To all our customers

Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: http://www.renesas.com

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

RENESAS
Renesas Technology Corp.

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation 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 Corporation 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 Corporation or a third party.
2. Renesas Technology Corporation 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 Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation 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 Corporation 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 Corporation by various means, including the Renesas Technology Corporation 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 Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation 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 Corporation or an authorized Renesas Technology Corporation 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 Corporation 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 Corporation for further details on these materials or the products contained therein.

Hitachi 16-Bit Single-Chip Microcomputer

H8S/2110B

H8S/2110B F-ZTAT$^{\text{TM}}$
HD64F2110BV

Hardware Manual

ADE-602-298

Rev. 1.0
09/19/02
Hitachi, Ltd.

Cautions

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

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 be 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. Contents
5. Overview
6. 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.

7. List of Registers
8. Electrical Characteristics
9. Appendix
10. Main Revisions and Additions in this Edition (only for revised versions)

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.

11. Index

Preface

The H8S/2110B is a microcomputer (MCU) made up of the H8S/2000 CPU employing Hitachi's original architecture as its core, and the peripheral functions required to configure a system.

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/2000 CPU can handle a 16-Mbyte linear address space.

This LSI is equipped with ROM, RAM, a 14-bit PWM timer (PWMX), a 16-bit free-running timer (FRT), an 8-bit timer (TMR), a watchdog timer (WDT), a serial communication interface (SCI), a keyboard buffer controller, a host interface LPC interface (LPC), an I^2C bus interface (IIC), and I/O ports as on-chip peripheral modules, required for system configuration.

A flash memory (F-ZTAT™*) version is available for this LSI's ROM. This provides flexibility as it can be reprogrammed in no time to cope with all situations from the early stages of mass production to full-scale mass production. This is particularly applicable to application devices with specifications that will most probably change.

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

Target Users: This manual was written for users who will be using the H8S/2110B 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/2110B 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 20, List of Registers.

Rules: Register name: The following notation is used for cases when the same or a similar function, e.g. serial communication interface, 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.hitachisemiconductor.com/

H8S/2110B manuals:

Manual Title	**ADE No.**
H8S/2110B Hardware Manual	This manual
H8S/2600 Series, H8S/2000 Series Programming Manual	ADE-602-083

User's manuals for development tools:

Manual Title	**ADE No.**
H8S, H8/300 Series C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual	ADE-702-247
H8S, H8/300 Series Simulator/Debugger User's Manual	ADE-702-282
H8S, H8/300 Series Hitachi Embedded Workshop, Hitachi Debugging Interface Tutorial	ADE-702-231
Hitachi Embedded Workshop User's Manual	ADE-702-201

Contents

Section 9 16-Bit Free-Running Timer (FRT) 153

Figures

Section 1 Overview

Section 2 CPU

Section 3 MCU Operating Modes

Section 4 Exception Handling

Section 5 Interrupt Controller

Section 8 14-Bit PWM Timer (PWMX)

Section 9 16-Bit Free-Running Timer (FRT)

Section 10 8-Bit Timer (TMR)

Section 11 Watchdog Timer (WDT)

Section 12 Serial Communication Interface (SCI)

Section 13 I^{2}C Bus Interface (IIC)

Section 14 Keyboard Buffer Controller

Section 15 Host Interface LPC Interface (LPC)

Section 17 ROM

Section 18 Clock Pulse Generator

Section 19 Power-Down Modes

Section 21 Electrical Characteristics

Appendix C Package Dimensions

Tables

Section 1 Overview

Section 2 CPU

Section 3 MCU Operating Modes

Section 4 Exception Handling

Section 5 Interrupt Controller

Section 7 I/O Ports

Section 8 14-Bit PWM Timer (PWMX)

Section 9 16-Bit Free-Running Timer (FRT)

Section 10 8-Bit Timer (TMR)

Section 11 Watchdog Timer (WDT)

Section 12 Serial Communication Interface (SCI)

Section 13 I^{2}C Bus Interface (IIC)

Section 14 Keyboard Buffer Controller

Section 15 Host Interface LPC Interface (LPC)

Section 17 ROM

Section 18 Clock Pulse Generator

Section 19 Power-Down Modes

Section 21 Electrical Characteristics

Appendix A I/O Port States in Each Processing State

Section 1 Overview

1.1 Features

- High-speed H8S/2000 central processing unit with an internal 16-bit architecture

  Upward-compatible with H8/300 and H8/300H CPUs on an object level

  Sixteen 16-bit general registers

  65 basic instructions

- Various peripheral function

  14-bit PWM timer (PWMX)

  16-bit free-running timer (FRT)

  8-bit timer (TMR)

  Watchdog timer (WDT)

  Asynchronous or clocked synchronous serial communication interface (SCI)

  I^{2}C bus interface (IIC)

  Keyboard buffer controller

  Host interface LPC interface (LPC)

  Clock pulse generator

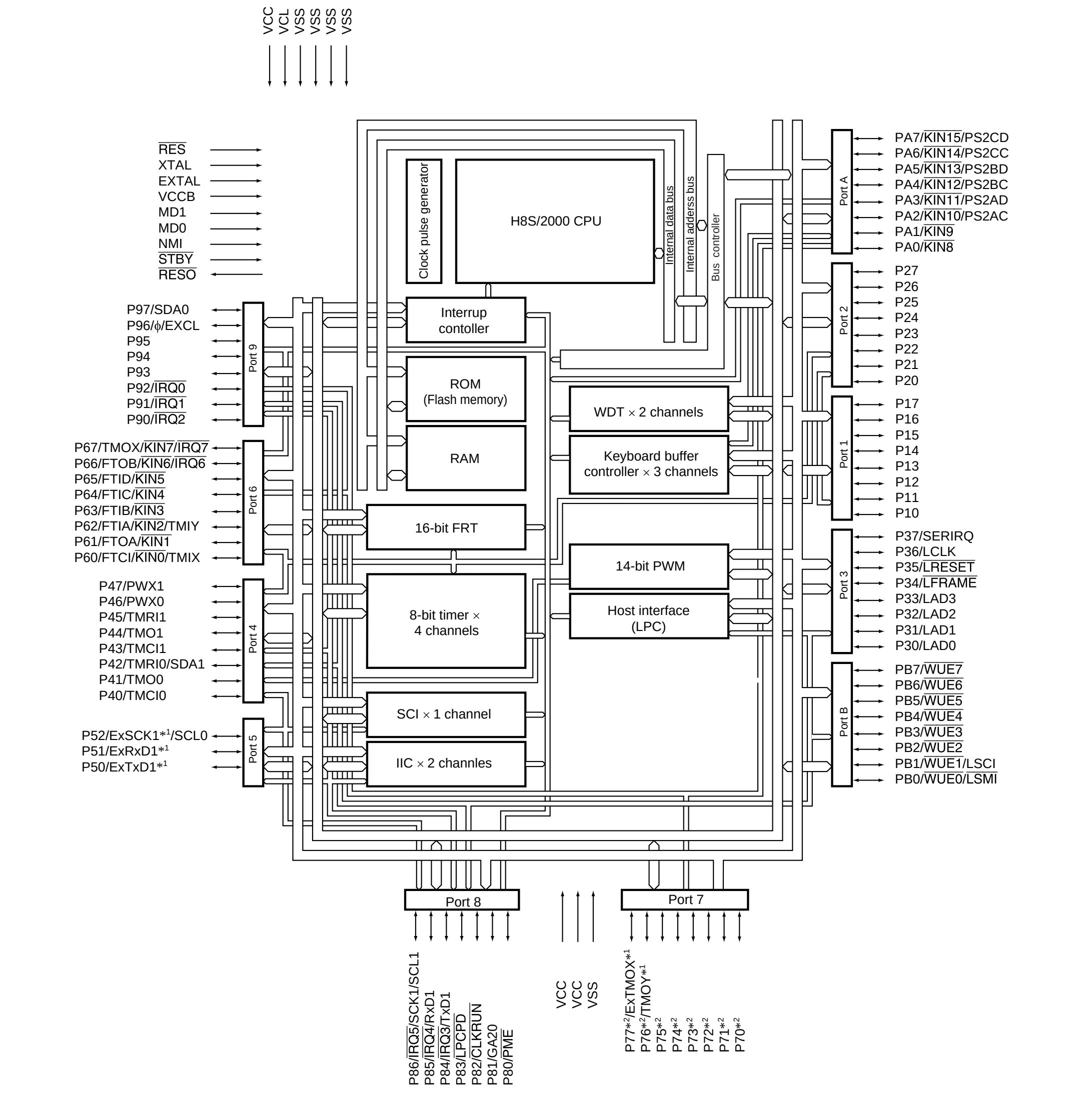
- On-chip memory

ROM	Model	ROM	RAM	Remarks
F-ZTAT Version	HD64F2110BV	64 kbytes	2 kbytes	

- General I/O ports
  I/O pins: 82
- Supports various power-down states
- Compact package

Product	Package	Code	Body Size	Pin Pitch
H8S/2110B	QFP-100B	FP-100B	16.0 × 16.0 mm	0.5 mm
	TQFP-100B	TFP-100B		

1.2 Block Diagram

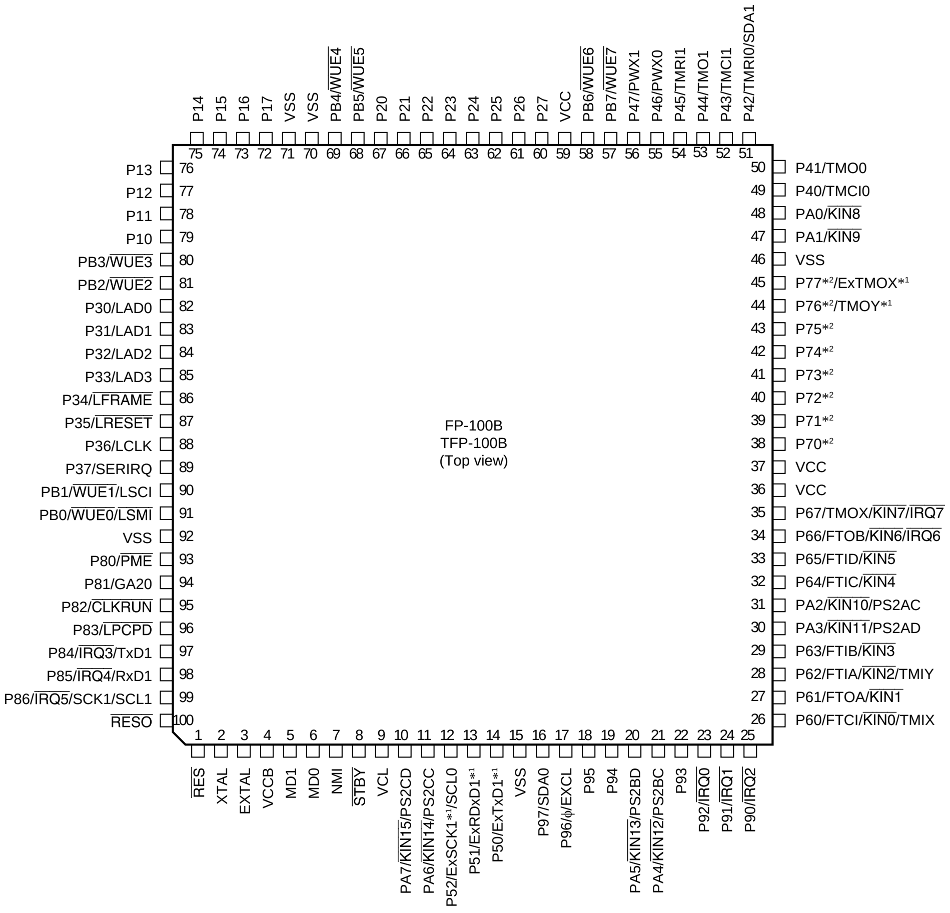


Notes: 1. The program development tool (emulator) does not support this function.
2. The program development tool (emulator) does not support the output.

Figure 1.1 Internal Block Diagram of H8S/2110B

1.3 Pin Arrangement and Functions

1.3.1 Pin Arrangement



Notes: 1. The program development tool (emulator) does not support this function.
2. The program development tool (emulator) does not support the output.

Figure 1.2 Pin Arrangement of H8S/2110B

1.3.2 Pin Functions in Each Operating Mode

Table 1.1 Pin Functions in Each Operating Mode

	Pin Name	
Pin No.	**Single-Chip Modes**	**Flash Memory Programmer Mode**
FP-100B **TFP-100B**	**Mode 2, Mode 3** **(EXPE = 0)**	
1	$\overline{\text{RES}}$	$\overline{\text{RES}}$
2	XTAL	XTAL
3	EXTAL	EXTAL
4	VCCB	VCC
5	MD1	VSS
6	MD0	VSS
7	NMI	FA9
8	$\overline{\text{STBY}}$	VCC
9	VCL	VCC
10 (B)	PA7/$\overline{\text{KIN15}}$/PS2CD	NC
11 (B)	PA6/$\overline{\text{KIN14}}$/PS2CC	NC
12 (N)	P52/ExSCK1*[1]/SCL0	NC
13	P51/ExRxD1*[1]	FA17
14	P50/ExTxD1*[1]	NC
15	VSS	VSS
16 (N)	P97/SDA0	VCC
17	P96/ϕ/EXCL	NC
18	P95	FA16
19	P94	FA15
20 (B)	PA5/$\overline{\text{KIN13}}$/PS2BD	NC
21 (B)	PA4/$\overline{\text{KIN12}}$/PS2BC	NC

	Pin Name	
Pin No.	**Single-Chip Modes**	**Flash Memory Programmer Mode**
FP-100B **TFP-100B**	**Mode 2, Mode 3** **(EXPE = 0)**	
22	P93	$\overline{\mathrm{WE}}$
23	P92/$\overline{\mathrm{IRQ0}}$	VSS
24	P91/$\overline{\mathrm{IRQ1}}$	VCC
25	P90/$\overline{\mathrm{IRQ2}}$	VCC
26	P60/FTCI/$\overline{\mathrm{KIN0}}$/TMIX	NC
27	P61/FTOA/$\overline{\mathrm{KIN1}}$	NC
28	P62/FTIA/$\overline{\mathrm{KIN2}}$/TMIY	NC
29	P63/FTIB/$\overline{\mathrm{KIN3}}$	NC
30 (B)	PA3/$\overline{\mathrm{KIN11}}$/PS2AD	NC
31 (B)	PA2/$\overline{\mathrm{KIN10}}$/PS2AC	NC
32	P64/FTIC/$\overline{\mathrm{KIN4}}$	NC
33	P65/FTID/$\overline{\mathrm{KIN5}}$	NC
34	P66/FTOB/$\overline{\mathrm{KIN6}}$/$\overline{\mathrm{IRQ6}}$	NC
35	P67/TMOX/$\overline{\mathrm{KIN7}}$/$\overline{\mathrm{IRQ7}}$	VSS
36	VCC	VCC
37	VCC	VCC
38	P70*2	NC
39	P71*2	NC
40	P72*2	NC
41	P73*2	NC

	Pin Name	
Pin No.	**Single-Chip Modes**	**Flash Memory Programmer Mode**
FP-100B **TFP-100B**	**Mode 2, Mode 3** **(EXPE = 0)**	
42	P74*[2]	NC
43	P75*[2]	NC
44	P76*[2]/TMOY*[1]	NC
45	P77*[2]/ExTMOX*[1]	NC
46	VSS	VSS
47 (B)	PA1/$\overline{\text{KIN9}}$	NC
48 (B)	PA0/$\overline{\text{KIN8}}$	NC
49	P40/TMCI0	NC
50	P41/TMO0	NC
51 (N)	P42/TMRI0/SDA1	NC
52	P43/TMCI1	NC
53	P44/TMO1	NC
54	P45/TMRI1	NC
55	P46/PWX0	NC
56	P47/PWX1	NC
57	PB7/$\overline{\text{WUE7}}$	NC
58	PB6/$\overline{\text{WUE6}}$	NC
59	VCC	VCC
60	P27	$\overline{\text{CE}}$
61	P26	FA14
62	P25	FA13
63	P24	FA12
64	P23	FA11

	Pin Name	
Pin No.	**Single-Chip Modes**	**Flash Memory Programmer Mode**
FP-100B **TFP-100B**	**Mode 2, Mode 3** **(EXPE = 0)**	
65	P22	FA10
66	P21	$\overline{\text{OE}}$
67	P20	FA8
68	PB5/$\overline{\text{WUE5}}$	NC
69	PB4/$\overline{\text{WUE4}}$	NC
70	VSS	VSS
71	VSS	VSS
72	P17	FA7
73	P16	FA6
74	P15	FA5
75	P14	FA4
76	P13	FA3
77	P12	FA2
78	P11	FA1
79	P10	FA0
80	PB3/$\overline{\text{WUE3}}$	NC
81	PB2/$\overline{\text{WUE2}}$	NC
82	P30/LAD0	FO0
83	P31/LAD1	FO1
84	P32/LAD2	FO2
85	P33/LAD3	FO3
86	P34/$\overline{\text{LFRAME}}$	FO4
87	P35/$\overline{\text{LRESET}}$	FO5
88	P36/LCLK	FO6
89	P37/SERIRQ	FO7
90	PB1/$\overline{\text{WUE1}}$/LSCI	NC
91	PB0/$\overline{\text{WUE0}}$/$\overline{\text{LSMI}}$	NC

	Pin Name	
Pin No.	Single-Chip Modes	Flash Memory Programmer Mode
FP-100B TFP-100B	Mode 2, Mode 3 (EXPE = 0)	
92	VSS	VSS
93	P80/$\overline{\text{PME}}$	NC
94	P81/GA20	NC
95	P82/$\overline{\text{CLKRUN}}$	NC
96	P83/$\overline{\text{LPCPD}}$	NC
97	P84/$\overline{\text{IRQ3}}$/TxD1	NC
98	P85/$\overline{\text{IRQ4}}$/RxD1	NC
99 (N)	P86/$\overline{\text{IRQ5}}$/SCK1/SCL1	NC
100	$\overline{\text{RESO}}$	NC

Notes: The (B) in Pin No. means the VCCB drive and the (N) in Pin No. means the NMOS push-pull/open-drain drive.

*1 The program development tool (emulator) does not support this function.

*2 The program development tool (emulator) does not support the output.

1.3.3 Pin Functions

Table 1.2 Pin Functions

		Pin No.		
Type	**Symbol**	**FP-100B, TFP-100B**	**I/O**	**Name and Function**
Power	VCC	36, 37, 59	Input	Power supply pin. Connect the pin to the system power supply.
	VCL	9	Input	Power supply pin. Connect the pin to VCC.
	VCCB	4	Input	The power supply for the port A input/output buffer.
	VSS	15, 46, 70, 71, 92	Input	Ground pin. Connect to the system power supply (0 V).
Clock	XTAL	2	Input	Pins for connection to crystal resonators. The EXTAL pin can also input an external clock. See section 18, Clock Pulse Generator, for typical connection diagrams.
	EXTAL	3	Input	
	ϕ	17	Output	Supplies the system clock to external devices.
	EXCL	17	Input	Input a 32.768 kHz external subclock.
Operating mode control	MD1 MD0	5 6	Input	These pins set the operating mode. These pins should not be changed while the MCU is operating.
System control	$\overline{\text{RES}}$	1	Input	Reset pin. When this pin becomes low, the chip is reset.
	$\overline{\text{RESO}}$	100	Output	Outputs a reset signal to external device.
	$\overline{\text{STBY}}$	8	Input	When this pin is driven low, a transition is made to hardware standby mode.
Interrupt signals	NMI	7	Input	Input pin for a nonmaskable interrupt request.
	$\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$	23 to 25, 97 to 99, 34, 35	Input	These pins request a maskable interrupt.
16-bit free-running timer (FRT)	FTCI	26	Input	The counter clock input pin.
	FTOA	27	Output	The output compare A output pin.
	FTOB	34	Output	The output compare B output pin.
	FTIA	28	Input	The input capture A input pin.

Type	Symbol	Pin No. FP-100B, TFP-100B	I/O	Name and Function
16-bit free-running timer (FRT)	FTIB	29	Input	The input capture B input pin.
	FTIC	32	Input	The input capture C input pin.
	FTID	33	Input	The input capture D input pin.
8-bit timer (TMR_0, TMR_1, TMR_X, TMR_Y)	TMO0 TMO1 TMOX ExTMOX*[1] TMOY*[1]	50 53 35 45 44	Output	The waveform output pins for the output compare function.
	TMCI0 TMCI1	49 52	Input	Input pins for the external clock input to counters.
	TMRI0 TMRI1	51 54	Input	The counter reset input pins.
8-bit timer (TMR_X, TMR_Y)	TMIX TMIY	26 28	Input	The counter event input and counter reset input pins.
14-bit PWM timer (PWMX)	PWX0 PWX1	55 56	Output	PWM D/A pulse output pins.
Serial communication interface (SCI_1)	ExTxD1*[1] TxD1	14 97	Output	Transmit data output pins.
	ExRxD1*[1] RxD1	13 98	Input	Receive data input pins.
	ExSCK1*[1] SCK1	12 99	Input/ Output	Clock input/output pins. The output type is NMOS push-pull.
Keyboard buffer controller	PS2AC PS2BC PS2CC	31 21 11	Input/ Output	Keyboard buffer controller synchronization clock input/output pins.
	PS2AD PS2BD PS2CD	30 20 10	Input/ Output	Keyboard buffer controller data input/output pins.

		Pin No.		
Type	**Symbol**	**FP-100B, TFP-100B**	**I/O**	**Name and Function**
Host interface (LPC)	LAD3 to LAD0	85 to 82	Input/ Output	LPC command, address, and data input/output pins.
	$\overline{\text{LFRAME}}$	86	Input	Input pin that indicates the start of an LPC cycle or forced termination of an abnormal LPC cycle.
	$\overline{\text{LRESET}}$	87	Input	Input pin that indicates an LPC reset.
	LCLK	88	Input	The LPC clock input pin.
	SERIRQ	89	Input/ Output	Input/output pin for LPC serialized host interrupts (HIRQ1, SMI, HIRQ6, HIRQ9 to HIRQ12).
	LSCI, $\overline{\text{LSMI}}$, $\overline{\text{PME}}$	90, 91, 93	Input/ Output	LPC auxiliary output pins. Functionally, they are general I/O ports.
	GA20	94	Input/ Output	A20 gate control signal output pin. Output state monitoring input is possible.
	$\overline{\text{CLKRUN}}$	95	Input/ Output	Input/output pin that requests the start of LCLK operation when LCLK is stopped.
	$\overline{\text{LPCPD}}$	96	Input	Input pin that controls LPC module shutdown.
Keyboard buffer controller	$\overline{\text{KIN0}}$ to $\overline{\text{KIN15}}$	26 to 29, 32 to 35, 48, 47, 31, 30, 21, 20, 11, 10	Input	Matrix keyboard input pins. $\overline{\text{KIN0}}$ to $\overline{\text{KIN15}}$ are used as key-scan inputs, and P10 to P17 and P20 to P27 are used as key-scan outputs. This allows a maximum 16-output × 16-input, 256-key matrix to be configured.
	$\overline{\text{WUE0}}$ to $\overline{\text{WUE7}}$	91, 90, 81, 80, 69, 68, 58, 57	Input	Wakeup event input pins. These pins allow the same kind of wakeup as key-wakeup from various sources.
I^2C bus interface (IIC)	SCL0 SCL1	12 99	Input/ Output	I^2C clock I/O pins. The output type is NMOS open-drain output.
	SDA0 SDA1	16 51	Input/ Output	I^2C data I/O pins. The output type is NMOS open-drain output.

		Pin No.		
Type	Symbol	FP-100B, TFP-100B	I/O	Name and Function
I/O ports	P17 to P10	72 to 79	Input/ Output	Eight input/output pins.
	P27 to P20	60 to 67	Input/ Output	Eight input/output pins.
	P37 to P30	89 to 82	Input/ Output	Eight input/output pins.
	P47 to P40	56 to 49	Input/ Output	Eight input/output pins. (The output type of P42 is NMOS push-pull.)
	P52 to P50	12 to 14	Input/ Output	Three input/output pins. (The output type of P52 is NMOS push-pull.)
	P67 to P60	35 to 32 29 to 26	Input/ Output	Eight input/output pins.
	P77 to P70	45 to 38	Input/ Output*[2]	Eight input/output pins.*[2]
	P86 to P80	99 to 93	Input/ Output	Seven input/output pins. (The output type of P86 is NMOS push-pull.)
	P97 to P90	16 to 19 22 to 25	Input/ Output	Eight input/output pins. (The output type of P97 is NMOS push-pull.)
	PA7 to PA0	10, 11, 20, 21, 30, 31, 47, 48	Input/ Output	Eight input/output pins.
	PB7 to PB0	57, 58, 68, 69, 80, 81, 90, 91	Input/ Output	Eight input/output pins.

Notes: 1. The program development tool (emulator) does not support this function.

2. The program development tool (emulator) does not support the output.

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, refer to 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 × 8-bit register-register multiply: 12 states (MULXU.B), 13 states (MULXS.B)
  16 ÷ 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

		Execution States	
Instruction	**Mnemonic**	**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.
- Expanded 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 on 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.

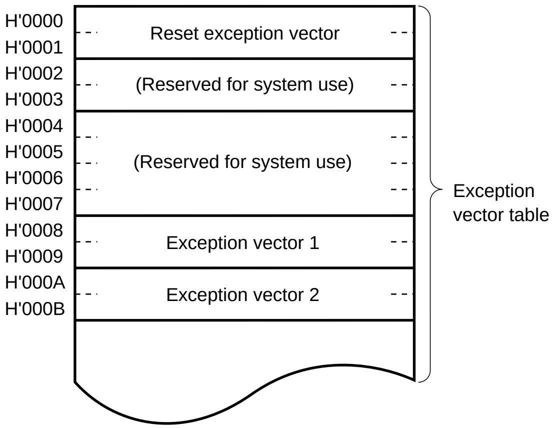


Figure 2.1 Exception Vector Table (Normal Mode)

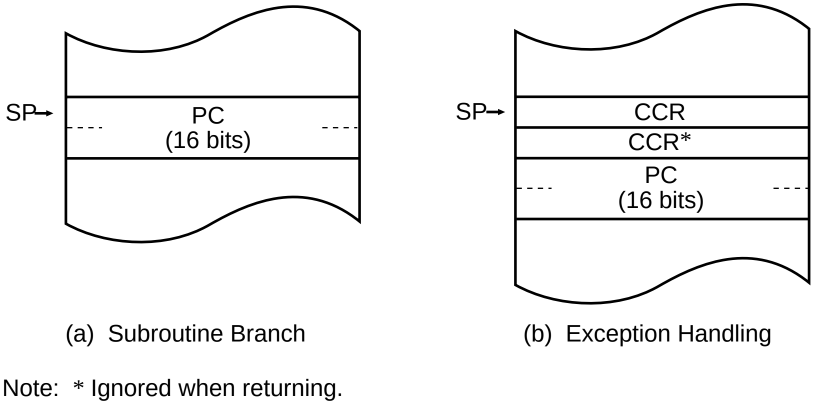


Figure 2.2 Stack Structure in Normal Mode

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 on the exception vector table, see section 4, Exception Handling.

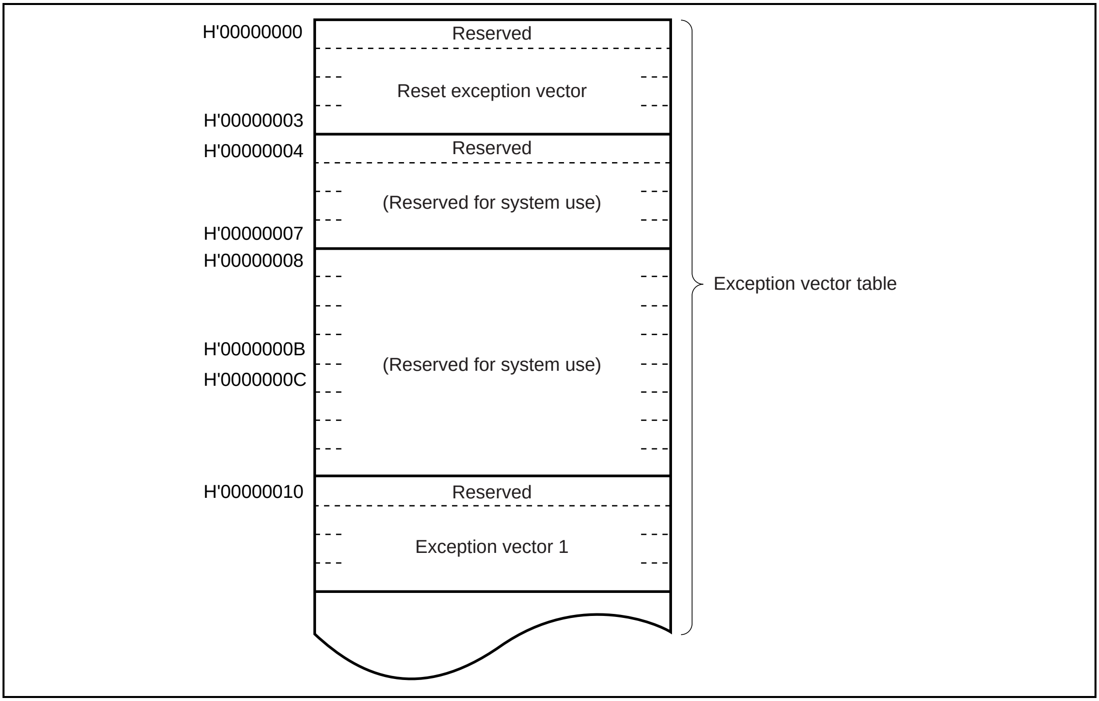


Figure 2.3 Exception Vector Table (Advanced Mode)

  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.

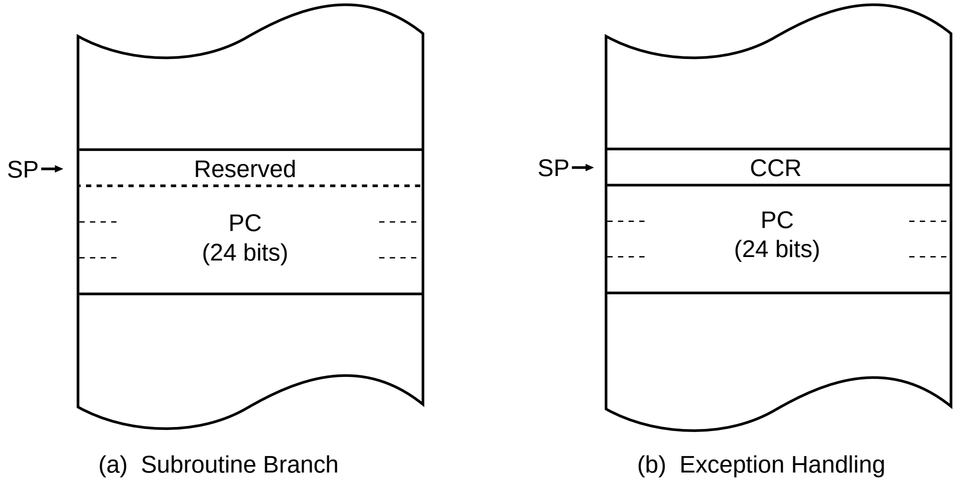


Figure 2.4 Stack Structure in Advanced Mode

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, refer to section 3, MCU Operating Modes.

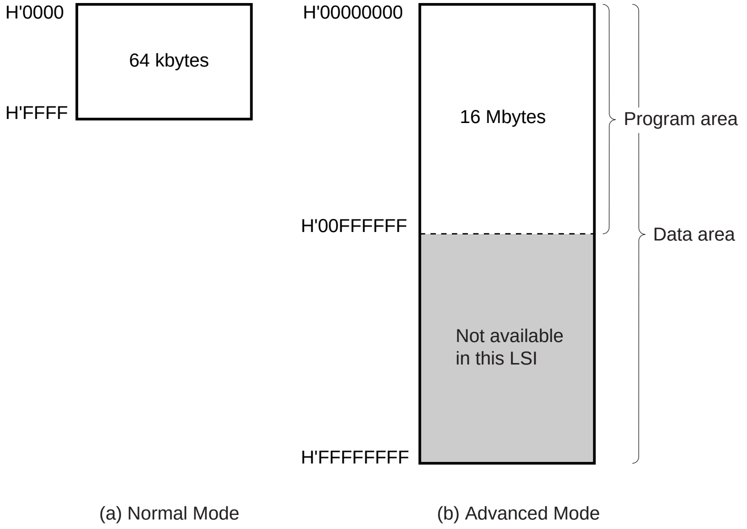
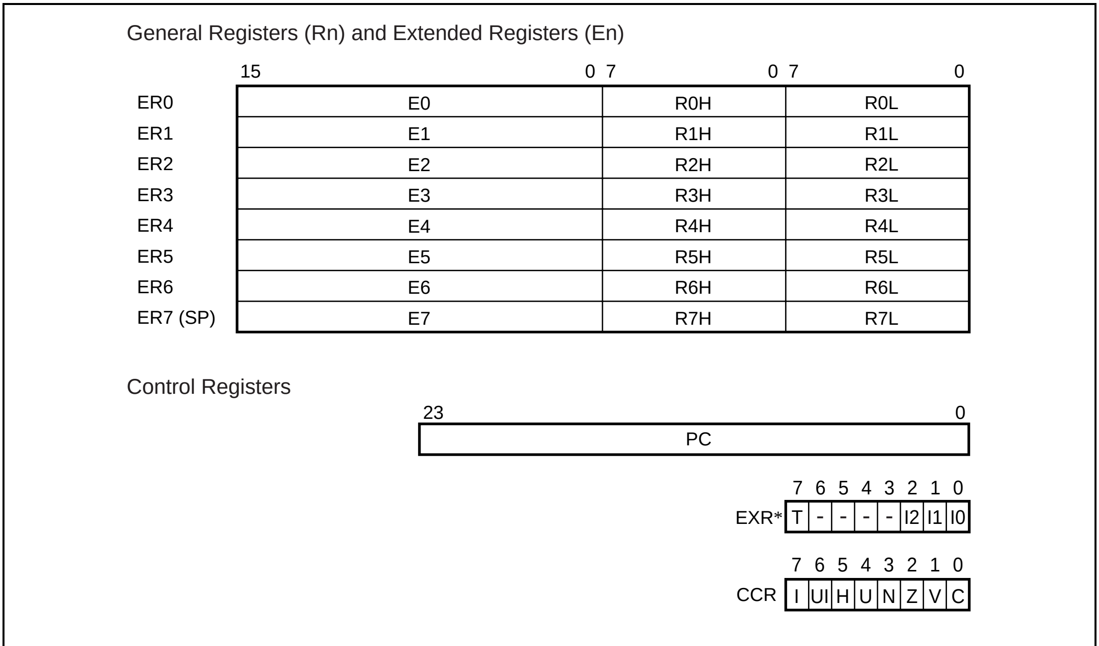


Figure 2.5 Memory Map

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).



Legend

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

Note: * Does not affect operation in this LSI.

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.

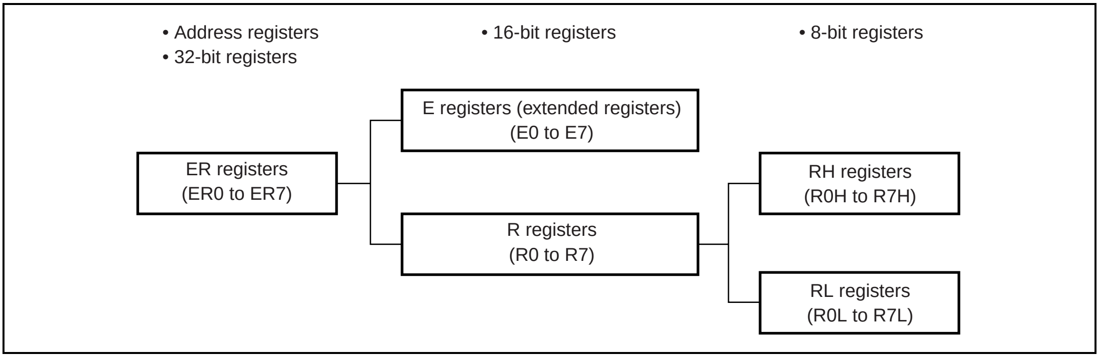


Figure 2.7 Usage of General Registers

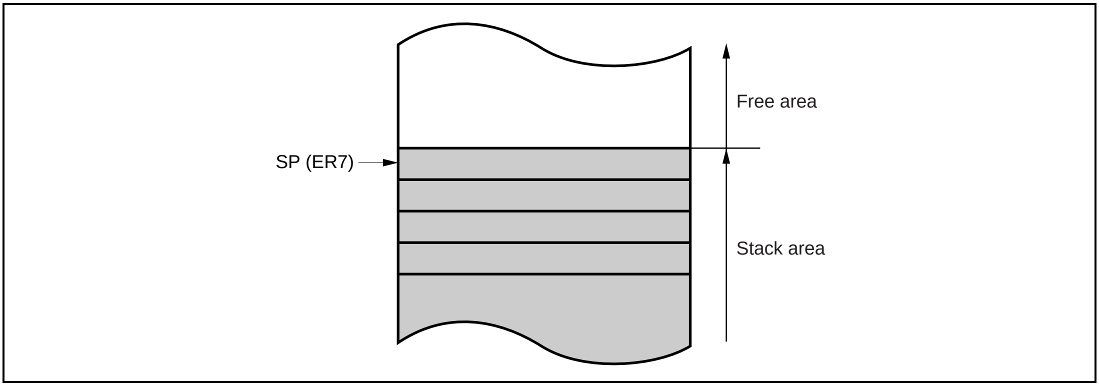


Figure 2.8 Stack

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 to 0	I2	1	R/W	Interrupt Mask Bits 2 to 0
	I1	1	R/W	Do not affect operation in this LSI.
	I0	1	R/W	

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	Interrupt Mask Bit 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, refer to section 5, Interrupt Controller.
6	UI	Undefined	R/W	User Bit or Interrupt Mask Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
5	H	Undefined	R/W	Half-Carry Flag When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, 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.
4	U	Undefined	R/W	User Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag Stores the value of the most significant bit of data as a sign bit.
2	Z	Undefined	R/W	Zero Flag Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.
0	C	Undefined	R/W	Carry Flag Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by • Add instructions, to indicate a carry • Subtract instructions, to indicate a borrow • Shift and rotate instructions, to indicate a carry The carry flag is also used as a bit accumulator by bit manipulation instructions.

2.4.5 Initial Register Values

The program counter (PC) among CPU internal registers is initialized when reset exception handling loads a start address from a vector table. The trace (T) bit in EXR is cleared to 0, and the interrupt mask (I) bits in CCR and EXR are set 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, …, 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.

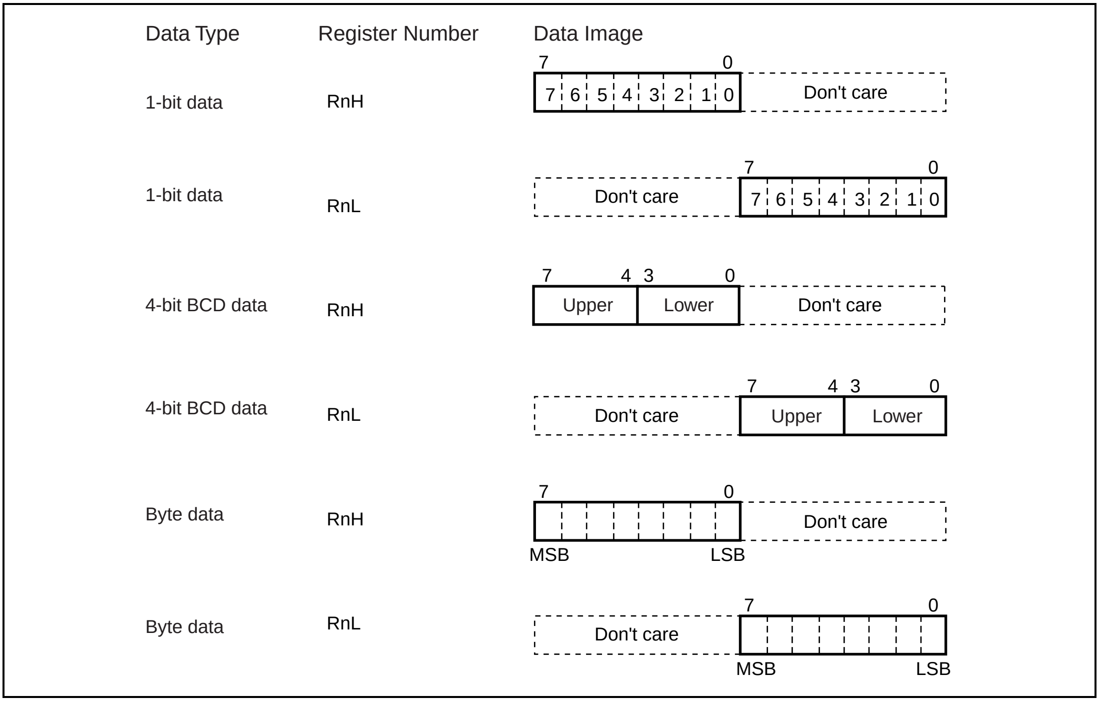
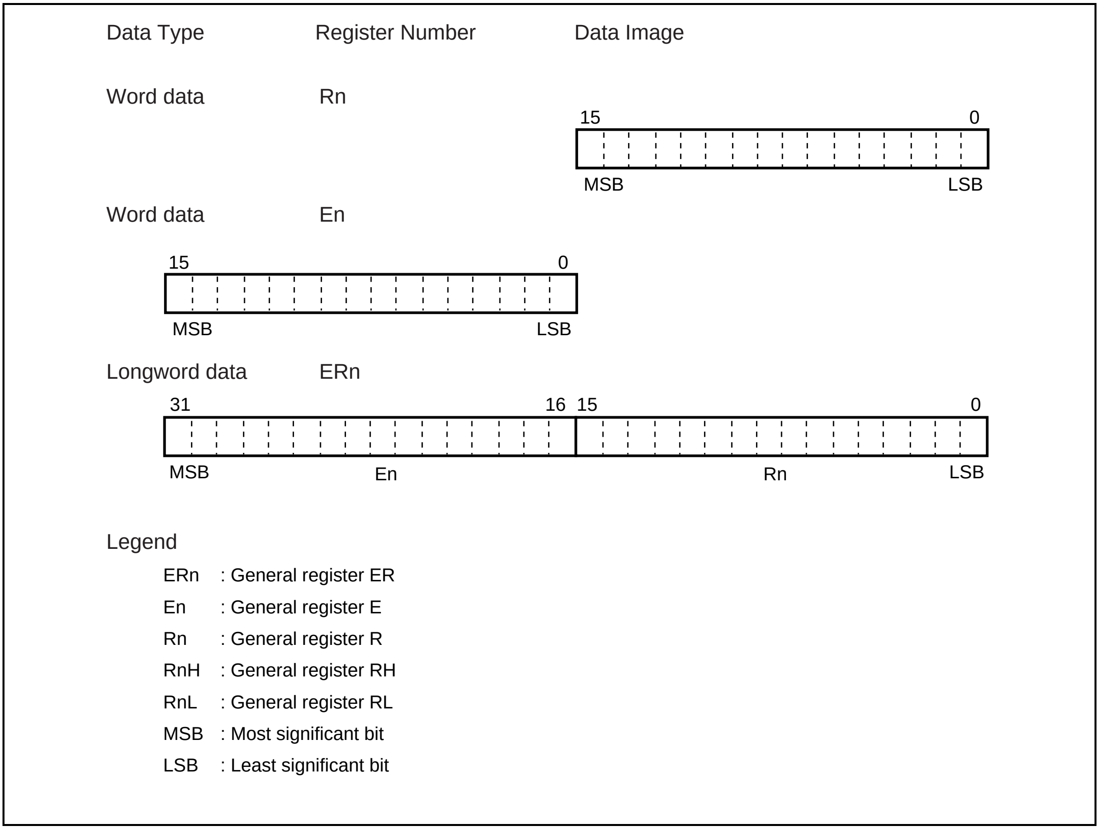


Figure 2.9 General Register Data Formats (1)



Legend

ERn : General register ER

En : General register E

Rn : General register R

RnH : General register RH

RnL : General register RL

MSB : Most significant bit

LSB : Least significant bit

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.

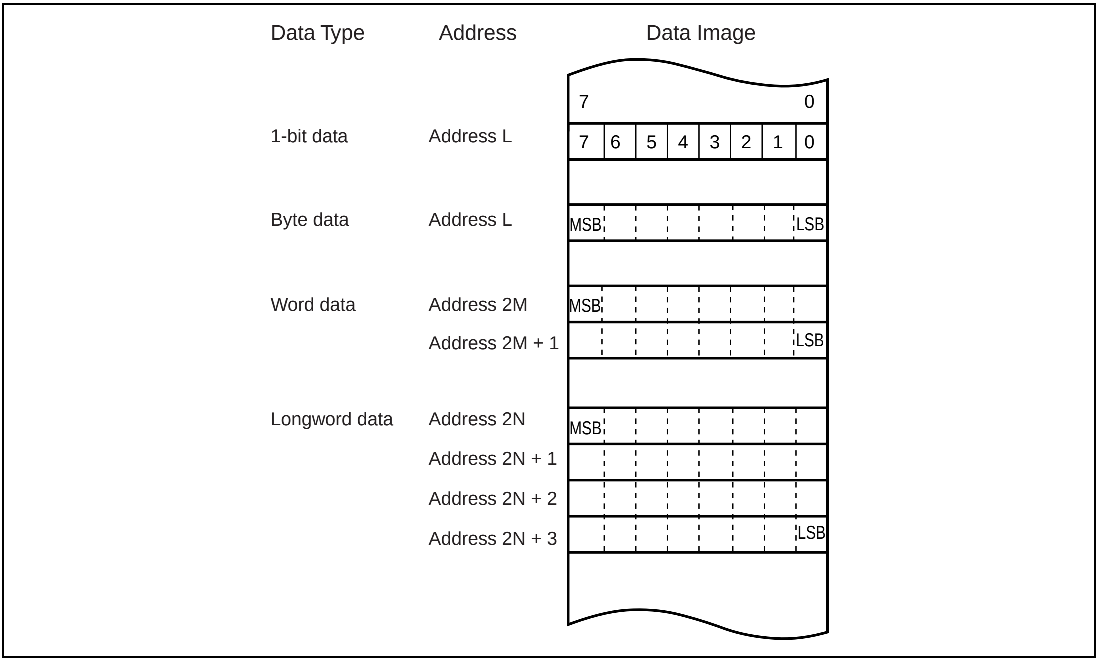


Figure 2.10 Memory Data Formats

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*[1], PUSH*[1]	W/L	
	LDM*[5], STM*[5]	L	
	MOVFPE*[3], MOVTPE*[3]	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*[4]	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_{CC}*[2], JMP, BSR, JSR, RTS	—	5
System control	TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	—	9
Block data transfer	EEPMOV	—	1
			Total: 65

Notes: B: Byte size; W: Word size; L: Longword size.

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. B_{CC} is the general name for conditional branch instructions.
3. Cannot be used in this LSI.
4. When using the TAS instruction, use registers ER0, ER1, ER4, and ER5.
5. ER7 is not used as the register that can be saved (STM)/restored (LDM) when using STM/LDM instruction, because ER7 is the stack pointer.

2.6.1 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

Instruction	Size*[1]	Function
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.
MOVFPE	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*[2]	L	@SP+ → Rn (register list) Pops two or more general registers from the stack.
STM*[2]	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. ER7 is not used as the register that can be saved (STM)/restored (LDM) when using STM/LDM instruction, because ER7 is the stack pointer.

Table 2.4 Arithmetic Operations Instructions (1)

Instruction	Size*	Function
ADD SUB	B/W/L	Rd ± Rs → Rd, Rd ± #IMM → Rd 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 SUBX	B	Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd Performs addition or subtraction with carry on data in two general registers, or on immediate data and data in a general register.
INC DEC	B/W/L	Rd ± 1 → Rd, Rd ± 2 → Rd 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 SUBS	L	Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd Adds or subtracts the value 1, 2, or 4 to or from data in a 32-bit register.
DAA DAS	B	Rd (decimal adjust) → Rd 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 × Rs → Rd Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits.
MULXS	B/W	Rd × Rs → Rd Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits.
DIVXU	B/W	Rd ÷ Rs → Rd Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 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)

Instruction	Size*[1]	Function
DIVXS	B/W	Rd ÷ Rs → Rd Performs signed division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 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 → Rd Takes the two's complement (arithmetic complement) of data in a general register.
EXTU	W/L	Rd (zero extension) → 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) → 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*[2]	B	@ERd – 0, 1 → (<bit 7> of @ERd) Tests memory contents, and sets the most significant bit (bit 7) to 1.

Notes: 1. Size refers to the operand size.
B: Byte
W: Word
L: Longword
2. When using the TAS instruction, use registers ER0, ER1, ER4 and ER5.

Table 2.5 Logic Operations Instructions

Instruction	Size*	Function
AND	B/W/L	Rd ∧ Rs → Rd, Rd ∧ #IMM → Rd Performs a logical AND operation on a general register and another general register or immediate data.
OR	B/W/L	Rd ∨ Rs → Rd, Rd ∨ #IMM → Rd Performs a logical OR operation on a general register and another general register or immediate data.
XOR	B/W/L	Rd ⊕ Rs → Rd, Rd ⊕ #IMM → Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data.
NOT	B/W/L	~ Rd → 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

Instruction	Size*	Function
SHAL SHAR	B/W/L	Rd (shift) → Rd Performs an arithmetic shift on data in a general register. 1-bit or 2 bit shift is possible.
SHLL SHLR	B/W/L	Rd (shift) → Rd Performs a logical shift on data in a general register. 1-bit or 2 bit shift is possible.
ROTL ROTR	B/W/L	Rd (rotate) → Rd Rotates data in a general register. 1-bit or 2 bit rotation is possible.
ROTXL ROTXR	B/W/L	Rd (rotate) → Rd 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)

Instruction	Size*	Function
BSET	B	1 → (<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 → (<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	~ (<bit-No.> of <EAd>) → (<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	~ (<bit-No.> of <EAd>) → 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 ∧ (<bit-No.> of <EAd>) → 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 ∧ (<bit-No.> of <EAd>) → 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 ∨ (<bit-No.> of <EAd>) → 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 ∨ (~ <bit-No.> of <EAd>) → 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)

Instruction	Size*	Function
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → 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 ⊕ ~ (<bit-No.> of <EAd>) → 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	(<bit-No.> of <EAd>) → C Transfers a specified bit in a general register or memory operand to the carry flag.
BILD	B	~ (<bit-No.> of <EAd>) → 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 → (<bit-No.> of <EAd>) Transfers the carry flag value to a specified bit in a general register or memory operand.
BIST	B	~ C → (<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

<table>
<tr><th>Instruction</th><th>Size</th><th>Function</th></tr>
<tr><td>Bcc</td><td>—</td><td>Branches to a specified address if a specified condition is true. The branching conditions are listed below.
<table>
<tr><th>Mnemonic</th><th>Description</th><th>Condition</th></tr>
<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>$C \vee Z = 0$</td></tr>
<tr><td>BLS</td><td>Low or same</td><td>$C \vee Z = 1$</td></tr>
<tr><td>BCC (BHS)</td><td>Carry clear
(high or same)</td><td>$C = 0$</td></tr>
<tr><td>BCS (BLO)</td><td>Carry set (low)</td><td>$C = 1$</td></tr>
<tr><td>BNE</td><td>Not equal</td><td>$Z = 0$</td></tr>
<tr><td>BEQ</td><td>Equal</td><td>$Z = 1$</td></tr>
<tr><td>BVC</td><td>Overflow clear</td><td>$V = 0$</td></tr>
<tr><td>BVS</td><td>Overflow set</td><td>$V = 1$</td></tr>
<tr><td>BPL</td><td>Plus</td><td>$N = 0$</td></tr>
<tr><td>BMI</td><td>Minus</td><td>$N = 1$</td></tr>
<tr><td>BGE</td><td>Greater or equal</td><td>$N \oplus V = 0$</td></tr>
<tr><td>BLT</td><td>Less than</td><td>$N \oplus V = 1$</td></tr>
<tr><td>BGT</td><td>Greater than</td><td>$Z \vee (N \oplus V) = 0$</td></tr>
<tr><td>BLE</td><td>Less or equal</td><td>$Z \vee (N \oplus V) = 1$</td></tr>
</table>
</td></tr>
<tr><td>JMP</td><td>—</td><td>Branches unconditionally to a specified address.</td></tr>
<tr><td>BSR</td><td>—</td><td>Branches to a subroutine at a specified address</td></tr>
<tr><td>JSR</td><td>—</td><td>Branches to a subroutine at a specified address</td></tr>
<tr><td>RTS</td><td>—</td><td>Returns from a subroutine</td></tr>
</table>

Table 2.9 System Control Instructions

Instruction	Size*	Function
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

Instruction	Size	Function

EEPMOV.B	—	if R4L ≠ 0 then
		    Repeat @ER5 + → @ER6+
		        R4L–1 → R4L
		    Until R4L = 0
		else next;

EEPMOV.W	—	if R4 ≠ 0 then
		    Repeat @ER5 + → @ER6+
		        R4–1 → 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.

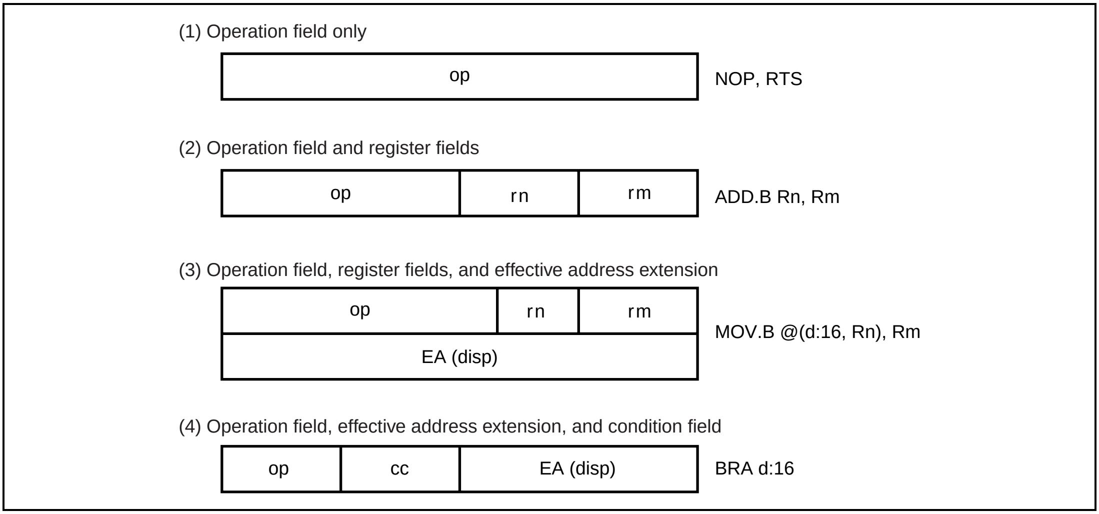

(1) Operation field only
op
NOP, RTS
(2) Operation field and register fields
op
rn
rm
ADD.B Rn, Rm
(3) Operation field, register fields, and effective address extension
op
rn
rm
EA (disp)
MOV.B @(d:16, Rn), Rm
(4) Operation field, effective address extension, and condition field
op
cc
EA (disp)
BRA d:16

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 an 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, refer to 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.)

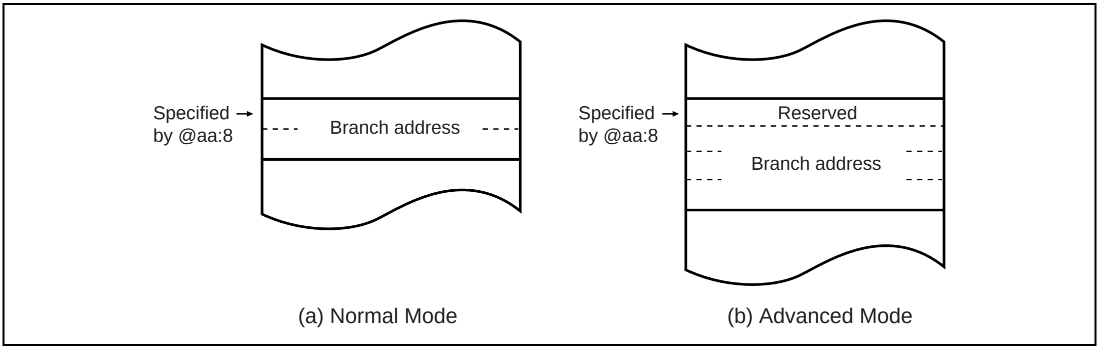
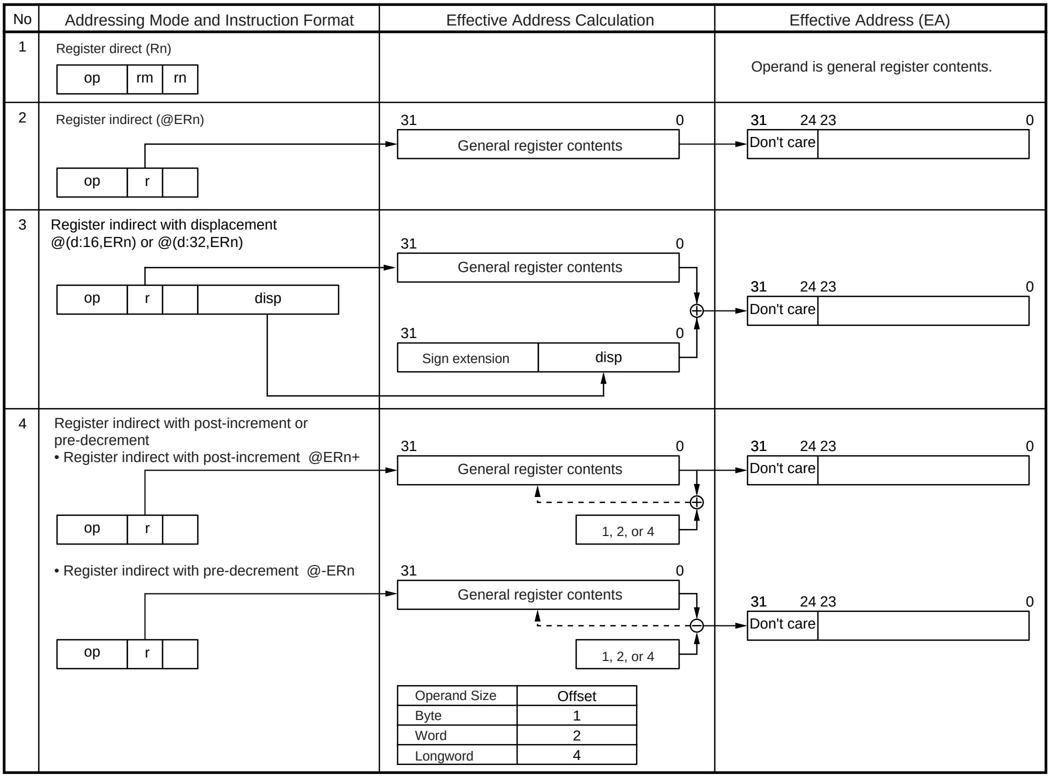


Figure 2.12 Branch Address Specification in Memory Indirect Addressing Mode

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.

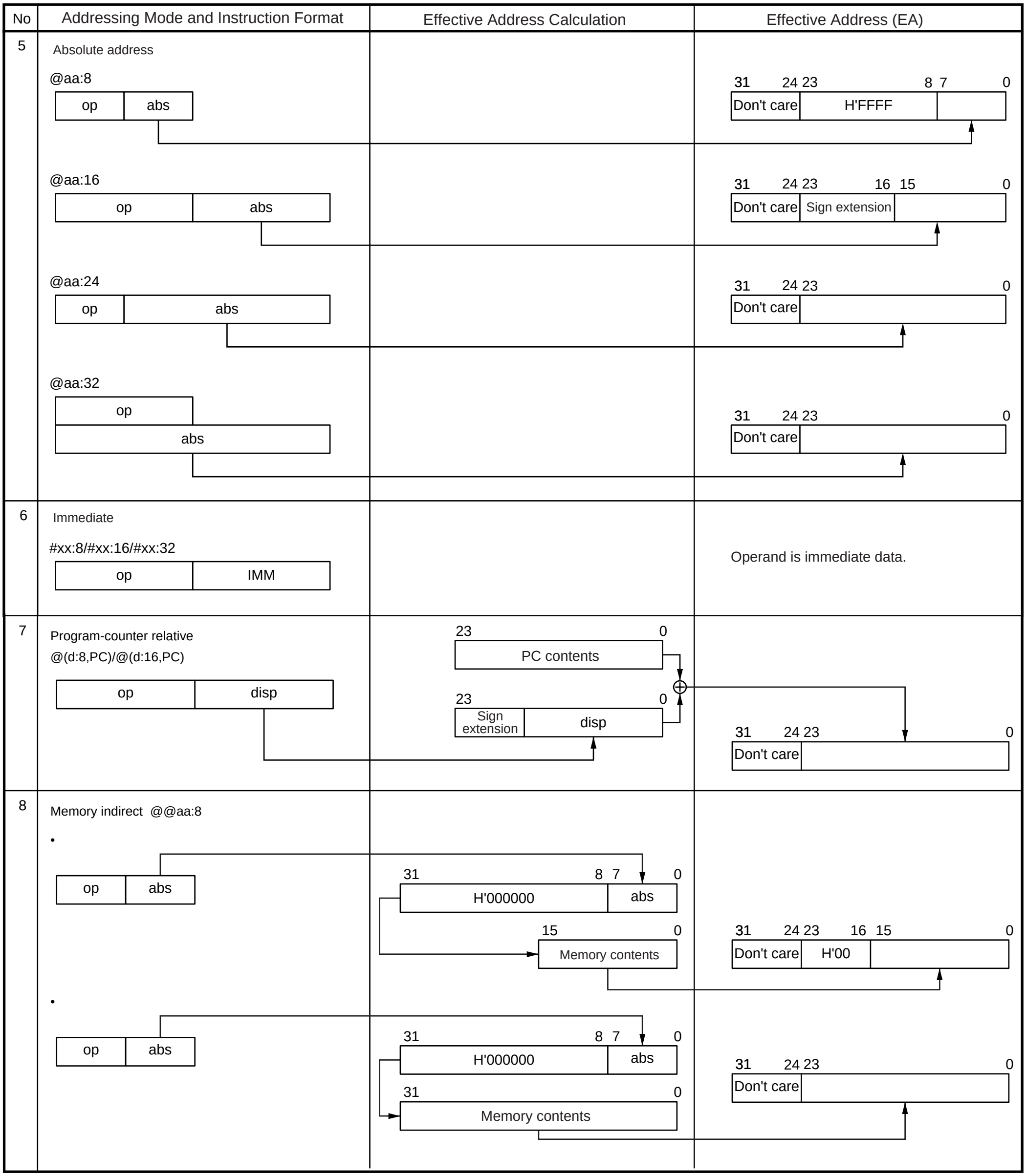
Table 2.13 Effective Address Calculation (1)



No
Addressing Mode and Instruction Format
Effective Address Calculation
Effective Address (EA)
1
Register direct (Rn)
op
rm
rn
Operand is general register contents.
2
Register indirect (@ERn)
op
r
31
0
General register contents
31
24 23
0
Don't care
3
Register indirect with displacement
@(d:16,ERn) or @(d:32,ERn)
op
r
disp
31
0
General register contents
31
0
Sign extension
disp
31
24 23
0
Don't care
4
Register indirect with post-increment or pre-decrement
• Register indirect with post-increment @ERn+
op
r
31
0
General register contents
1, 2, or 4
31
24 23
0
Don't care
• Register indirect with pre-decrement @-ERn
op
r
31
0
General register contents
1, 2, or 4
31
24 23
0
Don't care

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 op | abs		31 24 23 8 7 0 Don't care | H'FFFF |
	@aa:16 op | abs		31 24 23 16 15 0 Don't care | Sign extension |
	@aa:24 op | abs		31 24 23 0 Don't care |
	@aa:32 op abs		31 24 23 0 Don't care |
6	Immediate #xx:8/#xx:16/#xx:32 op | IMM		Operand is immediate data.
7	Program-counter relative @(d:8,PC)/@(d:16,PC) op | disp	23 0 PC contents 23 0 Sign extension | disp	31 24 23 0 Don't care |
8	Memory indirect @@aa:8 • op | abs	31 8 7 0 H'000000 | abs 15 0 Memory contents	31 24 23 16 15 0 Don't care | H'00 |
	• op | abs	31 8 7 0 H'000000 | abs 31 0 Memory contents	31 24 23 0 Don't care |

2.8 Processing States

The H8S/2000 CPU has four main processing states: the reset state, exception handling state, program execution 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, refer to section 4, Exception Handling.

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

- Exception-handling state

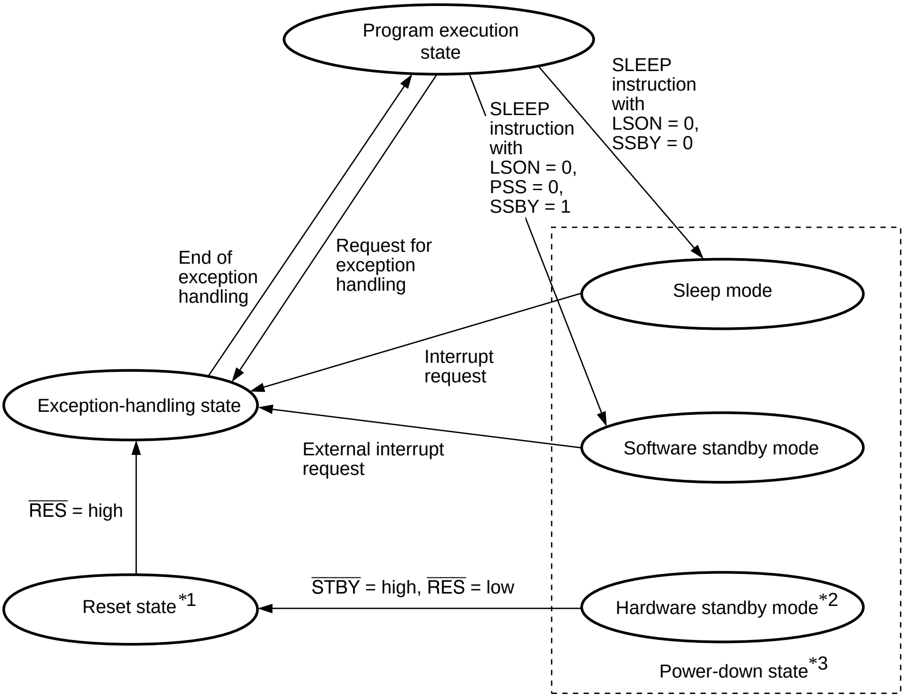
  The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to an exception source, such as, a reset, trace, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception vector table and branches to that address. For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

- Program execution state

  In this state the CPU executes program instructions in sequence.

- 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, refer to section 19, Power-Down Modes.


Program execution state
SLEEP instruction with LSON = 0, SSBY = 0
SLEEP instruction with LSON = 0, PSS = 0, SSBY = 1
End of exception handling
Request for exception handling
Sleep mode
Interrupt request
Exception-handling state
Software standby mode
External interrupt request
$\overline{\text{RES}}$ = high
$\overline{\text{STBY}}$ = high, $\overline{\text{RES}}$ = low
Reset state*1
Hardware standby mode*2
Power-down state*3

Notes: 1. From any state except hardware standby mode, a transition to the reset state occurs whenever $\overline{\text{RES}}$ goes low. A transition can also be made to the reset state when the watchdog timer overflows.

2. From any state, a transition to hardware standby mode occurs when $\overline{\text{STBY}}$ goes low.

3. The power-down state also includes watch mode, subactive mode, subsleep mode, etc. For details, refer to section 19, Power-Down Modes.

Figure 2.13 State Transitions

2.9 Usage Notes

2.9.1 Note on TAS Instruction Usage

When using the TAS instruction, use registers ER0, ER1, ER4 and ER5.

The TAS instruction is not generated by the Hitachi H8S and H8/300 series C/C++ compilers. When the TAS instruction is used as a user-defined intrinsic function, use registers ER0, ER1, ER4 and ER5.

2.9.2 Note on STM/LDM Instruction Usage

ER7 is not used as the register that can be saved (STM)/restored (LDM) when using STM/LDM instruction, because ER7 is the stack pointer. Two, three, or four registers can be saved/restored by one STM/LDM instruction. The following ranges can be specified in the register list.

Two registers: ER0—ER1, ER2—ER3, or ER4—ER5

Three registers: ER0—ER2 or ER4—ER6

Four registers: ER0—ER3

The STM/LDM instruction including ER7 is not generated by the Hitachi H8S and H8/300 series C/C++ compilers.

2.9.3 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:

	P47	P46	P45	P44	P43	P42	P41	P40
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:

	P47	P46	P45	P44	P43	P42	P41	P40
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

Operation:

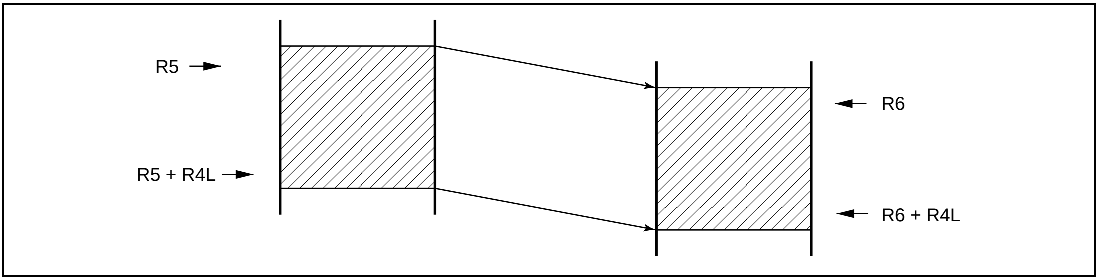
1. When the BCLR instruction is executed, first the CPU reads P4DDR.

   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.

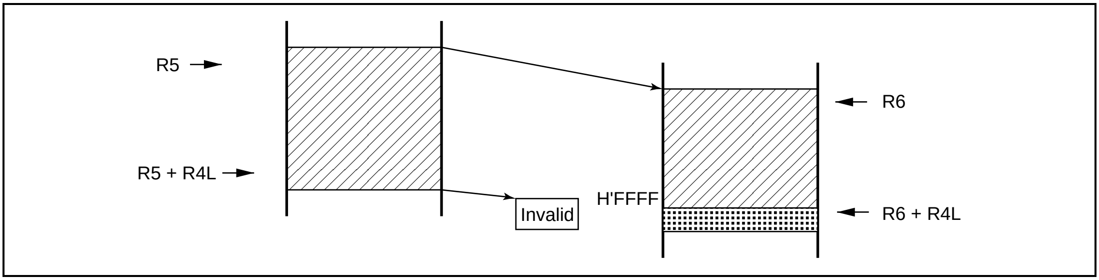
As a result of the BCLR instruction, bit 0 in DDR is set to 0, and P40 becomes an input pin. However, bits 7 and 6 of DDR are modified to 1, therefore P47 and P46 become output pins.

2.9.4 EEPMOV Instruction

1. EEPMOV is a block-transfer instruction and transfers the byte size of data indicated by R4L, which starts from the address indicated by R5, to the address indicated by R6.



2. Set R4L and R6 so that the end address of the destination address (value of R6 + R4L) does not exceed H'FFFF (the value of R6 must not change from H'FFFF to H'0000 during execution).



Section 3 MCU Operating Modes

3.1 MCU Operating Mode Selection

This LSI has two operating modes (modes 2 and 3). The operating mode is determined by the setting of the mode pins (MD1 and MD0). Table 3.1 shows the MCU operating mode selection.

Table 3.1 lists the MCU operating modes.

Table 3.1 MCU Operating Mode Selection

MCU Operating Mode	MD1	MD0	CPU Operating Mode	Description	On-Chip ROM
2	1	0	Advanced	Single-chip mode	Enabled
3		1	Normal	Single-chip mode	

Modes 2 and 3 set the operation in single-chip mode.

Modes 0 and 1 cannot be used in this LSI. 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.

3.2 Register Descriptions

The following registers are related to the operating mode.

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

3.2.1 Mode Control Register (MDCR)

MDCR is used to monitor the current operating mode.

Bit	Bit Name	Initial Value	R/W	Description
7	EXPE	0	R/W	Reserved The initial value should not be changed.
6 to 2	—	All 0	R	Reserved These bits are always read as 0. These bits cannot be modified.
1	MDS1	—*	R	Mode Select 1 and 0
0	MDS0	—*	R	These bits indicate the input levels at mode pins (MD1 and MD0) (the current operating mode). Bits MDS1 and MDS0 correspond to MD1 and MD0, respectively. These bits are read-only bits and they cannot be written to. The mode pin (MD1 and MD0) input levels are latched into these bits when MDCR is read. These latches are canceled by a reset.

Note:* The initial values are determined by the settings of the 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, pin location selection, 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 and 6	—	All 0	R/W	Reserved The initial value should not be changed.
5 4	INTM1 INTM0	0 0	R R/W	These bits select the control mode of the interrupt controller. For details on the interrupt control modes and interrupt control select modes 1 and 0, see section 5.6, Interrupt Control Modes and Interrupt Operation. 00: Interrupt control mode 0 01: Interrupt control mode 1 10: Setting prohibited 11: Setting prohibited
3	XRST	1	R	External Reset This bit indicates the reset source. A reset is caused by an external reset input, or when the watchdog timer overflows. 0: A reset is caused when the watchdog timer overflows. 1: A reset is caused by an external reset.

Bit	Bit Name	Initial Value	R/W	Description
2	NMIEG	0	R/W	NMI Edge Select Selects the valid edge of the NMI interrupt input. 0: An interrupt is requested at the falling edge of NMI input 1: An interrupt is requested at the rising edge of NMI input
1	HIE	0	R/W	Host Interface Enable Controls CPU access to the keyboard matrix interrupt, input pull-up MOS control registers (KMIMR, KMPCR, and KMIMRA), and the 8-bit timer (TMR_X and TMR_Y) registers (TCR_X/TCR_Y, TCSR_X/TCSR_Y, TICRR/TCORA_Y, TICRF/TCORB_Y, TCNT_X/TCNT_Y, TCORC/TISR, TCORA_X, and TCORB_X, TCONRI, and TCONRS). 0: In areas H'(FF)FFF0 to H'(FF)FFF7 and H'(FF)FFFC to H'(FF)FFFF, CPU access to 8-bit timer (TMR_X and TMR_Y) is permitted. 1: In areas H'(FF)FFF0 to H'(FF)FFF7 and H'(FF)FFFC to H'(FF)FFFF, CPU access to keyboard matrix interrupt and input pull-up MOS control registers is permitted.
0	RAME	1	R/W	RAM Enable Enables or disables on-chip RAM. The RAME bit is initialized when the reset state is released. 0: On-chip RAM is disabled 1: On-chip RAM is enabled

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	IICS	0	R/W	I^{2}C Extra Buffer Select Specifies bits 7 to 4 of port A as output buffers similar to SLC and SDA. These pins are used to implement an I^{2}C interface only by software. 0: PA7 to PA4 are normal input/output pins. 1: PA7 to PA4 are input/output pins enabling bus driving.
6	IICX1	0	R/W	I^{2}C 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^{2}C bus mode register (ICMR). For details on the transfer rate, refer to table 13.3.
4	IICE	0	R/W	I^{2}C 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. 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 in power-down state (SBYCR, LPWRCR, MSTPCRH, MSTPCRL), and control registers of on-chip peripheral modules (PCSR, SYSCR2). 0: Registers in power-down state and control registers of 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
0	ICKS0	0	R/W	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, refer to section 10.3.4, Timer Control Register (TCR).

3.3 Operating Mode Descriptions

3.3.1 Mode 2

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

3.3.2 Mode 3

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

3.4 Address Map in Each Operating Mode

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

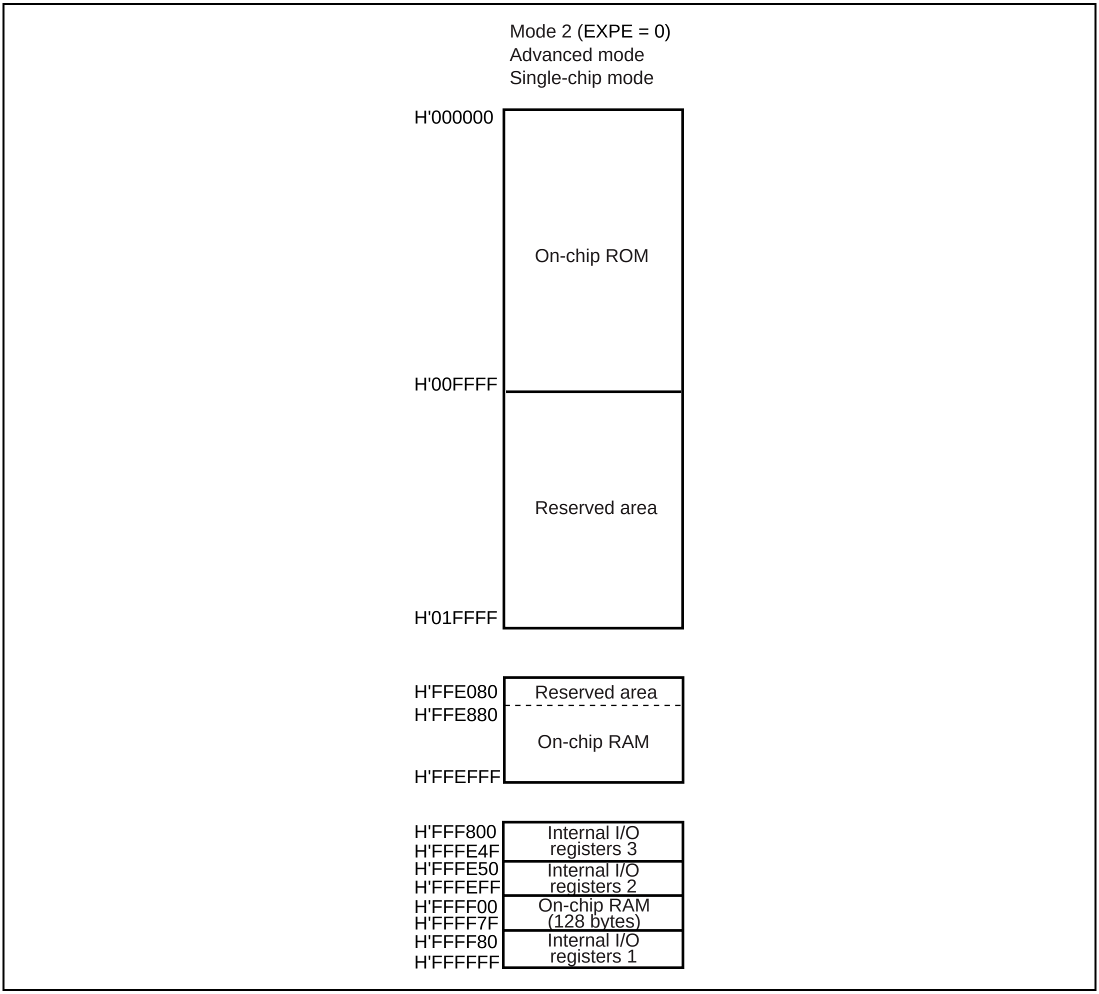


Figure 3.1 Address Map for H8S/2110B (1)

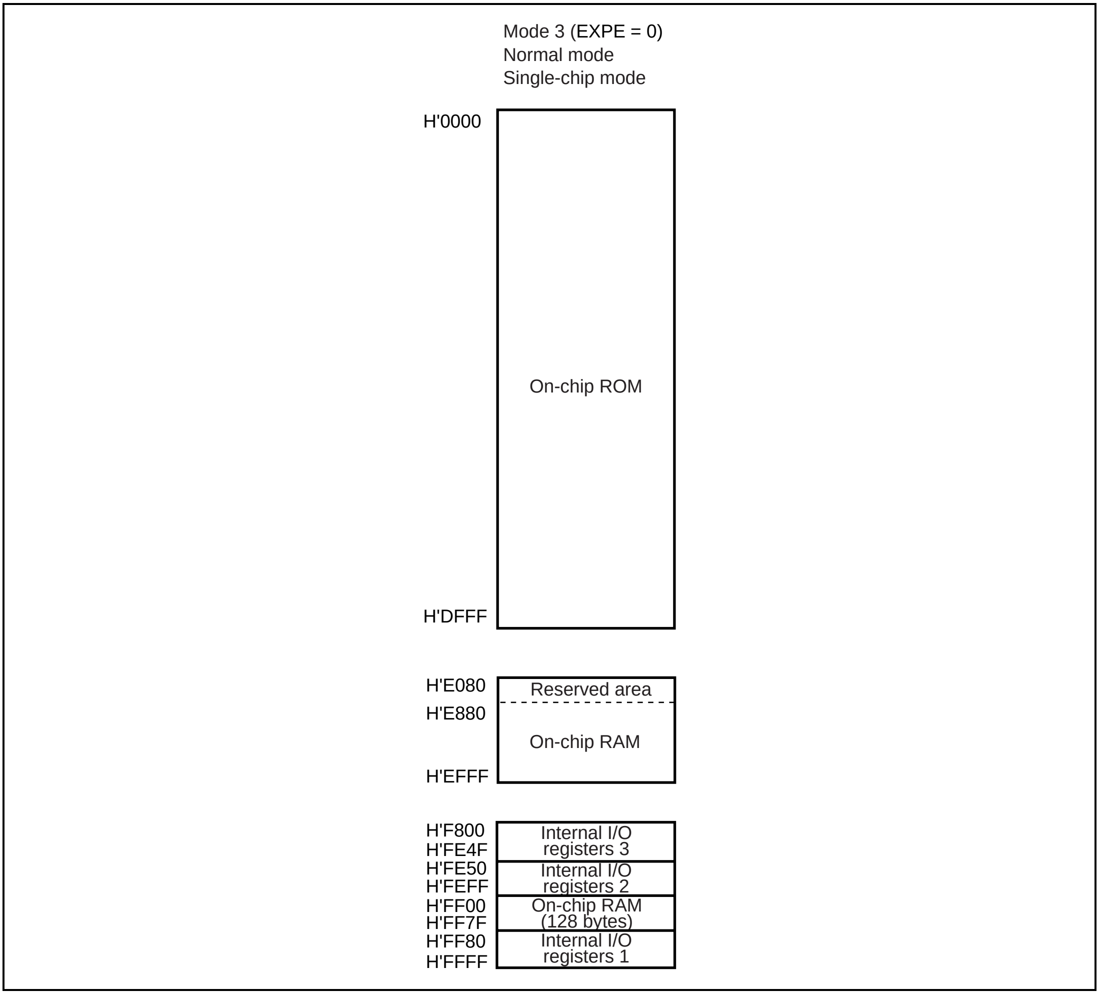


Figure 3.2 Address Map for H8S/2110B (2)

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	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.
Low	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

			Vector Address	
Exception Source		**Vector Number**	**Normal Mode**	**Advanced Mode**
Reset		0	H'0000 to H'0001	H'000000 to H'000003
Reserved for system use		1 | 5	H'0002 to H'0003 | H'000A to H'000B	H'000004 to H'000007 | 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
Reserved for system use		12 | 15	H'0018 to H'0019 | H'001E to H'001F	H'000030 to H'000033 | H'00003C to H'00003F
External interrupt	IRQ0	16	H'0020 to H'0021	H'000040 to H'000043
	IRQ1	17	H'0022 to H'0023	H'000044 to H'000047
	IRQ2	18	H'0024 to H'0025	H'000048 to H'00004B
	IRQ3	19	H'0026 to H'0027	H'00004C to H'00004F
	IRQ4	20	H'0028 to H'0029	H'000050 to H'000053
	IRQ5	21	H'002A to H'002B	H'000054 to H'000057
	IRQ6	22	H'002C to H'002D	H'000058 to H'00005B
	IRQ7	23	H'002E to H'002F	H'00005C to H'00005F
Internal interrupt*		24 | 111	H'0030 to H'0031 | H'00DE to H'00DF	H'000060 to H'000063 | H'0001BC to H'0001BF

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

4.3 Reset

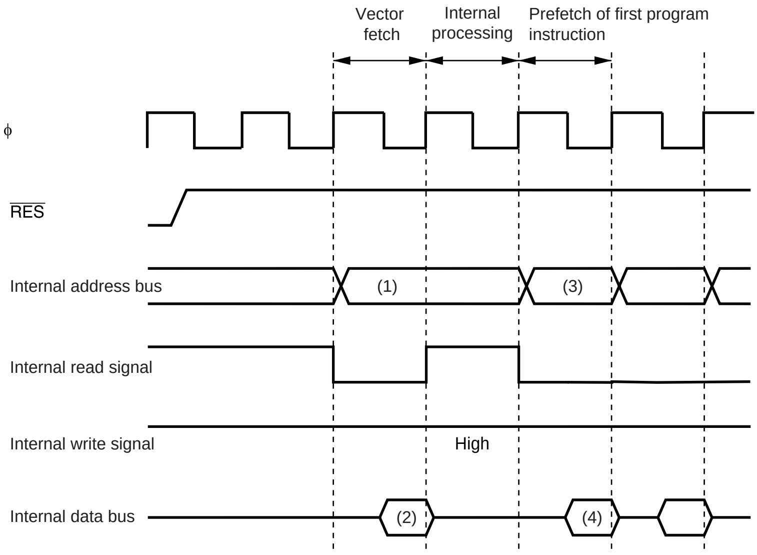
A reset has the highest exception priority. When the $\overline{\text{RES}}$ pin goes low, all processing halts and this LSI enters the reset. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-on. To reset the chip during operation, hold the $\overline{\text{RES}}$ 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 11, 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.



(1) Reset exception handling vector address ((1) = H'0000)
(2) Start address (contents of reset exception handling vector address)
(3) Start address ((3) = (2))
(4) First program instruction

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) are initialized, and all modules 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, IRQ7 to IRQ0, KIN15 to KIN0, and WUE7 to WUE0) and internal interrupt sources from the on-chip peripheral modules. NMI is an interrupt with the highest priority. For details, refer to 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

	CCR	
Interrupt Control Mode	**I**	**UI**
0	1	—
1	1	1

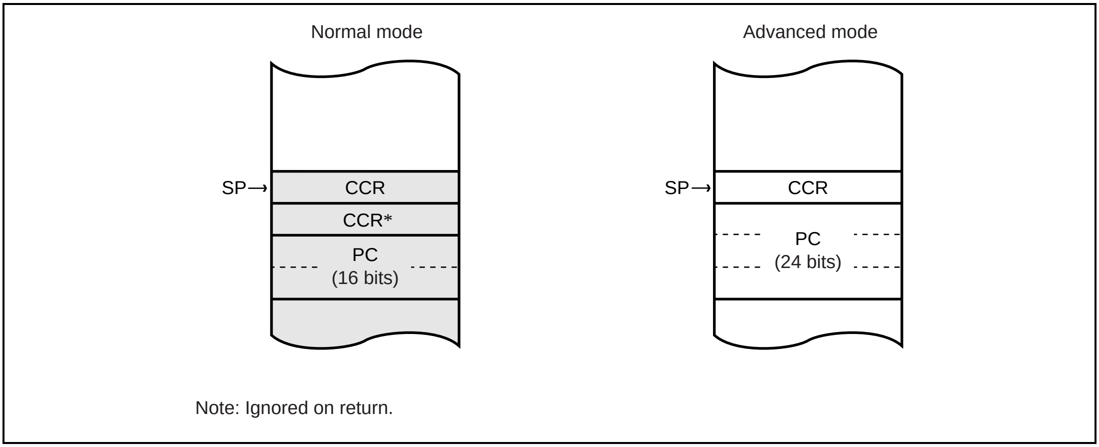
Legend

1: Set to 1

—: Retains value prior to execution

4.6 Stack Status after Exception Handling

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



Note: Ignored on return.

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.

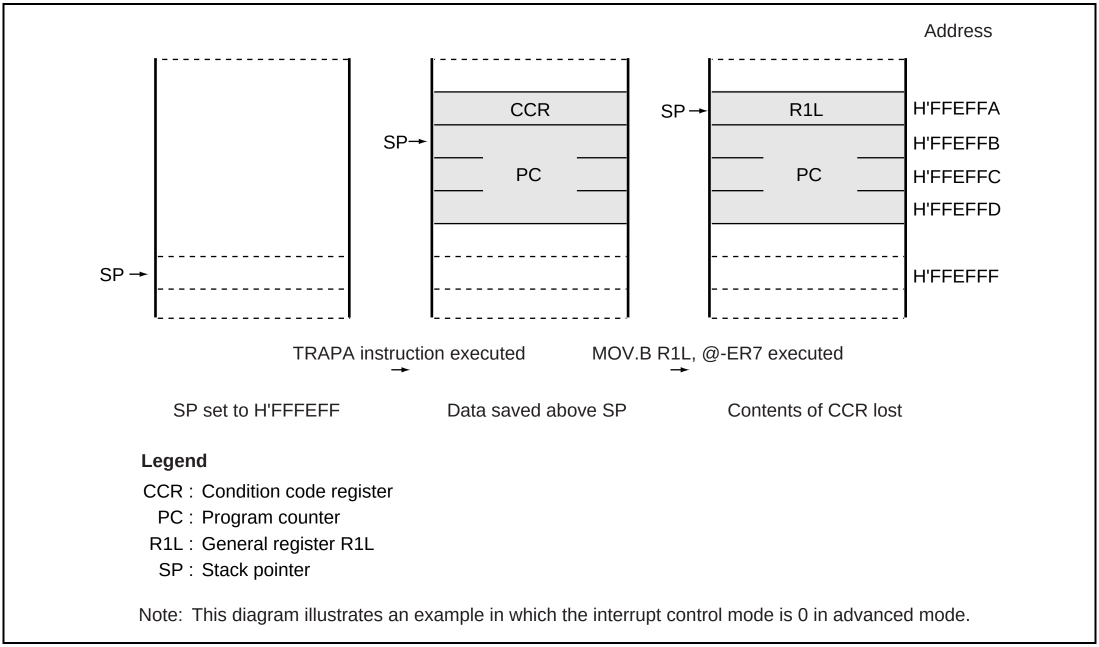
Use the following instructions to save registers:

```
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.



Legend

CCR : Condition code register
PC : Program counter
R1L : General register R1L
SP : Stack pointer

Note: This diagram illustrates an example in which the interrupt control mode is 0 in advanced mode.

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

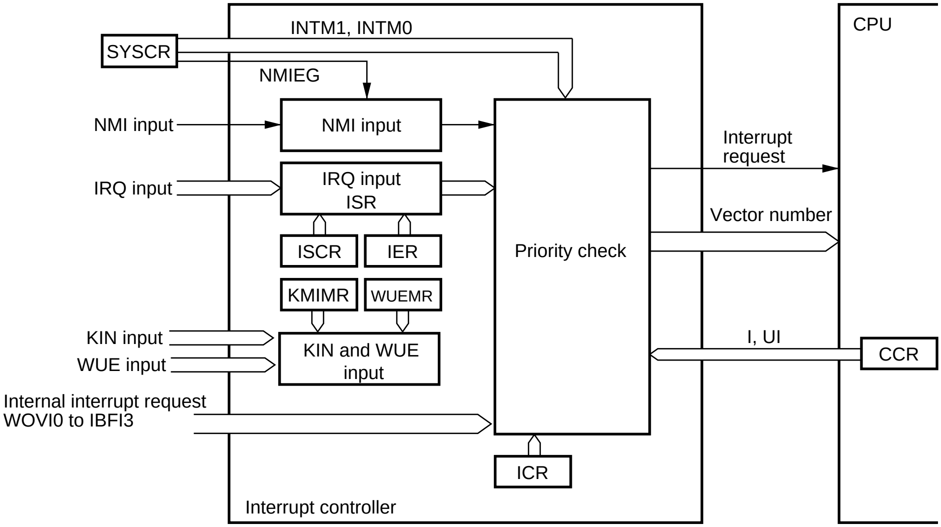
  An interrupt control register (ICR) is provided for setting interrupt priorities. Three priority levels can be set for each module for all interrupts except NMI and address break.

- 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-one 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 $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$. The IRQ6 interrupt is shared by the interrupt from the $\overline{\text{IRQ6}}$ pin and eight external interrupt inputs ($\overline{\text{KIN7}}$ to $\overline{\text{KIN0}}$), and the IRQ7 interrupt is shared by the interrupt from the $\overline{\text{IRQ7}}$ pin and sixteen external interrupt inputs ($\overline{\text{KIN15}}$ to $\overline{\text{KIN8}}$ and $\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$). $\overline{\text{KIN15}}$ to $\overline{\text{KIN0}}$ and $\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$ can be masked individually by the user program.



Legend:

ICR	: Interrupt control register
ISCR	: IRQ sense control register
IER	: IRQ enable register
ISR	: IRQ status register
KMIMR	: Keyboard matrix interrupt mask register
WUEMR	: Wake-up event interrupt mask register
SYSCR	: System control register

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
$\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$	Input	Maskable external interrupts Rising edge, falling edge, both edges, or level sensing, can be selected individually for each pin.
$\overline{\text{KIN15}}$ to $\overline{\text{KIN0}}$	Input	Maskable external interrupts Falling edge or level sensing can be selected.
$\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$	Input	Maskable external interrupts Falling edge or level sensing can be selected.

5.3 Register Descriptions

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

- Interrupt control registers A to C (ICRA to ICRC)
- Address break control register (ABRKCR)
- Break address registers A to C (BARA to BARC)
- IRQ sense control registers (ISCRH, ISCRL)
- IRQ enable register (IER)
- IRQ status register (ISR)
- Keyboard matrix interrupt mask registers (KMIMRA, KMIMR)
- Wake-up event interrupt mask register (WUEMRB)

5.3.1 Interrupt Control Registers A to C (ICRA to ICRC)

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

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	ICRn7 to IRCn0	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)

n: A to C

Table 5.2 Correspondence between Interrupt Source and ICR

		Register		
Bit	**Bit Name**	**ICRA**	**ICRB**	**ICRC**
7	ICRn7	IRQ0	—	—
6	ICRn6	IRQ1	FRT	SCI_1
5	ICRn5	IRQ2, IRQ3	—	—
4	ICRn4	IRQ4, IRQ5	—	IIC_0
3	ICRn3	IRQ6, IRQ7	TMR_0	IIC_1
2	ICRn2	—	TMR_1	—
1	ICRn1	WDT_0	TMR_X , TMR_Y	LPC
0	ICRn0	WDT_1	Keyboard buffer controller	—

n: A to C

—: 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	CMF	0	R	Condition Match Flag Address break source flag. Indicates that an address specified by BARA to BARC is prefetched. [Setting condition] When an address specified by BARA to BARC is prefetched while the BIE flag is set to 1. [Clearing condition] When an exception handling is executed for an address break interrupt.
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	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

5.3.4 IRQ Sense Control Registers (ISCRH, ISCRL)

The ISCR registers select the source that generates an interrupt request at pins $\overline{\mathrm{IRQ7}}$ to $\overline{\mathrm{IRQ0}}$.

- 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 00: Interrupt request generated at low level of $\overline{\mathrm{IRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\mathrm{IRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\mathrm{IRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\mathrm{IRQn}}$ input (n = 7 to 4)
5	IRQ6SCB	0	R/W	
4	IRQ6SCA	0	R/W	
3	IRQ5SCB	0	R/W	
2	IRQ5SCA	0	R/W	
1	IRQ4SCB	0	R/W	
0	IRQ4SCA	0	R/W	

- ISCRL

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

5.3.5 IRQ Enable Register (IER)

IER controls the enabling and disabling of interrupt requests IRQ7 to IRQ0.

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7E	0	R/W	IRQn Enable (n = 7 to 0) The IRQn interrupt request is enabled when this bit is 1.
6	IRQ6E	0	R/W	
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 Register (ISR)

The ISR register is a flag register that indicates the status of IRQ7 to IRQ0 interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7F	0	R/(W)*	[Setting condition] When the interrupt source selected by the ISCR registers occurs [Clearing conditions] • When reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag • When interrupt exception handling is executed when low-level detection is set and $\overline{\text{IRQn}}$ input is high (n = 7 to 0) • When IRQn interrupt exception handling is executed when falling-edge, rising-edge, or both-edge detection is set
6	IRQ6F	0	R/(W)*	
5	IRQ5F	0	R/(W)*	
4	IRQ4F	0	R/(W)*	
3	IRQ3F	0	R/(W)*	
2	IRQ2F	0	R/(W)*	
1	IRQ1F	0	R/(W)*	
0	IRQ0F	0	R/(W)*	

Note: * Only 0 can be written, for flag clearing.

5.3.7 Keyboard Matrix Interrupt Mask Registers (KMIMRA, KMIMR) Wake-Up Event Interrupt Mask Register (WUEMRB)

The KMIMRA, KMIMR, and WUEMRB registers enable or disable key-sensing interrupt inputs ($\overline{\text{KIN15}}$ to $\overline{\text{KIN0}}$), and wake-up event interrupt inputs ($\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$).

- KMIMRA

Bit	Bit Name	Initial Value	R/W	Description
7	KMIMR15	1	R/W	Keyboard Matrix Interrupt Mask 15 to 8 These bits enable or disable a key-sensing input interrupt request (KIN15 to KIN8). 0: Enables a key-sensing input interrupt request 1: Disables a key-sensing input interrupt request
6	KMIMR14	1	R/W	
5	KMIMR13	1	R/W	
4	KMIMR12	1	R/W	
3	KMIMR11	1	R/W	
2	KMIMR10	1	R/W	
1	KMIMR9	1	R/W	
0	KMIMR8	1	R/W	

- KMIMR

Bit	Bit Name	Initial Value	R/W	Description
7	KMIMR7	1	R/W	Keyboard Matrix Interrupt Mask 7 to 0 These bits enable or disable a key-sensing input interrupt request (KIN7 to KIN0). KMIMR6 also performs interrupt request mask control for pin $\overline{\text{IRQ6}}$. 0: Enables a key-sensing input interrupt request 1: Disables a key-sensing input interrupt request
6	KMIMR6	0	R/W	
5	KMIMR5	1	R/W	
4	KMIMR4	1	R/W	
3	KMIMR3	1	R/W	
2	KMIMR2	1	R/W	
1	KMIMR1	1	R/W	
0	KMIMR0	1	R/W	

- WUEMRB

Bit	Bit Name	Initial Value	R/W	Description
7	WUEMR7	1	R/W	Wake-Up Event Interrupt Mask 7 to 0 These bits enable or disable a wake-up event input interrupt request (WUE7 to WUE0). 0: Enables a wake-up event input interrupt request 1: Disables a wake-up event input interrupt request
6	WUEMR6	1	R/W	
5	WUEMR5	1	R/W	
4	WUEMR4	1	R/W	
3	WUEMR3	1	R/W	
2	WUEMR2	1	R/W	
1	WUEMR1	1	R/W	
0	WUEMR0	1	R/W	

Figure 5.2 shows the relationship between interrupts IRQ7 and IRQ6, interrupts KIN15 to KIN0, interrupts WUE7 to WUE0, and registers KMIMRA, KMIMR, and WUEMRB.

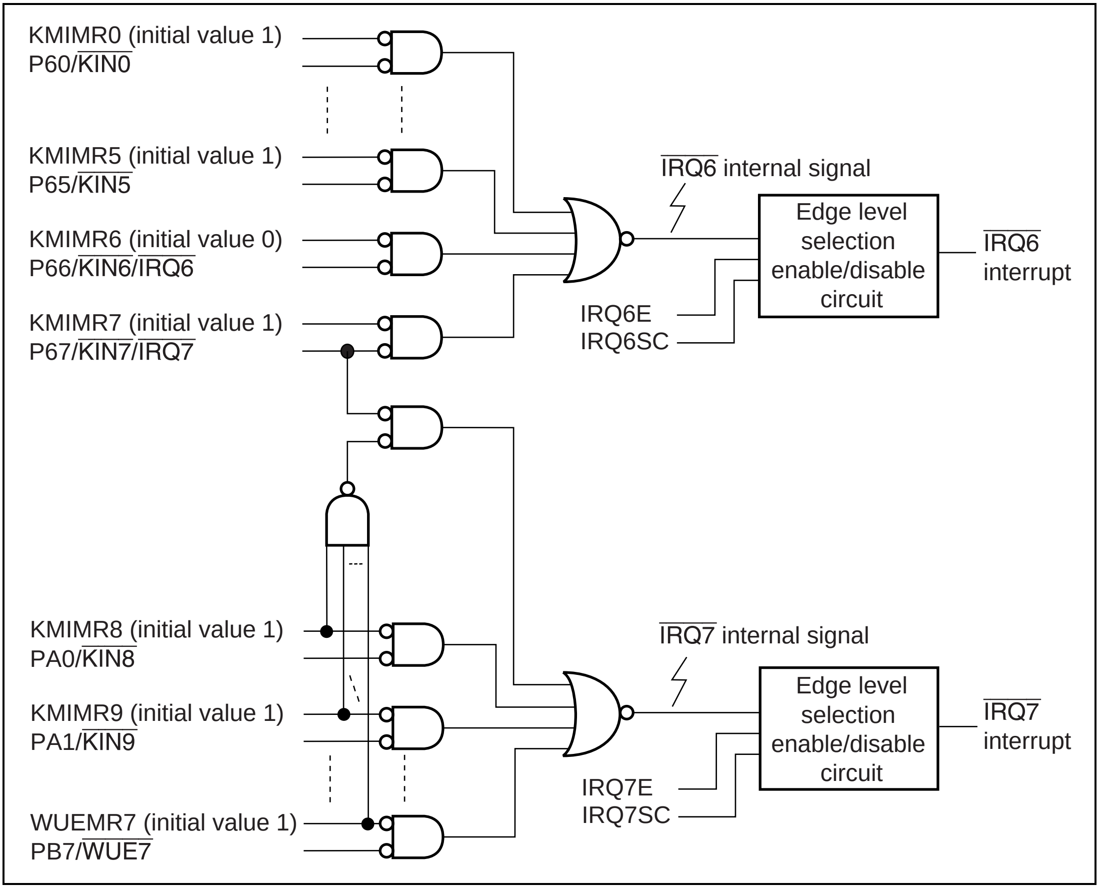


Figure 5.2 Relationship between Interrupts IRQ7 and IRQ6, Interrupts KIN15 to KIN0, Interrupts WUE7 to WUE0, and Registers KMIMR, KMIMRA, and WUEMRB

If any of bits KMIMR15 to KMIMR8 or WUEMRB7 to WUEMRB0 is cleared to 0, interrupt input from the IRQ7 pin will be ignored. When pins KIN7 to KIN0, KIN15 to KIN8, or WUE7 to WUE0 are used as key-sense interrupt input pins or wakeup event interrupt input pins, either low-level sensing or falling-edge sensing must be designated as the interrupt sense condition for the corresponding interrupt source (IRQ6 or IRQ7).

5.4 Interrupt Sources

5.4.1 External Interrupts

There are four types of external interrupts: NMI, IRQ7 to IRQ0, KIN15 to KIN0 and WUE7 to WUE0. WUE7 to WUE0 and KIN15 to KIN8 share the IRQ7 interrupt source, and KIN7 to KIN0 share the IRQ6 interrupt source. Of these, NMI, IRQ7, IRQ6, and IRQ2 to IRQ0 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.

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

- The interrupt exception handling for interrupt requests IRQ7 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 $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.
- Enabling or disabling of interrupt requests IRQ7 to IRQ0 can be selected with IER.
- Interrupt control levels can be specified by the ICR settings.
- The status of interrupt requests IRQ7 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

The detection of IRQ7 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 DDR to 0 to use the pin as an I/O pin for another function.

A block diagram of interrupts IRQ7 to IRQ0 is shown in figure 5.3.

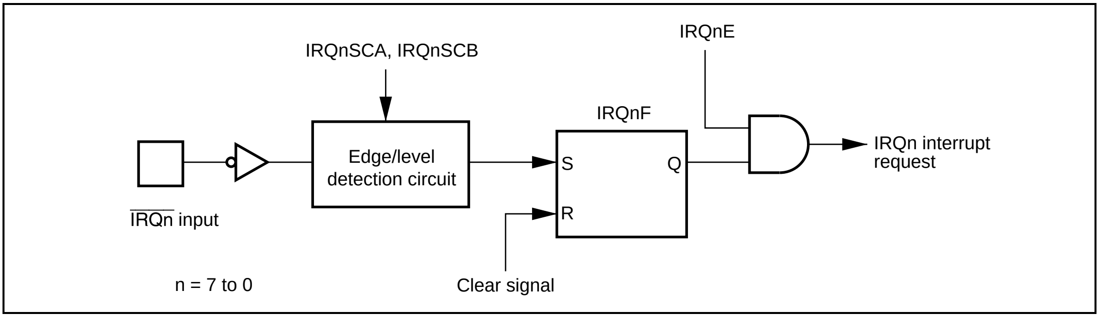


Figure 5.3 Block Diagram of Interrupts IRQ7 to IRQ0

When pin $\overline{\text{IRQ6}}$ is used as an IRQ6 interrupt input pin, clear the KMIMR6 bit to 0.

When pin $\overline{\text{IRQ7}}$ is used as an IRQ7 interrupt pin, set all of bits KMIMR15 to KMIMR8 and WUEMR7 to WUEMR0 to 1. If any of these bits is cleared to 0, IRQ7 interrupt input from the $\overline{\text{IRQ7}}$ pin will be ignored.

Since interrupt request flags IRQ7F to IRQ0F are set each time the setting condition is satisfied, regardless of the IER setting, refer to a needed flag only.

KIN15 to KIN0 Interrupts, WUE7 to WUE0 Interrupts: Interrupts KIN15 to KIN0 and WUE7 to WUE0 are requested by an input signal at pins $\overline{\text{KIN15}}$ to $\overline{\text{KIN0}}$ and $\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$. When pins $\overline{\text{KIN15}}$ to $\overline{\text{KIN0}}$ and $\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$ are used for key-sense input or wakeup event, clear the corresponding KMIMR and WUEMR bits to 0 in order to enable their key-sense input and wakeup event interrupts. Remaining unused KMIMR and WUEMR bits for key-sense input should be set to 1 in order to disable interrupts. Interrupts WUE7 to WUE0 and KIN15 to KIN8 generate IRQ7 interrupts, and interrupts KIN7 to KIN0 generate IRQ6 interrupts. The pin conditions for interrupt request generation, enable of interrupt requests, settings of interrupt control levels, and status display of interrupt requests depend on each setting and display of the IRQ7 or IRQ6 interrupt.

When pins $\overline{\text{KIN7}}$ to $\overline{\text{KIN0}}$, $\overline{\text{KIN15}}$ to $\overline{\text{KIN8}}$, or $\overline{\text{WUE7}}$ to $\overline{\text{WUE0}}$ are used as key-sense interrupt input pins or wakeup event interrupt input pins, either low-level sensing or falling-edge sensing must be designated as the interrupt sense condition for the corresponding interrupt source (IRQ6 or IRQ7).

5.4.2 Internal Interrupts

Internal interrupts issued from the on-chip peripheral modules have the following features:

1. 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.
2. The control level for each interrupt can be set by ICR.

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 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 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 IRQ3	18 19	H'0024 H'0026	H'000048 H'00004C	ICRA5	
	IRQ4 IRQ5	20 21	H'0028 H'002A	H'000050 H'000054	ICRA4	
	IRQ6, KIN7 to KIN0 IRQ7, KIN15 to KIN8, WUE7 to WUE0	22 23	H'002C H'002E	H'000058 H'00005C	ICRA3	
—	Reserved for system use	24	H'0030	H'000060	—	
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	—	
—	Reserved for system use	28 to 47	H’0038 to H’005E	H’000070 to H’0000BC	—	
FRT	ICIA (Input capture A) ICIB (Input capture B) ICIC (Input capture C) ICID (Input capture D) OCIA (Output compare A) OCIB (Output compare B) FOVI (Overflow) Reserved for system use	48 49 50 51 52 53 54 55	H'0060 H'0062 H'0064 H'0066 H'0068 H'006A H'006C H'006E	H'0000C0 H'0000C4 H'0000C8 H'0000CC H'0000D0 H'0000D4 H'0000D8 H'0000DC	ICRB6	
—	Reserved for system use	56 to 63	H’0070 to H’007E	H’0000E0 to H’0000FC	—	
TMR_0	CMIA0 (Compare match A) CMIB0 (Compare match A) OVI0 (Overflow) Reserved for system use	64 65 66 67	H'0080 H'0082 H'0084 H'0086	H'000100 H'000104 H'000108 H'00010C	ICRB3	Low

Origin of Interrupt Source	Name	Vector Number	Vector Address		ICR	Priority
			Normal Mode	Advanced Mode		
TMR_1	CMIA1 (Compare match A)	68	H'0088	H'000110	ICRB2	High
	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_X, TMR_Y	CMIAY (Compare match A)	72	H'0090	H'000120	ICRB1	
	CMIBY (Compare match B)	73	H'0092	H'000124		
	OVIY (Overflow)	74	H'0094	H'000128		
	ICIX (Input capture X)	75	H'0096	H'00012C		
—	Reserved for system use	76 to 83	H'0098 to H'00A6	H'000130 to H'00014C	—	
SCI_1	ERI1 (Reception error 1)	84	H'00A8	H'000150	ICRC6	
	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		
—	Reserved for system use	88 to 91	H'00B0 to H'00B6	H'000160 to H'00016C	—	
IIC_0	IICI0 (1-byte transmission/ reception completion)	92	H'00B8	H'000170	ICRC4	
	Reserved for system use	93	H'00BA	H'000174		
IIC_1	IICI1 (1-byte transmission/ reception completion)	94	H’00BC	H’000178	ICRC3	
	Reserved for system use	95	H’00BE	H’00017C		
Keyboard buffer controller	KBIA (Reception completion A)	96	H’00C0	H’000180	ICRB0	
	KBIB (Reception completion B)	97	H’00C2	H’000184		
	KBIC (Reception completion C)	98	H’00C4	H’000188		
	Reserved for system use	99	H'00C6	H’00018C		
—	Reserved for system use	100 to 107	H’00C8 to H’00D6	H’000190 to H’0001AC	—	
LPC	ERRI (Transfer error)	108	H’00D8	H’0001B0	ICRC1	
	IBF1 (IDR1 reception completion)	109	H’00DA	H’0001B4		
	IBF2 (IDR2 reception completion)	110	H’00DC	H’0001B8		
	IBF3 (IDR3 reception completion)	111	H’00DE	H’0001BC		Low

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.

5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests other than NMI and address breaks are masked by ICR and the I bit of the CCR in the CPU. Figure 5.4 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.

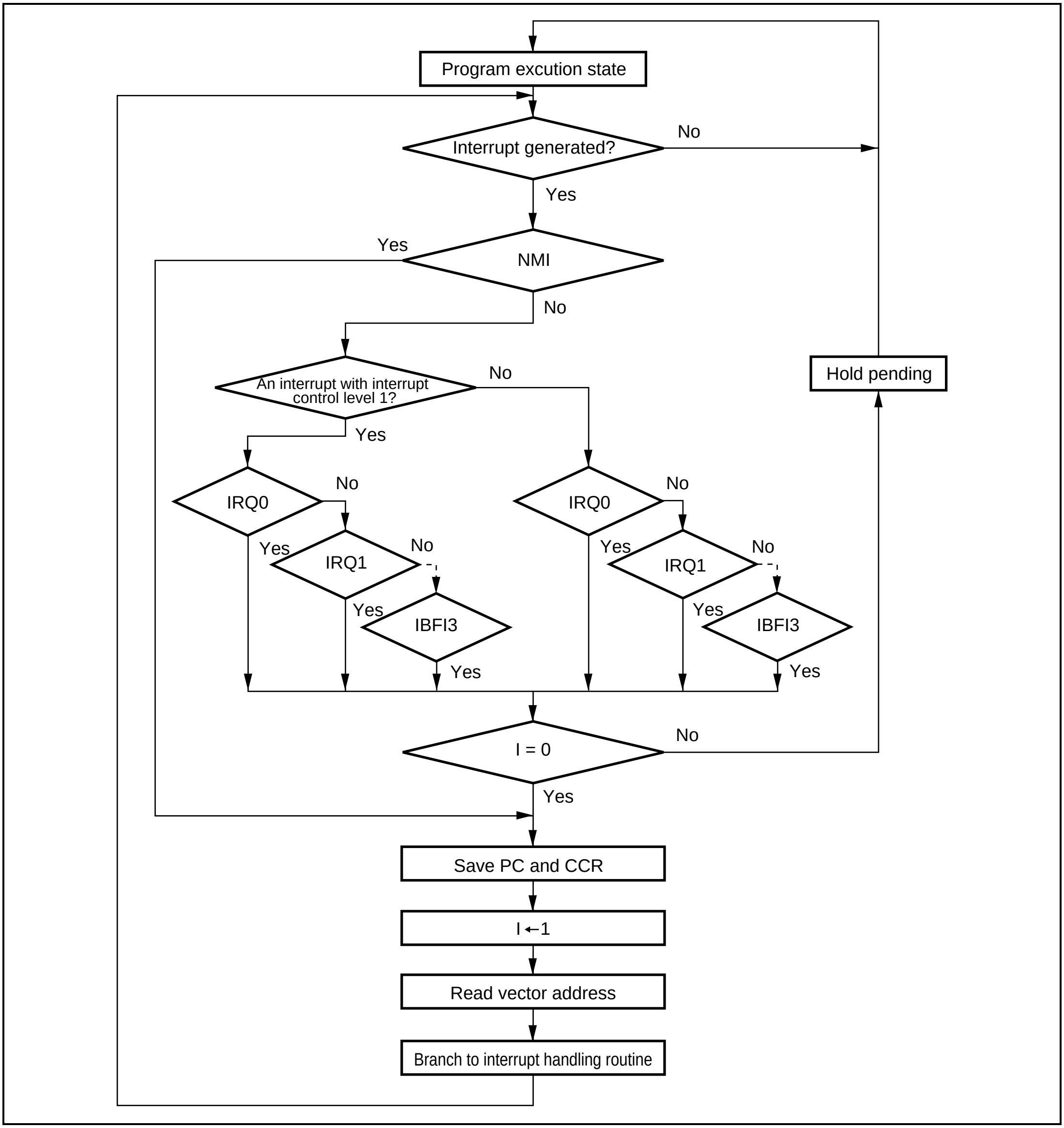


Figure 5.4 Flowchart of Procedure up to Interrupt Acceptance in Interrupt Control Mode 0

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.

- 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
- 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 ICRC are set to H’20, H’00, and H’00, respectively (IRQ2 and IRQ3 interrupts are set to control level 1, and other interrupts are set to control level 0) is shown below. Figure 5.5 shows a state transition diagram.

- All interrupt requests are accepted when I = 0. (Priority order: NMI > IRQ2 > IRQ3 > address break > IRQ0 > IRQ1 …)
- Only NMI, IRQ2, IRQ3 and address break interrupt requests are accepted when I = 1 and UI = 0.
- Only an NMI and address break interrupt request is accepted when I = 1 and UI = 1.

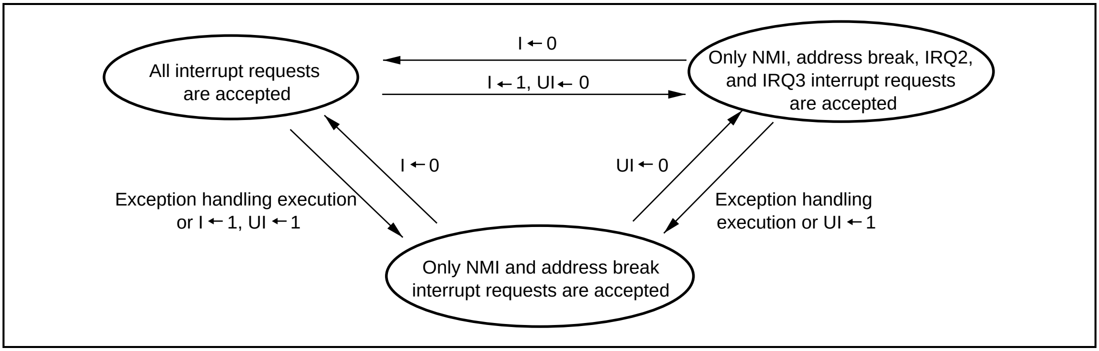

I ← 0
All interrupt requests are accepted
I ← 1, UI ← 0
Only NMI, address break, IRQ2, and IRQ3 interrupt requests are accepted
I ← 0
UI ← 0
Exception handling execution or I ← 1, UI ← 1
Exception handling execution or UI ← 1
Only NMI and address break interrupt requests are accepted

Figure 5.5 State Transition in Interrupt Control Mode 1

Figure 5.6 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.

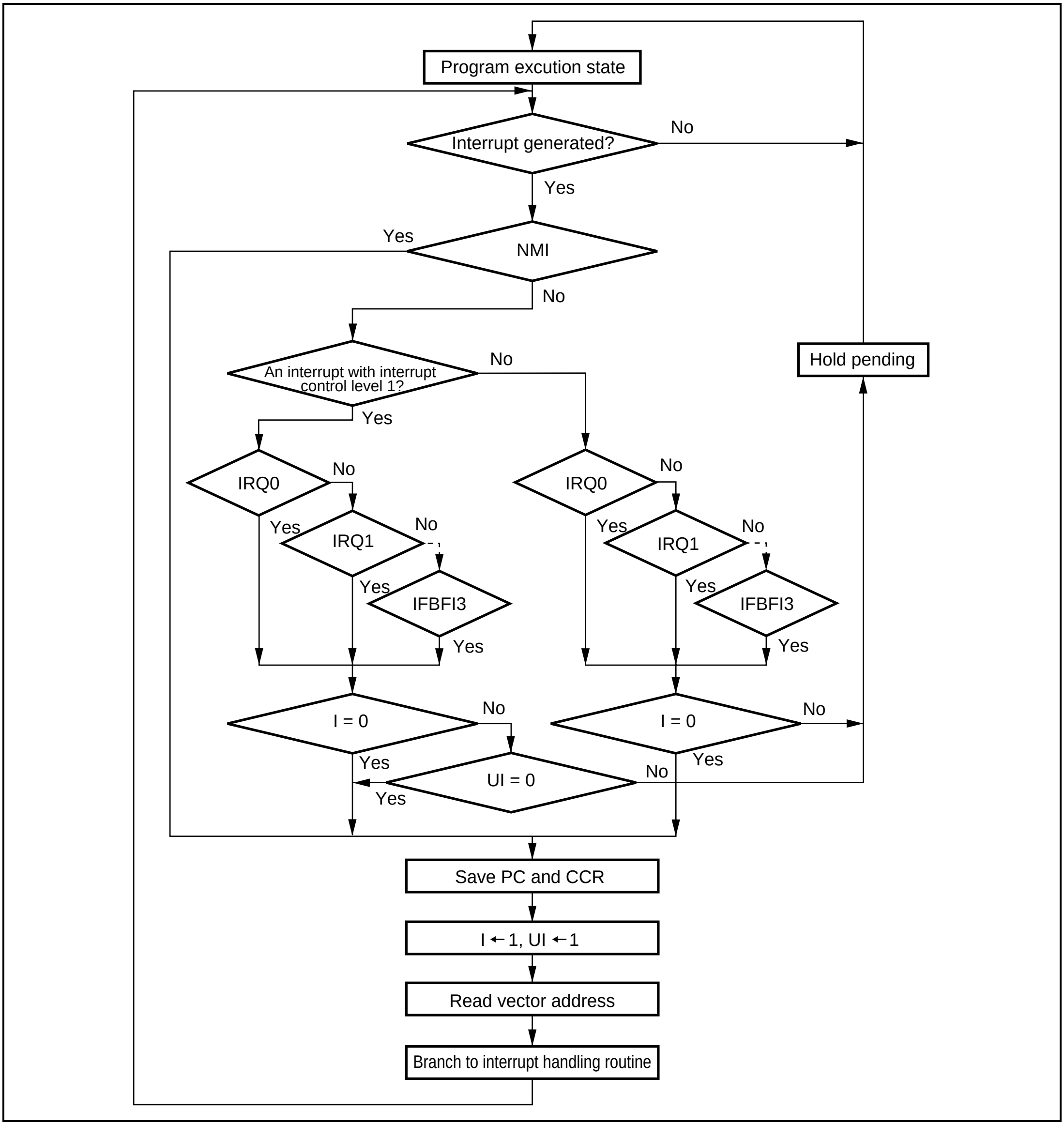
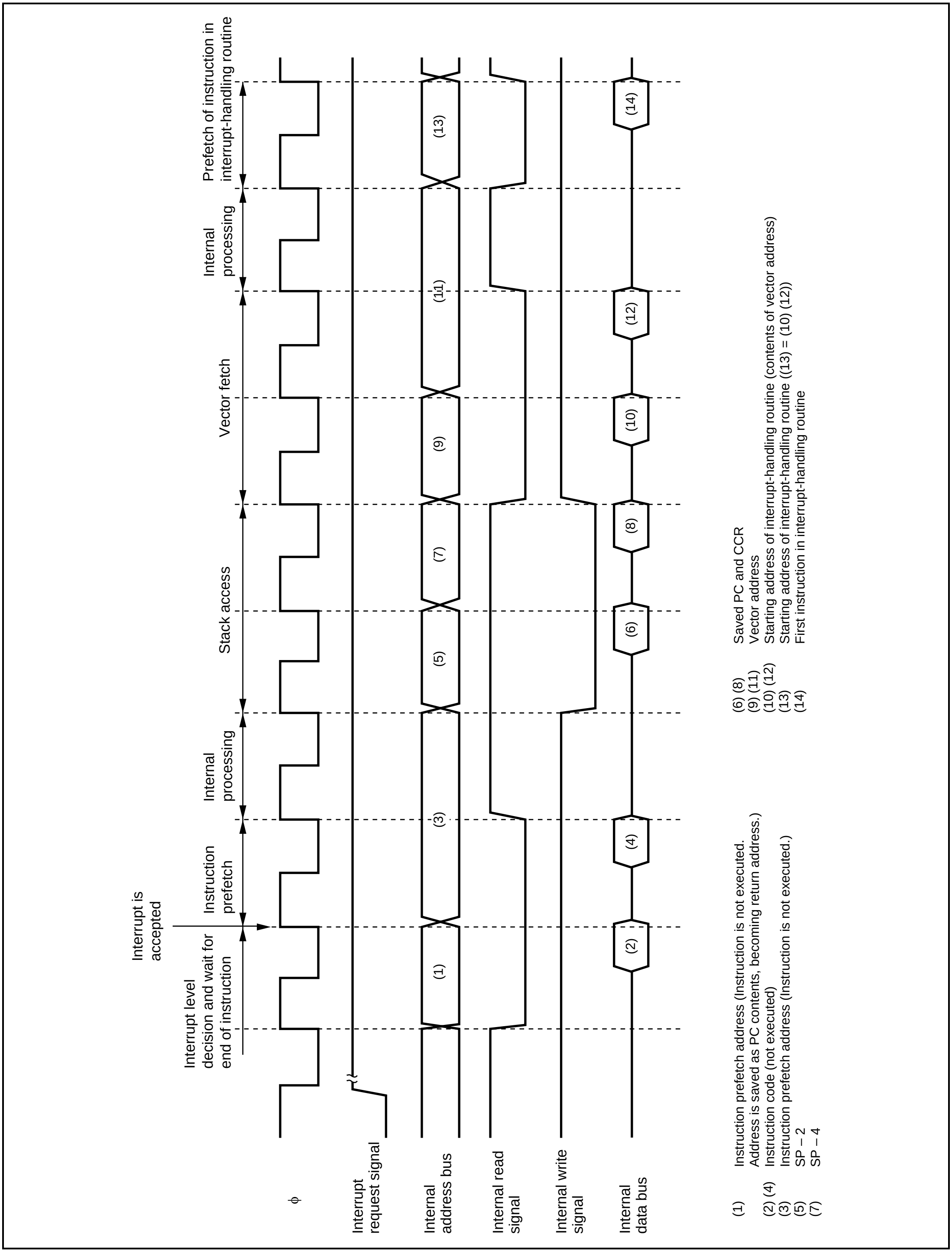


Figure 5.6 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1

5.6.3 Interrupt Exception Handling Sequence

Figure 5.7 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.



(1) Instruction prefetch address (Instruction is not executed.
Address is saved as PC contents, becoming return address.)
(2) (4) Instruction code (not executed)
(3) Instruction prefetch address (Instruction is not executed.)
(5) SP – 2
(7) SP – 4
(6) (8) Saved PC and CCR
(9) (11) Vector address
(10) (12) Starting address of interrupt-handling routine (contents of vector address)
(13) Starting address of interrupt-handling routine ((13) = (10) (12))
(14) First instruction in interrupt-handling routine

Figure 5.7 Interrupt Exception Handling

5.6.4 Interrupt Response Times

Table 5.5 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 5.5 are explained in table 5.6.

Table 5.5 Interrupt Response Times

No.	Execution Status	Normal Mode	Advanced Mode
1	Interrupt priority determination*[1]	3	
2	Number of wait states until executing instruction ends*[2]	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*[3]	$2 \cdot S_I$	
6	Internal processing*[4]	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 5.6 Number of States in Interrupt Handling Routine Execution Status

	Object of Access
Symbol	**Internal Memory**
Instruction fetch S_I	1
Branch address read S_J	
Stack manipulation S_K	

5.7 Address Break

5.7.1 Features

This LSI can determine the specific address prefetch by the CPU to generate an address break interrupt by setting ABRKCR and BAR. If an address break interrupt is generated, the address break interrupt exception handling is performed.

With this function, the execution start point of a program containing a bug is detected and execution is branched to the correcting program.

5.7.2 Block Diagram

Figure 5.8 shows a block diagram of the address break.

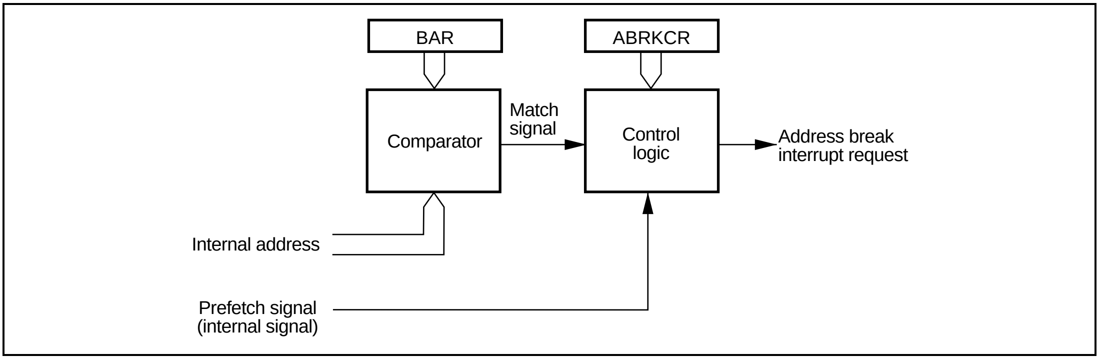

BAR
ABRKCR
Comparator
Match signal
Control logic
Address break interrupt request
Internal address
Prefetch signal (internal signal)

Figure 5.8 Address Break Block Diagram

5.7.3 Operation

If the CPU prefetches an address specified in BAR by setting ABRKCR and BAR, an address break interrupt can be generated. This address break function generates an interrupt request to the interrupt controller at prefetch, and determines the priority by the interrupt controller. When an interrupt is accepted, an interrupt exception handling is activated after the current instruction has been completed. Note that the interrupt mask control according to the I and UI bits in CCR of the CPU is invalid to an address break interrupt.

To use the address break function, set each register as follows:

1. Set a break address in the A23 to A1 bits in BAR.
2. Set the BIE bit in ABRKCR to 1 to enable the address break.
   When the BIE bit is cleared to 0, an address break is not requested.

When the setting conditions are satisfied, the CMF flag in ABRKCR is set to 1 to request an interrupt. The interrupt source should be determined by the interrupt handling routine if necessary.

5.7.4 Usage Notes

1. In an address break, the break address should be an address where the first byte of the instruction exists. Otherwise, a break condition will not be satisfied.
2. In normal mode, addresses A23 to A16 are not compared.
3. When the branch instructions (Bcc, BSR), jump instructions (JMP, JSR), RST instruction, and RTE instruction are placed immediately prior to the address specified by BAR, a prefetch signal to the address may be output to request an address break by executing these instruction. It is necessary to take countermeasures: do not set a break address to an address immediately after these instructions, or determine whether interrupt handling is performed by satisfaction of a normal condition.
4. An address break interrupt is generated by combining the internal prefetch signal and an address. Therefore, the timing to enter the interrupt exception handling differs according to the instructions at the specified and at prior addresses and execution cycles.

Figure 5.9 shows an example of address timing.

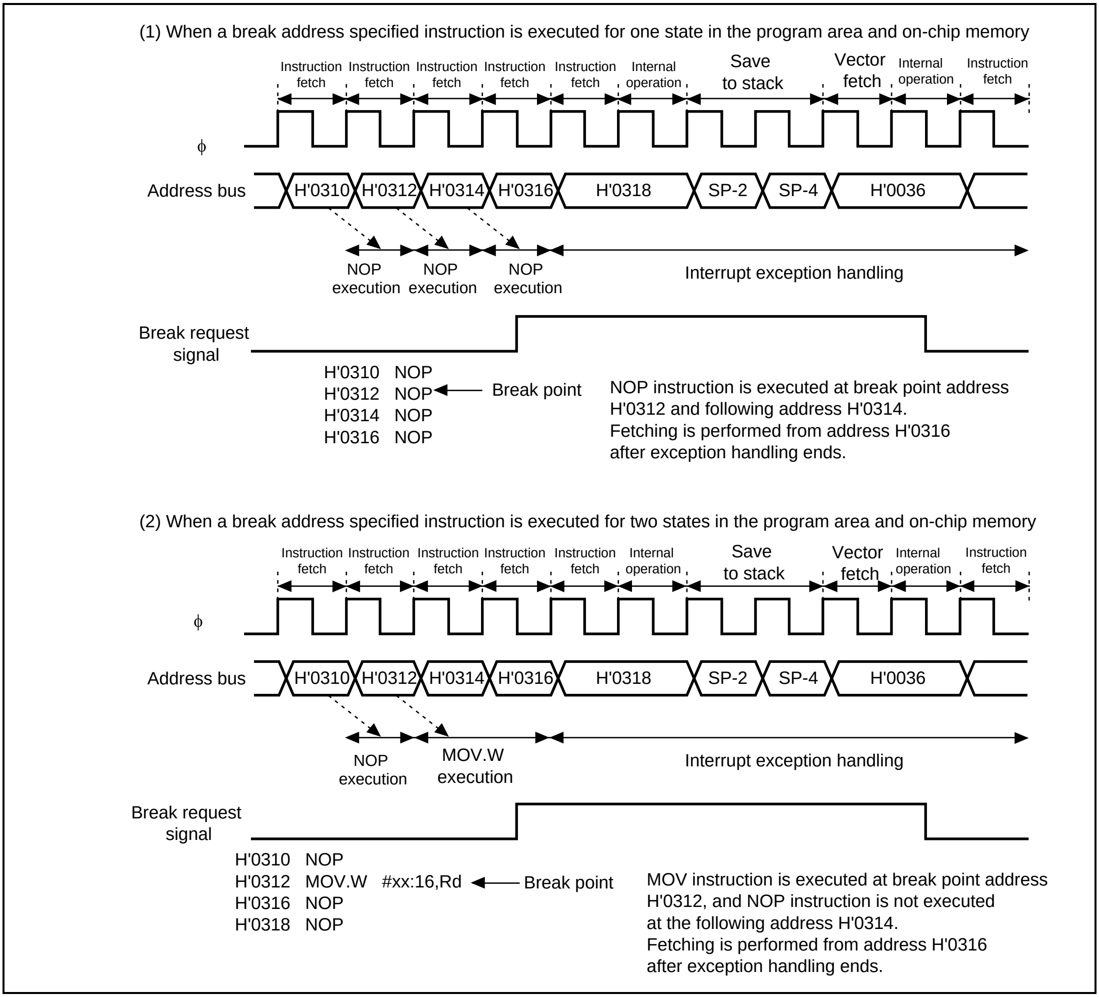


Figure 5.9 Address Break Timing Example

5.8 Usage Notes

5.8.1 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.

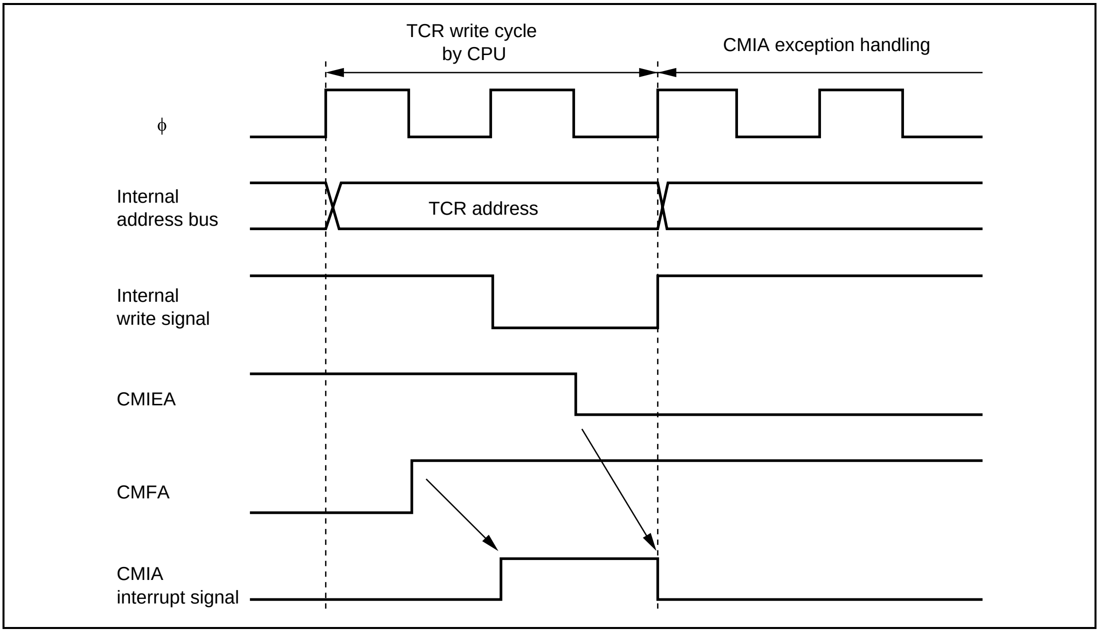


Figure 5.10 Conflict between Interrupt Generation and Disabling

5.8.2 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.

5.8.3 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
```

5.8.4 IRQ Status Register (ISR)

According to the pin status after a reset, IRQnF may be set to 1, so ISR should be read after a reset to write 0. (n = 7 to 0)

Section 6 Bus Controller (BSC)

Since this LSI does not have an externally extended function, it does not have an on-chip bus controller (BSC). Considering the software compatibility with similar products, you must be careful to set appropriate values to the control registers for the bus controller.

6.1 Register Descriptions

The bus controller has the following registers.

- Bus control register (BCR)
- Wait state control register (WSCR)

6.1.1 Bus Control Register (BCR)

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R/W	Reserved The initial value should not be changed.
6	ICIS0	1	R/W	Idle Cycle Insertion The initial value should not be changed.
5	BRSTRM	0	R/W	Burst ROM Enable The initial value should not be changed.
4	BRSTS1	1	R/W	Burst Cycle Select 1 The initial value should not be changed.
3	BRSTS0	0	R/W	Burst Cycle Select 0 The initial value should not be changed.
2	—	0	R/W	Reserved The initial value should not be changed.
1	IOS1	1	R/W	IOS Select 1, 0
0	IOS0	1	R/W	The initial value should not be changed.

6.1.2 Wait State Control Register (WSCR)

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved
6	—	0	R/W	The initial value should not be changed.
5	ABW	1	R/W	Bus Width Control The initial value should not be changed.
4	AST	1	R/W	Access State Control The initial value should not be changed.
3	WMS1	0	R/W	Wait Mode Select 1, 0
2	WMS0	0	R/W	The initial value should not be changed.
1	WC1	1	R/W	Wait Count 1, 0
0	WC0	1	R/W	The initial value should not be changed.

Section 7 I/O Ports

7.1 Overview

This LSI has eleven I/O ports (ports 1 to 9*, A, and B).

Table 7.1 is a summary of the port functions. The pins of each port also have other functions.

Each port includes a data direction register (DDR) that controls input/output and data registers (DR, ODR) that store output data.

Ports 1 to 3, 6, A, and B have an on-chip input pull-up MOS function. For ports A and B, 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 MOS.

Ports 1 to 9*, A, and B 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. Ports 1, 2, and 3 can drive an LED (10 mA sink current).

Port A input and output use by the VccB power supply, which is independent of the Vcc power supply. When the VccB voltage is 5V, the pins on port A will be 5-V tolerant.

PA4 to PA7 of port A have bus-buffer drive capability.

P52 in port 5, P97 in port 9, P86 in port 8 and P42 in port 4 are NMOS push-pull outputs. P52, P97, P86 and P42 are thus 5-V tolerant, with DC characteristics that are dependent on the Vcc voltage.

For the P42, P52/ExSCK1, P86/SCK1, and P97 outputs, connect pull-up resistors to pins to raise output-high-level voltage.

Note:* The program development tool (emulator) does not support the output of port 7.

Table 7.1 Port Functions of H8S/2110B

Port	Description	Mode 2, 3	I/O Status
Port 1	General I/O port	P17	On-chip input pull-up MOSs
		P16	
		P15	
		P14	
		P13	
		P12	
		P11	
		P10	
Port 2	General I/O port	P27	On-chip input pull-up MOSs
		P26	
		P25	
		P24	
		P23	
		P22	
		P21	
		P20	
Port 3	General I/O port also functioning as LPC input/output pins	P37/SERIRQ	On-chip input pull-up MOSs
		P36/LCLK	
		P35/$\overline{\text{LRESET}}$	
		P34/$\overline{\text{LFRAME}}$	
		P33/LAD3	
		P32/LAD2	
		P31/LAD1	
		P30/LAD0	

Port	Description	Mode 2, 3	I/O Status
Port 4	General I/O port also functioning as PWMX output, TMR_0 and TMR_1 input/output, and IIC_1 input/output pins	P47/PWX1 P46/PWX0 P45/TMRI1 P44/TMO1 P43/TMCI1 P42/TMRI0/SDA1 P41/TMO0 P40/TMCI0	
Port 5	General I/O port also functioning as SCI_1 extended input/output and IIC_0 input/output pins	P52/ExSCK1*[1]/SCL0 P51/ExRxD1*[1] P50/ExTxD1*[1]	
Port 6	General I/O port also functioning as interrupt input, FRT input/output, TMR_X input/output, TMR_Y output, and key-sense interrupt input	P67/TMOX/$\overline{\text{KIN7}}$/$\overline{\text{IRQ7}}$ P66/FTOB/$\overline{\text{KIN6}}$/$\overline{\text{IRQ6}}$ P65/FTID/$\overline{\text{KIN5}}$ P64/FTIC/$\overline{\text{KIN4}}$ P63/FTIB/$\overline{\text{KIN3}}$ P62/FTIA/$\overline{\text{KIN2}}$/TMIY P61/FTOA/$\overline{\text{KIN1}}$ P60/FTCI/$\overline{\text{KIN0}}$/TMIX	On-chip input pull-up MOSs

Port	Description	Mode 2, 3	I/O Status
Port 7	General I/O port*2 also functioning as TMR_X extended output and TMR_Y output pins	P77*2/ExTMOX*1	
		P76*2/TMOY*1	
		P75*2	
		P74*2	
		P73*2	
		P72*2	
		P71*2	
		P70*2	
Port 8	General I/O port also functioning as interrupt input, SCI_1 input/output, LPC input/output, and IIC_1 input/output pins	P86/$\overline{\text{IRQ5}}$/SCK1/SCL1	
		P85/$\overline{\text{IRQ4}}$/RxD1	
		P84/$\overline{\text{IRQ3}}$/TxD1	
		P83/$\overline{\text{LPCPD}}$	
		P82/$\overline{\text{CLKRUN}}$	
		P81/GA20	
		P80/$\overline{\text{PME}}$	
Port 9	General I/O port also functioning as IIC_0 input/output, subclock input, ϕ output, and interrupt input	P97/SDA0	
		P96/ϕ/EXCL	
		P95	
		P94	
		P93	
		P92/$\overline{\text{IRQ0}}$	
		P91/$\overline{\text{IRQ1}}$	
		P90/$\overline{\text{IRQ2}}$	

Port	Description	Mode 2, 3	I/O Status
Port A	General I/O port also functioning as key-sense interrupt input, and keyboard buffer controller input/output pins	PA7/$\overline{\text{KIN15}}$/PS2CD PA6/$\overline{\text{KIN14}}$/PS2CC PA5/$\overline{\text{KIN13}}$/PS2BD PA4/$\overline{\text{KIN12}}$/PS2BC PA3/$\overline{\text{KIN11}}$/PS2AD PA2/$\overline{\text{KIN10}}$/PS2AC PA1/$\overline{\text{KIN9}}$ PA0/$\overline{\text{KIN8}}$	On-chip input pull-up MOSs
Port B	General I/O port also functioning as wakeup event interrupt input, and LPC input/output pins	PB7/$\overline{\text{WUE7}}$ PB6/$\overline{\text{WUE6}}$ PB5/$\overline{\text{WUE5}}$ PB4/$\overline{\text{WUE4}}$ PB3/$\overline{\text{WUE3}}$ PB2/$\overline{\text{WUE2}}$ PB1/$\overline{\text{WUE1}}$/LSCI PB0/$\overline{\text{WUE0}}$/$\overline{\text{LSMI}}$	On-chip input pull-up MOSs

Notes: 1. The program development tool (emulator) does not support this function.

2. The program development tool (emulator) does not support the output.

7.2 Port 1

Port 1 is an 8-bit I/O port. Port 1 has an on-chip input pull-up MOS function that can be controlled by software. 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)

7.2.1 Port 1 Data Direction Register (P1DDR)

P1DDR specifies input or output for the pins of port 1 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P17DDR	0	W	The corresponding port 1 pins are output ports when the P1DDR bits are set to 1, and input ports when the P1DDR bits are cleared to 0.
6	P16DDR	0	W	
5	P15DDR	0	W	
4	P14DDR	0	W	
3	P13DDR	0	W	
2	P12DDR	0	W	
1	P11DDR	0	W	
0	P10DDR	0	W	

7.2.2 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	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	

7.2.3 Port 1 Pull-Up MOS Control Register (P1PCR)

P1PCR controls the on/off status of the port 1 on-chip input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P17PCR	0	R/W	When a P1PCR bit is set to 1 with the input port setting, the input pull-up MOS is turned on.
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	

7.2.4 Pin Functions

- P17 to P10

  The pin function is switched as shown below according to the status of the P1nDDR bit.

P1nDDR	0	1
Pin Function	P17 to P10 input pins	P17 to P10 output pins

Legend

n = 7 to 0

7.2.5 Port 1 Input Pull-Up MOS

Port 1 has an on-chip input pull-up MOS function that can be controlled by software. This input pull-up MOS function can be specified as on or off on a bit-by-bit basis.

Table 7.2 summarizes the input pull-up MOS states.

Table 7.2 Input Pull-Up MOS States (Port 1)

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

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when the pin is in the input state, P1DDR = 0, and P1PCR = 1; otherwise off.

7.3 Port 2

Port 2 is an 8-bit I/O port. Port 2 has an on-chip input pull-up MOS function that can be controlled by software. 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)

7.3.1 Port 2 Data Direction Register (P2DDR)

P2DDR specifies input or output for the pins of port 2 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P27DDR	0	W	The corresponding port 2 pins are output ports when P2DDR bits are set to 1, and input ports when P2DDR bits are cleared to 0.
6	P26DDR	0	W	
5	P25DDR	0	W	
4	P24DDR	0	W	
3	P23DDR	0	W	
2	P22DDR	0	W	
1	P21DDR	0	W	
0	P20DDR	0	W	

7.3.2 Port 2 Data Register (P2DR)

P2DR stores output data for port 2.

Bit	Bit Name	Initial Value	R/W	Description
7	P27DR	0	R/W	If a port 2 read is performed while P2DDR bits are set to 1, the P2DR values are read directly, regardless of the actual pin states. If a port 2 read is performed while 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	

7.3.3 Port 2 Pull-Up MOS Control Register (P2PCR)

P2PCR controls the port 2 on-chip input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P27PCR	0	R/W	The input pull-up MOS is turned on when a P2PCR bit is set to 1 in the input port state.
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	

7.3.4 Pin Functions

- P27, P26, P25, P24, P23, P22, P21, P20

  The pin function is switched as shown below according to the status of the P2nDDR bit.

P2nDDR	0	1
Pin Function	P27 to P20 input pins	P27 to P20 output pins

Legend

n = 7 to 0

7.3.5 Port 2 Input Pull-Up MOS

Port 2 has an on-chip input pull-up MOS function that can be controlled by software. This input pull-up MOS function can be specified as on or off on a bit-by-bit basis.

Table 7.3 summarizes the input pull-up MOS states.

Table 7.3 Input Pull-Up MOS States (Port 2)

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

Legend

Off: Input pull-up MOS is always off.

On/Off: On when the pin is in the input state, P2DDR = 0, and P2PCR = 1; otherwise off.

7.4 Port 3

Port 3 is an 8-bit I/O port. Port 3 pins also function as LPC input/output pins. 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)

7.4.1 Port 3 Data Direction Register (P3DDR)

P3DDR specifies input or output for the pins of port 3 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DDR	0	W	The corresponding port 3 pins are output ports when P3DDR bits are set to 1, and input ports when P3DDR bits are cleared to 0.
6	P36DDR	0	W	
5	P35DDR	0	W	
4	P34DDR	0	W	
3	P33DDR	0	W	
2	P32DDR	0	W	
1	P31DDR	0	W	
0	P30DDR	0	W	

7.4.2 Port 3 Data Register (P3DR)

P3DR stores output data of port 3.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DR	0	R/W	If a port 3 read is performed while P3DDR bits are set to 1, the P3DR values are read directly, regardless of the actual pin states. If a port 3 read is performed while P3DDR bits are cleared to 0, the pin states are read.
6	P36DR	0	R/W	
5	P35DR	0	R/W	
4	P34DR	0	R/W	
3	P33DR	0	R/W	
2	P32DR	0	R/W	
1	P31DR	0	R/W	
0	P30DR	0	R/W	

7.4.3 Port 3 Pull-Up MOS Control Register (P3PCR)

P3PCR controls the port 3 on-chip input pull-up MOSs on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P37PCR	0	R/W	The input pull-up MOS is turned on when a P3PCR bit is set to 1 in the input port state. The input pull-up MOS function cannot be used when the host interface is enabled.
6	P36PCR	0	R/W	
5	P35PCR	0	R/W	
4	P34PCR	0	R/W	
3	P33PCR	0	R/W	
2	P32PCR	0	R/W	
1	P31PCR	0	R/W	
0	P30PCR	0	R/W	

7.4.4 Pin Functions

- P37/SERIRQ, P36/LCLK, P35/$\overline{\text{LRESET}}$, P34/$\overline{\text{LFRAME}}$, P33/LAD3, P32/LAD2, P31/LAD1, P30/LAD0

  The pin function is switched as shown below according to the combination of the HI12E bit in SYSCR2, the LPC3E to LPC1E bits in HICR0 of host interface (LPC), and the P3nDDR bit.

LPCmE	All 0		Not all 0
HI12E	0		0
P3nDDR	0	1	0
Pin Function	P37 to P30 input pins	P37 to P30 output pins	LPC input/output pins

Note:* The combination of bits not described in the above table must not be used.

m = 3 to 1: LPC input/output pins (SERIRQ, LCLK, LRESET, LFRAME, LAD3 to LAD0) when at least one of LPC3E to LPC1E is set to 1.

n = 7 to 0

7.4.5 Port 3 Input Pull-Up MOS

Port 3 has an on-chip input pull-up MOS function that can be controlled by software. This input pull-up MOS function can be specified as on or off on a bit-by-bit basis.

Table 7.4 summarizes the input pull-up MOS states.

Table 7.4 Input Pull-Up MOS States (Port 3)

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

Legend

Off: Input pull-up MOS is always off.

On/Off: On when the pin is in the input state, P3DDR = 0, and P3PCR = 1; otherwise off.

7.5 Port 4

Port 4 is an 8-bit I/O port. Port 4 pins also function as PWMX output pins, TMR_0 and TMR_1 I/O pins, and the IIC_1 I/O pin. The output type of P42 is NMOS push-pull output. The output type of SDA1 is NMOS open-drain output. Port 4 has the following registers.

- Port 4 data direction register (P4DDR)
- Port 4 data register (P4DR)

7.5.1 Port 4 Data Direction Register (P4DDR)

P4DDR specifies input or output for the pins of port 4 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DDR	0	W	When a bit in P4DDR is set to 1, the corresponding pin functions as an output port, and when cleared to 0, as an input port. As 14-bit PWM is initialized in software standby mode, the pin states are determined by the TMR_0, TMR_1, IIC_1, P4DDR, and P4DR specifications.
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	

7.5.2 Port 4 Data Register (P4DR)

P4DR stores output data for port 4.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DR	0	R/W	If a port 4 read is performed while P4DDR bits are set to 1, the P4DR values are read directly, regardless of the actual pin states. If a port 4 read is performed while P4DDR bits are cleared to 0, the pin states are read.
6	P46DR	0	R/W	
5	P45DR	0	R/W	
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	

7.5.3 Pin Functions

- P47/PWX1

  The pin function is switched as shown below according to the combination of the OEB bit in DACR of the 14-bit PWM and the P47DDR bit.

OEB	0		1
P47DDR	0	1	—
Pin Function	P47 input pin	P47 output pin	PWX1 output pin

- P46/PWX0

  The pin function is switched as shown below according to the combination of the OEA bit in DACR of the 14-bit PWM and the P46DDR bit.

OEA	0		1
P46DDR	0	1	—
Pin Function	P46 input pin	P46 output pin	PWX0 output pin

- P45/TMRI1

  The pin function is switched as shown below according to the status of the P45DDR bit.

P45DDR	0	1
Pin Function	P45 input pin	P45 output pin
	TMRI1 input pin	

Note:* When bits CCLR1 and CCLR0 in TCR1 of TMR_1 are set to 1, this pin is used as the TMRI1 input pin.

- P44/TMO1

  The pin function is switched as shown below according to the combination of the OS3 to OS0 bits in TCSR of TMR_1 and the P44DDR bit.

OS3 to OS0	All 0		Not all 0
P44DDR	0	1	—
Pin Function	P44 input pin	P44 output pin	TMO1 output pin

- P43/TMCI1

  The pin function is switched as shown below according to the status of the P43DDR bit.

P43DDR	0	1
Pin Function	P43 input pin	P43 output pin
	TMCI1 input pin*	

Note:* When the external clock is selected by bits CKS2 to CKS0 in TCR1 of TMR_1, this pin is used as the TMCI1 input pin.

- P42/TMRI0/SDA1

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC_1 and the P42DDR bit.

ICE	0		1
P42DDR	0	1	—
Pin Function	P42 input pin	P42 output pin	SDA1 I/O pin
	TMRI0 input pin*		

Note:* SDA1 is an NMOS-only output, and has direct bus drive capability.

When bits CCLR1 and CCLR0 in TCR0 of TMR_0 are set to 1, this pin is used as the TMRI0 input pin.

When the P42 output pin is set, the output type is NMOS push-pull output.

- P41/TMO0

  The pin function is switched as shown below according to the combination of the OS3 to OS0 bits in TCSR of TMR_0 and the P41DDR bit.

OS3 to OS0	All 0		Not all 0
P41DDR	0	1	—
Pin Function	P41 input pin	P41 output pin	TMO0 output pin

- P40/TMCI0

  The pin function is switched as shown below according to the status of the P40DDR bit.

P40DDR	0	1
Pin Function	P40 input pin	P40 output pin
	TMCI0 input pin*	

Note:* When an external clock is selected with bits CKS2 to CKS0 in TCR0 of TMR_0, this pin is used as the TMCI0 input pin.

7.6 Port 5

Port 5 is a 3-bit I/O port. Port 5 pins also function as SCI_1 extended I/O pins, and the IIC_0 I/O pin. P52 and ExSCK1 are NMOS push-pull outputs, and SCL0 is an NMOS open-drain output. Port 5 has the following registers.

- Port 5 data direction register (P5DDR)
- Port 5 data register (P5DR)

7.6.1 Port 5 Data Direction Register (P5DDR)

P5DDR specifies input or output for the pins of port 5 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 1	—	Reserved The initial value should not be changed.
2	P52DDR	0	W	The corresponding port 5 pins are output ports when P5DDR bits are set to 1, and input ports when cleared to 0. As SCI_1 is initialized in software standby mode, the pin states are determined by the IIC_0 ICCR, P5DDR, and P5DR specifications.
1	P51DDR	0	W	
0	P50DDR	0	W	

7.6.2 Port 5 Data Register (P5DR)

P5DR stores output data for port 5 pins.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 1	—	Reserved The initial value should not be changed.
2	P52DR	0	R/W	If a port 5 read is performed while P5DDR bits are set to 1, the P5DR values are read directly, regardless of the actual pin states. If a port 5 read is performed while P5DDR bits are cleared to 0, the pin states are read.
1	P51DR	0	R/W	
0	P50DR	0	R/W	

7.6.3 Pin Functions

- P52/ExSCK1*/SCL0

  The pin function is switched as shown below according to the combination of the CKE1 and CKE0 bits in SCR, the $C/\overline{A}$ bit in SMR of SCI_1, the SPS1 bit* in SPSR, the ICE bit in ICCR of IIC_0, and the P52DDR bit.

<table>
<tr><td>SPS1*</td><td colspan="3">0</td><td colspan="6">1</td></tr>
<tr><td>ICE</td><td colspan="2">0</td><td>1</td><td colspan="5">0</td><td>1</td></tr>
<tr><td>CKE1</td><td>—</td><td>—</td><td>—</td><td colspan="4">0</td><td>1</td><td>0</td></tr>
<tr><td>$C/\overline{A}$</td><td>—</td><td>—</td><td>—</td><td colspan="3">0</td><td>1</td><td>—</td><td>0</td></tr>
<tr><td>CKE0</td><td>—</td><td>—</td><td>—</td><td colspan="2">0</td><td>1</td><td>—</td><td>—</td><td>0</td></tr>
<tr><td>P52DDR</td><td>0</td><td>1</td><td>—</td><td>0</td><td>1</td><td>—</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td>Pin Function</td><td>P52 input pin</td><td>P52 output pin</td><td>SCL0 I/O pin</td><td>P52 input pin</td><td>P52 output pin</td><td>ExSCK1* output pin</td><td>ExSCK1* output pin</td><td>ExSCK1 input pin</td><td>SCL0 I/O pin</td></tr>
</table>

Note:* When this pin is used as the SCL0 I/O pin by setting 1 to the SPS1 bit of SPSR, bits CKE1 and CKE0 in SCR of SCI_1 and bit $C/\overline{A}$ in SMR must all be cleared to 0. SCL0 is an NMOS open-drain output.

When set as the P52 output pin or ExSCK1 output pin, this pin is an NMOS push-pull output.

The program development tool (emulator) does not support this function.

- P51/ExRxD1*

  The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI_1, the SPS1 bit* in SPSR, and the P51DDR bit.

SPS1*	0		1		
RE	—		0		1
P51DDR	0	1	0	1	—
Pin Function	P51 input pin	P51 output pin	P51 input pin	P51 output pin	ExRxD1 input pin*

Note:* The program development tool (emulator) does not support this function.

- P50/ExTxD1*

  The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI_1, the SPS1 bit* in SPSR, and the P50DDR bit.

SPS1*	0		1		
TE	—		0		1
P50DDR	0	1	0	1	—
Pin Function	P50 input pin	P50 output pin	P50 input pin	P50 output pin	ExTxD1 output pin*

Note:* The program development tool (emulator) does not support this function.

7.7 Port 6

Port 6 is an 8-bit I/O port. Port 6 pins also function as the FRT I/O pins, TMR_X I/O pins, TMR_Y input pin, key-sense interrupt input pins, and interrupt input pins. The port 6 input level can be switched in four stages. Port 6 has the following registers.

- Port 6 data direction register (P6DDR)
- Port 6 data register (P6DR)
- Port 6 pull-up MOS control register (KMPCR)
- System control register 2 (SYSCR2)

7.7.1 Port 6 Data Direction Register (P6DDR)

P6DDR specifies input or output for the pins of port 6 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P67DDR	0	W	The corresponding port 6 pins are output ports when P6DDR bits are set to 1, and input ports when cleared to 0.
6	P66DDR	0	W	
5	P65DDR	0	W	
4	P64DDR	0	W	
3	P63DDR	0	W	
2	P62DDR	0	W	
1	P61DDR	0	W	
0	P60DDR	0	W	

7.7.2 Port 6 Data Register (P6DR)

P6DR stores output data for port 6.

Bit	Bit Name	Initial Value	R/W	Description
7	P67DR	0	R/W	If a port 6 read is performed while P6DDR bits are set to 1, the P6DR values are read directly, regardless of the actual pin states. If a port 6 read is performed while P6DDR bits are cleared to 0, the pin states are read.
6	P66DR	0	R/W	
5	P65DR	0	R/W	
4	P64DR	0	R/W	
3	P63DR	0	R/W	
2	P62DR	0	R/W	
1	P61DR	0	R/W	
0	P60DR	0	R/W	

7.7.3 Port 6 Pull-Up MOS Control Register (KMPCR)

KMPCR controls the port 6 on-chip input pull-up MOSs on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	KM7PCR	0	R/W	The input pull-up MOS is turned on when a KMPCR bit is set to 1 with the input port setting.
6	KM6PCR	0	R/W	
5	KM5PCR	0	R/W	
4	KM4PCR	0	R/W	
3	KM3PCR	0	R/W	
2	KM2PCR	0	R/W	
1	KM1PCR	0	R/W	
0	KM0PCR	0	R/W	

7.7.4 System Control Register 2 (SYSCR2)

SYSCR2 controls the port 6 operations.

Bit	Bit Name	Initial Value	R/W	Description
7	KWUL1	0	R/W	Key wakeup levels 1 and 0
6	KWUL0	0	R/W	Sets the input level of port 6. The input level of pins functioning as port 6 is also changed. 00: Standard input level 01: Input level 1 10: Input level 2 11: Input level 3
5	P6PUE	0	R/W	Port 6 input pull-up MOS extra Selects the power specifications of the input pull-up MOS for port 6. 0: Standard power specifications 1: Restricted power specifications
4	—	0	R/W	Reserved The initial value should not be changed.
3 to 1	—	All 0	R/W	Reserved The initial value should not be changed.
0	HI12E	0	R/W	Host interface enabled The initial value should not be changed.

7.7.5 Pin Functions

- P67/TMOX/$\overline{\text{KIN7}}$/$\overline{\text{IRQ7}}$

The pin function is switched as shown below according to the combination of the OS3 to OS0 bits in TCSR of TMR_X, the OSX bit*[2] in TCRXY, and the P67DDR bit.

OSX*[2]	0			1	
OS3 to OS0	All 0		Not all 0	—	
P67DDR	0	1	—	0	1
Pin Function	P67 input pin	P67 output pin	TMOX output pin	P67 input pin	P67 output pin
	$\overline{\text{IRQ7}}$ input pin, $\overline{\text{KIN7}}$ input pin*[1]				

Notes: 1. This pin is used as the $\overline{\text{IRQ7}}$ input pin when bit IRQ7E is set to 1 in IER. It can always be used as the $\overline{\text{KIN7}}$ input pin.

2. The program development tool (emulator) does not support this function.

- P66/FTOB/$\overline{\text{KIN6}}$/$\overline{\text{IRQ6}}$

The pin function is switched as shown below according to the combination of the OEB bit in TOCR of the FRT and the P66DDR bit.

OEB	0		1
P66DDR	0	1	—
Pin Function	P66 input pin	P66 output pin	FTOB output pin
	$\overline{\text{IRQ6}}$ input pin, $\overline{\text{KIN6}}$ input pin*		

Note:* This pin is used as the $\overline{\text{IRQ6}}$ input pin when bit IRQ6E is set to 1 in IER while the KMIMR6 bit in KMIMR is 0. It can always be used as the $\overline{\text{KIN6}}$ input pin.

- P65/FTID/$\overline{\text{KIN5}}$

P65DDR	0	1
Pin Function	P65 input pin	P65 output pin
	FTID input pin, $\overline{\text{KIN5}}$ input pin*	

Note:* This pin can always be used as the FTID or $\overline{\text{KIN5}}$ input pin.

- P64/FTIC/$\overline{\text{KIN4}}$

  The pin function is switched as shown below according to the status of the P64DDR bit.

P64DDR	0	1
Pin Function	P64 input pin	P64 output pin
	FTIC input pin, $\overline{\text{KIN4}}$ input pin*	

Note:* This pin can always be used as the FTIC or $\overline{\text{KIN4}}$ input pin.

- P63/FTIB/$\overline{\text{KIN3}}$

P63DDR	0	1
Pin Function	P63 input pin	P63 output pin
	FTIB input pin, $\overline{\text{KIN3}}$ input pin*	

Note:* This pin can always be used as the FTIB or $\overline{\text{KIN3}}$ input pin.

- P62/FTIA/$\overline{\text{KIN2}}$/TMIY

P62DDR	0	1
Pin Function	P62 input pin	P62 output pin
	FTIA input pin, TMIY input pin, $\overline{\text{KIN2}}$ input pin*	

Note:* This pin can always be used as the FTIA, TMIY, or $\overline{\text{KIN2}}$ input pin.

- P61/FTOA/$\overline{\text{KIN1}}$

  The pin function is switched as shown below according to the combination of the OEA bit in TOCR of the FRT, and the P61DDR bit.

OEA	0		1
P61DDR	0	1	—
Pin Function	P61 input pin	P61 output pin	FTOA input pin
	$\overline{\text{KIN1}}$ input pin*		

Note:* This pin can always be used as the $\overline{\text{KIN1}}$ input pin.

- P60/FTCI/$\overline{\text{KIN0}}$/TMIX

P60DDR	0	1
Pin Function	P60 input pin	P60 output pin
	FTCI input pin, TMIX input pin, $\overline{\text{KIN0}}$ input pin*	

Note:* This pin is used as the FTCI input pin when an external clock is selected with bits CKS1 and CKS0 in TCR of the FRT. It can always be used as the TMIX or $\overline{\text{KIN0}}$ input pin.

7.7.6 Port 6 Input Pull-Up MOS

Port 6 has an on-chip input pull-up MOS function that can be controlled by software. This input pull-up MOS function can be specified as on or off on a bit-by-bit basis.

The input pull-up MOS current specification can be changed by means of the P6PUE bit. When a pin is designated as an on-chip peripheral module output pin, the input pull-up MOS is always off.

Table 7.5 summarizes the input pull-up MOS states.

Table 7.5 Input Pull-Up MOS States (Port 6)

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

Legend:

Off: Input pull-up MOS is always off.

On/Off: On when the pin is in the input state, P6DDR = 0, and KMPCR = 1; otherwise off.

7.8 Port 7

Port 7 is an 8-bit I/O port*[1]. Port 7 pins also function as the TMR_X extended output pins (ExTMOX)*[2] and TMR_Y output pins (TMOY)*[2]. Port 7 has the following registers.

- Port 7 input data register (P7PIN)
- Port 7 data direction register (P7DDR)*[2]
- Port 7 output data register (P7ODR)*[2]

Notes: 1. The program development tool (emulator) does not support the output.

2. The program development tool (emulator) does not support this function.

7.8.1 Port 7 Input Data Register (P7PIN)

P7PIN reflects the pin states of port 7.

Bit	Bit Name	Initial Value	R/W	Description
7	P77PIN	Undefined*	R	When a P7PIN read is performed, the pin states are always read. P7PIN has the same address as PBDDR; if a write is performed, data will be written into PBDDR and the port B setting will be changed.
6	P76PIN	Undefined*	R	
5	P75PIN	Undefined*	R	
4	P74PIN	Undefined*	R	
3	P73PIN	Undefined*	R	
2	P72PIN	Undefined*	R	
1	P71PIN	Undefined*	R	
0	P70PIN	Undefined*	R	

Note:* Determined by the pin states of P77 to P70.

7.8.2 Port 7 Data Direction Register (P7DDR)

P7DDR specifies input or output for the pins of port 7 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P77DDR	0	W	The corresponding port 7 pins are output ports when P7DDR bits are set to 1, and input ports when P7DDR bits are cleared to 0.
6	P76DDR	0	W	
5	P75DDR	0	W	
4	P74DDR	0	W	
3	P73DDR	0	W	
2	P72DDR	0	W	
1	P71DDR	0	W	
0	P70DDR	0	W	

Note: The program development tool (emulator) does not support this register.

7.8.3 Port 7 Output Data Register (P7ODR)

P7ODR stores output for the pins of port 7.

Bit	Bit Name	Initial Value	R/W	Description
7	P77ODR	0	R/W	P7ODR can always be read or written to, regardless of the contents of P7DDR.
6	P76ODR	0	R/W	
5	P75ODR	0	R/W	
4	P74ODR	0	R/W	
3	P73ODR	0	R/W	
2	P72ODR	0	R/W	
1	P71ODR	0	R/W	
0	P70ODR	0	R/W	

Note: The program development tool (emulator) does not support this register.

7.8.4 Pin Functions

- P77/ExTMOX*

  The pin function is switched as shown below according to the combination of the OSX bit* in TCRXY of TMR_X, OS3 to OS0 bits, and P77DDR*.

OSX*	0		1		
OS3 to OS0	—		All 0		Not all 0
P77DDR*	0	1	0	1	—
Pin Function	P77 input pin	P77 output pin*	P77 input pin	P77 output pin*	ExTMOX output pin*

Note:* The program development tool (emulator) does not support this function.

- P76/TMOY*

  The pin function is switched as shown below according to the combination of the OEY bit* in TCRXY of TMR_X, OS3 to OS0 bits, and P76DDR*.

OEY*	0		1		
OS3 to OS0	—		All 0		Not all 0
P76DDR*	0	1	0	1	—
Pin Function	P76 input pin	P76 output pin*	P76 input pin	P76 output pin*	TMOY output pin*

Note:* The program development tool (emulator) does not support this function.

- P75, P74, P73, P72, P71, P70

  The pin function is switched as shown below according to the status of P7nDDR*.

P7nDDR*	0	1
Pin Function	P7n input pin	P7n output pin*

Note:* The program development tool (emulator) does not support this function. (n = 5 to 0)

7.9 Port 8

Port 8 is an 8-bit I/O port. Port 8 pins also function as SCI_1 I/O pins, the IIC_1 I/O pin, LPC I/O pins, and interrupt input pins. The output type of P86 and SCK1 is NMOS push-pull output. The output type of SCL1 is NMOS open-drain output and direct bus driving is enabled. Port 8 has the following registers.

- Port 8 data direction register (P8DDR)
- Port 8 data register (P8DR)

7.9.1 Port 8 Data Direction Register (P8DDR)

P8DDR specifies input or output for the pins of port 8 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	—	Reserved The initial value should not be changed.
6	P86DDR	0	W	P8DDR has the same address as PBPIN, and if read, the port B state will be returned. The corresponding port 8 pins are output ports when P8DDR bits are set to 1, and input ports when cleared to 0.
5	P85DDR	0	W	
4	P84DDR	0	W	
3	P83DDR	0	W	
2	P82DDR	0	W	
1	P81DDR	0	W	
0	P80DDR	0	W	

7.9.2 Port 8 Data Register (P8DR)

P8DR stores output data for the port 8 pins (P86 to P80).

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	—	Reserved The initial value should not be changed.
6	P86DR	0	R/W	If a port 8 read is performed while P8DDR bits are set to 1, the P8DR values are read directly, regardless of the actual pin states. If a port 8 read is performed while P8DDR bits are cleared to 0, the pin states are read.
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	

7.9.3 Pin Functions

- P86/$\overline{\text{IRQ5}}$/ SCK1/SCL1

  The pin function is switched as shown below according to the combination of the CKE1 and CKE0 bits in SCR of SCI_1, the C/$\overline{\text{A}}$ bit in SMR of SCI_1, the SPS1 bit*[2] in SPSR, the ICE bit in ICCR of IIC_1, and the P86DDR bit.

SPS1*[2]	0						1		
ICE	0					1	0		1
CKE1	0				1	0	—	—	—
C/$\overline{\text{A}}$	0			1	—	0	—	—	—
CKE0	0		1	—	—	0	—	—	—
P86DDR	0	1	—	—	—	—	0	1	—
Pin Function	P86 input pin	P86 output pin	SCK1 output pin	SCK1 output pin	SCK1 input pin	SCL1 I/O pin	P86 input pin	P86 output pin	SCL1 I/O pin
	$\overline{\text{IRQ}}$ input pin*[1]								

Notes: 1. When the IRQ5E bit in IER is set to 1, this pin is used as the $\overline{\text{IRQ5}}$ input pin. When this pin is used as the SCL1 I/O pin, bits CKE1 and CKE0 in SCR of SCI_1 and bit C/$\overline{\text{A}}$ in SMR of SCI_1 must all be cleared to 0. When the P86 output pin and SCK1 output pin are set, the output type is NMOS push-pull output. SCL1 is an NMOS-only output, and has direct bus drive capability.

2. The program development tool (emulator) does not support this function.

- P85/$\overline{\text{IRQ4}}$/RxD1

  The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI_1, the SPS1 bit*[2] in SPSR, and the P85DDR bit.

SPS1*[2]	0			1	
RE	0		1	—	
P85DDR	0	1	—	0	1
Pin Function	P85 input pin	P85 output pin	RxD1 input pin	P85 input pin	P85 output pin
	$\overline{\text{IRQ4}}$ input pin*[1]				

Notes: 1. When the IRQ4E bit in IER is set to 1, this pin is used as the $\overline{\text{IRQ4}}$ input pin.

2. The program development tool (emulator) does not support this function.

- P84/$\overline{\text{IRQ3}}$/TxD1

  The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI_1, the SPS1 bit*[2] in SPSR, and the P84DDR bit.

SPS1*[2]	0			1	
TE	0		1	—	
P84DDR	0	1	—	0	1
Pin Function	P84 input pin	P84 output pin	TxD1 output pin	P84 input pin	P84 output pin
	$\overline{\text{IRQ3}}$ input pin*[1]				

Notes: 1. When the IRQ3E bit in IER is set to 1, this pin is used as the $\overline{\text{IRQ3}}$ input pin.

2. The program development tool (emulator) does not support this function.

- P83/$\overline{\text{LPCPD}}$

  The pin function is switched as shown below according to the status of the P83DDR bit.

P83DDR	0	1
Pin Function	P83 input pin	P83 output pin
	$\overline{\text{LPCPD}}$ input pin*	

Note:* When at least one of bits LPC3E to LPC1E is set to 1 in HICR0, this pin is used as the $\overline{\text{LPCPD}}$ input pin.

- P82/$\overline{\text{CLKRUN}}$

  The pin function is switched as shown below according to the combination of the HI12E bit in SYSCR2, the LPC3E to LPC1E bits in HICR0, and the P82DDR bit.

LPC3E to LPC1E	All 0		Not all 0
HI12E	0		0*
P82DDR	0	1	0*
Pin Function	P82 input pin	P82 output pin	$\overline{\text{CLKRUN}}$ I/O pin

Note:* When at least one of bits LPC3E to LPC1E is set to 1, bits HI12E and P82DDR should be cleared to 0.

- P81/GA20

  The pin function is switched as shown below according to the combination of the HI12E bit in SYSCR2, the FGA20E bit in HICR0, and the P81DDR bit.

FGA20E	0		1
HI12E	0		0*
P81DDR	0	1	0*
Pin Function	P81 input pin	P81 output pin	GA20 output pin
	GA20 input pin		

Note:* When bit FGA20E is set to 1 in HICR0, bits HI12E and P81DDR should be cleared to 0.

- P80/$\overline{\text{PME}}$

  The pin function is switched as shown below according to the combination of the HI12E bit in SYSCR2, the PMEE bit in HICR0, and the P80DDR bit.

PMEE	0		1
HI12E	0		0*
P80DDR	0	1	0*
Pin Function	P80 input pin	P80 output pin	$\overline{\text{PME}}$ output pin
	$\overline{\text{PME}}$ input pin		

Note:* When bit PMEE is set to 1 in HICR0, bits HI12E and P80DDR should be cleared to 0.

7.10 Port 9

Port 9 is an 8-bit I/O port. Port 9 pins also function as the interrupt input pins, IIC_0 I/O pin, subclock input pin, and system clock (ϕ) output pin. P97 is an NMOS push-pull output. SDA0 is an NMOS open-drain output, and has direct bus drive capability. Port 9 has the following registers.

- Port 9 data direction register (P9DDR)
- Port 9 data register (P9DR)

7.10.1 Port 9 Data Direction Register (P9DDR)

P9DDR specifies input or output for the pins of port 9 on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	P97DDR	0	W	When the corresponding P9DDR bits are set to 1, pin P96 functions as the ϕ output pin and pins P97 and P95 to P90 become output ports. When P9DDR bits are cleared to 0, the corresponding pins become input ports.
6	P96DDR	0	W	
5	P95DDR	0	W	
4	P94DDR	0	W	
3	P93DDR	0	W	
2	P92DDR	0	W	
1	P91DDR	0	W	
0	P90DDR	0	W	

7.10.2 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	With the exception of P96, if a port 9 read is performed while P9DDR bits are set to 1, the P9DR values are read directly, regardless of the actual pin states. If a port 9 read is performed while P9DDR bits are cleared to 0, the pin states are read. For P96, the pin state is always read.
6	P96DR	Undefined*	R	
5	P95DR	0	R/W	
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.

7.10.3 Pin Functions

- P97/SDA0

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC_0 and the P97DDR bit.

ICE	0		1
P97DDR	0	1	—
Pin Function	P97 input pin	P97 output pin	SDA0 I/O pin

Note:* When this pin is set as the P97 output pin, it is an NMOS push-pull output. SDA0 is an NMOS open-drain output, and has direct bus drive capability.

- P96/ϕ/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	ϕ output pin

Note:* When this pin is used as the EXCL input pin, P96DDR should be cleared to 0.

- P95

  The pin function is switched as shown below according to the status of the P95DDR bit.

P95DDR	0	1
Pin Function	P95 input pin	P95 output pin

- P94

  The pin function is switched as shown below according to the status of the P94DDR bit.

P94DDR	0	1
Pin Function	P94 input pin	P94 output pin

- P93

  The pin function is switched as shown below according to the status of the P93DDR bit.

P93DDR	0	1
Pin Function	P93 input pin	P93 output pin

- P92/$\overline{\text{IRQ0}}$

  The pin function is switched as shown below according to the status of the P92DDR bit.

P92DDR	0	1
Pin Function	P92 input pin	P92 output pin
	$\overline{\text{IRQ0}}$ input pin*	

Note:* When bit IRQ0E in IER is set to 1, this pin is used as the $\overline{\text{IRQ0}}$ input pin.

- P91/$\overline{\text{IRQ1}}$

  The pin function is switched as shown below according to the status of the P91DDR bit.

P91DDR	0	1
Pin Function	P91 input pin	P91 output pin
	$\overline{\text{IRQ1}}$ input pin*	

Note:* When bit IRQ1E in IER is set to 1, this pin is used as the $\overline{\text{IRQ1}}$ input pin.

- P90/$\overline{\text{IRQ2}}$

  The pin function is switched as shown below according to the status of the P90DDR bit.

P90DDR	0	1
Pin Function	P90 input pin	P90 output pin
	$\overline{\text{IRQ2}}$ input pin*	

Note:* When the IRQ2E bit in IER is set to 1, this pin is used as the $\overline{\text{IRQ2}}$ input pin.

7.11 Port A

Port A is an 8-bit I/O port. Port A pins also function as keyboard buffer controller I/O pins, and key-sense interrupt input pins. Port A input/output operates by VccB power independent from the Vcc power. Up to 5 V can be applied to port A pins if VccB power is 5 V. 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)

7.11.1 Port A Data Direction Register (PADDR)

PADDR specifies input or output for the pins of port A on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	PA7DDR	0	W	The corresponding port A pins are output ports when PADDR bits are set to 1, and input ports when cleared to 0. PA7 to PA2 pins are used as the keyboard buffer controller I/O pins by setting the KBIOE bit to 1, while the I/O direction according to PA7DDR to PA2DDR is ignored. PADDR has the same address as PAPIN, if read, port A status is returned.
6	PA6DDR	0	W	
5	PA5DDR	0	W	
4	PA4DDR	0	W	
3	PA3DDR	0	W	
2	PA2DDR	0	W	
1	PA1DDR	0	W	
0	PA0DDR	0	W	

7.11.2 Port A Output Data Register (PAODR)

PAODR stores output data for port A.

Bit	Bit Name	Initial Value	R/W	Description
7	PA7ODR	0	R/W	PAODR can always be read or written to, regardless of the contents of PADDR.
6	PA6ODR	0	R/W	
5	PA5ODR	0	R/W	
4	PA4ODR	0	R/W	
3	PA3ODR	0	R/W	
2	PA2ODR	0	R/W	
1	PA1ODR	0	R/W	
0	PA0ODR	0	R/W	

7.11.3 Port A Input Data Register (PAPIN)

PAPIN indicates the port A state.

Bit	Bit Name	Initial Value	R/W	Description
7	PA7PIN	Undefined*	R	Reading PAPIN always returns the pin states. PAPIN has the same address as PADDR. If a write is performed, the port A settings will change.
6	PA6PIN	Undefined*	R	
5	PA5PIN	Undefined*	R	
4	PA4PIN	Undefined*	R	
3	PA3PIN	Undefined*	R	
2	PA2PIN	Undefined*	R	
1	PA1PIN	Undefined*	R	
0	PA0PIN	Undefined*	R	

Note:* The initial value is determined according to the PA7 to PA0 pin states.

7.11.4 Pin Functions

- PA7/$\overline{\text{KIN15}}$/PS2CD

  The pin function is switched as shown below according to the combination of the KBIOE bit in KBCRH_2 of the keyboard buffer controller, and the PA7DDR bit.

KBIOE	0		1
PA7DDR	0	1	—
Pin Function	PA7 input pin	PA7 output pin	PS2CD output pin
	$\overline{\text{KIN15}}$ input pin, PS2CD input pin*		

Note:* When the KBIOE bit is set to 1 or the IICS bit in STCR is set to 1, this pin is an NMOS open-drain output, and has direct bus drive capability. This pin can always be used as the PS2CD or $\overline{\text{KIN15}}$ input pin.

- PA6/$\overline{\text{KIN14}}$/PS2CC

  The pin function is switched as shown below according to the combination of the KBIOE bit in KBCRH_2 of the keyboard buffer controller, and the PA6DDR bit.

KBIOE	0		1
PA6DDR	0	1	—
Pin Function	PA6 input pin	PA6 output pin	PS2CC output pin
	$\overline{\text{KIN14}}$ input pin, PS2CC input pin*		

Note:* When the KBIOE bit is set to 1 or the IICS bit in STCR is set to 1, this pin is an NMOS open-drain output, and has direct bus drive capability. This pin can always be used as the PS2CC or $\overline{\text{KIN14}}$ input pin.

- PA5/$\overline{\text{KIN13}}$/PS2BD

  The pin function is switched as shown below according to the combination of the KBIOE bit in KBCRH_1 of the keyboard buffer controller, and the PA5DDR bit.

KBIOE	0		1
PA5DDR	0	1	—
Pin Function	PA5 input pin	PA5 output pin	PS2BD output pin
	$\overline{\text{KIN13}}$ input pin, PS2BD input pin*		

Note:* When the KBIOE bit is set to 1 or the IICS bit in STCR is set to 1, this pin is an NMOS open-drain output, and has direct bus drive capability. This pin can always be used as the PS2BD or $\overline{\text{KIN13}}$ input pin.

- PA4/$\overline{\text{KIN12}}$/PS2BC

  The pin function is switched as shown below according to the combination of the KBIOE bit in KBCRH_1 of the keyboard buffer controller, and the PA4DDR bit.

KBIOE	0		1
PA4DDR	0	1	—
Pin Function	PA4 input pin	PA4 output pin	PS2BC output pin
	$\overline{\text{KIN12}}$ input pin, PS2BC input pin*		

Note:* When the KBIOE bit is set to 1 or the IICS bit in STCR is set to 1, this pin is an NMOS open-drain output, and has direct bus drive capability. This pin can always be used as the PS2BC or $\overline{\text{KIN12}}$ input pin.

- PA3/$\overline{\text{KIN11}}$/PS2AD

  The pin function is switched as shown below according to the combination of the KBIOE bit in KBCRH_0 of the keyboard buffer controller, and the PA3DDR bit.

KBIOE	0		1
PA3DDR	0	1	—
Pin Function	PA3 input pin	PA3 output pin	PS2AD output pin
	$\overline{\text{KIN11}}$ input pin, PS2AD input pin*		

Note:* When the KBIOE bit is set to 1, this pin is an NMOS open-drain output, and has direct bus drive capability. This pin can always be used as the PS2AD or $\overline{\text{KIN11}}$ input pin.

- PA2/$\overline{\text{KIN10}}$/PS2AC

  The pin function is switched as shown below according to the combination of the KBIOE bit in KBCRH_0 of the keyboard buffer controller, and the PA2DDR bit.

KBIOE	0		1
PA2DDR	0	1	—
Pin Function	PA2 input pin	PA2 output pin	PS2AC output pin
	$\overline{\text{KIN10}}$ input pin, PS2AC input pin*		

Note:* When the KBIOE bit is set to 1, this pin is an NMOS open-drain output, and has direct bus drive capability. This pin can always be used as the PS2AC or $\overline{\text{KIN10}}$ input pin.

- PA1/$\overline{\text{KIN9}}$, PA0/$\overline{\text{KIN8}}$

  The pin function is switched as shown below according to the status of the PAnDDR bit.

PAnDDR	0	1
Pin Function	PAn input pin	PAn output pin
	$\overline{\text{KINm}}$ input pin*	

Note:* This pin can always be used as the $\overline{\text{KINm}}$ input pin. (n = 1 or 0, m = 9 or 8)

7.11.5 Port A Input Pull-Up MOS

Port A has an on-chip input pull-up MOS function that can be controlled by software. This input pull-up MOS function can be specified as on or off on a bit-by-bit basis.

The input pull-up MOS for pins PA7 to PA4 is always off when IICS is set to 1. When the keyboard buffer control pin function is selected for pins PA7 to PA2, the input pull-up MOS is always off.

Table 7.6 summarizes the input pull-up MOS states.

Table 7.6 Input Pull-Up MOS States (Port A)

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

Legend

Off: Input pull-up MOS is always off.

On/Off: On when the pin is in the input state, PADDR = 0, and PAODR = 1; otherwise off.

7.12 Port B

Port B is an 8-bit I/O port. Port B pins also have LPC input/output pins, and wakeup event interrupt input pins function. Port B has the following registers.

- Port B data direction register (PBDDR)
- Port B output data register (PBODR)
- Port B input data register (PBPIN)

7.12.1 Port B Data Direction Register (PBDDR)

PBDDR specifies input or output for the pins of port B on a bit-by-bit basis.

Bit	Bit Name	Initial Value	R/W	Description
7	PB7DDR	0	W	PBDDR has the same address as P7PIN, and if read, the port 7 pin states will be returned. A port B pin becomes an output port if the corresponding PBDDR bit is set to 1, and an input port if the bit is cleared to 0.
6	PB6DDR	0	W	
5	PB5DDR	0	W	
4	PB4DDR	0	W	
3	PB3DDR	0	W	
2	PB2DDR	0	W	
1	PB1DDR	0	W	
0	PB0DDR	0	W	

7.12.2 Port B Output Data Register (PBODR)

PBODR stores output data for port B.

Bit	Bit Name	Initial Value	R/W	Description
7	PB7ODR	0	R/W	PBODR can always be read or written to, regardless of the contents of PBDDR.
6	PB6ODR	0	R/W	
5	PB5ODR	0	R/W	
4	PB4ODR	0	R/W	
3	PB3ODR	0	R/W	
2	PB2ODR	0	R/W	
1	PB1ODR	0	R/W	
0	PB0ODR	0	R/W	

7.12.3 Port B Input Data Register (PBPIN)

PBPIN indicates the port B state.

Bit	Bit Name	Initial Value	R/W	Description
7	PB7PIN	Undefined*	R	Reading PBPIN always returns the pin states. PBPIN has the same address as P8DDR. If a write is performed, data will be written to P8DDR and the port 8 settings will change.
6	PB6PIN	Undefined*	R	
5	PB5PIN	Undefined*	R	
4	PB4PIN	Undefined*	R	
3	PB3PIN	Undefined*	R	
2	PB2PIN	Undefined*	R	
1	PB1PIN	Undefined*	R	
0	PB0PIN	Undefined*	R	

Note:* The initial value is determined according to the PB7 to PB0 pin states.

7.12.4 Pin Functions

- PB7/$\overline{\text{WUE7}}$, PB6/$\overline{\text{WUE6}}$, PB5/$\overline{\text{WUE5}}$, PB4/$\overline{\text{WUE4}}$, PB3/$\overline{\text{WUE3}}$, PB2/$\overline{\text{WUE2}}$

  The pin function is switched as shown below according to the status of the PBnDDR bit.

PBnDDR	0	1
Pin Function	PBn input pin	PBn output pin
	$\overline{\text{WUEn}}$ input pin*	

Note:* This pin can always be used as the $\overline{\text{WUEn}}$ input pin. (n = 7 to 2)

- PB1/$\overline{\text{WUE1}}$/LSCI

  The pin function is switched as shown below according to the combination of the HI12E bit in SYSCR2, the LSCIE bits in HICR0 of host interface (LPC), and the PB1DDR bit.

LSCIE	0		1
HI12E	0		0*^{1}
PB1DDR	0	1	0*^{1}
Pin Function	PB1input pin	PB1 output pin	LSCI output pin
	$\overline{\text{WUE1}}$ input pin*^{2}, LSCI input pin*^{2}		

Notes: 1. When bit LSCIE is set to 1 in HICR0, bits HI12E and PB1DDR should be cleared to 0.

2. This pin can always be used as the $\overline{\text{WUE1}}$ or LSCI input pin.

- PB0/$\overline{\text{WUE0}}$/$\overline{\text{LSMI}}$

  The pin function is switched as shown below according to the combination of the HI12E bit in SYSCR2, the LSMIE bits in HICR0 of host interface (LPC), and the PB0DDR bit.

LSMIE	0		1
HI12E	0		0*^{1}
PB0DDR	0	1	0*^{1}
Pin Function	PB0 input pin	PB0 output pin	$\overline{\text{LSMI}}$ output pin
	$\overline{\text{WUE0}}$ input pin*^{2}, $\overline{\text{LSMI}}$ input pin*^{2}		

Notes: 1. When bit LSMIE is set to 1 in HICR0, bits HI12E and PB0DDR should be cleared to 0.

2. This pin can always be used as the $\overline{\text{WUE0}}$ or $\overline{\text{LSMI}}$ input pin.

7.12.5 Port B Input Pull-Up MOS

Port B has an on-chip input pull-up MOS function that can be controlled by software. This input pull-up MOS function can be specified as on or off on a bit-by-bit basis.

When a pin is designated as an on-chip peripheral module output pin, the input pull-up MOS is always off.

Table 7.7 summarizes the input pull-up MOS states.

Table 7.7 Input Pull-Up MOS States (Port B)

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

Legend

Off: Input pull-up MOS is always off.

On/Off: On when the pin is in the input state, PBDDR = 0, and PBODR = 1; otherwise off.

Section 8 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.

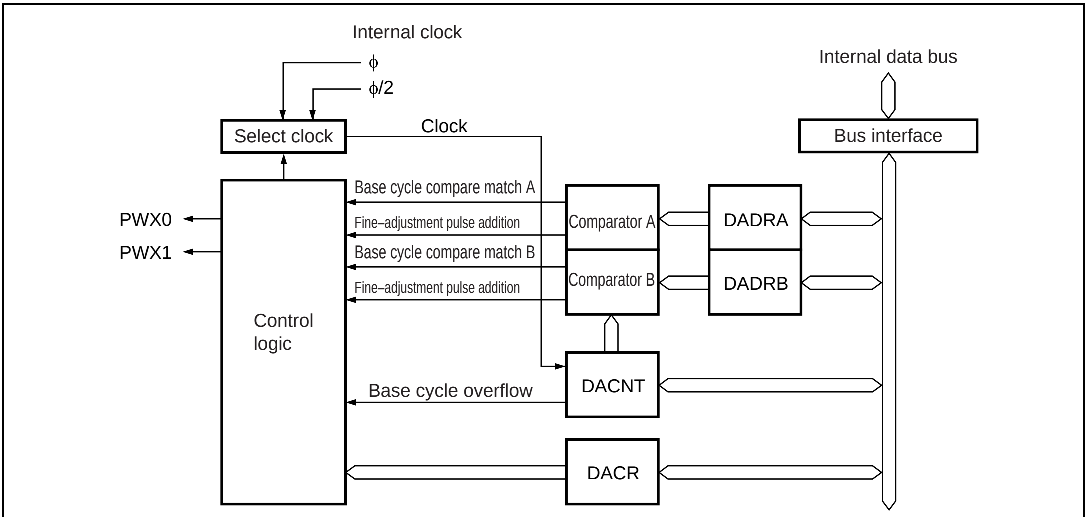
8.1 Features

- Division of pulse into multiple base cycles to reduce ripple
- Two resolution settings

  The resolution can be set equal to one or two system clock cycles.
- Two base cycle settings

  The base cycle can be set equal to T × 64 or T × 256, where T is the resolution.
- Four operating speeds
- Four operation clocks (by combination of two resolution settings and two base cycle settings)

Figure 8.1 shows a block diagram of the PWM (D/A) module.



Legend

DACR : PWM D/A control register
DADRA : PWM D/A data register A
DADRB : PWM D/A data register B
DACNT : PWM D/A counter

Figure 8.1 PWM (D/A) Block Diagram

8.2 Input/Output Pins

Table 8.1 lists the PWM (D/A) module input and output pins.

Table 8.1 Pin Configuration

Name	Abbreviation	I/O	Function
PWM output pin X0	PWX0	Output	PWM output of PWMX channel A
PWM output pin X1	PWX1	Output	PWM output of PWMX channel B

8.3 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). 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)

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.

8.3.1 PWM (D/A) Counters H and 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. Since DACNT consists of 16-bit data, DACNT transfers data to the CPU via the temporary register (TEMP). For details, refer to section 8.4, Bus Master Interface.

	DACNTH								DACNTL							
Bit (CPU) :	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit (Counter) :	7	6	5	4	3	2	1	0	8	9	10	11	12	13	-	-
															-	REGS

- 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 DADRB and DACNT, are located at the same addresses. The REGS bit specifies which registers can be accessed. 0: DADRA and DADRB can be accessed 1: DACR and DACNT can be accessed

8.3.2 PWM (D/A) Data Registers A and B (DADRA, DADRB)

DADRA corresponds to PWM (D/A) channel A, and DADRB to PWM (D/A) channel B. Since DADR consists of 16-bit data, DADR transfers data to the CPU via the temporary register (TEMP). For details, refer to section 8.4, Bus Master Interface.

- DADRA

Bit	Bit Name	Initial Value	R/W	Description
15	DA13	1	R/W	D/A Data 13 to 0 These bits set a digital value to be converted to an analog value. 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. 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.
14	DA12	1	R/W	
13	DA11	1	R/W	
12	DA10	1	R/W	
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	
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 These bits set a digital value to be converted to an analog value. 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. 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.
14	DA12	1	R/W	
13	DA11	1	R/W	
12	DA10	1	R/W	
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	
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	REGS	1	R/W	Register Select DADRA and DACR, and DADRB and DACNT, are located at the same addresses. The REGS bit specifies which registers can be accessed. 0: DADRA and DADRB can be accessed 1: DACR and DACNT can be accessed

8.3.3 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	TEST	0	R/W	Test Mode Selects test mode, which is used in testing this LSI. Normally this bit should be cleared to 0. 0: PWM (D/A) in user state: Normal operation 1: PWM (D/A) in test state: Correct conversion results unobtainable
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	—	1	R	Reserved
4	—	1	R	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

Bit	Bit Name	Initial Value	R/W	Description
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 (φ) frequency is 10 MHz, resolutions of 100 ns and 200 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}) × 2

8.4 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 8.2 Read and Write Access Methods for 16-Bit Registers

	Read		Write	
Register Name	**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.

8.5 Operation

A PWM waveform like the one shown in figure 8.2 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 8.3 and 8.4 show the types of waveform output available.

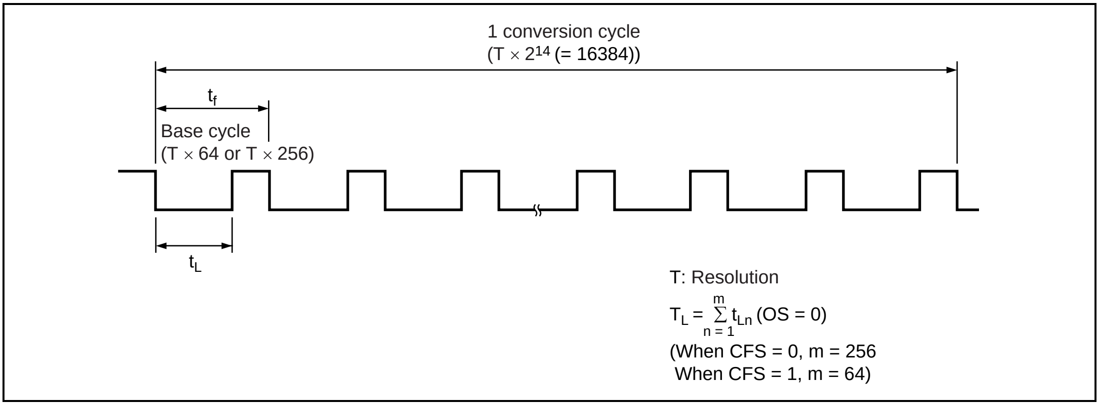


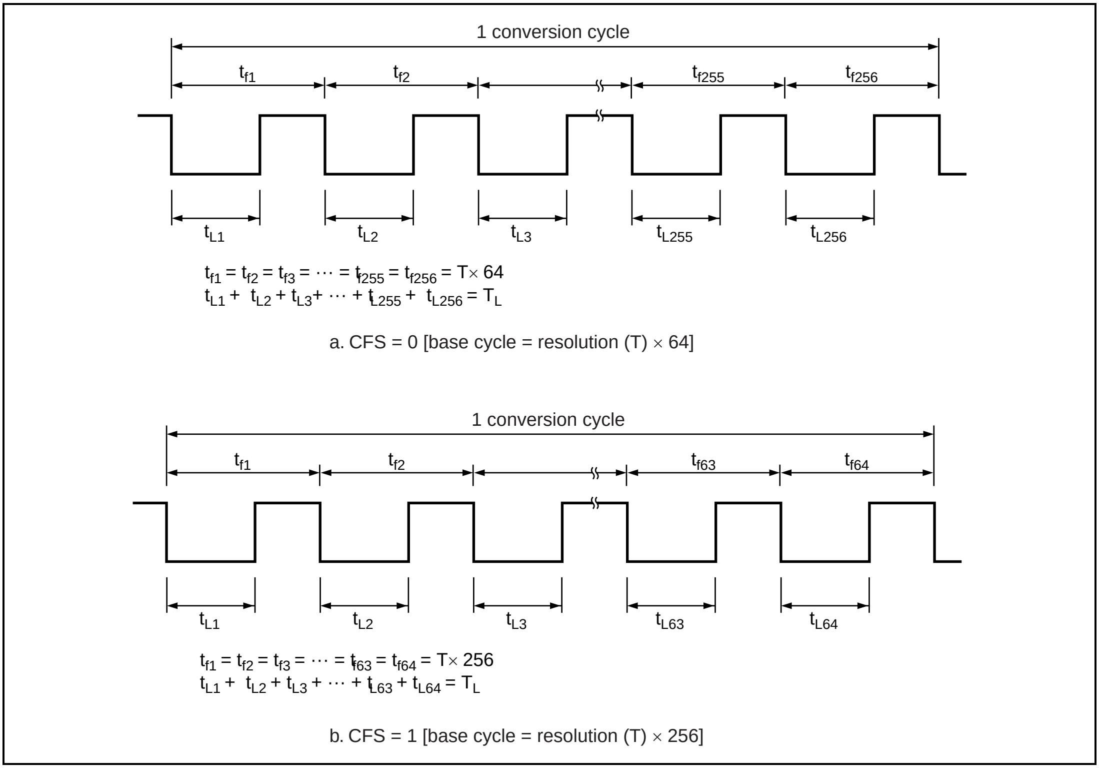
Figure 8.2 PWM (D/A) Operation

Table 8.3 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 8.3 Settings and Operation (Examples when ϕ = 10 MHz)

						Fixed DADR Bits					
							Bit Data				
CKS	Resolution T (μs)	CFS	Base Cycle (μs)	Conversion Cycle (μs)	T_L (if OS = 0) T_H (if OS = 1)	Accuracy (Bits)	3	2	1	0	Conversion Cycle* (μs)
0	0.1	0	6.4	1638.4	1. Always low (or high) (DADR = H'0001 to H'03FD)	14					1638.4
						12			0	0	409.6
					2. (Data value) × T (DADR = H'0401 to H'FFFD)	10	0	0	0	0	102.4
		1	25.6		1. Always low (or high) (DADR = H'0003 to H'00FF)	14					1638.4
						12			0	0	409.6
					2. (Data value) × T (DADR = H'0103 to H'FFFF)	10	0	0	0	0	102.4
1	0.2	0	12.8	3276.8	1. Always low (or high) (DADR = H'0001 to H'03FD)	14					3276.8
						12			0	0	819.2
					2. (Data value) × T (DADR = H'0401 to H'FFFD)	10	0	0	0	0	204.8
		1	51.2		1. Always low (or high) (DADR = H'0003 to H'00FF)	14					3276.8
						12			0	0	819.2
					2. (Data value) × T (DADR = H'0103 to H'FFFF)	10	0	0	0	0	204.8

Note:* This column indicates the conversion cycle when specific DADR bits are fixed.


1 conversion cycle
tf1
tf2
tf255
tf256
tL1
tL2
tL3
tL255
tL256
tf1 = tf2 = tf3 = ··· = tf255 = tf256 = T× 64
tL1 + tL2 + tL3+ ··· + tL255 + tL256 = TL
1 conversion cycle
tf1
tf2
tf63
tf64
tL1
tL2
tL3
tL63
tL64
tf1 = tf2 = tf3 = ··· = tf63 = tf64 = T× 256
tL1 + tL2 + tL3 + ··· + tL63 + tL64 = TL

a. CFS = 0 [base cycle = resolution (T) × 64]

b. CFS = 1 [base cycle = resolution (T) × 256]

Figure 8.3 Output Waveform (OS = 0, DADR corresponds to T_L)

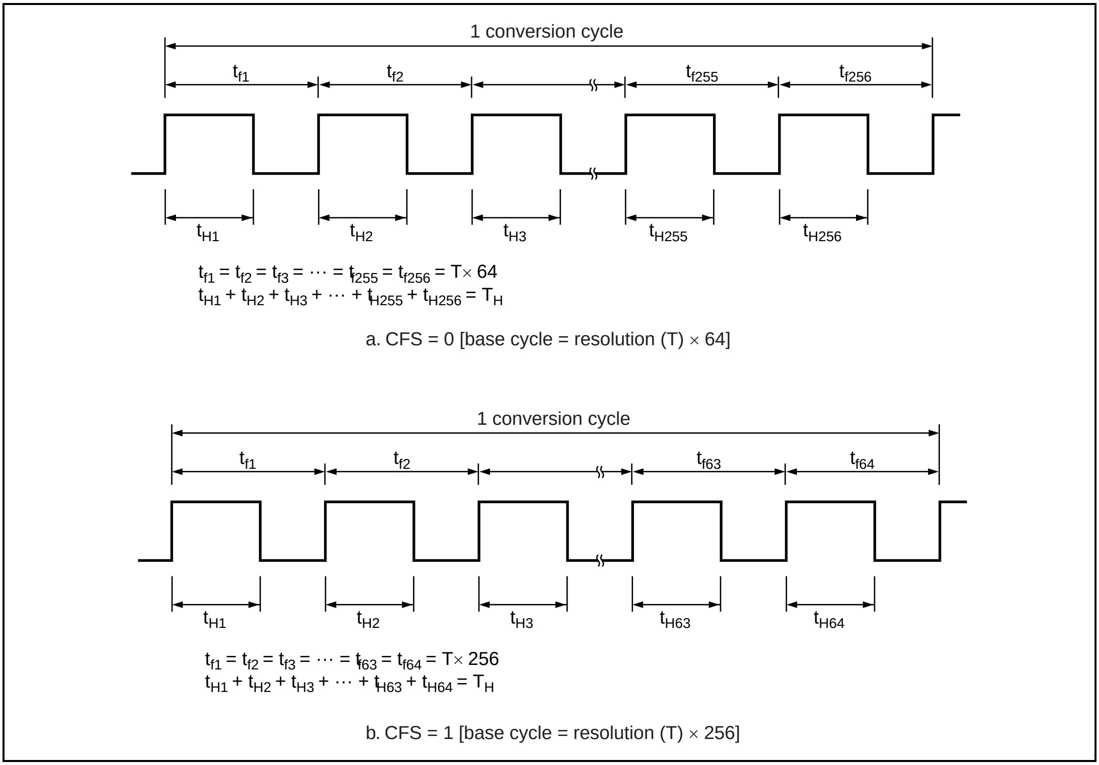

1 conversion cycle
tf1
tf2
tf255
tf256
tH1
tH2
tH3
tH255
tH256
tf1 = tf2 = tf3 = ⋯ = tf255 = tf256 = T× 64
tH1 + tH2 + tH3 + ⋯ + tH255 + tH256 = TH
a. CFS = 0 [base cycle = resolution (T) × 64]
1 conversion cycle
tf1
tf2
tf63
tf64
tH1
tH2
tH3
tH63
tH64
tf1 = tf2 = tf3 = ⋯ = tf63 = tf64 = T× 256
tH1 + tH2 + tH3 + ⋯ + tH63 + tH64 = TH
b. CFS = 1 [base cycle = resolution (T) × 256]

Figure 8.4 Output Waveform (OS = 1, DADR corresponds to T_H)

An example of setting CFS to 1 (basic cycle = resolution (T) × 256) and OS to 1 (PWMX inverted output) is shown as an additional pulse. When CFS is set to 1, the duty ratio of the basic pulse is determined by the upper eight bits (DA13 to DA6) in DADR, and the position of the additional pulse is determined by the following six bits (DA5 to DA0) as shown in figure 8.5.

Tables 8.4 to 8.6 show the position of the additional pulse.

DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	CFS	——
Basic pulse duty ratio								Additional pulse position						1	1

Figure 8.5 D/A Data Register Configuration when CFS = 1

Here, the case of DADR = H’0207 (B’0000 0010 0000 0111) is considered. Figure 8.6 shows an output waveform. Because CFS = 1 and the value of upper eight bits is B’0000 0010, the duty ratio of the basic pulse is 2/256 × (T) of high width.

Since the value of the following six bits is B’0000 01, the additional pulse is output at the position of basic pulse No. 63 as shown in table 8.4. Only 1/256 × (T) of the additional pulse is added to the basic pulse.

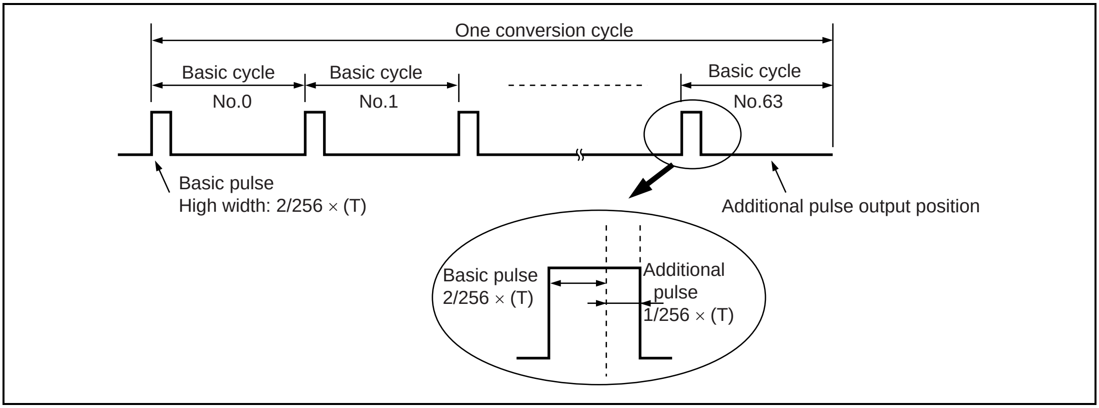


Figure 8.6 Output Waveform when DADR = H'0207 (OS = 1)

Note that the case of CFS = 0 (basic cycle = resolution (T) × 64) is similar other than the duty ratio of the basic pulse is determined by the upper six bits, and the position of the additional pulse is determined by the following eight bits.

Table 8.4 Position of Pulse to be Added to Basic Pulse with 14-Bit Conversion Accuracy (CFS = 1)

||||||||Basic Pulse No.|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Lower 6 bits							0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
0	0	0	0	0	0	0																																																																
0	0	0	0	0	1	1																																																															Yes
0	0	0	0	1	0	2																															Yes																																	Yes
0	0	0	0	1	1	3																															Yes																	Yes																	Yes
0	0	0	1	0	0	4															Yes																	Yes																	Yes																	Yes
0	0	0	1	0	1	5															Yes																	Yes																	Yes								Yes									Yes
0	0	0	1	1	0	6															Yes								Yes								Yes																	Yes								Yes									Yes
0	0	0	1	1	1	7															Yes								Yes								Yes								Yes								Yes								Yes									Yes
0	0	1	0	0	0	8							Yes								Yes								Yes								Yes								Yes								Yes								Yes								Yes
0	0	1	0	0	1	9							Yes								Yes								Yes								Yes								Yes								Yes								Yes				Yes				Yes
0	0	1	0	1	0	10							Yes								Yes								Yes				Yes				Yes								Yes								Yes								Yes				Yes				Yes
0	0	1	0	1	1	11							Yes								Yes								Yes				Yes				Yes								Yes				Yes				Yes								Yes				Yes				Yes
0	0	1	1	0	0	12							Yes				Yes				Yes								Yes				Yes				Yes								Yes				Yes				Yes								Yes				Yes				Yes
0	0	1	1	0	1	13							Yes				Yes				Yes								Yes				Yes				Yes								Yes				Yes				Yes				Yes				Yes				Yes				Yes
0	0	1	1	1	0	14							Yes				Yes				Yes				Yes				Yes				Yes				Yes								Yes				Yes				Yes				Yes				Yes				Yes				Yes
0	0	1	1	1	1	15							Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes
0	1	0	0	0	0	16				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes
0	1	0	0	0	1	17				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes		Yes		Yes
0	1	0	0	1	0	18				Yes				Yes				Yes				Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes				Yes				Yes				Yes		Yes		Yes
0	1	0	0	1	1	19				Yes				Yes				Yes				Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes
0	1	0	1	0	0	20				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes
0	1	0	1	0	1	21				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes
0	1	0	1	1	0	22				Yes				Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes
0	1	0	1	1	1	23				Yes				Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes
0	1	1	0	0	0	24				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes
0	1	1	0	0	1	25				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes
0	1	1	0	1	0	26				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes
0	1	1	0	1	1	27				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes
0	1	1	1	0	0	28				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes
0	1	1	1	0	1	29				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes
0	1	1	1	1	0	30				Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes				Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes
0	1	1	1	1	1	31				Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes	Yes		Yes		Yes		Yes
1	0	0	0	0	0	32		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes
1	0	0	0	0	1	33		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes
1	0	0	0	1	0	34		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes
1	0	0	0	1	1	35		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes
1	0	0	1	0	0	36		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes
1	0	0	1	0	1	37		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes	Yes	Yes
1	0	0	1	1	0	38		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes	Yes	Yes
1	0	0	1	1	1	39		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes	Yes	Yes
1	0	1	0	0	0	40		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes	Yes	Yes
1	0	1	0	0	1	41		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	0	1	0	1	0	42		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	0	1	0	1	1	43		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	0	1	1	0	0	44		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	0	1	1	0	1	45		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	0	1	1	1	0	46		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	0	1	1	1	1	47		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	1	0	0	0	0	48		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	1	0	0	0	1	49		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	0	0	1	0	50		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	0	0	1	1	51		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	0	1	0	0	52		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	0	1	0	1	53		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	0	1	1	0	54		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	0	1	1	1	55		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	0	0	0	56		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	0	0	1	57		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	0	1	0	58		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	0	1	1	59		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	1	0	0	60		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	1	0	1	61		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	1	1	0	62		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	1	1	1	63		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Table 8.5 Position of Pulse to be Added to Basic Pulse with 12-Bit Conversion Accuracy (CFS = 1)

							Basic Pulse No.																																																												
							0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
Lower 6 bits																																																																							
0	0	0	0	0	0	0																																																												
0	0	0	1	0	0	4																Yes																Yes																Yes																Yes
0	0	1	0	0	0	8								Yes								Yes								Yes								Yes								Yes								Yes								Yes								Yes
0	0	1	1	0	0	12								Yes				Yes				Yes								Yes				Yes				Yes								Yes				Yes				Yes								Yes				Yes				Yes
0	1	0	0	0	0	16				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes
0	1	0	1	0	0	20				Yes				Yes				Yes		Yes		Yes				Yes	Yes			Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes				Yes				Yes				Yes		Yes		Yes
0	1	1	0	0	0	24				Yes		Yes		Yes				Yes		Yes		Yes				Yes	Yes			Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes				Yes		Yes		Yes
0	1	1	1	0	0	28				Yes		Yes		Yes		Yes		Yes		Yes		Yes				Yes	Yes			Yes		Yes		Yes		Yes		Yes				Yes		Yes		Yes		Yes		Yes		Yes		Yes				Yes		Yes		Yes		Yes		Yes		Yes		Yes
1	0	0	0	0	0	32		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes			Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes
1	0	0	1	0	0	36		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes	Yes			Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes	Yes	Yes
1	0	1	0	0	0	40		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes	Yes		Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes
1	0	1	1	0	0	44		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes	Yes		Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes		Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	1	0	0	0	0	48		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes		Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes
1	1	0	1	0	0	52		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes		Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	0	0	0	56		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes
1	1	1	1	0	0	60		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Table 8.6 Position of Pulse to be Added to Basic Pulse with 10-Bit Conversion Accuracy (CFS = 1)

							Basic Pulse No.																															
							0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Lower 6 bits																																						
0	0	0	0	0	0	0																																
0	1	0	0	0	0	16				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes
1	0	0	0	0	0	32		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes
1	1	0	0	0	0	48		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes

							Basic Pulse No.																															
							32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
Lower 6 bits																																						
0	0	0	0	0	0	0																																
0	1	0	0	0	0	16				Yes				Yes				Yes				Yes				Yes				Yes				Yes				Yes
1	0	0	0	0	0	32		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes		Yes
1	1	0	0	0	0	48		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes		Yes	Yes	Yes

8.6 Usage Note

8.6.1 Module Stop Mode Setting

PWMX operation can be enabled or disabled using the module stop control register. The initial setting is for PWMX operation to be halted. Register access is enabled by canceling the module stop mode. For details, refer to section 19, Power-Down Modes.

Section 9   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.

9.1 Features

- Selection of four clock sources

  One of the three internal clocks (ϕ/2, ϕ/8, or ϕ/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

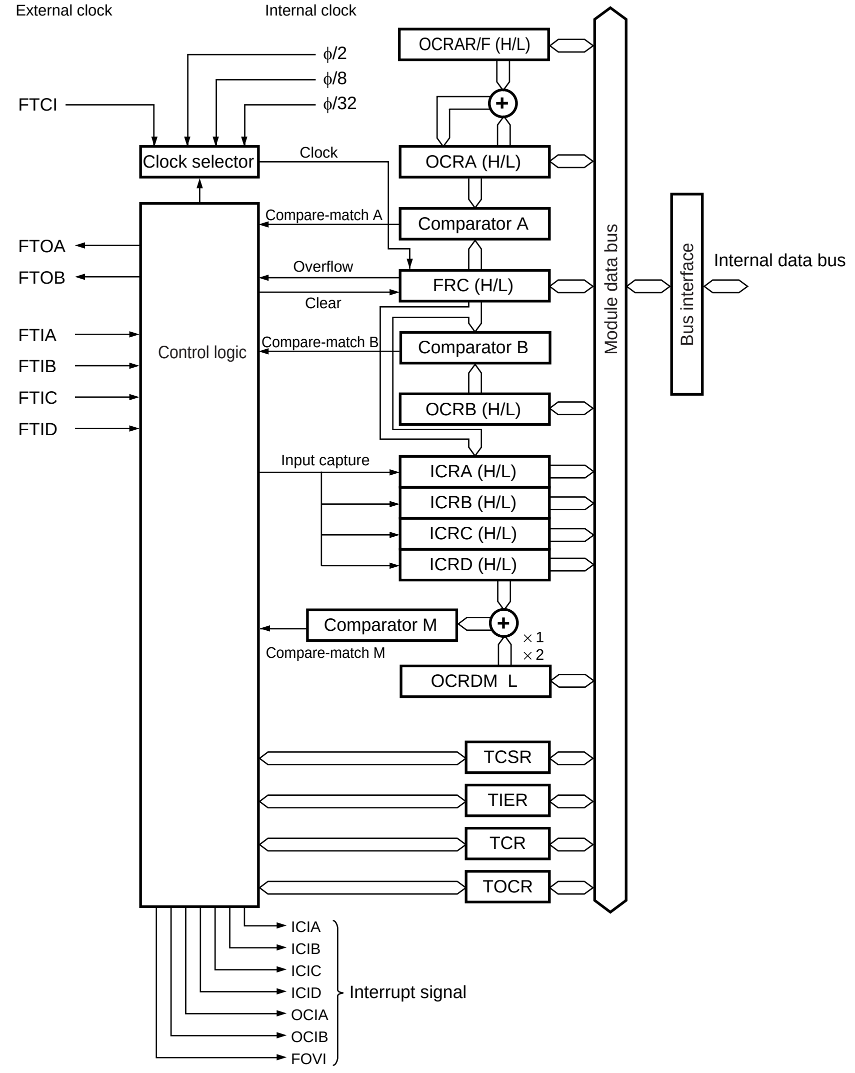
  The free-running counters can be cleared on compare-match A.
- Seven independent interrupts

  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 × 2, enabling input capture operations in this interval to be restricted.

TIM8FR1A_010020020800

Figure 9.1 shows a block diagram of the FRT.



Legend

OCRA, OCRB	: Output compare register A, B (16-bit)
OCRAR,OCRAF	: Output compare register AR, AF (16-bit)
OCRDM	: Output compare register DM (16-bit)
FRC	: Free-running counter (16-bit)
ICRA to D	: Input capture registers A to D (16-bit)
TCSR	: Timer control/status register (8-bit)
TIER	: Timer interrupt enable register (8-bit)
TCR	: Timer control register (8-bit)
TOCR	: Timer output compare control register (8-bit)

Figure 9.1 Block Diagram of 16-Bit Free-Running Timer

9.2 Input/Output Pins

Table 9.1 lists the FRT input and output pins.

Table 9.1 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

9.3 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.

9.3.1 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.

9.3.2 Output Compare Registers A and B (OCRA, 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.

9.3.3 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 (ø) for a single edge. When triggering is enabled on both edges, the input capture pulse width should be at least 2.5 system clocks (ϕ).

ICRA to ICRD should always be accessed in 16-bit units; cannot be accessed in 8-bit units. ICR is initialized to H'0000.

9.3.4 Output Compare Registers AR and AF (OCRAR, 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 ø/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.

9.3.5 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.

9.3.6 Timer Interrupt Enable Register (TIER)

TIER enables and disables interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7	ICIAE	0	R/W	Input Capture Interrupt A Enable Selects whether to enable input capture interrupt A request (ICIA) when input capture flag A (ICFA) in TCSR is set to 1. 0: ICIA requested by ICFA is disabled 1: ICIA requested by ICFA is enabled
6	ICIBE	0	R/W	Input Capture Interrupt B Enable Selects whether to enable input capture interrupt B request (ICIB) when input capture flag B (ICFB) in TCSR is set to 1. 0: ICIB requested by ICFB is disabled 1: ICIB requested by ICFB is enabled
5	ICICE	0	R/W	Input Capture Interrupt C Enable Selects whether to enable input capture interrupt C request (ICIC) when input capture flag C (ICFC) in TCSR is set to 1. 0: ICIC requested by ICFC is disabled 1: ICIC requested by ICFC is enabled
4	ICIDE	0	R/W	Input Capture Interrupt D Enable Selects whether to enable input capture interrupt D request (ICID) when input capture flag D (ICFD) in TCSR is set to 1. 0: ICID requested by ICFD is disabled 1: ICID requested by ICFD is enabled
3	OCIAE	0	R/W	Output Compare Interrupt A Enable Selects whether to enable output compare interrupt A request (OCIA) when output compare flag A (OCFA) in TCSR is set to 1. 0: OCIA requested by OCFA is disabled 1: OCIA requested by OCFA is enabled

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 1 and cannot be modified.

9.3.7 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. Only 0 can be written to this bit to clear the flag. [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)*	Input Capture Flag B 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. Only 0 can be written to this bit to clear the flag. [Setting condition] When an input capture signal causes the FRC value to be transferred to ICRB [Clearing condition] Read ICFB when ICFB = 1, then write 0 to ICFB
5	ICFC	0	R/(W)*	Input Capture Flag C 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. Only 0 can be written to this bit to clear the flag. [Setting condition] When an input capture signal is received [Clearing condition] Read ICFC when ICFC = 1, then write 0 to ICFC
4	ICFD	0	R/(W)*	Input Capture Flag D 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. Only 0 can be written to this bit to clear the flag. [Setting condition] When an input capture signal is received [Clearing condition] Read ICFD when ICFD = 1, then write 0 to ICFD

Bit	Bit Name	Initial Value	R/W	Description
3	OCFA	0	R/(W)*	Output Compare Flag A This status flag indicates that the FRC value matches the OCRA value. Only 0 can be written to this bit to clear the flag. [Setting condition] When FRC = OCRA [Clearing condition] Read OCFA when OCFA = 1, then write 0 to OCFA
2	OCFB	0	R/(W)*	Output Compare Flag B This status flag indicates that the FRC value matches the OCRB value. Only 0 can be written to this bit to clear the flag. [Setting condition] When FRC = OCRB [Clearing condition] Read OCFB when OCFB = 1, then write 0 to OCFB
1	OVF	0	R/(W)*	Timer Overflow This status flag indicates that the FRC has overflowed. Only 0 can be written to this bit to clear the flag. [Setting condition] When FRC overflows (changes from H'FFFF to H'0000) [Clearing condition] Read OVF when OVF = 1, then write 0 to OVF
0	CCLRA	0	R/W	Counter Clear A This bit selects whether the FRC is to be cleared at compare-match A (when the FRC and OCRA values match). 0: FRC clearing is disabled 1: FRC is cleared at compare-match A

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

9.3.8 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		Select clock source for FRC. 00: ϕ/2 internal clock source 01: ϕ/8 internal clock source 10: ϕ/32 internal clock source 11: External clock source (counting at FTCI rising edge)

9.3.9 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	Output Compare Register Select 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. 0: OCRA is selected 1: OCRB is selected
3	OEA	0	R/W	Output Enable A Enables or disables output of the output compare A output pin (FTOA). 0: Output compare A output is disabled 1: Output compare A output is enabled
2	OEB	0	R/W	Output Enable B Enables or disables output of the output compare B output pin (FTOB). 0: Output compare B output is disabled 1: Output compare B output is enabled
1	OLVLA	0	R/W	Output Level A 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. 0: 0 is output at compare-match A 1: 1 is output at compare-match A
0	OLVLB	0	R/W	Output Level B 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). 0: 0 is output at compare-match B 1: 1 is output at compare-match B

9.4 Operation

9.4.1 Pulse Output

Figure 9.2 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.

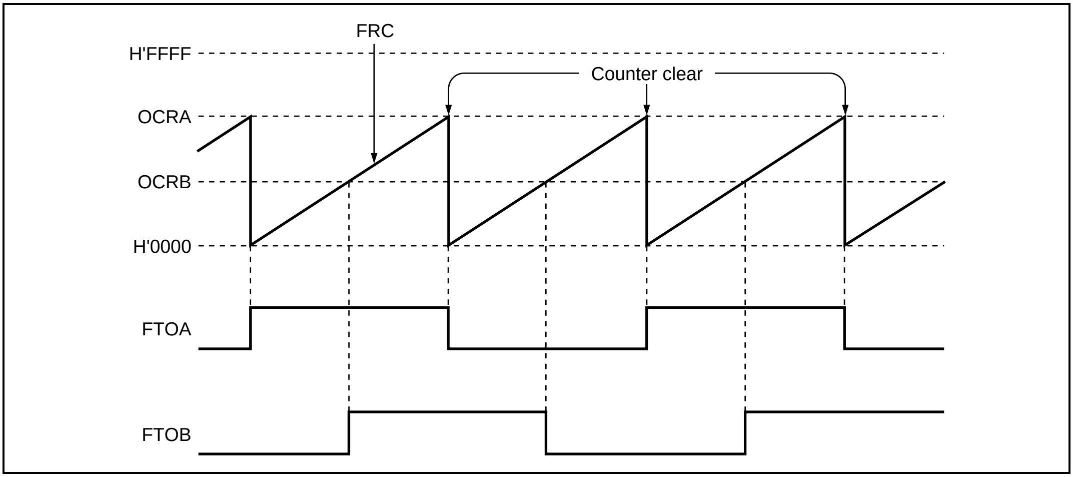


Figure 9.2 Example of Pulse Output

9.5 Operation Timing

9.5.1 FRC Increment Timing

Figure 9.3 shows the FRC increment timing with an internal clock source. Figure 9.4 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 (ϕ). The counter will not increment correctly if the pulse width is shorter than 1.5 system clocks (ϕ).

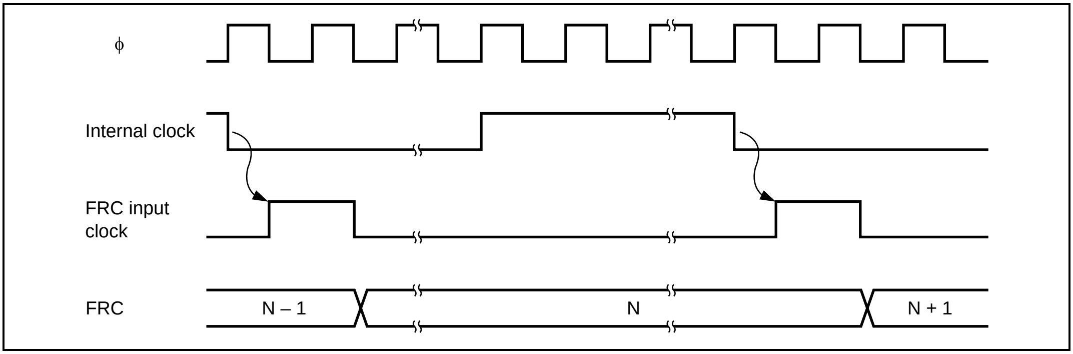


Figure 9.3 Increment Timing with Internal Clock Source

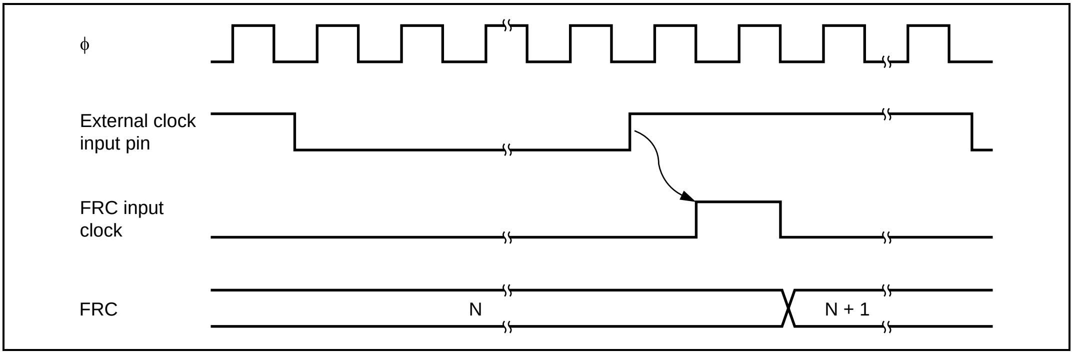
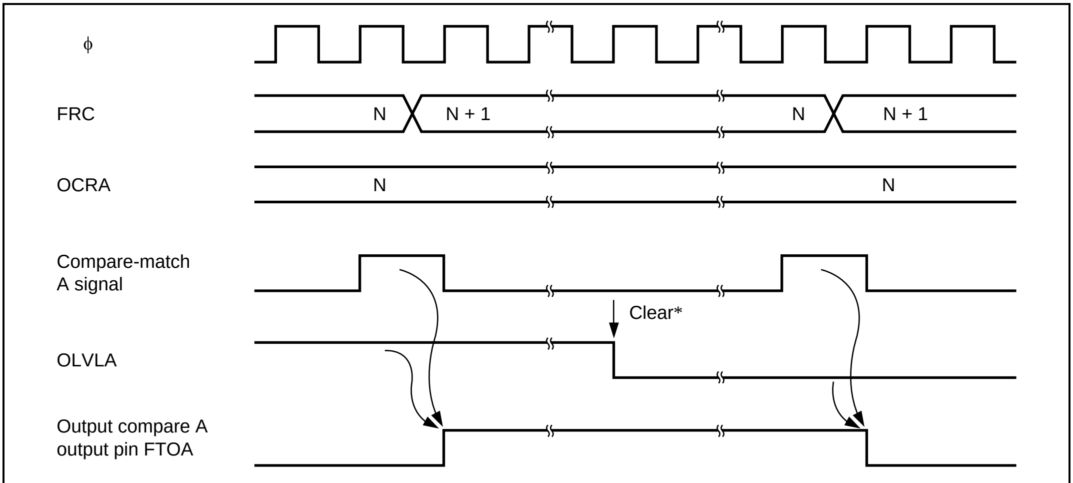


Figure 9.4 Increment Timing with External Clock Source

9.5.2 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 9.5 shows the timing of this operation for compare-match A.



Note : * Indicates instruction execution by software.

Figure 9.5 Timing of Output Compare A Output

9.5.3 FRC Clear Timing

FRC can be cleared when compare-match A occurs. Figure 9.6 shows the timing of this operation.

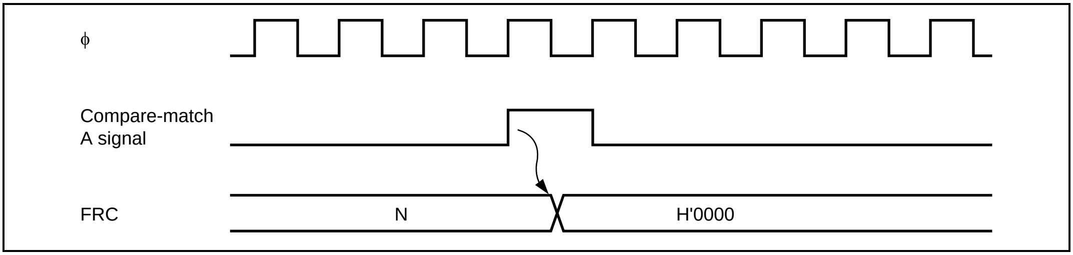


Figure 9.6 Clearing of FRC by Compare-Match A Signal

9.5.4 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 9.7 shows the usual input capture timing when the rising edge is selected.

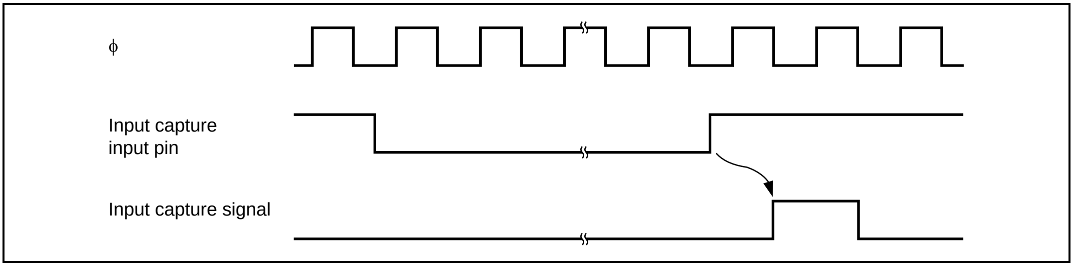


Figure 9.7 Input Capture Input Signal Timing (Usual Case)

If ICRA to ICRAD are read when the corresponding input capture signal arrives, the internal input capture signal is delayed by one system clock (ϕ). Figure 9.8 shows the timing for this case.

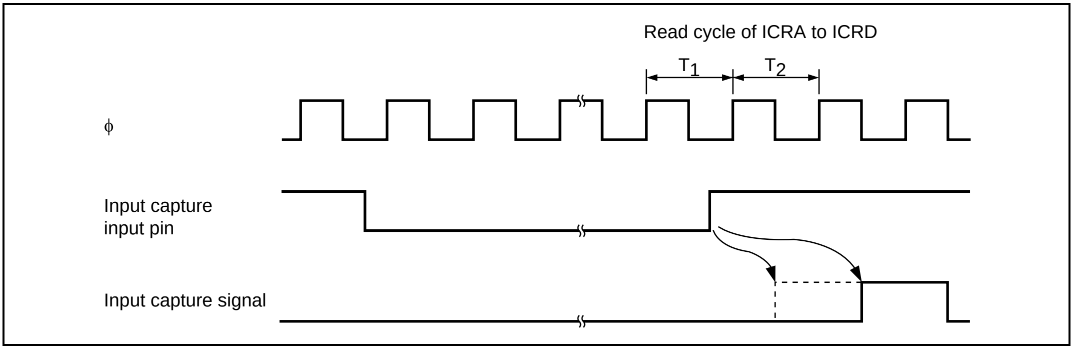


Figure 9.8 Input Capture Input Signal Timing (When ICRA to ICRD are Read)

9.5.5 Buffered Input Capture Input Timing

ICRC and ICRD can operate as buffers for ICRA and ICRB, respectively. Figure 9.9 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.

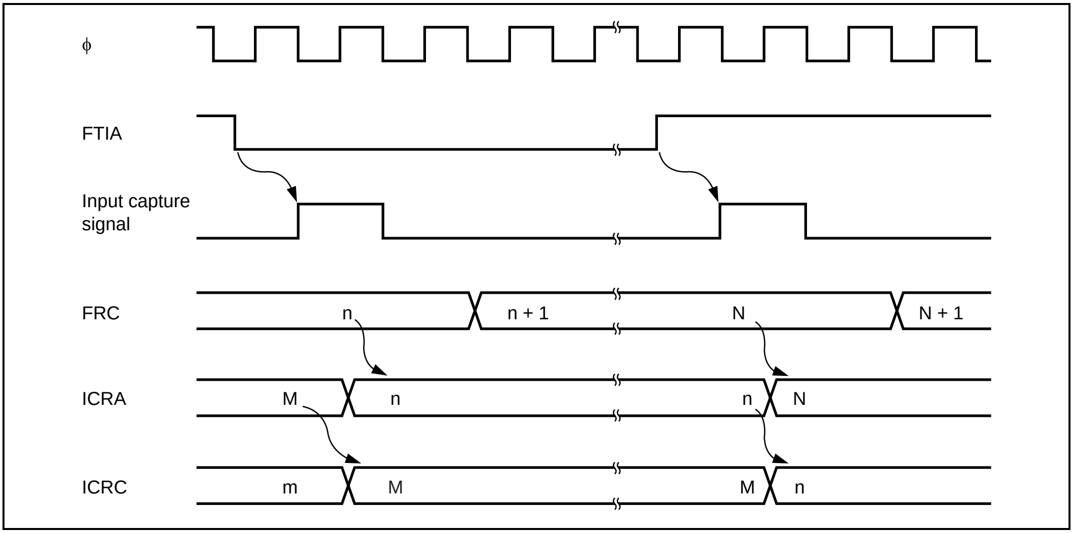


Figure 9.9 Buffered Input Capture Timing

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 (ϕ). Figure 9.10 shows the timing when BUFEA = 1.

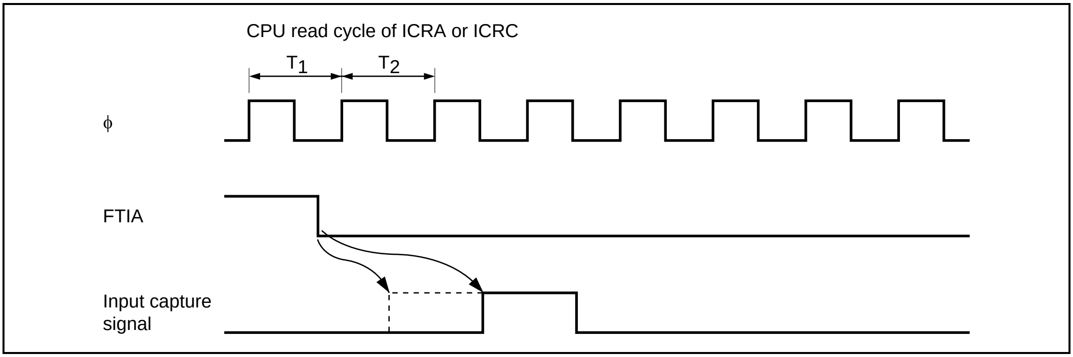


Figure 9.10 Buffered Input Capture Timing (BUFEA = 1)

9.5.6 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 9.11 shows the timing of setting the ICFA to ICFD flag.

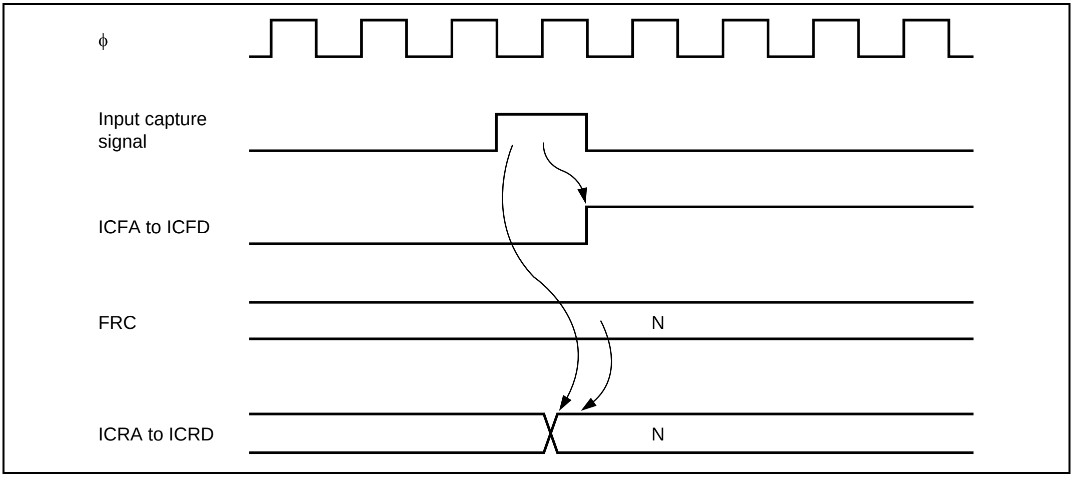


Figure 9.11 Timing of Input Capture Flag (ICFA, ICFB, ICFC, or ICFD) Setting

9.5.7 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 9.12 shows the timing of setting the OCFA or OCFB flag.

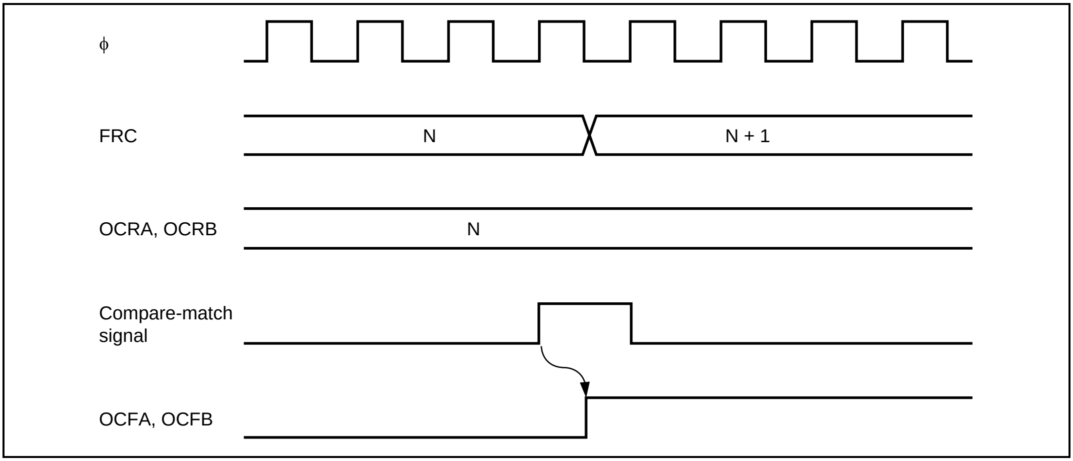


Figure 9.12 Timing of Output Compare Flag (OCFA or OCFB) Setting

9.5.8 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 9.13 shows the timing of setting the OVF flag.

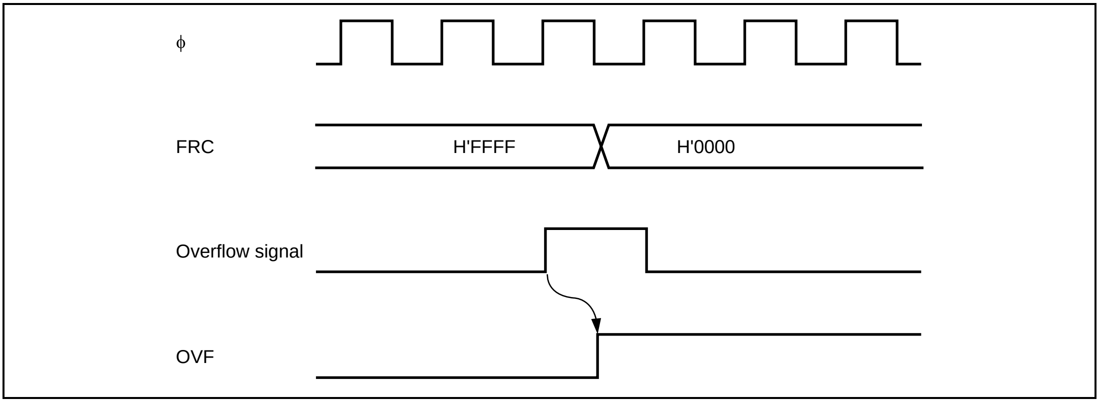


Figure 9.13 Timing of Overflow Flag (OVF) Setting

9.5.9 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 9.14 shows the OCRA write timing.

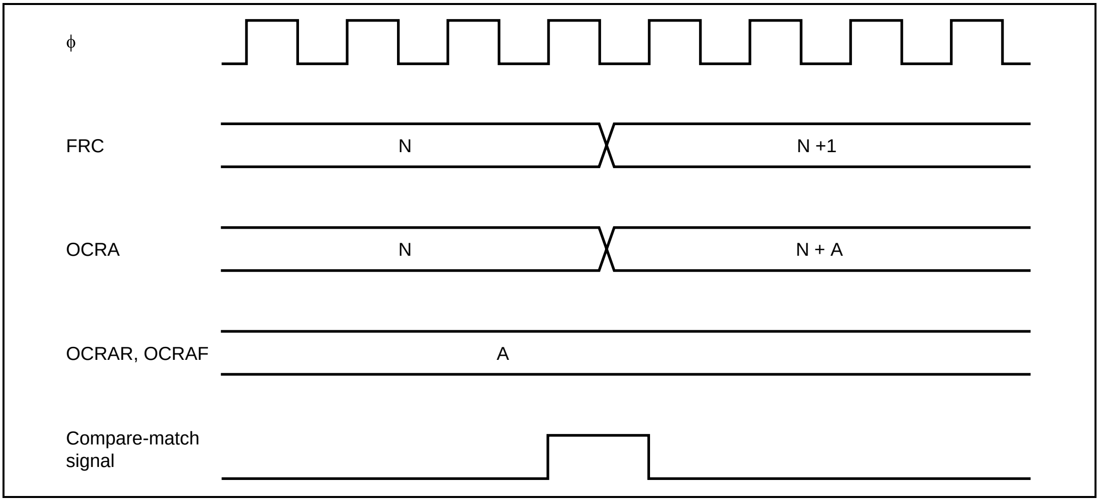


Figure 9.14 OCRA Automatic Addition Timing

9.5.10 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 9.15 shows the timing of setting the mask signal. Figure 9.16 shows the timing of clearing the mask signal.

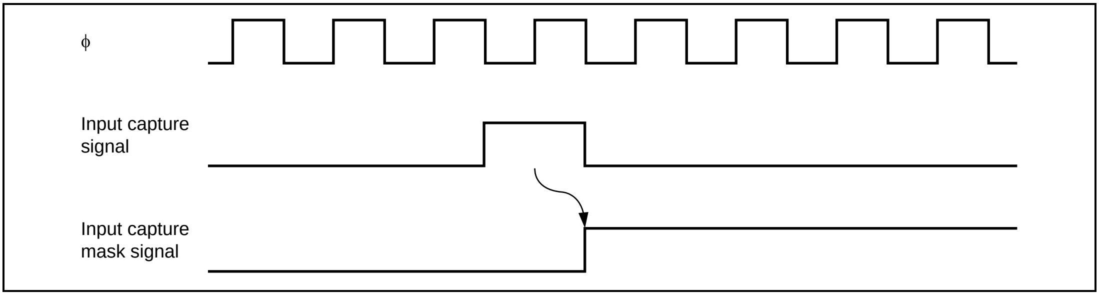


Figure 9.15 Timing of Input Capture Mask Signal Setting

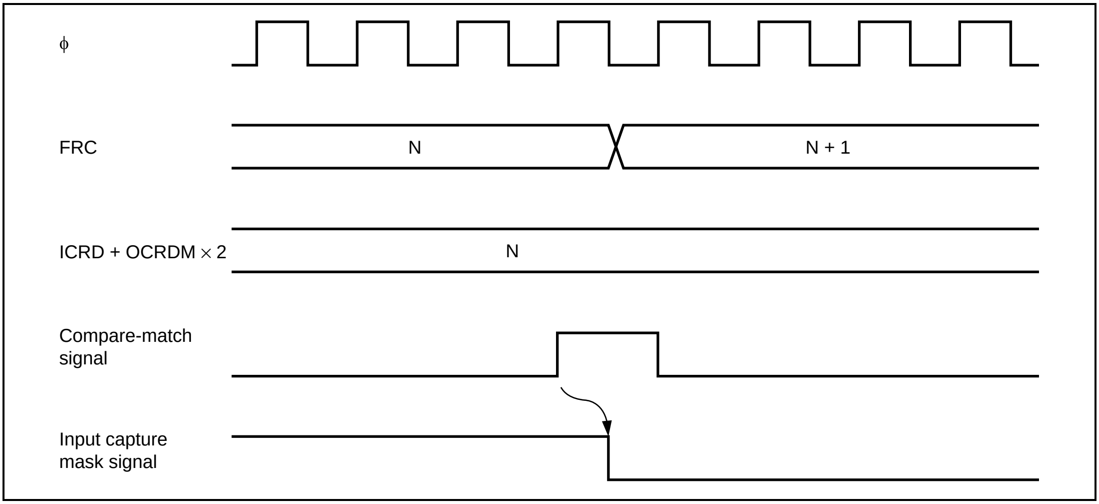


Figure 9.16 Timing of Input Capture Mask Signal Clearing

9.6 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 9.2 lists the sources and priorities of these interrupts.

Table 9.2 FRT Interrupt Sources

Interrupt	Interrupt Source	Interrupt Flag	Priority
ICIA	Input capture of ICRA	ICFA	High
ICIB	Input capture of ICRB	ICFB	↑
ICIC	Input capture of ICRC	ICFC	
ICID	Input capture of ICRD	ICFD	
OCIA	Compare match of OCRA	OCFA	
OCIB	Compare match of OCRB	OCFB	
FOVI	Overflow of FRC	OVF	Low

9.7 Usage Notes

9.7.1 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 9.17 shows the timing for this type of conflict.

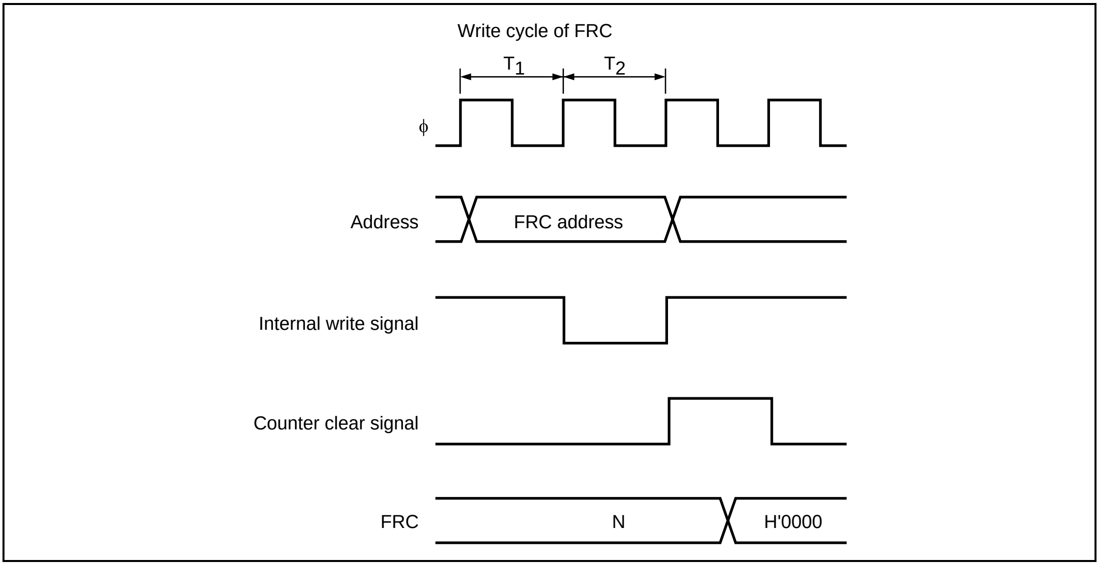


Figure 9.17 FRC Write-Clear Conflict

9.7.2 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 9.18 shows the timing for this type of conflict.

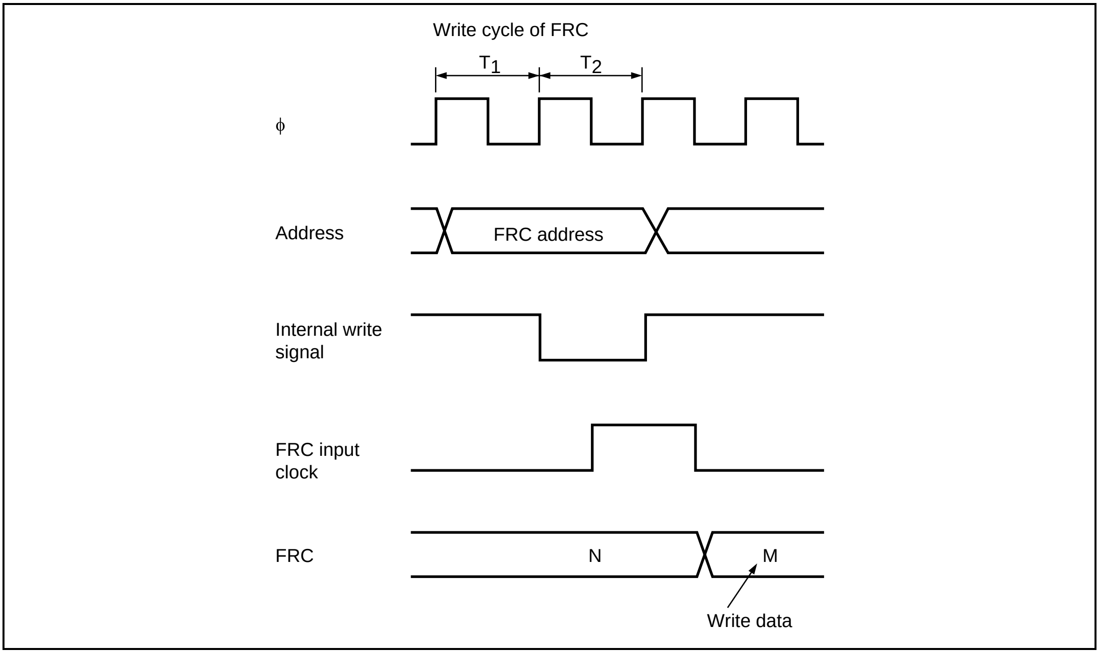


Figure 9.18 FRC Write-Increment Conflict

9.7.3 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 9.19 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 9.20 shows the timing for this type of conflict.

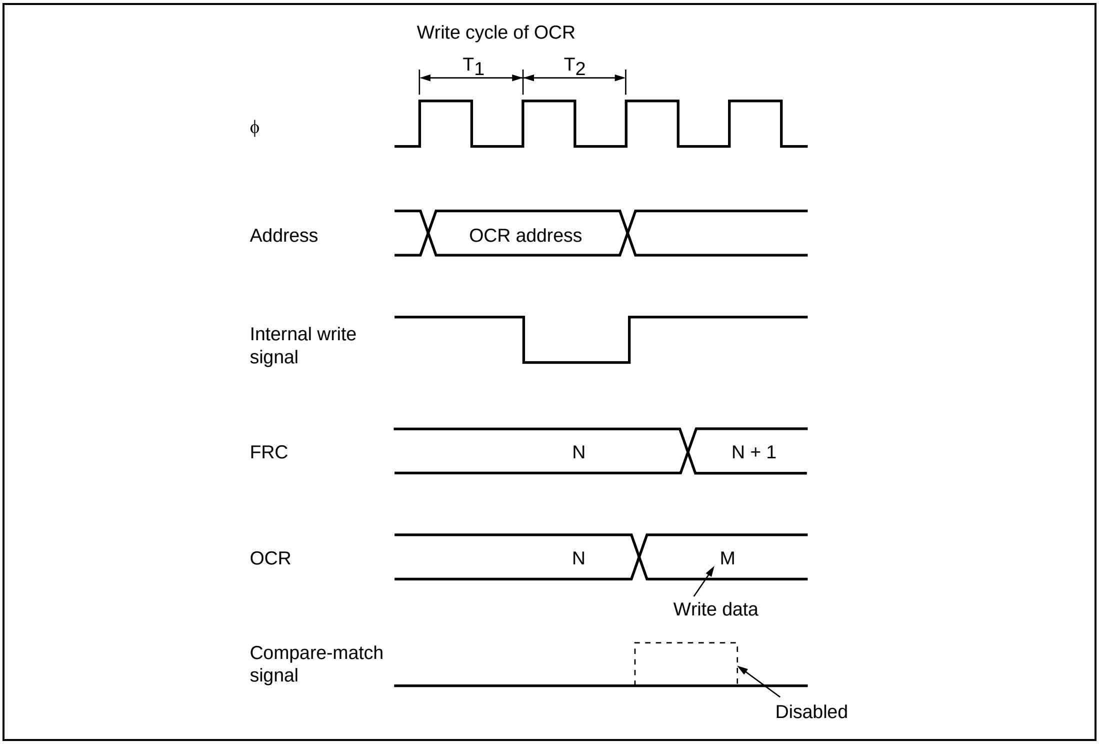


Figure 9.19 Conflict between OCR Write and Compare-Match (When Automatic Addition Function is Not Used)

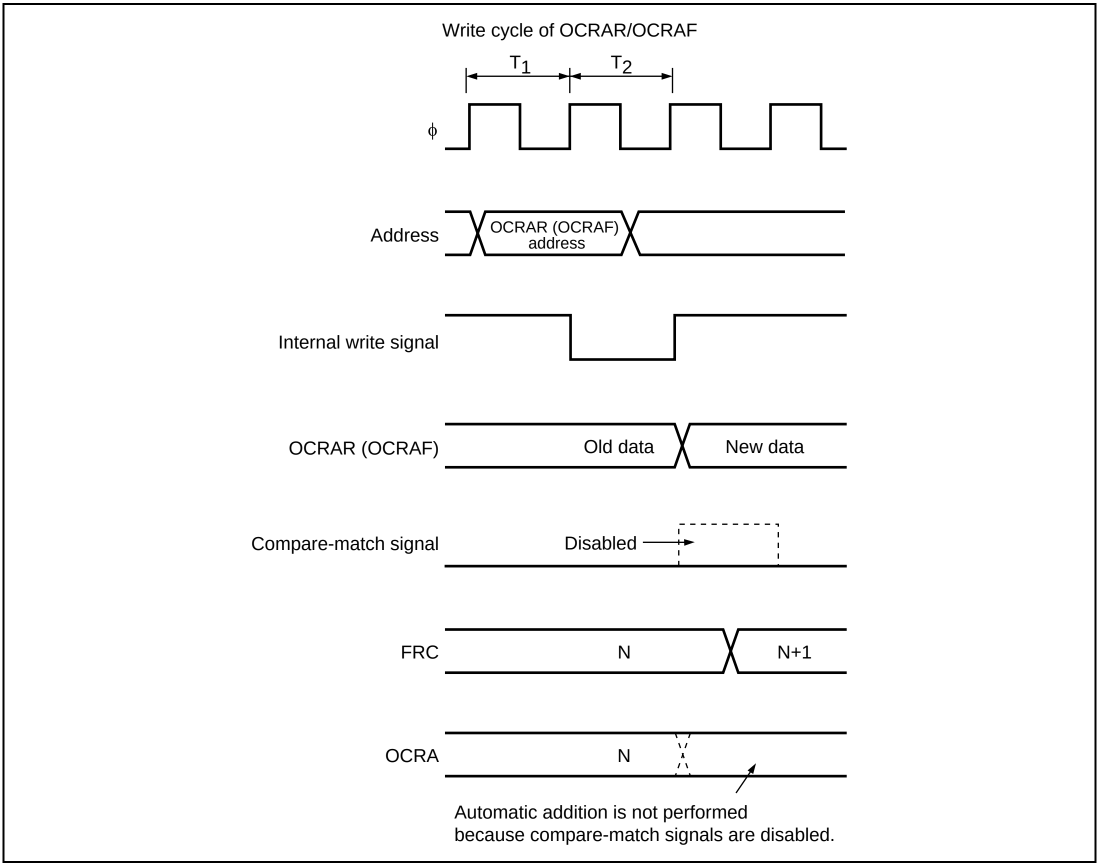


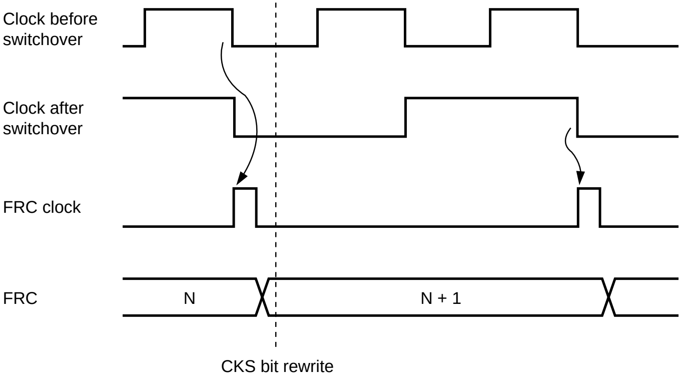
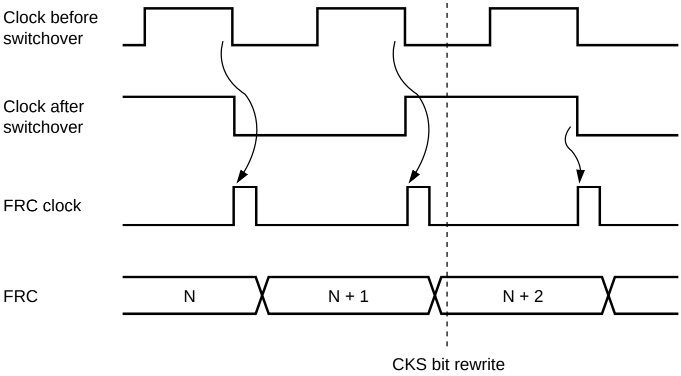
Figure 9.20 Conflict between OCRAR/OCRAF Write and Compare-Match (When Automatic Addition Function is Used)

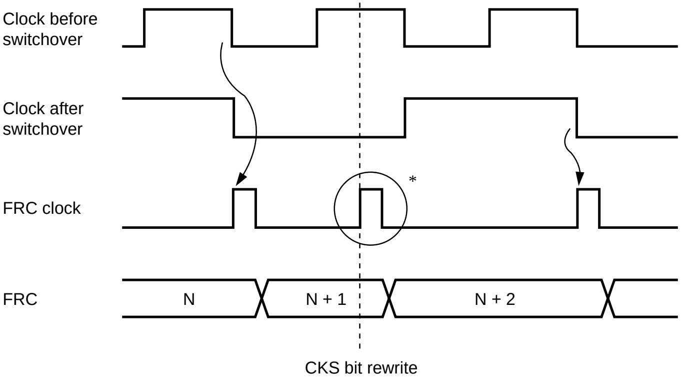
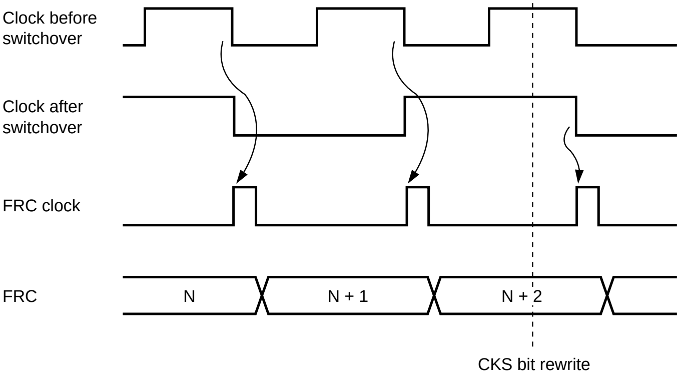
9.7.4 Switching of Internal Clock and FRC Operation

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 9.3.

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 (ϕ). If the clock is changed when the old source is high and the new source is low, as in case no. 3 in table 9.3, 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 9.3 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	 Clock before switchover Clock after switchover FRC clock FRC N N + 1 CKS bit rewrite
2	Switching from low to high	 Clock before switchover Clock after switchover FRC clock FRC N N + 1 N + 2 CKS bit rewrite

No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	FRC Operation
3	Switching from high to low	 Clock before switchover Clock after switchover FRC clock FRC N N + 1 N + 2 * CKS bit rewrite
4	Switching from high to high	 Clock before switchover Clock after switchover FRC clock FRC N N + 1 N + 2 CKS bit rewrite

Note:* Generated on the assumption that the switchover is a falling edge; FRC is incremented.

9.7.5 Module Stop Mode Setting

FRT operation can be enabled or disabled using the module stop control register. The initial setting is for FRT operation to be halted. Register access is enabled by canceling the module stop mode. For details, refer to section 19, Power-Down Modes.

Section 10 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.

10.1 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 six internal clocks*[1] 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).
  - Cascading of TMR_Y and TMR_X*[2]

    Operation as a 16-bit timer can be performed using TMR_Y as the upper half and TMR_X as the lower half (16-bit count mode).

    TMR_X can be used to count TMR_Y 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: One type of interrupt: Input capture

Notes: 1. The program development tool (emulator) supports three internal clocks.

2. The program development tool (emulator) does not support this function.

Figures 10.1 and 10.2 show block diagrams of 8-bit timers.

An input capture function is added to TMR_X.

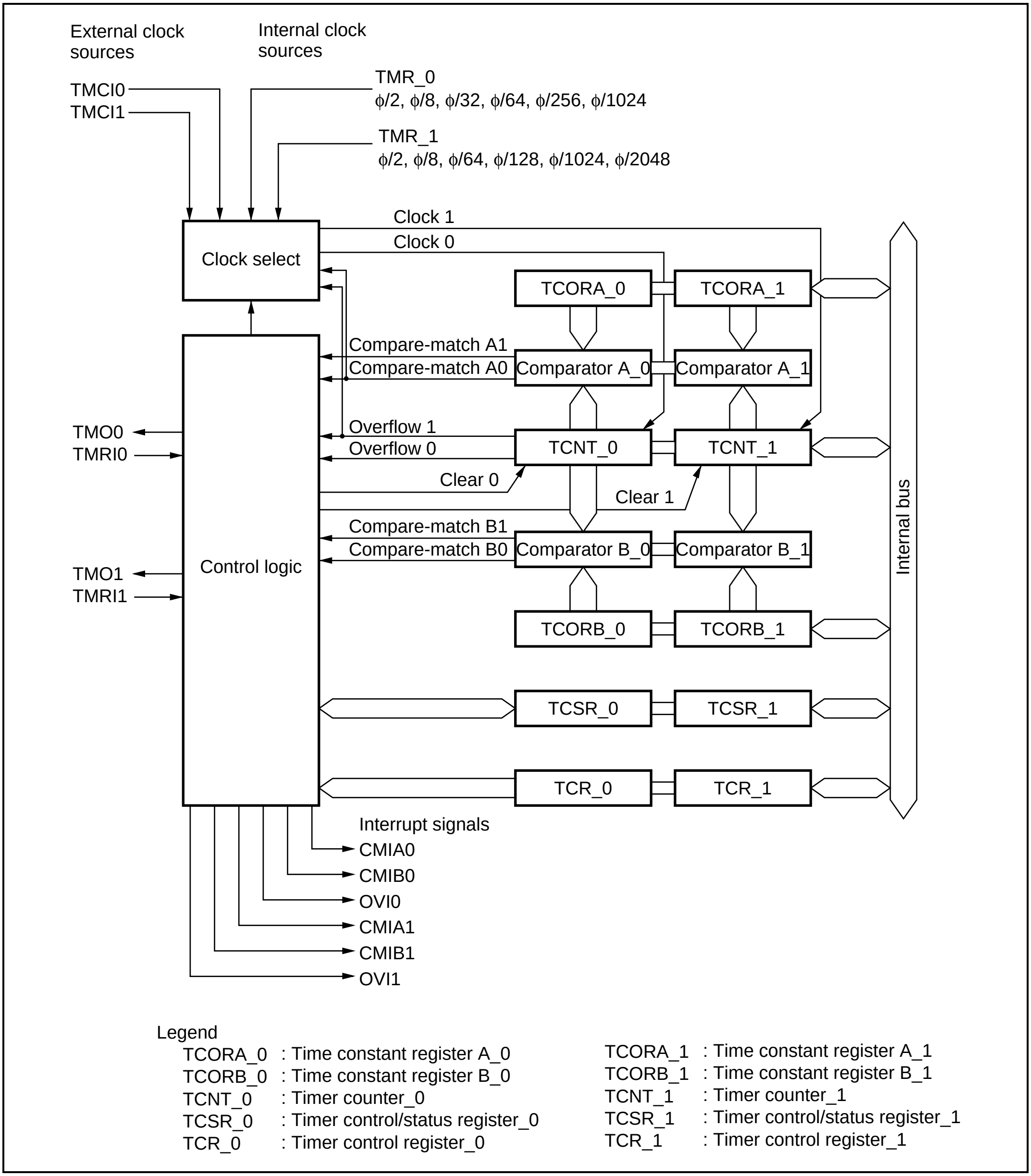
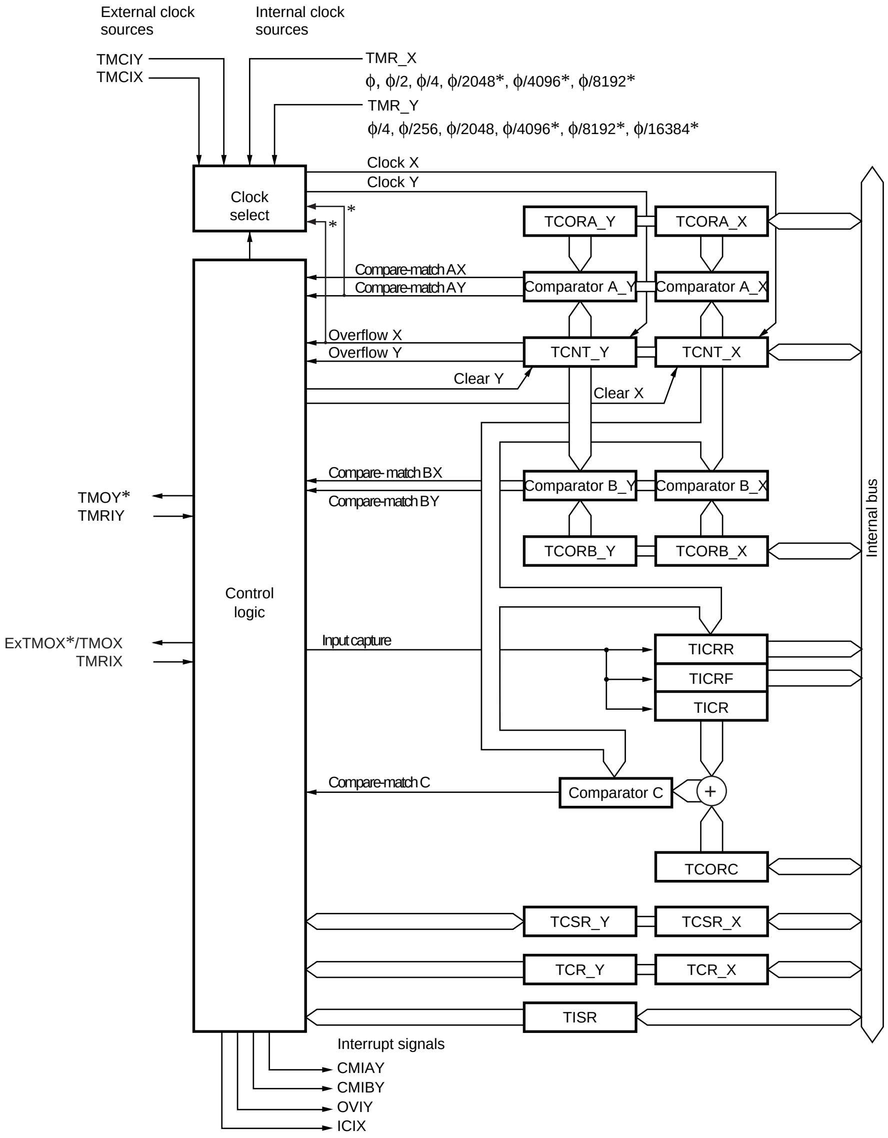


Figure 10.1 Block Diagram of 8-Bit Timer (TMR_0 and TMR_1)



Legend

TCORA_Y: Time constant register A_Y
TCORB_Y: Time constant register B_Y
TCNT_Y: Timer counter_Y
TCSR_Y: Timer control/status register_Y
TCR_Y: Timer control register_Y
TISR: Timer input select register

TCORA_X: Time constant register A_X
TCORB_X: Time constant register B_X
TCNT_X: Timer counter_X
TCSR_X: Timer control/status register_X
TCR_X: Timer control register_X
TICR: Input capture register
TCORC: Time constant register C
TICRR: Input capture register R
TICRF: Input capture register F

Note: The program development tool (emulator) does not support this function.

Figure 10.2 Block Diagram of 8-Bit Timer (TMR_Y and TMR_X)

10.2 Input/Output Pins

Table 10.1 summarizes the input and output pins of the TMR.

Table 10.1 Pin Configuration

Channel	Name	Symbol	I/O	Function
TMR_0	Timer output	TMO0	Output	Output controlled by compare-match
	Timer clock input	TMCI0	Input	External clock input for the counter
	Timer reset input	TMRI0	Input	External reset input for the counter
TMR_1	Timer output	TMO1	Output	Output controlled by compare-match
	Timer clock input	TMCI1	Input	External clock input for the counter
	Timer reset input	TMRI1	Input	External reset input for the counter
TMR_Y	Timer clock/reset input	TMIY (TMCIY/TMRIY)	Input	External clock input/external reset input for the counter
TMR_Y	Timer output	TMOY*	Output	Output controlled by compare-match
TMR_X	Timer output	TMOX/ ExTMOX*	Output	Output controlled by compare-match
	Timer clock/reset input	TMIX (TMCIX/TMRIX)	Input	External clock input/external reset input for the counter

Note: * The program development tool (emulator) does not support this pin.

10.3 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).

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)
- Timer connection register S (TCONRS)

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)
- Timer connection register I (TCONRI)

For both TMR_Y and TMR_X

- Timer XY control register (TCRXY)

Note: Some of the registers of TMR_X and TMR_Y use the same address. The registers can be switched by the TMRX/Y bit in TCONRS.

10.3.1 Timer Counter (TCNT)

Each TCNT is an 8-bit readable/writable up-counter. TCNT_0 and TCNT_1 comprise a single 16-bit register, so they can be accessed together by 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.

TCNT_Y can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 1. TCNT_X can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 0.

10.3.2 Time Constant Register A (TCORA)

TCORA is an 8-bit readable/writable register. TCORA_0 and TCORA_1 comprise a single 16-bit register, so they can be accessed together by 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.

TCORA_Y can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 1. TCORA_X can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 0.

10.3.3 Time Constant Register B (TCORB)

TCORB is an 8-bit readable/writable register. TCORB_0 and TCORB_1 comprise a single 16-bit register, so they can be accessed together by 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.

TCORB_Y can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 1. TCORB_X can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 0.

10.3.4 Timer Control Register (TCR)

TCR selects the TCNT clock source and the condition by which TCNT is cleared, and enables/disables interrupt requests.

TCR_Y can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 1. TCR_X can be accessed when the HIE bit in SYSCR is 0 and the TMRX/Y bit in TCONRS is 0.

Bit	Bit Name	Initial Value	R/W	Description
7	CMIEB	0	R/W	Compare-Match Interrupt Enable B Selects whether the CMFB interrupt request (CMIB) is enabled or disabled when the CMFB flag in TCSR is set to 1. For TMR_X, a CMIB interrupt does not occur irrespective of the value of this bit. 0: CMFB interrupt request (CMIB) is disabled 1: CMFB interrupt request (CMIB) is enabled
6	CMIEA	0	R/W	Compare-Match Interrupt Enable A Selects whether the CMFA interrupt request (CMIA) is enabled or disabled when the CMFA flag in TCSR is set to 1. For TMR_X, a CMIA interrupt does not occur irrespective of the value of this bit. 0: CMFA interrupt request (CMIA) is disabled 1: CMFA interrupt request (CMIA) is enabled
5	OVIE	0	R/W	Timer Overflow Interrupt Enable Selects whether the OVF interrupt request (OVI) is enabled or disabled when the OVF flag in TCSR is set to 1. For TMR_X, an OVI interrupt does not occur irrespective of the value of this bit. 0: OVF interrupt request (OVI) is disabled 1: OVF interrupt request (OVI) is enabled
4	CCLR1	0	R/W	Counter Clear 1, 0
3	CCLR0	0	R/W	These bits select the method by which the timer counter is cleared. 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
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 10.2.
0	CKS0	0	R/W	

Table 10.2 Clock Input to TCNT and Count Condition (1)

	TCR			STCR		
Channel	**CKS2**	**CKS1**	**CKS0**	**ICKS1**	**ICKS0**	**Description**
TMR_0	0	0	0	—	—	Disables clock input
	0	0	1	—	0	Increments at falling edge of internal clock φ/8
	0	0	1	—	1	Increments at falling edge of internal clock φ/2
	0	1	0	—	0	Increments at falling edge of internal clock φ/64
	0	1	0	—	1	Increments at falling edge of internal clock φ/32
	0	1	1	—	0	Increments at falling edge of internal clock φ/1024
	0	1	1	—	1	Increments at falling edge of internal clock φ/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 φ/8
	0	0	1	1	—	Increments at falling edge of internal clock φ/2
	0	1	0	0	—	Increments at falling edge of internal clock φ/64
	0	1	0	1	—	Increments at falling edge of internal clock φ/128
	0	1	1	0	—	Increments at falling edge of internal clock φ/1024
	0	1	1	1	—	Increments at falling edge of internal clock φ/2048
	1	0	0	—	—	Increments at compare-match A from TCNT_0*

Channel	TCR			STCR		Description
	CKS2	CKS1	CKS0	ICKS1	ICKS0	
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

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. These settings should not be made.

Table 10.2 Clock Input to TCNT and Count Condition (2)

Channel	TCR			TCRXY*[2]		Description
	CKS2	CKS1	CKS0	CKSX	CKSY	
TMR_Y	0	0	0	—	0	Disables clock input
	0	0	1	—	0	Increments at ϕ/4
	0	1	0	—	0	Increments at ϕ/256
	0	1	1	—	0	Increments at ϕ/2048
	1	0	0	—	0	Disables clock input
	0	0	0	—	1	Disables clock input
	0	0	1	—	1	Increments at ϕ/4096
	0	1	0	—	1	Increments at ϕ/8192
	0	1	1	—	1	Increments at ϕ/16384
	1	0	0	—	1	Increments at overflow signal from TCNT_X*[1]
	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

	TCR			TCRXY*[2]		
Channel	CKS2	CKS1	CKS0	CKSX	CKSY	Description
TMR_X	0	0	0	0	—	Disables clock input
	0	0	1	0	—	Increments at ϕ
	0	1	0	0	—	Increments at ϕ/2
	0	1	1	0	—	Increments at ϕ/4
	1	0	0	0	—	Disables clock input
	0	0	0	1	—	Disables clock input
	0	0	1	1	—	Increments at ϕ/2048
	0	1	0	1	—	Increments at ϕ/4096
	0	1	1	1	—	Increments at ϕ/8192
	1	0	0	1	—	Increments at compare-match A from TCNT_Y*[1]
	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

Notes: 1. If the TMR_Y clock input is set as the TCNT_X overflow signal and the TMR_X clock input is set as the TCNT_Y compare-match signal simultaneously, a count-up clock cannot be generated. These settings should not be made.

2. The program development tool (emulator) does not support TCRXY. Selection of the internal clock is only available when CKSX = 0 and CKSY = 0.

10.3.5 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] When the values of TCNT_0 and TCORB_0 match [Clearing condition] Read CMFB when CMFB = 1, then write 0 in CMFB
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] When the values of TCNT_0 and TCORA_0 match [Clearing condition] Read CMFA when CMFA = 1, then write 0 in CMFA
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] When TCNT_0 overflows from H'FF to H'00 [Clearing condition] Read OVF when OVF = 1, then write 0 in OVF
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
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)

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 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] When the values of TCNT_1 and TCORB_1 match [Clearing condition] Read CMFB when CMFB = 1, then write 0 in CMFB
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] When the values of TCNT_1 and TCORA_1 match [Clearing condition] Read CMFA when CMFA = 1, then write 0 in CMFA
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] When TCNT_1 overflows from H'FF to H'00 [Clearing condition] Read OVF when OVF = 1, then write 0 in OVF
4	—	1	R	Reserved This bit is always read as 1 and cannot be modified.

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 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

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*	Compare-Match Flag B [Setting condition] When the values of TCNT_X and TCORB_X match [Clearing condition] Read CMFB when CMFB = 1, then write 0 in CMFB
6	CMFA	0	R/(W)*	Compare-Match Flag A [Setting condition] When the values of TCNT_X and TCORA_X match [Clearing condition] Read CMFA when CMFA = 1, then write 0 in CMFA

Bit	Bit Name	Initial Value	R/W	Description
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] When TCNT_X overflows from H'FF to H'00 [Clearing condition] Read OVF when OVF = 1, then write 0 in OVF
4	ICF	0	R/(W)*	Input Capture Flag [Setting condition] When a rising edge and falling edge is detected in the external reset signal in that order. [Clearing condition] Read ICF when ICF = 1, then write 0 in ICF
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)
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)*[1]	Compare-Match Flag B [Setting condition] When the values of TCNT_Y and TCORB_Y match [Clearing condition] Read CMFB when CMFB = 1, then write 0 in CMFB
6	CMFA	0	R/(W)*[1]	Compare-Match Flag A [Setting condition] When the values of TCNT_Y and TCORA_Y match [Clearing condition] Read CMFA when CMFA = 1, then write 0 in CMFA
5	OVF	0	R/(W)*[1]	Timer Overflow Flag [Setting condition] When TCNT_Y overflows from H'FF to H'00 [Clearing condition] Read OVF when OVF = 1, then write 0 in OVF
4	ICIE	0	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
3	OS3	0	R/W	Output Select 3, 2
2	OS2	0	R/W	These bits specify how the TMOY pin*[2] 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)

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 TMOY pin*[2] 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)

Notes: 1. Only 0 can be written, for flag clearing.

2. The program development tool (emulator) does not support this pin.

10.3.6 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.

10.3.7 Input Capture Registers R and F (TICRR and TICRF)

TICRR and TICRF are 8-bit read-only registers. While the ICST bit in TCONRI is set to 1, the contents of TCNT are transferred at the rising edge and falling edge of the external reset input (TMRIX) in that order. The ICST bit is cleared to 0 when one capture operation ends. TICRR and TICRF are initialized to H'00.

10.3.8 Timer Input Select Register (TISR)

TISR permits or prohibits 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 a timer clock/reset input pin (TMIY) as the signal source of external clock/reset input for the TMR_Y counter. 0: Input is prohibited 1: TMIY (TMCIY/TMRIY) is permitted for input

10.3.9 Timer Connection Register I (TCONRI)

TCONRI controls the input capture function.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	R/W	Reserved The initial value should not be changed.
4	ICST	0	R/W	Input Capture Start Bit TMR_X has input capture registers (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 TMRIX 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. [Clearing condition] When a rising edge followed by a falling edge is detected on TMRIX [Setting condition] When 1 is written in ICST after reading ICST = 0
3 to 0	—	All 0	R/W	Reserved The initial values should not be modified.

10.3.10 Timer Connection Register S (TCONRS)

TCONRS selects whether to access TMR_X or TMR_Y registers.

Bit	Bit Name	Initial Value	R/W	Description
7	TMR_X/Y	0	R/W	TMR_X/TMR_Y Access Select For details, see table 10.3. 0: The TMR_X registers are accessed at addresses H'(FF)FFF0 to H'(FF)FFF5 1: The TMR_Y registers are accessed at addresses H'(FF)FFF0 to H'(FF)FFF5
6 to 0		All 0	R/W	Reserved The initial values should not be modified.

Table 10.3 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		

10.3.11 Timer XY Control Register (TCRXY)

TCRXY selects the TMR_X and TMR_Y output pins and internal clock.

Bit	Bit Name	Initial Value	R/W	Description
7	OSX	0	R/W	TMR_X Output Select 0: Output to P67/TMOX 1: Output to P77/ExTMOX
6	OEY	0	R/W	TMR_Y Output Enable 0: Output to P76/TMOY is prohibited 1: Output to P76/TMOY is permitted
5	CKSX	0	R/W	TMR_X Clock Select For details about selection, see the clock conditions in table 10.2.
4	CKSY	0	R/W	TMR_Y Clock Select For details about selection, see the clock conditions in table 10.2.
3 to 0	—	All 0	R/W	Reserved The initial value should not be changed.

Note: * The program development tool (emulator) does not support TCRXY.

10.4 Operation

10.4.1 Pulse Output

Figure 10.3 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.

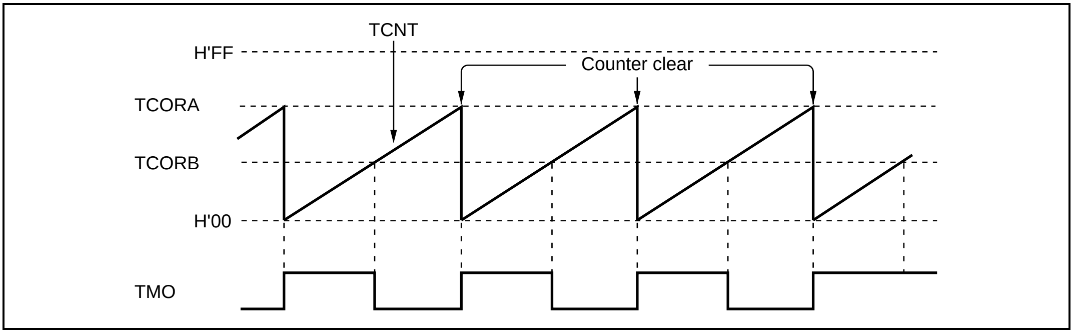


Figure 10.3 Pulse Output Example

10.5 Operation Timing

10.5.1 TCNT Count Timing

Figure 10.4 shows the TCNT count timing with an internal clock source. Figure 10.5 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 (ϕ) for a single edge and at least 2.5 system clocks (ϕ) for both edges. The counter will not increment correctly if the pulse width is less than these values.

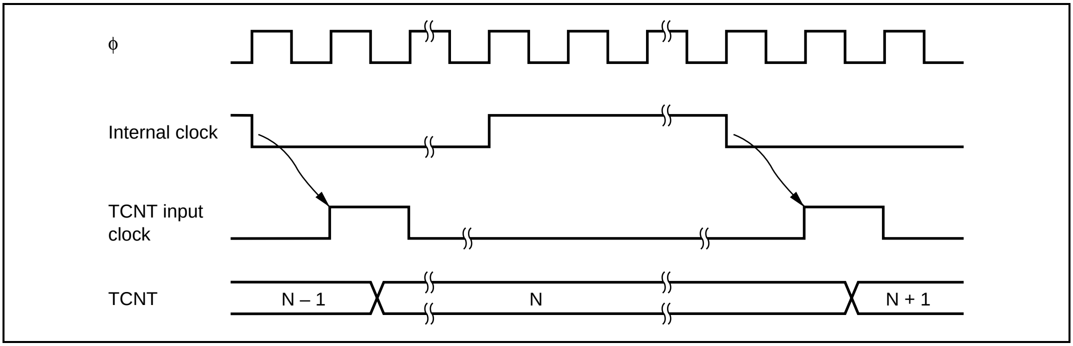


Figure 10.4 Count Timing for Internal Clock Input

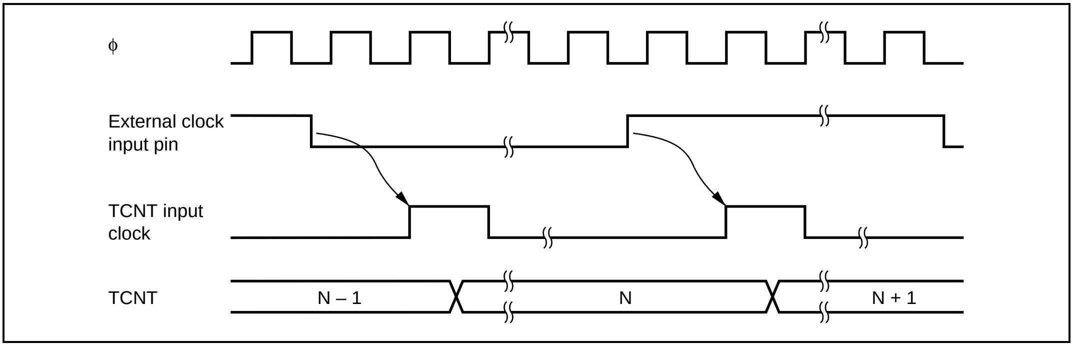


Figure 10.5 Count Timing for External Clock Input (Both Edges)

10.5.2 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 10.6 shows the timing of CMF flag setting.

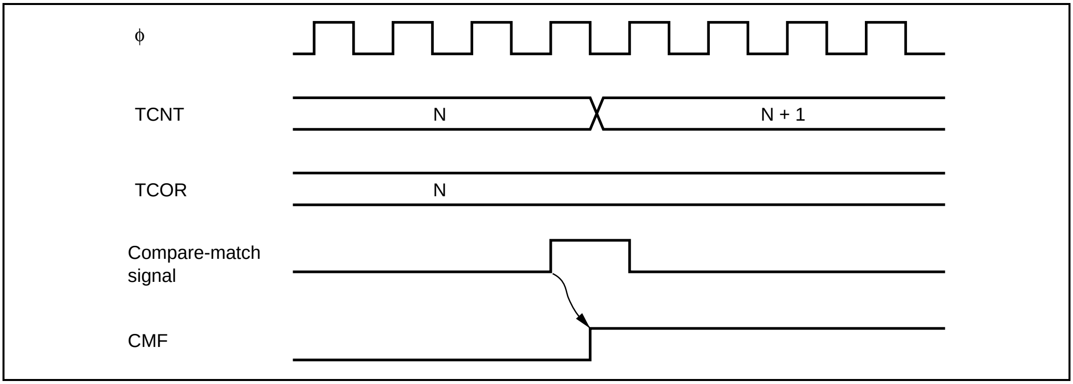


Figure 10.6 Timing of CMF Setting at Compare-Match

10.5.3 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 10.7 shows the timing of timer output when the output is set to toggle by a compare-match A signal.

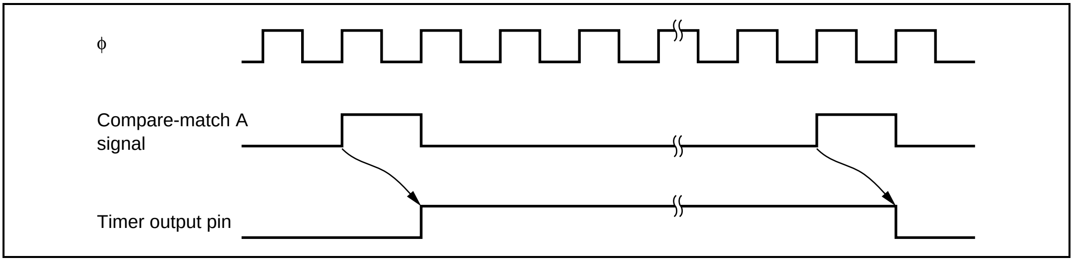


Figure 10.7 Timing of Toggled Timer Output by Compare-Match A Signal

10.5.4 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 10.8 shows the timing of clearing the counter by a compare-match.

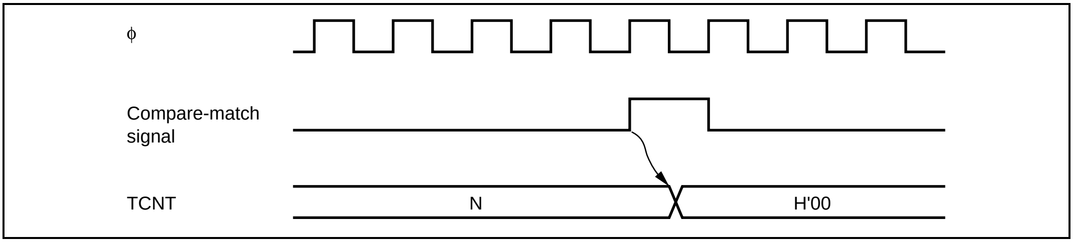


Figure 10.8 Timing of Counter Clear by Compare-Match

10.5.5 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 10.9 shows the timing of clearing the counter by an external reset input.

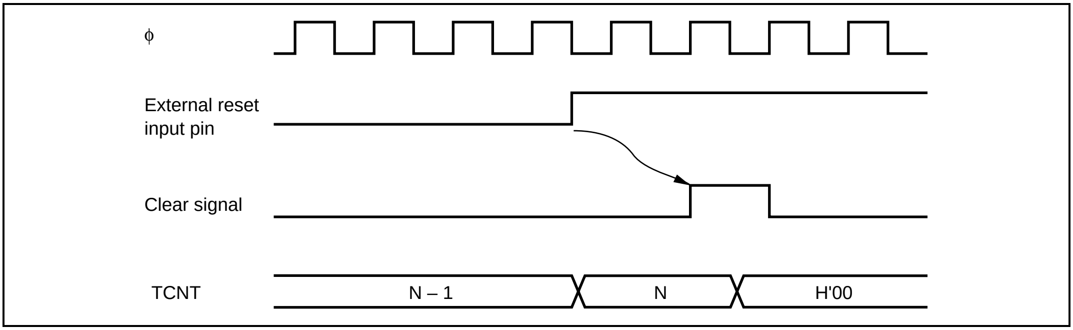


Figure 10.9 Timing of Counter Clear by External Reset Input

10.5.6 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 10.10 shows the timing of OVF flag setting.

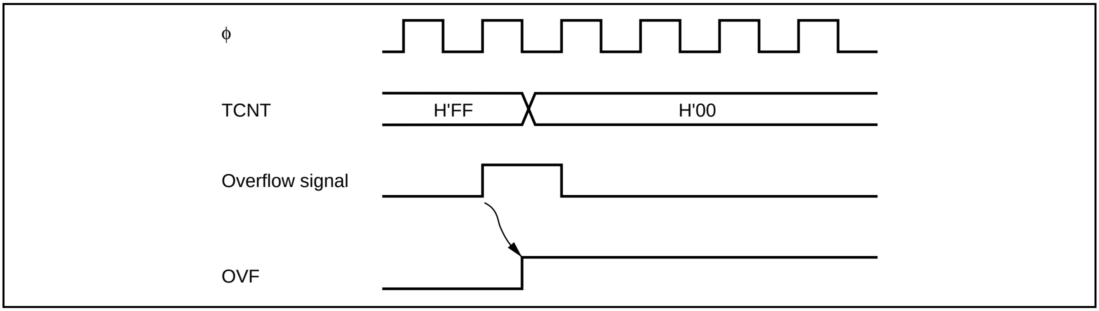


Figure 10.10 Timing of OVF Flag Setting

10.6 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, the 16-bit count mode or compare-match count mode is available.

10.6.1 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 TMR_0 occupying the upper 8 bits and TMR_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.

10.6.2 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 TMR_0. TMR_0 and TMR_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 or TMR_0 and TMR_1.

10.7 TMR_Y and TMR_X Cascaded Connection

If bits CKS2 to CKS0 in either TCR_Y or TCR_X are set to B'100, the 8-bit timers of the two channels are cascaded. With this configuration, 16-bit count mode or compare-match count mode can be selected by the settings of the CKSX and CKSY bits in TCRXY.

10.7.1 16-Bit Count Mode

When bits CKS2 to CKS0 in TCR_Y are set to B'100 and the CKSY bit in TCRXY is set to 1, the timer functions as a single 16-bit timer with TMR_Y occupying the upper eight bits and TMR_X occupying the lower 8 bits.

- Setting of compare-match flags
  - The CMF flag in TCSR_Y is set to 1 when an upper 8-bit compare-match occurs.
  - The CMF flag in TCSR_X is set to 1 when a lower 8-bit compare-match occurs.
- Counter clear specification
  - If the CCLR1 and CCLR0 bits in TCR_Y have been set for counter clear at compare-match, only the upper eight bits of TCNT_Y are cleared. The upper eight bits of TCNT_Y are also cleared when counter clear by the TMRIY pin has been set.
  - The settings of the CCLR1 and CCLR0 bits in TCR_X are enabled, and the lower 8 bits of TCNT_X can be cleared by the counter.
- Pin output
  - Control of output from the TMOY pin by bits OS3 to OS0 in TCSR_Y is in accordance with the upper 8-bit compare-match conditions.
  - Control of output from the TMOX pin by bits OS3 to OS0 in TCSR_X is in accordance with the lower 8-bit compare-match conditions.

Note: The program development tool (emulator) does not support 16-bit count mode.

10.7.2 Compare-Match Count Mode

When bits CKS2 to CKS0 in TCR_X are set to B'100 and the CKSX bit in TCRXY is set to 1, TCNT_X counts the occurrence of compare-match A for TMR_Y. TMR_X and TMR_Y 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.

Note: The program development tool (emulator) does not support compare-match count mode.

10.7.3 Input Capture Operation

TMR_X has input capture registers (TICRR and TICRF). A narrow pulse width can be measured with TICRR and TICRF, using a single capture. If the falling edge of TMRIX (TMR_X input capture input signal) is detected after its rising edge has been detected, the value of TCNT_X at that time is transferred to both TICRR and TICRF.

Input Capture Signal Input Timing: Figure 10.11 shows the timing of the input capture operation.

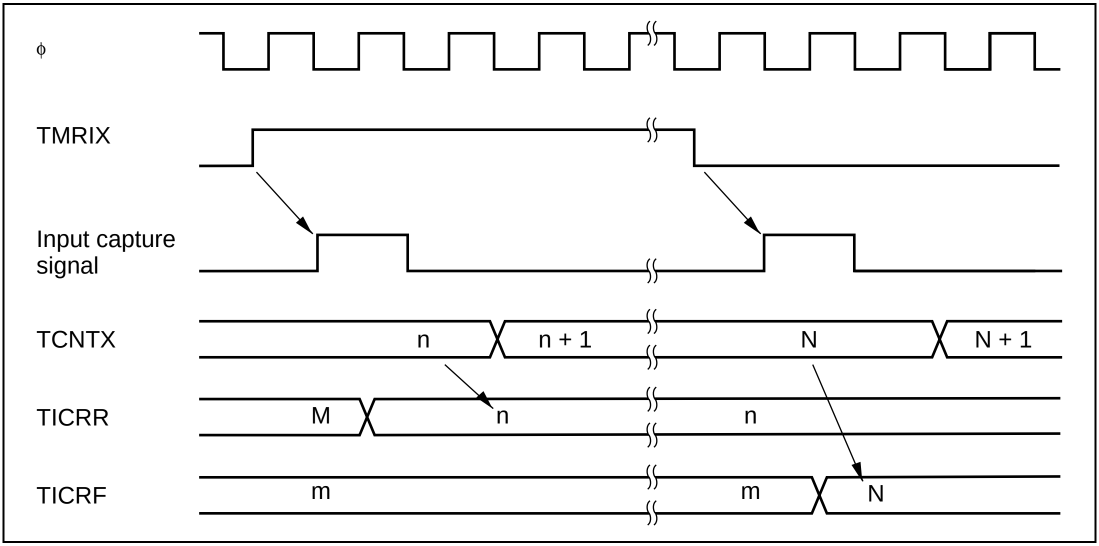


Figure 10.11 Timing of Input Capture Operation

If the input capture signal is input while TICRR and TICRF are being read, the input capture signal is delayed by one system clock (φ) cycle. Figure 10.12 shows the timing of this operation.

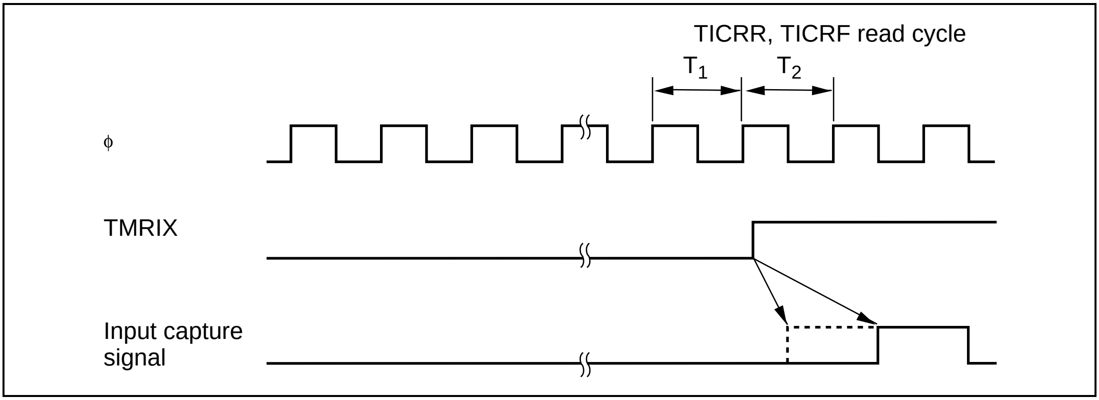


Figure 10.12 Timing of Input Capture Signal (Input capture signal is input during TICRR and TICRF read)

Selection of Input Capture Signal Input: TMRIX (input capture input signal of TMR_X) is selected according to the setting of the ICST bit in TCONRI of the timer connection. The input capture signal selection is shown in table 10.4.

Table 10.4 Input Capture Signal Selection

TCONRI Bit 4 ICST	Description
0	Input capture function not used
1	TMIX pin input selection

10.8 Interrupt Sources

TMR_0, TMR_1, and TMR_Y can generate three types of interrupts: CMIA, CMIB, and OVI. TMR_X can generate an ICIX interrupt. Table 10.5 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.

Table 10.5 Interrupt Sources of 8-Bit Timers TMR_0, TMR_1, TMR_Y, and TMR_X

Channel	Name	Interrupt Source	Interrupt Flag	Interrupt Priority
TMR_0	CMIA0	TCORA_0 compare-match	CMFA	High
	CMIB0	TCORB_0 compare-match	CMFB	↑
	OVI0	TCNT_0 overflow	OVF	
TMR_1	CMIA1	TCORA_1 compare-match	CMFA	
	CMIB1	TCORB_1 compare-match	CMFB	
	OVI1	TCNT_1 overflow	OVF	
TMR_Y	CMIAY	TCORA_Y compare-match	CMFA	
	CMIBY	TCORB_Y compare-match	CMFB	
	OVIY	TCNT_Y overflow	OVF	
TMR_X	ICIX	Input capture	ICF	Low

10.9 Usage Notes

10.9.1 Conflict between TCNT Write and Counter Clear

If a counter clear signal is generated during the T_2 state of a TCNT write cycle as shown in figure 10.13, clearing takes priority and the counter write is not performed.

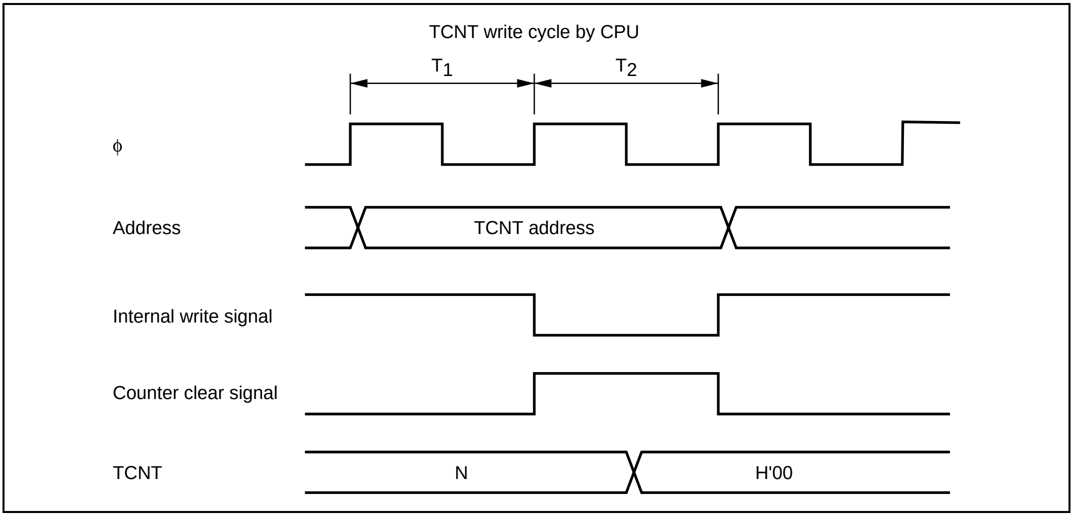


Figure 10.13 Conflict between TCNT Write and Clear

10.9.2 Conflict between TCNT Write and Count-Up

If a count-up occurs during the T_2 state of a TCNT write cycle as shown in figure 10.14, the counter write takes priority and the counter is not incremented.

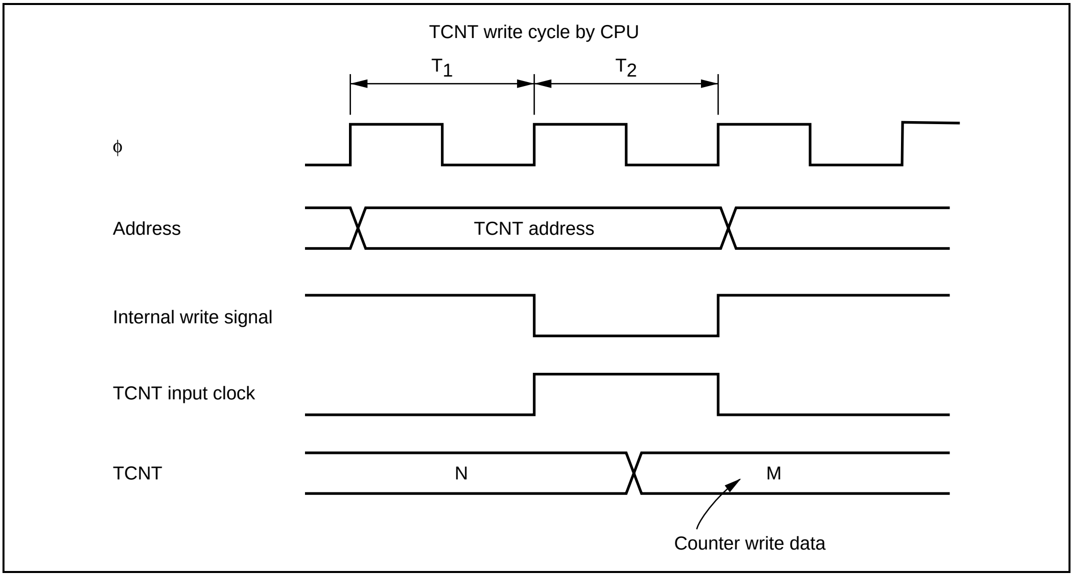


Figure 10.14 Conflict between TCNT Write and Count-Up

10.9.3 Conflict between TCOR Write and Compare-Match

If a compare-match occurs during the T_2 state of a TCOR write cycle as shown in figure 10.15, the TCOR write takes priority and the compare-match signal is disabled. With TMR_X, a TICR 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.

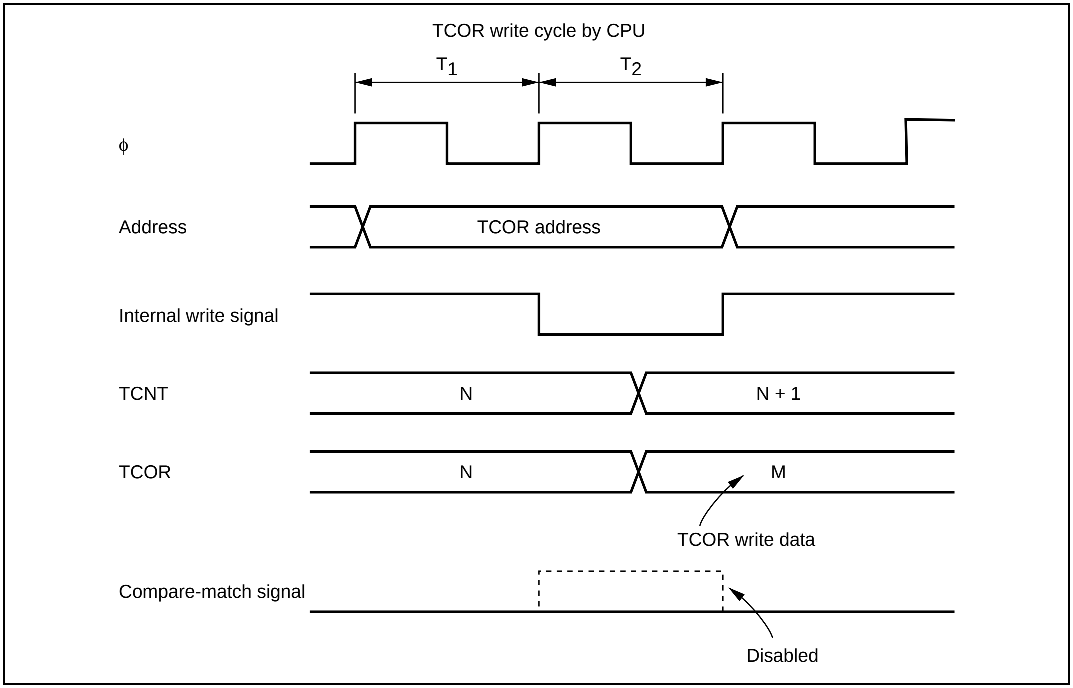


Figure 10.15 Conflict between TCOR Write and Compare-Match

10.9.4 Conflict between Compare-Matches A and B

If compare-matches A and B occur at the same time, the operation follows the output status that is defined for compare-match A or B, according to the priority of the timer output shown in table 10.6.

Table 10.6 Timer Output Priorities

Output Setting	Priority
Toggle output	High
1 output	↑
0 output	↑
No change	Low

10.9.5 Switching of Internal Clocks and TCNT Operation

TCNT may increment erroneously when the internal clock is switched over. Table 10.7 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 10.7, 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 10.7 Switching of Internal Clocks and TCNT Operation

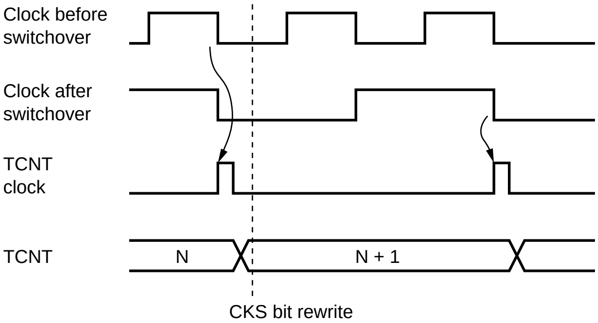
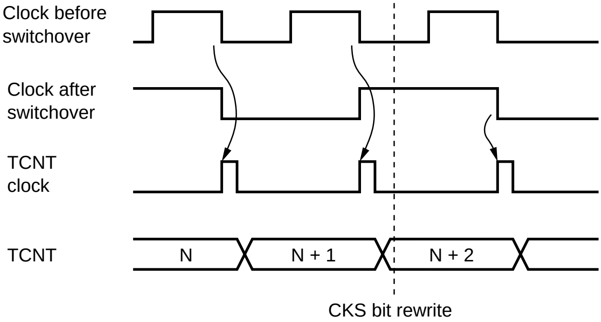
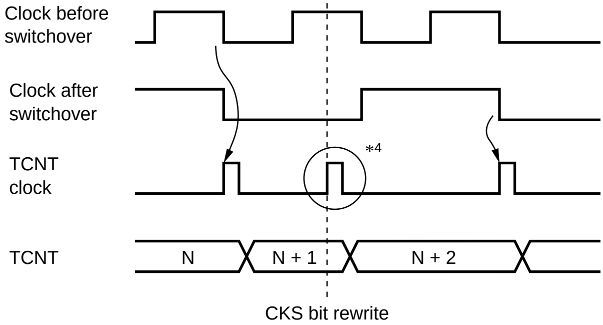
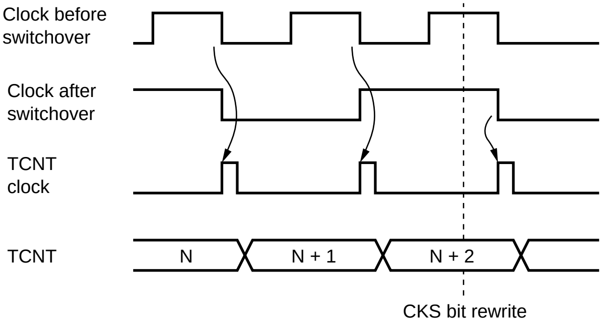
No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	TCNT Clock Operation
1	Clock switching from low to low level*[1]	 Clock before switchover Clock after switchover TCNT clock TCNT N N + 1 CKS bit rewrite

Table 10.7 Switching of Internal Clocks and TCNT Operation (cont)

No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	TCNT Clock Operation
2	Clock switching from low to high level[*2]	 Clock before switchover Clock after switchover TCNT clock TCNT N N + 1 N + 2 CKS bit rewrite
3	Clock switching from high to low level[*3]	 Clock before switchover Clock after switchover *4 TCNT clock TCNT N N + 1 N + 2 CKS bit rewrite
4	Clock switching from high to high level	 Clock before switchover Clock after switchover TCNT clock TCNT N N + 1 N + 2 CKS bit rewrite

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.

10.9.6 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, and TCNT_X and TCNT_Y are not generated, and thus the counters will stop operating. Simultaneous setting of these two modes should therefore be avoided.

10.9.7 Module Stop Mode Setting

TMR operation can be enabled or disabled using the module stop control register. The initial setting is for TMR operation to be halted. Register access is enabled by canceling the module stop mode. For details, refer to section 19, Power-Down Modes.

Section 11 Watchdog Timer (WDT)

This LSI incorporates two watchdog timer channels (WDT_0 and WDT_1). The watchdog timer can generate an internal reset signal or an internal NMI interrupt signal if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow. Simultaneously, it can output an overflow signal ($\overline{\text{RESO}}$) externally.

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_0 and WDT_1 is shown in figure 11.1.

11.1 Features

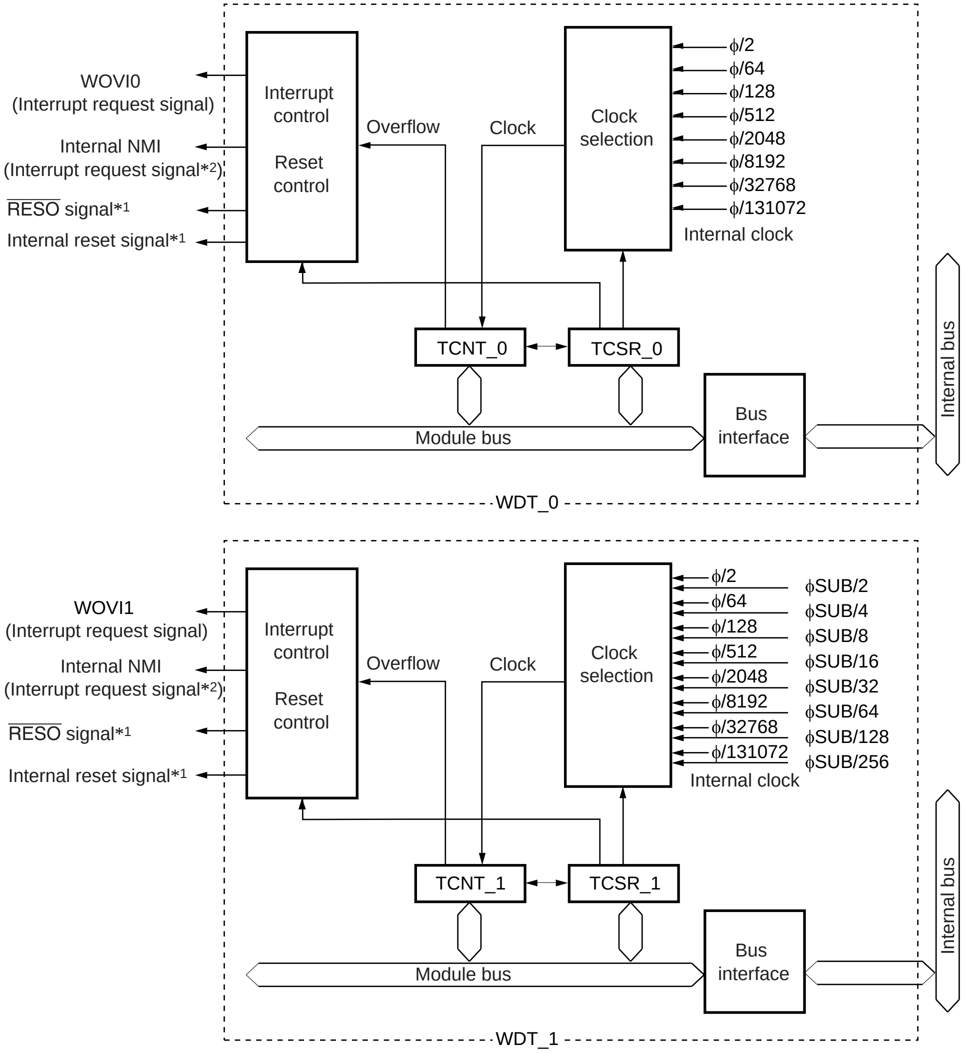
- Selectable from eight (WDT_0) or 16 (WDT_1) counter input clocks.
- Switchable between watchdog timer mode and interval timer mode

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.

Interval Timer Mode:

- If the counter overflows, an interval timer interrupt (WOVI) is generated.



Legend

TCSR_0	: Timer control/status register_0
TCNT_0	: Timer counter_0
TCSR_1	: Timer control/status register_1
TCNT_1	: Timer counter_1

Notes: 1. The $\overline{\text{RESO}}$ signal outputs the low level signal when the internal reset signal is generated due to a TCNT overflow of either WDT_0 or WDT_1. The internal reset signal first resets the WDT in which the overflow has occurred first.

2. The internal NMI interrupt signal can be independently output from either WDT_0 or WDT_1. The interrupt controller does not distinguish the NMI interrupt request from WDT_0 from that from WDT_1.

Figure 11.1 Block Diagram of WDT

11.2 Input/Output Pins

The WDT has the pins listed in table 11.1.

Table 11.1 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

11.3 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, refer to section 11.6.1, Notes on Register Access. For details on the system control register, refer to section 3.2.2, System Control Register (SYSCR).

- Timer counter (TCNT)
- Timer control/status register (TCSR)

11.3.1 Timer Counter (TCNT)

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

TCNT is initialized to H'00 when the TME bit in the timer control/status register (TCSR) is cleared to 0.

11.3.2 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)*[1]	Overflow Flag Indicates that TCNT has overflowed (changes from H’FF to H’00). [Setting condition] When TCNT overflows (changes from H’FF to H’00) However, when internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset. [Clearing conditions] • When TCSR is read when OVF = 1*[2], then 0 is written to OVF • When 0 is written to TME
6	WT/$\overline{\text{IT}}$	0	R/W	Timer Mode Select Selects whether the WDT is used as a watchdog timer or interval timer. 0: Interval timer mode 1: Watchdog timer mode
5	TME	0	R/W	Timer Enable When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00.
4	—	0	R/(W)	Reserved The initial value should not be changed.
3	RST/$\overline{\text{NMI}}$	0	R/W	Reset or NMI Selects to request an internal reset or an NMI interrupt when TCNT has overflowed. 0: An NMI interrupt is requested 1: An internal reset is requested

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 ø = 10 MHz is enclosed in parentheses. 000: ϕ/2 (frequency: 51.2 µs) 001: ϕ/64 (frequency: 1.64 ms) 010: ϕ/128 (frequency: 3.28 ms) 011: ϕ/512 (frequency: 13.1 ms) 100: ϕ/2048 (frequency: 52.4 ms) 101: ϕ/8192 (frequency: 209.7 ms) 110: ϕ/32768 (frequency: 0.84 s) 111: ϕ/131072 (frequency: 3.36 s)
0	CKS0	0	R/W	

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.

- TCSR_1

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*[1]	Overflow Flag Indicates that TCNT has overflowed (changes from H’FF to H’00). [Setting condition] When TCNT overflows (changes from H’FF to H’00) However, when internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset. [Clearing conditions] When TCSR is read when OVF = 1*[2], then 0 is written to OVF When 0 is written to TME
6	WT/$\overline{\text{IT}}$	0	R/W	Timer Mode Select Selects whether the WDT is used as a watchdog timer or interval timer. 0: Interval timer mode 1: Watchdog timer mode
5	TME	0	R/W	Timer Enable When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00.

Bit	Bit Name	Initial Value	R/W	Description
4	PSS	0	R/W	Prescaler Select Selects the clock source to be input to TCNT. 0: Counts the divided cycle of ø–based prescaler (PSM) 1: Counts the divided cycle of øSUB–based prescaler (PSS)
3	RST/$\overline{\text{NMI}}$	0	R/W	Reset or NMI Selects to request an internal reset or an NMI interrupt when TCNT has overflowed. 0: An NMI interrupt is requested 1: An internal reset is requested
2 1 0	CKS2 CKS1 CKS0	0 0 0	R/W R/W R/W	Clock Select 2 to 0 Selects the clock source to be input to TCNT. The overflow cycle for ø = 10 MHz and øSUB = 32.768 kHz is enclosed in parentheses. When PSS = 0: 000: ϕ/2 (frequency: 51.2 μs) 001: ϕ/64 (frequency: 1.64 ms) 010: ϕ/128 (frequency: 3.28 ms) 011: ϕ/512 (frequency: 13.1 ms) 100: ϕ/2048 (frequency: 52.4 ms) 101: ϕ/8192 (frequency: 209.7 ms) 110: ϕ/32768 (frequency: 0.84 s) 111: ϕ/131072 (frequency: 3.36 s) When PSS = 1: 000: ϕSUB/2 (cycle: 15.6 ms) 001: ϕSUB/4 (cycle: 31.3 ms) 010: ϕSUB/8 (cycle: 62.5 ms) 011: ϕSUB/16 (cycle: 125 ms) 100: ϕSUB/32 (cycle: 250 ms) 101: ϕSUB/64 (cycle: 500 ms) 110: ϕSUB/128 (cycle: 1 s) 111: ϕ/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.

11.4 Operation

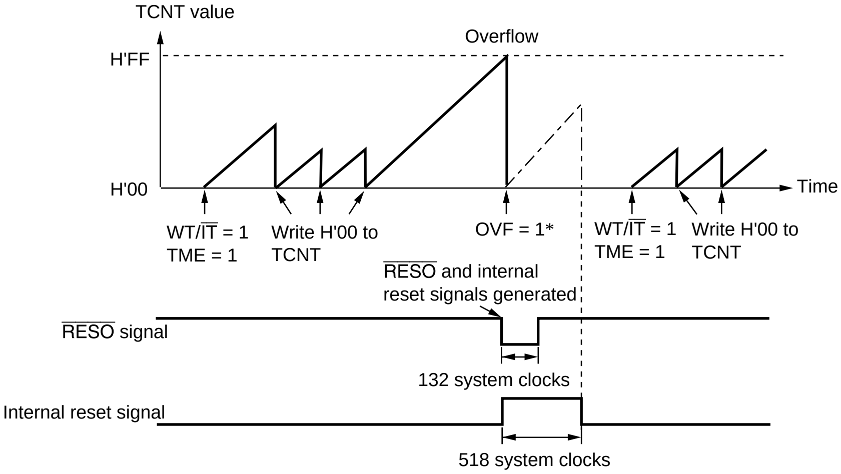
11.4.1 Watchdog Timer Mode

To use the WDT as a watchdog timer, set the WT/$\overline{\text{IT}}$ 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 be writing H'00) before overflows occurs.

If the RST/$\overline{\text{NMI}}$ 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 11.2. If the RST/$\overline{\text{NMI}}$ 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.



WT/$\overline{\text{IT}}$: Timer mode select bit
TME : Timer enable bit
OVF : Overflow flag

Note: * After the OVF bit becomes 1, it is cleared to 0 by an internal reset.
The XRST bit is also cleared to 0.

Figure 11.2 Watchdog Timer Mode (RST/$\overline{\text{NMI}}$ = 1) Operation

11.4.2 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 11.3. 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 in figure 11.4.

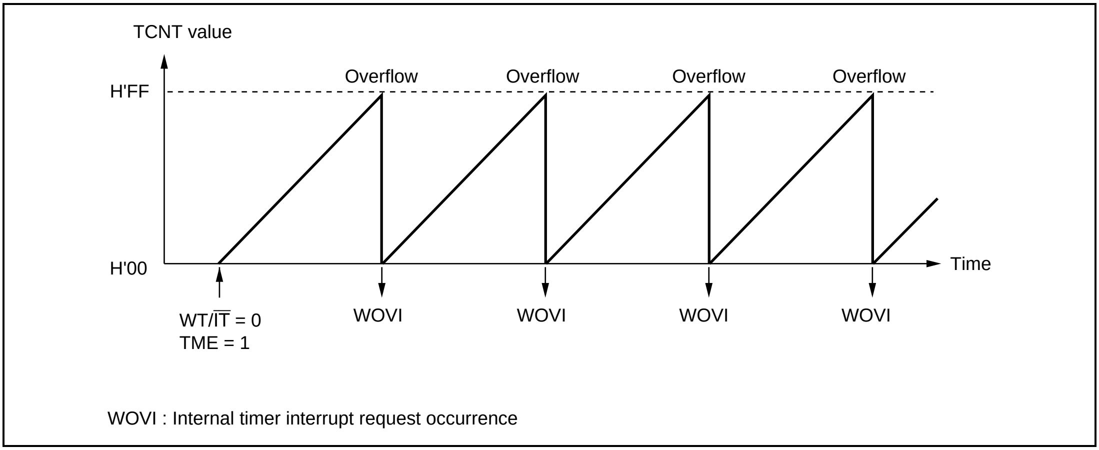


Figure 11.3 Interval Timer Mode Operation

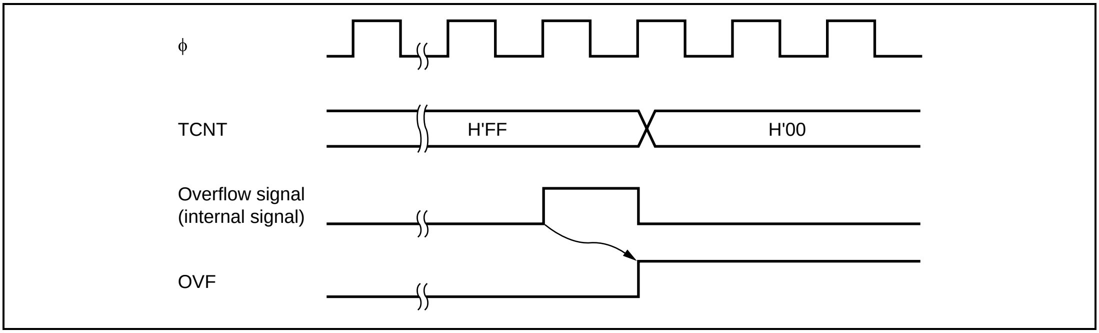


Figure 11.4 OVF Flag Set Timing

11.4.3 $\overline{\text{RESO}}$ 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 11.5.

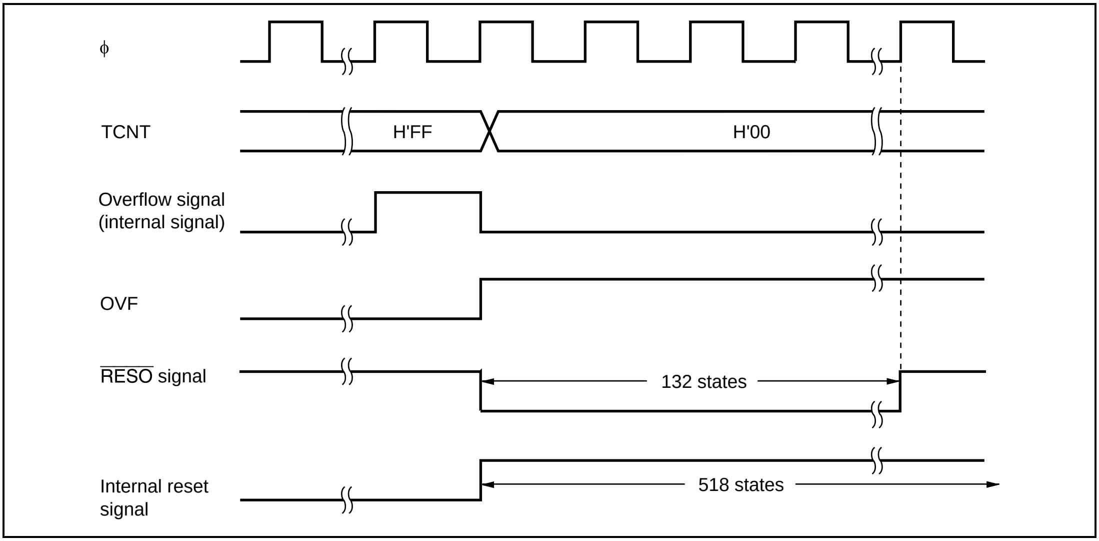

ϕ
TCNT
H'FF
H'00
Overflow signal (internal signal)
OVF
$\overline{\text{RESO}}$ signal
132 states
Internal reset signal
518 states

Figure 11.5 Output Timing of $\overline{\text{RESO}}$ signal

11.5 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 11.2 WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag
WOVI	TCNT overflow	OVF

11.6 Usage Notes

11.6.1 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 11.6 to write to TCNT or TCSR. To write to TCNT, the upper 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 upper bytes must contain the value H’A5 and the lower bytes must contain the write data.

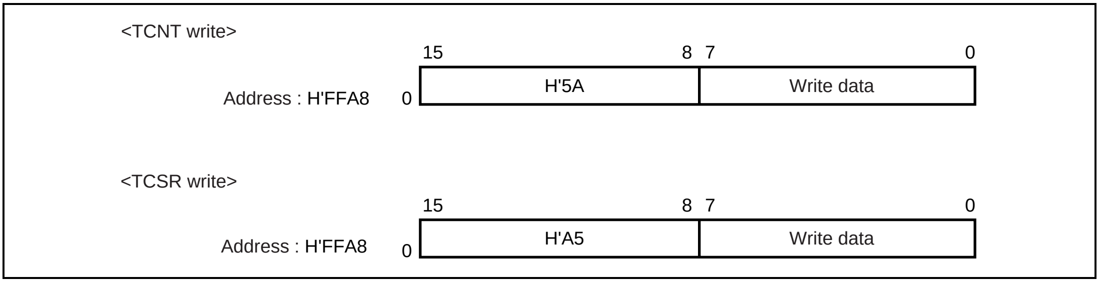


Figure 11.6 Writing to TCNT and TCSR (WDT_0)

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.

11.6.2 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 11.7 shows this operation.

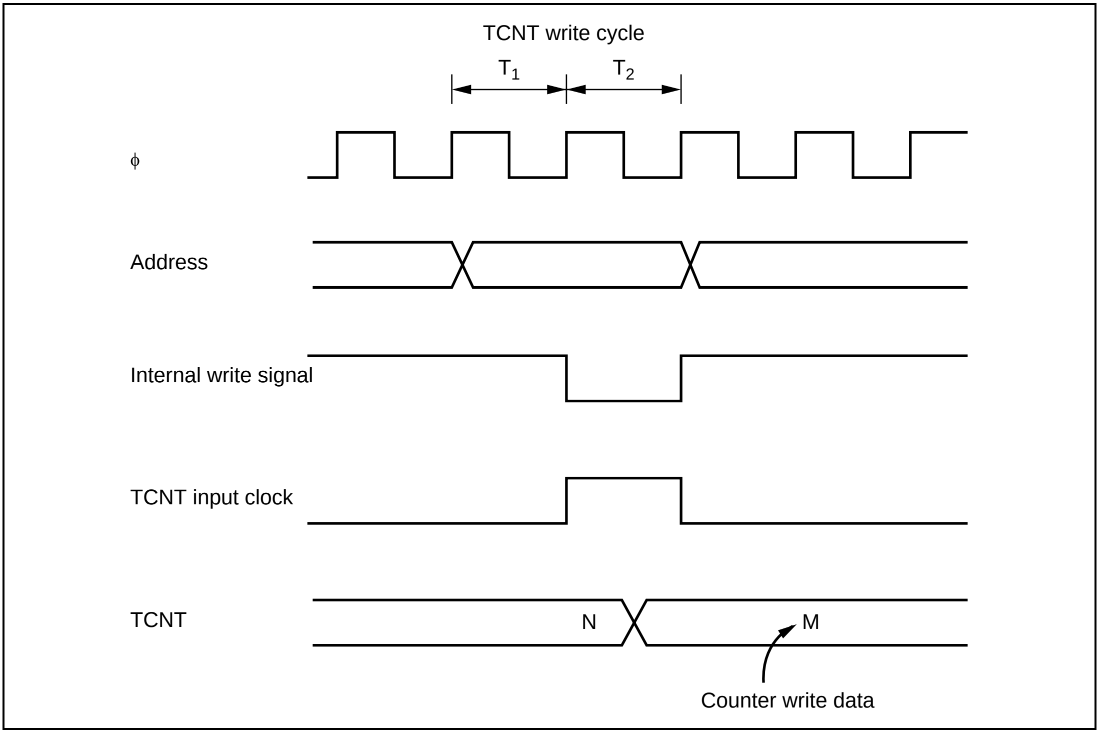


Figure 11.7 Conflict between TCNT Write and Increment

11.6.3 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.

11.6.4 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.

11.6.5 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 11.8.

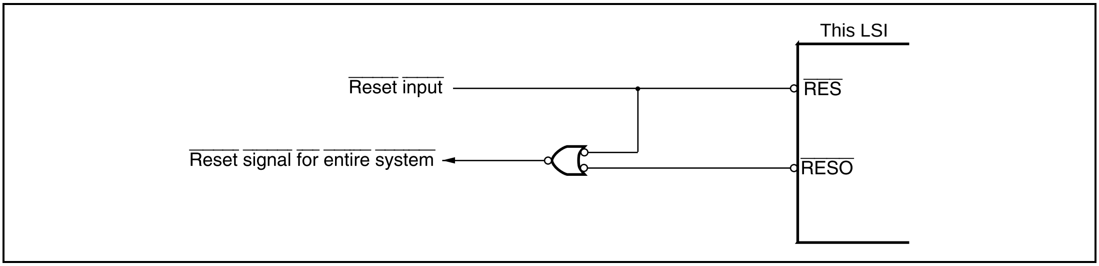

This LSI
$\overline{\text{Reset}}$ $\overline{\text{input}}$
$\overline{\text{RES}}$
$\overline{\text{Reset signal for entire system}}$
$\overline{\text{RESO}}$

Figure 11.8 Sample Circuit for Resetting System by $\overline{\text{RESO}}$ Signal

11.6.6 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 oscillator 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 12 Serial Communication Interface (SCI)

This LSI has a serial communication interface (SCI). 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) in asynchronous mode.

12.1 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.
- The on-chip baud rate generator allows any bit rate to be selected

  An external clock can be selected as a transfer clock source.
- 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.

Asynchronous Mode:

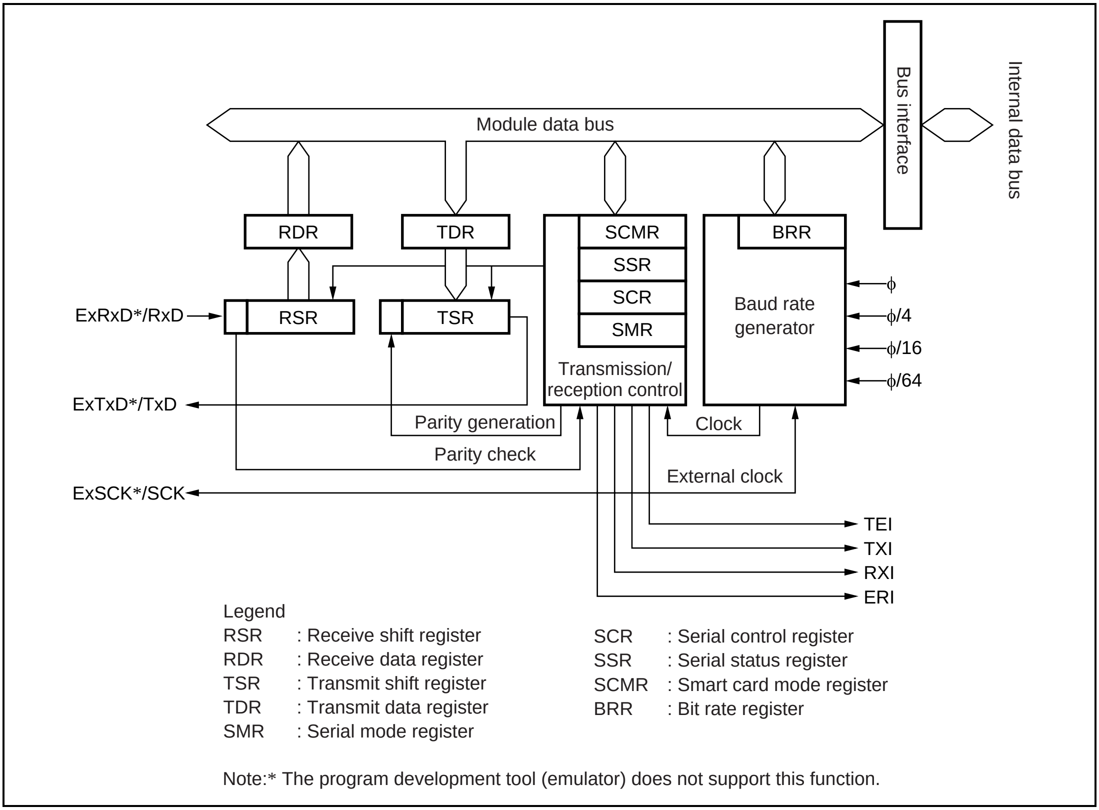
- Data length: 7 or 8 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the RxD pin level directly in case of a framing error

SCI0022C_000020020800

Clocked Synchronous Mode:

- Data length: 8 bits
- Receive error detection: Overrun errors
- Serial data communication with other LSIs that have the clock synchronized communication function

A block diagram of the SCI is shown in figure 12.1.



Legend

RSR	: Receive shift register
RDR	: Receive data register
TSR	: Transmit shift register
TDR	: Transmit data register
SMR	: Serial mode register
SCR	: Serial control register
SSR	: Serial status register
SCMR	: Smart card mode register
BRR	: Bit rate register

Note:* The program development tool (emulator) does not support this function.

Figure 12.1 Block Diagram of SCI

12.2 Input/Output Pins

Table 12.1 shows the input/output pins for each SCI channel.

Table 12.1 Pin Configuration

Channel	Symbol*[1]	Input/Output	Function
1	SCK1/ ExSCK1*[2]	Input/Output	Channel 1 clock input/output
	RxD1/ ExRxD1*[2]	Input	Channel 1 receive data input
	TxD1/ ExTxD1*[2]	Output	Channel 1 transmit data output

Notes: 1. Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

2. The program development tool (emulator) does not support this function.

12.3 Register Descriptions

The SCI has the following registers.

- 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)
- Serial interface mode register (SCMR)
- Bit rate register (BRR)
- Serial pin select register (SPSR)*

Note:* The program development tool (emulator) does not support this function.

12.3.1 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.

12.3.2 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 can 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. RDR is initialized to H'00.

12.3.3 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. TDR is initialized to H'FF.

12.3.4 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.

12.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the on-chip baud rate generator clock source.

Bit	Bit Name	Initial Value	R/W	Description
7	C/$\overline{A}$	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/$\overline{E}$	0	R/W	Parity Mode (enabled only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity.

Bit	Bit Name	Initial Value	R/W	Description
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.
2	MP	0	R/W	Multiprocessor Mode (enabled only in asynchronous mode) When this bit is set to 1, the multiprocessor communication function is enabled. The PE bit and O/$\overline{\text{E}}$ bit settings are invalid in multiprocessor mode.
1	CKS1	0	R/W	Clock Select 1,0
0	CKS0	0	R/W	These bits select the clock source for the on-chip baud rate generator. 00: φ clock (n = 0) 01: φ/4 clock (n = 1) 10: φ/16 clock (n = 2) 11: φ/64 clock (n = 3) For the relation between the bit rate register setting and the baud rate, see section 12.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR.

12.3.6 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, refer to section 12.7, Interrupt Sources.

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, refer to section 12.5, 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	These bits select the clock source and SCK pin function. Asynchronous mode 00: Internal clock (SCK pin functions as I/O port.) 01: Internal clock (Outputs a clock of the same frequency as the bit rate from the SCK pin.) 1X: External clock (Inputs a clock with a frequency 16 times the bit rate from the SCK pin.) Clocked synchronous mode 0X: Internal clock (SCK pin functions as clock output.) 1X: External clock (SCK pin functions as clock input.)

Legend

X: Don’t care

12.3.7 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.

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] • When the TE bit in SCR is 0 • When data is transferred from TDR to TSR and TDR is ready for data write [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates that receive data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing conditions] • When 0 is written to RDRF after reading RDRF = 1 The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.
5	ORER	0	R/(W)*	Overrun Error [Setting condition] • When the next data is received while RDRF = 1 [Clearing condition] • When 0 is written to ORER after reading ORER = 1

Bit	Bit Name	Initial Value	R/W	Description
4	FER	0	R/(W)*	Framing Error [Setting condition] • When the stop bit is 0 [Clearing condition] • When 0 is written to FER after reading FER = 1 In 2-stop-bit mode, only the first stop bit is checked.
3	PER	0	R/(W)*	Parity Error [Setting condition] • When a parity error is detected during reception [Clearing condition] • When 0 is written to PER after reading PER = 1
2	TEND	1	R	Transmit End [Setting conditions] • When the TE bit in SCR is 0 • When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1
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.

12.3.8 Serial Interface Mode Register (SCMR)

SCMR selects SCI functions 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	Data Transfer Direction Selects the serial/parallel conversion format. 0: TDR contents are transmitted with LSB-first. Receive data is stored as LSB first in RDR. 1: TDR contents are transmitted with MSB-first. Receive data is stored 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	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/$\overline{E}$ 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	Serial Communication Interface Mode Select: 0: Normal asynchronous or clocked synchronous mode 1: Reserved mode

12.3.9 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 12.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clocked synchronous 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 12.2 Relationships between N Setting in BRR and Bit Rate B

Mode	Bit Rate	Error
Asynchronous mode	$B = \dfrac{\phi \times 10^6}{64 \times 2^{2n-1} \times (N+1)}$	$\text{Error (\%)} = \left\{ \dfrac{\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1 \right\} \times 100$
Clocked synchronous mode	$B = \dfrac{\phi \times 10^6}{64 \times 2^{2n-1} \times (N+1)}$	—

Legend

B: Bit rate (bit/s)

N: BRR setting for baud rate generator ($0 \leq N \leq 255$)

ϕ: Operating frequency (MHz)

n: Determined by the SMR settings shown in the following table.

SMR Setting		
CKS1	**CKS0**	**n**
0	0	0
0	1	1
1	0	2
1	1	3

Table 12.3 shows sample N settings in BRR in normal asynchronous mode. Table 12.4 shows the maximum bit rate settable for each frequency. Table 12.6 shows sample N settings in BRR in clocked synchronous mode. Tables 12.5 and 12.7 show the maximum bit rates with external clock input.

Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (1)

	Operating Frequency ϕ (MHz)											
	2			2.097152			2.4576			3		
Bit Rate (bit/s)	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	—	—	—

	Operating Frequency ϕ (MHz)											
	3.6864			4			4.9152			5		
Bit Rate (bit/s)	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%.

Table 12.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (2)

	Operating Frequency ϕ (MHz)											
	6			6.144			7.3728			8		
Bit Rate (bit/s)	**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	—	—	—

	Operating Frequency ϕ (MHz)					
	9.8304			10		
Bit Rate (bit/s)	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**
110	2	174	–0.26	2	177	–0.25
150	2	127	0.00	2	129	0.16
300	1	255	0.00	2	64	0.16
600	1	127	0.00	1	129	0.16
1200	0	255	0.00	1	64	0.16
2400	0	127	0.00	0	129	0.16
4800	0	63	0.00	0	64	0.16
9600	0	31	0.00	0	32	–1.36
19200	0	15	0.00	0	15	1.73
31250	0	9	–1.70	0	9	0.00
38400	0	7	0.00	0	7	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 12.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

ϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
2	62500	0	0
2.097152	65536	0	0
2.4576	76800	0	0
3	93750	0	0
3.6864	115200	0	0
4	125000	0	0
4.9152	153600	0	0
5	156250	0	0
6	187500	0	0
6.144	192000	0	0
7.3728	230400	0	0
8	250000	0	0
9.8304	307200	0	0
10	312500	0	0

Table 12.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)

ϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
2	0.5000	31250
2.097152	0.5243	32768
2.4576	0.6144	38400
3	0.7500	46875
3.6864	0.9216	57600
4	1.0000	62500
4.9152	1.2288	76800
5	1.2500	78125
6	15.000	93750
6.144	1.5360	96000
7.3728	1.8432	115200
8	2.0000	125000
9.8304	2.4576	153600
10	2.5000	156250

Table 12.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)

Bit Rate (bit/s)	Operating Frequency φ (MHz)							
	2		4		8		10	
	n	N	n	N	n	N	n	N
110	3	70	—	—				
250	2	124	2	249	3	124	—	—
500	1	249	2	124	2	249	—	—
1k	1	124	1	249	2	124	—	—
2.5k	0	199	1	99	1	199	1	249
5k	0	99	0	199	1	99	1	124
10k	0	49	0	99	0	199	0	249
25k	0	19	0	39	0	79	0	99
50k	0	9	0	19	0	39	0	49
100k	0	4	0	9	0	19	0	24
250k	0	1	0	3	0	7	0	9
500k	0	0*	0	1*	0	3	0	4
1M			0	0	0	1		
2.5M							0	0*
5M								

Legend

Blank:	Cannot be set.
—:	Can be set, but there will be a degree of error.
*:	Continuous transfer or reception is not possible.

Table 12.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)

φ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
2	0.3333	333333.3
4	0.6667	666666.7
6	1.0000	1000000.0
8	1.3333	1333333.3
10	1.6667	1666666.7

12.3.10 Serial Pin Select Register (SPSR)

SPSR selects the serial I/O pins. SPSR should be set before initialization. Do not set during communication.

Bit	Bit Name	Initial Value	R/W	Description
7	SPS1	0	R/W	Serial Port Select Selects the serial I/O pins. 0: P86/SCK1, P85/RxD1, P84/TxD1 1: P52/ExSCK1, P51/ExRxD1, P50/ExTxD1
6 to 0	—	All 0	R/W	Reserved The initial value should not be changed.

Note: The program development tool (emulator) does not support SPSR.

12.4 Operation in Asynchronous Mode

Figure 12.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transmit/receive data, a parity bit, and finally stop bits (high level). In asynchronous serial communication, the 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.

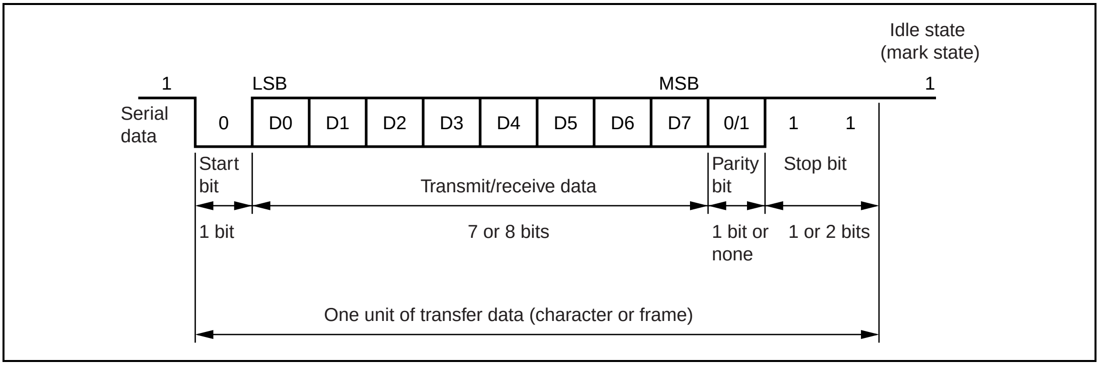


Figure 12.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

12.4.1 Data Transfer Format

Table 12.8 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, refer to section 12.5, Multiprocessor Communication Function.

Table 12.8 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	

12.4.2 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 12.3. Thus the reception margin in asynchronous mode is determined by formula (1) below.

$$M = \{ (0.5 - \frac{1}{2N}) - \frac{D - 0.5}{N} (1 + F) - (L - 0.5) F \} \times 100 \quad [\%] \quad \cdots \quad \text{Formula (1)}$$

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.

$$M = \{0.5 - 1/(2 \times 16)\} \times 100 \quad [\%] = 46.875 \ \%$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

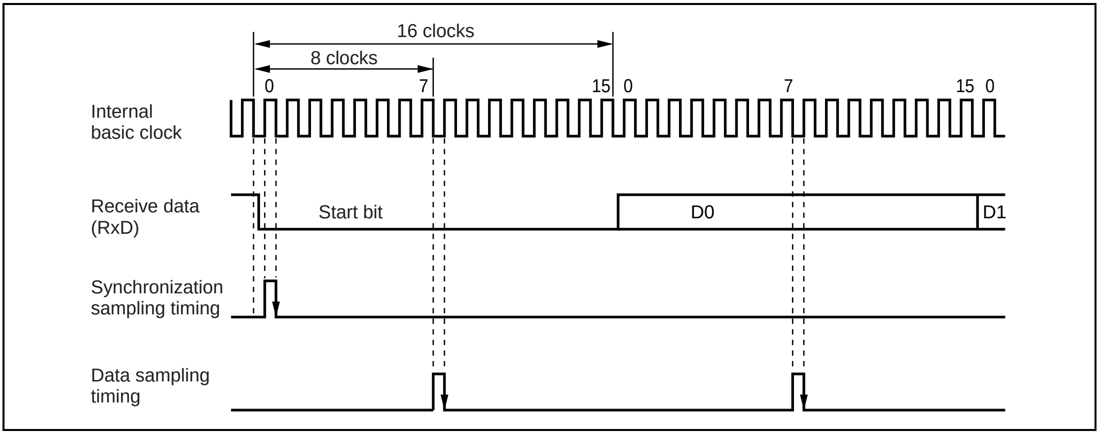


Figure 12.3 Receive Data Sampling Timing in Asynchronous Mode

12.4.3 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 12.4.

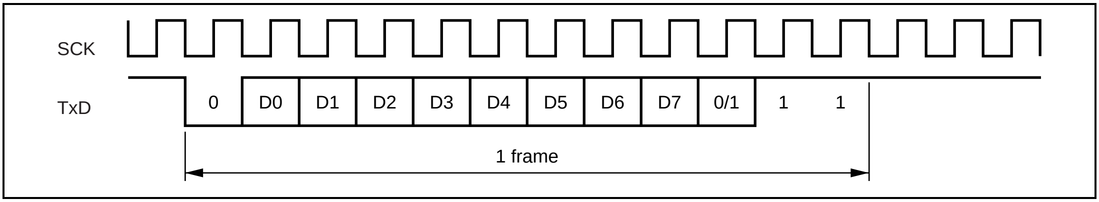
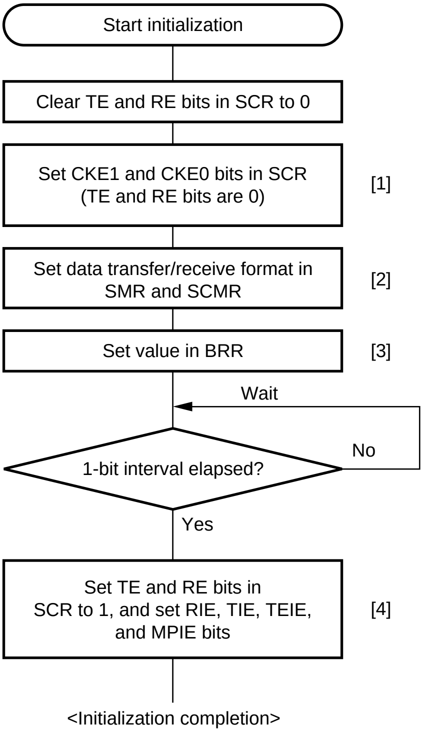


Figure 12.4 Relation between Output Clock and Transmit Data Phase (Asynchronous Mode)

12.4.4 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 12.5. 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 an external clock is used in asynchronous mode, the clock must be supplied even during initialization.


Start initialization
Clear TE and RE bits in SCR to 0
Set CKE1 and CKE0 bits in SCR (TE and RE bits are 0)
[1]
Set data transfer/receive format in SMR and SCMR
[2]
Set value in BRR
[3]
Wait
1-bit interval elapsed?
No
Yes
Set TE and RE bits in SCR to 1, and set RIE, TIE, TEIE, and MPIE bits
[4]
<Initialization completion>

[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock is selected in asynchronous mode, it is output immediately after SCR settings are made.

[2] Set the data transfer/receive format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits.

Setting the TE and RE bits enables the TxD and RxD pins to be used.

Figure 12.5 Sample SCI Initialization Flowchart

12.4.5 Data Transmission (Asynchronous Mode)

Figure 12.6 shows an example of the operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is cleared to 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the 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 12.7 shows a sample flowchart for transmission in asynchronous mode.

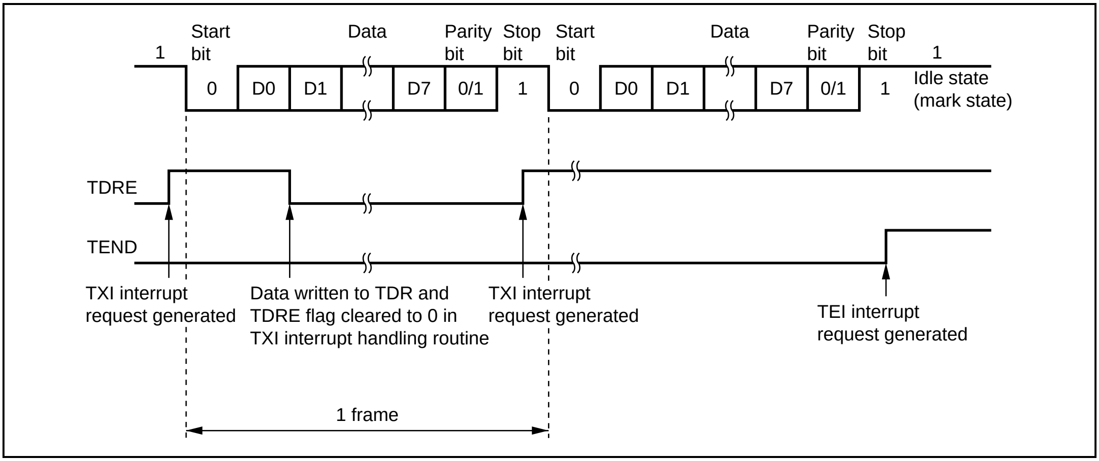
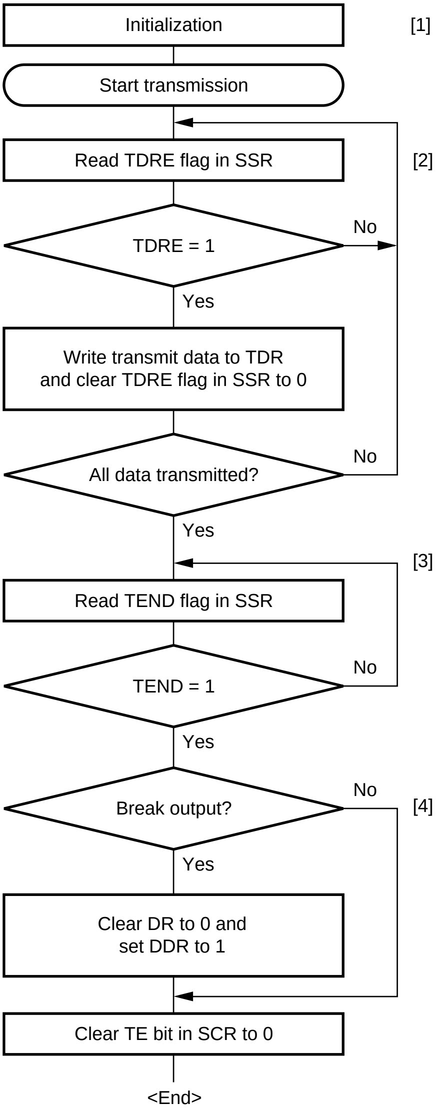

1
Start bit
Data
Parity bit
Stop bit
Start bit
Data
Parity bit
Stop bit
1
0
D0
D1
D7
0/1
1
0
D0
D1
D7
0/1
1
Idle state (mark state)
TDRE
TEND
TXI interrupt request generated
Data written to TDR and TDRE flag cleared to 0 in TXI interrupt handling routine
TXI interrupt request generated
TEI interrupt request generated
1 frame

Figure 12.6 Example of SCI Transmit Operation in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:

To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and clear the TDRE flag to 0.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port corresponding to the TxD pin to 1, clear DR to 0, then clear the TE bit in SCR to 0.

Figure 12.7 Sample Serial Transmission Flowchart

12.4.6 Serial Data Reception (Asynchronous Mode)

Figure 12.8 shows an example of the operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, 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.

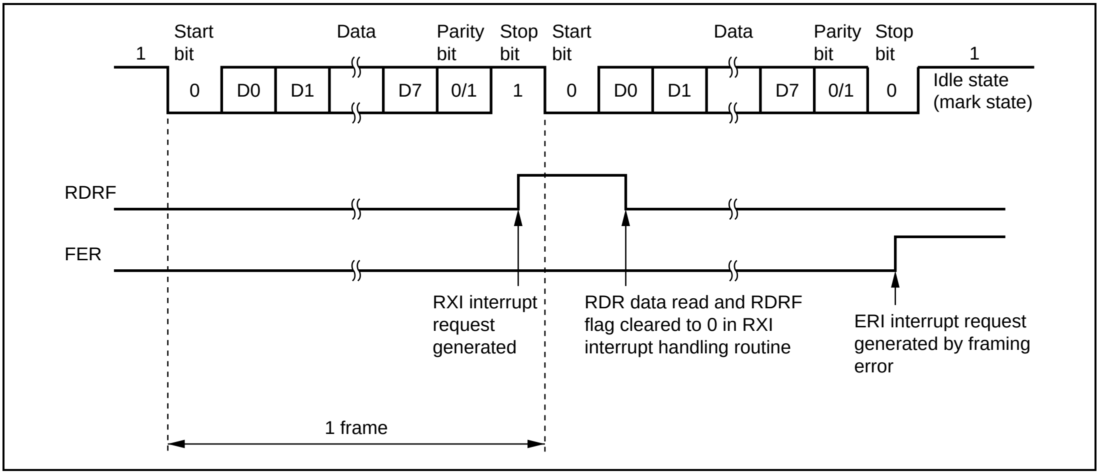

1
Start bit
0
D0
D1
Data
D7
Parity bit
0/1
Stop bit
1
Start bit
0
D0
D1
Data
D7
Parity bit
0/1
Stop bit
0
1
Idle state (mark state)
RDRF
FER
RXI interrupt request generated
RDR data read and RDRF flag cleared to 0 in RXI interrupt handling routine
ERI interrupt request generated by framing error
1 frame

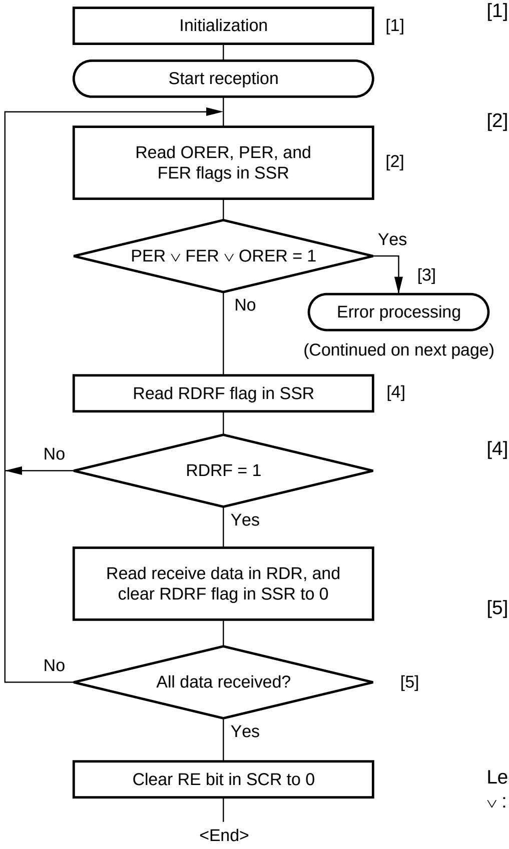
Figure 12.8 Example of SCI Receive Operation in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)

Table 12.9 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 12.9 shows a sample flow chart for serial data reception.

Table 12.9 SSR Status Flags and Receive Data Handling

SSR Status Flag					
RDRF*	**ORER**	**FER**	**PER**	**Receive Data**	**Receive Error Type**
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

Note:* The RDRF flag retains the state it had before data reception.



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing and break detection:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.

[4] SCI status check and receive data read:
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag, read RDR, and clear the RDRF flag to 0.

Legend:
∨ : Logical OR

Figure 12.9 Sample Serial Reception Flowchart (1)

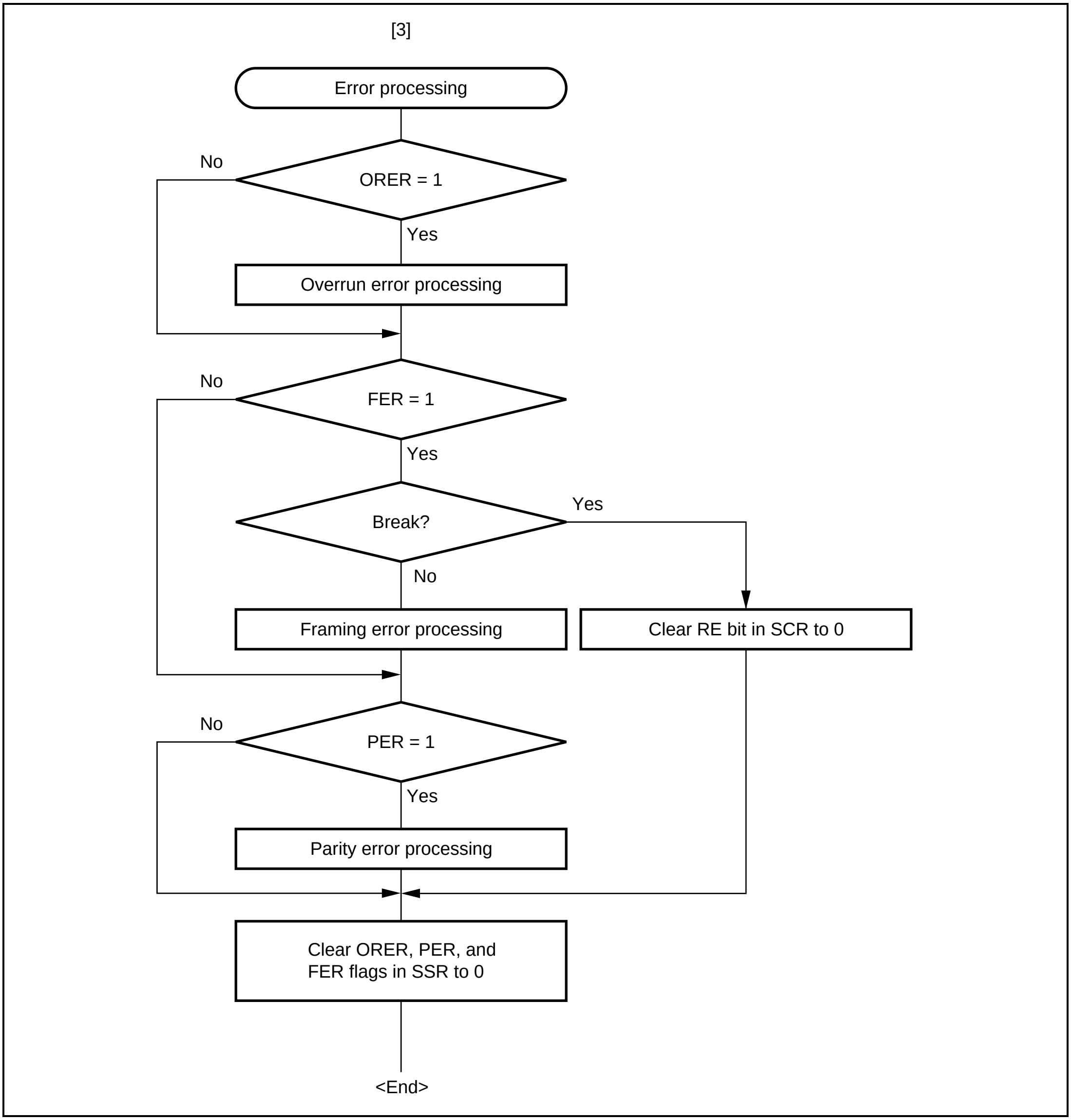


Figure 12.9 Sample Serial Reception Flowchart (2)

12.5 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 12.10 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.

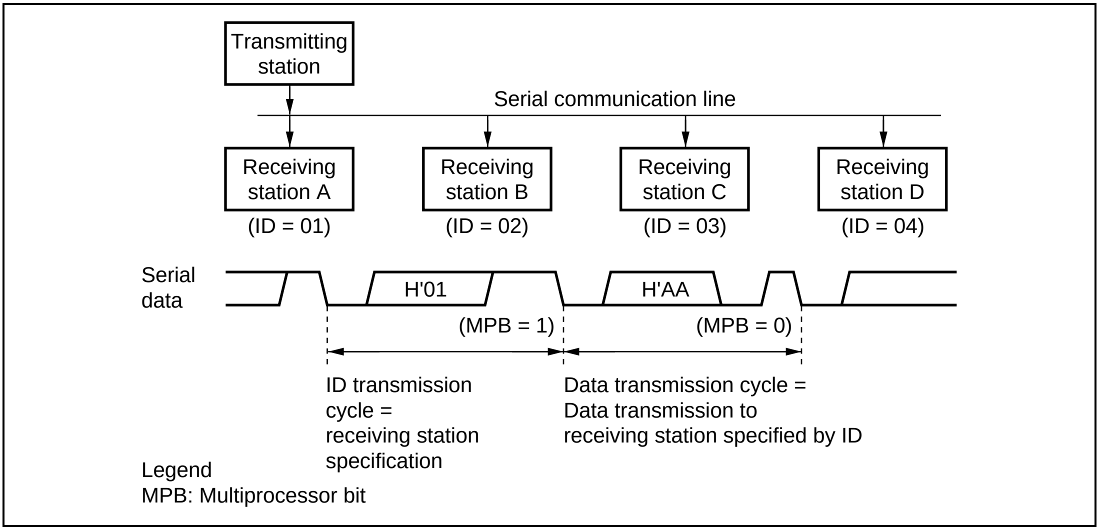
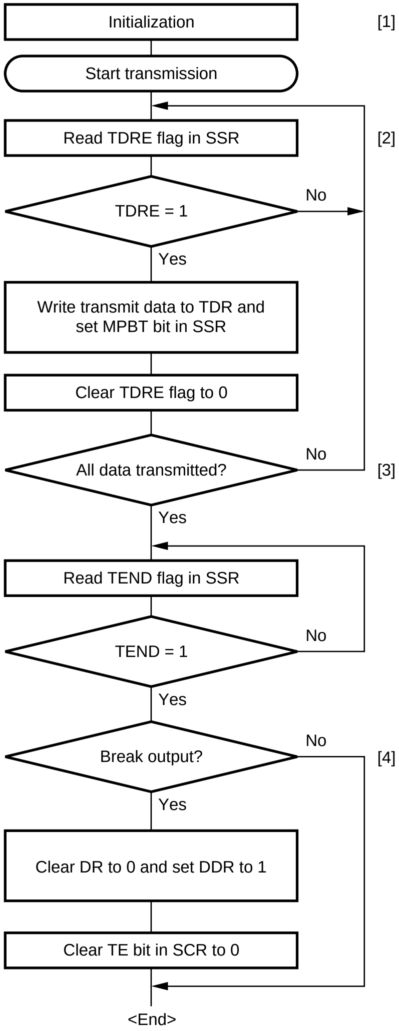


Figure 12.10 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)

12.5.1 Multiprocessor Serial Data Transmission

Figure 12.11 shows a sample flowchart for multiprocessor serial data transmission. For an ID transmission cycle, set the MPBT bit in SSR to 1 before transmission. For a data transmission cycle, clear the MPBT bit in SSR to 0 before transmission. All other SCI operations are the same as those in asynchronous mode.



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. Set the MPBT bit in SSR to 0 or 1. Finally, clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set port DDR to 1, clear DR to 0, and then clear the TE bit in SCR to 0.

Figure 12.11 Sample Multiprocessor Serial Transmission Flowchart

12.5.2 Multiprocessor Serial Data Reception

Figure 12.13 shows a sample flowchart for multiprocessor serial data reception. If the MPIE bit in SCR is set to 1, data is skipped until data with a 1 multiprocessor bit is sent. On receiving data with a 1 multiprocessor bit, the receive data is transferred to RDR. An RXI interrupt request is generated at this time. All other SCI operations are the same as in asynchronous mode. Figure 12.12 shows an example of SCI operation for multiprocessor format reception.

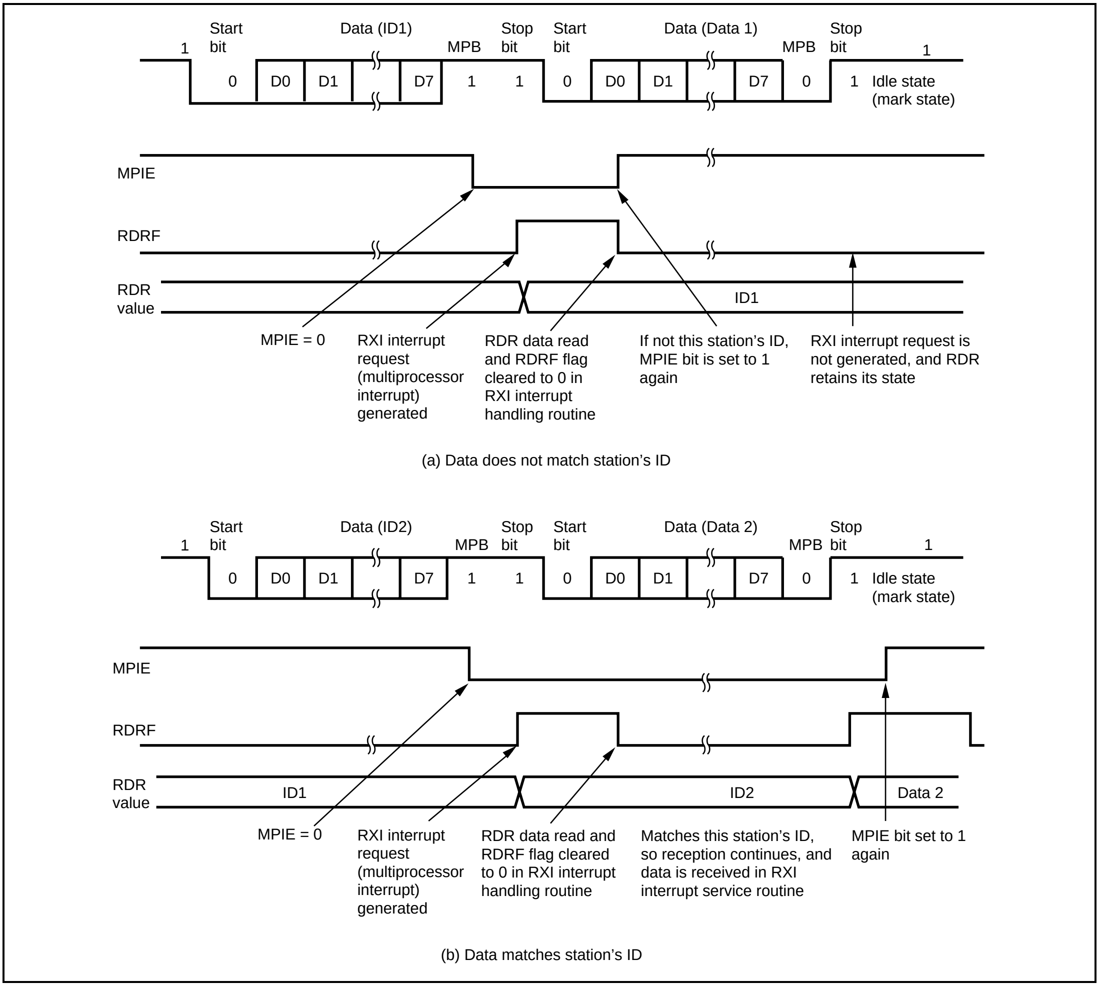
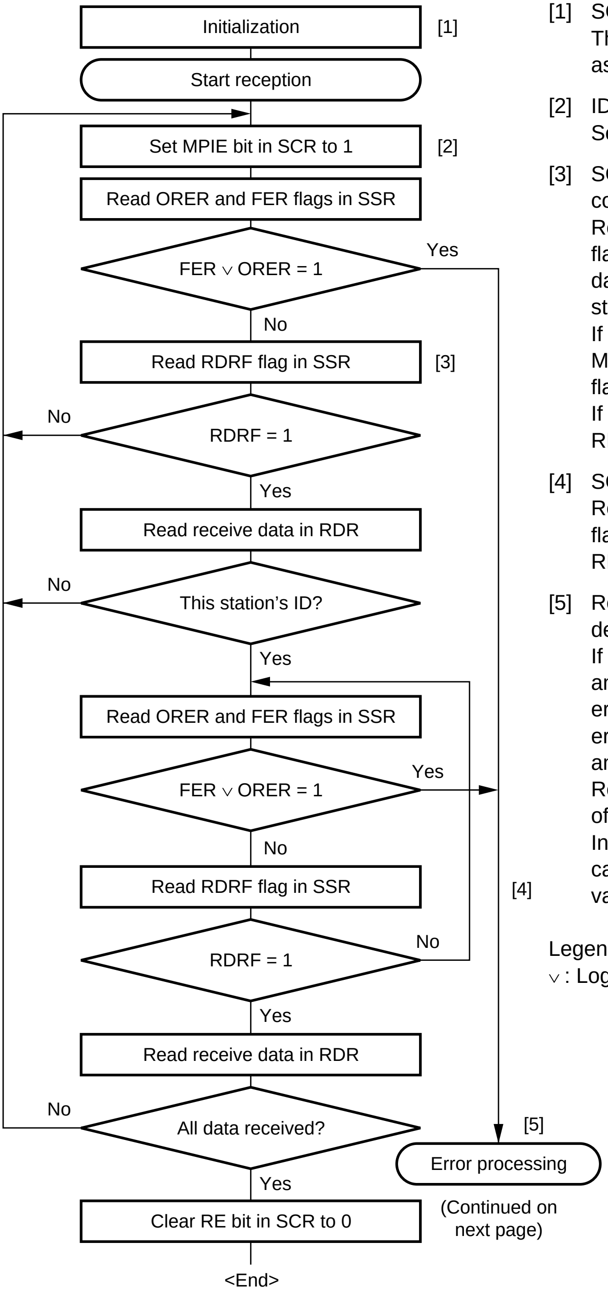

Start bit
Data (ID1)
MPB
Stop bit
Start bit
Data (Data 1)
MPB
Stop bit
1
0
D0
D1
D7
1
1
0
D0
D1
D7
0
1
Idle state (mark state)
1
MPIE
RDRF
RDR value
ID1
MPIE = 0
RXI interrupt request (multiprocessor interrupt) generated
RDR data read and RDRF flag cleared to 0 in RXI interrupt handling routine
If not this station’s ID, MPIE bit is set to 1 again
RXI interrupt request is not generated, and RDR retains its state
(a) Data does not match station’s ID
Start bit
Data (ID2)
MPB
Stop bit
Start bit
Data (Data 2)
MPB
Stop bit
1
0
D0
D1
D7
1
1
0
D0
D1
D7
0
1
Idle state (mark state)
1
MPIE
RDRF
RDR value
ID1
ID2
Data 2
MPIE = 0
RXI interrupt request (multiprocessor interrupt) generated
RDR data read and RDRF flag cleared to 0 in RXI interrupt handling routine
Matches this station’s ID, so reception continues, and data is received in RXI interrupt service routine
MPIE bit set to 1 again
(b) Data matches station’s ID

Figure 12.12 Example of SCI Receive Operation (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] ID reception cycle:
Set the MPIE bit in SCR to 1.

[3] SCI status check, ID reception and comparison:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID.
If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0.
If the data is this station's ID, clear the RDRF flag to 0.

[4] SCI status check and data reception:
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.

[5] Receive error processing and break detection:
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER and FER flags are all cleared to 0.
Reception cannot be resumed if either of these flags is set to 1.
In the case of a framing error, a break can be detected by reading the RxD pin value.

Legend:
∨ : Logical OR

Figure 12.13 Sample Multiprocessor Serial Reception Flowchart (1)

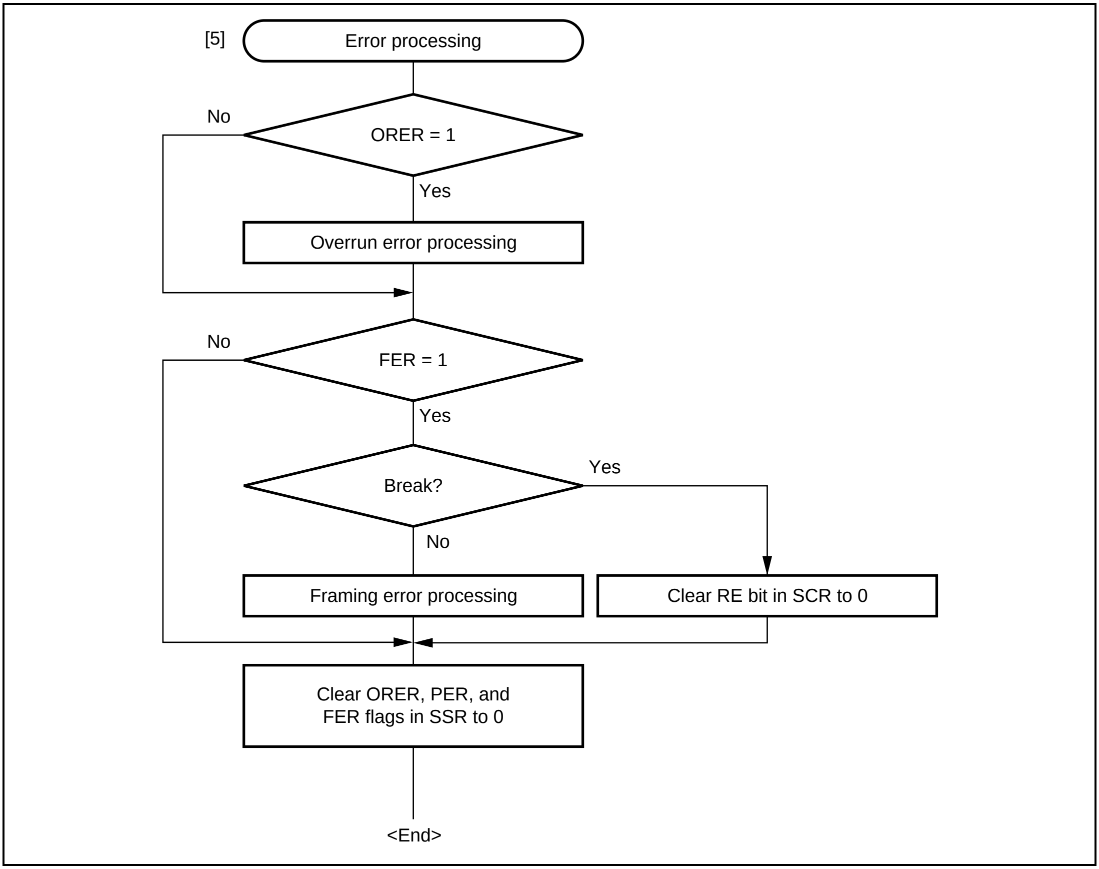
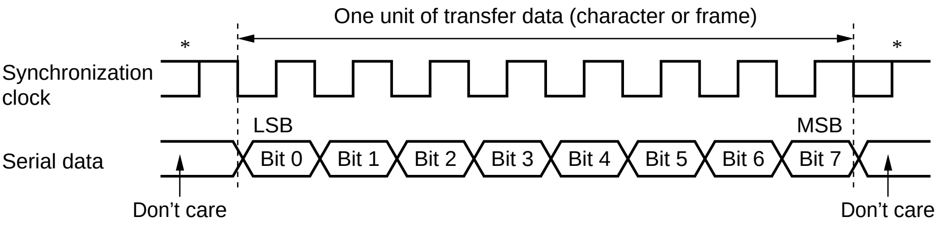


Figure 12.13 Sample Multiprocessor Serial Reception Flowchart (2)

12.6 Operation in Clocked Synchronous Mode

Figure 12.14 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses. One character in transfer data consists of 8-bit data. In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the MSB 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.



Note:* High except in continuous transfer/reception

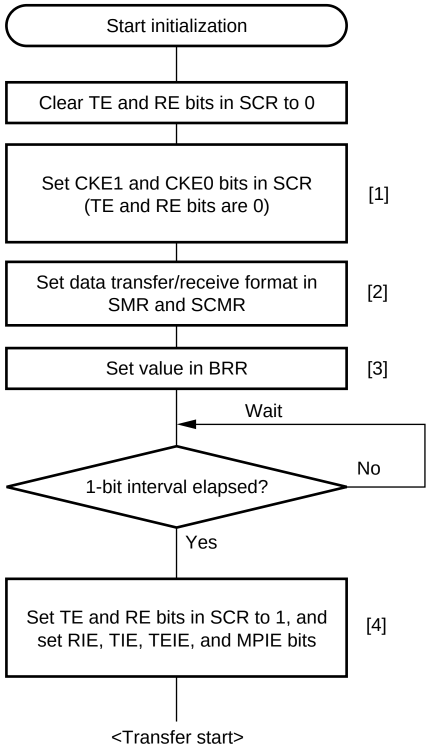
Figure 12.14 Data Format in Clocked Synchronous Communication (LSB-First)

12.6.1 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.

12.6.2 SCI Initialization (Clocked Synchronous Mode)

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 12.15. 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.


Start initialization
Clear TE and RE bits in SCR to 0
Set CKE1 and CKE0 bits in SCR (TE and RE bits are 0)
[1]
Set data transfer/receive format in SMR and SCMR
[2]
Set value in BRR
[3]
Wait
No
1-bit interval elapsed?
Yes
Set TE and RE bits in SCR to 1, and set RIE, TIE, TEIE, and MPIE bits
[4]
<Transfer start>

[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, TE and RE to 0.

[2] Set the data transfer/receive format in SMR and SCMR.

[3] Write a value corresponding to the bit rate to BRR. This step is not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1.
Also set the RIE, TIE TEIE, and MPIE bits.
Setting the TE and RE bits enables the TxD and RxD pins to be used.

Note:* In simultaneous transmit and receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

Figure 12.15 Sample SCI Initialization Flowchart

12.6.3 Serial Data Transmission (Clocked Synchronous Mode)

Figure 12.16 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 12.17 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.

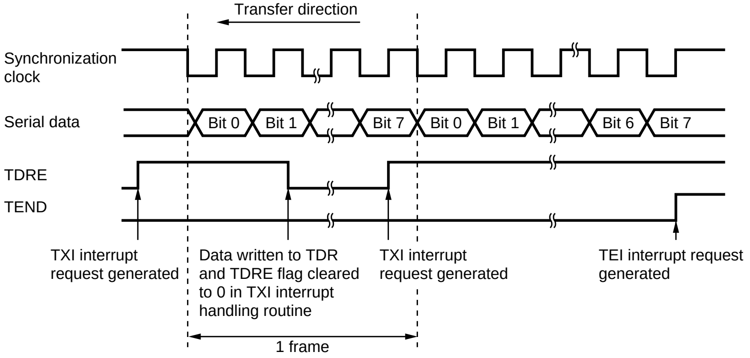
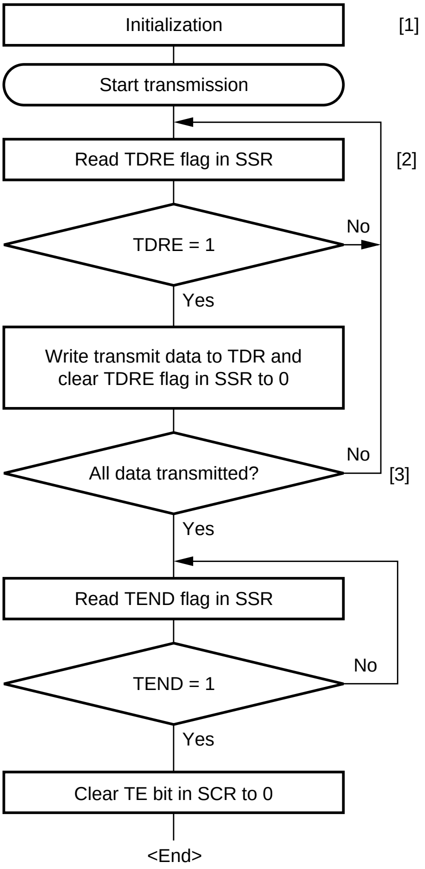

Transfer direction
Synchronization clock
Serial data
Bit 0
Bit 1
Bit 7
Bit 0
Bit 1
Bit 6
Bit 7
TDRE
TEND
TXI interrupt request generated
Data written to TDR and TDRE flag cleared to 0 in TXI interrupt handling routine
TXI interrupt request generated
TEI interrupt request generated
1 frame

Figure 12.16 Example of SCI Transmit Operation in Clocked Synchronous Mode



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0.

Figure 12.17 Sample Serial Transmission Flowchart

12.6.4 Serial Data Reception (Clocked Synchronous Mode)

Figure 12.18 shows an example of SCI operation for reception in clocked synchronous mode. In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with a synchronization clock input or output, starts receiving data, and stores the receive data in RSR.
2. If an overrun error (when reception of the next data is completed while the RDRF flag 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.

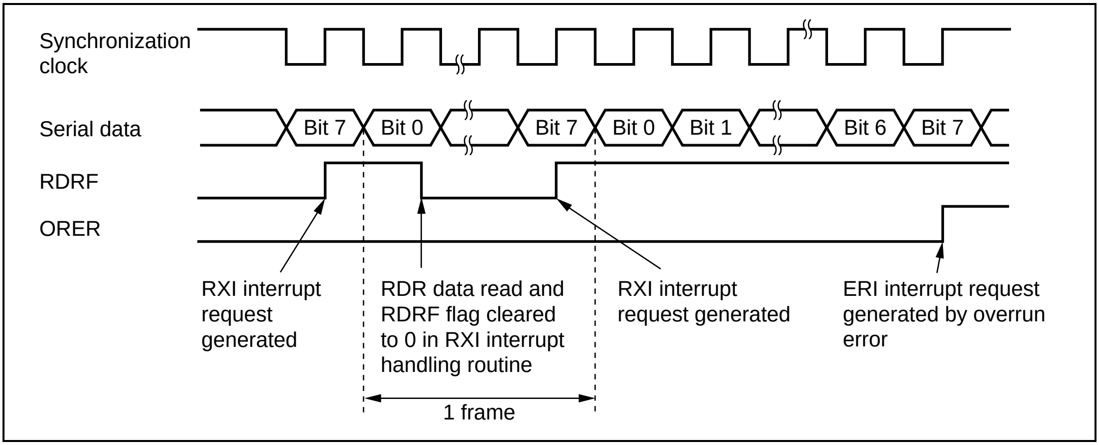
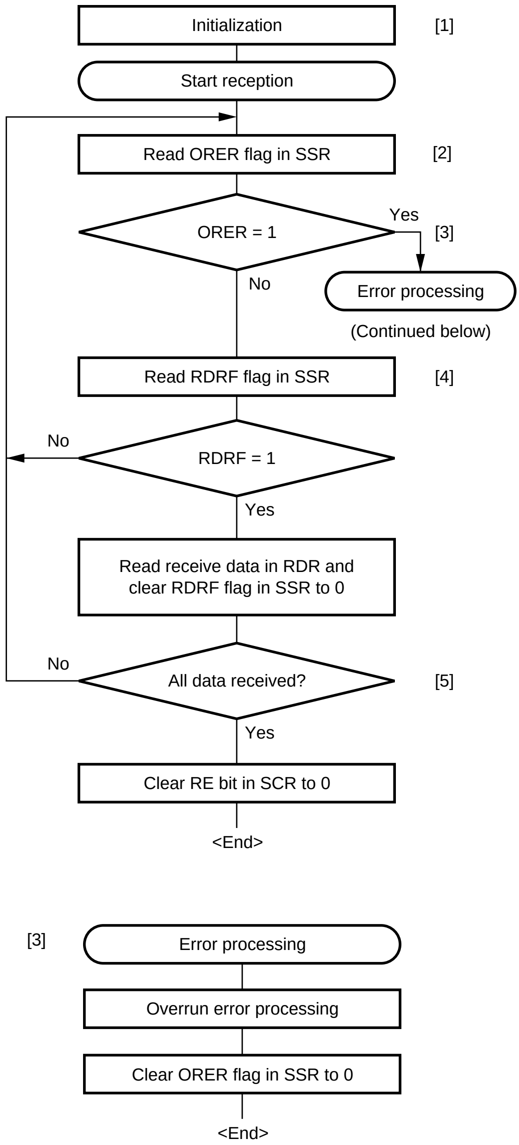

Synchronization clock
Serial data
Bit 7
Bit 0
Bit 7
Bit 0
Bit 1
Bit 6
Bit 7
RDRF
ORER
RXI interrupt request generated
RDR data read and RDRF flag cleared to 0 in RXI interrupt handling routine
RXI interrupt request generated
ERI interrupt request generated by overrun error
1 frame

Figure 12.18 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 12.19 shows a sample flowchart for serial data reception.



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Transfer cannot be resumed if the ORER flag is set to 1.

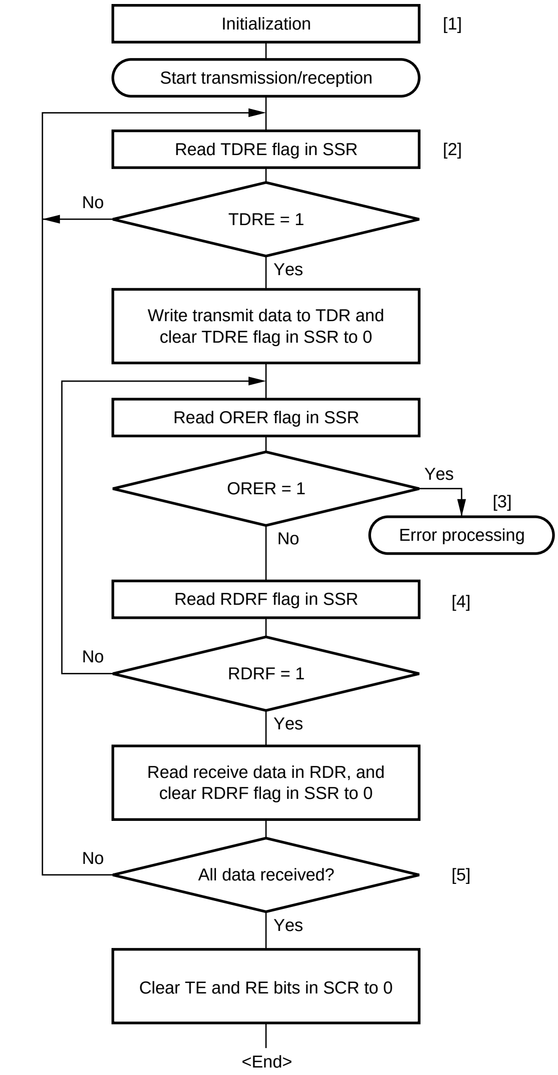
[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0.
Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the MSB (bit 7) of the current frame is received, reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0 should be finished.

Figure 12.19 Sample Serial Reception Flowchart

12.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode)

Figure 12.20 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, check 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, and then set the TE and RE bits to 1 simultaneously with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, check that the SCI has finished reception, and 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, set the TE and RE bits to 1 simultaneously with a single instruction.


Initialization
[1]
Start transmission/reception
Read TDRE flag in SSR
[2]
No
TDRE = 1
Yes
Write transmit data to TDR and clear TDRE flag in SSR to 0
Read ORER flag in SSR
ORER = 1
Yes
[3]
Error processing
No
Read RDRF flag in SSR
[4]
No
RDRF = 1
Yes
Read receive data in RDR, and clear RDRF flag in SSR to 0
No
All data received?
[5]
Yes
Clear TE and RE bits in SCR to 0
<End>

[1] SCI initialization:
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.
Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.

[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/ reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.

Note:* When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

Figure 12.20 Sample Flowchart of Simultaneous Serial Transmission and Reception

12.7 Interrupt Sources

Table 12.10 shows the interrupt sources in serial communication interface. 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.

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.

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.

Table 12.10 SCI Interrupt Sources

Channel	Name	Interrupt Source	Interrupt Flag	Priority
1	ERI1	Receive error	ORER, FER, PER	High
	RXI1	Receive data full	RDRF	↑
	TXI1	Transmit data empty	TDRE	
	TEI1	Transmit end	TEND	Low

12.8 Usage Notes

12.8.1 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, refer to section 19, Power-Down Modes.

12.8.2 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.

12.8.3 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 the 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.

12.8.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) is 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.

12.8.5 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.

12.8.6 SCI Operations during Mode Transitions

Transmission: Before making a 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 a transition is made during data transmission, the data being transmitted will be undefined.

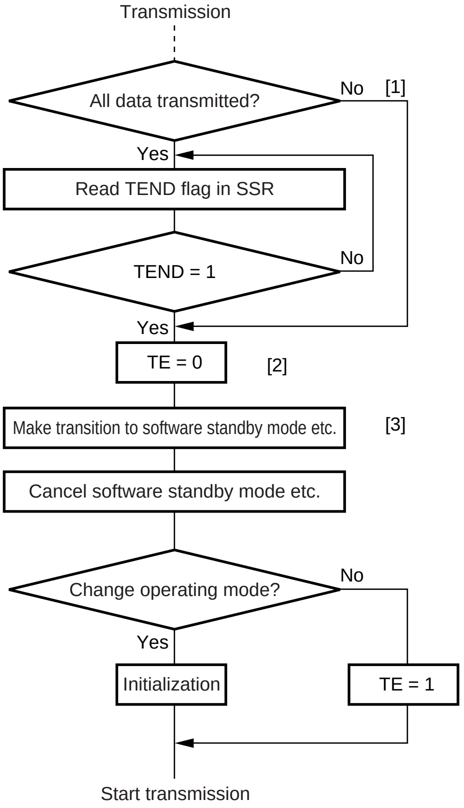
To transmit data in the same transmission mode after mode cancellation, set TE to 1, read SSR, write to TDR, clear TDRE in this order, and then start transmission. To transmit data in a different transmission mode, initialize the SCI first.

Figure 12.21 shows a sample flowchart for mode transition during transmission. Figures 12.22 and 12.23 show the pin states during transmission.

Reception: Before making a transition to module stop, software standby, watch, sub-active, or sub-sleep mode, stop reception (RE = 0). RSR, RDR, and SSR are reset. If a 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 12.24 shows a sample flowchart for mode transition during reception.



[1] Data being transmitted is lost halfway. Data can be normally transmitted from the CPU by setting TE to 1, reading SSR, writing to TDR, and clearing TDRE to 0 after mode cancellation.

[2] Also clear TIE and TEIE to 0 when they are 1.

[3] Module stop, watch, sub-active, and sub-sleep modes are included.

Figure 12.21 Sample Flowchart for Mode Transition during Transmission

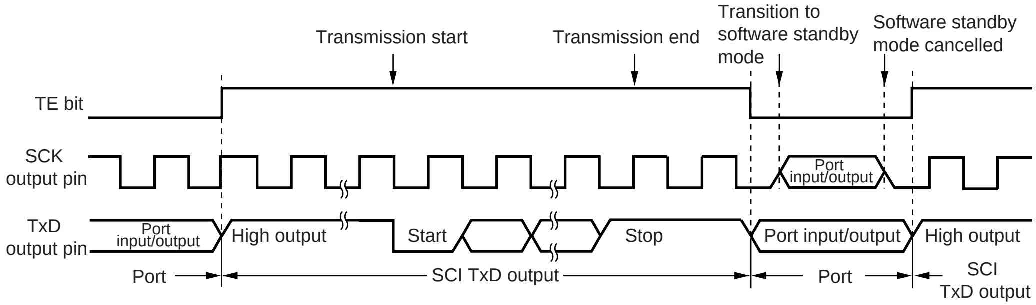


Figure 12.22 Pin States during Transmission in Asynchronous Mode (Internal Clock)

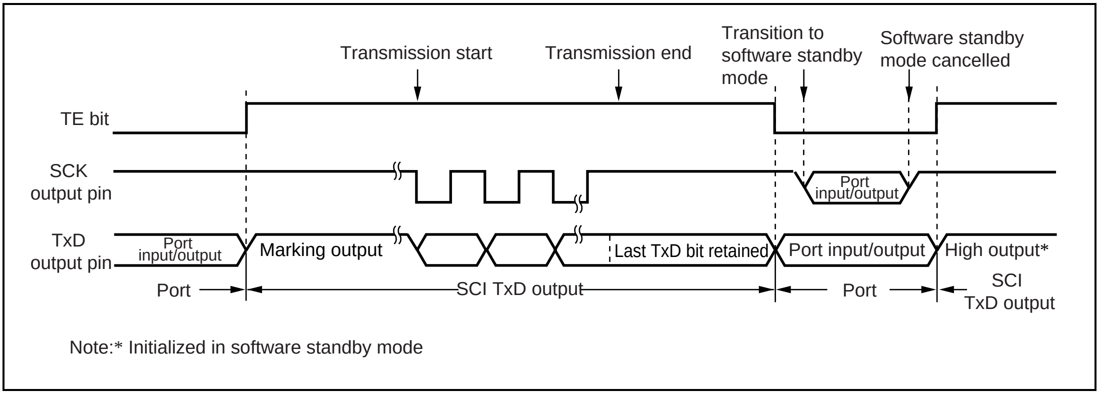
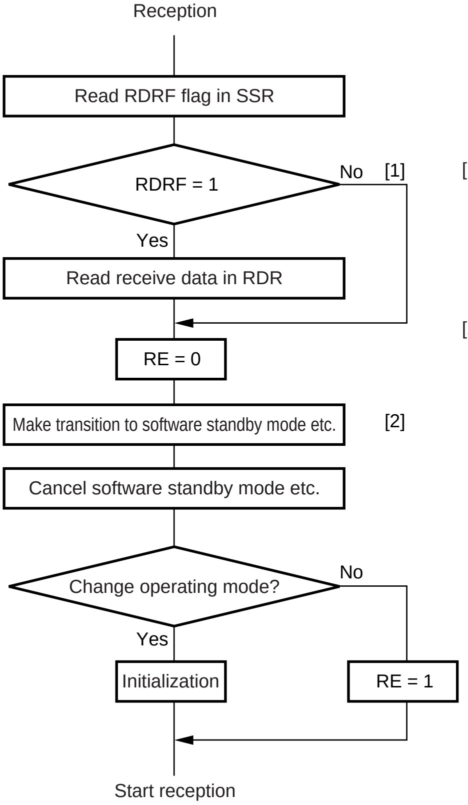


Figure 12.23 Pin States during Transmission in Clocked Synchronous Mode (Internal Clock)



[1] Data being received will be invalid.

[2] Module stop, watch, sub-active, and sub-sleep modes are included.

Figure 12.24 Sample Flowchart for Mode Transition during Reception

12.8.7 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 12.25.

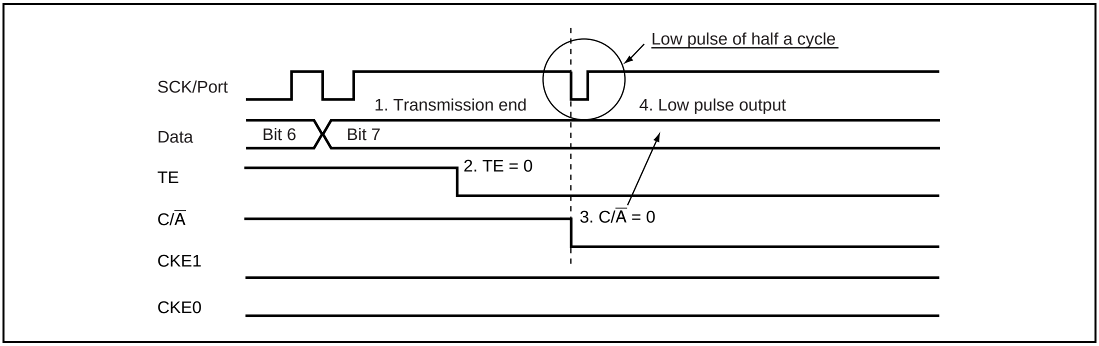


Figure 12.25 Switching from SCK Pins to Port Pins

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/$\overline{\text{A}}$ = 1, CKE1 = 0, CKE1 = 0, and TE = 1.

1. End serial data transmission
2. TE bit = 0
3. CKE1 bit = 1
4. C/$\overline{\text{A}}$ bit = 0 (switch to port output)
5. CKE1 bit = 0

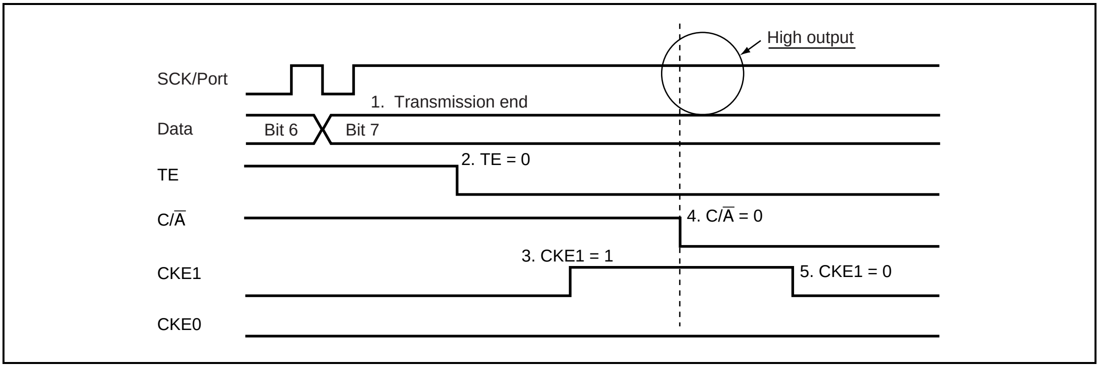


Figure 12.26 Prevention of Low Pulse Output at Switching from SCK Pins to Port Pins

Section 13 I^{2}C Bus Interface (IIC)

The I^{2}C bus interface is provided as an optional function. Note the following point when using this optional function.

- Although the product type name is identical, please contact Hitachi before using this optional function on an F-ZTAT version product.

This LSI has a two-channel I^{2}C bus interface. The I^{2}C bus interface conforms to and provides a subset of the Philips I^{2}C bus (inter-IC bus) interface functions. The register configuration that controls the I^{2}C bus differs partly from the Philips configuration, however.

13.1 Features

- Selection of addressing format or non-addressing format
  - I^{2}C bus format: addressing format with an acknowledge bit, for master/slave operation
  - Clocked synchronous serial format: non-addressing format without an acknowledge bit, for master operation only
- Conforms to Philips I^{2}C bus interface (I^{2}C bus format)
- Two ways of setting slave address (I^{2}C bus format)
- Start and stop conditions generated automatically in master mode (I^{2}C bus format)
- Selection of the acknowledge output level in reception (I^{2}C bus format)
- Automatic loading of an acknowledge bit in transmission (I^{2}C bus format)
- Wait function in master mode (I^{2}C bus format)
  - 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.
- Wait function (I^{2}C bus format)
  - A wait request can be generated by driving the SCL pin low after data transfer.
  - 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^{2}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^{2}C bus format (including address reception after loss of master arbitration)
  - Start condition detection (in master mode)
  - Stop condition detection (in slave mode)
- Selection of 16 internal clocks (in master mode)

- Direct bus drive (SCL/SDA pin)
  - Four pins—P52/SCL0, P97/SDA0, P86/SCL1, and P42/SDA1 —(normally NMOS push-pull outputs) function as NMOS open-drain outputs when the bus drive function is selected.

Figure 13.1 shows a block diagram of the I^{2}C bus interface. Figure 13.2 shows an example of I/O pin connections to external circuits. Since I^{2}C bus interface I/O pins are different in structure from normal port pins, they have different specifications for permissible applied voltages. For details, see section 21, Electrical Characteristics.

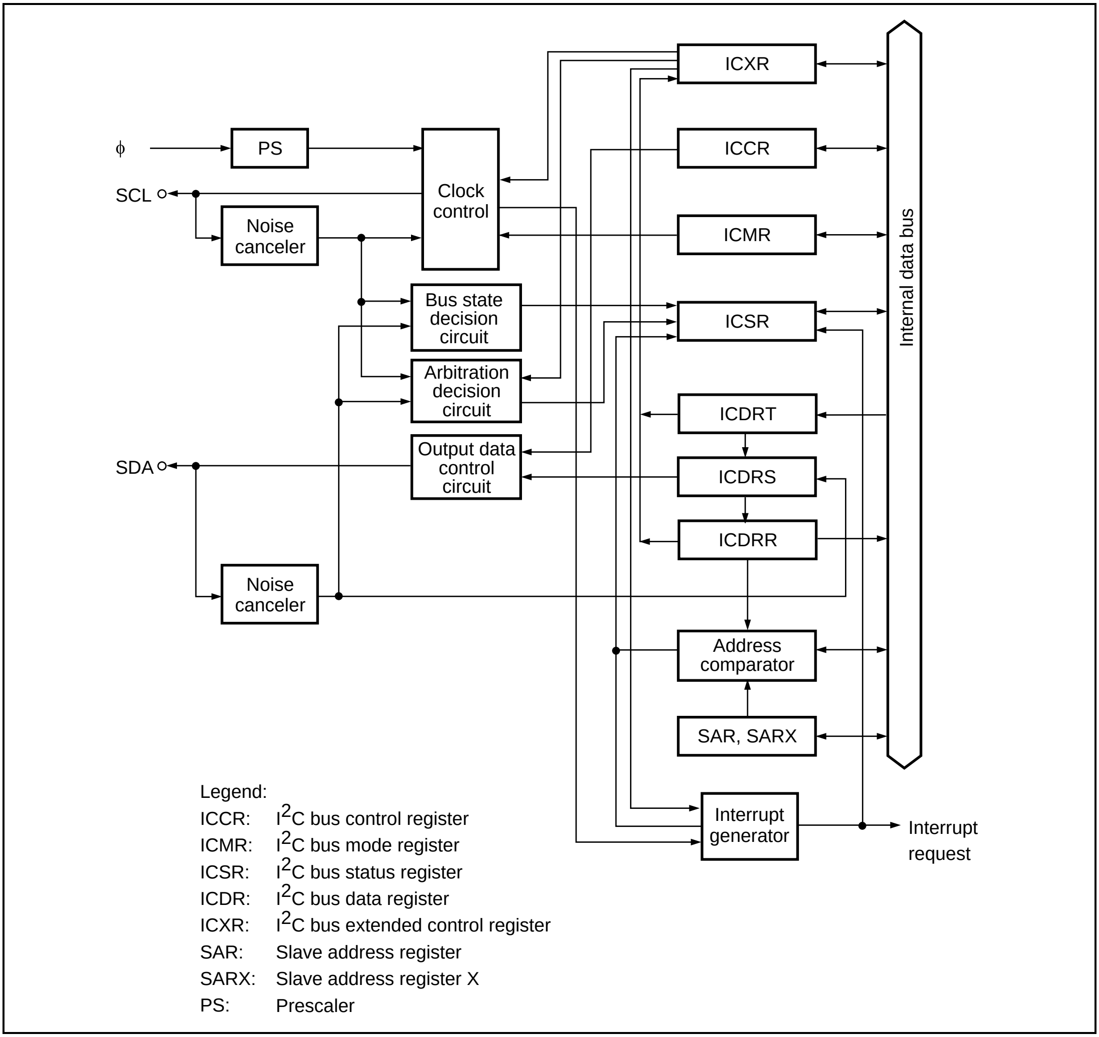

ICXR
ICCR
ICMR
ICSR
ICDRT
ICDRS
ICDRR
Address comparator
SAR, SARX
Internal data bus
φ
PS
SCL
Clock control
Noise canceler
Bus state decision circuit
Arbitration decision circuit
Output data control circuit
SDA
Noise canceler
Interrupt generator
Interrupt request
Legend:
ICCR: I2C bus control register
ICMR: I2C bus mode register
ICSR: I2C bus status register
ICDR: I2C bus data register
ICXR: I2C bus extended control register
SAR: Slave address register
SARX: Slave address register X
PS: Prescaler

Figure 13.1 Block Diagram of I^{2}C Bus Interface

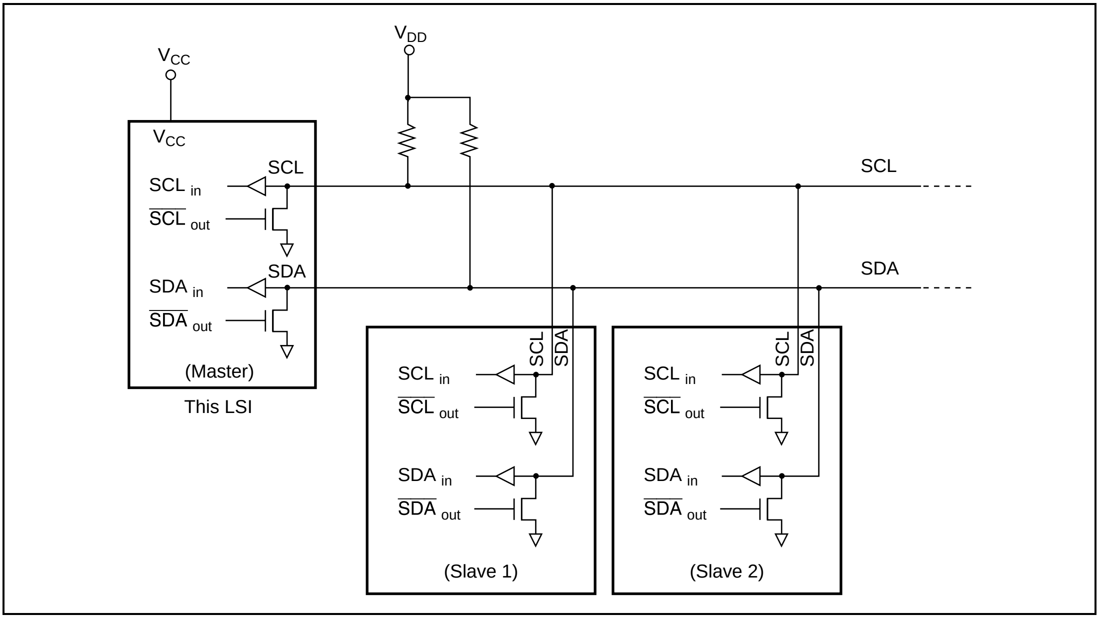


Figure 13.2 I^{2}C Bus Interface Connections (Example: This LSI as Master)

13.2 Input/Output Pins

Table 13.1 summarizes the input/output pins used by the I^{2}C bus interface.

Table 13.1 Pin Configuration

Channel	Symbol*	Input/Output	Function
0	SCL0	Input/Output	Serial clock input/output pin of IIC_0
	SDA0	Input/Output	Serial data input/output pin of IIC_0
1	SCL1	Input/Output	Serial clock input/output pin of IIC_1
	SDA1	Input/Output	Serial data input/output pin of IIC_1

Note:* In the text, the channel subscript is omitted, and only SCL and SDA are used.

13.3 Register Descriptions

The I^{2}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, refer to section 3.2.3, Serial Timer Control Register (STCR).

- I^{2}C bus control register (ICCR)
- I^{2}C bus status register (ICSR)
- I^{2}C bus data register (ICDR)
- I^{2}C bus mode register (ICMR)
- Slave address register (SAR)
- Second slave address register (SARX)
- I^{2}C bus extended control register (ICXR)
- DDC switch register (DDCSWR)

13.3.1 I^{2}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 internally divided into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). Data transfers among these 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.

In master transmit mode with the I^{2}C bus format, writing transmit data to ICDR should be performed after start condition detection. When the start condition is detected, previous write data is ignored. In slave transmit mode, writing should be performed after the slave addresses match and the TRS bit is automatically changed to 1.

If the IIC is in transmit mode (TRS = 1) and ICDRT has the next transmit data (the ICDRE flag is 0) after successful transmission/reception of one frame of data using ICDRS, data is transferred automatically from ICDRT to ICDRS.

If the IIC is in transmit mode (TRS = 1) and ICDRT has the next data (the ICDRE flag is 0), data is transferred automatically from ICDRT to ICDRS, following transmission of one frame of data using ICDRS. When the ICDRE flag is 1 and the next transmit data writing is waited, data is transferred automatically from ICDRT to ICDRS by writing to ICDR. If I^{2}C is in receive mode (TRS = 0), no data is transferred from ICDRT to ICDRS. Note that data should not be written to ICDR in receive mode.

Reading receive data from ICDR is performed after data is transferred from ICDRS to ICDRR.

If I^2C is in receive mode and no previous data remains in ICDRR (the ICDRF flag is 0), data is transferred automatically from ICDRS to ICDRR, following reception of one frame of data using ICDRS. If additional data is received while the ICDRF flag is 1, data is transferred automatically from ICDRS to ICDRR by reading from ICDR. In transmit mode, no data is transferred from ICDRS to ICDRR. Always set I^2C 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.

13.3.2 Slave Address Register (SAR)

SAR sets the slave address and selects the communication format. If the LSI is in slave mode with the I^2C bus format selected, when the FS bit is set to 0 and 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. Refer to table 13.2. This bit should be set to 0 when general call address recognition is performed.

13.3.3 Second Slave Address Register (SARX)

SARX sets the second slave address and selects the communication format. If the LSI is in slave mode with the I^{2}C bus format selected, when the FSX bit is set to 0 and 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. Refer to table 13.2.

Table 13.2 Communication Format

SAR	SARX	
FS	FSX	Operating Mode
0	0	I^{2}C bus format • SAR and SARX slave addresses recognized • General call address recognized
	1	I^{2}C bus format • SAR slave address recognized • SARX slave address ignored • General call address recognized
1	0	I^{2}C bus format • SAR slave address ignored • SARX slave address recognized • General call address ignored
	1	Clocked synchronous serial format • SAR and SARX slave addresses ignored • General call address ignored

- I^{2}C bus format: addressing format with an acknowledge bit
- Clocked synchronous serial format: non-addressing format without an acknowledge bit, for master mode only

13.3.4 I^{2}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^{2}C bus format is used.
6	WAIT	0	R/W	Wait Insertion Bit This bit is valid only in master mode with the I^{2}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 (8th clock), 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, refer to section 13.4.7, IRIC Setting Timing and SCL Control.
5	CKS2	0	R/W	Transfer Clock Select 2 to 0
4	CKS1	0	R/W	These bits are used only in master mode.
3	CKS0	0	R/W	These bits select the required transfer rate, together with the IICX1 (IIC_1) and IICX0 (IIC_0) bits in STCR. Refer to table 13.3.

Bit	Bit Name	Initial Value	R/W	Description
2	BC2	0	R/W	Bit Counter 2 to 0 These bits specify the number of bits to be transferred next. 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 000, the setting should be made while the SCL line is low. The bit counter is initialized to 000 when a start condition is detected. The value returns to 000 at the end of a data transfer. I^2C 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
1	BC1	0	R/W	
0	BC0	0	R/W	

Table 13.3 I^{2}C Transfer Rate

STCR	ICMR						
Bits 5 and 6	Bit 5	Bit 4	Bit 3		Transfer Rate		
IICX	CKS2	CKS1	CKS0	Clock	φ = 5 MHz	φ = 8 MHz	φ = 10 MHz
0	0	0	0	φ/28	179 kHz	286 kHz	357 kHz
0	0	0	1	φ/40	125 kHz	200 kHz	250 kHz
0	0	1	0	φ/48	104 kHz	167 kHz	208 kHz
0	0	1	1	φ/64	78.1 kHz	125 kHz	156 kHz
0	1	0	0	φ/80	62.5 kHz	100 kHz	125 kHz
0	1	0	1	φ/100	50.0 kHz	80.0 kHz	100 kHz
0	1	1	0	φ/112	44.6 kHz	71.4 kHz	89.3 kHz
0	1	1	1	φ/128	39.1 kHz	62.5 kHz	78.1 kHz
1	0	0	0	φ/56	89.3 kHz	143 kHz	179 kHz
1	0	0	1	φ/80	62.5 kHz	100 kHz	125 kHz
1	0	1	0	φ/96	52.1 kHz	83.3 kHz	104 kHz
1	0	1	1	φ/128	39.1 kHz	62.5 kHz	78.1 kHz
1	1	0	0	φ/160	31.3 kHz	50.0 kHz	62.5 kHz
1	1	0	1	φ/200	25.0 kHz	40.0 kHz	50.0 kHz
1	1	1	0	φ/224	22.3 kHz	35.7 kHz	44.6 kHz
1	1	1	1	φ/256	19.5 kHz	31.3 kHz	39.1 kHz

13.3.5 I^{2}C Bus Control Register (ICCR)

ICCR controls the I^{2}C bus interface and performs interrupt flag confirmation.

Bit	Bit Name	Initial Value	R/W	Description
7	ICE	0	R/W	I^{2}C Bus Interface Enable 0: I^{2}C bus interface modules are stopped and I^{2}C bus interface module internal state is initialized. SAR and SARX can be accessed. 1: I^{2}C bus interface modules can perform transfer operation, and the ports function as the SCL and SDA input/output pins. ICMR and ICDR can be accessed.
6	IEIC	0	R/W	I^{2}C Bus Interface Interrupt Enable 0: Disables interrupts from the I^{2}C bus interface to the CPU 1: Enables interrupts from the I^{2}C bus interface to the CPU.
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 contention in master mode with the I^{2}C bus format. In slave receive mode with I^{2}C bus format, the R/W bit in the first frame immediately after the start condition sets these bits in receive mode or transmit mode automatically by hardware. Modification of the TRS bit during transfer is deferred until transfer is completed, and the changeover is made after completion of the transfer.

Bit	Bit Name	Initial Value	R/W	Description
5	MST	0	R/W	[MST clearing conditions] 1. When 0 is written by software 2. When lost in bus contention in I^{2}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 contention in I^{2}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 as the $R/\overline{W}$ bit after the first frame address matching in I^{2}C bus format slave mode
4	TRS	0		
3	ACKE	0	R/W	Acknowledge Bit Decision and Selection 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 ACKB bit in ICSR, which is always 0. 1: If the received acknowledge bit is 1, continuous transfer is halted. 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.

Bit	Bit Name	Initial Value	R/W	Description
2	BBSY	0	R/W	Bus Busy
0	SCP	1	W	Start Condition/Stop Condition Prohibit In master mode: • Writing 0 in BBSY and 0 in SCP: A stop condition is issued • Writing 1 in BBSY and 0 in SCP: A start condition and a restart condition are issued In slave mode: • Writing to the BBSY flag is disabled. [BBSY setting condition] When the SDA level changes from high to low under the condition of SCL = high, assuming that the start condition has been issued. [BBSY clearing condition] When the SDA level changes from low to high under the condition of SCL = high, assuming that the stop condition has been issued. To issue a start/stop condition, use the MOV instruction. The I^2C 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^2C bus (SCL, SDA) is busy or free. The SCP bit is always read as 1. If 0 is written, the data is not stored.

Bit	Bit Name	Initianl Value	R/W	Description
1	IRIC	0	R/W	I^{2}C Bus Interface Interrupt Request Flag Indicates that the I^{2}C bus interface has issued an interrupt request to the CPU. IRIC is set at different times depending on the FS bit in SAR, the FSX bit in SARX, and the WAIT bit in ICMR. See section 13.4.7, IRIC Setting Timing and SCL Control. The conditions under which IRIC is set also differ depending on the setting of the ACKE bit in ICCR. [Setting conditions] I^{2}C bus format master mode: • 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) • When a wait is inserted between the data and acknowledge bit when the WAIT bit is 1 (fall of the 8th transmit/receive clock) • At the end of data transfer (rise of the 9th transmit/receive clock while no wait is inserted) • When a slave address is received after bus arbitration is lost (the first frame after the start condition) • If 1 is received as the acknowledge bit (when the ACKB bit in ICSR is set to 1) when the ACKE bit is 1 • When the AL flag is set to 1 after bus arbitration is lost while the ALIE bit is 1 I^{2}C bus format slave mode: • 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) • When the general call address is detected (when 0 is received as the R/$\overline{W}$ 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) • If 1 is received as the acknowledge bit (when the ACKB bit in ICSR is set to 1) while the ACKE bit is 1 • When a stop condition is detected (when the STOP or ESTP flag in ICSR is set to 1) while the STOPIM bit is 0

Bit	Bit Name	Initial Value	R/W	Description
1	IRIC	0	R/W	Clocked synchronous serial format mode: • At the end of data transfer (rise of the 8th transmit/receive) • When a start condition is detected When the ICDRE or ICDRF flag is set to 1 in any operating mode: • 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) • When data is transferred among the ICDR register and buffer (when data is transferred from ICDRT to ICDRS in transmit mode and the ICDRE flag is set to 1, or when data is transferred from ICDRS to ICDRR in receive mode and the ICDRF flag is set to 1) [Clearing conditions] • When 0 is written in IRIC after reading IRIC = 1

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

When, with the I^{2}C bus format selected, IRIC is set to 1 and an interrupt is generated, other flags must be checked in order to identify the source that set IRIC 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 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^{2}C bus format slave mode.

Tables 13.4 and 13.5 show the relationship between the flags and the transfer states.

Table 13.4 Flags and Transfer States (Master Mode)

MST	TRS	BBSY	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	ICDRF	ICDRE	State
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 (ACKE=1 and ACKB=1)
1	1	1	0	0	1↑	0	0	0	0	0	—	1↑	Transmission end with ICDRE=0
1	1	1	0	0	—	0	0	0	0	0	—	0↓	ICDR write with the above state
1	1	1	0	0	—	0	0	0	0	0	—	1	Transmission end with ICDRE=1
1	1	1	0	0	—	0	0	0	0	0	—	0↓	ICDR write with the above state or after start condition detected
1	1	1	0	0	1↑	0	0	0	0	0	—	1↑	Automatic data transfer from ICDRT to ICDRS with the above state
1	0	1	0	0	1↑	0	0	0	0	—	1↑	—	Reception end with ICDRF=0
1	0	1	0	0	—	0	0	0	0	—	0↓	—	ICDR read with the above state
1	0	1	0	0	—	0	0	0	0	—	1	—	Reception end with ICDRF=1
1	0	1	0	0	—	0	0	0	0	—	0↓	—	ICDR read with the above state
1	0	1	0	0	1↑	0	0	0	0	—	1↑	—	Automatic data transfer from ICDRS to ICDRR with the above state

Table 13.4 Flags and Transfer States (Master Mode) (cont)

MST	TRS	BBSY	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	ICDRF	ICDRE	State
0↓	0↓	1	0	0	—	0	1↑	0	0	—	—	—	Arbitration lost
1	—	0↓	0	0	—	0	0	0	0	—	—	0↓	Stop condition detected

Legend

0:	0-state retained
1:	1-state retained
—:	Previous state retained
0↓:	Cleared to 0
1↑:	Set to 1

Table 13.5 Flags and Transfer States (Slave Mode)

MST	TRS	BBSY	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	ICDRF	ICDRE	State
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 in first frame (SARX≠SAR)
0	0	1	0	0	0	0	—	1↑	1↑	0	1↑	1	General call address match in first frame (SARX≠H'00)
0	1↑/0 *1	1	0	0	1↑	1↑	—	0	0	0	1↑	1	SARS match in first frame (SAR≠SARX)
0	1	1	0	0	—	—	—	—	0	1↑	—	—	Transmission end (ACKE=1 and ACKB=1)
0	1	1	0	0	1↑/0 *1	—	—	—	0	0	—	1↑	Transmission end with ICDRE=0
0	1	1	0	0	—	—	0↓	0↓	0	0	—	0↓	ICDR write with the above state
0	1	1	0	0	—	—	—	—	1	0		1	Transmission end with ICDRE=1
0	1	1	0	0	—	—	0↓	0↓	0	0		0↓	ICDR write with the above state
0	1	1	0	0	1↑/0 *2	—	0	0	0	0		1↑	Automatic data transfer from ICDRT to ICDRS with the above state
0	0	1	0	0	1↑/0 *2	—	—	—	—	—	1↑	—	Reception end with ICDRF=0
0	0	1	0	0	—	—	0↓	0↓	0↓	—	0↓	—	ICDR read with the above state

Table 13.5 Flags and Transfer States (Slave Mode) (cont)

MST	TRS	BBSY	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	ICDRF	ICDRE	State
0	0	1	0	0	—	—	—	—	—	—	1	—	Reception end with ICDRF=1
0	0	1	0	0	—	—	0↓	0↓	0↓	—	0↓	—	ICDR read with the above state
0	0	1	0	0	1↑/0 *[2]	—	0	0	0	—	1↑	—	Automatic data transfer from ICDRS to ICDRR with the above state
0	—	0↓	1↑/0 *[3]	0/1↑ *[3]	—	—	—	—	—	—	—	0↓	Stop condition detected

Legend

0: 0-state retained

1: 1-state retained

—: Previous state retained

0↓: Cleared to 0

1↑: Set to 1

Notes: 1. Set to 1 when 1 is received as a R/$\overline{\text{W}}$ 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.

13.3.6 I^{2}C Bus Status Register (ICSR)

ICSR consists of status flags. Also see tables 13.4 and 13.5.

Bit	Bit Name	Initial Value	R/W	Description
7	ESTP	0	R/(W)*	Error Stop Condition Detection Flag This bit is valid in I^{2}C bus format slave mode. [Setting condition] When a stop condition is detected during frame transfer. [Clearing conditions] • When 0 is written in ESTP after reading ESTP = 1 • When the IRIC flag in ICCR is cleared to 0
6	STOP	0	R/(W)*	Normal Stop Condition Detection Flag This bit is valid in I^{2}C bus format slave mode. [Setting condition] When a stop condition is detected after frame transfer completion. [Clearing conditions] • When 0 is written in STOP after reading STOP = 1 • When the IRIC flag is cleared to 0
5	IRTR	0	R/(W)*	I^{2}C Bus Interface Continuous Transfer Interrupt Request Flag Indicates that the I^{2}C bus interface has issued an interrupt request to the CPU, and the source is completion of reception/transmission of one frame in continuous transmission/reception. When the IRTR flag is set to 1, the IRIC flag is also set to 1 at the same time. [Setting conditions] I^{2}C bus format slave mode: • When the ICDRE or ICDRF flag in ICDR is set to 1 when AASX = 1 Master mode or clocked synchronous serial format mode with I^{2}C bus format: • When the ICDRE or ICDRF flag is set to 1 [Clearing conditions] • When 0 is written after reading IRTR = 1 • When the IRIC flag is cleared to 0 while ICE is 1

Bit	Bit Name	Initial Value	R/W	Description
4	AASX	0	R/(W)*	Second Slave Address Recognition Flag In I^{2}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. [Setting condition] When the second slave address is detected in slave receive mode and FSX = 0 in SARX [Clearing conditions] • When 0 is written in AASX after reading AASX = 1 • When a start condition is detected • In master mode
3	AL	0	R/(W)*	Arbitration Lost Flag Indicates that arbitration was lost in master mode. [Setting conditions] When ALSL=0 • If the internal SDA and SDA pin disagree at the rise of SCL in master transmit mode • If the internal SCL line is high at the fall of SCL in master transmit mode When ALSL=1 • If the internal SDA and SDA pin disagree at the rise of SCL in master transmit mode • If the SDA pin is driven low by another device before the I^{2}C bus interface drives the SDA pin low, after the start condition instruction was executed in master transmit mode [Clearing conditions] • When ICDR is written to (transmit mode) or read from (receive mode) • When 0 is written in AL after reading AL = 1

Bit	Bit Name	Initial Value	R/W	Description
2	AAS	0	R/(W)*	Slave Address Recognition Flag In I^2C 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. [Setting condition] When the slave address or general call address (one frame including a $R/\overline{W}$ bit is H’00) is detected in slave receive mode and FS = 0 in SAR [Clearing conditions] • When ICDR is written to (transmit mode) or read from (receive mode) • When 0 is written in AAS after reading AAS = 1 • In master mode
1	ADZ	0	R/(W)*	General Call Address Recognition Flag In I^2C 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). [Setting condition] When the general call address (one frame including a $R/\overline{W}$ bit is H’00) is detected in slave receive mode and FS = 0 or FSX = 0 [Clearing conditions] • When ICDR is written to (transmit mode) or read from (receive mode) • When 0 is written in ADZ after reading ADZ = 1 • In master mode 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).

Bit	Bit Name	Initial Value	R/W	Description
0	ACKB	0	R/W	Acknowledge Bit Stores acknowledge data. Transmit mode: [Setting condition] When 1 is received as the acknowledge bit when ACKE=1 in transmit mode [Clearing conditions] • When 0 is received as the acknowledge bit when ACKE=1 in transmit mode • When 0 is written to the ACKE bit Receive mode: 0: Returns 0 as acknowledge data after data reception 1: Returns 1 as acknowledge data after data reception 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. When this bit is written, acknowledge data that is returned after receiving is rewritten regardless of the TRS value. If the ICSR register bit 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. Write the ACKE bit to 0 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.

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

13.3.7 DDC Switch Register (DDCSWR)

DDCSWR controls IIC internal latch clearance.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	R/W	Reserved The initial value should not be changed.
4	—	0	R	Reserved
3	CLR3	1	W*	IIC Clear 3 to 0
2	CLR2	1	W*	Controls initialization of the internal state of IIC_0 and IIC_1.
1	CLR1	1	W*	00--: Setting prohibited
0	CLR0	1	W*	0100: Setting prohibited 0101: IIC_0 internal latch cleared 0110: IIC_1 internal latch cleared 0111: IIC_0 and IIC_1 internal latches cleared 1---: Invalid setting When a write operation is performed on these bits, a clear signal is generated for the internal latch circuit of the corresponding module, and the internal state of the IIC module is initialized. These bits can only be written to; they are always read as 1. Write data to this bit 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. When clearing is required again, all the bits must be written to in accordance with the setting.

Note:* This bit is always read as 1.

13.3.8 I^{2}C Bus Extended Control Register (ICXR)

ICXR enables or disables the I^{2}C bus interface interrupt generation and continuous receive operation, and indicates the status of receive/transmit operations.

Bit	Bit Name	Initial Value	R/W	Description
7	STOPIM	0	R/W	Stop Condition Interrupt Source Mask Enables or disables the interrupt generation when the stop condition is detected in slave mode. 0: Enables IRIC flag setting and interrupt generation when the stop condition is detected (STOP = 1 or ESTP = 1) in slave mode. 1: Disables IRIC flag setting and interrupt generation when the stop condition is detected.
6	HNDS	0	R/W	Handshake Receive Operation Select Enables or disables continuous receive operation in receive mode. 0: Enables continuous receive operation 1: Disables continuous receive operation When the HNDS bit is cleared to 0, receive operation is performed continuously after data has been received successfully while ICDRF flag is 0. When the HNDS bit is set to 1, SCL is fixed to the low level and the next data transfer is disabled after data has been received successfully while the ICDRF flag is 0. The bus line is released and next receive operation is enabled by reading the receive data in ICDR.

Bit	Bit Name	Initial Value	R/W	Description
5	ICDRF	0	R	Receive Data Read Request Flag Indicates the ICDR (ICDRR) status in receive mode. 0: Indicates that the data has been already read from ICDR (ICDRR) or ICDR is initialized. 1: Indicates that data has been received successfully and transferred from ICDRS to ICDRR, and the data is ready to be read out. [Setting conditions] • When data is received successfully and transferred from ICDRS to ICDRR. (1) When data is received successfully while ICDRF = 0 (at the rise of the 9th clock pulse). (2) When ICDR is read successfully in receive mode after data was received while ICDRF = 1. [Clearing conditions] • When ICDR (ICDRR) is read. • When 0 is written to the ICE bit. • When the IIC is internally initialized using the CLR3 to CLR0 bits in DDCSWR. When ICDRF is set due to the condition (2) above, ICDRF is temporarily cleared to 0 when ICDR (ICDRR) is read; however, since data is transferred from ICDRS to ICDRR immediately, ICDRF is set to 1 again. Note that ICDR cannot be read successfully in transmit mode (TRS = 1) because data is not transferred from ICDRS to ICDRR. Be sure to read data from ICDR in receive mode (TRS = 0).

Bit	Bit Name	Initial Value	R/W	Description
4	ICDRE	0	R	Transmit Data Write Request Flag Indicates the ICDR (ICDRT) status in transmit mode. 0: Indicates that the data has been already written to ICDR (ICDRT) or ICDR is initialized. 1: Indicates that data has been transferred from ICDRT to ICDRS and is being transmitted, or the start condition has been detected or transmission has been complete, thus allowing the next data to be written to. [Setting conditions] • When the start condition is detected from the bus line state with I^{2}C bus format or serial format. • When data is transferred from ICDRT to ICDRS. 1. When data transmission completed while ICDRE = 0 (at the rise of the 9th clock pulse). 2. When data is written to ICDR in transmit mode after data transmission was completed while ICDRE = 1. [Clearing conditions] • When data is written to ICDR (ICDRT). • When the stop condition is detected with I^{2}C bus format or serial format. • When 0 is written to the ICE bit. • When the IIC is internally initialized using the CLR3 to CLR0 bits in DDCSWR. Note that if the ACKE bit is set to 1 with I^{2}C bus format thus enabling acknowledge bit decision, ICDRE is not set when data transmission is completed while the acknowledge bit is 1. When ICDRE is set due to the condition (2) above, ICDRE is temporarily cleared to 0 when data is written to ICDR (ICDRT); however, since data is transferred from ICDRT to ICDRS immediately, ICDRE is set to 1 again. Do not write data to ICDR when TRS = 0 because the ICDRE flag value is invalid during the time.

Bit	Bit Name	Initial Value	R/W	Description
3	ALIE	0	R/W	Arbitration Lost Interrupt Enable Enables or disables IRIC flag setting and interrupt generation when arbitration is lost. 0: Disables interrupt request when arbitration is lost. 1: Enables interrupt request when arbitration is lost.
2	ALSL	0	R/W	Arbitration Lost Condition Select Selects the condition under which arbitration is lost. 0: When the SDA pin state disagrees with the data that IIC bus interface outputs at the rise of SCL, or when the SCL pin is driven low by another device. 1: When the SDA pin state disagrees with the data that IIC bus interface outputs at the rise of SCL, or when the SDA line is driven low by another device in idle state or after the start condition instruction was executed.
1	FNC1	0	R/W	Function Bit
0	FNC0	0	R/W	Cancels some restrictions on usage. For details, refer to section 13.6, Usage Notes. 00: Restrictions on operation remaining in effect 01: Setting prohibited 10: Setting prohibited 11: Restrictions on operation canceled

13.4 Operation

The I^2C bus interface has an I^2C bus format and a serial format.

13.4.1 I^2C Bus Data Format

The I^2C bus format is an addressing format with an acknowledge bit. This is shown in figure 13.3. The first frame following a start condition always consists of 9 bits.

The serial format is a non-addressing format with no acknowledge bit. This is shown in figure 13.4.

Figure 13.5 shows the I^2C bus timing.

The symbols used in figures 13.3 to 13.5 are explained in table 13.6.

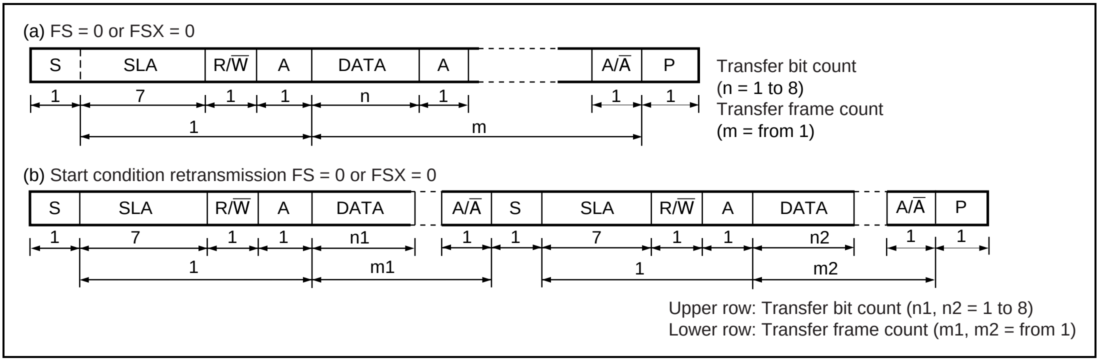


Figure 13.3 I^2C Bus Data Format (I^2C Bus Format)

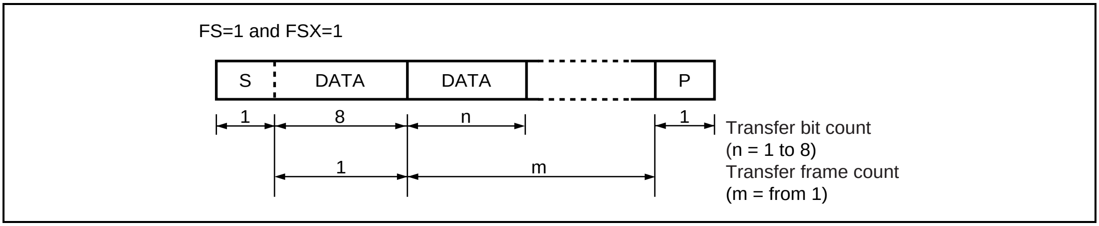


Figure 13.4 I^2C Bus Data Format (Serial Format)

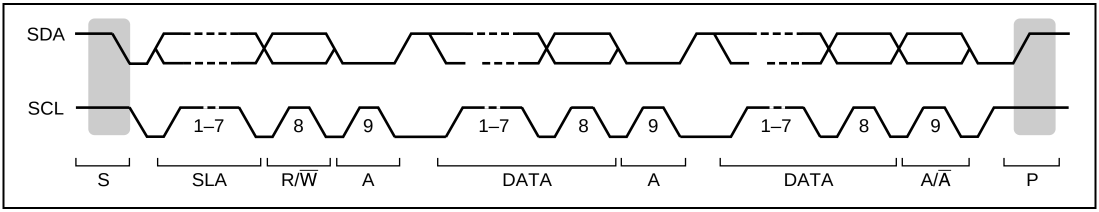


Figure 13.5 I^{2}C Bus Timing

Table 13.6 I^{2}C Bus Data Format Symbols

Legend	
S	Start condition. The master device drives SDA from high to low while SCL is high
SLA	Slave address. The master device selects the slave device.
R/$\overline{W}$	Indicates the direction of data transfer: from the slave device to the master device when R/$\overline{W}$ is 1, or from the master device to the slave device when R/$\overline{W}$ is 0
A	Acknowledge. The receiving device drives SDA low to acknowledge a transfer. (The slave device returns acknowledge in master transmit mode, and the master device returns acknowledge in master receive mode.)
DATA	Transferred data. The bit length of transferred data is set with the BC2 to BC0 bits in ICMR. The MSB first or LSB first is switched with the MLS bit in ICMR.
P	Stop condition. The master device drives SDA from low to high while SCL is high

13.4.2 Initialization

Initialize the IIC by the procedure shown in figure 13.6 before starting transmission/reception of data.

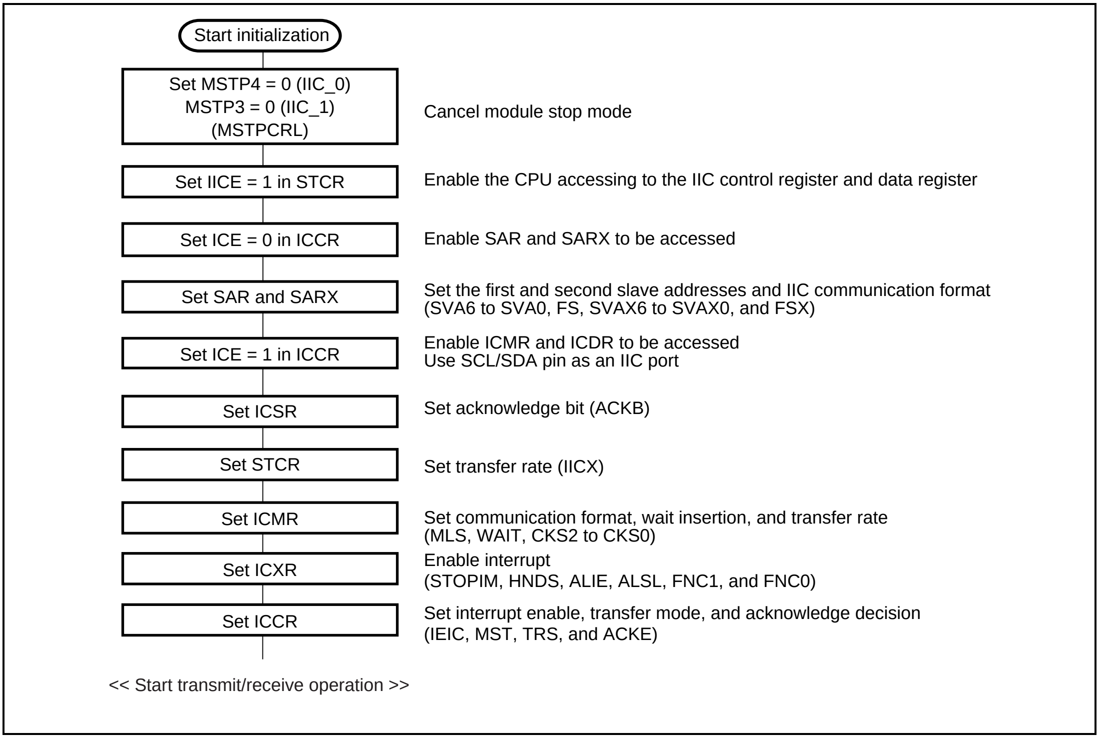

Start initialization
Set MSTP4 = 0 (IIC_0)
MSTP3 = 0 (IIC_1)
(MSTPCRL)
Cancel module stop mode
Set IICE = 1 in STCR
Enable the CPU accessing to the IIC control register and data register
Set ICE = 0 in ICCR
Enable SAR and SARX to be accessed
Set SAR and SARX
Set the first and second slave addresses and IIC communication format
(SVA6 to SVA0, FS, SVAX6 to SVAX0, and FSX)
Set ICE = 1 in ICCR
Enable ICMR and ICDR to be accessed
Use SCL/SDA pin as an IIC port
Set ICSR
Set acknowledge bit (ACKB)
Set STCR
Set transfer rate (IICX)
Set ICMR
Set communication format, wait insertion, and transfer rate
(MLS, WAIT, CKS2 to CKS0)
Set ICXR
Enable interrupt
(STOPIM, HNDS, ALIE, ALSL, FNC1, and FNC0)
Set ICCR
Set interrupt enable, transfer mode, and acknowledge decision
(IEIC, MST, TRS, and ACKE)
<< Start transmit/receive operation >>

Figure 13.6 Sample Flowchart for IIC Initialization

Note: Be sure to modify the ICMR register after transmit/receive operation has been completed. If the ICMR register is modified during transmit/receive operation, bit counter BC2 to BC0 will be modified erroneously, thus causing incorrect operation.

13.4.3 Master Transmit Operation

In I^{2}C bus format master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal.

Figure 13.7 shows the sample flowchart for the operations in master transmit mode.

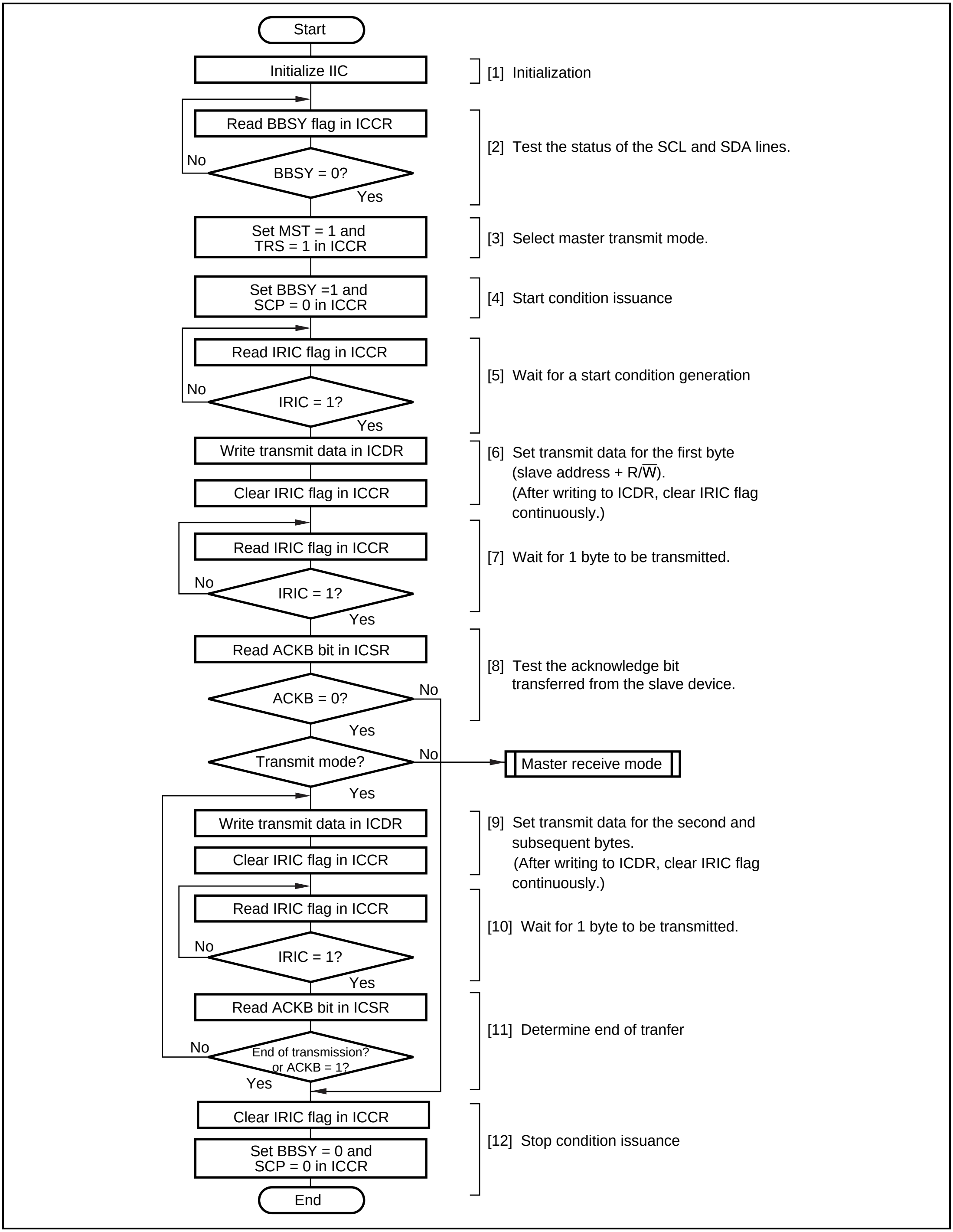

Start
Initialize IIC
[1] Initialization
Read BBSY flag in ICCR
BBSY = 0?
No
Yes
[2] Test the status of the SCL and SDA lines.
Set MST = 1 and TRS = 1 in ICCR
[3] Select master transmit mode.
Set BBSY =1 and SCP = 0 in ICCR
[4] Start condition issuance
Read IRIC flag in ICCR
IRIC = 1?
No
Yes
[5] Wait for a start condition generation
Write transmit data in ICDR
Clear IRIC flag in ICCR
[6] Set transmit data for the first byte (slave address + R/W). (After writing to ICDR, clear IRIC flag continuously.)
Read IRIC flag in ICCR
IRIC = 1?
No
Yes
[7] Wait for 1 byte to be transmitted.
Read ACKB bit in ICSR
ACKB = 0?
No
Yes
[8] Test the acknowledge bit transferred from the slave device.
Transmit mode?
No
Yes
Master receive mode
Write transmit data in ICDR
Clear IRIC flag in ICCR
[9] Set transmit data for the second and subsequent bytes. (After writing to ICDR, clear IRIC flag continuously.)
Read IRIC flag in ICCR
IRIC = 1?
No
Yes
[10] Wait for 1 byte to be transmitted.
Read ACKB bit in ICSR
End of transmission? or ACKB = 1?
No
Yes
[11] Determine end of tranfer
Clear IRIC flag in ICCR
Set BBSY = 0 and SCP = 0 in ICCR
[12] Stop condition issuance
End

Figure 13.7 Sample Flowchart for Operations in Master Transmit Mode

The transmission procedure and operations by which data is sequentially transmitted in synchronization with ICDR (ICDRT) write operations, are described below.

1. Initialize the IIC as described in section 13.4.2, Initialization.
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^{2}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 ($R/\overline{W}$).

   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 transmission clock and the data written to ICDR. 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 ACKE in ICCR, to clear received ACKB contents 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.

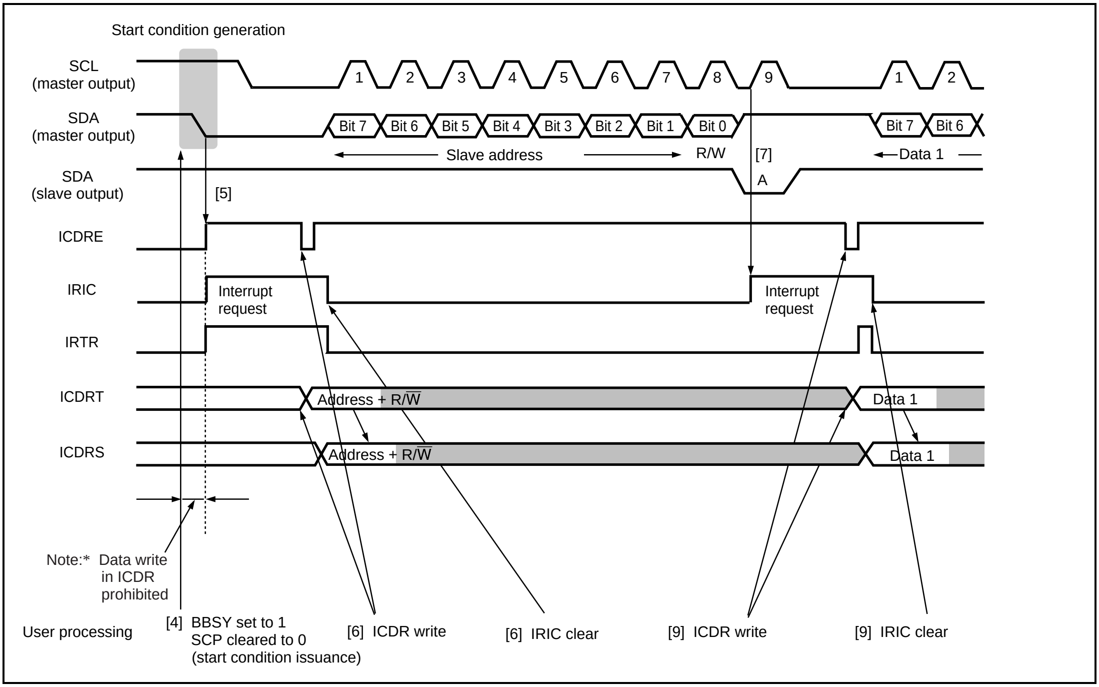


Figure 13.8 Example of Operation Timing in Master Transmit Mode (MLS = WAIT = 0)

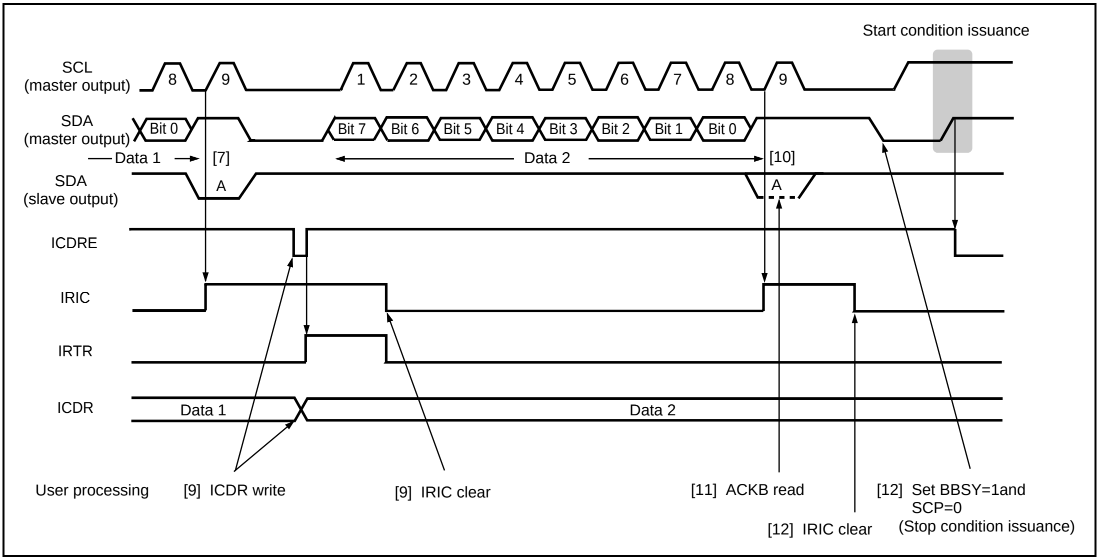


Figure 13.9 Example of Stop Condition Issuance Operation Timing in Master Transmit Mode (MLS = WAIT = 0)

13.4.4 Master Receive Operation

In I^2C 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 master device transmits data containing the slave address and $R/\overline{W}$ (1: read) in the first frame following the start condition issuance in master transmit mode, selects the slave device, and then switches the mode for receive operation.

Receive Operation Using the HNDS Function (HNDS = 1):

Figure 13.10 shows the sample flowchart for the operations in master receive mode (HNDS = 1).

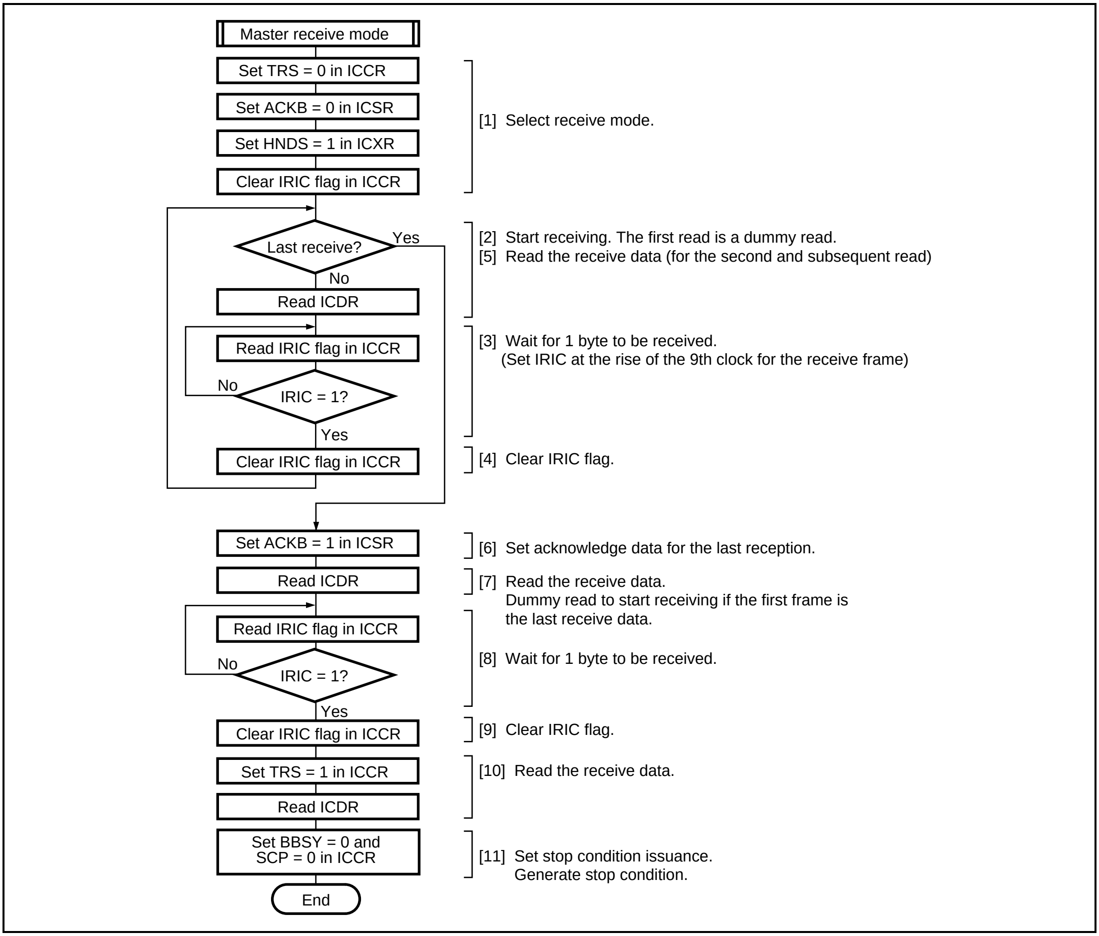


Figure 13.10 Sample Flowchart for Operations in Master Receive Mode (HNDS = 1)

The reception procedure and operations using the HNDS function, by which the data reception process is provided in 1-byte units with SCL fixed low at each data reception, are described below.

1. Clear the TRS bit in ICCR to 0 to switch from transmit mode to receive mode.

   Clear the ACKB bit in ICSR to 0 (acknowledge data setting).

   Set the HNDS bit in ICXR to 1.

   Clear the IRIC flag to 0 to determine the end of reception.

   Go to step [6] to halt reception operation if the first frame is the last receive data.
2. When ICDR is read (dummy data read), reception is started, the receive clock is output in synchronization with the internal clock, and data is received. (Data from the SDA pin is sequentially transferred to ICDRS in synchronization with the rise of the receive clock pulses.)
3. The master device drives SDA low to return the acknowledge data at the 9th receive clock pulse. The receive data is transferred from ICDRS to ICDRR at the rise of the 9th clock pulse, setting the ICDRF, IRIC, and IRTR flags to 1. If the IEIC bit has been set to 1, an interrupt request is sent to the CPU.

   The master device drives SCL low from the fall of the 9th receive clock pulse to the ICDR data reading.
4. Clear the IRIC flag to clear the wait state.

   Go to step [6] to halt reception operation if the next frame is the last receive data.
5. Read ICDR receive data. This clears the ICDRF flag to 0. The master device outputs the receive clock continuously to receive the next data.

Data can be received continuously by repeating steps [3] to [5].

6. Set the ACKB bit to 1 so as to return the acknowledge data for the last reception.
7. Read ICDR receive data. This clears the ICDRF flag to 0. The master device outputs the receive clock to receive data.
8. When one frame of data has been received, the ICDRF, IRIC, and IRTR flags are set to 1 at the rise of the 9th receive clock pulse.
9. Clear the IRIC flag to 0.
10. Read ICDR receive data after setting the TRS bit. This clears the ICDRF flag to 0.
11. 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.

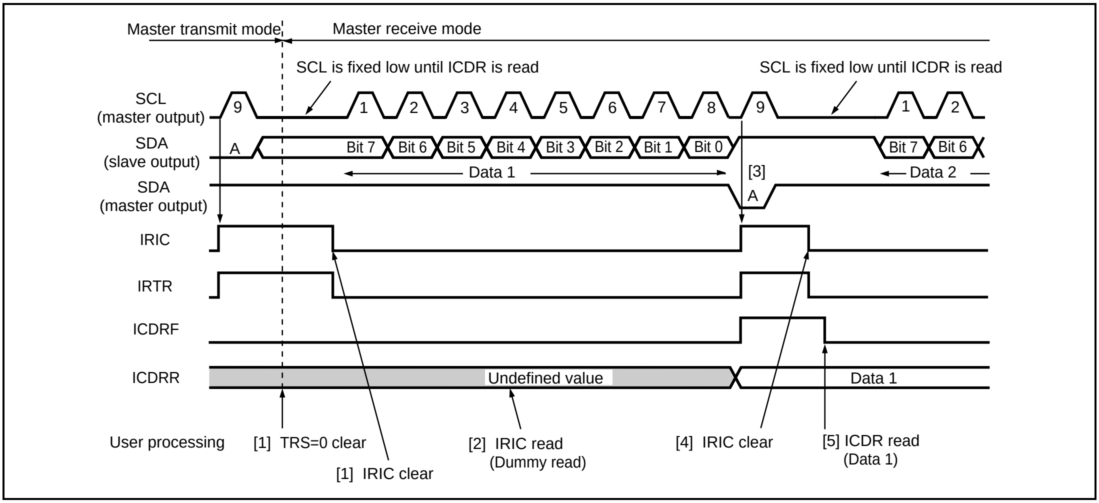


Figure 13.11 Example of Operation Timing in Master Receive Mode (MLS = WAIT = 0, HNDS = 1)

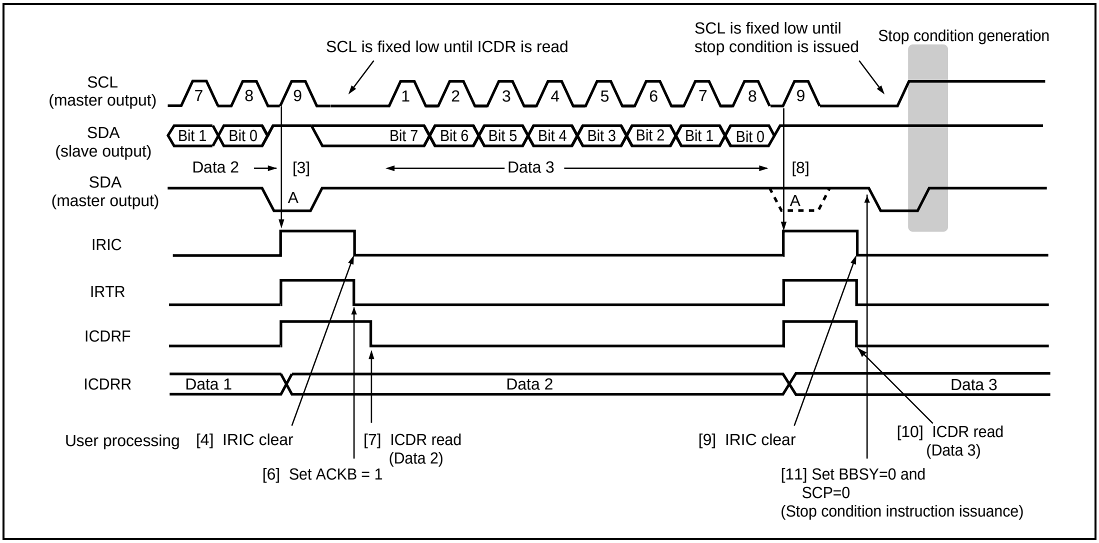
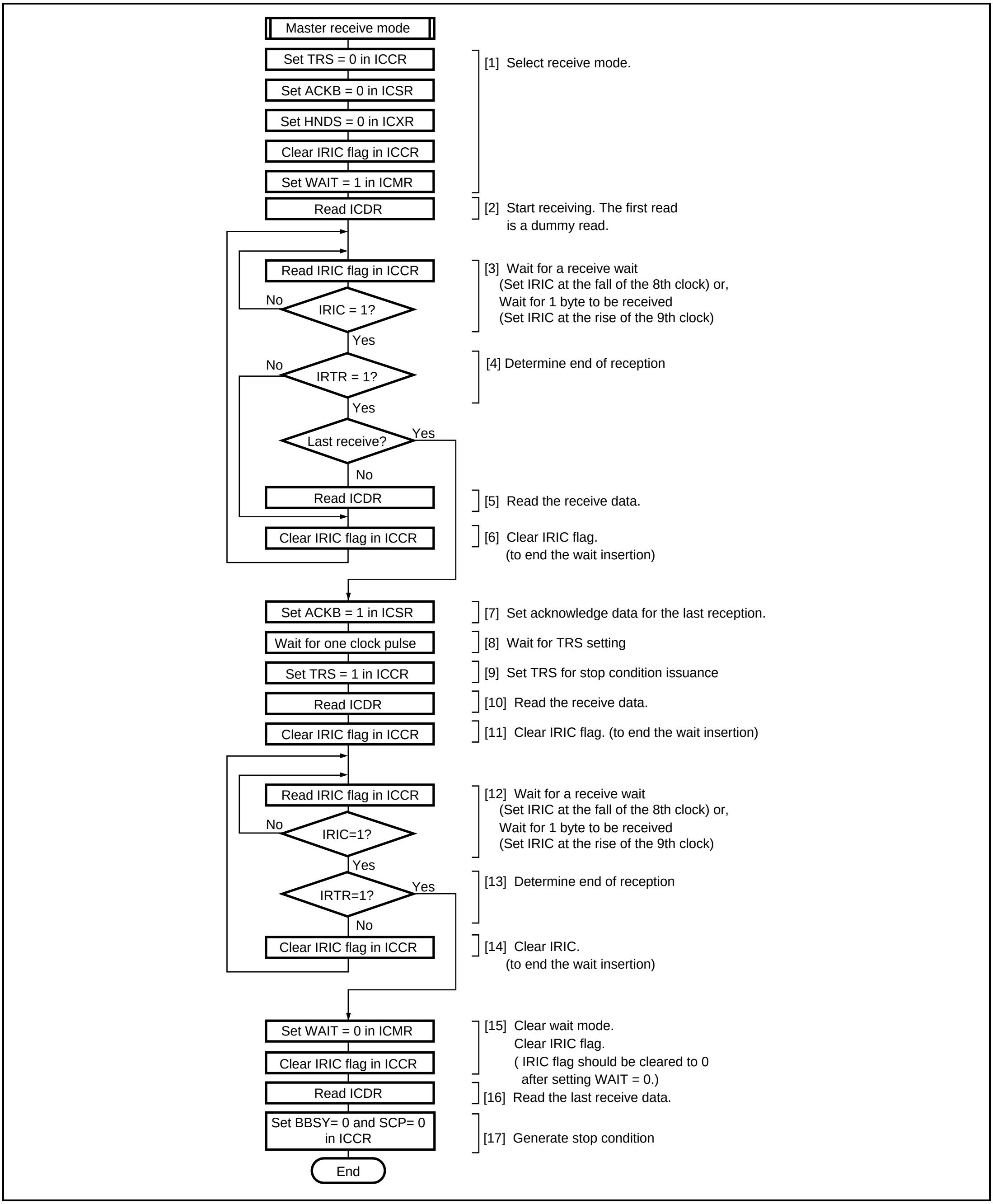


Figure 13.12 Example of Stop Condition Issuance Operation Timing in Master Receive Mode (MLS = WAIT = 0, HNDS = 1)

Receive Operation Using the Wait Function:

Figures 13.13 and 13.14 show the sample flowcharts for the operations in master receive mode (WAIT = 1).



Master receive mode
Set TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 0 in ICXR
Clear IRIC flag in ICCR
Set WAIT = 1 in ICMR
Read ICDR
Read IRIC flag in ICCR
IRIC = 1?
No
Yes
IRTR = 1?
No
Yes
Last receive?
Yes
No
Read ICDR
Clear IRIC flag in ICCR
Set ACKB = 1 in ICSR
Wait for one clock pulse
Set TRS = 1 in ICCR
Read ICDR
Clear IRIC flag in ICCR
Read IRIC flag in ICCR
IRIC=1?
No
Yes
IRTR=1?
Yes
No
Clear IRIC flag in ICCR
Set WAIT = 0 in ICMR
Clear IRIC flag in ICCR
Read ICDR
Set BBSY= 0 and SCP= 0 in ICCR
End
[1] Select receive mode.
[2] Start receiving. The first read is a dummy read.
[3] Wait for a receive wait (Set IRIC at the fall of the 8th clock) or, Wait for 1 byte to be received (Set IRIC at the rise of the 9th clock)
[4] Determine end of reception
[5] Read the receive data.
[6] Clear IRIC flag. (to end the wait insertion)
[7] Set acknowledge data for the last reception.
[8] Wait for TRS setting
[9] Set TRS for stop condition issuance
[10] Read the receive data.
[11] Clear IRIC flag. (to end the wait insertion)
[12] Wait for a receive wait (Set IRIC at the fall of the 8th clock) or, Wait for 1 byte to be received (Set IRIC at the rise of the 9th clock)
[13] Determine end of reception
[14] Clear IRIC. (to end the wait insertion)
[15] Clear wait mode. Clear IRIC flag. (IRIC flag should be cleared to 0 after setting WAIT = 0.)
[16] Read the last receive data.
[17] Generate stop condition

Figure 13.13 Sample Flowchart for Operations in Master Receive Mode (receiving multiple bytes) (WAIT = 1)

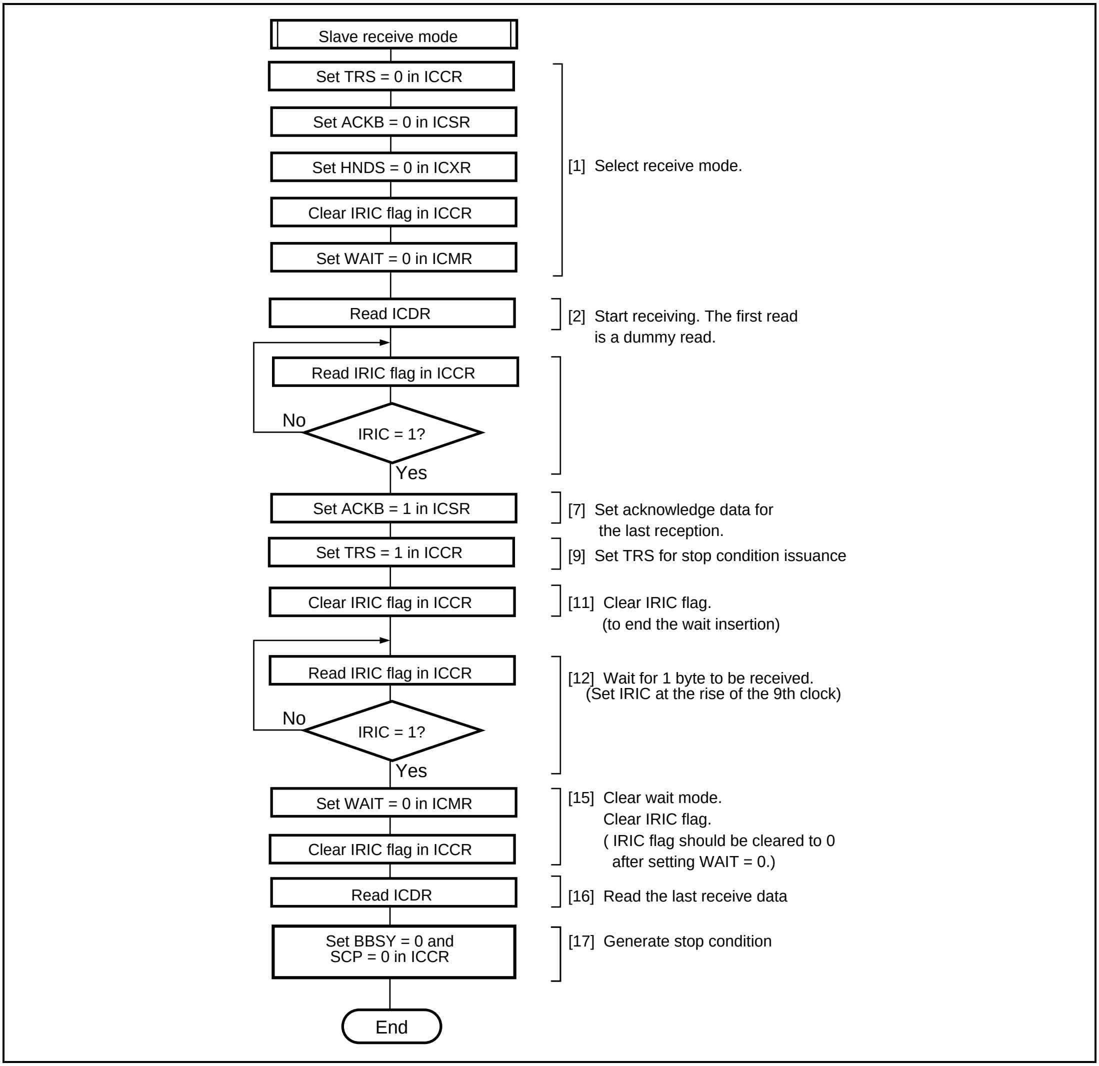

Slave receive mode
Set TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 0 in ICXR
Clear IRIC flag in ICCR
Set WAIT = 0 in ICMR
[1] Select receive mode.
Read ICDR
[2] Start receiving. The first read is a dummy read.
Read IRIC flag in ICCR
No
IRIC = 1?
Yes
Set ACKB = 1 in ICSR
[7] Set acknowledge data for the last reception.
Set TRS = 1 in ICCR
[9] Set TRS for stop condition issuance
Clear IRIC flag in ICCR
[11] Clear IRIC flag. (to end the wait insertion)
Read IRIC flag in ICCR
[12] Wait for 1 byte to be received. (Set IRIC at the rise of the 9th clock)
No
IRIC = 1?
Yes
Set WAIT = 0 in ICMR
Clear IRIC flag in ICCR
[15] Clear wait mode. Clear IRIC flag. (IRIC flag should be cleared to 0 after setting WAIT = 0.)
Read ICDR
[16] Read the last receive data
Set BBSY = 0 and SCP = 0 in ICCR
[17] Generate stop condition
End

Figure 13.14 Sample Flowchart for Operations in Master Receive Mode (receiving a single byte) (WAIT = 1)

The reception procedure and operations using the wait function (WAIT bit), by which data is sequentially received in synchronization with ICDR (ICDRR) read operations, are described below.

The following describes the multiple-byte reception procedure. In single-byte reception, some steps of the following procedure are omitted. At this time, follow the procedure shown in figure 13.14.

1. Clear the TRS bit in ICCR to 0 to switch from transmit mode to receive mode.

   Clear the ACKB bit in ICSR to 0 to set the acknowledge data.

   Clear the HNDS bit in ICXR to 0 to cancel the handshake function.

   Clear the IRIC flag to 0, and then set the WAIT bit in ICMR to 1.
2. When ICDR is read (dummy data is read), reception is started, the receive clock is output in synchronization with the internal clock, and data is received.
3. The IRIC flag is set to 1 in either of the following cases. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU.
   - At the fall of the 8th receive clock pulse for one frame

     SCL is automatically fixed low in synchronization with the internal clock until the IRIC flag clearing.
   - At the rise of the 9th receive clock pulse for one frame

     The IRTR and ICDRF flags are set to 1, indicating that one frame of data has been received. The master device outputs the receive clock continuously to receive the next data.
4. Read the IRTR flag in ICSR.

   If the IRTR flag is 0, execute step [6] to clear the IRIC flag to 0 to release the wait state.

   If the IRTR flag is 1 and the next data is the last receive data, execute step [7] to halt reception.
5. If IRTR flag is 1, read ICDR receive data.
6. Clear the IRIC flag. When the flag is set as the first case in step [3], 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 [3] to [6].

7. Set the ACKB bit in ICSR to 1 so as to return the acknowledge data for the last reception.
8. After the IRIC flag is set to 1, wait for at least one clock pulse until the rise of the first clock pulse for the next receive data.
9. Set the TRS bit in ICCR to 1 to switch from receive mode to transmit mode. The TRS bit value becomes valid when the rising edge of the next 9th clock pulse is input.
10. Read the ICDR receive data.
11. Clear the IRIC flag to 0.

12. The IRIC flag is set to 1 in either of the following cases.

    — At the fall of the 8th receive clock pulse for one frame

      SCL is automatically fixed low in synchronization with the internal clock until the IRIC flag is cleared.

    — At the rise of the 9th receive clock pulse for one frame

      The IRTR and ICDRF flags are set to 1, indicating that one frame of data has been received. The master device outputs the receive clock continuously to receive the next data.

13. Read the IRTR flag in ICSR.

    If the IRTR flag is 0, execute step [14] to clear the IRIC flag to 0 to release the wait state.

    If the IRTR flag is 1 and data reception is complete, execute step [15] to issue the stop condition.

14. If IRTR flag is 0, clear the IRIC flag to 0 to release the wait state.

    Execute step [12] to read the IRIC flag to detect the end of reception.

15. Clear the WAIT bit in CMR to cancel the wait mode.

    Then, clear the IRIC flag. 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 may not be issued correctly.)

16. Read the last ICDR receive data.

17. 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.

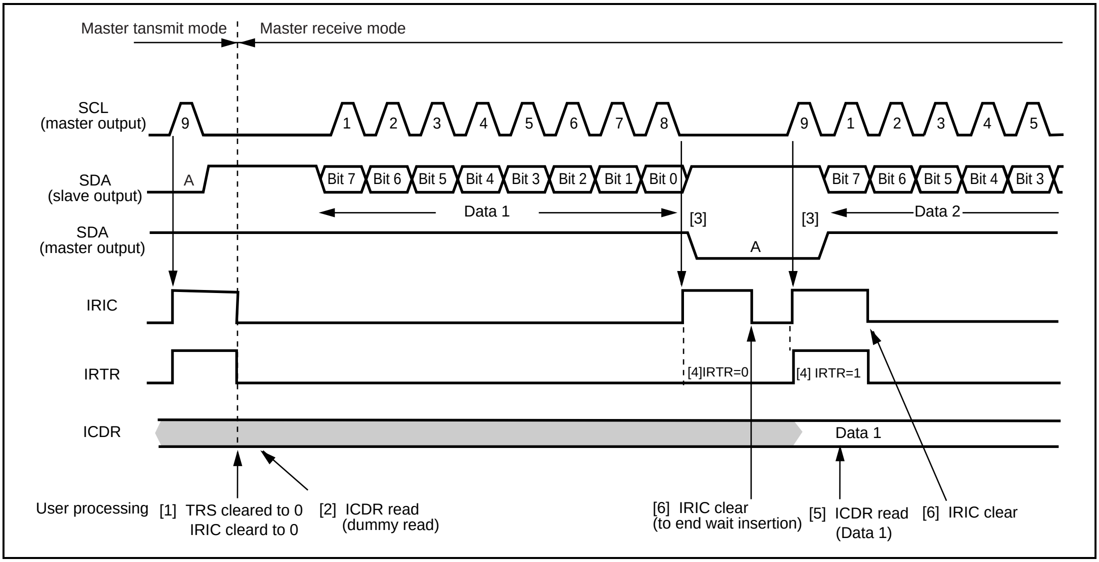


Figure 13.15 Example of Master Receive Mode Operation Timing (MLS = ACKB = 0, WAIT = 1)

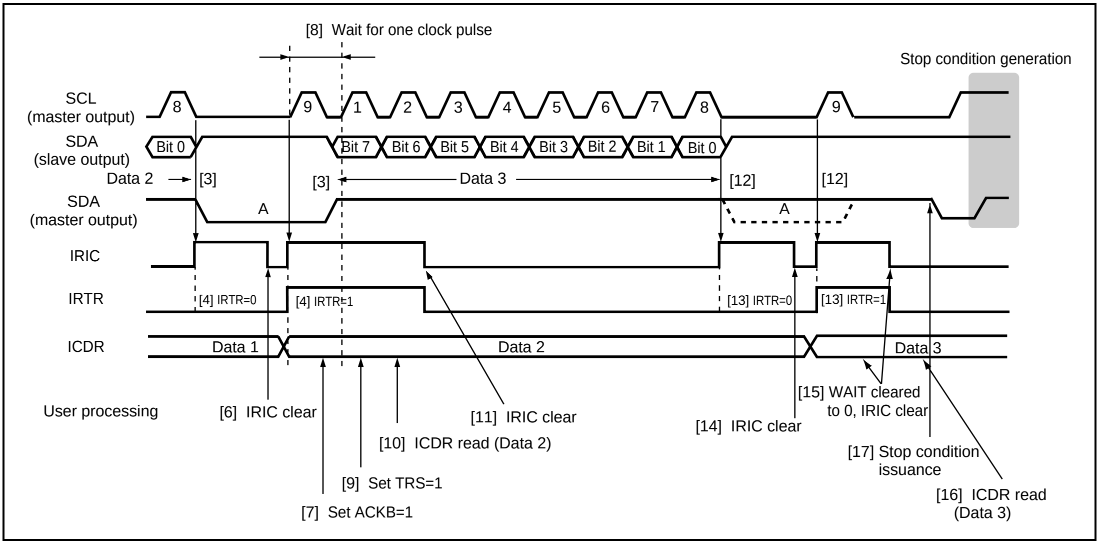


Figure 13.16 Example of Stop Condition Issuance Timing in Master Receive Mode (MLS = ACKB = 0, WAIT = 1)

13.4.5 Slave Receive Operation

In I^{2}C bus format slave receive mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal.

The slave device operates as the device specified by the master device when the slave address in the first frame following the start condition that is issued by the master device matches its own address.

Receive Operation Using the HNDS Function (HNDS = 1):

Figure 13.17 shows the sample flowchart for the operations in slave receive mode (HNDS = 1).

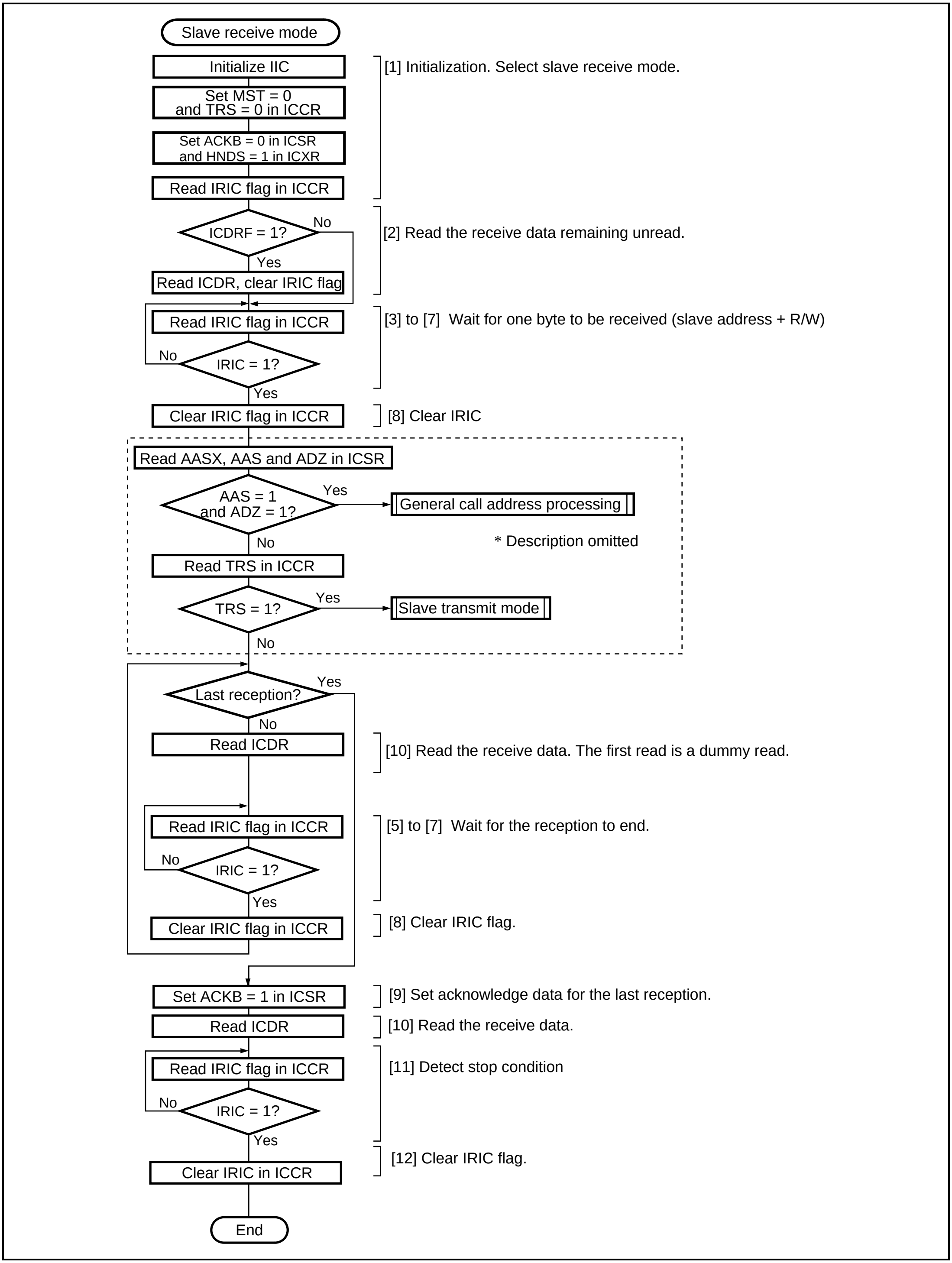

Slave receive mode
Initialize IIC
Set MST = 0 and TRS = 0 in ICCR
Set ACKB = 0 in ICSR and HNDS = 1 in ICXR
Read IRIC flag in ICCR
[1] Initialization. Select slave receive mode.
ICDRF = 1?
No
Yes
Read ICDR, clear IRIC flag
[2] Read the receive data remaining unread.
Read IRIC flag in ICCR
No
IRIC = 1?
Yes
[3] to [7] Wait for one byte to be received (slave address + R/W)
Clear IRIC flag in ICCR
[8] Clear IRIC
Read AASX, AAS and ADZ in ICSR
AAS = 1 and ADZ = 1?
Yes
General call address processing
No
* Description omitted
Read TRS in ICCR
TRS = 1?
Yes
Slave transmit mode
No
Last reception?
Yes
No
Read ICDR
[10] Read the receive data. The first read is a dummy read.
Read IRIC flag in ICCR
No
IRIC = 1?
Yes
[5] to [7] Wait for the reception to end.
Clear IRIC flag in ICCR
[8] Clear IRIC flag.
Set ACKB = 1 in ICSR
[9] Set acknowledge data for the last reception.
Read ICDR
[10] Read the receive data.
Read IRIC flag in ICCR
No
IRIC = 1?
Yes
[11] Detect stop condition
Clear IRIC in ICCR
[12] Clear IRIC flag.
End

Figure 13.17 Sample Flowchart for Operations in Slave Receive Mode (HNDS = 1)

The reception procedure and operations using the HNDS bit function, by which data reception process is provided in 1-byte unit with SCL being fixed low at every data reception, are described below.

1. Initialize the IIC as described in section 13.4.2, Initialization.

   Clear the MST and TRS bits to 0 to set slave receive mode, and set the HNDS bit to 1 and the ACKB bit to 0. Clear the IRIC flag in ICCR to 0 to see the end of reception.
2. Confirm that the ICDRF flag is 0. If the ICDRF flag is set to 1, read the ICDR and then clear the IRIC flag to 0.
3. When the start condition output by the master device is detected, the BBSY flag in ICCR is set to 1. The master device then outputs the 7-bit slave address and transmit/receive direction (R/W), in synchronization with the transmit clock pulses.
4. 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/$\overline{\text{W}}$) is 0, the TRS bit remains cleared to 0, and slave receive operation is performed. If the 8th data bit (R/$\overline{\text{W}}$) is 1, the TRS bit is set to 1, and slave transmit operation is performed. When the slave address does not match, receive operation is halted until the next start condition is detected.
5. At the 9th clock pulse of the receive frame, the slave device returns the data in the ACKB bit as an acknowledge signal.
6. At the rise of the 9th clock pulse, the IRIC flag is set to 1. If the IEIC bit has been set to 1, an interrupt request is sent to the CPU.

   If the AASX bit has been set to 1, IRTR flag is also set to 1.
7. At the rise of the 9th clock pulse, the receive data is transferred from ICDRS to ICDRR, setting the ICDRF flag to 1. The slave device drives SCL low from the fall of the 9th receive clock pulse until data is read from ICDR.
8. Confirm that the STOP bit is cleared to 0, and clear the IRIC flag to 0.
9. If the next frame is the last receive frame, set the ACKB bit to 1.
10. If ICDR is read, the ICDRF flag is cleared to 0, releasing the SCL bus line. This enables the master device to transfer the next data.

Receive operations can be performed continuously by repeating steps [5] to [10].

11. When the stop condition is detected (SDA is changed from low to high when SCL is high), the BBSY flag is cleared to 0 and the STOP bit is set to 1. If the STOPIM bit has been cleared to 0, the IRIC flag is set to 1.
12. Confirm that the STOP bit is set to 1, and clear the IRIC flag to 0.

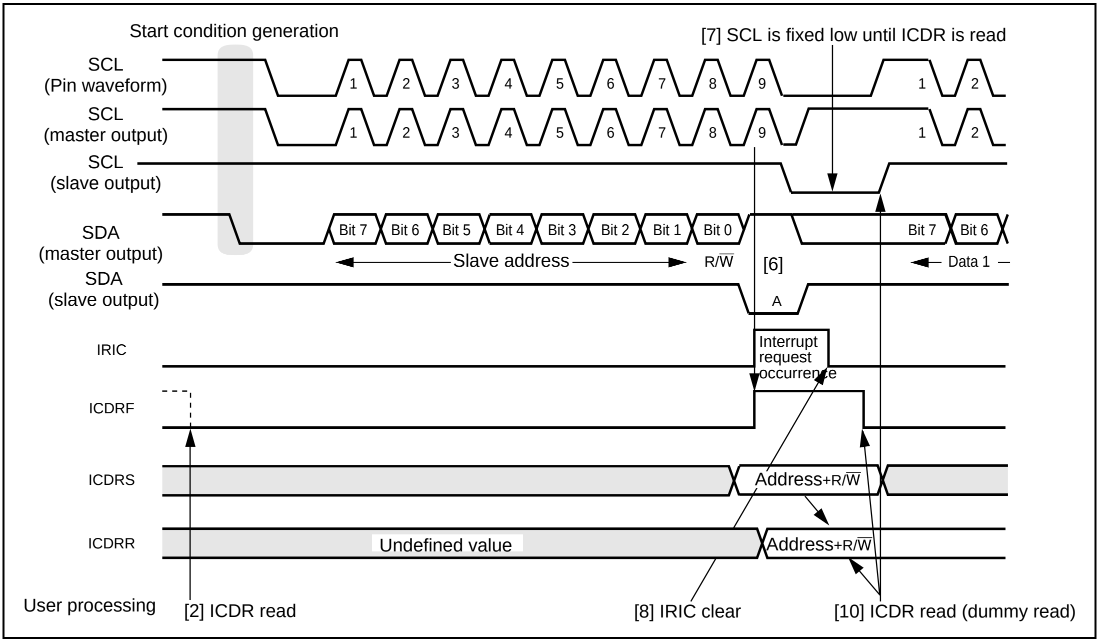


Figure 13.18 Example of Slave Receive Mode Operation Timing (1) (MLS = 0, HNDS= 1)

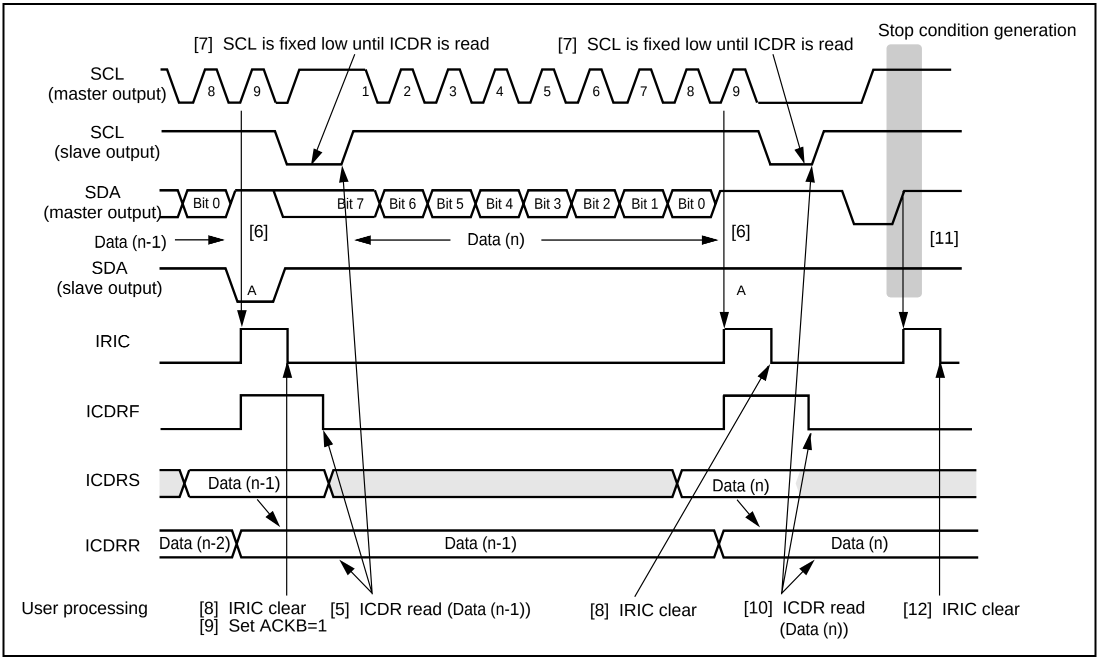


Figure 13.19 Example of Slave Receive Mode Operation Timing (2) (MLS = 0, HNDS= 1)

Continuous Receive Operation:

Figure 13.20 shows the sample flowchart for the operations in slave receive mode (HNDS = 0).

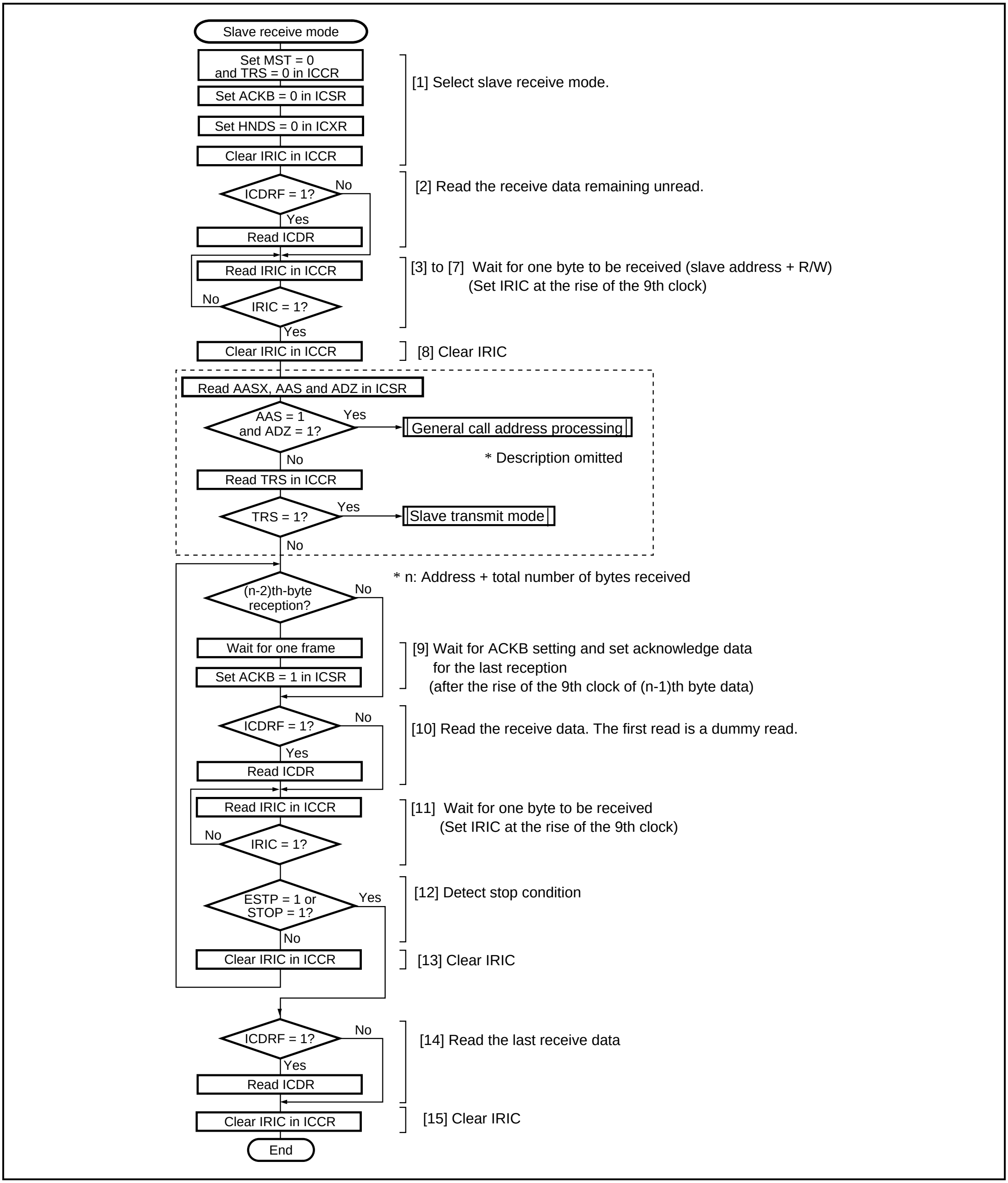

Slave receive mode
Set MST = 0 and TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 0 in ICXR
Clear IRIC in ICCR
[1] Select slave receive mode.
ICDRF = 1?
No
Yes
Read ICDR
[2] Read the receive data remaining unread.
Read IRIC in ICCR
IRIC = 1?
No
Yes
[3] to [7] Wait for one byte to be received (slave address + R/W)
(Set IRIC at the rise of the 9th clock)
Clear IRIC in ICCR
[8] Clear IRIC
Read AASX, AAS and ADZ in ICSR
AAS = 1 and ADZ = 1?
Yes
General call address processing
No
* Description omitted
Read TRS in ICCR
TRS = 1?
Yes
Slave transmit mode
No
* n: Address + total number of bytes received
(n-2)th-byte reception?
No
Wait for one frame
Set ACKB = 1 in ICSR
[9] Wait for ACKB setting and set acknowledge data
for the last reception
(after the rise of the 9th clock of (n-1)th byte data)
ICDRF = 1?
No
Yes
Read ICDR
[10] Read the receive data. The first read is a dummy read.
Read IRIC in ICCR
No
IRIC = 1?
[11] Wait for one byte to be received
(Set IRIC at the rise of the 9th clock)
ESTP = 1 or STOP = 1?
Yes
No
[12] Detect stop condition
Clear IRIC in ICCR
[13] Clear IRIC
ICDRF = 1?
No
Yes
Read ICDR
[14] Read the last receive data
Clear IRIC in ICCR
[15] Clear IRIC
End

Figure 13.20 Sample Flowchart for Operations in Slave Receive Mode (HNDS = 0)

The reception procedure and operations in slave receive are described below.

1. Initialize the IIC as described in section 13.4.2, Initialization.

   Clear the MST and TRS bits to 0 to set slave receive mode, and set the HNDS and ACKB bits to 0. Clear the IRIC flag in ICCR to 0 to see the end of reception.
2. Confirm that the ICDRF flag is 0. If the ICDRF flag is set to 1, read the ICDR and then clear the IRIC flag to 0.
3. When the start condition output by the master device is detected, the BBSY flag in ICCR is set to 1. The master device then outputs the 7-bit slave address and transmit/receive direction (R/W) in synchronization with the transmit clock pulses.
4. 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/$\overline{W}$) is 0, the TRS bit remains cleared to 0, and slave transmit operation is performed. When the slave address does not match, receive operation is halted until the next start condition is detected.
5. At the 9th clock pulse of the receive frame, the slave device returns the data in the ACKB bit as an acknowledge signal.
6. At the rise of the 9th clock pulse, the IRIC flag is set to 1. If the IEIC bit has been set to 1, an interrupt request is sent to the CPU.

   If the AASX bit has been set to 1, the IRTR flag is also set to 1.
7. At the rise of the 9th clock pulse, the receive data is transferred from ICDRS to ICDRR, setting the ICDRF flag to 1.
8. Confirm that the STOP bit is cleared to 0 and clear the ICIC flag to 0.
9. If the next read data is the third last receive frame, wait for at least one frame time to set the ACKB bit. Set the ACKB bit after the rise of the 9th clock pulse of the second last receive frame.
10. Confirm that the ICDRF flag is set to 1 and read ICDR. This clears the ICDRF flag to 0.
11. At the rise of the 9th clock pulse or when the receive data is transferred from IRDRS to ICDRR due to ICDR read operation, the IRIC and ICDRF flags are set to 1.
12. When the stop condition is detected (SDA is changed from low to high when SCL is high), the BBSY flag is cleared to 0 and the STOP or ESTP flag is set to 1. If the STOPIM bit has been cleared to 0, the IRIC flag is set to 1. In this case, execute step [14] to read the last receive data.
13. Clear the IRIC flag to 0.

Receive operations can be performed continuously by repeating steps [9] to [13].

14. Confirm that the ICDRF flag is set to 1, and read ICDR.
15. Clear the IRIC flag.

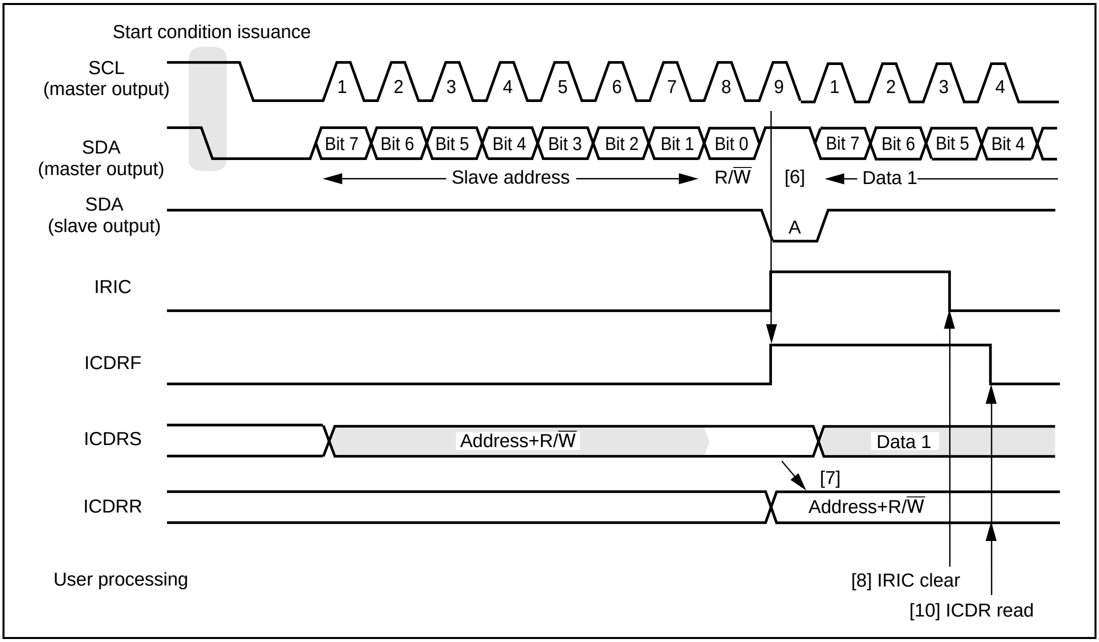


Figure 13.21 Example of Slave Receive Mode Operation Timing (1)
(MLS = ACKB = 0, HNDS = 0)

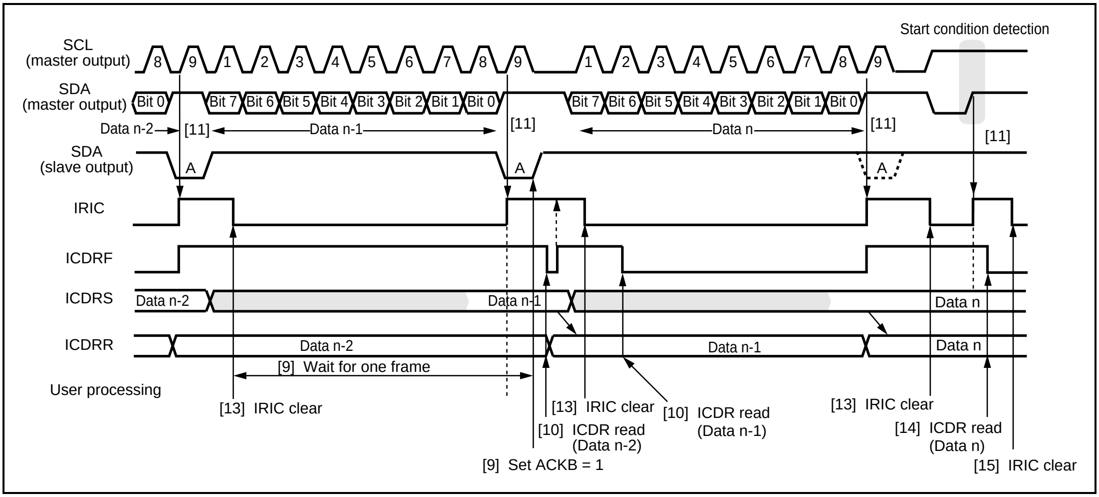


Figure 13.22 Example of Slave Receive Mode Operation Timing (2)
(MLS = ACKB = 0, HNDS = 0)

13.4.6 Slave Transmit Operation

If the slave address matches to the address in the first frame (address reception frame) following the start condition detection when the 8th bit data (R/$\overline{W}$) is 1 (read), the TRS bit in ICCR is automatically set to 1 and the mode changes to slave transmit mode.

Figure 13.23 shows the sample flowchart for the operations in slave transmit mode.

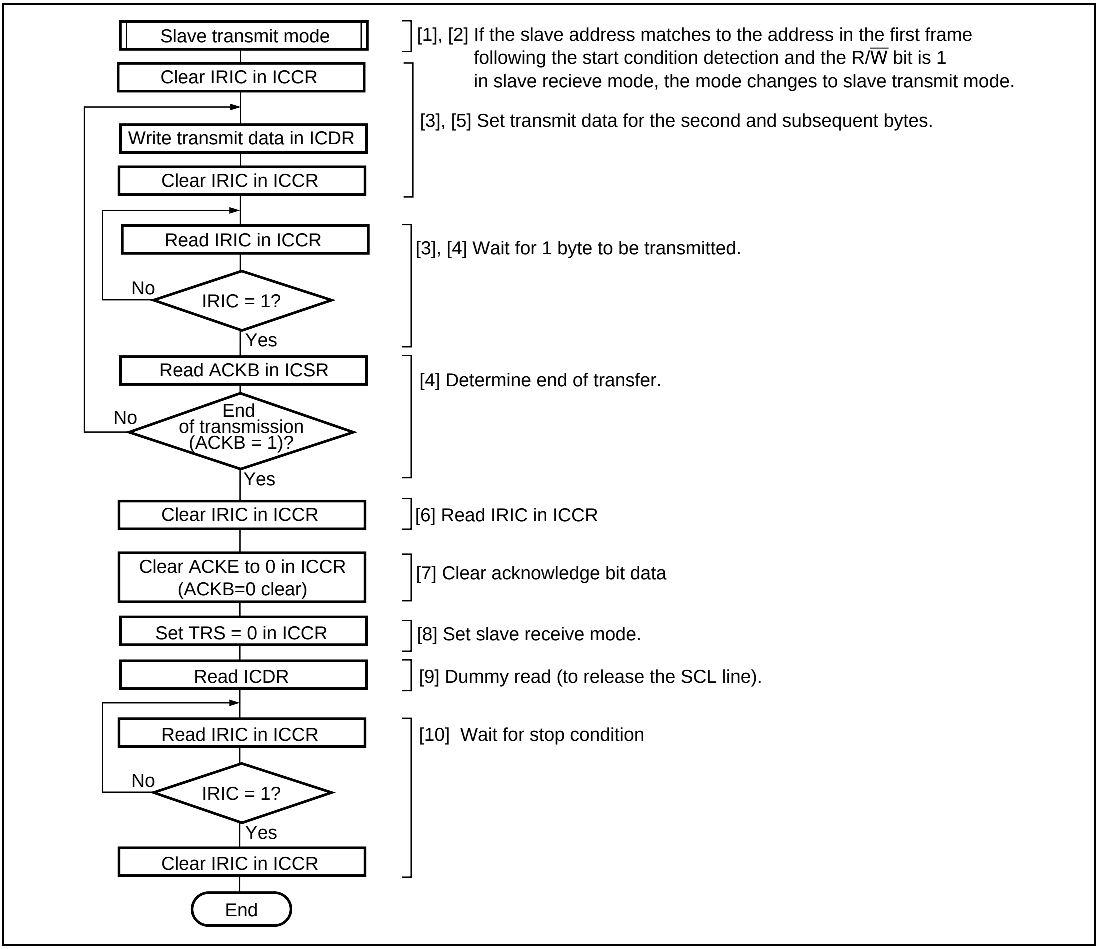

Slave transmit mode
Clear IRIC in ICCR
Write transmit data in ICDR
Clear IRIC in ICCR
Read IRIC in ICCR
No
IRIC = 1?
Yes
Read ACKB in ICSR
No
End of transmission (ACKB = 1)?
Yes
Clear IRIC in ICCR
Clear ACKE to 0 in ICCR (ACKB=0 clear)
Set TRS = 0 in ICCR
Read ICDR
Read IRIC in ICCR
No
IRIC = 1?
Yes
Clear IRIC in ICCR
End
[1], [2] If the slave address matches to the address in the first frame following the start condition detection and the R/W bit is 1 in slave recieve mode, the mode changes to slave transmit mode.
[3], [5] Set transmit data for the second and subsequent bytes.
[3], [4] Wait for 1 byte to be transmitted.
[4] Determine end of transfer.
[6] Read IRIC in ICCR
[7] Clear acknowledge bit data
[8] Set slave receive mode.
[9] Dummy read (to release the SCL line).
[10] Wait for stop condition

Figure 13.23 Sample Flowchart for Slave Transmit Mode

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. Initialize slave receive mode and wait for slave address reception.
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. If the 8th data bit (R/W) is 1, the TRS bit in ICCR is set to 1, and the mode changes to slave transmit mode automatically. The IRIC flag is set to 1 at the rise of the 9th clock. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU. At the same time, the ICDRE flag is set to 1. The slave device drives SCL low from the fall of the transmit clock until ICDR data is written, to disable the master device to output the next transfer clock.
3. After clearing the IRIC flag to 0, write data to ICDR. At this time, the ICDRE flag is cleared to 0. The written data is transferred to ICDRS, and the ICDRE and IRIC flags are set to 1 again. The slave device sequentially sends the data written into ICDRS in accordance with the clock output by the master device.

   The IRIC flag is cleared to 0 to detect the end of transmission. Processing from the ICDR register writing to the IRIC flag clearing should be performed continuously. Prevent any other interrupt processing from being inserted.
4. 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 successfully. 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. When the ICDRE flag is 0, the data written into ICDR is transferred to ICDRS, transmission starts, and the ICDRE and IRIC flags are set to 1 again. If the ICDRE 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.
5. To continue transmission, write the next data to be transmitted into ICDR. The ICDRE flag is cleared to 0. The IRIC flag is cleared to 0 to detect the end of transmission. Processing from the ICDR register writing to the IRIC flag clearing should be performed continuously. Prevent any other interrupt processing from being inserted.

Transmit operations can be performed continuously by repeating steps [4] and [5].

6. Clear the IRIC flag to 0.
7. To end transmission, clear the ACKE bit in ICCR to 0, to clear the acknowledge bit stored in the ACKB bit to 0.
8. Clear the TRS bit to 0 for the next address reception, to set slave receive mode.
9. Dummy-read ICDR to release SDA on the slave side.

10. When the stop condition is detected, that is, when SDA is changed from low to high when SCL is high, the BBSY flag in ICCR is cleared to 0 and the STOP flag in ICSR is set to 1. When the STOPIM bit in ICXR is 0, the IRIC flag is set to 1. If the IRIC flag has been set, it is cleared to 0.

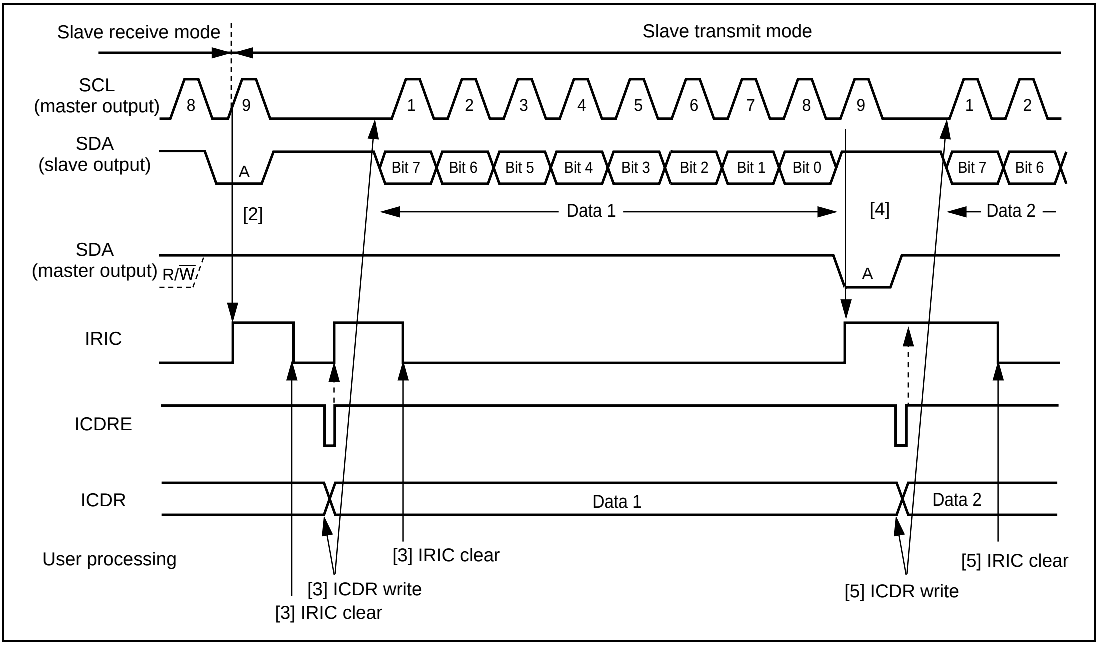


Figure 13.24 Example of Slave Transmit Mode Operation Timing (MLS = 0)

13.4.7 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 flag is set to 1, SCL is automatically held low after one frame has been transferred in synchronization with the internal clock. Figures 13.25 to 13.27 show the IRIC set timing and SCL control.

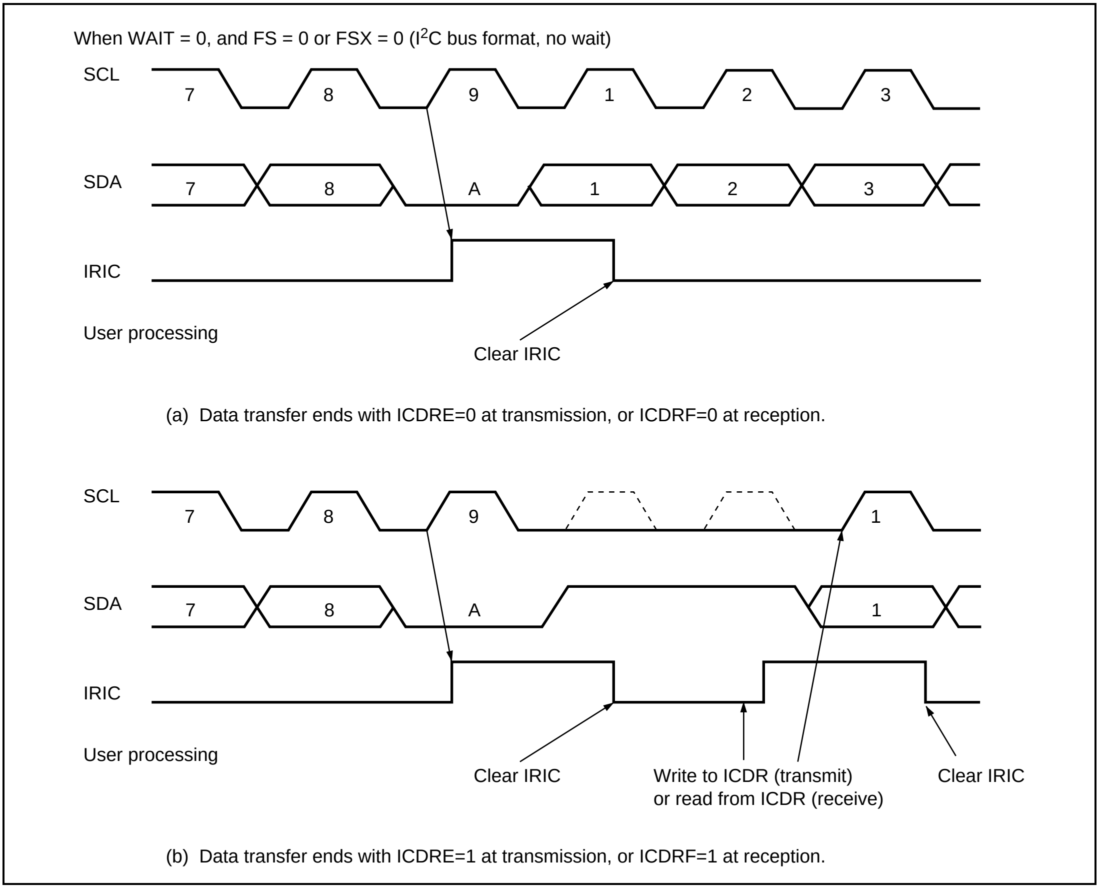

When WAIT = 0, and FS = 0 or FSX = 0 (I2C bus format, no wait)
SCL
7
8
9
1
2
3
SDA
7
8
A
1
2
3
IRIC
User processing
Clear IRIC
(a) Data transfer ends with ICDRE=0 at transmission, or ICDRF=0 at reception.
SCL
7
8
9
1
SDA
7
8
A
1
IRIC
User processing
Clear IRIC
Write to ICDR (transmit)
or read from ICDR (receive)
Clear IRIC
(b) Data transfer ends with ICDRE=1 at transmission, or ICDRF=1 at reception.

Figure 13.25 IRIC Setting Timing and SCL Control (1)

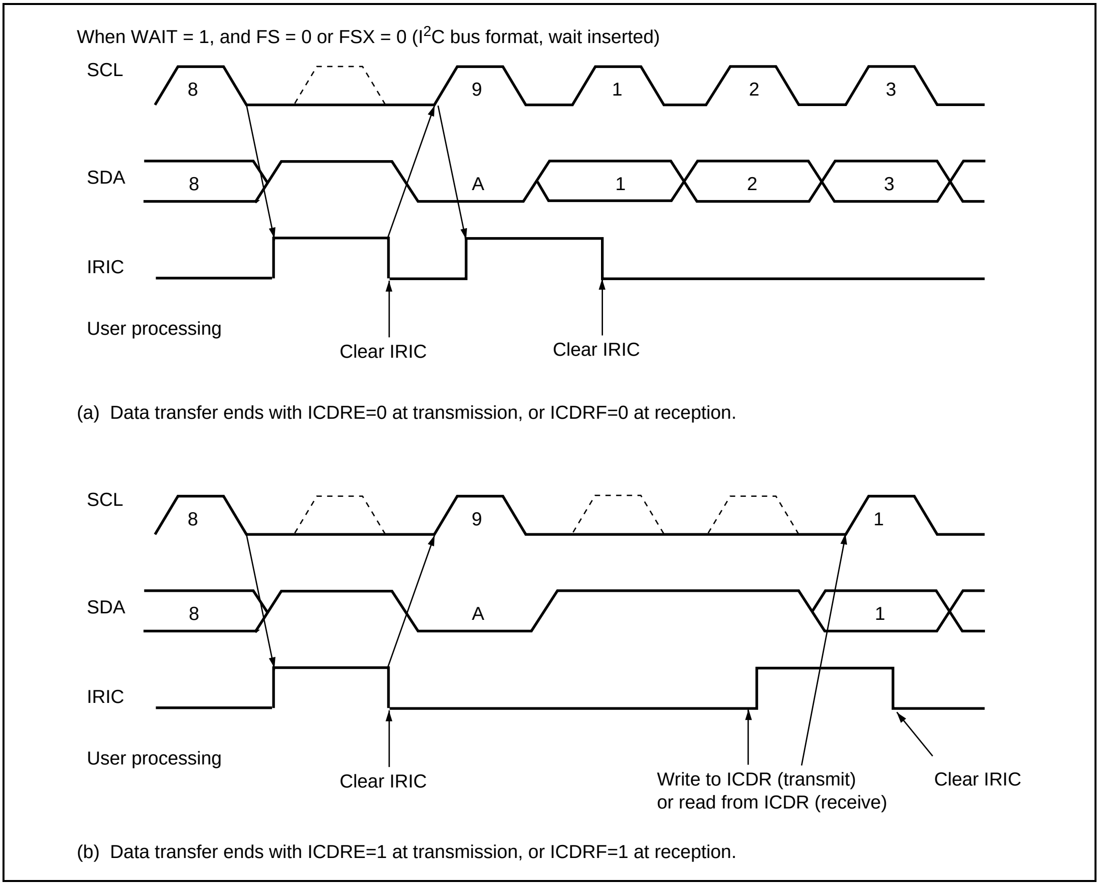

When WAIT = 1, and FS = 0 or FSX = 0 (I2C bus format, wait inserted)
SCL
8
9
1
2
3
SDA
8
A
1
2
3
IRIC
User processing
Clear IRIC
Clear IRIC
(a) Data transfer ends with ICDRE=0 at transmission, or ICDRF=0 at reception.
SCL
8
9
1
SDA
8
A
1
IRIC
User processing
Clear IRIC
Write to ICDR (transmit)
or read from ICDR (receive)
Clear IRIC
(b) Data transfer ends with ICDRE=1 at transmission, or ICDRF=1 at reception.

Figure 13.26 IRIC Setting Timing and SCL Control (2)

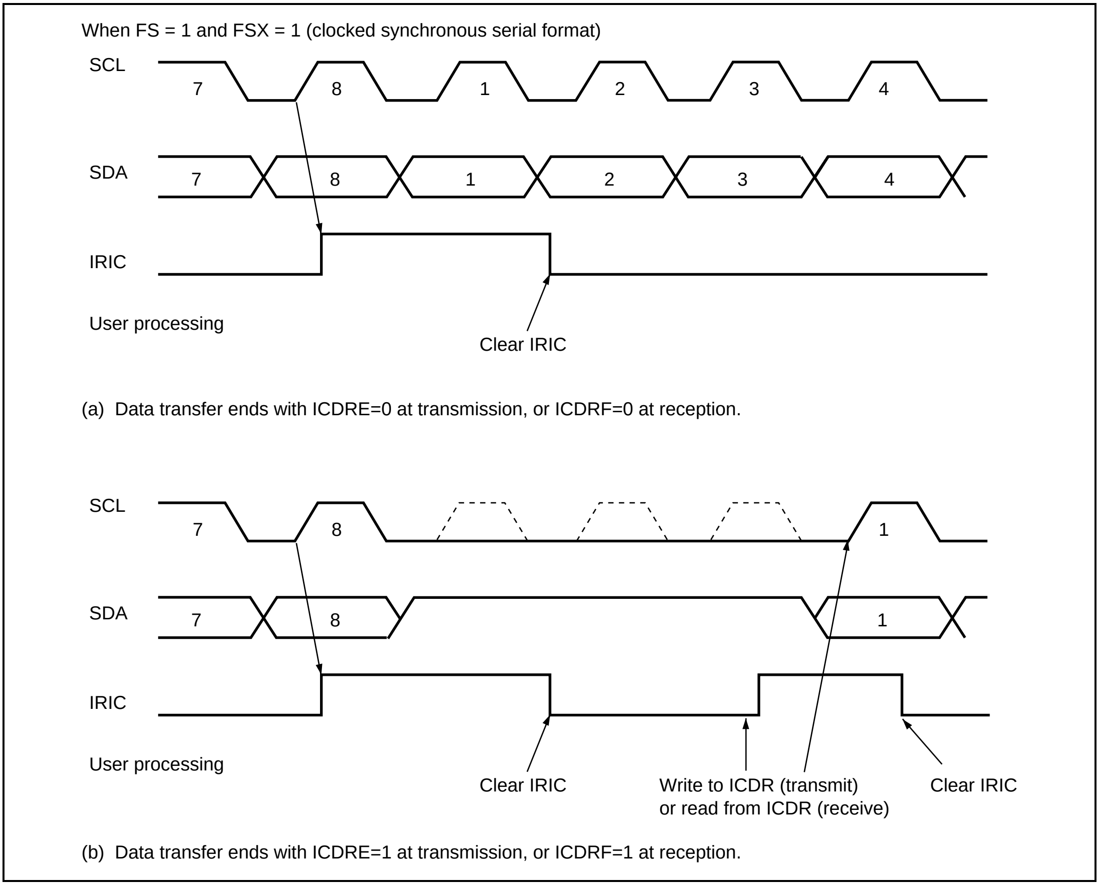

When FS = 1 and FSX = 1 (clocked synchronous serial format)
SCL
7
8
1
2
3
4
SDA
7
8
1
2
3
4
IRIC
User processing
Clear IRIC
(a) Data transfer ends with ICDRE=0 at transmission, or ICDRF=0 at reception.
SCL
7
8
1
SDA
7
8
1
IRIC
User processing
Clear IRIC
Write to ICDR (transmit)
or read from ICDR (receive)
Clear IRIC
(b) Data transfer ends with ICDRE=1 at transmission, or ICDRF=1 at reception.

Figure 13.27 IRIC Setting Timing and SCL Control (3)

13.4.8 Noise Canceler

The logic levels at the SCL and SDA pins are routed through noise cancelers before being latched internally. Figure 13.28 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.

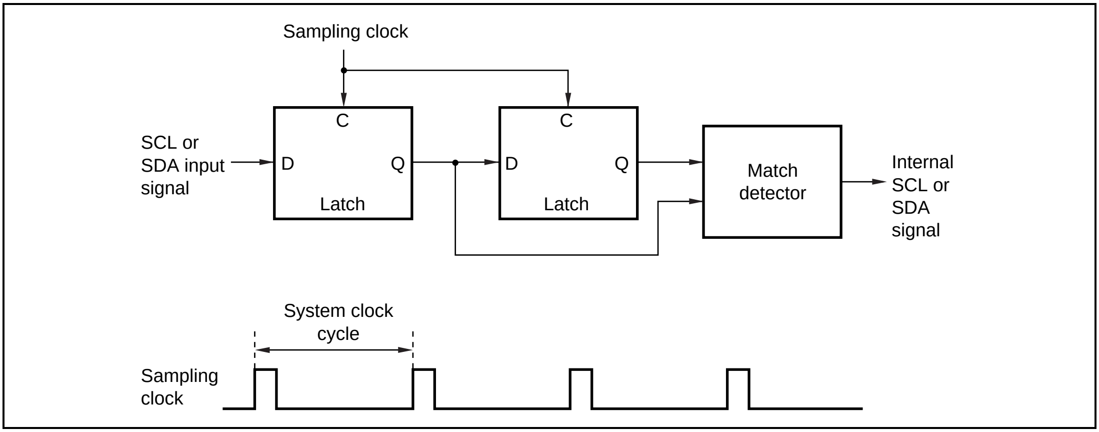


Figure 13.28 Block Diagram of Noise Canceler

13.4.9 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 DDCSWR or clearing ICE bit. For details on the setting of bits CLR3 to CLR0, see section 13.3.7, DDC Switch Register (DDCSWR).

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 (except for the ICDRE and ICDRF flags)
- Internal latches used to retain register read information for setting/clearing flags in ICMR, ICCR, and ICSR
- The value of the ICMR 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 DDCSWR, 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.

13.5 Interrupt Sources

The IIC has interrupt source IICI. Table 13.7 shows the interrupt sources and priority. Individual interrupt sources can be enabled or disabled using the enable bits in ICCR, and are sent to the interrupt controller independently.

Table 13.7 IIC Interrupt Sources

Channel	Name	Enable Bit	Interrupt Source	Interrupt Flag	Priority
0	IICI0	IEIC	I^2C bus interface interrupt request	IRIC	High ↑ Low
1	IICI1	IEIC	I^2C bus interface interrupt request	IRIC	

13.6 Usage Notes

1. In master mode, if an instruction to generate a start condition is issued and then an instruction to generate a stop condition is issued before the start condition is output to the I^2C bus, neither condition will be output correctly. To output the start condition followed by the stop condition, after issuing the instruction that generates the start condition, read DR in each I^2C bus output pin, and check that SCL and SDA are both low. The pin states can be monitored by reading DR even if the ICE bit is set to 1. 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 ICDRT to ICDRS)
   - Read from ICDR when ICE = 1 and TRS = 0 (including automatic transfer from ICDRS to ICDRR)
3. Table 13.8 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 13.8 I²C Bus Timing (SCL and SDA Outputs)

Item	Symbol	Output Timing	Unit	Notes
SCL output cycle time	t_{SCLO}	$28t_{cyc}$ to $256t_{cyc}$	ns	See figure 21.21.
SCL output high pulse width	t_{SCLHO}	$0.5t_{SCLO}$	ns	
SCL output low pulse width	t_{SCLLO}	$0.5t_{SCLO}$	ns	
SDA output bus free time	t_{BUFO}	$0.5t_{SCLO} – 1t_{cyc}$	ns	
Start condition output hold time	t_{STAHO}	$0.5t_{SCLO} – 1t_{cyc}$	ns	
Retransmission start condition output setup time	t_{STASO}	$1t_{SCLO}$	ns	
Stop condition output setup time	t_{STOSO}	$0.5t_{SCLO} + 2t_{cyc}$	ns	
Data output setup time (master)	t_{SDASO}	$1t_{SCLLO} – 3t_{cyc}$	ns	
Data output setup time (slave)		$1t_{SCLL} – (6t_{cyc}$ or $12t_{cyc}$*)		
Data output hold time	t_{SDAHO}	$3t_{cyc}$	ns	

Note:* $6t_{cyc}$ when IICX is 0, $12t_{cyc}$ 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_{cyc}, as shown in section 21, Electrical Characteristics. Note that the I²C bus interface AC timing specifications will not be met with a system clock frequency of less than 5 MHz.
5. The I²C 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 I²C 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 I²C 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 13.9.

Table 13.9 Permissible SCL Rise Time (t_{sr}) Values

			Time Indication [ns]			
IICX	t_{cyc} Indication		I^2C Bus Specification (Max.)	φ = 5 MHz	φ = 8 MHz	φ = 10 MHz
0	7.5 t_{cyc}	Standard mode	1000	1000	937	750
		High-speed mode	300	300	300	300
1	17.5 t_{cyc}	Standard mode	1000	1000	1000	1000
		High-speed mode	300	300	300	300

6. The I^2C bus interface specifications for the SCL and SDA rise and fall times are under 1000 ns and 300 ns. The I^2C bus interface SCL and SDA output timing is prescribed by t_{cyc}, as shown in table 13.8. However, because of the rise and fall times, the I^2C bus interface specifications may not be satisfied at the maximum transfer rate. Table 13.10 shows output timing calculations for different operating frequencies, including the worst-case influence of rise and fall times.

   t_{BUFO} fails to meet the I^2C bus interface specifications at any frequency. The solution is either (a) to provide coding to secure the necessary interval (approximately 1 µs) between issuance of a stop condition and issuance of a start condition, or (b) to select devices whose input timing permits this output timing for use as slave devices connected to the I^2C bus.

   t_{SCLLO} in high-speed mode and t_{STASO} in standard mode fail to satisfy the I^2C bus interface specifications for worst-case calculations of t_{Sr}/t_{Sf}. Possible solutions that should be investigated include (a) adjusting the rise and fall times by means of a pull-up resistor and capacitive load, (b) reducing the transfer rate to meet the specifications, or (c) selecting devices whose input timing permits this output timing for use as slave devices connected to the I^2C bus.

Table 13.10 I^{2}C Bus Timing (with Maximum Influence of t_{Sr}/t_{Sf})

			Time Indication (at Maximum Transfer Rate) [ns]				
Item	**t_{cyc} Indication**		**t_{Sr}/t_{Sf} Influence (Max.)**	**I^{2}C Bus Specification (Min.)**	**φ = 5 MHz**	**φ = 8 MHz**	**φ = 10 MHz**
t_{SCLHO}	0.5 t_{SCLO} ($-t_{Sr}$)	Standard mode	–1000	4000	4000	4000	4000
		High-speed mode	–300	600	950	950	950
t_{SCLLO}	0.5 t_{SCLO} ($-t_{Sf}$)	Standard mode	–250	4700	4750	4750	4750
		High-speed mode	–250	1300	1000*[1]	1000*[1]	1000*[1]
t_{BUFO}	0.5 t_{SCLO} –1 t_{cyc} ($-t_{Sr}$)	Standard mode	–1000	4700	3800*[1]	3875*[1]	3900*[1]
		High-speed mode	–300	1300	750*[1]	825*[1]	850*[1]
t_{STAHO}	0.5 t_{SCLO} –1 t_{cyc} ($-t_{Sf}$)	Standard mode	–250	4000	4550	4625	4650
		High-speed mode	–250	600	800	875	900
t_{STASO}	1 t_{SCLO} ($-t_{Sr}$)	Standard mode	–1000	4700	9000	9000	9000
		High-speed mode	–300	600	2200	2200	2200
t_{STOSO}	0.5 t_{SCLO} + 2 t_{cyc} ($-t_{Sr}$)	Standard mode	–1000	4000	4400	4250	4200
		High-speed mode	–300	600	1350	1200	1150
t_{SDASO} (master)	1 t_{SCLLO}*[3] –3 t_{cyc} ($-t_{Sr}$)	Standard mode	–1000	250	3100	3325	3400
		High-speed mode	–300	100	400	625	700
t_{SDASO} (slave)	1 t_{SCLL}*[3] –12 t_{cyc}*[2] ($-t_{Sr}$)	Standard mode	–1000	250	1300	2200	2500
		High-speed mode	–300	100	–1400*[1]	–500*[1]	–200*[1]
t_{SDAHO}	3 t_{cyc}	Standard mode	0	0	600	375	300
		High-speed mode	0	0	600	375	300

Notes: 1. Does not meet the I^{2}C bus interface specification. 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 CKS0 to CKS2. Depending on the frequency it may not be possible to achieve the maximum transfer rate; therefore, whether or not the I^{2}C bus interface specifications are met must be determined in accordance with the actual setting conditions.

2. Value when the IICX bit is set to 1. When the IICX bit is cleared to 0, the value is (t_{SCLL} – 6t_{cyc}).

3. Calculated using the I^{2}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 at the end 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 SDA from low to high when SCL 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 (ICDRR), 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 of 0 to BBSY and SCP in ICCR) and the actual generation of the stop condition, the clock may not be output correctly in subsequent master transmission.

Clearing of the MST bit after completion of master transmission/reception, or other modifications of IIC control bits to change the transmit/receive operating mode or settings, must be carried out during interval (a) in figure 13.29 (after confirming that the BBSY bit in ICCR has been cleared to 0).

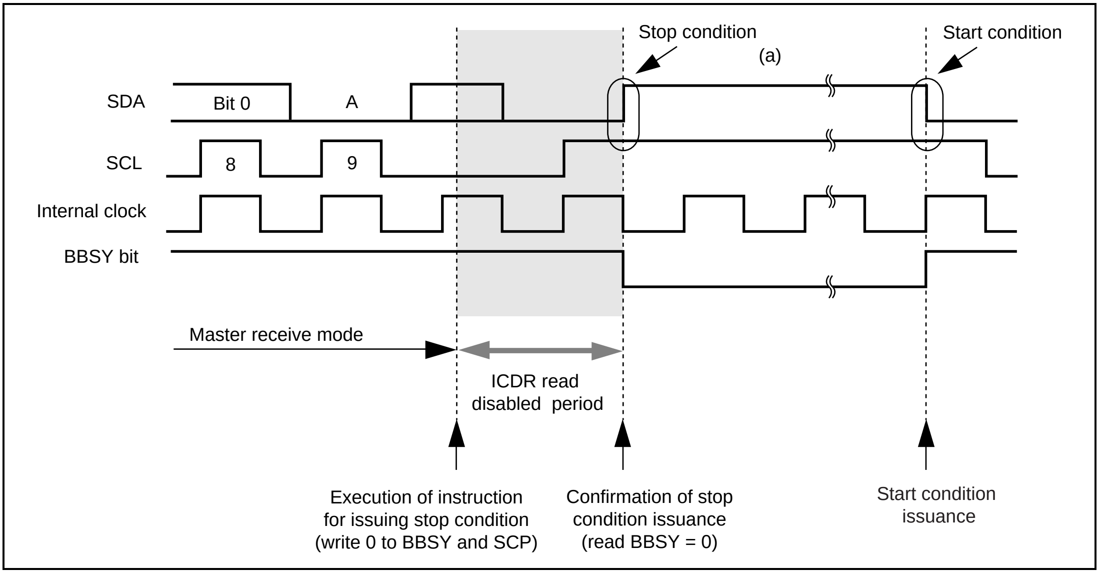
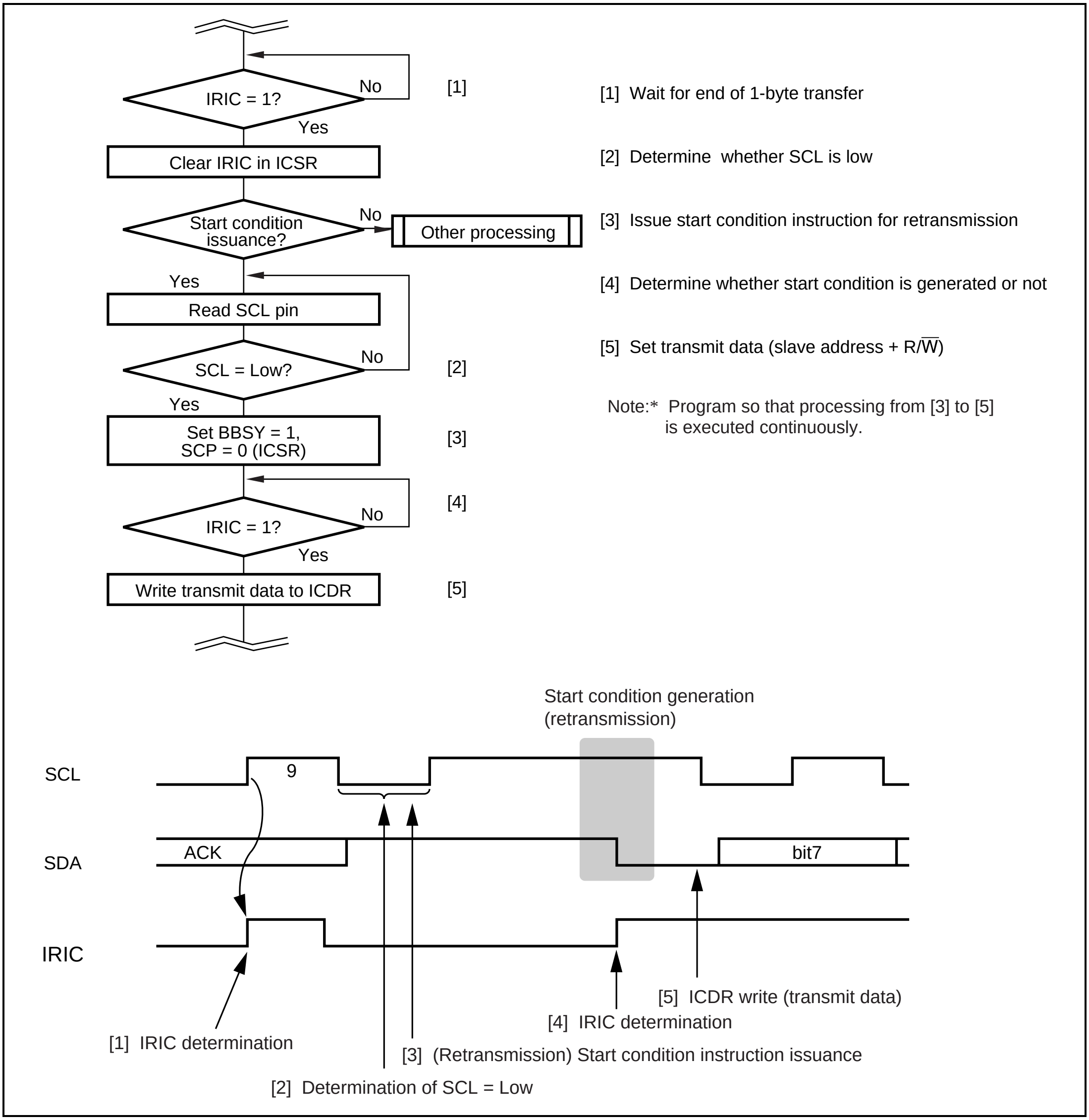


Figure 13.29 Notes on Reading Master Receive Data

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to 1 in ICXR.

8. Notes on start condition issuance for retransmission

   Figure 13.30 shows the timing of start condition issuance for retransmission, and the timing for subsequently writing data to ICDR, together with the corresponding flowchart. Write the transmit data to ICDR after the start condition for retransmission is issued and then the start condition is actually generated.


IRIC = 1?
No
Yes
[1]
Clear IRIC in ICSR
Start condition issuance?
No
Other processing
Yes
Read SCL pin
SCL = Low?
No
[2]
Yes
Set BBSY = 1, SCP = 0 (ICSR)
[3]
[4]
IRIC = 1?
No
Yes
Write transmit data to ICDR
[5]
Start condition generation (retransmission)
SCL
9
SDA
ACK
bit7
IRIC
[5] ICDR write (transmit data)
[4] IRIC determination
[1] IRIC determination
[3] (Retransmission) Start condition instruction issuance
[2] Determination of SCL = Low

[1] Wait for end of 1-byte transfer

[2] Determine whether SCL is low

[3] Issue start condition instruction for retransmission

[4] Determine whether start condition is generated or not

[5] Set transmit data (slave address + R/$\overline{W}$)

Note:* Program so that processing from [3] to [5] is executed continuously.

Figure 13.30 Flowchart for Start Condition Issuance Instruction for Retransmission and Timing

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to 1 in ICXR.

9. Note on when I^{2}C bus interface stop condition instruction is issued

In cases 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.

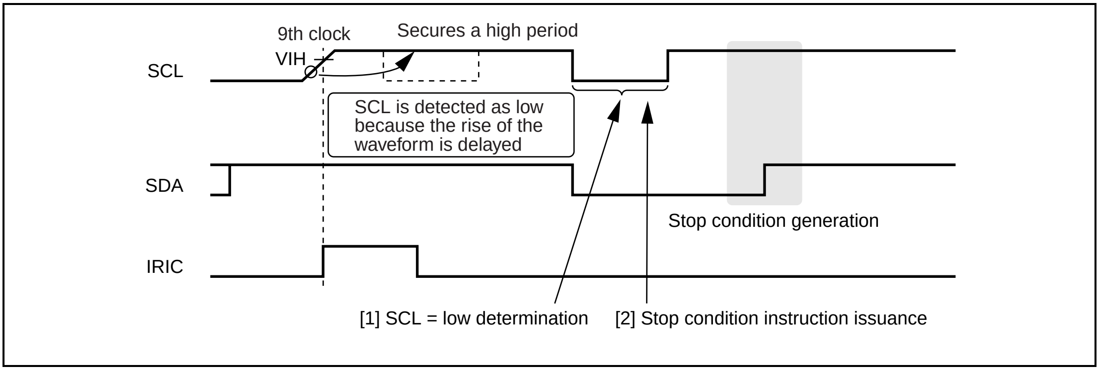

9th clock
Secures a high period
VIH
SCL
SCL is detected as low because the rise of the waveform is delayed
SDA
Stop condition generation
IRIC
[1] SCL = low determination
[2] Stop condition instruction issuance

Figure 13.31 Stop Condition Issuance Timing

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to 1 in ICXR.

10. Note on IRIC flag clear when the wait function is used

If the rise time of SCL exceeds the stipulated value or a slave device in which a wait can be inserted by driving the SCL pin low is used when the wait function is used in I^{2}C bust interface master mode, the IRIC flag should be cleared after determining that the SCL is low, as described below.

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.

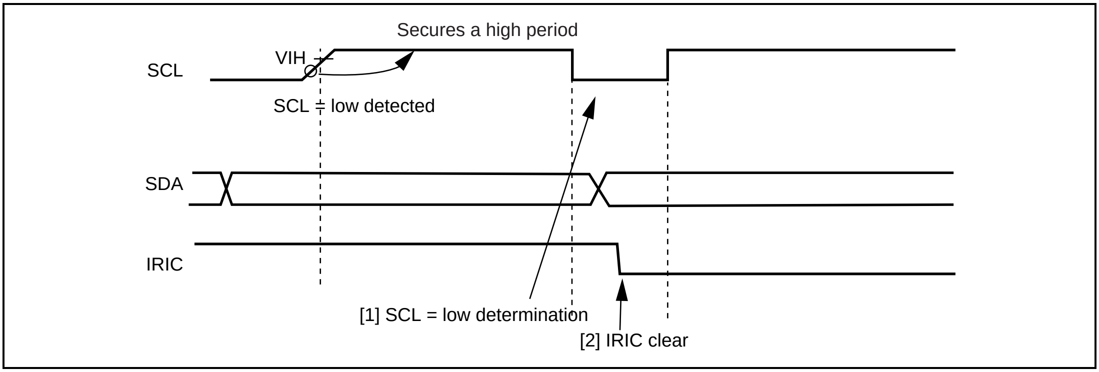


Figure 13.32 IRIC Flag Clearing Timing when WAIT = 1

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to 1 in ICXR.

11. Note on ICDR read and ICCR access in slave transmit mode

In I^{2}C bus interface slave transmit mode, do not read ICDR or do not read/write from/to ICCR during the time shaded in figure 13.33. 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/write from/to ICCR before starting the receive operation of the next slave address.

— Monitor the BC2 to BC0 bit 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 ICDR or read/write from/to ICCR during the time other than the shaded time.

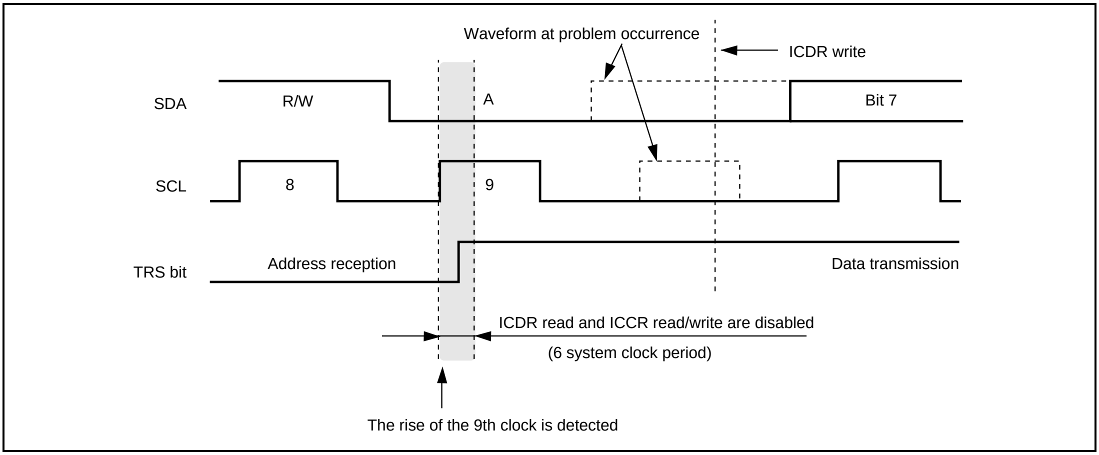

Waveform at problem occurrence
ICDR write
SDA
R/W
A
Bit 7
SCL
8
9
TRS bit
Address reception
Data transmission
ICDR read and ICCR read/write are disabled
(6 system clock period)
The rise of the 9th clock is detected

Figure 13.33 ICDR Read and ICCR Access Timing in Slave Transmit Mode

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to 1 in ICXR.

12. Note on TRS bit setting in slave mode

In I^{2}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 13.34), 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 13.34), 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 13.34. 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.

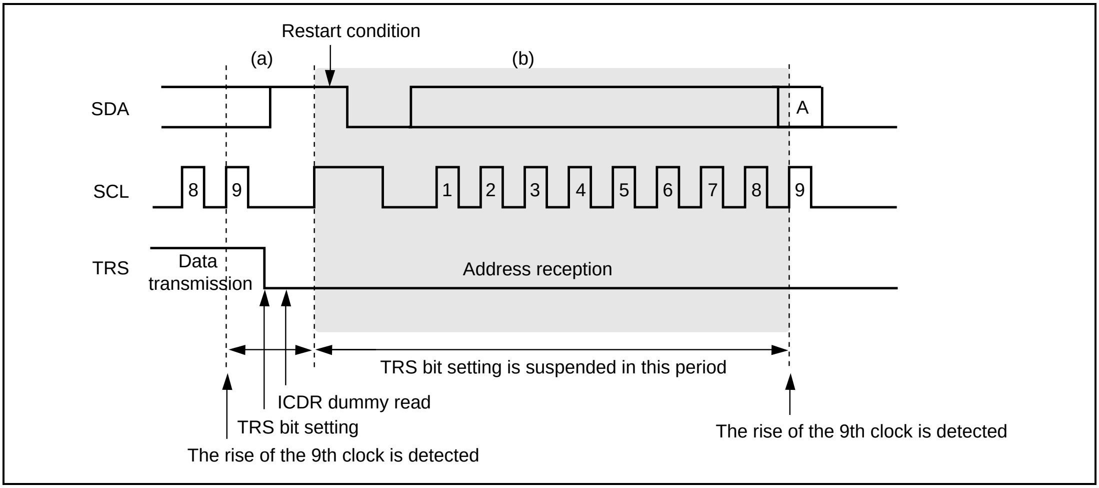

Restart condition
(a)
(b)
SDA
A
SCL
8
9
1
2
3
4
5
6
7
8
9
TRS
Data transmission
Address reception
TRS bit setting is suspended in this period
ICDR dummy read
TRS bit setting
The rise of the 9th clock is detected
The rise of the 9th clock is detected

Figure 13.34 TRS Bit Set Timing in Slave Mode

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to 1 in ICXR.

13. Note on ICDR read in transmit mode and ICDR write in receive mode

    If 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 ICDR after setting receive mode or write to ICDR after setting transmit mode.

14. Note on ACKE and TRS bits in slave mode

    In the I^2C 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), the IRIC flag may be set after the ICDRE flag is set and 1 received as the acknowledge bit value (ACKB = 1), thus causing an interrupt source even when the address does not match.

    To use the I^2C bus interface module in slave mode, be sure to follow the procedures below.

    A. 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 once to initialize the ACKB bit to 0.

    B. Set receive mode (TRS = 0) before the next start condition is input in slave mode. Complete transmit operation by the procedure shown in figure 13.23, in order to switch from slave transmit mode to slave receive mode.

13.6.1 Module Stop Mode Setting

The IIC operation can be enabled or disabled using the module stop control register. The initial setting is for the IIC operation to be halted. Register access is enabled by canceling module stop mode. For details, refer to section 19, Power-Down Modes.

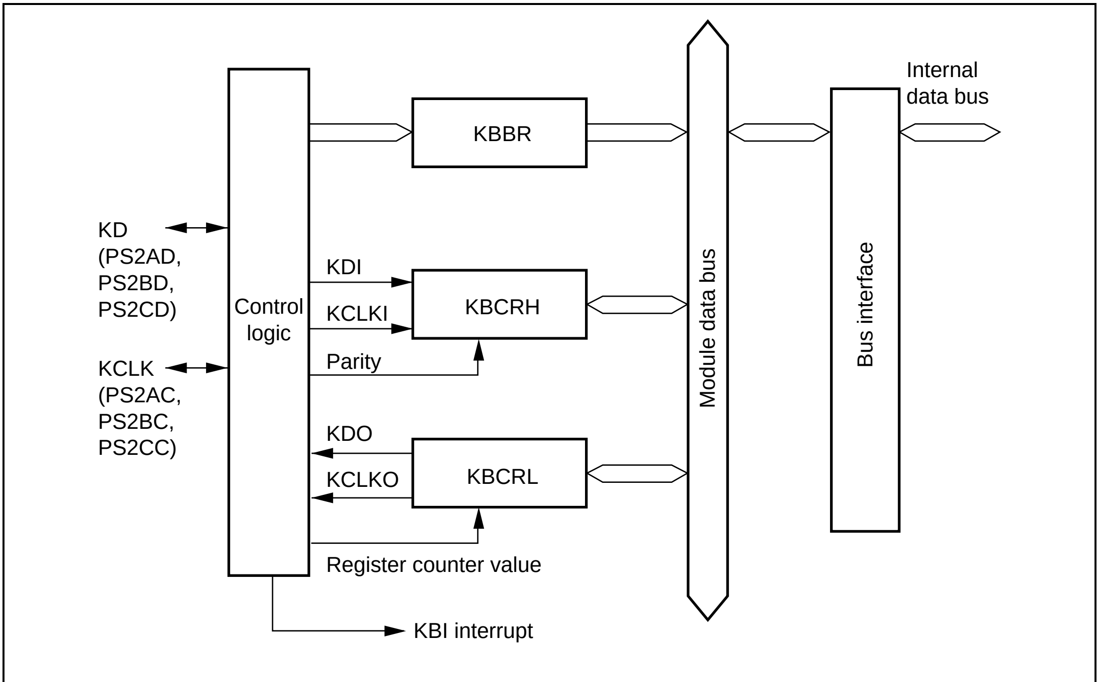
Section 14 Keyboard Buffer Controller

This LSI has three on-chip keyboard buffer controller channels. The keyboard buffer controller is provided with functions conforming to the PS/2 interface specifications.

Data transfer using the keyboard buffer controller employs a data line (KD) and a clock line (KCLK), providing economical use of connectors, board surface area, etc. Figure 14.1 shows a block diagram of the keyboard buffer controller.

14.1 Features

- Conforms to PS/2 interface specifications
- Direct bus drive (via the KCLK and KD pins)
- Interrupt sources: on completion of data reception and on detection of clock edge
- Error detection: parity error and stop bit monitoring



Legend	
KD:	KBC data I/O pin
KCLK:	KBC clock I/O pin
KBBR:	Keyboard data buffer register
KBCRH:	Keyboard control register H
KBCRL:	Keyboard control register L

Figure 14.1 Block Diagram of Keyboard Buffer Controller

Figure 14.2 shows how the keyboard buffer controller is connected.

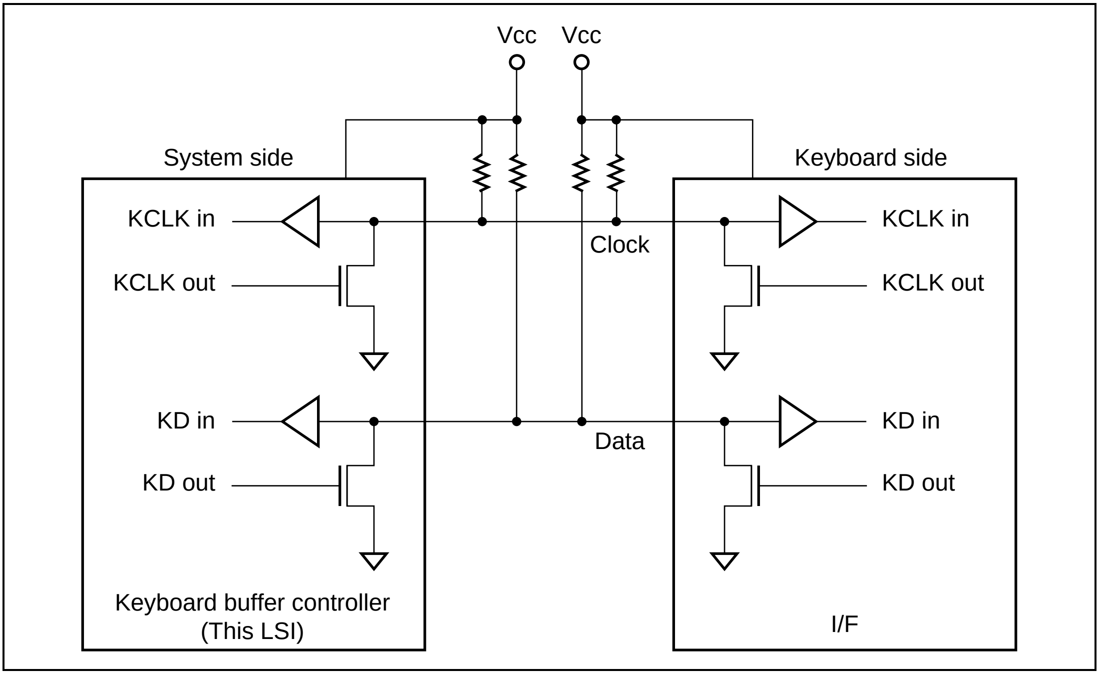


Figure 14.2 Keyboard Buffer Controller Connection

14.2 Input/Output Pins

Table 14.1 lists the input/output pins used by the keyboard buffer controller.

Table 14.1 Pin Configuration

Channel	Name	Abbreviation*	I/O	Function
0	KBC clock I/O pin (KCLK0)	PS2AC	I/O	KBC clock input/output
	KBC data I/O pin (KD0)	PS2AD	I/O	KBC data input/output
1	KBC clock I/O pin (KCLK1)	PS2BC	I/O	KBC clock input/output
	KBC data I/O pin (KD1)	PS2BD	I/O	KBC data input/output
2	KBC clock I/O pin (KCLK2)	PS2CC	I/O	KBC clock input/output
	KBC data I/O pin (KD2)	PS2CD	I/O	KBC data input/output

Note:* These are the external I/O pin names. In the text, clock I/O pins are referred to as KCLK and data I/O pins as KD, omitting the channel designations.

14.3 Register Descriptions

The keyboard buffer controller has the following registers for each channel.

- Keyboard control register H (KBCRH)
- Keyboard control register L (KBCRL)
- Keyboard data buffer register (KBBR)

14.3.1 Keyboard Control Register H (KBCRH)

KBCRH indicates the operating status of the keyboard buffer controller.

Bit	Bit Name	Initial Value	R/W	Description
7	KBIOE	0	R/W	Keyboard In/Out Enable Selects whether or not the keyboard buffer controller is used. 0: The keyboard buffer controller is non-operational (KCLK and KD signal pins have port functions) 1: The keyboard buffer controller is enabled for transmission and reception (KCLK and KD signal pins are in the bus drive state)
6	KCLKI	1	R	Keyboard Clock In Monitors the KCLK I/O pin. This bit cannot be modified. 0: KCLK I/O pin is low 1: KCLK I/O pin is high
5	KDI	1	R	Keyboard Data In: Monitors the KDI I/O pin. This bit cannot be modified. 0: KD I/O pin is low 1: KD I/O pin is high
4	KBFSEL	1	R/W	Keyboard Buffer Register Full Select Selects whether the KBF bit is used as the keyboard buffer register full flag or as the KCLK fall interrupt flag. When KBFSEL is cleared to 0, the KBE bit in KBCRL should be cleared to 0 to disable reception. 0: KBF bit is used as KCLK fall interrupt flag 1: KBF bit is used as keyboard buffer register full flag

Bit	Bit Name	Initial Value	R/W	Description
3	KBIE	0	R/W	Keyboard Interrupt Enable Enables or disables interrupts from the keyboard buffer controller to the CPU. 0: Interrupt requests are disabled 1: Interrupt requests are enabled
2	KBF	0	R/(W)*	Keyboard Buffer Register Full Indicates that data reception has been completed and the received data is in KBBR. 0: [Clearing condition] Read KBF when KBF =1, then write 0 in KBF 1: [Setting conditions] • When data has been received normally and has been transferred to KBBR while KBFSEL = 1 (keyboard buffer register full flag) • When a KCLK falling edge is detected while KBFSEL = 0 (KCLK interrupt flag)
1	PER	0	R/(W)*	Parity Error Indicates that an odd parity error has occurred. 0: [Clearing condition] Read PER when PER =1, then write 0 in PER 1: [Setting condition] When an odd parity error occurs
0	KBS	0	R	Keyboard Stop Indicates the receive data stop bit. Valid only when KBF = 1. 0: 0 stop bit received 1: 1 stop bit received

Note:* Only 0 can be written for clearing the flag.

14.3.2 Keyboard Control Register L (KBCRL)

KBCRL enables the receive counter count and controls the keyboard buffer controller pin output.

Bit	Bit Name	Initial Value	R/W	Description
7	KBE	0	R/W	Keyboard Enable Enables or disables loading of receive data into KBBR. 0: Loading of receive data into KBBR is disabled 1: Loading of receive data into KBBR is enabled
6	KCLKO	1	R/W	Keyboard Clock Out Controls KBC clock I/O pin output. 0: KBC clock I/O pin is low 1: KBC clock I/O pin is high
5	KDO	1	R/W	Keyboard Data Out Controls KBC data I/O pin output. 0: KBC data I/O pin is low 1: KBC data I/O pin is high
4	—	1	—	Reserved This bit is always read as 1 and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
3	RXCR3	0	R	Receive Counter These bits indicate the received data bit. Their value is incremented on the fall of KCLK. These bits cannot be modified. The receive counter is initialized to 0000 by a reset and when 0 is written in KBE. Its value returns to 0000 after a stop bit is received. 0000: — 0001: Start bit 0010: KB0 0011: KB1 0100: KB2 0101: KB3 0110: KB4 0111: KB5 1000: KB6 1001: KB7 1010: Parity bit 1011: — 11- - : —
2	RXCR2	0	R	
1	RXCR1	0	R	
0	RXCR0	0	R	

14.3.3 Keyboard Data Buffer Register (KBBR)

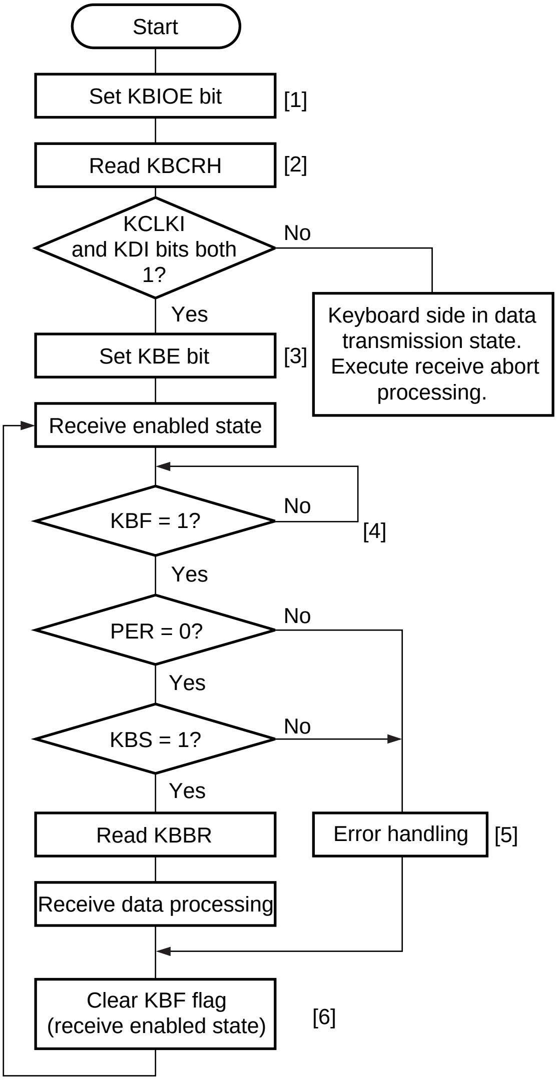
KBBR stores receive data. Its value is valid only when KBF = 1.

Bit	Bit Name	Initial Value	R/W	Description
7	KB7	0	R	Keyboard Data 7 to 0 8-bit read only data. Initialized to H'00 by a reset, in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode, and when KBIOE is cleared to 0.
6	KB6	0	R	
5	KB5	0	R	
4	KB4	0	R	
3	KB3	0	R	
2	KB2	0	R	
1	KB1	0	R	
0	KB0	0	R	

14.4 Operation

14.4.1 Receive Operation

In a receive operation, both KCLK (clock) and KD (data) are outputs on the keyboard side and inputs on this LSI chip (system) side. KD receives a start bit, 8 data bits (LSB-first), an odd parity bit, and a stop bit, in that order. The KD value is valid when KCLK is low. A sample receive processing flowchart is shown in figure 14.3, and the receive timing in figure 14.4.



[1] Set the KBIOE bit to 1 in KBCRL.

[2] Read KBCRH, and if the KCLKI and KDI bits are both 1, set the KBE bit (receive enabled state).

[3] Detect the start bit output on the keyboard side and receive data in synchronization with the fall of KCLK.

[4] When a stop bit is received, the keyboard buffer controller drives KCLK low to disable keyboard transmission (automatic I/O inhibit).
If the KBIE bit is set to 1 in KBCRH, an interrupt request is sent to the CPU at the same time.

[5] Perform receive data processing.

[6] Clear the KBF flag to 0 in KBCRL. At the same time, the system automatically drives KCLK high, setting the receive enabled state.

The receive operation can be continued by repeating steps [3] to [6].

Figure 14.3 Sample Receive Processing Flowchart

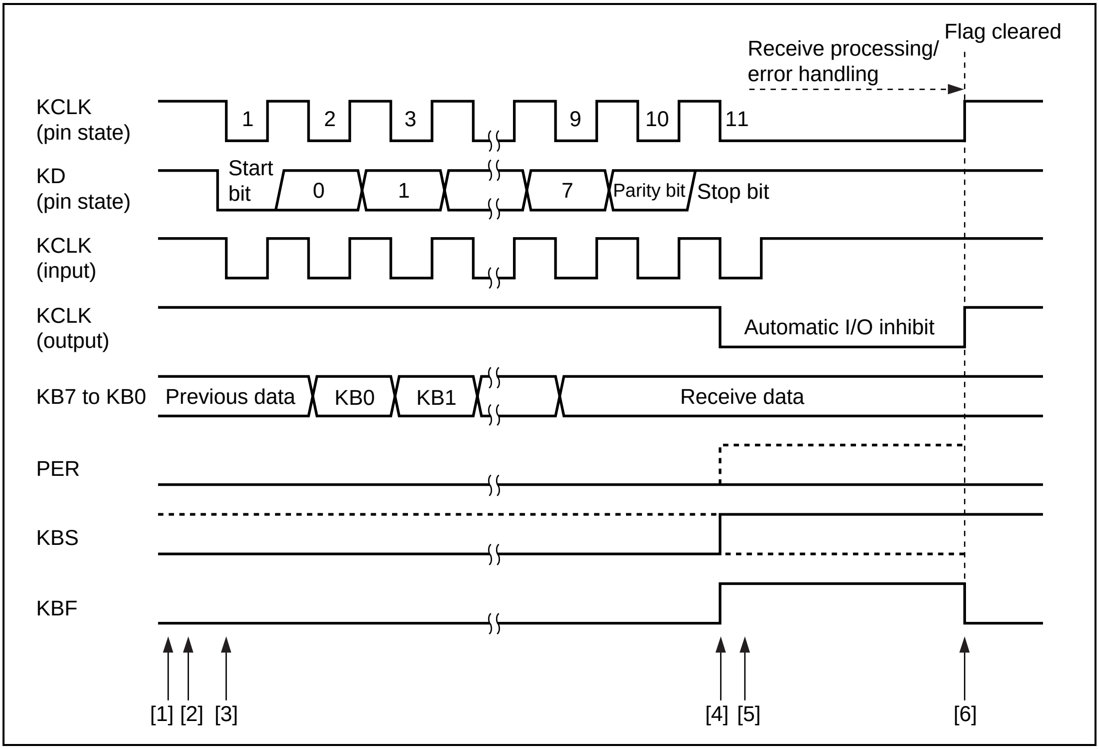
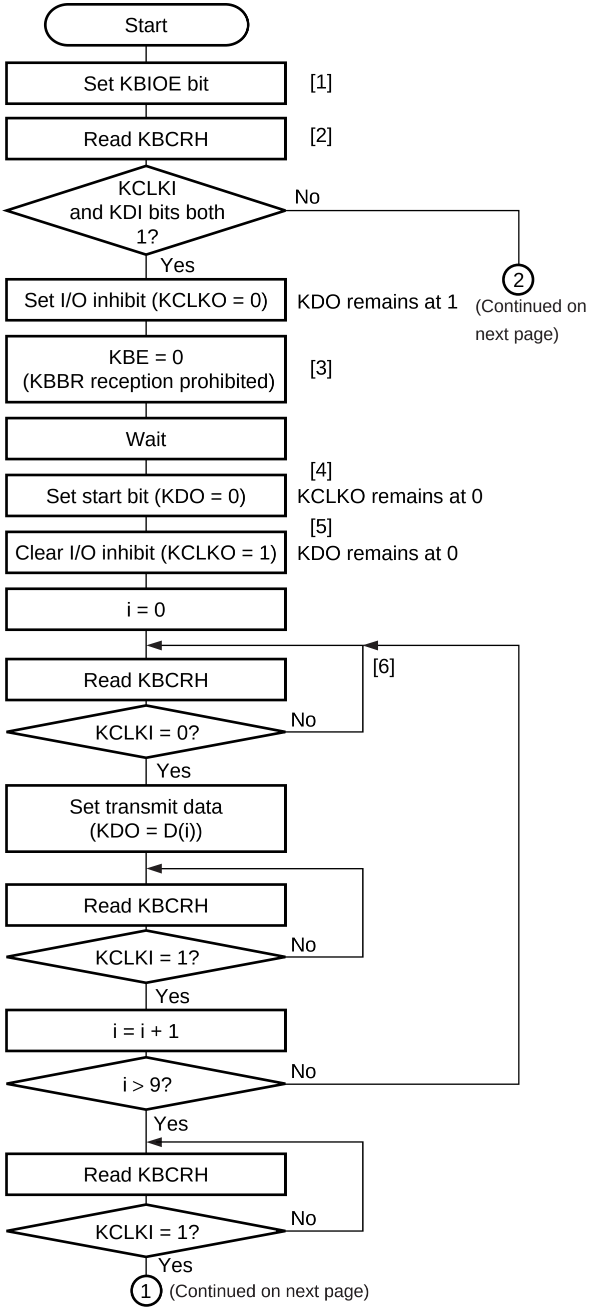


Figure 14.4 Receive Timing

14.4.2 Transmit Operation

In a transmit operation, KCLK (clock) is an output on the keyboard side, and KD (data) is an output on the chip (system) side. KD outputs a start bit, 8 data bits (LSB-first), an odd parity bit, and a stop bit, in that order. The KD value is valid when KCLK is high. A sample transmit processing flowchart is shown in figure 14.5, and the transmit timing in figure 14.6.



[1] Set the KBE bit to 1 in KBCRH.

[2] Read KBCRH, and if the KCLKI and KDI bits are both 1, write 0 in the KCLKO bit (set I/O inhibit).

[3] Write 0 in the KBE bit (prohibit KBBR receive operation).

[4] Write 0 in the KDO bit (set start bit).

[5] Write 1 in the KCLKO bit (clear I/O inhibit).

[6] Read KBCRH, and when KCLKI = 0, set the transmit data in the KDO bit (LSB-first). Next, set the parity bit and stop bit in the KDO bit.

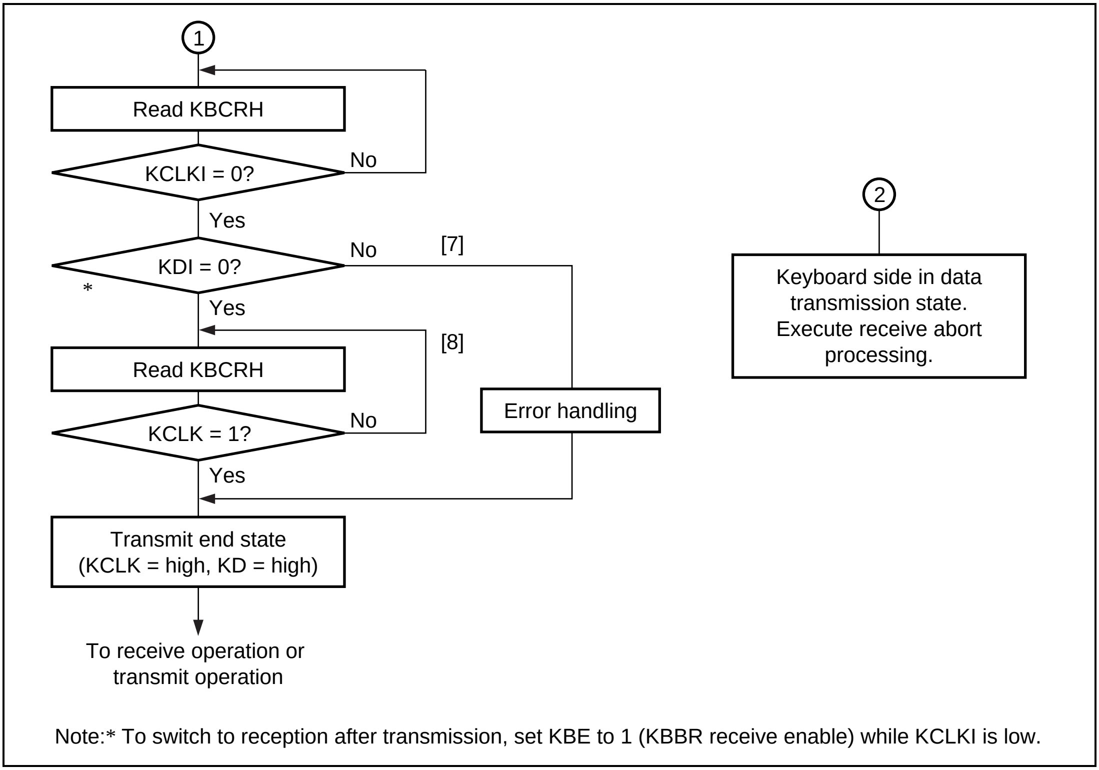
[7] After transmitting the stop bit, read KBCRL and confirm that KDI = 0 (receive completed notification from the keyboard).

[8] Read KBCRH. Confirm that the KCLKI and KDI bits are both 1.

The transmit operation can be continued by repeating steps [2] to [8].

i = 0 to 7: Transmit data
i = 8: Parity bit
i = 9: Stop bit

Figure 14.5 (1) Sample Transmit Processing Flowchart



Note:* To switch to reception after transmission, set KBE to 1 (KBBR receive enable) while KCLKI is low.

Figure 14.5 (2) Sample Transmit Processing Flowchart

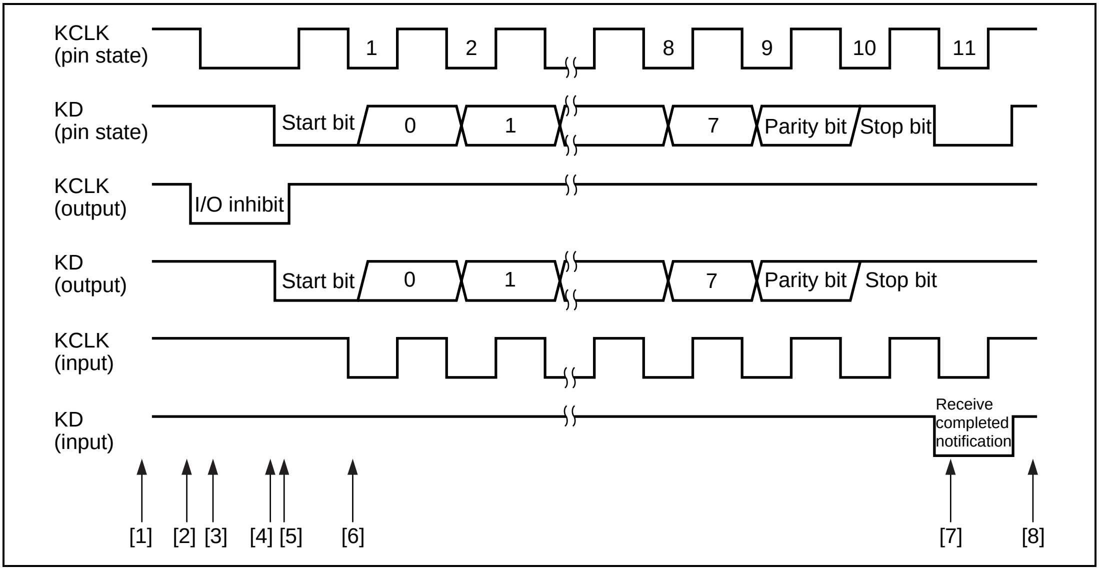
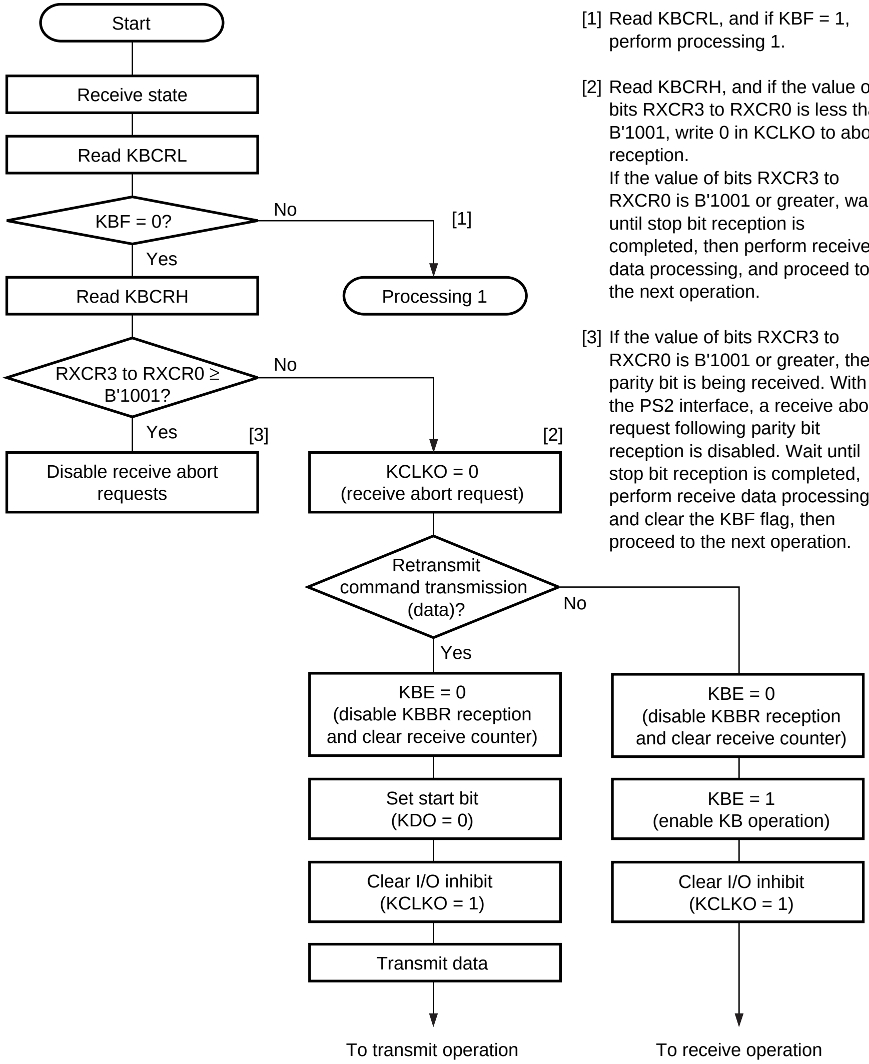


Figure 14.6 Transmit Timing

14.4.3 Receive Abort

This LSI (system side) can forcibly abort transmission from the device connected to it (keyboard side) in the event of a protocol error, etc. In this case, the system holds the clock low. During reception, the keyboard also outputs a clock for synchronization, and the clock is monitored when the keyboard output clock is high. If the clock is low at this time, the keyboard judges that there is an abort request from the system, and data transmission from the keyboard is aborted. Thus the system can abort reception by holding the clock low for a certain period. A sample receive abort processing flowchart is shown in figure 14.7, and the receive abort timing in figure 14.8.


Start
Receive state
Read KBCRL
KBF = 0?
No
[1]
Yes
Read KBCRH
Processing 1
RXCR3 to RXCR0 ≥ B'1001?
No
Yes
[3]
[2]
Disable receive abort requests
KCLKO = 0 (receive abort request)
Retransmit command transmission (data)?
No
Yes
KBE = 0 (disable KBBR reception and clear receive counter)
KBE = 0 (disable KBBR reception and clear receive counter)
Set start bit (KDO = 0)
KBE = 1 (enable KB operation)
Clear I/O inhibit (KCLKO = 1)
Clear I/O inhibit (KCLKO = 1)
Transmit data
To transmit operation
To receive operation

[1] Read KBCRL, and if KBF = 1, perform processing 1.

[2] Read KBCRH, and if the value of bits RXCR3 to RXCR0 is less than B'1001, write 0 in KCLKO to abort reception.
If the value of bits RXCR3 to RXCR0 is B'1001 or greater, wait until stop bit reception is completed, then perform receive data processing, and proceed to the next operation.

[3] If the value of bits RXCR3 to RXCR0 is B'1001 or greater, the parity bit is being received. With the PS2 interface, a receive abort request following parity bit reception is disabled. Wait until stop bit reception is completed, perform receive data processing and clear the KBF flag, then proceed to the next operation.

Figure 14.7 (1) Sample Receive Abort Processing Flowchart

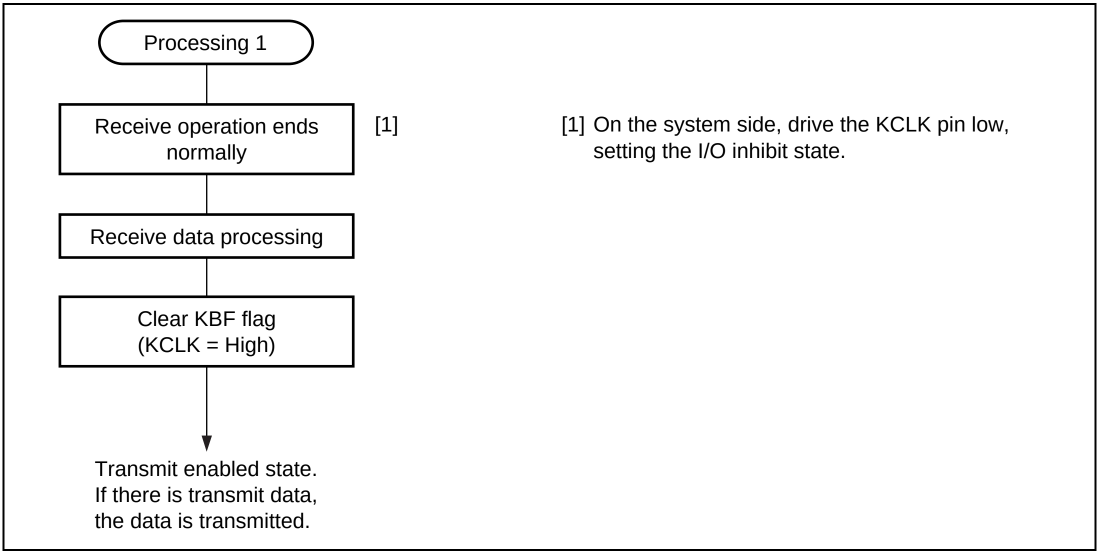

Processing 1
Receive operation ends normally
[1]
Receive data processing
Clear KBF flag (KCLK = High)
Transmit enabled state. If there is transmit data, the data is transmitted.
[1] On the system side, drive the KCLK pin low, setting the I/O inhibit state.

Figure 14.7 (2) Sample Receive Abort Processing Flowchart

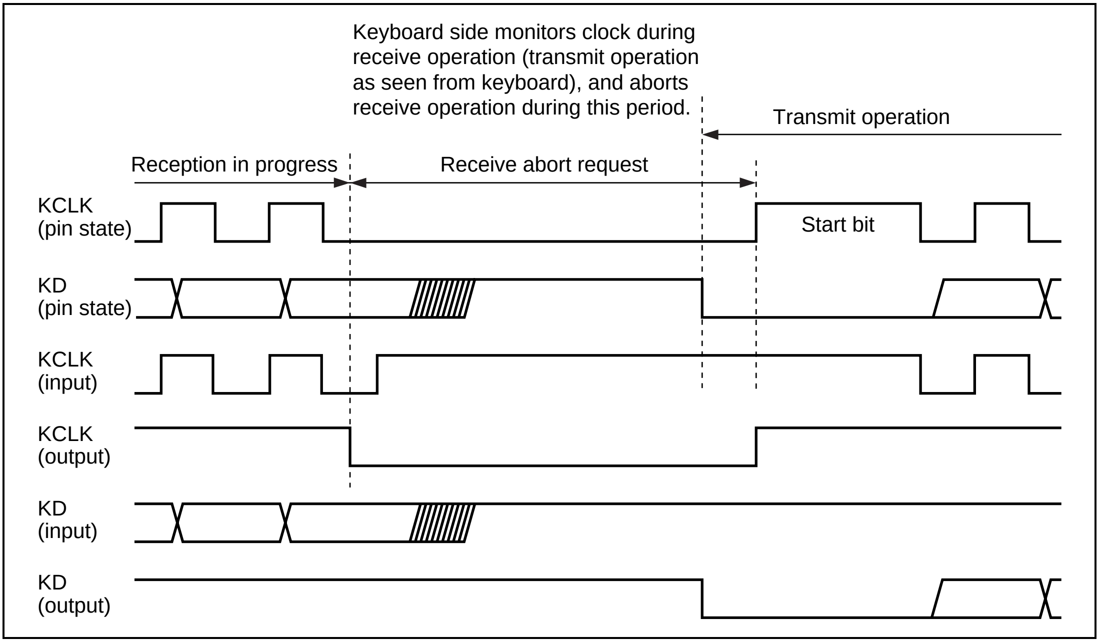
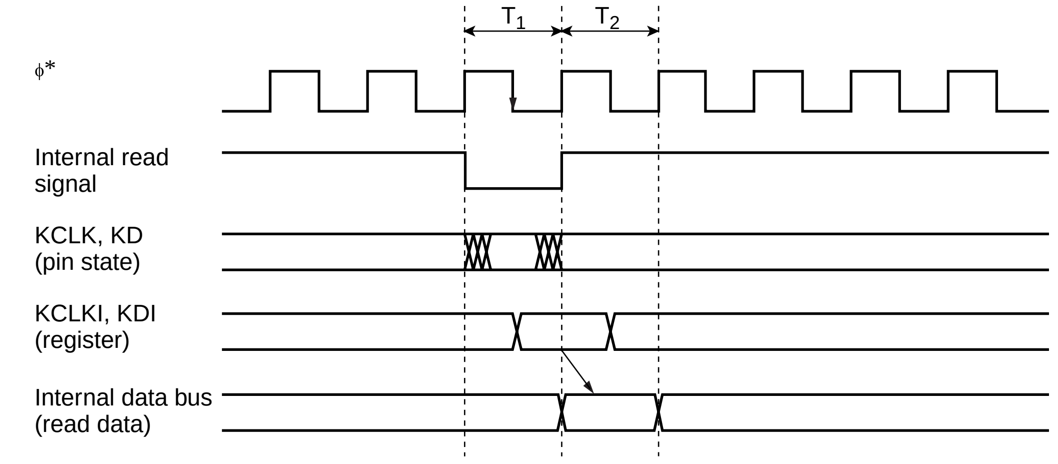

Keyboard side monitors clock during receive operation (transmit operation as seen from keyboard), and aborts receive operation during this period.
Transmit operation
Reception in progress
Receive abort request
KCLK (pin state)
Start bit
KD (pin state)
KCLK (input)
KCLK (output)
KD (input)
KD (output)

Figure 14.8 Receive Abort and Transmit Start (Transmission/Reception Switchover) Timing

14.4.4 KCLKI and KDI Read Timing

Figure 14.9 shows the KCLKI and KDI read timing.

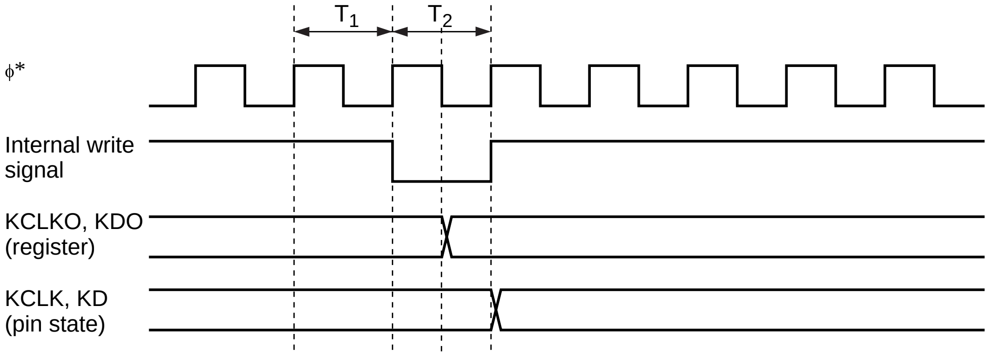


Note:* The ϕ clock shown here is scaled by 1/N in medium-speed mode when the operating mode is active mode.

Figure 14.9 KCLKI and KDI Read Timing

14.4.5 KCLKO and KDO Write Timing

Figure 14.10 shows the KLCKO and KDO write timing and the KCLK and KD pin states.

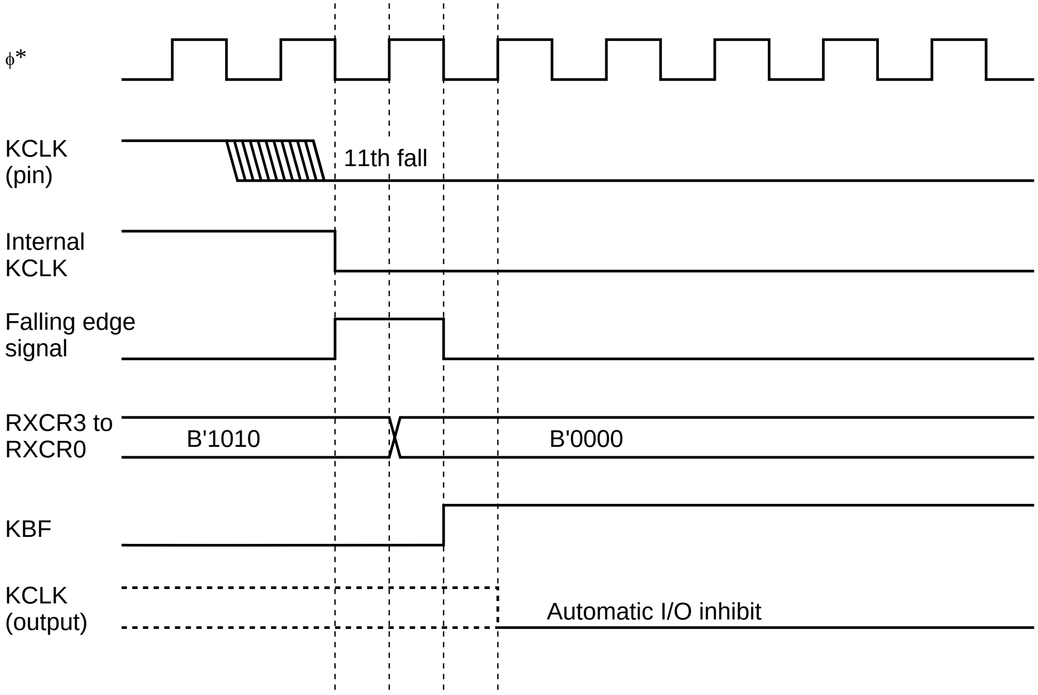


Note:* The ϕ clock shown here is scaled by 1/N in medium-speed mode when the operating mode is active mode.

Figure 14.10 KCLKO and KDO Write Timing

14.4.6 KBF Setting Timing and KCLK Control

Figure 14.11 shows the KBF setting timing and the KCLK pin states.

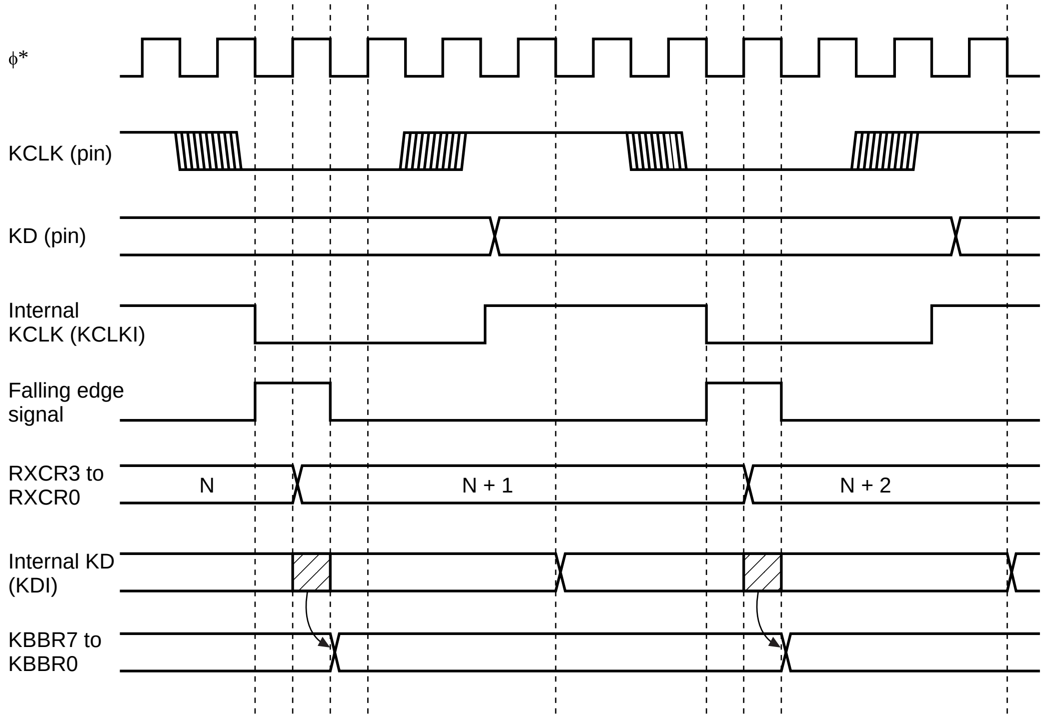


Note:* The φ clock shown here is scaled by 1/N in medium-speed mode when the operating mode is active mode.

Figure 14.11 KBF Setting and KCLK Automatic I/O Inhibit Generation Timing

14.4.7 Receive Timing

Figure 14.12 shows the receive timing.



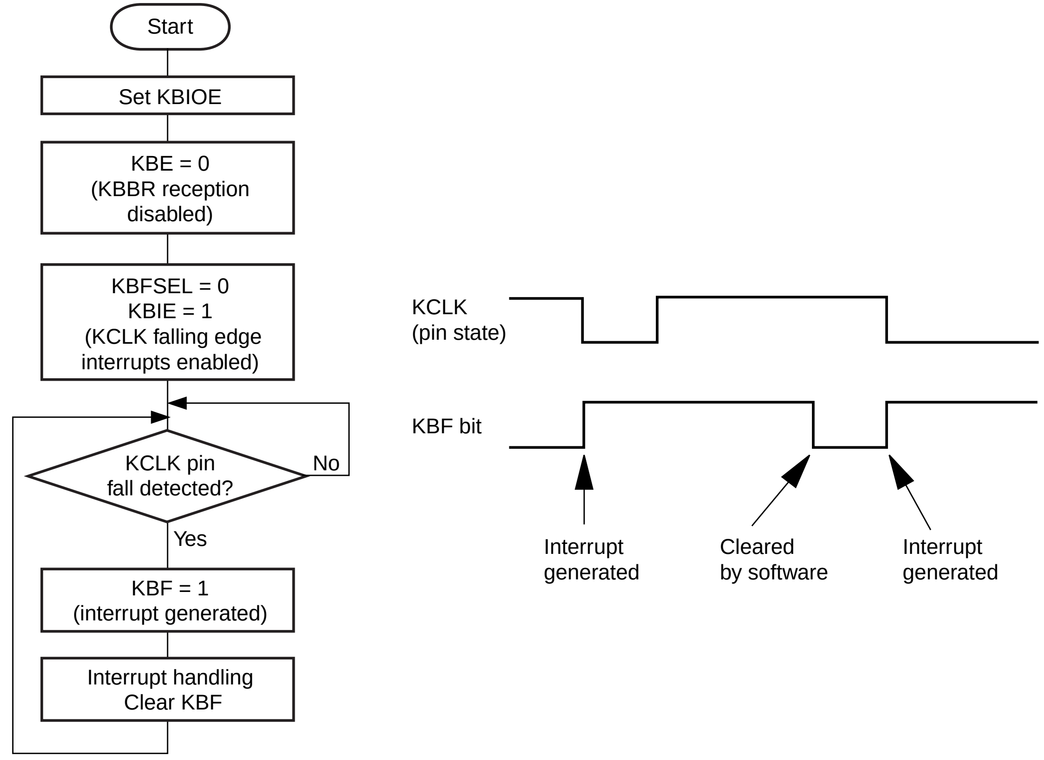
Note:* The ϕ clock shown here is scaled by 1/N in medium-speed mode when the operating mode is active mode.

Figure 14.12 Receive Counter and KBBR Data Load Timing

14.4.8 KCLK Fall Interrupt Operation

In this device, clearing the KBFSEL bit to 0 in KBCRH enables the KBF bit in KBCRL to be used as a flag for the interrupt generated by the fall of KCLK input.

Figure 14.13 shows the setting method and an example of operation.



Note:* The KBF setting timing is the same as the timing of KBF setting and KCLK automatic I/O inhibit bit generation in figure 14.11. When the KBF bit is used as the KCLK input fall interrupt flag, the automatic I/O inhibit function does not operate.

Figure 14.13 Example of KCLK Input Fall Interrupt Operation

14.5 Usage Notes

14.5.1 KBIOE Setting and KCLK Falling Edge Detection

When KBIOE is 0, the internal KCLK and internal KD settings are fixed at 1. Therefore, if the KCLK pin is low when the KBIOE bit is set to 1, the edge detection circuit operates and the KCLK falling edge is detected.

If the KBFSEL bit and KBE bit are both 0 at this time, the KBF bit is set. Figure 14.14 shows the timing of KBIOE setting and KCLK falling edge detection.

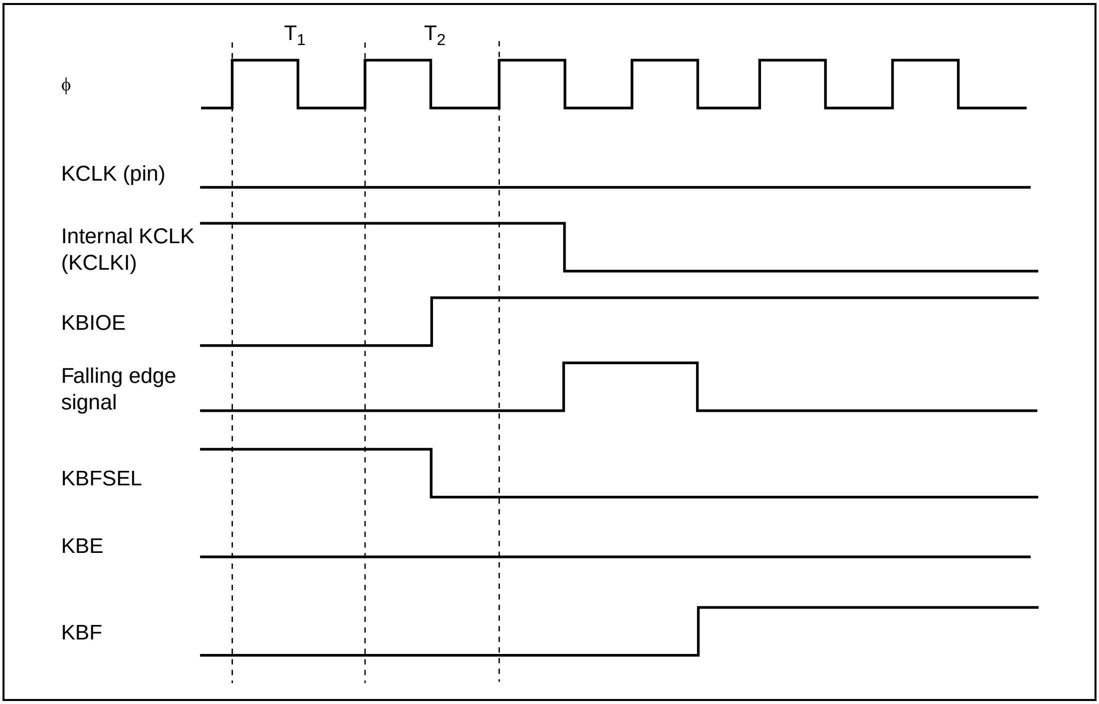


Figure 14.14 KBIOE Setting and KCLK Falling Edge Detection Timing

14.5.2 Module Stop Mode Setting

Keyboard buffer controller operation can be enabled or disabled using the module stop control register. The initial setting is for keyboard buffer controller operation to be halted. Register access is enabled by canceling module stop mode. For details, refer to section 19, Power-Down Modes.

Section 15 Host Interface LPC Interface (LPC)

This LSI has an on-chip LPC interface.

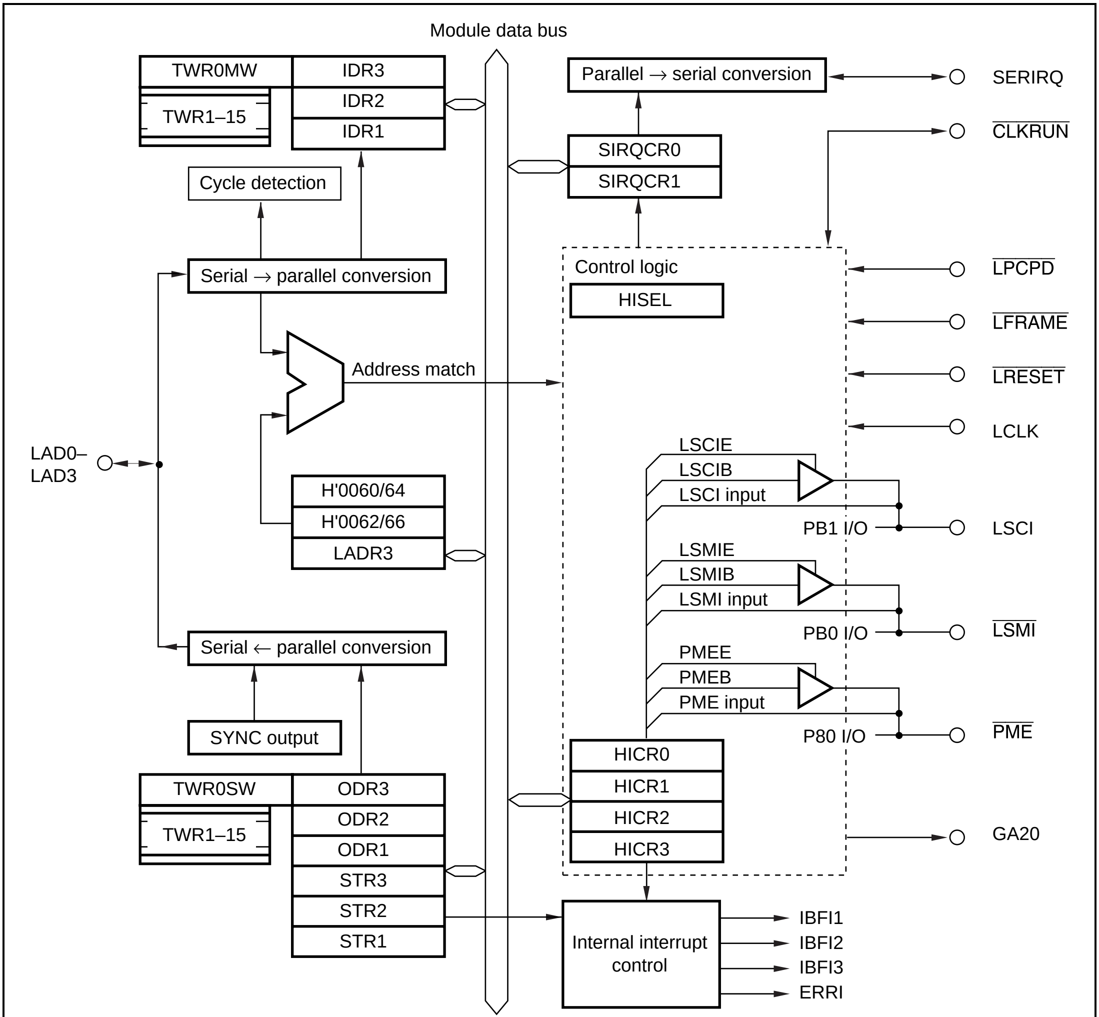
The LPC performs serial transfer of cycle type, address, and data, synchronized with the 33-MHz PCI clock. It uses four signal lines for address/data, and one for host interrupt requests. This LPC module supports only I/O read cycle and I/O write cycle transfers.

It is also provided with power-down functions that can control the PCI clock and shut down the host interface.

15.1 Features

- Supports LPC interface I/O read cycles and I/O write cycles
  - Uses four signal lines (LAD3 to LAD0) to transfer the cycle type, address, and data.
  - Uses three control signals: clock (LCLK), reset ($\overline{\mathrm{LRESET}}$), and frame ($\overline{\mathrm{LFRAME}}$).
- Has three register sets comprising data and status registers
  - The basic register set comprises three bytes: an input register (IDR), output register (ODR), and status register (STR).
  - Channels 1 and 2 have fixed I/O addresses of H'60/H'64 and H'62/H'66, respectively. A fast A20 gate function is also provided.
  - The I/O address can be set for channel 3. Sixteen bidirectional data register bytes can be manipulated in addition to the basic register set.
- Supports SERIRQ
  - Host interrupt requests are transferred serially on a single signal line (SERIRQ).
  - On channel 1, HIRQ1 and HIRQ12 can be generated.
  - On channels 2 and 3, SMI, HIRQ6, and HIRQ9 to HIRQ11 can be generated.
  - Operation can be switched between quiet mode and continuous mode.
  - The $\overline{\mathrm{CLKRUN}}$ signal can be manipulated to restart the PCI clock (LCLK).
- Eleven interrupt sources
  - The LPC module can be shut down by inputting the $\overline{\mathrm{LPCPD}}$ signal.
  - Three pins, $\overline{\mathrm{PME}}$, $\overline{\mathrm{LSMI}}$, and LSCI, are provided for general input/output.

Figure 15.1 shows a block diagram of the LPC.



Legend
HICR0 to HICR3: Host interface control registers 0 to 3
LADR3H, 3L: LPC channel 3 address register 3H and 3L
IDR1 to IDR3: Input data registers 1 to 3
ODR1 to DOR3: Output data registers 1 to 3
STR1 to STR3: Status registers 1 to 3
TWR0MW: Two-way register 0MW
TWR0SW: Two-way register 0SW
TWR1 to TWR15: Two-way data registers 1 to 15
SERIRQ0, 1: SERIEQ control registers 0 and 1
HISEL: Host interface select register

Figure 15.1 Block Diagram of LPC

15.2 Input/Output Pins

Table 15.1 lists the input and output pins of the LPC module.

Table 15.1 Pin Configuration

Name	Abbreviation	Port	I/O	Function
LPC address/ data 3 to 0	LAD3 to LAD0	P33 to P30	Input/ output	Serial (4-signal-line) transfer cycle type/address/data signals, synchronized with LCLK
LPC frame	$\overline{\text{LFRAME}}$	P34	Input*[1]	Transfer cycle start and forced termination signal
LPC reset	$\overline{\text{LRESET}}$	P35	Input*[1]	LPC interface reset signal
LPC clock	LCLK	P36	Input	33 MHz PCI clock signal
Serialized interrupt request	SERIRQ	P37	Input/ output*[1]	Serialized host interrupt request signal, synchronized with LCLK (SMI, IRQ1, IRQ6, IRQ9 to IRQ12)
LSCI general output	LSCI	PB1	Output*[1], *[2]	General output
LSMI general output	$\overline{\text{LSMI}}$	PB0	Output*[1], *[2]	General output
PME general output	$\overline{\text{PME}}$	P80	Output*[1], *[2]	General output
GATE A20	GA20	P81	Output*[1], *[2]	A20 gate control signal output
LPC clock run	$\overline{\text{CLKRUN}}$	P82	Input/ output*[1], *[2]	LCLK restart request signal in case of serial host interrupt request
LPC power-down	$\overline{\text{LPCPD}}$	P83	Input*[1]	LPC module shutdown signal

Notes:
1. Pin state monitoring input is possible in addition to the LPC interface control input/output function.
2. Only 0 can be output. If 1 is output, the pin goes to the high-impedance state, so an external resistor is necessary to pull the signal up to V_{CC}.

15.3 Register Descriptions

The LPC has the following registers. The settings of the HI12E bit in SYSCR2 do not affect the operation of the LPC. For reasons relating to the configuration of the program development tool (emulator), when the LPC is used, the HI12E bit in SYSCR2 should not be set to 1. For details, see section 3.2.2, System Control Register (SYSCR), and section 7.7.4, System Control Register 2 (SYSCR2).

- Host interface control register 0 (HICR0)
- Host interface control register 1 (HICR1)
- Host interface control register 2 (HICR2)
- Host interface control register 3 (HICR3)
- LPC channel 3 address registers (LADR3H, LADR3L)
- Input data register 1 (IDR1)
- Output data register 1 (ODR1)
- Status register 1 (STR1)
- Input data register 2 (IDR2)
- Output data register 2 (ODR2)
- Status register 2 (STR2)
- Input data register 3 (IDR3)
- Output data register 3 (ODR3)
- Status register 3 (STR3)
- Bidirectional data registers 0 to 15 (TWR0 to TWR15)
- SERIRQ control register 0 (SIRQCR0)
- SERIRQ control register 1 (SIRQCR1)
- Host interface select register (HISEL)

15.3.1 Host Interface Control Registers 0 and 1 (HICR0, HICR1)

HICR0 and HICR1 contain control bits that enable or disable host interface functions, control bits that determine pin output and the internal state of the host interface, and status flags that monitor the internal state of the host interface.

- HICR0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	LPC3E	0	R/W	—	LPC Enable 3 to 1 Enable or disable the host interface function in single-chip mode. When the host interface is enabled (one of the three bits is set to 1), processing for data transfer between the slave processor (this LSI) and the host processor is performed using pins LAD3 to LAD0, $\overline{\text{LFRAME}}$, $\overline{\text{LRESET}}$, LCLK, SERIRQ, $\overline{\text{CLKRUN}}$, and $\overline{\text{LPCPD}}$. • LPC3E 0: LPC channel 3 operation is disabled No address (LADR3) matches for IDR3, ODR3, STR3, or TWR0 to TWR15 1: LPC channel 3 operation is enabled • LPC2E 0: LPC channel 2 operation is disabled No address (H'0062, 66) matches for IDR2, ODR2, or STR2 1: LPC channel 2 operation is enabled • LPC1E 0: LPC channel 1 operation is disabled No address (H'0060, 64) matches for IDR1, ODR1, or STR1 1: LPC channel 1 operation is enabled
6	LPC2E	0	R/W	—	
5	LPC1E	0	R/W	—	

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
4	FGA20E	0	R/W	—	Fast A20 Gate Function Enable Enables or disables the fast A20 gate function. When the fast A20 gate is disabled, the normal A20 gate can be implemented by firmware operation of the P81 output. When the fast A20 gate function is enabled, the DDR bit for P81 must not be set to 1. 0: Fast A20 gate function disabled • Other function of pin P81 is enabled • GA20 output internal state is initialized to 1 1: Fast A20 gate function enabled • GA20 pin output is open-drain (external VCC pull-up resistor required)
3	SDWNE	0	R/W	—	LPC Software Shutdown Enable Controls host interface shutdown. For details of the LPC shutdown function, and the scope of initialization by an LPC reset and an LPC shutdown, see section 15.4.4, Host Interface Shutdown Function (LPCPD). 0: Normal state, LPC software shutdown setting enabled [Clearing conditions] • Writing 0 • LPC hardware reset or LPC software reset • LPC hardware shutdown release (rising edge of $\overline{\text{LPCPD}}$ signal) 1: LPC hardware shutdown state setting enabled • Hardware shutdown state when $\overline{\text{LPCPD}}$ signal is low [Setting condition] • Writing 1 after reading SDWNE = 0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
2	PMEE	0	R/W	—	PME output Enable Controls PME output in combination with the PMEB bit in HICR1. $\overline{\text{PME}}$ pin output is open-drain, and an external pull-up resistor is needed to pull the output up to V_{CC}. When the PME output function is used, the DDR bit for P80 must not be set to 1. PMEE PMEB 0 x: PME output disabled, other function of pin is enabled 1 0: PME output enabled, $\overline{\text{PME}}$ pin output goes to 0 level 1 1: PME output enabled, $\overline{\text{PME}}$ pin output is high-impedance
1	LSMIE	0	R/W	—	LSMI output Enable Controls LSMI output in combination with the LSMIB bit in HICR1. $\overline{\text{LSMI}}$ pin output is open-drain, and an external pull-up resistor is needed to pull the output up to V_{CC}. When the LSMI output function is used, the DDR bit for PB0 must not be set to 1. LSMIE LSMIB 0 x: LSMI output disabled, other function of pin is enabled 1 0: LSMI output enabled, $\overline{\text{LSMI}}$ pin output goes to 0 level 1 1: LSMI output enabled, $\overline{\text{LSMI}}$ pin output is high-impedance

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	LSCIE	0	R/W	—	LSCI output Enable Controls LSCI output in combination with the LSCIB bit in HICR1. LSCI pin output is open-drain, and an external pull-up resistor is needed to pull the output up to V_{cc} When the LSCI output function is used, the DDR bit for PB1 must not be set to 1. LSCIE LSCIB 0 x: LSCI output disabled, other function of pin is enabled 1 0: LSCI output enabled, LSCI pin output goes to 0 level 1 1: LSCI output enabled, LSCI pin output is high-impedance

Legend

X: Don't care

- HICR1

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	LPCBSY	0	R/W	—	LPC Busy Indicates that the host interface is processing a transfer cycle. 0: Host interface is in transfer cycle wait state • Bus idle, or transfer cycle not subject to processing is in progress • Cycle type or address indeterminate during transfer cycle [Clearing conditions] • LPC hardware reset or LPC software reset • LPC hardware shutdown or LPC software shutdown • Forced termination (abort) of transfer cycle subject to processing • Normal termination of transfer cycle subject to processing 1: Host interface is performing transfer cycle processing [Setting condition] • Match of cycle type and address

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
6	CLKREQ	0	R	—	LCLK Request Indicates that the host interface's SERIRQ output is requesting a restart of LCLK. 0: No LCLK restart request [Clearing conditions] • LPC hardware reset or LPC software reset • LPC hardware shutdown or LPC software shutdown • SERIRQ is set to continuous mode • There are no further interrupts for transfer to the host in quiet mode 1: LCLK restart request issued [Setting condition] • In quiet mode, SERIRQ interrupt output becomes necessary while LCLK is stopped
5	IRQBSY	0	R	—	SERIRQ Busy Indicates that the host interface's SERIRQ signal is engaged in transfer processing. 0: SERIRQ transfer frame wait state [Clearing conditions] • LPC hardware reset or LPC software reset • LPC hardware shutdown or LPC software shutdown • End of SERIRQ transfer frame 1: SERIRQ transfer processing in progress [Setting condition] • Start of SERIRQ transfer frame

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
4	LRSTB	0	—	—	LPC Software Reset Bit Resets the host interface. For the scope of initialization by an LPC reset, see section 15.4.4, Host Interface Shutdown Function (LPCPD). 0: Normal state [Clearing conditions] • Writing 0 • LPC hardware reset 1: LPC software reset state [Setting condition] • Writing 1 after reading LRSTB = 0
3	SDWNB	0	R/W	—	LPC Software Shutdown Bit Controls host interface shutdown. For details of the LPC shutdown function, and the scope of initialization by an LPC reset and an LPC shutdown, see section 15.4.4, Host Interface Shutdown Function (LPCPD). 0: Normal state [Clearing conditions] • Writing 0 • LPC hardware reset or LPC software reset • LPC hardware shutdown • LPC hardware shutdown release (rising edge of $\overline{\text{LPCPD}}$ signal when SDWNE = 0) 1: LPC software shutdown state [Setting condition] • Writing 1 after reading SDWNB = 0
2	PMEB	0	R/W	—	PME Output Bit Controls PME output in combination with the PMEE bit. For details, refer to description on the PMEE bit in HICR0.
1	LSMIB	0	R/W	—	LSMI Output Bit Controls LSMI output in combination with the LSMIE bit. For details, refer to description on the LSMIE bit in HICR0.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	LSCIB	0	R/W	—	LSCI output Bit Controls LSCI output in combination with the LSCIE bit in HICR1. For details, refer to description on the LSCIE bit.

15.3.2 Host Interface Control Registers 2 and 3 (HICR2, HICR3)

Bits 6 to 0 in HICR2 control interrupts from the host interface (LPC) module to the slave processor (this LSI). Bit 7 in HICR2 and HICR3 monitor host interface pin states.

The pin states can be monitored regardless of the host interface operating state or the operating state of the functions that use pin multiplexing.

- HICR2

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	GA20	Undefined	R	—	GA20 Pin Monitor
6	LRST	0	R/(W)*	—	LPC Reset Interrupt Flag This bit is a flag that generates an ERRI interrupt when an LPC hardware reset occurs. 0: [Clearing conditions] • Writing 0 after reading LRST = 1 1: [Setting condition] • LRESET pin falling edge detection
5	SDWN	0	R/(W)*	—	LPC Shutdown Interrupt Flag This bit is a flag that generates an ERRI interrupt when an LPC hardware shutdown request is generated. 0: [Clearing conditions] • Writing 0 after reading SDWN = 1 • LPC hardware reset and LPC software reset 1: [Setting condition] • LPCPD pin falling edge detection

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
4	ABRT	0	R/(W)*	—	LPC Abort Interrupt Flag This bit is a flag that generates an ERRI interrupt when a forced termination (abort) of an LPC transfer cycle occurs. 0: [Clearing conditions] • Writing 0 after reading ABRT = 1 • LPC hardware reset and LPC software reset • LPC hardware shutdown and LPC software shutdown 1: [Setting condition] • $\overline{\text{LFRAME}}$ pin falling edge detection during LPC transfer cycle
3	IBFIE3	0	R/W	—	IDR3 and TWR Receive Completion Interrupt Enable Enables or disables IBFI3 interrupt to the slave processor (this LSI). 0: Input data register IDR3 and TWR receive completed interrupt requests disabled 1: [When TWRIE = 0 in LADR3] Input data register (IDR3) receive completed interrupt requests enabled [When TWRIE = 1 in LADR3] Input data register (IDR3) and TWR receive completed interrupt requests enabled
2	IBFIE2	0	R/W	—	IDR2 Receive Completion Interrupt Enable Enables or disables IBFI2 interrupt to the slave processor (this LSI). 0: Input data register (IDR2) receive completed interrupt requests disabled 1: Input data register (IDR2) receive completed interrupt requests enabled
1	IBFIE1	0	R/W	—	IDR1 Receive Completion Interrupt Enable Enables or disables IBFI1 interrupt to the slave processor (this LSI). 0: Input data register (IDR1) receive completed interrupt requests disabled 1: Input data register (IDR1) receive completed interrupt requests enabled

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	ERRIE	0	R/W	—	Error Interrupt Enable Enables or disables ERRI interrupt to the slave processor (this LSI). 0: Error interrupt requests disabled 1: Error interrupt requests enabled

Note:* Only 0 can be written to bits 6 to 4, to clear the flag.

- HICR3

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	LFRAME	Undefined	R	—	$\overline{\text{LFRAME}}$ Pin Monitor
6	CLKRUN	Undefined	R	—	$\overline{\text{CLKRUN}}$ Pin Monitor
5	SERIRQ	Undefined	R	—	SERIRQ Pin Monitor
4	LRESET	Undefined	R	—	$\overline{\text{LRESET}}$ Pin Monitor
3	LPCPD	Undefined	R	—	$\overline{\text{LPCPD}}$ Pin Monitor
2	PME	Undefined	R	—	$\overline{\text{PME}}$ Pin Monitor
1	LSMI	Undefined	R	—	$\overline{\text{LSMI}}$ Pin Monitor
0	LSCI	Undefined	R	—	LSCI Pin Monitor

15.3.3 LPC Channel 3 Address Register (LADR3)

LADR3 comprises two 8-bit readable/writable registers that perform LPC channel-3 host address setting and control the operation of the bidirectional data registers. The contents of the address field in LADR3 must not be changed while channel 3 is operating (while LPC3E is set to 1).

- LADR3H

Bit	Bit Name	Initial Value	R/W	Description
7	Bit 15	0	R/W	Channel 3 Address Bits 15 to 8:
6	Bit 14	0	R/W	When LPC3E = 1, an I/O address received in an LPC I/O cycle is compared with the contents of LADR3. When determining an IDR3, ODR3, or STR3 address match, bit 0 of LADR3 is regarded as 0, and the value of bit 2 is ignored. When determining a TWR0 to TWR15 address match, bit 4 of LADR3 is inverted, and the values of bits 3 to 0 are ignored. Register selection according to the bits ignored in address match determination is as shown in table 15.2.
5	Bit 13	0	R/W	
4	Bit 12	0	R/W	
3	Bit 11	0	R/W	
2	Bit 10	0	R/W	
1	Bit 9	0	R/W	
0	Bit 8	0	R/W	

- LADR3L

Bit	Bit Name	Initial Value	R/W	Description
7	Bit 7	0	R/W	Channel 3 Address Bits 7 to 3
6	Bit 6	0	R/W	
5	Bit 5	0	R/W	
4	Bit 4	0	R/W	
3	Bit 3	0	R/W	
2	—	0	R/W	Reserved This bit is readable/writable, however, only 0 should be written to this bit.
1	Bit 1	0	R/W	Channel 3 Address Bit 1
0	TWRE	0	R/W	Bidirectional Data Register Enable Enables or disables bidirectional data register operation. 0: TWR operation is disabled TWR-related I/O address match determination is halted 1: TWR operation is enabled

Table 15.2 Register Selection

I/O Address					Transfer	
Bit 4	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Host Register Selection**
Bit 4	Bit 3	0	Bit 1	0	I/O write	IDR3 write, C/$\overline{\text{D}}$3 ← 0
Bit 4	Bit 3	1	Bit 1	0	I/O write	IDR3 write, C/$\overline{\text{D}}$3 ← 1
Bit 4	Bit 3	0	Bit 1	0	I/O read	ODR3 read
Bit 4	Bit 3	1	Bit 1	0	I/O read	STR3 read
$\overline{\text{Bit 4}}$	0	0	0	0	I/O write	TWR0MW write
$\overline{\text{Bit 4}}$	0	0	0	1	I/O write	TWR1 to TWR15 write
	1	1	1	1		
$\overline{\text{Bit 4}}$	0	0	0	0	I/O read	TWR0SW read
$\overline{\text{Bit 4}}$	0	0	0	1	I/O read	TWR1 to TWR15 read
	1	1	1	1		

15.3.4 Input Data Registers 1 to 3 (IDR1 to IDR3)

The IDR registers are 8-bit read-only registers for the slave processor (this LSI), and 8-bit write-only registers for the host processor. The registers selected from the host according to the I/O address are shown in the following table. For information on IDR3 selection, see section 15.3.3, LPC Channel 3 Address Register (LADR3). Data transferred in an LPC I/O write cycle is written to the selected register. The state of bit 2 of the I/O address is latched into the C/$\overline{\text{D}}$ bit in STR, to indicate whether the written information is a command or data. The initial values of IDR1 to IDR3 are undefined.

I/O Address					Transfer	
Bits 15 to 4	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Host Register Selection**
0000 0000 0110	0	0	0	0	I/O write	IDR1 write, C/$\overline{\text{D}}$1 ← 0
0000 0000 0110	0	1	0	0	I/O write	IDR1 write, C/$\overline{\text{D}}$1 ← 1
0000 0000 0110	0	0	1	0	I/O write	IDR2 write, C/$\overline{\text{D}}$2 ← 0
0000 0000 0110	0	1	1	0	I/O write	IDR2 write, C/$\overline{\text{D}}$2 ← 1

15.3.5 Output Data Registers 1 to 3 (ODR1 to ODR3)

The ODR registers are 8-bit readable/writable registers for the slave processor (this LSI), and 8-bit read-only registers for the host processor. The registers selected from the host according to the I/O address are shown in the following table. For information on ODR3 selection, see section 15.3.3, LPC Channel 3 Address Register (LADR3). In an LPC I/O read cycle, the data in the selected register is transferred to the host. The initial values of ODR1 to ODR3 are undefined.

I/O Address					Transfer	
Bits 15 to 4	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Host Register Selection**
0000 0000 0110	0	0	0	0	I/O read	ODR1 read
0000 0000 0110	0	0	1	0	I/O read	ODR2 read

15.3.6 Bidirectional Data Registers 0 to 15 (TWR0 to TWR15)

The TWR registers are sixteen 8-bit readable/writable registers to both the slave processor (this LSI) and the host processor. In TWR0, however, two registers (TWR0MW and TWR0SW) are allocated to the same address for both the host address and the slave address. TWR0MW is a write-only register for the host processor, and a read-only register for the slave processor, while TWR0SW is a write-only register for the slave processor and a read-only register for the host processor. When the host and slave processors begin a write, after the respective TWR0 registers have been written to, access right arbitration for simultaneous access is performed by checking the status flags to see if those writes were valid. For the registers selected from the host according to the I/O address, see section 15.3.3, LPC Channel 3 Address Register (LADR3).

Data transferred in an LPC I/O write cycle is written to the selected register; in an LPC I/O read cycle, the data in the selected register is transferred to the host. The initial values of TWR0 to TWR15 are undefined.

15.3.7 Status Registers 1 to 3 (STR1 to STR3)

The STR registers are 8-bit registers that indicate status information during host interface processing. Bits 3, 1, and 0 of STR1 to STR3, and bits 7 to 4 of STR3, are read-only bits for both the host processor and the slave processor (this LSI). However, only 0 can be written to bit 0 of STR1 to STR3 and bits 6 and 4 of STR3, from the slave processor (this LSI), in order to clear the flags to 0. The registers selected from the host processor according to the I/O address are shown in the following table. For information on STR3 selection, see section 15.3.3, LPC Channel 3 Address Register (LADR3). In an LPC I/O read cycle, the data in the selected register is transferred to the host processor. The initial values of STR1 to STR3 are H'00.

I/O Address					Transfer Cycle	Host Register Selection
Bits 15 to 4	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**		
0000 0000 0110	0	1	0	0	I/O read	STR1 read
0000 0000 0110	0	1	1	0	I/O read	STR2 read

- STR1

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	DBU17	0	R/W	R	Defined by User
6	DBU16	0	R/W	R	The user can use these bits as necessary.
5	DBU15	0	R/W	R	
4	DBU14	0	R/W	R	
3	C/$\overline{\text{D}}$1	0	R	R	Command/Data When the host processor writes to an IDR register, bit 2 of the I/O address is written into this bit to indicate whether IDR contains data or a command. 0: Contents of data register (IDR) are data 1: Contents of data register (IDR) are a command
2	DBU12	0	R/W	R	Defined by User The user can use this bit as necessary.
1	IBF1	0	R	R	Input Buffer Full Set to 1 when the host processor writes to IDR. This bit is an internal interrupt source to the slave processor (this LSI). IBF is cleared to 0 when the slave processor reads IDR. The IBF1 flag setting and clearing conditions are different when the fast A20 gate is used. For details, see table 15.3. 0: [Clearing condition] When the slave processor reads IDR 1: [Setting condition] When the host processor writes to IDR using I/O write cycle
0	OBF1	0	R/(W)*	R	Output Buffer Full Set to 1 when the slave processor (this LSI) writes to ODR. Cleared to 0 when the host processor reads ODR. 0: [Clearing condition] When the host processor reads ODR using I/O read cycle, or the slave processor writes 0 to the OBF bit 1: [Setting condition] When the slave processor writes to ODR

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

- STR2

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	DBU27	0	R/W	R	Defined by User The user can use these bits as necessary.
6	DBU26	0	R/W	R	
5	DBU25	0	R/W	R	
4	DBU24	0	R/W	R	
3	C/$\overline{\text{D}}$2	0	R	R	Command/Data When the host processor writes to an IDR register, bit 2 of the I/O address is written into this bit to indicate whether IDR contains data or a command. 0: Contents of data register (IDR) are data 1: Contents of data register (IDR) are a command
2	DBU22	0	R/W	R	Defined by User The user can use this bit as necessary.
1	IBF2	0	R	R	Input Buffer Full Set to 1 when the host processor writes to IDR. This bit is an internal interrupt source to the slave processor (this LSI). IBF is cleared to 0 when the slave processor reads IDR. The IBF1 flag setting and clearing conditions are different when the fast A20 gate is used. For details, see table 15.3. 0: [Clearing condition] When the slave processor reads IDR 1: [Setting condition] When the host processor writes to IDR using I/O write cycle
0	OBF2	0	R/(W)*	R	Output Buffer Full Set to 1 when the slave processor (this LSI) writes to ODR. Cleared to 0 when the host processor reads ODR. 0: [Clearing condition] When the host processor reads ODR using I/O read cycle, or the slave processor writes 0 to the OBF bit 1: [Setting condition] When the slave processor writes to ODR

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

- STR3 (TWRE = 1 or SELSTR3 = 0)

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	IBF3B	0	R	R	Bidirectional Data Register Input Buffer Full Set to 1 when the host processor writes to TWR15. This is an internal interrupt source to the slave processor (this LSI). IBF3B is cleared to 0 when the slave processor reads TWR15. 0: [Clearing condition] When the slave processor reads TWR15 1: [Setting condition] When the host processor writes to TWR15 using I/O write cycle
6	OBF3B	0	R/(W)*	R	Bidirectional Data Register Output Buffer Full Set to 1 when the slave processor (this LSI) writes to TWR15. OBF3B is cleared to 0 when the host processor reads TWR15. 0: [Clearing condition] When the host processor reads TWR15 using I/O read cycle, or the slave processor writes 0 to the OBF3B bit 1: [Setting condition] When the slave processor writes to TWR15
5	MWMF	0	R	R	Master Write Mode Flag Set to 1 when the host processor writes to TWR0. MWMF is cleared to 0 when the slave processor (this LSI) reads TWR15. 0: [Clearing condition] When the slave processor reads TWR15 1: [Setting condition] When the host processor writes to TWR0 using I/O write cycle while SWMF = 0

Bit	Bit Name	Initial Value	R/W Slave	R/W Host	Description
4	SWMF	0	R/(W)*	R	Slave Write Mode Flag Set to 1 when the slave processor (this LSI) writes to TWR0. In the event of simultaneous writes by the master and the slave, the master write has priority. SWMF is cleared to 0 when the host reads TWR15 0: [Clearing condition] When the host processor reads TWR15 using I/O read cycle, or the slave processor writes 0 to the SWMF bit 1: [Setting condition] When the slave processor writes to TWR0 while MWMF = 0
3	C/D3	0	R	R	Command/Data When the host processor writes to an IDR register, bit 2 of the I/O address is written into this bit to indicate whether IDR contains data or a command. 0: Contents of data register (IDR) are data 1: Contents of data register (IDR) are a command
2	DBU32	0	R/W	R	Defined by User The user can use this bit as necessary.
1	IBF3A	0	R	R	Input Buffer Full Set to 1 when the host processor writes to IDR. This bit is an internal interrupt source to the slave processor (this LSI). IBF is cleared to 0 when the slave processor reads IDR. The IBF1 flag setting and clearing conditions are different when the fast A20 gate is used. For details, see table 15.3. 0: [Clearing condition] When the slave processor reads IDR 1: [Setting condition] When the host processor writes to IDR using I/O write cycle

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	OBF3A	0	R/(W)*	R	Output Buffer Full Set to 1 when the slave processor (this LSI) writes to ODR. OBF3A is cleared to 0 when the host processor reads ODR. 0: [Clearing condition] When the host processor reads ODR using I/O read cycle, or the slave processor writes 0 to the OBF bit 1: [Setting condition] When the slave processor writes to ODR

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

- STR3 (TWRE = 0 and SELSTR3 = 1)

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	DBU37	0	R/W	R	Defined by User The user can use these bits as necessary.
6	DBU36	0	R/W	R	
5	DBU35	0	R/W	R	
4	DBU34	0	R/W	R	
3	C/$\overline{D}$3	0	R	R	Command/Data When the host processor writes to an IDR register, bit 2 of the I/O address is written into this bit to indicate whether IDR contains data or a command. 0: Contents of data register (IDR) are data 1: Contents of data register (IDR) are a command
2	DBU32	0	R/W	R	Defined by User The user can use this bit as necessary.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
1	IBF3A	0	R	R	Input Buffer Full Set to 1 when the host processor writes to IDR. This bit is an internal interrupt source to the slave processor (this LSI). IBF is cleared to 0 when the slave processor reads IDR. The IBF1 flag setting and clearing conditions are different when the fast A20 gate is used. For details, see table 15.3. 0: [Clearing condition] When the slave processor reads IDR 1: [Setting condition] When the host processor writes to IDR using I/O write cycle
0	OBF3A	0	R/(W)*	R	Output Buffer Full Set to 1 when the slave processor (this LSI) writes to ODR. OBF3A is cleared to 0 when the host processor reads ODR. 0: [Clearing condition] When the host processor reads ODR using I/O read cycle, or the slave processor writes 0 to the OBF bit 1: [Setting condition] When the slave processor writes to ODR

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

15.3.8 SERIRQ Control Registers 0 and 1 (SIRQCR0, SIRQCR1)

The SIRQCR registers contain status bits that indicate the SERIRQ operating mode and bits that specify SERIRQ interrupt sources.

- SIRQCR0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	$Q/\overline{C}$	0	R	—	Quiet/Continuous Mode Flag Indicates the mode specified by the host at the end of an SERIRQ transfer cycle (stop frame). 0: Continuous mode [Clearing conditions] • LPC hardware reset, LPC software reset • Specification by SERIRQ transfer cycle stop frame 1: Quiet mode [Setting condition] • Specification by SERIRQ transfer cycle stop frame.
6	SELREQ	0	R/W	—	Start Frame Initiation Request Select Selects whether start frame initiation is requested when one or more interrupt requests are cleared, or when all interrupt requests are cleared, in quiet mode. 0: Start frame initiation is requested when all interrupt requests are cleared in quiet mode. 1: Start frame initiation is requested when one or more interrupt requests are cleared in quiet mode.
5	IEDIR	0	R/W	—	Interrupt Enable Direct Mode Specifies whether LPC channel 2 and channel 3 SERIRQ interrupt source (SMI, IRQ6, IRQ9 to IRQ11) generation is conditional upon OBF, or is controlled only by the host interrupt enable bit. 0: Host interrupt is requested when host interrupt enable bit and corresponding OBF are both set to 1 1: Host interrupt is requested when host interrupt enable bit is set to 1

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
4	SMIE3B	0	R/W	—	Host SMI Interrupt Enable 3B Enables or disables a host SMI interrupt request when OBF3B is set by a TWR15 write. 0: Host SMI interrupt request by OBF3B and SMIE3B is disabled [Clearing conditions] • Writing 0 to SMIE3B • LPC hardware reset, LPC software reset • Clearing OBF3B to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host SMI interrupt request by setting OBF3B to 1 is enabled [When IEDIR = 1] Host SMI interrupt is requested [Setting condition] • Writing 1 after reading SMIE3B = 0
3	SMIE3A	0	R/W	—	Host SMI Interrupt Enable 3A Enables or disables a host SMI interrupt request when OBF3A is set by an ODR3 write. 0: Host SMI interrupt request by OBF3A and SMIE3A is disabled [Clearing conditions] • Writing 0 to SMIE3A • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host SMI interrupt request by setting OBF3A to 1 is enabled [When IEDIR = 1] Host SMI interrupt is requested [Setting condition] • Writing 1 after reading SMIE3A = 0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
2	SMIE2	0	R/W	—	Host SMI Interrupt Enable 2 Enables or disables a host SMI interrupt request when OBF2 is set by an ODR2 write. 0: Host SMI interrupt request by OBF2 and SMIE2 is disabled [Clearing conditions] • Writing 0 to SMIE2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host SMI interrupt request by setting OBF2 to 1 is enabled [When IEDIR = 1] Host SMI interrupt is requested [Setting condition] • Writing 1 after reading SMIE2 = 0
1	IRQ12E1	0	R/W	—	Host IRQ12 Interrupt Enable 1 Enables or disables a host IRQ12 interrupt request when OBF1 is set by an ODR1 write. 0: Host IRQ12 interrupt request by OBF1 and IRQ12E1 is disabled [Clearing conditions] • Writing 0 to IRQ12E1 • LPC hardware reset, LPC software reset • Clearing OBF1 to 0 1: Host IRQ12 interrupt request by setting OBF1 to 1 is enabled [Setting condition] • Writing 1 after reading IRQ12E1 = 0

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
0	IRQ1E1	0	R/W	—	Host IRQ1 Interrupt Enable 1 Enables or disables a host IRQ1 interrupt request when OBF1 is set by an ODR1 write. 0: Host IRQ1 interrupt request by OBF1 and IRQ1E1 is disabled [Clearing conditions] • Writing 0 to IRQ1E1 • LPC hardware reset, LPC software reset • Clearing OBF1 to 0 1: Host IRQ1 interrupt request by setting OBF1 to 1 is enabled [Setting condition] • Writing 1 after reading IRQ1E1 = 0

- SIRQCR1

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
7	IRQ11E3	0	R/W	—	Host IRQ11 Interrupt Enable 3 Enables or disables a host IRQ11 interrupt request when OBF3A is set by an ODR3 write. 0: Host IRQ11 interrupt request by OBF3A and IRQ11E3 is disabled [Clearing conditions] • Writing 0 to IRQ11E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ11 interrupt request by setting OBF3A to 1 is enabled [When IEDIR = 1] Host IRQ11 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ11E3 = 0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
6	IRQ10E3	0	R/W	—	Host IRQ10 Interrupt Enable 3 Enables or disables a host IRQ10 interrupt request when OBF3A is set by an ODR3 write. 0: Host IRQ10 interrupt request by OBF3A and IRQ10E3 is disabled [Clearing conditions] • Writing 0 to IRQ10E3 • LPC hardware reset, LPC software reset • Clearing OB3FA to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ10 interrupt request by setting OBF3A to 1 is enabled [When IEDIR = 1] Host IRQ10 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ10E3 = 0
5	IRQ9E3	0	R/W	—	Host IRQ9 Interrupt Enable 3 Enables or disables a host IRQ9 interrupt request when OBF3A is set by an ODR3 write. 0: Host IRQ9 interrupt request by OBF3A and IRQ9E3 is disabled [Clearing conditions] • Writing 0 to IRQ9E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ9 interrupt request by setting OBF3A to 1 is enabled [When IEDIR = 1] Host IRQ9 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ9E3 = 0

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
4	IRQ6E3	0	R/W	—	Host IRQ6 Interrupt Enable 3 Enables or disables a host IRQ6 interrupt request when OBF3A is set by an ODR3 write. 0: Host IRQ6 interrupt request by OBF3A and IRQ6E3 is disabled [Clearing conditions] • Writing 0 to IRQ6E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ6 interrupt request by setting OBF3A to 1 is enabled [When IEDIR = 1] Host IRQ6 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ6E3 = 0
3	IRQ11E2	0	R/W	—	Host IRQ11 Interrupt Enable 2 Enables or disables a host IRQ11 interrupt request when OBF2 is set by an ODR2 write. 0: Host IRQ11 interrupt request by OBF2 and IRQ11E2 is disabled [Clearing conditions] • Writing 0 to IRQ11E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ11 interrupt request by setting OBF2 to 1 is enabled [When IEDIR = 1] Host IRQ11 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ11E2 = 0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
2	IRQ10E2	0	R/W	—	Host IRQ10 Interrupt Enable 2 Enables or disables a host IRQ10 interrupt request when OBF2 is set by an ODR2 write. 0: Host IRQ10 interrupt request by OBF2 and IRQ10E2 is disabled [Clearing conditions] • Writing 0 to IRQ10E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ10 interrupt request by setting OBF2 to 1 is enabled [When IEDIR = 1] Host IRQ10 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ10E2 = 0
1	IRQ9E2	0	R/W	—	Host IRQ9 Interrupt Enable 2 Enables or disables a host IRQ9 interrupt request when OBF2 is set by an ODR2 write. 0: Host IRQ9 interrupt request by OBF2 and IRQ9E2 is disabled [Clearing conditions] • Writing 0 to IRQ9E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ9 interrupt request by setting OBF2 to 1 is enabled [When IEDIR = 1] Host IRQ9 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ9E2 = 0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	IRQ6E2	0	R/W	—	Host IRQ6 Interrupt Enable 2 Enables or disables a host IRQ6 interrupt request when OBF2 is set by an ODR2 write. 0: Host IRQ6 interrupt request by OBF2 and IRQ6E2 is disabled [Clearing conditions] • Writing 0 to IRQ6E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR = 0) 1: [When IEDIR = 0] Host IRQ6 interrupt request by setting OBF2 to 1 is enabled [When IEDIR = 1] Host IRQ6 interrupt is requested. [Setting condition] • Writing 1 after reading IRQ6E2 = 0

15.3.9 Host Interface Select Register (HISEL)

HISEL selects the function of bits 7 to 4 in STR3 and specifies the output of the host interrupt request signal of each frame.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	SELSTR3	0	W		STR3 Register Function Select 3 Selects the function of bits 7 to 4 in STR3 in combination with the TWRE bit in LADR3L. See description on STR3 in section 15.3.7, Status Registers 1 to 3 (STR1 to STR3), for details. 0: Bits 7 to 4 in STR3 are status bits of the host interface. 1: [When TWRE = 1] Bits 7 to 4 in STR3 are status bits of the host interface. [When TWRE = 0] Bits 7 to 4 in STR3 are user bits.
6	SELIRQ11	0	W	—	SERIRQ Output Select Selects the pin output status of host interrupt requests (HIRQ11, HIRQ10, HIRQ9, HIRQ6, SMI, HIRQ12, and HIRQ1) of the LPC. 0: [When host interrupt request is cleared] SERIRQ pin output is in the high-impedance state. [When host interrupt request is set] SERIRQ pin output is 0. 1: [When host interrupt request is cleared] SERIRQ pin output is 0. [When host interrupt request is set] SERIRQ pin output is in the high-impedance state.
5	SELIRQ10	0	W	—	
4	SELIRQ9	0	W	—	
3	SELIRQ6	0	W	—	
2	SELSMI	0	W	—	
1	SELIRQ12	1	W	—	
0	SELIRQ1	1	W	—	

15.4 Operation

15.4.1 Host Interface Activation

The host interface is activated by setting one of bits LPC3E to LPC1E in HICR0 to 1 in single-chip mode. When the host interface is activated, the related I/O ports (ports 37 to 30, ports 83 and 82) function as dedicated host interface input/output pins. In addition, setting the FGA20E, PMEE, LSMIE, and LSCIE bits to 1 adds the related I/O ports (ports 81 and 80, ports PB0 and PB1) to the host interface's input/output pins.

Use the following procedure to activate the host interface after a reset release.

1. Read the signal line status and confirm that the LPC module can be connected. Also check that the LPC module is initialized internally.
2. When using channel 3, set LADR3 to determine the channel 3 I/O address and whether bidirectional data registers are to be used.
3. Set the enable bit (LPC3E to LPC1E) for the channel to be used.
4. Set the enable bits (GA20E, PMEE, LSMIE, and LSCIE) for the additional functions to be used.
5. Set the selection bits for other functions (SDWNE, IEDIR).
6. As a precaution, clear the interrupt flags (LRST, SDWN, ABRT, OBF). Read IDR or TWR15 to clear IBF.
7. Set interrupt enable bits (IBFIE3 to IBFIE1, ERRIE) as necessary.

15.4.2 LPC I/O Cycles

There are ten kinds of LPC transfer cycle: memory read, memory write, I/O read, I/O write, DMA read, DMA write, bus master memory read, bus master memory write, bus master I/O read, and bus master I/O write. Of these, the chip's LPC supports only I/O read and I/O write cycles.

An LPC transfer cycle is started when the $\overline{\text{LFRAME}}$ signal goes low in the bus idle state. If the $\overline{\text{LFRAME}}$ signal goes low when the bus is not idle, this means that a forced termination (abort) of the LPC transfer cycle has been requested.

In an I/O read cycle or I/O write cycle, transfer is carried out using LAD3 to LAD0 in the following order, in synchronization with LCLK. The host can be made to wait by sending back a value other than B′0000 in the slave’s synchronization return cycle, but with the chip’s LPC a value of B′0000 is always returned.

If the received address matches the host address in an LPC register (IDR, ODR, STR, TWR), the host interface enters the busy state; it returns to the idle state by output of a state count 12 turnaround. Register and flag changes are made at this timing, so in the event of a transfer cycle forced termination (abort) before state #12, registers and flags are not changed.

	I/O Read Cycle			I/O Write Cycle		
State Count	**Contents**	**Drive Source**	**Value (3 to 0)**	**Contents**	**Drive Source**	**Value (3 to 0)**
1	Start	Host	0000	Start	Host	0000
2	Cycle type/direction	Host	0000	Cycle type/direction	Host	0010
3	Address 1	Host	Bits 15 to 12	Address 1	Host	Bits 15 to 12
4	Address 2	Host	Bits 11 to 8	Address 2	Host	Bits 11 to 8
5	Address 3	Host	Bits 7 to 4	Address 3	Host	Bits 7 to 4
6	Address 4	Host	Bits 3 to 0	Address 4	Host	Bits 3 to 0
7	Turnaround (recovery)	Host	1111	Data 1	Host	Bits 3 to 0
8	Turnaround	None	ZZZZ	Data 2	Host	Bits 7 to 4
9	Synchronization	Slave	0000	Turnaround (recovery)	Host	1111
10	Data 1	Slave	Bits 3 to 0	Turnaround	None	ZZZZ
11	Data 2	Slave	Bits 7 to 4	Synchronization	Slave	0000
12	Turnaround (recovery)	Slave	1111	Turnaround (recovery)	Slave	1111
13	Turnaround	None	ZZZZ	Turnaround	None	ZZZZ

The timing of the $\overline{\text{LFRAME}}$, LCLK, and LAD signals is shown in figures 15.2 and 15.3.

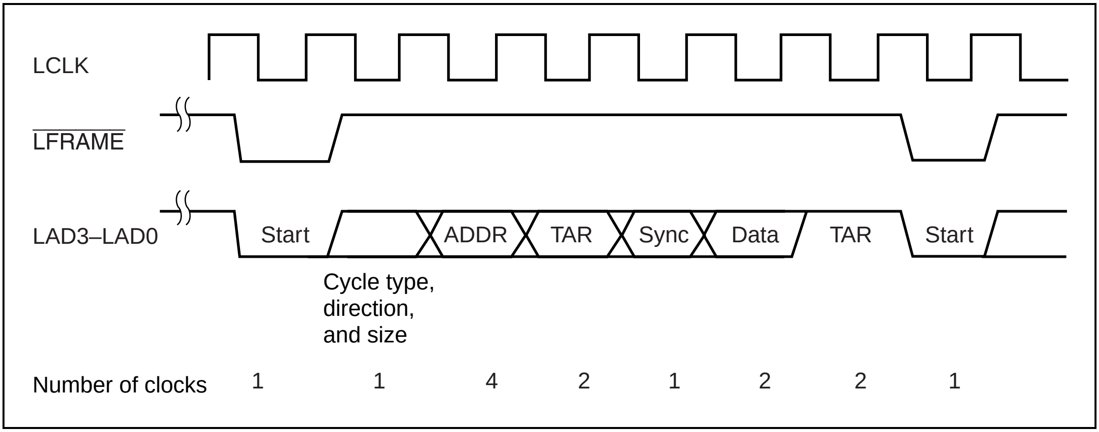


Figure 15.2 Typical $\overline{\text{LFRAME}}$ Timing

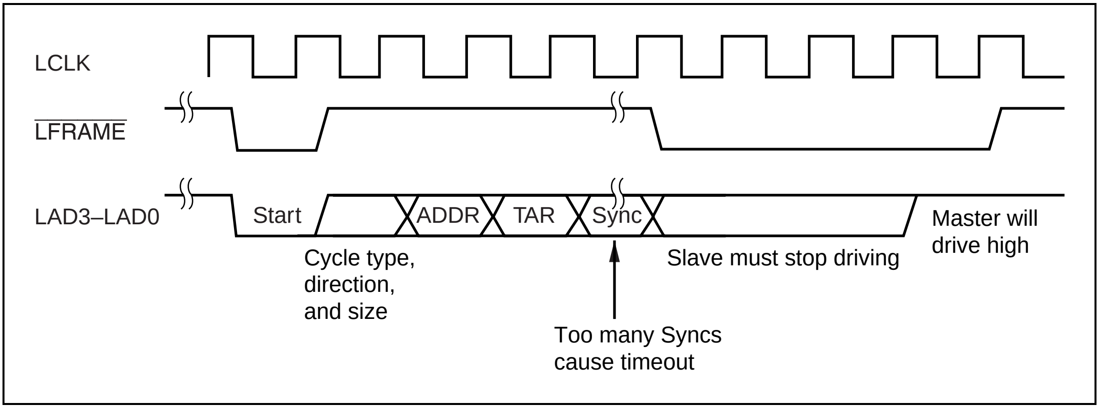


Figure 15.3 Abort Mechanism

15.4.3 A20 Gate

The A20 gate signal can mask address A20 to emulate an addressing mode used by personal computers with an 8086*-family CPU. A regular-speed A20 gate signal can be output under firmware control. The fast A20 gate function that is speeded up by hardware is enabled by setting the FGA20E bit to 1 in HICR0.

Note: An Intel microprocessor

Regular A20 Gate Operation: Output of the A20 gate signal can be controlled by an H'D1 command followed by data. When the slave processor (this LSI) receives data, it normally uses an interrupt routine activated by the IBF1 interrupt to read IDR1. At this time, firmware copies bit 1 of data following an H'D1 command and outputs it at the gate A20 pin.

Fast A20 Gate Operation: The internal state of GA20 output is initialized to 1 when FGA20E = 0. When the FGA20E bit is set to 1, P81/GA20 is used for output of a fast A20 gate signal. The state of the P81/GA20 pin can be monitored by reading the GA20 bit in HICR2.

The initial output from this pin will be a logic 1, which is the initial value. Afterward, the host processor can manipulate the output from this pin by sending commands and data. This function is only available via the IDR1 register. The host interface decodes commands input from the host. When an H'D1 host command is detected, bit 1 of the data following the host command is output from the GA20 output pin. This operation does not depend on firmware or interrupts, and is faster than the regular processing using interrupts. Table 15.3 shows the conditions that set and clear GA20 (P81). Figure 15.4 shows the GA20 output in flowchart form. Table 15.4 indicates the GA20 output signal values.

Table 15.3 GA20 (P81) Set/Clear Timing

Pin Name	Setting Condition	Clearing Condition
GA20 (P81)	When bit 1 of the data that follows an H'D1 host command is 1	When bit 1 of the data that follows an H'D1 host command is 0

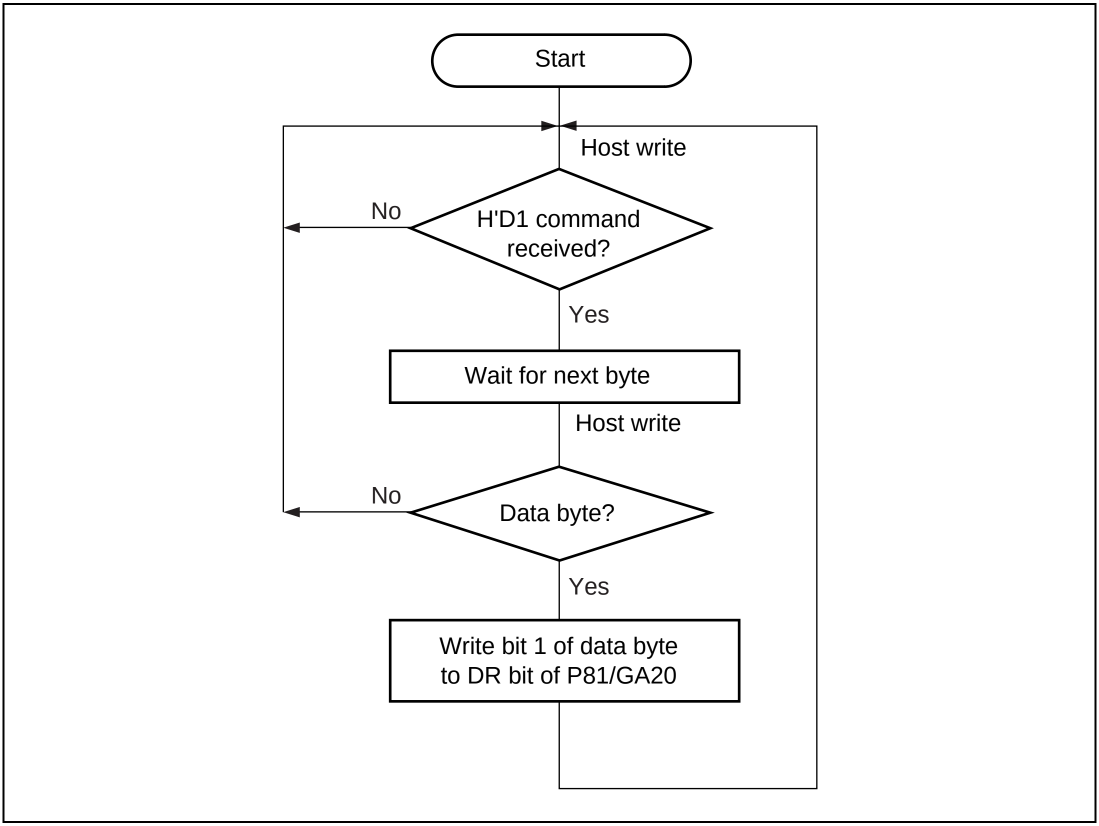


Figure 15.4 GA20 Output

Table 15.4 Fast A20 Gate Output Signals

HA0	Data/Command	Internal CPU Interrupt Flag (IBF)	GA20 (P81)	Remarks
1	H'D1 command	0	Q	Turn-on sequence
0	1 data*[1]	0	1	
1	H'FF command	0	Q (1)	
1	H'D1 command	0	Q	Turn-off sequence
0	0 data*[2]	0	0	
1	H'FF command	0	Q (0)	
1	H'D1 command	0	Q	Turn-on sequence (abbreviated form)
0	1 data*[1]	0	1	
1/0	Command other than H'FF and H'D1	1	Q (1)	
1	H'D1 command	0	Q	Turn-off sequence (abbreviated form)
0	0 data*[2]	0	0	
1/0	Command other than H'FF and H'D1	1	Q (0)	
1	H'D1 command	0	Q	Cancelled sequence
1	Command other than H'D1	1	Q	
1	H'D1 command	0	Q	Retriggered sequence
1	H'D1 command	0	Q	
1	H'D1 command	0	Q	Consecutively executed sequences
0	Any data	0	1/0	
1	H'D1 command	0	Q (1/0)	

Notes: 1. Arbitrary data with bit 1 set to 1.

2. Arbitrary data with bit 1 cleared to 0.

15.4.4 Host Interface Shutdown Function (LPCPD)

The host interface can be placed in the shutdown state according to the state of the $\overline{\text{LPCPD}}$ pin. There are two kinds of host interface shutdown state: LPC hardware shutdown and LPC software shutdown. The LPC hardware shutdown state is controlled by the $\overline{\text{LPCPD}}$ pin, while the software shutdown state is controlled by the SDWNB bit. In both states, the host interface enters the reset state by itself, and is no longer affected by external signals other than the $\overline{\text{LRESET}}$ and $\overline{\text{LPCPD}}$ signals.

Placing the slave processor in sleep mode or software standby mode is effective in reducing current dissipation in the shutdown state. If software standby mode is set, some means must be provided for exiting software standby mode before clearing the shutdown state with the $\overline{\text{LPCPD}}$ signal.

If the SDWNE bit has been set to 1 beforehand, the LPC hardware shutdown state is entered at the same time as the $\overline{\text{LPCPD}}$ signal falls, and prior preparation is not possible. If the LPC software shutdown state is set by means of the SDWNB bit, on the other hand, the LPC software shutdown state cannot be cleared at the same time as the rise of the $\overline{\text{LPCPD}}$ signal. Taking these points into consideration, the following operating procedure uses a combination of LPC software shutdown and LPC hardware shutdown.

1. Clear the SDWNE bit to 0.
2. Set the ERRIE bit to 1 and wait for an interrupt by the SDWN flag.
3. When an ERRI interrupt is generated by the SDWN flag, check the host interface internal status flags and perform any necessary processing.
4. Set the SDWNB bit to 1 to set LPC software standby mode.
5. Set the SDWNE bit to 1 and make a transition to LPC hardware standby mode. The SDWNB bit is cleared automatically.
6. Check the state of the $\overline{\text{LPCPD}}$ signal to make sure that the $\overline{\text{LPCPD}}$ signal has not risen during steps 3 to 5. If the signal has risen, clear SDWNE to 0 to return to the state in step 1.
7. Place the slave processor in sleep mode or software standby mode as necessary.
8. If software standby mode has been set, exit software standby mode by some means independent of the LPC.
9. When a rising edge is detected in the $\overline{\text{LPCPD}}$ signal, the SDWNE bit is automatically cleared to 0. If the slave processor has been placed in sleep mode, the mode is exited by means of $\overline{\text{LRESET}}$ signal input, on completion of the LPC transfer cycle, or by some other means.

Table 15.5 shows the scope of the host interface pin shutdown.

Table 15.5 Scope of Host Interface Pin Shutdown

Abbreviation	Port	Scope of Shutdown	I/O	Notes
LAD3 to LAD0	P33–P30	O	I/O	Hi-Z
$\overline{\text{LFRAME}}$	P34	O	Input	Hi-Z
$\overline{\text{LRESET}}$	P35	×	Input	LPC hardware reset function is active
LCLK	P36	O	Input	Hi-Z
SERIRQ	P37	O	I/O	Hi-Z
LSCI	PB1	Δ	I/O	Hi-Z, only when LSCIE = 1
$\overline{\text{LSMI}}$	PB0	Δ	I/O	Hi-Z, only when LSMIE = 1
$\overline{\text{PME}}$	P80	Δ	I/O	Hi-Z, only when PMEE = 1
GA20	P81	Δ	I/O	Hi-Z, only when FGA20E = 1
$\overline{\text{CLKRUN}}$	P82	O	I/O	Hi-Z
$\overline{\text{LPCPD}}$	P83	×	Input	Needed to clear shutdown state

Legend

O:	Pin that is shutdown by the shutdown function
Δ:	Pin that is shutdown only when the LPC function is selected by register setting
×:	Pin that is not shutdown

In the LPC shutdown state, the LPC's internal state and some register bits are initialized. The order of priority of LPC shutdown and reset states is as follows.

1. System reset (reset by $\overline{\text{STBY}}$ or $\overline{\text{RES}}$ pin input, or WDT0 overflow)
   — All register bits, including bits LPC3E to LPC1E, are initialized.
2. LPC hardware reset (reset by $\overline{\text{LRESET}}$ pin input)
   — LRSTB, SDWNE, and SDWNB bits are cleared to 0.
3. LPC software reset (reset by LRSTB)
   — SDWNE and SDWNB bits are cleared to 0.
4. LPC hardware shutdown
   — SDWNB bit is cleared to 0.
5. LPC software shutdown

The scope of the initialization in each mode is shown in table 15.6.

Table 15.6 Scope of Initialization in Each Host Interface Mode

Items Initialized	System Reset	LPC Reset	LPC Shutdown
LPC transfer cycle sequencer (internal state), LPCBSY and ABRT flags	Initialized	Initialized	Initialized
SERIRQ transfer cycle sequencer (internal state), CLKREQ and IRQBSY flags	Initialized	Initialized	Initialized
Host interface flags (IBF1, IBF2, IBF3A, IBF3B, MWMF, C/$\overline{\text{D}}$1, C/$\overline{\text{D}}$2, C/$\overline{\text{D}}$3, OBF1, OBF2, OBF3A, OBF3B, SWMF, DBU), GA20 (internal state)	Initialized	Initialized	Retained
Host interrupt enable bits (IRQ1E1, IRQ12E1, SMIE2, IRQ6E2, IRQ9E2 to IRQ11E2, SMIE3B, SMIE3A, IRQ6E3, IRQ9E3 to IRQ11E3), Q/$\overline{\text{C}}$ flag, SELREQ bit	Initialized	Initialized	Retained
LRST flag	Initialized (0)	Can be set/cleared	Can be set/cleared
SDWN flag	Initialized (0)	Initialized (0)	Can be set/cleared
LRSTB bit	Initialized (0)	HR: 0 SR: 1	0 (can be set)
SDWNB bit	Initialized (0)	Initialized (0)	HS: 0 SS: 1
SDWNE bit	Initialized (0)	Initialized (0)	HS: 1 SS: 0 or 1
Host interface operation control bits (LPC3E to LPC1E, FGA20E, LADR3, IBFIE1 to IBFIE3, PMEE, PMEB, LSMIE, LSMIB, LSCIE, LSCIB, TWRE, SELSTR3, SELIRQ1, SELSMI, SELIRQ6, SELIRQ9, SELIRQ10, SELIRQ11, SELIRQ12)	Initialized	Retained	Retained
$\overline{\text{LRESET}}$ signal	Input (port function	Input	Input
$\overline{\text{LPCPD}}$ signal		Input	Input
LAD3 to LAD0, $\overline{\text{LFRAME}}$, LCLK, SERIRQ, $\overline{\text{CLKRUN}}$ signals		Input	Hi-Z
$\overline{\text{PME}}$, $\overline{\text{LSMI}}$, LSCI, GA20 signals (when function is selected)		Output	Hi-Z
$\overline{\text{PME}}$, $\overline{\text{LSMI}}$, LSCI, GA20 signals (when function is not selected)		Port function	Port function

Note: System reset: Reset by STBY input, RES input, or WDT overflow
LPC reset: Reset by LPC hardware reset (HR) or LPC software reset (SR)
LPC shutdown: Reset by LPC hardware shutdown (HS) or LPC software shutdown (SS)

Figure 15.5 shows the timing of the $\overline{\text{LPCPD}}$ and $\overline{\text{LRESET}}$ signals.

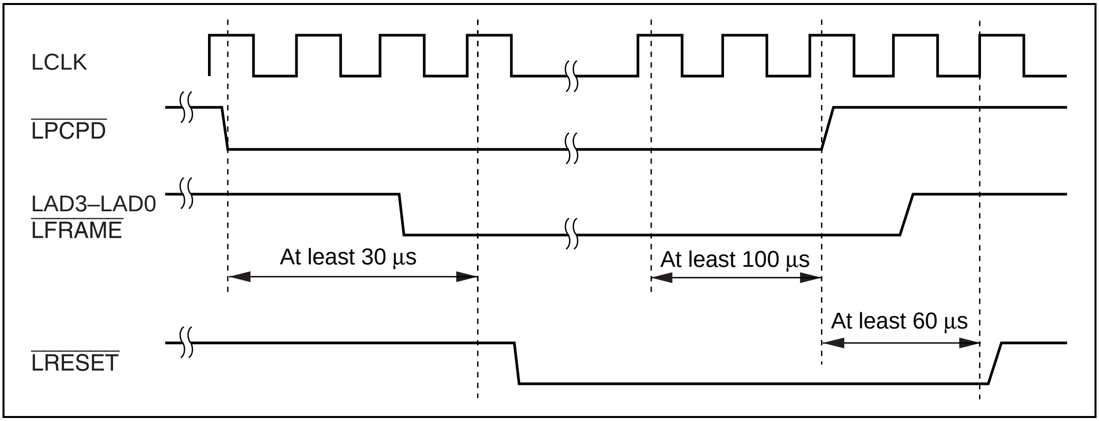

LCLK
$\overline{\text{LPCPD}}$
LAD3–LAD0
$\overline{\text{LFRAME}}$
At least 30 µs
At least 100 µs
At least 60 µs
$\overline{\text{LRESET}}$

Figure 15.5 Power-Down State Termination Timing

15.4.5 Host Interface Serialized Interrupt Operation (SERIRQ)

A host interrupt request can be issued from the host interface by means of the SERIRQ pin. In a host interrupt request via the SERIRQ pin, LCLK cycles are counted from the start frame of the serialized interrupt transfer cycle generated by the host or a peripheral function, and a request signal is generated by the frame corresponding to that interrupt. The timing is shown in figure 15.6.

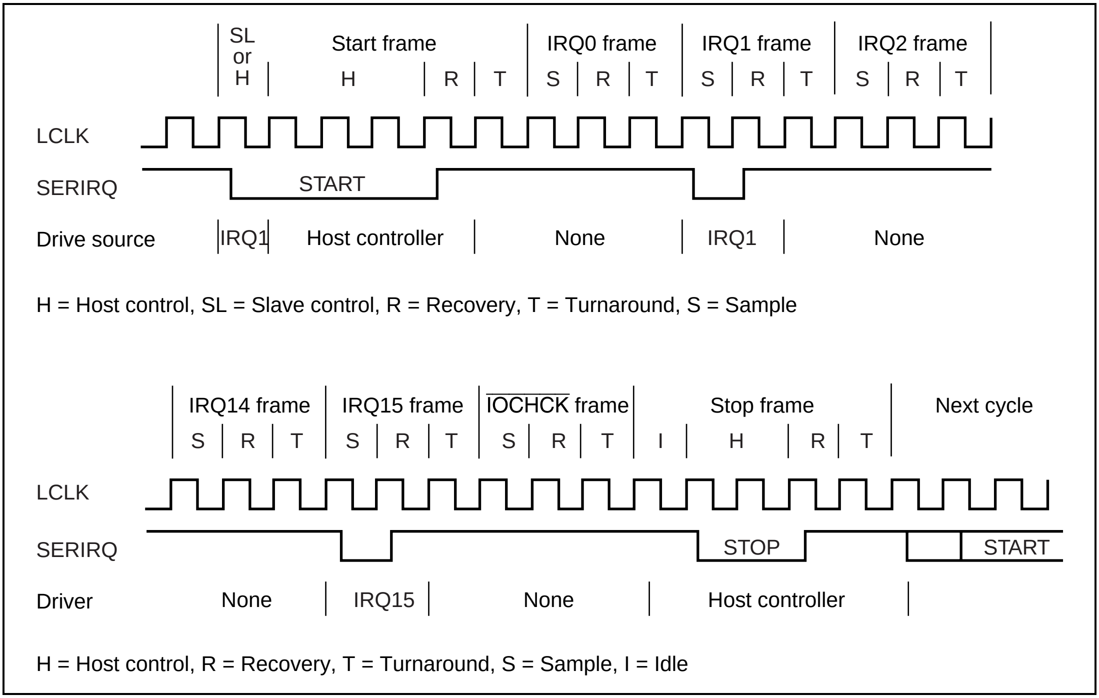


Figure 15.6 SERIRQ Timing

The frame configuration of the serialized interrupt transfer cycle is as follows. Two of the states comprising each frame are the recover state in which the SERIRQ signal is returned to the 1-level at the end of the frame, and the turnaround state in which the SERIRQ signal is not driven. The recover state must be driven by the host or slave processor that was driving the preceding state.

Table 15.7 Frame Configuration of Serial Interrupt Transfer Cycle

	Serial Interrupt Transfer Cycle			
Frame Count	Contents	Drive Source	Number of States	Notes
0	Start	Slave Host	6	In quiet mode only, slave drive possible in first state, then next 3 states 0-driven by host
1	IRQ0	Slave	3	
2	IRQ1	Slave	3	Drive possible in LPC channel 1
3	SMI	Slave	3	Drive possible in LPC channels 2 and 3
4	IRQ3	Slave	3	
5	IRQ4	Slave	3	
6	IRQ5	Slave	3	
7	IRQ6	Slave	3	Drive possible in LPC channels 2 and 3
8	IRQ7	Slave	3	
9	IRQ8	Slave	3	
10	IRQ9	Slave	3	Drive possible in LPC channels 2 and 3
11	IRQ10	Slave	3	Drive possible in LPC channels 2 and 3
12	IRQ11	Slave	3	Drive possible in LPC channels 2 and 3
13	IRQ12	Slave	3	Drive possible in LPC channel 1
14	IRQ13	Slave	3	
15	IRQ14	Slave	3	
16	IRQ15	Slave	3	
17	IOCHCK	Slave	3	
18	Stop	Host	Undefined	First, 1 or more idle states, then 2 or 3 states 0-driven by host 2 states: Quiet mode next 3 states: Continuous mode next

There are two modes—continuous mode and quiet mode—for serialized interrupts. The mode initiated in the next transfer cycle is selected by the stop frame of the serialized interrupt transfer cycle that ended before that cycle.

In continuous mode, the host initiates host interrupt transfer cycles at regular intervals. In quiet mode, the slave processor with interrupt sources requiring a request can also initiate an interrupt transfer cycle, in addition to the host. In quiet mode, since the host does not necessarily initiate interrupt transfer cycles, it is possible to suspend the clock (LCLK) supply and enter the power-down state. In order for a slave to transfer an interrupt request in this case, a request to restart the clock must first be issued to the host. For details, see section 15.4.6, Host Interface Clock Start Request (CLKRUN).

15.4.6 Host Interface Clock Start Request (CLKRUN)

A request to restart the clock (LCLK) can be sent to the host processor by means of the $\overline{\text{CLKRUN}}$ pin. With LPC data transfer and SERIRQ in continuous mode, a clock restart is never requested since the transfer cycles are initiated by the host. With SERIRQ in quiet mode, when a host interrupt request is generated the $\overline{\text{CLKRUN}}$ signal is driven and a clock (LCLK) restart request is sent to the host. The timing for this operation is shown in figure 15.7.

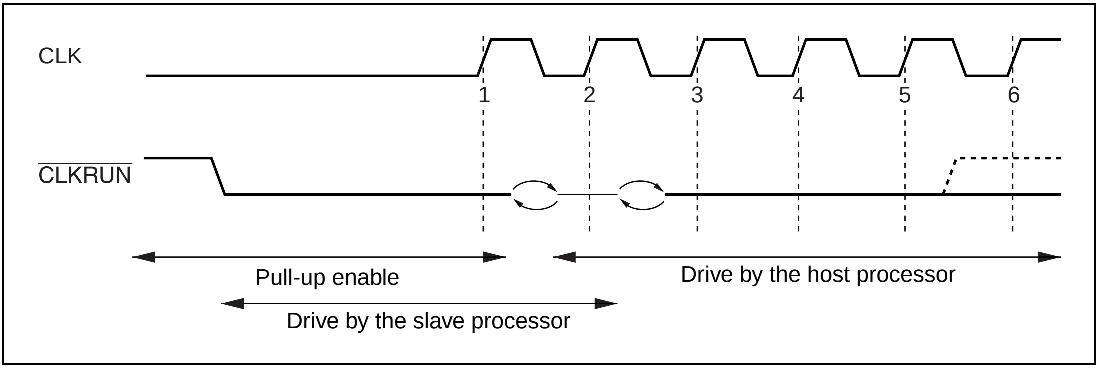

CLK
1
2
3
4
5
6
$\overline{\text{CLKRUN}}$
Pull-up enable
Drive by the host processor
Drive by the slave processor

Figure 15.7 Clock Start Request Timing

Cases other than SERIRQ in quiet mode when clock restart is required must be handled with a different protocol, using the $\overline{\text{PME}}$ signal, etc.

15.5 Interrupt Sources

15.5.1 IBFI1, IBFI2, IBFI3, and ERRI

The host interface has four interrupt requests for the slave processor (this LSI): IBF1, IBF2, IBF3, and ERRI. IBFI1, IBFI2, and IBFI3 are IDR receive complete interrupts for IDR1, IDR2, and IDR3 and TWR, respectively. The ERRI interrupt indicates the occurrence of a special state such as an LPC reset, LPC shutdown, or transfer cycle abort. An interrupt request is enabled by setting the corresponding enable bit.

Table 15.8 Receive Complete Interrupts and Error Interrupt

Interrupt	Description
IBFI1	When IBFIE1 is set to 1 and IDR1 reception is completed
IBFI2	When IBFIE2 is set to 1 and IDR2 reception is completed
IBFI3	When IBFIE3 is set to 1 and IDR3 reception is completed, or when TWRE and IBFIE3 are set to 1 and reception is completed up to TWR15
ERRI	When ERRIE is set to 1 and one of LRST, SDWN and ABRT is set to 1

15.5.2 SMI, HIRQ1, HIRQ6, HIRQ9, HIRQ10, HIRQ11, and HIRQ12

The host interface can request seven kinds of host interrupt by means of SERIRQ. HIRQ1 and HIRQ12 are used on LPC channel 1 only, while SMI, HIRQ6, HIRQ9, HIRQ10, and HIRQ11 can be requested from LPC channel 2 or 3.

There are two ways of clearing a host interrupt request.

When the IEDIR bit is cleared to 0 in SIRQCR0, host interrupt sources and LPC channels are all linked to the host interrupt request enable bits. When the OBF flag is cleared to 0 by a read of ODR or TWR15 by the host in the corresponding LPC channel, the corresponding host interrupt enable bit is automatically cleared to 0, and the host interrupt request is cleared.

When the IEDIR bit is set to 1 in SIRQCR0, LPC channel 2 and 3 interrupt requests are dependent only upon the host interrupt enable bits. The host interrupt enable bit is not cleared when OBF for channel 2 or 3 is cleared. Therefore, SMIE2, SMIE3A and SMIE3B, IRQ6E2 and IRQ6E3, IRQ9E2 and IRQ9E3, IRQ10E2 and IRQ10E3, and IRQ11E2 and IRQ11E3 lose their respective functional differences. In order to clear a host interrupt request, it is necessary to clear the host interrupt enable bit.

Table 15.9 summarizes the methods of setting and clearing these bits, and Figure 15.8 shows the processing flowchart.

Table 15.9 HIRQ Setting and Clearing Conditions

Host Interrupt	Setting Condition	Clearing Condition
HIRQ1 (independent from IEDIR)	Internal CPU writes to ODR1, then reads 0 from bit IRQ1E1 and writes 1	Internal CPU writes 0 to bit IRQ1E1, or host reads ODR1
HIRQ12 (independent from IEDIR)	Internal CPU writes to ODR1, then reads 0 from bit IRQ12E1 and writes 1	Internal CPU writes 0 to bit IRQ12E1, or host reads ODR1
SMI (IEDIR = 0)	Internal CPU • writes to ODR2, then reads 0 from bit SMIE2 and writes 1 • writes to ODR3, then reads 0 from bit SMIE3A and writes 1 • writes to TWR15, then reads 0 from bit SMIE3B and writes 1	Internal CPU • writes 0 to bit SMIE2, or host reads ODR2 • writes 0 to bit SMIE3A, or host reads ODR3 • writes 0 to bit SMIE3B, or host reads TWR15
SMI (IEDIR = 1)	Internal CPU • reads 0 from bit SMIE2, then writes 1 • reads 0 from bit SMIE3A, then writes 1 • reads 0 from bit SMIE3B, then writes 1	Internal CPU • writes 0 to bit SMIE2 • writes 0 to bit SMIE3A • writes 0 to bit SMIE3B
HIRQi (i = 6, 9, 10, 11) (IEDIR = 0)	Internal CPU • writes to ODR2, then reads 0 from bit IRQiE2 and writes 1 • writes to ODR3, then reads 0 from bit IRQiE3 and writes 1	Internal CPU • writes 0 to bit IRQiE2, or host reads ODR2 • CPU writes 0 to bit IRQiE3, or host reads ODR3
HIRQi (i = 6, 9, 10, 11) (IEDIR = 1)	Internal CPU • reads 0 from bit IRQiE2, then writes 1 • reads 0 from bit IRQiE3, then writes 1	Internal CPU • writes 0 to bit IRQiE2 • writes 0 to bit IRQiE3

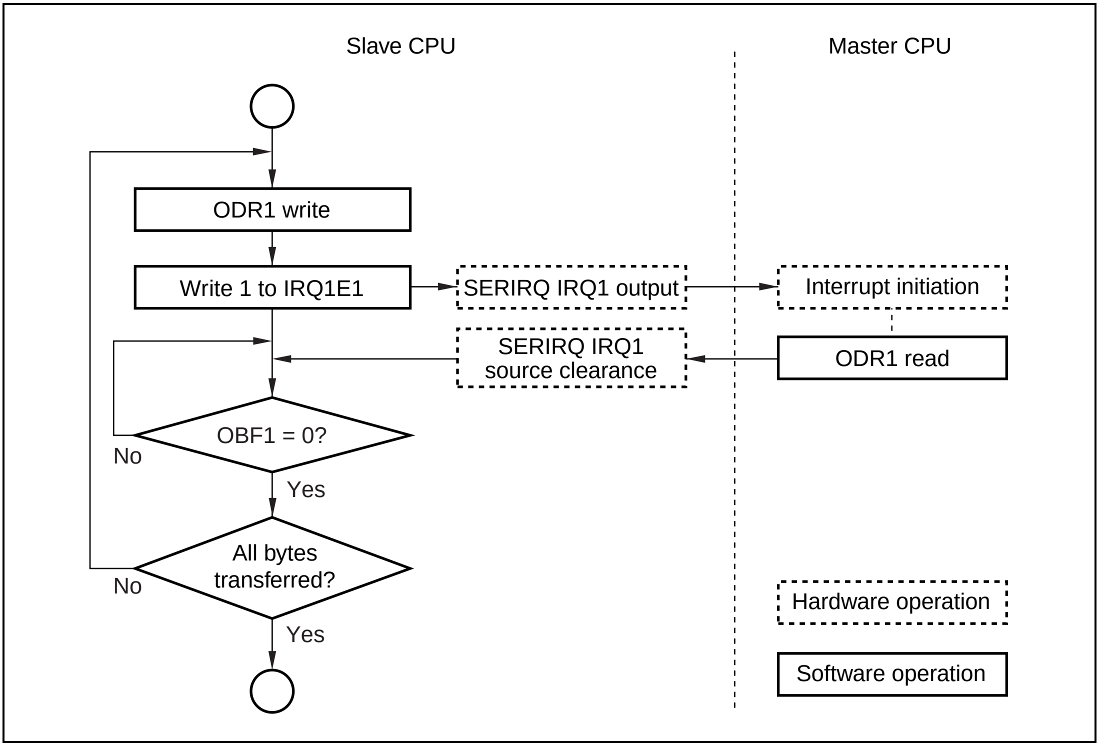


Figure 15.8 HIRQ Flowchart (Example of Channel 1)

15.6 Usage Notes

15.6.1 Module Stop Mode Setting

LPC operation can be enabled or disabled using the module stop control register. The initial setting is for LPC operation to be halted. Register access is enabled by canceling module stop mode. For details, refer to section 19, Power-Down Modes.

15.6.2 Notes on Using Host Interface

The host interface provides buffering of asynchronous data from the host processor and slave processor (this LSI), but an interface protocol that uses the flags in STR must be followed to avoid data contention. For example, if the host and slave processor both try to access IDR or ODR at the same time, the data will be corrupted. To prevent simultaneous accesses, IBF and OBF must be used to allow access only to data for which writing has finished.

Unlike the IDR and ODR registers, the transfer direction is not fixed for the bidirectional data registers (TWR). MWMF and SWMF are provided in STR to handle this situation. After writing to TWR0, MWMF and SWMF must be used to confirm that the write authority for TWR1 to TWR15 has been obtained.

Table 15.10 shows host address examples for LADR3 and registers, IDR3, ODR3, STR3, TWR0MW, TWR0SW, and TWR1 to TWR15 when LADR3 = H'A24F and LADR3 = H'3FD0.

Table 15.10 Host Address Example

Register	Host Address when LADR3 = H'A24F	Host Address when LADR3 = H'3FD0
IDR3	H'A24A and H'A24E	H'3FD0 and H'3FD4
ODR3	H'A24A	H'3FD0
STR3	H'A24E	H'3FD4
TWR0MW	H'A250	H'3FC0
TWR0SW	H'A250	H'3FC0
TWR1	H'A251	H'3FC1
TWR2	H'A252	H'3FC2
TWR3	H'A253	H'3FC3
TWR4	H'A254	H'3FC4
TWR5	H'A255	H'3FC5
TWR6	H'A256	H'3FC6
TWR7	H'A257	H'3FC7
TWR8	H'A258	H'3FC8
TWR9	H'A259	H'3FC9
TWR10	H'A25A	H'3FCA
TWR11	H'A25B	H'3FCB
TWR12	H'A25C	H'3FCC
TWR13	H'A25D	H'3FCD
TWR14	H'A25E	H'3FCE
TWR15	H'A25F	H'3FCF

Section 16 RAM

This LSI has an 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, refer to section 3.2.2, System Control Register (SYSCR).

Product Classification		RAM Capacitance	RAM Address
Flash memory version	H8S/2110B	2 kbytes	H′E880 to H′EFFF, H′FF00 to H′FF7F

Section 17 ROM

This LSI has an on-chip ROM (flash memory or masked ROM). The features of the flash memory are summarized below.

A block diagram of the flash memory is shown in figure 17.1.

17.1 Features

- Size

Product Classification	ROM Capacitance	ROM Address
H8S/2110B	64 kbytes	H’000000 to H’00FFFF (mode 2) H’0000 to H’DFFF (mode 3)

- 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:

  — 8 kbytes × 2 blocks, 16 kbytes × 1 block, 28 kbytes × 1 block, and 1 kbyte × 4 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 µs (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.

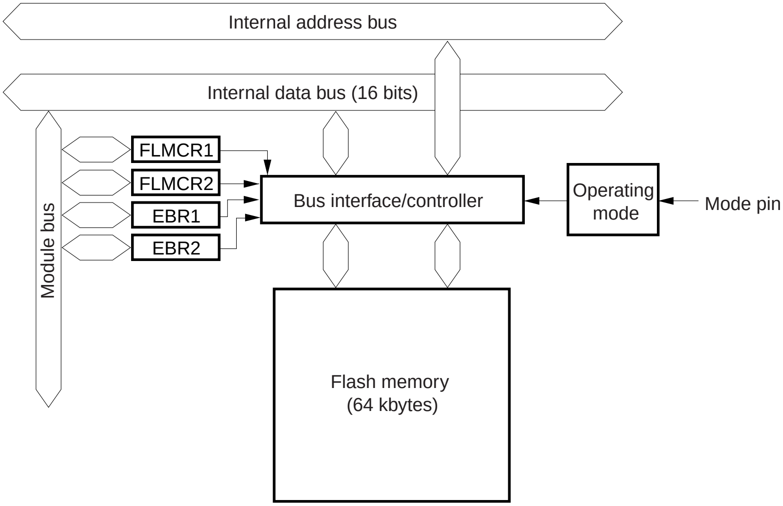
- Programming/erasing protection

  Sets protection against flash memory programming/erasing via hardware, software, or error protection.

ROMF360A_000020020800

- Programmer mode

  In addition to on-board programming mode, programmer mode is supported to program or erase the flash memory using a PROM programmer.



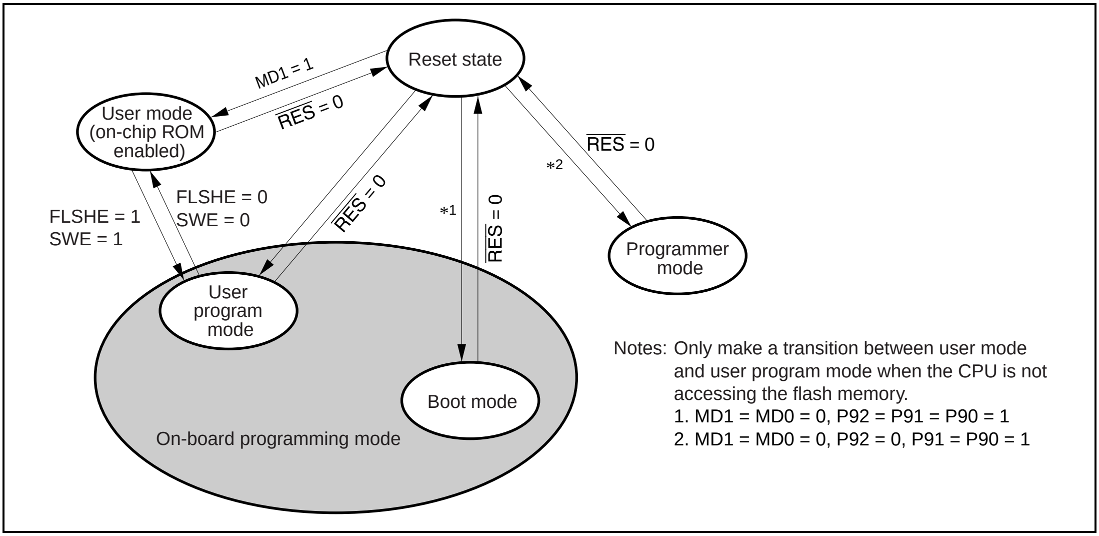
Legend
FLMCR1 : Flash memory control register 1
FLMCR2 : Flash memory control register 2
EBR1 : Erase block register 1
EBR2 : Erase block register 2

Figure 17.1 Block Diagram of Flash Memory

17.2 Mode Transitions

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 17.2. 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 17.1. Figure 17.3 shows the boot mode and figure 17.4 shows the user program mode.



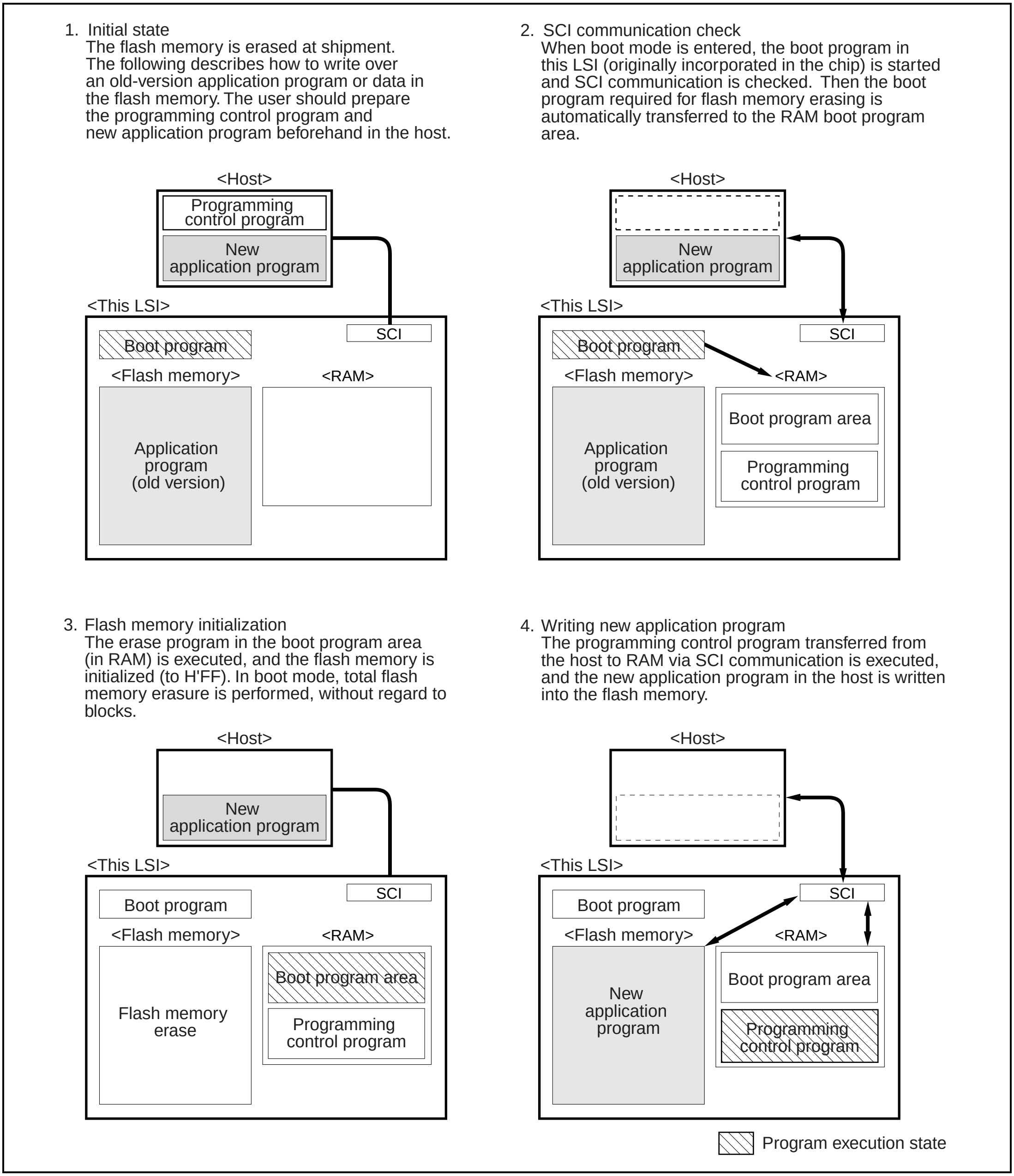
Notes: Only make a transition between user mode and user program mode when the CPU is not accessing the flash memory.
1. MD1 = MD0 = 0, P92 = P91 = P90 = 1
2. MD1 = MD0 = 0, P92 = 0, P91 = P90 = 1

Figure 17.2 Flash Memory State Transitions

Table 17.1 Differences between Boot Mode and User Program Mode

	Boot Mode	User Program Mode
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.



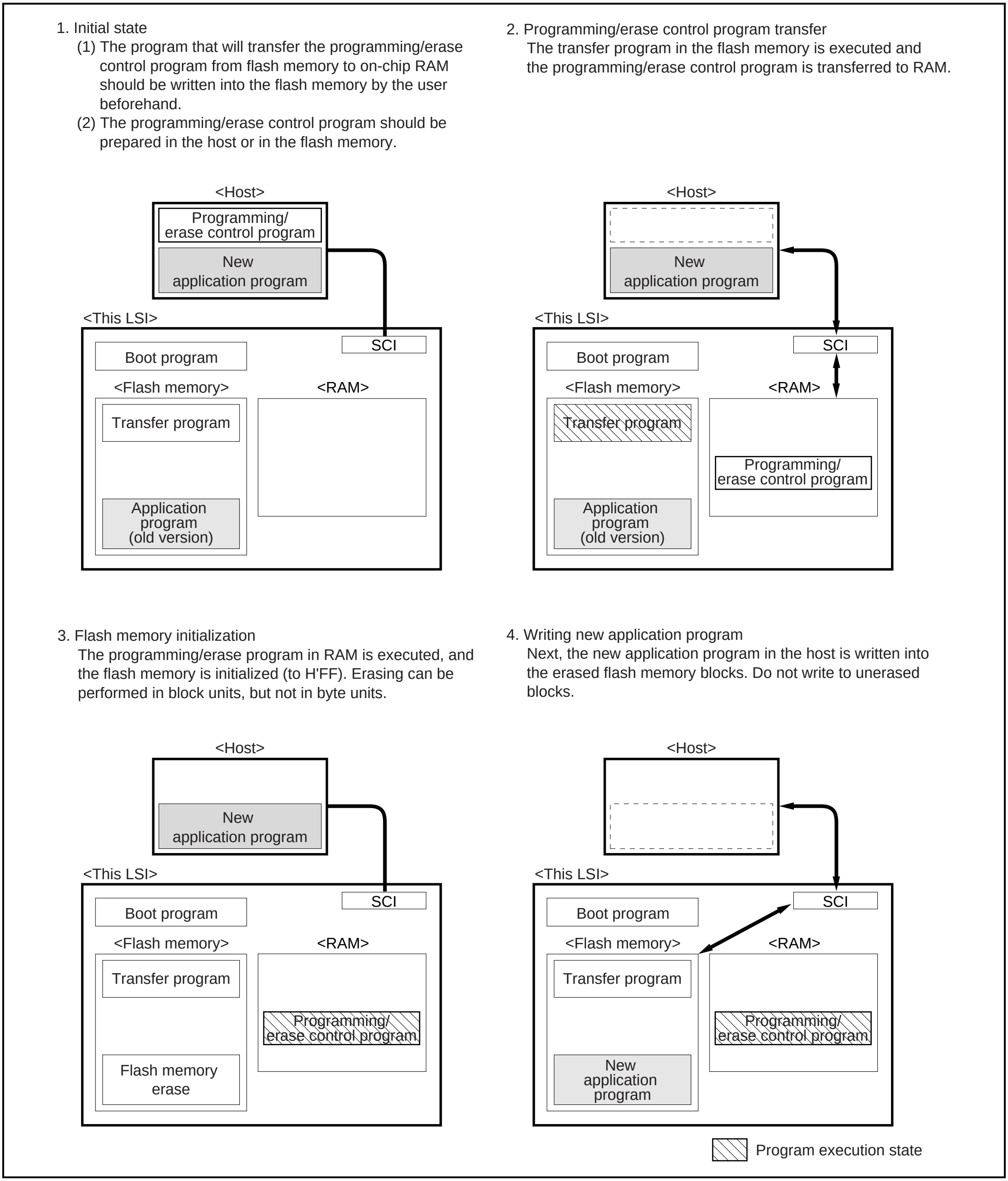
1. Initial state
The flash memory is erased at shipment. The following describes how to write over an old-version application program or data in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.

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.

Figure 17.3 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.

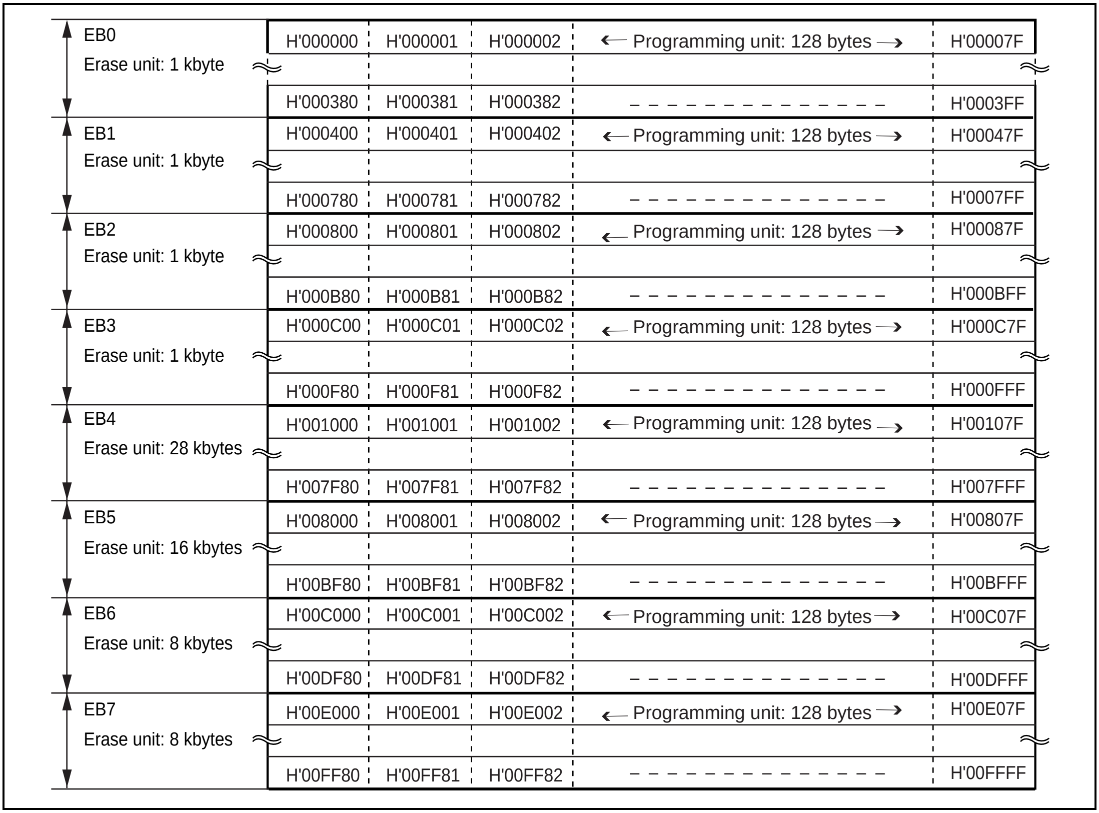
3. Flash memory initialization
   The programming/erase program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.

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.

Figure 17.4 User Program Mode (Example)

17.3 Block Configuration

Figure 17.5 shows the block configuration of 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 8 kbytes (2 blocks), 16 kbytes (1 block), 28 kbytes (1 block), and 1 kbyte (4 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: 1 kbyte	H'000000	H'000001	H'000002	← Programming unit: 128 bytes →	H'00007F
		H'000380	H'000381	H'000382	– – – – – – – – – – – – – – –	H'0003FF
EB1	Erase unit: 1 kbyte	H'000400	H'000401	H'000402	← Programming unit: 128 bytes →	H'00047F
		H'000780	H'000781	H'000782	– – – – – – – – – – – – – – –	H'0007FF
EB2	Erase unit: 1 kbyte	H'000800	H'000801	H'000802	← Programming unit: 128 bytes →	H'00087F
		H'000B80	H'000B81	H'000B82	– – – – – – – – – – – – – – –	H'000BFF
EB3	Erase unit: 1 kbyte	H'000C00	H'000C01	H'000C02	← Programming unit: 128 bytes →	H'000C7F
		H'000F80	H'000F81	H'000F82	– – – – – – – – – – – – – – –	H'000FFF
EB4	Erase unit: 28 kbytes	H'001000	H'001001	H'001002	← Programming unit: 128 bytes →	H'00107F
		H'007F80	H'007F81	H'007F82	– – – – – – – – – – – – – – –	H'007FFF
EB5	Erase unit: 16 kbytes	H'008000	H'008001	H'008002	← Programming unit: 128 bytes →	H'00807F
		H'00BF80	H'00BF81	H'00BF82	– – – – – – – – – – – – – – –	H'00BFFF
EB6	Erase unit: 8 kbytes	H'00C000	H'00C001	H'00C002	← Programming unit: 128 bytes →	H'00C07F
		H'00DF80	H'00DF81	H'00DF82	– – – – – – – – – – – – – – –	H'00DFFF
EB7	Erase unit: 8 kbytes	H'00E000	H'00E001	H'00E002	← Programming unit: 128 bytes →	H'00E07F
		H'00FF80	H'00FF81	H'00FF82	– – – – – – – – – – – – – – –	H'00FFFF

Figure 17.5 Flash Memory Block Configuration

17.4 Input/Output Pins

The flash memory is controlled by means of the pins shown in table 17.2.

Table 17.2 Pin Configuration

Pin Name	I/O	Function
$\overline{\text{RES}}$	Input	Reset
MD1	Input	Sets this LSI’s operating mode
MD0	Input	Sets this LSI’s operating mode
P92	Input	Sets this LSI’s operating mode
P91	Input	Sets this LSI’s operating mode
P90	Input	Sets this LSI’s operating mode
TxD1	Output	Serial transmit data output
RxD1	Input	Serial receive data input

17.5 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, refer to 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)

17.5.1 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, refer to section 17.8, Flash Memory Programming/Erasing.FLMCR1 is initialized to H'80 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	1	R	Flash Write Enable Controls programming/erasing of on-chip flash memory. This bit is always read as 0, and cannot be modified.
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 EBR1 and EBR2 bits cannot be set to 1. Do not clear these bits and SWE to 0 simultaneously.
5	—	0	R	Reserved
4	—	0	R	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.

17.5.2 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	Flash memory error 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 17.9.3, Error Protection.
6 to 2	—	All 0	R/(W)	Reserved The initial values should not be modified.
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.

17.5.3 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, sub-sleep mode, or watch 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 0	—	All 0	R/(W)	Reserved The initial values should not be modified.

- EBR2

Bit	Bit Name	Initial Value	R/W	Description
7	EB7	0	R/W*	When this bit is set to 1, 8 kbytes of EB7 (H'00E000 to H’00FFFF) are to be erased.
6	EB6	0	R/W	When this bit is set to 1, 8 kbytes of EB6 (H'00C000 to H’00DFFF) are to be erased.
5	EB5	0	R/W	When this bit is set to 1, 16 kbytes of EB5 (H'008000 to H'00BFFF) are to be erased.
4	EB4	0	R/W	When this bit is set to 1, 28 kbytes of EB4 (H'001000 to H'007FFF) are to be erased.
3	EB3	0	R/W	When this bit is set to 1, 1 kbyte of EB3 (H'000C00 to H'000FFF) is to be erased.
2	EB2	0	R/W	When this bit is set to 1, 1 kbyte of EB2 (H'000800 to H'000BFF) is to be erased.
1	EB1	0	R/W	When this bit is set to 1, 1 kbyte of EB1 (H'000400 to H'0007FF) is to be erased.
0	EB0	0	R/W	When this bit is set to 1, 1 kbyte of EB0 (H'000000 to H'0003FF) is to be erased.

Note:* In normal mode, this bit is always read as 0 and cannot be modified.

17.6 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.

In normal mode (mode 3), up to 56 kbytes of ROM can be used.

Table 17.3 Operating Modes and ROM

Operating Modes			Mode Pins		
MCU Operating Mode	CPU Operating Mode	Mode	MD1	MD0	On-Chip ROM
Mode 2	Advanced	Single-chip mode	1	0	Enabled (64 kbytes)
Mode 3	Normal	Single-chip mode	1	1	Enabled (56 kbytes)

17.7 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 17.4 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 17.2.

Table 17.4 On-Board Programming Mode Settings

Mode Setting		MD1	MD0	P92	P91	P90
Boot mode		0	0	1*	1*	1*
User program mode	Mode 2 (advanced mode)	1	0	—	—	—
	Mode 3 (normal mode)	1	1	—	—	—

Note:* Can be used as an I/O port after the boot mode activation.

17.7.1 Boot Mode

Table 17.5 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 17.8, 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 17.6.
5. In boot mode, a part of the on-chip RAM area is used by the boot program. Addresses H'FFE080 to H'FFE87F*[1] is the area to which the programming control program is transferred from the host. Note, however, that ID codes are assigned to addresses H'FFE080 to H'FFE087. The boot program area cannot be used until the execution state in boot mode switches to the programming control program. Figure 17.6 shows the on-chip RAM area in boot mode.
6. Before branching to the programming control program (H'FFE088 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*[2] 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.
9. All interrupts are disabled during programming or erasing of the flash memory.

Notes: 1. This area is reserved for boot mode. Do not use this area for any other purpose.

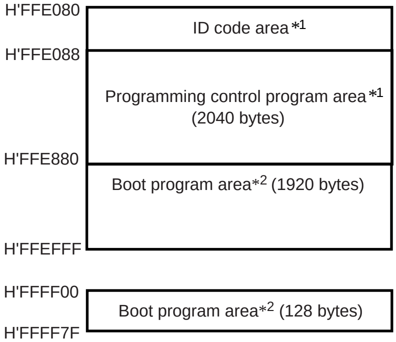
2. After reset is cancelled, mode pin input settings must satisfy the mode programming setup time (t_{MDS} = 4 states).

Table 17.5 Boot Mode Operation

Item	Host Operation	Communications Contents	LSI Operation
	Processing Contents		Processing Contents
Boot mode start			Branches to boot program at reset-start. Boot program 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	• Measures low-level period of receive data H'00. • Calculates bit rate and sets it in BRR of SCI_1. • Transmits data H'00 to host as adjustment end indication. 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 17.6 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 10 MHz
9600 bps	4 to 10 MHz
4800 bps	2 to 10 MHz


H'FFE080
ID code area*1
H'FFE088
Programming control program area*1 (2040 bytes)
H'FFE880
Boot program area*2 (1920 bytes)
H'FFEFFF
H'FFFF00
Boot program area*2 (128 bytes)
H'FFFF7F

Notes: 1. This area is reserved for boot mode. Do not use this area for any other purpose.
2. The boot program area and area which is not used cannot be used until a transition is made to the execution state for the programming control program transferred to RAM.
Note that the contents of the boot program area in RAM are remained after a branch is made to the programming control program.

Figure 17.6 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.

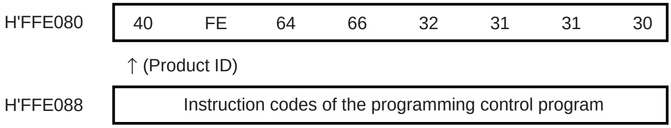


Figure 17.7 ID Code Area

17.7.2 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 17.8 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 17.8, Flash Memory Programming/Erasing.

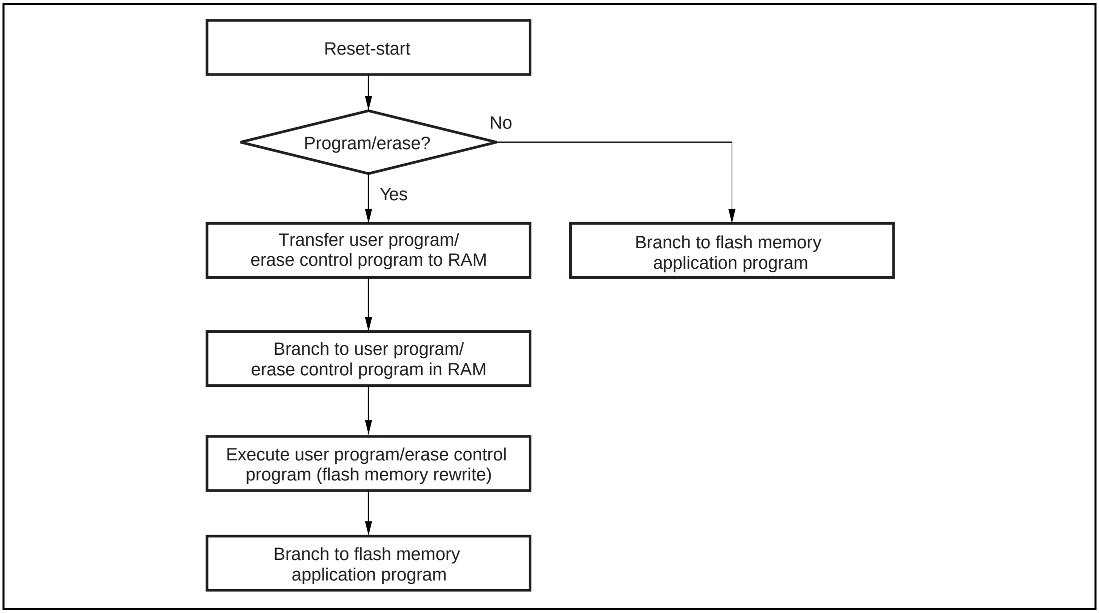

Reset-start
Program/erase?
No
Yes
Transfer user program/ erase control program to RAM
Branch to flash memory application program
Branch to user program/ erase control program in RAM
Execute user program/erase control program (flash memory rewrite)
Branch to flash memory application program

Figure 17.8 Programming/Erasing Flowchart Example in User Program Mode

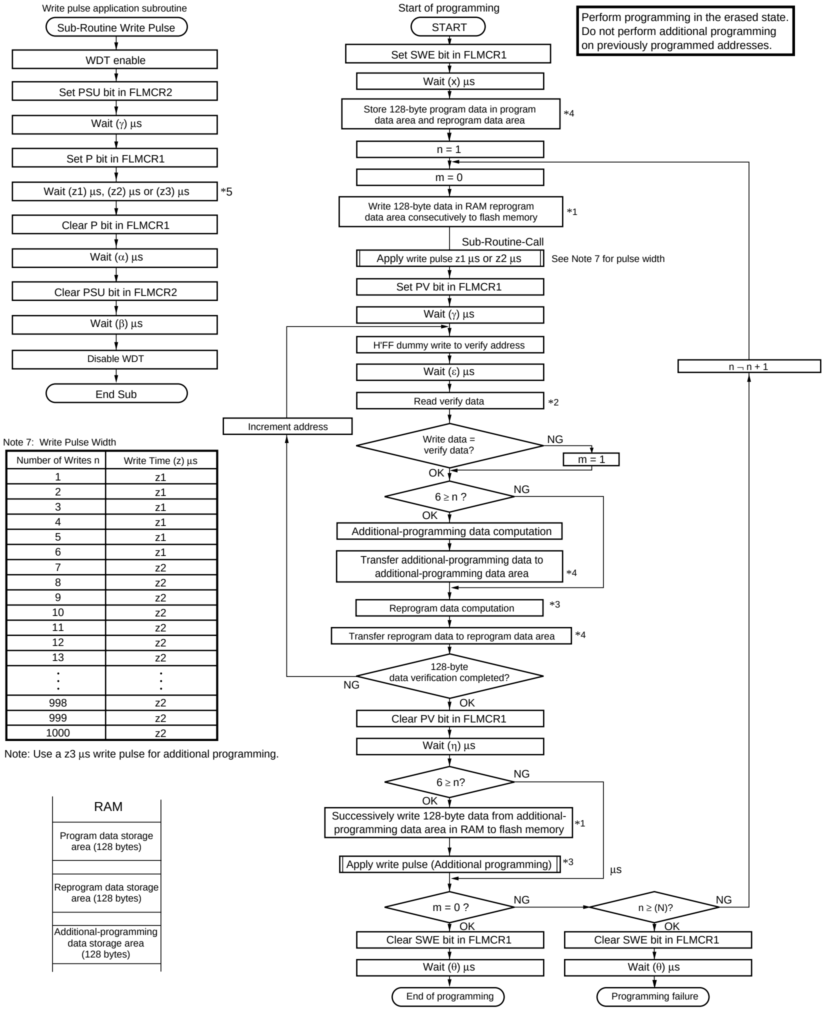
17.8 Flash Memory Programming/Erasing

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, program-verify mode, erase 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 17.8.1, Program/Program-Verify and section 17.8.2, Erase/Erase-Verify, respectively.

17.8.1 Program/Program-Verify

When writing data or programs to the flash memory, the program/program-verify flowchart shown in figure 17.9 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 17.9.
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 17.9 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 ($y + z2 + \alpha + \beta$) µs.
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).


Write pulse application subroutine
Sub-Routine Write Pulse
WDT enable
Set PSU bit in FLMCR2
Wait (γ) µs
Set P bit in FLMCR1
Wait (z1) µs, (z2) µs or (z3) µs
*5
Clear P bit in FLMCR1
Wait (α) µs
Clear PSU bit in FLMCR2
Wait (β) µs
Disable WDT
End Sub
Start of programming
START
Perform programming in the erased state. Do not perform additional programming on previously programmed addresses.
Set SWE bit in FLMCR1
Wait (x) µs
Store 128-byte program data in program data area and reprogram data area
*4
n = 1
m = 0
Write 128-byte data in RAM reprogram data area consecutively to flash memory
*1
Sub-Routine-Call
Apply write pulse z1 µs or z2 µs
See Note 7 for pulse width
Set PV bit in FLMCR1
Wait (γ) µs
H'FF dummy write to verify address
Wait (ε) µs
Read verify data
*2
Increment address
n ¬ n + 1
Write data = verify data?
NG
m = 1
OK
6 ≥ n ?
NG
OK
Additional-programming data computation
Transfer additional-programming data to additional-programming data area
*4
Reprogram data computation
*3
Transfer reprogram data to reprogram data area
*4
128-byte data verification completed?
NG
OK
Clear PV bit in FLMCR1
Wait (η) µs
6 ≥ n?
NG
OK
Successively write 128-byte data from additional-programming data area in RAM to flash memory
*1
Apply write pulse (Additional programming)
*3
µs
m = 0 ?
NG
OK
n ≥ (N)?
NG
OK
Clear SWE bit in FLMCR1
Wait (θ) µs
End of programming
Clear SWE bit in FLMCR1
Wait (θ) µs
Programming failure
RAM
Program data storage area (128 bytes)
Reprogram data storage area (128 bytes)
Additional-programming data storage area (128 bytes)

Note 7: Write Pulse Width

Number of Writes n	Write Time (z) µs
1	z1
2	z1
3	z1
4	z1
5	z1
6	z1
7	z2
8	z2
9	z2
10	z2
11	z2
12	z2
13	z2
⋮	⋮
998	z2
999	z2
1000	z2

Note: Use a z3 µs write pulse for additional programming.

Notes: 1. Data transfer is performed by byte transfer. The lower 8 bits of the first address written to must be H'00 or H'80. A 128-byte data transfer must be performed even if writing fewer than 128 bytes; in this case, H'FF data must be written to the extra addresses.
2. Verify data is read in 16-bit (word) units.
3. Even bits for which programming has been completed will be subjected to programming once again if the result of the subsequent verify operation is NG.
4. A 128-byte area for storing program data, a 128-byte area for storing reprogram data, and a 128-byte area for storing additional data must be provided in RAM. The contents of the reprogram data area and additional data area are modified as programming proceeds.
5. A write pulse of z1 µs or z2 µs is applied according to the progress of the programming operation. See Note7 for details of the pulse widths. When writing of additional-programming data is executed, a z3 µs write pulse should be applied. Reprogram data X' means reprogram data when the write pulse is applied.
6. The values of x, y, z1, z2, z3, α, β, γ, ε, η, θ, and N are shown in section 21.1.4, Flash Memory Characteristics.

Reprogram Data Computation Table

Original Data (D)	Verify Data (V)	Reprogram Data (X)	Comments
0	0	1	Programming completed
0	1	0	Programming incomplete; reprogram
1	0	1	
1	1	1	Still in erased state; no action

Additional-Programming Data Computation Table

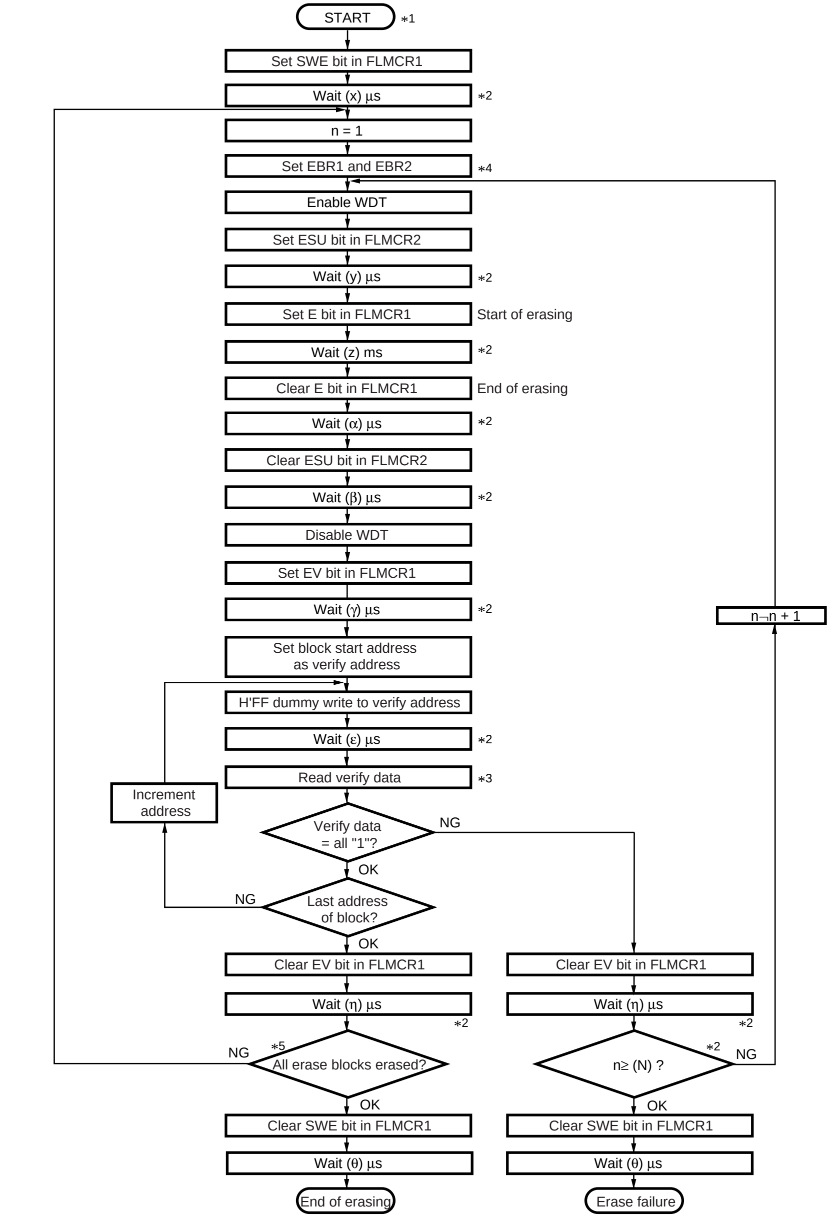
Reprogram Data (X')	Verify Data (V)	Additional-Programming Data (Y)	Comments
0	0	0	Additional programming to be executed
0	1	1	Additional programming not to be executed
1	0	1	
1	1	1	Additional programming not to be executed

Figure 17.9 Program/Program-Verify Flowchart

17.8.2 Erase/Erase-Verify

When erasing flash memory, the erase/erase-verify flowchart shown in figure 17.10 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 + α + β) 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 longwords 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.



Notes: 1. Prewriting (writing 0 to all data in erased block) is not necessary.
2. The values of x, y, z, α, β, γ, ε, η, θ, and N are shown in section 21.1.4, Flash Memory Characteristics.
3. Verify data is read in 16-bit (word) units.
4. Set only a single bit in EBR1 and EBR2. Do not set more than one bit.
5. Erasing is performed in block units. To erase multiple blocks, each block must be erased in turn.

Figure 17.10 Erase/Erase-Verify Flowchart

17.9 Program/Erase Protection

There are three kinds of flash memory program/erase protection: hardware protection, software protection, and error protection.

17.9.1 Hardware Protection

Hardware protection is a state in which programming/erasing of flash memory is forcibly disabled or aborted 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 registers 1 and 2 (FLMCR1 and 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.

17.9.2 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.

17.9.3 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

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.

17.10 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*[1].

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*[2].
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.

17.11 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 Hitachi 64-kbyte flash memory on-chip MCU device*. Figure 17.11 shows a memory map in programmer mode.

Note: Set the programming voltage of the PROM programmer to 3.3V.

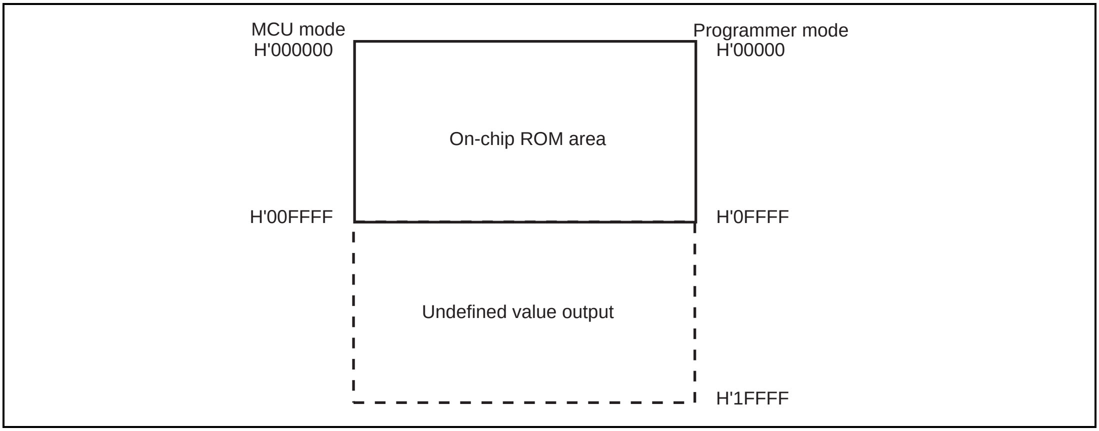


Figure 17.11 Memory Map in Programmer Mode

17.12 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 Hitachi 64-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.
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 μs 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.

Section 18 Clock Pulse Generator

This LSI incorporates a clock pulse generator, which generates the system clock (φ), bus master clock, and internal clock.

The clock pulse generator consists of an oscillator, duty correction circuit, clock select circuit, medium-speed clock divider, bus master clock select circuit, subclock input circuit, and waveform forming circuit. Figure 18.1 shows a block diagram of the clock pulse generator.

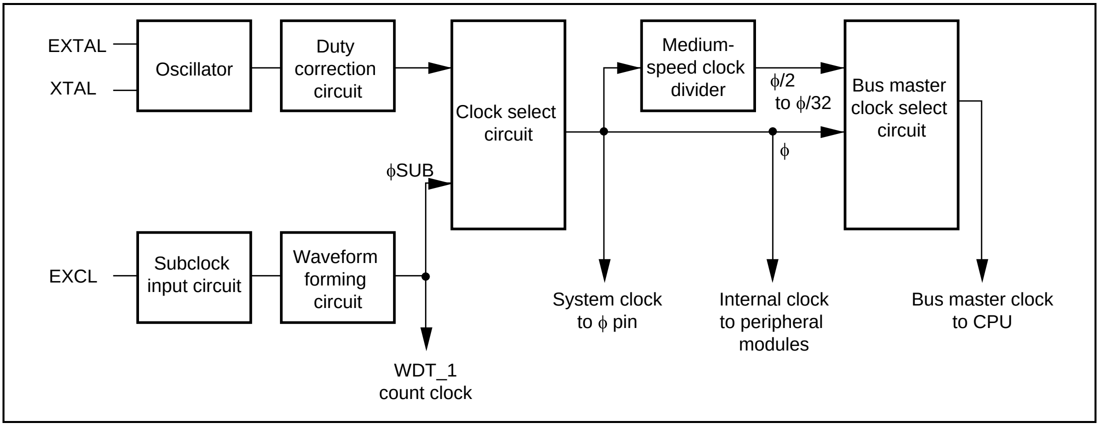


Figure 18.1 Block Diagram of Clock Pulse Generator

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. For details on the standby control register, refer to section 19.1.1, 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, refer to section 19.1.2, Low Power Control Register (LPWRCR).

18.1 Oscillator

Clock pulses can be supplied either by connecting a crystal resonator, or by providing external clock input.

18.1.1 Connecting Crystal Resonator

Figure 18.2 shows a typical method of connecting a crystal resonator. An appropriate damping resistance R_d, given in table 18.1, should be used. An AT-cut parallel-resonance crystal resonator should be used.

Figure 18.3 shows the equivalent circuit of a crystal resonator. A resonator having the characteristics given in table 18.2 should be used.

A crystal resonator with frequency identical to that of the system clock (ϕ) should be used.

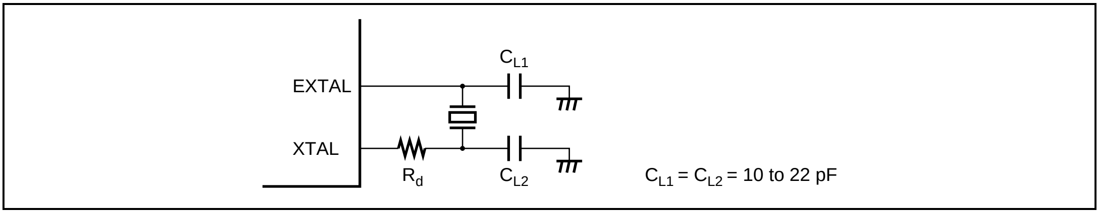


Figure 18.2 Typical Connection to Crystal Resonator

Table 18.1 Damping Resistance Values

Frequency (MHz)	2	4	8	10
R_d (Ω)	1 k	500	200	0

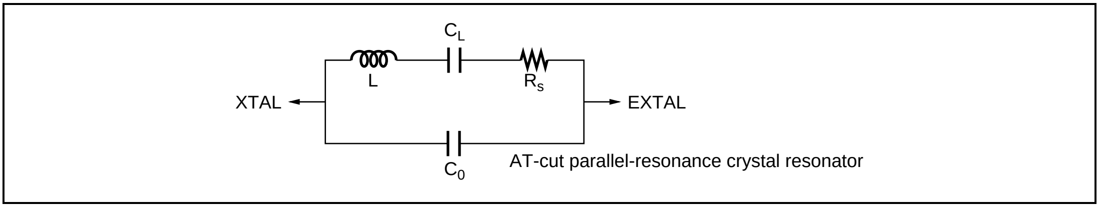


Figure 18.3 Equivalent Circuit of Crystal Resonator

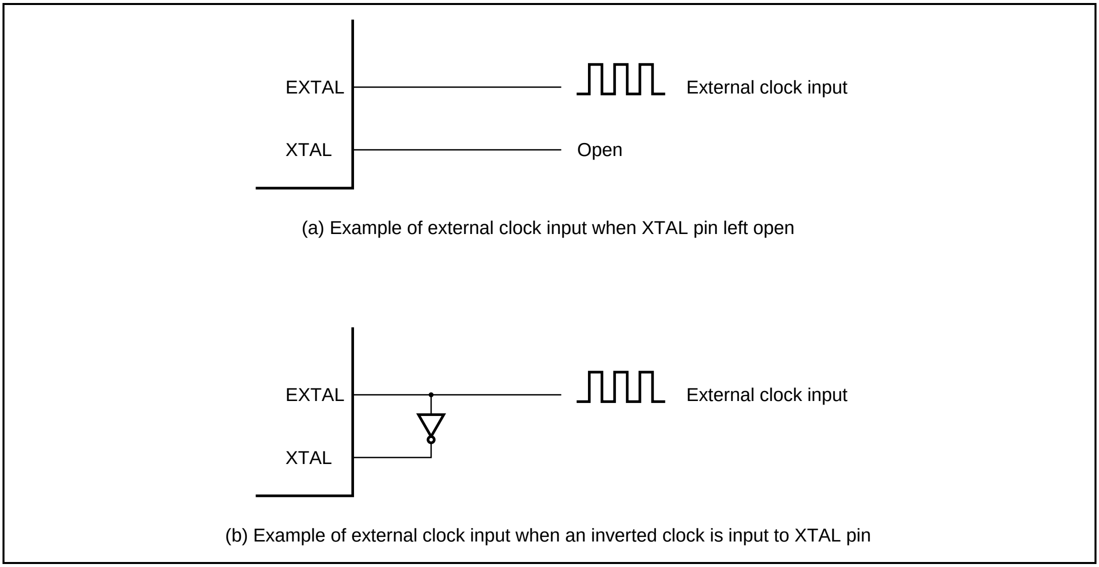
Table 18.2 Crystal Resonator Parameters

Frequency (MHz)	2	4	8	10
R_S (max) (Ω)	500	120	80	70
C_0 (max) (pF)	7			

18.1.2 External Clock Input Method

Figure 18.4 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 18.3. The frequency of the external clock should be the same as that of the system clock (ϕ).



(a) Example of external clock input when XTAL pin left open

(b) Example of external clock input when an inverted clock is input to XTAL pin

Figure 18.4 Example of External Clock Input

Table 18.3 External Clock Input Conditions

		V_{cc} = 2.7 to 3.6 V				
Item	**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**	
External clock input pulse width low level	t_{EXL}	40	—	ns	Figure 18.5	
External clock input pulse width high level	t_{EXH}	40	—	ns		
External clock rising time	t_{EXr}	—	10	ns		
External clock falling time	t_{EXf}	—	10	ns		
Clock pulse width low level	t_{CL}	0.4	0.6	t_{cyc}	$\phi \geq 5$ MHz	Figure 21.5
		80	—	ns	$\phi < 5$ MHz	
Clock pulse width high level	t_{CH}	0.4	0.6	t_{cyc}	$\phi \geq 5$ MHz	
		80	—	ns	$\phi < 5$ MHz	

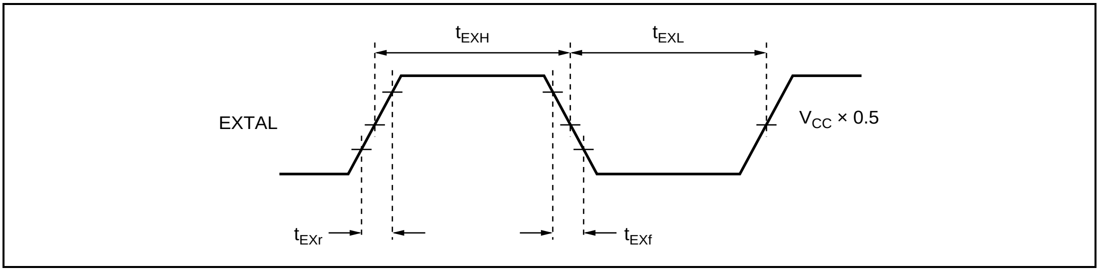


Figure 18.5 External Clock Input Timing

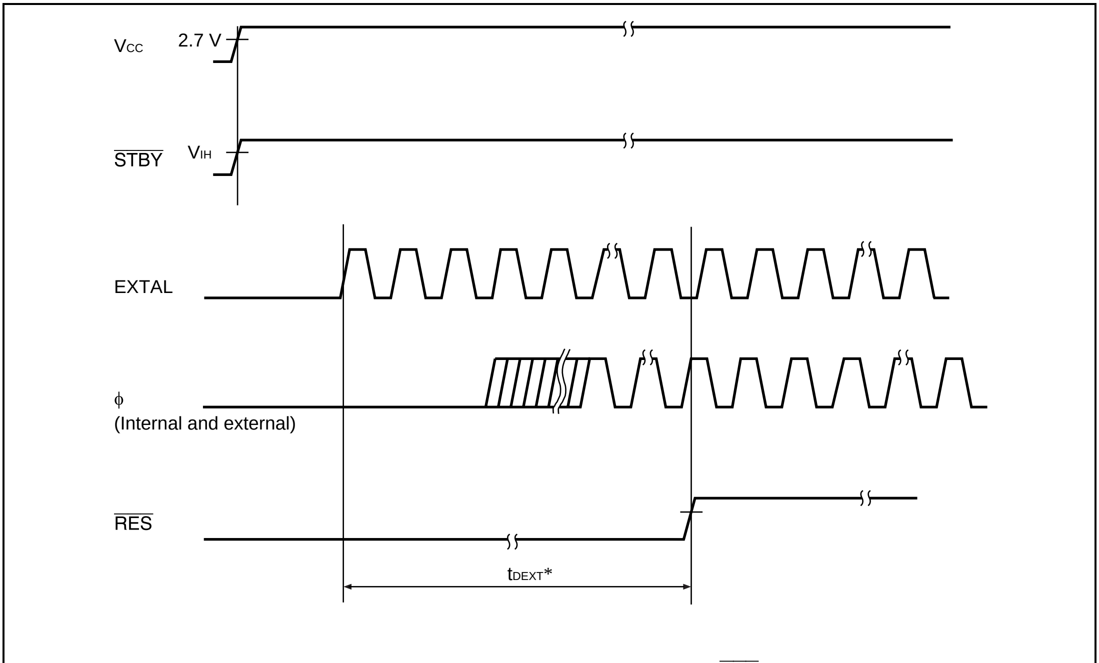
The oscillator and duty correction circuit have a function to adjust the waveform of the external clock input that is input to the EXTAL pin. When a specified clock signal is input to the EXTAL pin, internal clock signal output is determined after the external clock output stabilization delay time (t_{DEXT}) 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 18.4 shows the external clock output stabilization delay time. Figure 18.6 shows the timing of the external clock output stabilization delay time.

Table 18.4 External Clock Output Stabilization Delay Time

Condition: V_{CC} = 2.7 V to 3.6 V, V_{SS} = 0 V

Item	Symbol	Min.	Max.	Unit	Remarks
External clock output stabilization delay time	t_{DEXT}*	500	—	µs	Figure 18.6

Note:* t_{DEXT} includes a $\overline{\text{RES}}$ pulse width (t_{RESW}).



Note:* The external clock output stabilization delay time (t_{DEXT}) includes a $\overline{\text{RES}}$ pulse width (t_{RESW}).

Figure 18.6 Timing of External Clock Output Stabilization Delay Time

18.2 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 (φ).

18.3 Medium-Speed Clock Divider

The medium-speed clock divider divides the system clock (φ), and generates φ/2, φ/4, φ/8, φ/16, and φ/32 clocks.

18.4 Bus Master Clock Select Circuit

The bus master clock select circuit selects a clock to supply the bus master with either the system clock (φ) or medium-speed clock (φ/2, φ/4, φ/8, φ/16, or φ/32) by the SCK2 to SCK0 bits in SBYCR.

18.5 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 18.5. When the subclock is not used, subclock input should not be enabled.

Table 18.5 Subclock Input Conditions

		Vcc = 2.7 to 3.6 V				
Item	**Symbol**	**Min**	**Typ**	**Max**	**Unit**	**Measurement Condition**
Subclock input pulse width low level	t_{EXCLL}	—	15.26	—	µs	Figure 18.7
Subclock input pulse width high level	t_{EXCLH}	—	15.26	—	µs	
Subclock input rising time	t_{EXCLr}	—	—	10	ns	
Subclock input falling time	t_{EXCLf}	—	—	10	ns	

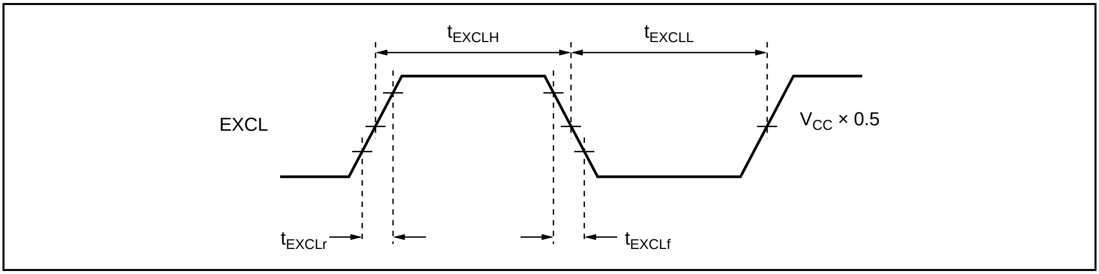


Figure 18.7 Subclock Input Timing

18.6 Waveform Forming Circuit

To remove noise from the subclock input at the EXCL pin, the subclock is sampled by a divided φ 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.

18.7 Clock Select Circuit

The clock select circuit selects the system clock that is used in this LSI.

A clock generated by an oscillator to which the EXTAL and XTAL pins are input is selected as a system clock when returning from high-speed mode, medium-speed mode, sleep mode, reset state, or 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 φSUB. The count clock and sampling clock for each timer are divided φSUB clocks.

18.8 Usage Notes

18.8.1 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 oscillator pins does not exceed the maximum rating.

18.8.2 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 XTAL and EXTAL pins.

Other signal lines should be routed away from the oscillator circuit to prevent inductive interference with the correct oscillation as shown in figure 18.8.

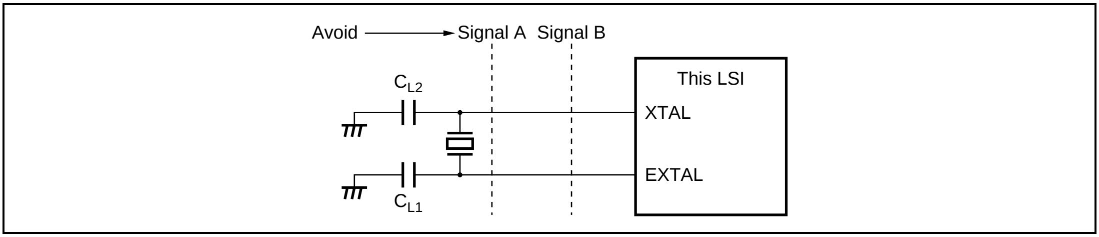


Figure 18.8 Note on Board Design of Oscillator Circuit Section

Section 19 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.

19.1 Register Descriptions

Power-down modes are controlled by the following registers. To access SBYCR, LPWRCR, 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)
- Module stop control register H (MSTPCRH)
- Module stop control register L (MSTPCRL)

19.1.1 Standby Control Register (SBYCR)

SBYCR controls power-down modes.

Bit	Bit Name	Initial Value	R/W	Description
7	SSBY	0	R/W	Software Standby Specifies the operating mode to be entered after executing the SLEEP instruction. When the SLEEP instruction is executed in high-speed mode or medium-speed mode: 0: Shifts to sleep mode 1: Shifts to software standby mode, subactive mode, or watch mode When the SLEEP instruction is executed in subactive mode: 0: Shifts to subsleep mode 1: Shifts to watch mode or high-speed mode Note that the SSBY bit is not changed even if a mode transition occurs by an interrupt.
6 5 4	STS2 STS1 STS0	0 0 0	R/W R/W R/W	Standby Timer Select 2 to 0 Selects 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 19.1 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.
3	—	0	R	Reserved This bit is always read as 0, and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
2	SCK2	0	R/W	System Clock Select 2 to 0
1	SCK1	0	R/W	Selects a clock for the bus master in high-speed mode or medium-speed mode. When making a transition to subactive mode or watch mode, SCK2 to SCK0 must be cleared to 0. 000: High-speed mode 001: Medium-speed clock: ϕ/2 010: Medium-speed clock: ϕ/4 011: Medium-speed clock: ϕ/8 100: Medium-speed clock: ϕ/16 101: Medium-speed clock: ϕ/32 11X: —
0	SCK0	0	R/W	

Legend

X: Don't care

Table 19.1 Operating Frequency and Wait Time

STS2	STS1	STS0	Wait Time	10 MHz	8 MHz	6 MHz	4 MHz	2 MHz	Unit
0	0	0	8192 states	0.8	1.0	1.3	20.	4.1	ms
0	0	1	16384 states	1.6	2.0	2.7	4.1	8.2	
0	1	0	32768 states	3.3	4.1	5.5	8.2	16.4	
0	1	1	65536 states	6.6	8.2	10.9	16.4	32.8	
1	0	0	131072 states	13.1	16.4	21.8	32.8	65.5	
1	0	1	262144 states	26.2	32.8	43.6	65.6	131.2	
1	1	0	Reserved	—	—	—	—	—	—
1	1	1	Reserved	—	—	—	—	—	—

Shaded cells indicate the recommended specification.

19.1.2 Low-Power Control Register (LPWRCR)

LPWRCR controls power-down modes.

Bit	Bit Name	Initial Value	R/W	Description
7	DTON	0	R/W	Direct Transfer On Flag Specifies the operating mode to be entered after executing the SLEEP instruction. When the SLEEP instruction is executed in high-speed mode or medium-speed mode: 0: Shifts to sleep mode, software standby mode, or watch mode 1: Shifts directly to subactive mode, or shifts to sleep mode or software standby mode When the SLEEP instruction is executed in subactive mode: 0: Shifts to subsleep mode or watch mode 1: Shifts directly to high-speed mode, or shifts to subsleep mode
6	LSON	0	R/W	Low-Speed On Flag 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. When the SLEEP instruction is executed in high-speed mode or medium-speed mode: 0: Shifts to sleep mode, software standby mode, or watch mode 1: Shifts to watch mode or subactive mode When the SLEEP instruction is executed in subactive mode: 0: Shifts directly to watch mode or high-speed mode 1: Shifts to subsleep mode or watch mode When watch mode is cancelled: 0: Shifts to high-speed mode 1: Shifts to subactive mode

Bit	Bit Name	Initial Value	R/W	Description
5	NESEL	0	R/W	Noise Elimination Sampling Frequency Select Selects the frequency by which the subclock (φSUB) input from the EXCL pin is sampled using the clock (φ) generated by the system clock pulse generator. Clear this bit to 0 when ø is 5 MHz or more. 0: Sampling using φ/32 clock 1: Sampling using φ/4 clock
4	EXCLE	0	R/W	Subclock Input Enable Enables/disables subclock input from the EXCL pin. 0: Disables subclock input from the EXCL pin 1: Enables subclock input from the EXCL pin
3	—	0	R/W	Reserved An undefined value is read from this bit. This bit should not be set to 1.
2 to 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

19.1.3 Module Stop Control Registers H and L (MSTPCRH, MSTPCRL)

MSTPCRH and MSTPCRL 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*[1]	R/W	—
6	MSTP14	0*[1]	R/W	—
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	14-bit PWM timer (PWMX)
2	MSTP10	1*[2]	R/W	—
1	MSTP9	1*[2]	R/W	—
0	MSTP8	1	R/W	8-bit timers (TMR_X, TMR_Y)

Notes: 1. Do not set this bit to 1.

2. Do not clear this bit to 0.

- MSTPCRL

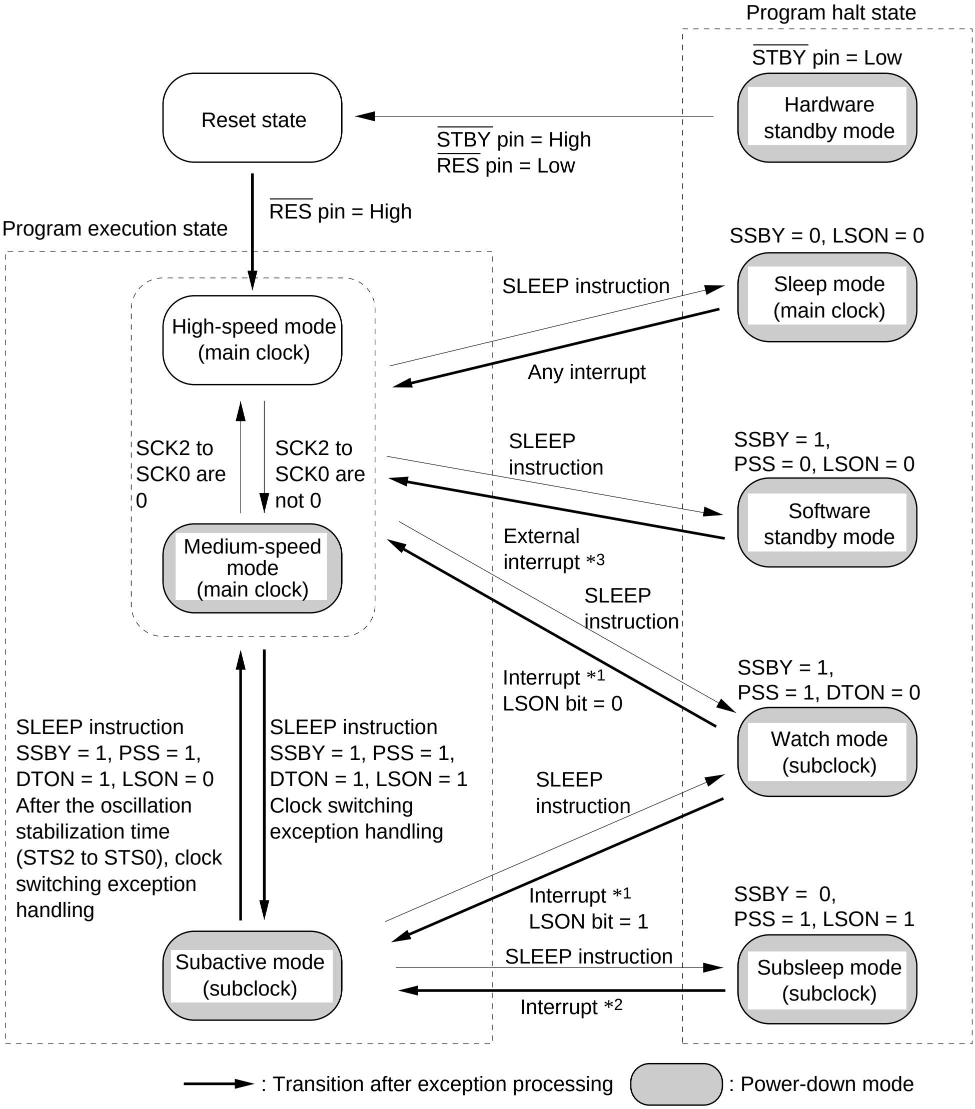
Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	MSTP7	1*[1]	R/W	—
6	MSTP6	1	R/W	Serial communication interface_1 (SCI_1)
5	MSTP5	1*[1]	R/W	—
4	MSTP4	1	R/W	I^2C bus interface_0 (IIC_0)
3	MSTP3	1	R/W	I^2C bus interface_1 (IIC_1)
2	MSTP2	1	R/W	Keyboard buffer controller, keyboard matrix interrupt mask register (KMIMR), keyboard matrix interrupt mask register A (KMIMRA), port 6 pull-up MOS control register (KMPCR)
1	MSTP1	1*[2]	R/W	—
0	MSTP0	1	R/W	Host interface (LPC), wake-up event interrupt mask register B (WUEMRB)

Notes: 1. Do not clear this bit to 0.

2. This bit can be read from or written to, however, operation is not affected.

19.2 Mode Transitions and LSI States

Figure 19.1 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 $\overline{\text{STBY}}$ input causes a mode transition from any state to hardware standby mode. The $\overline{\text{RES}}$ input causes a mode transition from a state other than hardware standby mode to the reset state. Table 19.2 shows the LSI internal states in each operating mode.


Program halt state
STBY pin = Low
Hardware standby mode
Reset state
STBY pin = High
RES pin = Low
RES pin = High
Program execution state
SSBY = 0, LSON = 0
SLEEP instruction
Sleep mode (main clock)
High-speed mode (main clock)
Any interrupt
SCK2 to SCK0 are 0
SCK2 to SCK0 are not 0
SLEEP instruction
SSBY = 1, PSS = 0, LSON = 0
Software standby mode
External interrupt *3
Medium-speed mode (main clock)
SLEEP instruction
Interrupt *1 LSON bit = 0
SSBY = 1, PSS = 1, DTON = 0
Watch mode (subclock)
SLEEP instruction SSBY = 1, PSS = 1, DTON = 1, LSON = 0 After the oscillation stabilization time (STS2 to STS0), clock switching exception handling
SLEEP instruction SSBY = 1, PSS = 1, DTON = 1, LSON = 1 Clock switching exception handling
SLEEP instruction
Interrupt *1 LSON bit = 1
SSBY = 0, PSS = 1, LSON = 1
Subactive mode (subclock)
SLEEP instruction
Subsleep mode (subclock)
Interrupt *2
: Transition after exception processing
: Power-down mode

Notes: 1. NMI, IRQ0 to IRQ2, IRQ6, IRQ7, and WDT1 interrupts
2. NMI, IRQ0 to IRQ7, WDT0, WDT1, TMR0, and TMR1 interrupts
3. NMI, IRQ0 to IRQ2, IRQ6, and IRQ7 interrupts

- When a transition is made between modes by means of an interrupt, the transition cannot be made on interrupt source generation alone. Ensure that interrupt handling is performed after accepting the interrupt request.
- Always select high-speed mode before making a transition to watch mode or sub-active mode.

Figure 19.1 Mode Transition Diagram

Table 19.2 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	Medium-speed operation	Halted	Functioning	Halted	Subclock operation	Halted	Halted	Halted
	Registers			Retained		Retained		Retained	Retained	Undefined
External interrupts	NMI	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Functioning	Halted
	IRQ0 to IRQ7									
	KIN0 to KIN15									
	WUE0 to WUE7									
Peripheral modules	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						Halted (retained)	Halted (retained)		
	TMR_X, TMR_Y									
	IIC_0									
	IIC_1									
	LPC									
	SCI_1				Functioning/Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)	

Function		High-Speed	Medium-Speed	Sleep	Module Stop	Watch	Sub-Active	Sub-Sleep	Software Standby	Hardware Standby
Peripheral modules	PWMX	Functioning	Functioning	Functioning	Functioning/Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)
	Keyboard buffer controller									
	RAM				Functioning	Retained	Functioning	Retained	Retained	Retained
	I/O				Functioning	Retained	Functioning	Functioning	Retained	High impedance

Notes: “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).

19.3 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 $\phi/32$). On-chip peripheral modules other than the bus masters always operate on the system clock (ϕ).

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, medium-speed mode is cancelled and a transition is made to hardware standby mode.

Figure 19.2 shows an example of medium-speed mode timing.

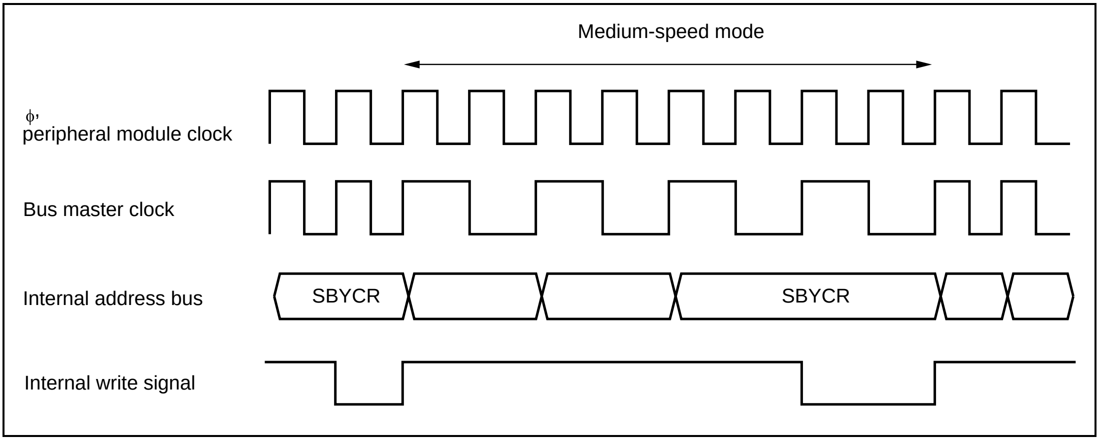


Figure 19.2 Medium-Speed Mode Timing

19.4 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, sleep mode is cancelled and a transition is made to hardware standby mode.

19.5 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 and PWMX, are retained as long as the prescribed voltage is supplied.

Software standby mode is cleared by an external interrupt (NMI, IRQ0 to IRQ2, IRQ6, or IRQ7), 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 clearing software standby mode with an IRQ0 to IRQ2, IRQ6, or IRQ7 interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ2, IRQ6, and IRQ7 is generated. Software standby mode cannot be cleared if an interrupt enable bit corresponding to an IRQ0 to IRQ2, IRQ6, or IRQ7 interrupt is cleared to 0 or if the interrupt has been masked on the CPU side.

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 19.3 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.

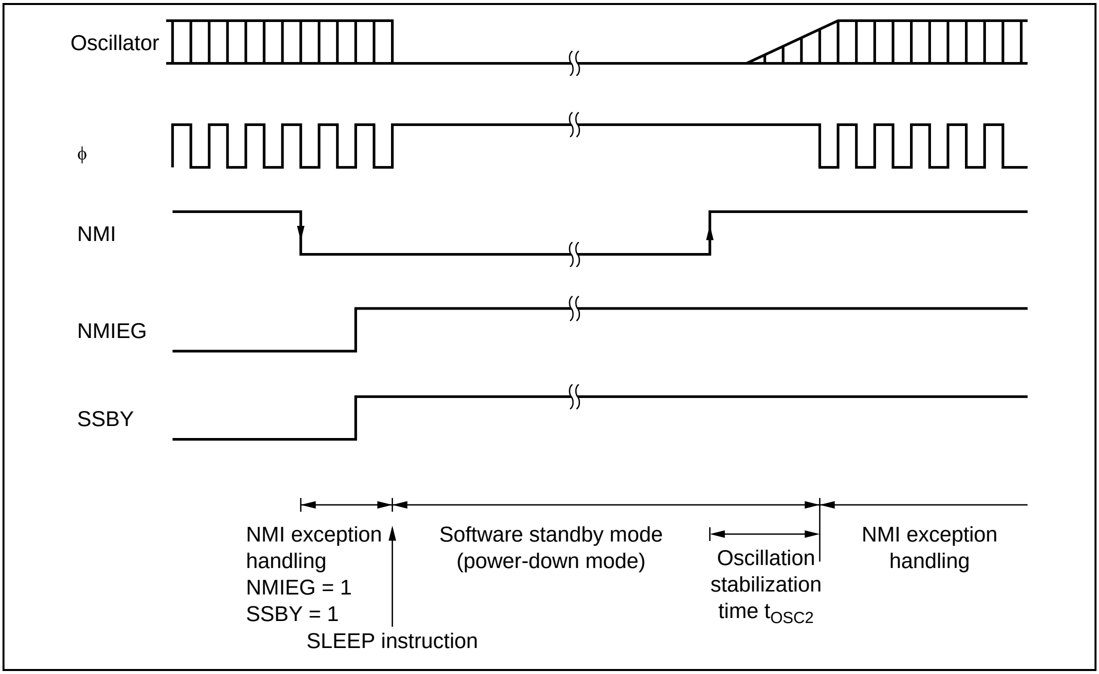


Figure 19.3 Application Example in Software Standby Mode

19.6 Hardware Standby Mode

The CPU makes a transition to hardware standby mode from any mode when the $\overline{\text{STBY}}$ 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 $\overline{\text{STBY}}$ pin low. Do not change the state of the mode pins (MD1 and MD0) while this LSI is in hardware standby mode.

Hardware standby mode is cleared by the $\overline{\text{STBY}}$ pin input or the $\overline{\text{RES}}$ pin input.

When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until system clock oscillation stabilizes. When the $\overline{\text{RES}}$ pin is subsequently driven high after the clock oscillation stabilization time has passed, reset exception handling starts.

Figure 19.4 shows an example of hardware standby mode timing.

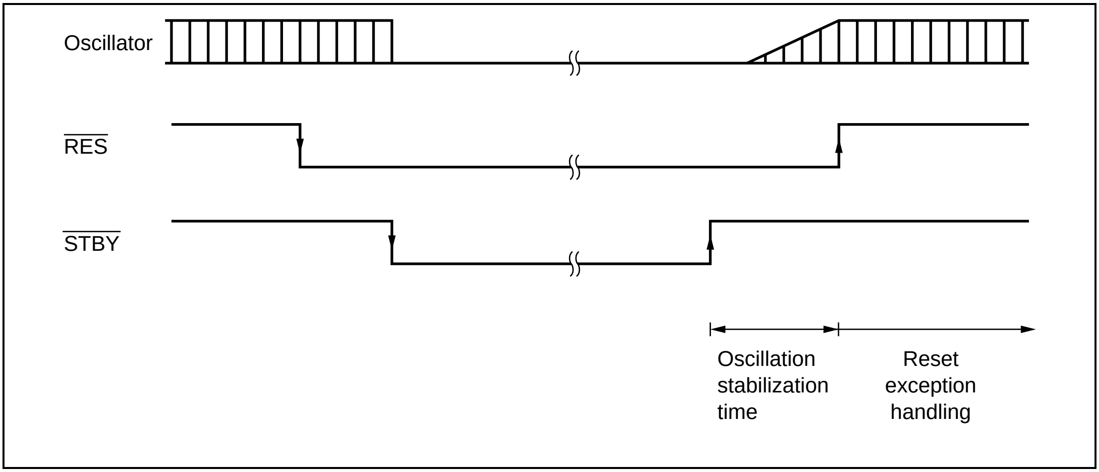


Figure 19.4 Hardware Standby Mode Timing

19.7 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, IRQ0 to IRQ2, IRQ6, or IRQ7), $\overline{\text{RES}}$ 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 IRQ0 to IRQ2, IRQ6, or IRQ7 interrupt, watch mode is not exited if the corresponding enable bit has been cleared to 0. In the case of interrupts 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.

19.8 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, IRQ0 to IRQ7), 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 IRQ0 to IRQ7 interrupt, subsleep mode is not exited if the corresponding enable bit has been cleared to 0. In the case of interrupts 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.

19.9 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, RES pin input, or 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 19.11, Direct Transitions.

When the 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 RES pin must be held low until the clock oscillation is stabilized. If the RES pin is driven high after the clock oscillation stabilization time has passed, the CPU begins reset exception handling.

If the STBY pin is driven low, the LSI enters hardware standby mode.

19.10 Module Stop Mode

Module stop mode can be individually set for each on-chip peripheral module.

When the corresponding MSTP bit in MSTPCR is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. 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 SCI and PWMX are retained.

After the reset state is cancelled, all modules are in module stop mode.

While an on-chip peripheral module is in module stop mode, read/write access to its registers is disabled.

19.11 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 LPWRCR 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 LPWRCR 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 LPWRCR set to 01, and the PSS bit in TSCR (WDT_1) set to 1.

19.12 Usage Notes

19.12.1 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.

19.12.2 Current Consumption when Waiting for Oscillation Stabilization

The current consumption increases during oscillation stabilization.

Section 20 List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register Addresses (address order)

- Registers are listed from the lower allocation addresses.
- 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.

4. Register Select Conditions

- Register states are described in the same order as the Register Addresses (address order) above.
- For details on the register select conditions, refer to section 3.2.2, System Control Register (SYSCR), 3.2.3, Serial Timer Control Register (STCR), 19.1.3, Module Stop Control Registers H and L (MSTPCRH, MSTPCRL), and the register descriptions for each module.

20.1 Register Addresses (Address Order)

The data bus width indicates the numbers of bits by which the register is accessed.

The number of access states indicates the number of states based on the specified reference clock.

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Timer XY control register*	TCRXY	8	H'FE00	TMR_X, TMR_Y	8	3
Port 7 output data register*	P7ODR	8	H'FE02	PORT	8	3
Port 7 data direction register*	P7DDR	8	H'FE03	PORT	8	3
Serial pin select register*	SPSR	8	H'FE0F	SCI_1	8	3
Bidirectional data register 0MW	TWR0MW	8	H'FE20	LPC	8	3
Bidirectional data register 0SW	TWR0SW	8	H'FE20	LPC	8	3
Bidirectional data register 1	TWR1	8	H'FE21	LPC	8	3
Bidirectional data register 2	TWR2	8	H'FE22	LPC	8	3
Bidirectional data register 3	TWR3	8	H'FE23	LPC	8	3
Bidirectional data register 4	TWR4	8	H'FE24	LPC	8	3
Bidirectional data register 5	TWR5	8	H'FE25	LPC	8	3
Bidirectional data register 6	TWR6	8	H'FE26	LPC	8	3
Bidirectional data register 7	TWR7	8	H'FE27	LPC	8	3
Bidirectional data register 8	TWR8	8	H'FE28	LPC	8	3
Bidirectional data register 9	TWR9	8	H'FE29	LPC	8	3
Bidirectional data register 10	TWR10	8	H'FE2A	LPC	8	3
Bidirectional data register 11	TWR11	8	H'FE2B	LPC	8	3
Bidirectional data register 12	TWR12	8	H'FE2C	LPC	8	3
Bidirectional data register 13	TWR13	8	H'FE2D	LPC	8	3
Bidirectional data register 14	TWR14	8	H'FE2E	LPC	8	3
Bidirectional data register 15	TWR15	8	H'FE2F	LPC	8	3
Input data register 3	IDR3	8	H'FE30	LPC	8	3
Output data register 3	ODR3	8	H'FE31	LPC	8	3
Status register 3	STR3	8	H'FE32	LPC	8	3
LPC channel address register H	LADR3H	8	H'FE34	LPC	8	3
LPC channel address register L	LADR3L	8	H'FE35	LPC	8	3
SERIRQ control register 0	SIRQCR0	8	H'FE36	LPC	8	3
SERIRQ control register 1	SIRQCR1	8	H'FE37	LPC	8	3
Input data register 1	IDR1	8	H'FE38	LPC	8	3
Output data register 1	ODR1	8	H'FE39	LPC	8	3
Status register 1	STR1	8	H'FE3A	LPC	8	3

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Input data register 2	IDR2	8	H'FE3C	LPC	8	3
Output data register 2	ODR2	8	H'FE3D	LPC	8	3
Status register 2	STR2	8	H'FE3E	LPC	8	3
Host interface select register	HISEL	8	H'FE3F	LPC	8	3
Host interface control register 0	HICR0	8	H'FE40	LPC	8	3
Host interface control register 1	HICR1	8	H'FE41	LPC	8	3
Host interface control register 2	HICR2	8	H'FE42	LPC	8	3
Host interface control register 3	HICR3	8	H'FE43	LPC	8	3
Wakeup event interrupt mask register B	WUEMRB	8	H'FE44	INT	8	3
I^{2}C bus extended control register_0	ICXR_0	8	H'FED4	IIC_0	8	2
I^{2}C bus extended control register_1	ICXR_1	8	H'FED5	IIC_1	8	2
Keyboard control register H_0	KBCRH_0	8	H'FED8	Keyboard buffer controller _0	8	2
Keyboard control register L_0	KBCRL_0	8	H'FED9	Keyboard buffer controller _0	8	2
Keyboard data buffer register_0	KBBR_0	8	H'FEDA	Keyboard buffer controller _0	8	2
Keyboard control register H_1	KBCRH_1	8	H'FEDC	Keyboard buffer controller _1	8	2
Keyboard control register L_1	KBCRL_1	8	H'FEDD	Keyboard buffer controller _1	8	2
Keyboard data buffer register_1	KBBR_1	8	H'FEDE	Keyboard buffer controller _1	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Keyboard control register H_2	KBCRH_2	8	H'FEE0	Keyboard buffer controller_ 2	8	2
Keyboard control register L_2	KBCRL_2	8	H'FEE1	Keyboard buffer controller_ 2	8	2
Keyboard data buffer register_2	KBBR_2	8	H'FEE2	Keyboard buffer controller_ 2	8	2
DDC switch register	DDCSWR	8	H'FEE6	IIC_0	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	ISCRL	8	H'FEED	INT	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
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 register 1	EBR1	8	H'FF82	FLASH	8	2
System control register 2	SYSCR2	8	H'FF83	SYSTEM	8	2
Erase block register 2	EBR2	8	H'FF83	FLASH	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
Serial mode register_1	SMR_1	8	H'FF88	SCI_1	8	2
I^{2}C bus control register_1	ICCR_1	8	H'FF88	IIC_1	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Bit rate register_1	BRR_1	8	H'FF89	SCI_1	8	2
I^{2}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^{2}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^{2}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
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

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
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
PWM (D/A) control register	DACR	8	H'FFA0	PWMX	8	2
PWM (D/A) data register AH	DADRAH	8	H'FFA0	PWMX	8	2
PWM (D/A) data register AL	DADRAL	8	H'FFA1	PWMX	8	2
PWM (D/A) counter H	DACNTH	8	H'FFA6	PWMX	8	2
PWM (D/A) data register BH	DADRBH	8	H'FFA6	PWMX	8	2
PWM (D/A) counter L	DACNTL	8	H'FFA7	PWMX	8	2
PWM (D/A) data register BL	DADRBL	8	H'FFA7	PWMX	8	2
Timer control/status register_0	TCSR_0	8	H'FFA8	WDT_0	8	2
Timer counter_0	TCNT_0	8	H'FFA8 (write)	WDT_0	8	2
Timer counter_0	TCNT_0	8	H'FFA9 (read)	WDT_0	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
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

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Port 6 data register	P6DR	8	H'FFBB	PORT	8	2
Port B output data register	PBODR	8	H'FFBC	PORT	8	2
Port B input data register	PBPIN	8	H'FFBD (read)	PORT	8	2
Port 8 data direction register	P8DDR	8	H'FFBD (write)	PORT	8	2
Port 7 input data register	P7PIN	8	H'FFBE (read)	PORT	8	2
Port B data direction register	PBDDR	8	H'FFBE (write)	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
Interrupt 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	8	2
Timer control register_1	TCR_1	8	H'FFC9	TMR_1	8	2
Timer control/status register_0	TCSR_0	8	H'FFCA	TMR_0	8	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
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
I^{2}C bus control register_0	ICCR_0	8	H'FFD8	IIC_0	8	2
I^{2}C bus status register_0	ICSR_0	8	H'FFD9	IIC_0	8	2
I^{2}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

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
I^{2}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
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
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
Keyboard matrix interrupt register 6	KMIMR	8	H'FFF1	INT	8	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
Pull-up MOS control register	KMPCR	8	H'FFF2	PORT	8	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
Keyboard matrix interrupt register A	KMIMRA	8	H'FFF3	INT	8	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
Timer counter_X	TCNT_X	8	H'FFF4	TMR_X	16	2
Timer counter_Y	TCNT_Y	8	H'FFF4	TMR_Y	16	2
Timer constant register C	TCORC	8	H'FFF5	TMR_X	16	2
Timer input select register	TISR	8	H'FFF5	TMR_Y	16	2
Timer constant register A_X	TCORA_X	8	H'FFF6	TMR_X	16	2
Timer constant register B_X	TCORB_X	8	H'FFF7	TMR_X	16	2
Timer connection register I	TCONRI	8	H'FFFC	TMR_X	8	2
Timer connection register S	TCONRS	8	H'FFFE	TMR_Y	8	2

Note:* The program development tool (emulator) does not support this register.

20.2 Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

16-bit registers are shown as 2 lines.

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
TCRXY*[3]	OSX	OEY	CKSX	CKSY	—	—	—	—	TMR_X, TMR_Y
P7ODR*[3]	P77ODR	P76ODR	P75ODR	P74ODR	P73ODR	P72ODR	P71ODR	P70ODR	PORT
P7DDR*[3]	P77DDR	P76DDR	P75DDR	P74DDR	P73DDR	P72DDR	P71DDR	P70DDR	
SPSR*[3]	SPS1	—	—	—	—	—	—	—	SCI_1
TWR0MW	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	LPC
TWR0SW	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR4	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR5	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR6	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR7	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR9	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR10	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR11	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR12	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR13	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR14	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TWR15	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
IDR3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ODR3	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
STR3*[1]	IBF3B	OBF3B	MWMF	SWMF	C/$\overline{\text{D}}$3	DBU32	IBF3A	OBF3A	
STR3*[2]	DBU37	DBU36	DBU35	DBU34	C/$\overline{\text{D}}$3	DBU32	IBF3A	OBF3A	
LADR3H	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	
LADR3L	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	—	Bit 1	TWRE	

Register									
Abbreviation	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Module**
SIRQCR0	Q/$\overline{C}$	SELREQ	IEDIR	SMIE3B	SMIE3A	SMIE2	IRQ12E1	IRQ1E1	LPC
SIRQCR1	IRQ11E3	IRQ10E3	IRQ9E3	IRQ6E3	IRQ11E2	IRQ10E2	IRQ9E2	IRQ6E2	
IDR1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ODR1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
STR1	DBU17	DBU16	DBU15	DBU14	C/$\overline{D}$1	DBU12	IBF1	OBF1	
IDR2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
ODR2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
STR2	DBU27	DBU26	DBU25	DBU24	C/$\overline{D}$2	DBU22	IBF2	OBF2	
HISEL	SELSTR3	SELIRQ11	SELIRQ10	SELIRQ9	SELIRQ6	SELSMI	SELIRQ12	SELIRQ1	
HICR0	LPC3E	LPC2E	LPC1E	FGA20E	SDWNE	PMEE	LSMIE	LSCIE	
HICR1	LPCBSY	CLKREQ	IRQBSY	LRSTB	SDWNB	PMEB	LSMIB	LSCIB	
HICR2	GA20	LRST	SDWN	ABRT	IBFIE3	IBFIE2	IBFIE1	ERRIE	
HICR3	LFRAME	CLKRUN	SERIRQ	LRESET	LPCPD	PME	LSMI	LSCI	
WUEMRB	WUEMR7	WUEMR6	WUEMR5	WUEMR4	WUEMR3	WUEMR2	WUEMR1	WUEMR0	INT
ICXR_0	STOPIM	HNDS	ICDRF	ICDRE	ALIE	ALSL	FNC1	FNC0	IIC_0
ICXR_1	STOPIM	HNDS	ICDRF	ICDRE	ALIE	ALSL	FNC1	FNC0	IIC_1
KBCRH_0	KBIOE	KCLKI	KDI	KBFSEL	KBIE	KBF	PER	KBS	Keyboard buffer controller _0
KBCRL_0	KBE	KCLKO	KDO	—	RXCR3	RXCR2	RXCR1	RXCR0	
KBBR_0	KB7	KB6	KB5	KB4	KB3	KB2	KB1	KB0	
KBCRH_1	KBIOE	KCLKI	KDI	KBFSEL	KBIE	KBF	PER	KBS	Keyboard buffer controller _1
KBCRL_1	KBE	KCLKO	KDO	—	RXCR3	RXCR2	RXCR1	RXCR0	
KBBR_1	KB7	KB6	KB5	KB4	KB3	KB2	KB1	KB0	
KBCRH_2	KBIOE	KCLKI	KDI	KBFSEL	KBIE	KBF	PER	KBS	Keyboard buffer controller _2
KBCRL_2	KBE	KCLKO	KDO	—	RXCR3	RXCR2	RXCR1	RXCR0	
KBBR_2	KB7	KB6	KB5	KB4	KB3	KB2	KB1	KB0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DDCSWR	—	—	—	—	CLR3	CLR2	CLR1	CLR0	IIC_0
ICRA	ICRA7	ICRA6	ICRA5	ICRA4	ICRA3	ICRA2	ICRA1	ICRA0	INT
ICRB	ICRB7	ICRB6	ICRB5	ICRB4	ICRB3	ICRB2	ICRB1	ICRB0	
ICRC	ICRC7	ICRC6	ICRC5	ICRC4	ICRC3	ICRC2	ICRC1	ICRC0	
ISR	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
ISCRH	IRQ7SCB	IRQ7SCA	IRQ6SCB	IRQ6SCA	IRQ5SCB	IRQ5SCA	IRQ4SCB	IRQ4SCA	
ISCRL	IRQ3SCB	IRQ3SCA	IRQ2SCB	IRQ2SCA	IRQ1SCB	IRQ1SCA	IRQ0SCB	IRQ0SCA	
ABRKCR	CMF	—	—	—	—	—	—	BIE	
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	—	
FLMCR1	FWE	SWE	—	—	EV	PV	E	P	FLASH
FLMCR2	FLER	—	—	—	—	—	ESU	PSU	
EBR1	—	—	—	—	—	—	—	—	
SYSCR2	KWUL1	KWUL0	P6PUE	—	SDE	CS4E	CS3E	HI12E	SYSTEM
EBR2	EB7	EB6	EB5	EB4	EB3	EB2	EB1	EB0	FLASH
SBYCR	SSBY	STS2	STS1	STS0	—	SCK2	SCK1	SCK0	SYSTEM
LPWRCR	DTON	LSON	NESEL	EXCLE	—	—	—	—	
MSTPCRH	MSTP15	MSTP14	MSTP13	MSTP12	MSTP11	MSTP10	MSTP9	MSTP8	
MSTPCRL	MSTP7	MSTP6	MSTP5	MSTP4	MSTP3	MSTP2	MSTP1	MSTP0	
SMR_1	C/$\overline{\text{A}}$	CHR	PE	O/$\overline{\text{E}}$	STOP	MP	CKS1	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	RDRF	ORER	FER	PER	TEND	MPB	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	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
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	
OCRAL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
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	
DACR	TEST	PWME	—	—	OEB	OEA	OS	CKS	PWMX
DADRAH	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	
DADRAL	DA5	DA4	DA3	DA2	DA1	DA0	CFS	—	
DACNTH	UC7	UC6	UC5	UC4	UC3	UC2	UC1	UC0	
DADRBH	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	
DACNTL	UC8	UC9	UC10	UC11	UC12	UC13	—	REGS	
DADRBL	DA5	DA4	DA3	DA2	DA1	DA0	CFS	REGS	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
TCSR_0	OVF	WT/$\overline{\text{IT}}$	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	PA7ODR	PA6ODR	PA5ODR	PA4ODR	PA3ODR	PA2ODR	PA1ODR	PA0ODR	PORT
PAPIN	PA7PIN	PA6PIN	PA5PIN	PA4PIN	PA3PIN	PA2PIN	PA1PIN	PA0PIN	
PADDR	PA7DDR	PA6DDR	PA5DDR	PA4DDR	PA3DDR	PA2DDR	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	—	—	—	—	—	P52DDR	P51DDR	P50DDR	
P6DDR	P67DDR	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	
P5DR	—	—	—	—	—	P52DR	P51DR	P50DR	
P6DR	P67DR	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
PBODR	PB7ODR	PB6ODR	PB5ODR	PB4ODR	PB3ODR	PB2ODR	PB1ODR	PB0ODR	
PBPIN	PB7PIN	PB6PIN	PB5PIN	PB4PIN	PB3PIN	PB2PIN	PB1PIN	PB0PIN	
P8DDR	—	P86DDR	P85DDR	P84DDR	P83DDR	P82DDR	P81DDR	P80DDR	
P7PIN	P77PIN	P76PIN	P75PIN	P74PIN	P73PIN	P72PIN	P71PIN	P70PIN	
PBDDR	PB7DDR	PB6DDR	PB5DDR	PB4DDR	PB3DDR	PB2DDR	PB1DDR	PB0DDR	
P8DR	—	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	IICS	IICX1	IICX0	IICE	FLSHE	—	ICKS1	ICKS0	SYSTEM
SYSCR	—	—	INTM1	INTM0	XRST	NMIEG	HIE	RAME	
MDCR	EXPE	—	—	—	—	—	MDS1	MDS0	
BCR	—	ICIS0	BRSTRM	BRSTS1	BRSTS0	—	IOS1	IOS0	BSC
WSCR	—	—	ABW	AST	WMS1	WMS0	WC1	WC0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
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	
ICCR_0	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_0
ICSR_0	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	
ICDR_0	ICDR7	ICDR6	ICDR5	ICDR4	ICDR3	ICDR2	ICDR1	ICDR0	
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	
TCSR_1	OVF	WT/$\overline{\text{IT}}$	TME	PSS	RST/$\overline{\text{NMI}}$	CKS2	CKS1	CKS0	WDT_1
TCNT_1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCR_X	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_X
TCR_Y	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_Y
KMIMR	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0	INT
TCSR_X	CMFB	CMFA	OVF	ICF	OS3	OS2	OS1	OS0	TMR_X
TCSR_Y	CMFB	CMFA	OVF	ICIE	OS3	OS2	OS1	OS0	TMR_Y
KMPCR	KMIMR7	KMIMR6	KMIMR5	KMIMR4	KMIMR3	KMIMR2	KMIMR1	KMIMR0	PORT
TICRR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_X
TCORA_Y	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_Y
KMIMRA	KMIMR15	KMIMR14	KMIMR13	KMIMR12	KMIMR11	KMIMR10	KMIMR9	KMIMR8	INT
TICRF	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_X
TCORB_Y	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_Y
TCNT_X	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_X
TCNT_Y	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_Y
TCORC	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_X
TISR	—	—	—	—	—	—	—	IS	TMR_Y

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
TCORA_X	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	TMR_X
TCORB_X	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
TCONRI	—	—	—	ICST	—	—	—	—	
TCONRS	TMRX/Y	—	—	—	—	—	—	—	TMR_Y

Notes: 1. When TWRE = 1 or SELSTR3 = 0 in LADR3L

2. When TWRE = 0 and SELSTR3 = 1 in LADR3L

3. The program development tool (emulator) does not support this register.

20.3 Register States in Each Operating Mode

Register Abbrevia-tion	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
TCRXY*	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X, TMR_Y
P7ODR*	Initialized	—	—	—	—	—	—	—	Initialized	PORT
P7DDR*	Initialized	—	—	—	—	—	—	—	Initialized	
SPSR*	Initialized	—	—	—	—	—	—	—	Initialized	SCI_1
TWR0MW	—	—	—	—	—	—	—	—	—	LPC
TWR0SW	—	—	—	—	—	—	—	—	—	
TWR1	—	—	—	—	—	—	—	—	—	
TWR2	—	—	—	—	—	—	—	—	—	
TWR3	—	—	—	—	—	—	—	—	—	
TWR4	—	—	—	—	—	—	—	—	—	
TWR5	—	—	—	—	—	—	—	—	—	
TWR6	—	—	—	—	—	—	—	—	—	
TWR7	—	—	—	—	—	—	—	—	—	
TWR8	—	—	—	—	—	—	—	—	—	
TWR9	—	—	—	—	—	—	—	—	—	
TWR10	—	—	—	—	—	—	—	—	—	
TWR11	—	—	—	—	—	—	—	—	—	
TWR12	—	—	—	—	—	—	—	—	—	
TWR13	—	—	—	—	—	—	—	—	—	
TWR14	—	—	—	—	—	—	—	—	—	
TWR15	—	—	—	—	—	—	—	—	—	
IDR3	—	—	—	—	—	—	—	—	—	
ODR3	—	—	—	—	—	—	—	—	—	
STR3	Initialized	—	—	—	—	—	—	—	Initialized	
LADR3H	Initialized	—	—	—	—	—	—	—	Initialized	
LADR3L	Initialized	—	—	—	—	—	—	—	Initialized	
SIRQCR0	Initialized	—	—	—	—	—	—	—	Initialized	
SIRQCR1	Initialized	—	—	—	—	—	—	—	Initialized	
IDR1	—	—	—	—	—	—	—	—	—	
ODR1	—	—	—	—	—	—	—	—	—	

Register		High-Speed/								
Abbrevia-		Medium-			Sub-		Module	Software	Hardware	
tion	Reset	Speed	Watch	Sleep	Active	Sub-Sleep	Stop	Standby	Standby	Module
STR1	Initialized	—	—	—	—	—	—	—	Initialized	LPC
IDR2	—	—	—	—	—	—	—	—	—	
ODR2	—	—	—	—	—	—	—	—	—	
STR2	Initialized	—	—	—	—	—	—	—	Initialized	
HISEL	Initialized	—	—	—	—	—	—	—	Initialized	
HICR0	Initialized	—	—	—	—	—	—	—	Initialized	
HICR1	Initialized	—	—	—	—	—	—	—	Initialized	
HICR2	Initialized	—	—	—	—	—	—	—	Initialized	
HICR3	—	—	—	—	—	—	—	—	—	
WUEMRB	Initialized	—	—	—	—	—	—	—	Initialized	INT
ICXR_0	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
ICXR_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
KBCRH_0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	Keyboard buffer controller_0
KBCRL_0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
KBBR_0	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
KBCRH_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	Keyboard buffer controller_1
KBCRL_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
KBBR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
KBCRH_2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	Keyboard buffer controller_2
KBCRL_2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
KBBR_2	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DDCSWR	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
ICRA	Initialized	—	—	—	—	—	—	—	Initialized	INT
ICRB	Initialized	—	—	—	—	—	—	—	Initialized	
ICRC	Initialized	—	—	—	—	—	—	—	Initialized	
ISR	Initialized	—	—	—	—	—	—	—	Initialized	
ISCRH	Initialized	—	—	—	—	—	—	—	Initialized	
ISCRL	Initialized	—	—	—	—	—	—	—	Initialized	
ABRKCR	Initialized	—	—	—	—	—	—	—	Initialized	
BARA	Initialized	—	—	—	—	—	—	—	Initialized	
BARB	Initialized	—	—	—	—	—	—	—	Initialized	
BARC	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbrevia-tion	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
FLMCR1	Initialized	—	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	FLASH
FLMCR2	Initialized	—	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	
EBR1	Initialized	—	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	FLASH
SYSCR2	Initialized	—	—	—	—	—	—	—	Initialized	SYSTEM
EBR2	Initialized	—	Initialized	—	Initialized	Initialized	—	Initialized	Initialized	FLASH
SBYCR	Initialized	—	—	—	—	—	—	—	Initialized	SYSTEM
LPWRCR	Initialized	—	—	—	—	—	—	—	Initialized	
MSTPCRH	Initialized	—	—	—	—	—	—	—	Initialized	
MSTPCRL	Initialized	—	—	—	—	—	—	—	Initialized	
SMR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SCI_1
ICCR_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
BRR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SCI_1
ICSR_1	Initialized	—	—	—	—	—	—	—	Initialized	IIC_1
SCR_1	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	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	—	Initialized	Initialized	Initialized	Initialized	Initialized	
ICDR_1	—	—	—	—	—	—	—	—	—	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	
OCRAH	Initialized	—	—	—	—	—	—	—	Initialized	
OCRBH	Initialized	—	—	—	—	—	—	—	Initialized	
OCRAL	Initialized	—	—	—	—	—	—	—	Initialized	
OCRBL	Initialized	—	—	—	—	—	—	—	Initialized	
TCR	Initialized	—	—	—	—	—	—	—	Initialized	
TOCR	Initialized	—	—	—	—	—	—	—	Initialized	
ICRAH	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
OCRARH	Initialized	—	—	—	—	—	—	—	Initialized	FRT
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	
DACR	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWMX
DADRAH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DADRAL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DACNTH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DADRBH	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DACNTL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
DADRBL	Initialized	—	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
TCSR_0	Initialized	—	—	—	—	—	—	—	Initialized	WDT_0
TCNT_0	Initialized	—	—	—	—	—	—	—	Initialized	
PAODR	Initialized	—	—	—	—	—	—	—	Initialized	PORT
PAPIN	—	—	—	—	—	—	—	—	—	
PADDR	Initialized	—	—	—	—	—	—	—	Initialized	
P1PCR	Initialized	—	—	—	—	—	—	—	Initialized	
P2PCR	Initialized	—	—	—	—	—	—	—	Initialized	
P3PCR	Initialized	—	—	—	—	—	—	—	Initialized	PORT
P1DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P2DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P1DR	Initialized	—	—	—	—	—	—	—	Initialized	
P2DR	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbrevia-tion	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
P3DDR	Initialized	—	—	—	—	—	—	—	Initialized	PORT
P4DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P3DR	Initialized	—	—	—	—	—	—	—	Initialized	
P4DR	Initialized	—	—	—	—	—	—	—	Initialized	
P5DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P6DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P5DR	Initialized	—	—	—	—	—	—	—	Initialized	
P6DR	Initialized	—	—	—	—	—	—	—	Initialized	
PBODR	Initialized	—	—	—	—	—	—	—	Initialized	
PBPIN	—	—	—	—	—	—	—	—	—	
P8DDR	Initialized	—	—	—	—	—	—	—	Initialized	
P7PIN	—	—	—	—	—	—	—	—	—	
PBDDR	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	
TCORB_0	Initialized	—	—	—	—	—	—	—	Initialized	TMR_0, TMR_1
TCORB_1	Initialized	—	—	—	—	—	—	—	Initialized	
TCNT_0	Initialized	—	—	—	—	—	—	—	Initialized	
TCNT_1	Initialized	—	—	—	—	—	—	—	Initialized	

Register Abbrevia-tion	Reset	High-Speed/ Medium-Speed	Watch	Sleep	Sub-Active	Sub-Sleep	Module Stop	Software Standby	Hardware Standby	Module
ICCR_0	Initialized	—	—	—	—	—	—	—	Initialized	IIC_0
ICSR_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICDR_0	—	—	—	—	—	—	—	—	—	
SARX_0	Initialized	—	—	—	—	—	—	—	Initialized	
ICMR_0	Initialized	—	—	—	—	—	—	—	Initialized	
SAR_0	Initialized	—	—	—	—	—	—	—	Initialized	
TCSR_1	Initialized	—	—	—	—	—	—	—	Initialized	WDT_1
TCNT_1	Initialized	—	—	—	—	—	—	—	Initialized	
TCR_X	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TCR_Y	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y
KMIMR	Initialized	—	—	—	—	—	—	—	Initialized	INT
TCSR_X	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TCSR_Y	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y
KMPCR	Initialized	—	—	—	—	—	—	—	Initialized	PORT
TICRR	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TCORA_Y	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y
KMIMRA	Initialized	—	—	—	—	—	—	—	Initialized	INT
TICRF	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TCORB_Y	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y
TCNT_X	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TCNT_Y	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y
TCORC	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TISR	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y
TCORA_X	Initialized	—	—	—	—	—	—	—	Initialized	TMR_X
TCORB_X	Initialized	—	—	—	—	—	—	—	Initialized	
TCONRI	Initialized	—	—	—	—	—	—	—	Initialized	
TCONRS	Initialized	—	—	—	—	—	—	—	Initialized	TMR_Y

Note:* The program development tool (emulator) does not support this register.

20.4 Register Select Conditions

Lower Address	Register Name	Register Select Condition	Module Name
H'FE00*[2]	TCRXY	No condition	TMR_X, TMR_Y
H'FE02*[2]	P7ODR	No condition	PORT
H'FE03*[2]	P7DDR		
H'FE0F*[2]	SPSR	No condition	SCI_1
H'FE20	TWR0MW	MSTP = 0, (HI12E = 0)*[1]	LPC
	TWR0SW		
H'FE21	TWR1		
H'FE22	TWR2		
H'FE23	TWR3		
H'FE24	TWR4		
H'FE25	TWR5		
H'FE26	TWR6		
H'FE27	TWR7		
H'FE28	TWR8		
H'FE29	TWR9		
H'FE2A	TWR10		
H'FE2B	TWR11		
H'FE2C	TWR12		
H'FE2D	TWR13		
H'FE2E	TWR14		
H'FE2F	TWR15		
H'FE30	IDR3		
H'FE31	ODR3		
H'FE32	STR3		
H'FE34	LADR3H		
H'FE35	LADR3L		
H'FE36	SIRQCR0		
H'FE37	SIRQCR1		
H'FE38	IDR1		
H'FE39	ODR1		
H'FE3A	STR1		
H'FE3C	IDR2		

Lower Address	Register Name	Register Select Condition	Module Name
H'FE3D	ODR2	MSTP = 0, (HI12E = 0)*[1]	LPC
H'FE3E	STR2		
H'FE3F	HISEL		
H'FE40	HICR0		
H'FE41	HICR1		
H'FE42	HICR2		
H'FE43	HICR3		
H'FE44	WUEMRB	No condition	INT
H'FED4	ICXR_0	No condition	IIC_0
H'FED5	ICXR_1		IIC_1
H'FED8	KBCRH_0	MSTP2 = 0	Keyboard buffer controller
H'FED9	KBCRL_0		
H'FEDA	KBBR_0		
H'FEDC	KBCRH_1		
H'FEDD	KBCRL_1		
H'FEDE	KBBR_1		
H'FEE0	KBCRH_2		
H'FEE1	KBCRL_2		
H'FEE2	KBBR_2		
H'FEE6	DDCSWR	MSTP4 = 0	IIC_0
H'FEE8	ICRA	No condition	INT
H'FEE9	ICRB		
H'FEEA	ICRC		
H'FEEB	ISR		
H'FEEC	ISCRH		
H'FEED	ISCRL		
H'FEF4	ABRKCR		
H'FEF5	BARA		
H'FEF6	BARB		
H'FEF7	BARC		

Lower Address	Register Name	Register Select Condition		Module Name
H'FF80	FLMCR1	FLSHE = 1 in STCR		FLASH
H'FF81	FLMCR2			
H'FF82	EBR1			
H'FF83	SYSCR2	FLSHE = 0 in STCR		SYSTEM
	EBR2	FLSHE = 1 in STCR		FLASH
H'FF84	SBYCR	FLSHE = 0 in STCR		SYSTEM
H'FF85	LPWRCR			
H'FF86	MSTPCRH			
H'FF87	MSTPCRL			
H'FF88	SMR_1	MSTP6 = 0, IICE = 0 in STCR		SCI_1
	ICCR_1	MSTP3 = 0, IICE = 1 in STCR		IIC_1
H'FF89	BRR_1	MSTP6 = 0, IICE = 0 in STCR		SCI_1
	ICSR_1	MSTP3 = 0, IICE = 1 in STCR		IIC_1
H'FF8A	SCR_1	MSTP6 = 0		SCI_1
H'FF8B	TDR_1			
H'FF8C	SSR_1			
H'FF8D	RDR_1			
H'FF8E	SCMR_1	MSTP6 = 0, IICE = 0 in STCR		
	ICDR_1	MSTP3 = 0, IICE = 1 in STCR	ICE = 1 in ICCR1	IIC_1
	SARX_1		ICE = 0 in ICCR1	
H'FF8F	ICMR_1		ICE = 1 in ICCR1	
	SAR_1		ICE = 0 in ICCR1	
H'FF90	TIER	MSTP13 = 0		FRT
H'FF91	TCSR			
H'FF92	FRCH			
H'FF93	FRCL			
H'FF94	OCRAH		OCRS = 0 in TOCR	
	OCRBH		OCRS = 1 in TOCR	
H'FF95	OCRAL		OCRS = 0 in TOCR	
	OCRBL		OCRS = 1 in TOCR	
H'FF96	TCR			
H'FF97	TOCR			
H'FF98	ICRAH		ICRS = 0 in TOCR	
	OCRARH		ICRS = 1 in TOCR	

Lower Address	Register Name	Register Select Condition		Module Name
H'FF99	ICRAL	MSTP13 = 0	ICRS = 0 in TOCR	FRT
	OCRARL		ICRS = 1 in TOCR	
H'FF9A	ICRBH		ICRS = 0 in TOCR	
	OCRAFH		ICRS = 1 in TOCR	
H'FF9B	ICRBL		ICRS = 0 in TOCR	
	OCRAFL		ICRS = 1 in TOCR	
H'FF9C	ICRCH		ICRS = 0 in TOCR	
	OCRDMH		ICRS = 1 in TOCR	
H'FF9D	ICRCL		ICRS = 0 in TOCR	
	OCRDML		ICRS = 1 in TOCR	
H'FF9E	ICRDH			
H'FF9F	ICRDL			
H'FFA0	DACR	MSTP11 = 0, IICE = 1 in STCR	REGS = 1 in DACNT/ DADRB	PWMX
	DADRAH		REGS = 0 in DACNT/ DADRB	
H'FFA1	DADRAL	MSTP11 = 0, IICE = 1 in STCR	REGS = 0 in DACNT/ DADRB	
H'FFA6	DACNTH	MSTP11 = 0, IICE = 1 in STCR	REGS = 1 in DACNT/ DADRB	PWMX
	DADRBH		REGS = 0 in DACNT/ DADRB	
H'FFA7	DACNTL		REGS = 1 in DACNT/ DADRB	
	DADRBL		REGS = 0 in DACNT/ DADRB	
H'FFA8	TCSR_0	No condition		WDT_0
	TCNT_0 (write)			
H'FFA9	TCNT_0 (read)			

Lower Address	Register Name	Register Select Condition	Module Name
H'FFAA	PAODR	No condition	PORT
H'FFAB	PAPIN (read)		
	PADDR (write)		
H'FFAC	P1PCR		
H'FFAD	P2PCR		
H'FFAE	P3PCR		
H'FFB0	P1DDR		
H'FFB1	P2DDR		
H'FFB2	P1DR		
H'FFB3	P2DR		
H'FFB4	P3DDR		
H'FFB5	P4DDR		
H'FFB6	P3DR		
H'FFB7	P4DR		
H'FFB8	P5DDR		
H'FFB9	P6DDR		
H'FFBA	P5DR		
H'FFBB	P6DR		
H'FFBC	PBODR		
H'FFBD	PBPIN (read)		
	P8DDR (write)		
H'FFBE	P7PIN (read)		
	PBDDR (write)		
H'FFBF	P8DR		
H'FFC0	P9DDR		
H'FFC1	P9DR		
H'FFC2	IER	No condition	INT
H'FFC3	STCR	No condition	SYSTEM
H'FFC4	SYSCR		
H'FFC5	MDCR		
H'FFC6	BCR	No condition	BSC
H'FFC7	WSCR		

Lower Address	Register Name	Register Select Condition		Module Name
H'FFC8	TCR_0	MSTP12 = 0		TMR_0, TMR_1
H'FFC9	TCR_1			
H'FFCA	TCSR_0			
H'FFCB	TCSR_1			
H'FFCC	TCORA_0			
H'FFCD	TCORA_1			
H'FFCE	TCORB_0			
H'FFCF	TCORB_1			
H'FFD0	TCNT_0			
H'FFD1	TCNT_1			
H'FFD8	ICCR_0	MSTP4 = 0, IICE = 1 in STCR		IIC_0
H'FFD9	ICSR_0			
H'FFDE	ICDR_0	MSTP4 = 0, IICE = 1 in STCR	ICE = 1 in ICCR0	
	SARX_0		ICE = 0 in ICCR0	
H'FFDF	ICMR_0		ICE = 1 in ICCR0	
	SAR_0		ICE = 0 in ICCR0	
H'FFEA	TCSR_1	No condition		WDT_1
	TCNT_1 (write)			
H'FFEB	TCNT_1 (read)			
H'FFF0	TCR_X	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
	TCR_Y		TMRX/Y = 1 in TCONRS	TMR_Y
H'FFF1	KMIMR	MSTP2 = 0, HIE = 1 in SYSCR		INT
	TCSR_X	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
	TCSR_Y		TMRX/Y = 1 in TCONRS	TMR_Y
H'FFF2	KMPCR	MSTP2 = 0, HIE = 1 in SYSCR		PORT
	TICRR	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
	TCORA_Y		TMRX/Y = 1 in TCONRS	TMR_Y

Lower Address	Register Name	Register Select Condition		Module Name
H'FFF3	KMIMRA	MSTP2 = 0, HIE = 1 in SYSCR		INT
	TICRF	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
	TCORB_Y		TMRX/Y = 1 in TCONRS	TMR_Y
H'FFF4	TCNT_X	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
	TCNT_Y		TMRX/Y = 1 in TCONRS	TMR_Y
H'FFF5	TCORC	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
	TISR		TMRX/Y = 1 in TCONRS	TMR_Y
H'FFF6	TCORA_X	MSTP8 = 0, HIE = 0 in SYSCR	TMRX/Y = 0 in TCONRS	TMR_X
H'FFF7	TCORB_X			
H'FFFC	TCONRI	MSTP8 = 0, HIE = 0 in SYSCR		TMR_X
H'FFFE	TCONRS	MSTP8 = 0, HIE = 0 in SYSCR		TMR_Y

Notes: 1. Although the settings of the HI12E bit in SYSCR2 do not affect the LPC operation, this bit must not be set to 1 according to the limitation depending on the program development tool (emulator) configuration.

2. The program development tool (emulator) does not support this register.

Section 21 Electrical Characteristics

21.1 Electrical Characteristics

21.1.1 Absolute Maximum Ratings

Table 21.1 lists the absolute maximum ratings.

Table 21.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage	V_{CC}, V_{CL}	–0.3 to +4.3	V
I/O buffer power supply voltage	$V_{CC}B$	–0.3 to +7.0	V
Input voltage (except ports A, P97, P86, P52, and P42)	V_{in}	–0.3 to V_{CC} +0.3	V
Input voltage (Port A)	V_{in}	–0.3 to $V_{CC}B$ +0.3	V
Input voltage (P97, P86, P52, P42)	V_{in}	–0.3 to +7.0	V
Operating temperature	T_{opr}	–20 to +75	°C
Operating temperature (flash memory programming/erasing)	T_{opr}	–20 to +75	°C
Storage temperature	T_{stg}	–55 to +125	°C

Caution: Permanent damage to the chip may result if absolute maximum ratings are exceeded.

Ensure so that the impressed voltage does not exceed 4.3 V for pins for which the maximum rating is determined by the voltage on the V_{CC} and V_{CL} pins, or 7.0 V for pins for which the maximum rating is determined by $V_{CC}B$.

The V_{CC} and V_{CL} pins must be connected to the V_{CC} power supply.

21.1.2 DC Characteristics

Table 21.2 lists the DC characteristics. Permitted output current values and bus drive characteristics are shown in tables 21.3 and 21.4, respectively.

Table 21.2 DC Characteristics (1)

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V, T_a = –20 to +75°C

Item			Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	P67 to P60*[1], $\overline{\text{KIN15}}$ to $\overline{\text{KIN8}}$, $\overline{\text{IRQ2}}$ to $\overline{\text{IRQ0}}$, $\overline{\text{IRQ5}}$ to $\overline{\text{IRQ3}}$	(1)*[4]	V_T^-	$V_{CC} \times 0.2$ $V_{CC}B \times 0.2$	—	—	V	
			V_T^+	—	—	$V_{CC} \times 0.7$ $V_{CC}B \times 0.7$		
			$V_T^+ - V_T^-$	$V_{CC} \times 0.05$ $V_{CC}B \times 0.05$	—	—		
Schmitt trigger input voltage (in level switching)*[6]	P67 to P60 (KWUL = 00)		V_T^-	$V_{CC} \times 0.2$	—	—	V	
			V_T^+	—	—	$V_{CC} \times 0.7$		
			$V_T^+ - V_T^-$	$V_{CC} \times 0.05$	—	—		
	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	$\overline{\text{RES}}$, $\overline{\text{STBY}}$, NMI, MD1, MD0	(2)	V_{IH}	$V_{CC} \times 0.9$	—	V_{CC} +0.3	V	
	EXTAL			$V_{CC} \times 0.7$	—	V_{CC} +0.3		
	PA7 to PA0			$V_{CC}B \times 0.7$	—	$V_{CC}B + 0.3$		

Item			Symbol	Min	Typ	Max	Unit	Test Conditions
Input high voltage	P97, P86, P52, P42	(2)	V_{IH}	$V_{CC} \times 0.7$	—	5.5	V	
	Input pins except (1) and (2) above			$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
Input low voltage	$\overline{RES}$, $\overline{STBY}$, MD1, MD0	(3)	V_{IL}	–0.3	—	$V_{CC} \times 0.1$	V	
	PA7 to PA0			–0.3	—	$V_{CC}B \times 0.2$		$V_{CC}B$ = 3.0 V to 4.0 V
						0.8		$V_{CC}B$ = 4.0 V to 5.5 V
	NMI, EXTAL, input pins except (1) and (3) above			–0.3	—	$V_{CC} \times 0.2$		V_{CC} = 3.0 V to 3.6 V
Output high voltage	All output pins (except P97, P86, P52, and P42) *2, *3, *4		V_{OH}	V_{CC} – 0.5 $V_{CC}B$ – 0.5	—	—	V	I_{OH} = –200 µA
				V_{CC} – 1.0 $V_{CC}B$ – 1.0	—	—	V	I_{OH} = –1 mA, (V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 4.5 V)
	P97, P86, P52, and P42*2			0.5	—	—	V	I_{OH} = –200 µA

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Output low voltage	All output pins (except $\overline{\text{RESO}}$)*[3]	V_{OL}	—	—	0.4	V	I_{OL} = 1.6 mA
	Ports 1 to 3		—	—	1.0	V	I_{OL} = 5 mA
	$\overline{\text{RESO}}$		—	—	0.4	V	I_{OL} = 1.6 mA

Notes: 1. P67 to P60 include peripheral module inputs multiplexed on those pins.

2. P52/ExSCK1/SCL0, P97/SDA0, P86/SCK1/SCL1, and P42/SDA1 are NMOS push-pull outputs.

When the SCL0, SDA0, SCL1, or SDA1 (ICE = 1) pin is used as an output, it is NMOS open-drain output. Therefore, an external pull-up resistor must be connected in order to output high level.

P52/ExSCK1, P97, P86/SCK1, and P42 (ICE = 0) high levels are driven by NMOS.

An external pull-up resistor is necessary to provide high-level output from SCK1 and ExSCK1.

3. When IICS = 0, ICE = 0, and KBIOE = 0. Low-level output when the bus drive function is selected is determined separately.

4. The port A characteristics depend on V_{CC}B, and the other pins characteristics depend on V_{CC}.

Table 21.2 DC Characteristics (2)

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input leakage current	$\overline{\text{RES}}$	$\lvert I_{in} \rvert$	—	—	10.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V
	$\overline{\text{STBY}}$, NMI, MD1, MD0		—	—	1.0		
Three-state leakage current (off state)	Ports 1 to 9, A*[3], and B	$\lvert I_{TSI} \rvert$	—	—	1.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V, V_{in} = 0.5 to $V_{CC}B$ – 0.5 V
Input pull-up MOS current	Ports 1 to 3	$–I_P$	5	—	150	μA	V_{in} = 0 V, V_{CC} = 3.0 V to 3.6 V $V_{CC}B$ = 3.0 V to 5.5 V
	Ports 6 (P6PUE = 0) and B		30	—	300		
	Port A*[3]		30	—	600		
	Port 6 (P6PUE = 1)		3	—	100		
Input capacitance	$\overline{\text{RES}}$ (4)	C_{in}	—	—	80	pF	V_{in} = 0 V, f = 1 MHz, T_a = 25°C
	NMI		—	—	50	pF	
	P52, P97, P42, P86, PA7 to PA2		—	8	20	pF	
	Input pins except (4) above		—	—	15	pF	
Current dissipation*[1]	Normal operation	I_{CC}	—	30	40	mA	f = 10 MHz
	Sleep mode		—	20	32	mA	f = 10 MHz
	Standby mode*[2]		—	1	5.0	μA	T_a ≤ 50°C
			—	—	20.0		50°C < T_a
RAM standby voltage		V_{RAM}	2.0	—	—	V	

Notes: 1. Current dissipation values are for V_{IH} min = V_{CC} – 0.2 V, $V_{CC}B$ – 0.2 V, and V_{IL} max = 0.2 V with all output pins unloaded and the on-chip pull-up MOSs in the off state.

2. The values are for $V_{RAM} \leq V_{CC} < 3.0$ V, V_{IH} min = V_{CC}– 0.2 V, $V_{CC}B$ – 0.2 V, and V_{IL} max = 0.2 V.

3. The port A characteristics depend on $V_{CC}B$, and the other pins characteristics depend on V_{CC}.

Table 21.2 DC Characteristics (3) When LPC Function is Used

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V, T_a = –20 to +75°C

Item		Symbol	Min	Max	Unit	Test Conditions
Input high voltage	P37 to P30, P83 to P80, PB1, PB0	V_{IH}	$V_{CC} \times 0.5$	—	V	
Input low voltage	P37 to P30, P83 to P80, PB1, PB0	V_{IL}	—	$V_{CC} \times 0.3$	V	
Output high voltage	P37, P33 to P30, P82 to P80, PB1, PB0	V_{OH}	$V_{CC} \times 0.9$	—	V	I_{OH} = –0.5 mA
Output low voltage	P37, P33 to P30, P82 to P80, PB1, PB0	V_{OL}	—	$V_{CC} \times 0.1$	V	I_{OL} = 1.5 mA

Table 21.3 Permissible Output Currents

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit
Permissible output low current (per pin)	SCL1, SCL0, SDA1, SDA0, PS2AC to PS2CC, PS2AD to PS2CD, PA7 to PA4 (bus drive function selected)	I_{OL}	—	—	10	mA
	Ports 1, 2, 3		—	—	2	
	$\overline{\text{RESO}}$		—	—	1	
	Other output pins		—	—	1	
Permissible output low current (total)	Total of ports 1, 2, and 3	ΣI_{OL}	—	—	40	mA
	Total of all output pins, including the above		—	—	60	
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	mA
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	30	mA

Notes: 1. To protect chip reliability, do not exceed the output current values in table 21.3.

2. When driving a Darlington pair or LED, always insert a current-limiting resistor in the output line, as show in figures 21.1 and 21.2.

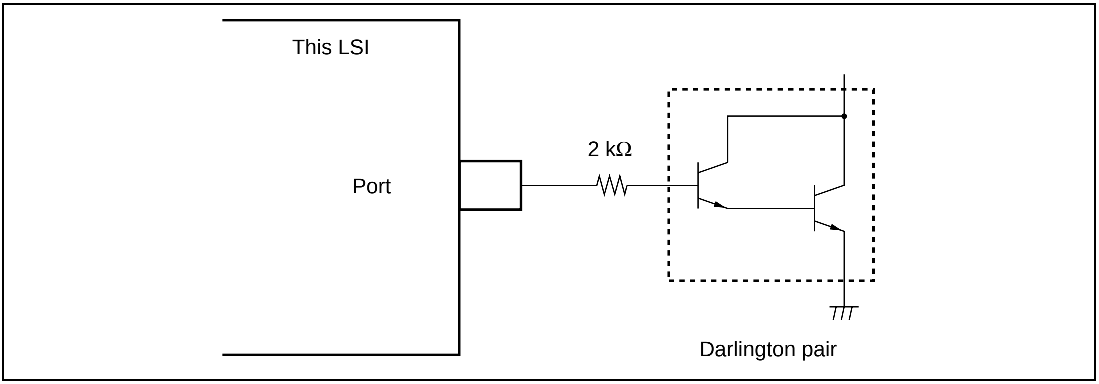


Figure 21.1 Darlington Pair Drive Circuit (Example)

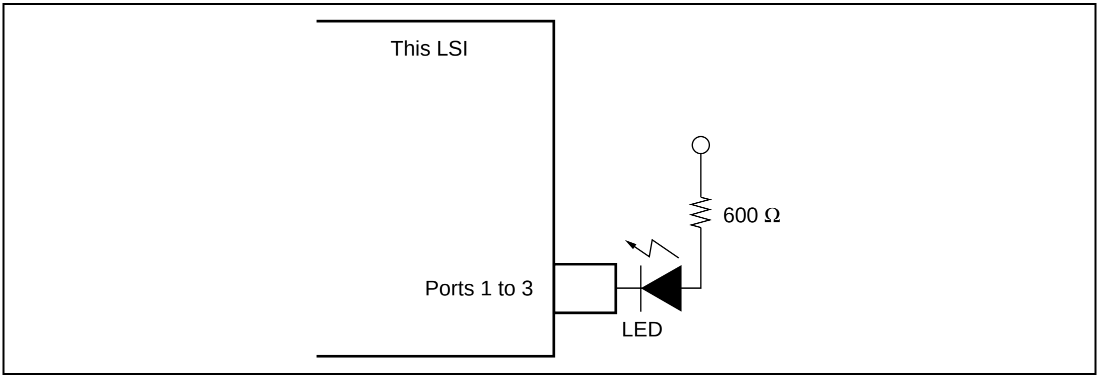


Figure 21.2 LED Drive Circuit (Example)

Table 21.4 Bus Drive Characteristics

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, Ta = –20 to +75°C

Applicable Pins: SCL1, SCL0, SDA1, SDA0 (bus drive function selected)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Schmitt trigger input voltage	V_T^-	V_{CC} × 0.3	—	—	V	V_{CC} = 3.0 V to 3.6 V
	V_T^+	—	—	V_{CC} × 0.7		V_{CC} = 3.0 V to 3.6 V
	$V_T^+ – V_T^-$	V_{CC} × 0.05	—	—		V_{CC} = 3.0 V to 3.6 V
Input high voltage	V_{IH}	V_{CC} × 0.7	—	5.5	V	V_{CC} = 3.0 V to 3.6 V
Input low voltage	V_{IL}	–0.5	—	V_{CC} × 0.3		V_{CC} = 3.0 V to 3.6 V
Output low voltage	V_{OL}	—	—	0.5	V	I_{OL} = 8 mA
		—	—	0.4		I_{OL} = 3 mA
Input capacitance	C_{in}	—	—	20	pF	V_{in} = 0 V, f = 1 MHz, T_a = 25°C
Three-state leakage current (off state)	$\lvert I_{TSI} \rvert$	—	—	1.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V
SCL, SDA output fall time	t_{Of}	20 + 0.1Cb	—	250	ns	V_{CC} = 3.0 V to 3.6 V

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{CC}B = 3.0 V to 5.5 V, V_{SS} = 0 V, Ta = –20 to +75°C

Applicable Pins: PS2AC, PS2AD, PS2BC, PS2BD, PS2CC, PS2CD, PA7 to PA4 (bus drive function selected)

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Output low voltage	V_{OL}	—	—	0.8	V	I_{OL} = 16 mA, V_{CC}B = 4.5 V to 5.5 V
		—	—	0.5		I_{OL} = 8 mA
		—	—	0.4		I_{OL} = 3 mA

21.1.3 AC Characteristics

Figure 21.3 shows the test conditions for the AC characteristics.

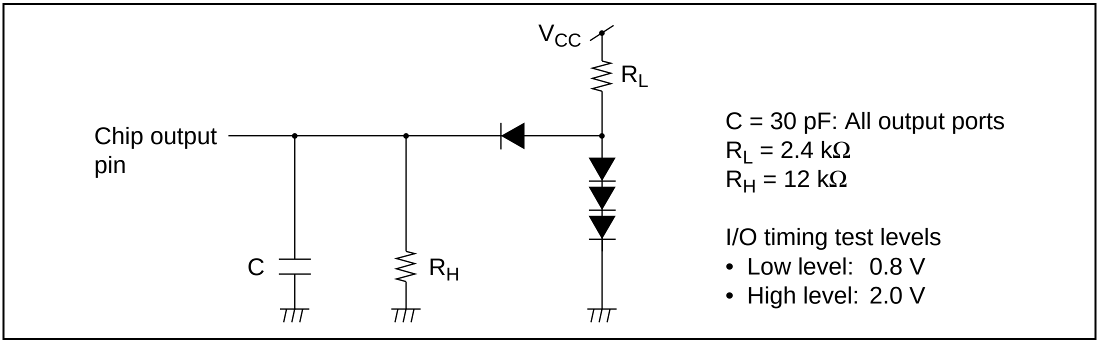


Figure 21.3 Output Load Circuit

Clock Timing: Table 21.5 shows the clock timing. The clock timing specified here covers clock (ϕ) output and clock pulse generator (crystal) and external clock input (EXTAL pin) oscillation settling times. For details on external clock input (EXTAL pin and EXCL pin) timing, see section 18, Clock Pulse Generator.

Table 21.5 Clock Timing

Condition: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V,
ϕ = 2 MHz to maximum operating frequency, T_a = –20 to +75°C

		Condition			
		10 MHz			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Reference**
Clock cycle time	t_{cyc}	100	500	ns	Figure 21.5
Clock high pulse width	t_{CH}	30	—	ns	
Clock low pulse width	t_{CL}	30	—	ns	
Clock rise time	t_{Cr}	—	20	ns	
Clock fall time	t_{Cf}	—	20	ns	
Oscillation settling time at reset (crystal)	t_{OSC1}	20	—	ms	Figure 21.6 Figure 21.7
Oscillation settling time in software standby (crystal)	t_{OSC2}	8	—	ms	
External clock output stabilization delay time	t_{DEXT}	500	—	µs	

Control Signal Timing: Table 21.6 shows the control signal timing. The only external interrupts that can operate on the subclock (ϕ = 32.768 kHz) are NMI and IRQ0, 1, 2, 6, and 7.

Table 21.6 Control Signal Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V,
ϕ = 32.768 kHz, 2 MHz to maximum operating frequency, T_a = –20 to +75°C

		Condition			
		10 MHz			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**
$\overline{\text{RES}}$ setup time	t_{RESS}	300	—	ns	Figure 21.8
$\overline{\text{RES}}$ pulse width	t_{RESW}	20	—	t_{cyc}	
NMI setup time (NMI)	t_{NMIS}	250	—	ns	Figure 21.9
NMI hold time (NMI)	t_{NMIH}	10	—	ns	
NMI pulse width (exiting software standby mode)	t_{NMIW}	200	—	ns	
IRQ setup time ($\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$)	t_{IRQS}	250	—	ns	
IRQ hold time($\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$)	t_{IRQH}	10	—	ns	
IRQ pulse width ($\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ2}}$ to $\overline{\text{IRQ0}}$) (exiting software standby mode)	t_{IRQW}	200	—	ns	

Timing of On-Chip Peripheral Modules: Tables 21.7 to 21.10 show the on-chip peripheral module timing. The only on-chip peripheral modules that can operate in subclock operation (ø = 32.768 kHz) are the I/O ports, external interrupts (NMI and IRQ0, 1, 2, 6, and 7), the watchdog timer, and the 8-bit timer (channels 0 and 1).

Table 21.7 Timing of On-Chip Peripheral Modules

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V, ϕ = 32.768 kHz*,
2 MHz to maximum operating frequency, T_a = –20 to +75°C

				Condition			
				10 MHz			
Item			**Symbol**	**Min**	**Max**	**Unit**	**Test Conditions**
I/O ports	Output data delay time		t_{PWD}	—	100	ns	Figure 21.10
	Input data setup time		t_{PRS}	50	—		
	Input data hold time		t_{PRH}	50	—		
FRT	Timer output delay time		t_{FTOD}	—	100	ns	Figure 21.11
	Timer input setup time		t_{FTIS}	50	—		
	Timer clock input setup time		t_{FTCS}	50	—		Figure 21.12
	Timer clock pulse width	Single edge	t_{FTCWH}	1.5	—	t_{cyc}	
		Both edges	t_{FTCWL}	2.5	—		
TMR	Timer output delay time		t_{TMOD}	—	100	ns	Figure 21.13
	Timer reset input setup time		t_{TMRS}	50	—		Figure 21.15
	Timer clock input setup time		t_{TMCS}	50	—		Figure 21.14
	Timer clock pulse width	Single edge	t_{TMCWH}	1.5	—	t_{cyc}	
		Both edges	t_{TMCWL}	2.5	—		
PWMX	Pulse output delay time		t_{PWOD}	—	100	ns	Figure 21.16
SCI	Input clock cycle	Asynchronous	t_{Scyc}	4	—	t_{cyc}	Figure 21.17
		Synchronous		6	—		
	Input clock pulse width		t_{SCKW}	0.4	0.6	t_{Scyc}	
	Input clock rise time		t_{SCKr}	—	1.5	t_{cyc}	
	Input clock fall time		t_{SCKf}	—	1.5		

			Condition			
			10 MHz			
Item		Symbol	Min	Max	Unit	Test Conditions
SCI	Transmit data delay time (synchronous)	t_{TXD}	—	100	ns	Figure 21.18
	Receive data setup time (synchronous)	t_{RXS}	100	—	ns	
	Receive data hold time (synchronous)	t_{RXH}	100	—	ns	
WDT	$\overline{\text{RESO}}$ output delay time	t_{RESD}	—	200	ns	Figure 21.19
	$\overline{\text{RESO}}$ output pulse width	t_{RESOW}	132	—	t_{cyc}	

Note:* Only peripheral modules that can be used in subclock operation

Table 21.8 Keyboard Buffer Controller Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, $V_{CC}B$ = 3.0 V to 5.5 V, V_{SS} = 0 V, ϕ = 2 MHz to maximum operating frequency, T_a = –20 to +75°C

		Ratings					
Item	Symbol	Min	Typ	Max	Unit	Test Conditions	Notes
KCLK, KD output fall time	t_{KBF}	20 + 0.1Cb	—	250	ns		Figure 21.19
KCLK, KD input data hold time	t_{KBIH}	150	—	—	ns		
KCLK, KD input data setup time	t_{KBIS}	150	—	—	ns		
KCLK, KD output delay time	t_{KBOD}	—	—	450	ns		
KCLK, KD capacitive load	C_b	—	—	400	pF		

Table 21.9 I^{2}C Bus Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, ϕ = 5 MHz to maximum operating frequency,
T_a = –20 to +75°C

		Ratings					
Item	**Symbol**	**Min**	**Typ**	**Max**	**Unit**	**Test Conditions**	**Notes**
SCL input cycle time	t_{SCL}	12	—	—	t_{cyc}		Figure 21.21
SCL input high pulse width	t_{SCLH}	3	—	—	t_{cyc}		
SCL input low pulse width	t_{SCLL}	5	—	—	t_{cyc}		
SCL, SDA input rise time	t_{Sr}	—	—	7.5*	t_{cyc}		
SCL, SDA input fall time	t_{Sf}	—	—	300	ns		
SCL, SDA input spike pulse elimination time	t_{SP}	—	—	1	t_{cyc}		
SDA input bus free time	t_{BUF}	5	—	—	t_{cyc}		
Start condition input hold time	t_{STAH}	3	—	—	t_{cyc}		
Retransmission start condition input setup time	t_{STAS}	3	—	—	t_{cyc}		
Stop condition input setup time	t_{STOS}	3	—	—	t_{cyc}		
Data input setup time	t_{SDAS}	0.5	—	—	t_{cyc}		
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 I^{2}C module. For details, see section 13.6, Usage Notes.

Table 21.10 LPC Module Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS}= 0 V, ϕ = 2 MHz to maximum operating frequency,
T_a = –20 to +75°C

Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Input clock cycle	t_{Lcyc}	30	—	—	ns	Figure 21.22
Input clock pulse width (H)	t_{LCKH}	11	—	—		
Input clock pulse width (L)	t_{LCKL}	11	—	—		
Transmit signal delay time	t_{TXD}	2	—	11		
Transmit signal floating delay time	t_{OFF}	—	—	28		
Receive signal setup time	t_{RXS}	7	—	—		
Receive signal hold time	t_{RXH}	0	—	—		

21.1.4 Flash Memory Characteristics

Table 21.11 shows the flash memory characteristics.

Table 21.11 Flash Memory Characteristics

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, T_a = –20 to +75°C

Item		Symbol	Min	Typ	Max	Unit	Test Condition
Programming time*[1], *[2],*[4]		t_P	—	10	200	ms/ 128 bytes	
Erase time*[1], *[3],*[6]		t_E	—	100	1200	ms/ block	
Reprogramming count		N_{WEC}	—	—	100	times	
Programming	Wait time after SWE-bit setting*[1]	x	1	—	—	µs	
	Wait time after PSU-bit setting*[1]	y	50	—	—	µs	
	Wait time after P-bit setting*[1], *[4]	z1	28	30	32	µs	1 ≤ n ≤ 6
		z2	198	200	202	µs	7 ≤ n ≤ 1000
		z3	8	10	12	µs	Additional write
	Wait time after P-bit clear*[1]	α	5	—	—	µs	
	Wait time after PSU-bit clear*[1]	β	5	—	—	µs	
	Wait time after PV-bit setting*[1]	γ	4	—	—	µs	
	Wait time after dummy write*[1]	ε	2	—	—	µs	
	Wait time after PV-bit clear*[1]	η	2	—	—	µs	
	Wait time after SWE-bit clear*[1]	θ	100	—	—	µs	
	Maximum programming count*[1], *[4],*[5]	N	—	—	1000	times	

	Item	Symbol	Min	Typ	Max	Unit	Test Conditions
Erase	Wait time after SWE-bit setting*[1]	x	1	—	—	µs	
	Wait time after ESU-bit setting*[1]	y	100	—	—	µs	
	Wait time after E-bit setting*[1], *[6]	z	10	—	100	ms	
	Wait time after E-bit clear*[1]	α	10	—	—	µs	
	Wait time after ESU-bit clear*[1]	β	10	—	—	µs	
	Wait time after EV-bit setting*[1]	γ	20	—	—	µs	
	Wait time after dummy write*[1]	ε	2	—	—	µs	
	Wait time after EV-bit clear*[1]	η	4	—	—	µs	
	Wait time after SWE-bit clear*[1]	θ	100	—	—	µs	
	Maximum erase count*[1], *[6], *[7]	N	—	—	120	times	

Notes: 1. Set the times according to the program/erase algorithms.

2. Programming time per 128 bytes (Shows the total period for which the P-bit in FLMCR1 is set. It does not include the programming 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) = (wait time after P-bit setting (z1) + (z3)) × 6
+ wait time after P-bit setting (z2) × ((N) – 6)

5. The maximum number of writes (N) should be set according to the actual set value of z1, z2 and z3 to allow programming within the maximum programming time (t_P (max)).

The wait time after P-bit setting (z1, z2, and z3) should be alternated according to the number of writes (n) as follows:

$1 \le n \le 6$ z1 = 30µs, z3 = 10µs
$7 \le n \le 1000$ z2 = 200µs

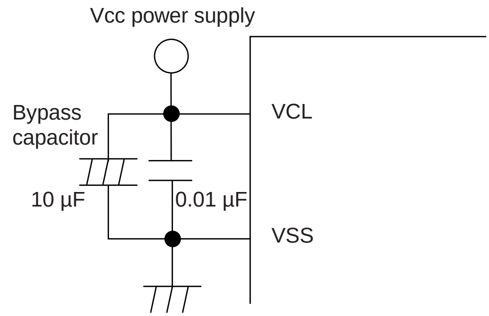
6. Maximum erase time (t_E (max))

t_E (max) = Wait time after E-bit setting (z) × maximum erase count (N)

7. The maximum number of erases (N) should be set according to the actual set value of z to allow erasing within the maximum erase time (t_E (max)).

21.1.5 Usage Note

The method of connecting an external capacitor is shown in figure 21.4. Connect the system power supply to the VCL pin together with the VCC pins.



< Vcc = 3.0 V to 3.6 V >
Connect the Vcc power supply to the chip's VCL pin in the same way as the VCC pins.
It is recommended that a bypass capacitor be connected to the power supply pins. (Values are reference values.)

Figure 21.4 Connection of VCL Capacitor

21.2 Timing Chart

21.2.1 Clock Timing

The clock timings are shown below.

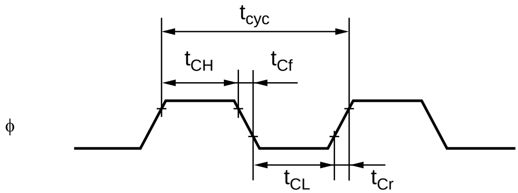


Figure 21.5 System Clock Timing

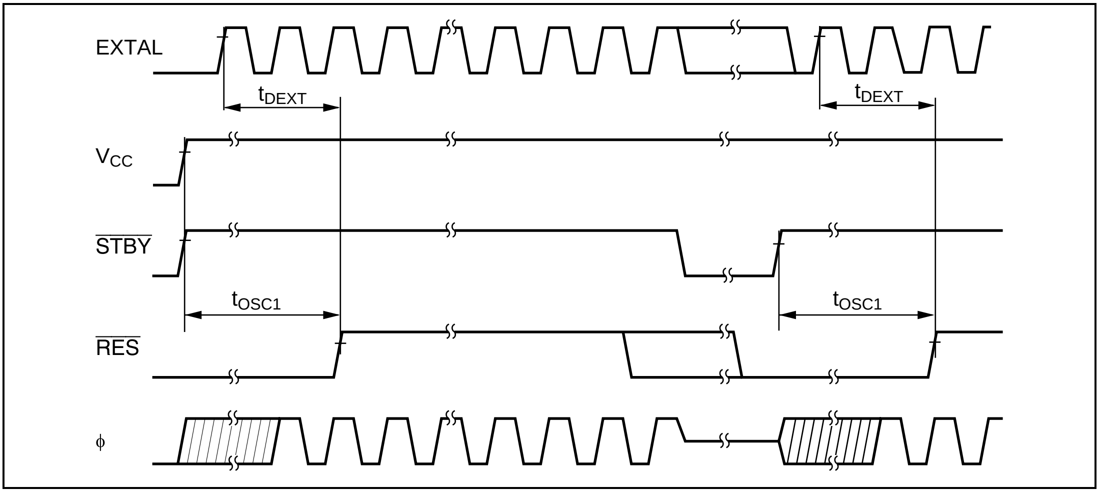


Figure 21.6 Oscillation Settling Timing

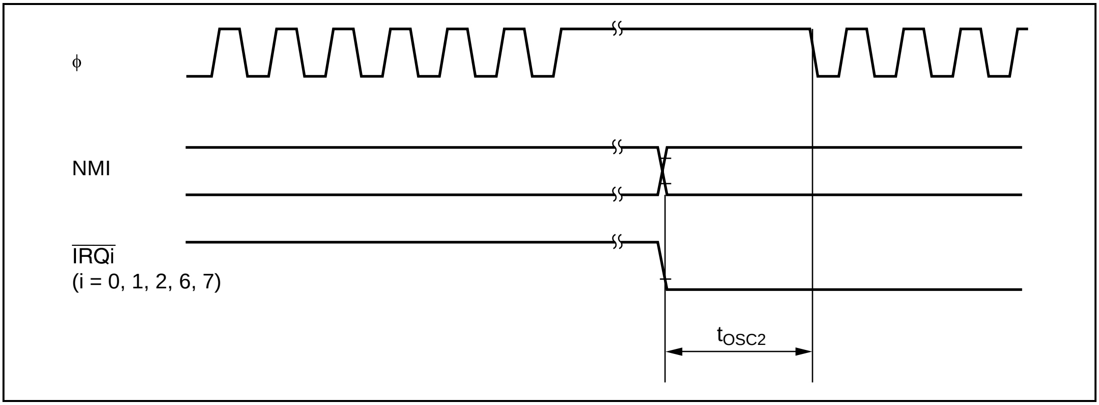


Figure 21.7 Oscillation Setting Timing (Exiting Software Standby Mode)

21.2.2 Control Signal Timing

The control signal timings are shown below.

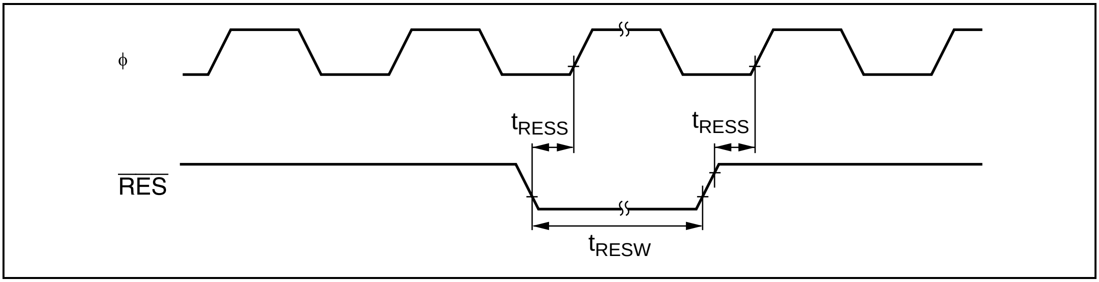


Figure 21.8 Reset Input Timing

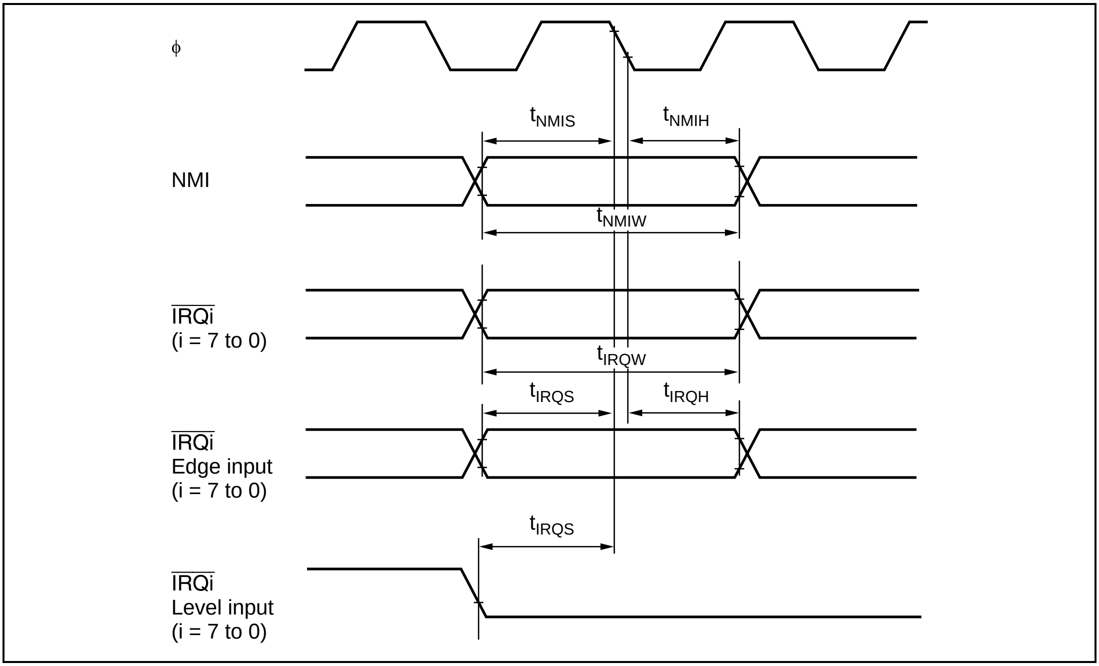


Figure 21.9 Interrupt Input Timing

21.2.3 On-Chip Peripheral Module Timing

The on-chip peripheral module timings are shown below.

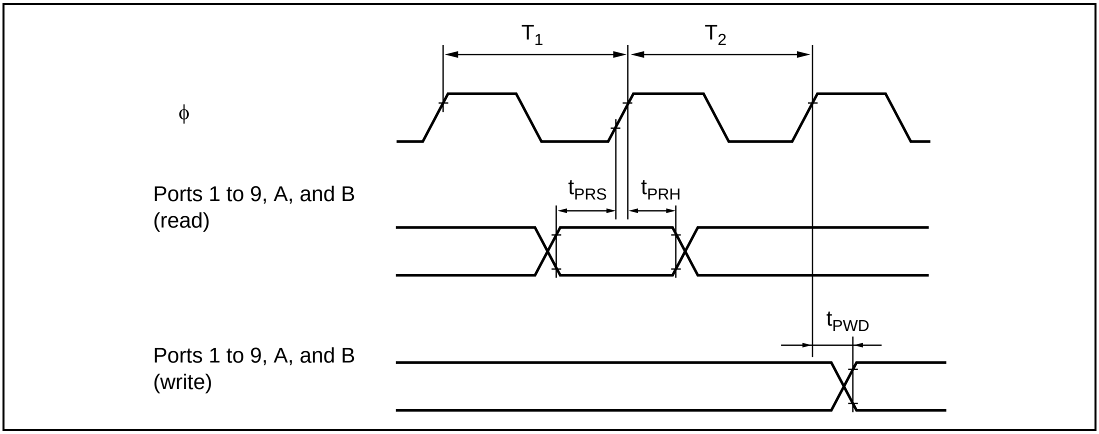


Figure 21.10 I/O Port Input/Output Timing

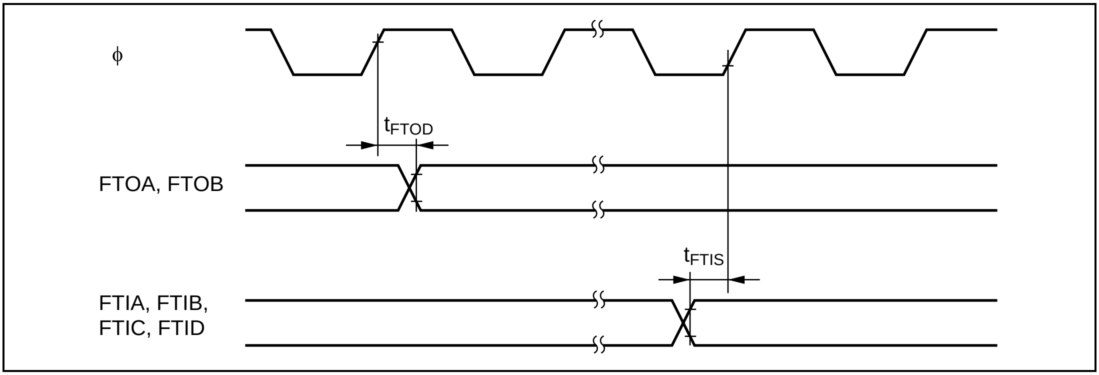


Figure 21.11 FRT Input/Output Timing

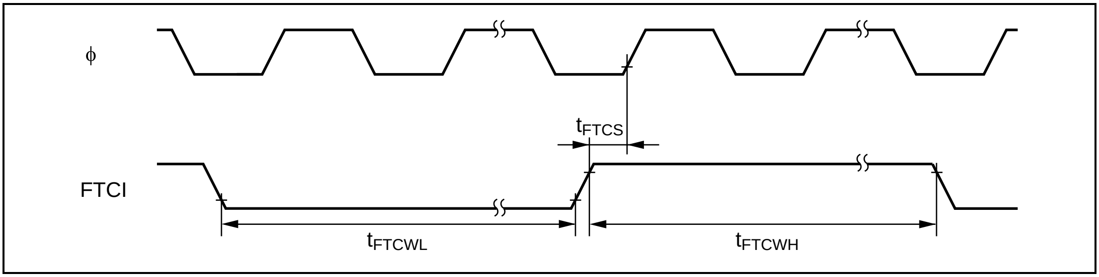


Figure 21.12 FRT Clock Input Timing

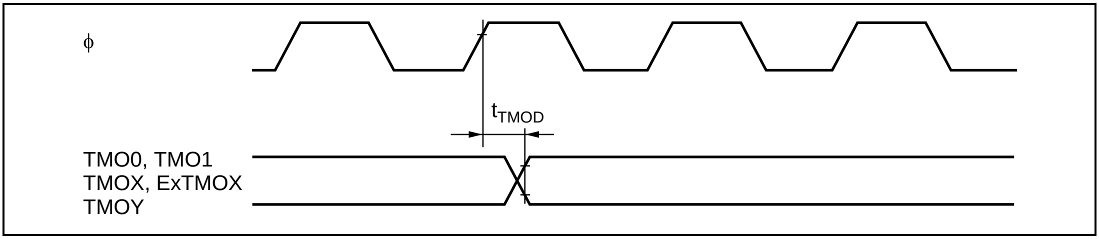


Figure 21.13 8-Bit Timer Output Timing

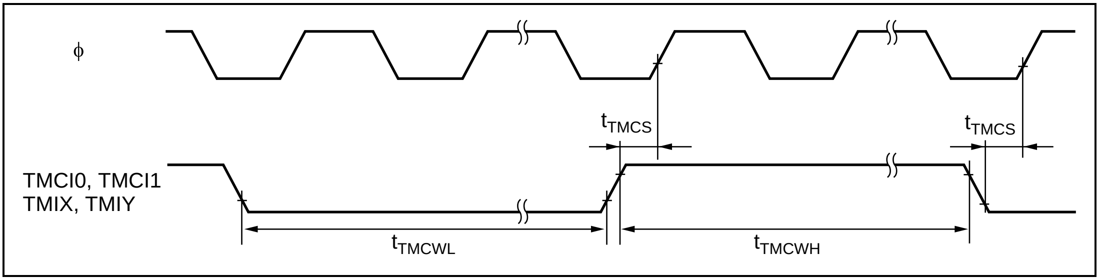


Figure 21.14 8-Bit Timer Clock Input Timing

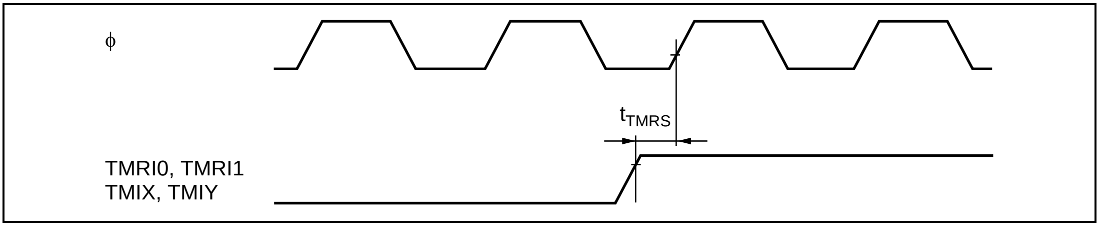


Figure 21.15 8-Bit Timer Reset Input Timing

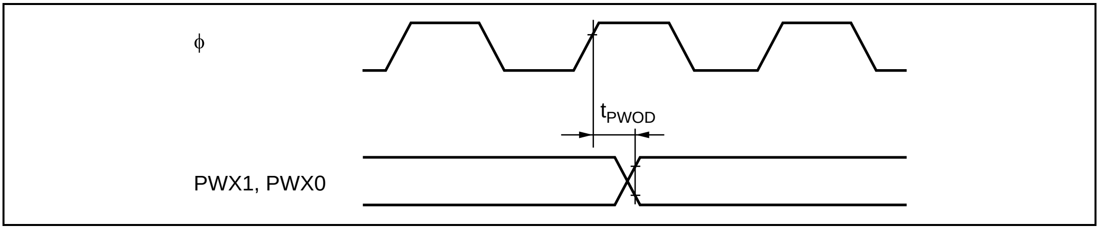


Figure 21.16 PWMX Output Timing

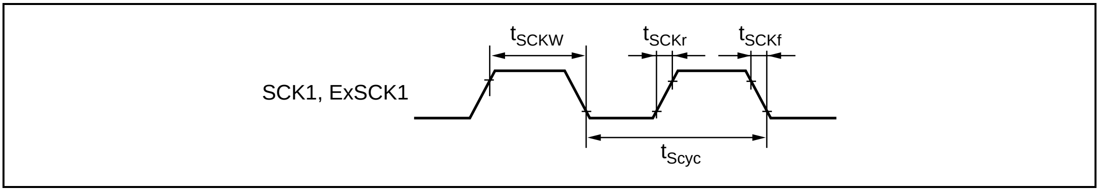


Figure 21.17 SCK Clock Input Timing

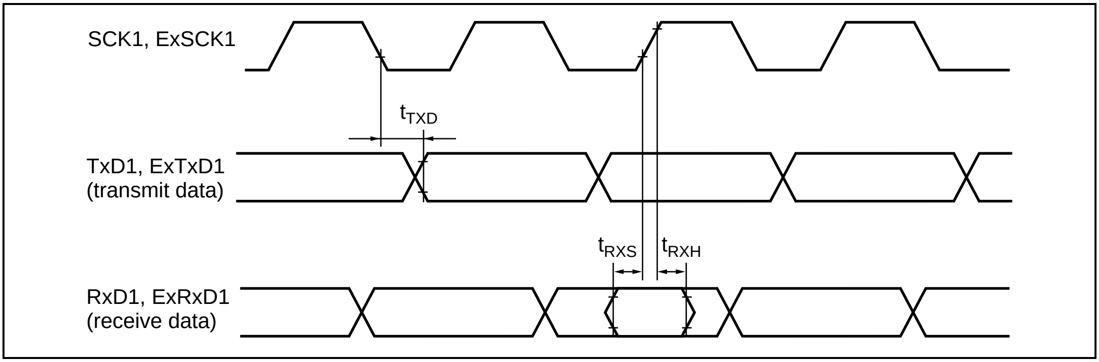


Figure 21.18 SCI Input/Output Timing (Synchronous Mode)

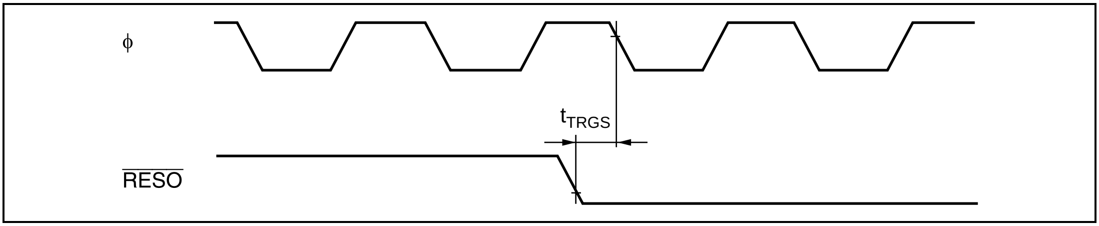
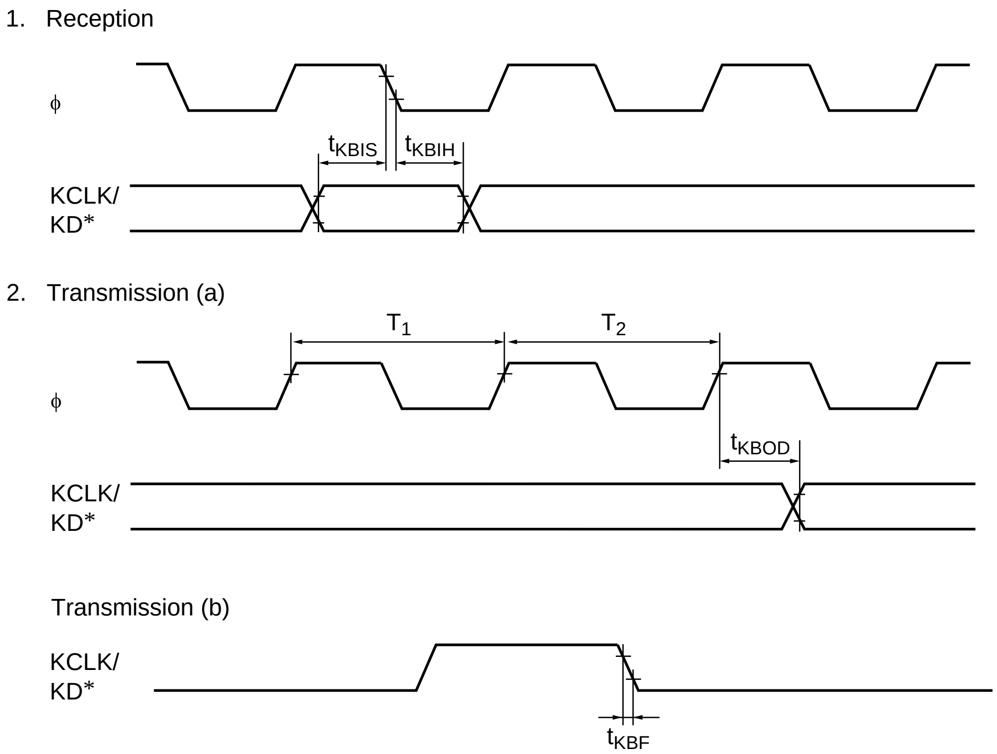


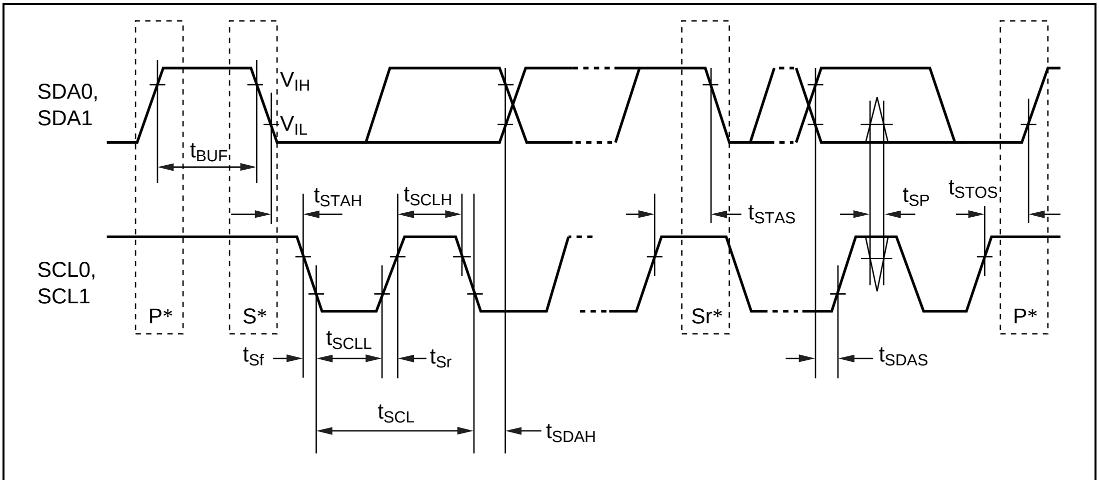
Figure 21.19 WDT Output Timing ($\overline{\text{RESO}}$)



Note: φ shown here is the clock scaled by 1/N when the operating mode is active medium-speed mode.

* KCLK: PS2AC to PS2CC
KD: PS2AD to PS2CD

Figure 21.20 Keyboard Buffer Controller Timing



Note:* S, P, and Sr indicate the following conditions.

S: Start condition
P: Stop condition
Sr: Retransmission start condition

Figure 21.21 I²C Bus Interface Input/Output Timing

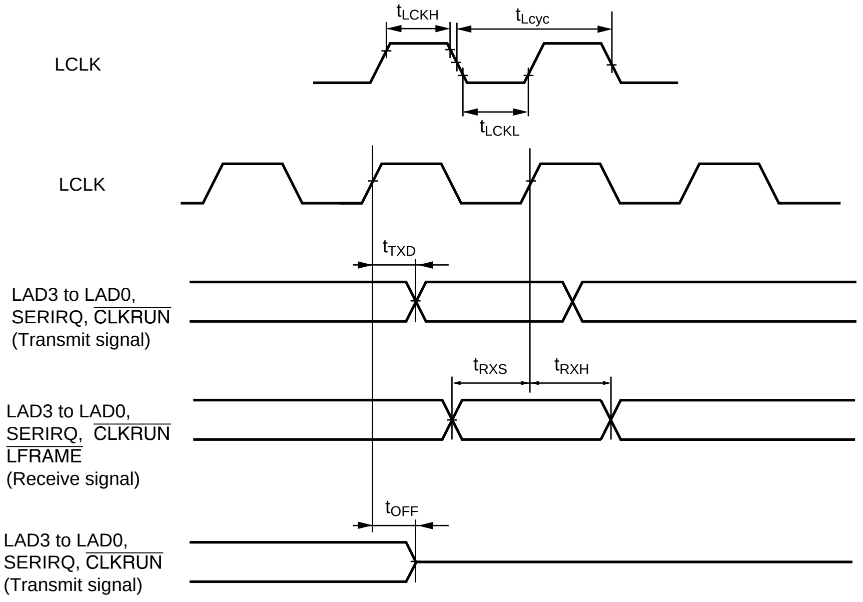


Figure 21.22 Host Interface (LPC) Timing

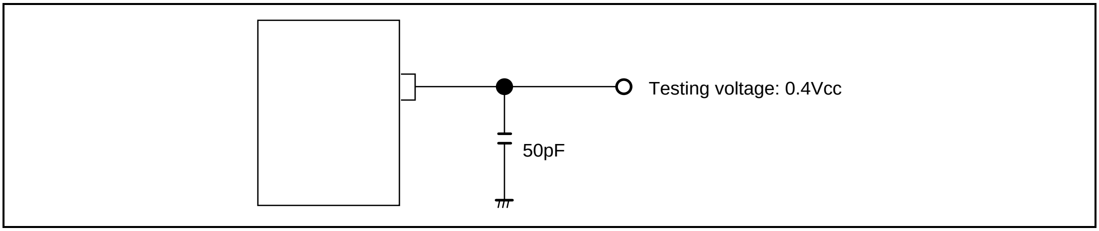


Figure 21.23   Tester Measurement Condition

Appendix A I/O Port States in Each Processing State

Table A.1 I/O Port States in Each Processing State

Port Name Pin Name	Reset	Hardware Standby Mode	Software Standby Mode	Watch Mode	Sleep Mode	Sub-sleep Mode	Subactive Mode	Program Execution State
Port 1	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 2	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 3	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 4	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 5	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 6	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 7	T	T	keep	keep	keep	keep	I/O port*	I/O port*
Port 8	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 97	T	T	keep	keep	keep	keep	I/O port	I/O port
Port 96 φ EXCL	T	T	[DDR = 1] H [DDR = 0] T	EXCL input	[DDR = 1] clock output [DDR = 0] T	EXCL input	EXCL input	Clock output/ EXCL input/ input port
Ports 95 to 90	T	T	keep	keep	keep	keep	I/O port	I/O port
Port A	T	T	keep	keep	keep	keep	I/O port	I/O port
Port B	T	T	keep	keep	keep	keep	I/O port	I/O port

Legend

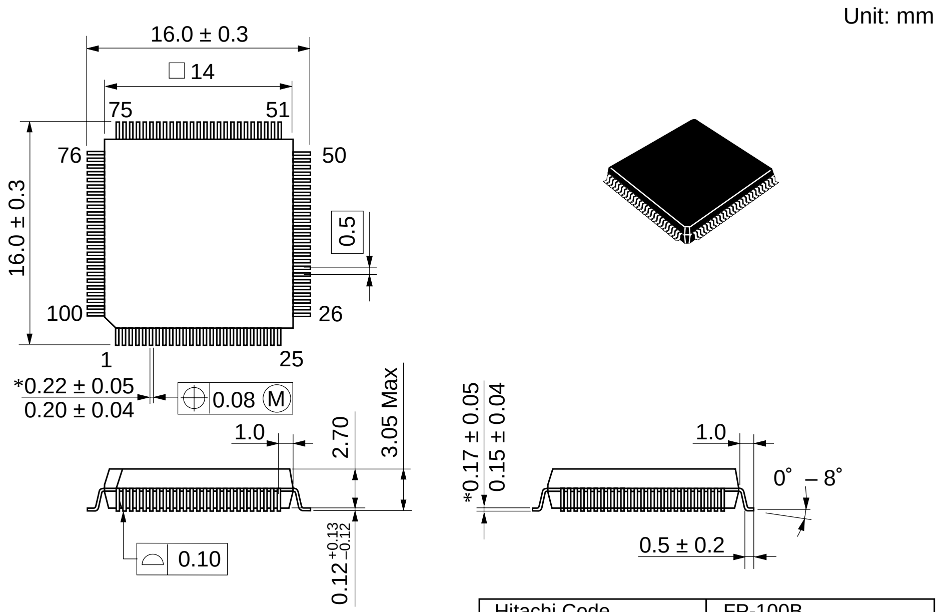
H:	High
L:	Low
T:	High-impedance state
keep:	Input ports are in the high-impedance state (when DDR = 0 and PCR = 1, input pull-up MOSs remain on). Output ports maintain their previous state. Depending on the pins, the on-chip peripheral modules may be initialized and the I/O port function determined by DDR and DR used.
DDR:	Data direction register

Note:* The program development tool (emulator) does not support the output.

Appendix B Product Codes

Product Type		Product Code	Mark Code	Package (Hitachi Package Code)
H8S/2110B	Flash memory version (3-V version)	HD64F2110BV	F2110BVFA10	100-pin QFP (FP-100B)
			F2110BVTE10	100-pin TQFP (TFP-100B)

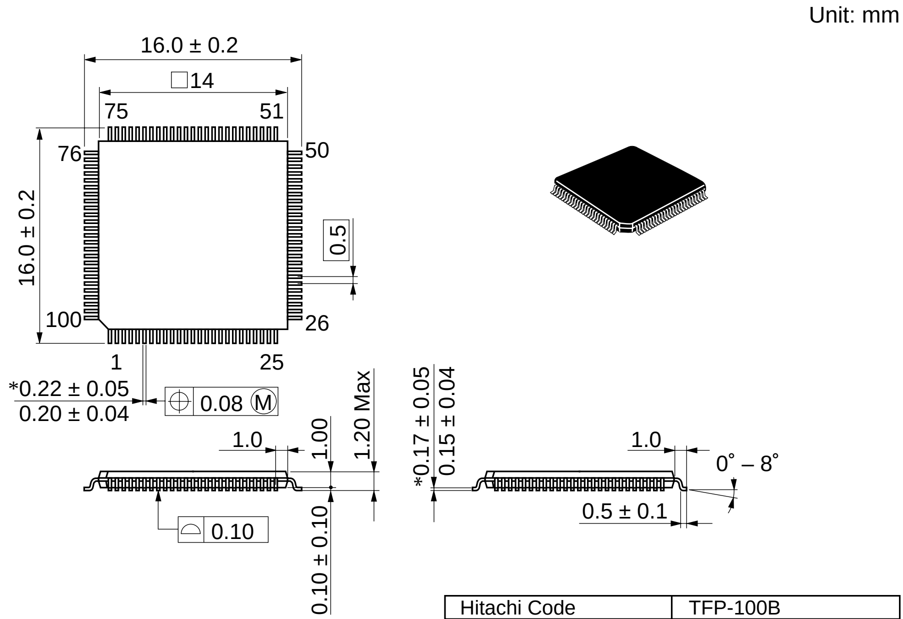
Appendix C Package Dimensions



*Dimension including the plating thickness
Base material dimension

Hitachi Code	FP-100B
JEDEC	—
EIAJ	Conforms
Weight (reference value)	1.2 g

Figure C.1 Package Dimensions (FP-100B)



*Dimension including the plating thickness

Base material dimension

Hitachi Code	TFP-100B
JEDEC	—
JEITA	Conforms
Mass (reference value)	0.5 g

Figure C.2 Package Dimensions (TFP-100B)

Index

H8S/2110B Hardware Manual

Publication Date: 1st Edition, September 2002
Published by: Business Operation Division
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.