To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

Notice

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

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

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

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

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

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

User's Manual

16

The revision list can be viewed directly by clicking the title page.
The revision list summarizes the locations of revisions and additions.
Details should always be checked by referring to the relevant text.

H8S/2153 Group

Hardware Manual

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

H8S/2153 R4F2153

Renesas Electronics
www.renesas.com

Rev.3.00 2009.09

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

General Precautions in the Handling of MPU/MCU Products

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

1. Handling of Unused Pins

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

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

2. Processing at Power-on

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

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

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

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

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

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

Configuration of This Manual

This manual comprises the following items:

1. General Precautions in the Handling of MPU/MCU Products
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 for 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/2153 Group are single-chip microcomputers made up of the high-speed H8S/2600 CPU employing Renesas Technology original architecture as its core, and the peripheral functions required to configure a system. The H8S/2600 CPU has an instruction set that is compatible with the H8/300 and H8/300H CPUs.

Target Users: This manual was written for users who will be using the H8S/2153 Group 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/2153 Group to the target users.
Refer to the H8S/2600 Series, H8S/2000 Series Software 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 Software 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 23, List of Registers.

Examples:	Register name:	The following notation is used for cases when the same or a similar function, e.g. 16-bit timer pulse unit or serial communication, is implemented on more than one channel: XXX_N (XXX is the register name and N is the channel number)
	Bit order:	The MSB is on the left and the LSB is on the right.

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

H8S/2153 Group manuals:

Document Title	Document No.
H8S/2153 Group Hardware Manual	This manual
H8S/2600 Series, H8S/2000 Series Software Manual	REJ09B0139

User's manuals for development tools:

Document Title	Document No.
H8S, H8/300 Series C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual	REJ10B0161
H8S, H8/300 Series Simulator/Debugger User's Manual	REJ10B0211
High-performance Embedded Workshop User's Manual	REJ10J2000

Contents

Section 21 Clock Pulse Generator 625

Section 22 Power-Down Modes 631

Figures

Section 1 Overview

Section 2 CPU

Section 3 MCU Operating Modes

Section 4 Exception Handling

Section 5 Interrupt Controller

Section 6 Bus Controller (BSC)

Section 7 Data Transfer Controller (DTC)

Section 8 I/O Ports

Section 9 14-Bit PWM Timer (PWMX)

Section 10 16-Bit Free-Running Timer (FRT)

Section 11 8-Bit Timer (TMR)

Section 12 Watchdog Timer (WDT)

Section 13 Serial Communication Interface (SCI)

Section 14 CRC Operation Circuit (CRC)

Section 15 I^{2}C Bus Interface (IIC)

Section 16 LPC Interface (LPC)

Section 17 A/D Converter

Section 19 Flash Memory

Section 20 Boundary Scan (JTAG)

Section 21 Clock Pulse Generator

Section 22 Power-Down Modes

Section 24 Electrical Characteristics

Appendix

Tables

Section 7 Data Transfer Controller (DTC)

Section 8 I/O Ports

Section 9 14-Bit PWM Timer (PWMX)

Section 10 16-Bit Free-Running Timer (FRT)

Section 11 8-Bit Timer (TMR)

Section 12 Watchdog Timer (WDT)

Section 13 Serial Communication Interface (SCI)

Section 15 I^{2}C Bus Interface (IIC)

Section 16 LPC Interface (LPC)

Section 17 A/D Converter

Section 19 Flash Memory

Section 20 Boundary Scan (JTAG)

Section 21 Clock Pulse Generator

Section 22 Power-Down Modes

Section 24 Electrical Characteristics

Appendix

Section 1 Overview

1.1 Overview

- High-speed H8S/2600 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
  69 basic instructions
  Multiplication and accumulation instructions
- Various peripheral functions
  Data transfer controller (DTC)
  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)
  CRC operation circuit (CRC)
  I^{2}C bus interface (IIC)
  LPC interface (LPC)
  10-bit A/D converter
  Boundary scan (JTAG)
  Clock pulse generator
- On-chip memory

ROM Type	Model	ROM	RAM	Remarks
Flash memory Version	R4F2153	512 Kbytes	40 Kbytes	

- General I/O ports

  I/O pins: 65

  Input-only pins: 9
- Supports various power-down states
- Compact package

Package (code)	Body Size	Pin Pitch
PLBG0112GA-A (BP-112)	10 × 10 mm	0.8 mm

1.2 Internal Block Diagram

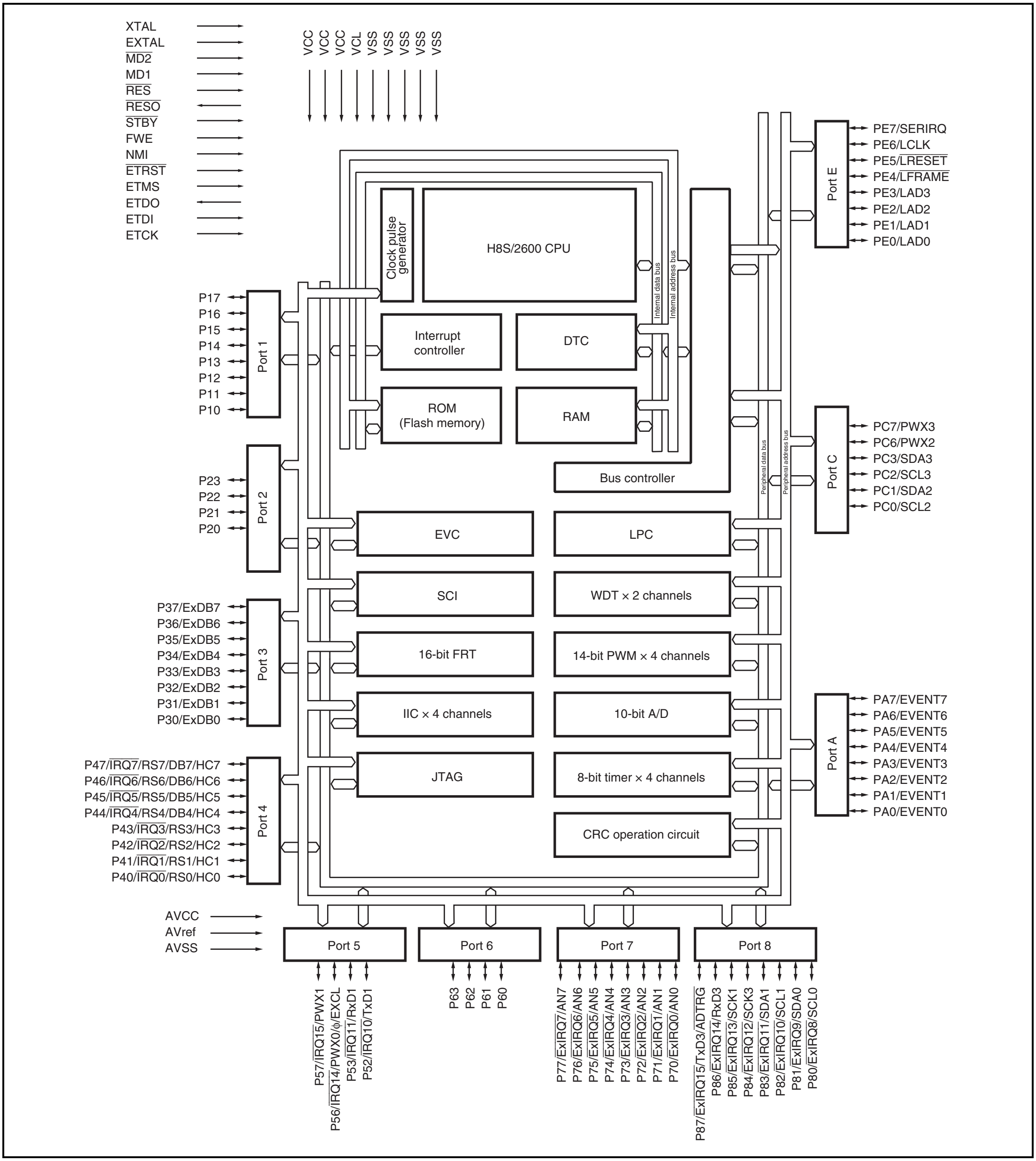


Figure 1.1 Internal Block Diagram

1.3 Pin Description

1.3.1 Pin Assignment

	A	B	C	D	E	F	G	H	J	K	L	
11	P13	P14	P16	P21	VSS	ETDI	ETMS	P62	AVref	P76	P74	11
10	P11	P12	P15	P17	P23	EXCK	NC	P61	AVCC	P75	P73	10
9	P30	P10	NC	NC	P22	ETDO	VCC	P60	NC	P72	P71	9
8	P33	P32	P31	VSS	P20	$\overline{\text{ETRST}}$	P63	P77	NC	P70	PE0	8
7	P36	NC	P35	P34	H8S/2153Group PLBG0112GA-A BP-112 (Top view)			AVSS	PE1	PE2	PE3	7
6	P40	P41	P37	P42				PE4	PE7	PE5	PE6	6
5	P43	P52	P53	P44				P82	P81	NC	P80	5
4	FWE	VSS	NC	P47	NMI	PC3	PA6	VSS	P85	P84	P83	4
3	$\overline{\text{RESO}}$	XTAL	NC	VSS	$\overline{\text{STBY}}$	$\overline{\text{MD2}}$	PC0	NC	NC	P87	P86	3
2	EXTAL	P45	P56	$\overline{\text{RES}}$	NC	PC6	PC1	PA5	PA4	PA1	PA0	2
1	VCC	P46	P57	MD1	VCL	PC7	PC2	PA7	VCC	PA3	PA2	1
	A	B	C	D	E	F	G	H	J	K	L	

▨ :NC pin

Figure 1.2 Pin Assignment (BP-112)

1.3.2 Pin Assignment in Each Operating Mode

Table 1.1 Pin Assignment in Each Operating Mode

Pin No.	Pin Name	
BP-112	**Single-Chip Mode**	**Flash Memory Programmer Mode**
A1	VCC	VCC
B2	P45/$\overline{\text{IRQ5}}$/RS5/DB5/HC5	FA13
B1	P46/$\overline{\text{IRQ6}}$/RS6/DB6/HC6	FA14
D4	P47/$\overline{\text{IRQ7}}$/RS7/DB7/HC7	FA15
C2	P56/$\overline{\text{IRQ14}}$/PWX0/ϕ/EXCL	NC
C1	P57/$\overline{\text{IRQ15}}$/PWX1	NC
D3	VSS	VSS
D2	$\overline{\text{RES}}$	$\overline{\text{RES}}$
D1	MD1	VSS
E4	NMI	FA9
E3	$\overline{\text{STBY}}$	VCC
E1	VCL	VCL
E2	NC	NC
F3	$\overline{\text{MD2}}$	VCC
F1	PC7/PWX3	$\overline{\text{WE}}$
F2	PC6/PWX2	NC
F4	PC3/SDA3	NC
G1	PC2/SCL3	NC
G2	PC1/SDA2	NC
G3	PC0/SCL2	NC
H1	PA7/EVENT7	VCC
G4	PA6/EVENT6	VCC
H2	PA5/EVENT5	VSS
J1	VCC	VCC
H3	NC	NC
J2	PA4/EVENT4	NC

Pin No.	Pin Name	
BP-112	**Single-Chip Mode**	**Flash Memory Programmer Mode**
K1	PA3/EVENT3	FA19
J3	NC	NC
L1	PA2/EVENT2	FA18
K2	PA1/EVENT1	FA17
L2	PA0/EVENT0	FA16
H4	VSS	VSS
K3	P87/$\overline{\text{ExIRQ15}}$/TxD3/$\overline{\text{ADTRG}}$	NC
L3	P86/$\overline{\text{ExIRQ14}}$/RxD3	NC
J4	P85/$\overline{\text{ExIRQ13}}$/SCK1	NC
K4	P84/$\overline{\text{ExIRQ12}}$/SCK3	NC
L4	P83/$\overline{\text{ExIRQ11}}$/SDA1	NC
H5	P82/$\overline{\text{ExIRQ10}}$/SCL1	NC
J5	P81/$\overline{\text{ExIRQ9}}$/SDA0	NC
L5	P80/$\overline{\text{ExIRQ8}}$/SCL0	NC
K5	NC	NC
J6	PE7/SERIRQ	NC
L6	PE6/LCLK	NC
K6	PE5/$\overline{\text{LRESET}}$	NC
H6	PE4/$\overline{\text{LFRAME}}$	NC
L7	PE3/LAD3	NC
K7	PE2/LAD2	NC
J7	PE1/LAD1	NC
L8	PE0/LAD0	NC
H7	AVSS	VSS
K8	P70/$\overline{\text{ExIRQ0}}$/AN0	NC
L9	P71/$\overline{\text{ExIRQ1}}$/AN1	NC
J8	NC	NC
K9	P72/$\overline{\text{ExIRQ2}}$/AN2	NC
L10	P73/$\overline{\text{ExIRQ3}}$/AN3	NC

Pin No.	Pin Name	
BP-112	Single-Chip Mode	Flash Memory Programmer Mode
J9	NC	NC
L11	P74/$\overline{\text{ExIRQ4}}$/AN4	NC
K10	P75/$\overline{\text{ExIRQ5}}$/AN5	NC
K11	P76/$\overline{\text{ExIRQ6}}$/AN6	NC
H8	P77/$\overline{\text{ExIRQ7}}$/AN7	NC
J10	AVCC	VCC
J11	AVref	VCC
H9	P60	NC
H10	P61	NC
H11	P62	NC
G8	P63	NC
G9	VCC	VCC
G11	ETMS	NC
G10	NC	NC
F9	ETDO	NC
F11	ETDI	NC
F10	ETCK	NC
F8	$\overline{\text{ETRST}}$	$\overline{\text{RES}}$
E11	VSS	NC
E10	P23	FA11
E9	P22	FA10
D11	P21	OE
E8	P20	FA8
D10	P17	FA7
C11	P16	FA6
D9	NC	NC
C10	P15	FA5
B11	P14	FA4
C9	NC	NC

Pin No.	Pin Name	
BP-112	Single-Chip Mode	Flash Memory Programmer Mode
A11	P13	FA3
B10	P12	FA2
A10	P11	FA1
D8	VSS	VSS
B9	P10	FA0
A9	P30/ExDB0	FO0
C8	P31/ExDB1	FO1
B8	P32/ExDB2	FO2
A8	P33/ExDB3	FO3
D7	P34/ExDB4	FO4
C7	P35/ExDB5	FO5
A7	P36/ExDB6	FO6
B7	NC	NC
C6	P37/ExDB7	FO7
A6	P40/$\overline{\text{IRQ0}}$/RS0/HC0	NC
B6	P41/$\overline{\text{IRQ1}}$/RS1/HC1	NC
D6	P42/$\overline{\text{IRQ2}}$/RS2/HC2	NC
A5	P43/$\overline{\text{IRQ3}}$/RS3/HC3	NC
B5	P52/$\overline{\text{IRQ10}}$/TxD1	VCC
C5	P53/$\overline{\text{IRQ11}}$/RxD1	VSS
A4	FWE	FWE
D5	P44/$\overline{\text{IRQ4}}$/RS4/DB4/HC4	FA12
B4	VSS	VSS
A3	$\overline{\text{RESO}}$	NC
C4	NC	NC
B3	XTAL	XTAL
A2	EXTAL	EXTAL
C3	NC	NC

1.3.3 Pin Functions

Table 1.2 Pin Functions

		Pin No.		
Type	**Symbol**	**BP-112**	**I/O**	**Name and Function**
Power supply	VCC	A1, J1, G9	Input	Power supply pins. Connect all these pins to the system power supply. Connect the bypass capacitor between VCC and VSS (near VCC).
	VCL	E1	Input	External capacitance pin for internal step-down power. Connect this pin to Vss through an external capacitor (that is located near this pin) to stabilize internal step-down power.
	VSS	D3, H4, E11, D8, B4	Input	Ground pins. Connect all these pins to the system power supply (0V).
Clock	XTAL	B3	Input	For connection to a crystal resonator. An external clock can be supplied from the EXTAL pin. For an example of crystal resonator connection, see section 21, Clock Pulse Generator.
	EXTAL	A2	Input	
	ϕ	C2	Output	Supplies the system clock to external devices.
	EXCL	C2	Input	32.768-kHz external clock for sub clock should be supplied.
Operating mode control	$\overline{\text{MD2}}$	F3	Input	These pins set the operating mode. Inputs at these pins should not be changed during operation.
	MD1	D1		
System control	$\overline{\text{RES}}$	D2	Input	Reset pin. When this pin is low, the chip is reset.
	$\overline{\text{RESO}}$	A3	Output	Outputs a reset signal to an external device.
	$\overline{\text{STBY}}$	E3	Input	When this pin is low, a transition is made to hardware standby mode.
	FWE	A4	Input	Pin for use by flash memory.

		Pin No.		
Type	**Symbol**	**BP-112**	**I/O**	**Name and Function**
Interrupts	NMI	E4	Input	Nonmaskable interrupt request input pin
	$\overline{\mathrm{IRQ15}}$, $\overline{\mathrm{IRQ14}}$, $\overline{\mathrm{IRQ11}}$, $\overline{\mathrm{IRQ10}}$, $\overline{\mathrm{IRQ7}}$ to $\overline{\mathrm{IRQ0}}$	C1, C2, C5, B5, D4, B1, B2, D5, A5, D6; B6, A6	Input	These pins request a maskable interrupt. Selectable to which pin of $\overline{\mathrm{IRQn}}$ or $\overline{\mathrm{ExIRQn}}$ to insert IRQ15 to IRQ0 interrupts.
	$\overline{\mathrm{ExIRQ15}}$ to $\overline{\mathrm{ExIRQ0}}$	K3, L3, J4, K4, L4, H5, J5, L5, H8, K11, K10, L11, L10, K9, L9, K8		
Boundary scan	$\overline{\mathrm{ETRST}}$	F8	Input	Boundary scan interface pins
	ETMS	G11	Input	
	ETDO	F9	Output	
	ETDI	F11	Input	
	ETCK	F10	Input	
14-bit PWM timer (PWMX)	PWX0 PWX1 PWX2 PWX3	C2 C1 F2 F1	Output	PWM D/A pulse output pins
Serial communication interface (SCI_1 and SCI_3)	TxD1, TxD3	B5, K3	Output	Transmit data output pins
	RxD1, RxD3	C5, L3	Input	Receive data input pins
	SCK1, SCK3	J4, K4	Input/ Output	Clock input/output pins.
I^{2}C bus interface (IIC)	SCL0, SCL1, SCL2, SCL3	L5, H5, G3, G1	Input/ Output	IIC clock input/output pins. These pins can drive a bus directly with the NMOS open drain output.
	SDA0, SDA1, SDA2, SDA3,	J5, L4, G2, F4	Input/ Output	IIC data input/output pins. These pins can drive a bus directly with the NMOS open drain output.

		Pin No.		
Type	**Symbol**	**BP-112**	**I/O**	**Name and Function**
A/D converter	AN7 to AN0	H8, K11, K10, L11, L10, K9, L9, K8	Input	Analog input pins
	AVCC	J10	Input	Analog power supply pins for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, these pins should be connected to the system power supply (+3.3 V).
	AVref	J11	Input	Reference voltage input pin for the A/D converter and D/A converter. When the A/D converter and D/A converter are not used, this pin should be connected to the system power supply (+3.3 V).
	AVSS	H7	Input	Ground pins for the A/D converter and D/A converter. These pins should be connected to the system power supply (0 V).
	$\overline{\text{ADTRG}}$	K3	Input	External trigger input pin to start A/D conversion
LPC interface (LPC)	LAD3 to LAD0	L7, K7, J7, L8	Input/ Output	Transfer cycle type/address/data I/O pins
	$\overline{\text{LFRAME}}$	H6	Input	Input pin indicating transfer cycle start and forced termination
	$\overline{\text{LRESET}}$	K6	Input	LPC reset pin. When this pin is low, a reset state is entered.
	LCLK	L6	Input	PCI clock input pin
	SERIRQ	J6	Input/ Output	LPC serialized host interrupt request signal
Event counter	EVENT7 to EVENT0	H1, G4, H2, J2, K1, L1, K2, L2	Input	Event counter input pins.
Retain state output pins	RS7 to RS0	D4, B1, B2, D5, A5, D6, B6, A6	Output	The outputs on these pins are only initialized by a system reset.
Debounced input pins	DB7 to DB4	D4, B1, B2, D5	Input	Pins with a noise eliminating function.
	ExDB7 to ExDB0	C6, A7, C7, D7, A8, B8, C8, A9		

		Pin No.		
Type	**Symbol**	**BP-112**	**I/O**	**Name and Function**
Large current output pins	HC7 to HC0	D4, B1, B2, D5, A5, D6, B6, A6	Output	These pins can be used to drive LEDs or for other purposes where large currents are required.
I/O ports	P17 to P10	D10, C11, C10, B11, A11, B10, A10, B9	Input/ Output	8-bit input/output pins
	P23 to P20	E10, E9, D11, E8	Input/ Output	4-bit input/output pins
	P37 to P30	C6, A7, C7, D7, A8, B8, C8, A9	Input/ Output	8-bit input/output pins
	P47 to P40	D4, B1, B2, D5, A5, D6, B6, A6	Input/ Output	8-bit input/output pins
	P57, P56, P53, P52	C1, C2, C5, B5	Input/ Output	4-bit input/output pins
	P63 to P60	G8, H11, H10, H9	Input/ Output	4-bit input/output pins
	P77 to P70	H8, K11, K10, L11, L10, K9, L9, K8	Input	8-bit input pins
	P87 to P80	K3, L3, J4, K4, L4, H5, J5, L5	Input/ Output	8-bit input/output pins
	PA7 to PA0	H1, G4, H2, J2, K1, L1, K2, L2	Input/ Output	8-bit input/output pins
	PC7, PC6, PC3 to PC0	F1, F2, F4, G1, G2, G3	Input/ Output	6-bit input/output pins
	PE7 to PE0	J6, L6, K6, H6, L7, K7, J7, L8	Input/ Output	8-bit input/output pins

Section 2 CPU

The H8S/2600 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/2600 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/2600 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-compatible with H8/300 and H8/300H CPUs
  - Can execute H8/300 and H8/300H CPUs object programs
- General-register architecture
  - Sixteen 16-bit general registers also usable as sixteen 8-bit registers or eight 32-bit registers
- Sixty-nine basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
  - Multiply-and-accumulate instruction
- 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 execute in one or two states
  - 8/16/32-bit register-register add/subtract: 1 state
  - 8 × 8-bit register-register multiply: 2 states

- 16 ÷ 8-bit register-register divide: 12 states
- 16 × 16-bit register-register multiply: 3 states
- 32 ÷ 16-bit register-register divide: 20 states

- Two CPU operating modes
  - Normal mode*
  - Advanced mode
- Power-down state
  - Transition to power-down state by the SLEEP instruction
  - CPU clock speed selection

Note: * Normal mode is not available in this LSI.

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 shown below.

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

		Execution States	
Instruction	Mnemonic	H8S/2600	H8S/2000
MULXU	MULXU.B Rs, Rd	2*	12
	MULXU.W Rs, ERd	2*	20
MULXS	MULXS.B Rs, Rd	3*	13
	MULXS.W Rs, ERd	3*	21
CLRMAC	CLRMAC	1*	Not supported
LDMAC	LDMAC ERs,MACH	1*	
	LDMAC ERs,MACL	1*	
STMAC	STMAC MACH,ERd	1*	
	STMAC MACl,ERd	1*	

Note: * This becomes one state greater immediately after a MAC instruction.
In addition, there are differences in address space, CCR and EXR register functions, and 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/2600 CPU has the following enhancements:

- More general registers and control registers
  - Eight 16-bit extended registers, and one 8-bit and two 32-bit control registers, 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.
  - A multiply-and-accumulate instruction has been added.
  - Two-bit shift 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 execute twice as fast.

2.1.3 Differences from H8/300H CPU

In comparison to the H8/300H CPU, the H8S/2600 CPU has the following enhancements:

- More control registers
  - One 8-bit and two 32-bit control registers have been added.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - A multiply-and-accumulate instruction has been added.
  - Two-bit shift 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 execute twice as fast.

2.2 CPU Operating Modes

The H8S/2600 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte total address space. The mode is selected by the mode pins.

2.2.1 Normal Mode

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

- Address Space

  Linear access to a 64-kbyte maximum address space is provided.

- 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 En is used as a 16-bit register it can contain any value, even when the corresponding general register (Rn) is used as an address register. If the general register is referenced in the register indirect addressing mode with pre-decrement (@–Rn) or post-increment (@Rn+) and a carry or borrow occurs, however, the value in the corresponding extended register (En) will be affected.

- Instruction Set

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

- Exception Vector Table and Memory Indirect Branch Addresses

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

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

- Stack Structure

  When the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.2. EXR is not pushed onto the stack in interrupt control mode 0. For details, see section 4, Exception Handling.

Note: Normal mode is not available in this LSI.

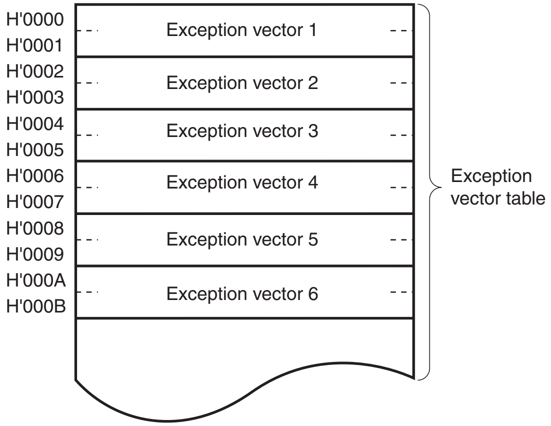
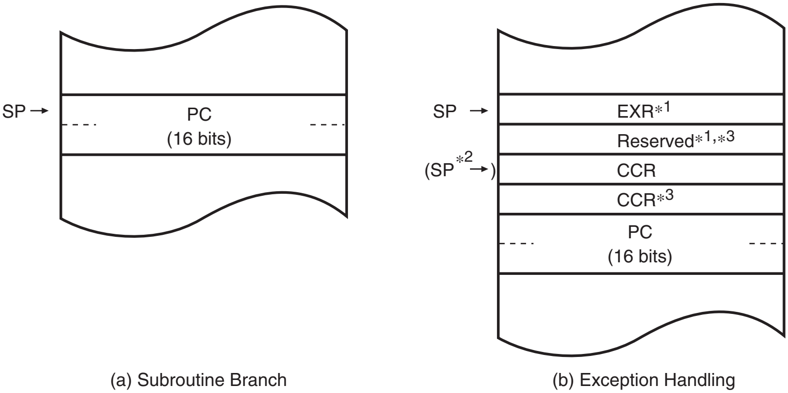


Figure 2.1 Exception Vector Table (Normal Mode)



Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

Figure 2.2 Stack Structure in Normal Mode

2.2.2 Advanced Mode

- Address Space

  Linear access to a 16-Mbyte maximum address space is provided.

- Extended Registers (En)

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

- Instruction Set

  All instructions and addressing modes can be used.

- Exception Vector Table and Memory Indirect Branch Addresses

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

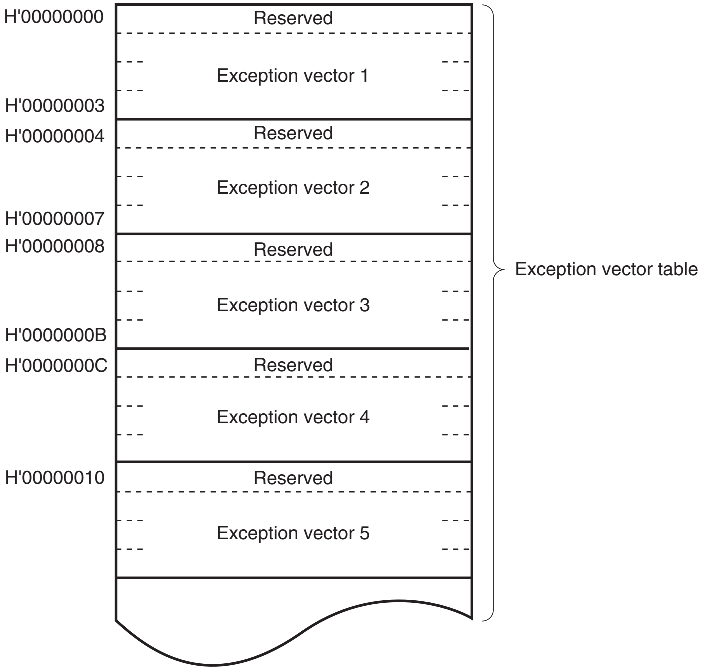
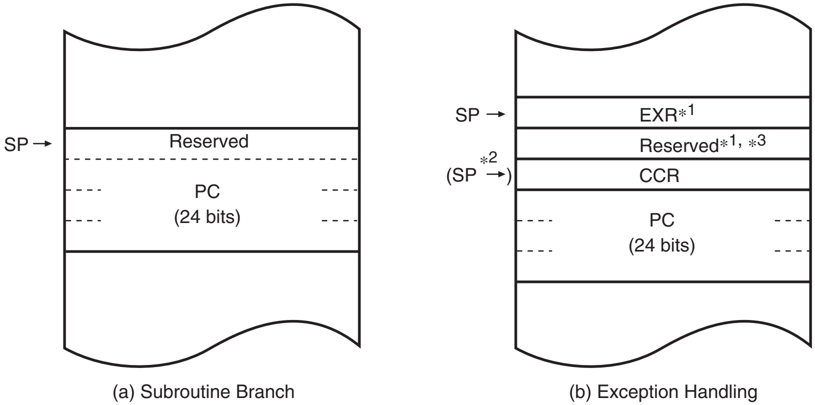


Figure 2.3 Exception Vector Table (Advanced Mode)

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In advanced mode the operand is a 32-bit longword operand, providing a 32-bit branch address. The upper 8 bits of these 32 bits is 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 first part 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, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.4. When EXR is not pushed onto the stack in interrupt control mode 0. For details, see section 4, Exception Handling.



Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

Figure 2.4 Stack Structure in Advanced Mode

2.3 Address Space

Figure 2.5 shows a memory map for the H8S/2600 CPU. The H8S/2600 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.

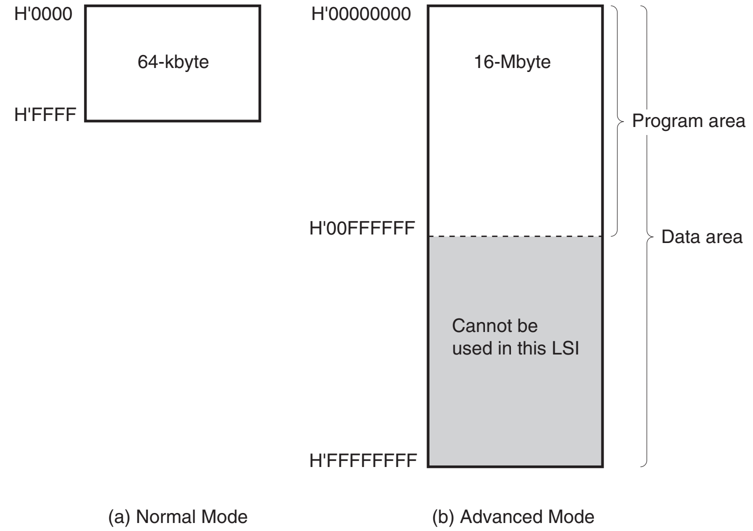
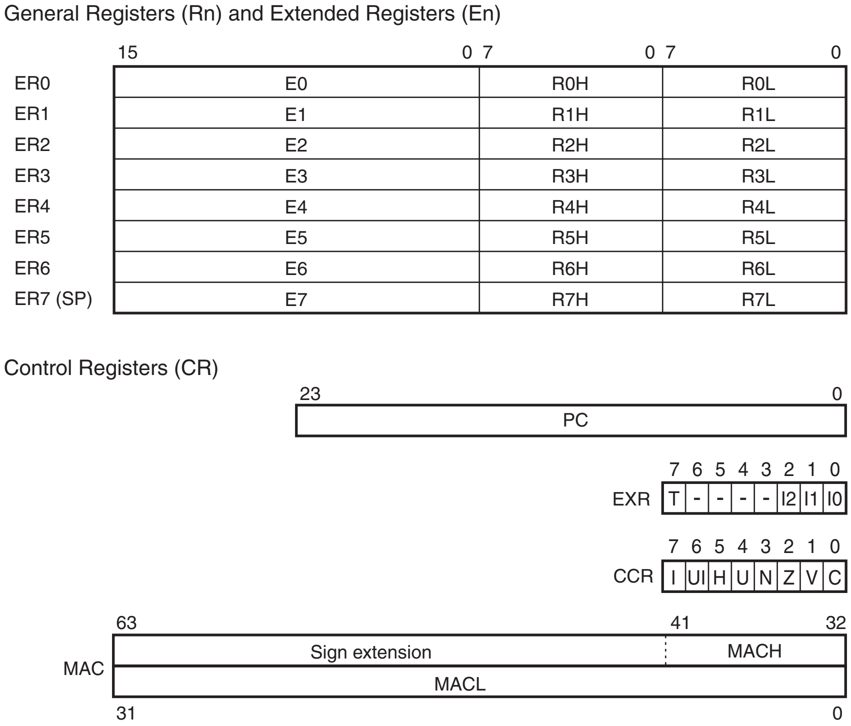


Figure 2.5 Memory Map

2.4 Registers

The H8S/2600 CPU has the internal registers shown in figure 2.6. There are two types of registers; general registers and control registers. The control registers are a 24-bit program counter (PC), an 8-bit extended control register (EXR), an 8-bit condition code register (CCR), and a 64-bit multiply-accumulate register (MAC).



[Legend]

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

Figure 2.6 CPU Registers

2.4.1 General Registers

The H8S/2600 CPU has eight 32-bit general registers. These general registers are all functionally identical 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).

The ER registers divide 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 of sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

The R registers divide 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 of sixteen 8-bit registers.

The usage of each register can be selected independently.

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

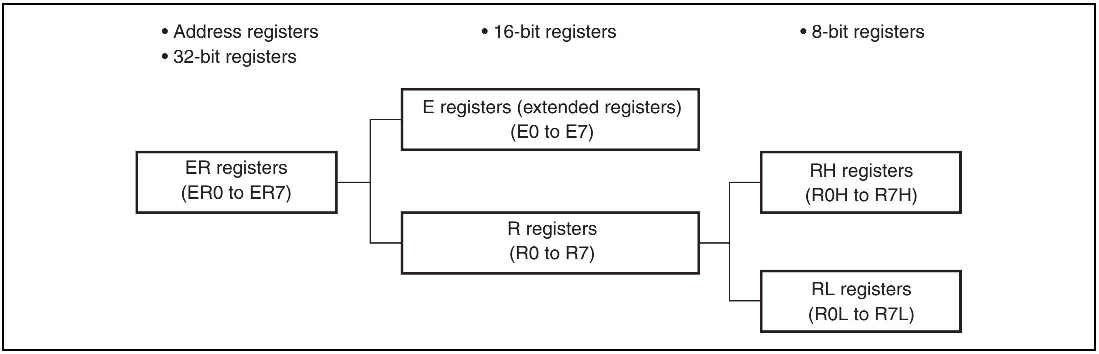

• Address registers
• 32-bit registers
• 16-bit registers
• 8-bit registers
ER registers (ER0 to ER7)
E registers (extended registers) (E0 to E7)
R registers (R0 to R7)
RH registers (R0H to R7H)
RL registers (R0L to R7L)

Figure 2.7 Usage of General Registers

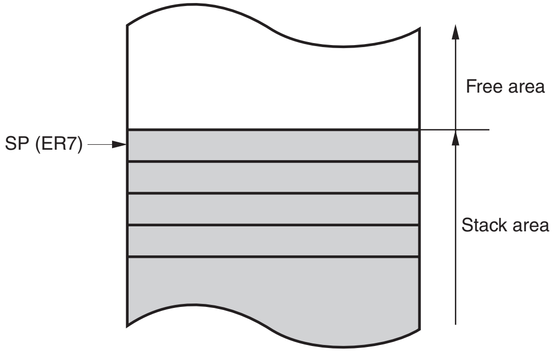


Figure 2.8 Stack

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, the least significant PC bit is regarded as 0).

2.4.3 Extended Control Register (EXR)

EXR is an 8-bit register that manipulates the LDC, STC, ANDC, ORC, and XORC instructions. When these instructions, except for the STC instruction, are executed, all interrupts including NMI will be masked for three states after execution is completed.

Bit	Bit Name	Initial Value	R/W	Description
7	T	0	R/W	Trace Bit When this bit is set to 1, a trace exception is generated each time an instruction is executed. When this bit is cleared to 0, instructions are executed in sequence.
6 to 3	—	All 1	—	Reserved These bits are always read as 1.
2	I2	1	R/W	These bits designate the interrupt mask level (0 to 7). For details, refer to section 5, Interrupt Controller.
1	I1	1	R/W	
0	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 read or written by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit cannot be used as an interrupt mask bit in this LSI.
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 read or written 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.

Bit	Bit Name	Initial Value	R/W	Description
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.
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 Multiply-Accumulate Register (MAC)

This 64-bit register stores the results of multiply-and-accumulate operations. It consists of two 32-bit registers denoted MACH and MACL. The lower 10 bits of MACH are valid; the upper bits are a sign extension.

2.4.6 Initial Values of CPU Registers

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

2.5 Data Formats

The H8S/2600 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 in general registers.

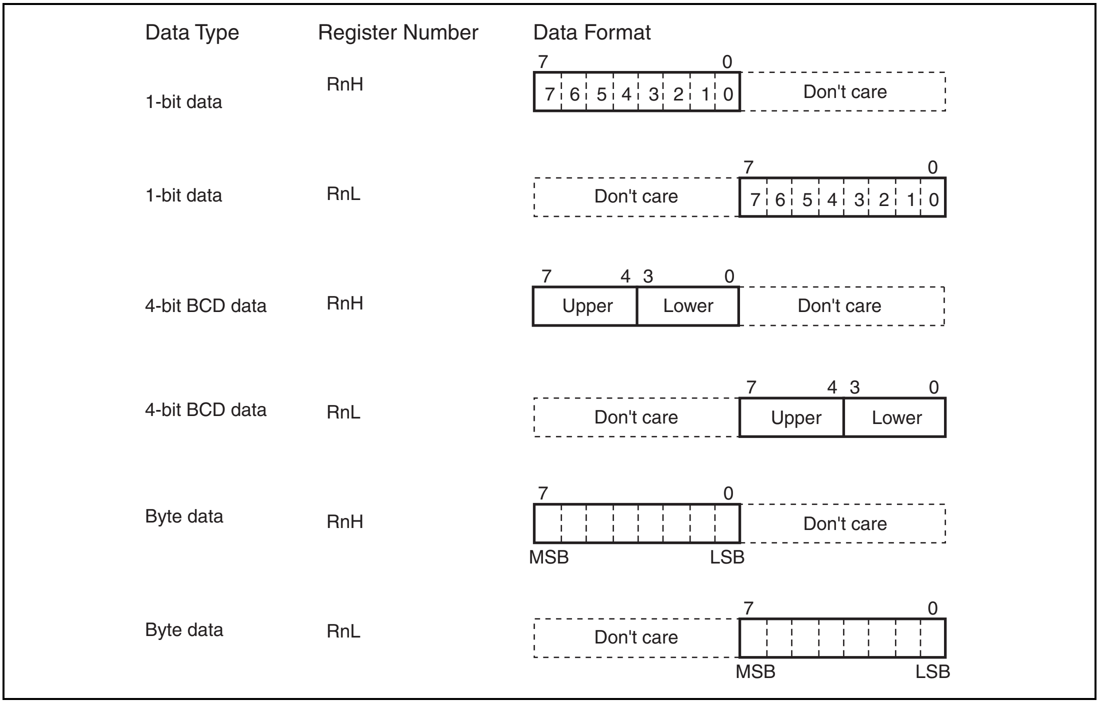
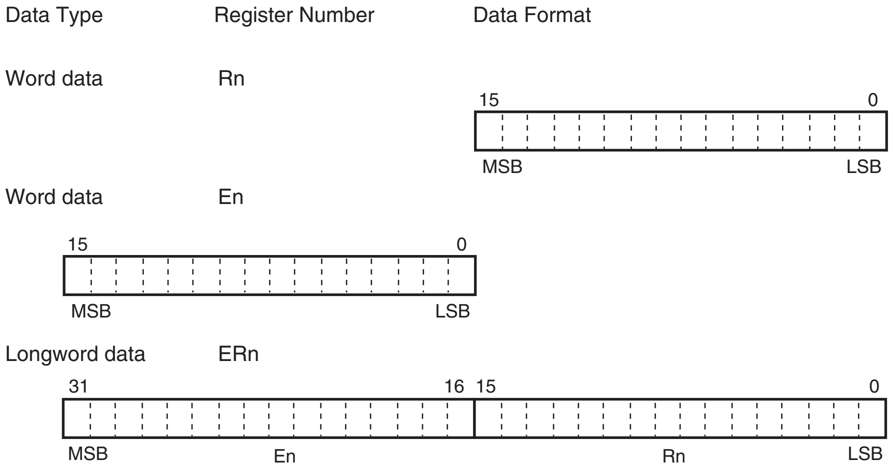


Figure 2.9 General Register Data Formats (1)



[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/2600 CPU can access word data and longword data in memory, however 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, an address error does not occur, however the least significant bit of the address is regarded as 0, so access begins the preceding address. This also applies to instruction fetches.

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

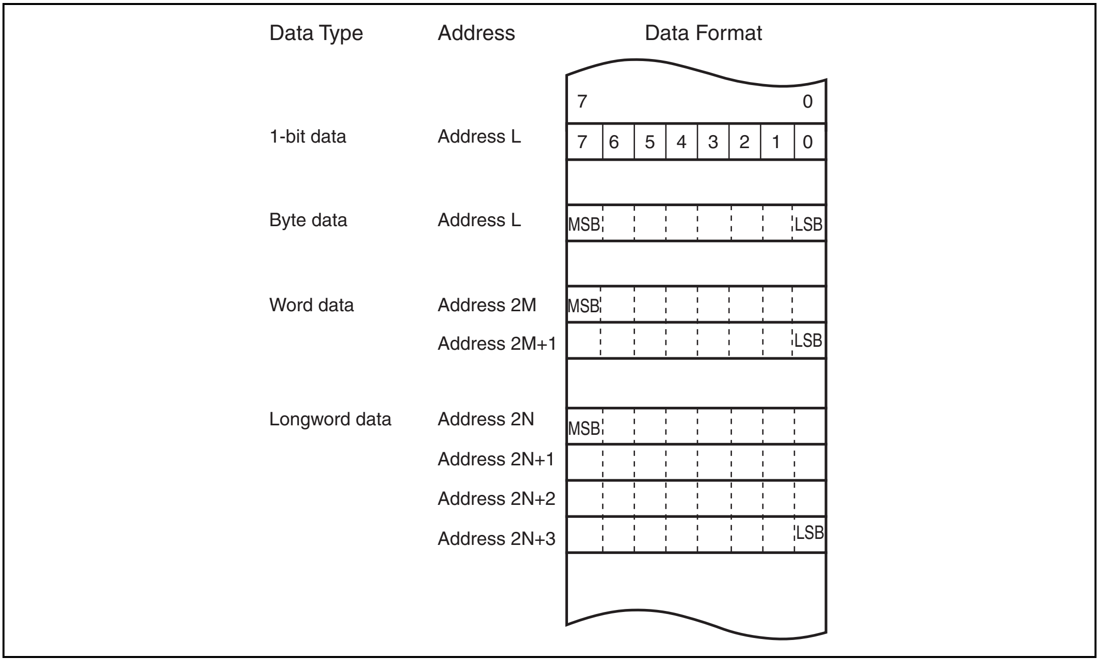


Figure 2.10 Memory Data Formats

2.6 Instruction Set

The H8S/2600 CPU has 69 instructions. The instructions are classified by function 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, STM	L	
	MOVFPE*[3], MOVTPE*[3]	B	
Arithmetic operation	ADD, SUB, CMP, NEG	B/W/L	23
	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	
	MAC, LDMAC, STMAC, CLRMAC	—	
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	Bcc*[2], JMP, BSR, JSR, RTS	—	5
System control	TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	—	9
Block data transfer	EEPMOV	—	1

Total: 69

Notes: B-byte; W-word; L-longword.

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. Bcc is the general name for conditional branch instructions.
3. Cannot be used in this LSI.
4. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

2.6.1 Table of Instructions Classified by Function

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

Table 2.2 Operation Notation

Symbol	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
MAC	Multiply-accumulate 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 XOR
→	Move
~	NOT (logical complement)

Symbol	Description
: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*	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	L	@SP+ → Rn (register list) Pops two or more general registers from the stack.
STM	L	Rn (register list) → @–SP Pushes two or more general registers onto the stack.

Note: * Refers to the operand size.

B: Byte

W: Word

L: Longword

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 (immediate byte data cannot be subtracted from byte data in a general register. Use the SUBX or ADD instruction.)
ADDX SUBX	B	Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd Performs addition or subtraction with carry on byte 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 Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.)
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 the 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: * 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 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.
MAC	—	(EAs) × (EAd) + MAC → MAC Performs signed multiplication on memory contents and adds the result to the multiply-accumulate register. The following operations can be performed: 16 bits × 16 bits + 32 bits → 32 bits, saturating 16 bits × 16 bits + 42 bits → 42 bits, non-saturating
CLRMAC	—	0 → MAC Clears the multiply-accumulate register to zero.
LDMAC STMAC	L	Rs → MAC, MAC → Rd Transfers data between a general register and a multiply-accumulate register.

Note: 1. Refers to the operand size.

B: Byte

W: Word

L: Longword

2. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

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 general register contents.

Note: * 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 general register contents. 1-bit or 2-bit shifts are possible.
SHLL SHLR	B/W/L	Rd (shift) → Rd Performs a logical shift on general register contents. 1-bit or 2-bit shifts are possible.
ROTL ROTR	B/W/L	Rd (rotate) → Rd Rotates general register contents. 1-bit or 2-bit rotations are possible.
ROTXL ROTXR	B/W/L	Rd (rotate) → Rd Rotates general register contents through the carry flag. 1-bit or 2-bit rotations are possible.

Note: * 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 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 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 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 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: * 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 XORs 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 XORs 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: * 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 general register or memory 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. 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 $\wedge$ #IMM → CCR, EXR $\wedge$ #IMM → EXR Logically ANDs the CCR or EXR contents with immediate data.
ORC	B	CCR $\vee$ #IMM → CCR, EXR $\vee$ #IMM → EXR Logically ORs the CCR or EXR contents with immediate data.
XORC	B	CCR $\oplus$ #IMM → CCR, EXR $\oplus$ #IMM → EXR Logically XORs the CCR or EXR contents with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

Note: * 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/2600 CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

Figure 2.11 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. 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.

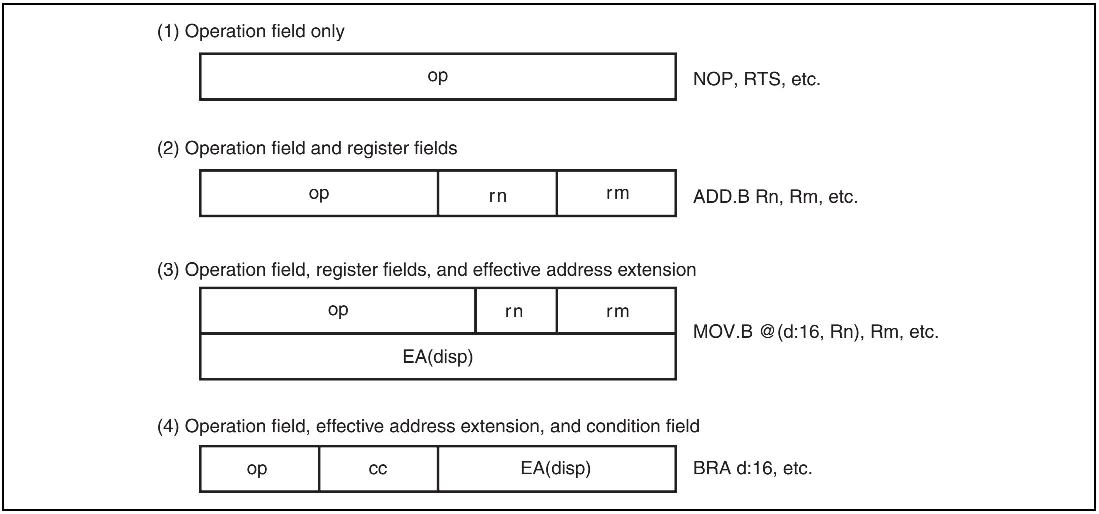

(1) Operation field only
op
NOP, RTS, etc.
(2) Operation field and register fields
op
rn
rm
ADD.B Rn, Rm, etc.
(3) Operation field, register fields, and effective address extension
op
rn
rm
EA(disp)
MOV.B @(d:16, Rn), Rm, etc.
(4) Operation field, effective address extension, and condition field
op
cc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

Figure 2.11 Instruction Formats (Examples)

2.7 Addressing Modes and Effective Address Calculation

The H8S/2600 CPU supports the eight addressing modes listed in table 2.11. Each instruction uses a subset of these addressing modes. Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit manipulation instructions use register direct, register indirect, or the 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 Register indirect with pre-decrement	@ERn+ @–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 specifies an 8-, 16-, or 32-bit general register containing 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 the operand on memory. 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 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 transfer instruction, or 4 for longword transfer instruction. For the 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 is 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 transfer instruction, or 4 for longword transfer instruction. For the 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 24 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address the upper 16 bits are a sign extension. A 32-bit absolute address can access the entire address space.

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)		

Note: Normal mode is not available in this LSI.

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

The instruction contains 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data as an operand.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. 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 is used in the Bcc and BSR instructions. An 8-bit or 16-bit displacement contained in the instruction is sign-extended and added to the 24-bit PC contents to generate a 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.

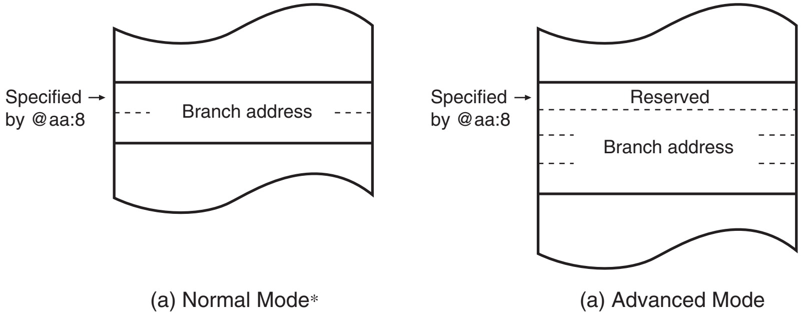
2.7.8 Memory Indirect—@@aa:8

This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand. This memory operand contains a branch address. The upper bits of the 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 first part of the address range is also 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 instruction code to be fetched at the address preceding the specified address. (For further information, see section 2.5.2, Memory Data Formats.)

Note: Normal mode is not available in this LSI.



Note: * Normal mode is not available in this LSI.

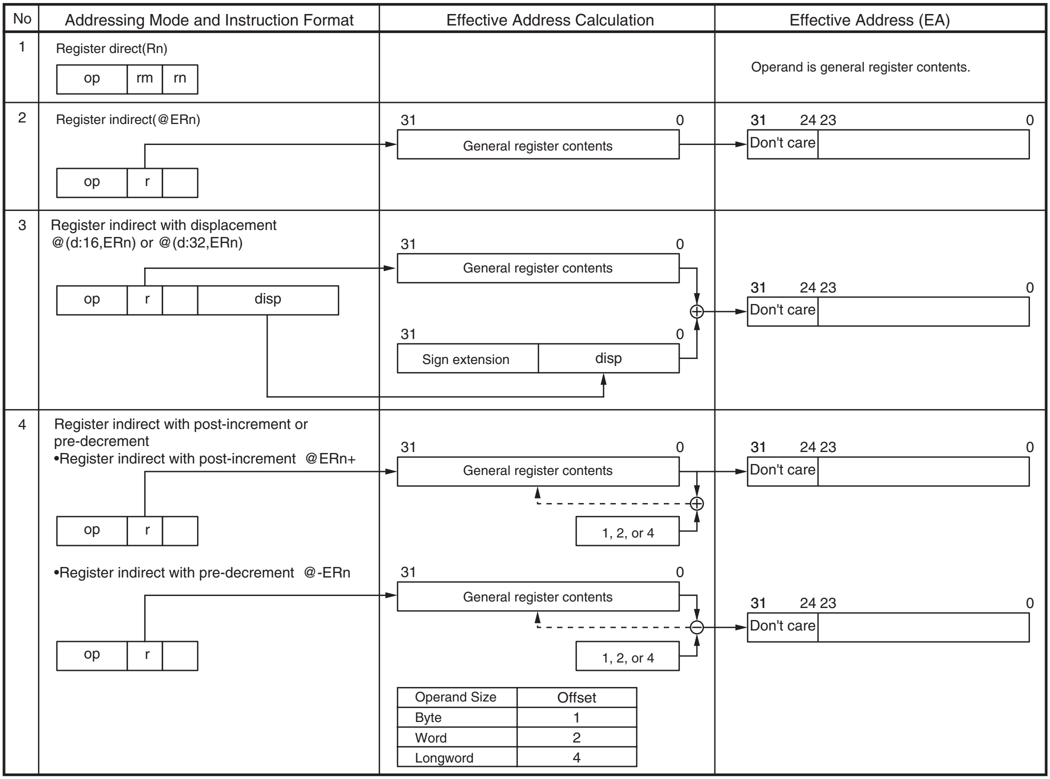
Figure 2.12 Branch Address Specification in Memory Indirect Mode

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.

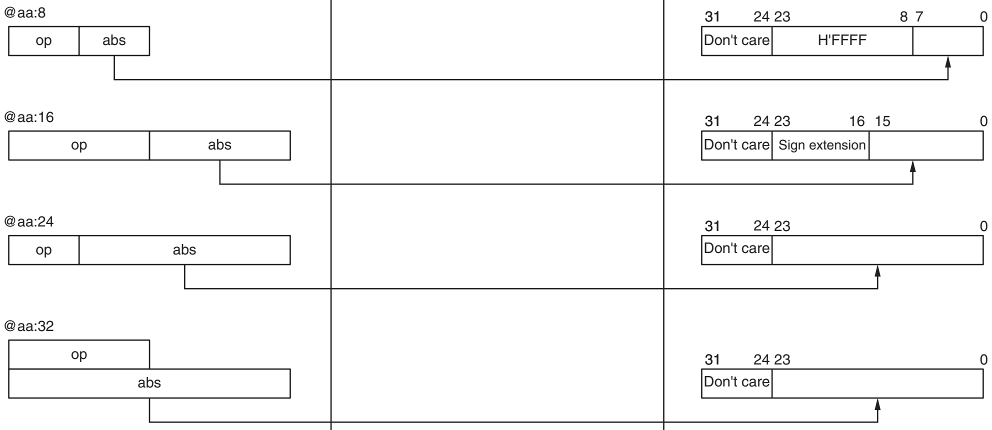
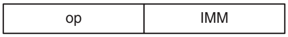
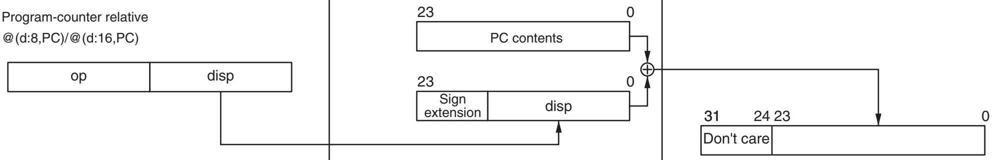
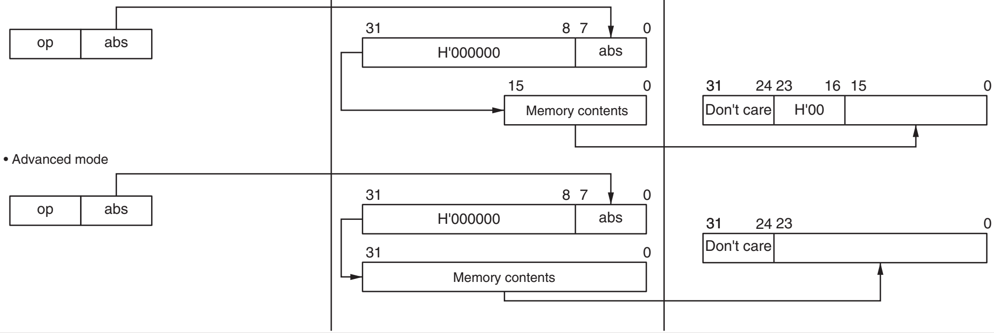
Note: Normal mode is not available in this LSI.

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		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 • Normal mode*	 op abs 31 8 7 0 H'000000 abs 15 0 Memory contents 31 24 23 16 15 0 Don't care H'00 • Advanced mode op abs 31 8 7 0 H'000000 abs 31 0 Memory contents 31 24 23 0 Don't care	

Note: * Normal mode is not available in this LSI.

2.8 Processing States

The H8S/2600 CPU has four main processing states: the reset state, exception handling state, program execution state and power-down state. Figure 2.13 indicates the state transitions.

- Reset State

  In this state, the CPU and all on-chip peripheral modules are initialized and not operating. 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

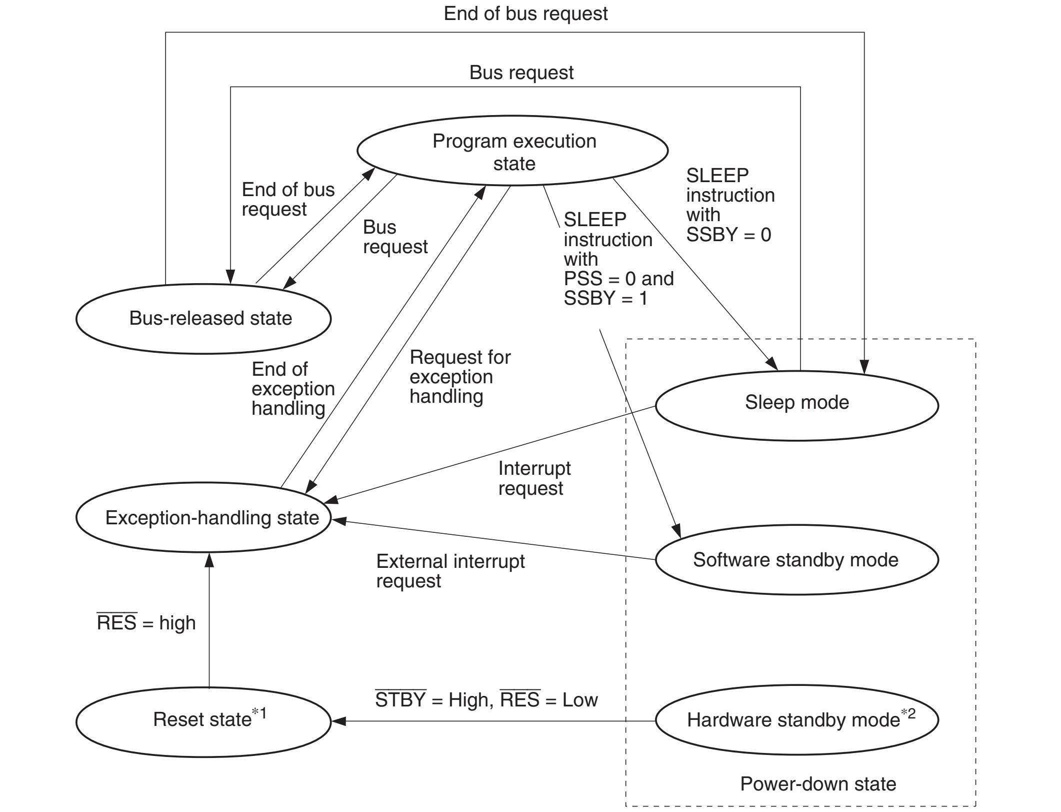
  The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to an exception source, such as a reset, trace, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception vector table and branches to that address. For further 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 software standby mode. For further details, refer to section 22, Power-Down Modes.



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.

Figure 2.13 State Transitions

2.9 Usage Note

2.9.1 Notes on Using the Bit Operation Instruction

Instructions BSET, BCLR, BNOT, BST, and BIST read data in byte units, and write data in byte units after bit operation. Therefore, attention must be paid when these instructions are used for ports or registers including write-only bits.

Instruction BCLR can be used to clear the flag in the internal I/O register to 0. If it is obvious that the flag has been set to 1 by the interrupt processing routine, it is unnecessary to read the flag beforehand.

Section 3 MCU Operating Modes

3.1 Operating Mode Selection

This MCU supports three operating modes (modes 2, 4, and 6). The operating mode is determined by the setting of the mode pins ($\overline{\text{MD2}}$ and MD1). Table 3.1 shows the MCU operating mode selection.

Table 3.1 MCU Operating Mode Selection

MCU Operating Mode	$\overline{\text{MD2}}$	MD1	CPU Operating Mode	Description
2	1	1	Advanced	Extended mode with on-chip ROM Single-chip mode
4	0	0	—	Flash programming/erasing
6	0	1	Emulation	On-chip emulation mode

Mode 2 is single-chip mode after a reset. The CPU can switch to extended mode by setting bit EXPE in MDCR to 1.

Modes 0, 1, 3, 5, and 7 are not available in this LSI. Modes 4 and 6 are operating mode for a special purpose. Thus, mode pins should be set to enable mode 2 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 set an operating mode and to monitor the current operating mode.

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

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

3.2.2 System Control Register (SYSCR)

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

Bit	Bit Name	Initial Value	R/W	Description
7 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, 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.
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	—	0	R/W	Reserved The initial value should not be changed.
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	IICX2	0	R/W	IIC Transfer Rate Select 2, 1 and 0
6	IICX1	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^2C bus mode register (ICMR). For details on the transfer rate, see table 15.3. The IICXn bit controls IIC_n. (n = 0 to 2)
5	IICX0	0	R/W	
4	—	0	R/W	Reserved The initial value should not be changed.
3	FLSHE	0	R/W	Flash Memory Control Register Enable Enables or disables CPU access for flash memory registers (FCCS, FPCS, FECS, FKEY, FMATS, FTDAR), control registers of power-down states (SBYCR, LPWRCR, MSTPCRH, MSTPCRL), and a control register of on-chip peripheral modules (PCSR). 0: Area from H'FFFE88 to H'FFFE8F is reserved. Control registers of power-down states and on-chip peripheral modules are accessed in an area from H'FFFF80 to H'FFFF87. 1: Control registers of flash memory are accessed in an area from H'FFFE88 to H'FFFE8F. Area from H'FFFF80 to H'FFFF87 is reserved.
2	—	1	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, see section 11.2.4, Timer Control Register (TCR).

3.3 Operating Mode Descriptions

3.3.1 Mode 2

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

3.4 Address Map

Figure 3.1 shows an address map in operating mode.

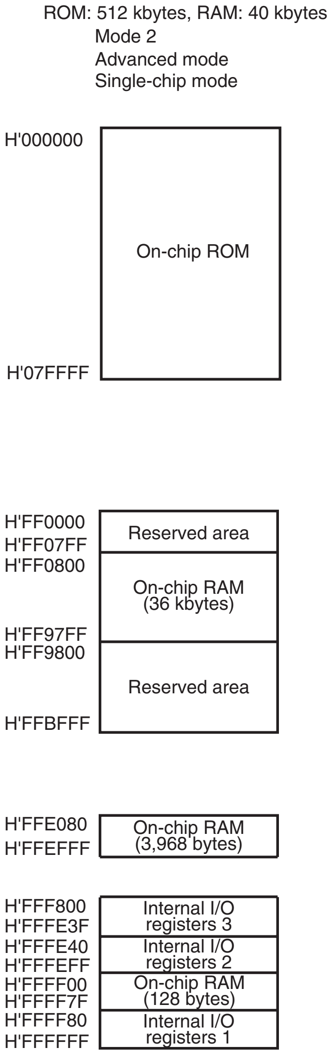


Figure 3.1 Address Map

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.
↑	Illegal instruction	Started by execution of an undefined code.
↑	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.
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**	**Advanced Mode**
Reset		0	H'000000 to H'000003
Reserved for system use		1 | 3	H'000004 to H'000007 | H'00000C to H'00000F
Illegal instruction exception		4	H'000010 to H'000013
Reserved for system use		5	H'000014 to H'000017
		6	H'000018 to H'00001B
External interrupt (NMI)		7	H'00001C to H'00001F
Trap instruction (four sources)		8	H'000020 to H'000023
		9	H'000024 to H'000027
		10	H'000028 to H'00002B
		11	H'00002C to H'00002F
Reserved for system use		12 | 15	H'000030 to H'000033 | H'00003C to H'00003F
External interrupt	IRQ0	16	H'000040 to H'000043
	IRQ1	17	H'000044 to H'000047
	IRQ2	18	H'000048 to H'00004B
	IRQ3	19	H'00004C to H'00004F
	IRQ4	20	H'000050 to H'000053
	IRQ5	21	H'000054 to H'000057
	IRQ6	22	H'000058 to H'00005B
	IRQ7	23	H'00005C to H'00005F

Exception Source		Vector Number	Vector Address Advanced Mode
Internal interrupt*		24 | 29	H'000060 to H'000063 | H'000074 to H'000077
Reserved for system use		30 | 33	H'000078 to H'00007B | H'000084 to H'000087
Internal interrupt*		34 | 55	H'000088 to H'00008B | H'0000DC to H'0000DF
External interrupt	IRQ8	56	H'0000E0 to H'0000E3
	IRQ9	57	H'0000E4 to H'0000E7
	IRQ10	58	H'0000E8 to H'0000EB
	IRQ11	59	H'0000EC to H'0000EF
	IRQ12	60	H'0000F0 to H'0000F3
	IRQ13	61	H'0000F4 to H'0000F7
	IRQ14	62	H'0000F8 to H'0000FB
	IRQ15	63	H'0000FC to H'0000FF
Internal interrupt*		64 | 107	H'000100 to H'000103 | H'0001AC to H'0001AF

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

4.3 Reset

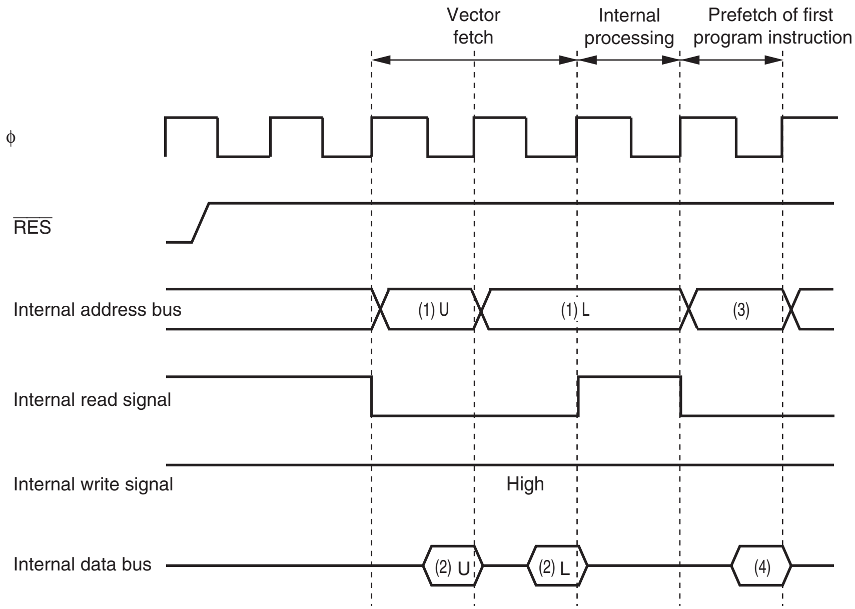
A reset has the highest exception priority. When the $\overline{\text{RES}}$ pin goes low, all processing halts and this LSI enters the reset. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-on. To reset the chip during operation, hold the $\overline{\text{RES}}$ 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 12, 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 in CCR is set to 1.
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) U = H'000000 (1) L = H'000002
(2) Start address (contents of reset exception handling vector address)
(3) Start address ((3) = (2)U + (2)L)
(4) First program instruction

Figure 4.1 Reset Sequence

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, MSTPCRA, and SUBMSTPB) are initialized, and all modules except the DTC operate in module stop mode. Therefore, the registers of on-chip peripheral modules cannot be read from or written to. To read from and write to these registers, clear module stop mode.

4.4 Interrupt Exception Handling

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

Interrupt exception handling is conducted as follows:

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

4.5 Trap Instruction Exception Handling

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

Trap instruction exception handling is conducted as follows:

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

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

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

Table 4.3 Status of CCR after Trap Instruction Exception Handling

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

4.6 Stack Status after Exception Handling

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

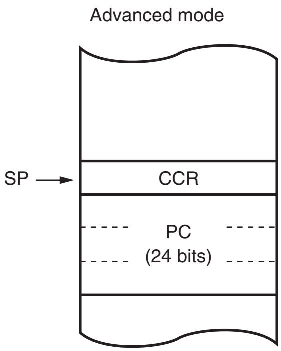


Figure 4.2 Stack Status after Exception Handling

4.7 Usage Note

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

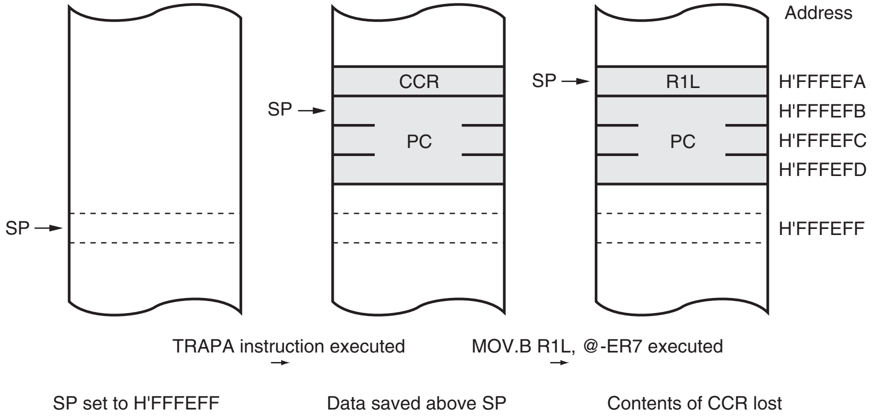
Use the following instructions to save registers:

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

Use the following instructions to restore registers:

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

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



[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.

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. Priority levels can be set for each module for all interrupts except NMI.
- Three-level interrupt mask control

  By means of the interrupt control mode, I and UI bits in CCR, and ICR, 3-level interrupt mask control is performed.
- 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.
- Twenty-nine 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 $\overline{\text{IRQn}}$ (n = 15, 14, 11, 10, and 7 to 0) and $\overline{\text{ExIRQm}}$ (m = 15 to 0).
- DTC control

  The DTC can be activated by an interrupt request.

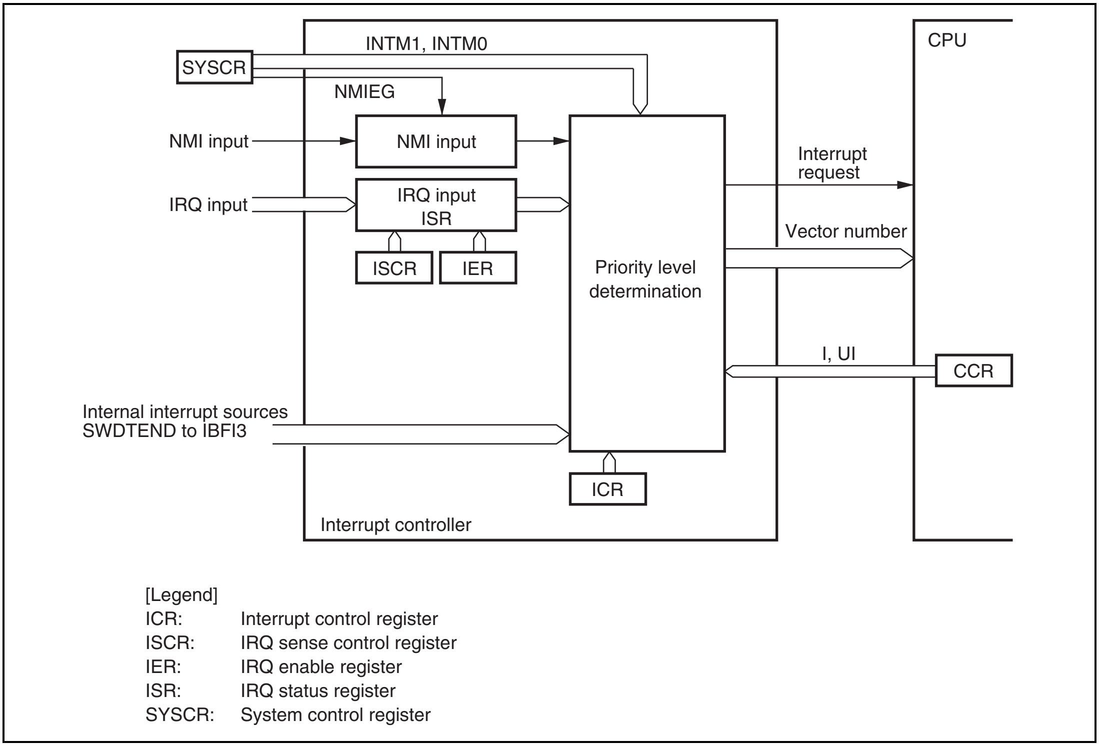


Figure 5.1 Block Diagram of Interrupt Controller

5.2 Input/Output Pins

Table 5.1 summarizes the pins of the interrupt controller.

Table 5.1 Pin Configuration

Symbol	I/O	Function
NMI	Input	Nonmaskable external interrupt Rising edge or falling edge can be selected
$\overline{\text{IRQ15}}$, $\overline{\text{IRQ14}}$, $\overline{\text{IRQ11}}$ $\overline{\text{IRQ10}}$, $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ $\overline{\text{ExIRQ15}}$ to $\overline{\text{ExIRQ0}}$	Input	Maskable external interrupts Rising edge, falling edge, or both edges, or level sensing can be selected individually for each pin. Pin of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ to input $\overline{\text{IRQn}}$ (n = 15, 14, 11, 10, and 7 to 0) interrupt can be selected.

5.3 Register Descriptions

The interrupt controller has the following registers. For details on the system control register (SYSCR), see section 3.2.2, System Control Register (SYSCR), and for details on the IRQ sense port select registers (ISSR16 and ISSR), see section 8.12.1, IRQ Sense Port Select Register 16 (ISSR16), IRQ Sense Port Select Register (ISSR).

- Interrupt control registers A to D (ICRA to ICRD)
- Address break control register (ABRKCR)
- Break address registers A to C (BARA to BARC)
- IRQ sense control registers (ISCR16H, ISCR16L, ISCRH, and ISCRL)
- IRQ enable registers (IER16 and IER)
- IRQ status registers (ISR16 and ISR)

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

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

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

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

[Legend]

n: A to D

Table 5.2 Correspondence between Interrupt Source and ICR

		Register			
Bit	**Bit Name**	**ICRA**	**ICRB**	**ICRC**	**ICRD**
7	ICRn7	IRQ0	A/D converter	SCI_3	IRQ8 to IRQ11
6	ICRn6	IRQ1	FRT	SCI_1	IRQ12 to IRQ15
5	ICRn5	IRQ2, IRQ3	—	—	—
4	ICRn4	IRQ4, IRQ5	TMR_X	IIC_0	—
3	ICRn3	IRQ6, IRQ7	TMR_0	IIC_1	—
2	ICRn2	DTC	TMR_1	IIC_2, IIC_3	—
1	ICRn1	WDT_0	TMR_Y	LPC	—
0	ICRn0	WDT_1	—	—	—

[Legend]]

n: A to D

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

5.3.2 Address Break Control Register (ABRKCR)

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

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

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

The BAR registers specify an address that is to be a break address. An address in which the first byte of an instruction exists should be set as a break address.

- 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 (ISCR16H, ISCR16L, ISCRH, ISCRL)

The ISCR registers select the source that generates an interrupt request at pins $\overline{\text{IRQ15}}$, $\overline{\text{IRQ14}}$, $\overline{\text{IRQ11}}$, $\overline{\text{IRQ10}}$, and $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ or pins $\overline{\text{ExIRQ15}}$ to $\overline{\text{ExIRQ0}}$.

- ISCR16H

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ15SCB	0	R/W	IRQn Sense Control B IRQn Sense Control A 00: Interrupt request generated at low level of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input (n = 15 to 12) Note: * $\overline{\text{IRQn}}$ here only stands for $\overline{\text{IRQ15}}$ and $\overline{\text{IRQ14}}$.
6	IRQ15SCA	0	R/W	
5	IRQ14SCB	0	R/W	
4	IRQ14SCA	0	R/W	
3	IRQ13SCB	0	R/W	
2	IRQ13SCA	0	R/W	
1	IRQ12SCB	0	R/W	
0	IRQ12SCA	0	R/W	

Note: For n = 13 or 12, only $\overline{\text{ExIRQ}}$ can be selected.

- ISCR16L

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ11SCB	0	R/W	IRQn Sense Control B IRQn Sense Control A 00: Interrupt request generated at low level of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input (n = 11 to 8) Note: * $\overline{\text{IRQn}}$ here only stands for $\overline{\text{IRQ11}}$ and $\overline{\text{IRQ10}}$.
6	IRQ11SCA	0	R/W	
5	IRQ10SCB	0	R/W	
4	IRQ10SCA	0	R/W	
3	IRQ9SCB	0	R/W	
2	IRQ9SCA	0	R/W	
1	IRQ8SCB	0	R/W	
0	IRQ8SCA	0	R/W	

Note: For n = 9 or 8, only $\overline{\text{ExIRQ}}$ can be selected.

- ISCRH

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7SCB	0	R/W	IRQn Sense Control B IRQn Sense Control A 00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input (n = 7 to 4)
6	IRQ7SCA	0	R/W	
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 $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ or $\overline{\text{ExIRQn}}$ 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 Registers (IER16, IER)

The IER registers control the enabling and disabling of interrupt requests IRQ15 to IRQ0.

- IER16

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	IRQ15E to IRQ8E	All 0	R/W	IRQn Enable (n = 15 to 8) The IRQn interrupt request is enabled when this bit is 1.

Note: * IRQn stands for IRQ15, IRQ14, IRQ11, and IRQ10.

- IER

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	IRQ7E to IRQ0E	All 0	R/W	IRQn Enable (n = 7 to 0) The IRQn interrupt request is enabled when this bit is 1.

5.3.6 IRQ Status Registers (ISR16, ISR)

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

- ISR16

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	IRQ15F to IRQ8F	All 0	R/W	[Setting condition] • When the interrupt source selected by the ISCR16 registers occurs [Clearing conditions] • When reading 1, then writing 0 • When interrupt exception handling is executed when low-level detection is set and $\overline{\text{IRQn}}$* or $\overline{\text{ExIRQn}}$ input is high • When IRQn interrupt exception handling is executed when falling-edge, rising-edge, or both-edge detection is set (n = 15 to 8) Note: * $\overline{\text{IRQn}}$ stands for $\overline{\text{IRQ15}}$, $\overline{\text{IRQ14}}$, $\overline{\text{IRQ11}}$ and $\overline{\text{IRQ10}}$.

- ISR

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

5.4 Interrupt Sources

5.4.1 External Interrupts

The external interrupts are NMI are IRQ15 to IRQ0. These interrupts can be used to restore this LSI from software standby mode.

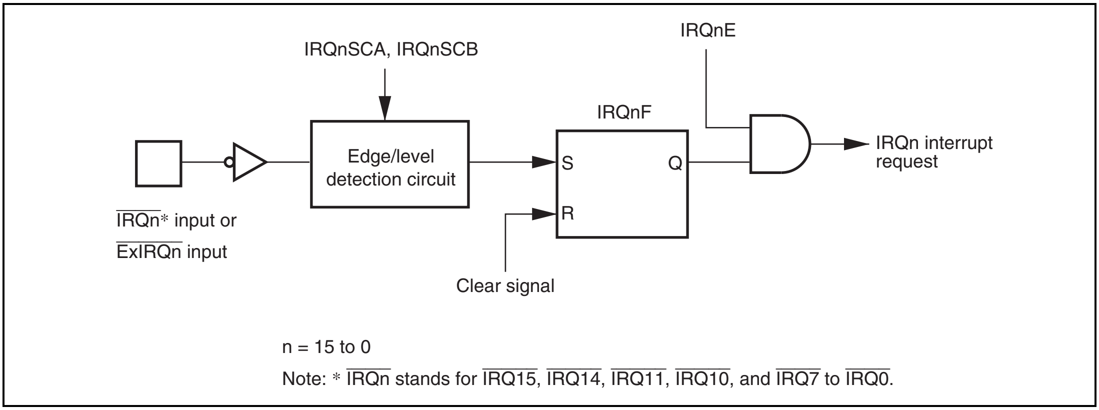
NMI Interrupt: NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode or the status of the CPU interrupt mask bits. The NMIEG bit in SYSCR can be used to select whether an interrupt is requested at a rising edge or a falling edge on the NMI pin.

IRQ15 to IRQ0 Interrupts: Interrupts IRQ15 to IRQ0 are requested by an input signal at pins $\overline{\text{IRQ15}}$, $\overline{\text{IRQ14}}$, $\overline{\text{IRQ11}}$, $\overline{\text{IRQ10}}$, $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ or pins $\overline{\text{ExIRQ15}}$ to $\overline{\text{ExIRQ0}}$. Interrupts IRQ15 to IRQ0 have the following features:

- The interrupt exception handling for interrupt requests IRQ15 to IRQ0 can be started at an independent vector address.
- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins $\overline{\text{IRQ15}}$, $\overline{\text{IRQ14}}$, $\overline{\text{IRQ11}}$, $\overline{\text{IRQ10}}$, $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ or pins $\overline{\text{ExIRQ15}}$ to $\overline{\text{ExIRQ0}}$.
- Enabling or disabling of interrupt requests IRQ15 to IRQ0 can be selected with IER.
- The status of interrupt requests IRQ15 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

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

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


IRQnSCA, IRQnSCB
IRQnE
IRQnF
Edge/level detection circuit
S
Q
R
IRQn interrupt request
$\overline{\text{IRQn}}$* input or
$\overline{\text{ExIRQn}}$ input
Clear signal

n = 15 to 0

Note: * $\overline{\text{IRQn}}$ stands for $\overline{\text{IRQ15}}$, $\overline{\text{IRQ14}}$, $\overline{\text{IRQ11}}$, $\overline{\text{IRQ10}}$, and $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.

Figure 5.2 Block Diagram of Interrupts IRQ15 to IRQ0

5.4.2 Internal Interrupts

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

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

5.5 Interrupt Exception Handling Vector Table

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

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

Table 5.3 Interrupt Sources, Vector Addresses, and Interrupt Priorities

Origin of Interrupt Source	Name	Vector Number	Vector Address	ICR	Priority
			Advanced Mode		
External pin	NMI	7	H'00001C	—	High
	IRQ0	16	H'000040	ICRA7	
	IRQ1	17	H'000044	ICRA6	
	IRQ2 IRQ3	18 19	H'000048 H'00004C	ICRA5	
	IRQ4 IRQ5	20 21	H'000050 H'000054	ICRA4	
	IRQ6 IRQ7	22 23	H'000058 H'00005C	ICRA3	
DTC	SWDTEND (Software activation data transfer end)	24	H'000060	ICRA2	
WDT_0	WOVI0 (Interval timer)	25	H'000064	ICRA1	
WDT_1	WOVI1 (Interval timer)	26	H'000068	ICRA0	
—	Address break	27	H'00006C	—	
A/D converter	ADI (A/D conversion end)	28	H'000070	ICRB7	
EVC	EVENTI	29	H'000074	—	
TMR_X	CMIAX (Compare match A) CMIBX (Compare match B) OVIX (Overflow)	44 45 46	H'0000B0 H'0000B4 H'0000B8	ICRB4	
FRT	OCIA (Output compare A) OCIB (Output compare B) FOVI (Overflow)	52 53 54	H'0000D0 H'0000D4 H'0000D8	ICRB6	Low

Origin of Interrupt Source	Name	Vector Number	Vector Address	ICR	Priority
			Advanced Mode		
External pin	IRQ8	56	H'0000E0	ICRD7	High
	IRQ9	57	H'0000E4		
	IRQ10	58	H'0000E8		
	IRQ11	59	H'0000EC		
	IRQ12	60	H'0000F0	ICRD6	
	IRQ13	61	H'0000F4		
	IRQ14	62	H'0000F8		
	IRQ15	63	H'0000FC		
TMR_0	CMIA0 (Compare match A)	64	H'000100	ICRB3	
	CMIB0 (Compare match B)	65	H'000104		
	OVI0 (Overflow)	66	H'000108		
TMR_1	CMIA1 (Compare match A)	68	H'000110	ICRB2	
	CMIB1 (Compare match B)	69	H'000114		
	OVI1 (Overflow)	70	H'000118		
TMR_Y	CMIAY (Compare match A)	72	H'000120	ICRB1	
	CMIBY (Compare match B)	73	H'000124		
	OVIY (Overflow)	74	H'000128		
IIC_2	IICI2	76	H'000130	ICRC2	
IIC_3	IICI3	78	H'000138		
SCI_3	ERI3 (Reception error 3)	80	H'000140	ICRC7	
	RXI3 (Reception completion 3)	81	H'000144		
	TXI3 (Transmission data empty 3)	82	H'000148		
	TEI3 (Transmission end 3)	83	H'00014C		
SCI_1	ERI1 (Reception error 1)	84	H'000150	ICRC6	
	RXI1 (Reception completion 1)	85	H'000154		
	TXI1 (Transmission data empty 1)	86	H'000158		
	TEI1 (Transmission end 1)	87	H'00015C		
IIC_0	IICI0	94	H'000178	ICRC4	
IIC_1	IICI1	98	H'000188	ICRC3	
LPC	ERR1(transfer error, etc.)	104	H'0001A0	ICRC1	
	IBFI1 (IDR1 reception completion)	105	H'0001A4		
	IBFI2 (IDR2 reception completion)	106	H'0001A8		
	IBFI3 (IDR3 reception completion)	107	H'0001AC		Low

Note: Vector numbers not listed above are reserved by the system.

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	SYSCR		Priority		
Control			Setting	Interrupt	
Mode	INTM1	INTM0	Registers	Mask Bits	Description
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 and UI bits. Priority levels can be set with ICR.

Figure 5.3 shows a block diagram of the priority decision circuit.

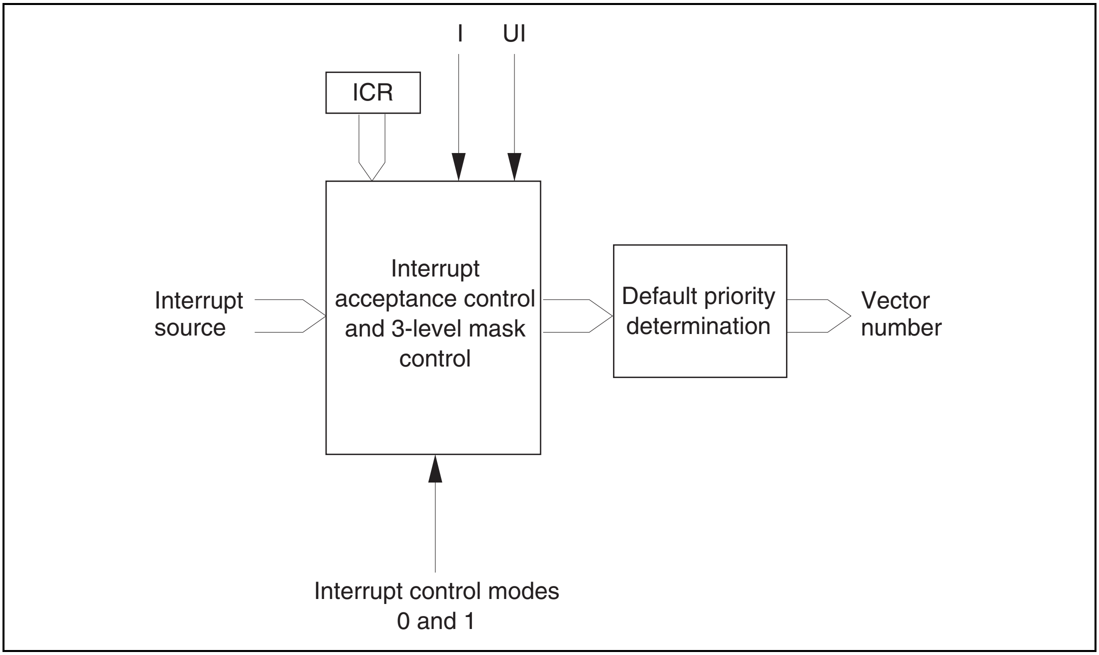

I
UI
ICR
Interrupt source
Interrupt acceptance control and 3-level mask control
Default priority determination
Vector number
Interrupt control modes 0 and 1

Figure 5.3 Block Diagram of Interrupt Control Operation

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

Table 5.5 shows the interrupts selected in each interrupt control mode.

Table 5.5 Interrupts Selected in Each Interrupt Control Mode

	Interrupt Mask Bits		
Interrupt Control Mode	**I**	**UI**	**Selected Interrupts**
0	0	x	All interrupts (interrupt control level 1 has priority)
	1	x	NMI and address break interrupts
1	0	x	All interrupts (interrupt control level 1 has priority)
	1	0	NMI, address break, and interrupt control level 1 interrupts
		1	NMI and address break interrupts

[Legend]

x: Don't care

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

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

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

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

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

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

[Legend]

O: Interrupt operation control performed

IM: Used as an interrupt mask bit

PR: Sets priority

—: Not used

5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupts other than NMI 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 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 interrupt requests 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. The EVENTI interrupt is enabled or disabled by the I bit.
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.

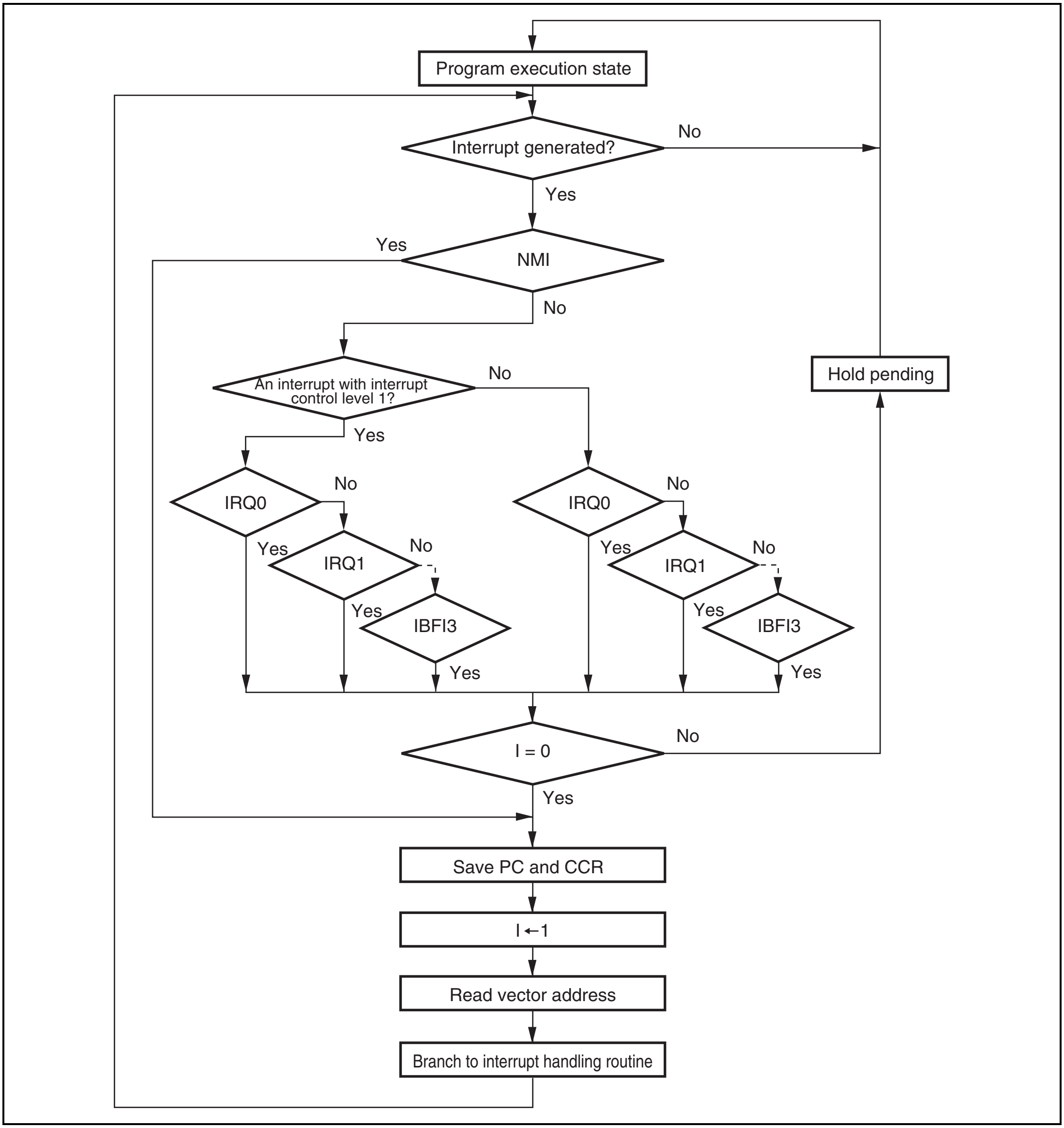


Figure 5.4 Flowchart of Procedure up to Interrupt Acceptance in Interrupt Control Mode 0

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.

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

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

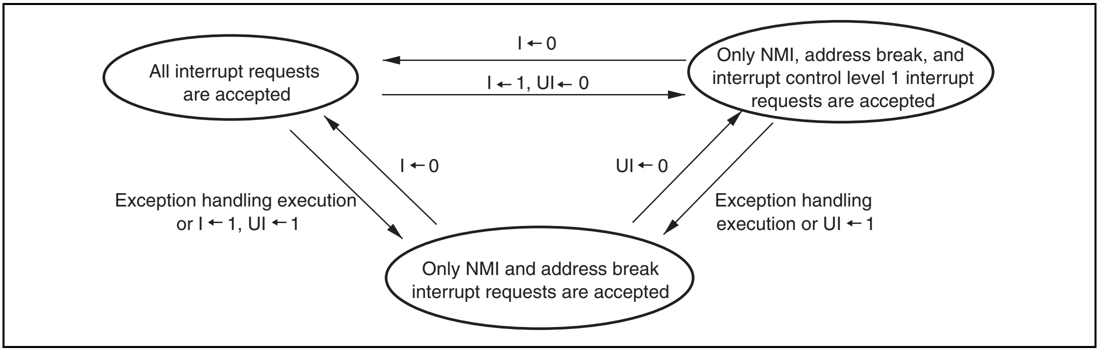

All interrupt requests are accepted
I ← 0
I ← 1, UI ← 0
Only NMI, address break, and interrupt control level 1 interrupt requests are accepted
I ← 0
UI ← 0
Exception handling execution or I ← 1, UI ← 1
Exception handling execution or UI ← 1
Only NMI and address break interrupt requests are accepted

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 both the I and UI bits are set to 1, only NMI and address break interrupt requests are 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 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.

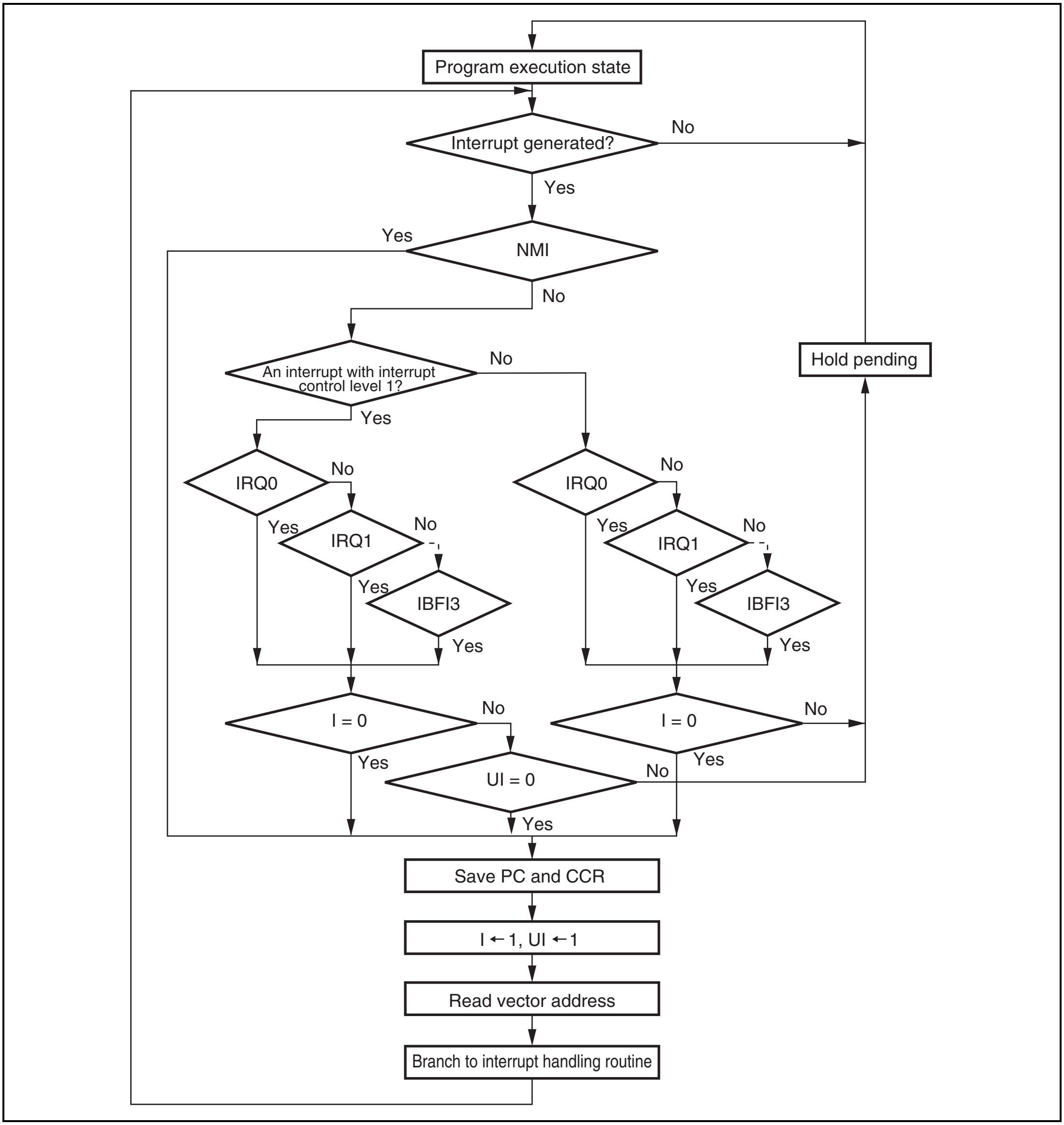
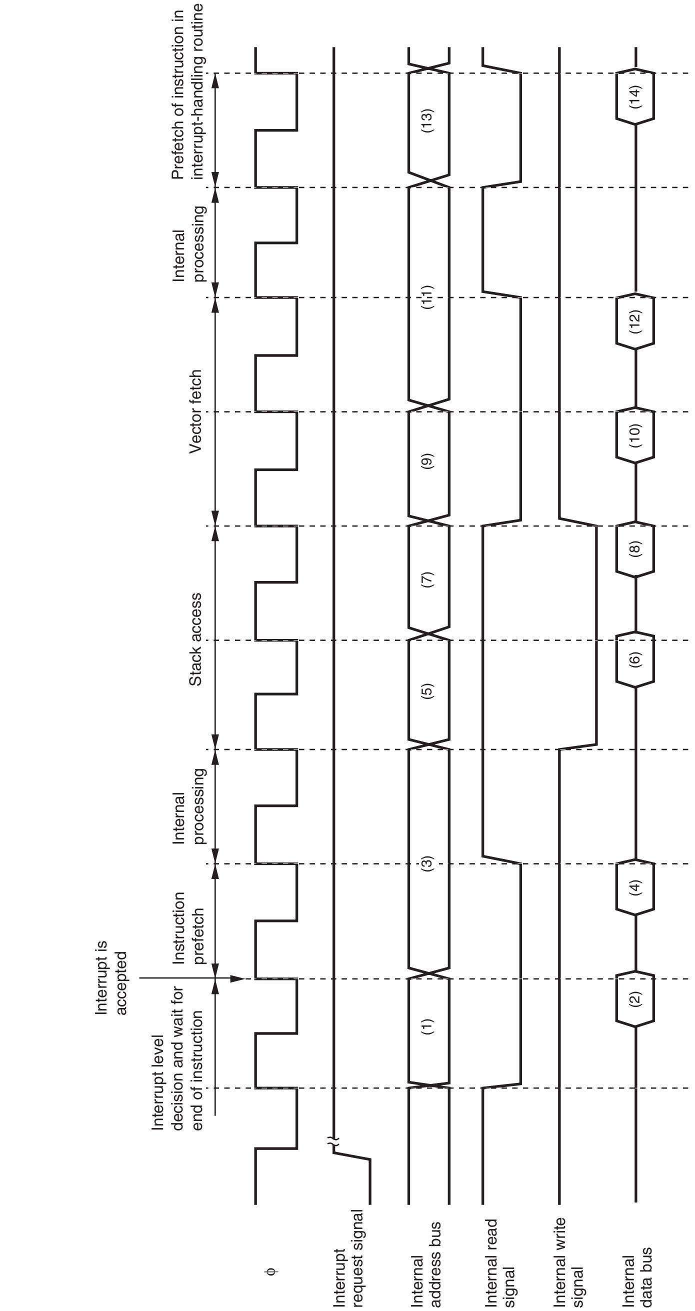


Figure 5.6 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1

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

Table 5.7 Interrupt Response Times

No.	Execution Status	Advanced Mode
1	Interrupt priority determination*[1]	3
2	Number of wait states until executing instruction ends*[2]	1 to (19 + 2·S_I)
3	PC, CCR stack save	2·S_K
4	Vector fetch	2·S_I
5	Instruction fetch*[3]	2·S_I
6	Internal processing*[4]	2
	Total (using on-chip memory)	12 to 32

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

Table 5.8 Number of States in Interrupt Handling Routine Execution Status

Symbol	Internal Memory
Instruction fetch S_I	1
Branch address read S_J	
Stack manipulation S_K	

5.6.5 DTC Activation by Interrupt

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

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

For details of interrupt requests that can be used to activate the DTC, see section 7, Data Transfer Controller (DTC). Figure 5.8 shows a block diagram of the DTC and interrupt controller.

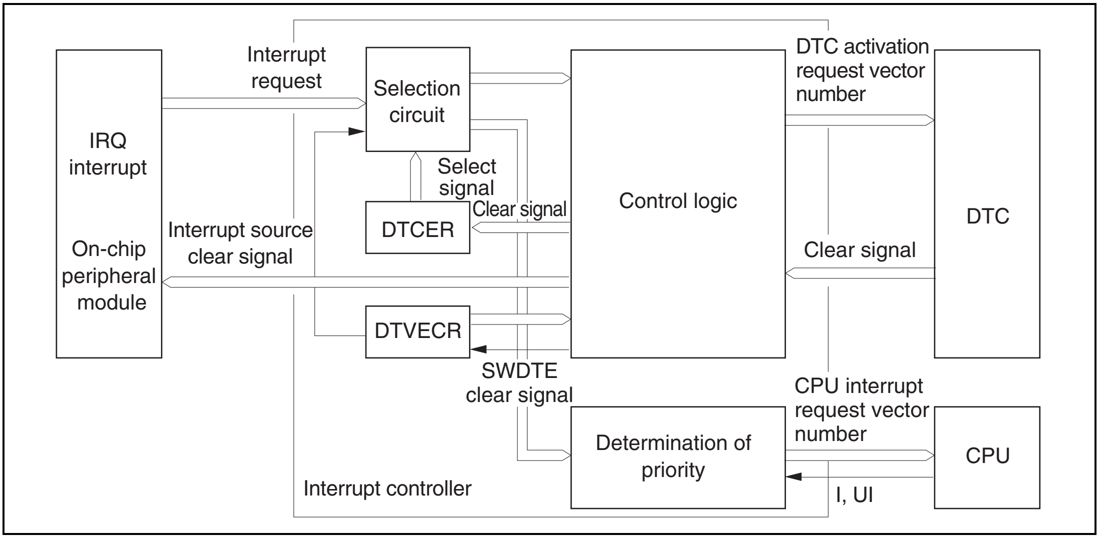


Figure 5.8 Interrupt Control for DTC

The interrupt controller has three main functions in DTC control.

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

Determination of Priority: The DTC activation source is selected in accordance with the default priority order, and is not affected by mask or priority levels. See section 7.5, Location of Register Information and DTC Vector Table, for the respective priorities.

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

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

Table 5.9 Interrupt Source Selection and Clearing Control

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

[Legend]

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

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

×: The relevant interrupt cannot be used.

*: Don't care

5.7 Usage Notes

5.7.1 Conflict between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupt requests, the disabling becomes effective after execution of the instruction. When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, and if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored. The same rule is also applied when an interrupt source flag is cleared to 0. Figure 5.9 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.

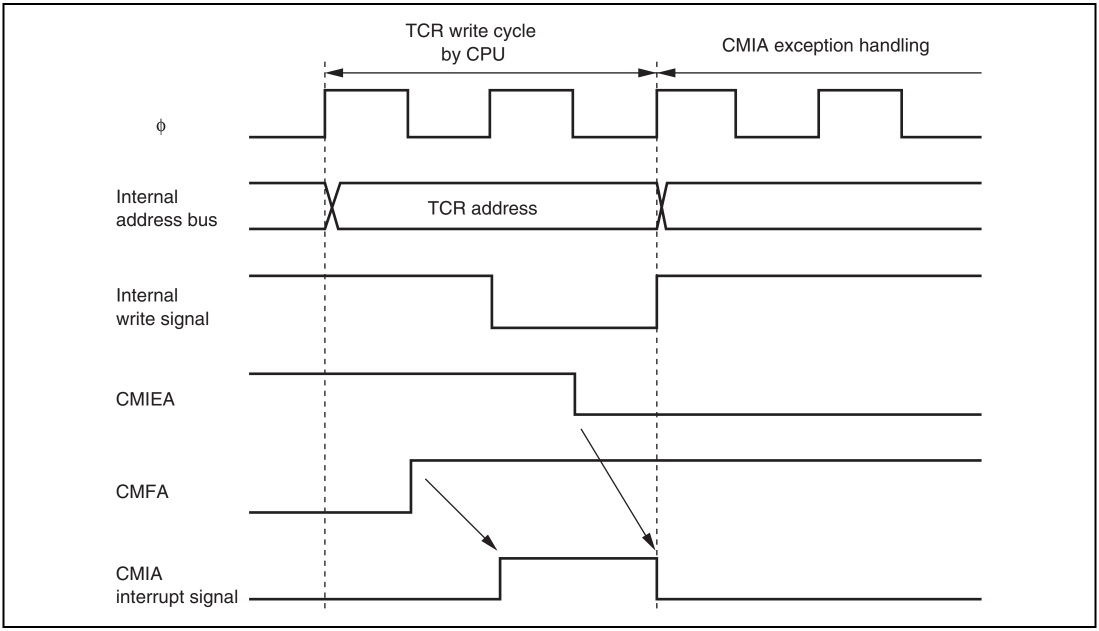


Figure 5.9 Conflict between Interrupt Generation and Disabling

5.7.2 Instructions that Disable Interrupts

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

5.7.3 Interrupts during Execution of EEPMOV Instruction

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

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

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

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

5.7.4 IRQ Status Registers (ISR16, ISR)

Since IRQnF may be set to 1 according to the pin status after a reset, the ISR16 and the ISR should be read after a reset, and then write 0 in IRQnF (n = 15 to 0).

Section 6 Bus Controller (BSC)

This LSI has an on-chip bus controller (BSC). The BSC has a bus arbitration function, and controls the operation of the internal bus masters – CPU and data transfer controller (DTC).

6.1 Features

- Bus arbitration function

  Includes a bus arbiter that arbitrates bus mastership between the CPU and DTC.

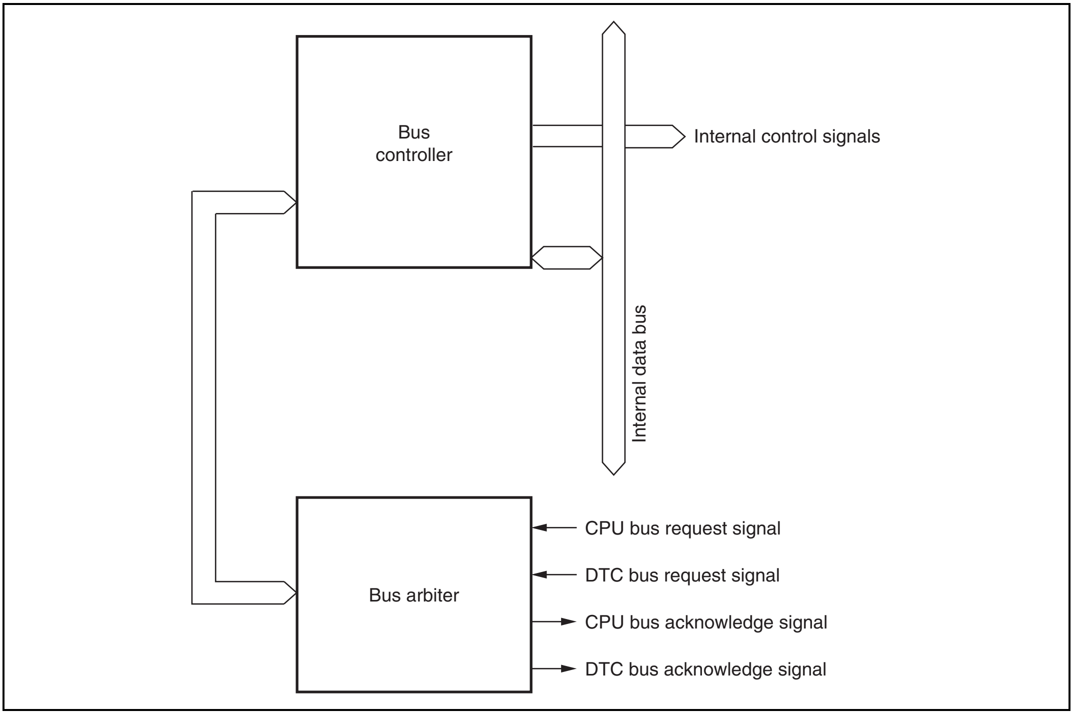


Figure 6.1 Block Diagram of Bus Controller

6.2 Bus Arbitration

6.2.1 Overview

The BSC has a bus arbiter that arbitrates bus master operations. There are two bus masters – the CPU and DTC – that perform read/write operations while they have bus mastership.

6.2.2 Priority of Bus Mastership

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

(High) DTC > CPU (Low)

6.2.3 Bus Mastership Transfer Timing

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

(1) CPU

The CPU is the lowest-priority bus master, and if a bus mastership request is received from the DTC, the bus arbiter transfers the bus mastership to the DTC. The timing for transferring the bus mastership is as follows:

1. Bus mastership is transferred at a break between bus cycles. However, if bus cycle is executed in discrete operations, as in the case of a long-word size access, the bus is not transferred at a break between the operations. For details see section 2.7, Bus States During Instruction Execution in the H8S/2600 Series, H8S/2000 Series Software Manual.
2. If the CPU is in sleep mode, it transfers the bus mastership immediately.

(2) DTC

The DTC sends the bus arbiter a request for the bus mastership when a request for DTC activation occurs. The DTC releases the bus mastership after a series of processes has completed.

Section 7 Data Transfer Controller (DTC)

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

Figure 7.1 shows a block diagram of the DTC. The DTC's register information is stored in the on-chip RAM. When the DTC is used, the RAME bit in SYSCR must be set to 1. A 32-bit bus connects the DTC to addresses H'FFEC00 to H'FFEFFF in on-chip RAM (1 kbyte), enabling 32-bit/1-state reading and writing of the DTC register information.

7.1 Features

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

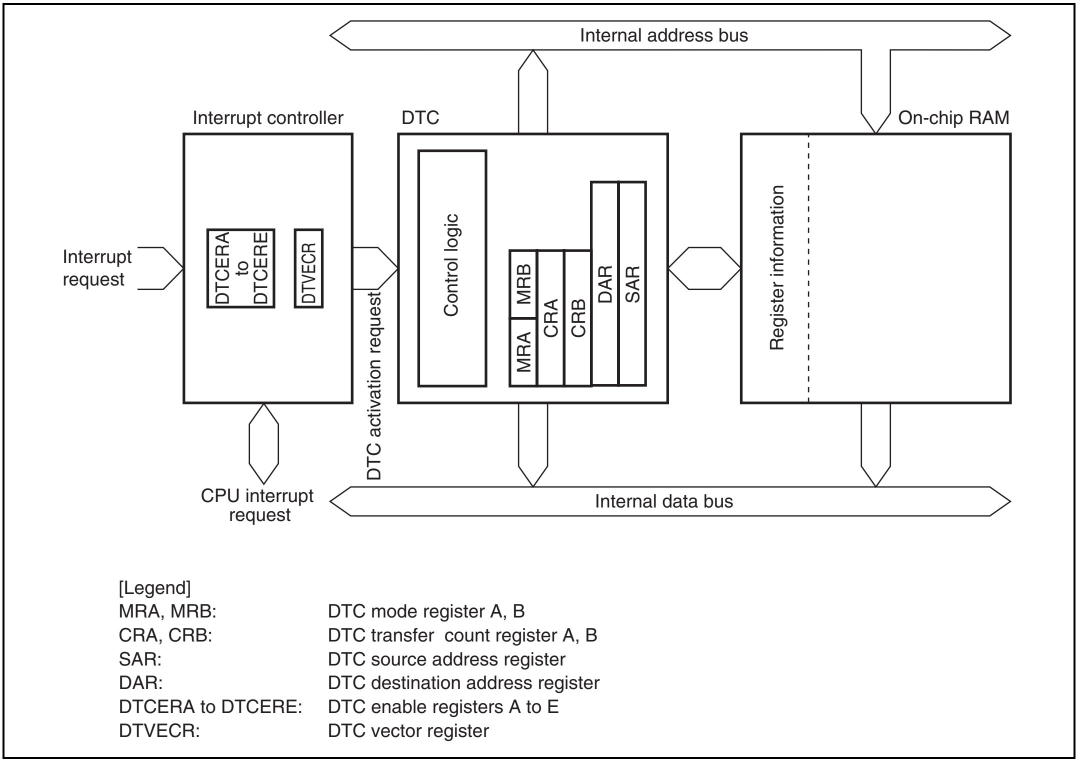

Internal address bus
Interrupt controller
DTC
On-chip RAM
Interrupt request
DTCERA to DTCERE
DTVECR
Control logic
MRB
MRA
CRA
CRB
DAR
SAR
Register information
DTC activation request
CPU interrupt request
Internal data bus
[Legend]
MRA, MRB: DTC mode register A, B
CRA, CRB: DTC transfer count register A, B
SAR: DTC source address register
DAR: DTC destination address register
DTCERA to DTCERE: DTC enable registers A to E
DTVECR: DTC vector register

Figure 7.1 Block Diagram of DTC

7.2 Register Descriptions

The DTC has the following registers.

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

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

- DTC enable registers (DTCER)
- DTC vector register (DTVECR)
- Keyboard comparator control register (KBCOMP)
- Event counter control register (ECCR)
- Event counter status register (ECS)

7.2.1 DTC Mode Register A (MRA)

MRA selects the DTC operating mode.

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

Note: x Don't care

7.2.2 DTC Mode Register B (MRB)

MRB selects the DTC operating mode.

Bit	Bit Name	Initial Value	R/W	Description
7	CHNE	Undefined	—	DTC Chain Transfer Enable When this bit is set to 1, a chain transfer will be performed. For details, see section 7.6.4, Chain Transfer. In data transfer with CHNE set to 1, determination of the end of the specified number of data transfers, clearing of the interrupt source flag, and clearing of DTCER are not performed.
6	DISEL	Undefined	—	DTC Interrupt Select When this bit is set to 1, a CPU interrupt request is generated every time data transfer ends. When this bit is cleared to 0, a CPU interrupt request is generated only when the specified number of data transfer ends.
5 to 0	—	Undefined	—	Reserved These bits have no effect on DTC operation. The write value should always be 0.

7.2.3 DTC Source Address Register (SAR)

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

7.2.4 DTC Destination Address Register (DAR)

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

7.2.5 DTC Transfer Count Register A (CRA)

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

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

The number of times data is transferred is one when the setting value of CRA is H'0001, 65,535 when the setting value is H'FFFF, and 65,536 when the setting value is H'0000.

In repeat transfer mode CRA is divided in two, with the highest eight bits designated as CRAH and the lowest eight bits as CRAL. CRAH holds the value for the number of data transfers, and CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are transferred when the counter value reaches H'00. The number of times data is transferred is one when CRAH = CRAL = H'01, 255 when CRAH = CRAL = H'FF, and 256 when CRAH = CRAL = H'00.

In block transfer mode CRA is divided in two, with the highest eight bits designated as CRAH and the lowest eight bits as CRAL. CRAH holds the value for the block size, and CRAL functions as an 8-bit block size counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are transferred when the counter value reaches H'00. The block size is one byte (or one word) when CRAH = CRAL = H'01, 255 bytes (or 255 words) when CRAH = CRAL = H'FF, and 256 bytes (or 256 words) when CRAH = CRAL = H'00.

7.2.6 DTC Transfer Count Register B (CRB)

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

7.2.7 DTC Enable Registers (DTCER)

DTCER specifies DTC activation interrupt sources. DTCER is comprised of five registers: DTCERA to DTCERE. The correspondence between interrupt sources and DTCE bits is shown in tables 7.1 and 7.4. For DTCE bit setting, use bit manipulation instructions such as BSET and BCLR. Multiple DTC activation sources can be set at one time (only at the initial setting) by masking all interrupts and writing data after executing a dummy read on the relevant register.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	DTCE7 to DTCE0	All 0	R/W	DTC Activation Enable Setting this bit to 1 specifies a relevant interrupt source as a DTC activation source. [Clearing conditions] • When data transfer has ended with the DISEL bit in MRB set to 1 • When the specified number of transfers have ended These bits are not cleared when the DISEL bit is 0 and the specified number of transfers have not been completed

Table 7.1 Correspondence between Interrupt Sources and DTCER

		Register				
Bit	**Bit Name**	**DTCERA**	**DTCERB**	**DTCERC**	**DTCERD**	**DTCERE**
7	DTCEn7	(16)IRQ0	—	—	(86)TXI1	—
6	DTCEn6	(17)IRQ1	(76)IICI2	—	—	—
5	DTCEn5	(18)IRQ2	(94)IICI0	—	—	—
4	DTCEn4	(19)IRQ3	—	(29)EVENTI	(78)IICI3	—
3	DTCEn3	(28)ADI	—	—	(98)IICI1	(104)ERR1
2	DTCEn2	—	—	(81)RXI3	—	(105)IBFI1
1	DTCEn1	—	—	(82)TXI3	—	(106)IBFI2
0	DTCEn0	—	—	(85)RXI1	—	(107)IBFI3

[Legend]

n:	A to E
( ):	Vector number
—:	Reserved. The write value should always be 0.

7.2.8 DTC Vector Register (DTVECR)

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

Bit	Bit Name	Initial Value	R/W	Description
7	SWDTE	0	R/W	DTC Software Activation Enable Setting this bit to 1 activates DTC. Only 1 can be written to this bit. [Clearing conditions] • When the DISEL bit is 0 and the specified number of transfers have not ended • When 0 is written to the DISEL bit after a software-activated data transfer end interrupt (SWDTEND) request has been sent to the CPU. This bit will not be cleared when the DISEL bit is 1 and data transfer has ended or when the specified number of transfers has ended.
6 to 0	DTVEC6 to DTVEC0	All 0	R/W	DTC Software Activation Vectors 6 to 0 These bits specify a vector number for DTC software activation. The vector address is expressed as H'0400 + (vector number × 2). For example, when DTVEC6 to DTVEC0 = H'10, the vector address is H'0420. When the SWDTE bit is 0, these bits can be written to.

7.2.9 Keyboard Comparator Control Register (KBCOMP)

KBCOMP enables or disables the comparator scan function of event counter.

Bit	Bit Name	Initial Value	R/W	Description
7	EVENTE	0	R/W	Event Count Enable 0: Disables event count function 1: Enables event count function
6, 5	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
4 to 0	—	All 0	R/W	Reserved The initial value should not be changed.

7.2.10 Event Counter Control Register (ECCR)

ECCR selects the event counter channels for use and the detection edge.

Bit	Bit Name	Initial Value	R/W	Description
7	EDSB	0	R/W	Event Counter Edge Select Selects the detection edge for the event counter. 0: Counts the rising edges 1: Counts the falling edges
6 to 4	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
3	—	0	R/W	Reserved The initial value should not be changed.

Bit	Bit Name	Initial Value	R/W	Description
2 to 0	ECSB2 to ECSB0	All 0	R/W	Event Counter Channel Select 2 to 0 These bits select pins for event counter input. A series of pins are selected starting from EVENT0. When PAnDDR is set to 1, inputting events to EVENT0 to EVENT7 is ignored. 000: EVENT0 is used 001: EVENT0 to EVENT1 are used 010: EVENT0 to EVENT2 are used 011: EVENT0 to EVENT3 are used 100: EVENT0 to EVENT4 are used 101: EVENT0 to EVENT5 are used 110: EVENT0 to EVENT6 are used 111: EVENT0 to EVENT7 are used

7.2.11 Event Counter Status Register (ECS)

ECS is a 16-bit register that holds events temporarily. The DTC decides the counter to be incremented according to the state of this register. Reading this register allows the monitoring of events that are not yet counted by the event counter. Access in 8-bit unit is not allowed.

Bit	Bit Name	Initial Value	R/W	Description
15 to 8	—	All 0	R	Reserved
7 to 0	E7 to E0	0	R	Event Monitor 7 to 0 These bits indicate processed/unprocessed states of the events that are input to EVENT7 to EVENT0. 0: The corresponding event is already processed 1: The corresponding event is not yet processed

7.3 DTC Event Counter

To count events of EVENT 0 to EVENT7 by the DTC event counter function, set DTC as below.

Table 7.2 DTC Event Counter Conditions

Register	Bit	Bit Name	Description
MRA	7, 6	SM1, SM0	00: SAR is fixed.
	5, 4	DM1, DM0	00: DAR is fixed.
	3, 2	MD1, MD0	01: Repeat transfer mode
	1	DTS	0: Destination is repeat area
	0	Sz	1: Word size transfer
MRB	7	CHNE	0: Chain transfer is disabled
	6	DISEL	0: Interrupt request is generated when data is transferred by the number of specified times
	5 to 0	—	B'000000
SAR	23 to 0	—	Identical optional RAM address. Its lower five bits are B'00000.
DAR	23 to 0	—	The start address of 16 words is this address. They are incremented every time an event is detected in EVENT0 to EVENT15.
CRAH	7 to 0	—	H'FF
CRAL	7 to 0	—	H'FF
CRBH	7 to 0	—	H'FF
CRBL	7 to 0	—	H'FF
DTCERC	4	DTCEC4	1: DTC function of the event counter is enabled
KBCOMP	7	EVENTE	1: Event counter enable
RAM	—	—	(SAR, DAR) : Result of EVENT0 count (SAR, DAR) + 2: Result of EVENT 1 count (SAR, DAR) + 4: Result of EVENT 2 count ↓ (SAR, DAR) + 14: Result of EVENT 7 count

The corresponding flag to ECS input pin is set to 1 when the event pins that are specified by the ECSB2 to ECSB0 in ECCR detect the edge events specified by the EDSB in ECCR. For this flag state, status/address codes are generated.

An EVENTI interrupt request is generated even if only one bit in ECS is set to 1.

The EVENTI interrupt request activates the DTC and transfers data from RAM to RAM in the same address. Data is incremented in the DTC. The lower five bits of SAR and DAR are replaced with address code that is generated by the ECS flag status.

When the DTC transfer is completed, the ECS flag for transfer is cleared.

Table 7.3 Flag Status/Address Code

ECS								Address Code
7	6	5	4	3	2	1	0	
							1	B'00000
						1	0	B'00010
					1	0	0	B'00100
				1	0	0	0	B'00110
			1	0	0	0	0	B'01000
		1	0	0	0	0	0	B'01010
	1	0	0	0	0	0	0	B'01100
1	0	0	0	0	0	0	0	B'01110

7.3.1 Event Counter Handling Priority

Counting of EVENT0 to EVENT7 is processed in the priority shown as below.

High   Low

EVENT0 > EVENT1 · · · · · · · · · · EVENT6 > EVENT7

7.3.2 Usage Notes

There are following usage notes for this event counter because it uses the DTC.

1. Continuous events that are input from the same pin and out of DTC handling are ignored because the count up is operated by means of the DTC.
2. If some events are generated in short intervals, the priority of event counter handling is not ordered and events are not handled in order of arrival.
3. If the counter overflows, this event counter counts from H'0000 without generating an interrupt.

7.4 Activation Sources

The DTC is activated by an interrupt request or by a write to DTVECR by software. The interrupt request source to activate the DTC is selected by DTCER. At the end of a data transfer (or the last consecutive transfer in the case of chain transfer), the interrupt flag that became the activation source or the corresponding DTCER bit is cleared. The activation source flag, in the case of RXI0, for example, is the RDRF flag in SCI_0.

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

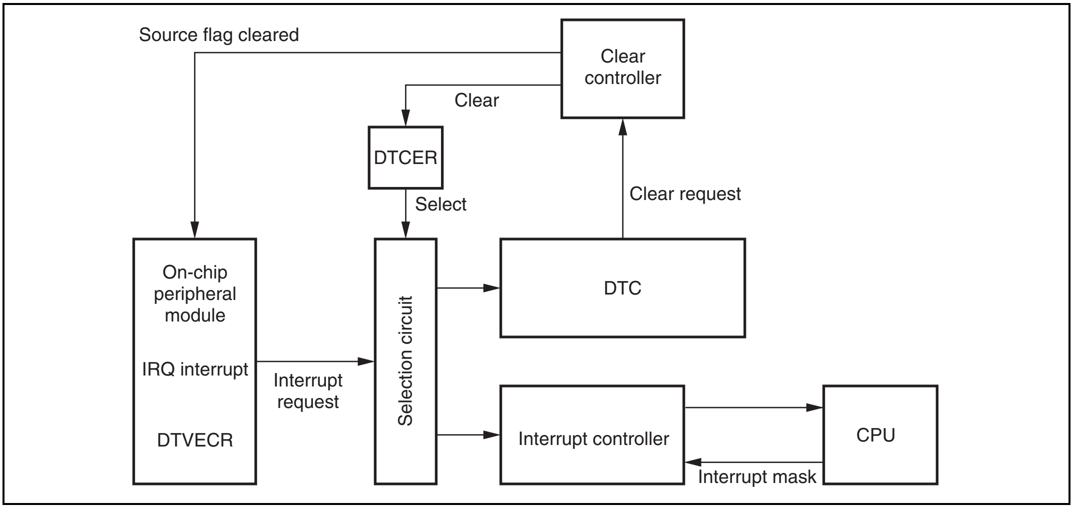


Figure 7.2 Block Diagram of DTC Activation Source Control

7.5 Location of Register Information and DTC Vector Table

Locate the register information in the on-chip RAM (addresses: H'FFEC00 to H'FFEFFF). Register information should be located at an address that is a multiple of four within the range. The method for locating the register information in address space is shown in figure 7.3. Locate MRA, SAR, MRB, DAR, CRA, and CRB, in that order, from the start address of the register information. In the case of chain transfer, register information should be located in consecutive areas as shown in figure 7.3, and the register information start address should be located at the vector address corresponding to the interrupt source in the DTC vector table. The DTC reads the start address of the register information from the vector table set for each activation source, and then reads the register information from that start address.

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

The configuration of the vector address is a 2-byte unit. Specify the lower two bytes of the register information start address.

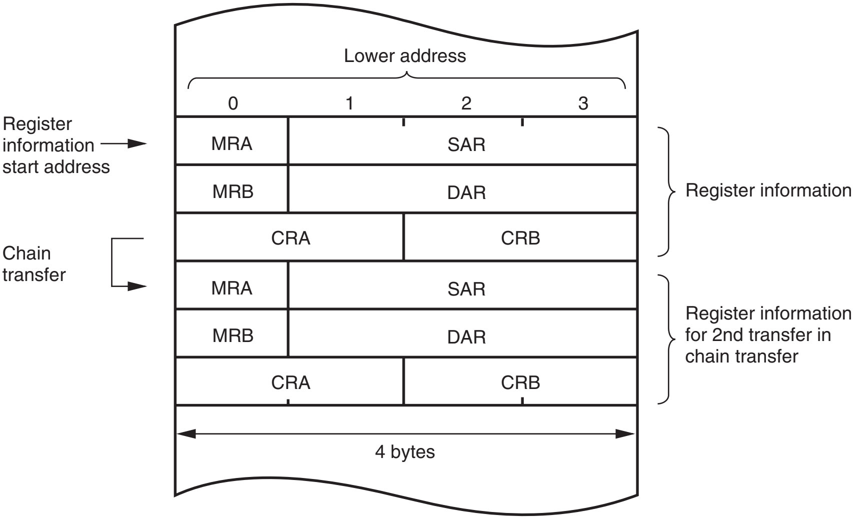


Figure 7.3 DTC Register Information Location in Address Space

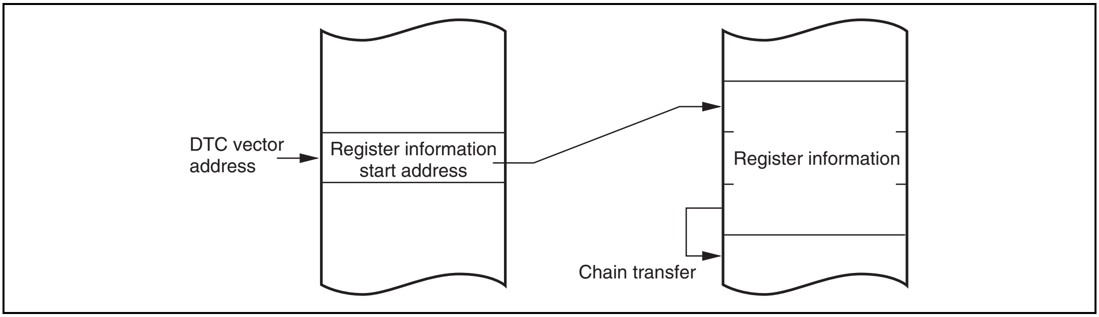


Figure 7.4 Correspondence between DTC Vector Address and Register Information

Table 7.4 Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs

Activation Source Origin	Activation Source	Vector Number	DTC Vector Address	DTCE*	Priority
Software	Write to DTVECR	DTVECR	H'0400 + (vector number x 2)	—	High
External pins	IRQ0	16	H'0420	DTCEA7	
	IRQ1	17	H'0422	DTCEA6	
	IRQ2	18	H'0424	DTCEA5	
	IRQ3	19	H'0426	DTCEA4	
A/D converter	ADI	28	H'0438	DTCEA3	
EVC	EVENTI	29	H'043A	DTCEC4	
IIC_2	IICI2	76	H'0498	DTCEB6	
IIC_3	IICI3	78	H'049C	DTCED4	
SCI_3	RXI3	81	H'04A2	DTCEC2	
	TXI3	82	H'04A4	DTCEC1	
SCI_1	RXI1	85	H'04AA	DTCEC0	
	TXI1	86	H'04AC	DTCED7	
IIC_0	IICI0	94	H'04BC	DTCEB5	
IIC_1	IICI1	98	H'04C4	DTCED3	
LPC	ERRI	104	H'04D0	DTCEE3	
	IBFI1	105	H'04D2	DTCEE2	
	IBFI2	106	H'04D4	DTCEE1	
	IBFI3	107	H'04D6	DTCEE0	Low

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

7.6 Operation

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

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

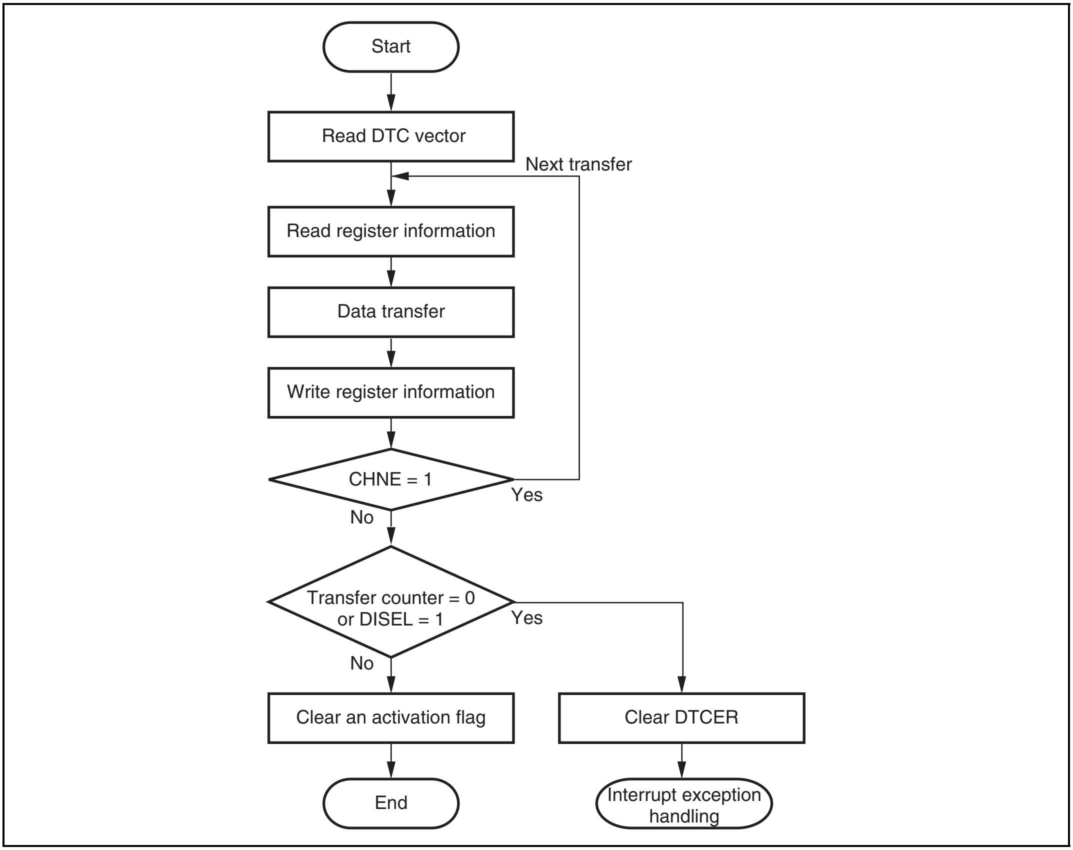


Figure 7.5 DTC Operation Flowchart

7.6.1 Normal Transfer Mode

In normal transfer mode, one activation source transfers one byte or one word of data. Table 7.5 lists the register functions in normal transfer mode. From 1 to 65,536 transfers can be specified. Once the specified number of transfers has been completed, a CPU interrupt can be requested.

Table 7.5 Register Functions in Normal Transfer Mode

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

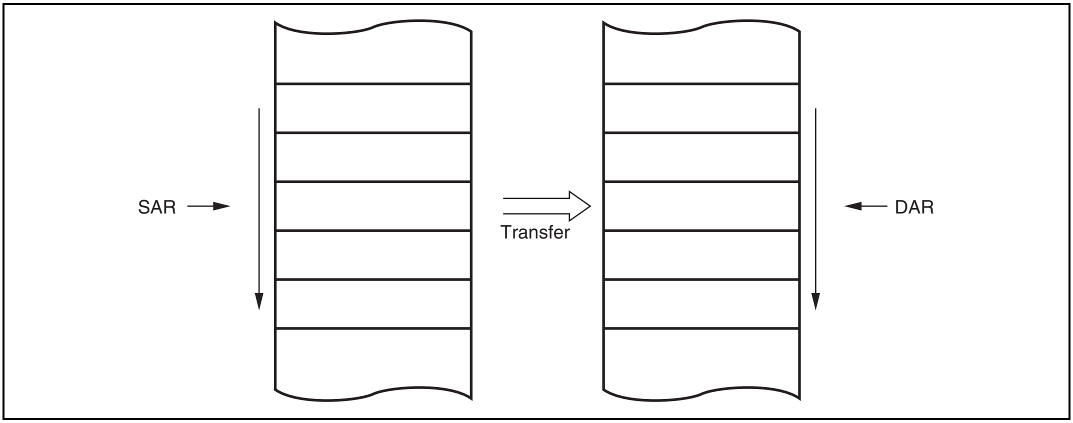


Figure 7.6 Memory Mapping in Normal Transfer Mode

7.6.2 Repeat Transfer Mode

In repeat transfer mode, one activation source transfers one byte or one word of data. Table 7.6 lists the register functions in repeat transfer mode. From 1 to 256 transfers can be specified. Once the specified number of transfers has been completed, the initial states of the transfer counter and the address register that is specified as the repeat area is restored, and transfer is repeated. In repeat transfer mode, the transfer counter value does not reach H'00, and therefore CPU interrupts cannot be requested when the DISEL bit in MRB is cleared to 0.

Table 7.6 Register Functions in Repeat Transfer Mode

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

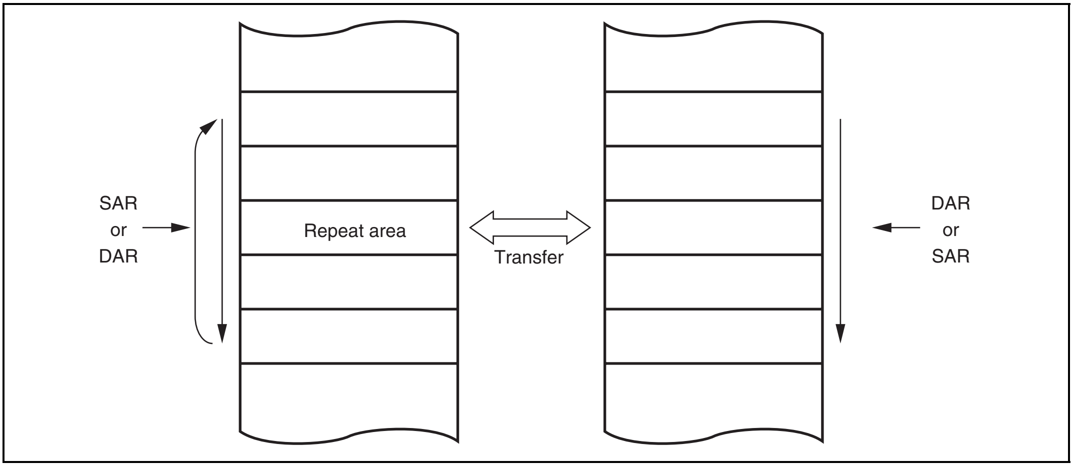


Figure 7.7 Memory Mapping in Repeat Transfer Mode

7.6.3 Block Transfer Mode

In block transfer mode, one activation source transfers one block of data. Either the transfer source or the transfer destination is designated as a block area. Table 7.7 lists the register functions in block transfer mode. The block size can be between 1 and 256. When the transfer of one block ends, the initial state of the block size counter and the address register that is specified as the block area is restored. The other address register is then incremented, decremented, or left fixed according to the register information. From 1 to 65,536 transfers can be specified. Once the specified number of transfers has been completed, a CPU interrupt is requested.

Table 7.7 Register Functions in Block Transfer Mode

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

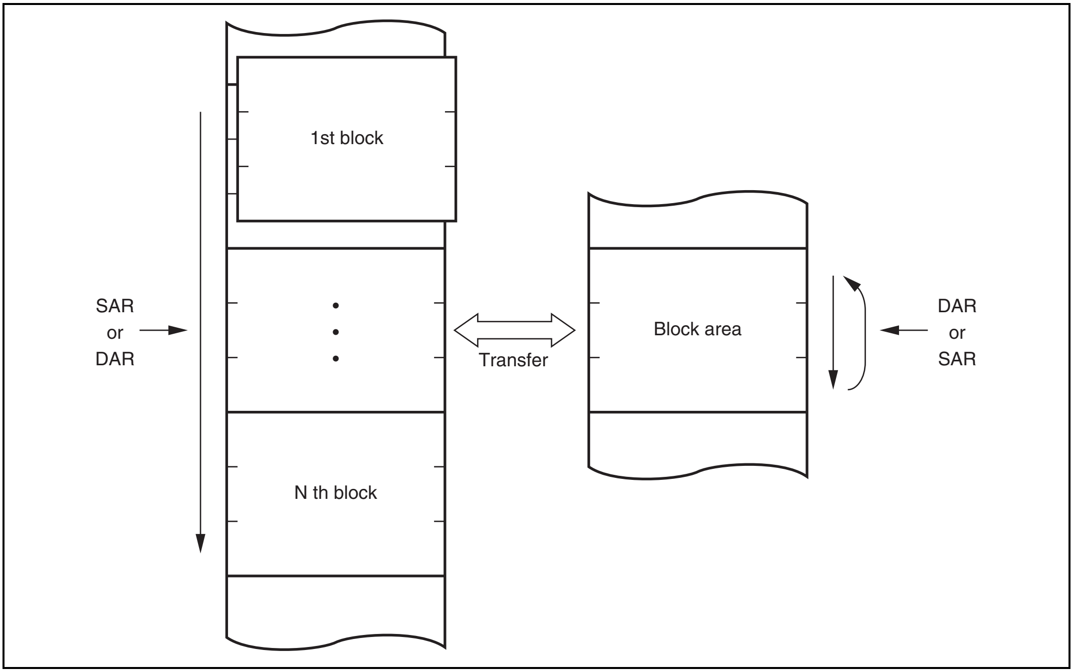


Figure 7.8 Memory Mapping in Block Transfer Mode

7.6.4 Chain Transfer

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

Figure 7.9 shows the overview of chain transfer operation. When activated, the DTC reads the register information start address stored at the DTC vector address, and then reads the first register information at that start address. After the data transfer, the CHNE bit will be tested. When it has been set to 1, DTC reads the next register information located in a consecutive area and performs the data transfer. These sequences are repeated until the CHNE bit is cleared to 0.

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

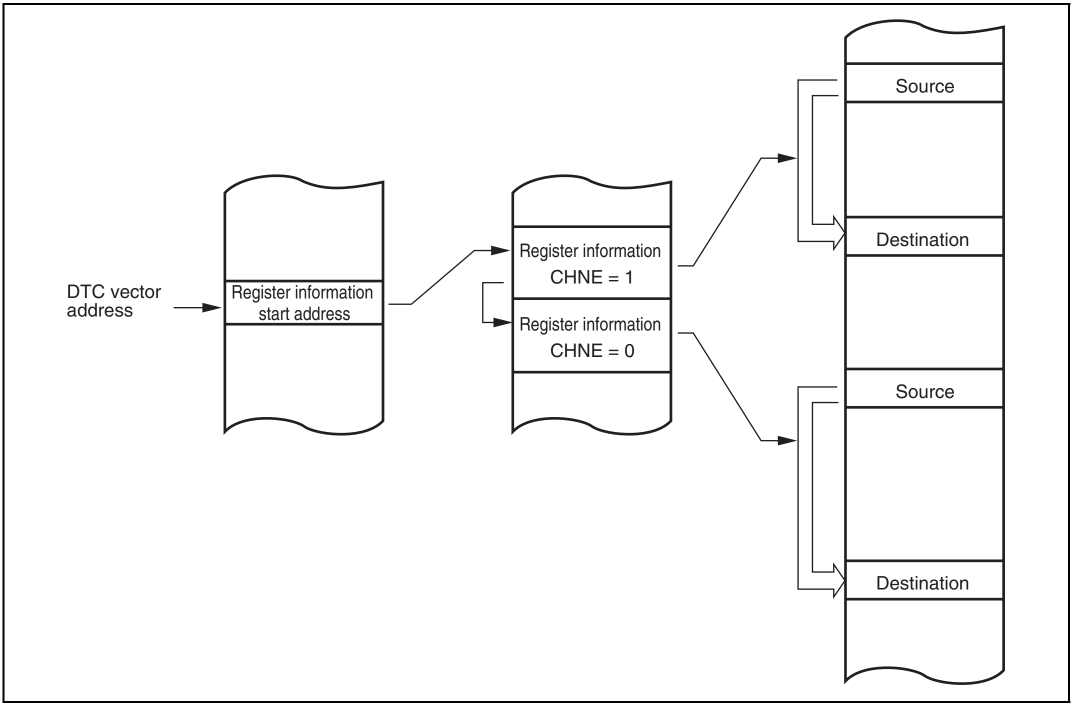


Figure 7.9 Chain Transfer Operation

7.6.5 Interrupt Sources

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

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

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

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

7.6.6 Operation Timing

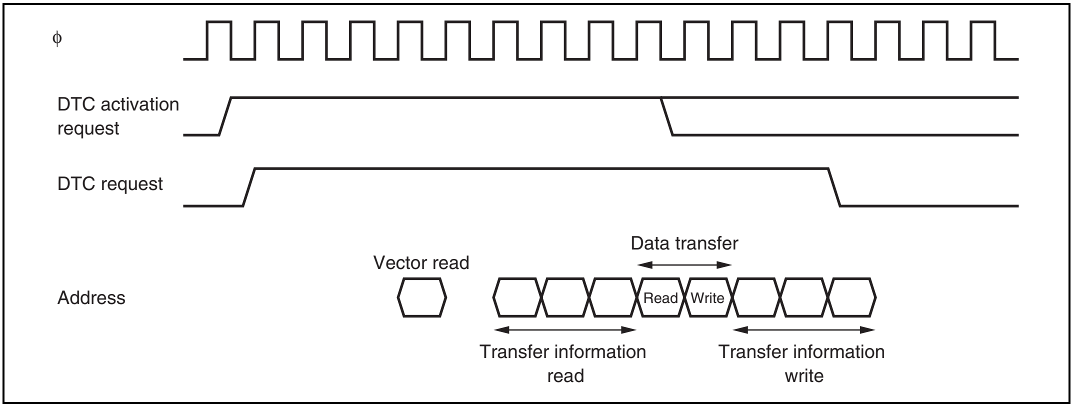


Figure 7.10 DTC Operation Timing (Example in Normal Transfer Mode or Repeat Transfer Mode)

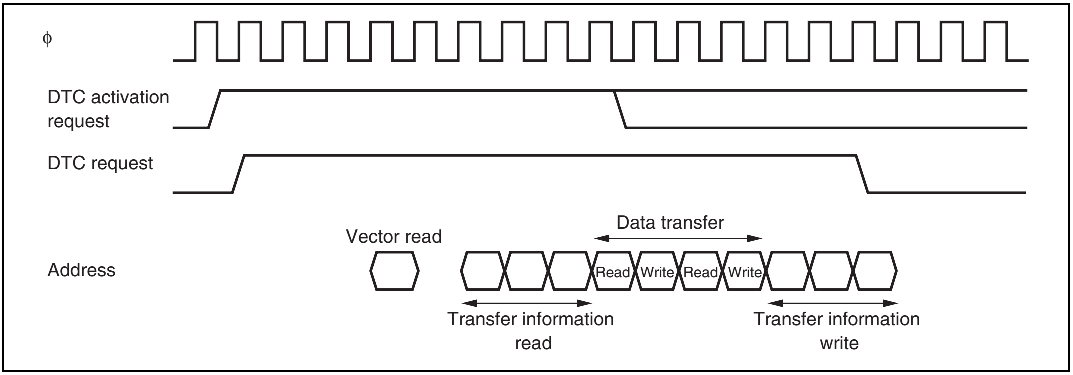


Figure 7.11 DTC Operation Timing (Example of Block Transfer Mode, with Block Size of 2)

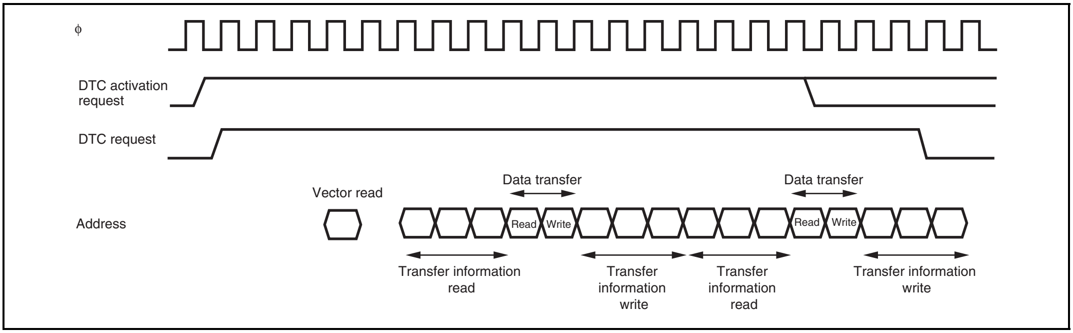

φ
DTC activation request
DTC request
Address
Vector read
Data transfer
Data transfer
Read
Write
Read
Write
Transfer information read
Transfer information write
Transfer information read
Transfer information write

Figure 7.12 DTC Operation Timing (Example of Chain Transfer)

7.6.7 Number of DTC Execution States

Table 7.8 lists the execution status for a single DTC data transfer, and table 7.9 shows the number of states required for each execution status.

Table 7.8 DTC Execution Status

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

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

Table 7.9 Number of States Required for Each Execution Status

Object to be Accessed		On-Chip RAM (H'FFEC00 to H'FFEFFF)	On-Chip RAM (On-chip RAM area other than H'FFEC00 to H'FFEFFF)	On-Chip ROM	On-Chip I/O Registers	
Bus width		32	16	16	8	16
Access states		1	1	1	2	2
Execution status	Vector read S_I	—	—	1	—	—
	Register information read/write S_J	1	—	—	—	—
	Byte data read S_K	1	1	1	2	2
	Word data read S_K	1	1	1	4	2
	Byte data write	1	1	1	2	2
	Word data write S_L	1	1	1	4	2
	Internal operation S_M	1	1	1	1	1

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

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

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

7.7 Procedures for Using DTC

7.7.1 Activation by Interrupt

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

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

7.7.2 Activation by Software

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

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

7.8 Examples of Use of the DTC

7.8.1 Normal Transfer Mode

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

1. Set MRA to a fixed source address (SM1 = SM0 = 0), incrementing destination address (DM1 = 1, DM0 = 0), normal transfer mode (MD1 = MD0 = 0), and byte size (Sz = 0). The DTS bit can have any value. Set MRB for one data transfer by one interrupt (CHNE = 0, DISEL = 0). Set the SCI, RDR address in SAR, the start address of the RAM area where the data will be received in DAR, and 128 (H′0080) in CRA. CRB can be set to any value.
2. Set the start address of the register information at the DTC vector address.
3. Set the corresponding bit in DTCER to 1.
4. Set the SCI to the appropriate receive mode. Set the RIE bit in SCR to 1 to enable the reception complete (RXI) interrupt. Since the generation of a receive error during the SCI reception operation will disable subsequent reception, the CPU should be enabled to accept receive error interrupts.
5. Each time the reception of one byte of data has been completed on the SCI, the RDRF flag in SSR is set to 1, an RXI interrupt is generated, and the DTC is activated. The receive data is transferred from RDR to RAM by the DTC. DAR is incremented and CRA is decremented. The RDRF flag is automatically cleared to 0.
6. When CRA becomes 0 after 128 data transfers have been completed, the RDRF flag is held at 1, the DTCE bit is cleared to 0, and an RXI interrupt request is sent to the CPU. The interrupt handling routine will perform wrap-up processing.

7.8.2 Software Activation

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

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

7.9 Usage Notes

7.9.1 Module Stop Mode Setting

DTC operation can be enabled or disabled by the module stop control register (MSTPCR). In the initial state, DTC operation is enabled. Access to DTC registers is disabled when module stop mode is set. Note that when the DTC is being activated, module stop mode cannot be specified. For details, refer to section 22, Power-Down Modes.

7.9.2 On-Chip RAM

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

7.9.3 DTCE Bit Setting

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

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

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

Section 8 I/O Ports

Table 8.1 is a summary of the port functions. The pins of each port also function as input/output pins of peripheral modules and interrupt input pins. Each input/output port includes a data direction register (DDR) that controls input/output and a data register (DR) that stores output data. DDR and DR are not provided for an input-only port.

Ports 1 to 4, 6, and A have built-in input pull-up MOSs. For port A, the on/off status of the input pull-up MOS is controlled by DDR and ODR. Ports 1 to 4, and 6 have an input pull-up MOS control register (PCR), in addition to DDR and DR, to control the on/off status of the input pull-up MOSs.

Port 3 pins and pins 47 to 44 have built-in de-bouncers (DBn) that eliminate noises in the input signals.

Port 4 are designed for retain state outputs (RSn), which retain the output values on the pins even if a reset is generated when watchdog timer has overflowed.

Ports 1 to 6, and 8 to E can drive a single TTL load and 30 pF capacitive load. All the I/O ports can drive a Darlington transistor in output mode. Ports 8, and C0 to C3 are NMOS push-pull output.

Table 8.1 Port Functions

Port	Description	Single-Chip Mode	I/O Status
Port 1	General I/O port	P17	Built-in input pull-up MOS
		P16	
		P15	
		P14	
		P13	
		P12	
		P11	
		P10	
Port 2	General I/O port	P23	Built-in input pull-up MOS
		P22	
		P21	
		P20	

Port	Description	Single-Chip Mode	I/O Status
Port 3	General I/O port also functioning as de-bounce input	P37/ExDB7 P36/ExDB6 P35/ExDB5 P34/ExDB4 P33/ExDB3 P32/ExDB2 P31/ExDB1 P30/ExDB0	Built-in input pull-up MOS
Port 4	General I/O port also functioning as interrupt input, retain state output, and de-bounce input	P47/$\overline{\text{IRQ7}}$/RS7/DB7/HC7 P46/$\overline{\text{IRQ6}}$/RS6/DB6/HC6 P45/$\overline{\text{IRQ5}}$/RS5/DB5/HC5 P44/$\overline{\text{IRQ4}}$/RS4/DB4/HC4 P43/$\overline{\text{IRQ3}}$/RS3/HC3 P42/$\overline{\text{IRQ2}}$/RS2/HC2 P41/$\overline{\text{IRQ1}}$/RS1/HC1 P40/$\overline{\text{IRQ0}}$/RS0/HC0	Built-in input pull-up MOS (sink current 12 mA)
Port 5	General I/O port also functioning as interrupt input, system clock output, PWMX output, and SCI_1 input/output	P57/$\overline{\text{IRQ15}}$/PWX1 P56/$\overline{\text{IRQ14}}$/PWX0/ϕ/EXCL P53/$\overline{\text{IRQ11}}$/RxD1 P52/$\overline{\text{IRQ10}}$/TxD1	
Port 6	General I/O port	P63 P62 P61 P60	Built-in input pull-up MOS
Port 7	General I/O port also functioning as A/D converter analog input and interrupt input	P77/$\overline{\text{ExIRQ7}}$/AN7 P76/$\overline{\text{ExIRQ6}}$/AN6 P75/$\overline{\text{ExIRQ5}}$/AN5 P74/$\overline{\text{ExIRQ4}}$/AN4 P73/$\overline{\text{ExIRQ3}}$/AN3 P72/$\overline{\text{ExIRQ2}}$/AN2 P71/$\overline{\text{EXIRQ1}}$/AN1 P70/$\overline{\text{EXIRQ0}}$/AN0	

Port	Description	Single-Chip Mode	I/O Status
Port 8	General I/O port also functioning as A/D converter external trigger input, interrupt input, SCI_1 and SCI_3 input/output, and IIC_0 and IIC_1 input/output	P87/$\overline{\text{ExIRQ15}}$/TxD3/$\overline{\text{ADTRG}}$ P86/$\overline{\text{ExIRQ14}}$/RxD3 P85/$\overline{\text{ExIRQ13}}$/SCK1 P84/$\overline{\text{ExIRQ12}}$/SCK3 P83/$\overline{\text{ExIRQ11}}$/SDA1 P82/$\overline{\text{ExIRQ10}}$/SCL1 P81/$\overline{\text{ExIRQ9}}$/SDA0 P80/$\overline{\text{ExIRQ8}}$/SCL0	NMOS push-pull output
Port A	General I/O port also functioning as DTC event counter input	PA7/EVENT7 PA6/EVENT6 PA5/EVENT5 PA4/EVENT4 PA3/EVENT3 PA2/EVENT2 PA1/EVENT1 PA0/EVENT0	Built-in input pull-up MOS
Port C	General I/O port also functioning as PWMX output, and IIC_2 and IIC_3 input/output	PC7/PWX3 PC6/PWX2 PC3/SDA3 PC2/SCL3 PC1/SDA2 PC0/SCL2	NMOS push-pull output (PC0 to PC3)
Port E	General I/O port also functioning as LPC input/output	PE7/SERIRQ PE6/LCLK PE5/$\overline{\text{LRESET}}$ PE4/$\overline{\text{LFRAME}}$ PE3/LAD3 PE2/LAD2 PE1/LAD1 PE0/LAD0	

8.1 Port 1

Port 1 is an 8-bit I/O port. 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)

8.1.1 Port 1 Data Direction Register (P1DDR)

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

Bit	Bit Name	Initial Value	R/W	Description
7	P17DDR	0	W	The corresponding port 1 pins function as address output port when the P1DDR bits are set to 1, and as input port when 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	

8.1.2 Port 1 Data Register (P1DR)

P1DR stores output data for the port 1 pins.

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

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

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

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

8.1.4 Port 1 Input Pull-Up MOS

Port 1 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used regardless of the operating mode. Table 8.2 summarizes the input pull-up MOS states.

Table 8.2 Port 1 Input Pull-Up MOS States

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

[Legend]

Off: Always off.

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

8.2 Port 2

Port 2 is a 4-bit I/O port. 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)

8.2.1 Port 2 Data Direction Register (P2DDR)

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

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	W	Reserved
6	—	0	W	
5	—	0	W	
4	—	0	W	
3	P23DDR	0	W	The corresponding port 2 pins function as address output port when the P2DDR bits are set to 1, and as input port when cleared to 0.
2	P22DDR	0	W	
1	P21DDR	0	W	
0	P20DDR	0	W	

8.2.2 Port 2 Data Register (P2DR)

P2DR stores output data for the port 2 pins.

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

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

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

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

8.2.4 Port 2 Input Pull-Up MOS

Port 2 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used regardless of the operating mode. Table 8.3 summarizes the input pull-up MOS states.

Table 8.3 Port 2 Input Pull-Up MOS States

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

[Legend]

Off: Always off.

On/Off: On when P2DDR = 0 and P2PCR = 1; otherwise off.

8.3 Port 3

Port 3 is an 8-bit I/O port. Port 3 pins also function as de-bounce input. 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)
- Noise canceler enable register (P3NCE)
- Noise canceler mode control register (P3NCMC)
- Noise cancel cycle setting register (NCCS)

8.3.1 Port 3 Data Direction Register (P3DDR)

The individual bits of P3DDR specify input or output for the pins of port 3.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DDR	0	W	The corresponding port 3 pins function as output port when the P3DDR bits are set to 1, and as input port when 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	

8.3.2 Port 3 Data Register (P3DR)

P3DR stores output data for the port 3 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DR	0	R/W	P3DR stores output data for the port 3 pins that are used as the general output port. If a port 3 read is performed while the P3DDR bits are set to 1, the P3DR values are read. When the P3DDR bits are cleared to 0, 1 is read.
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	

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

P3PCR controls the port 3 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P37PCR	0	R/W	When the pins are in input state, the corresponding input pull-up MOS is turned on when a P3PCR bit is set to 1.
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	

8.3.4 Noise Canceler Enable Register (P3NCE)

P3NCE enables or disables the noise canceler circuit at port 3.

Bit	Bit Name	Initial Value	R/W	Description
7	P37NCE	0	R/W	When the noise canceler circuit is enabled, the pin state is fetched in the P3DR in the sampling cycle set by the NCCS. The operating state changes according to the other control bits. Check the pin functions.
6	P36NCE	0	R/W	
5	P35NCE	0	R/W	
4	P34NCE	0	R/W	
3	P33NCE	0	R/W	
2	P32NCE	0	R/W	
1	P31NCE	0	R/W	
0	P30NCE	0	R/W	

8.3.5 Noise Canceler Mode Control Register (P3NCMC)

P3NCMC controls whether 1 or 0 is expected for the input signal to port 3 in bit units.

Bit	Bit Name	Initial Value	R/W	Description
7	P37NCMC	1	R/W	1 expected: 1 is stored in the port data register while 1 is input stably 0 expected: 0 is stored in the port data register while 0 is input stably
6	P36NCMC	1	R/W	
5	P35NCMC	1	R/W	
4	P34NCMC	1	R/W	
3	P33NCMC	1	R/W	
2	P32NCMC	1	R/W	
1	P31NCMC	1	R/W	
0	P30NCMC	1	R/W	

8.3.6 Noise Canceler Cycle Setting Register (NCCS)

NCCS controls the sampling cycles of the noise canceler.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	Undefined	R/W	Reserved The read value is undefined.
2	NCCK2	0	R/W	These bits set the sampling cycle of the noise cancelers. • When ϕ = 25 MHz 000: 80 ns ϕ/2 001: 1.28 µs ϕ/32 010: 20.48 µs ϕ/512 011: 327.7 µs ϕ/8192 100: 1.31 ms ϕ/32768 101: 2.62 ms ϕ/65536 110: 5.24 ms ϕ/131072 111: 10.49 ms ϕ/262144
1	NCCK1	0	R/W	
0	NCCK0	0	R/W	

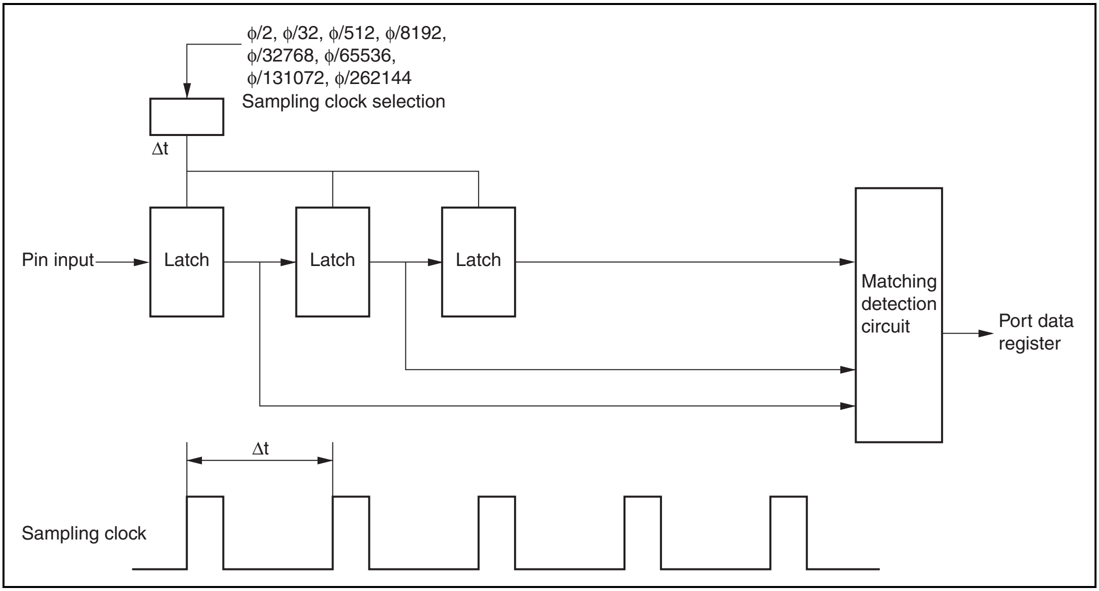


Figure 8.1 Noise Canceler Circuit

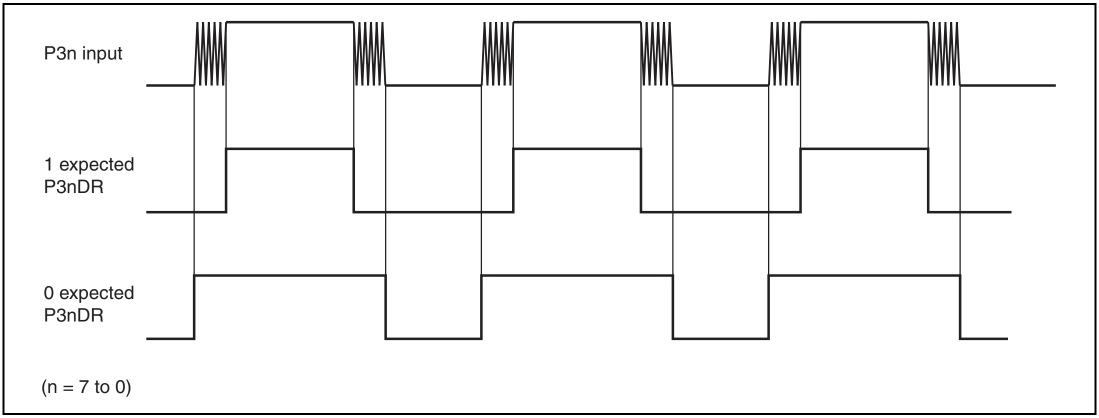


Figure 8.2 Noise Canceler Operation

8.3.7 Pin Functions

The pin function is switched as shown below according to the combination of the PnDDR bit and the P3nNCE bit.

P3nDDR	0		1
P3nNCE	0	1	x
Pin function	ExDBn input pin	P3n input pin	P3n output pin

[Legend]

n = 7 to 0

x: Don't care

8.3.8 Port 3 Input Pull-Up MOS

Port 3 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used in single-chip mode. Table 8.4 summarizes the input pull-up MOS states.

Table 8.4 Port 3 Input Pull-Up MOS States

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

[Legend]

Off: Always off.

On/Off: On when input state and P3PCR = 1; otherwise off.

8.4 Port 4

Port 4 is an 8-bit I/O port. Port 4 pins also function as external interrupt input and de-bounce input. Port 4 has the following registers.

- Port 4 data direction register (P4DDR)
- Port 4 data register (P4DR)
- Port 4 pull-up MOS control register (P4PCR)
- Noise canceler enable register (P4NCE)
- Noise canceler mode control register (P4NCMC)
- Noise cancel cycle setting register (NCCS)

8.4.1 Port 4 Data Direction Register (P4DDR)

The individual bits of P4DDR specify input or output for the pins of port 4.

These bits are initialized only by a system reset, and retain their values even when an internal reset signal is generated by the WDT.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DDR	0	W	If port 4 pins are specified for use as the general I/O port, the corresponding port 4 pins function as output port when the P4DDR bits are set to 1, and as input port when cleared to 0.
6	P46DDR	0	W	
5	P45DDR	0	W	
4	P44DDR	0	W	
3	P43DDR	0	W	
2	P42DDR	0	W	
1	P41DDR	0	W	
0	P40DDR	0	W	

8.4.2 Port 4 Data Register (P4DR)

P4DR stores output data for the port 4 pins.

These bits are initialized only by a system reset, and retain their values even when an internal reset signal is generated by the WDT.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DR	0	R/W	P4DR stores output data for the port 4 pins that are used as the general output port. If a port 4 read is performed while the P4DDR bits are set to 1, the P4DR values are read. If a port 4 read is performed while the P4DDR bits are cleared to 0, the pin states are read.
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	

8.4.3 Port 4 Pull-Up MOS Control Register (P4PCR)

P4PCR controls the port 4 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7	P47PCR	0	R/W	When the pins are in input state, the corresponding input pull-up MOS is turned on when a P4PCR bit is set to 1.
6	P46PCR	0	R/W	
5	P45PCR	0	R/W	
4	P44PCR	0	R/W	
3	P43PCR	0	R/W	
2	P42PCR	0	R/W	
1	P41PCR	0	R/W	
0	P40PCR	0	R/W	

8.4.4 Noise Canceler Enable Register (P4NCE)

P4NCE enables or disables the noise canceler circuit at port 4.

Bit	Bit Name	Initial Value	R/W	Description
7	P47NCE	0	R/W	When the noise canceler circuit is enabled, the pin state is fetched in the P4DR in the sampling cycle set by the NCCS. The operating state changes according to the other control bits. Check the pin functions.
6	P46NCE	0	R/W	
5	P45NCE	0	R/W	
4	P44NCE	0	R/W	
3 to 0	—	All 0	R/W	Reserved

8.4.5 Noise Canceler Mode Control Register (P4NCMC)

P4NCMC controls whether 1 or 0 is expected for the input signal to port 4 in bit units.

Bit	Bit Name	Initial Value	R/W	Description
7	P47NCMC	1	R/W	1 expected: 1 is stored in the port data register while 1 is input stably 0 expected: 0 is stored in the port data register while 0 is input stably
6	P46NCMC	1	R/W	
5	P45NCMC	1	R/W	
4	P44NCMC	1	R/W	
3 to 0	—	All 1	R/W	Reserved

8.4.6 Noise Canceler Cycle Setting Register (NCCS)

NCCS controls the sampling cycles of the noise cancelers.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	Undefined	R/W	Reserved The read value is undefined.
2	NCCK2	0	R/W	These bits set the sampling cycle of the noise cancelers. • When ϕ = 25 MHz 000: 80 ns ϕ/2 001: 1.28 μs ϕ/32 010: 20.48 μs ϕ/512 011: 327.7 μs ϕ/8192 100: 1.31 ms ϕ/32768 101: 2.62 ms ϕ/65536 110: 5.24 ms ϕ/131072 111: 10.49 ms ϕ/262144
1	NCCK1	0	R/W	
0	NCCK0	0	R/W	

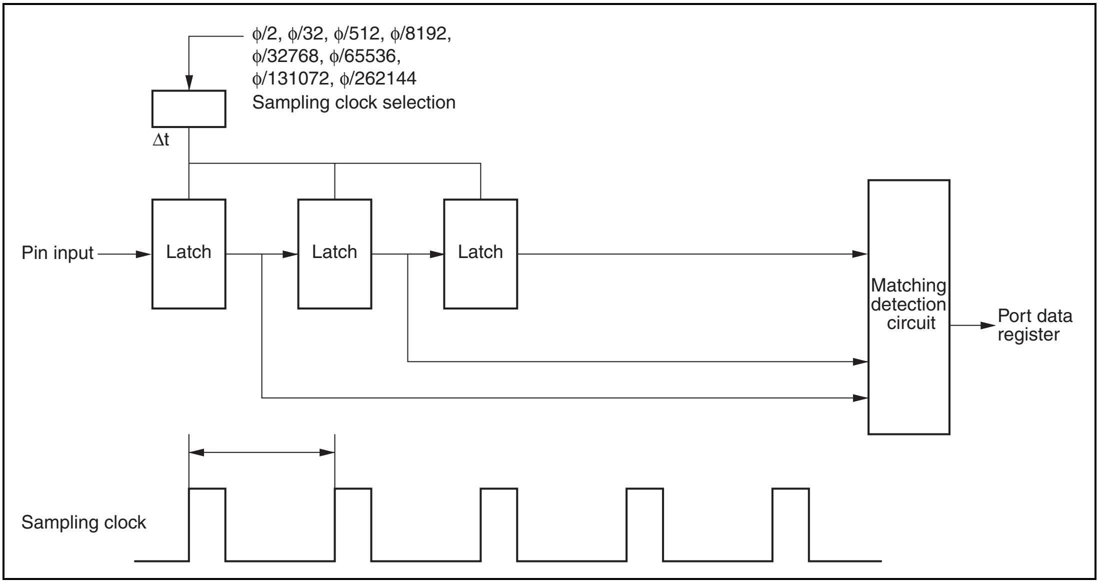


Figure 8.3 Noise Canceler Circuit

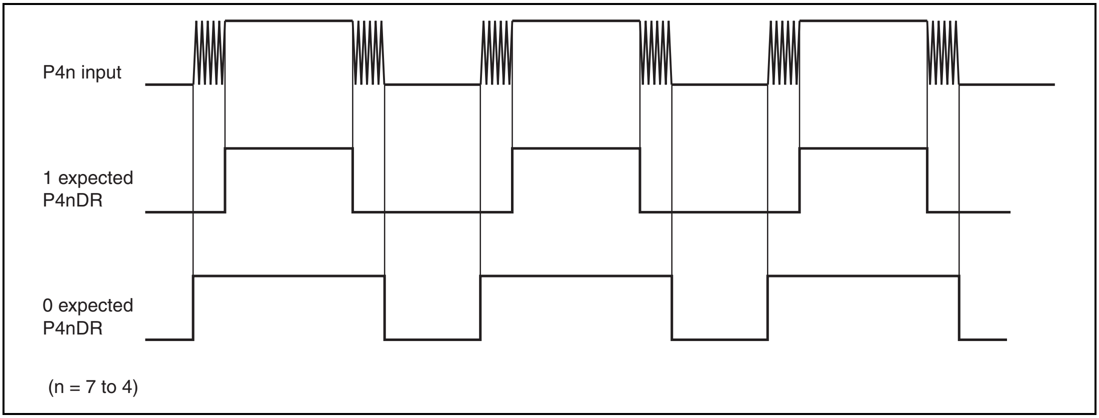


Figure 8.4 Noise Canceler Operation

8.4.7 Pin Functions

The relationship between register setting values and pin functions are as follows.

- P47 to P44

The pin function is switched as shown below according to the P4nDDR bit and P4nNCE bit.

When the ISSn bit in ISSR is cleared to 0 and the IRQnE bit in IER of the interrupt controller is set to 1, this pin can be used as the IRQn input pin. To use this pin as the IRQn input pin, clear the P4nDDR bit to 0.

P4nDDR	0		1
P4nNCE	0	1	x
Pin function	P4n input pin	DBn input	P4n output pin
	$\overline{\text{IRQn}}$ input pin	$\overline{\text{IRQn}}$ input pin (with the noise canceler)	

[Legend]
n = 7 to 4
x: Don't care

- P43 to P40

The pin function is switched as shown below according to the P4nDDR bit.

When the ISSn bit in ISSR is cleared to 0 and the IRQnE bit in IER of the interrupt controller is set to 1, this pin can be used as the IRQn input pin. To use this pin as the IRQn input pin, clear the P4nDDR bit to 0.

P4nDDR	0	1
Pin function	P4n input pin	P4n output pin
	$\overline{\text{IRQn}}$ input pin	

[Legend]
n = 3 to 0

8.4.8 Port 4 Input Pull-Up MOS

Port 4 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used in single-chip mode. Table 8.5 summarizes the input pull-up MOS states.

Table 8.5 Port 4 Input Pull-Up MOS States

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

[Legend]
Off: Always off.
On/Off: On when input state and P4PCR = 1: otherwise off.

8.5 Port 5

Port 5 is a 4-bit I/O port. Port 5 pins also function as interrupt input, PWMX output, and SCI_1 input/output. Port 5 has the following registers.

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

8.5.1 Port 5 Data Direction Register (P5DDR)

The individual bits of P5DDR specify input or output for the pins of port 5.

Bit	Bit Name	Initial Value	R/W	Description
7	P57DDR	0	W	If port 5 pins are specified for use as the general I/O port, the corresponding port 5 pins function as output port when the P5DDR bits are set to 1, and as input port when cleared to 0.
6	P56DDR	0	W	The corresponding port 5 pin functions as the system clock output pin (ϕ) when this bit is set to 1, and as the general I/O port when cleared to 0.
5, 4	—	All 0	W	Reserved
3	P53DDR	0	W	If port 5 pins are specified for use as the general I/O port, the corresponding port 5 pins function as output port when the P5DDR bits are set to 1, and as input port when cleared to 0.
2	P52DDR	0	W	
1, 0	—	All 0	W	Reserved

8.5.2 Port 5 Data Register (P5DR)

P5DR stores output data for the port 5 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P57DR	0	R/W	P5DR stores output data for the port 5 pins that are used as the general output port. If a port 5 read is performed while the P5DDR bits are set to 1, the P5DR values are read. If a port 5 read is performed while the P5DDR bits are cleared to 0, the pin states are read.
6	P56DR	Undefined*	R	
5, 4	—	All 0	R/W	Reserved
3	P53DR	0	R/W	See the description of bits 7 and 6.
2	P52DR	0	R/W	
1, 0	—	All 0	R/W	Reserved

Note: * The initial value is determined in accordance with the pin state of P56.

8.5.3 Pin Functions

The relationship between register setting values and pin functions are as follows.

- P57/$\overline{\text{IRQ15}}$/PWX1

  The pin function is switched as shown below according to the combination of the OEB bit in DACR of PWMX and the P57DDR bit.

  When the ISS15 bit in ISSR16 is cleared to 0 and the IRQ15E bit in IER16 of the interrupt controller is set to 1, this pin can be used as the $\overline{\text{IRQ15}}$ input pin. To use this pin as the $\overline{\text{IRQ15}}$ input pin, clear the P57DDR bit to 0.

OEB	0		1
P57DDR	0	1	x
Pin function	P57 input pin	P57 output pin	PWX1 output pin
	$\overline{\text{IRQ15}}$ input pin		

[Legend]

x: Don't care

- P56/$\overline{\text{IRQ14}}$/PWX0/φ/EXCL

  The pin function is switched as shown below according to the combination of the OEA bit in DACR of PWMX, the EXCLE bit in LPWRCR, and the P56DDR bit.

  When the ISS14 bit in ISSR16 is cleared to 0 and the IRQ14E bit in IER16 of the interrupt controller is set to 1, this pin can be used as the $\overline{\text{IRQ14}}$ input pin. To use this pin as the $\overline{\text{IRQ14}}$ input pin, clear the P56DDR bit to 0.

OEA	0			1
P56DDR	0		1	x
EXCLE	0	1	x	x
Pin function	P56 input pin $\overline{\text{IRQ14}}$ input pin	EXCL input pin	φ output pin	PWX0 output pin

[Legend]

x: Don't care

- P53/$\overline{\text{IRQ11}}$/RxD1

  The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI_1, the SMIF bit in SCMR, and the P53DDR bit.

  When the ISS11 bit in ISSR16 is cleared to 0 and the IRQ11E bit in IER16 of the interrupt controller is set to 1, this pin can be used as the $\overline{\text{IRQ11}}$ input pin. To use this pin as the $\overline{\text{IRQ11}}$ input pin, clear the P53DDR bit to 0.

RE	x		0	1
SMIF	0		1	
P53DDR	0	1	x	x
Pin function	P53 input pin $\overline{\text{IRQ11}}$ input pin	P53 output pin	RxD1 input pin	RxD1 input/output pin

[Legend]

x: Don't care

- P52/$\overline{\text{IRQ10}}$/TxD1

  The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI_1, the SMIF bit in SCMR, and the P52DDR bit.

  When the ISS10 bit in ISSR16 is cleared to 0 and the IRQ10E bit in IER16 of the interrupt controller is set to 1, this pin can be used as the $\overline{\text{IRQ10}}$ input pin. To use this pin as the $\overline{\text{IRQ10}}$ input pin, clear the P52DDR bit to 0.

TE	x		0	1	
SMIF	0		1		
P52DDR	0	1	x	0	1
Pin function	P52 input pin	P52 output pin	TxD1 output pin	P52 input pin	P52 output pin
	$\overline{\text{IRQ10}}$ input pin			$\overline{\text{IRQ10}}$ input pin	

[Legend]

x: Don't care

8.6 Port 6

Port 6 is a 4-bit I/O port. Port 6 has the following registers.

- Port 6 data direction register (P6DDR)
- Port 6 data register (P6DR)
- Port 6 pull-up MOS control register (P6PCR)

8.6.1 Port 6 Data Direction Register (P6DDR)

The individual bits of P6DDR specify input or output for the pins of port 6.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	W	Reserved
6	—	0	W	
5	—	0	W	
4	—	0	W	
3	P63DDR	0	W	If port 6 pins are specified for use as the general I/O port, the corresponding port 6 pins function as output port when the P6DDR bits are set to 1, and as input ports when cleared to 0.
2	P62DDR	0	W	
1	P61DDR	0	W	
0	P60DDR	0	W	

8.6.2 Port 6 Data Register (P6DR)

P6DR stores output data for the port 6 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved
6	—	0	R/W	
5	—	0	R/W	
4	—	0	R/W	
3	P63DR	0	R/W	P6DR stores output data for the port 6 pins that are used as the general output port. If a port 6 read is performed while the P6DDR bits are set to 1, the P6DR values are read. If a port 6 read is performed while the P6DDR bits are cleared to 0, the pin states are read.
2	P62DR	0	R/W	
1	P61DR	0	R/W	
0	P60DR	0	R/W	

8.6.3 Port 6 Pull-Up MOS Control Register (P6PCR)

P6PCR controls the port 6 built-in input pull-up MOSs.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	W	Reserved
3	P63PCR	0	W	When the pins are in input state, the corresponding input pull-up MOS is turned on when a P6PCR bit is set to 1.
2	P62PCR	0	W	
1	P61PCR	0	W	
0	P60PCR	0	W	

8.6.4 Port 6 Input Pull-Up MOS

Port 6 has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used in single-chip mode. Table 8.6 summarizes the input pull-up MOS states.

Table 8.6 Port 6 Input Pull-Up MOS States

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

[Legend]

Off: Always off.

On/Off: On when input state and P6PCR = 1: otherwise off.

8.7 Port 7

Port 7 is an 8-bit input port. Port 7 pins also function as A/D converter analog input and interrupt input. Port 7 has the following register.

- Port 7 input data register (P7PIN)

8.7.1 Port 7 Input Data Register (P7PIN)

P7PIN indicates the pin states.

Bit	Bit Name	Initial Value	R/W	Description
7	P77PIN	Undefined*	R	When a P7PIN read is performed, the pin states are always read.
6	P76PIN	Undefined*	R	
5	P75PIN	Undefined*	R	
4	P74PIN	Undefined*	R	
3	P73PIN	Undefined*	R	
2	P72PIN	Undefined*	R	
1	P71PIN	Undefined*	R	
0	P70PIN	Undefined*	R	

Note: * The initial value is determined in accordance with the pin states of P77 to P70.

8.7.2 Pin Functions

Each pin of port 7 can also be used as the interrupt input pins ($\overline{\text{ExIRQ0}}$ to $\overline{\text{ExIRQ7}}$) and analog input pins of the A/D converter (AN0 to AN7). By setting the ISS bit of the ISSR to 1, the pins can also be used as the interrupt input pin ($\overline{\text{ExIRQ0}}$ to $\overline{\text{ExIRQ7}}$).

When the interrupt input pin is set, do not use the pins for the A/D converter.

- P77/$\overline{\text{ExIRQ7}}$/AN7

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter and the ISS7 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

CH2 to CH0	B'111	Other than B'111	
ISS7	0	0	1
Pin function	AN7 input pin	P77 input pin	$\overline{\text{ExIRQ7}}$ input pin

- P76/$\overline{\text{ExIRQ6}}$/AN6

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE bit in ADCR, and the ISS6 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1		
CH2 to CH0	B'110	Other than B'110		B'110 to B'111	B'000 to B'101	
ISS6	0	0	1	0	0	1
Pin function	AN6 input pin	P76 input pin	$\overline{\text{ExIRQ6}}$ input pin	AN6 input pin	P76 input pin	$\overline{\text{ExIRQ6}}$ input pin

[Legend]

x: Don't care

- P75/$\overline{\text{ExIRQ5}}$/AN5

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE bit in ADCR, and the ISS5 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1		
CH2 to CH0	B'101	Other than B'101		B'101 to B'111	B'000 to B'100	
ISS5	0	0	1	0	0	1
Pin function	AN5 input pin	P75 input pin	$\overline{\text{ExIRQ5}}$ input pin	AN5 input pin	P75 input pin	$\overline{\text{ExIRQ5}}$ input pin

[Legend]

x: Don't care

- P74/$\overline{\text{ExIRQ4}}$/AN4

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE bit in ADCR, and the ISS4 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1		
CH2 to CH0	B'100	Other than B'100		B'100 to B'111	B'000 to B'011	
ISS4	0	0	1	0	0	1
Pin function	AN4 input pin	P74 input pin	$\overline{\text{ExIRQ4}}$ input pin	AN4 input pin	P74 input pin	$\overline{\text{ExIRQ4}}$ input pin

[Legend]

x: Don't care

- P73/$\overline{\text{ExIRQ3}}$/AN3

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE and SCANS bits in ADCR, and the ISS3 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1					
SCANS	x			0			1		
CH2 to CH0	B'011	Other than B'011		B'011	Other than B'011		B'011 to B'111	B'000 to B'010	
ISS3	0	0	1	0	0	1	0	0	1
Pin function	AN3 input pin	P73 input pin	$\overline{\text{ExIRQ3}}$ input pin	AN3 input pin	P73 input pin	$\overline{\text{ExIRQ3}}$ input pin	AN3 input pin	P73 input pin	$\overline{\text{ExIRQ3}}$ input pin

[Legend]

x: Don't care

- P72/$\overline{\text{ExIRQ2}}$/AN2

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE and SCANS bits in ADCR, and the ISS2 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1					
SCANS	x			0			1		
CH2 to CH0	B'010	Other than B'010		B'010 to B'011	Other than B'010 to B'011		B'010 to B'111	B'000 to B'001	
ISS2	0	0	1	0	0	1	0	0	1
Pin function	AN2 input pin	P72 input pin	$\overline{\text{ExIRQ2}}$ input pin	AN2 input pin	P72 input pin	$\overline{\text{ExIRQ2}}$ input pin	AN2 input pin	P72 input pin	$\overline{\text{ExIRQ2}}$ input pin

[Legend]

x: Don't care

- P71/$\overline{\text{ExIRQ1}}$/AN1

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE and SCANS bits in ADCR, and the ISS1 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1					
SCANS	x			0			1		
CH2 to CH0	B'001	Other than B'001		B'001 to B'011	Other than B'001 to B'011		B'001 to B'111	B'000	
ISS1	0	0	1	0	0	1	0	0	1
Pin function	AN1 input pin	P71 input pin	$\overline{\text{ExIRQ1}}$ input pin	AN1 input pin	P71 input pin	$\overline{\text{ExIRQ1}}$ input pin	AN1 input pin	P71 input pin	$\overline{\text{ExIRQ1}}$ input pin

[Legend]

x: Don't care

- P70/$\overline{\text{ExIRQ0}}$/AN0

  The pin function is switched as shown below according to the combination of the CH2 to CH0 bits in ADCSR of the A/D converter, the SCANE and SCANS bits in ADCR, and the ISS0 bit in ISSR of the interrupt controller. Do not set these bits to other values than those shown in the following table.

SCANE	0			1			
SCANS	x			0			1
CH2 to CH0	B'000	Other than B'000		B'000 to B'011	B'000 to B'011		B'000 to B'111
ISS0	0	0	1	0	0	1	0
Pin function	AN0 input pin	P70 input pin	$\overline{\text{ExIRQ0}}$ input pin	AN0 input pin	P70 input pin	$\overline{\text{ExIRQ0}}$ input pin	AN0 input pin

[Legend]

x: Don't care

Table 8.7 Analog Port Effective Condition

ADCR		ADCSR			Analog Port							
SCANE	SCANS	CH2	CH1	CH0	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
0	x	0	0	0	—	—	—	—	—	—	—	ON
0	x	0	0	1	—	—	—	—	—	—	ON	—
0	x	0	1	0	—	—	—	—	—	ON	—	—
0	x	0	1	1	—	—	—	—	ON	—	—	—
0	x	1	0	0	—	—	—	ON	—	—	—	—
0	x	1	0	1	—	—	ON	—	—	—	—	—
0	x	1	1	0	—	ON	—	—	—	—	—	—
0	x	1	1	1	ON	—	—	—	—	—	—	—
1	0	0	0	0	—	—	—	—	—	—	—	ON
1	0	0	0	1	—	—	—	—	—	—	ON	ON
1	0	0	1	0	—	—	—	—	—	ON	ON	ON
1	0	0	1	1	—	—	—	—	ON	ON	ON	ON
1	0	1	0	0	—	—	—	ON	—	—	—	—
1	0	1	0	1	—	—	ON	ON	—	—	—	—
1	0	1	1	0	—	ON	ON	ON	—	—	—	—
1	0	1	1	1	ON	ON	ON	ON	—	—	—	—
1	1	0	0	0	—	—	—	—	—	—	—	ON
1	1	0	0	1	—	—	—	—	—	—	ON	ON
1	1	0	1	0	—	—	—	—	—	ON	ON	ON
1	1	0	1	1	—	—	—	—	ON	ON	ON	ON
1	1	1	0	0	—	—	—	ON	ON	ON	ON	ON
1	1	1	0	1	—	—	ON	ON	ON	ON	ON	ON
1	1	1	1	0	—	ON	ON	ON	ON	ON	ON	ON
1	1	1	1	1	ON	ON	ON	ON	ON	ON	ON	ON

[Legend]

x: Don't care

8.8 Port 8

Port 8 is an 8-bit I/O port. Port 8 pins also function as the A/D converter external trigger input, SCI_1 and SCI_3 input/output, IIC_0 and IIC_1 input/output, and interrupt input. Pins 83 to 80 perform the NMOS push-pull output. Port 8 has the following registers.

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

8.8.1 Port 8 Data Direction Register (P8DDR)

The individual bits of P8DDR specify input or output for the pins of port 8.

Bit	Bit Name	Initial Value	R/W	Description
7	P87DDR	0	W	If port 8 pins are specified for use as the general I/O port, the corresponding port 8 pins function as output port when the P8DDR bits are set to 1, and as input port when cleared to 0.
6	P86DDR	0	W	
5	P85DDR	0	W	
4	P84DDR	0	W	
3	P83DDR	0	W	
2	P82DDR	0	W	
1	P81DDR	0	W	
0	P80DDR	0	W	

8.8.2 Port 8 Data Register (P8DR)

P8DR stores output data for the port 8 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P87DR	0	R/W	P8DR stores output data for the port 8 pins that are used as the general output port. If a port 8 read is performed while the P8DDR bits are set to 1, the P8DR values are read. If a port 8 read is performed while the P8DDR bits are cleared to 0, the pin states are read.
6	P86DR	0	R/W	
5	P85DR	0	R/W	
4	P84DR	0	R/W	
3	P83DR	0	R/W	
2	P82DR	0	R/W	
1	P81DR	0	R/W	
0	P80DR	0	R/W	

8.8.3 Pin Functions

The relationship between register setting values and pin functions are as follows.

- P87/$\overline{\text{ExIRQ15}}$/TxD3/$\overline{\text{ADTRG}}$

  The pin function is switched as shown below according to the combination of the TE bit in SCR of SCI_3, the SMIF bit in SCMR, and the P87DDR bit.

  When the TRGS1 and EXTRGS bits are both set to 1 and the TRGS0 bit is cleared to 0 in ADCR of the A/D converter, this pin can be used as the $\overline{\text{ADTRG}}$ input pin.

  When the ISS15 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ15}}$ input pin. To use this pin as the $\overline{\text{ExIRQ15}}$ input pin, clear the P87DDR bit to 0.

P87DDR	0		1		
SMIF	0	1	0	1	0
TE	0	x	0	x	1
Pin function	P87 input pin		P87 input pin		TxD3 output pin
	$\overline{\text{ExIRQ15}}$ input pin/ $\overline{\text{ADTRG}}$ input pin				

[Legend]

x: Don't care

- P86/$\overline{\text{ExIRQ14}}$/RxD3

  The pin function is switched as shown below according to the combination of the RE bit in SCR of SCI_3, the SMIF bit in SCMR, and the P86DDR bit.

  When the ISS14 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ14}}$ input pin. To use this pin as the $\overline{\text{ExIRQ14}}$ input pin, clear the P86DDR bit to 0.

P86DDR	0			1
SMIF	0		1	0
RE	0	1		0
Pin function	P86 input pin	RxD3 input pin	RxD3 input/output pin	P86 output pin
	$\overline{\text{ExIRQ14}}$ input pin			

- P85/$\overline{\text{ExIRQ13}}$/SCK1

  The pin function is switched as shown below according to the combination of the C/$\overline{\text{A}}$ bit in SMR of SCI_1, the CKE1 and CKE0 bits in SCR, and the P85DDR bit.

  When the ISS13 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ13}}$ input pin. To use this pin as the $\overline{\text{ExIRQ13}}$ input pin, clear the P85DDR bit to 0.

CKE1	0				1
C/$\overline{\text{A}}$	0			1	x
CKE0	0		1	x	x
P85DDR	0	1	x	x	x
Pin function	P85 input pin	P85 output pin	SCK1 output pin	SCK1 output pin	SCK1 input pin
	$\overline{\text{ExIRQ13}}$ input pin				

[Legend]

x: Don't care

- P84/$\overline{\text{ExIRQ12}}$/SCK3

  The pin function is switched as shown below according to the combination of the C/$\overline{\text{A}}$ bit in SMR of SCI_3, the CKE1 and CKE0 bits in SCR, and the P84DDR bit.

  When the ISS12 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ12}}$ input pin. To use this pin as the $\overline{\text{ExIRQ12}}$ input pin, clear the P84DDR bit to 0.

CKE1	0				1
C/$\overline{\text{A}}$	0			1	x
CKE0	0		1	x	x
P84DDR	0	1	x	x	x
Pin function	P84 input pin $\overline{\text{ExIRQ12}}$ input pin	P84 output pin	SCK3 output pin	SCK3 output pin	SCK3 input pin

[Legend]

x: Don't care

- P83/$\overline{\text{ExIRQ11}}$/SDA1

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC_1 and the P83DDR bit.

  When the ISS11 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ11}}$ input pin. To use this pin as the $\overline{\text{ExIRQ11}}$ input pin, clear the P83DDR bit to 0.

  When this pin is used as the P83 output pin, the output format is NMOS push-pull output. The output format for SDA1 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P83DDR	0	1	x
Pin function	P83 input pin $\overline{\text{ExIRQ11}}$ input pin	P83 output pin	SDA1 input/output pin

[Legend]

x: Don't care

- P82/$\overline{\text{ExIRQ10}}$/SCL1

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC_1 and the P82DDR bit.

  When the ISS10 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ10}}$ input pin. To use this pin as the $\overline{\text{ExIRQ10}}$ input pin, clear the P82DDR bit to 0.

  When this pin is used as the P82 output pin, the output format is NMOS push-pull output. The output format for SCL1 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P82DDR	0	1	x
Pin function	P82 input pin	P82 output pin	SCL1 input/output pin
	$\overline{\text{ExIRQ10}}$ input pin		

[Legend]

x: Don't care

- P81/$\overline{\text{ExIRQ9}}$/SDA0

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC_0 and the P81DDR bit.

  When the ISS9 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ9}}$ input pin. To use this pin as the $\overline{\text{ExIRQ9}}$ input pin, clear the P81DDR bit to 0.

  When this pin is used as the P81 output pin, the output format is NMOS push-pull output. The output format for SDA0 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P81DDR	0	1	x
Pin function	P81 input pin	P81 output pin	SDA0 input/output pin
	$\overline{\text{ExIRQ9}}$ input pin		

[Legend]

x: Don't care

- P80/$\overline{\text{ExIRQ8}}$/SCL0

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of IIC_0 and the P80DDR bit.

  When the ISS8 bit in ISSR16 is set to 1, this pin can be used as the $\overline{\text{ExIRQ8}}$ input pin. To use this pin as the $\overline{\text{ExIRQ8}}$ input pin, clear the P80DDR bit to 0.

  When this pin is used as the P80 output pin, the output format is NMOS push-pull output. The output format for SCL0 is NMOS open-drain output, and direct bus drive is possible.

ICE	0		1
P80DDR	0	1	x
Pin function	P80 input pin	P80 output pin	SCL0 input/output pin
	$\overline{\text{ExIRQ8}}$ input pin		

[Legend]

x: Don't care

8.9 Port A

Port A is an 8-bit I/O port. Port A pins also function as event counter input. Pin functions are switched depending on the operating mode. Port A has the following registers.

- Port A data direction register (PADDR)
- Port A output data register (PAODR)
- Port A input data register (PAPIN)

8.9.1 Port A Data Direction Register (PADDR)

The individual bits of PADDR specify input or output for the pins of port A.

Bit	Bit Name	Initial Value	R/W	Description
7	PA7DDR	0	W	The corresponding port A pins function as output port when the PADDR bits are set to 1, and as input port when cleared to 0. This register is assigned to the same address as that of PAPIN. When this address is read, the port A states are 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	

8.9.2 Port A Output Data Register (PAODR)

PAODR stores output data for the port A pins.

Bit	Bit Name	Initial Value	R/W	Description
7	PA7ODR	0	R/W	PAODR stores output data for the port A pins that are used as the general output port.
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	

8.9.3 Port A Input Data Register (PAPIN)

PAPIN indicates the pin states.

Bit	Bit Name	Initial Value	R/W	Description
7	PA7PIN	Undefined*	R	When a PAPIN read is performed, the pin states are always read. This register is assigned to the same address as that of PADDR. When this register is written to, data is written to PADDR and the port A setting is then changed.
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 values are determined in accordance with the pin states of PA7 to PA0.

8.9.4 Pin Functions

The relationship between the operating mode, register setting values, and pin functions are as follows.

Port A functions as event counter input and also as an I/O port, and input or output can be specified in bit units.

- PA7/EVENT7, PA6/EVENT6, PA5/EVENT5, PA4/EVENT4, PA3/EVENT3, PA2/EVENT2, PA1/EVENT1, PA0/EVENT0

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

  When this pin is used as EVENT input pin according to bits ECSB3 to ECSB0 in ECCR of the data transfer controller settings, clear the PAnDDR bit to 0. Though this pin has been set to the EVENT input pin, to use as the PAn output pin, set the PAnDDR bit to 1.

PAnDDR	0	1
Pin function	PAn input pin	PAn output pin
	EVENTn input pin	

[Legend]
n = 7 to 0

8.9.5 Input Pull-Up MOS

Port A has a built-in input pull-up MOS that can be controlled by software. This input pull-up MOS can be used in any operating mode, and can be specified as on or off on a bit-by-bit basis.

PAnDDR	0		1
PAnODR	1	0	x
PAn pull-up MOS	ON	OFF	OFF

[Legend]
n = 7 to 0
x: Don't care

The input pull-up MOS is in the off state after a reset and in hardware standby mode. The prior state is retained in software standby mode.

Table 8.8 summarizes the input pull-up MOS states.

Table 8.8 PortA Input Pull-Up MOS States

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

[Legend]
Off: Always off.
On/Off: On when PADDR = 0 and PAODR = 1; otherwise off.

8.10 Port C

Port C is a 6-bit I/O port. Port C pins also function as PWMX output, and IIC_2 and IIC_3 input/output. The output format of ports C0 to C3 is NMOS push-pull output. Port C has the following registers.

- Port C data direction register (PCDDR)
- Port C output data register (PCODR)
- Port C input data register (PCPIN)

8.10.1 Port C Data Direction Register (PCDDR)

PCDDR is used to specify the input/output attribute of each pin of port C.

Bit	Bit Name	Initial Value	R/W	Description
7	PC7DDR	0	W	The corresponding port C pins function as output port when the PCDDR bits are set to 1, and as input port when cleared to 0. This register is assigned to the same address as that of PCPIN. When this address is read, the port C states are returned. Bits 5 and 4 are reserved.
6	PC6DDR	0	W	
5	—	0	W	
4	—	0	W	
3	PC3DDR	0	W	
2	PC2DDR	0	W	
1	PC1DDR	0	W	
0	PC0DDR	0	W	

8.10.2 Port C Output Data Register (PCODR)

PCODR stores output data for port C.

Bit	Bit Name	Initial Value	R/W	Description
7	PC7ODR	0	R/W	The PCODR register stores the output data for the pins that are used as a general output port. Bits 5 and 4 are reserved.
6	PC6ODR	0	R/W	
5	—	0	R/W	
4	—	0	R/W	
3	PC3ODR	0	R/W	
2	PC2ODR	0	R/W	
1	PC1ODR	0	R/W	
0	PC0ODR	0	R/W	

8.10.3 Port C Input Data Register (PCPIN)

PCPIN indicates the pin states of port C.

Bit	Bit Name	Initial Value	R/W	Description
7	PC7PIN	Undefined*	R	When this register is read, the pin state is read. This register is assigned to the same address as that of PCDDR. When this register is written to, data is written to PCDDR and the port C setting is then changed. Bits 5 and 4 are reserved.
6	PC6PIN	Undefined*	R	
5	—	Undefined	R	
4	—	Undefined	R	
3	PC3PIN	Undefined*	R	
2	PC2PIN	Undefined*	R	
1	PC1PIN	Undefined*	R	
0	PC0PIN	Undefined*	R	

Note: * The initial values are determined in accordance with the states of PC7, PC6, and PC3 to PC0 pins.

8.10.4 Pin Functions

Port C functions as IIC_2 and IIC_3 input/output, and PWMX output. The relationship between register setting values and pin functions are as follows.

- PC7/PWX3

  The pin function is switched as shown below according to the combination of the OEB bit in DACR of the 14-bit PWMX and the PC7DDR bit.

OEB	0		1
PC7DDR	0	1	x
Pin function	PC7 input pin	PC7 output pin	PWX3 output pin

[Legend]
x: Don't care

- PC6/PWX2

  The pin function is switched as shown below according to the combination of the OEA bit in DACR of the 14-bit PWMX and the PC6DDR bit.

OEA	0		1
PC6DDR	0	1	x
Pin function	PC6 input pin	PC6 output pin	PWX2 output pin

[Legend]
x: Don't care

- PC3/SDA3

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of the IIC_3 and the PC3DDR bit.

ICE	0		1
PC3DDR	0	1	x
Pin function	PC3 input pin	PC3 output pin	SDA3 input/output pin

[Legend]
x: Don't care

- PC2/SCL3

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of the IIC_3 and the PC2DDR bit.

ICE	0		1
PC2DDR	0	1	x
Pin function	PC2 input pin	PC2 output pin	SCL3 input/output pin

[Legend]

x: Don't care

- PC1/SDA2

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of the IIC_2 and the PC1DDR bit.

ICE	0		1
PC1DDR	0	1	x
Pin function	PC1 input pin	PC1 output pin	SDA2 input/output pin

[Legend]

x: Don't care

- PC0/SCL2

  The pin function is switched as shown below according to the combination of the ICE bit in ICCR of the IIC_2 and the PC0DDR bit.

ICE	0		1
PC0DDR	0	1	x
Pin function	PC0 input pin	PC0 output pin	SCL2 input/output pin

[Legend]

x: Don't care

8.11 Port E

Port E is an 8-bit I/O port. Port E pins also function as LPC input/output. Port E has the following registers.

- Port E data direction register (PEDDR)
- Port E output data register (PEODR)
- Port E input data register (PEPIN)

8.11.1 Port E Data Direction Register (PEDDR)

PEDDR is used to specify the input/output attribute of each pin of port E.

Bit	Bit Name	Initial Value	R/W	Description
7	PE7DDR	0	W	The corresponding port E pins function as output port when the PEDDR bits are set to 1, and as input port when cleared to 0. This register is assigned to the same address as that of PEPIN. When this address is read, the port E states are returned.
6	PE6DDR	0	W	
5	PE5DDR	0	W	
4	PE4DDR	0	W	
3	PE3DDR	0	W	
2	PE2DDR	0	W	
1	PE1DDR	0	W	
0	PE0DDR	0	W	

8.11.2 Port E Output Data Register (PEODR)

PEODR stores output data for the port E pins.

Bit	Bit Name	Initial Value	R/W	Description
7	PE7ODR	0	R/W	The PEODR register stores the output data for the pins that are used a general output port.
6	PE6ODR	0	R/W	
5	PE5ODR	0	R/W	
4	PE4ODR	0	R/W	
3	PE3ODR	0	R/W	
2	PE2ODR	0	R/W	
1	PE1ODR	0	R/W	
0	PE0ODR	0	R/W	

8.11.3 Port E Input Data Register (PEPIN)

PEPIN indicates the pin states of port E.

Bit	Bit Name	Initial Value	R/W	Description
7	PE7PIN	Undefined*	R	Pin states can be read by performing a read cycle on this register. This register is assigned to the same address as that of PEDDR. When this register is written to, data is written to PEDDR and the port E setting is then changed.
6	PE6PIN	Undefined*	R	
5	PE5PIN	Undefined*	R	
4	PE4PIN	Undefined*	R	
3	PE3PIN	Undefined*	R	
2	PE2PIN	Undefined*	R	
1	PE1PIN	Undefined*	R	
0	PE0PIN	Undefined*	R	

Note: * The initial value of these pins is determined in accordance with the state of pins PE7 to PE0.

8.11.4 Pin Functions

Port E also functions as LPC input/output. The pin function is switched with LPC enabled or disabled. The LPC module is disabled when the LPC1E, LPC2E, and LPC3E bits in HICR0 of LPC are all 0.

- PE7/SERIRQ

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE7DDR bit.

LPC	Disabled		Enabled
PE7DDR	0	1	x
Pin function	PE7 input pin	PE7 output pin	SERIRQ input/output pin

[Legend]

x: Don't care

- PE6/LCLK

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE6DDR bit.

LPC	Disabled		Enabled
PE6DDR	0	1	x
Pin function	PE6 input pin	PE6 output pin	LCLK input pin

[Legend]

x: Don't care

- PE5/$\overline{\text{LRESET}}$

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE5DDR bit.

LPC	Disabled		Enabled
PE5DDR	0	1	x
Pin function	PE5 input pin	PE5 output pin	$\overline{\text{LRESET}}$ input pin

[Legend]

x: Don't care

- PE4/$\overline{\text{LFRAME}}$

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE4DDR bit.

LPC	Disabled		Enabled
PE4DDR	0	1	x
Pin function	PE4 input pin	PE4 output pin	$\overline{\text{LFRAME}}$ input pin

[Legend]

x: Don't care

- PE3/LAD3

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE3DDR bit.

LPC	Disabled		Enabled
PE3DDR	0	1	x
Pin function	PE3 input pin	PE3 output pin	LAD3 input/output pin

[Legend]

x: Don't care

- PE2/LAD2

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE2DDR bit.

LPC	Disabled		Enabled
PE2DDR	0	1	x
Pin function	PE2 input pin	PE2 output pin	LAD2 input/output pin

[Legend]

x: Don't care

- PE1/LAD1

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE1DDR bit.

LPC	Disabled		Enabled
PE1DDR	0	1	x
Pin function	PE1 input pin	PE1 output pin	LAD1 input/output pin

[Legend]

x: Don't care

- PE0/LAD0

  The pin function is switched as shown below according to the LPC enabled/disabled and the PE0DDR bit.

LPC	Disabled		Enabled
PE0DDR	0	1	x
Pin function	PE0 input pin	PE0 output pin	LAD0 input/output pin

[Legend]

x: Don't care

8.12 Change of Peripheral Function Pins

I/O ports that also function as peripheral modules, such as the external interrupt pins, can be changed.

I/O ports that also function as the external interrupt pins are changed according to the setting of ISSR16 and ISSR. I/O ports that also function as the 8-bit timer input pins are changed according to the setting of PTCNT0. The pin name of the peripheral function is indicated by adding ‘Ex’ at the head of the original pin name. In each peripheral function description, the original pin name is used.

8.12.1 IRQ Sense Port Select Register 16 (ISSR16), IRQ Sense Port Select Register (ISSR)

ISSR16 and ISSR select ports that also function as $\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$ input pins.

- ISSR16

Bit	Bit Name	Initial Value	R/W	Description
15	ISS15	0	R/W	0: P57/$\overline{\text{IRQ15}}$ is selected 1: P87/$\overline{\text{ExIRQ15}}$ is selected
14	ISS14	0	R/W	0: P56/$\overline{\text{IRQ14}}$ is selected 1: P86/$\overline{\text{ExIRQ14}}$ is selected
13	ISS13	0	R/W	P85/$\overline{\text{ExIRQ13}}$ is always selected
12	ISS12	0	R/W	P84/$\overline{\text{ExIRQ12}}$ is always selected
11	ISS11	0	R/W	0: P53/$\overline{\text{IRQ11}}$ is selected 1: P83/$\overline{\text{ExIRQ11}}$ is selected
10	ISS10	0	R/W	0: P52/$\overline{\text{IRQ10}}$ is selected 1: P82/$\overline{\text{ExIRQ10}}$ is selected
9	ISS9	0	R/W	P81/$\overline{\text{ExIRQ9}}$ is always selected
8	ISS8	0	R/W	P80/$\overline{\text{ExIRQ8}}$ is always selected

- ISSR

Bit	Bit Name	Initial Value	R/W	Description
7	ISS7	0	R/W	0: P47/$\overline{\text{IRQ7}}$ is selected 1: P77/$\overline{\text{ExIRQ7}}$ is selected
6	ISS6	0	R/W	0: P46/$\overline{\text{IRQ6}}$ is selected 1: P76/$\overline{\text{ExIRQ6}}$ is selected
5	ISS5	0	R/W	0: P45/$\overline{\text{IRQ5}}$ is selected 1: P75/$\overline{\text{ExIRQ5}}$ is selected
4	ISS4	0	R/W	0: P44/$\overline{\text{IRQ4}}$ is selected 1: P74/$\overline{\text{ExIRQ4}}$ is selected
3	ISS3	0	R/W	0: P43/$\overline{\text{IRQ3}}$ is selected 1: P73/$\overline{\text{ExIRQ3}}$ is selected
2	ISS2	0	R/W	0: P42/$\overline{\text{IRQ2}}$ is selected 1: P72/$\overline{\text{ExIRQ2}}$ is selected
1	ISS1	0	R/W	0: P41/$\overline{\text{IRQ1}}$ is selected 1: P71/$\overline{\text{ExIRQ1}}$ is selected
0	ISS0	0	R/W	0: P40/$\overline{\text{IRQ0}}$ is selected 1: P70/$\overline{\text{ExIRQ0}}$ is selected

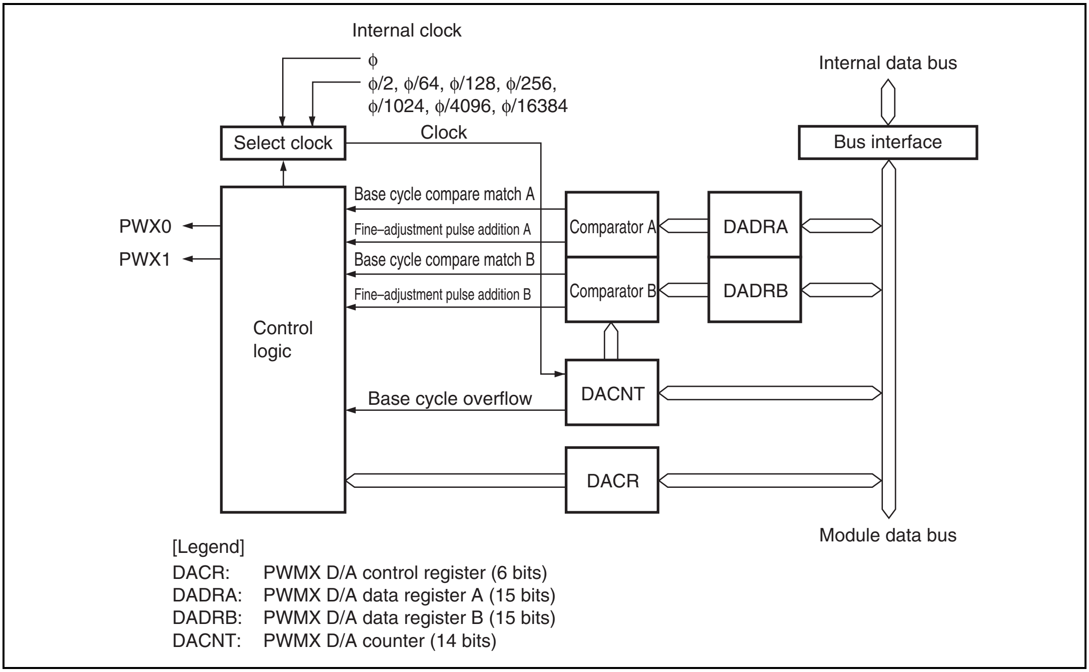
Section 9 14-Bit PWM Timer (PWMX)

This LSI has an on-chip 14-bit pulse-width modulator (PWM) timer with four output channels. It can be connected to an external low-pass filter to operate as a 14-bit D/A converter.

9.1 Features

- Division of pulse into multiple base cycles to reduce ripple
- Eight resolution settings
  The resolution can be set to 1, 2, 64, 128, 256, 1024, 4096, or 16384 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.
- Sixteen operation clocks (by combination of eight resolution settings and two base cycle settings)

Figure 9.1 shows a block diagram of the PWM (D/A) module.



[Legend]
DACR: PWMX D/A control register (6 bits)
DADRA: PWMX D/A data register A (15 bits)
DADRB: PWMX D/A data register B (15 bits)
DACNT: PWMX D/A counter (14 bits)

Figure 9.1 PWMX (D/A) Block Diagram

9.2 Input/Output Pins

Table 9.1 lists the PWMX (D/A) module input and output pins.

Table 9.1 Pin Configuration

Name	Abbreviation	I/O	Function
PWMX output pin 0	PWX0	Output	PWM timer pulse output of PWMX_0 channel A
PWMX output pin 1	PWX1	Output	PWM timer pulse output of PWMX_0 channel B
PWMX output pin 2	PWX2	Output	PWM timer pulse output of PWMX_1 channel A
PWMX output pin 3	PWX3	Output	PWM timer pulse output of PWMX_1 channel B

9.3 Register Descriptions

The PWMX (D/A) module has the following registers. For details on the module stop control register, see section 22.1.3, Module Stop Control Registers H, L, and A (MSTPCRH, MSTPCRL, MSTPCRA).

- PWMX (D/A) counter (DACNT)
- PWMX (D/A) data register A (DADRA)
- PWMX (D/A) data register B (DADRB)
- PWMX (D/A) control register (DACR)
- Peripheral clock select register (PCSR)

Note: The same addresses are shared by DADRA and DACR, and by DADRB and DACNT. Switching is performed by the REGS bit in DACNT or DADRB.

9.3.1 PWMX (D/A) Counter (DACNT)

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 PWMX (D/A) channels. When a channel operates with 14-bit precision, it uses all DACNT bits. When a channel operates with 12-bit precision, it uses the lower 12 bits and ignores the upper two bits. DACNT cannot be accessed in 8-bit units. DACNT should always be accessed in 16-bit units. For details, see section 9.4, Bus Master Interface.

- DACNT

Bit	Bit Name	Initial Value	R/W	Description
15 to 8	UC7 to UC0	All 0	R/W	Lower Up-Counter
7 to 2	UC8 to UC13	All 0	R/W	Upper 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. When changing the register to be accessed, set this bit in advance. 0: DADRA and DADRB can be accessed 1: DACR and DACNT can be accessed

9.3.2 PWMX (D/A) Data Registers A and B (DADRA and DADRB)

DADRA corresponds to PWMX (D/A) channel A, and DADRB to PWMX (D/A) channel B. The DADR registers cannot be accessed in 8-bit units. The DADR registers should always be accessed in 16-bit units. For details, see section 9.4, Bus Master Interface.

- DADRA

Bit	Bit Name	Initial Value	R/W	Description
15 to 2	DA13 to DA0	All 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 fixing DA0 and DA1 to 0. The two data bits are not compared with UC12 and UC13 of DACNT.
1	CFS	1	R/W	Carrier Frequency Select 0: Base cycle = resolution (T) × 64 The range of DA13 to DA0: H'0100 to H'3FFF 1: Base cycle = resolution (T) × 256 The range of DA13 to DA0: H'0040 to H'3FFF
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 to 2	DA13 to DA0	All 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 fixing DA0 and DA1 to 0. The two data bits are not compared with UC12 and UC13 of DACNT.
1	CFS	1	R/W	Carrier Frequency Select 0: Base cycle = resolution (T) × 64 DA13 to DA0 range = H'0100 to H'3FFF 1: Base cycle = resolution (T) × 256 DA13 to DA0 range = H'0040 to H'3FFF
0	REGS	1	R/W	Register Select DADRA and DACR, and DADRB and DACNT, are located at the same addresses. The REGS bit specifies which registers can be accessed. When changing the register to be accessed, set this bit in advance. 0: DADRA and DADRB can be accessed 1: DACR and DACNT can be accessed

9.3.3 PWMX (D/A) Control Register (DACR)

DACR enables the PWM outputs, and selects the output phase and operating speed.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved The initial value should not be changed.
6	PWME	0	R/W	PWMX Enable Starts or stops the PWM D/A counter (DACNT). 0: DACNT operates as a 14-bit up-counter 1: DACNT halts at H'0003
5, 4	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.
3	OEB	0	R/W	Output Enable B Enables or disables output on PWMX (D/A) channel B. 0: PWMX (D/A) channel B output (at the PWX1, PWX3 pins) is disabled 1: PWMX (D/A) channel B output (at the PWX1, PWX3 pins) is enabled
2	OEA	0	R/W	Output Enable A Enables or disables output on PWMX (D/A) channel A. 0: PWMX (D/A) channel A output (at the PWX0, PWX2 pin) is disabled 1: PWMX (D/A) channel A output (at the PWX0, PWX2 pins) is enabled
1	OS	0	R/W	Output Select Selects the phase of the PWMX (D/A) output. 0: Direct PWMX (D/A) output 1: Inverted PWMX (D/A) output
0	CKS	0	R/W	Clock Select Selects the PWMX (D/A) resolution. Eight kinds of resolution 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, × 64, × 128, × 256, × 1024, × 4096, and × 16384.

9.3.4 Peripheral Clock Select Register (PCSR)

PCSR and the CKS bit of DACR select the operating speed.

Bit	Bit Name	Initial Value	R/W	Description
7	PWCKX1B	0	R/W	PWMX_1 Clock Select
6	PWCKX1A	0	R/W	These bits select a clock cycle with the CKS bit of DACR of PWMX_1 being 1. See table 9.2.
5	PWCKX0B	0	R/W	PWMX_0 Clock Select
4	PWCKX0A	0	R/W	These bits select a clock cycle with the CKS bit of DACR of PWMX_0 being 1. See table 9.2.
3	PWCKX1C	0	R/W	PWMX_1 Clock Select This bit selects a clock cycle with the CKS bit of DACR of PWMX_1 being 1. See table 9.2.
2	—	0	R/W	Reserved
1	—	0	R/W	The initial value should not be changed.
0	PWCKX0C	0	R/W	PWMX_0 Clock Select This bit selects a clock cycle with the CKS bit of DACR of PWMX_0 being 1. See table 9.2.

Table 9.2 Clock Select of PWMX_1 and PWMX_0

PWCKX0C PWCKX1C	PWCKX0B PWCKX1B	PWCKX0A PWCKX1A	Resolution (T)
0	0	0	Operates on the system clock cycle (t_{cyc}) x 2
0	0	1	Operates on the system clock cycle (t_{cyc}) x 64
0	1	0	Operates on the system clock cycle (t_{cyc}) x 128
0	1	1	Operates on the system clock cycle (t_{cyc}) x 256
1	0	0	Operates on the system clock cycle (t_{cyc}) x 1024
1	0	1	Operates on the system clock cycle (t_{cyc}) x 4096
1	1	0	Operates on the system clock cycle (t_{cyc}) x 16384
1	1	1	Setting prohibited

9.4 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
```

9.5 Operation

A PWM waveform like the one shown in figure 9.2 is output from the PWX pin. DA13 to DA0 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 DA13 to DA0 in DADR value corresponds to the total width (T_H) of the high (1) output pulses. Figures 9.3 and 9.4 show the types of waveform output available.

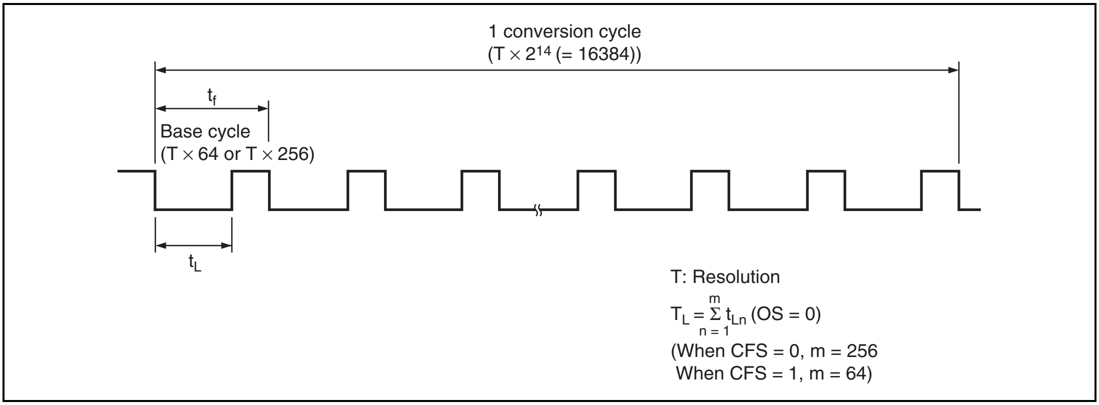


Figure 9.2 PWMX (D/A) Operation

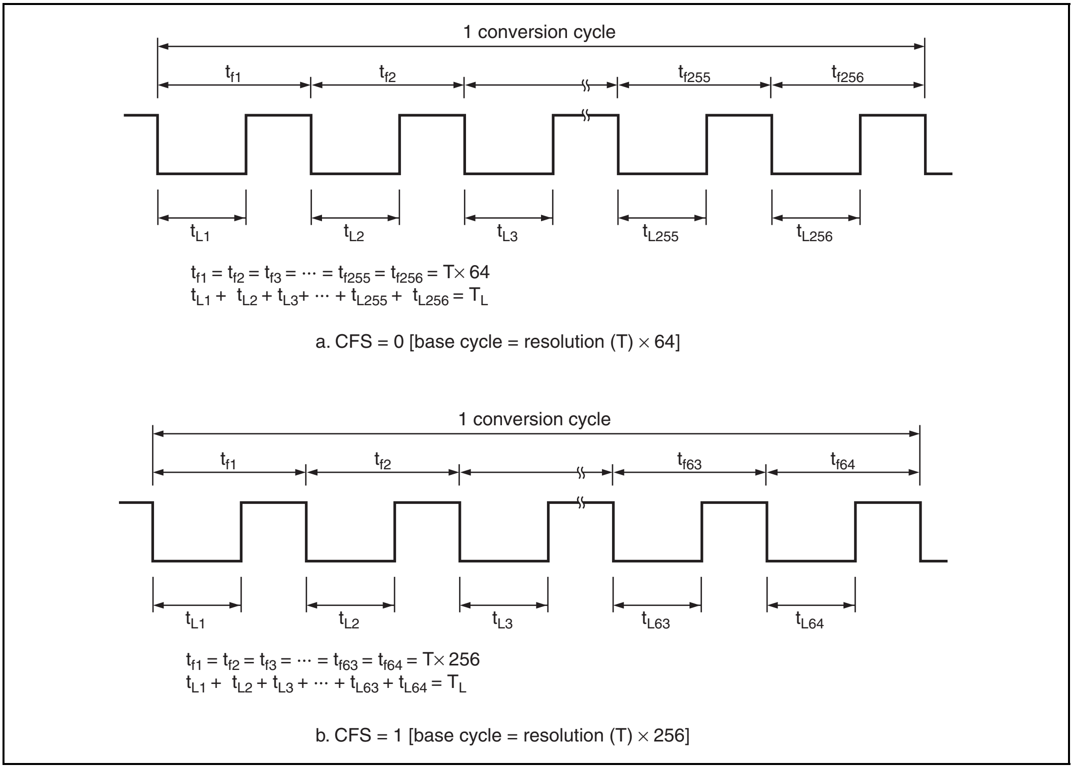
Table 9.3 summarizes the relationships between the CKS and CFS bit settings and the resolution, base cycle, and conversion cycle. The PWM output remains fixed unless DA13 to DA0 in DADR contain at least a certain minimum value. The relationship between the OS bit and the output waveform is shown in figures 9.3 and 9.4.

Table 9.3 Settings and Operation (Examples when ϕ = 25 MHz)

PCSR									Fixed DADR Bits					
PWCKX0 PWCKX1				Reso-lution T		Base	Conver-sion	TL/TH	Accuracy	Bit Data				Conversion
C	B	A	CKS	(µs)	CFS	Cycle	Cycle	(OS = 0/OS = 1)	(Bits)	DA3	DA2	DA1	DA0	Cycle*
—	—	—	0	0.04 (ϕ)	0	2.56 µs/ 390.6 kHz	655.36 µs	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					655.36 µs
									12			0	0	163.84 µs
									10	0	0	0	0	40.96 µs
					1	10.24 µs/ 97.7 kHz	655.36 µs	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					655.36 µs
									12			0	0	163.84 µs
									10	0	0	0	0	40.96 µs
0	0	0	1	0.08 (ϕ/2)	0	5.12 µs/ 195.3 kHz	1.311ms	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					1.311 ms
									12			0	0	0.328 ms
									10	0	0	0	0	0.082 ms
					1	20.48 µs/ 48.8 kHz	1.311 ms	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					1.311 ms
									12			0	0	0.328 ms
									10	0	0	0	0	0.082 ms
0	0	1	1	2.56 (ϕ/64)	0	163.8 µs/ 6.1 kHz	41.943 ms	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					41.943 ms
									12			0	0	10.486 ms
									10	0	0	0	0	2.621 ms
					1	655.4 µs/ 1.5 kHz	41.943 ms	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					41.943 ms
									12			0	0	10.486 ms
									10	0	0	0	0	2.621 ms
0	1	0	1	5.12 (ϕ/128)	0	327.7 µs/ 3.1 kHz	83.886 ms	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					83.886 ms
									12			0	0	20.972 ms
									10	0	0	0	0	5.243 ms
					1	1310.7 µs/ 0.8 kHz	83.886 ms	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					83.886 ms
									12			0	0	20.972 ms
									10	0	0	0	0	5.243 ms

PCSR									Fixed DADR Bits					
PWCKX0 PWCKX1										Bit Data				
C	B	A	CKS	Reso-lution T (µs)	CFS	Base Cycle	Conver-sion Cycle	TL/TH (OS = 0/OS = 1)	Accuracy (Bits)	DA3	DA2	DA1	DA0	Conversion Cycle*
0	1	1	1	10.24 (φ/256)	0	655.4 µs/ 1.5 kHz	167.77 ms	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					167.77 ms
									12			0	0	41.94 ms
									10	0	0	0	0	10.49 ms
					1	2621.4 µs/ 0.4 kHz	167.77 ms	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					167.77 ms
									12			0	0	41.94 ms
									10	0	0	0	0	10.49 ms
1	0	0	1	40.96 (φ/1024)	0	2.62 ms/ 381.5 Hz	671.09 ms	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					671.09 ms
									12			0	0	167.77 ms
									10	0	0	0	0	41.94 ms
					1	10.49 ms/ 95.4 Hz	671.09 ms	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					671.09 ms
									12			0	0	167.77 ms
									10	0	0	0	0	41.94 ms
1	0	1	1	163.84 (φ/4096)	0	10.49 ms/ 95.4 Hz	2.684 s	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					2.684 s
									12			0	0	0.671 s
									10	0	0	0	0	0.168 s
					1	41.94 ms/ 23.8 Hz	2.684 s	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					2.684 s
									12			0	0	0.671 s
									10	0	0	0	0	0.168 s
1	1	0	1	655.36 (φ/16384)	0	41.94 ms/ 23.8 Hz	10.737 s	Always low/high output DA13 to 0 = H'0000 to H'00FF (Data value) × T DA13 to 0 = H'0100 to H'3FFF	14					10.737 s
									12			0	0	2.684 s
									10	0	0	0	0	0.671 s
					1	167.77 ms/ 6.0 Hz	10.737 s	Always low/high output DA13 to 0 = H'0000 to H'003F (Data value) × T DA13 to 0 = H'0040 to H'3FFF	14					10.737 s
									12			0	0	2.684 s
									10	0	0	0	0	0.671 s
1	1	1	1	Setting prohibited	—	—	—	—	—	—	—	—	—	—

Note: * Indicates the conversion cycle when specific DA3 to DA0 bits are fixed.



$t_{f1} = t_{f2} = t_{f3} = \cdots = t_{f255} = t_{f256} = T \times 64$
$t_{L1} + t_{L2} + t_{L3} + \cdots + t_{L255} + t_{L256} = T_L$

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

$t_{f1} = t_{f2} = t_{f3} = \cdots = t_{f63} = t_{f64} = T \times 256$
$t_{L1} + t_{L2} + t_{L3} + \cdots + t_{L63} + t_{L64} = T_L$

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

Figure 9.3 Output Waveform (OS = 0, DADR corresponds to T_L)

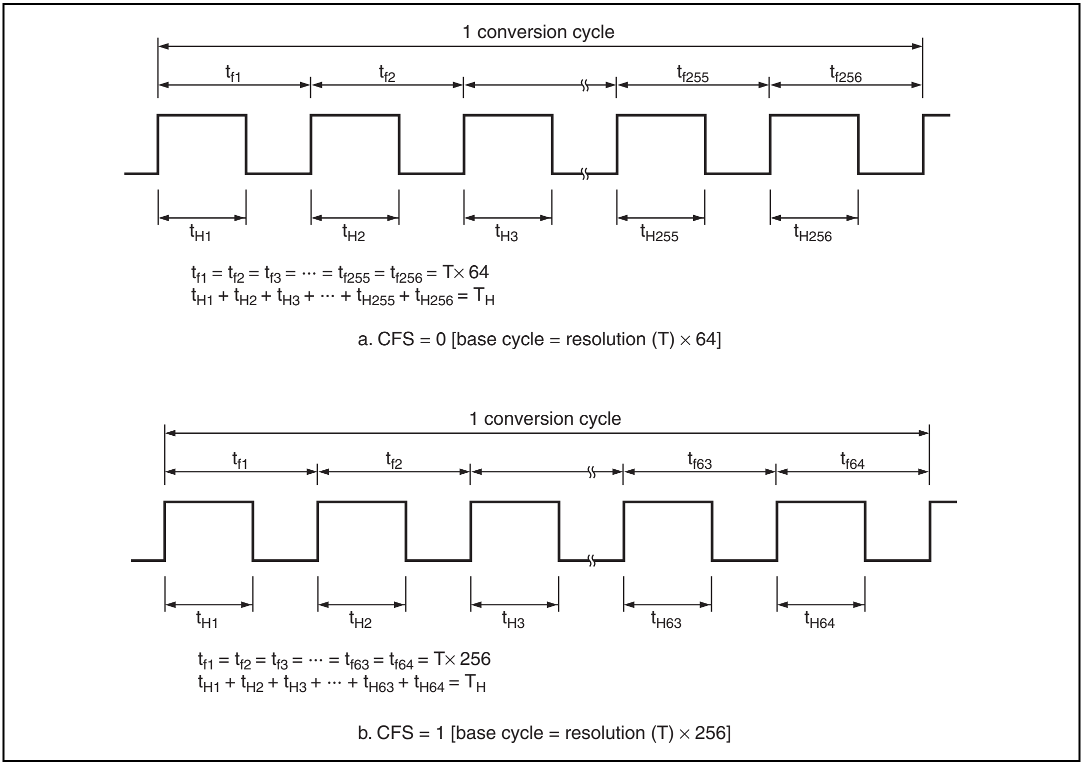
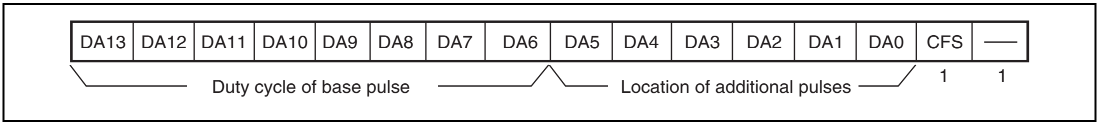

1 conversion cycle
t_{f1}
t_{f2}
t_{f255}
t_{f256}
t_{H1}
t_{H2}
t_{H3}
t_{H255}
t_{H256}
$t_{f1} = t_{f2} = t_{f3} = \cdots = t_{f255} = t_{f256} = T \times 64$
$t_{H1} + t_{H2} + t_{H3} + \cdots + t_{H255} + t_{H256} = T_H$
a. CFS = 0 [base cycle = resolution (T) × 64]
1 conversion cycle
t_{f1}
t_{f2}
t_{f63}
t_{f64}
t_{H1}
t_{H2}
t_{H3}
t_{H63}
t_{H64}
$t_{f1} = t_{f2} = t_{f3} = \cdots = t_{f63} = t_{f64} = T \times 256$
$t_{H1} + t_{H2} + t_{H3} + \cdots + t_{H63} + t_{H64} = T_H$
b. CFS = 1 [base cycle = resolution (T) × 256]

Figure 9.4 Output Waveform (OS = 1, DADR corresponds to T_H)

An example of the additional pulses when CFS = 1 (base cycle = resolution (T) × 256) and OS = 1 (inverted PWM output) is described below. When CFS = 1, the upper eight bits (DA13 to DA6) in DADR determine the duty cycle of the base pulse while the subsequent six bits (DA5 to DA0) determine the locations of the additional pulses as shown in figure 9.5.

Table 9.4 lists the locations of the additional pulses.



DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0	CFS	—
Duty cycle of base pulse								Location of additional pulses						1	1

Figure 9.5 D/A Data Register Configuration when CFS = 1

In this example, DADR = H'0207 (B'0000 0010 0000 0111). The output waveform is shown in figure 9.6. Since CFS = 1 and the value of the upper eight bits is B'0000 0010, the high width of the base pulse duty cycle is 2/256 × (T).

Since the value of the subsequent six bits is B'0000 01, an additional pulse is output only at the location of base pulse No. 63 according to table 9.4. Thus, an additional pulse of 1/256 × (T) is to be added to the base pulse.

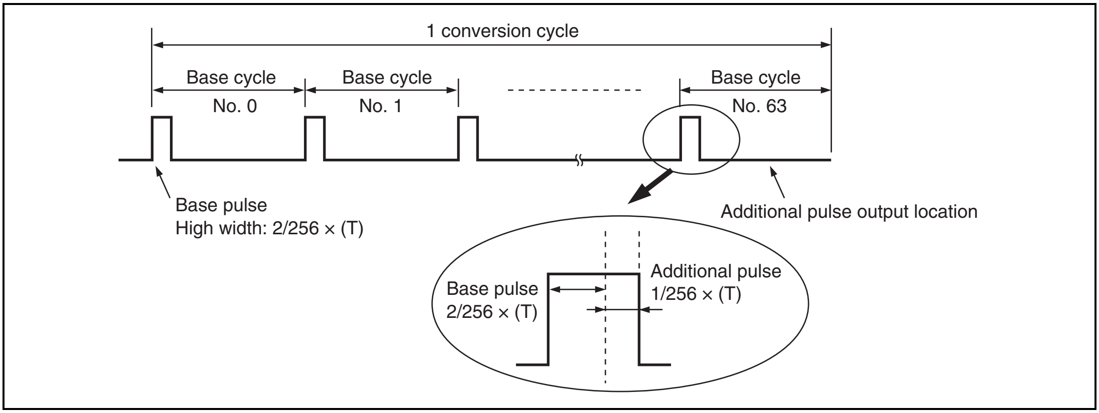


Figure 9.6 Output Waveform when DADR = H'0207 (OS = 1)

However, when CFS = 0 (base cycle = resolution (T) × 64), the duty cycle of the base pulse is determined by the upper six bits and the locations of the additional pulses by the subsequent eight bits with a method similar to as above.

Table 9.4 Locations of Additional Pulses Added to Base Pulse (When CFS = 1)

Lower 6 bits							Base 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
0	0	0	0	0	0	0																																																												
0	0	0	0	0	1	1																																																												○
0	0	0	0	1	0	2																															○																															○
0	0	0	0	1	1	3																															○															○															○
0	0	0	1	0	0	4															○																○															○															○
0	0	0	1	0	1	5															○																○															○								○								○
0	0	0	1	1	0	6															○								○								○															○								○								○
0	0	0	1	1	1	7															○								○								○								○								○								○								○
0	0	1	0	0	0	8								○								○								○								○								○								○								○								○
0	0	1	0	0	1	9								○								○								○								○								○								○								○				○				○
0	0	1	0	1	0	10								○								○								○				○				○								○								○								○				○				○
0	0	1	0	1	1	11								○								○								○				○				○								○				○				○								○				○				○
0	0	1	1	0	0	12								○				○				○								○				○				○								○				○				○								○				○				○
0	0	1	1	0	1	13								○				○				○								○				○				○								○				○				○				○				○				○				○
0	0	1	1	1	0	14								○				○				○				○				○				○				○								○				○				○				○				○				○				○
0	0	1	1	1	1	15								○				○				○				○				○				○				○				○				○				○				○				○				○				○				○
0	1	0	0	0	0	16				○				○				○				○				○				○				○				○				○				○				○				○				○				○				○				○
0	1	0	0	0	1	17				○				○				○				○				○				○				○				○				○				○				○				○				○				○				○		○		○
0	1	0	0	1	0	18				○				○				○				○				○				○				○		○		○				○				○				○				○				○				○		○		○
0	1	0	0	1	1	19				○				○				○				○				○				○				○		○		○				○				○				○		○		○				○				○		○		○
0	1	0	1	0	0	20				○				○				○		○		○				○				○				○		○		○				○				○				○		○		○				○				○		○		○
0	1	0	1	0	1	21				○				○				○		○		○				○				○				○		○		○				○				○				○		○		○				○		○		○				○		○		○
0	1	0	1	1	0	22				○				○				○		○		○				○		○		○				○		○		○				○				○				○		○		○				○		○		○				○		○		○
0	1	0	1	1	1	23				○				○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○
0	1	1	0	0	0	24				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○
0	1	1	0	0	1	25				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○				○		○		○		○		○		○		○
0	1	1	0	1	0	26				○		○		○				○		○		○				○		○		○		○		○		○		○				○		○		○				○		○		○				○		○		○		○		○		○		○
0	1	1	0	1	1	27				○		○		○				○		○		○				○		○		○		○		○		○		○				○		○		○		○		○		○		○				○		○		○		○		○		○		○
0	1	1	1	0	0	28				○		○		○		○		○		○		○				○		○		○		○		○		○		○				○		○		○		○		○		○		○				○		○		○		○		○		○		○
0	1	1	1	0	1	29				○		○		○		○		○		○		○				○		○		○		○		○		○		○				○		○		○		○		○		○		○		○		○		○		○		○		○		○		○
0	1	1	1	1	0	30				○		○		○		○		○		○		○		○		○		○		○		○		○		○		○				○		○		○		○		○		○		○		○		○		○		○		○		○		○		○
0	1	1	1	1	1	31				○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○
1	0	0	0	0	0	32		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○
1	0	0	0	0	1	33		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○	○	○
1	0	0	0	1	0	34		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○	○	○		○		○		○		○		○		○		○		○		○		○		○		○		○		○	○	○
1	0	0	0	1	1	35		○		○		○		○		○		○		○		○		○		○		○		○		○		○		○	○	○		○		○		○		○		○		○		○	○	○		○		○		○		○		○		○	○	○
1	0	0	1	0	0	36		○		○		○		○		○		○		○	○	○		○		○		○		○		○	○	○		○		○		○		○		○		○		○	○	○		○		○		○		○		○		○	○	○
1	0	0	1	0	1	37		○		○		○		○		○		○		○	○	○		○		○		○		○		○	○	○		○		○		○		○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○
1	0	0	1	1	0	38		○		○		○		○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○		○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○
1	0	0	1	1	1	39		○		○		○		○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○
1	0	1	0	0	0	40		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○
1	0	1	0	0	1	41		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○	○	○	○	○
1	0	1	0	1	0	42		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○	○	○	○	○
1	0	1	0	1	1	43		○		○		○	○	○		○		○		○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○
1	0	1	1	0	0	44		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○
1	0	1	1	0	1	45		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○
1	0	1	1	1	0	46		○		○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○
1	0	1	1	1	1	47		○		○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○
1	1	0	0	0	0	48		○	○	○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○
1	1	0	0	0	1	49		○	○	○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○
1	1	0	0	1	0	50		○	○	○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○
1	1	0	0	1	1	51		○	○	○		○	○	○		○		○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○
1	1	0	1	0	0	52		○	○	○		○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○
1	1	0	1	0	1	53		○	○	○		○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○
1	1	0	1	1	0	54		○	○	○		○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○		○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○
1	1	0	1	1	1	55		○	○	○		○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○
1	1	1	0	0	0	56		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○
1	1	1	0	0	1	57		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
1	1	1	0	1	0	58		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
1	1	1	0	1	1	59		○	○	○	○	○	○	○		○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
1	1	1	1	0	0	60		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
1	1	1	1	0	1	61		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
1	1	1	1	1	0	62		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
1	1	1	1	1	1	63		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

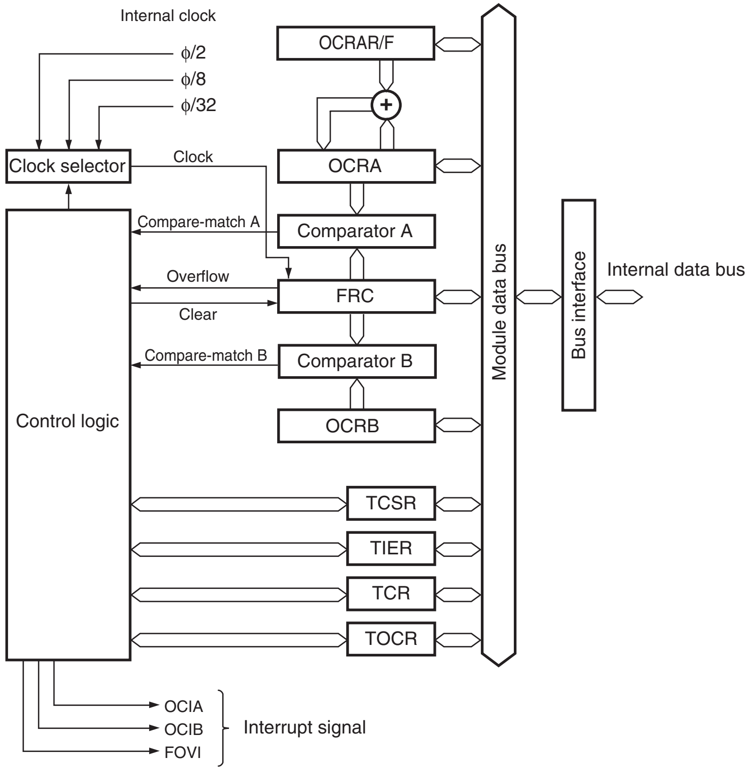
Section 10 16-Bit Free-Running Timer (FRT)

This LSI has an on-chip 16-bit free-running timer (FRT).

10.1 Features

- Selection of four clock sources
  - One of the three internal clocks (ϕ/2, ϕ/8, or ϕ/32) can be selected.
- Two independent comparators
- Counter clearing
  - The free-running counters can be cleared on compare-match A.
- Three independent interrupts
  - Two compare-match interrupts and one overflow interrupt can be requested independently.

Figure 10.1 shows a block diagram of the FRT.



[Legend]

OCRA, OCRB:	Output compare registers A and B (16 bits)
OCRAR,OCRAF:	Output compare registers AR and AF (16 bits)
FRC:	Free-running counter (16 bits)
TCSR:	Timer control/status register (8 bits)
TIER:	Timer interrupt enable register (8 bits)
TCR:	Timer control register (8 bits)
TOCR:	Timer output compare control register (8 bits)

Figure 10.1 Block Diagram of 16-Bit Free-Running Timer

10.2 Register Descriptions

The FRT has the following registers.

- Free-running counter (FRC)
- Output compare register A (OCRA)
- Output compare register B (OCRB)
- Output compare register AR (OCRAR)
- Output compare register AF (OCRAF)
- 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.

10.2.1 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.

10.2.2 Output Compare Registers A and B (OCRA and OCRB)

The FRT has two output compare registers, OCRA and OCRB, each of which is a 16-bit readable/writable register whose contents are continually compared with the value in FRC. When a match is detected (compare-match), the corresponding output compare flag (OCFA or OCFB) is set to 1 in TCSR. OCR should always be accessed in 16-bit units; cannot be accessed in 8-bit units. OCR is initialized to H'FFFF.

10.2.3 Output Compare Registers AR and AF (OCRAR and OCRAF)

OCRAR and OCRAF are 16-bit readable/writable registers. They are accessed when the ICRS bit in TOCR is set to 1. 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.

When using the OCRA automatic addition function, do not select internal clock ϕ/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.

10.2.4 Timer Interrupt Enable Register (TIER)

TIER enables and disables interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
3	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
2	OCIBE	0	R/W	Output Compare Interrupt B Enable Selects whether to enable output compare interrupt B request (OCIB) when output compare flag B (OCFB) in TCSR is set to 1. 0: OCIB requested by OCFB is disabled 1: OCIB requested by OCFB is enabled
1	OVIE	0	R/W	Timer Overflow Interrupt Enable Selects whether to enable a free-running timer overflow request interrupt (FOVI) when the timer overflow flag (OVF) in TCSR is set to 1. 0: FOVI requested by OVF is disabled 1: FOVI requested by OVF is enabled
0	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

10.2.5 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 to 4	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
3	OCFA	0	R/(W)*	Output Compare Flag A This status flag indicates that the FRC value matches the OCRA value. [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. [Setting condition] When FRC = OCRB [Clearing condition] Read OCFB when OCFB = 1, then write 0 to OCFB
1	OVF	0	R/(W)*	Overflow Flag This status flag indicates that the FRC has overflowed. [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.

10.2.6 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 to 2	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
1	CKS1	0	R/W	Clock Select 1 and 0
0	CKS0	0	R/W	Select clock source for FRC. 00: ϕ/2 internal clock source 01: ϕ/8 internal clock source 10: ϕ/32 internal clock source 11: Reserved

10.2.7 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, and controls the OCRA operating modes.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
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 Controls the access to OCRAR and OCRAF. 0: Access is disabled. 1: Access is enabled.
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 to 0	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.

10.3 Operation Timing

10.3.1 FRC Increment Timing

Figure 10.2 shows the FRC increment timing with an internal clock source.

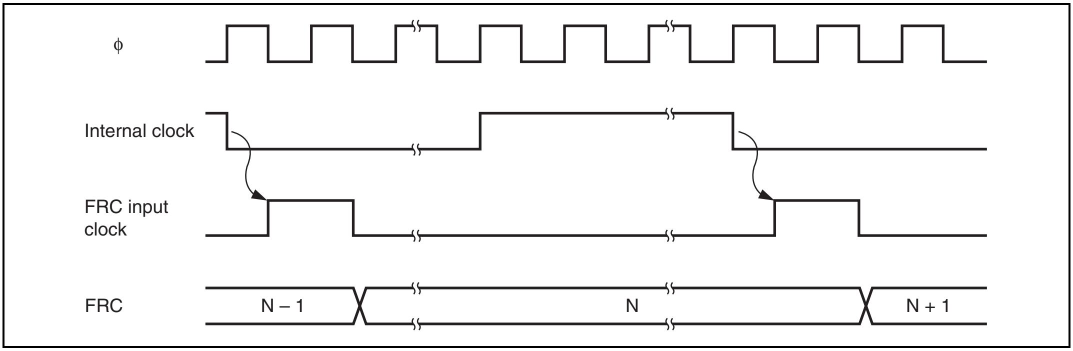


Figure 10.2 Increment Timing with Internal Clock Source

10.3.2 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 10.3 shows the timing of this operation for compare-match A.

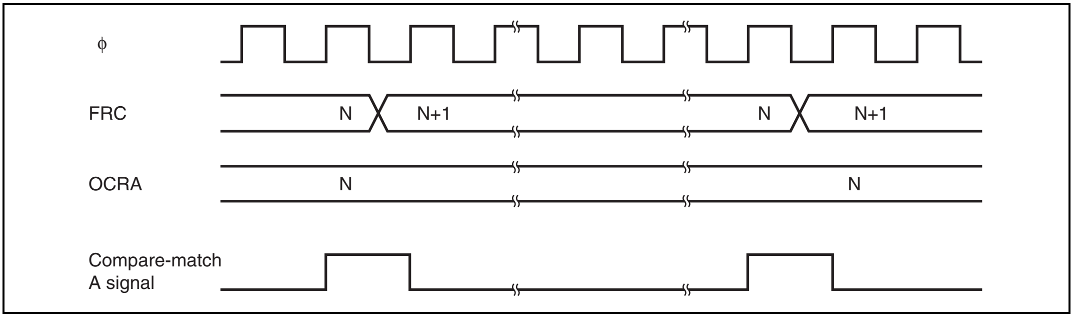


Figure 10.3 Timing of Output Compare A Output

10.3.3 FRC Clear Timing

FRC can be cleared when compare-match A occurs. Figure 10.4 shows the timing of this operation.

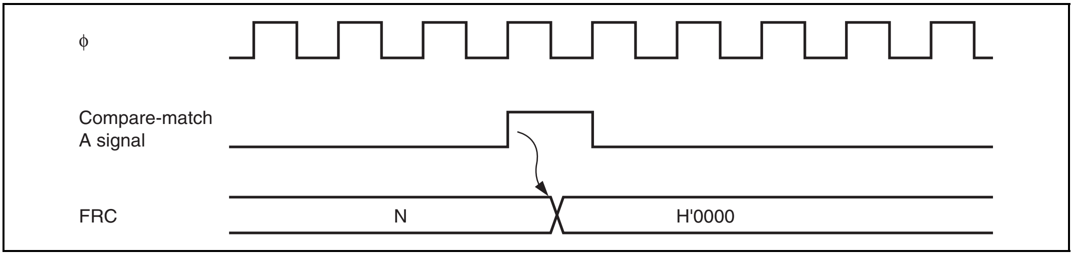


Figure 10.4 Clearing of FRC by Compare-Match A Signal

10.3.4 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 10.5 shows the timing of setting the OCFA or OCFB flag.

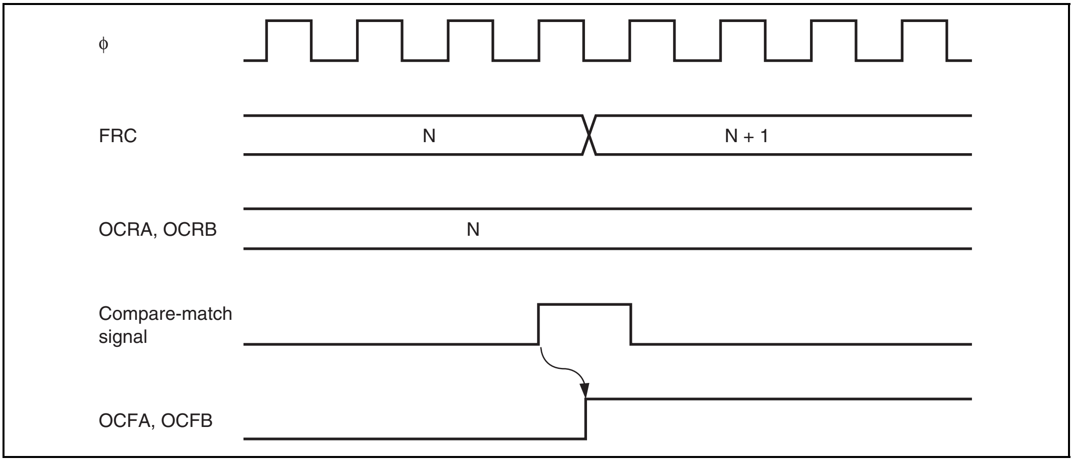


Figure 10.5 Timing of Output Compare Flag (OCFA or OCFB) Setting

10.3.5 Timing of FRC Overflow Flag (OVF) Setting

The FRC overflow flag (OVF) is set to 1 when FRC overflows (changes from H'FFFF to H'0000). Figure 10.6 shows the timing of setting the OVF flag.

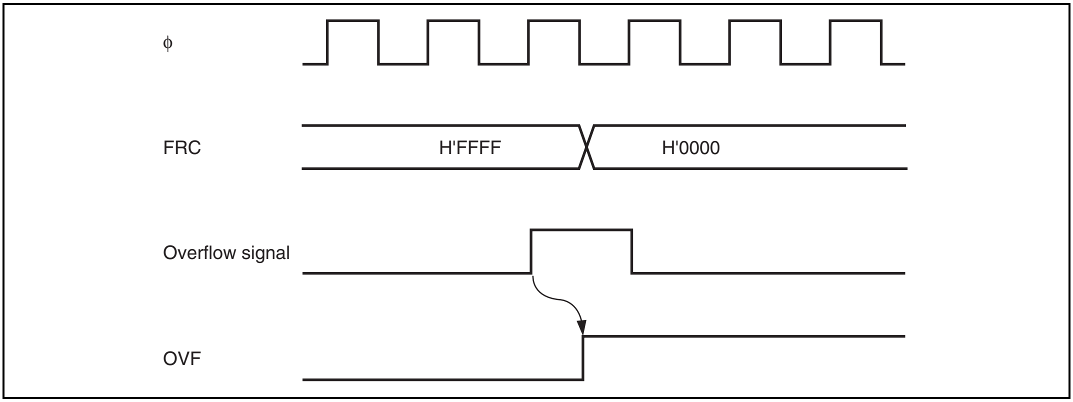


Figure 10.6 Timing of Overflow Flag (OVF) Setting

10.3.6 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 10.7 shows the OCRA write timing.

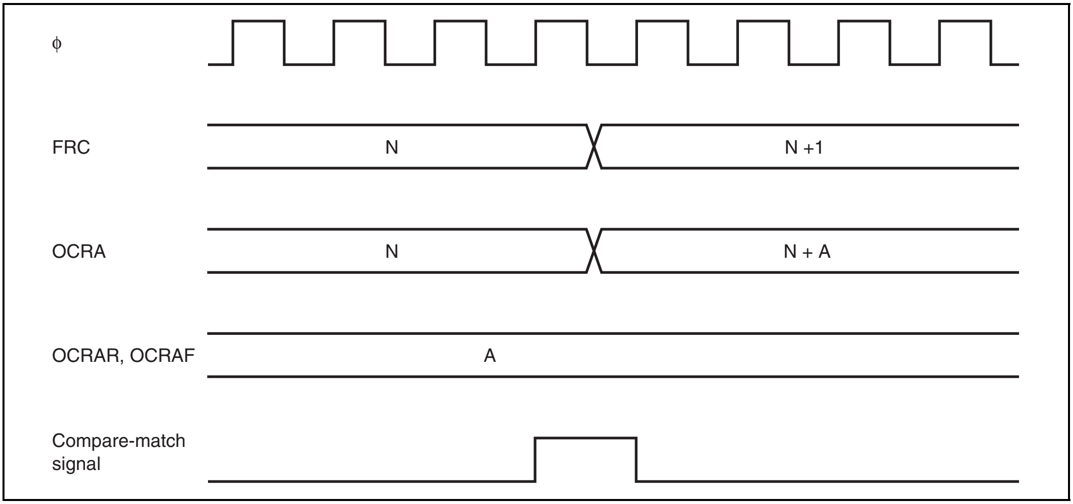


Figure 10.7 OCRA Automatic Addition Timing

10.4 Interrupt Sources

The free-running timer can request three interrupts: 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 10.1 lists the sources and priorities of these interrupts.

The OCIA and OCIB interrupts can be used as the on-chip DTC activation sources.

Table 10.1 FRT Interrupt Sources

Interrupt	Interrupt Source	Interrupt Flag	DTC Activation	Priority
OCIA	Compare match of OCRA	OCFA	Possible	High
OCIB	Compare match of OCRB	OCFB	Possible	↑
FOVI	Overflow of FRC	OVF	Not possible	Low

10.5 Usage Notes

10.5.1 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 10.8 shows the timing for this type of conflict.

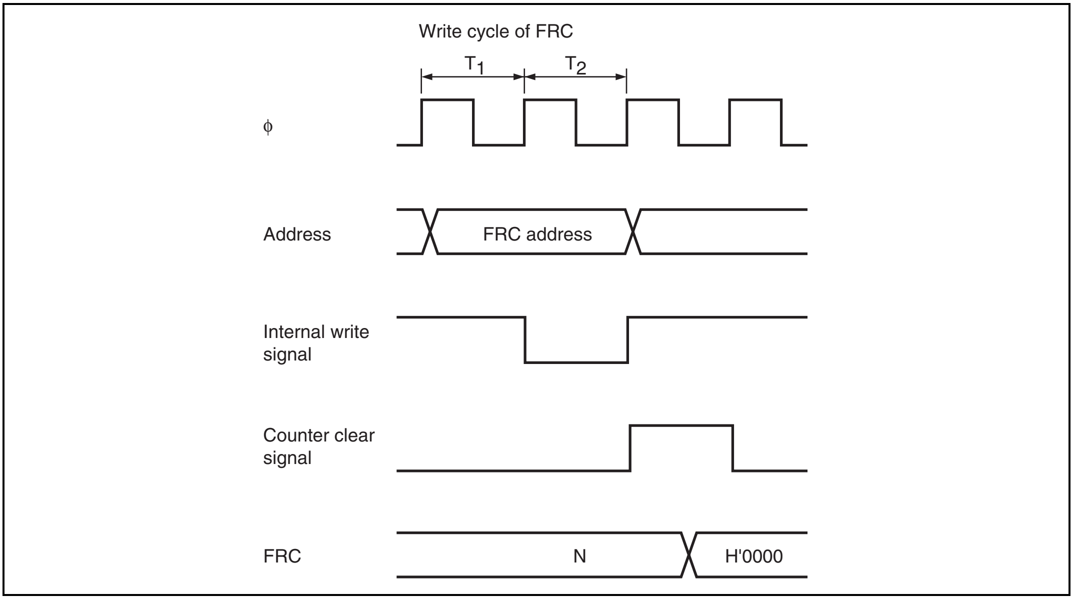


Figure 10.8 Conflict between FRC Write and Clear

10.5.2 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 10.9 shows the timing for this type of conflict.

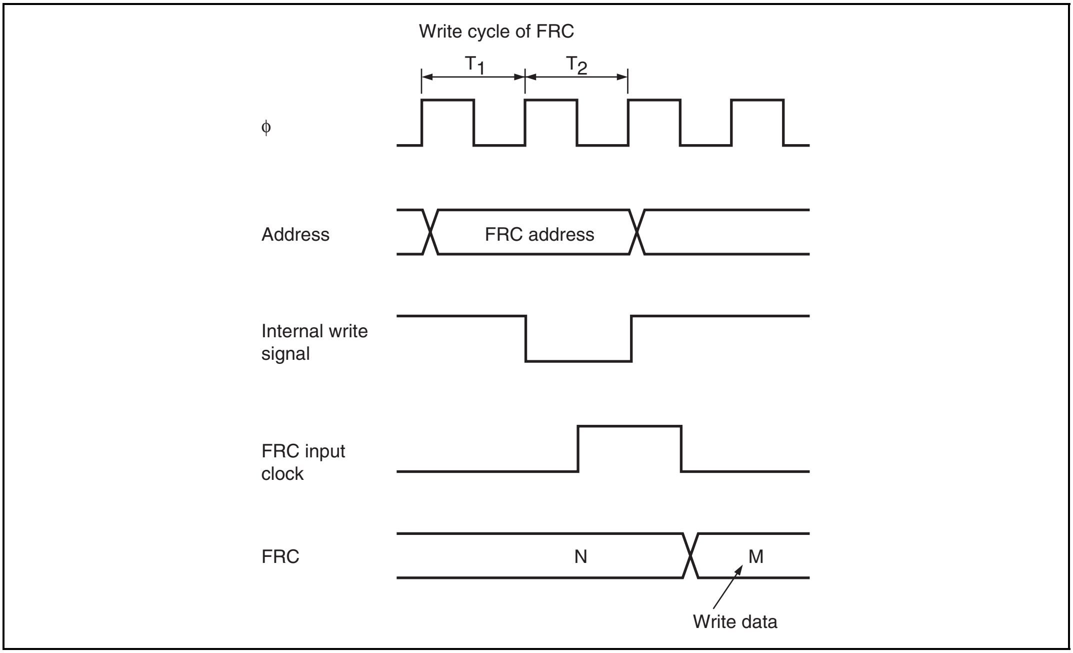


Figure 10.9 Conflict between FRC Write and Increment

10.5.3 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 10.10 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 10.11 shows the timing for this type of conflict.

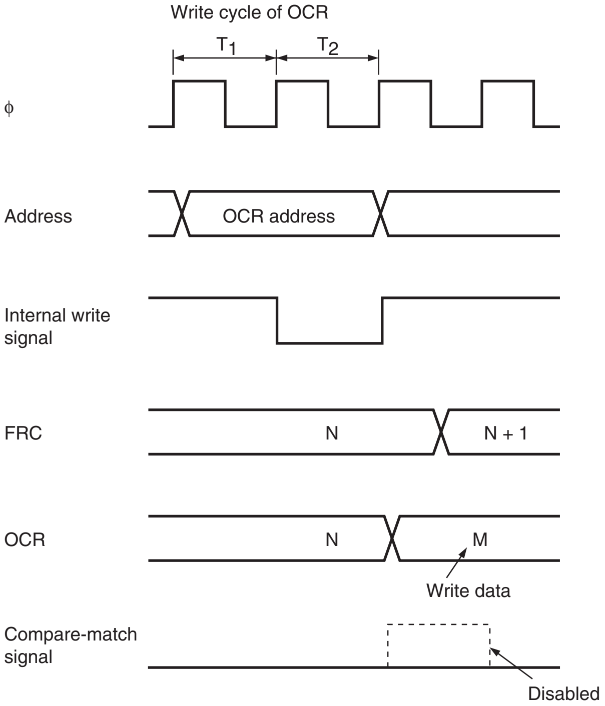


Figure 10.10 Conflict between OCR Write and Compare-Match (When Automatic Addition Function is Not Used)

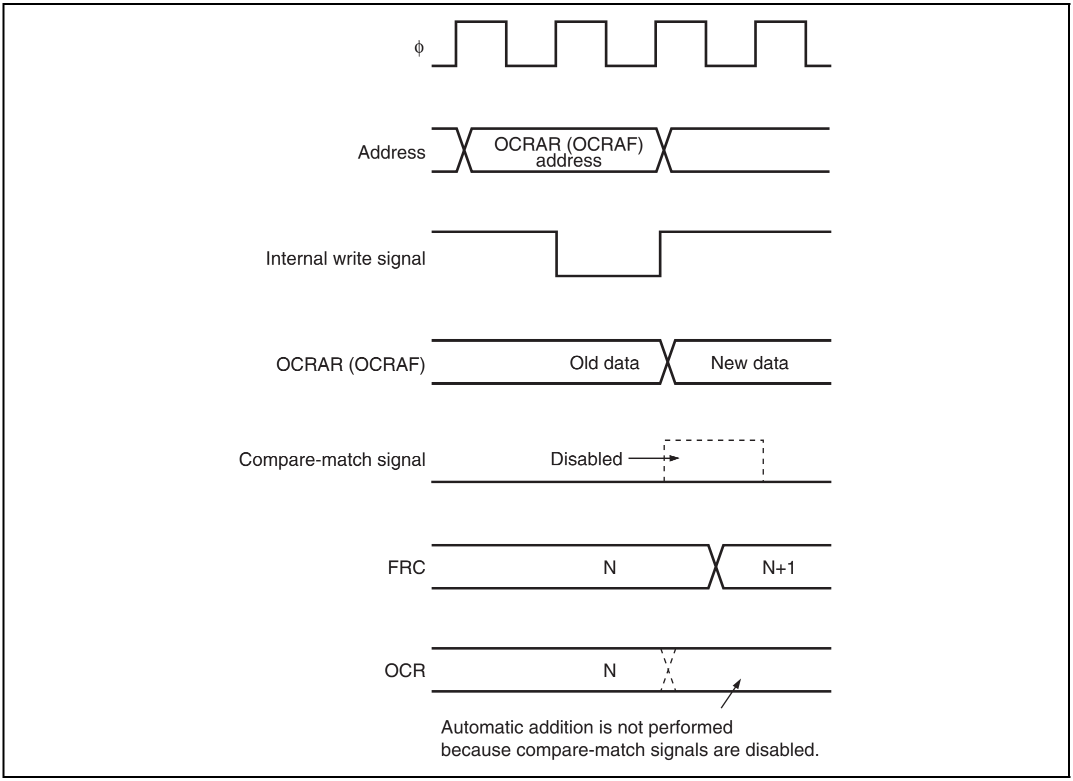


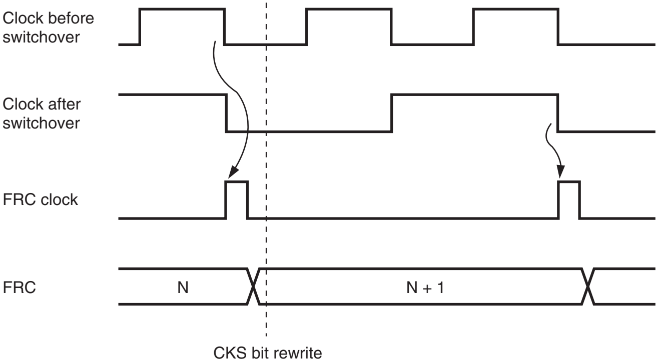
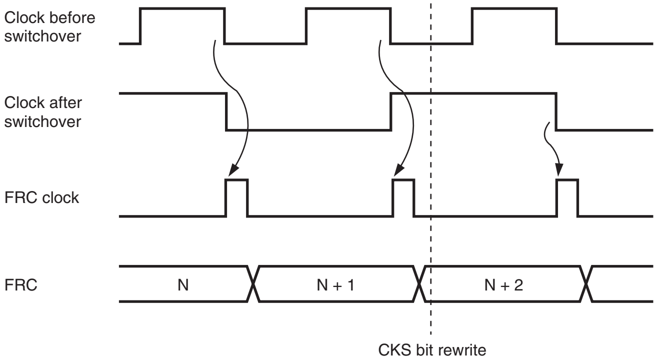
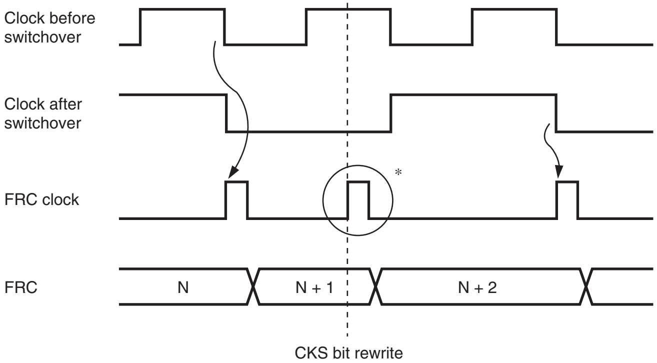
Figure 10.11 Conflict between OCR Write and Compare-Match (When Automatic Addition Function is Used)

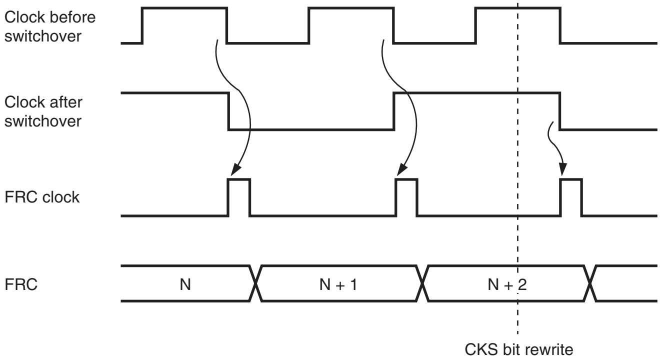
10.5.4 Switching of Internal Clock and FRC Operation

When the internal clock is changed, the changeover may source FRC to increment. This depends on the time at which the clock is switched (bits CKS1 and CKS0 are rewritten), as shown in table 10.2.

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 10.2, 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 source FRC to increment.

Table 10.2 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
3	Switching from high to low	 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
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.

Section 11 8-Bit Timer (TMR)

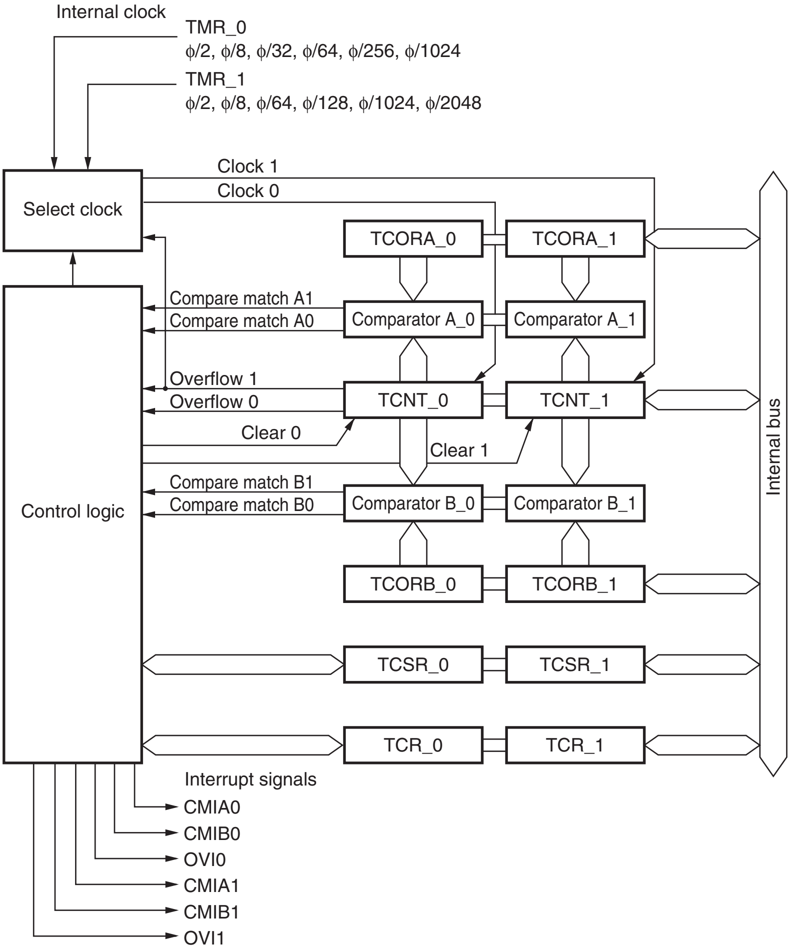
This LSI has an on-chip 8-bit timer module (TMR_0 and TMR_1) with two channels operating on the basis of an 8-bit counter.

This LSI also has two channels of similar 8-bit timer modules (TMR_Y and TMR_X).

11.1 Features

- Selection of clock sources
  - TMR_0, TMR_1: The counter input clock can be selected from six internal clocks.
  - TMR_Y, TMR_X: The counter input clock can be selected from three internal clocks.
- Selection of two ways to clear the counters
  - The counters can be cleared on compare-match A and compare-match B.
- Cascading of TMR_0 and TMR_1
  (Cascading of TMR_Y and TMR_X is not allowed)
  - Operation as a 16-bit timer can be performed using TMR_0 as the upper half and TMR_1 as the lower half (16-bit count mode). TMR_1 can be used to count TMR_0 compare match occurrences (compare-match count mode).
- Multiple interrupt sources for each channel
  - TMR_0, TMR_1,
    and TMR_Y: One interrupt: Overflow
  - TMR_X: Three interrupts: Compare-match A, compare-match B, and overflow

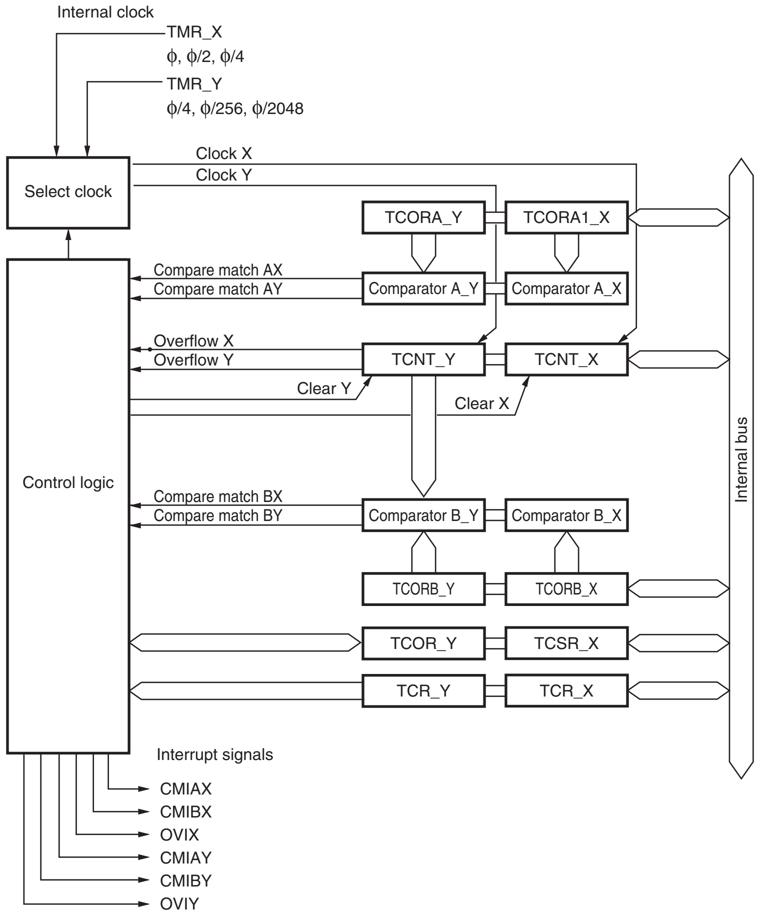
Figures 11.1 and 11.2 show block diagrams of 8-bit timers.



[Legend]

TCORA_0:	Time constant register A_0	TCORA_1:	Time constant register A_1
TCORB_0:	Time constant register B_0	TCORB_1:	Time constant register B_1
TCNT_0:	Timer counter_0	TCNT_1:	Timer counter_1
TCSR_0:	Timer control/status register_0	TCSR_1:	Timer control/status register_1
TCR_0:	Timer control register_0	TCR_1:	Timer control register_1

Figure 11.1 Block Diagram of 8-Bit Timer (TMR_0 and TMR_1)



[Legend]

TCORA_Y:	Time constant register A_Y	TCORA_X:	Time constant register A_X
TCORB_Y:	Time constant register B_Y	TCORB_X:	Time constant register B_X
TCNT_Y:	Timer counter_Y	TCNT_X:	Timer counter_X
TCSR_Y:	Timer control / status register_Y	TCSR_X:	Timer control / status register_X
TCR_Y:	Timer control register_Y	TCR_X:	Timer control register_X

Figure 11.2 Block Diagram of 8-Bit Timer (TMR_Y and TMR_X)

11.2 Register Descriptions

The TMR has the following registers for each channel. For details on the serial timer control register, see section 3.2.3, Serial Timer Control Register (STCR).

- Timer counter (TCNT)
- Time constant register A (TCORA)
- Time constant register B (TCORB)
- Timer control register (TCR)
- Timer control/status register (TCSR)
- Timer connection register S (TCONRS)*

Notes: 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.

* Only for the TMR_X

11.2.1 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 a 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 TMRX/Y bit in TCONRS is 1. TCNT_X can be accessed when the TMRX/Y bit in TCONRS is 0. See section 11.2.6, Timer Connection Register S (TCONRS).

11.2.2 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. TCORA is initialized to H'FF.

TCORA_Y can be accessed when the TMRX/Y bit in TCONRS is 1. TCORA_X can be accessed when the TMRX/Y bit in TCONRS is 0. See section 11.2.6, Timer Connection Register S (TCONRS).

11.2.3 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. TCORB is initialized to H'FF.

TCORB_Y can be accessed when the TMRX/Y bit in TCONRS is 1. TCORB_X can be accessed when the TMRX/Y bit in TCONRS is 0. See section 11.2.6, Timer Connection Register S (TCONRS).

11.2.4 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 TMRX/Y bit in TCONRS is 1. TCR_X can be accessed when the TMRX/Y bit in TCONRS is 0. See section 11.2.6, Timer Connection Register S (TCONRS).

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. 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. 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. 0: OVF interrupt request (OVI) is disabled 1: OVF interrupt request (OVI) is enabled
4	CCLR1	0	R/W	Counter Clear 1 and 0
3	CCLR0	0	R/W	Specify the cleaning conditions of TCNT. 00: Counter clear is disabled. 01: Counter clear is enabled on compare-match A 10: Counter clear is enabled on compare-match B 11: Setting prohibited
2 to 0	CKS2 to CKS0	All 0	R/W	Clock Select 2 to 0 These bits select the clock input to TCNT and count condition, together with the ICKS1 and ICKS0 bits in STCR. For details, see table 11.1.

Table 11.1 (1) Clock Input to TCNT and Count Condition (TMR_0)

	TCR		STCR	
CKS2	**CKS1**	**CKS0**	**ICKS0**	**Description**
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

Table 11.1 (2) Clock Input to TCNT and Count Condition (TMR_1)

	TCR		STCR	
CKS2	**CKS1**	**CKS0**	**ICKS1**	**Description**
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

Table 11.1 (3) Clock Input to TCNT and Count Condition (TMR_Y, TMR_X, Common)

	TCR			
Channel	CKS2	CKS1	CKS0	Description
TMR_Y	0	0	0	Disables clock input
	0	0	1	Increments at falling edge of internal clock φ/4
	0	1	0	Increments at falling edge of internal clock φ/256
	0	1	1	Increments at falling edge of internal clock φ/2048
	1	0	0	Setting prohibited
TMR_X	0	0	0	Disables clock input
	0	0	1	Increments at falling edge of internal clock φ
	0	1	0	Increments at falling edge of internal clock φ/2
	0	1	1	Increments at falling edge of internal clock φ/4
	1	0	0	Setting prohibited
Common	1	0	1	Setting prohibited
	1	1	0	Setting prohibited
	1	1	1	Setting prohibited

11.2.5 Timer Control/Status Register (TCSR)

TCSR indicates the status flags and controls compare-match output. About the TCSR_Y and TCSR_X accesses see section 11.2.6, Timer Connection Register S (TCONRS).

- 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 Selects whether the A/D conversion start request on compare match A is enabled or disabled. 0: A/D conversion start request is disabled 1: A/D conversion start request is enabled
3 to 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

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

- 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 to 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

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

- TCSR_Y

  This register can be accessed when the TMRX/Y bit in TCONRS is 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_Y and TCORB_Y 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_Y and TCORA_Y 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_Y overflows from H'FF to H'00 [Clearing condition] Read OVF when OVF = 1, then write 0 in OVF
4 to 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

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

- TCSR_X

  This register can be accessed when the TMRX/Y bit in TCONRS is 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_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
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 to 0	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.

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

11.2.6 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	TMRX/Y	0	R/W	TMR_X/TMR_Y Access Select For details, see table 11.2. 0: The TMR_X registers are accessed at addresses H'FFFFF0 to H'FFFFF5 1: The TMR_Y registers are accessed at addresses H'FFFFF0 to H'FFFFF5
6 to 0	—	All 0	R/W	Reserved The initial values should not be changed.

Table 11.2 Registers Accessible by TMR_X/TMR_Y

TMRX/Y	H'FFFFF0	H'FFFFF1	H'FFFFF2	H'FFFFF3	H'FFFFF4	H'FFFFF5	H'FFFFF6	H'FFFFF7
0	TMR_X TCR_X	TMR_X TCSR_X	TMR_X	TMR_X	TMR_X TCNT_X	TMR_X	TMR_X TCORA_X	TMR_X TCORB_X
1	TMR_Y TCR_Y	TMR_Y TCSR_Y	TMR_Y TCORA_Y	TMR_Y TCORB_Y	TMR_Y TCNT_Y	TMR_Y		

11.3 Operation Timing

11.3.1 TCNT Count Timing

Figure 11.3 shows the TCNT count timing with an internal clock source.

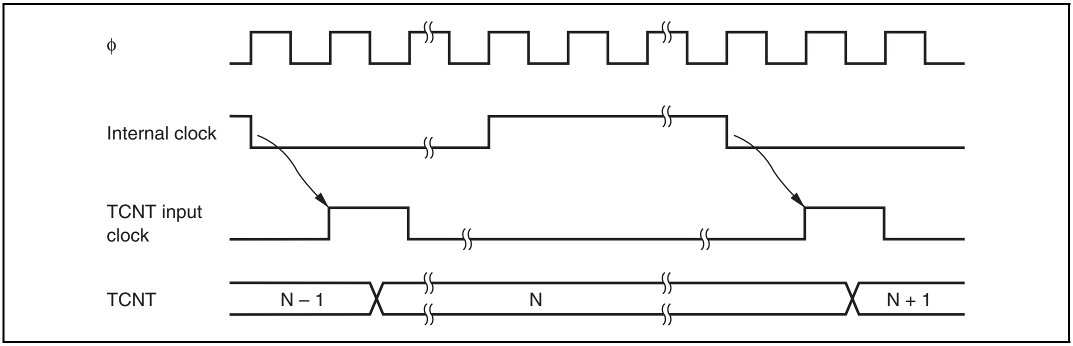


Figure 11.3 Count Timing for Internal Clock Input

11.3.2 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 11.4 shows the timing of CMF flag setting.

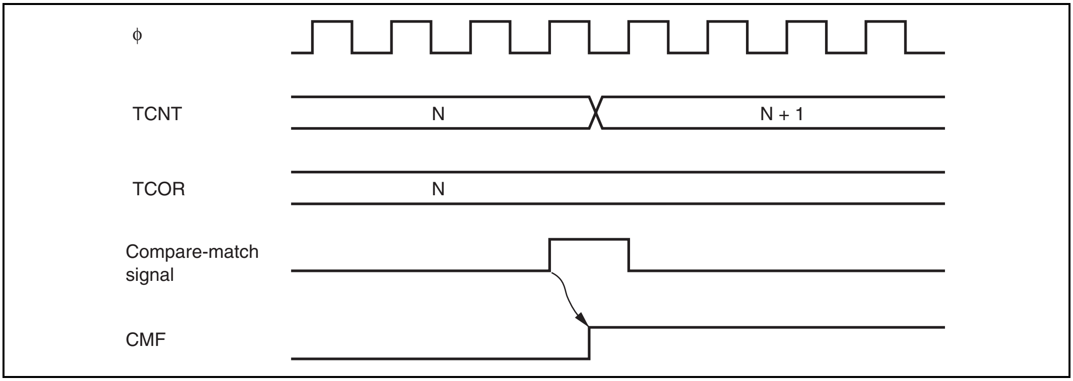


Figure 11.4 Timing of CMF Setting at Compare-Match

11.3.3 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 11.5 shows the timing of clearing the counter by a compare-match.

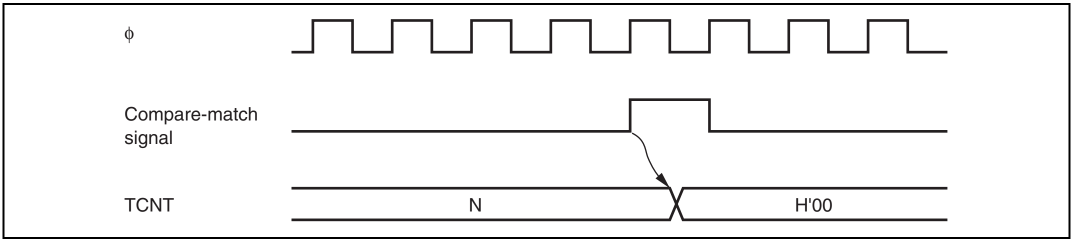


Figure 11.5 Timing of Counter Clear by Compare-Match

11.3.4 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 11.6 shows the timing of OVF flag setting.

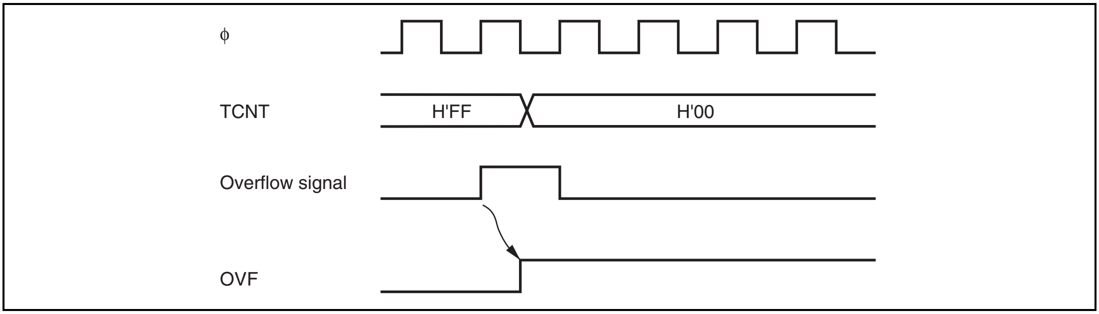


Figure 11.6 Timing of OVF Flag Setting

11.4 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, 16-bit count mode or compare-match count mode can be selected.

11.4.1 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 eight bits and TMR_1 occupying the lower eight 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.

11.4.2 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, and counter clearing are in accordance with the settings for each channel.

11.5 Interrupt Sources

TMR_0, TMR_1, TMR_Y and TMR_X can generate three types of interrupts: CMIA, CMIB, and OVI.

Table 11.3 shows the interrupt sources and priorities. Each interrupt source can be enabled or disabled independently by interrupt enable bits in TCR or TCSR. Independent signals are sent to the interrupt controller for each interrupt.

The CMIA and CMIB interrupts can be used as DTC activation interrupt sources.

Table 11.3 Interrupt Sources of 8-Bit Timers TMR_0, TMR_1, TMR_Y, and TMR_X

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	Interrupt Priority
TMR_X	CMIAX	TCORA_X compare-match	CMFA	Possible	High
	CMIBX	TCORB_X compare-match	CMFB	Possible	↑
	OVIX	TCNT_X overflow	OVF	Not possible	
TMR_0	CMIA0	TCORA_0 compare-match	CMFA	Possible	
	CMIB0	TCORB_0 compare-match	CMFB	Possible	
	OVI0	TCNT_0 overflow	OVF	Not possible	
TMR_1	CMIA1	TCORA_1 compare-match	CMFA	Possible	
	CMIB1	TCORB_1 compare-match	CMFB	Possible	
	OVI1	TCNT_1 overflow	OVF	Not possible	
TMR_Y	CMIAY	TCORA_Y compare-match	CMFA	Possible	
	CMIBY	TCORB_Y compare-match	CMFB	Possible	
	OVIY	TCNT_Y overflow	OVF	Not possible	Low

11.6 Usage Notes

11.6.1 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 11.7, the counter clear takes priority and the write is not performed.

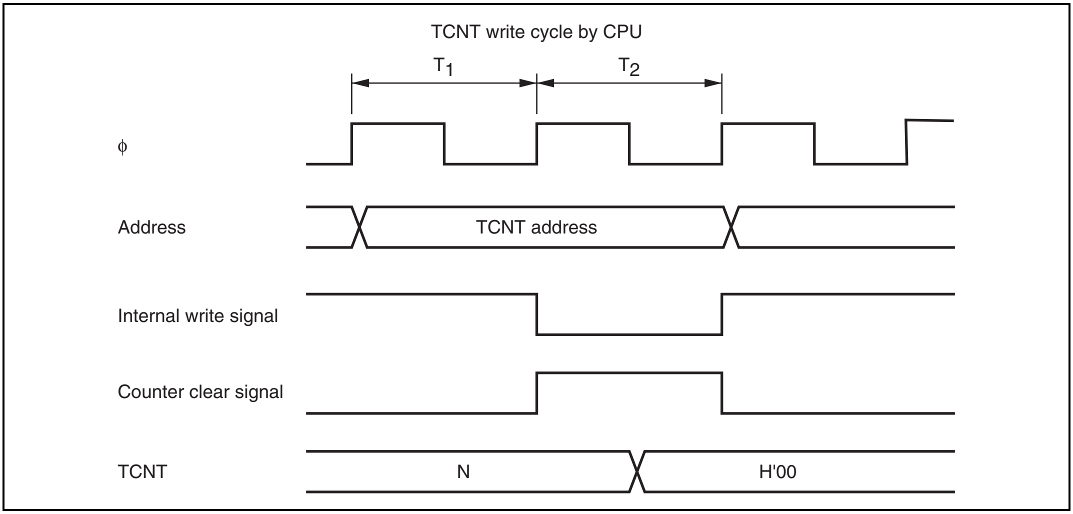


Figure 11.7 Conflict between TCNT Write and Counter Clear

11.6.2 Conflict between TCNT Write and Increment

If a TCNT input clock is generated during the T_2 state of a TCNT write cycle as shown in figure 11.8, the write takes priority and the counter is not incremented.

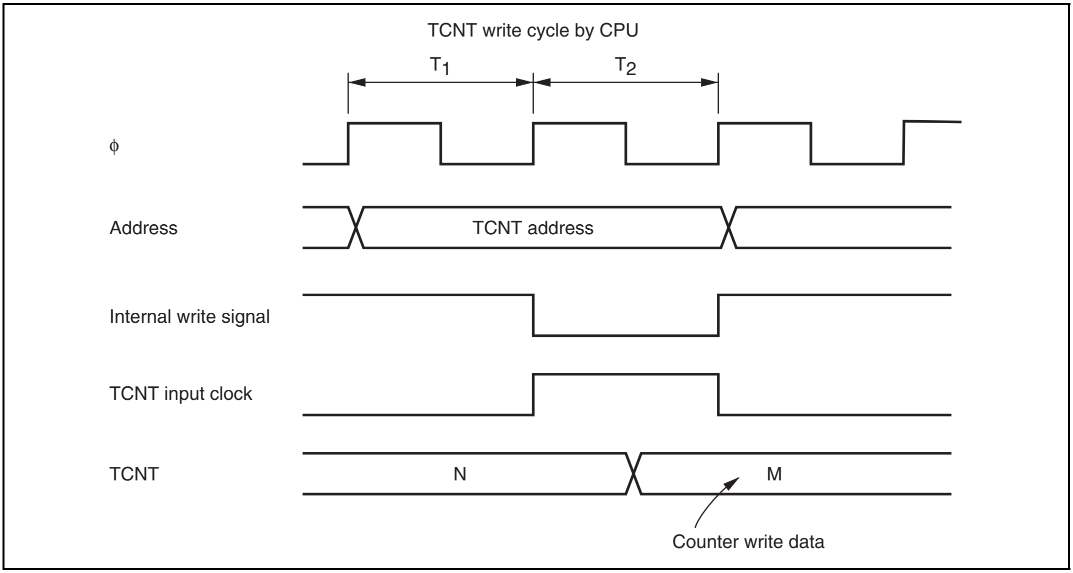


Figure 11.8 Conflict between TCNT Write and Increment

11.6.3 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 11.9, the TCOR write takes priority and the compare-match signal is disabled.

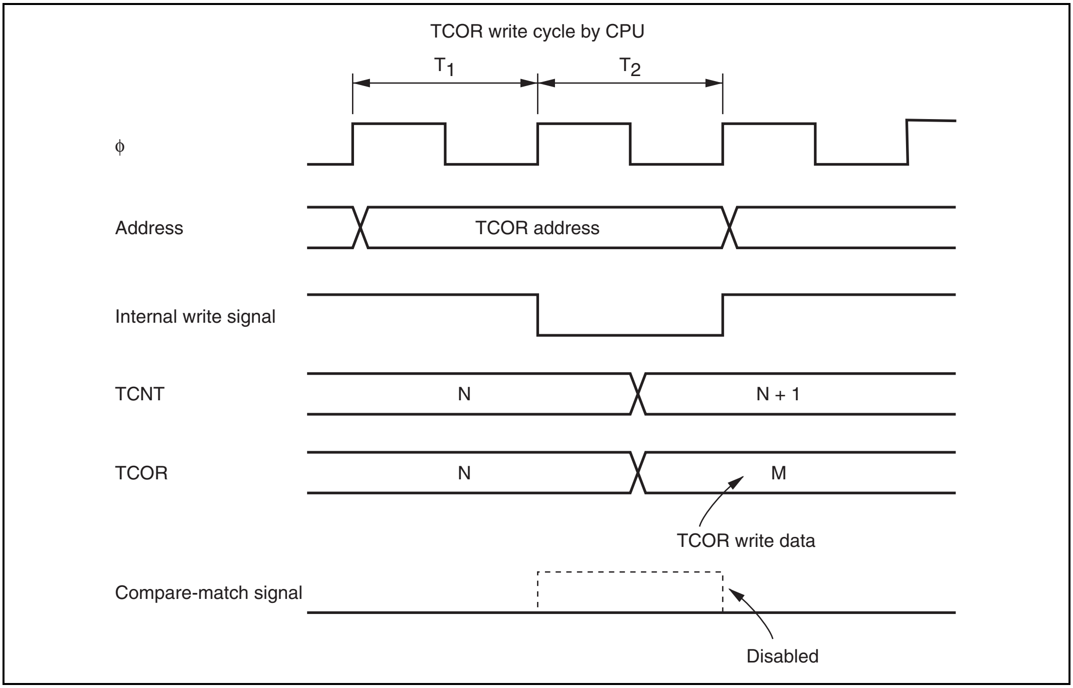


Figure 11.9 Conflict between TCOR Write and Compare-Match

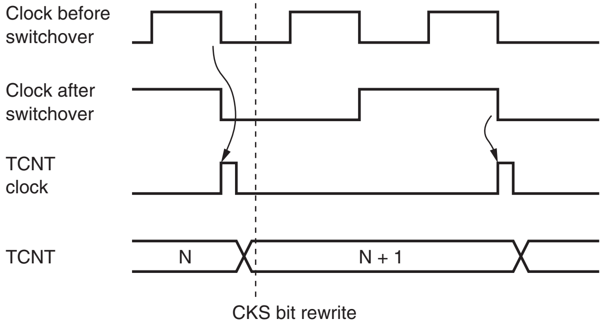
11.6.4 Switching of Internal Clocks and TCNT Operation

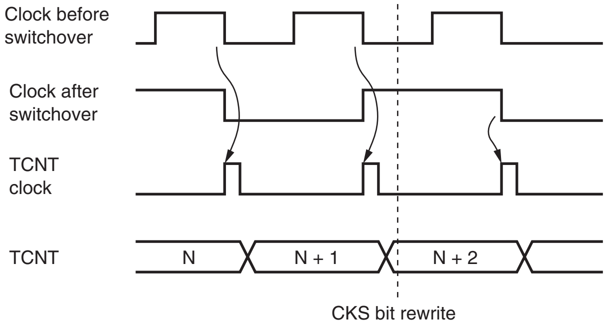
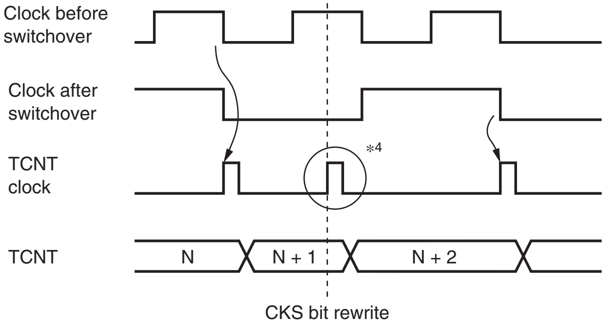
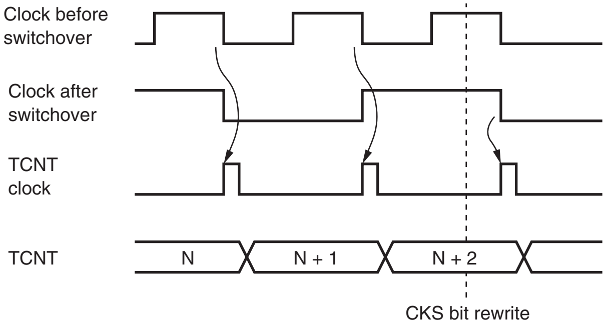
TCNT may increment erroneously when the internal clock is switched over. Table 11.4 shows the relationship between the timing at which the internal clock is switched (by writing to the CKS1 and CKS0 bits) and the TCNT operation.

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 11.4, 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 11.4 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

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.

11.6.5 Mode Setting with Cascaded Connection

If the 16-bit count mode and compare-match count mode are set simultaneously, the input clock pulses for TCNT_0 and TCNT_1 are not generated, and thus the counters will stop operating. Simultaneous setting of these two modes should therefore be avoided.

Section 12 Watchdog Timer (WDT)

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

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows. A block diagram of the WDT_0 and WDT_1 are shown in figure 12.1.

12.1 Features

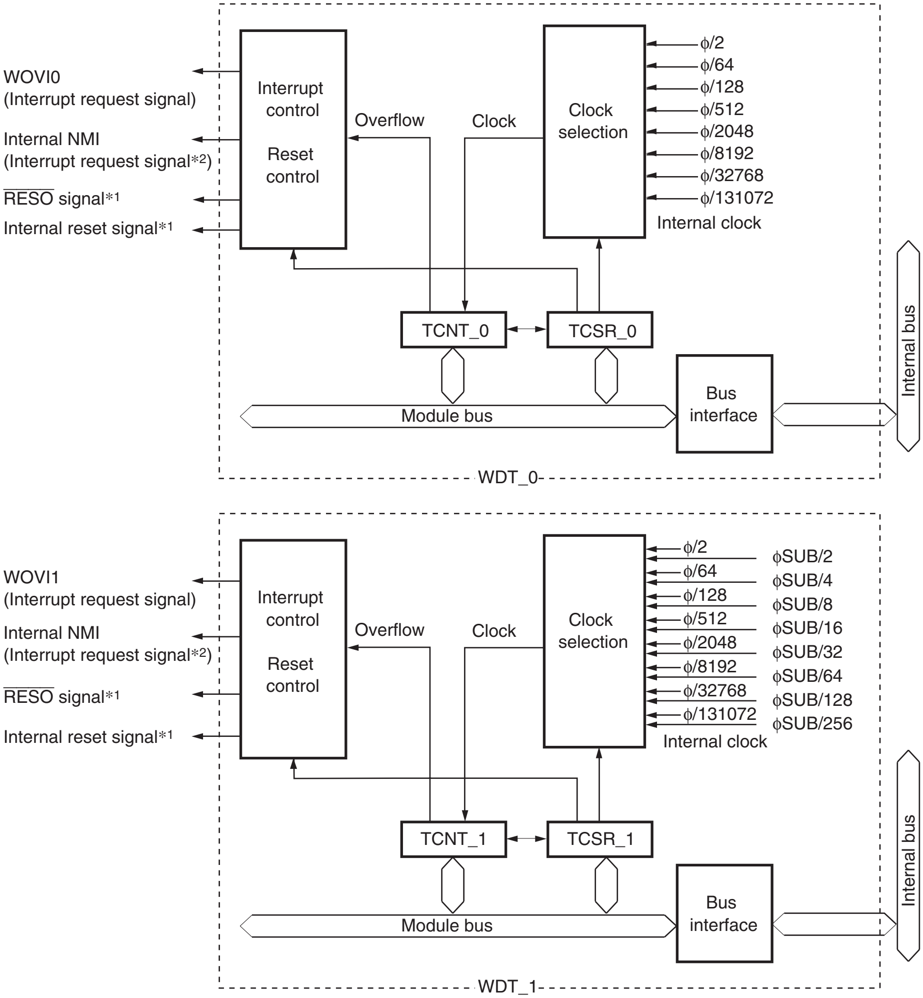
- Selectable from eight (WDT_0) or 16 (WDT_1) counter input clocks.
- Switchable between watchdog timer mode and interval timer mode

Watchdog Timer Mode:

- If the counter overflows, an internal reset or an internal NMI interrupt is generated.
- When the LSI is selected to be internally reset at counter overflow, a low level signal is output from the $\overline{\text{RESO}}$ pin if the counter overflows.

Internal Timer Mode:

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



[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 12.1 Block Diagram of WDT

12.2 Input/Output Pins

The WDT has the pins listed in table 12.1.

Table 12.1 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

12.3 Register Descriptions

The WDT has the following registers. To prevent accidental overwriting, TCSR and TCNT have to be written to in a method different from normal registers. For details, see section 12.6.1, Notes on Register Access. For details on the system control register, see section 3.2.2, System Control Register (SYSCR).

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

12.3.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 when the TME bit in timer control/status register (TCSR) is cleared to 0.

12.3.2 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)*	Overflow Flag Indicates that TCNT has overflowed (changes from H'FF to H'00). [Setting conditions] • When TCNT overflows (changes from H'FF to H'00) • 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, 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 to 0	CKS2 to CKS0	All 0	R/W	Clock Select 2 to 0 Select the clock to be input to TCNT. The overflow period for ϕ = 25 MHz is enclosed in parentheses. 000: ϕ/2 (frequency: 20.48 μs) 001: ϕ/64 (frequency: 655.36 μs) 010: ϕ/128 (frequency: 1.311 ms) 011: ϕ/512 (frequency: 5.243 ms) 100: ϕ/2048 (frequency: 20.97 ms) 101: ϕ/8192 (frequency: 83.89 ms) 110: ϕ/32768 (frequency: 335.5 ms) 111: ϕ/131072 (frequency: 1.34 s)

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

- TCSR_1

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*1	Overflow Flag Indicates that TCNT has overflowed (changes from H'FF to H'00). [Setting conditions] • When TCNT overflows (changes from H'FF to H'00) • 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. When the PSS bit is 1, TCNT is not initialized. Write H'00 to initialize TCNT.
4	PSS	0	R/W	Prescaler Select Selects the clock source to be input to TCNT. 0: Counts the divided cycle of φ–based prescaler (PSM) 1: Counts the divided cycle of φSUB–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

Bit	Bit Name	Initial Value	R/W	Description
2 to 0	CKS2 to CKS0	All 0	R/W	Clock Select 2 to 0 Select the clock to be input to TCNT. The overflow period for ϕ = 25 MHz and ϕSUB = 32.768 kHz is enclosed in parentheses. When PSS = 0: 000: ϕ/2 (frequency: 20.48 μs) 001: ϕ/64 (frequency: 655.36 μs) 010: ϕ/128 (frequency: 1.311 ms) 011: ϕ/512 (frequency: 5.243 ms) 100: ϕ/2048 (frequency: 20.97 ms) 101: ϕ/8192 (frequency: 83.89 ms) 110: ϕ/32768 (frequency: 335.5 ms) 111: ϕ/131072 (frequency: 1.34 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: ϕSUB/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.

12.4 Operation

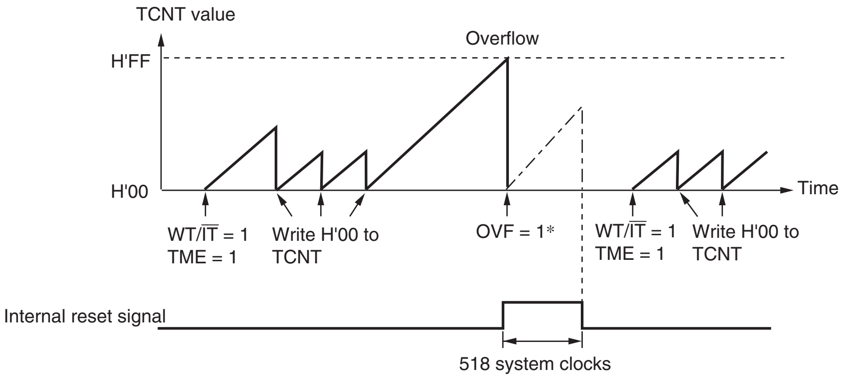
12.4.1 Watchdog Timer Mode

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 12.2. 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.



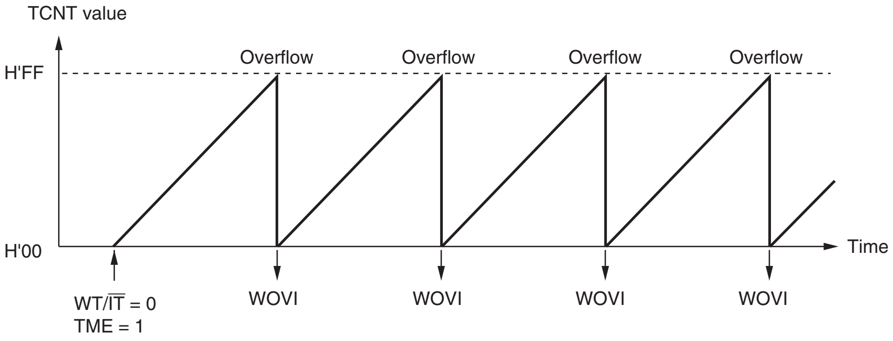
WT/IT: Timer mode select bit
TME: Timer enable bit
OVF: Overflow flag

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 12.2 Watchdog Timer Mode (RST/NMI = 1) Operation

12.4.2 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 12.3. 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 12.4.



WOVI : Interval timer interrupt request occurrence

Figure 12.3 Interval Timer Mode Operation

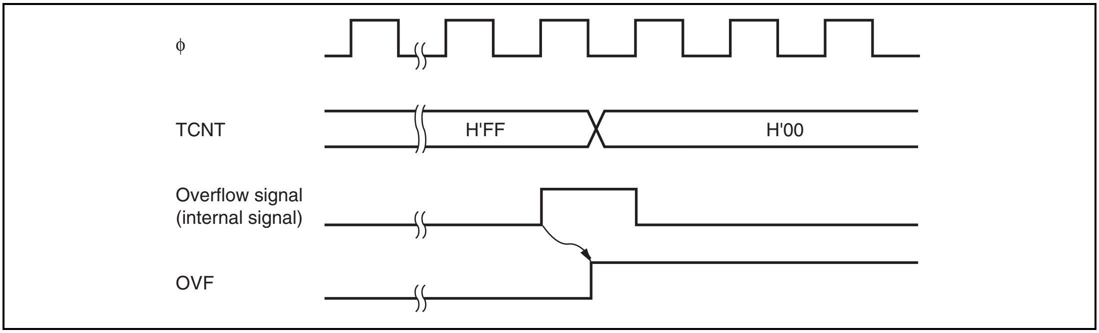


Figure 12.4 OVF Flag Set Timing

12.4.3 $\overline{\text{RESO}}$ 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 12.5.

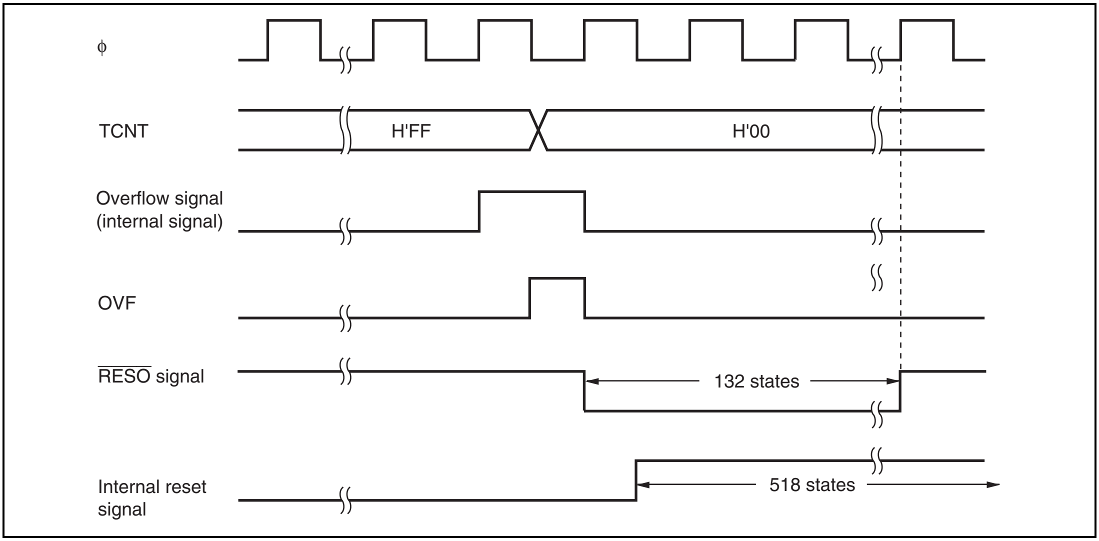


Figure 12.5 Output Timing of $\overline{\text{RESO}}$ Signal

This LSI has retain state pins, which are only initialized by a system reset. The outputs on these pins are retained even when an internal reset is generated by the overflow signal of the WDT. For more information, see section 8, I/O Ports.

12.5 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 12.2 WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag	DTC Activation
WOVI	TCNT overflow	OVF	Not possible

12.6 Usage Notes

12.6.1 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 12.6 to write to TCNT or TCSR. To write to TCNT, the higher bytes must contain the value H'5A and the lower bytes must contain the write data. To write to TCSR, the higher bytes must contain the value H'A5 and the lower bytes must contain the write data.

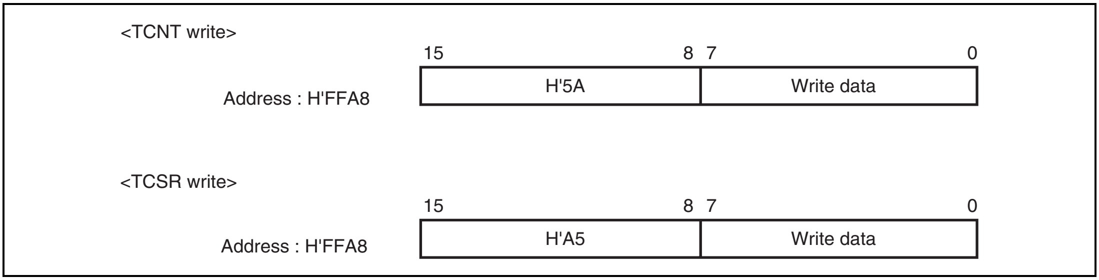


Figure 12.6 Writing to TCNT and TCSR (WDT_0)

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.

12.6.2 Conflict between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the T_2 state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 12.7 shows this operation.

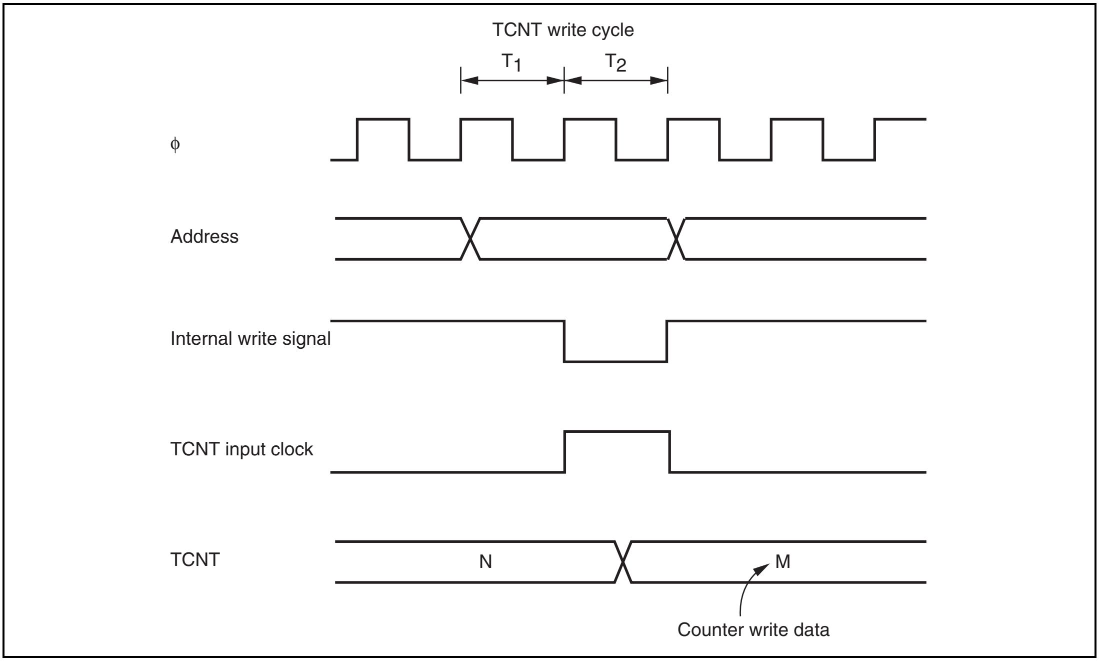


Figure 12.7 Conflict between TCNT Write and Increment

12.6.3 Changing Values of CKS2 to CKS0 Bits

If CKS2 to CKS0 bits 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 CKS2 to CKS0 bits.

12.6.4 Changing Value of PSS Bit

If the PSS bit in TCSR_1 is written to while the WDT is operating, errors could occur in the operation. Stop the watchdog timer (by clearing the TME bit to 0) before changing the values of PSS bit.

12.6.5 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from/to watchdog timer to/from interval timer, while the WDT is operating, errors could occur in the operation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

12.6.6 System Reset by $\overline{\text{RESO}}$ Signal

Inputting the $\overline{\text{RESO}}$ output signal to the $\overline{\text{RES}}$ 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 12.8.

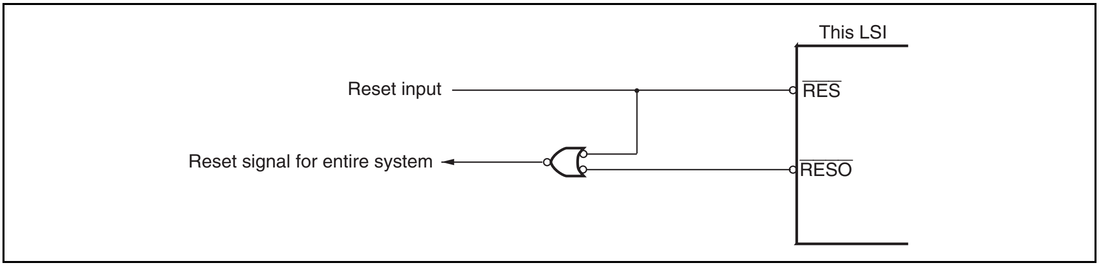


Figure 12.8 Sample Circuit for Resetting the System by the $\overline{\text{RESO}}$ Signal

Section 13 Serial Communication Interface (SCI)

This LSI has two independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clock synchronous serial communication. Asynchronous serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface based on ISO/IEC 7816-3 (Identification Card) as an enhanced asynchronous communication function.

13.1 Features

- Choice of asynchronous or clock synchronous serial communication mode
- Full-duplex communication capability

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected

  The external clock can be selected as a transfer clock source (except for the smart card interface).
- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode 7-bit data)
- Four interrupt sources

  Four interrupt sources — transmit-end, transmit-data-empty, receive-data-full, and receive error — that can issue requests.

  The transmit-data-empty and receive-data-full interrupt sources can activate DTC.
- Module stop mode availability

Asynchronous Mode:

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

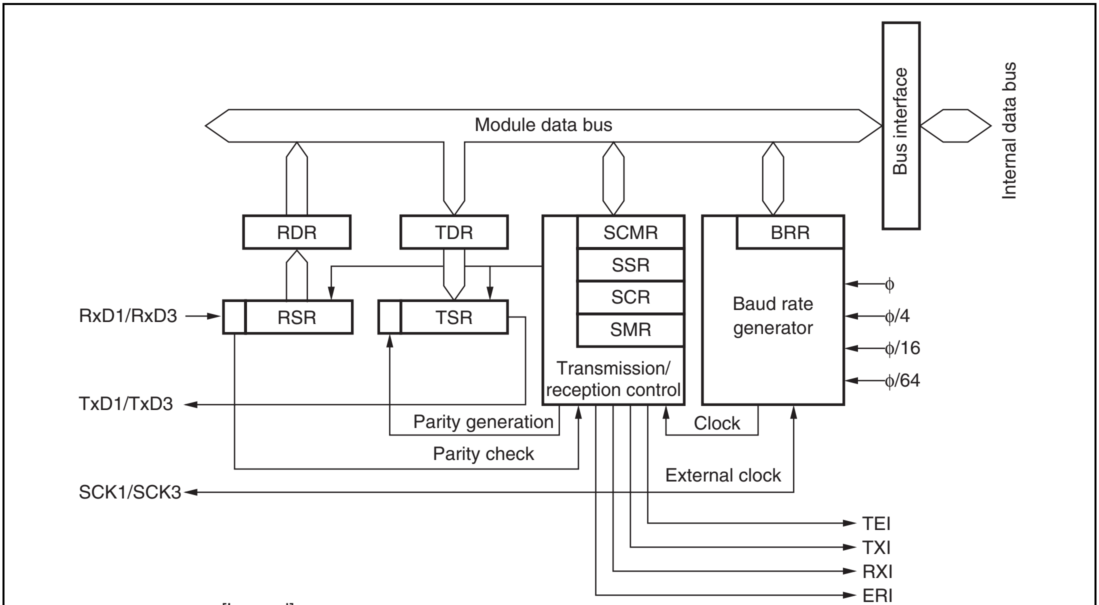
Clock Synchronous Mode:

- Data length: 8 bits
- Receive error detection: Overrun errors

Smart Card Interface:

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on detection of a error signal during transmission
- Both direct convention and inverse convention are supported

Figure 13.1 shows a block diagram of SCI_1 and SCI_3.



[Legend]

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

Figure 13.1 Block Diagram of SCI_1 and SCI_3

13.2 Input/Output Pins

Table 13.1 shows the input/output pins for each SCI channel.

Table 13.1 Pin Configuration

Channel	Symbol*	Input/Output	Function
1	SCK1	Input/Output	Channel 1 clock input/output
	RxD1	Input	Channel 1 receive data input
		Input/Output	Channel 1 transmit/receive data input/output (when smart card interface is selected)
	TxD1	Output	Channel 1 transmit data output
3	SCK3	Input/Output	Channel 3 clock input/output
	RxD3	Input	Channel 3 receive data input
		Input/Output	Channel 3 transmit/receive data input/output (when smart card interface is selected)
	TxD3	Output	Channel 3 transmit data output

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

13.3 Register Descriptions

The SCI has the following registers for each channel. Some bits in the serial mode register (SMR), serial status register (SSR), and serial control register (SCR) have different functions in different modes—normal serial communication interface mode and smart card interface mode; therefore, the bits are described separately for each mode in the corresponding register sections.

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

13.3.1 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.

13.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one frame of serial data, it transfers the received serial data from RSR to RDR where it is stored. After this, RSR can receive the next data. Since RSR and RDR function as a double buffer in this way, continuous receive operations be performed. After confirming that the RDRF bit in SSR is set to 1, read RDR for only once. RDR cannot be written to by the CPU.

13.3.3 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.

13.3.4 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.

13.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock source. Some bits in SMR have different functions in normal mode and smart card interface mode.

- Bit Functions in Normal Serial Communication Interface Mode (when SMIF in SCMR = 0)

Bit	Bit Name	Initial Value	R/W	Description
7	C/$\overline{\text{A}}$	0	R/W	Communication Mode 0: Asynchronous mode 1: Clock 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 clock 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/$\overline{\text{E}}$	0	R/W	Parity Mode (enabled only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity.
3	STOP	0	R/W	Stop Bit Length (enabled only in asynchronous mode) Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.
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.

Bit	Bit Name	Initial Value	R/W	Description
1	CKS1	0	R/W	Clock Select 1 and 0
0	CKS0	0	R/W	These bits select the clock source for the 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 13.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 13.3.9, Bit Rate Register (BRR)).

- Bit Functions in Smart Card Interface Mode (when SMIF in SCMR = 1)

Bit	Bit Name	Initial Value	R/W	Description
7	GM	0	R/W	GSM Mode Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 etu* from the start and the clock output control function is appended. For details, see section 13.7.8, Clock Output Control.
6	BLK	0	R/W	Setting this bit to 1 allows block transfer mode operation. For details, see section 13.7.3, Block Transfer Mode.
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. Set this bit to 1 in smart card interface mode.
4	O/$\overline{\text{E}}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode) 0: Selects even parity 1: Selects odd parity For details on the usage of this bit in smart card interface mode, see section 13.7.2, Data Format (Except in Block Transfer Mode).

Bit	Bit Name	Initial Value	R/W	Description
3	BCP1	0	R/W	Basic Clock Pulse 1 and 0
2	BCP0	0	R/W	These bits select the number of basic clock cycles in a 1-bit data transfer time in smart card interface mode. 00: 32 clock cycles (S = 32) 01: 64 clock cycles (S = 64) 10: 372 clock cycles (S = 372) 11: 256 clock cycles (S = 256) For details, see section 13.7.4, Receive Data Sampling Timing and Reception Margin. S is described in section 13.3.9, Bit Rate Register (BRR).
1	CKS1	0	R/W	Clock Select 1 and 0
0	CKS0	0	R/W	These bits select the clock source for the 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 13.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 13.3.9, Bit Rate Register (BRR)).

Note: * etu: Element Time Unit (time taken to transfer one bit)

13.3.6 Serial Control Register (SCR)

SCR is a register that performs enabling or disabling of SCI transfer operations and interrupt requests, and selection of the transfer clock source. For details on interrupt requests, see section 13.8, Interrupt Sources. Some bits in SCR have different functions in normal mode and smart card interface mode.

- Bit Functions in Normal Serial Communication Interface Mode (when SMIF in SCMR = 0)

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, a TXI interrupt request is enabled.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (enabled only when the MP bit in SMR is 1 in asynchronous mode) When this bit is set to 1, receive data in which the multiprocessor bit is 0 is skipped, and setting of the RDRF, FER, and ORER status flags in SSR is disabled. On receiving data in which the multiprocessor bit is 1, this bit is automatically cleared and normal reception is resumed. For details, see section 13.5, 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 and 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.) Clock 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

- Bit Functions in Smart Card Interface Mode (when SMIF in SCMR = 1)

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1,a TXI interrupt request is enabled.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (enabled only when the MP bit in SMR is 1 in asynchronous mode) Write 0 to this bit in smart card interface mode.
2	TEIE	0	R/W	Transmit End Interrupt Enable Write 0 to this bit in smart card interface mode.
1	CKE1	0	R/W	Clock Enable 1 and 0
0	CKE0	0	R/W	These bits control the clock output from the SCK pin. In GSM mode, clock output can be dynamically switched. For details, see section 13.7.8, Clock Output Control. When GM in SMR = 0 00: Output disabled (SCK pin functions as I/O port.) 01: Clock output 1x: Reserved When GM in SMR = 1 00: Output fixed to low 01: Clock output 10: Output fixed to high 11: Clock output

[Legend]

x: Don't care.

13.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. TDRE, RDRF, ORER, PER, and FER can only be cleared. Some bits in SSR have different functions in normal mode and smart card interface mode.

- Bit Functions in Normal Serial Communication Interface Mode (when SMIF in SCMR = 0)

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	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 • When a TXI interrupt request is issued allowing DTC to write data to TDR
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 • When an RXI interrupt request is issued allowing DTC to read data from RDR The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error [Setting condition] When the next serial reception is completed while RDRF = 1 [Clearing condition] When 0 is written to ORER after reading ORER = 1
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 • When a TXI interrupt request is issued allowing DTC to write data to TDR
1	MPB	0	R	Multiprocessor Bit MPB stores the multiprocessor bit in the receive frame. When the RE bit in SCR is cleared to 0 its previous state is retained.
0	MPBT	0	R/W	Multiprocessor Bit Transfer MPBT stores the multiprocessor bit to be added to the transmit frame.

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

- Bit Functions in Smart Card Interface Mode (when SMIF in SCMR = 1)

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*[1]	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 can be written to. [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When a TXI interrupt request is issued allowing DTC to write data to TDR
6	RDRF	0	R/(W)*[1]	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 • When an RXI interrupt request is issued allowing DTC to read data from RDR The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.
5	ORER	0	R/(W)*[1]	Overrun Error [Setting condition] When the next serial reception is completed while RDRF = 1 [Clearing condition] When 0 is written to ORER after reading ORER = 1
4	ERS	0	R/(W)*[1]	Error Signal Status [Setting condition] When a low error signal is sampled [Clearing condition] When 0 is written to ERS after reading ERS = 1

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*[1]	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 TEND is set to 1 when the receiving end acknowledges no error signal and the next transmit data is ready to be transferred to TDR. [Setting conditions] • When both TE in SCR and ERS are 0 • When ERS = 0 and TDRE = 1 after a specified time passed after the start of 1-byte data transfer. The set timing depends on the register setting as follows. When GM = 0 and BLK = 0, 2.5 etu*[2] after transmission start When GM = 0 and BLK = 1, 1.5 etu*[2] after transmission start When GM = 1 and BLK = 0, 1.0 etu*[2] after transmission start When GM = 1 and BLK = 1, 1.0 etu*[2] after transmission start [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When a TXI interrupt request is issued allowing DTC to write the next data to TDR
1	MPB	0	R	Multiprocessor Bit Not used in smart card interface mode.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Write 0 to this bit in smart card interface mode.

Notes: 1. Only 0 can be written to clear the flag.
2. etu: Element Time Unit (time taken to transfer one bit)

13.3.8 Smart Card Mode Register (SCMR)

SCMR selects smart card interface mode and its format.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These bits are always read as 1 and cannot be modified.
3	SDIR	0	R/W	Smart Card Data Transfer Direction Selects the serial/parallel conversion format. 0: TDR contents are transmitted with LSB-first. Stores receive data as LSB first in RDR. 1: TDR contents are transmitted with MSB-first. Stores receive data as MSB first in RDR. The SDIR bit is valid only when the 8-bit data format is used for transmission/reception; when the 7-bit data format is used, data is always transmitted/received with LSB-first.
2	SINV	0	R/W	Smart Card Data Invert Specifies inversion of the data logic level. The SINV bit does not affect the logic level of the parity bit. When the parity bit is inverted, invert the O/$\overline{\text{E}}$ bit in SMR. 0: TDR contents are transmitted as they are. Receive data is stored as it is in RDR. 1: TDR contents are inverted before being transmitted. Receive data is stored in inverted form in RDR.
1	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
0	SMIF	0	R/W	Smart Card Interface Mode Select When this bit is set to 1, smart card interface mode is selected. 0: Normal asynchronous or clock synchronous mode 1: Smart card interface mode

13.3.9 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 13.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clock synchronous mode, and smart card interface mode. The initial value of BRR is H′FF, and it can be read from or written to by the CPU at all times.

Table 13.2 Relationships between N Setting in BRR and Bit Rate B

Mode	Bit Rate	Error
Asynchronous mode	$B = \dfrac{\phi \times 10^6}{64 \times 2^{2n-1} \times (N+1)}$	$\text{Error (\%)} = \left\{ \dfrac{\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1 \right\} \times 100$
Clock synchronous mode	$B = \dfrac{\phi \times 10^6}{8 \times 2^{2n-1} \times (N+1)}$	—
Smart card interface mode	$B = \dfrac{\phi \times 10^6}{S \times 2^{2n+1} \times (N+1)}$	$\text{Error (\%)} = \left\{ \dfrac{\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N+1)} - 1 \right\} \times 100$

[Legend]

B: Bit rate (bit/s)

N: BRR setting for baud rate generator (0 ≤ N ≤ 255)

φ: Operating frequency (MHz)

n and S: 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

SMR Setting		
BCP1	**BCP0**	**S**
0	0	32
0	1	64
1	0	372
1	1	256

Table 13.3 shows sample N settings in BRR in normal asynchronous mode. Table 13.4 shows the maximum bit rate settable for each frequency. Table 13.6 and 13.8 show sample N settings in BRR in clock synchronous mode and smart card interface mode, respectively. In smart card interface mode, the number of basic clock cycles S in a 1-bit data transfer time can be selected. For details, see section 13.7.4, Receive Data Sampling Timing and Reception Margin. Tables 13.5 and 13.7 show the maximum bit rates with external clock input.

Table 13.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode)

	Operating Frequency ϕ (MHz)					
	20			25		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)
110	3	88	–0.25	3	110	–0.02
150	3	64	0.16	3	80	–0.47
300	2	129	0.16	2	162	0.15
600	2	64	0.16	2	80	–0.47
1200	1	129	0.16	1	162	0.15
2400	1	64	0.16	1	80	–0.47
4800	0	129	0.16	0	162	0.15
9600	0	64	0.16	0	80	–0.47
19200	0	32	–1.36	0	40	–0.76
31250	0	19	0.00	0	24	0.00
38400	0	15	1.73	0	19	1.73

Note: Make the settings so that the error does not exceed 1%.

Table 13.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

ϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
20	625000	0	0
25	781250	0	0

Table 13.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)

ϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
20	5.0000	312500
25	6.2500	390625

Table 13.6 BRR Settings for Various Bit Rates (Clock Synchronous Mode)

Bit Rate (bit/s)	Operating Frequency ϕ (MHz)			
	20		24	
	n	N	n	N
110				
250				
500	—	—	—	—
1k	—	—	—	—
2.5k	2	124	2	149
5k	1	249	2	74
10k	1	124	1	149
25k	0	199	0	239
50k	0	99	0	119
100k	0	49	0	59
250k	0	19	0	23
500k	0	9	0	11
1M	0	4	0	5
2.5M	0	1		
5M	0	0*		

[Legend]

—: Can be set, but there will be a degree of error.

*: Continuous transfer or reception is not possible.

Table 13.7 Maximum Bit Rate with External Clock Input (Clock Synchronous Mode)

ϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
20	3.3333	3333333.3
25	4.1667	4166666.7

Table 13.8 BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, s = 372)

Bit Rate	Operating Frequency φ (MHz)								
	20.00			21.4272			25		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	2	-6.65	0	2	0.00	0	3	-12.49

Table 13.9 Maximum Bit Rate for Each Frequency (Smart Card Interface Mode, S = 372)

φ (MHz)	Maximum Bit Rate (bit/s)	n	N
20.00	26882	0	0
21.4272	28800	0	0
25.00	33602	0	0

13.4 Operation in Asynchronous Mode

Figure 13.2 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.

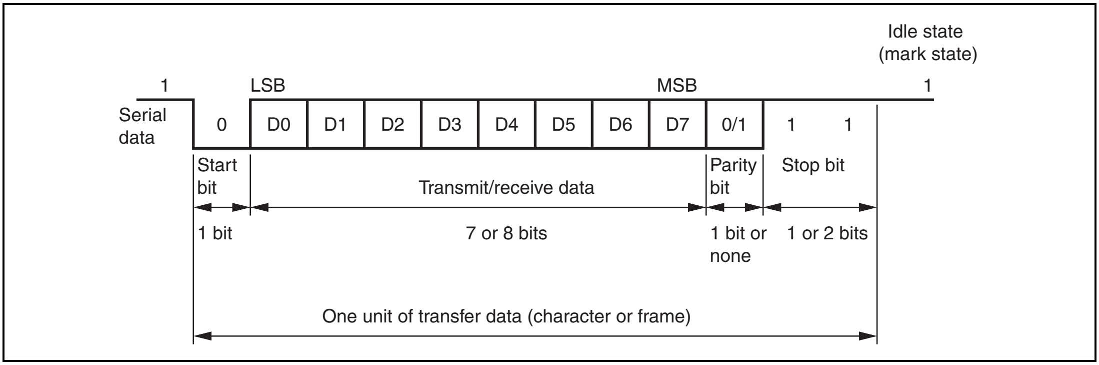


Figure 13.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

13.4.1 Data Transfer Format

Table 13.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting. For details on the multiprocessor bit, see section 13.5, Multiprocessor Communication Function.

Table 13.10 Serial Transfer Formats (Asynchronous Mode)

SMR Settings				Serial Transmit/Receive Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
0	—	1	0	S	8-bit data								MPB	STOP	
0	—	1	1	S	8-bit data								MPB	STOP	STOP
1	—	1	0	S	7-bit data							MPB	STOP		
1	—	1	1	S	7-bit data							MPB	STOP	STOP	

[Legend]

S:	Start bit
STOP:	Stop bit
P:	Parity bit
MPB:	Multiprocessor bit

13.4.2 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 13.3. 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.

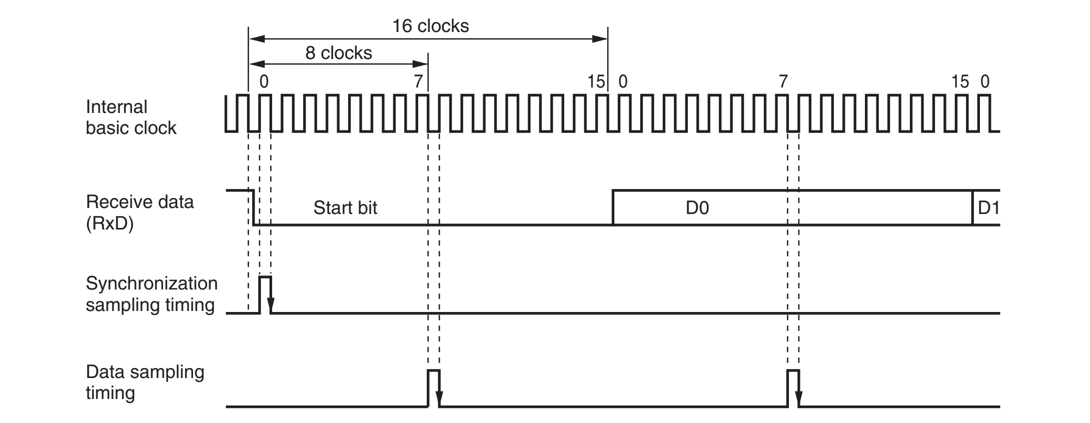


Figure 13.3 Receive Data Sampling Timing in Asynchronous Mode

13.4.3 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 13.4.

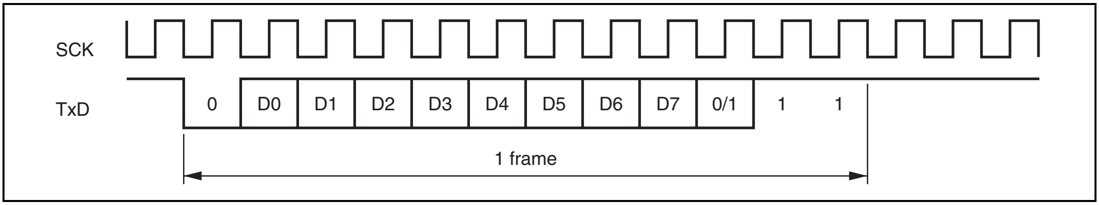
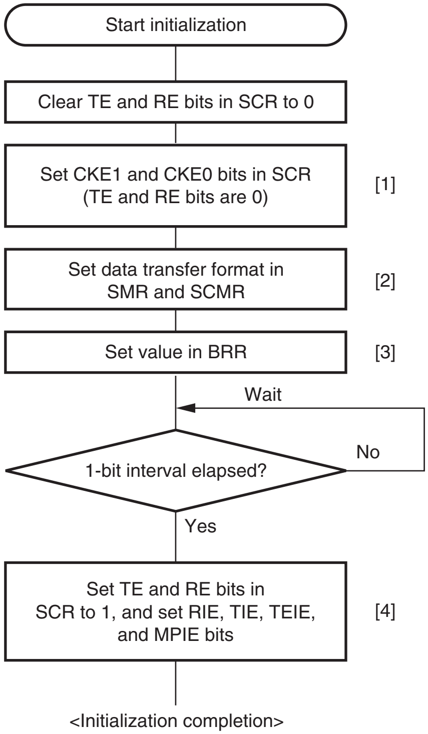


Figure 13.4 Relation between Output Clock and Transmit Data Phase (Asynchronous Mode)

13.4.4 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 13.5. 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 the 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 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]
<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 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 13.5 Sample SCI Initialization Flowchart

13.4.5 Serial Data Transmission (Asynchronous Mode)

Figure 13.6 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 13.7 shows a sample flowchart for transmission in asynchronous mode.

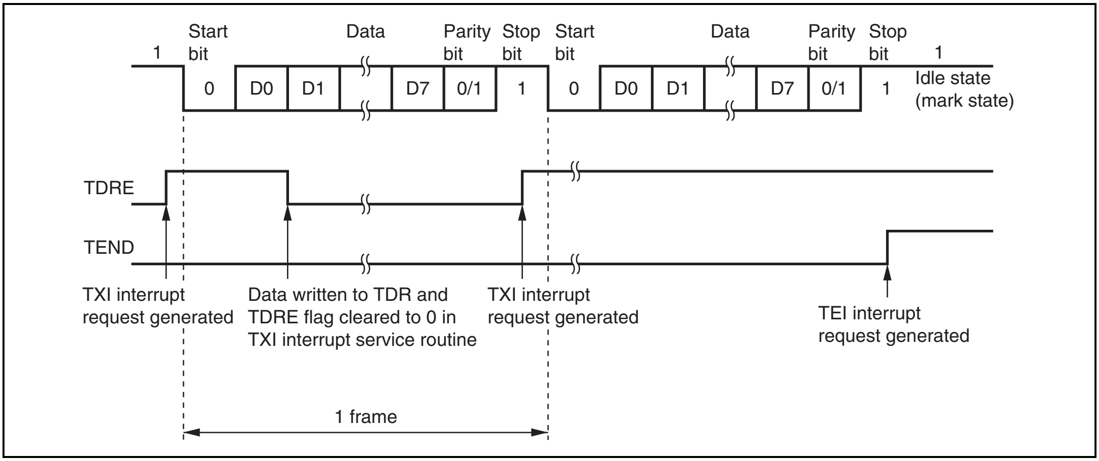
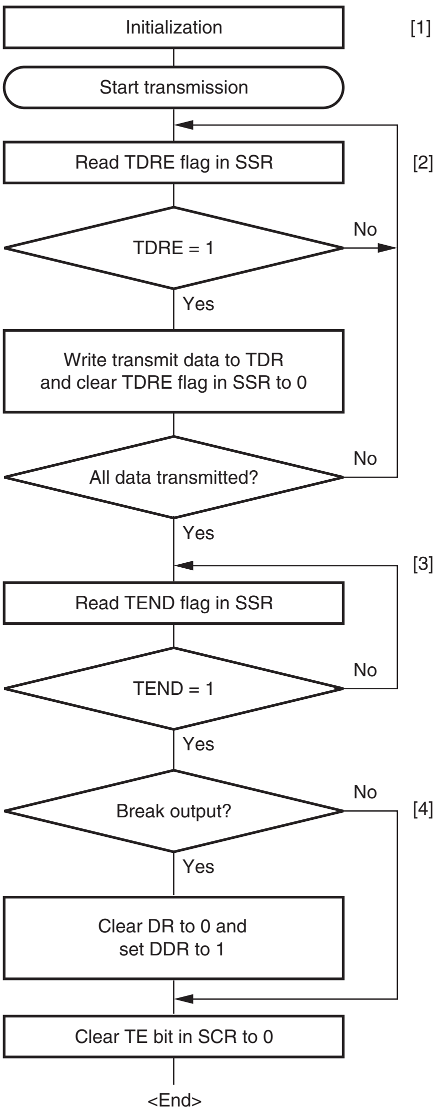

1
Start bit
Data
Parity bit
Stop bit
Start bit
Data
Parity bit
Stop bit
1
0
D0
D1
D7
0/1
1
0
D0
D1
D7
0/1
1
Idle state (mark state)
TDRE
TEND
TXI interrupt request generated
Data written to TDR and TDRE flag cleared to 0 in TXI interrupt service routine
TXI interrupt request generated
TEI interrupt request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

Figure 13.6 Example of Operation in Transmission 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. However, the TDRE flag is checked and cleared automatically when the DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

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

Figure 13.7 Sample Serial Transmission Flowchart

13.4.6 Serial Data Reception (Asynchronous Mode)

Figure 13.8 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.

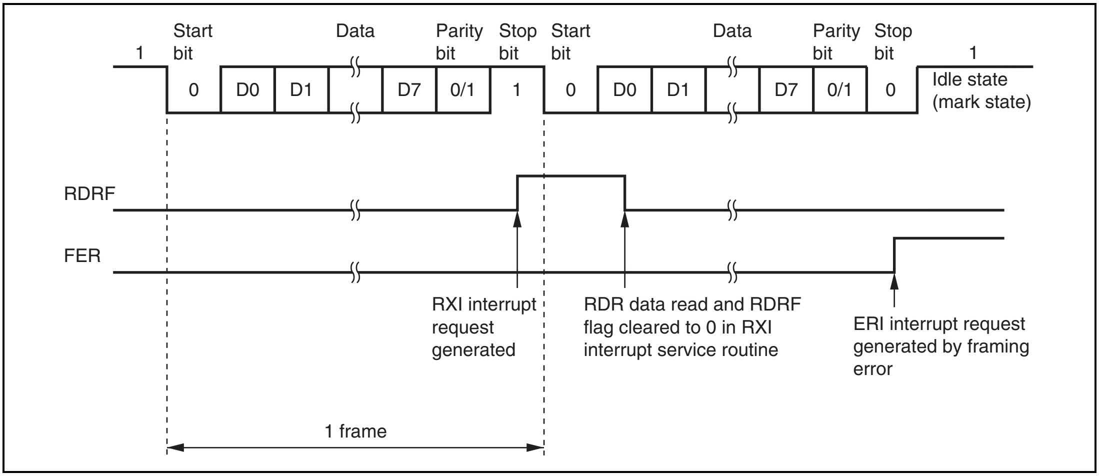

1
Start bit
Data
Parity bit
Stop bit
Start bit
Data
Parity bit
Stop bit
1
0
D0
D1
D7
0/1
1
0
D0
D1
D7
0/1
0
Idle state (mark state)
RDRF
FER
RXI interrupt request generated
RDR data read and RDRF flag cleared to 0 in RXI interrupt service routine
ERI interrupt request generated by framing error
1 frame

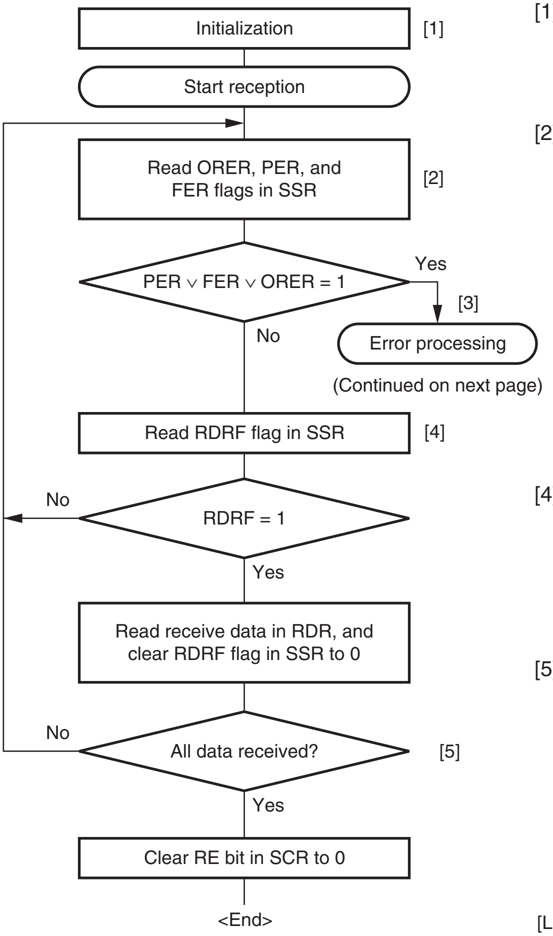
Figure 13.8 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit)

Table 13.11 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 13.9 shows a sample flowchart for serial data reception.

Table 13.11 SSR Status Flags and Receive Data Handling

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

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



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

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

[4] SCI 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. However, the RDRF flag is cleared automatically when the DTC is initiated by an RXI interrupt and reads data from RDR.

[Legend]
∨ : Logical add (OR)

Figure 13.9 Sample Serial Reception Flowchart (1)

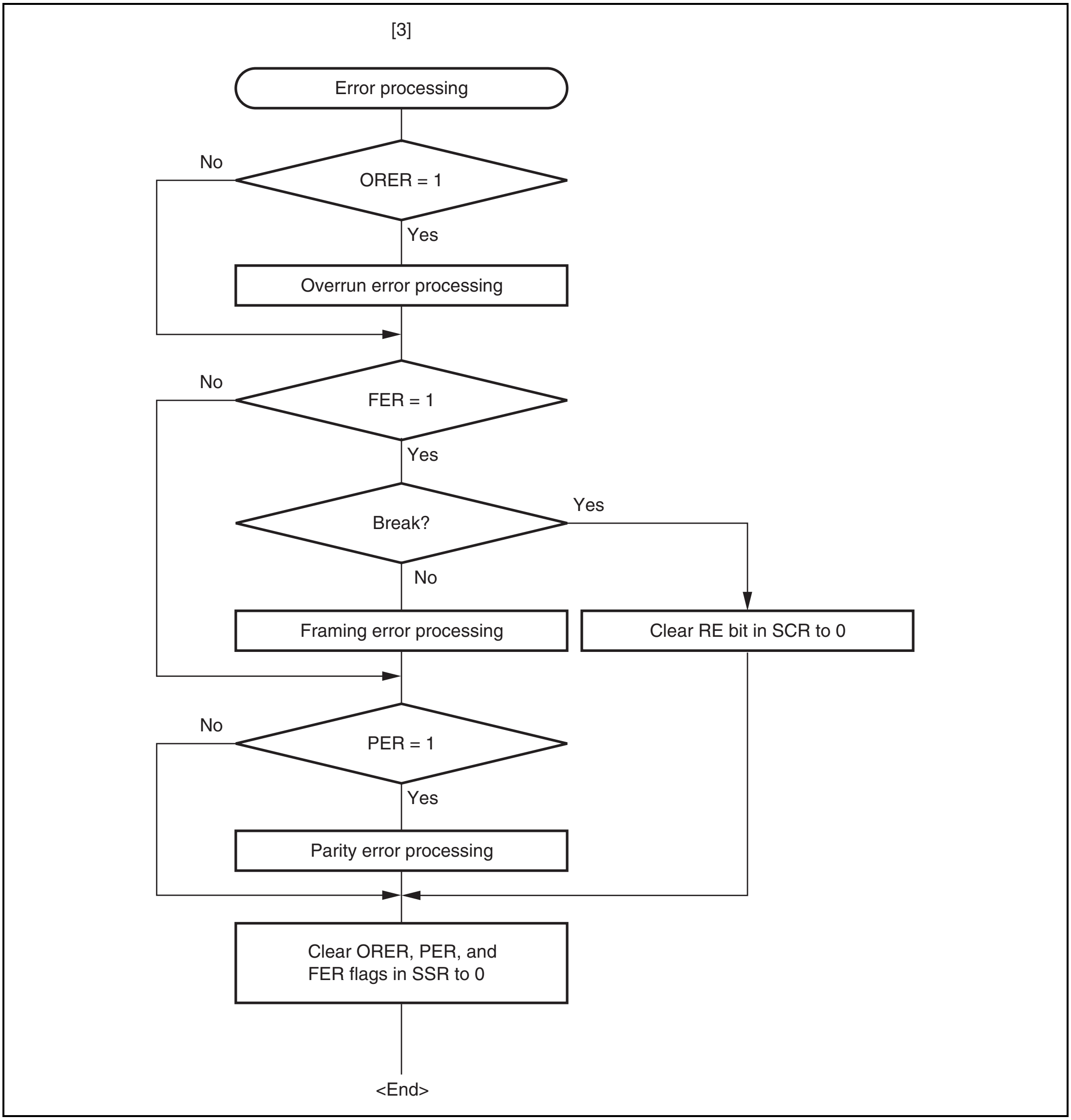


Figure 13.9 Sample Serial Reception Flowchart (2)

13.5 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 13.10 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. 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 RDRF, FER, and ORER status flags 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.

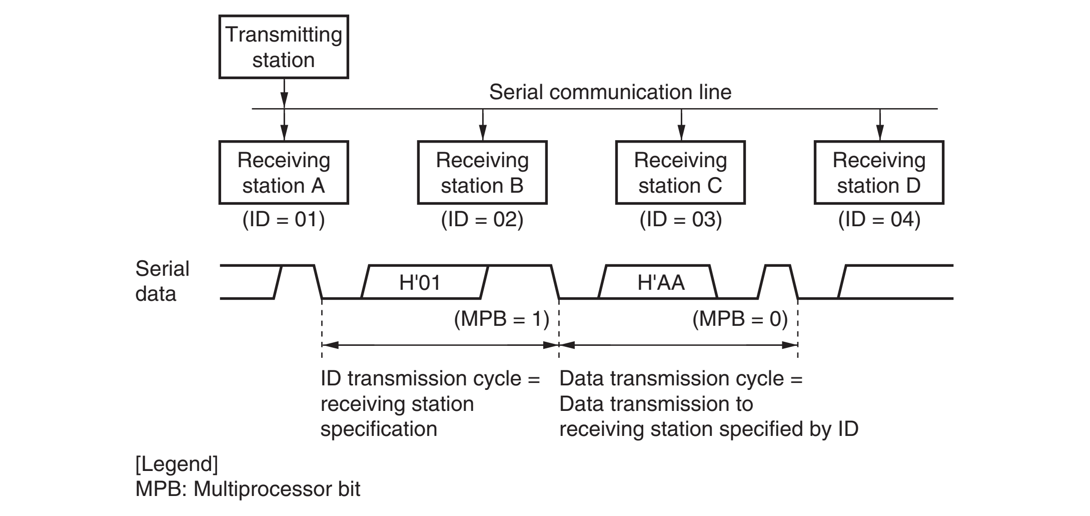
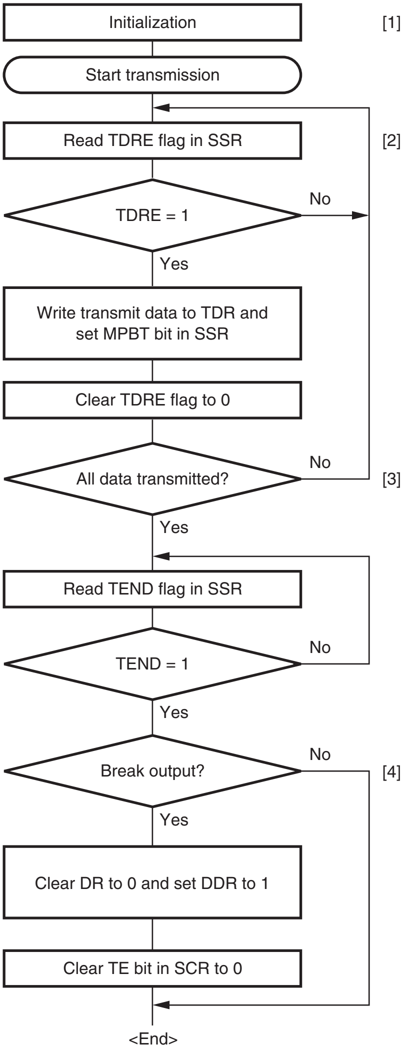


Figure 13.10 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)

13.5.1 Multiprocessor Serial Data Transmission

Figure 13.11 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.

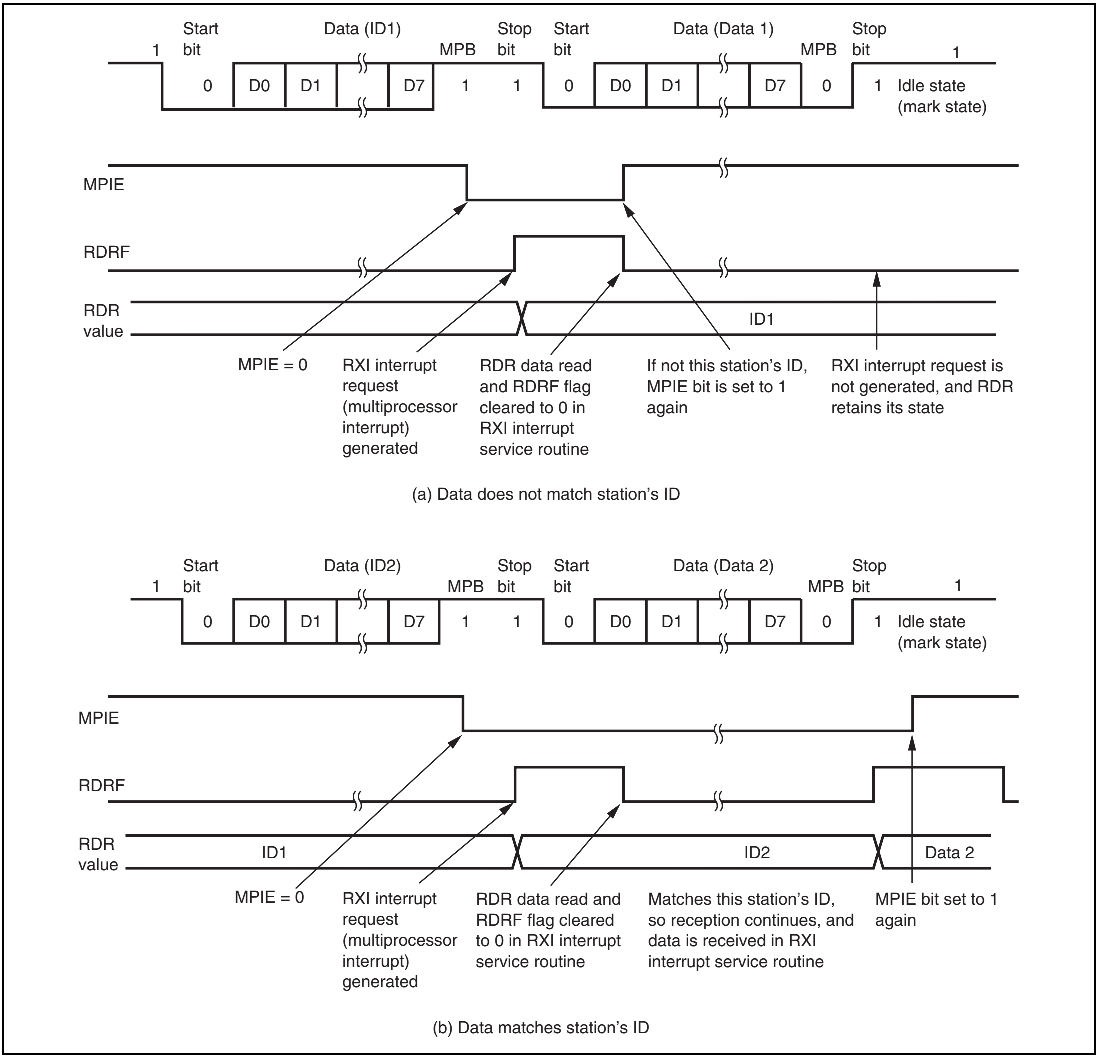
[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

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

Figure 13.11 Sample Multiprocessor Serial Transmission Flowchart

13.5.2 Multiprocessor Serial Data Reception

Figure 13.13 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 13.12 shows an example of SCI operation for multiprocessor format reception.



MPIE = 0

RXI interrupt request (multiprocessor interrupt) generated

RDR data read and RDRF flag cleared to 0 in RXI interrupt service 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

MPIE = 0

RXI interrupt request (multiprocessor interrupt) generated

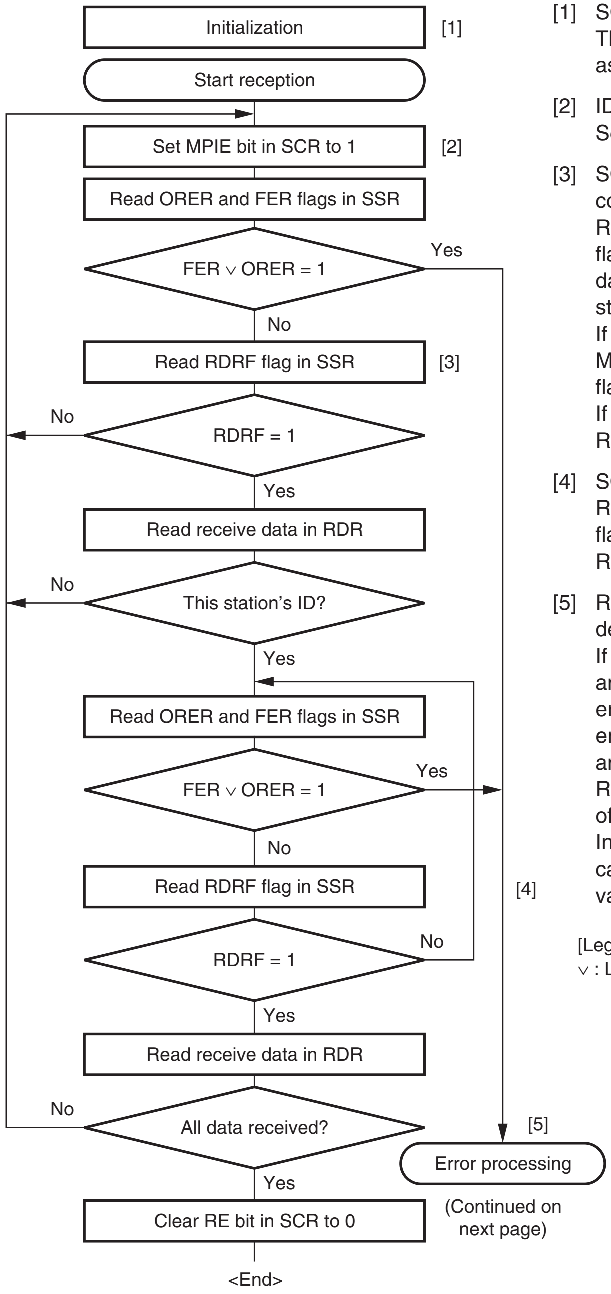
RDR data read and RDRF flag cleared to 0 in RXI interrupt service routine

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 13.12 Example of SCI Operation in Reception (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 add (OR)

Figure 13.13 Sample Multiprocessor Serial Reception Flowchart (1)

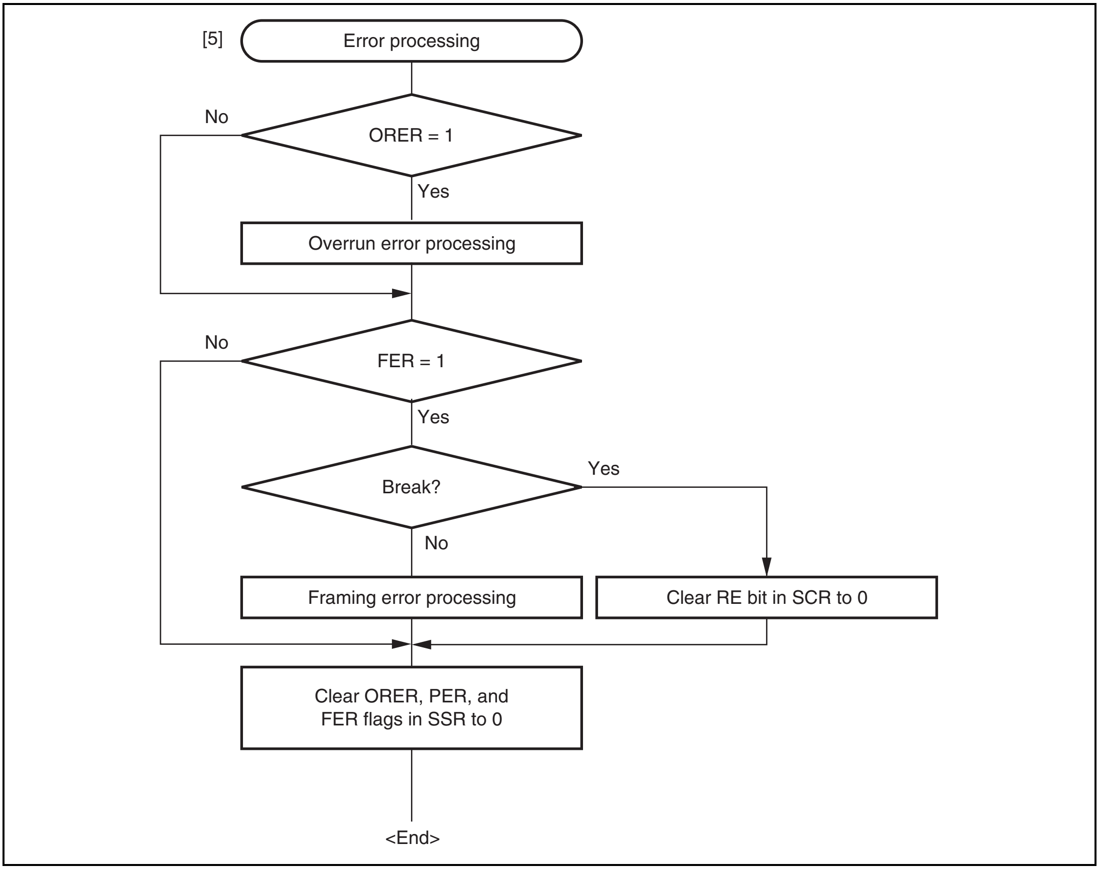
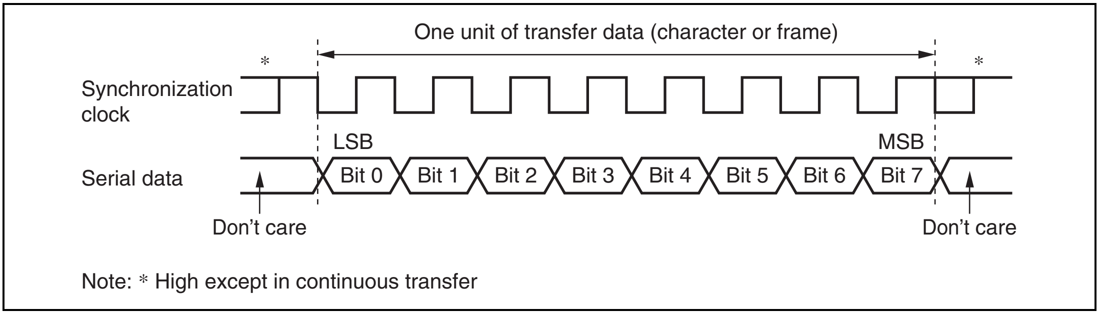


Figure 13.13 Sample Multiprocessor Serial Reception Flowchart (2)

13.6 Operation in Clock Synchronous Mode

Figure 13.14 shows the general format for clock synchronous communication. In clock 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 clock 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

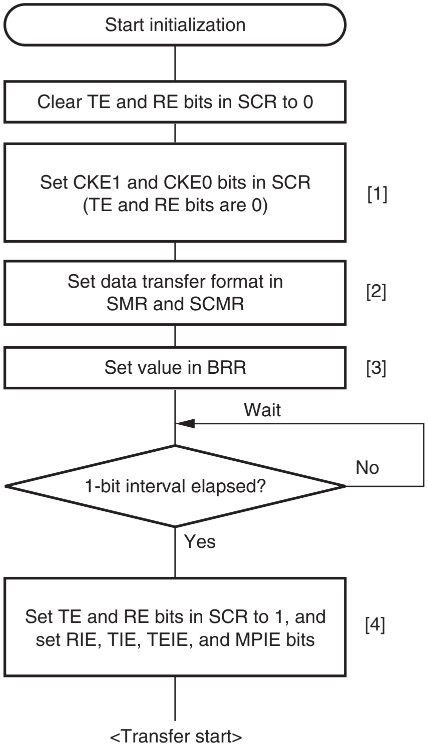
Figure 13.14 Data Format in Synchronous Communication (LSB-First)

13.6.1 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.

13.6.2 SCI Initialization (Clock 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 13.15. 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 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]
<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 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 13.15 Sample SCI Initialization Flowchart

13.6.3 Serial Data Transmission (Clock Synchronous Mode)

Figure 13.16 shows an example of SCI operation for transmission in clock 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 13.17 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.

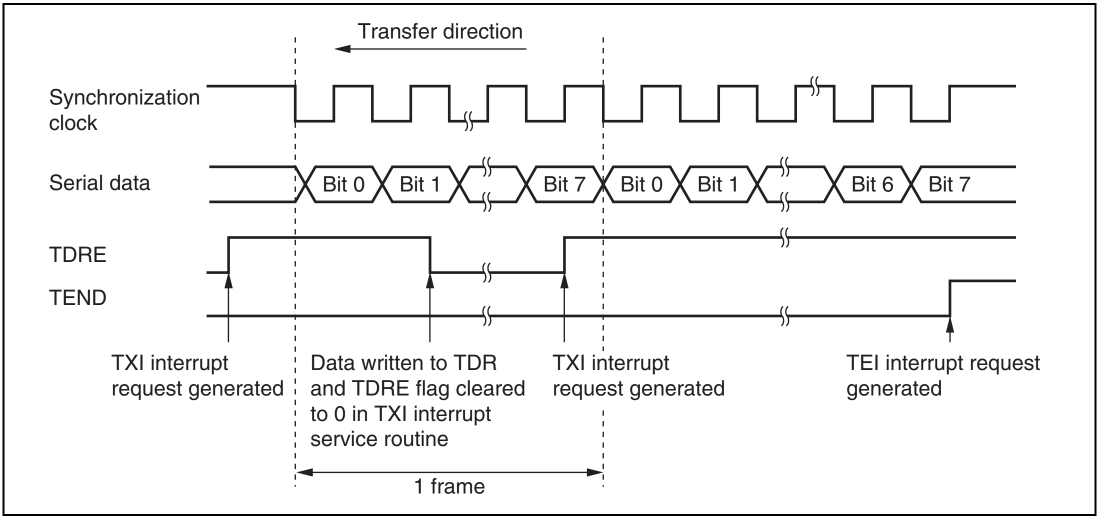
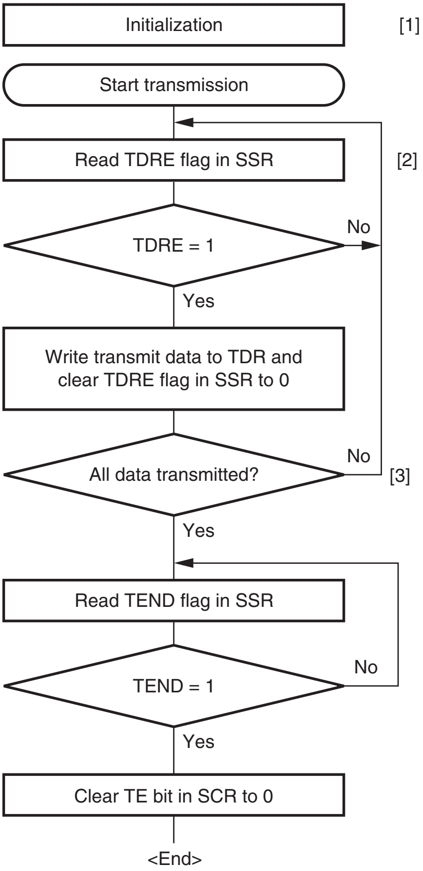

Transfer direction
Synchronization clock
Serial data
Bit 0
Bit 1
Bit 7
Bit 0
Bit 1
Bit 6
Bit 7
TDRE
TEND
TXI interrupt request generated
Data written to TDR and TDRE flag cleared to 0 in TXI interrupt service routine
TXI interrupt request generated
TEI interrupt request generated
1 frame

Figure 13.16 Sample SCI Transmission Operation in Clock 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.
However, the TDRE flag is checked and cleared automatically when the DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

Figure 13.17 Sample Serial Transmission Flowchart

13.6.4 Serial Data Reception (Clock Synchronous Mode)

Figure 13.18 shows an example of SCI operation for reception in clock 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.

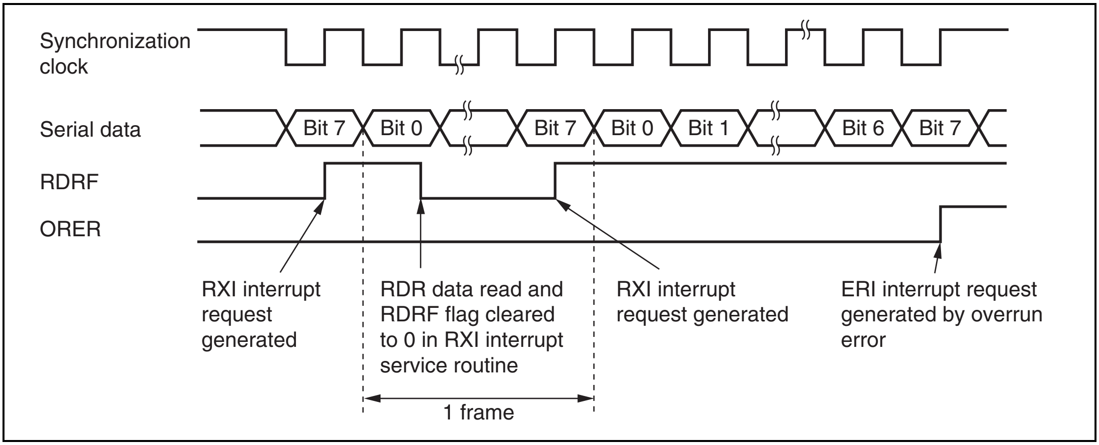
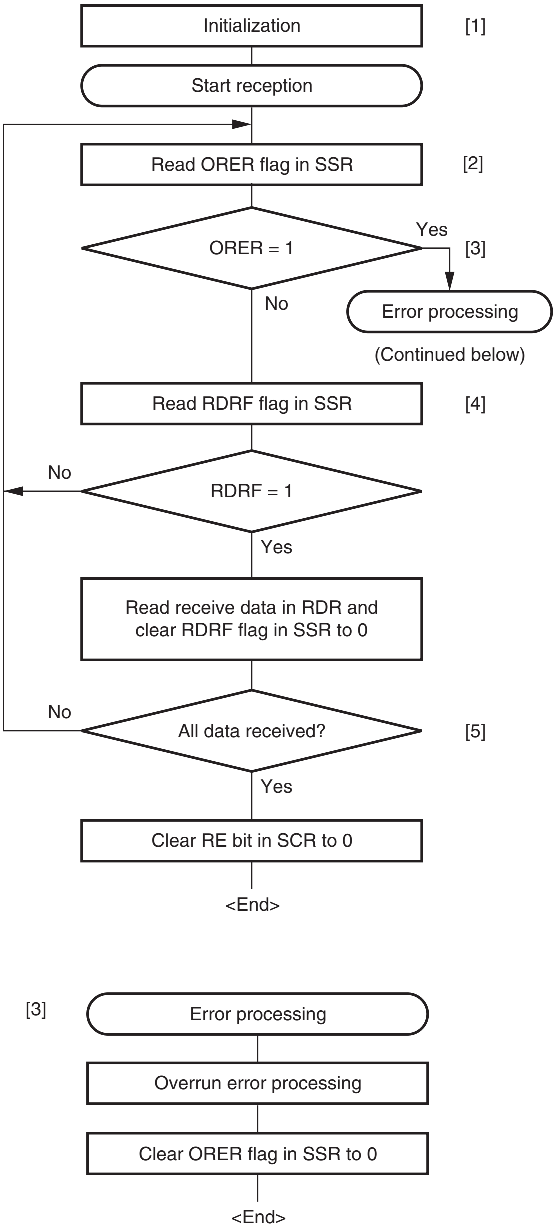

Synchronization clock
Serial data
Bit 7
Bit 0
Bit 7
Bit 0
Bit 1
Bit 6
Bit 7
RDRF
ORER
RXI interrupt request generated
RDR data read and RDRF flag cleared to 0 in RXI interrupt service routine
RXI interrupt request generated
ERI interrupt request generated by overrun error
1 frame

Figure 13.18 Example of SCI Receive Operation in Clock 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 13.19 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.

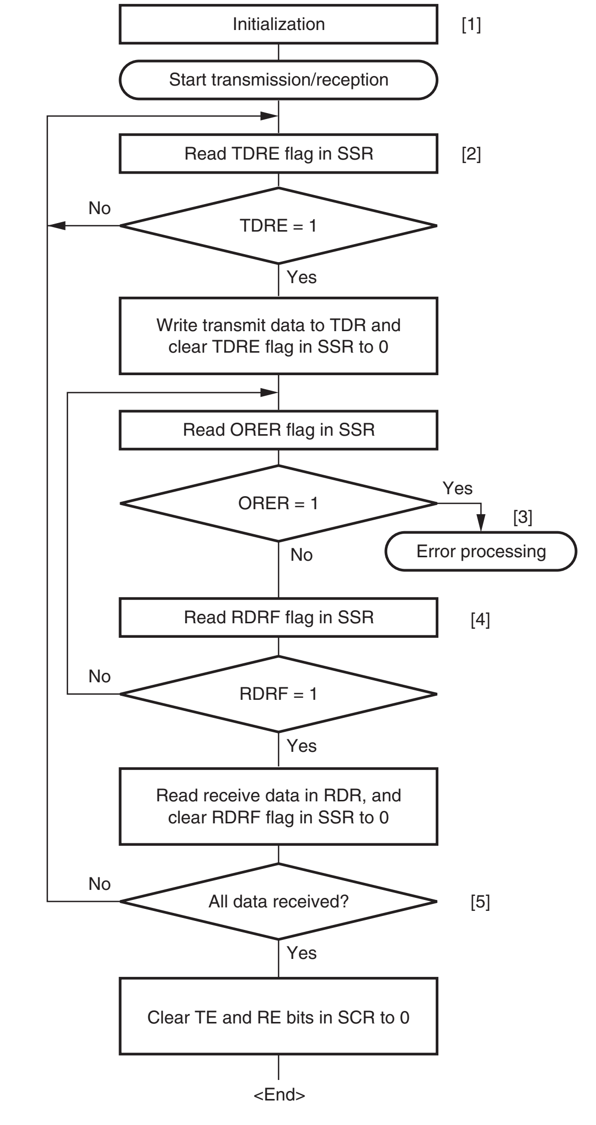
[4] SCI status check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0.
Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[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.
However, the RDRF flag is cleared automatically when the DTC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

Figure 13.19 Sample Serial Reception Flowchart

13.6.5 Simultaneous Serial Data Transmission and Reception (Clock Synchronous Mode)

Figure 13.20 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags in SSR are set to 1, clear the TE bit in SCR to 0. Then simultaneously set the TE and RE bits to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI has finished reception, clear the RE bit to 0. Then after checking that the RDRF bit in SSR and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set the TE and RE bits to 1 with a single instruction.


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>

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.

[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.
However, the TDRE flag is checked and cleared automatically when the DTC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR. Similarly, the RDRF flag is cleared automatically when the DTC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

Figure 13.20 Sample Flowchart of Simultaneous Serial Transmission and Reception

13.7 Smart Card Interface Description

The SCI supports the IC card (smart card) interface based on the ISO/IEC 7816-3 (Identification Card) standard as an enhanced serial communication interface function. Smart card interface mode can be selected using the appropriate register.

13.7.1 Sample Connection

Figure 13.21 shows a sample connection between the smart card and this LSI. This LSI communicates with the IC card using a single transmission line. When the SMIF bit in SCMR is set to 1, the TxD and RxD pins are interconnected inside the LSI, which makes the RxD pin function as an I/O pin. Pull up the data transmission line to VCC using a resistor. Setting the RE and TE bits in SCR to 1 with the IC card not connected enables closed transmission/reception allowing self diagnosis. To supply the IC card with the clock pulses generated by the SCI, input the SCK pin output to the CLK pin of the IC card. A reset signal can be supplied via the output port of this LSI.

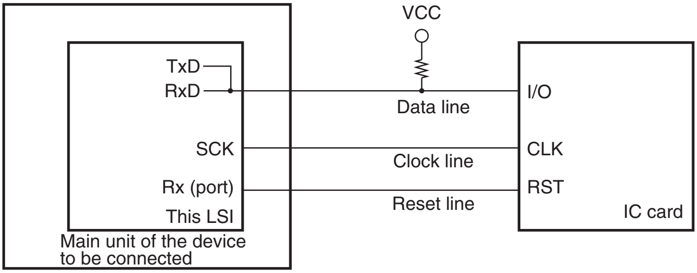


Figure 13.21 Pin Connection for Smart Card Interface

13.7.2 Data Format (Except in Block Transfer Mode)

Figure 13.22 shows the data transfer formats in smart card interface mode.

- One frame contains 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etu (elementary time unit: time required for transferring one bit) is secured as a guard time after the end of the parity bit before the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etu has passed from the start bit.
- If an error signal is sampled during transmission, the same data is automatically re-transmitted after two or more etu.

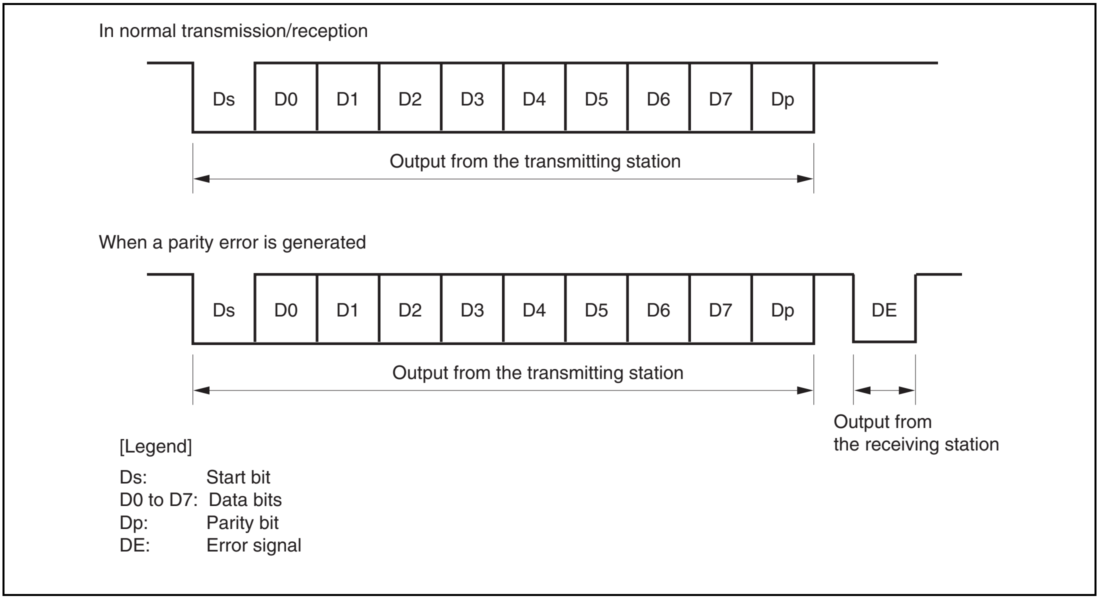


Figure 13.22 Data Formats in Normal Smart Card Interface Mode

For communication with the IC cards of the direct convention and inverse convention types, follow the procedure below.

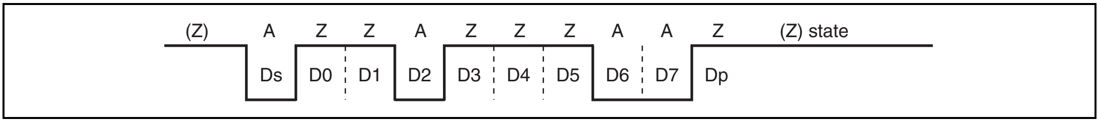


Figure 13.23 Direct Convention (SDIR = SINV = O/$\overline{\text{E}}$ = 0)

For the direct convention type, logic levels 1 and 0 correspond to states Z and A, respectively, and data is transferred with LSB-first as the start character, as shown in figure 13.23. Therefore, data in the start character in the figure is H'3B. When using the direct convention type, write 0 to both the SDIR and SINV bits in SCMR. Write 0 to the O/$\overline{\text{E}}$ bit in SMR in order to use even parity, which is prescribed by the smart card standard.

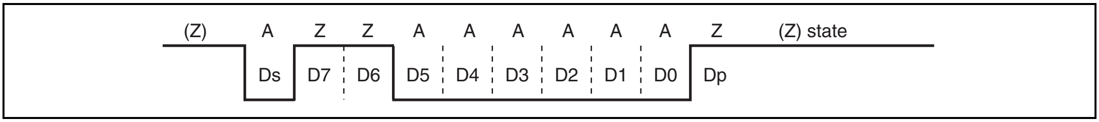


Figure 13.24 Inverse Convention (SDIR = SINV = O/$\overline{\text{E}}$ = 1)

For the inverse convention type, logic levels 1 and 0 correspond to states A and Z, respectively and data is transferred with MSB-first as the start character, as shown in figure 13.24. Therefore, data in the start character in the figure is H'3F. When using the inverse convention type, write 1 to both the SDIR and SINV bits in SCMR. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to state Z. Since the SINV bit of this LSI only inverts data bits D7 to D0, write 1 to the O/$\overline{\text{E}}$ bit in SMR to invert the parity bit in both transmission and reception.

13.7.3 Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following respects.

- If a parity error is detected during reception, no error signal is output. Since the PER bit in SSR is set by error detection, clear the bit before receiving the parity bit of the next frame.
- During transmission, at least 1 etu is secured as a guard time after the end of the parity bit before the start of the next frame.
- Since the same data is not re-transmitted during transmission, the TEND flag in SSR is set 11.5 etu after transmission start.
- Although the ERS flag in block transfer mode displays the error signal status as in normal smart card interface mode, the flag is always read as 0 because no error signal is transferred.

13.7.4 Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the internal baud rate generator can be used as a communication clock in smart card interface mode. In this mode, the SCI can operate using a basic clock with a frequency of 32, 64, 372, or 256 times the bit rate according to the BCP1 and BCP0 settings (the frequency is always 16 times the bit rate in normal asynchronous mode). At reception, the falling edge of the start bit is sampled using the internal basic clock in order to perform internal synchronization. Receive data is sampled at the 16th, 32nd, 186th and 128th rising edges of the basic clock pulses so that it can be latched at the center of each bit as shown in figure 13.25. The reception margin here is determined by the following formula.

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

M: Reception margin (%)
N: Ratio of bit rate to clock (N = 32, 64, 372, 256)
D: Clock duty (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock rate deviation

Assuming values of F = 0, D = 0.5, and N = 372 in formula (1), the reception margin is determined by the formula below.

$$M = (0.5 - 1/2 \times 372) \times 100 \ [\%] = 49.866\%$$

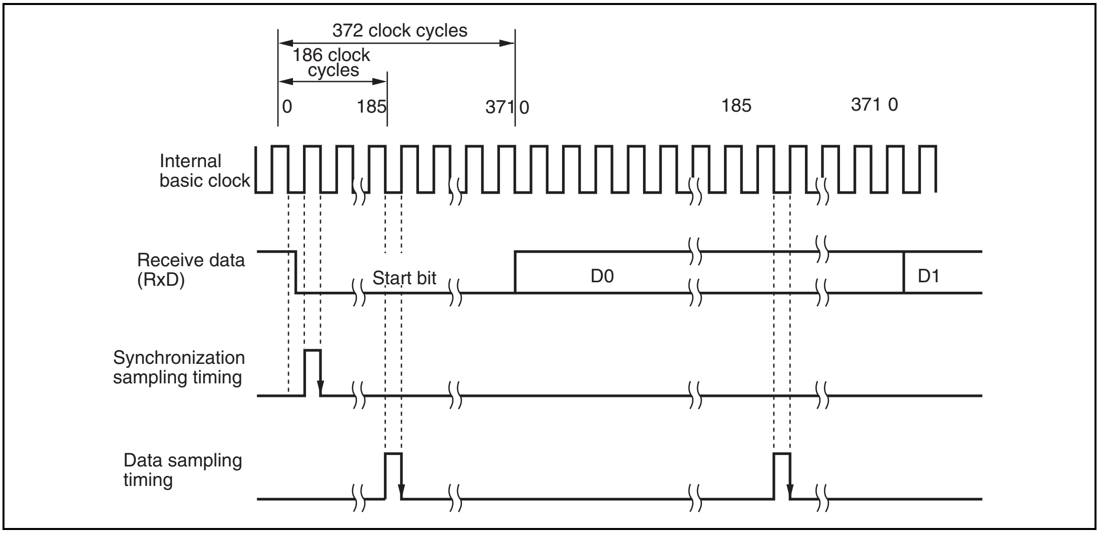


Figure 13.25 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency is 372 Times the Bit Rate)

13.7.5 Initialization

Before starting transmitting and receiving data, initialize the SCI using the following procedure. Initialization is also necessary before switching from transmission to reception and vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Clear the error flags ORER, ERS, and PER in SSR to 0.
3. Set the GM, BLK, O/$\overline{\text{E}}$, BCP1, BCP0, CKS1, and CKS0 bits in SMR appropriately. Also set the PE bit to 1.
4. Set the SMIF, SDIR, and SINV bits in SCMR appropriately. When the SMIF bit is set to 1, the TxD and RxD pins are changed from port pins to SCI pins, placing the pins into high impedance state.
5. Set the value corresponding to the bit rate in BRR.
6. Set the CKE1 and CKE0 bits in SCR appropriately. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0 simultaneously. When the CKE0 bit is set to 1, the SCK pin is allowed to output clock pulses.
7. Set the TIE, RIE, TE, and RE bits in SCR appropriately after waiting for at least 1 bit interval. Setting prohibited the TE and RE bits to 1 simultaneously except for self diagnosis.

To switch from reception to transmission, first verify that reception has completed, and initialize the SCI. At the end of initialization, RE and TE should be set to 0 and 1, respectively. Reception completion can be verified by reading the RDRF flag or PER and ORER flags. To switch from transmission to reception, first verify that transmission has completed, and initialize the SCI. At the end of initialization, TE and RE should be set to 0 and 1, respectively. Transmission completion can be verified by reading the TEND flag.

13.7.6 Serial Data Transmission (Except in Block Transfer Mode)

Data transmission in smart card interface mode (except in block transfer mode) is different from that in normal serial communication interface mode in that an error signal is sampled and data is re-transmitted. Figure 13.26 shows the data re-transfer operation during transmission.

1. If an error signal from the receiving end is sampled after one frame of data has been transmitted, the ERS bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the ERS bit to 0 before the next parity bit is sampled.
2. For the frame in which an error signal is received, the TEND bit in SSR is not set to 1. Data is re-transferred from TDR to TSR allowing automatic data retransmission.
3. If no error signal is returned from the receiving end, the ERS bit in SSR is not set to 1. In this case, one frame of data is determined to have been transmitted including re-transfer, and the TEND bit in SSR is set to 1. Here, a TXI interrupt request is generated if the TIE bit in SCR is set to 1. Writing transmit data to TDR starts transmission of the next data.

Figure 13.28 shows a sample flowchart for transmission. All the processing steps are automatically performed using a TXI interrupt request to activate the DTC. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request when TIE in SCR is set. This activates the DTC by a TXI request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DTC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DTC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, TEND remains as 0, thus not activating the DTC. Therefore, the SCI and DTC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DTC, be sure to set and enable it prior to making SCI settings. For DTC settings, see section 7, Data Transfer Controller (DTC).

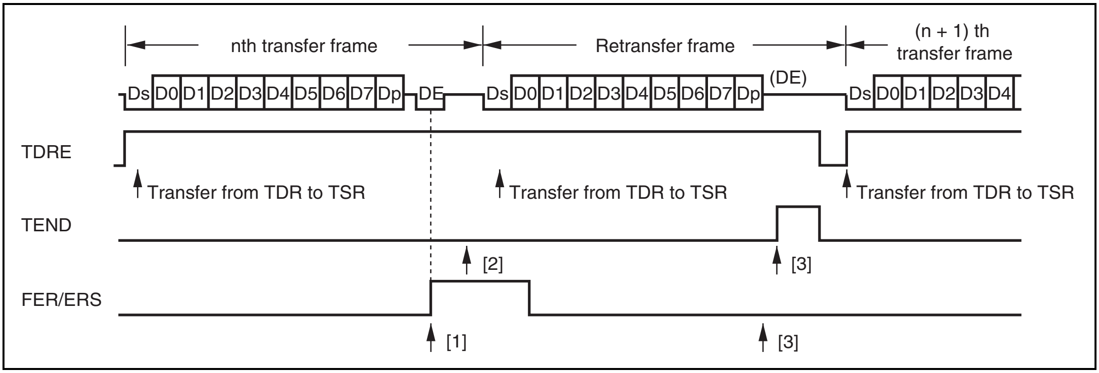
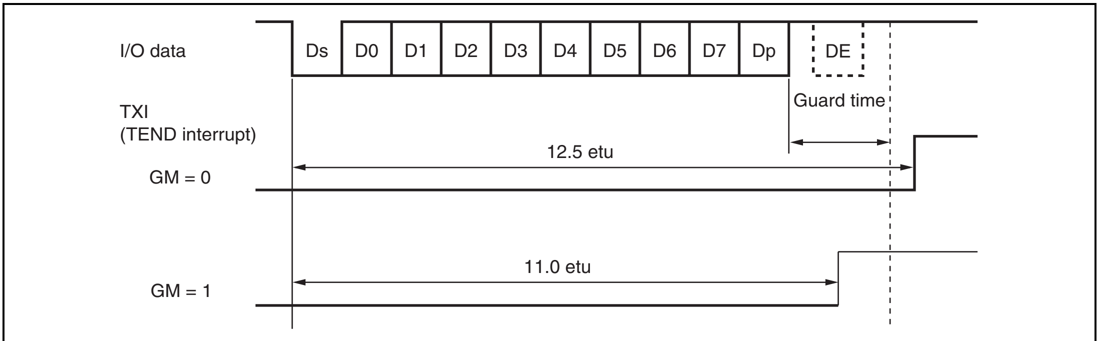


Figure 13.26 Data Re-transfer Operation in SCI Transmission Mode

Note that the TEND flag is set in different timings depending on the GM bit setting in SMR, which is shown in figure 13.27.



[Legend]

Ds: Start bit
D0 to D7: Data bits
Dp: Parity bit
DE: Error signal
etu: Element Time Unit (time taken to transfer one bit)

Figure 13.27 TEND Flag Set Timings during Transmission

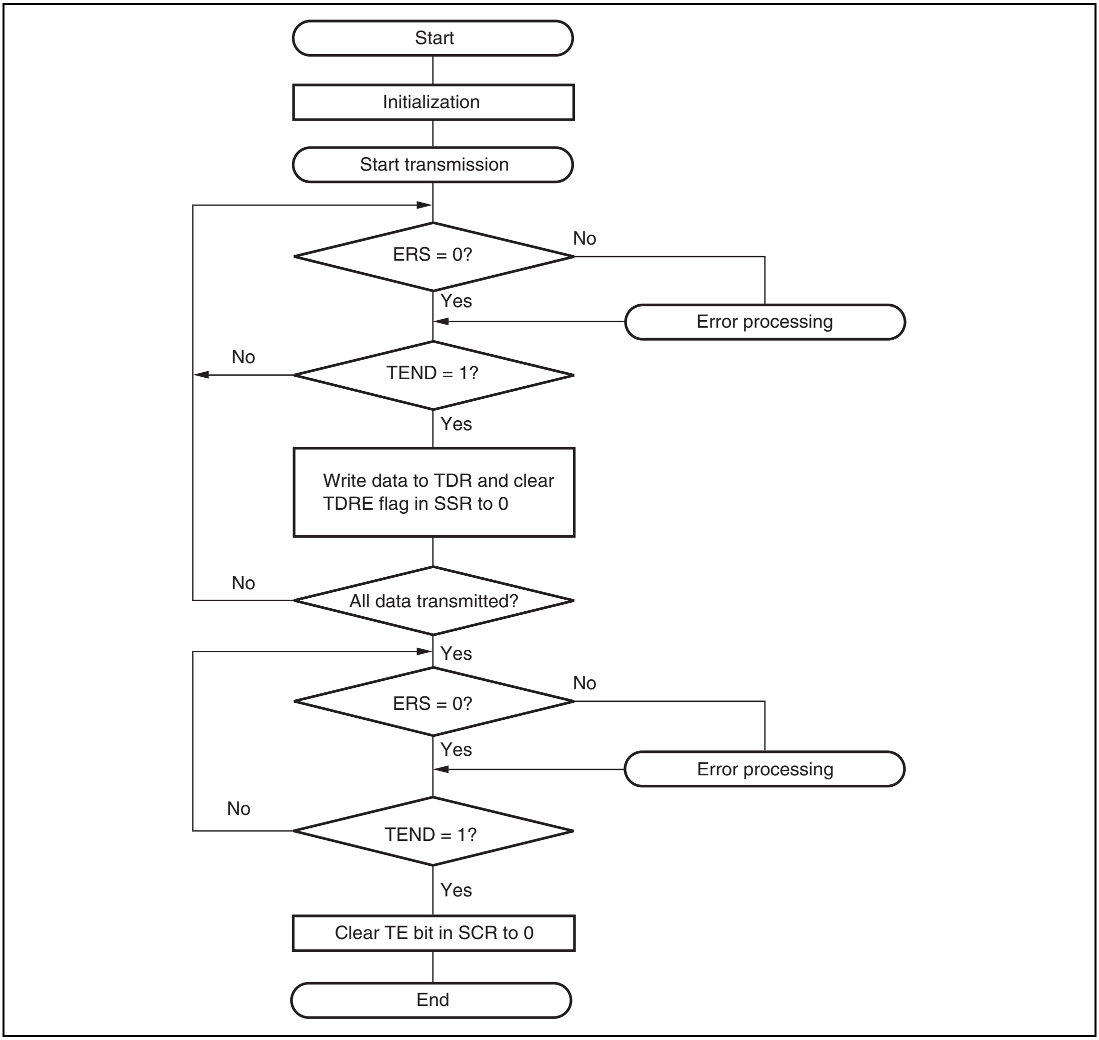


Figure 13.28 Sample Transmission Flowchart

13.7.7 Serial Data Reception (Except in Block Transfer Mode)

Data reception in smart card interface mode is identical to that in normal serial communication interface mode. Figure 13.29 shows the data re-transfer operation during reception.

1. If a parity error is detected in receive data, the PER bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the PER bit to 0 before the next parity bit is sampled.
2. For the frame in which a parity error is detected, the RDRF bit in SSR is not set to 1.
3. If no parity error is detected, the PER bit in SSR is not set to 1. In this case, data is determined to have been received successfully, and the RDRF bit in SSR is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set.

Figure 13.30 shows a sample flowchart for reception. All the processing steps are automatically performed using an RXI interrupt request to activate the DTC. In reception, setting the RIE bit to 1 allows an RXI interrupt request to be generated when the RDRF flag is set to 1. This activates DTC by an RXI request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DTC activate beforehand. The RDRF flag is automatically cleared to 0 at data transfer by DTC. If an error occurs during reception, i.e., either the ORER or PER flag is set to 1, a transmit/receive error interrupt (ERI) request is generated and the error flag must be cleared. If an error occurs, DTC is not activated and receive data is skipped, therefore, the number of bytes of receive data specified in DTC are transferred. Even if a parity error occurs and PER is set to 1 in reception, receive data is transferred to RDR, thus allowing the data to be read.

Note: For operations in block transfer mode, see section 13.4, Operation in Asynchronous Mode.

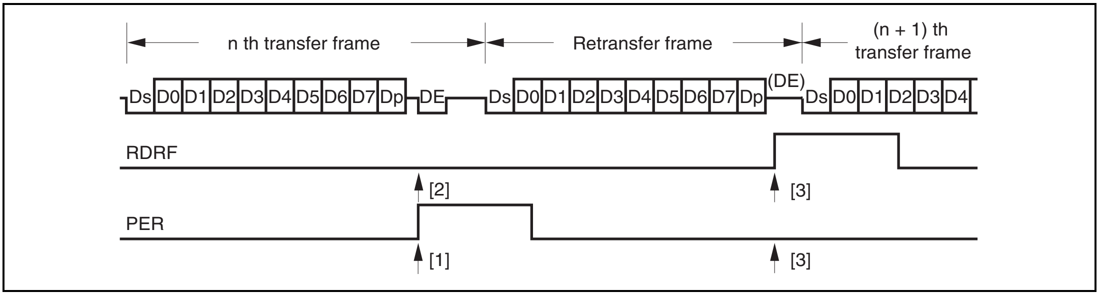

n th transfer frame
Retransfer frame
(n + 1) th transfer frame
Ds D0 D1 D2 D3 D4 D5 D6 D7 Dp DE
Ds D0 D1 D2 D3 D4 D5 D6 D7 Dp (DE)
Ds D0 D1 D2 D3 D4
RDRF
[2]
[3]
PER
[1]
[3]

Figure 13.29 Data Re-transfer Operation in SCI Reception Mode

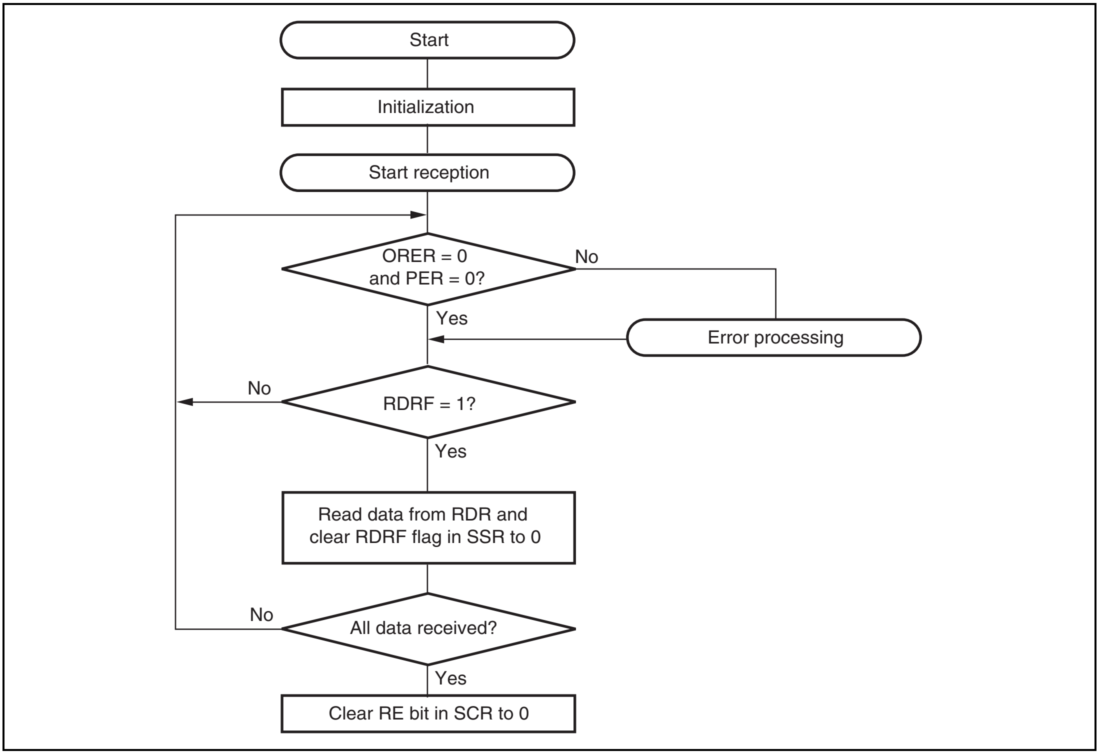


Figure 13.30 Sample Reception Flowchart

13.7.8 Clock Output Control

Clock output can be fixed using the CKE1 and CKE0 bits in SCR when the GM bit in SMR is set to 1. Specifically, the minimum width of a clock pulse can be specified.

Figure 13.31 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 0.

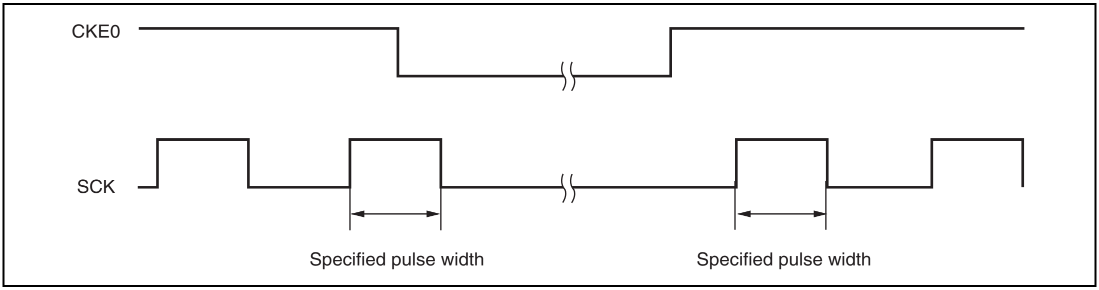


Figure 13.31 Clock Output Fixing Timing

At power-on and transitions to/from software standby mode, use the following procedure to secure the appropriate clock duty ratio.

At Power-On:

To secure the appropriate clock duty ratio simultaneously with power-on, use the following procedure.

1. Initially, port input is enabled in the high-impedance state. To fix the potential level, use a pull-up or pull-down resistor.
2. Fix the SCK pin to the specified output using the CKE1 bit in SCR.
3. Set SMR and SCMR to enable smart card interface mode.
4. Set the CKE0 bit in SCR to 1 to start clock output.

At Transition from Smart Card Interface Mode to Software Standby Mode:

1. Set the port data register (DR) and data direction register (DDR) corresponding to the SCK pins to the values for the output fixed state in software standby mode.
2. Write 0 to the TE and RE bits in SCR to stop transmission/reception. Simultaneously, set the CKE1 bit to the value for the output fixed state in software standby mode.
3. Write 0 to the CKE0 bit in SCR to stop the clock.
4. Wait for one cycle of the serial clock. In the mean time, the clock output is fixed to the specified level with the duty ratio retained.
5. Make the transition to software standby mode.

At Transition from Software Standby Mode to Smart Card Interface Mode:

1. Cancel software standby mode.
2. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty ratio is then generated.

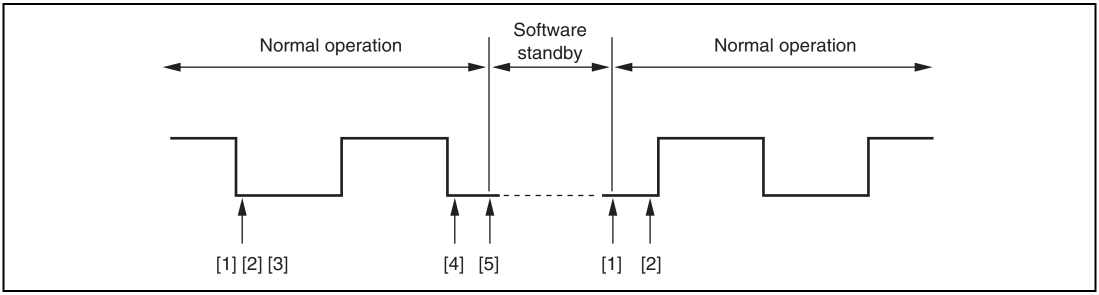


Figure 13.32 Clock Stop and Restart Procedure

13.8 Interrupt Sources

13.8.1 Interrupts in Normal Serial Communication Interface Mode

Table 13.12 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt can activate the DTC to allow data transfer. The TDRE flag is automatically cleared to 0 at data transfer by the DTC.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DTC to allow data transfer. The RDRF flag is automatically cleared to 0 at data transfer by the DTC.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared simultaneously by the TXI interrupt routine, the SCI cannot branch to the TEI interrupt routine later.

Table 13.12 SCI Interrupt Sources

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority
1	ERI1	Receive error	ORER, FER, PER	Not possible	High
	RXI1	Receive data full	RDRF	Possible	↑
	TXI1	Transmit data empty	TDRE	Possible	
	TEI1	Transmit end	TEND	Not possible	
3	ERI3	Receive error	ORER, FER, PER	Not possible	
	RXI3	Receive data full	RDRF	Possible	
	TXI3	Transmit data empty	TDRE	Possible	
	TEI3	Transmit end	TEND	Not possible	Low

13.8.2 Interrupts in Smart Card Interface Mode

Table 13.13 shows the interrupt sources in smart card interface mode. A TEI interrupt request cannot be used in this mode.

Table 13.13 SCI Interrupt Sources

Channel	Name	Interrupt Source	Interrupt Flag	DTC Activation	Priority
1	ERI1	Receive error, error signal detection	ORER, PER, ERS	Not possible	High
	RXI1	Receive data full	RDRF	Possible	↑
	TXI1	Transmit data empty	TEND	Possible	
3	ERI3	Receive error, error signal detection	ORER, PER, ERS	Not possible	
	RXI3	Receive data full	RDRF	Possible	
	TXI3	Transmit data empty	TEND	Possible	Low

Data transmission/reception using the DTC is also possible in smart card interface mode, similar to in the normal SCI mode. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request. This activates the DTC by a TXI interrupt request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DTC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DTC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DTC. Therefore, the SCI and DTC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag in SSR, which is set at error occurrence, is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit in SCR to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DTC, be sure to set and enable the DTC prior to making SCI settings. For DTC settings, see section 7, Data Transfer Controller (DTC).

In reception, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. This activates the DTC by an RXI interrupt request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DTC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DTC. If an error occurs, the RDRF flag is not set but the error flag is set. Therefore, the DTC is not activated and an ERI interrupt request is issued to the CPU instead; the error flag must be cleared.

13.9 Usage Notes

13.9.1 Module Stop Mode Setting

SCI operation can be disabled or enabled using the module stop control register. The initial setting is for SCI operation to be halted. Register access is enabled by clearing module stop mode. For details, see section 22, Power-Down Modes.

13.9.2 Break Detection and Processing

When framing error detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag 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.

13.9.3 Mark State and Break Sending

When the TE bit in SCR is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR of the port. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line at mark state until TE is set to 1, set both DDR and DR to 1. Since the TE bit is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set DDR to 1 and DR to 0, and then clear the TE bit to 0. When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

13.9.4 Receive Error Flags and Transmit Operations (Clock Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) in SSR is set to 1, even if the TDRE flag in SSR is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that the receive error flags cannot be cleared to 0 even if the RE bit in SCR is cleared to 0.

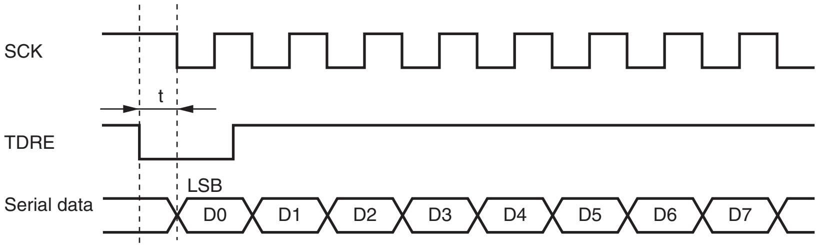
13.9.5 Relation between Writing to TDR and TDRE Flag

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.

13.9.6 Restrictions on Using DTC

When the external clock source is used as a synchronization clock, update TDR by the DTC and wait for at least five ϕ clock cycles before allowing the transmit clock to be input. If the transmit clock is input within four clock cycles after TDR modification, the SCI may malfunction (figure 13.33).

When using the DTC to read RDR, be sure to set the receive end interrupt source (RXI) as a DTC activation source.



Note: When external clock is supplied, t must be more than four clock cycles.

Figure 13.33 Sample Transmission using DTC in Clock Synchronous Mode

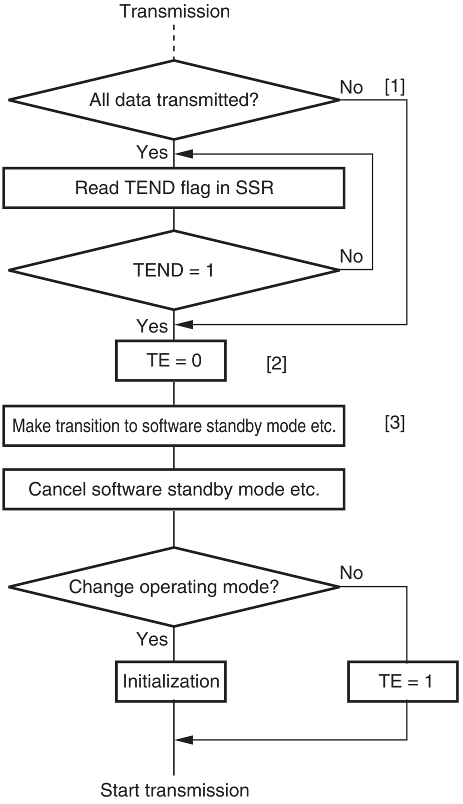
13.9.7 SCI Operations during Mode Transitions

Transmission: Before making the transition to module stop or software standby, stop all transmit operations (TE = TIE = TEIE = 0). TSR, TDR, and SSR are reset. The states of the output pins during each mode depend on the port settings, and the pins output a high-level signal after mode cancellation. If the transition is made during data transmission, the data being transmitted will be undefined.

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

Figure 13.34 shows a sample flowchart for mode transition during transmission. Figures 13.35 and 13.36 show the pin states during transmission.

Before making the transition from the transmission mode using DTC transfer to module stop or software standby, stop all transmit operations (TE = TIE = TEIE = 0). Setting TE and TIE to 1 after mode cancellation generates a TXI interrupt request to start transmission using the DTC.



[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; however, if the DTC has been initiated, the data remaining in DTC RAM will be transmitted when TE and TIE are set to 1.

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

[3] Module stop mode is included.

Figure 13.34 Sample Flowchart for Mode Transition during Transmission

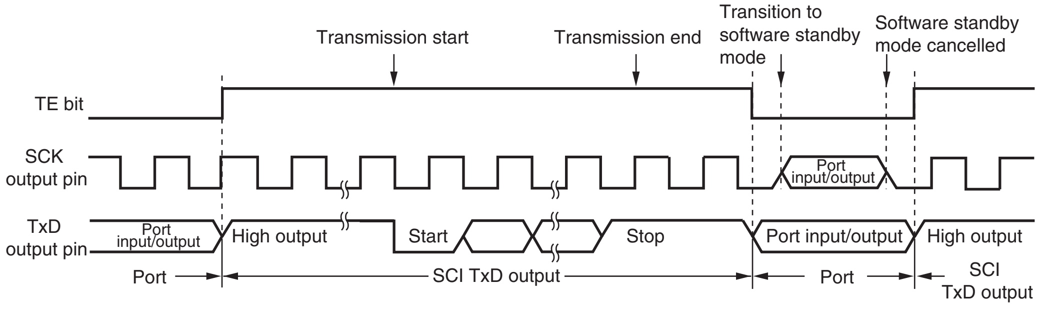
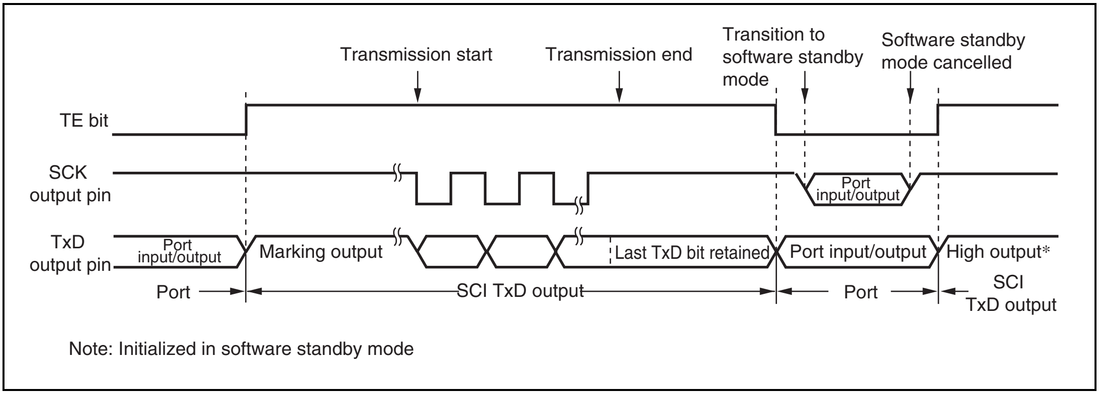


Figure 13.35 Pin States during Transmission in Asynchronous Mode (Internal Clock)



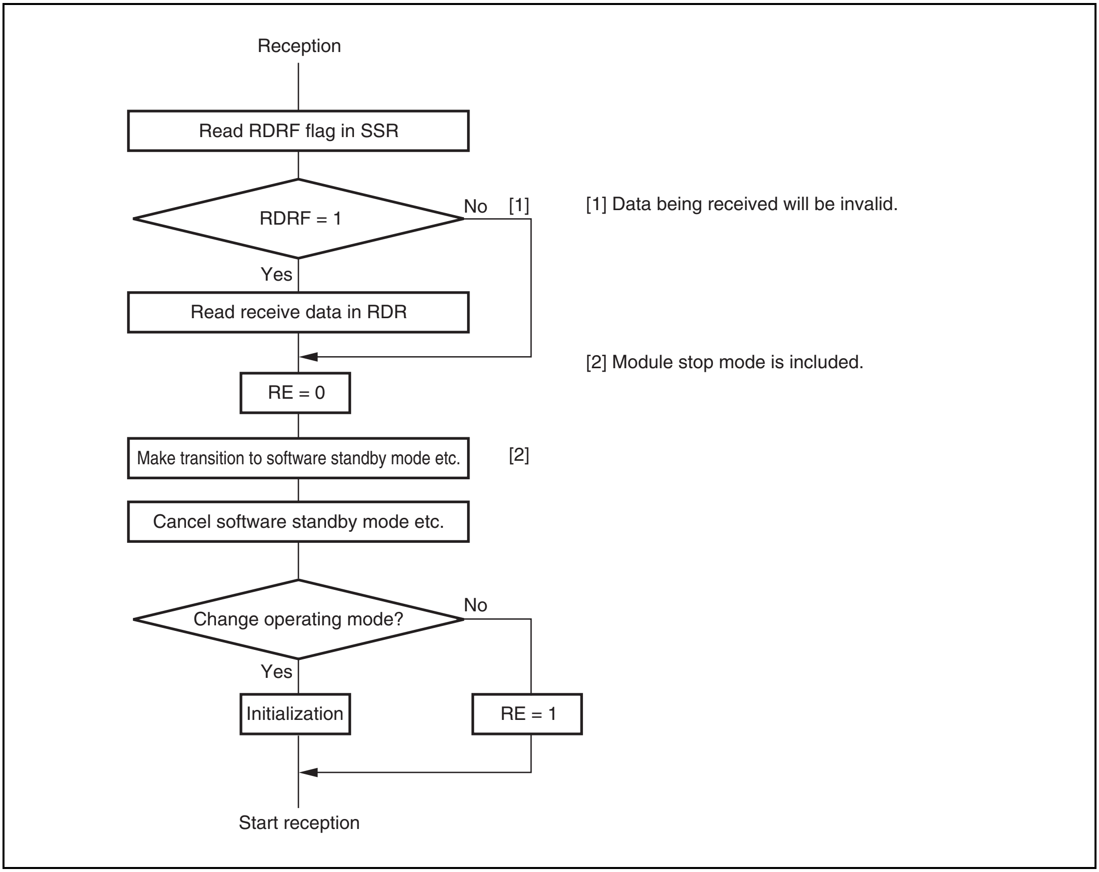
Note: Initialized in software standby mode

Figure 13.36 Pin States during Transmission in Clock Synchronous Mode (Internal Clock)

Reception: Before making the transition to module stop or software standby, stop reception (RE = 0). RSR, RDR, and SSR are reset. If transition is made during data reception, the data being received will be invalid.

To receive data in the same reception mode after mode cancellation, set RE to 1, and then start reception. To receive data in a different reception mode, initialize the SCI first.

Figure 13.37 shows a sample flowchart for mode transition during reception.



[1] Data being received will be invalid.

[2] Module stop mode is included.

Figure 13.37 Sample Flowchart for Mode Transition during Reception

13.9.8 Notes on Switching from SCK Pins to Port Pins

When SCK pins are switched to port pins after transmission has completed, pins are enabled for port output after outputting a low pulse of half a cycle as shown in figure 13.38.

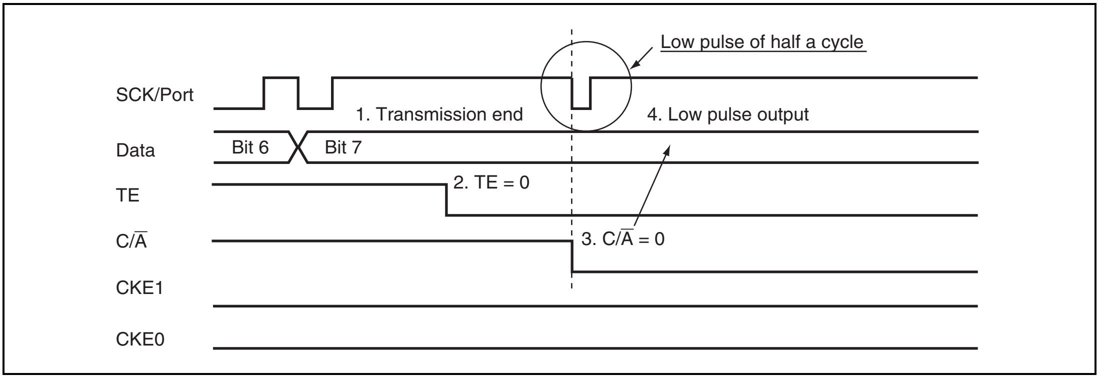


Figure 13.38 Switching from SCK Pins to Port Pins

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

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

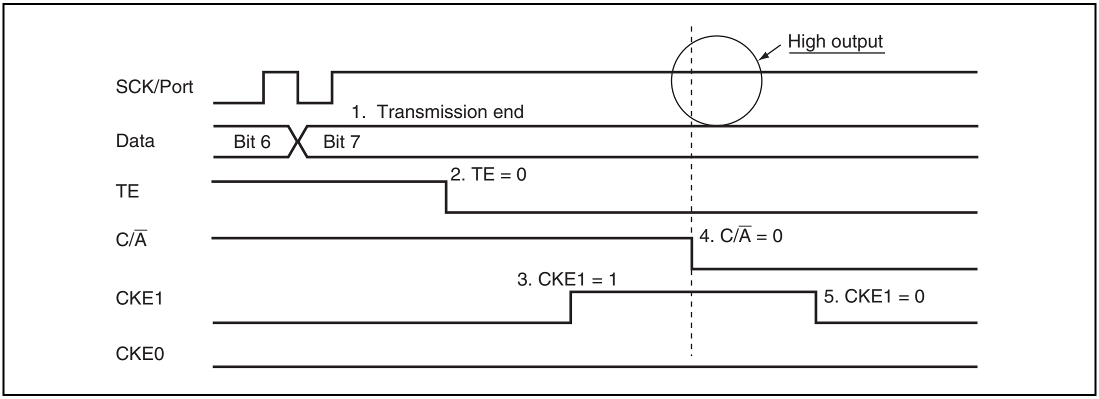


Figure 13.39 Prevention of Low Pulse Output at Switching from SCK Pins to Port Pins

Section 14 CRC Operation Circuit (CRC)

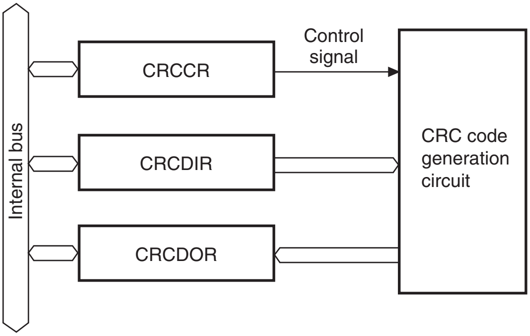
This LSI has a cyclic redundancy check (CRC) operation circuit to enhance the reliability of data transfer in high-speed communications, etc. The CRC operation circuit detects errors in data blocks.

14.1 Features

The features of the CRC operation circuit are listed below.

- CRC code generated for any desired data length in an 8-bit unit
- CRC operation executed on eight bits in parallel
- One of three generating polynomials selectable
- CRC code generation for LSB-first or MSB-first communication selectable

Figure 14.1 shows a block diagram of the CRC operation circuit.



[Legend]
CRCCR: CRC control register
CRCDIR: CRC data input register
CRCDOR: CRC data output register

Figure 14.1 Block Diagram of CRC Operation Circuit

14.2 Register Descriptions

The CRC operation circuit has the following registers.

- CRC control register (CRCCR)
- CRC data input register (CRCDIR)
- CRC data output register (CRCDOR)

14.2.1 CRC Control Register (CRCCR)

CRCCR initializes the CRC operation circuit, switches the operation mode, and selects the generating polynomial.

Bit	Bit Name	Initial Value	R/W	Description
7	DORCLR	0	W	CRCDOR Clear Setting this bit to 1 clears CRCDOR to H'0000.
6 to 3	—	All 0	R	Reserved The initial value should not be changed.
2	LMS	0	R/W	CRC Operation Switch Selects CRC code generation for LSB-first or MSB-first communication. 0: Performs CRC operation for LSB-first communication. The lower byte (bits 7 to 0) is first transmitted when CRCDOR contents (CRC code) are divided into two bytes to be transmitted in two parts. 1: Performs CRC operation for MSB-first communication. The upper byte (bits 15 to 8) is first transmitted when CRCDOR contents (CRC code) are divided into two bytes to be transmitted in two parts.
1	G1	0	R/W	CRC Generating Polynomial Select
0	G0	0	R/W	These bits select the polynomial. 00: Reserved 01: $X^8 + X^2 + X + 1$ 10: $X^{16} + X^{15} + X^2 + 1$ 11: $X^{16} + X^{12} + X^5 + 1$

14.2.2 CRC Data Input Register (CRCDIR)

CRCDIR is an 8-bit readable/writable register, to which the bytes to be CRC-operated are written. The result is obtained in CRCDOR.

14.2.3 CRC Data Output Register (CRCDOR)

CRCDOR is a 16-bit readable/writable register that contains the result of CRC operation when the bytes to be CRC-operated are written to CRCDIR after CRCDOR is cleared. When the CRC operation result is additionally written to the bytes to which CRC operation is to be performed, the CRC operation result will be H'0000 if the data contains no CRC error. When bits 1 and 0 in CRCCR (G1 and G0 bits) are set to 0 and 1, respectively, the lower byte of this register contains the result.

14.3 CRC Operation Circuit Operation

The CRC operation circuit generates a CRC code for LSB-first/MSB-first communications. An example in which a CRC code for hexadecimal data H'F0 is generated using the $X^{16} + X^{12} + X^5 + 1$ polynomial with the G1 and G0 bits in CRCCR set to B'11 is shown below.

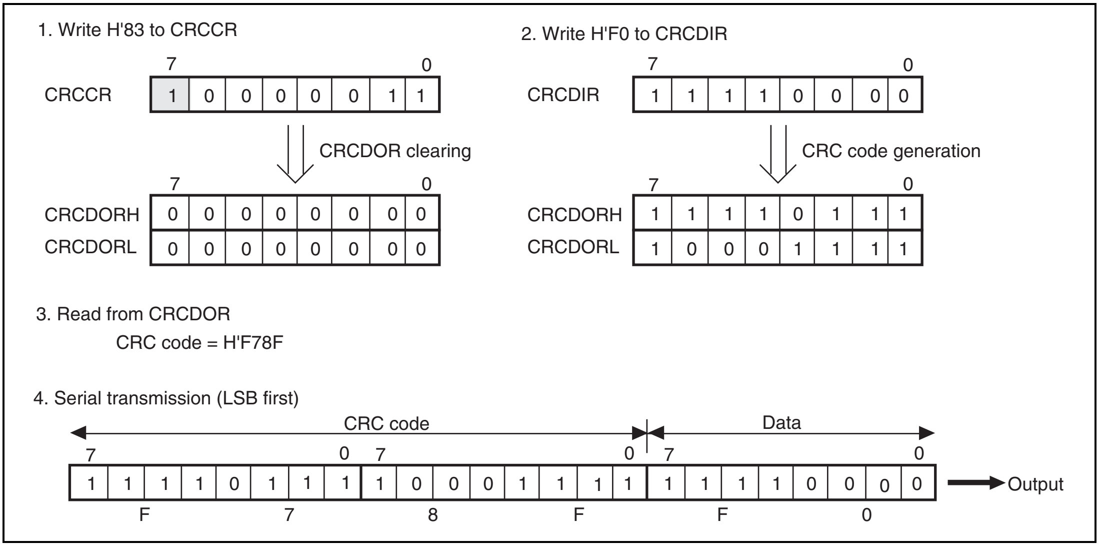


Figure 14.2 LSB-First Data Transmission

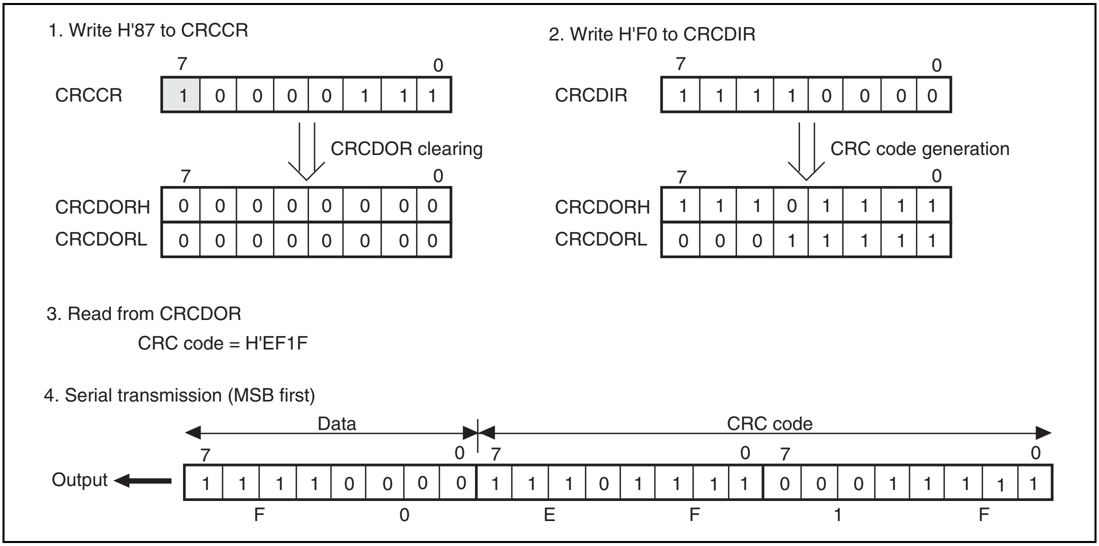


Figure 14.3 MSB-First Data Transmission

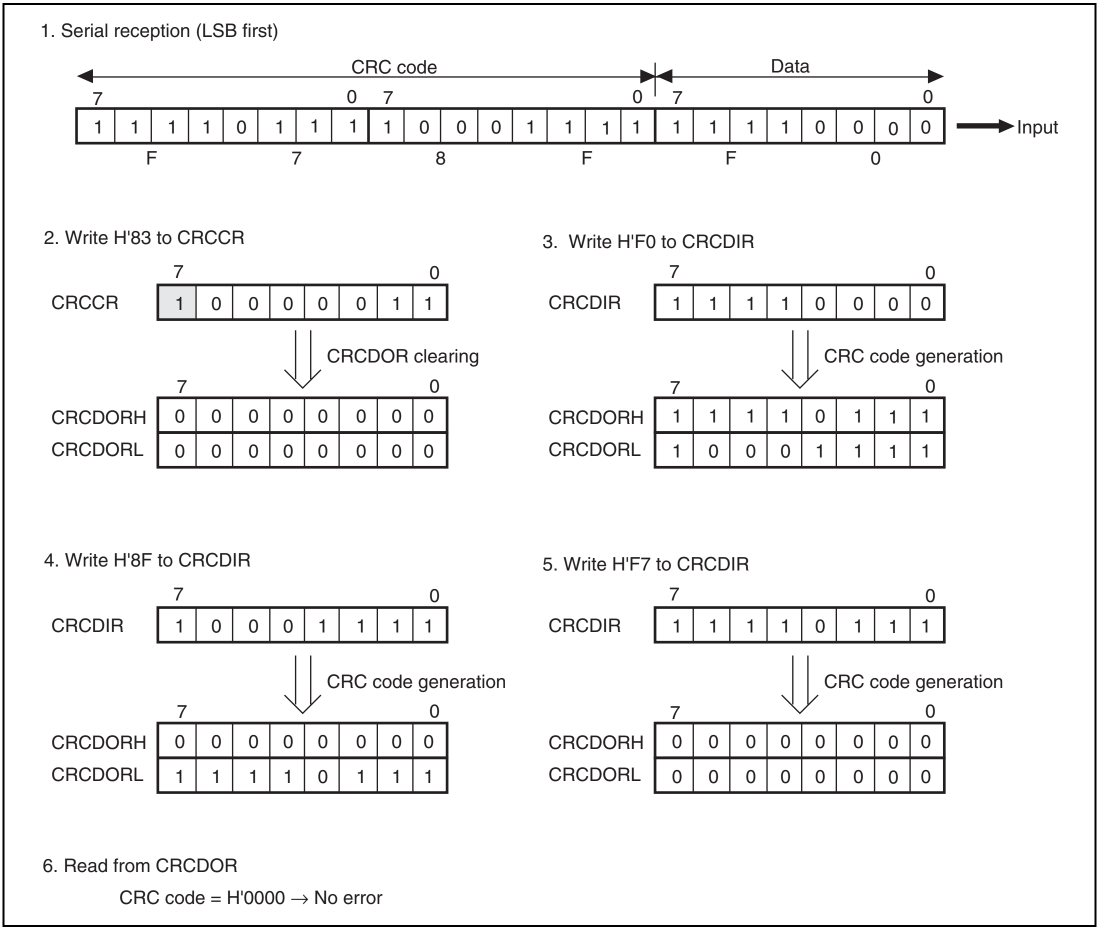


Figure 14.4 LSB-First Data Reception

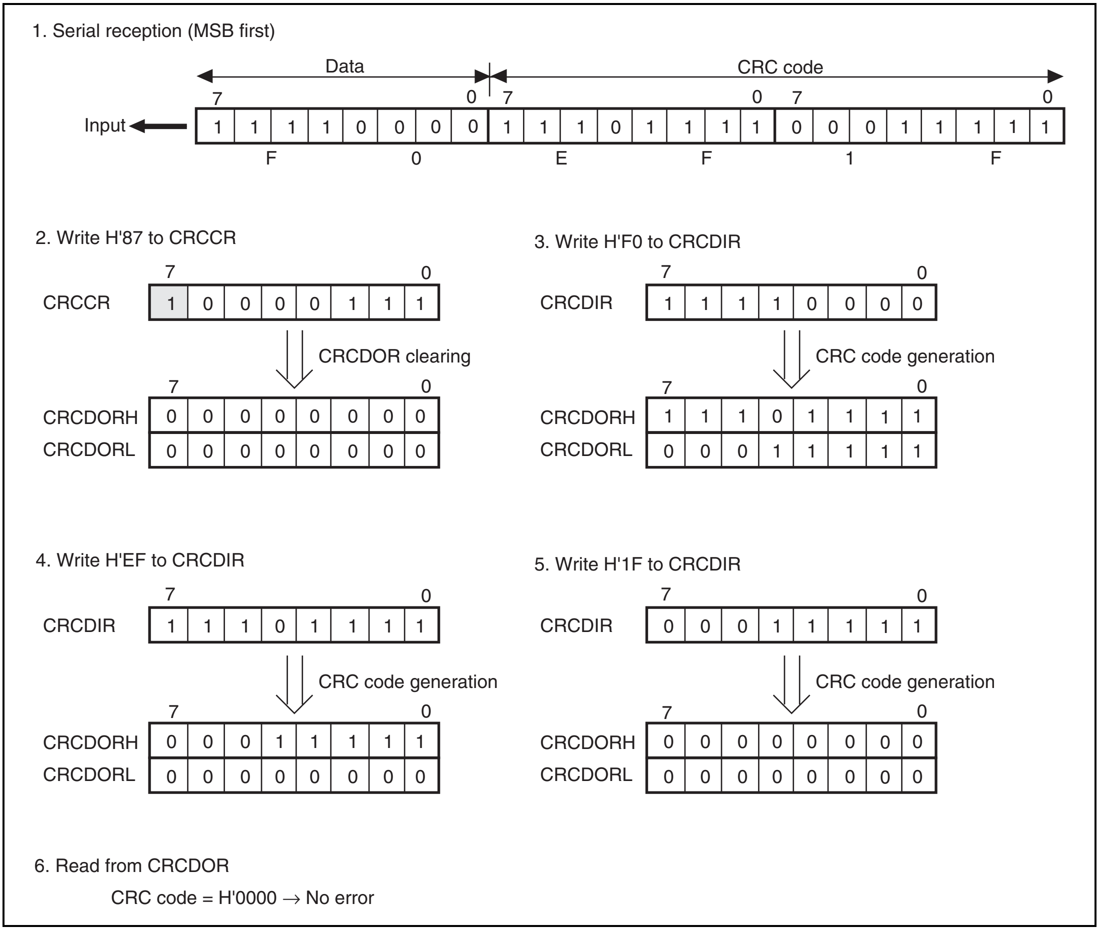


Figure 14.5 MSB-First Data Reception

14.4 Note on CRC Operation Circuit

Note that the sequence to transmit the CRC code differs between LSB-first transmission and MSB-first transmission.

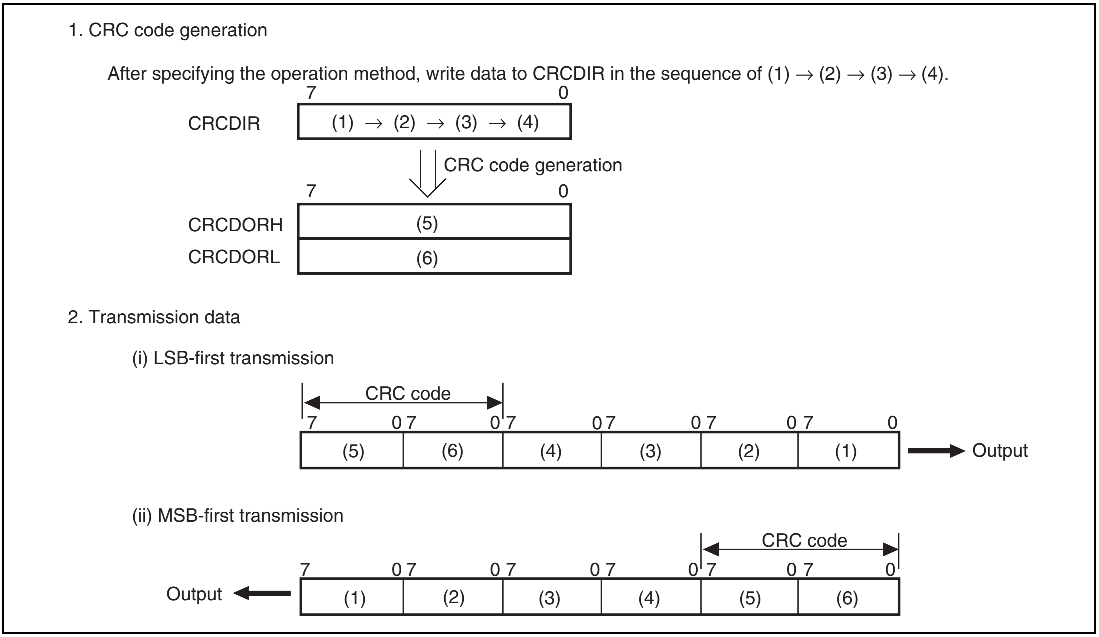


Figure 14.6 LSB-First and MSB-First Transmit Data

Section 15 I^{2}C Bus Interface (IIC)

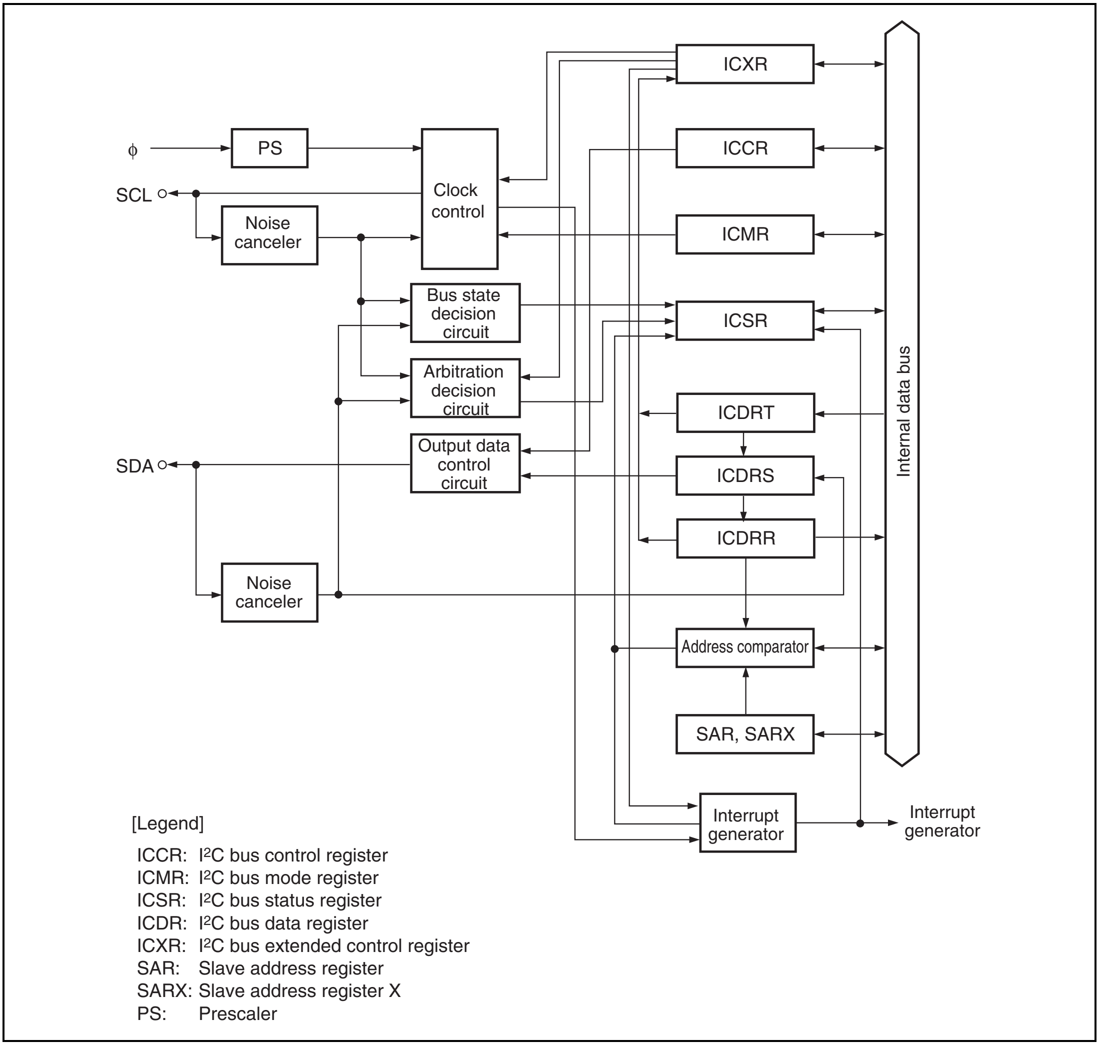
This LSI has four-channels of I^{2}C bus interface (IIC). The I^{2}C bus interface conforms to and provides a subset of the Philips I^{2}C bus (inter-IC bus) interface functions. The register configuration that controls the I^{2}C bus differs partly from the Philips configuration, however.

15.1 Features

- Selection of addressing format or non-addressing format
  - I^{2}C bus format: addressing format with acknowledge bit, for master/slave operation
  - Clocked synchronous serial format: non-addressing format without acknowledge bit, for master operation only
- 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 acknowledge output levels when receiving (I^{2}C bus format)
- Automatic loading of acknowledge bit when transmitting (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)
  - Arbitration loss
  - Start condition detection (in master mode)
  - Stop condition detection (in slave mode)
- Selection of 32 internal clocks (in master mode)
- Direct bus drive

— Pins—SCL0 to SCL3 and SDA0 to SDA3 — (normally NMOS push-pull outputs) function as NMOS open-drain outputs when the bus drive function is selected.

Figure 15.1 shows a block diagram of the I²C bus interface. Figure 15.2 shows an example of I/O pin connections to external circuits. Since I²C 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 24, Electrical Characteristics.



[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 15.1 Block Diagram of I²C Bus Interface

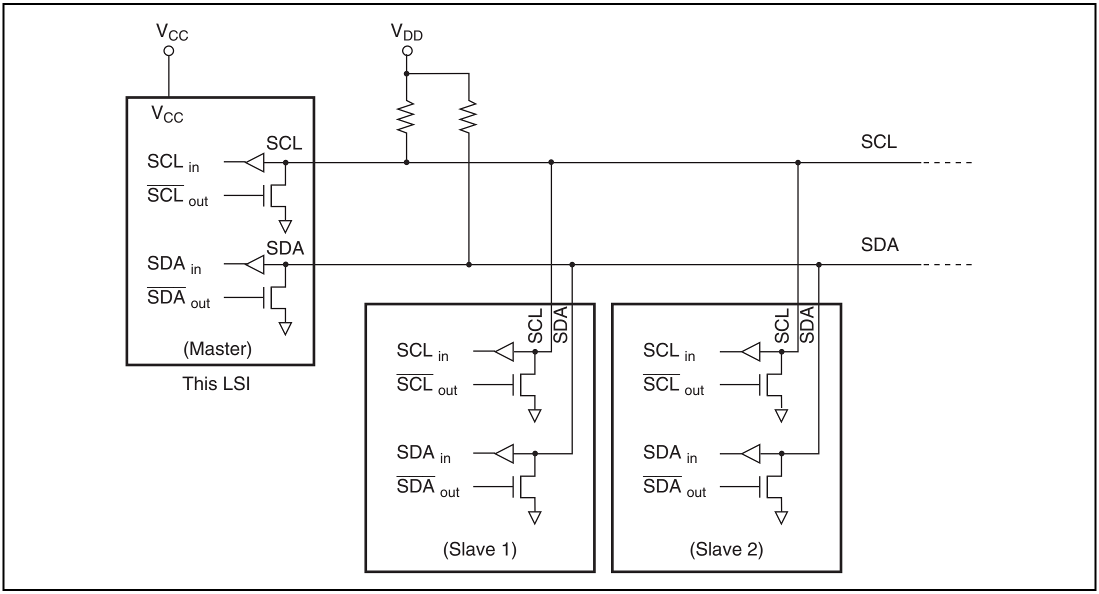


Figure 15.2 I²C Bus Interface Connections (Example: This LSI as Master)

15.2 Input/Output Pins

Table 15.1 summarizes the input/output pins used by the I^{2}C bus interface.

Table 15.1 Pin Configuration

Channel	Symbol*	Input/Output	Function
0	SCL0	Input/Output	Clock input/output pin of channel IIC_0
	SDA0	Input/Output	Data input/output pin of channel IIC_0
1	SCL1	Input/Output	Clock input/output pin of channel IIC_1
	SDA1	Input/Output	Data input/output pin of channel IIC_1
2	SCL2	Input/Output	Clock input/output pin of channel IIC_2
	SDA2	Input/Output	Data input/output pin of channel IIC_2
3	SCL3	Input/Output	Clock input/output pin of channel IIC_3
	SDA3	Input/Output	Data input/output pin of channel IIC_3

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

15.3 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.

- I^{2}C bus data register (ICDR)
- Slave address register (SAR)
- Second slave address register (SARX)
- I^{2}C bus mode register (ICMR)
- I^{2}C bus transfer rate select register (IICX3)
- I^{2}C bus control register (ICCR)
- I^{2}C bus status register (ICSR)
- I^{2}C bus extended control register (ICXR)
- I^{2}C SMbus control register (ICSMBCR)

15.3.1 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 divided internally into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). Data transfers among the three registers are performed automatically in accordance with changes in the bus state, and they affect the status of internal flags such as ICDRE and ICDRF.

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 IIC is in transmit mode (TRS=1) and the next data is in ICDRT (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 IIC 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 IIC 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 IIC 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.

15.3.2 Slave Address Register (SAR)

SAR sets the slave address and selects the communication format. When the LSI is in slave mode with the I^{2}C bus format selected, if 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	All 0	R/W	Slave Addresses 6 to 0
6	SVA5			Set a slave address.
5	SVA4			
4	SVA3			
3	SVA2			
2	SVA1			
1	SVA0			
0	FS	0	R/W	Format Select Selects the communication format together with the FSX bit in SARX. Refer to table 15.2. This bit should be set to 0 when general call address recognition is performed.

15.3.3 Second Slave Address Register (SARX)

SARX sets the second slave address and selects the communication format. In slave mode, transmit/receive operations by the DTC are possible when the received address matches the second slave address. When the LSI is in slave mode with the I^{2}C bus format selected, if 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	All 0	R/W	Second Slave Addresses 6 to 0
6	SVAX5			Set the second slave address.
5	SVAX4			
4	SVAX3			
3	SVAX2			
2	SVAX1			
1	SVAX0			
0	FSX	1	R/W	Format Select X Selects the communication format together with the FS bit in SAR. Refer to table 15.2.

Table 15.2 Transfer 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 acknowledge bit
- Clocked synchronous serial format: non-addressing format without acknowledge bit, for master mode only

15.3.4 I²C 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 I²C bus format is used.
6	WAIT	0	R/W	Wait Insertion Bit This bit is valid only in master mode with the I²C 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 15.4.7, IRIC Setting Timing and SCL Control.
5 4 3	CKS2 CKS1 CKS0	All 0	R/W	Transfer Clock Select These bits are used only in master mode. These bits select the required transfer rate, together with the IICX3 (channel 3) bit in IICX3, the IICX2 (channel 2), IICX1 (channel 1), and IICX0 (channel 0) bits in STCR. Refer to table 15.3.

Bit	Bit Name	Initial Value	R/W	Description
2	BC2	All 0	R/W	Bit Counter
1	BC1			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 B'000, the setting should be made while the SCL line is low.
0	BC0			The bit counter is initialized to B'000 when a start condition is detected. The value returns to B'000 at the end of a data transfer.
				I²C Bus Format　Clocked Synchronous Serial Mode
				B'000: 9 bits　B'000: 8 bits
				B'001: 2 bits　B'001: 1 bits
				B'010: 3 bits　B'010: 2 bits
				B'011: 4 bits　B'011: 3 bits
				B'100: 5 bits　B'100: 4 bits
				B'101: 6 bits　B'101: 5 bits
				B'110: 7 bits　B'110: 6 bits
				B'111: 8 bits　B'111: 7 bits

15.3.5 I^{2}C Bus Transfer Rate Select Register (IICX3)

IICX3 selects the IIC transfer rate clock and sets the transfer rate of IIC channel 3.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	—	—	Reserved These bits cannot be modified. The read values are undefined.
3	TCSS	0	R/W	Transfer Rate Clock Source Select This bit selects a clock rate to be applied to the I^{2}C bus transfer rate. 0: ϕ/2 1: ϕ/4
2, 1	—	—	—	Reserved These bits cannot be modified. The read values are undefined.
0	IICX3			IIC Transfer Rate Select 3 These bits are used to control IIC_3 operation. These bits select the transfer rate in master mode, together with the CKS2 to CKS0 bits in ICMR. For the transfer rate, see table 15.3.

Table 15.3 I^{2}C bus Transfer Rate (1)

- TCSS = 0

STCR/	ICMR					
IICX3	Bit 5	Bit 4	Bit 3		Transfer Rate (MHz)	
IICXn	CKS2	CKS1	CKS0	Clock	ϕ = 20 MHz	ϕ = 25 MHz
0	0	0	0	ϕ/28	714.3*	892.9*
			1	ϕ/40	500.0*	625.0*
		1	0	ϕ/48	416.7*	520.8*
			1	ϕ/64	312.5	390.6
	1	0	0	ϕ/80	250.0	312.5
			1	ϕ/100	200.0	250.0
		1	0	ϕ/112	178.6	223.2
			1	ϕ/128	156.3	195.3
1	0	0	0	ϕ/56	357.1	446.4*
			1	ϕ/80	250.0	312.5
		1	0	ϕ/96	208.3	260.4
			1	ϕ/128	156.3	195.3
	1	0	0	ϕ/160	125.0	156.3
			1	ϕ/200	100.0	125.0
		1	0	ϕ/224	89.3	111.6
			1	ϕ/256	78.1	97.7

Note: * The correct operation cannot be guaranteed since the value is outside the I^{2}C bus interface specifications (high-speed mode: max. 400 kHz).
(n = 0 to 3)

Table 15.3 I2C bus Transfer Rate (2)

- TCSS = 1

STCR/	ICMR					
IICX3	Bit 5	Bit 4	Bit 3		Transfer Rate (MHz)	
IICXn	CKS2	CKS1	CKS0	Clock	ϕ = 20 MHz	ϕ = 25 MHz
0	0	0	0	ϕ/56	357.1	446.4*
			1	ϕ/80	250.0	312.5
		1	0	ϕ/96	208.3	260.4
			1	ϕ/128	156.3	195.3
	1	0	0	ϕ/160	125.0	156.3
			1	ϕ/200	100.0	125.0
		1	0	ϕ/224	89.3	111.6
			1	ϕ/256	78.1	97.7
1	0	0	0	ϕ/112	178.6	223.2
			1	ϕ/160	125.0	156.3
		1	0	ϕ/190	104.2	130.2
			1	ϕ/256	78.1	97.7
	1	0	0	ϕ/320	62.5	78.1
			1	ϕ/400	50.0	62.5
		1	0	ϕ/448	44.6	55.8
			1	ϕ/512	39.1	48.8

Note: * The correct operation cannot be guaranteed since the value is outside the I^{2}C bus interface specifications (high-speed mode: max. 400 kHz).
(n = 0 to 3)

15.3.6 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 and reception, they are connected to the SCL and SDA pins, and the I^{2}C bus can be driven. 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 of the I^{2}C bus format. In slave receive mode with I^{2}C bus format, the R/$\overline{W}$ bit in the first frame immediately after the start condition automatically sets these bits in receive mode or transmit mode 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	R/W	
3	ACKE	0	R/W	Acknowledge Bit Decision 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 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 conditions] • 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^{2}C bus interface must be set in master transmit mode before the issue of a start condition. Set MST to 1 and TRS to 1 before writing 1 in BBSY and 0 in SCP. The BBSY flag can be read to check whether the I^{2}C bus (SCL, SDA) is busy or free.

Note: * If the BBSY bit is written to, the value of the flag is not changed.

Bit	Bit Name	Initial 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 and the WAIT bit in ICMR. See section 15.4.7, 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) • When a slave address is received after bus mastership is lost • 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 mastership 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 clock) • When the general call address is detected (when the 0 is received for R/$\overline{W}$ bit, and 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) • When 1 is received as an acknowledge bit while the ACKE bit is 1 (when the ACKB bit is set to 1) • When a stop condition is detected while the STOPIM bit is 0 (when the STOP or ESTP flag in ICSR is set to 1)

Bit	Bit Name	Initial Value	R/W	Description
1	IRIC	0	R/(W)*[1]	At the end of data transfer in clock synchronous serial format (rise of the 8th transmit/receive clock) When a start condition is detected with serial format selected When a condition occurs in which the ICDRE or ICDRF flag is set to 1. • When a start condition is detected in transmit mode (when a start condition is detected and the ICDRE flag is set to 1) • When transmitting the data in the ICDR register buffer (when data is transferred from ICDRT to ICDRS in transmit mode and the ICDRE flag is set to 1, or 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 • When ICDR is accessed by DTC * (This may not be a clearing condition. For details, see the description of the DTC operation on the next page.

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

When the DTC is used, IRIC is cleared automatically and transfer can be performed continuously without CPU intervention.

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 (the DTC start request flag) is not set at the end of a data transfer up to detection of a retransmission start condition or stop condition after a slave address (SVA) or general call address match in I^{2}C bus format slave mode.

Even when the IRIC flag and IRTR flag are set, the ICDRE or ICDRF flag may not be set. The IRIC and IRTR flags are not cleared at the end of the specified number of transfers in continuous transfer using the DTC. The ICDRE or ICDRF flag is cleared, however, since the specified number of ICDR reads or writes have been completed.

Tables 15.4 and 15.5 show the relationship between the flags and the transfer states.

Table 15.4 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
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 15.5 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	SAR 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	—	—	—	—	0	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

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.

15.3.7 I^{2}C Bus Status Register (ICSR)

ICSR consists of status flags. Refer to tables 15.4 and 15.5 as well.

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 is completed. [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 for which DTC activation is possible. When the IRTR flag is set to 1, the IRIC flag is also set to 1 at the same time. [Setting conditions] I^{2}C bus format slave mode: • When the ICDRE or ICDRF flag in ICDR is set to 1 when AASX = 1 I^{2}C bus format master mode or clocked synchronous serial format mode: • 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 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^{2}C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition matches bits SVA6 to SVA0 in SAR, or if the general call address (H'00) is detected. [Setting condition] When the slave address or general call address (one frame including a R/$\overline{\text{W}}$ 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^{2}C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition is the general call address (H'00). [Setting condition] When the general call address (one frame including a R/$\overline{\text{W}}$ 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.

15.3.8 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 after data has been received successfully while ICDRF flag is 0; thus disabling the next data to be transferred. 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 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 in I^2C bus format or serial format. • When data is transferred from ICDRT to ICDRS. 1. When data is transmitted completely while ICDRE = 0 (at the rise of the 9th clock pulse). 2. When data is written to ICDR completely in transmit mode after data was transmitted while ICDRE = 1. [Clearing conditions] • When data is written to ICDR (ICDRT). • When the stop condition is detected in I^2C bus format or serial format. • When 0 is written to the ICE bit. Note that if the ACKE bit is set to 1 in I^2C bus format thus enabling acknowledge bit decision, ICDRE is not set when data is transmitted completely 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, ICDRF 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 request 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: If the SDA pin state disagrees with the data that I^2C bus interface outputs at the rise of SCL and the SCL pin is driven low by another device. 1: If the SDA pin state disagrees with the data that I^2C bus interface outputs at the rise of SCL and 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	These bits cancel some restrictions on usage. For details, refer to section 15.6, Usage Notes. 00: Restrictions on operation remaining in effect 01: Setting prohibited 10: Setting prohibited 11: Restrictions on operation canceled

15.3.9 I^{2}C SMBus Control Register (ICSMBCR)

ICSMBCR is used to support the System Management Bus (SMBus) specifications. To support the SMBus specification, SDA output data hold time should be specified in the range of 300 ns to 1000 ns. Table 15.6 shows the relationship between the ICSMBCR setting and output data hold time.

When the SMBus is not supported, the initial value should not be changed. ICSMBCR is enabled to access when bit MSTP4 is cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7 6	— —	— —	— —	Reserved These bits cannot be modified. The read values are undefined.
5 4 3 2	SMB3E SMB2E SMB1E SMB0E	All 0	R/W	SMBus Enable These bits enable/disable to support the SMBus, combining with bits FSEL1 and FSEL0. The SMB3E bit controls IIC_3, the SMB2E bit controls IIC_2, the SMB1E bit controls IIC_1, the SMB0E bit controls IIC_0. 0: Disables to support the SMBus 1: Enables to support the SMBus
1 0	FSEL1 FSEL0	0 0	R/W R/W	Frequency Selection These bits must be specified to match the system clock frequency in order to support the SMBus. For details of the setting, see table 15.7.

Table 15.6 Output Data Hold Time

				Output Data Hold Time (ns)	
SMBnE	FSEL1	FSEL0	Min./Max.	ϕ = 20 MHz	ϕ = 25 MHz
0	—	—	Min.	100*	80*
			Max.	150*	120*
1	0	0	Min.	150*	120*
			Max.	250*	200*
		1	Min.	200*	160*
			Max.	350	280*
	1	0	Min.	300	240*
			Max.	550	440
		1	Min.	500	400
			Max.	950	760

[Legend] n = 0 to 3

Note: * Since the value is outside the SMBus specification, it should not be set.

Table 15.7 ISCMBCR Setting

System Clock	SMBnE	FSEL1	FSEL0
20 MHz	1	1	0
20 to 25 MHz	1	1	1

[Legend] n = 0 to 3

15.4 Operation

15.4.1 I²C Bus Data Format

The I²C bus interface has an I²C bus format and a serial format.

The I²C bus formats are addressing formats with an acknowledge bit. These are shown in figures 15.3 (a) and (b). 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 15.4.

Figure 15.5 shows the I²C bus timing.

The symbols used in figures 15.3 to 15.5 are explained in table 15.8.

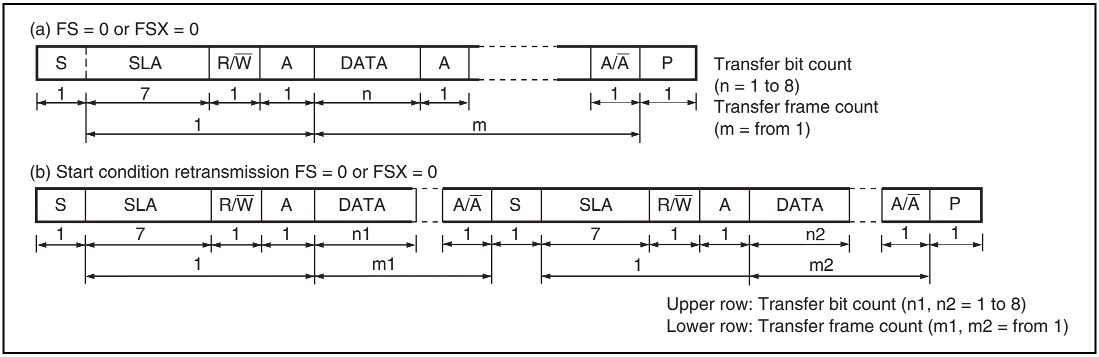


Figure 15.3 I²C Bus Data Formats (I²C Bus Formats)

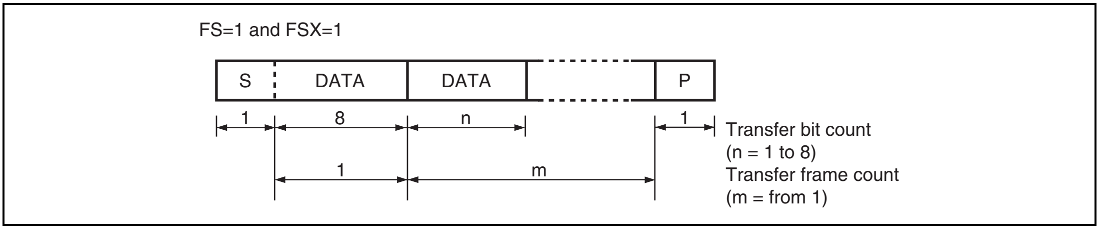


Figure 15.4 I²C Bus Data Formats (Serial Formats)

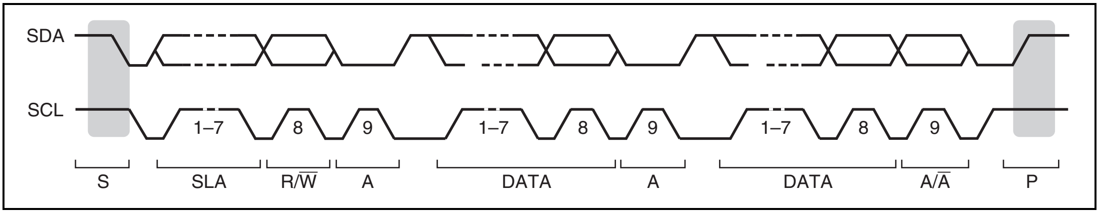


Figure 15.5 I^{2}C Bus Timing

Table 15.8 I^{2}C Bus Data Format Symbols

Symbol	Description
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/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

15.4.2 Initialization

Initialize the IIC by the procedure shown in figure 15.6 before starting transmission/reception of data.

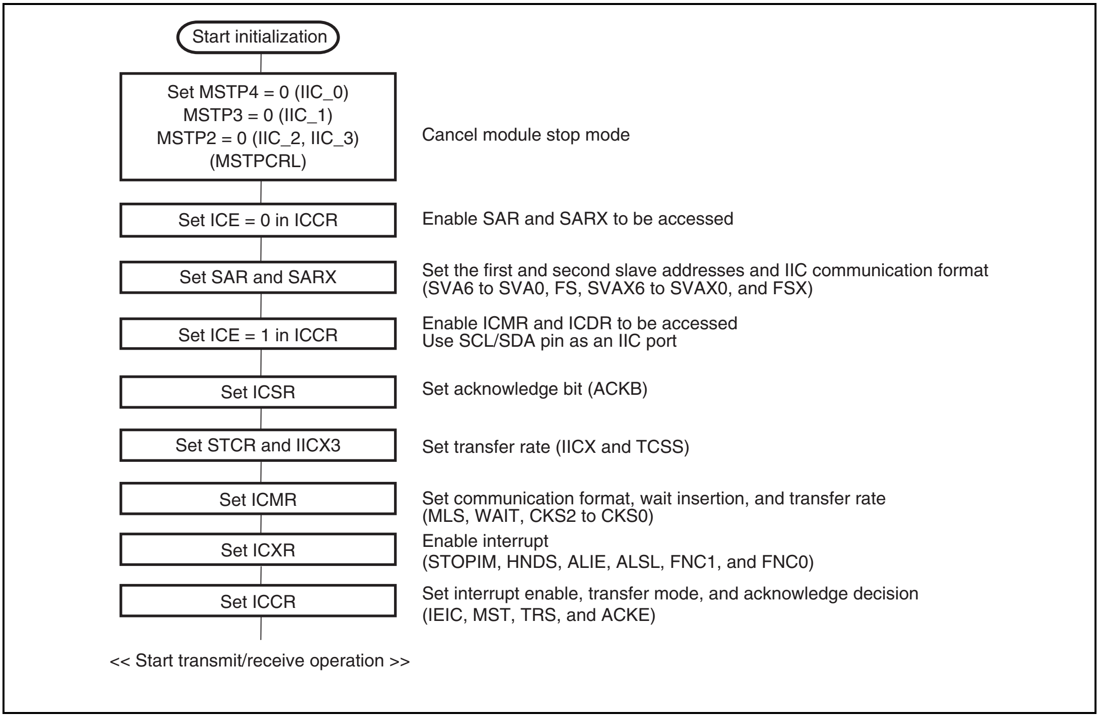

Start initialization
Set MSTP4 = 0 (IIC_0)
MSTP3 = 0 (IIC_1)
MSTP2 = 0 (IIC_2, IIC_3)
(MSTPCRL)
Cancel module stop mode
Set ICE = 0 in ICCR
Enable SAR and SARX to be accessed
Set SAR and SARX
Set the first and second slave addresses and IIC communication format
(SVA6 to SVA0, FS, SVAX6 to SVAX0, and 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 and IICX3
Set transfer rate (IICX and TCSS)
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 15.6 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.

15.4.3 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 15.7 shows the sample flowchart for the operations in master transmit mode.

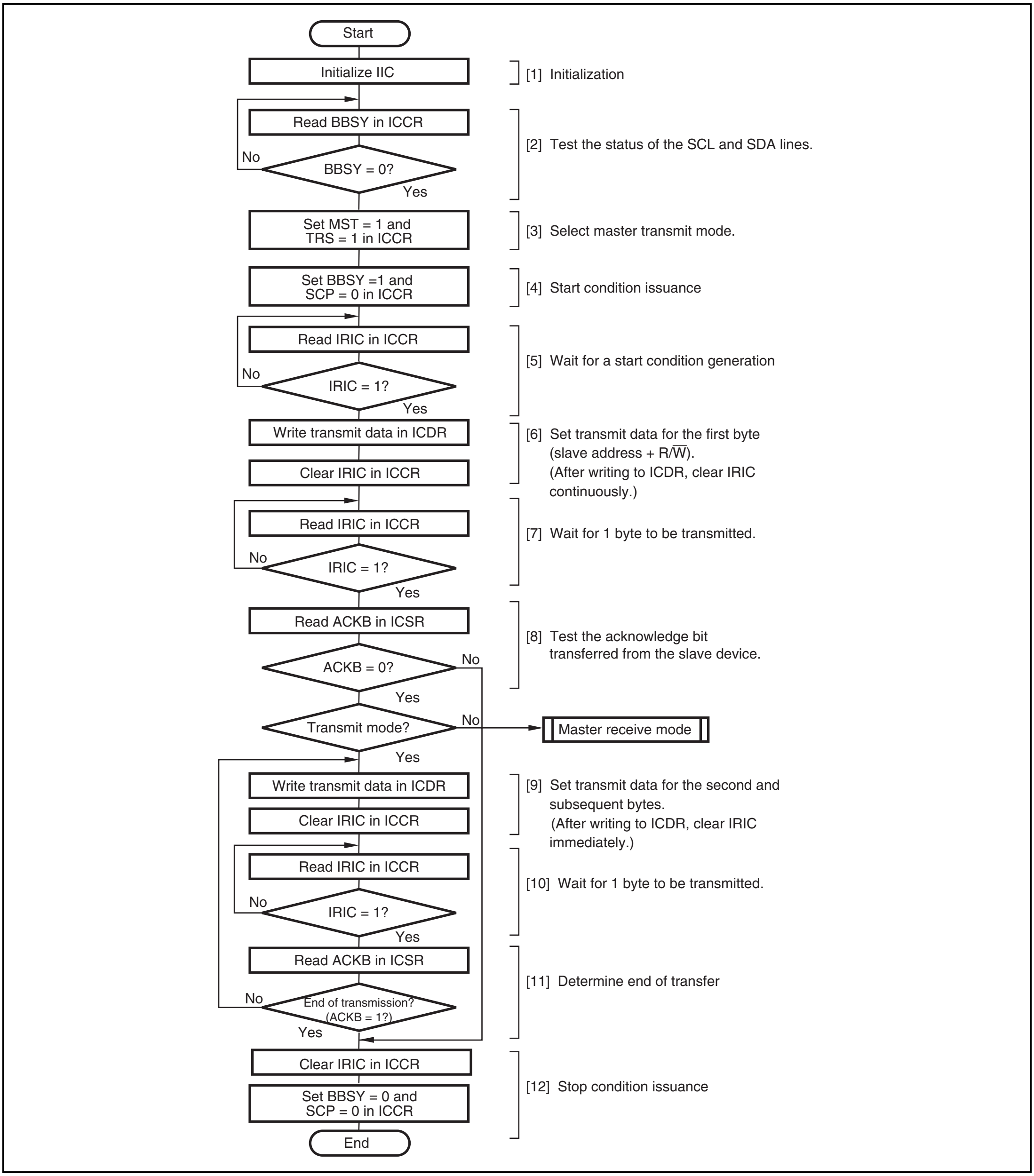

Start
Initialize IIC
[1] Initialization
Read BBSY in ICCR
BBSY = 0?
No
Yes
[2] Test the status of the SCL and SDA lines.
Set MST = 1 and TRS = 1 in ICCR
[3] Select master transmit mode.
Set BBSY =1 and SCP = 0 in ICCR
[4] Start condition issuance
Read IRIC in ICCR
IRIC = 1?
No
Yes
[5] Wait for a start condition generation
Write transmit data in ICDR
Clear IRIC in ICCR
[6] Set transmit data for the first byte (slave address + R/$\overline{W}$). (After writing to ICDR, clear IRIC continuously.)
Read IRIC in ICCR
IRIC = 1?
No
Yes
[7] Wait for 1 byte to be transmitted.
Read ACKB 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 in ICCR
[9] Set transmit data for the second and subsequent bytes. (After writing to ICDR, clear IRIC immediately.)
Read IRIC in ICCR
IRIC = 1?
No
Yes
[10] Wait for 1 byte to be transmitted.
Read ACKB in ICSR
End of transmission? (ACKB = 1?)
No
Yes
[11] Determine end of transfer
Clear IRIC in ICCR
Set BBSY = 0 and SCP = 0 in ICCR
[12] Stop condition issuance
End

Figure 15.7 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 15.4.2, 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/$\overline{\text{W}}$) 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/$\overline{\text{W}}$).

   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.

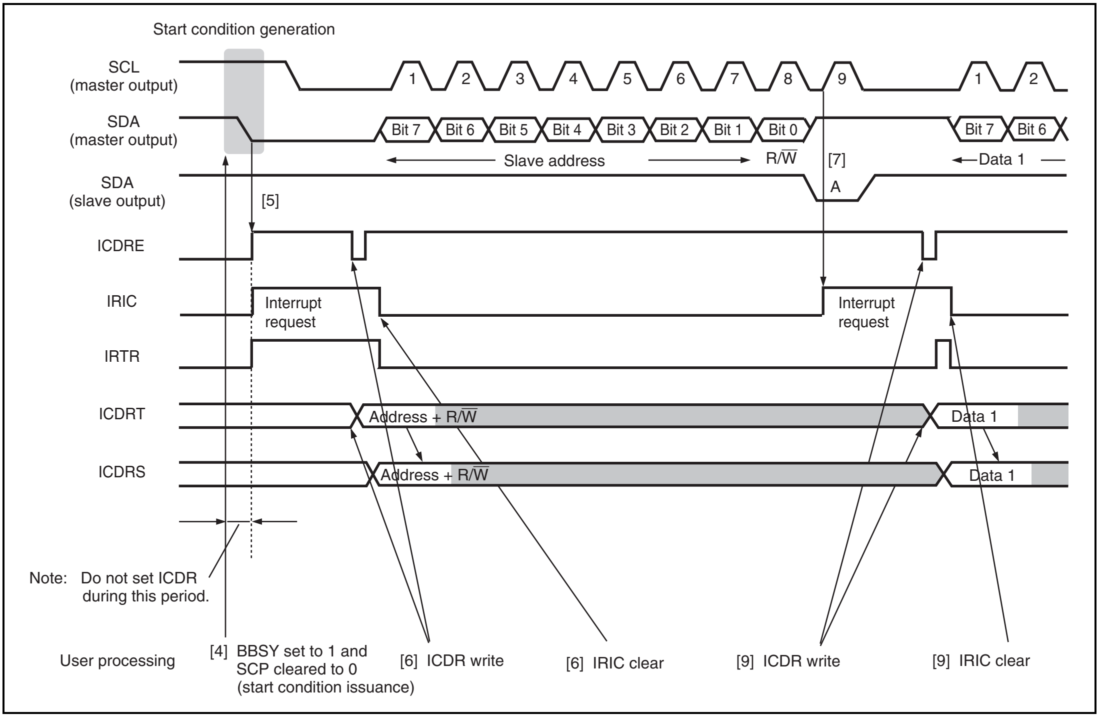


Figure 15.8 Operation Timing Example in Master Transmit Mode (MLS = WAIT = 0)

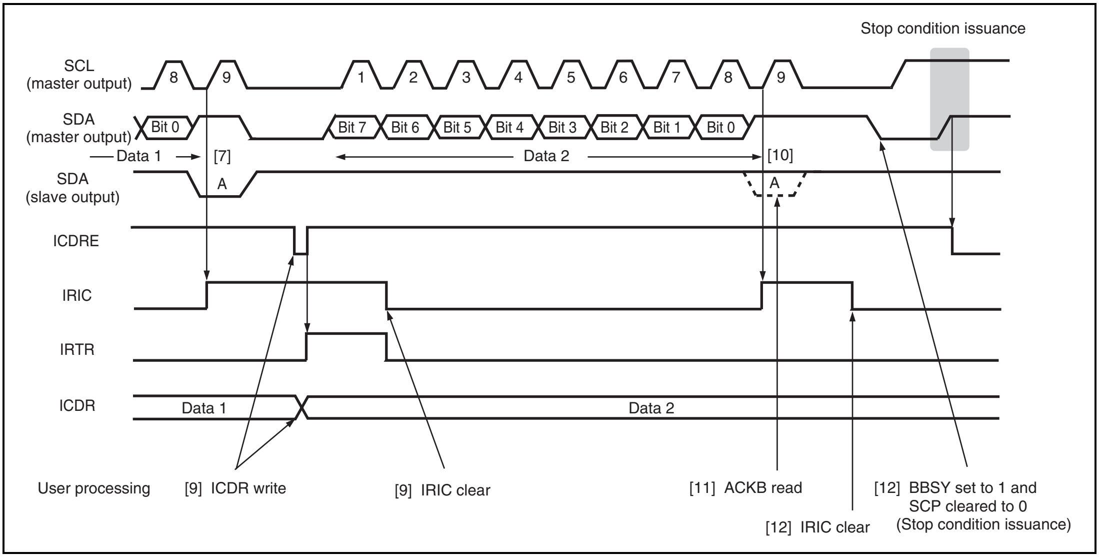


Figure 15.9 Stop Condition Issuance Operation Timing Example in Master Transmit Mode (MLS = WAIT = 0)

15.4.4 Master Receive Operation

In I^{2}C bus format master receive mode, the master device outputs the receive clock, receives data, and returns an acknowledge signal. The slave device transmits data.

The master device transmits data containing the slave address and R/$\overline{\text{W}}$ (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 15.10 shows the sample flowchart for the operations in master receive mode (HNDS = 1).

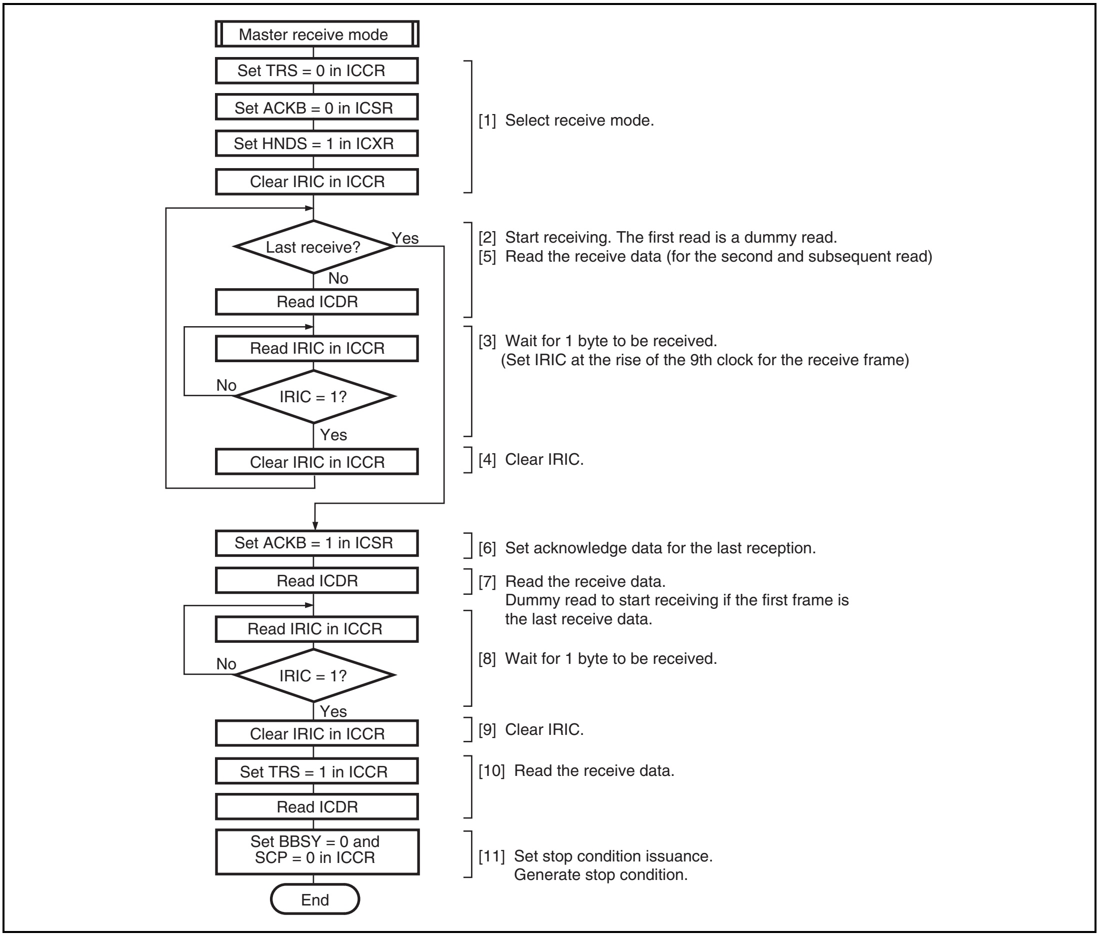

Master receive mode
Set TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 1 in ICXR
Clear IRIC in ICCR
Last receive?
Yes
No
Read ICDR
Read IRIC in ICCR
No
IRIC = 1?
Yes
Clear IRIC in ICCR
Set ACKB = 1 in ICSR
Read ICDR
Read IRIC in ICCR
No
IRIC = 1?
Yes
Clear IRIC in ICCR
Set TRS = 1 in ICCR
Read ICDR
Set BBSY = 0 and SCP = 0 in ICCR
End
[1] Select receive mode.
[2] Start receiving. The first read is a dummy read.
[5] Read the receive data (for the second and subsequent read)
[3] Wait for 1 byte to be received.
(Set IRIC at the rise of the 9th clock for the receive frame)
[4] Clear IRIC.
[6] Set acknowledge data for the last reception.
[7] Read the receive data.
Dummy read to start receiving if the first frame is the last receive data.
[8] Wait for 1 byte to be received.
[9] Clear IRIC.
[10] Read the receive data.
[11] Set stop condition issuance.
Generate stop condition.

Figure 15.10 Sample Flowchart for Operations in Master Receive Mode (HNDS = 1)

The reception procedure and operations 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, and the receive clock is output, and data received, in synchronization with the internal clock. (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 to ICDRR from ICDRS 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 determine the next interrupt.

   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.

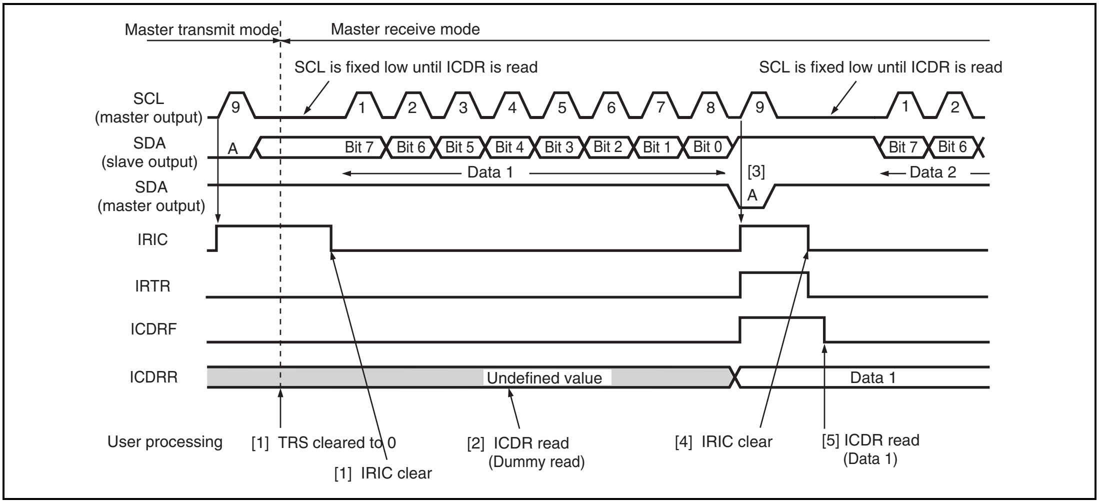


Figure 15.11 Master Receive Mode Operation Timing Example (MLS = WAIT = 0, HNDS = 1)

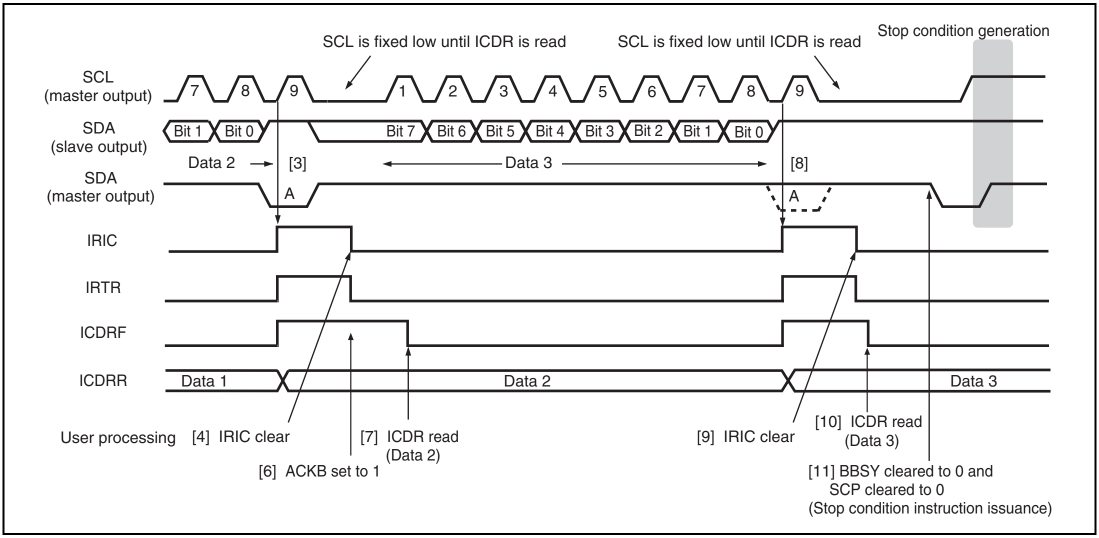


Figure 15.12 Stop Condition Issuance Timing Example in Master Receive Mode (MLS = WAIT = 0, HNDS = 1)

Receive Operation Using the Wait Function:

Figures 15.13 and 15.14 show the sample flowcharts for the operations in master receive mode (WAIT = 1).

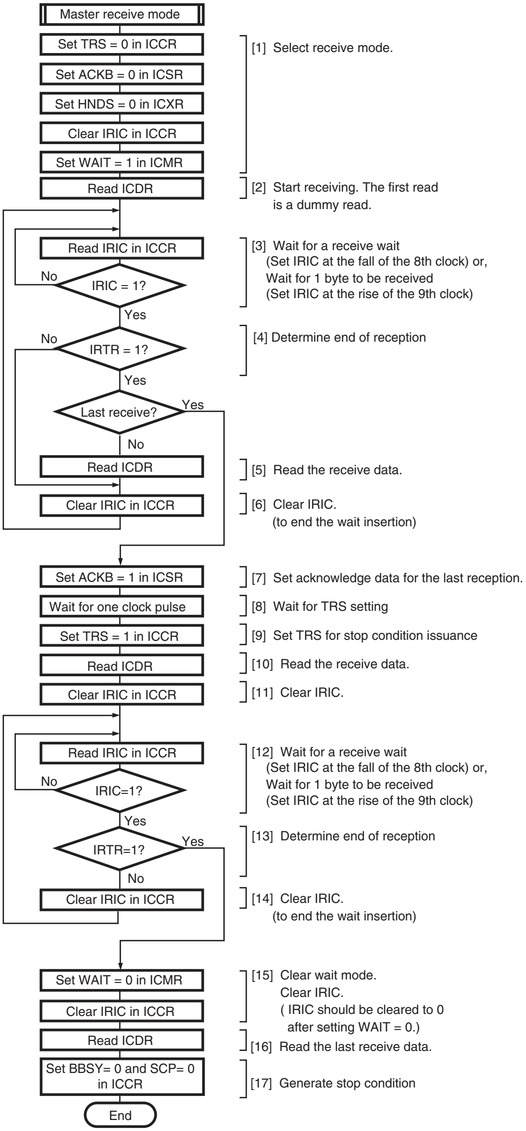

Master receive mode
Set TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 0 in ICXR
Clear IRIC in ICCR
Set WAIT = 1 in ICMR
[1] Select receive mode.
Read ICDR
[2] Start receiving. The first read is a dummy read.
Read IRIC in ICCR
IRIC = 1?
No
Yes
[3] Wait for a receive wait (Set IRIC at the fall of the 8th clock) or, Wait for 1 byte to be received (Set IRIC at the rise of the 9th clock)
IRTR = 1?
No
Yes
[4] Determine end of reception
Last receive?
Yes
No
Read ICDR
[5] Read the receive data.
Clear IRIC in ICCR
[6] Clear IRIC. (to end the wait insertion)
Set ACKB = 1 in ICSR
[7] Set acknowledge data for the last reception.
Wait for one clock pulse
[8] Wait for TRS setting
Set TRS = 1 in ICCR
[9] Set TRS for stop condition issuance
Read ICDR
[10] Read the receive data.
Clear IRIC in ICCR
[11] Clear IRIC.
Read IRIC in ICCR
IRIC=1?
No
Yes
[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)
IRTR=1?
Yes
No
[13] Determine end of reception
Clear IRIC in ICCR
[14] Clear IRIC. (to end the wait insertion)
Set WAIT = 0 in ICMR
Clear IRIC in ICCR
[15] Clear wait mode. Clear IRIC. (IRIC 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 15.13 Sample Flowchart for Operations in Master Receive Mode (receiving multiple bytes) (WAIT = 1)

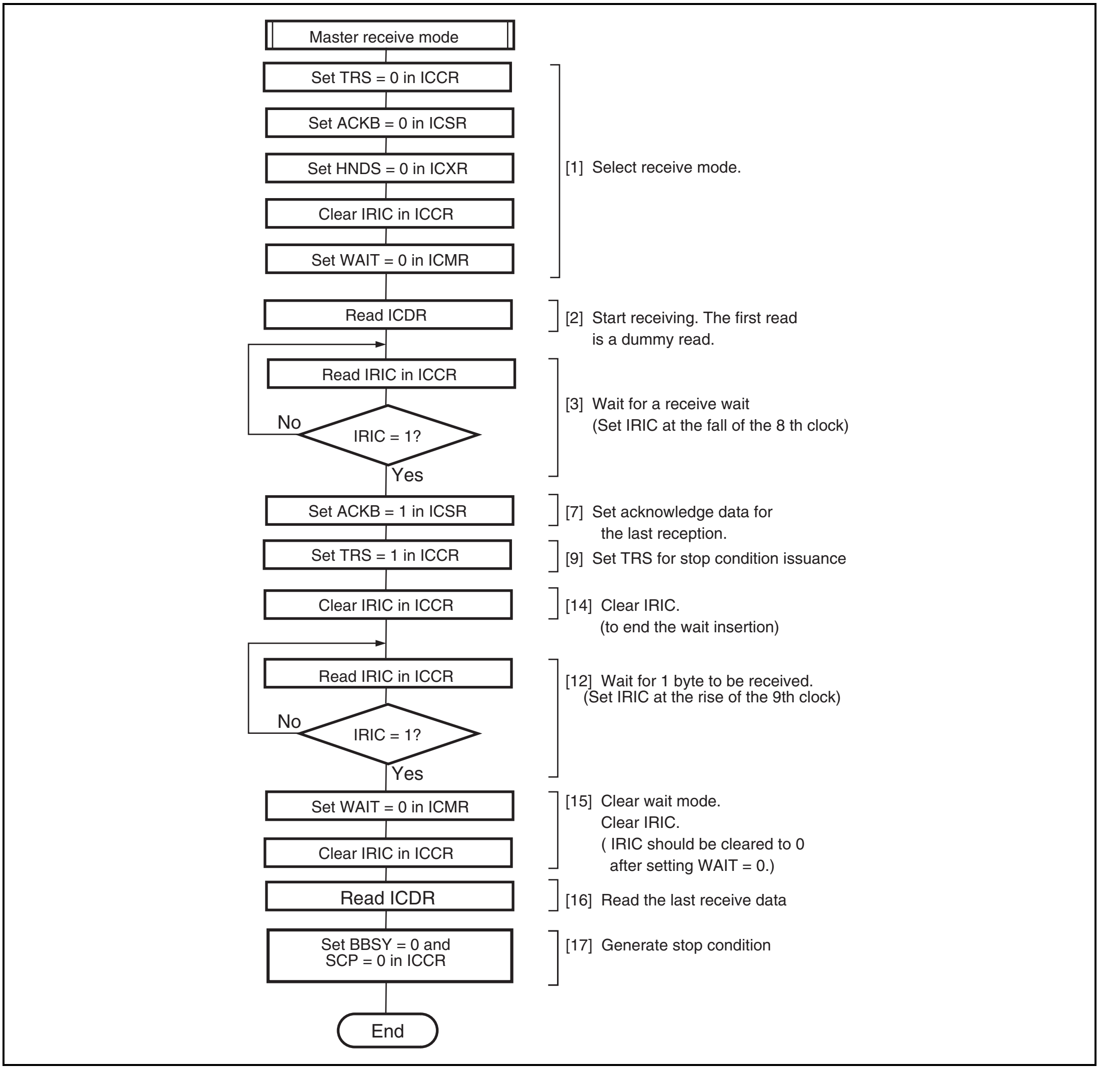

Master receive mode
Set TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 0 in ICXR
Clear IRIC in ICCR
Set WAIT = 0 in ICMR
[1] Select receive mode.
Read ICDR
[2] Start receiving. The first read is a dummy read.
Read IRIC in ICCR
No
IRIC = 1?
Yes
[3] Wait for a receive wait (Set IRIC at the fall of the 8 th clock)
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 in ICCR
[14] Clear IRIC. (to end the wait insertion)
Read IRIC in ICCR
No
IRIC = 1?
Yes
[12] Wait for 1 byte to be received. (Set IRIC at the rise of the 9th clock)
Set WAIT = 0 in ICMR
Clear IRIC in ICCR
[15] Clear wait mode. Clear IRIC. (IRIC 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 15.14 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 15.14.

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, and the receive clock is output, and data received, in synchronization with the internal clock.
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.

   (1) 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.

   (2) 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 (1) 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.

    (1) 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.

(2) 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.

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 ICMR to cancel the wait mode.

    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.

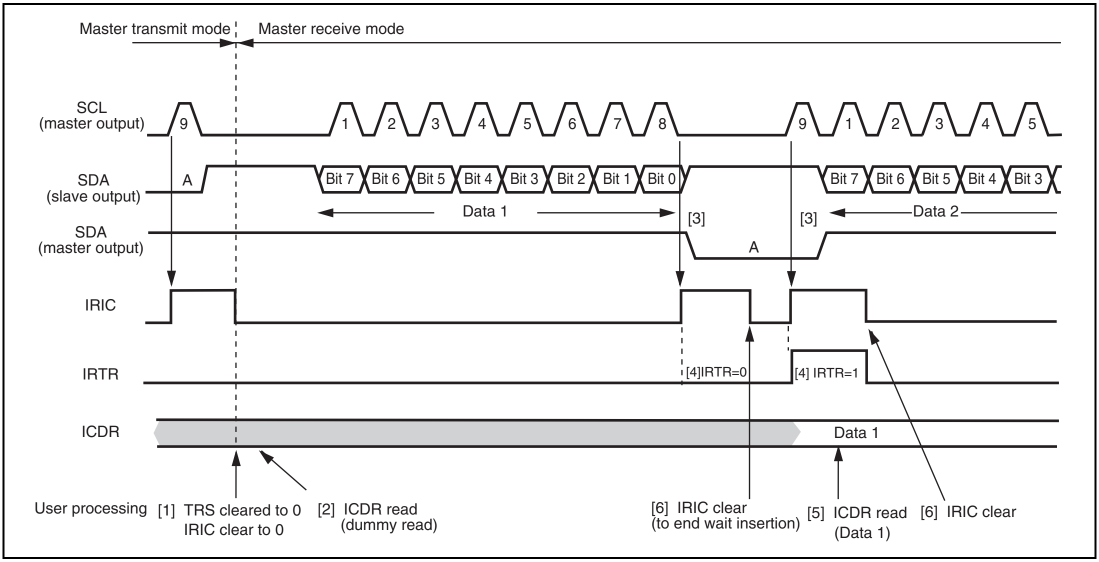


Figure 15.15 Master Receive Mode Operation Timing Example (MLS = ACKB = 0, WAIT = 1)

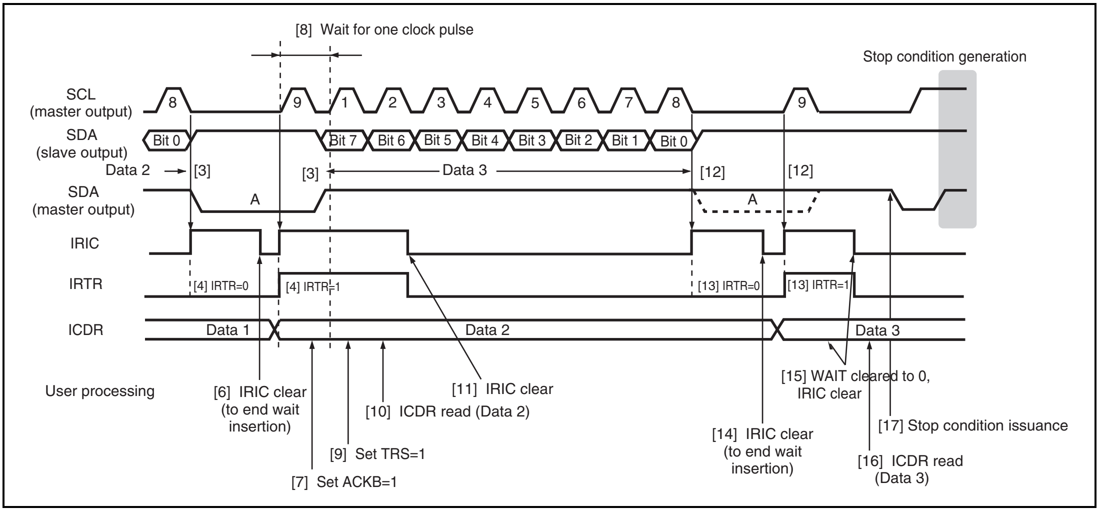


Figure 15.16 Stop Condition Issuance Timing Example in Master Receive Mode (MLS = ACKB = 0, WAIT = 1)

15.4.5 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 15.17 shows the sample flowchart for the operations in slave receive mode (HNDS = 1).

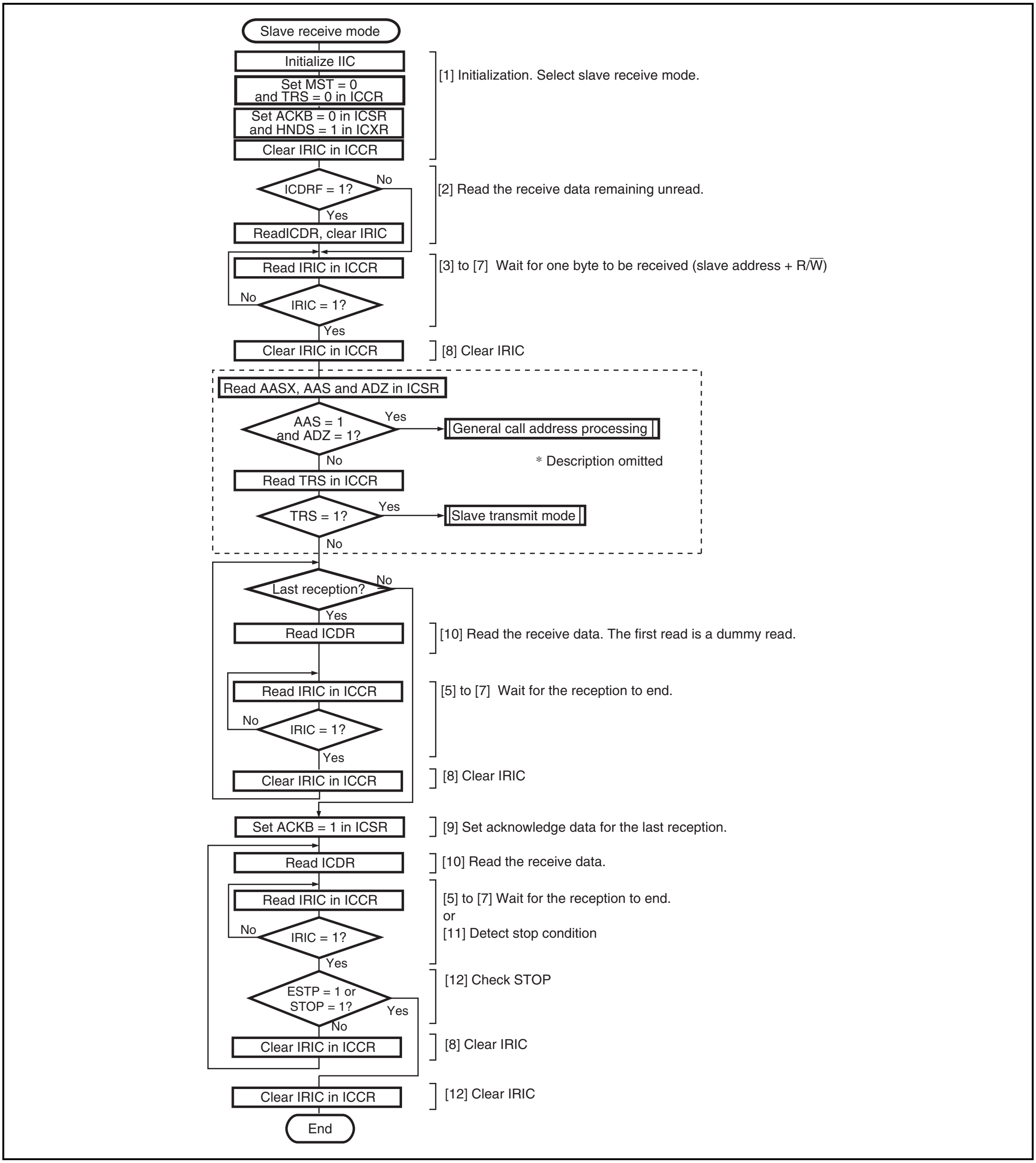

Slave receive mode
Initialize IIC
Set MST = 0 and TRS = 0 in ICCR
Set ACKB = 0 in ICSR and HNDS = 1 in ICXR
Clear IRIC in ICCR
[1] Initialization. Select slave receive mode.
ICDRF = 1?
No
Yes
ReadICDR, clear IRIC
[2] Read the receive data remaining unread.
Read IRIC in ICCR
No
IRIC = 1?
Yes
[3] to [7] Wait for one byte to be received (slave address + R/W)
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
Last reception?
No
Yes
Read ICDR
[10] Read the receive data. The first read is a dummy read.
Read IRIC in ICCR
No
IRIC = 1?
Yes
[5] to [7] Wait for the reception to end.
Clear IRIC in ICCR
[8] Clear IRIC
Set ACKB = 1 in ICSR
[9] Set acknowledge data for the last reception.
Read ICDR
[10] Read the receive data.
Read IRIC in ICCR
No
IRIC = 1?
Yes
[5] to [7] Wait for the reception to end.
or
[11] Detect stop condition
ESTP = 1 or STOP = 1?
Yes
No
[12] Check STOP
Clear IRIC in ICCR
[8] Clear IRIC
Clear IRIC in ICCR
[12] Clear IRIC
End

Figure 15.17 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 15.4.2, 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 ($R/\overline{W}$), 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 receive operation is performed. If the 8th data bit ($R/\overline{W}$) 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 the acknowledge data.
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.

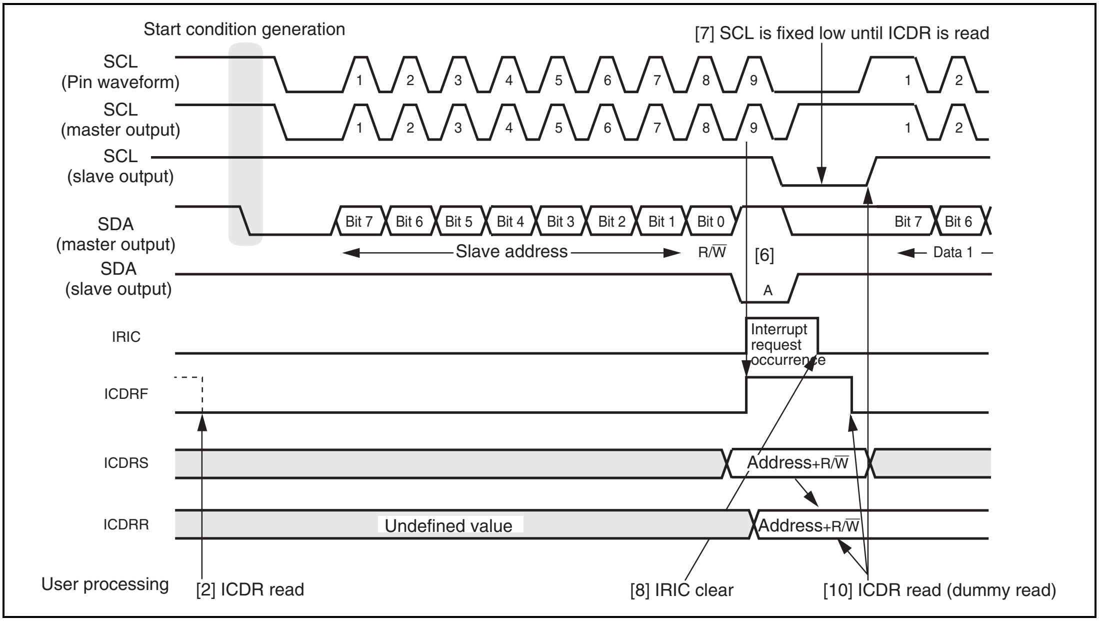


Figure 15.18 Slave Receive Mode Operation Timing Example (1) (MLS = 0, HNDS= 1)

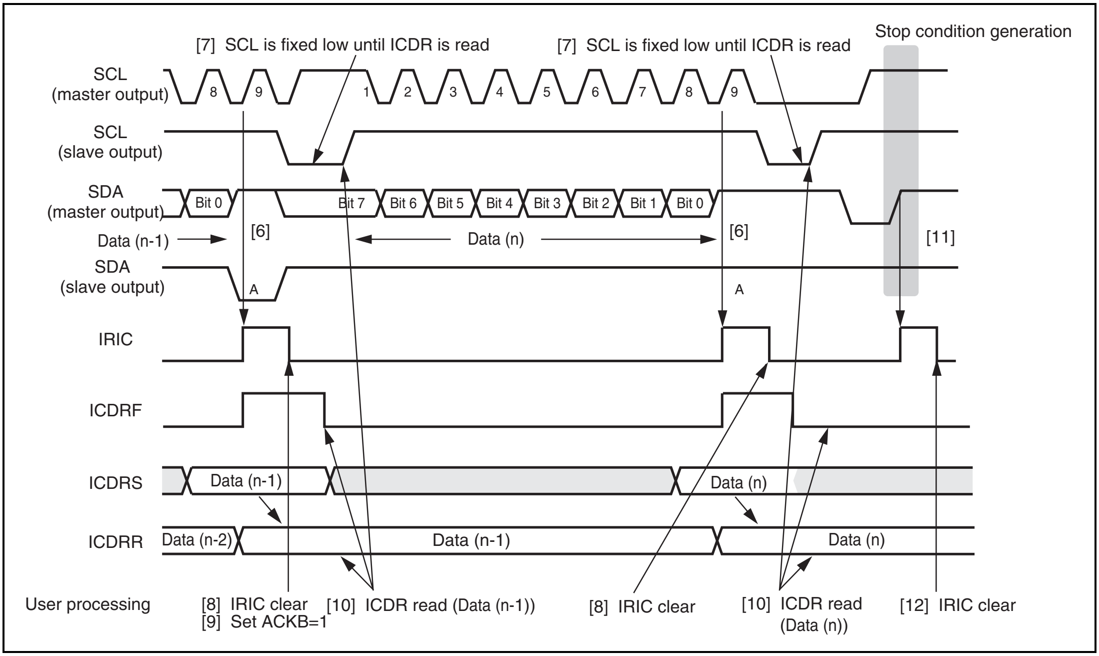


Figure 15.19 Slave Receive Mode Operation Timing Example (2) (MLS = 0, HNDS= 1)

Continuous Receive Operation:

Figure 15.20 shows the sample flowchart for the operations in slave receive mode (HNDS = 0).

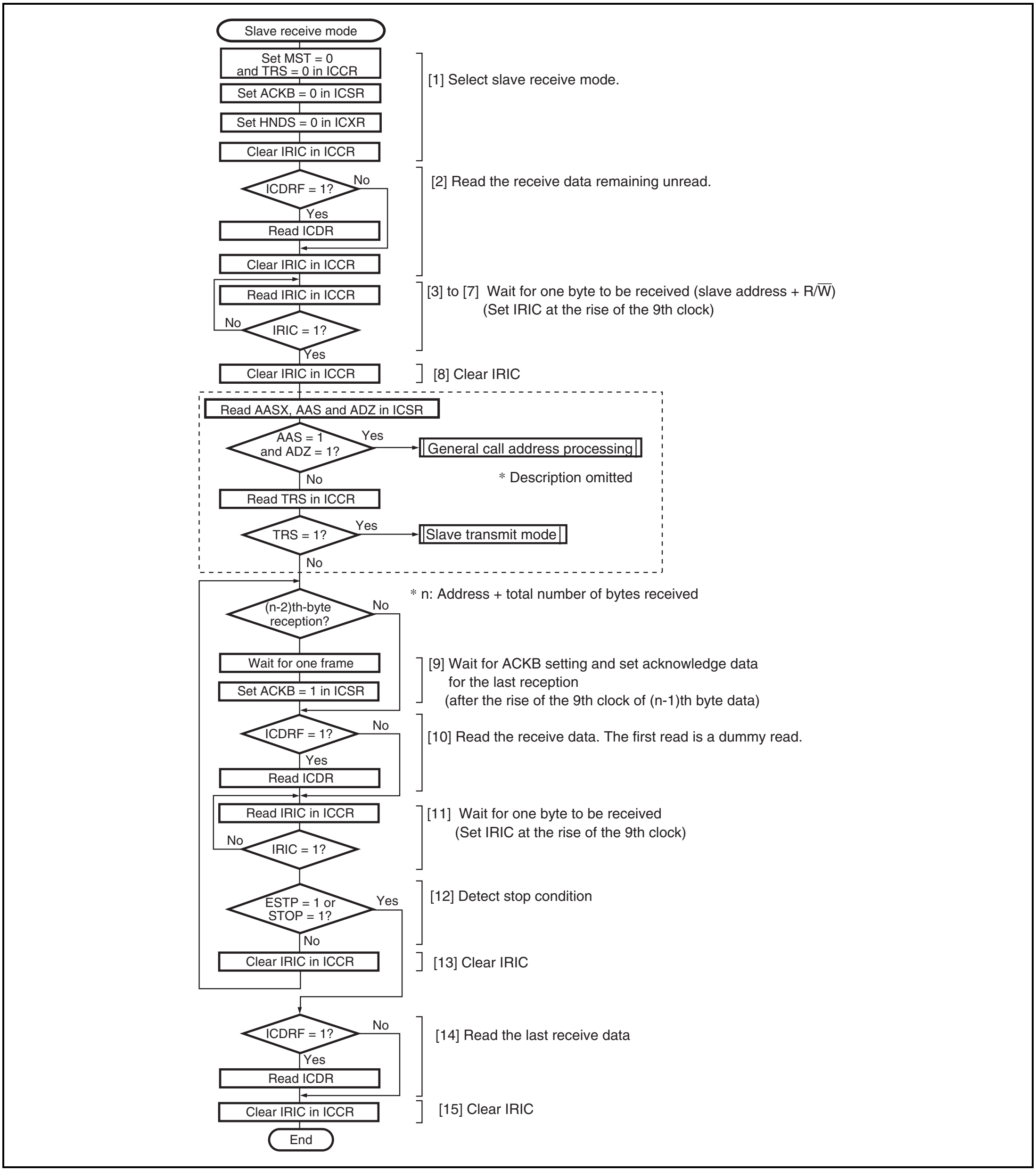

Slave receive mode
Set MST = 0 and TRS = 0 in ICCR
Set ACKB = 0 in ICSR
Set HNDS = 0 in ICXR
Clear IRIC in ICCR
[1] Select slave receive mode.
ICDRF = 1?
No
Yes
Read ICDR
Clear IRIC in ICCR
[2] Read the receive data remaining unread.
Read IRIC in ICCR
IRIC = 1?
No
Yes
[3] to [7] Wait for one byte to be received (slave address + R/W̄)
(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 15.20 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 15.4.2, 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/$\overline{\text{W}}$) 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 the acknowledge data.
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 IRIC 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.

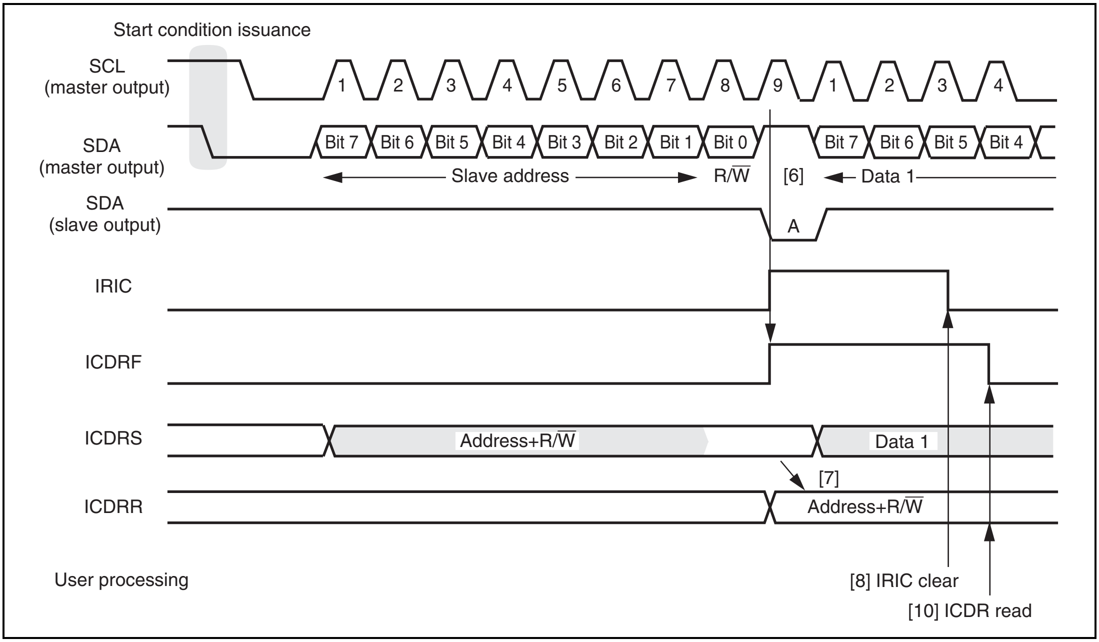


**Figure 15.21 Slave Receive Mode Operation Timing Example (1)
(MLS = ACKB = 0, HNDS = 0)**

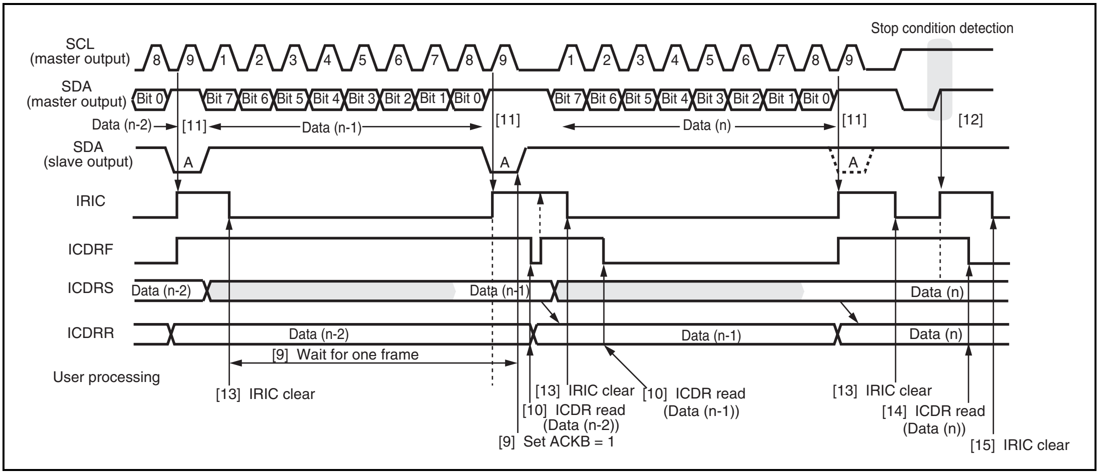


**Figure 15.22 Slave Receive Mode Operation Timing Example (2)
(MLS = ACKB = 0, HNDS = 0)**

15.4.6 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 15.23 shows the sample flowchart for the operations in slave transmit mode.

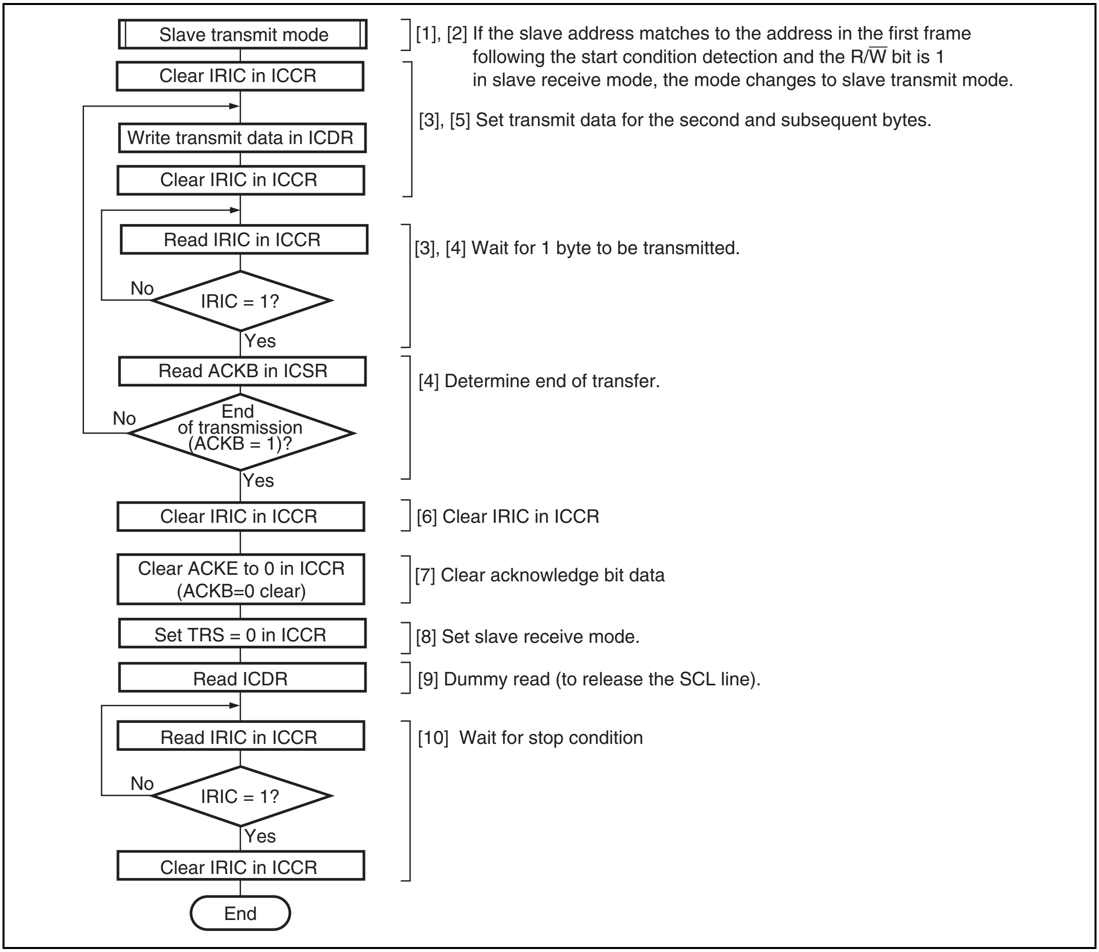

Slave transmit mode
Clear IRIC in ICCR
Write transmit data in ICDR
Clear IRIC in ICCR
Read IRIC in ICCR
No
IRIC = 1?
Yes
Read ACKB in ICSR
No
End of transmission (ACKB = 1)?
Yes
Clear IRIC in ICCR
Clear ACKE to 0 in ICCR (ACKB=0 clear)
Set TRS = 0 in ICCR
Read ICDR
Read IRIC in ICCR
No
IRIC = 1?
Yes
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 receive 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] Clear 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 15.23 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 ($R/\overline{W}$) 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 9th 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 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 9th 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 the ICCR register 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 SCL 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.

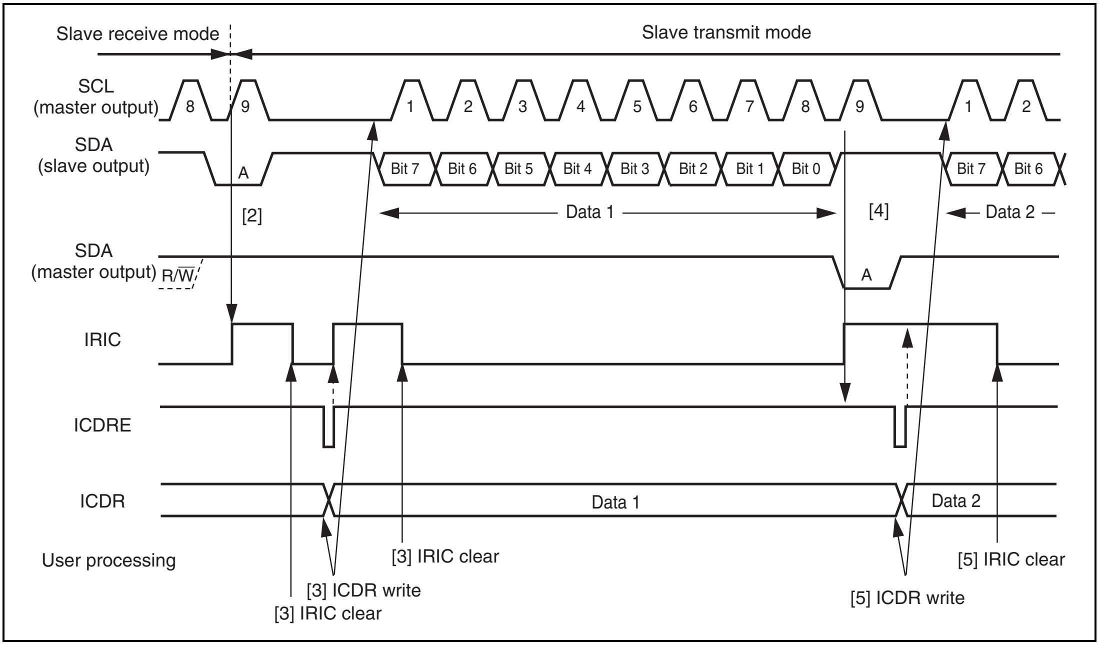


Figure 15.24 Slave Transmit Mode Operation Timing Example (MLS = 0)

15.4.7 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; this timing is synchronized with the internal clock. Figures 15.25 to 15.27 show the IRIC set timing and SCL control.

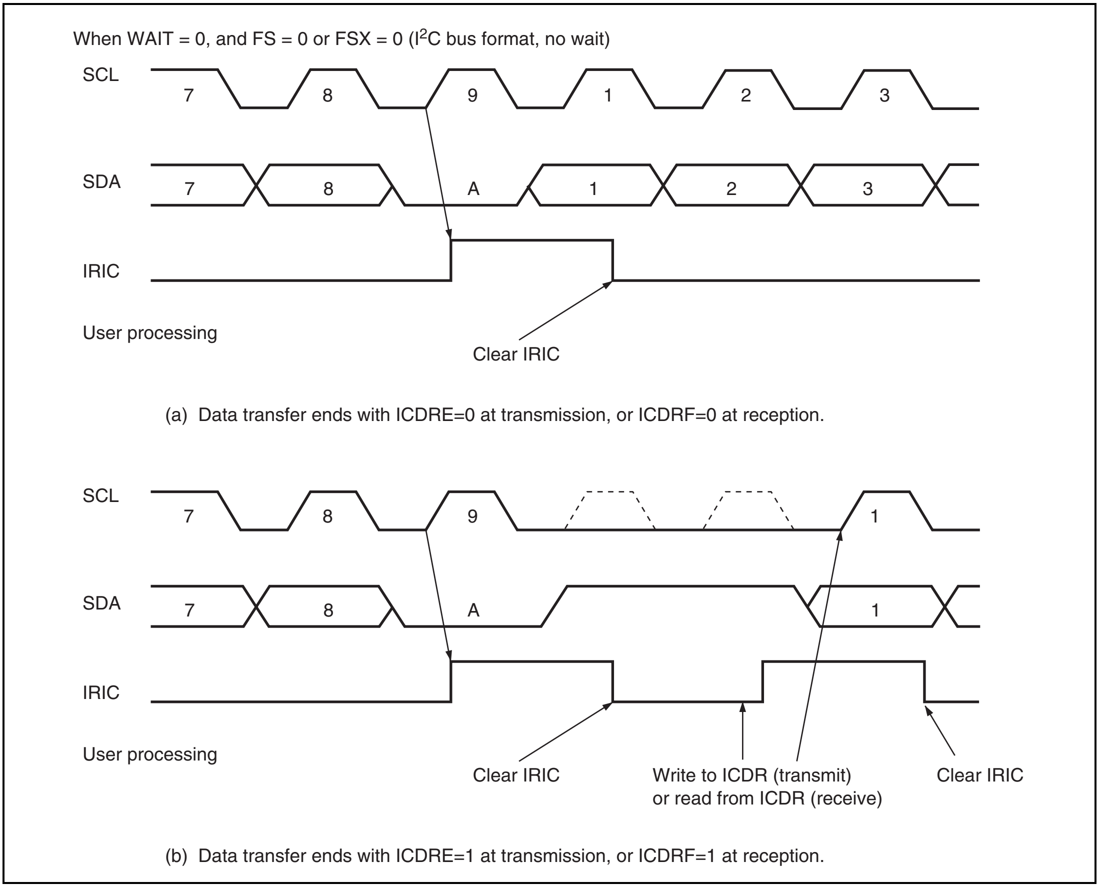
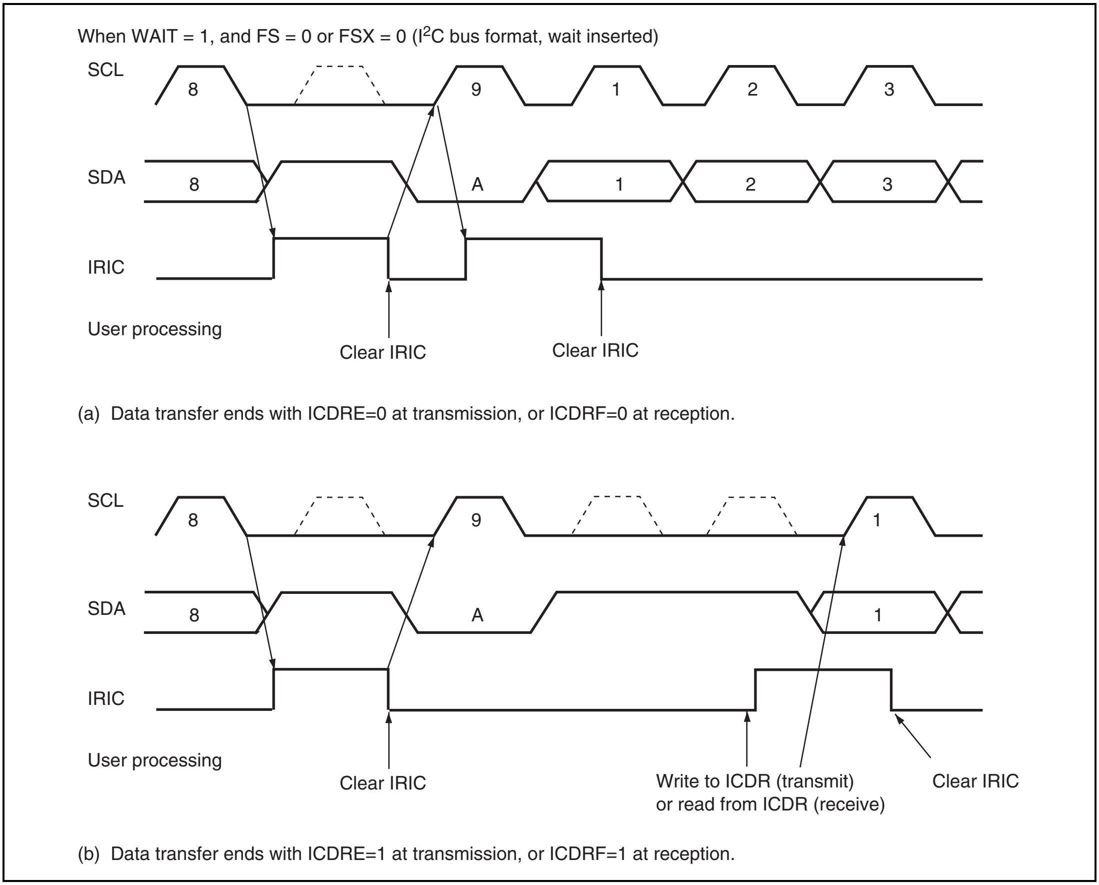

When WAIT = 0, and FS = 0 or FSX = 0 (I^{2}C bus format, no wait)
SCL
7
8
9
1
2
3
SDA
7
8
A
1
2
3
IRIC
User processing
Clear IRIC
(a) Data transfer ends with ICDRE=0 at transmission, or ICDRF=0 at reception.
SCL
7
8
9
1
SDA
7
8
A
1
IRIC
User processing
Clear IRIC
Write to ICDR (transmit)
or read from ICDR (receive)
Clear IRIC
(b) Data transfer ends with ICDRE=1 at transmission, or ICDRF=1 at reception.

Figure 15.25 IRIC Setting Timing and SCL Control (1)



When WAIT = 1, and FS = 0 or FSX = 0 (I^2C bus format, wait inserted)

(a) Data transfer ends with ICDRE=0 at transmission, or ICDRF=0 at reception.

(b) Data transfer ends with ICDRE=1 at transmission, or ICDRF=1 at reception.

Figure 15.26 IRIC Setting Timing and SCL Control (2)

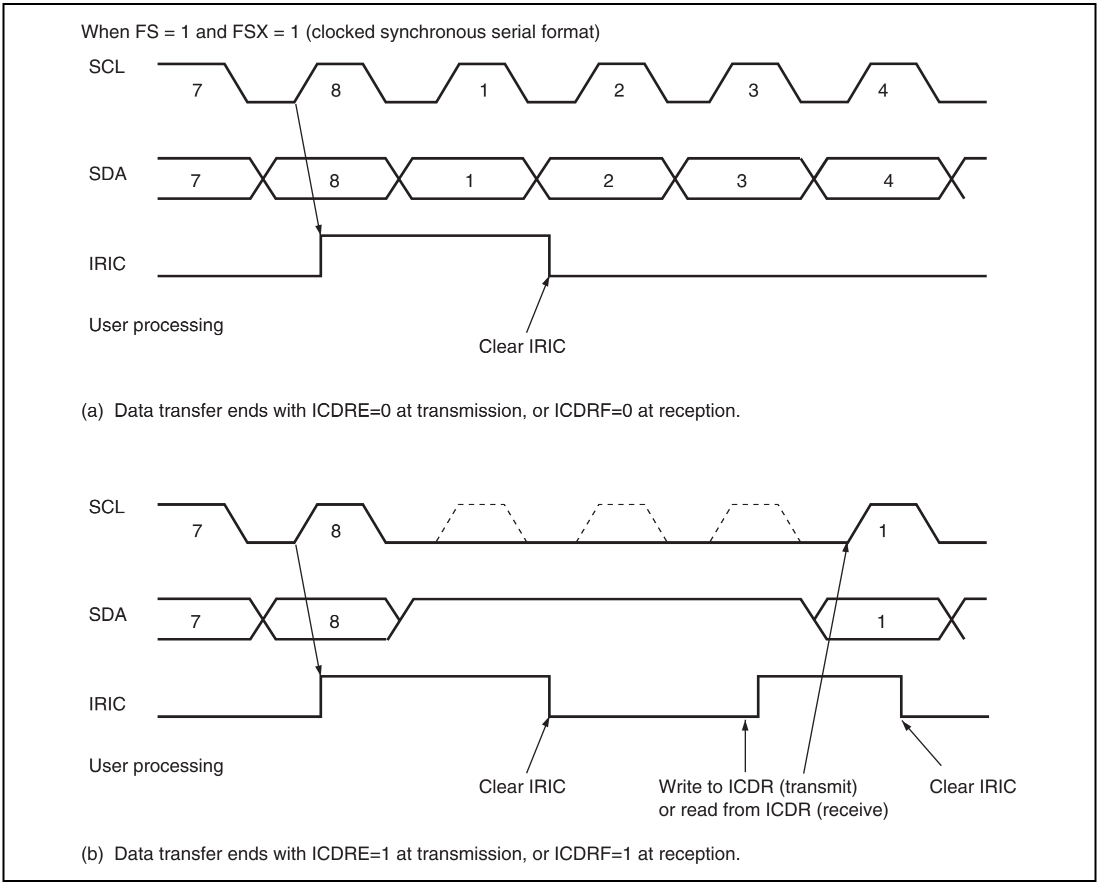

When FS = 1 and FSX = 1 (clocked synchronous serial format)
SCL
7
8
1
2
3
4
SDA
7
8
1
2
3
4
IRIC
User processing
Clear IRIC
(a) Data transfer ends with ICDRE=0 at transmission, or ICDRF=0 at reception.
SCL
7
8
1
SDA
7
8
1
IRIC
User processing
Clear IRIC
Write to ICDR (transmit)
or read from ICDR (receive)
Clear IRIC
(b) Data transfer ends with ICDRE=1 at transmission, or ICDRF=1 at reception.

Figure 15.27 IRIC Setting Timing and SCL Control (3)

15.4.8 Operation Using the DTC

This LSI provides the DTC to allow continuous data transfer. The DTC is initiated when the IRTR flag is set to 1, which is one of the two interrupt flags (IRTR and IRIC). When the ACKE bit is 0, the ICDRE, IRIC, and IRTR flags are set at the end of data transmission regardless of the acknowledge bit value. When the ACKE bit is 1, the ICDRE, IRIC, and IRTR flags are set if data transmission is completed with the acknowledge bit value of 0, and when the ACKE bit is 1, only the IRIC flag is set if data transmission is completed with the acknowledge bit value of 1.

When initiated, DTC transfers specified number of bytes, clears the ICDRE, IRIC, and IRTR flags to 0. Therefore, no interrupt is generated during continuous data transfer; however, if data transmission is completed with the acknowledge bit value of 1 when the ACKE bit is 1, DTC is not initiated, thus allowing an interrupt to be generated if enabled.

The acknowledge bit may indicate specific events such as completion of receive data processing for some receiving devices, and for other receiving devices, the acknowledge bit may be held to 1, indicating no specific events.

The I^{2}C bus format provides for selection of the slave device and transfer direction by means of the slave address and the R/$\overline{\text{W}}$ bit, confirmation of reception with the acknowledge bit, indication of the last frame, and so on. Therefore, continuous data transfer using the DTC must be carried out in conjunction with CPU processing by means of interrupts.

Table 15.9 shows some examples of processing using the DTC. These examples assume that the number of transfer data bytes is known in slave mode.

Table 15.9 Examples of Operation Using the DTC

Item	Master Transmit Mode	Master Receive Mode	Slave Transmit Mode	Slave Receive Mode
Slave address + $R/\overline{W}$ bit transmission/ reception	Transmission by DTC (ICDR write)	Transmission by CPU (ICDR write)	Reception by CPU (ICDR read)	Reception by CPU (ICDR read)
Dummy data read	—	Processing by CPU (ICDR read)	—	—
Actual data transmission/ reception	Transmission by DTC (ICDR write)	Reception by DTC (ICDR read)	Transmission by DTC (ICDR write)	Reception by DTC (ICDR read)
Dummy data (H'FF) write	—	—	Processing by DTC (ICDR write)	—
Last frame processing	Not necessary	Reception by CPU (ICDR read)	Not necessary	Reception by CPU (ICDR read)
Transfer request processing after last frame processing	1st time: Clearing by CPU 2nd time: Stop condition issuance by CPU	Not necessary	Automatic clearing on detection of stop condition during transmission of dummy data (H'FF)	Not necessary
Setting of number of DTC transfer data frames	Transmission: Actual data count + 1 (+1 equivalent to slave address + $R/\overline{W}$ bits)	Reception: Actual data count	Transmission: Actual data count + 1 (+1 equivalent to dummy data (H'FF))	Reception: Actual data count

15.4.9 Noise Canceler

The logic levels at the SCL and SDA pins are routed through noise cancelers before being latched internally. Figure 15.28 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.

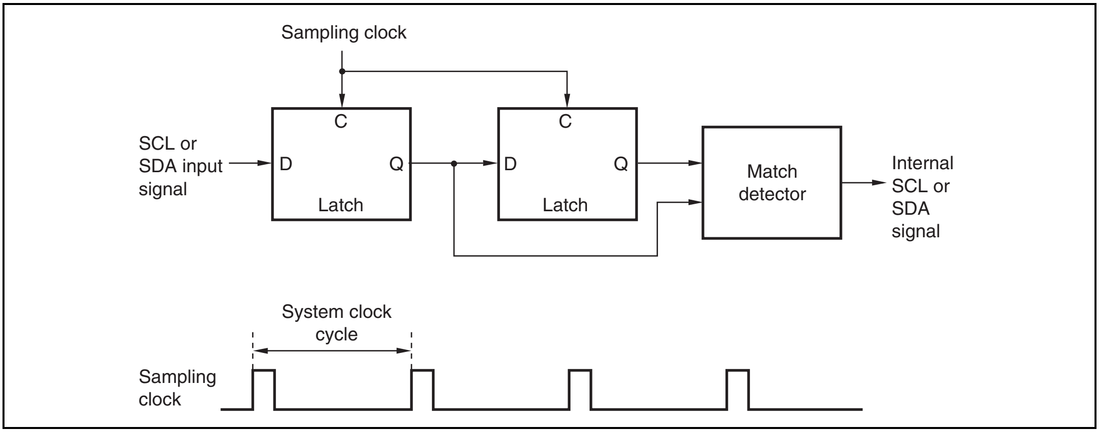


Figure 15.28 Block Diagram of Noise Canceler

15.4.10 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 clearing ICE bit.

Scope of Initialization: The initialization executed by this function covers the following items:

- ICDRE and ICDRF internal flags
- Transmit/receive sequencer and internal operating clock counter
- Internal latches for retaining the output state of the SCL and SDA pins (wait, clock, data output, etc.)

The following items are not initialized:

- Actual register values (ICDR, SAR, SARX, ICMR, ICCR, ICSR, ICXR(other than ICDRE and ICDRF))
- Internal latches used to retain register read information for setting/clearing flags in the ICMR, ICCR, and ICSR registers
- The value of the ICMR register bit counter (BC2 to BC0)
- Generated interrupt sources (interrupt sources transferred to the interrupt controller)

Notes on Initialization:

- Interrupt flags and interrupt sources are not cleared, and so flag clearing measures must be taken as necessary.
- Basically, other register flags are not cleared either, and so flag clearing measures must be taken as necessary.
- 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 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 ICE bit clearing.
4. Initialize (re-set) the IIC registers.

15.5 Interrupt Source

The IIC interrupt source is IICI. The IIC interrupt sources and their priority order are shown in table 15.10. Each interrupt source is enabled or disabled by the ICCR interrupt enable bit and transferred to the interrupt controller independently.

Table 15.10 IIC Interrupt Source

Channel	Bit Name	Enable Bit	Interrupt Source	Interrupt Flag	DTC Activation	Priority
2	IICI2	IEIC	I^2C bus interface interrupt request	IRIC	Possible	High
3	IICI3	IEIC	I^2C bus interface interrupt request	IRIC	Possible	
0	IICI0	IEIC	I^2C bus interface interrupt request	IRIC	Possible	
1	IICI1	IEIC	I^2C bus interface interrupt request	IRIC	Possible	Low

15.6 Usage Notes

1. In master mode, if an instruction to generate a start condition is immediately followed by an instruction to generate a stop condition, neither condition will be output correctly. To output consecutive start and stop conditions*, after issuing the instruction that generates the start condition, read the relevant DR registers of I^{2}C bus output pins, check that SCL and SDA are both low. If the ICE bit is set to 1, pin state can be monitored by reading DR register. Then issue the instruction that generates the stop condition. Note that SCL may not yet have gone low when BBSY is cleared to 0.

Note: * An illegal procedure in the I^{2}C bus specification.

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 15.11 shows the timing of SCL and SDA outputs in synchronization with the internal clock. Timings on the bus are determined by the rise and fall times of signals affected by the bus load capacitance, series resistance, and parallel resistance.

Table 15.11 I^{2}C Bus Timing (SCL and SDA Outputs)

Item	Symbol	Output Timing	Unit	Notes
SCL output cycle time	t_{SCLO}	$28t_{cyc}$ to $512t_{cyc}$	ns	See figure 24.18 (reference)
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)		$1t_{SCLLO} - (6t_{cyc}$ or $12t_{cyc}*)$		
Data output hold time	t_{SDAHO}	$3t_{cyc}$	ns	

Note: * $6t_{cyc}$ when IICXn is 0, $12t_{cyc}$ when IICXn is 1 (n = 0 to 3).

4. SCL and SDA input are sampled in synchronization with the internal clock. The AC timing therefore depends on the system clock cycle t_{cyc}, as shown in section 24, Electrical Characteristics. Note that the I^{2}C bus interface AC timing specification will not be met with a system clock frequency of less than 5 MHz.
5. The I^{2}C bus interface specification for the SCL rise time t_{sr} is 1000 ns or less (300 ns for high-speed mode). In master mode, the I^{2}C bus interface monitors the SCL line and synchronizes one bit at a time during communication. If t_{sr} (the time for SCL to go from low to V_{IH}) exceeds the time determined by the input clock of the I^{2}C bus interface, the high period of SCL is extended. The SCL rise time is determined by the pull-up resistance and load capacitance of the SCL line. To insure proper operation at the set transfer rate, adjust the pull-up resistance and load capacitance so that the SCL rise time does not exceed the values given in table 15.12.

Table 15.12 Permissible SCL Rise Time (t_{sr}) Values

				Time Indication [ns]		
TCSS	**IICXn**	**t_{cyc} Indication**		**I^{2}C Bus Specification (Max.)**	**ϕ = 20 MHz**	**ϕ = 25 MHz**
0	0	7.5 t_{cyc}	Standard mode	1000	375	300
			High-speed mode	300	300	300
	1	17.5 t_{cyc}	Standard mode	1000	875	700
1	0		High-speed mode	300	300	300
1	1	37.5 t_{cyc}	Standard mode	1000	1000	1000
			High-speed mode	300	300	300

[Legend] n = 0 to 3

6. The I^{2}C bus interface specifications for the SCL and SDA rise and fall times are under 1000 ns and 300 ns. The I^{2}C bus interface SCL and SDA output timing is prescribed by t_{cyc}, as shown in table 15.11. However, because of the rise and fall times, the I^{2}C bus interface specifications may not be satisfied at the maximum transfer rate. Table 15.13 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^{2}C 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^{2}C bus.

t_{SCLLO} in high-speed mode and t_{STASO} in standard mode fail to satisfy the I^{2}C 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^{2}C bus.

Table 15.13 I^2C 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^2C Bus Specification (Min.)	ϕ = 20 MHz	ϕ = 25 MHz
—	—	Standard mode	—	—	ϕ/200	ϕ/224
—	—	High-speed mode	—	—	ϕ/48	ϕ/56
t_{SCLHO}	0.5 t_{SCLO} ($-t_{Sr}$)	Standard mode	–1000	4000	4000	3480
		High-speed mode	300	600	900	820
t_{SCLLO}	0.5 t_{SCLO} ($-t_{Sf}$)	Standard mode	–250	4700	4750	4230
		High-speed mode	–250	1300	950*1	870*1
t_{BUFO}	0.5 t_{SCLO} –1 t_{cyc} ($-t_{Sr}$)	Standard mode	–1000	4700	3950*1	3440*1
		High-speed mode	–300	1300	850*1	780*1
t_{STAHO}	0.5 t_{SCLO} –1 t_{cyc} ($-t_{Sf}$)	Standard mode	–250	4000	4700	4190
		High-speed mode	–250	600	900	830
t_{STASO}	1 t_{SCLO} ($-t_{Sr}$)	Standard mode	–1000	4700	9000	7960
		High-speed mode	–300	600	2100	1940
t_{STOSO}	0.5 t_{SCLO} + 2 t_{cyc} ($-t_{Sr}$)	Standard mode	–1000	4000	4100	3560
		High-speed mode	300	600	1000	900
t_{SDASO} (master)	1 t_{SCLLO}*3 –3 t_{cyc} ($-t_{Sr}$)	Standard mode	–1000	250	3600	3100
		High-speed mode	–300	100	500	450
t_{SDASO} (slave)	1 t_{SCLL}*3 –12 t_{cyc}*2 ($-t_{Sr}$)	Standard mode	–1000	250	3100	3220
		High-speed mode	–300	100	400	520
t_{SDAHO}	3 t_{cyc}	Standard mode	0	0	150	120
		High-speed mode	0	0	150	120

Notes: 1. Does not meet the I^2C 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 bits TCSS, IICX3 to IICX0 and CKS2 to CKS0. Depending on the frequency it may not be possible to achieve the maximum transfer rate; therefore, whether or not the I^2C bus interface specifications are met must be determined in accordance with the actual setting conditions.

2. Value when the IICXn bit is set to 1. When the IICXn bit is cleared to 0, the value is (– 6t_{cyc}) (n = 0 to 3).

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 register 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, 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 the ICCR register is cleared to 0, the stop condition has been generated, and the bus has been released, then read the ICDR register 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 15.29 (after confirming that the BBSY bit has been cleared to 0 in the ICCR register).

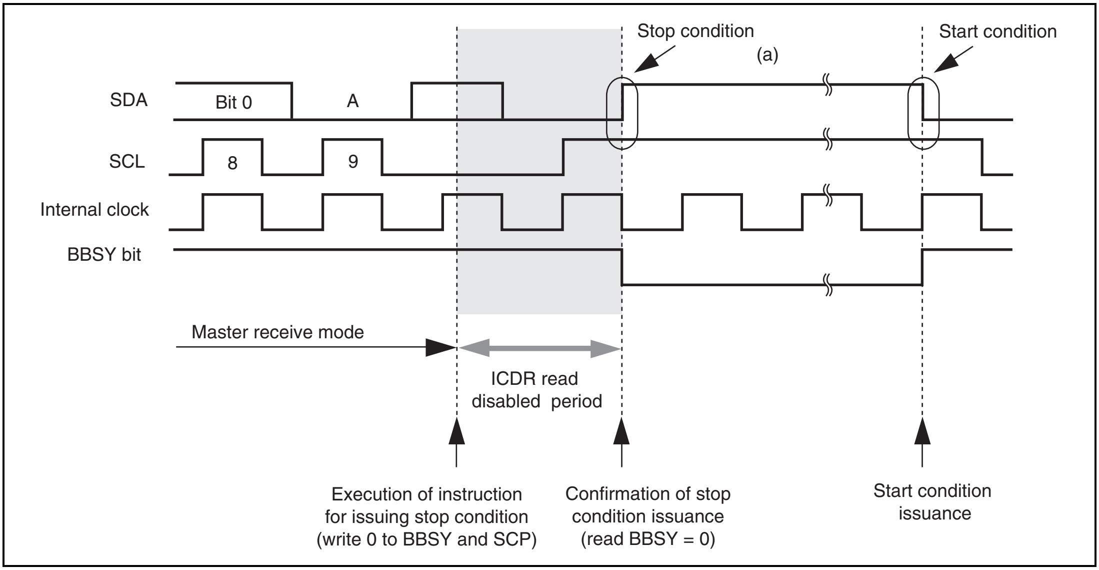

Stop condition
(a)
Start condition
SDA
Bit 0
A
SCL
8
9
Internal clock
BBSY bit
Master receive mode
ICDR read disabled period
Execution of instruction for issuing stop condition (write 0 to BBSY and SCP)
Confirmation of stop condition issuance (read BBSY = 0)
Start condition issuance

Figure 15.29 Notes on Reading Master Receive Data

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to B′11 in ICXR.

8. Notes on start condition issuance for retransmission

Figure 15.30 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.

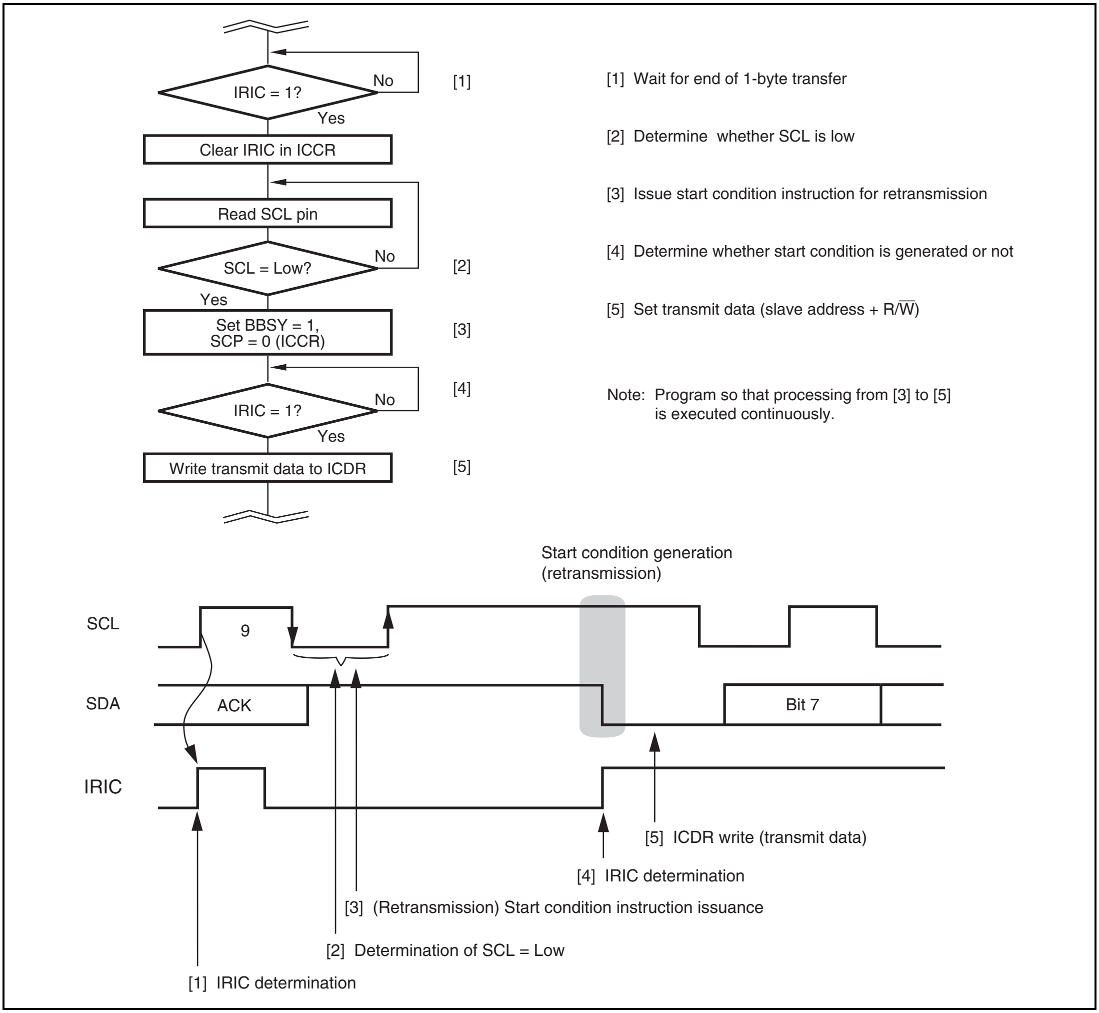

IRIC = 1?
No
Yes
[1]
Clear IRIC in ICCR
Read SCL pin
SCL = Low?
No
Yes
[2]
Set BBSY = 1,
SCP = 0 (ICCR)
[3]
IRIC = 1?
No
Yes
[4]
Write transmit data to ICDR
[5]
[1] Wait for end of 1-byte transfer
[2] Determine whether SCL is low
[3] Issue start condition instruction for retransmission
[4] Determine whether start condition is generated or not
[5] Set transmit data (slave address + R/W̄)
Note: Program so that processing from [3] to [5] is executed continuously.
Start condition generation (retransmission)
SCL
9
SDA
ACK
Bit 7
IRIC
[5] ICDR write (transmit data)
[4] IRIC determination
[3] (Retransmission) Start condition instruction issuance
[2] Determination of SCL = Low
[1] IRIC determination

Figure 15.30 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 B′11 in ICXR.

9. Note on when I^2C bus interface stop condition instruction is issued

   In a situation where the rise time of the 9th clock of SCL exceeds the stipulated value because of a large bus load capacity or where a slave device in which a wait can be inserted by driving the SCL pin low is used, the stop condition instruction should be issued after reading SCL after the rise of the 9th clock pulse and determining that it is low.

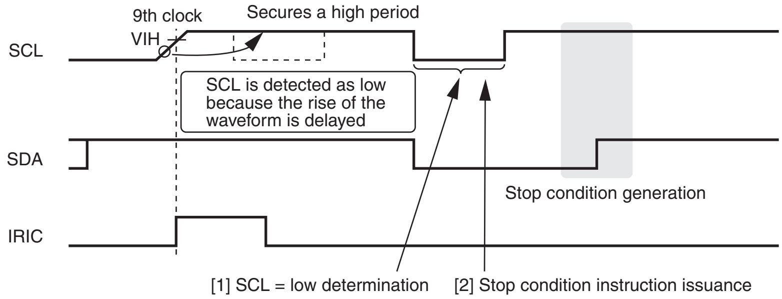

9th clock
Secures a high period
VIH
SCL
SCL is detected as low because the rise of the waveform is delayed
SDA
Stop condition generation
IRIC
[1] SCL = low determination
[2] Stop condition instruction issuance

Figure 15.31 Stop Condition Issuance Timing

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to B′11 in ICXR.

10. Note on IRIC flag clear when the wait function is used

    When the wait function is used in I^2C bus interface master mode and in a situation where the rise time of SCL exceeds the stipulated value or where a slave device in which a wait can be inserted by driving the SCL pin low is used, the IRIC flag should be cleared after determining that the SCL is low.

    If the IRIC flag is cleared to 0 when WAIT = 1 while the SCL is extending the high level time, the SDA level may change before the SCL goes low, which may generate a start or stop condition erroneously.

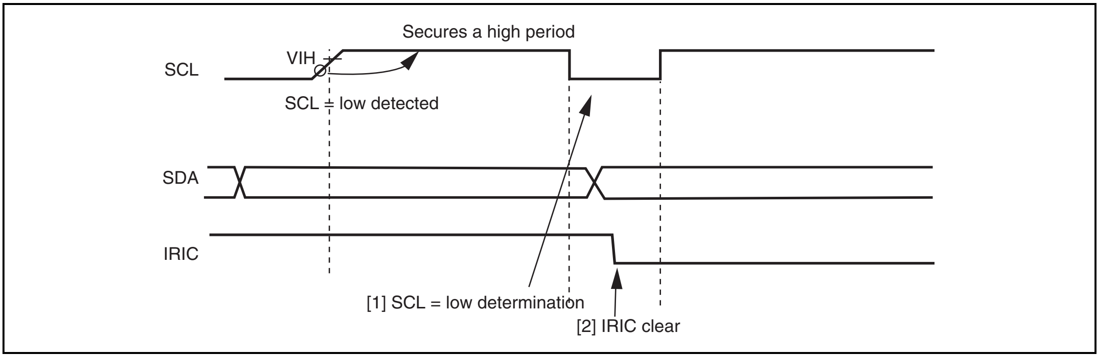


Figure 15.32 IRIC Flag Clearing Timing When WAIT = 1

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to B'11 in ICXR.

11. Note on ICDR register read and ICCR register 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 15.33. However, such read and write operations source 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 counter in ICMR; when the count is B'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.

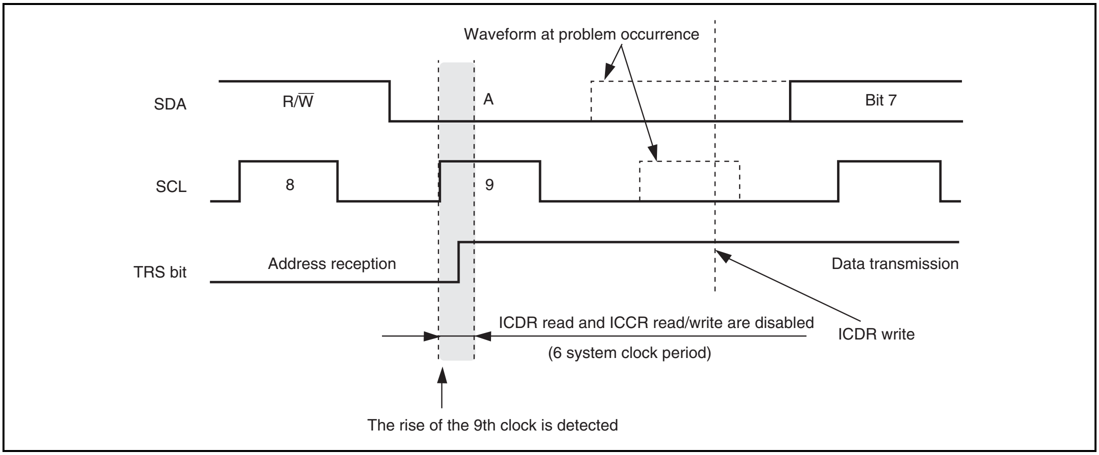

Waveform at problem occurrence
SDA
R/W̄
A
Bit 7
SCL
8
9
TRS bit
Address reception
Data transmission
ICDR read and ICCR read/write are disabled
(6 system clock period)
ICDR write
The rise of the 9th clock is detected

Figure 15.33 ICDR Register Read and ICCR Register Access Timing in Slave Transmit Mode

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to B'11 in ICXR.

12. Note on TRS bit setting in slave mode

In I²C 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 15.34), 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 15.34), 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 15.34. 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.

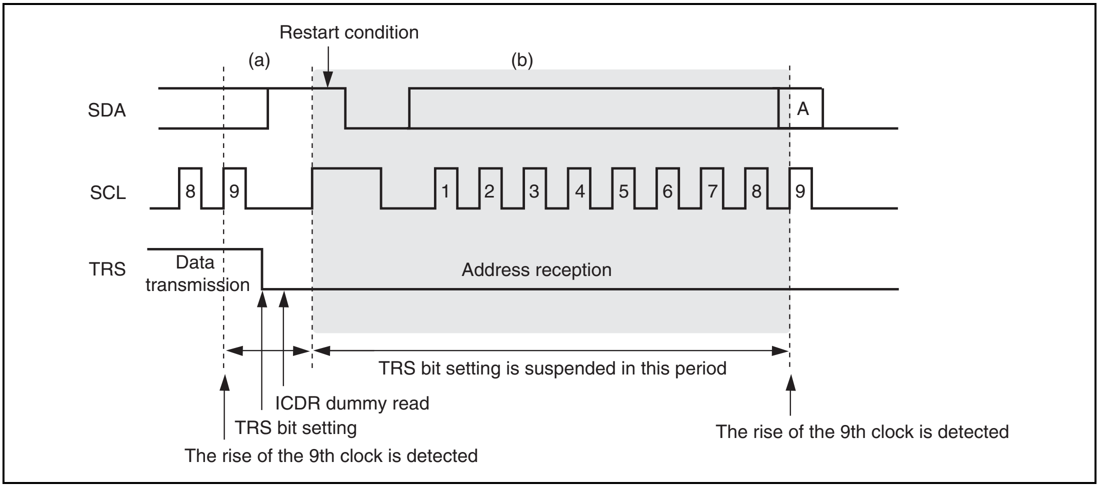

Restart condition
(a)
(b)
SDA
A
SCL
8
9
1
2
3
4
5
6
7
8
9
TRS
Data transmission
Address reception
TRS bit setting is suspended in this period
ICDR dummy read
TRS bit setting
The rise of the 9th clock is detected
The rise of the 9th clock is detected

Figure 15.34 TRS Bit Set Timing in Slave Mode

Note: This restriction on usage can be canceled by setting the FNC1 and FNC0 bits to B′11 in ICXR.

13. Note on ICDR read in transmit mode and ICDR write in receive mode

When ICDR is read in transmit mode (TRS = 1) or ICDR is written to in receive mode (TRS = 0), the SCL pin may not be held low in some cases after transmit/receive operation has been completed, thus inconveniently allowing clock pulses to be output on the SCL bus line before ICDR is accessed correctly. To access ICDR correctly, read the ICDR after setting receive mode or write to the ICDR after setting transmit mode.

14. Note on ACKE and TRS bits in slave mode

    In the I^{2}C bus interface, if 1 is received as the acknowledge bit value (ACKB = 1) in transmit mode (TRS = 1) and then the address is received in slave mode without performing appropriate processing, interrupt handling may start at the rising edge of the 9th clock pulse even when the address does not match. Similarly, if the start condition and address are transmitted from the master device in slave transmit mode (TRS = 1), the ICDRE flag is set, and 1 is received as the acknowledge bit value (ACKB = 1), the IRIC flag may be set thus causing an interrupt source even when the address does not match.

    To use the I^{2}C bus interface module in slave mode, be sure to follow the procedures below.

    — 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.

    — Set receive mode (TRS = 0) before the next start condition is input in slave mode. Complete transmit operation by the procedure shown in figure 15.23, in order to switch from slave transmit mode to slave receive mode.

15. Notes on Arbitration Lost in Master Mode Operation

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

    When arbitration is lost not in the first frame but in the second frame or subsequent frame, transmit/receive data that is not an address is compared with the value set in the SAR or SARX register as an address. If the receive data matches with the address in the SAR or SARX register, the I^{2}C bus interface erroneously recognizes that the address call has occurred. (See figure 15.35.)

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

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

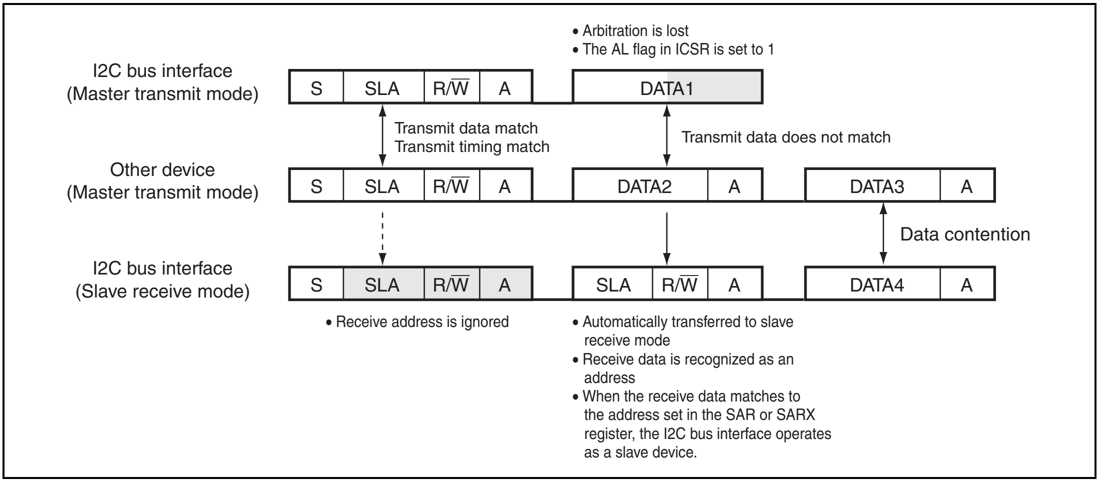

• Arbitration is lost
• The AL flag in ICSR is set to 1
I2C bus interface
(Master transmit mode)
S
SLA
R/$\overline{W}$
A
DATA1
Transmit data match
Transmit timing match
Transmit data does not match
Other device
(Master transmit mode)
S
SLA
R/$\overline{W}$
A
DATA2
A
DATA3
A
Data contention
I2C bus interface
(Slave receive mode)
S
SLA
R/$\overline{W}$
A
SLA
R/$\overline{W}$
A
DATA4
A
• Receive address is ignored
• Automatically transferred to slave receive mode
• Receive data is recognized as an address
• When the receive data matches to the address set in the SAR or SARX register, the I2C bus interface operates as a slave device.

Figure 15.35 Diagram of Erroneous Operation when Arbitration Lost

Though it is prohibited in the normal I^{2}C protocol, the same problem may occur when the MST bit is erroneously set to 1 and a transition to master mode is occurred during data transmission or reception in slave mode.

When the MST bit is set to 1 during data transmission or reception in slave mode, the arbitration decision circuit is enabled and arbitration is lost if conditions are satisfied. In this case, the transmit/receive data which is not an address may be erroneously recognized as an address.

In multi-master mode, pay attention to the setting of the MST bit when a bus conflict may occur. In this case, the MST bit in the ICCR register should be set to 1 according to the order below.

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

B. Set the MST bit to 1.

C. To confirm that the bus was not entered to the busy state while the MST bit is being set, check that the BBSY flag in the ICCR register is 0 immediately after the MST bit has been set.

Note: Above restrictions can be released by setting the bits FNC1 and FNC2 in ICXR to B'11.

Section 16 LPC Interface (LPC)

This LSI has an on-chip LPC interface.

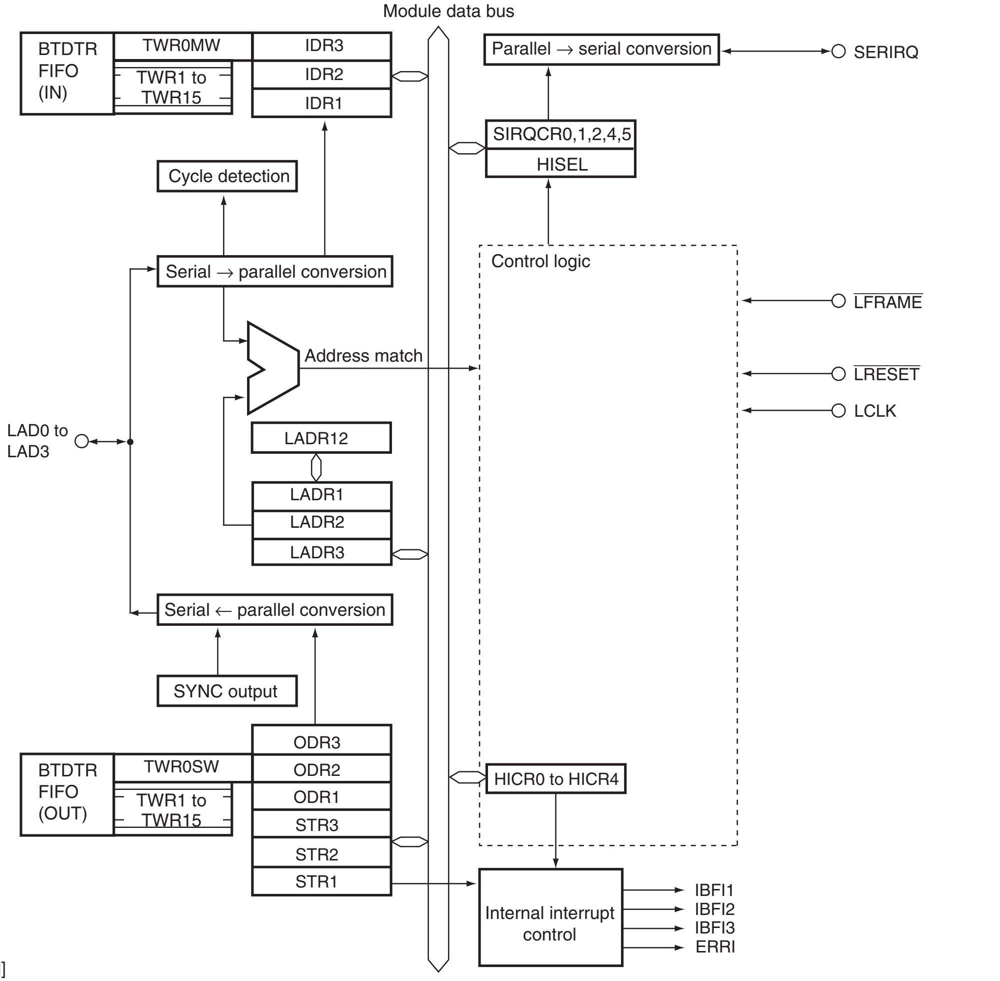
The LPC includes three register sets, each of which comprises data and status registers, control register, and the host interrupt request circuit.

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 I/O read and I/O write cycle transfers.

16.1 Features

- Supports LPC interface I/O read 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 ($\overline{\text{LRESET}}$), and frame ($\overline{\text{LFRAME}}$).
- 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).
  - I/O addresses from H'0000 to H'FFFF are selected for channels 1 to 3.
  - 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.
- Supports version 1.5 of the Intelligent Platform Management Interface (IPMI) specifications
  - Channel 3 supports the SMIC interface, KCS interface, and BT interface.

Figure 16.1 shows a block diagram of the LPC.



[Legend]

HICR0 to HICR4:	Host interface control registers 0 to 4	TWR0MW:	Bidirectional data register 0MW
LADR12H, LADR12L:	LPC channel 1, 2 address registers 12H and 12L	TWR0SW:	Bidirectional data register 0SW
LADR3H, LADR3L:	LPC channel 3 address registers 3H and 3L	TWR1 to TWR15:	Bidirectional data registers 1 to 15
IDR1 to IDR3:	Input data registers 1 to 3	SIRQCR0,1,2,4,5:	SERIRQ control registers 0,1,2,4,5
ODR1 to ODR3:	Output data registers 1 to 3	HISEL:	Host interface select register
STR1 to STR3:	Status registers 1 to 3		

Figure 16.1 Block Diagram of LPC

16.2 Input/Output Pins

Table 16.1 lists the LPC pin configuration.

Table 16.1 Pin Configuration

Name	Abbreviation	Port	I/O	Function
LPC address/ data 3 to 0	LAD3 to LAD0	PE to PE0	I/O	Cycle type/address/data signals serially (4-signal-line) transferred in synchronization with LCLK
LPC frame	$\overline{\text{LFRAME}}$	PE4	Input*	Transfer cycle start and forced termination signal
LPC reset	$\overline{\text{LRESET}}$	PE5	Input*	LPC interface reset signal
LPC clock	LCLK	PE6	Input	33-MHz PCI clock signal
Serialized interrupt request	SERIRQ	PE7	I/O*	Serialized host interrupt request signal (SMI, HIRQ1 to HIRQ15) in synchronization with LCLK

Note: * Pin state monitoring input is possible in addition to the LPC interface control input/output function.

16.3 Register Descriptions

The LPC has the following registers.

- Host interface control register 0 (HICR0)
- Host interface control register 1 (HICR1)
- Host interface control register 2 (HICR2)
- Host interface control register 3 (HICR3)
- Host interface control register 4 (HICR4)
- LPC channel 1, 2 address register H, L (LADR12H, LADR12L)
- LPC channel 3 address register H, L (LADR3H, LADR3L)
- Input data register 1 (IDR1)
- Input data register 2 (IDR2)
- Input data register 3 (IDR3)
- Output data register 1 (ODR1)
- Output data register 2 (ODR2)
- Output data register 3 (ODR3)
- Status register 1 (STR1)
- Status register 2 (STR2)
- Bidirectional data registers 0 to 15 (TWR0 to TWR15)
- SERIRQ control register 0 (SIRQCR0)
- SERIRQ control register 1 (SIRQCR1)
- SERIRQ control register 2 (SIRQCR2)
- SERIRQ control register 4 (SIRQCR4)
- SERIRQ control register 5 (SIRQCR5)
- Host interface select register (HISEL)

The following registers are necessary for SMIC mode

- SMIC flag register (SMICFLG)
- SMIC control/status register (SMICCSR)
- SMIC data register (SMICDTR)
- SMIC interrupt register 0 (SMICIR0)
- SMIC interrupt register 1 (SMICIR1)

The following registers are necessary for BT mode

- BT status register 0 (BTSR0)
- BT status register 1 (BTSR1)
- BT control/status register 0 (BTCSR0)
- BT control/status register 1 (BTCSR1)
- BT control register (BTCR)
- BT data buffer (BTDTR)
- BT interrupt mask register (BTIMSR)
- FIFO valid size register 0 (BTFVSR0)
- FIFO valid size register 1 (BTFVSR1)

16.3.1 Host Interface Control Registers 0 and 1 (HICR0 and HICR1)

HICR0 and HICR1 contain control bits that enable or disable LPC interface functions, control bits that determine pin output and the internal state of the LPC interface, and status flags that monitor the internal state of the LPC interface.

- HICR0

		Initial	R/W		
Bit	**Bit Name**	**Value**	**Slave**	**Host**	**Description**
7	LPC3E	0	R/W	—	LPC Enable 3 to 1
6	LPC2E	0	R/W	—	Enable or disable the LPC interface function. When the LPC interface is enabled (one of the three bits is set to 1), processing for data transfer between the slave (this LSI) and the host is performed using pins LAD3 to LAD0, $\overline{\text{LFRAME}}$, $\overline{\text{LRESET}}$, LCLK, and SERIRQ. • LPC3E 0: LPC channel 3 operation is disabled No address (LADR3) matches for IDR3, ODR3, STR3, TWR0 to TWR15, SMIC, KCS, or BT 1: LPC channel 3 operation is enabled • LPC2E 0: LPC channel 2 operation is disabled No address (LADR2) matches for IDR2, ODR2, or STR2 1: LPC channel 2 operation is enabled • LPC1E 0: LPC channel 1 operation is disabled No address (LADR1) matches for IDR1, ODR1, or STR1 1: LPC channel 1 operation is enabled
5	LPC1E	0	R/W	—	

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
4	—	0	R/W	—	Reserved The initial value should not be changed.
3	SDWNE	0	R/W	—	LPC Software Shutdown Enable Controls LPC interface shutdown. For details of the LPC shutdown function, and the scope of initialization by an LPC reset and an LPC shutdown, see section 16.4.5, LPC Interface Shutdown Function. 0: Normal state, LPC software shutdown setting enabled [Clearing conditions] • Writing 0 • LPC hardware reset or LPC software reset [Setting condition] Writing 1 after reading SDWNE = 0
2 to 0	—	All 0	R/W	—	Reserved The initial value should not be changed.

- HICR1

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
7	LPCBSY	0	R	—	LPC Busy Indicates that the LPC interface is processing a transfer cycle. 0: LPC 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 software shutdown • Forced termination (abort) of transfer cycle subject to processing • Normal termination of transfer cycle subject to processing 1: LPC interface is performing transfer cycle processing [Setting condition] Match of cycle type and address

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
6	CLKREQ	0	R	—	LCLK Request Indicates that the LPC 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 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 LPC interface's SERIRQ is engaged in transfer processing. 0: SERIRQ transfer frame wait state [Clearing conditions] • LPC hardware reset or LPC software reset • LPC software shutdown • End of SERIRQ transfer frame 1: SERIRQ transfer processing in progress [Setting condition] Start of SERIRQ transfer frame

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
4	LRSTB	0	R/W	—	LPC Software Reset Bit Resets the LPC interface. For the scope of initialization by an LPC reset, see section 16.4.5, LPC Interface Shutdown Function. 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 LPC interface shutdown. For details of the LPC shutdown function, and the scope of initialization by an LPC reset and an LPC shutdown, see section 16.4.5, LPC Interface Shutdown Function. 0: Normal state [Clearing conditions] • Writing 0 • LPC hardware reset or LPC software reset 1: LPC software shutdown state [Setting condition] Writing 1 after reading SDWNB = 0
2 to 0	—	All 0	R/W	—	Reserved The initial value should not be changed.

16.3.2 Host Interface Control Registers 2 and 3 (HICR2 and HICR3)

HICR2 controls interrupts to an LPC interface slave (this LSI). HICR3 monitors the states of the LPC interface pins. Bits 6 to 0 in HICR2 are initialized to H'00 by a reset. The states of other bits are decided by the pin states. The pin states can be monitored by the pin monitoring bits regardless of the LPC interface operating state or the operating state of the functions that use pin multiplexing.

- HICR2

		Initial	R/W		
Bit	**Bit Name**	**Value**	**Slave**	**Host**	**Description**
7	—	Undefined	R	—	Reserved
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 condition] Writing 0 after reading LRST = 1 1: [Setting condition] $\overline{\text{LRESET}}$ pin falling edge detection
5	—	0	R/(W)*	—	Reserved The initial value should not be changed.
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 ($\overline{\text{LRESET}}$ pin falling edge detection) • LPC software reset (LRSTB = 1) • LPC software shutdown (SDWNB = 1) 1: [Setting condition] $\overline{\text{LFRAME}}$ pin falling edge detection during LPC transfer cycle

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
3	IBFIE3	0	R/W	—	IDR3 and TWR Receive Complete interrupt Enable Enables or disables IBFI3 interrupt to the slave (this LSI). 0: Input data register (IDR3) and TWR receive complete interrupt requests and SMIC/BT mode interrupt requests disabled 1: [When TWRIE = 0 in LADR3] Input data register (IDR3) receive complete interrupt requests and SMIC/BT mode interrupt requests enabled [When TWRIE = 1 in LADR3] Input data register (IDR3) and TWR receive complete interrupt requests and SMIC/BT mode interrupt requests enabled
2	IBFIE2	0	R/W	—	IDR2 Receive Complete Interrupt Enable Enables or disables IBFI2 interrupt to the slave (this LSI). 0: Input data register (IDR2) receive complete interrupt requests disabled 1: Input data register (IDR2) receive complete interrupt requests enabled
1	IBFIE1	0	R/W	—	IDR1 Receive Complete Interrupt Enable Enables or disables IBFI1 interrupt to the slave (this LSI). 0: Input data register (IDR1) receive complete interrupt requests disabled 1: Input data register (IDR1) receive complete interrupt requests enabled

		Initial	R/W		
Bit	Bit Name	Value	Slave	Host	Description
0	ERRIE	0	R/W	—	Error Interrupt Enable Enables or disables ERRI interrupt to the slave (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	—	0: $\overline{\text{LFRAME}}$ Pin state is low level 1: $\overline{\text{LFRAME}}$ Pin state is high level
6	—	Undefined	R	—	Reserved
5	SERIRQ	Undefined	R	—	0: SERIRQ Pin state is low level 1: SERIRQ Pin state is high level
4	LRESET	Undefined	R	—	0: $\overline{\text{LRESET}}$ Pin state is low level 1: $\overline{\text{LRESET}}$ Pin state is high level
3	—	Undefined	R	—	Reserved
2	—	Undefined	R	—	Reserved
1	—	Undefined	R	—	Reserved
0	—	Undefined	R	—	Reserved

16.3.3 Host Interface Control Register 4 (HICR4)

HICR4 controls the operation of the KCS, SMIC, and BT interface functions on channel 3.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	LADR12SEL	0	R/W	—	Switches the channel accessed via LADR12H and LADR12L. 0: LADR1 is selected 1: LADR2 is selected
6	—	0	R/W	—	Reserved The initial value should not be changed.
5	CH2OFFSEL	0	R/W	—	Channel 2 Offset Selects the address offset for LPC channel 2. 0: Offset 4 1: Offset 1
4	CH1OFFSEL	0	R/W	—	Channel 1 Offset Selects the address offset for LPC channel 1. 0: Offset 4 1: Offset 1
3	SWENBL	0	R/W	—	In BT mode, H'5 (short wait) or H'6 (long wait) is returned to the host in the synchronized return cycle from slave, thus can make the host wait. 0: Short wait is issued 1: Long wait is issued
2	KCSENBL	0	R/W	—	Enables or disables the use of the KCS interface included in channel 3. When the LPC3E bit in HICR0 is 0, this bit is valid. 0: KCS interface operation is disabled No address (LADR3) matches for IDR3, ODR3, or STR3 in KCS mode 1: KCS interface operation is enabled

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
1	SMICENBL	0	R/W	—	Enables or disables the use of the SMIC interface included in channel 3. When the LPC3E bit in HICR0 is 0, this bit is valid. 0: SMIC interface operation is disabled No address (LADR3) matches for SMICFLG, SSMICCSR, or SMICDTR 1: SMIC interface operation is enabled
0	BTENBL	0	R/W	—	Enables or disables the use of the BT interface included in channel 3. When the LPC3E bit in HICR0 is 0, this bit is valid. 0: BT interface operation is disabled No address (LADR3) matches for BTIMSR, BTCR, or BTDTR 1: BT interface operation is enabled

16.3.4 LPC Channel 1, 2 Address Register H, L (LADR12H, LADR12L)

LADR12H and LADR12L are temporary registers for accessing internal registers LADR1H, LADR1L, LADR2H, and LADR2L.

When the LADR12SEL bit in HICR4 is 0, LPC channel 1 host addresses (LADR1H, LADR1L) are set through LADR12. The contents of the address field in LADR1 must not be changed while channel 1 is operating (while LPC1E is set to 1).

When the LADR12SEL bit is 1, LPC channel 2 host addresses (LADR2H, LADR2L) are set through LADR12. The contents of the address field in LADR2 must not be changed while channel 2 is operating (while LPC2E is set to 1).

Table 16.2 shows the initial value of each register. Table 16.3 shows the host register selection in address match determination. Table 16.4 shows the slave selection internal registers in slave (this LSI) access.

Table 16.2 LADR1, LADR2 Initial Values

Register Name	Initial Value	Description
LADR1	H'0060	I/O address of channel 1
LADR2	H'0062	I/O address of channel 2

Table 16.3 Host Register Selection

I/O Address					
Bits 15 to 3	**Bit 2**	**Bit 1**	**Bit 0**	**Transfer Cycle**	**Host Register Selection**
LADR1 (bits 15 to 3)	0	LADR1 (bit 1)	LADR1 (bit 0)	I/O write	IDR1 write (data), C/$\overline{\text{D}}$1 ← 0
LADR1 (bits 15 to 3)	1	LADR1 (bit 1)	LADR1 (bit 0)	I/O write	IDR1 write (command), C/$\overline{\text{D}}$1 ← 1
LADR1 (bits 15 to 3)	0	LADR1 (bit 1)	LADR1 (bit 0)	I/O read	ORD1 read
LADR1 (bits 15 to 3)	1	LADR1 (bit 1)	LADR1 (bit 0)	I/O read	STR1 read
LADR2 (bits 15 to 3)	0	LADR2 (bit 1)	LADR2 (bit 0)	I/O write	IDR2 write (data), C/$\overline{\text{D}}$2 ← 0
LADR2 (bits 15 to 3)	1	LADR2 (bit 1)	LADR2 (bit 0)	I/O write	IDR2 write (command), C/$\overline{\text{D}}$2 ← 1
LADR2 (bits 15 to 3)	0	LADR2 (bit 1)	LADR2 (bit 0)	I/O read	ODR2 read
LADR2 (bits 15 to 3)	1	LADR2 (bit 1)	LADR2 (bit 0)	I/O read	STR2 read

Table 16.4 Slave Selection Internal Registers

Slave (R/W)	Bus Width (B/W)	LADR12SEL	LADR12		Internal Register	
R/W	B	0	LADR12H		LADR1H	
R/W	B	1	LADR12H		LADR2H	
R/W	B	0		LADR12L		LADR1L
R/W	B	1		LADR12L		LADR2L
R/W	W	0	LADR12H	LADR12L	LADR1H	LADR1L
R/W	W	1	LADR12H	LADR12L	LADR2H	LADR2L

16.3.5 LPC Channel 3 Address Register H, L (LADR3H, LADR3L)

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

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 6 5 4 3 2 1 0	Bit 15 Bit 14 Bit 13 Bit 12 Bit 11 Bit 10 Bit 9 Bit 8	All 0	R/W	—	Channel 3 Address Bits 15 to 8 The host address of LPC channel 3 is set.

- LADR3L

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 6 5 4 3	Bit 7 Bit 6 Bit 5 Bit 4 Bit 3	All 0	R/W	—	Channel 3 Address Bits 7 to 3 The host address of LPC channel 3 is set.
2	—	0	R/W	—	Reserved The initial value should not be changed.
1	Bit 1	0	R/W	—	Channel 3 Address Bit 1 The host address of LPC channel 3 is set.
0	TWRE	0	R/W	—	Bidirectional Data Register Enable Enables or disables bidirectional data register operation. Clear this bit to 0 in KCS mode. 0: TWR operation is disabled TWR-related address (LADR3) match does not occur. 1: TWR operation is enabled

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 in 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. When determining an IDR3, ODR3, or STR3 address match in KCS mode, an SMICFLG, SMICCSR, SMICDTR address match in SMIC mode, and a BTDTR, BTCR, BTIMSR address match in BT mode, the values of bits 3 to 0 are ignored.

Register selection according to the bits ignored in address match determination is as shown in the following table.

I/O Address						Transfer	Host Register
Bits 15 to5	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Selection**
Bits 15 to5	Bit 4	Bit 3	0	Bit 1	0	I/O write	IDR3 write, C/$\overline{\text{D}}$3 ← 0
Bits 15 to5	Bit 4	Bit 3	1	Bit 1	0	I/O write	IDR3 write, C/$\overline{\text{D}}$3 ← 1
Bits 15 to5	Bit 4	Bit 3	0	Bit 1	0	I/O read	ODR3 read
Bits 15 to5	Bit 4	Bit 3	1	Bit 1	0	I/O read	STR3 read
Bits 15 to5	$\overline{\text{Bit 4}}$	0	0	0	0	I/O write	TWR0MW write
Bits 15 to5	$\overline{\text{Bit 4}}$	0 ● ● ● 1	0 ● ● ● 1	0 ● ● ● 1	1 ● ● ● 1	I/O write	TWR1 to TWR15 write
Bits 15 to5	$\overline{\text{Bit 4}}$	0	0	0	0	I/O read	TWR0SW read
Bits 15 to5	$\overline{\text{Bit 4}}$	0 ● ● ● 1	0 ● ● ● 1	0 ● ● ● 1	1 ● ● ● 1	I/O read	TWR1 to TWR15 read

- KCS mode

I/O Address						Transfer	
Bits 15 to5	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Host Register Selection**
Bits 15 to5	Bit 4	0	0	1	0	I/O write	IDR3 write, C/$\overline{\text{D}}$3 ← 0
Bits 15 to5	Bit 4	0	0	1	1	I/O write	IDR3 write, C/$\overline{\text{D}}$3 ← 1
Bits 15 to5	Bit 4	0	0	1	0	I/O read	ODR3 read
Bits 15 to5	Bit 4	0	0	1	1	I/O read	STR3 read

- BT mode

I/O Address						Transfer	
Bits 15 to5	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Host Register Selection**
Bits 15 to5	Bit 4	0	1	0	0	I/O write	BTCR write
Bits 15 to5	Bit 4	0	1	0	1	I/O write	BTDTR write
Bits 15 to5	Bit 4	0	1	1	0	I/O write	BTIMSR write
Bits 15 to5	Bit 4	0	1	0	0	I/O read	BTCR read
Bits 15 to5	Bit 4	0	1	0	1	I/O read	BTDTR read
Bits 15 to5	Bit 4	0	1	1	0	I/O read	BTIMSR read

- SMIC mode

I/O Address						Transfer	
Bits 15 to5	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Cycle**	**Host Register Selection**
Bits 15 to5	Bit 4	1	0	0	1	I/O write	SMICDTR write
Bits 15 to5	Bit 4	1	0	1	0	I/O write	SMICCSR write
Bits 15 to5	Bit 4	1	0	1	1	I/O write	SMICFLG write
Bits 15 to5	Bit 4	1	0	0	1	I/O read	SMICDTR read
Bits 15 to5	Bit 4	1	0	1	0	I/O read	SMICCSR read
Bits 15 to5	Bit 4	1	0	1	1	I/O read	SMICFLG read

16.3.6 Input Data Registers 1 to 3 (IDR1 to IDR3)

The IDR registers are 8-bit read-only registers to the slave processor (this LSI), and 8-bit write-only registers to the host processor. The registers selected from the host according to the I/O address are described in the following sections: for information on IDR1 and IDR2 selection, see section 16.3.4, LPC Channel 1, 2 Address Register H, L (LADR12H, LADR12L), and for information on IDR3 selection, see section 16.3.5, LPC Channel 3 Address Register H, L (LADR3H, LADR3L). 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 $C/\overline{D}$ bit in STR, to indicate whether the written information is a command or data.

The initial values of the IDR registers are undefined.

16.3.7 Output Data Registers 0 to 3 (ODR1 to ODR3)

The ODR registers are 8-bit readable/writable registers to the slave processor (this LSI), and 8-bit read-only registers to the host processor. The registers selected from the host according to the I/O address are described in the following sections: for information on ODR1 and ODR2 selection, see section 16.3.4, LPC Channel 1, 2 Address Register H, L (LADR12H, LADR12L), and for information on ODR3 selection, see section 16.3.5, LPC Channel 3 Address Register H, L (LADR3H, LADR3L). In an LPC I/O read cycle, the data in the selected register is transferred to the host.

The initial values of the ODR registers are undefined.

16.3.8 Bidirectional Data Registers 0 to 15 (TWR0 to TWR15)

TWR0 to TWR15 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 to the host processor, and a read-only register to the slave processor, while TWR0SW is a write-only register to the slave processor and a read-only register to 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 16.3.5, LPC Channel 3 Address Register H, L (LADR3H, LADR3L).

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.

16.3.9 Status Registers 1 to 3 (STR1 to STR3)

The STR registers are 8-bit registers that indicate status information during LPC interface processing. Bits 3, 1, and 0 in STR1 to STR3 are read-only bits to both the host processor and the slave processor (this LSI). However, 0 only can be written from the slave processor (this LSI) to bit 0 in STR1 to STR3, and bits 6 and 4 in STR3, in order to clear the flags to 0. The functions for bits 7 to 4 in STR3 differ according to the settings of bit SELSTR3 in HISEL and the TWRE bit in LADR3L. For details, see section 16.3.15, Host Interface Select Register (HISEL). The registers selected from the host processor according to the I/O address are described in the following sections. For information on STR1 and STR2 selection, see section 16.3.4, LPC Channel 1,2 Address Register H, L (LADR12H, LADR12L), and information on STR3 selection, see section 16.3.5, LPC Channel 3 Address Register H, L (LADR3H, LADR3L). In an LPC I/O read cycle, the data in the selected register is transferred to the host processor.

The STR registers are initialized to H'00 by a reset or in hardware standby mode.

- STR1

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	DBU17	All 0	R/W	R	Defined by User The user can use these bits as necessary.
6	DBU16				
5	DBU15				
4	DBU14				
3	C/$\overline{D}$1	0	R	R	Command/Data When the host processor writes to an IDR1 register, bit 2 of the I/O address (when CH1OFFSEL1 = 0) or bit 0 of the I/O address (when CH1OFFSEL1 = 1) is written to this bit to indicate whether IDR1 contains data or a command. 0: Content of input data register (IDR1) is data 1: Content of input data register (IDR1) is a command
2	DBU12	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	IBF1	0	R	R	Input Data Register Full Indicates whether or not there is receive data in IDR1. This bit is an internal interrupt source to the slave processor (this LSI). 0: There is not receive data in IDR1 [Clearing condition] When the slave processor reads IDR 1: There is receive data in IDR1 [Setting condition] When the host processor writes to IDR using I/O write cycle
0	OBF1	0	R/(W)*	R	Output Data Register Full Indicates whether or not there is transmit data in ODR1. 0: There is not transmit data in ODR1 [Clearing condition] When the host processor reads ODR1 using I/O read cycle, or the slave processor writes 0 to the OBF1 bit 1: There is transmit data in ODR1 [Setting condition] When the slave processor writes to ODR1

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
6	DBU26	0	R/W	R	The user can use these bits as necessary.
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 writes to IDR2, bit 2 of the I/O address (when CH2OFFSEL1 = 0) or bit 0 of the I/O address (when CH2OFFSEL1 = 1) is written to this bit to indicate whether IDR2 contains data or a command. 0: Content of input data register (IDR2) is a data 1: Content of input data register (IDR2) is 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 Data Register Full Indicates whether or not there is receive data in IDR2.This bit is an internal interrupt source to the slave (this LSI). 0: There is not receive data in IDR2. [Clearing condition] When the slave reads IDR2 1: There is receive data in IDR2. [Setting condition] When the host writes to IDR2 in an I/O write cycle

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
0	OBF2	0	R/(W)*	R	Output Data Register Full Indicates whether or not there is transmit data in ODR2. 0: There is not transmit data in ODR2 [Clearing conditions] • When the host reads ODR2 in an I/O read cycle • When the slave writes 0 to bit OBF2 1: There is transmit data in ODR2 [Setting condition] • When the slave writes to ODR2

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 Flag This is an internal interrupt source to the slave (this LSI). 0: [Clearing condition] When the slave reads TWR15 1: [Setting condition] When the host writes to TWR15 in I/O write cycle
6	OBF3B	0	R/(W)*	R	Bidirectional Data Register Output Buffer Full Flag 0: [Clearing conditions] • When the host reads TWR15 in I/O read cycle • When the slave writes 0 to the OBF3B bit 1: [Setting condition] When the slave writes to TWR15
5	MWMF	0	R	R	Master Write Mode Flag 0: [Clearing condition] When the slave reads TWR15 1: [Setting condition] When the host writes to TWR0 in I/O write cycle while SWMF = 0
4	SWMF	0	R/(W)*	R	Slave Write Mode Flag In the event of simultaneous writes by the master and the slave, the master write has priority. 0: [Clearing conditions] • When the host reads TWR15 in I/O read cycle • When the slave writes 0 to the SWMF bit 1: [Setting condition] When the slave writes to TWR0 while MWMF = 0

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
3	$C/\overline{D}3$	0	R	R	Command/Data Flag When the host writes to IDR3, bit 2 of the I/O address is written into this bit to indicate whether IDR3 contains data or a command. 0: Content of input data register (IDR3) is a data. 1: Content of input data register (IDR3) is 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 Data Register Full Indicates whether or not there is receive data in IDR3. This is an internal interrupt source to the slave (this LSI). 0: There is not receive data in IDR3. [Clearing condition] When the slave reads IDR3 1: There is receive data in IDR3. [Setting condition] When the host writes to IDR3 in an I/O write cycle
0	OBF3A	0	R/(W)*	R	Output Data Register Full Indicates whether or not there is transmit data in ODR3. 0: There is not transmit data in ODR3. [Clearing conditions] • When the host reads ODR3 in an I/O read cycle • When the slave writes 0 to bit OBF3A 1: There is transmit data in ODR3. [Setting condition] • When the slave writes to ODR3

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
6	DBU36	0	R/W	R	The user can use these bits as necessary.
5	DBU35	0	R/W	R	
4	DBU34	0	R/W	R	
3	C/$\overline{D}$3	0	R	R	Command/Data Flag When the host writes to IDR3, bit 2 of the I/O address is written into this bit to indicate whether IDR3 contains data or a command. 0: Content of input data register (IDR3) is a data. 1: Content of input data register (IDR3) is 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 Data Register Full Indicates whether or not there is receive data in IDR3. This bit is an internal interrupt source to the slave (this LSI). 0: There is not receive data in IDR3. [Clearing condition] When the slave reads IDR3 1: There is receive data in IDR3. [Setting condition] When the host writes to IDR3 in an I/O write cycle

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	OBF3A	0	R/(W)*	R	Output Data Register Full Indicates whether or not there is transmit data in ODR3. 0: There is not receive data in ODR3. [Clearing conditions] • When the host reads ODR3 in an I/O read cycle • When the slave writes 0 to bit OBF3A 1: There is receive data in ODR3. [Setting condition] • When the slave writes to ODR3

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

16.3.10 SERIRQ Control Register 0 (SIRQCR0)

SIRQCR0 contains status bits that indicate the SERIRQ operating mode and bits that specify SERIRQ interrupt sources.

			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 the condition of a start frame initiation request when a host interrupt request is cleared in quiet mode. 0: Start frame initiation is requested when all interrupt requests are cleared. 1: Start frame initiation is requested when one or more interrupt requests are cleared.
5	IEDIR2	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 and corresponding OBF bits 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 an 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 IEDIR3 = 0) 1: [When IEDIR3 = 0] Host SMI interrupt request by setting OBF3B to 1 is enabled. [When IEDIR3 = 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 an 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 IEDIR3 = 0) 1: [When IEDIR3 = 0] Host SMI interrupt request by setting is enabled [When IEDIR3 = 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 an 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 IEDIR2 = 0) 1: [When IEDIR2 = 0] Host SMI interrupt request by setting OBF2 to 1 is enabled. [When IEDIR2 = 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 an HIRQ12 interrupt request when OBF1 is set by an ODR1 write. 0: HIRQ12 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: HIRQ12 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 HIRQ1 interrupt request when OBF1 is set by an ODR1 write. 0: HIRQ1 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: HIRQ1 interrupt request by setting OBF1 to 1 is enabled [Setting condition] Writing 1 after reading IRQ1E1 = 0

16.3.11 SERIRQ Control Register 1 (SIRQCR1)

SIRQCR1 contains status bits that indicate the SERIRQ operating mode and bits that specify SERIRQ interrupt sources.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	IRQ11E3	0	R/W	—	Host IRQ11 Interrupt Enable 3 Enables or disables an HIRQ11 interrupt request when OBF3A is set by an ODR3 write. 0: HIRQ11 interrupt request by OBF3A and IRQE11E3 is disabled [Clearing conditions] • Writing 0 to IRQ11E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR3 = 0) 1: [When IEDIR3 = 0] HIRQ11 interrupt request by setting OBF3A to 1 is enabled [When IEDIR3 = 1] HIRQ11 interrupt is requested [Setting condition] Writing 1 after reading IRQ11E3 = 0
6	IRQ10E3	0	R/W	—	Host IRQ10 Interrupt Enable 3 Enables or disables an HIRQ10 interrupt request when OBF3A is set by an ODR3 write. 0: HIRQ10 interrupt request by OBF3A and IRQE10E3 is disabled. [Clearing conditions] • Writing 0 to IRQ10E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR3 = 0) 1: [When IEDIR3 = 0] HIRQ10 interrupt request by setting OBF3A to 1 is enabled. [When IEDIR3 = 1] HIRQ10 interrupt is requested. [Setting condition] Writing 1 after reading IRQ10E3 = 0

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
5	IRQ9E3	0	R/W	—	Host IRQ9 Interrupt Enable 3 Enables or disables an HIRQ9 interrupt request when OBF3A is set by an ODR3 write. 0: HIRQ9 interrupt request by OBF3A and IRQE9E3 is disabled. [Clearing conditions] • Writing 0 to IRQ9E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR3 = 0) 1: [When IEDIR3 = 0] HIRQ9 interrupt request by setting OBF3A to 1 is enabled. [When IEDIR3 = 1] HIRQ9 interrupt is requested. [Setting condition] Writing 1 after reading IRQ9E3 = 0
4	IRQ6E3	0	R/W	—	Host IRQ6 Interrupt Enable 3 Enables or disables an HIRQ6 interrupt request when OBF3A is set by an ODR3 write. 0: HIRQ6 interrupt request by OBF3A and IRQE6E3 is disabled. [Clearing conditions] • Writing 0 to IRQ6E3 • LPC hardware reset, LPC software reset • Clearing OBF3A to 0 (when IEDIR3 = 0) 1: [When IEDIR3 = 0] HIRQ6 interrupt request by setting OBF3A to 1 is enabled. [When IEDIR3 = 1] HIRQ6 interrupt is requested [Setting condition] Writing 1 after reading IRQ6E3 = 0

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
3	IRQ11E2	0	R/W	—	Host IRQ11 Interrupt Enable 2 Enables or disables an HIRQ11 interrupt request when OBF2 is set by an oDR2 write. 0: HIRQ11 interrupt request by OBF2 and IRQE11E2 is disabled. [Clearing conditions] • Writing 0 to IRQ11E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR2 = 0) 1: [When IEDIR2 = 0] HIRQ11 interrupt request by setting OBF2 to 1 is enabled. [When IEDIR2 = 1] HIRQ11 interrupt is requested. [Setting condition] Writing 1 after reading IRQ11E2 = 0
2	IRQ10E2	0	R/W	—	Host IRQ10 Interrupt Enable 2 Enables or disables an HIRQ10 interrupt request when OBF2 is set by an ODR2 write. 0: HIRQ10 interrupt request by OBF2 and IRQE10E2 is disabled. [Clearing conditions] • Writing 0 to IRQ10E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR2 = 0) 1: [When IEDIR2 = 0] HIRQ10 interrupt request by setting OBF2 to 1 is enabled. [When IEDIR2 = 1] HIRQ10 interrupt is requested. [Setting condition] Writing 1 after reading IRQ10E2 = 0

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
1	IRQ9E2	0	R/W	—	Host IRQ9 Interrupt Enable 2 Enables or disables an HIRQ9 interrupt request when OBF2 is set by an oDR2 write. 0: HIRQ9 interrupt request by OBF2 and IRQE9E2 is disabled. [Clearing conditions] • Writing 0 to IRQ9E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR2 = 0) 1: [When IEDIR2 = 0] HIRQ9 interrupt request by setting OBF2 to 1 is enabled. [When IEDIR2 = 1] HIRQ9 interrupt is requested [Setting condition] Writing 1 after reading IRQ9E2 = 0
0	IRQ6E2	0	R/W	—	Host IRQ6 Interrupt Enable 2 Enables or disables an HIRQ6 interrupt request when OBF2 is set by an oDR2 write. 0: HIRQ6 interrupt request by OBF2 and IRQE6E2 is disabled. [Clearing conditions] • Writing 0 to IRQ6E2 • LPC hardware reset, LPC software reset • Clearing OBF2 to 0 (when IEDIR2 = 0) 1: [When IEDIR2 = 0] HIRQ6 interrupt request by setting OBF2 to 1 is enabled. [When IEDIR2 = 1] HIRQ6 interrupt is requested. [Setting condition] Writing 1 after reading IRQ6E2 = 0

16.3.12 SERIRQ Control Register 2 (SIRQCR2)

SIRQCR2 contains bits that enable or disable SERIRQ interrupt requests and select the host interrupt request outputs.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	IEDIR3	0	R/W	—	Interrupt Enable Direct Mode 3 Selects whether an SERIRQ interrupt generation of LPC channel 3 is affected only by a host interrupt enable bit or by an OBF flag in addition to the enable bit. 0: A host interrupt is generated when both the enable bit and the corresponding OBF flag are set. 1: A host interrupt is generated when the enable bit is set.
6 to 0	—	All 0	R/W	—	Reserved The initial value should not be changed.

16.3.13 SERIRQ Control Register 4 (SIRQCR4)

SIRQCR4 controls LPC interrupt requests to the host.

		Initial	R/W		
Bit	Bit Name	Value	Slave	Host	Description
7	IRQ15E	0	R/W	—	Host IRQ15 Interrupt Enable 0: Disables HIRQ15 interrupt request by IRQ15E 1: Enables HIRQ15 interrupt request
6	IRQ14E	0	R/W	—	Host IRQ14 Interrupt Enable 0: Disables HIRQ14 interrupt request by IRQ14E 1: Enables HIRQ14 interrupt request
5	IRQ13E	0	R/W	—	Host IRQ13 Interrupt Enable 0: Disables HIRQ13 interrupt request by IRQ13E 1: Enables HIRQ13 interrupt request
4	IRQ8	0	R/W	—	Host IRQ8 Interrupt Enable 0: Disables HIRQ8 interrupt request by IRQ8E 1: Enables HIRQ8 interrupt request
3	IRQ7	0	R/W	—	Host IRQ7 Interrupt Enable 0: Disables HIRQ7 interrupt request by IRQ7E 1: Enables HIRQ7 interrupt request
2	IRQ5	0	R/W	—	Host IRQ5 Interrupt Enable 0: Disables HIRQ5 interrupt request by IRQ5E 1: Enables HIRQ5 interrupt request
1	IRQ4	1	R/W	—	Host IRQ4 Interrupt Enable 0: Disables HIRQ4 interrupt request by IRQ4E 1: Enables HIRQ4 interrupt request
0	IRQ3	1	R/W	—	Host IRQ3 Interrupt Enable 0: Disables HIRQ3 interrupt request by IRQ3E 1: Enables HIRQ3 interrupt request

16.3.14 SERIRQ Control Register 5 (SIRQCR5)

SIRQCR5 selects the output of the host interrupt request signal of each frame.

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
7	SELIRQ15	0	R/W	—	SERIRQ Output Select These bits select the state of the output on the pin for LPC host interrupt requests (HIRQ15, HIRQ14, HIRQ13, HIRQ8, HIRQ7, HIRQ5, HIRQ4, and HIRQ3). 0: [When host interrupt request is cleared] SERIRQ pin output is in the Hi-Z state. [When host interrupt request is set] SERIRQ pin output is low. 1: [When host interrupt request is cleared] SERIRQ pin output is low. [When host interrupt request is set] SERIRQ pin output is in the Hi-Z state.
6	SELIRQ14	0	R/W	—	
5	SELIRQ13	0	R/W	—	
4	SELIRQ8	0	R/W	—	
3	SELIRQ7	0	R/W	—	
2	SELIRQ5	0	R/W	—	
1	SELIRQ4	0	R/W	—	
0	SELIRQ3	0	R/W	—	

16.3.15 Host Interface Select Register (HISEL)

HISEL selects the function of bits 7 to 4 in STR3 and selects the output of the host interrupt request signal of each frame.

		Initial	R/W		
Bit	**Bit Name**	**Value**	**Slave**	**Host**	**Description**
7	SELSTR3	0	R/W	—	Status Register 3 Selection Selects the function of bits 7 to 4 in STR3 in combination with the TWRE bit in LADR3L. For details of STR3, see section 16.3.9, Status Registers 1 to 3 (STR1 to STR3). 0: Bits 7 to 4 in STR3 indicate processing status of the LPC interface. 1: [When TWRE = 1] Bits 7 to 4 in STR3 indicate processing status of the LPC interface. [When TWRE = 0] Bits 7 to 4 in STR3 are readable/writable bits which user can use as necessary.
6	SELIRQ11	0	R/W	—	Host IRQ Interrupt Select These bits select the state of the output on the SERIRQ pin. 0: [When host interrupt request is cleared] SERIRQ pin output is in the Hi-Z state. [When host interrupt request is set] SERIRQ pin output is low. 1: [When host interrupt request is cleared] SERIRQ pin output is low. [When host interrupt request is set] SERIRQ pin output is in the Hi-Z state.
5	SELIRQ10	0	R/W	—	
4	SELIRQ9	0	R/W	—	
3	SELIRQ6	0	R/W	—	
2	SELSMI	0	R/W	—	
1	SELIRQ12	1	R/W	—	
0	SELIRQ1	1	R/W	—	

16.3.16 SMIC Flag Register (SMICFLG)

SMICFLG is one of the registers used to implement SMIC mode. This register includes bits that indicate whether or not the system is ready to data transfer and those that are used for handshake of the transfer cycles.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	RX_DATA_RDY	0	R/W	R	Read Transfer Ready Indicates whether or not the slave is ready for the host read transfer. 0: Slave waits for ready status. 1: Slave is ready for the host read transfer
6	TX_DATA_RDY	0	R/W	R	Write Transfer Ready Indicates whether or not the slave is ready for the host next write transfer. 0: The slave waits for ready status. 1: The slave is ready for the host write transfer.
5	—	0	R/W	R	Reserved The initial value should not be changed.
4	SMI	0	R/W	R	SMI Flag This bit indicates that the SMI is asserted. 0: Indicates waiting for SMI assertion. 1: Indicates SMI assertion.
3	SEVT_ATN	0	R/W	R	Event Flag When the slave detects an event for the host, this bit is set. 0: Indicates waiting for event detection. 1: Indicates event detection.
2	SMS_ATN	0	R/W	R	SMS Flag When there is a message to be transmitted from the slave to the host, this bit is set. 0: There is not a message. 1: There is a message.

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
1	—	0	R/W	R	Reserved The initial value should not be changed.
0	BUSY	0	R/(W)*	W	SMIC Busy This bit indicates that the slave is now transferring data. This bit can be cleared only by the slave and set only by the host. The rising edge of this bit is a source of internal interrupt to the slave. 0: Transfer cycle wait state [Clearing conditions] After the slave reads BUSY = 1, writes 0 to this bit. 1: Transfer cycle in progress [Setting condition] When the host writes 1 to this bit.

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

16.3.17 SMIC Control Status Register (SMICCSR)

SMICCSR is one of the registers used to implement SMIC mode. This is an 8-bit readable/writable register that stores a control code issued from the host and a status code that is returned from the slave.

The control code is written to this register accompanied by the transfer between the host and slave. The status code is returned to this register to indicate that the slave has recognized the control code, and a specified transfer cycle has been completed.

16.3.18 SMIC Data Register (SMICDTR)

SMICDTR is one of the registers used to implement SMIC mode. This is an 8-bit register that is accessible (readable/writable) from both the slave processor (this LSI) and host processor. This is used for data transfer between the host and slave.

16.3.19 SMIC Interrupt Register 0 (SMICIR0)

SMICIR0 is one of the registers used to implement SMIC mode. This register includes the bits that indicate the source of interrupt to the slave.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 to 5	—	All 0	R/W	—	Reserved The initial value should not be changed.
4	HDTWI	0	R/(W)*	—	Transfer Data Transmission End Interrupt This is a status flag that indicates that the host has finished transmitting the transfer data to SMICDTR. When the IBFIE3 bit and HDTWIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: Transfer data transmission wait state [Clearing condition] After the slave reads HDTWI = 1, writes 0 to this bit. 1: Transfer data transmission end [Setting condition] The transfer cycle is write transfer and the host writes the transfer data to SMICDTR.
3	HDTRI	0	R/(W)*	—	Transfer Data Receive End Interrupt This is a status flag that indicates that the host has finished receiving the transfer data from SMICDTR. When the IBFIE3 bit and HDTRIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: Transfer data receive wait state [Clearing condition] After the slave reads HDTRI = 1, writes 0 to this bit. 1: Transfer data receive end [Setting condition] The transfer cycle is read transfer and the host reads the transfer data from SMICDTR.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
2	STARI	0	R/(W)*	—	Status Code Receive End Interrupt This is a status flag that indicates that the host has finished receiving the status code from SMICCSR. When the IBFIE3 bit and STARIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: Status code receive wait state [Clearing condition] After the slave reads STARI = 1, writes 0 to this bit. 1: Status code receive end [Setting condition] When the host reads the status code of SMICCSR.
1	CTLWI	0	R/(W)*	—	Control Code Transmission End Interrupt This is a status flag that indicates that the host has finished transmitting the control code to SMICCSR. When the IBFIE3 bit and CTLWIE bit are set to1, the IBFI3 interrupt is requested to the slave. 0: Control code transmission wait state [Clearing condition] After the slave reads CTLWI = 1, writes 0 to this bit. 1: Control code transmission end [Setting condition] When the host writes the status code to SMICCSR.
0	BUSYI		R/(W)*	—	Transfer Start Interrupt This is a status flag that indicates that the host starts transferring. When the IBFIE3 bit and BUSYIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: Transfer start wait state [Clearing condition] After the slave reads BUSYI = 1, writes 0 to this bit. 1: Transfer start [Setting condition] When the rising edge of the BUSY bit in SMICFLG is detected.

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

16.3.20 SMIC Interrupt Register 1 (SMICIR1)

SMICIR1 is one of the registers used to implement SMIC mode. This register includes the bits that enables/disables an interrupt to the slave. The IBFI3 interrupt is enabled by setting the IBFIE3 bit in HICR2 to 1.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 to 5	—	All 0	R/W	—	Reserved The initial value should not be changed.
4	HDTWIE	0	R/W	—	Transfer Data Transmission End Interrupt Enable Enables or disables HDTWI interrupt that is IBFI3 interrupt source to the slave. 0: Disables transfer data transmission end interrupt. 1: Enables transfer data transmission end interrupt.
3	HDTRIE	0	R/W	—	Transfer Data Receive End Interrupt Enable Enables or disables HDTRI interrupt that is IBFI3 interrupt source to the slave. 0: Disables transfer data receive end interrupt. 1: Enables transfer data receive end interrupt.
2	STARIE	0	R/W	—	Status Code Receive End Interrupt Enable Enables or disables STARI interrupt that is IBFI3 interrupt source to the slave. 0: Disables status code receive end interrupt. 1: Enables status code receive end interrupt.
1	CTLWIE	0	R/W	—	Control Code Transmission End Interrupt Enable Enables or disables CTLWI interrupt that is IBFI3 interrupt source to the slave. 0: Disables control code transmission end interrupt. 1: Enables control code transmission end interrupt.
0	BUSYIE	0	R/W	—	Transfer Start Interrupt Enable Enables or disables BUSYI interrupt that is IBFI3 interrupt source to the slave. 0: Disables transfer start interrupt. 1: Enables transfer start interrupt.

16.3.21 BT Status Register 0 (BTSR0)

BTSR0 is one of the registers used to implement BT mode. This register includes flags that control interrupts to the slave (this LSI).

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
7 to 5	—	All 0	R/W	—	Reserved The initial value should not be changed.
4	FRDI	0	R/(W)*	—	FIFO Read Request Interrupt This status flag indicates that host writes the data to BTDTR buffer with FIFO full state at the host write transfer. When the IBFIE3 bit and FRDIE bit are set to 1, IBFI3 interrupt is requested to the slave. The slave must clear the flag after creating an unused area by reading the data in FIFO. 0: FIFO read is not requested [Clearing condition] After the slave reads FRDI = 1, writes 0 to this bit. 1: FIFO read is requested. [Setting condition] After the host processor transfers data, the host writes the data with FIFO Full state.
3	HRDI	0	R/(W)*	—	BT Host Read Interrupt This status flag indicates that the host reads 1 byte from BTDTR buffer. When the IBFIE3 bit and HRDIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: Host BTDTR read wait state [Clearing condition] After the slave reads HRDI = 1, writes 0 to this bit. 1: The host reads from BTDTR. [Setting condition] The host reads one byte from BTDTR.

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
2	HWRI	0	R/(W)*	—	BT Host Write Interrupt This status flag indicates that the host writes 1byte to BTDTR buffer. When the IBFIE3 bit and HWRIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: Host BTDTR write wait state [Clearing condition] After the slave reads HWRI = 1, writes 0 to this bit. 1: The host writes to BTDTR [Setting condition] The host writes one byte to BTDTR.
1	HBTWI	0	R/(W)*	—	BTDTR Host Write Start Interrupt This status flag indicates that the host writes the first byte of valid data to BTDTR buffer. When the IBFIE3 bit and HBTWIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: BTDTR host write start wait state [Clearing condition] After the slave reads HBTWI = 1 and writes 0 to this bit. 1: BTDTR host write start [Setting condition] The host starts writing valid data to BTDTR.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
0	HBTRI	0	R/(W)*	—	BTDTR Host Read End Interrupt This status flag indicates that the host reads all valid data from BTDTR buffer. When the BFIE3 bit and HBTRIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: BTDTR host read end wait state [Clearing condition] After the slave reads HBTRI = 1 and writes 0 to this bit. 1: BTDTR host read end [Setting condition] When the host finished reading the valid data from BTDTR.

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

16.3.22 BT Status Register 1 (BTSR1)

BTSR1 is one of the registers used to implement the BT mode. This register includes a flag that controls an interrupt to the slave (this LSI).

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	—	0	R/W	—	Reserved The initial value should not be changed.
6	HRSTI	0	R/(W)*	—	BT Reset Interrupt This status flag indicates that the BMC_HWRST bit in BTIMSR is set to 1 by the host. When the IBFIE3 bit and HRSTIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: [Clearing condition] When the slave reads HRSTI = 1 and writes 0 to this bit. 1: [Setting condition] When the slave detects the rising edge of BMC_HWRST.
5	IRQCRI	0	R/(W)*	—	B2H_IRQ Clear Interrupt This status flag indicates that the B2H_IRQ bit in BTIMSR is cleared by the host. When the IBFIE3 bit and IRQCRIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: [Clearing condition] When the slave reads IRQCRI = 1 and writes 0 to this bit. 1: [Setting condition] When the slave detects the falling edge of B2H_IRQ.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
4	BEVTI	0	R/(W)*	—	BEVT_ATN Clear Interrupt This status flag indicates that the BEVT_ATN bit in BTCR is cleared by the host. When the IBFIE3 bit and BEVTIE bit are set to 1, IBFI3 interrupt is requested to the slave. 0: [Clearing condition] When the slave reads BEVTI = 1 and writes 0 to this bit. 1: [Setting condition] When the slave detects the falling edge of BEVT_ATN.
3	B2HI	0	R/(W)*	—	Read End Interrupt This status flag indicates that the host has finished reading all data from the BTDTR buffer. When the IBFIE3 bit and B2HIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: [Clearing condition] When the slave reads B2HI = 1 and writes 0 to this bit. 1: [Setting conditions] When the slave detects the falling edge of B2H_ATN.
2	H2BI	0	R/(W)*	—	Write End Interrupt This status flag indicates that the host has finished writing all data to the BTDTR buffer. When the IBFIE3 bit and H2BIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: [Clearing condition] After the slave reads H2BI = 1, writes 0 to this bit. 1: [Setting condition] When the slave detects the falling edge of H2B_ATN.

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
1	CRRPI	0	R/(W)*	—	Read Pointer Clear Interrupt This status flag indicates that the CLR_RD_PTR bit in BTCR is set to 1 by the host. When the IBFIE3 bit and CRRPIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: [Clearing condition] After the slave reads CRRPI = 1, writes 0 to this bit. 1: [Setting condition] When the slave detects the rising edge of CLR_RD_PTR.
0	CRWPI	0	R/(W)*	—	Write Pointer Clear Interrupt This status flag indicates that the CLR_WR_PTR bit in BTCR is set to 1 by the host. When the IBFIE3 bit and CRWPIE bit are set to 1, the IBFI3 interrupt is requested to the slave. 0: [Clearing condition] After the slave reads CRWPI = 1, writes 0 to this bit. 1: [Setting condition] When the slave detects the rising edge of CLR_WR_PTR.

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

16.3.23 BT Control Status Register 0 (BTCSR0)

BTCSR0 is one of the registers used to implement the BT mode. The BTCSR0 register contains the bits used to switch FIFOs in BT transfer, and enable or disable the interrupts to the slave (this LSI). The IBFI3 interrupt is enabled by setting the IBFIE3 bit in HICR2 to 1.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	—	0	R/W	—	Reserved The initial value should not be changed.
6 5	FSEL1 FSEL0	0 0	R/W R/W	— —	These bits select either FIFO during BT transfer FSEL1 FSEL0 0 X :FIFO disabled 1 X :FIFO enabled The FIFO size: 64 bytes (for host write transfer), additional 64 bytes (for host read transfer).
4	FRDIE	0	R/W	—	FIFO Read Request Interrupt Enable Enables or disables the FRDI interrupt which is an IBFI3 interrupt source to the slave. 0: FIFO read request interrupt is disabled. 1: FIFO read request interrupt is enabled.
3	HRDIE	0	R/W	—	BT Host Read Interrupt Enable Enables or disables the HRDI interrupt which is an IBFI3 interrupt source to the slave. When using FIFO, the HRDIE bit must not be set to 1. 0: BT host read interrupt is disabled. 1: BT host read interrupt is enabled.
2	HWRIE	0	R/W	—	BT Host Write Interrupt Enable Enables or disables the HWRI interrupt which is an IBFI3 interrupt source to the slave. When using FIFO, the HWRIE bit must not be set to 1. 0: BT host write interrupt is disabled. 1: BT host write interrupt is enabled.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
1	HBTWIE	0	R/W	—	BTDTR Host Write Start Interrupt Enable Enables or disables the HBTWI interrupt which is an IBFI3 interrupt source to the slave. 0: BTDTR host write start interrupt is disabled. 1: BTDTR host write start interrupt is enabled.
0	HBTRIE	0	R/W	—	BTDTR Host Read End Interrupt Enable Enables or disables the HBTRI interrupt which is an IBFI3 interrupt source to the slave. 0: BTDTR host read end interrupt is disabled. 1: BTDTR host read end interrupt is enabled.

Note: X Don't care.

16.3.24 BT Control Status Register 1 (BTCSR1)

BTCSR1 is one of the registers used to implement the BT mode. The BTCSR1 register contains the bits used to enable or disable interrupts to the slave (this LSI). The IBFI3 interrupt is enabled by setting the IBFIE3 bit in HICR2 to 1.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	RSTRENBL	0	R/W	—	Slave Reset Read Enable The host reads 0 from the BMC_HWRST bit in BTIMSR. When this bit is set to 1, the host can read 1 from the BMC_HWRST bit. 0: Host always reads 0 from BMC_HWRST 1: Host can reads 0 from BMC_HWRST
6	HRSTIE	0	R/W	—	BT Reset Interrupt Enable Enables or disables the HRSTI interrupt which is an IBFI3 interrupt source to the slave. 0: BT reset interrupt is disabled. 1: BT reset interrupt is enabled.

Bit	Bit Name	Initial Value	R/W		Description
			Slave	Host	
5	IRQCRIE	0	R/W	—	B2H_IRQ Clear Interrupt Enable Enables or disables the IRQCRI interrupt which is an IBFI3 interrupt source to the slave. 0: B2H_IRQ clear interrupt is disabled. 1: B2H_IRQ clear interrupt is enabled.
4	BEVTIE	0	R/W	—	BEVT_ATN Clear Interrupt Enable Enables or disables the BEVTI interrupt which is an IBFI3 interrupt source to the slave. 0: BEVT_ATN clear interrupt is disabled. 1: BEVT_ATN clear interrupt is enabled.
3	B2HIE	0	R/W	—	Read End Interrupt Enable Enables or disables the B2HI interrupt which is an IBFI3 interrupt source to the slave. 0: Read end interrupt is disabled. 1: Read end interrupt is enabled.
2	H2BIE	0	R/W	—	Write End Interrupt Enable Enables or disables the H2BI interrupt which is an IBFI3 interrupt source to the slave. 0: Write end interrupt is disabled. 1: Write end interrupt is enabled.
1	CRRPIE	0	R/W	—	Read Pointer Clear Interrupt Enable Enables or disables the CRRPI interrupt which is an IBFI3 interrupt source to the slave. 0: Read pointer clear interrupt is disabled. 1: Read pointer clear interrupt is enabled.
0	CRWPIE	0	R/W	—	Write Pointer Clear Interrupt Enable Enables or disables the CRWPI interrupt which is an IBFI3 interrupt source to the slave. 0: Write pointer clear interrupt is disabled. 1: Write pointer clear interrupt is enabled.

16.3.25 BT Control Register (BTCR)

BTCR is one of the registers used to implement BT mode. The BTCR register contains bits used in transfer cycle handshaking, and those indicating the completion of data transfer to the buffer.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	B_BUSY	1	R/W	R	BT Write Transfer Busy Flag Read-only bit from the host. Indicates that the BTDTR buffer is being used for BT write transfer (write transfer is in progress.) 0: Indicates waiting for BT write transfer 1: Indicates that the BTDTR buffer is being used
6	H_BUSY	0	R	(W)[*3]	BT Read Transfer Busy Flag This is a set/clear bit from the host. Indicates that the BTDTR buffer is being used for BT read transfer (read transfer is in progress.) 0: Indicates waiting for BT read transfer [Clearing condition] When the host writes a 1 while H_BUSY is set to 1. 1: Indicates that the BTDTR buffer is being used [Setting condition] When the host writes a 1 while H_BUSY is set to 0.
5	OEM0	0	R/W	R/(W)[*4]	User Defined Bit This bit is defined by the user, and validated only when set to 1 by a 0 written from the host. 0: [Clearing condition] When the slave writes a 0 after a 1 has been read from OEM0. 1: [Setting condition] When the slave writes a 1, after a 0 has been read from OEM0, or when the host writes a 0.

			R/W		
Bit	Bit Name	Initial Value	Slave	Host	Description
4	BEVT_ATN	0	R/(W)*[1]	R/(W)*[5]	Event Interrupt Sets when the slave detects an event to the host. Setting the B2H_IRQ_EN bit in the BTIMSR register enables the BEVT_ATN bit to be used as an interrupt source to the host. 0: No event interrupt request is available [Clearing condition] When the host writes a 1 to the bit. 1: An event interrupt request is available [Setting condition] When the slave writes a 1 after a 0 has been read from BEVT_ATN.
3	B2H_ATN	0	R/(W)*[1]	R/(W)*[5]	Slave Buffer Write End Indication Flag This status flag indicates that the slave has finished writing all data to the BTDTR buffer. Setting the B2H_IRQ_EN bit in the BTIMSR register enables the B2H_ATN bit to be used as an interrupt source to the host. 0: Host has completed reading the BTDTR buffer [Clearing condition] When the host writes a 1 1: Slave has completed writing to the BTDTR buffer. [Setting condition] When the slave writes a 1 after a 0 has been read from B2N_ATN.
2	H2B_ATN	0	R/(W)*[2]	R/(W)*[1]	Host Buffer Write End Indication Flag This status flag indicates that the host has finished writing all data to the BTDTR buffer. 0: Slave has completed reading the BTDTR buffer [Clearing condition] When the slave writes a 0 after a 1 has been read from H2B_ATN. 1: Host has completed writing to the BTDTR buffer [Setting condition] When the host writes a 1

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
1	CLR_RD_PTR	0	R/(W)*[2]	(W)*[1]	Read Pointer Clear This bit is used by the host to clear the read pointer during read transfer. A host read operation always yields 0 on readout. 0: Read pointer clear wait [Clearing condition] When the slave writes a 0 after a 1 has been read from CLR_RD_PTR. 1: Read pointer clear [Setting condition] When the host writes a 1.
0	CLR_WR_PTR	0	R/(W)*[2]	(W)*[1]	Write Pointer Clear This bit is used by the host to clear the write pointer during write transfer. A host read operation always yields 0 on readout. 0: Write pointer clear wait [Clearing condition] When the slave writes a 0 after a 1 has been read from CLR_WR_PTR. 1: Write pointer clear [Setting condition] When the host writes a 1.

Notes: 1. Only 1 can be written to set this flag.

2. Only 0 can be written to clear this flag.

3. Only 1 can be written to toggle this flag.

4. Only 0 can be written to set this flag.

5. Only 1 can be written to clear this flag.

16.3.26 BT Data Buffer (BTDTR)

BTDTR is used to implement the BT mode. BTDTR consists of two FIFOs: the host write transfer FIFO and the host read transfer FIFO. Their capacities are 64 bytes each. When using BTDTR, enable FIFO by means of the bits FSEL0 and FSEL1.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 to 0	bit7 to bit0	Undefined	R/W	R/W	The data written by the host is stored in FIFO (64 bytes) for host write transfer and read out by the slave in order of host writing. The data written by the slave is stored in FIFO (64 bytes) for host read transfer and read out by the host in order of slave writing.

16.3.27 BT Interrupt Mask Register (BTIMSR)

BTIMSR is one of the registers used to implement BT mode. The BTIMSR register contains the bits used to control the interrupts to the host.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	BMC_ HWRST	0	R/(W)*[2]	R/(W)*[1]	Slave Reset Performs a reset from the host to the slave. The host can only write a 1. Writing a 0 to this bit is invalid. The host will always return a 0 on read out. Setting the RSTRENBL bit enables a 1 to be read from the host. 0: The reset is cancelled [Clearing condition] When the slave writes a 0, after a 1 has been read from BMC_HWRST. 1: The reset is in progress. [Setting condition] When the host writes a 1.
6	—	0	R/W	R/W	Reserved
5	—	0	R/W	R/W	

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
4	OEM3	0	R/W	R/(W)*[4]	User Defined Bit These bits are defined by the user and are valid only when set to 1 by a 0 written from the host. 0: [Clearing condition] When the slave writes a 0, after a 1 has been read from OEM. 1: [Setting condition] When the slave writes a 1, after a 0 has been read from OEM, or when the host writes a 0.
3	OEM2	0	R/W	R/(W)*[4]	
2	OEM1	0	R/W	R/(W)*[4]	
1	B2H_IRQ	0	R/(W)*[1]	R/(W)*[3]	BMC to HOST Interrupt Informs the host that an interrupt has been requested when the BEVT_ATN or B2H_ATN bit has been set. The SERIRQ is not issued. To generate the SERIRQ, it should be issued by the program. 0: B2H_IRQ interrupt is not requested [Clearing condition] When the host writes a 1. 1: B2H_IRQ interrupt is requested [Setting condition] When the slave writes a 1, after a 0 has been read from B2H_IRQ
0	B2H_IRQ_EN	0	R	R/W	BMC to HOST Interrupt Enable Enables or disables the B2H_IRQ interrupt which is an interrupt source from the slave to the host. 0: B2H_IRQ interrupt is disabled [Clearing condition] When a 0 is written by the host. 1: B2H_IRQ interrupt is enabled [Setting condition] When a 1 is written by the host.

Notes: 1. Only 1 can be written to set this flag.

2. Only 0 can be written to clear this flag.

3. Only 1 can be written to clear this flag.

4. Only 0 can be written to set this flag.

16.3.28 BT FIFO Valid Size Register 0 (BTFVSR0)

BTFVSR0 is one of the registers used to implement BT mode. BTFVSR0 indicates a valid data size in the FIFO for host write transfer.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 to 0	N7 to N0	All 0	R	—	These bits indicate the number of valid bytes in the FIFO (the number of bytes which the slave can read) for host write transfer. When data is written from the host, the value in BTFVSR0 is incremented by the number of bytes that have been written to. Further, when data is read from the slave, the value is decremented by only the number of bytes that have been read.

16.3.29 BT FIFO Valid Size Register 1 (BTFVSR1)

BTFVSR1 is one of the registers used to implement BT mode. BTFVSR1 indicates a valid data size in the FIFO for host read transfer.

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7 to 0	N7 to N0	All 0	R	—	These bits indicate the number of valid bytes in the FIFO (the number of bytes which the host can read) for host read transfer. When data is written from the slave, the value in BTFVSR1 is incremented by the number of bytes that have been written to. Further, when data is read from the host, the value is decremented by only the number of bytes that have been read.

16.4 Operation

16.4.1 LPC interface Activation

The LPC interface is activated by setting any one of bits LPC3E to LPC1E in HICR0 to 1. When the LPC interface is activated, the related I/O port pins (PE7 to PE0) function as dedicated LPC interface input/output pins.

Use the following procedure to activate the LPC 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 channels 1 and 2, set LADR1 and LADR2 to determine the I/O address.
3. When using channel 3, set LADR3 to determine the I/O address and whether bidirectional data registers are to be used.
4. Set the enable bit (LPC3E to LPC1E) for the channel to be used. Also set SCIFE if the SCIF is to be used.
5. Set the selection bits for other functions (SDWNE, IEDIR).
6. As a precaution, clear the interrupt flags (LRST, SDWN, ABRT, and OBF). Read IDR or TWR15 to clear IBF.
7. Set receive complete interrupt enable bits (IBFIE3 to IBFIE1, and ERRIE) as necessary.

16.4.2 LPC I/O Cycles

There are 12 types of LPC transfer cycle: LPC memory read, LPC 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, bus master I/O write, FW memory read, and FW memory write. Of these, the LPC of this LSI supports I/O read and I/O write.

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 the LPC interface of this LSI always returns B'0000 (except for the BT interface).

If the received address matches the host address for an LPC register, the LPC 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), registers and flags are not changed.

The timing of the $\overline{\text{LFRAME}}$, LCLK, and LAD signals is shown in figures 16.2 and 19.3.

Table 16.5 LPC I/O Cycle

	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

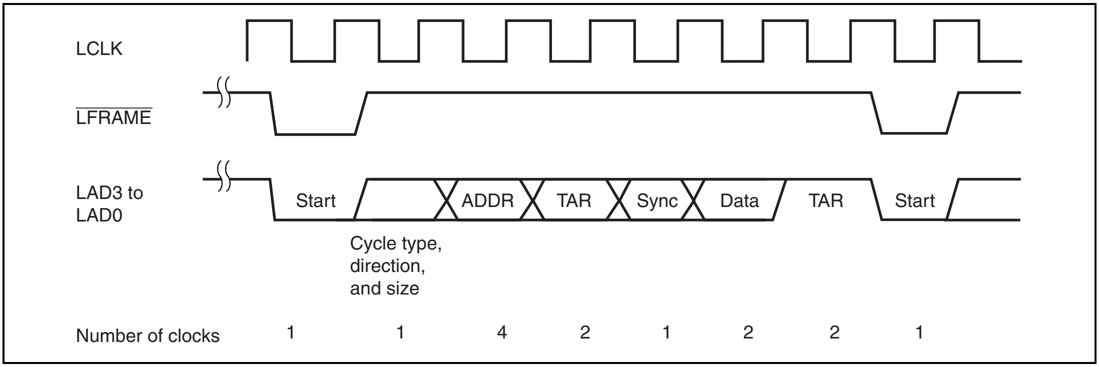


Figure 16.2 Typical $\overline{\text{LFRAME}}$ Timing

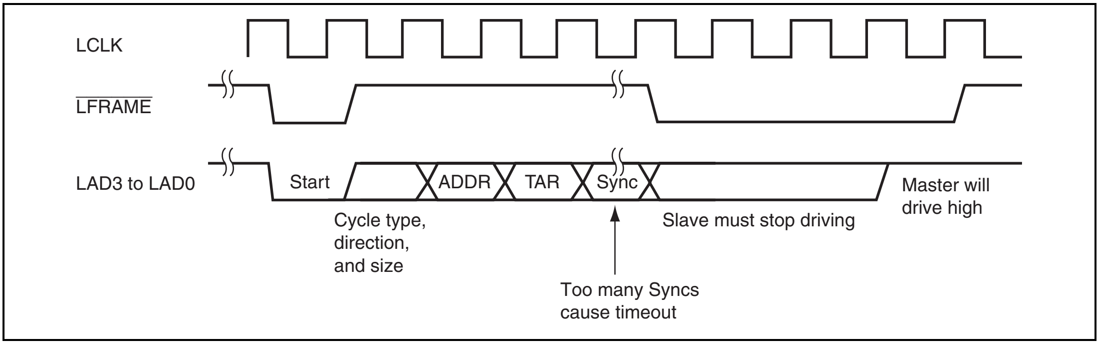


Figure 16.3 Abort Mechanism

16.4.3 SMIC Mode Transfer Flow

Figure 16.4 shows the write transfer flow and figure 16.5 shows the read transfer flow in SMIC mode.

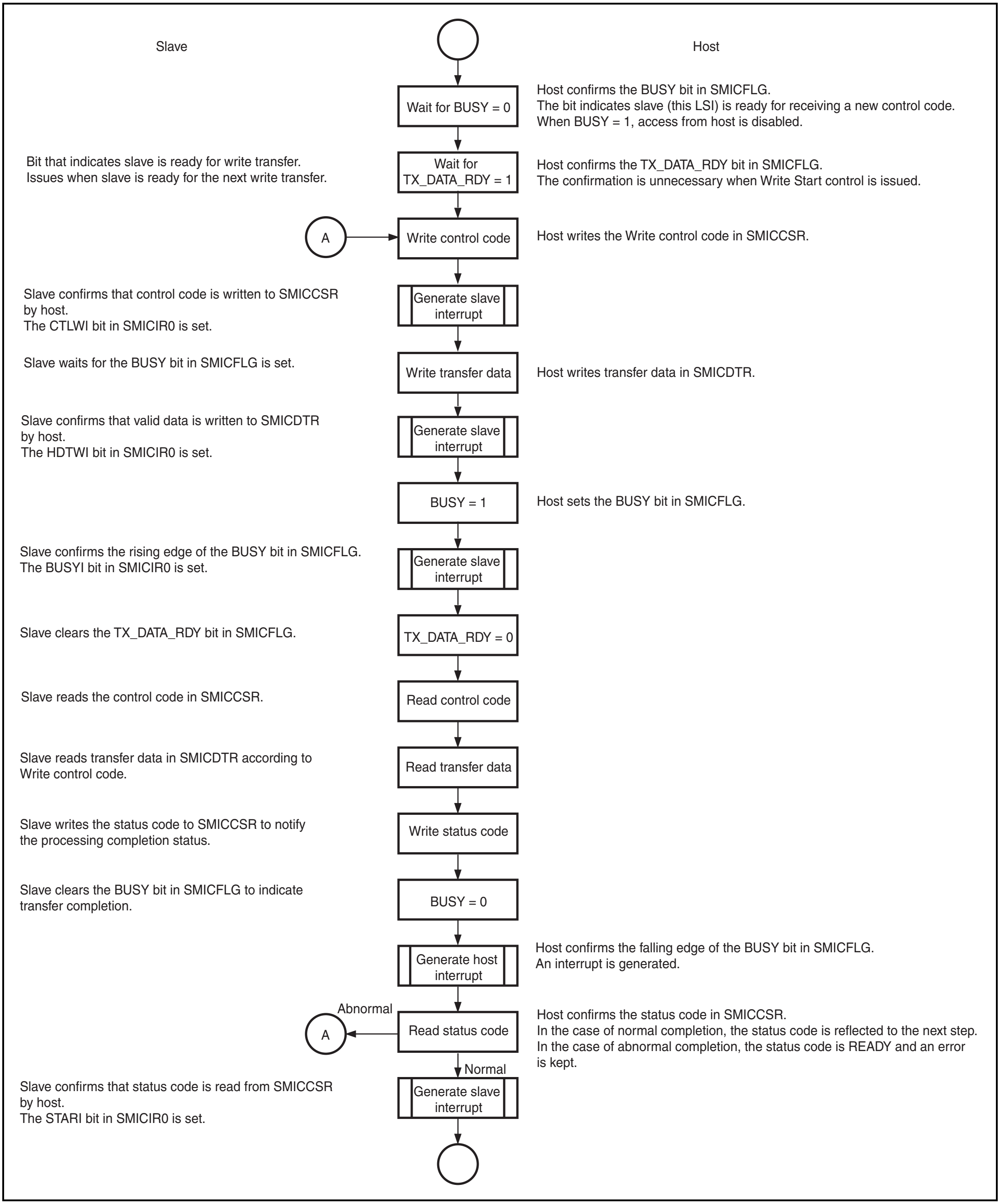

Slave
Host
Wait for BUSY = 0
Host confirms the BUSY bit in SMICFLG.
The bit indicates slave (this LSI) is ready for receiving a new control code.
When BUSY = 1, access from host is disabled.
Bit that indicates slave is ready for write transfer.
Issues when slave is ready for the next write transfer.
Wait for TX_DATA_RDY = 1
Host confirms the TX_DATA_RDY bit in SMICFLG.
The confirmation is unnecessary when Write Start control is issued.
A
Write control code
Host writes the Write control code in SMICCSR.
Slave confirms that control code is written to SMICCSR by host.
The CTLWI bit in SMICIR0 is set.
Generate slave interrupt
Slave waits for the BUSY bit in SMICFLG is set.
Write transfer data
Host writes transfer data in SMICDTR.
Slave confirms that valid data is written to SMICDTR by host.
The HDTWI bit in SMICIR0 is set.
Generate slave interrupt
BUSY = 1
Host sets the BUSY bit in SMICFLG.
Slave confirms the rising edge of the BUSY bit in SMICFLG.
The BUSYI bit in SMICIR0 is set.
Generate slave interrupt
Slave clears the TX_DATA_RDY bit in SMICFLG.
TX_DATA_RDY = 0
Slave reads the control code in SMICCSR.
Read control code
Slave reads transfer data in SMICDTR according to Write control code.
Read transfer data
Slave writes the status code to SMICCSR to notify the processing completion status.
Write status code
Slave clears the BUSY bit in SMICFLG to indicate transfer completion.
BUSY = 0
Generate host interrupt
Host confirms the falling edge of the BUSY bit in SMICFLG.
An interrupt is generated.
Abnormal
A
Read status code
Host confirms the status code in SMICCSR.
In the case of normal completion, the status code is reflected to the next step.
In the case of abnormal completion, the status code is READY and an error is kept.
Normal
Generate slave interrupt
Slave confirms that status code is read from SMICCSR by host.
The STARI bit in SMICIR0 is set.

Figure 16.4 SMIC Write Transfer Flow

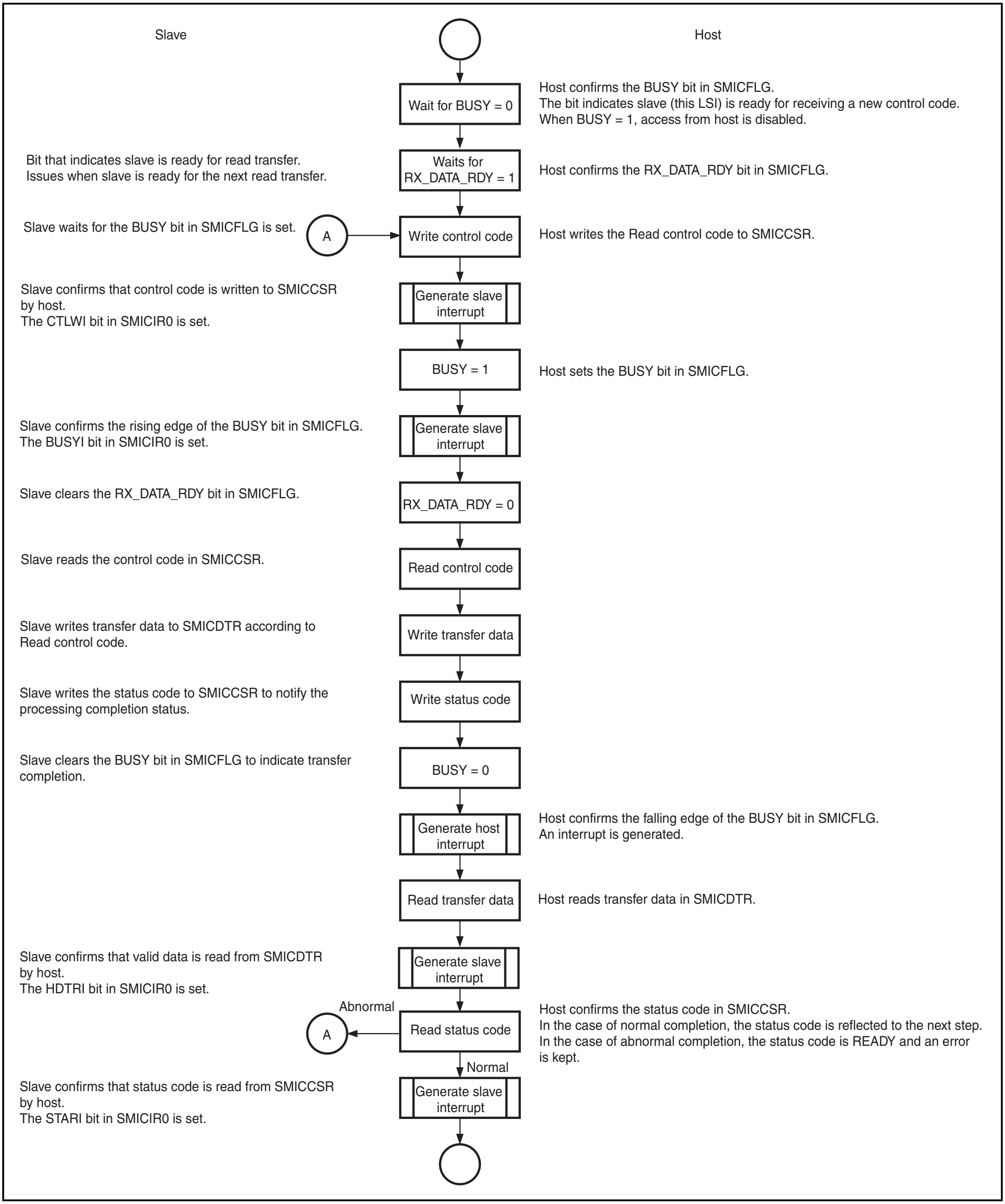

Slave
Host
Wait for BUSY = 0
Host confirms the BUSY bit in SMICFLG.
The bit indicates slave (this LSI) is ready for receiving a new control code.
When BUSY = 1, access from host is disabled.
Bit that indicates slave is ready for read transfer.
Issues when slave is ready for the next read transfer.
Waits for RX_DATA_RDY = 1
Host confirms the RX_DATA_RDY bit in SMICFLG.
Slave waits for the BUSY bit in SMICFLG is set.
A
Write control code
Host writes the Read control code to SMICCSR.
Slave confirms that control code is written to SMICCSR by host.
The CTLWI bit in SMICIR0 is set.
Generate slave interrupt
BUSY = 1
Host sets the BUSY bit in SMICFLG.
Slave confirms the rising edge of the BUSY bit in SMICFLG.
The BUSYI bit in SMICIR0 is set.
Generate slave interrupt
Slave clears the RX_DATA_RDY bit in SMICFLG.
RX_DATA_RDY = 0
Slave reads the control code in SMICCSR.
Read control code
Slave writes transfer data to SMICDTR according to Read control code.
Write transfer data
Slave writes the status code to SMICCSR to notify the processing completion status.
Write status code
Slave clears the BUSY bit in SMICFLG to indicate transfer completion.
BUSY = 0
Generate host interrupt
Host confirms the falling edge of the BUSY bit in SMICFLG.
An interrupt is generated.
Read transfer data
Host reads transfer data in SMICDTR.
Slave confirms that valid data is read from SMICDTR by host.
The HDTRI bit in SMICIR0 is set.
Generate slave interrupt
Abnormal
A
Read status code
Host confirms the status code in SMICCSR.
In the case of normal completion, the status code is reflected to the next step.
In the case of abnormal completion, the status code is READY and an error is kept.
Normal
Slave confirms that status code is read from SMICCSR by host.
The STARI bit in SMICIR0 is set.
Generate slave interrupt

Figure 16.5 SMIC Read Transfer Flow

16.4.4 BT Mode Transfer Flow

Figure 16.6 shows the write transfer flow and figure 16.7 shows the read transfer flow in BT mode.

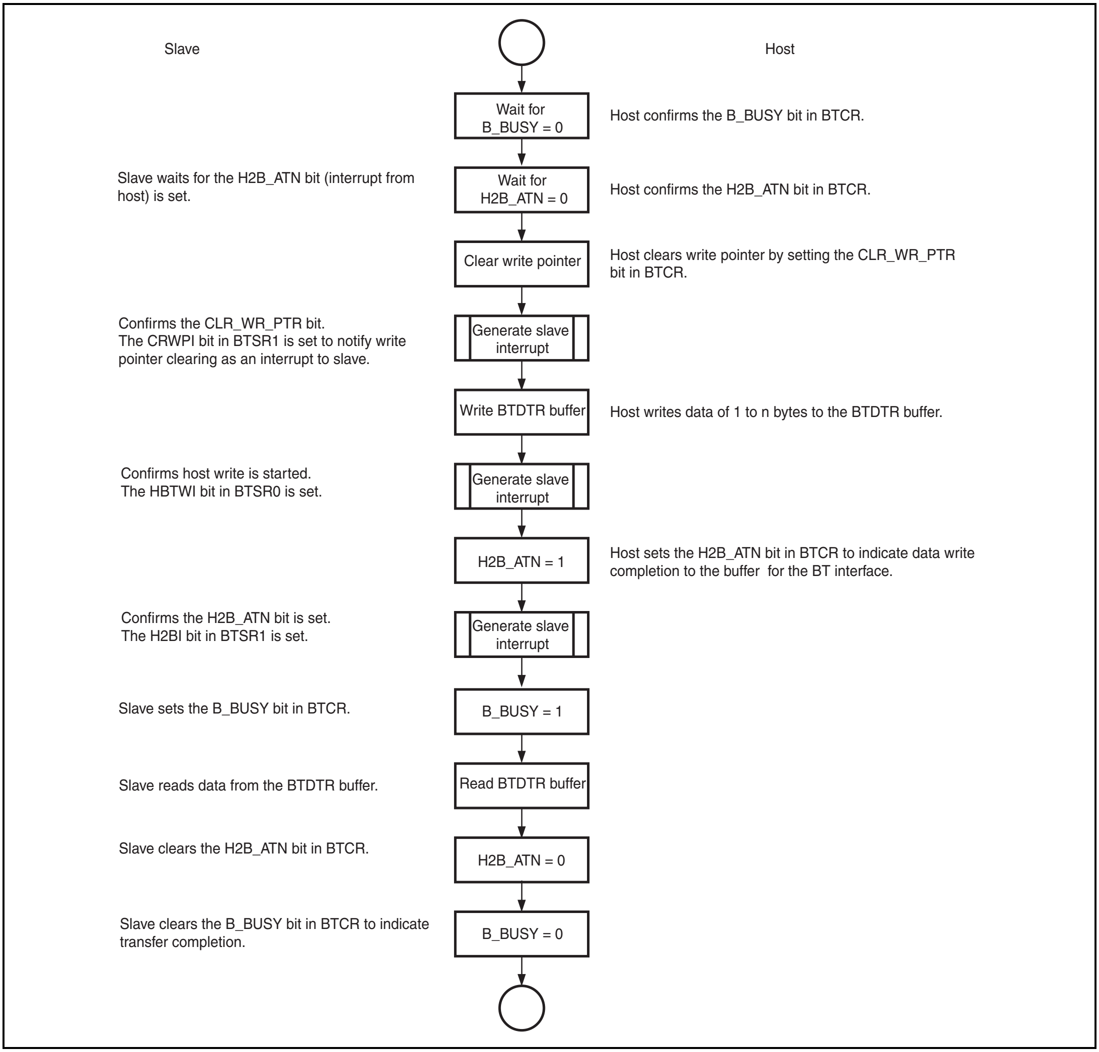

Slave
Host
Wait for B_BUSY = 0
Host confirms the B_BUSY bit in BTCR.
Slave waits for the H2B_ATN bit (interrupt from host) is set.
Wait for H2B_ATN = 0
Host confirms the H2B_ATN bit in BTCR.
Clear write pointer
Host clears write pointer by setting the CLR_WR_PTR bit in BTCR.
Confirms the CLR_WR_PTR bit.
The CRWPI bit in BTSR1 is set to notify write pointer clearing as an interrupt to slave.
Generate slave interrupt
Write BTDTR buffer
Host writes data of 1 to n bytes to the BTDTR buffer.
Confirms host write is started.
The HBTWI bit in BTSR0 is set.
Generate slave interrupt
H2B_ATN = 1
Host sets the H2B_ATN bit in BTCR to indicate data write completion to the buffer for the BT interface.
Confirms the H2B_ATN bit is set.
The H2BI bit in BTSR1 is set.
Generate slave interrupt
Slave sets the B_BUSY bit in BTCR.
B_BUSY = 1
Slave reads data from the BTDTR buffer.
Read BTDTR buffer
Slave clears the H2B_ATN bit in BTCR.
H2B_ATN = 0
Slave clears the B_BUSY bit in BTCR to indicate transfer completion.
B_BUSY = 0

Figure 16.6 BT Write Transfer Flow

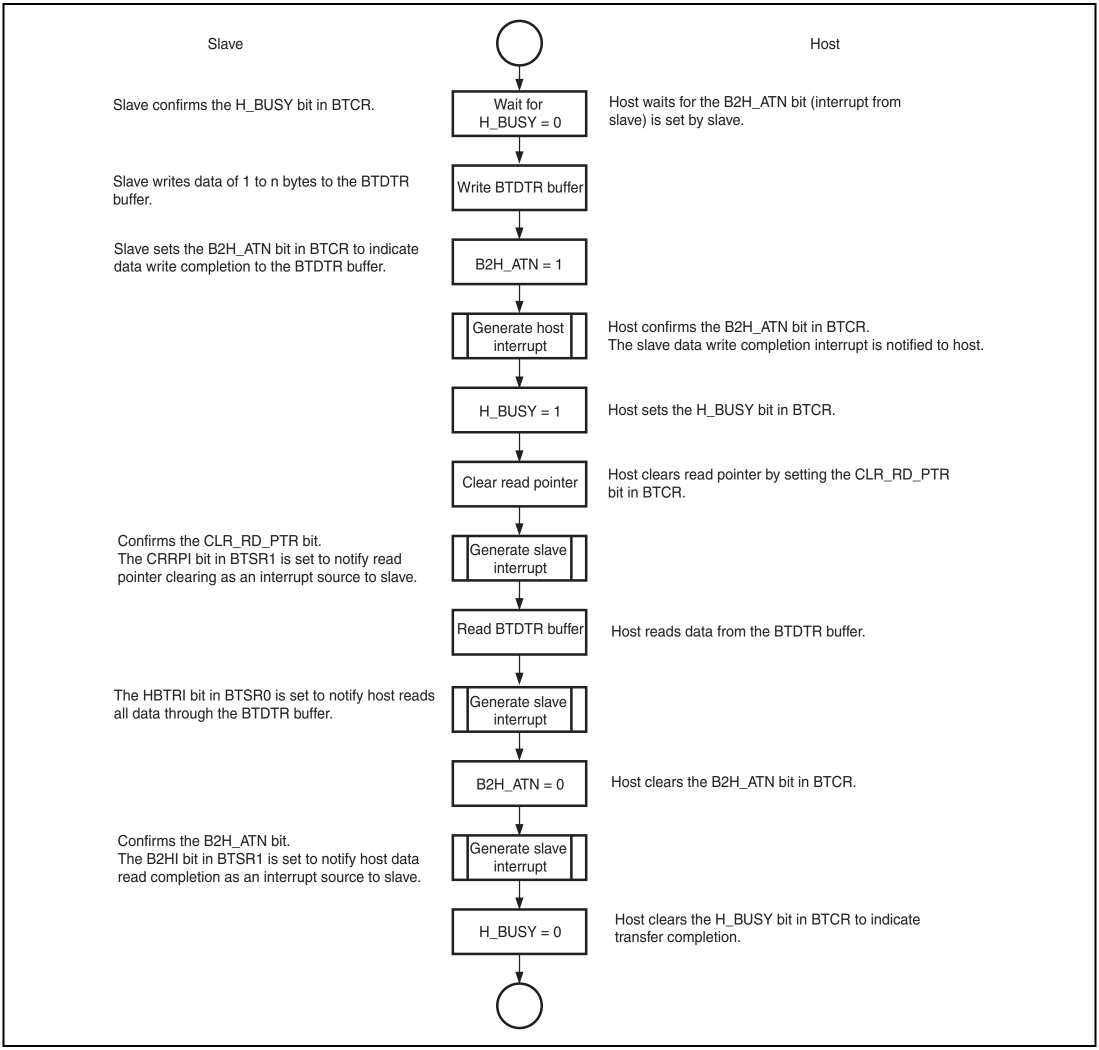

Slave
Host
Slave confirms the H_BUSY bit in BTCR.
Wait for H_BUSY = 0
Host waits for the B2H_ATN bit (interrupt from slave) is set by slave.
Slave writes data of 1 to n bytes to the BTDTR buffer.
Write BTDTR buffer
Slave sets the B2H_ATN bit in BTCR to indicate data write completion to the BTDTR buffer.
B2H_ATN = 1
Generate host interrupt
Host confirms the B2H_ATN bit in BTCR.
The slave data write completion interrupt is notified to host.
H_BUSY = 1
Host sets the H_BUSY bit in BTCR.
Clear read pointer
Host clears read pointer by setting the CLR_RD_PTR bit in BTCR.
Confirms the CLR_RD_PTR bit.
The CRRPI bit in BTSR1 is set to notify read pointer clearing as an interrupt source to slave.
Generate slave interrupt
Read BTDTR buffer
Host reads data from the BTDTR buffer.
The HBTRI bit in BTSR0 is set to notify host reads all data through the BTDTR buffer.
Generate slave interrupt
B2H_ATN = 0
Host clears the B2H_ATN bit in BTCR.
Confirms the B2H_ATN bit.
The B2HI bit in BTSR1 is set to notify host data read completion as an interrupt source to slave.
Generate slave interrupt
H_BUSY = 0
Host clears the H_BUSY bit in BTCR to indicate transfer completion.

Figure 16.7 BT Read Transfer Flow

16.4.5 LPC Interface Shutdown Function

The LPC software shutdown state is controlled by the SDWNB bit. The LPC 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 in sleep mode or software standby mode is effective in reducing current dissipation in the shutdown state.

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 $\overline{\text{RES}}$ 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 software shutdown

The scope of the initialization in each mode is shown in table 16.9.

Table 16.6 Scope of Initialization in Each LPC 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
LPC interface flags (IBF1, IBF2, IBF3A, IBF3B, MWMF, C/$\overline{\text{D}}$1, C/$\overline{\text{D}}$2, C/$\overline{\text{D}}$3, OBF1, OBF2, OBF3A, OBF3B, SWMF, DBU, SMICFLG, SMICIR0, BTSR0, BTSR1, BTIMSR, BTFVSR0, BTFVSR1)	Initialized	Initialized	Retained
Host interrupt enable bits (IRQ1E1, IRQ12E1, SMIE2, IRQ6E2, IRQ9E2 to IRQ11E2, SMIE3B, SMIE3A, IRQ6E3, IRQ9E3 to IRQ11E3, SELREQ, IEDIR2 to IEDIR3), Q/$\overline{\text{C}}$ flag	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
LPC interface operation control bits (LPC3E to LPC1E, LADR1 to LADR3, IBFIE1 to IBFIE3, SELSTR3, SELIRQ1, SELSMI, SELIRQ3 to SELIRQ15, HICR4, HICR5, HISEL, BTCSR0, BTCSR1)	Initialized	Retained	Retained
$\overline{\text{LRESET}}$ signal	Input (port function	Input	Input
LAD3 to LAD0, $\overline{\text{LFRAME}}$, LCLK, SERIRQ, $\overline{\text{CLKRUN}}$ signals		Input	Hi-Z

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 software shutdown (SS)

Figure 16.8 shows the timing of the $\overline{\text{LRESET}}$ signals.

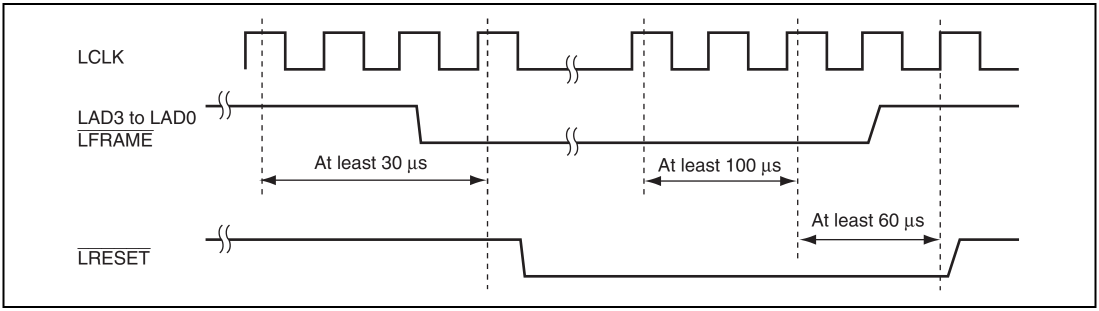


Figure 16.8   Power-Down State Termination Timing

16.4.6 LPC Interface Serialized Interrupt Operation (SERIRQ)

A host interrupt request can be issued from the LPC 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 16.9.

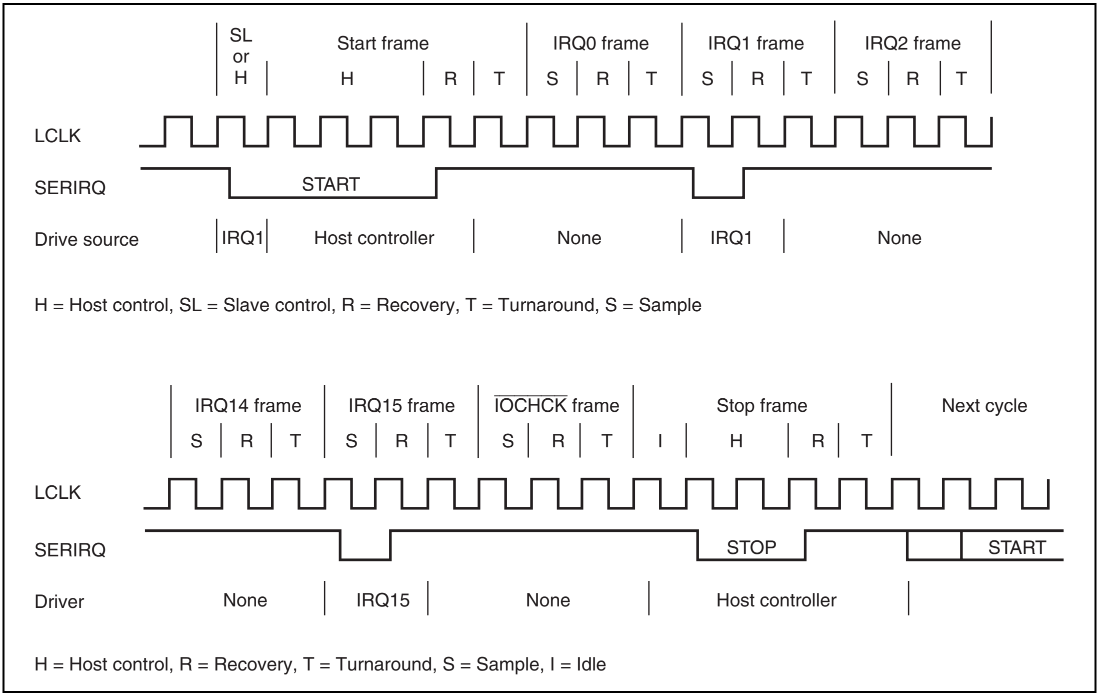


Figure 16.9 SERIRQ Timing

The serialized interrupt transfer cycle frame configuration 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 that was driving the preceding state.

Table 16.7 Serialized Interrupt Transfer Cycle Frame Configuration

	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 the first state, then next 3 states 0-driven by host
1	IRQ0	Slave	3	Drive impossible
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	Drive possible by IRQ3E
5	IRQ4	Slave	3	Drive possible by IRQ4E
6	IRQ5	Slave	3	Drive possible by IRQ5E
7	IRQ6	Slave	3	Drive possible in LPC channels 2 and 3
8	IRQ7	Slave	3	Drive possible in SCIF or by IRQ7E
9	IRQ8	Slave	3	Drive possible by IRQ8E
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	Drive possible by IRQ13E
15	IRQ14	Slave	3	Drive possible by IRQ14E
16	IRQ15	Slave	3	Drive possible by IRQ15E
17	IOCHCK	Slave	3	Drive impossible
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 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.

16.5 Interrupt Sources

16.5.1 IBFI1, IBFI2, IBFI3, and ERRI

The host has four interrupt requests for the slave (this LSI): IBF1, IBF2, IBF3, and ERRI. IBFI1, IBFI2, and IBFI3 are IDR receive complete interrupts for IDR1, IDR2, and IDR3 and TWR, respectively. IBFI3 is also used for SMIC mode and BT mode interrupt requests. The ERRI interrupt indicates the occurrence of a special state such as an LPC reset, LPC shutdown, or transfer cycle abort. The LMCI and LMCUI interrupts are command receive complete interrupts.

Table 16.8 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

16.5.2 SMI, HIRQ1, HIRQ3, HIRQ4, HIRQ5, HIRQ6, HIRQ7, HIRQ8, HIRQ9, HIRQ10, HIRQ11, HIRQ12, HIRQ13, HIRQ14, and HIRQ15

The LPC interface can request 15 kinds of host interrupt by means of SERIRQ. HIRQ1 and HIRQ12 are used on LPC channel 1, while SMI, HIRQ6, HIRQ9, HIRQ10, and HIRQ11 can be requested from LPC channels 2 and 3. For the SCIF, any one of 15 types of interrupts can be selected. In addition, by the setting of SCIFCR4, the SCIF can request eight types of host interrupts: HIRQ3, HIRQ4, HIRQ5, HIRQ7, HIRQ8, HIRQ13, HIRQ14, and HIRQ15.

There are two ways of clearing a host interrupt request when the LPC channels are used.

When the IEDIR bit in SIRQCR0is cleared to 0, 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 SIRQCR, a host interrupt is only requested by the host interrupt enable bits. The host interrupt enable bit is not cleared when OBF is cleared. Therefore, SMIE2, SMIE3A, SMIE3B, SMIE4 and IRQ6En, IRQ9En, IRQ10En, IRQ11En lose their respective functional differences (n = 2, 3). In order to clear a host interrupt request, it is necessary to clear the host interrupt enable bit. As for HIRQ3 to HIRQ5, HIRQ7, HIRQ8, and HIRQ13 to HIRQ15, setting the enable bit in SIRQCR4 to 1 requests the corresponding host interrupt, and clearing the enable bit to 0 clears the corresponding host interrupt request.

Table 16.9 summarizes the methods of setting and clearing these bits when the LPC channels are used. Figure 16.10 shows the processing flowchart.

Table 16.9 HIRQ Setting and Clearing Conditions when LPC Channels are Used

Host Interrupt	Setting Condition	Clearing Condition
HIRQ1	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	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 (IEDIR2 = 0 or IEDIR3 = 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 (IEDIR2 = 1 or IEDIR3 = 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) (IEDIR2 = 0 or IEDIR3 = 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 • writes 0 to bit IRQiE3, or host reads ODR3
HIRQi (i = 6, 9, 10, 11) (IEDIR2 = 1 or IEDIR3 = 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

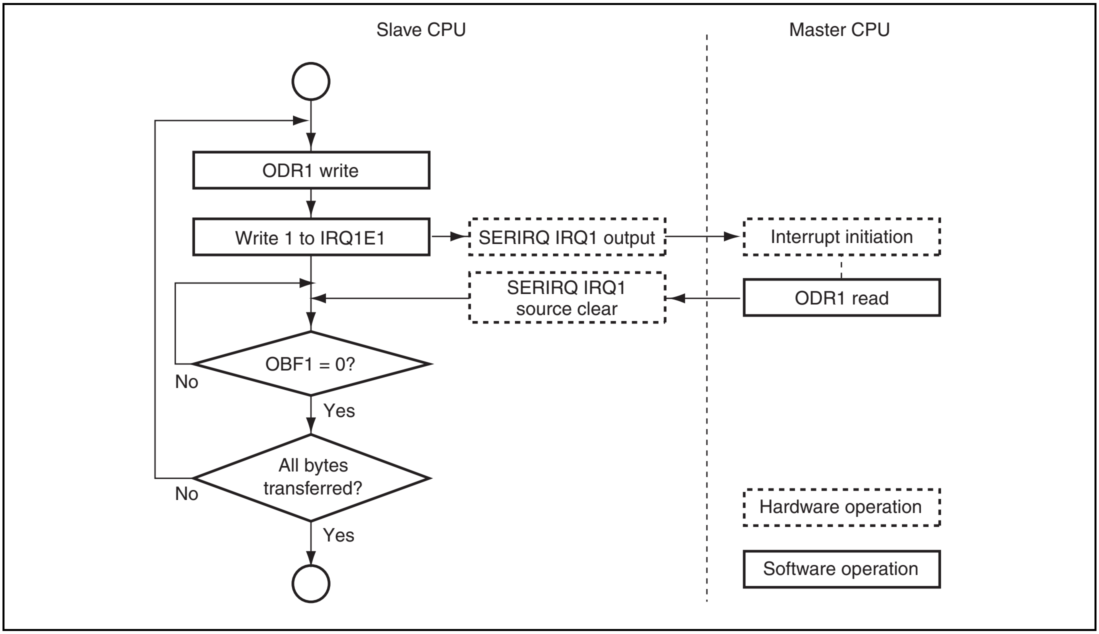


Figure 16.10 HIRQ Flowchart (Example of Channel 1)

16.6 Usage Note

16.6.1 Data Conflict

The LPC interface provides buffering of asynchronous data from the host and slave (this LSI), but an interface protocol that uses the flags in STR must be followed to avoid data conflict. For example, if the host and slave 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 16.10 shows host address examples for LADR3 and registers, IDR3, ODR3, STR3, TWR0MW, TWR0SW, and TWR1 to TWR15.

Table 16.10 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 17 A/D Converter

This LSI includes a successive-approximation-type 10-bit A/D converter that allows up to eight analog input channels to be selected.

A block diagram of the A/D converter is shown in figure 17.1.

17.1 Features

- 10-bit resolution
- Eight input channels
- Conversion time: 6.4 µs per channel (at 25-MHz operation)
- Two operating modes

  Single mode: Single-channel A/D conversion

  Scan mode: Continuous A/D conversion on 1 to 4 channels or continuous A/D conversion on 1 to 8 channels
- Eight data registers

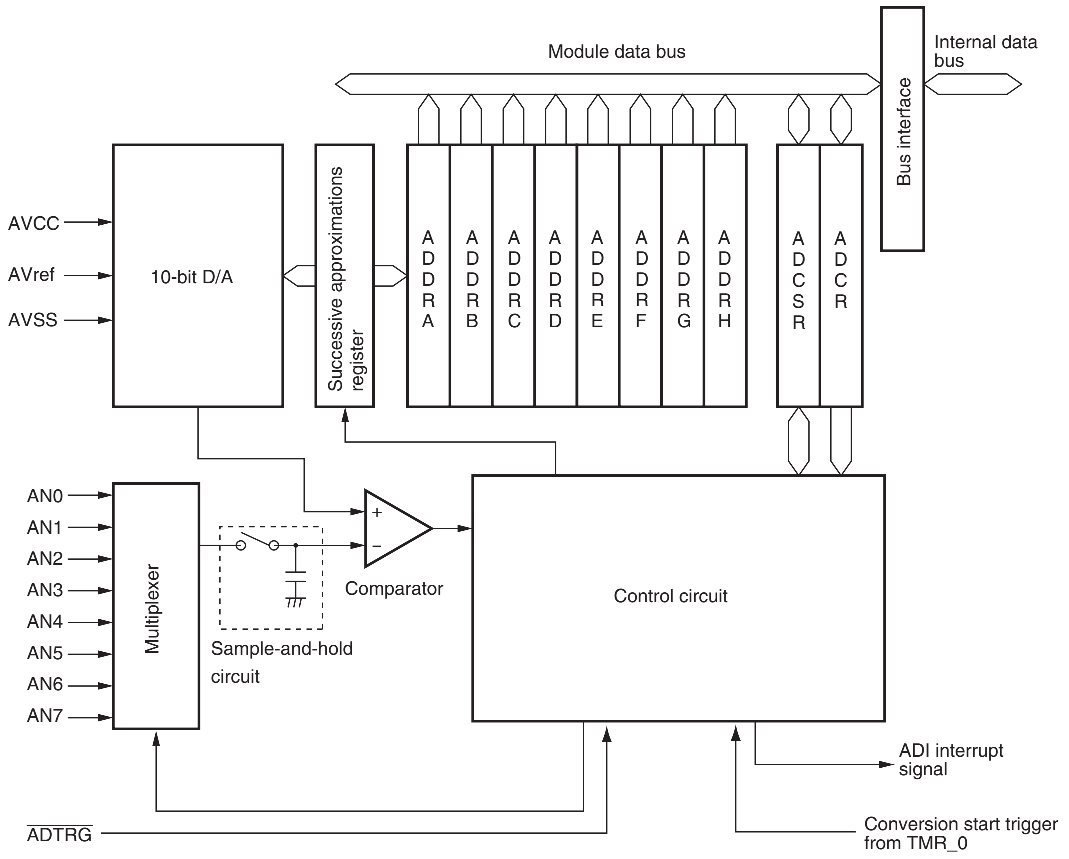
  Conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three ways of conversion start

  Conversion start trigger from TMR_0

  Software

  External trigger signal
- Interrupt request

  A/D conversion end interrupt (ADI) request can be generated
- Module stop mode can be set



[Legend]

ADCR: A/D control register

ADCSR: A/D control/status register

ADDRA: A/D data register A

ADDRB: A/D data register B

ADDRC: A/D data register C

ADDRD: A/D data register D

ADDRE: A/D data register E

ADDRF: A/D data register F

ADDRG: A/D data register G

ADDRH: A/D data register H

Figure 17.1 Block Diagram of the A/D Converter

17.2 Input/Output Pins

Table 17.1 summarizes the pins used by the A/D converter.

Table 17.1 Pin Configuration

Pin Name	Symbol	I/O	Function
Analog input pin 0	AN0	Input	Analog input pins
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin 3	AN3	Input	
Analog input pin 4	AN4	Input	
Analog input pin 5	AN5	Input	
Analog input pin 6	AN6	Input	
Analog input pin 7	AN7	Input	
External trigger input pin	$\overline{\text{ADTRG}}$	input	External trigger input for starting A/D conversion
Analog power supply pin	AVcc	Input	Analog block power supply
Analog ground pin	AVss	Input	Analog block ground
Reference power supply pin	AVref	Input	Reference voltage for A/D converter

17.3 Register Descriptions

The A/D converter has the following registers.

- A/D data register A (ADDRA)
- A/D data register B (ADDRB)
- A/D data register C (ADDRC)
- A/D data register D (ADDRD)
- A/D data register E (ADDRE)
- A/D data register F (ADDRF)
- A/D data register G (ADDRG)
- A/D data register H (ADDRH)
- A/D control/status register (ADCSR)
- A/D control register (ADCR)

17.3.1 A/D Data Registers A to H (ADDRA to ADDRH)

The ADDR are eight 16-bit read-only registers, ADDRA to ADDRH, which store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 17.2.

The converted 10-bit data is stored to bits 15 to 6. The lower 6-bit data is always read as 0.

The data bus between the CPU and the A/D converter is 16-bit width and can be read directly from the CPU. The ADDR must always be accessed in 16-bit unit. They cannot be accessed in 8-bit unit.

The results of A/D conversion are stored in each registers, when the ADF flag is set to 1.

Table 17.2 Analog Input Channels and Corresponding ADDR Registers

Analog Input Channel	A/D Data Register to Store A/D Conversion Results
AN0	ADDRA
AN1	ADDRB
AN2	ADDRC
AN3	ADDRD
AN4	ADDRE
AN5	ADDRF
AN6	ADDRG
AN7	ADDRH

17.3.2 A/D Control/Status Register (ADCSR)

The ADCSR controls the operation of the A/D conversion.

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W)*	A/D End Flag A status flag that indicates the end of A/D conversion. This flag indicates that the results of A/D conversion are stored in the A/D data registers. [Setting conditions] • When A/D conversion ends in single mode • When A/D conversion ends on all channels specified in scan mode [Clearing conditions] • When 0 is written after reading ADF = 1 • When DTC starts by an ADI interrupt and ADDR is read
6	ADIE	0	R/W	A/D Interrupt Enable Enables ADI interrupt by ADF when this bit is set to 1
5	ADST	0	R/W	A/D Start Clearing this bit to 0 stops A/D conversion and enters the idle state. Setting this bit to 1 starts A/D conversion. In single mode, this bit is cleared to 0 automatically when conversion on the specified channel ends. In scan mode, conversion continues sequentially on the specified channels until this bit is cleared to 0 by software, a reset, or a transition to the hardware standby mode.
4	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description		
3	—	0	R/W	Reserved The initial value should not be changed.		
2	CH2	All 0	R/W	Channel Select 2 to 0		
1	CH1			Select analog input channels together with the SCANE bit and the SCANS bit of ADCR.		
0	CH0					
				When SCANE = 0, and SCANS = x	When SCANE = 1 and SCANS = 0	When SCANE = 1 and SCANS = 1
				000: AN0	000: AN0	000: AN0
				001: AN1	001: AN0 and AN1	001: AN1 and AN1
				010: AN2	010: AN0 to AN2	010: AN0 to AN2
				011: AN3	011: AN0 to AN3	011: AN0 to AN3
				100: AN4	100: AN4	100: AN0 to AN4
				101: AN5	101: AN4 and AN5	101: AN0 to AN5
				110: AN6	110: AN4 to AN6	110: AN0 to AN6
				111: AN7	111: AN4 to AN7	111: AN0 to AN7

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

[Legend] x: Don't care

17.3.3 A/D Control Register (ADCR)

The ADCR sets the operation mode of A/D converter and the conversion time.

Bit	Bit Name	Initial Value	R/W	Description
7	TRGS1	0	R/W	Timer Trigger Select 1 and 0, Extended Trigger Select
6	TRGS0	0	R/W	Enable starting of A/D conversion by a trigger signal.
0	EXTRGS	0	R/W	These bits should be set while A/D conversion is stopped (ADSF = 0). 00 0: Disables starting by trigger signals. 10 0: Enables starting by a trigger from TMR_0. 10 1: Enables starting by the $\overline{\text{ADTRG}}$ pin input. Other than above: Setting prohibited
5	SCANE	0	R/W	Scan Mode
4	SCANS	0	R/W	Select the operation mode of A/D conversion 0x: Single mode 10: Scan mode (consecutive A/D conversion of channels 1 to 4) 11: Scan mode (consecutive A/D conversion of channels 1 to 8)
3	CKS1	0	R/W	Clock Select 1 and 0
2	CKS0	0	R/W	Set the A/D conversion time. Setting should be made while the conversion is stopped (ADST = 0). 00: Setting prohibited 01: Conversion time = 80 states (max) 10: Conversion time = 160 states (max) 11: Setting prohibited
1	ADSTCLR	0	R/W	A/D Start Clear Sets the automatic clearing of the ADST bit in scan mode. 0: Disables the automatic clearing of the ADST bit in scan mode. 1: Automatically clears the bit when A/D conversion of all of the selected channels are completed.

[Legend] x: Don't care

17.4 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes: single mode and scan mode. When changing the operating mode or analog input channel, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. The ADST bit can be set to 1 at the same time as the operating mode or analog input channel is changed.

17.4.1 Single Mode

In single mode, A/D conversion is performed only once on the specified single channel. Operations are as follows.

1. A/D conversion on the specified channel is started when the ADST bit in ADCSR is set to 1, by software or an external trigger input.
2. When A/D conversion is completed, the result is transferred to the A/D data register corresponding to the channel.
3. On completion of A/D conversion, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion. When conversion ends, the ADST bit is automatically cleared to 0, and the A/D converter enters the idle state. If the ADST bit is cleared during A/D conversion, the A/D converter stops conversion and enters the idle state.

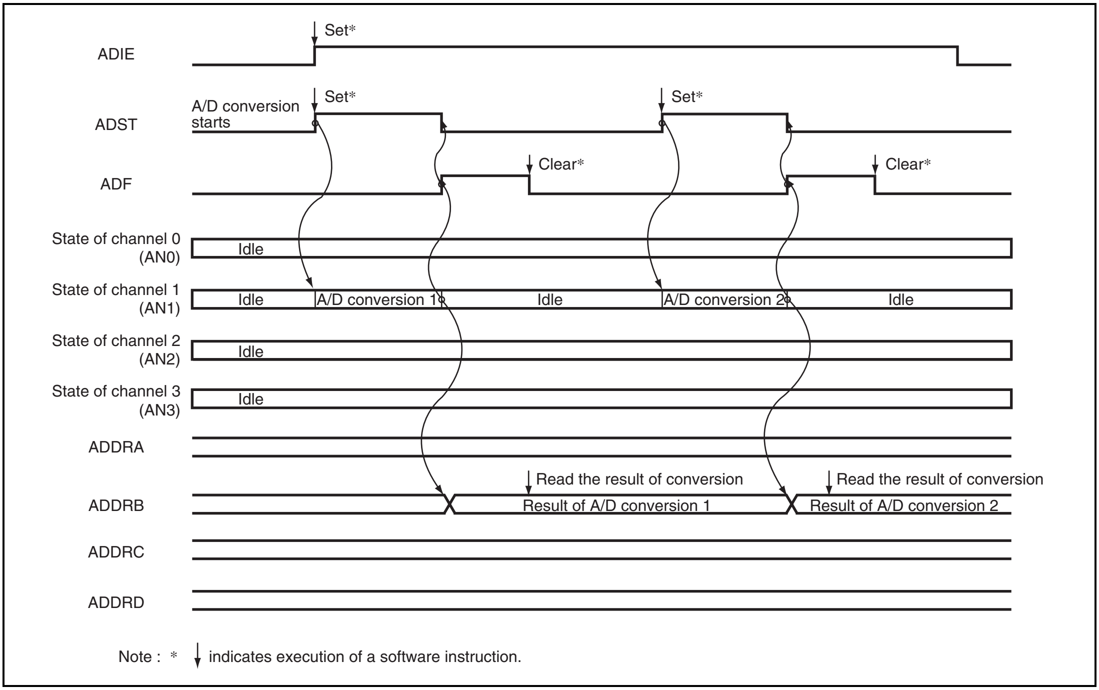


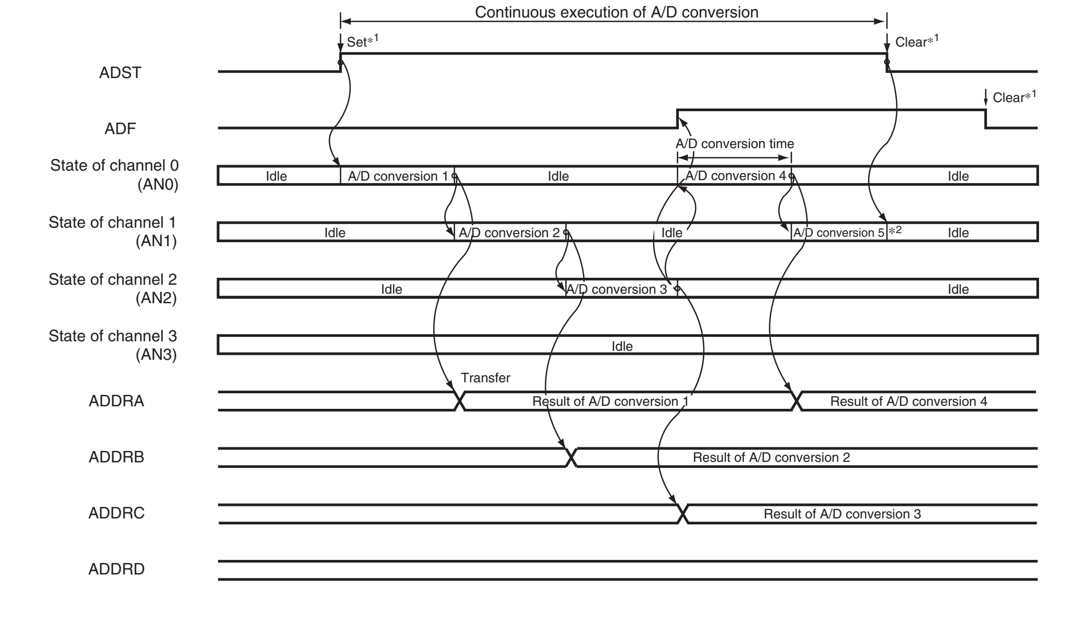
Figure 17.2 Example of A/D Converter Operation (When Channel 1 is Selected in Single Mode)

17.4.2 Scan Mode

In scan mode, A/D conversion is performed sequentially on the specified channels (four channels or eight channel maximum). Operations are as follows.

1. When the ADST bit in ADCSR is set to 1 by software or an external trigger input, A/D conversion starts from the first channel of the selected channel. Consecutive A/D conversion of either four channels maximum (SCANE and SCANS = B'10) or eight channels maximum (SCANE and SCANS = B'11) can be selected. In the case of consecutive A/D conversion on four channels, the operation starts from AN0 when CH2 = B'0, and starts from AN4 when CH2 = B'1. In the case of consecutive A/D conversion on eight channels, the operation starts from AN0.
2. When A/D conversion for each channel is completed, the result is sequentially transferred to the A/D data register corresponding to each channel.
3. When conversion of all the selected channels is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt is requested after A/D conversion ends. Conversion of the first channel in the group starts again.

4. The ADST bit is not automatically cleared to 0 and steps 2 to 3 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters the idle state. After that, when the ADST bit is set to 1, the operation starts from the first channel again.



Notes : 1. ↓ indicates execution of a software instruction.
2. The data being converted is ignored

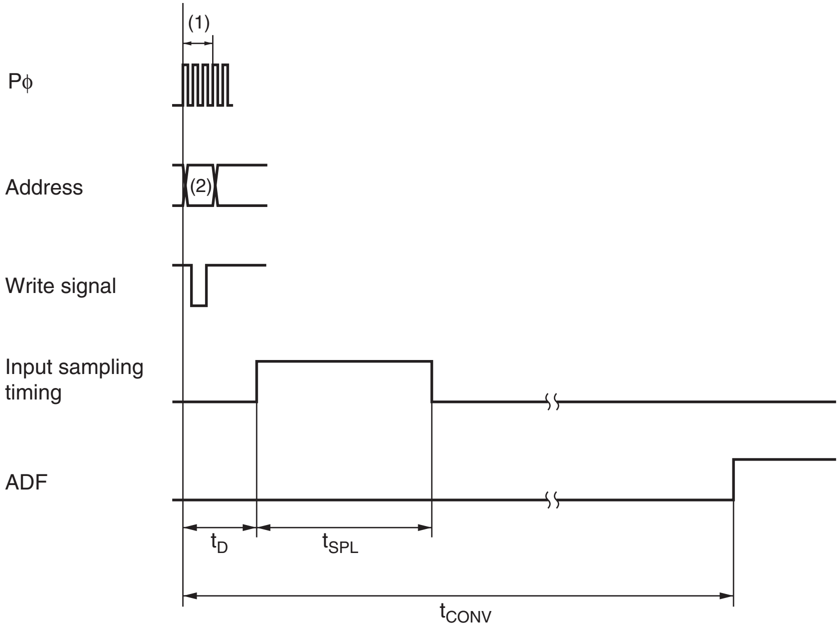
Figure 17.3 Example of A/D Converter Operation (When Channels AN0 to AN3 are Selected in Scan Mode)

17.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input when the A/D conversion start delay time (t_D) passes after the ADST bit in ADCSR is set to 1, then starts A/D conversion. Figure 17.4 shows the A/D conversion timing. Table 17.3 indicates the A/D conversion time.

As indicated in figure 17.4, the A/D conversion time (t_{CONV}) includes t_D and the input sampling time (t_{SPL}). The length of t_D varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 17.3.

In scan mode, the values given in table 17.3 apply to the first conversion time. In the second and subsequent conversions, the conversion time is as shown in table 17.4. In either case, set the CKS1 and CKS0 bits in ADCR so that the conversion time falls within the range of A/D conversion characteristics.



[Legend]
(1): ADCSR write cycle
(2): ADCSR address
t_D: A/D conversion start delay
t_{SPL}: Input sampling time
t_{CONV}: A/D conversion time

Figure 17.4 A/D Conversion Timing

Table 17.3 A/D Conversion Characteristics (Single Mode)

			CKS1 = 0			CKS1 = 1	
			CKS0 = 1			CKS0 = 0	
Item	**Symbol**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**
A/D conversion start delay time	t_D	(6)	—	(9)	(10)	—	(17)
Input sampling time	t_{SPL}	—	30	—	—	60	—
A/D conversion time	t_{CONV}	77	—	80	153	—	160

Note: Values in the table indicate the number of states.

Table 17.4 A/D Conversion Time (Scan Mode)

CKS1	CKS0	Conversion Time (States)
0	0	Setting prohibited
	1	80 (fixed)
0	0	160 (fixed)
	1	Setting prohibited

17.4.4 Timing of External Trigger Input

A/D conversion can also be started by an externally input trigger signal. Setting the TRGS1 and TRGS0 bits in ADCSR to B'11 selects the signal on the $\overline{\text{ADTRG}}$ pin as an external trigger. The ADST bit in ADCSR is set to 1 on the falling edge of $\overline{\text{ADTRG}}$, initiating A/D conversion. Other operations are the same as those in the case where the ADST bit is set to 1 by software, regardless of whether the converter is in single mode or scan mode. The timing of this operation is shown in figure 17.5.

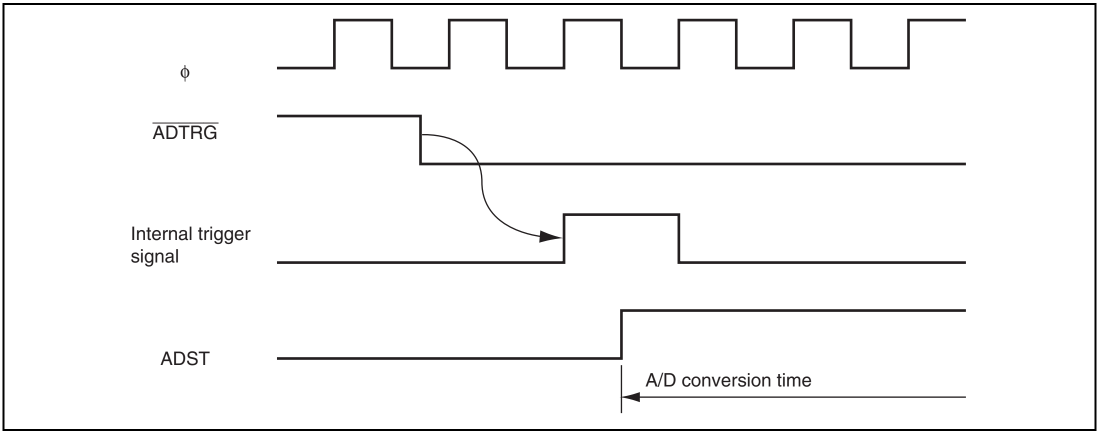


Figure 17.5 Timing of External Trigger Input

17.5 Interrupt Source

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 enables ADI interrupt requests while the ADF bit in ADCSR is set to 1 after A/D conversion ends. The ADI interrupt can be used to activate the DTC. Reading the converted data by the DTC activated by the ADI interrupt allows consecutive conversion to be performed without software overhead.

Table 17.5 A/D Converter Interrupt Source

Name	Interrupt Source	Interrupt Flag	DTC Activation
ADI	A/D conversion end	ADF	Possible

17.6 A/D Conversion Accuracy Definitions

This LSI’s A/D conversion accuracy definitions are given below.

- Resolution

  The number of A/D converter digital output codes
- Quantization error

  The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 17.6).
- Offset error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'00 0000 0000 (H'000) to B'00 0000 0001 (H'001) (see figure 17.7).
- Full-scale error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'11 1111 1110 (H'3FE) to B'11 1111 1111 (H'3FF) (see figure 17.7).
- Nonlinearity error

  The error with respect to the ideal A/D conversion characteristics between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error (see figure 17.7).
- Absolute accuracy

  The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.

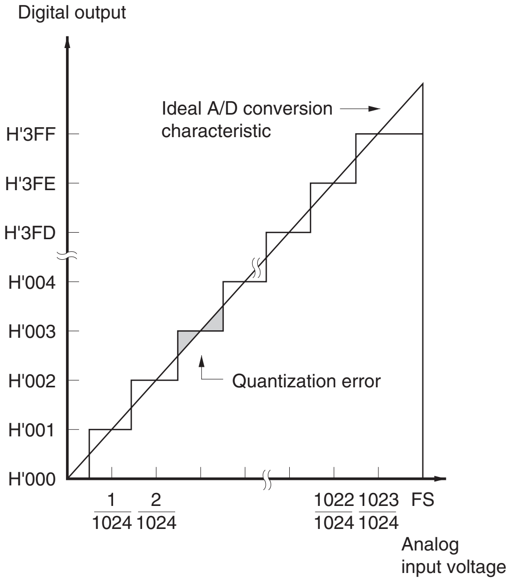


Figure 17.6 A/D Conversion Accuracy Definitions

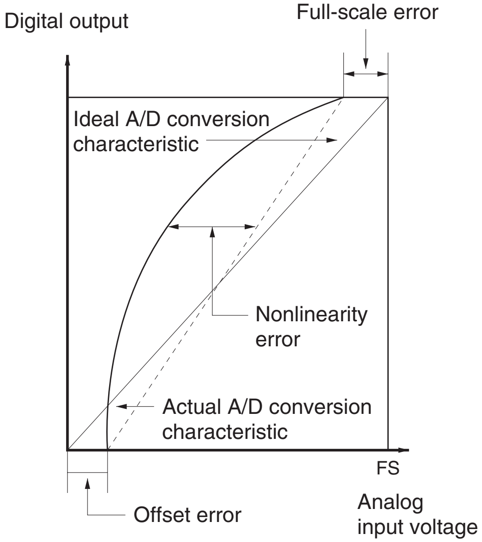


Figure 17.7 A/D Conversion Accuracy Definitions

17.7 Usage Notes

17.7.1 Setting of Module Stop Mode

Operation of the A/D converter can be enabled or disabled by setting the module stop control register. By default, the A/D converter is stopped. Registers of the A/D converter only become accessible when it is released from module stop mode. See section 22, Power-Down Modes, for details.

17.7.2 Permissible Signal Source Impedance

This LSI’s analog input is designed so that the conversion accuracy is guaranteed for an input signal for which the signal source impedance is 5 kΩ or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 5 kΩ, charging may be insufficient and it may not be possible to guarantee the A/D conversion accuracy. However, if a large capacitance is provided externally in single mode, the input load will essentially comprise only the internal input resistance of 10 kΩ, and the signal source impedance is ignored. However, since a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (voltage fluctuation ratio of 5 mV/μs or greater for example) (see figure 17.8). When converting a high-speed analog signal or converting in scan mode, a low-impedance buffer should be inserted.

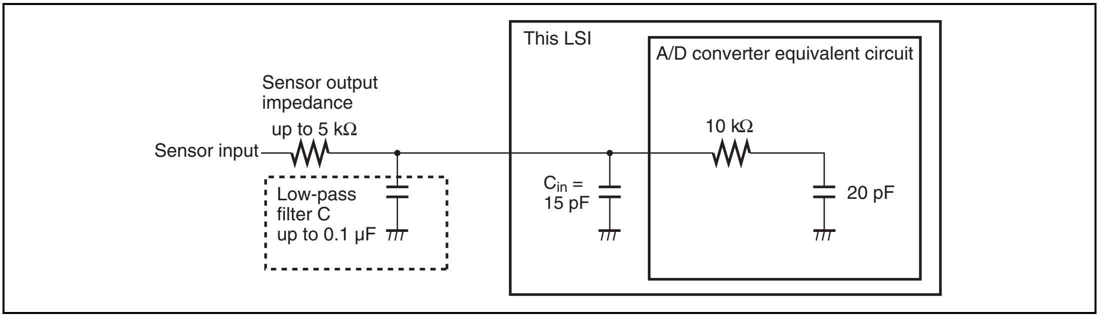


Figure 17.8 Example of Analog Input Circuit

17.7.3 Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect the absolute accuracy. Be sure to make the connection to an electrically stable GND such as AVss.

Care is also required to insure that filter circuits do not communicate with digital signals on the mounting board, so acting as antennas.

17.7.4 Setting Range of Analog Power Supply and Other Pins

If conditions shown below are not met, the reliability of this LSI may be adversely affected.

- Analog input voltage range

  The voltage applied to analog input pin ANn during A/D conversion should be in the range AVss ≤ V_{AN} ≤ AVref.
- Relation between AVcc, AVss and Vcc, Vss

  The relationship between AVcc, AVss and Vcc, Vss should be Avcc = Vcc ± 0.3V and AVss = Vss. When the A/D converter is not used, set AVcc = Vcc and Avss = Vss.
- AVref pin reference voltage specification range

  The reference voltage of the AVref pin should be in the range AVref ≤ AVcc.

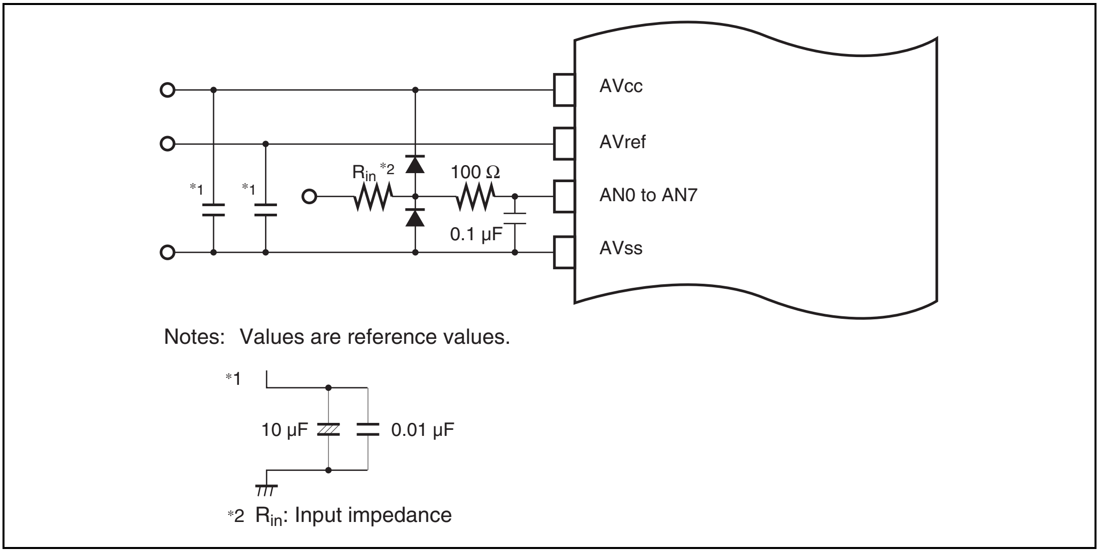
17.7.5 Notes on Board Design

In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values. Also, digital circuitry must be isolated from the analog input signals (AN0 to AN7), the analog reference voltage (AVref) and analog power supply (AVcc) by the analog ground (AVss). Also, the analog ground (AVss) should be connected at one point to a stable digital ground (Vss) on the board.

17.7.6 Notes on Noise Countermeasures

In order to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN7), a protection circuit should be connected between AVcc and AVss as shown in figure 17.9. Also, the bypass capacitors connected to AVcc and the filter capacitors connected to AN0 to AN7 must be connected to AVss.

When a filter capacitor is connected, the input currents at the analog input pins (AN0 to AN7) are averaged which may cause an error. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance (R_{in}), an error will arise in the analog input pin voltage. Therefore, careful consideration is required upon deciding the circuit constants.



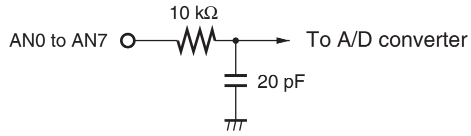
Notes: Values are reference values.

*2 R_{in}: Input impedance

Figure 17.9 Example of Analog Input Protection Circuit

Table 17.6 Standard of Analog Pins

Item	Min.	Max.	Unit
Analog input capacitance	—	20	pF
Acceptable signal source impedance	—	5	kΩ



Note: Values are reference values.

Figure 17.10 Analog Input Pin Equivalent Circuit

17.7.7 Note on the Usage in Software Standby Mode

If this LSI enters software standby mode with the A/D conversion enabled, the content of the A/D converter is retained and about the same amount of analog supply current may flow as that flows when A/D conversion in progress. If the analog supply current must be reduced in software standby mode, clear the ADST bit to disable the A/D conversion.

Section 18 RAM

This LSI has 40 kbytes of on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling one-state access by the CPU to both byte data and word data.

The on-chip RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on SYSCR, see section 3.2.2, System Control Register (SYSCR).

Section 19 Flash Memory

The flash memory has the following features. Figure 19.1 shows a block diagram of the flash memory.

19.1 Features

- Size

  512 Kbytes (ROM address: H'000000 to H'07FFFF)
- Programming/erasing interface by the download of on-chip program

  This LSI has a dedicated programming/erasing program. After downloading this program to the on-chip RAM, programming/erasing can be performed by setting the argument parameter.
- Programming/erasing time

  The flash memory programming time is 1 ms (typ) in 128-byte simultaneous programming and approximately 7.8 µs per byte. The erasing time is 600 ms (typ) per 64-Kbyte block.
- Number of programming

  The number of flash memory programming can be up to 1000 times at the minimum. (The value ranged from 1 to 1000 is guaranteed.)
- Three on-board programming modes
  - Boot mode

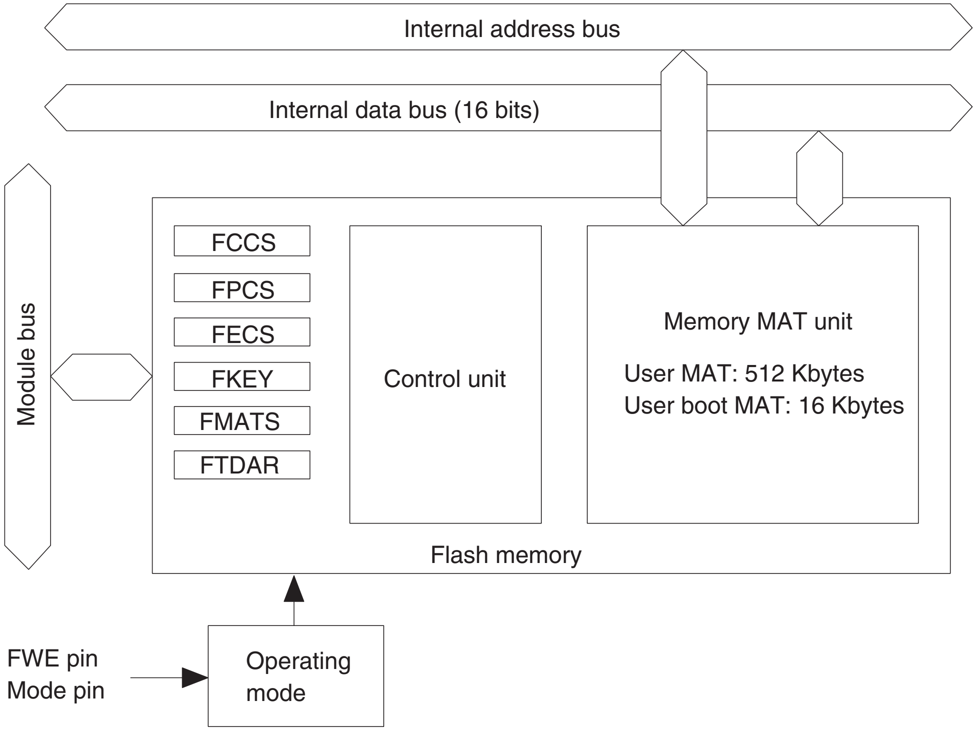
    This mode is a program mode that uses an on-chip SCI interface. The user MAT and user boot MAT can be programmed. This mode can automatically adjust the bit rate between host and this LSI.
  - User program mode

    The user MAT can be programmed by using the optional interface.
  - User boot mode

    The user boot program of the optional interface can be made and the user MAT can be programmed.
- Programming/erasing protection

  Sets protection against flash memory programming/erasing via hardware, software, or error protection.
- Programmer mode

  This mode uses the PROM programmer. The user MAT and user boot MAT can be programmed.



[Legend]
FCCS: Flash code control status register
FPCS: Flash program code select register
FECS: Flash erase code select register
FKEY: Flash key code register
FMATS: Flash MAT select register
FTDAR: Flash transfer destination address register

Note: To read from or write to the registers, the FLSHE bit in the serial timer control register (STCR) must be set to 1.

Figure 19.1 Block Diagram of Flash Memory

19.1.1 Operating Mode

When each mode pin and the FWE pin are set in the reset state and reset start is performed, this LSI enters each operating mode as shown in figure 19.2.

- Flash memory can be read in user mode, but cannot be programmed or erased.
- Flash memory can be read, programmed, or erased on the board only in boot mode, user program mode, and user boot mode.
- Flash memory can be read, programmed, or erased by means of the PROM programmer in programmer mode.

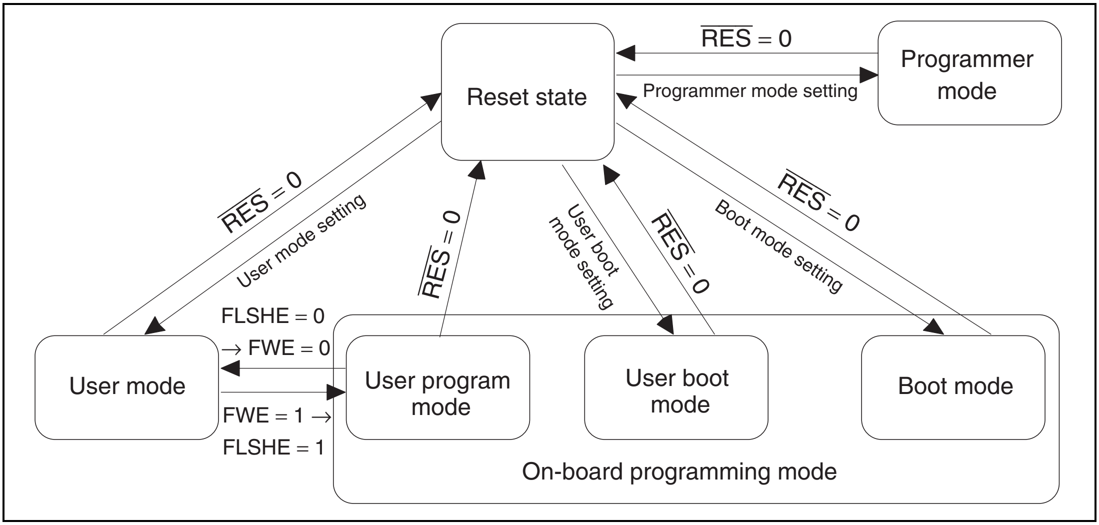


Figure 19.2 Mode Transition of Flash Memory

19.1.2 Mode Comparison

The comparison table of programming and erasing related items about boot mode, user program mode, user boot mode, and programmer mode is shown in table 19.1.

Table 19.1 Comparison of Programming Modes

	Boot mode	User program mode	User boot mode	Programmer mode
Programming/ erasing environment	On-board	On-board	On-board	PROM programmer
Programming/ erasing enable MAT	User MAT User boot MAT	User MAT	User MAT	User MAT User boot MAT
All erasure	○ (Automatic)	○	○	○ (Automatic)
Block division erasure	○ *1	○	○	×
Program data transfer	From host via SCI	Via optional device	Via optional device	Via programmer
Reset initiation MAT	Embedded program storage MAT	User MAT	User boot MAT*2	—
Transition to user mode	Changing mode setting and reset	Changing FLSHE bit and FWE pin	Changing mode setting and reset	—

Notes: 1. All-erasure is performed. After that, the specified block can be erased.

2. Firstly, the reset vector is fetched from the embedded program storage MAT. After the flash memory related registers are checked, the reset vector is fetched from the user boot MAT.

- The user boot MAT can be programmed or erased only in boot mode and programmer mode.
- The user MAT and user boot MAT are erased in boot mode. Then, the user MAT and user boot MAT can be programmed by means of the command method. However, the contents of the MAT cannot be read until this state.
  Only user boot MAT is programmed and the user MAT is programmed in user boot mode or only user MAT is programmed because user boot mode is not used.
- The boot operation of the optional interface can be performed by the mode pin setting different from user program mode in user boot mode.

19.1.3 Flash Memory MAT Configuration

This LSI's flash memory is configured by the 16-Kbyte user boot MAT and 256-Kbyte user MAT.

The start address is allocated to the same address in the user MAT and user boot MAT. Therefore, when the program execution or data access is performed between two MATs, the MAT must be switched by using FMATS.

The user MAT or user boot MAT can be read in all modes. However, the user boot MAT can be programmed only in boot mode and programmer mode.

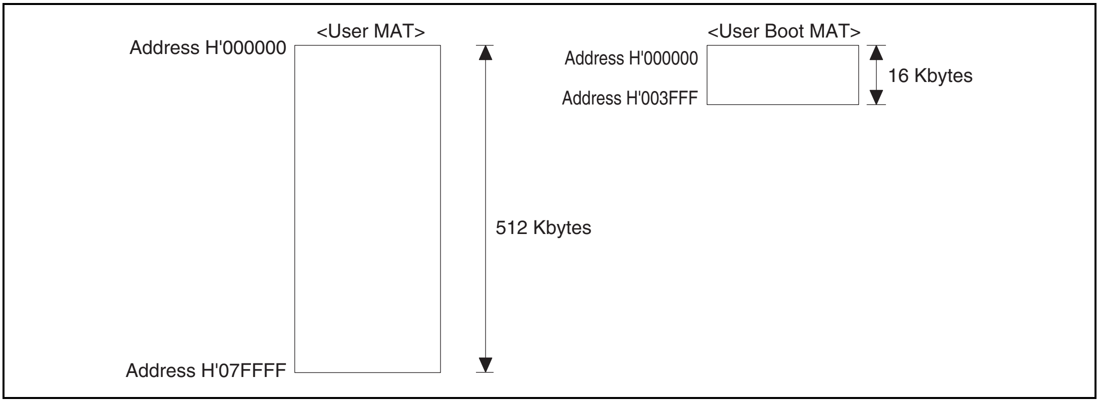
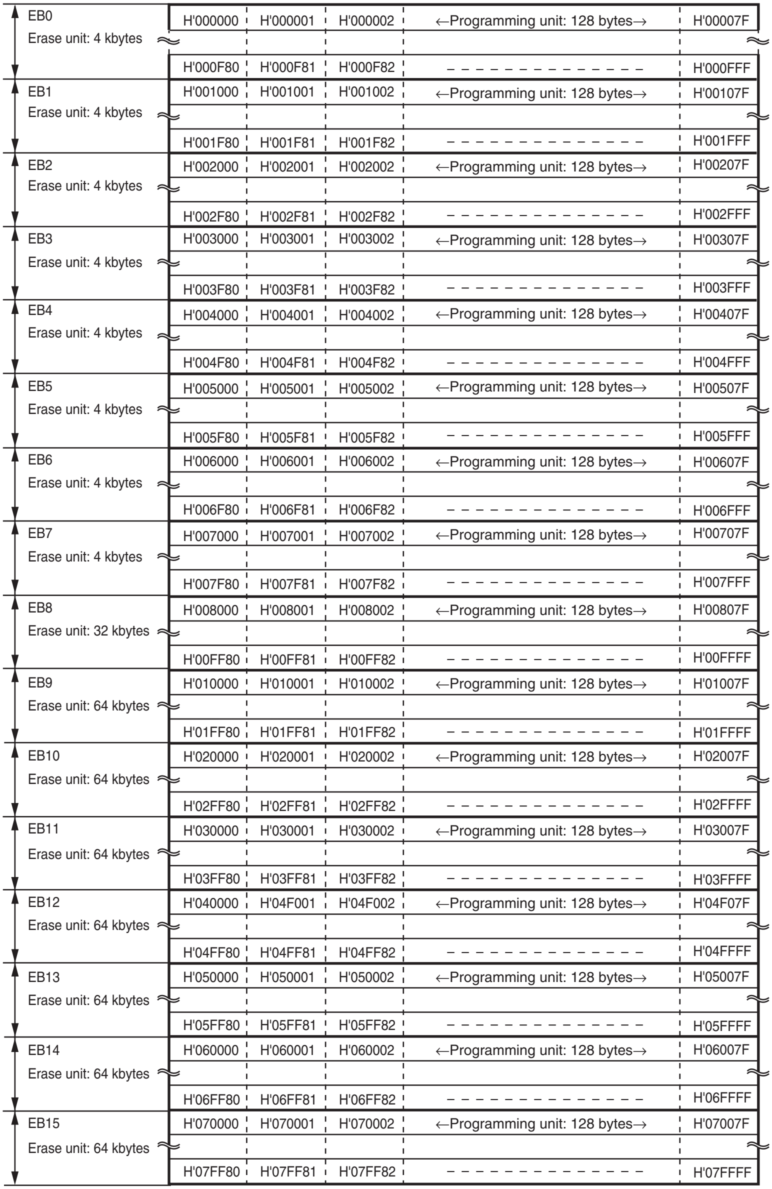


Figure 19.3 Flash Memory Configuration

The size of the user MAT is different from that of the user boot MAT. An address which exceeds the size of the 16-Kbyte user boot MAT should not be accessed. If the attempt is made, data is read as undefined value.

19.1.4 Block Division

The user MAT is divided into seven 64-Kbyte blocks, one 32-Kbyte block, and eight 4-Kbyte blocks as shown in figure 19.4. The user MAT can be erased in this divided-block units, and the erase-block number of EB0 to EB15 is specified when erasing. Programming is performed in 128-byte units starting at the addresses whose lowest-order byte is H'00 or H'80.



EB0	Erase unit: 4 kbytes	H'000000	H'000001	H'000002	←Programming unit: 128 bytes→	H'00007F
		H'000F80	H'000F81	H'000F82	– – – – – – – – – – – – – –	H'000FFF
EB1	Erase unit: 4 kbytes	H'001000	H'001001	H'001002	←Programming unit: 128 bytes→	H'00107F
		H'001F80	H'001F81	H'001F82	– – – – – – – – – – – – – –	H'001FFF
EB2	Erase unit: 4 kbytes	H'002000	H'002001	H'002002	←Programming unit: 128 bytes→	H'00207F
		H'002F80	H'002F81	H'002F82	– – – – – – – – – – – – – –	H'002FFF
EB3	Erase unit: 4 kbytes	H'003000	H'003001	H'003002	←Programming unit: 128 bytes→	H'00307F
		H'003F80	H'003F81	H'003F82	– – – – – – – – – – – – – –	H'003FFF
EB4	Erase unit: 4 kbytes	H'004000	H'004001	H'004002	←Programming unit: 128 bytes→	H'00407F
		H'004F80	H'004F81	H'004F82	– – – – – – – – – – – – – –	H'004FFF
EB5	Erase unit: 4 kbytes	H'005000	H'005001	H'005002	←Programming unit: 128 bytes→	H'00507F
		H'005F80	H'005F81	H'005F82	– – – – – – – – – – – – – –	H'005FFF
EB6	Erase unit: 4 kbytes	H'006000	H'006001	H'006002	←Programming unit: 128 bytes→	H'00607F
		H'006F80	H'006F81	H'006F82	– – – – – – – – – – – – – –	H'006FFF
EB7	Erase unit: 4 kbytes	H'007000	H'007001	H'007002	←Programming unit: 128 bytes→	H'00707F
		H'007F80	H'007F81	H'007F82	– – – – – – – – – – – – – –	H'007FFF
EB8	Erase unit: 32 kbytes	H'008000	H'008001	H'008002	←Programming unit: 128 bytes→	H'00807F
		H'00FF80	H'00FF81	H'00FF82	– – – – – – – – – – – – – –	H'00FFFF
EB9	Erase unit: 64 kbytes	H'010000	H'010001	H'010002	←Programming unit: 128 bytes→	H'01007F
		H'01FF80	H'01FF81	H'01FF82	– – – – – – – – – – – – – –	H'01FFFF
EB10	Erase unit: 64 kbytes	H'020000	H'020001	H'020002	←Programming unit: 128 bytes→	H'02007F
		H'02FF80	H'02FF81	H'02FF82	– – – – – – – – – – – – – –	H'02FFFF
EB11	Erase unit: 64 kbytes	H'030000	H'030001	H'030002	←Programming unit: 128 bytes→	H'03007F
		H'03FF80	H'03FF81	H'03FF82	– – – – – – – – – – – – – –	H'03FFFF
EB12	Erase unit: 64 kbytes	H'040000	H'04F001	H'04F002	←Programming unit: 128 bytes→	H'04F07F
		H'04FF80	H'04FF81	H'04FF82	– – – – – – – – – – – – – –	H'04FFFF
EB13	Erase unit: 64 kbytes	H'050000	H'050001	H'050002	←Programming unit: 128 bytes→	H'05007F
		H'05FF80	H'05FF81	H'05FF82	– – – – – – – – – – – – – –	H'05FFFF
EB14	Erase unit: 64 kbytes	H'060000	H'060001	H'060002	←Programming unit: 128 bytes→	H'06007F
		H'06FF80	H'06FF81	H'06FF82	– – – – – – – – – – – – – –	H'06FFFF
EB15	Erase unit: 64 kbytes	H'070000	H'070001	H'070002	←Programming unit: 128 bytes→	H'07007F
		H'07FF80	H'07FF81	H'07FF82	– – – – – – – – – – – – – –	H'07FFFF

Figure 19.4 Block Division of User MAT

19.1.5 Programming/Erasing Interface

Programming/erasing is executed by downloading the on-chip program to the on-chip RAM and specifying the program address/data and erase block by using the interface register/parameter.

The procedure program is made by the user in user program mode and user boot mode. An overview of the procedure is given as follows. For details, see section 19.4.2, User Program Mode.

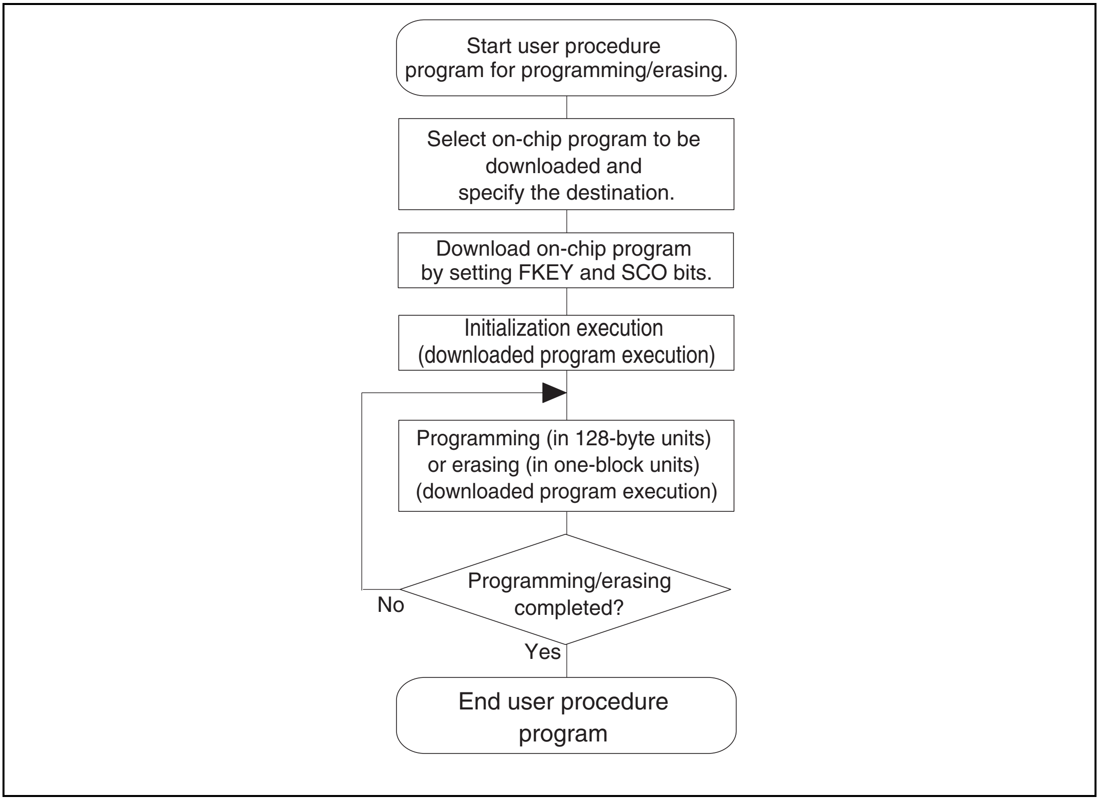

Start user procedure program for programming/erasing.
Select on-chip program to be downloaded and specify the destination.
Download on-chip program by setting FKEY and SCO bits.
Initialization execution (downloaded program execution)
Programming (in 128-byte units) or erasing (in one-block units) (downloaded program execution)
Programming/erasing completed?
No
Yes
End user procedure program

Figure 19.5 Overview of User Procedure Program

1. Selection of on-chip program to be downloaded

   For programming/erasing execution, the FLSHE bit in STCR must be set to 1 to transition to user program mode.

   This LSI has programming/erasing programs which can be downloaded to the on-chip RAM. The on-chip program to be downloaded is selected by setting the corresponding bits in the programming/erasing interface register. The address of the programming destination is specified by the flash transfer destination address register (FTDAR).

2. Download of on-chip program

   The on-chip program is automatically downloaded by setting the flash key code register (FKEY) and the SCO bit in the flash code control status register (FCCS), which are programming/erasing interface registers.

   The flash memory is replaced to the embedded program storage area when downloading. Since the flash memory cannot be read when programming/erasing, the procedure program, which is working from download to completion of programming/erasing, must be executed in the space other than the flash memory to be programmed/erased (for example, on-chip RAM).

   Since the result of download is returned to the programming/erasing interface parameter, whether the normal download is executed or not can be confirmed.

3. Initialization of programming/erasing

   The operating frequency is set before execution of programming/erasing. This setting is performed by using the programming/erasing interface parameter.

4. Programming/erasing execution

   For programming/erasing execution, the FLSHE bit in STCR and the FWE pin must be set to 1 to transition to user program mode.

   The program data/programming destination address is specified in 128-byte units when programming.

   The block to be erased is specified in erase-block units when erasing.

   These specifications are set by using the programming/erasing interface parameter and the on-chip program is initiated. The on-chip program is executed by using the JSR or BSR instruction and performing the subroutine call of the specified address in the on-chip RAM. The execution result is returned to the programming/erasing interface parameter.

   The area to be programmed must be erased in advance when programming flash memory.

   All interrupts are prohibited during programming and erasing. Interrupts must be masked within the user system.

5. When programming/erasing is executed consecutively

   When the processing is not ended by the 128-byte programming or one-block erasure, the program address/data and erase-block number must be updated and consecutive programming/erasing is required.

   Since the downloaded on-chip program is left in the on-chip RAM after the processing, download and initialization are not required when the same processing is executed consecutively.

19.2 Input/Output Pins

Table 19.2 shows the flash memory pin configuration.

Table 19.2 Pin Configuration

Pin Name	Input/Output	Function
$\overline{\text{RES}}$	Input	Reset
FWE	Input	Flash memory programming/erasing enable pin
$\overline{\text{MD2}}$	Input	Sets operating mode of this LSI
MD1	Input	Sets operating mode of this LSI
TxD1	Output	Serial transmit data output (used in boot mode)
RxD1	Input	Serial receive data input (used in boot mode)

19.3 Register Descriptions

The registers/parameters which control flash memory are shown in the following. To read from or write to these registers/parameters, the FLSHE bit in the serial timer control register (STCR) must be set to 1. For details on STCR, see section 3.2.3, Serial Timer Control Register (STCR).

- Flash code control status register (FCCS)
- Flash program code select register (FPCS)
- Flash erase code select register (FECS)
- Flash key code register (FKEY)
- Flash MAT select register (FMATS)
- Flash transfer destination address register (FTDAR)
- Download pass/fail result (DPFR)
- Flash pass/fail result (FPFR)
- Flash multipurpose address area (FMPAR)
- Flash multipurpose data destination area (FMPDR)
- Flash erase Block select (FEBS)
- Flash programming/erasing frequency control (FPEFEQ)

There are several operating modes for accessing flash memory, for example, read mode/program mode.

There are two memory MATs: user MAT and user boot MAT. The dedicated registers/parameters are allocated for each operating mode and MAT selection. The correspondence of operating modes and registers/parameters for use is shown in table 19.3.

Table 19.3 Register/Parameter and Target Mode

		Download	Initialization	Programming	Erasure	Read
Programming/ Erasing Interface Register	FCCS	○	—	—	—	—
	FPCS	○	—	—	—	—
	FECS	○	—	—	—	—
	FKEY	○	—	○	○	—
	FMATS	—	—	○ *[1]	○ *[1]	○ *[2]
	FTDAR	○	—	—	—	—
Programming/ Erasing Interface Parameter	DPFR	○	—	—	—	—
	FPFR	—	○	○	○	—
	FPEFEQ	—	○	—	—	—
	FMPAR	—	—	○	—	—
	FMPDR	—	—	○	—	—
	FEBS	—	—	—	○	—

Notes: 1. The setting is required when programming or erasing user MAT in user boot mode.

2. The setting may be required according to the combination of initiation mode and read target MAT.

19.3.1 Programming/Erasing Interface Register

The programming/erasing interface registers are as described below. They are all 8-bit registers that can be accessed in byte. These registers are initialized at a reset or in hardware standby mode.

- Flash Code Control Status Register (FCCS)

FCCS is configured by bits which request the monitor of the FWE pin state and error occurrence during programming or erasing flash memory and the download of on-chip program.

Bit	Bit Name	Initial Value	R/W	Description
7	FWE	1/0	R	Flash Program Enable Monitors the signal level input to the FWE pin and enables or disables programming/erasing flash memory. 0: Programming/erasing disabled 1: Programming/erasing enabled
6, 5	—	All 0	R/W	Reserved The initial value should not be changed.

Bit	Bit Name	Initial Value	R/W	Description
4	FLER	0	R	Flash Memory Error Indicates an error occurs during programming and erasing flash memory. When FLER is set to 1, flash memory enters the error protection state. When FLER is set to 1, high voltage is applied to the internal flash memory. To reduce the damage to flash memory, the reset must be released after the reset period of 100 μs which is longer than normal. 0: Flash memory operates normally. Programming/erasing protection for flash memory (error protection) is invalid. [Clearing condition] • At a reset or in hardware standby mode 1: An error occurs during programming/erasing flash memory. Programming/erasing protection for flash memory (error protection) is valid. [Setting conditions] • When an interrupt, such as NMI, occurs during programming/erasing flash memory. • When the flash memory is read during programming/erasing flash memory (including a vector read or an instruction fetch). • When the SLEEP instruction is executed during programming/erasing flash memory (including software-standby mode) • When a bus master other than the CPU, such as the DTC, gets bus mastership during programming/erasing flash memory.

Bit	Bit Name	Initial Value	R/W	Description
3	WEINTE	0	R/W	Program/Erase Enable Modifies the space for the interrupt vector table, when interrupt vector data is not read successfully during programming/erasing flash memory or switching between a user MAT and a user boot MAT. When this bit is set to 1, interrupt vector data is read from address spaces H'FFE080 to H'FFE0FF (on-chip RAM space), instead of from address spaces H'000000 to H'00007F (up to vector number 31). Therefore, make sure to set the vector table in the on-chip RAM space before setting this bit to 1. The interrupt exception handling on and after vector number 32 should not be used because the correct vector is not read, resulting in the CPU runaway. 0: The space for the interrupt vector table is not modified. When interrupt vector data is not read successfully, the operation for the interrupt exception handling cannot be guaranteed. An occurrence of any interrupts should be masked. 1: The space for the interrupt vector table is modified. Even when interrupt vector data is not read successfully, the interrupt exception handling up to vector number 31 is enabled.
2, 1	—	All 0	R/W	Reserved The initial value should not be changed.

Bit	Bit Name	Initial Value	R/W	Description
0	SCO	0	(R)/W*	Source Program Copy Operation Requests the on-chip programming/erasing program to be downloaded to the on-chip RAM. When this bit is set to 1, the on-chip program which is selected by FPCS/FECS is automatically downloaded in the on-chip RAM specified by FTDAR. In order to set this bit to 1, H′A5 must be written to FKEY and this operation must be executed in the on-chip RAM. Four NOP instructions must be executed immediately after setting this bit to 1. Since this bit is cleared to 0 when download is completed, this bit cannot be read as 1. All interrupts must be disabled. This should be made in the user system. 0: Download of the on-chip programming/erasing program to the on-chip RAM is not executed. [Clearing condition] When download is completed 1: Request that the on-chip programming/erasing program is downloaded to the on-chip RAM is occurred. [Setting conditions] When all of the following conditions are satisfied and 1 is set to this bit • H'A5 is written to FKEY • During execution in the on-chip RAM

Note: * This bit is a write only bit. This bit is always read as 0.

- Flash Program Code Select Register (FPCS)

FPCS selects the on-chip programming program to be downloaded.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R/W	Reserved The initial value should not be changed.
0	PPVS	0	R/W	Program Pulse Verify Selects the programming program. 0: On-chip programming program is not selected. [Clearing condition] When transfer is completed 1: On-chip programming program is selected.

- Flash Erase Code Select Register (FECS)

FECS selects download of the on-chip erasing program.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R/W	Reserved The initial value should not be changed.
0	EPVB	0	R/W	Erase Pulse Verify Block Selects the erasing program. 0: On-chip erasing program is not selected. [Clearing condition] When transfer is completed 1: On-chip erasing program is selected.

- Flash Key Code Register (FKEY)

FKEY is a register for software protection that enables download of on-chip program and programming/erasing of flash memory. Before setting the SCO bit to 1 in order to download on-chip program or executing the downloaded programming/erasing program, these processing cannot be executed if the key code is not written.

Bit	Bit Name	Initial Value	R/W	Description
7	K7	0	R/W	Key Code Only when H'A5 is written, writing to the SCO bit is valid. When the value other than H'A5 is written to FKEY, 1 cannot be set to the SCO bit. Therefore downloading to the on-chip RAM cannot be executed. Only when H'5A is written, programming/erasing can be executed. Even if the on-chip programming/erasing program is executed, the flash memory cannot be programmed or erased when the value other than H'5A is written to FKEY. H'A5: Writing to the SCO bit is enabled. (The SCO bit cannot be set by the value other than H'A5.) H'5A: Programming/erasing is enabled. (The value other than H'A5 is in software protection state.) H'00: Initial value
6	K6	0	R/W	
5	K5	0	R/W	
4	K4	0	R/W	
3	K3	0	R/W	
2	K2	0	R/W	
1	K1	0	R/W	
0	K0	0	R/W	

- Flash MAT Select Register (FMATS)

FMATS specifies whether user MAT or user boot MAT is selected.

Bit	Bit Name	Initial Value	R/W	Description
7	MS7	0/1*	R/W	MAT Select
6	MS6	0	R/W	These bits are in user-MAT selection state when the value other than H'AA is written and in user-boot-MAT selection state when H'AA is written.
5	MS5	0/1*	R/W	The MAT is switched by writing the value in FMATS.
4	MS4	0	R/W	When the MAT is switched, follow section 19.6, Switching between User MAT and User Boot MAT. (The user boot MAT cannot be programmed in user program mode if user boot MAT is selected by FMATS. The user boot MAT must be programmed in boot mode or in programmer mode.)
3	MS3	0/1*	R/W	H'AA: The user boot MAT is selected (in user-MAT selection state when the value of these bits are other than H'AA) Initial value when these bits are initiated in user boot mode.
2	MS2	0	R/W	H'00: Initial value when these bits are initiated in a mode except for user boot mode (in user-MAT selection state)
1	MS1	0/1*	R/W	[Programmable condition] These bits are in the execution state in the on-chip RAM.
0	MS0	0	R/W	

Note: * Set to 1 when in user boot mode, otherwise set to 0.

- Flash Transfer Destination Address Register (FTDAR)

FTDAR is a register that specifies the address to download an on-chip program. This register must be specified before setting the SCO bit in FCCS to 1.

Bit	Bit Name	Initial Value	R/W	Description
7	TDER	0	R/W	Transfer Destination Address Setting Error This bit is set to 1 when the address specified by bits TDA6 to TDA0, which is the start address to download an on-chip program, is over the range. Whether or not the range specified by bits TDA6 to TDA0 is within the range of H'00 to H'03 is determined when an on-chip program is downloaded by setting the SCO bit in FCCS to 1. Make sure that this bit is cleared to 0 before setting the SCO bit to 1 and the value specified by TDA6 to TDA0 is within the range of H'00 to H'03. 0: The value specified by bits TDA6 to TDA0 is within the range. 1: The value specified by is TDA6 to TDA0 is over the range (H'04 to H'FF) and the download is stopped.
6	TDA6	0	R/W	Transfer Destination Address Specifies the start address to download an on-chip program. H'00 to H'03 can be specified as the start address in the on-chip RAM space. H'00: H'FFE080 is specified as a start address to download an on-chip program. H'01: H'FF0800 is specified as a start address to download an on-chip program. H'02: H'FF1800 is specified as a start address to download an on-chip program. H'03: H'FF8800 is specified as a start address to download an on-chip program. H'04 to H'FF: Setting prohibited. Specifying this value sets the TDER bit to 1 and stops the download.
5	TDA5	0	R/W	
4	TDA4	0	R/W	
3	TDA3	0	R/W	
2	TDA2	0	R/W	
1	TDA1	0	R/W	
0	TDA0	0	R/W	

19.3.2 Programming/Erasing Interface Parameter

The programming/erasing interface parameter specifies the operating frequency, storage place for program data, programming destination address, and erase block and exchanges the processing result for the downloaded on-chip program. This parameter uses the general registers of the CPU (ER0 and ER1) or the on-chip RAM area. The initial value is undefined at a reset or in hardware standby mode.

When download, initialization, or on-chip program is executed, registers of the CPU except for R0L are stored. The return value of the processing result is written in R0L. Since the stack area is used for storing the registers except for R0L, the stack area must be saved at the processing start. (A maximum size of a stack area to be used is 128 bytes.)

The programming/erasing interface parameter is used in the following four items.

1. Download control
2. Initialization before programming or erasing
3. Programming
4. Erasing

These items use different parameters. The correspondence table is shown in table 19.4. The meaning of the bits in FPFR varies in each processing program: initialization, programming, or erasure. For details, see descriptions of FPFR for each process.

Table 19.4 Parameters and Target Modes

Name of Parameter	Abbreviation	Down Load	Initialization	Programming	Erasure	R/W	Initial Value	Allocation
Download pass/fail result	DPFR	○	—	—	—	R/W	Undefined	On-chip RAM*
Flash pass/fail result	FPFR	—	○	○	○	R/W	Undefined	R0L of CPU
Flash programming/ erasing frequency control	FPEFEQ	—	○	—	—	R/W	Undefined	ER0 of CPU
Flash multipurpose address area	FMPAR	—	—	○	—	R/W	Undefined	ER1 of CPU
Flash multipurpose data destination area	FMPDR	—	—	○	—	R/W	Undefined	ER0 of CPU
Flash erase block select	FEBS	—	—	—	○	R/W	Undefined	R0L of CPU

Note: * A single byte of the start address to download an on-chip program, which is specified by FTDAR

(1) Download Control

The on-chip program is automatically downloaded by setting the SCO bit to 1. The on-chip RAM area to be downloaded is the 3-Kbyte area starting from the address specified by FTDAR.

Download control is set by the program/erase interface registers, and the DPFR parameter indicates the return value.

(a) Download pass/fail result parameter (DPFR: single byte of start address specified by FTDAR)

This parameter indicates the return value of the download result. The value of this parameter can be used to determine if downloading is executed or not. Since the confirmation whether the SCO bit is set to 1 is difficult, the certain determination must be performed by writing the single byte of the start address specified by FTDAR to the value other than the return value of download (for example, H'FF) before the download start (before setting the SCO bit to 1).

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused Return 0
2	SS	—	R/W	Source Select Error Detect Only one type for the on-chip program which can be downloaded can be specified. When more than two types of the program are selected, the program is not selected, or the program is selected without mapping, error is occurred. 0: Download program can be selected normally 1: Download error is occurred (multi-selection or program which is not mapped is selected)
1	FK	—	R/W	Flash Key Register Error Detect Returns the check result whether the value of FKEY is set to H'A5. 0: KEY setting is normal (FKEY = H'A5) 1: Setting value of FKEY becomes error (FKEY = value other than H'A5)

Bit	Bit Name	Initial Value	R/W	Description
0	SF	—	R/W	Success/Fail Returns the result whether download is ended normally or not. The determination result whether program that is downloaded to the on-chip RAM is read back and then transferred to the on-chip RAM is returned. 0: Downloading on-chip program is ended normally (no error) 1: Downloading on-chip program is ended abnormally (error occurs)

(2) Programming/Erasing Initialization

The on-chip programming/erasing program to be downloaded includes the initialization program.

The specified period pulse must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. The operating frequency of the CPU must be set.

The initial program is set as a parameter of the programming/erasing program which has downloaded these settings.

(a) Flash programming/erasing frequency parameter (FPEFEQ: general register ER0 of CPU)

This parameter sets the operating frequency of the CPU. The settable range of the operating frequency in this LSI is 20 to 25 MHz.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	—	—	Unused This bit should be cleared to 0.
15 to 0	F15 to F0	—	R/W	Frequency Set Set the operating frequency of the CPU. With the PLL multiplication function, set the frequency multiplied. The setting value must be calculated as the following methods. 1. The operating frequency which is shown in MHz units must be rounded in a number to three decimal places and be shown in a number of two decimal places. 2. The value multiplied by 100 is converted to the binary digit and is written to the FPEFEQ parameter (general register ER0). For example, when the operating frequency of the CPU is 25.000 MHz, the value is as follows. 1. The number to three decimal places of 25.000 is rounded and the value is thus 25.00. 2. The formula that 25.00 × 100 = 2500 is converted to the binary digit and B'0000,1001,1101,0100 (H'09C4) is set to ER0.

(b) Flash pass/fail parameter (FPFR: general register R0L of CPU)

This parameter indicates the return value of the initialization result.

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	—	—	Unused Return 0
1	FQ	—	R/W	Frequency Error Detect Returns the check result whether the specified operating frequency of the CPU is in the range of the supported operating frequency. 0: Setting of operating frequency is normal 1: Setting of operating frequency is abnormal
0	SF	—	R/W	Success/Fail Indicates whether initialization is completed normally. 0: Initialization is ended normally (no error) 1: Initialization is ended abnormally (error occurs)

(3) Programming Execution

When flash memory is programmed, the programming destination address on the user MAT must be passed to the programming program in which the program data is downloaded.

1. The start address of the programming destination on the user MAT must be stored in a general register ER1. This parameter is called as flash multipurpose address area parameter (FMPAR).

   Since the program data is always in units of 128 bytes, the lower eight bits (A7 to A0) must be H'00 or H'80 as the boundary of the programming start address on the user MAT.
2. The program data for the user MAT must be prepared in the consecutive area. The program data must be in the consecutive space which can be accessed by using the MOV.B instruction of the CPU and in other than the flash memory space.

   When data to be programmed does not satisfy 128 bytes, the 128-byte program data must be prepared by filling with the dummy code H'FF.

   The start address of the area in which the prepared program data is stored must be stored in a general register ER0. This parameter is called as flash multipurpose data destination area parameter (FMPDR).

For details on the program processing procedure, see section 19.4.2, User Program Mode.

(a) Flash multipurpose address area parameter (FMPAR: general register ER1 of CPU)

This parameter stores the start address of the programming destination on the user MAT.

When the address in the area other than flash memory space is set, an error occurs.

The start address of the programming destination must be at the 128-byte boundary. If this boundary condition is not satisfied, an error occurs. The error occurrence is indicated by the WA bit (bit 1) in FPFR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOA31 to MOA0	—	R/W	Store the start address of the programming destination on the user MAT. The consecutive 128-byte programming is executed starting from the specified start address of the user MAT. Therefore, the specified programming start address becomes a 128-byte boundary and MOA6 to MOA0 are always 0.

(b) Flash multipurpose data destination parameter (FMPDR: general register ER0 of CPU):

This parameter stores the start address in the area which stores the data to be programmed in the user MAT. When the storage destination of the program data is in flash memory, an error occurs. The error occurrence is indicated by the WD bit in FPFR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MOD0	—	R/W	Store the start address of the area which stores the program data for the user MAT. The consecutive 128-byte data is programmed to the user MAT starting from the specified start address.

(c) Flash pass/fail parameter (FPFR: general register R0L of CPU)

This parameter indicates the return value of the program processing result.

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Return 0.

Bit	Bit Name	Initial Value	R/W	Description
6	MD	—	R/W	Programming Mode Related Setting Error Detect Returns the check result that a high level signal is input to the FWE pin and the error protection state is not entered. When the low level signal is input to the FWE pin or the error protection state is entered, 1 is written to this bit. The state can be confirmed with the FWE and FLER bits in FCCS. For conditions to enter the error protection state, see section 19.5.3, Error Protection. 0: FWE and FLER settings are normal (FWE = 1, FLER = 0) 1: Programming cannot be performed (FWE = 0 or FLER = 1)
5	EE	—	R/W	Programming Execution Error Detect 1 is returned to this bit when the specified data could not be written because the user MAT was not erased. If this bit is set to 1, there is a high possibility that the user MAT is partially rewritten. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when programming is performed. In this case, both the user MAT and user boot MAT are not rewritten. Programming of the user boot MAT should be performed in boot mode or programmer mode. 0: Programming has ended normally 1: Programming has ended abnormally (programming result is not guaranteed)
4	FK	—	R/W	Flash Key Register Error Detect Returns the check result of the value of FKEY before the start of the programming processing. 0: FKEY setting is normal (FKEY = H'5A) 1: FKEY setting is error (FKEY = value other than H′5A)
3	—	—	—	Unused Returns 0.

Bit	Bit Name	Initial Value	R/W	Description
2	WD	—	R/W	Write Data Address Detect When the address in the flash memory area is specified as the start address of the storage destination of the program data, an error occurs. 0: Setting of write data address is normal 1: Setting of write data address is abnormal
1	WA	—	R/W	Write Address Error Detect When the following items are specified as the start address of the programming destination, an error occurs. • When the programming destination address in the area other than flash memory is specified • When the specified address is not in a 128-byte boundary. (The lower eight bits of the address are other than H'00 and H'80.) 0: Setting of programming destination address is normal 1: Setting of programming destination address is abnormal
0	SF	—	R/W	Success/Fail Indicates whether the program processing is ended normally or not. 0: Programming is ended normally (no error) 1: Programming is ended abnormally (error occurs)

(4) Erasure Execution

When flash memory is erased, the erase-block number on the user MAT must be passed to the erasing program which is downloaded. This is set to the FEBS parameter (general register ER0).

One block is specified from the block number 0 to 15.

For details on the erasing processing procedure, see section 19.4.2, User Program Mode.

(a) Flash erase block select parameter (FEBS: general register ER0 of CPU)

This parameter specifies the erase-block number.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	—	—	Unused These bits should be cleared to H¢0.
15	EB15	—	R/W	Erase Block Set the erase-block number in the range from 0 to 15. 0 corresponds to the EB0 block, and 15 corresponds to the EB15 block.
14	EB14	—	R/W	
13	EB13	—	R/W	
12	EB12	—	R/W	
11	EB11	—	R/W	
10	EB10	—	R/W	
9	EB9	—	R/W	
8	EB8	—	R/W	
7	EB7	—	R/W	
6	EB6	—	R/W	
5	EB5	—	R/W	
4	EB4	—	R/W	
3	EB3	—	R/W	
2	EB2	—	R/W	
1	EB1	—	R/W	
0	EB0	—	R/W	

(b) Flash pass/fail parameter (FPFR: general register R0L of CPU)

This parameter returns value of the erasing processing result.

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Return 0.
6	MD	—	R/W	Programming Mode Related Setting Error Detect Returns the check result that a high level signal is input to the FWE pin and the error protection state is not entered. When the low level signal is input to the FWE pin or the error protection state is entered, 1 is written to this bit. The state can be confirmed with the FWE and FLER bits in FCCS. For conditions to enter the error protection state, see section 19.5.3, Error Protection. 0: FWE and FLER settings are normal (FWE = 1, FLER = 0) 1: Programming cannot be performed (FWE = 0 or FLER = 1)
5	EE	—	R/W	Erasure Execution Error Detect 1 is returned to this bit when the user MAT could not be erased or when flash-memory related register settings are partially changed. If this bit is set to 1, there is a high possibility that the user MAT is partially erased. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when erasure is performed. In this case, both the user MAT and user boot MAT are not erased. Erasing of the user boot MAT should be performed in boot mode or programmer mode. 0: Erasure has ended normally 1: Erasure has ended abnormally (erasure result is not guaranteed)
4	FK	—	R/W	Flash Key Register Error Detect Returns the check result of FKEY value before start of the erasing processing. 0: FKEY setting is normal (FKEY = H'5A) 1: FKEY setting is error (FKEY = value other than H'5A)

Bit	Bit Name	Initial Value	R/W	Description
3	EB	—	R/W	Erase Block Select Error Detect Returns the check result whether the specified erase-block number is in the block range of the user MAT. 0: Setting of erase-block number is normal 1: Setting of erase-block number is abnormal
2, 1	—	—	—	Unused Return 0.
0	SF	—	R/W	Success/Fail Indicates whether the erasing processing is ended normally or not. 0: Erasure is ended normally (no error) 1: Erasure is ended abnormally (error occurs)

19.4 On-Board Programming Mode

When the pin is set in on-board programming mode and the reset start is executed, the on-board programming state that can program/erase the on-chip flash memory is entered. On-board programming mode has three operating modes: boot mode, user program mode, and user boot mode.

For details of the pin setting for entering each mode, see table 19.5. For details of the state transition of each mode for flash memory, see figure 19.2.

Table 19.5 Setting On-Board Programming Mode

Mode Setting	FWE	$\overline{\text{MD2}}$	MD1	NMI
Boot mode	1	0	0	1
User program mode	1*	1	1	0/1
User boot mode	1	0	0	0

Note: * Before downloading the programming/erasing programs, the FLSHE bit must be set to 1 to transition to user program mode.

19.4.1 Boot Mode

Boot mode executes programming/erasing user MAT and user boot MAT by means of the control command and program data transmitted from the host using the on-chip SCI. The tool for transmitting the control command and program data must be prepared in the host. The SCI communication mode is set to asynchronous mode. When reset start is executed after this LSI's pin is set in boot mode, the boot program in the microcomputer is initiated. After the SCI bit rate is automatically adjusted, the communication with the host is executed by means of the control command method.

The system configuration diagram in boot mode is shown in figure 19.6. For details on the pin setting in boot mode, see table 19.5. The NMI and other interrupts are ignored in boot mode. However, the NMI and other interrupts should be disabled in the user system.

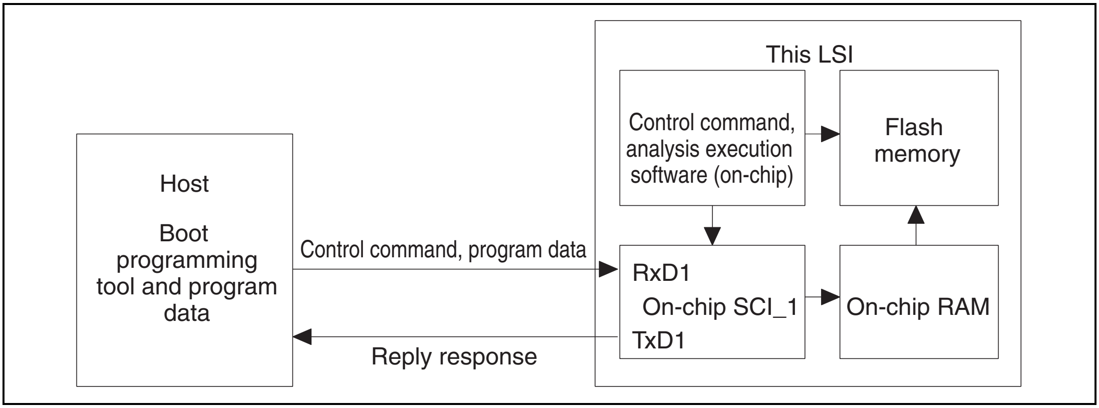

This LSI
Control command, analysis execution software (on-chip)
Flash memory
Host
Boot programming tool and program data
Control command, program data
RxD1
On-chip SCI_1
TxD1
On-chip RAM
Reply response

Figure 19.6 System Configuration in Boot Mode

(1) SCI Interface Setting by Host

When boot mode is initiated, this LSI measures the low period of asynchronous SCI-communication data (H'00), which is transmitted consecutively by the host. The SCI transmit/receive format is set to 8-bit data, 1 stop bit, and no parity. This LSI calculates the bit rate of transmission by the host by means of the measured low period and transmits the bit adjustment end sign (1 byte of H'00) to the host. The host must confirm that this bit adjustment end sign (H'00) has been received normally and transmits 1 byte of H'55 to this LSI. When reception is not executed normally, boot mode is initiated again (reset) and the operation described above must be executed. The bit rate between the host and this LSI is not matched by the bit rate of transmission by the host and system clock frequency of this LSI. To operate the SCI normally, the transfer bit rate of the host must be set to 9,600 bps or 19,200 bps.

The system clock frequency, which can automatically adjust the transfer bit rate of the host and the bit rate of this LSI, is shown in table 19.6. Boot mode must be initiated in the range of this system clock.

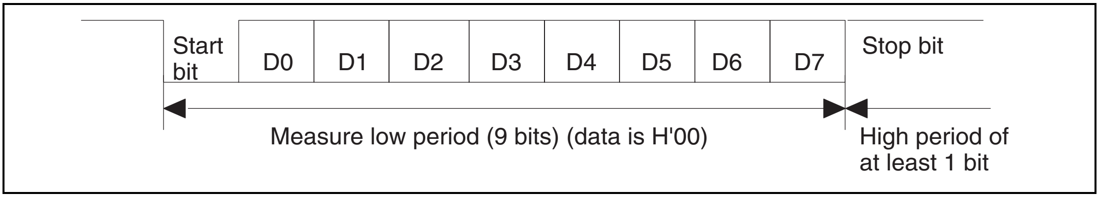

Start bit
D0
D1
D2
D3
D4
D5
D6
D7
Stop bit
Measure low period (9 bits) (data is H'00)
High period of at least 1 bit

Figure 19.7 Automatic-Bit-Rate Adjustment Operation of SCI

Table 19.6 System Clock Frequency for Automatic-Bit-Rate Adjustment by This LSI

Bit Rate of Host	System Clock Frequency
9,600 bps	20 to 25 MHz
19,200 bps	

(2) State Transition Diagram

The overview of the state transition diagram after boot mode is initiated is shown in figure 19.8.

1. Bit rate adjustment

   After boot mode is initiated, the bit rate of the SCI interface is adjusted with that of the host.
2. Waiting for inquiry set command

   For inquiries about user-MAT size and configuration, MAT start address, and support state, the required information is transmitted to the host.
3. Automatic erasure of all user MAT and user boot MAT

   After inquiries have finished, all user MAT and user boot MAT are automatically erased.
4. Waiting for programming/erasing command

   — When the program preparation notice is received, the state for waiting program data is entered. The programming start address and program data must be transmitted following the programming command. When programming is finished, the programming start address must be set to H'FFFFFFFF and transmitted. Then the state for waiting program data is returned to the state of programming/erasing command wait.

   — When the erasure preparation notice is received, the state for waiting erase-block data is entered. The erase-block number must be transmitted following the erasing command. When the erasure is finished, the erase-block number must be set to H'FF and transmitted. Then the state for waiting erase-block data is returned to the state for waiting programming/erasing command. The erasure must be used when the specified block is programmed without a reset start after programming is executed in boot mode. When programming can be executed by only one operation, all blocks are erased before the state for waiting programming/erasing/other command is entered. The erasing operation is not required.

   — There are many commands other than programming/erasing. Examples are sum check, blank check (erasure check), and memory read of the user MAT/user boot MAT and acquisition of current status information.

Note that memory read of the user MAT/user boot MAT can only read the programmed data after all user MAT/user boot MAT has automatically been erased.

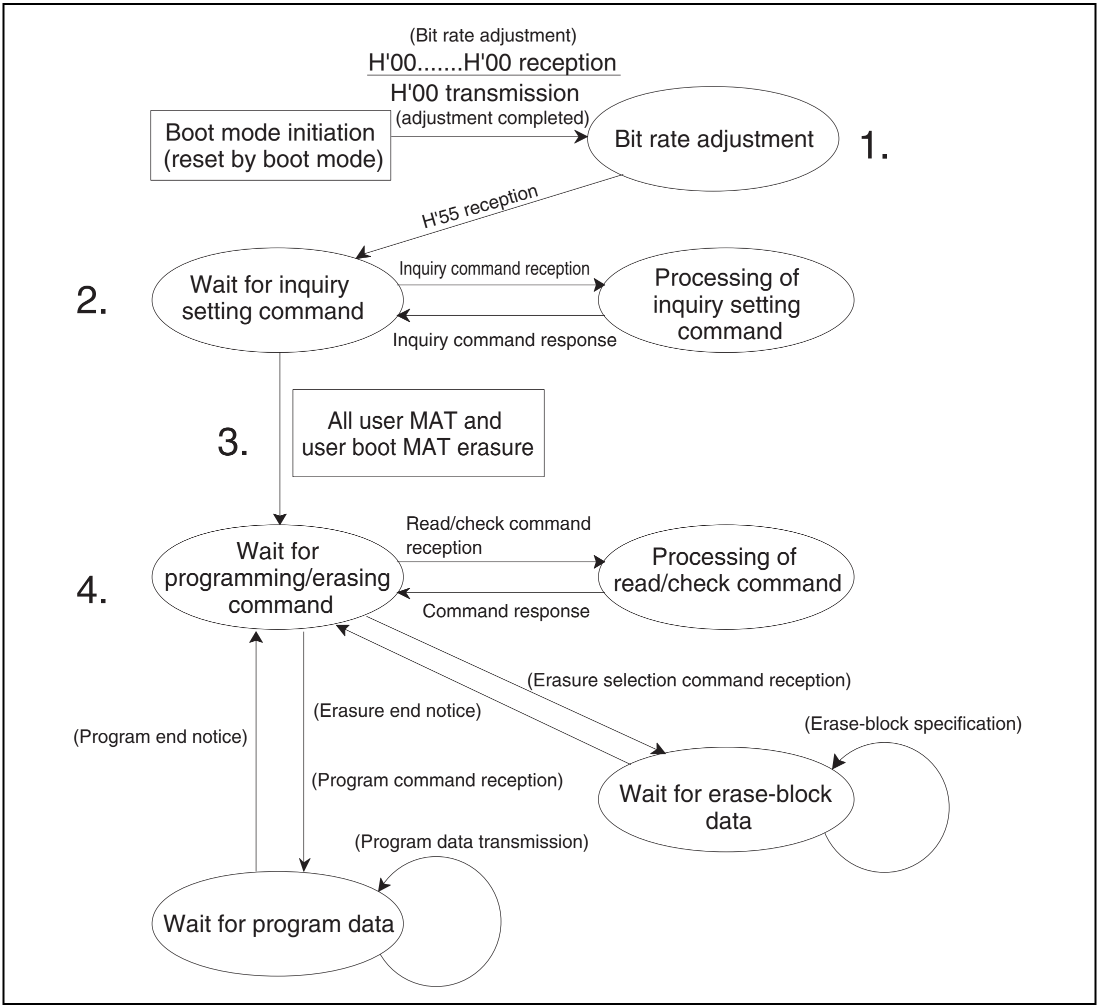

(Bit rate adjustment)
H'00.......H'00 reception
H'00 transmission
(adjustment completed)
Boot mode initiation (reset by boot mode)
Bit rate adjustment
1.
H'55 reception
2.
Wait for inquiry setting command
Inquiry command reception
Processing of inquiry setting command
Inquiry command response
3.
All user MAT and user boot MAT erasure
4.
Wait for programming/erasing command
Read/check command reception
Processing of read/check command
Command response
(Erasure selection command reception)
(Erasure end notice)
(Erase-block specification)
(Program end notice)
(Program command reception)
Wait for erase-block data
(Program data transmission)
Wait for program data

Figure 19.8 Overview of Boot Mode State Transition Diagram

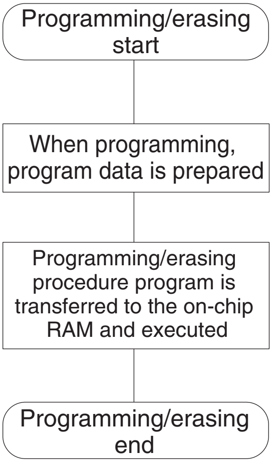
19.4.2 User Program Mode

The user MAT can be programmed/erased in user program mode. (The user boot MAT cannot be programmed/erased.)

Programming/erasing is executed by downloading the program in the microcomputer.

The overview flow is shown in figure 19.9.

High voltage is applied to internal flash memory during the programming/erasing processing. Therefore, transition to reset or hardware standby must not be executed. Doing so may damage or destroy flash memory. If reset is executed accidentally, reset must be released after the reset input period of 100 μs which is longer than normal.


Programming/erasing start
When programming, program data is prepared
Programming/erasing procedure program is transferred to the on-chip RAM and executed
Programming/erasing end

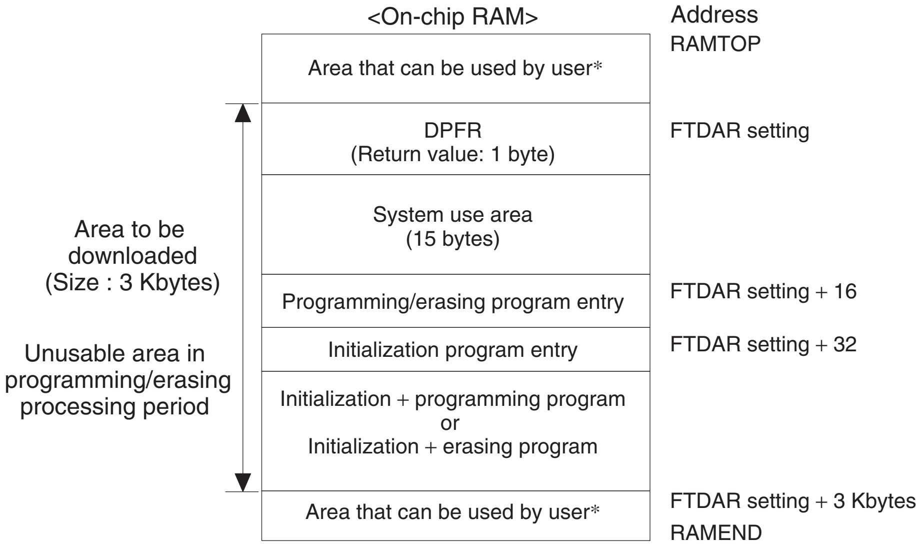
1. Make sure that the program data will not overlap the download destination specified by FTDAR.
2. The FWE bit is set to 1 by inputting a high level signal to the FWE pin.
3. Programming/erasing is executed only in the on-chip RAM. However, if program data is in a consecutive area and can be accessed by the MOV.B instruction of the CPU like RAM or ROM, the program data can be in an external space.
4. After programming/erasing is finished, input a low level signal to the FWE pin and transfer to the hardware protection state.

Figure 19.9 Programming/Erasing Overview Flow

(1) On-chip RAM Address Map when Programming/Erasing is Executed

Parts of the procedure program that are made by the user, like download request, programming/erasing procedure, and determination of the result, must be executed in the on-chip RAM. The on-chip program that is to be downloaded is all in the on-chip RAM. Note that area in the on-chip RAM must be controlled so that these parts do not overlap.

Figure 19.10 shows the program area to be downloaded.


<On-chip RAM>
Address
RAMTOP
Area that can be used by user*
DPFR
(Return value: 1 byte)
FTDAR setting
System use area
(15 bytes)
Area to be downloaded
(Size : 3 Kbytes)
Programming/erasing program entry
FTDAR setting + 16
Unusable area in programming/erasing processing period
Initialization program entry
FTDAR setting + 32
Initialization + programming program
or
Initialization + erasing program
Area that can be used by user*
FTDAR setting + 3 Kbytes
RAMEND

Note: * The on-chip RAM area in this LSI is split into H'FF0800 to H'FF97FF, H'FFE080 to H'FFEFFF, and H'FFFF00 to H'FFFF7F.
The area that can be used by the user is specified by FTDAR.

Figure 19.10 RAM Map When Programming/Erasing is Executed

(2) Programming Procedure in User Program Mode

The procedures for download, initialization, and programming are shown in figure 19.11.

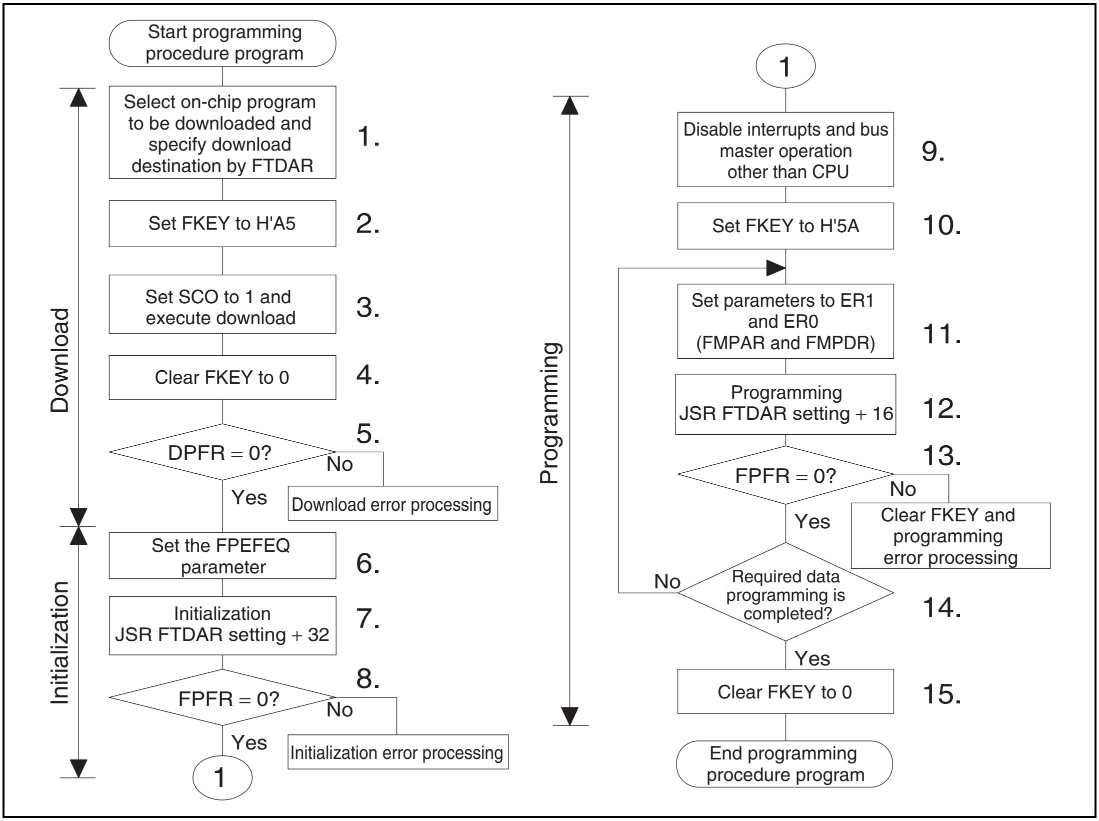

Start programming procedure program
Select on-chip program to be downloaded and specify download destination by FTDAR
1.
Set FKEY to H'A5
2.
Set SCO to 1 and execute download
3.
Clear FKEY to 0
4.
DPFR = 0?
5.
No
Yes
Download error processing
Download
Set the FPEFEQ parameter
6.
Initialization JSR FTDAR setting + 32
7.
FPFR = 0?
8.
No
Yes
Initialization error processing
Initialization
1
1
Disable interrupts and bus master operation other than CPU
9.
Set FKEY to H'5A
10.
Set parameters to ER1 and ER0 (FMPAR and FMPDR)
11.
Programming JSR FTDAR setting + 16
12.
FPFR = 0?
13.
No
Yes
Clear FKEY and programming error processing
No
Required data programming is completed?
14.
Yes
Clear FKEY to 0
15.
End programming procedure program
Programming

Figure 19.11 Programming Procedure

The procedure program must be executed in an area other than the flash memory to be programmed. Especially the part where the SCO bit in FCCS is set to 1 for downloading must be executed in the on-chip RAM.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 19.4.4, Procedure Program and Storable Area for Programming Data.

The following description assumes the area to be programmed on the user MAT is erased and program data is prepared in the consecutive area. When erasing is not executed, erasing is executed before writing.

128-byte programming is performed in one program processing. When more than 128-byte programming is performed, programming destination address/program data parameter is updated in 128-byte units and programming is repeated.

When less than 128-byte programming is performed, data must total 128 bytes by adding the invalid data. If the dummy data to be added is H'FF, the program processing period can be shortened.

1. Select the on-chip program to be downloaded and specify a download destination

   When the PPVS bit of FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are set, download is not performed and a download error is returned to the SS bit in DPFR. The start address of a download destination is specified by FTDAR.

2. Program H'A5 in FKEY

   If H'A5 is not written to FKEY for protection, 1 cannot be set to the SCO bit for download request.

3. 1 is set to the SCO bit of FCCS and then download is executed.

   To set 1 to the SCO bit, the following conditions must be satisfied.

   — H'A5 is written to FKEY.

   — The SCO bit writing is executed in the on-chip RAM.

   When the SCO bit is set to 1, download is started automatically. When the SCO bit is returned to the user procedure program, the SCO is cleared to 0. Therefore, the SCO bit cannot be confirmed to be 1 in the user procedure program.

   The download result can be confirmed only by the return value of DPFR. Before the SCO bit is set to 1, incorrect determination must be prevented by setting the one byte of the start address (to be used as DPFR) specified by FTDAR to a value other than the return value (e.g. H'FF).

   When download is executed, particular interrupt processing, which is accompanied by the bank switch as described below, is performed as an internal microcomputer processing. Four NOP instructions are executed immediately after the instructions that set the SCO bit to 1.

   — The user-MAT space is switched to the on-chip program storage area.

   — After the selection condition of the download program and the FTDAR setting are checked, the transfer processing to the on-chip RAM specified by FTDAR is executed.

   — The SCO bit in FCCS is cleared to 0.

   — The return value is set to the DPFR parameter.

- After the on-chip program storage area is returned to the user-MAT space, the user procedure program is returned.
- In the download processing, the values of general registers of the CPU are held.
- In the download processing, any interrupts are not accepted. However, interrupt requests are held. Therefore, when the user procedure program is returned, the interrupts occur.
- When the level-detection interrupt requests are to be held, interrupts must be input until the download is ended.
- When hardware standby mode is entered during download processing, the normal download cannot be guaranteed in the on-chip RAM. Therefore, download must be executed again.
- Since a stack area of 128 bytes at the maximum is used, the area must be allocated before setting the SCO bit to 1.
- If a flash memory access by the DTC signal is requested during downloading, the operation cannot be guaranteed. Therefore, an access request by the DTC signal must not be generated.

4. FKEY is cleared to H'00 for protection.

5. The value of the DPFR parameter must be checked and the download result must be confirmed.

   - Check the value of the DPFR parameter (one byte of start address of the download destination specified by FTDAR). If the value is H'00, download has been performed normally. If the value is not H'00, the source that caused download to fail can be investigated by the description below.
   - If the value of the DPFR parameter is the same as before downloading (e.g. H'FF), the address setting of the download destination in FTDAR may be abnormal. In this case, confirm the setting of the TDER bit (bit 7) in FTDAR.
   - If the value of the DPFR parameter is different from before downloading, check the SS bit (bit 2) and the FK bit (bit 1) in the DPFR parameter to ensure that the download program selection and FKEY setting were normal, respectively.

6. The operating frequency is set in the FPEFEQ parameter for initialization.

   - The current frequency of the CPU clock is set to the FPEFEQ parameter value (general register ER0).

     The settable range of the FPEFEQ parameter is 20 to 25 MHz. When the frequency is set to out of this range, an error is returned to the FPFR parameter of the initialization program and initialization is not performed. For details on the frequency setting, see the description in 19.3.2 (2) (a), Flash programming/erasing frequency parameter (FPEFEQ).

7. Initialization

   When a programming program is downloaded, the initialization program is also downloaded to the on-chip RAM. There is an entry point of the initialization program in the area from the start address specified by FTDAR + 32 bytes of the on-chip RAM. The subroutine is called and initialization is executed by using the following steps.

```
MOV.L   #DLTOP+32,ER2        ; Set entry address to ER2
JSR     @ER2                 ; Call initialization routine
NOP
```

   — The general registers other than R0L are held in the initialization program.

   — R0L is a return value of the FPFR parameter.

   — Since the stack area is used in the initialization program, 128-byte stack area at the maximum must be allocated in RAM.

   — Interrupts can be accepted during the execution of the initialization program. The program storage area and stack area in the on-chip RAM and register values must not be destroyed.

8. The return value in the initialization program, FPFR (general register R0L) is determined.

9. All interrupts and the use of a bus master other than the CPU are prohibited.
   The specified voltage is applied for the specified time when programming or erasing. If interrupts occur or the bus mastership is moved to other than the CPU during this time, the voltage for more than the specified time will be applied and flash memory may be damaged. Therefore, interrupts and bus mastership to other than the CPU, such as to the DTC, are prohibited.

   To disable interrupts, bit 7 (I) in the condition code register (CCR) of the CPU should be set to B'1 in interrupt control mode 0 or bits 7 and 6 (I and UI) should be set to B'11 in interrupt control mode 1. Interrupts other than NMI are held and not executed.

   The NMI interrupts must be masked within the user system.

   The interrupts that are held must be executed after all program processing.

   When the bus mastership is moved to other than the CPU, such as to the DTC, the error protection state is entered. Therefore, taking bus mastership by the DTC is prohibited.

10. FKEY must be set to H'5A and the user MAT must be prepared for programming.

11. The parameter which is required for programming is set.

    The start address of the programming destination of the user MAT (FMPAR) is set to general register ER1. The start address of the program data area (FMPDR) is set to general register ER0.

    — Example of the FMPAR setting

      FMPAR specifies the programming destination address. When an address other than one in the user MAT area is specified, even if the programming program is executed, programming is not executed and an error is returned to the return value parameter FPFR. Since the unit is 128 bytes, the lower eight bits of the address must be H'00 or H'80 as the boundary of 128 bytes.

    — Example of the FMPDR setting

      When the storage destination of the program data is flash memory, even if the program execution routine is executed, programming is not executed and an error is returned to the FPFR parameter. In this case, the program data must be transferred to the on-chip RAM and then programming must be executed.

12. Programming

    There is an entry point of the programming program in the area from the start address specified by FTDAR + 16 bytes of the on-chip RAM. The subroutine is called and programming is executed by using the following steps.

```
MOV.L    #DLTOP+16,ER2          ; Set entry address to ER2
JSR      @ER2                   ; Call programming routine
NOP
```

    — The general registers other than R0L are held in the programming program.

    — R0L is a return value of the FPFR parameter.

    — Since the stack area is used in the programming program, a stack area of 128 bytes at the maximum must be allocated in RAM.

13. The return value in the programming program, FPFR (general register R0L) is determined.

14. Determine whether programming of the necessary data has finished.

    If more than 128 bytes of data are to be programmed, specify FMPAR and FMPDR in 128-byte units, and repeat steps 12 to 14. Increment the programming destination address by 128 bytes and update the programming data pointer correctly. If an address which has already been programmed is written to again, not only will a programming error occur, but also flash memory will be damaged.

15. After programming finishes, clear FKEY and specify software protection.
    If this LSI is restarted by a reset immediately after user MAT programming has finished, secure the reset period (period of $\overline{\text{RES}}$ = 0) of 100 µs which is longer than normal.

(3) Erasing Procedure in User Program Mode

The procedures for download, initialization, and erasing are shown in figure 19.12.

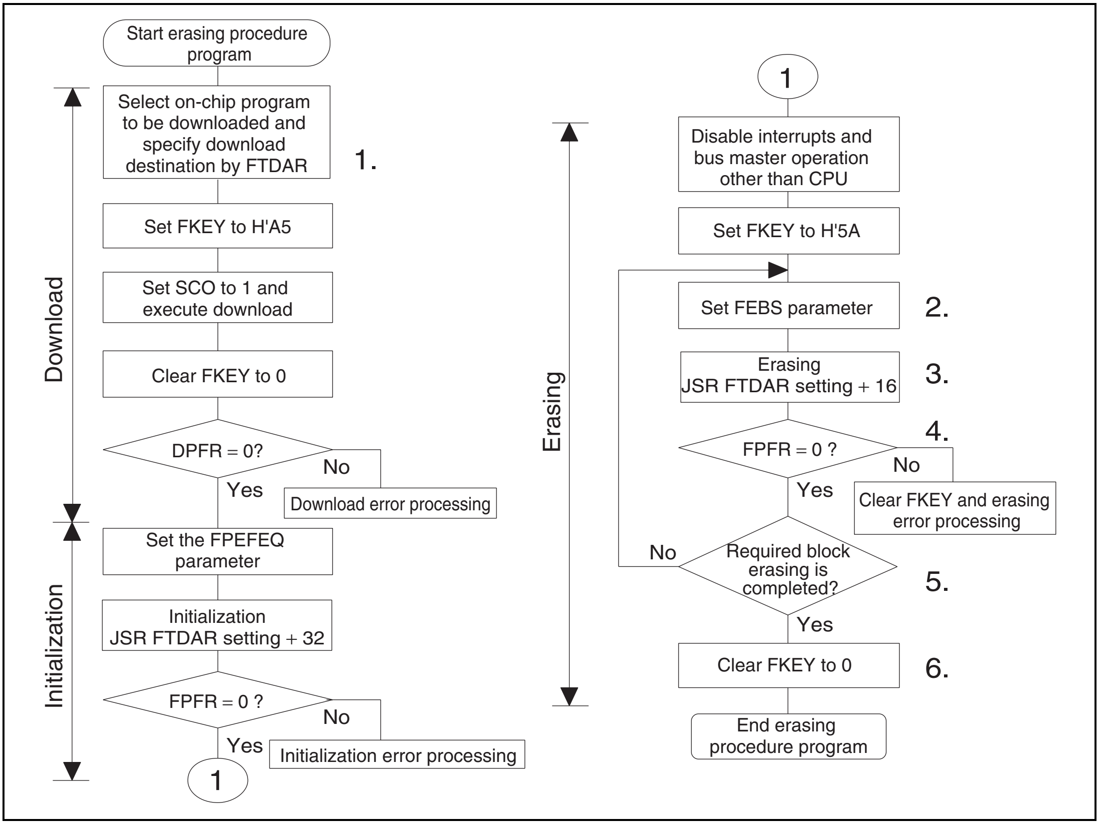

Start erasing procedure program
Download
Select on-chip program to be downloaded and specify download destination by FTDAR
1.
Set FKEY to H'A5
Set SCO to 1 and execute download
Clear FKEY to 0
DPFR = 0?
No
Yes
Download error processing
Initialization
Set the FPEFEQ parameter
Initialization JSR FTDAR setting + 32
FPFR = 0 ?
No
Yes
Initialization error processing
1
Erasing
1
Disable interrupts and bus master operation other than CPU
Set FKEY to H'5A
Set FEBS parameter
2.
Erasing JSR FTDAR setting + 16
3.
FPFR = 0 ?
4.
No
Yes
Clear FKEY and erasing error processing
No
Required block erasing is completed?
5.
Yes
Clear FKEY to 0
6.
End erasing procedure program

Figure 19.12 Erasing Procedure

The procedure program must be executed in an area other than the user MAT to be erased. Especially the part where the SCO bit in FCCS is set to 1 for downloading must be executed in the on-chip RAM.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 19.4.4, Procedure Program and Storable Area for Programming Data.

For the downloaded on-chip program area, refer to the RAM map for programming/erasing in figure 19.10.

A single divided block is erased by one erasing processing. For block divisions, refer to figure 19.4. To erase two or more blocks, update the erase block number and perform the erasing processing for each block.

1. Select the on-chip program to be downloaded

   Set the EPVB bit in FECS to 1.

   Several programming/erasing programs cannot be selected at one time. If several programs are set, download is not performed and a download error is reported to the SS bit in the DPFR parameter.

   Specify the start address of a download destination by FTDAR.

   The procedures to be carried out after setting FKEY, e.g. download and initialization, are the same as those in the programming procedure. For details, refer to section 19.4.2 (2), Programming Procedure in User Program Mode.

   The procedures after setting parameters for erasing programs are as follows:

2. Set the FEBS parameter necessary for erasure

   Set the erase block number of the user MAT in the flash erase block select parameter FEBS (general register ER0). If a value other than an erase block number of the user MAT is set, no block is erased even though the erasing program is executed, and an error is returned to the return value parameter FPFR.

3. Erasure

   Similar to as in programming, there is an entry point of the erasing program in the area from the start address of a download destination specified by FTDAR + 16 bytes of on-chip RAM. The subroutine is called and erasing is executed by using the following steps.

```
MOV.L   #DLTOP+16,ER2        ; Set entry address to ER2
JSR     @ER2                 ; Call erasing routine
NOP
```

   - The general registers other than R0L are held in the erasing program.
   - R0L is a return value of the FPFR parameter.
   - Since the stack area is used in the erasing program, a stack area of 128 bytes at the maximum must be allocated in RAM.

4. The return value in the erasing program, FPFR (general register R0L) is determined.

5. Determine whether erasure of the necessary blocks has completed.

   If more than one block is to be erased, update the FEBS parameter and repeat steps 2 to 5. Blocks that have already been erased can be erased again.

6. After erasure completes, clear FKEY and specify software protection.

   If this LSI is restarted by a reset immediately after user MAT erasure has completed, secure the reset period (period of $\overline{\text{RES}}$ = 0) of 100 µs which is longer than normal.

(4) Erasing and Programming Procedure in User Program Mode

By changing the on-chip RAM address of the download destination in FTDAR, the erasing program and programming program can be downloaded to separate on-chip RAM areas.

Figure 19.13 shows a repeating procedure of erasing and programming.

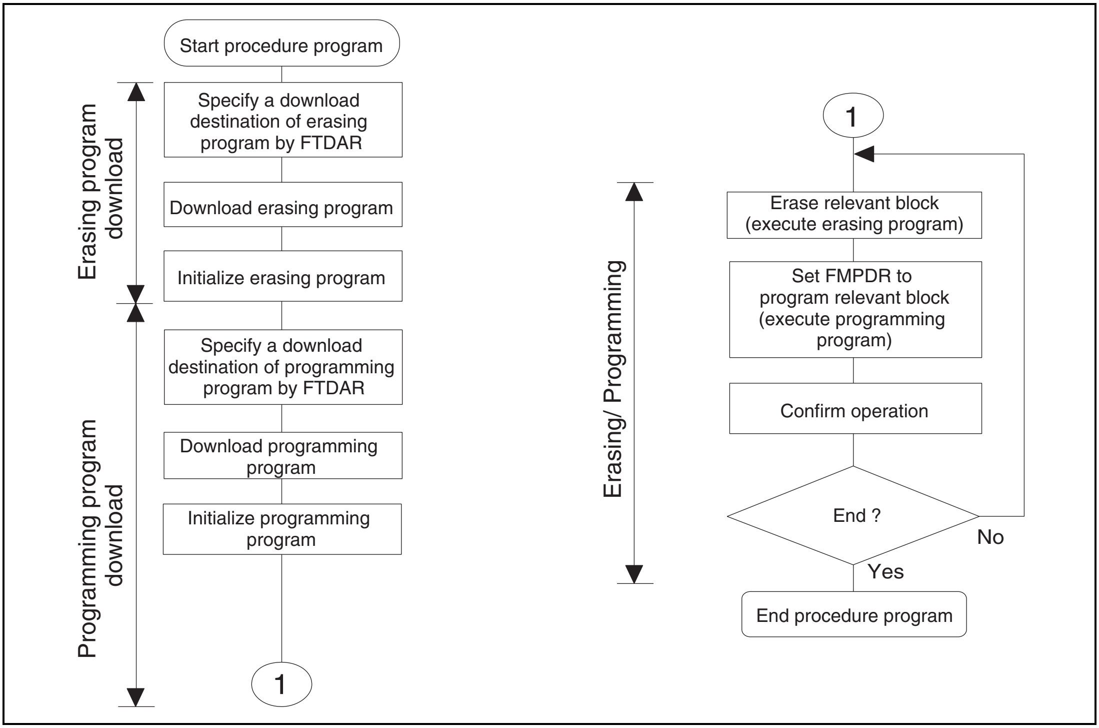

Start procedure program
Erasing program download
Specify a download destination of erasing program by FTDAR
Download erasing program
Initialize erasing program
Programming program download
Specify a download destination of programming program by FTDAR
Download programming program
Initialize programming program
1
1
Erasing/ Programming
Erase relevant block (execute erasing program)
Set FMPDR to program relevant block (execute programming program)
Confirm operation
End ?
No
Yes
End procedure program

Figure 19.13 Repeating Procedure of Erasing and Programming

In the above procedure, download and initialization are performed only once at the beginning.

In this kind of operation, note the following:

- Be careful not to damage on-chip RAM with overlapped settings.

  In addition to the erasing program area and programming program area, areas for the user procedure programs, work area, and stack area are reserved in on-chip RAM. Do not make settings that will overwrite data in these areas.

- Be sure to initialize both the erasing program and programming program.

  Initialization by setting the FPEFEQ parameter must be performed for both the erasing program and the programming program. Initialization must be executed for both entry addresses: (download start address for erasing program) + 32 bytes and (download start address for programming program) + 32 bytes.

19.4.3 User Boot Mode

This LSI has user boot mode which is initiated with different mode pin settings than those in boot mode or user program mode. User boot mode is a user-arbitrary boot mode, unlike boot mode that uses the on-chip SCI.

Only the user MAT can be programmed/erased in user boot mode. Programming/erasing of the user boot MAT is only enabled in boot mode or programmer mode.

(1) User Boot Mode Initiation

For the mode pin settings to start up user boot mode, see table 19.5.

When the reset start is executed in user boot mode, the built-in check routine runs. The user MAT and user boot MAT states are checked by this check routine.

While the check routine is running, NMI and all other interrupts cannot be accepted.

Next, processing starts from the execution start address of the reset vector in the user boot MAT. At this point, H'AA is set to FMATS because the execution MAT is the user boot MAT.

(2) User MAT Programming in User Boot Mode

For programming the user MAT in user boot mode, additional processing made by setting FMATS are required: switching from user-boot-MAT selection state to user-MAT selection state, and switching back to user-boot-MAT selection state after programming completes.

Figure 19.14 shows the procedure for programming the user MAT in user boot mode.

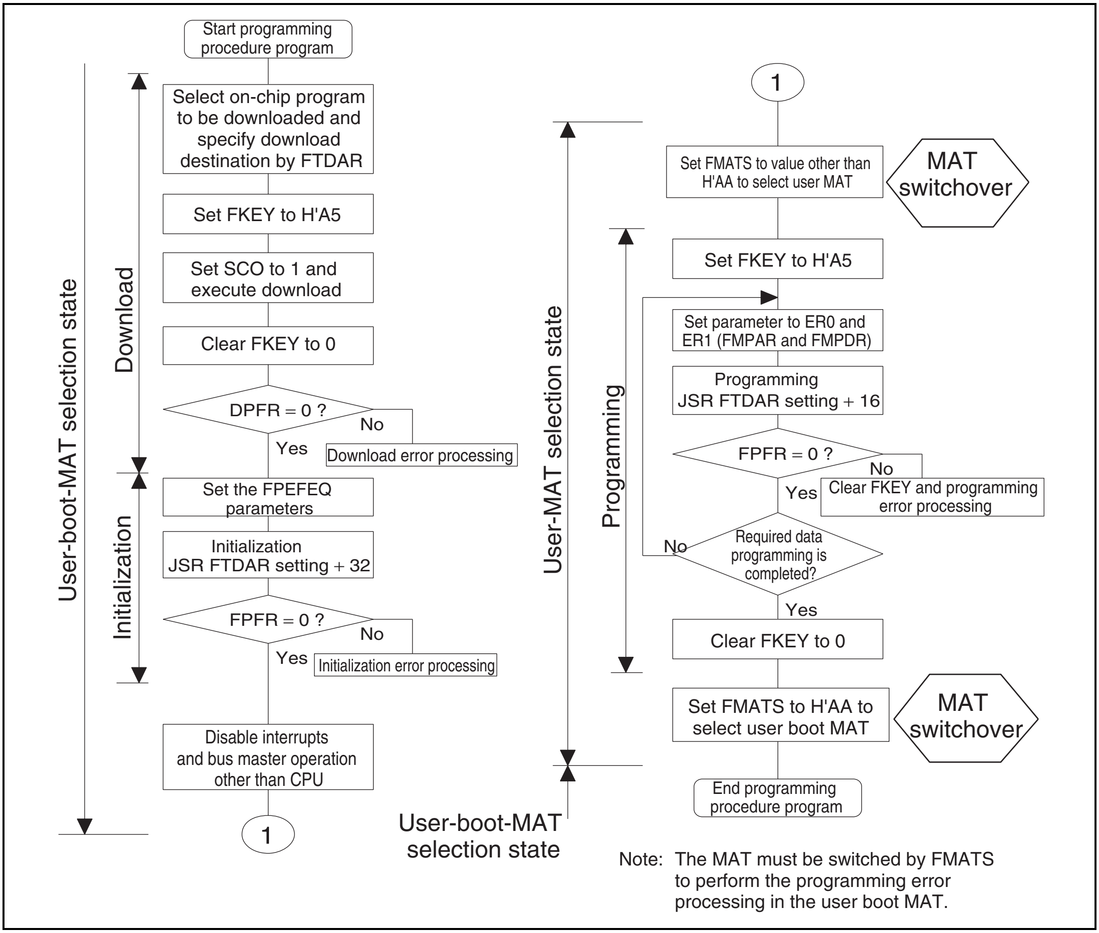

Start programming procedure program
Select on-chip program to be downloaded and specify download destination by FTDAR
Set FKEY to H'A5
Set SCO to 1 and execute download
Clear FKEY to 0
DPFR = 0 ?
No
Yes
Download error processing
Set the FPEFEQ parameters
Initialization JSR FTDAR setting + 32
FPFR = 0 ?
No
Yes
Initialization error processing
Disable interrupts and bus master operation other than CPU
1
User-boot-MAT selection state
Download
Initialization
1
Set FMATS to value other than H'AA to select user MAT
MAT switchover
Set FKEY to H'A5
Set parameter to ER0 and ER1 (FMPAR and FMPDR)
Programming JSR FTDAR setting + 16
FPFR = 0 ?
No
Yes
Clear FKEY and programming error processing
Required data programming is completed?
No
Yes
Clear FKEY to 0
Set FMATS to H'AA to select user boot MAT
MAT switchover
End programming procedure program
User-MAT selection state
Programming
User-boot-MAT selection state
Note: The MAT must be switched by FMATS to perform the programming error processing in the user boot MAT.

Figure 19.14 Procedure for Programming User MAT in User Boot Mode

The difference between the programming procedures in user program mode and user boot mode is whether the MAT is switched or not as shown in figure 19.14.

In user boot mode, the user boot MAT can be seen in the flash memory space with the user MAT hidden in the background. The user MAT and user boot MAT are switched only while the user MAT is being programmed. Because the user boot MAT is hidden while the user MAT is being programmed, the procedure program must be located in an area other than flash memory. After programming completes, switch the MATs again to return to the first state.

MAT switching is enabled by writing a specific value to FMATS. However note that while the MATs are being switched, the LSI is in an unstable state, e.g. access to a MAT is not allowed until MAT switching is completed, and if an interrupt occurs, from which MAT the interrupt vector is read is undetermined. Perform MAT switching in accordance with the description in section 19.6, Switching between User MAT and User Boot MAT.

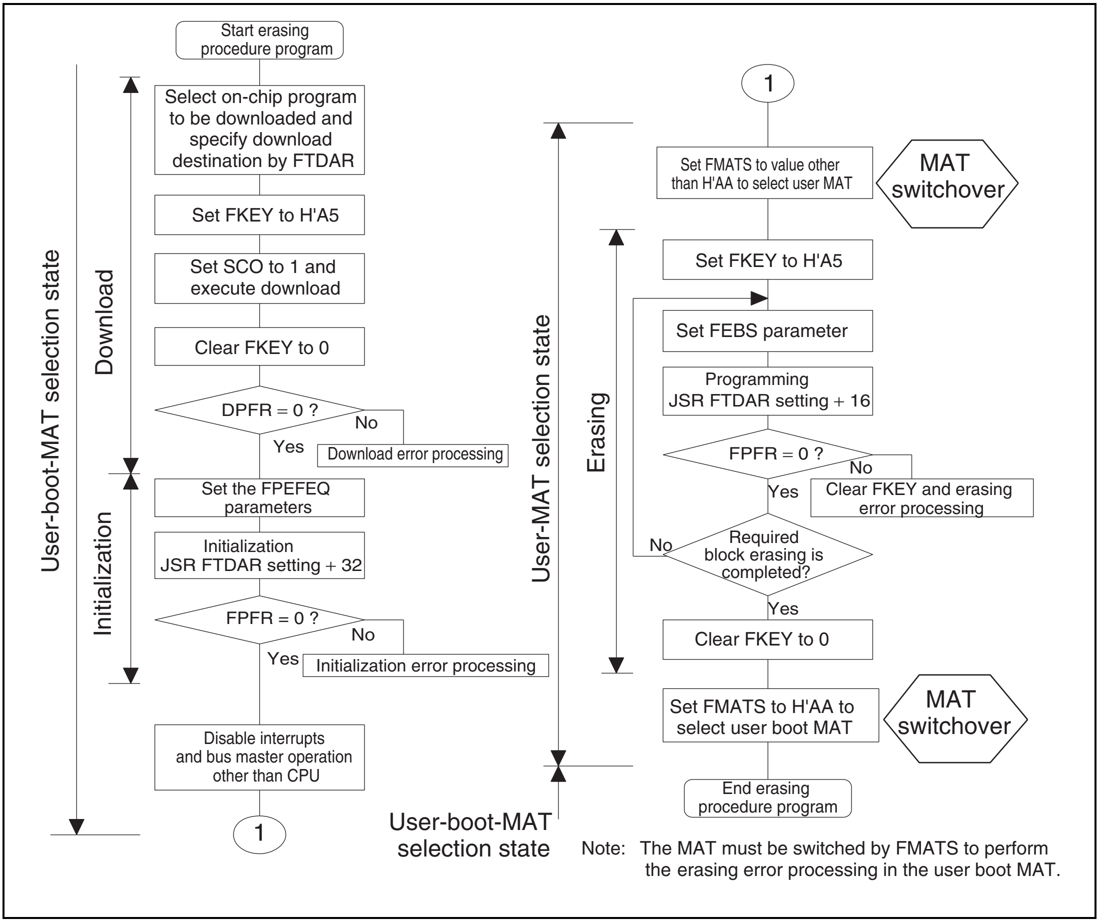
Except for MAT switching, the programming procedure is the same as that in user program mode.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 19.4.4, Procedure Program and Storable Area for Programming Data.

(3) User MAT Erasing in User Boot Mode

For erasing the user MAT in user boot mode, additional processing made by setting FMATS are required: switching from user-boot-MAT selection state to user-MAT selection state, and switching back to user-boot-MAT selection state after erasing completes.

Figure 19.15 shows the procedure for erasing the user MAT in user boot mode.


Start erasing procedure program
Select on-chip program to be downloaded and specify download destination by FTDAR
Set FKEY to H'A5
Set SCO to 1 and execute download
Clear FKEY to 0
DPFR = 0 ?
No
Yes
Download error processing
Set the FPEFEQ parameters
Initialization JSR FTDAR setting + 32
FPFR = 0 ?
No
Yes
Initialization error processing
Disable interrupts and bus master operation other than CPU
1
User-boot-MAT selection state
Download
Initialization
1
Set FMATS to value other than H'AA to select user MAT
MAT switchover
Set FKEY to H'A5
Set FEBS parameter
Programming JSR FTDAR setting + 16
FPFR = 0 ?
No
Yes
Clear FKEY and erasing error processing
Required block erasing is completed?
No
Yes
Clear FKEY to 0
Set FMATS to H'AA to select user boot MAT
MAT switchover
End erasing procedure program
User-MAT selection state
Erasing
User-boot-MAT selection state

Note: The MAT must be switched by FMATS to perform the erasing error processing in the user boot MAT.

Figure 19.15 Procedure for Erasing User MAT in User Boot Mode

The difference between the erasing procedures in user program mode and user boot mode depends on whether the MAT is switched or not as shown in figure 19.15.

MAT switching is enabled by writing a specific value to FMATS. However note that while the MATs are being switched, the LSI is in an unstable state, e.g. access to a MAT is not allowed until MAT switching is completed, and if an interrupt occurs, from which MAT the interrupt vector is read is undetermined. Perform MAT switching in accordance with the description in section 19.6, Switching between User MAT and User Boot MAT.

Except for MAT switching, the erasing procedure is the same as that in user program mode.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 19.4.4, Procedure Program and Storable Area for Programming Data.

19.4.4 Procedure Program and Storable Area for Programming Data

In the descriptions in the previous section, the programming/erasing procedure programs and storable areas for program data are assumed to be in the on-chip RAM. However, the program and the data can be stored in and executed from other areas, such as part of flash memory which is not to be programmed or erased, or somewhere in the external address space.

(1) Conditions that Apply to Programming/Erasing

1. The on-chip programming/erasing program is downloaded from the address in the on-chip RAM specified by FTDAR, therefore, this area is not available for use.
2. The on-chip programming/erasing program will use 128 bytes at the maximum as a stack. So, make sure that this area is secured.
3. Download by setting the SCO bit to 1 will lead to switching of the MAT. If, therefore, this operation is used, it should be executed from the on-chip RAM.
4. The flash memory is accessible until the start of programming or erasing, that is, until the result of downloading has been determined. When in a mode in which the external address space is not accessible, such as single-chip mode, the required procedure programs, NMI handling vector and NMI handler should be transferred to the on-chip RAM before programming/erasing of the flash memory starts.
5. The flash memory is not accessible during programming/erasing operations, therefore, the operation program is downloaded to the on-chip RAM to be executed. The NMI-handling vector and programs such as that which activate the operation program, and NMI handler should thus be stored in on-chip memory other than flash memory or the external address space.
6. After programming/erasing, the flash memory should be inhibited until FKEY is cleared.

   The reset state ($\overline{\text{RES}}$ = 0) must be in place for more than 100 μs when the LSI mode is changed to reset on completion of a programming/erasing operation.

   Transitions to the reset state, and hardware standby mode are inhibited during programming/erasing. When the reset signal is accidentally input to the chip, a longer period in the reset state than usual (100 μs) is needed before the reset signal is released.
7. Switching of the MATs by FMATS should be needed when programming/erasing of the user boot MAT is operated in user-boot mode. The program which switches the MATs should be executed from the on-chip RAM. See section 19.6, Switching between User MAT and User Boot MAT. Please make sure you know which MAT is selected when switching between them.
8. When the data storable area indicated by programming parameter FMPDR is within the flash memory area, an error will occur even when the data stored is normal. Therefore, the data

should be transferred to the on-chip RAM to place the address that FMPDR indicates in an area other than the flash memory.

In consideration of these conditions, there are three factors; operating mode, the bank structure of the user MAT, and operations.

The areas in which the programming data can be stored for execution are shown in tables.

Table 19.7 Executable MAT

	Initiated Mode	
Operation	**User Program Mode**	**User Boot Mode***
Programming	Table 19.8 (1)	Table 19.8 (3)
Erasing	Table 19.8 (2)	Table 19.8 (4)

Note: * Programming/Erasing is possible to user MATs.

Table 19.8 (1) Useable Area for Programming in User Program Mode

	Storable /Executable Area		Selected MAT	
Item	On-chip RAM	User MAT	User MAT	Embedded Program Storage Area
Storage Area for Program Data	○	×*	—	—
Operation for Selection of On-chip Program to be Downloaded	○	○	○	
Operation for Writing H'A5 to FKEY	○	○	○	
Execution of Writing SCO = 1 to FCCS (Download)	○	×		○
Operation for FKEY Clear	○	○	○	
Determination of Download Result	○	○	○	
Operation for Download Error	○	○	○	
Operation for Settings of Initial Parameter	○	○	○	
Execution of Initialization	○	×	○	
Determination of Initialization Result	○	○	○	
Operation for Initialization Error	○	○	○	
NMI Handling Routine	○	×	○	
Operation for Inhibit of Interrupt	○	○	○	
Operation for Writing H'5A to FKEY	○	○	○	
Operation for Settings of Program Parameter	○	×	○	

	Storable /Executable Area		Selected MAT	
Item	**On-chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage Area**
Execution of Programming	○	×	○	
Determination of Program Result	○	×	○	
Operation for Program Error	○	×	○	
Operation for FKEY Clear	○	×	○	

Note: * Transferring the data to the on-chip RAM enables this area to be used.

Table 19.8 (2) Useable Area for Erasure in User Program Mode

	Storable /Executable Area		Selected MAT	
Item	**On-chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage Area**
Operation for Selection of On-chip Program to be Downloaded	○	○	○	
Operation for Writing H'A5 to FKEY	○	○	○	
Execution of Writing SCO = 1 to FCCS (Download)	○	×		○
Operation for FKEY Clear	○	○	○	
Determination of Download Result	○	○	○	
Operation for Download Error	○	○	○	
Operation for Settings of Initial Parameter	○	○	○	
Execution of Initialization	○	×	○	
Determination of Initialization Result	○	○	○	
Operation for Initialization Error	○	○	○	
NMI Handling Routine	○	×	○	
Operation for Inhibit of Interrupt	○	○	○	
Operation for Writing H'5A to FKEY	○	○	○	
Operation for Settings of Erasure Parameter	○	×	○	
Execution of Erasure	○	×	○	
Determination of Erasure Result	○	×	○	

	Storable /Executable Area		Selected MAT	
Item	On-chip RAM	User MAT	User MAT	Embedded Program Storage Area
Operation for Erasure Error	○	×	○	
Operation for FKEY Clear	○	×	○	

Table 19.8 (3) Useable Area for Programming in User Boot Mode

	Storable/Executable Area		Selected MAT		
Item	On-chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage Area
Storage Area for Program Data	○	×*1	—	—	—
Operation for Selection of On-chip Program to be Downloaded	○	○		○	
Operation for Writing H'A5 to FKEY	○	○		○	
Execution of Writing SCO = 1 to FCCS (Download)	○	×			○
Operation for FKEY Clear	○	○		○	
Determination of Download Result	○	○		○	
Operation for Download Error	○	○		○	
Operation for Settings of Initial Parameter	○	○		○	
Execution of Initialization	○	×		○	
Determination of Initialization Result	○	○		○	
Operation for Initialization Error	○	○		○	
NMI Handling Routine	○	×		○	
Operation for Interrupt Inhibit	○	○		○	
Switching MATs by FMATS	○	×	○		
Operation for Writing H'5A to FKEY	○	×	○		

	Storable/Executable Area		**Selected MAT**		
Item	**On-chip RAM**	**User Boot MAT**	**User MAT**	**User Boot MAT**	**Embedded Program Storage Area**
Operation for Settings of Program Parameter	○	×	○		
Execution of Programming	○	×	○		
Determination of Program Result	○	×	○		
Operation for Program Error	○	×*[2]	○		
Operation for FKEY Clear	○	×	○		
Switching MATs by FMATS	○	×		○	

Notes: 1. Transferring the data to the on-chip RAM enables this area to be used.

2. Switching FMATS by a program in the on-chip RAM enables this area to be used.

Table 19.8 (4) Useable Area for Erasure in User Boot Mode

	Storable/Executable Area		Selected MAT		
Item	On-chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage Area
Operation for Selection of On-chip Program to be Downloaded	○	○		○	
Operation for Writing H'A5 to FKEY	○	○		○	
Execution of Writing SCO = 1 to FCCS (Download)	○	×			○
Operation for FKEY Clear	○	○		○	
Determination of Download Result	○	○		○	
Operation for Download Error	○	○		○	
Operation for Settings of Initial Parameter	○	○		○	
Execution of Initialization	○	×		○	
Determination of Initialization Result	○	○		○	
Operation for Initialization Error	○	○		○	
NMI Handling Routine	○	×		○	
Operation for Interrupt Inhibit	○	○		○	
Switching MATs by FMATS	○	×		○	
Operation for Writing H'5A to FKEY	○	×	○		

	Storable/Executable Area		Selected MAT		
Item	**On-chip RAM**	**User Boot MAT**	**User MAT**	**User Boot MAT**	**Embedded Program Storage Area**
Operation for Settings of Erasure Parameter	○	×	○		
Execution of Erasure	○	×	○		
Determination of Erasure Result	○	×	○		
Operation for Erasure Error	○	×*	○		
Operation for FKEY Clear	○	×	○		
Switching MATs by FMATS	○	×	○		

Note: * Switching FMATS by a program in the on-chip RAM enables this area to be used.

19.5 Protection

There are three kinds of flash memory program/erase protection: hardware, software, and error protection.

19.5.1 Hardware Protection

Programming and erasing of flash memory is forcibly disabled or suspended by hardware protection. In this state, the downloading of an on-chip program and initialization are possible. However, an activated program for programming or erasure cannot program or erase locations in a user MAT, and the error in programming/erasing is reported in the parameter FPFR.

Table 19.9 Hardware Protection

		Function to be Protected	
Item	**Description**	**Download**	**Program/Erase**
FWE pin protection	• When a low level signal is input to the FWE pin, the FWE bit in FCCS is cleared and the program/erase-protected state is entered.	—	○
Reset/standby protection	• The program/erase interface registers are initialized in the reset state (including a reset by the WDT) and standby mode and the program/erase-protected state is entered. • The reset state will not be entered by a reset using the $\overline{\text{RES}}$ pin unless the $\overline{\text{RES}}$ pin is held low until oscillation has stabilized after power is initially supplied. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the RES pulse width that is specified in the section on AC characteristics section. If a reset is input during programming or erasure, data values in the flash memory are not guaranteed. In this case, execute erasure and then execute program again.	○	○

19.5.2 Software Protection

Software protection is set up in any of two ways: by disabling the downloading of on-chip programs for programming and erasing and by means of a key code.

Table 19.10 Software Protection

		Function to be Protected	
Item	**Description**	**Download**	**Program/Erase**
Protection by the SCO bit	• The program/erase-protected state is entered by clearing the SCO bit in FCCS which disables the downloading of the programming/erasing programs.	○	○
Protection by the FKEY register	• Downloading and programming/erasing are disabled unless the required key code is written in FKEY. Different key codes are used for downloading and for programming/erasing.	○	○

19.5.3 Error Protection

Error protection is a mechanism for aborting programming or erasure when an error occurs, in the form of the microcomputer entering runaway during programming/erasing of the flash memory or operations that are not according to the established procedures for programming/erasing. Aborting programming or erasure in such cases prevents damage to the flash memory due to excessive programming or erasing.

If the microcomputer malfunctions during programming/erasing of the flash memory, the FLER bit in the FCCS register is set to 1 and the error-protection state is entered, and this aborts the programming or erasure.

The FLER bit is set in the following conditions:

1. When an interrupt such as NMI occurs during programming/erasing.
2. When the flash memory is read during programming/erasing (including a vector read or an instruction fetch).
3. When a SLEEP instruction (including software-standby mode) is executed during programming/erasing.

4. When a bus master other than the CPU, such as the DTC, gets bus mastership during programming/erasing.

Error protection is cancelled only by a reset or by hardware-standby mode. Note that the reset should be released after the reset period of 100 µs which is longer than normal. Since high voltages are applied during programming/erasing of the flash memory, some voltage may remain after the error-protection state has been entered. For this reason, it is necessary to reduce the risk of damage to the flash memory by extending the reset period so that the charge is released.

The state-transition diagram in figure 19.16 shows transitions to and from the error-protection state.

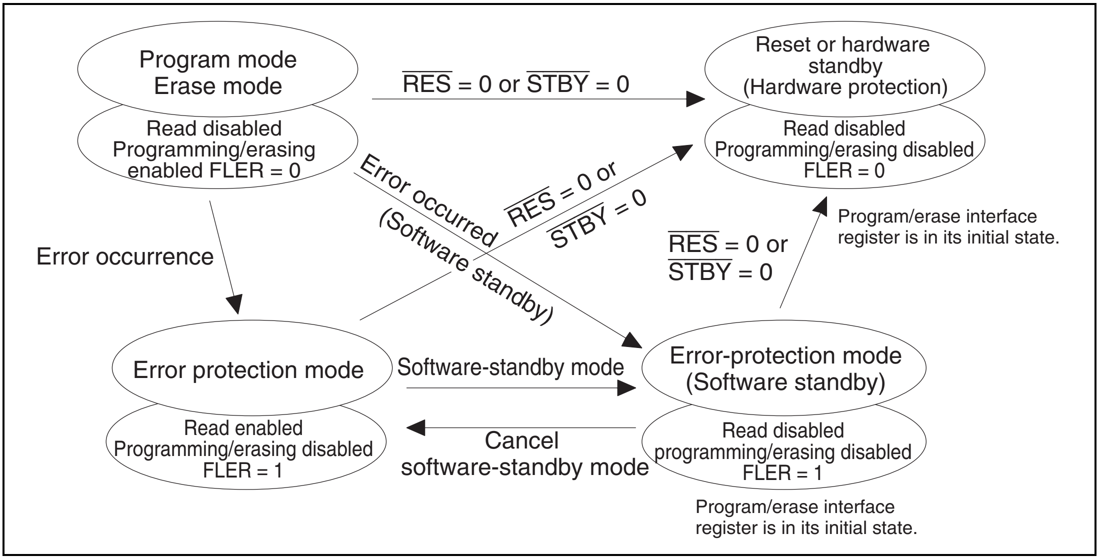
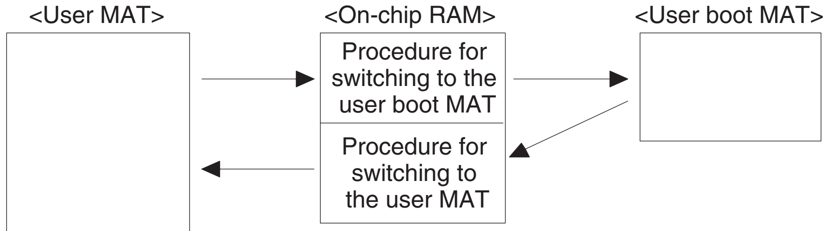


Figure 19.16 Transitions to Error-Protection State

19.6 Switching between User MAT and User Boot MAT

It is possible to alternate between the user MAT and user boot MAT. However, the following procedure is required because these MATs are allocated to address 0.
(Switching to the user boot MAT disables programming and erasing. Programming of the user boot MAT should take place in boot mode or programmer mode.)

1. MAT switching by FMATS should always be executed from the on-chip RAM.
2. To ensure that the MAT that has been switched to is accessible, execute four NOP instructions in the on-chip RAM immediately after writing to FMATS of the on-chip RAM (this prevents access to the flash memory during MAT switching).
3. If an interrupt has occurred during switching, there is no guarantee of which memory MAT is being accessed. Always mask the maskable interrupts before switching between MATs. In addition, configure the system so that NMI interrupts do not occur during MAT switching.
4. After the MATs have been switched, take care because the interrupt vector table will also have been switched. If interrupt processing is to be the same before and after MAT switching, transfer the interrupt-processing routines to the on-chip RAM and set the WEINTE bit in FCCS to place the interrupt-vector table in the on-chip RAM.
5. Memory sizes of the user MAT and user boot MAT are different. When accessing the user boot MAT, do not access addresses above the top of its 16-Kbyte memory space. If access goes beyond the 16-Kbyte space, the values read are undefined.


<User MAT>
<On-chip RAM>
<User boot MAT>
Procedure for switching to the user boot MAT
Procedure for switching to the user MAT

Procedure for switching to the user boot MAT
(1) Mask interrupts
(2) Write H'AA to FMATS.
(3) Execute four NOP instructions before accessing the user boot MAT.

Procedure for switching to the user MAT
(1) Mask interrupts
(2) Write a value other than H'AA to FMATS.
(3) Execute four NOP instructions before accessing the user MAT.

Figure 19.17 Switching between the User MAT and User Boot MAT

19.7 Programmer Mode

Along with its on-board programming mode, this LSI also has a programmer mode as a further mode for the programming and erasing of programs and data.

In the programmer mode, a general-purpose PROM programmer, which supports microcomputers with 512-Kbyte flash memory as a device type*[1], can freely be used to write programs to the on-chip ROM. Program/erase is possible on the user MAT and user boot MAT*[2].

A status-polling system is adopted for operation in automatic program, automatic erase, and status-read modes. In the status-read mode, details of the system's internal signals are output after execution of automatic programming or automatic erasure. In programmer mode, provide a 6-MHz input-clock signal.

Notes: 1. For the PROM programmer and the version of its program, see the instruction manuals for socket adapter.

2. In this LSI, set the programming voltage of the PROM programmer to 3.3 V.

19.8 Serial Communication Interface Specification for Boot Mode

Initiating boot mode enables the boot program to communicate with the host by using the internal SCI. The serial communication interface specification is shown below.

(1) Status

The boot program has three states.

1. Bit-Rate-Adjustment State

   In this state, the boot program adjusts the bit rate to communicate with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, the program enters the inquiry/selection state.
2. Inquiry/Selection State

   In this state, the boot program responds to inquiry commands from the host. The device name, clock mode, and bit rate are selected. After selection of these settings, the program is made to enter the programming/erasing state by the command for a transition to the programming/erasing state. The program transfers the libraries required for erasure to the on-chip RAM and erases the user MATs and user boot MATs before the transition.
3. Programming/erasing state

   Programming and erasure by the boot program take place in this state. The boot program is made to transfer the programming/erasing programs to the RAM by commands from the host. Sum checks and blank checks are executed by sending these commands from the host.

These boot program states are shown in figure 19.18.

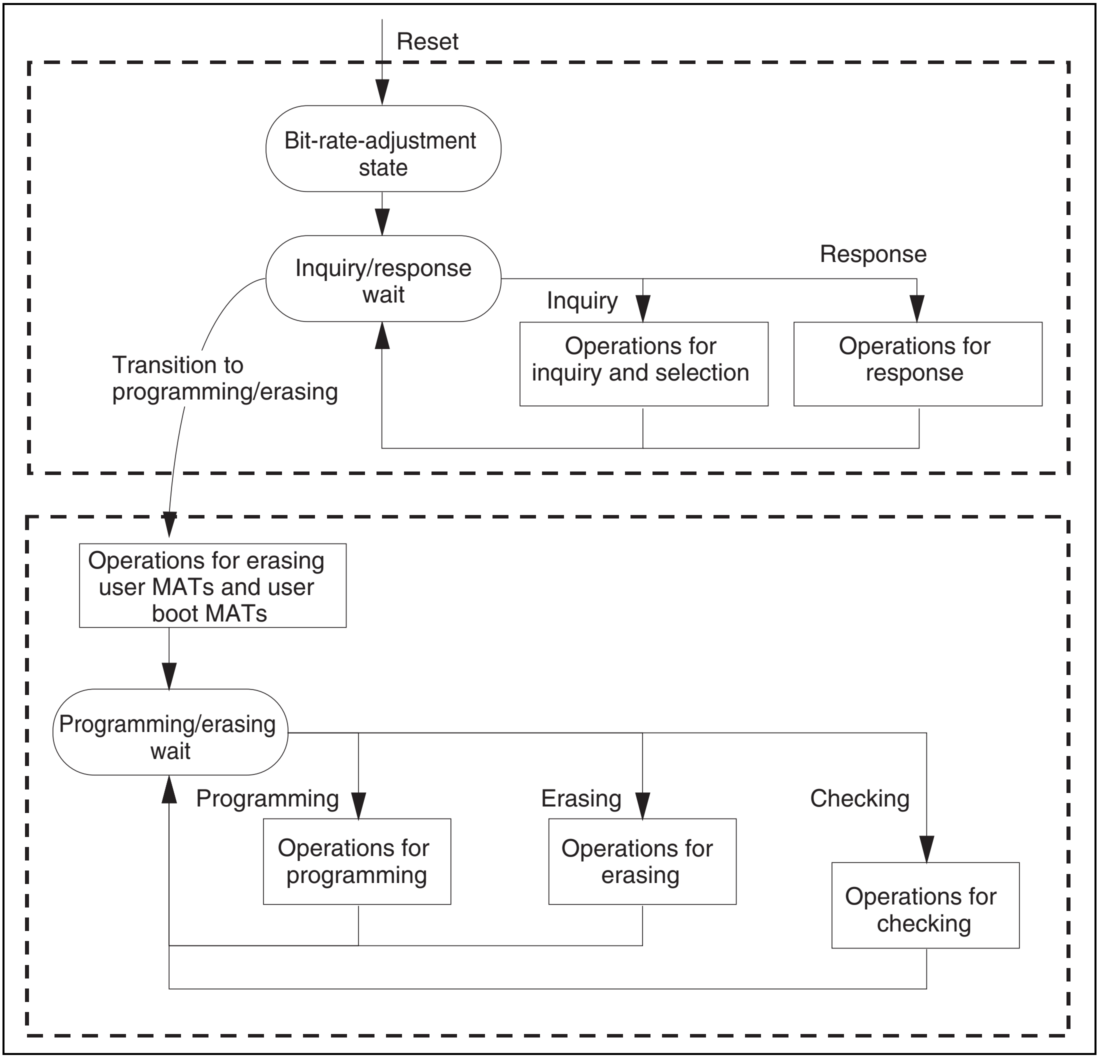

Reset
Bit-rate-adjustment state
Inquiry/response wait
Response
Inquiry
Operations for inquiry and selection
Operations for response
Transition to programming/erasing
Operations for erasing user MATs and user boot MATs
Programming/erasing wait
Programming
Erasing
Checking
Operations for programming
Operations for erasing
Operations for checking

Figure 19.18 Boot Program States

(2) Bit-Rate-Adjustment State

The bit rate is calculated by measuring the period of transfer of a low-level byte (H'00) from the host. The bit rate can be changed by the command for a new bit rate selection. After the bit rate has been adjusted, the boot program enters the inquiry and selection state. The bit-rate-adjustment sequence is shown in figure 19.19.

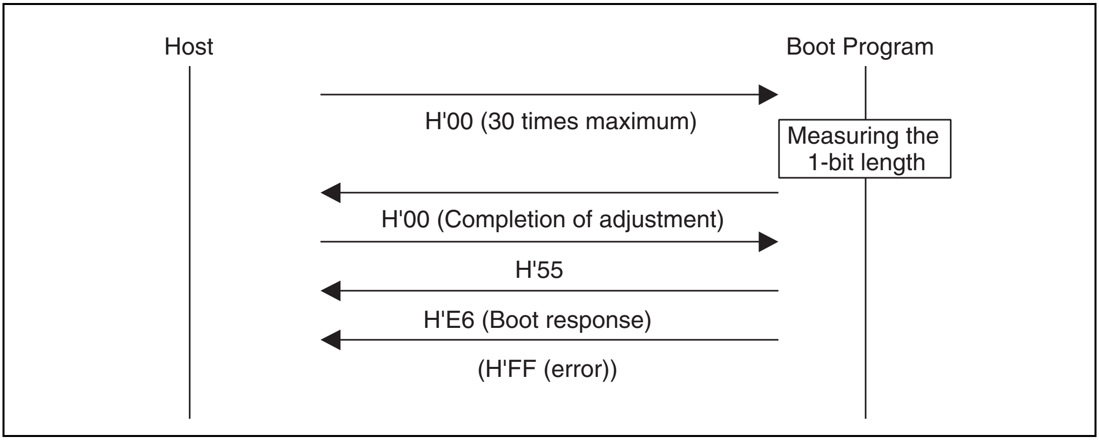


Figure 19.19 Bit-Rate-Adjustment Sequence

(3) Communications Protocol

After adjustment of the bit rate, the protocol for communications between the host and the boot program is as shown below.

1. 1-byte commands and 1-byte responses

   These commands and responses are comprised of a single byte. These are consists of the inquiries and the ACK for successful completion.
2. n-byte commands or n-byte responses

   These commands and responses are comprised of n bytes of data. These are selections and responses to inquiries.

   The amount of programming data is not included under this heading because it is determined in another command.
3. Error response

   The error response is a response to inquiries. It consists of an error response and an error code and comes two bytes.

4. Programming of 128 bytes

   The size is not specified in commands. The size of n is indicated in response to the programming unit inquiry.

5. Memory read response

   This response consists of 4 bytes of data.

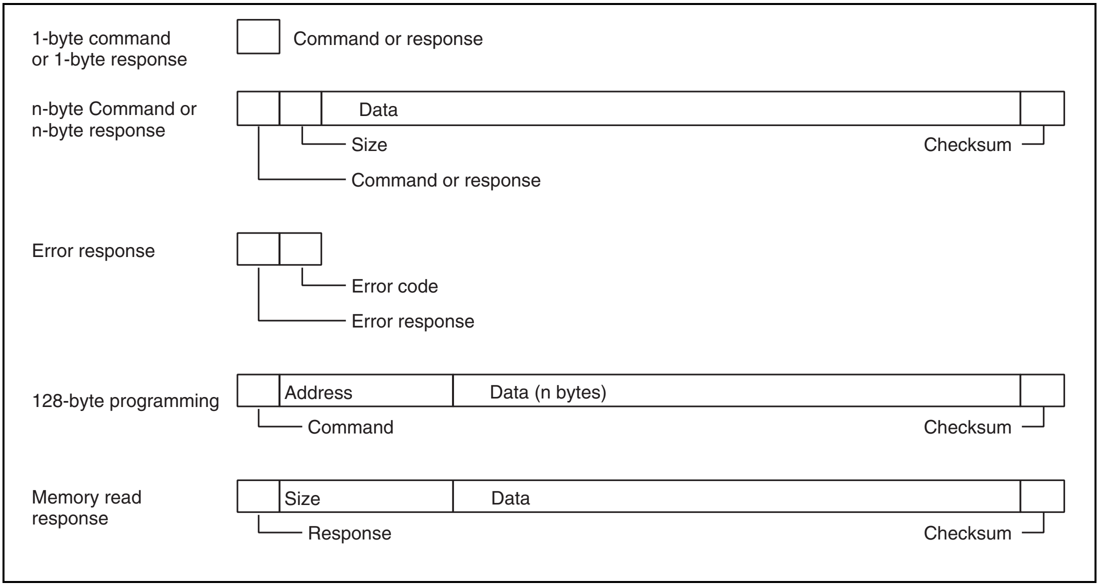


Figure 19.20 Communication Protocol Format

- Command (1 byte): Commands including inquiries, selection, programming, erasing, and checking
- Response (1 byte): Response to an inquiry
- Size (1 byte): The amount of data for transmission excluding the command, amount of data, and checksum
- Checksum (1 byte): The checksum is calculated so that the total of all values from the command byte to the SUM byte becomes H'00.
- Data (n bytes): Detailed data of a command or response
- Error response (1 byte): Error response to a command
- Error code (1 byte): Type of the error
- Address (4 bytes): Address for programming
- Data (n bytes): Data to be programmed (the size is indicated in the response to the programming unit inquiry.)

- Size (4 bytes): 4-byte response to a memory read

(4) Inquiry and Selection States

The boot program returns information from the flash memory in response to the host's inquiry commands and sets the device code, clock mode, and bit rate in response to the host's selection command.

Inquiry and selection commands are listed below.

Table 19.11 Inquiry and Selection Commands

Command	Command Name	Description
H'20	Supported Device Inquiry	Inquiry regarding device codes
H'10	Device Selection	Selection of device code
H'21	Clock Mode Inquiry	Inquiry regarding numbers of clock modes and values of each mode
H'11	Clock Mode Selection	Indication of the selected clock mode
H'22	Multiplication Ratio Inquiry	Inquiry regarding the number of frequency-multiplied clock types, the number of multiplication ratios, and the values of each multiple
H'23	Operating Clock Frequency Inquiry	Inquiry regarding the maximum and minimum values of the main clock and peripheral clocks
H'24	User Boot MAT Information Inquiry	Inquiry regarding the number of user boot MATs and the start and last addresses of each MAT
H'25	User MAT Information Inquiry	Inquiry regarding the a number of user MATs and the start and last addresses of each MAT
H'26	Block for Erasing Information Inquiry	Inquiry regarding the number of blocks and the start and last addresses of each block
H'27	Programming Unit Inquiry	Inquiry regarding the unit of programming data
H'3F	New Bit Rate Selection	Selection of new bit rate
H'40	Transition to Programming/Erasing State	Erasing of user MAT and user boot MAT, and entry to programming/erasing state
H'4F	Boot Program Status Inquiry	Inquiry into the operated status of the boot program

The selection commands, which are device selection (H'10), clock mode selection (H'11), and new bit rate selection (H'3F), should be sent from the host in that order. These commands will certainly be needed. When two or more selection commands are sent at once, the last command will be valid.

All of these commands, except for the boot program status inquiry command (H'4F), will be valid until the boot program receives the programming/erasing transition (H'40). The host can choose the needed commands out of the commands and inquiries listed above. The boot program status inquiry command (H'4F) is valid after the boot program has received the programming/erasing transition command (H'40).

(a) Supported Device Inquiry

The boot program will return the device codes of supported devices and the product code in response to the supported device inquiry.

Command	H'20

- Command, H'20, (1 byte): Inquiry regarding supported devices

Response	H'30	Size	Number of devices	
	Number of characters	Device code		Product name
	...			
	SUM			

- Response, H'30, (1 byte): Response to the supported device inquiry
- Size (1 byte): Number of bytes to be transmitted, excluding the command, size, and checksum, that is, the amount of data contributes by the number of devices, characters, device codes and product names
- Number of devices (1 byte): The number of device types supported by the boot program
- Number of characters (1 byte): The number of characters in the device codes and boot program's name
- Device code (4 bytes): ASCII code of the supporting product
- Product name (n bytes): Type name of the boot program in ASCII-coded characters
- SUM (1 byte): Checksum

  The checksum is calculated so that the total number of all values from the command byte to the SUM byte becomes H'00.

(b) Device Selection

The boot program will set the supported device to the specified device code. The program will return the selected device code in response to the inquiry after this setting has been made.

Command	H'10	Size	Device code	SUM

- Command, H'10, (1 byte): Device selection
- Size (1 byte): Amount of device-code data
  This is fixed at 4.
- Device code (4 bytes): Device code (ASCII code) returned in response to the supported device inquiry
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to the device selection command
  ACK will be returned when the device code matches.

Error response	H'90	ERROR

- Error response, H'90, (1 byte): Error response to the device selection command

ERROR : (1 byte): Error code
H'11: Sum check error
H'21: Device code error, that is, the device code does not match

(c) Clock Mode Inquiry

The boot program will return the supported clock modes in response to the clock mode inquiry.

Command	H'21

- Command, H'21, (1 byte): Inquiry regarding clock mode

Response	H'31	Size	Number of modes	Mode	···	SUM

- Response, H'31, (1 byte): Response to the clock-mode inquiry
- Size (1 byte): Amount of data that represents the number of modes and modes
- Number of clock modes (1 byte): The number of supported clock modes
  H'00 indicates no clock mode or the device allows to read the clock mode.
- Mode (1 byte): Values of the supported clock modes (i.e. H'01 means clock mode 1.)
- SUM (1 byte): Checksum

(d) Clock Mode Selection

The boot program will set the specified clock mode. The program will return the selected clock-mode information after this setting has been made.

The clock-mode selection command should be sent after the device-selection commands.

Command	H'11	Size	Mode	SUM

- Command, H'11, (1 byte): Selection of clock mode
- Size (1 byte): Amount of data that represents the modes
- Mode (1 byte): A clock mode returned in reply to the supported clock mode inquiry.
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to the clock mode selection command
  ACK will be returned when the clock mode matches.

Error Response	H'91	ERROR

- Error response, H'91, (1 byte) : Error response to the clock mode selection command
- ERROR : (1 byte): Error code
  H'11: Checksum error
  H'22: Clock mode error, that is, the clock mode does not match.

Even if the clock mode numbers are H'00 and H'01 by a clock mode inquiry, the clock mode must be selected using these respective values.

(e) Multiplication Ratio Inquiry

The boot program will return the supported multiplication and division ratios.

Command

H'22

- Command, H'22, (1 byte): Inquiry regarding multiplication ratio

Response

H'32	Size	Number of types					
Number of multiplication ratios	Multiplication ratio	...					
...							
SUM							

- Response, H'32, (1 byte): Response to the multiplication ratio inquiry
- Size (1 byte): The amount of data that represents the number of clock sources and multiplication ratios and the multiplication ratios
- Number of types (1 byte): The number of supported multiplied clock types (e.g. when there are two multiplied clock types, which are the main and peripheral clocks, the number of types will be H'02.)
- Number of multiplication ratios (1 byte): The number of multiplication ratios for each type (e.g. the number of multiplication ratios to which the main clock can be set and the peripheral clock can be set.)
- Multiplication ratio (1 byte)

  Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H'04.)

  Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

  The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are types.
- SUM (1 byte): Checksum

(f) Operating Clock Frequency Inquiry

The boot program will return the number of operating clock frequencies, and the maximum and minimum values.

Command	H'23

- Command, H'23, (1 byte): Inquiry regarding operating clock frequencies

Response	H'33	Size	Number of operating clock frequencies
	Minimum value of operating clock frequency		Maximum value of operating clock frequency
	...		
	SUM		

- Response, H'33, (1 byte): Response to operating clock frequency inquiry
- Size (1 byte): The number of bytes that represents the minimum values, maximum values, and the number of frequencies.
- Number of operating clock frequencies (1 byte): The number of supported operating clock frequency types

  (e.g. when there are two operating clock frequency types, which are the main and peripheral clocks, the number of types will be H'02.)
- Minimum value of operating clock frequency (2 bytes): The minimum value of the multiplied or divided clock frequency.

  The minimum and maximum values represent the values in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Maximum value (2 bytes): Maximum value among the multiplied or divided clock frequencies.

  There are as many pairs of minimum and maximum values as there are operating clock frequencies.
- SUM (1 byte): Checksum

(g) User Boot MAT Information Inquiry

The boot program will return the number of user boot MATs and their addresses.

Command	H'24

- Command, H'24, (1 byte): Inquiry regarding user boot MAT information

Response	H'34	Size	Number of areas	
	Area-start address			Area-last address
	...			
	SUM			

- Response, H'34, (1 byte): Response to user boot MAT information inquiry
- Size (1 byte): The number of bytes that represents the number of areas, area-start addresses, and area-last address
- Number of Areas (1 byte): The number of consecutive user boot MAT areas
  When user boot MAT areas are consecutive, the number of areas returned is H'01.
- Area-start address (4 byte): Start address of the area
- Area-last address (4 byte): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (1 byte): Checksum

(h) User MAT Information Inquiry

The boot program will return the number of user MATs and their addresses.

Command	H'25

- Command, H'25, (1 byte): Inquiry regarding user MAT information

Response	H'35	Size	Number of areas	
	Start address area			Last address area
	...			
	SUM			

- Response, H'35, (1 byte): Response to the user MAT information inquiry
- Size (1 byte): The number of bytes that represents the number of areas, area-start address and area-last address
- Number of areas (1 byte): The number of consecutive user MAT areas
  When the user MAT areas are consecutive, the number of areas is H'01.
- Area-start address (4 bytes): Start address of the area

- Area-last address (4 bytes): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (1 byte): Checksum

(i) Erased Block Information Inquiry

The boot program will return the number of erased blocks and their addresses.

Command

H'26

- Command, H'26, (1 byte): Inquiry regarding erased block information

Response

H'36	Size	Number of blocks
Block start address	Block last address	
...		
SUM		

- Response, H'36, (1 byte): Response to the number of erased blocks and addresses
- Size (three bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
- Number of blocks (1 byte): The number of erased blocks
- Block start address (4 bytes): Start address of a block
- Block last Address (4 bytes): Last address of a block
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (1 byte): Checksum

(j) Programming Unit Inquiry

The boot program will return the programming unit used to program data.

Command

H'27

- Command, H'27, (1 byte): Inquiry regarding programming unit

Response

H'37	Size	Programming unit	SUM

- Response, H'37, (1 byte): Response to programming unit inquiry
- Size (1 byte): The number of bytes that indicate the programming unit, which is fixed to 2
- Programming unit (2 bytes): A unit for programming
  This is the unit for reception of programming.
- SUM (1 byte): Checksum

(k) New Bit-Rate Selection

The boot program will set a new bit rate and return the new bit rate.

This selection should be sent after sending the clock mode selection command.

Command	H'3F	Size	Bit rate	Input frequency
	Number of multiplication ratios	Multiplication ratio 1	Multiplication ratio 2	
	SUM			

- Command, H'3F, (1 byte): Selection of new bit rate
- Size (1 byte): The number of bytes that represents the bit rate, input frequency, number of multiplication ratios, and multiplication ratio
- Bit rate (2 bytes): New bit rate
  One hundredth of the value (e.g. when the value is 19200 bps, it will be 192, which is H′00C0.)
- Input frequency (2 bytes): Frequency of the clock input to the boot program
  This is valid to the hundredths place and represents the value in MHz multiplied by 100. (E.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Number of multiplication ratios (1 byte): The number of multiplication ratios to which the device can be set.
- Multiplication ratio 1 (1 byte) : The value of multiplication or division ratios for the main operating frequency
  Multiplication ratio (1 byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)
  Division ratio: The inverse of the division ratio, as a negative number (e.g. when the clock frequency is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- Multiplication ratio 2 (1 byte): The value of multiplication or division ratios for the peripheral frequency
  Multiplication ratio (1 byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)
  (Division ratio: The inverse of the division ratio, as a negative number (E.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to selection of a new bit rate
  When it is possible to set the bit rate, the response will be ACK.

Error Response	H'BF	ERROR

- Error response, H'BF, (1 byte): Error response to selection of new bit rate
- ERROR: (1 byte): Error code

H'11:	Sum checking error
H'24:	Bit-rate selection error The rate is not available.
H'25:	Error in input frequency This input frequency is not within the specified range.
H'26:	Multiplication-ratio error The ratio does not match an available ratio.
H'27:	Operating frequency error The frequency is not within the specified range.

(5) Received Data Check

The methods for checking of received data are listed below.

1. Input frequency

The received value of the input frequency is checked to ensure that it is within the range of minimum to maximum frequencies which matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

2. Multiplication ratio

The received value of the multiplication ratio or division ratio is checked to ensure that it matches the clock modes of the specified device. When the value is out of this range, an multiplication-ratio error is generated.

3. Operating frequency

Operating frequency is calculated from the received value of the input frequency and the multiplication or division ratio. The input frequency is input to the LSI and the LSI is operated at the operating frequency. The expression is given below.

Operating frequency = Input frequency × Multiplication ratio, or
Operating frequency = Input frequency ÷ Division ratio

The calculated operating frequency should be checked to ensure that it is within the range of minimum to maximum frequencies which are available with the clock modes of the specified device. When it is out of this range, an operating frequency error is generated.

4. Bit rate

To facilitate error checking, the value (n) of clock select (CKS) in the serial mode register (SMR), and the value (N) in the bit rate register (BRR), which are found from the peripheral operating clock frequency (ϕ) and bit rate (B), are used to calculate the error rate to ensure that it is less than 4%. If the error is more than 4%, a bit rate error is generated. The error is calculated using the following expression:

$$\text{Error (\%)} = \{[\frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{(2\times n - 1)}}] - 1\} \times 100$$

When the new bit rate is selectable, the rate will be set in the register after sending ACK in response. The host will send an ACK with the new bit rate for confirmation and the boot program will response with that rate.

Confirmation	H'06

- Confirmation, H'06, (1 byte): Confirmation of a new bit rate

Response	H'06

- Response, H'06, (1 byte): Response to confirmation of a new bit rate

The sequence of new bit-rate selection is shown in figure 19.21.

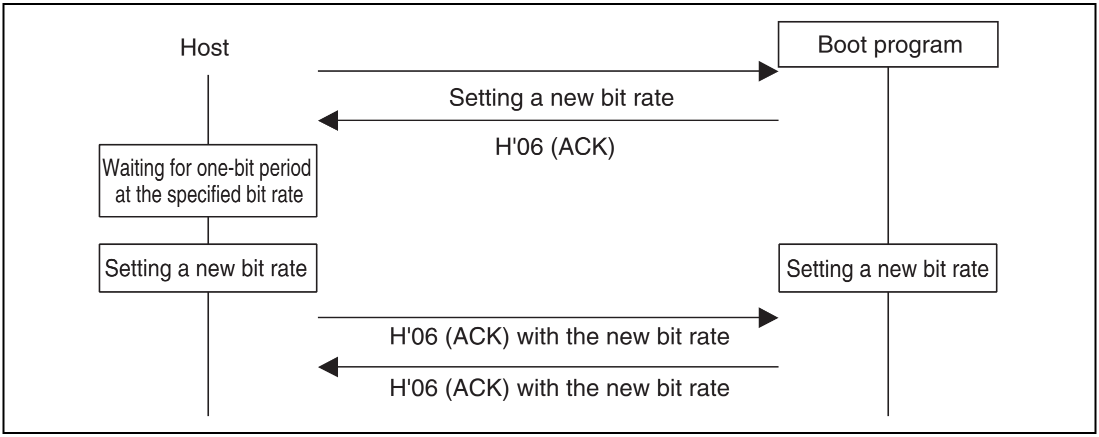

Host
Boot program
Setting a new bit rate
H'06 (ACK)
Waiting for one-bit period at the specified bit rate
Setting a new bit rate
Setting a new bit rate
H'06 (ACK) with the new bit rate
H'06 (ACK) with the new bit rate

Figure 19.21 New Bit-Rate Selection Sequence

(6) Transition to Programming/Erasing State

The boot program will transfer the erasing program, and erase the user MATs and user boot MATs in that order. On completion of this erasure, ACK will be returned and will enter the programming/erasing state.

The host should select the device code, clock mode, and new bit rate with device selection, clock-mode selection, and new bit-rate selection commands, and then send the command for the transition to programming/erasing state. These procedures should be carried out before sending of the programming selection command or program data.

Command	H'40

- Command, H'40, (1 byte): Transition to programming/erasing state

Response	H'06

- Response, H'06, (1 byte): Response to transition to programming/erasing state
  The boot program will send ACK when the user MAT and user boot MAT have been erased by the transferred erasing program.

Error Response	H'C0	H'51

- Error response, H'C0, (1 byte): Error response for user boot MAT blank check
- Error code, H'51, (1 byte): Erasing error
  An error occurred and erasure was not completed.

(7) Command Error

A command error will occur when a command is undefined, the order of commands is incorrect, or a command is unacceptable. Issuing a clock-mode selection command before a device selection or an inquiry command after the transition to programming/erasing state command, are examples.

Error Response	H'80	H'XX

- Error response, H'80, (1 byte): Command error
- Command, H'XX, (1 byte): Received command

(8) Command Order

The order for commands in the inquiry selection state is shown below.

1. A supported device inquiry (H'20) should be made to inquire about the supported devices.
2. The device should be selected from among those described by the returned information and set with a device-selection (H'10) command.
3. A clock-mode inquiry (H'21) should be made to inquire about the supported clock modes.
4. The clock mode should be selected from among those described by the returned information and set.
5. After selection of the device and clock mode, inquiries for other required information should be made, such as the multiplication-ratio inquiry (H'22) or operating frequency inquiry (H'23), which are needed for a new bit-rate selection.
6. A new bit rate should be selected with the new bit-rate selection (H'3F) command, according to the returned information on multiplication ratios and operating frequencies.
7. After selection of the device and clock mode, the information of the user boot MAT and user MAT should be made to inquire about the user boot MATs information inquiry (H'24), user MATs information inquiry (H'25), erased block information inquiry (H'26), and programming unit inquiry (H'27).
8. After making inquiries and selecting a new bit rate, issue the transition to programming/erasing state command (H'40). The boot program will then enter the programming/erasing state.

(9) Programming/Erasing State

A programming selection command makes the boot program select the programming method, a 128-byte programming command makes it program the memory with data, and an erasing selection command and block erasing command make it erase the block. The programming/erasing commands are listed below.

Table 19.12 Programming/Erasing Command

Command	Command Name	Description
H'42	User boot MAT programming selection	Transfers the user boot MAT programming program
H'43	User MAT programming selection	Transfers the user MAT programming program
H'50	128-byte programming	Programs 128 bytes of data
H'48	Erasing selection	Transfers the erasing program
H'58	Block erasing	Erases a block of data
H'52	Memory read	Reads the contents of memory
H'4A	User boot MAT sum check	Checks the checksum of the user boot MAT
H'4B	User MAT sum check	Checks the checksum of the user MAT
H'4C	User boot MAT blank check	Checks whether the contents of the user boot MAT are blank
H'4D	User MAT blank check	Checks whether the contents of the user MAT are blank
H'4F	Boot program status inquiry	Inquires into the boot program’s status

- Programming

  Programming is executed by a programming-selection command and a 128-byte programming command.

  Firstly, the host should send the programming-selection command and select the programming method and programming MATs. There are two programming selection commands, and selection is according to the area and method for programming.

  1. User boot MAT programming selection
  2. User MAT programming selection

  After issuing the programming selection command, the host should send the 128-byte programming command. The 128-byte programming command that follows the selection command represents the data programmed according to the method specified by the selection command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H'FFFFFFFF as the address will stop the programming. On completion of programming, the boot program will wait for selection of programming or erasing.

  Where the sequence of programming operations that is executed includes programming with another method or of another MAT, the procedure must be repeated from the programming selection command.

  The sequence for programming-selection and 128-byte programming commands is shown in figure 19.22.

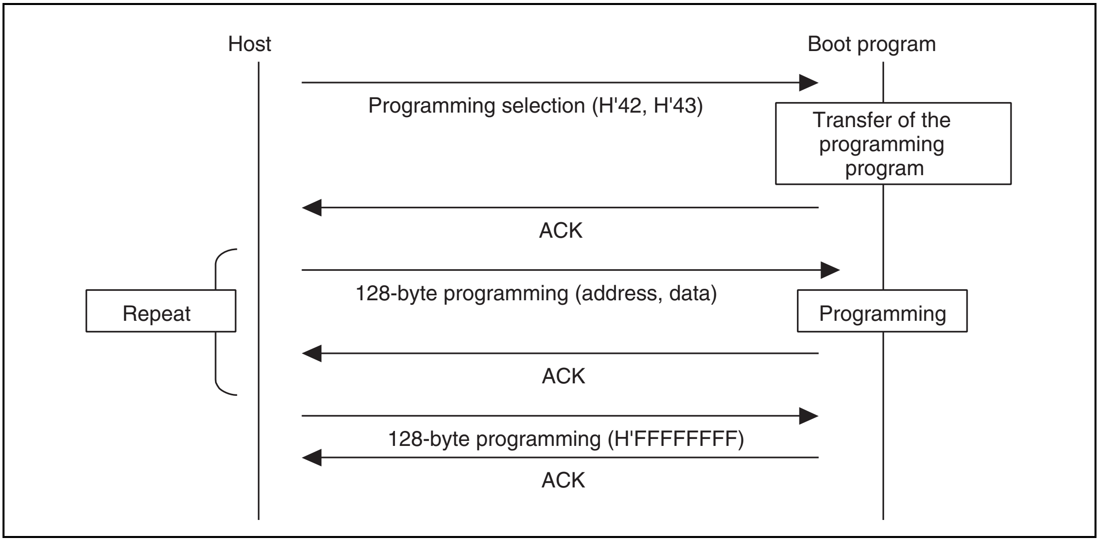

Host
Boot program
Programming selection (H'42, H'43)
Transfer of the programming program
ACK
128-byte programming (address, data)
Repeat
Programming
ACK
128-byte programming (H'FFFFFFFF)
ACK

Figure 19.22 Programming Sequence

(a) User boot MAT programming selection

The boot program will transfer a programming program. The data is programmed to the user boot MATs by the transferred programming program.

Command	H'42

- Command, H'42, (1 byte): User boot MAT programming selection

Response	H'06

- Response, H'06, (1 byte): Response to user boot MAT programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response	H'C2	ERROR

- Error response : H'C2 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code

  H'54: Selection processing error (transfer error occurs and processing is not completed)

- User MAT programming selection

  The boot program will transfer a program for programming. The data is programmed to the user MATs by the transferred program for programming.

Command	H'43

- Command, H'43, (1 byte): User MAT programming selection

Response	H'06

- Response, H'06, (1 byte): Response to user MAT programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response	H'C3	ERROR

- Error response : H'C3 (1 byte): Error response to user MAT programming selection
- ERROR : (1 byte): Error code

  H'54: Selection processing error (transfer error occurs and processing is not completed)

(b) 128-byte programming

The boot program will use the programming program transferred by the programming selection to program the user boot MATs or user MATs in response to 128-byte programming.

Command	H'50	Address						
	Data	···						
	···							
	SUM							

- Command, H'50, (1 byte): 128-byte programming
- Programming Address (4 bytes): Start address for programming
  Multiple of the size specified in response to the programming unit inquiry
  (i.e. H'00, H'01, H'00, H'00: H'00010000)
- Programming Data (128 bytes): Data to be programmed
  The size is specified in the response to the programming unit inquiry.
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response	H'D0	ERROR

- Error response, H'D0, (1 byte): Error response for 128-byte programming
- ERROR: (1 byte): Error code

  H'11: Checksum Error

  H'2A: Address Error

  H'53: Programming error
  A programming error has occurred and programming cannot be continued.

The specified address should match the unit for programming of data. For example, when the programming is in 128-byte units, the lower 8 bits of the address should be H'00 or H'80. When there are less than 128 bytes of data to be programmed, the host should fill the rest with H'FF.

Sending the 128-byte programming command with the address of H'FFFFFFFF will stop the programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.

Command	H'50	Address	SUM

- Command, H'50, (1 byte): 128-byte programming
- Programming Address (4 bytes): End code is H'FF, H'FF, H'FF, H'FF.
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response	H'D0	ERROR

- Error Response, H'D0, (1 byte): Error response for 128-byte programming
- ERROR: (1 byte): Error code

  H'11: Checksum error

  H'2A: Address error

  H'53: Programming error

  An error has occurred in programming and programming cannot be continued.

(10) Erasure

Erasure is performed with the erasure selection and block erasure command.

Firstly, erasure is selected by the erasure selection command and the boot program then erases the specified block. The command should be repeatedly executed if two or more blocks are to be erased. Sending a block-erasure command from the host with the block number H'FF will stop the erasure operating. On completion of erasing, the boot program will wait for selection of programming or erasing.

The sequences of issuing the erasure selection command and block-erasure command are shown in figure 19.23.

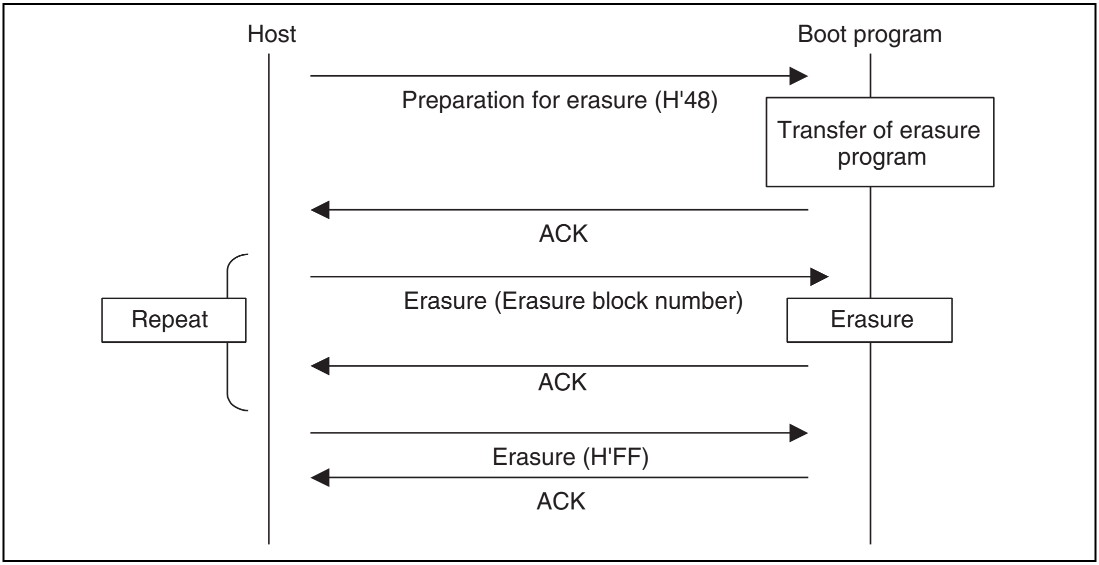

Host
Boot program
Preparation for erasure (H'48)
Transfer of erasure program
ACK
Repeat
Erasure (Erasure block number)
Erasure
ACK
Erasure (H'FF)
ACK

Figure 19.23 Erasure Sequence

(a) Erasure Selection

The boot program will transfer the erasure program. User MAT data is erased by the transferred erasure program.

Command	H'48

- Command, H'48, (1 byte): Erasure selection

Response	H'06

- Response, H'06, (1 byte): Response for erasure selection
  After the erasure program has been transferred, the boot program will return ACK.

Error Response	H'C8	ERROR

- Error Response, H'C8, (1 byte): Error response to erasure selection
- ERROR: (1 byte): Error code

  H'54: Selection processing error (transfer error occurs and processing is not completed)

(b) Block Erasure

The boot program will erase the contents of the specified block.

Command	H'58	Size	Block number	SUM

- Command, H'58, (1 byte): Erasure
- Size (1 byte): The number of bytes that represents the erasure block number
  This is fixed to 1.
- Block number (1 byte): Number of the block to be erased
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to Erasure
  After erasure has been completed, the boot program will return ACK.

Error Response	H'D8	ERROR

- Error Response, H'D8, (1 byte): Response to Erasure
- ERROR (1 byte): Error code

H'11:	Sum check error
H'29:	Block number error Block number is incorrect.
H'51:	Erasure error An error has occurred during erasure.

On receiving block number H'FF, the boot program will stop erasure and wait for a selection command.

Command	H'58	Size	Block number	SUM

- Command, H'58, (1 byte): Erasure
- Size, (1 byte): The number of bytes that represents the block number
  This is fixed to 1.
- Block number (1 byte): H'FF
  Stop code for erasure
- SUM (1 byte): Checksum

Response	H'06

- Response, H'06, (1 byte): Response to end of erasure (ACK)
  When erasure is to be performed after the block number H'FF has been sent, the procedure should be executed from the erasure selection command.

(11) Memory read

The boot program will return the data in the specified address.

Command	H'52	Size	Area	Read address
	Read size	SUM		

- Command: H'52 (1 byte): Memory read
- Size (1 byte): Amount of data that represents the area, read address, and read size (fixed at 9)
- Area (1 byte)
  H'00: User boot MAT
  H'01: User MAT
  An address error occurs when the area setting is incorrect.
- Read address (4 bytes): Start address to be read from
- Read size (4 bytes): Size of data to be read
- SUM (1 byte): Checksum

Response	H'52	Read size						
	Data	···						
	SUM							

- Response: H'52 (1 byte): Response to memory read
- Read size (4 bytes): Size of data to be read
- Data (n bytes): Data for the read size from the read address
- SUM (1 byte): Checksum

Error Response

H'D2	ERROR

- Error response: H'D2 (1 byte): Error response to memory read
- ERROR: (1 byte): Error code

  H'11: Sum check error

  H'2A: Address error
  The read address is not in the MAT.

  H'2B: Size error
  The read size exceeds the MAT.

(12) User Boot MAT Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user boot MAT, as a 4-byte value.

Command

H'4A

- Command, H'4A, (1 byte): Sum check for user-boot MAT

Response

H'5A	Size	Checksum of user boot program	SUM

- Response, H'5A, (1 byte): Response to the sum check of user-boot MAT
- Size (1 byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (4 bytes): Checksum of user boot MATs
  The total of the data is obtained in byte units.
- SUM (1 byte): Sum check for data being transmitted

(13) User MAT Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user MAT.

Command

H'4B

- Command, H'4B, (1 byte): Sum check for user MAT

Response

H'5B	Size	Checksum of user program	SUM

- Response, H'5B, (1 byte): Response to the sum check of the user MAT
- Size (1 byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (4 bytes): Checksum of user MATs
  The total of the data is obtained in byte units.
- SUM (1 byte): Sum check for data being transmitted

(14) User Boot MAT Blank Check

The boot program will check whether or not all user boot MATs are blank and return the result.

Command	H'4C

- Command, H'4C, (1 byte): Blank check for user boot MAT

Response	H'06

- Response, H'06, (1 byte): Response to the blank check of user boot MAT
  If all user MATs are blank (H'FF), the boot program will return ACK.

Error Response	H'CC	H'52

- Error Response, H'CC, (1 byte): Response to blank check for user boot MAT
- Error Code, H'52, (1 byte): Erasure has not been completed.

(15) User MAT Blank Check

The boot program will check whether or not all user MATs are blank and return the result.

Command	H'4D

- Command, H'4D, (1 byte): Blank check for user MATs

Response	H'06

- Response, H'06, (1 byte): Response to the blank check for user boot MATs
  If the contents of all user MATs are blank (H'FF), the boot program will return ACK.

Error Response	H'CD	H'52

- Error Response, H'CD, (1 byte): Error response to the blank check of user MATs.
- Error code, H'52, (1 byte): Erasure has not been completed.

(16) Boot Program State Inquiry

The boot program will return indications of its present state and error condition. This inquiry can be made in the inquiry/selection state or the programming/erasing state.

Command	H'4F

- Command, H'4F, (1 byte): Inquiry regarding boot program's state

Response	H'5F	Size	Status	ERROR	SUM

- Response, H'5F, (1 byte): Response to boot program state inquiry
- Size (1 byte): The number of bytes. This is fixed to 2.
- Status (1 byte): State of the boot program
- ERROR (1 byte): Error status

  ERROR = 0 indicates normal operation.
  ERROR = 1 indicates error has occurred.

- SUM (1 byte): Sum check

Table 19.13 Status Code

Code	Description
H'11	Device Selection Wait
H'12	Clock Mode Selection Wait
H'13	Bit Rate Selection Wait
H'1F	Programming/Erasing State Transition Wait (Bit rate selection is completed)
H'31	Programming State for Erasure
H'3F	Programming/Erasing Selection Wait (Erasure is completed)
H'4F	Programming Data Receive Wait (Programming is completed)
H'5F	Erasure Block Specification Wait (Erasure is completed)

Table 19.14 Error Code

Code	Description
H'00	No Error
H'11	Sum Check Error
H'12	Program Size Error
H'21	Device Code Mismatch Error
H'22	Clock Mode Mismatch Error
H'24	Bit Rate Selection Error
H'25	Input Frequency Error
H'26	Multiplication Ratio Error
H'27	Operating Frequency Error
H'29	Block Number Error
H'2A	Address Error
H'2B	Data Length Error
H'51	Erasure Error
H'52	Erasure Incomplete Error
H'53	Programming Error
H'54	Selection Processing Error
H'80	Command Error
H'FF	Bit-Rate-Adjustment Confirmation Error

19.9 Usage Notes

1. The initial state of the product at its shipment is in the erased state. For the product whose revision of erasing is undefined, we recommend to execute automatic erasure for checking the initial state (erased state) and compensating.
2. For the PROM programmer suitable for programmer mode in this LSI and its program version, refer to the instruction manual of the socket adapter.
3. If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.
4. If a voltage higher than the rated voltage is applied, the product may be fatally damaged. Use a PROM programmer that supports the 512-Kbyte flash memory on-chip MCU device at 3.3 V. Do not set the programmer to HN28F101 or the programming voltage to 5.0 V. Use only the specified socket adapter. If other adapters are used, the product may be damaged.
5. Do not remove the chip from the PROM programmer nor input a reset signal during programming/erasing. As a high voltage is applied to the flash memory during programming/erasing, doing so may damage or destroy flash memory permanently. If reset is executed accidentally, reset must be released after the reset input period of 100 μs which is longer than normal.
6. The flash memory is not accessible until FKEY is cleared after programming/erasing completes. If this LSI is restarted by a reset immediately after programming/erasing has finished, secure the reset period (period of $\overline{\text{RES}}$ = 0) of more than 100 μs. Though transition to the reset state or hardware standby state during programming/erasing is prohibited, if reset is executed accidentally, reset must be released after the reset input period of 100 μs which is longer than normal.
7. At powering on or off the Vcc power supply, fix the $\overline{\text{RES}}$ pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.
8. 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.
9. When the chip is to be reprogrammed with the programmer after execution of programming or erasure in on-board programming mode, it is recommend that automatic programming is performed after execution of automatic erasure.
10. To write data or programs to the flash memory, data or programs must be allocated to addresses higher than that of the external interrupt vector table (H'000040) and H'FF must be written to the areas that are reserved for the system in the exception handling vector table.

11. If data other than H'FFFFFFFF is written to the key code area (H'00003C to H'00003F) of flash memory, only H'00 can be read in programmer mode. (In this case, data is read as H'00. Rewrite is possible after erasing the data.) For reading in programmer mode, make sure to write H'FFFFFFFF to the entire key code area. If data other than H'FF is to be written to the key code area in programmer mode, a verification error will occur unless a software countermeasure is taken for the PROM programmer and the version of its program.
12. The programming program that includes the initialization routine and the erasing program that includes the initialization routine are each 3 Kbytes or less. Accordingly, when the CPU clock frequency is 25 MHz, the download for each program takes approximately 256 µs at the maximum.
13. While an instruction in on-chip RAM is being executed, the DTC can write to the SCO bit in FCCS that is used for a download request or FMATS that is used for MAT switching. Make sure that these registers are not accidentally written to, otherwise an on-chip program may be downloaded and damage RAM or a MAT switchover may occur and the CPU get out of control. Do not use DTC to program flash related registers.
14. A programming/erasing program for flash memory used in the conventional H8S F-ZTAT microcomputer which does not support download of the on-chip program by a SCO transfer request cannot run in this LSI. Be sure to download the on-chip program to execute programming/erasing of flash memory in this LSI.
15. Unlike the conventional H8S F-ZTAT microcomputer, no countermeasures are available for a runaway by WDT during programming/erasing. Prepare countermeasures (e.g. use of the periodic timer interrupts) for WDT with taking the programming/erasing time into consideration as required.

Section 20 Boundary Scan (JTAG)

The JTAG (Joint Test Action Group) is standardized as an international standard, IEEE Standard 1149.1, and is open to the public as IEEE Standard Test Access Port and Boundary-Scan Architecture. Although the name of the function is boundary scan and the name of the group who worked on standardization is the JTAG, the JTAG is commonly used as the name of a boundary scan architecture and a serial interface to access the devices having the architecture.

This LSI has a boundary scan function (JTAG). Using this function along with other LSIs facilitates testing a printed-circuit board.

20.1 Features

- Five test pins (ETCK, ETDI, ETDO, ETMS, and $\overline{\text{ETRST}}$)
- TAP controller
- Six instructions

  BYPASS mode

  EXTEST mode

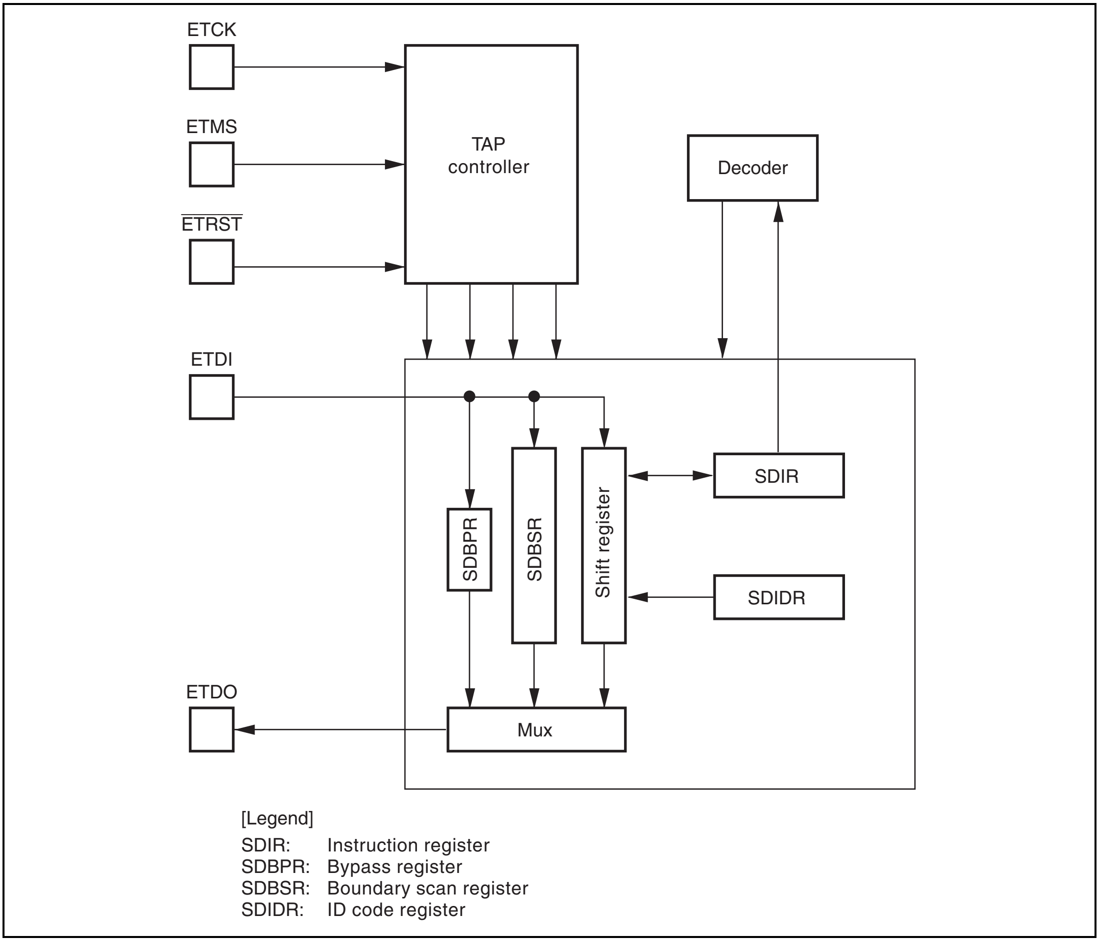
  SAMPLE/PRELOAD mode

  CLAMP mode

  HIGHZ mode

  IDCODE mode

  (These instructions are test modes corresponding to IEEE 1149.1.)



[Legend]

SDIR:	Instruction register
SDBPR:	Bypass register
SDBSR:	Boundary scan register
SDIDR:	ID code register

Figure 20.1 JTAG Block Diagram

20.2 Input/Output Pins

Table 20.1 shows the JTAG pin configuration.

Table 20.1 Pin Configuration

Pin Name	Abbreviation	I/O	Function
Test clock	ETCK	Input	Test Clock Input Provides an independent clock supply to the JTAG. As the clock input to the ETCK pin is supplied directly to the JTAG, a clock waveform with a duty cycle close to 50% should be input. For details, see section 24, Electrical Characteristics.
Test mode select	ETMS	Input	Test Mode Select Input Sampled on the rise of the ETCK pin. The ETMS pin controls the internal state of the TAP controller.
Test data input	ETDI	Input	Serial Data Input Performs serial input of instructions and data for JTAG registers. ETDI is sampled on the rise of the ETCK pin.
Test data output	ETDO	Output	Serial Data Output Performs serial output of instructions and data from JTAG registers. Transfer is performed in synchronization with the ETCK pin. If there is no output, the ETDO pin goes to the high-impedance state.
Test reset	$\overline{\text{ETRST}}$	Input	Test Reset Input Signal Initializes the JTAG asynchronously.

20.3 Register Descriptions

The JTAG has the following registers.

- Instruction register (SDIR)
- Bypass register (SDBPR)
- Boundary scan register (SDBSR)
- ID code register (SDIDR)

Instructions can be input to the instruction register (SDIR) by serial transfer from the test data input pin (ETDI). Data from SDIR can be output via the test data output pin (ETDO). The bypass register (SDBPR) is a 1-bit register to which the ETDI and ETDO pins are connected in BYPASS, CLAMP, or HIGHZ mode. The boundary scan register (SDBSR) is a 210-bit register to which the ETDI and ETDO pins are connected in SAMPLE/PRELOAD or EXTEST mode. The ID code register (SDIDR) is a 32-bit register; a fixed code can be output via the ETDO pin in IDCODE mode. All registers cannot be accessed directly by the CPU.

Table 20.2 shows the kinds of serial transfer possible with each JTAG register.

Table 20.2 JTAG Register Serial Transfer

Register	Serial Input	Serial Output
SDIR	Possible	Possible
SDBPR	Possible	Possible
SDBSR	Possible	Possible
SDIDR	Impossible	Possible

20.3.1 Instruction Register (SDIR)

SDIR is a 32-bit register. JTAG instructions can be transferred to SDIR by serial input from the ETDI pin. SDIR can be initialized when the $\overline{\text{ETRST}}$ pin is low or the TAP controller is in the Test-Logic-Reset state, but is not initialized by a reset or in standby mode.

Only 4-bit instructions can be transferred to SDIR. If an instruction exceeding 4 bits is input, the last 4 bits of the serial data will be stored in SDIR.

		Initial		
Bit	**Bit Name**	**Value**	**R/W**	**Description**
31	TS3	1	R/W	Test Set Bits
30	TS2	1	R/W	0000: EXTEST mode
29	TS1	1	R/W	0001: Setting prohibited
28	TS0	0	R/W	0010: CLAMP mode
				0011: HIGHZ mode
				0100: SAMPLE/PRELOAD mode
				0101: Setting prohibited
				: :
				1101: Setting prohibited
				1110: IDCODE mode (Initial value)
				1111: BYPASS mode
27 to 14	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
13	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
12	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
11	—	1	R	Reserved This bit is always read as 1 and cannot be modified.
10 to 1	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
0	—	1	R	Reserved This bit is always read as 1 and cannot be modified.

20.3.2 Bypass Register (SDBPR)

SDBPR is a 1-bit shift register. In BYPASS, CLAMP, or HIGHZ mode, SDBPR is connected between the ETDI and ETDO pins.

20.3.3 Boundary Scan Register (SDBSR)

SDBSR is a shift register provided on the PAD for controlling the I/O pins of this LSI.

Using EXTEST mode or SAMPLE/PRELOAD mode, a boundary scan test conforming to the IEEE1149.1 standard can be performed.

Table 20.3 shows the relationship between the pins of this LSI and the boundary scan register.

Table 20.3 Correspondence between Pins and Boundary Scan Register

		Input/	
Pin No.	Pin Name	Output	Bit No.
	from ETDI		
A01	VCC	—	—
		—	—
		—	—
B02	P45	Input	211
		Enable	210
		Output	209
B01	P46	Input	208
		Enable	207
		Output	206
D04	P47	Input	205
		Enable	204
		Output	203
C02	P56	Input	202
		Enable	201
		Output	200
C01	P57	Input	199
		Enable	198
		Output	197
D03	VSS	—	—
		—	—
		—	—
D02	$\overline{\text{RES}}$	—	—
		—	—
		—	—
D01	MD1	Input	196
		—	—
		—	—
E04	NMI	Input	195
		—	—
		—	—
E03	$\overline{\text{STBY}}$	—	—
		—	—
		—	—
E01	VCL	—	—
		—	—
		—	—
E02	NC	—	—
		—	—
		—	—
F03	$\overline{\text{MD2}}$	Input	194
		Reserved	193
		Reserved	192
F01	PC7	Input	191
		Enable	190
		Output	189
F02	PC6	Input	188
		Enable	187
		Output	186
F04	PC3	Input	185
		Enable	184
		Output	183
G01	PC2	Input	182
		Enable	181
		Output	180
G02	PC1	Input	179
		Enable	178
		Output	177

Pin No.	Pin Name	Input/ Output	Bit No.
G03	PC0	Input	176
		Enable	175
		Output	174
H01	PA7	Input	173
		Enable	172
		Output	171
G04	PA6	Input	170
		Enable	169
		Output	168
H02	PA5	Input	167
		Enable	166
		Output	165
J01	VCC	—	—
		—	—
		—	—
H03	NC	—	—
		—	—
		—	—
J02	PA4	Input	164
		Enable	163
		Output	162
K01	PA3	Input	161
		Enable	160
		Output	159
J03	NC	—	—
		—	—
		—	—
L01	PA2	Input	158
		Enable	157
		Output	156
K02	PA1	Input	155
		Enable	154
		Output	153
L02	PA0	Input	152
		Enable	151
		Output	150
H04	VSS	—	—
		—	—
		—	—
K03	P87	Input	149
		Enable	148
		Output	147
L03	P86	Input	146
		Enable	145
		Output	144
J04	P85	Input	143
		Enable	142
		Output	141
K04	P84	Input	140
		Enable	139
		Output	138
L04	P83	Input	137
		Enable	136
		Output	135
H05	P82	Input	134
		Enable	133
		Output	132
J05	P81	Input	131
		Enable	130
		Output	129
L05	P80	Input	128
		Enable	127
		Output	126
K05	NC	—	—
		—	—
		—	—
J06	PE7	Input	125
		Enable	124
		Output	123
L06	PE6	Input	122
		Enable	121
		Output	120

Pin No.	Pin Name	Input/ Output	Bit No.
K06	PE5	Input	119
		Enable	118
		Output	117
H06	PE4	Input	116
		Enable	115
		Output	114
L07	PE3	Input	113
		Enable	112
		Output	111
K07	PE2	Input	110
		Enable	109
		Output	108
J07	PE1	Input	107
		Enable	106
		Output	105
L08	PE0	Input	104
		Enable	103
		Output	102
H07	AVSS	—	—
		—	—
		—	—
K08	P70	Input	101
		—	—
		—	—
L09	P71	Input	100
		—	—
		—	—
J08	NC	—	—
		—	—
		—	—
K09	P72	Input	99
		—	—
		—	—
L10	P73	Input	98
		—	—
		—	—

Pin No.	Pin Name	Input/ Output	Bit No.
J09	NC	—	—
		—	—
		—	—
L11	P74	Input	97
		—	—
		—	—
K10	P75	Input	96
		—	—
		—	—
K11	P76	Input	95
		—	—
		—	—
H08	P77	Input	94
		—	—
		—	—
J10	AVCC	—	—
		—	—
		—	—
J11	AVref	—	—
		—	—
		—	—
H09	P60	Input	93
		Enable	92
		Output	91
H10	P61	Input	90
		Enable	89
		Output	88
H11	P62	Input	87
		Enable	86
		Output	85
G08	P63	Input	84
		Enable	83
		Output	82
G09	VCC	—	—
		—	—
		—	—

Pin No.	Pin Name	Input/ Output	Bit No.
G11	ETMS	—	—
		—	—
		—	—
G10	NC	—	—
		—	—
		—	—
F09	ETDO	—	—
		—	—
		—	—
F11	ETDI	—	—
		—	—
		—	—
F10	ETCK	—	—
		—	—
		—	—
F08	$\overline{\text{ETRST}}$	—	—
		—	—
		—	—
E11	VSS	—	—
		—	—
		—	—
E10	P23	Input	81
		Enable	80
		Output	79
E09	P22	Input	78
		Enable	77
		Output	76
D11	P21	Input	75
		Enable	74
		Output	73
E08	P20	Input	72
		Enable	71
		Output	70
D10	P17	Input	69
		Enable	68
		Output	67
C11	P16	Input	66
		Enable	65
		Output	64
D09	NC	—	—
		—	—
		—	—
C10	P15	Input	63
		Enable	62
		Output	61
B11	P14	Input	60
		Enable	59
		Output	58
C09	NC	—	—
		—	—
		—	—
A11	P13	Input	57
		Enable	56
		Output	55
B10	P12	Input	54
		Enable	53
		Output	52
A10	P11	Input	51
		Enable	50
		Output	49
D08	VSS	—	—
		—	—
		—	—
B09	P10	Input	48
		Enable	47
		Output	46
A09	P30	Input	45
		Enable	44
		Output	43
C08	P31	Input	42
		Enable	41
		Output	40

Pin No.	Pin Name	Input/ Output	Bit No.
B08	P32	Input	39
		Enable	38
		Output	37
A08	P33	Input	36
		Enable	35
		Output	34
D07	P34	Input	33
		Enable	32
		Output	31
C07	P35	Input	30
		Enable	29
		Output	28
A07	P36	Input	27
		Enable	26
		Output	25
B07	NC	—	—
		—	—
		—	—
C06	P37	Input	24
		Enable	23
		Output	22
A06	P40	Input	21
		Enable	20
		Output	19
B06	P41	Input	18
		Enable	17
		Output	16
D06	P42	Input	15
		Enable	14
		Output	13
A05	P43	Input	12
		Enable	11
		Output	10
B05	P52	Input	9
		Enable	8
		Output	7
C05	P53	Input	6
		Enable	5
		Output	4
A04	FWE	Input	3
		—	—
		—	—
D05	P44	Input	2
		Enable	1
		Output	0
B04	VSS	—	—
		—	—
		—	—
A03	$\overline{\text{RESO}}$	—	—
		—	—
		—	—
C04	NC	—	—
		—	—
		—	—
B03	XTAL	—	—
		—	—
		—	—
A02	EXTAL	—	—
		—	—
		—	—
C03	NC	—	—
		—	—
		—	—
to ETDO			

20.3.4 ID Code Register (SDIDR)

SDIDR is a 32-bit register. In IDCODE mode, SDIDR can output a fixed code, H'08039447, from the ETDO pin. However, no serial data can be written to SDIDR via the ETDI pin.

31 28	27 12	11 1	0
0000	1000 0000 0011 1001	0100 0100 011	1
Version (4 bits)	Part Number (16 bits)	Manufacture Identify (11 bits)	Fixed Code (1 bit)

20.4 Operation

20.4.1 TAP Controller State Transitions

Figure 20.2 shows the internal states of the TAP controller. State transitions basically conform to the IEEE1149.1 standard.

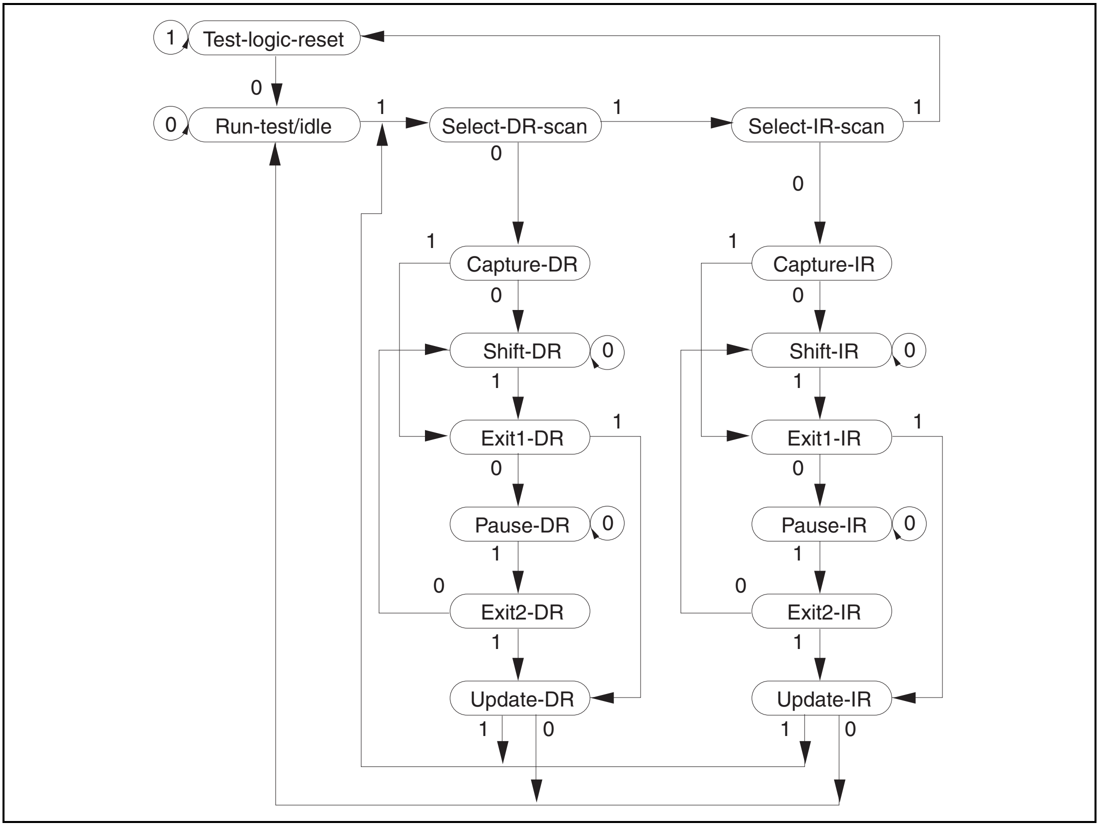


Figure 20.2 TAP Controller State Transitions

20.4.2 JTAG Reset

The JTAG can be reset in two ways.

- The JTAG is reset when the $\overline{\text{ETRST}}$ pin is held at 0.
- When $\overline{\text{ETRST}}$ = 1, the JTAG can be reset by inputting at least five ETCK clock cycles while ETMS = 1.

20.5 Boundary Scan

The JTAG pins can be placed in the boundary scan mode stipulated by the IEEE1149.1 standard by setting a command in SDIR.

20.5.1 Supported Instructions

This LSI supports the three essential instructions defined in the IEEE1149.1 standard (BYPASS, SAMPLE/PRELOAD, and EXTEST) and optional instructions (CLAMP, HIGHZ, and IDCODE).

(1) BYPASS (Instruction code: B'1111)

The BYPASS instruction is an instruction that operates the bypass register. This instruction shortens the shift path to speed up serial data transfer involving other chips on the printed circuit board. While this instruction is being executed, the test circuit has no effect on the system circuits.

(2) SAMPLE/PRELOAD (Instruction code: B'0100)

The SAMPLE/PRELOAD instruction inputs values from this LSI internal circuitry to the boundary scan register, outputs values from the scan path, and loads data onto the scan path. When this instruction is being executed, this LSI's input pin signals are transmitted directly to the internal circuitry, and internal circuit values are directly output externally from the output pins. This LSI system circuits are not affected by execution of this instruction.

In a SAMPLE operation, a snapshot of a value to be transferred from an input pin to the internal circuitry, or a value to be transferred from the internal circuitry to an output pin, is latched into the boundary scan register and read from the scan path. Snapshot latching does not affect normal operation of this LSI.

In a PRELOAD operation, an initial value is set in the parallel output latch of the boundary scan register from the scan path prior to the EXTEST instruction. Without a PRELOAD operation, when the EXTEST instruction was executed an undefined value would be output from the output pin until completion of the initial scan sequence (transfer to the output latch) (with the EXTEST instruction, the parallel output latch value is constantly output to the output pin).

(3) EXTEST (Instruction code: B'0000)

The EXTEST instruction is provided to test external circuitry when this LSI is mounted on a printed circuit board. When this instruction is executed, output pins are used to output test data (previously set by the SAMPLE/PRELOAD instruction) from the boundary scan register to the printed circuit board, and input pins are used to latch test results into the boundary scan register from the printed circuit board. If testing is carried out by using the EXTEST instruction N times, the Nth test data is scanned in when test data (N-1) is scanned out.

Data loaded into the output pin boundary scan register in the Capture-DR state is not used for external circuit testing (it is replaced by a shift operation).

(4) CLAMP (Instruction code: B'0010)

When the CLAMP instruction is enabled, the output pin outputs the value of the boundary scan register that has been previously set by the SAMPLE/PRELOAD instruction. While the CLAMP instruction is enabled, the state of the boundary scan register maintains the previous state regardless of the state of the TAP controller.

A bypass register is connected between the ETDI and ETDO pins. The related circuit operates in the same way when the BYPASS instruction is enabled.

(5) HIGHZ (Instruction code: B'0011)

When the HIGHZ instruction is enabled, all output pins enter a high-impedance state. While the HIGHZ instruction is enabled, the state of the boundary scan register maintains the previous state regardless of the state of the TAP controller.

A bypass register is connected between the ETDI and ETDO pins. The related circuit operates in the same way when the BYPASS instruction is enabled.

(6) IDCODE (Instruction code: B'1110)

When the IDCODE instruction is enabled, the value of the ID code register is output from the ETDO pin with LSB first when the TAP controller is in the Shift-DR state. While the IDCODE instruction is being executed, the test circuit does not affect the system circuit.

When the TAP controller is in the Test-Logic-Reset state, the instruction register is initialized to the IDCODE instruction.

Notes: 1. Boundary scan mode does not cover power-supply-related pins (VCC, VCL, VSS, AVCC, AVSS, and AVref).

2. Boundary scan mode does not cover clock-related pins (EXTAL, XTAL, and PFSEL).

3. Boundary scan mode does not cover reset- and standby-related pins ($\overline{\text{RES}}$, $\overline{\text{STBY}}$, and $\overline{\text{RESO}}$).

4. Boundary scan mode does not cover JTAG-related pins (ETCK, ETDI, ETDO, ETMS, and $\overline{\text{ETRST}}$).

5. Fix the $\overline{\text{MD2}}$ pin high.

6. Use the $\overline{\text{STBY}}$ pin in high state.

20.6 Usage Notes

1. A reset must always be executed by driving the $\overline{\text{ETRST}}$ pin to 0, regardless of whether or not the JTAG is to be activated. The $\overline{\text{ETRST}}$ pin must be held low for 20 ETCK clock cycles. For details, see section 24, Electrical Characteristics. To activate the JTAG after a reset, drive the $\overline{\text{ETRST}}$ pin to 1 and specify the ETCK, ETMS, and ETDI pins to any value. If the JTAG is not to be activated, drive the $\overline{\text{ETRST}}$, ETCK, ETMS, and ETDI pins to 1 or the high-impedance state.
2. The following must be considered when the power-on reset signal is applied to the $\overline{\text{ETRST}}$ pin.
   - The reset signal must be applied at power-on.
   - To prevent the LSI system operation from being affected by the $\overline{\text{ETRST}}$ pin of the board tester, circuits must be separated.
   - Alternatively, to prevent the $\overline{\text{ETRST}}$ pin of the board tester from being affected by the LSI system reset, circuits must be separated.

   Figure 20.3 shows a design example of the reset signal circuit wherein no reset signal interference occurs.

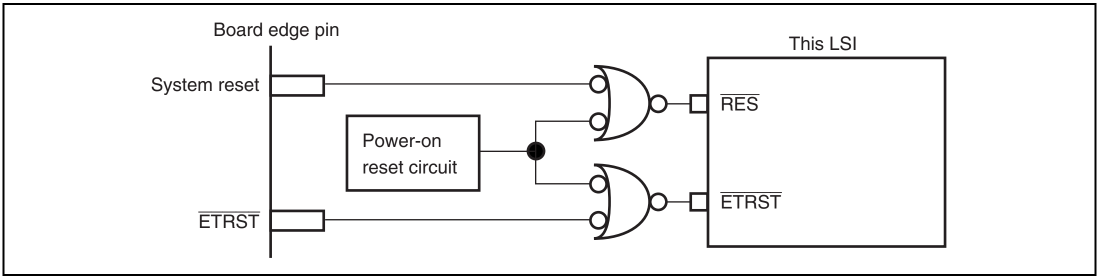


Figure 20.3 Reset Signal Circuit Without Reset Signal Interference

3. The registers are not initialized in standby mode. If the $\overline{\text{ETRST}}$ pin is set to 0 in standby mode, IDCODE mode will be entered.
4. The frequency of the ETCK pin must be lower than that of the system clock. For details, see section 24, Electrical Characteristics.
5. Data input/output in serial data transfer starts from the LSB. Figures 20.4 and 20.5 shows examples of serial data input/output.
6. When data that exceeds the number of bits of the register connected between the ETDI and ETDO pins is serially transferred, the serial data that exceeds the number of register bits and output from the ETDO pin is the same as that input from the ETDI pin.
7. If the JTAG serial transfer sequence is disrupted, the $\overline{\text{ETRST}}$ pin must be reset. Transfer should then be retried, regardless of the transfer operation.

8. If a pin with a pull-up function is sampled while its pull-up function is enabled, 1 can be detected at the corresponding input scan register. In this case, the corresponding enable scan register should be cleared to 0.
9. If a pin with an open-drain function is sampled while its open-drain function is enabled and its corresponding output scan register is 1, 0 can be detected at the corresponding enable scan register.

SDIR serial data input/output

SDIR is captured into the shift register in Capture-IR, and bits 0 to 31 of SDIR are output in that order from the ETDO pin in Shift-IR.

Data input from the ETDI pin is written to SDIR in Update-IR.

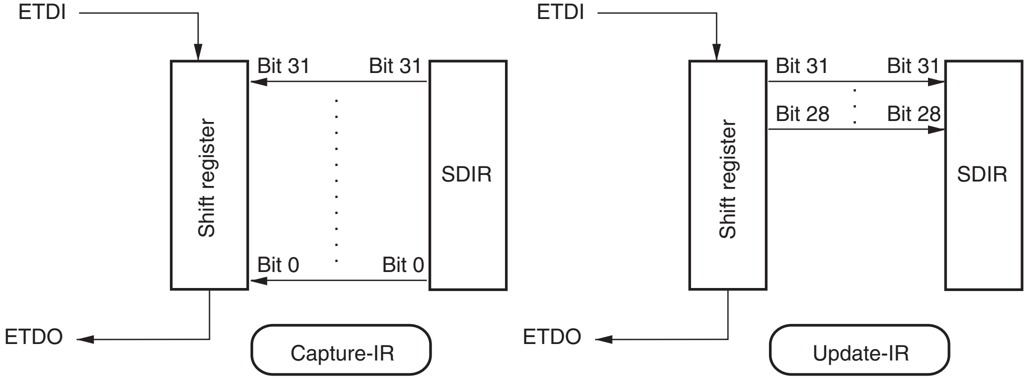


Figure 20.4 Serial Data Input/Output (1)

SDIDR serial data input/output

SDIDR is captured into the shift register in Capture-DR in IDCODE mode, and bits 0 to 31 of SDIDR are output in that order from the ETDO pin in Shift-DR.

Data input from the ETDI pin is not written to any register in Update-DR.

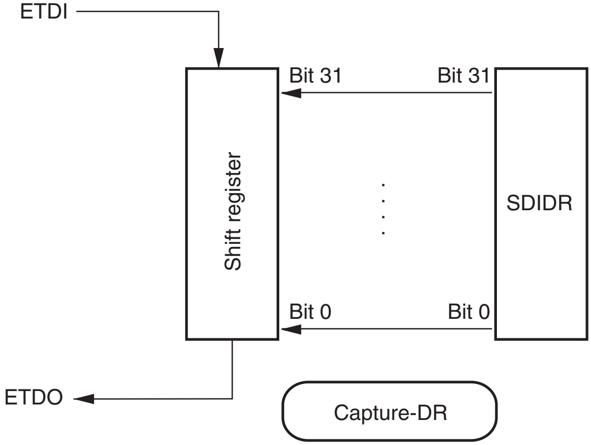


Figure 20.5 Serial Data Input/Output (2)

Section 21 Clock Pulse Generator

This LSI incorporates a clock pulse generator which generates the system clock (ϕ), internal clock, bus master clock, and subclock (ϕSUB). The clock pulse generator consists of an oscillator, PLL multiplier circuit, system clock select circuit, medium-speed clock divider, bus master clock select circuit, subclock input circuit, and subclock waveform shaping circuit. Figure 21.1 shows a block diagram of the clock pulse generator.

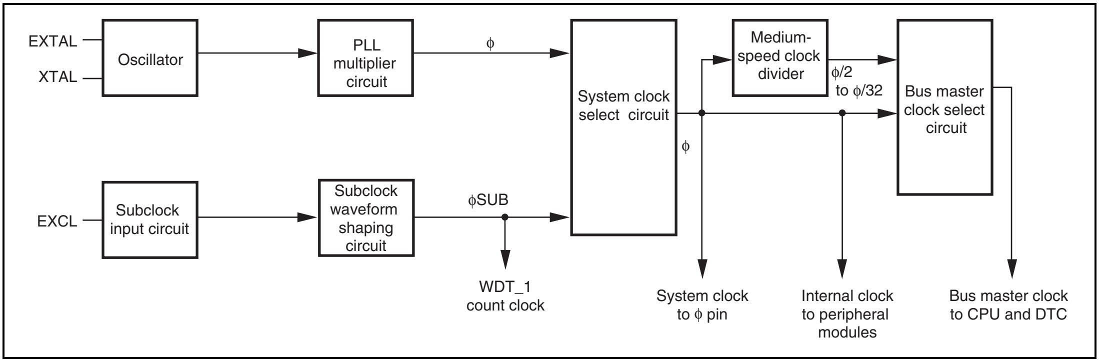

EXTAL
XTAL
Oscillator
PLL multiplier circuit
ϕ
System clock select circuit
Medium-speed clock divider
ϕ/2 to ϕ/32
Bus master clock select circuit
ϕ
EXCL
Subclock input circuit
Subclock waveform shaping circuit
ϕSUB
WDT_1 count clock
System clock to ϕ pin
Internal clock to peripheral modules
Bus master clock to CPU and DTC

Figure 21.1 Block Diagram of Clock Pulse Generator

The bus master clock is selected as either high-speed mode or medium-speed mode by software according to the settings of the SCK2 to SCK0 bits in the standby control register. Use of the medium-speed clock (ϕ/2 to ϕ/32) may be limited during CPU operation and when accessing the internal memory of the CPU. The operation speed of the DTC and the external space access cycle are thus stabilized regardless of the setting of medium-speed mode. For details on the standby control register, see section 22.1.1, Standby Control Register (SBYCR).

The subclock input is controlled by software according to the EXCLE bit setting in the low power control register. For details on the low power control register, see section 22.1.2, Low-Power Control Register (LPWRCR).

21.1 Oscillator

Clock pulses can be supplied either by connecting a crystal resonator or by providing external clock input.

21.1.1 Connecting Crystal Resonator

Figure 21.2 shows a typical method of connecting a crystal resonator. An appropriate damping resistance R_d, given in table 21.1, should be used. An AT-cut parallel-resonance crystal resonator should be used.

Figure 21.3 shows the equivalent circuit of a crystal resonator. A crystal resonator having the characteristics given in table 21.2 should be used.

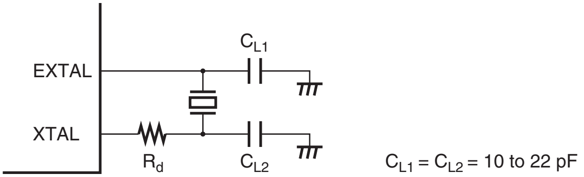


Figure 21.2 Typical Connection to Crystal Resonator

Table 21.1 Damping Resistance Values

Frequency (MHz)	5	6.25
R_d (Ω)	300	240

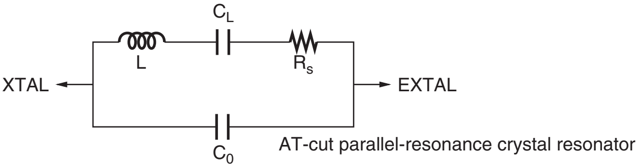


Figure 21.3 Equivalent Circuit of Crystal Resonator

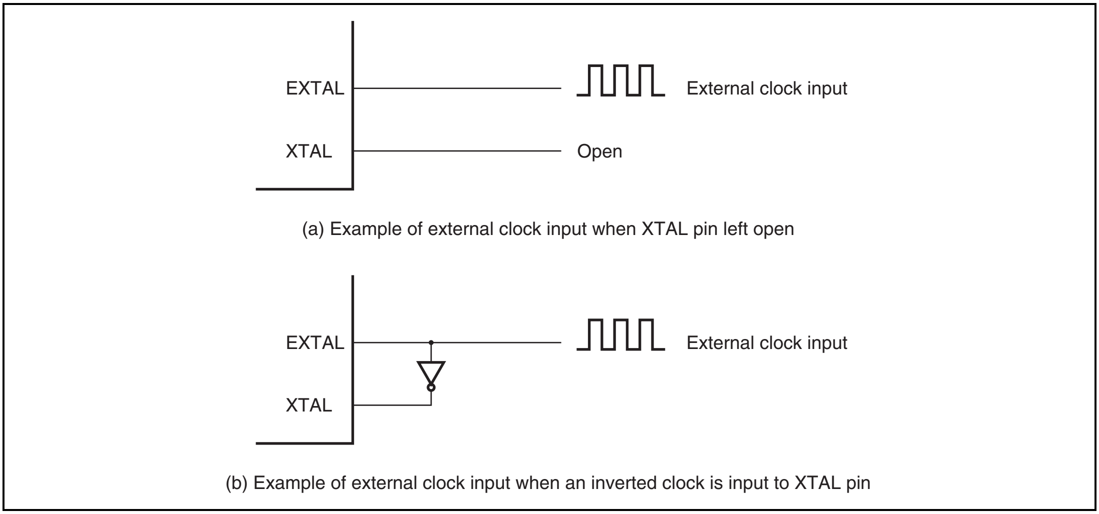
Table 21.2 Crystal Resonator Parameters

Frequency(MHz)	5	6.25
R_S (max) (Ω)	100	240
C_0 (max) (pF)	7	7

21.1.2 External Clock Input Method

Figure 21.4 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 tied to high in standby mode.



(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 21.4 Example of External Clock Input

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. For the external clock output stabilization delay time, refer to table 24.5 and figure 24.8 in section 24, Electrical Characteristics.

21.2 PLL Multiplier Circuit

The PLL multiplier circuit generates a clock of 4 times the frequency of its input clock. The frequency ranges of the multiplied clock are shown in table 21.5.

Table 21.3 Ranges of Multiplied Clock Frequency

	Input Clock (MHz)	Multiplier	System Clock (MHz)
Crystal Resonator, External Clock	5 to 6.25	4	20 to 25

21.3 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.

21.4 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.

21.5 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 P56DDR bit in P5DDR should be cleared to 0, and the EXCLE bit in LPWRCR should be set to 1.

When the subclock is not used, subclock input should not be enabled.

21.6 Subclock Waveform Shaping 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.

21.7 Clock Select Circuit

The clock select circuit selects the system clock that is used in this LSI.

A clock generated by the oscillator, to which the EXTAL and XTAL pins are input, and multiplied by the PLL circuit is selected as a system clock when returning from high-speed mode, medium-speed mode, sleep mode, the reset state, or standby mode.

21.8 Usage Notes

21.8.1 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 oscillation pins do not exceed the maximum rating.

21.8.2 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 EXTAL and XTAL pins. Other signal lines should be routed away from the oscillation circuit to prevent inductive interference with the correct oscillation as shown in figure 21.5.

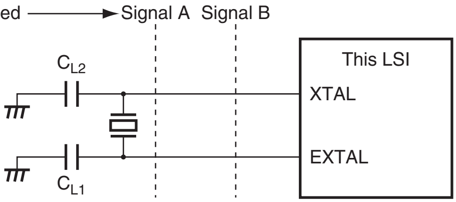


Figure 21.5 Note on Board Design of Oscillation Circuit Section

21.8.3 Note on Operation Check

This LSI may oscillate at several kHz of frequency even when a crystal resonator is not connected to the EXTAL and XTAL pins or an external clock is not input. Use this LSI after confirming that the LSI operates with appropriate frequency.

Section 22 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 φ/2, φ/4, φ/8, φ/16,or φ/32.
- Sleep mode

  The CPU stops but on-chip peripheral modules continue 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.

22.1 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)
- Module stop control register A (MSTPCRA)
- Sub-chip module stop control register BH, BL (SUBMSTPBH, SUBMSTPBL)

22.1.1 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 Note that the SSBY bit is not changed even if a mode transition occurs by an interrupt.

Bit	Bit Name	Initial Value	R/W	Description
6 5 4	STS2 STS1 STS0	0 0 0	R/W R/W R/W	Standby Timer Select 2 to 0 Select the wait time for clock settling from clock oscillation start when canceling software standby mode. Select a wait time of 8 ms (oscillation settling time) or more, depending on the operating frequency. With an external clock, select a wait time of 500 μs (external clock output settling delay time) or more, depending on the operating frequency. Table 22.1 shows the relationship between the STS2 to STS0 values and wait time.
3	DTSPEED	0	R/W	DTC Speed Specifies the operating clock for the bus masters (DTC) other than the CPU in medium-speed mode. 0: All bus masters operate based on the medium-speed clock. 1: The DTC operates based on the system clock. The operating clock is changed when a DTC transfer is requested even if the CPU operates based on the medium-speed clock.
2 1 0	SCK2 SCK1 SCK0	0 0 0	R/W R/W R/W	System Clock Select 2 to 0 Select a clock for the bus master in high-speed mode or medium-speed mode. 000: High-speed mode (Initial value) 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: Must not be set.

[Legend]

x: Don't care

Table 22.1 Operating Frequency and Wait Time

STS2	STS1	STS0	Wait Time	25M Hz	20 MHz	Unit
0	0	0	8192 states	0.3	0.4	ms
0	0	1	16384 states	0.7	0.8	
0	1	0	32768 states	1.3	1.6	
0	1	1	65536 states	2.6	3.3	
1	0	0	131072 states	5.2	6.6	
1	0	1	262144 states	10.5	13.1	
1	1	x	Reserved*	—	—	

░ Recommended specification

Note: * Setting prohibited.

[Legend] x: Don't care

22.1.2 Low-Power Control Register (LPWRCR)

LPWRCR controls power-down modes.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	0	R/W	Reserved The initial value should not be changed.
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. 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 to 0	—	All 0	R/W	Reserved The initial value should not be changed.

22.1.3 Module Stop Control Registers H, L, and A (MSTPCRH, MSTPCRL, MSTPCRA)

MSTPCR specifies on-chip peripheral modules to shift to module stop mode in module units. Each module can enter module stop mode by setting the corresponding bit to 1.

- MSTPCRH

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	MSTP15	0	R/W	Reserved The initial value should not be changed.
6	MSTP14	0	R/W	Data transfer controller (DTC)
5	MSTP13	1	R/W	16-bit free-running timer (FRT)
4	MSTP12	1	R/W	8-bit timers (TMR_0, TMR_1)
3	MSTP11	1	R/W	14-bit PWM timer (PWMX)
2	MSTP10	1	R/W	Reserved The initial value should not be changed.
1	MSTP9	1	R/W	A/D converter
0	MSTP8	1	R/W	8-bit timers (TMR_X, TMR_Y)

- MSTPCRL

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7	MSTP7	1	R/W	Serial communication interface 3 (SCI_3)
6	MSTP6	1	R/W	Serial communication interface 1 (SCI_1)
5	MSTP5	1	R/W	Reserved The initial value should not be changed.
4	MSTP4	1	R/W	I^{2}C bus interface channel 0 (IIC_0)
3	MSTP3	1	R/W	I^{2}C bus interface channel 1 (IIC_1)
2	MSTP2	1	R/W	I^{2}C bus interface channel 2, 3 (IIC_2, IIC_3)
1	MSTP1	1	R/W	CRC operation circuit
0	MSTP0	1	R/W	Reserved The initial value should not be changed.

- MSTPCRA

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7 to 3	MSTPA7 to MSTPA3	All 0	R/W	Reserved The initial values should not be changed.
2	MSTPA2	0	R/W	14-bit PWM timer (PWMX_1)
1	MSTPA1	0	R/W	14-bit PWM timer (PWMX_0)
0	MSTPA0	0	R/W	Reserved The initial value should not be changed.

MSTPCR sets operation and stop by the combination of bits as follows:

MSTPCRH (bit 3) MSTP11	MSTPCRA (bit 2) MSTPA2	Function
0	0	14-bit PWM timer (PWMX_1) operates.
0	1	14-bit PWM timer (PWMX_1) stops.
1	x	

MSTPCRH (bit 3) MSTP11	MSTPCRA (bit 1) MSTPA1	Function
0	0	14-bit PWM timer (PWMX_0) operates.
0	1	14-bit PWM timer (PWMX_0) stops.
1	x	

Note: Bit 3 of MSTPCRH is the module stop bit for PWMX_0 and PWMX_1.

[Legend] x: Don't care

22.1.4 Sub-Chip Module Stop Control Registers BH, BL (SUBMSTPBH, SUBMSTPBL)

SUBMSTPB specifies 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.

- SUBMSTPBH

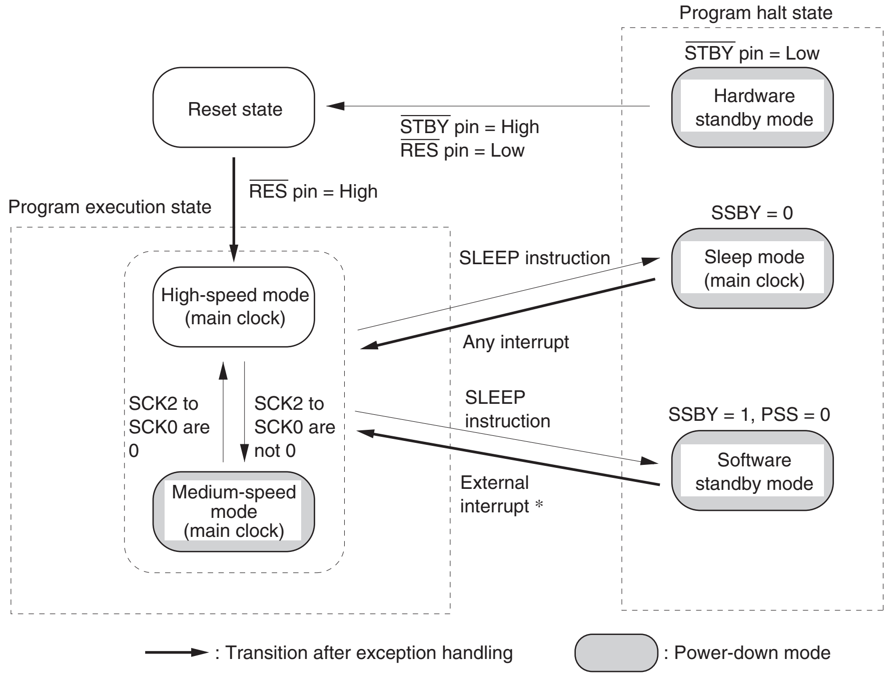
Bit	Bit Name	Initial Value	R/W	Corresponding Module
7 to 0	SMSTPB15 to SMSTPB8	All 1	R/W	Reserved The initial values should not be changed.

- SUBMSTPBL

Bit	Bit Name	Initial Value	R/W	Corresponding Module
7 to 2	SMSTPB7 to SMSTPB2	All 1	R/W	Reserved The initial values should not be changed.
1	SMSTPB1	1	R/W	LPC interface (LPC)
0	SMSTPB0	1	R/W	Reserved The initial value should not be changed.

22.2 Mode Transitions and LSI States

Figure 22.1 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 22.2 shows the LSI internal states in each operating mode.



Note: 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.

* NMI, IRQ0 to IRQ5

Figure 22.1 Mode Transition Diagram

Table 22.2 LSI Internal States in Each Mode

Function		High-Speed	Medium-Speed	Sleep	Module Stop	Software Standby	Hardware Standby
System clock pulse generator		Functioning	Functioning	Functioning	Functioning	Halted	Halted
Subclock pulse generator		Functioning	Functioning	Functioning	Functioning	Halted	Halted
CPU	Instruction execution	Functioning	Functioning in medium-speed mode	Halted	Functioning	Halted	Halted
	Registers			Retained		Retained	Undefined
External interrupts	NMI	Functioning	Functioning	Functioning	Functioning	Functioning	Halted
	IRQ0 to IRQ15						
Peripheral modules	DTC	Functioning	Functioning in medium-speed mode/ Functioning	Functioning	Functioning/ Halted (retained)	Halted (retained)	Halted (reset)
	WDT_1		Functioning		Functioning		
	WDT_0						
	TMR_0, TMR_1				Functioning/ Halted (retained)		
	LPC						
	FRT						
	TMR_X, TMR_Y						
	IIC_0 to IIC_3						
	CRC						
	D/A converter						
	SCI_1, SCI_3				Functioning/ Halted (retained/reset)	Halted (retained/reset)	Halted (reset)
	PWMX_0, PWMX_1				Functioning/ Halted (reset)	Halted (reset)	
	A/D converter						
	RAM			Functioning (DTC)	Functioning	Retained	Retained
	I/O			Functioning			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).

22.3 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 (ϕ/2, ϕ/4, ϕ/8, ϕ/16, or ϕ/32) specified by the SCK2 to SCK0 bits. The bus masters other than the CPU (DTC) also operate in medium-speed mode when the DTSPEED bit in SBYCR is cleared to 0. On-chip peripheral modules other than the bus masters always operate on the system clock (ϕ).

When the DTSPEED bit in SBYCR is set to 1, the ϕ clock can be used as the DTC operating clock.

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if ϕ/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, 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, 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, medium-speed mode is cancelled and operation shifts to the reset state. The same applies in the case of a reset caused by overflow of the watchdog timer.

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Figure 22.2 shows an example of medium-speed mode timing.

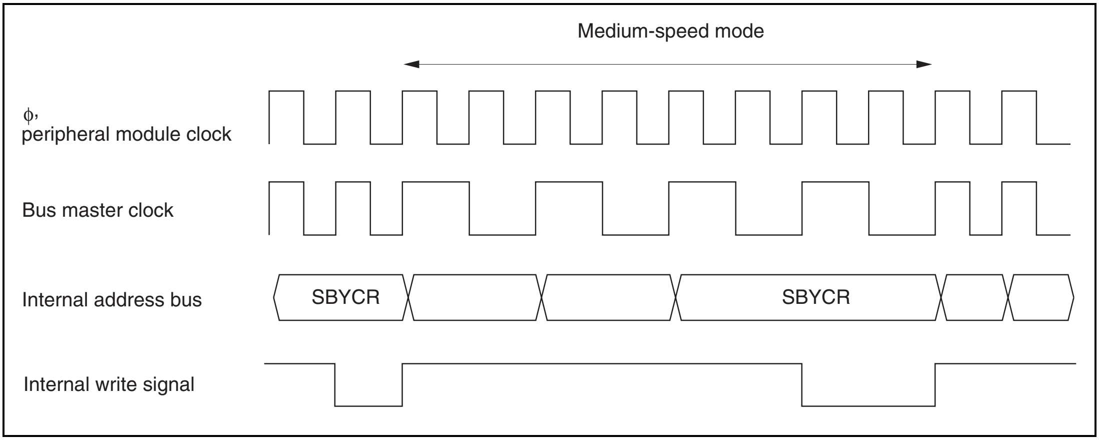


Figure 22.2 Medium-Speed Mode Timing

22.4 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. 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 settling time has passed, driving the $\overline{\text{RES}}$ pin high causes the CPU to start reset exception handling.

When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.

22.5 Software Standby Mode

The CPU makes a transition to software standby mode when the SLEEP instruction is executed with the SSBY bit in SBYCR set to 1, and the PSS bit in TCSR (WDT_1) 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 registers, on-chip RAM data, I/O ports, and the states of on-chip peripheral modules other than the PWMX, A/D converter, and part of the SCI are retained as long as the prescribed voltage is supplied.

Software standby mode is cleared by an external interrupt (NMI, IRQ0 to IRQ15), the $\overline{\text{RES}}$ pin input, or $\overline{\text{STBY}}$ pin input.

When an external interrupt request signal is input, system clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SBYCR, software standby mode is cleared, and interrupt exception handling is started. When exiting software standby mode by IRQ0 to IRQ15 interrupt, set the corresponding enable bit to 1 and ensure that any interrupt with a higher priority than IRQ0 to IRQ15 is not generated. Software standby mode is not exited if the corresponding enable bit is cleared to 0 or if the interrupt has been masked by the CPU.

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 settles. When the $\overline{\text{RES}}$ pin goes high after clock oscillation settles, 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 22.3 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.

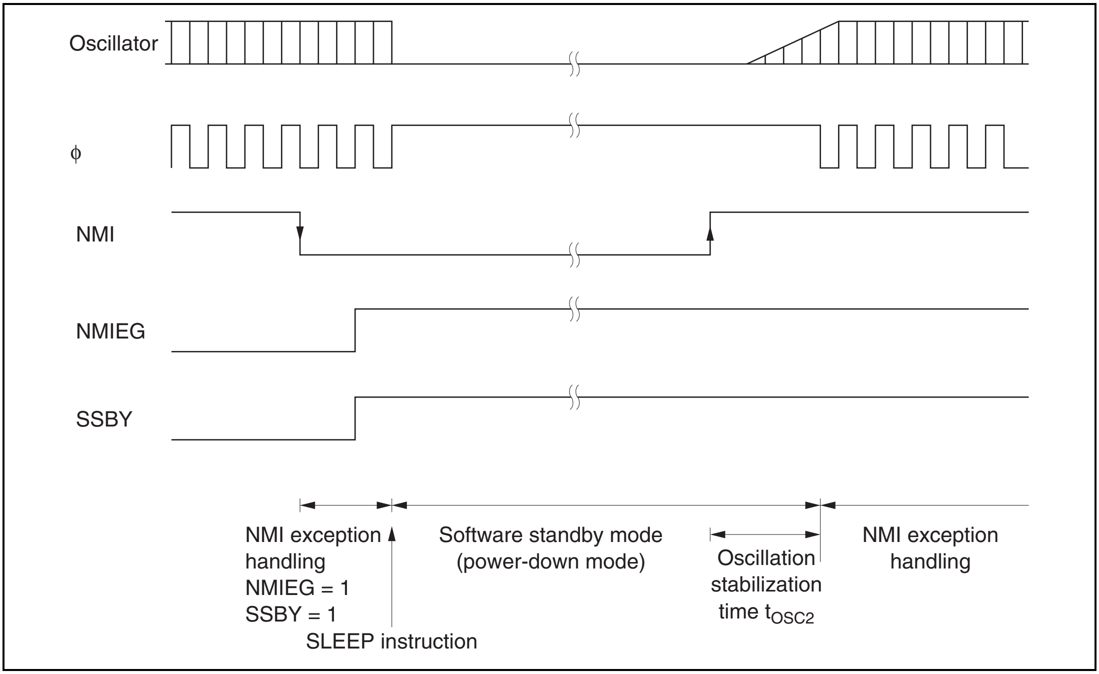


Figure 22.3 Software Standby Mode Application Example

22.6 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 ($\overline{\text{MD2}}$ and MD1) 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 settles. When the $\overline{\text{RES}}$ pin is subsequently driven high after the clock oscillation settling time has passed, reset exception handling starts.

Figure 22.4 shows an example of hardware standby mode timing.

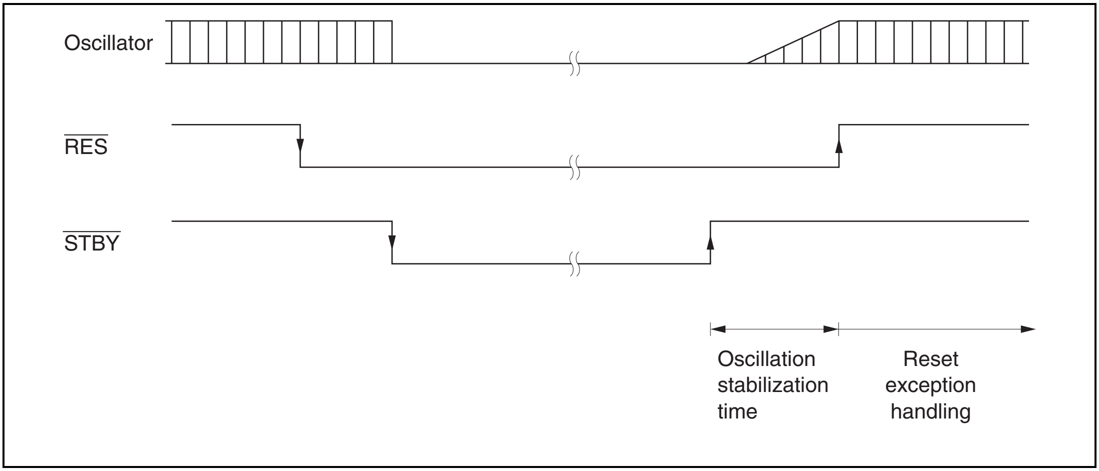


Figure 22.4 Hardware Standby Mode Timing

22.7 Module Stop Mode

Module stop mode can be individually set for each on-chip peripheral module.

When the corresponding MSTP bit in MSTPCR and SUBMSTP 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 on-chip peripheral modules other than the PWMX, A/D converter, and part of the SCI are retained.

After the reset state is cancelled, all modules other than DTC are in module stop mode.

While an on-chip peripheral module is in module stop mode, read/write access to its registers is disabled.

22.8 Usage Notes

22.8.1 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.

22.8.2 Current Consumption when Waiting for Oscillation Settling

The current consumption increases during oscillation settling.

22.8.3 DTC Module Stop Mode

If the DTC module stop mode specification and DTC bus request occur simultaneously, the bus is released to the DTC and the MSTP bit cannot be set to 1. After completing the DTC bus cycle, set the MSTP bit to 1 again.

22.8.4 Notes on Subclock Usage

When using the subclock, make a transition to power-down mode after setting the EXCLE bit in LPWRCR to 1 and loading the subclock two or more cycles. When not using the subclock, the EXCLE bit should not be set to 1.

Section 23 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.

23.1 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.

Note: Access to undefined or reserved addresses is prohibited. Since operation or continued operation is not guaranteed when these registers are accessed, do not attempt such access.

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Host interface control register_4	HICR4	8	H'FD00	LPC	16	2
BT status register 0	BTSR0	8	H'FD02	LPC	16	2
BT status register 1	BTSR1	8	H'FD03	LPC	16	2
BT control/status register 0	BTCSR0	8	H'FD04	LPC	16	2
BT control/status register 1	BTCSR1	8	H'FD05	LPC	16	2
BT control register	BTCR	8	H'FD06	LPC	16	2
BT interrupt mask register	BTIMSR	8	H'FD07	LPC	16	2
SMIC flag register	SMICFLG	8	H'FD08	LPC	16	2
SMIC control/status register	SMICCSR	8	H'FD0A	LPC	16	2
SMIC data register	SMICDTR	8	H'FD0B	LPC	16	2
SMIC interrupt register 0	SMICIR0	8	H'FD0C	LPC	16	2
SMIC interrupt register 1	SMICIR1	8	H'FD0E	LPC	16	2
Bidirectional data register 0MW	TWR0MW	8	H'FD10	LPC	16	2
Bidirectional data register 0SW	TWR0SW	8	H'FD10	LPC	16	2
Bidirectional data register 1	TWR1	8	H'FD11	LPC	16	2
Bidirectional data register 2	TWR2	8	H'FD12	LPC	16	2
Bidirectional data register 3	TWR3	8	H'FD13	LPC	16	2
Bidirectional data register 4	TWR4	8	H'FD14	LPC	16	2
Bidirectional data register 5	TWR5	8	H'FD15	LPC	16	2
Bidirectional data register 6	TWR6	8	H'FD16	LPC	16	2
Bidirectional data register 7	TWR7	8	H'FD17	LPC	16	2
Bidirectional data register 8	TWR8	8	H'FD18	LPC	16	2
Bidirectional data register 9	TWR9	8	H'FD19	LPC	16	2
Bidirectional data register 10	TWR10	8	H'FD1A	LPC	16	2
Bidirectional data register 11	TWR11	8	H'FD1B	LPC	16	2
Bidirectional data register 12	TWR12	8	H'FD1C	LPC	16	2
Bidirectional data register 13	TWR13	8	H'FD1D	LPC	16	2
Bidirectional data register 14	TWR14	8	H'FD1E	LPC	16	2
Bidirectional data register 15	TWR15	8	H'FD1F	LPC	16	2
Input data register 3	IDR3	8	H'FD20	LPC	16	2
Output data register 3	ODR3	8	H'FD21	LPC	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Status register 3	STR3	8	H'FD22	LPC	16	2
SERIRQ control register 4	SIRQCR4	8	H'FD23	LPC	16	2
LPC channel 3 address register H	LADR3H	8	H'FD24	LPC	16	2
LPC channel 3 address register L	LADR3L	8	H'FD25	LPC	16	2
SERIRQ control register 0	SIRQCR0	8	H'FD26	LPC	16	2
SERIRQ control register 1	SIRQCR1	8	H'FD27	LPC	16	2
Input data register 1	IDR1	8	H'FD28	LPC	16	2
Output data register 1	ODR1	8	H'FD29	LPC	16	2
Status register 1	STR1	8	H'FD2A	LPC	16	2
SERIRQ control register 5	SIRQCR5	8	H'FD2B	LPC	16	2
Input data register 2	IDR2	8	H'FD2C	LPC	16	2
Output data register 2	ODR2	8	H'FD2D	LPC	16	2
Status register 2	STR2	8	H'FD2E	LPC	16	2
Host interface select register	HISEL	8	H'FD2F	LPC	16	2
Host interface control register 0	HICR0	8	H'FD30	LPC	16	2
Host interface control register 1	HICR1	8	H'FD31	LPC	16	2
Host interface control register 2	HICR2	8	H'FD32	LPC	16	2
Host interface control register 3	HICR3	8	H'FD33	LPC	16	2
SERIRQ control register2	SIRQCR2	8	H'FD34	LPC	16	2
BT data buffer	BTDTR	8	H'FD35	LPC	16	2
BT FIFO valid size register 0	BTFVSR0	8	H'FD36	LPC	16	2
BT FIFO valid size register 1	BTFVSR1	8	H'FD37	LPC	16	2
LPC channel 1, 2 address register H	LADR12H	8	H'FD38	LPC	16	2
LPC channel 1, 2 address register L	LADR12L	8	H'FD39	LPC	16	2
Sub-chip module stop control register BH	SUBMSTPBH	8	H'FE3E	SYSTEM	8	2
Sub-chip module stop control register BL	SUBMSTPBL	8	H'FE3F	SYSTEM	8	2
Event count status register	ECS	16	H'FE40	EVC	16	2
Event count control register	ECCR	8	H'FE42	EVC	8	2
Module stop control register A	MSTPCRA	8	H'FE43	SYSTEM	8	2
Noise canceler enable register	P3NCE	8	H'FE44	PORT	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Noise canceler mode control register	P3NCMC	8	H'FE45	PORT	8	2
Noise canceler cycle setting register	NCCS	8	H'FE46	PORT	8	2
Port E output data register	PEODR	8	H'FE48	PORT	8	2
Port E input data register	PEPIN	8	H'FE4A	PORT	8	2
Port E data direction register	PEDDR	8	H'FE4A	PORT	8	2
Port C output data register	PCODR	8	H'FE4C	PORT	8	2
Port C input data register	PCPIN	8	H'FE4C	PORT	8	2
Port C data direction register	PCDDR	8	H'FE4E	PORT	8	2
Flash code control status register	FCCS	8	H'FE88	FLASH	8	2
Flash program code select register	FPCS	8	H'FE89	FLASH	8	2
Flash erase code select register	FECS	8	H'FE8A	FLASH	8	2
Flash key code register	FKEY	8	H'FE8C	FLASH	8	2
Flash MAT select register	FMATS	8	H'FE8D	FLASH	8	2
Flash transfer destination address register	FTDAR	8	H'FE8E	FLASH	8	2
Serial mode register_1	SMR_1	8	H'FE98	SCI_1	8	2
Bit rate register_1	BRR_1	8	H'FE99	SCI_1	8	2
Serial control register_1	SCR_1	8	H'FE9A	SCI_1	8	2
Transmit data register_1	TDR_1	8	H'FE9B	SCI_1	8	2
Serial status register_1	SSR_1	8	H'FE9C	SCI_1	8	2
Receive data register_1	RDR_1	8	H'FE9D	SCI_1	8	2
Smart card mode register_1	SCMR_1	8	H'FE9E	SCI_1	8	2
A/D data register A	ADDRA	16	H'FEA0	ADC	16	2
A/D data register B	ADDRB	16	H'FEA2	ADC	16	2
A/D data register C	ADDRC	16	H'FEA4	ADC	16	2
A/D data register D	ADDRD	16	H'FEA6	ADC	16	2
A/D data register E	ADDRE	16	H'FEA8	ADC	16	2
A/D data register F	ADDRF	16	H'FEAA	ADC	16	2
A/D data register G	ADDRG	16	H'FEAC	ADC	16	2
A/D data register H	ADDRH	16	H'FEAE	ADC	16	2
A/D control/status register	ADCSR	8	H'FEB0	ADC	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
A/D control register	ADCR	8	H'FEB1	ADC	8	2
Noise canceler enable register	P4NCE	8	H'FEBA	PORT	8	2
Noise canceler mode control register	P4NCMC	8	H'FEBB	PORT	8	2
Port 6 pull-up MOS control register	P6PCR	8	H'FEBC	PORT	8	2
Port 4 pull-up MOS control register	P4PCR	8	H'FEBF	PORT	8	2
I^2C bus control register_3	ICCR_3	8	H'FEC0	IIC_3	8	2
I^2C bus status register_3	ICSR_3	8	H'FEC1	IIC_3	8	2
I^2C bus data register_3	ICDR_3	8	H'FEC2	IIC_3	8	2
Second slave address register_3	SARX_3	8	H'FEC2	IIC_3	8	2
I^2C bus mode register_3	ICMR_3	8	H'FEC3	IIC_3	8	2
Slave address register_3	SAR_3	8	H'FEC3	IIC_3	8	2
I^2C bus control register_2	ICCR_2	8	H'FEC8	IIC_2	8	2
I^2C bus status register_2	ICSR_2	8	H'FEC9	IIC_2	8	2
I^2C bus data register_2	ICDR_2	8	H'FECA	IIC_2	8	2
Second slave address register_2	SARX_2	8	H'FECA	IIC_2	8	2
I^2C bus mode register_2	ICMR_2	8	H'FECB	IIC_2	8	2
Slave address register_2	SAR_2	8	H'FECB	IIC_2	8	2
PWMX (D/A) data register A_1	DADRA_1	16	H'FECC	PWMX_1	8	4
PWMX (D/A) control register_1	DACR_1	8	H'FECC	PWMX_1	8	2
PWMX (D/A) data register B_1	DADRB_1	16	H'FECE	PWMX_1	8	4
PWMX (D/A) counter_1	DACNT_1	16	H'FECE	PWMX_1	8	4
CRC control register	CRCCR	8	H'FED4	CRC	16	2
CRC data input register	CRCDIR	8	H'FED5	CRC	16	2
CRC data output register	CRCDOR	16	H'FED6	CRC	16	2
I^2C bus control extended register_0	ICXR_0	8	H'FED8	IIC_0	8	2
I^2C bus control extended register_1	ICXR_1	8	H'FED9	IIC_1	8	2
I^2C SMBus control register	ICSMBCR	8	H'FEDB	IIC	8	2
I^2C bus control extended register_2	ICXR_2	8	H'FEDC	IIC_2	8	2
I^2C bus control extended register_3	ICXR_3	8	H'FEDD	IIC_3	8	2
I^2C bus transfer select register	IICX3	8	H'FEDF	IIC	8	2
Keyboard comparator control register	KBCOMP	8	H'FEE4	EVC	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Interrupt control register D	ICRD	8	H'FEE7	INT	8	2
Interrupt control register A	ICRA	8	H'FEE8	INT	8	2
Interrupt control register B	ICRB	8	H'FEE9	INT	8	2
Interrupt control register C	ICRC	8	H'FEEA	INT	8	2
IRQ status register	ISR	8	H'FEEB	INT	8	2
IRQ sense control register H	ISCRH	8	H'FEEC	INT	8	2
IRQ sense control register L	ISCRL	8	H'FEED	INT	8	2
DTC enable register A	DTCERA	8	H'FEEE	DTD	8	2
DTC enable register B	DTCERB	8	H'FEEF	DTC	8	2
DTC enable register C	DTCERC	8	H'FEF0	DTC	8	2
DTC enable register D	DTCERD	8	H'FEF1	DTC	8	2
DTC enable register E	DTCERE	8	H'FEF2	DTC	8	2
DTC vector register	DTVECR	8	H'FEF3	DTC	8	2
Address break control register	ABRKCR	8	H'FEF4	INT	8	2
Break address register A	BARA	8	H'FEF5	INT	8	2
Break address register B	BARB	8	H'FEF6	INT	8	2
Break address register C	BARC	8	H'FEF7	INT	8	2
IRQ enable register 16	IER16	8	H'FEF8	INT	8	2
IRQ status register 16	ISR16	8	H'FEF9	INT	8	2
IRQ sense control register 16H	ISCR16H	8	H'FEEA	INT	8	2
IRQ sense control register 16L	ISCR16L	8	H'FEFB	INT	8	2
IRQ sense port select register 16	ISSR16	8	H'FEFC	PORT	8	2
IRQ sense port select register	ISSR	8	H'FEFD	PORT	8	2
Peripheral clock select register	PCSR	8	H'FF82	PWM	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
I^{2}C bus control register_1	ICCR_1	8	H'FF88	IIC_1	8	2
I^{2}C bus status register_1	ICSR_1	8	H'FF89	IIC_1	8	2
I^{2}C bus data register_1	ICDR_1	8	H'FF8E	IIC_1	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
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	FRC	16	H'FF92	FRT	16	2
Output Compare register A	OCRA	16	H'FF94	FRT	16	2
Output Compare register B	OCRB	16	H'FF94	FRT	16	2
Timer control register	TCR	8	H'FF96	FRT	16	2
Timer output compare control register	TOCR	8	H'FF97	FRT	16	2
Output Compare register AR	OCRAR	16	H'FF98	FRT	16	2
Output Compare register AF	OCRAF	16	H'FF9A	FRT	16	2
PWMX (D/A) data register A_0	DADRA_0	16	H'FFA0	PWMX_0	8	4
PWMX (D/A) control register_0	DACR_0	8	H'FFA0	PWMX_0	8	2
PWMX (D/A) data register B_0	DADRB_0	16	H'FFA6	PWMX_0	8	4
PWMX (D/A) counter_0	DACNT_0	16	H'FFA6	PWMX_0	8	4
Timer control/status register_0	TCSR_0	8	H'FFA8 (read)	WDT_0	16	2
Timer control/status register_0	TCSR_0	16	H'FFA8 (write)	WDT_0	16	2
Timer counter_0	TCNT_0	8	H'FFA9 (read)	WDT_0	16	2
Timer counter_0	TCNT_0	16	H'FFA8 (write)	WDT_0	16	2
Port A output data register	PAODR	8	H'FFAA	PORT	8	2
Port A input data register	PAPIN	8	H'FFAB (read)	PORT	8	2
Port A data direction register	PADDR	8	H'FFAB (write)	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

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
Port 2 data direction register	P2DDR	8	H'FFB1	PORT	8	2
Port 1 data register	P1DR	8	H'FFB2	PORT	8	2
Port 2 data register	P2DR	8	H'FFB3	PORT	8	2
Port 3 data direction register	P3DDR	8	H'FFB4	PORT	8	2
Port 4 data direction register	P4DDR	8	H'FFB5	PORT	8	2
Port 3 data register	P3DR	8	H'FFB6	PORT	8	2
Port 4 data register	P4DR	8	H'FFB7	PORT	8	2
Port 5 data direction register	P5DDR	8	H'FFB8	PORT	8	2
Port 6 data direction register	P6DDR	8	H'FFB9	PORT	8	2
Port 5 data register	P5DR	8	H'FFBA	PORT	8	2
Port 6 data register	P6DR	8	H'FFBB	PORT	8	2
Port 8 data direction register	P8DDR	8	H'FFBD	PORT	8	2
Port 7 input data register	P7PIN	8	H'FFBE (Read)	PORT	8	2
Port 8 data register	P8DR	8	H'FFBF	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
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	8	2
Time constant register A_0	TCORA_0	8	H'FFCC	TMR_0	8	2
Time constant register A_1	TCORA_1	8	H'FFCD	TMR_1	8	2
Time constant register B_0	TCORB_0	8	H'FFCE	TMR_0	8	2
Time constant register B_1	TCORB_1	8	H'FFCF	TMR_1	8	2
Timer counter_0	TCNT_0	8	H'FFD0	TMR_0	8	2
Timer counter_1	TCNT_1	8	H'FFD1	TMR_1	8	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

Register Name	Abbreviation	Number of Bits	Address	Module	Data Bus Width	Number of Access States
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
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
Serial mode register_3	SMR_3	8	H'FFE0	SCI_3	8	2
Bit rate register_3	BRR_3	8	H'FFE1	SCI_3	8	2
Serial control register_3	SCR_3	8	H'FFE2	SCI_3	8	2
Transmit data register_3	TDR_3	8	H'FFE3	SCI_3	8	2
Serial status register_3	SSR_3	8	H'FFE4	SCI_3	8	2
Receive data register_3	RDR_3	8	H'FFE5	SCI_3	8	2
Smart card mode register_3	SCMR_3	8	H'FFE6	SCI_3	8	2
Timer control/status register_1	TCSR_1	8	H'FFEA (read)	WDT_1	16	2
Timer control/status register_1	TCSR_1	16	H'FFEA (write)	WDT_1	16	2
Timer counter_1	TCNT_1	8	H’FFEB (read)	WDT_1	16	2
Timer counter_1	TCNT_1	16	H'FFEA (write)	WDT_1	16	2
Timer control register_X	TCR_X	8	H'FFF0	TMR_X	8	2
Timer control/status register_X	TCSR_X	8	H'FFF1	TMR_X	8	2
Timer counter_X	TCNT_X	8	H'FFF4	TMR_X	8	2
Time constant register A_X	TCORA_X	8	H'FFF6	TMR_X	8	2
Time constant register B_X	TCORB_X	8	H'FFF7	TMR_Y	8	2
Timer control register_Y	TCR_Y	8	H'FFF0	TMR_Y	8	2
Timer control/status register_Y	TCSR_Y	8	H'FFF1	TMR_Y	8	2
Time constant register A_Y	TCORA_Y	8	H'FFF2	TMR_Y	8	2
Time constant register B_Y	TCORB_Y	8	H'FFF3	TMR_Y	8	2
Timer counter_Y	TCNT_Y	8	H'FFF4	TMR_Y	8	2
Timer connection register S	TCONRS	8	H'FFFE	TMR	8	2

23.2 Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, so 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
HICR4	LADR12SEL	—	—	—	—	—	—	—	LPC
BTSR0	—	—	—	FRDI	HRDI	HWRI	HBTWI	HBTRI	
BTSR1	—	HRSTI	IRQCRI	BEVTI	B2HI	H2BI	CRRPI	CRWPI	
BTCSR0	—	FSEL1	FSEL0	FRDIE	HRDIE	HWRIE	HBTWIE	HBTRIE	
BTCSR1	RSTRENBL	HRSTIE	IRQCRIE	BEVTIE	B2HIE	H2BIE	CRRPIE	CRWPIE	
BTCR	B_BUSY	H_BUSY	OEM0	BEVT_ATN	B2H_ATN	H2B_ATN	CLR_RD_ PTR	CLR_WR_ PTR	
BTIMSR	BMC_ HWRST	—	—	OEM3	OEM2	OEM1	B2H_IRQ	B2H_IRQ_ EN	
SMICFLG	RX_DATA_ RDY	TX_DATA_ RDY	—	SMI	SEVT_ATN	SMS_ATN	—	BUSY	
SMICCSR	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
SMICDTR	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
SMICIR0	—	—	—	HDTWI	HDTRI	STARI	CTLWI	BUSYI	
SMICIR1	—	—	—	HDTWIE	HDTRIE	STARIE	CTLWIE	BUSYIE	
TWR0MW	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
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	

Register									
Abbreviation	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Module**
TWR11	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	LPC
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}}$3	DBU32	IBF3A	OBF3A	
SIRQCR4	IRQ15E	IRQ14E	IRQ13E	IRQ8E	IRQ7E	IRQ5E	IRQ4E	IRQ3E	
LADR3H	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	
LADR3L	bit 7	bit 6	bit 5	bit 4	bit 3	—	bit 1	TWRE	
SIRQCR0	Q/$\overline{\text{C}}$	SELREQ	IEDIR2	SMIE3B	SMIE3A	SMIE2	IRQ12E0	IRQ1E0	
SIRQCR1	IRQ11E3	IRQ10E3	IRQ9E3	IRQ6E3	IRQ11E2	IRQ10E2	IRQ9E2	IRQ6E2	
IDR1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ODR1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
STR1	DBU17	DBU16	DBU15	DBU14	C/$\overline{\text{D}}$0	DBU12	IBF1	OBF1	
SIRQCR5	SELIRQ15	SELIRQ14	SELIRQ13	SELIRQ8	SELIRQ7	SELIRQ5	SELIRQ4	SELIRQ3	
IDR2	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ODR2	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
STR2	DBU27	DBU26	DBU25	DBU24	C/$\overline{\text{D}}$2	DBU22	IBF2	OBF2	
HISEL	SELSTR3	SELIRQ11	SELIRQ10	SELIRQ9	SELIRQ6	SELSMI	SELIRQ12	SELIRQ1	
HICR0	—	LPC2E	LPC1E	—	SDWNE	—	—	—	
HICR1	LPCBSY	CLKREQ	IRQBSY	LRSTB	SDWNB	—	—	—	
HICR2	—	LRST	—	ABRT	—	IBFIE2	IBFIE1	ERRIE	
HICR3	LFRAME	—	SERIRQ	LRESET	—	—	—	—	
SIRQCR2	IEDIR3	—	—	—	—	—	—	—	
BTDTR	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
BTFVSR0	N7	N6	N5	N4	N3	N2	N1	N0	
BTFVSR1	N7	N6	N5	N4	N3	N2	N1	N0	
LADR12H	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
LADR12L	bit7	bit6	bit5	bit4	bit3	—	bit1	bit0	
SUBMSTPBH	SMSTPB15	SMSTPB14	SMSTPB13	SMSTPB12	SMSTPB11	SMSTPB10	SMSTPB9	SMSTPB8	SYSTEM
SUBMSTPBL	SMSTPB7	SMSTPB6	SMSTPB5	SMSTPB4	SMSTPB3	SMSTPB2	SMSTPB1	SMSTPB0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
ECS	—	—	—	—	—	—	—	—	EVC
	E7	E6	E5	E4	E3	E2	E1	E0	
ECCR	EDSB	—	—	—	ECSB3	ECSB2	ECSB1	ECSB0	
MSTPCRA	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0	SYSTEM
P3NCE	P37NCE	P36NCE	P35NCE	P34NCE	P33NCE	P32NCE	P31NCE	P30NCE	PORT
P3NCMC	P37NCMC	P36NCMC	P35NCMC	P34NCMC	P33NCMC	P32NCMC	P31NCMC	P30NCMC	
NCCS	—	—	—	—	—	NCCK2	NCCK1	NCCK0	
PEODR	PE7ODR	PE6ODR	PE5ODR	PE4ODR	PE3ODR	PE2ODR	PE1ODR	PE0ODR	
PEPIN	PE7PIN	PE6PIN	PE5PIN	PE4PIN	PE3PIN	PE2PIN	PE1PIN	PE0PIN	
PEDDR	PE7DDR	PE6DDR	PE5DDR	PE4DDR	PE3DDR	PE2DDR	PE1DDR	PE0DDR	
PCODR	PC7ODR	PC6ODR	PC5ODR	PC4ODR	PC3ODR	PC2ODR	PC1ODR	PC0ODR	
PCPIN	PC7PIN	PC6PIN	PC5PIN	PC4PIN	PC3PIN	PC2PIN	PC1PIN	PC0PIN	
PCDDR	PC7DDR	PC6DDR	PC5DDR	PC4DDR	PC3DDR	PC2DDR	PC1DDR	PC0DDR	
FCCS	FWE	—	—	FLER	WEINTE	—	—	SCO	FLASH
FPCS	—	—	—	—	—	—	—	PPVS	
FECS	—	—	—	—	—	—	—	EPVB	
FKEY	K7	K6	K5	K4	K3	K2	K1	K0	
FMATS	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0	
FTDAR	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0	
SMR_1*	C/$\overline{\text{A}}$ (GM)	CHR (BLK)	PE (PE)	O/$\overline{\text{E}}$ (O/$\overline{\text{E}}$)	STOP (BCP1)	MP (BCP0)	CKS1 (CKS1)	CKS0 (CKS0)	SCI_1
BRR_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SCR_1	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SSR_1*	TDRE (TDRE)	RDRF (RDRF)	ORER (ORER)	FER (ERS)	PER (PER)	TEND (TEND)	MPB (MPB)	MPBT (MPBT)	
RDR_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SCMR_1	—	—	—	—	SDIR	SINV	—	SMIF	
ADDRA	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	ADC
	AD1	AD0	—	—	—	—	—	—	
ADDRB	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRC	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRD	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
ADDRE	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	ADC
	AD1	AD0	—	—	—	—	—	—	
ADDRF	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRG	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADCSR	ADF	ADIE	ADST	—	—	CH2	CH1	CH0	
ADCR	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	ADSTCLR	EXTRGS	
P4NCE	P47NCE	P46NCE	P45NCE	P44NCE	—	—	—	—	PORT
P4NCMC	P47NCMC	P46NCMC	P45NCMC	P44NCMC	—	—	—	—	
P6PCR	—	—	—	—	P63PCR	P62PCR	P61PCR	P60PCR	
P4PCR	P47PCR	P46PCR	P45PCR	P44PCR	—	—	—	—	
ICCR_3	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_3
ICSR_3	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	
ICDR_3	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SARX_3	SVAX6	SVAX5	SVAX4	SVAX3	SVAX2	SVAX1	SVAX0	FSX	
ICMR_3	MLS	WAIT	CKS2	CKS1	CKS0	BC2	BC1	BC0	
SAR_3	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS	
ICCR_2	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_2
ICSR_2	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	
ICDR_2	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SARX_2	SVAX6	SVAX5	SVAX4	SVAX3	SVAX2	SVAX1	SVAX0	FSX	
ICMR_2	MLS	WAIT	CKS2	CKS1	CKS0	BC2	BC1	BC0	
SAR_2	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS	
DADRA_1	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	PWMX_1
	DA5	DA4	DA3	DA2	DA1	DA0	CFS	—	
DACR_1	—	PWME	—	—	OEB	OEA	OS	CKS	
DADRB_1	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	
	DA5	DA4	DA3	DA2	DA1	DA0	CFS	REGS	
DACNT_1	UC7	UC6	UC5	UC4	UC3	UC2	UC1	UC0	
	UC8	UC9	UC10	UC11	UC12	UC13	—	REGS	
CRCCR	DORCLR	—	—	—	—	LMS	G0	G0	CRC
CRCDIR	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
CRCDOR	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	CRC
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
ICXR_0	STOPIM	HNDS	ICDRF	ICDRE	ALIE	ALSL	FNC1	FNC0	IIC_0
ICXR_1	STOPIM	HNDS	ICDRF	ICDRE	ALIE	ALSL	FNC1	FNC0	IIC_1
ICSMBCR	SMB5E	SMB4E	SMB3E	SMB2E	SMB1E	SMB0E	FSEL1	FSEL0	IIC
ICXR_2	STOPIM	HNDS	ICDRF	ICDRE	ALIE	ALSL	FNC1	FNC0	IIC_2
ICXR_3	STOPIM	HNDS	ICDRF	ICDRE	ALIE	ALSL	FNC1	FNC0	IIC_3
IICX3	—	—	—	—	TCSS	—	—	IICX3	IIC
KBCOMP	EVENTE	—	—	—	—	—	—	—	EVC
ICRD	ICRD7	ICRD6	—	—	—	—	—	—	INT
ICRA	ICRA7	ICRA6	ICRA5	ICRA4	ICRA3	ICRA2	ICRA1	ICRA0	
ICRB	ICRB7	ICRB6	—	ICRB4	ICRB3	ICRB2	ICRB1	—	
ICRC	ICRC7	ICRC6	—	ICRC4	ICRC3	ICRC2	ICRC1	—	
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	
DTCERA	DTCEA7	DTCEA6	DTCEA5	DTCEA4	DTCEA3	—	—	—	DTC
DTCERB	—	DTCEB6	DTCEB5	—	—	—	—	—	
DTCERC	—	—	—	DTCEC4	—	DTCEC2	DTCEC1	DTCEC0	
DTCERD	DTCED7	—	—	DTCED4	DTCED3	—	—	—	
DTCERE	—	—	—	—	DTCEE3	DTCEE2	DTCEE1	—	
DTVECR	SWDTE	DTVEC6	DTVEC5	DTVEC4	DTVEC3	DTVEC2	DTVEC1	DTVEC0	
ABRKCR	CMF	—	—	—	—	—	—	BIE	INT
BARA	A23	A22	A21	A20	A19	A18	A17	A16	
BARB	A15	A14	A13	A12	A11	A10	A9	A8	
BARC	A7	A6	A5	A4	A3	A2	A1	—	
IER16	IRQ15E	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E	
ISR16	IRQ15F	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F	
ISCR16H	IRQ15SCB	IRQ15SCA	IRQ14SCB	IRQ14SCA	IRQ13SCB	IRQ13SCA	IRQ12SCB	IRQ12SCA	
ISCR16L	IRQ11SCB	IRQ11SCA	IRQ10SCB	IRQ10SCA	IRQ9SCB	IRQ9SCA	IRQ8SCB	IRQ8SCA	
ISSR16	ISS15	ISS14	ISS13	ISS12	ISS11	ISS10	ISS9	ISS8	
ISSR	ISS7	ISS6	ISS5	ISS4	ISS3	ISS2	ISSR1	ISS0	
PCSR	PWCKX1B	PWCKX1A	PWCKX0B	PWCKX0A	PWCKX1C	PWCKB	PWCKA	PWCKX0C	SYSTEM
SBYCR	SSBY	STS2	STS1	STS0	DTSPEED	SCK2	SCK1	SCK0	
LPWRCR	—	—	NESEL	EXCLE	—	—	—	—	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
MSTPCRH	MSTP15	MSTP14	MSTP13	MSTP12	MSTP11	MSTP10	MSTP9	MSTP8	SYSTEM
MSTPCRL	MSTP7	MSTP6	MSTP5	MSTP4	MSTP3	MSTP2	MSTP1	MSTP0	
ICCR_1	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_1
ICSR_1	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	
ICDR_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SARX_1	SVAX6	SVAX5	SVAX4	SVAX3	SVAX2	SVAX1	SVAX0	FSX	
ICMR_1	MLS	WAIT	CKS2	CKS1	CKS0	BC2	BC1	BC0	
SAR_1	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS	
TIER	—	—	—	—	OCIAE	OCIBE	OVIE	—	FRT
TCSR	—	—	—	—	OCFA	OCFB	OVF	CCLRA	
FRC	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
OCRA	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
OCRB	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCR	—	—	—	—	—	—	CKS1	CKS0	
TOCR	—	OCRAMS	ICRS	OCRS	—	—	—	—	
OCRAR	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
OCRAF	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	
	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
DADRA_0	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	PWMX_0
	DA5	DA4	DA3	DA2	DA1	DA0	CFS	—	
DACR_0	—	PWME	—	—	OEB	OEA	OS	CKS	
DADRB_0	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	
	DA5	DA4	DA3	DA2	DA1	DA0	CFS	REGS	
DACNT_0	UC7	UC6	UC5	UC4	UC3	UC2	UC1	UC0	
	UC8	UC9	UC10	UC11	UC12	UC13	—	REGS	
TCSR_0	OVF	WT/$\overline{\text{IT}}$	TME	—	RST/$\overline{\text{NMI}}$	CKS2	CKS1	CKS0	WDT_0
TCNT_0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
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	—	—	—	—	P23PCR	P22PCR	P21PCR	P20PCR	

Register									
Abbreviation	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Module**
P3PCR	P37PCR	P36PCR	P35PCR	P34PCR	P33PCR	P32PCR	P31PCR	P30PCR	PORT
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	
P2DDR	—	—	—	—	P23DDR	P22DDR	P21DDR	P20DDR	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	—	—	—	—	P23DR	P22DR	P21DR	P20DR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P4DDR	P47DDR	P46DDR	P45DDR	P44DDR	P43DDR	P42DDR	P41DDR	P40DDR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P4DR	P47DR	P46DR	P45DR	P44DR	P43DR	P42DR	P41DR	P40DR	
P5DDR	P57DDR	P56DDR	—	—	P53DDR	P52DDR	—	—	
P6DDR	—	—	—	—	P63DDR	P62DDR	P61DDR	P60DDR	
P5DR	P57DR	P56DR	—	—	P53DR	P52DR	—	—	
P6DR	—	—	—	—	P63DR	P62DR	P61DR	P60DR	
P8DDR	P87DDR	P86DDR	P85DDR	P84DDR	P83DDR	P82DDR	P81DDR	P80DDR	
P7PIN	P77PIN	P76PIN	P75PIN	P74PIN	P73PIN	P72PIN	P71PIN	P70PIN	
P8DR	P87DR	P86DR	P85DR	P84DR	P83DR	P82DR	P81DR	P80DR	
IER	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	INT
STCR	IICX2	IICX1	IICX0	—	FLSHE	—	ICKS1	ICKS0	SYSTEM
SYSCR	—	—	INTM1	INTM0	XRST	NMIEG	—	RAME	
MDCR	—	—	—	—	—	MDS2	MDS1	—	
TCR_0	CMIEB	CMIEA	OVIE	—	—	CKS2	CKS1	CKS0	TMR_0
TCR_1	CMIEB	CMIEA	OVIE	—	—	CKS2	CKS1	CKS0	TMR_1
TCSR_0	CMFB	CMFA	OVF	ADTE	—	—	—	—	TMR_0
TCSR_1	CMFB	CMFA	OVF	—	—	—	—	—	TMR_1
TCORA_0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_0
TCORA_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_1
TCORB_0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_0
TCORB_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_1
TCNT_0	Bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_0
TCNT_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_1
ICCR_0	ICE	IEIC	MST	TRS	ACKE	BBSY	IRIC	SCP	IIC_0
ICSR_0	ESTP	STOP	IRTR	AASX	AL	AAS	ADZ	ACKB	
ICDR_0	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SARX_0	SVAX6	SVAX5	SVAX4	SVAX3	SVAX2	SVAX1	SVAX0	FSX	
ICMR_0	MLS	WAIT	CKS2	CKS1	CKS0	BC2	BC1	BC0	
SAR_0	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	FS	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
SMR_3*	C/$\overline{\text{A}}$ (GM)	CHR (BLK)	PE (PE)	O/$\overline{\text{E}}$ (O/$\overline{\text{E}}$)	STOP (BCP1)	MP (BCP0)	CKS1 (CKS1)	CKS0 (CKS0)	SCI_3
BRR_3	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SCR_3	TIE	RIE	TE	RE	MPIE	TIE	CKE1	CKE0	
TDR_3	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SSR_3*	TDRE (TDRE)	RDRF (RDRF)	ORER (ORER)	FER (ERS)	PER (PER)	TEND (TEND)	MPB (MPB)	MPBT (MPBT)	
RDR_3	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
SCMR_3	—	—	—	—	SDIR	SINV	—	SMIF	
TCSR_1	OVF	WT/$\overline{\text{IT}}$	TME	PSS	RST/$\overline{\text{NMI}}$	CKS2	CKS1	CKS0	WDT_1
TCNT_1	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCR_X	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_X
TCSR_X	CMFB	CMFA	OVF	—	—	—	—	—	
TCNT_X	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCORA_X	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCORB_X	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	TMR_Y
TCR_Y	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	
TCSR_Y	CMFB	CMFA	OVF	—	—	—	—	—	
TCORA_Y	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCORB_Y	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCNT_Y	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	
TCONRS	TMRX/Y	—	—	—	—	—	—	—	TMR

Note: Some bits have different names in normal mode and smart card interface mode. The Bit name in smart card interface mode is enclosed in parentheses.

23.3 Register States in Each Operating Mode

Register Abbreviation	Reset	WDT Reset	High-Speed/ Medium-Speed	Sleep	Module Stop	Software Standby	Hardware Standby	Module
HICR4	Initialized	Initialized	—	—	—	—	Initialized	LPC
BTSR0	Initialized	Initialized	—	—	—	—	Initialized	
BTSR1	Initialized	Initialized	—	—	—	—	Initialized	
BTCSR0	Initialized	Initialized	—	—	—	—	Initialized	
BTCSR1	Initialized	Initialized	—	—	—	—	Initialized	
BTCR	Initialized	Initialized	—	—	—	—	Initialized	
BTIMSR	Initialized	Initialized	—	—	—	—	Initialized	
SMICFLG	Initialized	Initialized	—	—	—	—	Initialized	
SMICCSR	—	—	—	—	—	—	—	
SMICDTR	—	—	—	—	—	—	—	
SMICIR0	Initialized	Initialized	—	—	—	—	Initialized	
SMICIR1	Initialized	Initialized	—	—	—	—	Initialized	
TWR0MW	—	—	—	—	—	—	—	
TWR0SW	—	—	—	—	—	—	—	
TWR1	—	—	—	—	—	—	—	
TWR2	—	—	—	—	—	—	—	
TWR3	—	—	—	—	—	—	—	
TWR4	—	—	—	—	—	—	—	
TWR5	—	—	—	—	—	—	—	
TWR6	—	—	—	—	—	—	—	
TWR7	—	—	—	—	—	—	—	
TWR8	—	—	—	—	—	—	—	
TWR9	—	—	—	—	—	—	—	
TWR10	—	—	—	—	—	—	—	
TWR11	—	—	—	—	—	—	—	
TWR12	—	—	—	—	—	—	—	
TWR13	—	—	—	—	—	—	—	
TWR14	—	—	—	—	—	—	—	

Register Abbreviation	Reset	WDT Reset	High-Speed/ Medium-Speed	Sleep	Module Stop	Software Standby	Hardware Standby	Module
TWR15	—	—	—	—	—	—	—	LPC
IDR3	—	—	—	—	—	—	—	
ODR3	—	—	—	—	—	—	—	
STR3	Initialized	Initialized	—	—	—	—	—	
SIRQCR4	Initialized	Initialized	—	—	—	—	Initialized	
LADR3H	Initialized	Initialized	—	—	—	—	Initialized	
LADR3L	Initialized	Initialized	—	—	—	—	Initialized	
SIRQCR0	Initialized	Initialized	—	—	—	—	Initialized	
SIRQCR1	Initialized	Initialized	—	—	—	—	Initialized	
IDR1	—	—	—	—	—	—	—	
ODR1	—	—	—	—	—	—	—	
STR1	Initialized	Initialized	—	—	—	—	Initialized	
SIRQCR5	Initialized	Initialized	—	—	—	—	Initialized	
IDR2	—	—	—	—	—	—	—	
ODR2	—	—	—	—	—	—	—	
STR2	Initialized	Initialized	—	—	—	—	Initialized	
HISEL	Initialized	Initialized	—	—	—	—	Initialized	
HICR0	Initialized	Initialized	—	—	—	—	Initialized	
HICR1	Initialized	Initialized	—	—	—	—	Initialized	
HICR2	Initialized	Initialized	—	—	—	—	Initialized	
HICR3	—	—	—	—	—	—	—	
SIRQCR2	Initialized	Initialized	—	—	—	—	Initialized	
BTDTR	—	—	—	—	—	—	—	
BTFVSR0	Initialized	Initialized	—	—	—	—	Initialized	
BTFVSR1	Initialized	Initialized	—	—	—	—	Initialized	
LADR12H	Initialized	Initialized	—	—	—	—	Initialized	
LADR12L	Initialized	Initialized	—	—	—	—	Initialized	
SUBMSTPBH	Initialized	Initialized	—	—	—	—	Initialized	SYSTEM
SUBMSTPBL	Initialized	Initialized	—	—	—	—	Initialized	
ECS	Initialized	Initialized	—	—	—	—	Initialized	EVC
ECCR	Initialized	Initialized	—	—	—	—	Initialized	
MSTPCRA	Initialized	Initialized	—	—	—	—	Initialized	SYSTEM
P3NCE	Initialized	Initialized	—	—	—	—	Initialized	PORT
P3NCMC	Initialized	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	WDT Reset	High-Speed/ Medium- Speed	Sleep	Module Stop	Software Standby	Hardware Standby	Module
NCCS	Initialized	Initialized	—	—	—	—	Initialized	PORT
PEODR	Initialized	Initialized	—	—	—	—	Initialized	
PEPIN	Initialized	Initialized	—	—	—	—	Initialized	
PEDDR	Initialized	Initialized	—	—	—	—	Initialized	
PCODR	Initialized	Initialized	—	—	—	—	Initialized	
PCPIN	Initialized	Initialized	—	—	—	—	Initialized	
PCDDR	Initialized	Initialized	—	—	—	—	Initialized	
FCCS	Initialized	Initialized	—	—	—	—	Initialized	FLASH
FPCS	Initialized	Initialized	—	—	—	—	Initialized	
FECS	Initialized	Initialized	—	—	—	—	Initialized	
FKEY	Initialized	Initialized	—	—	—	—	Initialized	
FMATS	Initialized	Initialized	—	—	—	—	Initialized	
FTDAR	Initialized	Initialized	—	—	—	—	Initialized	
SMR_1	Initialized	Initialized	—	—	—	—	Initialized	SCI_1
BRR_1	Initialized	Initialized	—	—	—	—	Initialized	
SCR_1	Initialized	Initialized	—	—	—	—	Initialized	
TDR_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
SSR_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
RDR_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
SCMR_1	Initialized	Initialized	—	—	—	—	Initialized	
ADDRA	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	ADC
ADDRB	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADDRC	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADDRD	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADDRE	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADDRF	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADDRG	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADDRH	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADCSR	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
ADCR	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
P4NCE	Initialized	Initialized	—	—	—	—	Initialized	PORT
P4NCMC	Initialized	Initialized	—	—	—	—	Initialized	
P6PCR	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
P4PCR	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	

Register Abbreviation	Reset	WDT Reset	High-Speed/ Medium-Speed	Sleep	Module Stop	Software Standby	Hardware Standby	Module
ICCR_3	Initialized	Initialized	—	—	—	—	Initialized	IIC_3
ICSR_3	Initialized	Initialized	—	—	—	—	Initialized	
ICDR_3	—	—	—	—	—	—	—	
SARX_3	Initialized	Initialized	—	—	—	—	Initialized	
ICMR_3	Initialized	Initialized	—	—	—	—	Initialized	
SAR_3	Initialized	Initialized	—	—	—	—	Initialized	
ICCR_2	Initialized	Initialized	—	—	—	—	Initialized	IIC_2
ICSR_2	Initialized	Initialized	—	—	—	—	Initialized	
ICDR_2	—	—	—	—	—	—	—	
SARX_2	Initialized	Initialized	—	—	—	—	Initialized	
ICMR_2	Initialized	Initialized	—	—	—	—	Initialized	
SAR_2	Initialized	Initialized	—	—	—	—	Initialized	
DADRA_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	PWMX_1
DACR_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
DADRB_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
DACNT_1	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
CRCCR	Initialized	Initialized	—	—	—	—	Initialized	CRC
CRCDIR	Initialized	Initialized	—	—	—	—	Initialized	
CRCDOR	Initialized	Initialized	—	—	—	—	Initialized	
ICXR_0	Initialized	Initialized	—	—	—	—	Initialized	IIC_0
ICXR_1	Initialized	Initialized	—	—	—	—	Initialized	IIC_1
ICSMBCR	Initialized	Initialized	—	—	—	—	Initialized	IIC
ICXR_2	Initialized	Initialized	—	—	—	—	Initialized	IIC_2
ICXR_3	Initialized	Initialized	—	—	—	—	Initialized	IIC_3
IICX3	Initialized	Initialized	—	—	—	—	Initialized	IIC
KBCOMP	Initialized	Initialized	—	—	—	—	Initialized	EVC
ICRD	Initialized	Initialized	—	—	—	—	Initialized	INT
ICRA	Initialized	Initialized	—	—	—	—	Initialized	
ICRB	Initialized	Initialized	—	—	—	—	Initialized	
ICRC	Initialized	Initialized	—	—	—	—	Initialized	
ISR	Initialized	Initialized	—	—	—	—	Initialized	
ISCRH	Initialized	Initialized	—	—	—	—	Initialized	
ISCRL	Initialized	Initialized	—	—	—	—	Initialized	
DTCERA	Initialized	Initialized	—	—	—	—	Initialized	DTC
DTCERB	Initialized	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Reset	WDT Reset	High-Speed/ Medium- Speed	Sleep	Module Stop	Software Standby	Hardware Standby	Module
DTCERC	Initialized	Initialized	—	—	—	—	Initialized	DTC
DTCERD	Initialized	Initialized	—	—	—	—	Initialized	
DTCERE	Initialized	Initialized	—	—	—	—	Initialized	
DTVECR	Initialized	Initialized	—	—	—	—	Initialized	
ABRKCR	Initialized	Initialized	—	—	—	—	Initialized	INT
BARA	Initialized	Initialized	—	—	—	—	Initialized	
BARB	Initialized	Initialized	—	—	—	—	Initialized	
BARC	Initialized	Initialized	—	—	—	—	Initialized	
IER16	Initialized	Initialized	—	—	—	—	Initialized	
ISR16	Initialized	Initialized	—	—	—	—	Initialized	
ISCR16H	Initialized	Initialized	—	—	—	—	Initialized	
ISCR16L	Initialized	Initialized	—	—	—	—	Initialized	
ISSR16	Initialized	Initialized	—	—	—	—	Initialized	
ISSR	Initialized	Initialized	—	—	—	—	Initialized	
PCSR	Initialized	Initialized	—	—	—	—	Initialized	SYSTEM
SBYCR	Initialized	Initialized	—	—	—	—	Initialized	
LPWRCR	Initialized	Initialized	—	—	—	—	Initialized	
MSTPCRH	Initialized	Initialized	—	—	—	—	Initialized	
MSTPCRL	Initialized	Initialized	—	—	—	—	Initialized	
ICCR_1	Initialized	Initialized	—	—	—	—	Initialized	IIC_1
ICSR_1	Initialized	Initialized	—	—	—	—	Initialized	
ICDR_1	—	—	—	—	—	—	—	
SARX_1	Initialized	Initialized	—	—	—	—	Initialized	
ICMR_1	Initialized	Initialized	—	—	—	—	Initialized	
SAR_1	Initialized	Initialized	—	—	—	—	Initialized	
TIER	Initialized	Initialized	—	—	—	—	Initialized	FRT
TCSR	Initialized	Initialized	—	—	—	—	Initialized	
FRC	Initialized	Initialized	—	—	—	—	Initialized	
OCRA	Initialized	Initialized	—	—	—	—	Initialized	
OCRB	Initialized	Initialized	—	—	—	—	Initialized	
TCR	Initialized	Initialized	—	—	—	—	Initialized	
TOCR	Initialized	Initialized	—	—	—	—	Initialized	
OCRAR	Initialized	Initialized	—	—	—	—	Initialized	
OCRAF	Initialized	Initialized	—	—	—	—	Initialized	

			High-Speed/					
Register			Medium-			Software	Hardware	
Abbreviation	Reset	WDT Reset	Speed	Sleep	Module Stop	Standby	Standby	Module
DADRA_0	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	PWMX_0
DACR_0	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
DADRB_0	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
DACNT_0	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
TCSR_0	Initialized	Initialized	—	—	—	—	Initialized	WDT_0
TCNT_0	Initialized	Initialized	—	—	—	—	Initialized	
PAODR	Initialized	Initialized	—	—	—	—	Initialized	PORT
PAPIN	—	—	—	—	—	—	—	
PADDR	Initialized	Initialized	—	—	—	—	Initialized	
P1PCR	Initialized	Initialized	—	—	—	—	Initialized	
P2PCR	Initialized	Initialized	—	—	—	—	Initialized	
P3PCR	Initialized	Initialized	—	—	—	—	Initialized	
P1DDR	Initialized	Initialized	—	—	—	—	Initialized	
P2DDR	Initialized	Initialized	—	—	—	—	Initialized	
P1DR	Initialized	Initialized	—	—	—	—	Initialized	
P2DR	Initialized	Initialized	—	—	—	—	Initialized	
P3DDR	Initialized	Initialized	—	—	—	—	Initialized	
P4DDR	Initialized	—	—	—	—	—	Initialized	
P3DR	Initialized	Initialized	—	—	—	—	Initialized	
P4DR	Initialized	—	—	—	—	—	Initialized	
P5DDR	Initialized	Initialized	—	—	—	—	Initialized	
P6DDR	Initialized	Initialized	—	—	—	—	Initialized	
P5DR	Initialized	Initialized	—	—	—	—	Initialized	
P6DR	Initialized	Initialized	—	—	—	—	Initialized	
P8DDR	Initialized	Initialized	—	—	—	—	Initialized	
P7PIN	—	—	—	—	—	—	—	
P8DR	Initialized	Initialized	—	—	—	—	Initialized	
IER	Initialized	Initialized	—	—	—	—	Initialized	INT
STCR	Initialized	Initialized	—	—	—	—	Initialized	SYSTEM
SYSCR	Initialized	Initialized	—	—	—	—	Initialized	
MDCR	Initialized	Initialized	—	—	—	—	Initialized	
TCR_0	Initialized	Initialized	—	—	—	—	Initialized	TMR_0
TCR_1	Initialized	Initialized	—	—	—	—	Initialized	TMR_1
TCSR_0	Initialized	Initialized	—	—	—	—	Initialized	TMR_0
TCSR_1	Initialized	Initialized	—	—	—	—	Initialized	TMR_1

Register Abbreviation	Reset	WDT Reset	High-Speed/ Medium- Speed	Sleep	Module Stop	Software Standby	Hardware Standby	Module
TCORA_0	Initialized	Initialized	—	—	—	—	Initialized	TMR_0
TCORA_1	Initialized	Initialized	—	—	—	—	Initialized	TMR_1
TCORB_0	Initialized	Initialized	—	—	—	—	Initialized	TMR_0
TCORB_1	Initialized	Initialized	—	—	—	—	Initialized	TMR_1
TCNT_0	Initialized	Initialized	—	—	—	—	Initialized	TMR_0
TCNT_1	Initialized	Initialized	—	—	—	—	Initialized	TMR_1
ICCR_0	Initialized	Initialized	—	—	—	—	Initialized	IIC_0
ICSR_0	Initialized	Initialized	—	—	—	—	Initialized	
ICDR_0	—	—	—	—	—	—	—	IIC_0
SARX_0	Initialized	Initialized	—	—	—	—	Initialized	
ICMR_0	Initialized	Initialized	—	—	—	—	Initialized	
SAR_0	Initialized	Initialized	—	—	—	—	Initialized	
SMR_3	Initialized	Initialized	—	—	—	—	Initialized	SCI_3
BRR_3	Initialized	Initialized	—	—	—	—	Initialized	
SCR_3	Initialized	Initialized	—	—	—	—	Initialized	
TDR_3	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
SSR_3	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
RDR_3	Initialized	Initialized	—	—	Initialized	Initialized	Initialized	
SCMR_3	Initialized	Initialized	—	—	—	—	Initialized	
TCSR_1	Initialized	Initialized	—	—	—	—	Initialized	WDT_1
TCNT_1	Initialized	Initialized	—	—	—	—	Initialized	
TCR_X	Initialized	Initialized	—	—	—	—	Initialized	TMR_X
TCSR_X	Initialized	Initialized	—	—	—	—	Initialized	
TCNT_X	Initialized	Initialized	—	—	—	—	Initialized	
TCORA_X	Initialized	Initialized	—	—	—	—	Initialized	
TCORB_X	Initialized	Initialized	—	—	—	—	Initialized	TMR_Y
TCR_Y	Initialized	Initialized	—	—	—	—	Initialized	
TCSR_Y	Initialized	Initialized	—	—	—	—	Initialized	
TCORA_Y	Initialized	Initialized	—	—	—	—	Initialized	
TCORB_Y	Initialized	Initialized	—	—	—	—	Initialized	
TCNT_Y	Initialized	Initialized	—	—	—	—	Initialized	
TCONRS	Initialized	Initialized	—	—	—	—	Initialized	TMR

Section 24 Electrical Characteristics

24.1 Absolute Maximum Ratings

Table 24.1 lists the absolute maximum ratings.

Table 24.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage* (pins multiplexed with analog input)	VCC	–0.3 to +4.3	V
Input voltage (pins multiplexed with IIC functions) (1)	V_{in}	–0.3 to AVCC +0.3	
Input voltage (2)	V_{in}	–0.3 to +6.5	
Input voltage (except (1), (2))	V_{in}	–0.3 to VCC +0.3	
Reference power supply voltage	AVref	–0.3 to AVCC +0.3	
Analog power supply voltage	AVCC	–0.3 to +4.3	
Analog input voltage (AN0 to AN7)	V_{AN}	–0.3 to AVCC +0.3	
Operating temperature	T_{opr}	–40 to +85	°C
Operating temperature (when flash memory is programmed or erased)	T_{opr}	0 to +75	
Storage temperature	T_{stg}	–55 to +125	

Caution: Permanent damage to this LSI may result if absolute maximum ratings are exceeded.

Note: * Voltage applied to the VCC pin.
Make sure power is not applied to the VCL pin.

24.2 DC Characteristics

Table 24.2 lists the DC characteristics. Table 24.3 lists the permissible output currents. Table 24.4 lists the bus drive characteristics.

Table 24.2 DC Characteristics (1)

Conditions: VCC = 3.0 V to 3.6 V, AVCC*1 = 3.0 V to 3.6 V,
AVref*1 = 3.0 V to AVCC, VSS = AVSS*1 = 0 V

Item			Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Schmitt trigger input voltage	DB7 to DB4, ExDB7 to ExDB0, EVENT7 to EVENT0, $\overline{\text{(Ex)IRQ15}}$, $\overline{\text{(Ex)IRQ14}}$, $\overline{\text{ExIRQ13}}$, $\overline{\text{ExIRQ12}}$, $\overline{\text{(Ex)IRQ11}}$, $\overline{\text{(Ex)IRQ10}}$, $\overline{\text{ExIRQ9}}$, $\overline{\text{ExIRQ8}}$, $\overline{\text{(Ex)IRQ7}}$ to $\overline{\text{(Ex)IRQ0}}$, $\overline{\text{ETRST}}$, XTAL, EXCL, $\overline{\text{ADTRG}}$	(1)	V_T^-	VCC × 0.2	—	—	V	
			V_T^+	—	—	VCC × 0.7		
			$V_T^+ - V_T^-$	VCC × 0.05	—	—		
	SCL3 to SCL0, SDA3 to SDA0		V_T^-	VCC × 0.3	—	—		
			V_T^+	—	—	VCC × 0.7		
			$V_T^+ - V_T^-$	VCC × 0.05	—	—		
Input high voltage	$\overline{\text{RES}}$, $\overline{\text{STBY}}$, NMI, FWE, $\overline{\text{MD2}}$, MD1	(2)	V_{IH}	VCC × 0.9	—	VCC + 0.3	V	
	EXTAL			VCC × 0.7	—	VCC + 0.3		
	Port 7			2.2	—	AVCC + 0.3		
	SCL3 to SCL0, SDA3 to SDA0, Port 80 to 83, C0 to C3			—	—	5.5		
	SERIRQ, LAD3 to LAD0, LCLK, $\overline{\text{LRESET}}$, $\overline{\text{LFRAME}}$			VCC × 0.5	—	VCC + 0.3		
				2.0	—	VCC + 0.3		
	Input pins other than (1) and (2) above			2.2	—	VCC + 0.3		

Item			Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Input low voltage	$\overline{RES}$, $\overline{STBY}$, NMI, FWE, $\overline{MD2}$, MD1	(3)	V_{IL}	–0.3	—	VCC × 0.1	V	
	EXTAL			–0.3	—	VCC × 0.2		
	Port 7			–0.3	—	AVCC × 0.2		
	SERIRQ, LAD3 to LAD0, LCLK, $\overline{LRESET}$, $\overline{LFRAME}$			–0.3	—	VCC × 0.3		
	Input pins other than (1) and (3) above			–0.3	—	VCC × 0.2		
Output high voltage	SCL3 to SCL0, SDA3 to SDA0[*2]	(4)	V_{OH}	—	—	—	V	
	Port 80 to 83, C0 to C3[*3]			0.5	—	—		I_{OH} = –200 μA
	SERIRQ, LAD3 to LAD0			VCC × 0.9	—	—		I_{OH} = –0.5 mA
	Output pins other than (4) above			VCC – 0.5	—	—		I_{OH} = –200 μA
				VCC – 1.0	—	—		I_{OH} = –1 mA
Output low voltage	SCL3 to SCL0, SDA3 to SDA0	(5)	V_{OL}	—	—	0.5	V	I_{OL} = 8 mA
				—	—	0.4		I_{OL} = 3 mA
	SERIRQ, LAD3 to LAD0			—	—	VCC × 0.1		I_{OL} = 1.5 mA
	Output pins other than (5) above			—	—	0.4		I_{OL} = 1.6 mA
	HC7 to HC0			—	—	1.0		I_{OL} = 12 mA

Table 24.2 DC Characteristics (2)

Conditions: VCC = 3.0 V to 3.6 V, AVCC*[1] = 3.0 V to 3.6 V,
AVref*[1] = 3.0 V to AVCC, VSS = AVSS*[1] = 0 V

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Input leakage current	$\overline{\text{RES}}$, $\overline{\text{STBY}}$, NMI, FWE, $\overline{\text{MD2}}$, MD1	$\lvert I_{in} \rvert$	—	—	1.0	µA	V_{IN} = 0.5 to VCC – 0.5 V
	Port 7		—	—	1.0		V_{IN} = 0.5 to AVCC – 0.5 V
Three-state leakage current (off state)	Ports 1 to 6 Ports 8 to E	$\lvert I_{TSI} \rvert$	—	—	1.0		V_{IN} = 0.5 to VCC – 0.5 V
Input pull-up MOS current	Ports 1 to 4, 6, A	$-I_P$	20	—	300		V_{IN} = 0 V
Supply current*[4]	Normal operation	I_{CC}	—	45	60	mA	f = 25 MHz, high-speed mode, All modules operating
	Sleep mode		—	35	45		f = 25 MHz
	Standby mode*[5]		—	40	100	µA	Ta ≤ 50 °C
			—	—	250		50 °C < Ta
Analog power supply current	During A/D conversion	AI_{cc}	—	1.0	2.0	mA	
	A/D conversion standby		—	2.5	5.0	µA	
Reference power supply current	During A/D conversion	AI_{ref}	—	0.1	1.0	mA	
	A/D conversion standby		—	0.5	5.0	µA	
Input capacitance	All input pin	C_{in}	—	—	10	pF	V_{in} = 0 V, f = 1 MHz, T_a = 25 °C
RAM standby voltage		V_{RAM}	3.0	—	—	V	
VCC start voltage		VCC_{START}	—	0	0.8	V	
VCC rising edge		SVCC	—	—	20	ms/V	

Notes: 1. Do not leave the AVCC, AVref, and AVSS pins open even if the A/D converter or D/A converter is not used.

Even if the A/D converter or D/A converter is not used, apply a value in the range from 3.0 V to 3.6 V to the AVCC and AVref pins by connecting them to the power supply (VCC). The relationship between these two pins should be AVref ≤ AVCC.

2. An external pull-up resistor is necessary to provide high-level output from SCL3 to SCL0 and SDA3 to SDA0 (ICE bit in ICCR is 1).

3. Pins P80 to P83 and PC0 to PC3 are NMOS push-pull outputs.

   High levels on pins P80 to P83 and PC0 to PC3 are driven by NMOS. An external pull-up resistor is necessary to provide high-level output from these pins when they are used as an output.
4. Supply current values are for V_{IH} min = VCC – 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.
5. When VCC = 3.0 V, V_{IH} min = VCC – 0.2 V, and V_{IL} max = 0.2 V.

Table 24.3 Permissible Output Currents

Conditions: VCC = 3.0 V to 3.6 V, AVCC = 3.0 V to 3.6 V,
AVref = 3.0 V to AVCC, VSS = AVSS = 0 V

Item		Symbol	Min.	Typ.	Max.	Unit
Permissible output low current (per pin)	SCL3 to SCL0, SDA3 to SDA0	I_{OL}	—	—	10	mA
	HC7 to HC0		—	—	12	
	Other output pins		—	—	1.6	
Permissible output low current (total)	Total of HC7 to HC0	ΣI_{OL}	—	—	48	
	Total of all output pins, including the above		—	—	90	
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	60	

Notes: 1. To protect LSI reliability, do not exceed the output current values in table 24.3.

2. When driving a Darlington transistor or LED, always insert a current-limiting resistor in the output line, as show in figures 24.1 and 24.2.

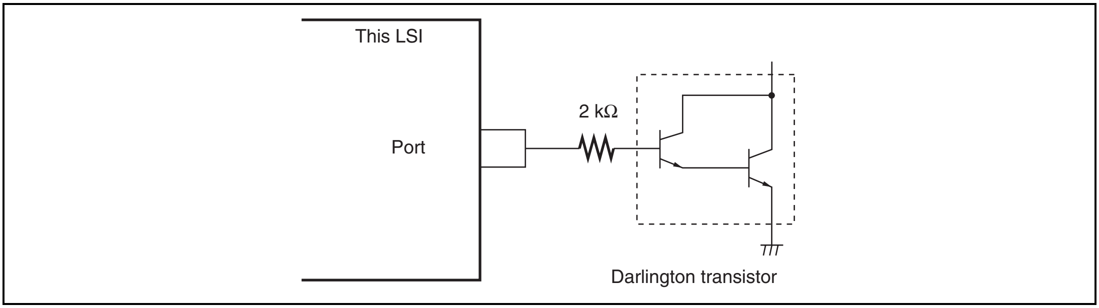


Figure 24.1 Darlington Transistor Drive Circuit (Example)

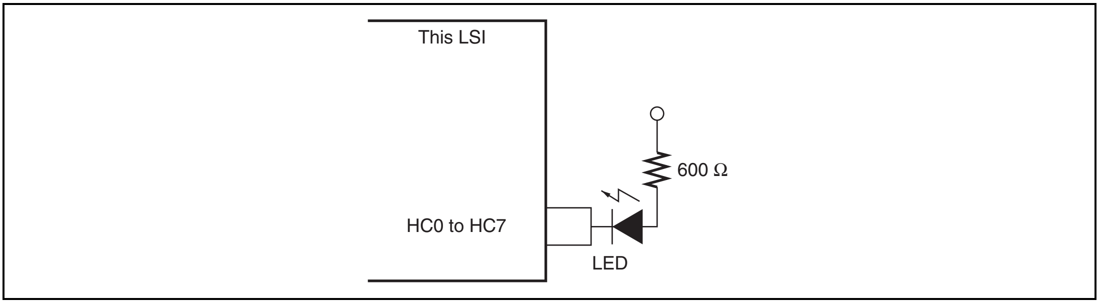


Figure 24.2 LED Drive Circuit (Example)

24.3 AC Characteristics

Figure 24.3 shows the test conditions for the AC characteristics.

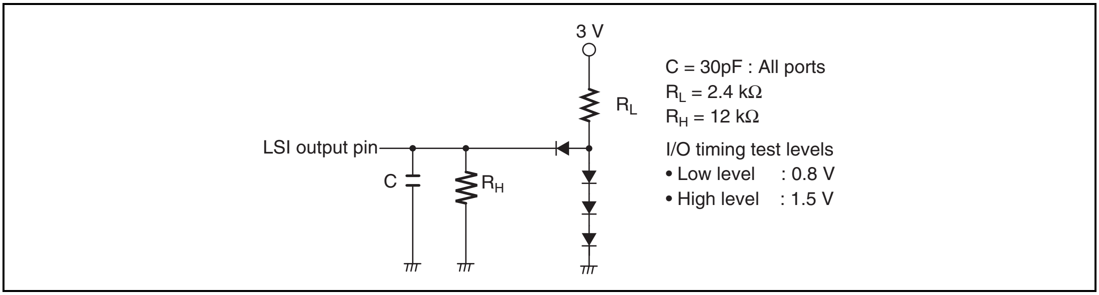


Figure 24.3 Output Load Circuit

24.3.1 Clock Timing

Table 24.4 shows the clock timing. The clock timing specified here covers clock output (ϕ) and clock pulse generator (crystal) and external clock input (EXTAL pin) oscillation stabilization times. For details of external clock input (EXTAL pin and EXCL pin) timing, see table 24.5 and 24.6.

Table 24.4 Clock Timing

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Max.	Unit	Reference
Clock cycle time	t_{cyc}	40	50	ns	Figure 24.4
Clock high level pulse width	t_{CH}	10	—		
Clock low level pulse width	t_{CL}	10	—		
Clock rise time	t_{Cr}	—	5		
Clock fall time	t_{Cf}	—	5		
Reset oscillation stabilization (crystal)	t_{OSC1}	10	—	ms	Figure 24.5
Software standby oscillation stabilization time (crystal)	t_{OSC2}	8	—		Figure 24.6

Table 24.5 External Clock Input Conditions

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Max.	Unit	Test Conditions
External clock input low level pulse width	t_{EXL}	80	—	ns	Figure 24.7
External clock input high level pulse width	t_{EXH}	80	—	ns	
External clock input rising time	t_{EXr}	—	5	ns	
External clock input falling time	t_{EXf}	—	5	ns	
Clock low level pulse width	t_{CL}	0.4	0.6	t_{cyc}	Figure 24.4
Clock high level pulse width	t_{CH}	0.4	0.6	t_{cyc}	
External clock output stabilization delay time	t_{DEXT}*	500	—	μs	Figure 24.8

Note: * t_{DEXT} includes a $\overline{\text{RES}}$ pulse width (t_{RESW}).

Table 24.6 Subclock Input Conditions

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Subclock input low level pulse width	t_{EXCLL}	—	15.26	—	μs	Figure 24.9
Subclock input high level pulse width	t_{EXCLH}	—	15.26	—	μs	
Subclock input rising time	t_{EXCLr}	—	—	10	ns	
Subclock input falling time	t_{EXCLf}	—	—	10	ns	
Clock low level pulse width	t_{CL}	0.4	—	0.6	t_{cyc}	Figure 24.4
Clock high level pulse width	t_{CH}	0.4	—	0.6	t_{cyc}	

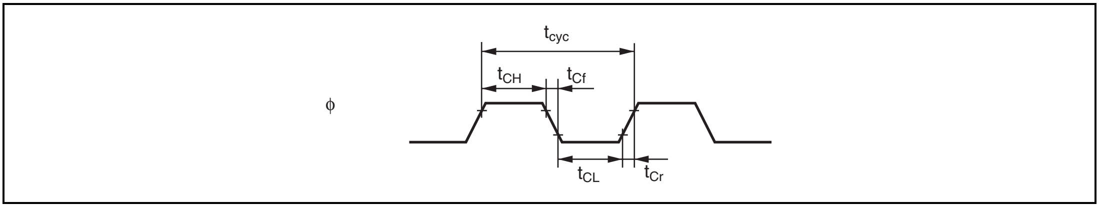


Figure 24.4 System Clock Timing

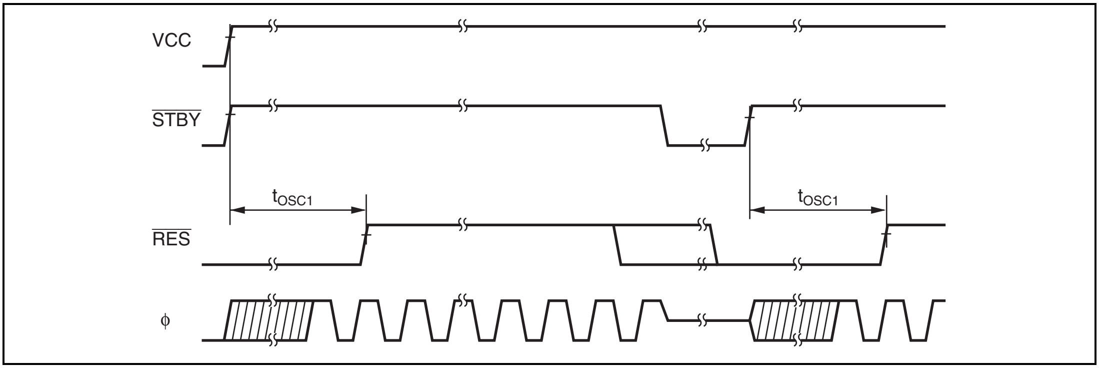


Figure 24.5 Oscillation Stabilization Timing

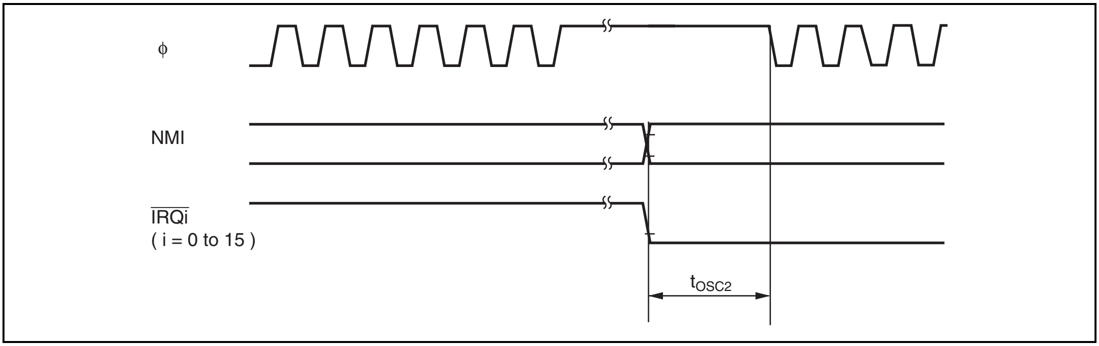


Figure 24.6 Oscillation Stabilization Timing (Exiting Software Standby Mode)

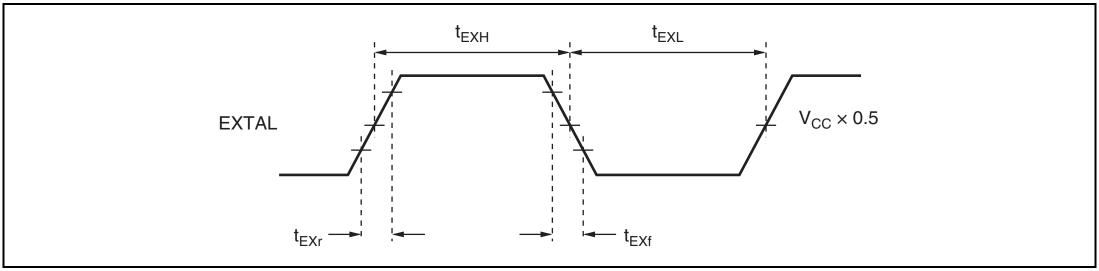
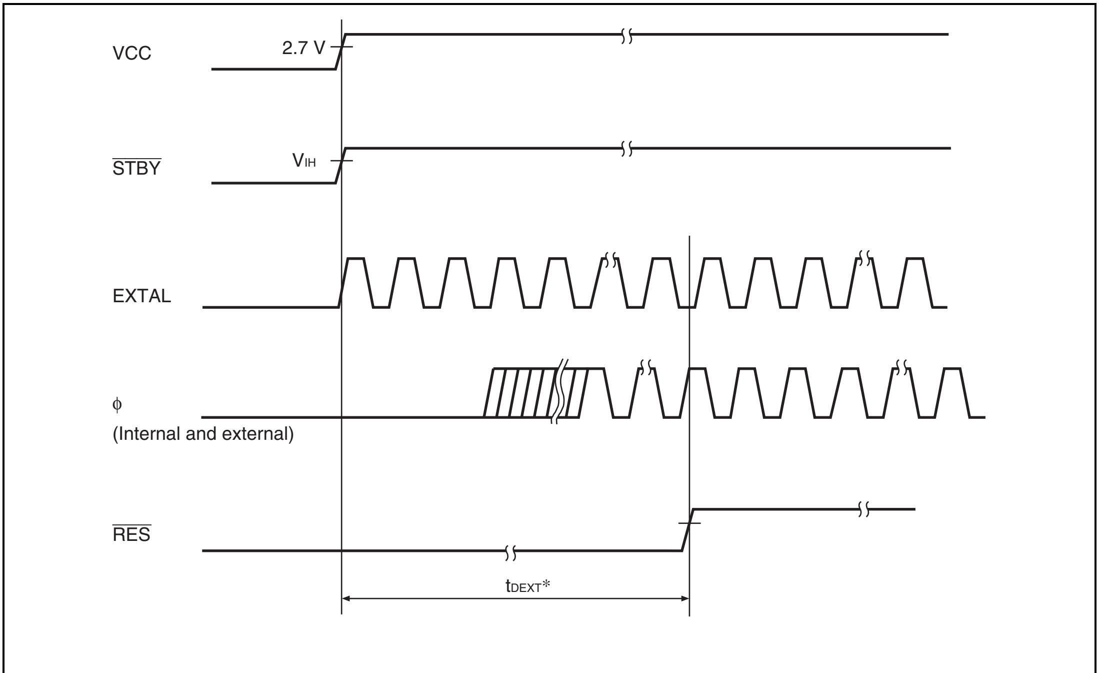


Figure 24.7 External Clock Input Timing



Note: The external clock output stabilization delay time (t_{DEXT}) includes a $\overline{\text{RES}}$ pulse width (t_{RESW}).

Figure 24.8 Timing of External Clock Output Stabilization Delay Time

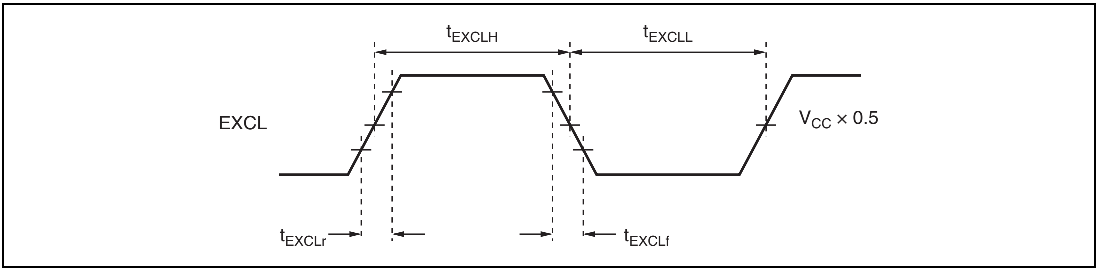


Figure 24.9 Subclock Input Timing

24.3.2 Control Signal Timing

Table 24.7 shows the control signal timing. Only external interrupts NMI and IRQ0 to IRQ15 can be operated based on the subclock (ϕSUB = 32.768 kHz).

Table 24.7 Control Signal Timing

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{\text{RES}}$ setup time	t_{RESS}	200	—	ns	Figure 24.10
$\overline{\text{RES}}$ pulse width	t_{RESW}	20	—	t_{cyc}	
NMI setup time	t_{NMIS}	150	—	ns	Figure 24.11
NMI hold time	t_{NMIH}	10	—		
NMI pulse width (exiting software standby mode)	t_{NMIW}	200	—		
IRQ setup time ($\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$)	t_{IRQS}	150	—		
IRQ hold time ($\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$)	t_{IRQH}	10	—		
IRQ pulse width ($\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$) (exiting software standby mode)	t_{IRQW}	200	—		

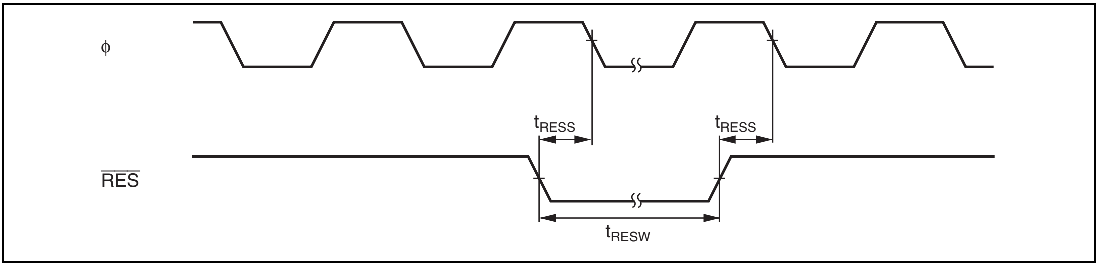


Figure 24.10 Reset Input Timing

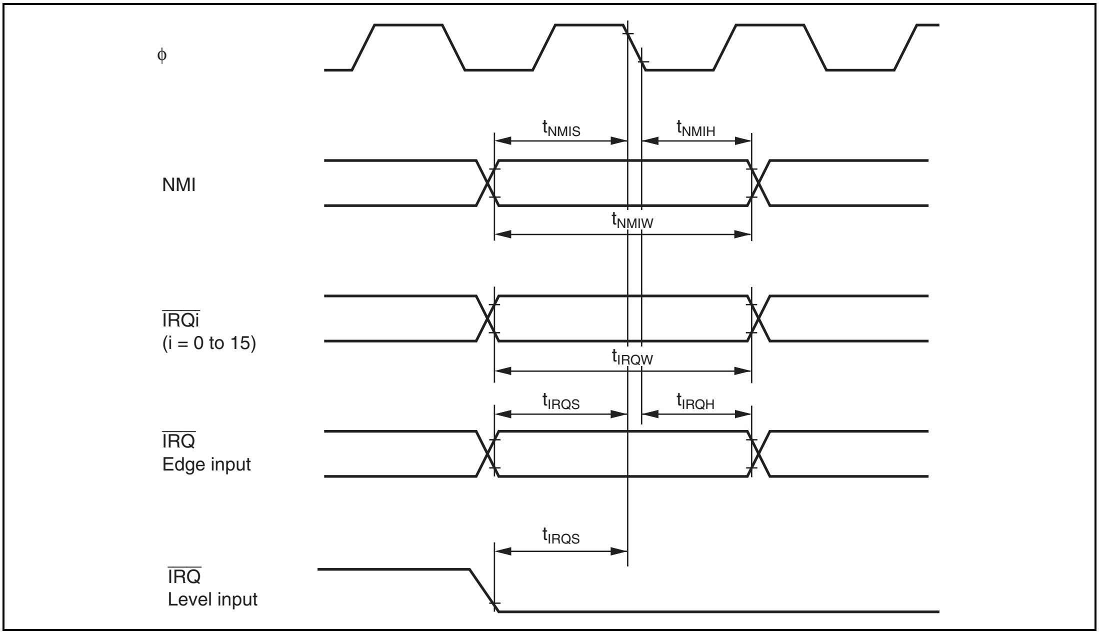


Figure 24.11 Interrupt Input Timing

24.3.3 Timing of On-Chip Peripheral Modules

Tables 24.8 to 24.11 show the on-chip peripheral module timing. The on-chip peripheral modules that can be operated by the subclock (ϕSUB = 32.768 kHz) are I/O ports, external interrupts (NMI and IRQ0 to IRQ15), watchdog timer, and 8-bit timer (channels 0 and 1) only.

Table 24.8 Timing of On-Chip Peripheral Modules

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item			Symbol	Min.	Max.	Unit	Test Conditions
I/O ports	Output data delay time		t_{PWD}	—	30	ns	Figure 24.12
	Input data setup time		t_{PRS}	20	—		
	Input data hold time		t_{PRH}	20	—		
PWMX	Timer output delay time		t_{PWOD}	—	30	ns	Figure 24.13
SCI	Input clock cycle	Asynchronous	t_{Scyc}	4	—	t_{cyc}	Figure 24.14
		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		
	Transmit data delay time (synchronous)		t_{TXD}	—	30	ns	Figure 24.15
	Receive data setup time (synchronous)		t_{RXS}	20	—		
	Receive data hold time (synchronous)		t_{RXH}	20	—		
A/D converter	Trigger input setup time		t_{TRGS}	20	—	ns	Figure 24.16
WDT	$\overline{\text{RESO}}$ output delay time		t_{RESD}	—	50	ns	Figure 24.17
	$\overline{\text{RESO}}$ output pulse width		t_{RESOW}	132	—	t_{cyc}	

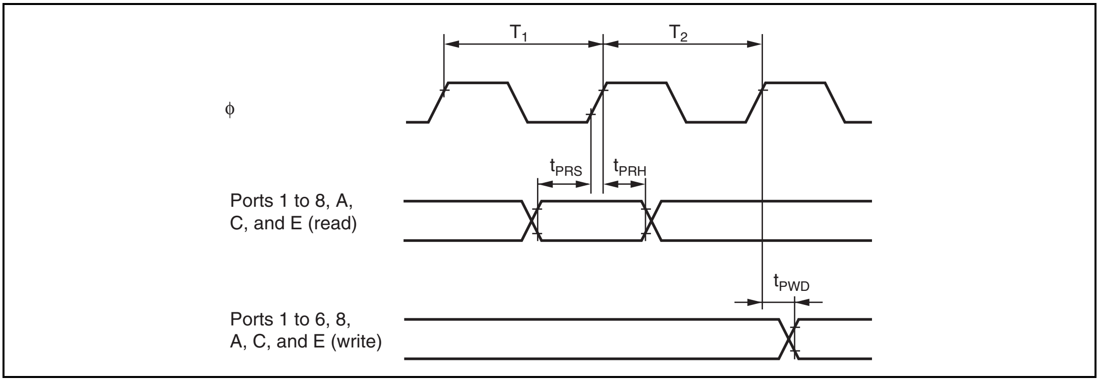

T_1
T_2
ϕ
t_{PRS}
t_{PRH}
Ports 1 to 8, A, C, and E (read)
t_{PWD}
Ports 1 to 6, 8, A, C, and E (write)

Figure 24.12 I/O Port Input/Output Timing

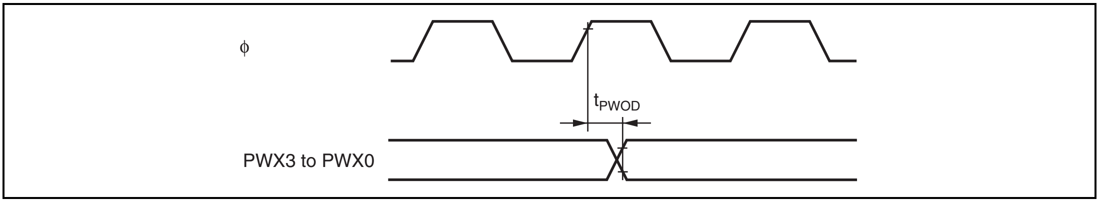


Figure 24.13 PWMX Output Timing

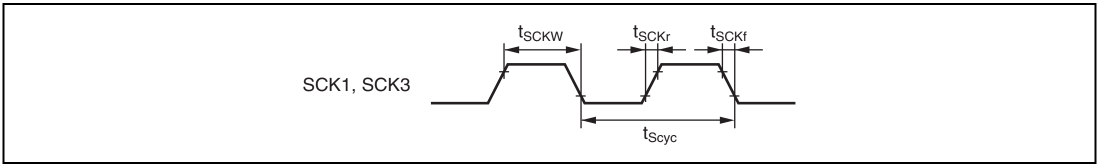


Figure 24.14 SCK Clock Input Timing

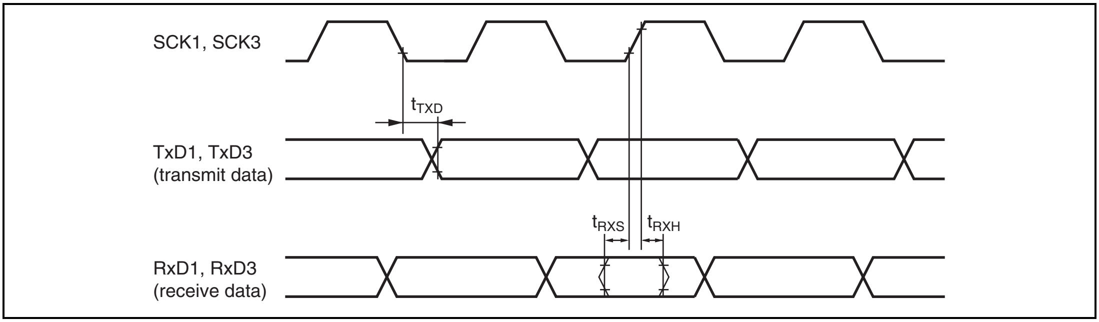


Figure 24.15 SCI Input/Output Timing (Clock Synchronous Mode)

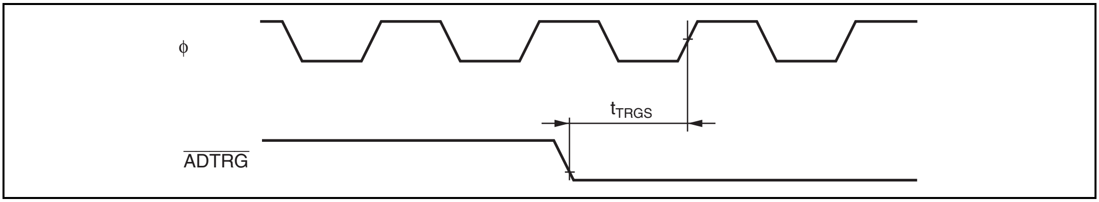


Figure 24.16 A/D Converter External Trigger Input Timing

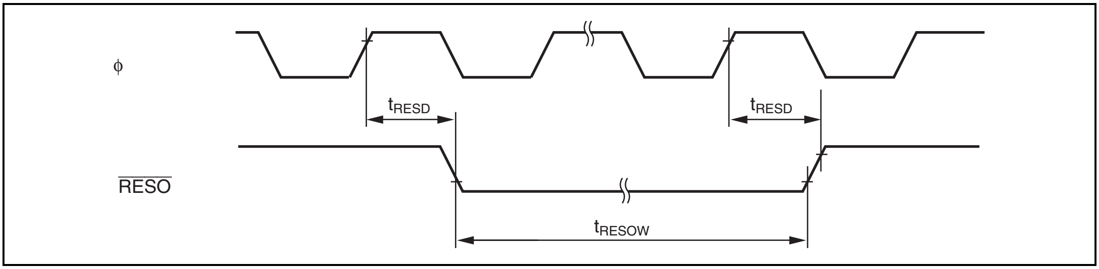


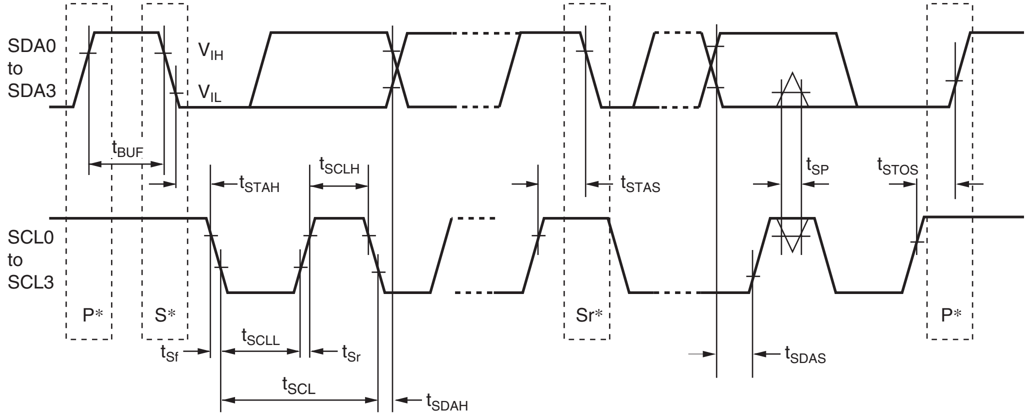
Figure 24.17 WDT Output Timing ($\overline{\text{RESO}}$)

Table 24.9 I^2C Bus Timing

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
SCL input cycle time	t_{SCL}	12	—	—	t_{cyc}	Figure 24.18
SCL input high pulse width	t_{SCLH}	3	—	—		
SCL input low pulse width	t_{SCLL}	5	—	—		
SCL, SDA input rise time	t_{Sr}	—	—	7.5*		
SCL, SDA input fall time	t_{Sf}	—	—	300	ns	
SCL, SDA output fall time	t_{Of}	20 + 0.1 C_b	—	250		
SCL, SDA input spike pulse elimination time	t_{SP}	—	—	1	t_{cyc}	
SDA input bus free time	t_{BUF}	5	—	—		
Start condition input hold time	t_{STAH}	3	—	—		
Retransmission start condition input setup time	t_{STAS}	3	—	—		
Stop condition input setup time	t_{STOS}	3	—	—		
Data input setup time	t_{SDAS}	0.5	—	—		
Data input hold time	t_{SDAH}	0	—	—	ns	
SCL, SDA capacitive load	C_b	—	—	400	pF	

Note: * 17.5 t_{cyc} or 37.5 t_{cyc} can be set according to the clock selected for use by the IIC module.



Note: * S, P, and Sr indicate the following conditions:
S: Start condition
P: Stop condition
Sr: Retransmission start condition

Figure 24.18 I²C Bus Interface Input/Output Timing

Table 24.10 LPC Module Timing

Conditions: VCC = 3.0 V to 3.6V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Input clock cycle	t_{Lcyc}	30	—	—	ns	Figure 24.19
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	—	—		

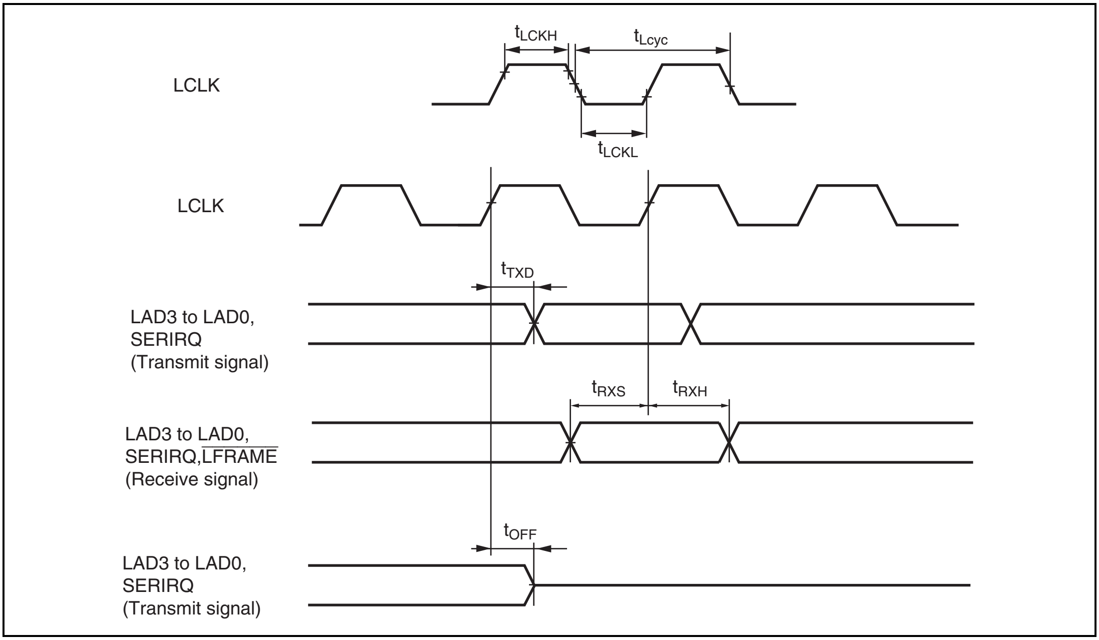


Figure 24.19 LPC Interface (LPC) Timing

Table 24.11 JTAG Timing

Condition: VCC = 3.0 V to 3.6 V, VSS = 0 V, ϕ = 20 MHz to 25 MHz

Item	Symbol	Min.	Max.	Unit	Test Conditions
ETCK clock cycle time	t_{TCKcyc}	40*	50*	ns	Figure 24.20
ETCK clock high pulse width	t_{TCKH}	15	—		
ETCK clock low pulse width	t_{TCKL}	15	—		
ETCK clock rise time	t_{TCKr}	—	5		
ETCK clock fall time	t_{TCKf}	—	5		
$\overline{\text{ETRST}}$ pulse width	t_{TRSTW}	20	—	t_{cyc}	Figure 24.21
Reset hold transition pulse width	t_{RSTHW}	3	—		
ETMS setup time	t_{TMSS}	20	—	ns	Figure 24.22
ETMS hold time	t_{TMSH}	20	—		
ETDI setup time	t_{TDIS}	20	—		
ETDI hold time	t_{TDIH}	20	—		
ETDO data delay time	t_{TDOD}	—	20		

Note: * When $t_{cyc} \leq t_{TCKcyc}$

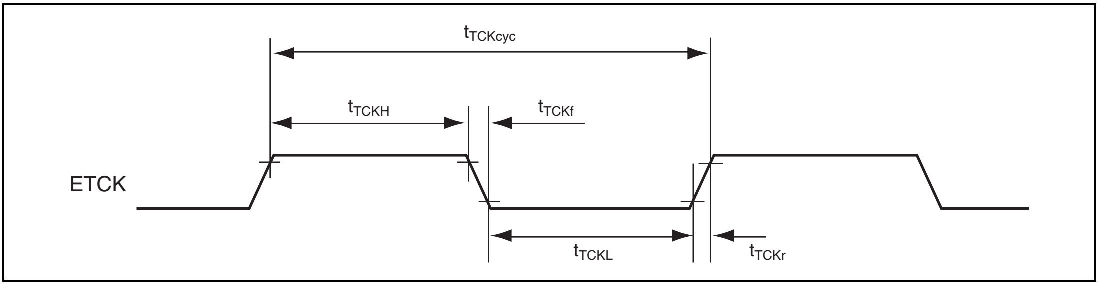


Figure 24.20 JTAG ETCK Timing

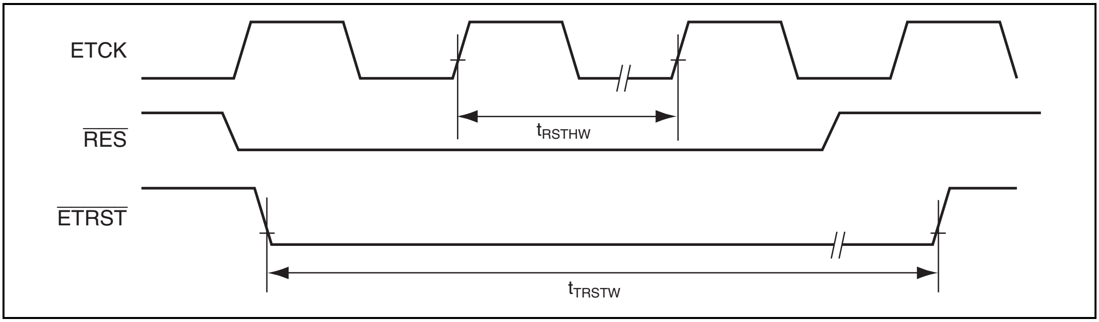


Figure 24.21 Reset Hold Timing

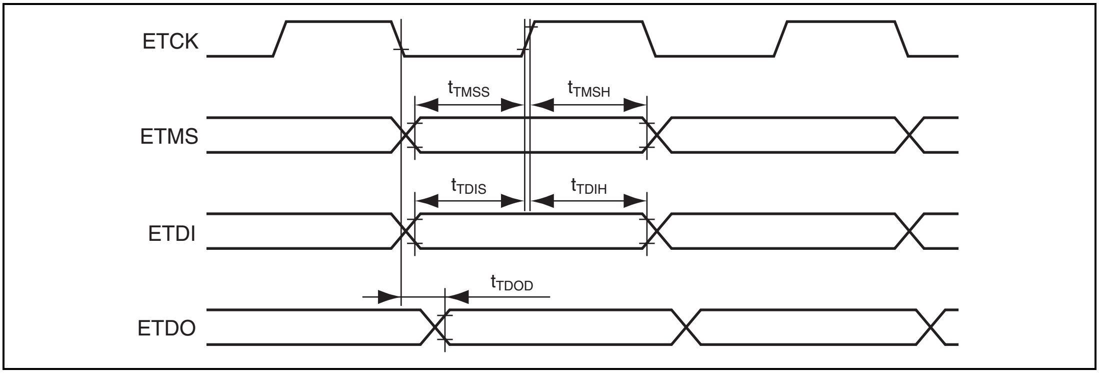


Figure 24.22 JTAG Input/Output Timing

24.4 A/D Conversion Characteristics

Table 24.12 lists the A/D conversion characteristics.

Table 24.12 A/D Conversion Characteristics
(AN7 to AN0 Input: 80/160-State Conversion)

Condition A: VCC = 3.0 V to 3.6 V, AVCC = 3.0 V to 3.6 V, AVref = 3.0 V to AVCC
VSS = AVSS = 0 V, ϕ = 20 MHz

Condition B: VCC = 3.0 V to 3.6 V, AVCC = 3.0 V to 3.6 V, AVref = 3.0 V to AVCC,
VSS = AVSS = 0 V, ϕ = 20 MHz to 25 MHz

	Condition A			Condition B			
Item	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Unit**
Resolution		10			10		Bits
Conversion time	—	—	4.0*[1]	—	—	4.7*[2]	µs
Analog input capacitance	—	—	20	—	—	20	pF
Permissible signal-source impedance	—	—	5	—	—	5	kΩ
Nonlinearity error	—	—	±7.0	—	—	±7.0	LSB
Offset error	—	—	±7.5	—	—	±7.5	
Full-scale error	—	—	±7.5	—	—	±7.5	
Quantization error	—	—	±0.5	—	—	±0.5	
Absolute accuracy	—	—	±8.0	—	—	±8.0	

Notes: 1. Value when using the maximum operating frequency in single mode of 80 states.
2. Value when using the maximum operating frequency in single mode of 160 states.

24.5 Flash Memory Characteristics

Table 24.13 and 24.14 lists the flash memory characteristics.

Table 24.13 Flash Memory Characteristics (100 Programming/Erasing Cycles Specification)

Condition: VCC = 3.0 V to 3.6 V, AVCC = 3.0 V to 3.6 V, Avref = 3.0 V to AVCC, VSS = AVSS = 0 V
Ta = 0°C to +75°C (operating temperature range for programming/erasing)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time*1*2*4	t_P	—	1	10	ms/128 bytes	
Erase time*1*2*4	t_E	—	40	130	ms/4-Kbyte block	
		—	300	800	ms/32-Kbyte block	
		—	600	1500	ms/64-Kbyte block	
Programming time (total)*1*2*4	Σt_P	—	4.5	12	s/512 Kbytes	Ta = 25°C
Erase time (total)*1*2*4	Σt_E	—	4.5	12	s/512 Kbytes	Ta = 25°C
Programming and Erase time (total)*1*2*4	Σt_{PE}	—	9.0	24	s/512 Kbytes	Ta = 25°C
Reprogramming count*5	N_{WEC}	100*3	1000	—	Times	
Data retention time*4	t_{DRP}	10	—	—	Years	

Notes: 1. Programming and erase time depends on the data.

2. Programming and erase time do not include data transfer time.

3. This value indicates the minimum number of which the flash memory are reprogrammed with all characteristics guaranteed. (The guaranteed value ranges from 1 to the minimum number.)

4. This value indicates the characteristics while the flash memory is reprogrammed within the specified range (including the minimum number).

5. Reprogramming count in each erase block.

Table 24.14 Flash Memory Characteristics (1,000 Programming/Erasing Cycles Specification)

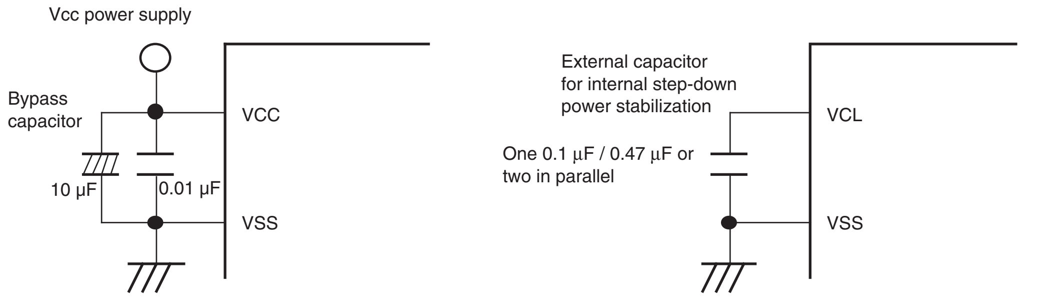
Condition: VCC = 3.0 V to 3.6 V, AVCC = 3.0 V to 3.6 V, Avref = 3.0 V to AVCC, VSS = AVSS = 0 V
Ta = 0°C to +75°C (operating temperature range for programming/erasing)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time*1*2*4	t_P	—	1	20	ms/128 bytes	
Erase time*1*2*4	t_E	—	40	260	ms/4-Kbyte block	
		—	300	1500	ms/32-Kbyte block	
		—	600	3000	ms/64-Kbyte block	
Programming time (total)*1*2*4	$\Sigma\, t_P$	—	4.5	24	s/512 Kbytes	Ta = 25°C
Erase time (total)*1*2*4	$\Sigma\, t_E$	—	4.5	24	s/512 Kbytes	Ta = 25°C
Programming and Erase time (total)*1*2*4	$\Sigma\, t_{PE}$	—	9.0	48	s/512 Kbytes	Ta = 25°C
Reprogramming count*5	N_{WEC}	1000*3	—	—	Times	
Data retention time*4	t_{DRP}	10	—	—	Years	

Notes: 1. Programming and erase time depends on the data.
2. Programming and erase time do not include data transfer time.
3. This value indicates the minimum number of which the flash memory are reprogrammed with all characteristics guaranteed. (The guaranteed value ranges from 1 to the minimum number.)
4. This value indicates the characteristics while the flash memory is reprogrammed within the specified range (including the minimum number).
5. Reprogramming count in each erase block.

24.6 Usage Notes

It is necessary to connect a bypass capacitor between the VCC pin and VSS pin and a capacitor between the VCL pin and VSS pin for stable internal step-down power. An example of connection is shown in figure 24.23.


Vcc power supply
Bypass capacitor
VCC
10 μF
0.01 μF
VSS
External capacitor for internal step-down power stabilization
One 0.1 μF / 0.47 μF or two in parallel
VCL
VSS

It is recommended that a bypass capacitor be connected to the VCC pin. (The values are reference values.)
When connecting, place a bypass capacitor near the pin.

Do not connect Vcc power supply to the VCL pin.
Always connect a capacitor for internal step-down power stabilization.
Use one or two ceramic multilayer capacitor(s) (0.1 μF / 0.47 μF: connect in parallel when using two) and place it (them) near the pin.

Figure 24.23 Connection of VCL Capacitor

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	Sleep Mode	Program Execution State
Port 1	T	T	kept	kept	I/O port
Port 2	T	T	kept	kept	I/O port
Port 3	T	T	kept	kept	I/O port
Port 4	T	T	kept	kept	I/O port
Port 57	T	T	kept	kept	I/O port
Port 56 ϕ, EXCL	T	T	[DDR = 1] : H [DDR = 0] : T	[DDR = 1] : Clock output [DDR = 0] : T	Clock output/ EXCL input/ Input port
Port 53, 52	T	T	kept	kept	I/O port
Port 6	T	T	kept	kept	I/O port
Port 7	T	T	T	T	Input port
Port 8	T	T	kept	kept	I/O port
Port A	T	T	kept	kept	I/O port
Port C	T	T	kept	kept	I/O port
Port E	T	T	kept	kept	I/O port

[Legend]

H: High level

L: Low level

T: High impedance

kept: Input port pins are in the high-impedance state (when DDR = 0 and PCR = 1, the input pull-up MOS remains on).

Output port pins retain their states.

Functions of some pins will be changed to the I/O port function, which is determined by DDR and DR, because the on-chip peripheral module associated with that pin function is initialized.

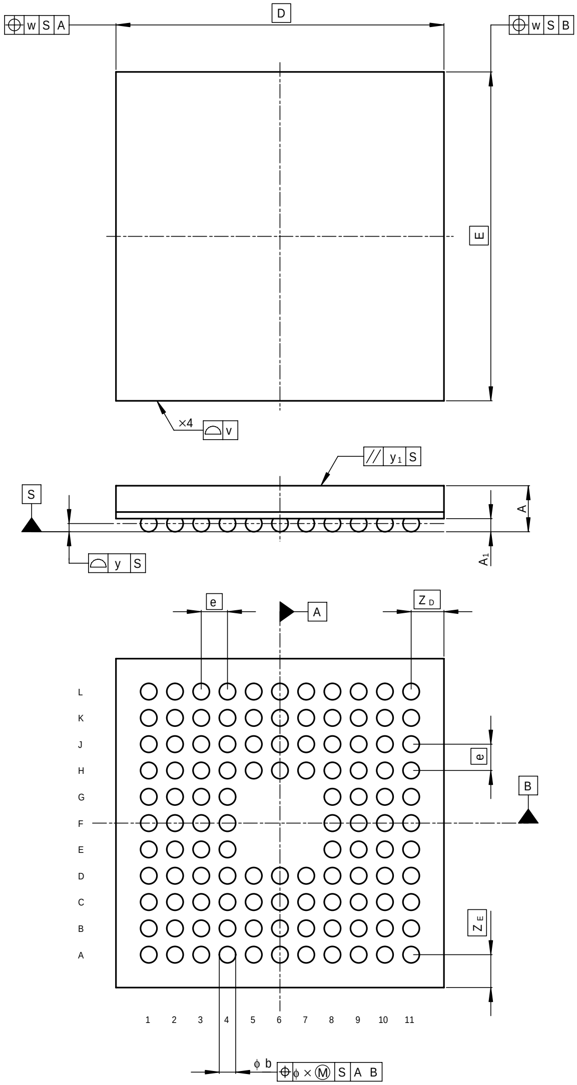
DDR: Data direction register

B. Product Lineup

Product Type		Type Code	Mark Code	Package (Code)
R4F2153	F-ZTAT version	R4F2153	F2153VBR25KDV	PLBG0112GA-A

C. Package Dimensions

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LFBGA112-10x10-0.80	PLBG0112GA-A	BP-112/BP-112V	0.3g



Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
$\boxed{D}$	——	10.00	——
$\boxed{E}$	——	10.00	——
v	——	——	0.15
w	——	——	0.20
A	——	——	1.40
A_1	0.35	0.40	0.45
$\boxed{e}$	——	0.80	——
b	0.45	0.50	0.55
x	——	——	0.08
y	——	——	0.10
y_1	——	——	0.2
$\boxed{S_D}$	——	——	——
$\boxed{S_E}$	——	——	——
$\boxed{Z_D}$	——	1.00	——
$\boxed{Z_E}$	——	1.00	——

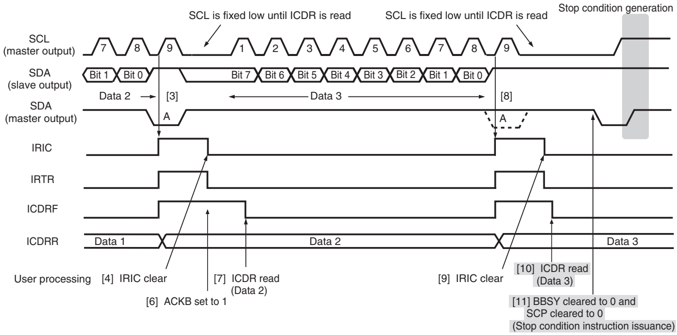
Figure C.1 Package Dimensions (PLBG0112GA-A)

Main Revisions for This Edition

<table>
<tr><th>Item</th><th>Page</th><th>Revision (See Manual for Details)</th></tr>
<tr><td>3.1 Operating Mode Selection</td><td>51</td><td>Description amended
This MCU supports three operating modes (modes 2, 4, and 6). …</td></tr>
<tr><td>Table 3.1 MCU Operating Mode Selection</td><td></td><td>Table amended
<table>
<tr><th>MCU Operating Mode</th><th>$\overline{\text{MD2}}$</th><th>MD1</th><th>CPU Operating Mode</th><th>Description</th></tr>
<tr><td>2</td><td>1</td><td>1</td><td>Advanced</td><td>Extended mode with on-chip ROM
Single-chip mode</td></tr>
<tr><td>4</td><td>0</td><td>0</td><td>—</td><td>Flash programming/erasing</td></tr>
<tr><td>6</td><td>0</td><td>1</td><td>Emulation</td><td>On-chip emulation mode</td></tr>
</table>
</td></tr>
<tr><td>5.5 Interrupt Exception Handling Vector Table
Table 5.3 Interrupt Sources, Vector Addresses, and Interrupt Priorities</td><td>77</td><td>Note added
Note: Vector numbers not listed above are reserved by the system.</td></tr>
<tr><td>Section 7 Data Transfer Controller (DTC)</td><td>97 to 124</td><td>Description amended
normal mode → normal transfer mode
repeat mode → repeat transfer mode</td></tr>
</table>

Item	Page	Revision (See Manual for Details)
7.2.5 DTC Transfer Count Register A (CRA)	102	Description amended … It is decremented by 1 every time data is transferred, and transfer ends when the count reaches H'0000. The number of times data is transferred is one when the setting value of CRA is H'0001, 65,535 when the setting value is H'FFFF, and 65,536 when the setting value is H'0000. In repeat transfer mode CRA is divided in two, with the highest eight bits designated as CRAH and the lowest eight bits as CRAL. CRAH holds the value for the number of data transfers, and CRAL functions as an 8-bit transfer counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are transferred when the counter value reaches H'00. The number of times data is transferred is one when CRAH = CRAL = H'01, 255 when CRAH = CRAL = H'FF, and 256 when CRAH = CRAL = H'00. In block transfer mode CRA is divided in two, with the highest eight bits designated as CRAH and the lowest eight bits as CRAL. CRAH holds the value for the block size, and CRAL functions as an 8-bit block size counter (1 to 256). CRAL is decremented by 1 every time data is transferred, and the contents of CRAH are transferred when the counter value reaches H'00. The block size is one byte (or one word) when CRAH = CRAL = H'01, 255 bytes (or 255 words) when CRAH = CRAL = H'FF, and 256 bytes (or 256 words) when CRAH = CRAL = H'00.
7.5 Location of Register Information and DTC Vector Table Figure 7.4 Correspondence between DTC Vector Address and Register Information	111	Newly added
10.1 Features • Special functions provided by automatic addition function	193	Description deleted

Item	Page	Revision (See Manual for Details)
15.4.4 Master Receive Operation Figure 15.12 Stop Condition Issuance Timing Example in Master Receive Mode (MLS = WAIT = 0, HNDS = 1)	368	Figure amended



16.3.2 Host Interface Control Registers 2 and 3 (HICR2 and HICR3) • HICR3	417	Table amended

			R/W		
Bit	**Bit Name**	**Initial Value**	**Slave**	**Host**	**Description**
7	LFRAME	Undefined	R	—	0: $\overline{\text{LFRAME}}$ Pin state is low level 1: $\overline{\text{LFRAME}}$ Pin state is high level
6	—	Undefined	R	—	Reserved
5	SERIRQ	Undefined	R	—	0: SERIRQ Pin state is low level 1: SERIRQ Pin state is high level
4	LRESET	Undefined	R	—	0: $\overline{\text{LRESET}}$ Pin state is low level 1: $\overline{\text{LRESET}}$ Pin state is high level
3	—	Undefined	R	—	Reserved
2	—	Undefined	R	—	Reserved
1	—	Undefined	R	—	Reserved
0	—	Undefined	R	—	Reserved

Item	Page	Revision (See Manual for Details)
24.2 DC Characteristics Table 24.2 DC Characteristics (1)	674	Table amended

Item			Symbol	Min.	Typ.	Max.	Unit
Schmitt trigger input voltage	DB7 to DB4, ExDB7 to ExDB0, EVENT7 to EVENT0, (Ex)IRQ15, (Ex)IRQ14, ExIRQ13, ExIRQ12, (Ex)IRQ11, (Ex)IRQ10, ExIRQ9, ExIRQ8, (Ex)IRQ7 to (Ex)IRQ0, ETRST, XTAL, EXCL, ADTRG	(1)	V_T^-	VCC × 0.2	—	—	V
			V_T^+	—	—	VCC × 0.7	
			$V_T^+ - V_T^-$	VCC × 0.05	—	—	
	SCL3 to SCL0, SDA3 to SDA0		V_T^-	VCC × 0.3	—	—	
			V_T^+	—	—	VCC × 0.7	
			$V_T^+ - V_T^-$	VCC × 0.05	—	—	
Input high voltage	RES, STBY, NMI, FWE, MD2, MD1	(2)	V_{IH}	VCC × 0.9	—	VCC + 0.3	V
	EXTAL			VCC × 0.7	—	VCC + 0.3	
	Port 7			2.2	—	AVCC + 0.3	
	SCL3 to SCL0, SDA3 to SDA0, Port 80 to 83, C0 to C3			—	—	5.5	
	SERIRQ, LAD3 to LAD0, LCLK, LRESET, LFRAME			VCC × 0.5	—	VCC + 0.3	
				2.0	—	VCC + 0.3	
	Input pins other than (1) and (2) above			2.2	—	VCC + 0.3	

Item	Page	Revision (See Manual for Details)
24.5 Flash Memory Characteristics Table 24.13 Flash Memory Characteristics (100 Programming/ Erasing Cycles Specification)	694	Title and Table amended

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time (total)*1*2*4	Σt_P	—	4.5	12	s/512 Kbytes	Ta = 25°C
Erase time (total)*1*2*4	Σt_E	—	4.5	12	s/512 Kbytes	Ta = 25°C
Programming and Erase time (total)*1*2*4	Σt_{PE}	—	9.0	24	s/512 Kbytes	Ta = 25°C
Reprogramming count*5	N_{WEC}	100*3	1000	—	Times	
Data retention time*4	T_{DRP}	10	—	—	Years	

Item	Page	Revision (See Manual for Details)
Table 24.14 Flash Memory Characteristics (1,000 Programming/ Erasing Cycles Specification)	695	Newly added
B. Product Lineup	698	Table amended

Product Type		Type Code	Mark Code	Package (Code)
R4F2153	F-ZTAT version	R4F2153	F2153VBR25KDV	PLBG0112GA-A

Index

Renesas 16-Bit Single-Chip Microcomputer
Hardware Manual
H8S/2153 Group

Publication Date: Rev.1.00, March 17, 2008
Rev.3.00, September 28, 2009
Published by: Sales Strategic Planning Div.
Renesas Technology Corp.
Edited by: Customer Support Department
Global Strategic Communication Div.
Renesas Solutions Corp.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7858/7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2377-3473

Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 3518-3399

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

Colophon 6.2

H8S/2153 Group
Hardware Manual

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan