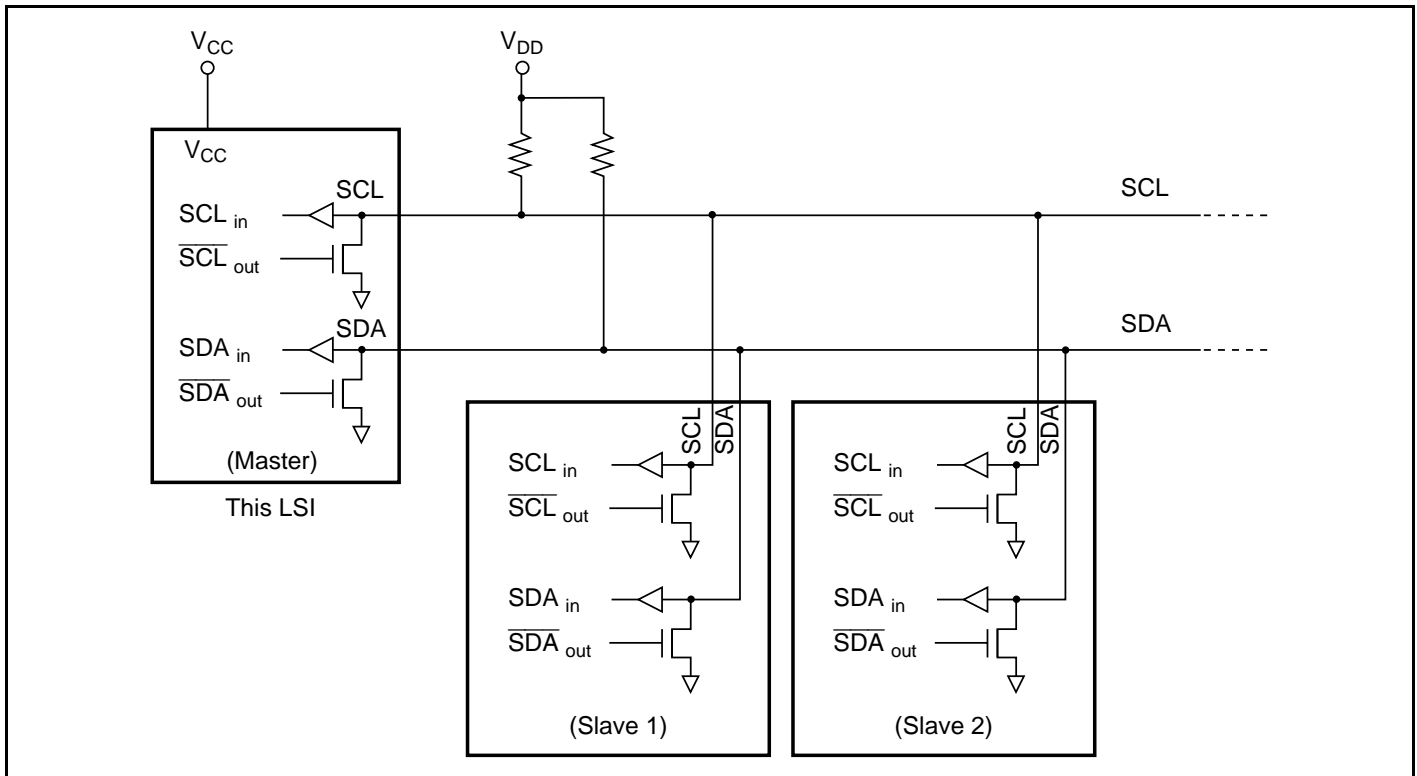
### 17.1 Features

- Selection of I<sup>2</sup>C bus format or clocked synchronous serial format
  - I<sup>2</sup>C bus format: Addressing format with acknowledge bit, for master/slave operation
  - Clocked synchronous serial format: Non-addressing format without acknowledge bit, for master operation only
- For I<sup>2</sup>C bus format, two ways of setting slave address
- For I<sup>2</sup>C bus format, start and stop conditions generated automatically in master mode
- For I<sup>2</sup>C bus format, selection of acknowledge output levels when receiving
- For I<sup>2</sup>C bus format, automatic loading of acknowledge bit when transmitting
- For I<sup>2</sup>C bus format, wait bit function in master mode
  - A wait can be inserted by driving the SCL pin low after data transfer, excluding acknowledgement.
  - The wait can be cleared by clearing the interrupt flag.
- For I<sup>2</sup>C bus format, wait function is available
  - A wait request can be generated by driving the SCL pin low after data transfer, excluding acknowledgement.
  - The wait request is cleared when the next transfer becomes possible.
- Interrupt sources
  - Data transfer end (including when a transition to transmit mode with I<sup>2</sup>C bus format occurs, when ICDR data is transferred, or during a wait state)
  - Address match: When any slave address matches or the general call address is received in slave receive mode with I<sup>2</sup>C bus format (including address reception after loss of master arbitration)
  - Arbitration loss
  - Start condition detection (in master mode)
  - Stop condition detection (in slave mode)
- Selection of 16 internal clocks (in master mode)

- Direct bus drive  
Two pins, SCL and SDA, function as NMOS open-drain outputs when the bus drive function is selected.
- Operation using the operation reservation adapter

Figure 17.1 shows a block diagram of the I<sup>2</sup>C bus interface. Figure 17.2 shows an example of I/O pin connections to external circuits. I/O pins are NMOS open drain, and 5.8 V or less should be applied as the power supply voltage.

Figure 17.1 Block Diagram of I<sup>2</sup>C Bus Interface



**Figure 17.2 I<sup>2</sup>C Bus Interface Connections (Example: This LSI as Master)**

## 17.2 Input/Output Pins

Table 17.1 summarizes the input/output pins used by the I<sup>2</sup>C bus interface.

**Table 17.1 Pin Configuration**

Channel	Symbol	Input/Output	Function
0	SCL0	Input/Output	Clock input/output pin of channel 0
	SDA0	Input/Output	Data input/output pin of channel 0
1	SCL1	Input/Output	Clock input/output pin of channel 1
	SDA1	Input/Output	Data input/output pin of channel 1

## 17.3 Register Descriptions

The I<sup>2</sup>C bus interface has the following registers. Registers ICDR and SARX and registers ICMR and SAR are allocated to the same addresses. Accessible registers differ depending on the ICE bit in ICCR. When the ICE bit is cleared to 0, SAR and SARX can be accessed, and when the ICE bit is set to 1, ICMR and ICDR can be accessed. For details on the serial timer control register, see section 3.2.3, Serial Timer Control Register (STCR).



- I<sup>2</sup>C bus data register (ICDR)
- Slave address register (SAR)
- Second slave address register (SARX)
- I<sup>2</sup>C bus mode register (ICMR)
- I<sup>2</sup>C bus control register (ICCR)
- I<sup>2</sup>C bus status register (ICSR)
- IIC operation reservation adapter control register (ICCRX)
- IIC operation reservation adapter status register A (ICSRA)
- IIC operation reservation adapter status register B (ICSRB)
- IIC operation reservation adapter status register C (ICSRC)
- IIC operation reservation adapter data register (ICDRX)
- IIC data shift register (ICDRS)
- IIC operation reservation adapter count register (ICCNT)
- IIC operation reservation adapter command register (ICCMD)

### 17.3.1 I<sup>2</sup>C Bus Data Register (ICDR)

ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. ICDR is divided internally into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). Data transfers among the three registers are performed automatically in accordance with changes in the bus state, and they affect the status of internal flags such as ICDRE and ICDRF. When ICDRE is 1 and the transmit buffer is empty, ICDRE shows that the next transmit data can be written from the CPU. When ICDRF is 1, it shows that valid receive data is stored in the receive buffer.

If I<sup>2</sup>C is in transmit mode and the next transmit data is in the transmit buffer (the ICDRE flag is 0) after successful transmission/reception of one frame of data using the shift register, data is transferred automatically from the transmit buffer to the shift register. If I<sup>2</sup>C is in receive mode and no previous data remains in the receive buffer (the ICDRF flag is 0), data is transferred automatically from the shift register to the receive buffer. Note however that no data is transferred from the transmit buffer to the shift register in receive mode, and from the shift register to the receive buffer in transmit mode. If ICDR is read from in transmit mode, data in the receive buffer can be read but data in the shift register cannot. Always set I<sup>2</sup>C to receive mode before reading from ICDR.

If the number of bits in a frame, excluding the acknowledge bit, is less than eight, transmit data and receive data are stored differently. Transmit data should be written justified toward the MSB side when MLS = 0 in ICMR, and toward the LSB side when MLS = 1. Receive data bits should be read from the LSB side when MLS = 0, and from the MSB side when MLS = 1.

ICDR can be written to and read from only when the ICE bit is set to 1 in ICCR. The initial value of ICDR is undefined.

The ICDRE and ICDRF flags are set and cleared under the conditions shown below. Setting the ICDRE and ICDRF flags affects the status of the interrupt flags.

Bit	Bit Name	Initial Value	R/W	Description
—	ICDRE	—	—	<p>Transmit Data Register Empty</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>When satisfaction of a start condition is detected in the bus line state with the I<sup>2</sup>C bus format or serial format selected</li> <li>When data is transferred from the transmit buffer to the shift register (Data transfer from the transmit buffer to the shift register if the shift register is empty when ICDRE = 0 in transmit mode) (Do not write to ICDR in receive mode because the ICDRE flag value is invalid)</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When transmit data is written in ICDR (transmit buffer) in transmit mode</li> <li>When satisfaction of a stop condition is detected in the bus line state with the I<sup>2</sup>C bus format or serial format selected If transmit data is written to ICDR (transmit buffer) in transmit mode when ICDR does not contain data to be transmitted, ICDRE is cleared to 0. However, since data is transferred from the transmit buffer to the shift register immediately, ICDRE is set to 1 again.</li> <li>Internal state initialization (Writing 0 to the TRS bit in ICCR during transfer is valid after reception of a frame containing an acknowledge bit)</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
—	ICDRF	—	—	<p>Receive Data Register Full</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When data is transferred from the shift register to the receive buffer</li> </ul> <p>(Data transfer from the shift register to the receive buffer if there is receive data in the shift register when ICDRF = 0 in receive mode)</p> <p>(Data is not transferred from the shift register to the receive buffer in transmit mode. To read data in the shift register, read ICDR in receive mode.)</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When receive data in ICDR (receive buffer) is read in receive mode</li> </ul> <p>If receive data is read from ICDR (receive buffer) in receive mode when the shift register contains the next receive data, ICDRF is cleared to 0. However, since data is transferred from the shift register to the receive buffer immediately, ICDRF is set to 1 again.</p> <ul style="list-style-type: none"> <li>Internal state initialization</li> </ul>

### 17.3.2 Slave Address Register (SAR)

SAR sets the slave address and selects the communication format. When the LSI is in slave mode with the addressing format selected, 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 specified by the master device. SAR can be accessed only when the ICE bit in ICCR is cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7	SVA6	0	R/W	Slave Address 6 to 0
6	SVA5	0	R/W	Set a slave address.
5	SVA4	0	R/W	
4	SVA3	0	R/W	
3	SVA2	0	R/W	
2	SVA1	0	R/W	
1	SVA0	0	R/W	
0	FS	0	R/W	Format Select Selects the communication format together with the FSX bit in SARX. For details, see table 17.2. This bit should be cleared to 0 when general call address recognition is performed.

### 17.3.3 Second Slave Address Register (SARX)

SARX sets the second slave address and selects the communication format. In slave mode, transmit/receive operations by the DTC are possible when the received address matches the second slave address. When the LSI is in slave mode with the I<sup>2</sup>C bus format selected, if the upper 7 bits of SARX match the upper 7 bits of the first frame received after a start condition, the LSI operates as the slave device specified by the master device. SARX can be accessed only when the ICE bit in ICCR is cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7	SVAX6	0	R/W	Second Slave Address 6 to 0
6	SVAX5	0	R/W	Set the second slave address.
5	SVAX4	0	R/W	
4	SVAX3	0	R/W	
3	SVAX2	0	R/W	
2	SVAX1	0	R/W	
1	SVAX0	0	R/W	
0	FSX	1	R/W	Format Select X Selects the communication format together with the FS bit in SAR. For details, see table 17.2.

**Table 17.2 Communication Format**

SAR	SARX	Communication Format
FS	FSX	
0	0	SAR and SARX are used as the slave addresses with the I <sup>2</sup> C bus format.
0	1	Only SAR is used as the slave address with the I <sup>2</sup> C bus format.
1	0	Only SARX is used as the slave address with the I <sup>2</sup> C bus format.
1	1	Clocked synchronous serial format (SAR and SARX are invalid)

### 17.3.4 I<sup>2</sup>C Bus Mode Register (ICMR)

ICMR sets the communication format and transfer rate. It can only be accessed when the ICE bit in ICCR is set to 1.

Bit	Bit Name	Initial Value	R/W	Description
7	MLS	0	R/W	MSB-First/LSB-First Select 0: MSB-first 1: LSB-first Set this bit to 0 when the I <sup>2</sup> C bus format is used.
6	WAIT	0	R/W	Wait Insertion Bit This bit is valid only in master mode with the I <sup>2</sup> C bus format. 0: Data and the acknowledge bit are transferred consecutively with no wait inserted. 1: After the fall of the clock for the final data bit, the IRIC flag is set to 1 in ICCR, and a wait state begins (with SCL at the low level). When the IRIC flag is cleared to 0 in ICCR, the wait ends and the acknowledge bit is transferred. For details, see section 17.5.6, IRIC Setting Timing and SCL Control.
5	CKS2	0	R/W	Serial Clock Select 2 to 0
4	CKS1	0	R/W	These bits are valid only in master mode.
3	CKS0	0	R/W	These bits select the required transfer rate, together with bit IICX1 (channel 1) or IICX0 (channel 0) in STCR. See table 17.3.

Bit	Bit Name	Initial Value	R/W	Description																		
2	BC2	0	R/W	Bit Counter 2 to 0																		
1	BC1	0	R/W	These bits specify the number of bits to be transferred next. With the I <sup>2</sup> C bus format, the data is transferred with one additional acknowledge bit. Bit BC2 to BC0 settings should be made during an interval between transfer frames. If bits BC2 to BC0 are set to a value other than B'000, the setting should be made while the SCL line is low. The value returns to B'000 when a start condition is detected or when data transfer including the acknowledge bit ends.																		
0	BC0	0	R/W																			
				<table border="0"> <tr> <td>I<sup>2</sup>C Bus Format</td> <td>Clocked Synchronous Serial Mode</td> </tr> <tr> <td>000: 9 bits</td> <td>000: 8 bits</td> </tr> <tr> <td>001: 2 bits</td> <td>001: 1 bits</td> </tr> <tr> <td>010: 3 bits</td> <td>010: 2 bits</td> </tr> <tr> <td>011: 4 bits</td> <td>011: 3 bits</td> </tr> <tr> <td>100: 5 bits</td> <td>100: 4 bits</td> </tr> <tr> <td>101: 6 bits</td> <td>101: 5 bits</td> </tr> <tr> <td>110: 7 bits</td> <td>110: 6 bits</td> </tr> <tr> <td>111: 8 bits</td> <td>111: 7 bits</td> </tr> </table>	I <sup>2</sup> C Bus Format	Clocked Synchronous Serial Mode	000: 9 bits	000: 8 bits	001: 2 bits	001: 1 bits	010: 3 bits	010: 2 bits	011: 4 bits	011: 3 bits	100: 5 bits	100: 4 bits	101: 6 bits	101: 5 bits	110: 7 bits	110: 6 bits	111: 8 bits	111: 7 bits
I <sup>2</sup> C Bus Format	Clocked Synchronous Serial Mode																					
000: 9 bits	000: 8 bits																					
001: 2 bits	001: 1 bits																					
010: 3 bits	010: 2 bits																					
011: 4 bits	011: 3 bits																					
100: 5 bits	100: 4 bits																					
101: 6 bits	101: 5 bits																					
110: 7 bits	110: 6 bits																					
111: 8 bits	111: 7 bits																					

Table 17.3 I<sup>2</sup>C Transfer Rate

STCR		ICMR			Transfer Rate							
Bit 6/5	Bit 5	Bit 4	Bit 3	Clock	$\phi = 5$	$\phi = 8$	$\phi = 10$	$\phi = 12$	$\phi = 16$	$\phi = 20$	$\phi = 24$	
IICX1/ IICX0	CKS2	CKS1	CKS0		MHz	MHz	MHz	MHz	MHz	MHz	MHz	
0	0	0	0	$\phi/28$	179 kHz	286 kHz	357 kHz	429 kHz*	517 kHz*	714 kHz*	857 kHz*	
			1	$\phi/40$	125 kHz	200 kHz	250 kHz	300 kHz	400 kHz	500 kHz*	600 kHz*	
		1	0	$\phi/48$	104 kHz	167 kHz	208 kHz	250 kHz	333 kHz	417 kHz*	500 kHz*	
			1	$\phi/64$	78.1 kHz	125 kHz	156 kHz	188 kHz	250 kHz	313 kHz	375 kHz	
	1	0	0	$\phi/80$	62.5 kHz	100 kHz	125 kHz	150 kHz	200 kHz	250 kHz	300 kHz	
			1	$\phi/100$	50.0 kHz	80.0 kHz	100 kHz	120 kHz	160 kHz	200 kHz	240 kHz	
		1	0	$\phi/112$	44.6 kHz	71.4 kHz	89.3 kHz	107 kHz	143 kHz	179 kHz	214 kHz	
			1	$\phi/128$	39.1 kHz	62.5 kHz	78.1 kHz	93.8 kHz	125 kHz	156 kHz	188 kHz	
	1	0	0	0	$\phi/56$	89.3 kHz	143 kHz	179 kHz	212 kHz	286 kHz	357 kHz	429 kHz
				1	$\phi/80$	62.5 kHz	100 kHz	125 kHz	150 kHz	200 kHz	250 kHz	300 kHz
			1	0	$\phi/96$	52.1 kHz	83.3 kHz	104 kHz	125 kHz	167 kHz	208 kHz	250 kHz
				1	$\phi/128$	39.1 kHz	62.5 kHz	78.1 kHz	93.8 kHz	125 kHz	156 kHz	188 kHz
1		0	0	$\phi/160$	31.3 kHz	50.0 kHz	62.5 kHz	75.0 kHz	100 kHz	125 kHz	150 kHz	
			1	$\phi/200$	25.0 kHz	40.0 kHz	50.0 kHz	60.0 kHz	80.0 kHz	100 kHz	120 kHz	
		1	0	$\phi/224$	22.3 kHz	35.7 kHz	44.6 kHz	53.5 kHz	71.4 kHz	89.3 kHz	107 kHz	
			1	$\phi/256$	19.5 kHz	31.3 kHz	39.1 kHz	46.9 kHz	62.5 kHz	78.1 kHz	93.8 kHz	

Note: \* Outside the I<sup>2</sup>C bus interface specifications (standard mode: max. 100 kHz; high-speed mode: max. 400 kHz)



### 17.3.5 I<sup>2</sup>C Bus Control Register (ICCR)

ICCR consists of the control bits and interrupt request flags of the I<sup>2</sup>C bus interface.

Bit	Bit Name	Initial Value	R/W	Description
7	ICE	0	R/W	<p>I<sup>2</sup>C Bus Interface Enable</p> <p>0: This module is stopped and disconnected from the SCL and SDA pins. SAR and SARX can be accessed.</p> <p>1: This module can perform transfer and reception, they are connected to the SCL and SDA pins, and the I<sup>2</sup>C bus can be driven. ICMR and ICDR can be accessed.</p>
6	IEIC	0	R/W	<p>I<sup>2</sup>C Bus Interface Interrupt Enable</p> <p>When this bit is set to 1, IRIC interrupts are enabled.</p>

Bit	Bit Name	Initial Value	R/W	Description
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	Transmit/Receive Select 00: Slave receive mode 01: Slave transmit mode 10: Master receive mode 11: Master transmit mode Both these bits will be cleared by hardware when they lose in a bus conflict in master mode of the I <sup>2</sup> C bus format. In slave receive mode, the $\overline{R/W}$ bit in the first frame immediately after the start condition automatically sets these bits in receive mode or transmit mode by hardware. The settings can be made again for the bits that were set/cleared by hardware, by reading these bits. When the TRS bit is intended to change during a transfer, the bit will not be switched until data transfer ends. [MST clearing conditions] 1. When 0 is written by software 2. When lost in bus conflict in I <sup>2</sup> C bus format master mode [MST setting conditions] 1. When 1 is written by software (for MST clearing condition 1) 2. When 1 is written in MST after reading MST = 0 (for MST clearing condition 2) [TRS clearing conditions] 1. When 0 is written by software (except for TRS setting condition 3) 2. When 0 is written in TRS after reading TRS = 1 (for TRS setting condition 3) 3. When lost in bus conflict in I <sup>2</sup> C bus format master mode [TRS setting conditions] 1. When 1 is written by software (except for TRS clearing condition 3) 2. When 1 is written in TRS after reading TRS = 0 (for TRS clearing condition 3) 3. When 1 is received for the $\overline{R/W}$ bit value of the first frame in I <sup>2</sup> C bus format slave mode

Bit	Bit Name	Initial Value	R/W	Description
3	ACKE	0	R/W	<p>Acknowledge Bit Decision Selection</p> <p>0: The value of the received acknowledge bit is ignored, and continuous transfer is performed. The value of the received acknowledge bit is not indicated by the ACKB bit in ICSR, which is always 0.</p> <p>1: If the received acknowledge bit is 1, continuous transfer is halted.</p> <p>Depending on the receiving device, the acknowledge bit may be significant, in indicating completion of processing of the received data, for instance, or may be fixed at 1 and have no significance.</p>
2	BBSY	0	R/W	Bus Busy
0	SCP	1	W	<p>Start Condition/Stop Condition Prohibit</p> <p>In master mode:</p> <p>Writing 0 in BBSY and 0 in SCP: A stop condition is issued</p> <p>Writing 1 in BBSY and 0 in SCP: A start condition and a restart condition are issued</p> <p>In slave mode:</p> <p>Writing to the BBSY flag is disabled.</p> <p>[BBSY setting condition]</p> <ul style="list-style-type: none"> <li>When the SDA level changes from high to low under the condition of SCL = high, assuming that the start condition has been issued.</li> </ul> <p>[BBSY clearing condition]</p> <ul style="list-style-type: none"> <li>When the SDA level changes from low to high under the condition of SCL = high, assuming that the start condition has been issued.</li> </ul> <p>To issue a start/stop condition, use the MOV instruction. The I<sup>2</sup>C bus interface must be set in master transmit mode before the issue of a start condition. Set MST to 1 and TRS to 1 before writing 1 in BBSY and 0 in SCP. The BBSY flag can be read to check whether the I<sup>2</sup>C bus (SCL, SDA) is busy or free.</p> <p>The SCP bit is always read as 1. If 0 is written, the data is not stored.</p>

Bit	Bit Name	Initial Value	R/W	Description
1	IRIC	0	R/(W)*	<p>I<sup>2</sup>C Bus Interface Interrupt Request Flag</p> <p>Indicates that the IIC module has issued an interrupt request to the CPU.</p> <p>This flag is set at different times depending on the FS bit in SAR, the FSX bit in SARX, and the WAIT bit in ICMR. For details, see section 17.5.6, IRIC Setting Timing and SCL Control. The conditions under which this flag is set also differ depending on the setting of the ACKE bit in ICCR.</p> <p>[Setting conditions]</p> <p>I<sup>2</sup>C bus format master mode:</p> <ul style="list-style-type: none"> <li>• When a start condition is detected in the bus line state after a start condition is issued (when the ICDRE flag is set to 1 because of first frame transmission)</li> <li>• When a wait is inserted between the data and acknowledge bit when the WAIT bit is 1 (fall of the 8th transmit/receive clock)</li> <li>• At the end of data transfer (rise of the 9th transmit/receive clock without no waits)</li> <li>• When a slave address is received after bus arbitration is lost (first frame after start condition)</li> <li>• If 1 is received as the acknowledge bit (when the ACKB bit in ICSR is set to 1) when the ACKE bit is 1</li> <li>• When the AL flag is set to 1 after bus arbitration is lost while the ALIE bit is 1</li> </ul> <p>I<sup>2</sup>C bus format slave mode:</p> <ul style="list-style-type: none"> <li>• When the slave address (SVA or SVAX) matches (when the AAS or AASX flag in ICSR is set to 1) and at the end of data transfer up to the subsequent retransmission start condition or stop condition detection (rise of the 9th transmit/receive clock)</li> <li>• When the general call address is detected (when 0 is received as the R/W bit and the ADZ flag in ICSR is set to 1) and at the end of data reception up to the subsequent retransmission start condition or stop condition detection (rise of the 9th receive clock)</li> <li>• If 1 is received as the acknowledge bit (when the ACKB bit in ICSR is set to 1) when the ACKE bit is 1</li> <li>• When a stop condition is detected (when the STOP or ESTP flag in ICSR is set to 1) when the STOPIM bit is 0</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
				<p>Clocked synchronous serial format mode:</p> <ul style="list-style-type: none"> <li>• At the end of data transfer (rise of the 8th transmit/receive clock with serial format selected)</li> <li>• When a start condition is detected with serial format selected</li> </ul> <p>When the ICDRE or ICDRF flag is set to 1 in any operating mode:</p> <ul style="list-style-type: none"> <li>• When a start condition is detected in transmit mode (when a start condition is detected in transmit mode and the ICDRE flag is set to 1)</li> <li>• When data is transferred among the ICDR register and buffer (when data is transferred from the transmit buffer to the shift register in transmit mode and the ICDRE flag is set to 1, or when data is transferred from the shift register to the receive buffer in receive mode and the ICDRF flag is set to 1)</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written in IRIC after reading IRIC = 1</li> <li>• When ICDR is read from or written to by the DTC (This may not function as a clearing condition depending on the situation. For details, see the description of the DTC operation given below.)</li> </ul>

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

When the DTC is used, the IRIC flag is cleared automatically and transfer can be performed continuously without CPU intervention.

When, with the I<sup>2</sup>C bus format selected, the IRIC flag is set to 1 and an interrupt is generated, other flags must be checked in order to identify the source that set the IRIC flag to 1. Although each source has a corresponding flag, caution is needed at the end of a transfer.

When the ICDRE or ICDRF flag is set, the IRTR flag may or may not be set. The IRTR flag (the DTC start request flag) is not set at the end of a data transfer up to detection of a retransmission start condition or stop condition after a slave address (SVA) or general call address match in I<sup>2</sup>C bus format slave mode.

Even when the IRIC flag and IRTR flag are set, the ICDRE or ICDRF flag may not be set. The IRIC and IRTR flags are not cleared at the end of the specified number of transfers in continuous transfer using the DTC. The ICDRE or ICDRF flag is cleared, however, since the specified number of ICDR reads or writes have been completed. Table 17.4 shows the relationship between the flags and the transfer states.

Table 17.4 Flags and Transfer States

Operating Mode	MST	TRS	BBSY	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	ICDRE	ICDRF	State	
Master mode	1	1	0	0	0	0	0↓	0	0↓	0↓	0	0	—	Idle state (flag clearing required)	
	1	1	1↑	0	0	1↑	0	0	0	0	0	1↑	—	Start condition detected	
	1	—	1	0	0	—	0	0	0	0	—	—	—	Wait state	
	1	1	1	0	0	—	0	0	0	0	1↑	—	—	Transmission end (when ACKB = 1 received)	
	1	1	1	0	0	1↑	0	0	0	0	0	1↑	—	Transmission end (when previous state is ICDRE = 0)	
	1	1	1	0	0	—	0	0	0	0	0	0↓	—	Write to ICDR in above state	
	1	1	1	0	0	—	0	0	0	0	0	1	—	Transmission end (when previous state is ICDRE = 1)	
	1	1	1	0	0	—	0	0	0	0	0	0↓	—	Write to ICDR in above state or after start condition is detected	
	1	1	1	0	0	1↑	0	0	0	0	0	0	1↑	—	Data transfer from transmit buffer to shift register (automatic) in above state
	1	0	1	0	0	1↑	0	0	0	0	—	—	1↑	Reception end (when previous state is ICDRF = 0)	
	1	0	1	0	0	—	0	0	0	0	—	—	0↓	Write to ICDR in above state	
	1	0	1	0	0	—	0	0	0	0	—	—	1	Reception end (when previous state is ICDRF = 1)	
	1	0	1	0	0	—	0	0	0	0	—	—	0↓	Write to ICDR in above state	
	1	0	1	0	0	1↑	0	0	0	0	—	—	1↑	Data transfer from shift register to receive buffer (automatic) in above state	
	0↓	0↓	1	0	0	—	0	1↑	0	0	—	—	—	—	Arbitration lost
	1	—	0↓	0	0	—	0	0	0	0	—	—	0↓	—	Stop condition detected
Slave mode	0	0	0	0	0	0	0	0	0	0	0	0	—	Idle state (flag clearing required)	
	0	0	1↑	0	0	0	0↓	0	0	0	0	1↑	—	Start condition detected	
	0	1↑/0*1	1	0	0	0	0	—	1↑	0	0	1	1↑	SAR match by first frame (SARX ≠ SAR)	
	0	0	1	0	0	0	0	—	1↑	1↑	0	1	1↑	General call address match by first frame (SARX ≠ H'00)	
	0	1↑/0*1	1	0	0	1↑	1↑	—	0	0	0	1	1↑	SARX match by first frame (SAR ≠ SARX)	

Operating Mode	MST	TRS	BBSY	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	ICDRE	ICDRF	State
Slave mode	0	1	1	0	0	—	—	—	—	0	1↑	—	—	Transmission end (when ACKB = 1 received)
	0	1	1	0	0	1↑/0* <sup>2</sup>	—	—	—	0	0	1↑	—	Transmission end (when previous state is ICDRE = 0)
	0	1	1	0	0	—	—	0↓	0↓	0	0	0↓	—	Write to ICDR in above state
	0	1	1	0	0	—	—	—	—	0	0	1	—	Transmission end (when previous state is ICDRE = 1)
	0	1	1	0	0	—	—	0↓	0↓	0	0	0↓	—	Write to ICDR in above state or after start condition is detected
	0	1	1	0	0	1↑/0* <sup>2</sup>	—	0	0	0	0	1↑	—	Data transfer from transmit buffer to shift register (automatic) in above state
	0	0	1	0	0	1↑/0* <sup>2</sup>	—	—	—	—	—	—	1↑	Reception end (when previous state is ICDRF = 0)
	0	0	1	0	0	—	—	0↓	0↓	0↓	—	—	0↓	Write to ICDR in above state
	0	0	1	0	0	—	—	—	—	—	—	—	1	Reception end (when previous state is ICDRF = 1)
	0	0	1	0	0	—	—	0↓	0↓	0↓	—	—	0↓	Write to ICDR in above state
	0	0	1	0	0	1↑/0* <sup>2</sup>	—	0	0	0	—	—	1↑	Data transfer from shift register to receive buffer (automatic) in above state
	0	—	0↓	1↑/0* <sup>3</sup>	0/1↑* <sup>3</sup>	—	—	—	—	—	—	0↓	—	Stop condition detected

## Legend:

0: Retains 0

1: Retains 1

—: Retains the previous state

0↓: Cleared to 0

1↑: Set to 1

- Notes:
1. Set to 1 when 1 is received as a R $\bar{W}$  bit following an address.
  2. Set to 1 when the AASX bit is set to 1.
  3. When ESTP = 1, STOP is 0, or when STOP = 1, ESTP is 0.

### 17.3.6 I<sup>2</sup>C Bus Status Register (ICSR)

ICSR consists of status flags. Also see table 17.4.

Bit	Bit Name	Initial Value	R/W	Description
7	ESTP	0	R/(W)*	<p>Error Stop Condition Detection Flag</p> <p>This bit is valid in I<sup>2</sup>C bus format slave mode.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a stop condition is detected during frame transfer.</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written in ESTP after reading ESTP = 1</li> <li>When the IRIC flag in ICCR is cleared to 0</li> </ul>
6	STOP	0	R/(W)*	<p>Normal Stop Condition Detection Flag</p> <p>This bit is valid in I<sup>2</sup>C bus format slave mode.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a stop condition is detected during frame transfer.</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written in STOP after reading STOP = 1</li> <li>When the IRIC flag in ICCR is cleared to 0</li> </ul>
5	IRTR	0	R/(W)*	<p>I<sup>2</sup>C Bus Interface Continuous Transfer Interrupt Request Flag</p> <p>Indicates that the IIC module has issued an interrupt request to the CPU, and the source is completion of reception/transmission of one frame in continuous transmission/reception for which DTC activation is possible. When the IRTR flag is set to 1, the IRIC flag is also set to 1 at the same time.</p> <p>[Setting condition in I<sup>2</sup>C bus format slave mode]</p> <ul style="list-style-type: none"> <li>When the ICDRE or ICDRF flag in ICXR is set to 1 when AASX = 1</li> </ul> <p>[Setting condition in I<sup>2</sup>C bus format modes other than slave mode]</p> <ul style="list-style-type: none"> <li>When the ICDRE or ICDRF flag in ICXR is set to 1</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written in IRTR after reading IRTR = 1</li> <li>When the IRIC flag in ICCR is cleared to 0 (while ICE = 1 or ICXE = 1)</li> </ul>



Bit	Bit Name	Initial Value	R/W	Description
4	AASX	0	R/(W)*	<p>Second Slave Address Recognition Flag</p> <p>In I<sup>2</sup>C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition matches bits SVAX6 to SVAX0 in SARX.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the second slave address is detected in slave receive mode and FSX = 0 in SARX</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is written in AASX after reading AASX = 1</li> <li>When a start condition is detected</li> <li>In master mode</li> </ul>
3	AL	0	R/(W)*	<p>Arbitration Lost Flag</p> <p>Indicates that arbitration was lost in master mode.</p> <p>[Setting conditions]</p> <p>When ALSL = 0</p> <ul style="list-style-type: none"> <li>If the internal SDA and SDA pin disagree at the rise of SCL in master transmit mode</li> <li>If the internal SCL line is high at the fall of SCL in master transmit mode</li> </ul> <p>When ALSL = 1</p> <ul style="list-style-type: none"> <li>If the internal SDA and SDA pin disagree at the rise of SCL in master transmit mode</li> <li>If the SDA pin is driven low by another device before the I<sup>2</sup>C bus interface drives the SDA pin low, after the start condition instruction was executed in master transmit mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICDR is written to (transmit mode) or read from (receive mode)</li> <li>When 0 is written in AL after reading AL = 1</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	AAS	0	R/(W)*	<p>Slave Address Recognition Flag</p> <p>In I<sup>2</sup>C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition matches bits SVA6 to SVA0 in SAR, or if the general call address (H'00) is detected.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the slave address or general call address is detected in slave receive mode and FS = 0 in SAR</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICDR is written to (transmit mode) or read from (receive mode)</li> <li>When 0 is written in AAS after reading AAS = 1</li> <li>In master mode</li> </ul>
1	ADZ	0	R/(W)*	<p>General Call Address Recognition Flag</p> <p>In I<sup>2</sup>C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition is the general call address (H'00).</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the general call address is detected in slave receive mode and FS = 0 in SAR or FSX = 0 in SARX</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICDR is written to (transmit mode) or read from (receive mode)</li> <li>When 0 is written in ADZ after reading ADZ = 1</li> <li>In master mode</li> </ul> <p>If a general call address is detected while FS = 1 and FSX = 0, the ADZ flag is set to 1; however, the general call address is not recognized (AAS flag is not set to 1).</p>

Bit	Bit Name	Initial Value	R/W	Description
0	ACKB	0	R/W	<p>Acknowledge Bit</p> <p>Stores acknowledge data.</p> <p>Transmit mode:</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When 1 is received as the acknowledge bit when ACKE = 1 in transmit mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When 0 is received as the acknowledge bit when ACKE = 1 in transmit mode</li> <li>When 0 is written to the ACKE bit</li> </ul> <p>Receive mode:</p> <p>0: Returns 0 as acknowledge data after data reception</p> <p>1: Returns 1 as acknowledge data after data reception</p> <p>When this bit is read, the value loaded from the bus line (returned by the receiving device) is read in transmission (when TRS = 1). In reception (when TRS = 0), the value set by internal software is read.</p> <p>When this bit is written, acknowledge data that is returned after receiving is rewritten regardless of the TRS value. If a flag in ICSR is written using bit manipulation instructions, the acknowledge data should be re-set since the acknowledge data setting is rewritten by the ACKB bit reading value.</p> <p>Write 0 in the ACKE bit to clear the ACKB flag to 0, before transmission is ended and a stop condition is issued in master mode, or before transmission is ended and SDA is released to issue a stop condition by a master device.</p>

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

### 17.3.7 IIC Operation Reservation Adapter Control Register (ICCRX)

ICCRX controls the operation of the IIC operation reservation adapter.

Bit	Bit Name	Initial Value	R/W	Description
7	ICXE	0	R/W	<p>IIC Operation Reservation Adapter Enable</p> <p>Selects whether to control the conventional IIC module or enable and control the IIC operation reservation adapter.</p> <p>0: Directly controls the conventional IIC module; disables the IIC operation reservation adapter</p> <p>1: Enables the IIC operation reservation adapter; restricts direct control of the conventional IIC module</p> <p>When this bit is set to 1, the SCL and SDA pins can drive the I<sup>2</sup>C bus, similar to when the ICE bit in ICCR is set to 1.</p>
6	CRIC	0	R/W	<p>Command Request Interrupt Enable</p> <p>Enables or disables the IIC operation reservation command execution end/next command write request interrupt.</p> <p>0: Disables the command write request interrupt</p> <p>1: Enables the command write request interrupt</p>
5	MRIC	0	R/W	<p>Master Mode Transmission/Reception Interrupt Enable</p> <p>0: Disables the master mode transmission/reception interrupt (MRREQ, MTREQ)</p> <p>1: Enables the master mode transmission/reception interrupt (MRREQ, MTREQ)</p>
4	SRIC	0	R/W	<p>Slave Mode Transmission/Reception Interrupt Enable</p> <p>0: Disables the slave mode transmission/reception interrupt (SRREQ, STREQ)</p> <p>1: Enables the slave mode transmission/reception interrupt (SRREQ, STREQ)</p>
3	BBSYX	0	R	<p>Bus Busy X</p> <p>0: Bus is released</p> <p>1: Bus is occupied</p>
2	—	0	R	<p>Reserved</p> <p>This bit is always read as 0 and cannot be modified.</p>

Bit	Bit Name	Initial Value	R/W	Description
1	AASHIT	0	R	Slave Address Match 0: In master mode or slave address does not match 1: Slave address matches
0	ACKBX	0	R	Acknowledge Bit Transmission Reserve 0: Reserves transmission of acknowledge bit 0 1: Reserves transmission of acknowledge bit 1
3	CLR3	—	W	Clear 3 to 0  Writing B'0101 to these bits with an MOV instruction initializes the internal latch circuit and state machine of the conventional IIC module and the IIC operation reservation adapter. The appropriate control bits of the conventional IIC module and the IIC operation reservation adapter must be initialized so as to deactivate the IIC module.  When bits 7 to 4 in this register are set/cleared using a bit manipulation instruction, the contents in BBSYX and AASHIT are written to the CLR3 to CLR0 bits by read/modify/write operation. However, since bit 2 is always read as 0, clearing operation cannot be executed.
2	CLR2	—	W	
1	CLR1	—	W	
0	CLR0	—	W	

### 17.3.8 IIC Operation Reservation Adapter Status Register A (ICSRA)

ICSRA monitors the operating status of the IIC module.

Bit	Bit Name	Initial Value	R/W	Description
7	SDAO	1	R	SDA Output Value Monitors the output value from the SDA pin.
6	SCLO	1	R	SCL Output Value Monitors the output value from the SCL pin.
5	SDAI	—	R	SDA Input Level Monitors the input level to the SDA pin.
4	SCLI	—	R	SCL Input Level Monitors the input level to the SCL pin.

Bit	Bit Name	Initial Value	R/W	Description
3	MSTX	0	R	Master/Slave State X
2	TRSX	0	R	Transmit/Receive State X 00: Slave receive mode 01: Slave transmit mode 10: Master receive mode 11: Master transmit mode These bits are automatically set by an operation reservation command.
1	WAITX	0	R	Wait Insertion Bit X This bit is valid only in master mode with the I <sup>2</sup> C bus format. This bit is automatically set by an operation reservation command. 0: Data and the acknowledge bit are transferred consecutively with no wait inserted. 1: After the fall of the clock for the final data bit, a wait state begins (with SCL at the low level).
0	ACKXE	0	R	Acknowledge Bit Decision Selection X 0: The value of the acknowledge bit is ignored, and continuous transfer is performed. The value of the received acknowledge bit is not indicated by the ACKBX bit in ICCRX, which is always 0. 1: If the acknowledge bit is 1, continuous transfer is halted. This bit is automatically set by an operation reservation command.

### 17.3.9 IIC Operation Reservation Adapter Status Register B (ICSRB)

ICSRB monitors operating state transitions and error status of the IIC operation reservation adapter.

Bit	Bit Name	Initial Value	R/W	Description														
7	CREQ	0	R/(W)*	<p>Operation Reservation Command Write Request Interrupt Flag</p> <p>0: No command write request is generated 1: Command write request is generated</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is read from or written to</li> <li>When 0 is written to CREQ after reading CREQ = 1</li> </ul> <p>[Setting condition]</p> <table border="1"> <thead> <tr> <th>CREQ Generation Source</th> <th>Generation Timing</th> </tr> </thead> <tbody> <tr> <td>Automatic command transition after slave address match</td> <td>Rise of 9th clock</td> </tr> <tr> <td>ACKB = 1 received (NACK = 1) when the acknowledge bit is enabled in transmit mode</td> <td>Rise of 9th clock</td> </tr> <tr> <td>Stop condition detected (STOPIMX = 0 while STOP = 1 or ABRT = 1)</td> <td>Stop condition detected</td> </tr> <tr> <td>Arbitration lost (ALST = 1)</td> <td>Always</td> </tr> <tr> <td>Timeover generated (TOVR = 1)</td> <td>Always</td> </tr> <tr> <td>Undefined command is written</td> <td>Always</td> </tr> </tbody> </table>	CREQ Generation Source	Generation Timing	Automatic command transition after slave address match	Rise of 9th clock	ACKB = 1 received (NACK = 1) when the acknowledge bit is enabled in transmit mode	Rise of 9th clock	Stop condition detected (STOPIMX = 0 while STOP = 1 or ABRT = 1)	Stop condition detected	Arbitration lost (ALST = 1)	Always	Timeover generated (TOVR = 1)	Always	Undefined command is written	Always
CREQ Generation Source	Generation Timing																	
Automatic command transition after slave address match	Rise of 9th clock																	
ACKB = 1 received (NACK = 1) when the acknowledge bit is enabled in transmit mode	Rise of 9th clock																	
Stop condition detected (STOPIMX = 0 while STOP = 1 or ABRT = 1)	Stop condition detected																	
Arbitration lost (ALST = 1)	Always																	
Timeover generated (TOVR = 1)	Always																	
Undefined command is written	Always																	
6	CERR	0	R/(W)*	<p>Operation Reservation Command Error Flag</p> <p>0: No command error is generated 1: Command error is generated</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to CERR after reading CERR = 1</li> </ul> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>When arbitration loss is generated</li> <li>When a timeover is generated</li> <li>When a start/stop condition is detected during transfer</li> <li>When NAK is received</li> </ul>														

Bit	Bit Name	Initial Value	R/W	Description
5	STOP	0	R/(W)*	<p>Stop Condition Detection</p> <p>0: No stop condition is detected</p> <p>1: Stop condition is detected</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to STOP after reading STOP = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a stop condition is detected</li> </ul>
4	ABRT	0	R	<p>Abort</p> <p>0: Normal end</p> <p>1: Abort</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to ABRT after reading ABRT = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a start/stop condition is detected during data transfer</li> </ul>
3	ALST	0	R/(W)*	<p>Arbitration Lost</p> <p>0: Normal operation</p> <p>1: Arbitration loss is generated</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to ALST after reading ALST = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When arbitration loss is generated in master mode <ul style="list-style-type: none"> <li>(1) Internal SDA disagrees with the SDA pin level at rise of SCL</li> <li>(2) Internal SDA is high when the start condition is detected</li> </ul> </li> </ul>



Bit	Bit Name	Initial Value	R/W	Description
2	DERR	0	R/(W)*	<p>Transmit Data Disagree Error</p> <p>0: Normal operation 1: Transmit data disagree</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to DERR after reading DERR = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The DERR bit is set to 1 when a stop condition is detected in slave transmit mode. When the STOP and DERR bits are set to 1 simultaneously, the DERR value is ignored because this is normal operation and not an error.</li> <li>When internal transmit data disagrees with the SDA pin level during transmission</li> </ul>
1	TOVR	0	R/(W)*	<p>Timeover</p> <p>0: Normal operation 1: A timeover is generated</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to TOVR after reading TOVR = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a timeover is generated</li> </ul>
0	NACK	0	R/(W)*	<p>NACK Receive</p> <p>0: Normal operation 1: NACK is received</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>When ICCMD is written to</li> <li>When 0 is written to NACK after reading NACK = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When NACK is received after data transmission is completed (ACKBX = 1 received)</li> </ul>

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

### 17.3.10 IIC Operation Reservation Adapter Status Register C (ICSRC)

ICSRC monitors the transmission/reception status of the IIC operation reservation adapter. See figure 17.3 for details on the TDRE, SDRF, and RDRF bits.

Bit	Bit Name	Initial Value	R/W	Description
7	MTREQ	0	R/(W)*1	<p>Master Mode Transmit Data Write Request Interrupt Flag</p> <p>0: No transmit data write request is generated in master mode</p> <p>1: A transmit data write request is generated and an interrupt is requested in master mode</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When ICDRX is written to</li> <li>• When ICCMD is written to</li> <li>• When 0 is written to MTREQ after reading MTREQ = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When data transmission is completed in master mode</li> </ul>
6	MRREQ	0	R/(W)*1	<p>Master Mode Receive Data Read Request Interrupt Flag</p> <p>0: No receive data read request is generated in master mode</p> <p>1: A receive data read request is generated and an interrupt is requested in master mode</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When ICDRX is written to</li> <li>• When ICCMD is written to</li> <li>• When 0 is written to MRREQ after reading MRREQ = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When data reception is completed in master mode</li> </ul>

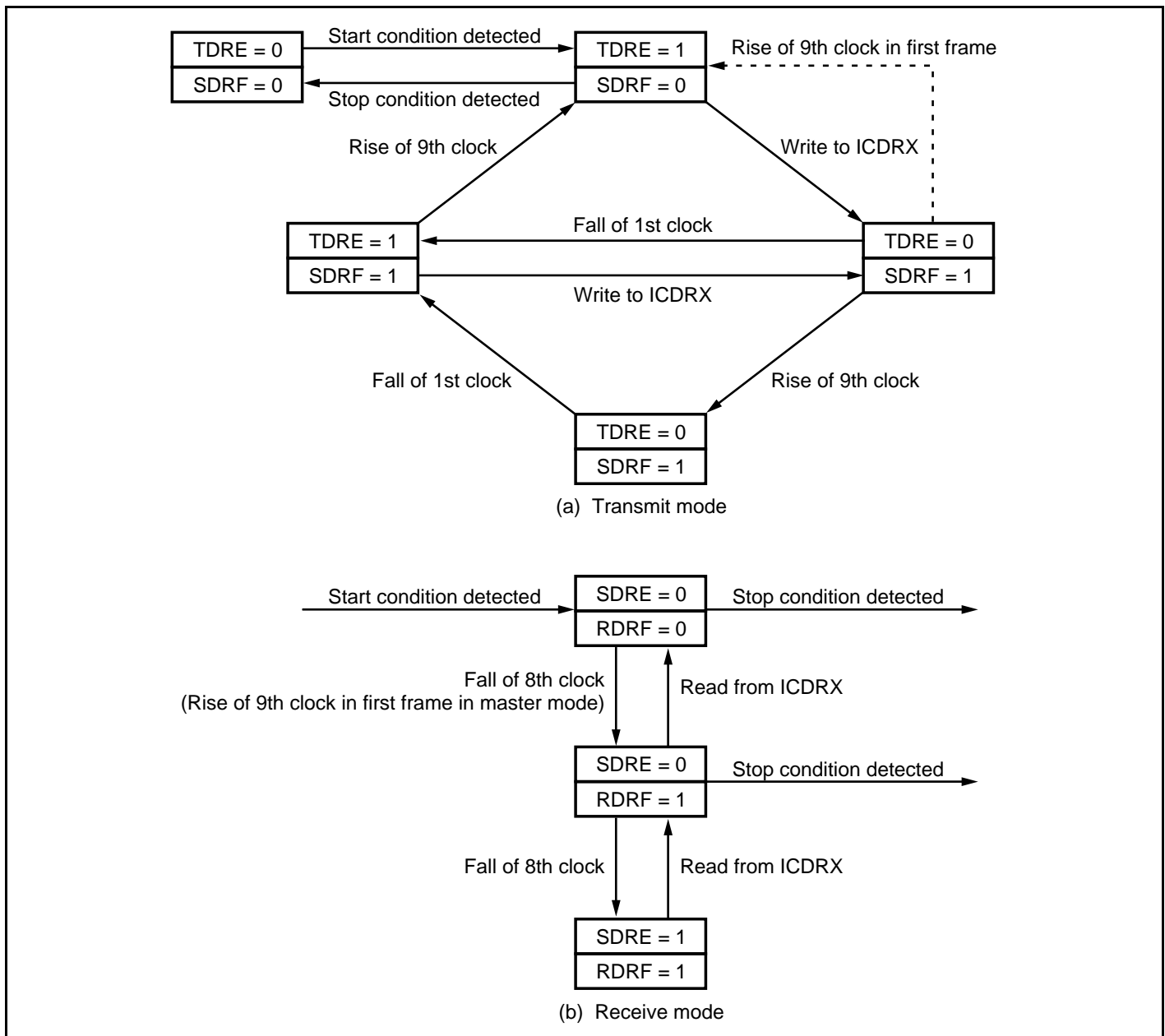
Bit	Bit Name	Initial Value	R/W	Description
5	STREQ	0	R/(W)*1	<p>Slave Mode Transmit Data Write Request Interrupt Flag:</p> <p>0: No transmit data write request is generated in slave mode</p> <p>1: A transmit data write request is generated and an interrupt is requested in slave mode</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When ICDRX is written to</li> <li>• When ICCMD is written to</li> <li>• When 0 is written to STREQ after reading STREQ = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When data transmission is completed in slave mode</li> </ul>
4	SRREQ	0	R/(W)*1	<p>Slave Mode Receive Data Read Request Interrupt Flag</p> <p>0: No receive data read request is generated in slave mode</p> <p>1: A receive data read request is generated and an interrupt is requested in slave mode</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When ICDRX is written to</li> <li>• When ICCMD is written to</li> <li>• When 0 is written to SRREQ after reading SRREQ = 1</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>• When data reception is completed in slave mode</li> </ul>
3	MASX	0	R	<p>Master Mode Address Select X</p> <p>0: DTC cannot be activated</p> <p>Address disagrees in slave mode or AAS and ADZ addresses agree</p> <p>1: DTC can be activated</p> <p>In master mode, or AASX address agrees in slave mode</p>

Bit	Bit Name	Initial Value	R/W	Description
2	TDRE	0	R	<p>Transmit Data Register Empty</p> <p>0: Transmit buffer (ICDRT) contains transmit data            1: Transmit buffer (ICDRT) contains no transmit data;            write to ICDRT is possible</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When start condition is detected (when ICDRX is not written to before the start condition is detected)<sup>*2</sup></li> <li>• When transfer of first frame (address + R/W) ends (rise of 9th clock)<sup>*2</sup></li> <li>• When transmission of second or subsequent frames starts (fall of 1st clock) (except for when transferring the last byte in DTC transfer)<sup>*2</sup></li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When ICDRX is written to<sup>*2</sup></li> <li>• When ACKB = 1 is received</li> <li>• When stop condition is detected</li> </ul> <p>This bit is enabled during transmission by the IIC operation reservation adapter.</p> <p>For transmission/reception with the conventional method, see the description of the ICDRE and ICDRF flags in ICDR.</p>

Bit	Bit Name	Initial Value	R/W	Description
1	SDRF	0	R	<p>Shift Data Register Full</p> <p>0: Shift register (ICDRS) contains no receive data remaining to be read or data remaining to be transmitted</p> <p>1: Shift register (ICDRS) contains receive data remaining to be read or data remaining to be transmitted</p> <p>[Setting condition in transmit mode]</p> <ul style="list-style-type: none"> <li>When ICDRX is written to, or when data transmission ends (rise of 9th clock) with the next transmit data set in ICDRT (TDRE = 0)<sup>*2</sup></li> </ul> <p>[Clearing conditions in transmit mode]</p> <ul style="list-style-type: none"> <li>When data transmission ends (rise of 9th clock) with the next transmit data not set in ICDRT (TDRE = 1)<sup>*2</sup></li> <li>When stop condition is detected</li> <li>When receive mode is entered</li> <li>Arbitration lost</li> </ul> <p>[Setting condition in receive mode]</p> <ul style="list-style-type: none"> <li>When data reception ends (fall of 8th clock) with RDRF = 1<sup>*3</sup></li> </ul> <p>[Clearing conditions in receive mode]</p> <ul style="list-style-type: none"> <li>When ICDRX is read from</li> <li>When start condition is detected</li> <li>When transmit mode is entered</li> </ul> <p>This bit is enabled during transmission or reception by the IIC operation reservation adapter.</p>

Bit	Bit Name	Initial Value	R/W	Description
0	RDRF	0	R	<p>Receive Data Register Full</p> <p>0: Receive buffer (ICDRR) contains no receive data            1: Receive buffer (ICDRR) contains receive data; read from ICDRR is possible</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When transmission of first frame (address + R/W = 1) in master mode ends (rise of 9th clock)</li> <li>• When reception of second or subsequent frames ends (fall of 8th clock)*<sup>3</sup></li> <li>• When ICDRX is read from with receive data in the shift register (SDRF = 1) in receive mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When ICDRX is read from with no receive data in the shift register (SDRF = 0) in receive mode</li> <li>• When start condition is detected</li> </ul> <p>This bit is enabled during reception by the IIC operation reservation adapter.</p>

- Notes:
1. Only 0 can be written to clear the flag.
  2. Address disagree in master mode or slave mode transmission.
  3. Address (including general call address) match in master mode or slave mode reception.



**Figure 17.3 State Transitions of TDRE, SDRF, and RDRF Bits**

### 17.3.11 IIC Operation Reservation Adapter Data Register (ICDRX)

ICDRX is an 8-bit register identical to ICDR. When this register is accessed for read, the contents in the receive buffer (ICDRR) are read out; and when this register is accessed for write, the write data is written to the transmit buffer (ICDRT). However, the IIC module operates in the way defined by the operation reservation adapter when this register is read from or written to.

When the ICXE bit in ICCRX is set to 1, accesses to ICDR are invalid. The initial value of ICDRX is undefined.

### 17.3.12 IIC Data Shift Register (ICDRS)

ICDRS is an 8-bit read-only register, which is ICDR's readable shift register (ICDRS).

During transfer operation, the ICDRS values change in accordance with the frame bit count. The ICDRS values can be associated with the BBC3 to BBC0 bits in ICCNT by simultaneously reading from ICDRS and ICCNT in words. The ICDRS values read when the BBC3 to BBC0 bits are set to B'1111 match the receive data.

When receive data is to be read with the SCL clock stopped while the ACK/NACK bit is to be selected after reading receive data, read from ICDRS using the above method.

### 17.3.13 IIC Operation Reservation Adapter Count Register (ICCNT)

ICCNT controls monitoring of timeout-related operations of the IIC operation reservation adapter.

#### **Timeout Occurrence Condition:**

In transmission/reception operation using the IIC operation reservation adapter, a timeout occurs when the interrupt flag is not set and the SCL pin has not changed for a period exceeding the time specified by the CNTS1 and CNTS0 bits. When a timeout occurs, the TOVR and CERR flags in ICSR are set to 1 and a command request interrupt (CREQ) request occurs.

#### **Timeout Counter Clear Conditions:**

- At a reset
- When ICXE = 0 in ICCRX (IIC operation reservation adapter disabled)
- When CNTE = 0 (timeout counter stopped)
- When BBSYX = 0 in ICCRX (bus released state)
- When a bit among CREQ in ICSR, and MTREQ, MRREQ, STREQ, and SRREQ in ICSRC is set to 1 (interrupt is requested)
- When a rising edge or falling edge is input to the SCL pin



Bit	Bit Name	Initial Value	R/W	Description
7	CNTE	0	R/W	<p>Timeout Counter Enable</p> <p>Starts or stops the timeout counter.</p> <p>0: Stops the timeout counter and clears the internal counter</p> <p>1: Operates the timeout counter</p>
6	STOPIX	0	R/W	<p>Stop Condition Interrupt Source Mask X</p> <p>Selects whether to enable CREQ interrupt requests by stop conditions.</p> <p>0: CREQ interrupt requests by stop conditions are enabled</p> <p>1: CREQ interrupt requests by stop conditions are disabled</p>
5	CNTS1	0	R/W	Counter Select
4	CNTS0	0	R/W	<p>These bits specify the number of clock cycles for the timeout counter. The clock is selected by the IICX1 and IICX0 bits in STCR and the CKS2 to CKS0 bits in ICMR.</p> <p>00: 32 clock cycles</p> <p>01: 34 clock cycles</p> <p>10: 128 clock cycles</p> <p>11: 256 clock cycles</p>
3	BBC3	1	R	Frame Bit Count
2	BBC2	1	R	<p>These bits count the frame bit at each rising edge of SCL. These bits are cleared to B'0000 at the first rising edge, and are incremented by 1 at each rising edge. After being incremented to B'0111 at the eighth rising edge, these bits are initialized to B'1111 at the next rising edge (for acknowledge).</p>
1	BBC1	1	R	
0	BBC0	1	R	

### 17.3.14 IIC Operation Reservation Adapter Command Register (ICCMD)

ICCMD is an 8-bit readable/writable register that specifies an IIC operation reservation adapter command. When the reserved operation is completed, ICCMD may be automatically modified to contain the next related command. For details on commands and operations, see section 17.4, IIC Operation Reservation Adapter.

When the ICXE bit in ICCRX is set to 1, ICCMD is automatically set to H'A0. When the ICXE bit in ICCRX is cleared to 0, ICCMD is cleared to H'00.

If a value not defined as an operation reservation command is written to ICCMD when the ICXE bit is set to 1, the CERR bit in ICSR is set to 1 and a CREQ interrupt request is generated. In this case, ICCMD is reset to H'A0. When the ICXE bit is cleared to 0, no value can be written to ICCMD.

## 17.4 IIC Operation Reservation Adapter

### 17.4.1 Restrictions on Accessing IIC Registers

There are restrictions when accessing registers other than those related to the IIC operation reservation adapter when the IIC operation reservation adapter is enabled (ICXE = 1 in ICCRX), as shown in table 17.5.

**Table 17.5 Restrictions on Accessing IIC Registers**

Register	Bit	Bit Name	R/W	Description
ICDR	7 to 0	—	R/W	Writing is disabled. Reading/writing to this register does not initiate any data transfer. Writing has no affect on the bits.
SAR	7 to 1	SVA6 to SVA0	R/W	Write the slave address.
	0	FS	R/W	Select the I <sup>2</sup> C bus format.
SARX	7 to 1	SVAX6 to SVAX0	R/W	Write the second slave address as required.
	0	FSX	R/W	Select the I <sup>2</sup> C bus format.
ICMR	7	MLS	R/W	Select MSB-first.
	6	WAIT	R/W	The value written to this bit is ignored because the function of this bit is replaced by ICCMD. The function of this bit can be monitored by the WAITX bit in ICSRA.
	5 to 3	CKS2 to CKS0	R/W	Select the transfer rate.
	2 to 0	BC2 to BC0	R/W	Set to B'000 (9 bits).
	7	ICE	R/W	—
ICCR	6	IEIC	R/W	Even when set to 1, the conventional interrupt is ignored.
	5	MST	R/W	The values written to these bits are ignored because the functions of these bits are replaced by ICCMD. The functions of these bits can be monitored by the MSTX, TRSX, and ACKXE bits in ICSRA.
	4	TRS	R/W	
	3	ACKE	R/W	
	2	BBSY	R/W	Writing B'10 or B'00 is ignored.
	1	IRIC	R/W	—
	0	SCP	W	Writing B'10 or B'00 is ignored.
	ICSR	7	ESTP	R/(W)
6		STOP	R/(W)	
5		IRTR	R/(W)	
4		AASX	R/(W)	
3		AL	R/(W)	
2		AAS	R/(W)	
1		ADZ	R/(W)	
0		ACKB	R/W	The value written to this bit is ignored because the function of this bit when written to is replaced by the ACKXB bit in ICCRX.

## 17.4.2 Operation Reservation Commands

Table 17.6 lists the operation reservation commands that can be set in ICCMD. If values other than H'A0 to H'AF, and H'C0 to H'CF are set, H'A0 is automatically set as the command.

The operation reservation commands reserve various operations such as start condition issuance, stop condition issuance, data transmission (including acknowledge (ACK)/non-acknowledge (NAK) judgment), and data reception (including ACK/NAK transmission and stop condition issuance). Since 16 bits can be simultaneously written to ICCMD and ICDRX, a series of required operations can be written as the transfer data (data + command) for data transmission.

Table 17.7 shows the changes in interrupt flag status and automatic command transition when the operation reservation command is completed. Each IIC data set consists of the first frame (address + read/write) and the subsequent frames (data). Each frame consists of 8-bit data and ACK/NAK; transmission of ACK or NAK and enabling or disabling NAK judgment must be specified using the command. The corresponding commands are updated by a command request interrupt (CREQ) issued after the operation reservation command is completed or automatically updated by the automatic command transition function. The commands corresponding to the first frame allow automatic transition. The commands corresponding to the subsequent frames are divided into two types; some are used for continuous data, which do not allow automatic transition, and others are used for the last data or a stop condition, which allow automatic transition. A command also allows automatic transition when arbitration loss occurs.

The commands for continuous data generate a data transfer request interrupt (MRREQ, MTREQ, SRREQ, or STREQ) after the operation specified by the command is completed (one byte is processed). To process the last data, update the command at interrupt generation.

**Table 17.6 Operation Reservation Commands**

<b>Mode</b>	<b>Command</b>	<b>Description</b>	
Initial value	00	IIC operation reservation adapter disabled	
Slave	A0	Waits for address reception (ACK transmission/NACK enabled, reception using single buffer)	
	A1	Waits for address reception (NAK transmission/NACK disabled, reception using single buffer)	
	A2	Waits for address reception (ACK transmission/NACK enabled, reception using double buffer)	
	A3	Waits for address reception (NAK transmission/NACK disabled, reception using double buffer)	
	A4	Waits for slave reception, reserves ACK transmission, receives using single buffer	
	A5	Waits for slave reception, reserves NAK transmission, receives using single buffer	
	A6	Waits for slave reception, reserves ACK transmission, receives using double buffer	
	A7	Waits for slave reception, reserves NAK transmission, receives using double buffer	
	A8, AA	Reserves slave transmission, automatically stops at NAK reception	
	A9, AB	Reserves slave transmission, disables NAK	
	AC to AF	Reserved (setting prohibited)	
	C0 to C3	Reserved (setting prohibited)	
	Master	C4	Reserves master reception, reserves ACK transmission, receives using single buffer
		C5	Reserves master reception, reserves NAK transmission, receives using single buffer
C6		Reserves master reception, reserves ACK transmission, receives using double buffer	
C7		Reserves master reception, reserves NAK transmission, receives using double buffer	
C8		Issues a start condition, reserves master transmission, enables NAK	
C9		Issues a start condition, reserves master transmission, disables NAK	
CA		Issues a start condition, reserves master transmission, issues a stop condition at NAK reception	
CB		Issues a start condition, reserves master transmission, issues a stop condition at NAK reception	
CC		Issues a retransmission start condition	
CD		Transmits ACK, reserves stop condition issuance	
CE		Transmits NAK, reserves stop condition issuance	
CF		Reserves stop condition issuance	

**Table 17.7 Operation When the Operation Reservation Command Is Completed**

Command	Trigger of Starting Operation	Operation Completion	Automatic Transition Conditions	Transition Destination Commands	Interrupt Flags
A0 to A3	Address reception	Address reception completed	AAS or ADZ = 1, R/W = 0	A4 to A7	CREQ, SRREQ
	Address reception	Address reception completed	AAS or ADZ = 1, R/W = 1	A8 to AB	CREQ, STREQ
	Address reception	Address reception completed	AASX = 1, R/W = 0	A4 to A7	SRREQ
	Address reception	Address reception completed	AASX = 1, R/W = 1	A8 to AB	STREQ
A4	ICDRX read	(ACK transmitted), reception completed	—	—	SRREQ
	ICDRX read	ACK transmitted, stop condition detected	—	A0	CREQ
A5	ICDRX read	(NAK transmitted), reception completed	—	—	SRREQ
	ICDRX read	NAK transmitted, stop condition detected	—	A0	CREQ
A6	ICDRX read	Reception completed, ACK transmitted	—	—	SRREQ
	ICDRX read	Stop condition detected	—	A0	CREQ
A7	ICDRX read	Reception completed, NAK transmitted	—	—	SRREQ
	ICDRX read	Stop condition detected	—	A0	CREQ
A8, AA	ICDRX write	Transmission completed	ACK received	—	STREQ
	ICDRX write	Transmission completed	NAK received	—	CREQ
	ICDRX write	Stop condition detected	—	A0	CREQ
A9, AB	ICDRX write	Transmission completed	—	—	STREQ
	ICDRX write	Stop condition detected	—	A0	CREQ

Command	Trigger of Starting Operation	Operation Completion	Automatic Transition Conditions	Transition Destination Commands	Interrupt Flags
C8 to CB	ICCMD write	Start condition detected	—	—	MTREQ
	ICDRX write	Address transmission completed	$R/\overline{W} = 1$	—	MTREQ
	Simultaneous write to ICCMD and ICDRX	Start condition issued, address transmission completed	$R/\overline{W} = 1$	—	MTREQ
	ICDRX write	Address transmission completed	$R/\overline{W} = 0$	C4 to C7	MRREQ
	Simultaneous write to ICCMD and ICDRX	Start condition issued, address transmission completed	$R/\overline{W} = 0$	C4 to C7	MRREQ
	ICDRX write	Arbitration lost	—	A0	CREQ
C8	ICDRX write	Data transmission started	—	—	MTREQ
	—	—	NAK received*	—	CREQ
C9	ICDRX write	Transmission completed	—	—	MTREQ
CA, CB	ICDRX write	Transmission completed	ACK received	—	MTREQ
	ICDRX write	Transmission completed → stop condition issued	NAK received	A0	CREQ
C4	ICDRX read	(ACK transmitted), reception completed	—	—	MRREQ
C5	ICDRX read	(NAK transmitted), reception completed	—	—	MRREQ
C6	ICDRX read	Reception completed, ACK transmitted	—	—	MRREQ
C7	ICDRX read	Reception completed, NAK transmitted	—	—	MRREQ
CC	ICCMD write	Retransmission start condition issued	—	—	CREQ
CD	ICCMD write	ACK transmitted, stop condition issued	—	A0	CREQ
CE	ICCMD write	ACK transmitted, stop condition issued	—	A0	CREQ
CF	ICCMD write	Stop condition issued	—	A0	CREQ

Note: \* Data in the buffer is ignored.

## 17.5 Operation

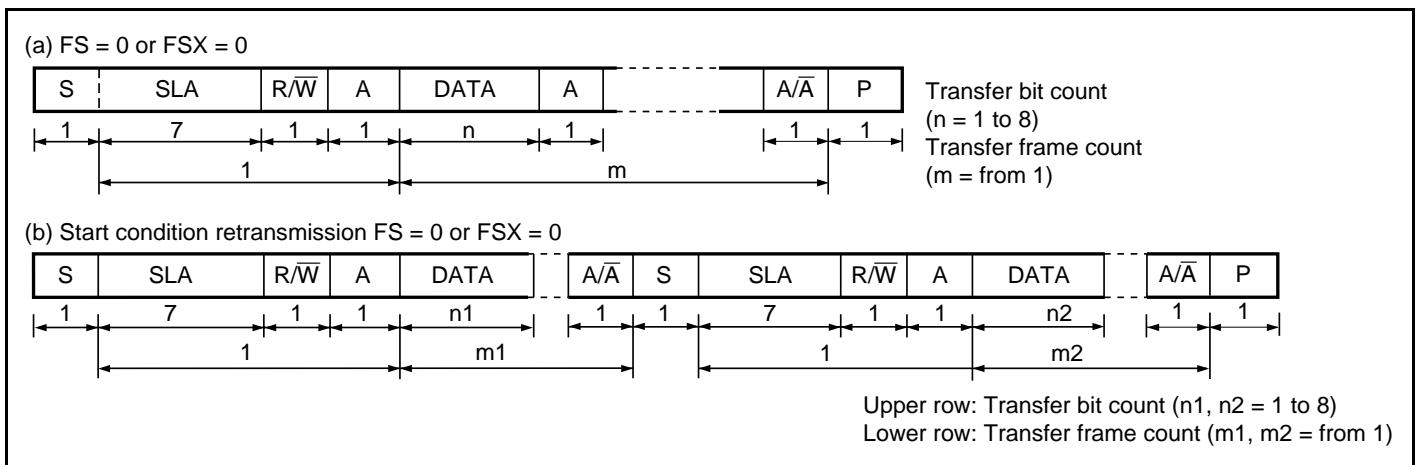
The I<sup>2</sup>C bus interface has an I<sup>2</sup>C bus format and a serial format.

### 17.5.1 I<sup>2</sup>C Bus Data Format

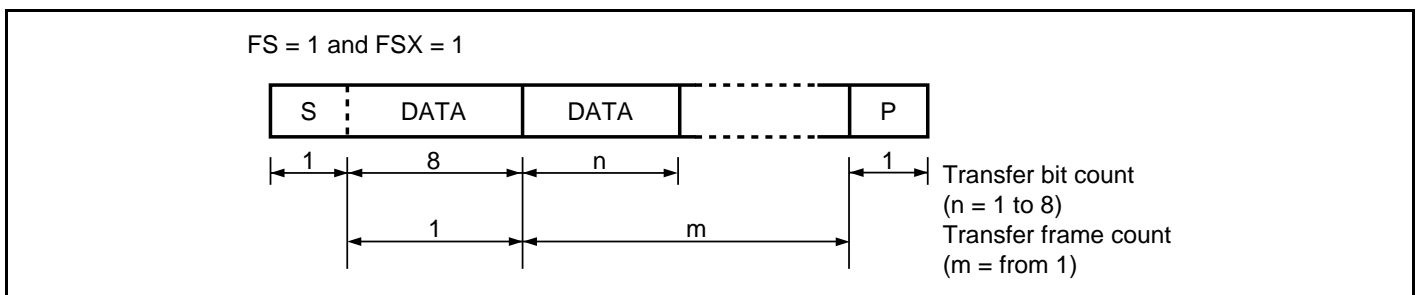
The I<sup>2</sup>C bus formats are addressing formats and an acknowledge bit is inserted. The first frame following a start condition always consists of 9 bits. The I<sup>2</sup>C bus format is shown in figure 17.4.

The serial formats are non-addressing formats with no acknowledge bit inserted. The serial format is shown in figure 17.5.

The I<sup>2</sup>C bus timing is shown in figure 17.6.

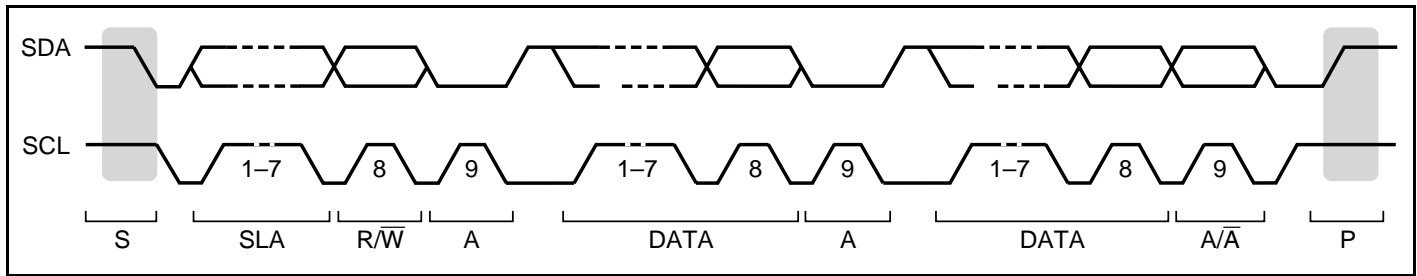


**Figure 17.4 I<sup>2</sup>C Bus Data Formats (I<sup>2</sup>C Bus Formats)**



**Figure 17.5 I<sup>2</sup>C Bus Formats (Serial Formats)**





**Figure 17.6 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/ $\bar{W}$ : Indicates the direction of data transfer: From the slave device to the master device when R/ $\bar{W}$  is 1, or from the master device to the slave device when R/ $\bar{W}$  is 0.

A: Acknowledge signal. The receiving device drives SDA to low.

DATA: Transmit/Receive data

P: Stop condition. The master device drives SDA from low to high while SCL is high.

### 17.5.2 Master Transmit Operation

When data is set to ICDR during the period between the execution of an instruction to issue a start condition and the creation of the start condition, the data may not be output normally, because there will be a conflict between generation of a start condition and output of data. Although data H'FF is to be sent to ICDR by a dummy write operation before an issue of a stop condition, the H'FF data may be output by the dummy write operation if the execution of the instruction to issue a stop condition is delayed. To prevent these problems, follow the flowchart shown below during the master transmit operation.

In I<sup>2</sup>C bus format master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The transmission procedure and operations for sequential data transmission in synchronization with the ICDR writing are described below.

1. Set the ICE bit in ICCR to 1. Set bits MLS, WAIT, and CKS2 to CKS0 in ICMR, and bits IICX1 and IICX0 in STCR, according to the operating mode.
2. Read the BBSY flag in ICCR to confirm that the bus is free.
3. Set bits MST and TRS to 1 in ICCR to select master transmit mode.
4. Write 1 to BBSY and 0 to SCP in ICCR. This changes SDA from high to low when SCL is high, and generates the start condition.

5. Then the IRIC and IRTR flags are set to 1. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU.

6. Write the data (slave address +  $R/\overline{W}$ ) to ICDR.

With the I<sup>2</sup>C bus format (when the FS bit in SAR or the FSX bit in SARX is 0), the first frame data following the start condition indicates the 7-bit slave address and transmit/receive direction.

To determine the end of the transfer, the IRIC flag is cleared to 0. After writing to ICDR, clear IRIC continuously so no other interrupt handling routine is executed. If the time for transmission of one frame of data has passed before the IRIC clearing, the end of transmission cannot be determined. The master device sequentially sends the transmit clock and the data written to ICDR using the timing shown in figure 17.7. The selected slave device (i.e. the slave device with the matching slave address) drives SDA low at the 9th transmit clock pulse and returns an acknowledge signal.

7. When one frame of data has been transmitted, the IRIC flag is set to 1 at the rise of the 9th transmit clock pulse. After one frame has been transmitted, SCL is automatically fixed low in synchronization with the internal clock until the next transmit data is written.

8. Read the ACKB bit in ICSR to confirm that ACKB is cleared to 0. When the slave device has not acknowledged (ACKB bit is 1), operate step [12] to end transmission, and retry the transmit operation.

9. Write the transmit data to ICDR.

As indicating the end of the transfer, the IRIC flag is cleared to 0. Perform the ICDR write and the IRIC flag clearing sequentially, just as in step [6]. Transmission of the next frame is performed in synchronization with the internal clock.

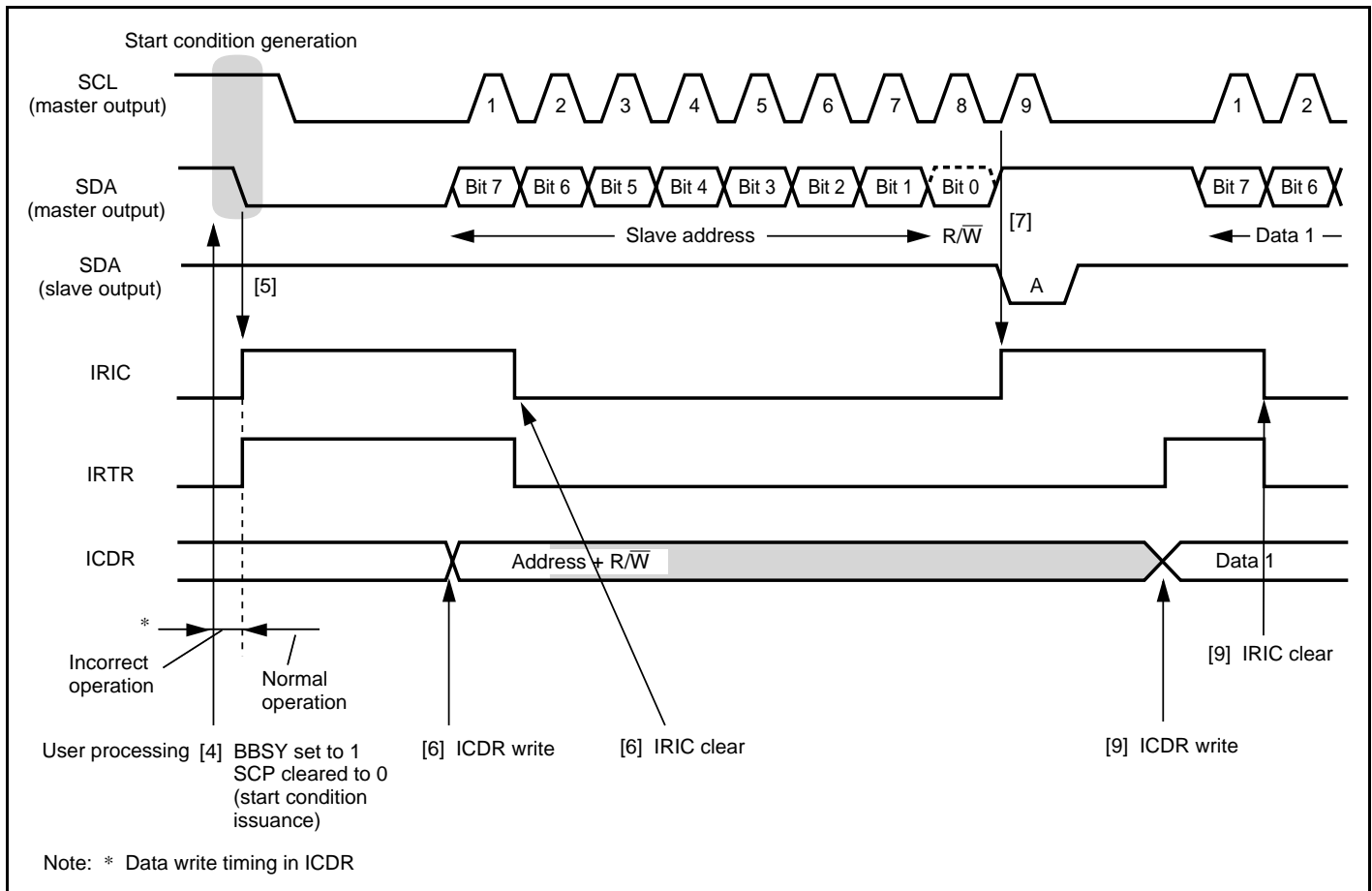
10. When one frame of data has been transmitted, the IRIC flag is set to 1 at the rise of the 9th transmit clock pulse. After one frame has been transmitted, SCL is automatically fixed low in synchronization with the internal clock until the next transmit data is written.

11. Read the ACKB bit in ICSR.

Confirm that the slave device has been acknowledged (ACKB bit is 0). When there is still data to be transmitted, go to step [9] to continue the next transmission operation. When the slave device has not acknowledged (ACKB bit is set to 1), operate step [12] to end transmission.

12. Clear the IRIC flag to 0.

Write 0 to BBSY and SCP in ICCR. This changes SDA from low to high when SCL is high, and generates the stop condition.



**Figure 17.7 Master Transmit Mode Operation Timing Example  
(MLS = WAIT = 0)**

### 17.5.3 Master Receive Operation

The data buffer of the IIC module can receive data consecutively since it consists of ICDRR and ICDRS. However, if the completion of receiving the last data is delayed, there will be a conflict between the instruction to issue a stop condition and the SCL clock output to receive the next data. This may generate unnecessary clocks or fix the output level of the SDA line as low.

The switch timing of the ACKB bit in ICSR should be controlled because the acknowledge bit does not return an acknowledge signal after receiving the last data in master mode.

These problems can be avoided by using the WAIT function. Follow the procedure shown below.

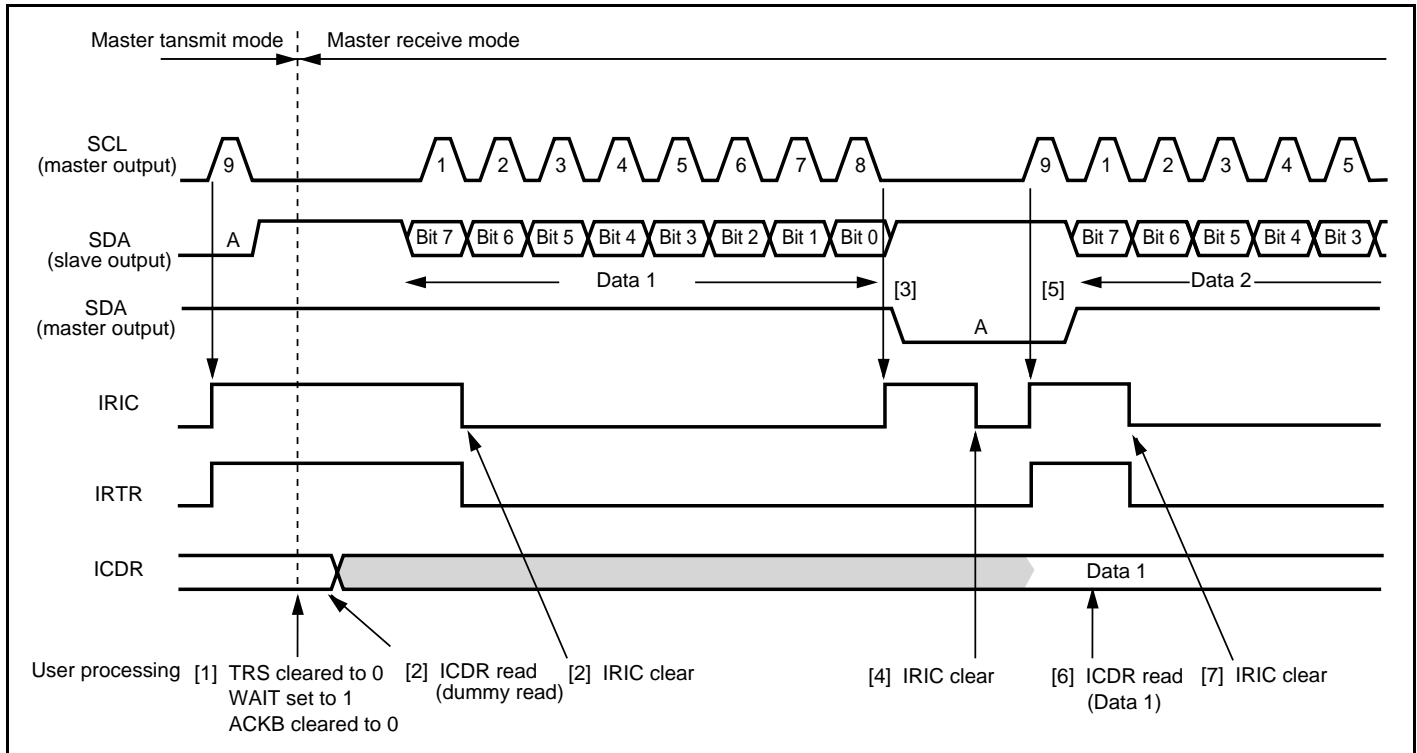
In I<sup>2</sup>C bus format master receive mode, the master device outputs the receive clock, receives data, and returns an acknowledge signal. The slave device transmits data. The reception procedure and operations for sequential data reception with the wait function in synchronization with the ICDR read operation are shown below.

1. Clear the TRS bit in ICCR to 0 to switch from transmit mode to receive mode.  
Set the WAIT bit in ICMR to 1.  
Clear the ACKB bit in ICSR to 0 (acknowledge data setting).
2. When ICDR is read (dummy data read), reception is started, and the receive clock is output, and data received, in synchronization with the internal clock.  
In order to detect wait operation, clear the IRIC flag in ICCR to 0. After reading ICDR, clear IRIC continuously so no other interrupt handling routine is executed. If the time for reception of one frame of data has passed before the IRIC clearing, the end of reception cannot be determined.
3. The IRIC flag is set to 1 at the fall of the 8th receive clock pulse. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU.  
SCL is automatically fixed low in synchronization with the internal clock until the IRIC flag clearing. If the first frame is the last receive data, execute step [10] to halt reception.
4. Clear the IRIC flag to clear the wait state.  
The master device outputs the 9th clock and drives SDA low at the 9th receive clock pulse to return an acknowledge signal.
5. When one frame of data has been received, the IRIC flag in ICCR and the IRTR flag in ICSR are set to 1 at the rise of the 9th receive clock pulse. The master device outputs the receive clock to receive the next data.
6. Read ICDR receive data.
7. Clear the IRIC flag to 0 to detect the next wait operation.  
Data reception process from steps [5] to [7] should be executed during one byte reception period after IRIC flag clearing in step [4] or [9] to release the wait state.
8. The IRIC flag is set to 1 at the fall of the 8th receive clock pulse.  
SCL is automatically fixed low in synchronization with the internal clock until the IRIC flag clearing. If this frame is the last receive data, execute step [10] to halt reception.
9. Clear the IRIC flag in ICCR to cancel wait state.  
The master device outputs the 9th clock and drives SDA low at the 9th receive clock pulse to return an acknowledge signal.  
Data can be received continuously by repeating steps [5] to [9].
10. Set the ACKB bit in ICSR to 1 so as to return the acknowledge data for the last reception.  
Set the TRS bit in ICCR to 1 to switch from receive mode to transmit mode.
11. Clear the IRIC flag to 0 to release the wait state.
12. When one frame of data has been received, the IRIC flag is set to 1 at the rise of the 9th receive clock pulse.

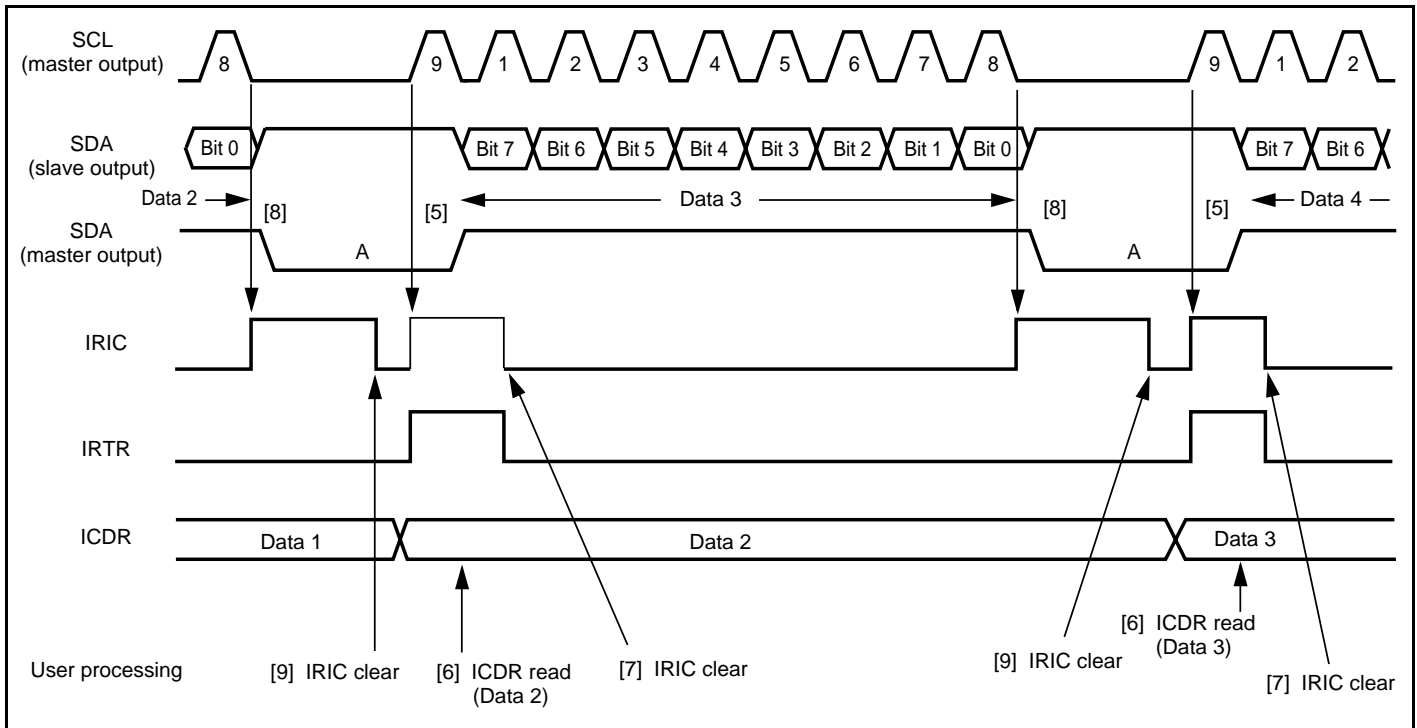
## 13. Clear the WAIT bit in ICMR to 0 to clear wait mode.

Read ICDR receive data and clear the IRIC flag to 0. Clearing of the IRIC flag should be done while WAIT = 0. (If the WAIT bit is cleared to 0 after clearing the IRIC flag and then an instruction to issue a stop condition is executed, the stop condition cannot be issued because the output level of SDA is fixed as low.)

## 14. Clear the BBSY bit and SCP bit to 0 in ICCR. This changes SDA from low to high when SCL is high, and generates the stop condition.



**Figure 17.8 Master Receive Mode Operation Timing Example (1)**  
(MLS = ACKB = 0, WAIT = 1)



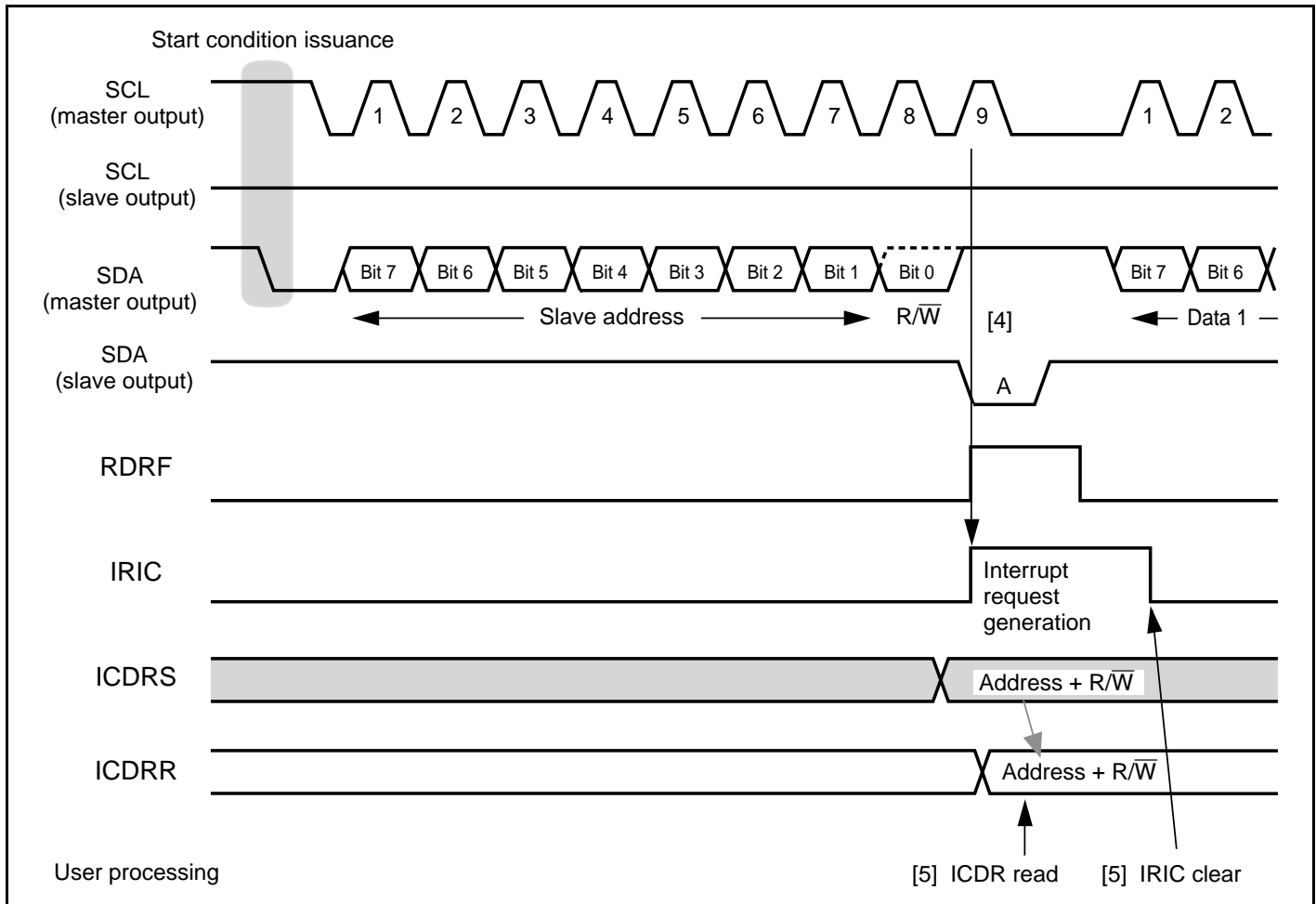
**Figure 17.9 Master Receive Mode Operation Timing Example (2)**  
(MLS = ACKB = 0, WAIT = 1)

### 17.5.4 Slave Receive Operation

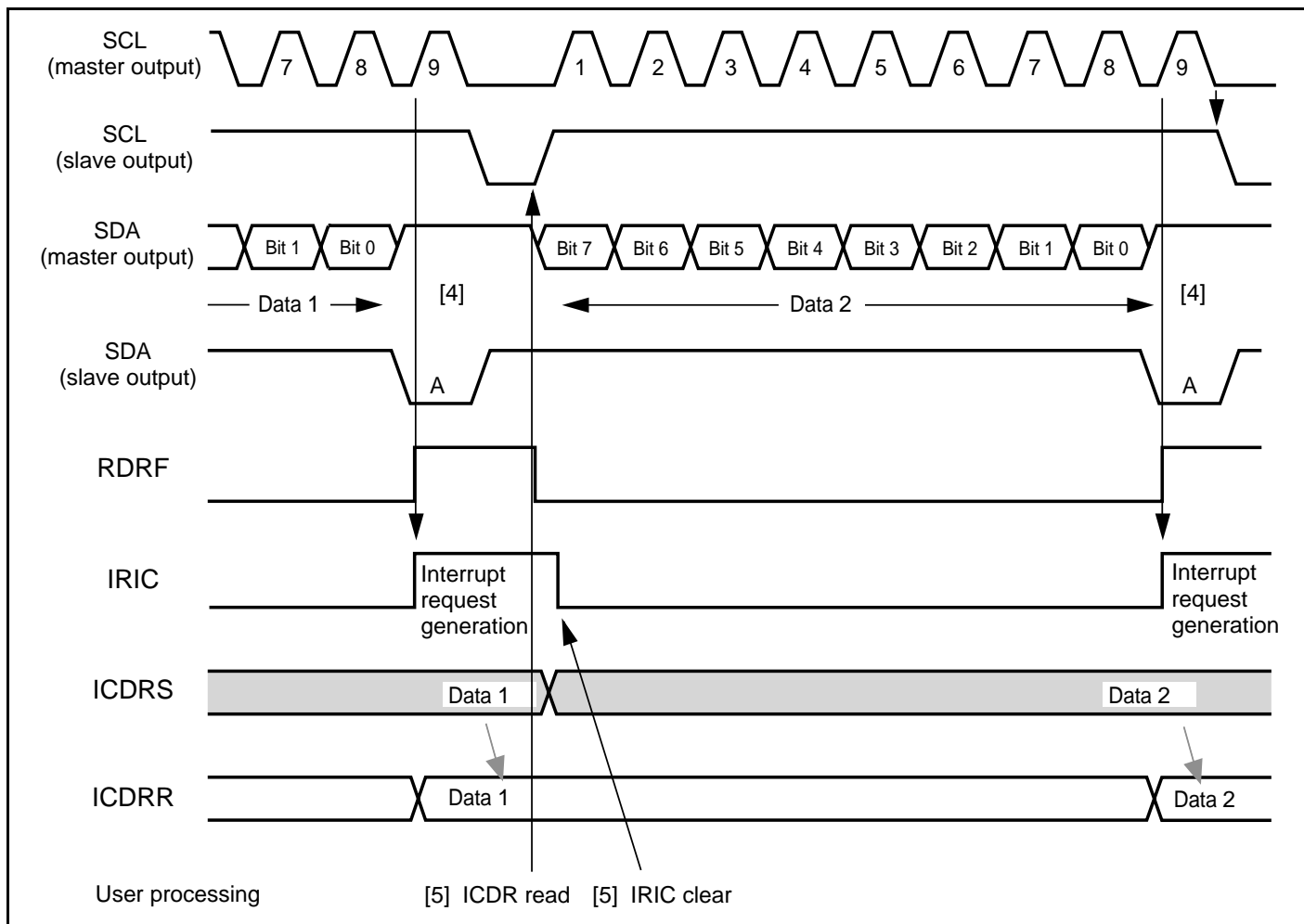
In slave receive mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The reception procedure and operations in slave receive mode are described below.

1. Set the ICE bit in ICCR to 1. Set the MLS bit in ICMR and the MST and TRS bits in ICCR according to the operating mode.
2. When the start condition output by the master device is detected, the BBSY flag in ICCR is set to 1.
3. When the slave address matches in the first frame following the start condition, the device operates as the slave device specified by the master device. If the 8th data bit (R/W) is 0, the TRS bit in ICCR remains cleared to 0, and slave receive operation is performed.
4. At the 9th clock pulse of the receive frame, the slave device drives SDA low and returns an acknowledge signal. At the same time, the IRIC flag in ICCR is set to 1. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU. If the RDRF internal flag has been cleared to 0, it is set to 1, and the receive operation continues. If the RDRF internal flag has been set to 1, the slave device drives SCL low from the fall of the receive clock until data is read into ICDR.
5. Read ICDR and clear the IRIC flag in ICCR to 0. The RDRF flag is cleared to 0.

Receive operations can be performed continuously by repeating steps [4] and [5]. When SDA is changed from low to high when SCL is high and the stop condition is detected, the BBSY flag in ICCR is cleared to 0.



**Figure 17.10 Slave Receive Mode Operation Timing Example (1)**  
(MLS = ACKB = 0)



**Figure 17.11 Slave Receive Mode Operation Timing Example (2)**  
**(MLS = ACKB = 0)**

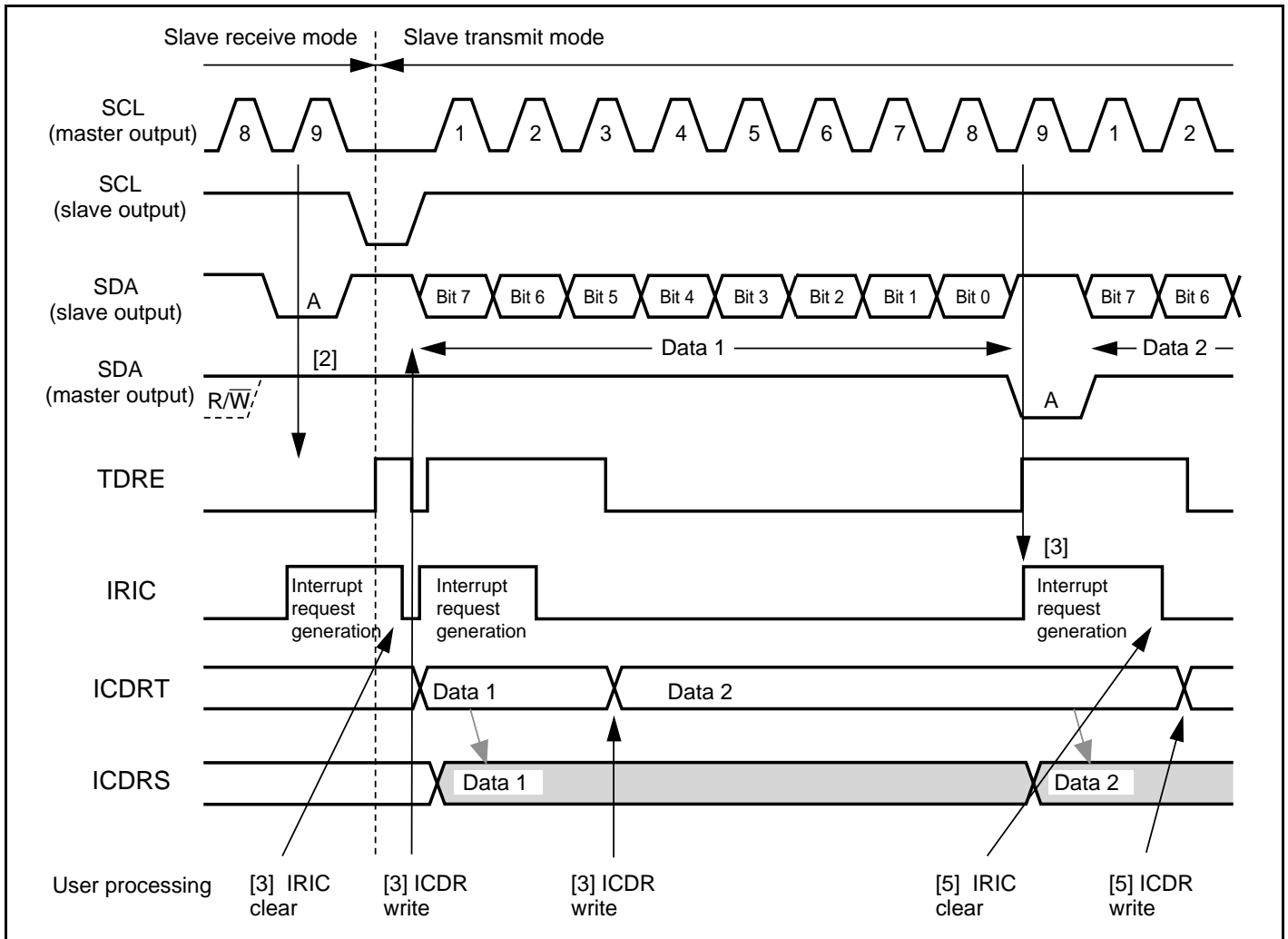


### 17.5.5 Slave Transmit Operation

In slave transmit mode, the slave device outputs the transmit data, while the master device outputs the receive clock and returns an acknowledge signal. The transmission procedure and operations in slave transmit mode are described below.

1. Set the ICE bit in ICCR to 1. Set the MLS bit in ICMR and the MST and TRS bits in ICCR according to the operating mode.
2. When the slave address matches in the first frame following detection of the start condition, the slave device drives SDA low at the 9th clock pulse and returns an acknowledge signal. At the same time, the IRIC flag in ICCR is set to 1. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU. If the 8th data bit ( $R/\overline{W}$ ) is 1, the TRS bit in ICCR is set to 1, and the mode changes to slave transmit mode automatically. The TDRE internal flag is set to 1. The slave device drives SCL low from the fall of the transmit clock until ICDR data is written.
3. After clearing the IRIC flag to 0, write data to ICDR. The TDRE internal flag is cleared to 0. The written data is transferred to ICDRS, and the TDRE internal flag and IRIC and IRTR flags are set to 1 again. After clearing the IRIC flag to 0, write the next data to ICDR. The slave device sequentially sends the data written into ICDR in accordance with the clock output by the master device at the timing shown in figure 17.12.
4. When one frame of data has been transmitted, the IRIC flag in ICCR is set to 1 at the rise of the 9th transmit clock pulse. If the TDRE internal flag has been set to 1, this slave device drives SCL low from the fall of the transmit clock until data is written to ICDR. The master device drives SDA low at the 9th clock pulse, and returns an acknowledge signal. As this acknowledge signal is stored in the ACKB bit in ICSR, this bit can be used to determine whether the transfer operation was performed normally. When the TDRE internal flag is 0, the data written into ICDR is transferred to ICDRS and the TDRE internal flag and IRIC and IRTR flags are set to 1 again.
5. To continue transmission, clear the IRIC flag to 0, then write the next data to be transmitted into ICDR. The TDRE internal flag is cleared to 0.

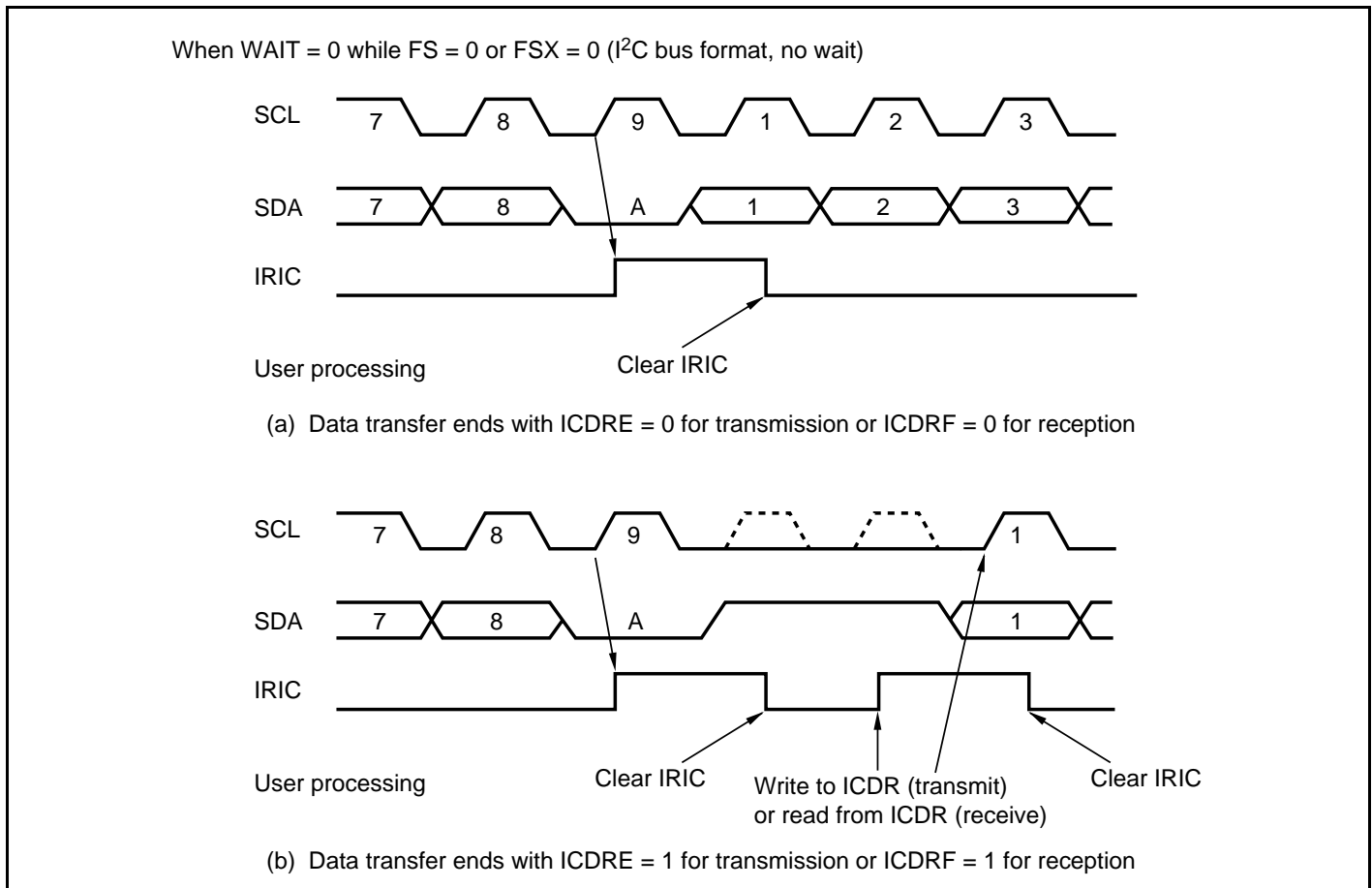
Transmit operations can be performed continuously by repeating steps [4] and [5]. To end transmission, write H'FF to ICDR. When SDA is changed from low to high when SCL is high and the stop condition is detected, the BBSY flag in ICCR is cleared to 0.



**Figure 17.12 Slave Transmit Mode Operation Timing Example (MLS = 0)**

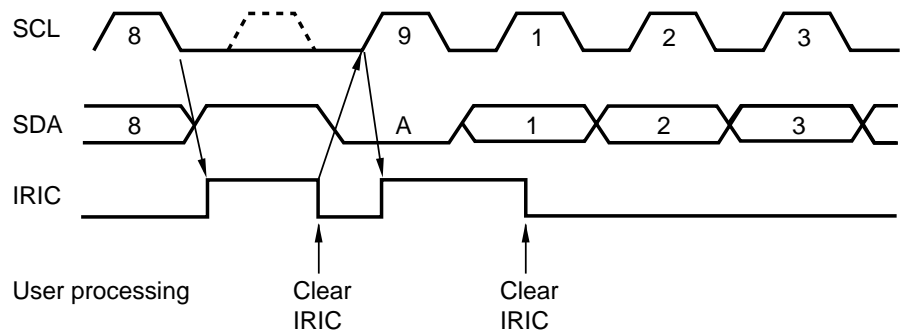
### 17.5.6 IRIC Setting Timing and SCL Control

The interrupt request flag (IRIC) is set at different times depending on the WAIT bit in ICMR, the FS bit in SAR, and the FSX bit in SARX. If the ICDRE or ICDRF internal flag is set to 1, SCL is automatically held low after one frame has been transferred; this timing is synchronized with the internal clock. Figures 17.13 to 17.15 show the IRIC flag timings and SCL control, and figure 17.16 shows an example of the interrupt flag timing of the operation reservation adapter.

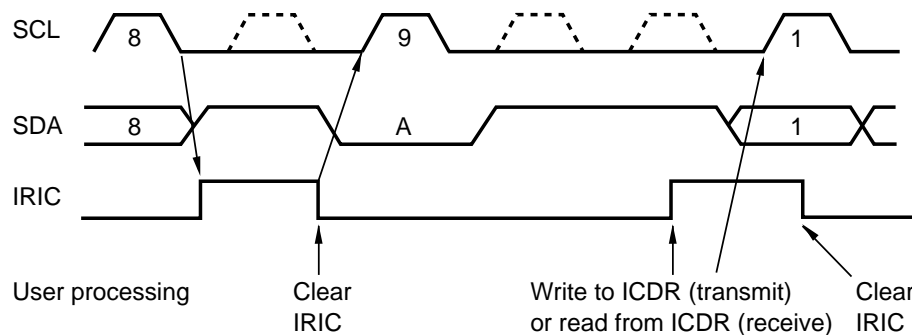


**Figure 17.13 IRIC Flag Timing and SCL Control (1)**

When WAIT = 1 while FS = 0 or FSX = 0 (I<sup>2</sup>C bus format, wait inserted)



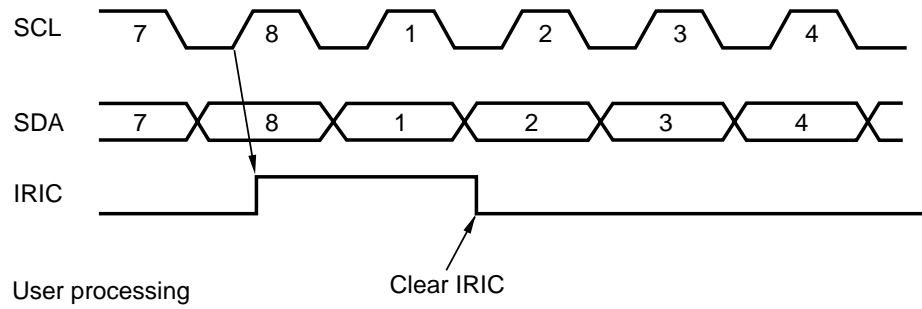
(a) Data transfer ends with ICDRE = 0 for transmission or ICDRF = 0 for reception



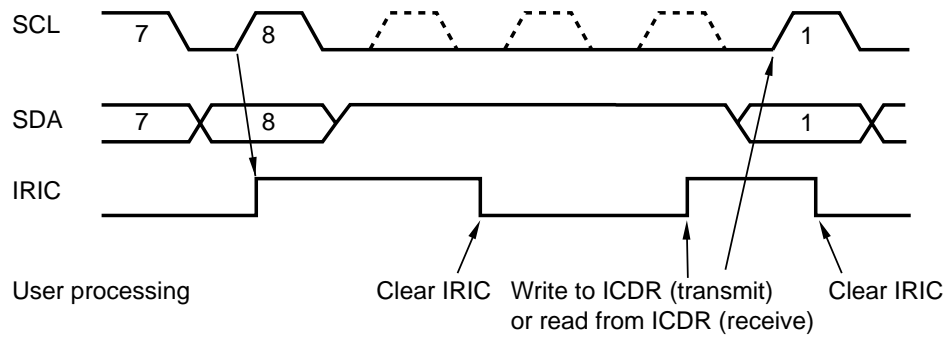
(b) Data transfer ends with ICDRE = 1 for transmission or ICDRF = 1 for reception

**Figure 17.14 IRIC Flag Timing and SCL Control (2)**

When FS = 1 while FSX = 1 (synchronous serial format)

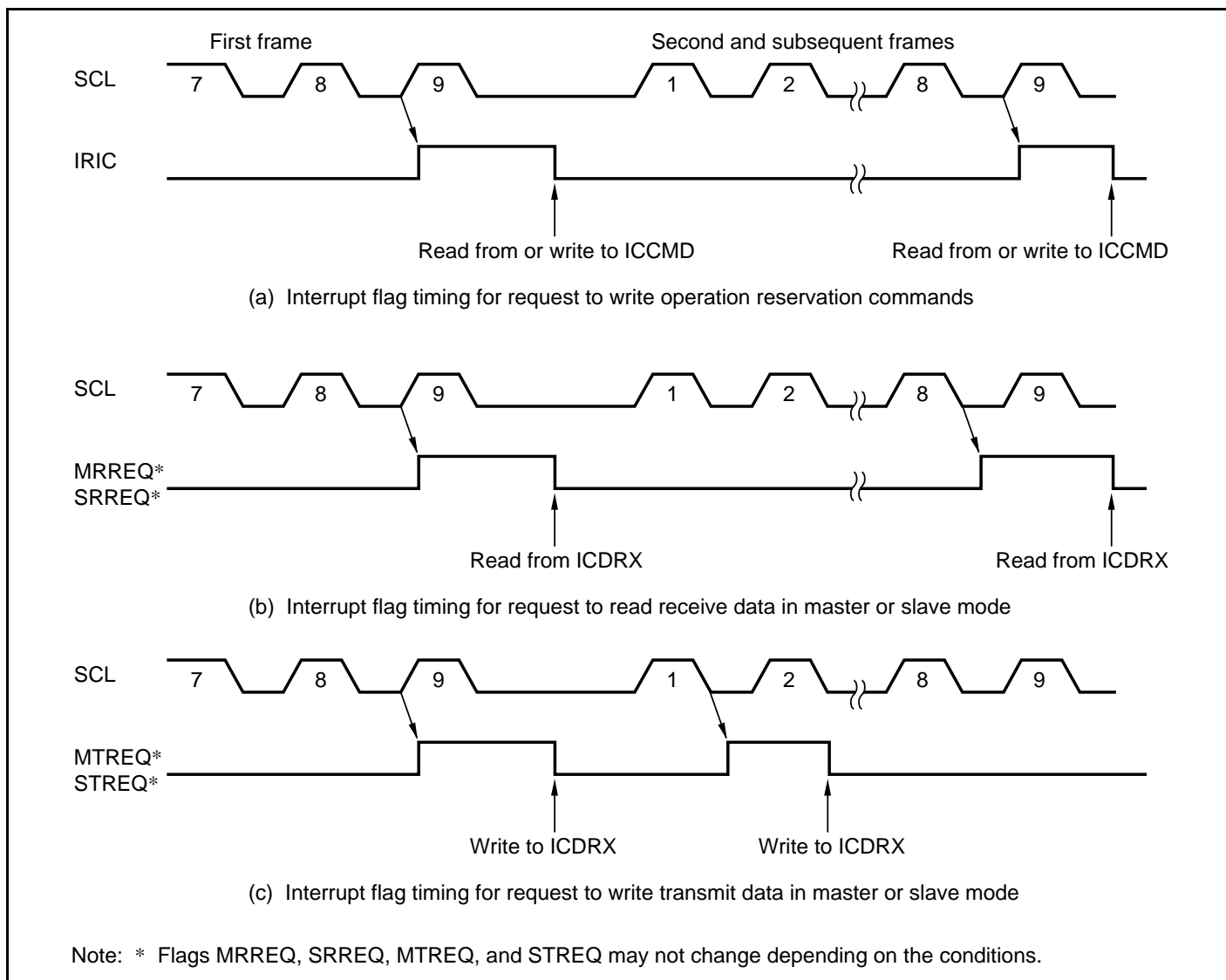


(a) Data transfer ends with ICDRE = 0 for transmission or ICDRF = 0 for reception



(b) Data transfer ends with ICDRE = 1 for transmission or ICDRF = 1 for reception

**Figure 17.15 IRIC Flag Timing and SCL Control (3)**



**Figure 17.16 Example of Interrupt Flag Timing of Operation Reservation Adapter**

### 17.5.7 Operation Using DTC

This LSI provides the DTC to allow continuous data transfer. The DTC is activated when the IRTR flag is set to 1, which is one of the two interrupt flags (IRTR and IRIC). When the ACKE bit is 0, the ICDRE, IRIC, and IRTR flags are set at the end of data transmission regardless of the acknowledge bit value. When the ACKE bit is 1, the ICDRE, IRIC, and IRTR flags are set if data transmission is completed with the acknowledge bit value of 0, and when the ACKE bit is 1, only the IRIC flag is set if data transmission is completed with the acknowledge bit value of 1.

When initiated, the DTC transfers specified number of bytes, and then clears the ICDRE, IRIC, and IRTR flags to 0. Therefore, no interrupt is generated during continuous data transfer; however, if data transmission is completed with the acknowledge bit value of 1 when the ACKE bit is 1, the DTC is not activated, thus allowing an interrupt to be generated if enabled.

The acknowledge bit may indicate specific events such as completion of receive data processing for some receiving devices, and for other receiving devices, the acknowledge bit may be held to 1, indicating no specific events.

The I<sup>2</sup>C bus format provides selection of the slave device and transfer direction by means of the slave address and the R/ $\overline{W}$  bit, and confirmation of reception and display of the last frame with the acknowledge bit. Therefore, continuous data transfer using the DTC must be carried out in conjunction with CPU processing by means of interrupts.

For transfer operations by the operation reservation adapter, a stop condition is automatically issued when transfer of the number of transfer data bytes set by the DTC ends in master mode.

Table 17.8 shows some examples of processing using the DTC. Table 17.9 shows some examples of operation reservation adapter processing using the DTC. These examples assume that the number of transfer data bytes is known in slave mode.

**Table 17.8 Examples of Operation Using DTC**

<b>Item</b>	<b>Master Transmit Mode</b>	<b>Master Receive Mode</b>	<b>Slave Transmit Mode</b>	<b>Slave Receive Mode</b>
Slave address + R/W bit transmission/reception	Transmission by DTC (ICDR write)	Transmission by CPU (ICDR write)	Reception by CPU (ICDR read)	Reception by CPU (ICDR read)
Dummy data read	—	Processing by CPU (ICDR read)	—	—
Actual data transmission/reception	Transmission by DTC (ICDR write)	Reception by DTC (ICDR read)	Transmission by DTC (ICDR write)	Reception by DTC (ICDR read)
Dummy data (H'FF) write	—	—	Processing by DTC (ICDR write)	—
Last frame processing	Not necessary	Reception by CPU (ICDR read)	Not necessary	Reception by CPU (ICDR read)
Transfer request processing after last frame processing	1st time: Clearing by CPU 2nd time: End condition issuance by CPU	Not necessary	Automatic clearing on detection of end condition during transmission of dummy data (H'FF)	Not necessary
Setting of number of DTC transfer data frames	Transmission: Actual data count + 1 (+1 equivalent to slave address + R/W bits)	Reception: Actual data count	Transmission: Actual data count + 1 (+1 equivalent to dummy data (H'FF))	Reception: Actual data count



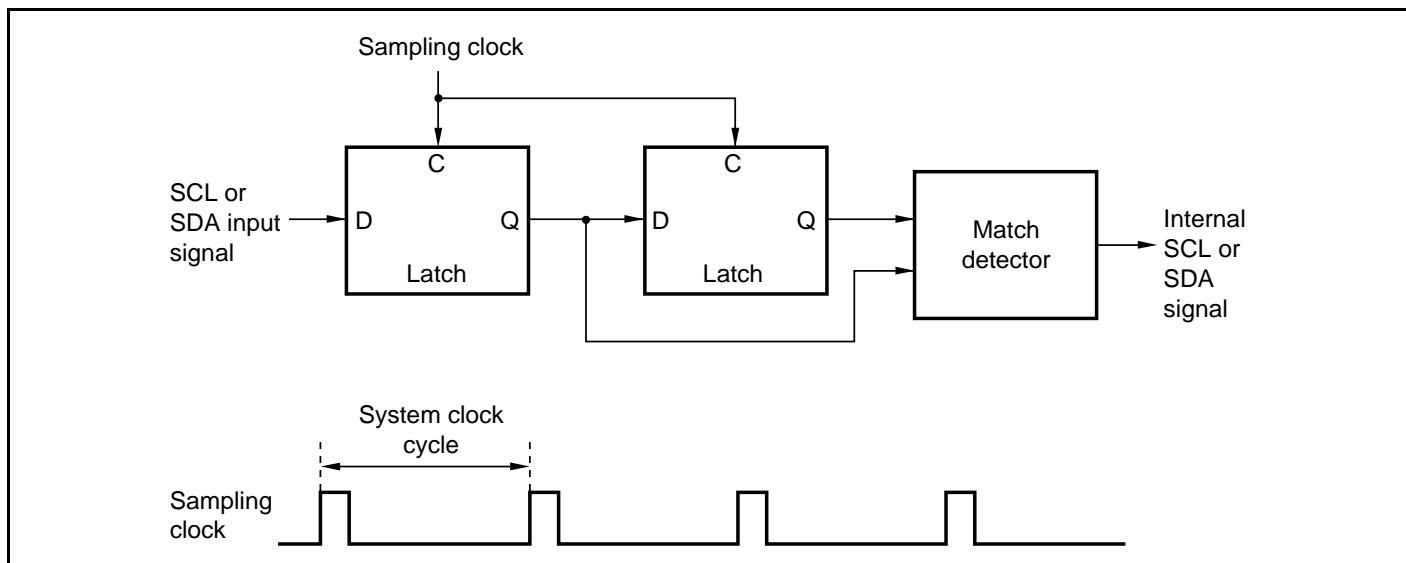
**Table 17.9 Examples of Operation Reservation Adapter Operation Using DTC**

<b>Item</b>	<b>Master Transmit Mode</b>	<b>Master Receive Mode</b>	<b>Slave Transmit Mode</b>	<b>Slave Receive Mode</b>
Slave address + R/W bit transmission/reception	Transmission by CPU (ICDRX + ICCMD write)	Transmission by CPU (ICDRX + ICCMD write)	Not necessary	Not necessary
Dummy data read	—	Processing by CPU (ICDRX read)	—	—
Actual data transmission/reception	Transmission by DTC (ICDRX write)	Reception by DTC (ICDRX read)	Transmission by DTC (ICDRX write)	Reception by DTC (ICDRX read)
Dummy data (H'FF) write	—	—	Not necessary	—
Last frame processing	Not necessary	Not necessary	Not necessary	Not necessary
Transfer request processing after last frame processing	MTREQ: Clearing by CPU CREQ: Clearing by CPU	MRREQ: Clearing by CPU CREQ: Clearing by CPU	STREQ: Clearing by CPU CREQ: Clearing by CPU	SRREQ: Clearing by CPU CREQ: Clearing by CPU
Setting of number of DTC transfer data frames	Transmission: Actual data count	Reception: Actual data count	Transmission: Actual data	Reception: Actual data count

### 17.5.8 Noise Canceler

The logic levels at the SCL and SDA pins are routed through noise cancelers before being latched internally. Figure 17.17 shows a block diagram of the noise canceler.

The noise canceler consists of two cascaded latches and a match detector. The SCL (or SDA) pin input signal is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.



**Figure 17.17 Block Diagram of Noise Canceler**

### 17.5.9 Initialization of Internal State

The IIC has a function for forcible initialization of its internal state if a deadlock occurs during communication.

Initialization is executed in accordance with the setting of bits CLR3 to CLR0 in ICCRX or clearing the ICE bit. For details on the setting of bits CLR3 to CLR0, see section 17.3.7, IIC Operation Reservation Adapter Control Register (ICCRX).

**Scope of Initialization:** The initialization executed by this function covers the following items:

- ICDRE and ICDRF internal flags
- Transmit/receive sequencer and internal operating clock counter
- Internal latches for retaining the output state of the SCL and SDA pins (wait, clock, data output, etc.)

The following items are not initialized:

- Actual register values (ICDR, SAR, SARX, ICMR, ICCR, ICSR, ICXR (other than ICDRE and ICDRF flags), ICCRX, ICSRA, ICSRB, ICSRC, ICRRX, ICDRS, ICCNT, ICCMD)
- Internal latches used to retain register read information for setting/clearing flags in ICMR, ICCR, ICSR, ICSRB, and ICSRC
- The value of the ICMR register bit counter (BC2 to BC0)
- Generated interrupt sources (interrupt sources transferred to the interrupt controller)

**Notes on Initialization:**

- Interrupt flags and interrupt sources are not cleared, and so flag clearing measures must be taken as necessary.
- Basically, other register flags are not cleared either, and so flag clearing measures must be taken as necessary.
- When initialization is executed by ICCRX, the write data for bits CLR3 to CLR0 is not retained. To perform IIC clearance, bits CLR3 to CLR0 must be written to simultaneously using an MOV instruction. Do not use a bit manipulation instruction such as BCLR.
- Similarly, when clearing is required again, all the bits must be written to simultaneously in accordance with the setting.
- If a flag clearing setting is made during transmission/reception, the IIC module will stop transmitting/receiving at that point and the SCL and SDA pins will be released. When transmission/reception is started again, register initialization, etc., must be carried out as necessary to enable correct communication as a system.

The value of the BBSY bit cannot be modified directly by this module clear function, but since the stop condition pin waveform is generated according to the state and release timing of the SCL and SDA pins, the BBSY bit may be cleared as a result. Similarly, state switching of other bits and flags may also have an effect.

To prevent problems caused by these factors, the following procedure should be used when initializing the IIC state.

1. Execute initialization of the internal state according to the setting of bits CLR3 to CLR0 or ICE bit clearing.
2. Execute a stop condition issuance instruction (write 0 to BBSY and SCP) to clear the BBSY bit to 0, and wait for two transfer rate clock cycles.
3. Re-execute initialization of the internal state according to the setting of bits CLR3 to CLR0 or ICE bit clearing.
4. Initialize (re-set) the IIC registers.

**17.5.10 Sample Flowcharts**

Figures 17.18 to 17.21 show sample flowcharts for using the I<sup>2</sup>C bus interface in each mode.

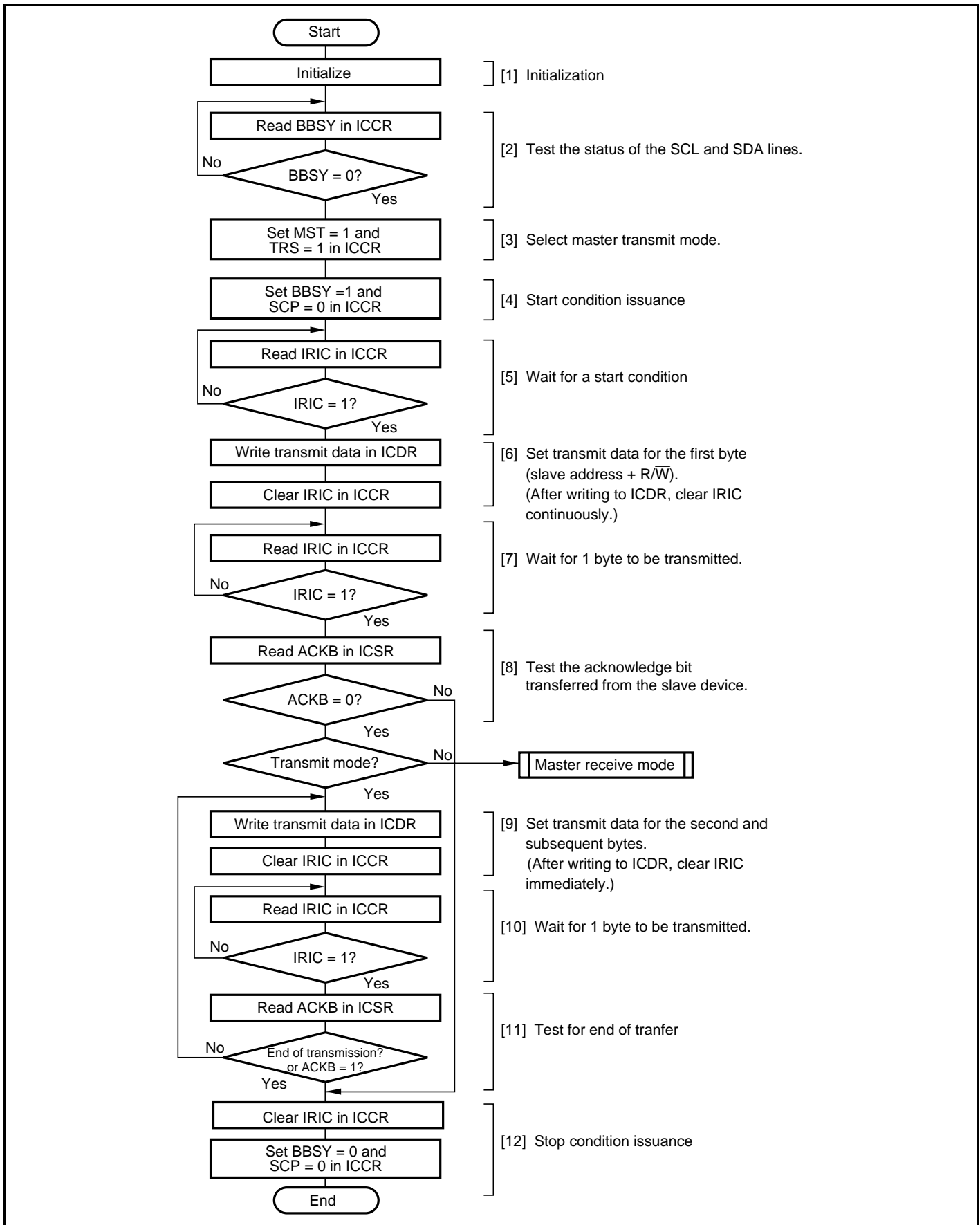


Figure 17.18 Sample Flowchart for Master Transmit Mode

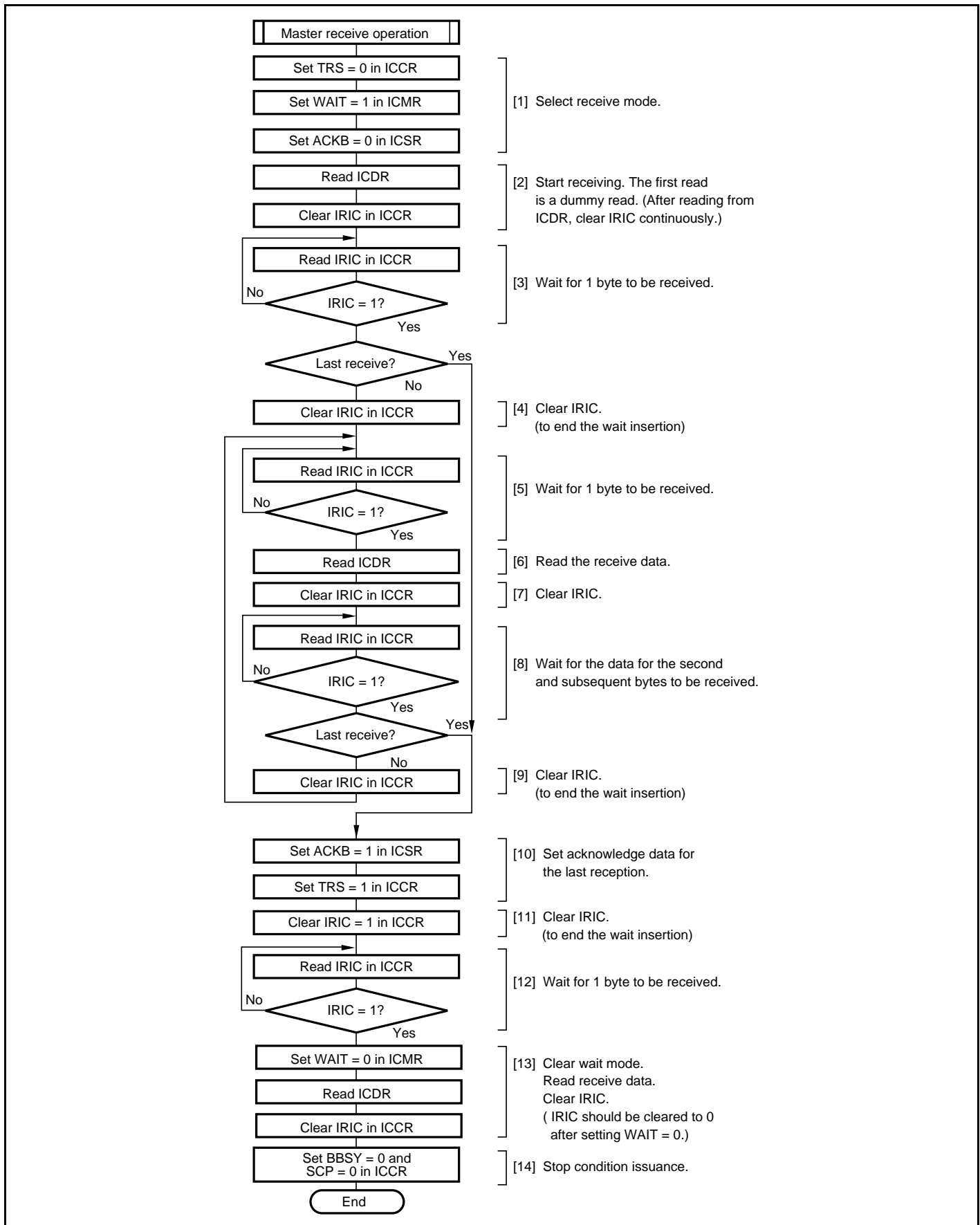


Figure 17.19 Sample Flowchart for Master Receive Mode

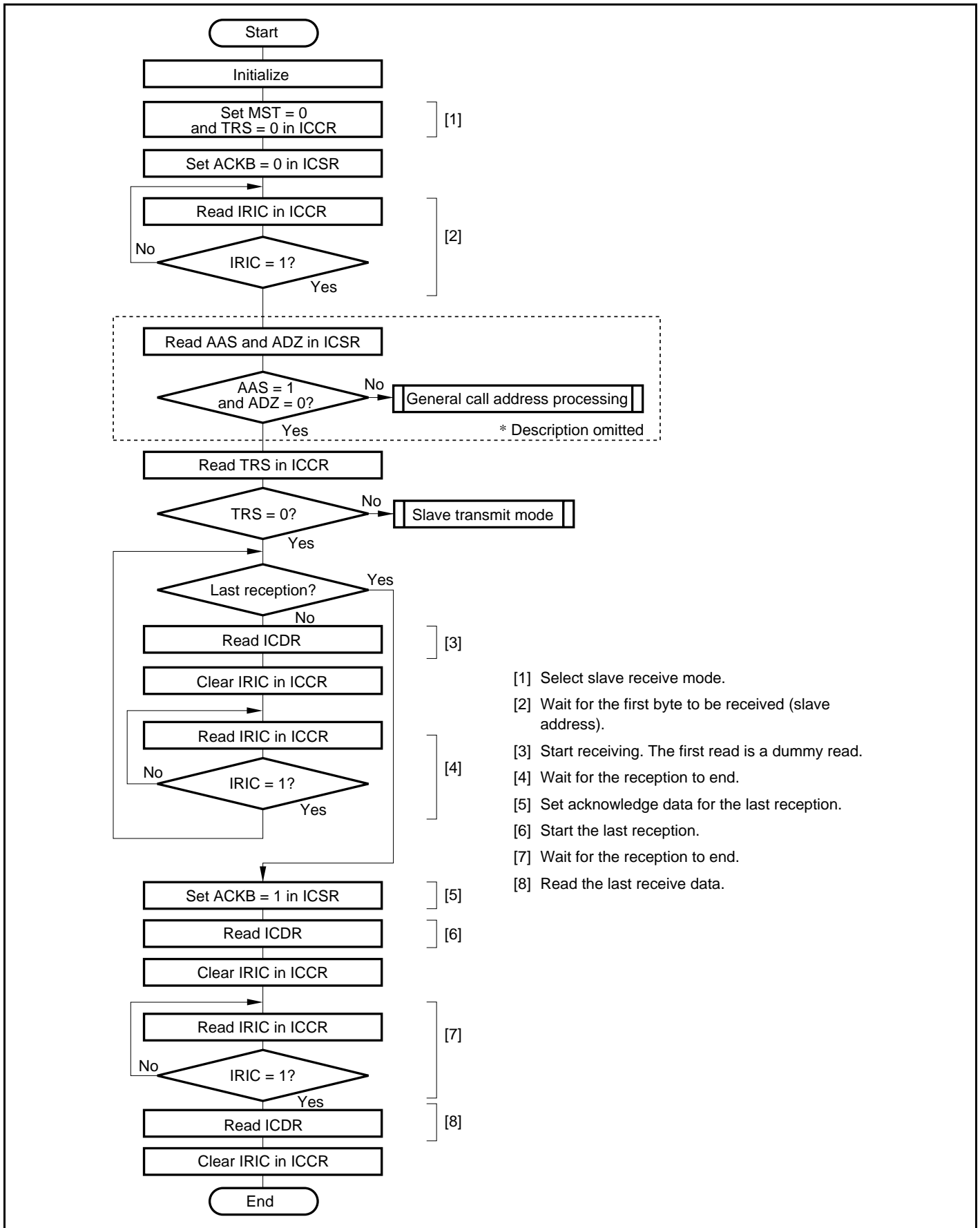
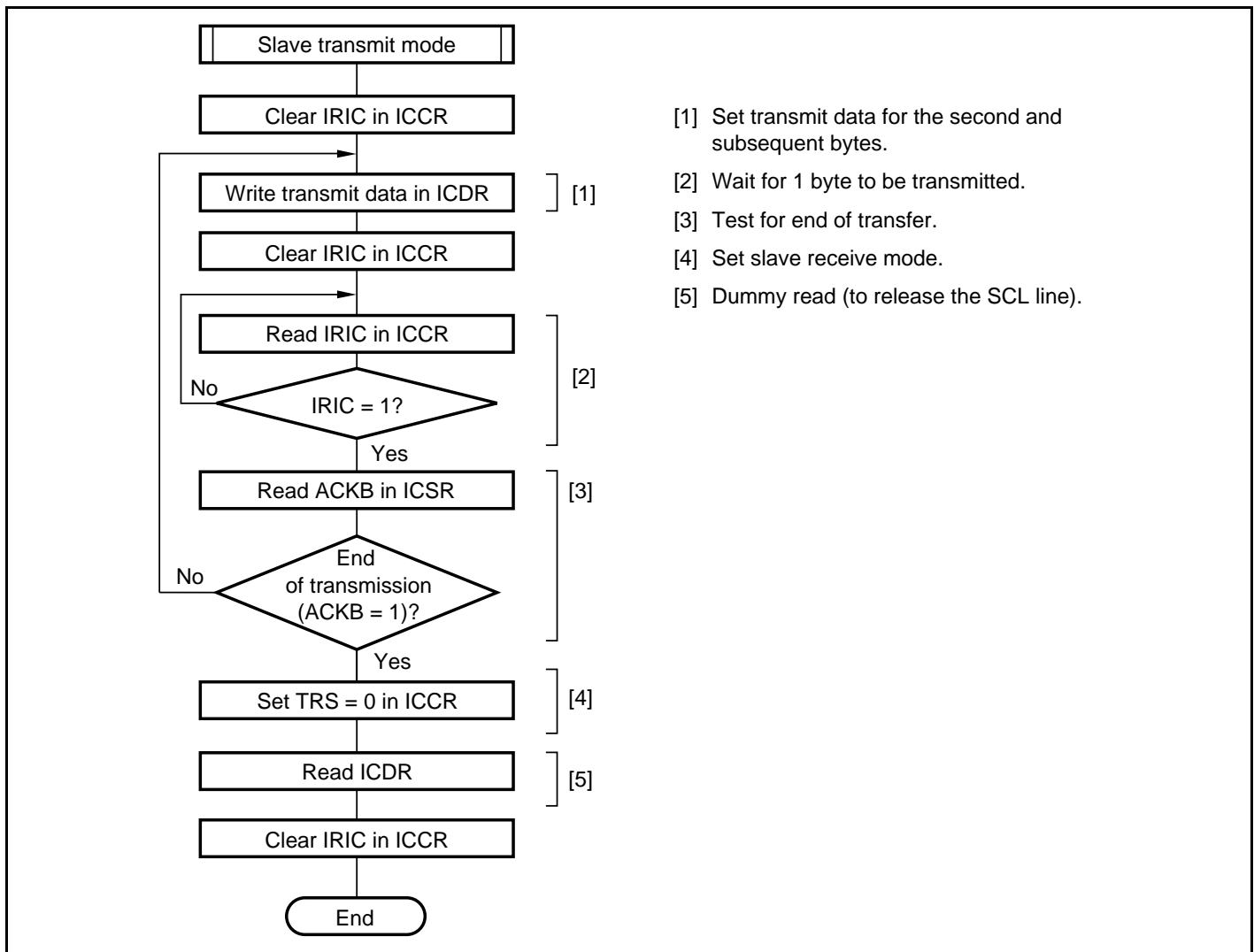


Figure 17.20 Sample Flowchart for Slave Receive Mode



**Figure 17.21 Sample Flowchart for Slave Transmit Mode**

## 17.6 Interrupt Sources

The I<sup>2</sup>C bus interface has four interrupt sources: IICC, IICM, IICR, and IICT. Table 17.10 shows the interrupt sources and priorities. Each interrupt source can be enabled or disabled independently by interrupt enable bits in ICCR, ICCRX, or ICCNT. Independent signals are sent to the interrupt controller for each interrupt.

The IICM, IICR, and IICT interrupts can be used as DTC activation interrupt sources.





## 17.7 Usage Notes

1. In master mode, if an instruction to generate a start condition is immediately followed by an instruction to generate a stop condition, neither condition will be output correctly. To output consecutive start and stop conditions, after issuing the instruction that generates the start condition, read the relevant ports, check that SCL and SDA are both low, then issue the instruction that generates the stop condition. Note that SCL may not yet have gone low when BBSY is cleared to 0.
2. Either of the following two conditions will start the next transfer. Pay attention to these conditions when accessing to ICDR.
  - Write to ICDR when ICE = 1 and TRS = 1 (including automatic transfer from transmit buffer to shift register)
  - Read from ICDR when ICE = 1 and TRS = 0 (including automatic transfer from shift register to receive buffer)
3. Table 17.11 shows the timing of SCL and SDA outputs in synchronization with the internal clock. Timings on the bus are determined by the rise and fall times of signals affected by the bus load capacitance, series resistance, and parallel resistance.

**Table 17.11 I<sup>2</sup>C Bus Timing (SCL and SDA Outputs)**

Item	Symbol	Output Timing	Unit	Notes
SCL output cycle time	t <sub>SCLO</sub>	28 t <sub>cyc</sub> to 256 t <sub>cyc</sub>	ns	
SCL output high pulse width	t <sub>SCLHO</sub>	0.5 t <sub>SCLO</sub>	ns	
SCL output low pulse width	t <sub>SCLLO</sub>	0.5 t <sub>SCLO</sub>	ns	
SDA output bus free time	t <sub>BUFO</sub>	0.5 t <sub>SCLO</sub> – 1 t <sub>cyc</sub>	ns	
Start condition output hold time	t <sub>STAHO</sub>	0.5 t <sub>SCLO</sub> – 1 t <sub>cyc</sub>	ns	
Retransmission start condition output setup time	t <sub>STASO</sub>	1 t <sub>SCLO</sub>	ns	
Stop condition output setup time	t <sub>STOSO</sub>	0.5 t <sub>SCLO</sub> + 2 t <sub>cyc</sub>	ns	
Data output setup time (master)	t <sub>SDASO</sub>	1 t <sub>SCLLO</sub> – 3 t <sub>cyc</sub>	ns	
Data output setup time (slave)		1 t <sub>SCLL</sub> – (6 t <sub>cyc</sub> or 12 t <sub>cyc</sub> *)		
Data output hold time	t <sub>SDAHO</sub>	3 t <sub>cyc</sub>	ns	

Note: \* 6 t<sub>cyc</sub> when IICX is 0, 12 t<sub>cyc</sub> when 1.

4. SCL and SDA inputs are sampled in synchronization with the internal clock. The AC timing therefore depends on the system clock cycle t<sub>cyc</sub>, as shown in table 29.10. Note that the I<sup>2</sup>C bus interface AC timing specifications will not be met with a system clock frequency of less than 5 MHz.

5. The I<sup>2</sup>C bus interface specification for the SCL rise time  $t_{sr}$  is 1000 ns or less (300 ns for high-speed mode). In master mode, the I<sup>2</sup>C bus interface monitors the SCL line and synchronizes one bit at a time during communication. If  $t_{sr}$  (the time for SCL to go from low to  $V_{IH}$ ) exceeds the time determined by the input clock of the I<sup>2</sup>C bus interface, the high period of SCL is extended. The SCL rise time is determined by the pull-up resistance and load capacitance of the SCL line. To insure proper operation at the set transfer rate, adjust the pull-up resistance and load capacitance so that the SCL rise time does not exceed the values given in table 17.12.

**Table 17.12 Permissible SCL Rise Time ( $t_{sr}$ ) Values**

IICX1, $t_{cyc}$	IICX0 Indication	I <sup>2</sup> C Bus Specification (Max.)	Time Indication[ns]						
			$\phi =$ 5 MHz	$\phi =$ 8 MHz	$\phi =$ 10 MHz	$\phi =$ 16 MHz	$\phi =$ 20 MHz	$\phi =$ 25 MHz	
0	7.5 $t_{cyc}$	Standard mode	1000	1000	937	750	468	375	300
		High-speed mode	300	300	300	300	300	300	300
1	17.5 $t_{cyc}$	Standard mode	1000	1000	1000	1000	1000	875	700
		High-speed mode	300	300	300	300	300	300	300

6. The I<sup>2</sup>C bus interface specifications for the SCL and SDA rise and fall times are under 1000 ns and 300 ns. The I<sup>2</sup>C bus interface SCL and SDA output timing is prescribed by  $t_{cyc}$ , as shown in table 17.11. However, because of the rise and fall times, the I<sup>2</sup>C bus interface specifications may not be satisfied at the maximum transfer rate. Table 17.13 shows output timing calculations for different operating frequencies, including the worst-case influence of rise and fall times. The values in the above table will vary depending on the settings of the IICX1, IICX0, and CKS2 to CKS0 bits. Depending on the frequency it may not be possible to achieve the maximum transfer rate; therefore, whether or not the I<sup>2</sup>C bus interface specifications are met must be determined in accordance with the actual setting conditions.  $t_{BUFO}$  fails to meet the I<sup>2</sup>C bus interface specifications at any frequency. The following solutions should be investigated.

- Provide coding to secure the necessary interval (approximately 1  $\mu$ s) between issuance of a stop condition and issuance of a start condition.
- Select devices whose input timing permits this output timing for use as slave devices connected to the I<sup>2</sup>C bus.

$t_{SCLLO}$  in high-speed mode and  $t_{STASO}$  in standard mode fail to satisfy the I<sup>2</sup>C bus interface specifications for worst-case calculations of  $t_{sr}/t_{sf}$ . The following solutions should be investigated.

- Adjusting the rise and fall times by means of a pull-up resistor and capacitive load.
- Reducing the transfer rate to meet the specifications.

- Selecting devices whose input timing permits this output timing for use as slave devices connected to the I<sup>2</sup>C bus.

**Table 17.13 I<sup>2</sup>C Bus Timing (with Maximum Influence of  $t_{Sr}/t_{Sf}$ )**

Item	$t_{cyc}$ Indication		Time Indication (at Maximum Transfer Rate) [ns]							
			$t_{Sr}/t_{Sf}$ Influence (Max.)	I <sup>2</sup> C Bus Specifi- (Min.)	$\phi =$	$\phi =$	$\phi =$	$\phi =$	$\phi =$	$\phi =$
					5 MHz	8 MHz	10 MHz	16 MHz	20 MHz	25 MHz
$t_{SCLHO}$	$0.5 t_{SCLO} (-t_{Sr})$	Standard mode	-1000	4000	4000	4000	4000	4000	4000	4000
		High-speed mode	-300	600	950	950	950	950	950	950
$t_{SCLLO}$	$0.5 t_{SCLO} (-t_{Sr})$	Standard mode	-250	4700	4750	4750	4750	4750	4750	4750
		High-speed mode	-250	1300	1000 <sup>*1</sup>	1000 <sup>*1</sup>	1000 <sup>*1</sup>	1000 <sup>*1</sup>	1000 <sup>*1</sup>	1000 <sup>*1</sup>
$t_{BUFO}$	$0.5 t_{SCLO} - 1 t_{cyc} (-t_{Sr})$	Standard mode	-1000	4700	3800 <sup>*1</sup>	3875 <sup>*1</sup>	3900 <sup>*1</sup>	3938 <sup>*1</sup>	3950 <sup>*1</sup>	3960 <sup>*1</sup>
		High-speed mode	-300	1300	750 <sup>*1</sup>	825 <sup>*1</sup>	850 <sup>*1</sup>	888 <sup>*1</sup>	900 <sup>*1</sup>	910 <sup>*1</sup>
$t_{STAHO}$	$0.5 t_{SCLO} - 1 t_{cyc} (-t_{Sr})$	Standard mode	-250	4000	4550	4625	4650	4688	4700	4710
		High-speed mode	-250	600	800	875	900	938	950	960
$t_{STASO}$	$1 t_{SCLO} (-t_{Sr})$	Standard mode	-1000	4700	9000	9000	9000	9000	9000	9000
		High-speed mode	-300	600	2200	2200	2200	2200	2200	2200
$t_{STOSO}$	$0.5 t_{SCLO} + 2 t_{cyc} (-t_{Sr})$	Standard mode	-1000	4000	4400	4250	4200	4125	4100	4080
		High-speed mode	-300	600	1350	1200	1150	1075	1050	1030
$t_{SDASO}$ (master)	$1 t_{SCLLO}^{*3} - 3 t_{cyc} (-t_{Sr})$	Standard mode	-1000	250	3100	3325	3400	3513	3550	3580
		High-speed mode	-300	100	400	625	700	813	850	880
$t_{SDASO}$ (slave)	$1 t_{SCLL}^{*3} - 3 t_{cyc}^{*2} (-t_{Sr})$	Standard mode	-1000	250	1300	2200	2500	2950	3100	3220
		High-speed mode	-300	100	-1400 <sup>*1</sup>	-500 <sup>*1</sup>	-200 <sup>*1</sup>	250	400	520
$t_{SDAHO}$	$3 t_{cyc}$	Standard mode	0	0	600	375	300	188	150	120
		High-speed mode	0	0	600	375	300	188	150	120

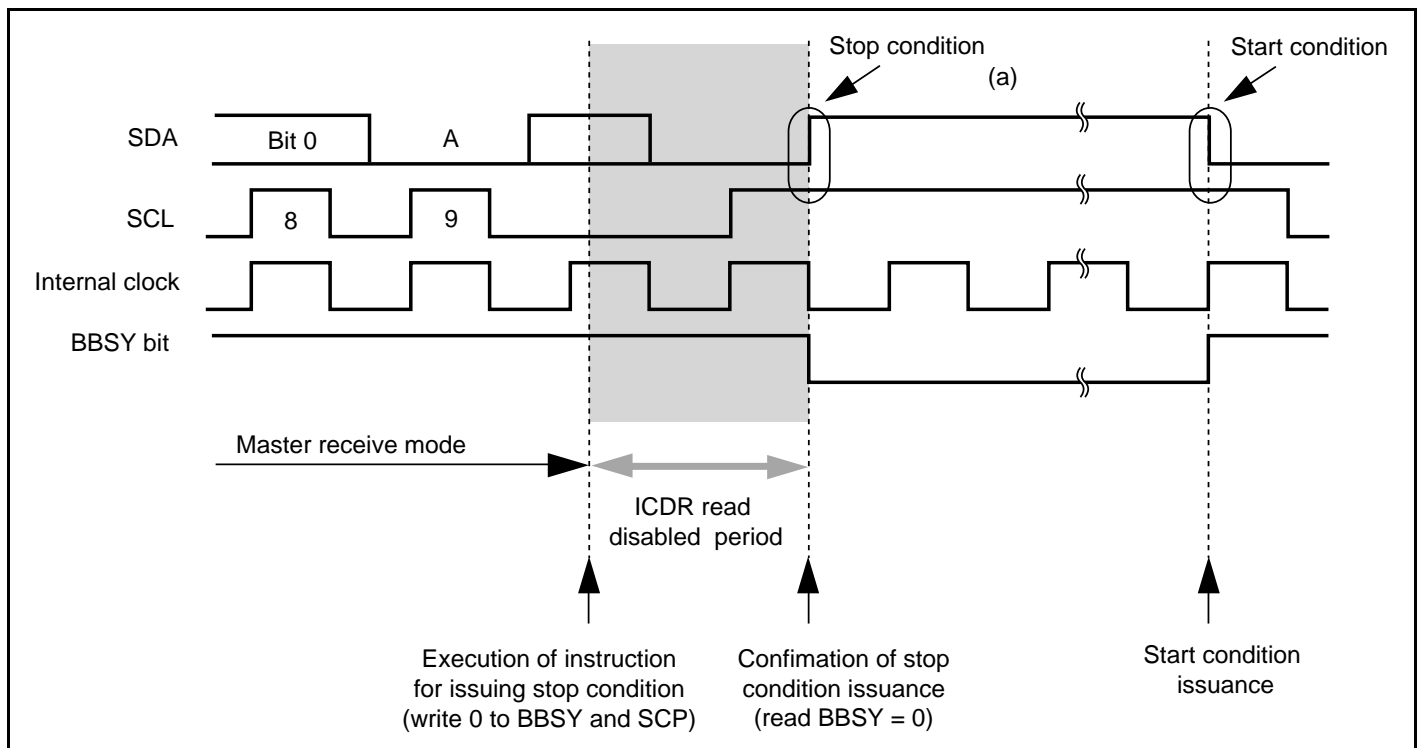
Notes: 1. Does not meet the I<sup>2</sup>C bus interface specifications. Remedial action such as the following is necessary: (a) secure a start/stop condition issuance interval; (b) adjust the rise and fall times by means of a pull-up resistor and capacitive load; (c) reduce the transfer rate; (d) select slave devices whose input timing permits this output timing.

The values in the above table will vary depending on the settings of the IICX bit and bits CKS2 to CKS0. Depending on the frequency it may not be possible to achieve the maximum transfer rate; therefore, whether or not the I<sup>2</sup>C bus interface specifications are met must be determined in accordance with the actual setting conditions.

- Value when the IICX bit is set to 1. When the IICX bit is cleared to 0, the value is ( $t_{SCLL} - 6 t_{cyc}$ ).
- Calculated using the I<sup>2</sup>C bus specification values (standard mode: 4700 ns min.; high-speed mode: 1300 ns min.).

## 7. Notes on ICDR read at end of master reception

To halt reception after completion of a receive operation in master receive mode, set the TRS bit to 1 and write 0 to BBSY and SCP in ICCR. This changes the SDA pin from low to high when the SCL pin is high, and generates the stop condition. After this, receive data can be read by means of an ICDR read, but if data remains in the buffer, the ICDRS receive data will not be transferred to ICDR, and so it will not be possible to read the second byte of data. If it is necessary to read the second byte of data, issue the stop condition in master receive mode (i.e. with the TRS bit cleared to 0). When reading the receive data, first confirm that the BBSY bit in ICCR is cleared to 0, the stop condition has been generated, and the bus has been released, then read ICDR with TRS cleared to 0. Note that if the receive data (ICDR data) is read in the interval between execution of the instruction for issuance of the stop condition (writing 0 to the BBSY bit in ICCR) and the actual generation of the stop condition, the clock may not be output correctly in subsequent master transmission. Rewriting of IIC control bits for changing the operating mode and settings for transmission/reception, such as clearing the MST bit after master transmit/receive operation has ended, must be done during interval (a) in figure 17.22.

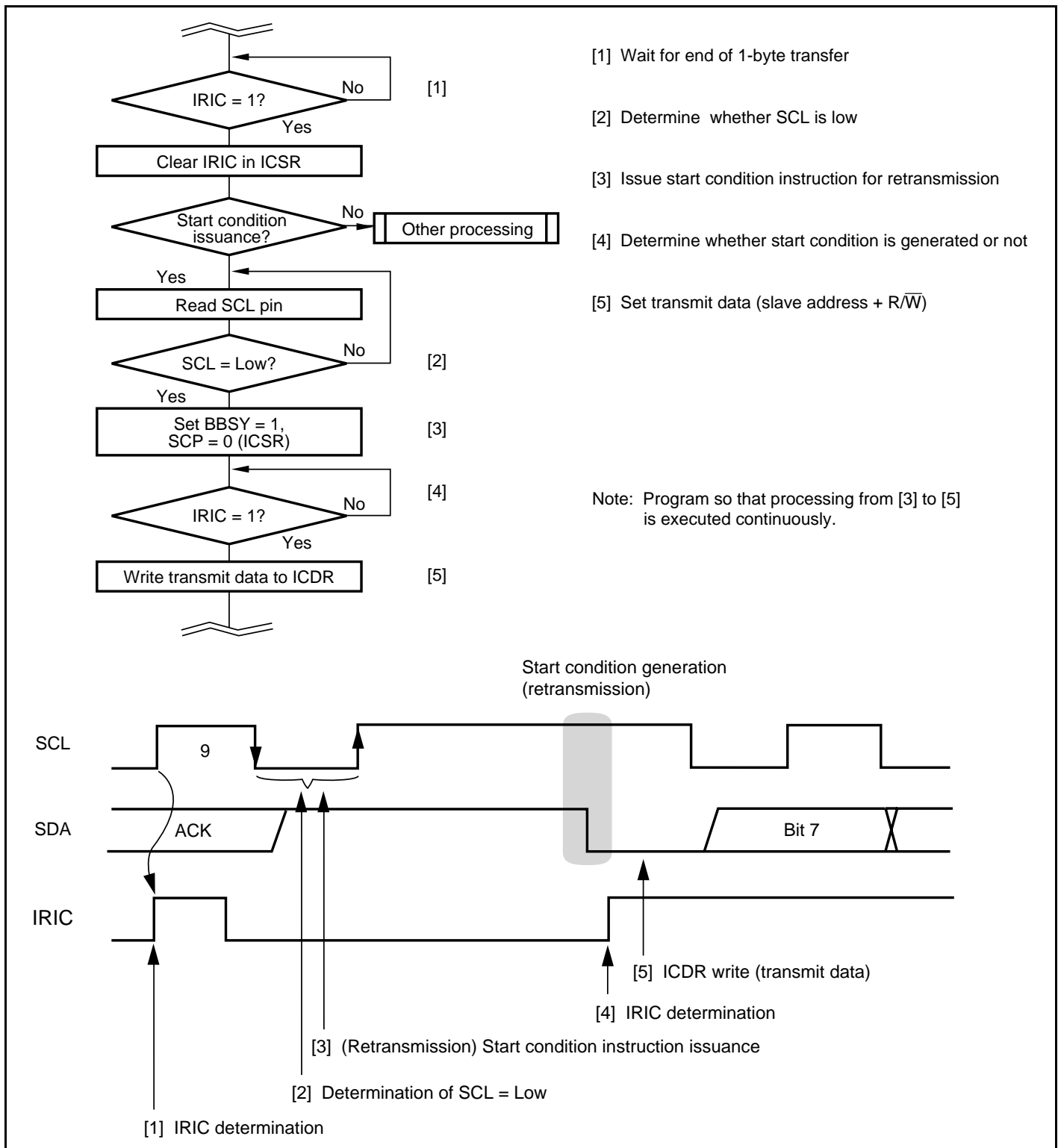


**Figure 17.22 Notes on Reading Master Receive Data**

## 8. Notes on start condition issuance for retransmission

Depending on the timing combination with the start condition issuance and the subsequently writing data to ICDR, it may not be possible to issue the retransmission condition and the data transmission after retransmission condition issuance.

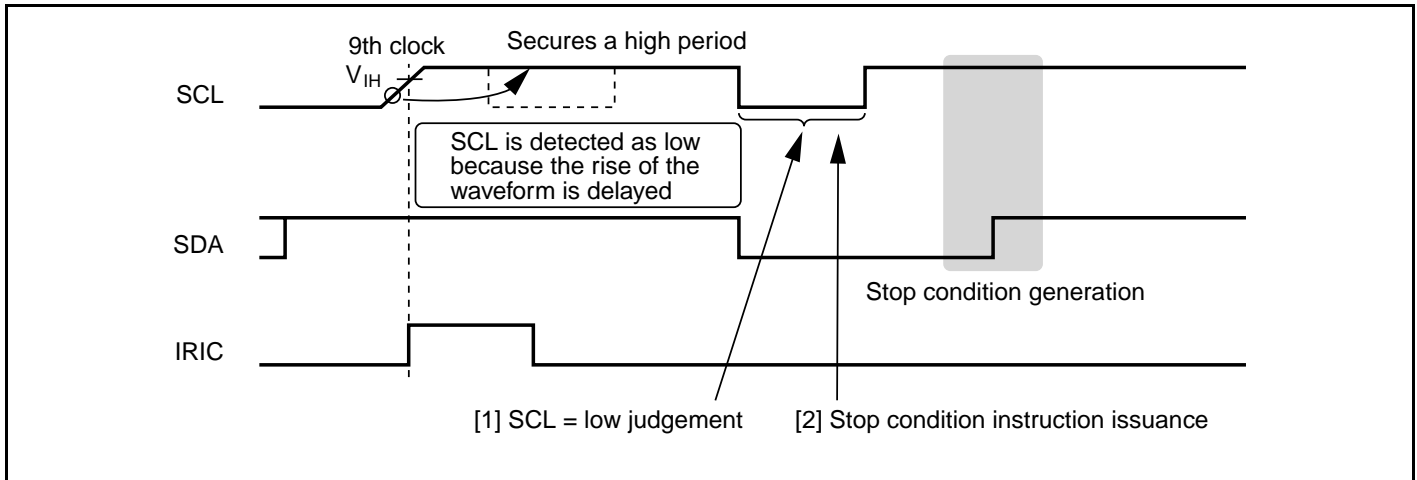
Therefore, after start condition issuance is done and the start condition is generated, write the transmit data to ICDR. Figure 17.23 shows the timing of start condition issuance for retransmission, and the timing for subsequently writing data to ICDR, together with the corresponding flowchart.



**Figure 17.23 Flowchart and Timing of Start Condition Issuance for Retransmission**

9. Note on when I<sup>2</sup>C bus interface stop condition instruction is issued

In a situation where the rise time of the 9th clock of SCL exceeds the stipulated value because of a large bus load capacity or where a slave device in which a wait can be inserted by driving the SCL pin low is used, the stop condition instruction should be issued after reading SCL after the rise of the 9th clock pulse and determining that it is low, as shown in figure 17.24.

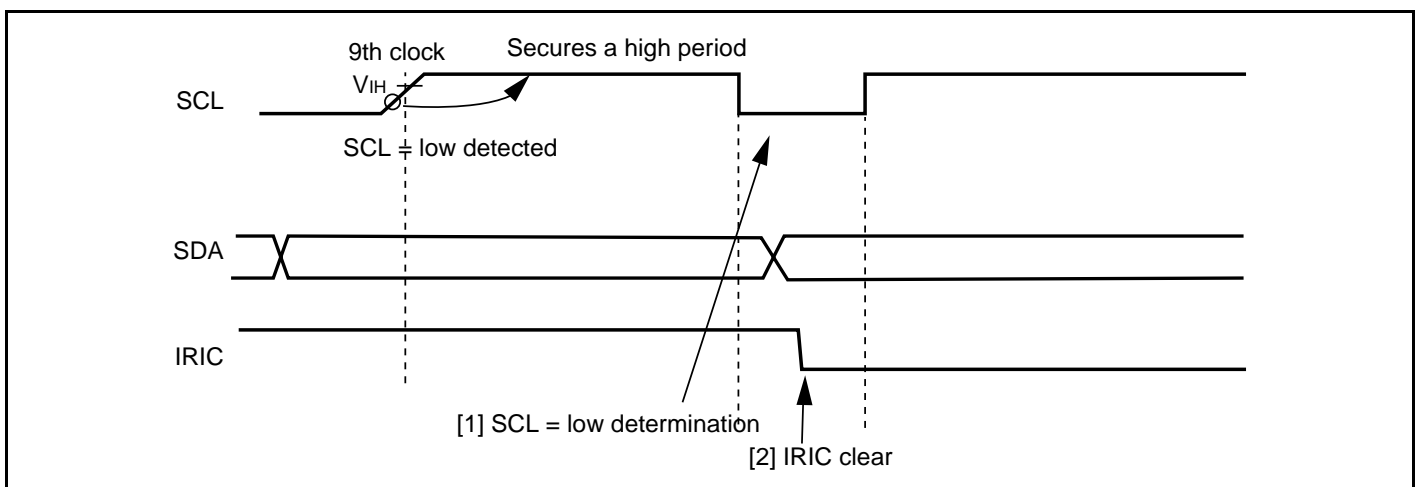


**Figure 17.24 Stop Condition Issuance Timing**

## 10. Note on IRIC flag clearing when wait function is used

When the wait function is used in I<sup>2</sup>C bus interface master mode and in a situation where the rise time of SCL exceeds the stipulated value or where a slave device in which a wait can be inserted by driving the SCL pin low is used, the IRIC flag should be cleared after determining that the SCL pin is low.

If the IRIC flag is cleared to 0 when WAIT = 1 while the SCL is extending the high level time, the SDA level may change before the SCL goes low, which may generate a start or stop condition erroneously.



**Figure 17.25 IRIC Flag Clearing Timing When WAIT = 1**

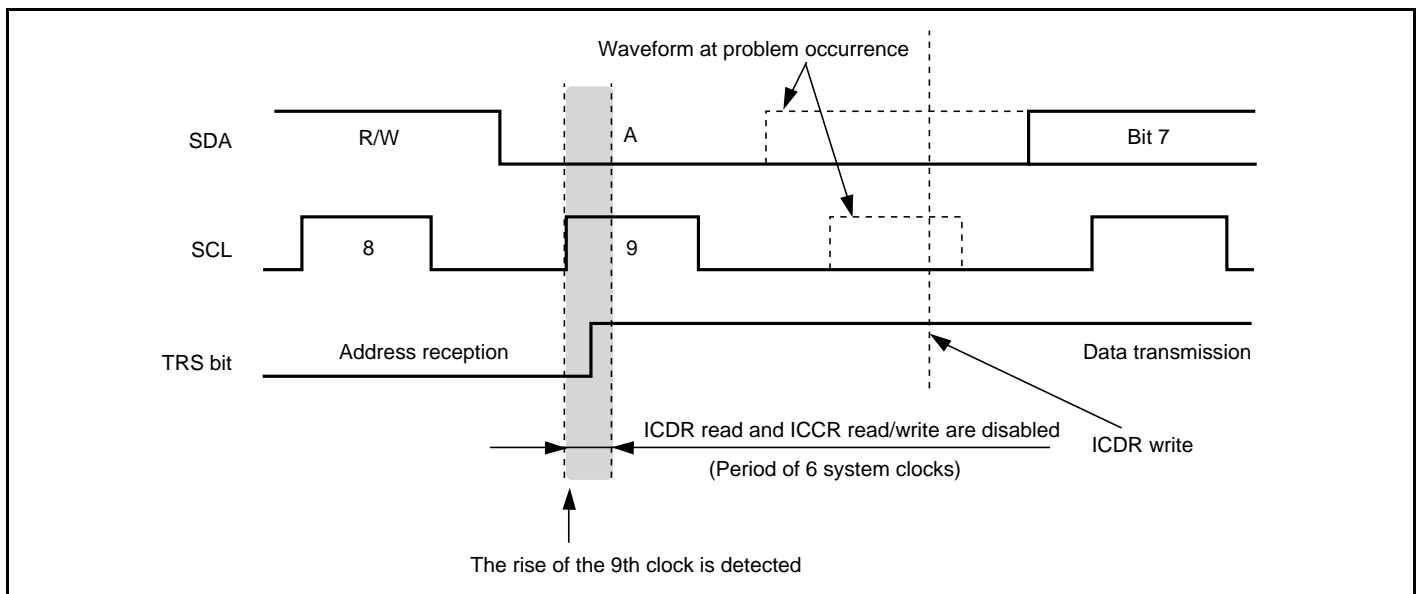
## 11. Note on ICDR read and ICCR access in slave transmit mode

In I<sup>2</sup>C bus interface slave transmit mode, do not read from ICDR or do not read from or write to ICCR during the time shaded in figure 17.26.

However, such read and write operations cause no problem in interrupt handling processing that is generated in synchronization with the rising edge of the 9th clock pulse because the shaded time has passed before making the transition to interrupt handling.

To handle interrupts securely, be sure to keep either of the following conditions.

- Read ICDR data that has been received so far or read from or write to ICCR before starting the receive operation of the next slave address.
- Monitor the BC2 to BC0 counter in ICMR; when the count is 000 (8th or 9th clock pulse), wait for at least two transfer clock times in order to read from ICDR or read from or write to ICCR during the time other than the shaded time.



**Figure 17.26 ICDR Read and ICCR Access Timing in Slave Transmit Mode**

## 12. Note on TRS bit setting in slave mode

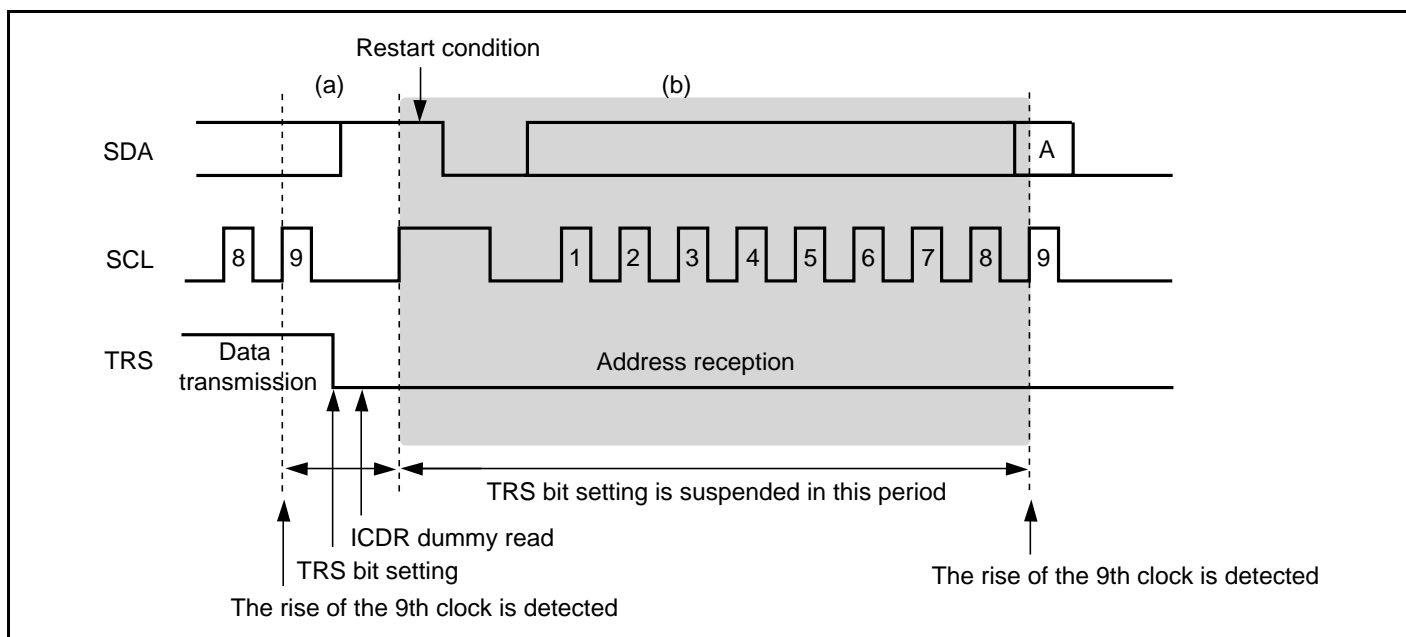
In I<sup>2</sup>C bus interface slave mode, if the TRS bit value in ICCR is set after detecting the rising edge of the 9th clock pulse or the stop condition before detecting the next rising edge on the SCL pin (the time indicated as (a) in figure 17.27), the bit value becomes valid immediately when it is set.

However, if the TRS bit is set during the other time (the time indicated as (b) in figure 17.27), the bit value is suspended and remains invalid until the rising edge of the 9th clock pulse or the stop condition is detected.

Therefore, when the address is received after the restart condition is input without the stop condition, the effective TRS bit value remains 1 (transmit mode) internally and thus the acknowledge bit is not transmitted after the address has been received at the 9th clock pulse.

To receive the address in slave mode, clear the TRS bit to 0 during the time indicated as (a) in figure 17.27.

To release the SCL low level that is held by means of the wait function in slave mode, clear the TRS bit to and then dummy-read ICDR.



**Figure 17.27 TRS Bit Set Timing in Slave Mode**

### 13. Note on ICDR read in transmit mode and ICDR write in receive mode

When ICDR is read in transmit mode (TRS = 1) or ICDR is written to in receive mode (TRS = 0), the SCL pin may not be held low in some cases after transmit/receive operation has been completed, thus inconveniently allowing clock pulses to be output on the SCL bus line before ICDR is accessed correctly.

To access ICDR correctly, read the ICDR after setting receive mode or write to the ICDR after setting transmit mode.

### 14. Note on ACKE and TRS bits in slave mode

In the I<sup>2</sup>C bus interface, if 1 is received as the acknowledge bit value (ACKB = 1) in transmit mode (TRS = 1) and then the address is received in slave mode without performing appropriate processing, interrupt handling may start at the rising edge of the 9th clock pulse even when the address does not match.

Similarly, if the start condition or address is transmitted from the master device in slave transmit mode (TRS = 1) and 1 is received as the acknowledge bit value (ACKB = 1), the IRIC flag may be set thus causing an interrupt source even when the address does not match.

To use the I<sup>2</sup>C bus interface module in slave mode, be sure to follow the procedures below.



1. When having received 1 as the acknowledge bit value for the last transmit data at the end of a series of transmit operation, clear the ACKE bit in ICCR to 0 once to initialize the ACKB bit to 0.
2. Set receive mode (TRS = 0) before the next start condition is input in slave mode. Complete transmit operation by the procedure shown in figure 17.21, in order to switch from slave transmit mode to slave receive mode.

#### 15. Notes on WAIT function

##### (a) Conditions to cause this phenomenon

When both of the following conditions are satisfied, the clock pulse of the 9th clock could be outputted continuously in master mode using the WAIT function due to the failure of the WAIT insertion after the 8th clock fall.

- (1) Setting the WAIT bit of the ICMR register to 1 and operating WAIT, in master mode
- (2) If the IRIC bit of interrupt flag is cleared from 1 to 0 between the fall of the 7th clock and the fall of the 8th clock.

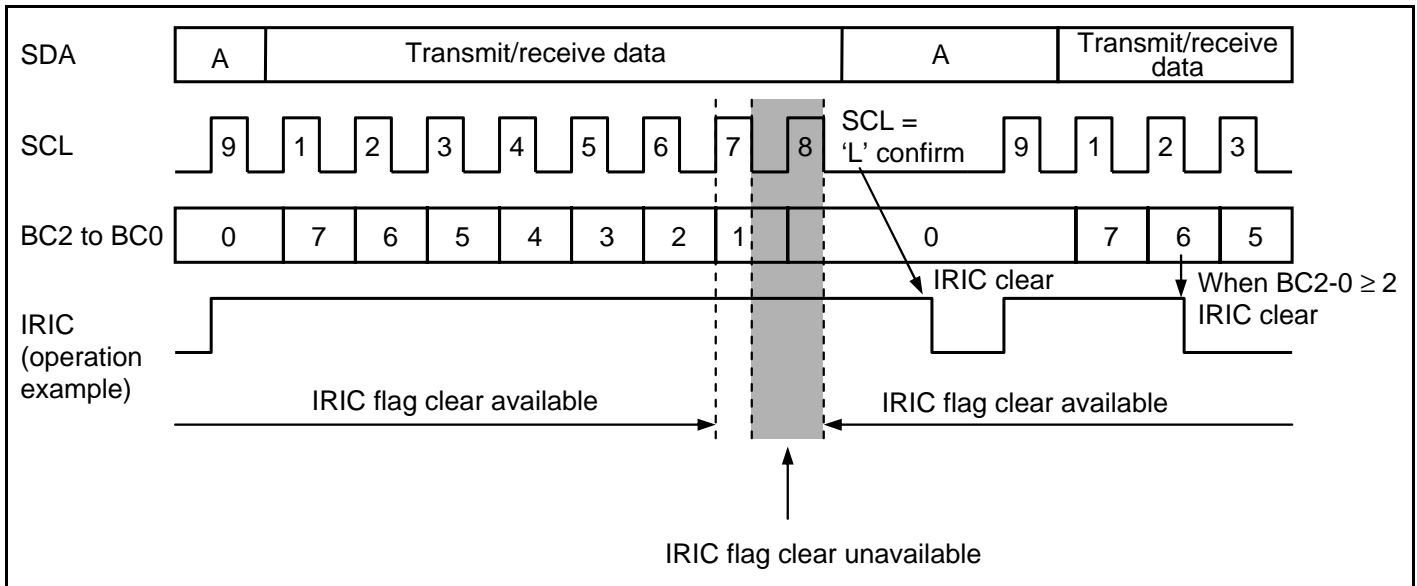
##### (b) Error phenomenon

Normally, WAIT State will be cancelled by clearing the IRIC flag bit from 1 to 0 after the fall of the 8th clock in WAIT State. In this case, if the IRIC flag bit is cleared between the 7th clock fall and the 8th clock fall, the IRIC flag-clear data will be retained internally. Therefore, the WAIT State will be cancelled right after WAIT insertion on 8th clock fall.

##### (c) Restrictions

Please clear the IRIC flag before the rise of the 7th clock (the counter value of BC2 through BC0 should be 2 or greater), after the IRIC flag is set to 1 on the rise of the 9th clock.

If the IRIC flag-clear is delayed due to the interrupt or other processes and the value of BC counter is turned to 1 or 0, please confirm the SCL pins are in L' state after the counter value of BC2 through BC0 is turned to 0, and clear the IRIC flag. (See figure 17.28.)



**Figure 17.28 IRIC Flag Clear Timing on WAIT Operation**

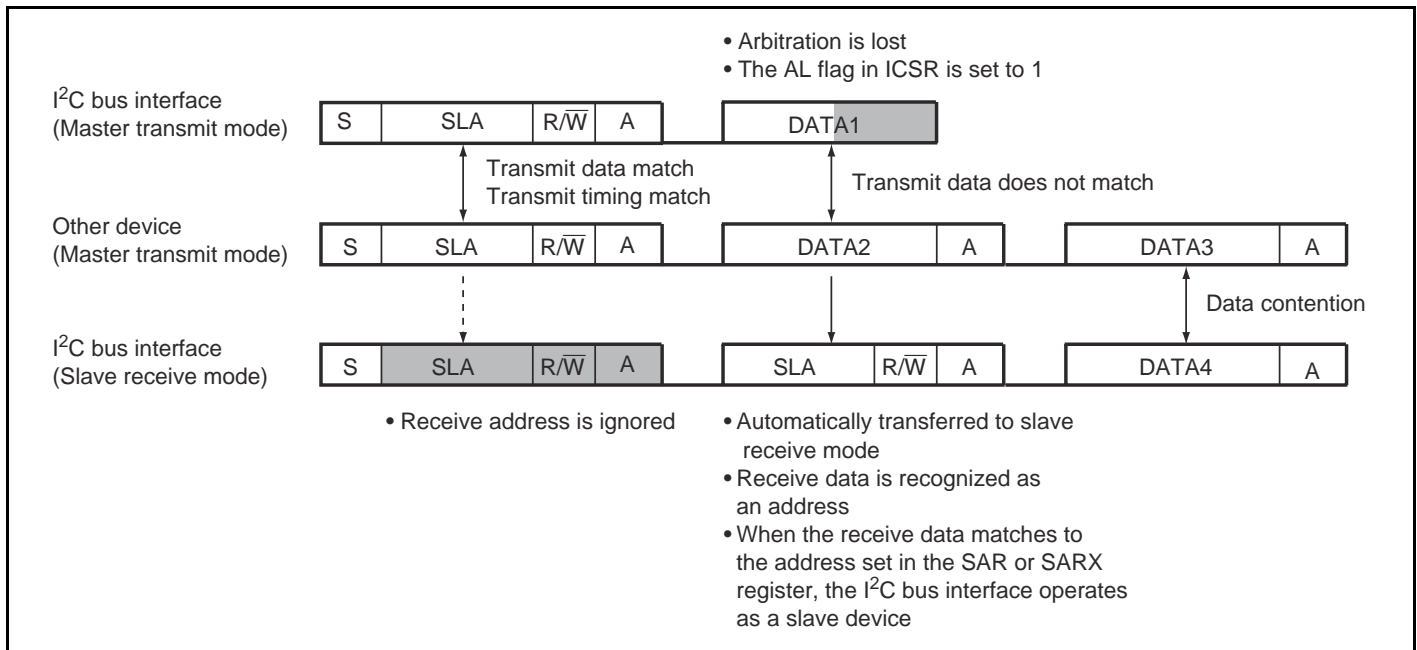
## 16. Notes on Arbitration Lost

The I<sup>2</sup>C bus interface recognizes the data in transmit/receive frame as an address when arbitration is lost in master mode and a transition to slave receive mode is automatically carried out.

When arbitration is lost not in the first frame but in the second frame or subsequent frame, transmit/receive data that is not an address is compared with the value set in the SAR or SARX register as an address. If the receive data matches with the address in the SAR or SARX register, the I<sup>2</sup>C bus interface erroneously recognizes that the address call has occurred. (See figure 17.29.)

In multi-master mode, a bus conflict could happen. When The I<sup>2</sup>C bus interface is operated in master mode, check the state of the AL bit in the ICSR register every time after one frame of data has been transmitted or received.

When arbitration is lost during transmitting the second frame or subsequent frame, take avoidance measures.



**Figure 17.29 Diagram of Erroneous Operation when Arbitration Is Lost**

Though it is prohibited in the normal I<sup>2</sup>C protocol, the same problem may occur when the MST bit is erroneously set to 1 and a transition to master mode is occurred during data transmission or reception in slave mode. In multi-master mode, pay attention to the setting of the MST bit when a bus conflict may occur. In this case, the MST bit in the ICCR register should be set to 1 according to the order below.

- Make sure that the BBSY flag in the ICCR register is 0 and the bus is free before setting the MST bit.
- Set the MST bit to 1.
- To confirm that the bus was not entered to the busy state while the MST bit is being set, check that the BBSY flag in the ICCR register is 0 immediately after the MST bit has been set.

Note: Above restriction can be cleared by setting bits FNC1 and FNC0 in the ICXR register.



## Section 18 Universal Serial Bus Interface (USB)

This LSI incorporates a universal serial bus (conforms to USB standard Rev. 1.1) function module. A USB is an interface for connecting personal computer peripheral devices. There are various types of peripheral devices on the market. To support this wide range of devices, the USB in this LSI provides several device classes such as MassStorage class and HID (Human Interface Device) class for end points that support control transfer, interrupt transfer, and bulk transfer.

### 18.1 Features

- USB standard Rev. 1.1 compliant
- USB function core executed by standard commands  
Executed by interpreting device class commands by the CPU (Firmware must be created.)
- Compound function consisting of HID device class and MassStorage device class
- EP0: USB control endpoint (for control transfer)  
EP0S, EP0I, and EP0O can use their specific FIFOs.
- EP1, EP2: HID control endpoint (for interrupt transfer)  
EP1 and EP2 use specific FIFOs.
- EP3: MassStorage control endpoint (for interrupt transfer)  
EP3 uses a specific FIFO.
- EP4, EP5: MassStorage data endpoint (for bulk transfer)  
EP4 and EP5 use a maximum of 2048-byte RFU-FIFOs.
- Full speed (12 Mbps) support
- System clock and external clock division/multiplication circuit
- Bus driver/receiver incorporated  
Driven by DrVCC/DrVSS power supply

Figure 18.1 shows the block diagram of the USB.

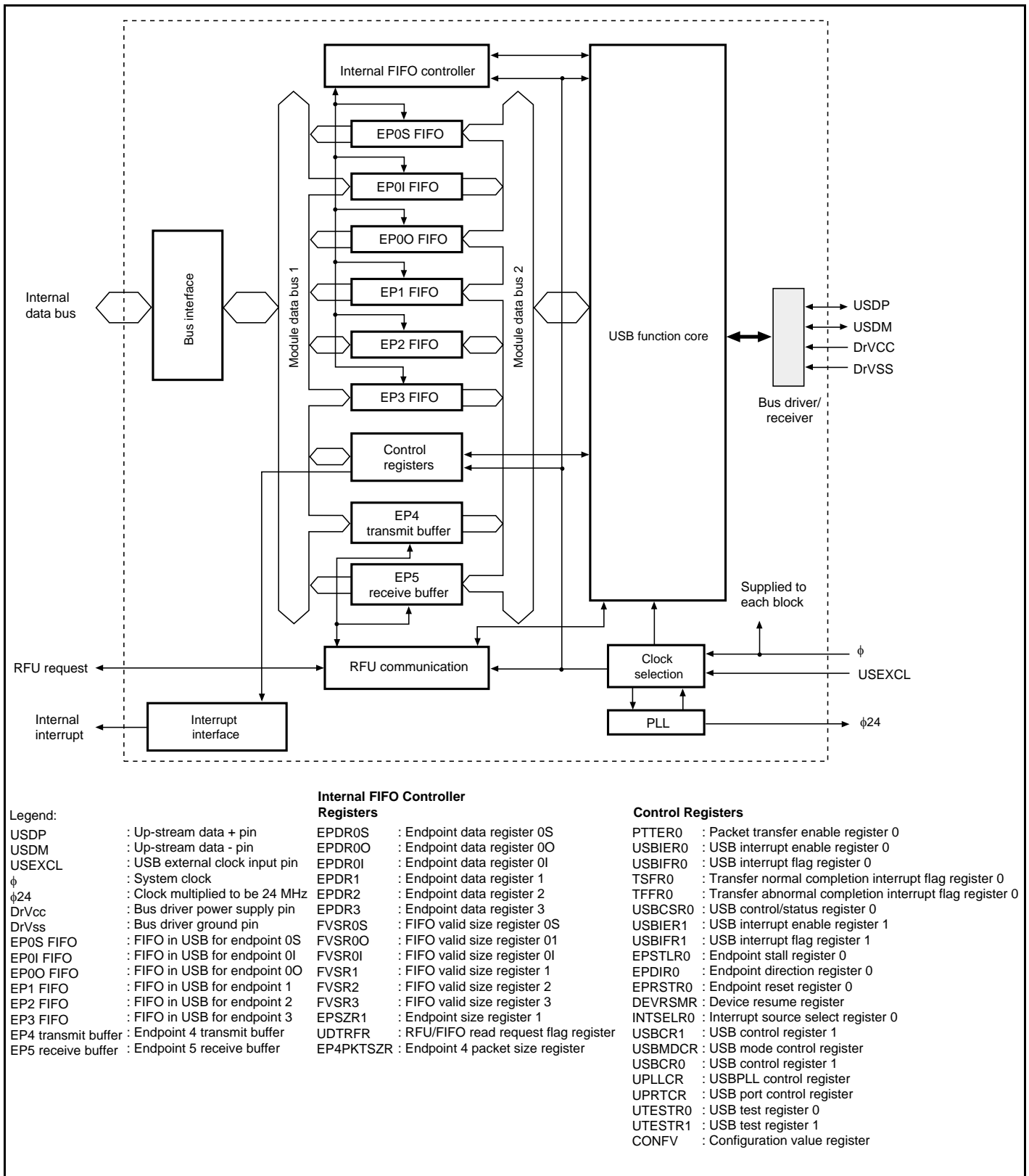


Figure 18.1 Block Diagram of USB

## 18.2 Input/Output Pins

Table 18.1 shows the USB pin configuration.

**Table 18.1 Pin Configuration**

Name	Symbol	Input/Output	Function
External clock input pin	USEXCL	Input	USB external clock input pin
Up-stream data + pin	USDP	Input/Output	I/O pin for USB serial data
Up-stream data - pin	USDM	Input/Output	I/O pin for USB serial data
Bus driver power supply pin	DrVCC	Input	On-chip bus driver/receiver power supply pin
Bus driver ground pin	DrVSS	Input	On-chip bus driver/receiver ground pin

## 18.3 Register Descriptions

In the USB protocol, the host sends a token to initiate a unit of data transmission (transaction). A transaction consists of a token packet, data packet, and handshake packet. A token packet contains information relating to the device addresses and endpoints, and transfer type. A data packet contains data. A handshake packet includes information relating to the transmission success or failure.

To transfer data from the host to the slave, the host first sends an OUT or SETUP token and then sends data to the slave (OUT transaction or SETUP transaction). To transfer data from the slave to the host, the host first sends an IN token to the slave and then waits for data sent from the slave (IN transaction).

In the following descriptions, IN and OUT operations based on the host may be described as input and output. In addition, host input transfer may be referred to as IN (IN transaction, IN-FIFO, or EP0in). Host output transfer may be referred to as OUT (OUT transaction, OUT-FIFO, or EP0out).

On the other hand, transmission (send) and reception (receive) mean transmission and reception based on the USB module or slave CPU unless it is noted as transmission by the host or reception by the host.

The USB has the following registers.

- Endpoint size register 1 (EPSZR1)
- Endpoint data register 0S (EPDR0S)
- Endpoint data register 0O (EPDR0O)
- Endpoint data register 0I (EPDR0I)
- Endpoint data register 1 (EPDR1)
- Endpoint data register 2 (EPDR2)
- Endpoint data register 3 (EPDR3)
- FIFO valid size register 0S (FVSR0S)
- FIFO valid size register 0O (FVSR0O)
- FIFO valid size register 0I (FVSR0I)
- FIFO valid size register 1 (FVSR1)
- FIFO valid size register 2 (FVSR2)
- FIFO valid size register 3 (FVSR3)
- Endpoint direction register 0 (EPDIR0)
- Packet transfer enable register 0 (PTTER0)
- USB interrupt enable register 0 (USBIER0)
- USB interrupt enable register 1 (USBIER1)
- USB interrupt flag register 0 (USBIFR0)
- USB interrupt flag register 1 (USBIRF1)
- Transfer normal completion interrupt flag register 0 (TSFR0)
- Transfer abnormal completion interrupt flag register 0 (TFFR0)
- USB control/status register 0 (USBCSR0)
- Endpoint stall register 0 (EPSTLR0)
- Endpoint reset register 0 (EPRSTR0)
- Device resume register (DEVRSMR)
- Interrupt source select register 0 (INTSELR0)
- USB control register 0 (USBCR0)
- USB control register 1 (USBCR1)
- USBPLL control register (UPLLCR)
- Configuration value register (CONFV)
- Endpoint 4 packet size register (EP4PKTSZR)
- RFU/FIFO read request flag register (UDTRFR)
- USB mode control register (USBMDCR)



- USB port control register (UPRTCR)
- USB test register 0 (UTESTR0)
- USB test register 1 (UTESTR1)

### 18.3.1 USB Data FIFO

FIFOs combined with EPDRs intervene in data transfer between this LSI (slave CPU) and the USB function core. The USB function core performs data transmission to or from the USB host (host).

This LSI incorporates 80 bytes of specific FIFOs, and a RAM-FIFO unit (RFU) that can be used as a maximum of 4096 bytes of FIFOs by using on-chip RAM.

As shown in table 18.2, specific FIFOs can be used in two ways based on whether EP2 is used or not. In EP0I, EP0O, EP1, and EP2, the maximum packed size of the data packet is specified as half of the FIFO size (bytes). In EP0S and EP3, the maximum packet size of the data packet is equal to the FIFO size (bytes). EP0S is a specific FIFO for setup command reception, which is enabled or disabled by the SETICNT bit in USBMDCR. For details on RAM-FIFOs that form EP4 and EP5, refer to section 8, RAM-FIFO Unit (RFU).

In the host input transfer, all data items sent from the slave are written to a specific FIFO before the slave transmission is initiated. In the host output transfer, the host transfer is completed before the slave reads all data items from the specific FIFO.

**Table 18.2 FIFO Configuration**

Endpoint		Transfer Direction	FIFO Size	Configuration	Description
Endpoint 0	EP0S	OUT (SETUP)	8 bytes	8 bytes × 1	Specific FIFO
	EP0O	OUT	16 bytes	8 bytes × 2	
	EP0I	IN	16 bytes	8 bytes × 2	
Endpoint 1	EP1	IN	16 bytes	8 bytes × 2	
			32 bytes	16 bytes × 2	
Endpoint 2	EP2	IN/OUT	16 bytes	8 bytes × 2	
			0 bytes	Not used	
Endpoint 3	EP3	IN	8 bytes	8 bytes × 1	
Endpoint 4	EP4	IN	Max. 2048 bytes	Max. 64 bytes × 32	RAM-FIFO (RFU)
Endpoint 5	EP5	OUT	Max. 2048 bytes	Max. 64 bytes × 32	

### 18.3.2 Endpoint Size Register 1 (EPSZR1)

EPSZR1 specifies the sizes of the FIFO (number of bytes) used in USB function core endpoints 1 and 2.

To use both endpoints 1 and 2 in this LSI, the size of FIFOs used for endpoints 1 and 2 must be specified as 16 bytes. To use only endpoint 1 in this LSI, the size of FIFO used for endpoint 1 must be specified as 16 bytes or 32 bytes. If the size of FIFO used for endpoint 1 is specified as 32 bytes, the FIFO for endpoint 2 must be specified as 0 bytes.

EPSZR1 is initialized to H'44 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7	EP1SZ3	0	R/W	EP1 FIFO Size
6	EP1SZ2	1	R/W	0000 : Setting prohibited
5	EP1SZ1	0	R/W	0001 : Setting prohibited
4	EP1SZ0	0	R/W	0010 : Setting prohibited 0011 : Setting prohibited 0100 : FIFO size, 16 bytes 0101 : FIFO size, 32 bytes 0110 : Setting prohibited 0111 : Setting prohibited 1XXX: Setting prohibited
3	EP2SZ3	0	R/W	EP2 FIFO Size
2	EP2SZ2	1	R/W	0000 : FIFO size, 0 byte
1	EP2SZ1	0	R/W	0001 : Setting prohibited
0	EP2SZ0	0	R/W	0010 : Setting prohibited 0011 : Setting prohibited 0100 : FIFO size, 16 bytes 0101 : Setting prohibited 0110 : Setting prohibited 0111 : Setting prohibited 1XXX: Setting prohibited

Legend:

X: Don't care

### 18.3.3 Endpoint Data Registers 0S, 0O, 0I, 1, 2, and 3 (EPDR0S, EPDR0O, EPDR0I, EPDR1, EPDR2, and EPDR3)

EPDRs intervene in the data transfer between the CPU and FIFOs in each host input transfer or host output transfer for the USB function core endpoints 1 and 2. EPDR0I, EPDR1, and EPDR3 are write-only registers used for host input transfer. EPDR0S and EPDR0O are read-only registers used for host output transfer. EPDR2 is specified depending on the transfer direction specified in EPDIR0. If EPDIR0 is specified as a host input transfer, EPDR2 is specified as a read-only register; if EPDIR0 is specified as a host output transfer, EPDR2 is specified as a write-only register.

Data written in EPDR0I, EPDR1, EPDR2 (write-only register), and EPDR3 is stored in the FIFOs and is enabled by setting the EPTE bit in PTTERR0. This enabled data is transferred to the USB function core according to the USB function core's request and then sent to the host.

Data sent to the host is stored in the FIFO by the USB function core and is enabled by returning an ACK handshake after all bytes of data packet has been received. When reading EPDR0S, EPDR0O, or EPDR2 (read-only register), data is stored in the FIFO and then enabled data is read out in order of transfer.

EPDR is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Note that EPDR for endpoints 4 and 5 is not supported. Data is handled by reading on-chip RAM directly.

#### EPDR0S

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	R	Endpoint 0 is used for input or output transfer and EPDR0S is specified as a read-only register. EPDR0S is a specific FIFO for setup command reception and is enabled when the SETICNT bit in USBMDCR is set to 1. EPDR0S access is disabled when the SETICNT bit is cleared to 0.

#### EPDR0O

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	R	Endpoint 0 is used for input or output transfer and EPDR0O is specified as a read-only register.

**EPDR0I**

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	W	Endpoint 0 is used for input or output transfer and EPDR0I is specified as a write-only register.

**EPDR1**

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	W	Endpoint 1 is used for input transfer and EPDR1 is specified as a write-only register.

**EPDR2**

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	R or W	EPDR2 is specified as a write-only or read-only register depending on the transfer direction specified by EPDIR0.

**EPDR3**

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	All 0	W	Endpoint 3 is used for input transfer and EPDR3 is specified as a write-only register.

### 18.3.4 Endpoint Valid Size Registers 0S, 0O, 0I, 1, 2, and 3 (FVSR0S, FVSR0O, FVSR0I, FVSR1, FVSR 2, and FVSR3)

FVSRs indicate the number of valid data items stored in FIFOs in each endpoint of the USB function core. Since endpoints 4 and 5 do not use FIFOs in the USB interface, they perform data transfer by using the RFU which has the function corresponding to FVSRs.

In host input transfer, FVSR indicates the number of bytes that the slave CPU can write to the FIFO (FIFO size — number of bytes that are written to the FIFO by the slave CPU but which are not read (transmitted) by the USB function core). In host output transfer, FVSRs indicate the number of bytes that are written to the FIFO by the USB function core but which are not read by the slave CPU.

In host input transfer, FVSR is decremented by the number of bytes to be written if the slave CPU writes to EPDR and sets the EPTE bit in PTTER0; FVSR is incremented by the number of bytes to be read if the USB function core reads the FIFO and an ACK handshake is received from the host.

In host output transfer, the FVSR is incremented by the number of bytes to be written if the USB function core writes to the FIFO and sends an ACK handshake; FVSR is decremented by 1 if the slave CPU reads the EPDR.

In certain situations, data must be re-transferred if a transfer error occurs. In this case, the FVSR is not modified and the FIFO of the channel to be re-transferred will be re-wound.

In the USB protocol, the DATA0 and DATA1 packets are transmitted or received alternatively during data transfer for each endpoint. Accordingly, the success of the data transfer can be checked by the DATA0 and DATA1 packet toggles. If the DATA0 or DATA1 packet toggle is not performed correctly, the USB function core stops the transaction processing and the FVSR value does not change.

FVSR is a 2-byte register but can indicate the FIFO status only by using the lower byte, since the FIFOs used in this LSI are 8 bytes, 16 bytes or 32 bytes. Therefore, only the lower byte of FSVR must be read.

The upper byte of FSVR cannot be accessed directly. The upper byte of FSVR is sent to the temporary register when the lower byte of FSVR is read and the contents of the temporary register can be read when the upper byte of FSVR is read. If FSVR is read by word access, the contents of the upper byte indicates the value when the lower byte has been read out.

FVSR0S, FVSR0O, and FVSR0I are automatically specified as H'0000, H'0000, and H'0010, respectively when a SETUP token has been received.

FVSR is initialized by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)) according to the transfer direction and the FIFO size specified by EPDIR0 and EPSZR1.

**FVSR0SH, FVSR0OH, FVSR0IH, FVSR1H, FVSR2H, FVSR3H**

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Initial Values
6	—	0	R	FVSR0SH: H'00
5	—	0	R	FVSR0OH: H'00
4	—	0	R	FVSR0IH: H'00
3	—	0	R	FVSR1H: H'00
2	—	0	R	FVSR2H: H'00
1	N9	0	R	FVSR3H: H'00
0	N8	0	R	

**FVSR0SL, FVSR0OL, FVSR0IL, FVSR1L, FVSR2L, FVSR3L**

Bit	Bit Name	Initial Value	R/W	Description
7	N7	0	R	Initial Values
6	N6	0	R	FVSR0SL: H'00
5	N5	0	R	FVSR0OL: H'00
4	N4	0/1 <sup>*1</sup>	R	FVSR0IL: H'10
3	N3	0/1 <sup>*2</sup>	R	FVSR1L: H'10
2	N2	0	R	FVSR2L: H'10
1	N1	0	R	FVSR3L: H'08
0	N0	0	R	

Notes: 1. Initial value of the N4 bit is 0 in FVSR0S, FVSR0O, and FVSR3 and 1 in other FVSRs.  
 2. Initial value of the N3 bit is 1 in FVSR3 and 0 in other FVSRs.

### 18.3.5 Endpoint Direction Register 0 (EPDIR0)

EPDIR0 controls the direction of data transfer for endpoints other than endpoint 0 of the USB function core. In this LSI, EP1, EP3 and EP4 must be specified as host input transfers, EP5 as a host output transfer, and EP2 as either a host input transfer or host output transfer.

EPDIR0 is initialized to H'3C by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	EP5DIR	0	R/W	Endpoint 5 Data Direction Control Flag Controls the data transfer direction of endpoint 5. 0: Endpoint 5 is specified as host output transfer 1: Setting prohibited
5	EP4DIR	1	R/W	Endpoint 4 Data Direction Control Flag Controls the data transfer direction of endpoint 4. 0: Setting prohibited 1: Endpoint 4 is specified as host input transfer
4	EP3DIR	1	R/W	Endpoint 3 Data Direction Control Flag Controls the data transfer direction of endpoint 3. 0: Setting prohibited 1: Endpoint 3 is specified as host input transfer
3	EP2DIR	1	R/W	Endpoint 2 Data Direction Control Flag Controls the data transfer direction of endpoint 2. 0: Endpoint 2 is specified as host output transfer 1: Endpoint 2 is specified as host input transfer
2	EP1DIR	1	R/W	Endpoint 1 Data Direction Control Flag Controls the data transfer direction of endpoint 1. 0: Setting prohibited 1: Endpoint 1 is specified as host input transfer
1, 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

### 18.3.6 Packet Transfer Enable Register 0 (PTTER0)

PTTER0 controls the FIFO valid size register used in the host input transfer of the USB function core.

In the USB protocol, communication is performed using packets. The minimum unit of data transfer is a transaction. A transaction is comprised of a token packet, a data packet, and a handshake packet.

In host input transfer, the USB function core receives an IN token (packet). On receiving the IN token, the USB core must send a data packet when it is not stalled or a NAK handshake if no data exists.

If an EPTE bit is set to 1 after the slave CPU has written the data that is to be transferred to the host in the FIFO, the contents of FVSR is modified. This enables the transmission of data written in FIFO. By controlling the data transmission using an EPTE bit, erroneous data transmission during data write from slave CPU to FIFO can be prevented effectively.

The FIFO used for endpoint 4 is assigned to the on-chip RAM area controlled by the RFU. If the EP4TE bit is set to 1, data transmission is initiated by an IN token and then data in the FIFO is sent to a buffer in the USB interface (pre-read).

To transfer a data packet of 0 bytes to the host, set the corresponding EPTE bit to 1 in the RAM-FIFO empty state.



Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
5	EP4TE	0	R/(W)*	Endpoint 4 Packet Transmission Enable Prepares the transmission of RAM-FIFO data for endpoint 4 0: Normal read value. 1: Prepares the transmission of RAM-FIFO data for endpoint 4. The EP4TE bit must be set 1 for each IN transfer in one transaction.
4	EP3TE	0	R/(W)*	Endpoint 3 Packet Transmission Enable Updates FVSR3 for endpoint 3 0: Normal read value. 1: Updates FVSR3 in endpoint 3-specific FIFO.
3	EP2TE	0	R/(W)*	Endpoint 2 Packet Transmission Enable Modifies FVSR2 for endpoint 2 if the EP2DIR bit is set to 1. 0: Normal read value. 1: Updates FVSR2 in endpoint 2-specific FIFO.
2	EP1TE	0	R/(W)*	Endpoint 1 Packet Transmission Enable Updates FVSR2 for endpoint 1. 0: Normal read value. 1: Updates FVSR1 in endpoint 1-specific FIFO.
1	EP0ITE	0	R/(W)*	Endpoint 1 Packet Transmission Enable Updates FVSR0I for endpoint 0. 0: Normal read value. 1: Updates FVSR0I in endpoint 0-specific FIFO.
0	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

Note: \* Only 1 can be written.

### 18.3.7 USB Interrupt Enable Registers 0 and 1 (USBIER0, USBIER1)

USBIER0 and USBIER1 provide interrupt enable bits that allow interrupt requests from the USB function core to the slave CPU.

USBIER0 and USBIER1 are initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

### USBIER0

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	UDTRE	0	R/W	RFU/FIFO Read Request Interrupt Enable 0: Disables EP5 RFU/FIFO read request interrupt 1: Enables EP5 RFU/FIFO read request interrupt
5	BRSTE	0	R/W	Bus Reset Interrupt Enable 0: Disables bus reset interrupt of USB function core 1: Enables bus reset interrupt of USB function core
4	SOFE	0	R/W	SOF Interrupt Enable 0: Disables USB function core SOF (start of frame) interrupt 1: Enables USB function core SOF (start of frame) interrupt
3	SPNDE	0	R/W	Suspend Interrupt Enable 0: Disables USB function core suspend OUT and IN interrupts 1: Enables USB function core suspend OUT and IN interrupts
2	TFE	0	R/W	Transfer Abnormal Completion Interrupt Enable 0: Disables transfer abnormal completion interrupt of the USB function core 1: Enables transfer abnormal completion interrupt of the USB function core
1	TSE	0	R/W	Transfer Normal Completion Interrupt Enable 0: Disables transfer normal completion interrupt of the USB function core 1: Enables transfer normal completion interrupt of the USB function core
0	SETUPE	0	R/W	Setup Interrupt Enable 0: Disables USB function core setup interrupt 1: Enables USB function core setup interrupt

**USBIER1**

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
1	SETCE	0	R/W	SetConfiguration Command Detection Interrupt Enable 0: Disables SetConfiguration command detection interrupt 1: Enables SetConfiguration command detection interrupt
0	SETIE	0	R/W	SetInterface Command Detection Interrupt Enable 0: Disables SetInterface command detection interrupt 1: Enables SetInterface command detection interrupt

**18.3.8 USB Interrupt Flag Registers 0 and 1 (USBIFR0, USBIFR1)**

USBIFR0 and USBIFR1 have interrupt flags which generate interrupts from the USB module to the slave CPU.

The USB module supports four interrupt sources: USBIA, USBIB, USBIC, and USBID. USBIA is specific to a setup interrupt. USBIB and USBIC can be used for either transfer normal completion interrupts or transfer abnormal completion interrupts. USBID can be used for other interrupts such as transfer normal completion interrupts, transfer abnormal completion interrupts, RFU/FIFO read request interrupts, bus reset interrupts, SOF interrupts, suspend OUT interrupts, suspend IN interrupts, SetConfiguration command detection interrupts, and SetInterface command detection interrupts.

USBIFR0 and USBIFR1 are initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

**USBIFR0**

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	TS	0	R	<p>Transfer Normal Completion Interrupt Status</p> <p>Indicates that data transfer for an USB core endpoint has been completed normally.</p> <p>If the TSE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU. In this case, if an interrupt source whose TS bit is set to 1 is specified to request an USBIB or USBIC interrupt, the USBIB or USBIC interrupt is processed prior to the USBID interrupt according to the interrupt priority specified in the slave CPU interrupt controller.</p> <p>0: All bits in TSFR0 are cleared to 0. 1: At least one bit in TSFR0 is set to 1.</p>
6	TF	0	R	<p>Transfer Abnormal Completion Interrupt Status</p> <p>Indicates that data transfer for an USB core endpoint has been completed abnormally.</p> <p>If the TFE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU. In this case, if an interrupt source whose TF bit is set to 1 is specified to request an USBIB or USBIC interrupt, the USBIB or USBIC interrupt is processed prior to the USBID interrupt according to the interrupt priority specified in the slave CPU interrupt controller.</p> <p>0: All bits in TFFR0 are cleared to 0. 1: At least one bit in TFFR0 is set to 1.</p>
5	UDTRF	0	R	<p>RFU/FIFO Read Request Interrupt Status</p> <p>Indicates that the host output transfer (out transaction) has been completed normally and that the RAM-FIFO is full and the receive buffer contains data.</p> <p>If the UDTRE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>0: All bits in UDTRFR are cleared to 0. 1: At least one bit in UDTRFR is set to 1.</p>

Bit	Bit Name	Initial Value	R/W	Description
4	BRSTF	0	R/(W)*	<p><b>Bus Reset Interrupt Status</b></p> <p>Indicates that the USB function core detects a bus reset by an up-stream.</p> <p>If the BRSTE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after BRSTF = 1 has been read.</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The USB function core detects a bus reset by an up-stream.</li> </ul>
3	SOFF	0	R/(W)*	<p><b>SOF Interrupt Status</b></p> <p>Indicates that the USB function core detects an SOF (Start of Frame) by an up-stream.</p> <p>If the SOFE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after SOFF = 1 has been read.</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The USB function core detects an SOF (Start of Frame).</li> </ul>
2	SPNDOF	0	R/(W)*	<p><b>Suspend OUT Interrupt Status</b></p> <p>Indicates that the USB function core detects a bus state transition from suspend state to normal state.</p> <p>If the SPNDE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after SPNDOF = 1 has been read.</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The USB function core detects a bus state transition from suspend state to normal state.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
1	SPNDIF	0	R/(W)*	<p>Suspend IN Interrupt Status</p> <p>Indicates that the USB function core detects an idle state for a specific period or more, and detects a bus state transition from normal state to suspend state.</p> <p>If the SPNDE bit in USBIER0 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after SPNDIF = 1 has been read.</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The USB function core detects a bus state transition from normal state to suspend state.</li> </ul>
0	SETUPF	0	R/(W)*	<p>Setup Interrupt Status</p> <p>The meaning of this bit differs depending on the SETICNT bit in USBMDCR.</p> <p>When SETICNT = 0, Indicates that endpoint 0 of the USB function core receives a SETUP token.</p> <p>When SETICNT = 1, Indicates that endpoint 0 of the USB function core receives a setup command that must be decoded by the slave CPU.</p> <p>If the SETUPE bit in USBIER0 is set to 1, an USBIA interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after SETUPF = 1 has been read.</li> </ul> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>Endpoint 0 of the USB function core receives a SETUP token (when SETICNT = 0).</li> <li>Endpoint 0 of the USB function core receives a setup command to be decoded by the slave CPU (when SETICNT = 1).</li> </ul>

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

**USBIFR1**

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
1	SETC	0	R/(W)*	<p>SetConfiguration Command Detection Interrupt Status</p> <p>Indicates that the USB function core detects a SetConfiguration command.</p> <p>If the SETCE bit in USBIER1 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after SETC = 1 has been read.</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The USB function core detects a SetConfiguration command.</li> </ul>
0	SETI	0	R/(W)*	<p>SetInterface Command Detection Interrupt Status</p> <p>Indicates that the USB function core detects a SetInterface command.</p> <p>If the SETIE bit in USBIER1 is set to 1, an USBID interrupt is requested to the slave CPU.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to after SETI = 1 has been read.</li> </ul> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>The USB function core detects a SetConfiguration command.</li> </ul>

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

### 18.3.9 Transfer Normal Completion Interrupt Flag Register 0 (TSFR0)

TSFR0 provides status flags indicating that the host input or host output transaction of each USB function core endpoint has been completed normally.

The normal completion of a transaction can be detected by an ACK handshake reception in host input transfer or by an ACK handshake transmission in host output transfer.

TSFR0 is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	EP5TS	0	R/(W)*	<p>Endpoint 5 Transfer Success Flag</p> <p>Indicates that the endpoint 5 host output transfer has been completed normally.</p> <p>When host output transfer is completed normally while the RAM-FIFO is full and data still remains in the receive buffer, this bit is not set to 1 and the EP5UDTR bit in UDTRFR is set to 1. For details, see section 18.3.20, RFU/FIFO Read Request Flag Register (UDTRFR).</p> <p>0: Indicates that the endpoint 5 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP5TS after EP5TS = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 5 host output transfer (OUT transaction) has been completed normally. [Setting condition]</p> <ul style="list-style-type: none"> <li>An ACK handshake has been achieved (ACK transmission) after OUT token reception and data transfer, and FIFO operation by the RFU has been completed normally.</li> </ul>
5	EP4TS	0	R/(W)*	<p>Endpoint 4 Transfer Success Flag</p> <p>Indicates that the endpoint 4 host input transfer has been completed normally.</p> <p>0: Indicates that the endpoint 4 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP4TS after EP4TS = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 4 host input transfer (IN transaction) has been completed normally. [Setting condition]</p> <ul style="list-style-type: none"> <li>An ACK handshake has been achieved (ACK transmission) after IN token reception and data transfer, and FIFO operation by the RFU has been completed normally.</li> </ul>



Bit	Bit Name	Initial Value	R/W	Description
4	EP3TS	0	R/(W)*	<p>Endpoint 3 Transfer Success Flag</p> <p>Indicates that the endpoint 3 host input transfer has been completed normally.</p> <p>0: Indicates that the endpoint 3 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP3TS after EP3TS = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 3 host input transfer (IN transaction) has been completed normally. [Setting condition]</p> <ul style="list-style-type: none"> <li>An ACK handshake has been achieved (ACK reception) after IN token reception and data transfer.</li> </ul>
3	EP2TS	0	R/(W)*	<p>Endpoint 2 Transfer Success Flag</p> <p>Indicates that the endpoint 2 host input/output transfer has been completed normally.</p> <p>0: Indicates that the endpoint 2 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP2TS after EP2TS = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 2 host input transfer (IN transaction) or host output transfer (out transaction) has been completed normally. [Setting conditions]</p> <ul style="list-style-type: none"> <li>An ACK handshake has been achieved (ACK transmission) after IN token reception and data transfer.</li> <li>An ACK handshake has been achieved (ACK transmission) after OUT token reception data transfer.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	EP1TS	0	R/(W)*	<p>Endpoint 1 Transfer Success Flag</p> <p>Indicates that the endpoint 1 host input transfer has been completed normally.</p> <p>0: Indicates that the endpoint 1 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP1TS after EP1TS = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 1 host input transfer (IN transaction) has been completed normally. [Setting condition]</p> <ul style="list-style-type: none"> <li>An ACK handshake has been achieved (ACK reception) after IN token reception and data transfer.</li> </ul>
1	EP0ITS	0	R/(W)*	<p>Endpoint 0 Transfer Success Flag</p> <p>Indicates that the endpoint 0 host input transfer has been completed normally.</p> <p>0: Indicates that the endpoint 0 is in a transfer wait state. [Clearing conditions]</p> <ul style="list-style-type: none"> <li>0 is written to EP0ITS after EP0ITS = 1 has been read.</li> <li>The endpoint 0 has received a SETUP token</li> </ul> <p>1: Indicates that the endpoint 1 host input transfer (IN transaction) has been completed normally. [Setting condition]</p> <ul style="list-style-type: none"> <li>An ACK handshake has been achieved (ACK reception) after IN token reception and data transfer.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
0	EP0OTS	0	R/(W)*	<p>Endpoint 0 Host Output Transfer Success Flag</p> <p>Indicates that the endpoint 0 host output transfer has been completed normally.</p> <p>Endpoint 0 host output transfer has two transactions: OUT transaction and SETUP transaction. Data transmission in these transactions are the same, but flag handling in these transactions differ.</p> <p>Since most of the commands sent by a SETUP transaction are processed in the USB function core, the EP0OTS flag is not set and the EP0OTF flag in TFFR0 is set to 1. For a command that cannot be processed in the USB core, the EP0OTS flag is set to 1. Note that neither the EP0OTS nor the EP0OTF flag is set to 1 if the SEICNT bit in USBMDCR is set to 1 regardless of whether the command can be processed in the USB core or not.</p> <p>0: Indicates that the end point 0 is in a host output transfer wait state.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• 0 is written to EP0OTS after EP0OTS = 1 has been read.</li> <li>• Endpoint 0 has received a SETUP token.</li> </ul> <p>1: Indicates that the endpoint 0 host output transfer (OUT transaction or SETUP transaction) has been completed normally.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• An ACK handshake has been achieved (ACK transmission) after OUT token reception and data transfer.</li> <li>• A received command needs to be processed in the slave CPU after a SETUP token has been received (only when the SETICNT bit is cleared to 0).</li> </ul>

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

### 18.3.10 Transfer Abnormal Completion Interrupt Flag Register 0 (TFFR0)

TFFR0 provides status flags (EPTF) indicating that the host input or host output transaction of each USB function core endpoint has been completed abnormally.

The abnormal completion of a transaction will be detected if a NAK handshake has been received or a NAK handshake has been sent because no transfer data has been received (FVSR = FIFO size: FIFO empty) in host input transfer, if a NAK handshake has been sent because the FIFO is full during data transfer in host output transfer, or if other transfer errors (DATA0/DATA1 toggle error, bit staffing error, bit count error, CRC error, and a transfer whose byte size exceeds Max Packet Size, etc.) occur.

TFFR0 is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	EP5TF	0	R/(W)*	Endpoint 5 Transfer Failure Flag Indicates that the endpoint 5 host output transfer has been completed abnormally. 0: Indicates that the endpoint 5 is in a transfer wait state. [Clearing condition] <ul style="list-style-type: none"> <li>0 is written to EP5TF after EP5TF = 1 has been read.</li> </ul> 1: Indicates that the endpoint 5 host output transfer (OUT transaction) has been completed abnormally. [Setting conditions] <ul style="list-style-type: none"> <li>Data cannot be received because both receive buffers for RFU and FIFO become full after an OUT token has been received (NAK transmission).</li> <li>A communication error occurs after an OUT token has been received.</li> </ul>
5	EP4TF	0	R/(W)*	Endpoint 4 Transfer Failure Flag Indicates that the endpoint 4 host input transfer has been completed abnormally. 0: Indicates that the endpoint 4 is in a transfer wait state. [Clearing condition] <ul style="list-style-type: none"> <li>0 is written to EP4TF after EP4TF = 1 has been read.</li> </ul> 1: Indicates that the endpoint 4 host input transfer (IN transaction) has been completed abnormally. [Setting conditions] <ul style="list-style-type: none"> <li>An ACK handshake cannot be achieved after an IN token has been received and data has been transferred.</li> <li>Data cannot be sent because both receive buffers for RFU and FIFO are empty after an IN token has been received (NAK transmission).</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
4	EP3TF	0	R/(W)*	<p>Endpoint 3 Transfer Failure Flag</p> <p>Indicates that the endpoint 3 host input transfer has been completed abnormally.</p> <p>0: Indicates that the endpoint 3 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP3TF after EP3TF = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 3 host input transfer (IN transaction) has been completed abnormally. [Setting conditions]</p> <ul style="list-style-type: none"> <li>An ACK handshake cannot be achieved after IN token has been received and data has been transferred.</li> <li>Data cannot be sent because the FIFO is empty after an IN token has been received (NAK transmission).</li> </ul>
3	EP2TF	0	R/(W)*	<p>Endpoint 2 Transfer Failure Flag</p> <p>Indicates that the endpoint 2 host input or output transfer has been completed abnormally.</p> <p>0: Indicates that the endpoint 2 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP2TF after EP2TF = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 2 host input transfer (IN transaction) or output transfer (OUT transaction) has been completed abnormally. [Setting conditions]</p> <ul style="list-style-type: none"> <li>An ACK handshake cannot be achieved after IN token has been received and data has been transferred.</li> <li>Data cannot be sent because the FIFO is empty after an IN token has been received (NAK transmission).</li> <li>Data cannot be received because the FIFO is full after an OUT token has been received (NAK transmission).</li> <li>A communication error occurs after the OUT token has been received.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	EP1TF	0	R/(W)*	<p>Endpoint 1 Transfer Failure Flag</p> <p>Indicates that the endpoint 1 host input transfer has been completed abnormally.</p> <p>0: Indicates that the endpoint 1 is in a transfer wait state. [Clearing condition]</p> <ul style="list-style-type: none"> <li>0 is written to EP1TF after EP1TF = 1 has been read.</li> </ul> <p>1: Indicates that the endpoint 1 host input transfer (IN transaction) has been completed abnormally. [Setting conditions]</p> <ul style="list-style-type: none"> <li>An ACK handshake cannot be achieved after an IN token has been received and data has been transferred.</li> <li>Data cannot be sent because the FIFO is empty after an IN token has been received (NAK transmission).</li> </ul>
1	EP0ITF	0	R/(W)*	<p>Endpoint 0 Transfer Failure Flag</p> <p>Indicates that the endpoint 0 host input transfer has been completed abnormally.</p> <p>0: Indicates that the endpoint 0 is in a transfer wait state. [Clearing conditions]</p> <ul style="list-style-type: none"> <li>0 is written to EP0ITF after EP0ITF = 1 has been read.</li> <li>Endpoint 0 has received a SETUP token.</li> </ul> <p>1: Indicates that the endpoint 0 host input transfer (IN transaction) has been completed abnormally. [Setting conditions]</p> <ul style="list-style-type: none"> <li>An ACK handshake cannot be achieved after an IN token has been received and data has been transferred.</li> <li>Data cannot be sent because the FIFO is empty after an IN token has been received (NAK transmission).</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
0	EP0OTF	0	R/(W)*	<p>Endpoint 0 Host Output Transfer Failure Flag</p> <p>Indicates that the endpoint 0 host output transfer has been completed abnormally.</p> <p>Endpoint 0 host output transfer has two transactions: OUT transaction and SETUP transaction. Data transmission in these transactions are the same, but flag handling in these transactions differ.</p> <p>Since most of the commands sent by a SETUP transaction are processed in the USB function core, the EP0OTS flag in TSFR0 is not set and the EP0OTF flag is set to 1. For a command that cannot be processed in the USB core, the EP0OTS flag is set to 1. Note that neither the EP0OTS nor the EP0OTF flag is set to 1 if the SEICNT bit in USBMDCR is set to 1 regardless of whether the command can be processed in the USB core or not.</p> <p>0: Indicates that the endpoint 0 is in a transfer wait state.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>0 is written to EP0OTF after EP0OTF = 1 has been read.</li> <li>Endpoint 0 has received a SETUP token.</li> </ul> <p>1: Indicates that the endpoint 0 host output transfer (OUT transaction or SETUP transaction) has been completed abnormally.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>Data cannot be sent because the FIFO is full after an OUT token has been received.</li> <li>Data cannot be sent because EP0OTC = 0 after an OUT token has been received (NAK transmission).</li> <li>A communication error occurs after the OUT token has been received.</li> <li>A received command can be processed in the USB function core (only when the SETICNT bit is cleared to 0) after a SETUP token has been received.</li> </ul>

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



### 18.3.11 USB Control /Status Register 0 (USBCSR0)

USBCSR0 controls the operation of the USB function core.

USBCSR0 is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
3	EP0STOP	0	R/W	<p>Endpoint 0 Stop</p> <p>Used to protect the contents of endpoint 0 FIFO in the USB function core. Setting this bit to 1 protects the data that is sent to the EP0 OUT-FIFO by a SETUP transaction.</p> <p>0: Indicates that the EP0 OUT-FIFO and specific FIFO is in an operating state.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• System reset</li> <li>• Function software reset</li> </ul> <p>1: Indicates that the EP0 OUT-FIFO is in a read stop state.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• FVSR00 contents are not changed by reading EPDR00.</li> </ul> <p>Indicates that the EP0 specific FIFO is in a write or transfer stop state.</p> <ul style="list-style-type: none"> <li>• FIFO contents are not changed by writing to EPDR0I.</li> <li>• FVSR0I contents are not changed by setting EP0ITE.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	EPIVLD	0	R/(W)*	<p>Endpoint Information Valid</p> <p>Enables USB function core operation.</p> <p>To enable the USB function core operation, endpoint information must be specified. After a system reset or function software reset, the USB function core has no endpoint information. Endpoint information of the USB function core in this LSI can be set by writing to EPDR0I sequentially (for details, see section 18.4.7, USB Module Startup Sequence). By setting this bit to 1 after writing all data in EPDR0I, endpoint information written to the USB function core becomes valid.</p> <p>0: Indicates that endpoint information (EPINFO) is not specified.</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"><li>• System reset</li><li>• Function software reset</li></ul> <p>1: Indicates that endpoint information (EPINFO) is specified.</p>

---

Bit	Bit Name	Initial Value	R/W	Description
1	EP0OTC	0	R/W	<p>Endpoint 0O Transfer Control</p> <p>Controls the USB function core endpoint 0 control transfer. Clearing this bit to 0 disables the write to the EP0 OUT-FIFO. This generates a transfer abnormal completion interrupt. A change in the direction of data transfer in the control transfer can be detected by this interrupt. In control transfer, a command is first received in a SETUP transaction (setup stage), data is then transferred in an OUT transaction or IN transaction (data stage), and a handshake is finally transferred in an IN transaction or OUT transaction (status stage).</p> <p>This bit is set to 1 when a SETUP token is received and command data reception is then enabled after FVSR initialization. This bit is cleared to 0 after a command data has been received to protect the EP0 OUT-FIFO contents. If the received command cannot be processed in the USB function core, the EP0OTS flag in TSFR0 is set to 1 and the slave CPU must analyze the command.</p> <p>After command analysis, if an OUT transaction is performed at the data stage, the slave CPU must set this bit to 1 to prepare the OUT transaction; if an IN transaction is performed at the data stage, the slave CPU leaves this bit cleared to 0. If the host CPU initiates an OUT transaction at the status stage, the EP0OTF flag in TFFR0 is set to 1 to generate a transfer abnormal completion interrupt. Therefore, the slave CPU can acknowledge the completion of the data stage. By the interrupt, the slave CPU sets this bit to 1 to receive status stage data that is retransferred.</p> <p>0: Indicates that EP0 OUT-FIFO is in a write stop state. (The following write to the EP0 OUT-FIFO is disabled.)</p> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• System reset</li> <li>• Function software reset</li> <li>• Command data reception in SETUP transaction (EP0OTS flag is set)</li> </ul> <p>1: Indicates that EP0 OUT-FIFO is in an operating state</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• A SETUP token has been received.</li> <li>• 0 is written to EP0OTC after EP0OTC = 1 has been read.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
0	—	0	R/(W)	Reserved The initial value should not be changed.

Note: \* Writing of 0 is disabled.

### 18.3.12 Endpoint Stall Register 0 (EPSTLR0)

EPSTLR0 stalls the USB function core endpoints. Endpoints whose EPSTL bits are set to 1 respond by a STALL handshake when a transaction has been initiated by receiving a token from the host. A stall state (STALL handshake is used to respond) can be set from both the USB function core and the host. A stall state can be cancelled only from the host. The stall state is specified in the USB function core internal bit. This internal bit can be set or cleared by the SetFeature/Clear Feature command of the host. If STALL handshaking is performed because the EPSTL bit is set to 1, the internal bit of the USB function core is also set to a stall state. Even if the host clears the USB function core internal bit, this internal bit remains to be set to a stall state until the corresponding EPSTL bit is set to 1.

EPSTLR0 is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

#### EPSTLR0

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	EP5STL	0	R/W	Endpoint 5 Stall Sets endpoint 5 in a stall state. 0: Endpoint 5 is in an operating state. (Stall state can be cancelled by the ClearFeature command) [Clearing condition] (SCME = 1) <ul style="list-style-type: none"> <li>STALL handshake response of endpoint 5 is performed.</li> </ul> 1: Endpoint 5 is in a stall state. [Clearing condition] (SCME = 1) <ul style="list-style-type: none"> <li>1 is written to EP5STL after EP5STL = 0 has been read.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
5	EP4STL	0	R/W	<p>Endpoint 5 Stall</p> <p>Sets endpoint 5 in a stall state.</p> <p>0: Endpoint 5 is in an operating state. (Stall state can be cancelled by the ClearFeature command)</p> <p>[Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>STALL handshake response of endpoint 5 is performed.</li> </ul> <p>1: Endpoint 5 is in a stall state. [Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>1 is written to EP5STL after EP5STL = 0 has been read.</li> </ul>
4	EP3STL	0	R/W	<p>Endpoint 4 Stall</p> <p>Sets endpoint 4 in a stall state.</p> <p>0: Endpoint 4 is in an operating state. (Stall state can be cancelled by the ClearFeature command)</p> <p>[Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>STALL handshake response of endpoint 4 is performed.</li> </ul> <p>1: Endpoint 4 is in a stall state. [Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>1 is written to EP4STL after EP4STL = 0 has been read.</li> </ul>
3	EP2STL	0	R/W	<p>Endpoint 3 Stall</p> <p>Sets endpoint 3 in a stall state.</p> <p>0: Endpoint 3 is in an operating state. (Stall state can be cancelled by the ClearFeature command)</p> <p>[Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>STALL handshake response of endpoint 3 is performed.</li> </ul> <p>1: Endpoint 3 is in a stall state. [Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>1 is written to EP3STL after EP3STL = 0 has been read.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
2	EP1STL	0	R/W	<p>Endpoint 2 Stall</p> <p>Sets endpoint 2 in a stall state.</p> <p>0: Endpoint 2 is in an operating state. (Stall state can be cancelled by the ClearFeature command)</p> <p>[Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>STALL handshake response of endpoint 2 is performed.</li> </ul> <p>1: Endpoint 2 is in a stall state.</p> <p>[Clearing condition] (SCME = 1)</p> <ul style="list-style-type: none"> <li>1 is written to EP2STL after EP2STL = 0 has been read.</li> </ul>
1	—	0	R	<p>Reserved</p> <p>This bit is always read as 0 and cannot be modified.</p>
0	EPOSTL	0	R/(W)*	<p>Endpoint 0 Stall</p> <p>Sets endpoint 0 in a stall state.</p> <p>0: Endpoint 0 is in an operating state. (Stall state can be cancelled by the ClearFeature command)</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Endpoint 0 receives a SETUP token.</li> </ul> <p>1: Endpoint 0 is in a stall state.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>1 is written to EPOSTL after EPOSTL = 0 has been read.</li> </ul>

Note: \* Writing of 0 is disabled.

### 18.3.13 Endpoint Reset Register 0 (EPRSTR0)

EPRSTR0 resets the FIFO pointers for each USB function core endpoint.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	EP5RST	0	R/(W)*	Endpoint 5 Reset Initializes endpoint 5 FIFO. 0: Normal read value 1: A command to reset the endpoint 5 FIFO is issued to the RFU
5	EP4RST	0	R/(W)*	Endpoint 4 Reset Initializes endpoint 4 FIFO. 0: Normal read value 1: A command to reset the endpoint 4 FIFO is issued to the RFU
4	EP3RST	0	R/(W)*	Endpoint 3 Reset Initializes endpoint 3 FIFO. 0: Normal read value 1: FVSR3 is initialized to H'0008
3	EP2RST	0	R/(W)*	Endpoint 2 Reset Initializes endpoint 2 FIFO. 0: Normal read value 1: FVSR2 is initialized to H'0000 (when EP2DIR = 0) FVSR2 is initialized to H'0010 (when EP2DIR = 1)

Bit	Bit Name	Initial Value	R/W	Description
2	EP1RST	0	R/(W)*	Endpoint 1 Reset Initializes endpoint 1 FIFO. 0: Normal read value 1: FVSR1 is initialized to H'0010 (EP1 FIFO size is 16 bytes) FVSR1 is initialized to H'0020 (EP1 FIFO size is 32 bytes)
1	EP0IRST	0	R/(W)*	Endpoint 0I Reset Initializes endpoint 0I FIFO. 0: Normal read value 1: FVSR0I is initialized to H'0010
0	EP0ORST	0	R/(W)*	Endpoint 0O Reset Initializes endpoint 0O FIFO. 0: Normal read value 1: FVSR0O is initialized to H'0000

Note: \* Only 1 can be written.



### 18.3.14 Device Resume Register (DEVRSMR)

DEVRSMR has a flag that shows whether remote wakeup is enabled or disabled and a bit that controls remote wakeup of the USB function core suspend state.

DEVRSMR is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
1	RMUPS	0	R	Remote Wakeup Status Indicates whether USB function core remote wakeup is enabled or disabled. Remote wakeup can only be enabled or disabled by a command issued from the host. 0: Remote wakeup disabled 1: Remote wakeup enabled
0	DVR	0	R/(W)*	Device Resume Cancels the suspend state. 0: Normal read value 1: Cancels the suspend state (remote wakeup)

Note: \* Only 1 can be written.

### 18.3.15 Interrupt Source Select Register 0 (INTSELR0)

INTSELR0 selects USBIB interrupt and USBIC interrupt sources for the USB module.

INTSELR0 is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7	TSELB	0	R/W	Transfer Select B  Combined with the EPIBS2 to EPIBS0 bits, selects a USBIB interrupt source.  0: Requests a USBIB interrupt by a TS flag interrupt in USBIFR0 and specifies a TS flag interrupt source endpoint by the EPIBS2 to EPIBS0 bits.  1: Requests a USBIB interrupt by a TF flag interrupt in USBIFR0 and specifies a TF flag interrupt source endpoint by the EPIBS2 to EPIBS0 bits.
6	EPIBS2	0	R/W	Interrupt B Endpoint Select 2 to 0
5	EPIBS1	0	R/W	Combined with the TSELB bit, selects a USBIB interrupt source.
4	EPIBS0	0	R/W	000: Selects no endpoint 001: Selects endpoint 1 010: Selects endpoint 2 011: Selects endpoint 3 100: Selects endpoint 4 101: Selects endpoint 5 11X: Setting prohibited
3	TSELC	0	R/W	Transfer Select C  Combined with EPICS2 to EPICS0 bits, selects a USBIC interrupt source.  0: Requests a USBIC interrupt by a TS flag interrupt in USBIFR0 and specifies a TS flag interrupt source endpoint by the EPICS2 to EPICS0 bits.  1: Requests a USBIC interrupt by a TF flag interrupt in USBIFR0 and specifies a TF flag interrupt source endpoint by the EPICS2 to EPICS0 bits.

---

Bit	Bit Name	Initial Value	R/W	Description
2	EPICS2	0	R/W	Interrupt C Endpoint Select 2 to 0
1	EPICS1	0	R/W	Combined with the TSELC bit, selects a USBIC interrupt source. 000: Selects no endpoint 001: Selects endpoint 1 010: Selects endpoint 2 011: Selects endpoint 3 100: Selects endpoint 4 101: Selects endpoint 5 11X: Setting prohibited
0	EPICS0	0	R/W	

---

Legend:

X: Don't care

### 18.3.16 USB Control Registers 0 and 1 (USBCR0, USBCR1)

USBCR0 selects the USB module data input/output method and controls the operation states and reset states of each unit. USBCR1 controls clock supply to the bus driver/receiver and USB function core.

USBCR0 is initialized to H'7F by a system reset.

USBCR1 is initialized to H'00 by a system reset.

#### USBCR0

Bit	Bit Name	Initial Value	R/W	Description
7	FADSEL	0	R/W	Function Input/Output Analog 1 Digital Selection Selects the USB module data input/output method. 0: Selects the USDP and USDM pins as USB module data input/output. 1: Multiplexes the control input/output of the driver/receiver that is compatible with PDIUSBP11A manufactured by Philips Electronics with port 6 pins and selects it as USB module data input/output (see table 18.3).
6, 5	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.
4	UIFRST	1	R/W	USB Interface Software Reset Resets EP4PKTSZR, EPSZR1, USBIER0, USBIER1, USBMDCR, EPDIR0, and INTSELR0. 0: Sets the above registers in operating state. 1: Sets the above registers in reset state.
3, 2	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
1	FPLL_RST	1	R/W	<p>Function PLL Software Reset</p> <p>Resets the USB bus clock circuit (DPLL) in the USB function core.</p> <p>Setting this bit to 1 resets the DPLL in the USB function core and stops bus clock synchronization. Clear this bit to 0 after the PLL operation is stabilized.</p> <p>0: Sets DPLL in operating state. 1: Sets DPLL in reset state.</p>
0	FSRST	1	R/W	<p>Function Core Internal State Software Reset</p> <p>Resets the internal state of the USB function core.</p> <p>Setting this bit 1 to 1 initializes all the internal states of the USB function core other than the bus clock circuit (DPLL). Clear this bit to 0 after the DPLL operation is stabilized.</p> <p>A function software reset is defined as the state where both the FSRST and UIFRST bits are set to 1.</p> <p>0: Sets USB function core other than DPLL in operating state. 1: Sets USB function core other than DPLL in reset state.</p>

**USBCR1**

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7 to 2	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
1	VBUSS	0	R/W	<p><b>VBUS Status</b></p> <p>Prevents bus driver/receiver feedthrough current from generating by controlling the VBUS line (USB cable) connection state. The VBUS status monitor circuit must be designed by using the external interrupt and general ports. In addition, external pull-up resistors must be turned ON or OFF by using the general ports.</p> <p>0: Prevents feedthrough current from generating by disconnecting VBUS. The USDP and USDM pins are placed in high-impedance state.</p> <p>1: VBUS connection. The USDP and USDM pins are pulled up and pulled down, respectively.</p>
0	CK48- READY	0	R/W	<p><b>CK48READY</b></p> <p>Controls the bus clock (48 MHz) supply to the USB function core. To stop or start up the PLL operation correctly in suspend or resume state, the bus clock supply must be stopped before stopping the PLL circuit, and the bus clock supply must be restarted after the PLL operation is stabilized.</p> <p>0: Disables the bus clock supply to the USB function core</p> <p>1: Enables the bus clock supply to the USB function core</p>

**Table 18.3 Port 6 Functions**

<b>Port 6</b>	<b>Control I/O of Driver/Receiver Compatible with PDIUSBP11A by Philips Electronics</b>		
P67 (DPLS)	Input	VP	Differential input (+)
P66 (DMNS)	Input	VM	Differential input (-)
P65 (XVERDATA)	Input	RCV	Data input
P64 (TXDPLS)	Output	VPO	Differential input (+)
P63 (TXDMNS)	Output	VMO	Differential input (-)
P62 (TXENL)	Output	$\overline{OE}$	Output enable
P61 (SUSPEND)	Output	SUSPEND	Suspend specification
P60 (SPEED)	Output	SPEED	Speed specification (Fixed to high for 12-Mbps specifications)

### 18.3.17 USB PLL Control Register (UPLLCR)

UPLLCR controls the generation method of the USB function core operating clock.

UPLLCR is initialized to H'01 by a system reset.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
5	PFSEL2	0	R/W	PLL Frequency Select 2 Combined with the PFSEL1 and PFSEL0 bits, this bit selects the frequency of the clock to be provided to the USB operating clock generation circuit (PLL). For details, refer to the description of the PFSEL1 and PFSEL0 bits.
4	CKSEL2	0	R/W	Clock Source Select 2 to 0
3	CKSEL1	0	R/W	These bits select the source of the clock to be provided to the USB operating clock generation circuit (PLL).
2	CKSEL0	0	R/W	0XX: PLL stops operation and no clock is input to the PLL. 100: Setting prohibited 101: PLL stops operation and the USEXCL pin input (48 MHz) is directly used instead of PLL output. 110: PLL operates using the system clock generator (XTAL) as a clock source. 111: PLL operates using the USEXCL pin input as a clock source.
1	PFSEL1	0	R/W	PLL Frequency Select 1 and 0
0	PFSEL0	1	R/W	The PFSEL2 to PFSEL0 bits select the frequency of the clock to be provided to the USB operating clock generation circuit (PLL). The PLL circuit generates a 48-MHz USB operating clock based on the clock source whose frequency is specified by the PFSEL1 and PFSEL2 bits. 000: PLL input clock frequency is 8 MHz 001: PLL input clock frequency is 12 MHz 010: PLL input clock frequency is 16 MHz 011: PLL input clock frequency is 20 MHz 100: PLL input clock frequency is 24 MHz Other than above: Setting prohibited

Legend:

X: Don't care



### 18.3.18 Configuration Value Register (CONFV)

CONFV stores the configuration, interface, and alternation values selected by the host by using the SetConfiguration and SetInterface command.

CONFV is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

#### CONFV

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
5	CONFV0	0	R	Configuration Value
4	INTV1	0	R	Interface Value
3	INTV0	0	R	
2	ALTV2	0	R	Alternate Value
1	ALTV1	0	R	
0	ALTV0	0	R	

### 18.3.19 Endpoint 4 Packet Size Register (EP4PKTSZR)

EP4PKTSZR specifies the MaxPacketSize for endpoint 4. In this LSI, only 64 bytes can be specified as MaxPacketSize. Endpoint 4 receives transmit data from the RFU. The packet size of endpoint 4 is controlled by the USB module and transmit data is transferred from the RFU until the number of transmit data bytes reaches the value specified in EP4PKTSZR.

EP4PKTSZR is initialized to H'40 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7	PS7	0	R/W	EP4 Maximum Packet Size
6	PS6	1	R/W	Only H'40 (64 bytes) can be specified; other settings are prohibited.
5	PS5	0	R/W	
4	PS4	0	R/W	
3	PS3	0	R/W	
2	PS2	0	R/W	
1	PS1	0	R/W	
0	PS0	0	R/W	

### 18.3.20 RFU/FIFO Read Request Flag Register (UDTRFR)

UDTRFR provides a flag indicating that endpoint 5 is placed in data transfer completion state. Endpoint 5 has a 2-byte receive buffer in the USB module to temporarily store data received from the host into the receive buffer before transferring it to the RAM-FIFO by the RFU.

The receive buffer holds the receive data until the RFU data transfer ends.

If the RAM-FIFO is full, the transaction may be completed normally even while receive data of one or two bytes still remain in the receive buffer. At this time, the EP5TS bit in TSFR0 is not set to 1 and the EP5UDTR bit is set to 1. By clearing the EP5UDTR bit to 0 when there is a space of at least two bytes in the RAM-FIFO, data in the receive buffer is transferred to the RAM-FIFO, and the EP5TS bit is set to 1 after transfer completes.

In the slave CPU, a space of at least two bytes must be secured in the RAM-FIFO and the EP5UDTR bit must be cleared to 0 by an UDTR interrupt.

UDTRFR is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
0	EP5UDTR	0	R/(W)	Endpoint 5 RFU/FIFO Read Request Flag 0: Indicates that endpoint 5 receive buffer is empty 1: Indicates that endpoint 5 holds [Clearing condition] <ul style="list-style-type: none"> <li>0 is written to EP5UDTR after EP5UDTR = 1 has been read.</li> </ul> [Setting condition] <ul style="list-style-type: none"> <li>USB transfer has been completed normally while the RAM-FIFO is full and the receive buffer holds data.</li> </ul>

**18.3.21 USB Mode Control Register (USBMDCR)**

USBMDCR controls SETUP transaction operations and stall cancellation procedures.

USBMDCR is initialized to H'00 by a system reset or function software reset (see section 18.3.16, USB Control Registers 0 and 1 (USBCR0, USBCR1)).

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
1	SCME	0	R/W	Stall Cancellation Mode Enable Specifies the auto-clear function of the EPSTL bit in EPSTLR0. 0: Does not specify the auto-clear function of the EPSTL bit 1: Specifies the auto-clear function of the EPSTL bit corresponding to the endpoint that responds to a STALL handshaking
0	SETICNT	0	R/W	Setup Interrupt Control Specifies the FIFO handling methods in SETUP transaction and interrupt control. 0: FIFO is used for EP0O and EP0I, and a SETUP transaction uses EP0O. An USBIA interrupt is used and its priority is specified as the highest. 1: FIFO is used for EP0O, EP0I, and EP0S, and a SETUP transaction uses EP0S. An USBIA interrupt is used and its priority is specified as the lowest.

### 18.3.22 USB Port Control Register (UPRTCR), and USB Test Registers 0 and 1 (UTESTR0 and UTESTR1)

UPRTCR, UTESTR0, and UTESTR1 are used for testing. Note that values other than initial values must not be set to UPRTCR, UTESTR0, and UTESTR1.

UPRTCR, UTESTR0, and UTESTR1 are initialized to H'00 by a system reset. UPRTCR, UTESTR0, and UTESTR1 are not initialized in software standby mode.

#### UPRTCR

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
2	PCNMD2	0	R/(W)	Test Mode Setting
1	PCNMD1	0	R/(W)	Initial values must not be modified.
0	PCNMD0	0	R/(W)	

#### UTESTR0

Bit	Bit Name	Initial Value	R/W	Description
15 to 8	TEST15 to TEST8	All 0	R/(W)	Test Mode Setting Initial values must not be modified.

#### UTESTR1

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	TEST7 to TEST0	All 0	R/(W)	Test Mode Setting Initial values must not be modified.

## 18.4 Operation

The USB is an interface for personal computer peripheral devices and is defined by the USB standard Rev. 1.1. The USB module in this LSI operates based on the USB standard Rev. 1.1.

### 18.4.1 USB Function Core Functions

The USB function core has five endpoints.

The USB function core can select three alternate specifications from a combination of endpoint 2 enable/disable and IN/OUT, and the maximum packet size (MaxPacketSize) of endpoint 1.

Configu- ration	Interface	Alternate Specification	Endpoint 0		Endpoint 1		Endpoint 2	
			IN/OUT	FIFO	IN/OUT	FIFO	IN/OUT	FIFO
1	0	0	IN/OUT	16 bytes for each	IN	16 bytes	IN	16 bytes
		1					OUT	16 bytes
		2					None	None

Configu- ration	Interface	Alternate Specification	Endpoint 3		Endpoint 4		Endpoint 5	
			IN/OUT	FIFO	IN/OUT	FIFO	IN/OUT	FIFO
1	1	0	IN	8 bytes	IN	2048 bytes (Maximum)	OUT	2048 bytes (Maximum)

The USB function core supports a control transfer by endpoint 0, interrupt transfer by endpoints 1 to 3, and bulk transfer by endpoints 4 and 5.

A control transfer is comprised of multiple transactions. In a SETUP transaction, a command sent from the host is first decoded in the USB function core.

When a SETUP token has been received, FVS00 and FVSR0I are initialized, the EP0OTC bit is set to 1 and command reception is enabled. If a USB standard command other than GetDescriptor or SetDescriptor is received, the SETUPF and EP0OTF flags are set to 1 and the USB standard command reception is informed to the slave CPU (when SETICNT = 0). In this case, the remaining transactions of the control transfer are processed in the USB function core and the slave CPU performs no operations.

On the other hand, if a GetDescriptor or SetDescriptor command or device class specific command is received, the SETUPF and EP0OTS flags are set to 1 (when SETICNT=0). The slave

CPU must read, interpret and execute a command from the FIFO. The slave CPU must process the remaining transactions of a control transfer using FIFOs.

When the SETICNT bit is set to 1, the FIFO for EP0S is used and the SETUPF flag is set only when a GetDescriptor, SetDescriptor, or device class specific command is received.

An interrupt transfer is comprised of a single IN or OUT transaction. The slave CPU must process an interrupt transfer using FIFOs.

A bulk transfer is comprised of a single IN or OUT transaction. Transfer data is directly transferred between on-chip RAM and USB modules by the RFU.

As described above, the USB module performs communication processing using both the USB function core and slave CPU as required. Table 18.4 shows the USB function core and slave CPU functions and the registers, flags, and bits used for interface.

**Table 18.4 USB Function Core and Slave CPU Functions**

No.	Function	Operating Hardware	Related Registers, Flags, and Bits
1	Analog ↔ digital conversion of the USDP/USDM signal	Port unit USB function core	—
2	Serial ↔ parallel conversion/ bit staffing, PID check/addition, CRC check/addition	USB function core	—
3	Token packet check/SETUP notification to the slave CPU	USB function core	SETUPF, TS, EPTS, TF, EPTF
4	Handshake packet check/generation DATA0/1 PID toggle, FIFO rewind, ACK/NAC detection/return	USB function core	FVSR, EPTE, RFU
	ACK handshake detection, notification to the slave CPU/ACK handshake return		TS, EPTS
	Data error detection, notification to the slave CPU/NAK handshake return		TF, EPTF
	STALL handshake return		EPSTL
5	Data packet reception/generation/transfer between the slave CPU	USB function core	FIFO, RFU
6	USB command interpretation and execution	USB function core, slave CPU	FIFO, RFU

The bus driver/receiver on the port unit and the USB function core process the electrical signal and signal bit stream on the USB bus line. The token, acknowledge type, and data byte are extracted, and the acknowledge and data byte are converted to electrical signals of the bit stream (no. 1 and 2 in table 18.4).

In a SETUP token reception, if a GetDescriptor or SetDescriptor command or a device class specific command is received, the SETUP and EP0OTS flags are set to 1 and SETUP token reception is informed to the slave CPU (no. 3 in table 18.4). The received command must be transferred using the FIFO and be interpreted and processed in the slave CPU (no. 6 in table 18.4). The remaining transactions in the control transfer must also be processed in the slave CPU using FIFOs (no. 4 and 5 in table 18.4).

In a control transfer, interrupt transfer, and bulk transfer, an IN or OUT token reception is not informed to the CPU and data transfer is performed continuously. In an IN transaction, transmit data is prepared in the FIFO in advance and transmission is initiated if the EPTE bit is set; otherwise a NAK handshaking is performed. After an IN transaction has been completed, transfer normal completion or abnormal completion is determined by the host handshaking and this result is informed to the CPU through the TS and TF bits in USBIFR0, the EPTS bit in TSFR0, and the EPTF bit in TFFR0. In an OUT transaction, an ACK handshaking is performed if the FIFO has received all data items; otherwise an NAK handshaking is performed. In an IN or OUT transaction, a STALL handshaking is performed if the endpoint is specified as a STALL state by EPSTL.

#### **18.4.2 Operation on Receiving a SETUP Token (Endpoint 0)**

Transactions sequentially initiated when the host has received a SETUP token are called control transfer. The control transfer is comprised of three stages: setup, data, and status stages. The control transfer includes two transfer types: control write transfer and control read transfer. The transfer type such as control write or read transfer and the number of bytes to be transferred in the data stage are determined by the 8-byte command transferred by an OUT transaction in setup stage.

The setup stage comprises a SETUP transaction. The data stage comprises no transaction or one or multiple data transactions. The status stage comprises a data transaction.

Table 18.5 shows the packets included in each transaction.



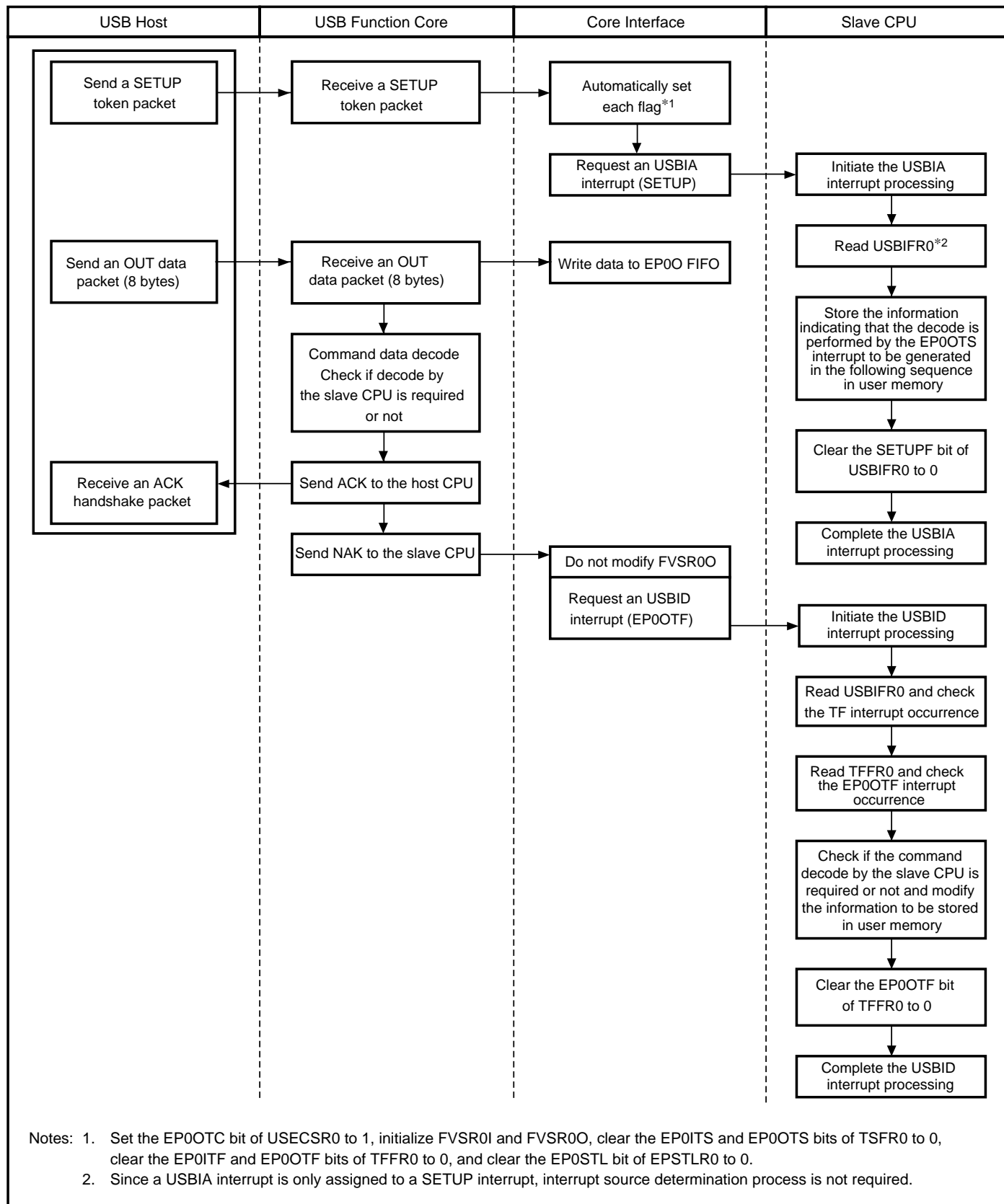
**Table 18.5 Packets Included in Each Transaction**

Stage		Token Phase	Data Phase	Handshake Phase <sup>*1</sup>
Setup stage		SETUP token packet	OUT data packet (8 bytes) (host to slave)	ACK handshake packet (slave to host)
Control write transfer	Data stage	OUT token packet	OUT data packet (host to slave)	ACK/NAK/STALL handshake packet (slave to host)
	Status stage	IN token packet	IN data packet (0 bytes) <sup>*2</sup> (slave to host)	ACK handshake packet (host to slave)
			NAK/STALL handshake packet (slave to host)	—
Control read transfer	Data stage	IN token packet	IN data packet (slave to host)	ACK handshake packet (host to slave)
			NAK/STALL handshake packet (slave to host)	—
	Status stage	OUT token packet	OUT data packet (host to slave)	ACK/NAK/STALL handshake packet (slave to host)
No data stage	Status stage	IN token packet	IN data packet (0 bytes) <sup>*2</sup> (slave to host)	ACK handshake packet (host to slave)
			NAK/STALL handshake packet (slave to host)	—

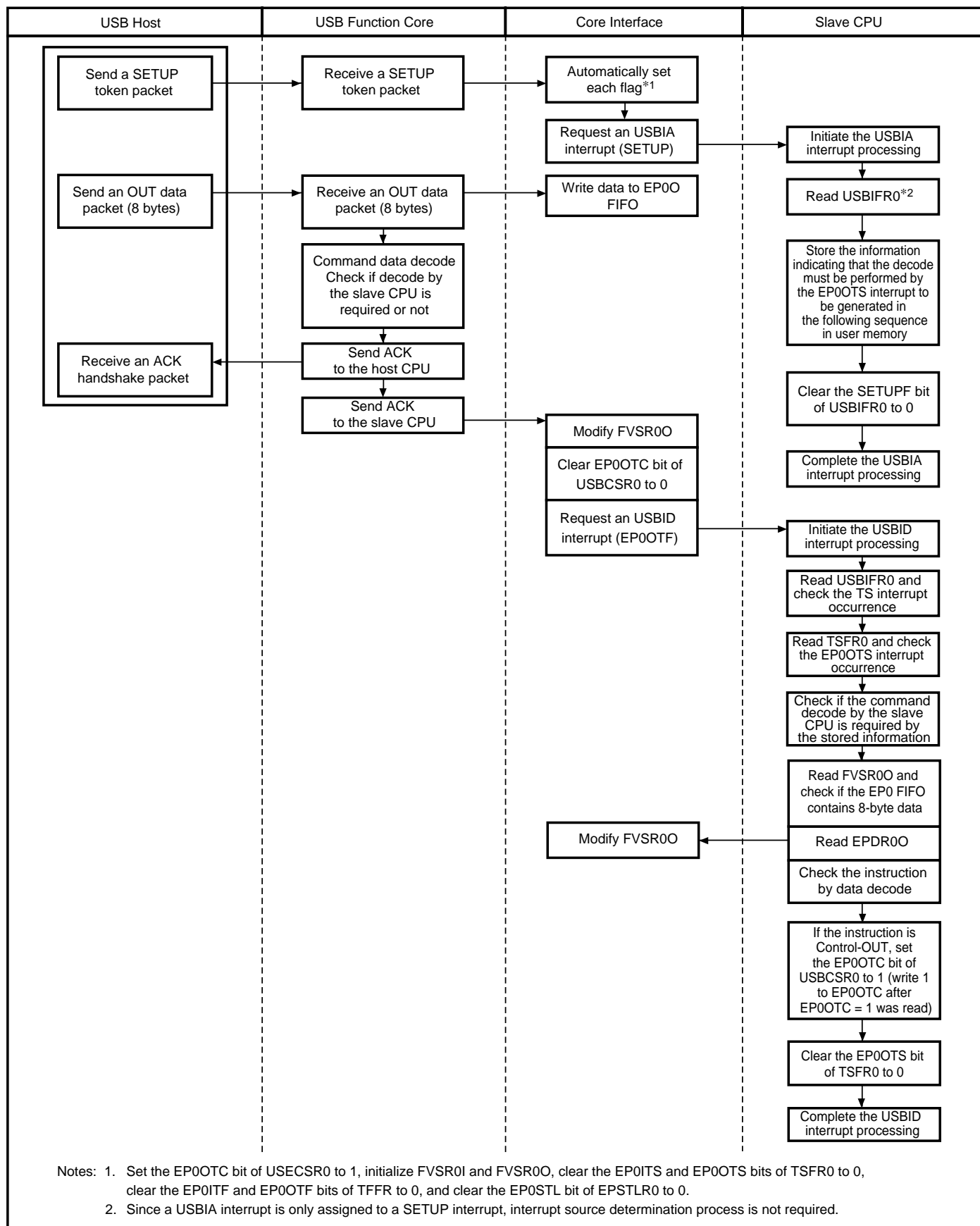
Notes: 1. This phase exists only when a data packet has been transferred in the data phase.

2. If the FIFO is empty after all data items in the FIFO have been transferred, the EPTE bit is cleared to 0. If an IN transaction is initiated at this time, a NAK handshake is returned. To transfer a 0-byte data packet, set the EPTE bit to 1 while the FIFO is empty.

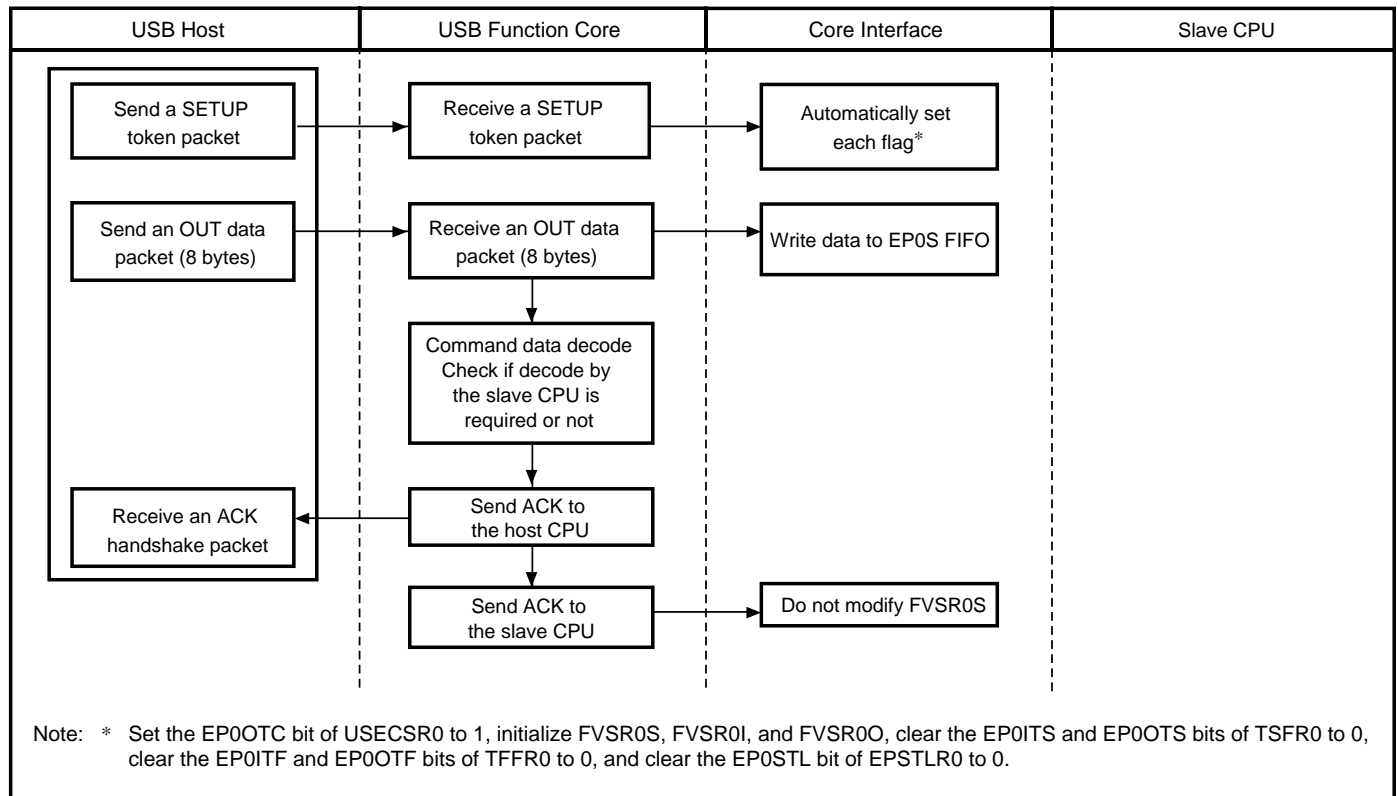
Figures 18.2 to 18.5 show the USB function core and LSI firmware operations when the USB function core receives a SETUP token (SETUP transaction).



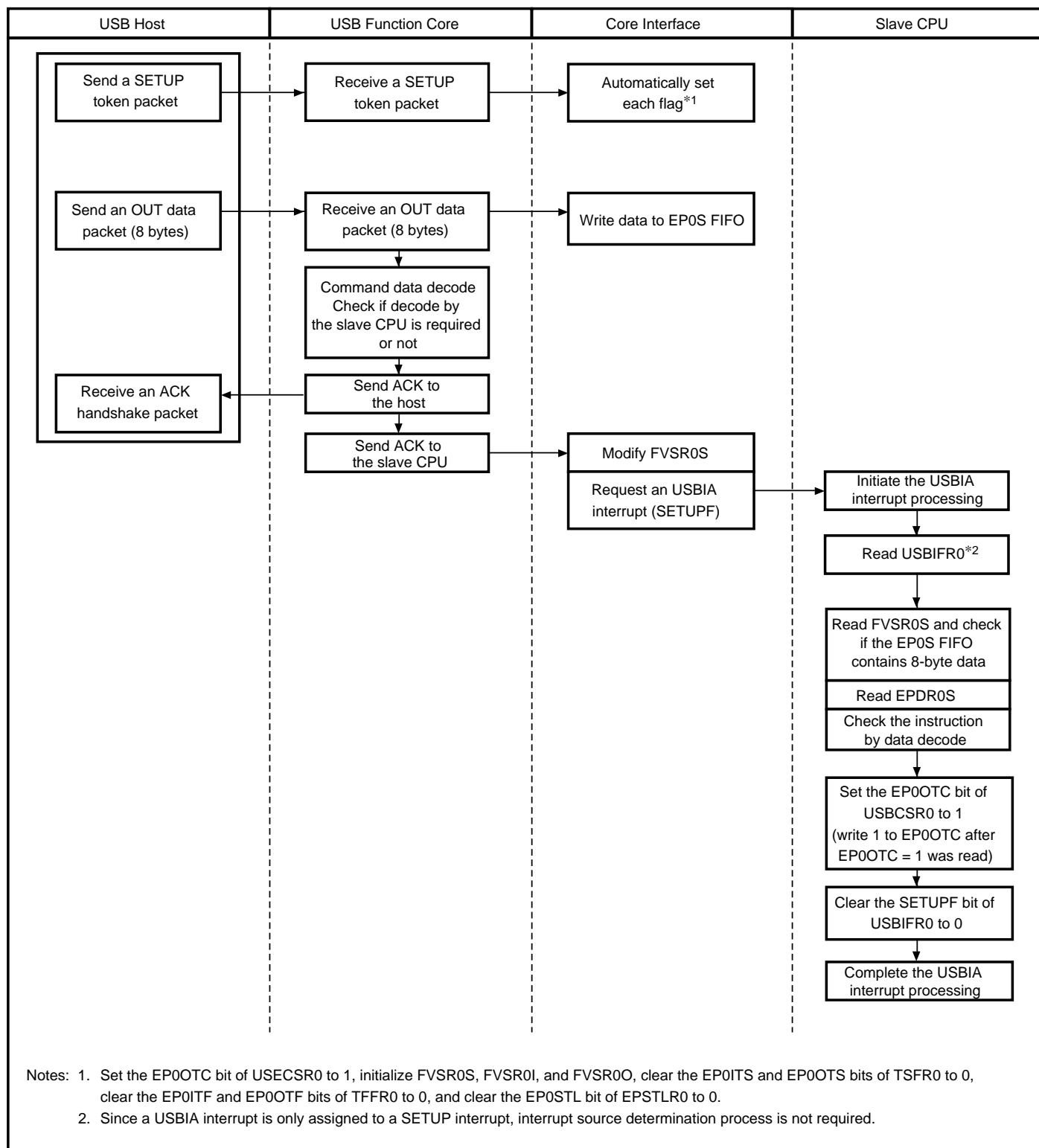
**Figure 18.2 Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is not Required and When SETICNT = 0)**



**Figure 18.3 Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is Required and When SETICNT = 0)**



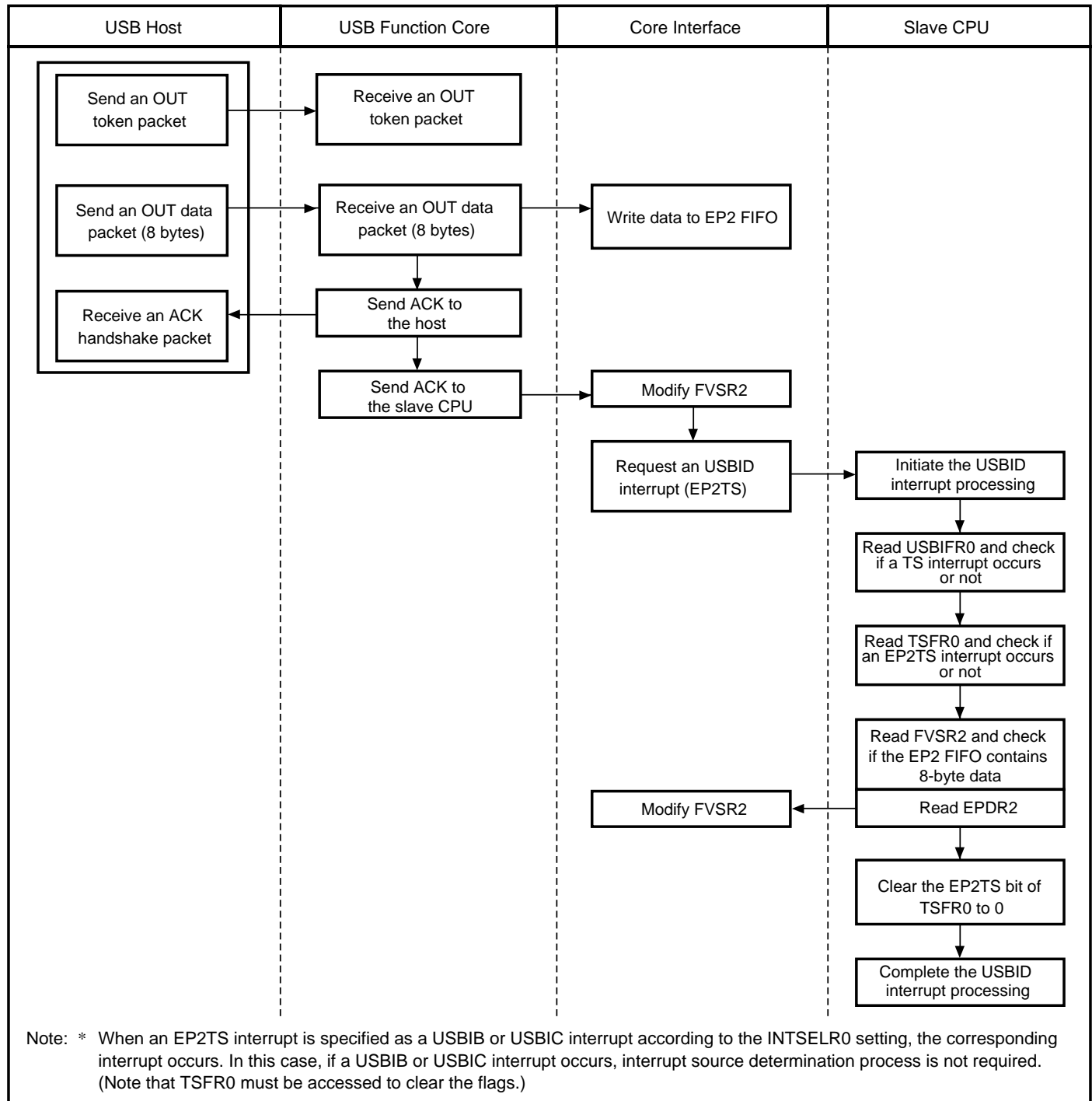
**Figure 18.4 Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is Not Required and When SETICNT = 1)**



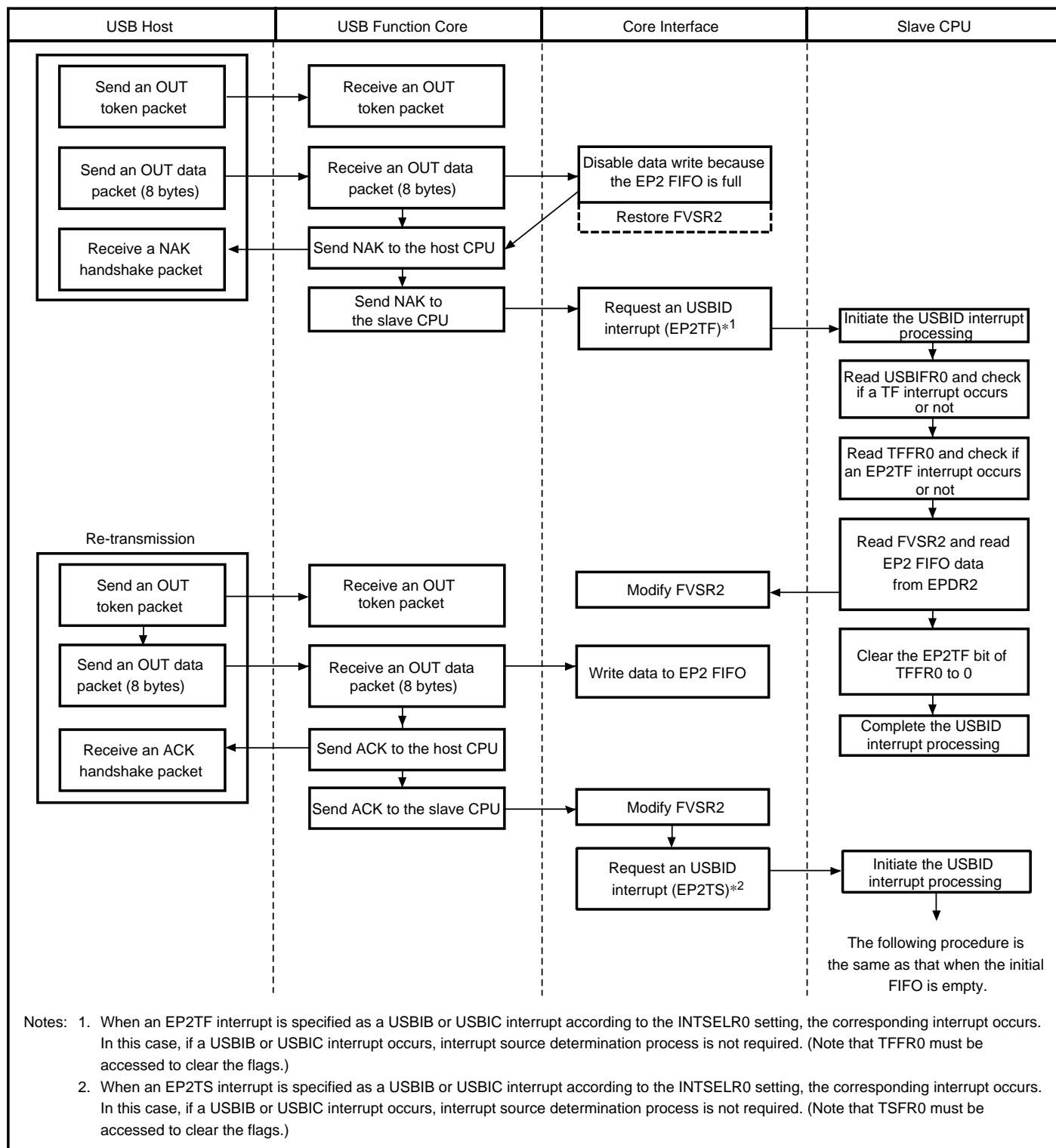
**Figure 18.5 Operation on Receiving a SETUP Token (When Decode by the Slave CPU Is Required and When SETICNT = 1)**

### 18.4.3 Operation on Receiving an OUT Token (Endpoints 0, 2, and 5)

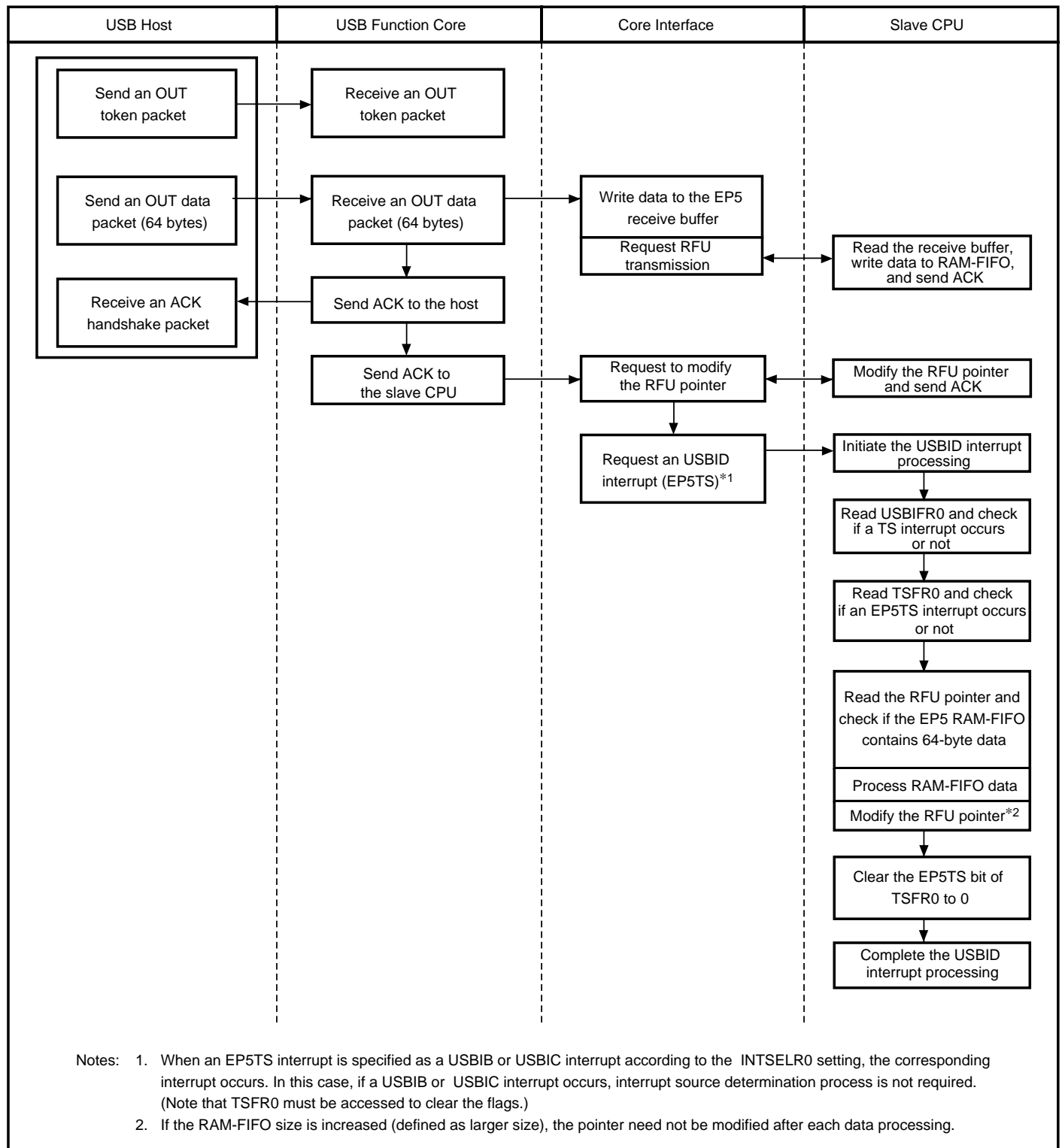
Figures 18.6 to 18.9 show the USB function core and LSI firmware operations when the USB function core receives an OUT token (OUT transaction). An OUT transaction is used for data stage and status stage of control transfer, interrupt transfer, and bulk transfer.



**Figure 18.6 Operation on Receiving an OUT Token (EP2-OUT: Initial FIFO Is Empty)**

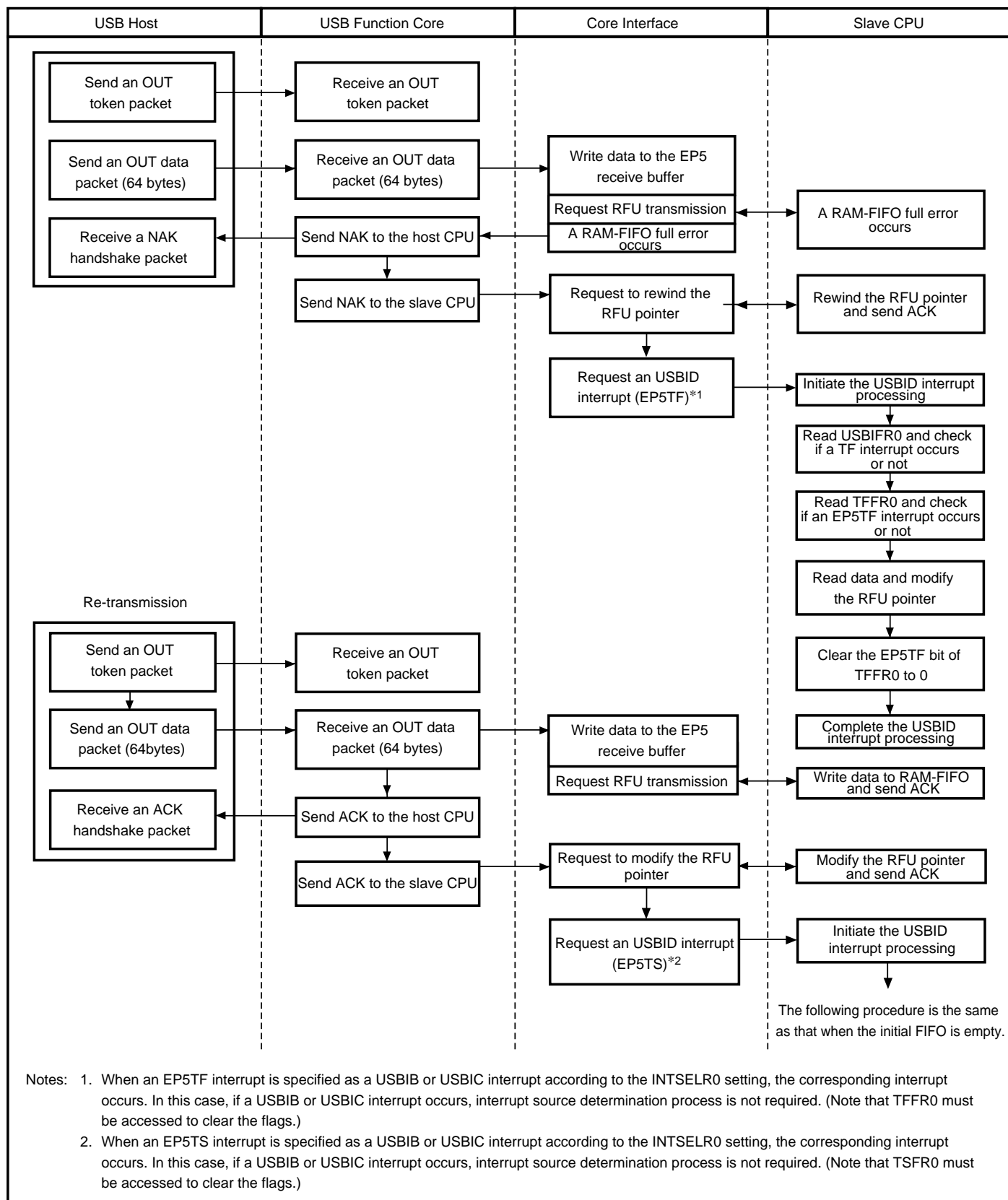


**Figure 18.7 Operation on Receiving an OUT Token (EP2-OUT: Initial FIFO Is Full)**



**Figure 18.8 Operation on Receiving an OUT Token (EP5-OUT: Initial FIFO Is Empty)**

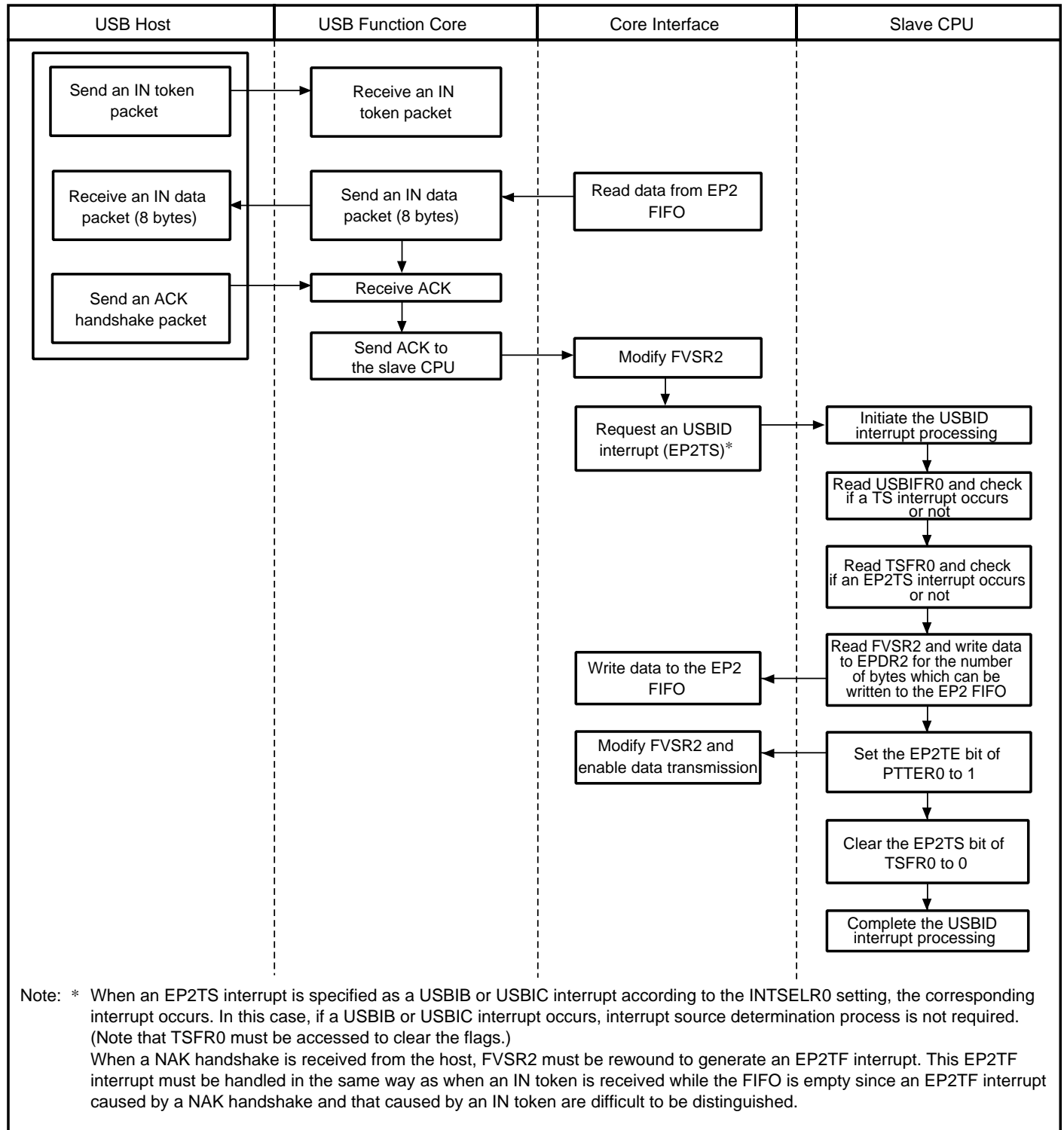




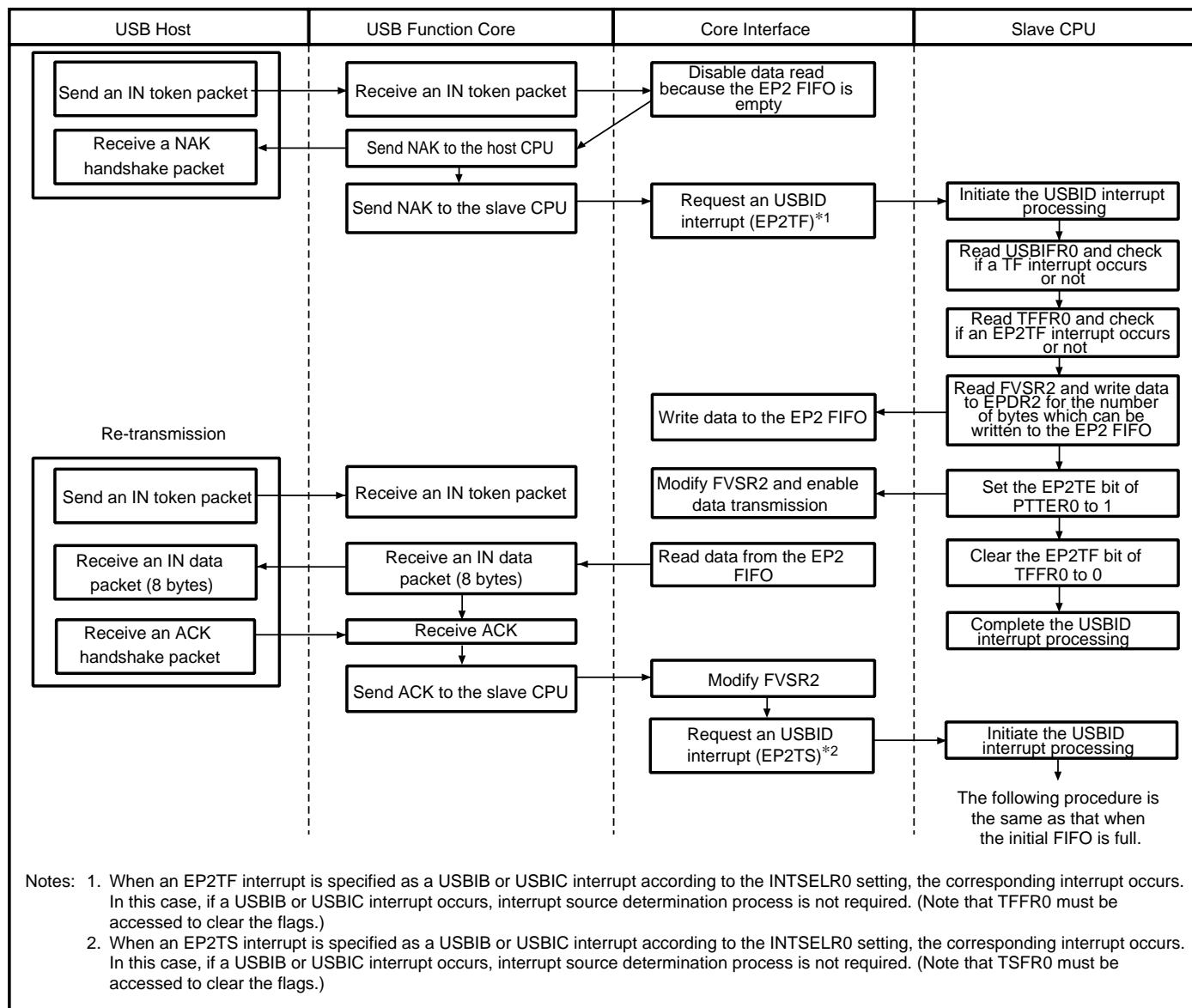
**Figure 18.9 Operation on Receiving an OUT Token (EP5-OUT: Initial FIFO Is Full)**

### 18.4.4 Operation on Receiving an IN Token (Endpoints 0, 1, 2, 3 and 4)

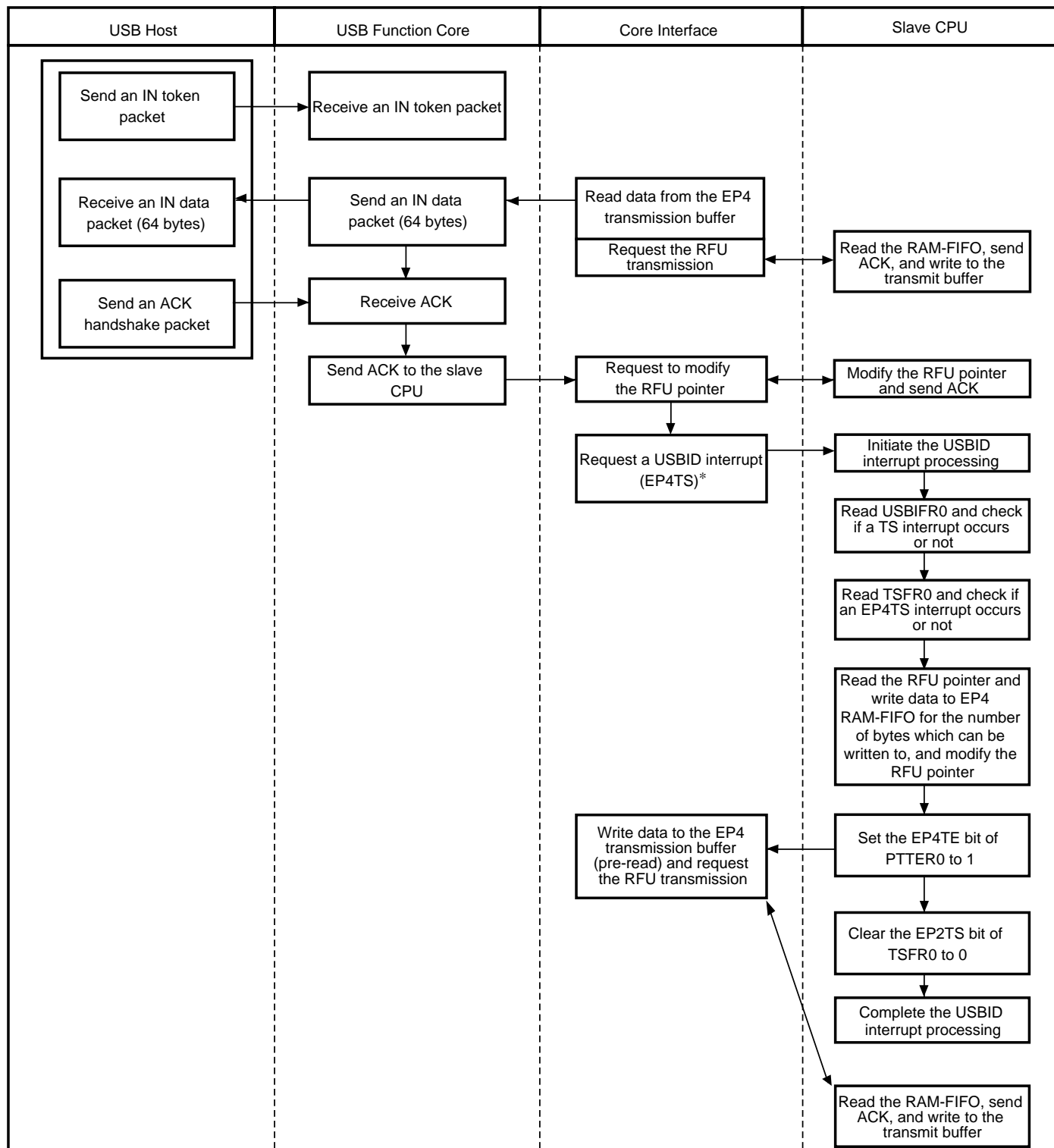
Figures 18.10 to 18.13 show the USB function core and LSI firmware operations when the USB function core receives an IN token (IN transaction). An IN transaction is used for data stage and status stage of control transfer, interrupt transfer, and bulk transfer.



**Figure 18.10 Operation on Receiving an IN Token (EP2-IN: Initial FIFO Is Full)**



**Figure 18.11 Operation on Receiving an IN Token (EP2-IN: Initial FIFO Is Empty)**

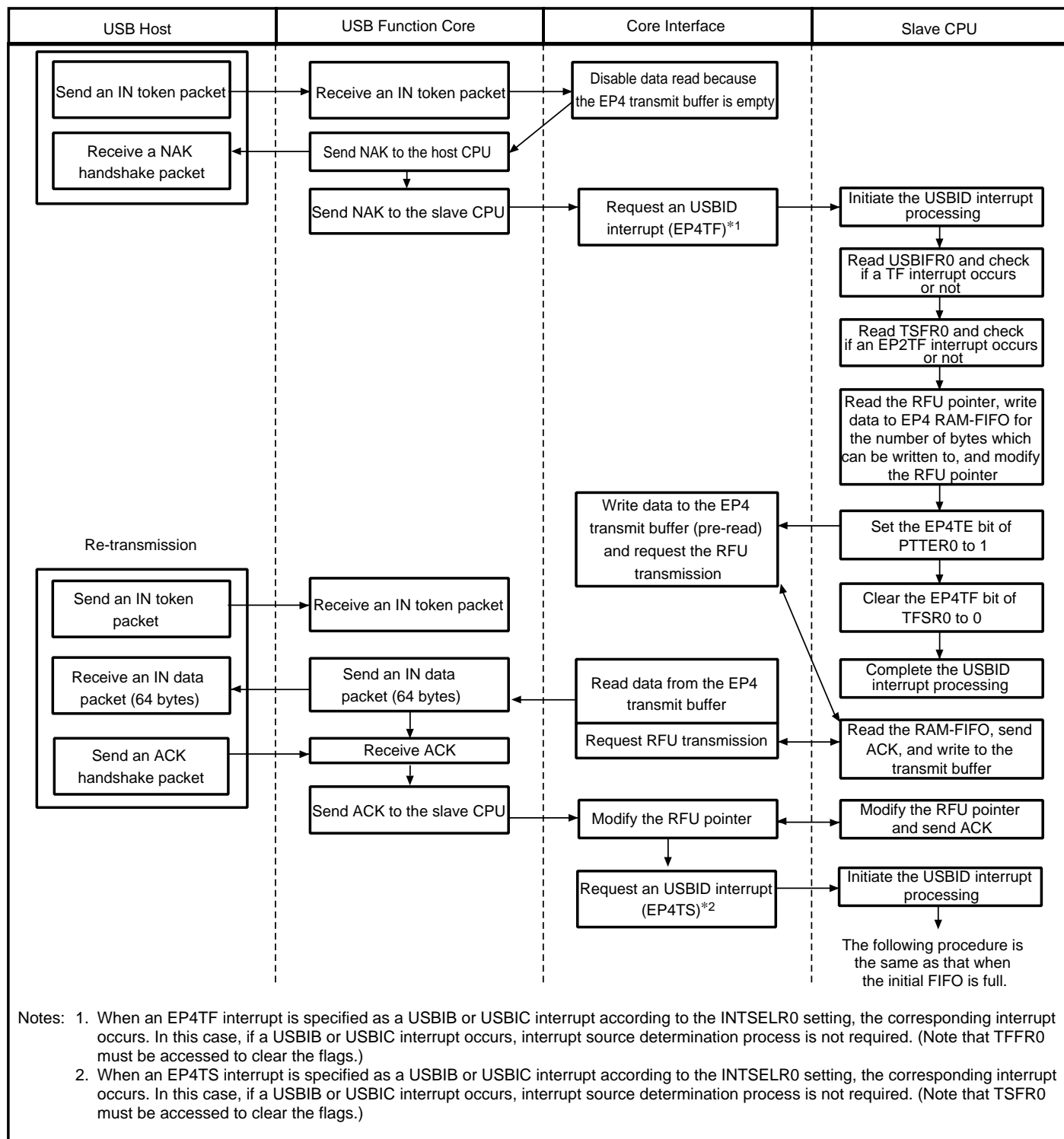


Note: \* When an EP4TS interrupt is specified as a USBIB or USBIC interrupt according to the INTSELR0 setting, the corresponding interrupt occurs. In this case, if a USBIB or USBIC interrupt occurs, interrupt source determination process is not required. (Note that TSFR0 must be accessed to clear the flags.)

When a NAK handshake is received from the host, the RFU pointer must be rewound to generate an EP4TF interrupt. This EP4TF interrupt must be handled in the same way as when an IN token is received while the FIFO is empty since an EP4TF interrupt caused by a NAK handshake and that caused by an IN token are difficult to be distinguished.

Setting the EP4TE bit to 1 to generate data write (pre-read) to the transmit buffer is necessary.

**Figure 18.12 Operation on Receiving an IN Token (EP4-IN: Initial FIFO Is Full)**



**Figure 18.13 Operation on Receiving an IN Token (EP4-IN: Initial FIFO Is Empty)**

### 18.4.5 Suspend/Resume Operation

The USB function core automatically enters a suspend state if the USB data line is placed in an idle state for a time equal to or greater than that specified by the USB standard Rev. 1.1.

A suspend state is automatically cancelled (resumed) when the upstream (host) resumes data transfer. A suspend state can also be forcibly cancelled (resumed) by the USB function (remote wakeup).

Suspend state and resume state transitions can be detected by the SPNDIF and SPNDOF flags. A SPNDOF interrupt can be accepted by waking up this LSI even if the LSI is placed in software standby mode. In this case, the oscillator and PLL circuit must be started up according to the startup sequence.

Whether remote wakeup is enabled or not is checked by the RWUPS flag of DEVRSMR. If remote wakeup is enabled, remote wakeup is performed by setting the DVR bit of DEVRSMR to 1.

### 18.4.6 USB Module Reset and Operation Stop Modes

The USB module can be placed in a reset state or operation stop mode by using multiple control bits. To startup the USB module by sequentially specifying these control bits, refer to section 18.4.7, USB Module Startup Sequence.

The USB supports the following reset and operating stop states. Hardware standby and reset states initialize the entire USB module. In each register description, initialization conditions are not described and only initial values are shown.

1. Hardware standby mode
2. Reset state
3. Module stop mode
4. Software standby mode
5. USB bus reset state
6. USB suspend state

**Hardware Standby Mode:** Hardware standby mode is entered by bringing the  $\overline{\text{STBY}}$  pin of the LSI to low. In hardware standby mode, registers that can be initialized and the internal status of the LSI are initialized and all pins of the LSI are placed in a high impedance state.

XTAL-EXTAL system clock oscillation stops in the clock pulse generator.

**Reset State:** A reset state is entered by bringing the  $\overline{\text{RES}}$  pin of the LSI to low. In a reset state, registers that can be initialized and the internal status of the LSI are initialized and all pins of the LSI are placed in input state.

XTAL-EXTAL system clock oscillation can be continued in the clock pulse generator.

**Module Stop Mode:** The USB module enters module stop mode when the SMSTPB1 bit in subchip module stop control register BL (SUBMSTPBL) is set to 1. In a module stop mode, the system clock supply to the USB module stops. Note that supply of the USB operating clock (48 MHz) does not stop so, the USB module continues operation. To place the USB module in a module stop mode, initialize the USBCR1 and UPLLCR. In addition, it is recommended that USBCR0 should be initialized to prepare a module stop mode cancellation. Since the module stop bit of SUBMSTPCR is initialized to 1 in hardware standby mode or by a reset, the USB module is placed in a module stop mode after reset cancellation.

**Software Standby Mode:** Software standby mode is entered if the SLEEP instruction is executed while the SSBY bit of SBYCR is set to 1. In software standby mode, the USB module is not placed in a reset or operation stop state. Note that if the slave CPU stops operation in software standby mode, the USB function core operation may not be performed completely. To specify software standby mode regardless of the host status, set the UIFRST, FPLLST, and FSRST bits of USBCR0 to 1 to stop the USB module operation before software standby mode is entered. Due to this, the registers shown in table 18.6 are initialized. In this case, external pull-up MOSs must be specified to be in a cable disconnected state.

These descriptions also apply to watch mode, subactive mode, and subsleep mode.

**Table 18.6 Registers Initialized by Bit UIFRST or FSRST**

Register Name	UIFRST/FSRST
EPDR3, EPDR2, EPDR1, EPDR0S, EPDR0O, EPDR0I	FSRST
FVSR3, FVSR2, FVSR1, FVSR0S, FVSR0O, FVSR0I	FSRST
EPSZR1	UIFRST
USBIER0, USBIER1	UIFRST
USBIFR0, USBIFR1, TSFR0, TFFR0, UDTRFR, CONFV	FSRST
USBCSR0, DEVRSMR, EPSTLR0	FSRST
EPDIR0	UIFRST
INTSELR0, EP4PKTSZR, USBMDCR	UIFRST

When the host enters a suspend state, software standby mode in which the host waits for cancellation (resume) can be specified.

**USB Bus Reset State:** When a new device is connected to the USB bus or when an error recovery process is executed, the USDP/USDM pin is placed in a bus reset state for a specific period.

In the USB module, the bus reset interrupt flag is set to 1 when a USB bus reset is detected. The USB module internal status and control registers that select the USB module operating state are not initialized by a USB bus reset. Registers must be initialized individually as required. When the USB function core is initialized by the FSRST bit in USBCR0, endpoint information (EPINFO) and bus reset detection information of the USB function core are also initialized. Therefore, initialization by the FSRST or UIFRST bit in USBCR0 should not be performed in the bus reset interrupt processing.

**USB Suspend State:** The USB function core enters a suspend state when the USB bus is placed in an idle state for a specific period. In a suspend state, power consumption can be reduced by stopping the PLL circuit after clearing the CK48READY bit in USBCR1 to 0. A suspend state combined with a low-power consumption mode can effectively reduce the power consumption while the system is not in operation.

When the USB function core enters a suspend state or when the suspend state is cancelled by the signal change on the USDP/USDM pin, the suspend IN and OUT interrupt flags are set to 1, respectively. An SPNDOF interrupt can be accepted by waking up this LSI even if this LSI is placed in software standby mode.



### 18.4.7 USB Module Startup Sequence

**USB Module Configuration:** The USB module is comprised of several components. To correctly operate these components and notify the host, the USB module must be started up by the firmware (this LSI's program) according to the sequence described later.

The USB module is comprised of:

- USB clock external input pin, USB operating clock PLL (48 MHz)
- USB bus clock synchronization DPLL (12 MHz)
- EPINFO: Endpoint information
- Slave CPU, core interface
- USB function core

(a) USB clock external input pin, USB operating clock PLL circuit

The PLL circuit that generates the USB operating clock multiplies the clock input from the USB clock external input pin or the oscillator to generate a 48-MHz clock. Therefore, the clock input to the PLL circuit must be 8, 12, 16, 20, or 24 MHz. The multiplication rate of the PLL circuit can be specified by the PFSEL bit in UPLLCR. The 48-MHz clock generated by the PLL circuit can be divided to generate a 24-MHz clock. This 24-MHz clock can be used as the system clock.

For USB operation clock PLL stabilization time, set the UIFRST, PFLLRST, and FSRST bits of the USBCR0 to 1 to place the USB bus clock synchronization DPLL and USB function core in a reset state.

(b) USB bus clock synchronization DPLL (12 MHz)

The USB data is transferred at 12 Mbps (maximum). The bit data is sampled at the timing using the 48-MHz USB operating clock and the phase is adjusted while the synchronization pattern is received prior to the packet. This mechanism is called the USB bus clock synchronization DPLL.

The USB bus clock synchronization DPLL stabilization time is defined by firmware. For the USB bus clock synchronization DPLL stabilization time, set the FSRST bit of USBCR0 to 1 to place the USB function core in a reset state.

It is recommended that the USB bus clock synchronization DPLL stabilization time should be 10 cycles or more in a 48-MHz clock.

## (c) EPINFO: Endpoint information

The USB function core of this LSI can support isochronous transfer. However, due to the CPU interface specifications, the USB function core of this LSI handles only control transfer, interrupt transfer, and bulk transfer.

At each USB function core initialization, the firmware writes endpoint information (EPINFO) such as the number of endpoints and their corresponding transfer type, maximum packet size (bytes) to the USB function core. This LSI prepares three alternatives that can be written in EPINFO. Table 18.7 shows the endpoint information (EPINFO) that can be written in the USB function core. A total of 65 bytes (A1, A2, ..., A5, B1, B2, ..., M4, M5) must be written to EDPR0I in this order.

**Table 18.7 Endpoint Information**

	1	2	3	4	5
A	H'00	H'00	H'11	H'00	H'00
B	H'14	H'38	H'10	H'00	H'01
C	H'24	H'38	H'10	H'00	H'02
D	H'14	H'78	H'10	H'00	H'01
E	H'24	H'70	H'10	H'00	H'02
F	H'14	H'B8	H'20	H'00	H'01
G	H'35	H'20	H'10	H'00	H'03
H	H'45	H'20	H'10	H'00	H'04
I	H'55	H'20	H'10	H'00	H'05
J	H'65	H'20	H'10	H'00	H'06
K	H'36	H'20	H'10	H'00	H'03
L	H'46	H'20	H'10	H'00	H'04
M	H'56	H'20	H'10	H'00	H'05

## (d) Slave CPU, core interface

The slave CPU and core interface are the basic components for firmware execution. The slave CPU starts operating immediately after a reset is cancelled. The core interface, however, can be accessed after module stop mode is cancelled.

## (e) USB function core

The USB function core is the main component of the USB interface. The USB bus interface can be achieved if all components described in (a) to (d) operate correctly.

**Initialization Sequence:** The USB module is initialized in the sequence shown in figure 18.14.

1. The LSI is placed in a power off state or hardware standby mode.
2. Turn the power on, apply a high level to the  $\overline{STBY}$  pin, and finally apply a high level to the  $\overline{RES}$  pin to initiate the LSI operation.
3. Check the VBUS line (USB cable) by firmware.
4. Cancel module stop mode of the USB module by firmware.
5. Set the VBUSS bit in USBCR1 to 1 by firmware.
6. Specify UPLLCR by firmware and wait for USB operating clock PLL stabilization time (3 ms).
7. Set the CK48READY bit in USBCR1 to 1 by firmware.
8. Clear the UIFRST bit in USBCR0 to 0 by firmware and specify the USB module related registers.
9. Clear the FPLL RST bit in USBCR0 to 0 by firmware.
10. After the DPLL operation stabilization time has passed, clear the FSRST bit in USBCSR0 to 0 by firmware.
11. Write EPINFO to the USB function core and set the EPIVLD bit in USBCR0 to 1 by firmware.
12. Specify external pull-up resistors to be connected by firmware.
13. Wait for a bus reset interrupt .
  - The host executes the bus reset.
  - The host configures the USB function core. → The USB function core initiates operation.

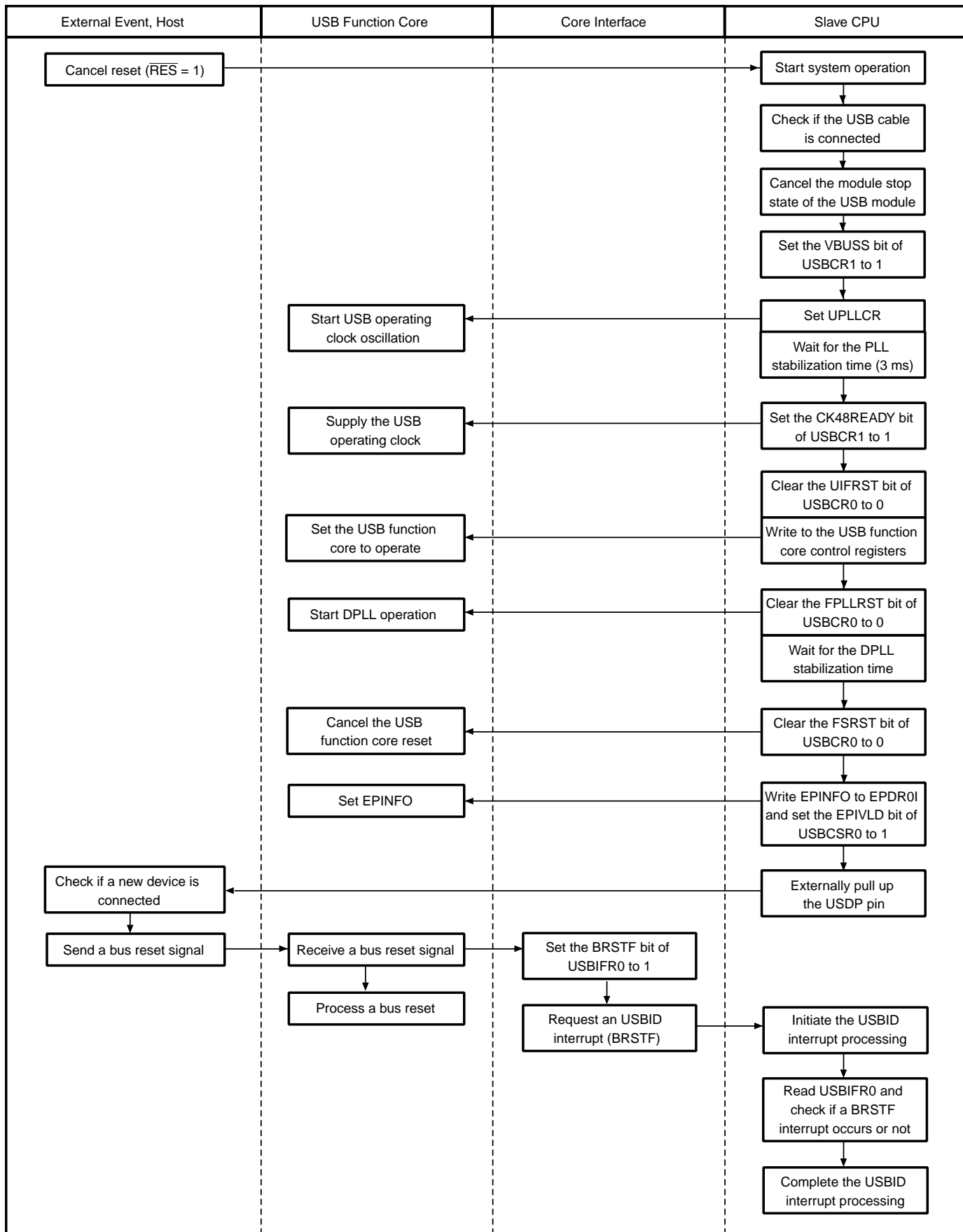


Figure 18.14 Operation Procedure for Initializing USB Module

## 18.5 Interrupt Sources

The USB module's slave CPU interrupts are USBIA, USBIB, UABIC, and USBID interrupts. Tables 18.8 and 18.9 show interrupt sources and their priorities. Interrupt sources are interrupt flags of USBIRF0, USBIFR1, TSFR0, TFFR0, and UDTRFR. Each interrupt can be enabled or disabled according to each interrupt enable bit of USBIER0 and USBIER1. Before initiating the USBID interrupt process routine, the interrupt source must be determined by reading USBIRF0, USBIFR1, TSFR0, TFFR0, and UDTRFR.

The USBIA, USBIB, UABIC, and USBID interrupts can be used as DTC activation interrupt sources.

**Table 18.8 USB Interrupt Sources (When SETICNT of USBMDCR Is 0)**

Interrupt Source	Description	DTC Activation	Priority
USBI0	USBIA An interrupt by SETUP	Possible	High
USBI1	USBIB An interrupt by EPTS or EPTF of the endpoint specified by INTSELR0	Possible	↑
USBI2	USBIC An interrupt by EPTS or EPTF of the endpoint specified by INTSELR0	Possible	
USBI3	USBID An interrupt by SOF, SPND, BRST, TS, TF, UDTR, SETI, or SETC	Possible	Low

**Table 18.9 USB Interrupt Sources (When SETICNT of USBMDCR Is 1)**

Interrupt Source	Description	DTC Activation	Priority
USBI0	USBIB An interrupt by EPTS or EPTF of the endpoint specified by INTSELR0	Possible	High
USBI1	USBIC An interrupt by EPTS or EPTF of the endpoint specified by INTSELR0	Possible	↑
USBI2	USBID An interrupt by SOF, SPND, BRST, TS, TF, UDTR, SETI, or SETC	Possible	
USBI3	USBIA An interrupt by SETUP	Possible	Low

## 18.6 Usage Notes

1. When activating the DTC by a USB interrupt source, correct operation is not guaranteed if the DISEL bit in DTC mode register B (MRB) is cleared to 0. Be sure to set the DISEL bit to 1 before DTC activation.
2. USB module operation can be enabled or disabled by the SMSTPB1 bit in subchip module stop control register BL (SUBMSTPBL). In the initial state, USB module operation is enabled. The USB module registers can be accessed by clearing module stop mode.
3. It is not recommended to use the USB simultaneously with the A/D converter or D/A converter because the bus driver/receiver power supply pin DrVCC/DrVSS is shared with the analog power supply pin AVCC/AVSS.

## Section 19 Multimedia Card Interface (MCIF)

This LSI supports a multimedia card (MultiMediaCard™\*<sup>1</sup>, hereafter referred to as MMC) interface (MCIF)\*<sup>2</sup>. The MCIF has two operating modes: MultiMediaCard mode (hereafter referred to as MMC mode) and SPI mode. The MCIF is a clocked-synchronous serial interface that transmits/receives data that is distinguished in terms of command and response.

A number of command/responses are predefined in the MMC. As the MCIF specifies a command code and command type/response type upon the issuance of a command, additional commands can be supported in future within the range of combinations of currently defined command types/response types. The command system of the MMC supports a command extension function specific for the application. The MCIF also supports commands extended by the secure multimedia card (Secure-MultiMediaCard™\*<sup>1</sup>, hereafter referred to as Secure-MMC).

- Notes: 1. MultiMediaCard™ is a trademark of Infineon Technologies AG.  
Secure-MultiMediaCard™ is a multimedia card with a content protection function.
2. Supports a multimedia card that conforms to the MultiMediaCard System Specification Version 2.11.  
SPI mode is not supported in a multimedia card that conforms to the MultiMediaCard System Specification Version 3.1.

### 19.1 Features

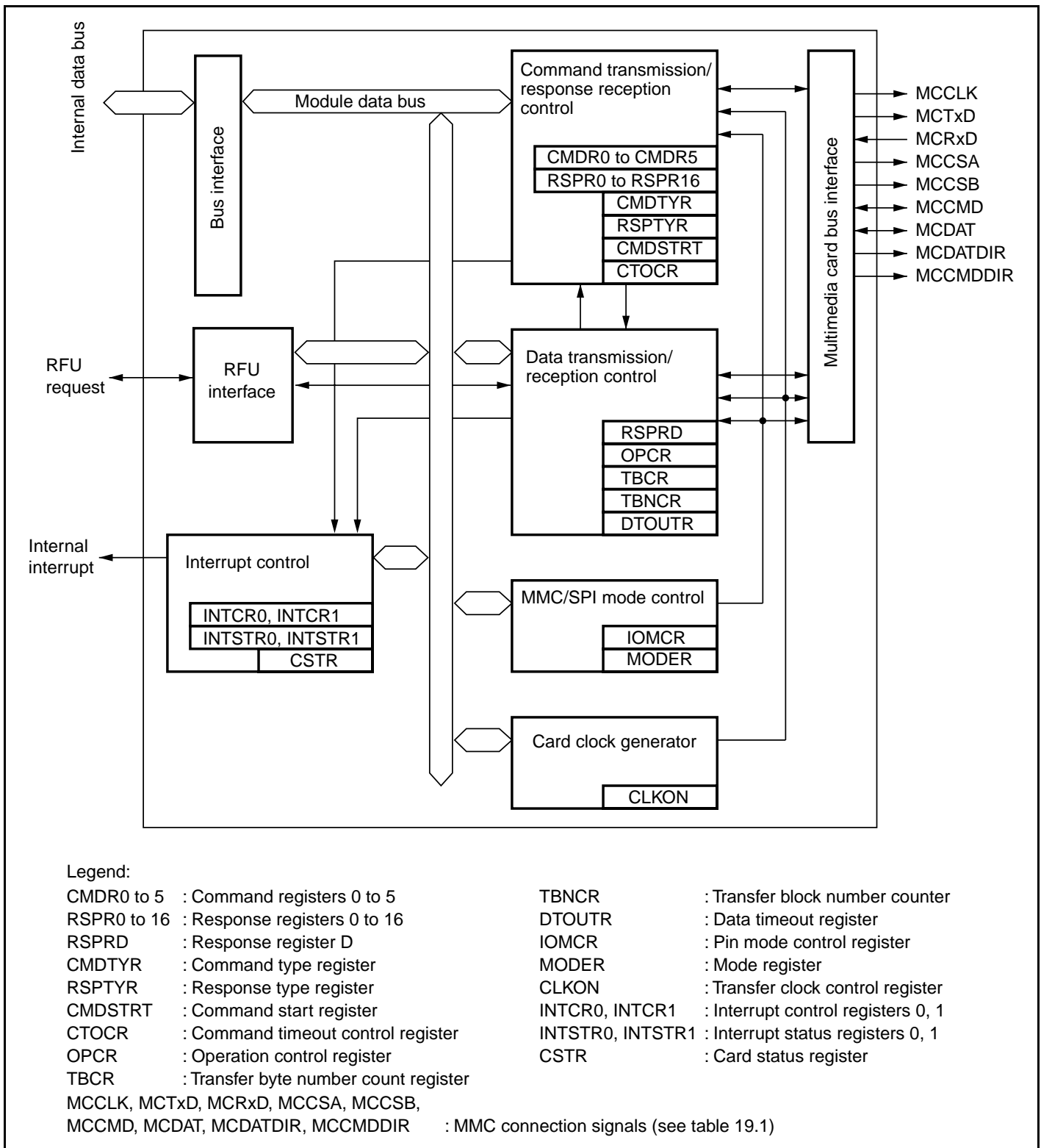
- Two operating modes: MMC mode and SPI mode
- Two MMCs can be inserted in SPI mode (two chip select outputs)
- 20-Mbps bit rate (max) (system clock of 20 MHz)
- Supports MMC commands including the Secure-MMC
- RAM-FIFO by the RFU can be used
- Three interrupt sources: FIFO empty/full, command/response/data transfer complete, and transfer error

MMC mode:

- Interface via the MCCLK pin (transfer clock output) pin, MCCMD pin (command transmission/response reception) , and MCDAT pin (data transmission/reception)
- Monitor output function of the MCCMD and MCDAT pin input/output status

SPI mode:

- Interface via the MCCLK pin (transfer clock output) , MCTxD pin (command transmission/data transmission) , MCRxD pin (response reception/data transmission) and MCCSA and MCCSB pins (chip select) for each MMC.



**Figure 19.1 Block Diagram of MCIF**



## 19.2 Input/Output Pins

Table 19.1 summarizes the pins of the MCIF.

**Table 19.1 Pin Configuration**

<b>Name</b>	<b>I/O</b>	<b>Function</b>
MCCLK, ExMCCLK	Output	Common clock output pins in MMC mode/SPI mode
MCTxD, ExMCTxD	Output	Command/data output pins in SPI mode These pins are connected to the Data in pin on the MMC side
MCRxD, ExMCRxD	Input	Response/data input pins in SPI mode These pins are connected to the Data out pin on the MMC side
MCCSA, ExMCCSA	Output	Card A selection output pins in SPI mode (active-low signal) These pins are connected to the CS pin on the MMC side
MCCSB, ExMCCSB	Output	Card B selection output pins in SPI mode (active-low signal) These pins are connected to the CS pin on the MMC side
MCCMD, ExMCCMD	Input/Output	Command output/response input pins in MMC mode
MCDAT, ExMCDAT	Input/Output	Data input/output pins in MMC mode
MCDATDIR, ExMCDATDIR	Output	Output pins indicating input/output directions of the MCDAT and ExMCDAT pins
MCCMDDIR, ExMCCMDDIR	Output	Output pins indicating input/output directions of the MCCMD and ExMCCMD pins

## 19.3 Register Descriptions

The MCIF has the following registers.

- Mode register (MODER)
- Command type register (CMDTYR)
- Response type register (RSPTYR)
- Transfer byte number count register (TBCR)
- Transfer block number counter (TBNCR)
- Command registers 0 to 5 (CMDR0 to CMDR5)
- Response registers 0 to 16 (RSPR0 to RSPR16)
- Response register D (RSPRD)
- Command start register (CMDSTRT)
- Operation control register (OPCR)
- Command timeout control register (CTOCR)
- Data timeout register (DTOUTR)
- Card status register (CSTR)
- Interrupt control registers 0, 1 (INTCR0, INTCR1)
- Interrupt status registers 0, 1 (INTSTR0, INTSTR1)
- Pin mode control register (IOMCR)
- Transfer clock control register (CLKON)

### 19.3.1 Mode Register (MODER)

MODER specifies the MCIF operating mode. The two operating modes have different input/output pin functions, usable command types, and command sequence. In each mode, the following sequence should be repeated when the MCIF uses the MMC: Send a command, wait for the end of the command sequence and the end of the data busy state, and send the next command.

The series of operations from command sending, command response reception, data transmission/reception, and data response reception is called the command sequence. The command sequence starts from sending a command by setting the START bit in CMDSTRT to 1, and ends when all necessary data transmission/reception and response reception has been completed. The MMC supports the data busy state in which the next command is not accepted to write/erase data to/from the flash memory in the MMC during command sequence execution and after command sequence execution has ended. The data busy state is indicated by a 0 output from the MMC side to the MCDAT pin in MMC mode, and to the MCRxD pin in SPI mode.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
2, 1	—	All 0	R/(W)	Reserved The initial value should not be changed.
0	SPI	0	R/W	SPI/MMC Mode Specifies the MCIF operating mode. 0: Operates in MMC mode 1: Operates in SPI mode

### 19.3.2 Command Type Register (CMDTYR)

CMDTYR specifies the command format in conjunction with RSPTYR. For details, refer to table 19.2.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
5	TY5	0	R/W	Specifies multiblock transfer when an extended command is used in SPI mode. Bits TY1 and TY0 should be set to B'01 or B'10. The transfer block size should be set in TBCR, and the number of transfer blocks should be set in TBNCR when a command to specify this bit is used.
4	TY4	0	R/W	This bit is set to 1 when the CMD12M command is specified. The CMD12M command can be used only in MMC mode. Bits TY1 and TY0 should be cleared to B'00.
3	TY3	0	R/W	Specifies stream transfer. Bits TY1 and TY0 should be set to B'01 or B'10. The stream transfer can be used only in MMC mode. The command sequence of the stream transfer specified by this bit ends when it is aborted by the CMD12M command.
2	TY2	0	R/W	Specifies multiblock transfer in MMC mode. Bits TY1 and TY0 should be set to B'01 or B'10. The command sequence of the multiblock transfer specified by this bit ends when it is aborted by the CMD12M command.
1	TY1	0	R/W	These bits specify the existence and direction of transfer data. 00: A command without data transfer 01: A command with read data reception 10: A command with write data transmission 11: Setting prohibited
0	TY0	0	R/W	

### 19.3.3 Response Type Register (RSPTYR)

RSPTYR specifies the command format in conjunction with CMDTYR. For details, refer to table 19.2.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
5	RTY5	0	R/W	This bit is specified by the R1b response in MMC mode/SPI mode.
4	RTY4	0	R/W	Makes settings so that the command response CRC is checked by bit CRC7. Bits RTY2 to RTY0 should be set to B'011, B'100, or B'101.
3	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
2	RTY2	0	R/W	These bits specify the number of command response bytes. 000: Command needs no command response 001: Command needs a 1-byte command response (specified by R1 and R1b responses in SPI mode) 010: Command needs a 2-byte command response (specified by R2 response in SPI mode) 011: Command needs a 5-byte command response (specified by R3 response in SPI mode) 100: Command needs a 6-byte command response (specified by R1, R1b, R3, R4, and R5 responses in MMC mode) 101: Command needs a 17-byte command response (specified by R2 response in MMC mode) 11X: Setting prohibited
1	RTY1	0	R/W	
0	RTY0	0	R/W	

Legend:

X: Don't Care

Table 19.2 summarizes the correspondence between the commands listed in the MultiMediaCard System Specification Version 2.11 and the settings of CMDTYR and RSPTYR. In MMC mode and SPI mode, the response format is different. Also, in SPI mode, there are commands that require a data response.

**Table 19.2 Correspondence between Commands and Settings of CMDTYR and RSPTYR**

Command Index	Abbreviation	resp <sup>*2</sup>	CMDTYR					RSPTYR		
			TY5	TY4	TY3	TY2	TY1, TY0	RTY5	RTY4 <sup>*3</sup>	RTY2 to RTY0 <sup>*2</sup>
CMD0	GO_IDLE_STATE	—/R1					00			000/001
CMD1	SEND_OP_COND	R3/R1					00			100/001
CMD2M <sup>*1</sup>	ALL_SEND_CID	R2					00			101
CMD3M <sup>*1</sup>	SET_RELATIVE_ADDR	R1					00	1		100
CMD4M <sup>*1</sup>	SET_DSR	—					00			000
CMD7M <sup>*1</sup>	SELECT/DESELECT_CARD	R1b					00	1	1	100
CMD9	SEND_CSD	R2/R1					00			101/001
CMD10	SEND_CID	R2/R1					00			101/001
CMD11M <sup>*1</sup>	READ_DAT_UNTIL_STOP	R1			1		01		1	100
CMD12M <sup>*1</sup>	STOP_TRANSMISSION	R1b		1			00	1	1	100
CMD13	SEND_STATUS	R1/R2					00		1/0	100/010
CMD15M <sup>*1</sup>	GO_INACTIVE_STATE	—					00			000
CMD16	SET_BLOCKLEN	R1					00		1/0	100/001
CMD17	READ_SINGLE_BLOCK	R1					01		1/0	100/001
CMD18M <sup>*1</sup>	READ_MULTIPLE_BLOCK	R1				1	01		1	100
CMD20M <sup>*1</sup>	WRITE_DAT_UNTIL_STOP	R1			1		10		1	100
CMD24	WRITE_SINGLE_BLOCK	R1					10		1/0	100/001
CMD25M <sup>*1</sup>	WRITE_MULTIPLE_BLOCK	R1				1	10		1	100
CMD26M <sup>*1</sup>	PROGRAM_CID	R1					10		1	100
CMD27	PROGRAM_CSD	R1					10		1/0	100/001
CMD28	SET_WRITE_PROT	R1b					00	1	1/0	100/001
CMD29	CLR_WRITE_PROT	R1b					00	1	1/0	100/001
CMD30	SEND_WRITE_PROT	R1					01		1/0	100/001
CMD32	TAG_SECTOR_START	R1					00		1/0	100/001
CMD33	TAG_SECTOR_END	R1					00		1/0	100/001
CMD34	UNTAG_SECTOR	R1					00		1/0	100/001
CMD35	TAG_ERASE_GROUP_START	R1					00		1/0	100/001
CMD36	TAG_ERASE_GROUP_END	R1					00		1/0	100/001
CMD37	UNTAG_ERASE_GROUP	R1					00		1/0	100/001

Command Index	Abbreviation	resp <sup>*2</sup>	CMDTYR					RSPTYR		
			TY5	TY4	TY3	TY2	TY1, TY0	RTY5	RTY4 <sup>*3</sup>	RTY2 to RTY0 <sup>*2</sup>
CMD38	ERASE	R1b					00	1	1/0	100/001
CMD39M <sup>*1</sup>	FAST_IO	R4					00		1	100
CMD40M <sup>*1</sup>	GO_IRQ_STATE	R5					00		1	100
CMD42	LOCK_UNLOCK	R1b					10 <sup>*2</sup>	1	1/0	100/001
CMD55	APP_CMD	R1					00		1/0	100/001
CMD56	GEN_CMD	R1b					10 <sup>*2</sup> , 01 <sup>*4</sup>	1	1/0	100/001
CMD58S <sup>*1</sup>	READ_OCR	R3					00			011
CMD59S <sup>*1</sup>	CRC_ON_OFF	R1					00			001

- Notes:
1. M is added to the end of the command index of commands that are available only in MMC mode. Similarly, S is added to the end of the command index of commands that are available only in SPI mode.
  2. If there is a difference between the response types in MMC mode and SPI mode, they are both listed in the resp column with MMC mode preceding SPI mode. Similarly, the difference between the number of response bytes in the two modes is listed in the RTY2 to RTY0 bit columns in the same order. A data response is used in a command for which the TY1 and TY0 bits are set to B'10 in SPI mode.
  3. Clear the RTY4 bit to 0 in SPI mode. When this bit is set to 1 in MMC mode, command response CRC can be checked.
  4. Can be used as a command with read data.

### 19.3.4 Transfer Byte Number Count Register (TBCR)

TBCR specifies the number of bytes to be transferred (block size) for a block transfer command. The block size is the number of data block bytes not including the start bit (byte in SPI mode) and CRC. The multiblock transfer command in MMC mode corresponds to the number of bytes of each data block. This register setting is ignored in the stream transfer command in MMC mode.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
3	CS3	0	R/W	Transfer Data Block Size
2	CS2	0	R/W	0000: 1 byte
1	CS1	0	R/W	0001: 2 bytes
0	CS0	0	R/W	0010: 4 bytes 0011: 8 bytes 0100: 16 bytes 0101: 32 bytes 0110: 64 bytes 0111: 128 bytes 1000: 256 bytes 1001: 512 bytes 1010: 1024 bytes 1011: 2048 bytes 11XX: Setting prohibited

Legend:

X: Don't Care

### 19.3.5 Transfer Block Number Counter (TBNCR)

TBNCR sets the number of blocks to be transferred when multiblock transfer is specified by bit TY5 in CMDTYR. The contents of TBNCR is decremented for every 1-block transfer completion. When the contents of TBNCR is 0, the command sequence is terminated, and an interrupt is generated.

Bit	Bit Name	Initial Value	R/W	Description
15 to 0	—	All 0	R/W	Transfer Block Number Counter



### 19.3.6 Command Registers 0 to 5 (CMDR0 to CMDR5)

A command is written to CMDR as shown in table 19.3, and a command is transmitted by setting the START bit in CMDSTRT to 1.

**Table 19.3 CMDR Configuration**

Register	Contents	Operation	Remarks
CMDR0	Start bit Host bit Command index of 6 bits	Command index writing Start bit is 0 Host bit is 1	Initial value: H'00
CMDR1 to CMDR4	Command argument of 32 bits	Command argument writing	Initial value: H'00
CMDR5	CRC of 7 bits End bit	Automatic calculation for CRC End bit is fixed to 1.	Initial value : Undefined Always read as H'00

#### CMDR0

Bit	Bit Name	Initial Value	R/W	Description
7	Start	0	R/W	Start Bit (This bit should be cleared to 0)
6	Host	0	R/W	Transmission Bit (This bit should be set to 1)
5 to 0	—	All 0	R/W	Command Indexes

#### CMDR1, CMDR2, CMDR3, CMDR4

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R/W	Command arguments

#### CMDR5

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	CRC	—	—	CRC code (automatic calculation)
0	End	—	—	End bit (fixed to 1)

### 19.3.7 Response Registers 0 to 16, and D (RSPR0 to RSPR16, and RSPRD)

The RSPR registers are eighteen 8-bit registers. RSPR0 to RSPR16 are command response registers. RSPRD is a data response register that is used in only SPI mode.

The number of command response bytes differs according to the command. The number of command response bytes can be specified by RSPTYR in the MCIF. The command response is shifted-in from bit 0 in RSPR16, and shifted to the number of command response bytes  $\times$  8 bits. Table 19.4 summarizes the correspondence between the number of command response bytes and valid RSPR register.

The data response is shifted-in from bit 0 in RSPRD, and shifted 8 bits only when a command includes write data in SPI mode. For other commands, the data response is not shifted. RSPRD is cleared to H'00 by writing an arbitrary value\*<sup>1</sup>.

The initial value of the RSPR registers is H'00. RSPR0 to RSPR16 are simple shift registers. A command response that has been shifted in is not automatically cleared, and is continuously shifted until it is shifted out from bit 7 in RSPR0. To clear unnecessary bytes to H'00, write arbitrary values to each RSPR\*<sup>2</sup>.

- Notes:
1. Bits 7 to 5 in RSPRD are fixed at 0.
  2. Reading the data response from RSPR should be executed after one transfer clock cycle following the DRPI interrupt occurrence. Clearing of RSPR is completed after two transfer clock cycles following the write of arbitrary values.

**Table 19.4 Correspondence between Number of Command Response Bytes and RSPR Register**

RSPR Register	SPI Mode Response			MMC Mode Response	
	1 Byte in SPI Mode (R1, R1b)	2 Bytes in SPI Mode (R2)	5 Bytes in SPI Mode (R3)	6 Bytes in MMC Mode (R1, R1b, R3, R4)	17 Bytes in MMC Mode (R2)
RSPR0	—	—	—	—	1st byte
RSPR1	—	—	—	—	2nd byte
RSPR2	—	—	—	—	3rd byte
RSPR3	—	—	—	—	4th byte
RSPR4	—	—	—	—	5th byte
RSPR5	—	—	—	—	6th byte
RSPR6	—	—	—	—	7th byte
RSPR7	—	—	—	—	8th byte
RSPR8	—	—	—	—	9th byte
RSPR9	—	—	—	—	10th byte
RSPR10	—	—	—	—	11th byte
RSPR11	—	—	—	1st byte	12th byte
RSPR12	—	—	1st byte	2nd byte	13th byte
RSPR13	—	—	2nd byte	3rd byte	14th byte
RSPR14	—	—	3rd byte	4th byte	15th byte
RSPR15	—	1st byte	4th byte	5th byte	16th byte
RSPR16	1st byte	2nd byte	5th byte	6th byte	17th byte

### 19.3.8 Command Start Register (CMDSTRT)

CMDSTRT triggers the start of command transmission, representing the start of a command sequence. The following operations should be completed before the command sequence starts.

- Analysis of prior command response, clearing the command response register write if necessary
- Analysis/transfer of receive data of prior command if necessary
- Preparation of transmission data of the next command if necessary
- Setting of CMDTYR, RSPTYR, TBCR, and TBNCR  
CMDTYR, RSPTYR, TBCR, and TBNCR should not be changed until the command sequence has ended.
- Setting of CMDR0 to CMDR4  
CMDR0 to CMDR4 should not be changed until the command sequence has ended (the CWRE flag in CSTR has been reset, or a command transmission end interrupt has been generated).

The command sequences are controlled by the sequencers in each MCIF side and MMC side. Normally, these operate synchronously, however, these may temporarily become unsynchronized when an error occurs or when a command is aborted. Take care to set the CMDOFF bit in OPCR, to issue the CMD12 command, and to process an error in MMC mode. A new command sequence should be started after confirming that the command sequences on both the MCIF and MMC sides have ended.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R/(W)	Reserved The initial value should not be changed.
0	START	0	R/W	Command Sequence Start Bit Starts command transmission when 1 is written. This bit is always read as 0. [Clearing condition] <ul style="list-style-type: none"> <li>• Automatically cleared when command transmission starts</li> </ul>

### 19.3.9 Operation Control Register (OPCR)

OPCR controls command operation abort, and suspends or continues data transfer.

Bit	Bit Name	Initial Value	R/W	Description
7	CMDOFF	0	W	<p>Command Off</p> <p>Always read as 0. Aborts all command operations (MCIF command sequence) when 1 is written after a command is transmitted.</p> <p>Write enable period: From command transmission completion to command sequence end</p> <p>0: Operation is not affected.</p> <p>1: Command sequence is forcibly aborted. Byte transfer during transfer is also suspended.</p> <p>After command sequence abort, the transfer clock output resumes if the transfer clock has been halted during the command sequence.</p>
6	—	0	R/(W)	<p>Reserved</p> <p>The initial value should not be changed.</p>
5	RD_CONTI	0	W	<p>Read Continue</p> <p>Read as 1 until resuming read when 1 is written. Otherwise, read as 0. Resumes transfer clock output and read data reception when the transfer clock is halted according to FIFO full or termination of block reading in multiblock read.</p> <p>Write enable period: While MCCLK for read data reception is halted</p> <p>0: Operation is not affected.</p> <p>1: Resumes MCCLK output and read data reception.</p>

Bit	Bit Name	Initial Value	R/W	Description
4	DATAEN	0	R/W	<p>Data Enable</p> <p>Read as 1 during data transfer period after 1 is written. Otherwise, read as 0. Starts write data transmission by a command with write data. Resumes transfer clock output and write data transmission when the transfer clock is halted according to FIFO empty or termination of one block writing in multiblock write.</p> <p>Write enable period:</p> <ul style="list-style-type: none"> <li>• After transmission of a command with write data</li> <li>• While transfer clock is halted according to FIFO empty</li> <li>• When one block writing in multiblock write is terminated</li> </ul> <p>0: Operation is not affected. 1: Starts or resumes transfer clock output and write data transmission.</p>
3 to 0	—	All 0	R/(W)	<p>Reserved</p> <p>The initial value should not be changed.</p>

The command sequence on the MMC side may be halted according to the status of MMC. Table 19.5 shows the MMC states in which the command sequence is halted. In this case, the command sequence should be aborted by setting the CMDOFF bit to 1 on the MCIF side as required.

**Table 19.5 Card States in which Command Sequence Is Halted**

Operating Mode		Error Status
MMC mode	Command response	When the error detection bit in the card status (32 bits) in the command response transmitted by the MMC is set.
	Data status	When the error detection bit in the CRC status (3 bits) to be transmitted from the MMC while block data is transmitted to the MMC is set.
SPI mode	Command response	When the error detection bit in the card status (8 bits) in the command response transmitted by the MMC is set.
	Data response	When the error detection bit in the data response (8 bits) to be transmitted from the MMC while block data is transmitted to the MMC is set.

For write data transmission, the contents of the command response and data response should be analyzed, and then transmission should be triggered. In addition, the transfer clock (MCCLK) output should be temporarily halted according to FIFO full/empty, and it should be resumed when preparation has been completed.

For multiblock transfer in MMC mode, the command sequence should be temporarily halted for every block break to select either to continue to the next block or to abort the multiblock transfer command by issuing the CMD12 command, and then the command sequence should be resumed. The transfer clock output is also halted while the command sequence is halted in a multiblock read. To continue to the next block, the RD\_CONTI and DATAEN bits should be set to 1. To issue the CMD12 command, the CMDOFF bit should be set to 1 to abort the command sequence on the MCIF side.

Any temporary halt of the transfer clock is determined one byte before the FIFO becomes full/empty. The DATAEN bit should not be set to 1 when only one byte of data remains in the transmit data FIFO. The receive data FIFO should be read when the FIFO becomes full, and the RD\_CONTI bit should be set to 1 after three bytes or more have been read.

### 19.3.10 Command Timeout Control Register (CTOCR)

CTOCR specifies the cycle to generate a timeout for the command response.

The counter (CTOUTC), to which the CPU does not have access, counts the transfer clock to monitor the command timeout. The CTOUTC starts counting the transfer clock from the start of command transmission. The CTOUTC stops counting the transfer clock when command response reception has been completed, or when the command sequence has been aborted by setting the CMDOFF bit to 1.

When the command response cannot be received, the CTOUTC continues counting the transfer clock, and enters the command timeout error state when the number of transfer clock reaches the number specified in CTOCR. When the CTERIE bit in INTCR1 is set to 1, the CTERI flag in INTSTR1 is set. As the CTOUTC continues counting the transfer clock, the CTERI flag setting condition is repeatedly generated. To perform command timeout error handling, the command sequence should be aborted by setting the CMDOFF bit to 1, and then the CTERI flag should be cleared to prevent extra-interrupt generation.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
0	CTSEL0	0	R/W	Command Timeout Select Specifies the number of transfer clocks from command transmission completion to response reception completion. 0: 128 transfer clocks 1: 256 transfer clocks

### 19.3.11 Data Timeout Register (DTOUTR)

DTOUTR specifies the cycle to generate a data timeout. The 16-bit counter (DTOUTC) and a prescaler, to which the CPU does not have access, count the system clock to monitor the data timeout. The prescaler always counts the system clock, and outputs a count pulse for every 10000 system clocks. The DTOUTC starts counting the prescaler output from the start of the command sequence. The DTOUTC is cleared when the command sequence has ended, or when the command sequence has been aborted by setting the CMDOFF bit to 1, after which the DTOUTC stops counting the prescaler output.

When the command sequence does not end, the DTOUTC continues counting the prescaler output, and enters the data timeout error state when the number of prescaler outputs reaches the number specified in DTOUTR. When the DTERIE bit in INTCR1 is set to 1, the DTERI flag in INTSTR1 is set. As the DTOUTC continues counting prescaler output, the DTERI flag setting condition is repeatedly generated. To perform data timeout error handling, the command sequence should be aborted by setting the CMDOFF bit to 1, and then the DTERI flag should be cleared to prevent extra-interrupt generation.

For a command with data busy status, as the command sequence is terminated before entering the data busy state, data timeout cannot be monitored. Timeout in the data busy state should be monitored by firmware.

Bit	Bit Name	Initial Value	R/W	Description
15 to 0	—	All 1	R/W	Data Timeout Time/10000 Data timeout time can be obtained by system clock cycle × DTOUTR setting value × 10000.



### 19.3.12 Card Status Register (CSTR)

CSTR indicates the MCIF status during command sequence execution.

Bit	Bit Name	Initial Value	R/W	Description
7	BUSY	0	R	<p>Command Busy</p> <p>Indicates command execution status. When the CMDOFF bit in OPCR is set to 1, this bit is cleared to 0 because the MCIF command sequence is aborted.</p> <p>0: Command sequence has ended.</p> <p>1: Command sequence execution in progress.</p>
6	FIFO_FULL	0	R	<p>FIFO Full</p> <p>Indicates whether receive data FIFO full has been detected.</p> <p>0: Receive data FIFO full is not detected.</p> <p>1: Receive data FIFO full is detected.</p> <p>After FIFO full detection, this bit is cleared to 0 when resuming to receive read data from the MMC or when the command sequence ends.</p>
5	FIFO_EMPTY	0	R	<p>FIFO Empty</p> <p>Indicates whether transmit data FIFO empty has been detected.</p> <p>0: Transmit data FIFO empty is not detected.</p> <p>1: Transmit data FIFO empty is detected.</p> <p>After FIFO empty detection, this bit is cleared to 0 when resuming to transmit data to the MMC or when the command sequence ends.</p>
4	CWRE	0	R	<p>Command Register Write Enable</p> <p>Indicates whether the CMDR command is being transmitted or has been transmitted.</p> <p>0: The CMDR command has been transmitted, or the START bit in CMDSTRT has not been set yet, so the new command can be written.</p> <p>1: The CMDR command is waiting for transmission or is being transmitted. If the new command is written, a malfunction will result.</p>

Bit	Bit Name	Initial Value	R/W	Description
3	DTBUSY	0	R	<p>Data Busy</p> <p>Indicates command execution status. Indicates that the MMC is in the busy state during or after the command sequence of a command without data transfer, which includes the busy state in the response, or of a command with write data has been ended.</p> <p>0: Idle state waiting for a command, or command sequence execution in progress</p> <p>1: MMC indicates data busy after command sequence ends.</p>
2	DTBUSY_TU	—	R	<p>Data Busy Pin Status</p> <p>Monitors level of the MCDAT pin in MMC mode or MCRxD pin in SPI mode.</p> <p>This bit is monitored to confirm whether the MMC is in busy state by deselecting the MMC in busy state, and then selecting the MMC, again.</p>
1	—	0	R	<p>Reserved</p> <p>This bit is always read as 0 and cannot be modified.</p>
0	REQ	0	R	<p>Interrupt Request</p> <p>Indicates whether an interrupt is requested. An interrupt request is the logical sum of the INTSTR0 and INTSTR1 flags. The INTSTR0 and INTSTR1 flags are set by the enable bits in INTCR0 and INTCR1.</p> <p>0: No interrupts requested.</p> <p>1: An interrupt is requested.</p>

### 19.3.13 Interrupt Control Registers 0, 1 (INTCR0, INTCR1)

The INTCR registers enable or disable an interrupt.

#### INTCR0

Bit	Bit Name	Initial Value	R/W	Description
7	FEIE	0	R/W	FIFO Empty Interrupt Enable When this bit is set to 1 while the INTRQ0E bit is 1, the data FIFO empty interrupt request is enabled.
6	FFIE	0	R/W	FIFO Full Interrupt Enable When this bit is set to 1 while the INTRQ0E bit is 1, the receive data FIFO full interrupt request is enabled.
5	DRPIE	0	R/W	Data Response Interrupt Enable When this bit is set to 1 in SPI mode with the INTRQ1E bit as 1, the data response interrupt request is enabled.
4	DTIE	0	R/W	Data Transfer End Interrupt Enable When this bit is set to 1 while the INTRQ1E bit is 1, the data transfer end interrupt request is enabled.
3	CRPIE	0	R/W	Command Response End Interrupt Enable When this bit is set to 1 while the INTRQ1E bit is 1, the command response end interrupt request is enabled.
2	CMDIE	0	R/W	Command Transmission End Interrupt Enable When this bit is set to 1 while the INTRQ1E bit is 1, the command transmission end interrupt request is enabled.
1	DBSYIE	0	R/W	Data Busy End Interrupt Enable When this bit is set to 1 while the INTRQ1E bit is 1, the data busy end interrupt request is enabled.
0	BTIE	0	R/W	Multiblock Transfer End Interrupt Enable When this bit is set to 1 with the INTRQ1E bit as 1, the multiblock transfer end interrupt request is enabled.

**INTCR1**

<b>Bit</b>	<b>Bit Name</b>	<b>Initial Value</b>	<b>R/W</b>	<b>Description</b>
7	INTRQ2E	0	R/W	Source 2 Interrupt Enable 0: Disables MCIF2 interrupt to the CPU. 1: Enables MCIF2 interrupt to the CPU.
6	INTRQ1E	0	R/W	Source 1 Interrupt Enable 0: Disables MCIF1 interrupt to the CPU. 1: Enables MCIF1 interrupt to the CPU.
5	INTRQ0E	0	R/W	Source 0 Interrupt Enable 0: Disables MCIF0 interrupt to the CPU. 1: Enables MCIF0 interrupt to the CPU.
4, 3	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
2	CRCERIE	0	R/W	CRC Error Interrupt Enable When this bit is set to 1 while the INTRQ2E bit is 1, the CRC error interrupt request is enabled.
1	DTERIE	0	R/W	Data Timeout Error Interrupt Enable When this bit is set to 1 while the INTRQ2E bit is 1, the data timeout error interrupt request is enabled.
0	CTERIE	0	R/W	Command Timeout Error Interrupt Enable When this bit is set to 1 while the INTRQ2E bit is 1, the command timeout error interrupt request is enabled.

### 19.3.14 Interrupt Status Registers 0, 1 (INTSTR0, INTSTR1)

The INTSTR registers enable or disable the MCIF10 to MCIF12 interrupt requests of MCIF.

#### INTSTR0

Bit	Bit Name	Initial Value	R/W	Description
7	FEI	0	R/(W)*	<p>FIFO Empty Interrupt Flag</p> <p>0: No interrupts 1: MCIF10 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When transmit data FIFO becomes empty while FEIE = 1 in INTCR0 (when the FIFO_EMPTY bit in CSTR is set)</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading FEI = 1.</li> </ul>
6	FFI	0	R/(W)*	<p>FIFO Full Interrupt Flag</p> <p>0: No interrupts 1: MCIF10 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When receive data FIFO becomes full while FFIE = 1 in INTCR0 (when the FIFO_FULL bit in CSTR is set)</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading FFI = 1.</li> </ul>
5	DRPI	0	R/(W)*	<p>Data Response Interrupt Flag</p> <p>0: No interrupts 1: MCIF11 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>If DRPIE = 1 in INTCR0, when CRC status (in MMC mode) or data response (in SPI mode) is received from the MMC after single-block transmission.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading DRPI = 1.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
4	DTI	0	R/(W)*	<p>Data Transfer End Interrupt Flag</p> <p>0: No interrupts 1: MCIF11 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the number of bytes of data transfer specified in TBCR ends while DTIE = 1 in INTCR0.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading DTI = 1.</li> </ul>
3	CRPI	0	R/(W)*	<p>Command Response End Interrupt Flag</p> <p>0: No interrupts 1: MCIF11 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When command response reception ends while CRPIE = 1 in INTCR0.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading CRPI = 1.</li> </ul>
2	CMDI	0	R/(W)*	<p>Command Transmission End Interrupt Flag</p> <p>0: No interrupts 1: MCIF11 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When command transmission ends while CMDIE = 1 in INTCR0. (When the CWRE bit in CSTR is cleared.)</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading CMDI = 1.</li> </ul>

Bit	Bit Name	Initial Value	R/W	Description
1	DBSYI	0	R/(W)*	<p>Data Busy End Interrupt Flag</p> <p>0: No interrupts 1: MCIF11 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When data busy state ends while DBSYIE = 1 in INTCR0. (When the DTBUSY bit in CSTR is cleared.)</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading DBSYI = 1.</li> </ul>
0	BTI	0	R/(W)*	<p>Multiblock Transfer End Interrupt Flag</p> <p>0: No interrupts 1: MCIF11 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When the number of bytes of data transfer specified in TBCR ends after TBNCR has been decremented to 0 by the extension command (multiblock transfer) in SPI mode while BTIE = 1 in INTCR0.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading BTI = 1.</li> </ul>

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

## INTSTR1

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
2	CRCERI	0	R/(W)*	<p>CRC Error Interrupt Flag</p> <p>0: No interrupts 1: MCIF12 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a CRC error for command response or receive data, and status error (CRC error) for transmission data response is detected while CRCERIE = 1 in INTCR1.</li> </ul> <p>For the command response, CRC should be checked when the RTY4 bit in RSPTYR is enabled.</p> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading CRCERI = 1.</li> </ul>
1	DTERI	0	R/(W)*	<p>Data Timeout Error Interrupt Flag</p> <p>0: No interrupts 1: MCIF12 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a data timeout error specified in DTOUTR occurs while DTERIE = 1 in INTCR1.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading DTERI = 1.</li> </ul>
0	CTERI	0	R/(W)*	<p>Command Timeout Error Interrupt Flag</p> <p>0: No interrupts 1: MCIF12 interrupt requested.</p> <p>[Setting condition]</p> <ul style="list-style-type: none"> <li>When a command timeout error specified in CTOCR occurs while CTERIE = 1 in INTCR1.</li> </ul> <p>[Clearing condition]</p> <ul style="list-style-type: none"> <li>Write 0 after reading CTERI = 1.</li> </ul>

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



### 19.3.15 Pin Mode Control Register (IOMCR)

IOMCR controls chip select pin operation in SPI mode and input/output direction output pin operation in MMC mode.

Bit	Bit Name	Initial Value	R/W	Description
7	SPCNUM	0	R/W	<p>Number of SPI Mode MMCs</p> <p>Specifies whether one or two MMCs are to be operated in SPI mode.</p> <p>0: One MMC is connected to the MCCSA pin. Disables MCCSB pin output.</p> <p>1: Maximum of two MMCs are connected to the MCCSA and MCCSB pins.</p>
6	CHIPSA	0	R/W	<p>MMC Selection <math>\bar{A}/B</math></p> <p>Specifies selection of two MMCs while SPCNUM = 1.</p> <p>0: Outputs CS signal from the MCCSA pin, and sets the MCCSB pin high (no selection).</p> <p>1: Sets the MCCSA pin high (no selection), and outputs CS signal from the MCCSB pin.</p>
5 to 2	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>
1	DIRME	0	R/W	<p>Signal Direction Output Enable</p> <p>Enables/disables the outputs of the MCCMDDIR and MCDATDIR pins that output the input/output direction of the MCCMD and MCDAT pins, in MMC mode.</p> <p>0: Disables MCCMDDIR pin and MCDATDIR pin outputs.</p> <p>1: Enables MCCMDDIR pin and MCDATDIR pin outputs.</p>

Bit	Bit Name	Initial Value	R/W	Description
0	MMCPE	0	R/W	<p>MCIF Pin Function Enable</p> <p>Enables/disables input/output of all MCIF input/output pins.</p> <p>0: Disables all inputs/outputs.</p> <p>1: Enables MCCLK, MCCMD/MCTxD, MCDAT/MCRxD, MCCSA/MCDATDIR, and MCCSB/MCCMDDIR pin inputs/outputs.</p> <p>Outputs of the MCCSB and MCDATDIR and MCCMDDIR pins are also disabled via the SPCNUM bit and DIRME bit, respectively.</p>

### 19.3.16 Transfer Clock Control Register (CLKON)

CLKON controls the transfer clock frequency and clock ON/OFF.

A 20-MHz system clock is needed, and bits CSEL2 to CSEL0 should be set to B'100 for a 20-Mbps transfer clock according to the limitation of the maximum operating frequency of this LSI. At this time, bits CSEL2 to CSEL0 should be cleared to B'000 for a 200-kbps transfer clock in the open drain format output status in MMC mode.

Bit	Bit Name	Initial Value	R/W	Description
7	CLKON	0	R/W	<p>Clock On</p> <p>0: Fixes the transfer clock output from the MCCLK pin to low.</p> <p>1: Outputs the transfer clock from the MCCLK pin.</p>
6 to 3	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0 and cannot be modified.</p>
2	CSEL2	0	R/W	Transfer Clock Frequency Select
1	CSEL1	0	R/W	000: Uses $\phi/100$ as a transfer clock.
0	CSEL0	0	R/W	<p>001: Uses <math>\phi/8</math> as a transfer clock.</p> <p>010: Uses <math>\phi/4</math> as a transfer clock.</p> <p>011: Uses <math>\phi/2</math> as a transfer clock.</p> <p>100: Uses <math>\phi</math> as a transfer clock.</p> <p>101 to 111: Setting prohibited</p>

## 19.4 MCIF Activation

The MMC is an external storage media that can be disconnected. The MCIF controls data transfer with the MMC, however, it cannot control insertion and ejection of the MMC successfully. Firmware should be used to control insertion and ejection of the MMC by using an external interrupt and general ports of this LSI.

### 19.4.1 Initial Status

The power supplied to the MMC is turned on or off by general port output of this LSI. Inputs/outputs of the input/output pins such as MCCLK, MCCMD/MCTxD, MCDAT/MCRxD, MCCSA/MCDATDIR, and MCCSB/MCCMDDIR are enabled/disabled by the MMCPE bit in IOMCR. When the MMCPE bit is set to 1, the CLKON bit should be cleared to 0 so that the transfer clock is applied after the other inputs/outputs are stabilized.

### 19.4.2 Activation Procedure

When MMC insertion is detected, the MCIF should be activated by the following procedures:

#### MMC Mode:

- Enable power supply to the MMC by general port output.
- Enable input/output signals by setting the MMCPE bit to 1.
- Start transfer clock application by setting the CLKON bit to 1.

#### SPI Mode:

- Enable power supply to the MMC by general port output.
- Specify bits for SPCNUM and CHIPSA in IOMCR according to the number of MMC slots and inserted MMCs.
- Enable input/output signals by setting the MMCPE bit to 1.
- Start transfer clock application by setting the CLKON bit to 1.

## 19.5 Operations in MMC Mode

MMC mode is an operating mode in which the transfer clock is output from the MCCLK pin, command transmission/response receive occurs via the MCCMD pin, and data is transmitted/received via the MCDAT pin. In this mode the next command can be issued while data is being transmitted/received. This feature can be applied to the multiblock transfer and stream transfer in which the number of bytes for data transfer does not need to be specified beforehand. In this case, the next command is the CMD12 command, which aborts the current command sequence.

In MMC mode, a broadcast command that simultaneously issues a command to multiple MMCs is supported. After the information for the MMC that is inserted by using the broadcast command is acknowledged, a relative address is given to each MMC. One MMC is selected by the relative address, other MMCs are deselected, and various commands are issued to the selected MMC.

### 19.5.1 Operation of Broadcast Commands

The CMD0, CMD1, CMD2M, and CMD4M are broadcast commands. The command sequence assigning relative addresses to individual MMCs consists of these commands and the CMD3M command. In this command sequence, the CMD output format of the MMC side is open drain, and the command response is wired-OR. During the issuance of this command sequence, the transfer clock frequency should be set sufficiently slow. Operation of a typical command is summarized below.

- All MMCs are initialized to the idle state by CMD0.
- The operation condition register (OCR) of all MMCs is read by CMD1 via wired-OR, and MMCs that cannot operate are deactivated. The deactivated MMCs enter the ready state.
- The card identification (CID) of all MMCs in the ready state is read by CMD2M via wired-OR. The individual MMC compares its CID and data on the CMD, and if different, aborts CID output. A single MMC in which the CID can be entirely output enters the acknowledge state.
- A relative address (RCA) is given to the MMC in the acknowledge state by the CMD3M. The MMC to which the RCA is given enters the standby state.
- CMD2M and CMD3M are repeated, assigning RCAs to all MMCs in the ready state, entering each into the standby state.

### 19.5.2 Operation of Relative Address Commands

The CMD7M, CMD9, CMD10, CMD13, CMD15M, CMD39M, and CMD55 are relative address commands that address the MMC by the RCA. The relative address commands are used to read MMC administration information and original information, and to change the specific MMC status. Operation of a typical command is summarized below.

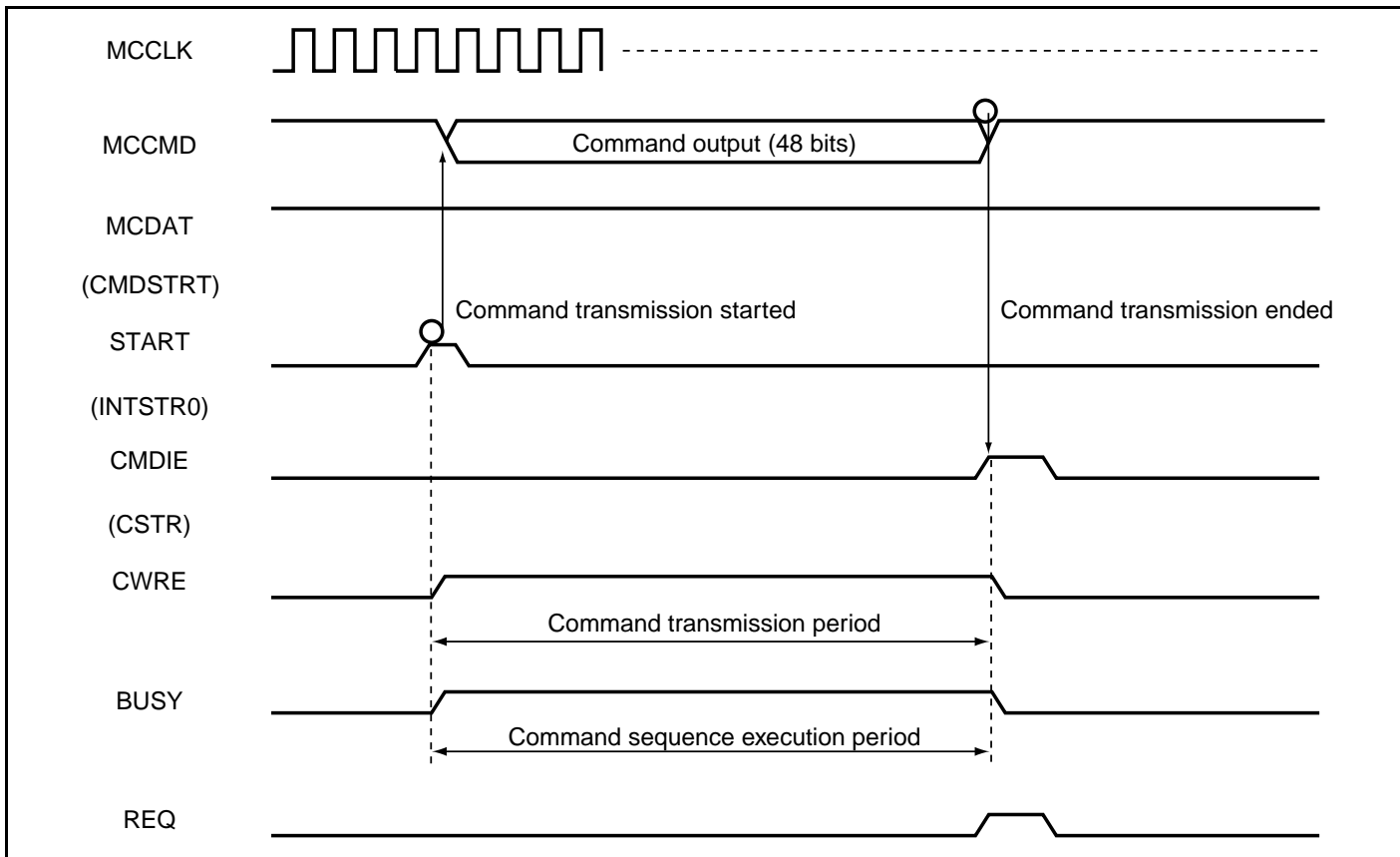
- CMD7M sets one addressed MMC to the transfer state, and other MMCs to the standby state. Commands requiring data transmission/reception can be executed for the MMC in the transfer state.
- CMD15M sets the addressed MMC to the inactive state.
- CMD55 sets the addressed MMC to the application original extension command acceptance state.

### 19.5.3 Operation of Commands Not Requiring Command Response

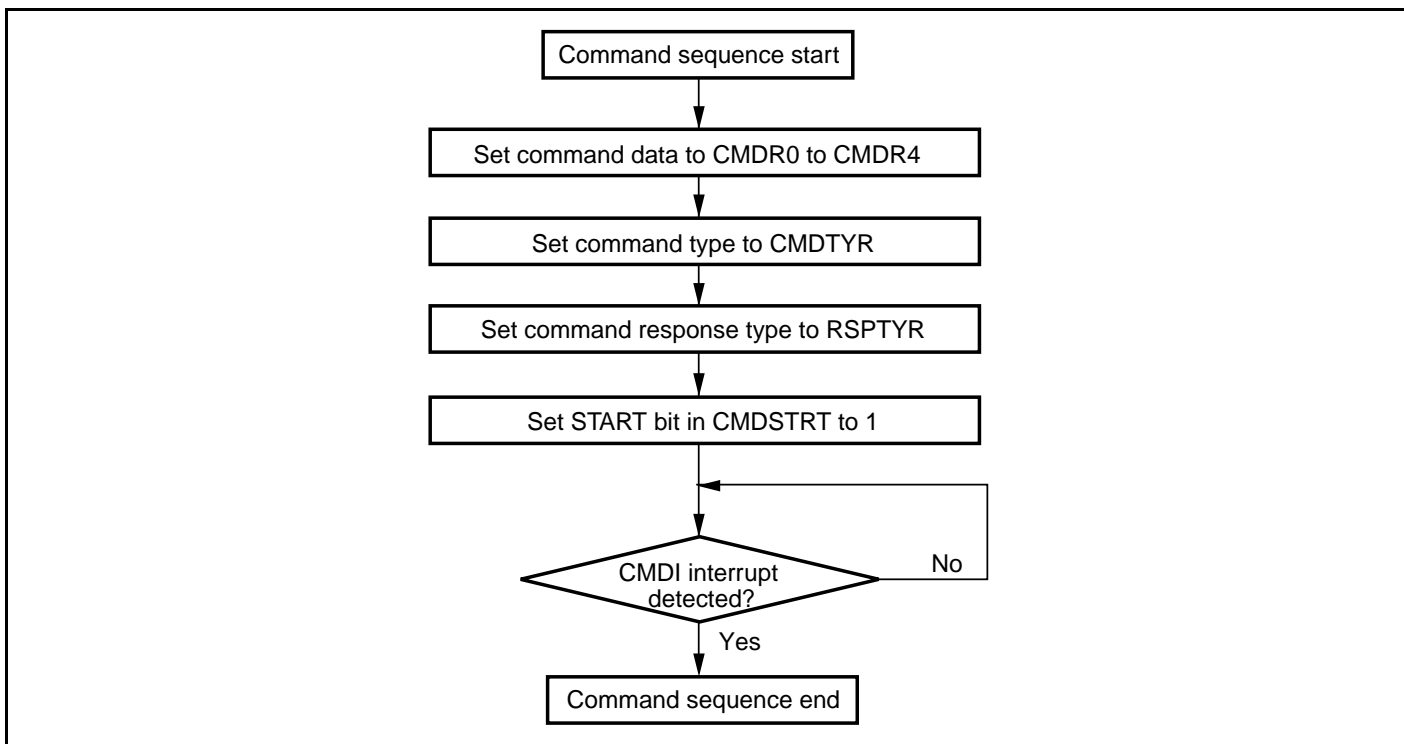
In MMC mode, commands do not require command responses.

Figure 19.2 shows an example of the command sequence for commands that do not require command responses. Figure 19.3 shows the operational flow for commands that do not require command responses.

- The settings needed to issue a command are made.
- The START bit in CMDSTRT is set to 1 to start command transmission.
- The end of a command sequence is detected by polling the BUSY flag in CSTR or by the command transmission end interrupt (CMDI).



**Figure 19.2 Example of Command Sequence for Commands that Do Not Require Command Response**



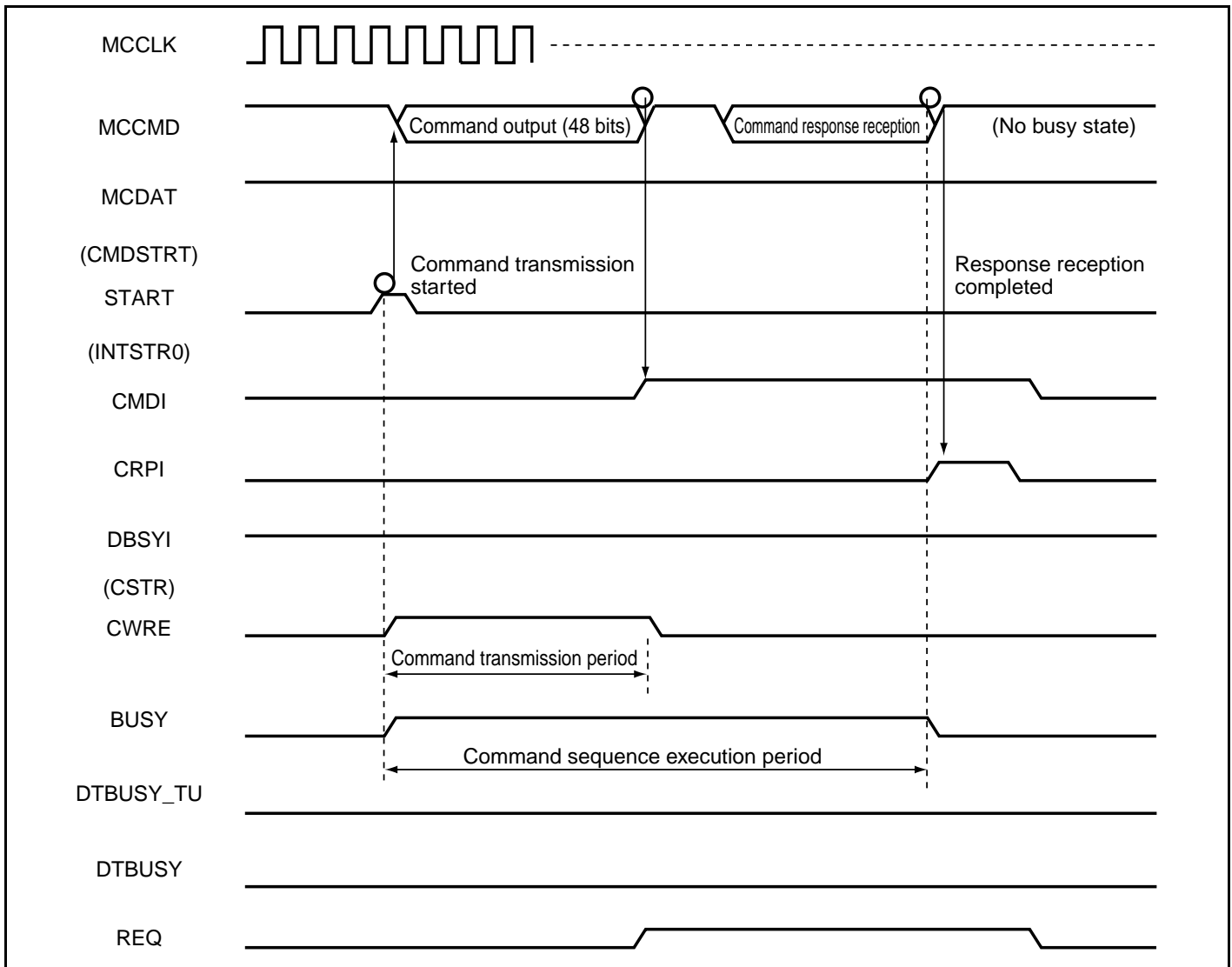
**Figure 19.3 Operational Flow for Commands that Do Not Require Command Response**

### 19.5.4 Operation of Commands without Data Transfer

The broadcast and relative address commands include a number of commands that do not include data transfer. Such commands execute the desired data transfer using command arguments and command responses. For a command that is related to time-consuming processing such as flash memory write/erase, the MMC indicates the data busy state via the DAT pin.

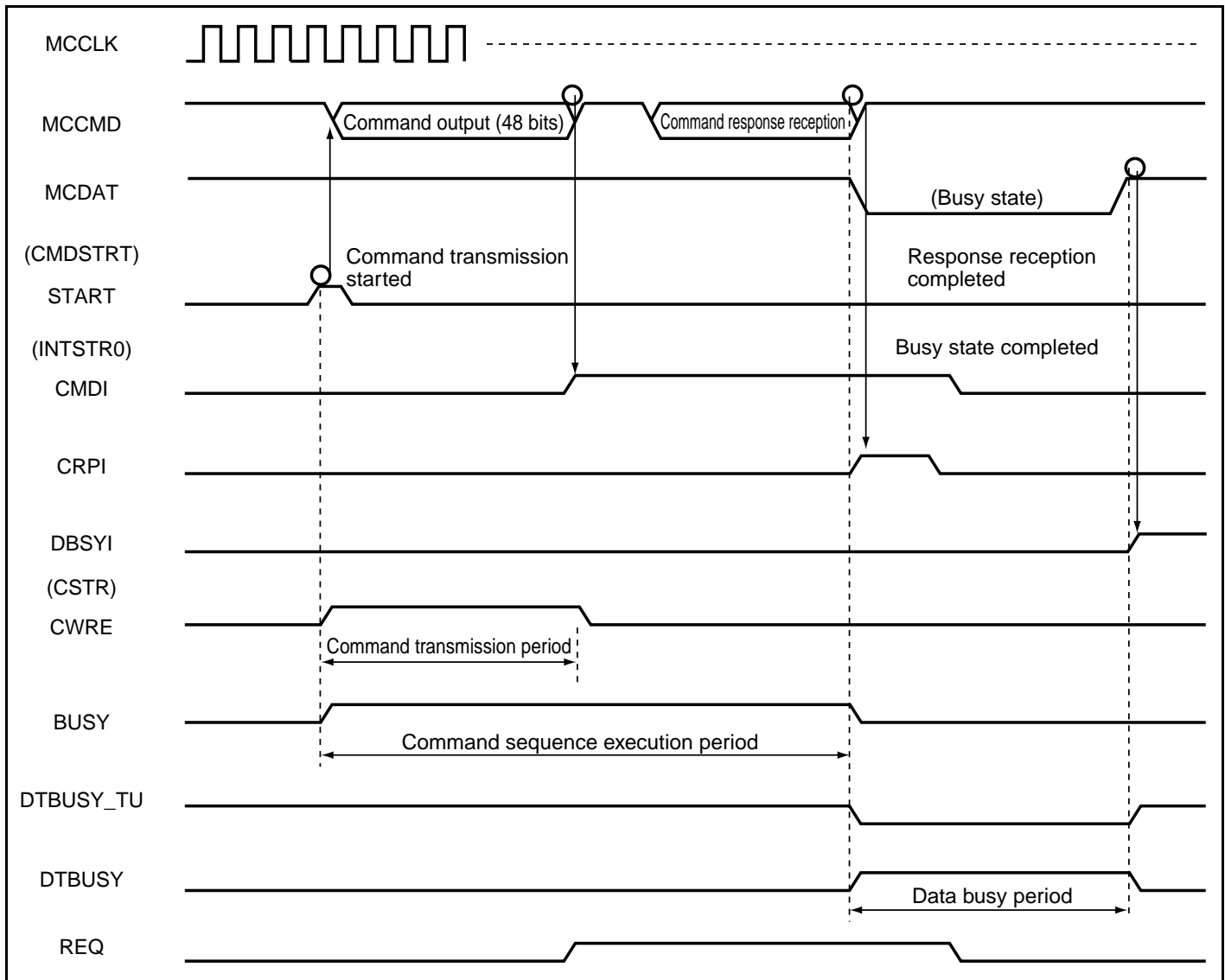
Figures 19.4 and 19.5 show examples of the command sequence for commands without data transfer. Figure 19.6 shows the operational flow for commands without data transfer.

- Settings needed to issue a command are made.
- The START bit in CMDSTRT is set to 1 to start command transmission. Command transmission complete can be confirmed by the command transmission end interrupt (CMDI).
- A command response is received from the MMC. If the MMC does not return the command response, the command response is detected by the command timeout error (CTERI).
- The end of a command sequence is detected by polling the BUSY flag in CSTR or by the command response end interrupt (CRPI).
- The end of the data busy state is detected by polling the DTBUSY flag in CSTR or by the data busy end interrupt (DBSYI).

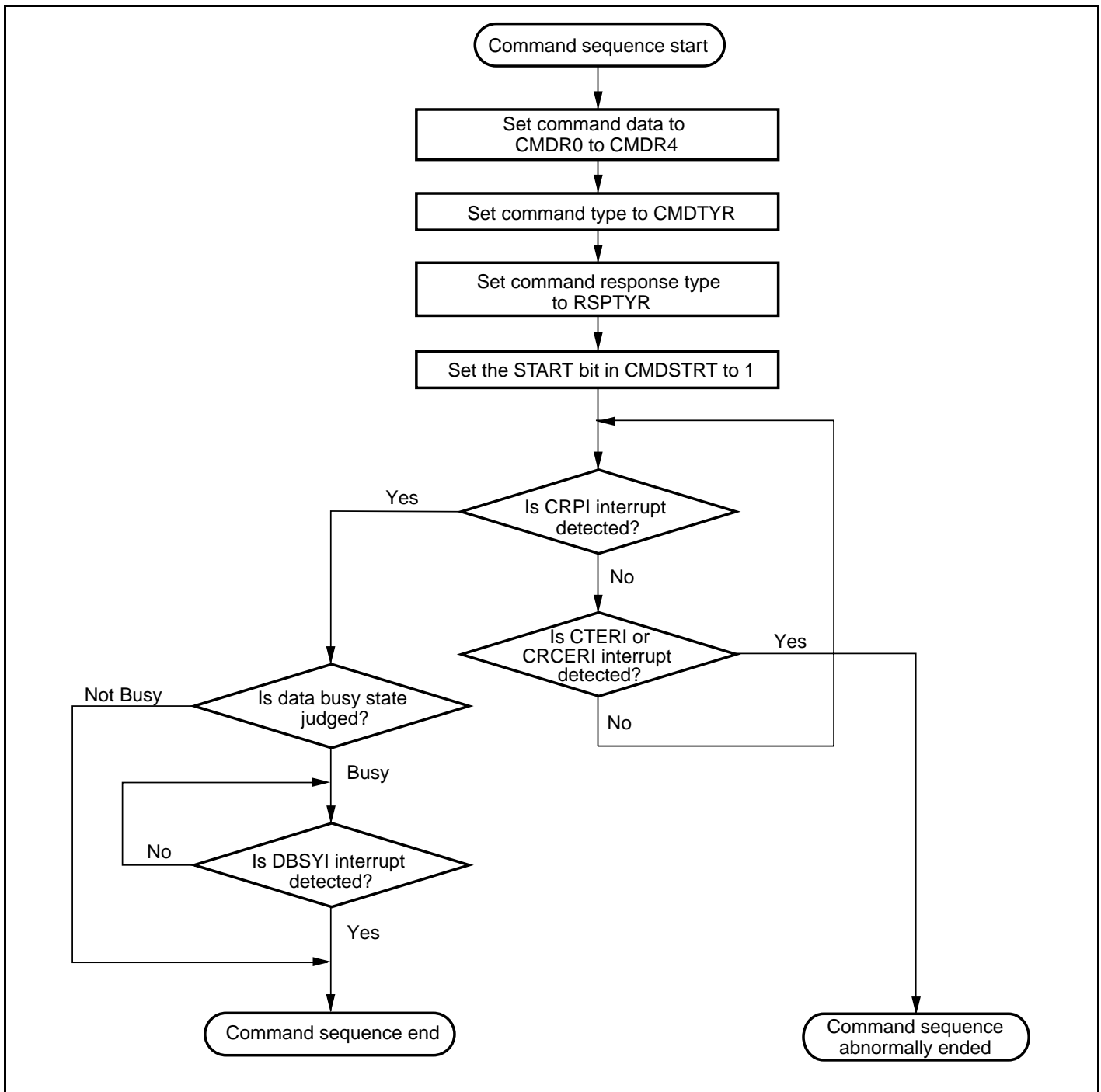


**Figure 19.4 Example of Command Sequence for Commands without Data Transfer (No Data Busy State)**





**Figure 19.5 Example of Command Sequence for Commands without Data Transfer (with Data Busy State)**



**Figure 19.6 Operational Flow for Commands without Data Transfer**

### 19.5.5 Commands with Read Data

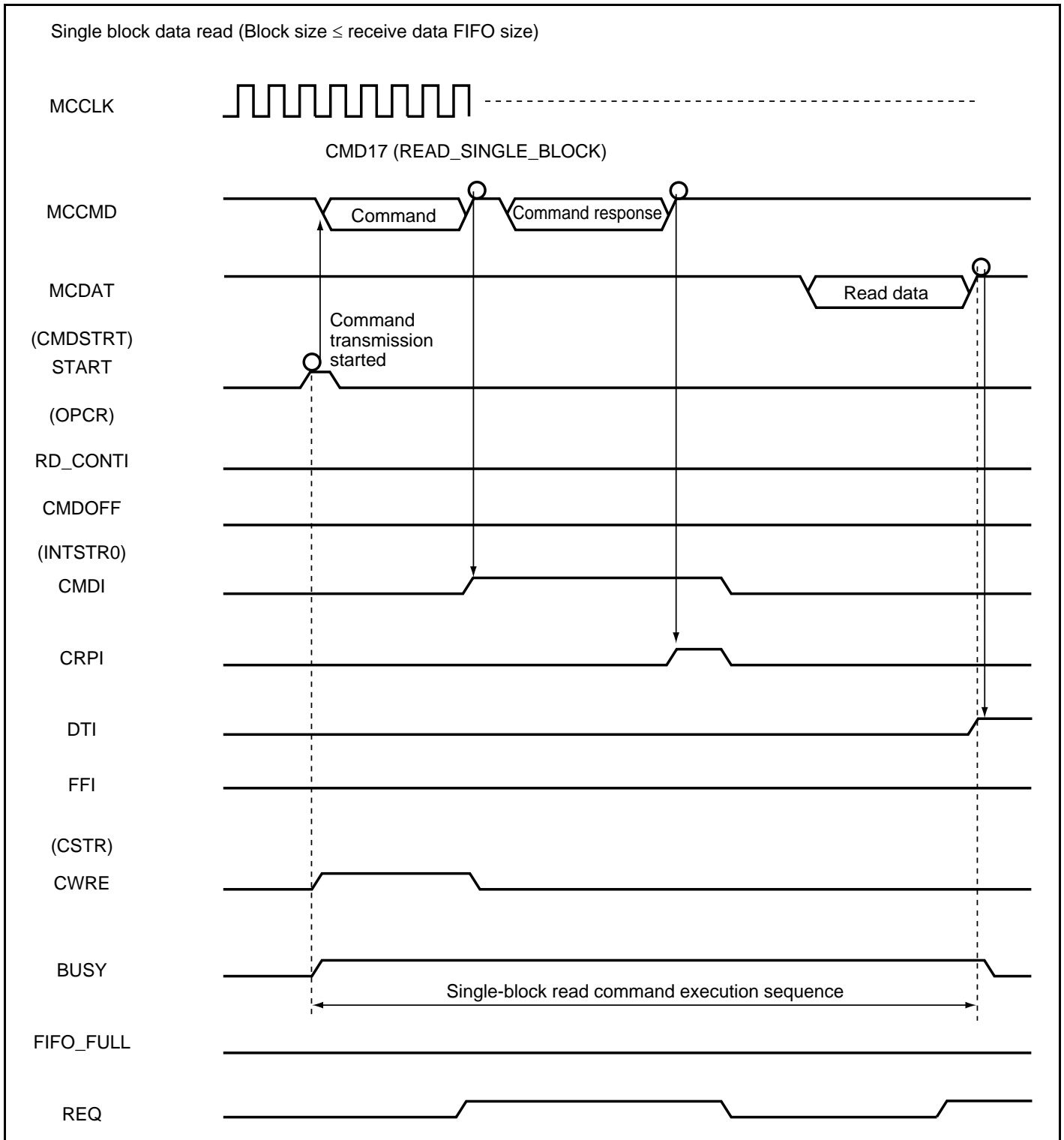
Commands involving read data confirm the MMC status by the command arguments and command responses, and then receive MMC information and flash memory data from the MCDAT pin.

The number of bytes of flash memory to be read is a block size specified by CMD16, or if not specified, reading is continued until it is aborted by CMD12M in multiblock transfer and stream transfer. For multiblock transfer, the transfer is suspended at the end of to wait for every block the instruction for continuing the command sequence.

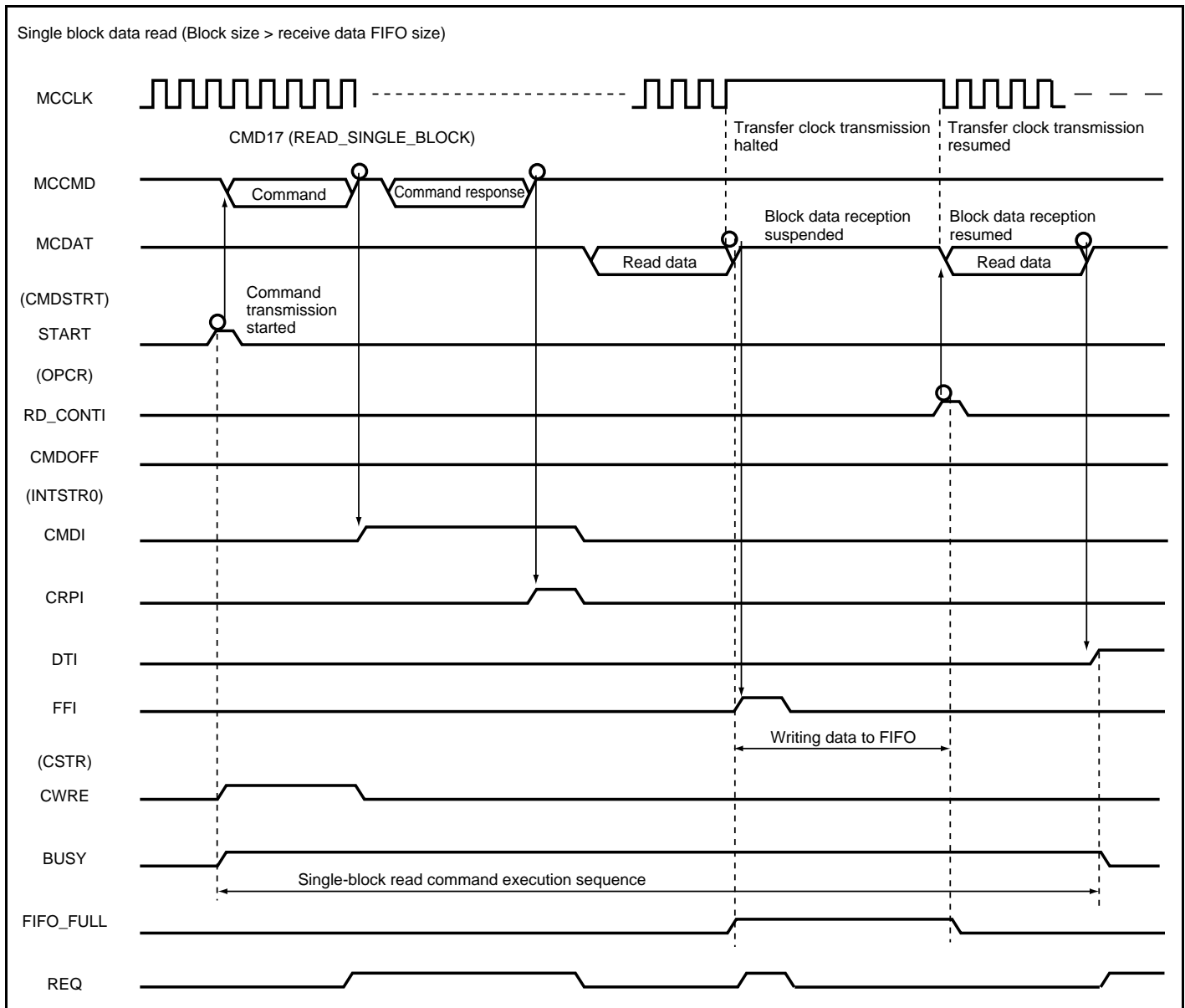
Whether the command sequence is suspended or not during the command sequence depends on the size of the block and receive data FIFO. The command sequence is executed without suspending the data transfer when block size  $\leq$  receive data FIFO size in single-block transfer. In multiblock transfer, the command sequence is suspended for every block. When block size  $>$  receive data FIFO size in single-block transfer, the command sequence is suspended by FIFO full. When the command sequence is suspended, data in the receive data FIFO is processed, and the command sequence is then continued.

Figures 19.7 to 19.10 show examples of the command sequence for commands with read data. Figure 19.11 shows the operational flow for commands with read data.

- Settings needed to issue a command are made. Receive data FIFO is cleared.
- The START bit in CMDSTRT is set to 1 to start command transmission. Command transmission complete can be confirmed by the command transmission end interrupt (CMDI).
- A command response is received from the MMC. If the MMC does not return the command response, the command response is detected by the command timeout error (CTERI).
- Read data from the MMC is detected.
- The suspension inter-blocks in multiblock transfer and suspension according to the receive data FIFO full are detected by the data transfer end interrupt (DTI) and FIFO full interrupt (FFI), respectively. To continue the command sequence, the RD\_CONTI bit in OPCR should be set to 1. To abort the command sequence, the CMDOFF bit in OPCR should be set to 1, and CMD12 should be issued.
- Detection of the end of a command sequence depends on the command types. In multiblock transfer of an extended command in SPI mode, the end of the command sequence is detected by the block transfer end interrupt (BTI) after the desired number of blocks has been received. In other commands, the end of the command sequence is detected by polling the BUSY flag in CSTR or by the data transfer end interrupt (DTI).



**Figure 19.7 Example of Command Sequence for Commands with Read Data (1)**



**Figure 19.8 Example of Command Sequence for Commands with Read Data (2)**

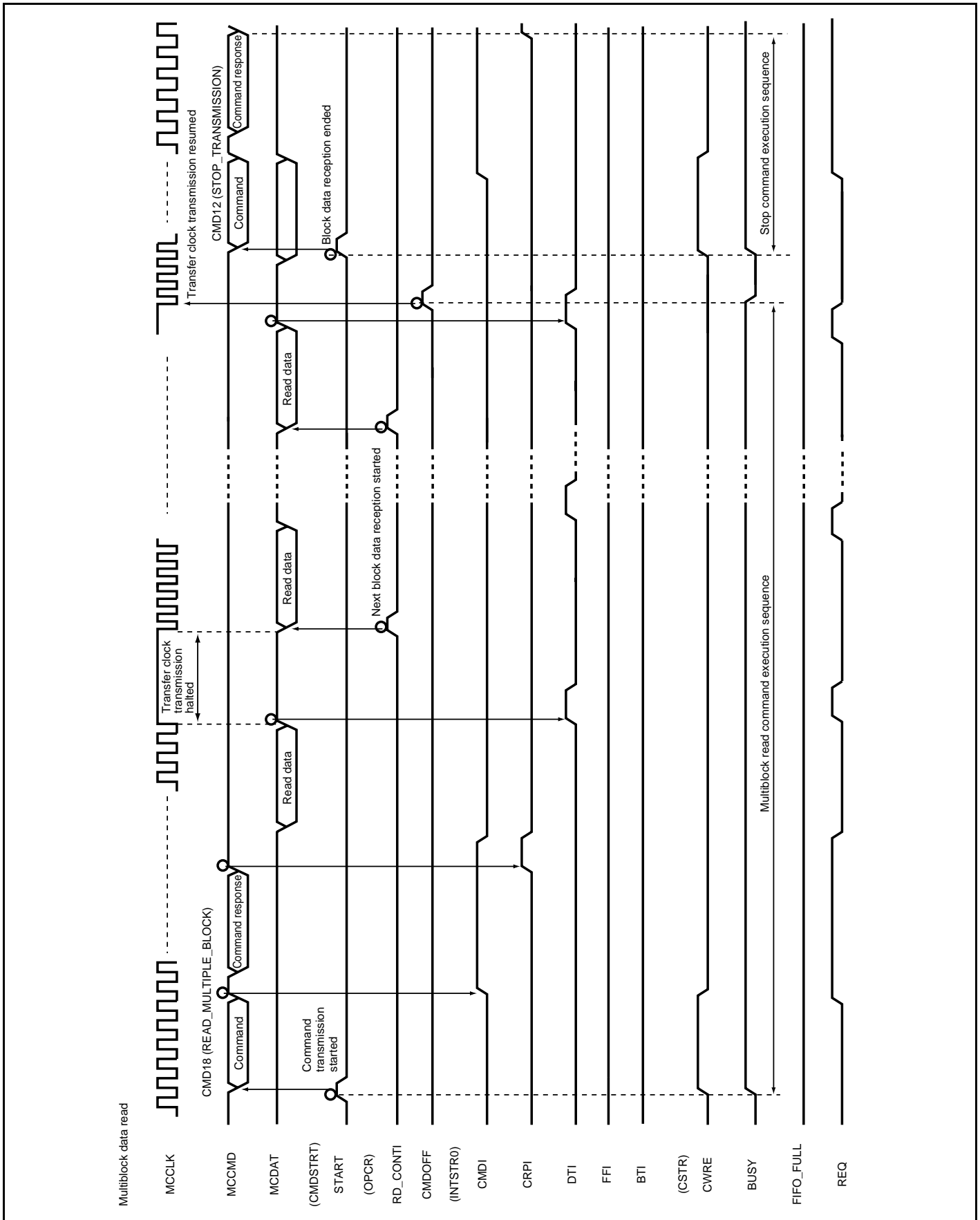


Figure 19.9 Example of Command Sequence for Commands with Read Data (3)

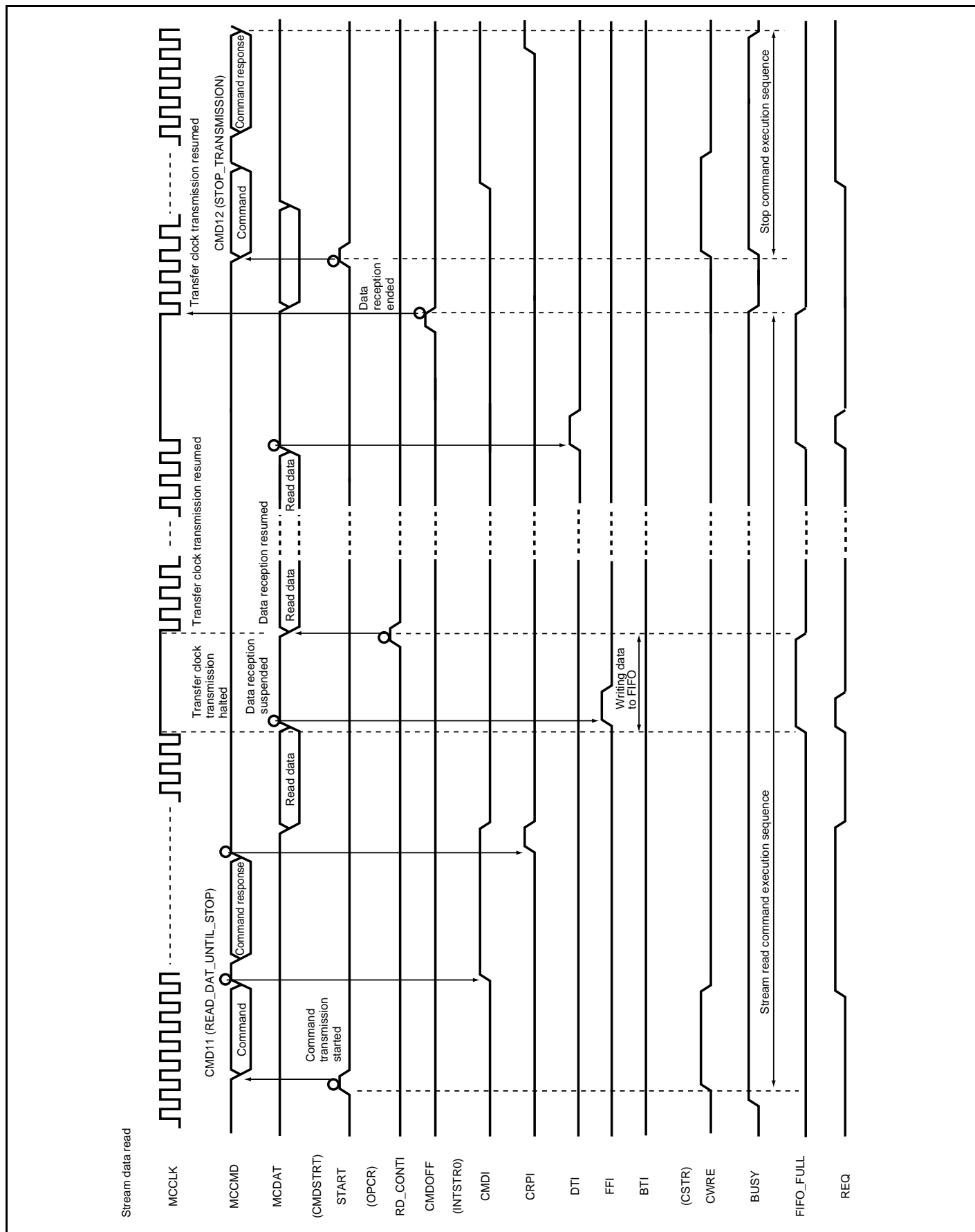


Figure 19.10 Example of Command Sequence for Commands with Read Data (4)

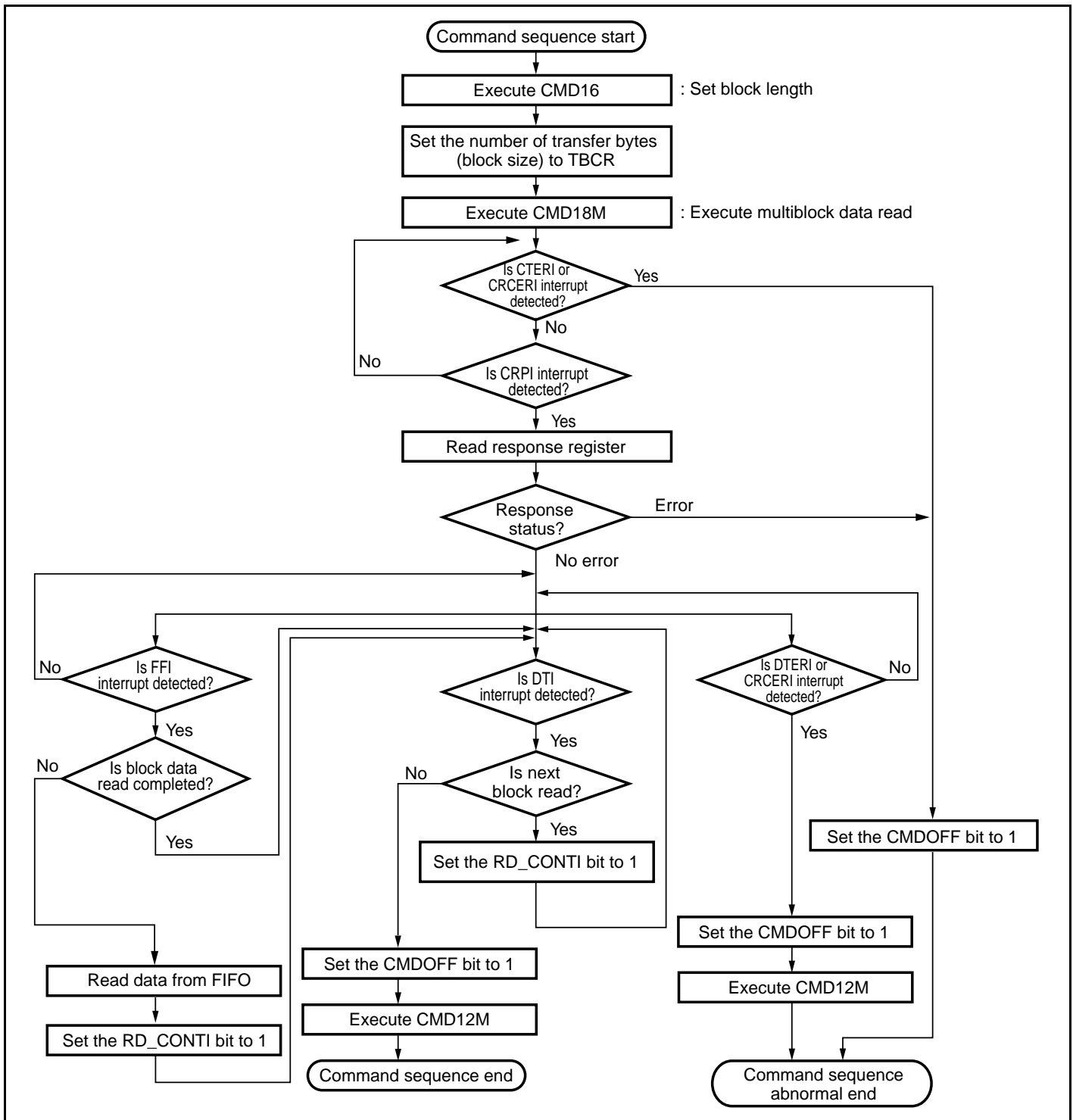


Figure 19.11 Operational Flow for Commands with Read Data



### 19.5.6 Commands with Write Data

Commands involving write data confirm the MMC status by the command responses after command transmission, and then transmit MMC information and flash memory data from the MCDAT pin. For a command that is related to time-consuming processing such as flash memory write, the MMC indicates the data busy state via the DAT pin.

The number of bytes of flash memory to be written is a block size specified by CMD16, or if not specified, writing is continued until it is aborted by CMD12M in multiblock transfer and stream transfer. For multiblock transfer, the instruction for continuing the command sequence is made by suspending the transfer for every block.

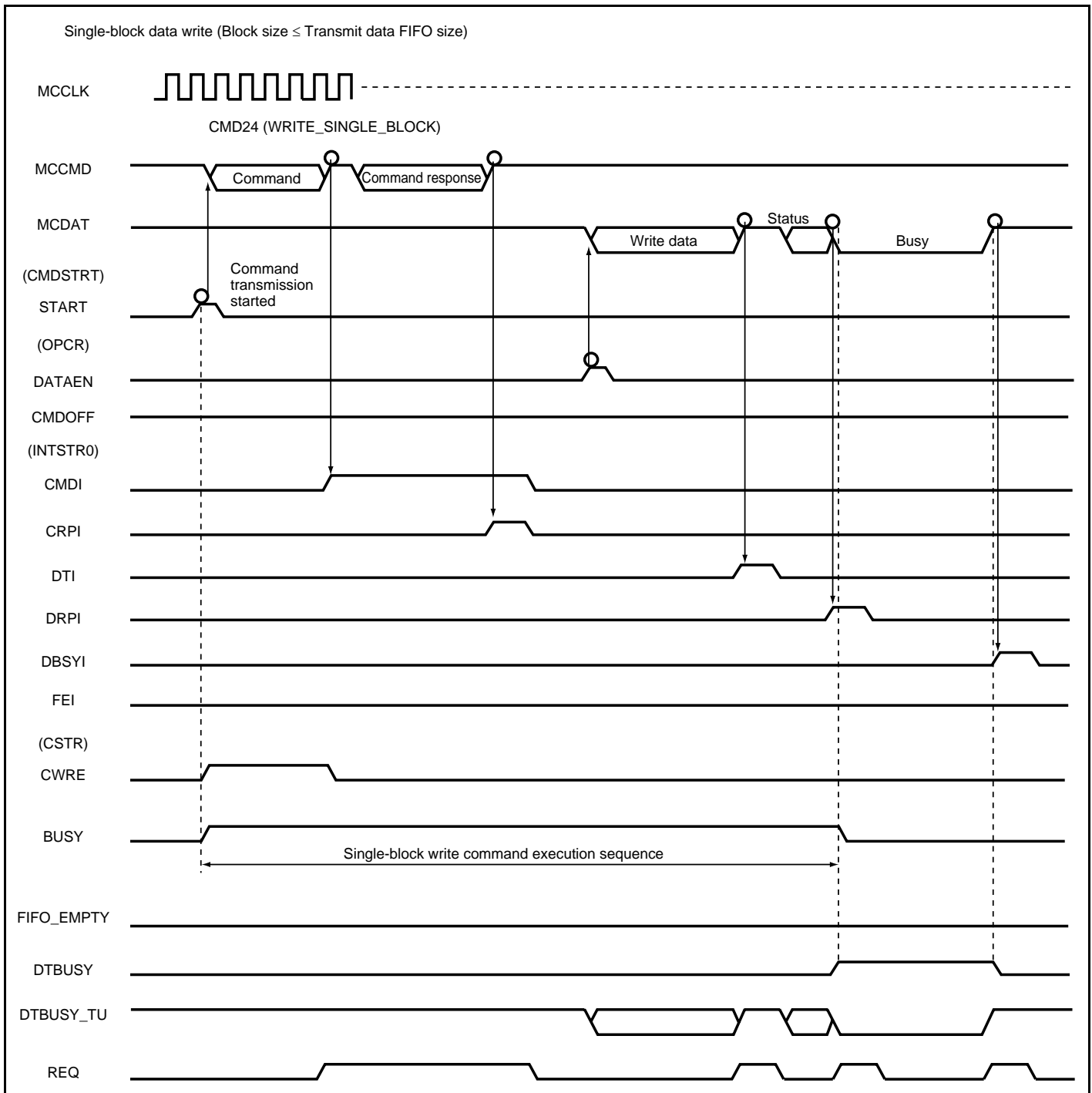
The suspension of the command sequence depends on the sizes of the block and transmit data FIFO. The command sequence is executed without suspending the data transfer when block size  $\leq$  transmit data FIFO size in single-block transfer. In multiblock transfer, the command sequence is suspended for every block. When block size  $>$  transmit data FIFO size, the command sequence is suspended by FIFO empty. When the command sequence is suspended, the next data is written to the transmit data FIFO, and the command sequence is then continued.

Figures 19.12 to 19.15 show examples of the command sequence for commands with write data. Figure 19.16 shows the operational flow for commands with write data.

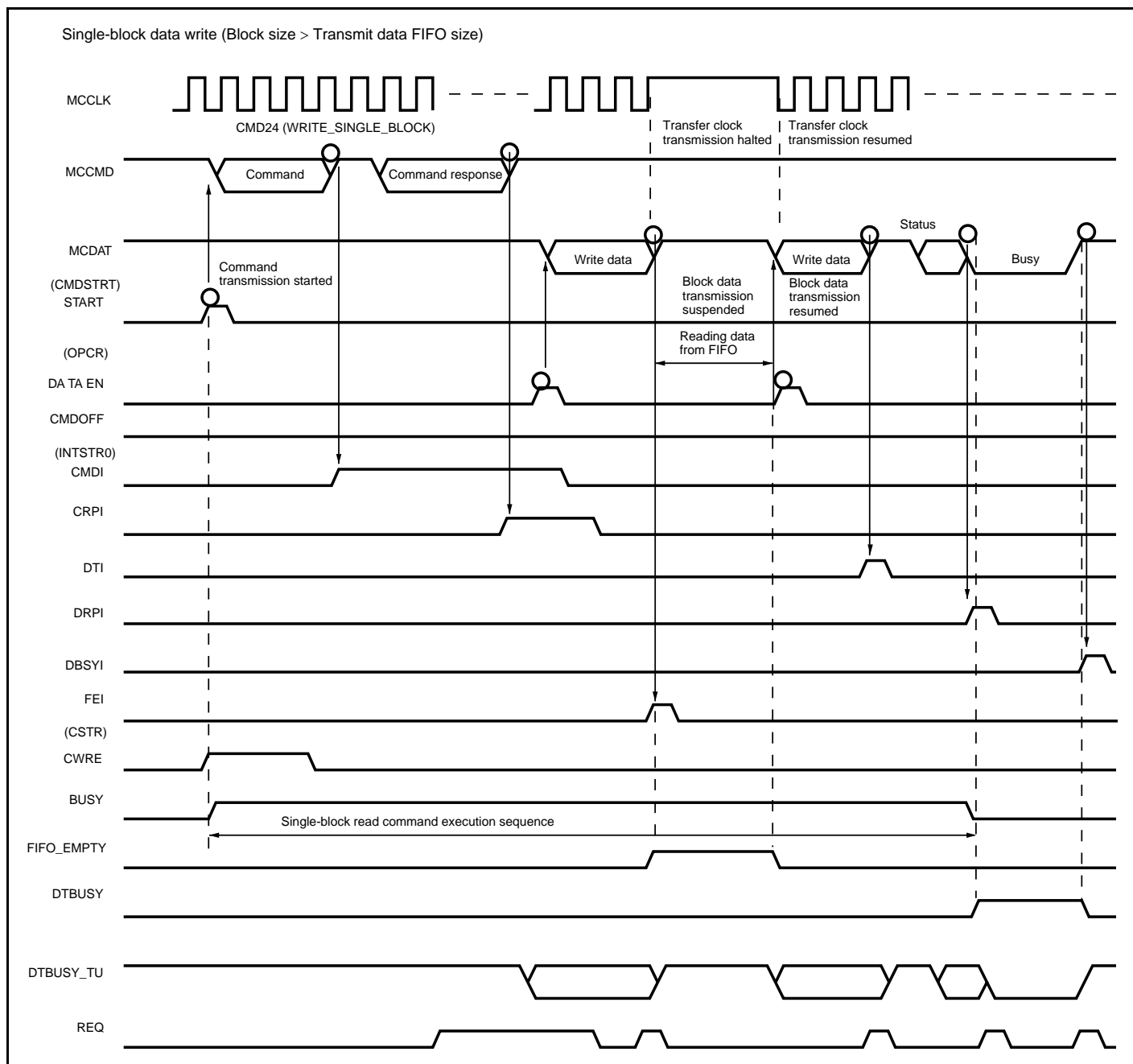
- Settings needed to issue a command are made. Write data is set to the transmit data FIFO.
- The START bit in CMDSTRT is set to 1 to start command transmission. Command transmission complete can be confirmed by the command transmission end interrupt (CMDI).
- A command response is received from the MMC. If the MMC does not return the command response, the command response is detected by the command timeout error (CTERI).
- The DATAEN bit in OPCR is set to start write data transmission.
- Suspension inter-blocks in multiblock transfer and suspension according to the transmit data FIFO empty are detected by the data transfer end interrupt (DTI), data response end interrupt (DRPI), and FIFO empty interrupt (FEI), respectively. In addition, when the command sequence is suspended by the end of data transfer, cancellation of the data busy status is detected by the data busy end interrupt (DBSYI). To continue the command sequence, data should be written to the transmit data FIFO, and the DATAEN bit in OPCR should be set to 1. To abort the command sequence, the CMDOFF bit in OPCR should be set to 1, and CMD12M should be issued.
- Detection of the end of a command sequence depends on the command types. In multiblock transfer of an extended command in SPI mode, the end of the command sequence is detected by the block transfer end interrupt (BTI) or data response end interrupt (DRPI) after the desired number of blocks has been transmitted. In other commands, the end of the command

sequence is detected by polling the BUSY flag in CSTR, the data transfer end interrupt (DTI), or the data response end interrupt (DPRI).

- The end of the data busy state is detected by polling the DTBUSY flag in CSTR or by the data busy end interrupt (DBSYI).



**Figure 19.12 Example of Command Sequence for Commands with Write Data (1)**



**Figure 19.13 Example of Command Sequence for Commands with Write Data (2)**

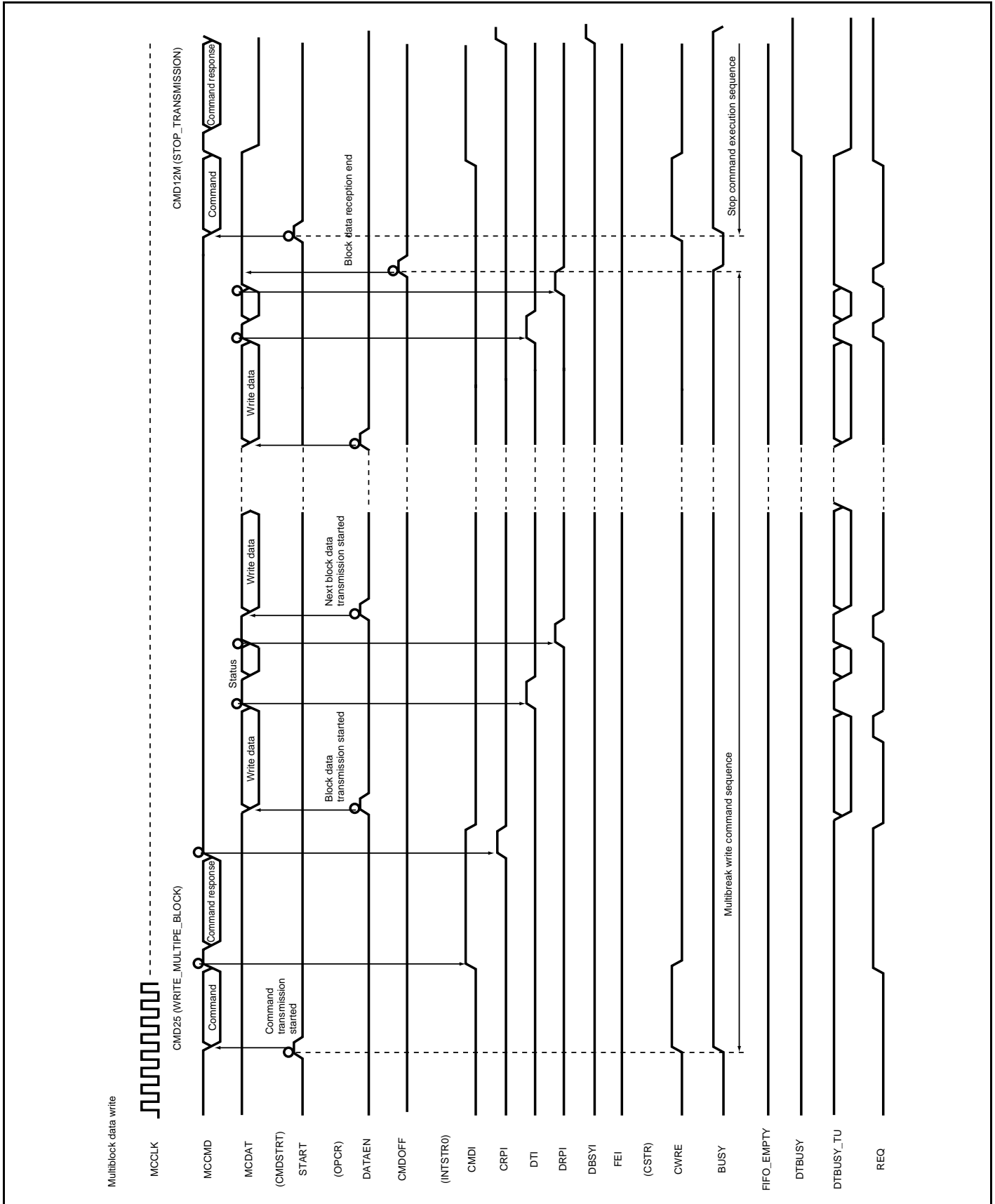


Figure 19.14 Example of Command Sequence for Commands with Write Data (3)

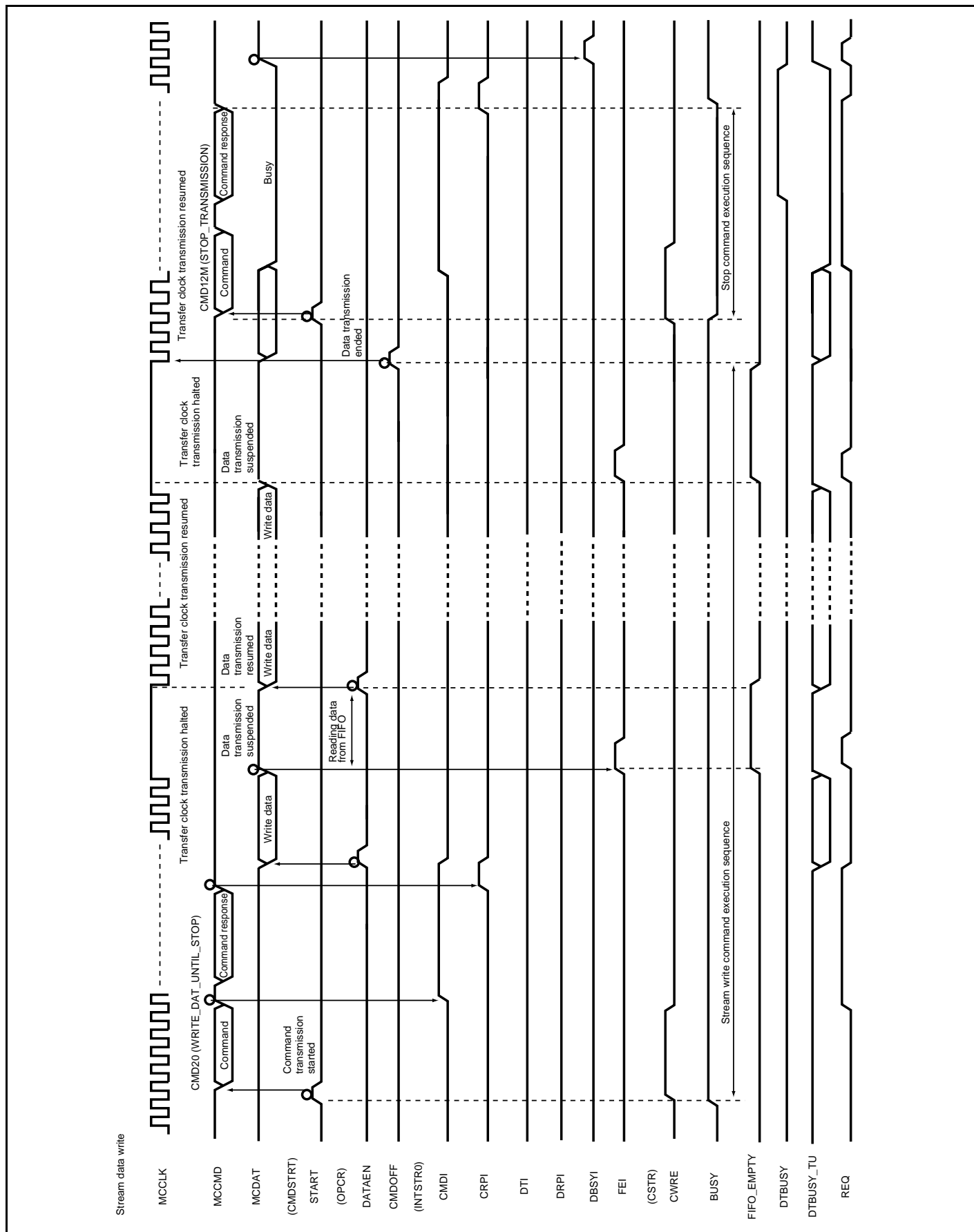
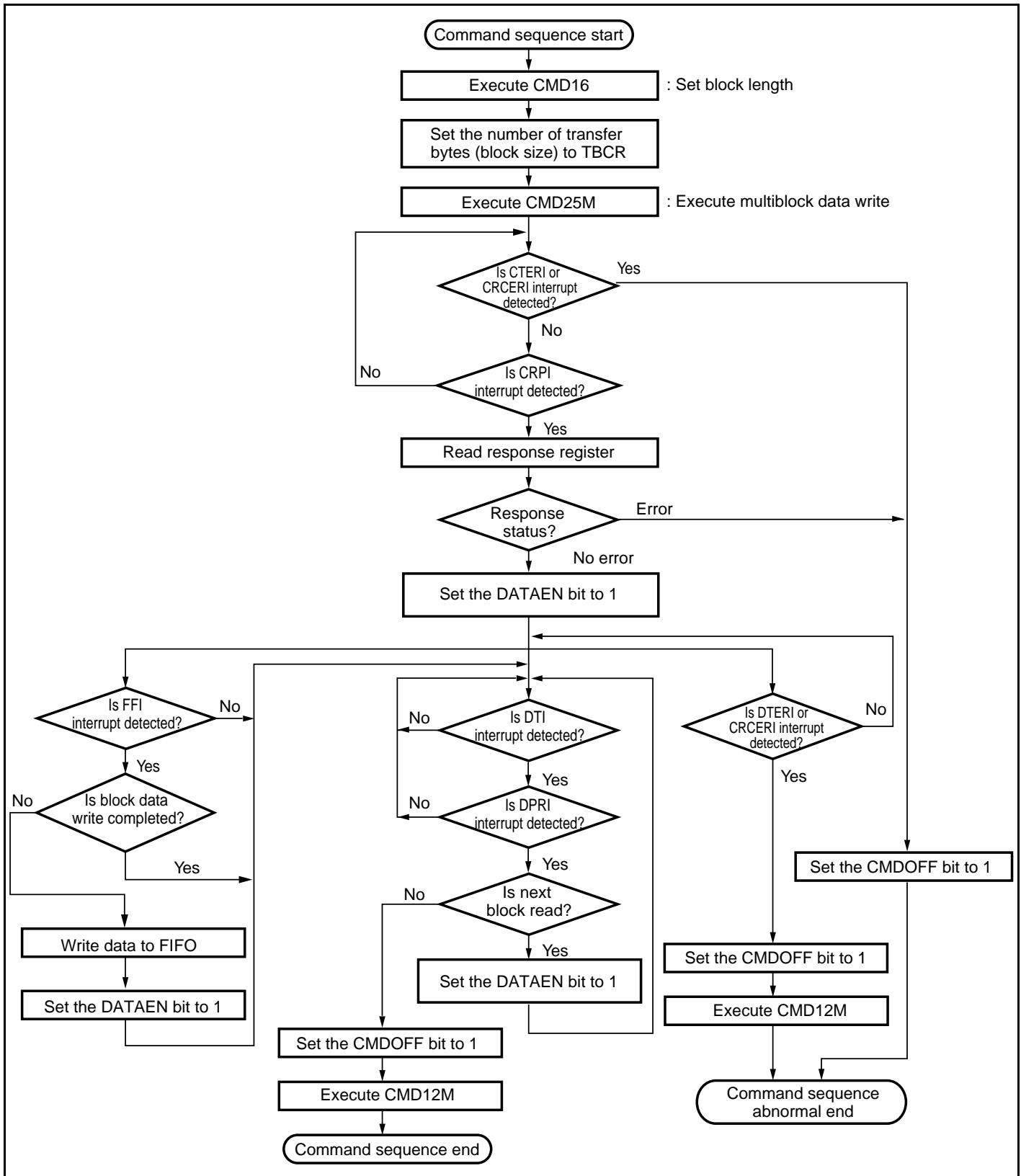


Figure 19.15 Example of Command Sequence for Commands with Write Data (4)



**Figure 19.16 Operational Flow for Commands with Write Data**

## 19.6 Operations in SPI Mode

SPI mode is an operating mode in which the transfer clock is output from the MCCLK pin, and command/response/data is input/output via the MCRxD pin and MCTxD pin.

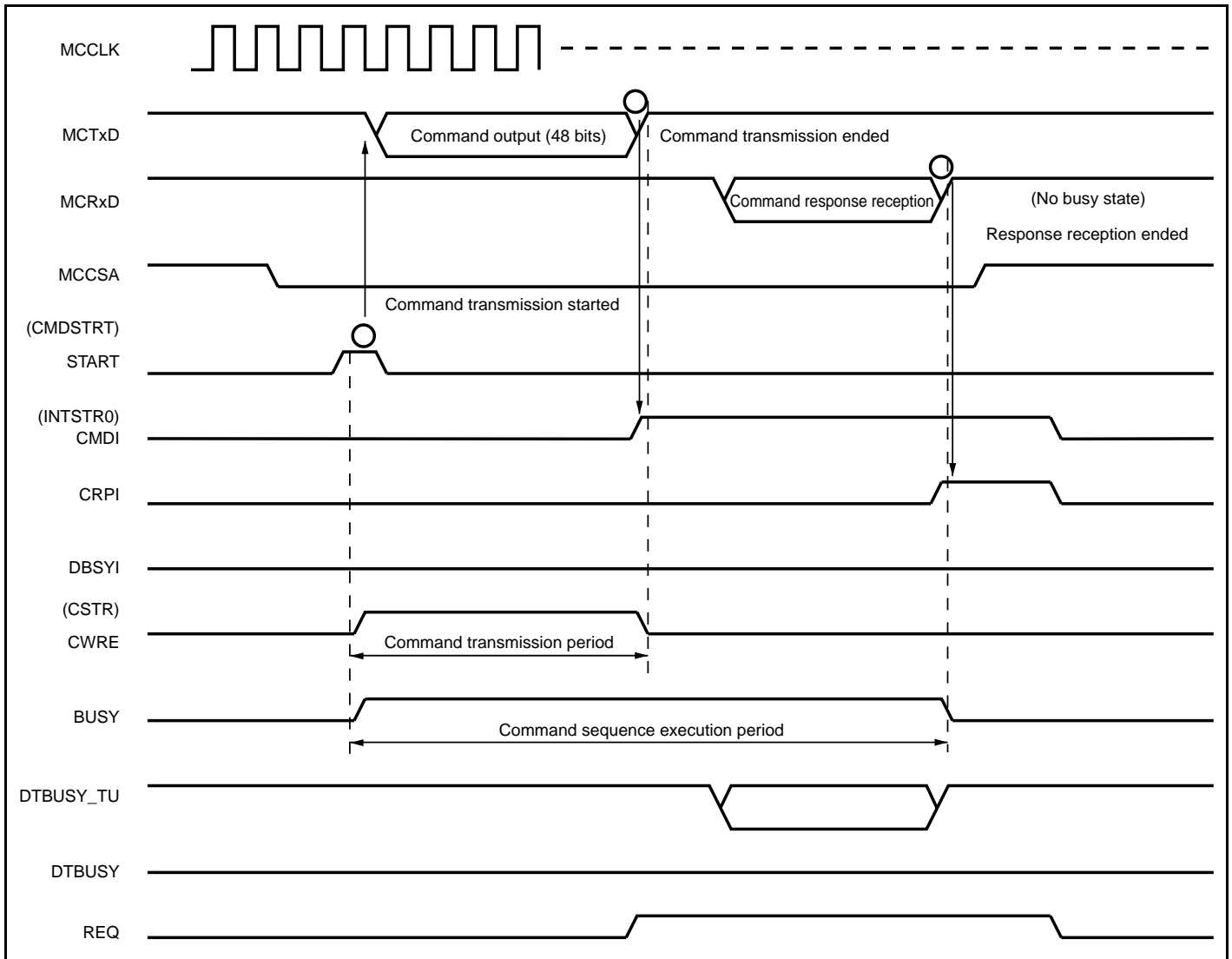
In SPI mode, one of multiple MMCs is selected by the chip select (CS) pin. Therefore, card selection using broadcast commands for MMC mode is not supported. In SPI mode, data response to write data is supported.

### 19.6.1 Operation of Commands without Data Transfer

Commands without data transfer execute the desired data transfer using command arguments and command responses. For a command that is related to time-consuming processing such as flash memory write/erase, the MMC displays the data busy state.

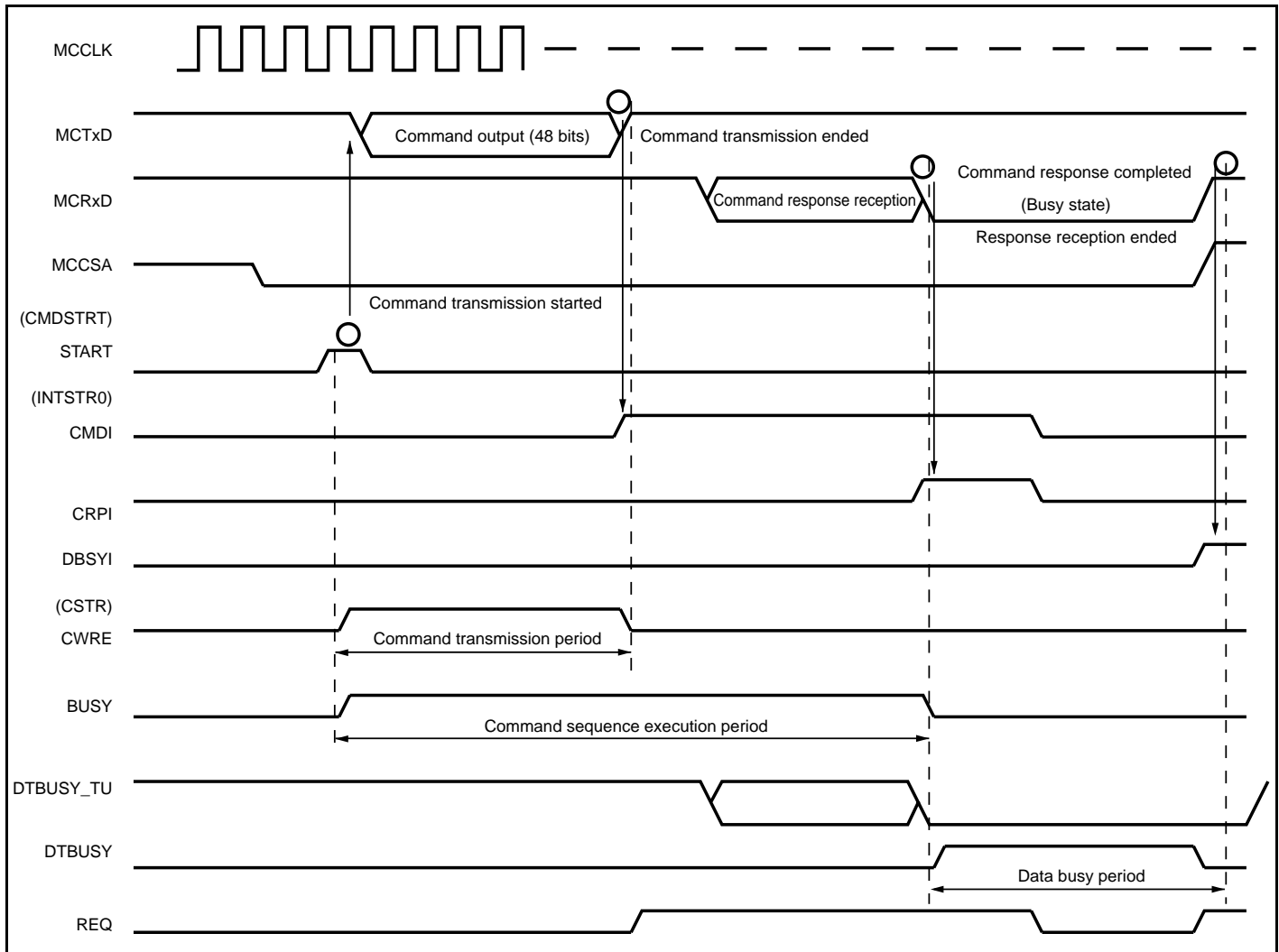
Figures 19.17 and 19.18 show examples of the command sequence for commands without data transfer. Figure 19.19 shows the operational flow for commands without data transfer.

- Settings needed to issue a command are made.
- The START bit in CMDSTRT is set to 1 to start command transmission. The CS signal goes low (select). Command transmission complete can be confirmed by the command transmission end interrupt (CMDI).
- A command response is received from the MMC. If the MMC does not return the command response, the command response is detected by the command timeout error (CTERI).
- When the command sequence ends, the CS signal goes high (not select). The end of the command sequence is detected by poling the BUSY flag in CSTR or by the command output end interrupt (CRPI).
- The end of the data busy state is detected by poling the DTBUSY flag in CSTR or by the data busy end interrupt (DBSYI).

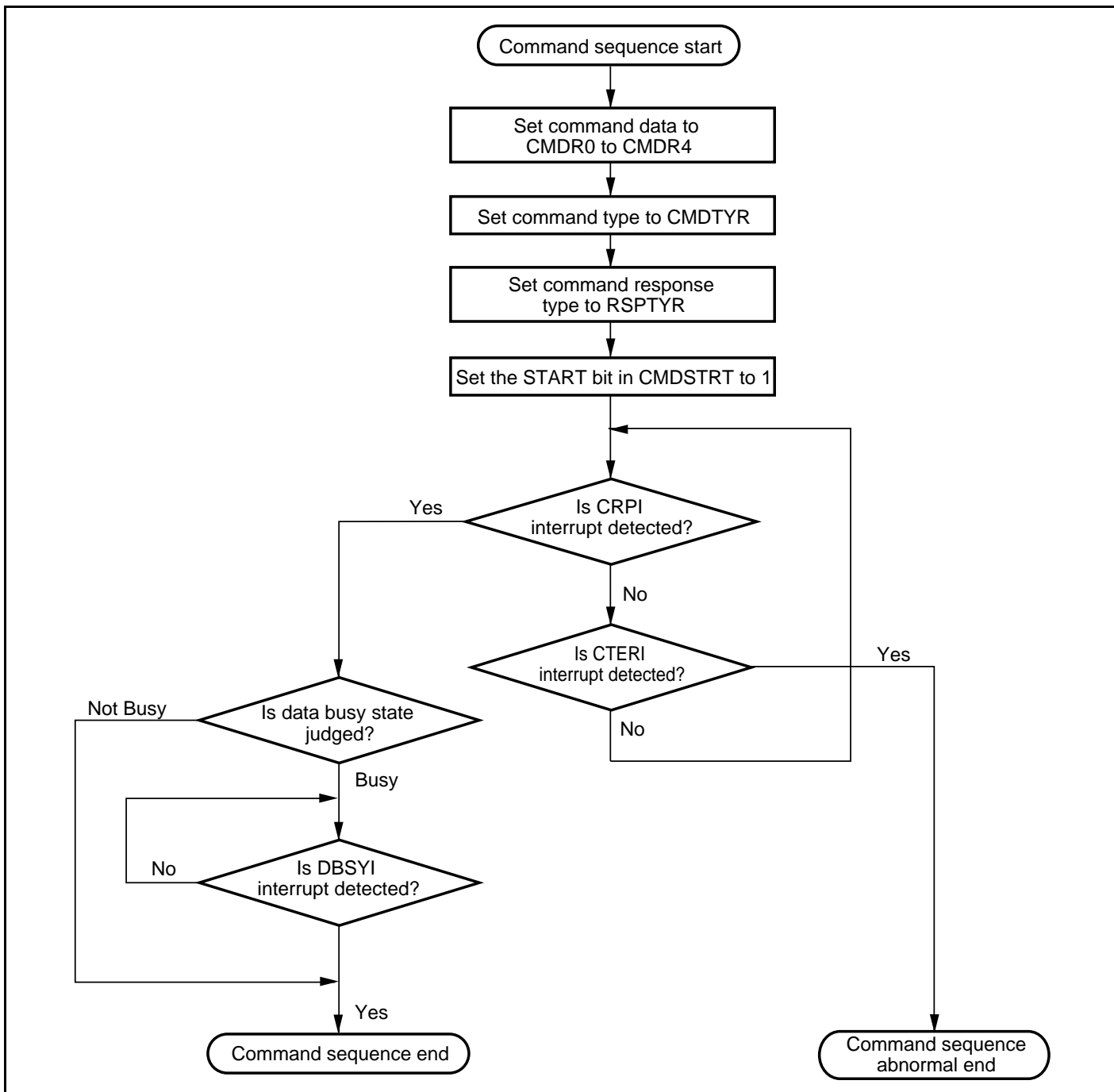


**Figure 19.17 Example of Command Sequence for Commands without Data Transfer (No Data Busy State)**





**Figure 19.18 Example of Command Sequence for Commands without Data Transfer (with Data Busy State)**



**Figure 19.19 Operational Flow for Commands without Data Transfer**

## 19.6.2 Commands with Read Data

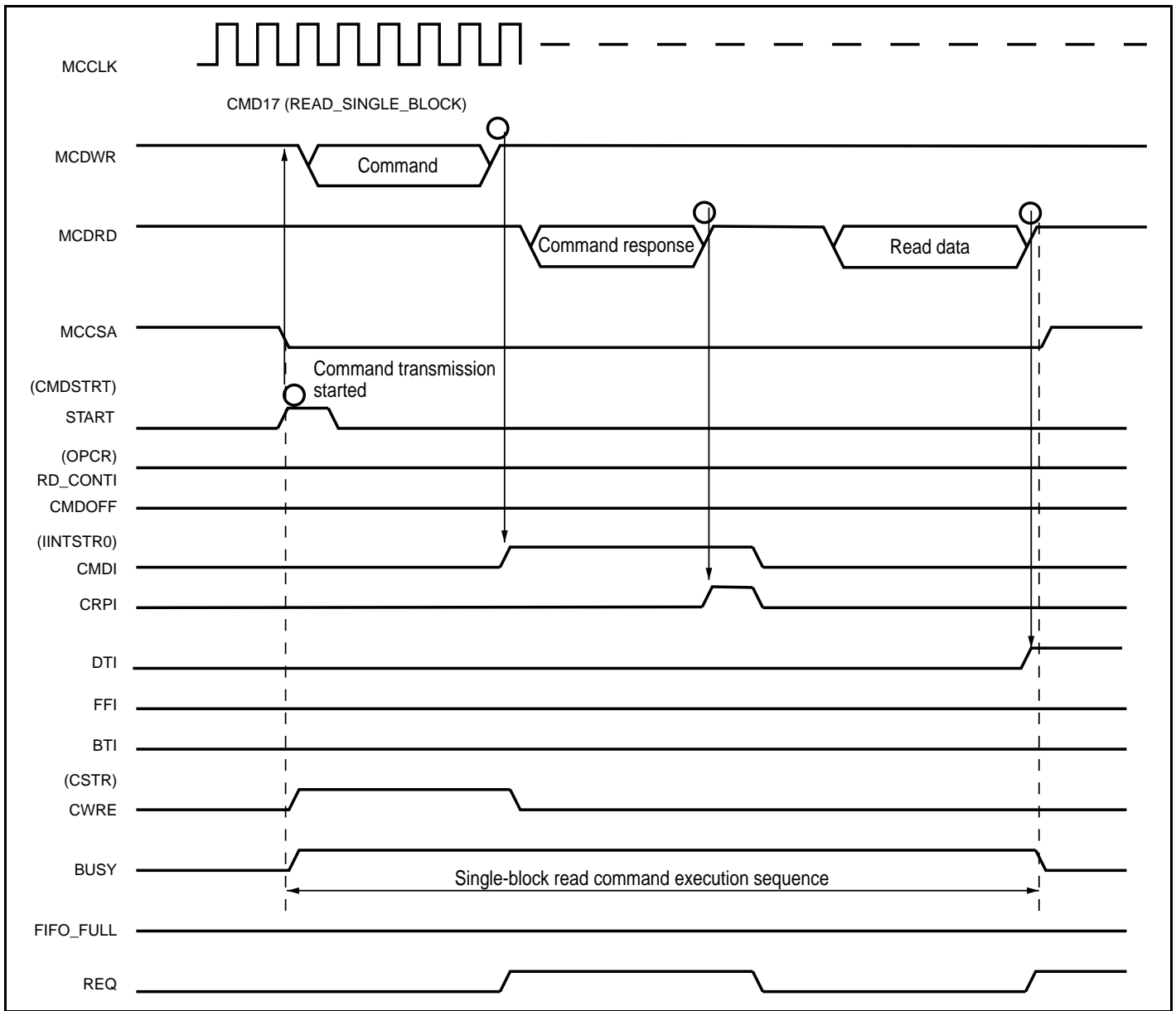
Commands with read data confirm the MMC status by the command responses, and then receive MMC information and flash memory data.

The number of bytes of flash memory to be read is a block size specified by CMD16.

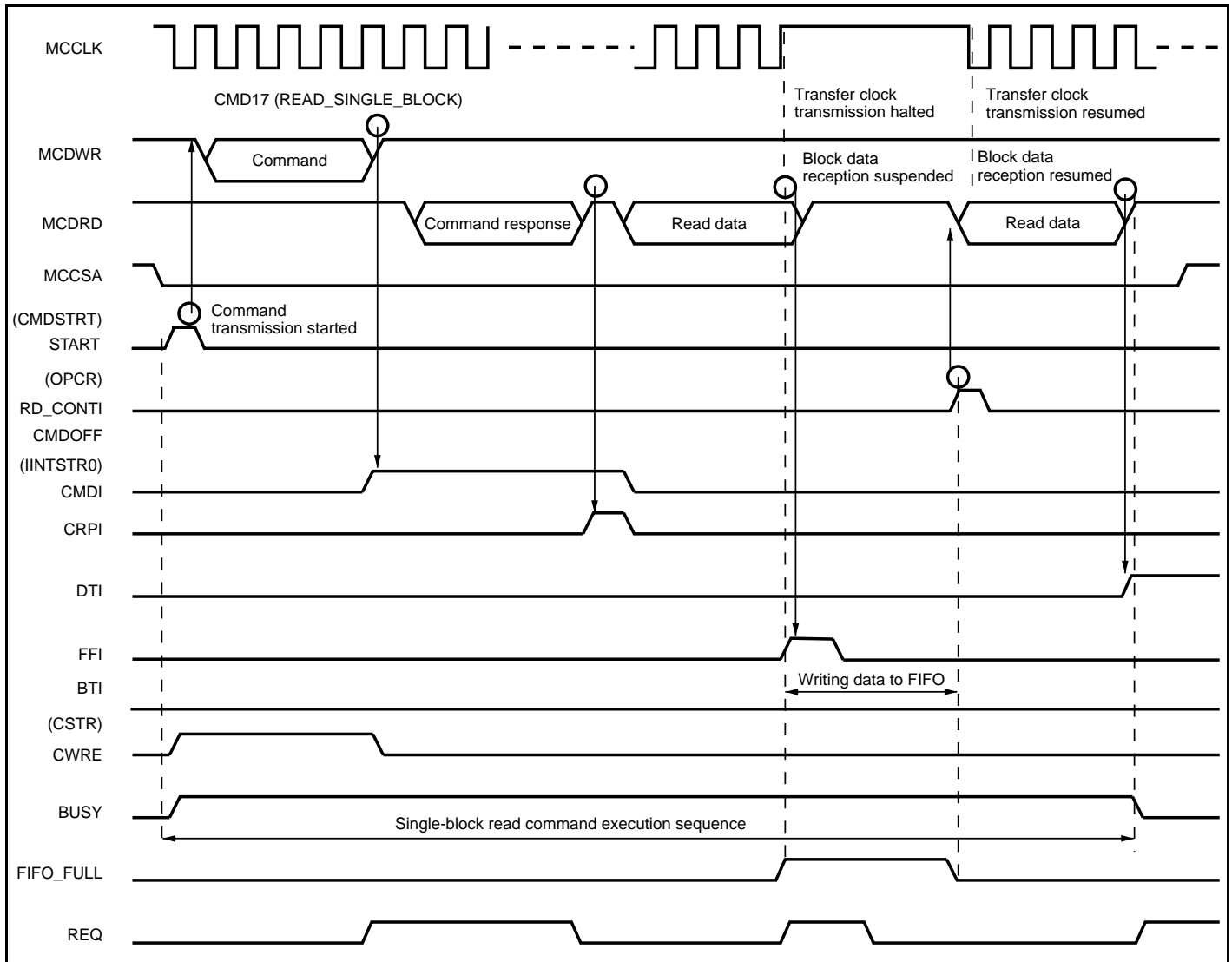
When block size > receive data FIFO size, the command sequence is suspended by FIFO full. When the command sequence is suspended, data in the receive data FIFO is processed, and the command sequence is then continued.

Figures 19.20 and 19.21 show examples of the command sequence for commands with read data. Figure 19.22 shows the operational flow for commands with read data.

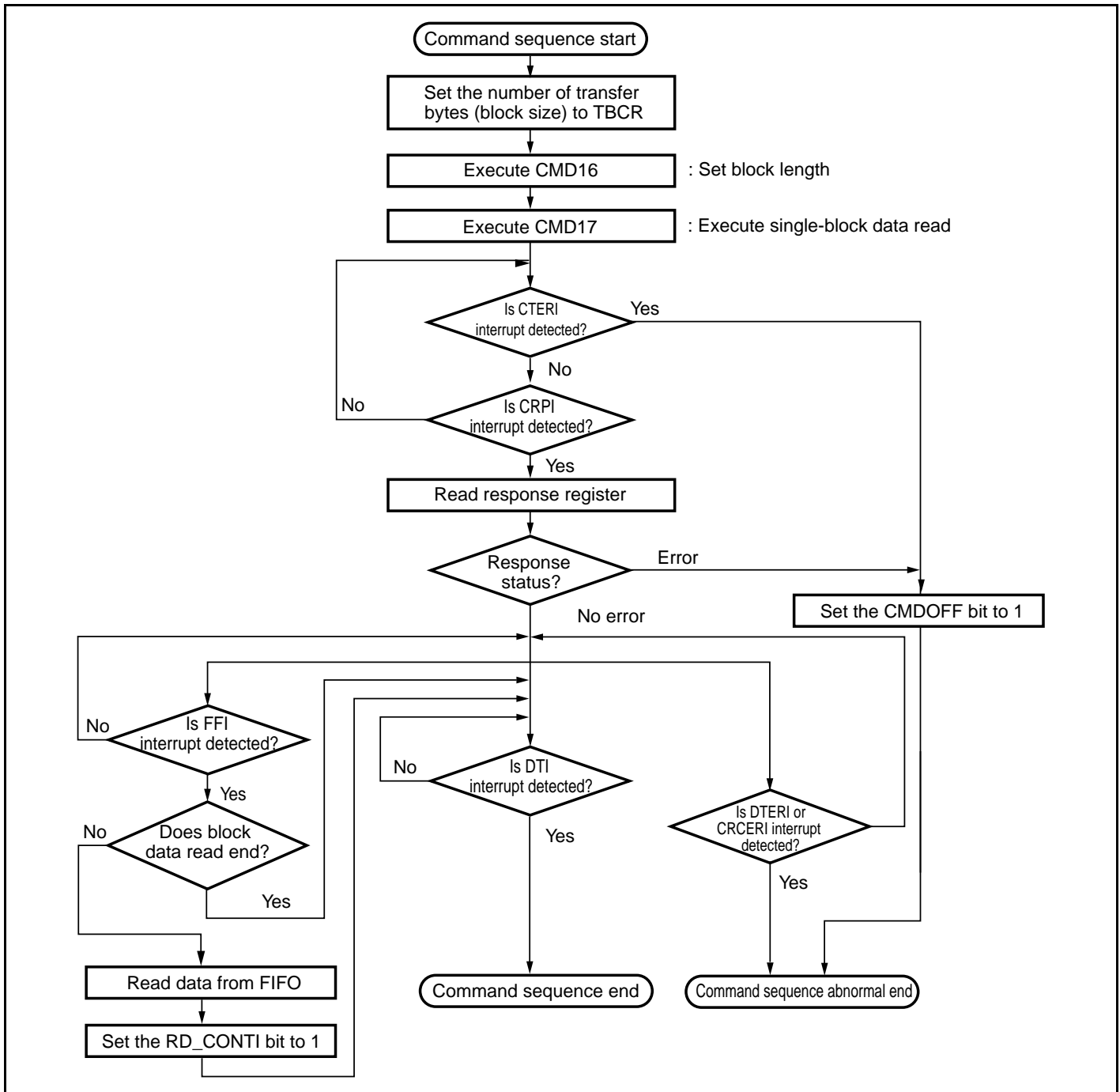
- Settings needed to issue a command are made.
- The START bit in CMDSTRT is set to 1 to start command transmission. The CS signal goes low (select). Command transmission complete can be confirmed by the command transmission end interrupt (CMDI).
- The command response is received from the MMC. If the MMC does not return the command response, the command response is detected by the command timeout error (CTERI).
- Read data is received from the MMC.
- Command abortion is detected according to the receive data FIFO full by the FIFO full interrupt (FFI). To continue the command sequence, the RD\_CONTI bit in OCR should be set to 1.
- When the command sequence ends, the CS signal goes high (not select). The end of the command sequence is detected by polling the BUSY flag in CSTR or by the data transfer end interrupt (DTI).



**Figure 19.20 Example of Command Sequence for Commands with Read Data (1)**



**Figure 19.21 Example of Command Sequence for Commands with Read Data (2)**



**Figure 19.22 Operational Flow for Commands with Read Data**

### 19.6.3 Commands with Write Data

Commands with write data confirm the MMC status by the command responses, and then transmit MMC information and flash memory data. For a command that is related to time-consuming processing such as flash memory write, the MMC indicates the data busy state.

The number of bytes of flash memory to be written is a block size specified by CMD16.

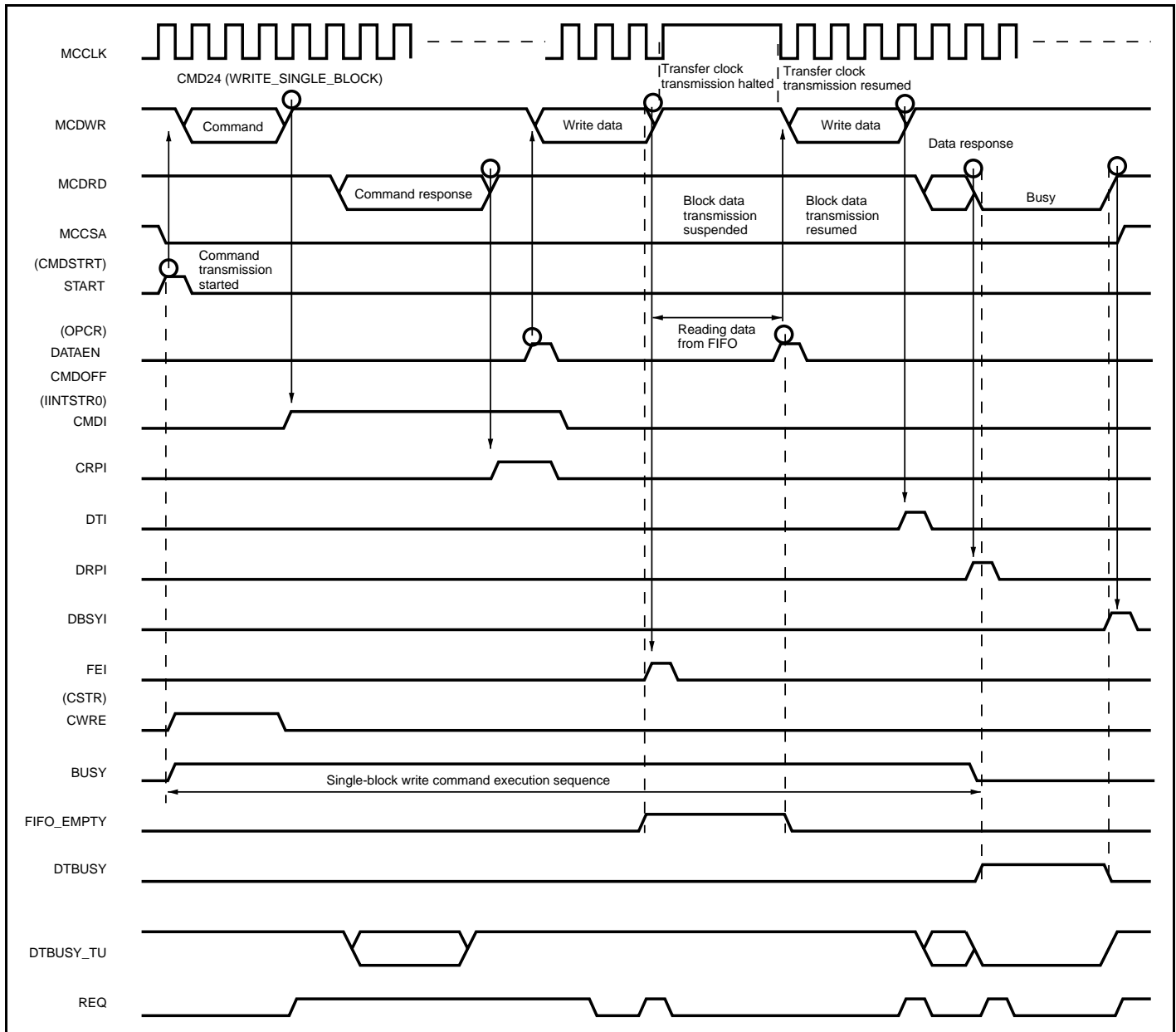
When block size > transmit data FIFO size, the command sequence is suspended by FIFO empty. When the command sequence is suspended, the next data is written to the transmit data FIFO, and the command sequence is then continued.

Figures 19.23 and 19.24 show examples of the command sequence for commands with write data. Figure 19.25 shows the operational flow for commands with write data.

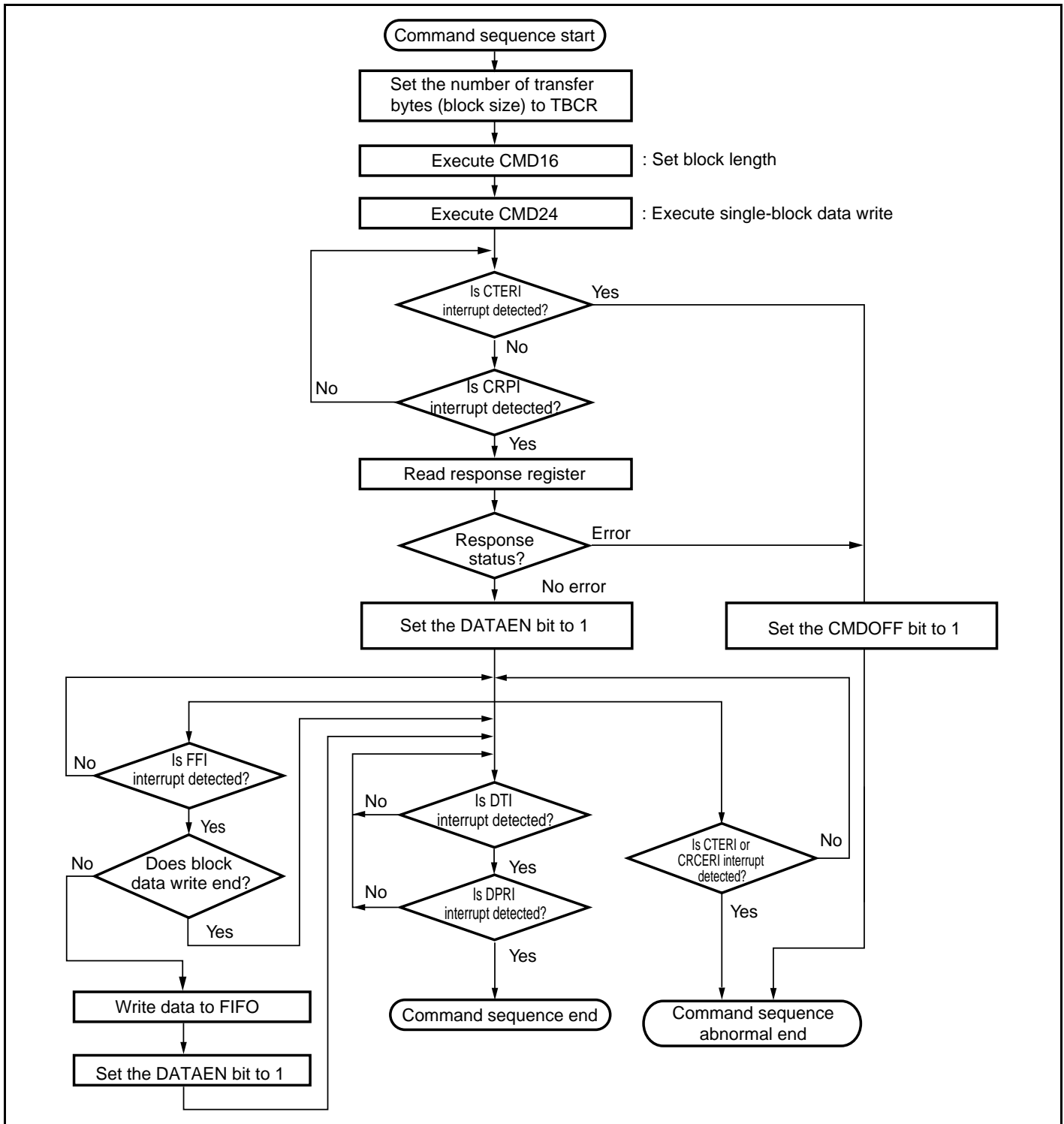
- Settings needed to issue a command are made. Write data is set to the transmit data FIFO.
- The START bit in CMDSTRT is set to 1 to start command transmission. The CS signal goes low (select). Command transmission complete can be confirmed by the command transmission end interrupt (CMDI).
- A command response is received from the MMC. If the MMC does not return the command response, the command response is detected by the command timeout error (CTERI).
- The DATAEN bit in OPCR is set to 1 to start write data transmission.
- Suspension according to the transmit data FIFO empty is detected by the FIFO empty interrupt (FEI). Data is written to the transmit data FIFO, and the DATAEN bit in OPCR is set to 1.
- The end of the command sequence is detected by polling the BUSY flag in CSTR or the data transfer end interrupt (DTI). Reception of data response can be confirmed by the data response end interrupt (DPRI).
- When the command sequence ends, the CS signal goes high (not select). The end of the data busy state is detected by polling the DTBUSY flag in CSTR or by the data busy end interrupt (DBSYI).







**Figure 19.24 Example of Command Sequence for Commands with Write Data (2)**



**Figure 19.25 Operational Flow for Commands with Write Data**

## 19.7 Interrupt Sources

Table 19.6 lists the MCIF interrupt sources. The interrupt sources are classified into three groups, each to which an interrupt vector is assigned. Each interrupt source can be individually enabled by the enable bits in INTCR0 and INTCR1. A disabled interrupt source does not set the flag.

The MMCIA interrupts can be used as DTC activation interrupt sources.

**Table 19.6 MCIF Interrupt Sources**

Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority	
MMCIA	MCIF10	FIFO empty	FEI	Possible	High
		FIFO full	FFI	Possible	
MMCIB	MCIF11	Data response	DPRI	Not possible	↑
		Data transfer end	DTI	Not possible	
		Command response end	CRPI	Not possible	
		Command transmission end	CMDI	Not possible	
		Data busy end	DBSYI	Not possible	
		Block transfer end	BTI	Not possible	
MMCIC	MCIF12	CRC error	CRCERI	Not possible	↓
		Data timeout error	DTERI	Not possible	
		Command timeout error	CTERI	Not possible	

## 19.8 Usage Notes

1. When activating the DTC with an MCIF interrupt source, correct operation is not guaranteed if the DISEL bit in DTC mode register B (MRB) is cleared to 0. Be sure to set the DISEL bit to 1 before DTC activation.
2. MCIF operation can be enabled or disabled by the SMSTPB6 bit in subchip module stop control register BL (SUBMSTPBL). In the initial state, MCIF operation is enabled. The MCIF registers can be accessed by clearing module stop mode.



## Section 20 Encryption Operation Circuit (DES and GF)

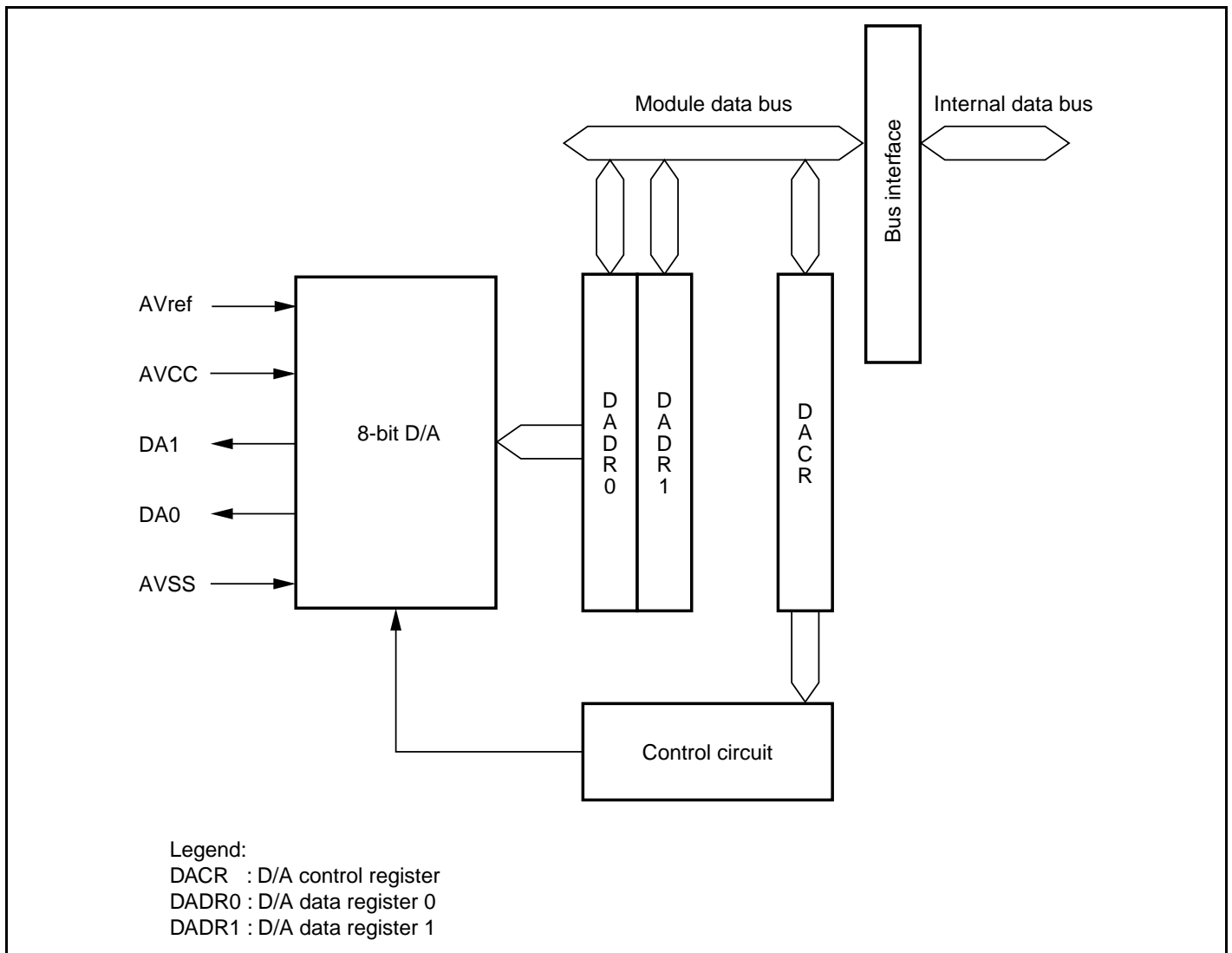
**This section will be made available on conclusion of a nondisclosure agreement.  
For details, contact your Renesas sales agency.**



## Section 21 D/A Converter

### 21.1 Features

- 8-bit resolution
- Two output channels
- Conversion time: Max. 10  $\mu\text{s}$  (when load capacitance is 20 pF)
- Output voltage: 0 V to  $AV_{\text{ref}}$
- D/A output retaining function in software standby mode



**Figure 21.1 Block Diagram of D/A Converter**

## 21.2 Input/Output Pins

Table 21.1 summarizes the input/output pins used by the D/A converter.

**Table 21.1 Pin Configuration**

<b>Pin Name</b>	<b>Symbol</b>	<b>I/O</b>	<b>Function</b>
Analog power supply pin	AVCC	Input	Analog block power supply
Analog ground pin	AVSS	Input	Analog block ground and reference voltage
Analog output pin 0	DA0	Output	Channel 0 analog output
Analog output pin 1	DA1	Output	Channel 1 analog output
Reference power supply pin	AVref	Input	Analog block reference voltage

## 21.3 Register Descriptions

The D/A converter has the following registers.

- D/A data register 0 (DADR0)
- D/A data register 1 (DADR1)
- D/A control register (DACR)

### 21.3.1 D/A Data Registers 0 and 1 (DADR0, DADR1)

DADR0 and DADR1 are 8-bit readable/writable registers that store data for D/A conversion. When analog output is permitted, D/A data register contents are converted and output to analog output pins.

### 21.3.2 D/A Control Register (DACR)

DACR controls D/A converter operation.



Bit	Bit Name	Initial Value	R/W	Description
7	DAOE1	0	R/W	D/A Output Enable 1 Controls D/A conversion and analog output. 0: Analog output DA1 is disabled 1: D/A conversion for channel 1 and analog output DA1 are enabled
6	DAOE0	0	R/W	D/A Output Enable 0 Controls D/A conversion and analog output. 0: Analog output DA0 is disabled 1: D/A conversion for channel 0 and analog output DA0 are enabled
5	DAE	0	R/W	D/A Enable Controls D/A conversion in conjunction with the DAOE0 and DAOE1 bits. When the DAE bit is cleared to 0, D/A conversion for channels 0 and 1 are controlled individually. When the DAE bit is set to 1, D/A conversion for channels 0 and 1 are controlled as one. Conversion result output is controlled by the DAOE0 and DAOE1 bits. For details, see table 21.2 below.
4 to 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

Table 21.2 D/A Channel Enable

Bit 7	Bit 6	Bit 5	Description
DAOE1	DAOE0	DAE	
0	0	X	Disables D/A conversion
	1	0	Enables D/A conversion for channel 0 Disables D/A conversion for channel 1
		1	Enables D/A conversion for channels 0 and 1
1	0	0	Disables D/A conversion for channel 0 Enables D/A conversion for channel 1
		1	Enables D/A conversion for channels 0 and 1
	1	X	Enables D/A conversion for channels 0 and 1

Legend:

X: Don't care

## 21.4 Operation

The D/A converter incorporates two channels of the D/A circuits and can be converted individually.

When the DAOE bit in DACR is set to 1, D/A conversion is enabled and conversion results are output.

An example of D/A conversion of channel 0 is shown below. The operation timing is shown in figure 21.2.

1. Write conversion data to DADR0.
2. When the DAOE0 bit in DACR is set to 1, D/A conversion starts. After the interval of  $t_{\text{DCONV}}$ , conversion results are output from the analog output pin DA0. The conversion results are output continuously until DADR0 is modified or the DAOE0 bit is cleared to 0. The output value is calculated by the following formula:

$$\text{DADR0 contents}/256 \times AV_{\text{ref}}$$

3. Conversion starts immediately after DADR0 is modified. After the interval of  $t_{\text{DCONV}}$ , conversion results are output.
4. When the DAOE bit is cleared to 0, analog output is disabled.

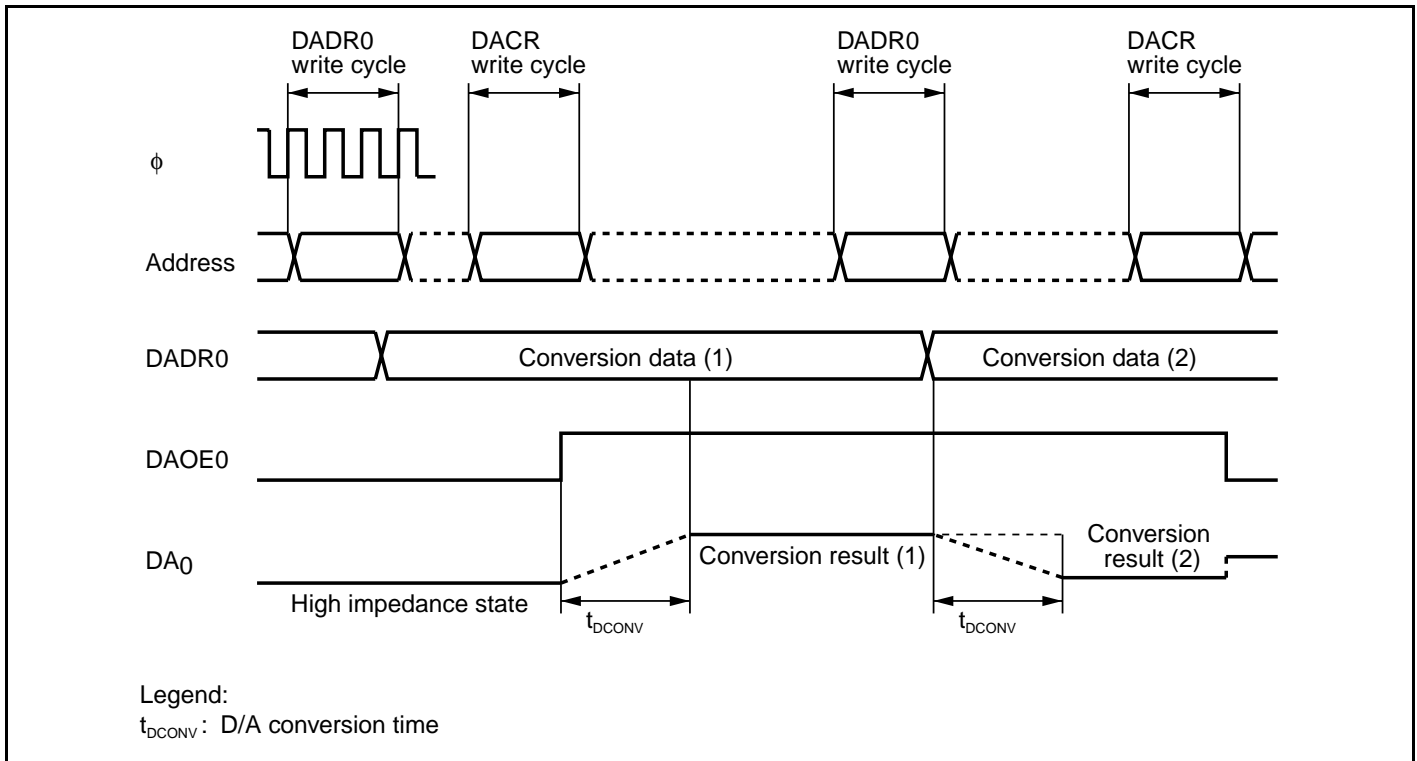


Figure 21.2 D/A Converter Operation Example

## 21.5 Usage Notes

1. When this LSI enters software standby mode with D/A conversion enabled, the D/A output is retained, and the analog power supply current is equal to as during D/A conversion. If the analog power supply current needs to be reduced in software standby mode, clear the DAOE1, DAOE0, and DAE bits all to 0 to disable D/A output.
2. It is not recommended to use the D/A converter or A/D converter simultaneously with the USB because the bus driver/receiver power supply pin DrVCC/DrVSS is shared with the analog power supply pin AVCC/AVSS.



## Section 22 A/D Converter

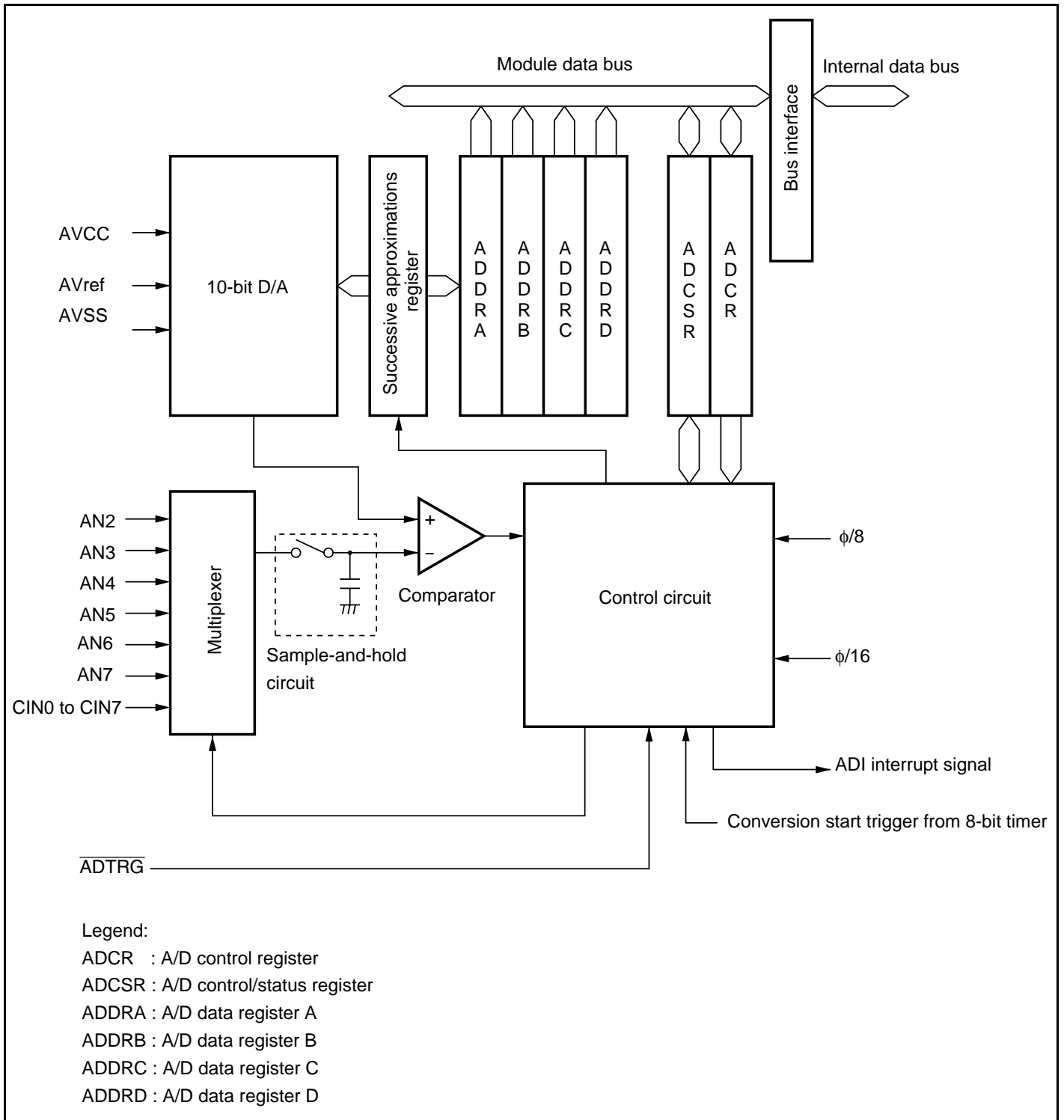
This LSI includes a successive-approximation-type 10-bit A/D converter that allows up to six analog input channels and up to eight digital input channels to be selected.

A/D conversion for digital input is effective as a comparator in multiple input testing.

### 22.1 Features

- 10-bit resolution
- Input channels: six analog input channels and eight digital input channels
- Analog conversion voltage range can be specified using the reference power supply voltage pin (AVref) as an analog reference voltage.
- Conversion time: Max. 5.36  $\mu$ s per channel (at 25-MHz operation)
- Two kinds of operating modes
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels
- Four data registers
  - Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three kinds of conversion start
  - Software, 8-bit timer (TMR) conversion start trigger, or external trigger signal.
- Interrupt request
  - A/D conversion end interrupt (ADI) request can be generated
- Module stop mode can be set

A block diagram of the A/D converter is shown in figure 22.1.



**Figure 22.1 Block Diagram of A/D Converter**

## 22.2 Input/Output Pins

Table 22.1 summarizes the pins used by the A/D converter. The 8 analog input pins are divided into two groups consisting of four channels. Analog input pins 0 to 3 (AN0 to AN3) comprising group 0 and analog input pins 4 to 7 (AN4 to AN7) comprising group 1.

Note that this LSI does not provide the AN0 and AN1 pins. Expanded A/D conversion input pins (CIN0 to CIN7) can be used instead of AN0. The AVCC and AVSS pins are the power supply pins for the analog block in the A/D converter.

**Table 22.1 Pin Configuration**

Pin Name	Symbol	I/O	Function
Analog power supply pin	AVCC	Input	Analog block power supply
Analog ground pin	AVSS	Input	Analog block ground and reference voltage
Reference power supply pin	AVref	Input	Analog block reference voltage
Analog input pin 0	AN0	Not Installed	Group 0 analog input pins
Analog input pin 1	AN1	Not Installed	
Analog input pin 2	AN2	Input	Group 1 analog input pins
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	$\overline{\text{ADTRG}}$	Input	External trigger input pin for starting A/D conversion
Expanded A/D conversion input pins 0 to 7	CIN0 to CIN7	Input	Expanded A/D conversion input (digital input) channels 0 to 7  Can be used as group 0 analog input pins instead of AN0.

## 22.3 Register Descriptions

The A/D converter has the following registers.

- A/D data register A (ADDRA)
- A/D data register B (ADDRB)
- A/D data register C (ADDRC)
- A/D data register D (ADDRD)
- A/D control/status register (ADCSR)
- A/D control register (ADCR)
- Keyboard comparator control register (KBCOMP)

### 22.3.1 A/D Data Registers A to D (ADDRA to ADDR D)

There are four 16-bit read-only ADDR registers, ADDRA to ADDR D, used to store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 22.2.

The converted 10-bit data is stored to bits 15 to 6. The lower 6-bit data is always read as 0.

The data bus between the CPU and the A/D converter is 8-bit width. The upper byte can be read directly from the CPU, but the lower byte should be read via a temporary register. The temporary register contents are transferred from the ADDR when the upper byte data is read. When reading the ADDR, read only the upper byte in word units.

**Table 22.2 Analog Input Channels and Corresponding ADDR Registers**

Analog Input Channel		A/D Data Register to Store A/D Conversion Results
Group 0	Group 1	
CIN0 to CIN7	AN4	ADDRA
—	AN5	ADDRB
AN2	AN6	ADDRC
AN3	AN7	ADDRD



### 22.3.2 A/D Control/Status Register (ADCSR)

ADCSR controls A/D conversion operations.

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W)*	<p>A/D End Flag</p> <p>A status flag that indicates the end of A/D conversion.</p> <p>[Setting conditions]</p> <ul style="list-style-type: none"> <li>• When A/D conversion ends in single mode</li> <li>• When A/D conversion ends on all channels specified in scan mode</li> </ul> <p>[Clearing conditions]</p> <ul style="list-style-type: none"> <li>• When 0 is written after reading ADF = 1</li> <li>• When DTC starts by an ADI interrupt and ADDR is read</li> </ul>
6	ADIE	0	R/W	<p>A/D Interrupt Enable</p> <p>Enables ADI interrupt by ADF when this bit is set to 1</p>
5	ADST	0	R/W	<p>A/D Start</p> <p>Setting this bit to 1 starts A/D conversion. In single mode, this bit is cleared to 0 automatically when conversion on the specified channel ends. In scan mode, conversion continues sequentially on the specified channels until this bit is cleared to 0 by software, a reset, or a transition to standby mode or module stop mode.</p>
4	SCAN	0	R/W	<p>Scan Mode</p> <p>Selects the A/D conversion operating mode.</p> <p>0: Single mode</p> <p>1: Scan mode</p>
3	CKS	0	R/W	<p>Clock Select</p> <p>Sets A/D conversion time.</p> <p>0: Conversion time is 266 states (max)</p> <p>1: Conversion time is 134 states (max)</p> <p>Switch conversion time while ADST is 0.</p>

Bit	Bit Name	Initial Value	R/W	Description
2	CH2	0	R/W	Channel Select 2 to 0
1	CH1	0	R/W	Select analog input channels.
0	CH0	0	R/W	When SCAN = 0 000: CIN0 to CIN7 001: Setting prohibited 010: AN2 011: AN3 100: AN4 101: AN5 110: AN6 111: AN7
				When SCAN = 1 000: CIN0 to CIN7 001: Setting prohibited 010: Setting prohibited 011: Setting prohibited 100: AN4 101: AN4 and AN5 110: AN4 to AN6 111: AN4 to AN7

Note: \* Only 0 can be written for clearing the flag.

### 22.3.3 A/D Control Register (ADCR)

ADCR enables A/D conversion started by an external trigger signal.

Bit	Bit Name	Initial Value	R/W	Description
7	TRGS1	0	R/W	Timer Trigger Select 1 and 0
6	TRGS0	0	R/W	Enable the start of A/D conversion by a trigger signal. Only set bits TRGS1 and TRGS0 while A/D conversion is stopped (ADST = 0). 00: A/D conversion start by external trigger is disabled 01: A/D conversion start by external trigger is disabled 10: A/D conversion start by conversion trigger from TMR 11: A/D conversion start by $\overline{\text{ADTRG}}$ pin
5 to 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

### 22.3.4 Keyboard Comparator Control Register (KBCOMP)

KBCOMP selects the CIN input channel for which A/D conversion is performed and enables or disables the comparator scan function of CIN7 to CIN0.

The DTC decides where to store the A/D conversion result according to settings of the KBCH2 to KBCH0 bits.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/(W)	Reserved The initial value should not be changed.
6, 5	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
4	SCANE	0	R/W	DTC Comparator Scan Enable Enables or disables the DTC comparator scan function. 0: Disables DTC comparator scan function 1: Enables DTC comparator scan function
3	KBADE	0	R/W	Keyboard A/D Enable Sets analog pin 0 (AN0) of the A/D converter to digital input pins (CIN7 to CIN0). 0: Selects AN0 (not mounted on this LSI) 1: Selects CIN7 to CIN0
2	KBCH2	0	R/W	Keyboard A/D Channel Select 2 to 0 These bits select a channel of digital input pins (CIN7 to CIN0) for A/D conversion. The input channel setting must be made while conversion is halted. When the SCANE bit is set to 1, these bits are automatically incremented by the DTC. 000: Selects CIN0 001: Selects CIN1 010: Selects CIN2 011: Selects CIN3 100: Selects CIN4 101: Selects CIN5 110: Selects CIN6 111: Selects CIN7
1	KBCH1	0	R/W	
0	KBCH0	0	R/W	

## 22.4 DTC Comparator Scan

The DTC should be set as shown in table 22.3 to scan CIN7 to CIN0 using the DTC comparator scan function.

**Table 22.3 CIN7 to CIN0 Scan by DTC Comparator Scan Function**

Register	Bit	Bit Name	Description
MRA	7, 6	SM1, SM0	00: SAR is fixed
	5, 4	DM1, DM0	00: DAR is fixed
	3, 2	MD1, MD0	01: Repeat mode
	1	DTS	0: Destination area is repeat area
	0	Sz	1: Word-size transfer
MRB	7	CHNE	0: Chain transfer is not performed
	6	DISEL	0: An interrupt request is generated when the specified number of data transfers are performed
SAR	23 to 0	—	H'(FF)FFE0: ADDR
DAR	23 to 0	—	Optional RAM address. Lower four bits should be 0. Conversion results of CIN0 to CIN7 are written to eight words leading from this address.
CRAH	7 to 0	—	H'FF
CRAL	7 to 0	—	H'FF
DTCERA	3	DTCEA3	1: Enables DTC activation by the A/D converter
KBCOMP	4	SCANE	1: Enables comparator scan function
	3	KBADE	1: Sets CIN7 to CIN0 as A/D converter input channel 0
ADCSR	6	ADIE	1: Enables an interrupt request generated by A/D conversion end
	4	SCAN	1: Selects scan mode
	2 to 0	CH2 to CH0	000: Selects input channel 0
ADCR	7, 6	TRGS1, TRGS0	00: Disables A/D conversion start according to an external trigger
RAM	—	—	(DAR): CIN0 conversion result (DAR) + 2: CIN1 conversion result (DAR) + 4: CIN2 conversion result (DAR) + 6: CIN3 conversion result (DAR) + 8: CIN4 conversion result (DAR) + 10: CIN5 conversion result (DAR) + 12: CIN6 conversion result (DAR) + 14: CIN7 conversion result

The A/D converter repeats A/D conversion of input channel 0 according to the settings of ADCSR and ADCR. Input channel 0 is connected to CIN7 to CIN0. The KBCH2 to KBCH0 bits in KBCOMP select one channel among CIN7 to CIN0. The KBCH2 to KBCH0 bits are automatically incremented by the DTC when the SCANE bit is set to 1.

When one A/D conversion is completed, the ADF flag in ADSCR is set to 1, and an ADI interrupt request is generated. The DTC is activated according to the ADI interrupt request, and data is transferred from ADDRA to on-chip RAM. At this time, the lower four bits of DAR are ignored, and replaced with the previous contents of bits KBCH2 to KBCH0  $\times$  2.

## 22.5 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes: single mode and scan mode. When changing the operating mode or analog input channel, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. The ADST bit can be set at the same time as the operating mode or analog input channel is changed.

### 22.5.1 Single Mode

In single mode, A/D conversion is to be performed only once on the specified single channel. Operations are as follows.

1. A/D conversion on the specified channel is started when the ADST bit in ADCSR is set to 1, by software or an external trigger input.
2. When A/D conversion is completed, the result is transferred to the A/D data register corresponding to the channel.
3. On completion of A/D conversion, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion. When conversion ends, the ADST bit is automatically cleared to 0, and the A/D converter enters wait state.

### 22.5.2 Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the specified channels (four channels max.). Operations are as follows.

1. When the ADST bit in ADCSR is set to 1 by software or an external trigger input, A/D conversion starts on the first channel in the group (CIN0 when the CH2 bit in ADCSR is 0 while the SCANE and KBADE bits in KBCOMP are B'11, or AN4 when the CH2 bit in ADCSR is 1).
2. When A/D conversion for each channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.
3. When conversion of all the 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 is requested after A/D conversion ends. Conversion of the first channel in the group starts again.
4. The ADST bit is not automatically cleared to 0 so 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.

### 22.5.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input when the A/D conversion start delay time ( $t_D$ ) passes after the ADST bit in ADCSR is set to 1, then starts A/D conversion. Figure 22.2 shows the A/D conversion timing. Table 22.4 indicates the A/D conversion time.

As indicated in figure 22.2, the A/D conversion time ( $t_{CONV}$ ) includes  $t_D$  and the input sampling time ( $t_{SPL}$ ). The length of  $t_D$  varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 22.4.

In scan mode, the values given in table 22.4 apply to the first conversion time. In the second and subsequent conversions, the conversion time is 266 states (fixed) when CKS = 0 and 134 states (fixed) when CKS = 1.

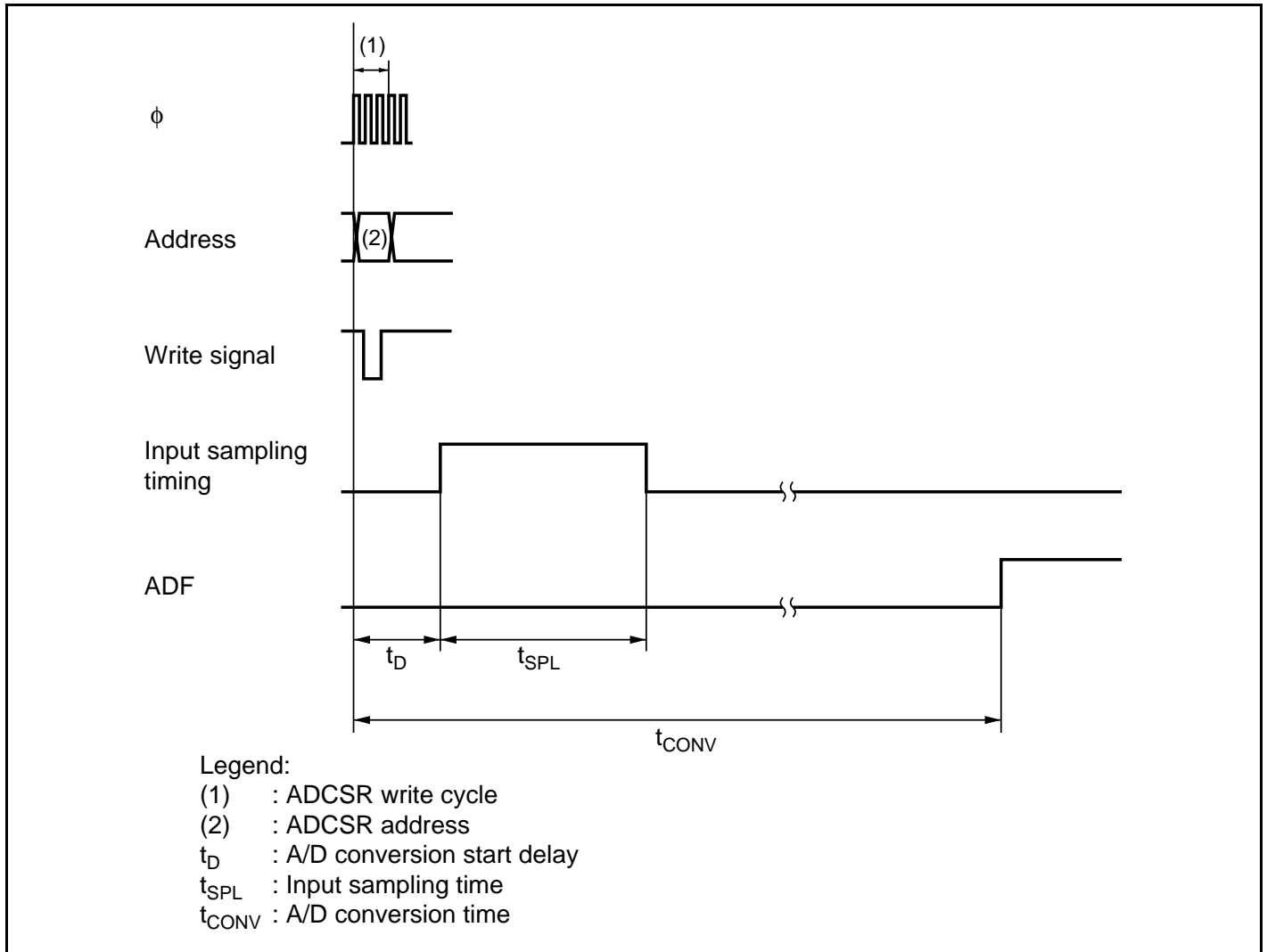


Figure 22.2 A/D Conversion Timing

Table 22.4 A/D Conversion Time (Single Mode)

Item	Symbol	CKS = 0			CKS = 1		
		min	typ	max	min	typ	max
A/D conversion start delay time	$t_D$	10	—	17	6	—	9
Input sampling time	$t_{SPL}$	—	63	—	—	31	—
A/D conversion time	$t_{CONV}$	259	—	266	131	—	134

Note: Values in the table indicate the number of states.

### 22.5.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to B'11 in ADCR, external trigger input is enabled at the  $\overline{\text{ADTRG}}$  pin. A falling edge at the  $\overline{\text{ADTRG}}$  pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 22.3 shows the timing.

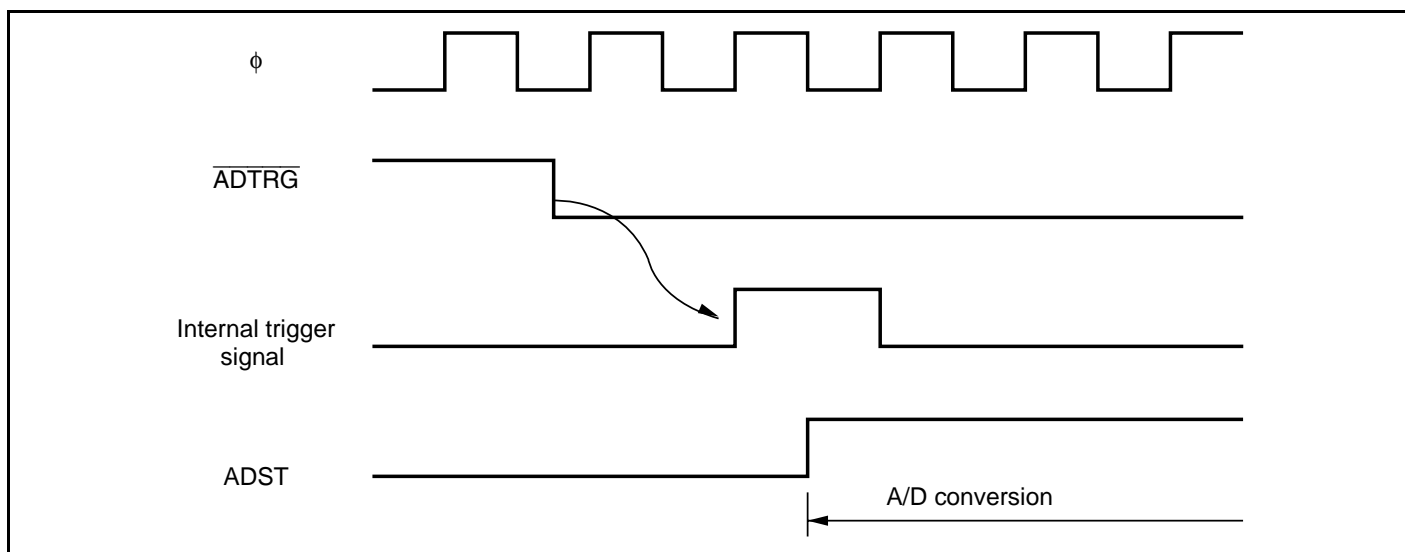


Figure 22.3 External Trigger Input Timing

## 22.6 Interrupt Source

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 enables ADI interrupt requests while the ADF bit in ADCSR is set to 1 after A/D conversion ends.

The ADI interrupt can be used as a DTC activation interrupt source.

Table 22.5 A/D Converter Interrupt Source

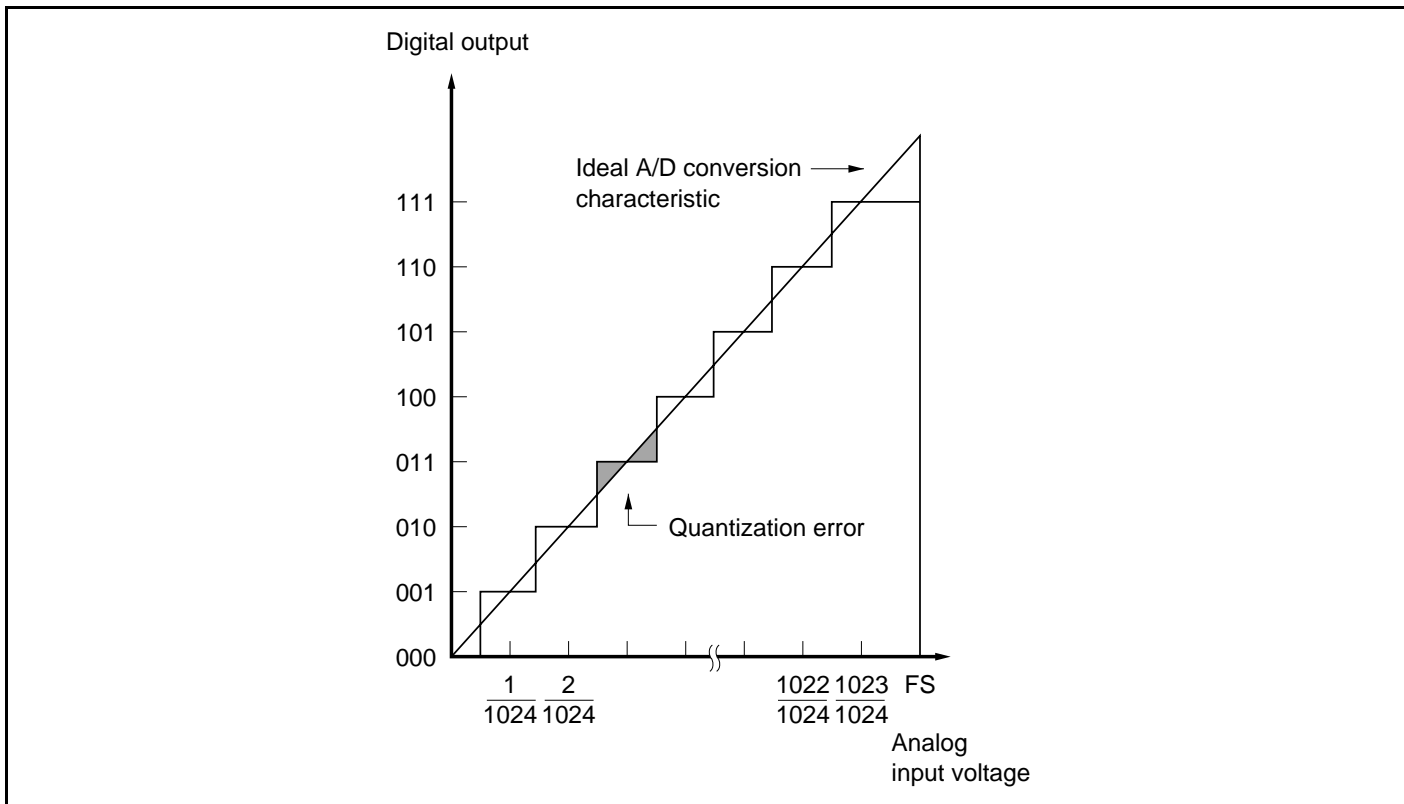
Name	Interrupt Source	Interrupt Flag	DTC Activation
ADI	A/D conversion end	ADF	Possible



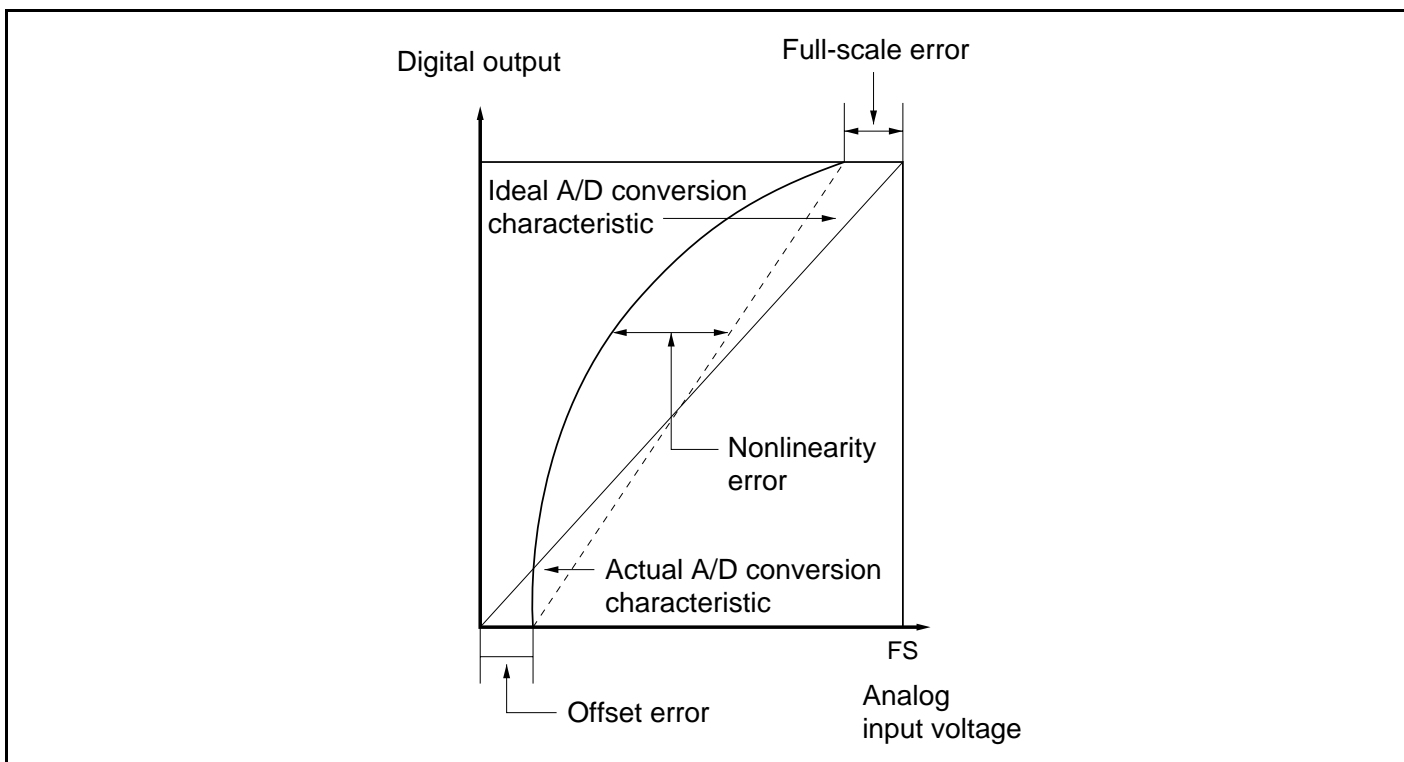
## 22.7 A/D Conversion Accuracy Definitions

This LSI's A/D conversion accuracy definitions are given below.

- Resolution  
The number of A/D converter digital output codes
- Quantization error  
The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 22.4).
- Offset error  
The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'00\_0000\_0000 (H'000) to B'00\_0000\_0001 (H'001) (see figure 22.5).
- 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'11\_1111\_1110 (H'3FE) to B'11\_1111\_1111 (H'3FF) (see figure 22.5).
- Nonlinearity error  
The error with respect to the ideal A/D conversion characteristics between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error (see figure 22.5).
- 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 22.4 A/D Conversion Accuracy Definitions**



**Figure 22.5 A/D Conversion Accuracy Definitions**

## 22.8 Usage Notes

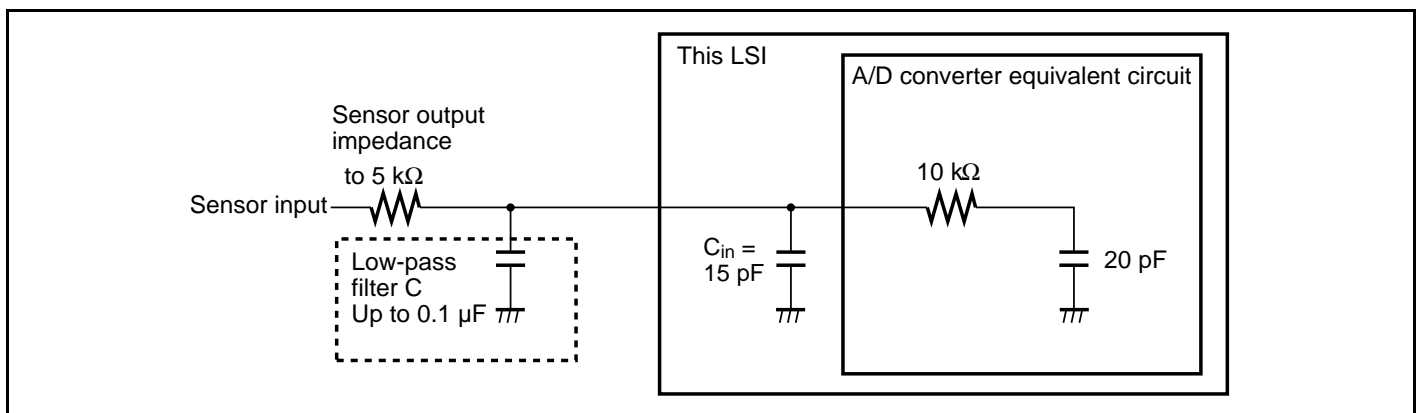
### 22.8.1 Permissible Signal Source Impedance

This LSI's analog input is designed so that the conversion accuracy is guaranteed for an input signal for which the signal source impedance is  $5\text{ k}\Omega$  or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds  $10\text{ k}\Omega$ , charging may be insufficient and it may not be possible to guarantee the A/D conversion accuracy. However, if a large capacitance is provided externally in single mode, the input load will essentially comprise only the internal input resistance of  $10\text{ k}\Omega$ , and the signal source impedance is ignored. However, since a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (e.g., voltage fluctuation ratio of  $5\text{ mV}/\mu\text{s}$  or greater) (see figure 22.6). When converting a high-speed analog signal or converting in scan mode, a low-impedance buffer should be inserted.

### 22.8.2 Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect the absolute accuracy. Be sure to make the connection to an electrically stable GND such as AVSS.

Care is also required to insure that filter circuits do not communicate with digital signals on the mounting board, so acting as antennas.



**Figure 22.6 Example of Analog Input Circuit**

### 22.8.3 Setting Range of Analog Power Supply and Other Pins

If conditions shown below are not met, the reliability of this 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  $AV_{SS} \leq ANn \leq AV_{ref}$  (n = 2 to 7).
- Digital input voltage range  
The voltage applied to digital input pin CINn should be in the range  $AV_{SS} \leq CINn \leq AV_{ref}$  and  $V_{SS} \leq CINn \leq V_{CC}$  (n = 0 to 7).
- Relation between  $AV_{CC}$ ,  $AV_{SS}$  and  $V_{CC}$ ,  $V_{SS}$   
For the relationship between  $AV_{CC}$ ,  $AV_{SS}$  and  $V_{CC}$ ,  $V_{SS}$ , set  $AV_{SS} = V_{SS}$ . If the A/D converter is not used, the AVCC and AVSS pins must on no account be left open.
- AVref pin reference voltage specification range  
The reference voltage of the AVref pin should be in the range  $AV_{ref} \leq AV_{CC}$ .
- It is not recommended to use the D/A converter or A/D converter simultaneously with the USB because the bus driver/receiver power supply pin DrVCC/DrVSS is shared with the analog power supply pin AVCC/AVSS.

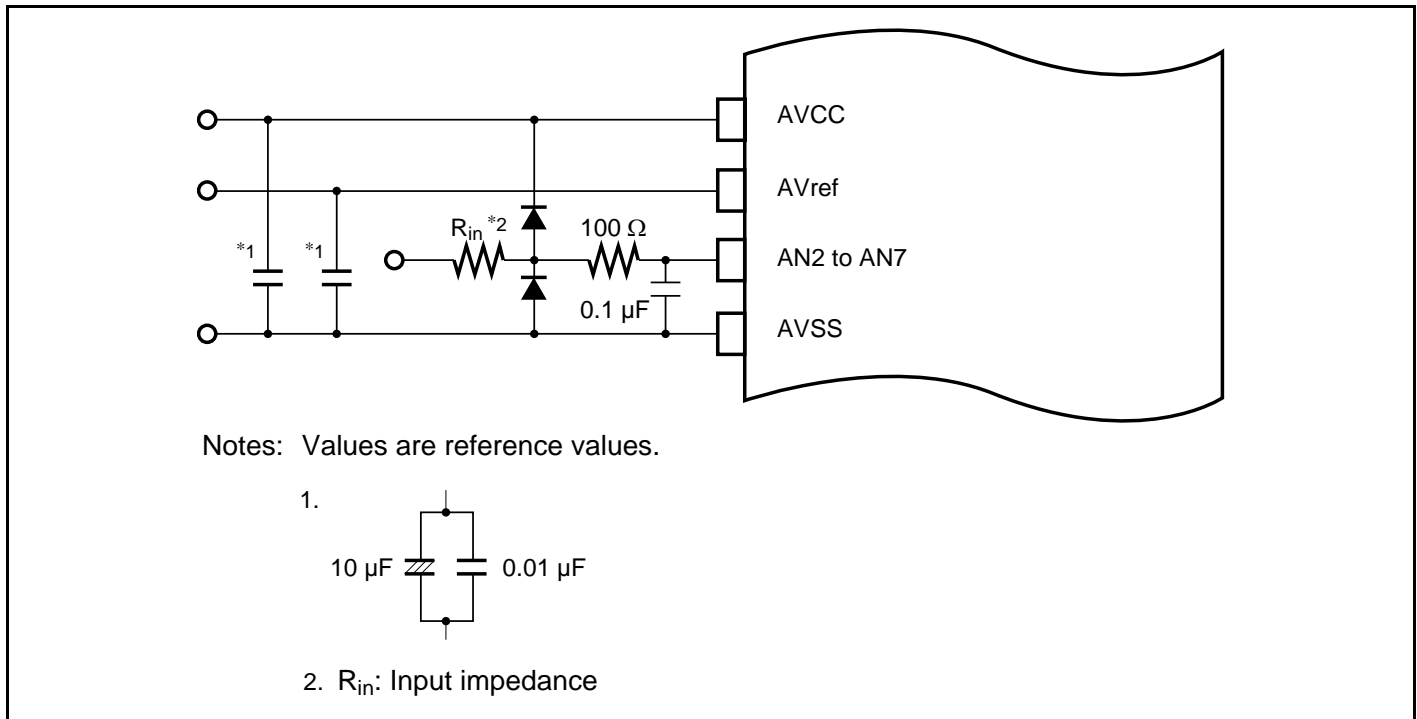
### 22.8.4 Notes on Board Design

In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values. Also, digital circuitry must be isolated from the analog input signals (AN2 to AN7), analog reference voltage ( $AV_{ref}$ ), and analog power supply ( $AV_{CC}$ ) by the analog ground ( $AV_{SS}$ ). Also, the analog ground ( $AV_{SS}$ ) should be connected at one point to a stable digital ground ( $V_{SS}$ ) on the board.

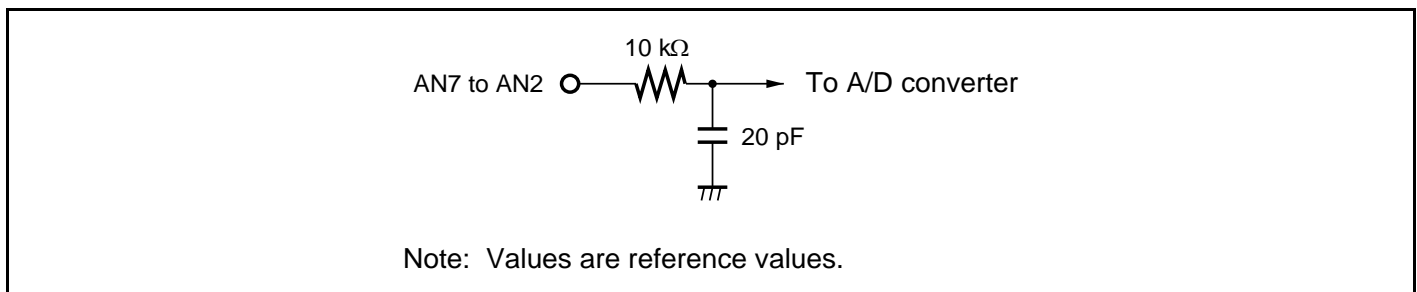
### 22.8.5 Notes on Noise Countermeasures

A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN2 to AN7) and analog reference voltage ( $AV_{ref}$ ) should be connected between AVCC and AVSS as shown in figure 22.7. Also, the bypass capacitors connected to AVCC and AVref, and the filter capacitors connected to AN7 to AN2 must be connected to AVSS.

If a filter capacitor is connected, the input currents at the analog input pins (AN2 to AN7) are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance ( $R_{in}$ ), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.



**Figure 22.7 Example of Analog Input Protection Circuit**



**Figure 22.8 Analog Input Pin Equivalent Circuit**



---

## Section 23 RAM

This LSI has 10 kbytes of on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data.

The on-chip RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on SYSCR, see section 3.2.2, System Control Register (SYSCR).





## Section 24 ROM

This LSI has an on-chip flash memory. The features of the flash memory are summarized below.

A block diagram of the flash memory is shown in figure 24.1.

### 24.1 Features

- Size: 256 kbytes
- Programming/erase methods

The flash memory is programmed 128 bytes at a time. Erase is performed in single-block units. The flash memory is configured as follows: 64 kbytes  $\times$  3 blocks, 32 kbytes  $\times$  1 block, and 4 kbytes  $\times$  8 blocks. To erase the entire flash memory, each block must be erased in turn.
- Programming/erase time

It takes 10 ms (typ.) to program the flash memory 128 bytes at a time; 80  $\mu$ s (typ.) per 1 byte. Erasing one block takes 100 ms (typ.).
- Reprogramming capability

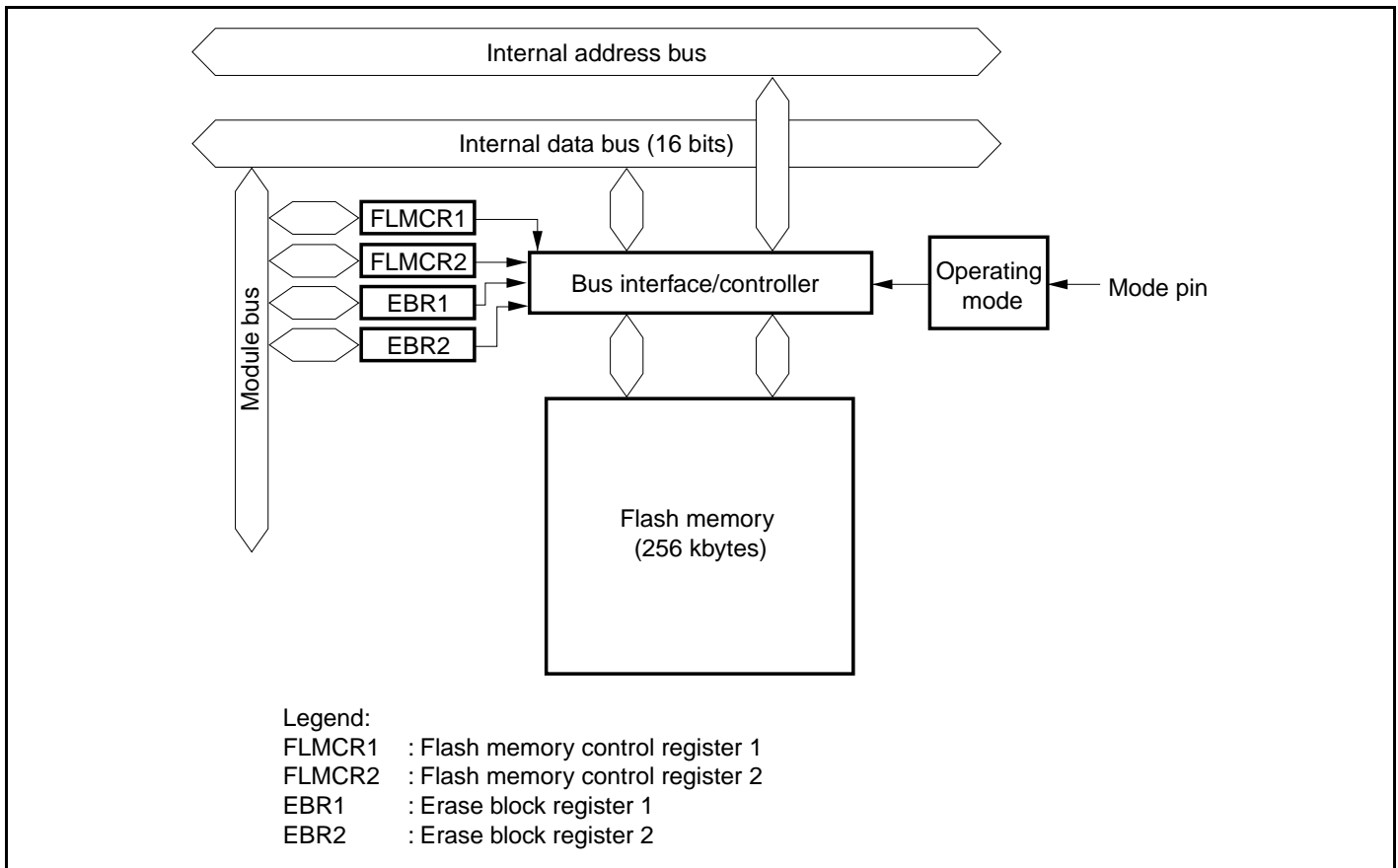
The flash memory can be reprogrammed up to 100 times.
- Two flash memory on-board programming modes
  - Boot mode
  - User program mode

On-board programming/erasing can be done in boot mode in which the boot program built into the chip is started for erase or programming of the entire flash memory. In user program mode, individual blocks can be erased or programmed.
- Automatic bit rate adjustment

With data transfer in boot mode, this LSI's bit rate can be automatically adjusted to match the transfer bit rate of the host.
- Programming/erasing protection

Sets protection against flash memory programming/erasing via hardware, software, or error protection.
- Programmer mode

In addition to on-board programming mode, programmer mode is supported to program or erase the flash memory using a PROM programmer.

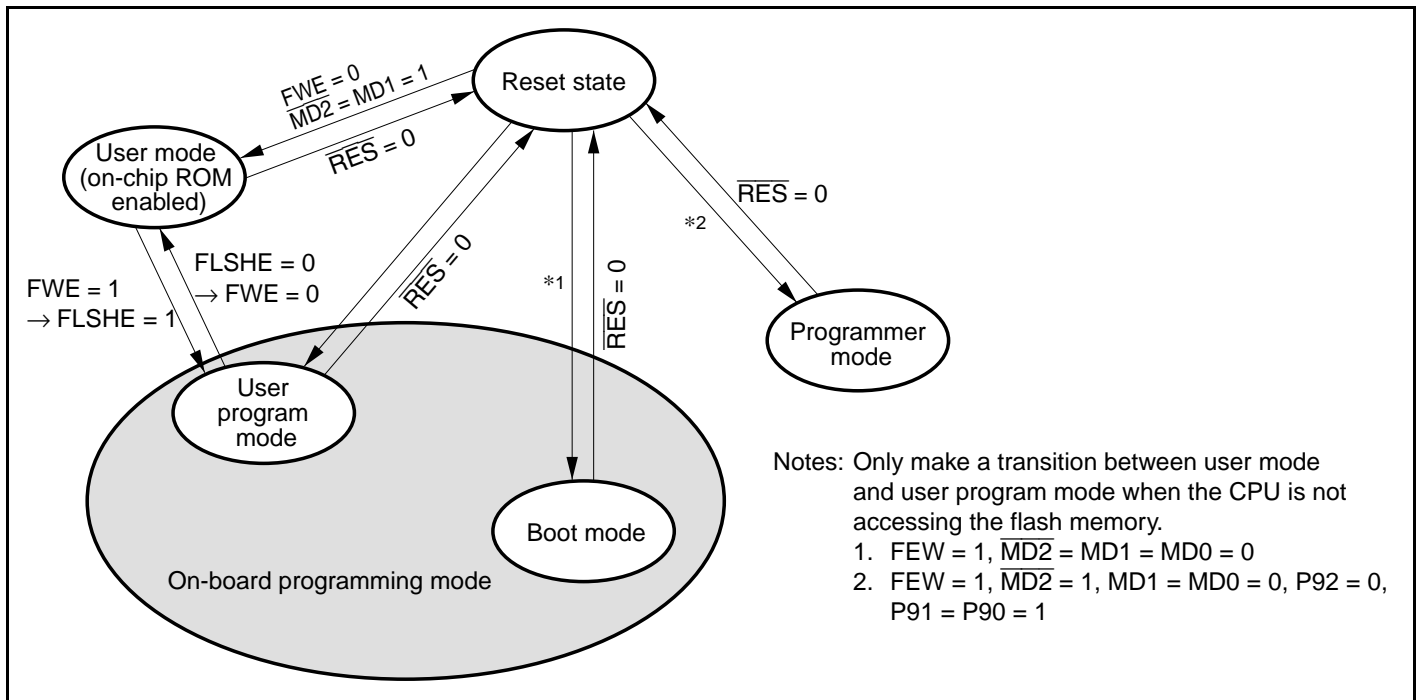


**Figure 24.1 Block Diagram of Flash Memory**

## 24.2 Mode Transition Diagrams

When the mode pins are set in the reset state and a reset-start is executed, this LSI enters an operating mode as shown in figure 24.2. In user mode, flash memory can be read but not programmed or erased. The boot, user program, and programmer modes are provided as modes to write and erase the flash memory.

The differences between boot mode and user program mode are shown in table 24.1. Figure 24.3 shows the boot mode and figure 24.4 shows the user program mode.



**Figure 24.2 Flash Memory State Transitions**

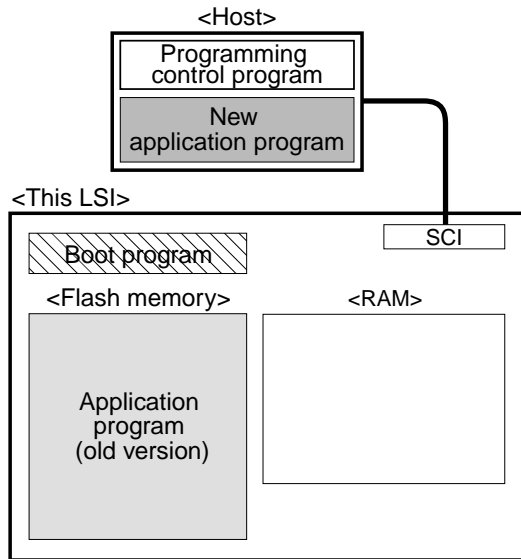
**Table 24.1 Differences between Boot Mode and User Program Mode**

	<b>Boot Mode</b>	<b>User Program Mode</b>
Total erase	Yes	Yes
Block erase	No	Yes
Programming control program*	Program/program-verify	Program/program-verify Erase/erase-verify

Note: \* Should be provided by the user, in accordance with the recommended algorithm.

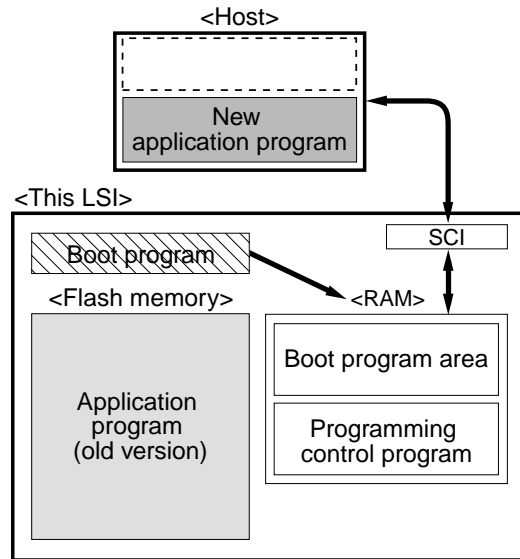
## 1. Initial state

The flash memory is erased at shipment. The following describes how to write over an old-version application program or data in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.



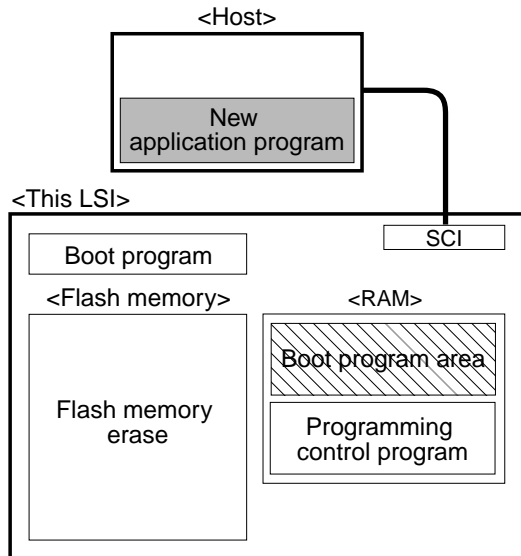
## 2. SCI communication check

When boot mode is entered, the boot program in this LSI (originally incorporated in the chip) is started and SCI communication is checked. Then the boot program required for flash memory erasing is automatically transferred to the RAM boot program area.



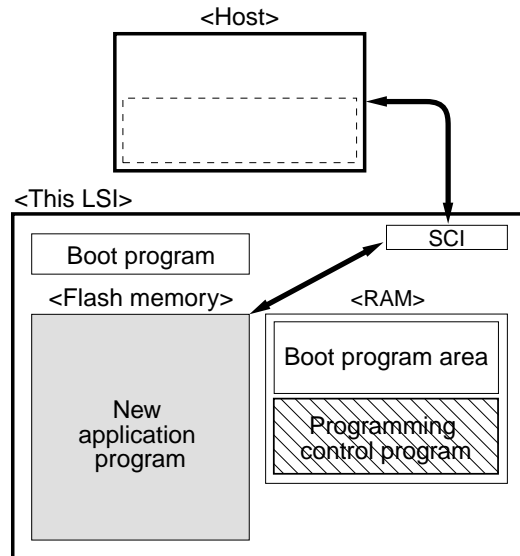
## 3. Flash memory initialization

The erase program in the boot program area (in RAM) is executed, and the flash memory is initialized (to H'FF). In boot mode, total flash memory erasure is performed, without regard to blocks.



## 4. Writing new application program

The programming control program transferred from the host to RAM via SCI communication is executed, and the new application program in the host is written into the flash memory.




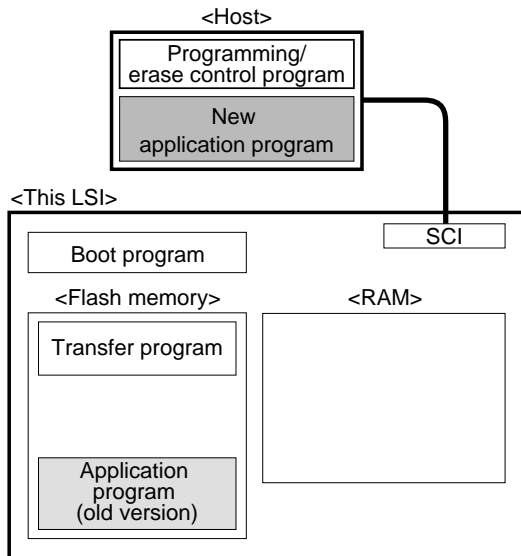
 Program execution state

Figure 24.3 Boot Mode

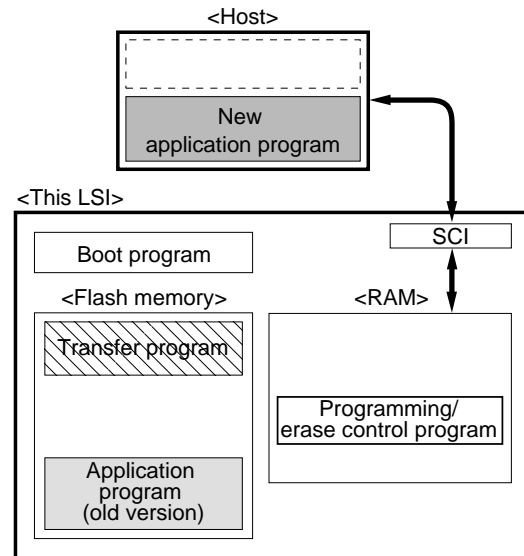
## 1. Initial state

- (1) The program that will transfer the programming/erase control program from flash memory to on-chip RAM should be written into the flash memory by the user beforehand.
- (2) The programming/erase control program should be prepared in the host or in the flash memory.



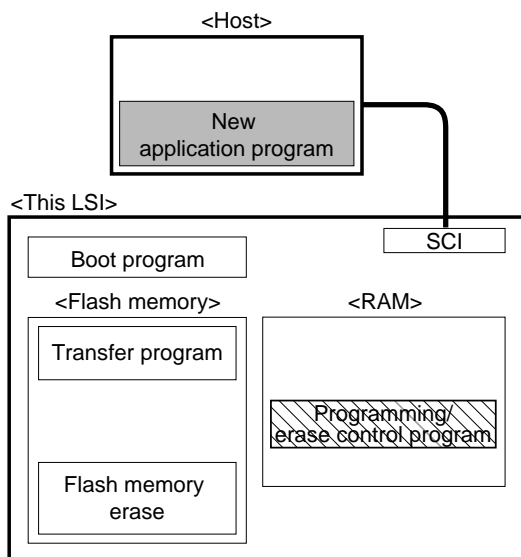
## 2. Programming/erase control program transfer

The transfer program in the flash memory is executed and the programming/erase control program is transferred to RAM.



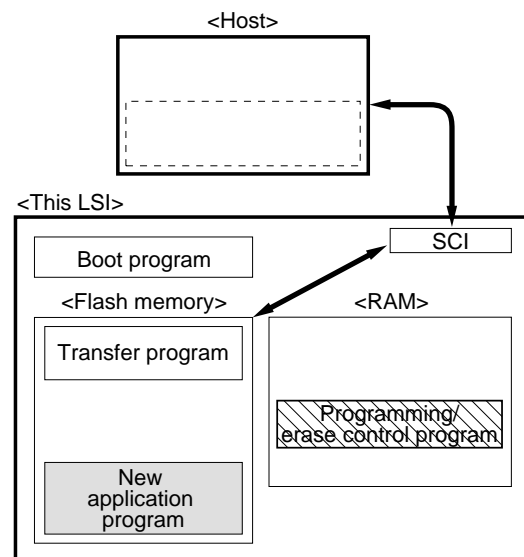
## 3. Flash memory initialization


The programming/erase program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.



## 4. Writing new application program

Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.



 Program execution state

**Figure 24.4 User Program Mode (Example)**

## 24.3 Block Configuration

Figure 24.5 shows the block configuration of 256-kbyte flash memory. The thick lines indicate erasing units, the narrow lines indicate programming units, and the values are addresses. The flash memory is divided into 64 kbytes (3 blocks), 32 kbytes (1 block), and 4 kbytes (8 blocks). Erasing is performed in these divided units. Programming is performed in 128-byte units starting from an address whose lower bits are H'00 or H'80.

EB0 Erase unit: 4 kbytes	H'000000	H'000001	H'000002	→Programming unit: 128 bytes→	H'00007F
	H'000F80	H'000F81	H'000F82	-----	H'000FFF
EB1 Erase unit: 4 kbytes	H'001000	H'001001	H'001002	→Programming unit: 128 bytes→	H'00107F
	H'001F80	H'001F81	H'001F82	-----	H'001FFF
EB2 Erase unit: 4 kbytes	H'002000	H'002001	H'002002	→Programming unit: 128 bytes→	H'00207F
	H'002F80	H'002F81	H'002F82	-----	H'002FFF
EB3 Erase unit: 4 kbytes	H'003000	H'003001	H'003002	→Programming unit: 128 bytes→	H'00307F
	H'003F80	H'003F81	H'003F82	-----	H'003FFF
EB4 Erase unit: 32 kbytes	H'004000	H'004001	H'004002	→Programming unit: 128 bytes→	H'00407F
	H'00BF80	H'00BF81	H'00BF82	-----	H'00BFFF
EB5 Erase unit: 4 kbytes	H'00C000	H'00C001	H'00C002	→Programming unit: 128 bytes→	H'00C07F
	H'00CF80	H'00CF81	H'00CF82	-----	H'00CFFF
EB6 Erase unit: 4 kbytes	H'00D000	H'00D001	H'00D002	→Programming unit: 128 bytes→	H'00D07F
	H'00DF80	H'00DF81	H'00DF82	-----	H'00DFFF
EB7 Erase unit: 4 kbytes	H'00E000	H'00E001	H'00E002	→Programming unit: 128 bytes→	H'00E07F
	H'00EF80	H'00EF81	H'00EF82	-----	H'00EFFF
EB8 Erase unit: 4 kbytes	H'00F000	H'00F001	H'00F002	→Programming unit: 128 bytes→	H'00F07F
	H'00FF80	H'00FF81	H'00FF82	-----	H'00FFFF
EB9 Erase unit: 64 kbytes	H'010000	H'010001	H'010002	→Programming unit: 128 bytes→	H'01007F
	H'01FF80	H'01FF81	H'01FF82	-----	H'01FFFF
EB10 Erase unit: 64 kbytes	H'020000	H'020001	H'020002	→Programming unit: 128 bytes→	H'02007F
	H'02FF80	H'02FF81	H'02FF82	-----	H'02FFFF
EB11 Erase unit: 64 kbytes	H'030000	H'030001	H'030002	→Programming unit: 128 bytes→	H'03007F
	H'03FF80	H'03FF81	H'03FF82	-----	H'03FFFF

Figure 24.5 Flash Memory Block Configuration

## 24.4 Input/Output Pins

The flash memory is controlled by means of the pins shown in table 24.2.

**Table 24.2 Pin Configuration**

Pin Name	I/O	Function
$\overline{\text{RES}}$	Input	Reset
$\overline{\text{MD2}}$	Input	Sets this LSI's operating mode
MD1	Input	Sets this LSI's operating mode
MD0	Input	Sets this LSI's operating mode
FWE	Input	Flash memory pin
TxD1	Output	Serial transmit data output
RxD1	Input	Serial receive data input

## 24.5 Register Descriptions

The flash memory has the following registers. To access FLMCR1, FLMCR2, EBR1, or EBR2, the FLSHE bit in the serial/timer control register (STCR) should be set to 1. For details on the serial/timer control register, see section 3.2.3, Serial Timer Control Register (STCR).

- Flash memory control register 1 (FLMCR1)
- Flash memory control register 2 (FLMCR2)
- Erase block register 1 (EBR1)
- Erase block register 2 (EBR2)

### 24.5.1 Flash Memory Control Register 1 (FLMCR1)

FLMCR1, used together with FLMCR2, makes the flash memory transit to program mode, program-verify mode, erase mode, or erase-verify mode. For details on register setting, see section 24.8, Flash Memory Programming/Erasing.

FLMCR1 is initialized to H'00 by a reset, or in hardware standby mode, software standby mode, sub-active mode, sub-sleep mode, or watch mode.

Bit	Bit Name	Initial Value	R/W	Description
7	FWE	0	R	Flash Write Enable Used to monitor the FWE pin state. When this bit is cleared to 0, flash memory write or erasure is protected by hardware. When this bit is set to 1, hardware protect is cancelled and the SWE bit can be read from or written to.
6	SWE	0	R/W	Software Write Enable When this bit is set to 1, flash memory programming/erasing is enabled. When this bit is cleared to 0, the EV, PV, E, and P bits in this register, the ESU and PSU bits in FLMCR2, and all EBR2 bits cannot be set. Do not clear these bits and SWE to 0 simultaneously.
5, 4	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
3	EV	0	R/W	Erase-Verify When this bit is set to 1 while SWE = 1, the flash memory transits to erase-verify mode. When it is cleared to 0, erase-verify mode is cancelled.
2	PV	0	R/W	Program-Verify When this bit is set to 1 while SWE = 1, the flash memory transits to program-verify mode. When it is cleared to 0, program-verify mode is cancelled.
1	E	0	R/W	Erase When this bit is set to 1 while SWE = 1 and ESU = 1, the flash memory transits to erase mode. When it is cleared to 0, erase mode is cancelled.
0	P	0	R/W	Program When this bit is set to 1 while SWE = 1 and PSU = 1, the flash memory transits to program mode. When it is cleared to 0, program mode is cancelled.



### 24.5.2 Flash Memory Control Register 2 (FLMCR2)

FLMCR2 monitors the state of flash memory programming/erasing protection (error protection) and sets up the flash memory to transit to programming/erasing mode. FLMCR2 is initialized to H'00 by a reset or in hardware standby mode. The ESU and PSU bits are cleared to 0 in software standby mode, sub-active mode, sub-sleep mode, or watch mode, or when the SWE bit in FLMCR1 is cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7	FLER	0	R	Indicates that an error has occurred during flash memory programming/erasing. When this bit is set to 1, flash memory goes to the error-protection state. For details, see section 24.9.3, Error Protection.
6 to 2	—	All 0	R/(W)	Reserved The initial value should not be changed.
1	ESU	0	R/W	Erase Setup When this bit is set to 1 while SWE = 1, the flash memory transits to the erase setup state. When it is cleared to 0, the erase setup state is cancelled. Set this bit to 1 before setting the E bit in FLMCR1 to 1.
0	PSU	0	R/W	Program Setup When this bit is set to 1 while SWE = 1, the flash memory transits to the program setup state. When it is cleared to 0, the program setup state is cancelled. Set this bit to 1 before setting the P bit in FLMCR1 to 1.

### 24.5.3 Erase Block Registers 1 and 2 (EBR1, EBR2)

EBR1 and EBR2 are used to specify the flash memory erase block. EBR1 and EBR2 are initialized to H'00 by a reset, or in hardware standby mode, software standby mode, sub-active mode, or sub-sleep mode, or when the SWE bit in FLMCR1 is cleared to 0. Set only one bit to 1 at a time, otherwise all bits in EBR1 and EBR2 are automatically cleared to 0.

**EBR1**

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/(W)	Reserved The initial value should not be changed.
3	EB11	0	R/W*	When this bit is set to 1, 64 kbytes of EB11 (H'030000 to H'03FFFF) are to be erased.
2	EB10	0	R/W*	When this bit is set to 1, 64 kbytes of EB10 (H'020000 to H'02FFFF) are to be erased.
1	EB9	0	R/W*	When this bit is set to 1, 64 kbytes of EB9 (H'010000 to H'01FFFF) are to be erased.
0	EB8	0	R/W*	When this bit is set to 1, 4 kbytes of EB8 (H'00F000 to H'00FFFF) are to be erased.

Note: \* In normal mode, this bit is always read as 0 and cannot be modified.

**EBR2**

Bit	Bit Name	Initial Value	R/W	Description
7	EB7	0	R/W*	When this bit is set to 1, 4 kbytes of EB7 (H'00E000 to H'00EFFF) are to be erased.
6	EB6	0	R/W	When this bit is set to 1, 4 kbytes of EB6 (H'00D000 to H'00DFFF) are to be erased.
5	EB5	0	R/W	When this bit is set to 1, 4 kbytes of EB5 (H'00C000 to H'00CFFF) are to be erased.
4	EB4	0	R/W	When this bit is set to 1, 32 kbytes of EB4 (H'004000 to H'00BFFF) are to be erased.
3	EB3	0	R/W	When this bit is set to 1, 4 kbytes of EB3 (H'003000 to H'003FFF) is to be erased.
2	EB2	0	R/W	When this bit is set to 1, 4 kbytes of EB2 (H'002000 to H'002FFF) is to be erased.
1	EB1	0	R/W	When this bit is set to 1, 4 kbytes of EB1 (H'001000 to H'001FFF) is to be erased.
0	EB0	0	R/W	When this bit is set to 1, 4 kbytes of EB0 (H'000000 to H'000FFF) is to be erased.

Note: \* In normal mode, this bit is always read as 0 and cannot be modified.

## 24.6 Operating Modes

The flash memory is connected to the CPU via a 16-bit data bus, enabling byte data and word data to be accessed in a single state. Even addresses are connected to the upper 8 bits and odd addresses are connected to the lower 8 bits. Note that word data must start from an even address.

On-chip ROM is enabled or disabled by the mode select pins ( $\overline{\text{MD2}}$ , MD1, and MD0) and the EXPE bit in MDCR, as summarized in table 24.3.

In normal mode, up to 56 kbytes of ROM can be used.

**Table 24.3 Operating Modes and ROM**

MCU Operating Mode	Operating Modes		Mode Pins			MDCR	On-Chip ROM
	CPU Operating Mode	Mode	$\overline{\text{MD2}}$	MD1	MD0	EXPE	
Mode 2	Advanced	Single-chip mode	1	1	0	0	Enabled (256 kbytes)
	Advanced	Extended mode with on-chip ROM	1	1	0	1	
Mode 3	Normal	Single-chip mode	1	1	1	0	Enabled (56 kbytes)
	Normal	Extended mode without on-chip ROM	1	1	1	1	

## 24.7 On-Board Programming Modes

An on-board programming mode is used to perform on-chip flash memory programming, erasing, and verification. This LSI has two on-board programming modes: boot mode and user program mode. Table 24.4 shows pin settings for boot mode. In user program mode, operation by software is enabled by setting control bits. For details on flash memory mode transitions, see figure 24.2.

**Table 24.4 On-Board Programming Mode Settings**

Mode Setting		FWE	$\overline{\text{MD2}}$	MD1	MD0
Boot mode		1	0	0	0
User program mode	Mode 2 (advanced mode)	1	1	1	0
	Mode 3 (normal mode)	1	1	1	1

### 24.7.1 Boot Mode

Table 24.5 shows the boot mode operations between reset end and branching to the programming control program.

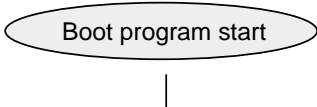
1. When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. Prepare a programming control program in accordance with the description in section 24.8, Flash Memory Programming/Erasing. In boot mode, if any data exists in the flash memory (except in the case that all data are 1), all blocks in the flash memory are erased. Use boot mode at initial writing in the on-board state, or forced recovery when user program mode cannot be executed because the program to be initiated in user program mode was mistakenly erased.
2. The SCI\_1 should be set to asynchronous mode, and the transfer format as follows: 8-bit data, 1 stop bit, and no parity.
3. When the boot program is initiated, this LSI measures the low-level period of asynchronous SCI communication data (H'00) transmitted continuously from the host. This LSI then calculates the bit rate of transmission from the host, and adjusts the SCI\_1 bit rate to match that of the host. The reset should end with the RxD1 pin high. The RxD1 and TxD1 pins should be pulled up on the board if necessary. After the reset ends, it takes approximately 100 states before this LSI is ready to measure the low-level period.
4. After matching the bit rates, this LSI transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to this LSI. If reception could not be performed normally, initiate boot mode again by a reset. Depending on the host's transfer bit rate and system clock frequency of this LSI, there will be a discrepancy between the bit rates of the host and this LSI. To operate the SCI properly, set the host's transfer bit rate and system clock frequency of this LSI within the ranges listed in table 24.6.
5. In boot mode, a part of the on-chip RAM area is used by the boot program. Addresses H'FF0800 to H'FF1FFF is the area to which the programming control program is transferred from the host. Note, however, that ID codes are assigned to addresses H'FF0800 to H'FF0807. The boot program area cannot be used until the execution state in boot mode switches to the programming control program. Figure 24.6 shows the on-chip RAM area in boot mode.
6. Before branching to the programming control program (H'FF0808 in the RAM area), this LSI terminates transfer operations by the SCI\_1 (by clearing the RE and TE bits in SCR to 0), but the adjusted bit rate value remains set in BRR. Therefore, the programming control program can still use it for transfer of write data or verify data with the host. The TxD1 pin is in high-level output state. The contents of the CPU general registers are undefined immediately after

branching to the programming control program. These registers must be initialized at the beginning of the programming control program, since the stack pointer (SP), in particular, is used implicitly in subroutine calls, etc.

7. Boot mode can be cleared by a reset. Cancel the reset\*<sup>1</sup> after driving the reset pin low, waiting at least 20 states, and then setting the mode pins. Boot mode is also cleared when a WDT overflow occurs.
8. Do not change the mode pin input levels in boot mode. If mode pin input levels are changed from low to high during reset, operating modes are switched and the state of ports that are also used for address output and bus control output signals ( $\overline{AS}$ ,  $\overline{RD}$ , and  $\overline{HWR}$ ) are changed.\*<sup>2</sup> Therefore, set these pins carefully not to be output signals during reset or not to conflict with LSI external signals.
9. All interrupts are disabled during programming or erasing of the flash memory.

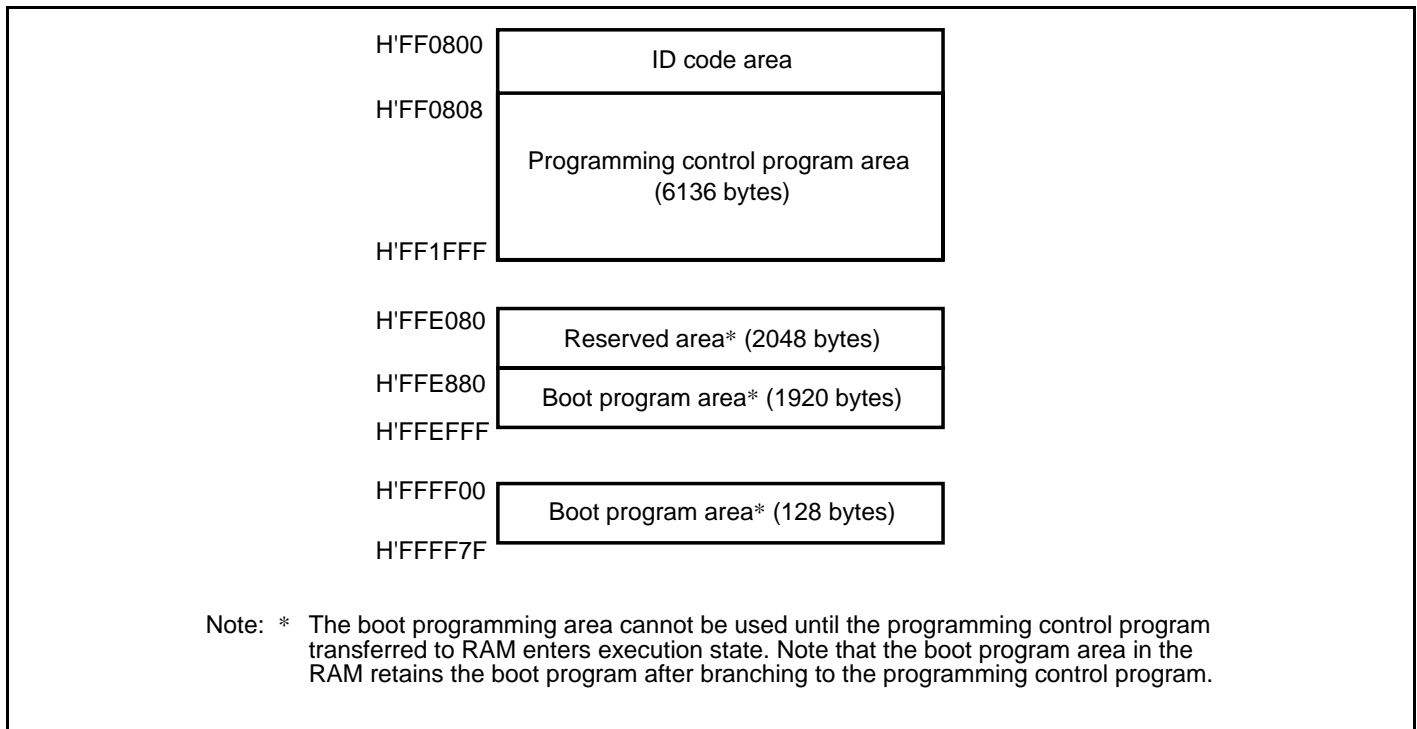
- Notes:
1. After reset is cancelled, mode pin input settings must satisfy the mode programming setup time ( $t_{MDS} = 4$  states).
  2. The ports that also have address output functions output low as address output when the MD1 pin is set to 0 during a reset. In other cases, it enters the high impedance state. Bus control output signals output high when the MD1 pin is set to 0 during a reset. In other cases, it enters the high impedance state.

**Table 24.5 Boot Mode Operation**

Item	Host Operation	Communications Contents	LSI Operation
	Processing Contents		Processing Contents
Boot mode start			Branches to boot program at reset-start.  
Bit rate adjustment	Continuously transmits data H'00 at specified bit rate. ↓ Transmits data H'55 when data H'00 is received error-free. ↓ Receives data H'AA.	H'00, H'00 ··· H'00 ← H'00 ← H'55 ← H'AA	<ul style="list-style-type: none"> <li>Measures low-level period of receive data H'00.</li> <li>Calculates bit rate and sets it in BRR of SCI_1.</li> <li>Transmits data H'00 to host as adjustment end indication.</li> </ul> ↓ After receiving data H'55, transmits data H'AA to host.
Transfer of programming control program	Transmits number of bytes (N) of programming control program to be transferred as 2-byte data (low-order byte following high-order byte). ↓ Transmits 1 byte of programming control program (repeated for N times).	High-order byte and low-order byte ← Echoback H'XX ← Echoback	Echobacks the 2-byte data received to host. ↓ Echobacks received data to host and also transfers it to RAM (repeated for N times).
Flash memory erase	↓ Boot program erase error ↓ Receives data H'AA.	← H'FF ← H'AA	Checks flash memory data, erases all flash memory blocks in case of written data existing, and transmits data H'AA to host. (If erase could not be done, transmits data H'FF to host and aborts operation.)
			↓ Branches to programming control program transferred to on-chip RAM and starts execution.

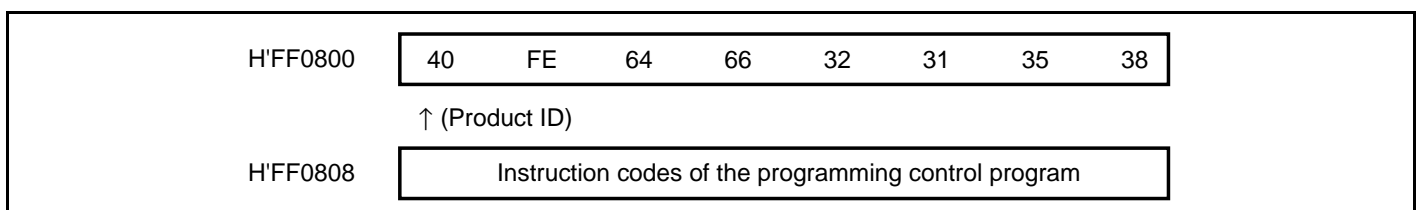
**Table 24.6 System Clock Frequencies for which Automatic Adjustment of LSI Bit Rate Is Possible**

Host Bit Rate	System Clock Frequency Range of LSI
19200 bps	8 to 25 MHz
9600 bps	5 to 25 MHz
4800 bps	5 to 25 MHz



**Figure 24.6 On-Chip RAM Area in Boot Mode**

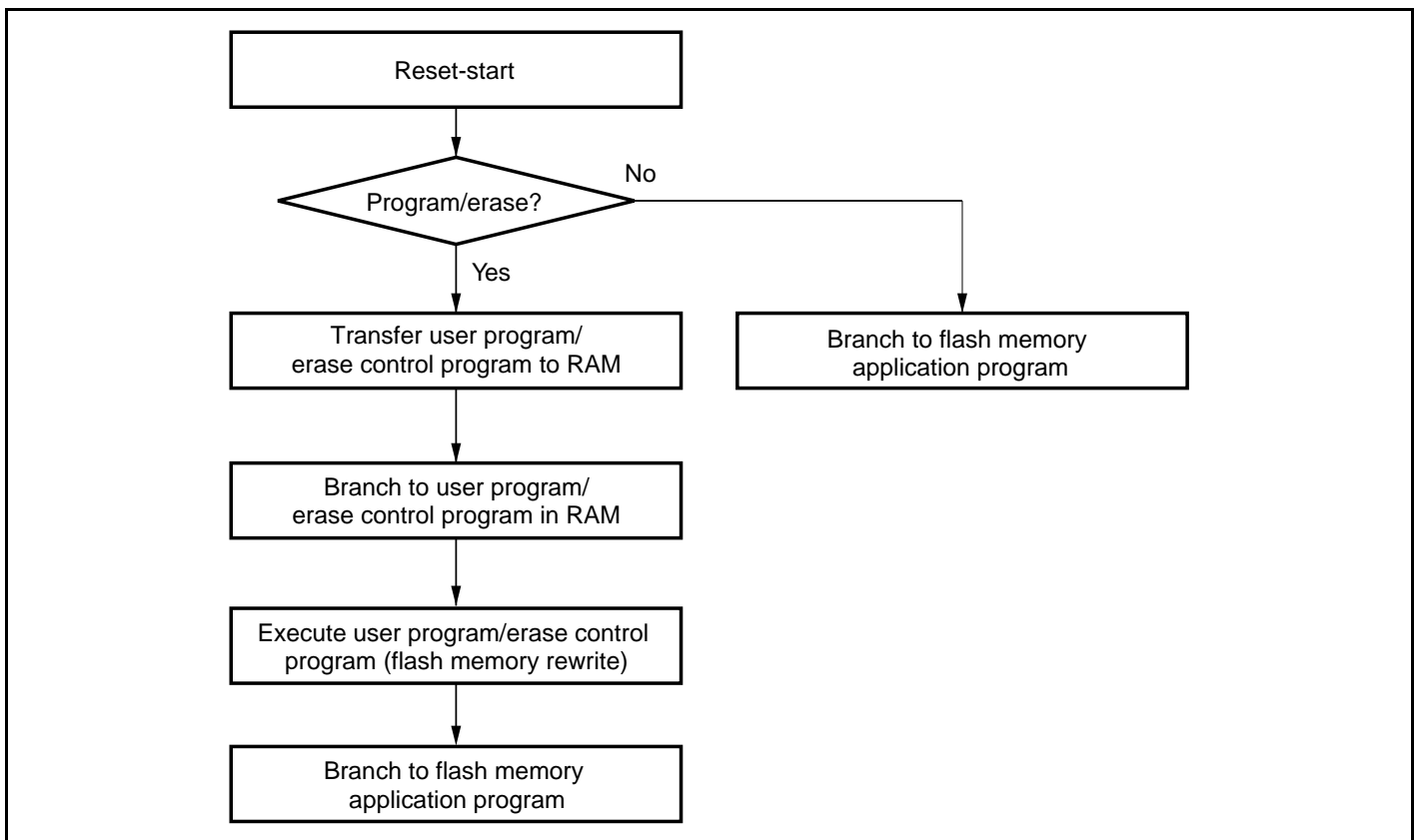
In boot mode, this LSI checks the contents of the 8-byte ID code area as shown below to confirm that the programming control program corresponds with this LSI. To originally write a programming control program to be used in boot mode, the above 8-byte ID code must be added at the beginning of the program.



**Figure 24.7 ID Code Area**

## 24.7.2 User Program Mode

On-board programming/erasing of an individual flash memory block can also be performed in user program mode by branching to a user program/erase control program. The user must set branching conditions and provide on-board means of supplying programming data. The flash memory must contain the user program/erase control program or a program which provides the user program/erase control program from external memory. Because the flash memory itself cannot be read during programming/erasing, transfer the user program/erase control program to on-chip RAM, as like in boot mode. Figure 24.8 shows a sample procedure for programming/erasing in user program mode. Prepare a user program/erase control program in accordance with the description in section 24.8, Flash Memory Programming/Erasing.



**Figure 24.8 Programming/Erasing Flowchart Example in User Program Mode**

## 24.8 Flash Memory Programming/Erasing

A software method, using the CPU, is employed to program and erase flash memory in the on-board programming modes. Depending on the FLMCR1 and FLMCR2 settings, the flash memory operates in one of the following four modes: program mode, erase mode, program-verify mode, and erase-verify mode. The programming control program in boot mode and the user program/erase control program in user program mode use these operating modes in combination to perform programming/erasing. Flash memory programming and erasing should be performed in



accordance with the descriptions in section 24.8.1, Program/Program-Verify and section 24.8.2, Erase/Erase-Verify, respectively.

### 24.8.1 Program/Program-Verify

When writing data or programs to the flash memory, the program/program-verify flowchart shown in figure 24.9 should be followed. Performing programming operations according to this flowchart will enable data or programs to be written to the flash memory without subjecting this LSI to voltage stress or sacrificing program data reliability.

1. Programming must be done to an empty address. Do not reprogram an address to which programming has already been performed.
2. Programming should be carried out 128 bytes at a time. A 128-byte data transfer must be performed even if writing fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
3. Prepare the following data storage areas in RAM: a 128-byte programming data area, a 128-byte reprogramming data area, and a 128-byte additional-programming data area. Perform reprogramming data computation and additional programming data computation according to figure 24.9.
4. Consecutively transfer 128 bytes of data in byte units from the reprogramming data area or additional-programming data area to the flash memory. The program address and 128-byte data are latched in the flash memory. The lower 8 bits of the start address in the flash memory destination area must be H'00 or H'80.
5. The time during which the P bit is set to 1 is the programming time. Figure 24.9 shows the allowable programming times.
6. The watchdog timer (WDT) is set to prevent overprogramming due to program runaway, etc. The overflow cycle should be longer than  $(y + z^2 + \alpha + \beta) \mu\text{s}$ .
7. For a dummy write to a verify address, write 1-byte data H'FF to an address whose lower 2 bits are B'00. Verify data can be read in words from the address to which a dummy write was performed.
8. The maximum number of repetitions of the program/program-verify sequence to the same bit is (N).

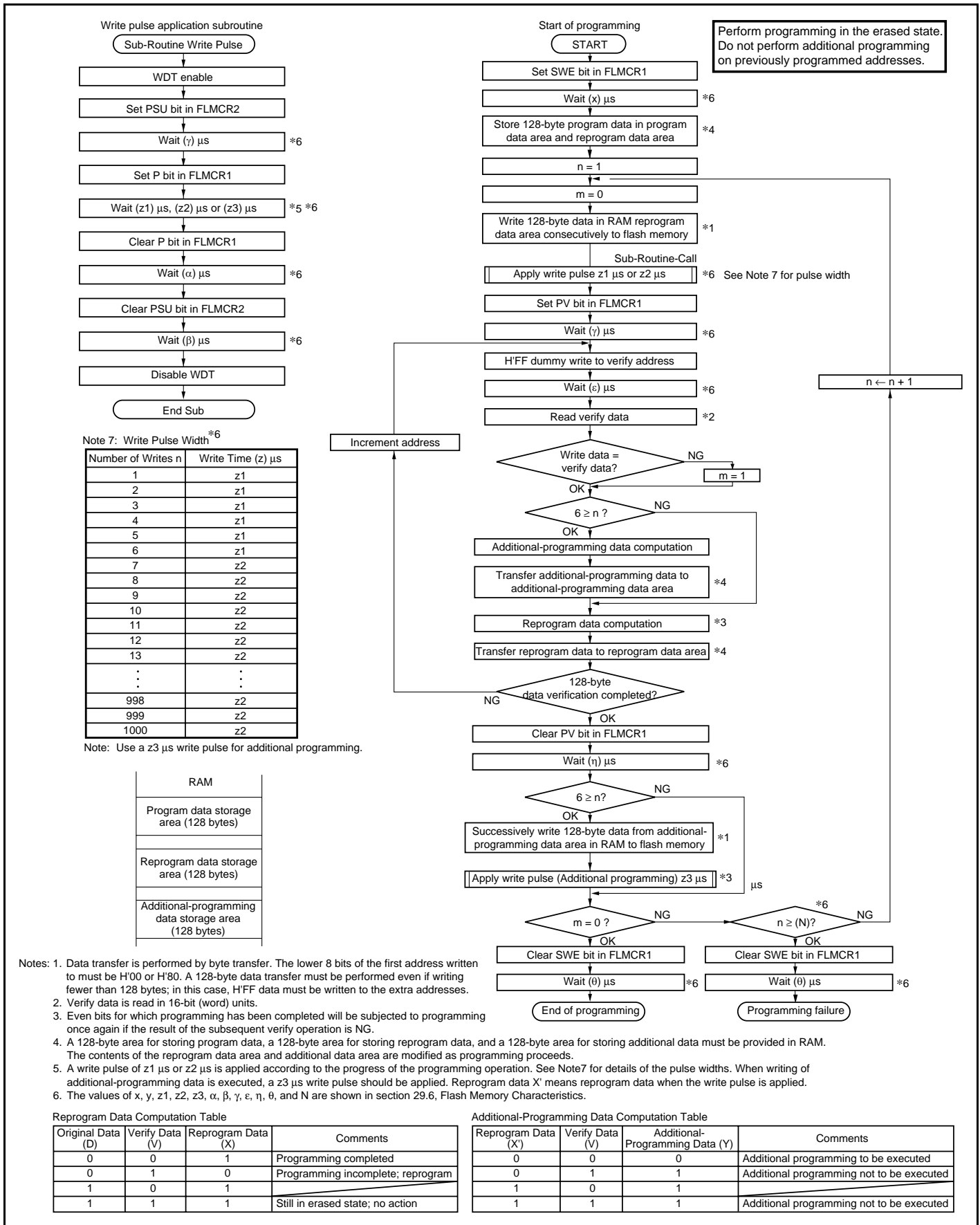


Figure 24.9 Program/Program-Verify Flowchart

## 24.8.2 Erase/Erase-Verify

When erasing flash memory, the erase/erase-verify flowchart shown in figure 24.10 should be followed.

1. Prewriting (setting erase block data to all 0) is not necessary.
2. Erasing is performed in block units. Make only a single-block specification in erase block registers 1 and 2 (EBR1 and EBR2). To erase multiple blocks, each block must be erased in turn.
3. The time during which the E bit is set to 1 is the flash memory erase time.
4. The watchdog timer (WDT) is set to prevent overprogramming due to program runaway, etc. An overflow cycle of approximately  $(y + z + \alpha + \beta)$  ms is allowed.
5. For a dummy write to a verify address, write 1-byte data H'FF to an address whose lower two bits are B'00. Verify data can be read in words from the address to which a dummy write was performed.
6. If the read data is unerased, set erase mode again, and repeat the erase/erase-verify sequence as before. The maximum number of repetitions of the erase/erase-verify sequence is N.

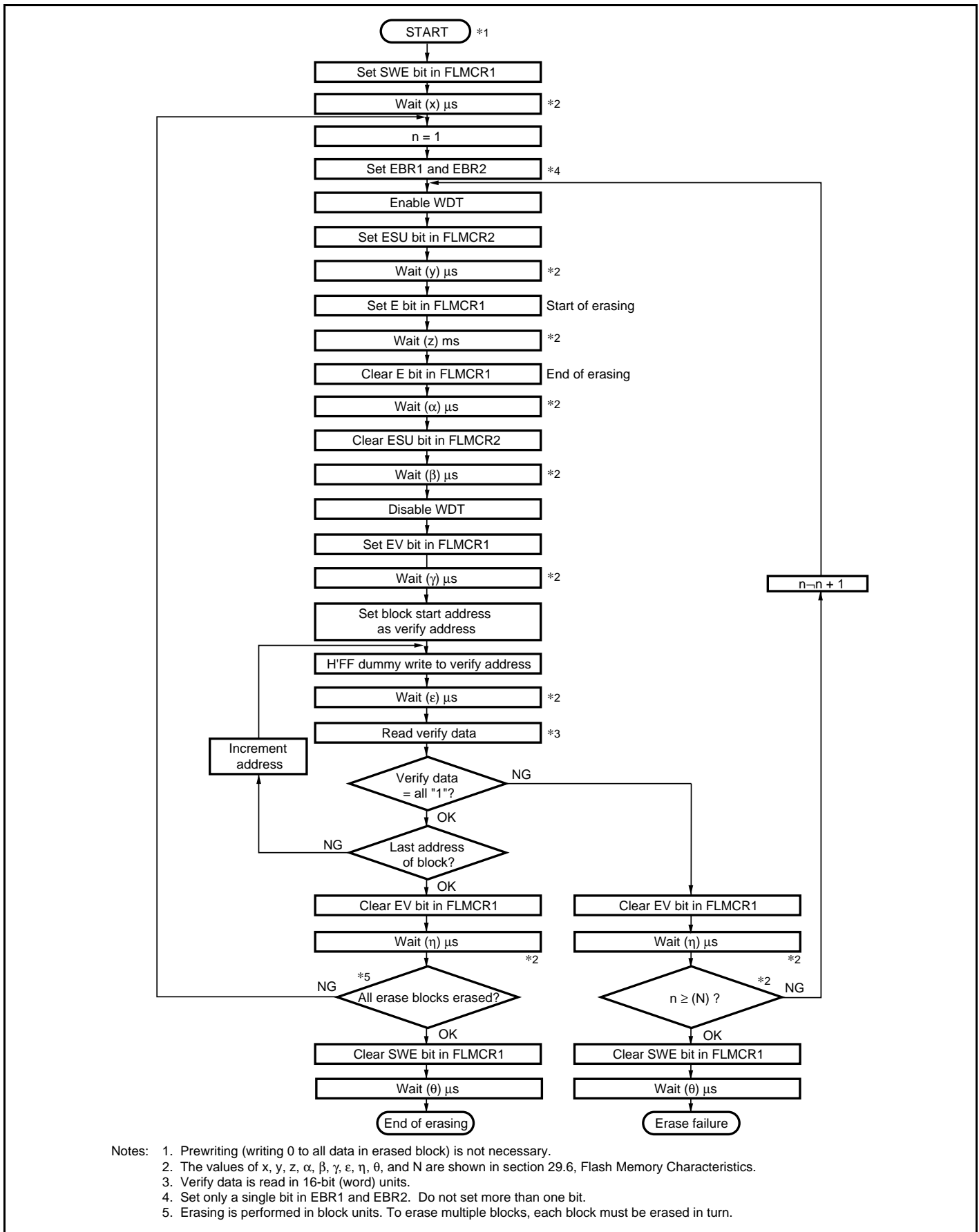


Figure 24.10 Erase/Eraser-Verify Flowchart

## 24.9 Program/Erase Protection

There are three kinds of flash memory program/erase protection: hardware protection, software protection, and error protection.

### 24.9.1 Hardware Protection

Hardware protection is a state in which programming/erasing of flash memory is forcibly disabled or aborted because of a low level input to the FEW pin, by a reset (including WDT overflow reset), or a transition to hardware standby mode, software standby mode, sub-active mode, sub-sleep mode, or watch mode. Flash memory control register 1 (FLMCR1), flash memory control register 2 (FLMCR2), and erase block registers 1 and 2 (EBR1 and EBR2) are initialized. In a reset via the  $\overline{\text{RES}}$  pin, the reset state is not entered unless the  $\overline{\text{RES}}$  pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the  $\overline{\text{RES}}$  pin low for the  $\overline{\text{RES}}$  pulse width specified in the AC Characteristics section.

### 24.9.2 Software Protection

Software protection can be implemented against programming/erasing of all flash memory blocks by clearing the SWE bit in FLMCR1 to 0. When software protection is in effect, setting the P or E bit in FLMCR1 does not cause a transition to program mode or erase mode. By setting the erase block registers 1 and 2 (EBR1 and EBR2), erase protection can be set for individual blocks. When EBR1 and EBR2 are set to H'00, erase protection is set for all blocks.

### 24.9.3 Error Protection

In error protection, an error is detected when the CPU's runaway occurs during flash memory programming/erasing, or operation is not performed in accordance with the program/erase algorithm, and the program/erase operation is aborted. Aborting the program/erase operation prevents damage to the flash memory due to overprogramming or overerasing.

When the following errors are detected during programming/erasing of flash memory, the FLER bit in FLMCR2 is set to 1, and the error protection state is entered.

- When the flash memory of is read during programming/erasing (including vector read and instruction fetch)
- Immediately after exception handling (excluding a reset) during programming/erasing
- When a SLEEP instruction is executed (transits to software standby mode, sleep mode, sub-active mode, sub-sleep mode, or watch mode) during programming/erasing
- CPU loses bus mastership during programming/erasing

The FLMCR1, FLMCR2, EBR1, and EBR2 settings are retained, but program mode or erase mode is aborted at the point at which the error occurred. Program mode or erase mode cannot be entered by setting the P or E bit to 1. However, because the PV and EV bit settings are retained, a transition to verify mode can be made. The error protection state can be cancelled by a reset or in hardware standby mode.

## 24.10 Interrupts during Flash Memory Programming/Erasing

In order to give the highest priority to programming/erasing operations, disable all interrupts including NMI input during flash memory programming/erasing (the P or E bit in FLMCR1 is set to 1) or boot program execution\*<sup>1</sup>.

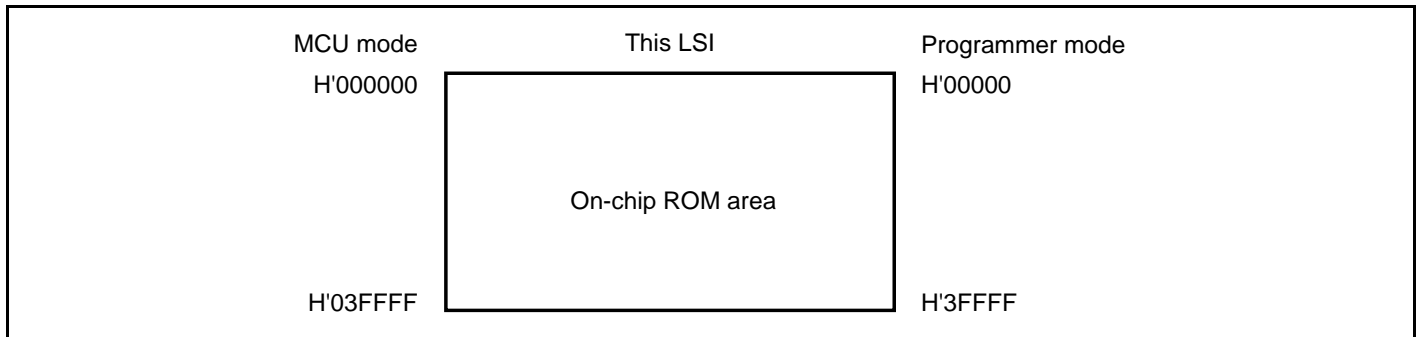
1. If an interrupt is generated during programming/erasing, operation in accordance with the program/erase algorithm is not guaranteed.
2. CPU runaway may occur because normal vector reading cannot be performed in interrupt exception handling during programming/erasing\*<sup>2</sup>.
3. If an interrupt occurs during boot program execution, the normal boot mode sequence cannot be executed.

- Notes:
1. Interrupt requests must be disabled inside and outside the CPU until the programming control program has completed programming.
  2. The vector may not be read correctly for the following two reasons:
    - If flash memory is read while being programmed or erased (while the P or E bit in FLMCR1 is set to 1), correct read data will not be obtained (undefined values will be returned).
    - If the interrupt entry in the vector table has not been programmed yet, interrupt exception handling will not be executed correctly.

## 24.11 Programmer Mode

In programmer mode, the on-chip flash memory can be programmed/erased by a PROM programmer via a socket adapter, just like for a discrete flash memory. Use a PROM programmer that supports the Renesas 256-kbyte flash memory on-chip MCU device. Figure 24.11 shows a memory map in programmer mode.

Note: \* For this LSI, set the programming voltage of the PROM programmer to 3.3V.



**Figure 24.11 Memory Map in Programmer Mode**

## 24.12 Usage Notes

The following lists notes on the use of on-board programming modes and programmer mode.

1. Perform programming/erasing with the specified voltage and timing.  
If a voltage higher than the rated voltage is applied, the product may be fatally damaged. Use a PROM programmer that supports the Renesas 256-kbyte flash memory on-chip MCU device at 3.3 V. Do not set the programmer to HN28F101 or the programming voltage to 5.0 V. Use only the specified socket adapter. If other adapters are used, the product may be damaged.
2. Notes on power on/off  
At powering on or off the Vcc power supply, fix the  $\overline{\text{RES}}$  pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.
3. Perform flash memory programming/erasing in accordance with the recommended algorithm  
In the recommended algorithm, flash memory programming/erasing can be performed without subjecting this LSI to voltage stress or sacrificing program data reliability. When setting the P or E bit in FLMCR1 to 1, set the watchdog timer against program runaway.
4. Do not set/clear the SWE bit during program execution in the flash memory.  
Do not set/clear the SWE bit during program execution in the flash memory. An interval of at least 100  $\mu\text{s}$  is necessary between program execution or data reading in flash memory and SWE bit clearing. When the SWE bit is set to 1, flash memory data can be modified, however, flash memory data can be read only in program-verify or erase-verify mode. Do not access the flash memory for a purpose other than verification during programming/erasing. Do not clear the SWE bit during programming, erasing, or verifying.
5. Do not use interrupts during flash memory programming/erasing  
In order to give the highest priority to programming/erasing operation, disable all interrupts including NMI input when the flash memory is programmed or erased.

6. Do not perform additional programming. Programming must be performed in the erased state. Program the area with 128-byte programming-unit blocks in on-board programming or programmer mode only once.  
Perform programming in the state where the programming-unit block is fully erased.
7. Ensure that the PROM programmer is correctly attached before programming.  
If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.
8. Do not touch the socket adapter or LSI while programming.  
Touching either of these can cause contact faults and write errors.
9. Allocate data or programs to be written to the flash memory to addresses higher than that of the external interrupt vector table.  
To write data or programs to the flash memory, data or programs must be allocated to addresses higher than that of the external interrupt vector table (H'00040 in advanced mode and H'0020 in normal mode) and H'FF must be written to the areas that are reserved for the system in the exception handling vector table.
10. Write H'FF to the entire key code area for reading in programmer mode.  
If data other than H'FF is written to the key code area (advanced mode: H'00003C to H'00003F, normal mode: H'001E to H'001F) of flash memory, reading cannot be performed in programmer mode. (In this case, data is read as H'00. Rewrite is possible after erasing the data.) For reading in programmer mode, make sure to write H'FF to the entire key code area.  
If data other than H'FF is to be written to the key code area in programmer mode, a verification error will occur unless a software countermeasure has been taken for the PROM programmer.



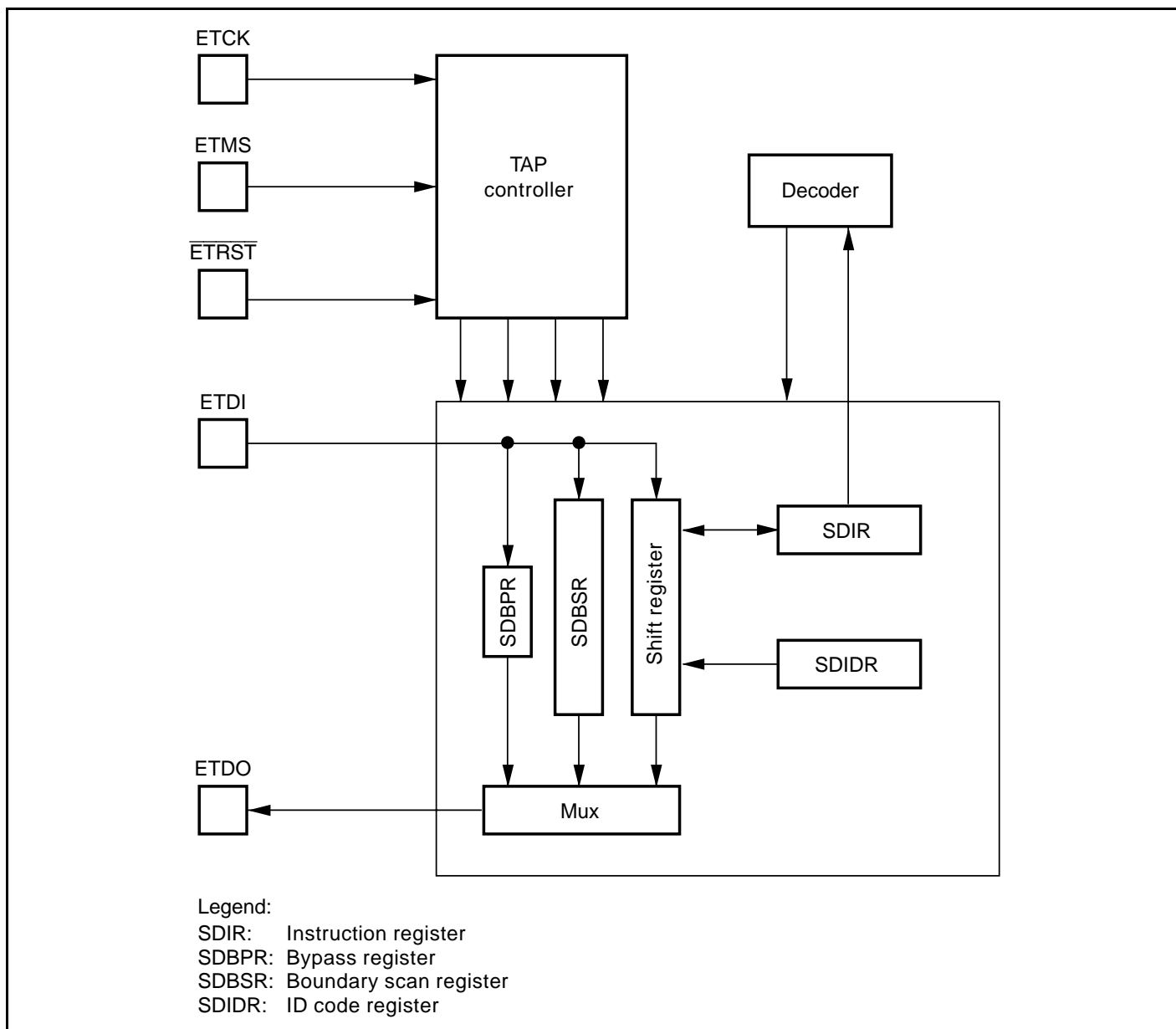
## Section 25 User Debug Interface (H-UDI)

The user debug interface (H-UDI) provides a boundary scan function using the JTAG (Joint Test Action Group, IEEE Std 1149.1, IEEE Standard Test Access Port and Boundary-Scan Architecture) and a pin-compatible serial interface. The H-UDI performs serial transfer by means of external signal control.

### 25.1 Features

The H-UDI has the following features conforming to the IEEE1149.1 standard.

- Five test pins (ETCK, ETDI, ETDO, ETMS, and  $\overline{\text{ETRST}}$ )
  - TAP controller
  - Six instructions
    - BYPASS mode
    - EXTEST mode
    - SAMPLE/PRELOAD mode
    - CLAMP mode
    - HIGHZ mode
    - IDCODE mode
- (These instructions are test modes corresponding to IEEE 1149.1.)



**Figure 25.1 Block Diagram of H-UDI**

## 25.2 Input/Output Pins

Table 25.1 shows the H-UDI pin configuration.

**Table 25.1 Pin Configuration**

Pin Name	Abbreviation	I/O	Function
Test clock	ETCK	Input	<p>Test clock input</p> <p>Provides an independent clock supply to the H-UDI. As the clock input to the ETCK pin is supplied directly to the H-UDI, a clock waveform with a duty cycle close to 50% should be input. For details, see section 29, Electrical Characteristics. If there is no input, the ETCK pin is fixed to 1 by an internal pull-up.</p>
Test mode select	ETMS	Input	<p>Test mode select input</p> <p>Sampled on the rise of the ETCK pin. The ETMS pin controls the internal state of the TAP controller. If there is no input, the ETMS pin is fixed to 1 by an internal pull-up.</p>
Test data input	ETDI	Input	<p>Serial data input</p> <p>Performs serial input of instructions and data for H-UDI registers. ETDI is sampled on the rise of the ETCK pin. If there is no input, the ETDI pin is fixed to 1 by an internal pull-up.</p>
Test data output	ETDO	Output	<p>Serial data output</p> <p>Performs serial output of instructions and data from H-UDI registers. Transfer is performed in synchronization with the ETCK pin. If there is no output, the ETDO pin goes to the high-impedance state.</p>
Test reset	$\overline{\text{ETRST}}$	Input	<p>Test reset input signal</p> <p>Initializes the H-UDI asynchronously. If there is no input, the <math>\overline{\text{ETRST}}</math> pin is fixed to 1 by an internal pull-up.</p>

## 25.3 Register Descriptions

The H-UDI has the following registers.

- Instruction register (SDIR)
- Bypass register (SDBPR)
- Boundary scan register (SDBSR)
- ID code register (SDIDR)

Instructions can be input to the instruction register (SDIR) by serial transfer from the test data input pin (ETDI). Data from SDIR can be output via the test data output pin (ETDO). The bypass register (SDBPR) is a 1-bit register to which the ETDI and ETDO pins are connected in BYPASS, CLAMP, or HIGHZ mode. The boundary scan register (SDBSR) is a 215-bit register to which the ETDI and ETDO pins are connected in SAMPLE/PRELOAD or EXTEST mode. The ID code register (SDIDR) is a 32-bit register; a fixed code can be output via the ETDO pin in IDCODE mode. All registers cannot be accessed directly by the CPU.

Table 25.2 shows the kinds of serial transfer possible with each H-UDI register.

**Table 25.2 H-UDI Register Serial Transfer**

Register	Serial Input	Serial Output
SDIR	Possible	Possible
SDBPR	Possible	Possible
SDBSR	Possible	Possible
SDIDR	Impossible	Possible

### 25.3.1 Instruction Register (SDIR)

SDIR is a 32-bit read-only register. H-UDI instructions can be transferred to SDIR by serial input from the ETDI pin. SDIR can be initialized when the  $\overline{\text{ETRST}}$  pin is low or the TAP controller is in the Test-Logic-Reset state, but is not initialized by a reset or in standby mode.

Only 4-bit instructions can be transferred to SDIR. If an instruction exceeding 4 bits is input, the last 4 bits of the serial data will be stored in SDIR.

Bit	Bit Name	Initial Value	R/W	Description
31	TS3	1	R	Test Set Bits
30	TS2	1	R	0000: EXTEST mode
29	TS1	1	R	0001: Setting prohibited
28	TS0	0	R	0010: CLAMP mode 0011: HIGHZ mode 0100: SAMPLE/PRELOAD mode 0101: Setting prohibited :          : 1101: Setting prohibited 1110: IDCODE mode (Initial value) 1111: BYPASS mode
27 to 14	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
13	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
12 to 10	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
9	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
8 to 1	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
0	—	1	R	Reserved This bit is always read as 1 and cannot be modified.

### 25.3.2 Bypass Register (SDBPR)

SDBPR is a 1-bit shift register. In BYPASS, CLAMP, or HIGHZ mode, SDBPR is connected between the ETDI and ETDO pins.

### 25.3.3 Boundary Scan Register (SDBSR)

SDBSR is a shift register provided on the PAD for controlling the I/O terminals of this LSI.

Using EXTEST mode or SAMPLE/PRELOAD mode, a boundary scan test conforming to the IEEE1149.1 standard can be performed.

Table 25.3 shows the relationship between the terminals of this LSI and the boundary scan register.

**Table 25.3 Correspondence between Pins and Boundary Scan Register**

Pin No.	Pin Name	Input/Output	Bit No.
<b>from ETDI</b>			
4	$\overline{\text{MD2}}$	Input	214
5	MD1	Input	213
6	MD0	Input	212
7	NMI	Input	211
13	P51	Input	210
		Enable	209
		Output	208
14	P50	Input	207
		Enable	206
		Output	205
16	P97	Input	204
		Enable	203
		Output	202
17	P96	Input	201
		Enable	200
		Output	199
18	P95	Input	198
		Enable	197
		Output	196
19	P94	Input	195
		Enable	194
		Output	193
20	P56	Input	192
		Enable	191
		Output	190
21	P57	Input	189
		Enable	188
		Output	187

<b>Pin No.</b>	<b>Pin Name</b>	<b>Input/Output</b>	<b>Bit No.</b>
22	P93	Input	186
		Enable	185
		Output	184
23	P92	Input	183
		Enable	182
		Output	181
24	P91	Input	180
		Enable	179
		Output	178
25	P90	Input	177
		Enable	176
		Output	175
26	P60	Input	174
		Enable	173
		Output	172
27	P61	Input	171
		Enable	170
		Output	169
28	P62	Input	168
		Enable	167
		Output	166
29	P63	Input	165
		Enable	164
		Output	163
32	P64	Input	162
		Enable	161
		Output	160
33	P65	Input	159
		Enable	158
		Output	157



Pin No.	Pin Name	Input/Output	Bit No.
34	P66	Input	156
		Enable	155
		Output	154
35	P67	Input	153
		Enable	152
		Output	151
38	USDP	Input	150
		Enable*	149
		Output	148
39	USDM	Input	147
		Enable*	146
		Output	145
40	P72	Input	144
41	P73	Input	143
42	P74	Input	142
43	P75	Input	141
44	P76	Input	140
45	P77	Input	139
47	PA1	Input	138
		Enable	137
		Output	136
48	PA0	Input	135
		Enable	134
		Output	133
49	P54	Input	132
		Enable	131
		Output	130
50	P55	Input	129
		Enable	128
		Output	127

Pin No.	Pin Name	Input/Output	Bit No.
51	P40	Input	126
		Enable	125
		Output	124
52	P41	Input	123
		Enable	122
		Output	121
53	P42	Input	120
		Enable	119
		Output	118
54	P43	Input	117
		Enable	116
		Output	115
55	P44	Input	114
		Enable	113
		Output	112
56	P45	Input	111
		Enable	110
		Output	109
57	P46	Input	108
		Enable	107
		Output	106
58	P47	Input	105
		Enable	104
		Output	103
60	P27	Input	102
		Enable	101
		Output	100
61	P26	Input	99
		Enable	98
		Output	97

Pin No.	Pin Name	Input/Output	Bit No.
62	P25	Input	96
		Enable	95
		Output	94
63	P24	Input	93
		Enable	92
		Output	91
64	P23	Input	90
		Enable	89
		Output	88
65	P22	Input	87
		Enable	86
		Output	85
66	P21	Input	84
		Enable	83
		Output	82
67	P20	Input	81
		Enable	80
		Output	79
72	P17	Input	78
		Enable	77
		Output	76
73	P16	Input	75
		Enable	74
		Output	73
74	P15	Input	72
		Enable	71
		Output	70
75	P14	Input	69
		Enable	68
		Output	67

<b>Pin No.</b>	<b>Pin Name</b>	<b>Input/Output</b>	<b>Bit No.</b>
76	P13	Input	66
		Enable	65
		Output	64
77	P12	Input	63
		Enable	62
		Output	61
78	P11	Input	60
		Enable	59
		Output	58
79	P10	Input	57
		Enable	56
		Output	55
80	P87	Input	54
		Enable	53
		Output	52
81	P86	Input	51
		Enable	50
		Output	49
82	P30	Input	48
		Enable	47
		Output	46
83	P31	Input	45
		Enable	44
		Output	43
84	P32	Input	42
		Enable	41
		Output	40
85	P33	Input	39
		Enable	38
		Output	37

Pin No.	Pin Name	Input/Output	Bit No.
86	P34	Input	36
		Enable	35
		Output	34
87	P35	Input	33
		Enable	32
		Output	31
88	P36	Input	30
		Enable	29
		Output	28
89	P37	Input	27
		Enable	26
		Output	25
90	P85	Input	24
		Enable	23
		Output	22
91	P84	Input	21
		Enable	20
		Output	19
93	P80	Input	18
		Enable	17
		Output	16
94	P81	Input	15
		Enable	14
		Output	13
95	P82	Input	12
		Enable	11
		Output	10
96	P83	Input	9
		Enable	8
		Output	7

Pin No.	Pin Name	Input/Output	Bit No.
97	P52	Input	6
		Enable	5
		Output	4
98	P53	Input	3
		Enable	2
		Output	1
99	FWE	Input	0

#### to ETDO

Notes: The enable signals are active-high. When an enable signal is driven high, the corresponding pin is driven with the output value.

- \* If either the enable signal for the USDP pin or that for the USDM pin is driven high, both pins are driven by the output values.

### 25.3.4 ID Code Register (SDIDR)

SDIDR is a 32-bit register. In IDCODE mode, SDIDR can output H'0018200F, which is a fixed code, from ETDO. However, no serial data can be written to SDIDR via ETDI.

31 28	27	12	11	1	0
0000	0000 0001 1000 0010		0000 0000 111		1
Version (4 bits)	Part Number (16 bits)		Manufacture Identify (11 bits)		Fixed Code (1 bit)

## 25.4 Operation

### 25.4.1 TAP Controller State Transitions

Figure 25.2 shows the internal states of the TAP controller. State transitions basically conform to the IEEE1149.1 standard.

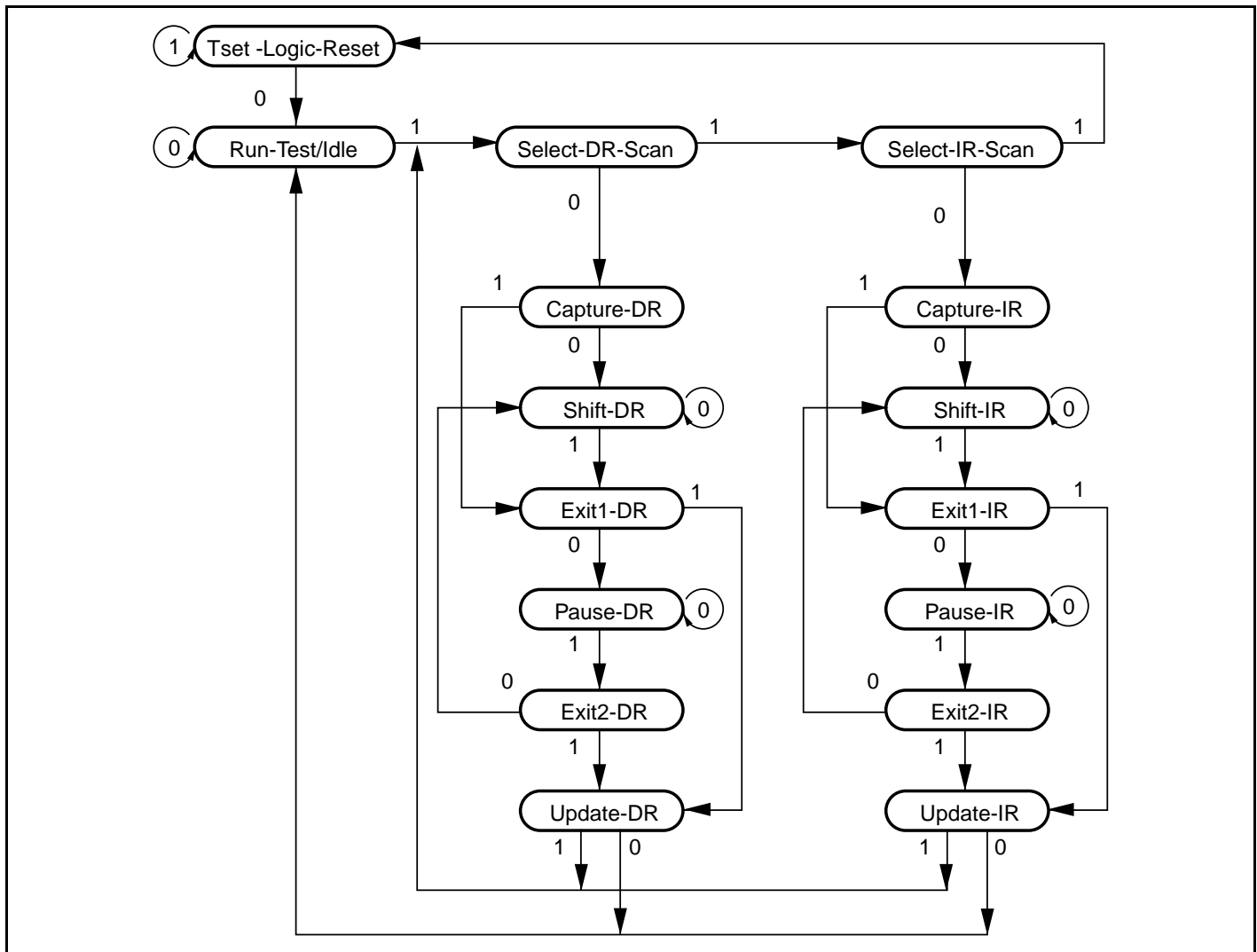


Figure 25.2 TAP Controller State Transitions

### 25.4.2 H-UDI Reset

The H-UDI can be reset in two ways.

- The H-UDI is reset when the  $\overline{\text{ETRST}}$  pin is held at 0.
- When  $\overline{\text{ETRST}} = 1$ , the H-UDI can be reset by inputting at least five ETCK clock cycles while  $\text{ETMS} = 1$ .

## 25.5 Boundary Scan

The H-UDI pins can be placed in the boundary scan mode stipulated by the IEEE1149.1 standard by setting a command in SDIR.

### 25.5.1 Supported Instructions

This LSI supports the three essential instructions defined in the IEEE1149.1 standard (BYPASS, SAMPLE/PRELOAD, and EXTEST) and optional instructions (CLAMP, HIGHZ, and IDCODE).

- **BYPASS** [Instruction code: B'1111]

The BYPASS instruction is an instruction that operates the bypass register. This instruction shortens the shift path to speed up serial data transfer involving other chips on the printed circuit board. While this instruction is being executed, the test circuit has no effect on the system circuits.

- **SAMPLE/PRELOAD** [Instruction code: B'0100]

The SAMPLE/PRELOAD instruction inputs values from this LSI internal circuitry to the boundary scan register, outputs values from the scan path, and loads data onto the scan path. When this instruction is being executed, this LSI's input pin signals are transmitted directly to the internal circuitry, and internal circuit values are directly output externally from the output pins. This LSI system circuits are not affected by execution of this instruction.

In a SAMPLE operation, a snapshot of a value to be transferred from an input pin to the internal circuitry, or a value to be transferred from the internal circuitry to an output pin, is latched into the boundary scan register and read from the scan path. Snapshot latching does not affect normal operation of this LSI.

In a PRELOAD operation, an initial value is set in the parallel output latch of the boundary scan register from the scan path prior to the EXTEST instruction. Without a PRELOAD operation, when the EXTEST instruction was executed an undefined value would be output from the output pin until completion of the initial scan sequence (transfer to the output latch) (with the EXTEST instruction, the parallel output latch value is constantly output to the output pin).

- **EXTEST** [Instruction code: B'0000]

The EXTEST instruction is provided to test external circuitry when this LSI is mounted on a printed circuit board. When this instruction is executed, output pins are used to output test data (previously set by the SAMPLE/PRELOAD instruction) from the boundary scan register to the printed circuit board, and input pins are used to latch test results into the boundary scan register from the printed circuit board. If testing is carried out by using the EXTEST instruction N times, the Nth test data is scanned in when test data (N-1) is scanned out.



Data loaded into the output pin boundary scan register in the Capture-DR state is not used for external circuit testing (it is replaced by a shift operation).

- CLAMP [Instruction code: B'0010]

When the CLAMP instruction is enabled, the output pin outputs the value of the boundary scan register that has been previously set by the SAMPLE/PRELOAD instruction. While the CLAMP instruction is enabled, the state of the boundary scan register maintains the previous state regardless of the state of the TAP controller.

A bypass register is connected between the ETDI and ETDO pins. The related circuit operates in the same way when the BYPASS instruction is enabled.

- HIGHZ [Instruction code: B'0011]

When the HIGHZ instruction is enabled, all output pins enter a high-impedance state. While the HIGHZ instruction is enabled, the state of the boundary scan register maintains the previous state regardless of the state of the TAP controller.

A bypass register is connected between the ETDI and ETDO pins. The related circuit operates in the same way when the BYPASS instruction is enabled.

- IDCODE [Instruction code: B'1110]

When the IDCODE instruction is enabled, the value of the ID code register is output from the ETDO pin with LSB first when the TAP controller is in the Shift-DR state. While the IDCODE instruction is being executed, the test circuit does not affect the system circuit.

When the TAP controller is in the Test-Logic-Reset state, the instruction register is initialized to the IDCODE instruction.

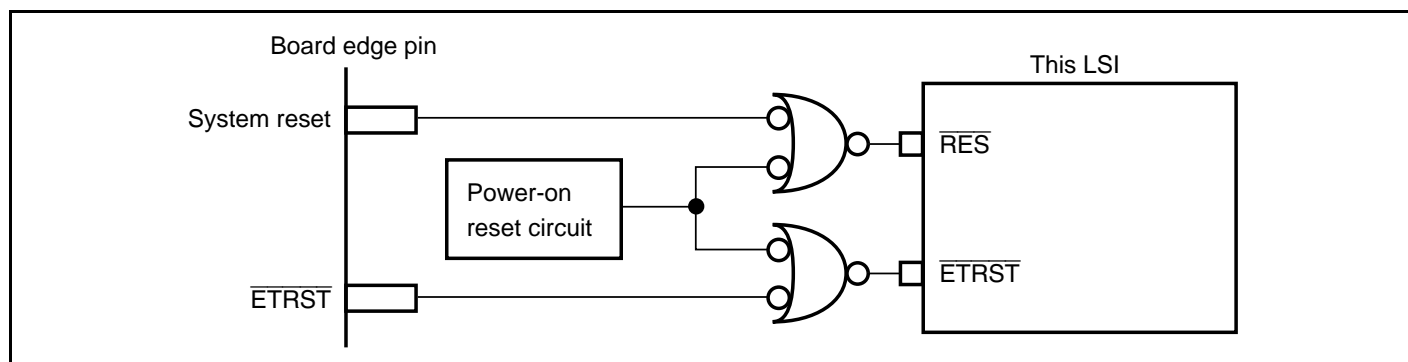
## 25.5.2 Notes

1. Boundary scan mode does not cover power-supply-related pins (VCC, VCL, VSS, AVCC/DrVCC, AVSS/DrVSS, and AVref).
2. Boundary scan mode covers clock-related pins (EXTAL, XTAL, X1, and X2).
3. Boundary scan mode does not cover reset- and standby-related pins ( $\overline{\text{RES}}$ ,  $\overline{\text{STBY}}$ , and  $\overline{\text{RESO}}$ ).
4. Boundary scan mode does not cover H-UDI-related pins (ETCK, ETDI, ETDO, ETMS, and  $\overline{\text{ETRST}}$ ).
5. Fix the  $\overline{\text{MD2}}$  pin high.

## 25.6 Usage Notes

1. A reset must always be executed by driving the  $\overline{\text{ETRST}}$  pin to 0, regardless of whether or not the H-UDI is to be activated. The  $\overline{\text{ETRST}}$  pin must be held low for 20 ETCK clock cycles. For details, see section 29, Electrical Characteristics. To activate the H-UDI after a reset, drive the  $\overline{\text{ETRST}}$  pin to 1 and specify the ETCK, ETMS, and ETDI pins to any value. If the H-UDI is not to be activated, drive the  $\overline{\text{ETRST}}$ , ETCK, ETMS, and ETDI pins to 1 or the high-impedance state. These pins are internally pulled up and are noted in standby mode.
2. The following must be considered when the power-on reset signal is applied to the  $\overline{\text{ETRST}}$  pin.
  - The reset signal must be applied at power-on.
  - To prevent the LSI system operation from being affected by the  $\overline{\text{ETRST}}$  pin of the board tester, circuits must be separated.
  - Alternatively, to prevent the  $\overline{\text{ETRST}}$  pin of the board tester from being affected by the LSI system reset, circuits must be separated.

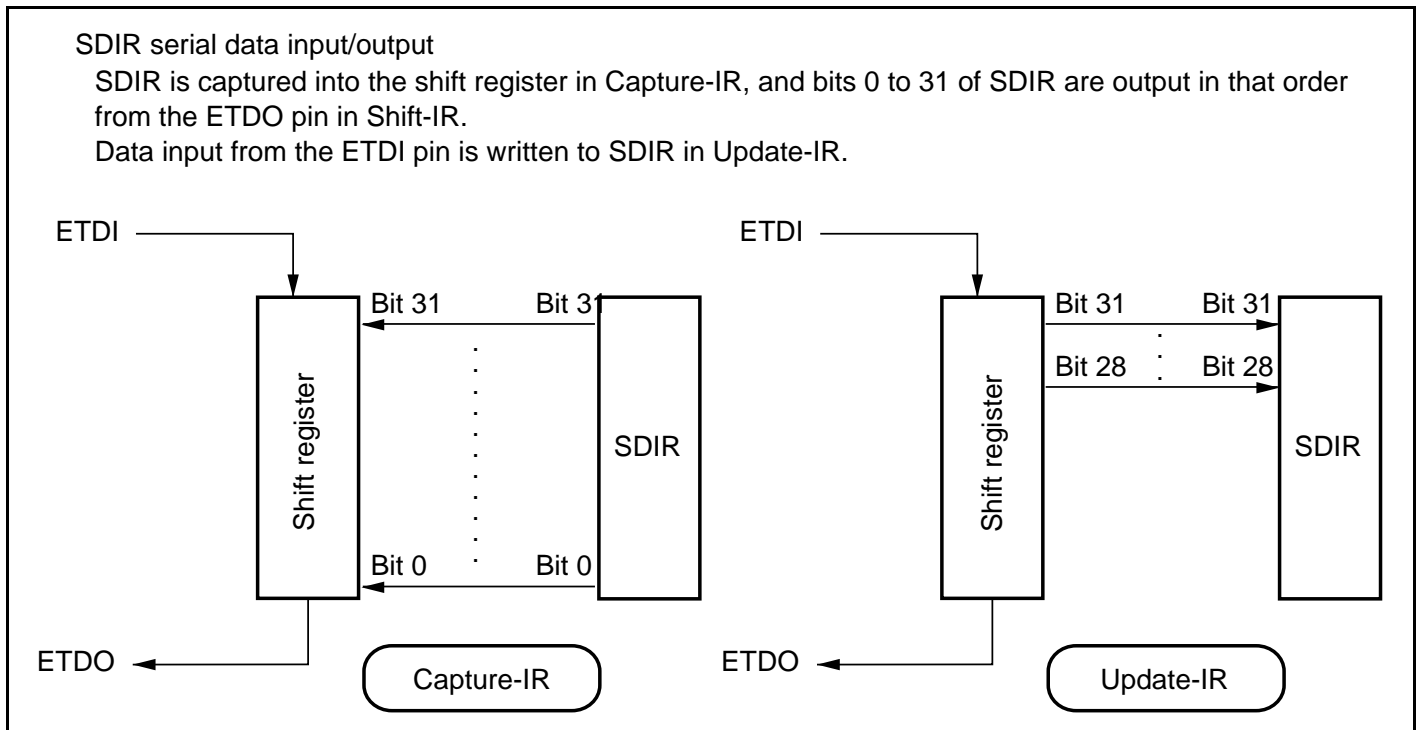
Figure 25.3 shows a design example of the reset signal circuit wherein no reset signal interference occurs.



**Figure 25.3 Reset Signal Circuit Without Reset Signal Interference**

3. The registers are not initialized in standby mode. If the  $\overline{\text{ETRST}}$  pin is set to 0 in standby mode, IDCODE mode will be entered.
4. The frequency of the ETCK pin must be lower than that of the system clock. For details, see section 29, Electrical Characteristics.
5. Data input/output in serial data transfer starts from the LSB. Figure 25.4 shows examples of serial data input/output.
6. When data that exceeds the number of bits of the register connected between the ETDI and ETDO pins is serially transferred, the serial data that exceeds the number of register bits and output from the ETDO pin is the same as that input from the ETDI pin.
7. If the H-UDI serial transfer sequence is disrupted, the  $\overline{\text{ETRST}}$  pin must be reset. Transfer should then be retried, regardless of the transfer operation.

8. If a pin with a pull-up function is sampled while its pull-up function is enabled, 1 can be detected at the corresponding input scan register. In this case, the corresponding enable scan register should be cleared to 0.
9. If a pin with an open-drain function is sampled while its open-drain function is enabled and its corresponding output scan register is 1, 0 can be detected at the corresponding enable scan register.

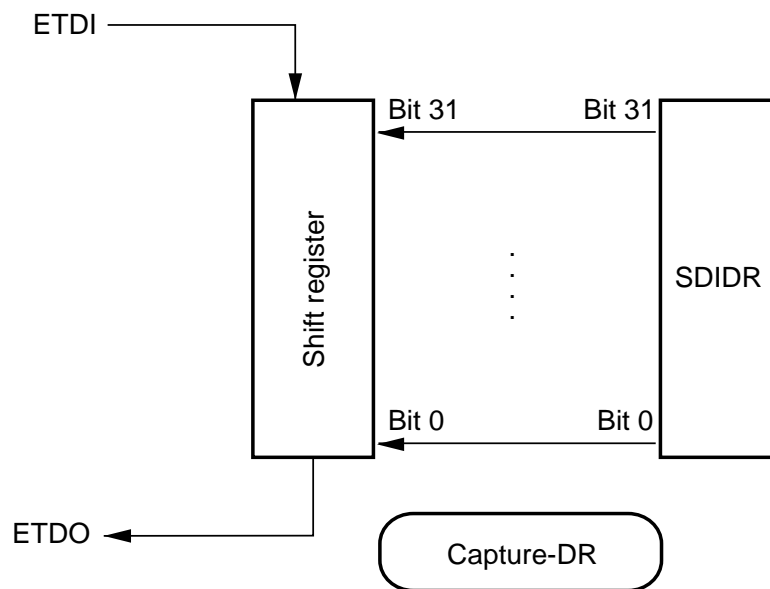


**Figure 25.4 Serial Data Input/Output (1)**

## SDIDR serial data input/output

SDIDR is captured into the shift register in Capture-DR in IDCODE mode, and bits 0 to 31 of SDIDR are output in that order from the ETDO pin in Shift-DR.

Data input from the ETDI pin is written to any register in Update-DR.

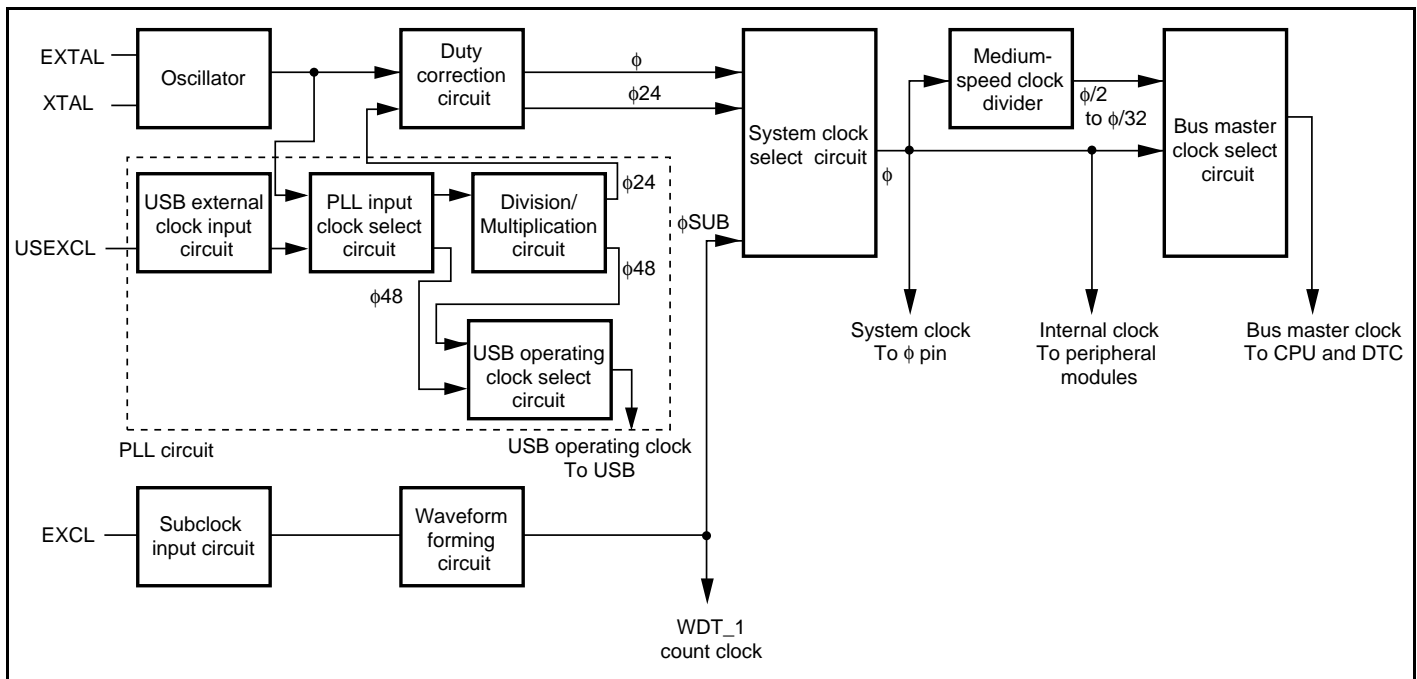


**Figure 25.4 Serial Data Input/Output (2)**

## Section 26 Clock Pulse Generator

This LSI incorporates a clock pulse generator which generates the system clock ( $\phi$ ), internal clock, bus master clock, and subclock ( $\phi$ SUB). The clock pulse generator consists of an oscillator, duty correction circuit, system clock select circuit, medium-speed clock divider, bus master clock select circuit, subclock input circuit, and waveform forming circuit. Figure 26.1 shows a block diagram of the clock pulse generator.

This LSI also incorporates a PLL (Phase Locked Loop) circuit that generates a 48-MHz clock ( $\phi$ 48) and a 24-MHz clock ( $\phi$ 24) as the USB operating clock. The 24-MHz clock can be used as the system clock ( $\phi$ ) by inputting it to the duty correction circuit instead of the oscillator output to which the EXTAL and XTAL pins are input. The PLL circuit consists of a USB external clock input circuit, PLL input clock select circuit, division/multiplication circuit, and USB operating clock select circuit.



**Figure 26.1 Block Diagram of Clock Pulse Generator**

The bus master clock is selected as either high-speed mode or medium-speed mode by software according to the settings of the SCK2 to SCK0 bits in the standby control register. Use of the medium-speed clock ( $\phi/2$  to  $\phi/32$ ) may be limited during CPU operation and when accessing the internal memory of the CPU. The operation speed of the DTC and RFU and the external space access cycle are thus stabilized regardless of the setting of medium-speed mode. For details on the standby control register, see section 27.1.1, Standby Control Register (SBYCR).

The subclock input is controlled by software according to the EXCLE bit setting in the low power control register. For details on the low power control register, see section 27.1.2, Low-Power Control Register (LPWRCR).

## 26.1 Oscillator

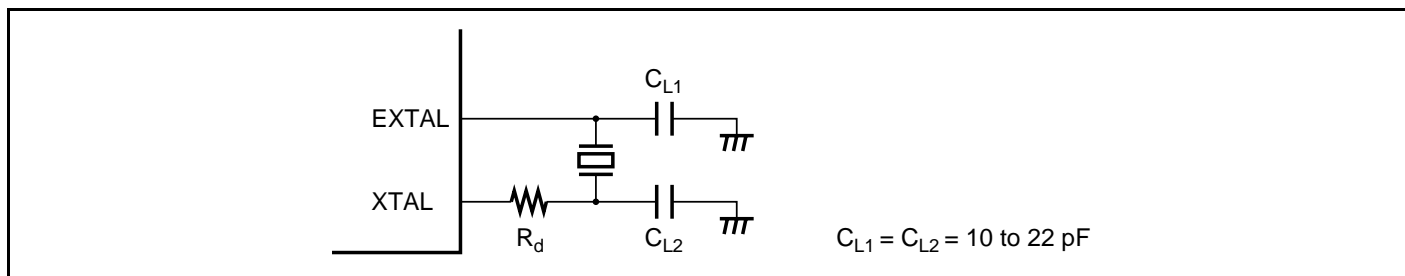
Clock pulses can be supplied either by connecting a crystal resonator or by providing external clock input.

### 26.1.1 Connecting a Crystal Oscillator

Figure 26.2 shows a typical method of connecting a crystal resonator. An appropriate damping resistance  $R_d$ , given in table 26.1, should be used. An AT-cut parallel-resonance crystal resonator should be used.

Figure 26.3 shows the equivalent circuit of a crystal resonator. A crystal resonator having the characteristics given in table 26.2 should be used.

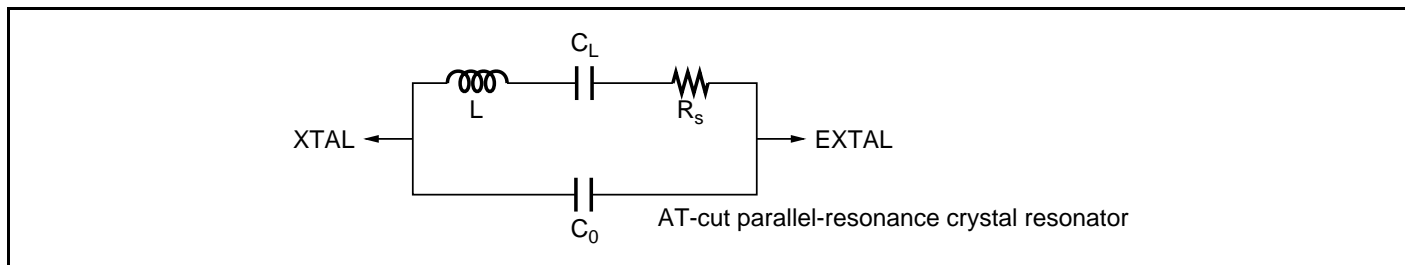
A crystal resonator with frequency identical to that of the system clock ( $\phi$ ) should be used.



**Figure 26.2 Typical Connection to Crystal Resonator**

**Table 26.1 Damping Resistance Values**

Frequency (MHz)	5	8	10	12	16	20	25
$R_d$ ( $\Omega$ )	300	200	0	0	0	0	0



**Figure 26.3 Equivalent Circuit of Crystal Resonator**

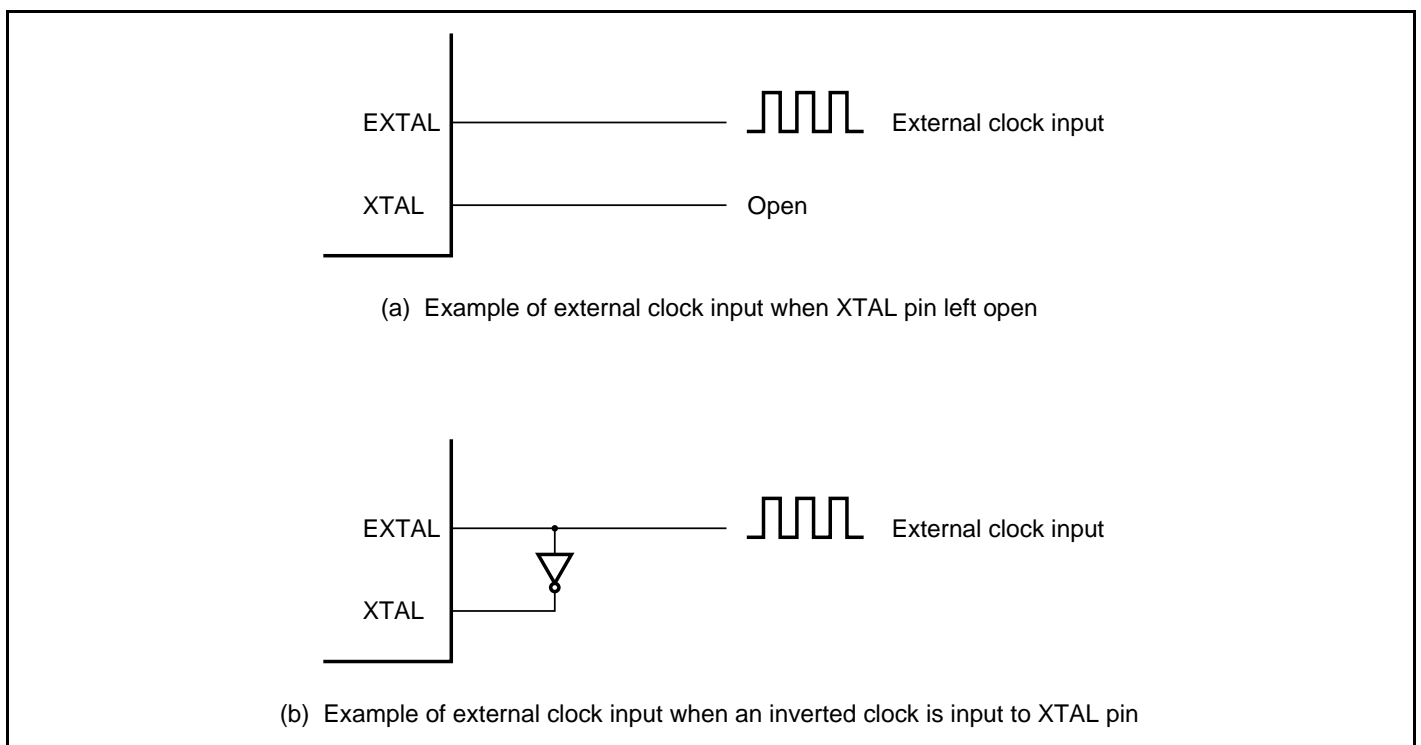
**Table 26.2 Crystal Resonator Parameters**

Frequency(MHz)	5	8	10	12	16	20	25
$R_s$ (max) ( $\Omega$ )	100	80	70	60	50	40	30
$C_0$ (max) (pF)	7	7	7	7	7	7	7

### 26.1.2 External Clock Input Method

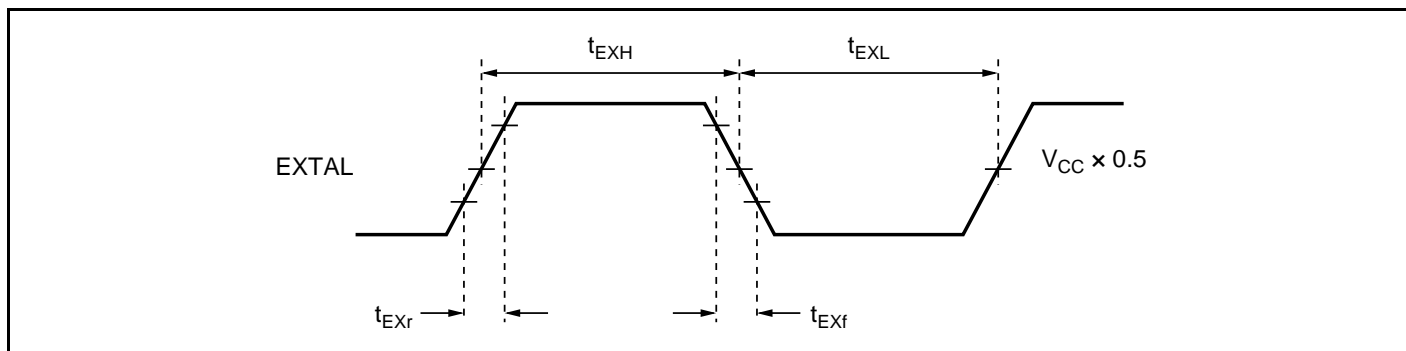
Figure 26.4 shows a typical method of connecting an external clock signal. To leave the XTAL pin open, incidental capacitance should be 10 pF or less.

To input an inverted clock to the XTAL pin, the external clock should be set to high in standby mode, subactive mode, subsleep mode, and watch mode. External clock input conditions are shown in table 26.3. The frequency of the external clock should be the same as that of the system clock ( $\phi$ ).

**Figure 26.4 Example of External Clock Input**

**Table 26.3 External Clock Input Conditions**

Item	Symbol	VCC = 3.0 to 3.6 V		VCC = 2.7 to 3.6 V		Unit	Test Conditions
		Min	Max	Min	Max		
External clock input pulse width low level	$t_{EXL}$	15	—	20	—	ns	Figure 26.5
External clock input pulse width high level	$t_{EXH}$	15	—	20	—	ns	
External clock rising time	$t_{EXr}$	—	5	—	5	ns	
External clock falling time	$t_{EXf}$	—	5	—	5	ns	
Clock pulse width low level	$t_{CL}$	0.4	0.6	0.4	0.6	$t_{cyc}$	Figure 29.4
Clock pulse width high level	$t_{CH}$	0.4	0.6	0.4	0.6	$t_{cyc}$	

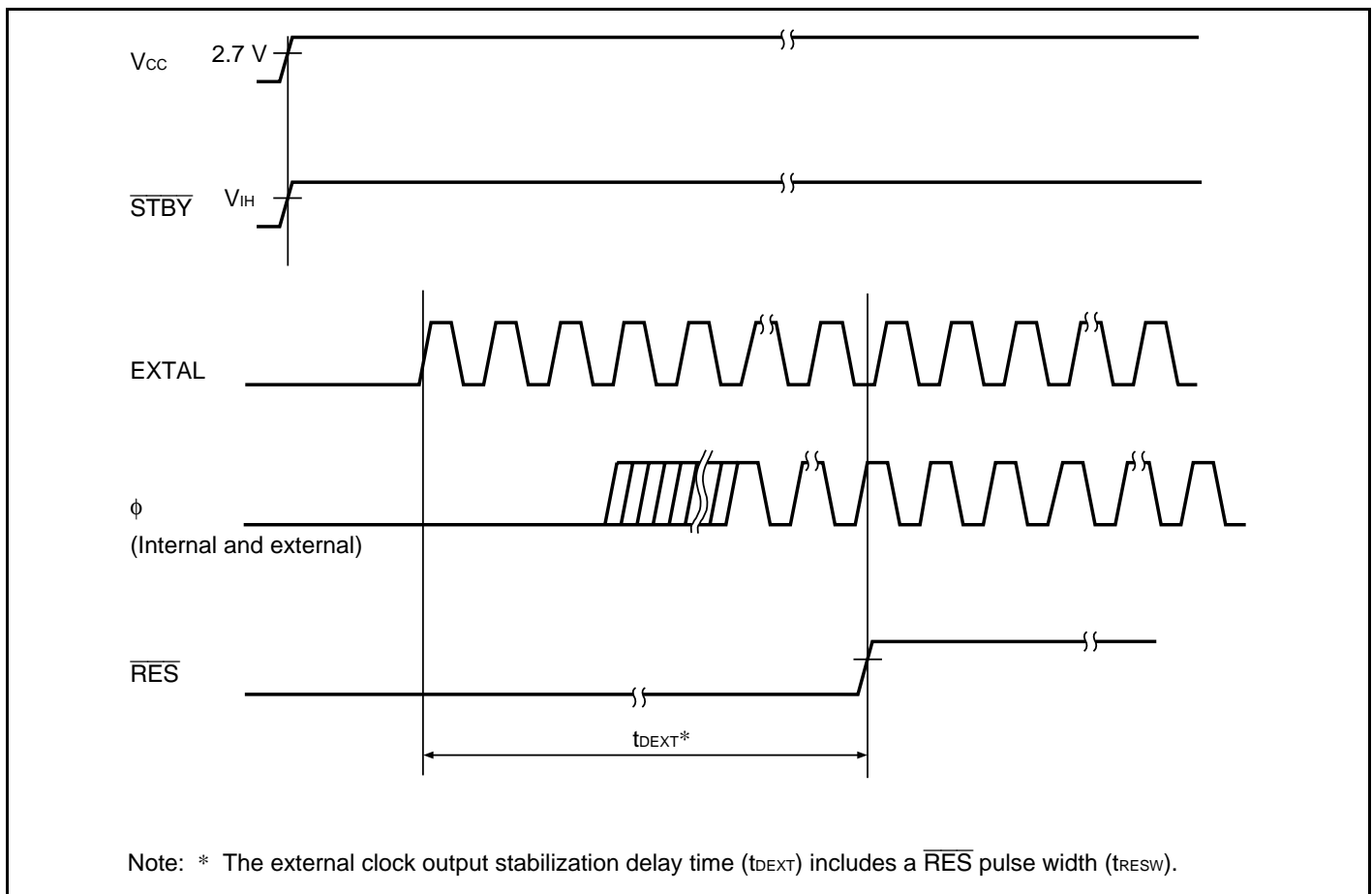
**Figure 26.5 External Clock Input Timing**

The oscillator and duty correction circuit have a function to adjust the waveform of the external clock input that is input to the EXTAL pin. When a specified clock signal is input to the EXTAL pin, internal clock signal output is determined after the external clock output stabilization delay time ( $t_{DEXT}$ ) has passed. As the clock signal output is not determined during the  $t_{DEXT}$  cycle, a reset signal should be set to low to hold it in reset state. Table 26.4 shows the external clock output stabilization delay time. Figure 26.6 shows the timing of the external clock output stabilization delay time.



**Table 26.4 External Clock Output Stabilization Delay Time**Condition:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = AV_{SS} = 0\text{ V}$ 

Item	Symbol	Min.	Max.	Unit	Remarks
External clock output stabilization delay time	$t_{\text{DEXT}}^*$	500	—	$\mu\text{s}$	Figure 26.6

Note: \*  $t_{\text{DEXT}}$  includes a  $\overline{\text{RES}}$  pulse width ( $t_{\text{RESW}}$ ).**Figure 26.6 Timing of External Clock Output Stabilization Delay Time**

## 26.2 Duty Correction Circuit

The duty correction circuit is valid when the oscillating frequency is 5 MHz or more. It corrects the duty of a clock that is output from the oscillator, and generates the system clock ( $\phi$ ).

## 26.3 Medium-Speed Clock Divider

The medium-speed clock divider divides the system clock ( $\phi$ ), and generates  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , and  $\phi/32$  clocks.

## 26.4 Bus Master Clock Select Circuit

The bus master clock select circuit selects a clock to supply the bus master with either the system clock ( $\phi$ ) or medium-speed clock ( $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , or  $\phi/32$ ) by the SCK2 to SCK0 bits in SBYCR.

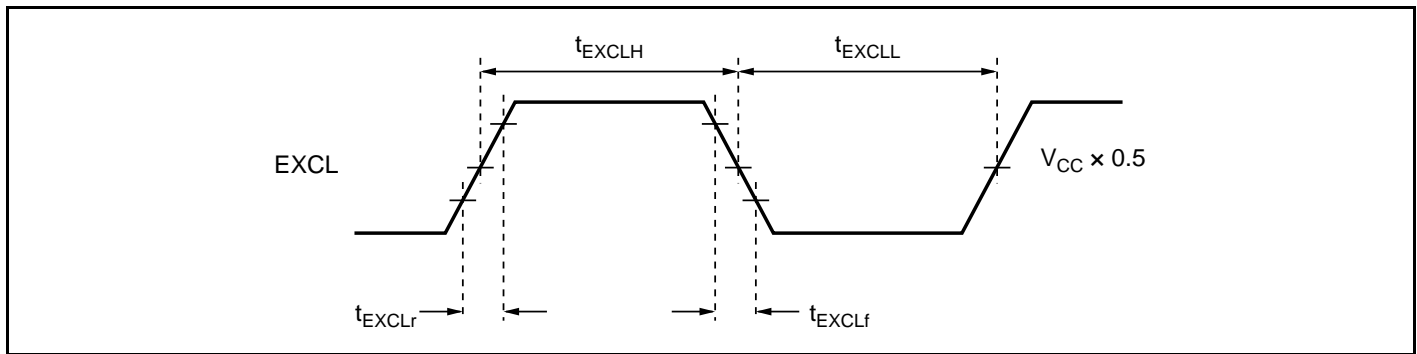
## 26.5 Subclock Input Circuit

The subclock input circuit controls subclock input from the EXCL pin. To use the subclock, a 32.768-kHz external clock should be input from the EXCL pin. At this time, the P96DDR bit in P9DDR should be cleared to 0, and the EXCLE bit in LPWRCR should be set to 1.

Subclock input conditions are shown in table 26.5. When the subclock is not used, subclock input should not be enabled.

**Table 26.5 Subclock Input Conditions**

Item	Symbol	VCC = 2.7 to 3.6 V			Unit	Measurement Condition
		Min	Typ	Max		
Subclock input pulse width low level	$t_{EXCLL}$	—	15.26	—	$\mu\text{s}$	Figure 26.7
Subclock input pulse width high level	$t_{EXCLH}$	—	15.26	—	$\mu\text{s}$	
Subclock input rising time	$t_{EXCLr}$	—	—	10	ns	
Subclock input falling time	$t_{EXCLf}$	—	—	10	ns	



**Figure 26.7 Subclock Input Timing**

## 26.6 Waveform Forming Circuit

To remove noise from the subclock input at the EXCL pin, the subclock is sampled by a divided  $\phi$  clock. The sampling frequency is set by the NESEL bit in LPWRCR.

The subclock is not sampled in subactive mode, subsleep mode, or watch mode.

## 26.7 Clock Select Circuit

The clock select circuit selects the system clock that is used in this LSI.

Either a clock generated by an oscillator to which the EXTAL and XTAL pins are input or a 24-MHz clock generated by multiplication in the PLL circuit is selected as a system clock when returning from high-speed mode, medium-speed mode, sleep mode, or software standby mode. A clock generated by an oscillator to which the EXTAL and XTAL pins are input is selected as a system clock when returning from the reset state or hardware standby mode.

A subclock input from the EXCL pin is selected as a system clock in subactive mode, subsleep mode, or watch mode. At this time, modules such as the CPU, TMR\_0, TMR\_1, WDT\_0, WDT\_1, ports, and interrupt controller and their functions operate depending on the  $\phi_{SUB}$ . The count clock and sampling clock for each timer are divided  $\phi_{SUB}$  clocks.

## 26.8 PLL Circuit

This LSI incorporates a PLL circuit which generates a 48-MHz clock ( $\phi_{48}$ ) or a 24-MHz clock ( $\phi_{24}$ ) obtained by dividing the 48-MHz clock by two as the USB operating clock. The clock source is the clock input from the USEXCL pin or the clock generated by an oscillator to which the EXTAL and XTAL pins are input. The PLL input clock must be 8, 12, 16, 20, or 24 MHz. The 48-MHz clock input to the USEXCL pin can be directly used as the USB operating clock instead

of the PLL circuit output clock. For details, see section 18.3.17, USB PLL Control Register (UPLLCR).

The 24-MHz clock generated by the PLL circuit can also be used as the system clock. For details, see section 27.1.3, System Control Register 2 (SYSCR2).

To activate the PLL circuit, first clear the SMSTPB1 bit in SUBMSTPBL to 0, then after clearing the USB module stop mode, make settings for UPLLCR. If the USB module is not used, after activating the PLL circuit, set the SMSTPB1 bit in SUBMSTPBL to 1 to make the USB module enter module stop mode.

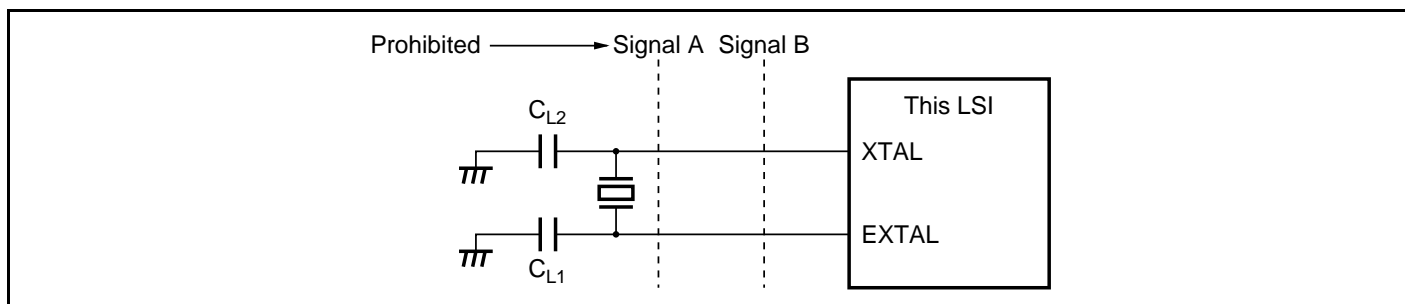
## 26.9 Usage Notes

### 26.9.1 Note on Resonator

Since all kinds of characteristics of the resonator are closely related to the board design by the user, use the example of resonator connection in this document for only reference; be sure to use an resonator that has been sufficiently evaluated by the user. Consult with the resonator manufacturer about the resonator circuit ratings which vary depending on the stray capacitances of the resonator and installation circuit. Make sure the voltage applied to the oscillation pins do not exceed the maximum rating.

### 26.9.2 Notes on Board Design

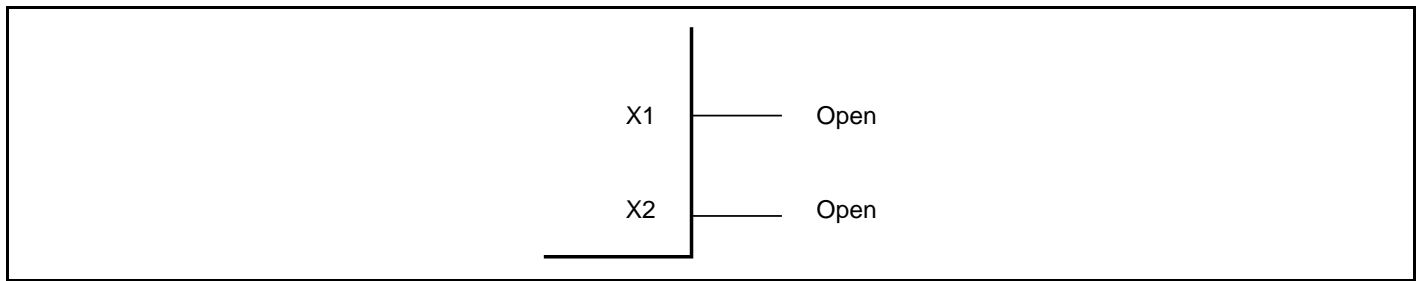
When using a crystal resonator, the crystal resonator and its load capacitors should be placed as close as possible to the EXTAL and XTAL pins. Other signal lines should be routed away from the oscillation circuit to prevent inductive interference with the correct oscillation as shown in figure 26.8.



**Figure 26.8 Note on Board Design of Oscillation Circuit Section**

### 26.9.3 Processing for X1 and X2 Pins

The X1 and X2 pins should be left open as shown in figure 26.9.



**Figure 26.9 Processing for X1 and X2 Pins**



## Section 27 Power-Down Modes

For operating modes after the reset state is cancelled, this LSI has not only the normal program execution state but also seven power-down modes in which power consumption is significantly reduced. In addition, there is also module stop mode in which reduced power consumption can be achieved by individually stopping on-chip peripheral modules.

- **Medium-speed mode**  
System clock frequency for the CPU operation can be selected as  $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , or  $\phi/32$ .
- **Subactive mode**  
The CPU operates based on the subclock and on-chip peripheral modules other than TMR\_0, TMR\_1, WDT\_0, and WDT\_1 stop operating.
- **Sleep mode**  
The CPU stops but on-chip peripheral modules continue operating.
- **Subsleep mode**  
The CPU and on-chip peripheral modules other than TMR\_0, TMR\_1, WDT\_0, and WDT\_1 stop operating.
- **Watch mode**  
The CPU and on-chip peripheral modules other than WDT\_1 stop operating.
- **Software standby mode**  
Clock oscillation stops, and the CPU and on-chip peripheral modules stop operating.
- **Hardware standby mode**  
Clock oscillation stops, and the CPU and on-chip peripheral modules enter reset state.
- **Module stop mode**  
Independently of above operating modes, on-chip peripheral modules that are not used can be stopped individually.

## 27.1 Register Descriptions

Power-down modes are controlled by the following registers. To access SBYCR, LPWRCR, SYSCR2, MSTPCRH, and MSTPCRL, the FLSHE bit in the serial timer control register (STCR) must be cleared to 0. For details on STCR, see section 3.2.3, Serial Timer Control Register (STCR).

- Standby control register (SBYCR)
- Low power control register (LPWRCR)
- System control register 2 (SYSCR2)
- Module stop control register H (MSTPCRH)
- Module stop control register L (MSTPCRL)
- Sub-chip module stop control register BH (SUBMSTPBH)
- Sub-chip module stop control register BL (SUBMSTPBL)

### 27.1.1 Standby Control Register (SBYCR)

SBYCR controls power-down modes.

Bit	Bit Name	Initial Value	R/W	Description
7	SSBY	0	R/W	<p>Software Standby</p> <p>Specifies the operating mode to be entered after executing the SLEEP instruction.</p> <p>When the SLEEP instruction is executed in high-speed mode or medium-speed mode:</p> <p>0: Shifts to sleep mode</p> <p>1: Shifts to software standby mode, subactive mode, or watch mode</p> <p>When the SLEEP instruction is executed in subactive mode:</p> <p>0: Shifts to subsleep mode</p> <p>1: Shifts to watch mode or high-speed mode</p> <p>Note that the SSBY bit is not changed even if a mode transition occurs by an interrupt.</p>



Bit	Bit Name	Initial Value	R/W	Description
6	STS2	0	R/W	Standby Timer Select 2 to 0
5	STS1	0	R/W	Select the wait time for clock stabilization from clock oscillation start when canceling software standby mode, watch mode, or subactive mode. Select a wait time of 8 ms (oscillation stabilization time) or more, depending on the operating frequency. Table 27.1 shows the relationship between the STS2 to STS0 values and wait time.  With an external clock, there are no specific wait requirements. Normally the minimum value is recommended.
4	STS0	0	R/W	
3	DTSPEED	0	R/W	<p>DTC/RFU Speed</p> <p>Specifies the operating clock for the bus masters (DTC and RFU) other than the CPU in medium-speed mode.</p> <p>0: All bus masters operate based on the medium-speed clock.</p> <p>1: The DTC/RFU operates based on the system clock.</p> <p>The operating clock is changed when a DTC/RFU transfer is requested even if the CPU operates based on the medium-speed clock. Note however that medium-speed mode for the RFU is not supported in this LSI. This bit must be set to 1 when the RFU is used in medium-speed mode. If the RFU is activated while this bit is cleared to 0, the program may go wild.</p>
2	SCK2	0	R/W	System Clock Select 2 to 0
1	SCK1	0	R/W	Select a clock for the bus master in high-speed mode or medium-speed mode.  When making a transition to subactive mode or watch mode, the SCK2 to SCK0 bits must be cleared to B'000.
0	SCK0	1	R/W	
				<p>000: High-speed mode</p> <p>001: Medium-speed clock: <math>\phi/2</math> (Initial value)</p> <p>010: Medium-speed clock: <math>\phi/4</math></p> <p>011: Medium-speed clock: <math>\phi/8</math></p> <p>100: Medium-speed clock: <math>\phi/16</math></p> <p>101: Medium-speed clock: <math>\phi/32</math></p> <p>11X: —</p>

Legend:

X: Don't care

**Table 27.1 Operating Frequency and Wait Time**

STS2	STS1	STS0	Wait Time	24M Hz	20 MHz	10 MHz	8 MHz	6 MHz	Unit
0	0	0	8192 states	0.3	0.4	0.8	1.0	1.3	ms
0	0	1	16384 states	0.7	0.8	1.6	2.0	2.7	
0	1	0	32768 states	1.3	2.0	3.3	4.1	5.5	
0	1	1	65536 states	2.7	4.1	6.6	8.2	10.9	
1	0	0	131072 states	5.5	8.2	13.1	16.4	21.8	
1	0	1	262144 states	10.9	16.4	26.2	32.8	43.6	
1	1	0	Reserved	—	—	—	—	—	—
1	1	1	16 states*	0.7	0.8	1.6	2.0	2.7	μs

Recommended specification

Note: \* This setting cannot be made in the flash-memory version of this LSI.

### 27.1.2 Low-Power Control Register (LPWRCR)

LPWRCR controls power-down modes.

Bit	Bit Name	Initial Value	R/W	Description
7	DTON	0	R/W	<p>Direct Transfer On Flag</p> <p>Specifies the operating mode to be entered after executing the SLEEP instruction.</p> <p>When the SLEEP instruction is executed in high-speed mode or medium-speed mode:</p> <p>0: Shifts to sleep mode, software standby mode, or watch mode</p> <p>1: Shifts directly to subactive mode, or shifts to sleep mode or software standby mode</p> <p>When the SLEEP instruction is executed in subactive mode:</p> <p>0: Shifts to subsleep mode or watch mode</p> <p>1: Shifts directly to high-speed mode, or shifts to subsleep mode</p>

Bit	Bit Name	Initial Value	R/W	Description
6	LSON	0	R/W	<p>Low-Speed On Flag</p> <p>Specifies the operating mode to be entered after executing the SLEEP instruction. This bit also controls whether to shift to high-speed mode or subactive mode when watch mode is cancelled.</p> <p>When the SLEEP instruction is executed in high-speed mode or medium-speed mode:</p> <p>0: Shifts to sleep mode, software standby mode, or watch mode</p> <p>1: Shifts to watch mode or subactive mode</p> <p>When the SLEEP instruction is executed in subactive mode:</p> <p>0: Shifts directly to watch mode or high-speed mode</p> <p>1: Shifts to subsleep mode or watch mode</p> <p>When watch mode is cancelled:</p> <p>0: Shifts to high-speed mode</p> <p>1: Shifts to subactive mode</p>
5	NESEL	0	R/W	<p>Noise Elimination Sampling Frequency Select</p> <p>Selects the frequency by which the subclock (<math>\phi</math>SUB) input from the EXCL pin is sampled using the clock (<math>\phi</math>) generated by the system clock pulse generator. Clear this bit to 0 in this LSI.</p> <p>0: Sampling using <math>\phi/32</math> clock</p> <p>1: Sampling using <math>\phi/4</math> clock</p>
4	EXCLE	0	R/W	<p>Subclock Input Enable</p> <p>Enables/disables subclock input from the EXCL pin.</p> <p>0: Disables subclock input from the EXCL pin</p> <p>1: Enables subclock input from the EXCL pin</p>
3 to 0	—	All 0	R/(W)	<p>Reserved</p> <p>The initial value should not be changed.</p>

### 27.1.3 System Control Register 2 (SYSCR2)

SYSCR2 controls switching of the system clock source. The system clock can be selected from the clock ( $\phi$ ) input from the EXTAL and XTAL pins, the subclock ( $\phi$ SUB) input from the EXCL pin, or the 24-MHz clock ( $\phi$ 24) generated by the PLL circuit. SYSCR2 selects between  $\phi$  and  $\phi$ 24.

Before using this function to switch the clock, the PLL circuit must be started up to provide a stable 24-MHz clock.

Bit	Bit Name	Initial Value	R/W	Description
7	KWUL1	0	R/W	For details on bits 7 to 5, see section 9.6.4, System Control Register 2 (SYSCR2).
6	KWUL0	0	R/W	
5	P6PUE	0	R/W	
4, 3	—	All 0	R/(W)	Reserved The initial value should not be changed.
2	CKCHGE	0	R/W	<p><b>Clock Change Enable</b></p> <p>Specifies the next operating mode and system clock source (<math>\phi</math> or <math>\phi_{24}</math>) when the SLEEP instruction is executed while the SSBY bit is set to 1 in high-speed mode or medium-speed mode. If the SLEEP instruction is executed while the SSBY bit is cleared to 0, the system clock source is not switched and operation shifts to sleep mode.</p> <p>0: Enters software standby mode or watch mode, and switches to the system clock source specified by the PLCKS bit.</p> <p>1: Directly switches to the system clock source specified by the PLCKS bit.</p>
1	—	0	R/(W)	Reserved The initial value should not be changed.
0	PLCKS	0	R/W	<p><b>PLL Clock Select</b></p> <p>Specifies <math>\phi</math> or <math>\phi_{24}</math> as the system clock source in high-speed mode or medium-speed mode. If the LSON bit in LPWCR and this bit are both set to 1 simultaneously, the subclock selection by the LSON bit has higher priority than clock selection by this bit.</p> <p>0: Specifies <math>\phi</math> as the system clock source.</p> <p>Executing the SLEEP instruction while PLCKS = 0 and SSBY = 1 can switch the clock source to <math>\phi</math>.</p> <p>Executing the SLEEP instruction while LSON = 1 and SSBY = 1 can switch the clock source to 32-kHz <math>\phi_{SUB}</math>.</p> <p>1: Specifies <math>\phi_{24}</math> as the system clock source.</p> <p>Executing the SLEEP instruction while PLCKS = 1 and SSBY = 1 can switch the clock source to <math>\phi_{24}</math>.</p>

### 27.1.4 Module Stop Control Registers H and L (MSTPCRH, MSTPCRL) Sub-Chip Module Stop Control Registers BH and BL (SUBMSTPBH, SUBMSTPBL)

MSTPCR and SUBMSTPB specify on-chip peripheral modules to shift to module stop mode in module units. Each module can enter module stop mode by setting the corresponding bit to 1.

- MSTPCRH

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	MSTP15	0*	R/W	
6	MSTP14	0	R/W	Data transfer controller (DTC)
5	MSTP13	1	R/W	16-bit free-running timer (FRT)
4	MSTP12	1	R/W	8-bit timers (TMR_0, TMR_1)
3	MSTP11	1	R/W	8-bit PWM timer (PWM), 14-bit PWM timer (PWMX)
2	MSTP10	1	R/W	D/A converter
1	MSTP9	1	R/W	A/D converter
0	MSTP8	1	R/W	8-bit timers (TMR_X, TMR_Y), timer connection

Note: \* Do not set this bit to 1.

- MSTPCRL

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	MSTP7	1	R/W	Serial communication interface 0 (SCI_0)
6	MSTP6	1	R/W	Serial communication interface 1 (SCI_1)
5	MSTP5	1	R/W	Serial communication interface 2 (SCI_2)
4	MSTP4	1	R/W	I <sup>2</sup> C bus interface channel 0 (IIC_0)
3	MSTP3	1	R/W	I <sup>2</sup> C bus interface channel 1 (IIC_1)
2	MSTP2	1*	R/W	
1	MSTP1	1	R/W	CRC operator
0	MSTP0	1*	R/W	

Note: \* Do not clear this bit to 0.

- SUBMSTPBH

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	SMSTPB15	1*	R/W	
6	SMSTPB14	1*	R/W	
5	SMSTPB13	1*	R/W	
4	SMSTPB12	1*	R/W	
3	SMSTPB11	1*	R/W	
2	SMSTPB10	1*	R/W	
1	SMSTPB9	1	R/W	Encryption operation circuit (GF)
0	SMSTPB8	1	R/W	Encryption operation circuit (DES)

Note: \* Do not clear this bit to 0.

- SUBMSTPBL

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	SMSTPB7	1*	R/W	
6	SMSTPB6	1	R/W	Multimedia card interface (MCIF)
5	SMSTPB5	1*	R/W	
4	SMSTPB4	1*	R/W	
3	SMSTPB3	1*	R/W	
2	SMSTPB2	1*	R/W	
1	SMSTPB1	1	R/W	Universal serial bus interface (USB)
0	SMSTPB0	1*	R/W	

Note: \* Do not clear this bit to 0.

## 27.2 Mode Transitions and LSI States

Figure 27.1 shows the enabled mode transition diagram. The mode transition from program execution state to program halt state is performed by the SLEEP instruction. The mode transition from program halt state to program execution state is performed by an interrupt. The  $\overline{STBY}$  input causes a mode transition from any state to hardware standby mode. The  $\overline{RES}$  input causes a mode transition from a state other than hardware standby mode to the reset state. Table 27.2 shows the LSI internal states in each operating mode.

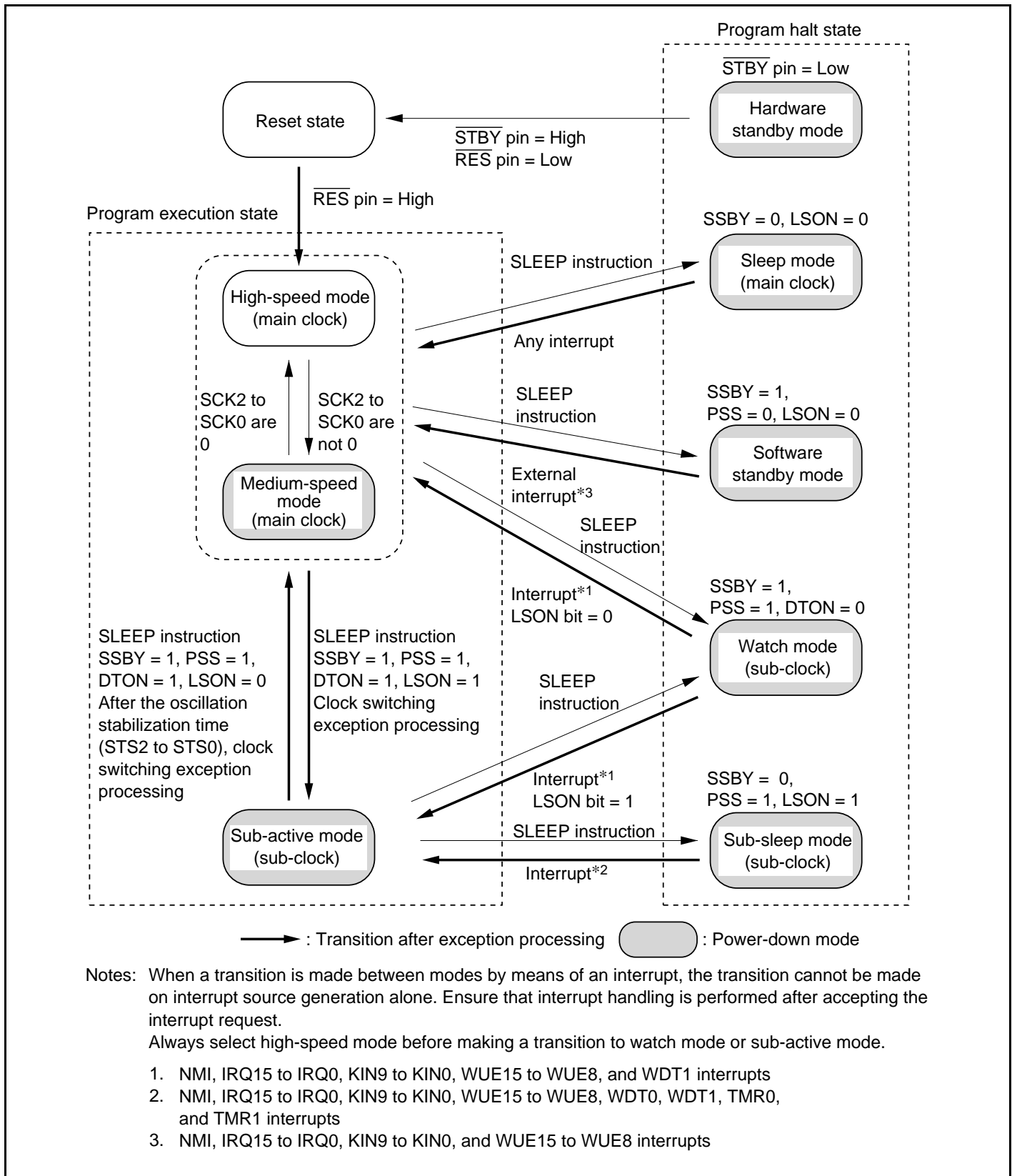


Figure 27.1 Mode Transition Diagram

**Table 27.2 LSI Internal States in Each Operating Mode**

Function		High-Speed	Medium-Speed	Sleep	Module Stop	Watch	Sub-Active	Sub-Sleep	Software Standby	Hardware Standby	
System clock pulse generator		Functioning	Functioning	Functioning	Functioning	Halted	Halted	Halted	Halted	Halted	
Subclock pulse generator		Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Halted	Halted	
CPU	Instruction execution	Functioning	Functioning in medium-speed mode	Halted	Functioning	Halted	Subclock operation	Halted	Halted	Halted	
	Registers			Retained		Retained		Retained			Undefined
External interrupts	NMI	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Halted	
	IRQ15 to IRQ0										
	KIN9 to KIN0										
	WUE15 to WUE8										
Peripheral modules	RFU	Functioning	Functioning	Functioning	Functioning	Functioning	Halted (retained)	Halted (retained)	Halted (retained)	Halted (reset)	
	DTC		Functioning in medium-speed mode/Functioning								Functioning/Halted (retained)
	WDT_1	Functioning	Functioning	Functioning	Functioning	Subclock operation	Subclock operation	Subclock operation	Halted (retained)	Halted (reset)	
	WDT_0										Halted (retained)
	TMR_0, TMR_1										Functioning/Halted (retained)
	FRT	Functioning	Functioning	Functioning	Functioning	Functioning/Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	
	TMR_X, TMR_Y										
	Timer connection										
	IIC_0										
	IIC_1										
	CRC										
	USB										
	DES, GF										
	MCIF	Functioning	Functioning	Functioning	Functioning/Halted (reset)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (reset)	
	SCI_0										
	SCI_1										
	SCI_2										
	PWM										
	PWMX										
	D/A converter										
A/D converter											
RAM	Functioning (DTC)	Functioning	Functioning	Functioning	Retained	Functioning	Retained	Retained	Retained		
I/O										Functioning	Functioning

Note: "Halted (retained)" means that internal register values are retained. The internal state is "operation suspended."

"Halted (reset)" means that internal register values and internal states are initialized.

In module stop mode, only modules for which a stop setting has been made are halted (reset or retained).



## 27.3 Medium-Speed Mode

The CPU makes a transition to medium-speed mode as soon as the current bus cycle ends according to the setting of the SCK2 to SCK0 bits in SBYCR. In medium-speed mode, the CPU operates on the operating clock ( $\phi/2$ ,  $\phi/4$ ,  $\phi/8$ ,  $\phi/16$ , or  $\phi/32$ ) specified by the SCK2 to SCK0 bits. The bus masters other than the CPU (DTC or RFU) also operate in medium-speed mode when the DTSPEED bit in SBYCR is cleared to 0. Note however that the DTSPEED bit must be set to 1 when the RFU is used in medium-speed mode. On-chip peripheral modules other than the bus masters always operate on the system clock ( $\phi$ ).

When the DTSPEED bit in SBYCR or the EXCKS bit in BCSR2 is set to 1, the  $\phi$  clock can be used as the DTC/RFU operating clock or external extended area bus cycle clock.

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if  $\phi/4$  is selected as the operating clock, on-chip memory is accessed in 4 states, and internal I/O registers in 8 states.

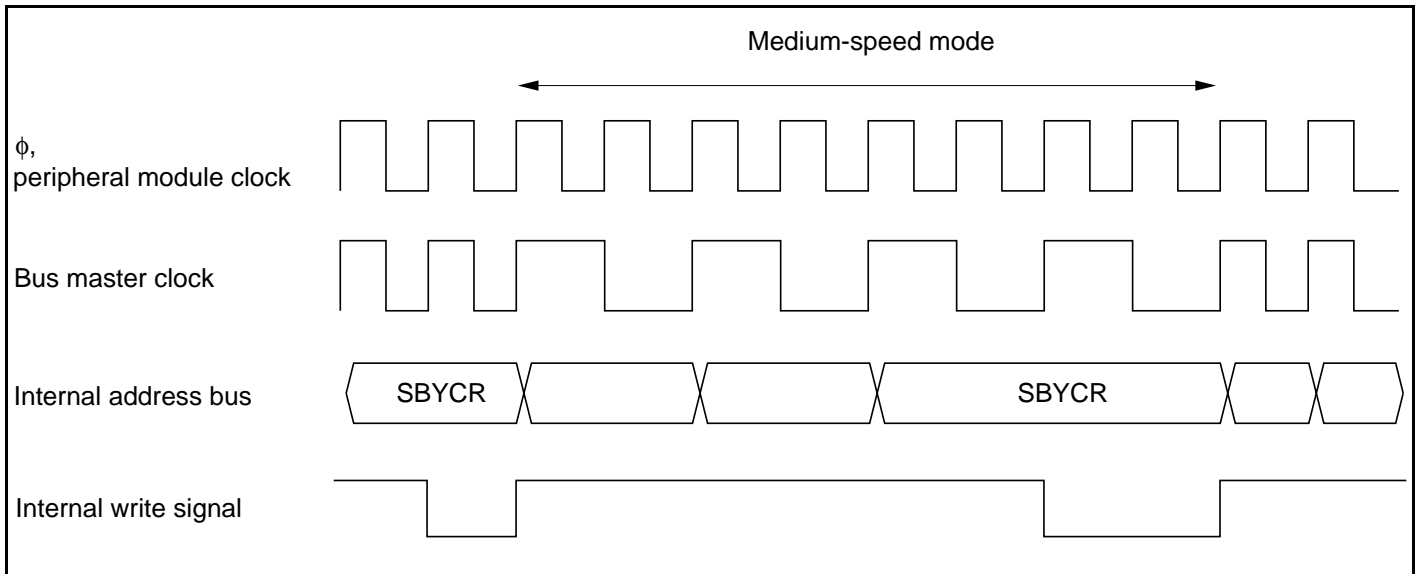
By clearing all of bits SCK2 to SCK0 to 0, a transition is made to high-speed mode at the end of the current bus cycle.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, and the LSON bit in LPWRCR is cleared to 0, a transition is made to sleep mode. When sleep mode is cleared by an interrupt, medium-speed mode is restored. When the SLEEP instruction is executed with the SSBY bit set to 1, the LSON bit cleared to 0, and the PSS bit in TCSR (WDT\_1) cleared to 0, operation shifts to software standby mode. When software standby mode is cleared by an external interrupt, medium-speed mode is restored.

When the  $\overline{\text{RES}}$  pin is set low and medium-speed mode is cancelled, operation shifts to the reset state. The same applies in the case of a reset caused by overflow of the watchdog timer.

When the  $\overline{\text{STBY}}$  pin is driven low, a transition is made to hardware standby mode.

Figure 27.2 shows an example of medium-speed mode timing.



**Figure 27.2 Medium-Speed Mode Timing**

## 27.4 Sleep Mode

The CPU makes a transition to sleep mode if the SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0 and the LSON bit in LPWRCR is cleared to 0. In sleep mode, CPU operation stops but the peripheral modules do not stop. The contents of the CPU's internal registers are retained.

Sleep mode is exited by any interrupt, the  $\overline{\text{RES}}$  pin, or the  $\overline{\text{STBY}}$  pin.

When an interrupt occurs, sleep mode is exited and interrupt exception handling starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.

Setting the  $\overline{\text{RES}}$  pin level low cancels sleep mode and selects the reset state. After the oscillation stabilization time has passed, driving the  $\overline{\text{RES}}$  pin high causes the CPU to start reset exception handling.

When the  $\overline{\text{STBY}}$  pin level is driven low, a transition is made to hardware standby mode.

## 27.5 Software Standby Mode

The CPU makes a transition to software standby mode when the SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1, the LSON bit in LPWRCR is cleared to 0, and the PSS bit in TCSR (WDT\_1) is cleared to 0.

In software standby mode, the CPU, on-chip peripheral modules, and clock pulse generator all stop. However, the contents of the CPU's internal registers, on-chip RAM data, I/O ports, and the

states of on-chip peripheral modules other than the SCI, PWM, and PWMX, are retained as long as the prescribed voltage is supplied.

Software standby mode is cleared by an external interrupt (NMI, IRQ15 to IRQ0, KIN9 to KIN0, or WUE15 to WUE8), the  $\overline{\text{RES}}$  pin input, or  $\overline{\text{STBY}}$  pin input.

When an external interrupt request signal is input, system clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SBYCR, software standby mode is cleared, and interrupt exception handling is started. When exiting software standby mode with an IRQ15 to IRQ0 interrupt, set the corresponding enable bit to 1. When exiting software standby mode with a KIN9 to KIN0 or WUE15 to WUE8 interrupt, enable the input. In these cases, ensure that no interrupt with a higher priority than interrupts IRQ15 to IRQ0 is generated. In the case of an IRQ15 to IRQ0 interrupt, software standby mode is not exited if the corresponding enable bit is cleared to 0 or if the interrupt has been masked by the CPU. In the case of a KIN9 to KIN0 or WUE15 to WUE8 interrupt, software standby mode is not exited if input is disabled or if the interrupt has been masked by the CPU.

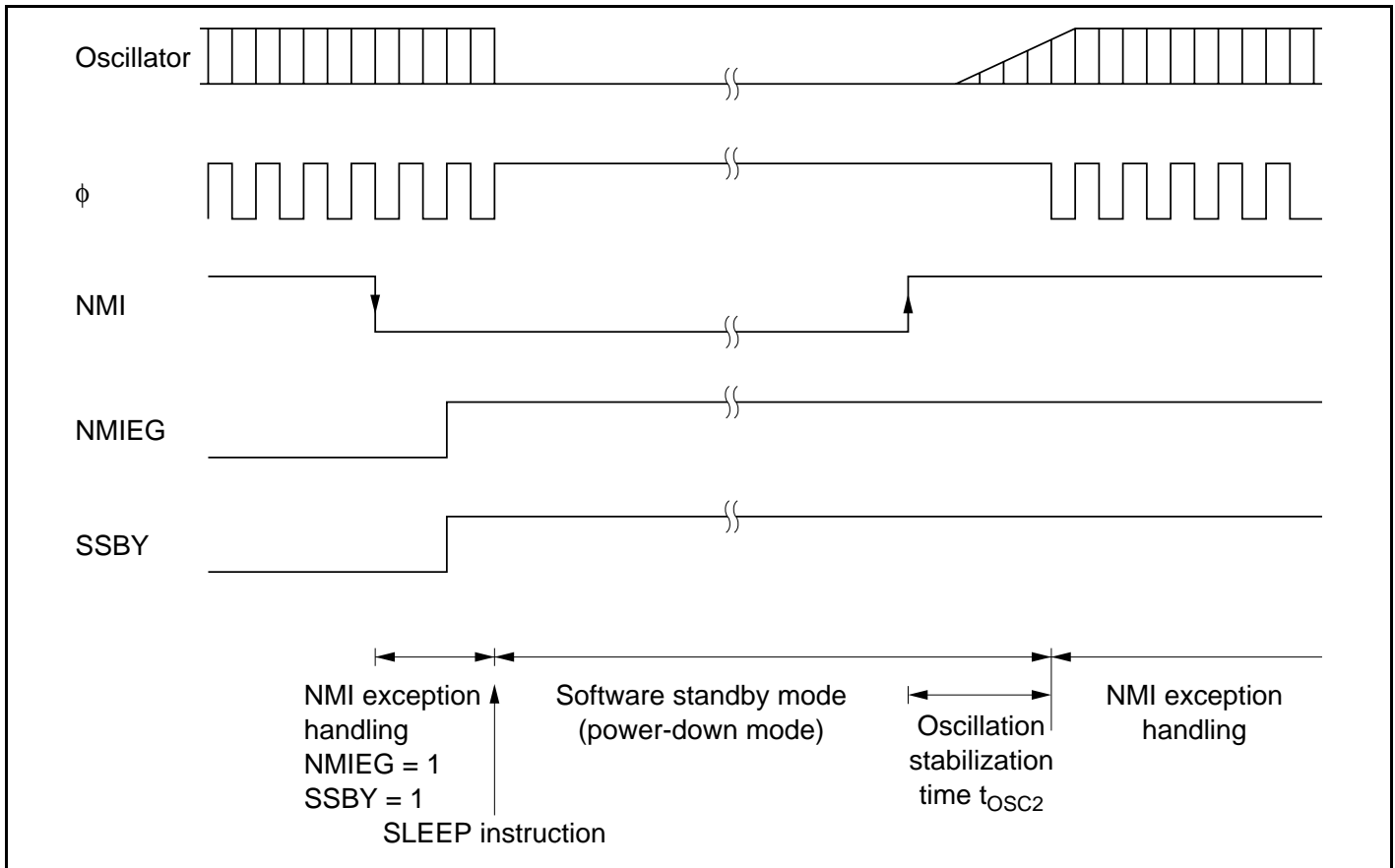
When the  $\overline{\text{RES}}$  pin is driven low, system clock oscillation is started. At the same time as system clock oscillation starts, the system clock is supplied to the entire LSI. Note that the  $\overline{\text{RES}}$  pin must be held low until clock oscillation stabilizes. When the  $\overline{\text{RES}}$  pin goes high after clock oscillation stabilizes, the CPU begins reset exception handling.

When the  $\overline{\text{STBY}}$  pin is driven low, software standby mode is cancelled and a transition is made to hardware standby mode.

Figure 27.3 shows an example in which a transition is made to software standby mode at the falling edge of the NMI pin, and software standby mode is cleared at the rising edge of the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in SYSCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge of the NMI pin.



**Figure 27.3 Software Standby Mode Application Example**

## 27.6 Hardware Standby Mode

The CPU makes a transition to hardware standby mode from any mode when the  $\overline{STBY}$  pin is driven low.

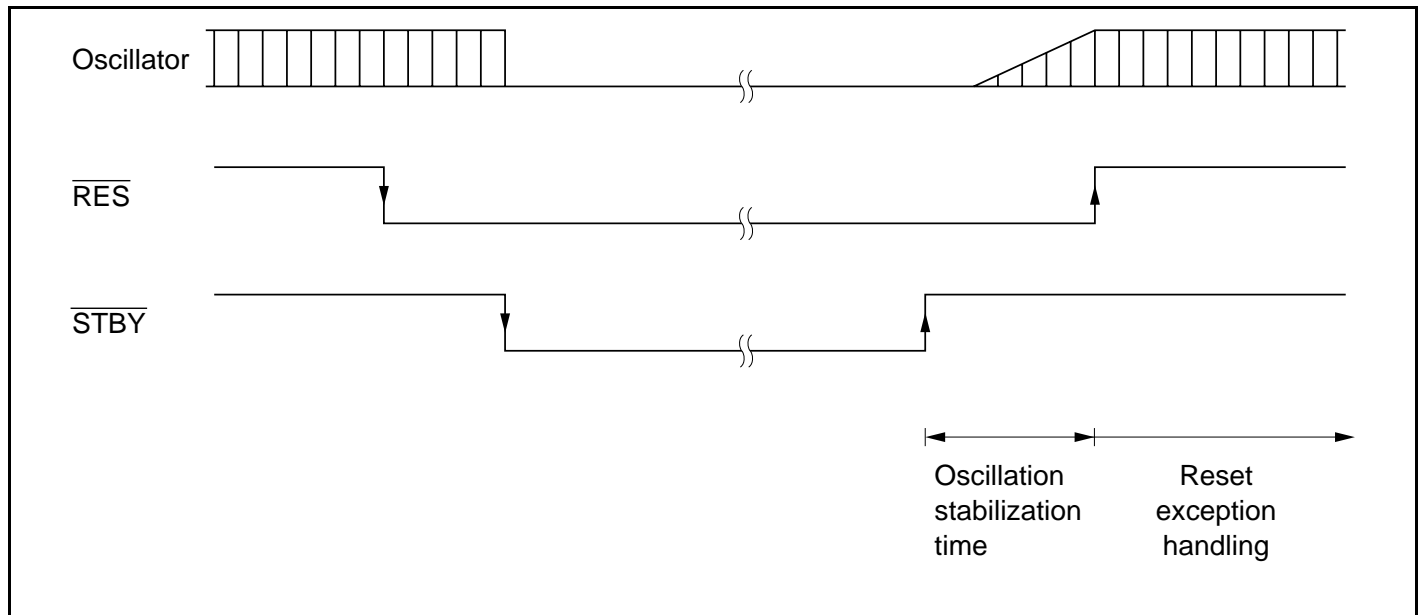
In hardware standby mode, all functions enter the reset state. As long as the prescribed voltage is supplied, on-chip RAM data is retained. The I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the  $\overline{STBY}$  pin low. Do not change the state of the mode pins ( $\overline{MD2}$ , MD1, and MD0) while this LSI is in hardware standby mode.

Hardware standby mode is cleared by the  $\overline{STBY}$  pin input or the  $\overline{RES}$  pin input.

When the  $\overline{STBY}$  pin is driven high while the  $\overline{RES}$  pin is low, clock oscillation is started. Ensure that the  $\overline{RES}$  pin is held low until system clock oscillation stabilizes. When the  $\overline{RES}$  pin is subsequently driven high after the clock oscillation stabilization time has passed, reset exception handling starts.

Figure 27.4 shows an example of hardware standby mode timing.



**Figure 27.4 Hardware Standby Mode Timing**

## 27.7 Watch Mode

The CPU makes a transition to watch mode when the SLEEP instruction is executed in high-speed mode or subactive mode with the SSBY bit in SBYCR set to 1, the DTON bit in LPWRCR cleared to 0, and the PSS bit in TCSR (WDT\_1) set to 1.

In watch mode, the CPU is stopped and peripheral modules other than WDT\_1 are also stopped. The contents of the CPU's internal registers, several on-chip peripheral module registers, and on-chip RAM data are retained and the I/O ports retain their values before transition as long as the prescribed voltage is supplied.

Watch mode is exited by an interrupt (WOVI1, NMI, IRQ15 to IRQ0, KIN9 to KIN0, or WUE15 to WUE8),  $\overline{\text{RES}}$  pin input, or  $\overline{\text{STBY}}$  pin input.

When an interrupt occurs, watch mode is exited and a transition is made to high-speed mode or medium-speed mode when the LSON bit in LPWRCR cleared to 0 or to subactive mode when the LSON bit is set to 1. When a transition is made to high-speed mode, a stable clock is supplied to the entire LSI and interrupt exception handling starts after the time set in the STS2 to STS0 bits in SBYCR has elapsed.

In the case of an IRQ15 to IRQ0 interrupt, watch mode is not exited if the corresponding enable bit has been cleared to 0 or the interrupt is masked by the CPU. In the case of a KIN9 to KIN0 or WUE15 to WUE8 interrupt, watch mode is not exited if input is disabled or the interrupt is

masked by the CPU. In the case of an interrupt from the on-chip peripheral modules, watch mode is not exited if the interrupt enable register has been set to disable the reception of that interrupt or the interrupt is masked by the CPU.

When the  $\overline{\text{RES}}$  pin is driven low, system clock oscillation starts. Simultaneously with the start of system clock oscillation, the system clock is supplied to the entire LSI. Note that the  $\overline{\text{RES}}$  pin must be held low until clock oscillation is stabilized. If the  $\overline{\text{RES}}$  pin is driven high after the clock oscillation stabilization time has passed, the CPU begins reset exception handling.

If the  $\overline{\text{STBY}}$  pin is driven low, the LSI enters hardware standby mode.

## 27.8 Subsleep Mode

The CPU makes a transition to subsleep mode when the SLEEP instruction is executed in subactive mode with the SSBY bit in SBYCR cleared to 0, the LSON bit in LPWRCR set to 1, and the PSS bit in TCSR (WDT\_1) set to 1.

In subsleep mode, the CPU is stopped. Peripheral modules other than TMR\_0, TMR\_1, WDT\_0, and WDT\_1 are also stopped. The contents of the CPU's internal registers, several on-chip peripheral module registers, and on-chip RAM data are retained and the I/O ports retain their values before transition as long as the prescribed voltage is supplied.

Subsleep mode is exited by an interrupt (interrupts by on-chip peripheral modules, NMI, IRQ15 to IRQ0, KIN9 to KIN0, or WUE15 to WUE8), the  $\overline{\text{RES}}$  pin input, or the  $\overline{\text{STBY}}$  pin input.

When an interrupt occurs, subsleep mode is exited and interrupt exception handling starts.

In the case of an IRQ15 to IRQ0 interrupt, subsleep mode is not exited if the corresponding enable bit has been cleared to 0 or the interrupt is masked by the CPU. In the case of a KIN9 to KIN0 or WUE15 to WUE8 interrupt, subsleep mode is not exited if input is disabled or the interrupt is masked by the CPU. In the case of an interrupt from the on-chip peripheral modules, subsleep mode is not exited if the interrupt enable register has been set to disable the reception of that interrupt or the interrupt is masked by the CPU.

When the  $\overline{\text{RES}}$  pin is driven low, system clock oscillation starts. Simultaneously with the start of system clock oscillation, the system clock is supplied to the entire LSI. Note that the  $\overline{\text{RES}}$  pin must be held low until clock oscillation is stabilized. If the  $\overline{\text{RES}}$  pin is driven high after the clock oscillation stabilization time has passed, the CPU begins reset exception handling.

If the  $\overline{\text{STBY}}$  pin is driven low, the LSI enters hardware standby mode.

## 27.9 Subactive Mode

The CPU makes a transition to subactive mode when the SLEEP instruction is executed in high-speed mode with the SSBY bit in SBYCR set to 1, the DTON bit and LSON bit in LPWRCR set to 1, and the PSS bit in TCSR (WDT\_1) set to 1. When an interrupt occurs in watch mode, and if the LSON bit in LPWRCR is 1, a direct transition is made to subactive mode. Similarly, if an interrupt occurs in subsleep mode, a transition is made to subactive mode.

In subactive mode, the CPU operates at a low speed based on the subclock and sequentially executes programs. Peripheral modules other than TMR\_0, TMR\_1, WDT\_0, and WDT\_1 are also stopped.

When operating the CPU in subactive mode, the SCK2 to SCK0 bits in SBYCR must be cleared to 0.

Subactive mode is exited by the SLEEP instruction,  $\overline{\text{RES}}$  pin input, or  $\overline{\text{STBY}}$  pin input.

When the SLEEP instruction is executed with the SSBY bit in SBYCR set to 1, the DTON bit in LPWRCR cleared to 0, and the PSS bit in TCSR (WDT\_1) set to 1, the CPU exits subactive mode and a transition is made to watch mode. When the SLEEP instruction is executed with the SSBY bit in SBYCR cleared to 0, the LSON bit in LPWRCR set to 1, and the PSS bit in TCSR (WDT\_1) set to 1, a transition is made to subsleep mode. When the SLEEP instruction is executed with the SSBY bit in SBYCR set to 1, the DTON bit and LSON bit in LPWRCR set to 10, and the PSS bit in TCSR (WDT\_1) set to 1, a direct transition is made to high-speed mode.

For details of direct transitions, see section 27.11, Direct Transitions.

When the  $\overline{\text{RES}}$  pin is driven low, system clock oscillation starts. Simultaneously with the start of system clock oscillation, the system clock is supplied to the entire LSI. Note that the  $\overline{\text{RES}}$  pin must be held low until the clock oscillation is stabilized. If the  $\overline{\text{RES}}$  pin is driven high after the clock oscillation stabilization time has passed, the CPU begins reset exception handling.

If the  $\overline{\text{STBY}}$  pin is driven low, the LSI enters hardware standby mode.

## 27.10 Module Stop Mode

Module stop mode can be individually set for each on-chip peripheral module.

When the corresponding MSTP bit in MSTPCR and SUBMSTPB is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. In turn, when the corresponding MSTP bit is cleared to 0, module stop mode is cancelled and the module operation

resumes at the end of the bus cycle. In module stop mode, the internal states of modules other than the MCIF, SCI, D/A converter, A/D converter, PWM, and PWMX are retained.

After the reset state is cancelled, all modules other than DTC are in module stop mode.

While an on-chip peripheral module is in module stop mode, read/write access to its registers is disabled.

## 27.11 Direct Transitions

The CPU executes programs in three modes: high-speed, medium-speed, and subactive. When a direct transition is made from high-speed mode to subactive mode, there is no interruption of program execution. A direct transition is enabled by setting the DTON bit in LPWRCCR to 1 and then executing the SLEEP instruction. After a transition, direct transition exception handling starts.

The CPU makes a transition to subactive mode when the SLEEP instruction is executed in high-speed mode with the SSBY bit in SBYCR set to 1, the LSON bit and DTON bit in LPWRCCR set to 11, and the PSS bit in TSCR (WDT\_1) set to 1.

To make a direct transition to high-speed mode after the time set in the STS2 to STS0 bits in SBYCR has elapsed, execute the SLEEP instruction in subactive mode with the SSBY bit in SBYCR set to 1, the LSON bit and DTON bit in LPWRCCR set to 01, and the PSS bit in TSCR (WDT\_1) set to 1.

In high-speed mode or medium-speed mode, the system clock source ( $\phi$  or  $\phi_{24}$ ) can be switched by using one of the following two methods according to the CKCHGE bit in SYSCR2. When the CKCHGE bit is cleared to 0, after a transition to software standby mode or watch mode is made, the system clock source is switched by a wakeup via an interrupt. When the CKCHGE bit is set to 1, a transition similar to active-subactive direct transition is made, and direct transition exception handling is executed after a direct transition.

In high-speed mode or medium-speed mode, do not execute a SLEEP instruction when a setting for making a direct transition to subactive mode and a setting for switching the system clock source at a direct transition are made. When the system clock source is to be switched at a direct transition, make sure the SLEEP instruction does not conflict with other interrupt sources.



## **27.12 Usage Notes**

### **27.12.1 I/O Port Status**

The status of the I/O ports is retained in software standby mode. Therefore, when a high level is output, the current consumption is not reduced by the amount of current to support the high level output.

### **27.12.2 Current Consumption when Waiting for Oscillation Stabilization**

The current consumption increases during oscillation stabilization.

### **27.12.3 DTC Module Stop Mode**

If the DTC module stop mode specification and DTC bus request occur simultaneously, the bus is released to the DTC and the MSTP bit cannot be set to 1. After completing the DTC bus cycle, set the MSTP bit to 1 again.



## Section 28 List of Registers

The register list gives information on the on-chip register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

### 1. Register Addresses (address order)

- Registers are listed from the lower allocation addresses.
- The MSB-side address is indicated for 16-bit addresses.
- Registers are classified by functional modules.
- The access size is indicated.

### 2. Register Bits

- Bit configurations of the registers are described in the same order as the Register Addresses (address order) above.
- Reserved bits are indicated by — in the bit name column.
- The bit number in the bit-name column indicates that the whole register is allocated as a counter or for holding data.
- 16-bit registers are indicated from the bit on the MSB side.

### 3. Register States in Each Operating Mode

- Register states are described in the same order as the Register Addresses (address order) above.
- The register states described here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

## 28.1 Register Addresses (Address Order)

The access size indicates the numbers of bits.

The number of access states indicates the number of states based on the specified reference clock.

Note: Access to undefined or reserved addresses is prohibited. Since operation or continued operation is not guaranteed when these registers are accessed, do not attempt such access.

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Command register 0	CMDR0	8	H'FBC0	MCIF	8	3
Command register 1	CMDR1	8	H'FBC1	MCIF	8	3
Command register 2	CMDR2	8	H'FBC2	MCIF	8	3
Command register 3	CMDR3	8	H'FBC3	MCIF	8	3
Command register 4	CMDR4	8	H'FBC4	MCIF	8	3
Command register 5	CMDR5	8	H'FBC5	MCIF	8	3
Command start register	CMDSTRT	8	H'FBC6	MCIF	8	3
Operation control register	OPCR	8	H'FBCA	MCIF	8	3
Card status register	CSTR	8	H'FBCB	MCIF	8	3
Interrupt control register 0	INTCR0	8	H'FBCC	MCIF	8	3
Interrupt control register 1	INTCR1	8	H'FBCE	MCIF	8	3
Interrupt status register 0	INTSTR0	8	H'FBCE	MCIF	8	3
Interrupt status register 1	INTSTR1	8	H'FBCE	MCIF	8	3
Transfer clock control register	CLKON	8	H'FBD0	MCIF	8	3
Command timeout control register	CTOCR	8	H'FBD1	MCIF	8	3
Pin mode control register	IOMCR	8	H'FBD3	MCIF	8	3
Transfer byte number count register	TBCR	8	H'FBD4	MCIF	8	3
Mode register	MODER	8	H'FBD6	MCIF	8	3
Command type register	CMDTYR	8	H'FBD8	MCIF	8	3
Response type register	RSPTYR	8	H'FBD9	MCIF	8	3
Transfer block number counter H	TBNCRH	8	H'FBDA	MCIF	8	3
Transfer block number counter L	TBNCRL	8	H'FBDB	MCIF	8	3
Response register 0	RSPR0	8	H'FBE0	MCIF	8	3
Response register 1	RSPR1	8	H'FBE1	MCIF	8	3
Response register 2	RSPR2	8	H'FBE2	MCIF	8	3
Response register 3	RSPR3	8	H'FBE3	MCIF	8	3
Response register 4	RSPR4	8	H'FBE4	MCIF	8	3
Response register 5	RSPR5	8	H'FBE5	MCIF	8	3
Response register 6	RSPR6	8	H'FBE6	MCIF	8	3
Response register 7	RSPR7	8	H'FBE7	MCIF	8	3
Response register 8	RSPR8	8	H'FBE8	MCIF	8	3

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Response register 9	RSPR9	8	H'FBE9	MCIF	8	3
Response register 10	RSPR10	8	H'FBEA	MCIF	8	3
Response register 11	RSPR11	8	H'FBEB	MCIF	8	3
Response register 12	RSPR12	8	H'FBEC	MCIF	8	3
Response register 13	RSPR13	8	H'FBED	MCIF	8	3
Response register 14	RSPR14	8	H'FBEE	MCIF	8	3
Response register 15	RSPR15	8	H'FBEF	MCIF	8	3
Response register 16	RSPR16	8	H'FBF0	MCIF	8	3
Response register D	RSPRD	8	H'FBF1	MCIF	8	3
Data timeout register H	DTOUTRH	8	H'FBF2	MCIF	8	3
Data timeout register L	DTOUTRL	8	H'FBF3	MCIF	8	3
Endpoint 4 packet size register	EP4PKTSZR	8	H'FD81	USB	8	3
Endpoint data register 0S	EPDR0S	8	H'FDAD	USB	8	3
FIFO valid size register 0SH	FVSR0SH	8	H'FDAE	USB	8	3
FIFO valid size register 0SL	FVSR0SL	8	H'FDAF	USB	8	3
USB interrupt enable register 1	USBIER1	8	H'FDB1	USB	8	3
USB interrupt flag register 1	USBIFR1	8	H'FDB2	USB	8	3
USB mode control register	USBMDCR	8	H'FDBC	USB	8	3
USB control register 1	USBCR1	8	H'FDBD	USB	8	3
Configuration value register	CONFV	8	H'FDBE	USB	8	3
RFU/FIFO read request flag register	UDTRFR	8	H'FDBF	USB	8	3
USB port control register	UPRTCR	8	H'FDC0	USB	8	3
USB test register 0	UTESTR0	8	H'FDC1	USB	8	3
USB test register 1	UTESTR1	8	H'FDC2	USB	8	3
Endpoint data register 3	EPDR3	8	H'FDDD	USB	8	3
FIFO valid size register 3H	FVSR3H	8	H'FDDE	USB	8	3
FIFO valid size register 3L	FVSR3L	8	H'FDDE	USB	8	3
Endpoint data register 2	EPDR2	8	H'FDE1	USB	8	3
FIFO valid size register 2H	FVSR2H	8	H'FDE2	USB	8	3
FIFO valid size register 2L	FVSR2L	8	H'FDE3	USB	8	3

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Endpoint size register 1	EPSZR1	8	H'FDE4	USB	8	3
Endpoint data register 1	EPDR1	8	H'FDE5	USB	8	3
FIFO valid size register 1H	FVSR1H	8	H'FDE6	USB	8	3
FIFO valid size register 1L	FVSR1L	8	H'FDE7	USB	8	3
Endpoint data register 00	EPDR00	8	H'FDE9	USB	8	3
FIFO valid size register 00H	FVSR00H	8	H'FDEA	USB	8	3
FIFO valid size register 00L	FVSR00L	8	H'FDEB	USB	8	3
Endpoint data register 01	EPDR01	8	H'FDED	USB	8	3
FIFO valid size register 01H	FVSR01H	8	H'FDEE	USB	8	3
FIFO valid size register 01L	FVSR01L	8	H'FDEF	USB	8	3
Packet transmission enable register 0	PTTER0	8	H'FDF0	USB	8	3
USB interrupt enable register 0	USBIER0	8	H'FDF1	USB	8	3
USB interrupt flag register 0	USBIFR0	8	H'FDF2	USB	8	3
Transfer normal completion interrupt flag register 0	TSFR0	8	H'FDF3	USB	8	3
Transfer abnormal completion interrupt flag register 0	TFFR0	8	H'FDF4	USB	8	3
USB control/status register 0	USBCSR0	8	H'FDF5	USB	8	3
Endpoint stall register 0	EPSTLR0	8	H'FDF6	USB	8	3
Endpoint direction register 0	EPDIR0	8	H'FDF7	USB	8	3
Endpoint reset register 0	EPRSTR0	8	H'FDF8	USB	8	3
Device resume register	DEVRSMR	8	H'FDF9	USB	8	3
Interrupt source select register 0	INTSELR0	8	H'FDFA	USB	8	3
USB control register 0	USBCR0	8	H'FDFD	USB	8	3
USB PLL control register 0	UPLLCR	8	H'FD FE	USB	8	3
Sub-chip module stop control register BH	SUBMSTPBH	8	H'FE3E	SYSTEM	8	3
Sub-chip module stop control register BL	SUBMSTPBL	8	H'FE3F	SYSTEM	8	3
FIFO status/register/pointer 0	FSTR0	8	H'FEA0	RFU	8	3
FIFO status/register/pointer 1	FSTR1	8	H'FEA1	RFU	8	3

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
FIFO status/register/pointer 2	FSTR2	8	H'FEA2	RFU	8	3
FIFO status/register/pointer 3	FSTR3	8	H'FEA3	RFU	8	3
Data transfer control register A	DTCRA	8	H'FEA4	RFU	8	3
Data transfer control register B	DTCRB	8	H'FEA5	RFU	8	3
Data transfer ID register	DTIDR	8	H'FEA6	RFU	8	3
Data transfer status register C	DTSTRC	8	H'FEA7	RFU	8	3
Data transfer ID read/write select register A	DTIDSRA	8	H'FEA8	RFU	8	3
Data transfer ID read/write select register B	DTIDSRB	8	H'FEA9	RFU	8	3
Data transfer status register A	DTSTRA	8	H'FEAA	RFU	8	3
Data transfer status register B	DTSTRB	8	H'FEAB	RFU	8	3
Data transfer control register D	DTCRD	8	H'FEAC	RFU	8	3
Data transfer interrupt enable register	DTIER	8	H'FEAD	RFU	8	3
Data transfer register select register	DTRSR	8	H'FEAE	RFU	8	3
IIC operation reservation adapter control register_0	ICCRX_0	8	H'FEB0	IIC_0	16	2
IIC operation reservation adapter status register A_0	ICSRA_0	8	H'FEB1	IIC_0	16	2
IIC operation reservation adapter status register B_0	ICSRB_0	8	H'FEB2	IIC_0	16	2
IIC operation reservation adapter status register C_0	ICSRC_0	8	H'FEB3	IIC_0	16	2
IIC operation reservation adapter data register_0	ICDRX_0	8	H'FEB4	IIC_0	16	2
IIC operation reservation adapter command register_0	ICCMD_0	8	H'FEB5	IIC_0	16	2
I <sup>2</sup> C data shift register_0	ICDRS_0	8	H'FEB6	IIC_0	16	2
IIC operation reservation adapter count register_0	ICCNT_0	8	H'FEB7	IIC_0	16	2
IIC operation reservation adapter control register_1	ICCRX_1	8	H'FEB8	IIC_1	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
IIC operation reservation adapter status register A_1	ICSRA_1	8	H'FEB9	IIC_1	16	2
IIC operation reservation adapter status register B_1	ICSRB_1	8	H'FEBA	IIC_1	16	2
IIC operation reservation adapter status register C_1	ICSRC_1	8	H'FEBB	IIC_1	16	2
IIC operation reservation adapter data register_1	ICDRX_1	8	H'FEBC	IIC_1	16	2
IIC operation reservation adapter command register_1	ICCMD_1	8	H'FEBD	IIC_1	16	2
I <sup>2</sup> C data shift register_1	ICDRS_1	8	H'FEBE	IIC_1	16	2
IIC operation reservation adapter count register_1	ICCNT_1	8	H'FEBF	IIC_1	16	2
Serial enhanced mode register_0	SEMR_0	8	H'FED0	SCI_0	8	2
Serial RFU enable register 0	SCIDTER_0	8	H'FED1	SCI_0	8	2
Serial enhanced mode register_2	SEMR_2	8	H'FED2	SCI_2	8	2
Serial RFU enable register 2	SCIDTER_2	8	H'FED3	SCI_2	8	2
CRC control register	CRCCR	8	H'FED4	CRC	8	2
CRC data input register	CRCDIR	8	H'FED5	CRC	8	2
CRC data output register H	CRCDORH	8	H'FED6	CRC	8	2
CRC data output register L	CRCDORL	8	H'FED7	CRC	8	2
Keyboard comparator control register	KBCOMP	8	H'FEE4	A/D converter	8	2
Serial interface control register	SCICR	8	H'FEE5	SCI_1	8	2
Interrupt control register D	ICRD	8	H'FEE7	INT	8	2
Interrupt control register A	ICRA	8	H'FEE8	INT	8	2
Interrupt control register B	ICRB	8	H'FEE9	INT	8	2
Interrupt control register C	ICRC	8	H'FEEA	INT	8	2
IRQ status register	ISR	8	H'FEEB	INT	8	2
IRQ sense control register H	ISCRH	8	H'FEEC	INT	8	2
IRQ sense control register L	ISCLR	8	H'FEED	INT	8	2



Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
DTC enable register A	DTCERA	8	H'FEEE	DTC	8	2
DTC enable register B	DTCERB	8	H'FE EF	DTC	8	2
DTC enable register C	DTCERC	8	H'FEF0	DTC	8	2
DTC enable register D	DTCERD	8	H'FEF1	DTC	8	2
DTC enable register E	DTCERE	8	H'FEF2	DTC	8	2
DTC vector register	DTVECR	8	H'FEF3	DTC	8	2
Address break control register	ABRKCR	8	H'FEF4	INT	8	2
Break address register A	BARA	8	H'FEF5	INT	8	2
Break address register B	BARB	8	H'FEF6	INT	8	2
Break address register C	BARC	8	H'FEF7	INT	8	2
IRQ enable register 16	IER16	8	H'FEF8	INT	8	2
IRQ status register 16	ISR16	8	H'FEF9	INT	8	2
IRQ sense control register 16H	ISCR16H	8	H'FEFA	INT	8	2
IRQ sense control register 16L	ISCR16L	8	H'FEFB	INT	8	2
IRQ sense port select register 16	ISSR16	8	H'FEFC	INT	8	2
IRQ sense port select register	ISSR	8	H'FEFD	INT	8	2
Port control register 0	PTCNT0	8	H'FEFE		8	2
Bus control register 2	BCR2	8	H'FF80	BSC	8	2
Wait state control register 2	WSCR2	8	H'FF81	BSC	8	2
Peripheral clock select register	PCSR	8	H'FF82	PWM	8	2
System control register 2	SYSCR2	8	H'FF83	SYSTEM	8	2
Standby control register	SBYCR	8	H'FF84	SYSTEM	8	2
Low power control register	LPWRCR	8	H'FF85	SYSTEM	8	2
Module stop control register H	MSTPCRH	8	H'FF86	SYSTEM	8	2
Module stop control register L	MSTPCRL	8	H'FF87	SYSTEM	8	2
Flash memory control register 1	FLMCR1	8	H'FF80	FLASH	8	2
Flash memory control register 2	FLMCR2	8	H'FF81	FLASH	8	2
Erase block setting register 1	EBR1	8	H'FF82	FLASH	8	2
Erase block setting register 2	EBR2	8	H'FF83	FLASH	8	2
Serial mode register_1	SMR_1	8	H'FF88	SCI_1	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
I <sup>2</sup> C bus control register_1	ICCR_1	8	H'FF88	IIC_1	8	2
Bit rate register_1	BRR_1	8	H'FF89	SCI_1	8	2
I <sup>2</sup> C bus status register_1	ICSR_1	8	H'FF89	IIC_1	8	2
Serial control register_1	SCR_1	8	H'FF8A	SCI_1	8	2
Transmit data register_1	TDR_1	8	H'FF8B	SCI_1	8	2
Serial status register_1	SSR_1	8	H'FF8C	SCI_1	8	2
Receive data register_1	RDR_1	8	H'FF8D	SCI_1	8	2
Smart card mode register_1	SCMR_1	8	H'FF8E	SCI_1	8	2
I <sup>2</sup> C bus data register_1	ICDR_1	8	H'FF8E	IIC_1	8	2
Second slave address register_1	SARX_1	8	H'FF8E	IIC_1	8	2
I <sup>2</sup> C bus mode register_1	ICMR_1	8	H'FF8F	IIC_1	8	2
Slave address register_1	SAR_1	8	H'FF8F	IIC_1	8	2
Timer interrupt enable register	TIER	8	H'FF90	FRT	8	2
Timer control/status register	TCSR	8	H'FF91	FRT	8	2
Free-running counter H	FRCH	8	H'FF92	FRT	8	2
Free-running counter L	FRCL	8	H'FF93	FRT	8	2
Output control register AH	OCRAH	8	H'FF94	FRT	8	2
Output control register BH	OCRBH	8	H'FF94	FRT	8	2
Output control register AL	OCRAL	8	H'FF95	FRT	8	2
Output control register BL	OCRBL	8	H'FF95	FRT	8	2
Timer control register	TCR	8	H'FF96	FRT	8	2
Timer output compare control register	TOCR	8	H'FF97	FRT	8	2
Input capture register AH	ICRAH	8	H'FF98	FRT	8	2
Output control register ARH	OCRARH	8	H'FF98	FRT	8	2
Input capture register AL	ICRAL	8	H'FF99	FRT	8	2
Output control register ARL	OCRARL	8	H'FF99	FRT	8	2
Input capture register BH	ICRBH	8	H'FF9A	FRT	8	2
Output control register AFH	OCRAFH	8	H'FF9A	FRT	8	2
Input capture register BL	ICRBL	8	H'FF9B	FRT	8	2
Output control register AFL	OCRAFL	8	H'FF9B	FRT	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Input capture register CH	ICRCH	8	H'FF9C	FRT	8	2
Output compare register DMH	OCRDMH	8	H'FF9C	FRT	8	2
Input capture register CL	ICRCL	8	H'FF9D	FRT	8	2
Output compare register DML	OCRDML	8	H'FF9D	FRT	8	2
Input capture register DH	ICRDH	8	H'FF9E	FRT	8	2
Input capture register DL	ICRDL	8	H'FF9F	FRT	8	2
Serial mode register_2	SMR_2	8	H'FFA0	SCI_2	8	2
PWMX (D/A) control register	DACR	8	H'FFA0	PWMX	8	2
PWMX (D/A) data register AH	DADRAH	8	H'FFA0	PWMX	8	2
PWMX (D/A) data register AL	DADRAL	8	H'FFA1	PWMX	8	2
Bit rate register_2	BRR_2	8	H'FFA1	SCI_2	8	2
Serial control register_2	SCR_2	8	H'FFA2	SCI_2	8	2
Transmit data register_2	TDR_2	8	H'FFA3	SCI_2	8	2
Serial status register_2	SSR_2	8	H'FFA4	SCI_2	8	2
Receive data register_2	RDR_2	8	H'FFA5	SCI_2	8	2
Smart card mode register_2	SCMR_2	8	H'FFA6	SCI_2	8	2
PWMX (D/A) data register BH	DADRBH	8	H'FFA6	PWMX	8	2
PWMX (D/A) counter H	DACNTH	8	H'FFA6	PWMX	8	2
PWMX (D/A) data register BL	DADRBL	8	H'FFA7	PWMX	8	2
PWMX (D/A) counter L	DACNTL	8	H'FFA7	PWMX	8	2
Timer control/status register_0	TCSR_0	8	H'FFA8	WDT	8	2
Timer counter_0	TCNT_0	8	H'FFA8 (write)	WDT	8	2
Timer counter_0	TCNT_0	8	H'FFA9 (read)	WDT	8	2
Port A output data register	PAODR	8	H'FFAA	PORT	8	2
Port A input data register	PAPIN	8	H'FFAB	PORT	8	2
Port A data direction register	PADDR	8	H'FFAB	PORT	8	2
Port 1 pull-up MOS control register	P1PCR	8	H'FFAC	PORT	8	2
Port 2 pull-up MOS control register	P2PCR	8	H'FFAD	PORT	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Port 3 pull-up MOS control register	P3PCR	8	H'FFAE	PORT	8	2
Port 1 data direction register	P1DDR	8	H'FFB0	PORT	8	2
Port 2 data direction register	P2DDR	8	H'FFB1	PORT	8	2
Port 1 data register	P1DR	8	H'FFB2	PORT	8	2
Port 2 data register	P2DR	8	H'FFB3	PORT	8	2
Port 3 data direction register	P3DDR	8	H'FFB4	PORT	8	2
Port 4 data direction register	P4DDR	8	H'FFB5	PORT	8	2
Port 3 data register	P3DR	8	H'FFB6	PORT	8	2
Port 4 data register	P4DR	8	H'FFB7	PORT	8	2
Port 5 data direction register	P5DDR	8	H'FFB8	PORT	8	2
Port 6 data direction register	P6DDR	8	H'FFB9	PORT	8	2
Port 5 data register	P5DR	8	H'FFBA	PORT	8	2
Port 6 data register	P6DR	8	H'FFBB	PORT	8	2
Port 8 data direction register	P8DDR	8	H'FFBD	PORT	8	2
Port 7 input data register	P7PIN	8	H'FFBE	PORT	8	2
Port 8 data register	P8DR	8	H'FFBF	PORT	8	2
Port 9 data direction register	P9DDR	8	H'FFC0	PORT	8	2
Port 9 data register	P9DR	8	H'FFC1	PORT	8	2
IRQ enable register	IER	8	H'FFC2	INT	8	2
Serial timer control register	STCR	8	H'FFC3	SYSTEM	8	2
System control register	SYSCR	8	H'FFC4	SYSTEM	8	2
Mode control register	MDCR	8	H'FFC5	SYSTEM	8	2
Bus control register	BCR	8	H'FFC6	BSC	8	2
Wait state control register	WSCR	8	H'FFC7	BSC	8	2
Timer control register_0	TCR_0	8	H'FFC8	TMR_0	16	2
Timer control register_1	TCR_1	8	H'FFC9	TMR_1	16	2
Timer control/status register_0	TCSR_0	8	H'FFCA	TMR_0	16	2
Timer control/status register_1	TCSR_1	8	H'FFCB	TMR_1	16	2
Time constant register A_0	TCORA_0	8	H'FFCC	TMR_0	16	2
Time constant register A_1	TCORA_1	8	H'FFCD	TMR_1	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Time constant register B_0	TCORB_0	8	H'FFCE	TMR_0	16	2
Time constant register B_1	TCORB_1	8	H'FFCF	TMR_1	16	2
Timer counter_0	TCNT_0	8	H'FFD0	TMR_0	16	2
Timer counter_1	TCNT_1	8	H'FFD1	TMR_1	16	2
PWM output enable register B	PWOERB	8	H'FFD2	PWM	8	2
PWM output enable register A	PWOERA	8	H'FFD3	PWM	8	2
PWM data polarity register B	PWDPRB	8	H'FFD4	PWM	8	2
PWM data polarity register A	PWDPRA	8	H'FFD5	PWM	8	2
PWM register select	PWSL	8	H'FFD6	PWM	8	2
PWM data registers 0–15	PWDR0 to 15	8	H'FFD7	PWM	8	2
Serial mode register_0	SMR_0	8	H'FFD8	SCI_0	8	2
I <sup>2</sup> C bus control register_0	ICCR_0	8	H'FFD8	IIC_0	8	2
Bit rate register_0	BRR_0	8	H'FFD9	SCI_0	8	2
I <sup>2</sup> C bus status register_0	ICSR_0	8	H'FFD9	IIC_0	8	2
Serial control register_0	SCR_0	8	H'FFDA	SCI_0	8	2
Transmit data register_0	TDR_0	8	H'FFDB	SCI_0	8	2
Serial status register_0	SSR_0	8	H'FFDC	SCI_0	8	2
Receive data register_0	RDR_0	8	H'FFDD	SCI_0	8	2
Smart card mode register_0	SCMR_0	8	H'FFDE	SCI_0	8	2
I <sup>2</sup> C bus data register_0	ICDR_0	8	H'FFDE	IIC_0	8	2
Second slave address register_0	SARX_0	8	H'FFDE	IIC_0	8	2
I <sup>2</sup> C bus mode register_0	ICMR_0	8	H'FFDF	IIC_0	8	2
Slave address register_0	SAR_0	8	H'FFDF	IIC_0	8	2
A/D data register AH	ADDRAH	8	H'FFE0	A/D converter	8	2
A/D data register AL	ADDRAL	8	H'FFE1	A/D converter	8	2
A/D data register BH	ADDRBH	8	H'FFE2	A/D converter	8	2
A/D data register BL	ADDRBL	8	H'FFE3	A/D converter	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
A/D data register CH	ADDRCH	8	H'FFE4	A/D converter	8	2
A/D data register CL	ADDRCL	8	H'FFE5	A/D converter	8	2
A/D data register DH	ADDRDH	8	H'FFE6	A/D converter	8	2
A/D data register DL	ADDRDL	8	H'FFE7	A/D converter	8	2
A/D control/status register	ADCSR	8	H'FFE8	A/D converter	8	2
A/D control register	ADCR	8	H'FFE9	A/D converter	8	2
Timer control/status register_1	TCSR_1	8	H'FFEA	WDT_1	8	2
Timer counter_1	TCNT_1	8	H'FFEA (write)	WDT_1	8	2
Timer counter_1	TCNT_1	8	H'FFEB (read)	WDT_1	8	2
Keyboard matrix interrupt mask register 6	KMIMR6	8	H'FFF1	INT	8	2
Port 6 pull-up MOS control register	KMPCR6	8	H'FFF2	PORT	8	2
Keyboard matrix interrupt mask register A	KMIMRA	8	H'FFF3	INT	8	2
Wakeup event interrupt mask register 3	WUEMR3	8	H'FFF4	INT	8	2
Timer control register_X	TCR_X	8	H'FFF0	TMR_X	16	2
Timer control register_Y	TCR_Y	8	H'FFF0	TMR_Y	16	2
Timer control/status register_X	TCSR_X	8	H'FFF1	TMR_X	16	2
Timer control/status register_Y	TCSR_Y	8	H'FFF1	TMR_Y	16	2
Input capture register R	TICRR	8	H'FFF2	TMR_X	16	2
Time constant register A_Y	TCORA_Y	8	H'FFF2	TMR_Y	16	2
Input capture register F	TICRF	8	H'FFF3	TMR_X	16	2
Time constant register B_Y	TCORB_Y	8	H'FFF3	TMR_Y	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Timer counter_X	TCNT_X	8	H'FFF4	TMR_X	16	2
Timer counter_Y	TCNT_Y	8	H'FFF4	TMR_Y	16	2
Time constant register	TCORC	8	H'FFF5	TMR_X	16	2
Timer input select register	TISR	8	H'FFF5	TMR_Y	16	2
Time constant register A_X	TCORA_X	8	H'FFF6	TMR_X	16	2
Time constant register B_X	TCORB_X	8	H'FFF7	TMR_X	16	2
D/A data register 0	DADR0	8	H'FFF8	D/A converter	8	2
D/A data register 1	DADR1	8	H'FFF9	D/A converter	8	2
D/A control register	DACR	8	H'FFFA	D/A converter	8	2
Timer connection register I	TCONRI	8	H'FFFC	Timer connection	8	2
Timer connection register O	TCONRO	8	H'FFFD	Timer connection	8	2
Timer connection register S	TCONRS	8	H'FFFE	Timer connection	8	2
Edge sense register	SEDGR	8	H'FFFF	Timer connection	8	2

## 28.2 Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, so 64-bit registers are shown as 8 lines and 16-bit registers as 2 lines.

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
CMDR0	Start	Host	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	MCIF
CMDR1	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	
CMDR2	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	
CMDR3	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
CMDR4	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
CMDR5	CRC	CRC	CRC	CRC	CRC	CRC	CRC	End	
CMDSTRT	—	—	—	—	—	—	—	START	
OPCR	CMDOFF	—	RD_CONTI	DATAEN	—	—	—	—	
CSTR	BUSY	FIFO_FULL	FIFO_EMPTY	CWRE	DTBUSY	DTBUSY_TU	—	REQ	
INTCR0	FEIE	FFIE	DRPIE	DTIE	CRPIE	CMDIE	DBSYIE	BTIE	
INTCR1	INTRQ2E	INTRQ1E	INTRQ0E	—	—	CRCERIE	DTERIE	CTERIE	
INTSTR0	FEI	FFI	DRPI	DTI	CRPI	CMDI	DBSYI	BTI	
INTSTR1	—	—	—	—	—	CRCERI	DTERI	CTERI	
CLKON	CLKON	—	—	—	—	CSEL2	CSEL1	CSEL0	
CTOCR	—	—	—	—	—	—	—	CTSEL0	
IOMCR	SPCNUM	CHIPSA	—	—	—	—	DIRME	MMCPE	
TBCR	—	—	—	—	C3	C2	C1	C0	
MODER	—	—	—	—	—	—	—	SPI	
CMDTYR	—	—	TY5	TY4	TY3	TY2	TY1	TY0	
RSPTYR	—	—	RTY5	RTY4	—	RTY2	RTY1	RTY0	
TBNCRH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
TBNCRL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
RSPR0	Bit 135	Bit 134	Bit 133	Bit 132	Bit 131	Bit 130	Bit 129	Bit 128	
RSPR1	Bit 127	Bit 126	Bit 125	Bit 124	Bit 123	Bit 122	Bit 121	Bit 120	
RSPR2	Bit 119	Bit 118	Bit 117	Bit 116	Bit 115	Bit 114	Bit 113	Bit 112	
RSPR3	Bit 111	Bit 110	Bit 109	Bit 108	Bit 107	Bit 106	Bit 105	Bit 104	
RSPR4	Bit 103	Bit 102	Bit 101	Bit 100	Bit 99	Bit 98	Bit 97	Bit 96	
RSPR5	Bit 95	Bit 94	Bit 93	Bit 92	Bit 91	Bit 90	Bit 89	Bit 88	
RSPR6	Bit 87	Bit 86	Bit 85	Bit 84	Bit 83	Bit 82	Bit 81	Bit 80	
RSPR7	Bit 79	Bit 78	Bit 77	Bit 76	Bit 75	Bit 74	Bit 73	Bit 72	



Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module	
RSPR8	Bit 71	Bit 70	Bit 69	Bit 68	Bit 67	Bit 66	Bit 65	Bit 64	MCIF	
RSPR9	Bit 63	Bit 62	Bit 61	Bit 60	Bit 59	Bit 58	Bit 57	Bit 56		
RSPR10	Bit 55	Bit 54	Bit 53	Bit 52	Bit 51	Bit 50	Bit 49	Bit 48		
RSPR11	Bit 47	Bit 46	Bit 45	Bit 44	Bit 43	Bit 42	Bit 41	Bit 40		
RSPR12	Bit 39	Bit 38	Bit 37	Bit 36	Bit 35	Bit 34	Bit 33	Bit 32		
RSPR13	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24		
RSPR14	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16		
RSPR15	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8		
RSPR16	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
PSPRD	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
DTOUTRH	DTOUT15	DTOUT14	DTOUT13	DTOUT12	DTOUT11	DTOUT10	DTOUT9	DTOUT8		
DTOUTRL	DTOUT7	DTOUT6	DTOUT5	DTOUT4	DTOUT3	DTOUT2	DTOUT1	DTOUT0		
EP4PKTSZR	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0		USB
EPDR0S	D7	D6	D5	D4	D3	D2	D1	D0		
FVSR0SH	—	—	—	—	—	—	N9	N8		
FVSR0SL	N7	N6	N5	N4	N3	N2	N1	N0		
USBIER1	—	—	—	—	—	—	SETCE	SETIE		
USBIFR1	—	—	—	—	—	—	SETC	SETI		
USBMDCR	—	—	—	—	—	—	SCME	SETICNT		
USBCR1	—	—	—	—	—	—	VBUSS	CK48READY		
CONFV	—	—	CONFV0	INTV1	INTV0	ALTV2	ALTV1	ALTV0		
UDTRFR	—	—	—	—	—	—	—	EP5UDTR		
UPRTCR	—	—	—	—	—	PCNMD2	PCNMD1	PCNMD0		
UTESTR0	TEST15	TEST14	TEST13	TEST12	TEST11	TEST10	TEST9	TEST8		
UTESTR1	TEST7	TEST6	TEST5	TEST4	TEST3	TEST2	TEST1	TEST0		
EPDR3	D7	D6	D5	D4	D3	D2	D1	D0		
FVSR3H	—	—	—	—	—	—	N9	N8		
FVSR3L	N7	N6	N5	N4	N3	N2	N1	N0		
EPDR2	D7	D6	D5	D4	D3	D2	D1	D0		
FVSR2H	—	—	—	—	—	—	N9	N8		
FVSR2L	N7	N6	N5	N4	N3	N2	N1	N0		

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
EPSZR1	EP1SZ3	EP1SZ2	EP1SZ1	EP1SZ0	EP2SZ3	EP2SZ2	EP2SZ1	EP2SZ0	USB
EPDR1	D7	D6	D5	D4	D3	D2	D1	D0	
FVSR1H	—	—	—	—	—	—	N9	N8	
FVSR1L	N7	N6	N5	N4	N3	N2	N1	N0	
EPDR00	D7	D6	D5	D4	D3	D2	D1	D0	
FVSR00H	—	—	—	—	—	—	N9	N8	
FVSR00L	N7	N6	N5	N4	N3	N2	N1	N0	
EPDR0I	D7	D6	D5	D4	D3	D2	D1	D0	
FVSR0IH	—	—	—	—	—	—	N9	N8	
FVSR0IL	N7	N6	N5	N4	N3	N2	N1	N0	
PTTER0	—	—	EP4TE	EP3TE	EP2TE	EP1TE	EP0ITE	—	
USBIER0	—	UDTRE	BRSTE	SOFE	SPNDE	TFE	TSE	SETUPE	
USBIFR0	TS	TF	UDTRF	BRSTF	SOFF	SPNDOF	SPNDIF	SETUPF	
TSFR0	—	EP5TS	EP4TS	EP3TS	EP2TS	EP1TS	EP0ITS	EP0OTS	
TFFR0	—	EP5TF	EP4TF	EP3TF	EP2TF	EP1TF	EP0ITF	EP0OTF	
USBCSR0	—	—	—	—	EP0STOP	EPIVLD	EP0OTC	—	
EPSTLR0	—	EP5STL	EP4STL	EP3STL	EP2STL	EP1STL	—	EP0STL	
EPDIR0	—	EP5DIR	EP4DIR	EP3DIR	EP2DIR	EP1DIR	—	—	
EPRSTR0	—	EP5RST	EP4RST	EP3RST	EP2RST	EP1RST	EP0IRST	EP0ORST	
DEVRSMR	—	—	—	—	—	—	RWUPS	DVR	
INTSELR0	TSELB	EPIBS2	EPIBS1	EPIBS0	TSELC	EPICS2	EPICS1	EPICS0	
USBCR0	FADSEL	—	—	UIFRST	—	—	FPLLRS	FSRST	
UPLLCR	—	—	PFSEL2	CKSEL2	CKSEL1	CKSEL0	PFSEL1	PFSEL0	
SUBMSTPBH	SMSTPB15	SMSTPB14	SMSTPB13	SMSTPB12	SMSTPB11	SMSTPB10	SMSTPB9	SMSTPB8	SYSTEM
SUBMSTPBL	SMSTPB7	SMSTPB6	SMSTPB5	SMSTPB4	SMSTPB3	SMSTPB2	SMSTPB1	SMSTPB0	
FSTR0	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	RFU
FSTR1	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	
FSTR2	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
FSTR3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DTCRA	IDE-A	IDE-B	PMD1	PMD0	Sz	BUD2	BUD1	BUD0	RFU
DTCRB	BOVF_RE	BOVF_WE	FULLE	EMPTYE	LOAD	MARK	REST	STCLR	
DTIDR	ID-A3	ID-A2	ID-A1	ID-A0	ID-B3	ID-B2	ID-B1	ID-B0	
DTSTRC	BOVF_R	BOVF_W	FULL	EMPTY	OVER_R	OVER_W	—	—	
DTIDSR	IDRW15	IDRW14	IDRW13	IDRW12	IDRW11	IDRW10	IDRW9	IDRW8	
DTIDSRB	IDRW7	IDRW6	IDRW5	IDRW4	IDRW3	IDRW2	IDRW1	IDRW0	
DTSTRA	—	—	—	—	DTF3	DTF2	DTF1	DTF0	
DTSTRB	—	—	—	—	DTEF3	DTEF2	DTEF1	DTEIE	
DTCRD	—	—	—	—	DTE3	DTE2	DTE1	DTE0	
DTIER	—	—	—	—	DTIE3	DTIE2	DTIE1	DTIE0	
DTRSR	CHS2	CHS1	CHS0	RS	POS1	POS0	STS1	STS0	
ICCRX_0	ICXE	CRIC	MRIC	SRIC	BBSYX/ CLR3	—/ CLR2	AASHIT/ CLR1	ACKBX/ CLR0	IIC_0
ICSRA_0	SDAO	SCLO	SDAI	SCLI	MSTX	TRSX	WAITX	ACKXE	
ICSRB_0	CREQ	CERR	STOP	ABRT	ALST	DERR	TOVR	NACK	
ICSRC_0	MTREQ	MRREQ	STREQ	SRREQ	MASX	TDRE	SDRF	RDRF	
ICDRX_0	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	
ICCMD_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ICDRS_0	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	
ICCNT_0	CNTE	STOPIX	CNTS1	CNTS0	BBC3	BBC2	BBC1	BBC0	
ICCRX_1	ICXE	CRIC	MRIC	SRIC	BBSYX/ CLR3	—/ CLR2	AASHIT/ CLR1	ACKBX/ CLR0	IIC_1
ICSRA_1	SDAO	SCLO	SDAI	SCLI	MSTX	TRSX	WAITX	ACKXE	
ICSRB_1	CREQ	CERR	STOP	ABRT	ALST	DERR	TOVR	NACK	
ICSRC_1	MTREQ	MRREQ	STREQ	SRREQ	MASX	TDRE	SDRF	RDRF	
ICDRX_1	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	
ICCMD_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ICDRS_1	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	
ICCNT_1	CNTE	STOPIX	CNTS1	CNTS0	BBC3	BBC2	BBC1	BBC0	
SEMR_0	SSE	—	—	ABCS	ACS4	ACS2	ACS1	ACS0	SCI_0
SCIDTER_0	TDRE_DTE	RDRF_DTE	—	—	—	—	—	—	
SEMR_2	SSE	—	—	ABCS	ACS4	ACS2	ACS1	ACS0	SCI_2
SCIDTER_2	TDRE_DTE	RDRF_DTE	—	—	—	—	—	—	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
CRCCR	DORCLR	—	—	—	—	LMS	G1	G0	CRC
CRCDIR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
CRCDORH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
CRCDORL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
KBCOMP	—	—	—	SCANE	KBADE	KBCH2	KBCH1	KBCH0	A/D converter
SCICR	IrE	IrCKS2	IrCKS1	IrCKS0	—	—	—	—	SCI_1
ICRD	IRQ8-11	IRQ12-15	—	—	—	—	—	MCIF	INT
ICRA	IRQ0	IRQ1	IRQ2-3	IRQ4-5	IRQ6-7	DTC	WDT_0	WDT_1	
ICRB	A/D converter	FRT	—	TMR_X	TMR_0	TMR_1	TMR_Y	—	
ICRC	SCI_0	SCI_1	SCI_2	IIC_0	IIC_1	—	—	USB	
ISR	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
ISCRH	IRQ7SCB	IRQ7SCA	IRQ6SCB	IRQ6SCA	IRQ5SCB	IRQ5SCA	IRQ4SCB	IRQ4SCA	
ISURL	IRQ3SCB	IRQ3SCA	IRQ2SCB	IRQ2SCA	IRQ1SCB	IRQ1SCA	IRQ0SCB	IRQ0SCA	
DTCERA	IRQ0	IRQ1	IRQ2	IRQ3	ADI	ICIA	ICIB	OCIA	DTC
DTCERB	OCIB	IICM0	IICR0	IICT0	—	CMIA0	CMIB0	CMIA1	
DTCERC	CMIB1	CMIAY	CMIBY	—	CMIAX	RXI0	TXI0	RXI1	
DTCERD	TXI1	RXI2	TXI2	IICM1	IICR1	IICT1	MMCIA	CMIBX	
DTCERE	USBI0	USBI1	USBI2	USBI3	—	—	—	—	
DTVECR	SWDTE	DTVEC6	DTVEC5	DTVEC4	DTVEC3	DTVEC2	DTVEC1	DTVEC0	
ABRKCR	CMF	—	—	—	—	—	—	BIE	INT
BARA	A23	A22	A21	A20	A19	A18	A17	A16	
BARB	A15	A14	A13	A12	A11	A10	A9	A8	
BARC	A7	A6	A5	A4	A3	A2	A1	—	
IER16	IRQ15E	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E	
ISR16	IRQ15F	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F	
ISCR16H	IRQ15SCB	IRQ15SCA	IRQ14SCB	IRQ14SCA	IRQ13SCB	IRQ13SCA	IRQ12SCB	IRQ12SCA	
ISCR16L	IRQ11SCB	IRQ11SCA	IRQ10SCB	IRQ10SCA	IRQ9SCB	IRQ9SCA	IRQ8SCB	IRQ8SCA	
ISSR16	ISS15	ISS14	ISS13	ISS12	ISS11	ISS10	ISS9	ISS8	
ISSR	ISS7	ISS6	ISS5	ISS4	ISS3	ISS2	ISSR1	ISS0	
PTCNT0	TMI0S	TMI1S	TMIXS	TMIYS	MMCS	—	—	—	PORT

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
BCR2	OWEAC	OWENC	ABWCP	ASTCP	ADFULLE	EXCKS	BUSDIVE	CPCSE	BSC
WSCR2	WMS10	WC11	WC10	WMS21	WMS20	WC22	WC21	WC20	
PCSR	—	—	PWCKXB	PWCKXA	—	PWCKB	PWCKA	—	PWM
SYSCR2	KWUL1	KWUL0	P6PUE	—	—	CKCHGE	—	PLCKS	SYSTEM
SBYCR	SSBY	STS2	STS1	STS0	DTSPEED	SCK2	SCK1	SCK0	
LPWRCR	DTON	LSON	NESEL	EXCLE	—	—	—	—	
MSTPCRH	MSTP15	MSTP14	MSTP13	MSTP12	MSTP11	MSTP10	MSTP9	MSTP8	
MSTPCRL	MSTP7	MSTP6	MSTP5	MSTP4	MSTP3	MSTP2	MSTP1	MSTP0	
FLMCR1	FWE	SWE	—	—	EV	PV	E	P	FLASH
FLMCR2	FLER	—	—	—	—	—	ESU	PSU	
EBR1	—	—	—	—	EB11	EB10	EB9	EB8	
EBR2	EB7	EB6	EB5	EB4	EB3	EB2	EB1	EB0	
SMR_1*	C/ $\bar{A}$ (GM)	CHR/ (BLK)	PE/ (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP/ (BCP0)	CKS1 (CKS1)	CKS0 (CKS0)	SCI_1
ICCR_1	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_1
BRR_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	SCI_1
ICSR_1	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	IIC_1
SCR_1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	SCI_1
TDR_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SSR_1*	TDRE (TDRE)	RDRF (RDRF)	ORER (ORER)	FER (ERS)	PER (PER)	TEND (TEND)	MPB (MPB)	MPBT (MPBT)	
RDR_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SCMR_1	—	—	—	—	SDIR	SINV	—	SMIF	
ICDR_1	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	IIC_1
SARX_1	SVAX6	SVAX5	SVAX4	SVAX3	SVAX2	SVAX1	SVAX0	FSX	
ICMR_1	MLS	WAIT	CKS2	CKS1	CKS0	BC2	BC1	BC0	
SAR_1	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS	
TIER	ICIAE	ICIBE	ICICE	ICIDE	OCIAE	OCIBE	OVIE	—	FRT
TCSR	ICFA	ICFB	ICFC	ICFD	OCFA	OCFB	OVF	CCLRA	
FRCH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
FRCL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
OCRAH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
OCRBH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
OCRAL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	FRT
OCRBL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCR	IEDGA	IEDGB	IEDGC	IEDGD	BUFEA	BUFEB	CKS1	CKS0	
TOCR	ICRDMS	OCRAMS	ICRS	OCRS	OEA	OEB	OLVLA	OLVLB	
ICRAH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
OCRARH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
ICRAL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
OCRARL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ICRBH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
OCRAFH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
ICRBL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
OCRAFL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ICRCH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
OCRDMH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
ICRCL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
OCRDML	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ICRDH	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
ICRDL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SMR_2*	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1 (CKS1)	CKS0 (CKS0)	SCI_2
DACR	—	PWME	—	—	OEB	OEA	OS	CKS	PWMX
DADRAH	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	
DADRAL	DA5	DA4	DA3	DA2	DA1	DA0	CFS	—	
BRR_2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	SCI_2
SCR_2	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SSR_2*	TDRE (TDRE)	RDRF (RDRF)	ORER (ORER)	FER (ERS)	PER (PER)	TEND (TEND)	MPB (MPB)	MPBT (MPBT)	
RDR_2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SCMR_2	—	—	—	—	SDIR	SINV	—	SMIF	
DADRBH	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	PWMX
DACNTH	UC7	UC6	UC5	UC4	UC3	UC2	UC1	UC0	
DADRBL	DA5	DA4	DA3	DA2	DA1	DA0	CFS	REGS	
DACNTL	UC8	UC9	UC10	UC11	UC12	UC13	—	REGS	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
TCSR_0	OVF	WT/IT	TME	—	RST/ $\overline{\text{NMI}}$	CKS2	CKS1	CKS0	WDT_0
TCNT_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
PAODR	—	—	—	—	—	—	PA1ODR	PA0ODR	PORT
PAPIN	—	—	—	—	—	—	PA1PIN	PA0PIN	
PADDR	—	—	—	—	—	—	PA1DDR	PA0DDR	
P1PCR	P17PCR	P16PCR	P15PCR	P14PCR	P13PCR	P12PCR	P11PCR	P10PCR	
P2PCR	P27PCR	P26PCR	P25PCR	P24PCR	P23PCR	P22PCR	P21PCR	P20PCR	
P3PCR	P37PCR	P36PCR	P35PCR	P34PCR	P33PCR	P32PCR	P31PCR	P30PCR	
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	
P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	P27DR	P26DR	P25DR	P24DR	P23DR	P22DR	P21DR	P20DR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P4DDR	P47DDR	P46DDR	P45DDR	P44DDR	P43DDR	P42DDR	P41DDR	P40DDR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P4DR	P47DR	P46DR	P45DR	P44DR	P43DR	P42DR	P41DR	P40DR	
P5DDR	P57DDR	P56DDR	P55DDR	P54DDR	P53DDR	P52DDR	P51DDR	P50DDR	
P6DDR	P67DDR	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	
P5DR	P57DR	P56DR	P55DR	P54DR	P53DR	P52DR	P51DR	P50DR	
P6DR	P67DR	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
P8DDR	P87DDR	P86DDR	P85DDR	P84DDR	P83DDR	P82DDR	P81DDR	P80DDR	
P7PIN	P77PIN	P76PIN	P75PIN	P74PIN	P73PIN	P72PIN	—	—	
P8DR	P87DR	P86DR	P85DR	P84DR	P83DR	P82DR	P81DR	P80DR	
P9DDR	P97DDR	P96DDR	P95DDR	P94DDR	P93DDR	P92DDR	P91DDR	P90DDR	
P9DR	P97DR	P96DR	P95DR	P94DR	P93DR	P92DR	P91DR	P90DR	
IER	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	INT
STCR	—	IICX1	IICX0	IICE	FLSHE	—	ICKS1	ICKS0	SYSTEM
SYSCR	CS256E	IOSE	INTM1	INTM0	XRST	NMIEG	KINWUE	RAME	
MDCR	EXPE	—	—	—	—	MDS2	MDS1	MDS0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
BCR	—	ICIS	BRSTRM	BRSTS1	BRSTS0	CFE	IOS1	IOS0	BSC
WSCR	ABW256	AST256	ABW	AST	WMS1	WMS0	WC1	WC0	
TCR_0	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_0, TMR_1
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	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORA_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORB_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORB_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCNT_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCNT_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
PWOERB	OE15	OE14	OE13	OE12	OE11	OE10	OE9	OE8	PWM
PWOERA	OE7	OE6	OE5	OE4	OE3	OE2	OE1	OE0	
PWDPRB	OS15	OS14	OS13	OS12	OS11	OS10	OS9	OS8	
PWDPRA	OS7	OS6	OS5	OS4	OS3	OS2	OS1	OS0	
PWSL	PWCKE	PWCKS	—	—	RS3	RS2	RS1	RS0	
PWDR0-15	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SMR_0*	C/ $\bar{A}$ (GM)	CHR (BLK)	PE (PE)	O/ $\bar{E}$ (O/ $\bar{E}$ )	STOP (BCP1)	MP (BCP0)	CKS1 (CKS1)	CKS0 (CKS0)	SCI_0
ICCR_0	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_0
BRR_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	SCI_0
ICSR_0	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	IIC_0
SCR_0	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	SCI_0
TDR_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SSR_0*	TDRE (TDRE)	RDRF (RDRF)	ORER (ORER)	FER (ERS)	PER (PER)	TEND (TEND)	MPB (MPB)	MPBT (MPBT)	
RDR_0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
SCMR_0	—	—	—	—	SDIR	SINV	—	SMIF	
ICDRS_0	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	IIC_0
SARX_0	SVAX6	SVAX5	SVAX4	SVAX3	SVAX2	SVAX1	SVAX0	FSX	
ICMR_0	MLS	WAIT	CKS2	CKS1	CKS0	BC2	BC1	BC0	
SAR_0	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS	



Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
ADDRAH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D converter
ADDRAL	AD1	AD0	—	—	—	—	—	—	
ADDRBH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
ADDRBL	AD1	AD0	—	—	—	—	—	—	
ADDRCH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
ADDRCL	AD1	AD0	—	—	—	—	—	—	
ADDRDH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
ADDRDL	AD1	AD0	—	—	—	—	—	—	
ADCSR	ADF	ADIE	ADST	SCAN	CKS	CH2	CH1	CH0	
ADCR	TRGS1	TRGS0	—	—	—	—	—	—	
TCSR_1	OVF	WT/IT	TME	PSS	RST/NM $\bar{I}$	CKS2	CKS1	CKS0	WDT_1
TCNT_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
KMIMR6	KMIM7	KMIM6	KMIM5	KMIM4	KMIM3	KMIM2	KMIM1	KMIM0	INT
KMPCR6	KM7PCR	KM6PCR	KM5PCR	KM4PCR	KM3PCR	KM2PCR	KM1PCR	KM0PCR	PORT
KMIMRA	—	—	—	—	—	—	KMIM9	KMIM8	INT
WUEMR3	WUEM15	WUEM14	WUEM13	WUEM12	WUEM11	WUEM10	WUEM9	WUEM8	
TCR_X	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_Y, TMR_X
TCR_Y	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	
TCSR_X	CMFB	CMFA	OVF	ICF	OS3	OS2	OS1	OS0	
TCSR_Y	CMFB	CMFA	OVF	ICIE	OS3	OS2	OS1	OS0	
TICRR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORA_Y	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TICRF	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORB_Y	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCNT_X	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCNT_Y	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORC	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TISR	—	—	—	—	—	—	—	IS	
TCORA_X	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCORB_X	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Register									
Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DADR0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	D/A converter
DADR1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
DACR	DAOE1	DAOE0	DAE	—	—	—	—	—	
TCONRI	SIMOD1	SIMOD0	SCONE	ICST	HFINV	VFINV	HIINV	VIINV	Timer connection
TCONRO	HOE	VOE	CLOE	CBOE	HOINV	VOINV	CLOINV	CBOINV	
TCONRS	TMRX/Y	ISGENE	HOMOD1	HOMOD0	VOMOD1	VOMOD0	CLMOD1	CLMOD0	
SEDGR	VEDG	HEDG	CEDG	HFEDG	VFEDG	PREDG	IHI	IVI	

Note: \* Some bits have different names in normal mode and smart card interface mode. The bit name in smart card interface mode is enclosed in parentheses.

## 28.3 Register States in Each Operating Mode

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
CMDR0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	MCIF
CMDR1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CMDR2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CMDR3	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CMDR4	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CMDR5	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CMDSTRT	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
OPCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CSTR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
INTCR0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
INTCR1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
INTSTR0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
INTSTR1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CLKON	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CTOCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
IOMCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
TBCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
MODER	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
CMDTYR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPTYR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
TBNCRH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
TBNCRL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR3	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR4	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR5	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR6	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR7	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR8	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR9	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR10	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR11	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR12	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
RSPR13	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	MCIF
RSPR14	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR15	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RSPR16	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PSPRD	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DTOUTRH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DTOUTRL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
EP4PKTSZR	Initialized	—*1	—	—	—	—	—	—	Initialized	USB
EPDR0S	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR0SH	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR0SL	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBIER1	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBIFR1	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBMDCR	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBCR1	Initialized	—	—	—	—	—	—	—	Initialized	
CONFV	Initialized	—*1	—	—	—	—	—	—	Initialized	
UDTRFR	Initialized	—*1	—	—	—	—	—	—	Initialized	
UPRTCR	Initialized	—	—	—	—	—	—	—	Initialized	
UTESTR0	Initialized	—	—	—	—	—	—	—	Initialized	
UTESTR1	Initialized	—	—	—	—	—	—	—	Initialized	
EPDR3	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR3H	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR3L	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPDR2	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR2H	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR2L	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPSZR1	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPDR1	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR1H	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR1L	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPDR00	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR00H	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR00L	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPDR0I	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR0IH	Initialized	—*1	—	—	—	—	—	—	Initialized	
FVSR0IL	Initialized	—*1	—	—	—	—	—	—	Initialized	
PTTER0	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
USBIER0	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBIFR0	Initialized	—*1	—	—	—	—	—	—	Initialized	USB
TSFR0	Initialized	—*1	—	—	—	—	—	—	Initialized	
TFFR0	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBCSR0	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPSTLR0	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPDIR0	Initialized	—*1	—	—	—	—	—	—	Initialized	
EPRSTR0	Initialized	—	—	—	—	—	—	—	Initialized	
DEVRSMR	Initialized	—*1	—	—	—	—	—	—	Initialized	
INTSELR0	Initialized	—*1	—	—	—	—	—	—	Initialized	
USBCR0	Initialized	—	—	—	—	—	—	—	Initialized	
UPLLCR	Initialized	—	—	—	—	—	—	—	Initialized	
SUBMSTPBH	Initialized	—	—	—	—	—	—	—	Initialized	SYSTEM
SUBMSTPBL	Initialized	—	—	—	—	—	—	—	Initialized	
FSTR0	Initialized	—	—	—	—	—	—	—	Initialized	RFU
FSTR1	Initialized	—	—	—	—	—	—	—	Initialized	
FSTR2	Initialized	—	—	—	—	—	—	—	Initialized	
FSTR3	Initialized	—	—	—	—	—	—	—	Initialized	
DTCRA	Initialized	—	—	—	—	—	—	—	Initialized	
DTCRB	Initialized	—	—	—	—	—	—	—	Initialized	
DTIDR	Initialized	—	—	—	—	—	—	—	Initialized	
DTSTRC	Initialized	—	—	—	—	—	—	—	Initialized	
DTIDSRA	Initialized	—	—	—	—	—	—	—	Initialized	
DTIDSRB	Initialized	—	—	—	—	—	—	—	Initialized	
DTSTRA	Initialized	—	—	—	—	—	—	—	Initialized	
DTSTRB	Initialized	—	—	—	—	—	—	—	Initialized	
DTCRD	Initialized	—	—	—	—	—	—	—	Initialized	
DTIER	Initialized	—	—	—	—	—	—	—	Initialized	
DTRSR	Initialized	—	—	—	—	—	—	—	Initialized	
ICCRX_0	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
ICSRA_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICSRB_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICSRC_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICDRX_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICCMD_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICDRS_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICCNT_0	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
ICCRX_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
ICSRA_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICSRB_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICSRC_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICDRX_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICCMD_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICDRS_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICCNT_1	Initialized	—	—	—	—	—	—	—	Initialized	
SEMR_0	Initialized	—	—	—	—	—	—	—	Initialized	SCI_0
SCIDTER_0	Initialized	—	—	—	—	—	—	—	Initialized	
SEMR_2	Initialized	—	—	—	—	—	—	—	Initialized	SCI_2
SCIDTER_2	Initialized	—	—	—	—	—	—	—	Initialized	
CRCCR	Initialized	—	—	—	—	—	—	—	Initialized	CRC
CRCDIR	Initialized	—	—	—	—	—	—	—	Initialized	
CRCDORH	Initialized	—	—	—	—	—	—	—	Initialized	
CRCDORL	Initialized	—	—	—	—	—	—	—	Initialized	
KBCOMP	Initialized	—	—	—	—	—	—	—	Initialized	A/D converter
SCICR	Initialized	—	—	—	—	—	—	—	Initialized	SCI_1
ICRD	Initialized	—	—	—	—	—	—	—	Initialized	INT
ICRA	Initialized	—	—	—	—	—	—	—	Initialized	
ICRB	Initialized	—	—	—	—	—	—	—	Initialized	
ICRC	Initialized	—	—	—	—	—	—	—	Initialized	
ISR	Initialized	—	—	—	—	—	—	—	Initialized	
ISCRH	Initialized	—	—	—	—	—	—	—	Initialized	
ISCR_L	Initialized	—	—	—	—	—	—	—	Initialized	
DTCERA	Initialized	—	—	—	—	—	—	—	Initialized	DTC
DTCERB	Initialized	—	—	—	—	—	—	—	Initialized	
DTCERC	Initialized	—	—	—	—	—	—	—	Initialized	
DTCERD	Initialized	—	—	—	—	—	—	—	Initialized	
DTCERE	Initialized	—	—	—	—	—	—	—	Initialized	
DTVECR	Initialized	—	—	—	—	—	—	—	Initialized	
ABRKCR	Initialized	—	—	—	—	—	—	—	Initialized	INT
BARA	Initialized	—	—	—	—	—	—	—	Initialized	
BARB	Initialized	—	—	—	—	—	—	—	Initialized	
BARC	Initialized	—	—	—	—	—	—	—	Initialized	
IER16	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
ISR16	Initialized	—	—	—	—	—	—	—	Initialized	INT
ISCR16H	Initialized	—	—	—	—	—	—	—	Initialized	
ISCR16L	Initialized	—	—	—	—	—	—	—	Initialized	
ISSR16	Initialized	—	—	—	—	—	—	—	Initialized	
ISSR	Initialized	—	—	—	—	—	—	—	Initialized	
PTCNT0	Initialized	—	—	—	—	—	—	—	Initialized	PORT
BCR2	Initialized	—	—	—	—	—	—	—	Initialized	BSC
WSCR2	Initialized	—	—	—	—	—	—	—	Initialized	
PCSR	Initialized	—	—	—	—	—	—	—	Initialized	PWM
SYSCR2	Initialized	—	—	—	—	—	—	—	Initialized	SYSTEM
SBYCR	Initialized	—	—	—	—	—	—	—	Initialized	
LPWRCR	Initialized	—	—	—	—	—	—	—	Initialized	
MSTPCRH	Initialized	—	—	—	—	—	—	—	Initialized	
MSTPCRL	Initialized	—	—	—	—	—	—	—	Initialized	
FLMCR1	Initialized	— <sup>*3</sup>	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	FLASH
FLMCR2	Initialized	— <sup>*2 *3</sup>	Initialized <sup>*2</sup>	—	Initialized <sup>*2</sup>	Initialized <sup>*2</sup>	—	Initialized <sup>*2</sup>	Initialized	
EBR1	Initialized	— <sup>*3</sup>	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	
EBR2	Initialized	— <sup>*3</sup>	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	
SMR_1	Initialized	—	—	—	—	—	—	—	Initialized	SCI_1
ICCR_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
BRR_1	Initialized	—	—	—	—	—	—	—	Initialized	SCI_1
ICSR_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
SCR_1	Initialized	—	—	—	—	—	—	—	Initialized	SCI_1
TDR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICDR_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
SARX_1	Initialized	—	—	—	—	—	—	—	Initialized	
ICMR_1	Initialized	—	—	—	—	—	—	—	Initialized	
SAR_1	Initialized	—	—	—	—	—	—	—	Initialized	
TIER	Initialized	—	—	—	—	—	—	—	Initialized	FRT
TCSR	Initialized	—	—	—	—	—	—	—	Initialized	
FRCH	Initialized	—	—	—	—	—	—	—	Initialized	
FRCL	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
OCRAH	Initialized	—	—	—	—	—	—	—	Initialized	FRT
OCRBH	Initialized	—	—	—	—	—	—	—	Initialized	
OCRAL	Initialized	—	—	—	—	—	—	—	Initialized	
OCRBL	Initialized	—	—	—	—	—	—	—	Initialized	
TCR	Initialized	—	—	—	—	—	—	—	Initialized	
TOCR	Initialized	—	—	—	—	—	—	—	Initialized	
ICRAH	Initialized	—	—	—	—	—	—	—	Initialized	
OCRARH	Initialized	—	—	—	—	—	—	—	Initialized	
ICRAL	Initialized	—	—	—	—	—	—	—	Initialized	
OCRARL	Initialized	—	—	—	—	—	—	—	Initialized	
ICRBH	Initialized	—	—	—	—	—	—	—	Initialized	
OCRAFH	Initialized	—	—	—	—	—	—	—	Initialized	
ICRBL	Initialized	—	—	—	—	—	—	—	Initialized	
OCRAFL	Initialized	—	—	—	—	—	—	—	Initialized	
ICRCH	Initialized	—	—	—	—	—	—	—	Initialized	
OCRDMH	Initialized	—	—	—	—	—	—	—	Initialized	
ICRCL	Initialized	—	—	—	—	—	—	—	Initialized	
OCRDML	Initialized	—	—	—	—	—	—	—	Initialized	
ICRDH	Initialized	—	—	—	—	—	—	—	Initialized	
ICRDL	Initialized	—	—	—	—	—	—	—	Initialized	
SMR_2	Initialized	—	—	—	—	—	—	—	Initialized	SCI_2
DACR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWMX
DADRAH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DADRAL	Initialized	—	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	
BRR_2	Initialized	—	—	—	—	—	—	—	Initialized	SCI_2
SCR_2	Initialized	—	—	—	—	—	—	—	Initialized	
TDR_2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_2	Initialized	—	—	—	—	—	—	—	Initialized	
DADRBH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWMX
DACNTH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DADRBL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DACNTL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
TCSR_0	Initialized	—	—	—	—	—	—	—	Initialized	WDT_0
TCNT_0	Initialized	—	—	—	—	—	—	—	Initialized	



Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
PAODR	Initialized	—	—	—	—	—	—	—	Initialized	PORT
PAPIN	Initialized	—	—	—	—	—	—	—	Initialized	
PADDR	Initialized	—	—	—	—	—	—	—	Initialized	
P1PCR	Initialized	—	—	—	—	—	—	—	Initialized	
P2PCR	Initialized	—	—	—	—	—	—	—	Initialized	
P3PCR	Initialized	—	—	—	—	—	—	—	Initialized	
P1DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P2DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P1DR	Initialized	—	—	—	—	—	—	—	Initialized	
P2DR	Initialized	—	—	—	—	—	—	—	Initialized	
P3DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P4DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P3DR	Initialized	—	—	—	—	—	—	—	Initialized	
P4DR	Initialized	—	—	—	—	—	—	—	Initialized	
P5DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P6DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P5DR	Initialized	—	—	—	—	—	—	—	Initialized	
P6DR	Initialized	—	—	—	—	—	—	—	Initialized	
P8DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P7PIN	Initialized	—	—	—	—	—	—	—	Initialized	
P8DR	Initialized	—	—	—	—	—	—	—	Initialized	
P9DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P9DR	Initialized	—	—	—	—	—	—	—	Initialized	
IER	Initialized	—	—	—	—	—	—	—	Initialized	INT
STCR	Initialized	—	—	—	—	—	—	—	Initialized	SYSTEM
SYSCR	Initialized	—	—	—	—	—	—	—	Initialized	
MDCR	Initialized	—	—	—	—	—	—	—	Initialized	
BCR	Initialized	—	—	—	—	—	—	—	Initialized	BSC
WSCR	Initialized	—	—	—	—	—	—	—	Initialized	
TCR_0	Initialized	—	—	—	—	—	—	—	Initialized	TMR_0, TMR_1
TCR_1	Initialized	—	—	—	—	—	—	—	Initialized	
TCSR_0	Initialized	—	—	—	—	—	—	—	Initialized	
TCSR_1	Initialized	—	—	—	—	—	—	—	Initialized	
TCORA_0	Initialized	—	—	—	—	—	—	—	Initialized	
TCORA_1	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
TCORB_0	Initialized	—	—	—	—	—	—	—	Initialized	TMR_0, TMR_1
TCORB_1	Initialized	—	—	—	—	—	—	—	Initialized	
TCNT_0	Initialized	—	—	—	—	—	—	—	Initialized	
TCNT_1	Initialized	—	—	—	—	—	—	—	Initialized	
PWOERB	Initialized	—	—	—	—	—	—	—	Initialized	PWM
PWOERA	Initialized	—	—	—	—	—	—	—	Initialized	
PWDPRB	Initialized	—	—	—	—	—	—	—	Initialized	
PWDpra	Initialized	—	—	—	—	—	—	—	Initialized	
PWSL	Initialized	—	—	—	—	—	—	—	Initialized	
PWDR0-15	Initialized	—	—	—	—	—	—	—	Initialized	
SMR_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICCR_0	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
BRR_0	Initialized	—	—	—	—	—	—	—	Initialized	SCI_0
ICSR_0	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
SCR_0	Initialized	—	—	—	—	—	—	—	Initialized	SCI_0
TDR_0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICDR_0	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
SARX_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICMR_0	Initialized	—	—	—	—	—	—	—	Initialized	
SAR_0	Initialized	—	—	—	—	—	—	—	Initialized	
ADDRAH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	A/D converter
ADDRAL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADDRBH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADDRBL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADDRCH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADDRCL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADDRDH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADDRDL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADCSR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ADCR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
TCSR_1	Initialized	—	—	—	—	—	—	—	Initialized	WDT_1
TCNT_1	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
KMIMR6	Initialized	—	—	—	—	—	—	—	Initialized	INT
KMPCR6	Initialized	—	—	—	—	—	—	—	Initialized	PORT
KMIMRA	Initialized	—	—	—	—	—	—	—	Initialized	INT
WUEMR3	Initialized	—	—	—	—	—	—	—	Initialized	
TCR_X	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y, TMR_X
TCR_Y	Initialized	—	—	—	—	—	—	—	Initialized	
TCSR_X	Initialized	—	—	—	—	—	—	—	Initialized	
TCSR_Y	Initialized	—	—	—	—	—	—	—	Initialized	
TICRR	Initialized	—	—	—	—	—	—	—	Initialized	
TCORA_Y	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y, TMR_X
TICRF	Initialized	—	—	—	—	—	—	—	Initialized	
TCORB_Y	Initialized	—	—	—	—	—	—	—	Initialized	
TCNT_X	Initialized	—	—	—	—	—	—	—	Initialized	
TCNT_Y	Initialized	—	—	—	—	—	—	—	Initialized	
TCORC	Initialized	—	—	—	—	—	—	—	Initialized	
TISR	Initialized	—	—	—	—	—	—	—	Initialized	
TCORA_X	Initialized	—	—	—	—	—	—	—	Initialized	
TCORB_X	Initialized	—	—	—	—	—	—	—	Initialized	
DADR0	Initialized	—	—	—	—	—	—	—	Initialized	D/A converter
DADR1	Initialized	—	—	—	—	—	—	—	Initialized	
DACR	Initialized	—	—	—	—	—	—	—	Initialized	
TCONRI	Initialized	—	—	—	—	—	—	—	Initialized	Timer connection
TCONRO	Initialized	—	—	—	—	—	—	—	Initialized	
TCONRS	Initialized	—	—	—	—	—	—	—	Initialized	
SEDGR	Initialized	—	—	—	—	—	—	—	Initialized	

- Notes:
1. Initialized by a function software reset.
  2. The FLER bit is not initialized.
  3. Initialized when the SWE bit in FLMCR1 is cleared to 0.



## Section 29 Electrical Characteristics

### 29.1 Absolute Maximum Ratings

Table 29.1 lists the absolute maximum ratings.

**Table 29.1 Absolute Maximum Ratings**

Item	Symbol	Value	Unit
Power supply voltage*	$V_{CC}$	-0.3 to +4.3	V
Power supply voltage (VCL pin)	$V_{CL}$	-0.3 to +4.3	
Input voltage (except ports 6 and 7)	$V_{in}$	-0.3 to $V_{CC} + 0.3$	
Input voltage (CIN input not selected in port 6)	$V_{in}$	-0.3 to $V_{CC} + 0.3$	
Input voltage (CIN input selected in port 6)	$V_{in}$	Lower of -0.3 to $V_{CC} + 0.3$ or -0.3 to $AV_{CC} + 0.3$	
Input voltage (port 7)	$V_{in}$	-0.3 to $AV_{CC} + 0.3$	
Reference power supply voltage	$AV_{ref}$	-0.3 to $AV_{CC} + 0.3$	
Analog power supply voltage/Bus driver power supply voltage	$AV_{CC}/DrV_{CC}$	-0.3 to +4.3	
Analog input voltage	$V_{AN}$	-0.3 to $AV_{CC} + 0.3$	
Operating temperature	$T_{opr}$	-20 to +75	°C
Operating temperature (when flash memory is programmed or erased)	$T_{opr}$	-20 to +75	
Storage temperature	$T_{stg}$	-55 to +125	

Caution: Permanent damage to this LSI may result if absolute maximum ratings are exceeded.

Take care the applied power supply does not exceed 4.3 V.

Note: \* Voltage applied to the VCC pin. Both the VCC pin and VCL pin should be connected to the  $V_{CC}$  power supply.

## 29.2 DC Characteristics

Table 29.2 lists the DC characteristics. Table 29.3 lists the permissible output currents. Table 29.4 lists the I<sup>2</sup>C bus drive characteristics. Table 29.5 lists the USB pin characteristics. Table 29.6 lists the multimedia card interface pin characteristics.

**Table 29.2 DC Characteristics (1)**

Conditions:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}^{*1}$ ,  $AV_{CC}^{*2} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{ref}^{*2} = 2.7\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS}^{*2} = 0\text{ V}$

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	P67 to P60 (KWUL = 00) <sup>*3 *4</sup> , $\overline{IRQ7}$ to $\overline{IRQ0}^{*3}$ , $\overline{IRQ15}$ to $\overline{IRQ8}$ , $\overline{KIN9}$ , $\overline{KIN8}$ , $\overline{WUE15}$ to $\overline{WUE8}$	$V_T^-$	$V_{CC} \times 0.2$	—	—	V
		$V_T^+$	—	—	$V_{CC} \times 0.7$	
		$V_T^- - V_T^+$	$V_{CC} \times 0.05$	—	—	
Schmitt trigger input voltage (at level switch) <sup>*4</sup>	P67 to P60 (KWUL = 01)	$V_T^-$	$V_{CC} \times 0.3$	—	—	
		$V_T^+$	—	—	$V_{CC} \times 0.7$	
		$V_T^+ - V_T^-$	$V_{CC} \times 0.05$	—	—	
	P67 to P60 (KWUL = 10)	$V_T^-$	$V_{CC} \times 0.4$	—	—	
		$V_T^+$	—	—	$V_{CC} \times 0.8$	
		$V_T^+ - V_T^-$	$V_{CC} \times 0.03$	—	—	
P67 to P60 (KWUL = 11)	$V_T^-$	$V_{CC} \times 0.45$	—	—		
	$V_T^+$	—	—	$V_{CC} \times 0.9$		
	$V_T^+ - V_T^-$	0.05	—	—		
Input high voltage	$\overline{RES}$ , $\overline{STBY}$ , $\overline{NMI}$ , $\overline{MD1}$ , $\overline{MD0}$ , $\overline{MD2}$ , $\overline{FWE}$	$V_{IH}$	$V_{CC} \times 0.9$	—	$V_{CC} + 0.3$	(2)
	Port 7	$V_{CC} \times 0.7$	—	$AV_{CC} + 0.3$		
	Input pins other than (1) and (2) above, and input pins other than applicable pins when IIC, USB, and MCIF are used	$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		

Item	Symbol	Min	Typ	Max	Unit	Test Conditions	
Input low voltage	$\overline{\text{RES}}$ , $\overline{\text{STBY}}$ , MD1, MD0, $\overline{\text{MD2}}$ , FWE (3)	$V_{\text{IL}}$	-0.3	—	$V_{\text{CC}} \times 0.1$	V	
	NMI, EXTAL, input pins other than (1) and (3) above, and input pins other than applicable pins when IIC, USB, and MCIF are used		-0.3	—	$V_{\text{CC}} \times 0.2$		
Output high voltage	All output pins (except for port 8 and applicable output pins when USB and MCIF are used) <sup>*5,*6</sup>	$V_{\text{OH}}$	$V_{\text{CC}} - 0.5$	—	—	$I_{\text{OH}} = -200 \mu\text{A}$	
			$V_{\text{CC}} - 1.0$	—	—	$I_{\text{OH}} = -1 \text{ mA}$	
	Port 8 <sup>*5</sup>		0.5	—	—	$I_{\text{OH}} = -200 \mu\text{A}$	
Output low voltage	All output pins (except for $\overline{\text{RESO}}$ and applicable output pins when IIC, USB, and MCIF are used) <sup>*6</sup>	$V_{\text{OL}}$	—	—	0.4	$I_{\text{OL}} = 1.6 \text{ mA}$	
			Ports 1 to 3	—	—	1.0	$I_{\text{OL}} = 5 \text{ mA}$
			$\overline{\text{RESO}}$	—	—	0.4	$I_{\text{OL}} = 1.6 \text{ mA}$

**Table 29.2 DC Characteristics (2)**

Conditions:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}^{*1}$ ,  $AV_{CC}^{*2} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{ref}^{*2} = 2.7\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS}^{*2} = 0\text{ V}$

Item	Symbol	Min	Typ	Max	Unit	Test Conditions		
Input leakage current	$\overline{RES}$	$ I_{in} $	—	—	10.0	$\mu\text{A}$ , $V_{in} = 0.5\text{ to }V_{CC} - 0.5\text{ V}$		
	$\overline{STBY}$ , NMI, MD1, MD0, $\overline{MD2}$ , FWE		—	—	1.0			
	Port 7		—	—	1.0	$V_{in} = 0.5\text{ to }AV_{CC} - 0.5\text{ V}$		
Three-state leakage current (off state)	Ports 1 to 6, 8, 9, A	$ I_{TSL} $	—	—	1.0	$V_{in} = 0.5\text{ to }V_{CC} - 0.5\text{ V}$		
Input pull-up MOS current	Ports 1 to 3	$-I_P$	5	—	150	$V_{in} = 0\text{ V}$		
	Ports 6 (P6PUE = 0), A		30	—	300			
	Port 6 (P6PUE=1)		3	—	100			
Input capacitance	$\overline{RES}$ (4)	$C_{IN}$	—	—	80	$\text{pF}$ , $V_{in} = 0\text{ V}$		
	NMI		—	—	50	$f = 1\text{ MHz}$		
	P80 to P83		—	—	20	$T_a = 25\text{ }^\circ\text{C}$		
	Input pins other than above (4)		—	—	15			
Current consumption <sup>*7</sup>	Normal operation	$I_{CC}$	—	105	135	$\text{mA}$	$V_{CC} = 3.0\text{ V to }3.6\text{ V}$ $f = 25\text{ MHz}$ , All modules operating, high-speed mode	
			—	80	115		$V_{CC} = 2.7\text{ V to }3.6\text{ V}$ $f = 20\text{ MHz}$ , All modules operating, high-speed mode	
			—	50	75		$V_{CC} = 2.7\text{ V to }3.6\text{ V}$ $f = 12\text{ MHz}$ , All modules operating, high-speed mode	
			—	75	100		$V_{CC} = 3.0\text{ V to }3.6\text{ V}$ $f = 25\text{ MHz}$	
	Sleep mode			—	60	85		$V_{CC} = 2.7\text{ V to }3.6\text{ V}$ $f = 20\text{ MHz}$
				—	35	60		$V_{CC} = 2.7\text{ V to }3.6\text{ V}$ $f = 12\text{ MHz}$
				—	0.1	5.0	$\mu\text{A}$	$T_a \leq 50\text{ }^\circ\text{C}$
				—	—	20.0		$50\text{ }^\circ\text{C} < T_a$
Standby mode <sup>*8</sup>								



**Table 29.2 DC Characteristics (3)**

Conditions:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}^{*1}$ ,  $AV_{CC}^{*2} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{ref}^{*2} = 2.7\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS}^{*2} = 0\text{ V}$

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Analog power supply current	During A/D or D/A conversion	$AI_{CC}$	—	1.2	2.0	mA	
	Waiting for A/D or D/A conversion		—	0.01	5.0	$\mu\text{A}$	$AV_{CC} = 2.0\text{ V to }3.6\text{ V}^{*2}$
Reference power supply current	During A/D conversion	$AI_{ref}$	—	0.5	1.0	mA	
	During A/D or D/A conversion		—	2.0	5.0		
	Waiting for A/D or D/A conversion		—	0.01	5.0	$\mu\text{A}$	$AV_{ref} = 2.0\text{ V to }AV_{CC}^{*2}$
Analog power supply voltage <sup>*1</sup>		$AV_{CC}$	2.7	—	3.6	V	Operating
			2.0	—	3.6		Idle/not used
RAM standby voltage		$V_{RAM}$	2.0	—	—		

- Notes:
- The flash memory must be programmed or erased within the conditions  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ .
  - Do not leave the AVCC, AVref, and AVSS pins open even if the A/D converter or D/A converter is not used.  
 Even if the A/D converter or D/A converter is not used, apply a value in the range from 2.0 V to 3.6 V to the AVCC and AVref pins by connection to the power supply ( $V_{CC}$ ). The relationship between these two pins should be  $AV_{ref} \leq AV_{CC}$ .
  - Includes peripheral module inputs multiplexed on the pin.
  - Maximum voltage applied to port 6 is  $V_{CC} + 0.3\text{ V}$  when the CIN input is not selected, or the lower of  $V_{CC} + 0.3\text{ V}$  or  $AV_{CC} + 0.3\text{ V}$  when the CIN input is selected. If port 6 is in output mode, the output voltage is applied to port 6.
  - P80/ $\overline{\text{EX}}\text{IRQ8}/\text{SCL0}$ , P81/ $\overline{\text{EX}}\text{IRQ9}/\text{SDA0}$ , P82/ $\overline{\text{EX}}\text{IRQ10}/\text{SCL1}$ , and P83/ $\overline{\text{EX}}\text{IRQ11}/\text{SDA1}$  are NMOS push-pull outputs.  
 An external pull-up resistor is necessary to provide high-level output from SCL0, SCL1, SDA0, and SDA1 (ICE in ICCR is 1).  
 P80 to P83, P87, P84/SCK0, P85/SCK1, and P86/SCK2 (ICE in ICCR is 0) high levels are driven by NMOS. An external pull-up resistor is necessary to provide high-level output from these pins when they are used as an output.
  - Indicates values when ICE in ICCR is 0. Low level output when the bus drive function is selected is rated separately.
  - Current consumption values are for  $V_{IH\text{ min}} = V_{CC} - 0.2\text{ V}$  and  $V_{IL\text{ max}} = 0.2\text{ V}$  with all output pins unloaded and the on-chip pull-up MOSs in the off state.
  - The values are for  $V_{RAM} \leq V_{CC} < 2.7\text{ V}$ ,  $V_{IH\text{ min}} = V_{CC} - 0.2\text{ V}$ , and  $V_{IL\text{ max}} = 0.2\text{ V}$ .

**Table 29.3 Permissible Output Currents**Conditions:  $V_{CC} = 2.7\text{ V}$  to  $3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ 

Item		Symbol	Min	Typ	Max	Unit
Permissible output low current (per pin)	SCL1, SCL0, SDA1, SDA0	$I_{OL}$	—	—	10	mA
	Ports 1 to 3		—	—	2	
	$\overline{\text{RESO}}$		—	—	1	
	Other output pins		—	—	1	
Permissible output low current (total)	Total of ports 1 to 3	$\Sigma I_{OL}$	—	—	40	
	Total of all output pins, including the above		—	—	60	
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	30	

Notes: 1. To protect LSI reliability, do not exceed the output current values in table 29.3.

2. When driving a Darlington transistor or LED, always insert a current-limiting resistor in the output line, as show in figures 29.1 and 29.2.

**Table 29.4 I<sup>2</sup>C Bus Drive Characteristics**Conditions:  $V_{CC} = 2.7\text{ V to }3.6\text{V}$ ,  $V_{SS} = 0\text{ V}$ 

Applicable Pins: SCL1 and SCL0, SDA1 and SDA0 (bus drive function selected)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	$V_T^-$	$V_{CC} \times 0.3$	—	—	V	
	$V_T^+$	—	—	$V_{CC} \times 0.7$		
	$V_T^+ - V_T^-$	$V_{CC} \times 0.05$	—	—		
Input high voltage	$V_{IH}$	$V_{CC} \times 0.7$	—	$V_{CC} + 0.5$		
Input low voltage	$V_{IL}$	$-0.5$	—	$V_{CC} \times 0.3$		
Output low voltage	$V_{OL}$	—	—	0.5		$I_{OL} = 8\text{ mA}$
		—	—	0.4		$I_{OL} = 3\text{ mA}$
Input capacitance	$C_{in}$	—	—	20	pF	$V_{in} = 0\text{ V}$ , $f = 1\text{ MHz}$ , $T_a = 25^\circ\text{C}$
Three-state leakage current (off state)	$ I_{TSI} $	—	—	1.0	$\mu\text{A}$	$V_{in} = 0.5\text{ to }V_{CC} - 0.5\text{ V}$

**Table 29.5 USB Pin Characteristics**Conditions:  $V_{CC} = 3.3 \text{ V} \pm 0.3 \text{ V}$ ,  $\text{Dr}V_{CC} = 3.3 \text{ V} \pm 0.3 \text{ V}$ ,  $\text{Dr}V_{SS} = V_{SS} = 0 \text{ V}$ 

Applicable Pins: Driver/receiver input/output (USDP, USDM), USEXCL

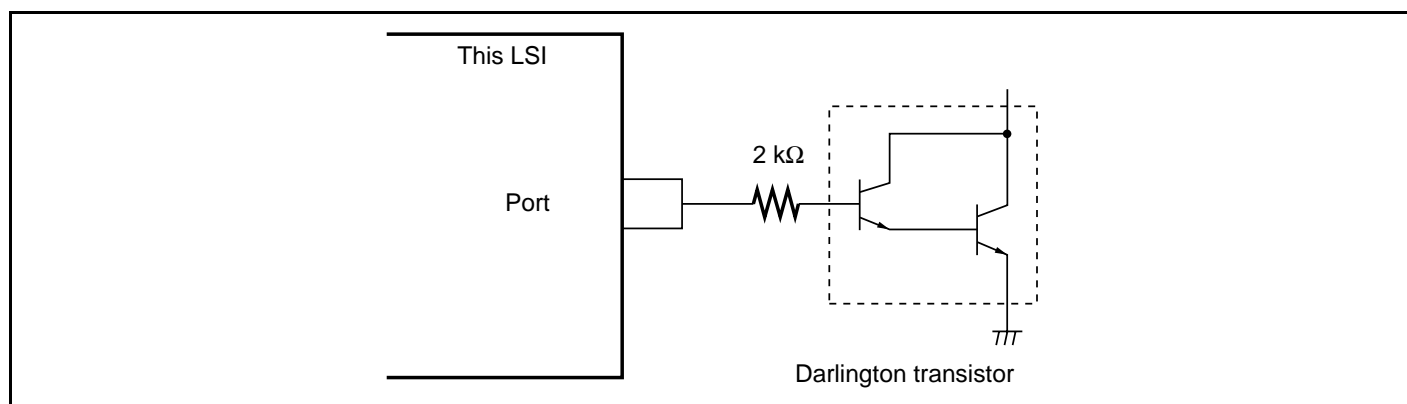
Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Differential input sensitivity		$V_{DI}$	0.2	—	—	V	$  (D+) - (D-)  $
Differential common mode range		$V_{CM}$	0.8	—	2.5		Including $V_{DI}$
Input high voltage	USEXCL	$V_{IH}$	$V_{CC} \times 0.7$	—	—		
	Driver/receiver		2.0	—	$\text{Dr}V_{CC} + 0.3$		
Input low voltage	USEXCL	$V_{IL}$	-0.3	—	$V_{CC} \times 0.2$		
	Driver/receiver		-0.3	—	0.8		
Output high voltage	Driver/receiver	$V_{OH}$	2.8	—	3.6		RL = 15 k $\Omega$ is connected between the pin and GND
Output low voltage	Driver/receiver	$V_{OL}$	0.0	—	0.3		RL = 1.5 k $\Omega$ is connected between the pin and power supply
Output resistor		$Z_{DRV}$	28	—	44	$\Omega$	
I/O pin capacitance		$C_{IN}$	—	—	15	pF	Between the pin and GND
Three-state leakage current		$I_{LO}$	—	—	1.0	$\mu\text{A}$	$0.5 \text{ V} < V_{in} < \text{Dr}V_{CC}$ -0.5 V
Dr $V_{CC}$ current consumption	Normal operation	$\text{DI}_{CC}$	—	5	10	mA	
	Standby mode		—	0.2	5.0	$\mu\text{A}$	

**Table 29.6 Multimedia Card Interface Pin Characteristics**

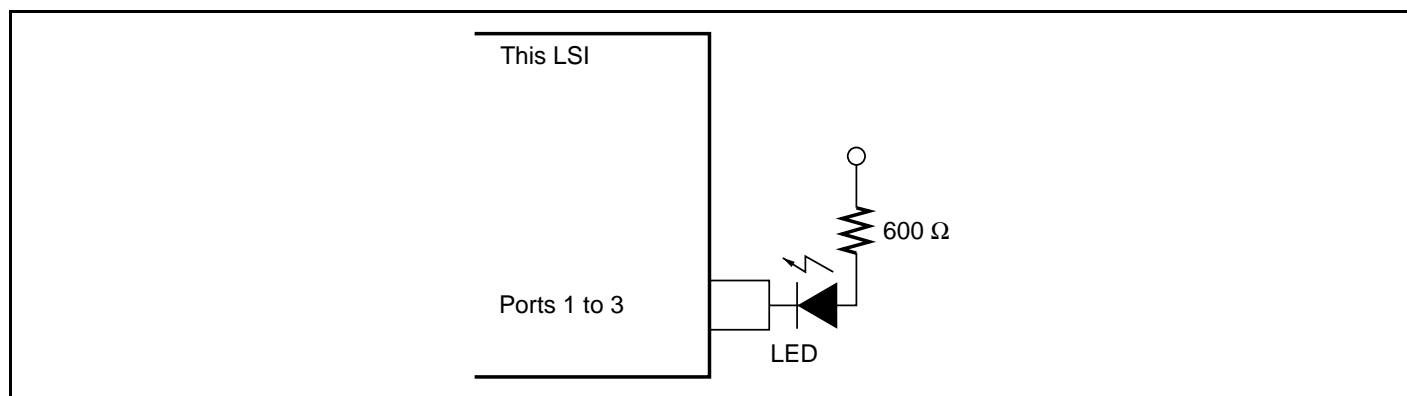
Conditions:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$

Applicable Pins: MCCLK, MCCSA, MCCSB, MCCMD, MCDAT, MCTxD, MCRxD, MCCMDDIR, MCDATDIR, ExMCCLK, ExMCCSA, ExMCCSB, ExMCCMD, ExMCDAT, ExMCTxD, ExMCRxD, ExMCCMDDIR, ExMCDATDIR

Item	Symbol	Min	Typ	Max	Unit
Input high voltage	$V_{IH}$	$V_{CC} \times 0.625$	—	—	V
Input low voltage	$V_{IL}$	—	—	$V_{CC} \times 0.25$	
Output high voltage	$V_{OH}$	$V_{CC} \times 0.75$	—	—	
Output low voltage	$V_{OL}$	—	—	$V_{CC} \times 0.125$	



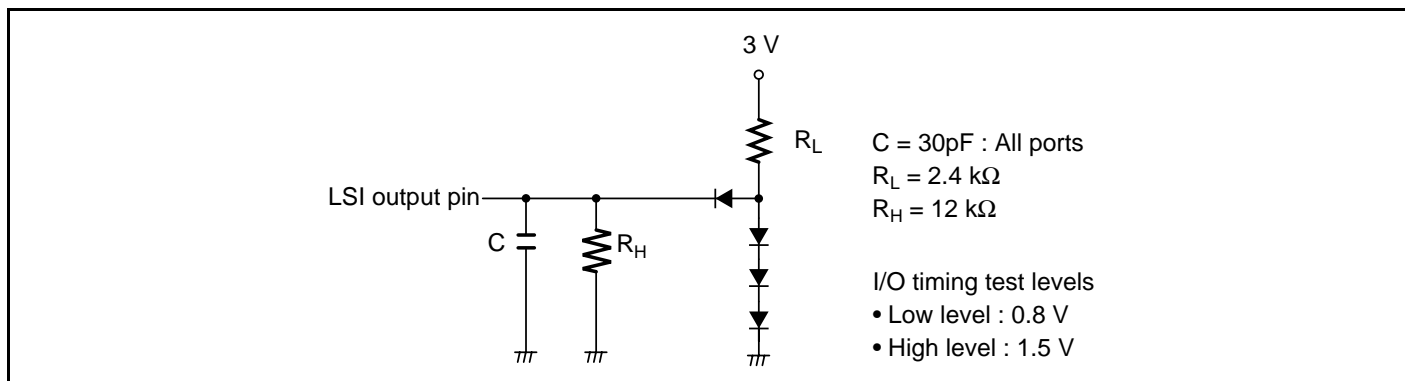
**Figure 29.1 Darlington Transistor Drive Circuit (Example)**



**Figure 29.2 LED Drive Circuit (Example)**

## 29.3 AC Characteristics

Figure 29.3 shows the test conditions for the AC characteristics.



**Figure 29.3 Output Load Circuit**

### 29.3.1 Clock Timing

Table 29.7 shows the clock timing. The clock timing specified here covers clock output ( $\phi$ ), clock pulse generator (crystal) and external clock input (EXTAL pin) oscillation stabilization times. For details of external clock input (EXTAL pin and EXCL pin) timing, see section 26, Clock Pulse Generator.

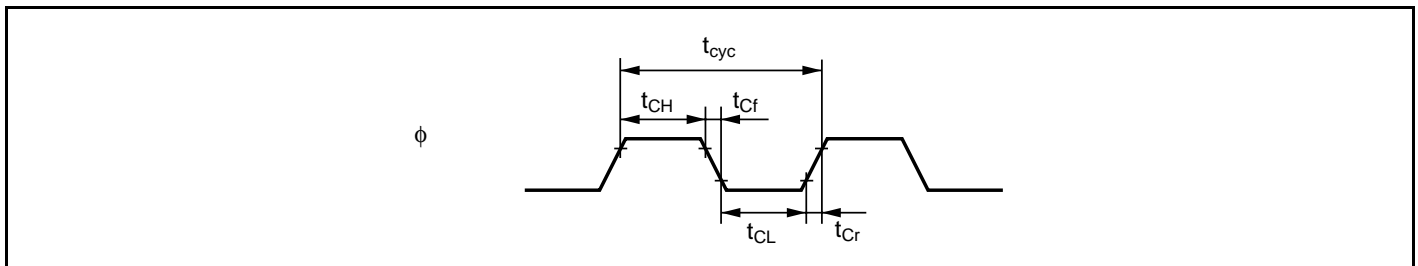
**Table 29.7 Clock Timing**

Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }25\text{ MHz}$

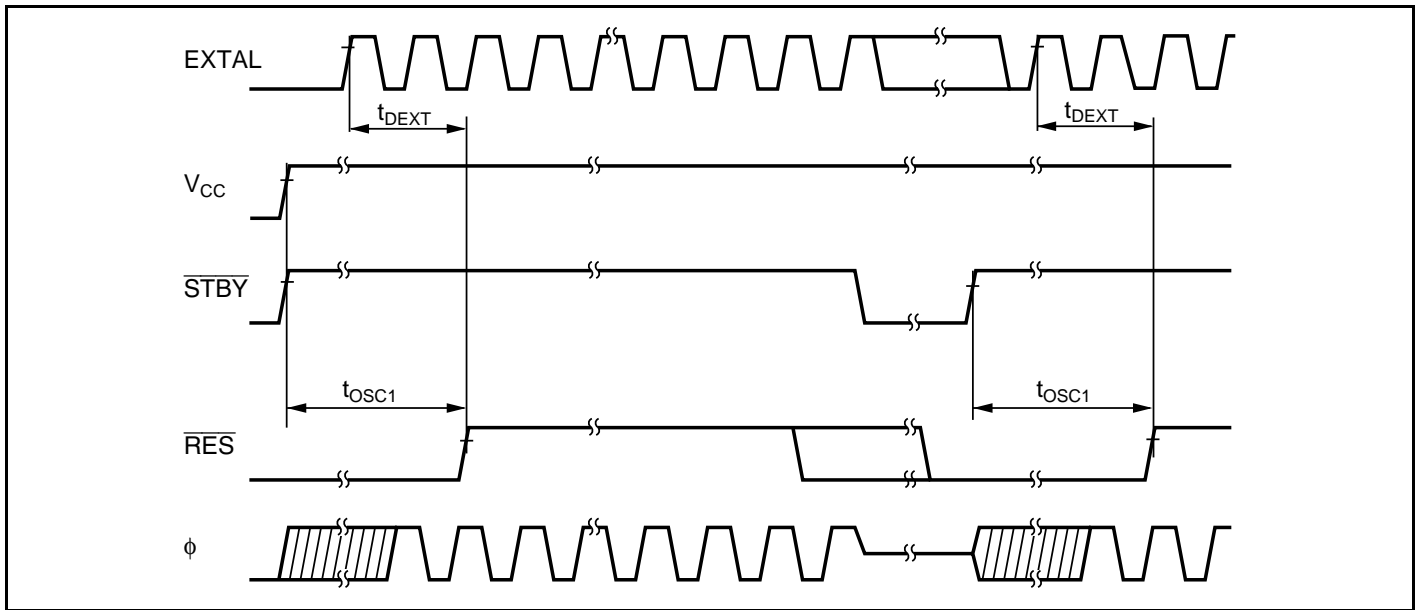
Condition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$

Item	Symbol	Condition A		Condition B		Unit	Reference
		Min	Max	Min	Max		
Clock cycle time	$t_{cyc}$	40	200	50	200	ns	Figure 29.4
Clock high pulse width	$t_{CH}$	15	—	20	—		
Clock low pulse width	$t_{CL}$	15	—	20	—		
Clock rise time	$t_{Cr}$	—	5	—	5		
Clock fall time	$t_{Cf}$	—	5	—	5		
Reset oscillation stabilization (crystal)	$t_{OSC1}$	10	—	10	—	ms	Figure 29.5
Software standby oscillation stabilization time (crystal)	$t_{OSC2}$	8	—	8	—		Figure 29.6
External clock output stabilization delay time	$t_{DEXT}$	500	—	500	—	$\mu\text{s}$	Figure 29.5

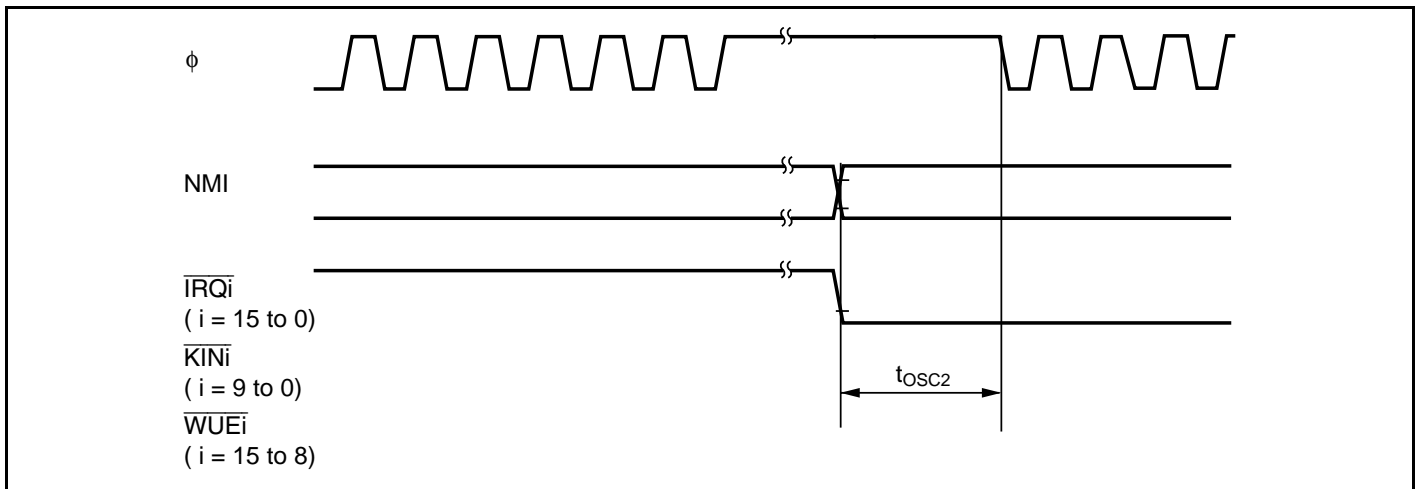
The clock timing is shown below.



**Figure 29.4 System Clock Timing**



**Figure 29.5 Oscillation Stabilization Timing**



**Figure 29.6 Oscillation Stabilization Timing (Exiting Software Standby Mode)**



### 29.3.2 Control Signal Timing

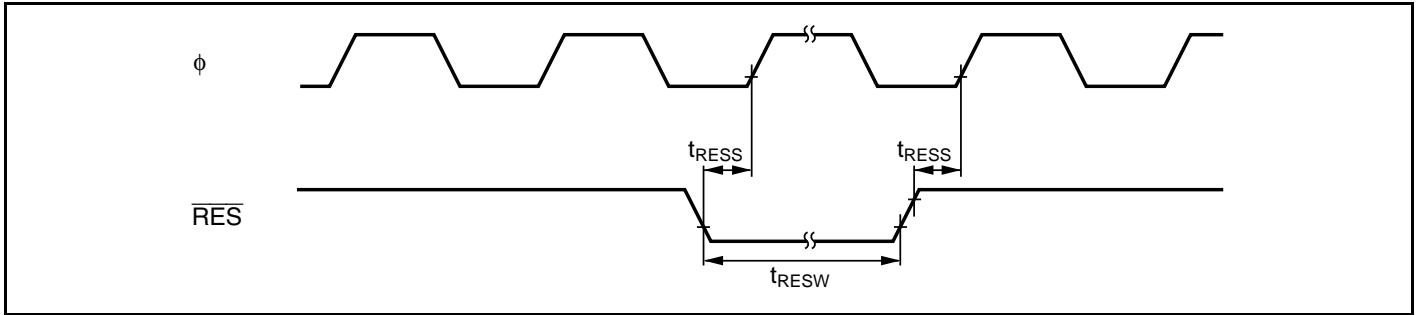
Table 29.8 shows the control signal timing. Only external interrupts NMI, IRQ0 to IRQ15, KIN0 to KIN9, and WUE8 to WUE15 can be operated based on the subclock ( $\phi = 32.768$  kHz).

**Table 29.8 Control Signal Timing**

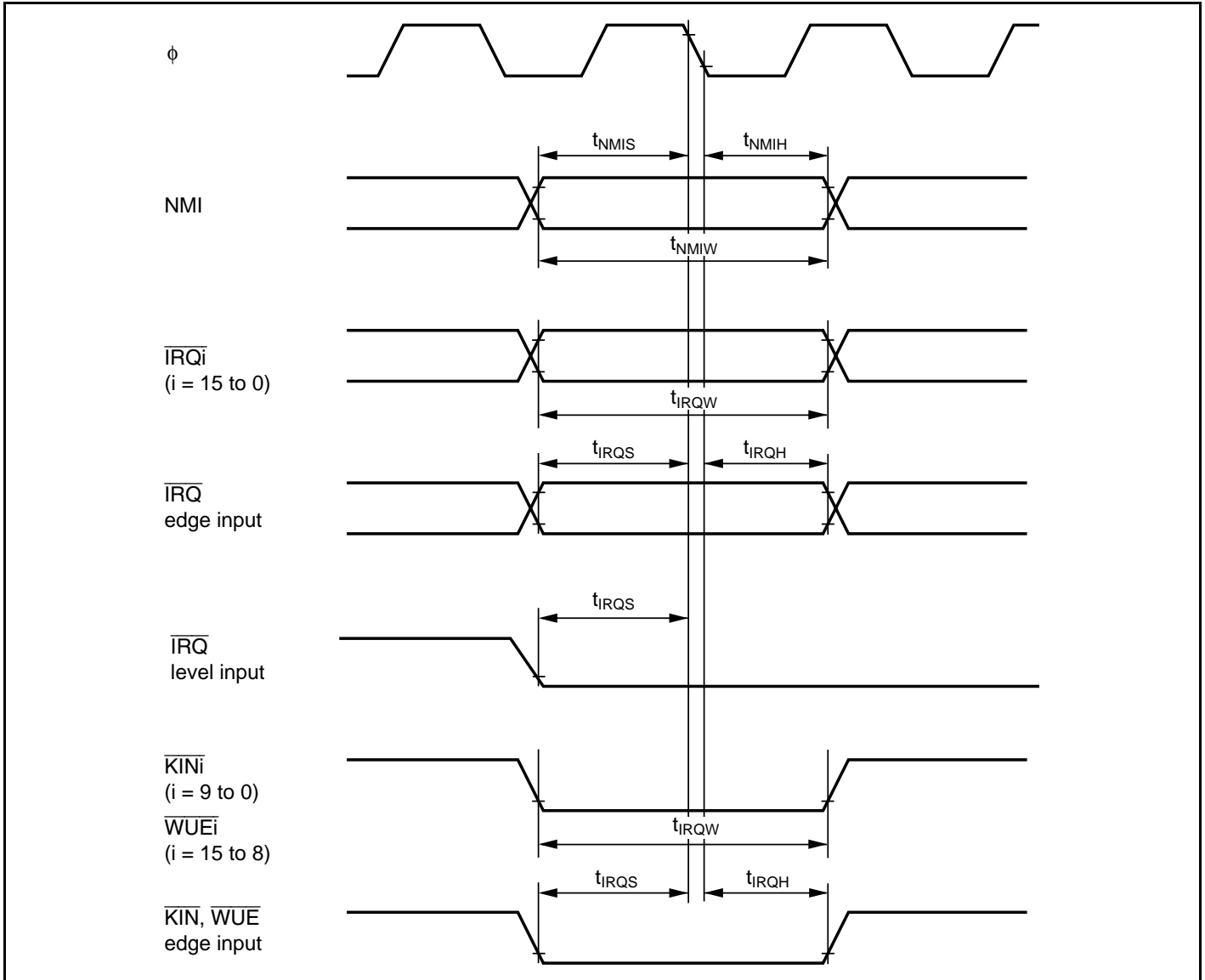
Condition A:  $V_{CC} = 3.0$  V to 3.6 V,  $V_{SS} = 0$  V,  $\phi = 5$  MHz to 25 MHz

Condition B:  $V_{CC} = 2.7$  V to 3.6 V,  $V_{SS} = 0$  V,  $\phi = 5$  MHz to 20 MHz

Item	Symbol	Condition A		Condition B		Unit	Test Conditions
		Min	Max	Min	Max		
$\overline{RES}$ setup time	$t_{RESS}$	200	—	200	—	ns	Figure 29.7
$\overline{RES}$ pulse width	$t_{RESW}$	20	—	20	—	$t_{cyc}$	
NMI setup time	$t_{NMIS}$	150	—	150	—	ns	Figure 29.8
NMI hold time	$t_{NMIH}$	10	—	10	—		
NMI pulse width (exiting software standby mode)	$t_{NMIW}$	200	—	200	—		
IRQ setup time ( $\overline{IRQ15}$ to $\overline{IRQ0}$ , $\overline{KIN9}$ to $\overline{KIN0}$ , $\overline{WUE15}$ to $\overline{WUE8}$ )	$t_{IRQS}$	150	—	150	—		
IRQ hold time ( $\overline{IRQ15}$ to $\overline{IRQ0}$ , $\overline{KIN9}$ to $\overline{KIN0}$ , $\overline{WUE15}$ to $\overline{WUE8}$ )	$t_{IRQH}$	10	—	10	—		
IRQ pulse width ( $\overline{IRQ15}$ to $\overline{IRQ0}$ , $\overline{KIN9}$ to $\overline{KIN0}$ , $\overline{WUE15}$ to $\overline{WUE8}$ ) (exiting software standby mode)	$t_{IRQW}$	200	—	200	—		



**Figure 29.7 Reset Input Timing**



**Figure 29.8 Interrupt Input Timing**

### 29.3.3 Bus Timing

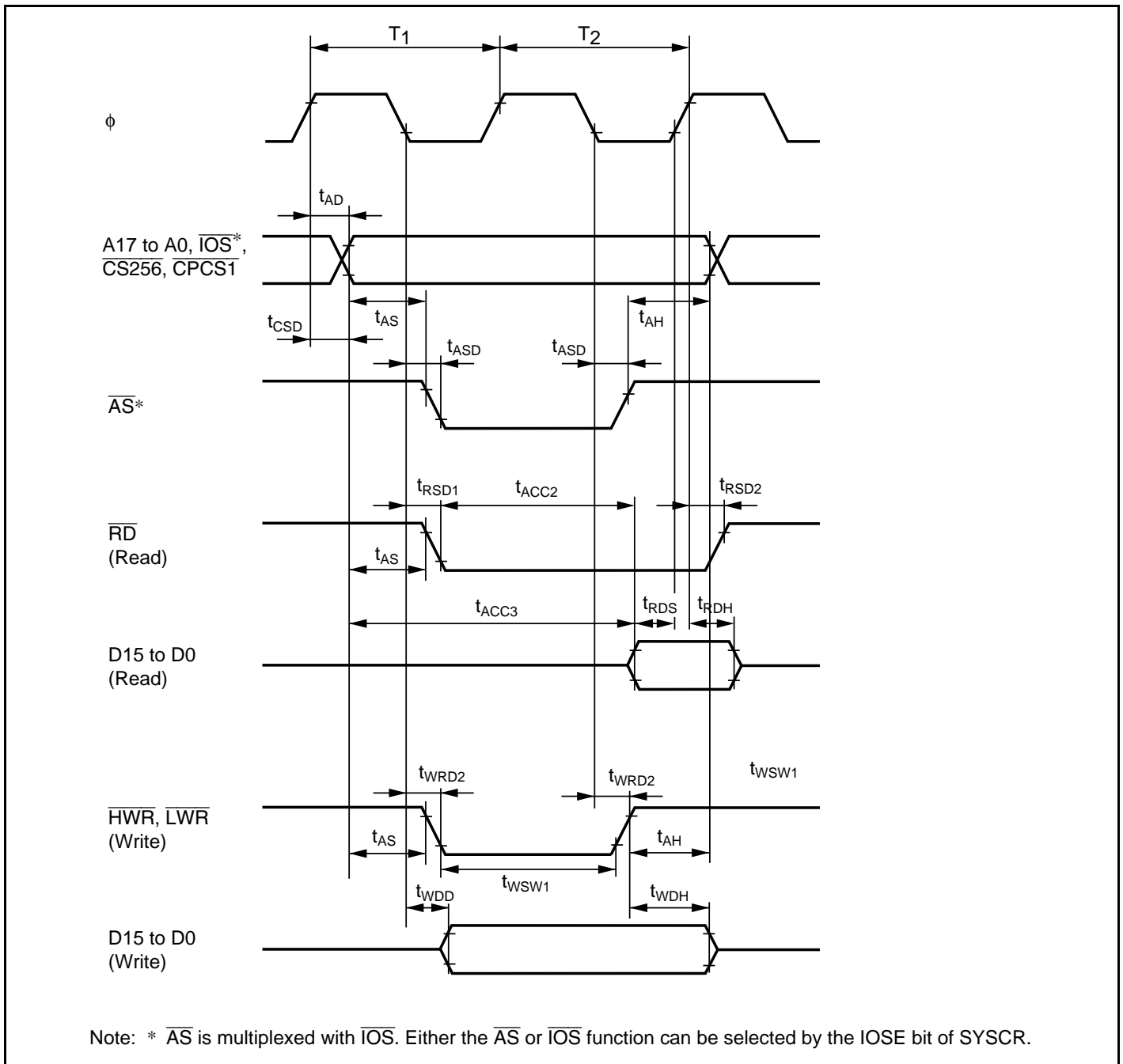
Table 29.9 shows the bus timing. In subclock ( $\phi = 32.768$  kHz) operation, external expansion mode operation cannot be guaranteed.

**Table 29.9 Bus Timing (1) (Normal Mode and Advanced Mode)**

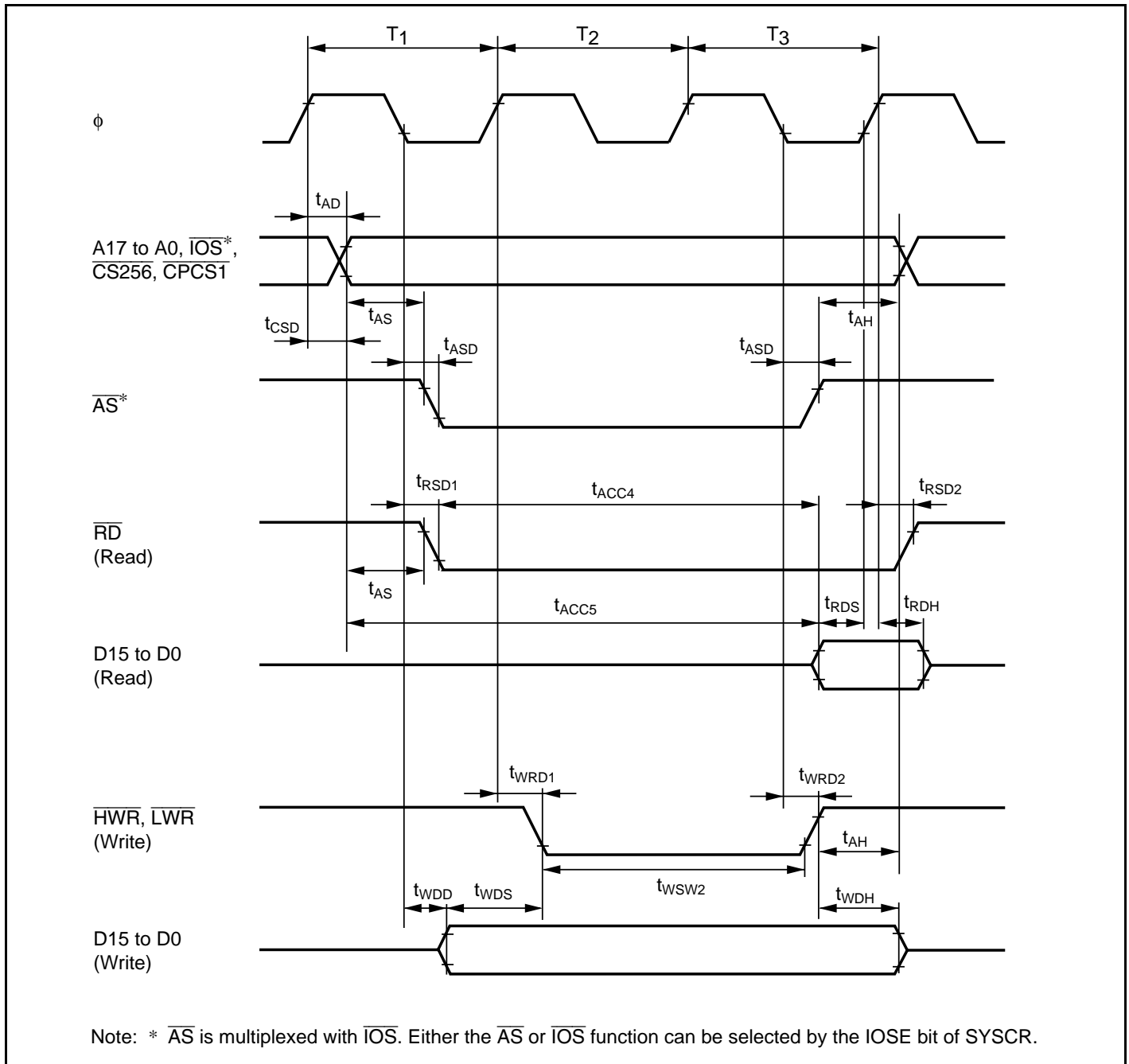
Condition A:  $V_{CC} = 3.0$  V to 3.6 V,  $V_{SS} = 0$  V,  $\phi = 5$  MHz to 25 MHz

Condition B:  $V_{CC} = 2.7$  V to 3.6 V,  $V_{SS} = 0$  V,  $\phi = 5$  MHz to 20 MHz

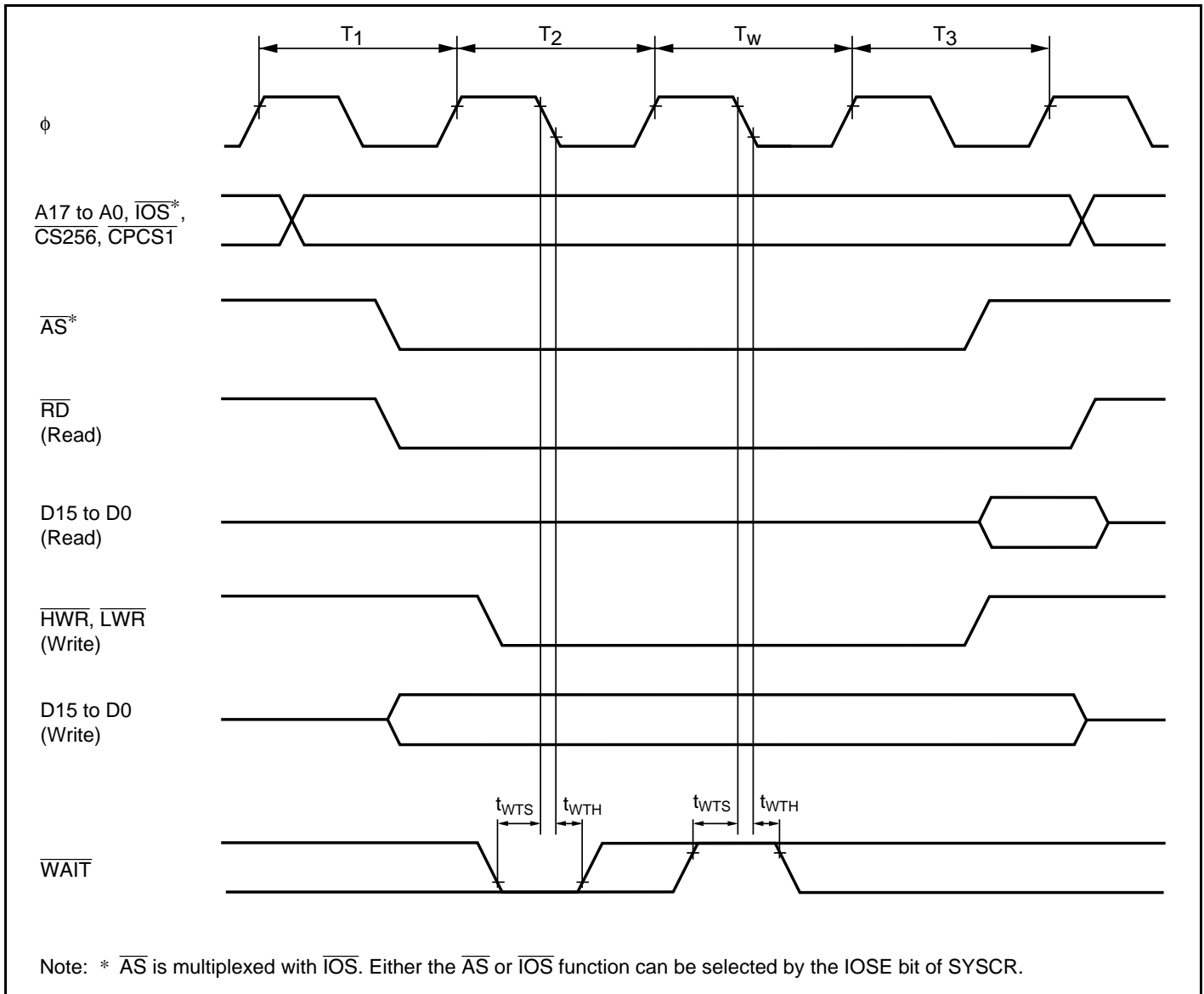
Item	Symbol	Condition A		Condition B		Unit	Test Conditions
		Min	Max	Min	Max		
Address delay time	$t_{AD}$	—	20	—	25	ns	Figures 29.9 to 29.14
Address setup time	$t_{AS}$	$0.5 \times t_{cyc} - 15$	—	$0.5 \times t_{cyc} - 15$	—		
Address hold time	$t_{AH}$	$0.5 \times t_{cyc} - 10$	—	$0.5 \times t_{cyc} - 10$	—		
$\overline{CS}$ delay time (IOS, CS256, CPCS1, CPCS2)	$t_{CSD}$	—	15	—	15		
$\overline{AS}$ delay time	$t_{ASD}$	—	15	—	15		
$\overline{RD}$ delay time 1	$t_{RSD1}$	—	15	—	15		
$\overline{RD}$ delay time 2	$t_{RSD2}$	—	15	—	15		
Read data setup time	$t_{RDS}$	15	—	15	—		
Read data hold time	$t_{RDH}$	0	—	0	—		
Read data access time 1	$t_{ACC1}$	—	$1.0 \times t_{cyc} - 30$	—	$1.0 \times t_{cyc} - 35$		
Read data access time 2	$t_{ACC2}$	—	$1.5 \times t_{cyc} - 25$	—	$1.5 \times t_{cyc} - 30$		
Read data access time 3	$t_{ACC3}$	—	$2.0 \times t_{cyc} - 30$	—	$2.0 \times t_{cyc} - 35$		
Read data access time 4	$t_{ACC4}$	—	$2.5 \times t_{cyc} - 25$	—	$2.5 \times t_{cyc} - 30$		
Read data access time 5	$t_{ACC5}$	—	$3.0 \times t_{cyc} - 30$	—	$3.0 \times t_{cyc} - 35$		
$\overline{WR}$ delay time 1	$t_{WRD1}$	—	15	—	15		
$\overline{WR}$ delay time 2	$t_{WRD2}$	—	15	—	15		
$\overline{WR}$ pulse width 1	$t_{WSW1}$	$1.0 \times t_{cyc} - 20$	—	$1.0 \times t_{cyc} - 20$	—		
$\overline{WR}$ pulse width 2	$t_{WSW2}$	$1.5 \times t_{cyc} - 20$	—	$1.5 \times t_{cyc} - 20$	—		
Write data delay time	$t_{WDD}$	—	30	—	35		
Write data setup time	$t_{WDS}$	0	—	0	—		
Write data hold time	$t_{WDH}$	10	—	10	—		
$\overline{WAIT}$ setup time	$t_{WTS}$	25	—	25	—		
$\overline{WAIT}$ hold time	$t_{WTH}$	5	—	5	—		



**Figure 29.9 Basic Bus Timing/2-State Access**



**Figure 29.10 Basic Bus Timing/3-State Access**



**Figure 29.11 Basic Bus Timing/3-State Access with One Wait State**

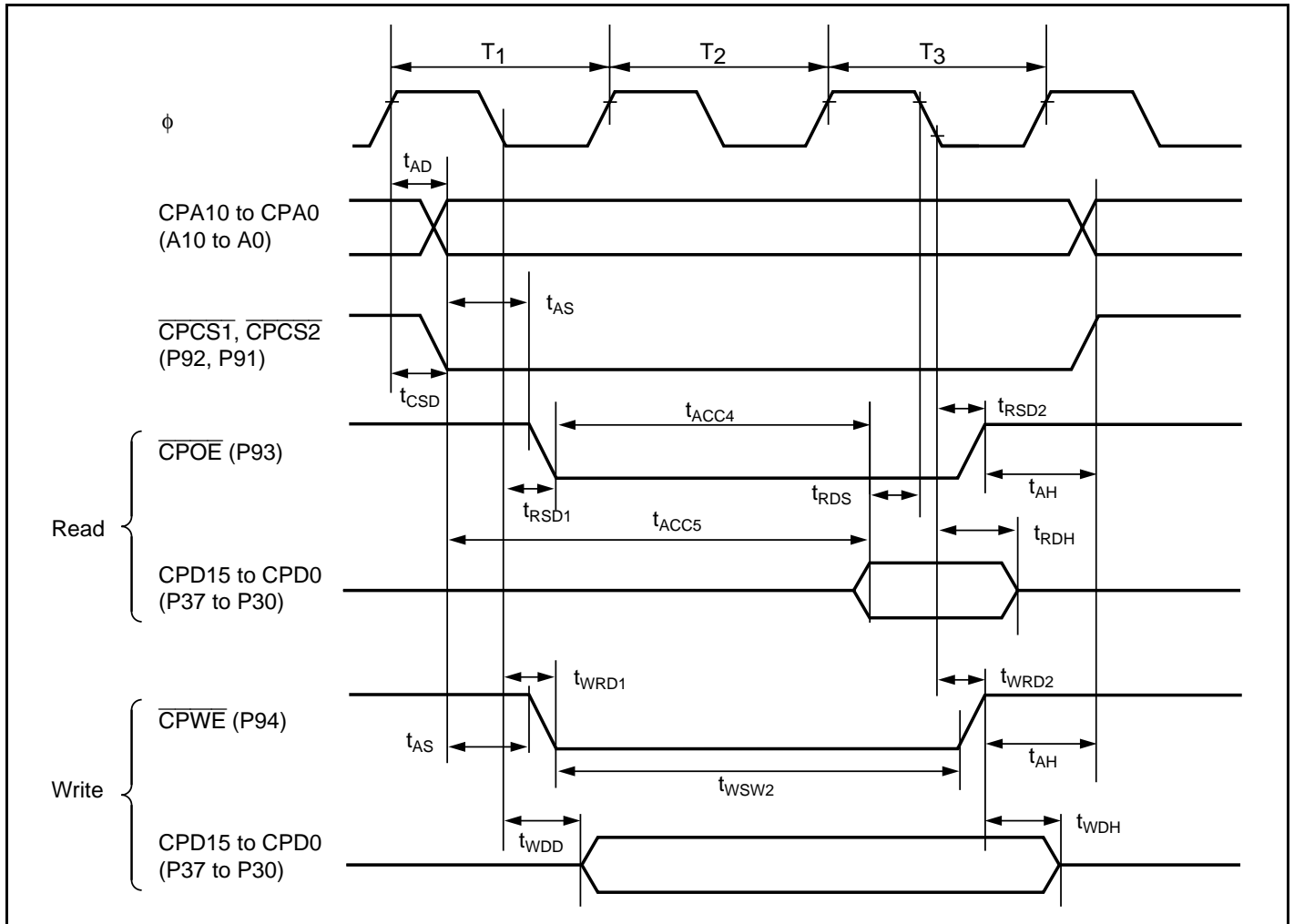
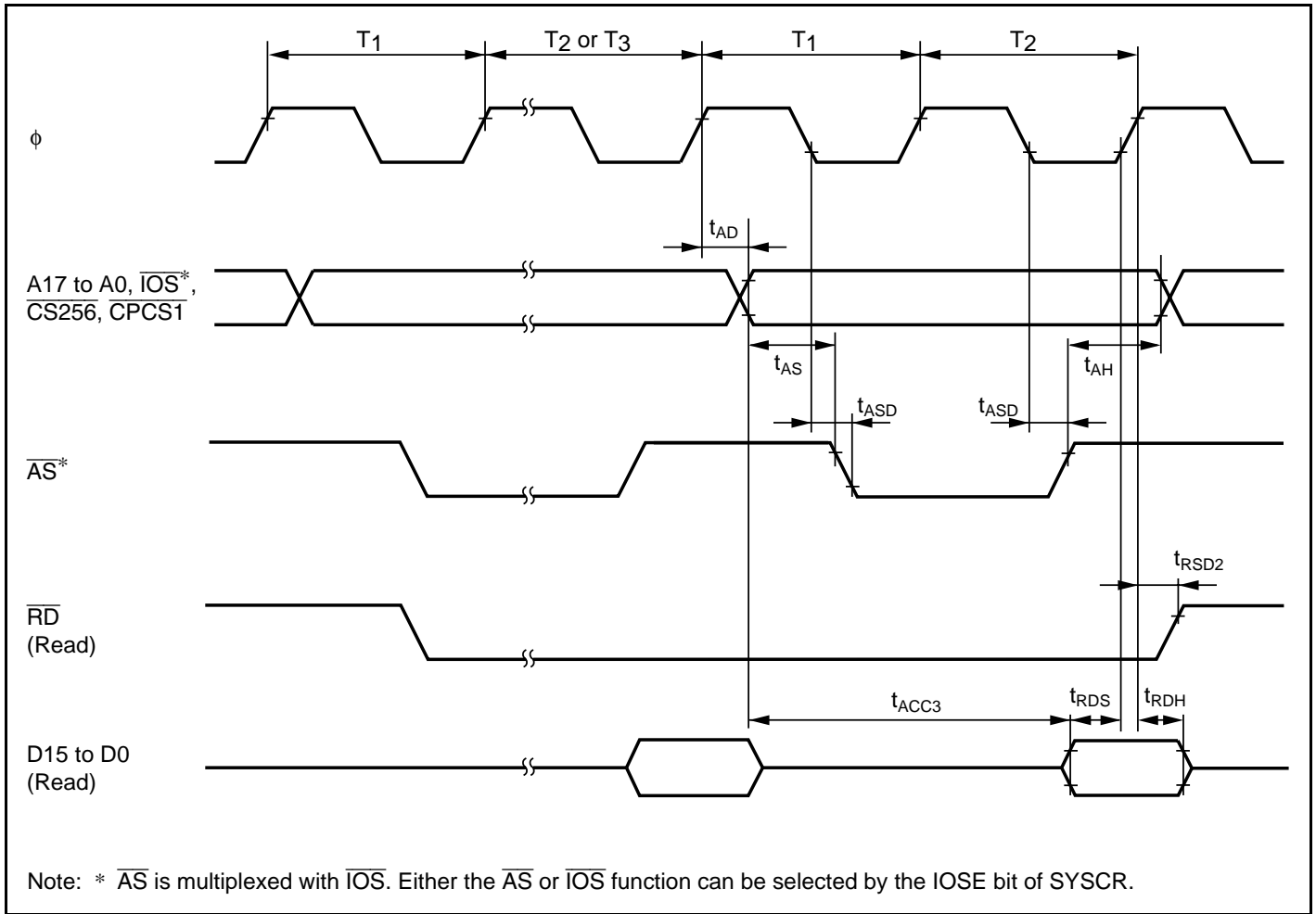
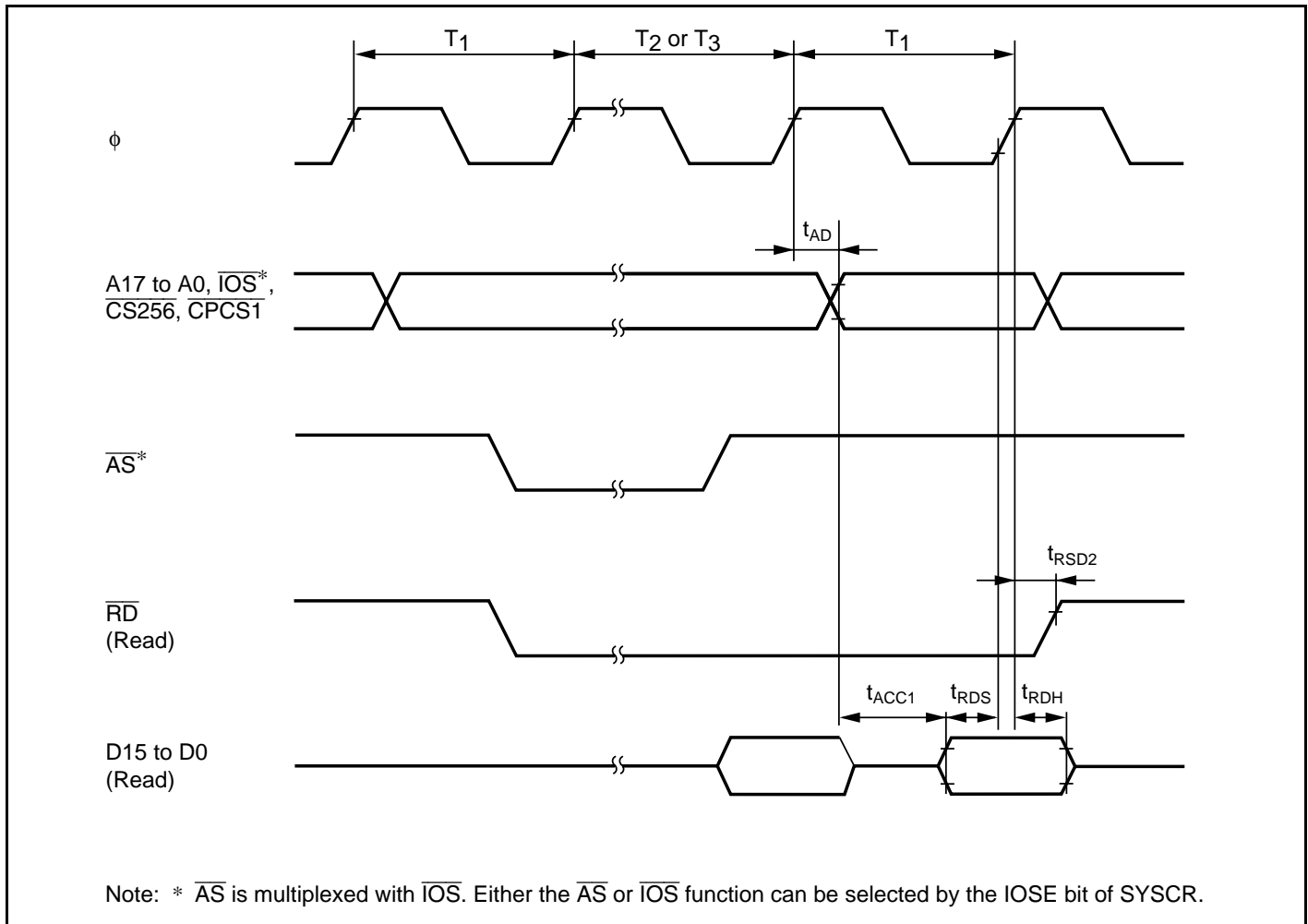


Figure 29.12 CF Interface Basic Timing/3-State Access



**Figure 29.13 Burst ROM Access Timing/2-State Access**





**Figure 29.14 Burst ROM Access Timing/1-State Access**

### 29.3.4 Timing of On-Chip Peripheral Modules

Tables 29.10 to 29.14 show the on-chip peripheral module timing. The on-chip peripheral modules that can be operated by the subclock ( $\phi = 32.768$  kHz) are I/O ports, external interrupts (NMI, IRQ15 to IRQ0, KIN9 to KIN0, and WUE15 to WUE8), watchdog timer, and 8-bit timer (channels 0 and 1) only.

**Table 29.10 Timing of On-Chip Peripheral Modules (1)**Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 32.768\text{ kHz}^*$ , 5 MHz to 25 MHzCondition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 32.768\text{ kHz}^*$ , 5 MHz to 20 MHz

Item	Symbol	Condition A		Condition B		Unit	Test Conditions		
		Min	Max	Min	Max				
I/O ports	Output data delay time	$t_{PWD}$	—	40	—	50	ns	Figure 29.15	
	Input data setup time	$t_{PRS}$	30	—	40	—			
	Input data hold time	$t_{PRH}$	30	—	40	—			
FRT	Timer output delay time	$t_{FTOD}$	—	40	—	50	ns	Figure 29.16	
	Timer input setup time	$t_{FTIS}$	30	—	40	—			
	Timer clock input setup time	$t_{FTCS}$	30	—	40	—		Figure 29.17	
	Timer clock pulse width	Single edge	$t_{FTCWH}$	1.5	—	1.5	—	$t_{cyc}$	
Both edges		$t_{FTCWL}$	2.5	—	2.5	—			
TMR	Timer output delay time	$t_{TMOD}$	—	40	—	50	ns	Figure 29.18	
	Timer reset input setup time	$t_{TMRS}$	30	—	40	—		Figure 29.20	
	Timer clock input setup time	$t_{TMCS}$	30	—	40	—		Figure 29.19	
	Timer clock pulse width	Single edge	$t_{TMCWH}$	1.5	—	1.5	—	$t_{cyc}$	
Both edges		$t_{TMCWL}$	2.5	—	2.5	—			
PWM, PWMX	Pulse output delay time	$t_{PWOD}$	—	40	—	50	ns	Figure 29.21	
SCI	Input clock cycle	Asynchronous	$t_{Scyc}$	4	—	4	—	$t_{cyc}$	Figure 29.22
		Synchronous		6	—	6	—		
	Input clock pulse width	$t_{SCKW}$	0.4	0.6	0.4	0.6	$t_{cyc}$		
	Input clock rise time	$t_{SCKr}$	—	1.5	—	1.5	$t_{cyc}$		
	Input clock fall time	$t_{SCKf}$	—	1.5	—	1.5			
	Transmit data delay time (synchronous)	$t_{TXD}$	—	40	—	50	ns	Figure 29.23	
	Receive data setup time (synchronous)	$t_{RXS}$	30	—	40	—			
	Receive data hold time (synchronous)	$t_{RXH}$	30	—	40	—			
A/D converter	Trigger input setup time	$t_{TRGS}$	30	—	40	—	ns	Figure 29.24	
WDT	$\overline{\text{RESO}}$ output delay time	$t_{RESD}$	—	200	—	200	ns	Figure 29.25	
	$\overline{\text{RESO}}$ output pulse width	$t_{RESOW}$	132	—	132	—	$t_{cyc}$		

Note: \* Only the on-chip peripheral modules that can be used in subclock operation.

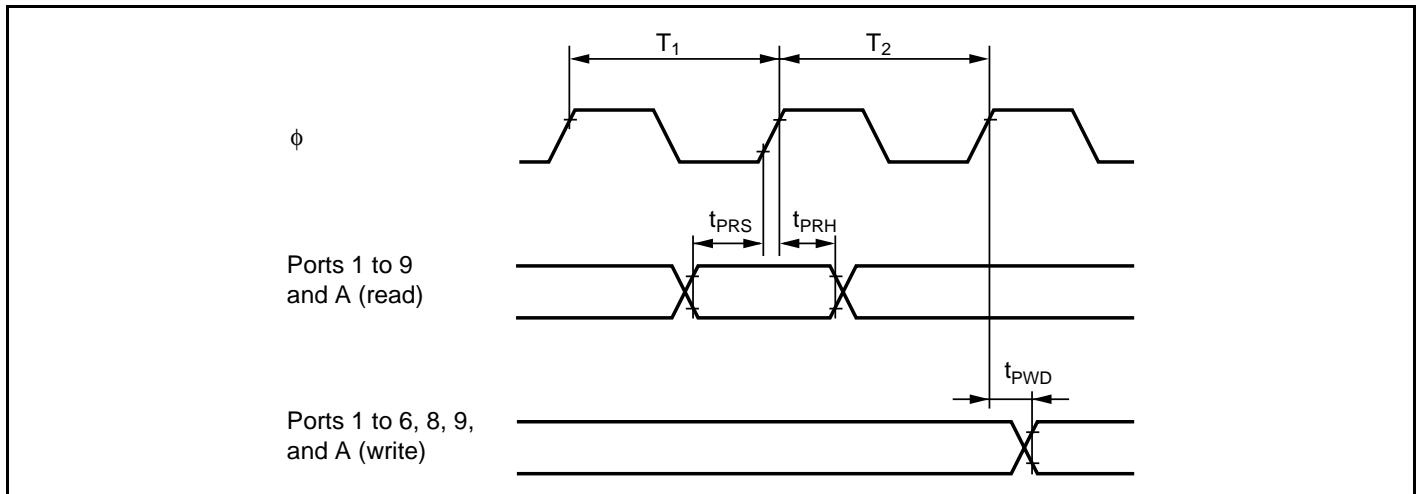


Figure 29.15 I/O Port Input/Output Timing

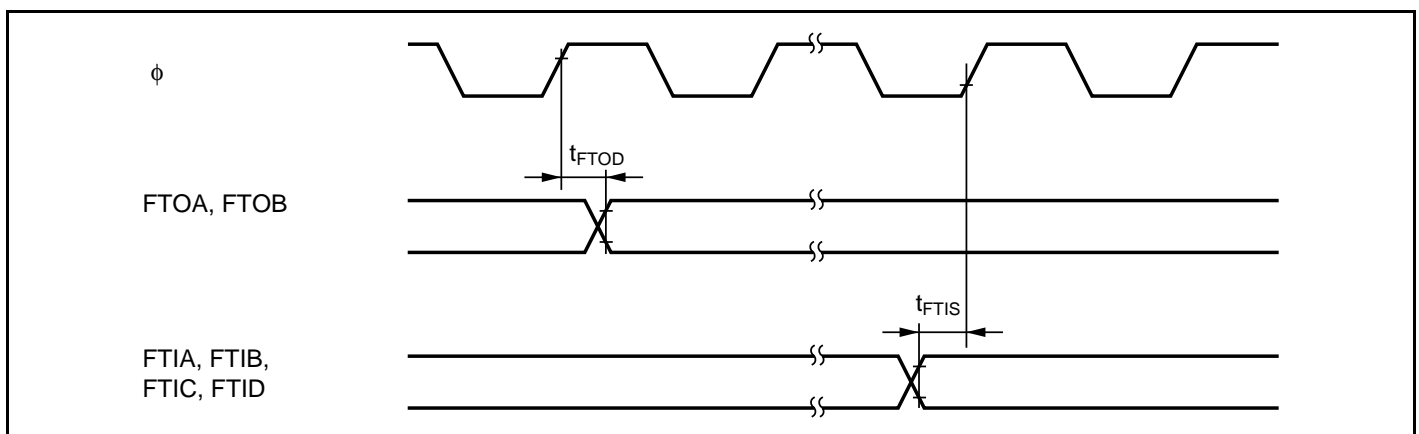


Figure 29.16 FRT Input/Output Timing

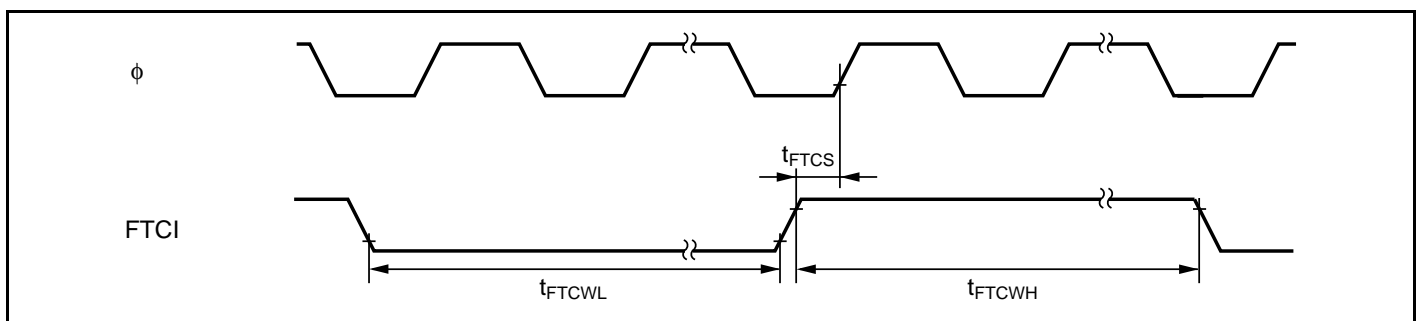
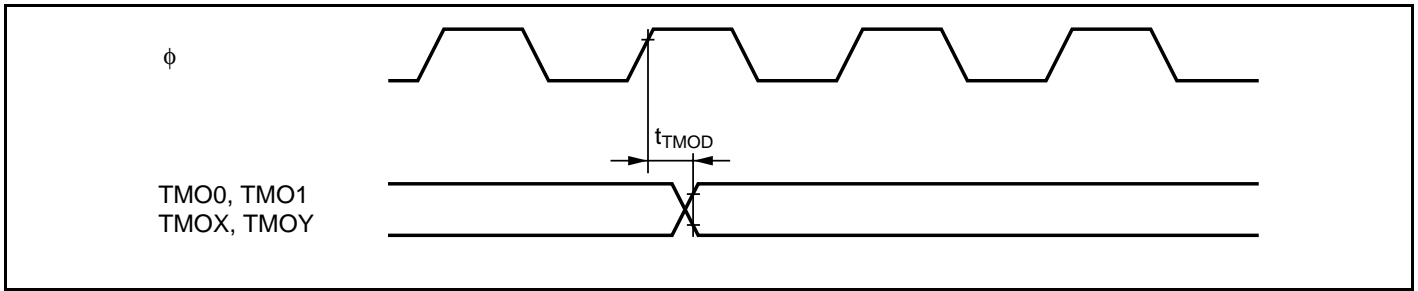
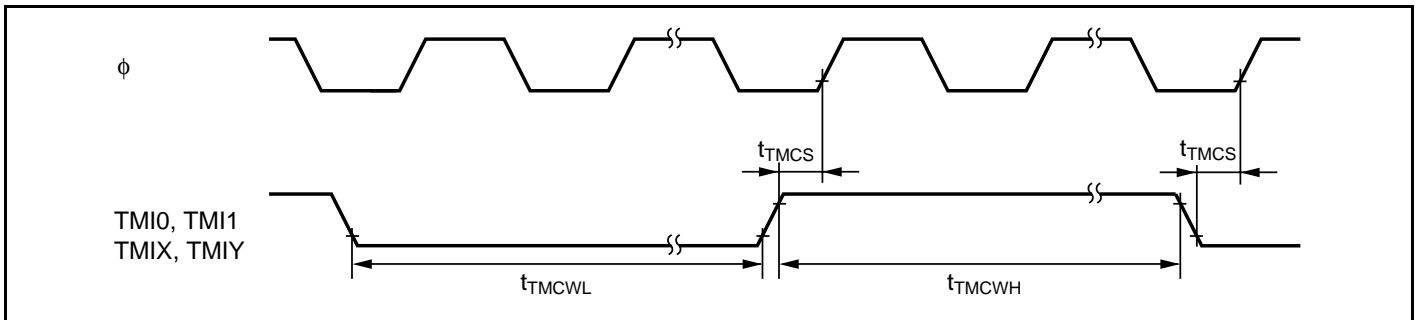


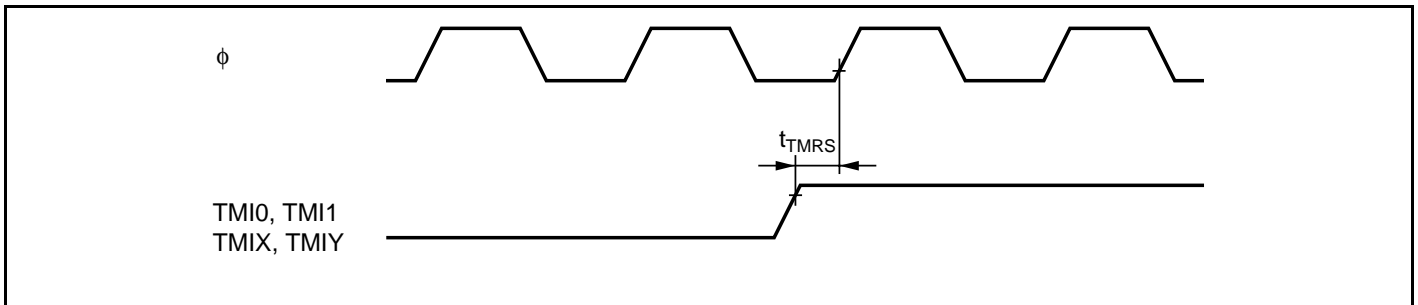
Figure 29.17 FRT Clock Input Timing



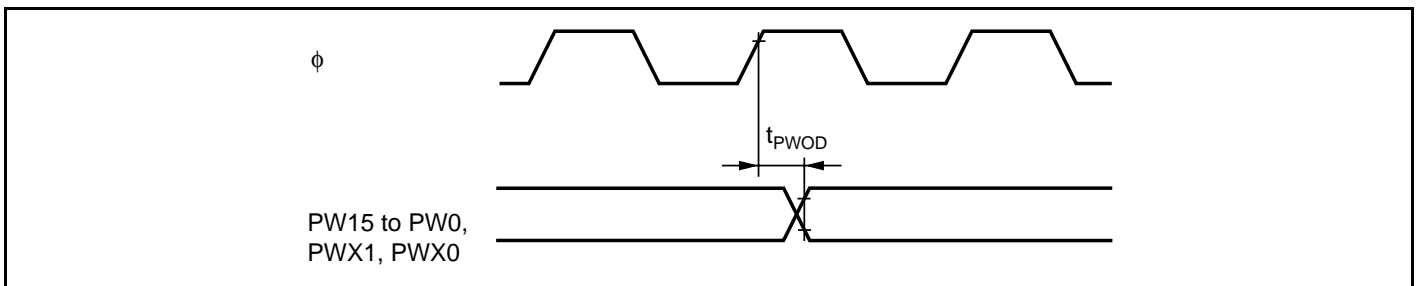
**Figure 29.18 8-Bit Timer Output Timing**



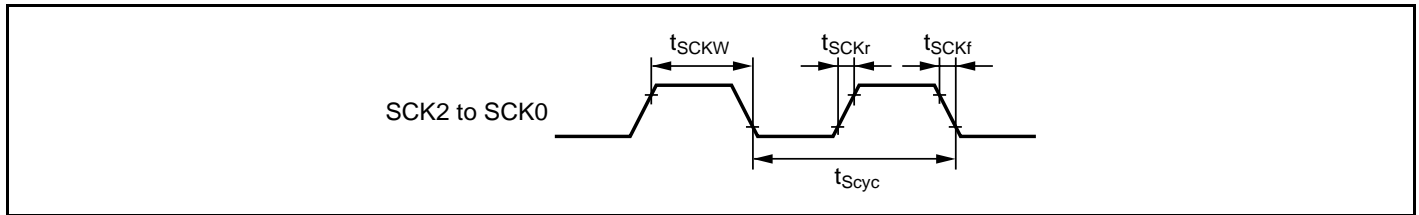
**Figure 29.19 8-Bit Timer Clock Input Timing**



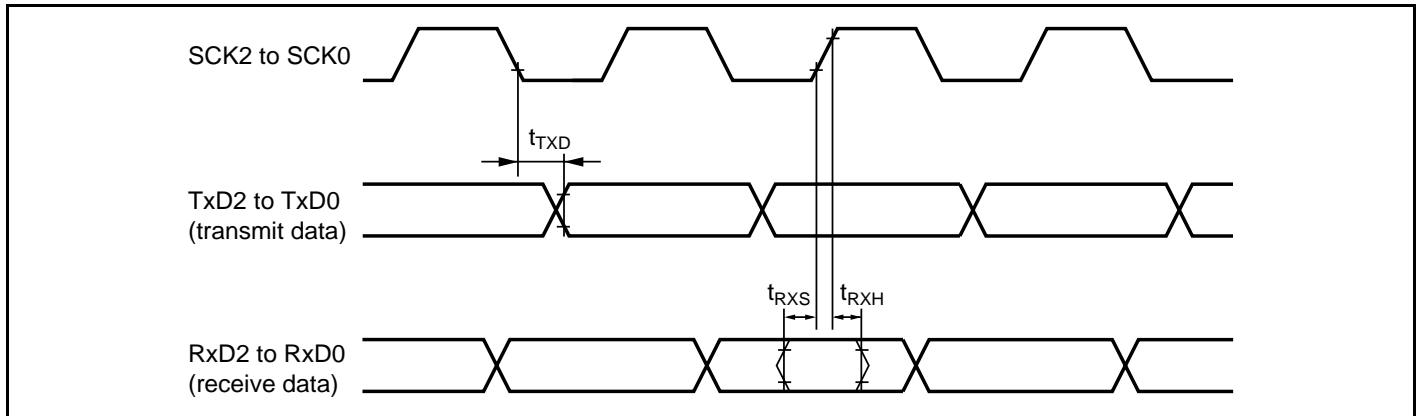
**Figure 29.20 8-Bit Timer Reset Input Timing**



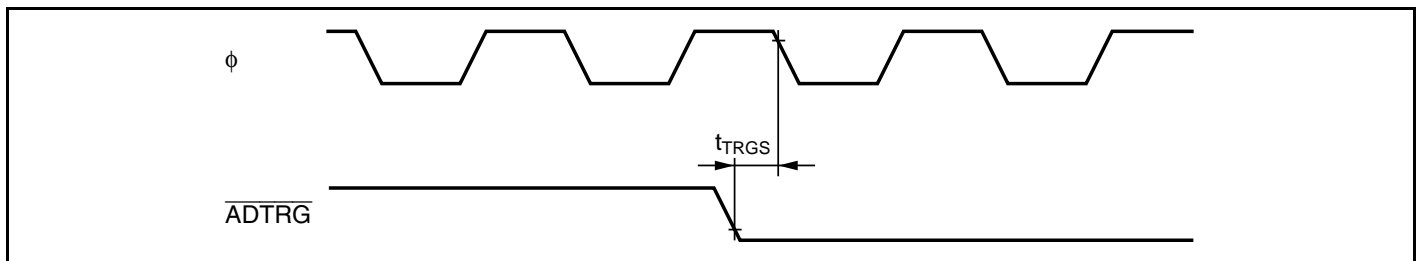
**Figure 29.21 PWM, PWMX Output Timing**



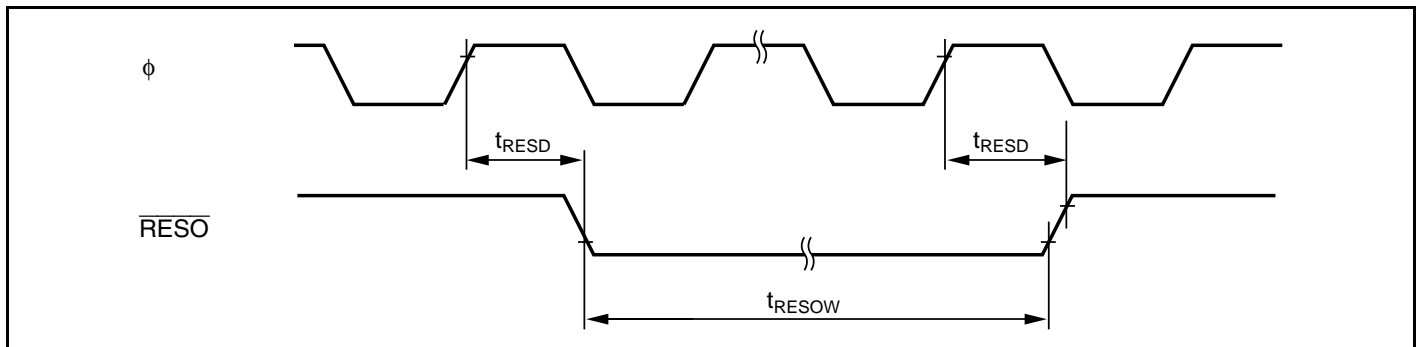
**Figure 29.22 SCK Clock Input Timing**



**Figure 29.23 SCI Input/Output Timing (Clock Synchronous Mode)**



**Figure 29.24 A/D Converter External Trigger Input Timing**



**Figure 29.25 WDT Output Timing ( $\overline{\text{RESO}}$ )**

**Table 29.11 I<sup>2</sup>C Bus Timing**Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }25\text{ MHz}$ Condition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$ 

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
SCL input cycle time	$t_{SCL}$	12	—	—	$t_{cyc}$	Figure 29.26
SCL input high pulse width	$t_{SCLH}$	3	—	—		
SCL input low pulse width	$t_{SCLL}$	5	—	—		
SCL, SDA input rise time	$t_{Sr}$	—	—	7.5*		
SCL, SDA input fall time	$t_{Sf}$	—	—	300	ns	
SCL, SDA output fall time	$t_{Of}$	20 + 0.1 $C_b$	—	250		
SCL, SDA input spike pulse elimination time	$t_{SP}$	—	—	1		
SDA input bus free time	$t_{BUF}$	5	—	—	$t_{cyc}$	
Start condition input hold time	$t_{STAH}$	3	—	—		
Retransmission start condition input setup time	$t_{STAS}$	3	—	—		
Stop condition input setup time	$t_{STOS}$	3	—	—		
Data input setup time	$t_{SDAS}$	0.5	—	—		
Data input hold time	$t_{SDAH}$	0	—	—	ns	
SCL, SDA capacitive load	$C_b$	—	—	400	pF	

Note: \* 17.5  $t_{cyc}$  can be set according to the clock selected for use by the IIC module.

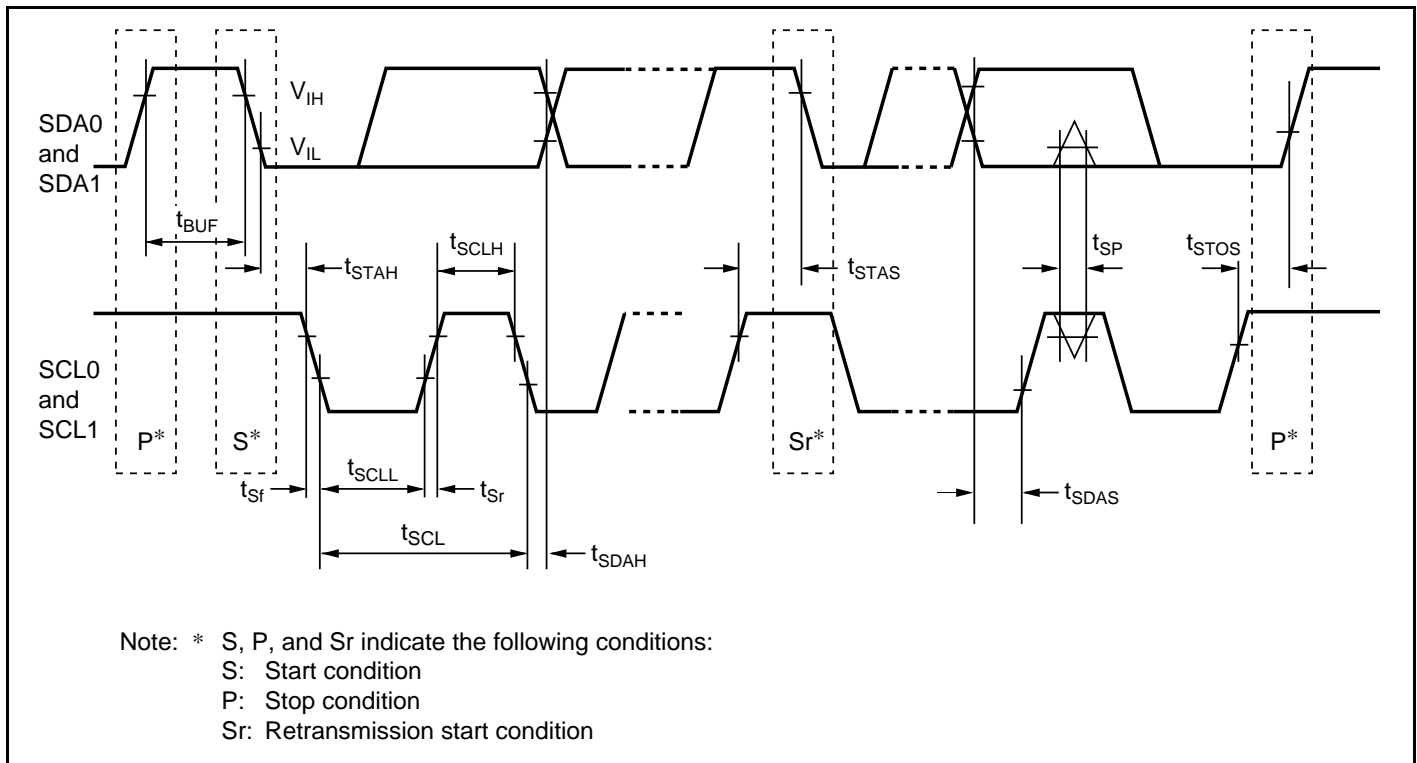
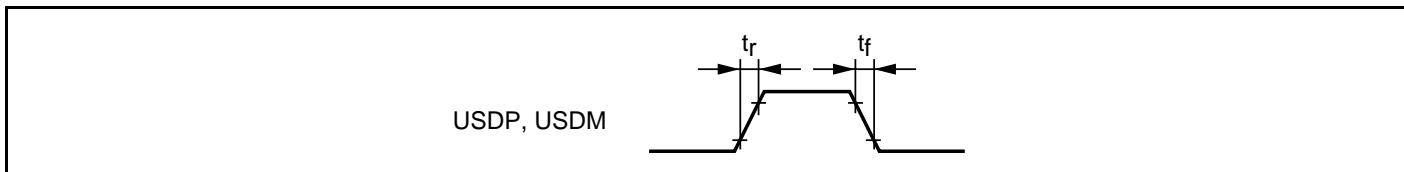
Figure 29.26 I<sup>2</sup>C Bus Interface Input/Output Timing

Table 29.12 USB Timing

Conditions:  $V_{CC} = 3.3 \text{ V} \pm 0.3 \text{ V}$ ,  $\text{Dr}V_{CC} = 3.3 \text{ V} \pm 0.3 \text{ V}$ ,  $\text{Dr}V_{SS} = V_{SS} = 0 \text{ V}$

Pin Functions: Driver/receiver input/output (USDP, USDM), USEXCL

Item	Symbol	Min	Max	Unit	Test Conditions	Remarks	
Driver/receiver (full speed)	Rise time	$t_r$	4	20	ns	Figure 29.27	Low: $0.1 \times \text{Dr}V_{CC}$ High: $0.9 \times \text{Dr}V_{CC}$ $t_r/t_f$
	Fall time	$t_f$	4	20			
	Differential signal time difference	$t_{RFM}$	90.0	111.11	%		
Driver/receiver output signal crossover voltage	$V_{CRS}$	1.3	2.0	V			
USB clock oscillation stabilization time (crystal)	$t_{OSCU}$	10	—	ms			



**Figure 29.27 USB Driver/Receiver Output Timing**

**Table 29.13 Multimedia Card Interface**

Conditions:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$

- MCIF: ExMCCLK/MCCLK timing

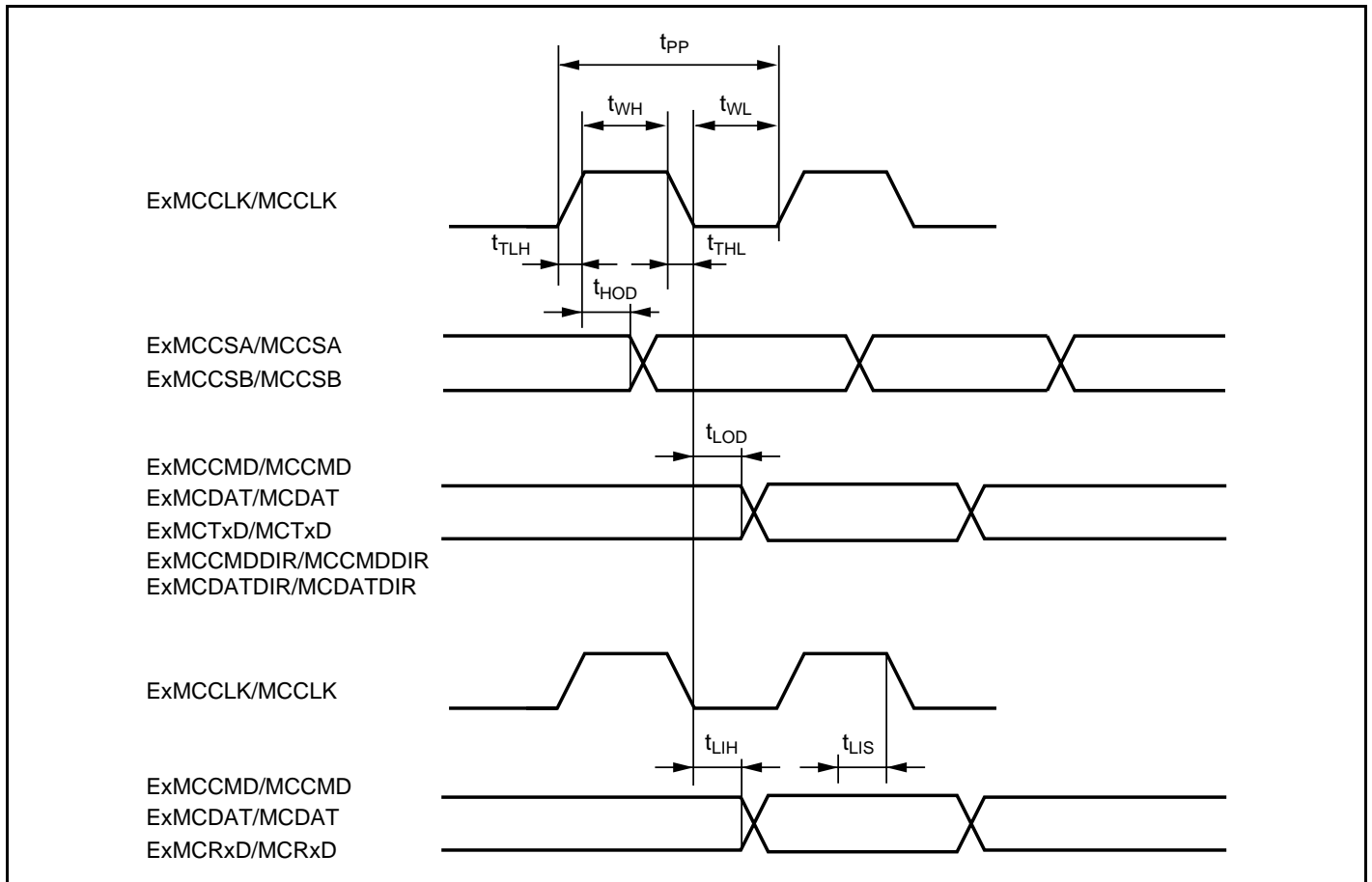
Item	Symbol	$CL \leq 100\text{ pF}^*$		$CL \leq 250\text{ pF}^*$		Unit	Test Conditions
		Min	Max	Min	Max		
MCCLK cycle time	$t_{PP}$	0	20	0	5	MHz	Figure 29.28
MCCLK high pulse width	$t_{WH}$	10	—	50	—	ns	
MCCLK low pulse width	$t_{WL}$	10	—	50	—		
MCCLK rise time	$t_{TLH}$	—	10	—	50		
MCCLK fall time	$t_{THL}$	—	10	—	50		

Note: \* Total load capacitance of external signal lines connected to the ExMCCLK/MCCLK pin

- MCIF: Card bus timing

Item	Symbol	Min	Max	Unit	Test Conditions
Output data delay time (based on MCCLK rising edge)	$t_{HOD}$	5	15	ns	Figure 29.28
Output data delay time (based on MCCLK falling edge)	$t_{LOD}$	—	10		
Input data hold time	$t_{LIH}$	10	—		
Input data setup time	$t_{LIS}$	5	—		

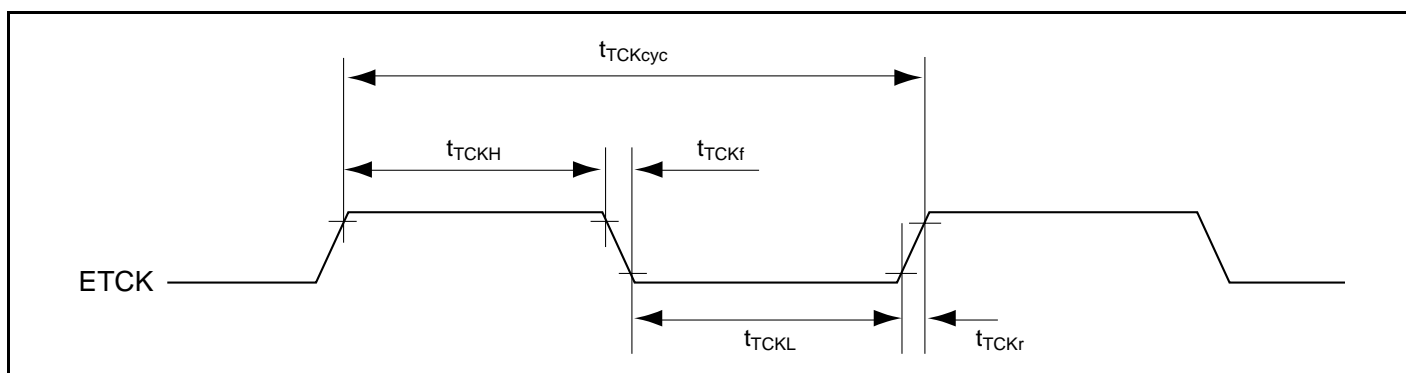


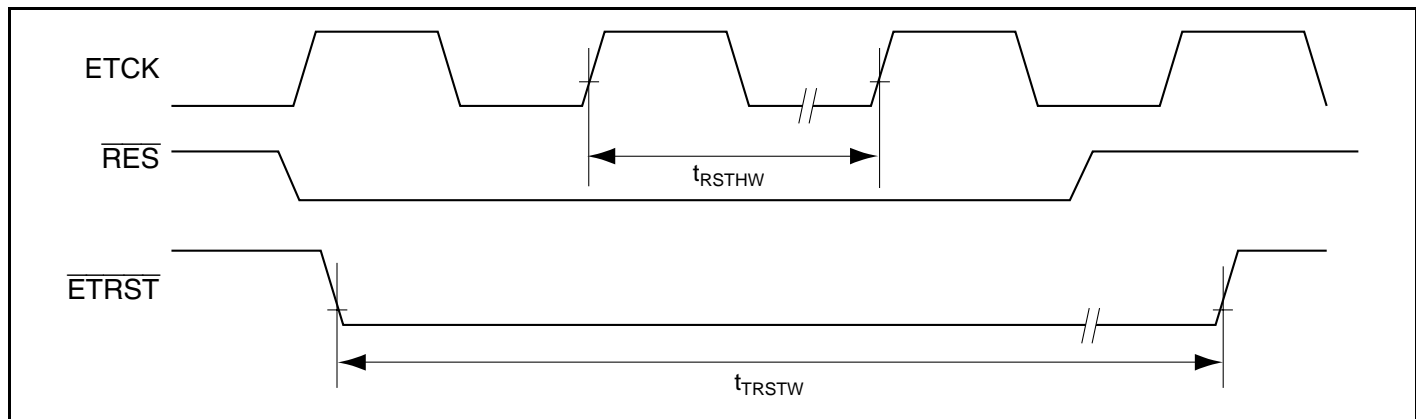


**Figure 29.28 Multimedia Card Interface Timing**

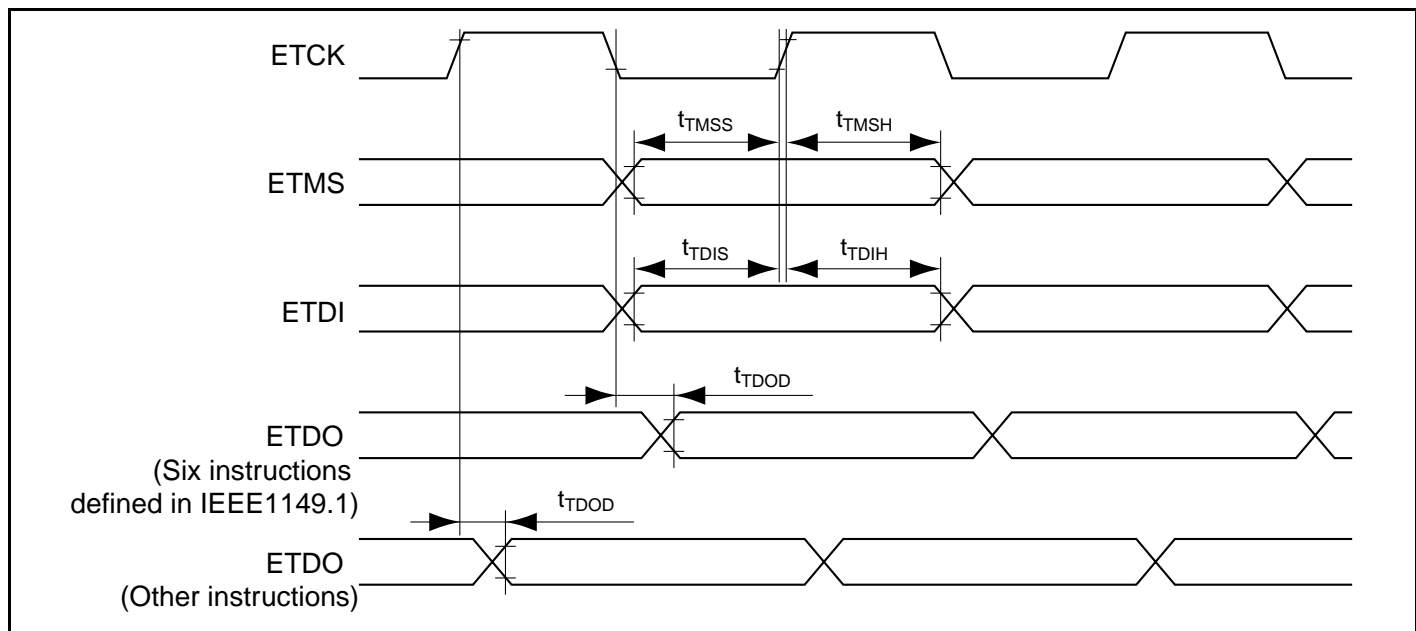
**Table 29.14 H-UDI Timing**Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }25\text{ MHz}$ Condition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$ 

Item	Symbol	Min	Max	Unit	Test Conditions
ETCK clock cycle time	$t_{TCKcyc}$	40*	500*	ns	Figure 29.29
ETCK clock high pulse width	$t_{TCKH}$	15	—		
ETCK clock low pulse width	$t_{TCKL}$	15	—		
ETCK clock rise time	$t_{TCKr}$	—	5		
ETCK clock fall time	$t_{TCKf}$	—	5		
ETRST pulse width	$t_{TRSTW}$	20	—	$t_{cyc}$	Figure 29.30
Reset hold transition pulse width	$t_{RSTHW}$	10	—		
ETMS setup time	$t_{TMSS}$	20	—	ns	Figure 29.31
ETMS hold time	$t_{TMSH}$	20	—		
ETDI setup time	$t_{TDIS}$	20	—		
ETDI hold time	$t_{TDIH}$	20	—		
ETDO data delay time	$t_{TDOD}$	—	20		

Note: \*  $t_{cyc} \geq t_{TCKcyc}$ **Figure 29.29 H-UDI ETCK Timing**



**Figure 29.30 Reset Hold Timing**



**Figure 29.31 H-UDI Input/Output Timing**

## 29.4 A/D Conversion Characteristics

Tables 29.15 and 29.16 list the A/D conversion characteristics.

**Table 29.15 A/D Conversion Characteristics  
(AN7 to AN2 Input: 134/266-State Conversion)**

Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{ref} = 3.0\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }25\text{ MHz}$

Condition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{ref} = 2.7\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$

Item	Condition A			Condition B			Unit
	Min	Typ	Max	Min	Typ	Max	
Resolution	10	10	10	10	10	10	Bits
Conversion time*	—	—	5.36	—	—	6.7	$\mu\text{s}$
Analog input capacitance	—	—	20	—	—	20	pF
Permissible signal-source impedance	—	—	5	—	—	5	k $\Omega$
Nonlinearity error	—	—	$\pm 7.0$	—	—	$\pm 7.0$	LSB
Offset error	—	—	$\pm 7.5$	—	—	$\pm 7.5$	
Full-scale error	—	—	$\pm 7.5$	—	—	$\pm 7.5$	
Quantization error	—	—	$\pm 0.5$	—	—	$\pm 0.5$	
Absolute accuracy	—	—	$\pm 8.0$	—	—	$\pm 8.0$	

Note: \* Value when using the maximum operating frequency in single mode of 134 states.

**Table 29.16 A/D Conversion Characteristics**  
**(CIN7 to CIN0 Input: 134/266-State Conversion)**

Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{ref} = 3.0\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }25\text{ MHz}$

Condition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{ref} = 2.7\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$

Item	Condition A			Condition B			Unit
	Min	Typ	Max	Min	Typ	Max	
Resolution	10	10	10	10	10	10	Bits
Conversion time*	—	—	5.36	—	—	6.7	$\mu\text{s}$
Analog input capacitance	—	—	20	—	—	20	pF
Permissible signal-source impedance	—	—	5	—	—	5	$\text{k}\Omega$
Nonlinearity error	—	—	$\pm 11.0$	—	—	$\pm 11.0$	LSB
Offset error	—	—	$\pm 11.5$	—	—	$\pm 11.5$	
Full-scale error	—	—	$\pm 11.5$	—	—	$\pm 11.5$	
Quantization error	—	—	$\pm 0.5$	—	—	$\pm 0.5$	
Absolute accuracy	—	—	$\pm 12.0$	—	—	$\pm 12.0$	

Note: \* Value when using the maximum operating frequency in single mode of 134 states.

## 29.5 D/A Conversion Characteristics

Table 29.17 lists the D/A conversion characteristics.

**Table 29.17 D/A Conversion Characteristics**

Condition A:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $AV_{ref} = 3.0\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }25\text{ MHz}$

Condition B:  $V_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{CC} = 2.7\text{ V to }3.6\text{ V}$ ,  $AV_{ref} = 2.7\text{ V to }AV_{CC}$ ,  
 $V_{SS} = AV_{SS} = 0\text{ V}$ ,  $\phi = 5\text{ MHz to }20\text{ MHz}$

Item		Min	Typ	Max	Unit
Resolution		8	8	8	Bits
Conversion time	Load capacitance 20 pF	—	—	10	$\mu\text{s}$
Absolute accuracy	Load resistance 2 M $\Omega$	—	$\pm 2.0$	$\pm 3.0$	LSB
	Load resistance 4 M $\Omega$	—	—	$\pm 2.0$	

## 29.6 Flash Memory Characteristics

Table 29.18 lists the flash memory characteristics.

**Table 29.18 Flash Memory Characteristics**

Condition:  $V_{CC} = 3.0\text{ V to }3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$

Item	Symbol	Min	Typ	Max	Unit	Test Conditions	
Programming time <sup>*1 *2 *4</sup>	tP	—	10	200	ms/ 128 bytes		
Erase time <sup>*1 *3 *6</sup>	tE	—	10	1200	ms/ block		
Reprogramming count	N <sub>WEC</sub>	100 <sup>*8</sup>	10,000 <sup>*9</sup>	—	Times		
Data retention time <sup>*10</sup>	t <sub>DRP</sub>	10	—	—	Years		
Programming	Wait time after SWE-bit setting <sup>*1</sup>	x	1	—	—	μs	
	Wait time after PSU-bit setting <sup>*1</sup>	y	50	—	—		
	Wait time after P-bit setting <sup>*1 *4</sup>	z1	28	30	32		1 ≤ n ≤ 6
		z2	198	200	202		7 ≤ n ≤ 1000
		z3	8	10	12		Additional programming
	Wait time after P-bit clear <sup>*1</sup>	α	5	—	—		
	Wait time after PSU-bit clear <sup>*1</sup>	β	5	—	—		
	Wait time after PV-bit setting <sup>*1</sup>	γ	4	—	—		
	Wait time after dummy write <sup>*1</sup>	ε	2	—	—		
	Wait time after PV-bit clear <sup>*1</sup>	η	2	—	—		
	Wait time after SWE-bit clear <sup>*1</sup>	θ	100	—	—		
	Maximum programming count <sup>*1 *4 *5</sup>	N	—	—	1000	Time	
	Erase	Wait time after SWE-bit setting <sup>*1</sup>	x	1	—	—	μs
Wait time after ESU-bit setting <sup>*1</sup>		y	100	—	—		
Wait time after E-bit setting <sup>*1 *6</sup>		z	10	—	100	ms	
Wait time after E-bit clear <sup>*1</sup>		α	10	—	—	μs	
Wait time after ESU-bit clear <sup>*1</sup>		β	10	—	—		
Wait time after EV-bit setting <sup>*1</sup>		γ	20	—	—		
Wait time after dummy write <sup>*1</sup>		ε	2	—	—		
Wait time after EV-bit clear <sup>*1</sup>		η	4	—	—		
Wait time after SWE-bit clear <sup>*1</sup>		θ	100	—	—		
Maximum erase count <sup>*1 *6 *7</sup>	N	—	—	120	Times		

Notes: 1. Make each time setting in accordance with the program/erase algorithm.

2. Programming time per 128 bytes (Shows the total period for which the P bit in FLMCR1 is set. It does not include the program verification time.)
3. Block erase time (Shows the total period for which the E bit in FLMCR1 is set. It does not include the erase verification time.)
4. Maximum programming time ( $t_{P\ max}$ )  
$$t_{P\ max} = (\text{wait time after P-bit setting } (z1) + (z3)) \times 6$$
$$+ \text{wait time after P-bit setting } (z2) \times ((N) - 6)$$
5. Set the maximum programming count (N) to a value less than the maximum programming time ( $t_{P\ max}$ ), with referencing the actual z1, z2, and z3 settings. The wait time after P-bit setting (z1, z2, and z3) should be changed depending on the programming count (n).  
Programming count(n)  
 $1 \leq n \leq 6 \quad z1 = 30 \mu\text{s}, z3 = 10 \mu\text{s}$   
 $7 \leq n \leq 1000 \quad z2 = 200 \mu\text{s}$
6. Maximum erase time ( $t_{E\ max}$ )  
 $t_{E\ max} = \text{wait time after E-bit setting } (z) \times \text{maximum erase count } (N)$
7. Set the maximum erase count (N) to a value less than the maximum erase time ( $t_{E\ max}$ ), with referencing the actual (z) setting.
8. Minimum number of times for which all characteristics are guaranteed after rewriting. (Guarantee range is 1 to minimum value.)
9. Reference value for 25°C (as a guideline, rewriting should normally function up to this value).
10. Data retention characteristic when rewriting is performed within the specification range, including the minimum value.



# Appendix

## A. I/O Port States in Each Pin State

Port Name Pin Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode	Watch Mode	Sleep Mode	Subsleep Mode	Subactive Mode	Program Execution State
Port 1 A7 to A0, CPA7 to CPA0	2, 3 (EXPE = 1)	T	T	kept*	kept*	kept*	kept*	Address output/ CompactFlash address output/ input port	Address output/ CompactFlash address output/ input port
	2, 3 (EXPE = 0)							I/O port	I/O port
Port 27 to 24 A15 to A12	2, 3 (EXPE = 1)	T	T	kept*	kept*	kept*	kept*	Address output/ I/O port	Address output/ I/O port
	2, 3 (EXPE = 0)							I/O port	I/O port
Port 23 A11, CPREG	2, 3 (EXPE = 1)	T	T	kept*	kept*	kept*	kept*	Address output/ CPREG/ I/O port	Address output/ CPREG/ I/O port
	2, 3 (EXPE = 0)							I/O port	I/O port
Port 22 to 20 A10 to A8, CPA10 to CPA8	2, 3 (EXPE = 1)	T	T	kept*	kept*	kept*	kept*	Address output/ CompactFlash address output/ I/O port	Address output/ CompactFlash address output/ I/O port
	2, 3 (EXPE = 0)							I/O port	I/O port
Port 3 D15 to D8, CPD15 to CPD8	2, 3 (EXPE = 1)	T	T	T	T	T	T	D15 to D8/ CPD15 to CPD8	D15 to D8/ CPD15 to CPD8
	2, 3 (EXPE = 0)			kept	kept	kept	kept	I/O port	I/O port
Port 4	2, 3 (EXPE = 1)	T	T	kept	kept	kept	kept	I/O port	I/O port
	2, 3 (EXPE = 0)								
Port 5	2, 3 (EXPE = 1)	T	T	kept	kept	kept	kept	I/O port	I/O port
	2, 3 (EXPE = 0)								
Port 6 D7 to D0, CPD7 to CPU0	2, 3 (EXPE = 1)	T	T	kept	kept	kept	kept	D7 to D0/ CPD7 to CPD0/ I/O port	D7 to D0/ CPD7 to CPD0/ I/O port
	2, 3 (EXPE = 0)							I/O port	I/O port

Port Name Pin Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode	Watch Mode	Sleep Mode	Subsleep Mode	Subactive Mode	Program Execution State
Port 7	2, 3 (EXPE = 1)	T	T	T	T	T	T	Input port	Input port
	2, 3 (EXPE = 0)								
Port 8	2, 3 (EXPE = 1)	T	T	kept	kept	kept	kept	I/O port	I/O port
	2, 3 (EXPE = 0)								
Port 97 WAIT, CPWAIT, CS256	2, 3 (EXPE = 1)	T	T	T/kept	T/kept	T/kept	T/kept	WAIT/ CPWAIT/ CS256/ I/O port	WAIT/ CPWAIT/ CS256/ I/O port
	2, 3 (EXPE = 0)			kept	kept	kept	kept	I/O port	I/O port
Port 96	2, 3 (EXPE = 1)	T	T	[DDR = 1] H	EXCL input	[DDR = 1] Clock output	EXCL input	EXCL input	Clock output/ EXCL input/ Input port
	2, 3 (EXPE = 0)			[DDR = 0] T		[DDR = 0] T			
φ, EXCL	2, 3 (EXPE = 0)								
Port 95 to 93 AS, IOS, HWR, CPWE, RD, CPOE	2, 3 (EXPE = 1)	T	T	H	H	H	H	AS/IOS, HWR/CPWE, RD/CPOE	AS/IOS, HWR/CPWE, RD/CPOE
	2, 3 (EXPE = 0)			kept	kept	kept	kept	I/O port	I/O port
Port 92, 91 CPCS1, CPCS2	2, 3 (EXPE = 1)	T	T	kept	kept	kept	kept	CPCS1, CPCS2/ I/O port	CPCS1, CPCS2/ I/O port
	2, 3 (EXPE = 0)							I/O port	I/O port
Port 90 LWR	2, 3 (EXPE = 1)	T	T	H/kept	H/kept	H/kept	H/kept	LWR/ I/O port	LWR/ I/O port
	2, 3 (EXPE = 0)			kept	kept	kept	kept	I/O port	I/O port
Port A A17, A16	2, 3 (EXPE = 1)	T	T	kept*	kept*	kept*	kept*	A17, A16/ I/O port	A17, A16/ I/O port
	2, 3 (EXPE = 0)							I/O port	I/O port

## Legend:

H: High level

L: Low level

T: High impedance

kept: Input ports are in the high-impedance state (when DDR = 0 and PCR = 1, the input pull-up MOS remains on).

Output ports maintain their previous state.

Depending on the pins, the on-chip peripheral modules may be initialized and the I/O port function determined by DDR and DR.

DDR: Data direction register

Note: \* In the case of address output, the last address accessed is retained.

---

**B. Product Lineup**

<b>Product Type</b>		<b>Type Code</b>	<b>Mark Code</b>	<b>Package (Code)</b>
H8S/2158	F-ZTAT version	HD64F2158	F2158VBQ25	112-pin TFBGA (TBP-112A)

---

## C. Package Dimensions

For package dimensions, dimensions described in Renesas Package Data Book have priority.

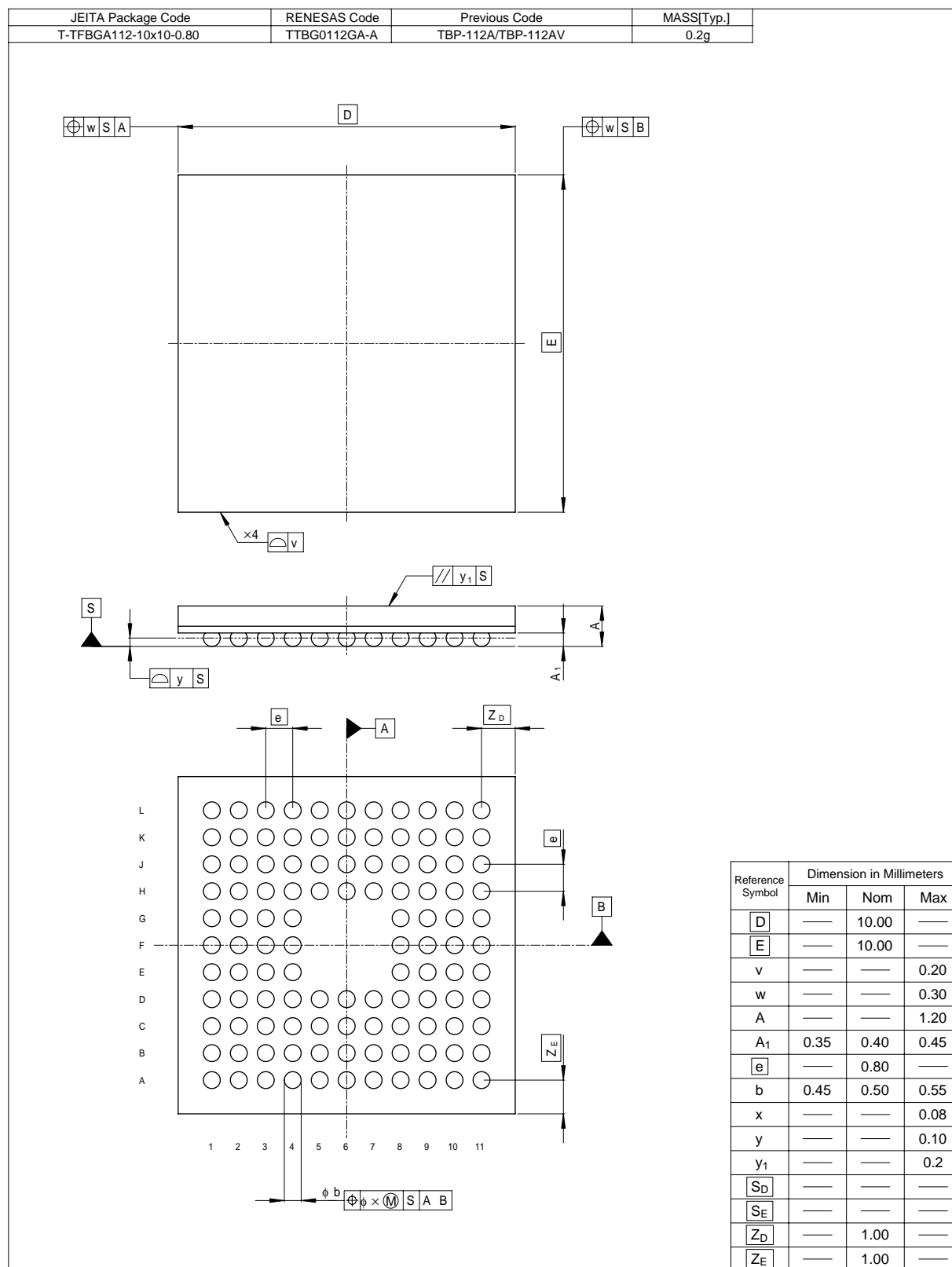


Figure C.1 Package Dimensions (TBP-112A)

# Index

14-Bit PWM Timer (PWMX).....	273	Block Transfer Mode .....	159
16-Bit Access Space .....	123	Boot Mode .....	728
16-Bit Count Mode .....	335	Boundary overflow.....	202
16-Bit Free-Running Timer (FRT) .....	287	Boundary Scan .....	756
16-Bit, 2-State Access Space .....	127	Branch Instructions .....	41
16-Bit, 3-State Access Space .....	130	Broadcast Commands.....	656
256-kbyte Expansion Area.....	119	bulk transfer .....	603
8-Bit Access Space .....	123	Burst ROM Interface.....	134
8-Bit PWM Timer (PWM).....	261	Bus Arbitration.....	142
8-Bit Timer (TMR).....	315	Bus Control .....	113
8-Bit, 2-State Access Space .....	125	Bus Controller .....	103
8-Bit, 3-State Access Space .....	126	Bus Master .....	142
		Bus Specifications of Basic Bus Interface	
A/D Conversion Time.....	706	.....	117
A/D Converter .....	697	Carrier frequency .....	264
Absolute Address.....	46	Cascaded Connection .....	335
Acknowledge signal.....	517	CBLANK Output .....	372
Activation by Interrupt.....	164	CF Expansion Area (Memory Card Mode)	
Activation by Software .....	164	.....	121
additional pulse.....	271	Chain Transfer.....	160
Address Map.....	62	Clamp Waveform.....	359
Address Ranges and External Address		Clock Pulse Generator.....	761
Spaces .....	115	Clocked Synchronous Mode .....	437
Address Space.....	24	Commands .....	634
Addressing Modes .....	44	Commands Not Requiring Command	
Advanced Mode.....	22, 121	Response .....	657
Arithmetic Operations Instructions.....	36	Commands with Read Data.....	663, 679
Asynchronous Mode .....	417	Commands with Write Data.....	669, 683
		Commands without Data Transfer ..	659, 675
base cycle.....	281	Compare-Match Count Mode .....	335
Basic Bus Interface .....	123	Condition field .....	43
Basic Expansion Area .....	118	Condition-Code Register (CCR).....	28
Basic Operation Timing.....	125, 135, 138	Consecutive Data Blocks .....	192
basic pulse.....	270	control transfer .....	602
Bit Manipulation Instructions .....	39	conversion cycle.....	281
bit rate .....	405	CP Expansion Area (Basic Mode) .....	120
blanking waveform .....	372	CPU.....	17
Block Data Transfer Instructions .....	43		

CPU Operating Modes.....	20	HSYNCO Output .....	370
CRC Operation Circuit .....	467	I/O Ports .....	205
Crystal Oscillator .....	762	I/O Select Signals.....	122
D/A Converter .....	691	I <sup>2</sup> C Bus Formats .....	516
data direction register .....	205	I <sup>2</sup> C bus interface.....	539
data register.....	205	I <sup>2</sup> C Bus Interface (IIC).....	473
Data Size and Data Alignment.....	123, 137	ID code.....	731
Data Transfer Controller (DTC) .....	145	ID Numbers.....	183
Data Transfer Instructions .....	35	Idle Cycle .....	141
Direct Transitions .....	788	IIC Operation Reservation Adapter .....	510
Divided Waveform Period Measurement	361	Immediate .....	46
DTC Comparator Scan .....	704	IN Token .....	614
DTC Vector Table .....	153	Input Capture .....	302
Effective Address.....	48	Input Capture Operation.....	336
Effective address extension .....	43	input pull-up MOS control register .....	205
Encryption Operation Circuit (DES and GF)	689	Instruction Set .....	33
.....	689	Internal Block Diagram.....	2
Endpoint.....	557, 622	Internal Synchronization Signal .....	367
Erase/Erase-Verify.....	735	Interrupt	
erasing units.....	722	ADI .....	708
Error Protection .....	737	CMIA .....	338, 339
Exception Handling .....	65	CMIB .....	338, 339
Exception Handling Vector Table .....	66	ERI .....	460
Extended Control Register.....	27	ICI .....	339
External Clock .....	763	ICIA .....	308
External Trigger.....	708	ICIB .....	308
FIFO empty.....	202	ICIC .....	308
FIFO full.....	202	ICID .....	308
FIFO overread.....	202	ICRA.....	820
FIFO overwrite .....	202	ICC .....	540
flash memory .....	717	IICM.....	540
framing error .....	427	IICR .....	540
General Registers.....	26	IICT.....	540
Hardware Protection .....	737	MMCIA.....	687
Hardware Standby Mode .....	784	MMCIB.....	687
		MMCIC.....	687
		OVI .....	339
		RXI.....	459, 460
		SWDTEND .....	161

TEI.....	459, 460	Number of DTC Execution States.....	163
TXI.....	459, 460. <i>See</i>	Number of FIFO Bytes.....	174
USB10 .....	625	On-Board Programming.....	727
USB11 .....	625	Operating Modes.....	55
USB12 .....	625	Operation field .....	43
USB13 .....	625	Operation Reservation Commands.....	513
WOVI .....	382	OUT Token .....	610
WUE15 to WUE8 Interrupts.....	85	Output Compare .....	301
Interrupt Control Modes .....	90	overrun error .....	427
Interrupt Controller .....	73	parity error .....	427
Interrupt Exception Handling .....	69	Pin Arrangement .....	3
Interrupt Exception Handling Sequence ...	96	Pin Arrangement in Each Operating Mode .	4
Interrupt Exception Handling Vector Table		Pin Functions.....	8
.....	86	PLL Circuit .....	767
Interrupt Mask Bit.....	28	pointer sets .....	167
interrupt transfer .....	603	Power-Down Modes.....	771
Interval Timer Mode.....	381	Processing States.....	50
IrDA.....	456	Program Counter .....	27
IRQ15 to IRQ0 Interrupts.....	84	Program/Erase Protection.....	737
		Program/Program-Verify .....	733
KIN9 to KIN0 Interrupts.....	85	Program-Counter Relative .....	47
		Programmer Mode .....	738
Logic Operations Instructions.....	38	programming units .....	722
LSI Internal States in Each Operating Mode		PWM conversion period .....	264
.....	780	PWM Decoding.....	358
Master Receive Operation.....	519	RAM .....	715
Master Transmit Operation .....	517	RAM-FIFO Unit .....	167
Medium-Speed Mode .....	781	Register Configuration.....	25
Memory Card Interface.....	137	Register Direct .....	45
Memory Indirect .....	47	Register field .....	43
MMC Mode .....	656	Register Indirect.....	45
Mode Transition Diagram.....	718, 779	Register Indirect with Displacement .....	45
Module Stop Mode .....	787	Register Indirect with Post-Increment.....	45
Multimedia Card Interface (MCIF).....	627	Register Indirect with Pre-Decrement.....	46
Multiprocessor Communication Function		Registers	
.....	431	ABRKCR .....	77, 797, 808, 818
		ADCR .....	702, 802, 813, 822
NMI Interrupt.....	84		
Normal Mode.....	20, 122, 157		

ADCSR.....	701, 802, 813, 822	DTCERE.....	150, 797, 808, 818
ADDR.....	700, 801, 813, 822	DTCRA.....	173, 795, 807, 817
BAR.....	170	DTCRB.....	175, 795, 807, 817
BARA.....	78, 797, 808, 818	DTCRC.....	180
BARB.....	78, 797, 808, 818	DTCRD.....	181, 795, 807, 817
BARC.....	78, 797, 808, 818	DTIDR.....	178, 795, 807, 817
BCR.....	106, 800, 812, 821	DTIDSRA.....	178, 795, 807, 817
BCR2.....	108, 797, 809, 819	DTIDSRB.....	179, 795, 807, 817
BRR.....	405, 801, 812, 822	DTIER.....	181, 795, 807, 817
CLKON.....	654, 792, 804, 815	DTOUTR.....	644, 793, 805, 816
CMDR0.....	637, 792, 804, 815	DTRSR.....	181, 795, 807, 817
CMDR1.....	637, 792, 804, 815	DTSTRA.....	179, 795, 807, 817
CMDR2.....	637, 792, 804, 815	DTSTRB.....	180, 795, 807, 817
CMDR3.....	637, 792, 804, 815	DTSTRC.....	176, 795, 807, 817
CMDR4.....	637, 792, 804, 815	DTVECR.....	151, 797, 808, 818
CMDR5.....	637, 792, 804, 815	EBR1.....	726, 797, 809, 819
CMDSTRT.....	640	EBR2.....	726, 797, 809, 819
CMDTYR.....	632, 792, 804, 815	EP4PKTSZR.....	598, 793, 805, 816
CONFV.....	597, 793, 805, 816	EPDIR0.....	563, 794, 806, 817
CRA.....	149	EPDR0I.....	560, 794, 806, 816
CRB.....	149	EPDR0O.....	559, 794, 806, 816
CRCCR.....	468, 796, 808, 818	EPDR0S.....	559, 793, 805, 816
CRCDIR.....	468, 796, 808, 818	EPDR1.....	560, 794, 806, 816
CRCDOR.....	468, 796, 808, 818	EPDR2.....	560, 793, 805, 816
CSTR.....	645, 792, 804, 815	EPDR3.....	560, 793, 805, 816
CTOCR.....	643, 792, 804, 815	EPRSTR0.....	587, 794, 806, 817
DACNTH.....	275, 799, 810, 820	EPSTLR0.....	584, 794, 806, 817
DACNTL.....	275, 799, 810, 820	EPSZR1.....	558, 794, 806, 816
DACR.....	277, 692, 799, 803, 810, 814, ..... 820, 823	FLMCR1.....	723, 797, 809, 819
DADR.....	692, 803, 814, 823	FLMCR2.....	725, 797, 809, 819
DADRA.....	276, 799, 810, 820	FRC.....	290, 798, 809, 819
DADRBB.....	277, 799, 810, 820	FREEN.....	172
DAR.....	149	FSTR.....	169, 794, 806, 817
DATAN.....	172	FVSR0IH.....	562, 794, 806, 816
DEVRSMR.....	589, 794, 806, 817	FVSR0IL.....	562, 794, 806, 816
DTCERA.....	150, 797, 808, 818	FVSR0OH.....	562, 794, 806, 816
DTCERB.....	150, 797, 808, 818	FVSR0OL.....	562, 794, 806, 816
DTCERC.....	150, 797, 808, 818	FVSR0SH.....	562, 793, 805, 816
DTCERD.....	150, 797, 808, 818	FVSR0SL.....	562, 793, 805, 816
		FVSR1H.....	562, 794, 806, 816



FVSR1L.....	562, 794, 806, 816	KMIMRA.....	83, 802, 813, 823
FVSR2H.....	562, 793, 805, 816	KMPCR6.....	234, 802, 813, 823
FVSR2L.....	562, 793, 805, 816	LPWRCR.....	774, 797, 809, 819
FVSR3H.....	562, 793, 805, 816	MDCR.....	56, 800, 811, 821
FVSR3L.....	562, 793, 805, 816	MODER.....	631, 792, 804, 815
ICCMD.....	510, 795, 807, 817	MRA.....	147
ICCNT.....	508, 795, 807, 817	MRB.....	148
ICCR.....	485, 801, 812, 822	MSTPCRH.....	777, 797, 809, 819
ICCRX.....	496, 795, 807, 817	MSTPCRL.....	777, 797, 809, 819
ICDR.....	477, 801, 822	NRA.....	172
ICDRS.....	508, 795, 796, 807, 812, 817	NWA.....	173
ICDRX.....	507, 795, 807, 817	OCRA.....	290, 798, 809, 820
ICMR.....	482, 801, 812, 822	OCRAF.....	291, 798, 810, 820
ICR.....	290, 798, 810	OCRAR.....	291, 798, 810, 820
ICRA.....	76, 796, 808, 818	OCRB.....	290, 798, 809, 820
ICRB.....	76, 796, 808, 818	OCRDM.....	291, 799, 810, 820
ICRC.....	76, 796, 808, 818	OPCR.....	641, 792, 804, 815
ICRD.....	76, 796, 808, 818	P1DDR.....	209, 800, 811, 821
ICSR.....	492, 801, 812, 822	P1DR.....	210, 800, 811, 821
ICSRA.....	497, 795, 807, 817	P1PCR.....	210, 799, 811, 821
ICSRB.....	499, 795, 807, 817	P2DDR.....	212, 800, 811, 821
ICSRC.....	502, 795, 807, 817	P2DR.....	213, 800, 811, 821
IER.....	81, 800, 811, 821	P2PCR.....	213, 799, 811, 821
IER16.....	81, 797, 808, 818	P3DDR.....	216, 800, 811, 821
INTCR0.....	647, 792, 804, 815	P3DR.....	216, 800, 811, 821
INTCR1.....	648, 792, 804, 815	P3PCR.....	217, 800, 811, 821
INTSELR0.....	590, 794, 806, 817	P4DDR.....	223, 800, 811, 821
INTSTR0.....	649, 792, 804, 815	P4DR.....	223, 800, 811, 821
INTSTR1.....	652, 792, 804, 815	P5DDR.....	228, 800, 811, 821
IOMCR.....	653, 792, 804, 815	P5DR.....	229, 800, 811, 821
ISCR16H.....	79, 797, 808, 819	P6DDR.....	233, 800, 811, 821
ISCR16L.....	79, 797, 808, 819	P6DR.....	233, 800, 811, 821
ISCRH.....	80, 796, 808, 818	P7PIN.....	244, 800, 811, 821
ISCR.....	80, 796, 808, 818	P8DDR.....	245, 800, 811, 821
ISR.....	82, 796, 808, 818	P8DR.....	246, 800, 811, 821
ISR16.....	82, 797, 808, 819	P9DDR.....	250, 800, 811, 821
ISSR.....	259, 797, 808, 819	P9DR.....	251, 800, 811, 821
ISSR16.....	258, 797, 808, 819	PADDR.....	254, 799, 811, 821
KBCOMP.....	703, 796, 808	PAODR.....	254, 799, 811, 821
KMIMR6.....	83, 802, 813, 823	PAPIN.....	255, 799, 811, 821

PCSR .....	269, 279, 797, 809, 819	TCONRI.....	348, 803, 814, 823
PTCNT0 .....	260, 797, 808, 819	TCONRO.....	352, 803, 814, 823
PTTER0.....	564, 794, 806, 816	TCONRS.....	354, 803, 814, 823
PWDPRA.....	266, 801, 812, 822	TCORA.....	320, 800, 812, 821
PWDPRB.....	267, 801, 812, 822	TCORB.....	320, 801, 812, 822
PWDR0 to 15.....	266, 801, 812, 822	TCORC.....	328, 803, 813, 823
PWOERA.....	267, 801, 812, 822	TCR.....	296, 321, 798, 800, 810, 812, 820, 821
PWOERB.....	268, 801, 812, 822	TCSR.....	293, 798, 809, 819
PWSL.....	264, 801, 812, 822	TCSR_0.....	323, 376, 799, 800, 811, 812, 820, 821
RAR.....	170	TCSR_1.....	324, 378, 800, 802, 812, 813, 821, 822
RDR.....	392, 801, 812, 822	TCSR_X.....	326, 802, 813, 823
RFU.....	167	TCSR_Y.....	327, 802, 813, 823
RSPR.....	638, 792, 804, 815	TDR.....	392, 801, 812, 822
RSPRD.....	638	TFFR0.....	576, 794, 806, 817
RSPTYR.....	633, 792, 804, 815	TICR.....	328
RSR.....	392	TICRF.....	329, 802, 813, 823
SAR.....	149, 480, 801, 812, 822	TICRR.....	329, 802, 813, 823
SARX.....	481, 801, 812, 822	TIER.....	292, 798, 809, 819
SBYCR.....	772, 797, 809, 819	TISR.....	329, 803, 813, 823
SCICR.....	412, 796, 808, 818	TMP.....	171
SCIDTER.....	416, 796, 807, 818	TOCR.....	297, 798, 810, 820
SCMR.....	404, 801, 812, 822	TSFR0.....	571, 794, 806, 817
SCR.....	396, 801, 812, 822	TSR.....	392
SDBPR.....	746	UDTRFR.....	599, 793, 805, 816
SDBSR.....	746	UPLLCR.....	595, 794, 806, 817
SDIDR.....	754	UPRTCR.....	601, 793, 805, 816
SDIR.....	744	USBCR0.....	592, 794, 806, 817
SEDGR.....	356, 803, 814, 823	USBCR1.....	594, 793, 805, 816
SEMR.....	412, 796, 807, 818	USBCSR0.....	581, 794, 806, 817
SMR.....	393, 801, 812, 822	USBIER0.....	566, 794, 806, 817
SSR.....	398, 801, 812, 822	USBIER1.....	567, 793, 805, 816
STCR.....	59, 800, 811, 821	USBIFR0.....	568, 794, 806, 817
SUBMSTPBH.....	778, 794, 806, 817	USBIFR1.....	571, 793, 805, 816
SUBMSTPBL.....	778, 794, 806, 817	USBMDCR.....	600, 793, 805, 816
SYSCR.....	57, 800, 811, 821	UTESTR0.....	601, 793, 805, 816
SYSCR2.....	235, 775, 797, 809, 819	UTESTR1.....	601, 793, 805, 816
TBCR.....	636, 792, 804, 815	WAR.....	171
TBNCR.....	636, 792, 804, 815		
TCNT.....	320, 375, 799, 801, 811, 812, 820, 822		

WSCR.....	110, 800, 812, 821	stack pointer (SP).....	26
WSCR2.....	112, 797, 809, 819	Stack Status.....	70
WUEMR3.....	83, 802, 813, 823	Start condition.....	517
Relative Address Commands.....	657	Stop condition.....	517
Repeat Mode.....	158	Subactive Mode.....	787
Reset.....	67	Subsleep Mode.....	786
Reset Exception Handling.....	68	Suspend/Resume.....	618
resolution.....	264, 281	System Control Instructions.....	42
RFU Bus Cycle.....	184, 188	TAP Controller.....	755
RFU Manipulation by MCIF.....	200	Timer Connection.....	345
RFU Manipulation by SCI.....	197	Transfer Rate.....	484
RFU Manipulation by USB.....	193	Trap Instruction Exception Handling.....	69
RFU Response Time.....	189	TRAPA instruction.....	46, 69
Scan Mode.....	706	Universal Serial Bus Interface (USB).....	553
Serial Communication Interface (SCI) ...	387	USB Data FIFO.....	557
Serial Data Reception.....	427, 441, 453	USB Function Core.....	602
Serial Data Transmission.....	425, 438, 450	User Debug Interface (H-UDI).....	741
Serial Formats.....	516	User Program Mode.....	732
SETUP Token.....	604	Valid Strokes.....	124, 138
Shift Instructions.....	38	vector number for the software activation interrupt.....	151
Single Data Block.....	191	VSYNCO Output.....	371
Single Mode.....	705	Wait Control.....	133, 136, 140
Slave address.....	517	Watch Mode.....	785
Slave Receive Operation.....	522	Watchdog Timer (WDT).....	373
Slave Transmit Operation.....	525	Watchdog Timer Mode.....	379
Sleep Mode.....	782		
Smart Card Interface.....	445		
Software Protection.....	737		
Software Standby Mode.....	782		
SPI Mode.....	675		



---

**Renesas 16-Bit Single-Chip Microcomputer  
Hardware Manual  
H8S/2158 Group, H8S/2158 F-ZTAT™**

Publication Date: 1st Edition, September 2001  
Rev.3.00, January 25, 2006

Published by: Sales Strategic Planning Div.  
Renesas Technology Corp.

Edited by: Customer Support Department  
Global Strategic Communication Div.  
Renesas Solutions Corp.

---

©2006. Renesas Technology Corp., All rights reserved. Printed in Japan.

**Renesas Technology Corp.** Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

---



<http://www.renesas.com>

---

## RENESAS SALES OFFICES

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 205, AZIA Center, No.133 Yincheng Rd (n), Pudong District, Shanghai 200120, China  
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

### **Renesas Technology Hong Kong Ltd.**

7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong  
Tel: <852> 2265-6688, Fax: <852> 2730-6071

### **Renesas Technology Taiwan Co., Ltd.**

10th Floor, No.99, Fushing North Road, Taipei, Taiwan  
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

### **Renesas Technology Singapore Pte. Ltd.**

1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632  
Tel: <65> 6213-0200, Fax: <65> 6278-8001

### **Renesas Technology Korea Co., Ltd.**

Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea  
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

### **Renesas Technology Malaysia Sdn. Bhd**

Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia  
Tel: <603> 7955-9390, Fax: <603> 7955-9510

# H8S/2158 Group, H8S/2158 F-ZTAT™ Hardware Manual



Renesas Technology Corp.

2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan