Everywhere you imagine.

RENESAS

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.

32

H8SX/1544 Group

Hardware Manual

Renesas 32-Bit CISC Microcomputer
H8SX Family/H8SX/1500 Series

H8SX/1544	R5F61544
	R5F61543

Hardware Manual

Rev.2.00
Revision Date: Oct. 16, 2007

Renesas Technology
www.renesas.com

RENESAS

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.

RENESAS

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 in this Edition

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

11. Index

RENESAS

Preface

The H8SX/1544 Group is a single-chip microcomputer made up of the high-speed internal 32-bit H8SX CPU as its core, and the peripheral functions required to configure a system. The H8SX CPU is upward compatible with the H8/300, H8/300H, and H8S CPUs.

Target Users: This manual was written for users who will be using the H8SX/1544Group 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 H8SX/1544 Group to the target users.
Refer to the H8SX Family 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, and peripheral functions.
- In order to understand the details of the CPU's functions
  Read the H8SX Family 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 25, List of Registers.

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

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

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

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

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

RENESAS

H8SX/1544 Group manuals:

Document Title	Document No.
H8SX/1544 Group Hardware Manual	This manual
H8SX Family Software Manual	REJ09B0102

RENESAS

Main Revisions for This Edition

Item	Page	Revision (See Manual for Details)
All	—	Description of SSU added
		SSU: Synchronous Serial communication Unit
4.2 Exception Sources and Exception Handling Vector Table Table 4.2 Exception Handling Vector Table	81	"Reserved for system use" description in vector number 76 to 79 deleted
7.1 Features	187	Description amended • DMAC activation methods are … External request: Low level or falling edge detection of DREQ signal can be selected External request is available for all four channels (In block transfer mode only low-level detection can be selected.)
7.3.7 DMA Address Control Register (DACR)	207, 208	Bits 15 and 7 description amended (Before) extended area overflow → (After) extended repeat area overflow
7.5.2 Transfer Modes	216	(1) Normal Transfer Mode DACK signal description deleted … The TEND signal is output only in the last DMA transfer. Figure 7.7 shows an example of signal timing …
	217	Repeat Transfer Mode DACK signal description deleted … The timing of the TEND signal are the same as in normal transfer mode. …
7.5.4 Bus Access Modes	223	(2) Burst Access Mode Description amended … However, setting the IBCCS bit in BCR2 of the bus controller makes the DMAC release the bus to pass the bus to another bus master. …
7.5.9 DMA Basic Bus Cycle Figure 7.23 Example of Bus Timing of DMA Transfer	236	Figure 7.23 amended (Before) RD → (After) $\overline{\text{RD}}$

RENESAS

Item	Page	Revision (See Manual for Details)
7.8 Interrupt Sources Figure 7.38 Interrupt and Interrupt Sources	256	Figure 7.38 amended (Before) TSIE bit → (After) TSEIE bit
8.2.1 Port 1	271	(1) P17/SCL0/ $\overline{\text{IRQ7-A}}$/$\overline{\text{ADTRG1}}$ Description amended

<table>
<tr><td rowspan="3"></td><td rowspan="3"></td><td colspan="2">Setting</td></tr>
<tr><td>IIC2_0</td><td>I/O Port</td></tr>
<tr><td>Module Name</td><td>Pin Function</td><td>SCL0_OE*</td><td>P17DDR</td></tr>
<tr><td>IIC2_0</td><td>SCL0 I/O</td><td>1</td><td>—</td></tr>
<tr><td rowspan="2">I/O port</td><td>P17 output</td><td>0</td><td>1</td></tr>
<tr><td>P17 input (initial value)</td><td>0</td><td>0</td></tr>
</table>

Item	Page	Revision (See Manual for Details)
8.2.12 Port K Table 8.5 Available Output Signals and Settings in Each Port	295, 296	Table 8.5 amended

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P1	2	DACK0_A_OE	DACK0_A		DACR_0.AMS = 1, DMDR_0.DACKE = 1
		SCK2_OE	SCK2		When SCMR_2.SMIF = 1: SCR_2.TE = 1 or SCR_2.RE = 1 while SMR_2.GM = 0, SCR_2.CKE[1,0] = 01 or while SMR_2.GM = 1
					When SCMR_2.SMIF = 0: SCR_2.TE = 1 or SCR_2.RE = 1 while SMR_2.C/$\overline{\text{A}}$ = 0, SCR_2.CKE[1,0] = 01 or while SMR_2.C/$\overline{\text{A}}$ = 1, SCR_2.CKE1 = 0
P2	2	SSCK1_OE	SSCK_1		SSU.SSCRH_1.MSS1 = 1, SSU.SSCRH_1.SCKS = 1
		TIOCC3_OE	TIOCC3		TPU.TMDR_3.BFA = 0, TPU.TIORL_3.IOC3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
		TxD0_OE	TxD0		SCR_0.TE = 1
	1	SSI1_OE	SSI_1		SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 0, SSU.SSCRH_1.BIDE = 0, SSU.SSER_1.TE = 1
		TIOCA3_OE	TIOCA3		TPU.TIORH_3.IOA3 = 0, TPU.TIORH_3.IOA[1,0] = 01/10/11
	0	TIOCB3_OE	TIOCB3		TPU.TIORH_3.IOB3 = 0, TPU.TIORH_3.IOB[1,0] = 01/10/11
		SCK0_OE	SCK0		When SCMR_0.SMIF = 1: SCR_0.TE = 1 or SCR_0.RE = 1 while SMR_0.GM = 0, SCR_0.CKE[1,0] = 01 or while SMR_0.GM = 1
					When SCMR_0.SMIF = 0: SCR_0.TE = 1 or SCR_0.RE = 1 while SMR_0.C/$\overline{\text{A}}$ = 0, SCR_0.CKE[1,0] = 01 or while SMR_0.C/$\overline{\text{A}}$ = 1, SCR_0.CKE1 = 0
P3	7	SGOUT3_OE	SGOUT3		SDG3.SGCR1.SGE = 1

Item	Page	Revision (See Manual for Details)
	299, 300	Output signal names amended PI1 (Before) SSI_0 → (After) SSI0 PI0 (Before) SSO_0 → (After) SSO0

RENESAS

Item	Page	Revision (See Manual for Details)
9.9.12 Conflict between TCNT Write and Overflow/Underflow Figure 9.54 Conflict between TCNT Write and Overflow	394	Figure 9.54 amended TGR write cycle T_1 T_2 Pϕ Address — TCNT address Write — TCNT write data TCNT — H'FFFF — M TCFV flag — Prohibited →
12.9.6 Writing to Registers during SCI Transmit or Receive	486	Newly added
13.3.5 I²C Bus Status Register (ICSR)	502	Description added
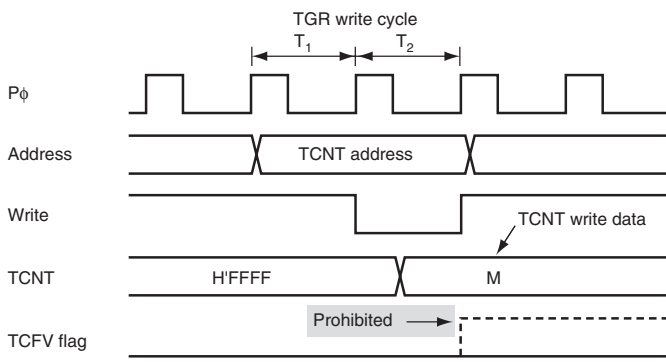
Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting condition] • When data is transferred from ICDRT to ICDRS and ICDRT becomes empty • When TRS is set to 1 • When the start condition has been issued • In slave mode, after changing from receive mode to transmit mode

Item	Page	Revision (See Manual for Details)
13.4.1 I²C Bus Format Figure 13.3 I²C Bus Formats	507	Figure 13.3 amended (a) I²C Bus Format m: Transfer frame count (m ≥1) I²C Bus Format (start condition retransmission) m1 and m2: 1: Transfer frame count (m1 and m2 ≥1)
13.4.4 Slave Transmit Operation	512	Description amended 1. … ICCRA to 1. Set the WAIT in ICMR and CKS3 to CKS0 in ICCRA, and perform other initial settings. …
13.4.5 Slave Receive Operation	515	Description amended 1. … ICCRA to 1. Set the WAIT in ICMR and CKS3 to CKS0 in ICCRA, and perform other initial settings. …
13.5 Interrupt Request Table 13.3 Interrupt Requests	521	Table 13.3 amended Abbreviation of NACK detection and arbitration lost amended (Before) MAKI → (After) NAKI

RENESAS

Item	Page	Revision (See Manual for Details)
14.2.3 Input/Output Pins Table 14.1 Pin Configuration of the RCAN-ET	527	Table 14.1 amended

Channel	Name	Abbreviation	I/O	Function
0	Transmit data pin	CTx_0	Output	CAN-bus transmit pin
	Receive data pin	CRx_0	Input	CAN-bus receive pin
1	Transmit data pin	CTx_1	Output	CAN-bus transmit pin
	Receive data pin	CRx_1	Input	CAN-bus receive pin

Item	Page	Revision (See Manual for Details)
14.2.4 Memory Map Figure 14.2 RCAN-ET Memory Map	528	Note added Note: The locations not used (between H'000 and H'2F2) are reserved and cannot be accessed. Addresses shown above are offset addresses. As for actual addresses, see section 25, List of Registers.
14.3.1 Mail Box Structure Table 14.2 Address Map for Each Mailbox	529	Table 14.2 amended

	Address			
	Control 0	LAFM	Data	Control 1
Mailbox	4 bytes	4 bytes	8 bytes	2 bytes
0 (Receive Only)	H'100 to H'103	H'104 to H'107	H'108 to H'10F	H'110 to H'111
1	H'120 to H'123	H'124 to H'127	H'128 to H'12F	H'130 to H'131
2	H'140 to H'143	H'144 to H'147	H'148 to H'14F	H'150 to H'151
3	H'160 to H'163	H'164 to H'167	H'168 to H'16F	H'170 to H'171
4	H'180 to H'183	H'184 to H'187	H'188 to H'18F	H'190 to H'191
5	H'1A0 to H'1A3	H'1A4 to H'1A7	H'1A8 to H'1AF	H'1B0 to H'1B1
6	H'1C0 to H'1C3	H'1C4 to H'1C7	H'1C8 to H'1CF	H'1D0 to H'1D1
7	H'1E0 to H'1E3	H'1E4 to H'1E7	H'1E8 to H'1EF	H'1F0 to H'1F1
8	H'200 to H'203	H'204 to H'207	H'208 to H'20F	H'210 to H'211
9	H'220 to H'223	H'224 to H'227	H'228 to H'22F	H'230 to H'231
10	H'240 to H'243	H'244 to H'247	H'248 to H'24F	H'250 to H'251
11	H'260 to H'263	H'264 to H'267	H'268 to H'26F	H'270 to H'271
12	H'280 to H'283	H'284 to H'287	H'288 to H'28F	H'290 to H'291
13	H'2A0 to H'2A3	H'2A4 to H'2A7	H'2A8 to H'2AF	H'2B0 to H'2B1
14	H'2C0 to H'2C3	H'2C4 to H'2C7	H'2C8 to H'2CF	H'2D0 to H'2D1
15	H'2E0 to H'2E3	H'2E4 to H'2E7	H'2E8 to H'2EF	H'2F0 to H'2F1

RENESAS

Item	Page	Revision (See Manual for Details)
14.3.1 Mail Box Structure Figure 14.3 Mailbox-n Structure	530	Figure 14.3 amended

MB0 (reception MB)

Regiter Name	Address
MB[0].CONTROL0H	H'100
MB[0].CONTROL0L	H'102
MB[0].LAFMH	H'104
MB[0].LAFML	H'106
MB[0].MSG_DATA[0][1]	H'108
MB[0].MSG_DATA[2][3]	H'10A
MB[0].MSG_DATA[4][5]	H'10C
MB[0].MSG_DATA[6][7]	H'10E
MB[0].CONTROL1H, L	H'110

MB1 to 15 (MB for transmission/reception)

Register Name	Address
MB[n].CONTROL0H	H'100 + n*32
MB[n].CONTROL0L	H'102 + n*32
MB[n].LAFMH	H'104 + n*32
MB[n].LAFML	H'106 + n*32
MB[n].MSG_DATA[0][1]	H'108 + n*32
MB[n].MSG_DATA[2][3]	H'10A + n*32
MB[n].MSG_DATA[4][5]	H'10C + n*32
MB[n].MSG_DATA[6][7]	H'10E + n*32
MB[n].CONTROL1H, L	H'110 + n*32

Note amended

Notes: n = 1 to 15 (Mailbox number)

1. All bits shadowed in grey are reserved and the write value should be 0. The value returned by a read may not always be 0 and should not be relied upon.

2 MBC1 bit in mailbox is fixed to 1.

3. ATX and DART are not supported by mailbox-0, and the MBC setting of mailbox-0 is limited.

4. When the MCR15 bit is 1, the order of STDID, RTR, IDE and EXTID of both message control and LAFM differs from HCAN2.

Item	Page	Revision (See Manual for Details)
14.4 Local Acceptance Filter Mask (LAFM) Figure 14.4 Acceptance Filter	535	Register names amended MB[N].LAFMH MB[N].LAFML
		Note added Note: N = 0 to 15 (Mailbox number)
14.4.1 Master Control Register (MCR) Figure 14.5 ID Reorder	538	Note added Note: N = 0 to 15 (Mailbox number)

Item	Page	Revision (See Manual for Details)
14.6.3 Message Transmission Sequence Figure 14.9 Transmission Request	577	Figure 14.9 amended Note added Note: n = 1 to 15 (Mailbox number)
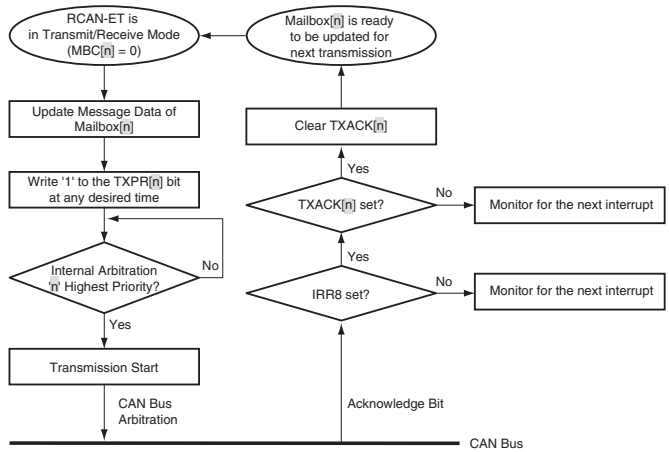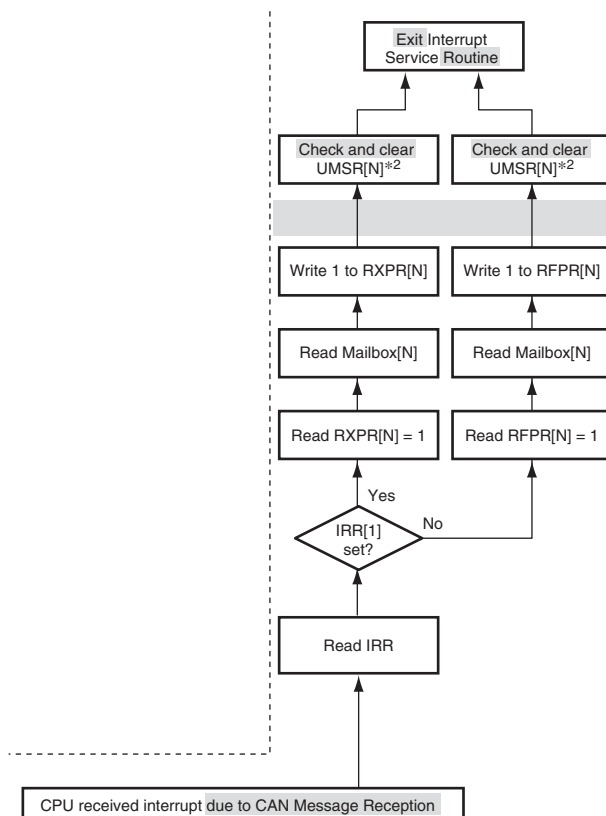
RENESAS

Item	Page	Revision (See Manual for Details)
14.6.4 Message Receive Sequence Figure 14.11 Message Receive Sequence	579	Figure 14.11 amended

Exit Interrupt Service Routine

Check and clear UMSR[N]*2 | Check and clear UMSR[N]*2

Write 1 to RXPR[N] | Write 1 to RFPR[N]

Read Mailbox[N] | Read Mailbox[N]

Read RXPR[N] = 1 | Read RFPR[N] = 1

Yes

IRR[1] set? No

Read IRR

CPU received interrupt due to CAN Message Reception

Notes: N = 0 to 15 (Mailbox number)

1. Only if CPU clears RXPR[N]/RFPR[N] at the same time that UMSR is set in overrun, RXPR[N]/RFPR[N] may be set again even though the message has not been updated.

2. In case overwrite configuration (NMC = 1) is used for the Mailbox N the message must be discarded when UMSR[N] = 1, UMSR[N] cleared and the full Interrupt Service Routine started again. In case of overrun configuration (NMC = 0) is used clear again RXPR[N]/RFPR[N]/UMSR[N] when UMSR[N] = 1 and consider the message obsolete.

RENESAS

Item	Page	Revision (See Manual for Details)
14.7 Interrupt Sources Table 14.10 RCAN-ET Interrupt Sources	583	Table 14.10 amended

Channel	Interrupt	Description	Interrupt Flag	DMAC Activation
0	ERS_0	Error Passive Mode (TEC ≥128 or REC ≥128)	IRR5	Not possible
		Bus Off (TEC ≥256)/Bus Off recovery	IRR6	
		Error warning (TEC ≥96)	IRR3	
		Error warning (REC ≥96)	IRR4	
	OVR_0	Message error detection	IRR13[*1]	
		Reset/halt/CAN sleep transition	IRR0	
		Overload frame transmission	IRR7	
		Unread message overwrite (overrun)	IRR9	
		Detection of CAN bus operation in CAN sleep mode	IRR12	
	SLE_0	Message transmission/transmission disabled (slot empty)	IRR8	
	RM1_0[*2]	Data frame reception/	IRR1[*3]	
	RM0_0[*2]	Remote frame reception	IRR2[*3]	Possible
1	ERS_1	Error Passive Mode (TEC ≥128 or REC ≥128)	IRR5	Not possible
		Bus Off (TEC ≥256)/Bus Off recovery	IRR6	
		Error warning (TEC ≥96)	IRR3	
		Error warning (REC ≥96)	IRR4	
	OVR_1	Message error detection	IRR13[*1]	
		Reset/halt/CAN sleep transition	IRR0	
		Overload frame transmission	IRR7	
		Unread message overwrite (overrun)	IRR9	
		Detection of CAN bus operation in CAN sleep mode	IRR12	
	SLE_1	Message transmission/transmission disabled (slot empty)	IRR8	
	RM1_1[*2]	Data frame reception/	IRR1[*3]	
	RM0_1[*2]	Remote frame reception	IRR2[*3]	Possible

Item	Page	Revision (See Manual for Details)
15.3.7 SS Transmit Data Registers 0 to 3 (SSTDR0 to SSTDR3)	604	Bit figure amended

• SSTDR0

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

• SSTDR1

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

• SSTDR2

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

• SSTDR3

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

RENESAS

Item	Page	Revision (See Manual for Details)
15.3.8 SS Receive Data Registers 0 to 3 (SSRDR0 to SSRDR3)	605	Description amended … Be sure not to access to invalid SSRDR. When the SSU has received 1-byte data, …
		Bit figure amended
15.4.5 SSU Mode Figure 15.6 Flowchart Example of Data Transmission (SSU Mode)	615	Figure 15.6 amended [2] Check that the SSU state and write transmit data: Write transmit data to SSTDR after reading and confirming that the TDRE bit in SSSR is 1. …
15.4.7 Clock Synchronous Communication Mode Figure 15.14 Flowchart Example of Transmission Operation (Clock Synchronous Communication Mode)	624	Figure 15.14 amended [2] Check that the SSU state and write transmit data: Write transmit data to SSTDR after reading and confirming that the TDRE bit in SSSR is 1. …
18.3.6 PWM Duty Registers A, C, E, G (PWDTRA, PWDTRC, PWDTRE, PWDTRG)	662	Description added …, data is transferred from the buffer register (PWBFR) to the duty register (PWDTR). PWDTR is initialized to H'00 when CST bit is 0.
18.4.2 8-Bit Data Registers	667	Description amended …; in this case, the lower eight bits are read as H'FF.

Item	Page	Revision (See Manual for Details)
22.6.1 Programming/ Erasing Interface Registers	717	(5) Flash Transfer Destination Address Register (FTDAR)
		Bit 7 description amended
		1: The value specified by bits TDER and TDA6 to TDA0 is between H'03 and H'FF and download has stopped
22.6.2 Programming/ Erasing Interface Parameters	725	(3) Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU)
		Description amended
		… The operating frequency available in this LSI ranges from 8 MHz to 40 MHz.
22.11 Standard Serial Communication Interface Specifications for Boot Mode (3) Inquiry and Selection States (h) Erased Block Information Inquiry	766	Description amended
		• Size (two bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
24.1 Features Table 24.1 Operating States	798	Note 4 amended
		Notes: 4. Some SCI registers, the motor control PWM, 16-bit PWM, RCAN-ET, SSU∗, and SDG are reset. Other peripheral modules retain their states.
25.1 Register Addresses (Address Order)	843	Note 3 amended
		Notes: 3. PWM10 is Motor Control PWM.
25.2 Register Bits	862	Note 1 amended and note 3 added
		Notes: 1. SSU: Synchronous Serial communication Unit 2. PWM16 is 16-bit PWM. 3. PWM10 is Motor Control PWM.
25.3 Register States in Each Operating Mode	862	Note 1 amended and note 3 added
		Notes: 1. SSU: Synchronous Serial communication Unit 2. PWM16 is 16-bit PWM. 3. PWM10 is Motor Control PWM.

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

RENESAS

Contents

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

Figures

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

Section 16 A/D Converter

Section 17 D/A Converter

Section 18 Motor Control PWM Timer (PWM)

RENESAS

Appendix

Tables

RENESAS

RENESAS

Section 23   Clock Pulse Generator

Section 24   Power-Down Modes

Section 26   Electrical Characteristics

RENESAS

Appendix

RENESAS

RENESAS

Section 1   Overview

1.1   Features

- 32-bit high-speed H8SX CPU

  Upward compatible at object level with the H8/300 CPU, H8/300H CPU, and H8S CPU

  Sixteen 16-bit general registers

  87 basic instructions

- Extensive peripheral functions

  DMA controller (DMAC)

  16-bit timer pulse unit (TPU)

  Watchdog timer (WDT)

  Watch timer (WAT)

  Motor control PWM timer

  16-bit PWM timer

  Sound generator (SDG)

  Asynchronous or clocked-synchronous serial communication interface (SCI)

  I^2C bus interface 2 (IIC2)

  Controller area network (RCAN-ET)

  Synchronous serial communication unit (SSU)

  10-bit A/D converter

  8-bit D/A converter

  Clock pulse generator

- On-chip memory

Product Classification		Part No.	ROM	RAM
Flash memory version	H8SX/1544	R5F61544	512 kbytes	24 kbytes
	H8SX/1543	R5F61543	384 kbytes	16 kbytes

- General I/O port

  95 input/output ports

  17 input ports

- Supports a variety of power-down modes

• Small package

Package	Code	Body Size	Pin Pitch
LQFP-144	FP-144L	20.0 × 20.0 mm	0.50 mm

1.2   Block Diagram
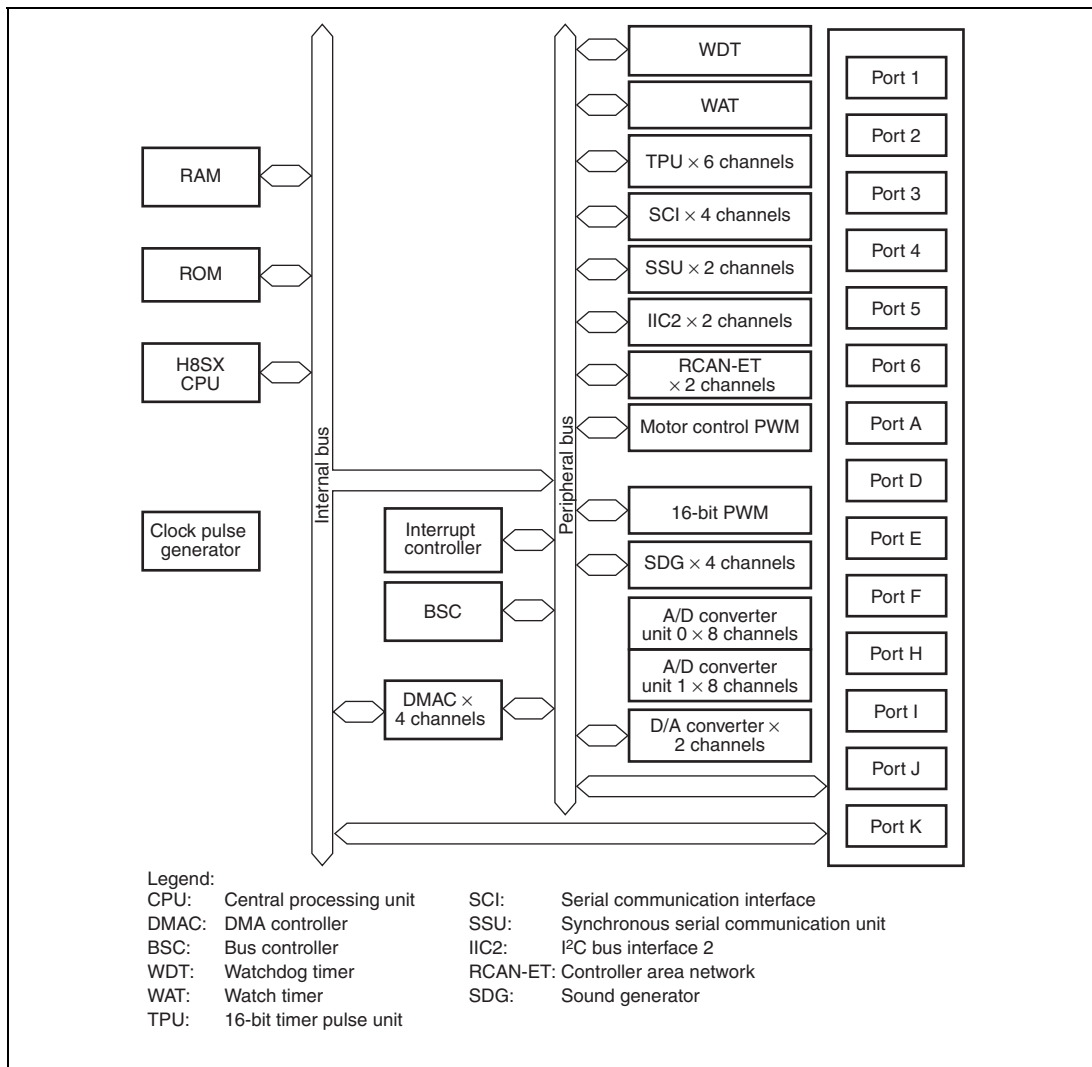
Figure 1.1   Block Diagram of H8SX/1544

RENESAS

1.3   Pin Assignments

1.3.1   Pin Assignments

Note: *   Emulator enable pin.
       This pin should be low in normal operating modes. When the EMLE pin is driven high,
       the emulation function is enabled and the TDO, TDI, TCK, TMS, and TRST pins are only used
       specific for the on-chip emulator E10A.
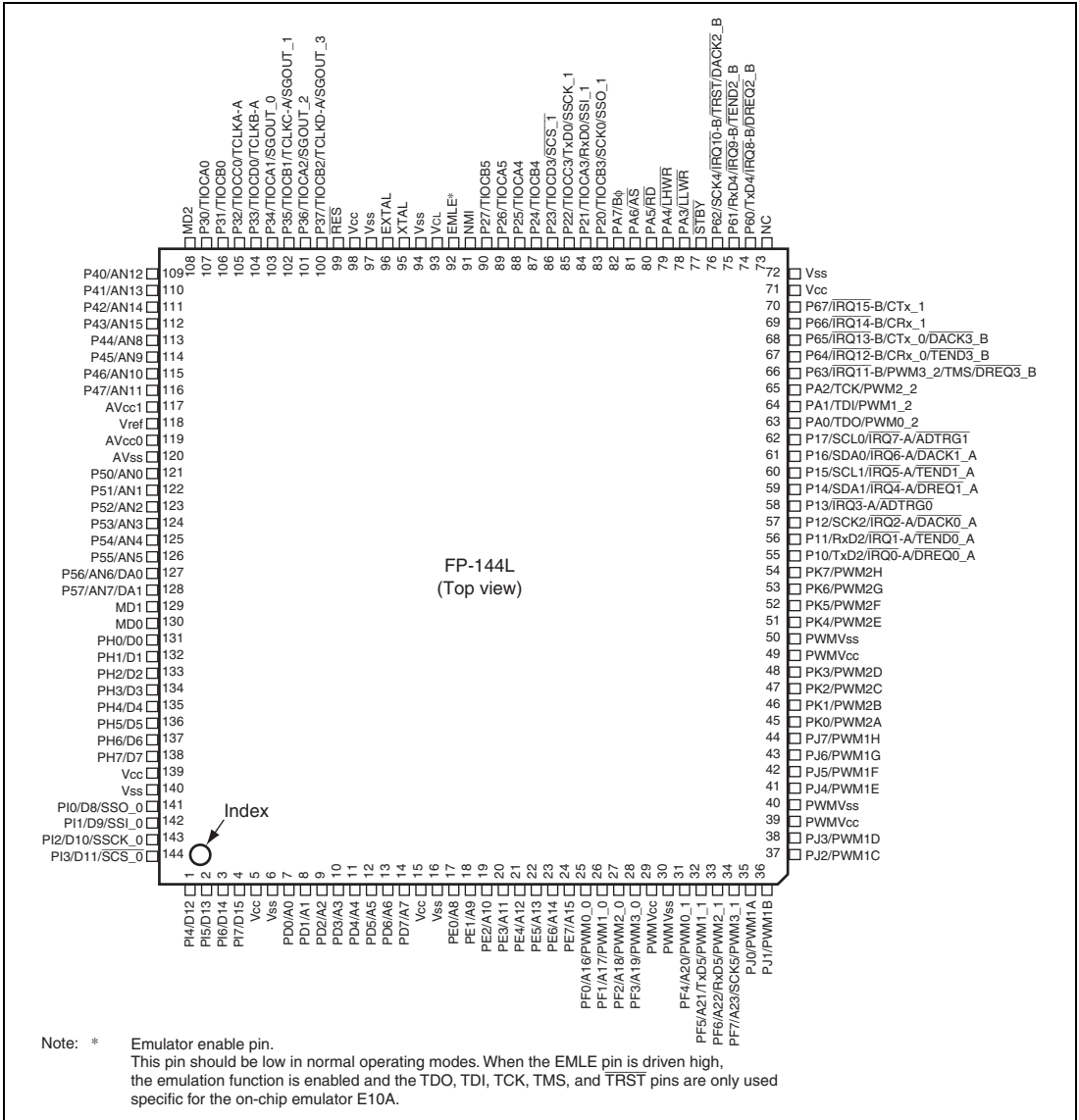
Figure 1.2   Pin Assignments of H8SX/1544

RENESAS

1.3.2   Pin Configuration in Each Operating Mode

Table 1.1   Pin Configuration in Each Operating Mode

Pin No.	Abbreviation in Mode 2, Mode 6, and Mode 7	Abbreviation in Mode 4 and Mode 5
1	PI4/D12	PI4/D12
2	PI5/D13	PI5/D13
3	PI6/D14	PI6/D14
4	PI7/D15	PI7/D15
5	Vcc	Vcc
6	Vss	Vss
7	PD0/A0	A0
8	PD1/A1	A1
9	PD2/A2	A2
10	PD3/A3	A3
11	PD4/A4	A4
12	PD5/A5	A5
13	PD6/A6	A6
14	PD7/A7	A7
15	Vcc	Vcc
16	Vss	Vss
17	PE0/A8	A8
18	PE1/A9	A9
19	PE2/A10	A10
20	PE3/A11	A11
21	PE4/A12	A12
22	PE5/A13	A13
23	PE6/A14	A14
24	PE7/A15	A15
25	PF0/A16/PWM0_0	PF0/A16/PWM0_0
26	PF1/A17/PWM1_0	PF1/A17/PWM1_0
27	PF2/A18/PWM2_0	PF2/A18/PWM2_0
28	PF3/A19/PWM3_0	PF3/A19/PWM3_0

RENESAS

Pin No.	Abbreviation in Mode 2, Mode 6, and Mode 7	Abbreviation in Mode 4 and Mode 5
29	PWMVcc	PWMVcc
30	PWMVss	PWMVss
31	PF4/A20/PWM0_1	PF4/A20/PWM0_1
32	PF5/A21/TxD5/PWM1_1	PF5/A21/TxD5/PWM1_1
33	PF6/A22/RxD5/PWM2_1	PF6/A22/RxD5/PWM2_1
34	PF7/A23/SCK5/PWM3_1	PF7/A23/SCK5/PWM3_1
35	PJ0/PWM1A	PJ0/PWM1A
36	PJ1/PWM1B	PJ1/PWM1B
37	PJ2/PWM1C	PJ2/PWM1C
38	PJ3/PWM1D	PJ3/PWM1D
39	PWMVcc	PWMVcc
40	PWMVss	PWMVss
41	PJ4/PWM1E	PJ4/PWM1E
42	PJ5/PWM1F	PJ5/PWM1F
43	PJ6/PWM1G	PJ6/PWM1G
44	PJ7/PWM1H	PJ7/PWM1H
45	PK0/PWM2A	PK0/PWM2A
46	PK1/PWM2B	PK1/PWM2B
47	PK2/PWM2C	PK2/PWM2C
48	PK3/PWM2D	PK3/PWM2D
49	PWMVcc	PWMVcc
50	PWMVss	PWMVss
51	PK4/PWM2E	PK4/PWM2E
52	PK5/PWM2F	PK5/PWM2F
53	PK6/PWM2G	PK6/PWM2G
54	PK7/PWM2H	PK7/PWM2H
55	P10/TxD2/$\overline{\text{IRQ0}}$-A/$\overline{\text{DREQ0}}$_A	P10/TxD2/$\overline{\text{IRQ0}}$-A/$\overline{\text{DREQ0}}$_A
56	P11/RxD2/$\overline{\text{IRQ1}}$-A/$\overline{\text{TEND0}}$_A	P11/RxD2/$\overline{\text{IRQ1}}$-A/$\overline{\text{TEND0}}$_A
57	P12/SCK2/$\overline{\text{IRQ2}}$-A/$\overline{\text{DACK0}}$_A	P12/SCK2/$\overline{\text{IRQ2}}$-A/$\overline{\text{DACK0}}$_A
58	P13/$\overline{\text{IRQ3}}$-A/$\overline{\text{ADTRG0}}$	P13/$\overline{\text{IRQ3}}$-A/$\overline{\text{ADTRG0}}$

RENESAS

Pin No.	Abbreviation in Mode 2, Mode 6, and Mode 7	Abbreviation in Mode 4 and Mode 5
59	P14/SDA1/$\overline{\text{IRQ4}}$-A/$\overline{\text{DREQ1}}$_A	P14/SDA1/$\overline{\text{IRQ4}}$-A/$\overline{\text{DREQ1}}$_A
60	P15/SCL1/$\overline{\text{IRQ5}}$-A/$\overline{\text{TEND1}}$_A	P15/SCL1/$\overline{\text{IRQ5}}$-A/$\overline{\text{TEND1}}$_A
61	P16/SDA0/$\overline{\text{IRQ6}}$-A/$\overline{\text{DACK1}}$_A	P16/SDA0/$\overline{\text{IRQ6}}$-A/$\overline{\text{DACK1}}$_A
62	P17/SCL0/$\overline{\text{IRQ7}}$-A/$\overline{\text{ADTRG1}}$	P17/SCL0/$\overline{\text{IRQ7}}$-A/$\overline{\text{ADTRG1}}$
63	PA0/TDO/PWM0_2	PA0/TDO/PWM0_2
64	PA1/TDI/PWM1_2	PA1/TDI/PWM1_2
65	PA2/TCK/PWM2_2	PA2/TCK/PWM2_2
66	P63/$\overline{\text{IRQ11}}$-B/PWM3_2/TMS/$\overline{\text{DREQ3}}$_B	P63/$\overline{\text{IRQ11}}$-B/PWM3_2/TMS/$\overline{\text{DREQ3}}$_B
67	P64/$\overline{\text{IRQ12}}$-B/CRx_0/$\overline{\text{TEND3}}$_B	P64/$\overline{\text{IRQ12}}$-B/CRx_0/$\overline{\text{TEND3}}$_B
68	P65/$\overline{\text{IRQ13}}$-B/CTx_0/$\overline{\text{DACK3}}$_B	P65/$\overline{\text{IRQ13}}$-B/CTx_0/$\overline{\text{DACK3}}$_B
69	P66/$\overline{\text{IRQ14}}$-B/CRx_1	P66/$\overline{\text{IRQ14}}$-B/CRx_1
70	P67/$\overline{\text{IRQ15}}$-B/CTx_1	P67/$\overline{\text{IRQ15}}$-B/CTx_1
71	Vcc	Vcc
72	Vss	Vss
73	NC	NC
74	P60/TxD4/$\overline{\text{IRQ8}}$-B/$\overline{\text{DREQ2}}$_B	P60/TxD4/$\overline{\text{IRQ8}}$-B/$\overline{\text{DREQ2}}$_B
75	P61/RxD4/$\overline{\text{IRQ9}}$-B/$\overline{\text{TEND2}}$_B	P61/RxD4/$\overline{\text{IRQ9}}$-B/$\overline{\text{TEND2}}$_B
76	P62/SCK4/$\overline{\text{IRQ10}}$-B/$\overline{\text{TRST}}$/$\overline{\text{DACK2}}$_B	P62/SCK4/$\overline{\text{IRQ10}}$-B/$\overline{\text{TRST}}$/$\overline{\text{DACK2}}$_B
77	$\overline{\text{STBY}}$	$\overline{\text{STBY}}$
78	PA3/$\overline{\text{LLWR}}$	PA3/$\overline{\text{LLWR}}$
79	PA4/$\overline{\text{LHWR}}$	PA4/$\overline{\text{LHWR}}$
80	PA5/$\overline{\text{RD}}$	PA5/$\overline{\text{RD}}$
81	PA6/$\overline{\text{AS}}$	PA6/$\overline{\text{AS}}$
82	PA7/Bϕ	PA7/Bϕ
83	P20/TIOCB3/SCK0/SSO_1	P20/TIOCB3/SCK0/SSO_1
84	P21/TIOCA3/RxD0/SSI_1	P21/TIOCA3/RxD0/SSI_1
85	P22/TIOCC3/TxD0/SSCK_1	P22/TIOCC3/TxD0/SSCK_1
86	P23/TIOCD3/$\overline{\text{SCS}}$_1	P23/TIOCD3/$\overline{\text{SCS}}$_1
87	P24/TIOCB4	P24/TIOCB4
88	P25/TIOCA4	P25/TIOCA4

RENESAS

Pin No.	Abbreviation in Mode 2, Mode 6, and Mode 7	Abbreviation in Mode 4 and Mode 5
89	P26/TIOCA5	P26/TIOCA5
90	P27/TIOCB5	P27/TIOCB5
91	NMI	NMI
92	EMLE*	EMLE*
93	V_{CL}	V_{CL}
94	Vss	Vss
95	XTAL	XTAL
96	EXTAL	EXTAL
97	Vss	Vss
98	Vcc	Vcc
99	$\overline{\text{RES}}$	$\overline{\text{RES}}$
100	P37/TIOCB2/TCLKD_A/SGOUT_3	P37/TIOCB2/TCLKD_A/SGOUT_3
101	P36/TIOCA2/SGOUT_2	P36/TIOCA2/SGOUT_2
102	P35/TIOCB1/TCLKC_A/SGOUT_1	P35/TIOCB1/TCLKC_A/SGOUT_1
103	P34/TIOCA1/SGOUT_0	P34/TIOCA1/SGOUT_0
104	P33/TIOCD0/TCLKB-A	P33/TIOCD0/TCLKB-A
105	P32/TIOCC0/TCLKA-A	P32/TIOCC0/TCLKA-A
106	P31/TIOCB0	P31/TIOCB0
107	P30/TIOCA0	P30/TIOCA0
108	MD2	MD2
109	P40/AN12	P40/AN12
110	P41/AN13	P41/AN13
111	P42/AN14	P42/AN14
112	P43/AN15	P43/AN15
113	P44/AN8	P44/AN8
114	P45/AN9	P45/AN9
115	P46/AN10	P46/AN10
116	P47/AN11	P47/AN11
117	AVcc1	AVcc1
118	V_{ref}	V_{ref}

Pin No.	Abbreviation in Mode 2, Mode 6, and Mode 7	Abbreviation in Mode 4 and Mode 5
119	AVcc0	AVcc0
120	AVss	AVss
121	P50/AN0	P50/AN0
122	P51/AN1	P51/AN1
123	P52/AN2	P52/AN2
124	P53/AN3	P53/AN3
125	P54/AN4	P54/AN4
126	P55/AN5	P55/AN5
127	P56/AN6/DA0	P56/AN6/DA0
128	P57/AN7/DA1	P57/AN7/DA1
129	MD1	MD1
130	MD0	MD0
131	PH0/D0	D0
132	PH1/D1	D1
133	PH2/D2	D2
134	PH3/D3	D3
135	PH4/D4	D4
136	PH5/D5	D5
137	PH6/D6	D6
138	PH7/D7	D7
139	Vcc	Vcc
140	Vss	Vss
141	PI0/D8/SSO_0	PI0/D8/SSO_0
142	PI1/D9/SSI_0	PI1/D9/SSI_0
143	PI2/D10/SSCK_0	PI2/D10/SSCK_0
144	PI3/D11/$\overline{SCS\_0}$	PI3/D11/$\overline{SCS\_0}$

Note: *   The EMLE (emulator enable) pin enables/disables the on-chip emulation function. This pin should be low in the normal operating modes. When the pin is driven high, the emulation function is enabled and the TDO, TDI, TCK, TMS, and $\overline{TRST}$ pins are only used for the on-chip emulator E10A.

RENESAS

1.3.3   Pin Functions

Table 1.2   Pin Functions

Classification	Abbreviation	Pin Number	I/O	Description
Power supply	V_{CC}	5, 15, 71, 98, 139	Input	Power supply pins Connect to the system power supply.
	V_{CL}	93	Input	Connect to Vss via a 0.1 μF capacitor, which should be placed close to this pin.
	V_{SS}	6, 16, 72 94, 97, 140	Input	Ground pins Connect to the system power supply (0 V).
Clock	XTAL	95	Input	These pins connect to crystal resonator.
	EXTAL	96	Input	EXTAL pin can be used to input external clock. For a connection example, see section 23, Clock Pulse Generator.
	Bφ	82	Output	Supplies the system clock to external devices.
Operating mode control	MD2	108	Input	These pins set the operating mode. The signal levels of the pins must not be changed during operation.
	MD1	129		
	MD0	130		
System control	$\overline{\text{RES}}$	99	Input	Reset signal input pin This LSI enters a reset state when the level on this pin goes low.
	$\overline{\text{STBY}}$	77	Input	This LSI enters hardware standby mode when the level on this pin goes low.
	EMLE	92	Input	Enables the on-chip emulator. Usually, the signal level should be fixed low.
Address bus	A23 to A20	34 to 31	Output	Address output pins
	A19 to A8	28 to 17		
	A7 to A0	14 to 7		
Data bus	D15 to D8	4 to 1, 144 to 141	I/O	Bidirectional data bus
	D7 to D0	138 to 131		

Classification	Abbreviation	Pin Number	I/O	Description
Bus control	$\overline{\text{RD}}$	80	Output	External address space is being read when the level on this pin is low.
	$\overline{\text{AS}}$	81	Output	Address outputs on the address bus are valid when the level on this pin is low.
	$\overline{\text{LHWR}}$	79	Output	External address space is being written to when the level on this pin is low. The upper half of the data bus is valid.
	$\overline{\text{LLWR}}$	78	Output	External address space is being written to when the level on this pin is low. The lower half of the data bus is valid.
Interrupts	NMI	91	Input	Non-maskable interrupt request pin
				When this pin is not used, this signal must be fixed high.
	$\overline{\text{IRQ15}}$-B	70	Input	Maskable interrupt request pins
	$\overline{\text{IRQ14}}$-B	69		
	$\overline{\text{IRQ13}}$-B	68		
	$\overline{\text{IRQ12}}$-B	67		
	$\overline{\text{IRQ11}}$-B	66		
	$\overline{\text{IRQ10}}$-B	76		
	$\overline{\text{IRQ9}}$-B	75		
	$\overline{\text{IRQ8}}$-B	74		
	$\overline{\text{IRQ7}}$-A	62		
	$\overline{\text{IRQ6}}$-A	61		
	$\overline{\text{IRQ5}}$-A	60		
	$\overline{\text{IRQ4}}$-A	59		
	$\overline{\text{IRQ3}}$-A	58		
	$\overline{\text{IRQ2}}$-A	57		
	$\overline{\text{IRQ1}}$-A	56		
	$\overline{\text{IRQ0}}$-A	55		
On-chip emulator	$\overline{\text{TRST}}$	76	Input	Interface pins for debugging with the on-chip emulator
	TMS	66	Input	
	TDO	63	Output	
	TDI	64	Input	
	TCK	65	Input	

RENESAS

Classification	Abbreviation	Pin Number	I/O	Description
DMA controller (DMAC)	$\overline{\text{DREQ0}}$_A	55	Input	Pins for requesting DMAC to activate
	$\overline{\text{DREQ1}}$_A	59		
	$\overline{\text{DREQ2}}$_B	74		
	$\overline{\text{DREQ3}}$_B	66		
	$\overline{\text{DACK0}}$_A	57	Output	Single address transfer acknowledge pins for DMAC
	$\overline{\text{DACK1}}$_A	61		
	$\overline{\text{DACK2}}$_B	76		
	$\overline{\text{DACK3}}$_B	68		
	$\overline{\text{TEND0}}$_A	56	Output	These pins indicate the transfer completion of DMAC data.
	$\overline{\text{TEND1}}$_A	60		
	$\overline{\text{TEND2}}$_B	75		
	$\overline{\text{TEND3}}$_B	67		
16-bit timer pulse unit (TPU) (unit 0)	TCLKA-A	105	Input	These pins supply the external clocks.
	TCLKB-A	104		
	TCLKC-A	102		
	TCLKD-A	100		
	TIOCA0	107	I/O	These pins are used for TGRA_0 to TGRD_0 input-capture inputs, output-compare outputs, and PWM outputs.
	TIOCB0	106		
	TIOCC0	105		
	TIOCD0	104		
	TIOCA1	103	I/O	These pins are used for TGRA_1 and TGRB_1 input-capture inputs, output-compare outputs, and PWM outputs.
	TIOCB1	102		
	TIOCA2	101	I/O	These pins are used for TGRA_2 and TGRB_2 input-capture inputs, output-compare outputs, and PWM outputs.
	TIOCB2	100		
	TIOCA3	84	I/O	These pins are used for TGRA_3 toTGRD_3 input-capture inputs, output-compare outputs, and PWM outputs.
	TIOCB3	83		
	TIOCC3	85		
	TIOCD3	86		
	TIOCA4	88	I/O	These pins are used for TGRA_4 and TGRB_4 input-capture inputs, output-compare outputs, and PWM outputs.
	TIOCB4	87		
	TIOCA5	89	I/O	These pins are used for TGRA_5 and TGRB_5 input-capture inputs, output-compare outputs, and PWM outputs.
	TIOCB5	90		

Classification	Abbreviation	Pin Number	I/O	Description
Motor control PWM timer	PWM1A to PWM1H	35 to 38, 41 to 44	Output	Output pins for the motor control PWM timer channels 1A to 1H.
	PWM2A to PWM2H	45 to 48, 51 to 54	Output	Output pins for the motor control PWM timer channels 2A to 2H.
16-bit PWM	PWM0_0 to PWM3_0	25 to 28	Output	Output pins for the 16-bit PWM timer channels 0_0 to 3_0.
	PWM0_1 to PWM3_1	31 to 34	Output	Output pins for the 16-bit PWM timer channels 0_1 to 3_1.
	PWM0_2 to PWM3_2	63 to 66	Output	Output pins for the 16-bit PWM timer channels 0_2 to 3_2
PWM power supply	PWMVcc	29, 39, 49	Input	Power supply pins for the PWM timer outputs.
	PWMVss	30, 40, 50	Input	Power supply pins for the PWM timer outputs. Connect to the system power supply (0 V).
Serial communication interface (SCI)	TxD0	85	Output	Transmit data output pins
	TxD2	55		
	TxD4	74		
	TxD5	32		
	RxD0	84	Input	Receive data input pins
	RxD2	56		
	RxD4	75		
	RxD5	33		
	SCK0	83	I/O	Clock signals input/output pins
	SCK2	57		
	SCK4	76		
	SCK5	34		
I²C bus interface 2 (IIC2)	SCL0	62	I/O	IIC2 clock signals input/output pins
	SCL1	60		
	SDA0	61	I/O	IIC2 data input/output pins
	SDA1	59		
Controller area network (RCAN-ET)	CTx_0	68	Output	RCAN-ET bus transmission
	CTx_1	70		
	CRx_0	67	Input	RCAN-ET bus reception
	CRx_1	69		

RENESAS

Classification	Abbreviation	Pin Number	I/O	Description
Synchronous serial communication unit (SSU)	SSO_1	83	I/O	Data input/output pins.
	SSO_0	141		
	SSI_1	84	I/O	Data input/output pins.
	SSI_0	142		
	SSCK_1	85	I/O	Clock input/output pins.
	SSCK_0	143		
	$\overline{\text{SCS\_1}}$	86	I/O	Chip select input/output pins.
	$\overline{\text{SCS\_0}}$	144		
Sound generator (SDG)	SGOUT3 to SGOUT0	100 to 103	Output	Sound generator output pin
A/D converter	AN15	112	Input	Inputs analog signals for the A/D converter.
	AN14	111		
	AN13	110		
	AN12	109		
	AN11	116		
	AN10	115		
	AN9	114		
	AN8	113		
	AN7	128		
	AN6	127		
	AN5	126		
	AN4	125		
	AN3	124		
	AN2	123		
	AN1	122		
	AN0	121		
	$\overline{\text{ADTRG1}}$	62	Input	Inputs the external trigger signal to start A/D conversion.
	$\overline{\text{ADTRG0}}$	58		

RENESAS

Classification	Abbreviation	Pin Number	I/O	Description
A/D converter	$AV_{cc}1$	117	Input	Analog power supply pins for the A/D converter and D/A converter. When neither the A/D converter nor the D/A converter is used, connect to the system power supply.
	$AV_{cc}0$	119		
	AV_{ss}	120	Input	Ground pin for the A/D converter and the D/A converter. Connect to the system power supply (0 V).
	Vref	118	Input	Reference voltage input pin for the A/D converter and the D/A converter. When neither the A/D converter nor the D/A converter is used, connect to the system power supply.
D/A converter	DA1	128	Output	Analog signal output pins
	DA0	127		
I/O port	P17	62	I/O	8-bit input/output pins
	P16	61		
	P15	60		
	P14	59		
	P13	58		
	P12	57		
	P11	56		
	P10	55		
	P27	90	I/O	8-bit input/output pins
	P26	89		
	P25	88		
	P24	87		
	P23	86		
	P22	85		
	P21	84		
	P20	83		

RENESAS

Classification	Abbreviation	Pin Number	I/O	Description
I/O port	P37	100	I/O	8-bit input/output pins
	P36	101		
	P35	102		
	P34	103		
	P33	104		
	P32	105		
	P31	106		
	P30	107		
	P47	116	Input	8-bit input pins
	P46	115		
	P45	114		
	P44	113		
	P43	112		
	P42	111		
	P41	110		
	P40	109		
	P57	128	Input	8-bit input pins
	P56	127		
	P55	126		
	P54	125		
	P53	124		
	P52	123		
	P51	122		
	P50	121		
	P67	70	I/O	8-bit input/output pins
	P66	69		
	P65	68		
	P64	67		
	P63	66		
	P62	76		
	P61	75		
	P60	74		

Classification	Abbreviation	Pin Number	I/O	Description
I/O port	PA7	82	Input	1-bit input pin
	PA6	81	I/O	7-bit input/output pins
	PA5	80		
	PA4	79		
	PA3	78		
	PA2	65		
	PA1	64		
	PA0	63		
	PD7	14	I/O	8-bit input/output pins
	PD6	13		
	PD5	12		
	PD4	11		
	PD3	10		
	PD2	9		
	PD1	8		
	PD0	7		
	PE7	24	I/O	8-bit input/output pins
	PE6	23		
	PE5	22		
	PE4	21		
	PE3	20		
	PE2	19		
	PE1	18		
	PE0	17		
	PF7	34	I/O	8-bit input/output pins
	PF6	33		
	PF5	32		
	PF4	31		
	PF3	28		
	PF2	27		
	PF1	26		
	PF0	25		

RENESAS

Classification	Abbreviation	Pin Number	I/O	Description
I/O port	PH7	138	I/O	8-bit input/output pins
	PH6	137		
	PH5	136		
	PH4	135		
	PH3	134		
	PH2	133		
	PH1	132		
	PH0	131		
	PI7	4	I/O	8-bit input/output pins
	PI6	3		
	PI5	2		
	PI4	1		
	PI3	144		
	PI2	143		
	PI1	142		
	PI0	141		
	PJ7	44	I/O	8-bit input/output pins
	PJ6	43		
	PJ5	42		
	PJ4	41		
	PJ3	38		
	PJ2	37		
	PJ1	36		
	PJ0	35		
	PK7	54	I/O	8-bit input/output pins
	PK6	53		
	PK5	52		
	PK4	51		
	PK3	48		
	PK2	47		
	PK1	46		
	PK0	45		

RENESAS

Section 2   CPU

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

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

2.1   Features

- Upward-compatible with H8/300, H8/300H, and H8S CPUs
  - Can execute H8/300, H8/300H, and H8S/2000 object programs
- Sixteen 16-bit general registers
  - Also usable as sixteen 8-bit registers or eight 32-bit registers
- 87 basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Bit field transfer instructions
  - Powerful bit-manipulation instructions
  - Bit condition branch instructions
  - Multiply-and-accumulate instruction
- Eleven addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:2,ERn), @(d:16,ERn), or @(d:32,ERn)]
  - Index register indirect with displacement [@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)]
  - Register indirect with pre-/post-increment or pre-/post-decrement [@+ERn, @ERn+, @−ERn, or @ERn−]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:3, #xx:4, #xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Program-counter relative with index register [@(RnL.B,PC), @(Rn.W,PC), or @(ERn.L,PC)]
  - Memory indirect [@@aa:8]
  - Extended memory indirect [@@vec:7]

RENESAS

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

Notes: 1. Advanced mode is only supported as the CPU operating mode of the H8SX/1544 Group. Normal, middle, and maximum modes are not supported.
   2. The multiplier and divider are supported by the H8SX/1544 Group.
   3. In the H8SX/1544 Group, an instruction is fetched in 32-bit mode.

RENESAS

2.2   CPU Operating Modes

The H8SX CPU has four operating modes: normal, middle, advanced, and maximum modes. For details on mode settings, see section 3.1, Operating Mode Selection.
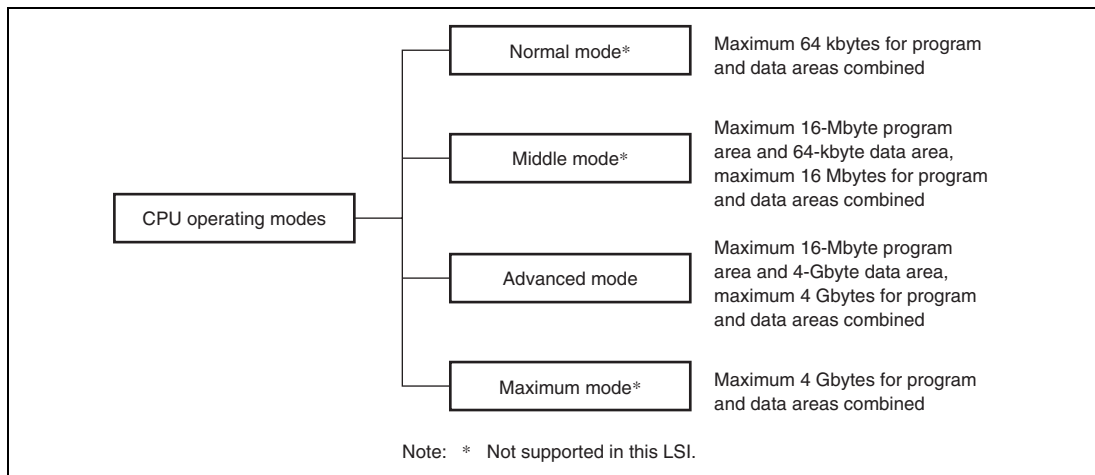
Figure 2.1   CPU Operating Modes

2.2.1   Normal Mode

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

Note: Normal mode is not supported in this LSI.

- Address Space

  The maximum address space of 64 kbytes can be accessed.
- Extended Registers (En)

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

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

- Exception Vector Table and Memory Indirect Branch Addresses

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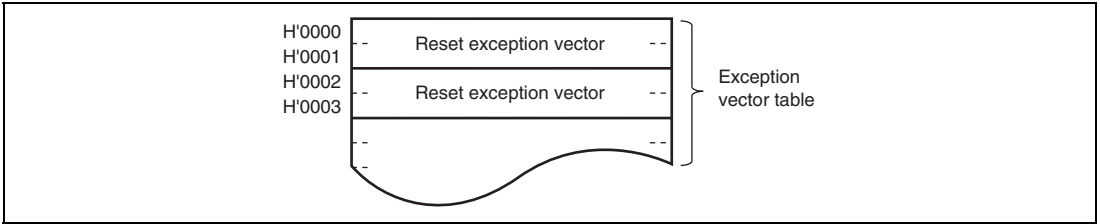
Figure 2.2   Exception Vector Table (Normal Mode)

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

- Stack Structure

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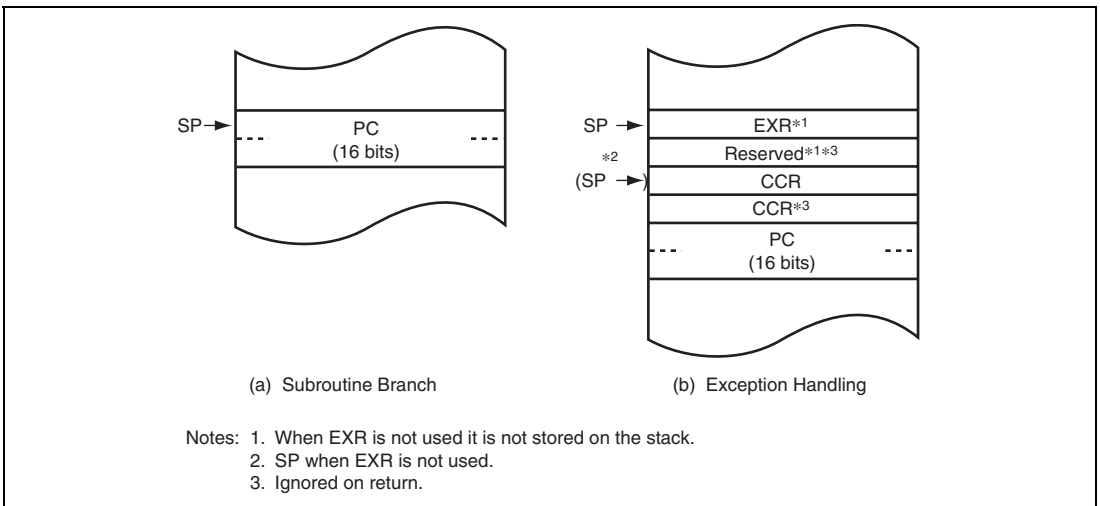
Figure 2.3   Stack Structure (Normal Mode)

2.2.2   Middle Mode

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

Note: Middle mode is not supported in this LSI.

- Address Space

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

- Extended Registers (En)

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

- Instruction Set

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

- Exception Vector Table and Memory Indirect Branch Addresses

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

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

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

- Stack Structure

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

2.2.3   Advanced Mode

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

- Address Space

  The maximum address space of 4 Gbytes can be linearly accessed. For individual areas, up to 16 Mbytes of the program area and up to 4 Gbytes of the data area can be allocated.
- Extended Registers (En)

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

  All instructions and addressing modes can be used.
- Exception Vector Table and Memory Indirect Branch Addresses

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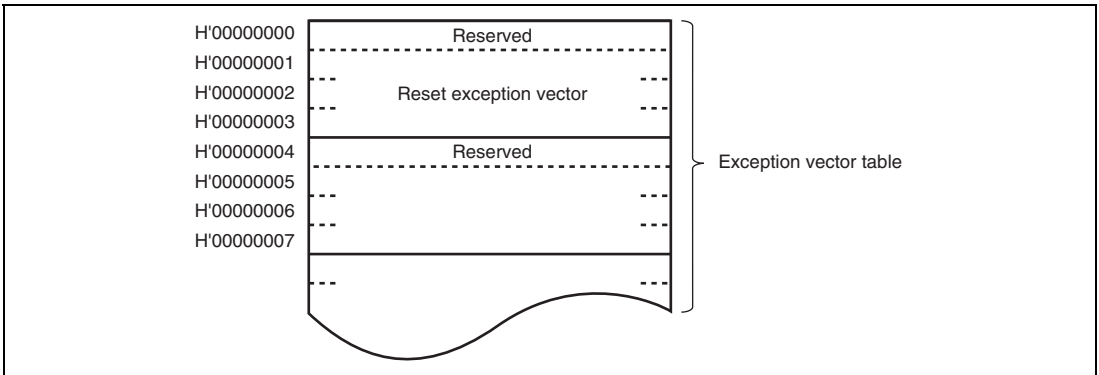
Figure 2.4   Exception Vector Table (Middle and Advanced Modes)

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

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

RENESAS

- Stack Structure

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

Figure 2.5   Stack Structure (Middle and Advanced Modes)

2.2.4   Maximum Mode

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

Note: Maximum mode is not supported in this LSI.

- Address Space

  The maximum address space of 4 Gbytes can be linearly accessed.
- Extended Registers (En)

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

  All instructions and addressing modes can be used.
- Exception Vector Table and Memory Indirect Branch Addresses

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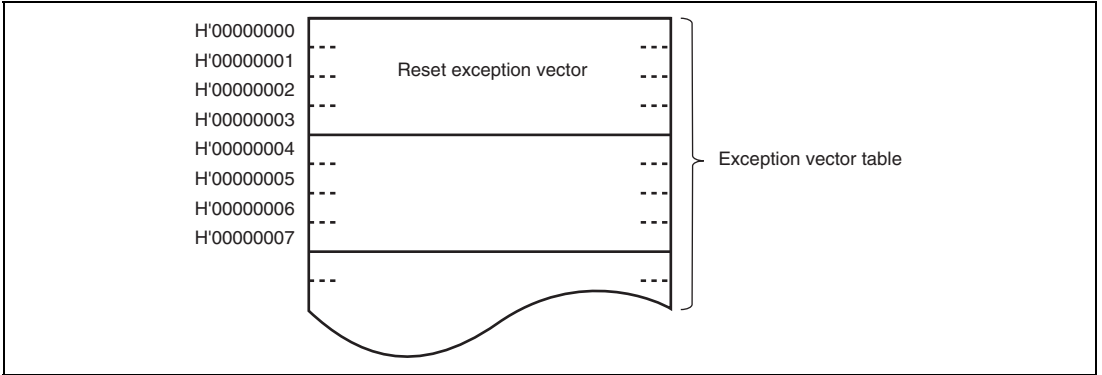
Figure 2.6   Exception Vector Table (Maximum Modes)

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

In maximum mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address.

- Stack Structure

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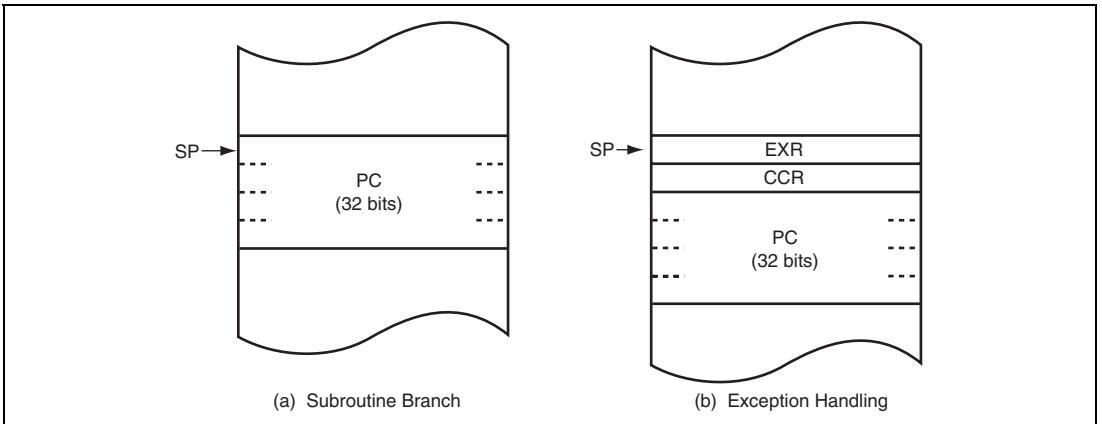
Figure 2.7   Stack Structure (Maximum Mode)

2.3   Instruction Fetch

The H8SX CPU has two modes for instruction fetch: 16-bit and 32-bit modes. It is recommended that the mode be set according to the bus width of the memory in which a program is stored. The instruction-fetch mode setting does not affect operation other than instruction fetch such as data accesses.

Note: In the H8SX/1544 Group, an instruction is fetched in 32-bit mode.

2.4   Address Space

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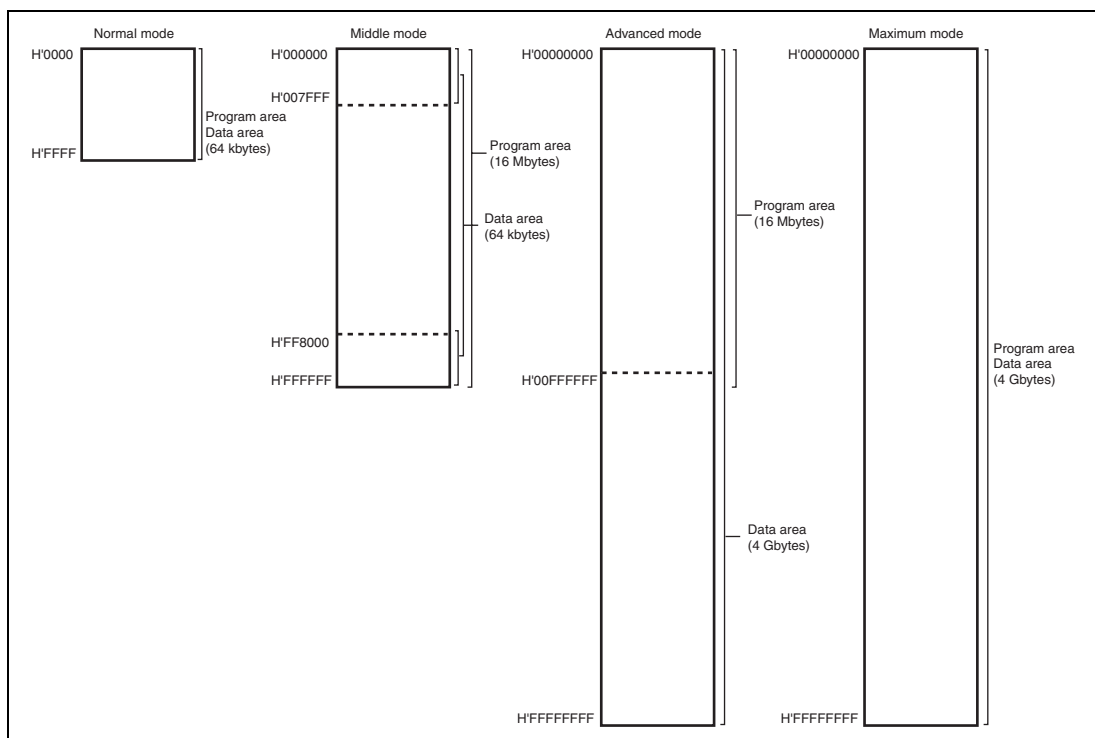
Figure 2.8   Memory Map

2.5 Registers

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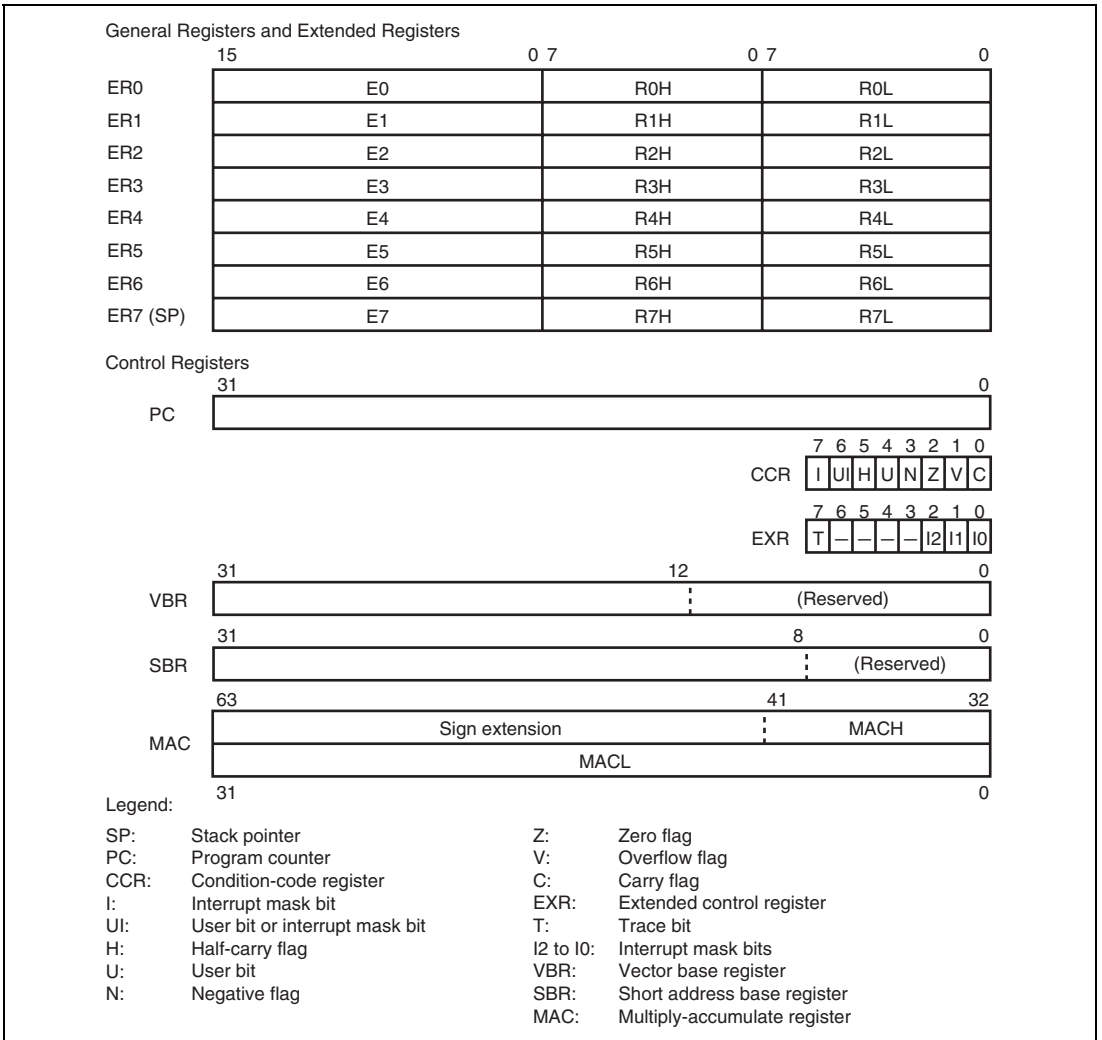
Figure 2.9 CPU Registers

2.5.1   General Registers

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

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

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

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

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

The usage of each register can be selected independently.
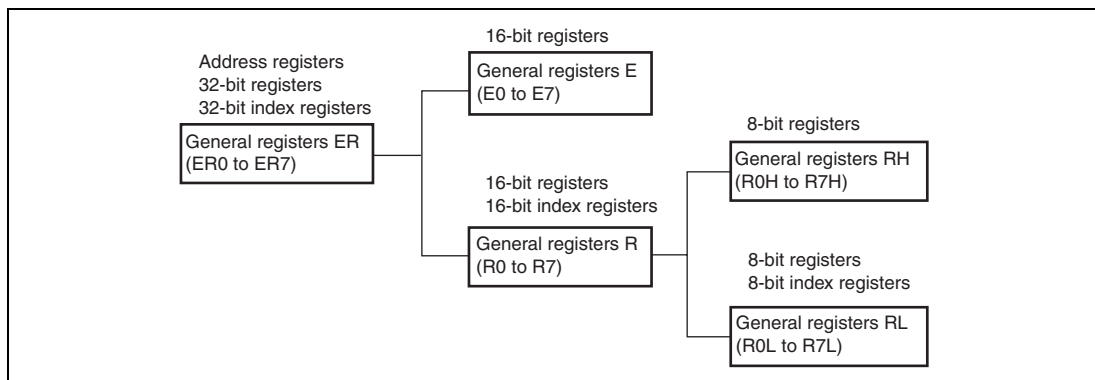
Figure 2.10   Usage of General Registers

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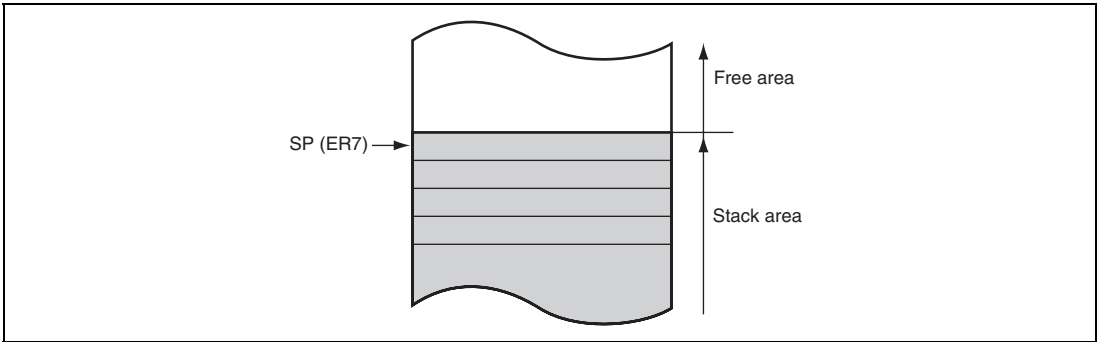
Figure 2.11   Stack

2.5.2   Program Counter (PC)

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

2.5.3   Condition-Code Register (CCR)

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

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

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	Interrupt Mask Bit
				Masks interrupts when set to 1. This bit is set to 1 at the start of an exception handling.
6	UI	Undefined	R/W	User Bit or Interrupt Mask Bit
				Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit can also be used as an interrupt mask bit.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
5	H	Undefined	R/W	Half-Carry Flag
				When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.
4	U	Undefined	R/W	User Bit
				Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag
				Stores the value of the most significant bit (regarded as sign bit) of data.
2	Z	Undefined	R/W	Zero Flag
				Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag
				Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.
0	C	Undefined	R/W	Carry Flag
				Set to 1 when a carry occurs, and cleared to 0 otherwise. A carry has the following types:
				• Carry from the result of addition
				• Borrow from the result of subtraction
				• Carry from the result of shift or rotation
				The carry flag is also used as a bit accumulator by bit manipulation instructions.

RENESAS

2.5.4   Extended Control Register (EXR)

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

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

For details, see section 4, Exception Handling.

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

2.5.5   Vector Base Register (VBR)

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

2.5.6   Short Address Base Register (SBR)

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

RENESAS

2.5.7   Multiply-Accumulate Register (MAC)

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

2.5.8   Initial Values of CPU Registers

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

2.6   2Data Formats

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

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

2.6.1   General Register Data Formats

Figure 2.12 shows the data formats in general registers.

RENESAS
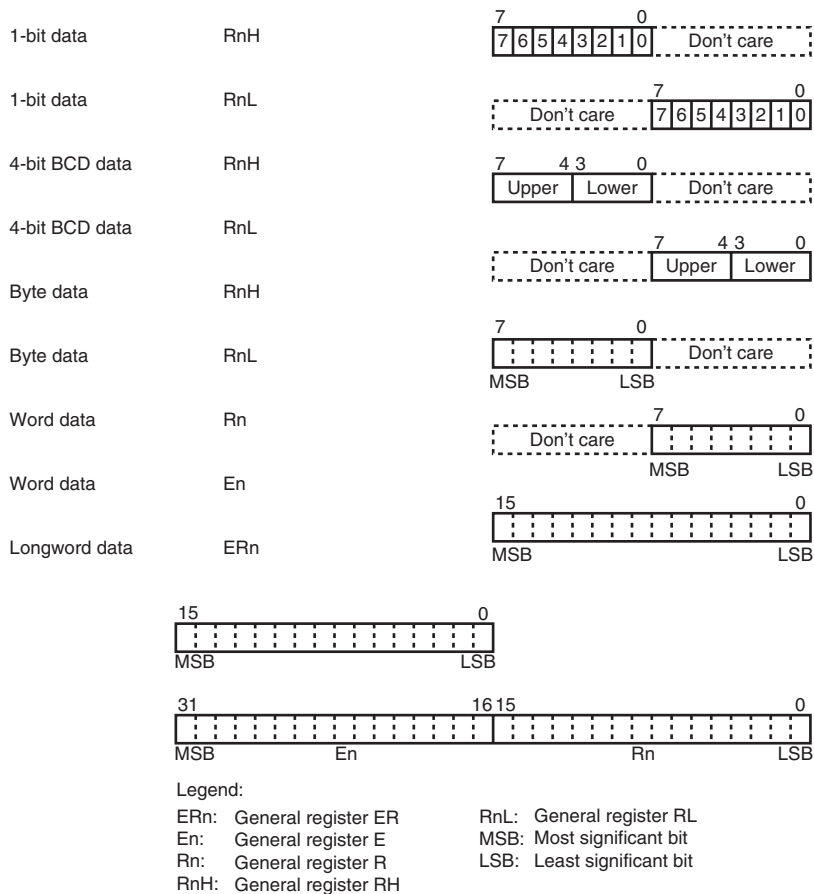
Figure 2.12   General Register Data Formats

2.6.2   Memory Data Formats

Figure 2.13 shows the data formats in memory.

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

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

When SP (ER7) is used as an address register to access the stack, the operand size should be word size or longword size.
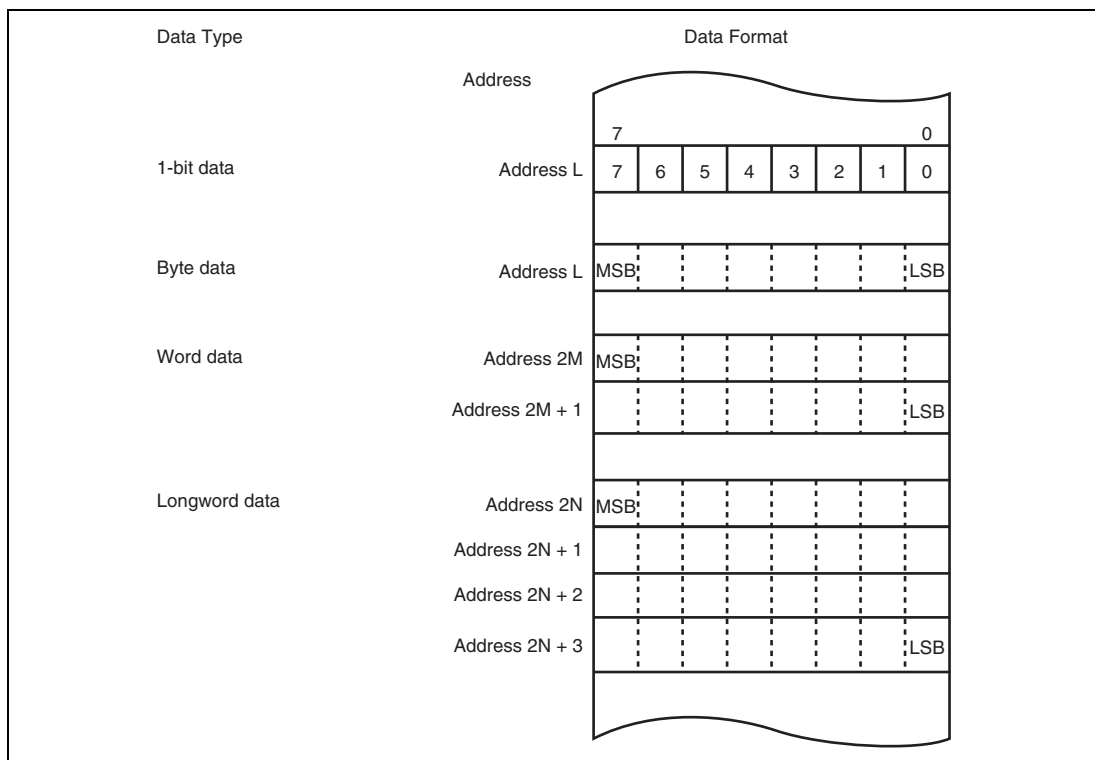
Figure 2.13   Memory Data Formats

2.7   Instruction Set

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

Table 2.1   Instruction Classification

Function	Instructions	Size	Types
Data transfer	MOV	B/W/L	6
	MOVFPE*[6], MOVTPE*[6]	B	
	POP, PUSH*[1]	W/L	
	LDM, STM	L	
	MOVA	B/W*[2]	
Block transfer	EEPMOV	B	3
	MOVMD	B/W/L	
	MOVSD	B	
Arithmetic operations	ADD, ADDX, SUB, SUBX, CMP, NEG, INC, DEC	B/W/L	27
	DAA, DAS	B	
	ADDS, SUBS	L	
	MULXU, DIVXU, MULXS, DIVXS	B/W	
	MULU, DIVU, MULS, DIVS	W/L	
	MULU/U, MULS/U	L	
	EXTU, EXTS	W/L	
	TAS	B	
	MAC	—	
	LDMAC, STMAC	—	
	CLRMAC	—	
Logic operations	AND, OR, XOR, NOT	B/W/L	4
Shift	SHLL, SHLR, SHAL, SHAR, ROTL, ROTR, ROTXL, ROTXR	B/W/L	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	B	20
	BSET/EQ, BSET/NE, BCLR/EQ, BCLR/NE, BSTZ, BISTZ	B	
	BFLD, BFST	B	

RENESAS

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

Legend:

B: Byte size

W: Word size

L: Longword size

Notes: 1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @−SP.

POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @−SP.

2. Size of data to be added with a displacement

3. Size of data to specify a branch condition

4. Bcc is the generic designation of a conditional branch instruction.

5. Size of general register to be restored

6. Not available in this LSI.

2.7.1   Instructions and Addressing Modes

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

Table 2.2   Combinations of Instructions and Addressing Modes (1)

Classifi-cation	Instruction	Size	#xx	Rn	@ERn	@(d,ERn)	@(d, RnL.B/ Rn.W/ ERn.L)	@−ERn/ @ERn+/ @ERn−/ @+ERn	@aa:8	@aa:16/ @aa:32	—
Data transfer	MOV	B/W/L	S	SD	SD	SD	SD	SD		SD	
		B		S/D					S/D		
	MOVFPE, MOVTPE*[12]	B		S/D						S/D*[1]	
	POP, PUSH	W/L		S/D				S/D*[2]			
	LDM, STM	L		S/D				S/D*[2]			
	MOVA*[4]	B/W		S	S	S	S	S		S	
Block transfer	EEPMOV	B									SD*[3]
	MOVMD	B/W/L									SD*[3]
	MOVSD	B									SD*[3]
Arithmetic operations	ADD, CMP	B	S	D	D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	SUB	B	S		D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	ADDX, SUBX	B/W/L	S	SD							
		B/W/L	S		SD						
		B/W/L	S					SD*[5]			
	INC, DEC	B/W/L		D							
	ADDS, SUBS	L		D							
	DAA, DAS	B		D							

RENESAS

			Addressing Mode								
Classification	Instruction	Size	#xx	Rn	@ERn	@(d,ERn)	@(d, RnL.B/ Rn.W/ ERn.L)	@−ERn/ @ERn+/ @ERn−/ @+ERn	@aa:8	@aa:16/ @aa:32	—
Arithmetic operations	MULXU, DIVXU	B/W	S:4	SD							
	MULU, DIVU	W/L	S:4	SD							
	MULXS, DIVXS	B/W	S:4	SD							
	MULS, DIVS	W/L	S:4	SD							
	NEG	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
	EXTU, EXTS	W/L		D	D	D	D	D		D	
	TAS	B			D						
	MAC	—									
	CLRMAC	—									O
	LDMAC	—		S							
	STMAC	—		D							
Logic operations	AND, OR, XOR	B	S	D	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	NOT	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
Shift	SHLL, SHLR	B		D	D	D	D	D	D	D	
		B/W/L*6		D	D	D	D	D		D	
		B/W/L*7		D							
	SHAL, SHAR ROTL, ROTR ROTXL, ROTXR	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
Bit manipulation	BSET, BCLR, BNOT, BTST, BSET/cc, BCLR/cc	B		D	D					D	D	
	BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST, BSTZ, BISTZ	B		D	D					D	D	

RENESAS

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

Legend:

d: d:16 or d:32

S: Can be specified as a source operand.

D: Can be specified as a destination operand.

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

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

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

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

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

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

4. Size of data to be added with a displacement

5. Only @ERn− is available

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

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

8. Size of data to specify a branch condition

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

10. Only @ERn+ is available

11. Only @−ERn is available

12. Not available in this LSI.

RENESAS

Table 2.2   Combinations of Instructions and Addressing Modes (2)

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

Legend:

d: d:8 or d:16

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

2.7.2   Table of Instructions Classified by Function

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

Table 2.3   Operation Notation

Operation Notation	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extended control register
CCR	Condition-code register
VBR	Vector base register
SBR	Short address base register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#IMM	Immediate data
disp	Displacement
+	Addition
−	Subtraction
×	Multiplication
÷	Division
∧	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Move
~	Logical not (logical complement)
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

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

RENESAS

Table 2.4   Data Transfer Instructions

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

Note: ∗   Not available in this LSI.

Table 2.5   Block Transfer Instructions

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

RENESAS

Table 2.6   Arithmetic Operation Instructions

Instruction	Size	Function
ADD SUB	B/W/L	(EAd) ± #IMM → (EAd), (EAd) ± (EAs) → (EAd) Performs addition or subtraction on data between immediate data, general registers, and memory. Immediate byte data cannot be subtracted from byte data in a general register.
ADDX SUBX	B/W/L	(EAd) ± #IMM ± C → (EAd), (EAd) ± (EAs) ± C → (EAd) Performs addition or subtraction with carry on data between immediate data, general registers, and memory. The addressing mode which specifies a memory location can be specified as register indirect with post-decrement or register indirect.
INC DEC	B/W/L	Rd ± 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 general 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 2-digit 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.
MULU	W/L	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.
MULU/U	L	Rd × Rs → Rd Performs unsigned multiplication on data in two general registers (32 bits × 32 bits → upper 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.
MULS	W/L	Rd × Rs → Rd Performs signed multiplication on data in two general registers: either 16 bits × 16 bits → 16 bits, or 32 bits × 32 bits → 32 bits.
MULS/U	L	Rd × Rs → Rd Performs signed multiplication on data in two general registers (32 bits × 32 bits → upper 32 bits).

RENESAS

Instruction	Size	Function
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.
DIVU	W/L	Rd ÷ Rs → Rd
		Performs unsigned division on data in two general registers: either 16 bits ÷ 16 bits → 16-bit quotient, or 32 bits ÷ 32 bits → 32-bit quotient.
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.
DIVS	W/L	Rd ÷ Rs → Rd
		Performs signed division on data in two general registers: either 16 bits ÷ 16 bits → 16-bit quotient, or 32 bits ÷ 32 bits → 32-bit quotient.
CMP	B/W/L	(EAd) − #IMM, (EAd) − (EAs)
		Compares data between immediate data, general registers, and memory and stores the result in CCR.
NEG	B/W/L	0 − (EAd) → (EAd)
		Takes the two's complement (arithmetic complement) of data in a general register or the contents of a memory location.
EXTU	W/L	(EAd) (zero extension) → (EAd)
		Performs zero-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size.
		The lower 8 bits to word or longword, or the lower 16 bits to longword can be zero-extended.
EXTS	W/L	(EAd) (sign extension) → (EAd)
		Performs sign-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size.
		The lower 8 bits to word or longword, or the lower 16 bits to longword can be sign-extended.
TAS	B	@ERd − 0, 1 → (<bit 7> of @EAd)
		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 MAC.
CLRMAC	—	0 → MAC
		Clears MAC to zero.

RENESAS

Instruction	Size	Function
LDMAC	—	Rs → MAC
		Loads data from a general register to MAC.
STMAC	—	MAC → Rd
		Stores data from MAC to a general register.

Table 2.7   Logic Operation Instructions

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

RENESAS

Table 2.8   Shift Operation Instructions

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

RENESAS

Table 2.9   Bit Manipulation Instructions

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

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

RENESAS

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

Table 2.10   Branch Instructions

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

RENESAS

Table 2.11   System Control Instructions

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

RENESAS

2.7.3   Basic Instruction Formats

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

Figure 2.14 shows examples of instruction formats.
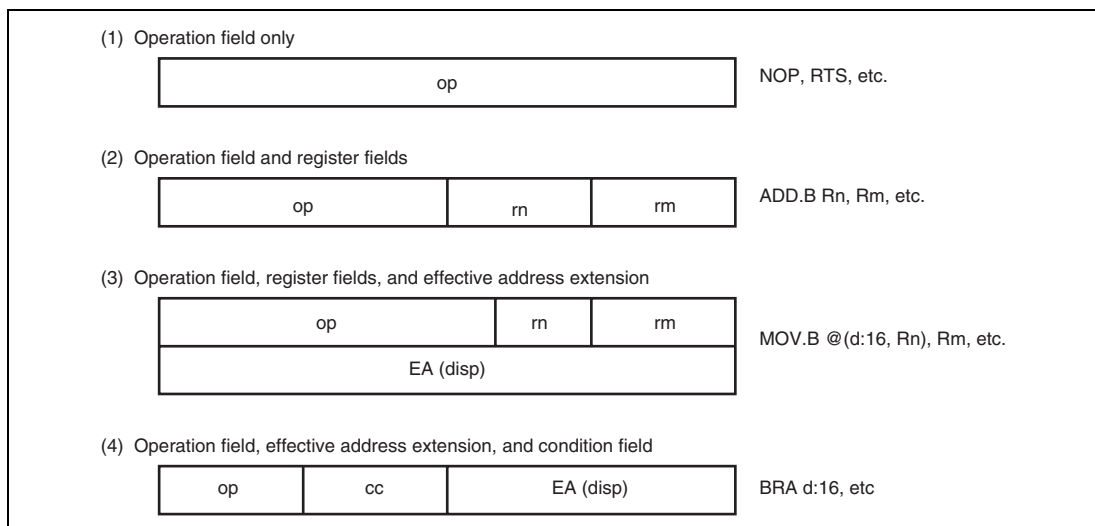
Figure 2.14   Instruction Formats

- Operation Field

  Indicates the function of the instruction, and specifies the addressing mode and operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.

- Register Field

  Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.

- Effective Address Extension

  8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.

- Condition Field

  Specifies the branch condition of Bcc instructions.

RENESAS

2.8   Addressing Modes and Effective Address Calculation

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

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

Table 2.12   Addressing Modes

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

RENESAS

2.8.1   Register Direct—Rn

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

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

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

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

2.8.2   Register Indirect—@ERn

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

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

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

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

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

RENESAS

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

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

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

(1) Register indirect with post-increment—@ERn+

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

(2) Register indirect with pre-decrement—@−ERn

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

(3) Register indirect with pre-increment—@+ERn

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

RENESAS

(4)   Register indirect with post-decrement—@ERn–

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

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

Example 1:

MOV.W   R0, @ER0+

When the value of ER0 before execution is H'12345678, H'567A is written to the address H'12345678.

Example 2:

MOV.B   @ER0+, @ER0+

When the value of ER0 before execution is H'00001000, the address H'00001000 is read and the contents are written to the address H'00001001.

The value of ER0 after execution is H'00001002.

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

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

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

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

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

Table 2.13 shows the accessible absolute address ranges.

Table 2.13   Absolute Address Access Ranges

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

RENESAS

2.8.7   Immediate—#xx

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

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

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

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

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

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

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

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

RENESAS

2.8.10   Memory Indirect—@@aa:8

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

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

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

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

Figure 2.15 shows an example of specification of a branch address using this addressing mode.
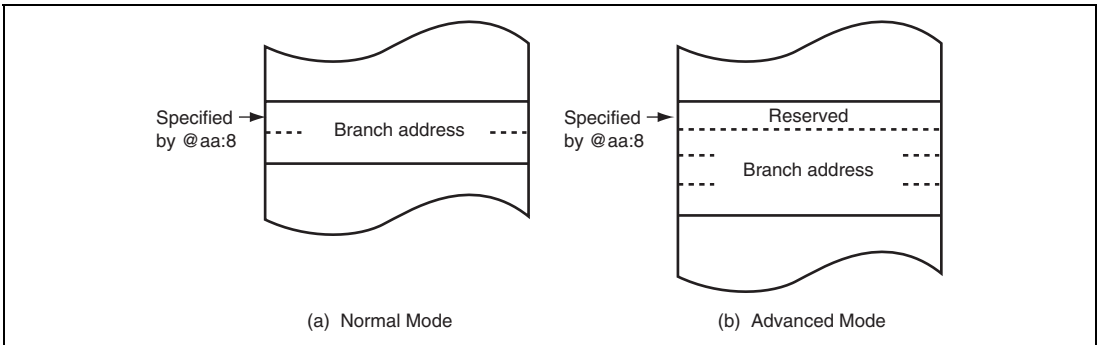
Figure 2.15   Branch Address Specification in Memory Indirect Mode

2.8.11   Extended Memory Indirect—@@vec:7

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

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

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

2.8.12   Effective Address Calculation

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

The valid bits in middle mode are as follows:

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

RENESAS

Table 2.14 Effective Address Calculation for Transfer and Operation Instructions
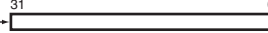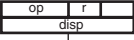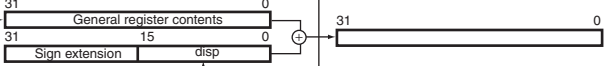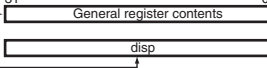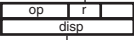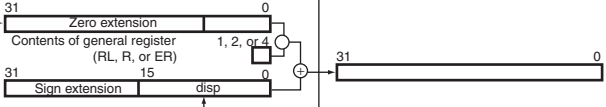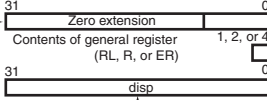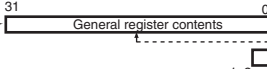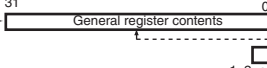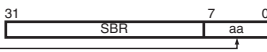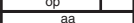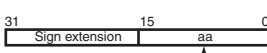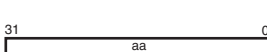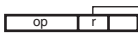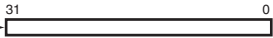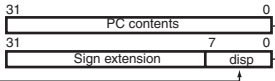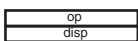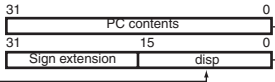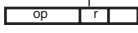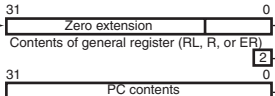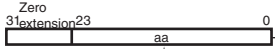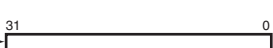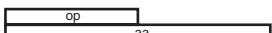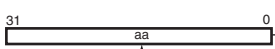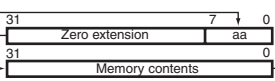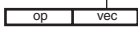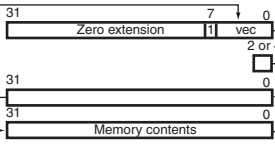
No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Immediate `op    IMM`		
2	Register direct `op  rm  rn`		
3	Register indirect `op  r`	31 0 General register contents	31 0

4	Register indirect with 16-bit displacement
	`op  r`
	`disp`
		31 0
		General register contents
		31 15 0
		Sign extension disp
			31 0

	Register indirect with 32-bit displacement
	`op  r`
	`disp`
		31 0
		General register contents
		disp
			31 0

5	Index register indirect with 16-bit displacement
	`op  r`
	`disp`
		31 0
		Zero extension
		Contents of general register 1, 2, or 4
		(RL, R, or ER)
		31 15 0
		Sign extension disp
			31 0

	Index register indirect with 32-bit displacement
	`op  r`
	`disp`
		31 0
		Zero extension
		Contents of general register 1, 2, or 4
		(RL, R, or ER)
		31 0
		disp
			31 0

6	Register indirect with post-increment or post-decrement `op  r`	31 0 General register contents 1, 2, or 4	31 0
	Register indirect with pre-increment or pre-decrement `op  r`	31 0 General register contents 1, 2, or 4	31 0
7	8-bit absolute address `op  aa`	31 7 0 SBR aa	31 0

	16-bit absolute address
	`op`
	`aa`
		31 15 0
		Sign extension aa
			31 0

	32-bit absolute address
	`op`
	`aa`
		31 0
		aa
			31 0

RENESAS

Table 2.15   Effective Address Calculation for Branch Instructions

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)

2.8.13   MOVA Instruction

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

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

2.9   Processing States

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

- Reset state

  In this state the CPU and internal peripheral modules are all initialized and stopped. When the $\overline{\text{RES}}$ input goes low, all current processing stops and the CPU enters the reset state. All interrupts are masked in the reset state. Reset exception handling starts when the $\overline{\text{RES}}$ signal changes from low to high. The reset state can also be entered by a watchdog timer overflow. For details, refer to section 4, Exception Handling.

- Exception-handling state

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

- Program execution state

  In this state the CPU executes program instructions in sequence.

- Bus-released state

  In this state, the bus has been released in response to a bus request from the DMA controller (DMAC). While the bus is released, the CPU halts operations.

- Program stop state

  This is a power-down state in which the CPU stops operating. The program stop state occurs when a SLEEP instruction is executed or the CPU enters software standby mode. For details, refer to section 24, Power-Down Modes.
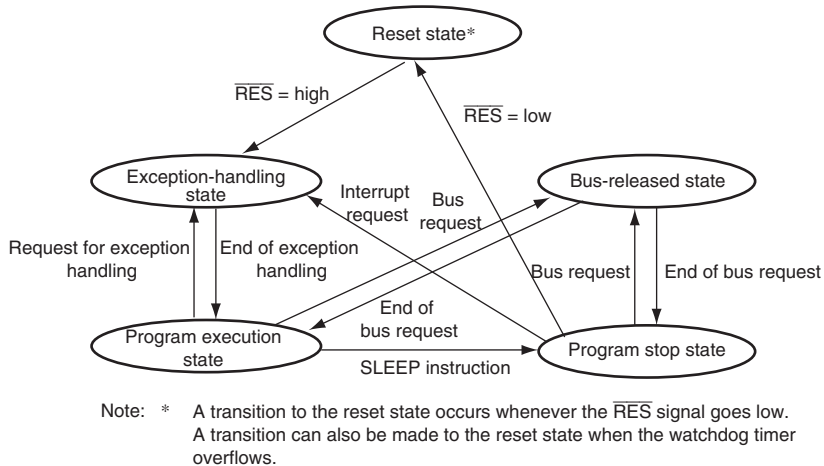
Figure 2.16   State Transitions

Section 3   MCU Operating Modes

3.1   Operating Mode Selection

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

Table 3.1   MCU Operating Mode Settings

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

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

Mode 2 is the boot mode in which the flash memory can be programmed and erased. For details on the boot mode, see section 22, Flash Memory.

Mode 7 is a single-chip mode. In the initial state, all areas are designated to 8-bit access space and all I/O ports can be used as general input/output ports. The external address space cannot be accessed in the initial state, but setting the EXPE bit in the system control register (SYSCR) to 1 enables to use the external address space. After the external address space is enabled, ports D, E, and F can be used as an address output bus and ports H and I as a data bus by specifying the data direction register (DDR) for each port.

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

RENESAS

3.2   Register Descriptions

The following registers are related to the operating mode setting.

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

3.2.1   Mode Control Register (MDCR)

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

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

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

Note: *   Determined by pins MD2 to MD0.

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

RENESAS

Bit	Bit Name	Initial Value	R/W	Descriptions
7	—	0	R	Reserved
6	—	1	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	—	1	R	
3	—	Undefined*	R	
2	—	Undefined*	R	
1	—	Undefined*	R	
0	—	Undefined*	R	

Note: *   Determined by pins MD2 to MD0.

Table 3.2   Settings of Bits MDS3 to MDS0

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

RENESAS

3.2.2   System Control Register (SYSCR)

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

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

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

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

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

RENESAS

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

Notes: 1. For details on instruction fetch mode, see section 2.3, Instruction Fetch.

2. The initial value depends on the LSI initiation mode.

RENESAS

3.3   Operating Mode Descriptions

3.3.1   Mode 2

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

3.3.2   Mode 4

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

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

3.3.3   Mode 5

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

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

3.3.4   Mode 6

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

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

3.3.5   Mode 7

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

In the initial state, all areas are designated to 8-bit access space and all I/O ports can be used as general input/output ports. The external address space cannot be accessed in the initial state, but setting the EXPE bit in the system control register (SYSCR) to 1 enables the external address space. After the external address space is enabled, ports D, E, and F can be used as an address output bus and ports H and I as a data bus by specifying the data direction register (DDR) for each port. For details, see section 8, I/O Ports.

3.3.6   Pin Functions

Table 3.3 lists the pin functions in each operating mode.

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

MCU Operating Mode		Port A					Port F			
		PA7	PA6 to PA3	PA2 to PA0	Port D	Port E	PF3 to PF0	PF7 to PF4	Port H	Port I
2	Boot mode	P∗/C	P∗/C	P	P∗/A	P∗/A	P∗/A	P∗/A	P∗/D	P∗/D
4	ROM disabled extended	C	C	P	A	A	A	A	D	P/D∗
5	ROM disabled extended	C	C	P	A	A	A	A	D	P∗/D
6	ROM enabled extended	C	C	P	P∗/A	P∗/A	P∗/A	P∗/A	D	P∗/D
7	Single chip mode	P∗/C	P∗/C	P	P∗/A	P∗/A	P∗/A	P∗/A	P∗/D	P∗/D

Legend:

P: I/O port

A: Address bus output

D: Data bus input/output

C: Control signals, clock input/output

∗:   Immediately after a reset

RENESAS

3.4   Address Map

3.4.1   Address Map

Figures 3.1 and 3.2 are the address maps of the H8SX/1544.

Notes: 1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.
       2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0.
       3. Do not access the reserved areas.
       4. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.
       5. Initial external bus width: 16 bits, Maximum external bus width: 16 bits

Figure 3.1   H8SX/1544 Address Map (1)

RENESAS

Notes: 1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.
       2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0.
       3. Do not access the reserved areas.
       4. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.
       5. Initial external bus width: 8 bits, Maximum external bus width: 16 bits

Figure 3.2   H8SX/1544 Address Map (2)

RENESAS

Figures 3.3 and 3.4 are the address maps of the H8SX/1543.

Mode 2
Boot mode
(Advanced mode)

Address	Region
H'000000	On-chip ROM (FLASH) 384 kbytes
H'060000	External address space/ reserved area*1*3
H'FD9000	Reserved area*3
H'FDC000	External address space/ reserved area*1*3
H'FF8000	On-chip RAM*2 16 kbytes
H'FFC000	External address space/ reserved area*1*3
H'FFEA00	On-chip I/O registers
H'FFFF00	External address space/ reserved area*1*3
H'FFFF20	On-chip I/O registers
H'FFFFFF	

Mode 4*5
On-chip ROM disabled
extended mode
(Advanced mode)

Address	Region
H'000000	External address space
H'FD9000	Reserved area*3
H'FDC000	External address space
H'FF8000	On-chip RAM 16 kbytes/ external address space*4
H'FFC000	External address space
H'FFEA00	On-chip I/O registers
H'FFFF00	External address space
H'FFFF20	On-chip I/O registers
H'FFFFFF	

Notes: 1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.
2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0.
3. Do not access the reserved areas.
4. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.
5. Initial external bus width: 16 bits, Maximum external bus width: 16 bits

Figure 3.3   H8SX/1543 Address Map (1)

RENESAS

	Mode 5*5 On-chip ROM disabled extended mode (Advanced mode)		Mode 6 On-chip ROM enabled extended mode (Advanced mode)		Mode 7 Shingle-chip mode (Advanced mode)
H'000000	External address space	H'000000	On-chip ROM (FLASH) 384 kbytes	H'000000	On-chip ROM (FLASH) 384 kbytes
		H'060000	External address space	H'060000	External address space/ reserved area*1*3
H'FD9000	Reserved area*3	H'FD9000	Reserved area*3	H'FD9000	Reserved area*3
H'FDC000	External address space	H'FDC000	External address space	H'FDC000	External address space/ reserved area*1*3
H'FF8000	On-chip RAM 16 kbytes/ external address space*4	H'FF8000	On-chip RAM 16 kbytes/ external address space*4	H'FF8000	On-chip RAM 16 kbytes/ external address space*4
H'FFC000	External address space	H'FFC000	External address space	H'FFC000	External address space/ reserved area*1*3
H'FFEA00	On-chip I/O registers	H'FFEA00	On-chip I/O registers	H'FFEA00	On-chip I/O registers
H'FFFF00	External address space	H'FFFF00	External address space	H'FFFF00	External address space/ reserved area*1*3
H'FFFF20	On-chip I/O registers	H'FFFF20	On-chip I/O registers	H'FFFF20	On-chip I/O registers
H'FFFFFF		H'FFFFFF		H'FFFFFF	

Notes: 1. This area is specified as the external address space when EXPE = 1 and the reserved area when EXPE = 0.
2. The on-chip RAM is used for flash memory programming. Do not clear the RAME bit to 0.
3. Do not access the reserved areas.
4. This area is specified as the external address space by clearing the RAME bit in SYSCR to 0.
5. Initial external bus width: 8 bits, Maximum external bus width: 16 bits

Figure 3.4   H8SX/1543 Address Map (2)

RENESAS

Section 4   Exception Handling

4.1   Exception Handling Types and Priority

As table 4.1 indicates, exception handling is caused by a reset, a trace, an address error, an interrupt, a trap instruction, and an illegal instruction (general illegal instruction or slot illegal instruction). Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Exception sources, the stack structure, and operation of the CPU vary depending on the interrupt control mode. For details on the interrupt control mode, see section 5, Interrupt Controller.

Table 4.1   Exception Types and Priority

Priority	Exception Type	Exception Handling Start Timing
High	Reset	Exception handling starts at the timing of level change from low to high on the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. The CPU enters the reset state when the $\overline{\text{RES}}$ pin is low.
	Illegal instruction	Exception handling starts when an undefined code is executed.
	Trace[1]	Exception handling starts after execution of the current instruction or exception handling, if the trace (T) bit in EXR is set to 1.
	Address error	After an address error has occurred, exception handling starts on completion of instruction execution.
	Interrupt	Exception handling starts after execution of the current instruction or exception handling, if an interrupt request has occurred.[2]
Low	Trap instruction[3]	Exception handling starts by execution of a trap instruction (TRAPA).

Notes: 1. Traces are enabled only in interrupt control mode 2. Trace exception handling is not executed after execution of an RTE instruction.

2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.

3. Trap instruction exception handling requests are accepted at all times in program execution state.

4.2 Exception Sources and Exception Handling Vector Table

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

Table 4.2 shows the correspondence between the exception sources and vector table address offsets. Table 4.3 shows the calculation method of exception handling vector table addresses.

Table 4.2   Exception Handling Vector Table

		Vector Table Address Offset*[1]	
Exception Source	**Vector Number**	**Normal Mode*[2]**	**Advanced, Middle*[2], Maximum*[2] Modes**
Reset	0	H'0000 to H'0001	H'0000 to H'0003
Reserved for system use	1	H'0002 to H'0003	H'0004 to H'0007
	2	H'0004 to H'0005	H'0008 to H'000B
	3	H'0006 to H'0007	H'000C to H'000F
Illegal instruction	4	H'0008 to H'0009	H'0010 to H'0013
Trace	5	H'000A to H'000B	H'0014 to H'0017
Reserved for system use	6	H'000C to H'000D	H'0018 to H'001B
Interrupt (NMI)	7	H'000E to H'000F	H'001C to H'001F
Trap instruction (#0)	8	H'0010 to H'0011	H'0020 to H'0023
(#1)	9	H'0012 to H'0013	H'0024 to H'0027
(#2)	10	H'0014 to H'0015	H'0028 to H'002B
(#3)	11	H'0016 to H'0017	H'002C to H'002F
CPU address error	12	H'0018 to H'0019	H'0030 to H'0033
DMA address error*[3]	13	H'001A to H'001B	H'0034 to H'0037
Reserved for system use	14 | 23	H'001C to H'001D | H'002E to H'002F	H'0038 to H'003B | H'005C to H'005F
User area (not used)	24 | 63	H'0030 to H'0031 | H'007E to H'007F	H'0060 to H'0063 | H'00FC to H'00FF

RENESAS

			Vector Table Address Offset[1]	
Exception Source		Vector Number	Normal Mode[2]	Advanced, Middle[2], Maximum[2] Modes
External interrupt	IRQ0	64	H'0080 to H'0081	H'0100 to H'0103
	IRQ1	65	H'0082 to H'0083	H'0104 to H'0107
	IRQ2	66	H'0084 to H'0085	H'0108 to H'010B
	IRQ3	67	H'0086 to H'0087	H'010C to H'010F
	IRQ4	68	H'0088 to H'0089	H'0110 to H'0113
	IRQ5	69	H'008A to H'008B	H'0114 to H'0117
	IRQ6	70	H'008C to H'008D	H'0118 to H'011B
	IRQ7	71	H'008E to H'008F	H'011C to H'011F
	IRQ8	72	H'0090 to H'0091	H'0120 to H'0123
	IRQ9	73	H'0092 to H'0093	H'0124 to H'0127
	IRQ10	74	H'0094 to H'0095	H'0128 to H'012B
	IRQ11	75	H'0096 to H'0097	H'012C to H'012F
	IRQ12	76	H'0098 to H'0099	H'0130 to H'0133
	IRQ13	77	H'009A to H'009B	H'0134 to H'0137
	IRQ14	78	H'009C to H'009D	H'0138 to H'013B
	IRQ15	79	H'009E to H'009F	H'013C to H'013F
Internal interrupt[4]		80	H'00A0 to H'00A1	H'0140 to H'0143
		|	|	|
		255	H'01FE to H'01FF	H'03FC to H'03FF

Notes: 1. Lower 16 bits of the address.

2. Not available in this LSI.

3. A DMA address error is generated by the DMAC.

4. For details of internal interrupt vectors, see section 5.5, Interrupt Exception Handling Vector Table.

Table 4.3   Calculation Method of Exception Handling Vector Table Address

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

Legend:

VBR: Vector base register

Note: Vector table address offset: See table 4.2.

4.3   Reset

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

The chip can also be reset by overflow of the watchdog timer. For details, see section 10, Watchdog Timer (WDT).

A reset initializes the internal state of the CPU and the registers of the on-chip peripheral modules. The interrupt control mode is 0 immediately after a reset.

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, VBR is cleared to H'00000000, the T bit is cleared to 0 in EXR, and the I bits are set to 1 in EXR and CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Figures 4.1 and 4.2 show examples of the reset sequence.

RENESAS

4.3.2   Interrupts after Reset

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

4.3.3   On-Chip Peripheral Functions after Reset Release

After the reset state is released, MSTPCRA is initialized to H'0FFF, MSTPCRB is initialized to H'FFFF, and MSTPCRC is initialized to H'FF00, and all modules except the DMAC enter module stop mode.

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

(1): Reset exception handling vector address (when reset, (1) = H'000000)
(2): Start address (contents of reset exception handling vector address)
(3) Start address ((3) = (2))
(4) First instruction in the exception handling routine
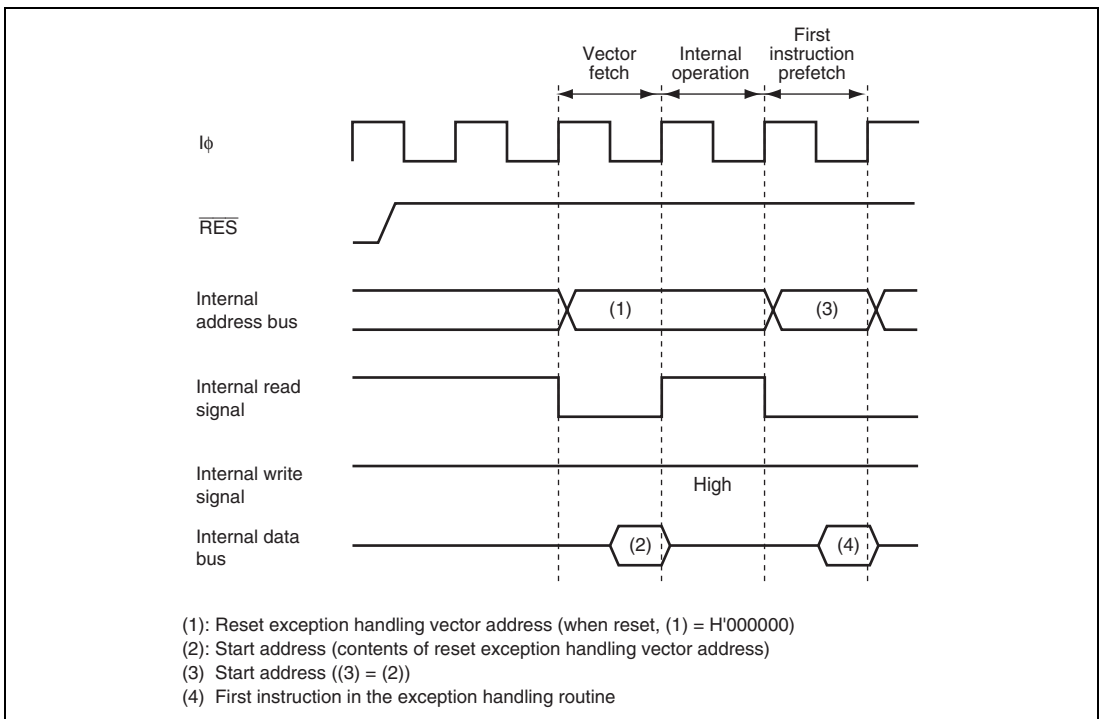
Figure 4.1   Reset Sequence (On-Chip ROM Enabled Advanced Mode)

Figure 4.2   Reset Sequence
(16-Bit External Access in On-Chip ROM Disabled Advanced Mode)

4.4   Traces

Traces are enabled in interrupt control mode 2. Trace mode is not activated in interrupt control mode 0, irrespective of the state of the T bit. Before changing interrupt control modes, the T bit must be cleared. For details on interrupt control modes, see section 5, Interrupt Controller.

If the T bit in EXR is set to 1, trace mode is activated. In trace mode, a trace exception occurs on completion of each instruction. Trace mode is not affected by interrupt masking by CCR. Table 4.4 shows the state of CCR and EXR after execution of trace exception handling. Trace mode is canceled by clearing the T bit in EXR to 0 during the trace exception handling. However, the T bit saved on the stack retains its value of 1, and when control is returned from the trace exception handling routine by the RTE instruction, trace mode resumes. Trace exception handling is not carried out after execution of the RTE instruction.

Interrupts are accepted even within the trace exception handling routine.

Table 4.4   Status of CCR and EXR after Trace Exception Handling

Interrupt Control Mode	CCR		EXR	
	I	UI	I2 to I0	T
0	Trace exception handling cannot be used.			
2	1	—	—	0

Legend:

1:      Set to 1

0:      Cleared to 0

—:   Retains the previous value.

4.5   Address Error

4.5.1   Address Error Source

Instruction fetch, stack operation, or data read/write shown in table 4.5 may cause an address error.

Table 4.5   Bus Cycle and Address Error

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

Notes: 1.   For on-chip peripheral module space, see section 6, Bus Controller (BSC).
2.   For the access-prohibited area, refer to figure 3.1, Address Map (Advanced Mode) in section 3.4, Address Map.

RENESAS

4.5.2   Address Error Exception Handling

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

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

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

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

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

- The ERRF bit of DMDR_0 in the DMAC is set to 1.
- The DTE bits of DMDRs for all channels in the DMAC are cleared to 0 to forcibly terminate transfer.

Table 4.6 shows the state of CCR and EXR after execution of the address error exception handling.

Table 4.6   Status of CCR and EXR after Address Error Exception Handling

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

Legend:
1:    Set to 1
0:    Cleared to 0
—:   Retains the previous value.

4.6   Interrupts

4.6.1   Interrupt Sources

Interrupt sources are NMI, IRQ0 to IRQ11, and on-chip peripheral modules, as shown in table 4.7.

Table 4.7   Interrupt Sources

Type	Source	Number of Sources
NMI	NMI pin (external input)	1
IRQ0 to IRQ15	Pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ15}}$ (external input)	16
On-chip peripheral module	DMA controller (DMAC)	8
	Watchdog timer (WDT)	1
	A/D converter	1
	16-bit timer pulse unit (TPU)	26
	Serial communications interface (SCI)	12
	I2C bus interface 2 (IIC2)	2
	Synchronous serial communication unit (SSU)	2
	Motor control PWM timer	2
	Controller area network (RCAN-ET)	4
	16-bit PWM timer	3
	Sound generator (SDG)	4
	Watch timer (WAT)	2

Different vector numbers and vector table offsets are assigned to different interrupt sources. For vector number and vector table offset, refer to table 5.2, Interrupt Sources, Vector Address Offsets, and Interrupt Priority in section 5, Interrupt Controller.

4.6.2   Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to eight priority/mask levels to enable multiple-interrupt control. The source to start interrupt exception handling and the vector address differ depending on the product. For details, refer to section 5, Interrupt Controller.

RENESAS

The interrupt exception handling is as follows:

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

4.7   Instruction Exception Handling

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

4.7.1   Trap Instruction

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

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

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

Table 4.8 shows the state of CCR and EXR after execution of trap instruction exception handling.

Table 4.8   Status of CCR and EXR after Trap Instruction Exception Handling

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

Legend:

1:     Set to 1

0:     Cleared to 0

—:   Retains the previous value.

RENESAS

4.7.2   Exception Handling by Illegal Instruction

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

The exception handling is as follows:

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

Table 4.9 shows the state of CCR and EXR after execution of illegal instruction exception handling.

Table 4.9   Status of CCR and EXR after Illegal Instruction Exception Handling

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

Legend:

1:      Set to 1

0:      Cleared to 0

—:   Retains the previous value.

RENESAS

4.8 Stack Status after Exception Handling

Figure 4.3 shows the stack after completion of exception handling.
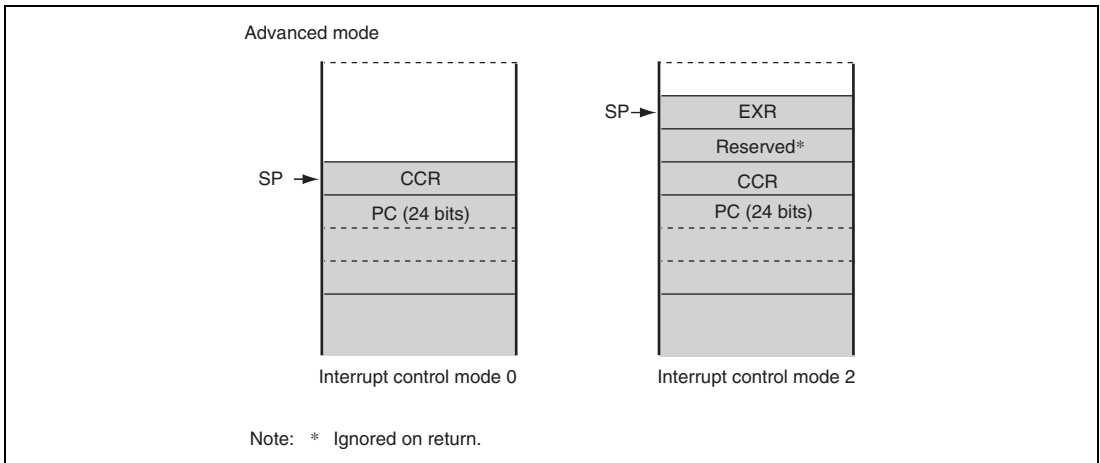
Figure 4.3   Stack Status after Exception Handling

4.9 Usage Note

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

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

Use the following instructions to restore registers:

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

Performing stack manipulation while SP is set to an odd value leads to an address error. Figure 4.4 shows an example of operation when the SP value is odd.
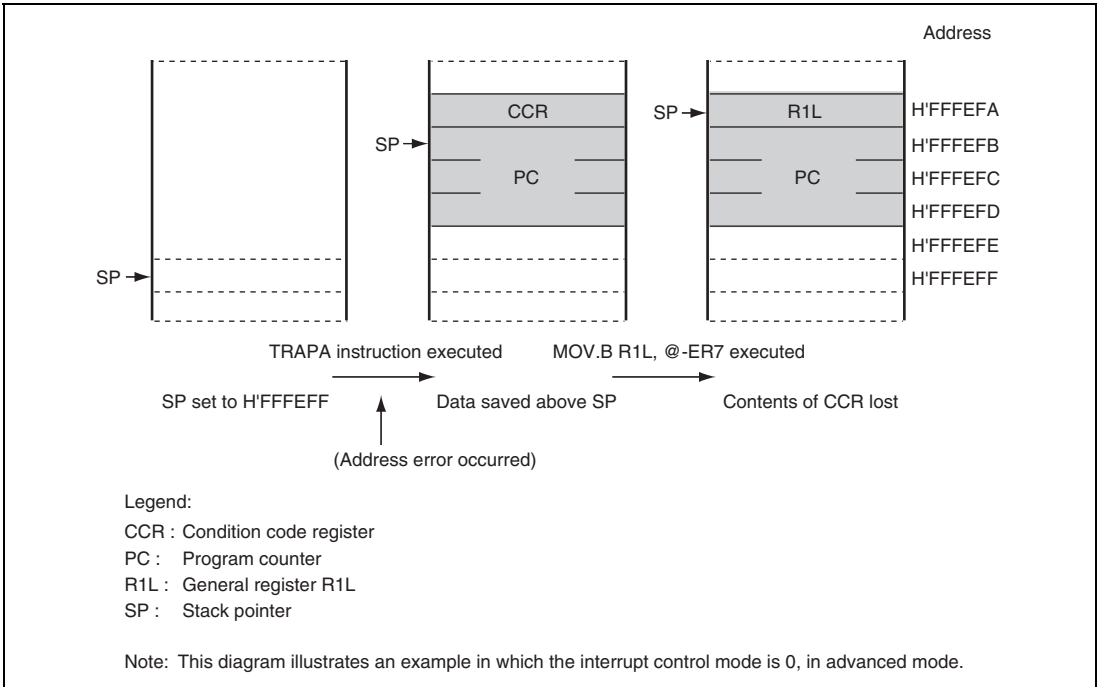
Figure 4.4   Operation when SP Value Is Odd

Section 5   Interrupt Controller

5.1   Features

- Two interrupt control modes

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

- Priority can be assigned by the interrupt priority register (IPR)

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

  — NMI
  — Illegal instructions
  — Trace
  — Trap instructions
  — CPU address error
  — DMA address error (occurred in the DMAC)

- Independent vector addresses

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

- Seventeen external interrupts

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

- DMAC control

  DMAC can be activated by means of interrupts.

- CPU priority control function

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

A block diagram of the interrupt controller is shown in figure 5.1.
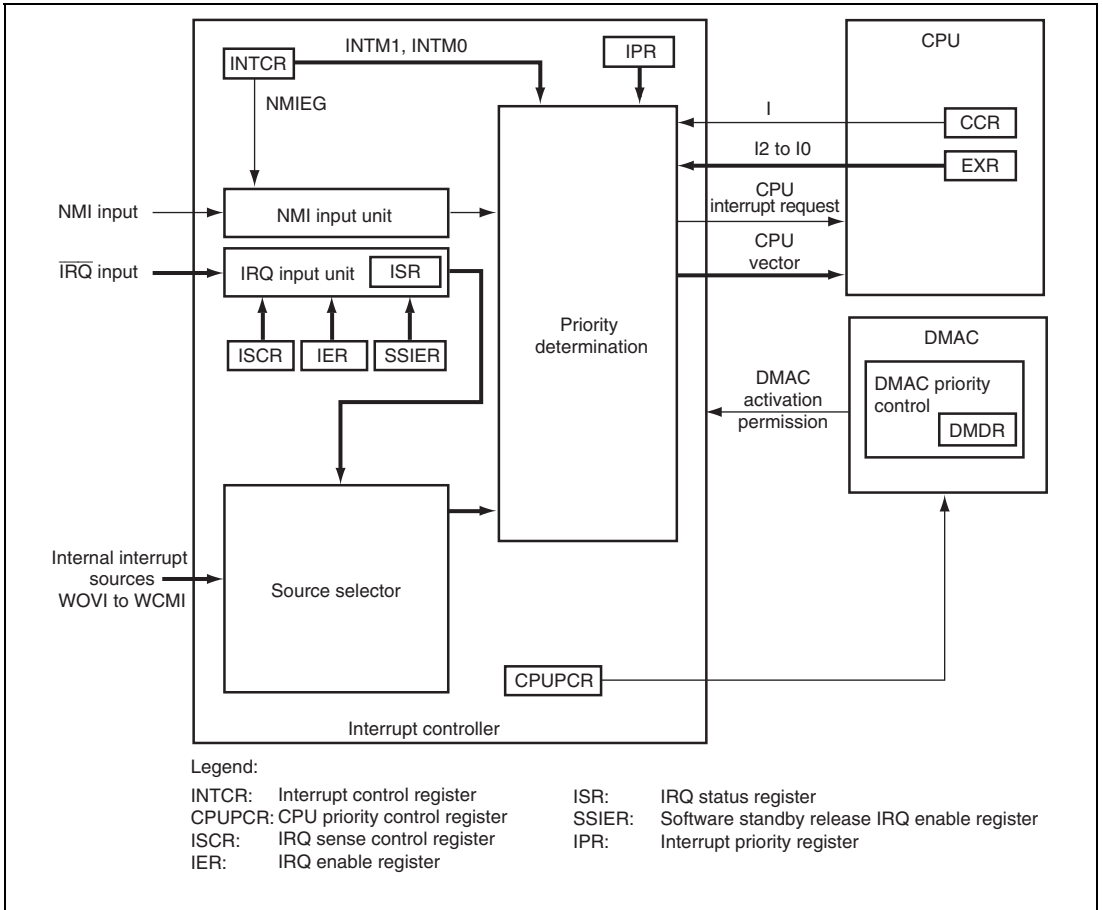
Figure 5.1   Block Diagram of Interrupt Controller

RENESAS

5.2   Input/Output Pins

Table 5.1 shows the pin configuration of the interrupt controller.

Table 5.1   Pin Configuration

Name	I/O	Function
NMI	Input	Nonmaskable External Interrupt
		Rising or falling edge can be selected.
$\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$	Input	Maskable External Interrupts
		Rising, falling, or both edges, or level sensing, can be selected.

5.3   Register Descriptions

The interrupt controller has the following registers.

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

5.3.1   Interrupt Control Register (INTCR)

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

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

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

RENESAS

5.3.2 CPU Priority Control Register (CPUPCR)

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

Bit	7	6	5	4	3	2	1	0
Bit Name	CPUPCE	—	—	—	IPSETE	CPUP2	CPUP1	CPUP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/(W)*	R/(W)*	R/(W)*

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

Bit	Bit Name	Initial Value	R/W	Description
7	CPUPCE	0	R/W	CPU Priority Control Enable
				Controls the CPU priority control function. Setting this bit to 1 enables the CPU priority control over the DMAC.
				0: CPU always has the lowest priority
				1: CPU priority control enabled
6 to 4	—	All 0	R	Reserved
				These bits are always read as 0. The write value should always be 0.
3	IPSETE	0	R/W	Interrupt Priority Set Enable
				Controls the function which automatically assigns the interrupt priority level of the CPU. Setting this bit to 1 automatically sets bits CPUP2 to CPUP0 by the CPU interrupt mask bit (I bit in CCR or bits I2 to I0 in EXR).
				0: Bits CPUP2 to CPUP0 are not updated automatically
				1: The interrupt mask bit value is reflected in bits CPUP2 to CPUP0

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

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

5.3.3   Interrupt Priority Registers A to G, I, K, L, O, Q, and R (IPRA to IPRG, IPRI, IPRK, IPRL, IPRO, IPRQ, and IPRR)

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

Setting a value in the range from B'000 to B'111 in the 3-bit groups of bits 14 to 12, 10 to 8, 6 to 4, and 2 to 0 assigns a priority level to the corresponding interrupt. For the correspondence between the interrupt sources and the IPR settings, see table 5.2.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	IPR14	IPR13	IPR12	—	IPR10	IPR9	IPR8
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved
				This is a read-only bit and cannot be modified.
14	IPR14	1	R/W	Sets the priority level of the corresponding interrupt source.
13	IPR13	1	R/W	
12	IPR12	1	R/W	000: Priority level 0 (lowest)
				001: Priority level 1
				010: Priority level 2
				011: Priority level 3
				100: Priority level 4
				101: Priority level 5
				110: Priority level 6
				111: Priority level 7 (highest)
11	—	0	R	Reserved
				This is a read-only bit and cannot be modified.
10	IPR10	1	R/W	Sets the priority level of the corresponding interrupt source.
9	IPR9	1	R/W	
8	IPR8	1	R/W	000: Priority level 0 (lowest)
				001: Priority level 1
				010: Priority level 2
				011: Priority level 3
				100: Priority level 4
				101: Priority level 5
				110: Priority level 6
				111: Priority level 7 (highest)
7	—	0	R	Reserved
				This is a read-only bit and cannot be modified.

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

RENESAS

5.3.4   IRQ Enable Register (IER)

IER enables or disables interrupt requests IRQ15 to IRQ0.

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ15E	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
8	IRQ8E	0	R/W	IRQ8 Enable
				The IRQ8 interrupt request is enabled when this bit is 1.
7	IRQ7E	0	R/W	IRQ7 Enable
				The IRQ7 interrupt request is enabled when this bit is 1.
6	IRQ6E	0	R/W	IRQ6 Enable
				The IRQ6 interrupt request is enabled when this bit is 1.
5	IRQ5E	0	R/W	IRQ5 Enable
				The IRQ5 interrupt request is enabled when this bit is 1.
4	IRQ4E	0	R/W	IRQ4 Enable
				The IRQ4 interrupt request is enabled when this bit is 1.
3	IRQ3E	0	R/W	IRQ3 Enable
				The IRQ3 interrupt request is enabled when this bit is 1.
2	IRQ2E	0	R/W	IRQ2 Enable
				The IRQ2 interrupt request is enabled when this bit is 1.
1	IRQ1E	0	R/W	IRQ1 Enable
				The IRQ1 interrupt request is enabled when this bit is 1.
0	IRQ0E	0	R/W	IRQ0 Enable
				The IRQ0 interrupt request is enabled when this bit is 1.

RENESAS

5.3.5   IRQ Sense Control Registers H and L (ISCRH, ISCRL)

ISCRH and ISCRL select the source that generates an interrupt request on pins $\overline{\text{IRQ15}}$ to $\overline{\text{IRQ0}}$.

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

- ISCRH

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ15SR	IRQ15SF	IRQ14SR	IRQ14SF	IRQ13SR	IRQ13SF	IRQ12SR	IRQ12SF
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- ISCRL

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- ISCRH

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

RENESAS

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

RENESAS

• ISCRL

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

RENESAS

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

RENESAS

5.3.6   IRQ Status Register (ISR)

ISR is an IRQ15 to IRQ0 interrupt request register.

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ15F	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: *   Only 0 can be written, to clear the flag. The bit manipulation instructions or memory operation instructions should be used to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ15F	0	R/(W)*	[Setting condition]
14	IRQ14F	0	R/(W)*	• When the interrupt selected by ISCR occurs
13	IRQ13F	0	R/(W)*	[Clearing conditions]
12	IRQ12F	0	R/(W)*	• Writing 0 after reading IRQnF = 1
11	IRQ11F	0	R/(W)*	• When interrupt exception handling is executed when
10	IRQ10F	0	R/(W)*	low-level sensing is selected and $\overline{IRQn}$ input is high
9	IRQ9F	0	R/(W)*	• When IRQn interrupt exception handling is executed
8	IRQ8F	0	R/(W)*	when falling-, rising-, or both-edge sensing is
7	IRQ7F	0	R/(W)*	selected
6	IRQ6F	0	R/(W)*	
5	IRQ5F	0	R/(W)*	
4	IRQ4F	0	R/(W)*	
3	IRQ3F	0	R/(W)*	
2	IRQ2F	0	R/(W)*	
1	IRQ1F	0	R/(W)*	
0	IRQ0F	0	R/(W)*	

Note: *   Only 0 can be written, to clear the flag. The flag should be cleared by a bit manipulation instruction or memory operation instruction.

RENESAS

5.3.7   Software Standby Release IRQ Enable Register (SSIER)

SSIER selects pins used to leave software standby mode from pins $\overline{IRQ15}$ to $\overline{IRQ0}$.

Bit	15	14	13	12	11	10	9	8
Bit Name	SSI15	SSI14	SSI13	SSI12	SSI11	SSI10	SSI9	SSI8
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

5.4   Interrupt Sources

5.4.1   External Interrupts

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

(1)   NMI Interrupts

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

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

- Sets the ERRF bit of DMDR_0 in the DMAC to 1
- Clears the DTE bits of DMDRs for all channels in the DMAC to 0 to forcibly terminate transfer

(2)   IRQn Interrupts

An IRQn interrupt is requested by a signal input on pins $\overline{\text{IRQn}}$ (n = 15 to 0). $\overline{\text{IRQn}}$ have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, on pins $\overline{\text{IRQn}}$.
- IRQn interrupt requests can be selected by IER.
- The interrupt priority can be set by IPR.
- The status of interrupt requests IRQn is indicated in ISR. ISR flags can be cleared to 0 by software. The bit manipulation instructions and memory operation instructions should be used to clear the flag.

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

RENESAS

A block diagram of interrupts IRQn is shown in figure 5.2.
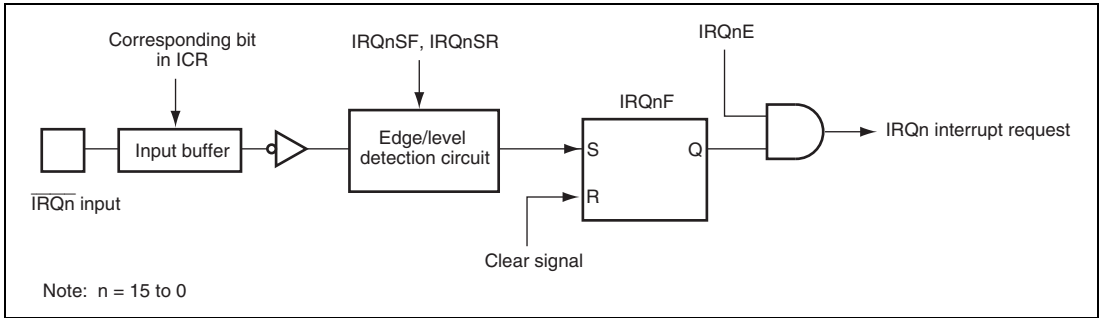
Figure 5.2   Block Diagram of Interrupts IRQn

When ISCR is set so that an IRQn interrupt request is generated at $\overline{\text{IRQn}}$ low-level input, $\overline{\text{IRQn}}$ input should be held low until interrupt handling starts. Then set the corresponding input signal $\overline{\text{IRQn}}$ to high in the interrupt handling routine and clear the IRQnF to 0. Interrupts may not be executed when the corresponding input signal $\overline{\text{IRQn}}$ is set to high before the interrupt handling begins.

5.4.2   Internal Interrupts

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

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

5.5   Interrupt Exception Handling Vector Table

Table 5.2 lists interrupt exception handling sources, vector address offsets, and interrupt priority.

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

Table 5.2   Interrupt Sources, Vector Address Offsets, and Interrupt Priority

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

Interrupt Source	Interrupt Name	Vector Number	Vector Address	IPR	DMAC Activation
Reserved	Reserved for system use	82	H'0148	—	—
		83	H'014C		—
		84	H'015C		—
		85	H'0154		—
A/D_0	ADI0	86	H'0158	IPRF10 to IPRF8	O
A/D_1	ADI1	87	H'015C		O
TPU_0	TGI0A	88	H'0160	IPRF6 to IPRF4	O
	TGI0B	89	H'0164		—
	TGI0C	90	H'0168		—
	TGI0D	91	H'016C		—
	TCI0V	92	H'0170		—
TPU_1	TGI1A	93	H'0174	IPRF2 to IPRF0	O
	TGI1B	94	H'0178		—
	TCI1V	95	H'017C		—
	TCI1U	96	H'0180		—
TPU_2	TGI2A	97	H'0184	IPRG14 to IPRG12	O
	TGI2B	98	H'0188		—
	TCI2V	99	H'018C		—
	TCI2U	100	H'0190		—
TPU_3	TGI3A	101	H'0194	IPRG10 to IPRG8	O
	TGI3B	102	H'0198		—
	TGI3C	103	H'019C		—
	TGI3D	104	H'01A0		—
	TCI3V	105	H'01A4		—
TPU_4	TGI4A	106	H'01A8	IPRG6 to IPRG4	O
	TGI4B	107	H'01AC		—
	TCI4V	108	H'01B0		—
	TCI4U	109	H'01B4		—

Interrupt Source	Interrupt Name	Vector Number	Vector Address	IPR	DMAC Activation
TPU_5	TGI5A	110	H'01B8	IPRG2 to IPRG0	O
	TGI5B	111	H'01BC		—
	TCI5V	112	H'01C0		—
	TCI5U	113	H'01C4		—
Reserved	Reserved for system use	114 | 127	H'01C8 | H'01FC	—	—
DMAC	DMTEND0	128	H'0200	IPRI14 to IPRI12	—
	DMTEND1	129	H'0204	IPRI10 to IPRI8	—
	DMTEND2	130	H'0208	IPRI6 to IPRI4	—
	DMTEND3	131	H'020C	IPRI2 to IPRI0	—
Reserved	Reserved for system use	132	H'0210	—	—
		133	H'0214		—
		134	H'0218		—
		135	H'021C		—
DMAC	DMEEND0	136	H'0220	IPRK14 to IPRK12	—
	DMEEND1	137	H'0224		—
	DMEEND2	138	H'0228		—
	DMEEND3	139	H'022C		—
Reserved	Reserved for system use	140	H'0230	—	—
		141	H'0234		—
		142	H'0238		—
		143	H'023C		—
SCI_0	ERI_0	144	H'0240	IPRK6 to IPRK4	—
	RXI_0	145	H'0244		O
	TXI_0	146	H'0248		O
	TEI_0	147	H'024C		—

RENESAS

Interrupt Source	Interrupt Name	Vector Number	Vector Address	IPR	DMAC Activation
Reserved	Reserved for system use	148	H'0250	—	—
		149	H'0254		—
		150	H'0258		—
		151	H'025C		—
SCI_2	ERI_2	152	H'0260	IPRL14 to IPRL12	—
	RXI_2	153	H'0264		O
	TXI_2	154	H'0268		O
	TEI_2	155	H'026C		—
Reserved	Reserved for system use	156 | 159	H'0270 | H'027C	—	—
SCI_4	ERI_4	160	H'0280	IPRL6 to IPRL4	—
	RXI_4	161	H'0284		O
	TXI_4	162	H'0288		O
	TEI_4	163	H'028C		—
Reserved	Reserved for system use	164 | 191	H'0290 | H'02FC	—	—
SCI_5	ERI_5	192	H'0300	IPRO2 to IPRO0	—
	RXI_5	193	H'0304		O
	TXI_5	194	H'0308		O
	TEI_5	195	H'030C		—
Reserved	Reserved for system use	196 | 215	H'0310 | H'035C	—	—
IIC2	IICI0	216	H'0360	IPRQ6 to IPRQ4	—
	Reserved for system use	217	H'0364		—
	IICI1	218	H'0368		—
	Reserved for system use	219	H'036C		—

Interrupt Source	Interrupt Name	Vector Number	Vector Address	IPR	DMAC Activation
RCAN-ET_0	RM0_0	220	H'0370	IPRQ2 to IPRQ0	O
RCAN-ET_1	RM0_1	221	H'0374		O
RCAN-ET_0	ERS0_0/OVR0_0/RM1_0/ SLE0_0	222	H'0378		—
RCAN-ET_1	ERS0_1/OVR0_1/RM1_1/ SLE0_1	223	H'037C		—
Motor control PWM_0	CMI0_0	224	H'0380	IPRR14 to IPRR12	O
Motor control PWM_1	CMI1_0	225	H'0384		O
Watch timer (WAT)	WCMI	226	H'0388		—
Reserved	Reserved for system use	227	H'038C		—
16-bit PWM_0	CMI0_1	228	H'0390	IPRR10 to IPRR8	O
16-bit PWM_1	CMI1_1	229	H'0394		O
16-bit PWM_2	CMI2_1	230	H'0398		—
Reserved	Reserved for system use	231	H'039C		—
SDG_0	SGI_0	232	H'03A0	IPRR6 to IPRR4	O
SDG_1	SGI_1	233	H'03A4		O
SSU*[1]_0	SSERI_0/SSRXI_0/ SSTXI_0	234	H'03A8		—
SSU*[1]_1	SSERI_1/SSRXI_1/ SSTXI_1	235	H'03AC		—
SDG_2	SGI_2	236	H'03B0	IPRR2 to IPRR0	O
SDG_3	SGI_3	237	H'03B4		O
Watch timer (WAT)*[2]	WCMI	238	H'03B8		—
Reserved	Reserved for system use	239	H'03BC		—

Notes: 1. SSU: Synchronous Serial communication Unit
2. Software standby mode is cancelled when this interrupt is generated.

RENESAS

5.6   Interrupt Control Modes and Interrupt Operation

The interrupt controller has two interrupt control modes: interrupt control mode 0 and interrupt control mode 2. Interrupt operations differ depending on the interrupt control mode. The interrupt control mode is selected by INTCR. Table 5.3 shows the differences between interrupt control mode 0 and interrupt control mode 2.

Table 5.3   Interrupt Control Modes

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

5.6.1   Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests except for NMI are masked by the I bit in CCR of the CPU. Figure 5.3 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, the interrupt request is sent to the interrupt controller.
2. If the I bit in CCR is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending. If the I bit is cleared to 0, an interrupt request is accepted.
3. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority, sends the request to the CPU, and holds other interrupt requests pending.
4. When the CPU accepts the interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR contents are saved to the stack area during the interrupt exception handling. The PC contents saved on the stack are the address of the first instruction to be executed after returning from the interrupt handling routine.
6. Next, the I bit in CCR is set to 1. This masks all interrupts except NMI.

7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.
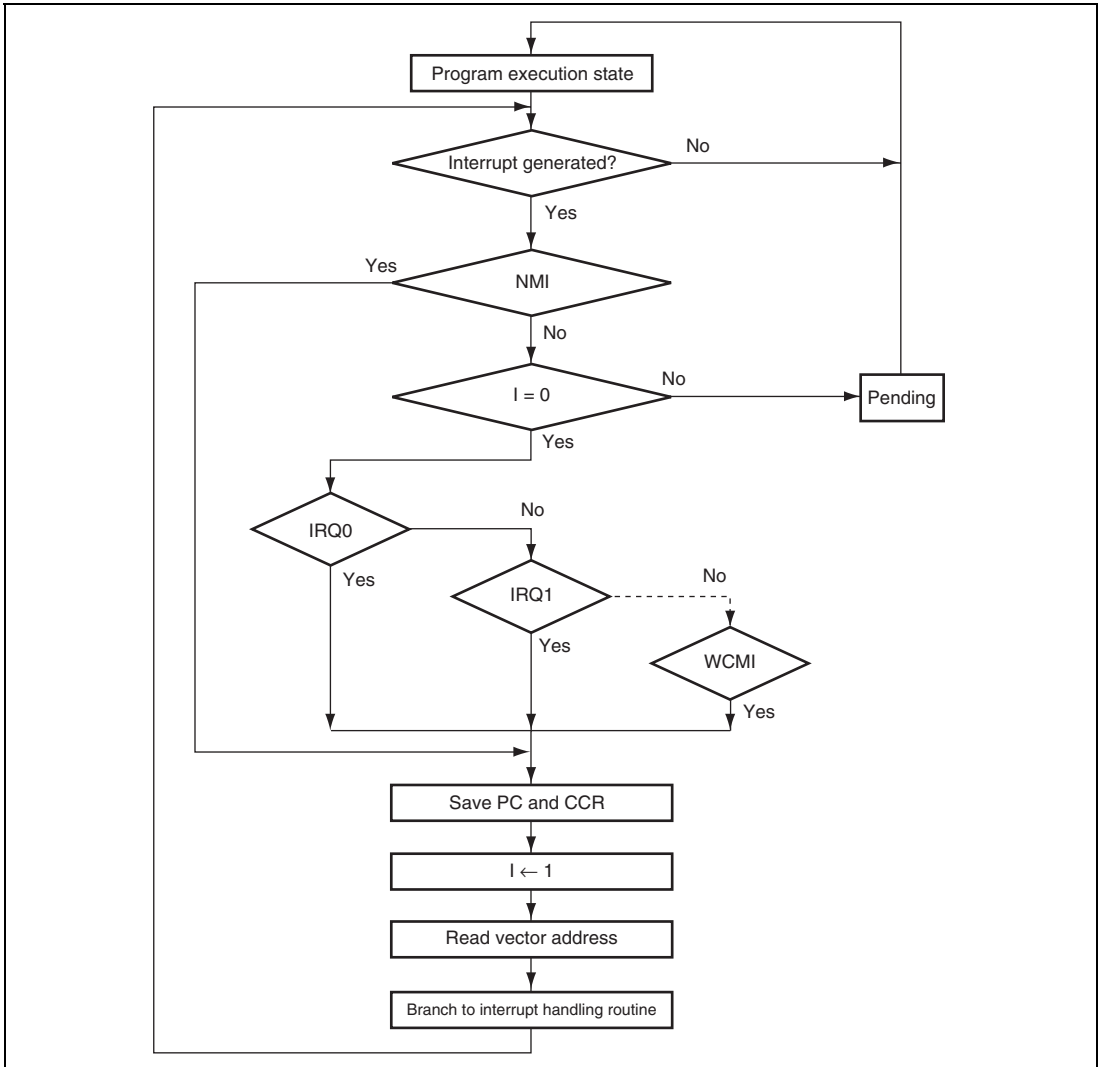
**Figure 5.3   Flowchart of Procedure Up to Interrupt Acceptance
in Interrupt Control Mode 0**

5.6.2   Interrupt Control Mode 2

In interrupt control mode 2, interrupt requests except for NMI are masked by comparing the interrupt mask level (I2 to I0 bits) in EXR of the CPU and the IPR setting. There are eight levels in mask control. Figure 5.4 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority according to the IPR setting, and holds other interrupt requests pending. If multiple interrupt requests have the same priority, an interrupt request is selected according to the default setting shown in table 5.2.
3. Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. When the interrupt request does not have priority over the mask level set, it is held pending, and only an interrupt request with a priority over the interrupt mask level is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC, CCR, and EXR contents are saved to the stack area during interrupt exception handling. The PC saved on the stack is the address of the first instruction to be executed after returning from the interrupt handling routine.
6. The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority of the accepted interrupt. If the accepted interrupt is NMI, the interrupt mask level is set to H'7.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.
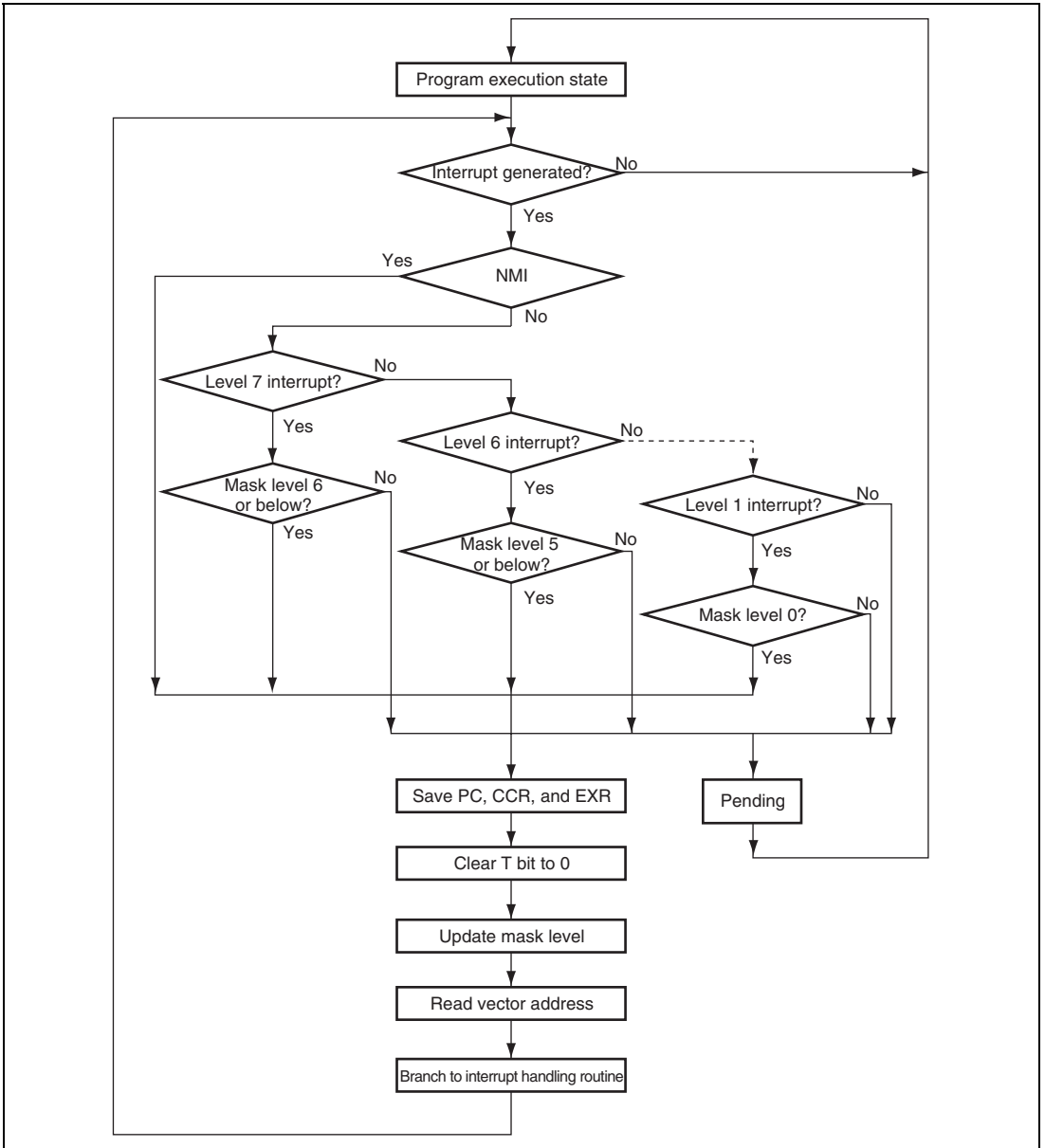
**Figure 5.4   Flowchart of Procedure Up to Interrupt Acceptance
in Interrupt Control Mode 2**

5.6.3   Interrupt Exception Handling Sequence

Figure 5.5 shows the interrupt exception handling sequence. The example is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.
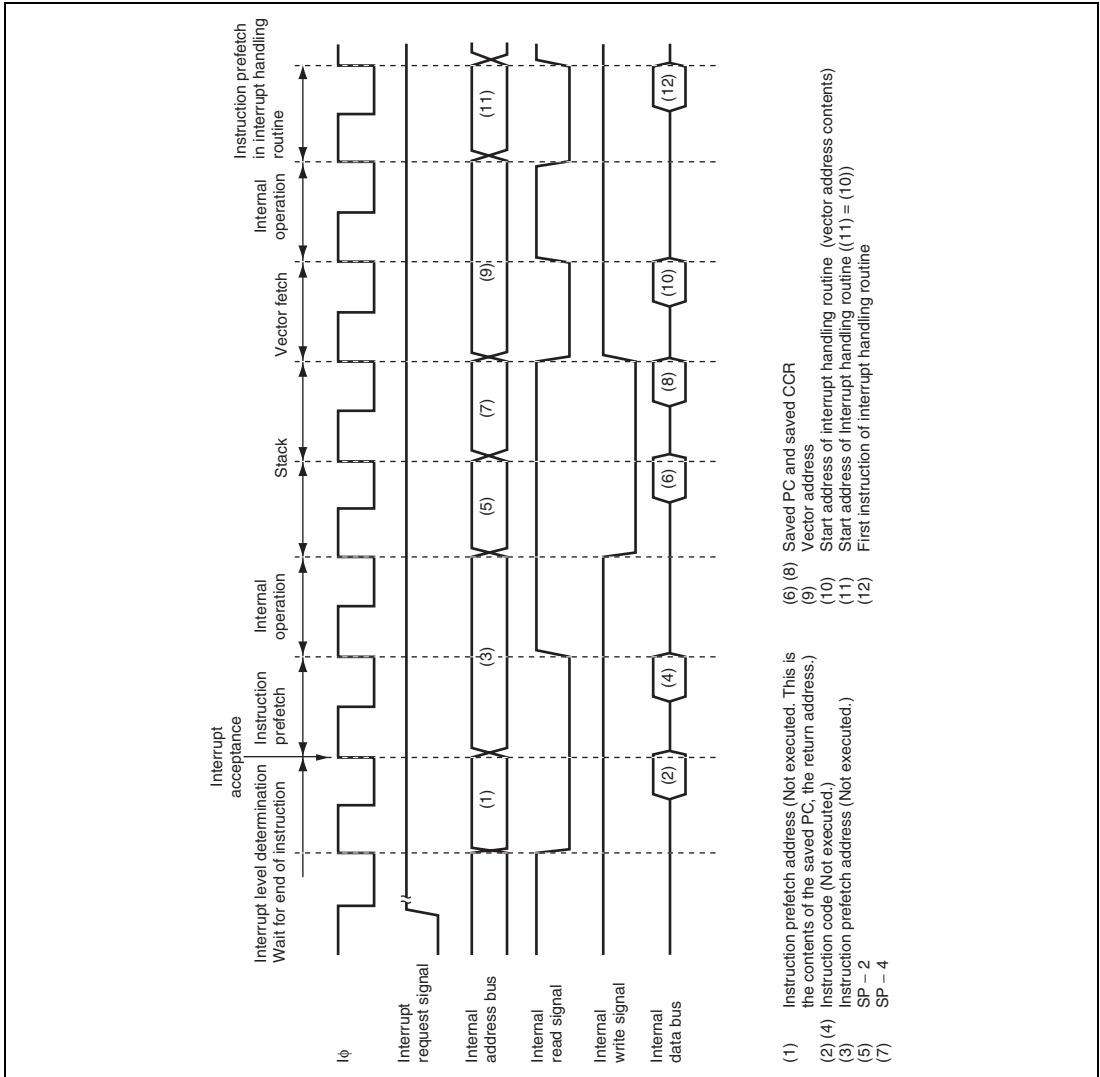
Figure 5.5   Interrupt Exception Handling

5.6.4 Interrupt Response Times

Table 5.4 shows interrupt response times—the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The symbols for execution states used in table 5.4 are explained in table 5.5.

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

Table 5.4   Interrupt Response Times

Execution State	Normal Mode[5]		Advanced Mode		Maximum Mode[5]	
	Interrupt Control Mode 0	Interrupt Control Mode 2	Interrupt Control Mode 0	Interrupt Control Mode 2	Interrupt Control Mode 0	Interrupt Control Mode 2
Interrupt priority determination[1]			3			
Number of states until executing instruction ends[2]			1 to $19 + 2 \cdot S_I$			
PC, CCR, EXR stacking	S_K to $2 \cdot S_K$[6]	$2 \cdot S_K$	S_K to $2 \cdot S_K$[6]	$2 \cdot S_K$	$2 \cdot S_K$	$2 \cdot S_K$
Vector fetch			S_h			
Instruction fetch[3]			$2 \cdot S_I$			
Internal processing[4]			2			
Total (using on-chip memory)	10 to 31	11 to 31	10 to 31	11 to 31	11 to 31	11 to 31

Notes: 1. Two states for an internal interrupt.
2. In the case of the MULXS or DIVXS instruction
3. Prefetch after interrupt acceptance or for an instruction in the interrupt handling routine.
4. Internal operation after interrupt acceptance or after vector fetch
5. Not available in this LSI.
6. When setting the SP value to 4n, the interrupt response time is S_K; when setting to 4n + 2, the interrupt response time is $2 \cdot S_K$.

RENESAS

Table 5.5   Number of Execution States in Interrupt Handling Routine

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

Legend:

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

5.6.5   DMAC Activation by Interrupt

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

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

For details on interrupt requests that can be used to activate the DMAC, see table 5.2 and section 7, DMA Controller (DMAC).

Figure 5.6 shows a block diagram of the DMAC and interrupt controller.
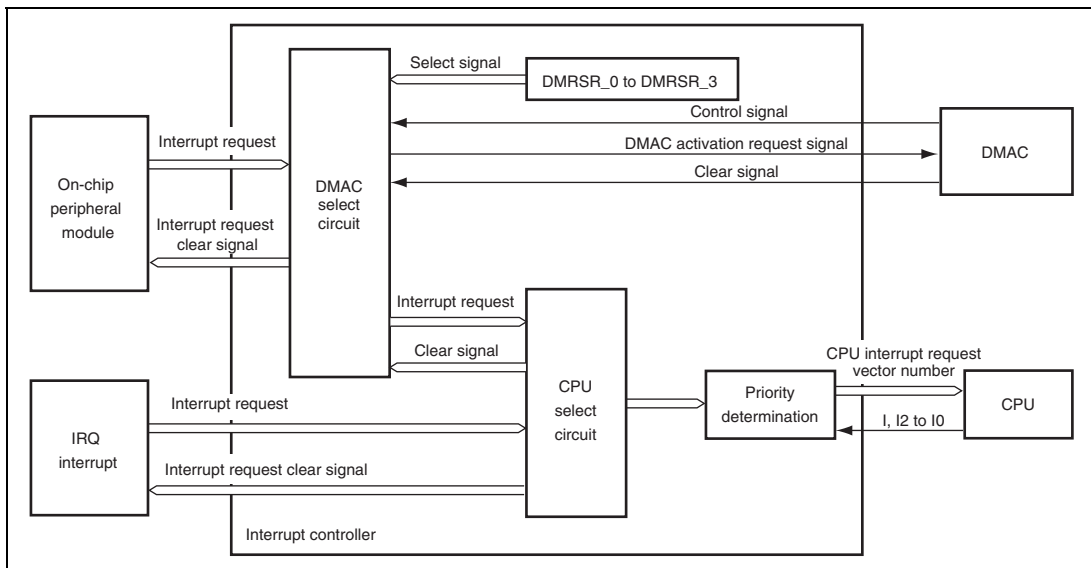
Figure 5.6   Block Diagram of DMAC and Interrupt Controller

(1)   Selection of Interrupt Sources

A DMAC activation request source for each channel is specified by DMRSR. The request is passed to the DMAC via a selector. When the DTA bit in DMDR is set to 1, the specified request source cannot be used as a CPU interrupt source. Other interrupt sources, meaning that interrupt sources are not controlled by the DMAC, are used as CPU interrupt sources.

When the same interrupt source is set as both the DMAC activation source and CPU interrupt source, the DMAC must be given priority over the CPU. Otherwise, DMAC transfer may not be performed or the DMAC may malfunction.

(2)   Operation Order

If the same interrupt is selected as both the DMAC activation source and CPU interrupt source, the respective operations are performed independently.

Table 5.6 lists the selection of interrupt sources and interrupt source clear control by means of the setting of the DTA bit in DMDR of the DMAC.

Table 5.6   Interrupt Source Selection and Clear Control

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

Legend:

√:   The corresponding interrupt is used. The interrupt source is cleared.

    (The interrupt source flag must be cleared in the CPU interrupt handling routine.)

O: The corresponding interrupt is used. The interrupt source is not cleared.

X: The corresponding interrupt is not available.

(3)   Usage Note

The interrupt sources of the SCI and A/D converter are cleared according to the setting shown in table 5.6, when the DMAC reads/writes the prescribed register.

To initiate multiple channels for the DMAC with the same interrupt, the same priority should be assigned.

5.7   CPU Priority Control Function over DMAC

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

The priority level of the CPU is assigned by bits CPUP2 to CPUP0 in CPUPCR. The priority level of the DMAC is assigned to each channel by bits DMAP2 to DMAP0 in the DMA mode control registers 0 to 3 (DMDR_0 to DMDR_3).

The priority control function over the DMAC is enabled by setting the CPUPCE bit in CPUPCR to 1. When the CPUPCE bit is 1, the DMAC activation source is controlled according to the respective priority level.

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

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

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

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

RENESAS

Table 5.7 shows the CPU priority control.

Table 5.7   CPU Priority Control

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

Table 5.8 shows a setting example of the priority control function over the DMAC and the transfer request control state. Although the DMAC priority levels can be assigned for each channel, table 5.8 gives a single channel description. Thus, transfer for each channel can be performed independently by assigning the different priority levels.

Table 5.8   Example of Priority Control Function Setting and Control State

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

RENESAS

5.8   Usage Notes

5.8.1   Conflict between Interrupt Generation and Disabling

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

When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request with priority over that interrupt, interrupt exception handling will be executed for the interrupt with priority, and another interrupt will be ignored. The same also applies when an interrupt source flag is cleared to 0. Figure 5.7 shows an example in which the TCIEV bit in TIER of the TPU is cleared to 0. The above conflict will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.
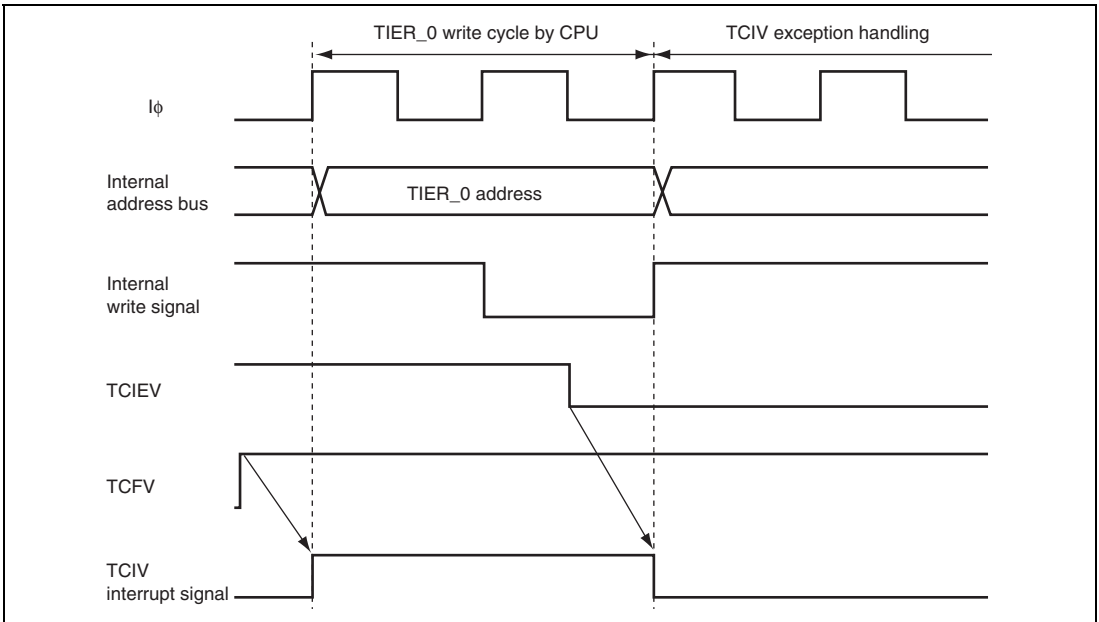
Figure 5.7   Conflict between Interrupt Generation and Disabling

5.8.2 Instructions that Disable Interrupts

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

5.8.3 Times when Interrupts Are Disabled

There are times when interrupt acceptance is disabled by the interrupt controller.

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

5.8.4 Interrupts during Execution of EEPMOV Instruction

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

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

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

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

5.8.5 Interrupts during Execution of MOVMD and MOVSD Instructions

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

5.8.6 Interrupt Flags of Peripheral Modules

To clear an interrupt request flag of a peripheral module by the CPU, the flag must be read from after being cleared in the interrupt service routine if a peripheral module interrupt request is used. This makes the request signal synchronized with the system clock.

RENESAS

Section 6   Bus Controller (BSC)

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

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

6.1   Features

- Manages external address space in area units

  Manages the external address space divided into eight areas

  Bus specifications can be set independently for each area

  8-bit access or 16-bit access can be selected for each area

  An endian conversion function is provided to connect a device of little endian

- Basic bus interface

  This interface can be connected to the SRAM and ROM

  2-state access or 3-state access can be selected for each area

  Program wait cycles can be inserted for each area

  The negation timing of the read strobe signal ($\overline{RD}$) can be modified

- Idle cycle insertion

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

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

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

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

- Write buffer function

  External write cycles and internal accesses can be executed in parallel

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

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

- Bus arbitration function

  Includes a bus arbiter that arbitrates bus mastership among the CPU and DMAC

- Multi-clock function

  The on-chip peripheral functions can be operated in synchronization with the peripheral module clock (Pφ). Accesses to the external address space can be operated in synchronization with the external bus clock (Bφ).

A block diagram of the bus controller is shown in figure 6.1.
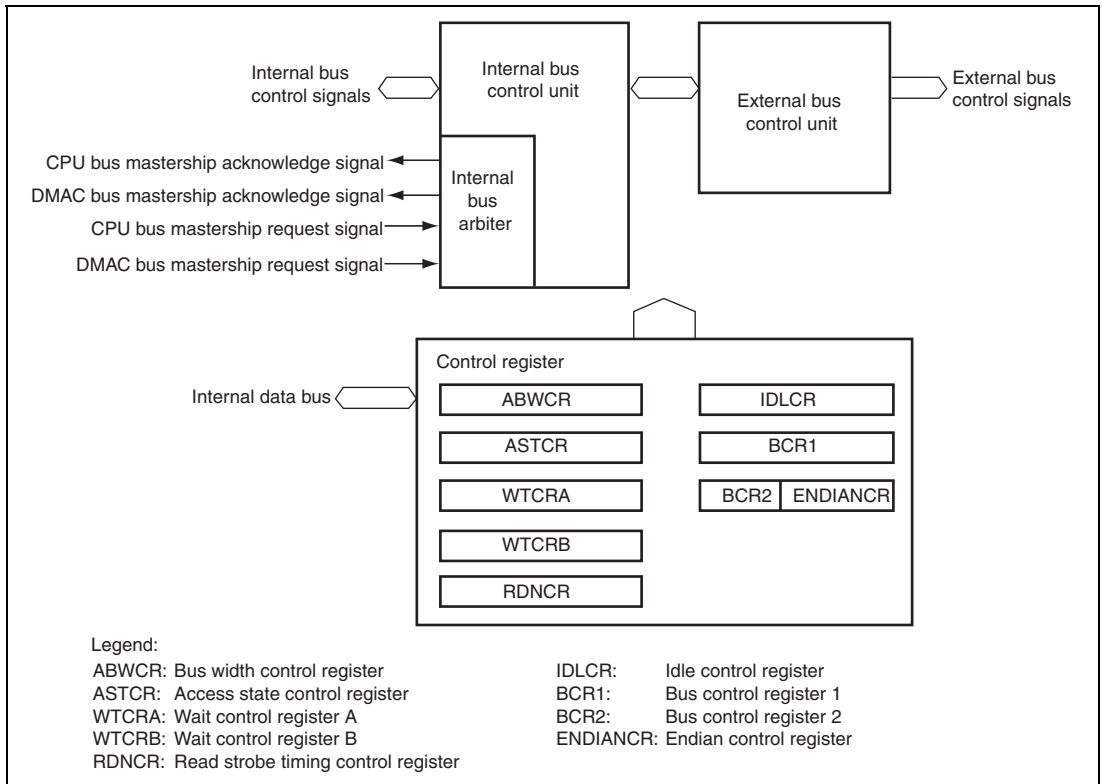
Figure 6.1   Block Diagram of Bus Controller

6.2 Register Descriptions

The bus controller has the following registers.

- Bus width control register (ABWCR)
- Access state control register (ASTCR)
- Wait control register A (WTCRA)
- Wait control register B (WTCRB)
- Read strobe timing control register (RDNCR)
- Idle control register (IDLCR)
- Bus control register 1 (BCR1)
- Bus control register 2 (BCR2)
- Endian control register (ENDIANCR)

6.2.1   Bus Width Control Register (ABWCR)

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

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

Bit	7	6	5	4	3	2	1	0
Bit Name	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1	ABWL0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

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

Legend:

×: Don't care

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

RENESAS

6.2.2   Access State Control Register (ASTCR)

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

Bit	15	14	13	12	11	10	9	8
Bit Name	AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

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

6.2.3 Wait Control Registers A and B (WTCRA, WTCRB)

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

• WTCRA

Bit	15	14	13	12	11	10	9	8
Bit Name	—	W72	W71	W70	—	W62	W61	W60
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	W52	W51	W50	—	W42	W41	W40
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

• WTCRB

Bit	15	14	13	12	11	10	9	8
Bit Name	—	W32	W31	W30	—	W22	W21	W20
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	W12	W11	W10	—	W02	W01	W00
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

RENESAS

- WTCRA

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

RENESAS

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

RENESAS

• WTCRB

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

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

RENESAS

6.2.4   Read Strobe Timing Control Register (RDNCR)

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

Bit	15	14	13	12	11	10	9	8
Bit Name	RDN7	RDN6	RDN5	RDN4	RDN3	RDN2	RDN1	RDN0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
15	RDN7	0	R/W	Read Strobe Timing Control
14	RDN6	0	R/W	These bits set the negation timing of the read strobe in a corresponding area read access.
13	RDN5	0	R/W	
12	RDN4	0	R/W	As shown in figure 6.2, the read strobe for an area for which the RDNn bit is set to 1 is negated one half-cycle earlier than that for an area for which the RDNn bit is cleared to 0. The read data setup and hold time are also given one half-cycle earlier.
11	RDN3	0	R/W	
10	RDN2	0	R/W	
9	RDN1	0	R/W	
8	RDN0	0	R/W	0: In an area n read access, the $\overline{\text{RD}}$ signal is negated at the end of the read cycle
				1: In an area n read access, the $\overline{\text{RD}}$ signal is negated one half-cycle before the end of the read cycle
				(n = 7 to 0)
7 to 0	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

RENESAS
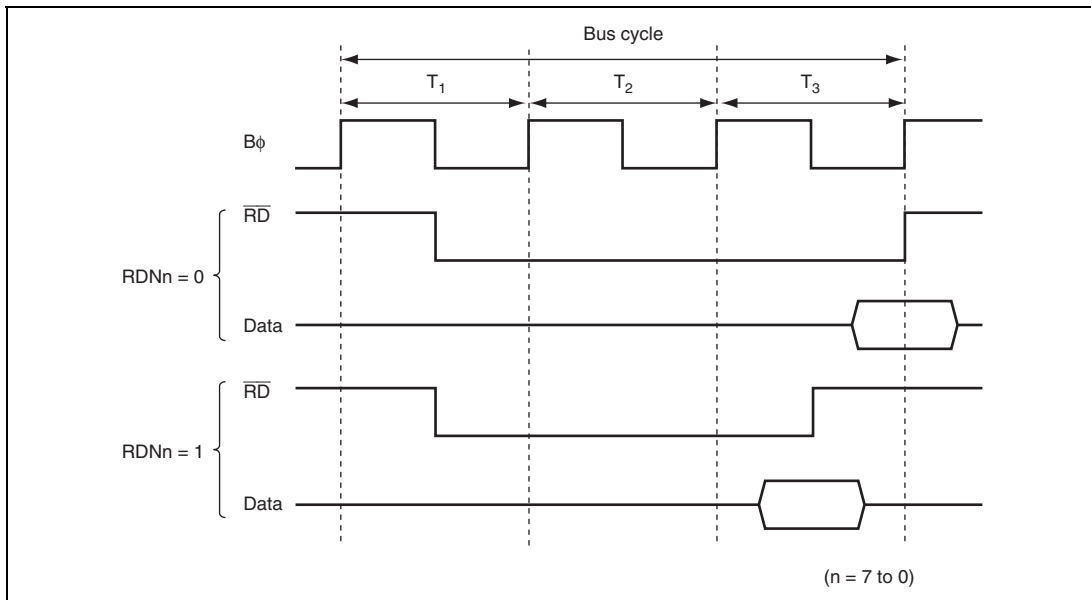
Figure 6.2   Read Strobe Negation Timing (Example of 3-State Access Space)

6.2.5   Idle Control Register (IDLCR)

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

Bit	15	14	13	12	11	10	9	8
Bit Name	IDLS3	IDLS2	IDLS1	IDLS0	IDLCB1	IDLCB0	IDLCA1	IDLCA0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	IDLSEL7	IDLSEL6	IDLSEL5	IDLSEL4	IDLSEL3	IDLSEL2	IDLSEL1	IDLSEL0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	IDLS3	1	R/W	Idle Cycle Insertion 3
				Inserts an idle cycle between the bus cycles when the DMAC single address transfer (write cycle) is followed by external access.
				0: No idle cycle is inserted
				1: An idle cycle is inserted
14	IDLS2	1	R/W	Idle Cycle Insertion 2
				Inserts an idle cycle between the bus cycles when the external write cycle is followed by external read cycle.
				0: No idle cycle is inserted
				1: An idle cycle is inserted
13	IDLS1	1	R/W	Idle Cycle Insertion 1
				Inserts an idle cycle between the bus cycles when the external read cycles of different areas continue.
				0: No idle cycle is inserted
				1: An idle cycle is inserted
12	IDLS0	1	R/W	Idle Cycle Insertion 0
				Inserts an idle cycle between the bus cycles when the external read cycle is followed by external write cycle.
				0: No idle cycle is inserted
				1: An idle cycle is inserted
11	IDLCB1	1	R/W	Idle Cycle State Number Select B
10	IDLCB0	1	R/W	Specifies the number of idle cycles to be inserted for the idle condition specified by IDLS1 and IDLS0.
				00: No idle cycle is inserted
				01: 2 idle cycles are inserted
				00: 3 idle cycles are inserted
				01: 4 idle cycles are inserted

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
9	IDLCA1	1	R/W	Idle Cycle State Number Select A
8	IDLCA0	1	R/W	Specifies the number of idle cycles to be inserted for the idle condition specified by IDLS3 to IDLS0.
				00: 1 idle cycle is inserted
				01: 2 idle cycles are inserted
				10: 3 idle cycles are inserted
				11: 4 idle cycles are inserted
7	IDLSEL7	0	R/W	Idle Cycle Number Select
6	IDLSEL6	0	R/W	Specifies the number of idle cycles to be inserted for
5	IDLSEL5	0	R/W	each area for the idle insertion condition specified by IDLS1 and IDLS0.
4	IDLSEL4	0	R/W	
3	IDLSEL3	0	R/W	0: Number of idle cycles to be inserted for area n is specified by IDLCA1 and IDLCA0.
2	IDLSEL2	0	R/W	1: Number of idle cycles to be inserted for area n is
1	IDLSEL1	0	R/W	specified by IDLCB1 and IDLCB0.
0	IDLSEL0	0	R/W	(n = 7 to 0)

6.2.6   Bus Control Register 1 (BCR1)

BCR1 is used for enabling/disabling of the write data buffer function.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	—	—	WDBE	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W

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

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
15, 14	⎯	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
13, 12	⎯	All 0	R	Reserved
				These are read-only bits and cannot be modified.
11, 10	⎯	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
9	WDBE	0	R/W	Write Data Buffer Enable
				The write data buffer function can be used for an external write cycle and a DMAC single address transfer cycle.
				The changed setting may not affect an external access immediately after the change.
				0: Write data buffer function not used
				1: Write data buffer function used
8	⎯	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.
7	DKC	0	R/W	$\overline{\text{DACK}}$ Control
				Selects the timing of DMAC transfer acknowledge signal assertion.
				0: $\overline{\text{DACK}}$ signal is asserted at the Bϕ falling edge
				1: $\overline{\text{DACK}}$ signal is asserted at the Bϕ rising edge
6	⎯	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.
5 to 0	⎯	All 0	R	Reserved
				These are read-only bits and cannot be modified.

RENESAS

6.2.7 Bus Control Register 2 (BCR2)

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

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

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

RENESAS

6.2.8   Endian Control Register (ENDIANCR)

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

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

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

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

6.3   Bus Configuration

Figure 6.3 shows the internal bus configuration of this LSI. The internal bus of this LSI consists of the following three types.

- Internal system bus

  A bus that connects the CPU, DMAC, on-chip RAM, on-chip ROM, internal peripheral bus, and external access bus.
- Internal peripheral bus

  A bus that accesses registers in the bus controller, interrupt controller, and DMAC, and registers of peripheral modules such as SCI and timer.
- External access cycle

  A bus that accesses external devices via the external bus interface.
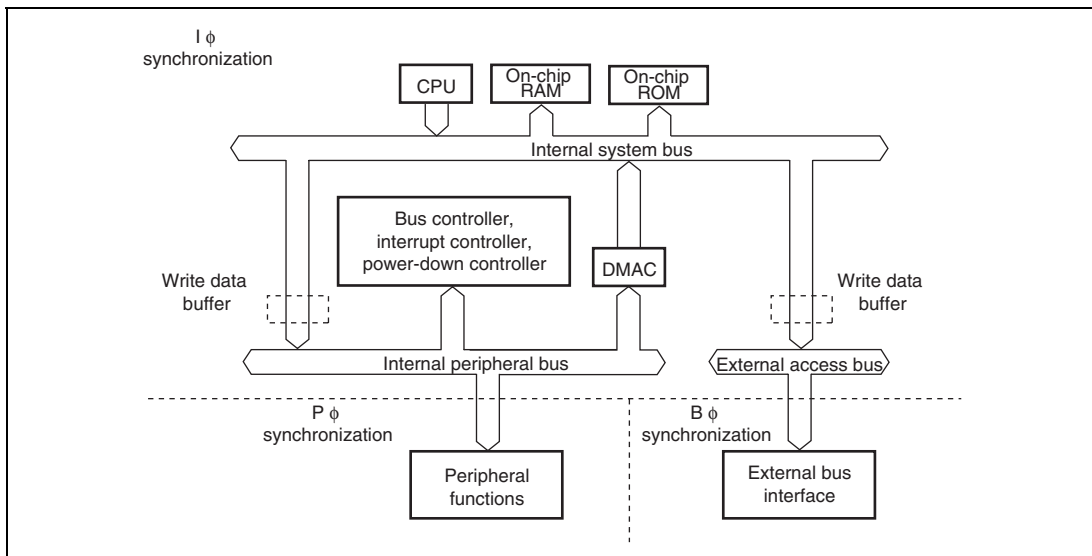
Figure 6.3   Internal Bus Configuration

6.4   Multi-Clock Function and Number of Access Cycles

The internal functions of this LSI operate synchronously with the system clock (Iφ), the peripheral module clock (Pφ), or the external bus clock (Bφ). Table 6.1 shows the synchronization clock and their corresponding functions.

Table 6.1   Synchronization Clocks and Corresponding Functions

Synchronization Clock	Function Name
Iφ	MCU operating mode
	Interrupt controller
	Bus controller
	CPU
	DMAC
	Internal memory
	Clock pulse generator
	Power down control
Pφ	I/O ports
	TPU
	PPG
	TMR
	WDT
	SCI
	IIC2
	A/D
	D/A
	SSU∗
	RCAN-ET
	WAT
	SDG
	Motor control PWM
	16-bit PWM
Bφ	External bus interface

Note: ∗   SSU: Synchronous Serial communication Unit

The frequency of each synchronization clock (Iφ, Pφ, and Bφ) is specified by the system clock control register (SCKCR) independently. For further details, see section 23, Clock Pulse Generator.

There will be cases when Pφ and Bφ are equal to Iφ and when Pφ and Bφ are different from Iφ according to the SCKCR specifications. In any case, access cycles for internal peripheral functions and external space is performed synchronously with Pφ and Bφ, respectively.

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

If the frequencies of Iφ, Pφ and Bφ are different, the start of bus cycle may not synchronize with Pφ or Bφ according to the bus cycle initiation timing. In this case, clock synchronization cycle (Tsy) is inserted at the beginning of each bus cycle.

For example, if an external address space access occurs when the frequency rate of Iφ and Bφ is n : 1, 0 to n-1 cycles of Tsy may be inserted. If an internal peripheral module access occurs when the frequency rate of Iφ and Pφ is m : 1, 0 to m-1 cycles of Tsy may be inserted.

Figure 6.4 shows the external 2-state access timing when the frequency rate of Iφ and Bφ is 4 : 1. Figure 6.5 shows the external 3-state access timing when the frequency rate of Iφ and Bφ is 2 : 1.
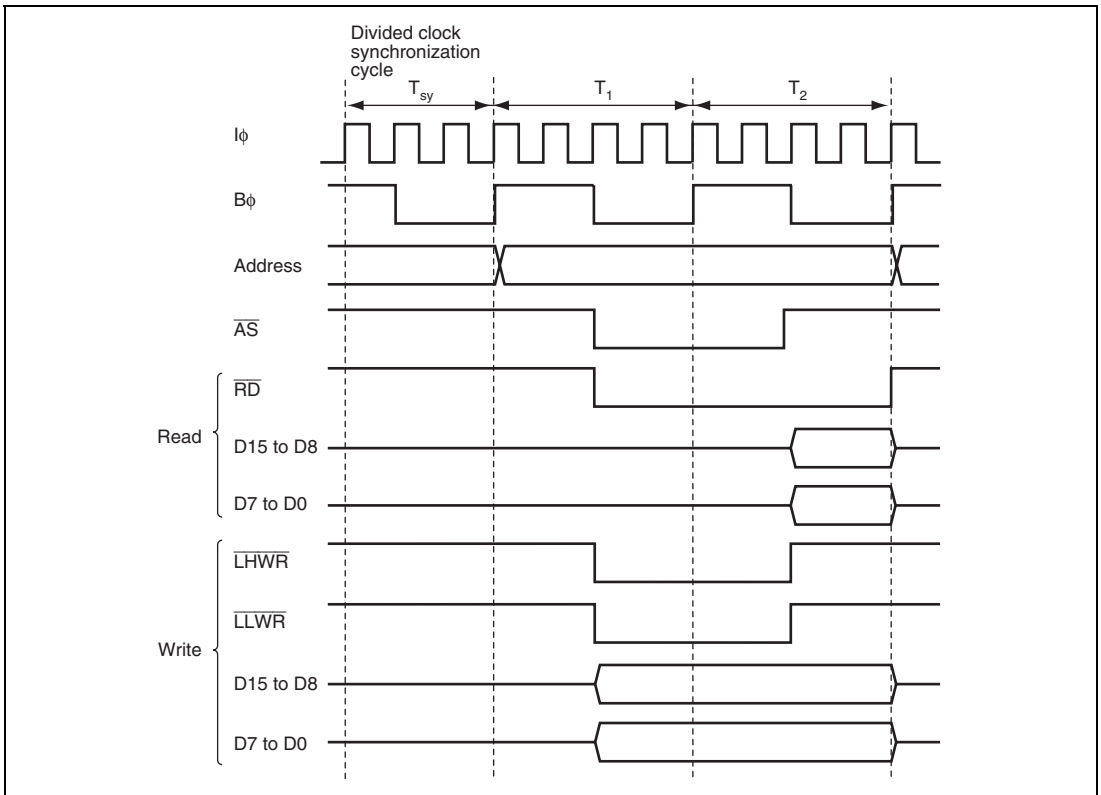
Figure 6.4   System Clock: External Bus Clock = 4:1, External 2-State Access
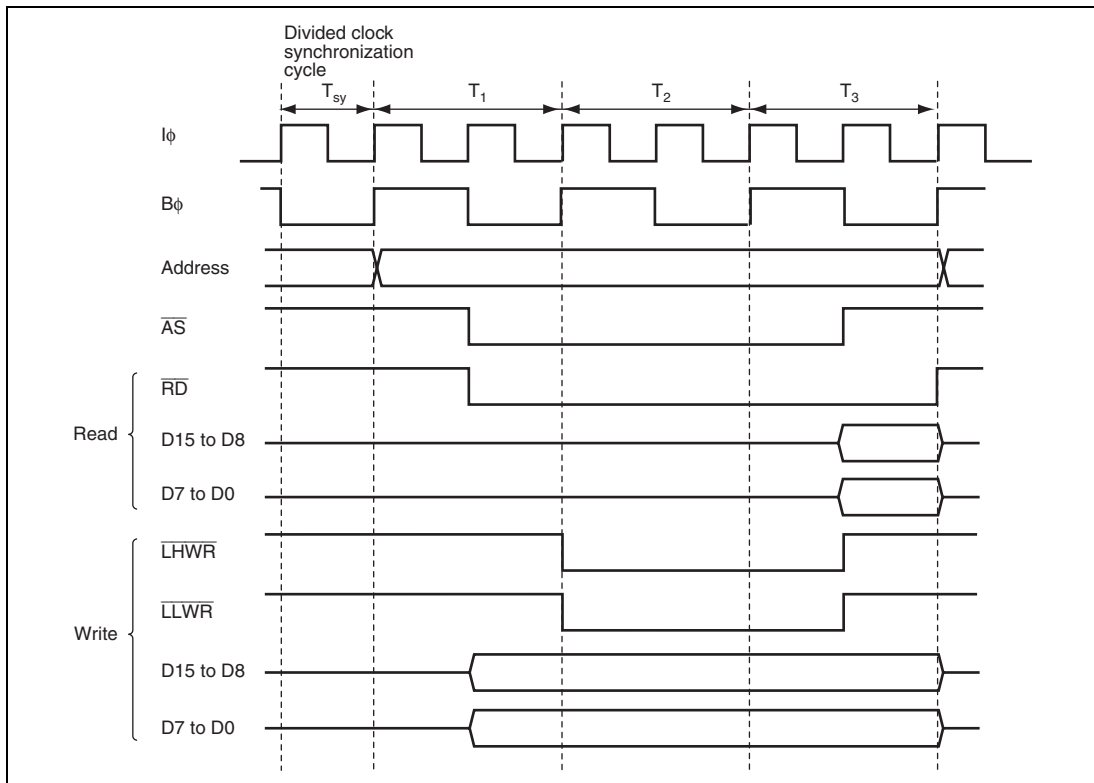
Figure 6.5   System Clock: External Bus Clock = 2:1, External 3-State Access

6.5 External Bus

6.5.1 Input/Output Pins

Table 6.2 shows the pin configuration of the bus controller and table 6.3 shows the pin functions on each interface.

Table 6.2   Pin Configuration

Name	Symbol	I/O	Function
Address strobe	$\overline{\text{AS}}$	Output	Strobe signal indicating that the basic bus space is being accessed and address output on address bus is valid
Read strobe	$\overline{\text{RD}}$	Output	Strobe signal indicating that the basic bus space is being read
Low-high write	$\overline{\text{LHWR}}$	Output	Strobe signal indicating that the basic bus space is being written to, and the upper byte (D15 to D8) of data bus is valid
Low-low write	$\overline{\text{LLWR}}$	Output	Strobe signal indicating that the basic bus space is being written to, and the lower byte (D7 to D0) of data bus is valid
Data transfer acknowledge 3 (DMAC_3)	$\overline{\text{DACK3}}$	Output	Data transfer acknowledge signal for DMAC_3 single address transfer
Data transfer acknowledge 2 (DMAC_2)	$\overline{\text{DACK2}}$	Output	Data transfer acknowledge signal for DMAC_2 single address transfer
Data transfer acknowledge 1 (DMAC_1)	$\overline{\text{DACK1}}$	Output	Data transfer acknowledge signal for DMAC_1 single address transfer
Data transfer acknowledge 0 (DMAC_0)	$\overline{\text{DACK0}}$	Output	Data transfer acknowledge signal for DMAC_0 single address transfer
External bus clock	Bϕ	Output	External bus clock

Table 6.3   Pin Functions in Each Interface

Pin Name	Initial State			Basic Bus		Remarks
	16	**8**	**Single-Chip**	**16**	**8**	
Bφ	Output	Output	—	O	O	
$\overline{\text{AS}}$	Output	Output	—	O	O	
$\overline{\text{RD}}$	Output	Output	—	O	O	
$\overline{\text{LHWR}}$	Output	Output	—	O	—	
$\overline{\text{LLWR}}$	Output	Output	—	O	O	

Legend:

O: Used as a bus control signal

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

6.5.2   Area Division

The bus controller divides the 16-Mbyte address space into eight areas, and performs bus control for the external address space in area units.

Figure 6.6 shows an area division of the 16-Mbyte address space. For details on address map, see section 3, MCU Operating Modes.
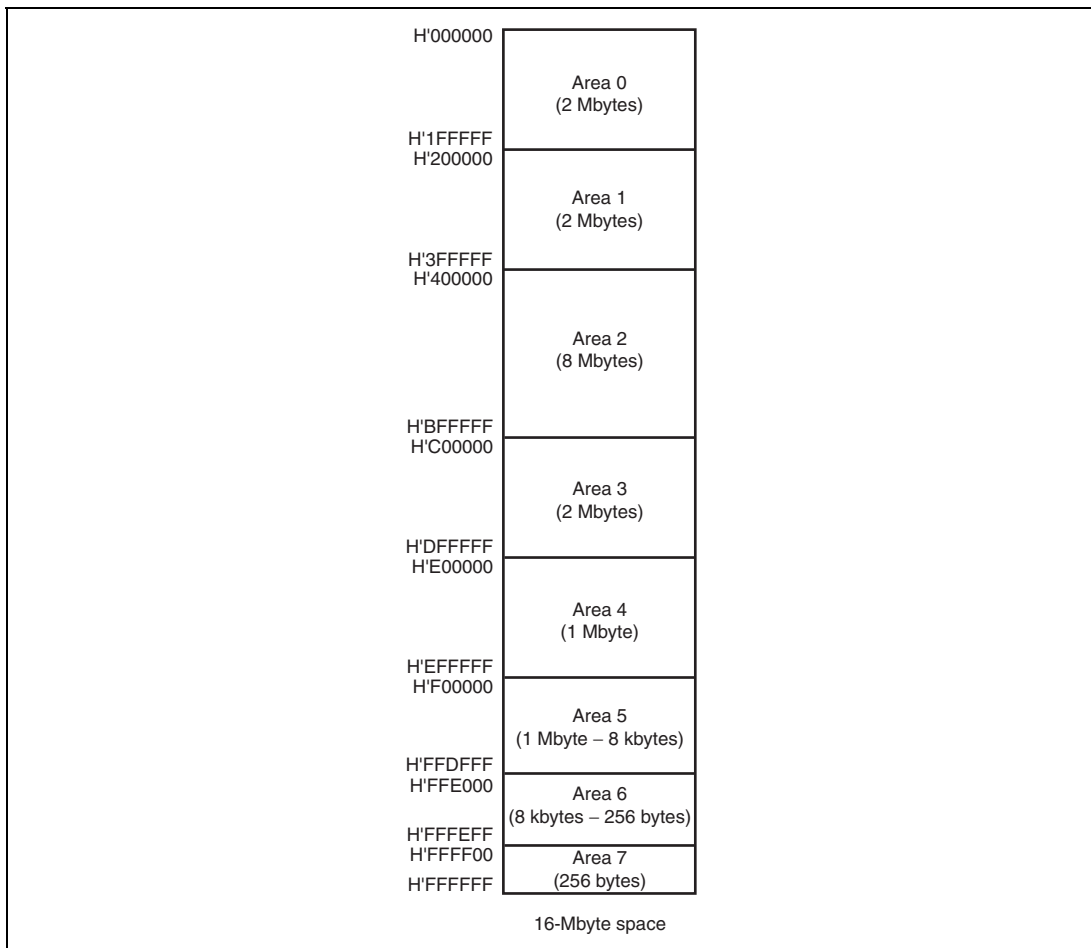
Figure 6.6   Address Space Area Division

6.5.3   External Bus Interface

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

(1)   Type of External Bus Interface

One type of external bus interfaces is provided and can be selected in area units. Table 6.4 shows each interface name, description, area name to be set for each interface. Table 6.5 shows the areas that can be specified for each interface. The initial state of each area is a basic bus interface.

Table 6.4   Interface Names and Area Names

Interface	Description	Area Name
Basic bus interface	Directly connected to ROM and SRAM	Basic bus space

Table 6.5   Areas Specifiable for Each Interface

Interface	Related Registers	Areas							
		0	1	2	3	4	5	6	7
Basic bus interface	—	O	O	O	O	O	O	O	O

(2)   Bus Width

A bus width of 8 or 16 bits can be selected with ABWCR. An area for which an 8-bit bus is selected functions as an 8-bit access space and an area for which a 16-bit bus is selected functions as a 16-bit access space.

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

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

(3)   Endian Format

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

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

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

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

(4)   Number of Access Cycles

1.   Basic Bus Interface

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

For the 2-state access space, a wait cycle insertion is disabled. For the 3-state access space, a program wait (0 to 7 cycles) specified by WTCRA and WTCRB can be inserted.

Number of access cycles in the basic bus interface
= number of basic cycles (2, 3) + number of program wait cycles (0 to 7)

Table 6.6 lists the number of access cycles for each interface.

Table 6.6   Number of Access Cycles

Basic bus interface	=	Th	+T1	+T2			+Tt	
		[0,1]	[1]	[1]			[0,1]	[2 to 4]
	=	Th	+T1	+T2	+Tpw	+T3	+Tt	
		[0,1]	[1]	[1]	[0 to 7]	[1]	[0,1]	[3 to 12]

Legend:
Numbers: Number of access cycles

RENESAS

(5)   Strobe Assert/Negate Timings

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

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

6.5.4   Area and External Bus Interface

(1)   Area 0

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

Note: Applied to the LSI version that incorporates the ROM.

(2)   Area 1

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

Note: Applied to the LSI version that incorporates the ROM.

(3)   Area 2

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

(4)   Area 3

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

(5)   Area 4

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

(6)   Area 5

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

(7)   Area 6

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

(8)   Area 7

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

6.5.5   Endian and Data Alignment

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

(1)   8-Bit Access Space

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

Figures 6.7 and 6.8 illustrate data alignment control for the 8-bit access space. Figure 6.7 shows the data alignment when the data endian format is specified as big endian. Figure 6.8 shows the data alignment when the data endian format is specified as little endian.

RENESAS

Data Size	Access Address	Access Count	Bus Cycle	Data Size	Data bus (D15 D8 / D7 D0)
Byte	n	1	1st	Byte	7–0
Word	n	2	1st	Byte	15–8
			2nd	Byte	7–0
Longword	n	4	1st	Byte	31–24
			2nd	Byte	23–16
			3rd	Byte	15–8
			4th	Byte	7–0

Figure 6.7   Access Sizes and Data Alignment Control for 8-Bit Access Space (Big Endian)

Data Size	Access Address	Access Count	Bus Cycle	Data Size	Data bus (D15 D8 / D7 D0)
Byte	n	1	1st	Byte	7–0
Word	n	2	1st	Byte	7–0
			2nd	Byte	15–8
Longword	n	4	1st	Byte	7–0
			2nd	Byte	15–8
			3rd	Byte	23–16
			4th	Byte	31–24

Figure 6.8   Access Sizes and Data Alignment Control for 8-Bit Access Space (Little Endian)

RENESAS

(2)   16-Bit Access Space

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

Figures 6.9 and 6.10 illustrate data alignment control for the 16-bit access space. Figure 6.9 shows the data alignment when the data endian format is specified as big endian. Figure 6.10 shows the data alignment when the data endian format is specified as little endian.

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

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

Figure 6.9   Access Sizes and Data Alignment Control for 16-Bit Access Space (Big Endian)

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

Figure 6.10   Access Sizes and Data Alignment Control for 16-Bit Access Space (Little Endian)

RENESAS

6.6   Basic Bus Interface

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

6.6.1   Data Bus

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and controls whether the upper byte data bus (D15 to D8) or lower byte data bus (D7 to D0) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space), the data size, and endian format when accessing external address space,. For details, see section 6.5.5, Endian and Data Alignment.

6.6.2   I/O Pins Used for Basic Bus Interface

Table 6.7 shows the pins used for basic bus interface.

Table 6.7   I/O Pins for Basic Bus Interface

Name	Symbol	I/O	Function
Address strobe	$\overline{AS}$	Output	Strobe signal indicating that an address output on the address bus is valid during access
Read strobe	$\overline{RD}$	Output	Strobe signal indicating the read access
Low-high write	$\overline{LHWR}$	Output	Strobe signal indicating that the upper byte (D15 to D8) is valid during write access
Low-low write	$\overline{LLWR}$	Output	Strobe signal indicating that the lower byte (D7 to D0) is valid during write access

RENESAS

6.6.3   Basic Timing

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

(1)   16-Bit 2-State Access Space

Figures 6.11 to 6.13 show the bus timing of 16-bit 2-state access space.

When accessing 16-bit access space, the upper byte data bus (D15 to D8) is used for even addresses access, and the lower byte data bus (D7 to D0) is used for odd addresses. No wait cycles can be inserted.
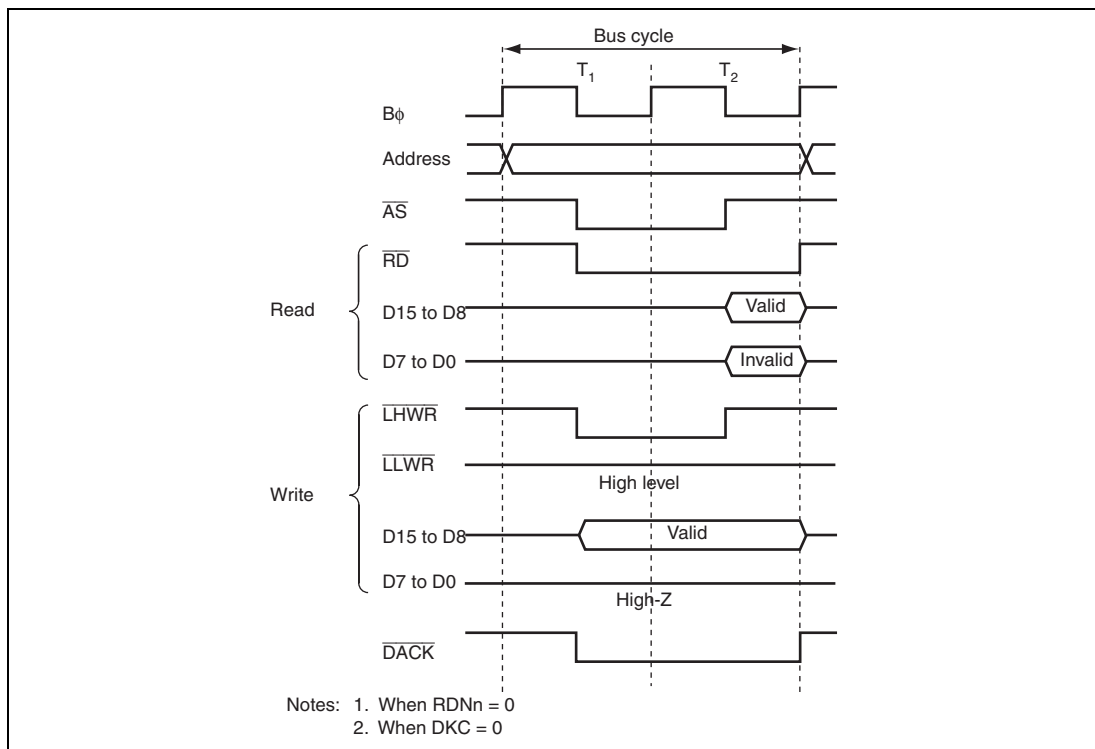
Figure 6.11   16-Bit 2-State Access Space Bus Timing (Byte Access for Even Address)

RENESAS
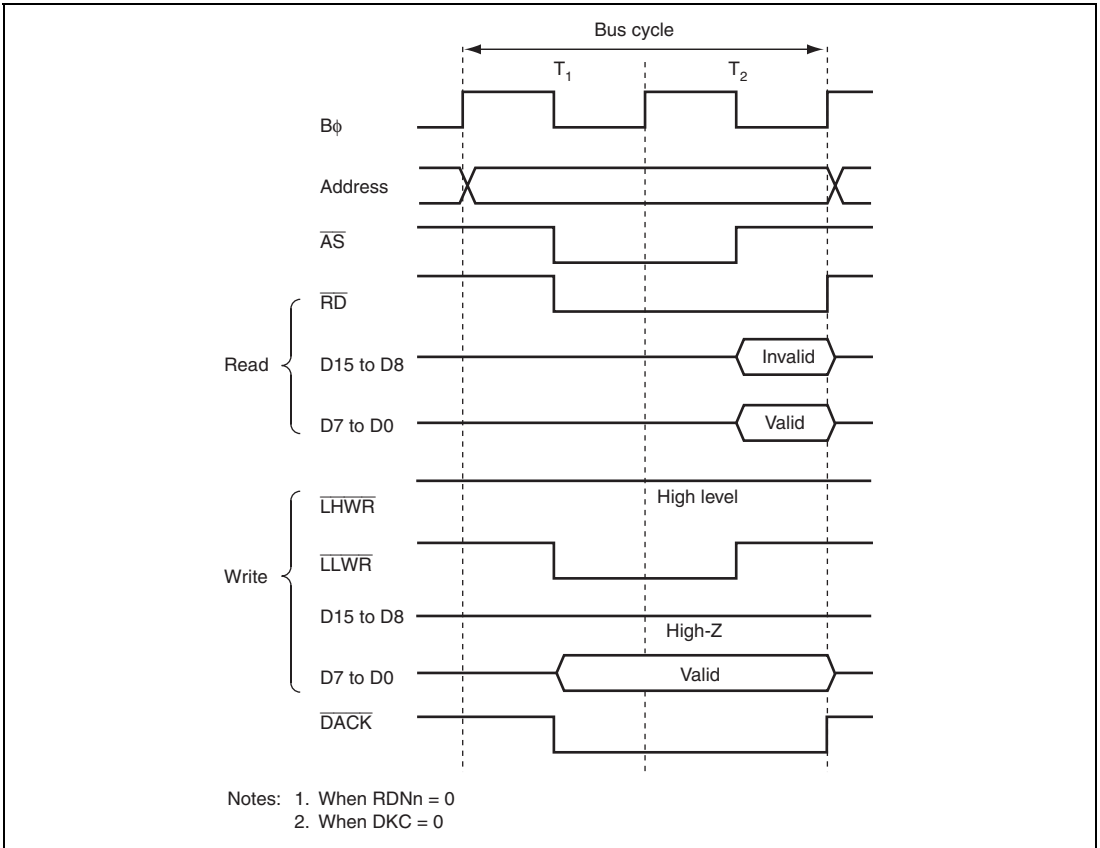
Figure 6.12   16-Bit 2-State Access Space Bus Timing (Byte Access for Odd Address)
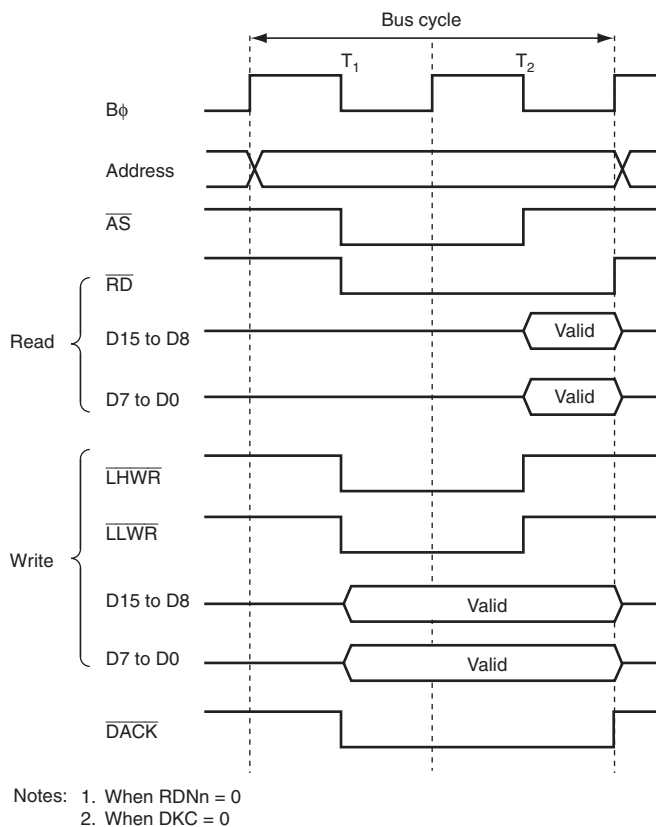
Figure 6.13   16-Bit 2-State Access Space Bus Timing
(Word Access for Even Address)

(2)   16-Bit 3-State Access Space

Figures 6.14 to 6.16 show the bus timing of 16-bit 3-state access space.

When accessing 16-bit access space, the upper byte data bus (D15 to D8) is used for even addresses, and the lower byte data bus (D7 to D0) is used for odd addresses. Wait cycles can be inserted.
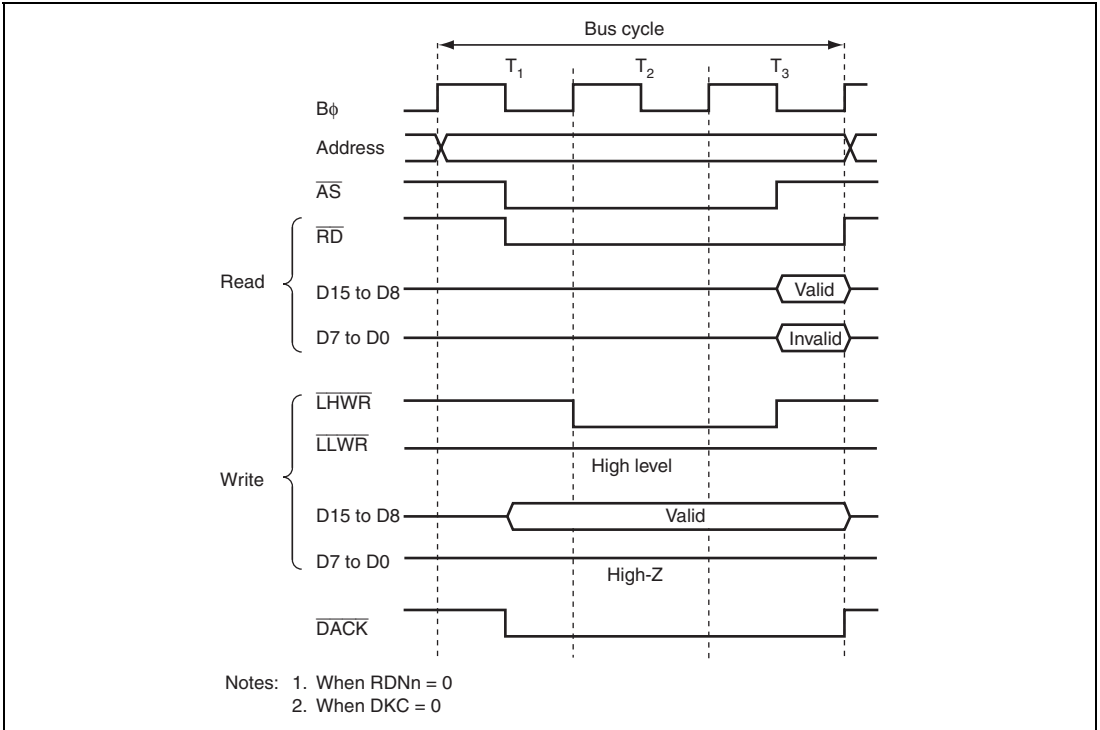
**Figure 6.14   16-Bit 3-State Access Space Bus Timing
(Byte Access for Even Address)**
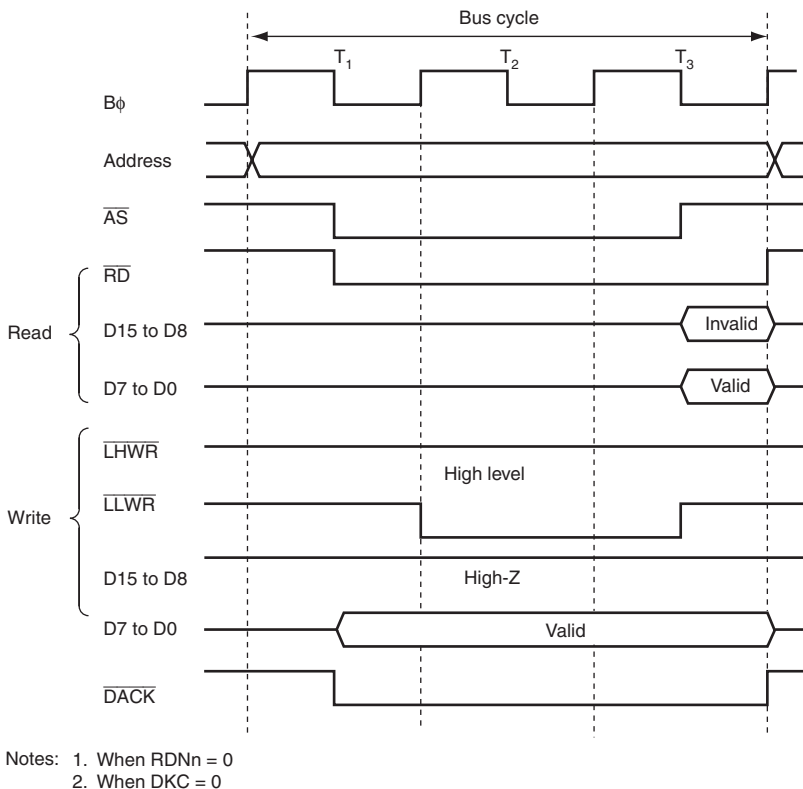
Figure 6.15   16-Bit 3-State Access Space Bus Timing (Word Access for Odd Address)
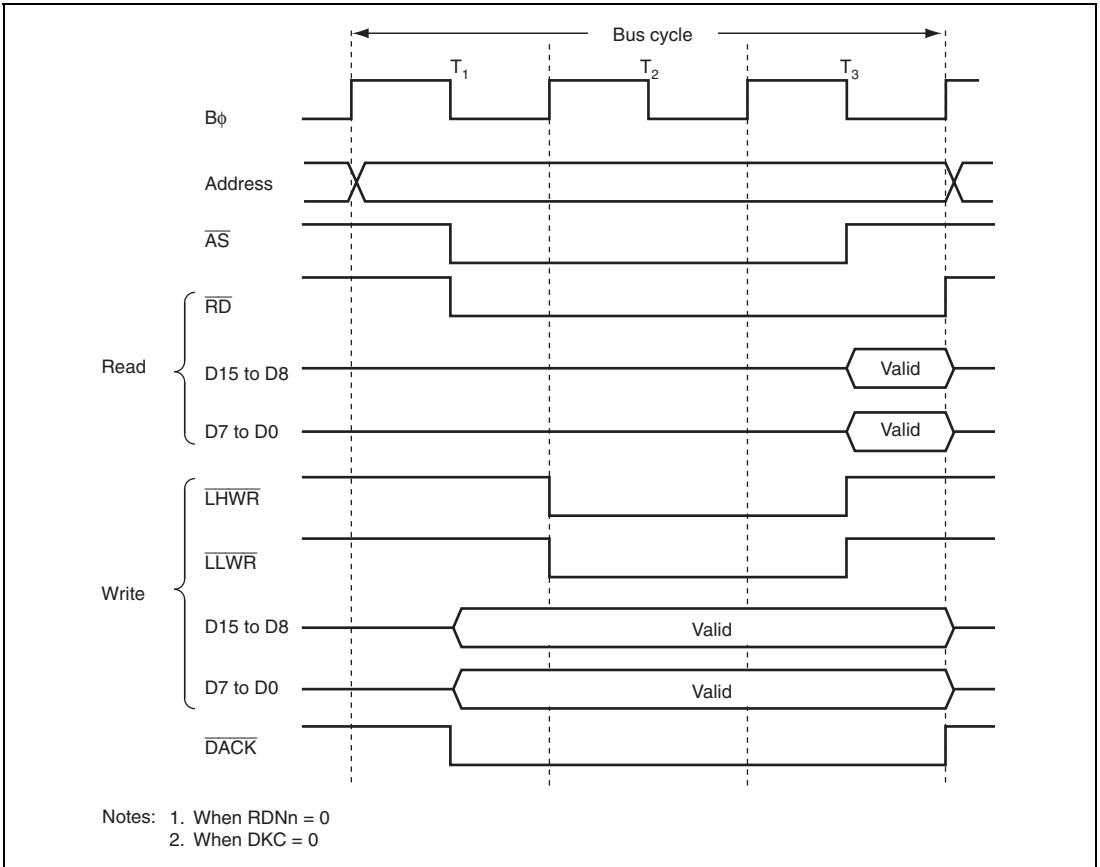
**Figure 6.16   16-Bit 3-State Access Space Bus Timing
(Word Access for Even Address)**

6.6.4   Wait Control

This LSI can extend the bus cycle by inserting wait cycles (Tw) when the external address space is accessed. There is one way of inserting wait cycle: program wait (Tpw) insertion.

(1)   Program Wait Insertion

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

Figure 6.17 shows an example of wait cycle insertion timing. After a reset, the 3-state access is specified and the program wait is inserted for seven cycles.
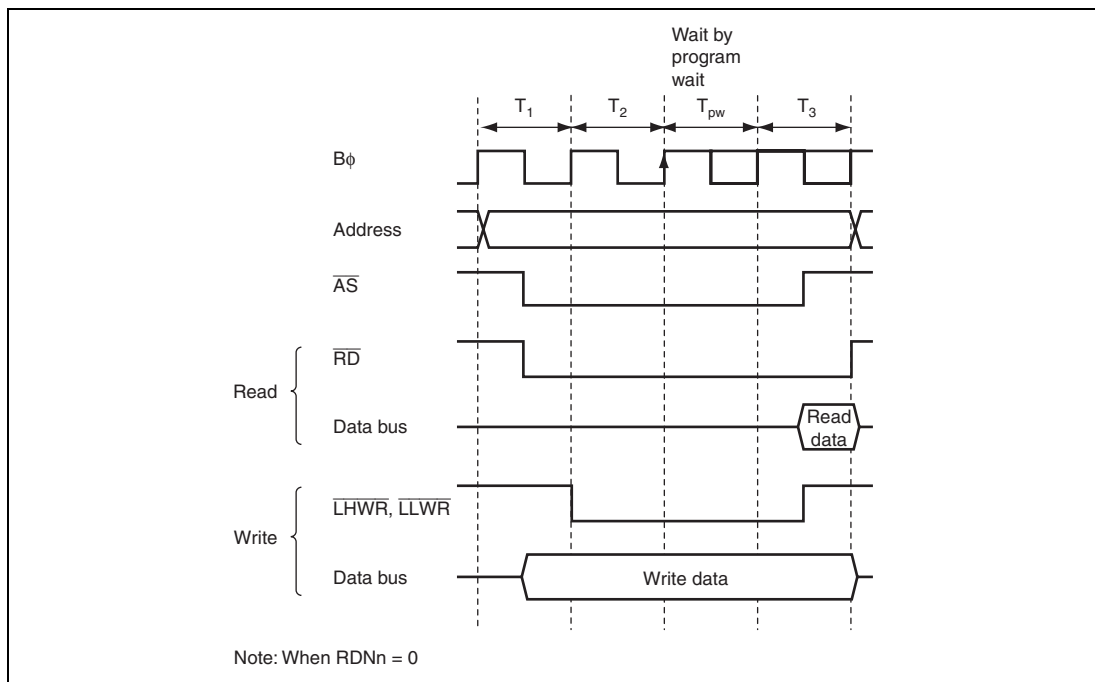
Figure 6.17   Example of Wait Cycle Insertion Timing

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

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

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

Figure 6.18 shows an example of timing when the read strobe timing is changed in the basic bus 3-state access space.
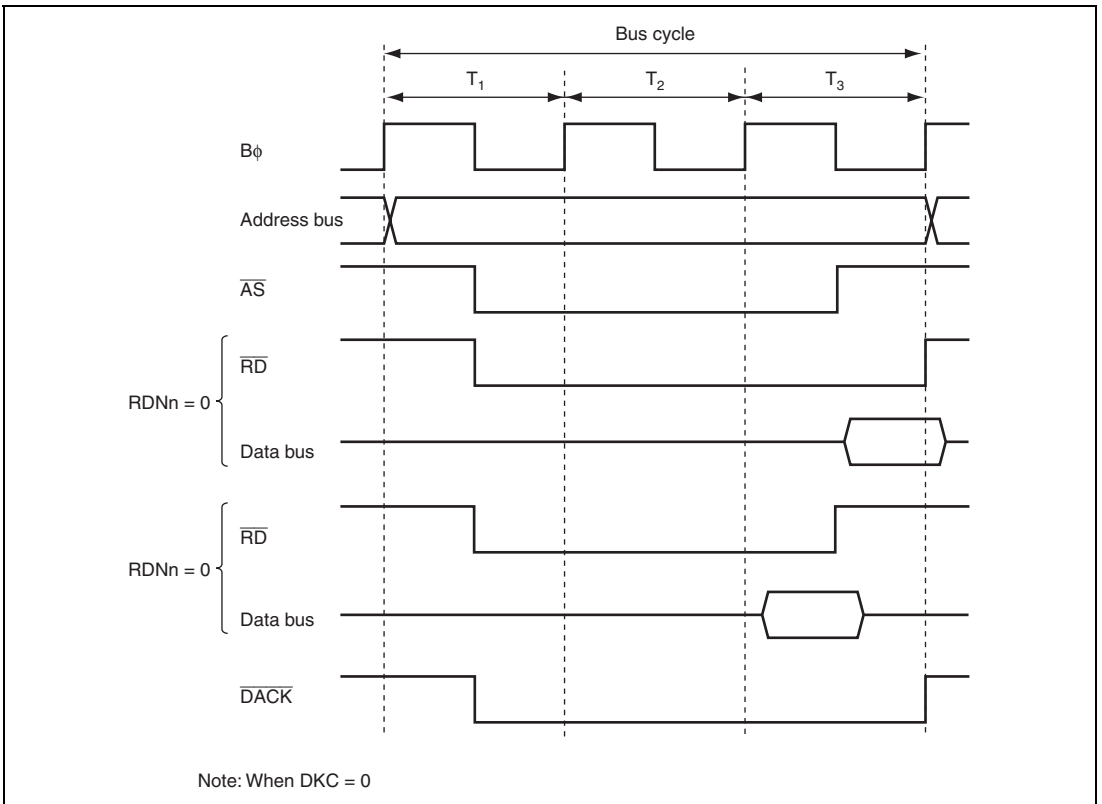
Figure 6.18   Example of Read Strobe Timing

RENESAS

6.6.6 $\overline{\text{DACK}}$ Signal Output Timing

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

Figure 6.19 shows the $\overline{\text{DACK}}$ signal output timing. Setting the DKC bit to 1 asserts the $\overline{\text{DACK}}$ signal a half cycle earlier.
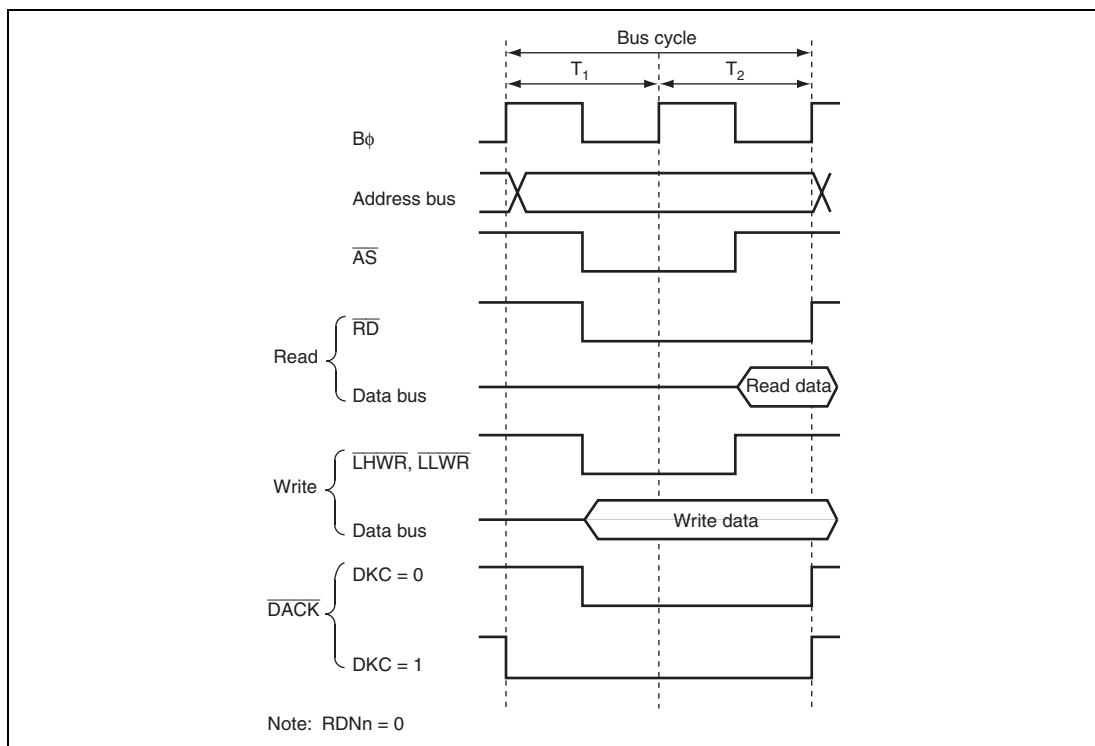
Figure 6.19 $\overline{\text{DACK}}$ Signal Output Timing

6.7   Idle Cycle

In this LSI, idle cycles can be inserted between the consecutive external accesses. By inserting the idle cycle, data conflicts between ROM read cycle whose output floating time is long and an access cycle from/to high-speed memory or I/O interface can be prevented.

6.7.1   Operation

When this LSI consecutively accesses external address space, it can insert an idle cycle between bus cycles in the following four cases. These conditions are determined by the sequence of read and write and previously accessed area.

1. When read cycles of different areas in the external address space occur consecutively
2. When an external write cycle occurs immediately after an external read cycle
3. When an external read cycle occurs immediately after an external write cycle
4. When an external access occurs immediately after a DMAC single address transfer (write cycle)

Up to four idle cycles can be inserted under the conditions shown above. The number of idle cycles to be inserted should be specified to prevent data conflicts between the output data from a previously accessed device and data from a subsequently accessed device.

Under conditions 1 and 2, which are the conditions to insert idle cycles after read, the number of idle cycles can be selected from setting A specified by bits IDLCA1 and IDLCA0 in IDLCR or setting B specified by bits IDLCB1 and IDLCB0 in IDLCR: Setting A can be selected from one to four cycles, and setting B can be selected from one or two to four cycles. Setting A or B can be specified for each area by setting bits IDLSEL7 to IDLSEL0 in IDLCR. Note that bits IDLSEL7 to IDLSEL0 correspond to the previously accessed area of the consecutive accesses.

The number of idle cycles to be inserted under conditions 3 and 4, which are conditions to insert idle cycles after write, can be determined by setting A as described above.

After the reset release, IDLCR is initialized to four idle cycle insertion under all conditions 1 to 4 shown above.

Table 6.8 shows the correspondence between conditions 1 to 4 and number of idle cycles to be inserted for each area. Table 6.9 shows the correspondence between the number of idle cycles to be inserted specified by settings A and B, and number of cycles to be inserted.

RENESAS

Table 6.8   Number of Idle Cycle Insertion Selection in Each Area

Insertion Condition	n	IDLSn Setting	IDLSELn n = 0 to 7	0	1	2	3	4	5	6	7
Consecutive reads in different areas	1	0	—				Invalid				
		1	0	A	A	A	A	A	A	A	A
			1	B	B	B	B	B	B	B	B
Write after read	0	0	—				Invalid				
		1	0	A	A	A	A	A	A	A	A
			1	B	B	B	B	B	B	B	B
Read after write	2	0	—				Invalid				
		1		A	A	A	A	A	A	A	A
External access after single address transfer	3	0	—				Invalid				
		1		A	A	A	A	A	A	A	A

The header above spans: **Bit Settings** over IDLSn and IDLSELn; **Area for Previous Access** over columns 0–7.

Legend:

A: Number of idle cycle insertion A is selected.

B: Number of idle cycle insertion B is selected.

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

Table 6.9   Number of Idle Cycle Insertions

A IDLCA1	A IDLCA0	B IDLCB1	B IDLCB0	Number of Cycles
—	—	0	0	0
0	0	—	—	1
0	1	0	1	2
1	0	1	0	3
1	1	1	1	4

(1)   Consecutive Reads in Different Areas

If consecutive reads in different areas occur while bit IDLS1 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 when bit IDLSELn in IDLCR is cleared to 0, or bits IDLCB1 and IDLCB0 when bit IDLSELn is set to 1 are inserted at the start of the second read cycle (n = 0 to 7).

Figure 6.20 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a read cycle for SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data conflict is prevented.
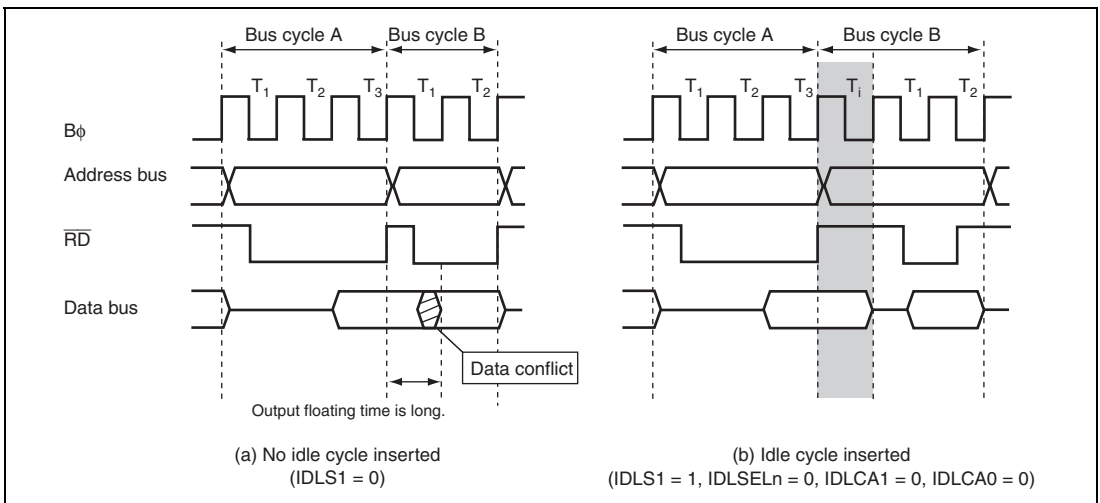
Figure 6.20   Example of Idle Cycle Operation (Consecutive Reads in Different Areas)

(2)   Write after Read

If an external write occurs after an external read while bit IDLS0 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 when bit IDLSELn in IDLCR is cleared to 0 when IDLSELn = 0, or bits IDLCB1 and IDLCB0 when IDLSELn is set to 1 are inserted at the start of the write cycle (n = 0 to 7).

Figure 6.21 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data conflict is prevented.
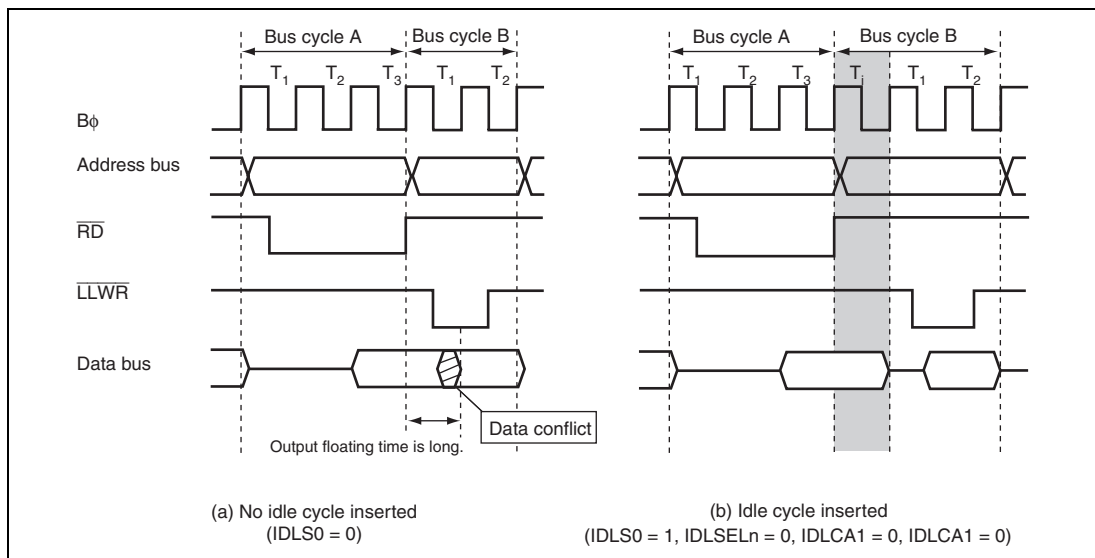
Figure 6.21   Example of Idle Cycle Operation (Write after Read)

(3)   Read after Write

If an external read occurs after an external write while bit IDLS2 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 are inserted at the start of the read cycle (n = 0 to 7).

Figure 6.22 shows an example of the operation in this case. In this example, bus cycle A is a CPU write cycle and bus cycle B is a read cycle from the SRAM. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the CPU write data and read data from an SRAM device. In (b), an idle cycle is inserted, and a data conflict is prevented.
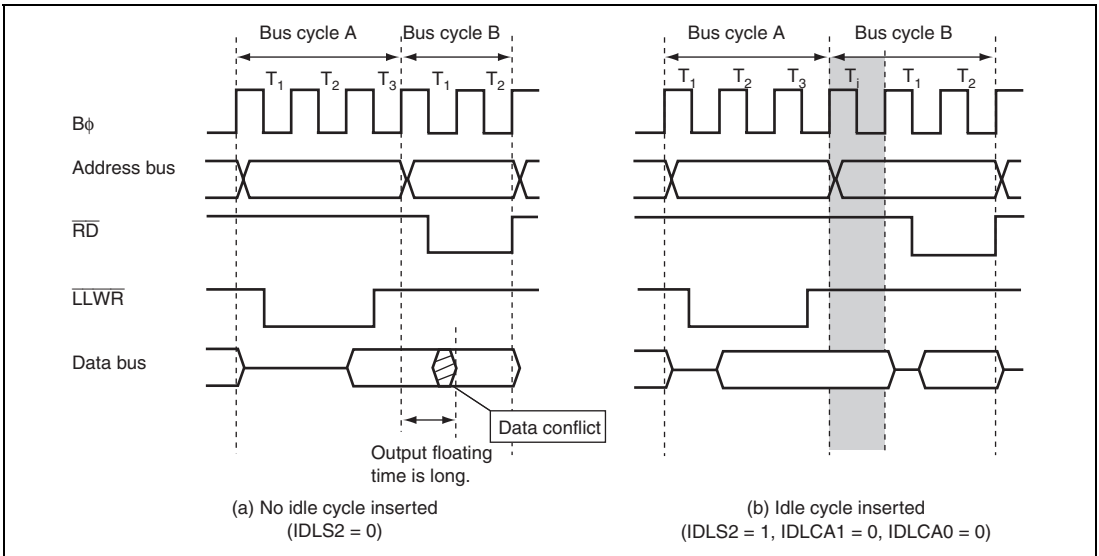
Figure 6.22   Example of Idle Cycle Operation (Read after Write)

(4)   External Access after Single Address Transfer Write

If an external access occurs after a single address transfer write while bit IDLS3 in IDLCR is set to 1, idle cycles specified by bits IDLCA1 and IDLCA0 are inserted at the start of the external access (n = 0 to 7).

Figure 6.23 shows an example of the operation in this case. In this example, bus cycle A is a single address transfer (write cycle) and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a conflict occurs in bus cycle B between the external device write data and this LSI write data. In (b), an idle cycle is inserted, and a data conflict is prevented.
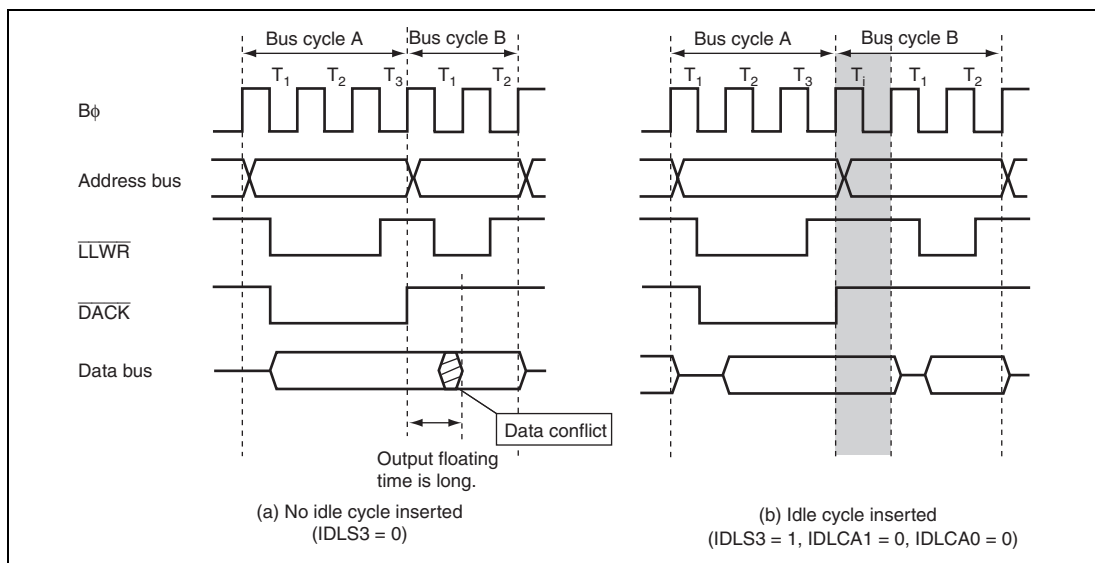
Figure 6.23   Example of Idle Cycle Operation (Write after Single Address Transfer Write)

(5)   External NOP Cycles and Idle Cycles

A cycle in which an external space is not accessed due to internal operations is called an external NOP cycle. Even when an external NOP cycle occurs between consecutive external bus cycles, an idle cycle can be inserted. In this case, the number of external NOP cycles is included in the number of idle cycles to be inserted.

Figure 6.24 shows an example of external NOP and idle cycle insertion.
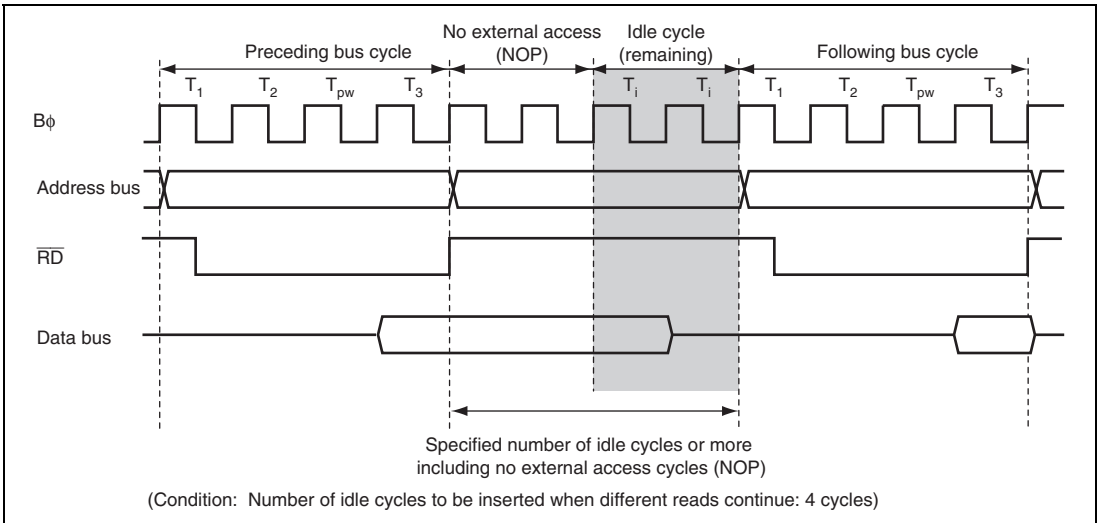
Figure 6.24   Idle Cycle Insertion Example

RENESAS

Table 6.10   Idle Cycles in Mixed Accesses to Normal Space

Previous Access	Next Access	IDLS				IDLSEL	IDLCA		IDLCB		Idle Cycle
		3	2	1	0	7 to 0	1	0	1	0	
Normal space read	Normal space read	—	—	0	—	—	—	—	—	—	Disabled
		—	—	1	—	0	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
						1	—	—	0	0	0 cycle inserted
									0	1	2 cycle inserted
									1	0	3 cycles inserted
									1	1	4 cycles inserted
Normal space read	Normal space write	—	—	—	0	—	—	—	—	—	Disabled
		—	—	—	1	0	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
						1	—	—	0	0	0 cycle inserted
									0	1	2 cycle inserted
									1	0	3 cycles inserted
									1	1	4 cycles inserted
Normal space write	Normal space read	—	0	—	—	—	—	—	—	—	Disabled
		—	1	—	—	—	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted
Single address transfer write	Normal space read	0	—	—	—	—	—	—	—	—	Disabled
		1	—	—	—	—	0	0	—	—	1 cycle inserted
							0	1			2 cycles inserted
							1	0			3 cycles inserted
							1	1			4 cycles inserted

RENESAS

6.7.2   Pin States in Idle Cycle

Table 6.11 shows the pin states in an idle cycle.

Table 6.11   Pin States in Idle Cycle

Pins	Pin State
A23 to A0	Contents of following bus cycle
D15 to D0	High impedance
$\overline{AS}$	High
$\overline{RD}$	High
$\overline{LHWR}$, $\overline{LLWR}$	High
$\overline{DACKn}$ (n = 3 to 0)	High

6.8   Internal Bus

6.8.1   Access to Internal Address Space

The internal address spaces of this LSI are the on-chip ROM space, on-chip RAM space, and register space for the on-chip peripheral modules. The number of cycles necessary for access differs according the space.

Table 6.12 shows the number of access cycles for each on-chip memory space.

Table 6.12   Number of Access Cycles for On-Chip Memory Spaces

Access Space	Access	Number of Access Cycles
On-chip ROM space	Read	One Iϕ cycle
	Write	Six Iϕ cycles
On-chip RAM space	Read	One Iϕ cycle
	Write	Two Iϕ cycle

RENESAS

In access to the registers for on-chip peripheral modules, the number of access cycles differs according to the register to be accessed. When the dividing ratio of the operating clock of a bus master and that of a peripheral module is 1 : n, synchronization cycles using a clock divided by 0 to n-1 are inserted for register access in the same way as for external bus clock division.

Table 6.13 lists the number of access cycles for registers of on-chip peripheral modules.

Table 6.13   Number of Access Cycles for Registers of On-Chip Peripheral Modules

| Module to be Accessed | Number of Cycles | | Write Data Buffer Function |
	Read	Write	
DMAC registers		2Iϕ	Disabled
MCU operating mode, clock pulse generator, power-down control registers, interrupt controller, and bus controller registers	2Iϕ	3Iϕ	Disabled
I/O port PFCR registers and WDT registers	2Pϕ	3Pϕ	Disabled
I/O port registers other than PFCR, TPU, IIC2, SCI, A/D, and D/A registers		2Pϕ	Enabled
RCANMON registers in RCAN-ET, SSU∗, WAT, SDG, motor control PWM, and 16-bit PWM registers		3Pϕ	Enabled
Registers other than RCANMON in RCAN-ET		4Pϕ	Enabled

Note: ∗   SSU: Synchronous Serial communication Unit

6.9 Write Data Buffer Function

6.9.1 Write Data Buffer Function for External Data Bus

This LSI has a write data buffer function for the external data bus. Using the write data buffer function enables internal accesses in parallel with external writes or DMAC single address transfers. The write data buffer function is made available by setting the WDBE bit to 1 in BCR1.

Figure 6.25 shows an example of the timing when the write data buffer function is used. When this function is used, if an external address space write or a DMAC single address transfer continues for two cycles or longer, and there is an internal access next, an external write only is executed in the first two cycles. However, from the next cycle onward, internal accesses (on-chip memory or internal I/O register read/write) and the external address space write rather than waiting until it ends are executed in parallel.
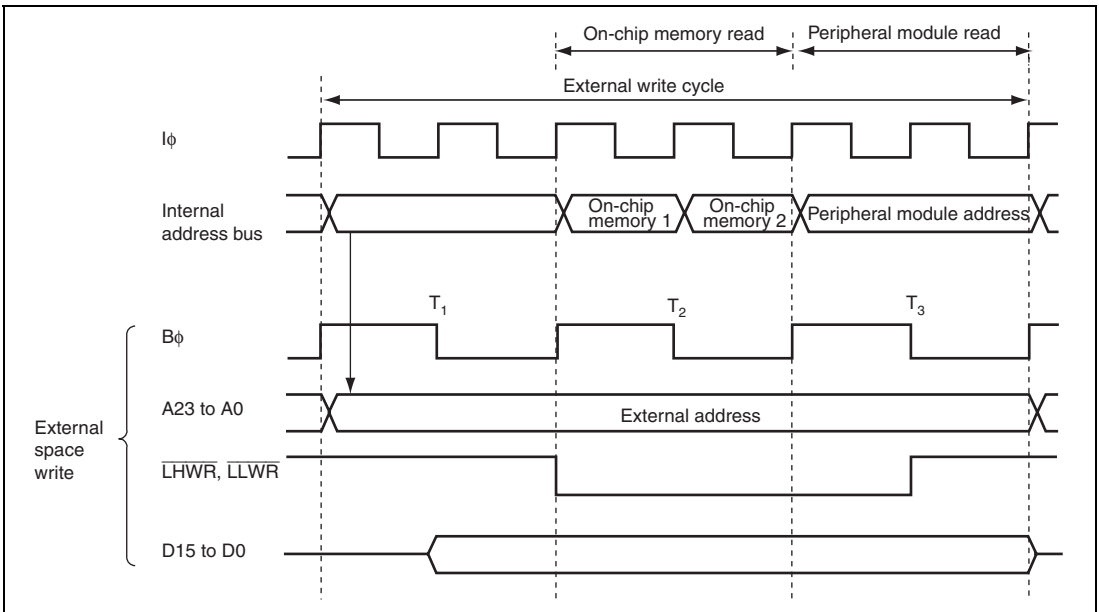
Figure 6.25   Example of Timing when Write Data Buffer Function Is Used

6.9.2   Write Data Buffer Function for Peripheral Modules

This LSI has a write data buffer function for the peripheral module access. Using the write data buffer function enables peripheral module writes and on-chip memory or external access to be executed in parallel. The write data buffer function is made available by setting the PWDBE bit in BCR2 to 1. For details on the on-chip peripheral module registers, see table 6.13, Number of Access Cycles for Registers of On-Chip Peripheral Modules in section 6.8, Internal Bus.

Figure 6.26 shows an example of the timing when the write data buffer function is used. When this function is used, if an internal I/O register write continues for two cycles or longer and then there is an on-chip RAM, an on-chip ROM, or an external access, internal I/O register write only is performed in the first two cycles. However, from the next cycle onward an internal memory or an external access and internal I/O register write are executed in parallel rather than waiting until it ends.
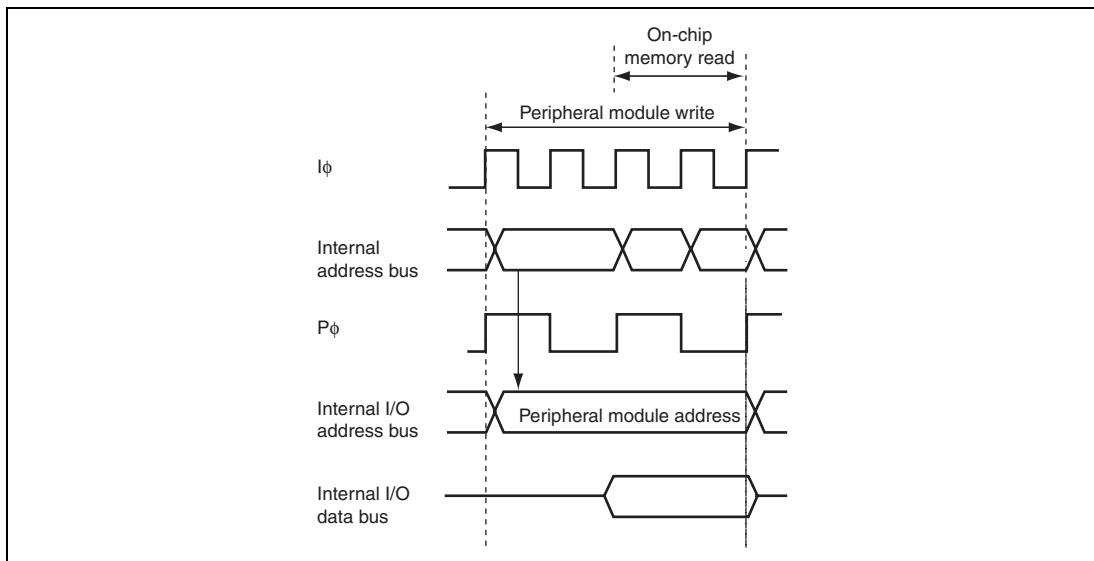
**Figure 6.26   Example of Timing when Peripheral Module
Write Data Buffer Function Is Used**

6.10   Bus Arbitration

This LSI has bus arbiters that arbitrate bus mastership operations (bus arbitration). This LSI incorporates internal access and external access bus arbiters that can be used and controlled independently. The internal bus arbiter handles the CPU and DMAC accesses.

The bus arbiters determine priorities at the prescribed timing, and permit use of the bus by means of the bus request acknowledge signal.

6.10.1   Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The priority of the internal bus arbitration:

> (High) DMAC > CPU (Low)

If the DMAC access continue, the CPU can be given priority over the DMAC to execute the bus cycles alternatively between them by setting the IBCCS bit in BCR2. In this case, the priority of the DMAC does not change.

6.10.2   Bus Handover Timing

Even if a bus request is received from a bus master with a higher priority over that of the bus master that has taken control of the bus and is currently operating, the bus is not necessarily handed over immediately. There are specific timings at which each bus master can release the bus.

(1)   CPU

The CPU is the lowest-priority bus master, and if a bus request is received from the DMAC, the bus arbiter grants the bus to the bus master that issued the request.

The timing of bus handover is at the end of the bus cycle. In sleep mode, the bus is handed over synchronously with the clock.

RENESAS

Note, however, that bus handover is prohibited in the following cases.

- The word or longword access is performed in some divisions.
- Stack handling is performed in multiple bus cycles.
- Transfer data read or write by memory transfer instructions, block transfer instructions, or TAS instruction.

  (In the block transfer instructions, the bus can be handed over between the write cycle and the following transfer data read cycle.)
- From the target read to write in the bit manipulation instructions or memory operation instructions.

  (In an instruction that performs no write operation according to the instruction condition, up to a cycle corresponding the write cycle)

(2)   DMAC

The DMAC sends the internal bus arbiter a request for the bus when an activation request is generated. When the DMAC accesses an external bus space, the DMAC first takes control of the bus from the internal bus arbiter and then requests a bus to the external bus arbiter.

After the DMAC takes control of the bus, it may continue the transfer processing cycles or release the bus at the end of every bus cycle depending on the conditions.

The DMAC continues transfers without releasing the bus in the following case:

- Between the read cycle in the dual-address mode and the write cycle corresponding to the read cycle

If no bus master of a higher priority than the DMAC requests the bus and the IBCSS bit in BCR2 is cleared to 0, the DMAC continues transfers without releasing the bus in the following cases:

- During transfer of one block in the block transfer mode

In other cases, the DMAC releases the bus at the end of the bus cycle.

6.11   Bus Controller Operation in Reset

In a reset, this LSI, including the bus controller, enters the reset state immediately, and any executing bus cycle is aborted.

6.12   Usage Notes

(1)   Setting Registers

The BSC registers must be specified before accessing the external address space. In on-chip ROM disabled mode, the BSC registers must be specified before accessing the external address space for other than an instruction fetch access.

RENESAS

Section 7   DMA Controller (DMAC)

This LSI includes a 4-channel DMA controller (DMAC).

7.1   Features

- Maximum of 4-G byte address space can be accessed
- Byte, word, or longword can be set as data transfer unit
- Maximum of 4-G bytes (4,294,967,295 bytes) can be set as total transfer size
  Supports free-running mode in which total transfer size setting is not needed
- DMAC activation methods are auto-request, on-chip module interrupt, and external request.
  Auto request: CPU activates (cycle stealing or burst access can be selected)
  On-chip module interrupt: Interrupt requests from on-chip peripheral modules can be selected as an activation source
  External request: Low level or falling edge detection of the $\overline{DREQ}$ signal can be selected
  External request is available for all four channels
  (In block transfer mode only low-level detection can be selected.)
- Dual or single address mode can be selected as address mode
  Dual address mode: Both source and destination are specified by addresses
  Single address mode: Either source or destination is specified by the $\overline{DREQ}$ signal and the other is specified by address
- Normal, repeat, or block transfer can be selected as transfer mode
  Normal transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  Repeat transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  Repeat size of data is transferred and then a transfer address returns to the transfer start address
  Up to 65,536 bytes/words/longwords can be set as repeat size
  Block transfer mode: One block data is transferred at a single transfer request
  Up to 65,536 bytes/words/longwords can be set as block size
- Extended repeat area function which repeats the addressees within a specified area using the transfer address with the fixed upper bits (ring buffer transfer can be performed, as an example) is available
  One bit (two bytes) to 27 bits (128 Mbytes) for transfer source and destination can be set as extended repeat areas

RENESAS

- Address update can be selected from fixed address, offset addition, and increment or decrement by 1, 2, or 4

  Address update by offset addition enables to transfer data at addresses which are not placed continuously

- Word or longword data can be transferred to an address which is not aligned with the respective boundary

  Data is divided according to its address (byte or word) when it is transferred

- Two types of interrupts can be requested to the CPU

  A transfer end interrupt is generated after the number of data specified by the transfer counter is transferred. A transfer escape end interrupt is generated when the remaining total transfer size is less than the transfer data size at a single transfer request, when the repeat size of data transfer is completed, or when the extended repeat area overflows.

A block diagram of the DMAC is shown in figure 7.1.
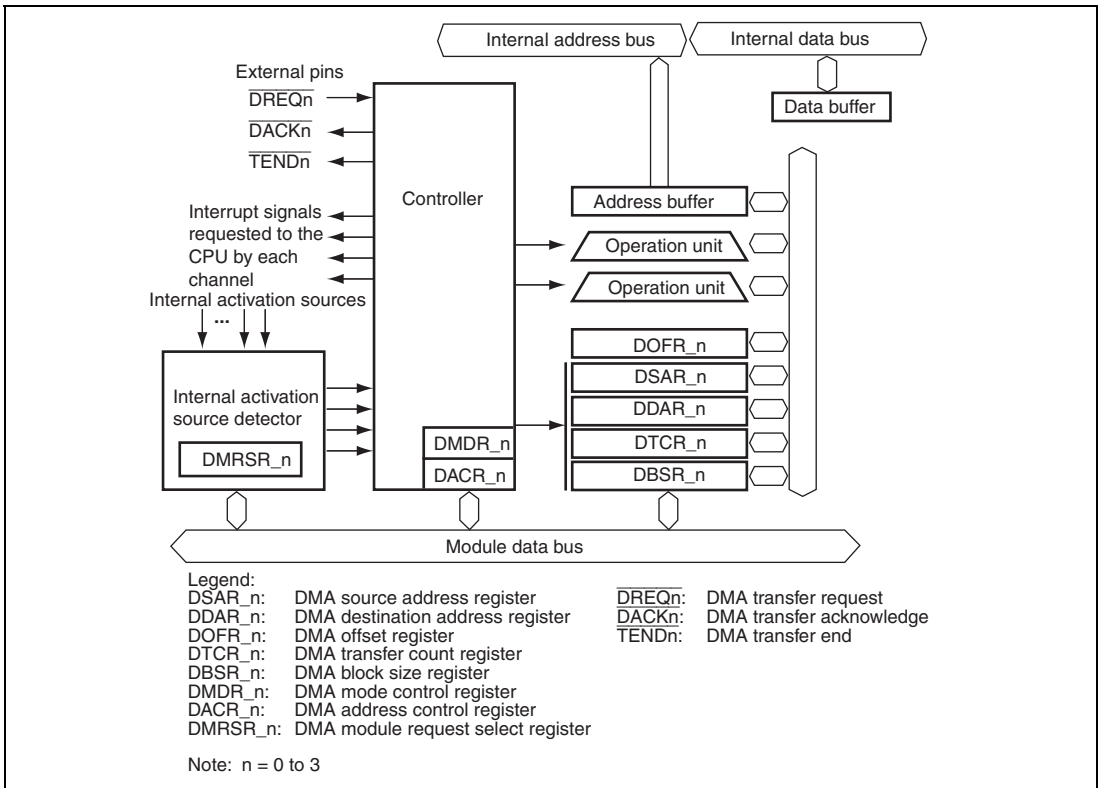
Figure 7.1   Block Diagram of DMAC

RENESAS

7.2   Input/Output Pins

Table 7.1 shows the pin configuration of the DMAC.

Table 7.1   Pin Configuration

Channel	Name	Abbreviation	I/O	Function
0	DMA transfer request 0	$\overline{\text{DREQ0}}$_A	Input	Channel 0 external request
	DMA transfer acknowledge 0	$\overline{\text{DACK0}}$_A	Output	Channel 0 single address transfer acknowledge
	DMA transfer end 0	$\overline{\text{TEND0}}$_A	Output	Channel 0 transfer end
1	DMA transfer request 1	$\overline{\text{DREQ1}}$_A	Input	Channel 1 external request
	DMA transfer acknowledge 1	$\overline{\text{DACK1}}$_A	Output	Channel 1 single address transfer acknowledge
	DMA transfer end 1	$\overline{\text{TEND1}}$_A	Output	Channel 1 transfer end
2	DMA transfer request 2	$\overline{\text{DREQ2}}$_A	Input	Channel 2 external request
	DMA transfer acknowledge 2	$\overline{\text{DACK2}}$_A	Output	Channel 2 single address transfer acknowledge
	DMA transfer end 2	$\overline{\text{TEND2}}$_A	Output	Channel 2 transfer end
3	DMA transfer request 3	$\overline{\text{DREQ3}}$_A	Input	Channel 3 external request
	DMA transfer acknowledge 3	$\overline{\text{DACK3}}$_A	Output	Channel 3 single address transfer acknowledge
	DMA transfer end 3	$\overline{\text{TEND3}}$_A	Output	Channel 3 transfer end

7.3   Register Descriptions

The DMAC has the following registers.

Channel 0

- DMA source address register_0 (DSAR_0)
- DMA destination address register_0 (DDAR_0)
- DMA offset register_0 (DOFR_0)
- DMA transfer count register_0 (DTCR_0)
- DMA block size register_0 (DBSR_0)
- DMA mode control register_0 (DMDR_0)
- DMA address control register_0 (DACR_0)
- DMA module request select register_0 (DMRSR_0)

Channel 1

- DMA source address register_1 (DSAR_1)
- DMA destination address register_1 (DDAR_1)
- DMA offset register_1 (DOFR_1)
- DMA transfer count register_1 (DTCR_1)
- DMA block size register_1 (DBSR_1)
- DMA mode control register_1 (DMDR_1)
- DMA address control register_1 (DACR_1)
- DMA module request select register_1 (DMRSR_1)

Channel 2

- DMA source address register_2 (DSAR_2)
- DMA destination address register_2 (DDAR_2)
- DMA offset register_2 (DOFR_2)
- DMA transfer count register_2 (DTCR_2)
- DMA block size register_2 (DBSR_2)
- DMA mode control register_2 (DMDR_2)
- DMA address control register_2 (DACR_2)
- DMA module request select register_2 (DMRSR_2)

RENESAS

Channel 3

- DMA source address register_3 (DSAR_3)
- DMA destination address register_3 (DDAR_3)
- DMA offset register_3 (DOFR_3)
- DMA transfer count register_3 (DTCR_3)
- DMA block size register_3 (DBSR_3)
- DMA mode control register_3 (DMDR_3)
- DMA address control register_3 (DACR_3)
- DMA module request select register_3 (DMRSR_3)

7.3.1   DMA Source Address Register (DSAR)

DSAR is a 32-bit readable/writable register that specifies the transfer source address. DSAR updates the transfer source address every time data is transferred. When DDAR is specified as the destination address (the DIRS bit in DACR is 1) in single address mode, DSAR is ignored.

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

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

7.3.2 DMA Destination Address Register (DDAR)

DDAR is a 32-bit readable/writable register that specifies the transfer destination address. DDAR updates the transfer destination address every time data is transferred. When DSAR is specified as the source address (the DIRS bit in DACR is 0) in single address mode, DDAR is ignored.

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

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

7.3.3   DMA Offset Register (DOFR)

DOFR is a 32-bit readable/writable register that specifies the offset to update the source and destination addresses. Although different values are specified for individual channels, the same values must be specified for the source and destination sides of a single channel.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

7.3.4   DMA Transfer Count Register (DTCR)

DTCR is a 32-bit readable/writable register that specifies the size of data to be transferred (total transfer size).

To transfer 1-byte data in total, set H'00000001 in DTCR. When H'00000000 is set in this register, it means that the total transfer size is not specified and data is transferred with the transfer counter stopped (free running mode). When H'FFFFFFFF is set, the total transfer size is 4 Gbytes (4,294,967,295), which is the maximum size. While data is being transferred, this register indicates the remaining transfer size. The value corresponding to its data access size is subtracted every time data is transferred (byte: −1, word: −2, and longword: −4).

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

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

7.3.5   DMA Block Size Register (DBSR)

DBSR specifies the repeat size or block size. DBSR is enabled in repeat transfer mode and block transfer mode and is disabled in normal transfer mode.

Bit	31	30	29	28	27	26	25	24
Bit Name	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	BKSZH31 to BKSZH16	Undefined	R/W	Specify the repeat size or block size. When H'0001 is set, the repeat or block size is one byte, one word, or one longword. When H'0000 is set, it means the maximum value (refer to table 7.2). While the DMA is in operation, the setting is fixed.
15 to 0	BKSZ15 to BKSZ0	Undefined	R/W	Indicate the remaining repeat or block size while the DMA is in operation. The value is decremented by 1 every time data is transferred. When the remaining size becomes 0, the value of the BKSZH bits is loaded. Set the same value as the BKSZH bits.

RENESAS

Table 7.2   Data Access Size, Valid Bits, and Settable Size

Mode	Data Access Size	BKSZH Valid Bits	BKSZ Valid Bits	Settable Size (Byte)
Repeat transfer and block transfer	Byte	31 to 16	15 to 0	1 to 65,536
	Word			2 to 131,072
	Longword			4 to 262,144

7.3.6   DMA Mode Control Register (DMDR)

DMDR controls the DMAC operation.

- DMDR_0

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	ERRF	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/(W)*	R	R/(W)*	R/(W)*

Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

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

RENESAS

- DMDR_1 to DMDR_3

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	—	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R/(W)*	R/(W)*

Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
31	DTE	0	R/W	Data Transfer Enable
				Enables/disables a data transfer for the corresponding channel. When this bit is set to 1, it indicates that the DMAC is in operation.
				Setting this bit to 1 starts a transfer when the auto-request is selected. When the on-chip module interrupt or external request is selected, a transfer request after setting this bit to 1 starts the transfer. While data is being transferred, clearing this bit to 0 stops the transfer.
				In block transfer mode, if writing 0 to this bit while data is being transferred, this bit is cleared to 0 after the current 1-block size data transfer.
				If an event which stops (sustains) a transfer occurs externally, this bit is automatically cleared to 0 to stop the transfer.
				Operating modes and transfer methods must not be changed while this bit is set to 1.
				0: Disables a data transfer
				1: Enables a data transfer (DMA is in operation)
				[Clearing conditions]
				• When the specified total transfer size of transfers is completed
				• When a transfer is stopped by an overflow interrupt by a repeat size end
				• When a transfer is stopped by an overflow interrupt by an extended repeat size end
				• When a transfer is stopped by a transfer size error interrupt
				• When clearing this bit to 0 to stop a transfer
				In block transfer mode, this bit changes after the current block transfer.
				• When an address error or an NMI interrupt is requested
				• In the reset state or hardware standby mode

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
30	DACKE	0	R/W	$\overline{\text{DACK}}$ Signal Output Enable
				Enables/disables the $\overline{\text{DACK}}$ signal output in single address mode. This bit is ignored in dual address mode.
				0: Enables $\overline{\text{DACK}}$ signal output
				1: Disables $\overline{\text{DACK}}$ signal output
29	TENDE	0	R/W	$\overline{\text{TEND}}$ Signal Output Enable
				Enables/disables the $\overline{\text{TEND}}$ signal output.
				0: Enables $\overline{\text{TEND}}$ signal output
				1: Disables $\overline{\text{TEND}}$ signal output
28	—	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.
27	DREQS	0	R/W	$\overline{\text{DREQ}}$ Select
				Selects whether a low level or the falling edge of the $\overline{\text{DREQ}}$ signal used in external request mode is detected.
				When a block transfer is performed in external request mode, clear this bit to 0 to select the low level detection.
				0: Low level detection
				1: Falling edge detection (the first transfer after a transfer enabled is detected on a low level)
26	NRD	0	R/W	Next Request Delay
				Selects the accepting timing of the next transfer request.
				0: Starts accepting the next transfer request after completion of the current transfer
				1: Starts accepting the next transfer request one cycle after completion of the current transfer
25, 24	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.
23	ACT	0	R	Active State
				Indicates the operating state for the channel.
				0: Waiting for a transfer request or a transfer disabled state by clearing the DTE bit to 0
				1: Active state
22 to 20	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
19	ERRF	0	R/(W)*	System Error Flag
				Indicates that an address error or an NMI interrupt has been generated. This bit is available only in DMDR_0. Setting this bit to 1 prohibits writing to the DTE bit for all the channels. This bit is reserved in DMDR_1 to DMDR_3. It is always read as 0 and cannot be modified.
				0: An address error or an NMI interrupt has not been generated
				1: An address error or an NMI interrupt has been generated
				[Clearing condition]
				• When clearing to 0 after reading ERRF = 1
				[Setting condition]
				• When an address error or an NMI interrupt has been generated
				However, when an address error or an NMI interrupt has been generated in module stop mode, this bit is not set.
18	—	0	R	Reserved
				This is a read-only bit and cannot be modified.
17	ESIF	0	R/(W)*	Transfer Escape Interrupt Flag
				Indicates that a transfer escape end interrupt has been requested. A transfer escape end means that a transfer is terminated before the transfer counter reaches 0.
				0: A transfer escape end interrupt has not been requested
				1: A transfer escape end interrupt has been requested
				[Clearing conditions]
				• When setting the DTE bit to 1
				• When clearing to 0 before reading ESIF = 1
				[Setting conditions]
				• When a transfer size error interrupt is requested
				• When a repeat size end interrupt is requested
				• When a transfer end interrupt by an extended repeat area overflow is requested

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
16	DTIF	0	R/(W)*	Data Transfer Interrupt Flag
				Indicates that a transfer end interrupt by the transfer counter has been requested.
				0: A transfer end interrupt by the transfer counter has not been requested
				1: A transfer end interrupt by the transfer counter has been requested
				[Clearing conditions]
				• When setting the DTE bit to 1
				• When clearing to 0 after reading DTIF = 1
				[Setting condition]
				• When DTCR reaches 0 and the transfer is completed
15	DTSZ1	0	R/W	Data Access Size 1 and 0
14	DTSZ0	0	R/W	Select the data access size for a transfer.
				00: Byte size (eight bits)
				01: Word size (16 bits)
				10: Longword size (32 bits)
				11: Setting prohibited
13	MDS1	0	R/W	Transfer Mode Select 1 and 0
12	MDS0	0	R/W	Select the transfer mode.
				00: Normal transfer mode
				01: Block transfer mode
				10: Repeat transfer mode
				11: Setting prohibited

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
11	TSEIE	0	R/W	Transfer Size Error Interrupt Enable
				Enables/disables a transfer size error interrupt.
				When the next transfer is requested while this bit is set to 1 and the contents of the transfer counter is less than the size of data to be transferred at a single transfer request, the DTE bit is cleared to 0. At this time, the ESIF bit is set to 1 to indicate that a transfer size error interrupt has been requested.
				The sources of a transfer size error are as follows:
				• In normal or repeat transfer mode, the total transfer size set in DTCR is less than the data access size
				• In block transfer mode, the total transfer size set in DTCR is less than the block size
				0: Disables a transfer size error interrupt request
				1: Enables a transfer size error interrupt request
10	—	0	R	Reserved
				This is a read-only bit and cannot be modified.
9	ESIE	0	R/W	Transfer Escape Interrupt Enable
				Enables/disables a transfer escape end interrupt request. When the ESIF bit is set to 1 with this bit set to 1, a transfer escape end interrupt is requested to the CPU. The transfer end interrupt request is cleared by clearing this bit or the ESIF bit to 0.
				0: Disables a transfer escape end interrupt
				1: Enables a transfer escape end interrupt
8	DTIE	0	R/W	Data Transfer End Interrupt Enable
				Enables/disables a transfer end interrupt request by the transfer counter. When the DTIF bit is set to 1 with this bit set to 1, a transfer end interrupt is requested to the CPU. The transfer end interrupt request is cleared by clearing this bit or the DTIF bit to 0.
				0: Disables a transfer end interrupt
				1: Enables a transfer end interrupt

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
7	DTF1	0	R/W	Data Transfer Factor 1 and 0
6	DTF0	0	R/W	Select a DMAC activation source. When the on-chip peripheral module setting is selected, the interrupt source should be selected by DMRSR. When the external request setting is selected, the sampling method should be selected by the DREQS bit.
				00: Auto request (cycle stealing)
				01: Auto request (burst access)
				10: On-chip module interrupt
				11: External request
5	DTA	0	R/W	Data Transfer Acknowledge
				This bit is valid in DMA transfer by the on-chip module interrupt source. This bit enables or disables the clearing of the source flag selected by DMRSR.
				0: Disables the clearing of the source in DMA transfer. Since the on-chip module interrupt source is not cleared in DMA transfer, it should be cleared by the CPU.
				1: Enables the clearing of the source in DMA transfer. Since the on-chip module interrupt source is cleared in DMA transfer, it does not require an interrupt by the CPU.
4, 3	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
2	DMAP2	0	R/W	DMA Priority Level 2 to 0
1	DMAP1	0	R/W	Select the priority level of the DMAC. When the CPU
0	DMAP0	0	R/W	has priority over the DMAC, the DMAC masks a transfer request and waits for the timing when the CPU priority becomes lower than the DMAC priority. The priority levels can be set to the individual channels. This bit is valid when the CPUPCE bit in CPUPCR is set to 1.
				000: Priority level 0 (low)
				001: Priority level 1
				010: Priority level 2
				011: Priority level 3
				100: Priority level 4
				101: Priority level 5
				110: Priority level 6
				111: Priority level 7 (high)

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

RENESAS

7.3.7   DMA Address Control Register (DACR)

DACR specifies the operating mode and transfer method.

Bit	31	30	29	28	27	26	25	24
Bit Name	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	SAT1	SAT0	—	—	DAT1	DAT0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

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

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
26	RPTIE	0	R/W	Repeat Size End Interrupt Enable
				Enables/disables a repeat size end interrupt request.
				In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat-size data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested. Even when the repeat area is not specified (ARS1 = 1 and ARS0 = 0), a repeat size end interrupt after a 1-block data transfer can be requested.
				In addition, in block transfer mode, when the next transfer is requested after 1-block data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested.
				0: Disables a repeat size end interrupt
				1: Enables a repeat size end interrupt
25	ARS1	0	R/W	Area Select 1 and 0
24	ARS0	0	R/W	Specify the block area or repeat area in block or repeat transfer mode.
				00: Specify the block area or repeat area on the source address
				01: Specify the block area or repeat area on the destination address
				10: Do not specify the block area or repeat area
				11: Setting prohibited
23, 22	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.
21	SAT1	0	R/W	Source Address Update Mode 1 and 0
20	SAT0	0	R/W	Select the update method of the source address (DSAR). When DSAR is not specified as the transfer source in single address mode, this bit is ignored.
				00: Source address is fixed
				01: Source address is updated by adding the offset
				10: Source address is updated by adding 1, 2, or 4 according to the data access size
				11: Source address is updated by subtracting 1, 2, or 4 according to the data access size

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
19, 18	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.
17	DAT1	0	R/W	Destination Address Update Mode 1 and 0
16	DAT0	0	R/W	Select the update method of the destination address (DDAR). When DDAR is not specified as the transfer destination in single address mode, this bit is ignored.
				00: Destination address is fixed
				01: Destination address is updated by adding the offset
				10: Destination address is updated by adding 1, 2, or 4 according to the data access size
				11: Destination address is updated by subtracting 1, 2, or 4 according to the data access size
15	SARIE	0	R/W	Interrupt Enable for Source Address Extended Repeat Area Overflow
				Enables/disables an interrupt request for an extended repeat area overflow on the source address.
				When an extended repeat area overflow on the source address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the source address is requested.
				When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which a transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped.
				When the extended repeat area is not specified, this bit is ignored.
				0: Disables an interrupt request for an extended repeat area overflow on the source address
				1: Enables an interrupt request for an extended repeat area overflow on the source address
14, 13	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
12	SARA4	0	R/W	Source Address Extended Repeat Area
11	SARA3	0	R/W	Specify the extended repeat area on the source address
10	SARA2	0	R/W	(DSAR). With the extended repeat area, the specified
9	SARA1	0	R/W	lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is
8	SARA0	0	R/W	specified from four bytes to 128 Mbytes in units of byte and a power of 2.
				When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively.
				When an overflow in the extended repeat area occurs with the SARIE bit set to 1, an interrupt can be requested. Table 7.3 shows the settings and areas of the extended repeat area.
7	DARIE	0	R/W	Destination Address Extended Repeat Area Overflow Interrupt Enable
				Enables/disables an interrupt request for an extended repeat area overflow on the destination address.
				When an extended repeat area overflow on the destination address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the destination address is requested.
				When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which the transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped.
				When the extended repeat area is not specified, this bit is ignored.
				0: Disables an interrupt request for an extended repeat area overflow on the destination address
				1: Enables an interrupt request for an extended repeat area overflow on the destination address
6, 5	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
4	DARA4	0	R/W	Destination Address Extended Repeat Area
3	DARA3	0	R/W	Specify the extended repeat area on the destination
2	DARA2	0	R/W	address (DDAR). With the extended repeat area, the specified lower address bits are updated and the
1	DARA1	0	R/W	remaining upper address bits are fixed. The extended
0	DARA0	0	R/W	repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2.
				When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively.
				When an overflow in the extended repeat area occurs with the DARIE bit set to 1, an interrupt can be requested. Table 7.3 shows the settings and areas of the extended repeat area.

RENESAS

Table 7.3   Settings and Areas of Extended Repeat Area

SARA4 to SARA0 or DARA4 to DARA0	Extended Repeat Area
00000	Not specified
00001	2 bytes specified as extended repeat area by the lower 1 bit of the address
00010	4 bytes specified as extended repeat area by the lower 2 bits of the address
00011	8 bytes specified as extended repeat area by the lower 3 bits of the address
00100	16 bytes specified as extended repeat area by the lower 4 bits of the address
00101	32 bytes specified as extended repeat area by the lower 5 bits of the address
00110	64 bytes specified as extended repeat area by the lower 6 bits of the address
00111	128 bytes specified as extended repeat area by the lower 7 bits of the address
01000	256 bytes specified as extended repeat area by the lower 8 bits of the address
01001	512 bytes specified as extended repeat area by the lower 9 bits of the address
01010	1 kbyte specified as extended repeat area by the lower 10 bits of the address
01011	2 kbytes specified as extended repeat area by the lower 11 bits of the address
01100	4 kbytes specified as extended repeat area by the lower 12 bits of the address
01101	8 kbytes specified as extended repeat area by the lower 13 bits of the address
01110	16 kbytes specified as extended repeat area by the lower 14 bits of the address
01111	32 kbytes specified as extended repeat area by the lower 15 bits of the address
10000	64 kbytes specified as extended repeat area by the lower 16 bits of the address
10001	128 kbytes specified as extended repeat area by the lower 17 bits of the address
10010	256 kbytes specified as extended repeat area by the lower 18 bits of the address
10011	512 kbytes specified as extended repeat area by the lower 19 bits of the address
10100	1 Mbyte specified as extended repeat area by the lower 20 bits of the address
10101	2 Mbytes specified as extended repeat area by the lower 21 bits of the address
10110	4 Mbytes specified as extended repeat area by the lower 22 bits of the address
10111	8 Mbytes specified as extended repeat area by the lower 23 bits of the address
11000	16 Mbytes specified as extended repeat area by the lower 24 bits of the address
11001	32 Mbytes specified as extended repeat area by the lower 25 bits of the address
11010	64 Mbytes specified as extended repeat area by the lower 26 bits of the address
11011	128 Mbytes specified as extended repeat area by the lower 27 bits of the address
111××	Setting prohibited

Legend:

×: Don't care

RENESAS

7.3.8   DMA Module Request Select Register (DMRSR)

DMRSR is an 8-bit readable/writable register that specifies the on-chip module interrupt source. The vector number of the interrupt source is specified in eight bits. However, 0 is regarded as no interrupt source. For the vector numbers of the interrupt sources, refer to table 7.5.

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

7.4   Transfer Modes

Table 7.4 shows the DMAC transfer modes. The transfer modes can be specified to the individual channels.

Table 7.4   Transfer Modes

Address Mode	Transfer mode	Activation Source	Common Function	Address Register Source	Address Register Destination
Dual address	• Normal transfer • Repeat transfer • Block transfer Repeat or block size = 1 to 65,536 bytes, 1 to 65,536 words, or 1 to 65,536 longwords	• Auto request (activated by CPU) • On-chip module interrupt • External request	• Total transfer size: 1 to 4 Gbytes or not specified • Offset addition • Extended repeat area function	DSAR	DDAR
Single address	• Instead of specifying the source or destination address registers, data is directly transferred from/to the external device using the $\overline{\text{DACK}}$ pin • The same settings as above are available other than address register setting (e.g., above transfer modes can be specified) • One transfer can be performed in one bus cycle (the types of transfer modes are the same as those of dual address modes)			DSAR/$\overline{\text{DACK}}$	$\overline{\text{DACK}}$/DDAR

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

7.5   Operations

7.5.1   Address Modes

(1)   Dual Address Mode

In dual address mode, the transfer source address is specified in DSAR and the transfer destination address is specified in DDAR. A transfer at a time is performed in two bus cycles (when the data bus width is less than the data access size or the access address is not aligned with the boundary of the data access size, the number of bus cycles are needed more than two because one bus cycle is divided into multiple bus cycles).

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

The read and write cycles are not separated. Other bus cycles (bus cycle by other bus masters, refresh cycle, and external bus release cycle) are not generated between read and write cycles.

The $\overline{\text{TEND}}$ signal output is enabled or disabled by the TENDE bit in DMDR. The $\overline{\text{TEND}}$ signal is output in two bus cycles. When an idle cycle is inserted before the bus cycle, the $\overline{\text{TEND}}$ signal is also output in the idle cycle. The $\overline{\text{DACK}}$ signal is not output.

RENESAS

Figure 7.2 shows an example of the signal timing in dual address mode and figure 7.3 shows the operation in dual address mode.
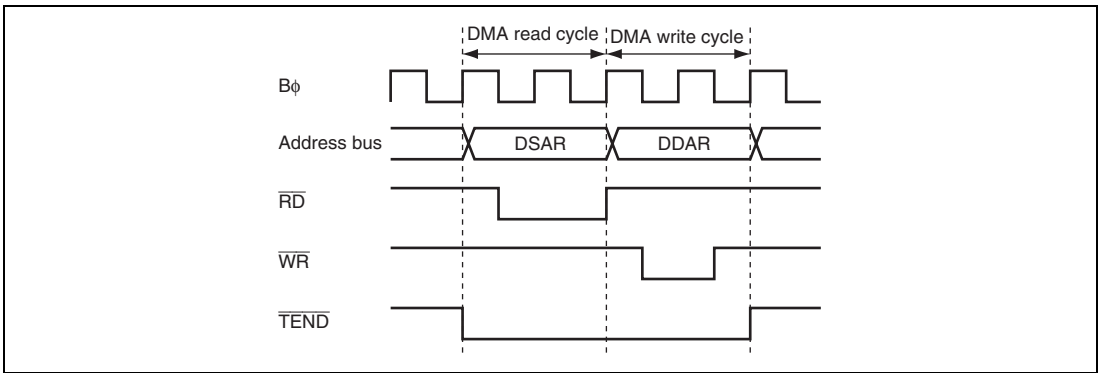
Figure 7.2   Example of Signal Timing in Dual Address Mode
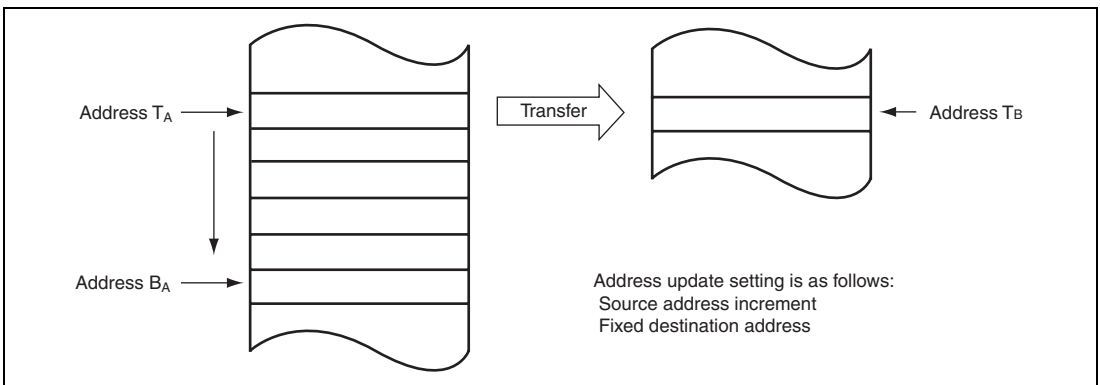
Figure 7.3   Operations in Dual Address Mode

(2)   Single Address Mode

In single address mode, data between an external device and an external memory is directly transferred using the $\overline{\text{DACK}}$ pin instead of DSAR or DDAR. A transfer at a time is performed in one bus cycle. In this mode, the data bus width must be the same as the data access size. For details on the data bus width, see section 6, Bus Controller (BSC).

The DMAC accesses an external device with $\overline{\text{DACK}}$ as the transfer source or destination by outputting the strobe signal to the external device ($\overline{\text{DACK}}$) and accesses the other transfer target by outputting the address. Accordingly, the DMA transfer is performed in one bus cycle. Figure 7.4 shows an example of a transfer between an external memory and an external device with the $\overline{\text{DACK}}$ pin. In this example, the external device outputs data on the data bus and the data is written to the external memory in the same bus cycle.

The transfer direction is decided by the DIRS bit in DACR which specifies an external device with the $\overline{\text{DACK}}$ pin as the transfer source or destination. When DIRS = 0, data is transferred from an external memory (DSAR) to an external device with the $\overline{\text{DACK}}$ pin. When DIRS = 1, data is transferred from an external device with the $\overline{\text{DACK}}$ pin to an external memory (DDAR). The settings of registers which are not used as the transfer source or destination are ignored.

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

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

Figure 7.5 shows an example of timing charts in single address mode and figure 7.6 shows an example of operation in single address mode.
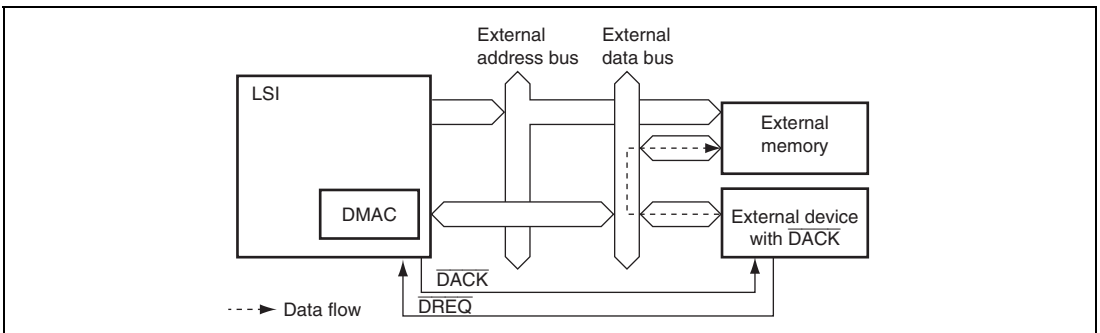
Figure 7.4   Data Flow in Single Address Mode

RENESAS

Transfer from external memory to external device with $\overline{\text{DACK}}$
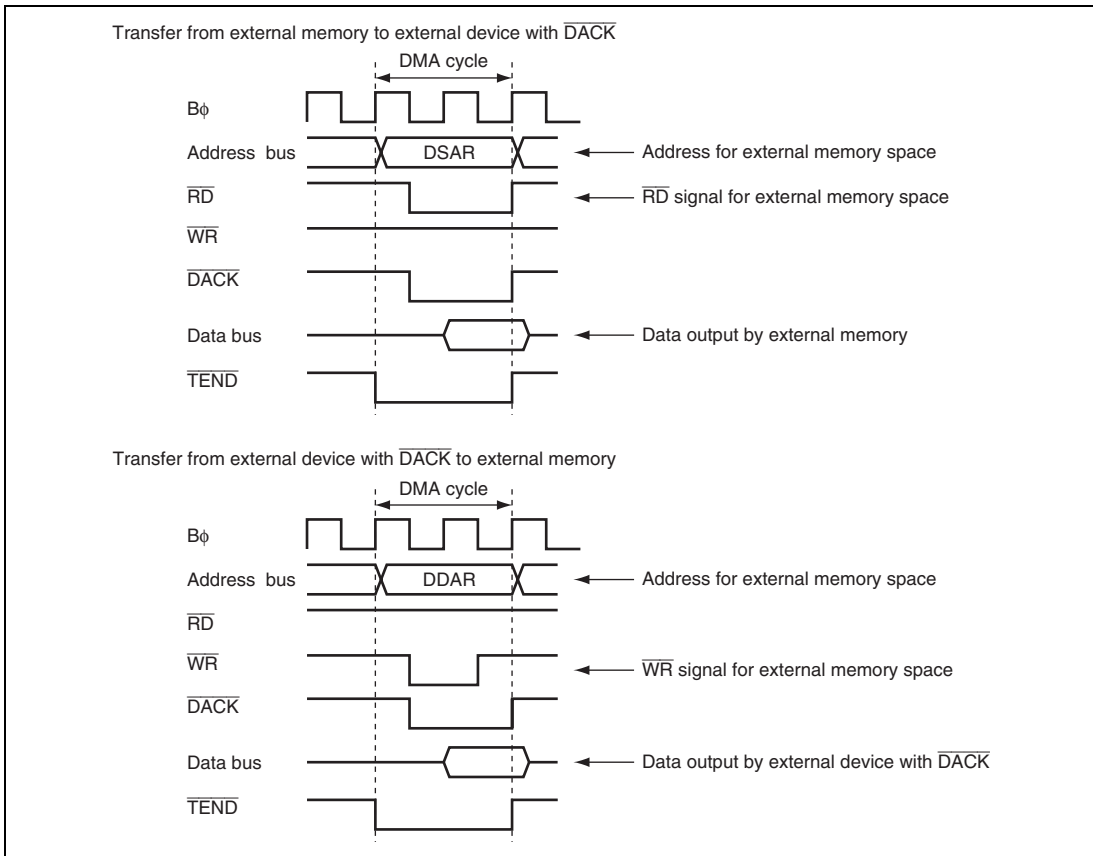
Figure 7.5   Example of Signal Timing in Single Address Mode
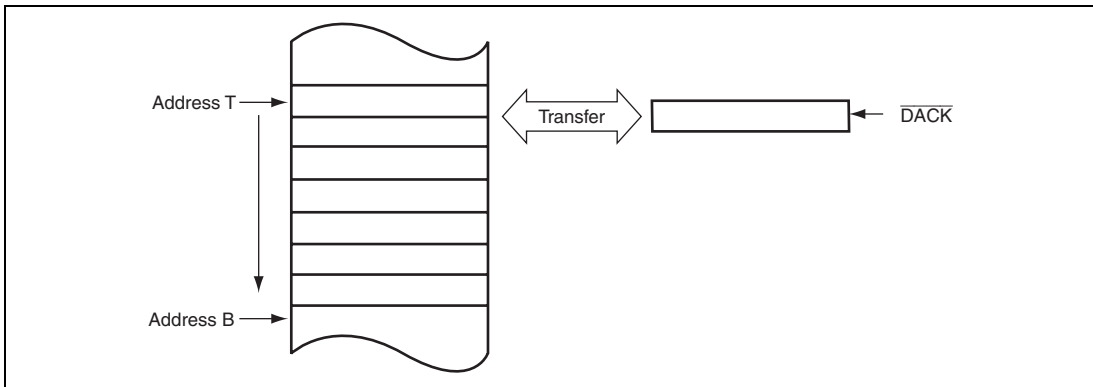
Figure 7.6   Operations in Single Address Mode

7.5.2   Transfer Modes

(1)   Normal Transfer Mode

In normal transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. DBSR is ignored in normal transfer mode.

The $\overline{\text{TEND}}$ signal is output only in the last DMA transfer.

Figure 7.7 shows an example of the signal timing in normal transfer mode and figure 7.8 shows the operation in normal transfer mode.
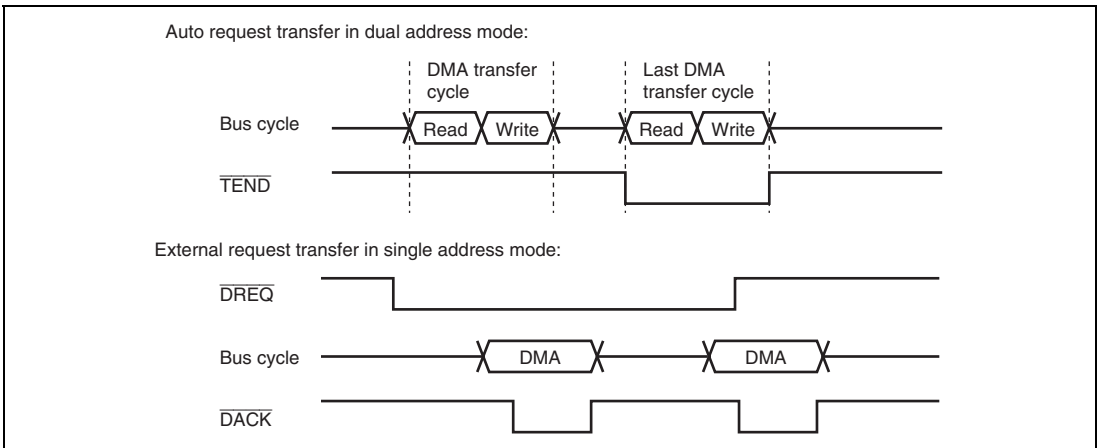
Figure 7.7   Example of Signal Timing in Normal Transfer Mode
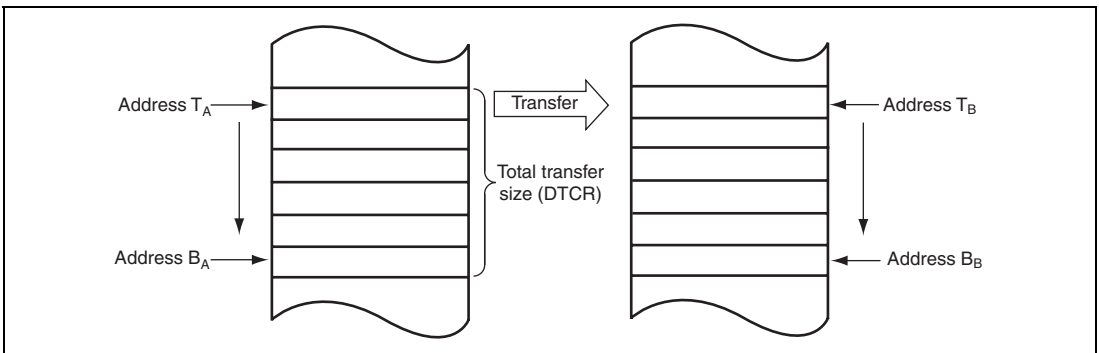
Figure 7.8   Operations in Normal Transfer Mode

RENESAS

(2)   Repeat Transfer Mode

In repeat transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. The repeat size can be specified in DBSR up to 65536 × data access size.

The repeat area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the repeat area returns to the transfer start address when the repeat size of transfers is completed. This operation is repeated until the total transfer size specified in DTCR is completed. When H'00000000 is specified in DTCR, it is regarded as the free running mode and repeat transfer is continued until the DTE bit in DMDR is cleared to 0.

In addition, a DMA transfer can be stopped and a repeat size end interrupt can be requested to the CPU when the repeat size of transfers is completed. When the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit is set to 1, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1 to complete the transfer. At this time, an interrupt is requested to the CPU when the ESIE bit in DMDR is set to 1.

The timings of the $\overline{\text{TEND}}$ signal are the same as in normal transfer mode.

Figure 7.9 shows the operation in repeat transfer mode while dual address mode is set.

When the repeat area is specified as neither source nor destination address side, the operation is the same as the normal transfer mode operation shown in figure 7.8. In this case, a repeat size end interrupt can also be requested to the CPU when the repeat size of transfers is completed.
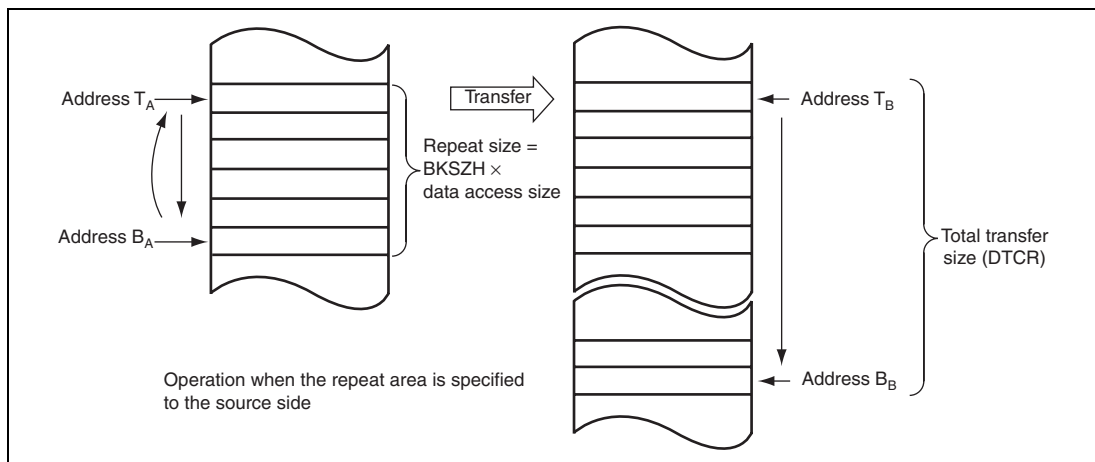
Figure 7.9   Operations in Repeat Transfer Mode

(3)   Block Transfer Mode

In block transfer mode, one block size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as total transfer size by DTCR. The block size can be specified in DBSR up to 65536 × data access size.

While one block of data is being transferred, transfer requests from other channels are suspended. When the transfer is completed, the bus is released to the other bus master.

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

The $\overline{\text{TEND}}$ signal is output every time 1-block data is transferred in the last DMA transfer cycle. When the external request is selected as an activation source, the low level detection of the $\overline{\text{DREQ}}$ signal (DREQS = 0) should be selected.

When an interrupt request by an extended repeat area overflow is used in block transfer mode, settings should be selected carefully. For details, see section 7.5.5, Extended Repeat Area Function.

Figure 7.10 shows an example of the DMA transfer timing in block transfer mode. The transfer conditions are as follows:

- Address mode: single address mode
- Data access size: byte
- 1-block size: three bytes

The block transfer mode operations in single address mode and in dual address mode are shown in figures 7.11 and 7.12, respectively.
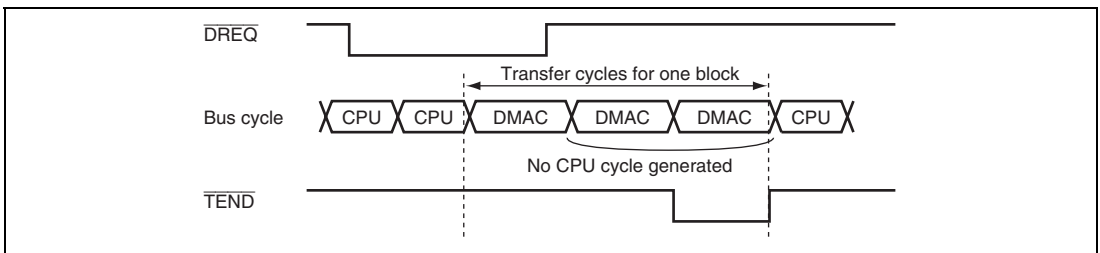
Figure 7.10   Operations in Block Transfer Mode

RENESAS
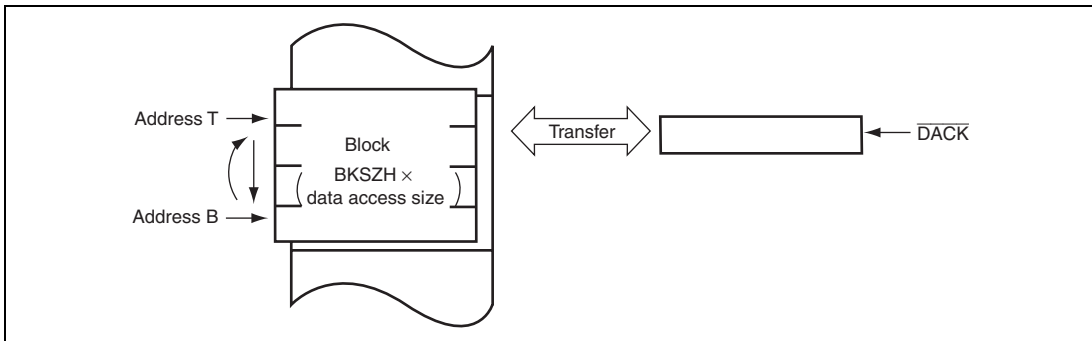
Figure 7.11   Operation in Single Address Mode in Block Transfer Mode (Block Area Specified)
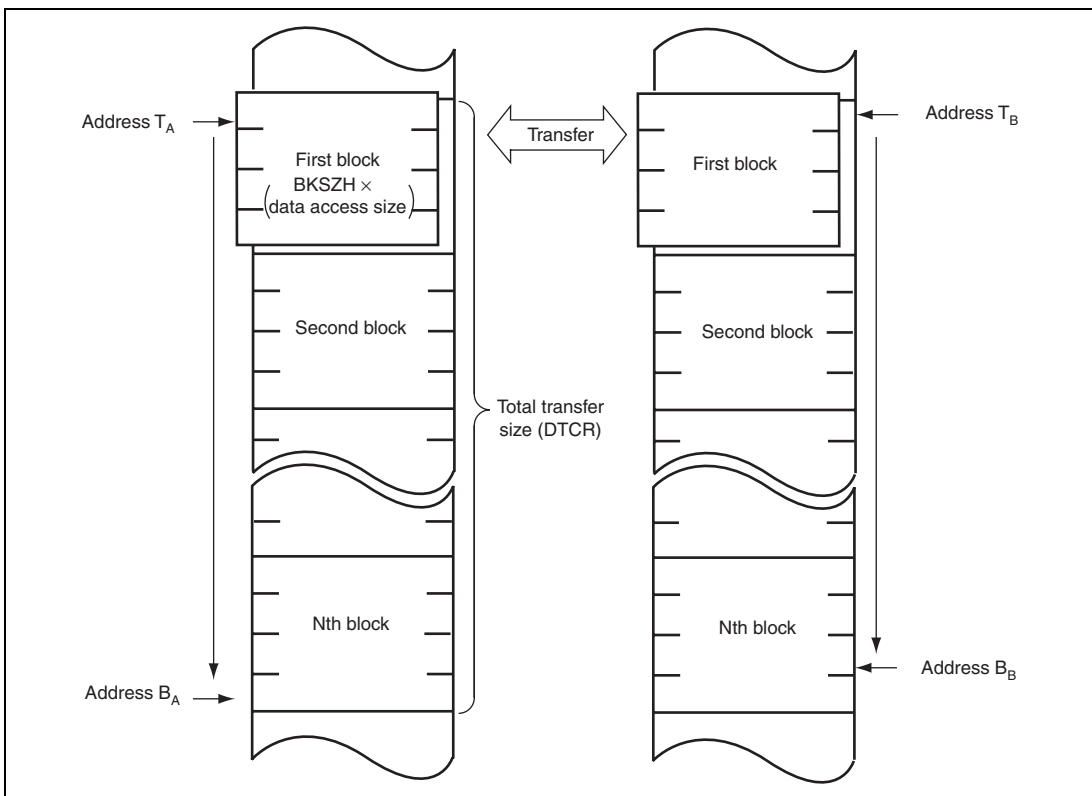
Figure 7.12   Operation in Dual Address Mode in Block Transfer Mode (Block Area Not Specified)

7.5.3   Activation Sources

The DMAC is activated by an auto request, an on-chip module interrupt, and an external request. The activation source is specified by bits DTF1 and DTF0 in DMDR.

(1)   Activation by Auto Request

The auto request activation is used when a transfer request from an external device or an on-chip peripheral module is not generated such as a transfer between memory and memory or between memory and an on-chip peripheral module which does not request a transfer. A transfer request is automatically generated inside the DMAC. In auto request activation, setting the DTE bit in DMDR starts a transfer. The bus mode can be selected from cycle stealing and burst modes.

(2)   Activation by On-Chip Module Interrupt

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

The activation source of the on-chip module interrupt is selected by the DMA module request select register (DMRSR). The activation sources are specified to the individual channels. Table 7.5 is a list of on-chip module interrupts for the DMAC.

The DMAC receives interrupt requests by on-chip peripheral modules independent of the interrupt controller. Therefore, the DMAC is not affected by priority given in the interrupt controller.

When the DMAC is activated while DTA = 1, the interrupt request flag is automatically cleared by a DMA transfer. If multiple channels use a single interrupt request as an activation source, when the channel with the highest priority is activated, the interrupt request flag is cleared. In this case, other channels may not be activated because the transfer request is not held in the DMAC.

When the DMAC is activated while DTA = 0, the interrupt request flag is not cleared by the DMAC. Use the CPU to clear the flag.

When an activation source is selected while DTE = 0, the activation source does not request a transfer to the DMAC. It requests an interrupt to the CPU.

In addition, make sure that an interrupt request flag as an on-chip module interrupt source is cleared to 0 before writing 1 to the DTE bit.

RENESAS

Table 7.5   List of On-Chip Module Interrupts to DMAC

On-Chip Module Interrupt Source	On-Chip Module	DMRSR (Vector Number)
ADI0 (A/D conversion end interrupt)	A/D_0	86
ADI1 (A/D conversion end interrupt)	A/D_1	87
TGI0A (TGR0A input capture/compare match)	TPU_0	88
TGI1A (TGR1A input capture/compare match)	TPU_1	93
TGI2A (TGR2A input capture/compare match)	TPU_2	97
TGI3A (TGR3A input capture/compare match)	TPU_3	101
TGI4A (TGR4A input capture/compare match)	TPU_4	106
TGI5A (TGR5A input capture/compare match)	TPU_5	110
RXI_0 (receive data full interrupt for SCI channel 0)	SCI_0	145
TXI_0 (transmit data empty interrupt for SCI channel 0)	SCI_0	146
RXI_2 (receive data full interrupt for SCI channel 2)	SCI_2	153
TXI_2 (transmit data empty interrupt for SCI channel 2)	SCI_2	154
RXI_4 (receive data full interrupt for SCI channel 4)	SCI_4	161
TXI_4 (transmit data empty interrupt for SCI channel 4)	SCI_4	162
RXI_5 (receive data full interrupt for SCI channel 5)	SCI_5	193
TXI_5 (transmit data empty interrupt for SCI channel 5)	SCI_5	194
RM0_0 (message reception in Mailbox 0 for RCAN-ET channel 0)	RCAN-ET_0	220
RM0_1 (message reception in Mailbox 1 for RCAN-ET channel 1)	RCAN-ET_1	221
CMI0_0 (compare match interrupt for motor control PWM channel 0)	PWM_0	224
CMI1_0 (compare match interrupt for motor control PWM channel 1)	PWM_1	225
CMI0_1 (compare match interrupt for 16-bit PWM channel 0)	PWM16_0	228
CMI1_1 (compare match interrupt for 16-bit PWM channel 1)	PWM16_1	229
SGI_0 (attenuation end interrupt for SDG channel 0)	SDG_0	232
SGI_1 (attenuation end interrupt for SDG channel 1)	SDG_1	233
SGI_2 (attenuation end interrupt for SDG channel 2)	SDG_2	236
SGI_3 (attenuation end interrupt for SDG channel 3)	SDG_3	237

RENESAS

(3)   Activation by External Request

A transfer is started by a transfer request signal ($\overline{\text{DREQ}}$) from an external device. When a DMA transfer is enabled (DTE = 1), the DMA transfer is started by the $\overline{\text{DREQ}}$ assertion.

A transfer request signal is input to the $\overline{\text{DREQ}}$ pin. The $\overline{\text{DREQ}}$ signal is detected on the falling edge or low level. Whether the falling edge or low level detection is used is selected by the DREQS bit in DMDR. To perform a block transfer, select the low level detection.

When an external request is selected as an activation source, clear the DDR bit to 0 and set the ICR bit to 1 for the corresponding pin. For details, see section 8, I/O Ports.

When a DMA transfer between on-chip peripheral modules is performed, select an activation source form the auto request and on-chip module interrupt (the external request cannot be used).

7.5.4   Bus Access Modes

There are two types of bus access modes: cycle stealing and burst.

When an activation source is the auto request, the cycle stealing or burst mode is selected by bit DTF0 in DMDR. When an activation source is the on-chip module interrupt or external request, the cycle stealing mode is selected.

(1)   Cycle Stealing Mode

In cycle stealing mode, the DMAC releases the bus every time one unit of transfers (byte, word, longword, or 1-block size) is completed. After that, when a transfer is requested, the DMAC obtains the bus to transfer 1-unit data and then releases the bus on completion of the transfer. This operation is continued until the transfer end condition is satisfied.

When a transfer is requested to another channel during a DMA transfer, the DMAC releases the bus and then transfers data for the requested channel. For details on operations when a transfer is requested to multiple channels, see section 7.5.8, Priority of Channels.

Figure 7.13 shows an example of timing in cycle stealing mode. The transfer conditions are as follows:

- Address mode: Single address mode
- Sampling method of the $\overline{\text{DREQ}}$ signal: Low level detection
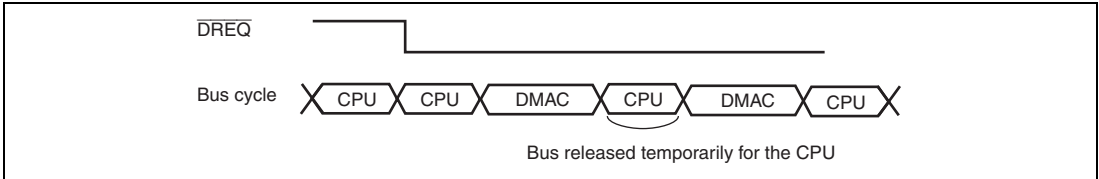
Figure 7.13   Example of Timing in Cycle Stealing Mode

(2)   Burst Access Mode

In burst mode, once it takes the bus, the DMAC continues a transfer without releasing the bus until the transfer end condition is satisfied. Even if a transfer is requested from another channel having priority, the transfer is not stopped once it is started. The DMAC releases the bus in the next cycle after the transfer for the channel in burst mode is completed. This is similarly to operation in cycle stealing mode. However, setting the IBCCS bit in BCR2 of the bus controller makes the DMAC release the bus to pass the bus to another bus master.

In block transfer mode, the burst mode setting is ignored (operation is the same as that in burst mode during one block of transfers). The DMAC is always operated in cycle stealing mode.

Clearing the DTE bit in DMDR stops a DMA transfer. A transfer requested before the DTE bit is cleared to 0 by the DMAC is executed. When an interrupt by a transfer size error, a repeat size end, or an extended repeat area overflow occurs, the DTE bit is cleared to 0 and the transfer ends.

Figure 7.14 shows an example of timing in burst mode.
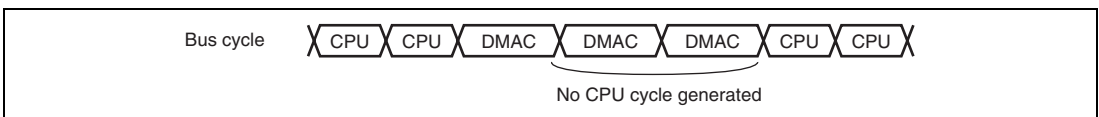
Figure 7.14   Example of Timing in Burst Mode

7.5.5   Extended Repeat Area Function

The source and destination address sides can be specified as the extended repeat area. The contents of the address register repeat addresses within the area specified as the extended repeat area. For example, to use a ring buffer as the transfer target, the contents of the address register should return to the start address of the buffer every time the contents reach the end address of the buffer (overflow on the ring buffer address). This operation can automatically be performed using the extended repeat area function of the DMAC.

The extended repeat areas can be specified independently to the source address register (DSAR) and destination address register (DDAR).

The extended repeat area on the source address is specified by bits SARA4 to SARA0 in DACR. The extended repeat area on the destination address is specified by bits DARA4 to DARA0 in DACR. The extended repeat area sizes for each side can be specified independently.

A DMA transfer is stopped and an interrupt by an extended repeat area overflow can be requested to the CPU when the contents of the address register reach the end address of the extended repeat area. When an overflow on the extended repeat area set in DSAR occurs while the SARIE bit in DACR is set to 1, the ESIF bit in DMDR is set to 1 and the DTE bit in DMDR is cleared to 0 to stop the transfer. At this time, if the ESIE bit in DMDR is set to 1, an interrupt by an extended repeat area overflow is requested to the CPU. When the DARIE bit in DACR is set to 1, an overflow on the extended repeat area set in DDAR occurs, meaning that the destination side is a target. During the interrupt handling, setting the DTE bit in DMDR resumes the transfer.

Figure 7.15 shows an example of the extended repeat area operation.

When the area represented by the lower three bits of DSAR (eight bytes)
is specified as the extended repeat area (SARA4 to SARA0 = B'00011)

```
External memory              Area specified
        ⋮                    by DSAR
   H'23FFFE
   H'23FFFF
   H'240000          H'240000  ◄── Repeat
   H'240001          H'240001
   H'240002          H'240002
   H'240003          H'240003
   H'240004          H'240004
   H'240005          H'240005
   H'240006          H'240006
   H'240007          H'240007  ──► An interrupt request by extended repeat area
   H'240008                        overflow can be generated.
   H'240009
        ⋮
```
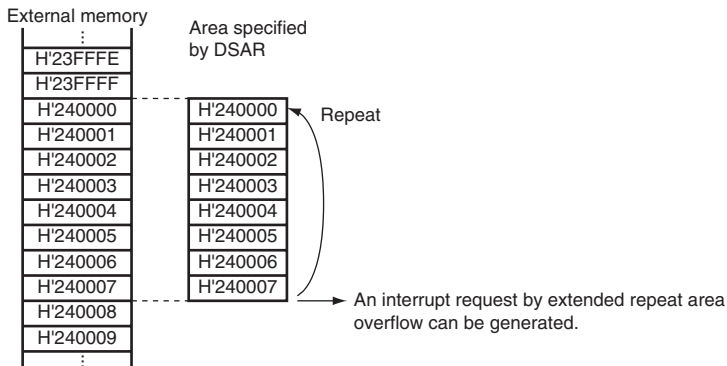
Figure 7.15   Example of Extended Repeat Area Operation

When an interrupt by an extended repeat area overflow is used in block transfer mode, the following should be taken into consideration.

When a transfer is stopped by an interrupt by an extended repeat area overflow, the address register must be set so that the block size is a power of 2 or the block size boundary is aligned with the extended repeat area boundary. When an overflow on the extended repeat area occurs during a transfer of one block, the interrupt by the overflow is suspended and the transfer overruns.

Figure 7.16 shows examples when the extended repeat area function is used in block transfer mode.

When the are represented by the lower three bits (eight bytes) of DSAR are specified as the extended repeat area (SARA4 to SARA0 = 3) and the block size in block transfer mode is specified to 5 (bits 23 to 16 in DTCR = 5).

```
External memory    Area specified   1st block      2nd block
        ⋮          by DSAR          transfer       transfer
   H'23FFFE
   H'23FFFF
   H'240000        H'240000         H'240000       H'240000  ◄
   H'240001        H'240001         H'240001       H'240001  ──► Interrupt
   H'240002        H'240002         H'240002                     request
   H'240003        H'240003         H'240003                     generated
   H'240004        H'240004         H'240004
   H'240005        H'240005                         H'240005
   H'240006        H'240006                         H'240006
   H'240007        H'240007                         H'240007   Block transfer
   H'240008                                                    continued
   H'240009
        ⋮
```

Figure 7.16   Example of Extended Repeat Area Function in Block Transfer Mode

7.5.6   Address Update Function using Offset

The source and destination addresses are updated by fixing, increment/decrement by 1, 2, or 4, or offset addition. When the offset addition is selected, the offset specified by the offset register (DOFR) is added to the address every time the DMAC transfers the data access size of data. This function realizes a data transfer where addresses are allocated to separated areas.

Figure 7.17 shows the address update method.
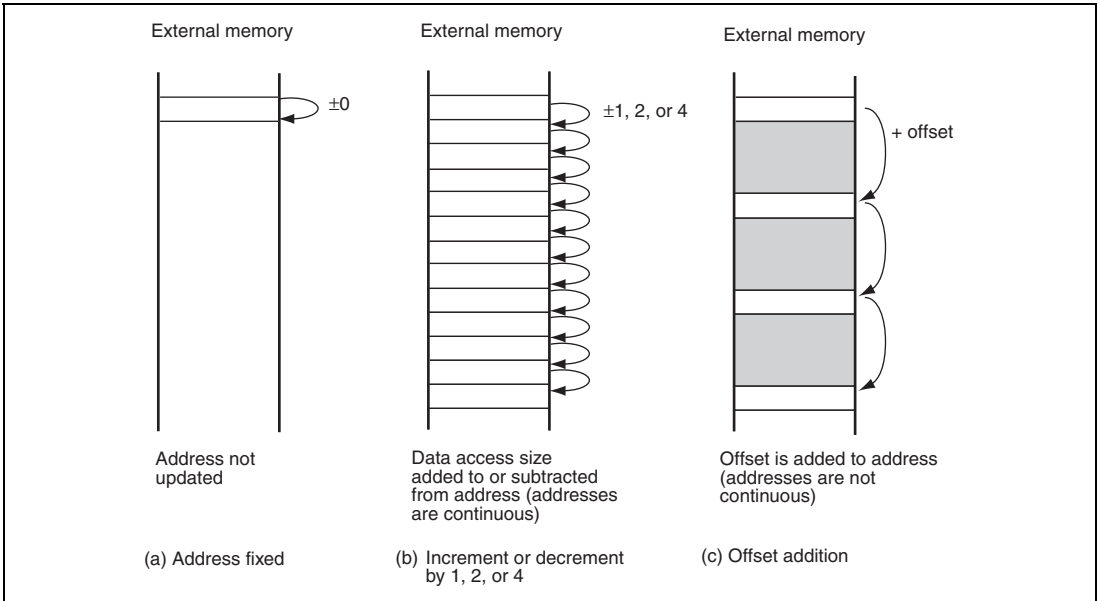
Figure 7.17   Address Update Method

In item (a), Address fixed, the transfer source or destination address is not updated indicating the same address.

In item (b), Increment or decrement by 1, 2, or 4, the transfer source or destination address is incremented or decremented by the value according to the data access size at each transfer. Byte, word, or longword can be specified as the data access size. The value of 1 for byte, 2 for word, and 4 for longword is used for updating the address. This operation realizes the data transfer placed in consecutive areas.

In item (c), Offset addition, the address update does not depend on the data access size. The offset specified by DOFR is added to the address every time the DMAC transfers data of the data access size.

The address is calculated by the offset set in DOFR and the contents of DSAR and DDAR. Although the DMAC calculates only addition, an offset subtraction can be realized by setting the negative value in DOFR. In this case, the negative value must be 2's complement.

(1)   Basic Transfer Using Offset

Figure 7.18 shows a basic operation of a transfer using the offset addition.
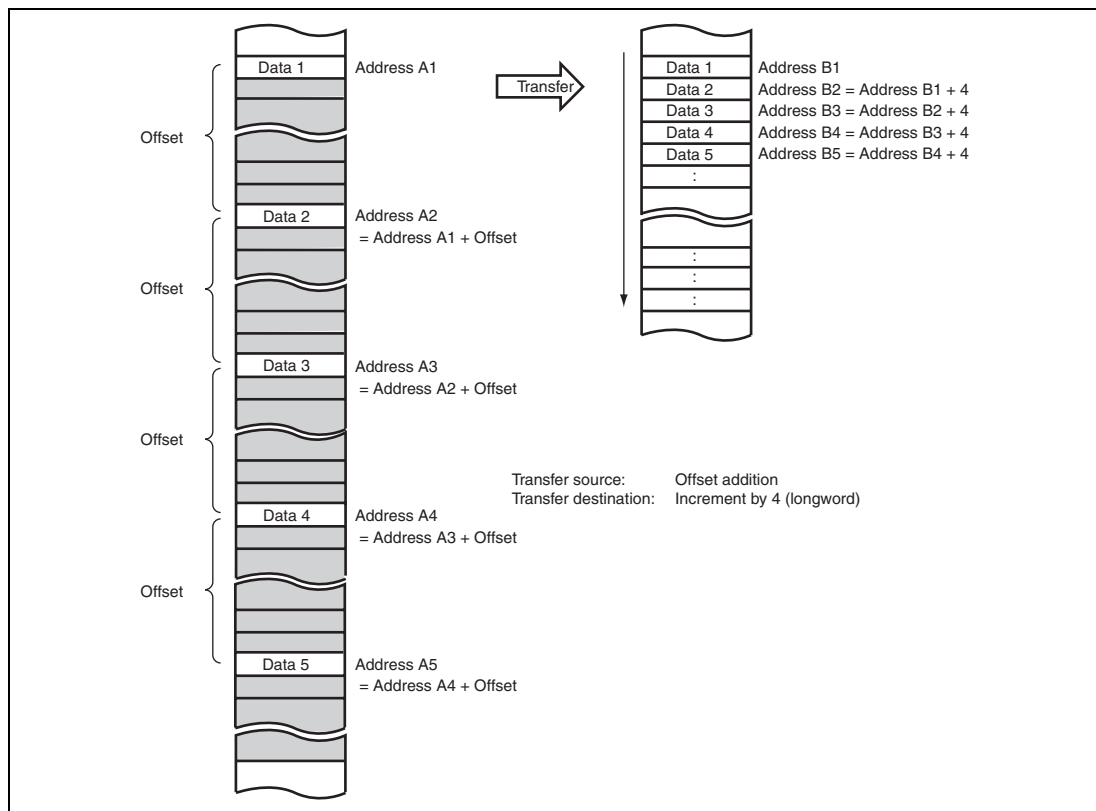
Figure 7.18   Operation of Offset Addition

In figure 7.18, the offset addition is selected as the transfer source address update and increment or decrement by 1, 2, or 4 is selected as the transfer destination address. The address update means that data at the address which is away from the previous transfer source address by the offset is read from. The data read from the address away from the previous address is written to the consecutive area in the destination side.

RENESAS

(2)   XY Conversion Using Offset

Figure 7.19 shows the XY conversion using the offset addition in repeat transfer mode.
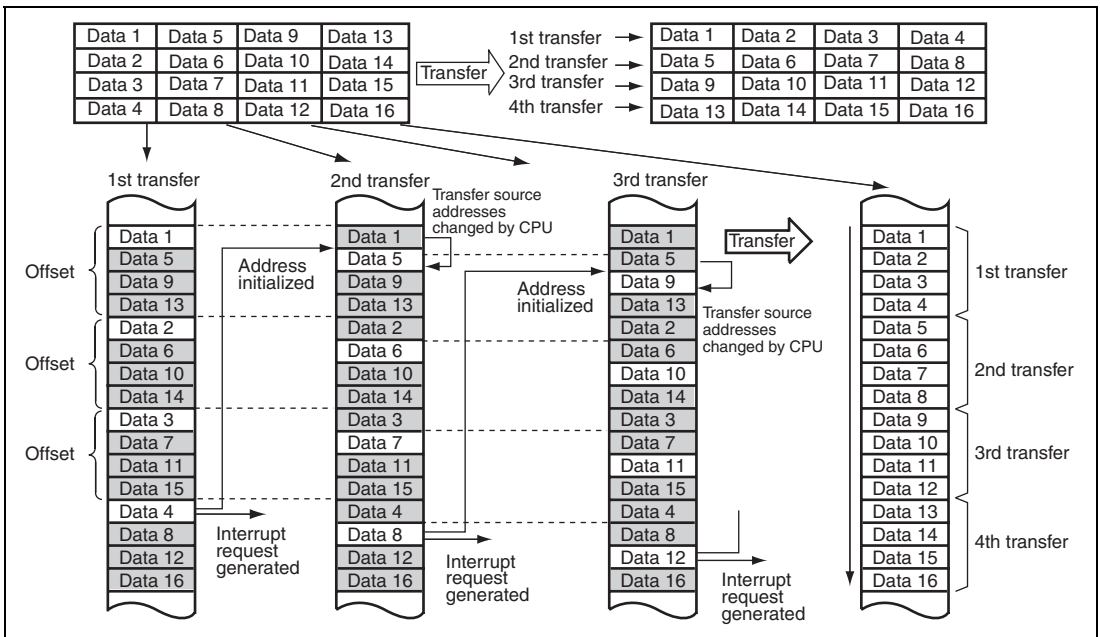
Figure 7.19   XY Conversion Operation Using Offset Addition in Repeat Transfer Mode

In figure 7.19, the source address side is specified to the repeat area by DACR and the offset addition is selected. The offset value is set to 4 × data access size (when the data access size is longword, H'00000010 is set in DOFR, as an example). The repeat size is set to 4 × data access size (when the data access size is longword, the repeat size is set to 4 × 4 = 16 bytes, as an example). The increment or decrement by 1, 2, or 4 is specified as the transfer destination address. A repeat size end interrupt is requested when the repeat size of transfers is completed.

When a transfer starts, the transfer source address is added to the offset every time data is transferred. The transfer data is written to the destination continuous addresses. When data 4 is transferred meaning that the repeat size of transfers is completed, the transfer source address returns to the transfer start address (address of data 1 on the transfer source) and a repeat size end interrupt is requested. While this interrupt stops the transfer temporarily, the contents of DSAR are written to the address of data 5 by the CPU (when the data access size is longword, write the data 1 address + 4). When the DTE bit in DMDR is set to 1, the transfer is resumed from the state when the transfer is stopped. Accordingly, operations are repeated and the transfer source data is transposed to the destination area (XY conversion).

RENESAS

Figure 7.29 shows a flowchart of the XY conversion.
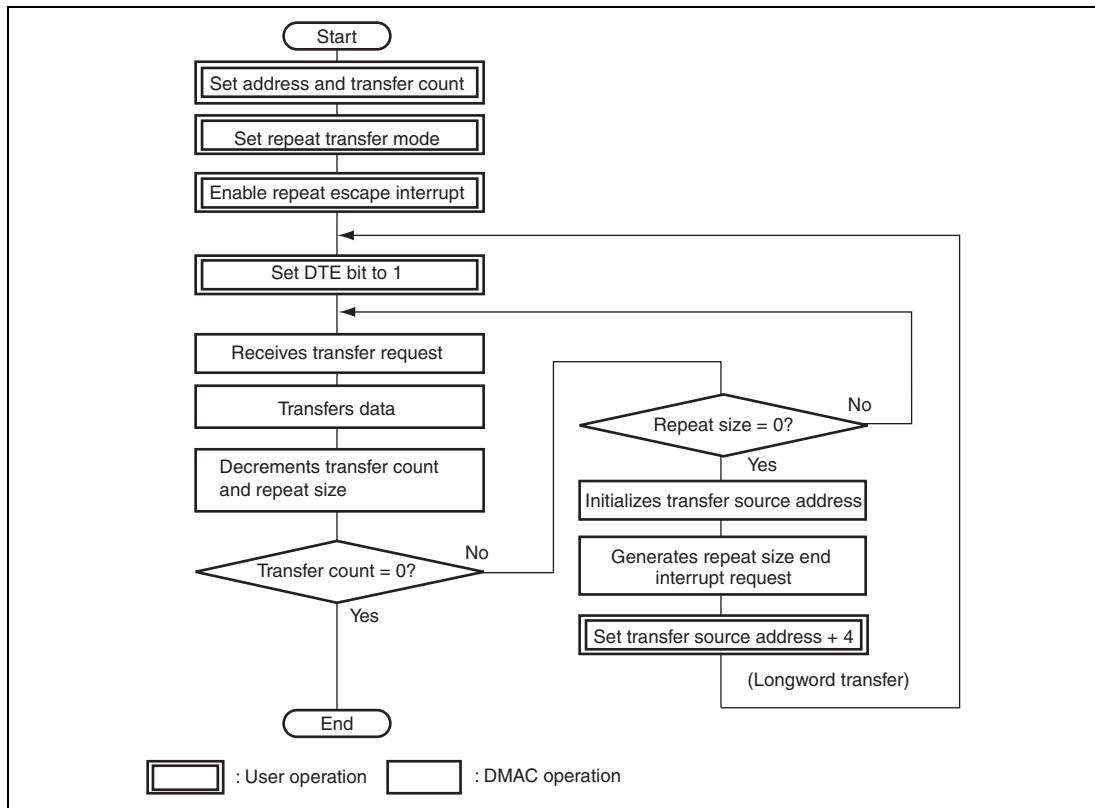
Figure 7.20   XY Conversion Flowchart Using Offset Addition in Repeat Transfer Mode

(3)   Offset Subtraction

When setting the negative value in DOFR, the offset value must be 2's complement. The 2's complement is obtained by the following formula.

2's complement of offset = 1 + ~offset (~: bit inversion)

Example: 2's complement of H'0001FFFF
= H'FFFE0000 + H'00000001
= H'FFFE0001

The value of 2's complement can be obtained by the NEG.L instruction.

7.5.7   Register during DMA Transfer

The DMAC registers are updated by a DMA transfer. The value to be updated differs according to the other settings and transfer state. The registers to be updated are DSAR, DDAR, DTCR, bits BKSZH and BKSZ in DBSR, and the DTE, ACT, ERRF, ESIF, and DTIF bits in DMDR.

(1)   DMA Source Address Register

When the transfer source address set in DSAR is accessed, the contents of DSAR are output and then are updated to the next address.

The increment or decrement can be specified by bits SAT1 and SAT0 in DACR. When SAT1 and SAT0 = B'00, the address is fixed. When SAT1 and SAT0 = B'01, the address is added with the offset. When SAT1 and SAT0 = B'10, the address is incremented. When SAT1 and SAT0 = B'11, the address is decremented. The size of increment or decrement depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the source address is word or longword, when the source address is not aligned with the word or longword boundary, the read bus cycle is divided into byte or word cycles. While data of one word or one longword is being read, the size of increment or decrement is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After one word or one longword of data is read, the address when the read cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the source address side, the source address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the source address side, operation follows the setting. The upper address bits are fixed and not affected by the address update.

While data is being transferred, DSAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DSAR during the transfer may be updated regardless of the access by the CPU. Moreover, DSAR for the channel being transferred must not be written to.

RENESAS

(2)   DMA Destination Address Register

When the transfer destination address set in DDAR is accessed, the contents of DDAR are output and then are updated to the next address.

The increment or decrement can be specified by bits DAT1 and DAT0 in DACR. When DAT1 and DAT0 = B'00, the address is fixed. When DAT1 and DAT0 = B'01, the address is added with the offset. When DAT1 and DAT0 = B'10, the address is incremented. When DAT1 and DAT0 = B'11, the address is decremented. The incrementing or decrementing size depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the destination address is word or longword, when the destination address is not aligned with the word or longword boundary, the write bus cycle is divided into byte and word cycles. While one word or one longword of data is being written, the incrementing or decrementing size is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After the one word or one longword of data is written, the address when the write cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the destination address side, the destination address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the destination address side, operation follows the setting. The upper address bits are fixed and not affected by the address update.

While data is being transferred, DDAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DDAR during the transfer may be updated regardless of the access by the CPU. Moreover, DDAR for the channel being transferred must not be written to.

(3)   DMA Transfer Count Register (DTCR)

A DMA transfer decrements the contents of DTCR by the transferred bytes. When byte data is transferred, DTCR is decremented by 1. When word data is transferred, DTCR is decremented by 2. When longword data is transferred, DTCR is decremented by 4. However, when DTCR = 0, the contents of DTCR are not changed since the number of transfers is not counted.

RENESAS

While data is being transferred, all the bits of DTCR may be changed. DTCR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DTCR during the transfer may be updated regardless of the access by the CPU. Moreover, DTCR for the channel being transferred must not be written to.

When a conflict occurs between the address update by DMA transfer and write access by the CPU, the CPU has priority. When a conflict occurs between change from 1, 2, or 4 to 0 in DTCR and write access by the CPU (other than 0), the CPU has priority in writing to DTCR. However, the transfer is stopped.

(4)   DMA Block Size Register (DBSR)

DBSR is enabled in block or repeat transfer mode. Bits 31 to 16 in DBSR function as BKSZH and bits 15 to 0 in DBSR function as BKSZ. The BKSZH bits (16 bits) store the block size and repeat size and its value is not changed. The BKSZ bits (16 bits) function as a counter for the block size and repeat size and its value is decremented every transfer by 1. When the BKSZ value is to change from 1 to 0 by a DMA transfer, 0 is not stored but the BKSZH value is loaded into the BKSZ bits.

Since the upper 16 bits of DBSR are not updated, DBSR can be accessed in words.

DBSR for the channel being transferred must not be written to.

(5)   DTE Bit in DMDR

Although the DTE bit in DMDR enables or disables data transfer by the CPU write access, it is automatically cleared to 0 according to the DMA transfer state by the DMAC.

The conditions for clearing the DTE bit by the DMAC are as follows:

- When the total size of transfers is completed
- When a transfer is completed by a transfer size error interrupt
- When a transfer is completed by a repeat size end interrupt
- When a transfer is completed by an extended repeat area overflow interrupt
- When a transfer is stopped by an NMI interrupt
- When a transfer is stopped by and address error
- Reset state
- Hardware standby mode
- When a transfer is stopped by writing 0 to the DTE bit

RENESAS

Writing to the registers for the channels when the corresponding DTE bit is set to 1 is prohibited (except for the DTE bit). When changing the register settings after writing 0 to the DTE bit, confirm that the DTE bit has been cleared to 0.

Figure 7.21 show the procedure for changing the register settings for the channel being transferred.
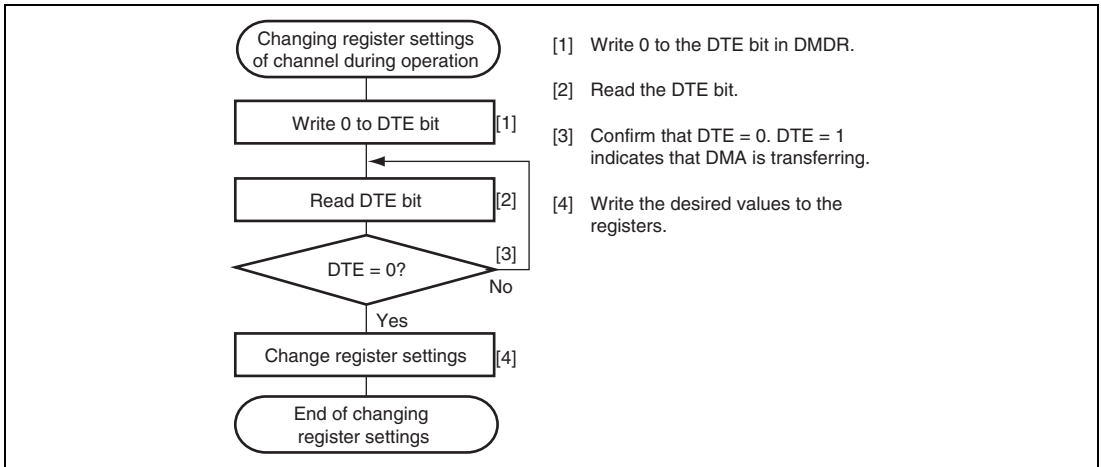
Figure 7.21   Procedure for Changing Register Setting for Channel Being Transferred

(6)   ACT Bit in DMDR

The ACT bit in DMDR indicates whether the DMAC is in the idle or active state. When DTE = 0 or DTE = 1 and the DMAC is waiting for a transfer request, the ACT bit is 0. Otherwise (the DMAC is in the active state), the ACT bit is 1. When individual transfers are stopped by writing 0 and the transfer is not completed, the ACT bit retains 1.

In block transfer mode, even if individual transfers are stopped by writing 0 to the DTE bit, the 1-block size of transfers is not stopped. The ACT bit retains 1 from writing 0 to the DTE bit to completion of a 1-block size transfer.

In burst mode, up to three times of DMA transfer are performed from the cycle in which the DTE bit is written to 0. The ACT bit retains 1 from writing 0 to the DTE bit to completion of DMA transfer.

(7)   ERRF Bit in DMDR

When an address error or an NMI interrupt occur, the DMAC clears the DTE bits for all the channels to stop a transfer. In addition, it sets the ERRF bit in DMDR_0 to 1 to indicate that an address error or an NMI interrupt has occurred regardless of whether or not the DMAC is in operation.

(8)   ESIF Bit in DMDR

When an interrupt by a transfer size error, a repeat size end, or an extended repeat area overflow is requested, the ESIF bit in DMDR is set to 1. When both the ESIF and ESIE bits are set to 1, a transfer escape interrupt is requested to the CPU.

The ESIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle of the interrupt source is completed.

The ESIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 7.8, Interrupt Sources.

(9)   DTIF Bit in DMDR

The DTIF bit in DMDR is set to 1 after the total transfer size of transfers is completed. When both the DTIF and DTIE bits in DMDR are set to 1, a transfer end interrupt by the transfer counter is requested to the CPU.

The DTIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle is completed.

The DTIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 7.8, Interrupt Sources.

7.5.8   Priority of Channels

The channels of the DMAC are given following priority levels: channel 0 > channel 1 > channel 2 > channel3. Table 7.6 shows the priority levels among the DMAC channels.

Table 7.6   Priority among DMAC Channels

Channel	Priority
Channel 0	High
Channel 1	
Channel 2	
Channel 3	Low

The channel having highest priority other than the channel being transferred is selected when a transfer is requested from other channels. The selected channel starts the transfer after the channel being transferred releases the bus. At this time, when a bus master other than the DMAC requests the bus, the cycle for the bus master is inserted.

In a burst transfer or a block transfer, channels are not switched.

Figure 7.22 shows a transfer example when multiple transfer requests from channels 0 to 2.
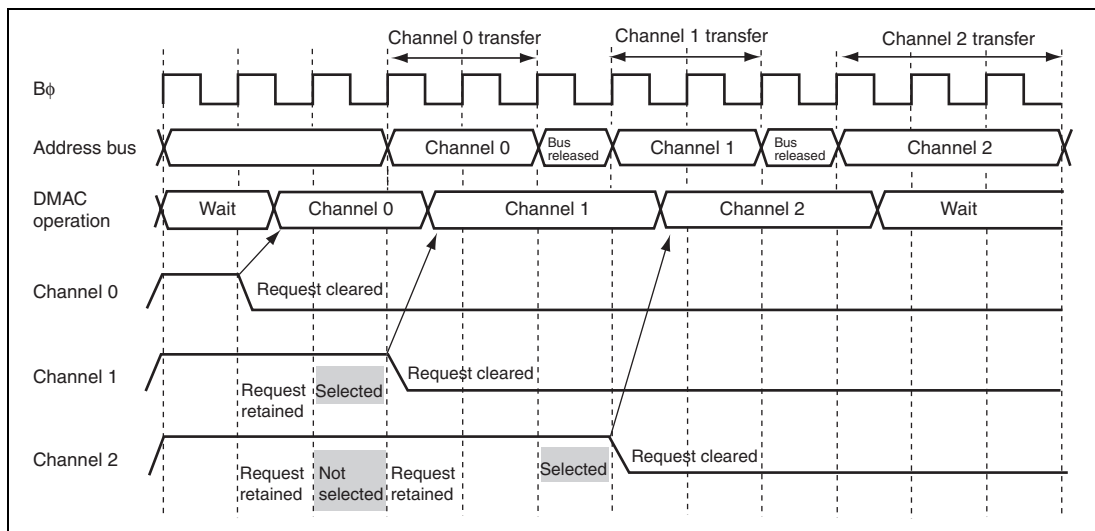
Figure 7.22   Example of Timing for Channel Priority

7.5.9   DMA Basic Bus Cycle

Figure 7.23 shows an examples of signal timing of a basic bus cycle. In figure 7.23, data is transferred in words from the 16-bit 2-state access space to the 8-bit 3-state access space. When the bus mastership is passed from the DMAC to the CPU, data is read from the source address and it is written to the destination address. The bus is not released between the read and write cycles by other bus requests. DMAC bus cycles follow the bus controller settings.
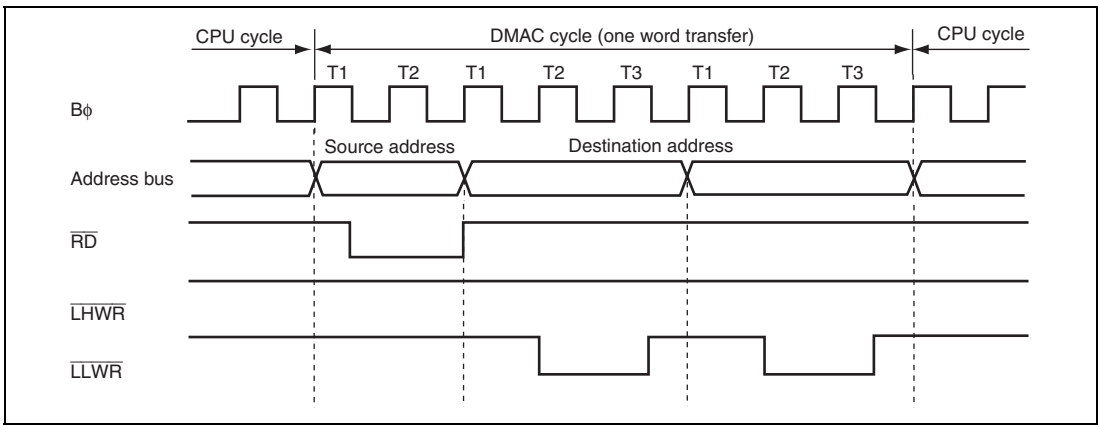
Figure 7.23   Example of Bus Timing of DMA Transfer

7.5.10   Bus Cycles in Dual Address Mode

(1)   Normal Transfer Mode (Cycle Stealing Mode)

In cycle stealing mode, the bus is released every time one transfer size of data (one byte, one word, or one longword) is completed. One bus cycle or more by the CPU are executed in the bus released cycles.

In figure 7.24, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by cycle stealing.
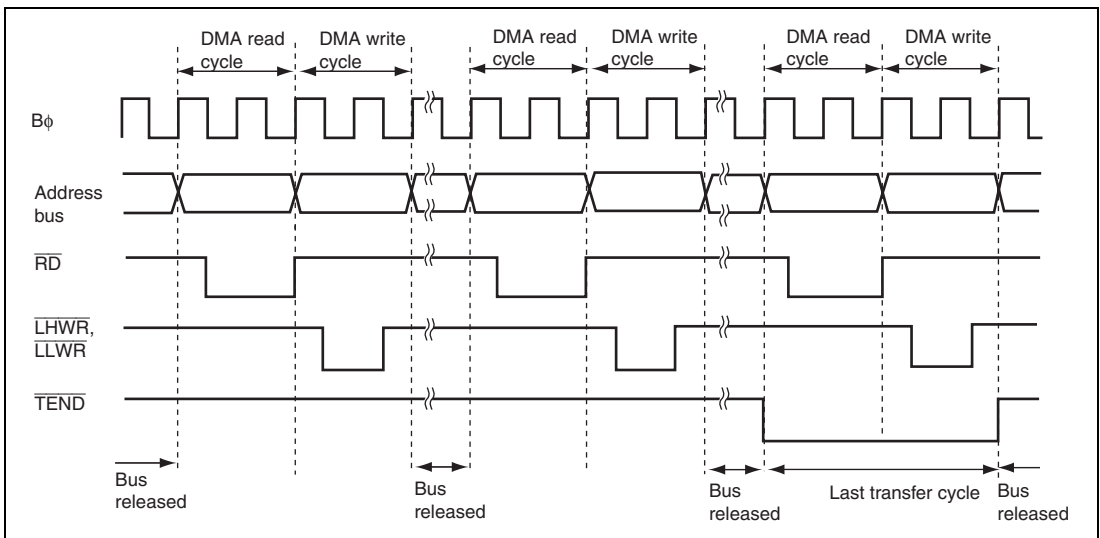
Figure 7.24   Example of Transfer in Normal Transfer Mode by Cycle Stealing

In figures 7.25 and 7.26, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in longwords from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by cycle stealing.

In figure 7.25, the transfer source (DSAR) is not aligned with a longword boundary and the transfer destination (DDAR) is aligned with a longword boundary.

In figure 7.26, the transfer source (DSAR) is aligned with a longword boundary and the transfer destination (DDAR) is not aligned with a longword boundary.
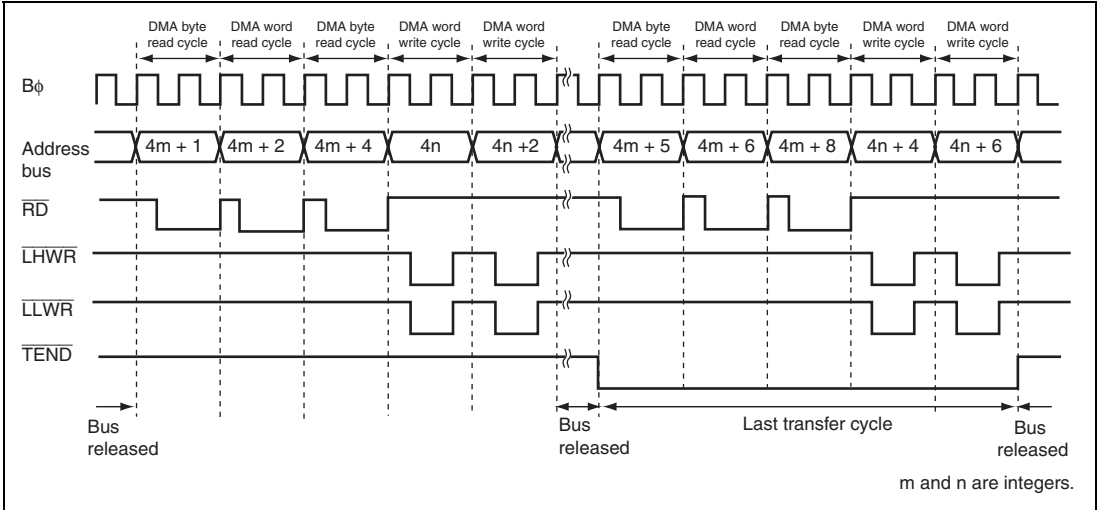
Figure 7.25   Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Source DSAR = Odd Address and Source Address Increment)
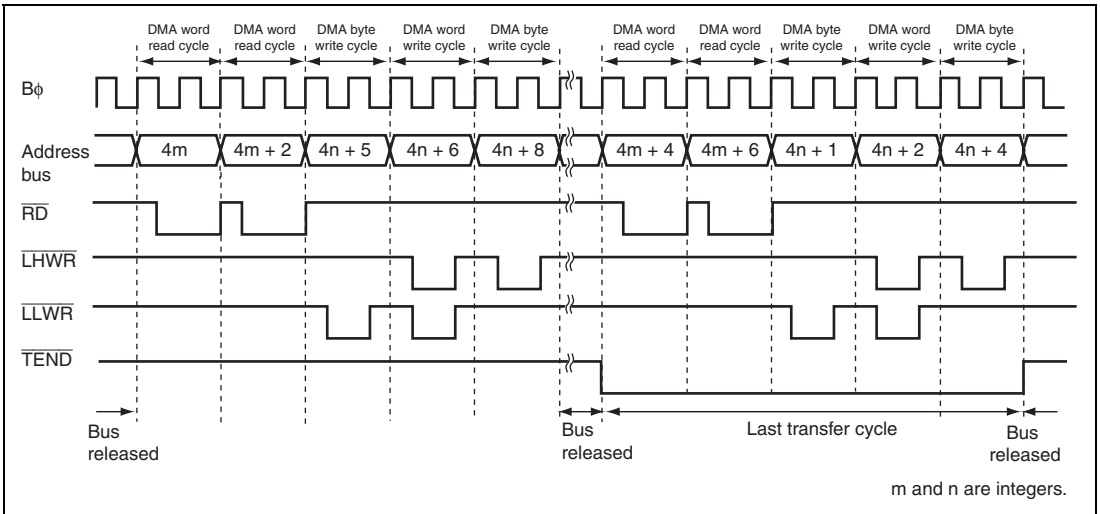
Figure 7.26   Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Destination DDAR = Odd Address and Destination Address Decrement)

(2)   Normal Transfer Mode (Burst Mode)

In burst mode, one byte, one word, or one longword of data continues to be transferred until the transfer end condition is satisfied.

When a burst transfer starts, a transfer request from a channel having priority is suspended until the burst transfer is completed.

In figure 7.27, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by burst access.
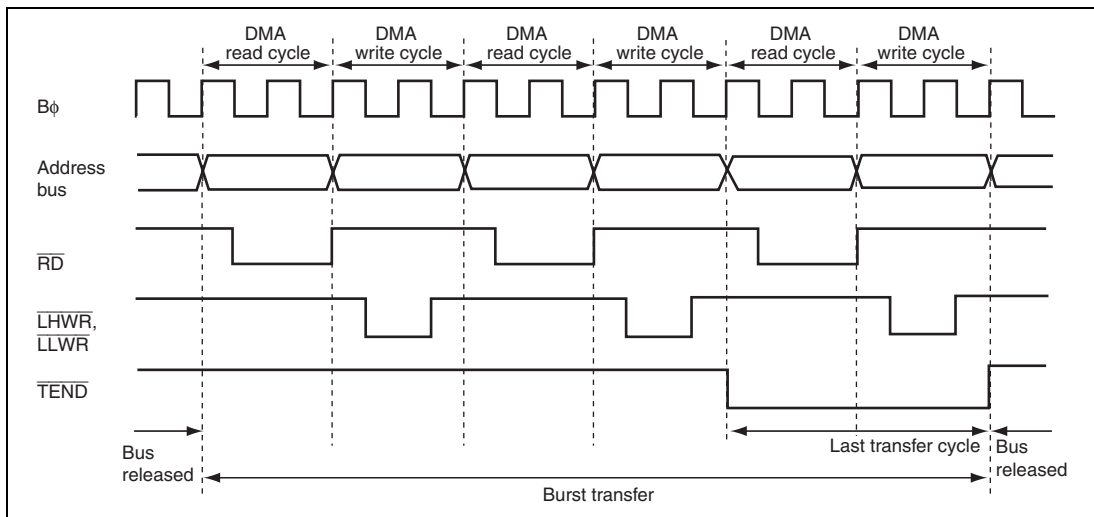
Figure 7.27   Example of Transfer in Normal Transfer Mode by Burst Access

RENESAS

(3)   Block Transfer Mode

In block transfer mode, the bus is released every time a 1-block size of transfers at a single transfer request is completed.

In figure 7.28, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in block transfer mode.
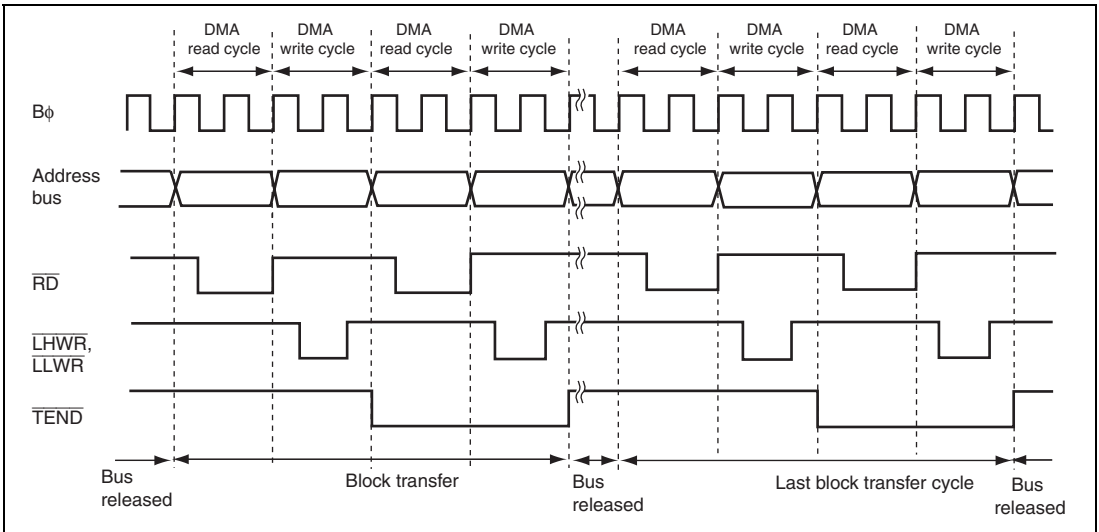
Figure 7.28   Example of Transfer in Block Transfer Mode

(4)   Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.29 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the DMA write cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.

[2][5]   The DMAC is activated and the transfer request is cleared.

[3][6]   A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bϕ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.

[4][7]   When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer request enable is resumed after completion of the write cycle. (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

**Figure 7.29   Example of Transfer in Normal Transfer Mode Activated
by $\overline{\text{DREQ}}$ Falling Edge**

(5)   Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 7.30 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5]   The DMAC is activated and the transfer request is cleared.
[3][6]   A DMA cycle is started.
[4][7]   Transfer request enable is resumed after completion of the write cycle.
   (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

**Figure 7.30   Example of Transfer in Normal Transfer Mode Activated
by $\overline{\text{DREQ}}$ Low Level**

RENESAS

Figure 7.31 shows an example of block transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the write cycle.
        (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.31   Example of Transfer in Block Transfer Mode Activated
by $\overline{\text{DREQ}}$ Low Level

RENESAS

(6)   Activation Timing by $\overline{\text{DREQ}}$ Low Level with NRD = 1

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 7.32 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level with NRD = 1.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5]   The DMAC is activated and the transfer request is cleared.
[3][6]   A DMA cycle is started.
[4][7]   Transfer request enable is resumed one cycle after completion of the write cycle.
   (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.32   Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1

RENESAS

7.5.11   Bus Cycles in Single Address Mode

(1)   Single Address Mode (Read and Cycle Stealing)

In single address mode, one byte, one word, or one longword of data is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU are executed in the bus released cycles.

In figure 7.33, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (read).
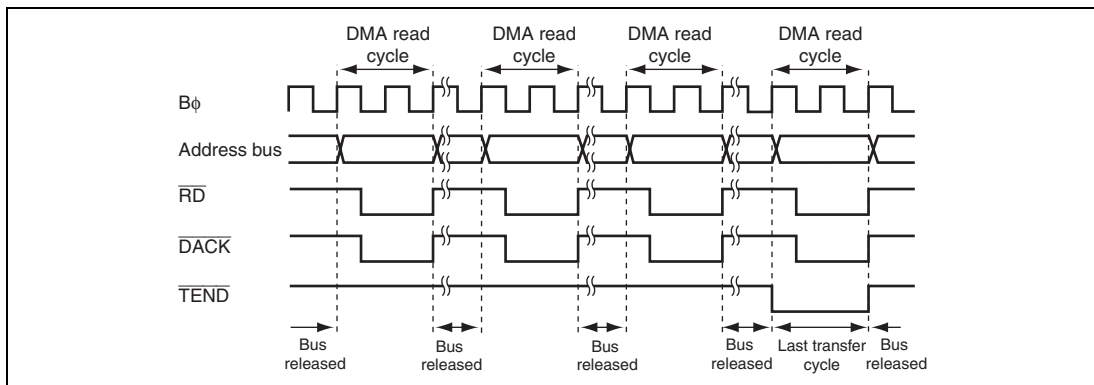
Figure 7.33   Example of Transfer in Single Address Mode (Byte Read)

(2)   Single Address Mode (Write and Cycle Stealing)

In single address mode, data of one byte, one word, or one longword is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU are executed in the bus released cycles.

In figure 7.34, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (write).
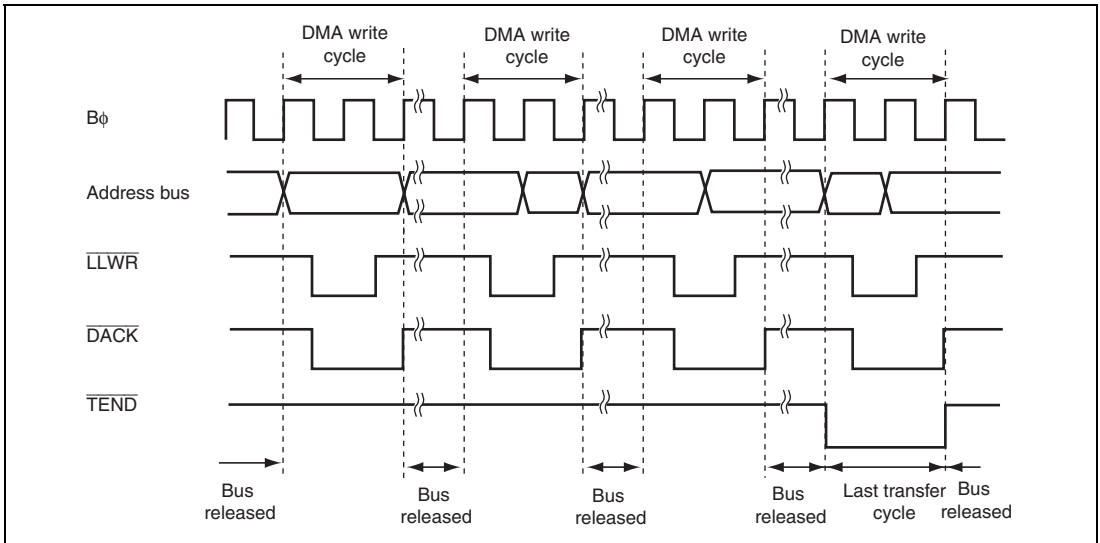
Figure 7.34   Example of Transfer in Single Address Mode (Byte Write)

(3)   Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.35 shows an example of single address mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the single cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5]   The DMAC is activated and the transfer request is cleared.
[3][6]   A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bϕ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.
[4][7]   When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer enable is resumed after completion of the write cycle.
      (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

**Figure 7.35   Example of Transfer in Single Address Mode Activated
by $\overline{\text{DREQ}}$ Falling Edge**

(4)   Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 7.36 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bφ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the single cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bφ signal and a transfer request is held.
[2][5]   The DMAC is activated and the transfer request is cleared.
[3][6]   A DMA cycle is started.
[4][7]   Transfer request enable is resumed after completion of the single cycle.
        (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bφ signal and a transfer request is held. This is the same as [1].)

**Figure 7.36   Example of Transfer in Single Address Mode Activated
by $\overline{\text{DREQ}}$ Low Level**

RENESAS

(5)   Activation Timing by $\overline{\text{DREQ}}$ Low Level with NRD = 1

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 7.37 shows an example of single address mode activated by the $\overline{\text{DREQ}}$ signal low level with NRD = 1.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bφ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after one cycle of the transfer request duration inserted by NRD = 1 on completion of the single cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.

[1]   After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bφ signal and a transfer request is held.
[2][5]   The DMAC is activated and the transfer request is cleared.
[3][6]   A DMA cycle is started.
[4][7]   Transfer request enable is resumed one cycle after completion of the single cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bφ signal and a transfer request is held. This is the same as [1].)
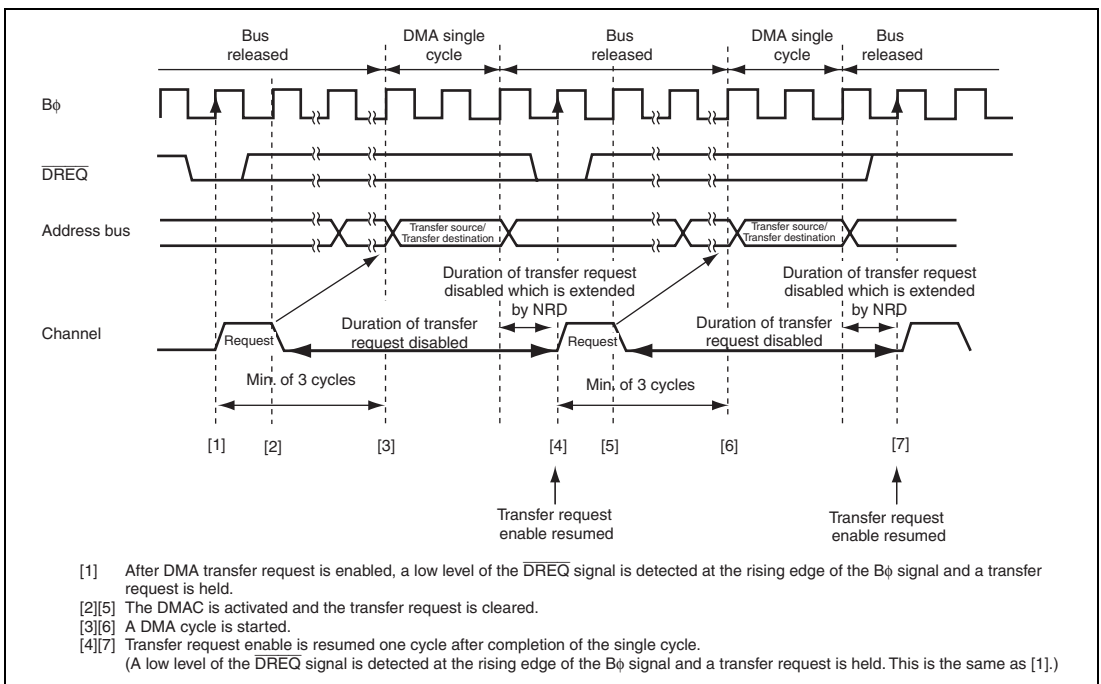
Figure 7.37   Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1

7.6   DMA Transfer End

Operations on completion of a transfer differ according to the transfer end condition. DMA transfer completion is indicated that the DTE and ACT bits in DMDR are changed from 1 to 0.

(1)   Transfer End by DTCR Change from 1, 2, or 4, to 0

When DTCR is changed from 1, 2, or 4 to 0, a DMA transfer for the channel is completed. The DTE bit in DMDR is cleared to 0 and the DTIF bit in DMDR is set to 1. At this time, when the DTIE bit in DMDR is set to 1, a transfer end interrupt by the transfer counter is requested. When the DTCR value is 0 before the transfer, the transfer is not stopped.

(2)   Transfer End by Transfer Size Error Interrupt

When the following conditions are satisfied while the TSEIE bit in DMDR is set to 1, a transfer size error occurs and a DMA transfer is terminated. At this time, the DTE bit in DMR is cleared to 0 and the ESIF bit in DMDR is set to 1.

- In normal transfer mode and repeat transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the data access size
- In block transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the block size

When the TSEIE bit in DMDR is cleared to 0, data is transferred until the DTCR value reaches 0. A transfer size error is not generated. Operation in each transfer mode is shown below.

- In normal transfer mode and repeat transfer mode, when the DTCR value is less than the data access size, data is transferred in bytes
- In block transfer mode, when the DTCR value is less than the block size, the specified size of data in DTCR is transferred instead of transferring the block size of data. The transfer is performed in bytes.

(3)   Transfer End by Repeat Size End Interrupt

In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit in DACR is set to 1, a repeat size end interrupt is requested. When the interrupt is requested to complete DMA transfer, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1. Under this condition, setting the DTE bit to 1 resumes the transfer.

In block transfer mode, when the next transfer is requested after completion of a 1-block size data transfer, a repeat size end interrupt can be requested.

RENESAS

(4)   Transfer End by Interrupt on Extended Repeat Area Overflow

When an overflow on the extended repeat area occurs while the extended repeat area is specified and the SARIE or DARIE bit in DACR is set to 1, an interrupt by an extended repeat area overflow is requested. When the interrupt is requested, the DMA transfer is terminated, the DTE bit in DMDR is cleared to 0, and the ESIF bit in DMDR is set to 1.

In dual address mode, even if an interrupt by an extended repeat area overflow occurs during a read cycle, the following write cycle is performed.

In block transfer mode, even if an interrupt by an extended repeat area overflow occurs during a 1-block transfer, the remaining data is transferred. The transfer is not terminated by an extended repeat area overflow interrupt unless the current transfer is complete.

(5)   Transfer End by Clearing DTE Bit in DMDR

When the DTE bit in DMDR is cleared to 0 by the CPU, a transfer is completed after the current DMA cycle and a DMA cycle in which the transfer request is accepted are completed.

In block transfer mode, a DMA transfer is completed after 1-block data is transferred.

(6)   Transfer End by NMI Interrupt

When an NMI interrupt is requested, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR_0 is set to 1. When an NMI interrupt is requested during a DMA transfer, the transfer is forced to stop. To perform DMA transfer after an NMI interrupt is requested, clear the ERRF bit to 0 and then set the DTE bits for the channels to 1.

The transfer end timings after an NMI interrupt is requested are shown below.

(a)   Normal Transfer Mode and Repeat Transfer Mode

In dual address mode, a DMA transfer is completed after completion of the write cycle for one transfer unit.
In single address mode, a DMA transfer is completed after completion of the bus cycle for one transfer unit.

(b)   Block Transfer Mode

A DMA transfer is forced to stop. Since a 1-block size of transfers is not completed, operation is not guaranteed.
In dual address mode, the write cycle corresponding to the read cycle is performed. This is similar to (a) in normal transfer mode.

RENESAS

(7)   Transfer End by Address Error

When an address error occurs, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR_0 is set to 1. When an address error occurs during a DMA transfer, the transfer is forced to stop. To perform a DMA transfer after an address error occurs, clear the ERRF bit to 0 and then set the DTE bits for the channels.

The transfer end timing after an address error is the same as that after an NMI interrupt.

(8)   Transfer End by Hardware Standby Mode or Reset

The DMAC is initialized by a reset and a transition to the hardware standby mode. A DMA transfer is not guaranteed.

7.7   Relationship among DMAC and Other Bus Masters

7.7.1   CPU Priority Control Function over DMAC

The CPU priority control function over DMAC can be used according to the CPU priority control register (CPUPCR) setting. For details, see section 5.7, CPU Priority Control Function over DMAC.

The priority level of the DMAC is specified by bits DMAP2 to DMAP0 and can be specified for each channel.

The priority level of the CPU is specified by bits CPUP2 to CPUP0. The value of bits CPUP2 to CPUP0 is updated according to the exception handling priority.

If the CPU priority control is enabled by the CPUPCE bit in CPUPCR, when the CPU has priority over the DMAC, a transfer request for the corresponding channel is masked and the transfer is not activated. When another channel has priority over or the same as the CPU, a transfer request is received regardless of the priority between channels and the transfer is activated.

If the priority level of the transfer request masked by the CPU priority control function is changed or the CPU priority is changed, the transfer request may be received and the transfer is started.

When the CPUPCE bit is cleared to 0, it is regarded as the lowest priority. Transfer requests masked are suspended. If a transfer request is suspended, it is cleared by clearing the DTE bit to 0.

RENESAS

7.7.2   Bus Arbitration among DMAC and Other Bus Masters

When DMA transfer cycles are consecutively performed, bus cycles of other bus masters may be inserted between the transfer cycles. The DMAC can release the bus temporarily to pass the bus to other bus masters.

The consecutive DMA transfer cycles may not be divided according to the transfer mode settings to achieve high-speed access.

The read and write cycles of a DMA transfer are not separated. Refreshing, external bus release, and on-chip bus master (CPU) cycles are not inserted between the read and write cycles of a DMA transfer.

In block transfer mode and an auto request transfer by burst access, bus cycles of the DMA transfer are consecutively performed. For this duration, since the DMAC has priority over the CPU, access to the external space is suspended (the IBCCS bit in the bus control register 2 (BCR2) is cleared to 0).

When the bus is passed to another channel or an auto request transfer by cycle stealing, bus cycles of the DMAC and on-chip bus master are performed alternatively.

When the arbitration function among the DMAC and on-chip bus masters is enabled by setting the IBCCS bit in BCR2, the bus is used alternatively except the bus cycles which are not separated. For details, see section 6, Bus Controller (BSC).

A conflict may occur between external space access of the DMAC and a refresh cycle or an external bus release cycle. Even if a burst or block transfer is performed by the DMAC, the transfer is stopped temporarily and a cycle of refresh or external bus release is inserted by the BSC (when the CPU external access does not have priority over a DMAC transfer, the transfer is not operated until the DMAC releases the bus).

In dual address mode, the DMAC releases the external bus after the external space write cycle. Since the read and write cycles are not separated, the bus is not released.

An internal space (on-chip memory and internal I/O registers) access of the DMAC and an external bus release cycle may be performed at the same time.

7.8   Interrupt Sources

The DMAC interrupt sources are a transfer end interrupt by the transfer counter and a transfer escape end interrupt which is generated when a transfer is terminated before the transfer counter reaches 0. Table 7.7 shows interrupt sources and priority.

Table 7.7   Interrupt Sources and Priority

Abbr.	Interrupt Sources	Priority
DMTEND0	Transfer end interrupt by channel 0 transfer counter	High
DMTEND1	Transfer end interrupt by channel 1 transfer counter	▲
DMTEND2	Transfer end interrupt by channel 2 transfer counter	
DMTEND3	Transfer end interrupt by channel 3 transfer counter	
DMEEND0	Interrupt by channel 0 transfer size error	
	Interrupt by channel 0 repeat size end	
	Interrupt by channel 0 extended repeat area overflow on source address	
	Interrupt by channel 0 extended repeat area overflow on destination address	
DMEEND1	Interrupt by channel 1 transfer size error	
	Interrupt by channel 1 repeat size end	
	Interrupt by channel 1 extended repeat area overflow on source address	
	Interrupt by channel 1 extended repeat area overflow on destination address	
DMEEND2	Interrupt by channel 2 transfer size error	
	Interrupt by channel 2 repeat size end	
	Interrupt by channel 2 extended repeat area overflow on source address	
	Interrupt by channel 2 extended repeat area overflow on destination address	
DMEEND3	Interrupt by channel 3 transfer size error	
	Interrupt by channel 3 repeat size end	
	Interrupt by channel 3 extended repeat area overflow on source address	
	Interrupt by channel 3 extended repeat area overflow on destination address	Low

Each interrupt is enabled or disabled by the DTIE and ESIE bits in DMDR for the corresponding channel. A DMTEND interrupt is generated by the combination of the DTIF and DTIE bits in DMDR. A DMEEND interrupt is generated by the combination of the ESIF and ESIE bits in DMDR. The DMEEND interrupt sources are not distinguished. The priority among channels is determined by the interrupt controller as shown in table 7.7. For details, see section 5, Interrupt Controller.

RENESAS

Each interrupt source is specified by the interrupt enable bit in the register for the corresponding channel. A transfer end interrupt by the transfer counter, a transfer size error interrupt, a repeat size end interrupt, an interrupt by an extended repeat area overflow on the source address, and an interrupt by an extended repeat area overflow on the destination address are enabled or disabled by the DTIE bit in DMDR, the TSEIE bit in DMDR, the RPTIE bit in DACR, SARIE bit in DACR, and the DARIE bit in DACR, respectively.

A transfer end interrupt by the transfer counter is generated when the DTIF bit in DMDR is set to 1. The DTIF bit is set to 1 when DTCR becomes 0 by a transfer while the DTIE bit in DMDR is set to 1.

An interrupt other than the transfer end interrupt by the transfer counter is generated when the ESIF bit in DMDR is set to 1. The ESIF bit is set to 1 when the conditions are satisfied by a transfer while the enable bit is set to 1.

A transfer size error interrupt is generated when the next transfer cannot be performed because the DTCR value is less than the data access size, meaning that the data access size of transfers cannot be performed. In block transfer mode, the block size is compared with the DTCR value for transfer error decision.

A repeat size end interrupt is generated when the next transfer is requested after completion of the repeat size of transfers in repeat transfer mode. Even when the repeat area is not specified in the address register, the transfer can be stopped periodically according to the repeat size. At this time, when a transfer end interrupt by the transfer counter is generated, the ESIF bit is set to 1.

An interrupt by an extended repeat area overflow on the source and destination addresses is generated when the address exceeds the extended repeat area (overflow). At this time, when a transfer end interrupt by the transfer counter, the ESIF bit is set to 1.

Figure 7.38 is a block diagram of interrupts and interrupt flags. To clear an interrupt, clear the DTIF or ESIF bit in DMDR to 0 in the interrupt handling routine or continue the transfer by setting the DTE bit in DMDR after setting the register. Figure 7.39 shows procedure to resume the transfer by clearing a interrupt.

Figure 7.38   Interrupt and Interrupt Sources

[1] Specify the values in the registers such as transfer counter and address register.
[2] Set the DTE bit in DMDR to 1 to resume DMA operation. Setting the DTE bit to 1 automatically clears the DTIF or ESIF bit in DMDR to 0 and an interrupt source is cleared.
[3] End the interrupt handling routine by the RTE instruction.
[4] Read that the DTIF or the ESIF bit in DMDR = 1 and then write 0 to the bit.
[5] Complete the interrupt handling routine and clear the interrupt mask.
[6] Specify the values in the registers such as transfer counter and address register.
[7] Set the DTE bit to 1 to resume DMA operation.

Figure 7.39   Procedure Example of Resuming Transfer by Clearing Interrupt Source

7.9   Notes on Usage

1. DMAC Register Access During Operation

   Except for clearing the DTE bit in DMDR, the settings for channels being transferred (including waiting state) must not be changed. The register settings must be changed during the transfer prohibited state.

2. Settings of Module Stop Function

   The DMAC operation can be enabled or disabled by the module stop control register. The DMAC is enabled by the initial value.

   Setting bit MSTPA13 in MSTPCRA stops the clock supplied to the DMAC and the DMAC enters the module stop state. However, when a transfer for a channel is enabled or when an interrupt is being requested, bit MSTPA13 cannot be set to 1. Clear the DTE bit to 0, clear the DTIF or DTIE bit in DMDR to 0, and then set bit MSTPA13.

   When the clock is stopped, the DMAC registers cannot be accessed. However, the following register settings are valid in the module stop state. Disable them before entering the module stop state, if necessary.

   — TENDE bit in DMDR is 1 (the TEND signal output enabled)

   — DACKE bit in DMDR is 1 (the DACK signal output enabled)

3. Activation by $\overline{\text{DREQ}}$ Falling Edge

   The $\overline{\text{DREQ}}$ falling edge detection is synchronized with the DMAC internal operation.

   A. Activation request waiting state: Waiting for detecting the $\overline{\text{DREQ}}$ low level. A transition to 2. is made.

   B. Transfer waiting state: Waiting for a DMAC transfer. A transition to 3. is made.

   C. Transfer prohibited state: Waiting for detecting the $\overline{\text{DREQ}}$ high level. A transition to 1. is made.

   After a DMAC transfer enabled, a transition to 1. is made. Therefore, the $\overline{\text{DREQ}}$ signal is sampled by low level detection at the first activation after a DMAC transfer enabled.

4. Acceptance of Activation Source

   At the beginning of an activation source reception, a low level is detected regardless of the setting of $\overline{\text{DREQ}}$ falling edge or low level detection. Therefore, if the $\overline{\text{DREQ}}$ signal is driven low before setting DMDR, the low level is received as a transfer request.

   When the DMAC is activated, clear the $\overline{\text{DREQ}}$ signal of the previous transfer.

RENESAS

Section 8   I/O Ports

Table 8.1 summarizes the port functions. The pins of each port also have other functions such as input/output pins of on-chip peripheral modules or external interrupt input pins. Each I/O port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, a port register (PORT) used to read the pin states, and an input buffer control register (ICR) that controls input buffer on/off.

Ports D to K have internal input pull-up MOSs and a pull-up MOS control register (PCR) that controls the on/off state of the input pull-up MOSs.

Ports 2 and F include an open-drain control register (ODR) that controls on/off of the output buffer PMOSs.

All of the I/O ports can drive a single TTL load and capacitive loads up to 30 pF.

All of the I/O ports can drive Darlington transistors when functioning as output ports.

Schmitt-trigger inputs are enabled when a port is used as the $\overline{\text{IRQ}}$, TPU, and IIC inputs.

Table 8.1   Port Functions

Port	Description	Bit	I/O	Function Input	Function Output	Schmitt-Trigger Input[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port 1	General I/O port also functioning as interrupt inputs, SCI I/Os, DMAC I/Os, A/D converter inputs, and IIC2 I/Os	7	P17/SCL0	$\overline{\text{IRQ7}}$-A/ $\overline{\text{ADTRG1}}$	—	$\overline{\text{IRQ7}}$-A, SCL0	—	[2]
		6	P16/SDA0	$\overline{\text{IRQ6}}$-A	$\overline{\text{DACK1}}$_A	$\overline{\text{IRQ6}}$-A, SDA0		[2]
		5	P15/SCL1	$\overline{\text{IRQ5}}$-A	$\overline{\text{TEND1}}$_A	$\overline{\text{IRQ5}}$-A, SCL1		[2]
		4	P14/SDA1	$\overline{\text{IRQ4}}$-A/ $\overline{\text{DREQ1}}$_A	—	$\overline{\text{IRQ4}}$-A, SDA1		[2]
		3	P13	$\overline{\text{IRQ3}}$-A/ $\overline{\text{ADTRG0}}$	—	$\overline{\text{IRQ3}}$-A		—
		2	P12/SCK2	$\overline{\text{IRQ2}}$-A	$\overline{\text{DACK0}}$_A	$\overline{\text{IRQ2}}$-A		
		1	P11	$\overline{\text{IRQ1}}$-A/ RxD2	$\overline{\text{TEND0}}$_A	$\overline{\text{IRQ1}}$-A		
		0	P10	$\overline{\text{IRQ0}}$-A/ $\overline{\text{DREQ0}}$_A	TxD2	$\overline{\text{IRQ0}}$-A		

Port	Description	Bit	I/O	Function Input	Function Output	Schmitt-Trigger Input *1	Input Pull-up MOS Function	Open-Drain Output Function
Port 2	General I/O port also functioning as interrupt inputs, TPU I/Os, SCI I/Os, and SSU* I/Os	7	P27/ TIOCB5	—	—	TIOCB5	—	O
		6	P26/ TIOCA5	—	—	TIOCA5		
		5	P25/ TIOCA4/	—	—	TIOCA4		
		4	P24/ TIOCB4	—	—	TIOCB4		
		3	P23/ TIOCD3/ $\overline{SCS\_1}$	—	—	TIOCD3		*3
		2	P22/ TIOCC3/ SSCK_1	—	TxD0	TIOCC3		*3
		1	P21/ TIOCA3/ SSI_1	RxD0	—	TIOCA3		*3
		0	P20/ TIOCB3/ SCK0/ SSO_1	—	—	TIOCB3		*3
Port 3	General I/O port also functioning as SDG outputs and TPU I/Os	7	P37/ TIOCB2	TCLKD-A	SGOUT_3	TIOCB2, TCLKD-A	—	—
		6	P36/ TIOCA2	—	SGOUT_2	TIOCA2		
		5	P35/ TIOCB1	TCLKC-A	SGOUT_1	TIOCB1, TCLKC-A		
		4	P34/ TIOCA1	—	SGOUT_0	TIOCA1		
		3	P33/ TIOCD0	TCLKB-A	—	TIOCD0, TCLKB-A		
		2	P32/ TIOCC0	TCLKA-A	—	TIOCC0, TCLKA-A		
		1	P31/ TIOCB0	—	—	TIOCB0		
		0	P30/ TIOCA0	—	—	TIOCA0		

RENESAS

Port	Description	Bit	Function			Schmitt-Trigger Input [1]	Input Pull-up MOS Function	Open-Drain Output Function
			I/O	Input	Output			
Port 4	General I/O port also functioning as A/D converter inputs	7	—	P47/AN11	—			
		6	—	P46/AN10	—			
		5	—	P45/AN9	—			
		4	—	P44/AN8	—			
		3	—	P43/AN15	—			
		2	—	P42/AN14	—			
		1	—	P41/AN13	—			
		0	—	P40/AN12	—			
Port 5	General I/O port also functioning as A/D converter inputs and D/A converter outputs	7	—	P57/AN7	DA1	—	—	—
		6	—	P56/AN6	DA0			
		5	—	P55/AN5	—			
		4	—	P54/AN4	—			
		3	—	P53/AN3	—			
		2	—	P52/AN2	—			
		1	—	P51/AN1	—			
		0	—	P50/AN0	—			
Port 6	General I/O port also functioning as SCI I/Os, DMAC I/Os, interrupt inputs, RCAN-ET I/Os, 16-bit PWM outputs, and H-UDI inputs	7	P67	$\overline{\text{IRQ15}}$-B	CTx_1	$\overline{\text{IRQ15}}$-B	—	—
		6	P66	$\overline{\text{IRQ14}}$-B/ CRx_1	—	$\overline{\text{IRQ14}}$-B		
		5	P65	$\overline{\text{IRQ13}}$-B	CTx_0/ $\overline{\text{DACK3}}$_B	$\overline{\text{IRQ13}}$-B		
		4	P64	$\overline{\text{IRQ12}}$-B/ CRx_0	$\overline{\text{TEND3}}$_B	$\overline{\text{IRQ12}}$-B		
		3	P63	$\overline{\text{IRQ11}}$-B/ $\overline{\text{DREQ3}}$_B/ TMS	PWM3_2	$\overline{\text{IRQ11}}$-B		
		2	P62/SCK4	$\overline{\text{IRQ10}}$-B/ $\overline{\text{TRST}}$	$\overline{\text{DACK2}}$_B	$\overline{\text{IRQ10}}$-B		
		1	P61	RxD4/ $\overline{\text{IRQ9}}$-B	$\overline{\text{TEND2}}$_B	$\overline{\text{IRQ9}}$-B		
		0	P60	$\overline{\text{IRQ8}}$-B/ $\overline{\text{DREQ2}}$_B	TxD4	$\overline{\text{IRQ8}}$-B		

Port	Description	Bit	Function I/O	Function Input	Function Output	Schmitt-Trigger Input *[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port A	General I/O port also functioning as bus control I/Os, 16-bit PWM outputs, and H-UDI I/Os	7	—	PA7	Bϕ	—	—	—
		6	PA6	—	$\overline{AS}$			
		5	PA5	—	$\overline{RD}$			
		4	PA4	—	$\overline{LHWR}$			
		3	PA3	—	$\overline{LLWR}$			
		2	PA2	TCK	PWM2_2			
		1	PA1	TDI	PWM1_2			
		0	PA0	—	PWM0_2/ TDO			
Port D	General I/O port also functioning as address outputs	7	PD7	—	A7	—	O	—
		6	PD6	—	A6			
		5	PD5	—	A5			
		4	PD4	—	A4			
		3	PD3	—	A3			
		2	PD2	—	A2			
		1	PD1	—	A1			
		0	PD0	—	A0			
Port E	General I/O port also functioning as address outputs	7	PE7	—	A15	—	O	—
		6	PE6	—	A14			
		5	PE5	—	A13			
		4	PE4	—	A12			
		3	PE3	—	A11			
		2	PE2	—	A10			
		1	PE1	—	A9			
		0	PE0	—	A8			

Port	Description	Bit	Function I/O	Function Input	Function Output	Schmitt-Trigger Input *1	Input Pull-up MOS Function	Open-Drain Output Function
Port F	General I/O port also functioning as address output, SCI I/O, and 16-bit PWM output	7	PF7/SCK5	—	A23/ PWM3_1	—	O*4	O*4
		6	PF6	RxD5	A22/ PWM2_1			
		5	PF5	—	TxD5/A21/ PWM1_1			
		4	PF4	—	A20/ PWM0_1			
		3	PF3	—	A19/ PWM3_0			
		2	PF2	—	A18/ PWM2_0			
		1	PF1	—	A17/ PWM1_0			
		0	PF0	—	A16/ PWM0_0			
Port H	General I/O port also functioning as bi-directional data bus	7	PH7/D7	—	—	—	O	—
		6	PH6/D6	—	—			
		5	PH5/D5	—	—			
		4	PH4/D4	—	—			
		3	PH3/D3	—	—			
		2	PH2/D2	—	—			
		1	PH1/D1	—	—			
		0	PH0/D0	—	—			
Port I	General I/O port also functioning as bi-directional data bus and SSU* I/O	7	PI7/D15	—	—	—	O	—
		6	PI6/D14	—	—			
		5	PI5/D13	—	—			
		4	PI4/D12	—	—			
		3	PI3/D11/ $\overline{SCS\_0}$	—	—		O*5	*3
		2	PI2/D10/ SSCK_0	—	—		O*5	*3
		1	PI1/D9/ SSI_0	—	—		O*5	*3
		0	PI0/D8/ SSO_0	—	—		O*5	*3

RENESAS

Port	Description	Bit	Function I/O	Function Input	Function Output	Schmitt-Trigger Input *[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port J	General I/O port also functioning as motor control PWM outputs	7	PJ7	—	PWM1H	—	O*[4]	—
		6	PJ6	—	PWM1G			
		5	PJ5	—	PWM1F			
		4	PJ4	—	PWM1E			
		3	PJ3	—	PWM1D			
		2	PJ2	—	PWM1C			
		1	PJ1	—	PWM1B			
		0	PJ0	—	PWM1A			
Port K	General I/O port also functioning as motor control PWM outputs	7	PK7	—	PWM2H	—	O*[4]	—
		6	PK6	—	PWM2G			
		5	PK5	—	PWM2F			
		4	PK4	—	PWM2E			
		3	PK3	—	PWM2D			
		2	PK2	—	PWM2C			
		1	PK1	—	PWM2B			
		0	PK0	—	PWM2A			

Notes: 1. Pins other than Schmitt-trigger input pin are CMOS input pins.

2. Open drain is valid only when the IIC function is used.

3. Open drain can be set only when the SSU (Synchronous Serial communication Unit) is used.

4. Input pull-up and open drain cannot be set when the PWM is used.

5. Input pull-up cannot be set when the SSU (Synchronous Serial communication Unit) is used.

* SSU: Synchronous Serial communication Unit

RENESAS

8.1   Register Descriptions

Table 8.2 lists each port registers.

Table 8.2   Register Configuration in Each Port

Port	Number of Pins	Registers					
		DDR	DR	PORT	ICR	PCR	ODR
Port 1	8	O	O	O	O	—	—
Port 2	8	O	O	O	O	—	O
Port 3	8	O	O	O	O	—	—
Port 4	8	—	—	O	O	—	—
Port 5	8	—	—	O	O	—	—
Port 6	8	O	O	O	O	—	—
Port A	8	O	O	O	O	—	—
Port D	8	O	O	O	O	O	—
Port E	8	O	O	O	O	O	—
Port F	8	O	O	O	O	O	O
Port H	8	O	O	O	O	O	—
Port I	8	O	O	O	O	O	—
Port J	8	O	O	O	O	O	—
Port K	8	O	O	O	O	O	—

Legend:

O: Register exists

—: No register exists

RENESAS

8.1.1   Data Direction Register (PnDDR) (n = 1, 2, 3, 6, A, D, E, F, H, I, J, and K)

DDR is an 8-bit write-only register that specifies the port input or output for each bit. A read from the DDR is invalid and DDR is always read as an undefined value.

The initial values of the port A change according to the start-up mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DDR	Pn6DDR	Pn5DDR	Pn4DDR	Pn3DDR	Pn2DDR	Pn1DDR	Pn0DDR
Initial Value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Table 8.3   Startup Mode and Initial Value

Port	Startup Mode	
	External Extended Mode	Single-Chip Mode
Port A	H'80	H'00
Other ports	H'00	

8.1.2   Data Register (PnDR) (n = 1, 2, 3, 6, A, D, E, F, H, I, J, and K)

DR is an 8-bit readable/writable register that stores the output data of the pins to be used as the general output port.

The initial value of DR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DR	Pn6DR	Pn5DR	Pn4DR	Pn3DR	Pn2DR	Pn1DR	Pn0DR
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RENESAS

8.1.3   Port Register (PORTn) (n = 1 to 6, A, D, E, F, H, I, J, and K)

PORT is an 8-bit read-only register that reflects the port pin state. A write to PORT is invalid. When PORT is read, the DR bits that correspond to the respective DDR bits set to 1 are read and the status of each pin whose corresponding DDR bit is cleared to 0 is also read regardless of the ICR value.

The initial value of PORT is undefined and is determined based on the port pin state.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	R	R	R	R	R	R	R	R

8.1.4   Input Buffer Control Register (PnICR) (n = 1 to 6, A, D, E, F, H, I, J, and K)

ICR is an 8-bit readable/writable register that controls the port input buffers.

For bits in ICR set to 1, the input buffers of the corresponding pins are valid. For bits in ICR cleared to 0, the input buffers of the corresponding pins are invalid and the input signals are fixed high.

When the pin functions as an input for the peripheral modules, the corresponding bits should be set to 1. The initial value should be written to a bit whose corresponding pin is not used as an input or is used as an analog input/output pin.

When PORT is read, the pin state is always read regardless of the ICR value. When the ICR value is cleared to 0 at this time, the read pin state is not reflected in a corresponding on-chip peripheral module.

If ICR is modified, an internal edge may occur depending on the pin state. Accordingly, ICR should be modified when the corresponding input pins are not used. For example, an $\overline{\text{IRQ}}$ input, modify ICR while the corresponding interrupt is disabled, clear the IRQF flag in ISR of the interrupt controller to 0, and then enable the corresponding interrupt. If an edge occurs after the ICR setting, the edge should be cancelled.

The initial value of ICR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ICR	Pn6ICR	Pn5ICR	Pn4ICR	Pn3ICR	Pn2ICR	Pn1ICR	Pn0ICR
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

8.1.5 Pull-Up MOS Control Register (PnPCR) (n = D to F, and H to K)

PCR is an 8-bit readable/writable register that controls on/off of the port input pull-up MOS.

If a bit in PCR is set to 1 while the pin is in input state, the input pull-up MOS corresponding to the bit in PCR is turned on.

The initial value of PCR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7PCR	Pn6PCR	Pn5PCR	Pn4PCR	Pn3PCR	Pn2PCR	Pn1PCR	Pn0PCR
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RENESAS

Table 8.4   Input Pull-Up MOS State

Port	Pin State	Reset	Hardware Standby Mode	Software Standby Mode	Other Operation
Port D	Address output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port E	Address output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port F	Address output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port H	Data input/output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port I	Data input/output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port J	Peripheral module output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF
Port K	Peripheral module output			OFF	
	Port output			OFF	
	Port input		OFF		ON/OFF

Legend:

OFF: The input pull-up MOS is always off.

ON/OFF: If PCR is set to 1, the input pull-up MOS is on; if PCR is cleared to 0, the input pull-up
          MOS is off.

8.1.6   Open-Drain Control Register (PnODR) (n = 2 and F)

ODR is an 8-bit readable/writable register that selects the open-drain output function.

If a bit in ODR is set to 1, the pin corresponding to that bit in ODR functions as an NMOS open-drain output. If a bit in ODR is cleared to 0, the pin corresponding to that bit in ODR functions as a CMOS output.

The initial value of ODR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ODR	Pn6ODR	Pn5ODR	Pn4ODR	Pn3ODR	Pn2ODR	Pn1ODR	Pn0ODR
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RENESAS

8.2   Output Buffer Control

This section describes the output priority of each pin.

The name of each peripheral module pin is followed by "_OE". This (for example: SCL0_OE) indicates whether the output of the corresponding function is valid (1) or if another setting is specified (0).

Each port output signal's valid setting is shown below. For details on the corresponding output signals, see the register description of each peripheral module.

For the I/O port pins whose initial values change according to the start-up mode, the initial values are indicated as "initial value E" for the start up in external expanded mode (expanded mode with on-chip ROM) and as "initial value S" for the start up in single-chip mode.

8.2.1   Port 1

(1)   P17/SCL0/$\overline{\text{IRQ7}}$-A/$\overline{\text{ADTRG1}}$

| | | Setting | |
| | | IIC2_0 | I/O Port |
Module Name	Pin Function	SCL0_OE*	P17DDR
IIC2_0	SCL0 I/O	1	—
I/O port	P17 output	0	1
	P17 input (initial value)	0	0

Note: *   When specified as I/O port: 1

(2)   P16/SDA0/$\overline{\text{IRQ6}}$-A/$\overline{\text{DACK1}}$_A

| | | Setting | | |
| | | DMAC_1 | IIC2_0 | I/O Port |
Module Name	Pin Function	$\overline{\text{DACK1}}$_A_OE	SDA0_OE*	P16DDR
DMAC_1	$\overline{\text{DACK1}}$_A output	1	—	—
IIC2_0	SDA0 I/O	0	1	—
I/O port	P16 output	0	0	1
	P16 input (initial value)	0	0	0

Note: *   When specified as I/O port: 1

(3)   P15/SCL1/$\overline{\text{IRQ5}}$-A/$\overline{\text{TEND1}}$_A

Module Name	Pin Function	Setting		
		DMAC_1	IIC2_0	I/O Port
		$\overline{\text{TEND1}}$_A_OE	SCL1_OE*	P15DDR
DMAC_1	$\overline{\text{TEND1}}$_A output	1	—	—
IIC2_0	SCL1 I/O	0	1	—
I/O port	P15 output	0	0	1
	P15 input (initial value)	0	0	0

Note: *   When specified as I/O port: 1

(4)   P14/SDA1/$\overline{\text{IRQ4}}$-A/$\overline{\text{DREQ1}}$_A

Module Name	Pin Function	Setting	
		IIC2_1	I/O Port
		SDA1_OE*	P14DDR
IIC2_1	SDA1 I/O	1	—
I/O port	P14 output	0	1
	P14 input (initial value)	0	0

Note: *   When specified as I/O port: 1

(5)   P13/$\overline{\text{IRQ3}}$-A/$\overline{\text{ADTRG0}}$_A

Module Name	Pin Function	Setting
		I/O Port
		P13DDR
I/O port	P13 output	1
	P13 input (initial value)	0

RENESAS

(6)   P12/SCK2/$\overline{\text{IRQ2}}$-A/$\overline{\text{DACK0}}$_A

Module Name	Pin Function	Setting		
		DMAC_0	SCI_2	I/O Port
		$\overline{\text{DACK0}}$_A_OE	SCK2_OE	P12DDR
DMAC_0	$\overline{\text{DACK0}}$_A output	1	—	—
SCI_2	SCK2 output	0	1	—
I/O port	P12 output	0	0	1
	P12 input (initial value)	0	0	0

(7)   P11/RxD2/$\overline{\text{IRQ1}}$-A/$\overline{\text{TEND0}}$_A

Module Name	Pin Function	Setting	
		DMAC_0	I/O Port
		$\overline{\text{TEND0}}$_A_OE	P11DDR
DMAC_0	$\overline{\text{TEND0}}$_A output	1	—
I/O port	P11 output	0	1
	P11 input (initial value)	0	0

(8)   P10/TxD2/$\overline{\text{IRQ0}}$-A/$\overline{\text{DREQ0}}$_A

Module Name	Pin Function	Setting	
		SCI_2	I/O Port
		TxD2_OE	P10DDR
SCI_2	TxD2 output	1	—
I/O port	P10 output	0	1
	P10 input (initial value)	0	0

8.2.2   Port 2

(1)   P27/TIOCB5

Module Name	Pin Function	Setting	
		TPU_5	**I/O Port**
		TIOCB5_OE	**P27DDR**
TPU_5	TIOCB5 output	1	—
I/O port	P27 output	0	1
	P27 input (initial value)	0	0

(2)   P26/TIOCA5

Module Name	Pin Function	Setting	
		TPU_5	**I/O Port**
		TIOCA5_OE	**P26DDR**
TPU_5	TIOCA5 output	1	—
I/O port	P26 output	0	1
	P26 input (initial value)	0	0

(3)   P25/TIOCA4

Module Name	Pin Function	Setting	
		TPU_4	**I/O Port**
		TIOCA4_OE	**P25DDR**
TPU_4	TIOCA4 output	1	—
I/O port	P25 output	0	1
	P25 input (initial value)	0	0

RENESAS

(4)   P24/TIOCB4

Module Name	Pin Function	Setting TPU_4 TIOCB4_OE	I/O Port P24DDR
TPU_4	TIOCB4 output	1	—
I/O port	P24 output	0	1
	P24 input (initial value)	0	0

(5)   P23/TIOCD3/$\overline{SCS\_1}$

Module Name	Pin Function	SSU*_1 $\overline{SCS\_1}$_OE	TPU_3 TIOCD3_OE	I/O Port P23DDR
SSU*_1	$\overline{SCS\_1}$ output	1	—	—
TPU_3	TIOCD3 output	0	1	—
I/O port	P23 output	0	0	1
	P23 input (initial value)	0	0	0

Note: *   SSU: Synchronous Serial communication Unit

(6)   P22/TIOCC3/TxD0/SSCK_1

Module Name	Pin Function	SSU*_1 SSCK_1_OE	TPU_3 TIOCC3_OE	SCI_0 TxD0_OE	I/O Port P22DDR
SSU*_1	SSCK_1 output	1	—	—	—
TPU_3	TIOCC3 output	0	1	—	—
SCI_0	TxD0 output	0	0	1	—
I/O port	P22 output	0	0	0	1
	P22 input (initial value)	0	0	0	0

Note: *   SSU: Synchronous Serial communication Unit

(7)   P21/TIOCA3/RxD0/SSI_1

Module Name	Pin Function	Setting		
		SSU*_1	TPU_3	I/O Port
		SSI_1_OE	TIOCA3_OE	P21DDR
SSU*_1	SSI_1 output	1	—	—
TPU_3	TIOCA3 output	0	1	—
I/O port	P21 output	0	0	1
	P21 input (initial value)	0	0	0

Note: *   SSU: Synchronous Serial communication Unit

(8)   P20/TIOCB3/SCK0/SSO_1

Module Name	Pin Function	Setting			
		SSU*_1	TPU_3	SCI_0	I/O Port
		SSO_1_OE	TIOCB3_OE	SCK0_OE	P20DDR
SSU*_1	SSO_1 output	1	—	—	—
TPU_3	TIOCB3 output	0	1	—	—
SCI_0	SCK0 output	0	0	1	—
I/O port	P20 output	0	0	0	1
	P20 input (initial value)	0	0	0	0

Note: *   SSU: Synchronous Serial communication Unit

RENESAS

8.2.3   Port 3

(1)   P37/TIOCB2/TCLKD-A/SGOUT_3

Module Name	Pin Function	Setting		
		SDG_3	TPU_2	I/O Port
		SGOUT_3_OE	TIOCB2_OE	P37DDR
SDG_3	SGOUT_3 output	1	—	—
TPU_2	TIOCB2 output	0	1	—
I/O port	P37 output	0	0	1
	P37 input (initial value)	0	0	0

(2)   P36/TIOCA2/SGOUT_2

Module Name	Pin Function	Setting		
		SDG_2	TPU_2	I/O Port
		SGOUT_2_OE	TIOCA2_OE	P36DDR
SDG_2	SGOUT_2 output	1	—	—
TPU_2	TIOCA2 output	0	1	—
I/O port	P36 output	0	0	1
	P36 input (initial value)	0	0	0

(3)   P35/TIOCB1/TCLKC-A/SGOUT_1

Module Name	Pin Function	Setting		
		SDG_1	TPU_1	I/O Port
		SGOUT_1_OE	TIOCB1_OE	P35DDR
SDG_1	SGOUT_1 output	1	—	—
TPU_1	TIOCB1 output	0	1	—
I/O port	P35 output	0	0	1
	P35 input (initial value)	0	0	0

(4)   P34/TIOCA1/SGOUT_0

| | | Setting | | |
| | | SDG_0 | TPU_1 | I/O Port |
Module Name	Pin Function	SGOUT_0_OE	TIOCA1_OE	P34DDR
SDG_0	SGOUT_0 output	1	—	—
TPU_1	TIOCA1 output	0	1	—
I/O port	P34 output	0	0	1
	P34 input (initial value)	0	0	0

(5)   P33/TIOCD0/TCLKB-A

| | | Setting | |
| | | TPU_0 | I/O Port |
Module Name	Pin Function	TIOCD0_OE	P33DDR
TPU_0	TIOCD0 output	1	—
I/O port	P33 output	0	1
	P33 input (initial value)	0	0

(6)   P32/TIOCC0/TCLKA-A

| | | Setting | |
| | | TPU_0 | I/O Port |
Module Name	Pin Function	TIOCC0_OE	P32DDR
TPU_0	TIOCC0 output	1	—
I/O port	P32 output	0	1
	P32 input (initial value)	0	0

(7)   P31/TIOCB0

		Setting	
		TPU_0	**I/O Port**
Module Name	**Pin Function**	**TIOCB0_OE**	**P31DDR**
TPU_0	TIOCB0 output	1	—
I/O port	P31 output	0	1
	P31 input (initial value)	0	0

(8)   P30/TIOCA0

		Setting	
		TPU_0	**I/O Port**
Module Name	**Pin Function**	**TIOCA0_OE**	**P30DDR**
TPU_0	TIOCA0 output	1	—
I/O port	P30 output	0	1
	P30 input (initial value)	0	0

8.2.4   Port 5

(1)   P57/AN7/DA1

Module Name	**Pin Function**
D/A	DA1 output

(2)   P56/AN6/DA0

Module Name	**Pin Function**
D/A	DA0 output

8.2.5   Port 6

(1)   P67/$\overline{\text{IRQ15}}$-B/CTx_1

Module Name	Pin Function	Setting	
		RCAN-ET_1	I/O Port
		CTx_1_OE	P67DDR
RCAN-ET_1	CTx_1 output	1	—
I/O port	P67 output	0	1
	P67 input (initial value)	0	0

(2)   P66/$\overline{\text{IRQ14}}$-B/CRx_1

Module Name	Pin Function	Setting
		I/O Port
		P66DDR
I/O port	P66 output	1
	P66 input (initial value)	0

(3)   P65/$\overline{\text{IRQ13}}$-B/CTx_0/$\overline{\text{DACK3}}$_B

Module Name	Pin Function	Setting		
		DMAC_3	RCAN-ET_0	I/O Port
		$\overline{\text{DACK3}}$_B_OE	CTx_0_OE	P65DDR
DMAC_3	$\overline{\text{DACK3}}$_B output	1	0	—
RCAN-ET_0	CTx_0 output	0	1	—
I/O port	P65 output	0	0	1
	P65 input (initial value)	0	0	0

(4)   P64/$\overline{\text{IRQ12}}$-B/CRx_0/$\overline{\text{TEND3}}$_B

Module Name	Pin Function	Setting DMAC_3 $\overline{\text{TEND3}}$_B_OE	I/O Port P64DDR
DMAC_3	$\overline{\text{TEND3}}$_B output	1	—
I/O port	P64 output	0	1
	P64 input (initial value)	0	0

(5)   P63/$\overline{\text{IRQ11}}$-B/PWM3_2/TMS/$\overline{\text{DREQ3}}$_B

Module Name	Pin Function	Setting 16-Bit PWM PWM3_2_OE	H-UDI HUDI_E	I/O Port P63DDR
16-bit PWM	PWM3_2 output	1	—	—
H-UDI	TMS input	0	1	—
I/O port	P63 output	0	0	1
	P63 input (initial value)	0	0	0

(6)   P62/SCK4/$\overline{\text{IRQ10}}$-B/$\overline{\text{TRST}}$/$\overline{\text{DACK2}}$_B

Module Name	Pin Function	Setting DMAC_2 $\overline{\text{DACK2}}$_B_OE	SCI_4 SCK4_OE	H-UDI HUDI_E	I/O Port P62DDR
DMAC_2	$\overline{\text{DACK2}}$_B output	1	—	—	—
SCI_4	SCK4 output	0	1	—	—
H-UDI	$\overline{\text{TRST}}$ input	0	0	1	—
I/O port	P62 output	0	0	0	1
	P62 input (initial value)	0	0	0	0

(7)   P61/RxD4/$\overline{\text{IRQ9}}$-B/$\overline{\text{TEND2}}$_B

Module Name	Pin Function	Setting	
		DMAC_2	I/O Port
		$\overline{\text{TEND2}}$_B_OE	P61DDR
DMAC_2	$\overline{\text{TEND2}}$_B output	1	—
I/O port	P61 output	0	1
	P61 input (initial value)	0	0

(8)   P60/TxD4/$\overline{\text{IRQ8}}$-B/$\overline{\text{DREQ2}}$_B

Module Name	Pin Function	Setting	
		SCI_4	I/O Port
		TxD4_OE	P60DDR
SCI_4	TxD4 output	1	—
I/O port	P60 output	0	1
	P60 input (initial value)	0	0

8.2.6   Port A

(1)   PA7/Bϕ

Module Name	Pin Function	Setting
		I/O Port
		PA7DDR
I/O port	Bϕ output (initial value E)	1
	PA7 input (initial value S)	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

RENESAS

(2)   PA6/$\overline{\text{AS}}$

		Setting	
		Bus Controller	I/O Port
Module Name	Pin Function	$\overline{\text{AS}}$_OE	PA6DDR
Bus controller	$\overline{\text{AS}}$ output* (initial value E)	1	—
I/O port	PA6 output	0	1
	PA6 input (initial value S)	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Note: *   Valid in external expanded mode (EXPE = 1)

(3)   PA5/$\overline{\text{RD}}$

		Setting	
		Bus Controller	I/O Port
Module Name	Pin Function	$\overline{\text{RD}}$_OE	PA5DDR
Bus controller	$\overline{\text{RD}}$ output* (initial value E)	1	—
I/O port	PA5 output	0	1
	PA5 input (initial value S)	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Note: *   Valid in external expanded mode (EXPE = 1)

(4)   PA4/$\overline{\text{LHWR}}$

Module Name	Pin Function	Setting	
		Bus Controller	I/O Port
		$\overline{\text{LHWR}}$_OE	PA4DDR
Bus controller	$\overline{\text{LHWR}}$ output* (initial value E)	1	—
I/O port	PA4 output	0	1
	PA4 input (initial value S)	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Note: *   Valid in external expanded mode (EXPE = 1)

(5)   PA3/$\overline{\text{LLWR}}$

Module Name	Pin Function	Setting	
		Bus Controller	I/O Port
		$\overline{\text{LLWR}}$_OE	PA3DDR
Bus controller	$\overline{\text{LLWR}}$ output* (initial value E)	1	—
I/O port	PA3 output	0	1
	PA3 input (initial value S)	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Note: *   Valid in external expanded mode (EXPE = 1)

RENESAS

(6)   PA2/PWM2_2/TCK

Module Name	Pin Function	Setting		
		16-Bit PWM	H-UDI	I/O Port
		PWM2_2_OE	HUDI_E	PA2DDR
16-bit PWM	PWM2_2 output	1	—	—
H-UDI	TCK input	0	1	—
I/O port	PA2 output	0	0	1
	PA2 input (initial value)	0	0	0

(7)   PA1/PWM1_2/TDI

Module Name	Pin Function	Setting		
		16-Bit PWM	H-UDI	I/O Port
		PWM1_2_OE	HUDI_E	PA1DDR
16-bit PWM	PWM1_2 output	1	—	—
H-UDI	TDI input	0	1	—
I/O port	PA1 output	0	0	1
	PA1 input (initial value)	0	0	0

(8)   PA0/PWM0_2/TDO

Module Name	Pin Function	Setting		
		16-Bit PWM	H-UDI	I/O Port
		PWM0_2_OE	HUDI_E	PA0DDR
16-bit PWM	PWM0_2 output	1	—	—
H-UDI	TDO output	0	1	—
I/O port	PA0 output	0	0	1
	PA1 input (initial value)	0	0	0

8.2.7   Port D

(1)   PD7/A7, PD6/A6, PD5/A5, PD4/A4, PD3/A3, PD2/A2, PD1/A1, PD0/A0

Module Name	Pin Function	MCU Operating Mode	Setting I/O Port PDnDDR
Bus controller	Address output	Expanded mode without on-chip ROM	—
		Expanded mode with on-chip ROM	1
I/O port	PDn output	Single-chip mode*	1
	PDn input (initial value)	Other than expanded mode without on-chip ROM	0

Notes: n = 7 to 0

*   In external expanded mode, an address can be output with PDnDDR = 1.

8.2.8   Port E

(1)   PE7/A15, PE6/A14, PE5/A13, PE4/A12, PE3/A11, PE2/A10, PE1/A9, PE0/A8

Module Name	Pin Function	MCU Operating Mode	Setting I/O Port PEnDDR
Bus controller	Address output	Expanded mode without on-chip ROM	—
		Expanded mode with on-chip ROM	1
I/O port	PEn output	Single-chip mode*	1
	PEn input (initial value)	Other than expanded mode without on-chip ROM	0

Notes: n = 7 to 0

*   In external expanded mode, an address can be output with PEnDDR = 1.

RENESAS

8.2.9 Port F

(1) PF7/A23/PWM3_1/SCK5

MCU Operating Mode	Module Name	Pin Function	Setting			
			16-Bit PWM	I/O Port	SCI	I/O Port
			PWM3_1_OE	A23_OE	SCK5_OE	PF7DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM3_1 output	1	—	—	—
	Bus controller	A23 output	0	—	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM3_1 output	1	—	—	—
	Bus controller	A23 output*	0	1	—	—
	SCI	SCK5 output	0	0	1	—
	I/O port	PF7 output	0	0	0	1
		PF7 input (initial value)	0	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

(2) PF6/A22/PWM2_1/RxD5

MCU Operating Mode	Module Name	Pin Function	Setting		
			16-Bit PWM	I/O Port	I/O Port
			PWM2_1_OE	A22_OE	PF6DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM2_1 output	1	—	—
	Bus controller	A22 output	0	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM2_1 output	1	—	—
	Bus controller	A22 output*	0	1	—
	I/O port	PF6 output	0	0	1
		PF6 input (initial value)	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

RENESAS

(3)   PF5/A21/$\overline{\text{PWM1}}$_1/TxD5

MCU Operating Mode	Module Name	Pin Function	Setting			
			16-Bit PWM	I/O Port	SCI	I/O Port
			PWM1_1_OE	A21_OE	TxD5_OE	PF5DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM1_1 output	1	—	—	—
	Bus controller	A21 output	0	—	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM1_1 output	1	—	—	—
	Bus controller	A21 output*	0	1	—	—
	SCI	TxD5	0	0	1	—
	I/O port	PF5 output	0	0	0	1
		PF5 input (initial value)	0	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

(4)   PF4/A20/PWM0_1

MCU Operating Mode	Module Name	Pin Function	Setting		
			16-Bit PWM	I/O Port	I/O Port
			PWM0_1_OE	A20_OE	PF4DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM0_1 output	1	—	—
	Bus controller	A20 output	0	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM0_1 output	1	—	—
	Bus controller	A20 output*	0	1	—
	I/O port	PF4 output	0	0	1
		PF4 input (initial value)	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

RENESAS

(5)   PF3/A19/PWM3_0

MCU Operating Mode	Module Name	Pin Function	Setting		
			16-Bit PWM	I/O Port	I/O Port
			PWM3_0_OE	A19_OE	PF3DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM3_0 output	1	—	—
	Bus controller	A19 output	0	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM3_0 output	1	—	—
	Bus controller	A19 output*	0	1	—
	I/O port	PF3 output	0	0	1
		PF3 input (initial value)	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

(6)   PF2/A18/PWM2_0

MCU Operating Mode	Module Name	Pin Function	Setting		
			16-Bit PWM	I/O Port	I/O Port
			PWM2_0_OE	A18_OE	PF2DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM2_0 output	1	—	—
	Bus controller	A18 output	0	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM2_0 output	1	—	—
	Bus controller	A18 output*	0	1	—
	I/O port	PF2 output	0	0	1
		PF2 input (initial value)	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

(7)   PF1/A17/PWM1_0

MCU Operating Mode	Module Name	Pin Function	Setting		
			16-Bit PWM	I/O Port	I/O Port
			PWM1_0_OE	A17_OE	PF1DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM1_0 output	1	—	—
	Bus controller	A17 output	0	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM1_0 output	1	—	—
	Bus controller	A17 output*	0	1	—
	I/O port	PF1 output	0	0	1
		PF1 input (initial value)	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

(8)   PF0/A16/PWM0_0

MCU Operating Mode	Module Name	Pin Function	Setting		
			16-Bit PWM	I/O Port	I/O Port
			PWM0_0_OE	A16_OE	PF0DDR
Expanded mode without on-chip ROM	16-bit PWM	PWM0_0 output	1	—	—
	Bus controller	A16 output	0	—	—
Other than expanded mode without on-chip ROM	16-bit PWM	PWM0_0 output	1	—	—
	Bus controller	A16 output*	0	1	—
	I/O port	PF0 output	0	0	1
		PF0 input (initial value)	0	0	0

Note: *   Valid in external expanded mode (EXPE = 1)

RENESAS

8.2.10   Port H

(1)   PH7/D7, PH6/D6, PH5/D5, PH4/D4, PH3/D3, PH2/D2, PH1/D1, PH0/D0

		Setting	
		Bus Controller	I/O Port
Module Name	Pin Function	16-Bit Bus Mode	PHnDDR
Bus controller	Data I/O∗ (initial value E)	1	—
I/O port	PHn output	0	1
	PHn input (initial value S)	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Notes: n = 7 to 0

　　　∗　Valid in external expanded mode (EXPE = 1)

8.2.11   Port I

(1)   PI7/D15, PI6/D14, PI5/D13, PI4/D12

		Setting	
		Bus Controller	I/O Port
Module Name	Pin Function	16-Bit Bus Mode	PInDDR
Bus controller	Data I/O∗ (initial value E)	1	—
I/O port	PIn output	0	1
	PIn input (initial value S)	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Notes: n = 7 to 4

　　　∗　Valid in external expanded mode (EXPE = 1)

(2)   PI3/D11/$\overline{\text{SCS\_0}}$

| Module Name | Pin Function | Setting | | |
| | | SSU_0*[2] | Bus Controller | I/O Port |
		$\overline{\text{SCS\_0}}$_OE	16-Bit Bus Mode	PI3DDR
SSU_0*[2]	$\overline{\text{SCS\_0}}$ output	1	—	—
Bus controller	Data I/O*[1] (initial value E)	0	1	—
I/O port	PI3 output	0	0	1
	PI3 input (initial value S)	0	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Notes: 1. Valid in external expanded mode (EXPE = 1)

2. When the SSU_0 is used, always access in 8-bit bus mode.
SSU: Synchronous Serial communication Unit

(3)   PI2/D10/SSCK_0

| Module Name | Pin Function | Setting | | |
| | | SSU_0*[2] | Bus Controller | I/O Port |
		SSCK_0_OE	16-Bit Bus Mode	PI2DDR
SSU_0*[2]	SSCK_0 output	1	—	—
Bus controller	Data I/O*[1] (initial value E)	0	1	—
I/O port	PI2 output	0	0	1
	PI2 input (initial value S)	0	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Notes: 1. Valid in external expanded mode (EXPE = 1)

2. When the SSU_0 is used, always access in 8-bit bus mode.
SSU: Synchronous Serial communication Unit

RENESAS

(4)   PI1/D9/SSI_0

Module Name	Pin Function	Setting SSU_0[2] SSI_0_OE	Bus Controller 16-Bit Bus Mode	I/O Port PI1DDR
SSU_0[2]	SSI_0 output	1	—	—
Bus controller	Data I/O[1] (initial value E)	0	1	—
I/O port	PI1 output	0	0	1
	PI1 input (initial value S)	0	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Notes: 1.   Valid in external expanded mode (EXPE = 1)

2.   When the SSU_0 is used, always access in 8-bit bus mode.
SSU: Synchronous Serial communication Unit

(5)   PI0/D8/SSO_0

Module Name	Pin Function	Setting SSU_0[2] SSO_0_OE	Bus Controller 16-Bit Bus Mode	I/O Port PI0DDR
SSU_0[2]	SSO_0 output	1	—	—
Bus controller	Data I/O[1] (initial value E)	0	1	—
I/O port	PI0 output	0	0	1
	PI0 input (initial value S)	0	0	0

Legend:

Initial value E: Initial value in external expanded mode

Initial value S: Initial value in single-chip mode

Notes: 1.   Valid in external expanded mode (EXPE = 1)

2.   When the SSU_0 is used, always access in 8-bit bus mode.
SSU: Synchronous Serial communication Unit

RENESAS

8.2.12　Port J

(1)　PJ7/PWM1H, PJ6/PWM1G, PJ5/PWM1F, PJ4/PWM1E, PJ3/PWM1D, PJ2/PWM1C, PJ1/PWM1B, PJ0/PWM1A

Module Name	Pin Function	Setting	
		PWM	I/O Port
		PWM1x_OE	PJnDDR
PWM	PWM1x output	1	—
I/O port	PJn output	0	1
	PJn input (initial value)	0	0

Notes:　x = A to H

　　　　n = 7 to 0

8.2.13　Port K

(1)　PK7/PWM2H, PK6/PWM2G, PK5/PWM2F, PK4/PWM2E, PK3/PWM2D, PK2/PWM2C, PK1/PWM2B, PK0/PWM2A

Module Name	Pin Function	Setting	
		PWM	I/O Port
		PWM2x_OE	PKnDDR
PWM	PWM2x output	1	—
I/O port	PKn output	0	1
	PKn input (initial value)	0	0

Notes:　x = A to H

　　　　n = 7 to 0

RENESAS

Table 8.5   Available Output Signals and Settings in Each Port

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P1	7	SCL0_OE	SCL0		ICCRA_0.ICE = 1
	6	$\overline{\text{DACK1}}$_A_OE	$\overline{\text{DACK1}}$_A		DACR_1.AMS = 1, DMDR_1.DACKE = 1
		SDA0_OE	SDA0		ICCRA_0.ICE = 1
	5	$\overline{\text{TEND1}}$_A_OE	$\overline{\text{TEND1}}$_A		DMDR_1.TENDE = 1
		SCL1_OE	SCL1		ICCRA_1.ICE = 1
	4	SDA1_OE	SDA1		ICCRA_1.ICE = 1
	3				
	2	$\overline{\text{DACK0}}$_A_OE	$\overline{\text{DACK0}}$_A		DACR_0.AMS = 1, DMDR_0.DACKE = 1
		SCK2_OE	SCK2		When SCMR_2.SMIF = 1: SCR_2.TE = 1 or SCR_2.RE = 1 while SMR_2.GM = 0, SCR_2.CKE[1,0] = 01 or while SMR_2.GM = 1 When SCMR_2.SMIF = 0: SCR_2.TE = 1 or SCR_2.RE = 1 while SMR_2.C/$\overline{\text{A}}$ = 0, SCR_2.CKE[1,0] = 01 or while SMR_2.C/$\overline{\text{A}}$ = 1, SCR_2.CKE1 = 0
	1	$\overline{\text{TEND0}}$_A_OE	$\overline{\text{TEND0}}$_A		DMDR_0.TENDE = 1
	0	TxD2_OE	TxD2		SCR_2.TE = 1
P2	7	TIOCB5_OE	TIOCB5		TPU.TIOR_5.IOB3 = 0, TPU.TIOR_5.IOB[1,0] = 01/10/11
	6	TIOCA5_OE	TIOCA5		TPU.TIOR_5.IOA3 = 0, TPU.TIOR_5.IOA[1,0] = 01/10/11
	5	TIOCA4_OE	TIOCA4		TPU.TIOR_4.IOA3 = 0, TPU.TIOR_4.IOA[1,0] = 01/10/11
	4	TIOCB4_OE	TIOCB4		TPU.TIOR_4.IOB3 = 0, TPU.TIOR_4.IOB[1,0] = 01/10/11
	3	$\overline{\text{SCS}}$_1_OE	$\overline{\text{SCS}}$_1		SSU.SSCRH_1.CSS1 = 1, SSU.SSCRH_1.CSS0 = 0, or SSU.SSCRH_1.CSS1 = 1, SSU.SSCRH_1.CSS0 = 1 while SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 1
		TIOCD3_OE	TIOCD3		TPU.TMDR_3.BFB = 0, TPU.TIORL_3.IOD3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
	2	SSCK1_OE	SSCK_1		SSU.SSCRH_1.MSS1 = 1, SSU.SSCRH_1.SCKS = 1
		TIOCC3_OE	TIOCC3		TPU.TMDR_3.BFA = 0, TPU.TIORL_3.IOC3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
		TxD0_OE	TxD0		SCR_0.TE = 1
	1	SSI1_OE	SSI_1		SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 0, SSU.SSCRH_1.BIDE = 0, SSU.SSER_1.TE = 1
		TIOCA3_OE	TIOCA3		TPU.TIORH_3.IOA3 = 0, TPU.TIORH_3.IOA[1,0] = 01/10/11

RENESAS

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P2	0	SSO1_OE	SSO_1		When SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 1: SSU.SSCRH_1.BIDE = 0, SSU.SSER_1.TE = 1, or SSU.SSCRH_1.BIDE = 1, SSU.SSER_1.RE = 0, SSU.SSER_1.TE = 1
					When SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 0: SSU.SSCRH_1.BIDE = 1, SSU.SSER_1.RE = 0, SSU.SSER_1.TE = 1
					When SSU.SSCRL_1.SSUMS = 1: SSU.SSER_1.TE = 1
		TIOCB3_OE	TIOCB3		TPU.TIORH_3.IOB3 = 0, TPU.TIORH_3.IOB[1,0] = 01/10/11
		SCK0_OE	SCK0		When SCMR_0.SMIF = 1: SCR_0.TE = 1 or SCR_0.RE = 1 while SMR_0.GM = 0, SCR_0.CKE[1,0] = 01 or while SMR_0.GM = 1
					When SCMR_0.SMIF = 0: SCR_0.TE = 1 or SCR_0.RE = 1 while SMR_0.C/$\overline{\text{A}}$ = 0, SCR_0.CKE[1,0] = 01 or while SMR_0.C/$\overline{\text{A}}$ = 1, SCR_0.CKE1 = 0
P3	7	SGOUT3_OE	SGOUT3		SDG3.SGCR1.SGE = 1
		TIOCB2_OE	TIOCB2		TPU.TIOR_2.IOB3 = 0, TPU.TIOR_2.IOB[1,0] = 01/10/11
	6	SGOUT2_OE	SGOUT2		SDG2.SGCR1.SGE = 1
		TIOCA2_OE	TIOCA2		TPU.TIOR_2.IOA3 = 0, TPU.TIOR_2.IOA[1,0] = 01/10/11
	5	SGOUT1_OE	SGOUT1		SDG1.SGCR1.SGE = 1
		TIOCB1_OE	TIOCB1		TPU.TIOR_1.IOB3 = 0, TPU.TIOR_1.IOB[1,0] = 01/10/11
	4	SGOUT0_OE	SGOUT0		SDG0.SGCR1.SGE = 1
		TIOCA1_OE	TIOCA1		TPU.TIOR_1.IOA3 = 0, TPU.TIOR_1.IOA[1,0] = 01/10/11
	3	TIOCD0_OE	TIOCD0		TPU.TMDR_0.BFB = 0, TPU.TIORL_0.IOD3 = 0, TPU.TIORL_0.IOD[1,0] = 01/10/11
	2	TIOCC0_OE	TIOCC0		TPU.TMDR_0.BFA = 0, TPU.TIORL_0.IOC3 = 0, TPU.TIORL_0.IOD[1,0] = 01/10/11
	1	TIOCB0_OE	TIOCB0		TPU.TIORH_0.IOB3 = 0, TPU.TIORH_0.IOB[1,0] = 01/10/11
	0	TIOCA0_OE	TIOCA0		TPU.TIORH_0.IOA3 = 0, TPU.TIORH_0.IOA[1,0] = 01/10/11
P4					
P5					

RENESAS

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P6	7	CTx_1_OE	CTx_1		RCAN-ET_1.MBCR.MBCRn = 0, RCAN-ET_1.TXRP.TXRn = 1 while RCAN-ET_1.RCANMON.RCANE = 1, RCAN-ET_1.RCANMON.TxSTP = 0 (n = 1 to 15)
	6				
	5	CTx_0_OE	CTx_0		RCAN-ET_0.MBCR.MBCRn = 0, RCAN-ET_0.TXRP.TXRn = 1 while RCAN-ET_0.RCANMON.RCANE = 1, RCAN-ET_0.RCANMON.TxSTP = 0 (n = 1 to 15)
		$\overline{DACK3}$_B_OE	$\overline{DACK3}$_B		DACR_3.AMS = 1, DMDR_3.DACKE = 1
	4	$\overline{TEND3}$_B_OE	$\overline{TEND3}$_B		DMDR_3.TENDE = 1
	3	PWM3_2_OE	PWM3_2		PWOCR3.OE32 = 1
		HUDI_E	TMS		PSELR2.PGSEL1 = 1
	2	$\overline{DACK2}$_B_OE	$\overline{DACK2}$_B		DACR_2.AMS = 1, DMDR_2.DACKE = 1
		SCK4_OE	SCK4		When SCMR_4.SMIF = 1: SCR_4.TE = 1 or SCR_4.RE = 1 while SMR_4.GM = 0, SCR_4.CKE[1,0] = 01 or while SMR_4.GM = 1
					When SCMR_4.SMIF = 0: SCR_4.TE = 1 or SCR_4.RE = 1 while SMR_4.C/$\overline{A}$ = 0, SCR_4.CKE[1,0] = 01 or while SMR_4.C/$\overline{A}$ = 1, SCR_4.CKE1 = 0
		HUDI_E	$\overline{TRST}$		PSELR2.PGSEL1 = 1
	1	$\overline{TEND2}$_B_OE	$\overline{TEND2}$_B		DMDR_2.TENDE = 1
	0	TxD4_OE	TxD4		SCR_4.TE = 1
PA	7	Bϕ_OE	Bϕ		PADDR.PA7DDR = 1, SCKCR.POSEL1 = 0
	6	$\overline{AS}$_OE	$\overline{AS}$		SYSCR.EXPE = 1, PFCR2.ASOE = 1
	5	$\overline{RD}$_OE	$\overline{RD}$		SYSCR.EXPE = 1
	4	$\overline{LHWR}$_OE	$\overline{LHWR}$		SYSCR.EXPE = 1
	3	$\overline{LLWR}$_OE	$\overline{LLWR}$		SYSCR.EXPE = 1
	2	PWM2_2_OE	PWM2_2		PWOCR3.OE22 = 1
		HUDI_E	TCK		PSELR2.PGSEL1 = 1
	1	PWM1_2_OE	PWM1_2		PWOCR3.OE12 = 1
		HUDI_E	TDI		PSELR2.PGSEL1 = 1
	0	PWM0_2_OE	PWM0_2		PWOCR3.OE02 = 1
		HUDI_E	TDO		PSELR2.PGSEL1 = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PD	7	A7_OE	A7		SYSCR.EXPE = 1, PDDDR.PD7DDR = 1
	6	A6_OE	A6		SYSCR.EXPE = 1, PDDDR.PD6DDR = 1
	5	A5_OE	A5		SYSCR.EXPE = 1, PDDDR.PD5DDR = 1
	4	A4_OE	A4		SYSCR.EXPE = 1, PDDDR.PD4DDR = 1
	3	A3_OE	A3		SYSCR.EXPE = 1, PDDDR.PD3DDR = 1
	2	A2_OE	A2		SYSCR.EXPE = 1, PDDDR.PD2DDR = 1
	1	A1_OE	A1		SYSCR.EXPE = 1, PDDDR.PD1DDR = 1
	0	A0_OE	A0		SYSCR.EXPE = 1, PDDDR.PD0DDR = 1
PE	7	A15_OE	A15		SYSCR.EXPE = 1, PEDDR.PE7DDR = 1
	6	A14_OE	A14		SYSCR.EXPE = 1, PEDDR.PE6DDR = 1
	5	A13_OE	A13		SYSCR.EXPE = 1, PEDDR.PE5DDR = 1
	4	A12_OE	A12		SYSCR.EXPE = 1, PEDDR.PE4DDR = 1
	3	A11_OE	A11		SYSCR.EXPE = 1, PEDDR.PE3DDR = 1
	2	A10_OE	A10		SYSCR.EXPE = 1, PEDDR.PE2DDR = 1
	1	A9_OE	A9		SYSCR.EXPE = 1, PEDDR.PE1DDR = 1
	0	A8_OE	A8		SYSCR.EXPE = 1, PEDDR.PE0DDR = 1
PF	7	PWM3_1_OE	PWM3_1		PWOCR2.OE31 = 1
		A23_OE	A23		SYSCR.EXPE = 1, PFCR4.A23E = 1
		SCK5_OE	SCK5		When SCMR_5.SMIF = 1: SCR_5.TE = 1 or SCR_5.RE = 1 while SMR_5.GM = 0, SCR_5.CKE[1, 0] = 01, or SMR_5.GM = 1 When SCMR_5.SMIF = 0: SCR_5.TE = 1 or SCR_5.RE = 1 while SMR_5.C/$\overline{A}$ = 0, SCR_5.CKE[1, 0] = 01 or SMR_5.C/$\overline{A}$ = 1, SCR_5.CKE1 = 0
	6	PWM2_1_OE	PWM2_1		PWOCR2.OE21 = 1
		A22_OE	A22		SYSCR.EXPE = 1, PFCR4.A22E = 1
	5	PWM1_1_OE	PWM1_1		PWOCR2.OE11 = 1
		A21_OE	A21		SYSCR.EXPE = 1, PFCR4.A21E = 1
		TxD5_OE	TxD5		SCR_5.TE = 1
	4	PWM0_1_OE	PWM0_1		PWOCR2.OE01 = 1
		A20_OE	A20		SYSCR.EXPE = 1, PFCR4.A20E = 1
	3	PWM3_0_OE	PWM3_0		PWOCR1.OE30 = 1
		A19_OE	A19		SYSCR.EXPE = 1, PFCR4.A19E = 1

RENESAS

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PF	2	PWM2_0_OE	PWM2_0		PWOCR1.OE20 = 1
		A18_OE	A18		SYSCR.EXPE = 1, PFCR4.A18E = 1
	1	PWM1_0_OE	PWM1_0		PWOCR1.OE10 = 1
		A17_OE	A17		SYSCR.EXPE = 1, PFCR4.A17E = 1
	0	PWM0_0_OE	PWM0_0		PWOCR1.OE00 = 1
		A16_OE	A16		SYSCR.EXPE = 1, PFCR4.A16E = 1
PH	7	D7_E	D7		SYSCR.EXPE = 1
	6	D6_E	D6		SYSCR.EXPE = 1
	5	D5_E	D5		SYSCR.EXPE = 1
	4	D4_E	D4		SYSCR.EXPE = 1
	3	D3_E	D3		SYSCR.EXPE = 1
	2	D2_E	D2		SYSCR.EXPE = 1
	1	D1_E	D1		SYSCR.EXPE = 1
	0	D0_E	D0		SYSCR.EXPE = 1
PI	7	D15_E	D15		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	6	D14_E	D14		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	5	D13_E	D13		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	4	D12_E	D12		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	3	$\overline{SCS0}$_OE	$\overline{SCS0}$		When ABWCR.ABW[H,L]n = 11: SSU.SSCRH_0.CSS1 = 1, SSU.SSCRH_0.CSS0 = 0, or SSU.SSCRH_0.CSS1 = 1, SSU.SSCRH_0.CSS0 = 1 while SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 1
		D11_E	D11		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	2	SSCK0_OE	SSCK0		When ABWCR.ABW[H,L]n = 11: SSU.SSCRH_0.MSS1 = 1, SSU.SSCRH_0.SCKS = 1
		D10_E	D10		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
	1	SSI_0_OE	SSI0		When ABWCR.ABW[H,L]n = 11: SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 0, SSU.SSCRH_0.BIDE = 0, SSU.SSER_0.TE = 1
		D9_E	D9		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01

RENESAS

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PI	0	SSO_0_OE	SSO0		When ABWCR.ABW[H,L]n = 11: When SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 1: SSU.SSCRH_0.BIDE = 0, SSU.SSER_0.TE = 1, or SSU.SSCRH_0.BIDE = 1, SSU.SSER_0.RE = 0, SSU.SSER_0.TE = 1
					When SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 0: SSU.SSCRH_0.BIDE = 1, SSU.SSER_0.RE = 0, SSU.SSER_0.TE = 1
					When SSU.SSCRL_0.SSUMS = 1: SSU.SSER_0.TE = 1
		D8_E	D8		SYSCR.EXPE = 1, ABWCR.ABW[H,L]n = 01
PJ	7	PWM1H_OE	PWM1H		PWOCR1.OE1H = 1
	6	PWM1G_OE	PWM1G		PWOCR1.OE1G = 1
	5	PWM1F_OE	PWM1F		PWOCR1.OE1F = 1
	4	PWM1E_OE	PWM1E		PWOCR1.OE1E = 1
	3	PWM1D_OE	PWM1D		PWOCR1.OE1D = 1
	2	PWM1C_OE	PWM1C		PWOCR1.OE1C = 1
	1	PWM1B_OE	PWM1B		PWOCR1.OE1B = 1
	0	PWM1A_OE	PWM1A		PWOCR1.OE1A = 1
PK	7	PWM2H_OE	PWM2H		PWOCR2.OE2H = 1
	6	PWM2G_OE	PWM2G		PWOCR2.OE2G = 1
	5	PWM2F_OE	PWM2F		PWOCR2.OE2F = 1
	4	PWM2E_OE	PWM2E		PWOCR2.OE2E = 1
	3	PWM2D_OE	PWM2D		PWOCR2.OE2D = 1
	2	PWM2C_OE	PWM2C		PWOCR2.OE2C = 1
	1	PWM2B_OE	PWM2B		PWOCR2.OE2B = 1
	0	PWM2A_OE	PWM2A		PWOCR2.OE2A = 1

Note: SSU: Synchronous Serial communication Unit

RENESAS

8.3 Port Function Controller

The port function controller incorporates the following registers.

- Port function control register 2 (PFCR2)
- Port function control register 4 (PFCR4)
- Port function control register 9 (PFCR9)

8.3.1 Port Function Control Register 2 (PFCR2)

PFCR2 enables/disables the bus control output.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
1	ASOE	1	R/W	$\overline{AS}$ Output Enable Enables/disables the $\overline{AS}$ output 0: Specifies pin PA6 as I/O port 1: Specifies pin PA6 as $\overline{AS}$ output pin
0	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.

RENESAS

8.3.2 Port Function Control Register 4 (PFCR4)

PFCR4 enables/disables the address output.

Bit	7	6	5	4	3	2	1	0
Bit Name	A23E	A22E	A21E	A20E	A19E	A18E	A17E	A16E
Initial Value	Undefined*	Undefined*	Undefined*	Undefined*	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	A23E	Undefined*	R/W	Address A23 Enable Enables/disables the address output (A23) 0: Disables the A23 output 1: Enables the A23 output
6	A22E	Undefined*	R/W	Address A22 Enable Enables/disables the address output (A22) 0: Disables the A22 output 1: Enables the A22 output
5	A21E	Undefined*	R/W	Address A21 Enable Enables/disables the address output (A21) 0: Disables the A21 output 1: Enables the A21 output
4	A20E	Undefined*	R/W	Address A20 Enable Enables/disables the address output (A20) 0: Disables the A20 output 1: Enables the A20 output

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	A19E	Undefined*	R/W	Address A19 Enable
				Enables/disables the address output (A19)
				0: Disables the A19 output
				1: Enables the A19 output
2	A18E	Undefined*	R/W	Address A18 Enable
				Enables/disables the address output (A18)
				0: Disables the A18 output
				1: Enables the A18 output
1	A17E	Undefined*	R/W	Address A17 Enable
				Enables/disables the address output (A17)
				0: Disables the A17 output
				1: Enables the A17 output
0	A16E	Undefined*	R/W	Address A16 Enable
				Enables/disables the address output (A16)
				0: Disables the A16 output
				1: Enables the A16 output

Note: *   The initial value depends on the operating mode.

8.3.3   Port Function Control Register 9 (PFCR9)

PFCR9 selects the multiple functions for the TPU (unit 0) I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	TPUMS5	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA5 function 0: Specifies pin P26 as output compare output and input capture 1: Specifies P27 as input capture input and P26 as output compare
6	TPUMS4	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA4 function 0: Specifies P25 as output compare output and input capture 1: Specifies P24 as input capture input and P25 as output compare
5	TPUMS3A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA3 function 0: Specifies P21 as output compare output and input capture 1: Specifies P20 as input capture input and P21 as output compare
4	TPUMS3B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC3 function 0: Specifies P22 as output compare output and input capture 1: Specifies P23 as input capture input and P22 as output compare

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	TPUMS2	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA2 function 0: Specifies P36 as output compare output and input capture 1: Specifies P37 as input capture input and P36 as output compare
2	TPUMS1	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA1 function 0: Specifies P34 as output compare output and input capture 1: Specifies P35 as input capture input and P34 as output compare
1	TPUMS0A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA0 function 0: Specifies P30 as output compare output and input capture 1: Specifies P31 as input capture input and P30 as output compare
0	TPUMS0B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC0 function 0: Specifies P32 as output compare output and input capture 1: Specifies P33 as input capture input and P32 as output compare

8.4 Usage Notes

8.4.1 Notes on Input Buffer Control Register (ICR) Setting

- When the ICR setting is changed, the LSI may malfunction due to an edge occurred internally according to the pin state. Before changing the ICR setting, fix the pin state high or disable the input function corresponding to the pin by the on-chip peripheral module settings.

- If an input is enabled by setting ICR while multiple input functions are assigned to the pin, the pin state is reflected in all the inputs. Care must be taken for each module settings for unused input functions.

- When a pin is used as an output, data to be output from the pin will be latched as the pin state if the input function corresponding to the pin is enabled. To use the pin as an output, disable the input function for the pin by setting ICR.

8.4.2 Notes on Port Function Control Register (PFCR) Settings

- Port function controller controls the I/O port.

  Before enabling a port function, select the input/output destination.

- When changing input pins, this LSI may malfunction due to the internal edge.

  To change input pins, the following procedure must be performed.

  1. Disable the input function by the corresponding on-chip peripheral module settings
  2. Select another input pin by PFCR
  3. Enable its input function by the corresponding on-chip peripheral module settings

- If a pin function has both a select bit that modifies the input/output destination and an enable bit that enables the pin function, first specify the input/output destination by the selection bit and then enable the pin function by the enable bit.

RENESAS

Section 9   16-Bit Timer Pulse Unit (TPU)

This LSI has an on-chip 16-bit timer pulse unit (TPU) that comprises six 16-bit timer channels.

Table 9.1 lists the 16-bit timer unit functions and figure 9.1 is a block diagram.

9.1   Features

- Maximum 16-pulse input/output
- Selection of eight counter input clocks for each channel
- The following operations can be set for each channel:
  — Waveform output at compare match
  — Input capture function
  — Counter clear operation
  — Synchronous operations:
    - Multiple timer counters (TCNT) can be written to simultaneously
    - Simultaneous clearing by compare match and input capture possible
    - Simultaneous input/output for registers possible by counter synchronous operation
    - Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
- Cascaded operation
- Fast access via internal 16-bit bus
- 26 interrupt sources
- Automatic transfer of register data
- Conversion start trigger for the A/D converter can be generated
- Module stop mode can be set

RENESAS

Table 9.1   TPU Functions

Item		Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
Count clock		Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 TCLKA TCLKB TCLKC TCLKD	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKA TCLKB TCNT2	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKA TCLKB TCLKC	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 Pϕ/1024 Pϕ/4096 TCLKA	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKA TCLKC TCNT5	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKA TCLKC TCLKD
General registers (TGR)		TGRA_0 TGRB_0	TGRA_1 TGRB_1	TGRA_2 TGRB_2	TGRA_3 TGRB_3	TGRA_4 TGRB_4	TGRA_5 TGRB_5
General registers/ buffer registers		TGRC_0 TGRD_0	—	—	TGRC_3 TGRD_3	—	—
I/O pins		TIOCA0 TIOCB0 TIOCC0 TIOCD0	TIOCA1 TIOCB1	TIOCA2 TIOCB2	TIOCA3 TIOCB3 TIOCC3 TIOCD3	TIOCA4 TIOCB4	TIOCA5 TIOCB5
Counter clear function		TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture
Compare match output	0 output	O	O	O	O	O	O
	1 output	O	O	O	O	O	O
	Toggle output	O	O	O	O	O	O
Input capture function		O	O	O	O	O	O
Synchronous operation		O	O	O	O	O	O
PWM mode		O	O	O	O	O	O
Phase counting mode		—	O	O	—	O	O
Buffer operation		O	—	—	O	—	—
DMAC activation		TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture	TGRA_5 compare match or input capture

RENESAS

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
A/D converter trigger	TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture	TGRA_5 compare match or input capture
Interrupt sources	5 sources	4 sources	4 sources	5 sources	4 sources	4 sources
	•Compare match or input capture 0A	•Compare match or input capture 1A	•Compare match or input capture 2A	•Compare match or input capture 3A	•Compare match or input capture 4A	•Compare match or input capture 5A
	•Compare match or input capture 0B	•Compare match or input capture 1B	•Compare match or input capture 2B	•Compare match or input capture 3B	•Compare match or input capture 4B	•Compare match or input capture 5B
	•Compare match or input capture 0C	•Overflow	•Overflow	•Compare match or input capture 3C	•Overflow	•Overflow
	•Compare match or input capture 0D	•Underflow	•Underflow	•Compare match or input capture 3D	•Underflow	•Underflow
	•Overflow			•Overflow		

Legend:

O : Possible

— : Not possible
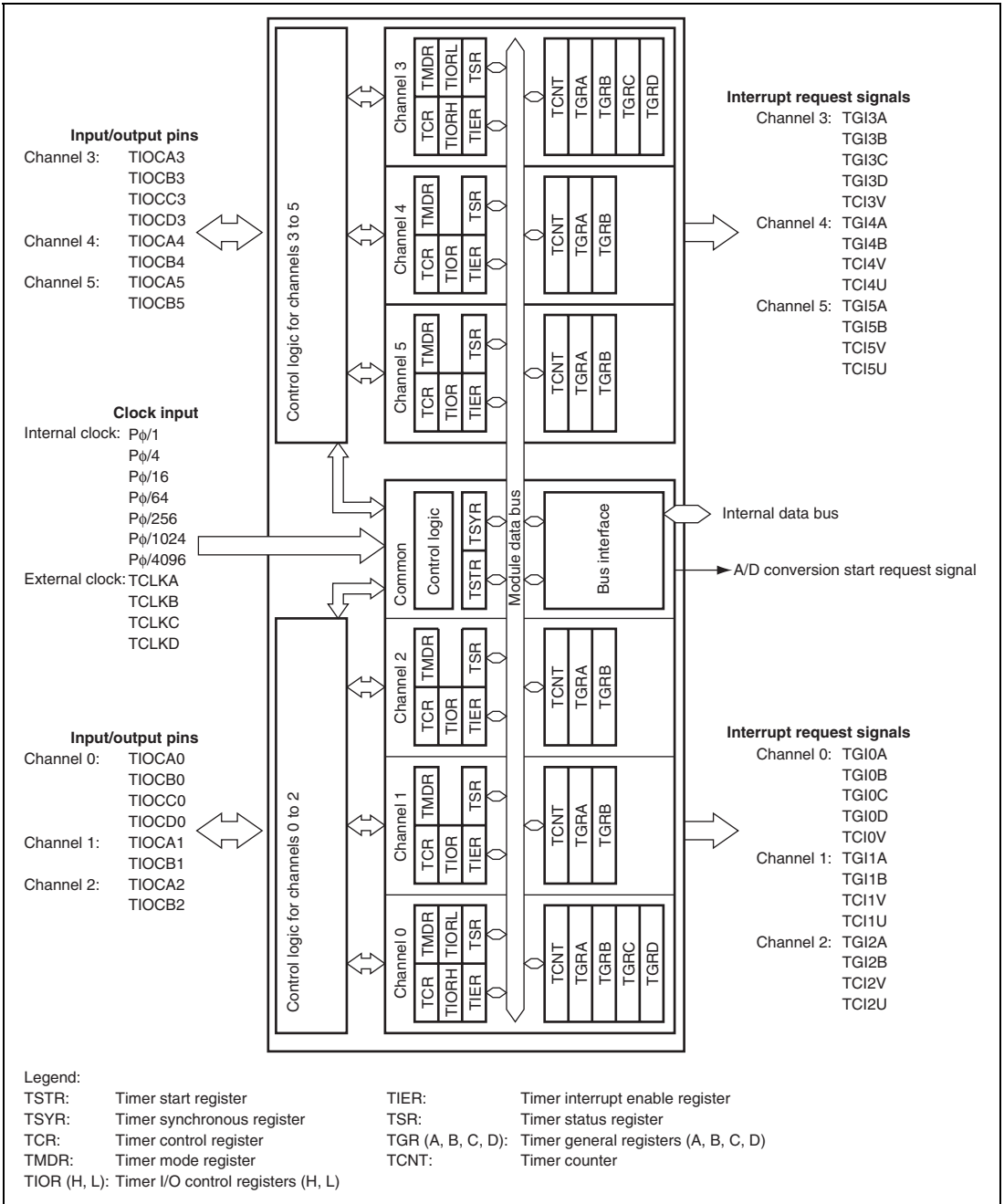
Figure 9.1   Block Diagram of TPU

9.2   Input/Output Pins

Table 9.2 shows TPU pin configurations.

Table 9.2   Pin Configuration

Channel	Symbol	I/O	Function
All	TCLKA	Input	External clock A input pin
			(Channel 1 and 5 phase counting mode A phase input)
	TCLKB	Input	External clock B input pin
			(Channel 1 and 5 phase counting mode B phase input)
	TCLKC	Input	External clock C input pin
			(Channel 2 and 4 phase counting mode A phase input)
	TCLKD	Input	External clock D input pin
			(Channel 2 and 4 phase counting mode B phase input)
0	TIOCA0	I/O	TGRA_0 input capture input/output compare output/PWM output pin
	TIOCB0	I/O	TGRB_0 input capture input/output compare output/PWM output pin
	TIOCC0	I/O	TGRC_0 input capture input/output compare output/PWM output pin
	TIOCD0	I/O	TGRD_0 input capture input/output compare output/PWM output pin
1	TIOCA1	I/O	TGRA_1 input capture input/output compare output/PWM output pin
	TIOCB1	I/O	TGRB_1 input capture input/output compare output/PWM output pin
2	TIOCA2	I/O	TGRA_2 input capture input/output compare output/PWM output pin
	TIOCB2	I/O	TGRB_2 input capture input/output compare output/PWM output pin
3	TIOCA3	I/O	TGRA_3 input capture input/output compare output/PWM output pin
	TIOCB3	I/O	TGRB_3 input capture input/output compare output/PWM output pin
	TIOCC3	I/O	TGRC_3 input capture input/output compare output/PWM output pin
	TIOCD3	I/O	TGRD_3 input capture input/output compare output/PWM output pin
4	TIOCA4	I/O	TGRA_4 input capture input/output compare output/PWM output pin
	TIOCB4	I/O	TGRB_4 input capture input/output compare output/PWM output pin
5	TIOCA5	I/O	TGRA_5 input capture input/output compare output/PWM output pin
	TIOCB5	I/O	TGRB_5 input capture input/output compare output/PWM output pin

9.3 Register Descriptions

The TPU has the following registers in each channel.

- Channel 0
  — Timer control register_0 (TCR_0)
  — Timer mode register_0 (TMDR_0)
  — Timer I/O control register H_0 (TIORH_0)
  — Timer I/O control register L_0 (TIORL_0)
  — Timer interrupt enable register_0 (TIER_0)
  — Timer status register_0 (TSR_0)
  — Timer counter_0 (TCNT_0)
  — Timer general register A_0 (TGRA_0)
  — Timer general register B_0 (TGRB_0)
  — Timer general register C_0 (TGRC_0)
  — Timer general register D_0 (TGRD_0)
- Channel 1
  — Timer control register_1 (TCR_1)
  — Timer mode register_1 (TMDR_1)
  — Timer I/O control register _1 (TIOR_1)
  — Timer interrupt enable register_1 (TIER_1)
  — Timer status register_1 (TSR_1)
  — Timer counter_1 (TCNT_1)
  — Timer general register A_1 (TGRA_1)
  — Timer general register B_1 (TGRB_1)
- Channel 2
  — Timer control register_2 (TCR_2)
  — Timer mode register_2 (TMDR_2)
  — Timer I/O control register_2 (TIOR_2)
  — Timer interrupt enable register_2 (TIER_2)
  — Timer status register_2 (TSR_2)
  — Timer counter_2 (TCNT_2)
  — Timer general register A_2 (TGRA_2)
  — Timer general register B_2 (TGRB_2)

RENESAS

- Channel 3
  - — Timer control register_3 (TCR_3)
  - — Timer mode register_3 (TMDR_3)
  - — Timer I/O control register H_3 (TIORH_3)
  - — Timer I/O control register L_3 (TIORL_3)
  - — Timer interrupt enable register_3 (TIER_3)
  - — Timer status register_3 (TSR_3)
  - — Timer counter_3 (TCNT_3)
  - — Timer general register A_3 (TGRA_3)
  - — Timer general register B_3 (TGRB_3)
  - — Timer general register C_3 (TGRC_3)
  - — Timer general register D_3 (TGRD_3)
- Channel 4
  - — Timer control register_4 (TCR_4)
  - — Timer mode register_4 (TMDR_4)
  - — Timer I/O control register _4 (TIOR_4)
  - — Timer interrupt enable register_4 (TIER_4)
  - — Timer status register_4 (TSR_4)
  - — Timer counter_4 (TCNT_4)
  - — Timer general register A_4 (TGRA_4)
  - — Timer general register B_4 (TGRB_4)
- Channel 5
  - — Timer control register_5 (TCR_5)
  - — Timer mode register_5 (TMDR_5)
  - — Timer I/O control register_5 (TIOR_5)
  - — Timer interrupt enable register_5 (TIER_5)
  - — Timer status register_5 (TSR_5)
  - — Timer counter_5 (TCNT_5)
  - — Timer general register A_5 (TGRA_5)
  - — Timer general register B_5 (TGRB_5)
- Common Registers
  - — Timer start register (TSTR)
  - — Timer synchronous register (TSYR)

9.3.1 Timer Control Register (TCR)

TCR controls the TCNT operation for each channel. The TPU has a total of six TCR registers, one for each channel. TCR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	CCLR2	0	R/W	Counter Clear 2 to 0
6	CCLR1	0	R/W	These bits select the TCNT counter clearing source. See tables 9.3 and 9.4 for details.
5	CCLR0	0	R/W	
4	CKEG1	0	R/W	Clock Edge 1 and 0
3	CKEG0	0	R/W	These bits select the input clock edge. For details, see table 9.5. When the input clock is counted using both edges, the input clock period is halved (e.g. Pϕ/4 both edges = Pϕ/2 rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority. Internal clock edge selection is valid when the input clock is Pϕ/4 or slower. This setting is ignored if the input clock is Pϕ/1, or when overflow/underflow of another channel is selected.
2	TPSC2	0	R/W	Timer Prescaler 2 to 0
1	TPSC1	0	R/W	These bits select the TCNT counter clock. The clock source can be selected independently for each channel. See tables 9.6 to 9.11 for details. To select the external clock as the clock source, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 8, I/O Ports.
0	TPSC0	0	R/W	

RENESAS

Table 9.3   CCLR2 to CCLR0 (Channels 0 and 3)

Channel	Bit 7 CCLR2	Bit 6 CCLR1	Bit 5 CCLR0	Description
0, 3	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]
	1	0	0	TCNT clearing disabled
	1	0	1	TCNT cleared by TGRC compare match/input capture*[2]
	1	1	0	TCNT cleared by TGRD compare match/input capture*[2]
	1	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]

Notes: 1.   Synchronous operation is selected by setting the SYNC bit in TSYR to 1.

2.   When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

Table 9.4   CCLR2 to CCLR0 (Channels 1, 2, 4, and 5)

Channel	Bit 7 Reserved[2]	Bit 6 CCLR1	Bit 5 CCLR0	Description
1, 2, 4, 5	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation[1]

Notes: 1. Synchronous operation is selected by setting the SYNC bit in TSYR to 1.
2. Bit 7 is reserved in channels 1, 2, 4, and 5. It is always read as 0 and cannot be modified.

Table 9.5   Input Clock Edge Selection

Clock Edge Selection		Input Clock	
CKEG1	CKEG0	Internal Clock	External Clock
0	0	Counted at falling edge	Counted at rising edge
0	1	Counted at rising edge	Counted at falling edge
1	×	Counted at both edges	Counted at both edges

Legend:

×: Don't care

RENESAS

Table 9.6   TPSC2 to TPSC0 (Channel 0)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
0	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	External clock: counts on TCLKD pin input

Table 9.7   TPSC2 to TPSC0 (Channel 1)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
1	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	Counts on TCNT2 overflow/underflow

Note: This setting is ignored when channel 1 is in phase counting mode.

Table 9.8   TPSC2 to TPSC0 (Channel 2)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
2	0	0	0	Internal clock: counts on Pφ/1
	0	0	1	Internal clock: counts on Pφ/4
	0	1	0	Internal clock: counts on Pφ/16
	0	1	1	Internal clock: counts on Pφ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	Internal clock: counts on Pφ/1024

Note: This setting is ignored when channel 2 is in phase counting mode.

Table 9.9   TPSC2 to TPSC0 (Channel 3)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
3	0	0	0	Internal clock: counts on Pφ/1
	0	0	1	Internal clock: counts on Pφ/4
	0	1	0	Internal clock: counts on Pφ/16
	0	1	1	Internal clock: counts on Pφ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	Internal clock: counts on Pφ/1024
	1	1	0	Internal clock: counts on Pφ/256
	1	1	1	Internal clock: counts on Pφ/4096

RENESAS

Table 9.10   TPSC2 to TPSC0 (Channel 4)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
4	0	0	0	Internal clock: counts on Pφ/1
	0	0	1	Internal clock: counts on Pφ/4
	0	1	0	Internal clock: counts on Pφ/16
	0	1	1	Internal clock: counts on Pφ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on Pφ/1024
	1	1	1	Counts on TCNT5 overflow/underflow

Note: This setting is ignored when channel 4 is in phase counting mode.

Table 9.11   TPSC2 to TPSC0 (Channel 5)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
5	0	0	0	Internal clock: counts on Pφ/1
	0	0	1	Internal clock: counts on Pφ/4
	0	1	0	Internal clock: counts on Pφ/16
	0	1	1	Internal clock: counts on Pφ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on Pφ/256
	1	1	1	External clock: counts on TCLKD pin input

Note: This setting is ignored when channel 5 is in phase counting mode.

RENESAS

9.3.2   Timer Mode Register (TMDR)

TMDR sets the operating mode for each channel. The TPU has six TMDR registers, one for each channel. TMDR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	BFB	BFA	MD3	MD2	MD1	MD0
Initial Value	1	1	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	⸺	All 1	R	Reserved
				These are read-only bits and cannot be modified.
5	BFB	0	R/W	Buffer Operation B
				Specifies whether TGRB is to normally operate, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD input capture/output compare is not generated.
				In channels 1, 2, 4, and 5, which have no TGRD, bit 5 is reserved. It is always read as 0 and cannot be modified.
				0: TGRB operates normally
				1: TGRB and TGRD used together for buffer operation
4	BFA	0	R/W	Buffer Operation A
				Specifies whether TGRA is to normally operate, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated.
				In channels 1, 2, 4, and 5, which have no TGRC, bit 4 is reserved. It is always read as 0 and cannot be modified.
				0: TGRA operates normally
				1: TGRA and TGRC used together for buffer operation
3	MD3	0	R/W	Modes 3 to 0
2	MD2	0	R/W	Set the timer operating mode.
1	MD1	0	R/W	MD3 is a reserved bit. The write value should always be 0. See table 9.12 for details.
0	MD0	0	R/W	

RENESAS

Table 9.12   MD3 to MD0

Bit 3 MD3*[1]	Bit 2 MD2*[2]	Bit 1 MD1	Bit 0 MD0	Description
0	0	0	0	Normal operation
0	0	0	1	Reserved
0	0	1	0	PWM mode 1
0	0	1	1	PWM mode 2
0	1	0	0	Phase counting mode 1
0	1	0	1	Phase counting mode 2
0	1	1	0	Phase counting mode 3
0	1	1	1	Phase counting mode 4
1	×	×	×	—

Legend:

×: Don't care

Notes: 1. MD3 is a reserved bit. The write value should always be 0.

2. Phase counting mode cannot be set for channels 0 and 3. In this case, 0 should always be written to MD2.

9.3.3   Timer I/O Control Register (TIOR)

TIOR controls TGR. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. Care is required since TIOR is affected by the TMDR setting.

The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified.

When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

To designate the input capture pin in TIOR, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 8, I/O Ports.

RENESAS

- TIORH_0, TIOR_1, TIOR_2, TIORH_3, TIOR_4, TIOR_5

Bit	7	6	5	4	3	2	1	0
Bit Name	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- TIORL_0, TIORL_3

Bit	7	6	5	4	3	2	1	0
Bit Name	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- TIORH_0, TIOR_1, TIOR_2, TIORH_3, TIOR_4, TIOR_5

Bit	Bit Name	Initial Value	R/W	Description
7	IOB3	0	R/W	I/O Control B3 to B0
6	IOB2	0	R/W	Specify the function of TGRB.
5	IOB1	0	R/W	For details, see tables 9.13, 9.15, 9.16, 9.17, 9.19, and
4	IOB0	0	R/W	9.20.
3	IOA3	0	R/W	I/O Control A3 to A0
2	IOA2	0	R/W	Specify the function of TGRA.
1	IOA1	0	R/W	For details, see tables 9.21, 9.23, 9.24, 9.25, 9.27, and
0	IOA0	0	R/W	9.28.

RENESAS

• TIORL_0, TIORL_3

Bit	Bit Name	Initial Value	R/W	Description
7	IOD3	0	R/W	I/O Control D3 to D0
6	IOD2	0	R/W	Specify the function of TGRD.
5	IOD1	0	R/W	For details, see tables 9.14 and 9.18.
4	IOD0	0	R/W	
3	IOC3	0	R/W	I/O Control C3 to C0
2	IOC2	0	R/W	Specify the function of TGRC.
1	IOC1	0	R/W	For details, see tables 9.22 and 9.26.
0	IOC0	0	R/W	

RENESAS

Table 9.13   TIORH_0

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description	
				TGRB_0 Function	TIOCB0 Pin Function
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB0 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCB0 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCB0 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 1/count clock
					Input capture at TCNT_1 count-up/count-down*

Legend:

×: Don't care

Note: When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pφ/1 is used as the TCNT_1 count clock, this setting is invalid and input capture is not generated.

RENESAS

Table 9.14   TIORL_0

Bit 7 IOD3	Bit 6 IOD2	Bit 5 IOD1	Bit 4 IOD0	Description	
				TGRD_0 Function	**TIOCD0 Pin Function**
0	0	0	0	Output compare register*[2]	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register*[2]	Capture input source is TIOCD0 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCD0 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCD0 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 1/count clock
					Input capture at TCNT_1 count-up/count-down*[1]

Legend:

×: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR_0 is set to 1 and TGRD_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

RENESAS

Table 9.15   TIOR_1

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description TGRB_1 Function	TIOCB1 Pin Function
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB1 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCB1 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCB1 pin
					Input capture at both edges
1	1	×	×		TGRC_0 compare match/input capture
					Input capture at generation of TGRC_0 compare match/input capture

Legend:

×: Don't care

RENESAS

Table 9.16   TIOR_2

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description	
				TGRB_2 Function	TIOCB2 Pin Function
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	×	0	0	Input capture register	Capture input source is TIOCB2 pin
					Input capture at rising edge
1	×	0	1		Capture input source is TIOCB2 pin
					Input capture at falling edge
1	×	1	×		Capture input source is TIOCB2 pin
					Input capture at both edges

Legend:

×: Don't care

RENESAS

Table 9.17   TIORH_3

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_3 Function**	**TIOCB3 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB3 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCB3 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCB3 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 4/count clock
					Input capture at TCNT_4 count-up/count-down*

Legend:

×: Don't care

Note: When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pφ/1 is used as the TCNT_4 count clock, this setting is invalid and input capture is not generated.

RENESAS

Table 9.18   TIORL_3

Bit 7 IOD3	Bit 6 IOD2	Bit 5 IOD1	Bit 4 IOD0	Description	
				TGRD_3 Function	TIOCD3 Pin Function
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCD3 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCD3 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCD3 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 4/count clock
					Input capture at TCNT_4 count-up/count-down*1

Legend:

×: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pφ/1 is used as the TCNT_4 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR_3 is set to 1 and TGRD_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 9.19   TIOR_4

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description	
				TGRB_4 Function	**TIOCB4 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB4 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCB4 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCB4 pin
					Input capture at both edges
1	1	×	×		Capture input source is TGRC_3 compare match/input capture
					Input capture at generation of TGRC_3 compare match/input capture

Legend:

×: Don't care

RENESAS

Table 9.20   TIOR_5

Bit 7 IOB3	Bit 6 IOB2	Bit 5 IOB1	Bit 4 IOB0	Description	
				TGRB_5 Function	**TIOCB5 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	$\times$	0	0	Input capture register	Capture input source is TIOCB5 pin
					Input capture at rising edge
1	$\times$	0	1		Capture input source is TIOCB5 pin
					Input capture at falling edge
1	$\times$	1	$\times$		Capture input source is TIOCB5 pin
					Input capture at both edges

Legend:

$\times$: Don't care

Table 9.21   TIORH_0

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_0 Function**	**TIOCA0 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA0 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCA0 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCA0 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 1/count clock
					Input capture at TCNT_1 count-up/count-down

Legend:

×: Don't care

RENESAS

Table 9.22   TIORL_0

				Description	
Bit 3 IOC3	**Bit 2 IOC2**	**Bit 1 IOC1**	**Bit 0 IOC0**	**TGRC_0 Function**	**TIOCC0 Pin Function**
0	0	0	0	Output compare register*	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register*	Capture input source is TIOCC0 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCC0 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCC0 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 1/count clock
					Input capture at TCNT_1 count-up/count-down

Legend:

×: Don't care

Note: *   When the BFA bit in TMDR_0 is set to 1 and TGRC_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 9.23   TIOR_1

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_1 Function**	**TIOCA1 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA1 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCA1 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCA1 pin
					Input capture at both edges
1	1	×	×		Capture input source is TGRA_0 compare match/input capture
					Input capture at generation of channel 0/TGRA_0 compare match/input capture

Legend:
×: Don't care

RENESAS

Table 9.24   TIOR_2

Bit 3 IOA3	Bit 2 IOA2	Bit 1 IOA1	Bit 0 IOA0	**Description**	
				TGRA_2 Function	**TIOCA2 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	×	0	0	Input capture register	Capture input source is TIOCA2 pin
					Input capture at rising edge
1	×	0	1		Capture input source is TIOCA2 pin
					Input capture at falling edge
1	×	1	×		Capture input source is TIOCA2 pin
					Input capture at both edges

Legend:

×: Don't care

Table 9.25   TIORH_3

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_3 Function**	**TIOCA3 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA3 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCA3 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCA3 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 4/count clock
					Input capture at TCNT_4 count-up/count-down

Legend:
×: Don't care

RENESAS

Table 9.26   TIORL_3

				Description	
Bit 3 IOC3	Bit 2 IOC2	Bit 1 IOC1	Bit 0 IOC0	TGRC_3 Function	TIOCC3 Pin Function
0	0	0	0	Output compare register*	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register*	Capture input source is TIOCC3 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCC3 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCC3 pin
					Input capture at both edges
1	1	×	×		Capture input source is channel 4/count clock
					Input capture at TCNT_4 count-up/count-down

Legend:

×: Don't care

Note: *   When the BFA bit in TMDR_3 is set to 1 and TGRC_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

RENESAS

Table 9.27   TIOR_4

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_4 Function**	**TIOCA4 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA4 pin
					Input capture at rising edge
1	0	0	1		Capture input source is TIOCA4 pin
					Input capture at falling edge
1	0	1	×		Capture input source is TIOCA4 pin
					Input capture at both edges
1	1	×	×		Capture input source is TGRA_3 compare match/input capture
					Input capture at generation of TGRA_3 compare match/input capture

Legend:

×: Don't care

RENESAS

Table 9.28   TIOR_5

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_5 Function**	**TIOCA5 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output
					0 output at compare match
0	0	1	0		Initial output is 0 output
					1 output at compare match
0	0	1	1		Initial output is 0 output
					Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output
					0 output at compare match
0	1	1	0		Initial output is 1 output
					1 output at compare match
0	1	1	1		Initial output is 1 output
					Toggle output at compare match
1	×	0	0	Input capture register	Input capture source is TIOCA5 pin
					Input capture at rising edge
1	×	0	1		Input capture source is TIOCA5 pin
					Input capture at falling edge
1	×	1	×		Input capture source is TIOCA5 pin
					Input capture at both edges

Legend:

×: Don't care

9.3.4   Timer Interrupt Enable Register (TIER)

TIER controls enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TTGE	—	TCIEU	TCIEV	TGIED	TGIEC	TGIEB	TGIEA
Initial Value	0	1	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
7	TTGE	0	R/W	A/D Conversion Start Request Enable
				Enables/disables generation of A/D conversion start requests by TGRA input capture/compare match.
				0: A/D conversion start request generation disabled
				1: A/D conversion start request generation enabled
6	—	1	R	Reserved
				This is a read-only bit and cannot be modified.
5	TCIEU	0	R/W	Underflow Interrupt Enable
				Enables/disables interrupt requests (TCIU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1, 2, 4, and 5.
				In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.
				0: Interrupt requests (TCIU) by TCFU disabled
				1: Interrupt requests (TCIU) by TCFU enabled
4	TCIEV	0	R/W	Overflow Interrupt Enable
				Enables/disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1.
				0: Interrupt requests (TCIV) by TCFV disabled
				1: Interrupt requests (TCIV) by TCFV enabled

RENESAS

Bit	Bit Name	Initial value	R/W	Description
3	TGIED	0	R/W	TGR Interrupt Enable D
				Enables/disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3.
				In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.
				0: Interrupt requests (TGID) by TGFD bit disabled
				1: Interrupt requests (TGID) by TGFD bit enabled
2	TGIEC	0	R/W	TGR Interrupt Enable C
				Enables/disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3.
				In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.
				0: Interrupt requests (TGIC) by TGFC bit disabled
				1: Interrupt requests (TGIC) by TGFC bit enabled
1	TGIEB	0	R/W	TGR Interrupt Enable B
				Enables/disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1.
				0: Interrupt requests (TGIB) by TGFB bit disabled
				1: Interrupt requests (TGIB) by TGFB bit enabled
0	TGIEA	0	R/W	TGR Interrupt Enable A
				Enables/disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1.
				0: Interrupt requests (TGIA) by TGFA bit disabled
				1: Interrupt requests (TGIA) by TGFA bit enabled

RENESAS

9.3.5   Timer Status Register (TSR)

TSR indicates the status of each channel. The TPU has six TSR registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TCFD	—	TCFU	TCFV	TGFD	TGFC	TGFB	TGFA
Initial Value	1	1	0	0	0	0	0	0
R/W	R	R	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: * Only 0 can be written to bits 5 to 0, to clear flags.

Bit	Bit Name	Initial value	R/W	Description
7	TCFD	1	R	Count Direction Flag
				Status flag that shows the direction in which TCNT counts in channels 1, 2, 4, and 5.
				In channels 0 and 3, bit 7 is reserved. It is always read as 1 and cannot be modified.
				0: TCNT counts down
				1: TCNT counts up
6	—	1	R	Reserved
				This is a read-only bit and cannot be modified.
5	TCFU	0	R/(W)*	Underflow Flag
				Status flag that indicates that a TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting mode.
				In channels 0 and 3, bit 5 is reserved. It is always read as 0 and cannot be modified.
				[Setting condition]
				• When the TCNT value underflows (changes from H'0000 to H'FFFF)
				[Clearing condition]
				• When a 0 is written to TCFU after reading TCFU = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
4	TCFV	0	R/(W)*	Overflow Flag
				Status flag that indicates that a TCNT overflow has occurred.
				[Setting condition]
				• When the TCNT value overflows (changes from H'FFFF to H'0000)
				[Clearing condition]
				• When a 0 is written to TCFV after reading TCFV = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	TGFD	0	R/(W)*	Input Capture/Output Compare Flag D
				Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3.
				In channels 1, 2, 4, and 5, bit 3 is reserved. It is always read as 0 and cannot be modified.
				[Setting conditions]
				• When TCNT = TGRD while TGRD is functioning as output compare register
				• When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register
				[Clearing condition]
				• When 0 is written to TGFD after reading TGFD = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial value	R/W	Description
2	TGFC	0	R/(W)*	Input Capture/Output Compare Flag C
				Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3.
				In channels 1, 2, 4, and 5, bit 2 is reserved. It is always read as 0 and cannot be modified.
				[Setting conditions]
				• When TCNT = TGRC while TGRC is functioning as output compare register
				• When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register
				[Clearing condition]
				• When 0 is written to TGFC after reading TGFC = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	TGFB	0	R/(W)*	Input Capture/Output Compare Flag B
				Status flag that indicates the occurrence of TGRB input capture or compare match.
				[Setting conditions]
				• When TCNT = TGRB while TGRB is functioning as output compare register
				• When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register
				[Clearing condition]
				• When 0 is written to TGFB after reading TGFB = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial value	R/W	Description
0	TGFA	0	R/(W)*	Input Capture/Output Compare Flag A
				Status flag that indicates the occurrence of TGRA input capture or compare match.
				[Setting conditions]
				• When TCNT = TGRA while TGRA is functioning as output compare register
				• When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register
				[Clearing conditions]
				• When DMAC is activated by a TGIA interrupt while the DTA bit in DMDR of DMAC is 1
				• When 0 is written to TGFA after reading TGFA = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

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

9.3.6   Timer Counter (TCNT)

TCNT is a 16-bit readable/writable counter. The TPU has six TCNT counters, one for each channel.

TCNT is initialized to H'0000 by a reset or in hardware standby mode.

TCNT cannot be accessed in 8-bit units. TCNT must always be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

9.3.7   Timer General Register (TGR)

TGR is a 16-bit readable/writable register with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channels 0 and 3 and two each for channels 1, 2, 4, and 5. TGRC and TGRD for channels 0 and 3 can also be designated for operation as buffer registers. The TGR registers cannot be accessed in 8-bit units; they must always be accessed in 16-bit units. TGR and buffer register combinations during buffer operations are TGRA−TGRC and TGRB−TGRD.

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

9.3.8   Timer Start Register (TSTR)

TSTR starts or stops operation for channels 0 to 5. When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	CST5	CST4	CST3	CST2	CST1	CST0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
7, 6	⎯⎯	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
5	CST5	0	R/W	Counter Start 5 to 0
4	CST4	0	R/W	These bits select operation or stoppage for TCNT.
3	CST3	0	R/W	If 0 is written to the CST bit during operation with the
2	CST2	0	R/W	TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR
1	CST1	0	R/W	is written to when the CST bit is cleared to 0, the pin
0	CST0	0	R/W	output level will be changed to the set initial output value.
				0: TCNT_5 to TCNT_0 count operation is stopped
				1: TCNT_5 to TCNT_0 performs count operation

RENESAS

9.3.9   Timer Synchronous Register (TSYR)

TSYR selects independent operation or synchronous operation for the TCNT counters of channels 0 to 5. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial value	R/W	Description
7, 6	⸺	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
5	SYNC5	0	R/W	Timer Synchronization 5 to 0
4	SYNC4	0	R/W	These bits select whether operation is independent of or synchronized with other channels.
3	SYNC3	0	R/W	
2	SYNC2	0	R/W	When synchronous operation is selected, synchronous presetting of multiple channels, and synchronous clearing through counter clearing on another channel are possible.
1	SYNC1	0	R/W	
0	SYNC0	0	R/W	To set synchronous operation, the SYNC bits for at least two channels must be set to 1. To set synchronous clearing, in addition to the SYNC bit, the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR.
				0: TCNT_5 to TCNT_0 operate independently (TCNT presetting/clearing is unrelated to other channels)
				1: TCNT_5 to TCNT_0 perform synchronous operation (TCNT synchronous presetting/synchronous clearing is possible)

RENESAS

9.4   Operation

9.4.1   Basic Functions

Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, periodic counting, and external event counting.

Each TGR can be used as an input capture register or output compare register.

(1)   Counter Operation

When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

(a)   Example of count operation setting procedure

Figure 9.2 shows an example of the count operation setting procedure.
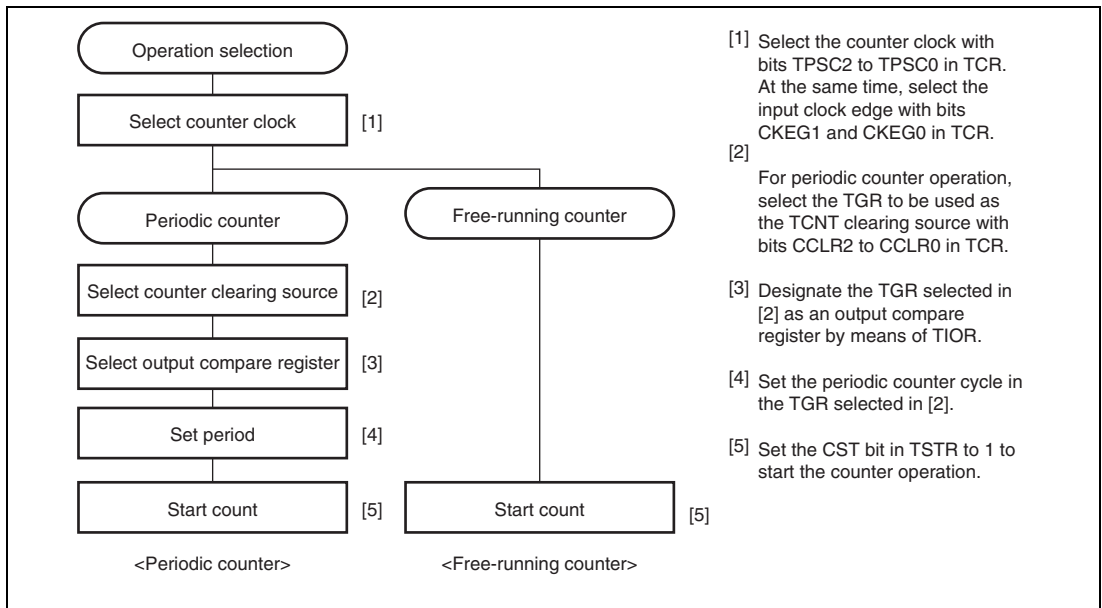
Figure 9.2   Example of Counter Operation Setting Procedure

(b)   Free-running count operation and periodic count operation

Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (changes from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000.

Figure 9.3 illustrates free-running counter operation.
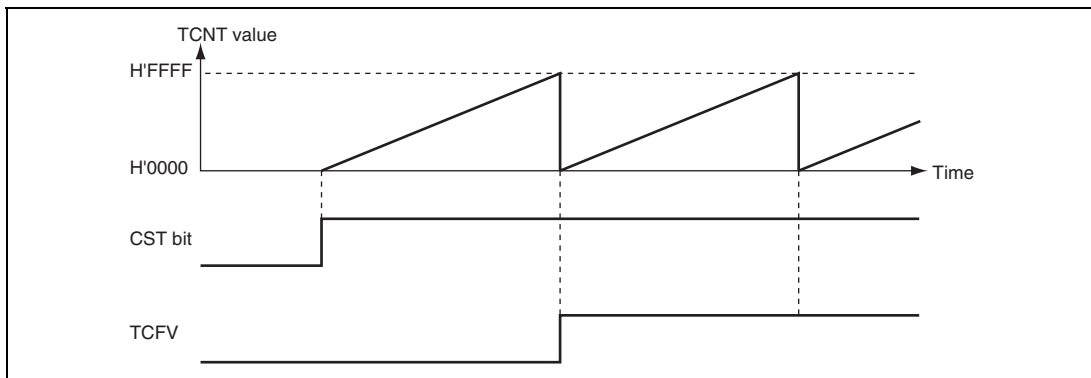
Figure 9.3   Free-Running Counter Operation

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts count-up operation as a periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000.

Figure 9.4 illustrates periodic counter operation.
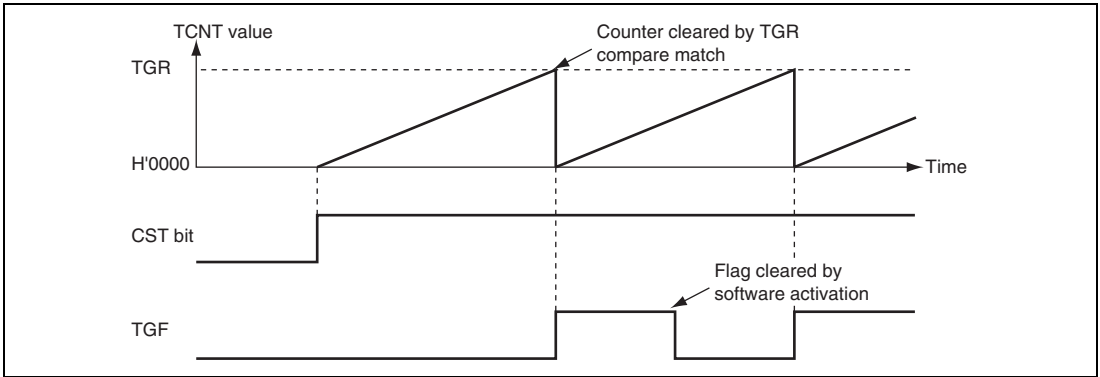
Figure 9.4   Periodic Counter Operation

(2)   Waveform Output by Compare Match

The TPU can perform 0, 1, or toggle output from the corresponding output pin using a compare match.

(a)   Example of setting procedure for waveform output by compare match

Figure 9.5 shows an example of the setting procedure for waveform output by a compare match.
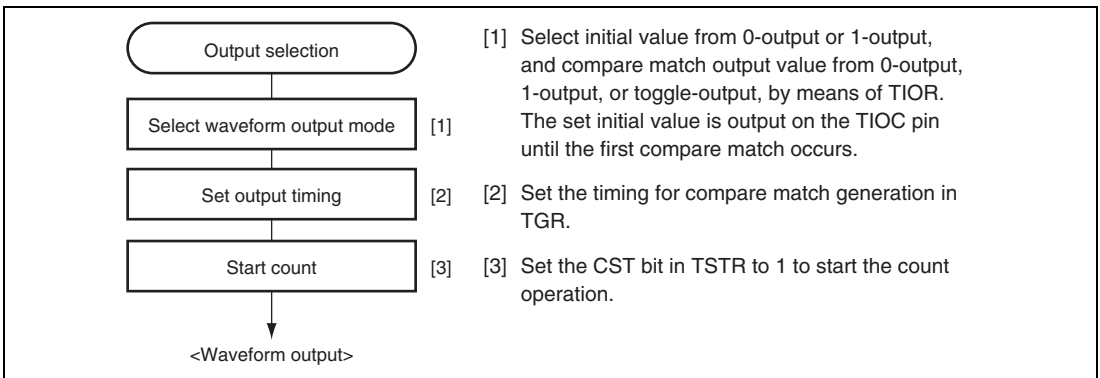
Figure 9.5   Example of Setting Procedure for Waveform Output by Compare Match

(b)   Examples of waveform output operation

Figure 9.6 shows an example of 0-output and 1-output.

In this example, TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level match, the pin level does not change.
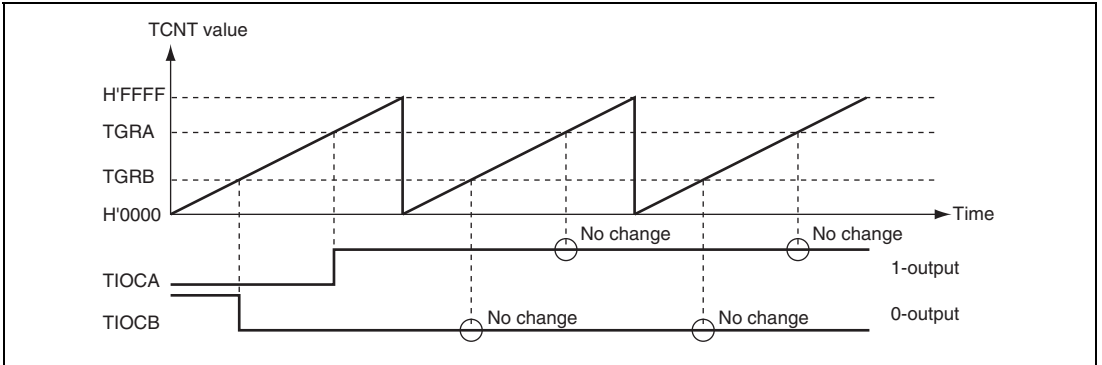
Figure 9.6   Example of 0-Output/1-Output Operation

Figure 9.7 shows an example of toggle output.

In this example, TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.
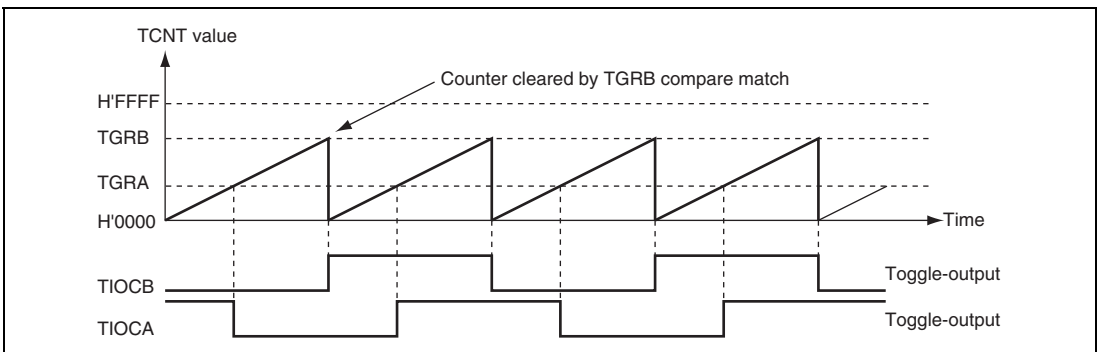
Figure 9.7   Example of Toggle Output Operation

(3)   Input Capture Function

The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detection edge. For channels 0, 1, 3, and 4, it is also possible to specify another channel's counter input clock or compare match signal as the input capture source.

Note: When another channel's counter input clock is used as the input capture input for channels 0 and 3, Pϕ/1 should not be selected as the counter input clock used for input capture input. Input capture will not be generated if Pϕ/1 is selected.

(a)   Example of setting procedure for input capture operation

Figure 9.8 shows an example of the setting procedure for input capture operation.

Input selection

Select input capture input   [1]

Start count   [2]

<Input capture operation>

[1]   Designate TGR as an input capture register by means of TIOR, and select the input capture source and input signal edge (rising edge, falling edge, or both edges).

[2]   Set the CST bit in TSTR to 1 to start the count operation.
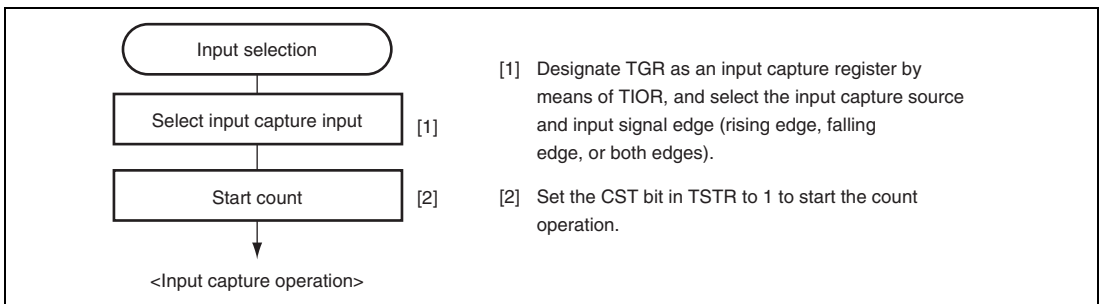
Figure 9.8   Example of Setting Procedure for Input Capture Operation

(b)   Example of input capture operation

Figure 9.9 shows an example of input capture operation.

In this example, both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.
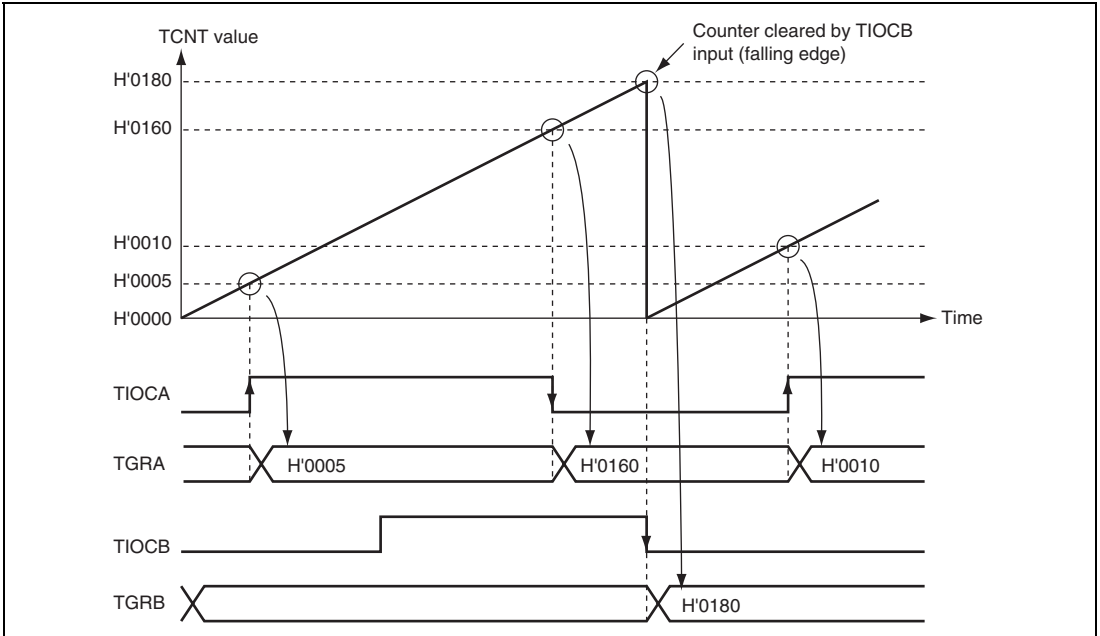
Figure 9.9   Example of Input Capture Operation

9.4.2   Synchronous Operation

In synchronous operation, the values in multiple TCNT counters can be rewritten simultaneously (synchronous presetting). Also, multiple TCNT counters can be cleared simultaneously (synchronous clearing) by making the appropriate setting in TCR.

Synchronous operation enables TGR to be incremented with respect to a single time base.

Channels 0 to 5 can all be designated for synchronous operation.

(1)   Example of Synchronous Operation Setting Procedure

Figure 9.10 shows an example of the synchronous operation setting procedure.

[1] Set the SYNC bits in TSYR corresponding to the channels to be designated for synchronous operation to 1.

[2] When the TCNT counter of any of the channels designated for synchronous operation is written to, the same value is simultaneously written to the other TCNT counters.

[3] Use bits CCLR2 to CCLR0 in TCR to specify TCNT clearing by input capture/output compare, etc.

[4] Use bits CCLR2 to CCLR0 in TCR to designate synchronous clearing for the counter clearing source.

[5] Set the CST bits in TSTR for the relevant channels to 1, to start the count operation.

Figure 9.10   Example of Synchronous Operation Setting Procedure

(2)   Example of Synchronous Operation

Figure 9.11 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGRB_0 compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source.

Three-phase PWM waveforms are output from pins TIOCA0, TIOCA1, and TIOCA2. At this time, synchronous presetting and synchronous clearing by TGRB_0 compare match are performed for channel 0 to 2 TCNT counters, and the data set in TGRB_0 is used as the PWM cycle.

For details on PWM modes, see section 9.4.5, PWM Modes.
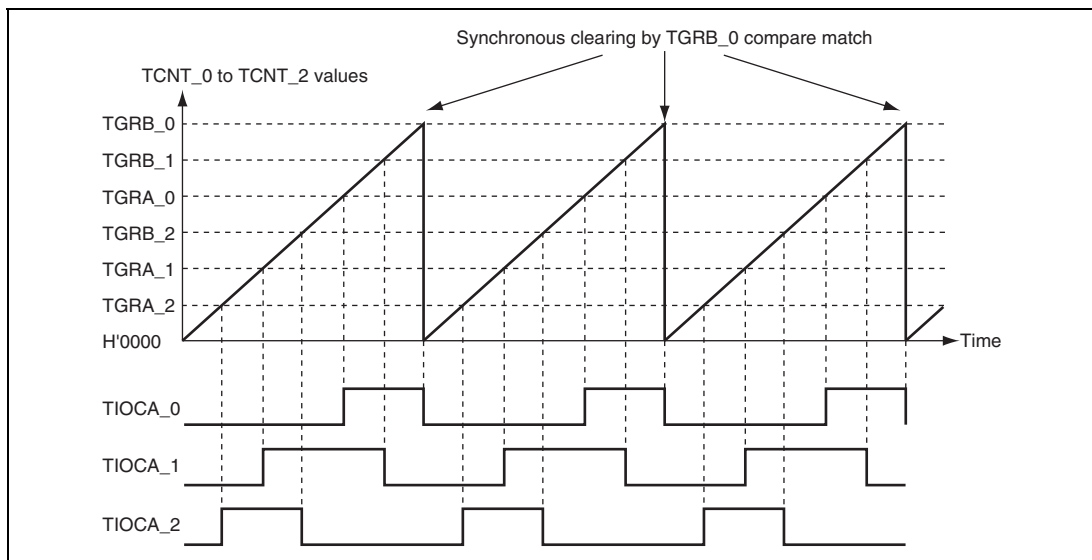
Figure 9.11   Example of Synchronous Operation

9.4.3   Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

Buffer operation differs depending on whether TGR has been designated as an input capture register or a compare match register.

Table 9.29 shows the register combinations used in buffer operation.

Table 9.29   Register Combinations in Buffer Operation

Channel	Timer General Register	Buffer Register
0	TGRA_0	TGRC_0
	TGRB_0	TGRD_0
3	TGRA_3	TGRC_3
	TGRB_3	TGRD_3

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.

  This operation is illustrated in figure 9.12.
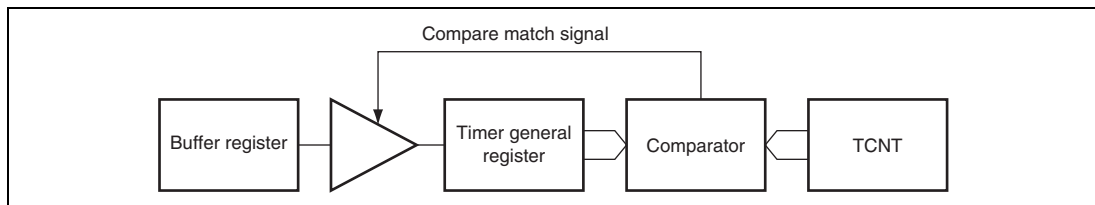
Figure 9.12   Compare Match Buffer Operation

• When TGR is an input capture register

When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in TGR is transferred to the buffer register.

This operation is illustrated in figure 9.13.
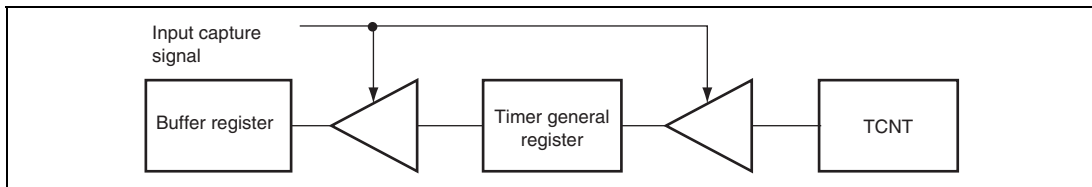
Figure 9.13   Input Capture Buffer Operation

(1)   Example of Buffer Operation Setting Procedure

Figure 9.14 shows an example of the buffer operation setting procedure.

Buffer operation

Select TGR function   [1]   [1]   Designate TGR as an input capture register or output compare register by means of TIOR.

Set buffer operation   [2]   [2]   Designate TGR for buffer operation with bits BFA and BFB in TMDR.

Start count   [3]   [3]   Set the CST bit in TSTR to 1 to start the count operation.

<Buffer operation>
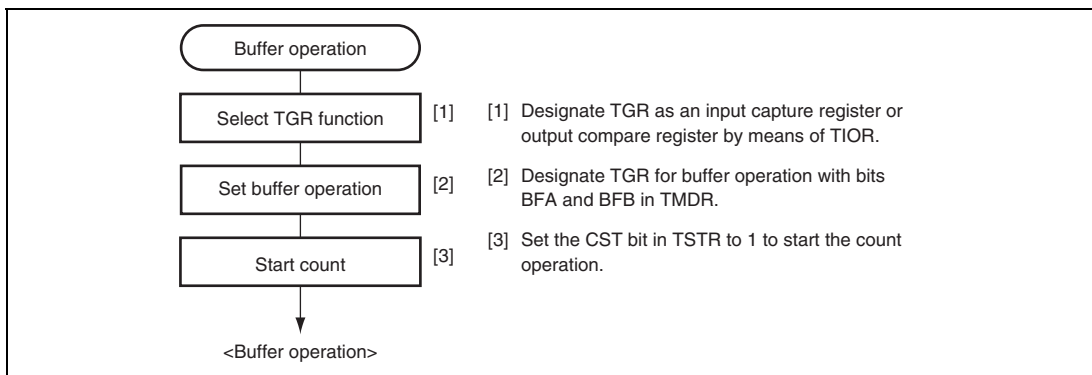
Figure 9.14   Example of Buffer Operation Setting Procedure

(2)   Examples of Buffer Operation

(a)   When TGR is an output compare register

Figure 9.15 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B.

As buffer operation has been set, when compare match A occurs, the output changes and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

For details on PWM modes, see section 9.4.5, PWM Modes.
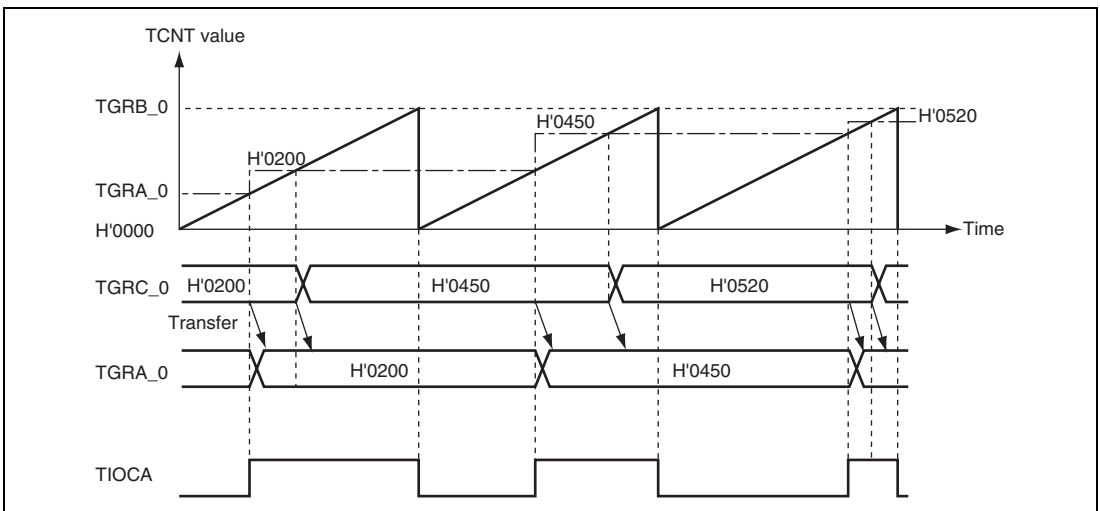
Figure 9.15   Example of Buffer Operation (1)

(b)   When TGR is an input capture register

Figure 9.16 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC.

Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge.

As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.
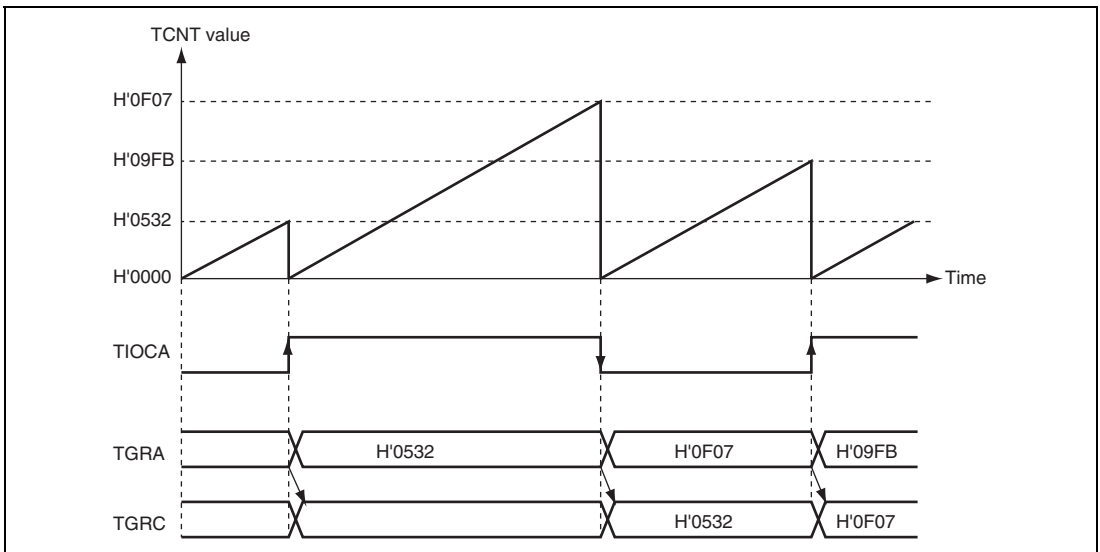
Figure 9.16   Example of Buffer Operation (2)

9.4.4   Cascaded Operation

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter.

This function works by counting the channel 1 (channel 4) counter clock at overflow/underflow of TCNT_2 (TCNT_5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

Table 9.30 shows the register combinations used in cascaded operation.

Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

Table 9.30   Cascaded Combinations

Combination	Upper 16 Bits	Lower 16 Bits
Channels 1 and 2	TCNT_1	TCNT_2
Channels 4 and 5	TCNT_4	TCNT_5

(1)   Example of Cascaded Operation Setting Procedure

Figure 9.17 shows an example of the setting procedure for cascaded operation.
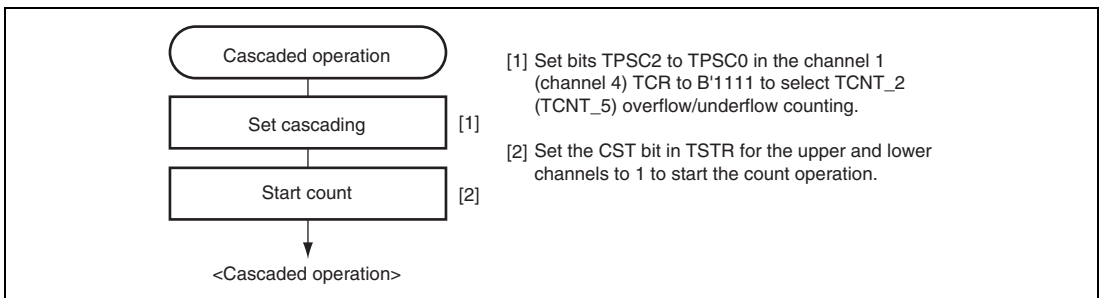
Figure 9.17   Example of Cascaded Operation Setting Procedure

(2)   Examples of Cascaded Operation

Figure 9.18 illustrates the operation when counting upon TCNT_2 overflow/underflow has been set for TCNT_1, TGRA_1 and TGRA_2 have been designated as input capture registers, and the TIOC pin rising edge has been selected.

When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGRA_1, and the lower 16 bits to TGRA_2.
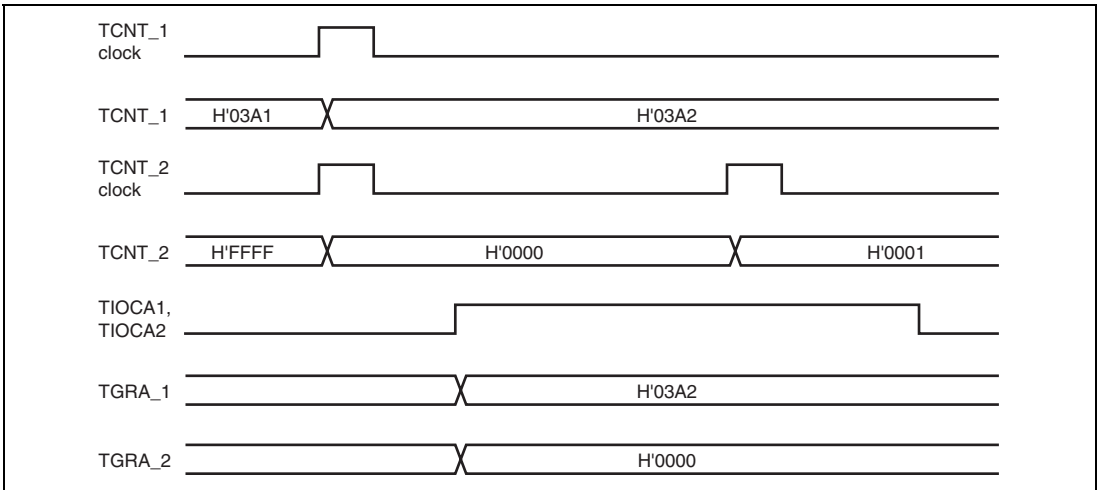
Figure 9.18   Example of Cascaded Operation (1)

Figure 9.19 illustrates the operation when counting upon TCNT_2 overflow/underflow has been set for TCNT_1, and phase counting mode has been designated for channel 2.

TCNT_1 is incremented by TCNT_2 overflow and decremented by TCNT_2 underflow.
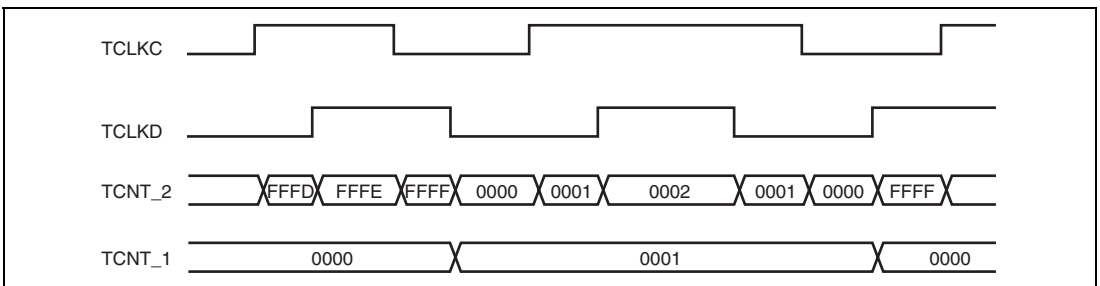
Figure 9.19   Example of Cascaded Operation (2)

9.4.5   PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0-, 1-, or toggle-output can be selected as the output level in response to compare match of each TGR.

Settings of TGR registers can output a PWM waveform in the range of 0% to 100% duty cycle.

Designating TGR compare match as the counter clearing source enables the cycle to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible.

There are two PWM modes, as described below.

(a) PWM mode 1

PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The outputs specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR are output from the TIOCA and TIOCC pins at compare matches A and C, respectively. The outputs specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR are output at compare matches B and D, respectively. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs.

In PWM mode 1, a maximum 8-phase PWM output is possible.

(b) PWM mode 2

PWM output is generated using one TGR as the cycle register and the others as duty cycle registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a synchronous register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty cycle registers are identical, the output value does not change when a compare match occurs.

In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

The correspondence between PWM output pins and registers is shown in table 9.31.

Table 9.31   PWM Output Registers and Output Pins

Channel	Registers	Output Pins	
		PWM Mode 1	**PWM Mode 2**
0	TGRA_0	TIOCA0	TIOCA0
	TGRB_0		TIOCB0
	TGRC_0	TIOCC0	TIOCC0
	TGRD_0		TIOCD0
1	TGRA_1	TIOCA1	TIOCA1
	TGRB_1		TIOCB1
2	TGRA_2	TIOCA2	TIOCA2
	TGRB_2		TIOCB2
3	TGRA_3	TIOCA3	TIOCA3
	TGRB_3		TIOCB3
	TGRC_3	TIOCC3	TIOCC3
	TGRD_3		TIOCD3
4	TGRA_4	TIOCA4	TIOCA4
	TGRB_4		TIOCB4
5	TGRA_5	TIOCA5	TIOCA5
	TGRB_5		TIOCB5

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the cycle is set.

RENESAS

(1)   Example of PWM Mode Setting Procedure

Figure 9.20 shows an example of the PWM mode setting procedure.
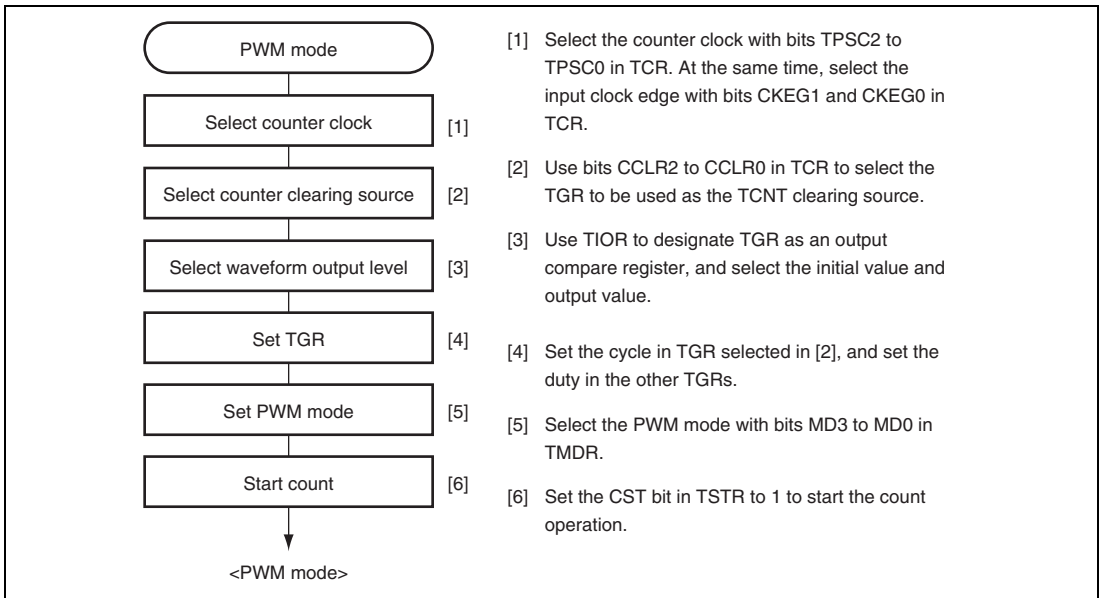
Figure 9.20   Example of PWM Mode Setting Procedure

(2)   Examples of PWM Mode Operation

Figure 9.21 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value.

In this case, the value set in TGRA is used as the cycle, and the value set in TGRB register as the duty cycle.

RENESAS
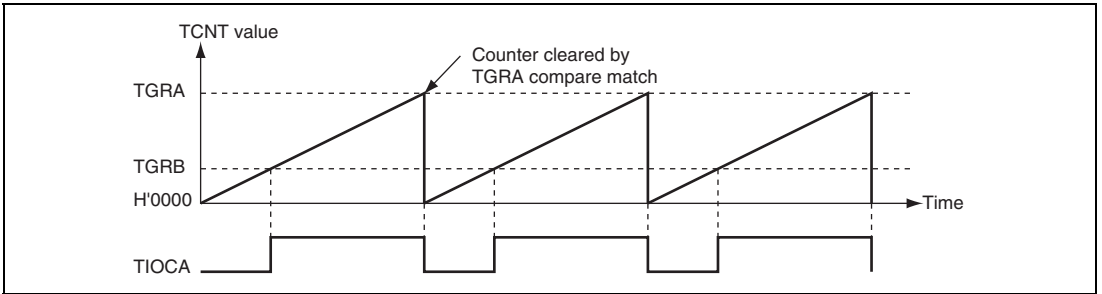
Figure 9.21   Example of PWM Mode Operation (1)

Figure 9.22 shows an example of PWM mode 2 operation.

In this example, synchronous operation is designated for channels 0 and 1, TGRB_1 compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGRA_0 to TGRD_0, TGRA_1), to output a 5-phase PWM waveform.

In this case, the value set in TGRB_1 is used as the cycle, and the values set in the other TGRs as the duty cycle.
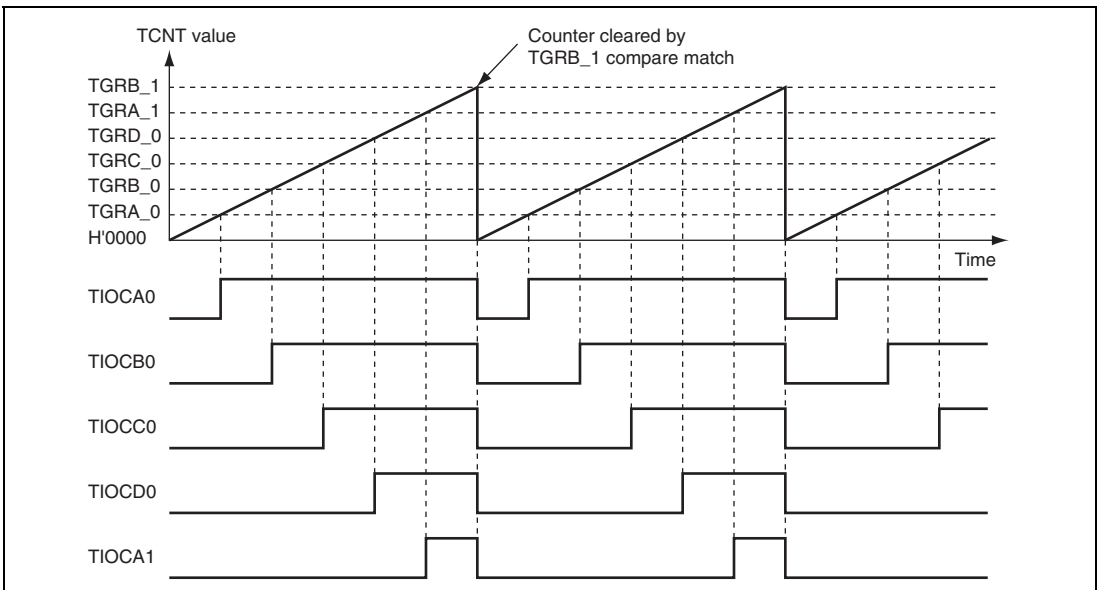
Figure 9.22   Example of PWM Mode Operation (2)

Figure 9.23 shows examples of PWM waveform output with 0% duty cycle and 100% duty cycle in PWM mode.
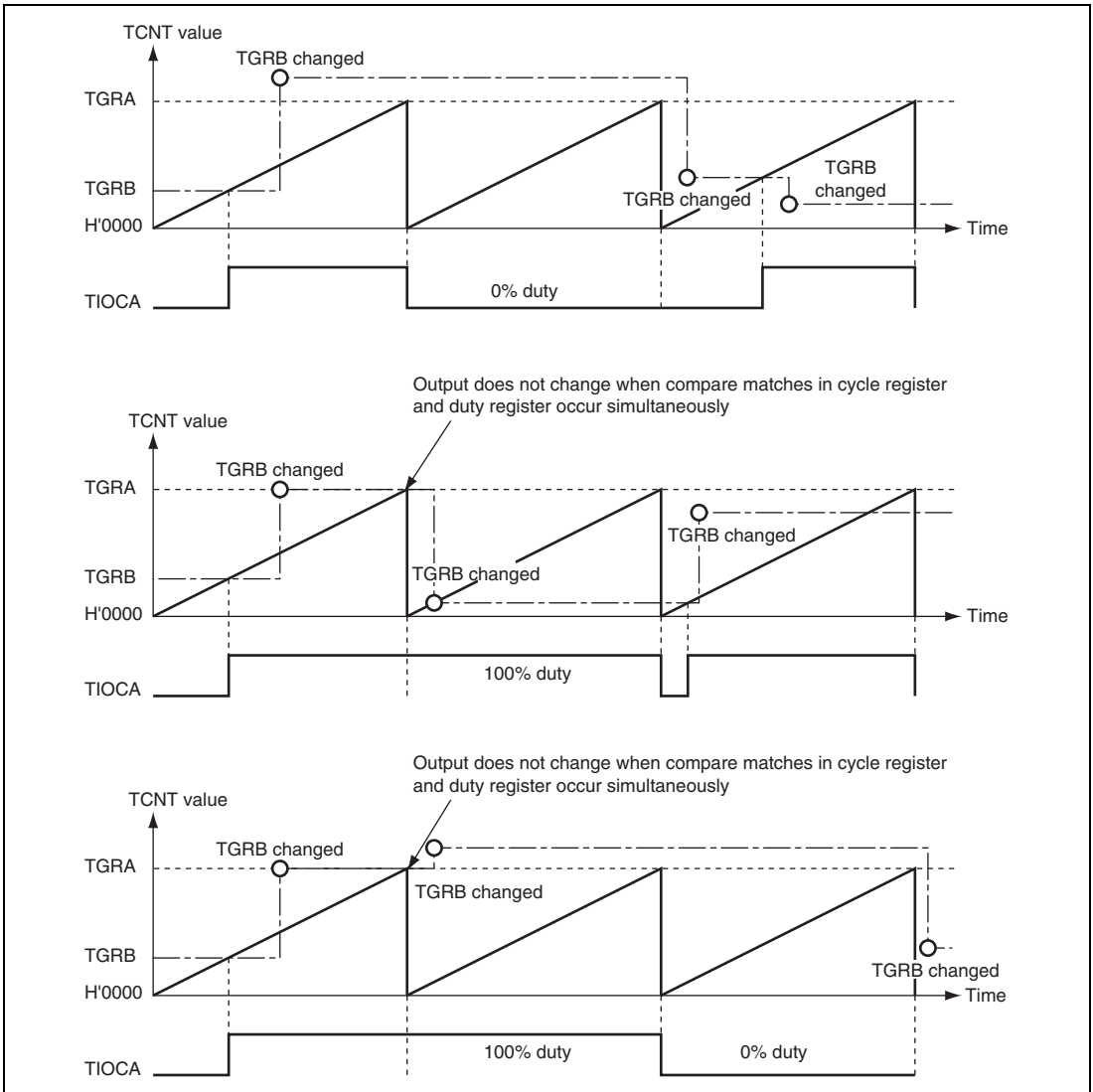
Figure 9.23   Example of PWM Mode Operation (3)

9.4.6   Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1, 2, 4, and 5.

When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used.

This can be used for two-phase encoder pulse input.

When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set.

The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 9.32 shows the correspondence between external clock pins and channels.

Table 9.32   Clock Input Pins in Phase Counting Mode

Channels	External Clock Pins	
	A-Phase	**B-Phase**
When channel 1 or 5 is set to phase counting mode	TCLKA	TCLKB
When channel 2 or 4 is set to phase counting mode	TCLKC	TCLKD

(1)   Example of Phase Counting Mode Setting Procedure

Figure 9.24 shows an example of the phase counting mode setting procedure.

RENESAS

[1] Select phase counting mode with bits MD3 to MD0 in TMDR.
[2] Set the CST bit in TSTR to 1 to start the count operation.
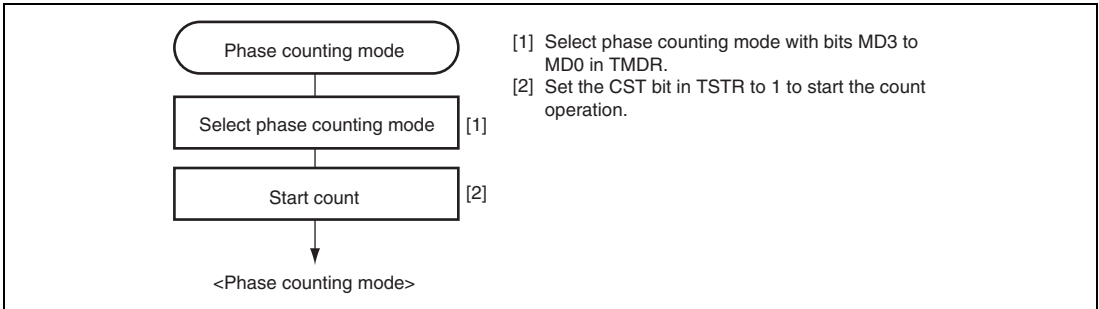
Figure 9.24   Example of Phase Counting Mode Setting Procedure

(2)   Examples of Phase Counting Mode Operation

In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

(a)   Phase counting mode 1

Figure 9.25 shows an example of phase counting mode 1 operation, and table 9.33 summarizes the TCNT up/down-count conditions.
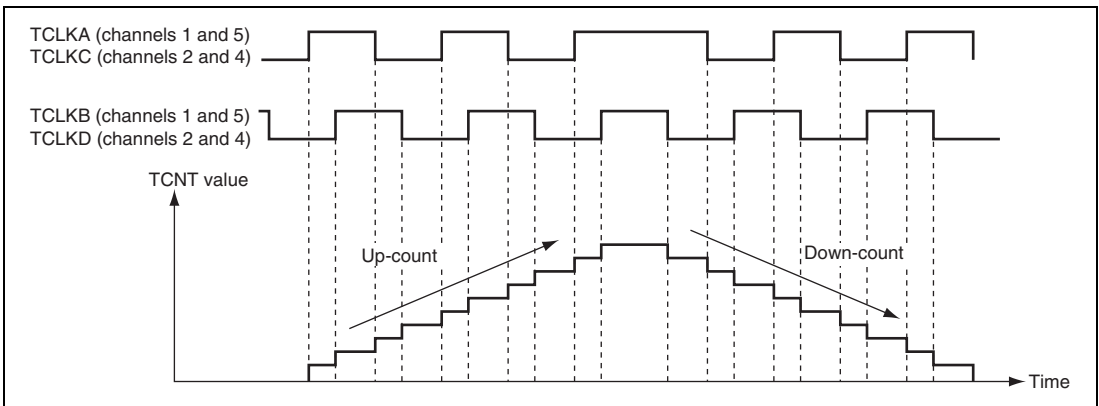
Figure 9.25   Example of Phase Counting Mode 1 Operation

Table 9.33   Up-/Down-Count Conditions in Phase Counting Mode 1

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	⤒ (rising edge)	Up-count
Low level	⤓ (falling edge)	
⤒ (rising edge)	Low level	
⤓ (falling edge)	High level	
High level	⤓ (falling edge)	Down-count
Low level	⤒ (rising edge)	
⤒ (rising edge)	High level	
⤓ (falling edge)	Low level	

Legend:

⤒ : Rising edge

⤓: Falling edge

RENESAS

(b)   Phase counting mode 2

Figure 9.26 shows an example of phase counting mode 2 operation, and table 9.34 summarizes the TCNT up-/down-count conditions.
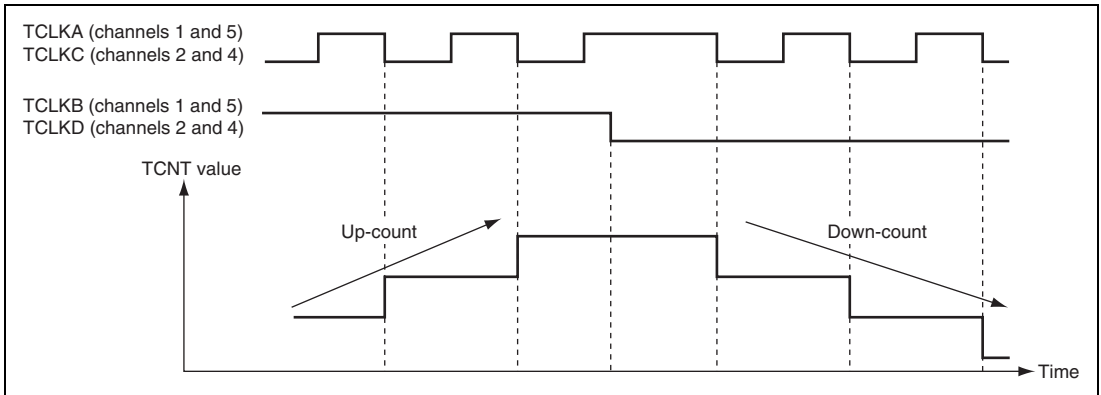
Figure 9.26   Example of Phase Counting Mode 2 Operation

Table 9.34   Up-/Down-Count Conditions in Phase Counting Mode 2

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	⌐ Rising edge	Don't care
Low level	⌐ Falling edge	
⌐ Rising edge	Low level	
⌐ Falling edge	High level	Up-count
High level	⌐ Falling edge	Don't care
Low level	⌐ Rising edge	
⌐ Rising edge	High level	
⌐ Falling edge	Low level	Down-count

Legend:
⌐ : Rising edge
⌐: Falling edge

(c)   Phase counting mode 3

Figure 9.27 shows an example of phase counting mode 3 operation, and table 9.35 summarizes the TCNT up-/down-count conditions.
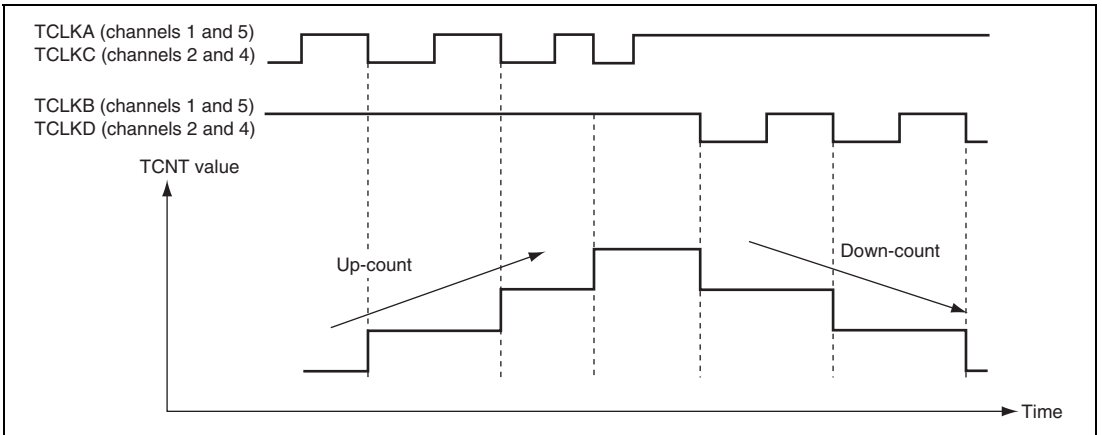
Figure 9.27   Example of Phase Counting Mode 3 Operation

Table 9.35   Up-/Down-Count Conditions in Phase Counting Mode 3

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	⌐ Rising edge	Don't care
Low level	¬ Falling edge	
⌐ Rising edge	Low level	
¬ Falling edge	High level	Up-count
High level	¬ Falling edge	Down-count
Low level	⌐ Rising edge	Don't care
⌐ Rising edge	High level	
¬ Falling edge	Low level	

Legend:
⌐ : Rising edge
¬ : Falling edge

(d)   Phase counting mode 4

Figure 9.28 shows an example of phase counting mode 4 operation, and table 9.36 summarizes the TCNT up-/down-count conditions.
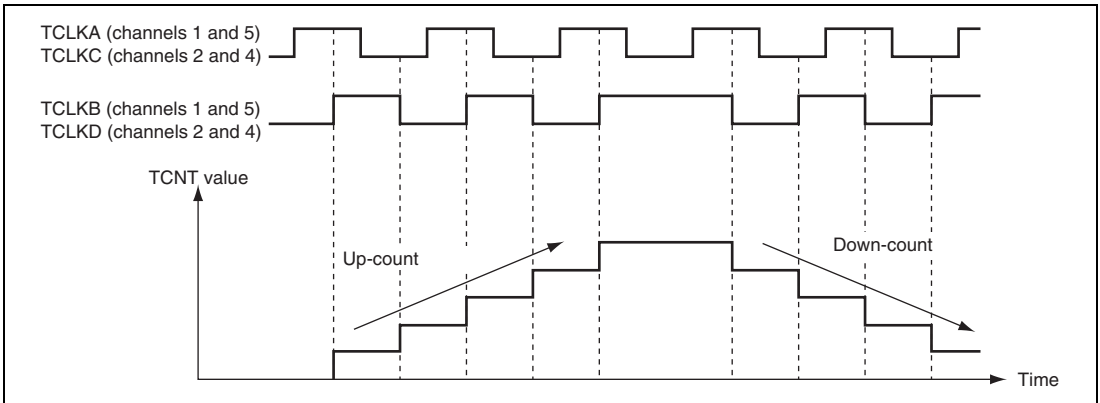
Figure 9.28   Example of Phase Counting Mode 4 Operation

Table 9.36   Up-/Down-Count Conditions in Phase Counting Mode 4

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	⤒	Up-count
Low level	⤓	
⤒	Low level	Don't care
⤓	High level	
High level	⤓	Down-count
Low level	⤒	
⤒	High level	Don't care
⤓	Low level	

Legend:
⤒ : Rising edge
⤓ : Falling edge

(3)   Phase Counting Mode Application Example

Figure 9.29 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

Channel 1 is set to phase counting mode 1, and the encoder pulse A-phase and B-phase are input to TCLKA and TCLKB.

Channel 0 operates with TCNT counter clearing by TGRC_0 compare match; TGRA_0 and TGRC_0 are used for the compare match function and are set with the speed control cycle and position control cycle. TGRB_0 is used for input capture, with TGRB_0 and TGRD_0 operating in buffer mode. The channel 1 counter input clock is designated as the TGRB_0 input capture source, and the pulse width of 2-phase encoder 4-multiplication pulses is detected.

TGRA_1 and TGRB_1 for channel 1 are designated for input capture, channel 0 TGRA_0 and TGRC_0 compare matches are selected as the input capture source, and the up-/down-counter values for the control cycles are stored.

This procedure enables accurate position/speed detection to be achieved.
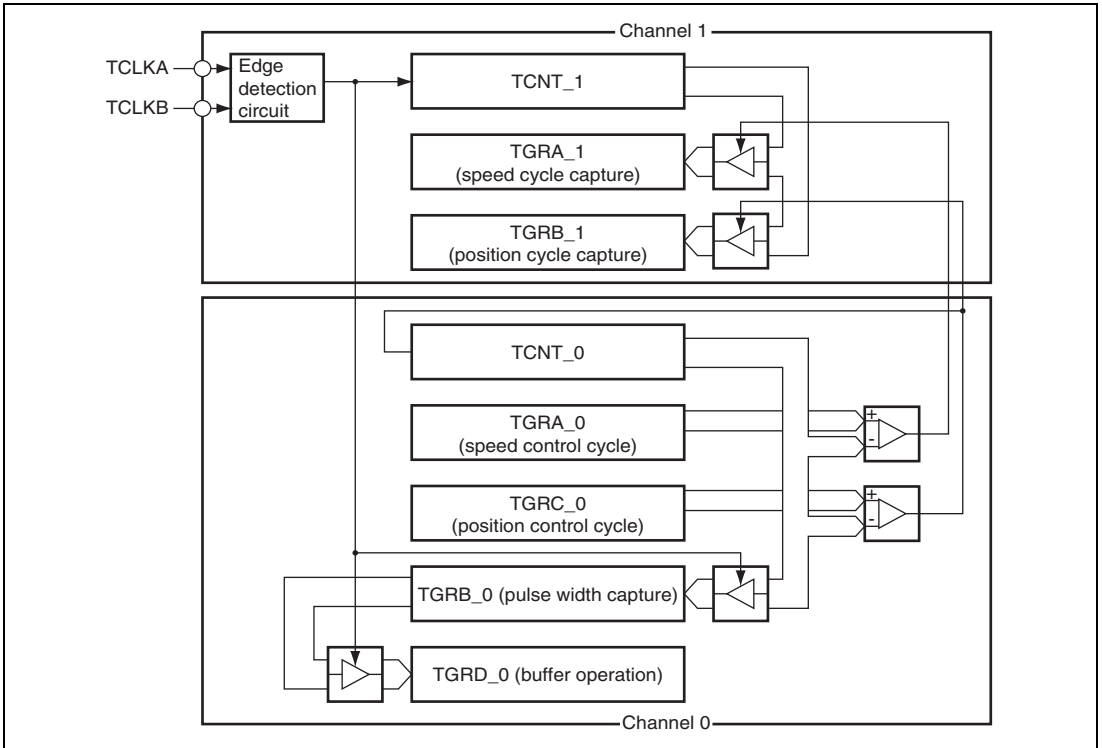
Figure 9.29   Phase Counting Mode Application Example

9.5   Interrupt Sources

There are three kinds of TPU interrupt sources. TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disable bit, allowing generation of interrupt request signals to be enabled or disabled individually.

When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0.

Relative channel priority levels can be changed by the interrupt controller, but the priority within a channel is fixed. For details, see section 5, Interrupt Controller.

Table 9.37 lists the TPU interrupt sources.

Table 9.37   TPU Interrupts

Channel	Name	Interrupt Source	Interrupt Flag	DMAC Activation
0	TGI0A	TGRA_0 input capture/compare match	TGFA_0	O
	TGI0B	TGRB_0 input capture/compare match	TGFB_0	—
	TGI0C	TGRC_0 input capture/compare match	TGFC_0	—
	TGI0D	TGRD_0 input capture/compare match	TGFD_0	—
	TCI0V	TCNT_0 overflow	TCFV_0	—
1	TGI1A	TGRA_1 input capture/compare match	TGFA_1	O
	TGI1B	TGRB_1 input capture/compare match	TGFB_1	—
	TCI1V	TCNT_1 overflow	TCFV_1	—
	TCI1U	TCNT_1 underflow	TCFU_1	—
2	TGI2A	TGRA_2 input capture/compare match	TGFA_2	O
	TGI2B	TGRB_2 input capture/compare match	TGFB_2	—
	TCI2V	TCNT_2 overflow	TCFV_2	—
	TCI2U	TCNT_2 underflow	TCFU_2	—

RENESAS

Channel	Name	Interrupt Source	Interrupt Flag	DMAC Activation
3	TGI3A	TGRA_3 input capture/compare match	TGFA_3	O
	TGI3B	TGRB_3 input capture/compare match	TGFB_3	—
	TGI3C	TGRC_3 input capture/compare match	TGFC_3	—
	TGI3D	TGRD_3 input capture/compare match	TGFD_3	—
	TCI3V	TCNT_3 overflow	TCFV_3	—
4	TGI4A	TGRA_4 input capture/compare match	TGFA_4	O
	TGI4B	TGRB_4 input capture/compare match	TGFB_4	—
	TCI4V	TCNT_4 overflow	TCFV_4	—
	TCI4U	TCNT_4 underflow	TCFU_4	—
5	TGI5A	TGRA_5 input capture/compare match	TGFA_5	O
	TGI5B	TGRB_5 input capture/compare match	TGFB_5	—
	TCI5V	TCNT_5 overflow	TCFV_5	—
	TCI5U	TCNT_5 underflow	TCFU_5	—

Legend:

O : Possible

— : Not possible

Note: This table shows the initial state immediately after a reset. The relative channel priority
levels can be changed by the interrupt controller.

(1)   Input Capture/Compare Match Interrupt

An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1
by the occurrence of a TGR input capture/compare match on a channel. The interrupt request is
cleared by clearing the TGF flag to 0. The TPU has 16 input capture/compare match interrupts,
four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

(2)   Overflow Interrupt

An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to
1 by the occurrence of a TCNT overflow on a channel. The interrupt request is cleared by clearing
the TCFV flag to 0. The TPU has six overflow interrupts, one for each channel.

(3)   Underflow Interrupt

An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of a TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has four underflow interrupts, one each for channels 1, 2, 4, and 5.

9.6   DMAC Activation

The DMAC can be activated by the TGRA input capture/compare match interrupt for a channel. For details, see section 7, DMA Controller (DMAC).

A total of six TPU input capture/compare match interrupts can be used as DMAC activation sources, one for each channel.

9.7   A/D Converter Activation

The TGRA input capture/compare match for each channel can activate the A/D converter.

If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

RENESAS

9.8   Operation Timing

9.8.1   Input/Output Timing

(1)   TCNT Count Timing

Figure 9.30 shows TCNT count timing in internal clock operation, and figure 9.31 shows TCNT count timing in external clock operation.
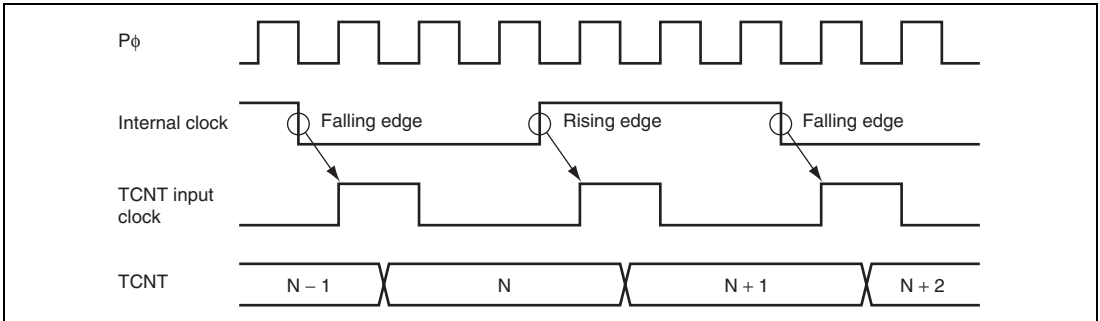
Figure 9.30   Count Timing in Internal Clock Operation
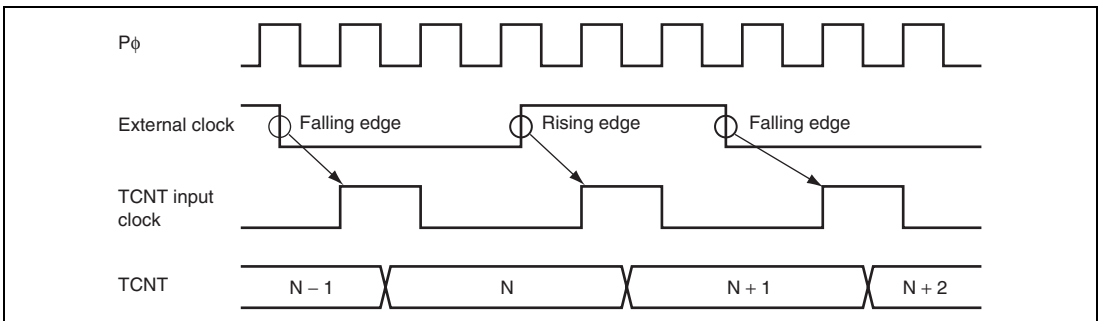
Figure 9.31   Count Timing in External Clock Operation

(2)   Output Compare Output Timing

A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin (TIOC pin). After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

Figure 9.32 shows output compare output timing.
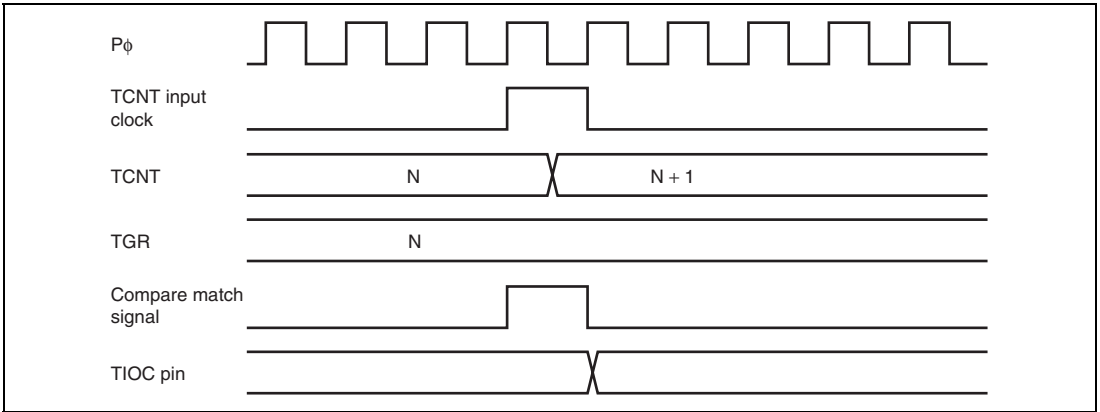
Figure 9.32   Output Compare Output Timing

(3)   Input Capture Signal Timing

Figure 9.33 shows input capture signal timing.
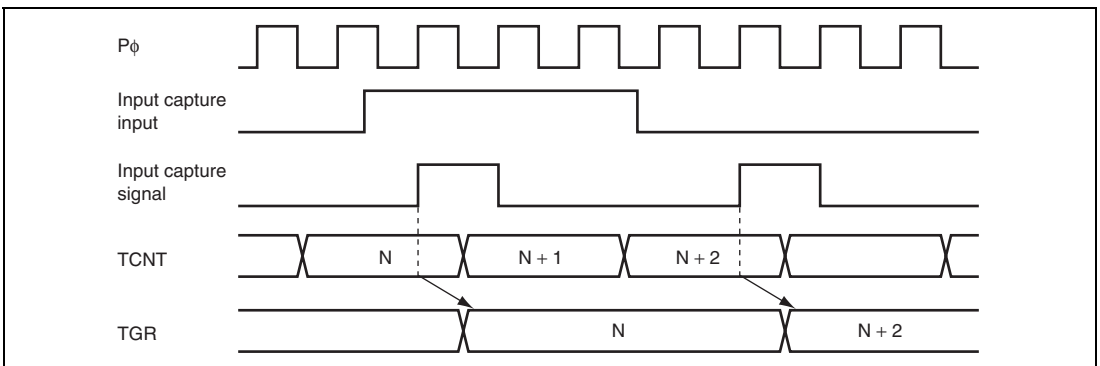
Figure 9.33   Input Capture Input Signal Timing

(4)   Timing for Counter Clearing by Compare Match/Input Capture

Figure 9.34 shows the timing when counter clearing by compare match occurrence is specified, and figure 9.35 shows the timing when counter clearing by input capture occurrence is specified.
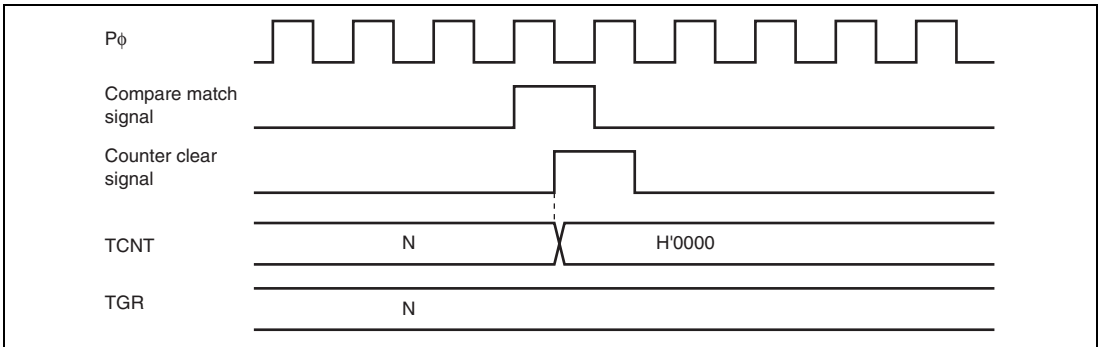
Figure 9.34   Counter Clear Timing (Compare Match)
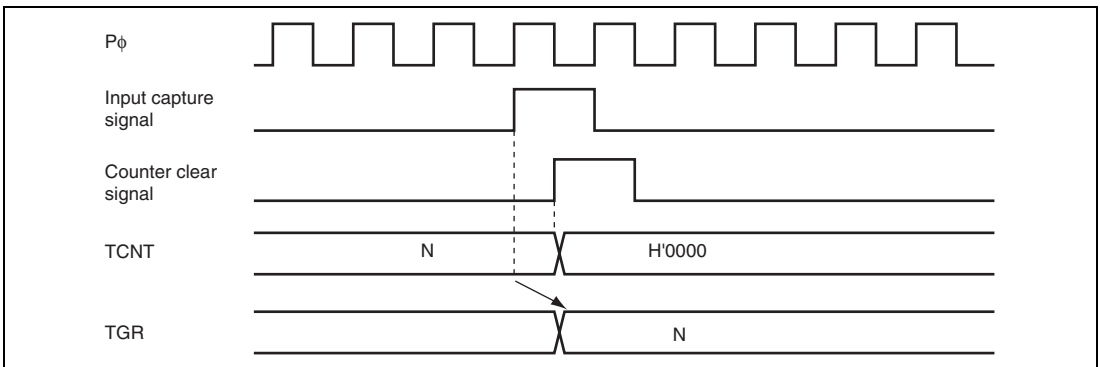
Figure 9.35   Counter Clear Timing (Input Capture)

(5)   Buffer Operation Timing

Figures 9.36 and 9.37 show the timings in buffer operation.
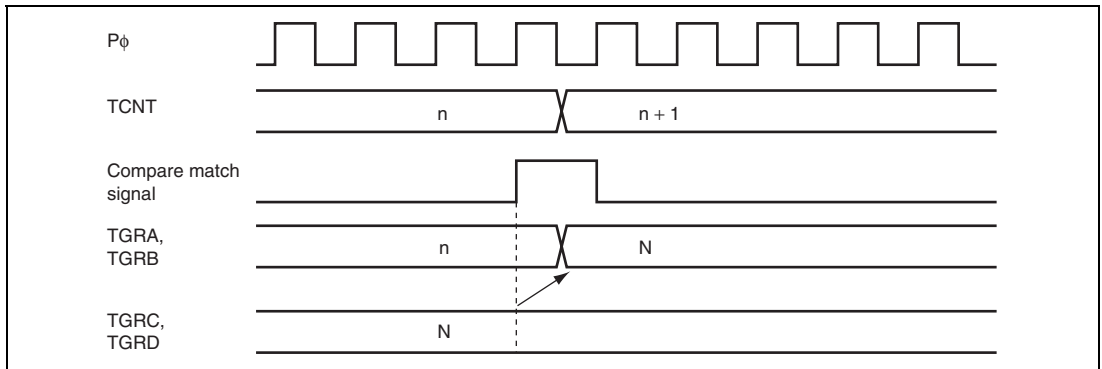
Figure 9.36   Buffer Operation Timing (Compare Match)
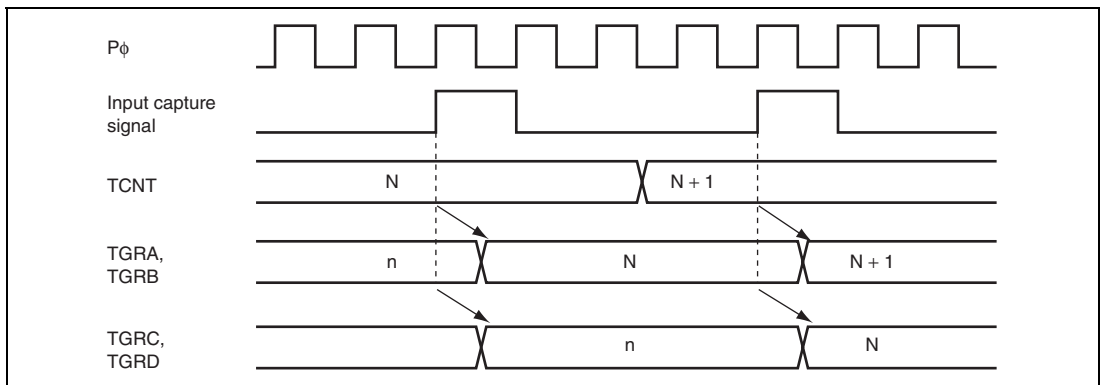
Figure 9.37   Buffer Operation Timing (Input Capture)

9.8.2   Interrupt Signal Timing

(1)   TGF Flag Setting Timing in Case of Compare Match

Figure 9.38 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and the TGI interrupt request signal timing.
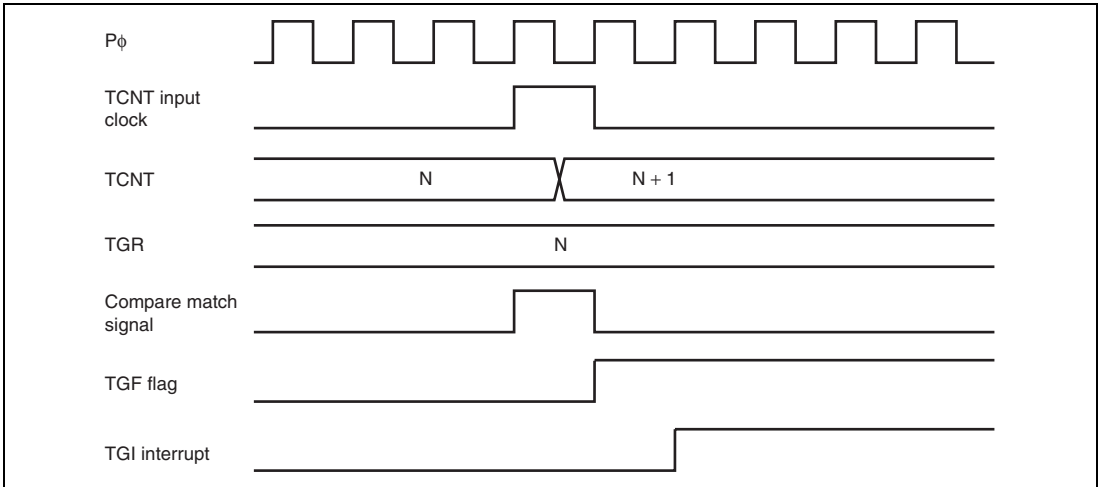
Figure 9.38   TGI Interrupt Timing (Compare Match)

(2)   TGF Flag Setting Timing in Case of Input Capture

Figure 9.39 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and the TGI interrupt request signal timing.
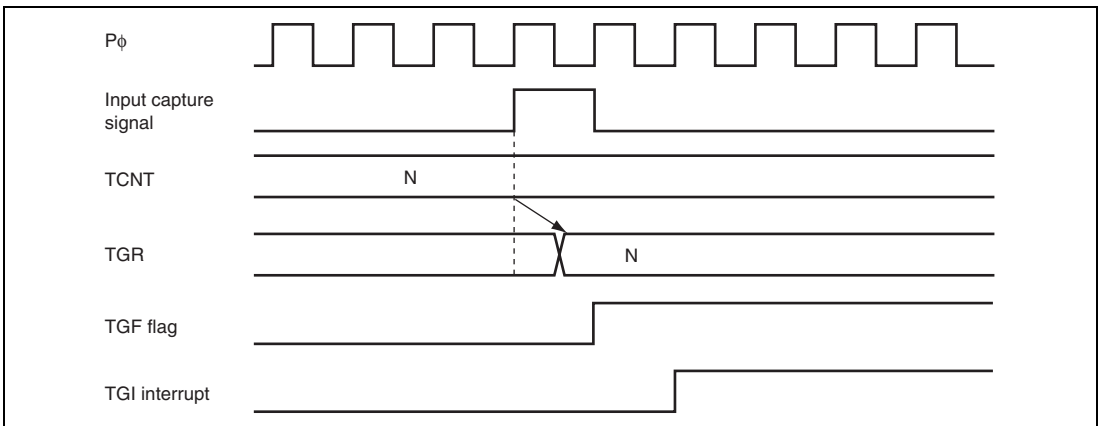
Figure 9.39   TGI Interrupt Timing (Input Capture)

(3)   TCFV Flag/TCFU Flag Setting Timing

Figure 9.40 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and the TCIV interrupt request signal timing.

Figure 9.41 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and the TCIU interrupt request signal timing.
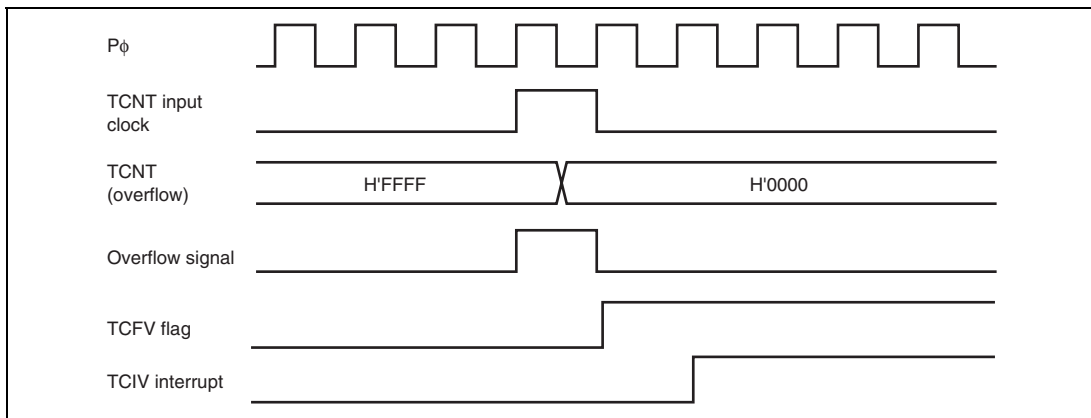
Figure 9.40   TCIV Interrupt Setting Timing
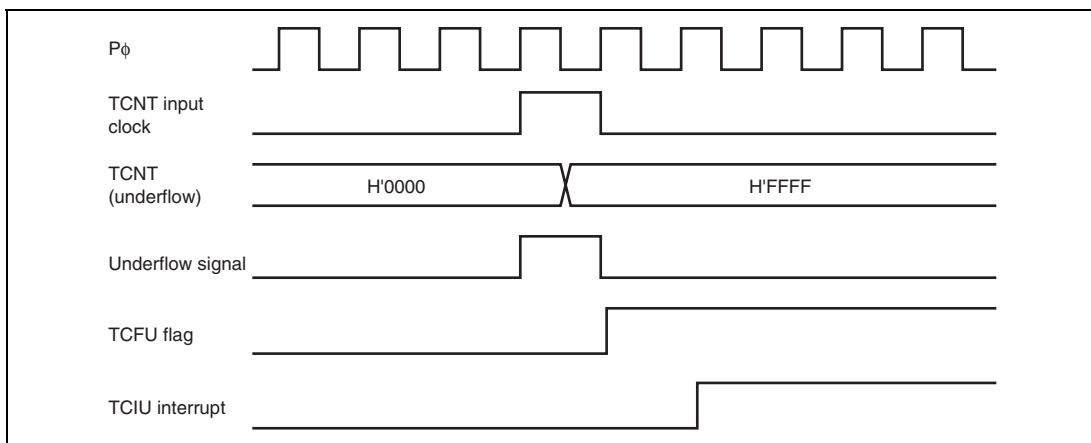
Figure 9.41   TCIU Interrupt Setting Timing

(4)   Status Flag Clearing Timing

After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DMAC is activated, the flag is cleared automatically. Figure 9.42 shows the timing for status flag clearing by the CPU, and figures 9.43 and 9.44 show the timing for status flag clearing by the DMAC.
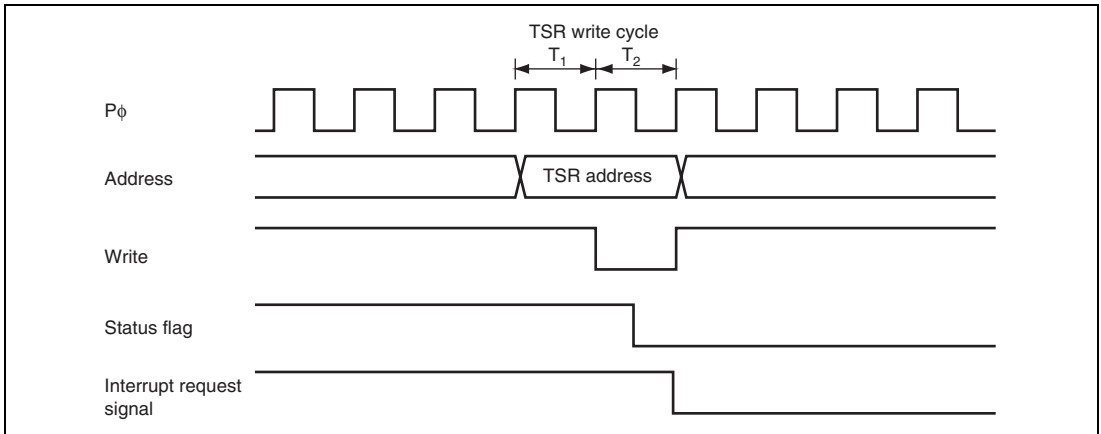
Figure 9.42   Timing for Status Flag Clearing by CPU

The status flag and interrupt request signal are cleared in synchronization with Pϕ after the DMAC transfer has started, as shown in figure 9.43. If conflict occurs for clearing the status flag and interrupt request signal due to activation of multiple DMAC transfers, it will take up to five clock cycles (Pϕ) for clearing them, as shown in figure 9.44. The next transfer request is masked for a longer period of either a period until the current transfer ends or a period for five clock cycles (Pϕ) from the beginning of the transfer.
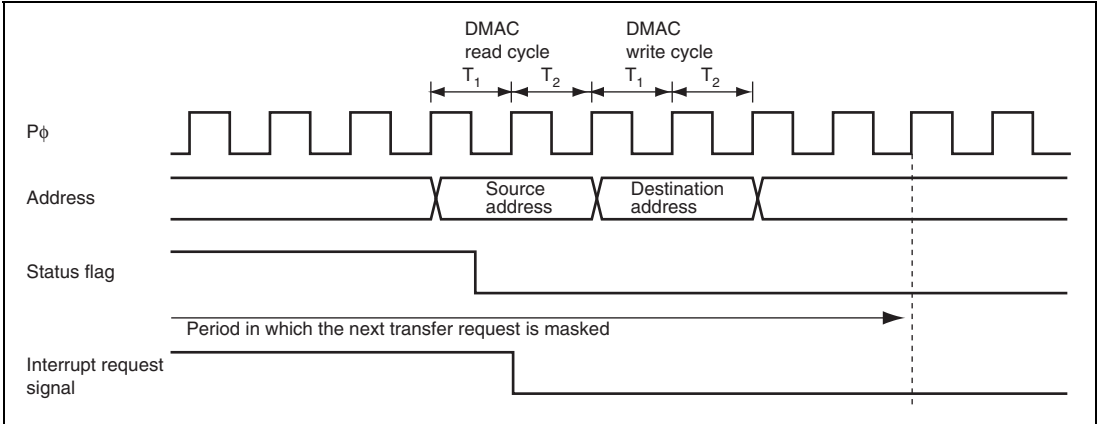
Figure 9.43   Timing for Status Flag Clearing by DMAC Activation (1)
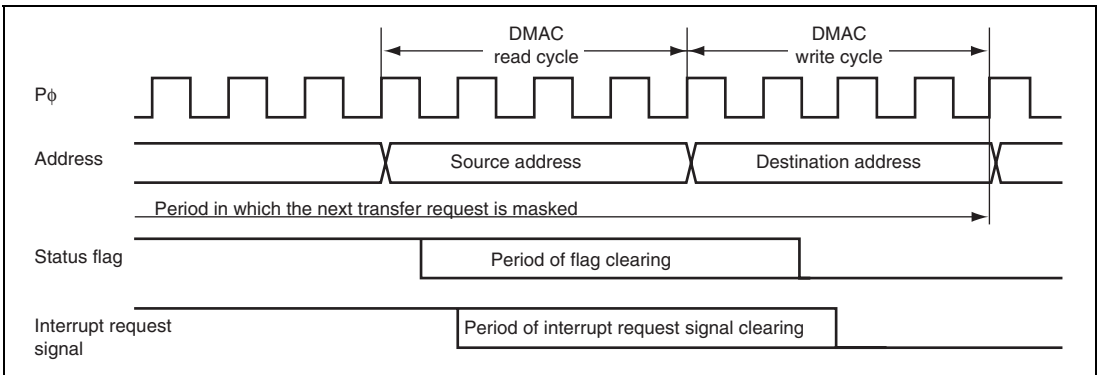
Figure 9.44   Timing for Status Flag Clearing by DMAC Activation (2)

9.9   Usage Notes

9.9.1   Module Stop Mode Setting

Operation of the TPU can be disabled or enabled using the module stop control register. The initial setting is for operation of the TPU to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 24, Power-Down Modes.

9.9.2   Input Clock Restrictions

The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width.

In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 9.45 shows the input clock conditions in phase counting mode.
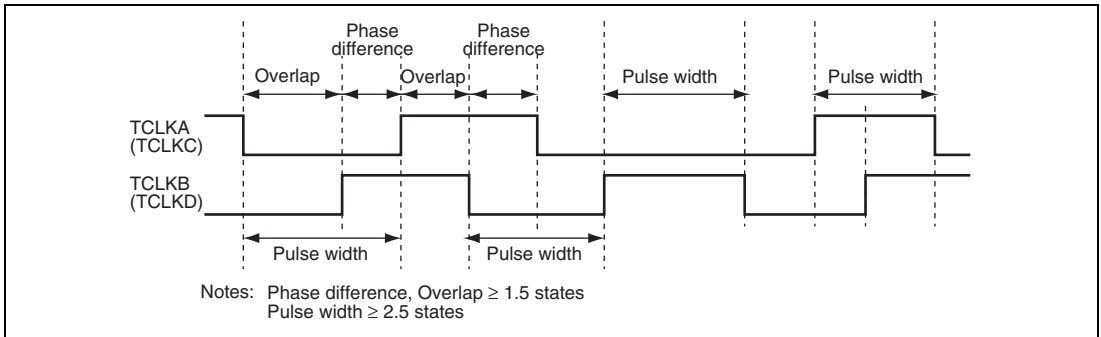
Figure 9.45   Phase Difference, Overlap, and Pulse Width in Phase Counting Mode

9.9.3   Caution on Cycle Setting

When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

$$f = \frac{P\phi}{(N + 1)}$$

f: Counter frequency
Pϕ: Operating frequency
N: TGR set value

9.9.4   Conflict between TCNT Write and Clear Operations

If the counter clearing signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed. Figure 9.46 shows the timing in this case.
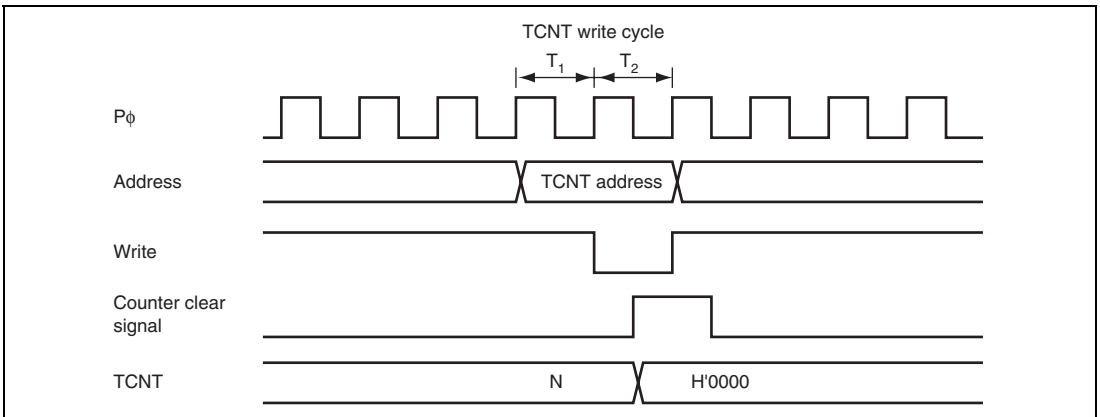
Figure 9.46   Conflict between TCNT Write and Clear Operations

9.9.5   Conflict between TCNT Write and Increment Operations

If incrementing occurs in the T2 state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented. Figure 9.47 shows the timing in this case.

RENESAS
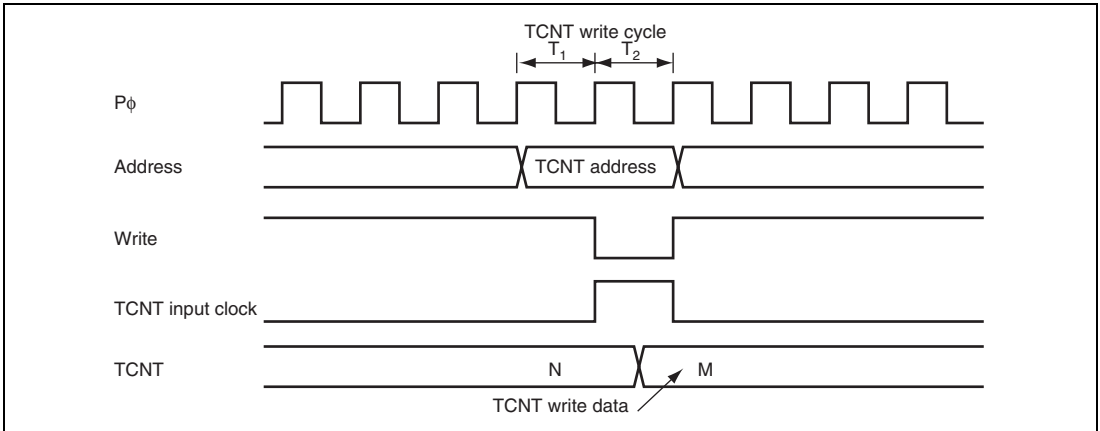
Figure 9.47   Conflict between TCNT Write and Increment Operations

9.9.6   Conflict between TGR Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the TGR write takes precedence and the compare match signal is disabled. A compare match also does not occur when the same value as before is written.

Figure 9.48 shows the timing in this case.
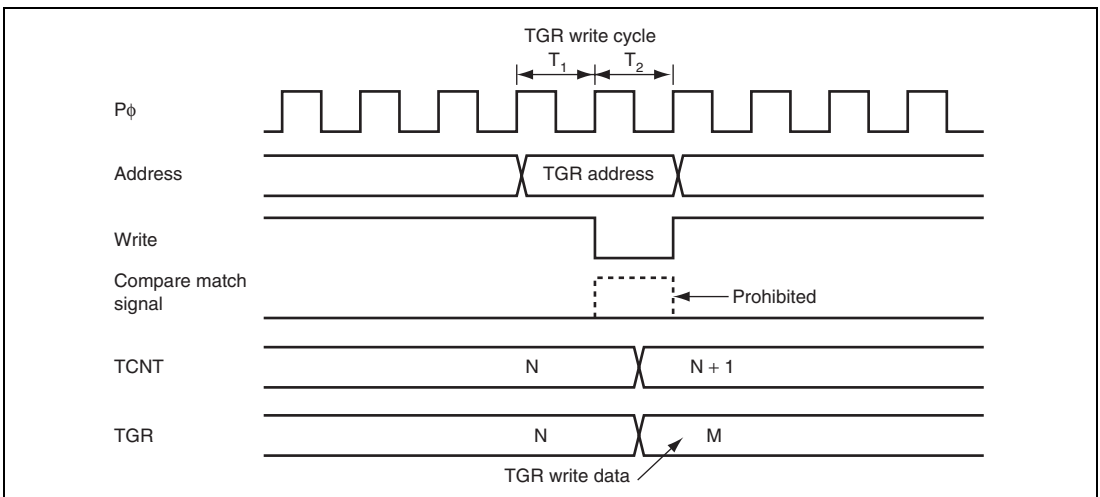
Figure 9.48   Conflict between TGR Write and Compare Match

9.9.7 Conflict between Buffer Register Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the write data.

Figure 9.49 shows the timing in this case.
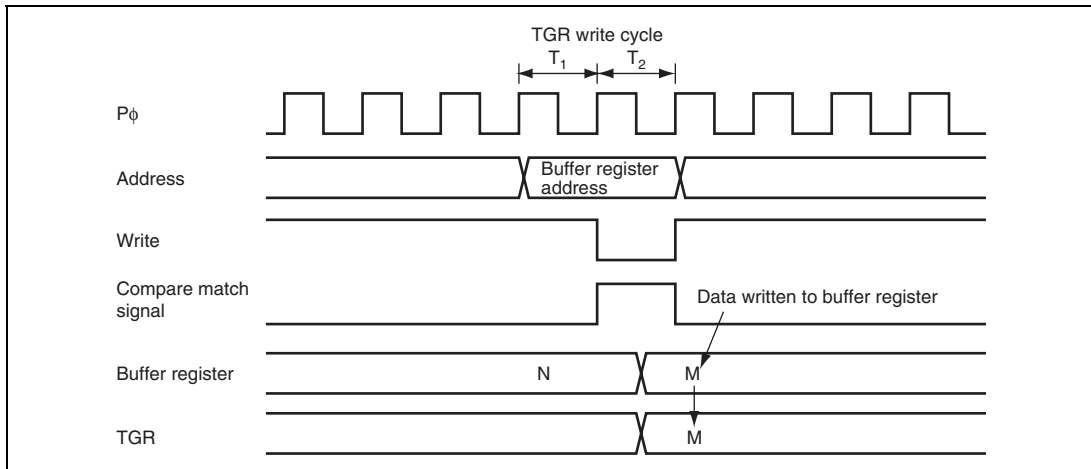
Figure 9.49   Conflict between Buffer Register Write and Compare Match

9.9.8 Conflict between TGR Read and Input Capture

If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is read will be the data after input capture transfer.

Figure 9.50 shows the timing in this case.
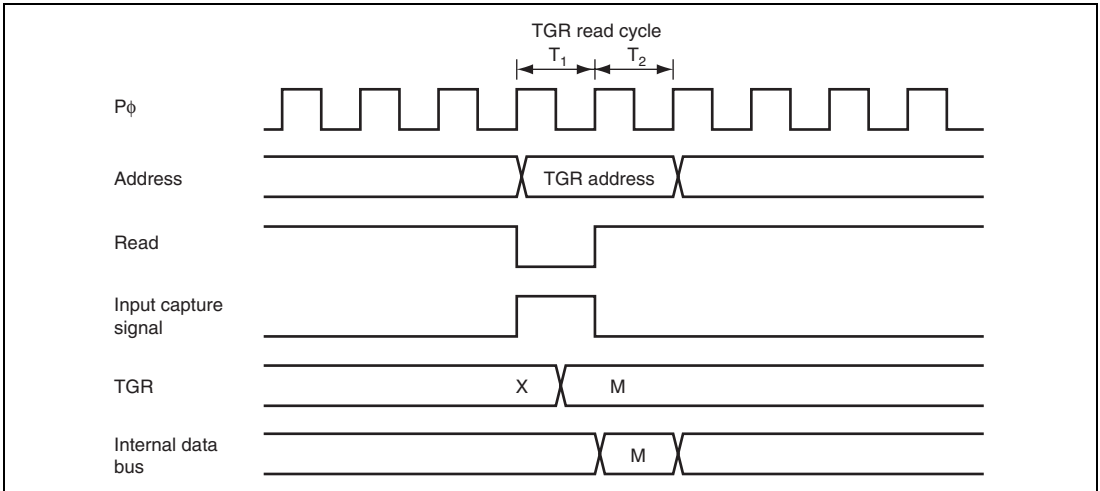
Figure 9.50   Conflict between TGR Read and Input Capture

9.9.9   Conflict between TGR Write and Input Capture

If the input capture signal is generated in the T2 state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

Figure 9.51 shows the timing in this case.
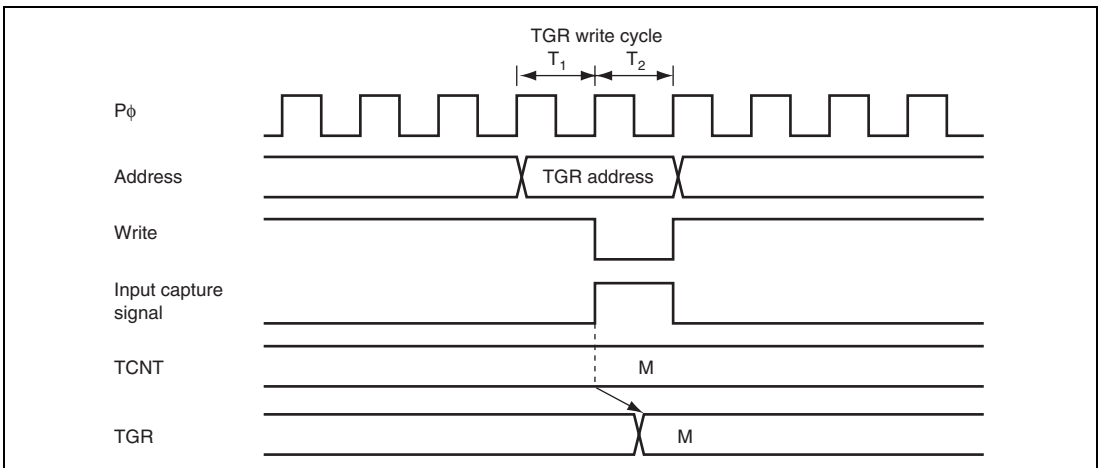
Figure 9.51   Conflict between TGR Write and Input Capture

9.9.10   Conflict between Buffer Register Write and Input Capture

If the input capture signal is generated in the T2 state of a buffer register write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

Figure 9.52 shows the timing in this case.
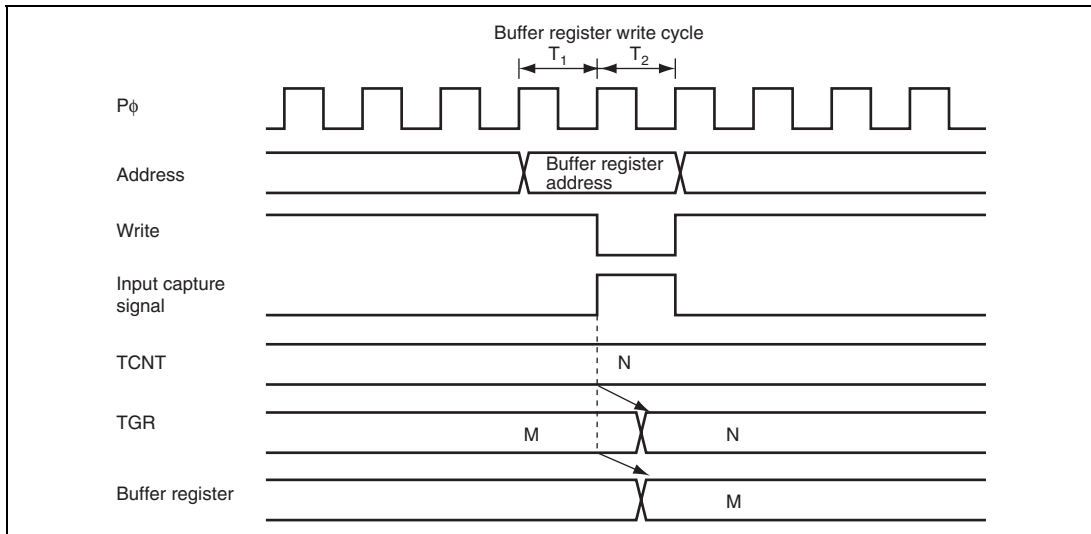
Figure 9.52   Conflict between Buffer Register Write and Input Capture

9.9.11   Conflict between Overflow/Underflow and Counter Clearing

If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

Figure 9.53 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.
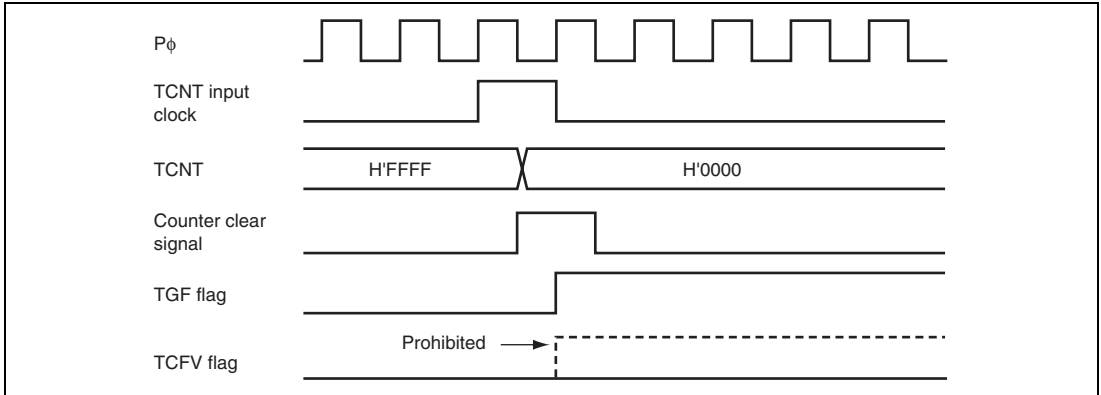
Figure 9.53   Conflict between Overflow and Counter Clearing

9.9.12   Conflict between TCNT Write and Overflow/Underflow

If an overflow/underflow occurs due to increment/decrement in the T2 state of a TCNT write cycle, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 9.54 shows the operation timing when there is conflict between TCNT write and overflow.
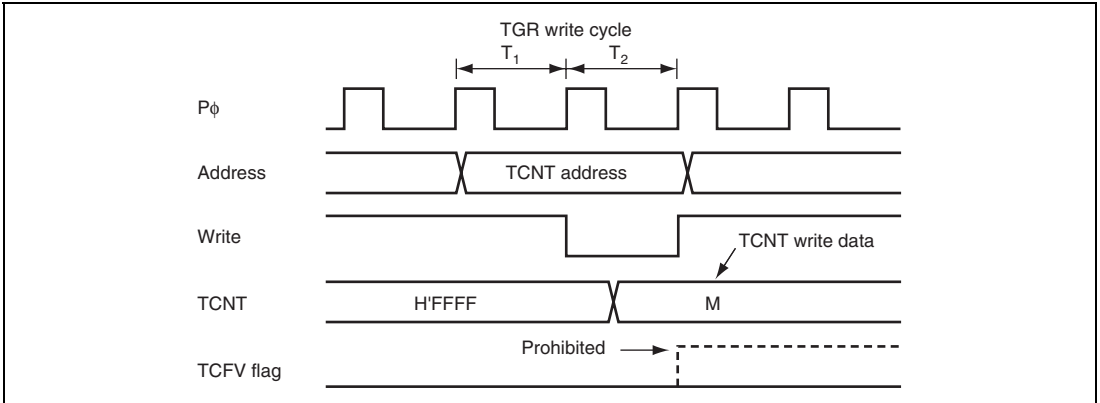
Figure 9.54   Conflict between TCNT Write and Overflow

9.9.13   Multiplexing of I/O Pins

In this LSI, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

9.9.14   Interrupts and Module Stop Mode

If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC activation source. Interrupts should therefore be disabled before entering module stop mode.

Section 10   Watchdog Timer (WDT)

The watchdog timer (WDT) is an 8-bit timer that outputs an internal reset signal if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

Figure 10.1 shows a block diagram of the WDT.

10.1   Features

- Selectable from eight counter input clocks
- Switchable between watchdog timer mode and interval timer mode
  — In watchdog timer mode
    If the counter overflows, this LSI can be initialized internally.
  — In interval timer mode
    If the counter overflows, the WDT generates an interval timer interrupt (WOVI).
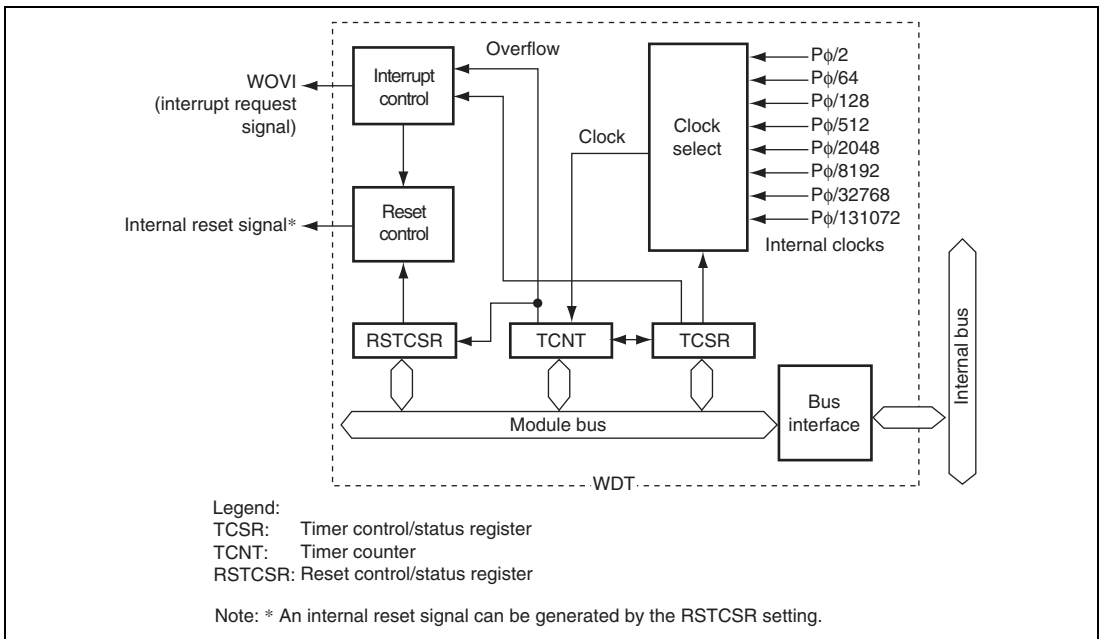
Figure 10.1   Block Diagram of WDT

10.2   Register Descriptions

The WDT has the following three registers. To prevent accidental overwriting, TCSR, TCNT, and RSTCSR have to be written to in a method different from normal registers. For details, see section 10.5.1, Notes on Register Access.

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

10.2.1   Timer Counter (TCNT)

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

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

10.2.2   Timer Control/Status Register (TCSR)

TCSR selects the clock source to be input to TCNT, and the timer mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	OVF	WT/$\overline{\text{IT}}$	TME	—	—	CKS2	CKS1	CKS0
Initial Value	0	0	0	1	1	0	0	0
R/W	R/(W)*	R/W	R/W	R	R	R/W	R/W	R/W

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

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*	Overflow Flag
				Indicates that TCNT has overflowed in interval timer mode. Only 0 can be written to this bit, to clear the flag.
				[Setting condition]
				• When TCNT overflows in interval timer mode (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 condition]
				• Cleared by reading TCSR when OVF = 1, then writing 0 to OVF
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
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
				When TCNT overflows, an interval timer interrupt (WOVI) is requested.
				1: Watchdog timer mode
				When TCNT overflows while RSTE = 1, this LSI is initialized initially.
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, 3	—	All 1	R	Reserved
				These are read-only bits and cannot be modified.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	Select the clock source to be input to TCNT. The overflow
0	CKS0	0	R/W	cycle for Pϕ = 20 MHz is indicated in parentheses.
				000: Clock Pϕ/2 (cycle: 25.6 μs)
				001: Clock Pϕ/64 (cycle: 819.2 μs)
				010: Clock Pϕ/128 (cycle: 1.6 ms)
				011: Clock Pϕ/512 (cycle: 6.6 ms)
				100: Clock Pϕ/2048 (cycle: 26.2 ms)
				101: Clock Pϕ/8192 (cycle: 104.9 ms)
				110: Clock Pϕ/32768 (cycle: 419.4 ms)
				111: Clock Pϕ/131072 (cycle: 1.68 s)

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

RENESAS

10.2.3   Reset Control/Status Register (RSTCSR)

RSTCSR controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal. RSTCSR is initialized to H'1F by a reset signal from the $\overline{\text{RES}}$ pin, but not by the WDT internal reset signal caused by WDT overflows.

Bit	7	6	5	4	3	2	1	0
Bit Name	WOVF	RSTE	—	—	—	—	—	—
Initial Value	0	0	0	1	1	1	1	1
R/W	R/(W)*	R/W	R/W	R	R	R	R	R

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

Bit	Bit Name	Initial Value	R/W	Description
7	WOVF	0	R/(W)*	Watchdog Timer Overflow Flag
				This bit is set when TCNT overflows in watchdog timer mode. This bit cannot be set in interval timer mode, and only 0 can be written.
				[Setting condition]
				• When TCNT overflows (changed from H'FF to H'00) in watchdog timer mode
				[Clearing condition]
				• Reading RSTCSR when WOVF = 1, and then writing 0 to WOVF
6	RSTE	0	R/W	Reset Enable
				Specifies whether or not this LSI is internally reset if TCNT overflows during watchdog timer operation.
				0: LSI is not reset even if TCNT overflows (Though this LSI is not reset, TCNT and TCSR in WDT are reset)
				1: LSI is reset if TCNT overflows
5	—	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.
4 to 0	—	All 1	R	Reserved
				These are read-only bits and cannot be modified.

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

RENESAS

10.3   Operation

10.3.1   Watchdog Timer Mode

To use the WDT in watchdog timer mode, set both the WT/$\overline{\text{IT}}$ and TME bits in TCSR to 1.

When TCNT overflows in watchdog timer mode, the WOVF bit in RSTCSR is set to 1.

When the watchdog timer mode is selected and the RSTE bit in RSTCSR is set to 1, if TCNT overflows without being rewritten because of a system crash or other error, this LSI is initialized internally. This ensures that TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally H'00 is written) before overflow occurs.

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 (TCNT has overflowed), the $\overline{\text{RES}}$ pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.

The internal reset signal is output for 519 cycles of Pφ.

When RSTE = 1, a signal to initialize this LSI internally is generated. Since this signal initializes the system click control register (SCKCR), the multiplication ratio of P φ clock is also initialized.

When RSTE = 0, the signal is not generated, meaning that the SCKCR value and multiplication ratio of P φ clock remain unchanged.
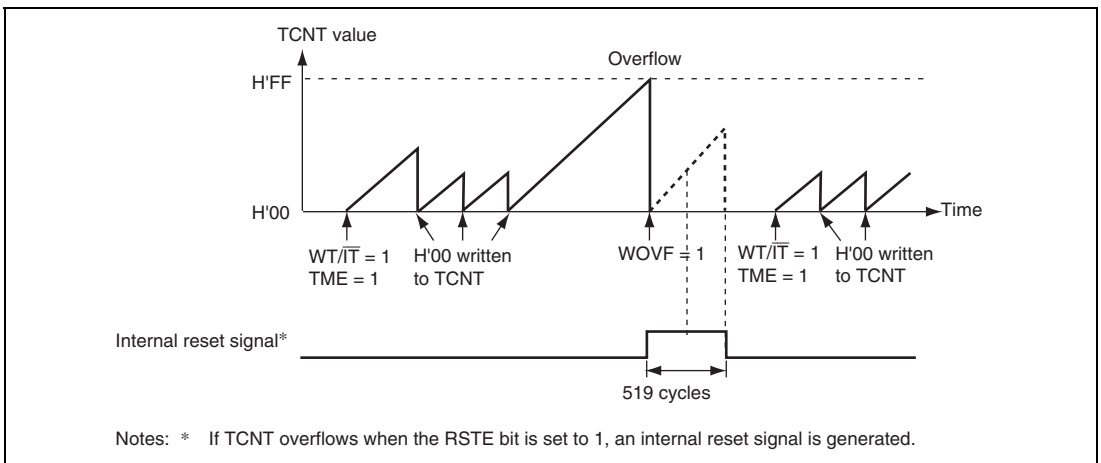
Figure 10.2   Operation in Watchdog Timer Mode

10.3.2   Interval Timer Mode

To use the WDT as an interval timer, set the WT/$\overline{\text{IT}}$ bit to 0 and the TME bit to 1 in TCSR.

When the WDT is used as an interval timer, an interval timer interrupt (WOVI) is generated each time the TCNT overflows. Therefore, an interrupt can be generated at intervals.

When the TCNT overflows in interval timer mode, an interval timer interrupt (WOVI) is requested at the same time the OVF bit in the TCSR is set to 1.
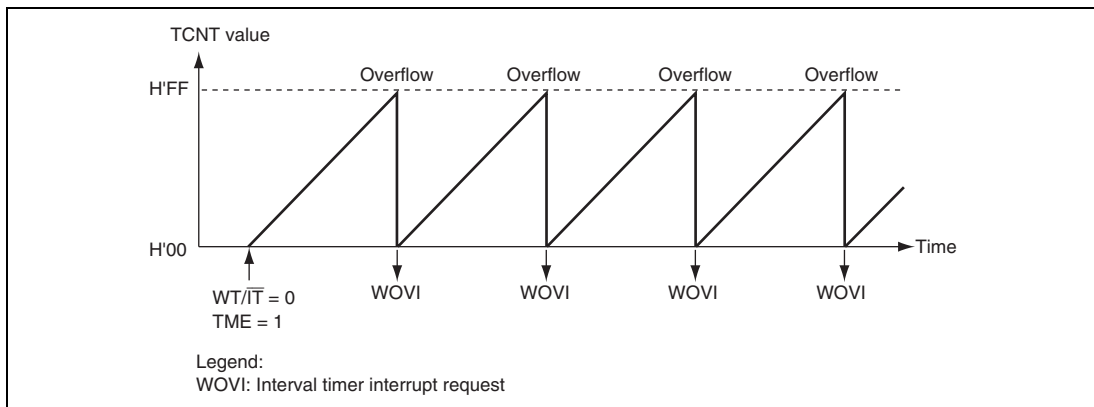
Figure 10.3   Operation in Interval Timer Mode

10.4   Interrupt Source

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR. The OVF flag must be cleared to 0 in the interrupt handling routine.

Table 10.1   WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag
WOVI	TCNT overflow	OVF

10.5   Usage Notes

10.5.1   Notes on Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in being more difficult to write to. The procedures for writing to and reading these registers are given below.

(1)   Writing to TCNT, TCSR, and RSTCSR

TCNT and TCSR must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

For writing, TCNT and TCSR are assigned to the same address. Accordingly, perform data transfer as shown in figure 10.4. The transfer instruction writes the lower byte data to TCNT or TCSR.

To write to RSTCSR, execute a word transfer instruction for address H'FFA6. A byte transfer instruction cannot be used to write to RSTCSR.

The method of writing 0 to the WOVF bit in RSTCSR differs from that of writing to the RSTE bit in RSTCSR. Perform data transfer as shown in figure 10.4.

At data transfer, the transfer instruction clears the WOVF bit to 0, but has no effect on the RSTE bit. To write to the RSTE bit, perform data transfer as shown in figure 10.4. In this case, the transfer instruction writes the value in bit 6 of the lower byte to the RSTE bit, but has no effect on the WOVF bit.

Figure 10.4   Writing to TCNT, TCSR, and RSTCSR

(2)   Reading from TCNT, TCSR, and RSTCSR

These registers can be read from in the same way as other registers. For reading, TCSR is assigned to address H'FFA4, TCNT to address H'FFA5, and RSTCSR to address H'FFA7.

10.5.2   Conflict between Timer Counter (TCNT) Write and Increment

If a TCNT clock pulse is generated during the T2 state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 10.5 shows this operation.
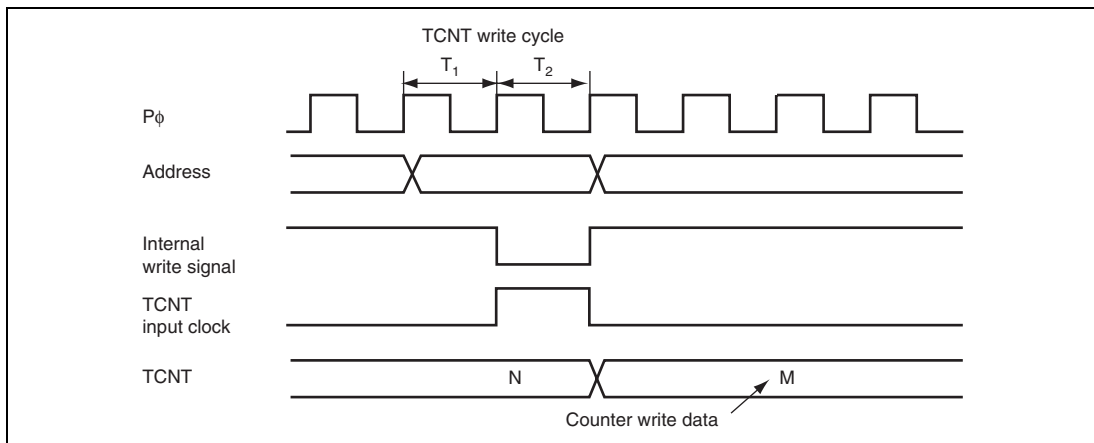
Figure 10.5   Conflict between TCNT Write and Increment

10.5.3   Changing Values of Bits CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before the values of bits CKS2 to CKS0 are changed.

10.5.4   Switching between Watchdog Timer Mode and Interval Timer Mode

If the timer mode is switched from watchdog timer mode to interval timer mode while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before switching the timer mode.

RENESAS

10.5.5   Transition to Watchdog Timer Mode or Software Standby Mode

When the WDT operates in watchdog timer mode, a transition to software standby mode is not made even when the SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1. Instead, a transition to sleep mode is made.

To transit to software standby mode, the SLEEP instruction must be executed after halting the WDT (clearing the TME bit to 0).

When the WDT operates in interval timer mode, a transition to software standby mode is made through execution of the SLEEP instruction when the SSBY bit in SBYCR is set to 1.

Section 11   Watch Timer (WAT)

The watch timer (WAT) is comprised of a counter that operates on the system clock and a 16-bit timer that operates on the subclock. Figure 11.1 shows a block diagram of the WAT.

11.1   Features

- Counter input clock selectable from eight types of main clocks and one type of subclock.
- Selectable modes include compare match timer mode and interval timer mode.
- Compare match timer mode
  — The compare match cycle can be changed using the watch timer constant register (WTCOR).
  — WCMI interrupts are generated when the watch timer counter (WTCNT) and the watch timer constant register (WTCOR) match.
- Interval timer mode
  — The WCM interrupt is generated whenever watch timer counter (WTCNT) overflow occurs.
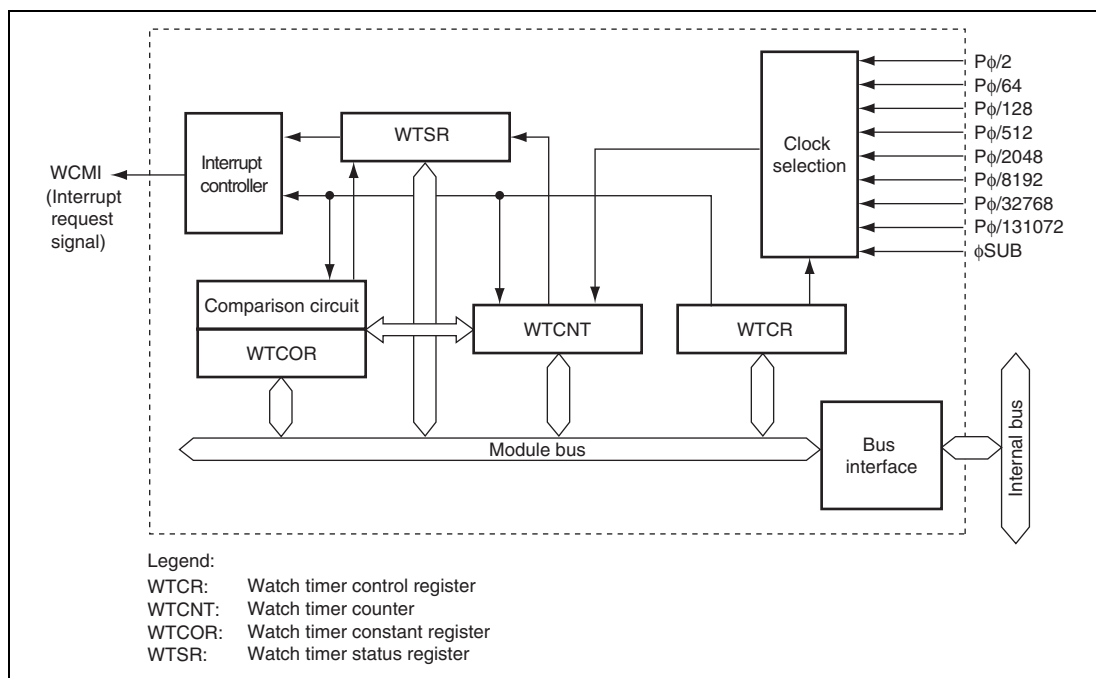
Figure 11.1   Block Diagram of WAT

11.2   Register Descriptions

The WAT has the following registers.

- Watch timer counter (WTCNT)
- Watch timer control register (WTCR)
- Watch timer status register (WTSR)
- Watch timer constant register (WTCOR)

11.2.1   Watch Timer Counter (WTCNT)

WTCNT is a 16-bit readable/writable up-counter. When the TME bit in WTCR is set to 1, WTCR starts incrementing on the internal clock selected by bits CKS2 to CKS0 and PSS in WTCR.

WTCNT is initialized to H'0000 when the TME bit in WTCR is cleared to 0.

Bit:	15	14	13	12	11	10	9	8
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

11.2.2   Watch Timer Control Register (WTCR)

WTCR selects the input clock for WTCNT and mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	CMT/$\overline{\text{IT}}$	TME	PSS	IE	CKS2	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	—	Reserved
				This bit is always read as 0. The write value should also be 0.
6	CMT/$\overline{\text{IT}}$	0	R/W	Timer Mode Select
				Selects whether the WAT is used as a compare match timer or an interval timer.
				0: Compare match timer mode
				1: Interval timer mode
5	TME	0	R/W	Timer Enable
				Starts WTCNT counting when this bit is set to 1. When this bit is cleared, WTCNT stops counting and is initialized to H'0000∗.
4	PSS	0	R/W	Counter Select
				Selects the base clock for WTCNT.
				This bit must be set to 1 when the WAT is operated as the watch timer.
				0: Operated on a Pϕ base clock
				1: Operated on a ϕSUB base clock.
3	IE	0	R/W	Interrupt Enable
				0: Disables Interrupts
				1: Enables compare-match or overflow flag interrupts

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	These bits select a clock input to WTCNT.
0	CKS0	0	R/W	Description in parentheses ( ) indicates the clock status when the PSS bit is 1.
				000: Pϕ/2 (ϕSUB)
				001: Pϕ/64 (Stopped)
				010: Pϕ/128 (Stopped)
				011: Pϕ/512 (Stopped)
				100: Pϕ/2048 (Stopped)
				101: Pϕ/8192 (Stopped)
				110: Pϕ/32768 (Stopped)
				111: Pϕ/131072 (Stopped)

Note: ∗   After the TME bit change from 1 to 0 is detected, WTCNT is initialized to H'0000.

11.2.3   Watch Timer Status Register (WTSR)

The WTSR consists of several status flags.

Bit:	7	6	5	4	3	2	1	0
Bit name:	CMF/OVF	—	—	—	—	—	WTCNT_WF	WTCR_WF
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)∗	—	—	—	—	—	R	R

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

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
7	CMF/OVF	0	R/(W)*	Compare Match/Overflow Flag
				Indicates if a compare match or an overflow occurs on WTCNT.
				[Setting conditions]
				• When WTCOR and WTCNT values match in compare match mode.
				• When WTCNT overflows
				[Clearing condition]
				• When 0 is written to after WTSR is read from with CMF/OVF = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
6 to 2	—	All 0	—	Reserved
				These bits are always read as 0. The write value should always be 0.
1	WTCNT_WF	0	R	Timer Counter Write Flag
				The flag is set to 1 when rewriting WTCNT is started. The flag is cleared to 0 when rewriting is completed.
				WTCNT must be read from or written to when this flag is 0. If WTCNT is written to when this flag is 1, the WTCNT value is not modified.
				[Setting condition]
				• When rewriting WTCNT is started (during rewriting)
				[Clearing condition]
				• When rewriting WTCNT is completed

Bit	Bit Name	Initial Value	R/W	Description
0	WTCR_WF	0	R	Timer Control Register Write Flag
				The flag is set to 1 when rewriting WTCR is started. The flag is cleared to 0 when rewriting is completed.
				WTCR must be read from or written to when this flag is 0. If WTCR is written to when this flag is 1, the WTCR value is not modified.
				[Setting condition]
				• When rewriting WTCR is started (during rewriting)
				[Clearing condition]
				• When rewriting WTCR is completed

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

11.2.4   Watch Timer Constant Register (WTCOR)

WTCOR is a 16-bit readable/writable register. When the WTCOR value matches the WTCNT value in compare match mode, the compare match flag is set. In interval timer mode, the compare match flag is not set even though the WTCOR value matches the WTCNT value.

Bit:	15	14	13	12	11	10	9	8
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

RENESAS

11.3 Operation

11.3.1 Mode Operation in Compare Match Timer

Follow the procedure below to select the use as compare match timer mode.

(1) Initial Settings

Make sure that WTCNT is stopped (the TME bit in WTCR is 0) for the following settings.

- Specify the timer cycle with WTCOR.
- Enter the initial values for WTCNT.
- The CMT/$\overline{\text{IT}}$, PSS, IE, and CKS2 to CKS0 bits in WTCR are used to set timer operating mode, operating clock and interrupt enable/disable status. Clearing the XTALSTP bit in SUBCKCR to 0 enables WTCNT operation to continue in software standby mode. For details of SUBCKCR, see section 23, Clock Pulse Generator.

(2) Activation

- The TME bit in WTCR is set to 1 after confirming that the WTCR value has been rewritten (the WTCR_WF bit in WTSR is 0).
- WTCNT begins incrementing once the WTCR value has been rewritten (the WTCR_WF bit in WTSR is 0).
- The CMF/OVF bit in WTSR is set to 1 each time the WTCNT and WTCOR values match.
- A timer interrupt is generated if the IE bit in WTCR is set to 1.

(3) Stopping

- If the TME bit in WTCR is set to 0, WTCNT will stop incrementing and TCNT is initialized to H'0000 after WTCR is rewritten (the WTCR_WF bit in WTSR is 0).
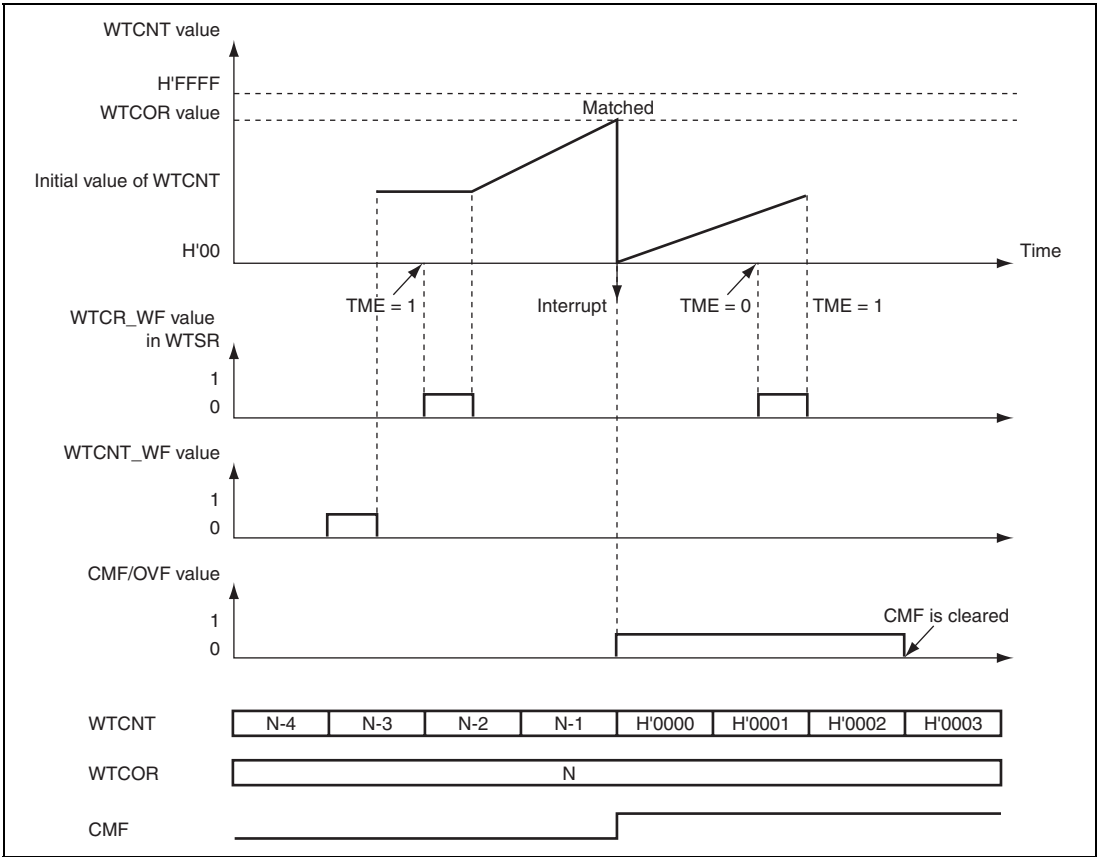
Figure 11.2   Operation in Compare Match Timer Mode

11.3.2   Operation in Interval Timer Mode

Follow the procedure below to select the use as interval timer mode.

(1)   Initial Settings

Make sure that WTCNT is stopped (the TME bit in WTCR is 0) for the following settings.

- Enter the initial values for WTCNT.
- The CMT/$\overline{\text{IT}}$, PSS, IE, and CKS2 to CKS0 bits in WTCR are used to set timer operating mode, operating clock, and interrupt enable/disable status. Clearing the XTALSTP bit in SUBCKCR to 0 enables WTCNT operation to continue in software standby mode. For details of SUBCKCR, see section 23, Clock Pulse Generator.

(2)   Activation

- The TME bit in WTCR is set to 1 after confirming that the WTCR value has been rewritten (the WTCR_WF bit in WTSR is 0).
- WTCNT begins incrementing once the WTCR value has been rewritten (the WTCR_WF bit in WTSR is 0).
- The CMF/OVF bit in WTSR is set to 1 when the WTCNT value changes from H'FFFF to H'0000.
- A timer interrupt is generated if the IE bit in WTCR is set to 1.

(3)   Stopping

- If the TME bit in WTCR is cleared to 0, WTCNT will stop incrementing and WTCNT will be initialized to H'0000 after WTCR is rewritten (the WTCR_WF bit in WTSR is 0).
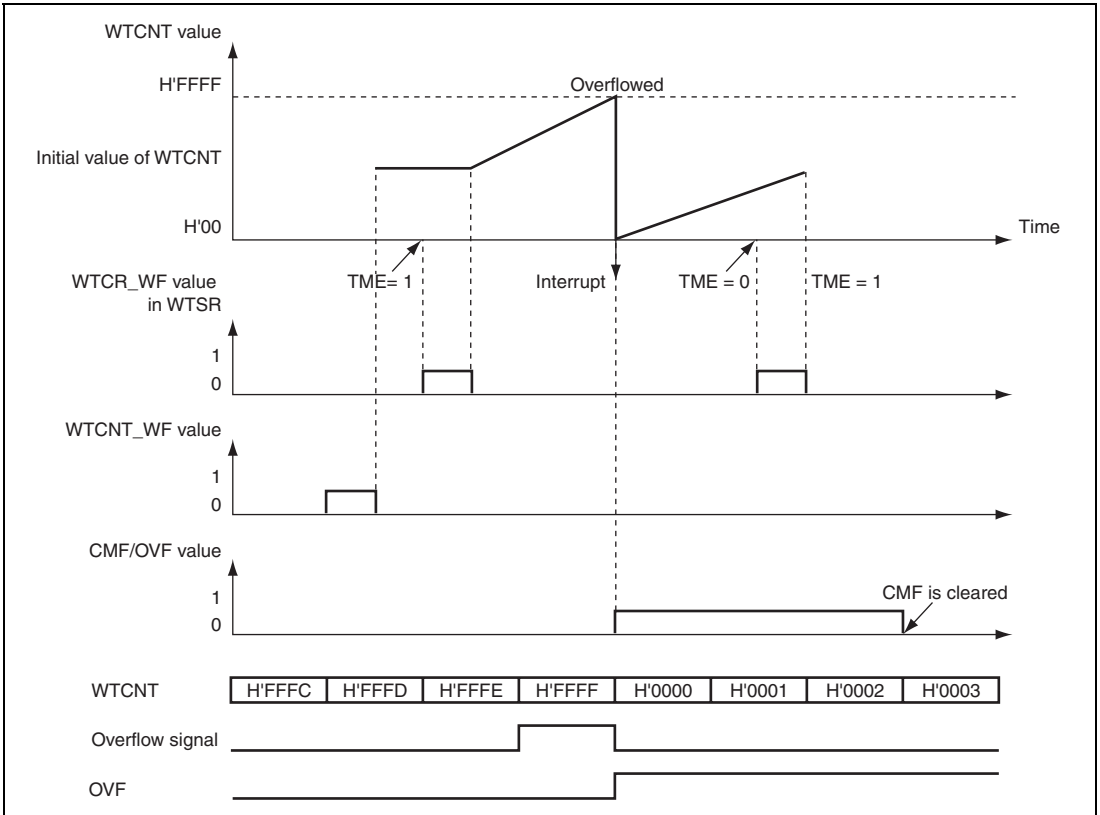
Figure 11.3   Operation in Interval Timer Mode

11.4   Usage Notes

11.4.1   Precautions for Accessing Registers

Due to the time required to rewrite the write values in WTCNT and WTCR, WTSR is provided to indicate the current write status.

(1)   Writing to WTCR

Make sure that the WTSR flag has been rewritten (WTCR_WF = 0) when writing to WTCR. If WTCR is written to when rewriting is in progress (WTCR_WF = 1), the register value is not modified.

(2)   Writing to WTCNT

Make sure that the WTSR flag has been rewritten (WTCNT_WF = 0) when writing to WTCNT. If WTCNT is written to when rewriting is in progress (WTCNT_WF = 1), the register value is not modified.

WTCNT includes two internal counters, one that operates on the system clock and the other on the subclock. The counter to be written to is selected by the PSS bit in WTCR. Therefore, when writing to WTCNT after rewriting the PSS bit in WTCR, make sure that both WTCR and WTCNT have been rewritten (WTCR_WF = 0 and WTCNT_WF = 0).

(3)   Reading from WTCNT or WTCR

When reading from WTCNT or WTCR, make sure that the WTSR flag has been rewritten (WTCR_WF = 0 or WTCNT_WF = 0).

If WTCNT or WTCR is read from during rewriting (WTCR_WF = 1 or WTCNT_WF = 1), the read value may be undefined.

WTCNT includes two internal counters, one that operates on the system clock and the other on the subclock. The counter to be read from is selected by the PSS bit in WTCR. Therefore, when reading from WTCNT after rewriting the PSS bit in WTCR, make sure that both WTCR and WTCNT have been rewritten (WTCR_WF = 0 and WTCNT_WF = 0).

11.4.2   Conflict between Write and Increment Processes of WTCNT

Even in instances where counter increments occur on the T2 state during the WTCNT write cycle, the increments will be ignored and writing to WTCNT will take precedence. This process is shown in figure 11.4.
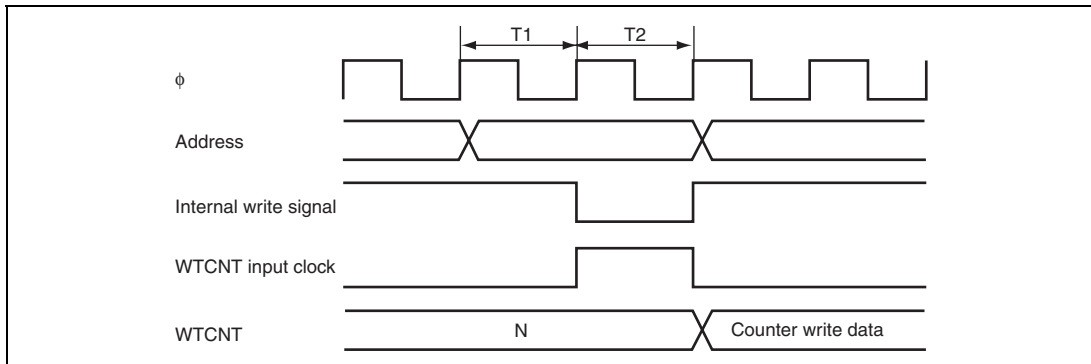
Figure 11.4   Conflict between WTCNT Write and Increment

11.4.3   Rewriting Bits CKS2 to CKS0

Note that incrementation errors may occur if bits CKS2 to CKS0 in WTCR are rewritten during counter operation. Always make sure to stop timer operation (clearing the TME bit to 0) prior to rewriting bits CKS2 to CKS0.

11.4.4   Switching between Compare Match Timer and Interval Timer Modes

Note that switching between compare match timer and interval timer modes during counter operation may result in operational error. Always make sure to stop timer operation (clearing the TME bit to 0) prior to switching the timer modes.

11.4.5   Rewriting PSS Bit

Note that rewriting the PSS bit during counter operation may result in operational error. Always make sure to stop timer operation (clearing the TME bit to 0) prior to rewriting the PSS bit.

11.4.6   WTCOR Setting Value and WTCNT Rewrite Value in Compare Match Timer Mode

Make sure the following condition is satisfied in compare match timer mode:

(WTCNT rewrite value) < (WTCOR setting value)

11.4.7   Interrupt Vector Address

The interrupt vector address used during normal operation and that of software standby mode differ. For details, see section 5, Interrupt Controller.

Section 12   Serial Communication Interface (SCI)

This LSI has four independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clocked synchronous serial communication. Asynchronous serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface conforming to ISO/IEC 7816-3 (Identification Card) as an extended asynchronous communication mode. Figure 12.1 shows a block diagram of the SCI.

12.1   Features

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

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

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

  The interrupt sources are transmit-end, transmit-data-empty, receive-data-full, and receive error. The transmit-data-empty and receive-data-full interrupt sources can activate the DMAC.
- Module stop mode can be set

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

Clocked Synchronous Mode:

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

Smart Card Interface:

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on receiving an error signal during transmission
- Both direct convention and inverse convention are supported
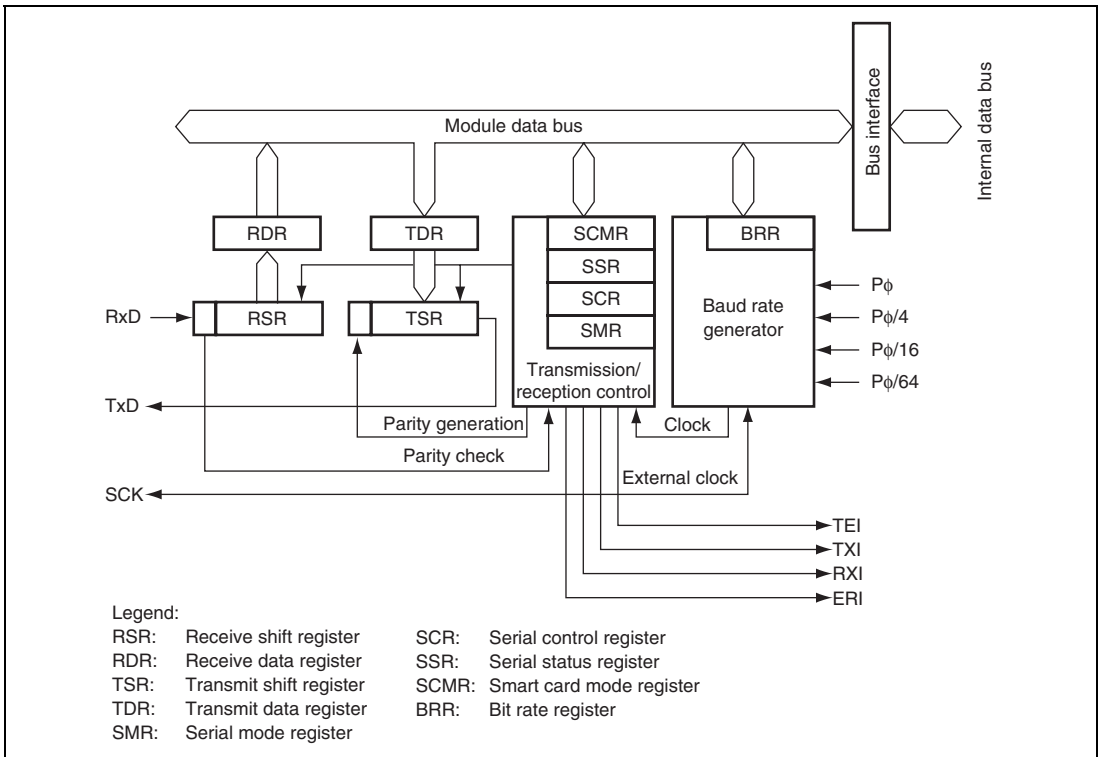
Figure 12.1   Block Diagram of SCI

12.2   Input/Output Pins

Table 12.1 lists the pin configuration of the SCI.

Table 12.1   Pin Configuration

Channel	Pin Name*	I/O	Function
0	SCK0	I/O	Channel 0 clock input/output
	RxD0	Input	Channel 0 receive data input
	TxD0	Output	Channel 0 transmit data output
2	SCK2	I/O	Channel 2 clock input/output
	RxD2	Input	Channel 2 receive data input
	TxD2	Output	Channel 2 transmit data output
4	SCK4	I/O	Channel 4 clock input/output
	RxD4	Input	Channel 4 receive data input
	TxD4	Output	Channel 4 transmit data output
5	SCK5	I/O	Channel 5 clock input/output
	RxD5	Input	Channel 5 receive data input
	TxD5	Output	Channel 5 transmit data output

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

12.3   Register Descriptions

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

Channel 0:

- Receive shift register_0 (RSR_0)
- Transmit shift register_0 (TSR_0)
- Receive data register_0 (RDR_0)
- Transmit data register_0 (TDR_0)
- Serial mode register_0 (SMR_0)
- Serial control register_0 (SCR_0)
- Serial status register_0 (SSR_0)
- Smart card mode register_0 (SCMR_0)
- Bit rate register_0 (BRR_0)

Channel 2:

- Receive shift register_2 (RSR_2)
- Transmit shift register_2 (TSR_2)
- Receive data register_2 (RDR_2)
- Transmit data register_2 (TDR_2)
- Serial mode register_2 (SMR_2)
- Serial control register_2 (SCR_2)
- Serial status register_2 (SSR_2)
- Smart card mode register_2 (SCMR_2)
- Bit rate register_2 (BRR_2)

RENESAS

Channel 4:

- Receive shift register_4 (RSR_4)
- Transmit shift register_4 (TSR_4)
- Receive data register_4 (RDR_4)
- Transmit data register_4 (TDR_4)
- Serial mode register_4 (SMR_4)
- Serial control register_4 (SCR_4)
- Serial status register_4 (SSR_4)
- Smart card mode register_4 (SCMR_4)
- Bit rate register_4 (BRR_4)

Channel 5:

- Receive shift register_5 (RSR_5)
- Transmit shift register_5 (TSR_5)
- Receive data register_5 (RDR_5)
- Transmit data register_5 (TDR_5)
- Serial mode register_5 (SMR_5)
- Serial control register_5 (SCR_5)
- Serial status register_5 (SSR_5)
- Smart card mode register_5 (SCMR_5)
- Bit rate register_5 (BRR_5)

RENESAS

12.3.1   Receive Shift Register (RSR)

RSR is a shift register which is used to receive serial data input from the RxD pin and converts it into parallel data. When one frame of data has been received, it is transferred to RDR automatically. RSR cannot be directly accessed by the CPU.

12.3.2   Receive Data Register (RDR)

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

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

12.3.3   Transmit Data Register (TDR)

TDR is an 8-bit register that stores transmit data. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structures of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR when one frame of data is transmitted, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read from or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR for only once after confirming that the TDRE bit in SSR is set to 1.

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

RENESAS

12.3.4   Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first automatically transfers transmit data from TDR to TSR, then sends the data to the TxD pin. TSR cannot be directly accessed by the CPU.

12.3.5   Serial Mode Register (SMR)

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

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	C/$\overline{\text{A}}$	CHR	PE	O/$\overline{\text{E}}$	STOP	MP	CKS1	CKS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	GM	BLK	PE	O/$\overline{\text{E}}$	BCP1	BCP0	CKS1	CKS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	C/$\overline{\text{A}}$	0	R/W	Communication Mode
				0: Asynchronous mode
				1: Clocked synchronous mode
6	CHR	0	R/W	Character Length (valid only in asynchronous mode)
				0: Selects 8 bits as the data length.
				1: Selects 7 bits as the data length. LSB-first is fixed and the MSB (bit 7) in TDR is not transmitted in transmission.
				In clocked synchronous mode, a fixed data length of 8 bits is used.
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode)
				When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. For a multiprocessor format, parity bit addition and checking are not performed regardless of the PE bit setting.
4	O/$\overline{\text{E}}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode)
				0: Selects even parity.
				1: Selects odd parity.
3	STOP	0	R/W	Stop Bit Length (valid only in asynchronous mode)
				Selects the stop bit length in transmission.
				0: 1 stop bit
				1: 2 stop bits
				In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.
2	MP	0	R/W	Multiprocessor Mode (valid only in asynchronous mode)
				When this bit is set to 1, the multiprocessor function is enabled. The PE bit and O/$\overline{\text{E}}$ bit settings are invalid in multiprocessor mode.

RENESAS

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: Pϕ clock (n = 0)
				01: Pϕ/4 clock (n = 1)
				10: Pϕ/16 clock (n = 2)
				11: Pϕ/64 clock (n = 3)
				For the relation between the settings of these bits and the baud rate, see section 12.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 12.3.9, Bit Rate Register (BRR)).

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	GM	0	R/W	GSM Mode
				Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 etu from the start and the clock output control function is appended. For details, see sections 12.7.6, Data Transmission (Except in Block Transfer Mode) and 12.7.8, Clock Output Control.
6	BLK	0	R/W	Setting this bit to 1 allows block transfer mode operation. For details, see section 12.7.3, 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 12.7.2, 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 12.7.4, Receive Data Sampling Timing and Reception Margin. S is described in section 12.3.9, 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: Pϕ clock (n = 0)
				01: Pϕ/4 clock (n = 1)
				10: Pϕ/16 clock (n = 2)
				11: Pϕ/64 clock (n = 3)
				For the relation between the settings of these bits and the baud rate, see section 12.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 12.3.9, Bit Rate Register (BRR)).

Note: etu: Elementary Time Unit (time for transfer of 1 bit)

12.3.6   Serial Control Register (SCR)

SCR is a register that enables/disables the following SCI transfer operations and interrupt requests, and selects the transfer clock source. For details on interrupt requests, see section 12.8, Interrupt Sources. Some bits in SCR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RENESAS

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable
				When this bit is set to 1, a TXI interrupt request is enabled.
				A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable
				When this bit is set to 1, RXI and ERI interrupt requests are enabled.
				RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.
5	TE	0	R/W	Transmit Enable
				When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format.
				If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.

Bit	Bit Name	Initial Value	R/W	Description
4	RE	0	R/W	Receive Enable
				When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format.
				Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the 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 12.5, Multiprocessor Communication Function.
				When receive data including MPB = 0 in SSR is being received, transfer of the received data from RSR to RDR, detection of reception errors, and the settings of RDRF, FER, and ORER flags in SSR are not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is automatically cleared to 0, and RXI and ERI interrupt requests (in the case where the TIE and RIE bits in SCR are set to 1) and setting of the FER and ORER flags are enabled.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
2	TEIE	0	R/W	Transmit End Interrupt Enable
				When this bit is set to 1, a TEI interrupt request is enabled. A TEI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0 in order to clear the TEND flag to 0, or by clearing the TEIE bit to 0.
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: On-chip baud rate generator
				(SCK pin functions as I/O port.)
				01: On-chip baud rate generator
				(Outputs a clock with the same frequency as the bit rate from the SCK pin.)
				1×: External clock
				(Inputs a clock with a frequency 16 times the bit rate from the SCK pin.)
				• Clocked synchronous mode
				0×: Internal clock
				(SCK pin functions as clock output.)
				1×: External clock
				(SCK pin functions as clock input.)

Note: ×: Don't care

RENESAS

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.
				A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable
				When this bit is set to 1, RXI and ERI interrupt requests are enabled.
				RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.
5	TE	0	R/W	Transmit Enable
				When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format.
				If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.
4	RE	0	R/W	Receive Enable
				When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format.
				Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the 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.

RENESAS

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 control the clock output from the SCK pin. In GSM mode, clock output can be dynamically switched. For details, see section 12.7.8, Clock Output Control.
				• When GM in SMR = 0
				00: Output disabled (SCK pin functions as I/O port.)
				01: Clock output
				1×: Reserved
				• When GM in SMR = 1
				00: Output fixed low
				01: Clock output
				10: Output fixed high
				11: Clock output

12.3.7   Serial Status Register (SSR)

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

• When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial Value	1	0	0	0	0	1	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

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

RENESAS

• When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	ERS	PER	TEND	MPB	MPBT
Initial Value	1	0	0	0	0	1	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

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

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	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 [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When a TXI interrupt request is issued allowing DMAC to write data to TDR (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial Value	R/W	Description
6	RDRF	0	R/(W)*	Receive Data Register Full
				Indicates whether 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 DMAC to read data from RDR
				The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.
				Note that when the next serial reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
5	ORER	0	R/(W)*	Overrun Error
				Indicates that an overrun error has occurred during reception and the reception ends abnormally.
				[Setting condition]
				• When the next serial reception is completed while RDRF = 1
				In RDR, receive data prior to an overrun error occurrence is retained, but data received after the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue.
				[Clearing condition]
				• When 0 is written to ORER after reading ORER = 1
				Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
4	FER	0	R/(W)*	Framing Error
				Indicates that a framing error has occurred during reception in asynchronous mode and the reception ends abnormally.
				[Setting condition]
				• When the stop bit is 0
				In 2-stop-bit mode, only the first stop bit is checked whether it is 1 but the second stop bit is not checked. Note that receive data when the framing error occurs is transferred to RDR, however, the RDRF flag is not set. In addition, when the FER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.
				[Clearing condition]
				• When 0 is written to FER after reading FER = 1
				Even when the RE bit in SCR is cleared, the FER flag is not affected and retains its previous value. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	PER	0	R/(W)*	Parity Error
				Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally.
				[Setting condition]
				• When a parity error is detected during reception
				Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.
				[Clearing condition]
				• When 0 is written to PER after reading PER = 1
				Even when the RE bit in SCR is cleared, the PER bit is not affected and retains its previous value. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
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 transmit character
				[Clearing conditions]
				• When 0 is written to TDRE after reading TDRE = 1
				• When a TXI interrupt request is issued allowing DMAC to write data to TDR (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	MPB	0	R	Multiprocessor Bit
				Stores the multiprocessor bit value 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
				Sets the multiprocessor bit value to be added to the transmit frame.

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

RENESAS

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty
				Indicates whether TDR contains transmit data.
				[Setting conditions]
				• When the TE bit in SCR is 0
				• When data is transferred from TDR to TSR
				[Clearing conditions]
				• When 0 is written to TDRE after reading TDRE = 1
				• When a TXI interrupt request is issued allowing DMAC to write data to TDR
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
6	RDRF	0	R/(W)*	Receive Data Register Full
				Indicates whether 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 DMAC to read data from RDR
				The RDRF flag is not affected and retains its previous value even when the RE bit in SCR is cleared to 0.
				Note that when the next reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error
				Indicates that an overrun error has occurred during reception and the reception ends abnormally.
				[Setting condition]
				• When the next serial reception is completed while RDRF = 1
				In RDR, the receive data prior to an overrun error occurrence is retained, but data received following the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue.
				[Clearing condition]
				• When 0 is written to ORER after reading ORER = 1
				Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
4	ERS	0	R/(W)*	Error Signal Status
				[Setting condition]
				• When a low error signal is sampled
				[Clearing condition]
				• When 0 is written to ERS after reading ERS = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error
				Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally.
				[Setting condition]
				• When a parity error is detected during reception
				Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue.
				[Clearing condition]
				• When 0 is written to PER after reading PER = 1
				Even when the RE bit in SCR is cleared, the PER flag is not affected and retains its previous value. (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
2	TEND	1	R	Transmit End
				This bit is set to 1 when no error signal is sent from the receiving side and the next transmit data is ready to be transferred to TDR.
				[Setting conditions]
				• When both the TE and ERS bits in SCR are 0
				• When ERS = 0 and TDRE = 1 after a specified time passed after completion of 1-byte data transfer. The set timing depends on the register setting as follows:
				When GM = 0 and BLK = 0, 2.5 etu after transmission start
				When GM = 0 and BLK = 1, 1.5 etu after transmission start
				When GM = 1 and BLK = 0, 1.0 etu after transmission start
				When GM = 1 and BLK = 1, 1.0 etu after transmission start
				[Clearing conditions]
				• When 0 is written to TEND after reading TEND = 1
				• When a TXI interrupt request is issued allowing DMAC to write the next data to TDR (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	MPB	0	R	Multiprocessor Bit
				Not used in smart card interface mode.
0	MPBT	0	R/W	Multiprocessor Bit Transfer
				Write 0 to this bit in smart card interface mode.

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

12.3.8   Smart Card Mode Register (SCMR)

SCMR selects smart card interface mode and its format.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	—	Reserved
				These bits are always read as 1.
3	SDIR	0	R/W	Smart Card Data Transfer Direction
				Selects the serial/parallel conversion format.
				0: Transfer with LSB-first
				1: Transfer with MSB-first
				This bit is valid only when the 8-bit data format is used for transmission/reception; when the 7-bit data format is used, data is always transmitted/received with LSB-first.
2	SINV	0	R/W	Smart Card Data Invert
				Inverts the transmit/receive data logic level. This bit does not affect the logic level of the parity bit. To invert the parity bit, invert the O/$\overline{\text{E}}$ 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	—	Reserved
				This bit is always read as 1.
0	SMIF	0	R/W	Smart Card Interface Mode Select
				When this bit is set to 1, smart card interface mode is selected.
				0: Normal asynchronous or clocked synchronous mode
				1: Smart card interface mode

RENESAS

12.3.9   Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 12.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clocked synchronous mode, 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 12.2   Relationships between N Setting in BRR and Bit Rate B

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

Legend:

B: Bit rate (bit/s)

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

Pϕ: Operating frequency (MHz)

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

SMR Setting				SMR Setting		
CKS1	CKS0	n		BCP1	BCP0	S
0	0	0		0	0	32
0	1	1		0	1	64
1	0	2		1	0	372
1	1	3		1	1	256

Table 12.3 shows sample N settings in BRR in normal asynchronous mode. Table 12.4 shows the maximum bit rate settable for each operating frequency. Tables 12.6 and 12.8 show sample N settings in BRR in clocked synchronous mode and smart card interface mode, respectively. In smart card interface mode, the number of basic clock cycles S in a 1-bit data transfer time can be selected. For details, see section 12.7.4, Receive Data Sampling Timing and Reception Margin. Tables 12.5 and 12.7 show the maximum bit rates with external clock input.

RENESAS

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

Bit Rate (bit/s)	Operating Frequency Pφ (MHz)											
	8			9.8304			10			12		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	141	0.03	2	174	−0.26	2	177	−0.25	2	212	0.03
150	2	103	0.16	2	127	0.00	2	129	0.16	2	155	0.16
300	1	207	0.16	1	255	0.00	2	64	0.16	2	77	0.16
600	1	103	0.16	1	127	0.00	1	129	0.16	1	155	0.16
1200	0	207	0.16	0	255	0.00	1	64	0.16	1	77	0.16
2400	0	103	0.16	0	127	0.00	0	129	0.16	0	155	0.16
4800	0	51	0.16	0	63	0.00	0	64	0.16	0	77	0.16
9600	0	25	0.16	0	31	0.00	0	32	−1.36	0	38	0.16
19200	0	12	0.16	0	15	0.00	0	15	1.73	0	19	−2.34
31250	0	7	0.00	0	9	−1.70	0	9	0.00	0	11	0.00
38400	—	—	—	0	7	0.00	0	7	1.73	0	9	−2.34

Bit Rate (bit/s)	Operating Frequency Pφ (MHz)											
	12.288			14			14.7456			16		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	217	0.08	2	248	−0.17	3	64	0.70	3	70	0.03
150	2	159	0.00	2	181	0.16	2	191	0.00	2	207	0.16
300	2	79	0.00	2	90	0.16	2	95	0.00	2	103	0.16
600	1	159	0.00	1	181	0.16	1	191	0.00	1	207	0.16
1200	1	79	0.00	1	90	0.16	1	95	0.00	1	103	0.16
2400	0	159	0.00	0	181	0.16	0	191	0.00	0	207	0.16
4800	0	79	0.00	0	90	0.16	0	95	0.00	0	103	0.16
9600	0	39	0.00	0	45	−0.93	0	47	0.00	0	51	0.16
19200	0	19	0.00	0	22	−0.93	0	23	0.00	0	25	0.16
31250	0	11	2.40	0	13	0.00	0	14	−1.70	0	15	0.00
38400	0	9	0.00	—	—	—	0	11	0.00	0	12	0.16

RENESAS

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

Bit Rate (bit/s)	Operating Frequency Pφ (MHz)										
	17.2032			18			19.6608			20		
	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	75	0.48	3	79	−0.12	3	86	0.31	3	88	−0.25
150	2	223	0.00	2	233	0.16	2	255	0.00	3	64	0.16
300	2	111	0.00	2	116	0.16	2	127	0.00	2	129	0.16
600	1	223	0.00	1	233	0.16	1	255	0.00	2	64	0.16
1200	1	111	0.00	1	116	0.16	1	127	0.00	1	129	0.16
2400	0	223	0.00	0	233	0.16	0	255	0.00	1	64	0.16
4800	0	111	0.00	0	116	0.16	0	127	0.00	0	129	0.16
9600	0	55	0.00	0	58	−0.69	0	63	0.00	0	64	0.16
19200	0	27	0.00	0	28	1.02	0	31	0.00	0	32	−1.36
31250	0	16	1.20	0	17	0.00	0	19	−1.70	0	19	0.00
38400	0	13	0.00	0	14	−2.34	0	15	0.00	0	15	1.73

Table 12.4   Maximum Bit Rate for Each Operating Frequency (Asynchronous Mode)

Pφ (MHz)	Maximum Bit Rate (bit/s)	n	N	Pφ (MHz)	Maximum Bit Rate (bit/s)	n	N
8	250000	0	0	14.7456	460800	0	0
9.8304	307200	0	0	16	500000	0	0
10	312500	0	0	17.2032	537600	0	0
12	375000	0	0	18	562500	0	0
12.288	384000	0	0	19.6608	614400	0	0
14	437500	0	0	20	625000	0	0

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

Pφ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	Pφ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	2.0000	125000	14.7456	3.6864	230400
9.8304	2.4576	153600	16	4.0000	250000
10	2.5000	156250	17.2032	4.3008	268800
12	3.0000	187500	18	4.5000	281250
12.288	3.0720	192000	19.6608	4.9152	307200
14	3.5000	218750	20	5.0000	312500

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

Bit Rate (bit/s)	Operating Frequency Pφ (MHz)							
	8		10		16		20	
	n	N	n	N	n	N	n	N
110								
250	3	124	—	—	3	249		
500	2	249	—	—	3	124	—	—
1k	2	124	—	—	2	249	—	—
2.5k	1	199	1	249	2	99	2	124
5k	1	99	1	124	1	199	1	249
10k	0	199	0	249	1	99	1	124
25k	0	79	0	99	0	159	0	199
50k	0	39	0	49	0	79	0	99
100k	0	19	0	24	0	39	0	49
250k	0	7	0	9	0	15	0	19
500k	0	3	0	4	0	7	0	9
1M	0	1			0	3	0	4
2.5M			0	0*			0	1
5M							0	0*

Legend:

Blank   : Setting prohibited.

—   : Can be set, but there will be error.

*   : Continuous transmission or reception is not possible.

RENESAS

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

Pφ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	Pφ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	1.3333	1333333.3	16	2.6667	2666666.7
10	1.6667	1666666.7	18	3.0000	3000000.0
12	2.0000	2000000.0	20	3.3333	3333333.3
14	2.3333	2333333.3			

Table 12.8   BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, S = 372)

Bit Rate	Operating Frequency Pφ (MHz)											
	7.1424			10.00			10.7136			13.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	0	0.00	0	1	30	0	1	25	0	1	8.99

Bit Rate	Operating Frequency Pφ (MHz)											
	14.2848			16.00			18.00			20.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	1	0.00	0	1	12.01	0	2	15.99	0	2	6.60

Table 12.9   Maximum Bit Rate for Each Operating Frequency (Smart Card Interface Mode, S = 372)

Pφ (MHz)	Maximum Bit Rate (bit/s)	n	N	Pφ (MHz)	Maximum Bit Rate (bit/s)	n	N
7.1424	9600	0	0	14.2848	19200	0	0
10.00	13441	0	0	16.00	21505	0	0
10.7136	14400	0	0	18.00	24194	0	0
13.00	17473	0	0	20.00	26882	0	0

RENESAS

12.4   Operation in Asynchronous Mode

Figure 12.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transmit/receive data, a parity bit, and finally stop bits (high level). In asynchronous serial communication, the communication line is usually held in the mark state (high level). The SCI monitors the communication line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transmission and reception.
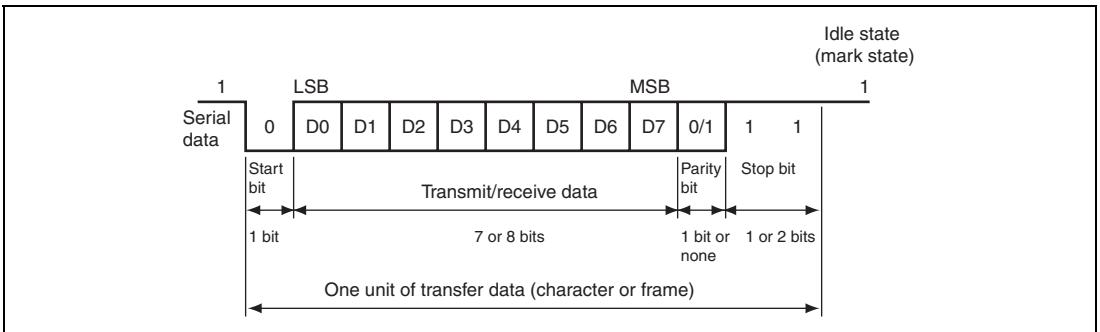
Figure 12.2   Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

12.4.1   Data Transfer Format

Table 12.10 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 12.5, Multiprocessor Communication Function.

Table 12.10 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

RENESAS

12.4.2   Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the bit rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Since receive data is sampled at the rising edge of the 8th pulse of the basic clock, data is latched at the middle of each bit, as shown in figure 12.3. Thus the reception margin in asynchronous mode is determined by formula (1) below.

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

M: Reception margin
N: Ratio of bit rate to clock (N = 16)
D: Duty cycle of clock (D = 0.5 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in formula (1), the reception margin is determined by the formula below.

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

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.
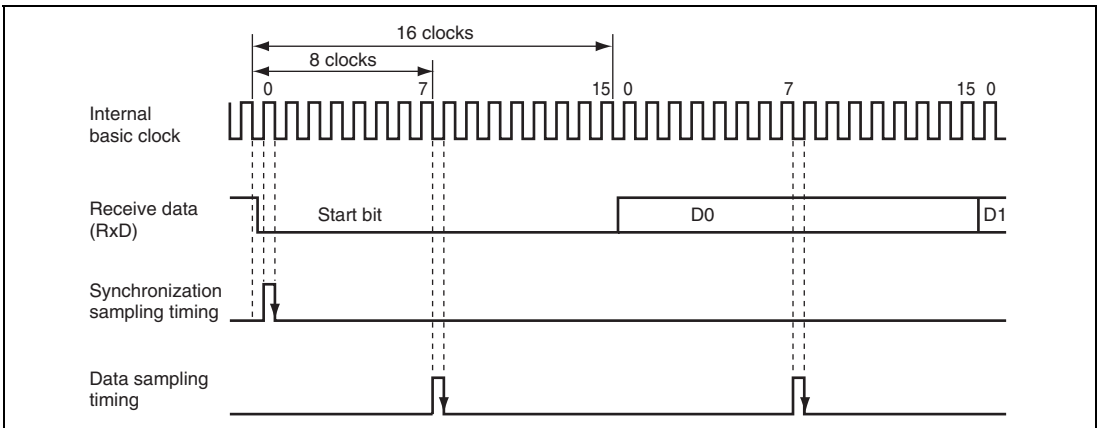
Figure 12.3   Receive Data Sampling Timing in Asynchronous Mode

12.4.3   Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input to 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 to the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 12.4.
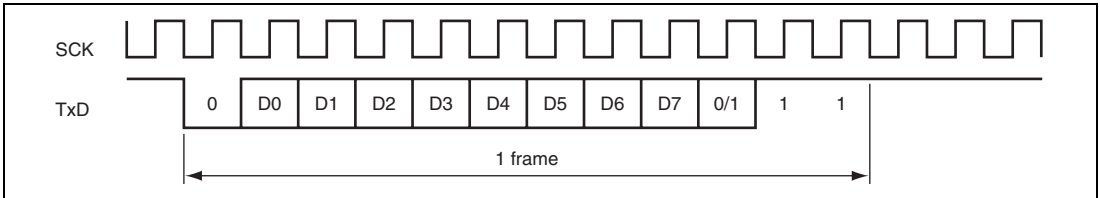
Figure 12.4   Phase Relation between Output Clock and Transmit Data (Asynchronous Mode)

12.4.4   SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 12.5. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization.
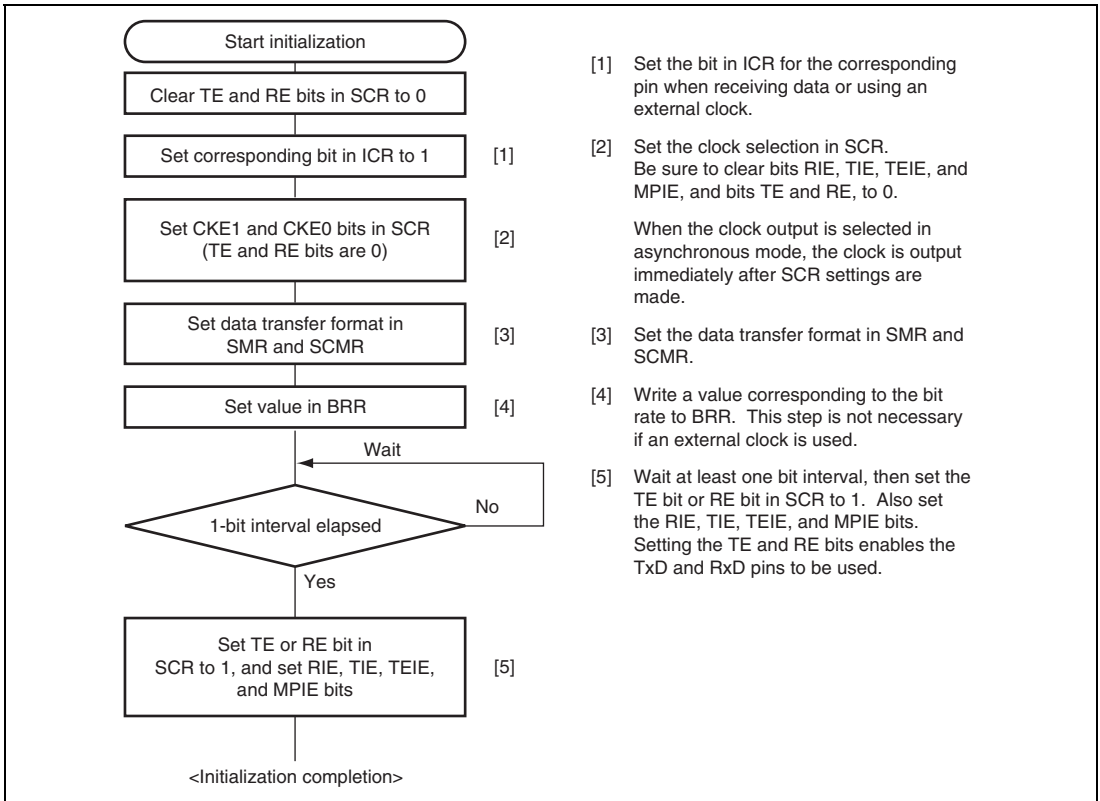
Figure 12.5   Sample SCI Initialization Flowchart

12.4.5   Serial Data Transmission (Asynchronous Mode)

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

1. The SCI monitors the TDRE flag in SSR, and if it is cleared to 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit or multiprocessor bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the next transmit data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 12.7 shows a sample flowchart for transmission in asynchronous mode.
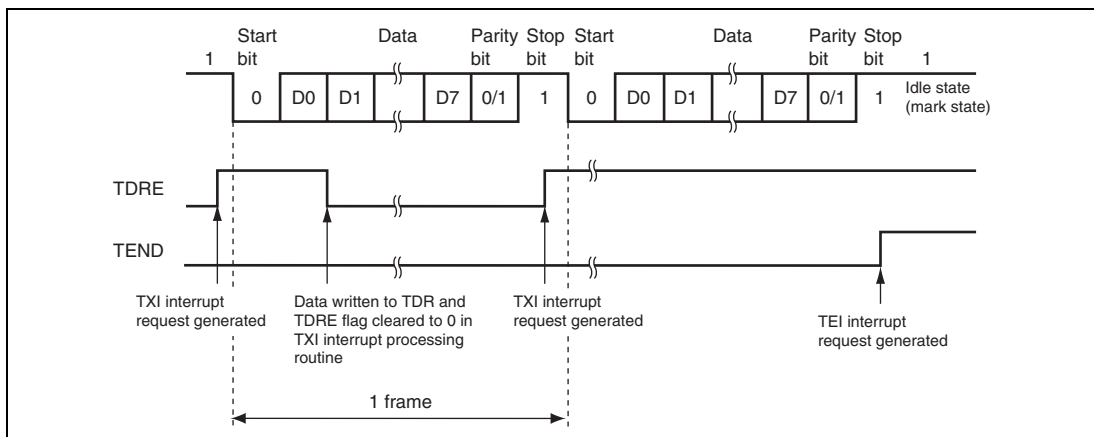
**Figure 12.6   Example of Operation for Transmission in Asynchronous Mode
(Example with 8-Bit Data, Parity, One Stop Bit)**

```
        ┌─────────────────────────┐
        │      Initialization     │  [1]
        └─────────────────────────┘
        (      Start transmission     )
                     │
        ┌─────────────────────────┐
  ┌────▶│   Read TDRE flag in SSR  │  [2]
  │     └─────────────────────────┘
  │              │
  │           ◇ TDRE = 1 ◇ ──No──┐
  │              │ Yes           │
  │     ┌─────────────────────────┐
  │     │  Write transmit data to TDR │
  │     │ and clear TDRE flag in SSR to 0 │
  │     └─────────────────────────┘
  │              │
  └──── ◇ All data transmitted? ◇ ──No
                 │ Yes
        ┌─────────────────────────┐
  ┌────▶│   Read TEND flag in SSR  │  [3]
  │     └─────────────────────────┘
  │              │
  └──── ◇ TEND = 1 ◇ ──No
                 │ Yes
        ◇ Break output ◇ ──No── [4]
                 │ Yes            │
        ┌─────────────────────────┐
        │   Clear DR to 0 and     │
        │     set DDR to 1        │
        └─────────────────────────┘
                 │                │
        ┌─────────────────────────┐
        │  Clear TE bit in SCR to 0 │
        └─────────────────────────┘
                 │
              <End>
```

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

[2]   SCI state 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 DMAC 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 12.7   Sample Serial Transmission Flowchart

RENESAS

12.4.6   Serial Data Reception (Asynchronous Mode)

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

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, stores receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.
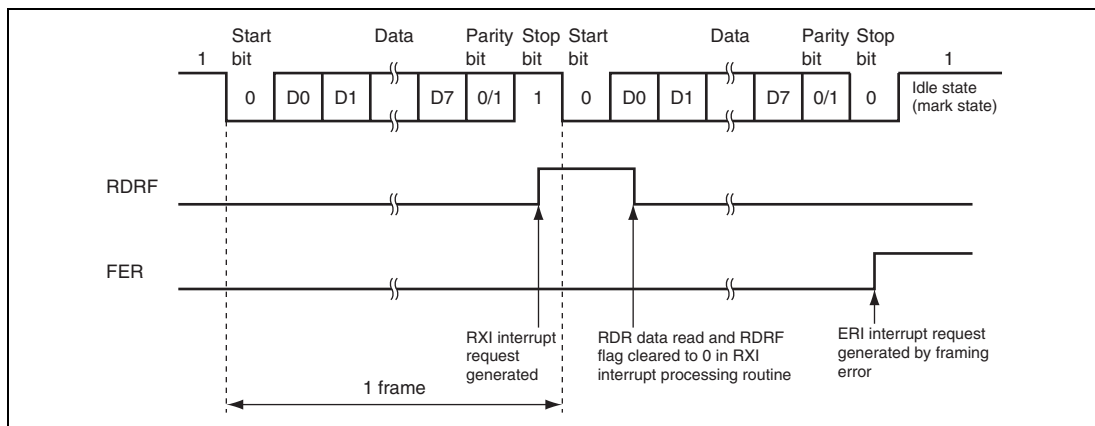
**Figure 12.8   Example of SCI Operation for Reception
(Example with 8-Bit Data, Parity, One Stop Bit)**

Table 12.11 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 12.9 shows a sample flowchart for serial data reception.

Table 12.11 SSR Status Flags and Receive Data Handling

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

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

RENESAS

Figure 12.9   Sample Serial Reception Flowchart (1)

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

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

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

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag and RDR, and clear the RDRF flag to 0. However, the RDRF flag is cleared automatically when the DMAC is initiated by an RXI interrupt and reads data from RDR.

[3]

Figure 12.9   Sample Serial Reception Flowchart (2)

12.5   Multiprocessor Communication Function

Use of the multiprocessor communication function enables data transfer to be performed among a number of processors sharing communication lines by means of asynchronous serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data. When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code. The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle for the specified receiving station. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle. If the multiprocessor bit is 1, the cycle is an ID transmission cycle, and if the multiprocessor bit is 0, the cycle is a data transmission cycle. Figure 12.10 shows an example of inter-processor communication using the multiprocessor format. The transmitting station first sends data which includes the ID code of the receiving station and a multiprocessor bit set to 1. It then transmits transmit data added with the multiprocessor bit cleared to 0. The receiving station skips data until data with the multiprocessor bit set to 1 is sent. When data with the multiprocessor bit set to 1 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 the multiprocessor bit set to 1 is again received.

The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set to 1, transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status flags, RDRF, FER, and ORER in SSR to 1 are prohibited until data with the multiprocessor bit set to 1 is received. On reception of a receive character with the multiprocessor bit set to 1, the MPBR bit in SSR is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt is generated.

When the multiprocessor format is selected, the parity bit setting is invalid. All other bit settings are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.
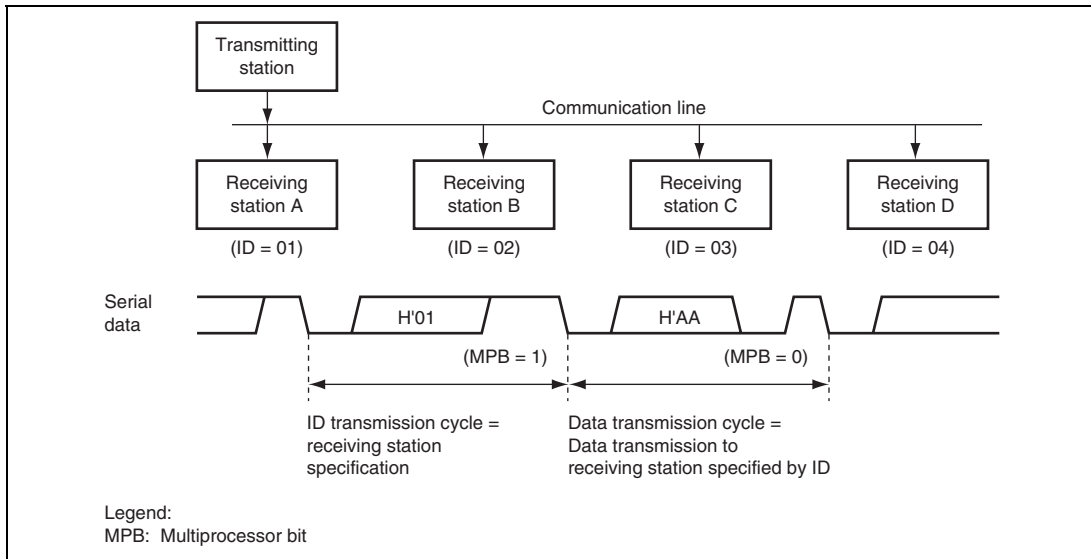
**Figure 12.10   Example of Communication Using Multiprocessor Format
(Transmission of Data H'AA to Receiving Station A)**

12.5.1   Multiprocessor Serial Data Transmission

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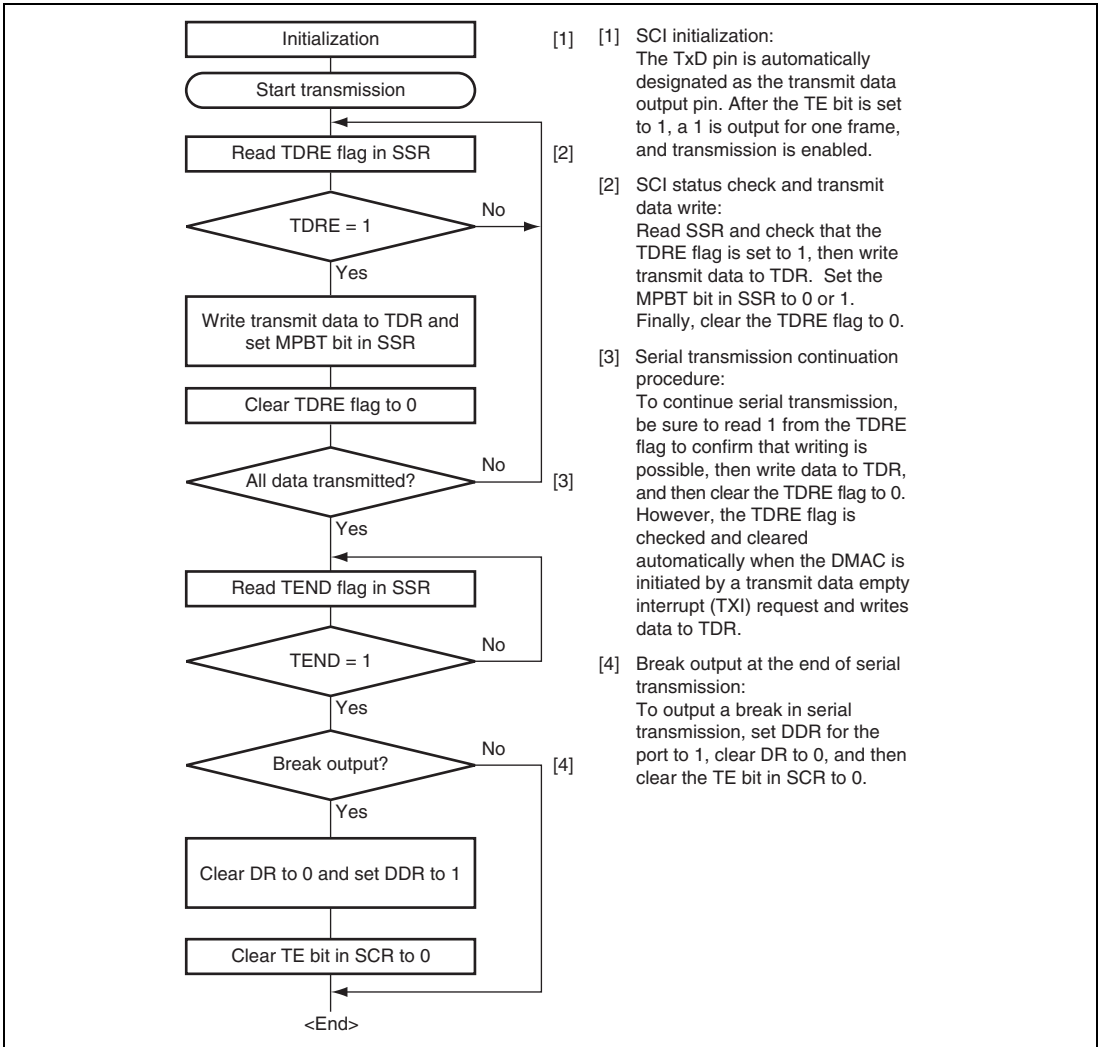
Figure 12.11   Sample Multiprocessor Serial Transmission Flowchart

12.5.2   Multiprocessor Serial Data Reception

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

(a) Data does not match station's ID

(b) Data matches station's ID
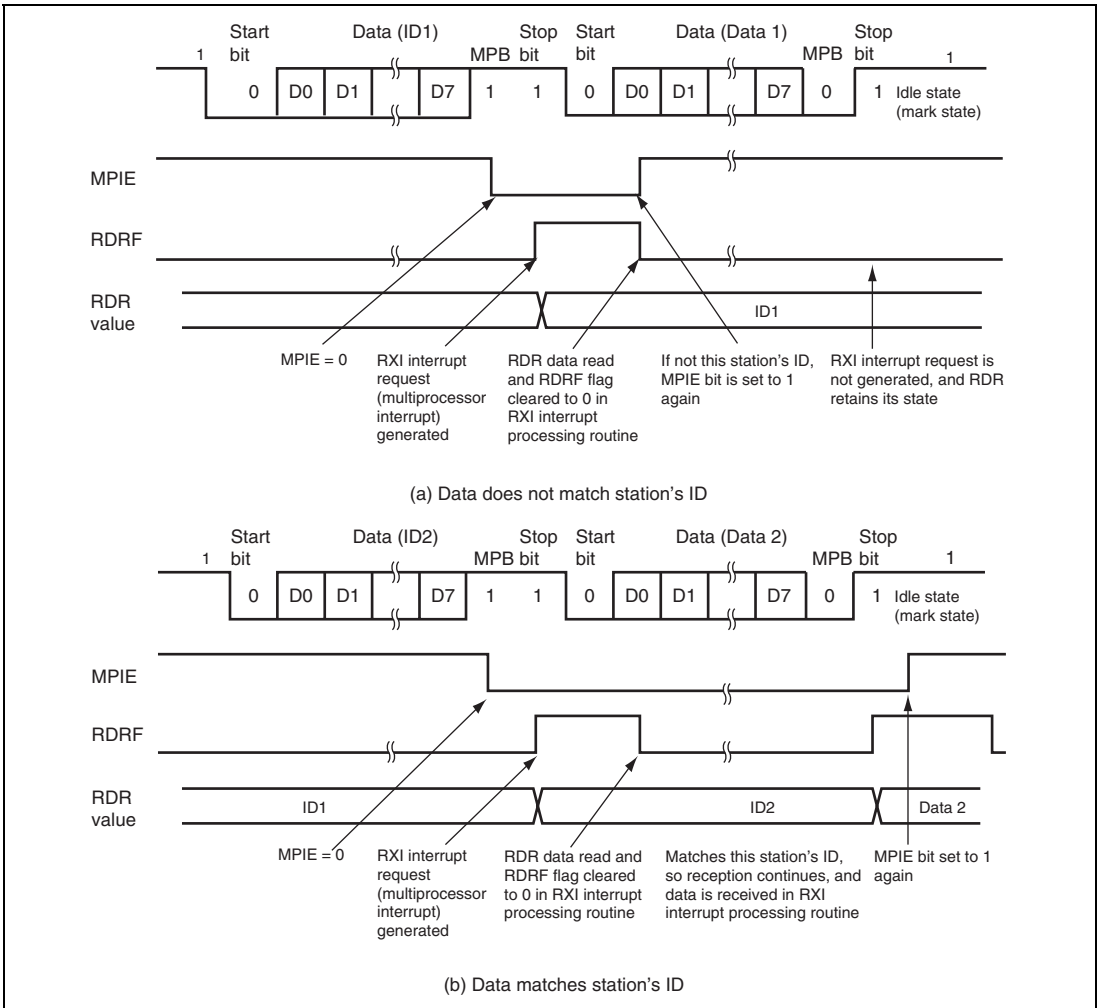
Figure 12.12   Example of SCI Operation for Reception
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)

Initialization [1]

Start reception

Set MPIE bit in SCR to 1 [2]

Read ORER and FER flags in SSR

FER ∨ ORER = 1   Yes / No

Read RDRF flag in SSR [3]

RDRF = 1   No / Yes

Read receive data in RDR

This station's ID?   No / Yes

Read ORER and FER flags in SSR

FER ∨ ORER = 1   Yes / No

Read RDRF flag in SSR [4]

RDRF = 1   No / Yes

Read receive data in RDR

All data received?   No / [5] Yes

Error processing

Clear RE bit in SCR to 0

(Continued on next page)

<End>

[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 state 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 state 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 both 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.

Figure 12.13   Sample Multiprocessor Serial Reception Flowchart (1)

Figure 12.13   Sample Multiprocessor Serial Reception Flowchart (2)

12.6   Operation in Clocked Synchronous Mode

Figure 12.14 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses. One character in transfer data consists of 8-bit data. In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the MSB output state. In clocked synchronous mode, no parity bit or multiprocessor bit is added. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that the next transmit data can be written during transmission or the previous receive data can be read during reception, enabling continuous data transfer.
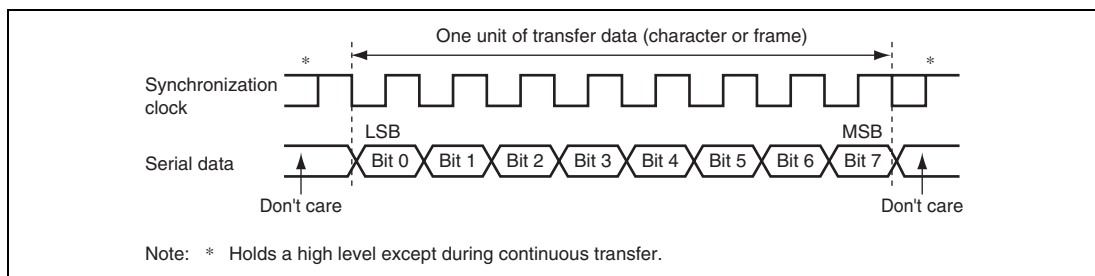
Figure 12.14   Data Format in Clocked Synchronous Communication (LSB-First)

12.6.1   Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCR. When the SCI is operated on an internal clock, the synchronization clock is output from the SCK pin. Eight synchronization clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. Note that in the case of reception only, the synchronization clock is output until an overrun error occurs or until the RE bit is cleared to 0.

12.6.2   SCI Initialization (Clocked Synchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 12.15. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. However, clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR.

The flowchart steps on the right:

[1]   Set the bit in ICR for the corresponding pin when receiving data or using an external clock.

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

[3]   Set the data transfer format in SMR and SCMR.

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

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

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

Figure 12.15   Sample SCI Initialization Flowchart

12.6.3　Serial Data Transmission (Clocked Synchronous Mode)

Figure 12.16 shows an example of the operation for transmission in clocked synchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. 8-bit data is sent from the TxD pin synchronized with the output clock when clock output mode has been specified and synchronized with the input clock when use of an external clock has been specified.
4. The SCI checks the TDRE flag at the timing for sending the last bit.
5. If the TDRE flag is cleared to 0, the next transmit data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TxD pin retains the output state of the last bit. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated. The SCK pin is fixed high.

Figure 12.17 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.
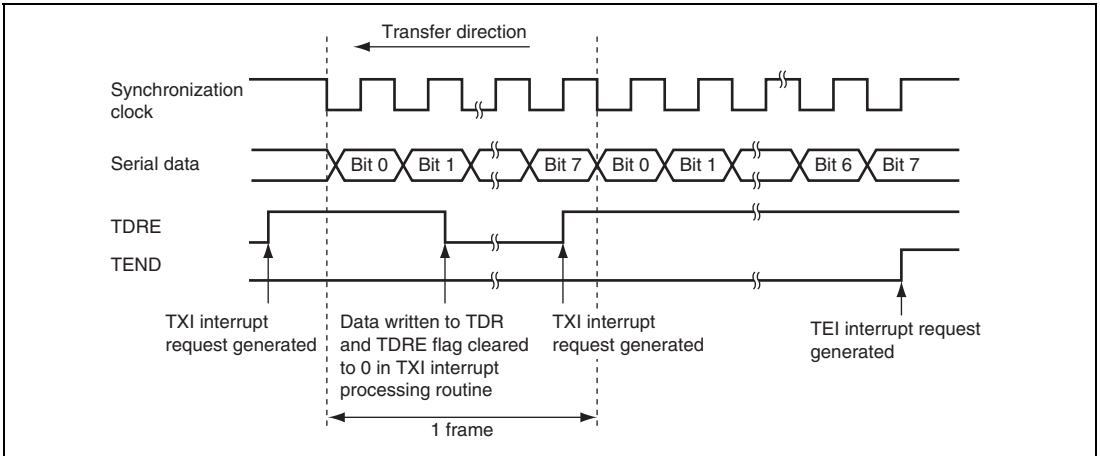
Figure 12.16   Example of Operation for Transmission in Clocked Synchronous Mode

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

[2] SCI state 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 DMAC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

Figure 12.17   Sample Serial Transmission Flowchart

12.6.4   Serial Data Reception (Clocked Synchronous Mode)

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

1. The SCI performs internal initialization in synchronization with a synchronization clock input or output, starts receiving data, and stores the receive data in RSR.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.
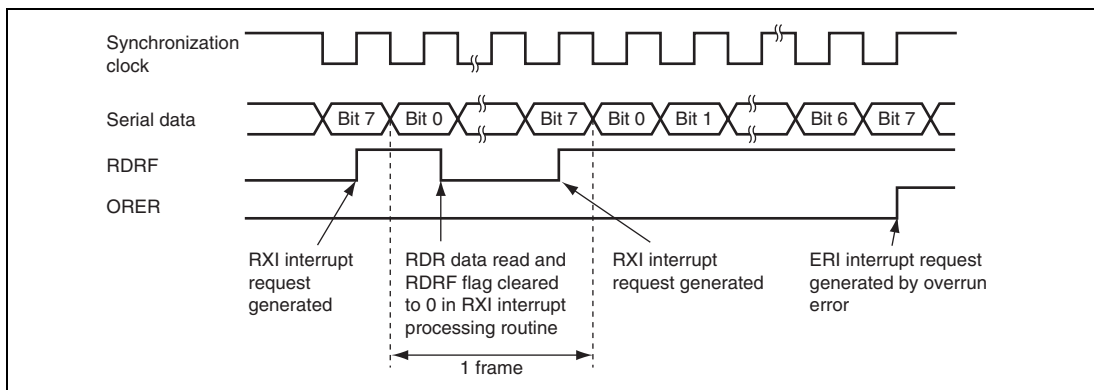
Figure 12.18   Example of Operation for Reception in Clocked Synchronous Mode

Transfer cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 12.19 shows a sample flowchart for serial data reception.
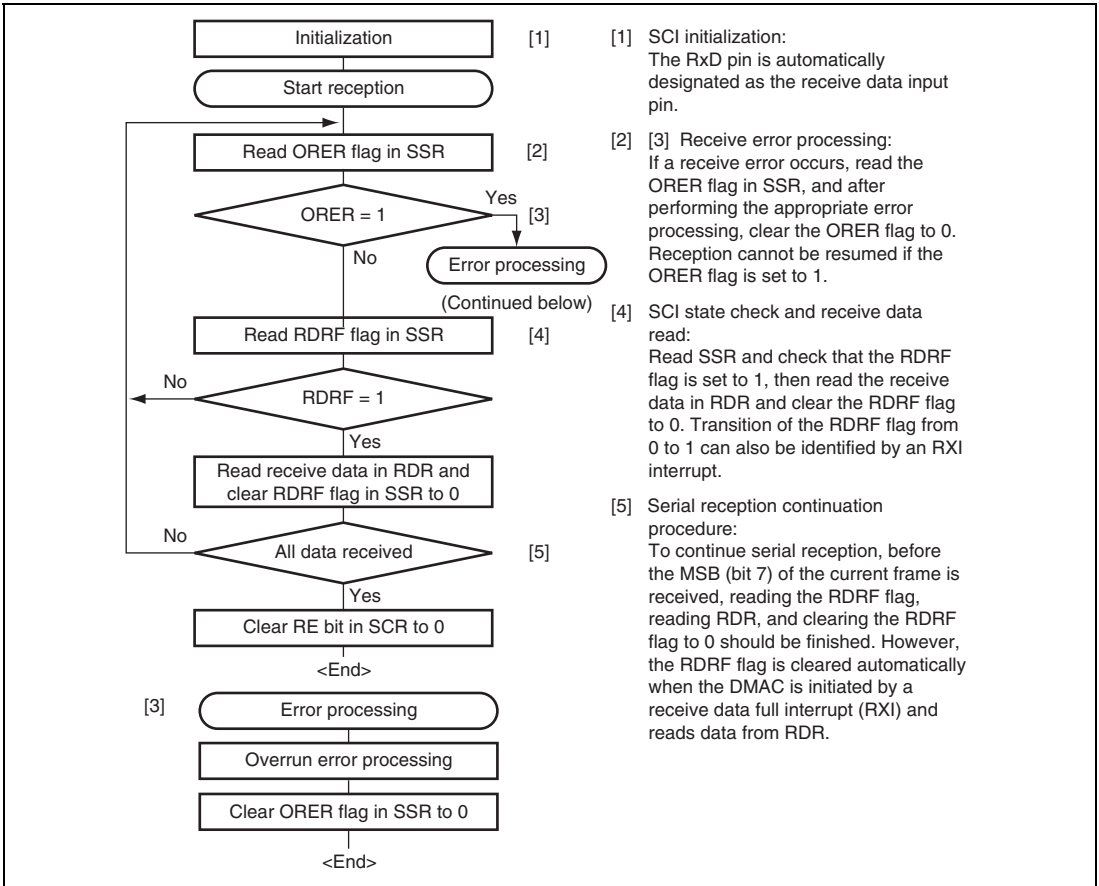
Figure 12.19   Sample Serial Reception Flowchart

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

Figure 12.20 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags are set to 1, clear the TE bit to 0. Then simultaneously set both the TE and RE bits to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI has finished reception, clear the RE bit to 0. Then after checking that the RDRF bit and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set both the TE and RE bits to 1 with a single instruction.
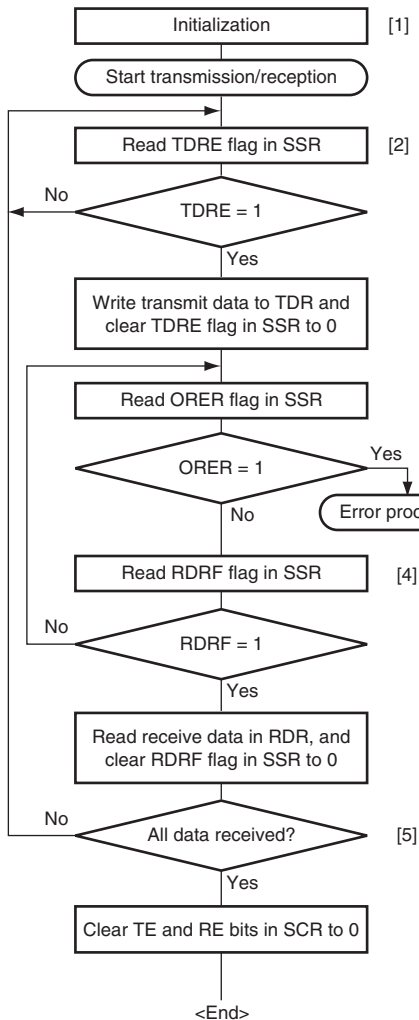
```
                Initialization           [1]

          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]
                       │ No          Error processing
            Read RDRF flag in SSR          [4]

No  ◄──────────    RDRF = 1
                       │ Yes
          Read receive data in RDR, and
            clear RDRF flag in SSR to 0

No  ◄──────────  All data received?       [5]
                       │ Yes
          Clear TE and RE bits in SCR to 0

                     <End>
```

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

[2] SCI state 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. Reception cannot be resumed if the ORER flag is set to 1.

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

[5] Serial transmission/reception continuation procedure:
     To continue serial transmission/ reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.
     However, the TDRE flag is checked and cleared automatically when the DMAC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR. Similarly, the RDRF flag is cleared automatically when the DMAC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

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

Figure 12.20   Sample Flowchart of Simultaneous Serial Transmission and Reception

12.7 Operation in Smart Card Interface Mode

The SCI supports the IC card (smart card) interface, conforming to ISO/IEC 7816-3 (Identification Card) standard, as an extended serial communication interface function. Smart card interface mode can be selected using the appropriate register.

12.7.1 Sample Connection

Figure 12.21 shows a sample connection between the smart card and this LSI. As in the figure, since this LSI communicates with the IC card using a single transmission line, interconnect the TxD and RxD pins and pull up the data transmission line to V_{cc} using a resistor. Setting the RE and TE bits to 1 with the IC card not connected enables closed transmission/reception allowing self diagnosis. To supply the IC card with the clock pulses generated by the SCI, input the SCK pin output to the CLK pin of the IC card. A reset signal can be supplied via the output port of this LSI.
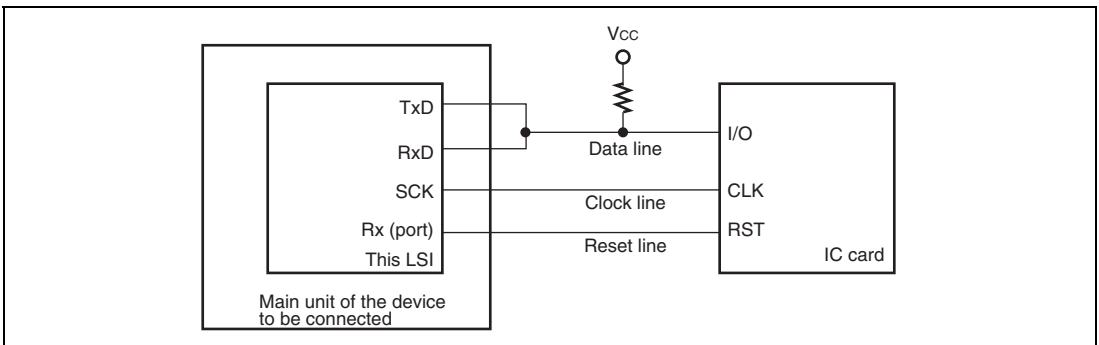
Figure 12.21   Pin Connection for Smart Card Interface

RENESAS

12.7.2   Data Format (Except in Block Transfer Mode)

Figure 12.22 shows the data transfer formats in smart card interface mode.

- One frame contains 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etu (elementary time unit: time required for transferring one bit) is secured as a guard time after the end of the parity bit before the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etu has passed from the start bit.
- If an error signal is sampled during transmission, the same data is automatically re-transmitted after at least 2 etu.
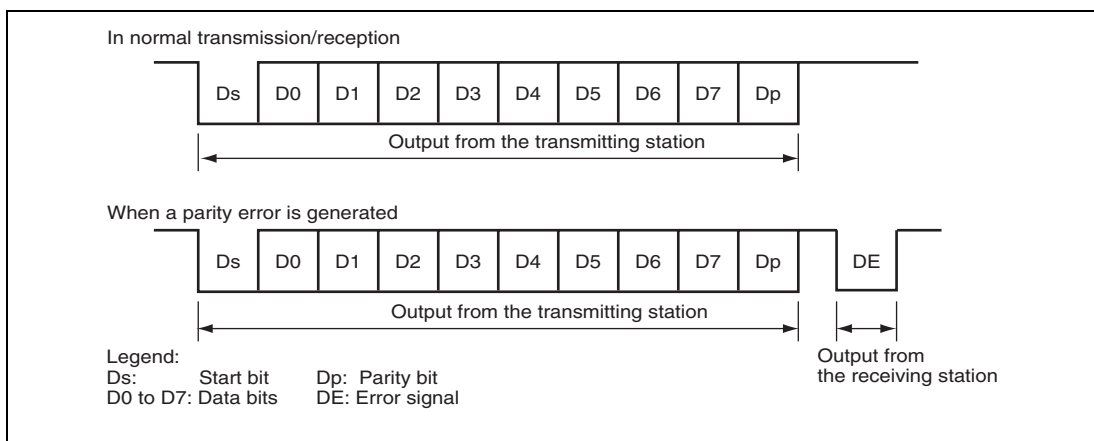
Figure 12.22   Data Formats in Normal Smart Card Interface Mode

For communication with the IC cards of the direct convention and inverse convention types, follow the procedure below.
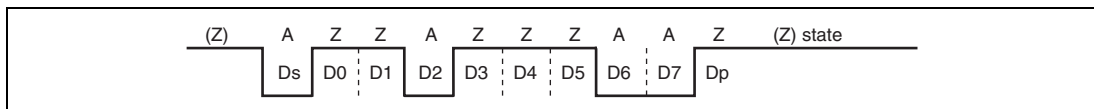
Figure 12.23   Direct Convention (SDIR = SINV = O/$\overline{E}$ = 0)

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 12.23. 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/$\overline{E}$ bit in SMR in order to use even parity, which is prescribed by the smart card standard.
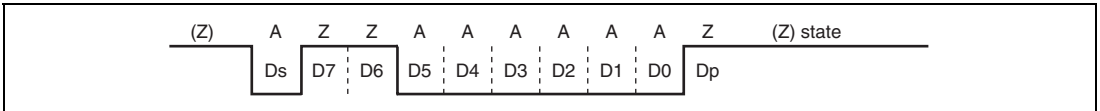
Figure 12.24   Inverse Convention (SDIR = SINV = O/$\overline{E}$ = 1)

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 12.24. Therefore, data in the start character in the figure is H'3F. When using the inverse convention type, write 1 to both the SDIR and SINV bits in SCMR. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to state Z. Since the SNIV bit of this LSI only inverts data bits D7 to D0, write 1 to the O/$\overline{E}$ bit in SMR to invert the parity bit in both transmission and reception.

12.7.3   Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following respects.

- Even if a parity error is detected during reception, no error signal is output. Since the PER bit in SSR is set by error detection, clear the PER bit before receiving the parity bit of the next frame.
- During transmission, at least 1 etu is secured as a guard time after the end of the parity bit before the start of the next frame.
- Since the same data is not re-transmitted during transmission, the TEND flag is set 11.5 etu after transmission start.
- Although the ERS flag in block transfer mode displays the error signal status as in normal smart card interface mode, the flag is always read as 0 because no error signal is transferred.

12.7.4   Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the on-chip baud rate generator can be used as a transfer clock in smart card interface mode. In this mode, the SCI can operate on a basic clock with a frequency of 32, 64, 372, or 256 times the bit rate according to the BCP1 and BCP0 bit settings (the frequency is always 16 times the bit rate in normal asynchronous mode). At reception, the falling edge of the start bit is sampled using the basic clock in order to perform internal synchronization. Receive data is sampled on the 16th, 32nd, 186th and 128th rising edges of the basic clock so that it can be latched at the middle of each bit as shown in figure 12.25. The reception margin here is determined by the following formula.

$$M = \left| \, (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}(1 + F) \, \right| \times 100\%$$

M: Reception margin (%)
N: Ratio of bit rate to clock (N = 32, 64, 372, 256)
D: Duty cycle of clock (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock frequency deviation

Assuming values of F = 0, D = 0.5, and N = 372 in the above formula, the reception margin is determined by the formula below.

$$M = \left(0.5 - \frac{1}{2 \times 372} \right) \times 100\% = 49.866\%$$
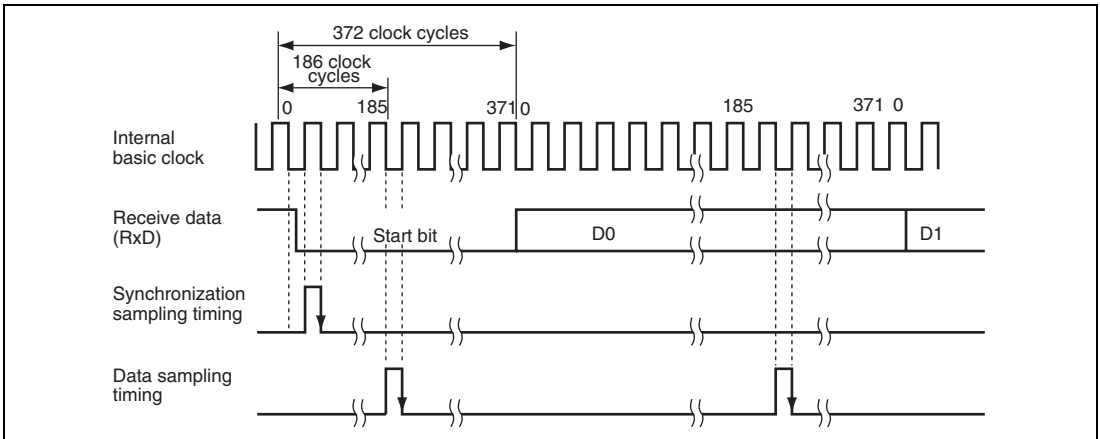
Figure 12.25   Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency Is 372 Times the Bit Rate)

12.7.5   Initialization

Before transmitting and receiving data, initialize the SCI using the following procedure. Initialization is also necessary before switching from transmission to reception and vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Set the ICR bit of the corresponding pin to 1.
3. Clear the error flags ERS, PER, and ORER in SSR to 0.
4. Set the GM, BLK, O/$\overline{\text{E}}$, BCP1, BCP0, CKS1, and CKS0 bits in SMR appropriately. Also set the PE bit to 1.
5. Set the SMIF, SDIR, and SINV bits in SCMR appropriately. When the DDR corresponding to the TxD pin is cleared to 0, the TxD and RxD pins are changed from port pins to SCI pins, placing the pins into high impedance state.
6. Set the value corresponding to the bit rate in BRR.
7. Set the CKE1 and CKE0 bits in SCR appropriately. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0 simultaneously.
   When the CKE0 bit is set to 1, the SCK pin is allowed to output clock pulses.
8. Set the TIE, RIE, TE, and RE bits in SCR appropriately after waiting for at least a 1-bit interval. Setting the TE and RE bits to 1 simultaneously is prohibited except for self diagnosis.

To switch from reception to transmission, first verify that reception has completed, then initialize the SCI. At the end of initialization, RE and TE should be set to 0 and 1, respectively. Reception completion can be verified by reading the RDRF, PER, or ORER flag. To switch from transmission to reception, first verify that transmission has completed, then initialize the SCI. At the end of initialization, TE and RE should be set to 0 and 1, respectively. Transmission completion can be verified by reading the TEND flag.

RENESAS

12.7.6   Data Transmission (Except in Block Transfer Mode)

Data transmission in smart card interface mode (except in block transfer mode) is different from that in normal serial communication interface mode in that an error signal is sampled and data can be re-transmitted. Figure 12.26 shows the data re-transfer operation during transmission.

1. If an error signal from the receiving end is sampled after one frame of data has been transmitted, the ERS bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the ERS bit to 0 before the next parity bit is sampled.
2. For the frame in which an error signal is received, the TEND bit in SSR is not set to 1. Data is re-transferred from TDR to TSR allowing automatic data retransmission.
3. If no error signal is returned from the receiving end, the ERS bit in SSR is not set to 1.
4. In this case, one frame of data is determined to have been transmitted including re-transfer, and the TEND bit in SSR is set to 1. Here, a TXI interrupt request is generated if the TIE bit in SCR is set to 1. Writing transmit data to TDR starts transmission of the next data.

Figure 12.28 shows a sample flowchart for transmission. All the processing steps are automatically performed using a TXI interrupt request to activate the DMAC. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request if the TIE bit in SCR has been set to 1. This activates the DMAC by a TXI request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DMAC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DMAC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, TEND remains as 0, thus not activating the DMAC. Therefore, the SCI and DMAC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DMAC, be sure to set and enable the DMAC prior to making SCI settings. For DMAC settings, see section 7, DMA Controller (DMAC).
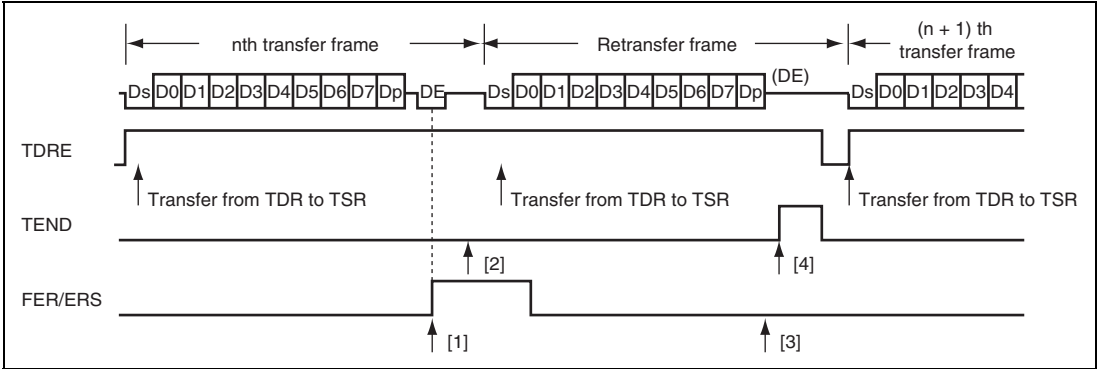
Figure 12.26   Data Re-Transfer Operation in SCI Transmission Mode

Note that the TEND flag is set in different timings depending on the GM bit setting in SMR. Figure 12.27 shows the TEND flag set timing.
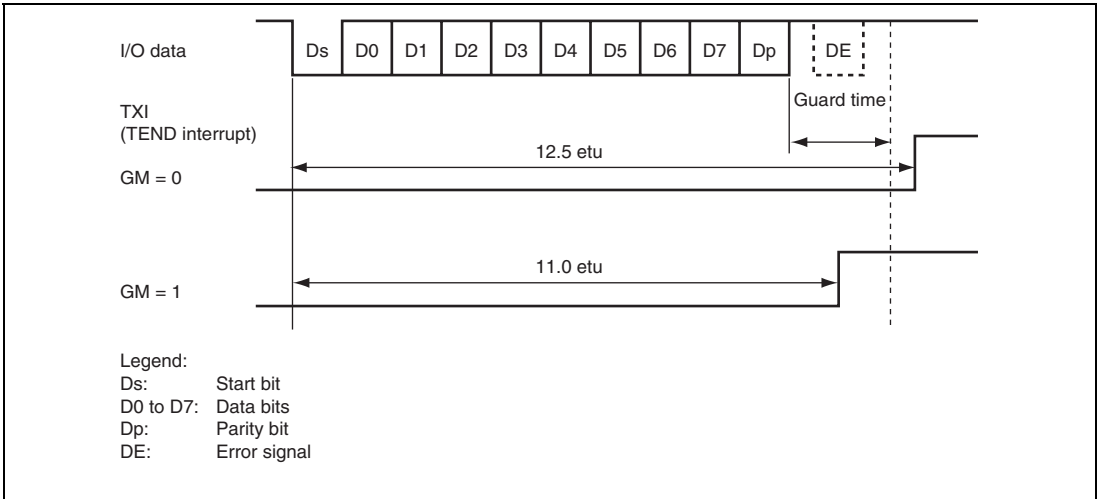
Figure 12.27   TEND Flag Set Timing during Transmission
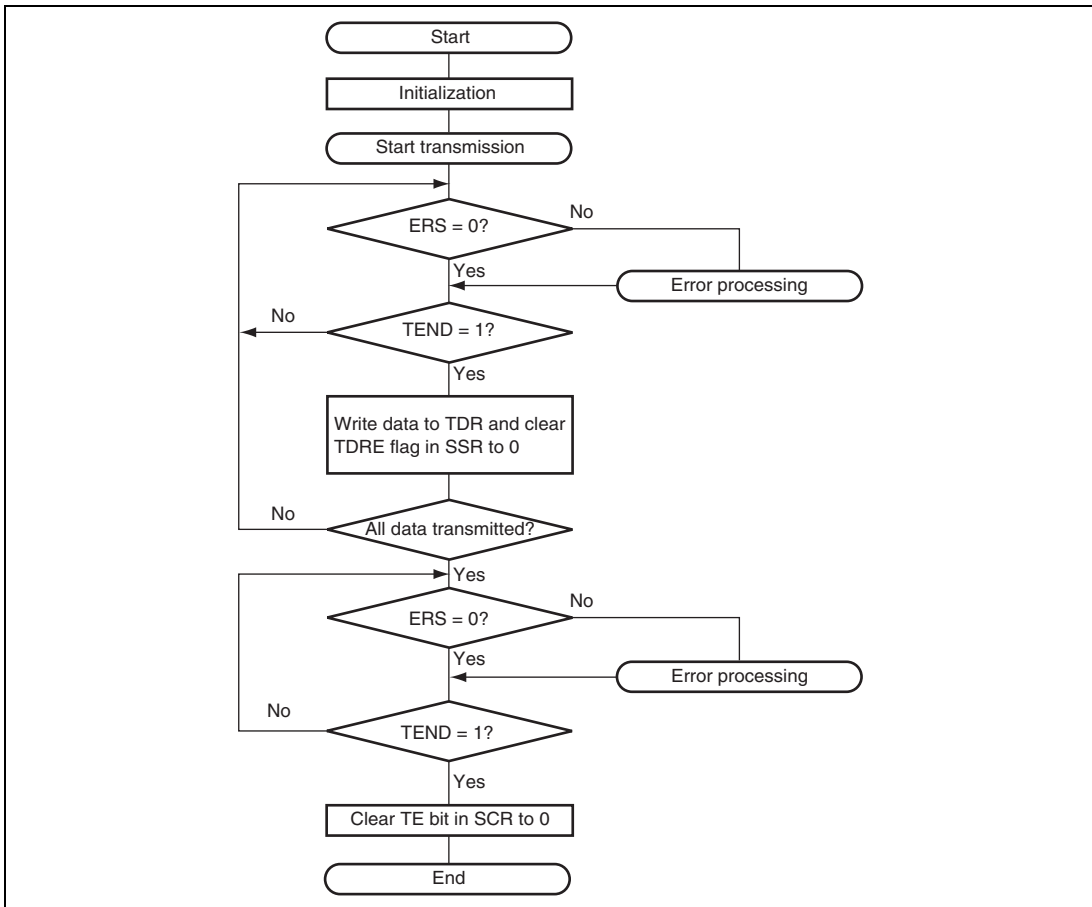
Figure 12.28   Sample Transmission Flowchart

RENESAS

12.7.7   Serial Data Reception (Except in Block Transfer Mode)

Data reception in smart card interface mode is similar to that in normal serial communication interface mode. Figure 12.29 shows the data re-transfer operation during reception.

1. If a parity error is detected in receive data, the PER bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the PER bit to 0 before the next parity bit is sampled.
2. For the frame in which a parity error is detected, the RDRF bit in SSR is not set to 1.
3. If no parity error is detected, the PER bit in SSR is not set to 1.
4. In this case, data is determined to have been received successfully, and the RDRF bit in SSR is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set to 1.

Figure 12.30 shows a sample flowchart for reception. All the processing steps are automatically performed using an RXI interrupt request to activate the DMAC. In reception, setting the RIE bit to 1 allows an RXI interrupt request to be generated when the RDRF flag is set to 1. This activates the DMAC by an RXI request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DMAC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC. If an error occurs during reception, i.e., either the ORER or PER flag is set to 1, a transmit/receive error interrupt (ERI) request is generated and the error flag must be cleared. If an error occurs, the DMAC is not activated and receive data is skipped, therefore, the number of bytes of receive data specified in the DMAC is transferred. Even if a parity error occurs and the PER bit is set to 1 in reception, receive data is transferred to RDR, thus allowing the data to be read.

Note: For operations in block transfer mode, see section 12.4, Operation in Asynchronous Mode.
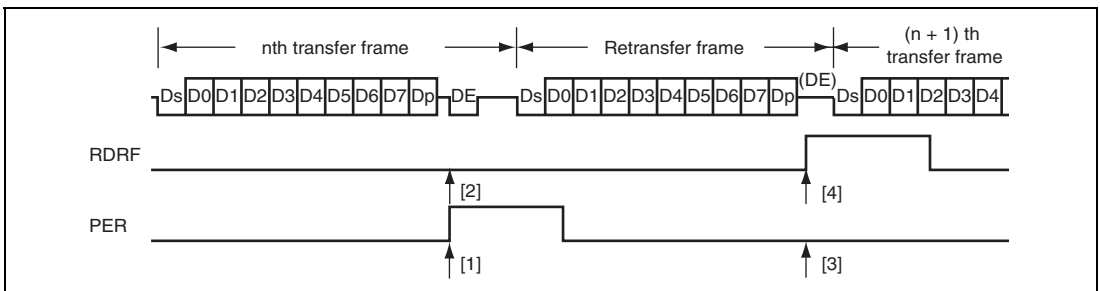
Figure 12.29   Data Re-Transfer Operation in SCI Reception Mode

RENESAS
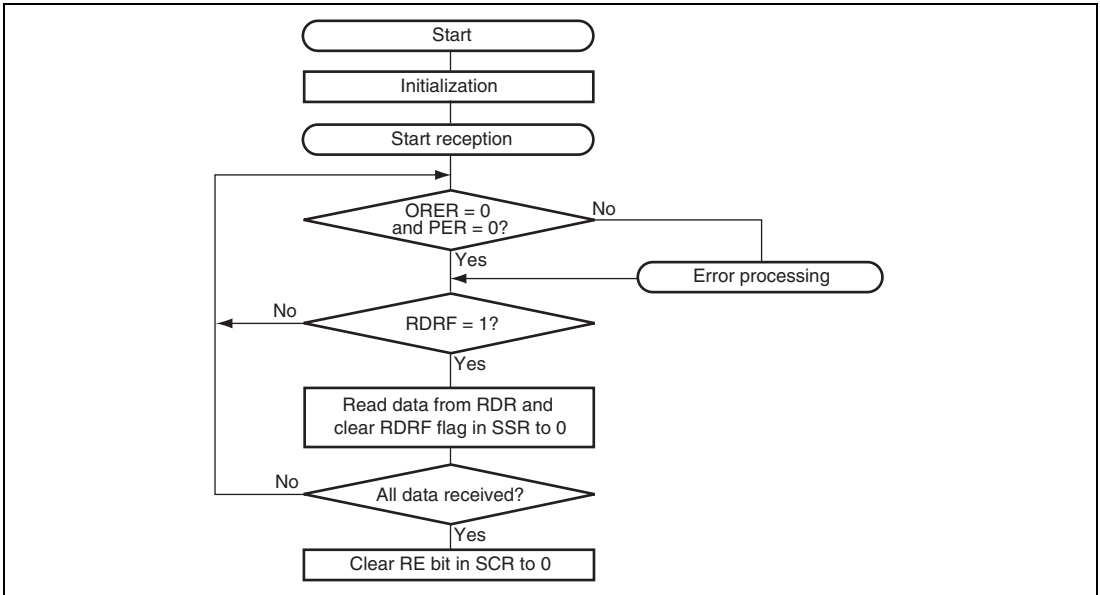
Figure 12.30   Sample Reception Flowchart

12.7.8   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 12.31 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 0.
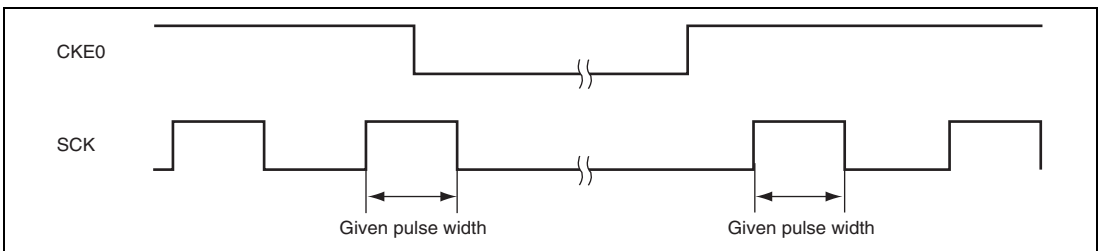
Figure 12.31   Clock Output Fixing Timing

At power-on and transitions to/from software standby mode, use the following procedure to secure the appropriate clock duty cycle.

- At power-on

  To secure the appropriate clock duty cycle simultaneously with power-on, use the following procedure.

  1. Initially, port input is enabled in the high-impedance state. To fix the potential level, use a pull-up or pull-down resistor.
  2. Fix the SCK pin to the specified output using the CKE1 bit in SCR.
  3. Set SMR and SCMR to enable smart card interface mode.

     Set the CKE0 bit in SCR to 1 to start clock output.

- At mode switching

  — At transition from smart card interface mode to software standby mode

  1. Set the data register (DR) and data direction register (DDR) corresponding to the SCK pin to the values for the output fixed state in software standby mode.
  2. Write 0 to the TE and RE bits in SCR to stop transmission/reception. Simultaneously, set the CKE1 bit to the value for the output fixed state in software standby mode.
  3. Write 0 to the CKE0 bit in SCR to stop the clock.
  4. Wait for one cycle of the serial clock. In the mean time, the clock output is fixed to the specified level with the duty cycle retained.
  5. Make the transition to software standby mode.

  — At transition from smart card interface mode to software standby mode

  1. Clear software standby mode.
  2. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty cycle is then generated.
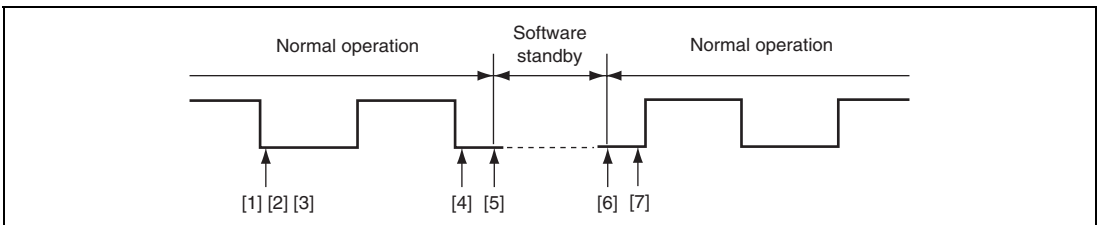
Figure 12.32   Clock Stop and Restart Procedure

12.8   Interrupt Sources

12.8.1   Interrupts in Normal Serial Communication Interface Mode

Table 12.12 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt request can activate the DMAC to allow data transfer. The TDRE flag is automatically cleared to 0 at data transfer by the DMAC.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DMAC to allow data transfer. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared to 0 simultaneously by the TXI interrupt processing routine, the SCI cannot branch to the TEI interrupt processing routine later.

Table 12.12 SCI Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DMAC Activation	Priority
ERI	Receive error	ORER, FER, or PER	Not possible	High
RXI	Receive data full	RDRF	Possible	↑
TXI	Transmit data empty	TDRE	Possible	
TEI	Transmit end	TEND	Not possible	Low

12.8.2   Interrupts in Smart Card Interface Mode

Table 12.13 shows the interrupt sources in smart card interface mode. A transmit end (TEI) interrupt request cannot be used in this mode.

Table 12.13 SCI Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DMAC Activation	Priority
ERI	Receive error or error signal detection	ORER, PER, or ERS	Not possible	High
RXI	Receive data full	RDRF	Possible	
TXI	Transmit data empty	TDRE	Possible	Low

Data transmission/reception using the DMAC is also possible in smart card interface mode, similar to in the normal SCI mode. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt. This activates the DMAC by a TXI request thus allowing transfer of transmit data if the TXI request is specified as a source of DMAC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DMAC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DMAC. Therefore, the SCI and DMAC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag in SSR, which is set at error occurrence, is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit in SCR to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DMAC, be sure to set and enable the DMAC prior to making SCI settings. For DMAC settings, see section 7, DMA Controller (DMAC).

In reception, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. This activates the DMAC by an RXI request thus allowing transfer of receive data if the RXI request is specified as a source of DMAC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC. If an error occurs, the RDRF flag is not set but the error flag is set. Therefore, the DMAC is not activated and an ERI interrupt request is issued to the CPU instead; the error flag must be cleared.

RENESAS

12.9   Usage Notes

12.9.1   Module Stop Mode Setting

Operation of the SCI can be disabled or enabled using the module stop control register. The initial setting is for operation of the SCI to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 24, Power-Down Modes.

12.9.2   Break Detection and Processing

When framing error detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set, and the PER flag may also be set. Note that, since the SCI continues the receive operation even after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

12.9.3   Mark State and Break Detection

When the TE bit is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line in mark state (the state of 1) until TE is set to 1, set both DDR and DR to 1. Since the TE bit is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set DDR to 1 and DR to 0, and then clear the TE bit to 0. When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

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

Transmission cannot be started when a receive error flag (ORER, FER, or RER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that the receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

12.9.5   Relation between Writing to TDR and TDRE Flag

The TDRE flag in SSR is a status flag which indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR irrespective of the TDRE flag status. However, if new data is written to TDR when the TDRE flag is 0, that is, when the previous data has not been transferred to TSR yet, the previous data in TDR is lost. Be sure to write transmit data to TDR after verifying that the TDRE flag is set to 1.

12.9.6   Writing to Registers during SCI Transmit or Receive

Do not write to SCR or SCMR during an SCI transmit or receive operation. The transmit or receive operation may not complete properly if SCR or SCMR is written to while it is in progress.

12.9.7   Restrictions on Using DMAC

- When the external clock source is used as a synchronization clock, update TDR by the DMAC 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 12.33).
- When using the DMAC to read RDR, be sure to set the receive end interrupt (RXI) as the DMAC activation source.

Note: When external clock is supplied, t must be more than four clock cycles.
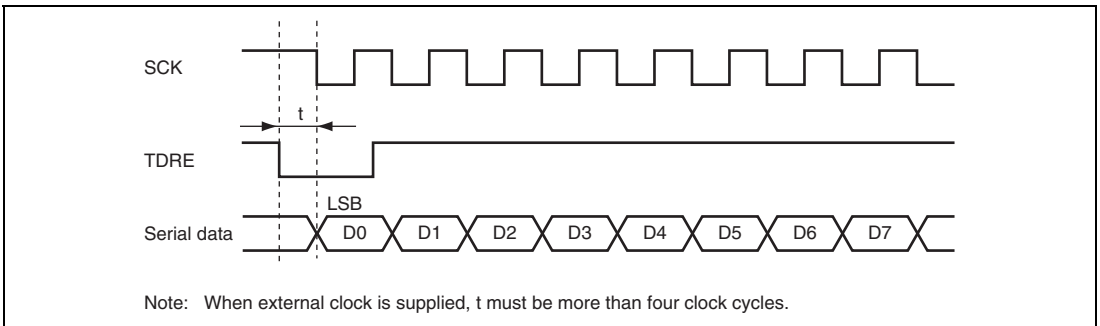
Figure 12.33   Sample Transmission Using DMAC in Clocked Synchronous Mode

12.9.8   SCI Operations during Mode Transitions

(1)   Transmission

Before making the transition to module stop mode or software standby mode, stop the transmit operations (TE = TIE = TEIE = 0). TSR, TDR, and SSR are reset. The states of the output pins during module stop mode or software standby mode depend on the port settings, and the pins output a high-level signal after mode cancellation. If the transition is made during data transmission, the data being transmitted will be undefined.

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

Figure 12.34 shows a sample flowchart for mode transition during transmission. Figures 12.35 and 12.36 show the port pin states during mode transition.

(2)   Reception

Before making the transition to module stop mode or software standby mode, stop the receive operations (RE = 0). RSR, RDR, and SSR are reset. If transition is made during data reception, the data being received will be invalid.

To receive data in the same reception mode after mode cancellation, set the RE bit to 1, and then start reception. To receive data in a different reception mode, initialize the SCI first.

Figure 12.37 shows a sample flowchart for mode transition during reception.
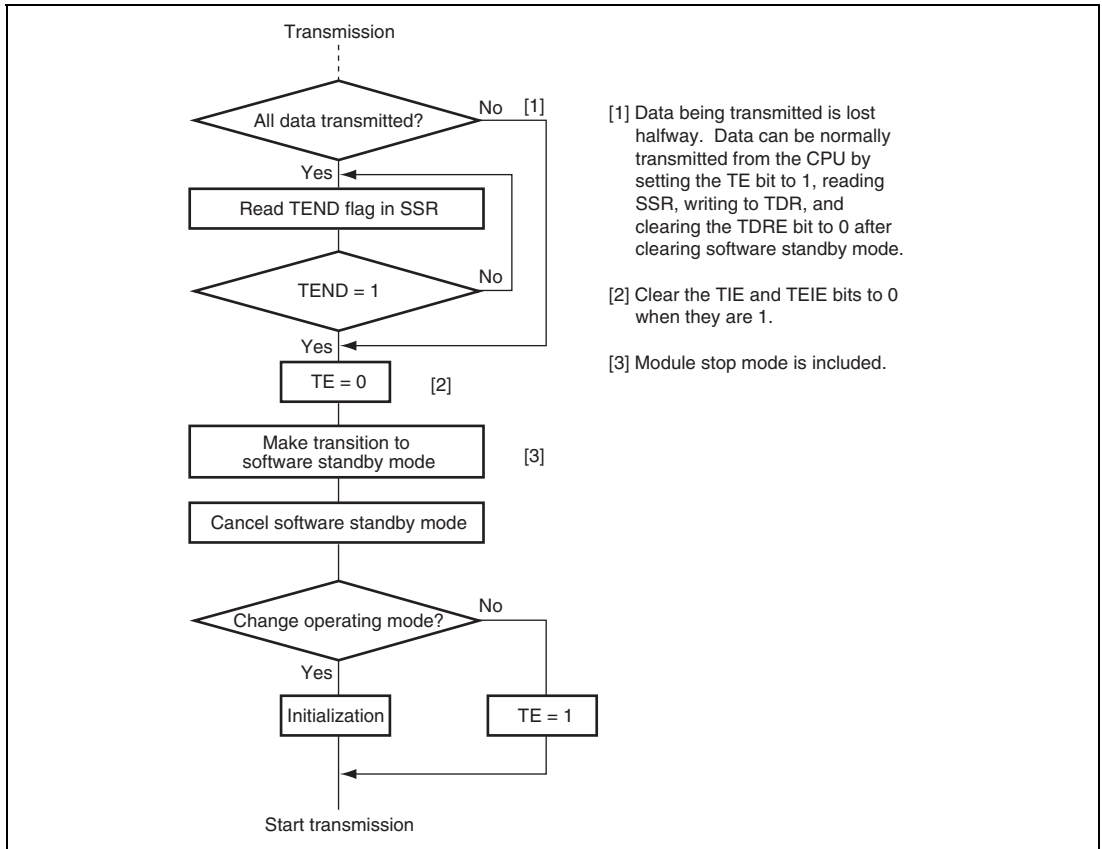
Figure 12.34   Sample Flowchart for Mode Transition during Transmission
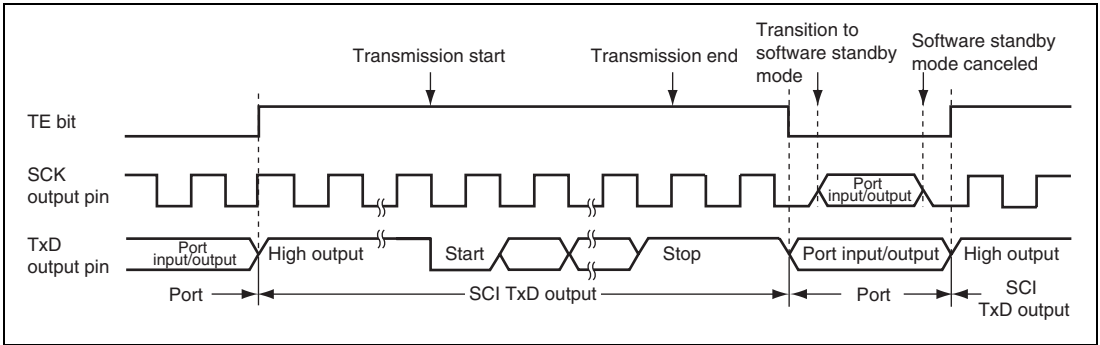
**Figure 12.35   Port Pin States during Mode Transition
(Internal Clock, Asynchronous Transmission)**
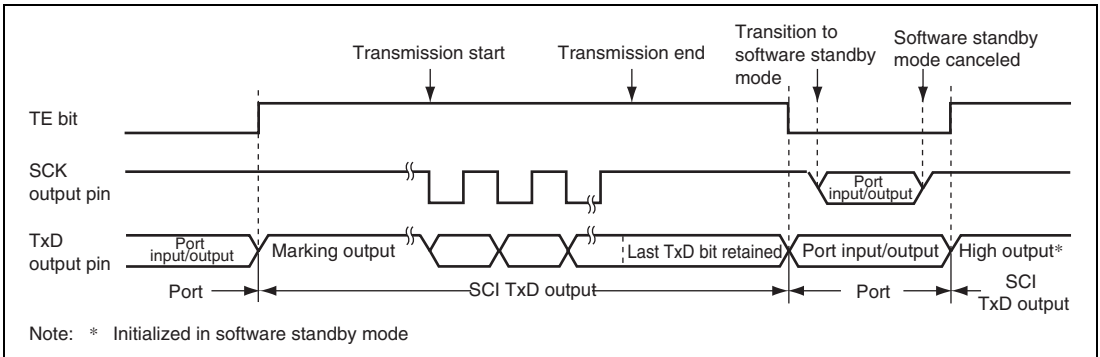
**Figure 12.36   Port Pin States during Mode Transition
(Internal Clock, Clocked Synchronous Transmission)**
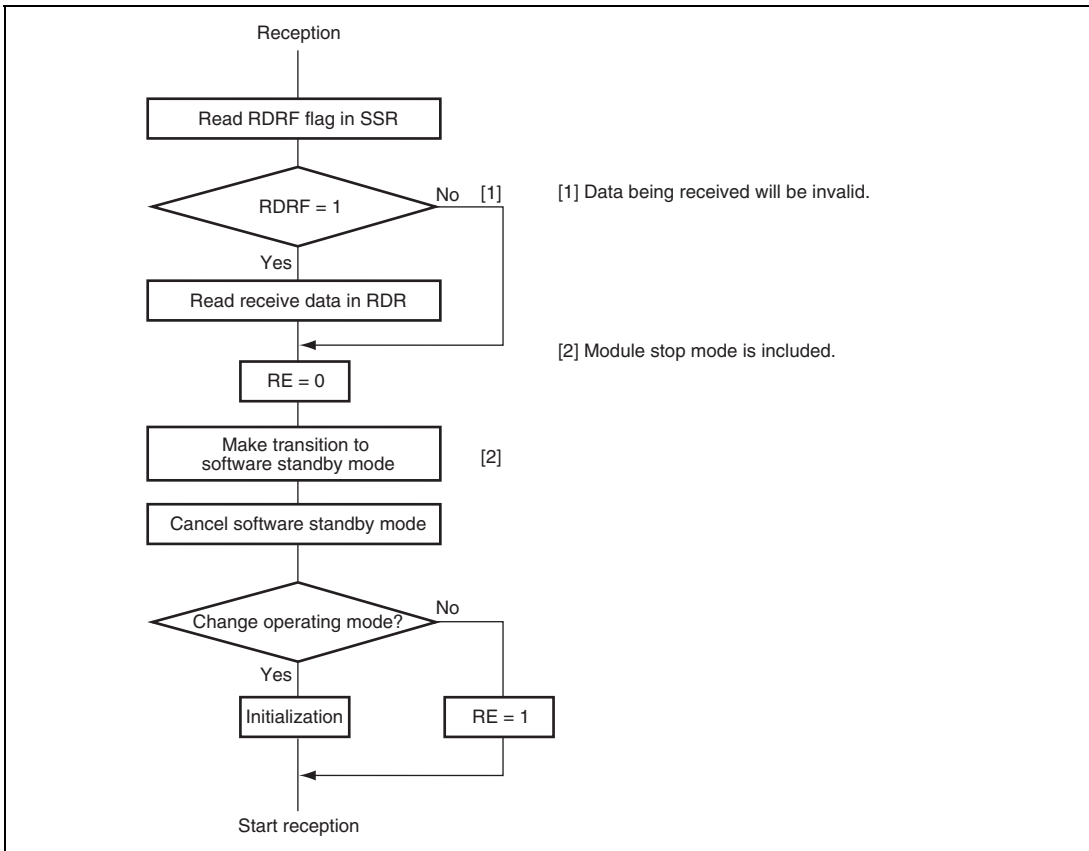
Figure 12.37   Sample Flowchart for Mode Transition during Reception

Section 13   I²C Bus Interface 2 (IIC2)

This LSI has a two-channel I²C bus interface.

The I²C bus interface conforms to and provides a subset of the Philips I²C bus (inter-IC bus) interface functions. The register configuration that controls the I²C bus differs partly from the Philips configuration, however.

Figure 13.1 shows the block diagram of the I²C bus interface 2.

Figure 13.2 shows an example of I/O pin connections to external circuits.

13.1   Features

- Continuous transmission/reception

  Since the shift register, transmit data register, and receive data register are independent from each other, the continuous transmission/reception can be performed.
- Start and stop conditions generated automatically in master mode
- Selection of acknowledge output levels when receiving
- Automatic loading of acknowledge bit when transmitting
- Bit synchronization/wait function

  In master mode, the state of SCL is monitored per bit, and the timing is synchronized automatically. If transmission or reception is not yet possible, drive the SCL signal low until preparations are completed
- Six interrupt sources

  Transmit-data-empty (including slave-address match), transmit-end, receive-data-full (including slave-address match), arbitration lost, NACK detection, and stop condition detection
- Direct bus drive

  Two pins, the SCL and SDA pins function as NMOS open-drain outputs.

Legend:
ICCRA: I²C bus control register A
ICCRB: I²C bus control register B
ICMR: I²C mode register
ICSR: I²C status register
ICIER: I²C interrupt enable register
ICDRT: I²C transmit data register
ICDRR: I²C receive data register
ICDRS: I²C bus shift register
SAR: Slave address register
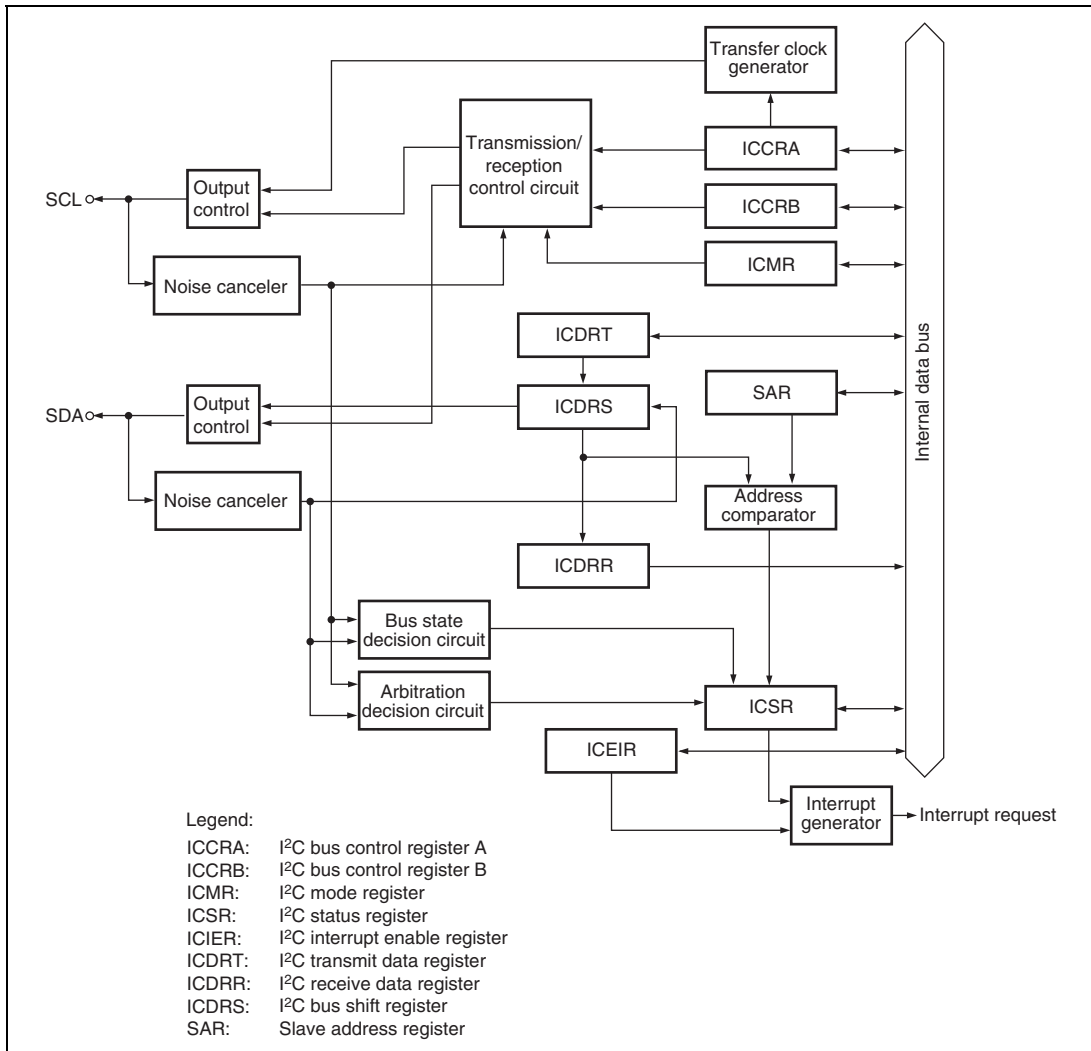
Figure 13.1   Block Diagram of I²C Bus Interface 2

RENESAS
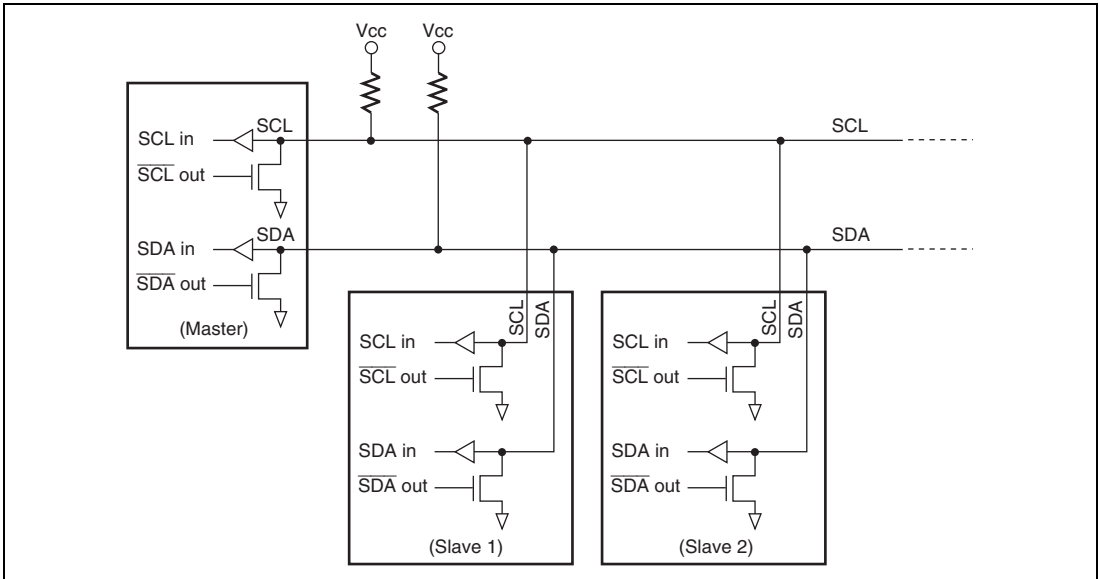
Figure 13.2   Connections to the External Circuit by the I/O Pins

13.2   Input/Output Pins

Table 13.1 shows the pin configuration of the I²C bus interface 2.

Table 13.1   Pin configuration of the I²C bus interface 2

Channel	Abbreviation	I/O	Function
0	SCL0	I/O	Channel 0 serial clock I/O pin
	SDA0	I/O	Channel 0 serial data I/O pin
1	SCL1	I/O	Channel 1 serial clock I/O pin
	SDA1	I/O	Channel 1 serial data I/O pin

Note: The pin symbols are represented as SCL and SDA; channel numbers are omitted in this manual.

13.3   Register Descriptions

The I²C bus interface 2 has the following registers.

Channel 0:

- I²C bus control register A_0 (ICCRA_0)
- I²C bus control register B_0 (ICCRB_0)
- I²C bus mode register_0 (ICMR_0)
- I²C bus interrupt enable register_0 (ICIER_0)
- I²C bus status register_0 (ICSR_0)
- Slave address register_0 (SAR_0)
- I²C bus transmit data register_0 (ICDRT_0)
- I²C bus receive data register_0 (ICDRR_0)
- I²C bus shift register_0 (ICDRS_0)

Channel 1:

- I²C bus control register A_1 (ICCRA_1)
- I²C bus control register B_1 (ICCRB_1)
- I²C bus mode register_1 (ICMR_1)
- I²C bus interrupt enable register_1 (ICIER_1)
- I²C bus status register_1 (ICSR_1)
- Slave address register_1 (SAR_1)
- I²C bus transmit data register_1 (ICDRT_1)
- I²C bus receive data register_1 (ICDRR_1)
- I²C bus shift register_1 (ICDRS_1)

RENESAS

13.3.1   I²C Bus Control Register A (ICCRA)

ICCRA enables or disables I²C bus interface, controls transmission or reception, and selects master or slave mode, transmission or reception, and transfer clock frequency in master mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	ICE	RCVD	MST	TRS	CKS3	CKS2	CKS1	CKS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	ICE	0	R/W	I²C Bus Interface Enable
				0: This module is halted
				1: This bit is enabled for transfer operations (SCL and SDA pins are bus drive state)
6	RCVD	0	R/W	Reception Disable
				This bit enables or disables the next operation when TRS is 0 and ICDRR is read.
				0: Enables next reception
				1: Disables next reception
5	MST	0	R/W	Master/Slave Select
4	TRS	0	R/W	Transmit/Receive Select
				When arbitration is lost in master mode, MST and TRS are both reset by hardware, causing a transition to slave receive mode. Modification of the TRS bit should be made between transfer frames.
				Operating modes are described below according to MST and TRS combination.
				00: Slave receive mode
				01: Slave transmit mode
				10: Master receive mode
				11: Master transmit mode
3	CKS3	0	R/W	Transfer Clock Select 3 to 0
2	CKS2	0	R/W	These bits are valid only in master mode. Make setting according to the required transfer rate. For details on the transfer rate, see table 13.2.
1	CKS1	0	R/W	
0	CKS0	0	R/W	

RENESAS

Table 13.2   Transfer Rate

Bit 3	Bit 2	Bit 1	Bit 0		Transfer Rate		
CKS3	CKS2	CKS1	CKS0	Clock	Pφ = 8 MHz	Pφ = 10 MHz	Pφ = 20 MHz
0	0	0	0	Pφ/28	286 kHz	357 kHz	714 kHz
			1	Pφ/40	200 kHz	250 kHz	500 kHz
		1	0	Pφ/48	167 kHz	208 kHz	417 kHz
			1	Pφ/64	125 kHz	156 kHz	313 kHz
	1	0	0	Pφ/168	47.6 kHz	59.5 kHz	119 kHz
			1	Pφ/100	80.0 kHz	100 kHz	200 kHz
		1	0	Pφ/112	71.4 kHz	89.3 kHz	179 kHz
			1	Pφ/128	62.5 kHz	78.1 kHz	156 kHz
1	0	0	0	Pφ/56	143 kHz	179 kHz	357 kHz
			1	Pφ/80	100 kHz	125 kHz	250 kHz
		1	0	Pφ/96	83.3 kHz	104 kHz	208 kHz
			1	Pφ/128	62.5 kHz	78.1 kHz	156 kHz
	1	0	0	Pφ/336	23.8 kHz	29.8 kHz	59.5 kHz
			1	Pφ/200	40.0 kHz	50.0 kHz	100 kHz
		1	0	Pφ/224	35.7 kHz	44.6 kHz	89.3 kHz
			1	Pφ/256	31.3 kHz	39.1 kHz	78.1 kHz

13.3.2   I²C Bus Control Register B (ICCRB)

ICCRB issues start/stop condition, manipulates the SDA pin, monitors the SCL pin, and controls reset in the I²C control module.

Bit	7	6	5	4	3	2	1	0
Bit Name	BBSY	SCP	SDAO	—	SCLO	—	IICRST	—
Initial Value	0	1	1	1	1	1	0	1
R/W	R/W	R/W	R	R/W	R	—	R/W	—

Bit	Bit Name	Initial Value	R/W	Description
7	BBSY	0	R/W	Bus Busy
				This bit indicates whether the I²C bus is occupied or released and to issue start and stop conditions in master mode. This bit is set to 1 when the SDA level changes from high to low under the condition of SCL = high, assuming that the start condition has been issued. This bit is cleared to 0 when the SDA level changes from low to high under the condition of SDA = high, assuming that the stop condition has been issued. Follow this procedure also when repeating a start condition. To issue a start or stop condition, use the MOV instruction.
6	SCP	1	R/W	Start/Stop Condition Issue
				This bit controls the issuance of start or stop condition in master mode.
				To issue a start condition, write 1 to BBSY and 0 to SCP. A repeated start condition is issued in the same way. To issue a stop condition, write 0 to BBSY and 0 to SCP. This bit is always read as 1. If 1 is written, the data is not stored.
5	SDAO	1	R	This bit monitors the output level of SDA.
				0: When reading, the SDA pin outputs a low level
				1: When reading the SDA pin outputs a high level
4	—	1	R/W	Reserved
				The write value should always be 1.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	SCLO	1	R	This bit monitors the SCL output level.
				When reading and SCLO is 1, the SCL pin outputs a high level. When reading and SCLO is 0, the SCL pin outputs a low level.
2	—	1	—	Reserved
				This bit is always read as 0.
1	IICRST	0	R/W	IIC Control Module Reset
				This bit reset the IIC control module except the I²C registers. If hang-up occurs because of communication failure during I²C operation, by setting this bit to 1, the
0	—	1	—	Reserved
				This bit is always read as 1.

13.3.3   I²C Bus Mode Register (ICMR)

ICMR selects MSB first or LSB first, controls the master mode wait and selects the number of transfer bits.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	WAIT	—	—	BCWP	BC2	BC1	BC0
Initial Value	0	0	1	1	1	0	0	0
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved
				The write value should always be 0.
6	WAIT	0	R/W	Wait Insertion
				This bit selects whether to insert a wait after data transfer except for the acknowledge bit. When this bit is set to 1, after the falling of the clock for the last data bit, the low period is extended for two transfer clocks. When this bit is cleared to 0, data and the acknowledge bit are transferred consecutively with no waits inserted. The setting of this bit is invalid in slave mode.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
5, 4	—	All 1	—	Reserved
				These bits are always read as 1.
3	BCWP	1	R/W	BC Write Protect
				This bit controls the modification of the BC2 to BC0 bits. When modifying, this bit should be cleared to 0 and the MOV instruction should be used.
				0: When writing, the values of BC2 to BC0 are set
				1: When reading, 1 is always read
				When writing, the settings of BC2 to BC0 are invalid.
2	BC2	0	R/W	Bit Counter 2 to 0
1	BC1	0	R/W	These bits specify the number of bits to be transferred
0	BC0	0	R/W	next. The settings of these bits should be made during intervals between transfer frames. When setting these bits to a value other than 000, the setting should be made while the SCL line is low. The value return to 000 at the end of a data transfer including the acknowledge bit.
				000: 9
				001: 2
				010: 3
				011: 4
				100: 5
				101: 6
				110: 7
				111: 8
				I²C control module can be reset without setting the ports and initializing the registers.

RENESAS

13.3.4   I²C Bus Interrupt Enable Register (ICIER)

ICIER enables or disables interrupt sources and the acknowledge bits, sets the acknowledge bits to be transferred, and confirms the acknowledge bit to be received.

Bit	7	6	5	4	3	2	1	0
Bit Name	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR	ACKBT
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable
				When the TDRE bit in ICSR is set to 1, this bit enables or disables the transmit data empty interrupt (TXI) request.
				0: Transmit data empty interrupt (TXI) request is disabled
				1: Transmit data empty interrupt (TXI) request is enabled
6	TEIE	0	R/W	Transmit End Interrupt Enable
				This bit enables or disables the transmit end interrupt (TEI) request at the rising of the ninth clock while the TDRE bit in ICSR is set to 1. The TEI request can be canceled by clearing the TEND bit or the TEIE bit to 0.
				0: Transmit end interrupt (TEI) request is disabled
				1: Transmit end interrupt (TEI) request is enabled
5	RIE	0	R/W	Receive Interrupt Enable
				This bit enables or disables the receive full interrupt (RXI) request when receive data is transferred from ICDRS to ICDRR and the RDRF bit in ICSR is set to 1. The RXI request can be canceled by clearing the RDRF or RIE bit to 0.
				0: Receive data full interrupt (RXI) request is disabled
				1: Receive data full interrupt (RXI) request is enabled

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
4	NAKIE	0	R/W	NACK Receive Interrupt Enable
				This bit enables or disables the NACK receive interrupt (NAKI) request when the NACKF and AL bits in ICSR are set to 1. The NAKI request can be canceled by clearing the NACKF or AL bit, or the NAKIE bit to 0.
				0: NACK receive interrupt (NAKI) request is disabled
				1: NACK receive interrupt (NAKI) request is enabled
3	STIE	0	R/W	Stop Condition Detection Interrupt Enable
				0: Stop condition detection interrupt (STPI) request is disabled
				1: Stop condition detection interrupt (STPI) request is enabled
2	ACKE	0	R/W	Acknowledge Bit Decision Select
				0: The value of the acknowledge bit is ignored and continuous transfer is performed
				1: If the acknowledge bit is 1, continuous transfer is suspended
1	ACKBR	0	R	Receive Acknowledge
				In transmit mode, this bit stores the acknowledge data that are returned by the receive device. This bit cannot be modified.
				0: Receive acknowledge = 0
				1: Receive acknowledge = 1
0	ACKBT	0	R/W	Transmit Acknowledge
				In receive mode, this bit specifies the bit to be sent at the acknowledge timing.
				0: 0 is sent at the acknowledge timing
				1: 1 is sent at the acknowledge timing

RENESAS

13.3.5   I²C Bus Status Register (ICSR)

ICSR confirms the interrupt request flags and status.

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	TEND	RDRF	NACKF	STOP	AL	AAS	ADZ
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	0	R/W	Transmit Data Register Empty [Setting condition] • When data is transferred from ICDRT to ICDRS and ICDRT becomes empty • When TRS is set to 1 • When the start condition has been issued • In slave mode, after changing from receive mode to transmit mode [Clearing conditions] • When 0 is written to this bit after reading TDRE = 1 • When data is written to ICDRT (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
6	TEND	0	R/W	Transmit End [Setting condition] • When the ninth clock of SCL rises while the TDRE flag is 1 [Clearing conditions] • When 0 is written to this bit after reading TEND = 1 • When data is written to ICDRT (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
5	RDRF	0	R/W	Receive Data Register Full
				[Setting condition]
				• When receive data is transferred from ICDRS to ICDRR
				[Clearing conditions]
				• When 0 is written to this bit after reading RDRF = 1
				• When data is read from ICDRR
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
4	NACKF	0	R/W	No Acknowledge Detection Flag
				[Setting condition]
				• When no acknowledge is detected from the receive device in transmission while the ACKE bit in ICIER is set to 1
				[Clearing condition]
				• When 0 is written to this bit after reading NACKF = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	STOP	0	R/W	Stop Condition Detection Flag
				[Setting conditions]
				• When a stop condition is detected after the frame transfer completion in master mode
				• In slave mode, when the slave address in the first byte after detecting the start condition and the address set in SAR match, and then a stop condition is detected.
				[Clearing condition]
				• When 0 is written to this bit after reading STOP = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial Value	R/W	Description
2	AL	0	R/W	Arbitration Lost Flag
				This flag indicates that arbitration was lost in master mode.
				When two or more master devices attempt to seize the bus at nearly the same time, the I²C bus monitors SDA, and if the I²C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master.
				[Setting conditions]
				• When the internal SDA and the SDA pin level disagree at the rising of SCL in master transmit mode
				• When the SDA pin outputs a high level in master mode while a start condition is detected
				[Clearing condition]
				• When 0 is written to this bit after reading AL = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	AAS	0	R/W	Slave Address Recognition Flag
				In slave receive mode, this flag is set to 1 when the first frame following a start condition matches bits SVA6 to SVA0 in SAR.
				[Setting conditions]
				• When the slave address is detected in slave receive mode
				• When the general call address is detected in slave receive mode
				[Clearing condition]
				• When 0 is written to this bit after reading AAS = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
0	ADZ	0	R/W	General Call Address Recognition Flag
				This bit is valid in slave receive mode.
				[Setting condition]
				• When the general call address is detected in slave receive mode
				[Clearing condition]
				• When 0 is written to this bit after reading ADZ = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

13.3.6   Slave Address Register (SAR)

SAR is sets the slave address. In slave mode, if the upper 7 bits of SAR match the upper 7 bits of the first frame received after a start condition, the LSI operates as the slave device.

Bit	7	6	5	4	3	2	1	0
Bit Name	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	SVA6	0	R/W	Slave Address 6 to 0
6	SVA5	0	R/W	These bits set a unique address differing from the addresses of other slave devices connected to the I²C bus.
5	SVA4	0	R/W	
4	SVA3	0	R/W	
3	SVA2	0	R/W	
2	SVA1	0	R/W	
1	SVA0	0	R/W	
0	—	0	R/W	Reserved
				Although this bit is readable/writable, only 0 should be written to.

13.3.7   I²C Bus Transmit Data Register (ICDRT)

ICDRT is an 8-bit readable/writable register that stores the transmit data. When ICDRT detects a space in the I²C bus shift register, it transfers the transmit data which has been written to ICDRT to ICDRS and starts transmitting data. If the next data is written to ICDRT during transmitting data to ICDRS, continuous transmission is possible.

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

13.3.8   I²C Bus Receive Data Register (ICDRR)

ICDRR is an 8-bit read-only register that stores the receive data. When one byte of data has been received, ICDRR transfers the receive data from ICDRS to ICDRR and the next data can be received. ICDRR is a receive-only register; therefore, this register cannot be written to by the CPU.

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

13.3.9   I²C Bus Shift Register (ICDRS)

ICDRS is an 8-bit write-only register that is used to transmit/receive data. In transmission, data is transferred from ICDRT to ICDRS and the data is sent from the SDA pin. In reception, data is transferred from ICDRS to ICDRR after one by of data is received. This register cannot be read from the CPU.

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

RENESAS

13.4　Operation

13.4.1　I²C Bus Format

Figure 13.3 shows the I²C bus formats. Figure 13.4 shows the I²C bus timing. The first frame following a start condition always consists of 8 bits.
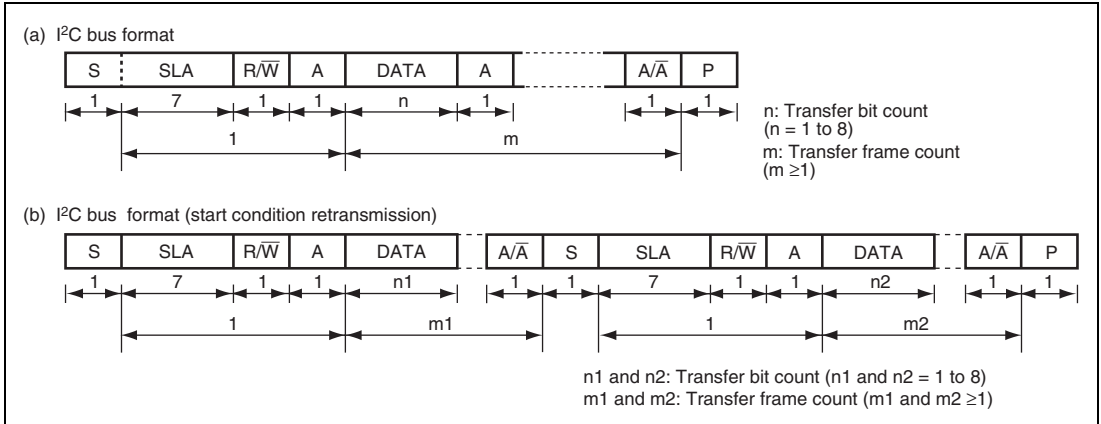
Figure 13.3　I²C Bus Formats
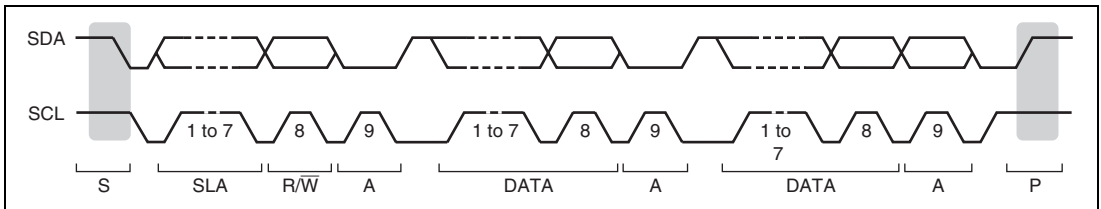
Figure 13.4　I²C Bus Timing

Legend:

S: 　　Start condition. The master device drives SDA from high to low while SCL is high.

SLA: 　Slave address

R/$\overline{\text{W}}$: 　Indicates the direction of data transfer; from the slave device to the master device when R/$\overline{\text{W}}$ is 1, or from the master device to the slave device when R/$\overline{\text{W}}$ is 0.

A: 　　Acknowledge. The receive device drives SDA low.

DATA: Transferred data

P: 　　Stop condition. The master device drives SDA from low to high while SCL is high.

13.4.2   Master Transmit Operation

In I²C bus format master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device return an acknowledge signal. Figures 13.5 and 13.6 show the operating timings in master transmit mode. The transmission procedure and operations in master transmit mode are described below.

1. Set the ICR bit in the corresponding register to 1. Set the ICE bit in ICCRA to 1. Set the WAIT bit in ICMR and the CKS3 to CKS0 bits in ICCRA to 1. (Initial setting)
2. Read the BSSY flag in ICCRB to confirm that the bus is free. Set the MST and TRS bits in ICCRA to select master transmit mode. Then, write 1 to BBSY and 0 to SCP using the MOV instruction. (The start condition is issued.) This generates the start condition.
3. After confirming that TDRE in ICSR has been set, write the transmit data (the first byte shows the slave address and R/W) to ICDRT. After this, when TDRE is automatically cleared to 0, data is transferred from ICDRT to ICDRS. TDRE is set again.
4. When transmission of one byte data is completed while TDRE is 1, TEND in ICSR is set to 1 at the rising of the ninth transmit clock pulse. Read the ACKBR bit in ICIER to confirm that the slave device has been selected. Then, write the second byte data to ICDRT. When ACKBR is 1, the slave device has not been acknowledged, so issue a stop condition. To issue the stop condition, write 0 to BBSY and SCP using the MOV instruction. SCL is fixed to a low level until the transmit data is prepared or the stop condition is issued.
5. The transmit data after the second byte is written to ICDRT every time TDRE is set.
6. Write the number of bytes to be transmitted to ICDRT. Wait until TEND is set (the end of last byte data transmission) while TDRE is 1, or wait for NACK (NACKF in ICSR is 1) from the receive device while CKE in ICIER is 1. Then, issue the stop condition to clear TEND or NACKF.
7. When the STOP bit in ICSR is set to 1, the operation returns to the slave receive mode.

RENESAS
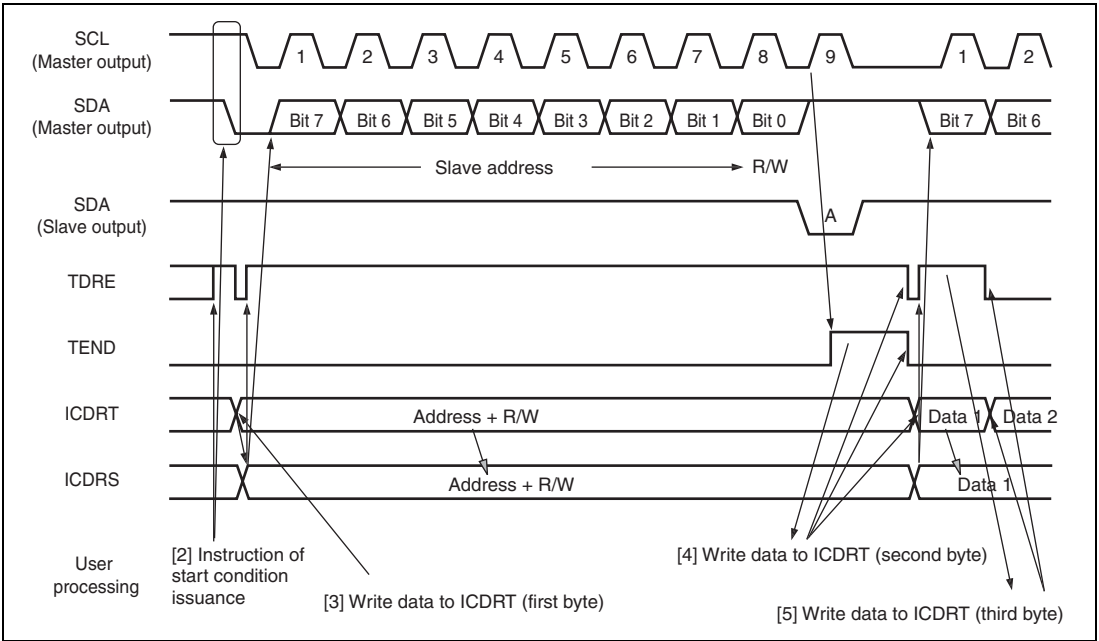
Figure 13.5   Master Transmit Mode Operation Timing 1
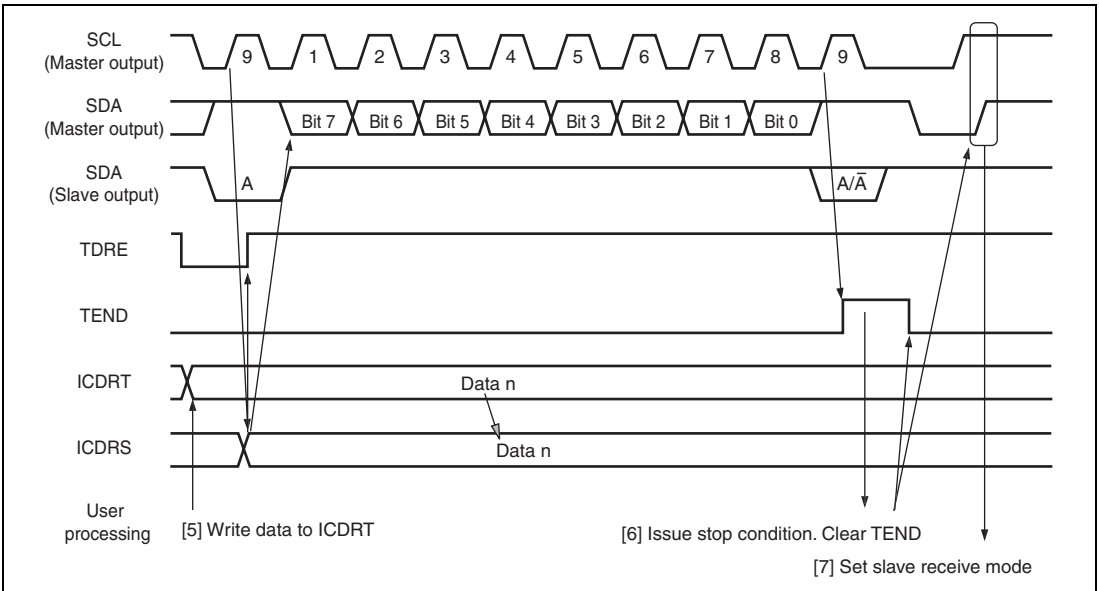
Figure 13.6   Master Transmit Mode Operation Timing 2

13.4.3   Master Receive Operation

In master receive mode, the master device outputs the receive clock, receives data from the slave device, and returns an acknowledge signal. Figures 13.7 and 13.8 show the operation timings in master receive mode. The reception procedure and operations in master receive mode are shown below.

1. Clear the TEND bit in ICSR to 0, then clear the TRS bit in ICCRA to 0 to switch from master transmit mode to master receive mode. Then, clear the TDRE bit to 0.
2. When ICDDR is read (dummy read), reception is started, the receive clock pulse is output, and data is received, in synchronization with the internal clock. The master mode outputs the level specified by the ACKBT in ICIER to SDA, at the ninth receive clock pulse.
3. After the reception of the first frame data is completed, the RDRF bit in ICSR is set to 1 at the rising of the ninth receive clock pulse. At this time, the received data is read by reading ICDRR. At the same time, RDRF is cleared.
4. The continuous reception is performed by reading ICDRR and clearing RDRF to 0 every time RDRF is set. If the eighth receive clock pulse falls after reading ICDRR by other processing while RDRF is 1, SCL is fixed to a low level until ICDRR is read.
5. If the next frame is the last receive data, set the RCVD bit in ICCR1 before reading ICDRR. This enables the issuance of the stop condition after the next reception.
6. When the RDRF bit is set to 1 at the rising of the ninth receive clock pulse, the stop condition is issued.
7. When the STOP bit in ICSR is set to 1, read ICDRR and clear RCVD to 0.
8. The operation returns to the slave receive mode.

RENESAS
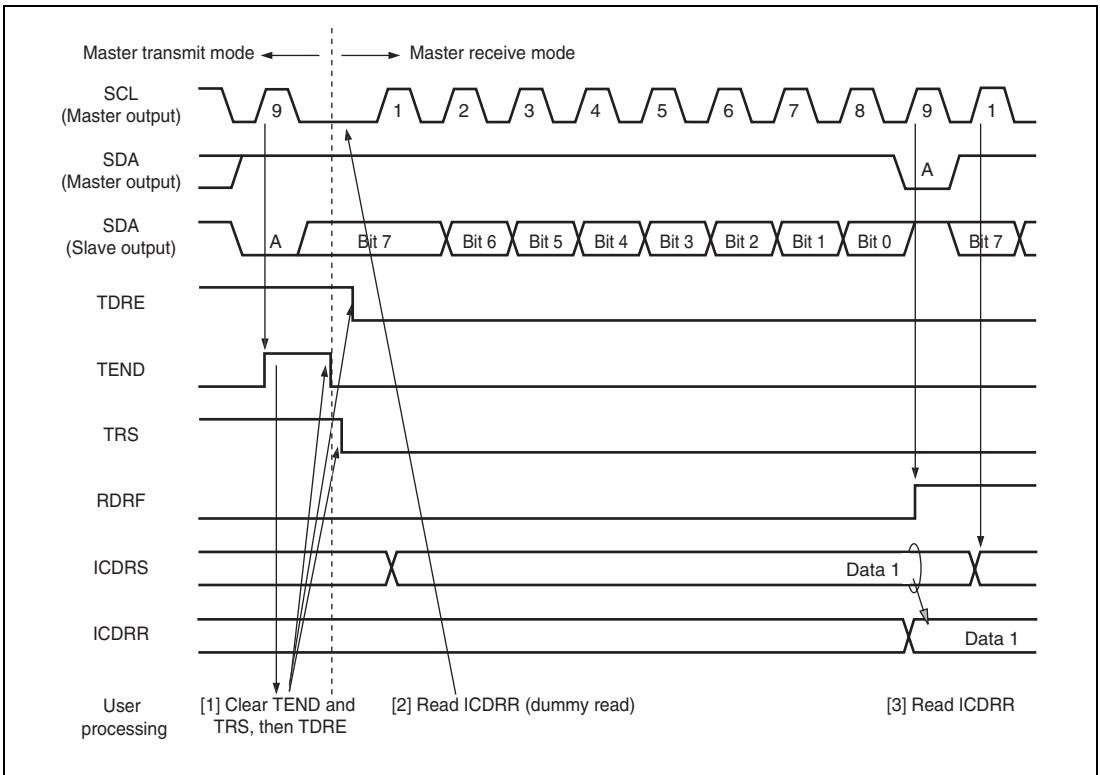
Figure 13.7   Master Receive Mode Operation Timing 1
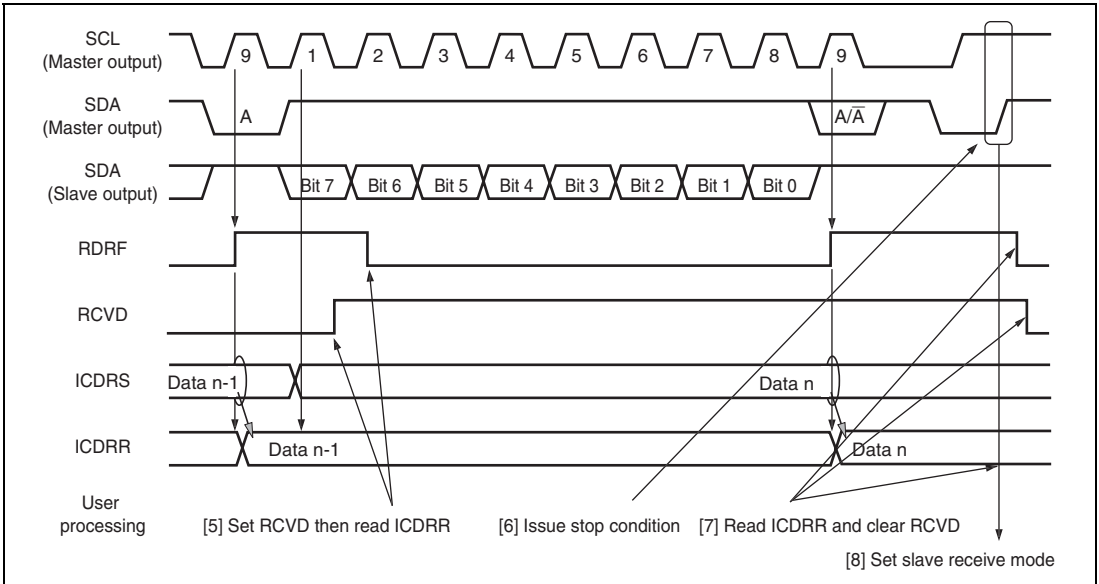
Figure 13.8   Master Receive Mode Operation Timing 2

13.4.4   Slave Transmit Operation

In slave transmit mode, the slave device outputs the transmit data, and the master device outputs the receive clock pulse and returns an acknowledge signal. Figures 13.9 and 13.10 show the operation timings in slave transmit mode. The transmission procedure and operations in slave transmit mode are described below.

1. Set the ICR bit in the corresponding register to 1, then set the ICE bit in ICCRA to 1. Set the WAIT in ICMR and CKS3 to CKS0 in ICCRA, and perform other initial settings. Set the MST and TRS bits in ICCRA to select slave receive mode, and wait until the slave address matches.

2. When the slave address matches in the first frame following the detection of the start condition, the slave device outputs the level specified by ACKBT in ICIER to SDA, at the rising of the ninth clock pulse. At this time, if the eighth bit data (R/$\overline{\text{W}}$) is 1, TRS in ICCRA and TDRE in ICSR are set to 1, and the mode changes to slave transmit mode automatically. The continuous transmission is performed by writing the transmit data to ICDRT every time TDRE is set.

3. If TDRE is set after writing the last transmit data to ICDRT, wait until TEND in ICSR is set to 1, with TDRE = 1. When TEND is set, clear TEND.

4. Clear TRS for end processing, and read ICDRR (dummy read) to free SCL.

5. Clear TDRE.

RENESAS
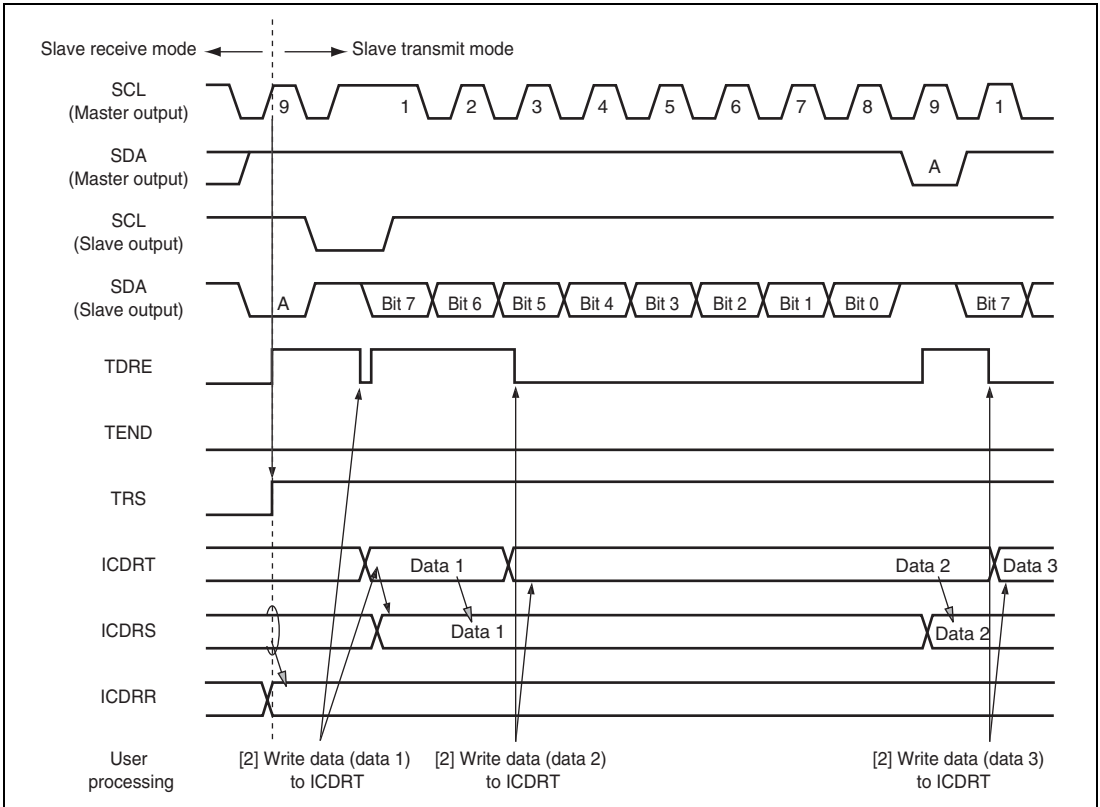
Figure 13.9   Slave Transmit Mode Operation Timing 1
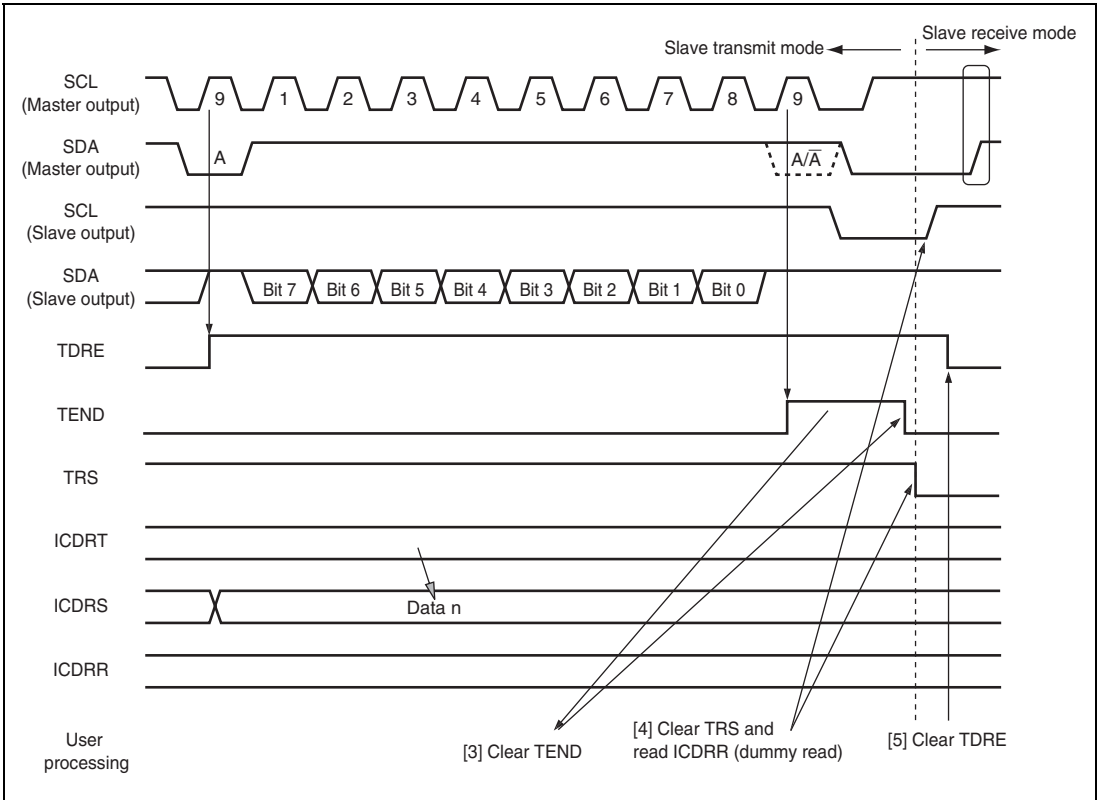
Figure 13.10   Slave Transmit Mode Operation Timing 2

13.4.5   Slave Receive Operation

In slave receive mode, the master device outputs the transmit clock and the transmit data, and the slave device returns an acknowledge signal. Figures 13.11 and 13.12 show the operation timings in slave receive mode. The reception procedure and operations in slave receive mode are described below.

1. Set the ICR bit in the corresponding register to 1. Then, set the ICE bit in ICCRA to 1. Set the WAIT in ICMR and CKS3 to CKS0 in ICCRA, and perform other initial settings. Set the MST and TRS bits in ICCRA to select slave receive mode and wait until the slave address matches.

2. When the slave address matches in the first frame following detection of the start condition, the slave address outputs the level specified by ACKBT in ICIER to SDA, at the rising of the ninth clock pulse. At the same time, RDRF in ICSR is set to read ICDRR (dummy read). (Since the read data shows the slave address and R/$\overline{\text{W}}$, it is not used).

3. Read ICDRR every time RDRF is set. If the eighth clock pulse falls while RDRF is 1, SCL is fixed to a low level until RDRF is cleared. The change of the acknowledge (ACKBT) setting before clearing RDRF to be returned to the master device is reflected in the next transmit frame.

4. The last byte data is read by reading ICDRR.
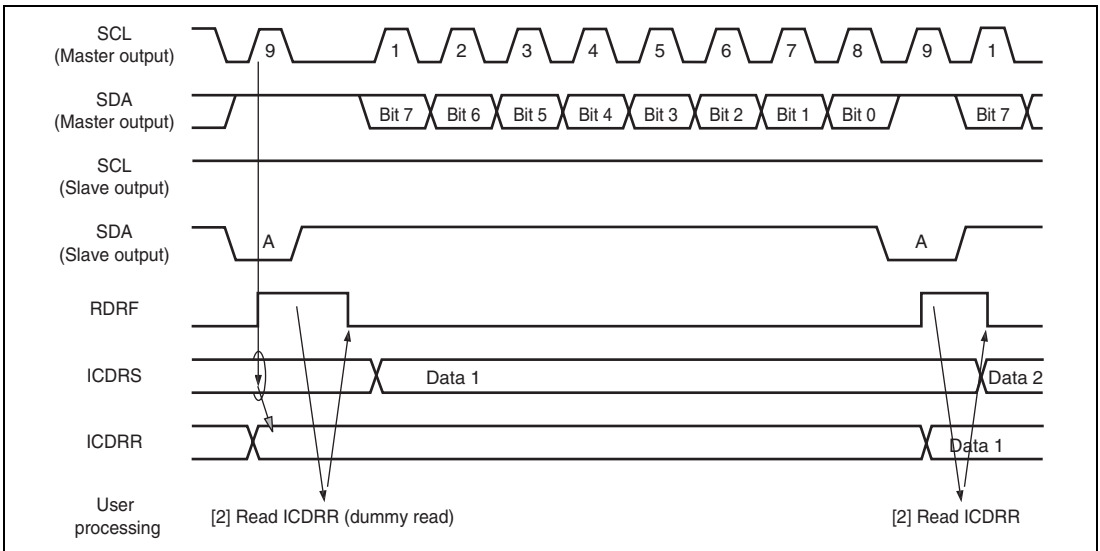
Figure 13.11   Slave Receive Mode Operation Timing 1
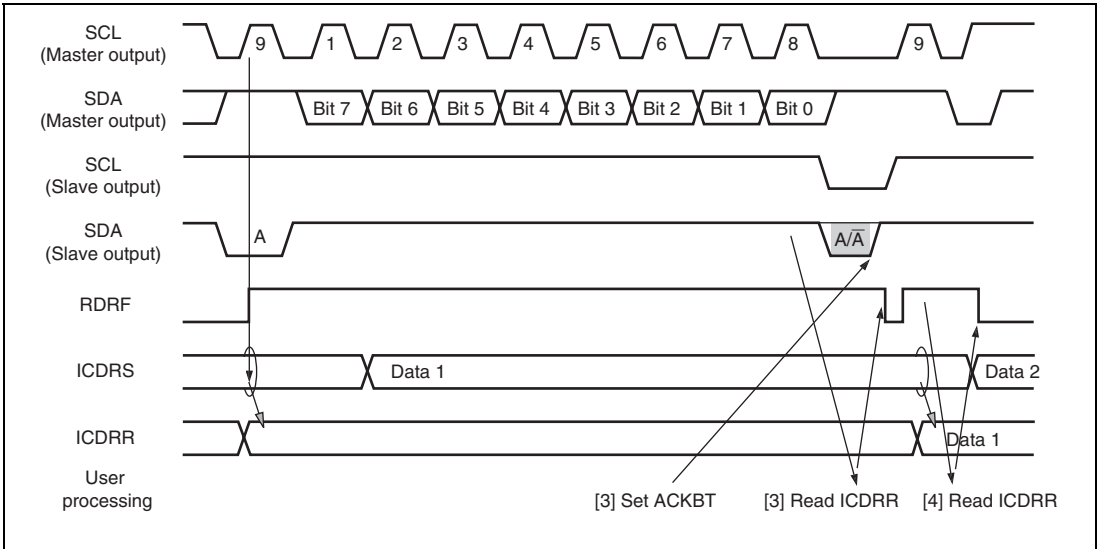
Figure 13.12   Slave Receive Mode Operation Timing 2

13.4.6   Noise Canceller

The logic levels at the SCL and SDA pins are routed through the noise cancellers before being latched internally. Figure 13.13 shows a block diagram of the noise canceller circuit.

The noise canceller consists of two cascaded latches and a match detector. The signal input to SCL (or SDA) is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.

Figure 13.13   Block Diagram of Noise Canceller

13.4.7   Example of Use

Sample flowcharts in respective modes that use the I²C bus interface are shown in figures 13.14 to 13.17.

The flowchart on the left shows the following steps with corresponding descriptions:

[1]   Detect the state of the SCL and SDA lines

[2]   Set to master transmit mode

[3]   Issue the start condition

[4]   Set the transmit data for the first byte (slave address + R/W)

[5]   Wait for 1 byte of data to be transmitted

[6]   Detect the acknowledge bit, transferred from the specified slave device

[7]   Set the transmit data for the second and subsequent data (except for the last byte)

[8]   Wait for ICDRT empty

[9]   Set the last byte of transmit data

[10]   Wait for the completion of transmission of the last byte

[11]   Clear the TEND flag

[12]   Clear the STOP flag

[13]   Issue the stop condition

[14]   Wait for the creation of the stop condition

[15]   Set to slave receive mode. Clear TDRE.

Figure 13.14   Sample Flowchart of Master Transmit Mode

```
┌──────────────────────────────┐
│      Master receive mode      │
├──────────────────────────────┤
│      Clear TEND in ICSR       │
├──────────────────────────────┤
│      Set TRS = 0 (ICCRA)      │  [1]
├──────────────────────────────┤
│      Clear TDRE in ICSR       │
├──────────────────────────────┤
│     Set ACKBT = 0 (ICIER)     │  [2]
├──────────────────────────────┤
│       Dummy read ICDRR        │  [3]
├──────────────────────────────┤
│       Dummy read ICSR         │
├──────────────────────────────┤
│  No       RDRF = 1?           │  [4]
├──────────────────────────────┤
│   Yes                         │
│     Last        Yes           │  [5]
│   receive -1?                 │
│      No                       │
├──────────────────────────────┤
│        Read ICDRR             │  [6]
├──────────────────────────────┤
│     Set ACKBT = 1 (ICIER)     │
├──────────────────────────────┤
│     Set RCVD = 1 (ICCRA)      │  [7]
├──────────────────────────────┤
│        Read ICDRR             │  [8]
├──────────────────────────────┤
│      Read RDRF in ICSR        │
├──────────────────────────────┤
│  No       RDRF = 1?           │  [9]
├──────────────────────────────┤
│   Yes                         │
│      Clear STOP in ICSR       │  [10]
├──────────────────────────────┤
│     Write BBSY = 0 and        │  [11]
│         SCP = 0               │
├──────────────────────────────┤
│       Read STOP in ICSR       │
├──────────────────────────────┤
│  No        STOP = 1?          │  [12]
├──────────────────────────────┤
│   Yes                         │
│        Read ICDRR             │  [13]
├──────────────────────────────┤
│     Set RCVD = 0 (ICCRA)      │  [14]
├──────────────────────────────┤
│      Set MST = 0 (ICCRA)      │  [15]
├──────────────────────────────┤
│           End                 │
└──────────────────────────────┘
```

[1] Clear TEND, set to master receive mode, then clear TDRE*

[2] Set acknowledge to the transmitting device*

[3] Dummy read ICDRR*

[4] Wait for 1 byte of data to be received

[5] Check if (last receive -1)

[6] Read the receive data

[7] Set acknowledge of the last byte. Disable continuous reception (RCVD = 1).

[8] Read receive data of (last byte -1).

[9] Wait for the last byte to be received

[10] Clear the STOP flag

[11] Issue the stop condition

[12] Wait for the creation of stop condition

[13] Read the receive data of the last byte

[14] Clear RCVD to 0

[15] Set to slave receive mode

Note: * Do not generate an interrupt during steps [1] to [3].

Figure 13.15   Sample Flowchart for Master Receive Mode

RENESAS
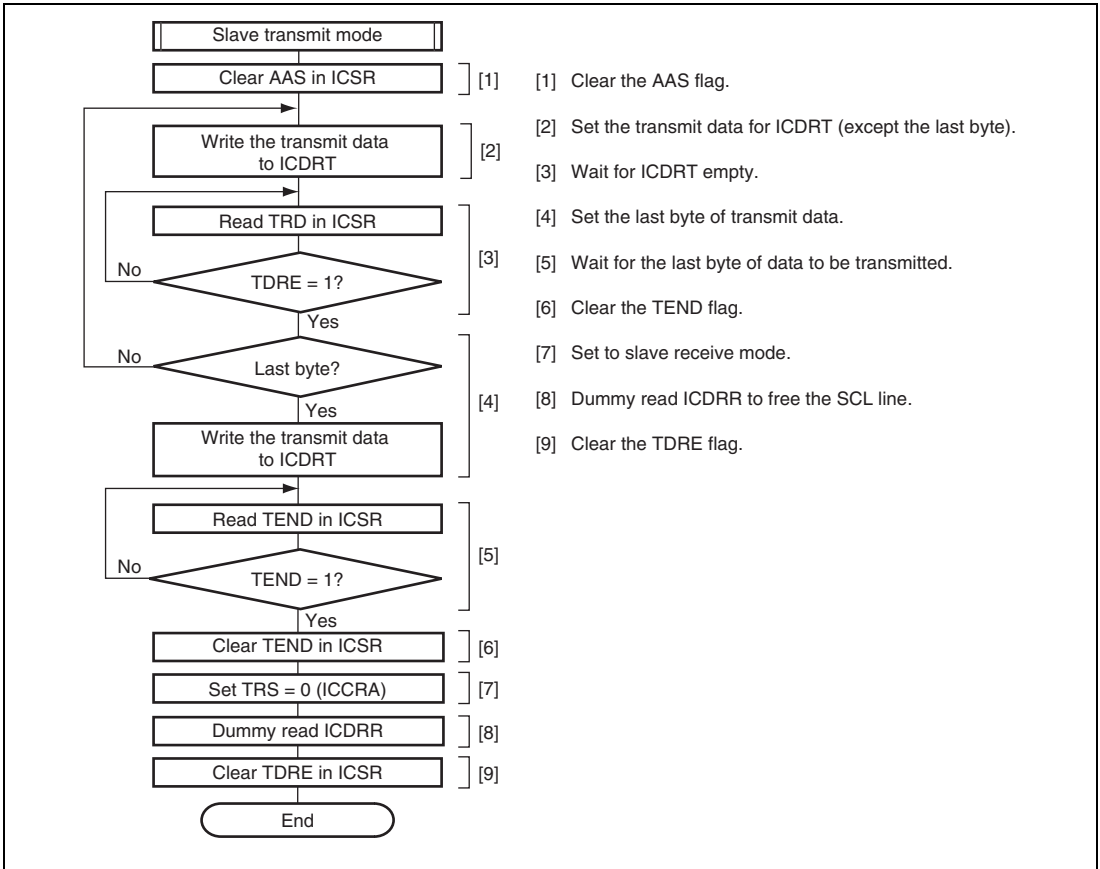
Figure 13.16   Sample Flowchart for Slave Transmit Mode

RENESAS

```
┌─────────────────────────────────────────────────────────────────────────────────────┐
│   ┌───────────────────────┐                                                           │
│   │   Slave receive mode  │                                                           │
│   └───────────────────────┘                                                           │
│   ┌───────────────────────┐  ] [1]     [1]   Clear the AAS flag.                       │
│   │   Clear AAS in ICSR    │                                                           │
│   └───────────────────────┘            [2]   Set the acknowledge for the transmit device. │
│   ┌───────────────────────┐  ] [2]                                                     │
│   │ Set ACKBT = 0 in ICIER │            [3]   Dummy read ICDRR                          │
│   └───────────────────────┘                                                            │
│   ┌───────────────────────┐  ] [3]     [4]   Wait for 1 byte of data to be received*    │
│   │   Dummy read ICDRR     │                                                            │
│   └───────────────────────┘            [5]   Detect (last reception -1)                 │
│   ┌───────────────────────┐                                                            │
│   │   Read RDRF in ICSR    │  ]        [6]   Read the receive data.                     │
│   └───────────────────────┘   [4]                                                      │
│ No      < RDRF = 1? >                   [7]   Set the acknowledge for the last byte.     │
│             │ Yes                       [8]   Read the receive data of (last byte -1).   │
│      < The last          > Yes  ] [5]                                                  │
│      < reception -1?     >              [9]   Wait for the reception of the last byte to be completed. │
│             │ No                                                                        │
│   ┌───────────────────────┐  ] [6]    [10]   Read the last byte of receive data.        │
│   │    Read ICDRR          │                                                            │
│   └───────────────────────┘                                                            │
│   ┌───────────────────────┐  ] [7]                                                     │
│   │ Set ACKBT = 1 in ICIER │                                                           │
│   └───────────────────────┘                                                            │
│   ┌───────────────────────┐  ] [8]                                                     │
│   │    Read ICDRR          │                                                           │
│   └───────────────────────┘                                                            │
│   ┌───────────────────────┐                                                            │
│   │   Read RDRF in ICSR    │  ] [9]                                                     │
│   └───────────────────────┘                                                            │
│ No      < RDRF = 1? >                                                                   │
│             │ Yes                                                                       │
│   ┌───────────────────────┐  ] [10]                                                    │
│   │    Read ICDRR          │                                                           │
│   └───────────────────────┘                                                            │
│         (   End   )                                                                     │
└─────────────────────────────────────────────────────────────────────────────────────┘
```

Figure 13.17   Sample Flowchart for Slave Receive Mode

RENESAS

13.5   Interrupt Request

There are six interrupt requests in this module; transmit data empty, transmit end, receive data full, NACK detection, STOP recognition, and arbitration lost. Table 13.3 shows the contents of each interrupt request.

Table 13.3   Interrupt Requests

Interrupt Request	Abbreviation	Interrupt Condition
Transmit Data Empty	TXI	(TDRE = 1) · (TIE = 1)
Transmit End	TEI	(TEND = 1) · (TEIE = 1)
Receive Data Full	RXI	(RDRF = 1) · (RIE = 1)
Stop Recognition	STPI	(STOP = 1) · (STIE = 1)
NACK Detection	NAKI	{(NACKF = 1) + (AL = 1)} · (NAKIE = 1)
Arbitration Lost		

If an interrupt condition described in table 13.3 is set to 1 and if the I bit in CCR is cleared to 0, the CPU executes the exception handling corresponding to the interrupt. During the exception handling, appropriate interrupt source needs to be cleared. However, note that TDRE and TEND are cleared automatically by writing transmit data to ICDRT and that RDRF is also cleared automatically by reading ICDRR. Especially not that TDRE is set to 1 again simultaneously when transmit data is written to ICDRT; thus, additional clearing TDRE may cause one more byte to be transmitted erroneously.

13.6 Bit Synchronous Circuit

This module has a possibility that the high-level period is shortened in the two states described below.

In master mode,

- When SCL is driven low by the slave device
- When the rising speed of SCL is lowered by the load on the SCL line (load capacitance or pull-up resistance)

Therefore, this module monitors SCL and communicates bit by bit in synchronization.

Figure 13.18 shows the timing of the bit synchronous circuit, and table 13.4 shows the time when SCL output changes from low to Hi-Z and the period which SCL is monitored.
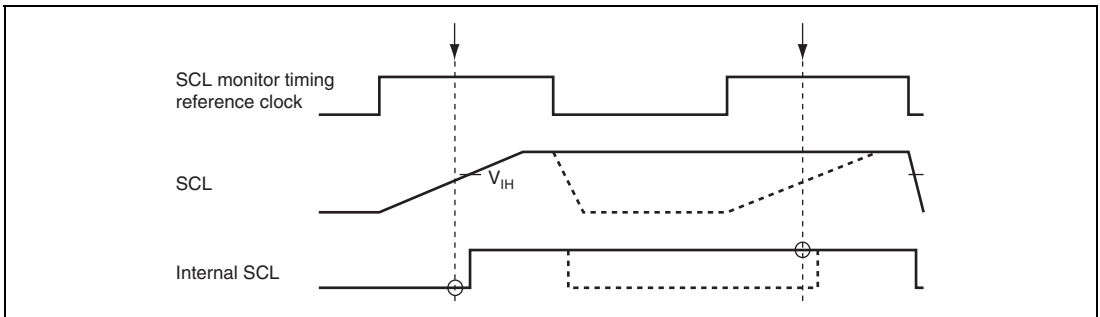
Figure 13.18   Timing of the Bit Synchronous Circuit

Table 13.4   Time for Monitoring SCL

CKS3	CKS2	Time for Monitoring SCL
0	0	7.5 tcyc
	1	19.5 tcyc
1	0	17.5 tcyc
	1	41.5 tcyc

Section 14   Controller Area Network (RCAN-ET)

14.1   Summary

14.1.1   Overview

This document primarily describes the programming interface for the RCAN-ET module. It serves to facilitate the hardware/software interface so that engineers involved in the RCAN-ET implementation can ensure the design is successful.

14.1.2   Scope

The CAN Data Link Controller function is not described in this document. It is the responsibility of the reader to investigate the CAN Specification Document (see references). The interfaces from the CAN Controller are described, in so far as they pertain to the connection with the User Interface.

The programming model is described in some detail. It is not the intention of this document to describe the implementation of the programming interface, but to simply present the interface to the underlying CAN functionality.

The document places no constraints upon the implementation of the RCAN-ET module in terms of process, packaging or power supply criteria. These issues are resolved where appropriate in implementation specifications.

14.1.3   Audience

In particular this document provides the design reference for software authors who are responsible for creating a CAN application using this module.

In the creation of the RCAN-ET user interface LSI engineers must use this document to understand the hardware requirements.

14.1.4   References

1. CAN Specification Version 2.0 part A, Robert Bosch GmbH, 1991
2. CAN Specification Version 2.0 part B, Robert Bosch GmbH, 1991
3. Implementation Guide for the CAN Protocol, CAN Specification 2.0 Addendum, CAN In Automation, Erlangen, Germany, 1997
4. Road vehicles - Controller area network (CAN): Part 1: Data link layer and physical signalling (ISO-11898-1, 2003)

14.1.5   Features

- Supports CAN specification 2.0B
- Bit timing compliant with ISO-11898-1
- 16 Mailbox version
- Clock 16 to 24 MHz*
- 15 programmable Mailboxes for transmit/receive + 1 receive-only mailbox
- Sleep mode for low power consumption and automatic recovery from sleep mode by detecting CAN bus activity
- Programmable receive filter mask (standard and extended identifier) supported by all Mailboxes
- Programmable CAN data rate up to 1Mbit/s
- Transmit message queuing with internal priority sorting mechanism against the problem of priority inversion for real-time applications
- Data buffer access without software handshake requirement in reception
- Flexible micro-controller interface
- Flexible interrupt structure

Note: *   Refer to section 23.7.1, Notes on Clock Pulse Generator.

RENESAS

14.2   Architecture

14.2.1   Block Diagram

The RCAN-ET device offers a flexible and sophisticated way to organise and control CAN frames, providing the compliance to CAN2.0B Active and ISO-11898-1. The module is formed from 5 different functional entities. These are the Micro Processor Interface (MPI), Mailbox, Mailbox Control, and CAN Interface. Figure 14.1 shows the block diagram of the RCAN-ET Module. The bus interface timing is designed according to the peripheral bus I/F required for each product.
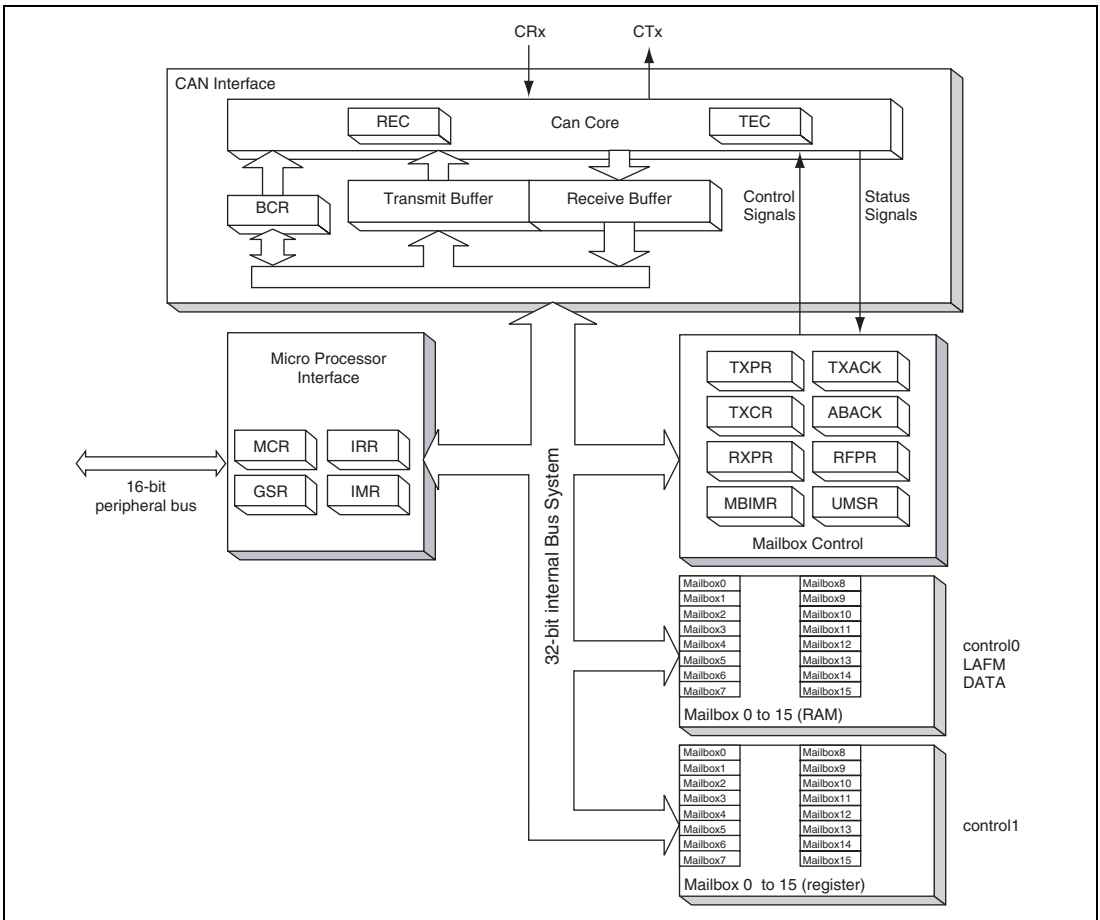
Figure 14.1   RCAN-ET Architecture

14.2.2   Important

Although core of RCAN-ET is designed based on a 32-bit bus system, the whole RCAN-ET including MPI for the CPU has 16-bit bus interface to CPU. In that case, LongWord (32-bit) access must be implemented as 2 consecutive word (16-bit) accesses. In this manual, LongWord access means the two consecutive accesses.

(1)   Micro Processor Interface (MPI)

The MPI allows communication between the Renesas CPU and RCAN-ET's registers/mailboxes to control the memory interface. It also contains the Wakeup Control logic that detects the CAN bus activities and notifies the MPI and the other parts of RCAN-ET so that the RCAN-ET can automatically exit the Sleep mode.

It contains registers such as MCR, IRR, GSR and IMR.

(2)   Mailbox

The Mailboxes consists of RAM configured as message buffers and registers. There are 16 Mailboxes, and each mailbox has the following information.

<RAM>

- CAN message control (identifier, rtr, ide,etc)
- CAN message data (for CAN Data frames)
- Local Acceptance Filter Mask for reception

<Registers>

- CAN message control (dlc)
- 3-bit wide Mailbox Configuration, Disable Automatic Re-Transmission bit, Auto-Transmission for Remote Request bit, New Message Control bit

RENESAS

(3)   Mailbox Control

The Mailbox Control handles the following functions:

- For received messages, compare the IDs and generate appropriate RAM addresses/data to store messages from the CAN Interface into the Mailbox and set/clear appropriate registers accordingly.
- To transmit messages, RCAN-ET will run the internal arbitration to pick the correct priority message, and load the message from the Mailbox into the Tx-buffer of the CAN Interface and set/clear appropriate registers accordingly.
- Arbitrates Mailbox accesses between the CPU and the Mailbox Control.
- Contains registers such as TXPR, TXCR, TXACK, ABACK, RXPR, RFPR, MBIMR, and UMSR.

(4)   CAN Interface

This block conforms to the requirements for the CAN Bus Data Link Controller which is specified in Ref. [2, 4]. It fulfils all the functions of the standard Data Link Controller as specified by the OSI 7 Layer Reference model. This functional entity also provides the registers and the logic which are specific to a given CAN bus, which includes the Receive Error Counter, Transmit Error Counter, the Bit Configuration Registers and various useful Test Modes. This block also contains functional entities to hold the data received and the data to be transmitted for the CAN Data Link Controller.

14.2.3   Input/Output Pins

Table 14.1 lists the RCAN-ET input and output pins.

Table 14.1   Pin Configuration of the RCAN-ET

Channel	Name	Abbreviation	I/O	Function
0	Transmit data pin	CTx_0	Output	CAN-bus transmit pin
	Receive data pin	CRx_0	Input	CAN-bus receive pin
1	Transmit data pin	CTx_1	Output	CAN-bus transmit pin
	Receive data pin	CRx_1	Input	CAN-bus receive pin

14.2.4   Memory Map

The diagram of the memory map is shown in figure 14.2.
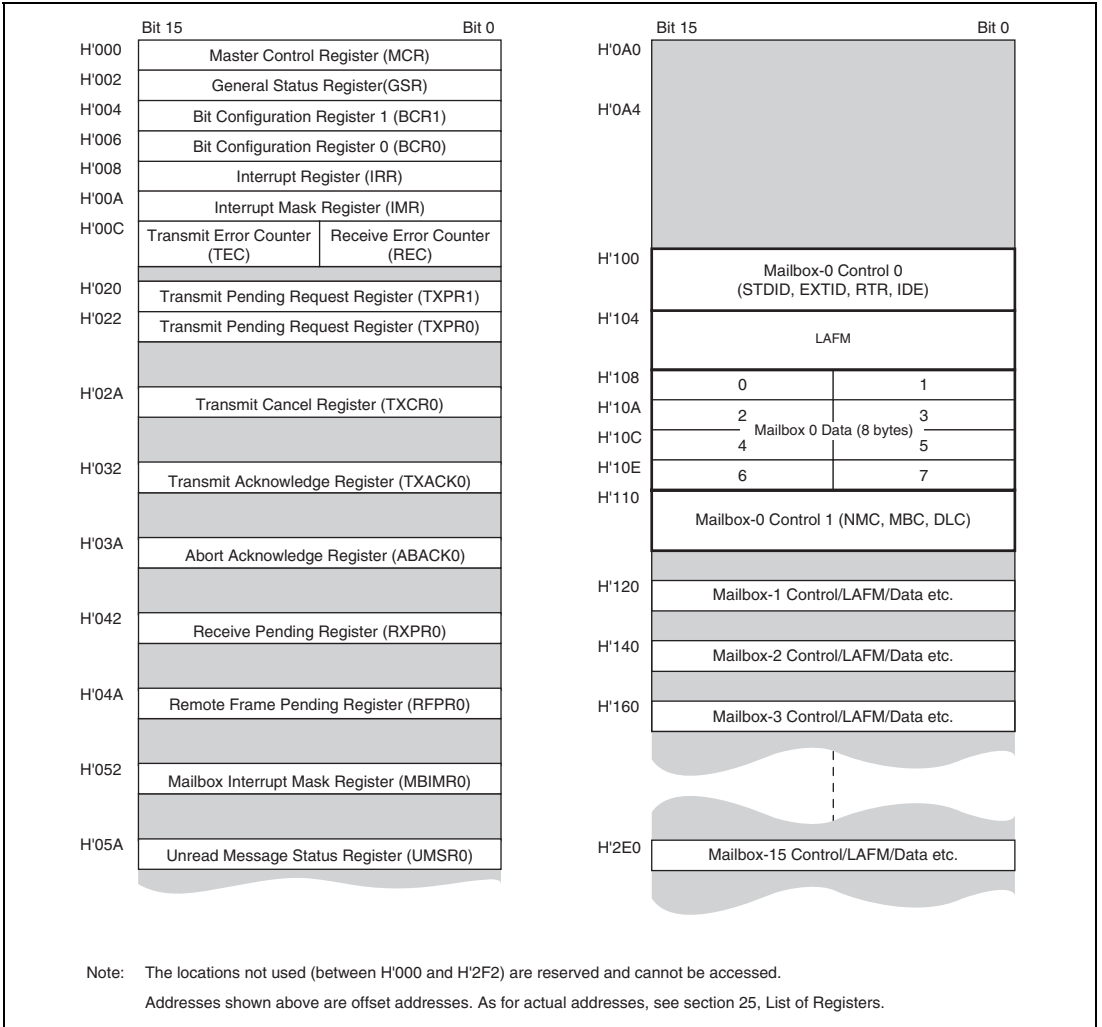
Figure 14.2   RCAN-ET Memory Map

14.3   Mailbox

14.3.1   Mailbox Structure

Mailboxes play a role as message buffers to transmit/receive CAN frames. Each Mailbox is comprised of 3 identical storage fields: Message Control, Local Acceptance Filter Mask, and Message Data. Table 14.2 shows the address map for the control, LAFM, data and addresses for each mailbox.

Table 14.2   Address Map for Each Mailbox

| Mailbox | Address | | | |
| | Control 0 | LAFM | Data | Control 1 |
	4 bytes	4 bytes	8 bytes	2 bytes
0 (Receive Only)	H'100 to H'103	H'104 to H'107	H'108 to H'10F	H'110 to H'111
1	H'120 to H'123	H'124 to H'127	H'128 to H'12F	H'130 to H'131
2	H'140 to H'143	H'144 to H'147	H'148 to H'14F	H'150 to H'151
3	H'160 to H'163	H'164 to H'167	H'168 to H'16F	H'170 to H'171
4	H'180 to H'183	H'184 to H'187	H'188 to H'18F	H'190 to H'191
5	H'1A0 to H'1A3	H'1A4 to H'1A7	H'1A8 to H'1AF	H'1B0 to H'1B1
6	H'1C0 to H'1C3	H'1C4 to H'1C7	H'1C8 to H'1CF	H'1D0 to H'1D1
7	H'1E0 to H'1E3	H'1E4 to H'1E7	H'1E8 to H'1EF	H'1F0 to H'1F1
8	H'200 to H'203	H'204 to H'207	H'208 to H'20F	H'210 to H'211
9	H'220 to H'223	H'224 to H'227	H'228 to H'22F	H'230 to H'231
10	H'240 to H'243	H'244 to H'247	H'248 to H'24F	H'250 to H'251
11	H'260 to H'263	H'264 to H'267	H'268 to H'26F	H'270 to H'271
12	H'280 to H'283	H'284 to H'287	H'288 to H'28F	H'290 to H'291
13	H'2A0 to H'2A3	H'2A4 to H'2A7	H'2A8 to H'2AF	H'2B0 to H'2B1
14	H'2C0 to H'2C3	H'2C4 to H'2C7	H'2C8 to H'2CF	H'2D0 to H'2D1
15	H'2E0 to H'2E3	H'2E4 to H'2E7	H'2E8 to H'2EF	H'2F0 to H'2F1

Mailbox-0 is a receive-only box, and all the other Mailboxes can operate as both receive and transmit boxes, dependant upon the MBC (Mailbox Configuration) bits in the Message Control. Figure 14.3 shows the structure of a Mailbox in detail.

Table 14.3   Roles of Mailboxes

	Tx	Rx
MB15 to 1	OK	OK
MB0	—	OK

MB0 (reception MB)   Byte: 8-bit access, Word: 16-bit access, LW (LongWord): 32-bit access

Regiter Name	Address	Data Bus 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Access Size	Field Name
MB[0].CONTROL0H	H'100	IDE	RTR	0	STDID[10:0]									EXTID[17:16]				Word/LW	Control 0
MB[0].CONTROL0L	H'102	EXTID[15:0]																Word	
MB[0].LAFMH	H'104	IDE_LAFM	0	0	STDID_LAFM[10:0]									EXTID_LAFM[17:16]				Word/LW	LAFM
MB[0].LAFML	H'106	EXTID_LAFM[15:0]																Word	
MB[0].MSG_DATA[0][1]	H'108	MSG_DATA_0 (first Rx/Tx Byte)								MSG_DATA_1								Byte/Word/LW	Data
MB[0].MSG_DATA[2][3]	H'10A	MSG_DATA_2								MSG_DATA_3								Byte/Word	
MB[0].MSG_DATA[4][5]	H'10C	MSG_DATA_4								MSG_DATA_5								Byte/Word/LW	
MB[0].MSG_DATA[6][7]	H'10E	MSG_DATA_6								MSG_DATA_7								Byte/Word	
MB[0].CONTROL1H, L	H'110	0	0	NMC	0	0	MBC[2:0]			0	0	0	0	DLC[3:0]				Byte/Word	Control 1

MB1 to 15 (MB for transmission/reception)

Register Name	Address	Data Bus 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Access Size	Field Name
MB[n].CONTROL0H	H'100 + n*32	IDE	RTR	0	STDID[10:0]									EXTID[17:16]				Word/LW	Control 0
MB[n].CONTROL0L	H'102 + n*32	EXTID[15:0]																Word	
MB[n].LAFMH	H'104 + n*32	IDE_LAFM	0	0	STDID_LAFM[10:0]									EXTID_LAFM[17:16]				Word/LW	LAFM
MB[n].LAFML	H'106 + n*32	EXTID_LAFM[15:0]																Word	
MB[n].MSG_DATA[0][1]	H'108 + n*32	MSG_DATA_0 (first Rx/Tx Byte)								MSG_DATA_1								Byte/Word/LW	Data
MB[n].MSG_DATA[2][3]	H'10A + n*32	MSG_DATA_2								MSG_DATA_3								Byte/Word	
MB[n].MSG_DATA[4][5]	H'10C + n*32	MSG_DATA_4								MSG_DATA_5								Byte/Word/LW	
MB[n].MSG_DATA[6][7]	H'10E + n*32	MSG_DATA_6								MSG_DATA_7								Byte/Word	
MB[n].CONTROL1H, L	H'110 + n*32	0	0	NMC	ATX	DART	MBC[2:0]			0	0	0	0	DLC[3:0]				Byte/Word	Control 1

Notes: n = 1 to 15 (Mailbox number)
1. All bits shadowed in grey are reserved and the write value should be 0. The value returned by a read may not always be 0 and should not be relied upon.
2. MBC1 bit in mailbox is fixed to 1.
3. ATX and DART are not supported by mailbox-0, and the MBC setting of mailbox-0 is limited.
4. When the MCR15 bit is 1, the order of STDID, RTR, IDE and EXTID of both message control and LAFM differs from HCAN2.

Figure 14.3   Mailbox-n Structure

14.3.2   Message Control Field

STDID[10:0]: These bits set the identifier (standard identifier) of data frames and remote frames.

EXTID[17:0]: These bits set the identifier (extended identifier) of data frames and remote frames.

RTR (Remote Transmission Request bit) : Used to distinguish between data frames and remote frames. This bit is overwritten by received CAN Frames depending on Data Frames or Remote Frames.

Important: Please note that, when ATX bit is set with the setting MBC = 001(bin), the RTR bit will never be set. When a Remote Frame is received, the CPU can be notified by the corresponding RFPR set or IRR[2] (Remote Frame Request Interrupt), however, as RCAN-ET needs to transmit the current message as a Data Frame, the RTR bit remains unchanged.

Important: In order to support automatic answer to remote frame when MBC = B'001 is used and ATX = 1 the RTR flag must be programmed to zero to allow data frame to be transmitted.

Note: When a Mailbox is specified to send a remote frame request, the DLC used for transmission is the one stored into the Mailbox.

RTR	Description
0	Data frame
1	Remote frame

IDE (Identifier Extension bit) : Used to distinguish between the standard format and extended format of CAN data frames and remote frames.

IDE	Description
0	Standard format
1	Extended format

RENESAS

- Mailbox-0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	NMC	0	0	MBC[2:0]			0	0	0	0	DLC[3:0]			
Initial value:	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R	R	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

Note: MBC[1] of MB0 is always "1".

- Mailbox-15 to 1

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	NMC	ATX	DART	MBC[2:0]			0	0	0	0	DLC[3:0]			
Initial value:	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

NMC (New Message Control): When this bit is set to '0', the Mailbox of which the RXPR or RFPR bit is already set does not store the new message but maintains the old one and sets the UMSR correspondent bit. When this bit is set to '1', the Mailbox of which the RXPR or RFPR bit is already set overwrites with the new message and sets the UMSR correspondent bit.

Important: Please note that if a remote frame is overwritten with a data frame or vice versa could be that both RXPR and RFPR flags (together with UMSR) are set for the same Mailbox. In this case the RTR bit within the Mailbox Control Field should be relied upon.

NMC	Description
0	Overrun mode (Initial value)
1	Overwrite mode

ATX (Automatic Transmission of Data Frame): When this bit is set to '1' and a Remote Frame is received into the Mailbox DLC is stored. Then, a Data Frame is transmitted from the same Mailbox using the current contents of the message data and updated DLC by setting the corresponding TXPR automatically. The scheduling of transmission is still governed by ID priority or Mailbox priority as configured with the Message Transmission Priority control bit (MCR2). In order to use this function, MBC[2:0] needs to be programmed to be B'001. When a transmission is performed by this function, the DLC (Data Length Code) to be used is the one that has been received. Application needs to guarantee that the DLC of the remote frame correspond to the DLC of the data frame requested.

Important: When ATX is used and MBC = B'001 the filter for the IDE bit cannot be used since ID of remote frame has to be exactly the same as that of data frame as the reply message.

RENESAS

Important: Please note that, when this function is used, the RTR bit will never be set despite receiving a Remote Frame. When a Remote Frame is received, the CPU will be notified by the corresponding RFPR set, however, as RCAN-ET needs to transmit the current message as a Data Frame, the RTR bit remains unchanged.

Important: Please note that in case of overrun condition (UMSR flag set when the Mailbox has its NMC = 0), the message received is discarded. In case a remote frame is causing overrun into a Mailbox configured with ATX = 1, a request to automatically transmit the corresponding message may be accepted.

ATX	Description
0	Automatic Transmission of Data Frame disabled (Initial value)
1	Automatic Transmission of Data Frame enabled

DART (Disable Automatic Re-Transmission): When this bit is set, it disables the automatic re-transmission of a message in the event of an error on the CAN bus or an arbitration lost on the CAN bus. In effect, when this function is used, the corresponding TXCR bit is automatically set at the start of transmission. When this bit is set to '0', RCAN-ET tries to transmit the message as many times as required until it is successfully transmitted or it is cancelled by the TXCR.

DART	Description
0	Re-transmission enabled (Initial value)
1	Re-Transmission disabled

MBC[2:0] (Mailbox Configuration): These bits configure the nature of each Mailbox as in table 14.4. When MBC = B'111, the Mailbox is inactive, i.e., it does not receive or transmit a message regardless of TXPR or other settings. The MBC = '110', '101' and '100' settings are prohibited. When the MBC is set to any other value, the LAFM field becomes available. Please don't set TXPR when MBC is set as reception. There is no hardware protection, and TXPR remains set. MBC[1] of Mailbox-0 is fixed to "1" by hardware.

Table 14.4   Mailbox Function Setting

MBC[2]	MBC[1]	MBC[0]	Data Frame Transmit	Remote Frame Transmit	Data Frame Receive	Remote Frame Receive	Remarks
0	0	0	Yes	Yes	No	No	• Not allowed for Mailbox-0
0	0	1	Yes	Yes	No	Yes	• Can be used with ATX∗ • Not allowed for Mailbox-0 • LAFM can be used
0	1	0	No	No	Yes	Yes	• Allowed for Mailbox-0 • LAFM can be used
0	1	1	No	No	Yes	No	• Allowed for Mailbox-0 • LAFM can be used
1	0	0	Setting prohibited				
1	0	1	Setting prohibited				
1	1	0	Setting prohibited				
1	1	1	Mailbox inactive (Initial value)				

Note: ∗   In order to support automatic retransmission, RTR shall be "0" when MBC = 001(bin) and ATX=1.
When using ATX with the setting 1, the filter for IDE must not be used.

DLC[3:0] (Data Length Code): These bits encode the number of data bytes from 0,1, 2, …, 8 that will be transmitted in a data frame. Please note that when a remote frame request is transmitted the DLC value to be used must be the same as the DLC of the data frame that is requested.

DLC[3]	DLC[2]	DLC[1]	DLC[0]	Description
0	0	0	0	Data Length = 0 bytes (Initial value)
0	0	0	1	Data Length = 1 byte
0	0	1	0	Data Length = 2 bytes
0	0	1	1	Data Length = 3 bytes
0	1	0	0	Data Length = 4 bytes
0	1	0	1	Data Length = 5 bytes
0	1	1	0	Data Length = 6 bytes
0	1	1	1	Data Length = 7 bytes
1	x	x	x	Data Length = 8 bytes

RENESAS

14.3.3 Local Acceptance Filter Mask (LAFM)

This area is used as Local Acceptance Filter Mask (LAFM) for receive boxes.

LAFM: When MBC is set to 001, 010, 011 (Bin), this field is used as LAFM Field. LAFM allows a Mailbox to accept more than one identifier. The LAFM is comprised of two 16-bit read/write areas as in figure 14.4.

Register Name	Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Acces Size	Feld Name
MB[N].LAFMH	H'104 + N*32	IDE_LAFM	0	0			STDID_LAFM[10:0]								EXTID_LAFM[17:16]			Word/LW	LAFM Field
MB[N].LAFML	H'106 + N*32						EXTID_LAFM[15:0]											Word	

Note: N = 0 to 15 (Mailbox number)

Figure 14.4   Acceptance Filter

If a bit is set in the LAFM, then the corresponding bit of a received CAN identifier is ignored when the RCAN-ET searches a Mailbox with the matching CAN identifier. If the bit is cleared, then the corresponding bit of a received CAN identifier must match to the STDID/IDE/EXTID set in the mailbox to be stored. The structure of the LAFM is same as the message control in a Mailbox. If this function is not required, it must be filled with '0'.

Important: RCAN-ET starts to find a matching identifier from Mailbox-15 down to Mailbox-0. As soon as RCAN-ET finds one matching, it stops the search. The message will be stored or not depending on the NMC and RXPR/RFPR flags. This means that, even using LAFM, a received message can only be stored into 1 Mailbox.

Important: When a message is received and a matching Mailbox is found, the whole message is stored into the Mailbox. This means that, if the LAFM is used, the STDID, RTR, IDE and EXTID may differ to the ones originally set as they are updated with the STDID, RTR, IDE, and EXTID of the received message.

STDID_LAFM[10:0] — Filter mask bits for the CAN base identifier [10:0] bits.

STDID_LAFM[10:0]	Description
0	Corresponding STD_ID bit is cared
1	Corresponding STD_ID bit is "don't cared"

EXTID_LAFM[17:0] — Filter mask bits for the CAN Extended identifier [17:0] bits.

EXTID_LAFM[17:0]	Description
0	Corresponding EXT_ID bit is cared
1	Corresponding EXT_ID bit is "don't cared"

IDE_LAFM — Filter mask bit for the CAN IDE bit.

IDE_LAFM	Description
0	Corresponding IDE_ID bit is cared
1	Corresponding IDE_ID bit is "don't cared"

14.3.4   Message Data Fields

Storage for the CAN message data that is transmitted or received. MSG_DATA[0] corresponds to the first data byte that is transmitted or received. The bit order on the CAN bus is bit 7 through to bit 0.

14.4   RCAN-ET Control Registers

The following sections describe RCAN-ET control registers. The address is mapped as in table 14.5.

Important: These registers can only be accessed in Word size (16-bit).

Table 14.5   RCAN-ET Control Registers Configuration

Description	Address	Name	Access Size (bits)
Master Control Register	000	MCR	Word
General Status Register	002	GSR	Word
Baud Rate Configuration Register 1	004	BCR1	Word
Baud Rate Configuration Register 0	006	BCR0	Word
Interrupt Register	008	IRR	Word
Interrupt Register	00A	IMR	Word
Error Counter Register	00C	TEC/REC	Word

14.4.1   Master Control Register (MCR)

The MCR is a 16-bit read/write register that controls RCAN-ET.

- MCR (Address = H'000)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MCR15	MCR14	—	—	—		TST[2:0]		MCR7	MCR6	MCR5	—	—	MCR2	MCR1	MCR0
Initial value:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
R/W:	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Bit 15—ID Reorder (MCR15): This bit changes the order of STDID, RTR, IDE, and EXTID of both message control and LAFM.

Bit 15 : MCR15	Description
0	RCAN-ET is the same as HCAN2
1	RCAN-ET is not the same as HCAN2 (Initial value)

MCR15 (ID Reorder) = 0

Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Access Size	Feld Name
H'100 + N∗32	0	STDID[10:0]											RTR	IDE	EXTID[17:16]		Word/LW	Control 0
H'102 + N∗32	EXTID[15:0]																Word	
H'104 + N∗32	0	STDID_LAFM[10:0]											0	IDE_LAFM	EXTID_LAFM [17:16]		Word/LW	LAFM Field
H'106 + N∗32	EXTID_LAFM[15:0]																Word	

MCR15 (ID Reorder) = 1

Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Access Size	Feld Name
H'100 + N∗32	IDE	RTR	0	STDID[10:0]											EXTID[17:16]		Word/LW	Control 0
H'102 + N∗32	EXTID[15:0]																Word	
H'104 + N∗32	IDE_LAFM	0	0	STDID_LAFM[10:0]											EXTID_LAFM [17:16]		Word/LW	LAFM Field
H'106 + N∗32	EXTID_LAFM[15:0]																Word	

Note: N = 0 to 15 (Mailbox number)

Figure 14.5   ID Reorder

This bit can be modified only in reset mode.

Bit 14—Auto Halt Bus Off (MCR14): If both this bit and MCR6 are set, MCR1 is automatically set as soon as RCAN-ET enters Bus Off.

Bit 14 : MCR14	Description
0	RCAN-ET remains in Bus Off for normal recovery sequence (128 ×11 Recessive Bits) (Initial value)
1	RCAN-ET moves directly into Halt Mode after it enters Bus Off if MCR6 is set.

This bit can be modified only in reset mode.

Bit 13—Reserved: The written value should always be '0' and the returned value is '0'.

Bit 12—Reserved: The written value should always be '0' and the returned value is '0'.

Bit 11—Reserved: The written value should always be '0' and the returned value is '0'.

RENESAS

Bits 10 to 8—Test Mode (TST[2:0]): These bits enable/disable the test modes. Please note that before activating the Test Mode it is requested to move RCAN-ET into Halt mode or Reset mode. This is to avoid that the transition to Test Mode could affect a transmission/reception in progress. For details, please refer to section 14.6.2, Test Mode Settings.

Please note that the test modes are allowed only for diagnosis and tests and not when RCAN-ET is used in normal operation.

Bit 10: TST2	Bit 9: TST1	Bit 8: TST0	Description
0	0	0	Normal Mode (initial value)
0	0	1	Listen-Only Mode (Receive-Only Mode)
0	1	0	Self Test Mode 1 (External)
0	1	1	Self Test Mode 2 (Internal)
1	0	0	Write Error Counter
1	0	1	Error Passive Mode
1	1	0	Setting prohibited
1	1	1	Setting prohibited

Bit 7—Auto-Wake Mode (MCR7): MCR7 enables or disables the Auto-wake mode. If this bit is set, the RCAN-ET automatically cancels the sleep mode (MCR5) by detecting CAN bus activity (dominant bit). If MCR7 is cleared the RCAN-ET does not automatically cancel the sleep mode.

RCAN-ET cannot store the message that wakes it up.

Note: MCR7 cannot be modified while in sleep mode.

Bit 7 : MCR7	Description
0	Auto-wake by CAN bus activity disabled (Initial value)
1	Auto-wake by CAN bus activity enabled

Bit 6—Halt during Bus Off (MCR6): MCR6 enables or disables entering Halt mode immediately when MCR1 is set during Bus Off. This bit can be modified only in Reset or Halt mode. Please note that when Halt is entered in Bus Off the CAN engine is also recovering immediately to Error Active mode.

Bit 6 : MCR6	Description
0	Don't enter Halt mode even if MCR1 is set during Bus Off, but wait up to end of recovery sequence (Initial value)
1	Enter Halt mode immediately during Bus Off if MCR[1] or MCR[14] are asserted.

Bit 5—Sleep Mode (MCR5): Enables or disables Sleep mode transition. If this bit is set, while RCAN-ET is in halt mode, the transition to sleep mode is enabled. Setting MCR5 is allowed after entering Halt mode. The two Error Counters (REC, TEC) will remain the same during Sleep mode. This mode will be exited in two ways:

1. by writing a '0' to this bit position,
2. or, if MCR[7] is enabled, after detecting a dominant bit on the CAN bus.

If Auto wake up mode is disabled, RCAN-ET will ignore all CAN bus activities until the sleep mode is terminated. When leaving this mode the RCAN-ET will synchronize to the CAN bus (by checking for 11 recessive bits) before joining CAN Bus activity. This means that, when the No.2 method is used, RCAN-ET will miss the first message to receive. CAN transceivers stand-by mode will also be unable to cope with the first message when exiting stand by mode, and the software needs to be designed in this manner.

In sleep mode only the following registers can be accessed: MCR, GSR, IRR, and IMR.

Important: RCAN-ET is required to be in Halt mode before requesting to enter in Sleep mode. That allows the CPU to clear all pending interrupts before entering sleep mode. Once all interrupts are cleared RCAN-ET must leave the Halt mode and enter Sleep mode simultaneously (by writing MCR[5] = 1 and MCR[1] = 0 at the same time).

Bit 5 : MCR5	Description
0	RCAN-ET sleep mode released (Initial value)
1	Transition to RCAN-ET sleep mode enabled

RENESAS

Bit 4—Reserved: The written value should always be '0' and the returned value is '0'.

Bit 3—Reserved: The written value should always be '0' and the returned value is '0'.

Bit 2—Message Transmission Priority (MCR2): MCR2 selects the order of transmission for pending transmit data. If this bit is set, pending transmit data are sent in order of the bit position in the Transmission Pending Register (TXPR).

The order of transmission starts from Mailbox-15 as the highest priority, and then down to Mailbox-1 (if those mailboxes are configured for transmission).

If MCR2 is cleared, all messages for transmission are queued with respect to their priority (by running internal arbitration). The highest priority message has the Arbitration Field (STDID + IDE bit + EXTID (if IDE = 1) + RTR bit) with the lowest digital value and is transmitted first. The internal arbitration includes the RTR bit and the IDE bit (internal arbitration works in the same way as the arbitration on the CAN Bus between two CAN nodes starting transmission at the same time).

This bit can be modified only in Reset or Halt mode.

Bit 2 : MCR2	Description
0	Transmission order determined by message identifier priority (Initial value)
1	Transmission order determined by mailbox number priority (Mailbox-15 $\rightarrow$ Mailbox-1)

Bit 1—Halt Request (MCR1): Setting the MCR1 bit causes the CAN controller to complete its current operation and then enter Halt mode (where it is cut off from the CAN bus). The RCAN-ET remains in Halt Mode until the MCR1 is cleared.

During the Halt mode, the CAN Interface does not join the CAN bus activity and does not store messages or transmit messages. All the user registers (including Mailbox contents and TEC/REC) remain unchanged with the exception of IRR0 and GSR4 which are used to notify the halt status itself.

If the CAN bus is in idle or intermission state regardless of MCR6, RCAN-ET will enter Halt Mode within one Bit Time. If MCR6 is set, a halt request during Bus Off will be also processed within one Bit Time. Otherwise the full Bus Off recovery sequence will be performed beforehand. Entering the Halt Mode can be notified by IRR0 and GSR4.

If both MCR14 and MCR6 are set, MCR1 is automatically set as soon as RCAN-ET enters Bus Off.

In the Halt mode, the RCAN-ET setting can be modified with the exception of the Bit Timing setting, as it does not join the bus activity.

MCR[1] has to be cleared by writing a '0' in order to re-join the CAN bus. After this bit has been cleared, RCAN-ET waits until it detects 11 recessive bits, and then joins the CAN bus.

Note: After issuing a Halt request the CPU is not allowed to set TXPR or TXCR or clear MCR1 until the transition to Halt mode is completed (notified by IRR0 and GSR4). After MCR1 is set this can be cleared only after entering Halt mode or through a reset operation (software or hardware).

Note: Transition into or recovery from HALT mode, is only possible if the BCR1 and BCR0 registers are configured to a proper Baud Rate.

Bit 1 : MCR1	Description
0	Halt mode request clear
1	Halt mode transition request

Bit 0—Reset Request (MCR0): Controls resetting of the RCAN-ET module. When this bit is changed from '0' to '1' the RCAN-ET controller enters its reset routine, re-initializing the internal logic, which then sets GSR3 and IRR0 to notify the reset mode. All user registers are initialised.

RCAN-ET can be reset while this bit is set (configuration mode). This bit has to be cleared by writing a '0' to join the CAN bus. After this bit is cleared, the RCAN-ET waits until it detects 11 recessive bits, and then joins the CAN bus.

The Baud Rate needs to be set up to a proper value in order to sample the value on the CAN Bus.

After Power On Reset, this bit and GSR3 are always set. This means that a reset request has been made and RCAN-ET needs to be set.

The Reset Request is equivalent to a Power On Reset but controlled by Software.

Bit 0 : MCR0	Description
0	Reset mode request clear
1	CAN Interface reset mode transition request (Initial value)

RENESAS

14.4.2   General Status Register (GSR)

The GSR is a 16-bit read-only register that indicates the status of RCAN-ET.

- GSR (Address = H'002)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	GSR5	GSR4	GSR3	GSR2	GSR1	GSR0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bits 15 to 6—Reserved: The written value should always be '0' and the returned value is '0'.

Bit 5—Error Passive Status Bit (GSR5): Indicates whether the CAN Interface is in Error Passive or not. This bit will be set high as soon as the RCAN-ET enters the Error Passive state and is cleared when the module enters again the Error Active state (this means the GSR5 will stay high during Error Passive and during Bus Off). Consequently to find out the correct state both GSR5 and GSR0 must be considered.

Bit 5 : GSR5	Description
0	RCAN-ET is not in Error Passive or in Bus Off status (Initial value)
	[Reset condition] RCAN-ET is in Error Active state
1	RCAN-ET is in Error Passive (if GSR0 = 0) or Bus Off (if GSR0 = 1)
	[Setting condition] When TEC $\geq$128 or REC $\geq$128 or error passive test mode is selected

Bit 4—Halt/Sleep Status Bit (GSR4): Indicates whether the CAN engine is in the halt/sleep state or not. Please note that the clearing time of this flag is not the same as the setting time of IRR12.

Please note that this flag reflects the status of the CAN engine and not of the full RCAN-ET IP. RCAN-ET exits sleep mode and can be accessed once MCR5 is cleared. The CAN engine exits sleep mode only after two additional transmission clocks on the CAN Bus.

Bit 4 : GSR4	Description
0	RCAN-ET is not in the Halt state or Sleep state (Initial value)
1	Halt mode (if MCR1 = 1) or Sleep mode (if MCR5 = 1)
	[Setting condition] If MCR1 is set and the CAN bus is either in intermission or idle or MCR5 is set and RCAN-ET is in the halt mode or RCAN-ET is moving to Bus Off when MCR14 and MCR6 are both set

Bit 3—Reset Status Bit (GSR3): Indicates whether the RCAN-ET is in the reset state or not.

Bit 3 : GSR3	Description
0	RCAN-ET is not in the reset state
1	Reset state (Initial value)
	[Setting condition] After an RCAN-ET internal reset (due to software or hardware reset)

Bit 2—Message Transmission in progress Flag (GSR2): Flag that indicates to the CPU if the RCAN-ET is in Bus Off or transmitting a message or an error/overload flag due to error detected during transmission. The timing to set TXACK is different from the time to clear GSR2. TXACK is set at the 7th bit of End Of Frame. GSR2 is set at the 3rd bit of intermission if there are no more messages ready to be transmitted. It is also set by arbitration lost, bus idle, reception, reset, or halt transition.

Bit 2 : GSR2	Description
0	RCAN-ET is in Bus Off or a transmission is in progress
1	[Setting condition] Not in Bus Off and no transmission in progress (Initial value)

RENESAS

Bit 1—Transmit/Receive Warning Flag (GSR1): Flag that indicates an error warning.

Bit 1 : GSR1	Description
0	[Reset condition] When (TEC <96 and REC <96) or Bus Off (Initial value)
1	[Setting condition] When 96 $\leq$ TEC <256 or 96 $\leq$ REC <256

Note: REC is incremented during Bus Off to count the recurrences of 11 recessive bits as requested by the Bus Off recovery sequence. However the flag GSR1 is not set in Bus Off.

Bit 0—Bus Off Flag (GSR0): Flag that indicates that RCAN-ET is in the bus off state.

Bit 0 : GSR0	Description
0	[Reset condition] Recovery from bus off state or after a hardware or software reset (Initial value)
1	[Setting condition] When TEC $\geq$ 256 (bus off state)

Note: Only the lower 8 bits of TEC are accessible from the user interface. The 9th bit is equivalent to GSR0.

14.4.3   Bit Configuration Registers 0 and 1 (BCR0 and BCR1)

The BCR0 and BCR1 are 2×16-bit read/write register that are used to set CAN bit timing parameters and the baud rate pre-scaler for the CAN Interface.

The Time quanta is defined as:

$$\text{Timequanta} = \frac{2 * \text{BRP}}{f_{clk}}$$

Where: BRP (Baud Rate Pre-scaler) is the value stored in BCR0 incremented by 1 and fclk is the used peripheral bus frequency.

- BCR1 (Address = H'004)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSG1[3:0]				—	TSG2[2:0]			—	—	SJW[1:0]		—	—	—	BSP
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R	R	R/W	R/W	R	R	R	R/W

Please refer to the table below for TSG1 and TSG2 setting.

Bits 15 to 12—Time Segment 1 (TSG1[3:0] = BCR1[15:12]): These bits are used to set the segment TSEG1 (= PRSEG + PHSEG1) to compensate for edges on the CAN Bus with a positive phase error. A value from 4 to 16 time quanta can be set.

Bit 15: TSG1[3]	Bit 14: TSG1[2]	Bit 13: TSG1[1]	Bit 12: TSG1[0]	Description
0	0	0	0	Setting prohibited (Initial value)
0	0	0	1	Setting prohibited
0	0	1	0	Setting prohibited
0	0	1	1	PRSEG + PHSEG1 = 4 time quanta
0	1	0	0	PRSEG + PHSEG1 = 5 time quanta
:	:	:	:	:
:	:	:	:	:
1	1	1	1	PRSEG + PHSEG1 = 16 time quanta

RENESAS

Bit 11—Reserved: The written value should always be '0' and the returned value is '0'.

Bits 10 to 8—Time Segment 2 (TSG2[2:0] = BCR1[10:8]): These bits are used to set the segment TSEG2 (= PHSEG2) to compensate for edges on the CAN Bus with a negative phase error. A value from 2 to 8 time quanta can be set as shown below.

Bit 10: TSG2[2]	Bit 9: TSG2[1]	Bit 8: TSG2[0]	Description
0	0	0	Setting prohibited (Initial value)
0	0	1	PHSEG2 = 2 time quanta (conditionally prohibited) See table 14.6 for TSG1 and TSG2 setting.
0	1	0	PHSEG2 = 3 time quanta
0	1	1	PHSEG2 = 4 time quanta
1	0	0	PHSEG2 = 5 time quanta
1	0	1	PHSEG2 = 6 time quanta
1	1	0	PHSEG2 = 7 time quanta
1	1	1	PHSEG2 = 8 time quanta

Bits 7 and 6—Reserved: The written value should always be '0' and the returned value is '0'.

Bits 5 and 4—ReSynchronisation Jump Width (SJW[1:0] = BCR0[5:4]): These bits set the synchronization jump width.

Bit 5: SJW[1]	Bit 4: SJW[0]	Description
0	0	Synchronization Jump width = 1 time quantum (Initial value)
0	1	Synchronization Jump width = 2 time quanta
1	0	Synchronization Jump width = 3 time quanta
1	1	Synchronization Jump width = 4 time quanta

RENESAS

Bits 3 to 1—Reserved: The written value should always be '0' and the returned value is '0'.

Bit 0—Bit Sample Point (BSP = BCR1[0]): Sets the point at which data is sampled.

Bit 0 : BSP	Description
0	Bit sampling at one point (end of time segment 1) (Initial value)
1	Bit sampling at three points (rising edge of the last three clock cycles of PHSEG1)

- BCR0 (Address = H'006)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—				BRP[7:0]				
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 8 to 15—Reserved: The written value should always be '0' and the returned value is '0'.

Bits 7 to 0—Baud Rate Pre-scale (BRP[7:0] = BCR0 [7:0]): These bits are used to define the peripheral bus clock periods contained in a Time Quantum.

Bit 7: BRP[7]	Bit 6: BRP[6]	Bit 5: BRP[5]	Bit 4: BRP[4]	Bit 3: BRP[3]	Bit 2: BRP[2]	Bit 1: BRP[1]	Bit 0: BRP[0]	Description
0	0	0	0	0	0	0	0	2 × peripheral bus clock (Initial value)
0	0	0	0	0	0	0	1	4 × peripheral bus clock
0	0	0	0	0	0	1	0	6 × peripheral bus clock
:	:	:	:	:	:	:	:	2*(register value+1) × peripheral bus clock
:	:	:	:	:	:	:	:	
1	1	1	1	1	1	1	1	512 × peripheral bus clock

- Requirements of Bit Configuration Register
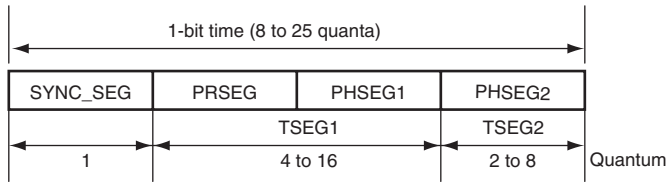
SYNC_SEG: Segment for establishing synchronization of nodes on the CAN bus. (Normal bit edge transitions occur in this segment.)

PRSEG: Segment for compensating for physical delay between networks.

PHSEG1:   Buffer segment for correcting phase drift (positive). (This segment is extended when synchronization (resynchronization) is established.)

PHSEG2:   Buffer segment for correcting phase drift (negative). (This segment is shortened when synchronization (resynchronization) is established)

TSEG1:   TSG1 + 1

TSEG2:   TSG2 + 1

The RCAN-ET Bit Rate Calculation is:

$$\text{Bit Rate} = \frac{f_{clk}}{2 \times (BRP + 1) \times (TSEG1 + TSEG2 + 1)}$$

where BRP is given by $2 \times$ register value $+ 1$, TSEG1 and TSEG2 are derived values from the descriptions of the tables below. The time segment '+ 1' in the above formula is for the Sync-Seg which duration is 1 time quanta.

f_{clk} = Peripheral Clock

BCR Setting Constraints

TSEG1min > TSEG2 $\geq$ SJWmax   (SJW = 1 to 4)

$8 \leq$ TSEG1 + TSEG2 + 1 $\leq$ 25 time quanta (TSEG1 + TSEG2 + 1 = 7 is not allowed)

TSEG2 $\geq$ 2

These constraints allow the setting range shown in table 14.6 for TSEG1 and TSEG2 in the Bit Configuration Register. The number in the table shows possible setting of SJW. "No" shows that there is no allowed combination of TSEG1 and TSEG2.

Table 14.6   TSG and TSEG Setting

		001	010	011	100	101	110	111	TSG2
		2	3	4	5	6	7	8	TSEG2
TSG1	TSEG1								
0011	4	No	1 to 3	No	No	No	No	No	
0100	5	1 to 2	1 to 3	1 to 4	No	No	No	No	
0101	6	1 to 2	1 to 3	1 to 4	1 to 4	No	No	No	
0110	7	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	No	No	
0111	8	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	No	
1000	9	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1001	10	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1010	11	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1011	12	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1100	13	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1101	14	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1110	15	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	
1111	16	1 to 2	1 to 3	1 to 4	1 to 4	1 to 4	1 to 4	1 to 4	

Example: To have a Bit rate of 500 Kbps with a frequency of fclk = 20 MHz it is possible to set: BRP = 1, TSEG1 = 6, TSEG2 = 3.

Then the configuration to write is BCR1 = 5200 and BCR0 = 0001.

14.4.4   Interrupt Request Register (IRR)

The IRR is a 16-bit read/write-clearable register containing status flags for the various interrupt sources.

- IRR (Address = H'008)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	IRR13	IRR12	—	—	IRR9	IRR8	IRR7	IRR6	IRR5	IRR4	IRR3	IRR2	IRR1	IRR0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
R/W:	R	R	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R/W

Bits 15 and 14—Reserved

Bit 13—Message Error Interrupt (IRR13): This interrupt indicates that:

- A message error has occurred when in test mode.
- Note: If a Message Overload condition occurs when in Test Mode, then this bit will not be set. When not in test mode this interrupt is inactive.

Bit 13: IRR13	Description
0	message error has not occurred in test mode (Initial value)
	[Clearing condition] Writing 1
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	[Setting condition] message error has occurred in test mode

Bit 12—Bus Activity while in Sleep Mode (IRR12): IRR12 indicates that a CAN bus activity is present. While the RCAN-ET is in sleep mode and a dominant bit is detected on the CAN bus, this bit is set. This interrupt is cleared by writing a '1' to this bit position. Writing a '0' has no effect. If auto wakeup is not used and this interrupt is not requested it needs to be disabled by the related interrupt mask register. If auto wake up is not used and this interrupt is requested it should be cleared only after recovering from sleep mode. This is to avoid that a new falling edge of the reception line causes the interrupt to get set again.

Please note that the setting time of this interrupt is different from the clearing time of GSR4.

Bit 12: IRR12	Description
0	Bus idle state (Initial value)
	[Clearing condition] Writing 1
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	CAN bus activity is detected in CAN sleep mode
	[Setting condition] dominant bit level detection on the Rx line while in sleep mode

Bits 11 and 10—Reserved

Bit 9—Message Overrun/Overwrite Interrupt Flag (IRR9): Flag indicating that a message has been received but the existing message in the matching Mailbox has not been read as the corresponding RXPR or RFPR is already set to '1' and not yet cleared by the CPU, thus the received message is either abandoned (overrun) or overwritten dependant upon the NMC (New Message Control) bit. This bit is cleared when all bit in UMSR (Unread Message Status Register) are cleared (by writing '1') or by setting MBIMR (MailBox interrupt Mast Register) for all UMSR flag set. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit position has no effect.

Bit 9: IRR9	Description
0	No pending notification of message overrun/overwrite
	[Clearing condition] Clearing of all bit in UMSR/setting MBIMR for all UMSR set (initial value)
1	A receive message has been discarded due to overrun condition or a message has been overwritten
	[Setting condition] Message is received while the corresponding RXPR and/or RFPR =1 and MBIMR =0

Bit 8—Mailbox Empty Interrupt Flag (IRR8): This bit is set when one of the messages set for transmission has been successfully sent (corresponding TXACK flag is set) or has been successfully aborted (ABACK flag corresponding to the message whose transmission cancel has been executed is set). The related TXPR is also cleared and this mailbox is now ready to accept a new message data for the next transmission. In effect, this bit is set by an OR'ed signal of the TXACK and ABACK bits not masked by the corresponding MBIMR flag. Therefore, this bit is automatically cleared when all the TXACK and ABACK bits are cleared. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit position has no effect.

Bit 8: IRR8	Description
0	Messages set for transmission or transmission cancellation request NOT progressed. (Initial value)
	[Clearing condition] All the TXACK and ABACK bits are cleared/setting MBIMR for all TXACK and ABACK set
1	Message has been transmitted or aborted (cancelled), and new message can be stored
	[Setting condition]
	When a TXACK or ABACK bit is set (if MBIMR = 0).

Bit 7—Overload Frame (IRR7): Flag indicating that the RCAN-ET has detected the transmission of an overload frame. IRR7 is cleared by writing a '1' to this bit position. Writing a '0' has no effect.

Bit 7: IRR7	Description
0	[Clearing condition] Writing 1 (Initial value)
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	[Setting condition] Overload condition detected

Bit 6—Bus Off Interrupt Flag (IRR6): This bit is set when RCAN-ET enters the Bus-off state or when RCAN-ET leaves Bus-off and returns to Error-Active. The cause therefore is the existing condition TEC ≥ 256 at the node or the end of the Bus-off recovery sequence (128×11 consecutive recessive bits) or the transition from Bus Off to Halt (automatic or manual). This bit remains set even if the RCAN-ET node leaves the bus-off condition, and needs to be explicitly cleared by software. The software is expected to read the GSR0 to judge whether RCAN-ET is in the bus-off or error active status. It is cleared by writing a '1' to this bit position even if the node is still bus-off. Writing a '0' has no effect.

Bit 6: IRR6	Description
0	[Clearing condition] Writing 1 (Initial value)
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Enter Bus off state caused by transmit error or Error Active state returning from Bus-off
	[Setting condition] When TEC becomes ≥256 or End of Bus-off after 128×11 consecutive recessive bits or transition from Bus Off to Halt

Bit 5—Error Passive Interrupt Flag (IRR5): Interrupt flag indicating the error passive state caused by the transmit or receive error counter or by Error Passive forced by test mode. This bit is reset by writing a '1' to this bit position, writing a '0' has no effect. If this bit is cleared the node may still be error passive. Please note that the software needs to check GSR0 and GSR5 to judge whether RCAN-ET is in Error Passive or Bus Off status.

Bit 5: IRR5	Description
0	[Clearing condition] Writing 1 (Initial value)
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Error passive state caused by transmit/receive error
	[Setting condition] When TEC ≥128 or REC ≥128 or Error Passive test mode is used

RENESAS

Bit 4—Receive Error Counter Warning Interrupt Flag (IRR4): This bit becomes set if the receive error counter (REC) reaches a value greater than 95 when RCAN-ET is not in the Bus Off status. The interrupt is reset by writing a '1' to this bit position, writing '0' has no effect.

Bit 4: IRR4	Description
0	[Clearing condition] Writing 1 (Initial value)
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Error warning state caused by receive error
	[Setting condition] When REC ≥96 and RCAN-ET is not in Bus Off

Bit 3—Transmit Error Counter Warning Interrupt Flag (IRR3): This bit becomes set if the transmit error counter (TEC) reaches a value greater than 95. The interrupt is reset by writing a '1' to this bit position, writing '0' has no effect.

Bit 3: IRR3	Description
0	[Clearing condition] Writing 1 (Initial value)
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Error warning state caused by transmit error
	[Setting condition] When TEC ≥96

Bit 2—Remote Frame Request Interrupt Flag (IRR2): Flag indicating that a remote frame has been received in a mailbox. This bit is set if at least one receive mailbox, with related MBIMR not set, contains a remote frame transmission request. This bit is automatically cleared when all bits in the Remote Frame Receive Pending Register (RFPR), are cleared. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit has no effect.

Bit 2: IRR2	Description
0	[Clearing condition] Clearing of all bits in RFPR (Initial value)
1	At least one remote request is pending
	[Setting condition] When remote frame is received and the corresponding MBIMR = 0

Bit 1—Data Frame Received Interrupt Flag (IRR1): IRR1 indicates that there are pending Data Frames received. If this bit is set at least one receive mailbox contains a pending message. This bit is cleared when all bits in the Data Frame Receive Pending Register (RXPR) are cleared, i.e. there is no pending message in any receiving mailbox. It is in effect a logical OR of the RXPR flags from each configured receive mailbox with related MBIMR not set. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit has no effect.

Bit 1: IRR1	Description
0	[Clearing condition] Clearing of all bits in RXPR (Initial value)
1	Data frame received and stored in Mailbox
	[Setting condition] When data is received and the corresponding MBIMR = 0

Bit 0—Reset/Halt/Sleep Interrupt Flag (IRR0): This flag can get set for three different reasons. It can indicate that:

1. Reset mode has been entered after a software (MCR0) or hardware reset
2. Halt mode has been entered after a Halt request (MCR1)
3. Sleep mode has been entered after a sleep request (MCR5) has been made while in Halt mode.

The GSR may be read after this bit is set to determine which state RCAN-ET is in.

Important : When a Sleep mode request needs to be made, the Halt mode must be used beforehand. Please refer to the MCR5 description and figure 14.8.

IRR0 is set by the transition from "0" to "1" of GSR3 or GSR4 or by transition from Halt mode to Sleep mode. So, IRR0 is not set if RCAN-ET enters Halt mode again right after exiting from Halt mode, without GSR4 being cleared. Similarly, IRR0 is not set by direct transition from Sleep mode to Halt Request. At the transition from Halt/Sleep mode to Transition/Reception, clearing GSR4 needs (one-bit time - TSEG2) to (one-bit time * 2 - TSEG2).

In the case of Reset mode, IRR0 is set, however, the interrupt to the CPU is not asserted since IMR0 is automatically set by initialization.

RENESAS

Bit 0: IRR0	Description
0	[Clearing condition] Writing 1
	(When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Transition to software reset mode or transition to halt mode or transition to sleep mode (Initial value)
	[Setting condition] When reset/halt/sleep transition is completed after a reset (MCR0 or hardware) or Halt mode (MCR1) or Sleep mode (MCR5) is requested

14.4.5   Interrupt Mask Register (IMR)

The IMR is a 16 bit register that protects all corresponding interrupts in the Interrupt Request Register (IRR) from generating an output signal on the IRQ. An interrupt request is masked if the corresponding bit position is set to '1'. This register can be read or written at any time. The IMR directly controls the generation of IRQ, but does not prevent the setting of the corresponding bit in the IRR.

- IMR (Address = H'00A)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0
Initial value:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 0: Maskable interrupt sources corresponding to IRR[15:0] respectively. When a bit is set, the interrupt signal is not generated, although setting the corresponding IRR bit is still performed.

Bit[15:0]: IMRn	Description
0	Corresponding IRR is not masked (IRQ is generated for interrupt conditions)
1	Corresponding interrupt of IRR is masked (Initial value)

14.4.6   Transmit Error Counter (TEC) and Receive Error Counter (REC)

The TEC and REC are 16-bit read/(write) register that functions as a counter indicating the number of transmit/receive message errors on the CAN Interface. The count value is stipulated in the CAN protocol specification Refs. [1], [2], [3], and [4]. In modes other than Write Error Counter test mode, this register is read only, and can only be modified by the CAN Interface. This register can be cleared by a Reset request (MCR0) or entering to bus off.

In Write Error Counter test mode (i.e. TST[2:0] = B'100), it is possible to write to this register. The same value can only be written to TEC/REC, and the value written into TEC is set to TEC and REC. When writing to this register, RCAN-ET needs to be put into Halt Mode. This feature is only intended for test purposes.

- TEC/REC (Address = H'00C)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*

Note: * It is only possible to write the value in test mode when TST[2:0] in MCR is 3'b100.

REC is incremented during Bus Off to count the recurrences of 11 recessive bits as requested by the Bus Off recovery sequence.

RENESAS

14.5   RCAN-ET Mailbox Registers

The following sections describe RCAN-ET Mailbox registers that control / flag individual Mailboxes. The address is mapped as follows.

Important : LongWord access is carried out as two consecutive Word accesses.

Table 14.7   RCAN-ET Mailbox Registers

Description	Address	Name	Access Size (bits)
Transmit Pending 1	H'020	TXPR1	LW
Transmit Pending 0	H'022	TXPR0	—
	H'024		
	H'026		
	H'028		
Transmit Cancel 0	H'02A	TXCR0	
	H'02C		
	H'02E		
	H'030		
Transmit Acknowledge 0	H'032	TXACK0	Word
	H'034		
	H'036		
	H'038		
Abort Acknowledge 0	H'03A	ABACK0	Word
	H'03C		
	H'03E		
	H'040		
Data Frame Receive Pending 0	H'042	RXPR0	Word
	H'044		
	H'046		
	H'048		
Remote Frame Receive Pending 0	H'04A	RFPR0	Word
	H'04C		
	H'04E		
	H'050		

Description	Address	Name	Access Size (bits)
Mailbox Interrupt Mask Register 0	H'052	MBIMR0	Word
	H'054		
	H'056		
	H'058		
Unread Message Status Register 0	H'05A	UMSR0	Word
	H'05C		
	H'05E		

14.5.1   Transmit Pending Registers 0 and 1 (TXPR0 and TXPR1)

The concatenation of TXPR0 and TXPR1 is a 32-bit register that contains any transmit pending flags for the CAN module. In the case of 16-bit bus interface, Long Word access is carried out as two consecutive word accesses.

<Longword Write Operation>

<upper word write>   <lower word write>

consecutive access

Data is stored into Temp instead of TXPR1.   Lower word data are stored into TXPR0.
TXPR1 is always H'0000.

<Longword Read Operation>

<upper word read>

<lower word read>

16-bit Peripheral bus

16-bit Peripheral bus

always
H'0000

consecutive access

Temp

Temp

TXPR1

TXPR0

TXPR1

TXPR0

H'020

H'022

H'020

H'022

TXPR0 is stored into Temp,
when TXPR1 (= H'0000) is read.

Temp is read instead of TXPR0.
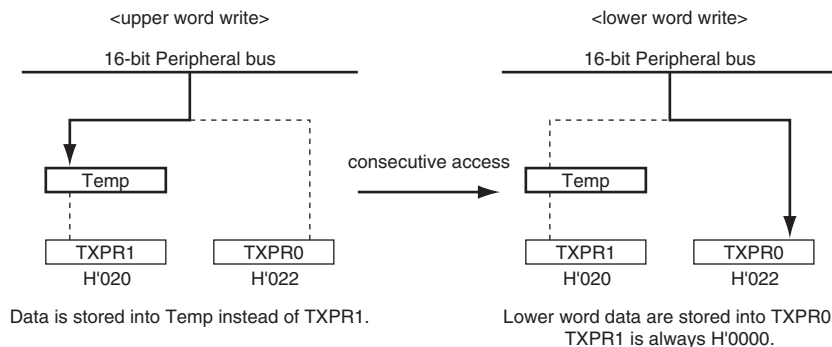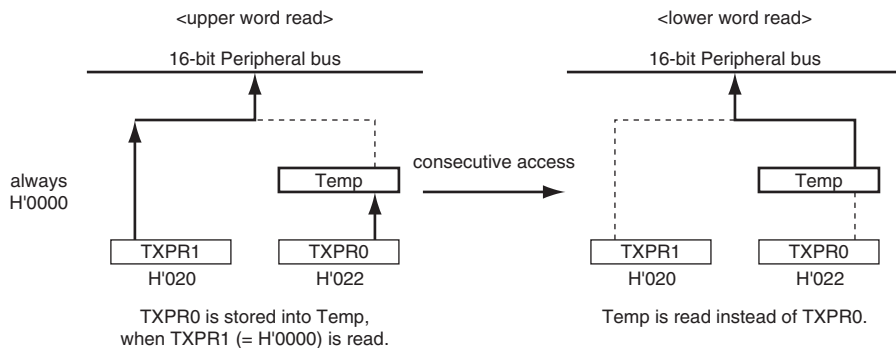
The TXPR1 register cannot be modified and it is always fixed to '0'. The TXPR0 controls Mailbox-15 to Mailbox-1. The CPU may set the TXPR bits to affect any message being considered for transmission by writing a '1' to the corresponding bit location. Writing a '0' has no effect, and TXPR cannot be cleared by writing a '0' and must be cleared by setting the corresponding TXCR bits. TXPR may be read by the CPU to determine which, if any, transmissions are pending or in progress. In effect there is a transmit pending bit for all Mailboxes except for the Mailbox-0. Writing a '1' to a bit location when the mailbox is not configured to transmit is not allowed.

The RCAN-ET will clear a transmit pending flag after successful transmission of its corresponding message or when a transmission abort is requested successfully from the TXCR. The TXPR flag is not cleared if the message is not transmitted due to the CAN node losing the arbitration process or due to errors on the CAN bus, and RCAN-ET automatically tries to transmit it again unless its DART bit (Disable Automatic Re-Transmission) is set in the Message-Control of the corresponding Mailbox. In such case (DART set), the transmission is cleared and notified through Mailbox Empty Interrupt Flag (IRR8) and the correspondent bit within the Abort Acknowledgement Register (ABACK).

If the status of the TXPR changes, the RCAN-ET shall ensure that in the identifier priority scheme (MCR2 = 0), the highest priority message is always presented for transmission in an intelligent way even under circumstances such as bus arbitration losses or errors on the CAN bus. Please refer to section 14.6, Application Note, for details.

RENESAS

When the RCAN-ET changes the state of any TXPR bit position to a '0', an empty slot interrupt (IRR8) may be generated. This indicates that either a successful or an aborted mailbox transmission has just been made. If a message transmission is successful it is signaled in the TXACK register, and if a message transmission abortion is successful it is signaled in the ABACK register. By checking these registers, the contents of the Message of the corresponding Mailbox may be modified to prepare for the next transmission.

- TXPR1

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								TXPR1[15:0]								
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
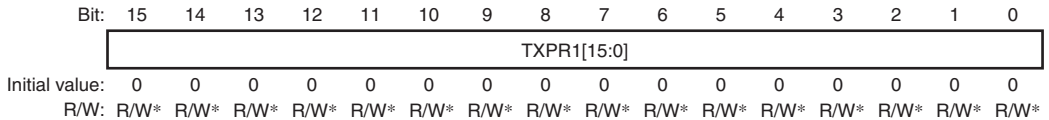
Note : *   Any write operation is ignored.

Read value is always H'0000. Long word access is mandatory when reading or writing TXPR1/TXPR0. Writing any value to TXPR1 is allowed, however, write operation to TXPR1 has no effect.

- TXPR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								TXPR0[15:1]								0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	—
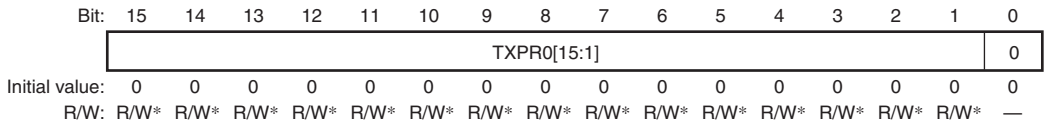
Note : *   It is possible only to write a '1' for a Mailbox set as transmitter. These bits are always read as H'0000. Regarding the read/write of TXPR/TXPR0, be sure to access with longwords. Writing 0 to the bit 0 in TXPR0 is invalid.

Bits 15 to 1: indicate that the corresponding Mailbox is requested to transmit a CAN Frame. The bits 15 to 1 correspond to Mailbox-15 to 1 respectively. When multiple bits are set, the order of the transmissions is governed by the MCR2 – CAN-ID or Mailbox number.

Bit[15:1]:TXPR0	Description
0	Transmit message idle state in corresponding mailbox (Initial value)
	[Clearing condition] Completion of message transmission or message transmission abortion (automatically cleared)
1	Transmission request made for corresponding mailbox

RENESAS

Bit 0—Reserved: This bit is always '0' as this is a receive-only Mailbox. Writing a '1' to this bit position has no effect. The returned value is '0'.

14.5.2   Transmit Cancel Register 0 (TXCR0)

The TXCR0 is a 16-bit read / conditionally-write registers. The TXCR0 controls Mailbox-15 to Mailbox-1.This register is used by the CPU to request the pending transmission requests in the TXPR to be cancelled. To clear the corresponding bit in the TXPR the CPU must write a '1' to the bit position in the TXCR. Writing a '0' has no effect.

When an abort has succeeded the CAN controller clears the corresponding TXPR + TXCR bits, and sets the corresponding ABACK bit. However, once a Mailbox has started a transmission, it cannot be cancelled by this bit. In such a case, if the transmission finishes in success, the CAN controller clears the corresponding TXPR + TXCR bit, and sets the corresponding TXACK bit, however, if the transmission fails due to a bus arbitration loss or an error on the bus, the CAN controller clears the corresponding TXPR + TXCR bit, and sets the corresponding ABACK bit. If an attempt is made by the CPU to clear a mailbox transmission that is not transmit-pending it has no effect. In this case the CPU will be not able at all to set the TXCR flag.

- TXCR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								TXCR0[15:1]								0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	—

Note : *   Only writing a '1' to a Mailbox that is requested for transmission and is configured as transmit.

Bits 15 to 1: request the corresponding Mailbox, that is in the queue for transmission, to cancel its transmission. The bits 15 to 1 correspond to Mailbox-15 to 1 (and TXPR0[15:1]) respectively.

Bit[15:1]:TXCR0	Description
0	Transmit message cancellation idle state in corresponding mailbox (Initial value)
	[Clearing condition] Completion of transmit message cancellation (automatically cleared)
1	Transmission cancellation request made for corresponding mailbox

Bit 0: This bit is always '0' as this is a receive-only mailbox. Writing a '1' to this bit position has no effect and always read back as a '0'.

RENESAS

14.5.3   Transmit Acknowledge Register 0 (TXACK0)

The TXACK0 is a 16-bit read / conditionally-write registers. This register is used to signal to the CPU that a mailbox transmission has been successfully made. When a transmission has succeeded the RCAN-ET sets the corresponding bit in the TXACK register. The CPU may clear a TXACK bit by writing a '1' to the corresponding bit location. Writing a '0' has no effect.

- TXACK0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXACK0[15:1]															0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	—

Note :   *   Only when writing a '1' to clear.

Bits 15 to 1: notify that the requested transmission of the corresponding Mailbox has been finished successfully. The bits 15 to 1 correspond to Mailbox-15 to 1 respectively.

Bit[15:1]:TXACK0	Description
0	[Clearing condition] Writing '1' (Initial value)
	When clearing this flag by the CPU in the interrupt handling, always read it after writing '1' to it.
1	Corresponding Mailbox has successfully transmitted message (Data or Remote Frame)
	[Setting condition] Completion of message transmission for corresponding mailbox

Bit 0: This bit is always '0' as this is a receive-only mailbox. Writing a '1' to this bit position has no effect and always read back as a '0'.

RENESAS

14.5.4   Abort Acknowledge Register 0 (ABACK0)

The ABACK0 is a 16-bit read / conditionally-write registers. This register is used to signal to the CPU that a mailbox transmission has been aborted as per its request. When an abort has succeeded the RCAN-ET sets the corresponding bit in the ABACK register. The CPU may clear the Abort Acknowledge bit by writing a '1' to the corresponding bit location. Writing a '0' has no effect. An ABACK bit position is set by the RCAN-ET to acknowledge that a TXPR bit has been cleared by the corresponding TXCR bit.

- ABACK0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ABACK0[15:1]								0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	—

Note :   *   Only when writing a '1' to clear.

Bits 15 to 1: notify that the requested transmission cancellation of the corresponding Mailbox has been performed successfully. The bits 15 to 1 correspond to Mailbox-15 to 1 respectively.

| Bit[15:1]:ABACK0	Description
0	[Clearing condition] Writing '1' (Initial value)
	When clearing this flag by the CPU in the interrupt handling, always read it after writing '1' to it.
1	Corresponding Mailbox has cancelled transmission of message (Data or Remote Frame)
	[Setting condition] Completion of transmission cancellation for corresponding mailbox

Bit 0: This bit is always '0' as this is a receive-only mailbox. Writing a '1' to this bit position has no effect and always read back as a '0'.

14.5.5   Data Frame Receive Pending Register 0 (RXPR0)

The RXPR0 is a 16-bit read / conditionally-write registers. The RXPR is a register that contains the received Data Frames pending flags associated with the configured Receive Mailboxes. When a CAN Data Frame is successfully stored in a receive mailbox the corresponding bit is set in the RXPR. The bit may be cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. However, the bit may only be set if the mailbox is configured by its MBC (Mailbox Configuration) to receive Data Frames. When a RXPR bit is set, it also sets IRR1 (Data Frame Received Interrupt Flag) if its MBIMR (Mailbox Interrupt Mask Register) is not set, and the interrupt signal is generated if IMR1 is not set. Please note that these bits are only set by receiving Data Frames and not by receiving Remote frames.

- RXPR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								RXPR0[15:0]								
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
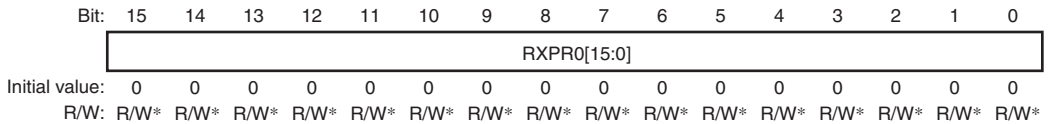
Note : *   Only when writing a '1' to clear.

Bits 15 to 0: Configurable receive mailbox locations corresponding to each mailbox position from 15 to 0 respectively.

| Bit[15:0]: RXPR0	Description
0	[Clearing condition] Writing '1' (Initial value)
	When clearing this flag by the CPU in the interrupt handling, always read it after writing '1' to it.
1	Corresponding Mailbox received a CAN Data Frame
	[Setting condition] Completion of Data Frame receive on corresponding mailbox

RENESAS

14.5.6   Remote Frame Receive Pending Register 0 (RFPR0)

The RFPR0 is a 16-bit read / conditionally-write registers. The RFPR is a register that contains the received Remote Frame pending flags associated with the configured Receive Mailboxes. When a CAN Remote Frame is successfully stored in a receive mailbox the corresponding bit is set in the RFPR. The bit may be cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. In effect there is a bit position for all mailboxes. However, the bit may only be set if the mailbox is configured by its MBC (Mailbox Configuration) to receive Remote Frames. When a RFPR bit is set, it also sets IRR2 (Remote Frame Request Interrupt Flag) if its MBIMR (Mailbox Interrupt Mask Register) is not set, and the interrupt signal is generated if IMR2 is not set. Please note that these bits are only set by receiving Remote Frames and not by receiving Data frames.

- RFPR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RFPR0[15:0]															
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
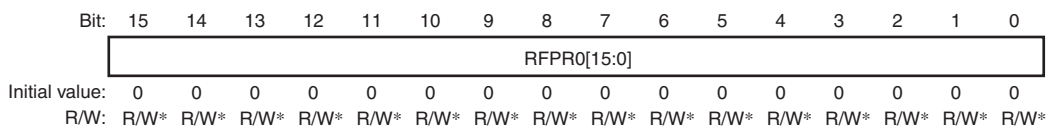
Note :   *   Only when writing a '1' to clear.

Bits 15 to 0: Remote Request pending flags for mailboxes 15 to 0 respectively.

| Bit[15:0]: RFPR0	Description
0	[Clearing condition] Writing '1' (Initial value)
	When clearing this flag by the CPU in the interrupt handling, always read it after writing '1' to it.
1	Corresponding Mailbox received Remote Frame
	[Setting condition] Completion of remote frame receive in corresponding mailbox

14.5.7   Mailbox Interrupt Mask Register 0 (MBIMR0)

The MBIMR0 is a16-bit read/write registers. The MBIMR only prevents the setting of IRR related to the Mailbox activities, that are IRR[1] – Data Frame Received Interrupt, IRR[2] – Remote Frame Request Interrupt, IRR[8] – Mailbox Empty Interrupt, and IRR[9] – Message OverRun/OverWrite Interrupt. If a mailbox is configured as receive, a mask at the corresponding bit position prevents the generation of a receive interrupt (IRR[1] and IRR[2] and IRR[9]) but does not prevent the setting of the corresponding bit in the RXPR or RFPR or UMSR. Similarly when a mailbox has been configured for transmission, a mask prevents the generation of an Interrupt signal and setting of an Mailbox Empty Interrupt due to successful transmission or abortion of transmission (IRR[8]), however, it does not prevent the RCAN-ET from clearing the corresponding TXPR/TXCR bit + setting the TXACK bit for successful transmission, and it does not prevent the RCAN-ET from clearing the corresponding TXPR/TXCR bit + setting the ABACK bit for abortion of the transmission.

A mask is set by writing a '1' to the corresponding bit position for the mailbox activity to be masked. At reset all mailbox interrupts are masked.

- MBIMR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								MBIMR0[15:0]								
Initial value:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
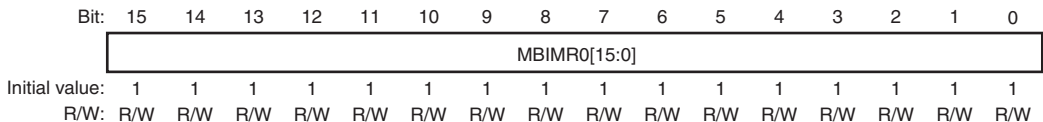
Bits 15 to 0: Enable or disable interrupt requests from individual Mailbox-15 to Mailbox-0 respectively.

Bit[15:0]: MBIMR0	Description
0	Interrupt Request from IRR1/IRR2/IRR8/IRR9 enabled
1	Interrupt Request from IRR1/IRR2/IRR8/IRR9 disabled (initial value)

14.5.8   Unread Message Status Register 0 (UMSR0)

The UMSR0 is a 16-bit read/conditionally write register and it records the mailboxes whose contents have not been accessed by the CPU prior to a new message being received. If the CPU has not cleared the corresponding bit in the RXPR or RFPR when a new message for that mailbox is received, the corresponding UMSR bit is set to '1'. This bit may be cleared by writing a '1' to the corresponding bit location in the UMSR. Writing a '0' has no effect.

If a mailbox is configured as transmit box, the corresponding UMSR will not be set.

- UMSR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								UMSR0[15:0]								
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*	R/W*
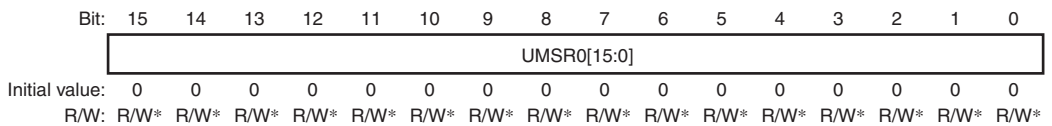
Bits 15 to 0: Indicate that an unread received message has been overwritten or overrun condition has occurred for Mailboxes 15 to 0.

| Bit[15:0]: UMSR0	Description
0	[Clearing condition] Writing '1' (initial value)
	When clearing this flag by the CPU in the interrupt handling, always read it after writing '1' to it.
1	Unread received message is overwritten by a new message or overrun condition
	[Setting condition] When a new message is received before RXPR or RFPR is cleared

14.6   Application Note

14.6.1   Configuration of RCAN-ET

RCAN-ET is considered in configuration mode or after a hardware (Power On Reset)/ software (MCR[0]) reset or when in Halt mode. In both conditions RCAN-ET cannot join the CAN Bus activity and configuration changes have no impact on the traffic on the CAN Bus.

(1)   After a reset request

The following sequence is an example to set the RCAN-ET after (software or hardware) reset. After reset, all the registers are initialised, therefore, RCAN-ET needs to be configured before joining the CAN bus activity. Please read the notes carefully.

RENESAS
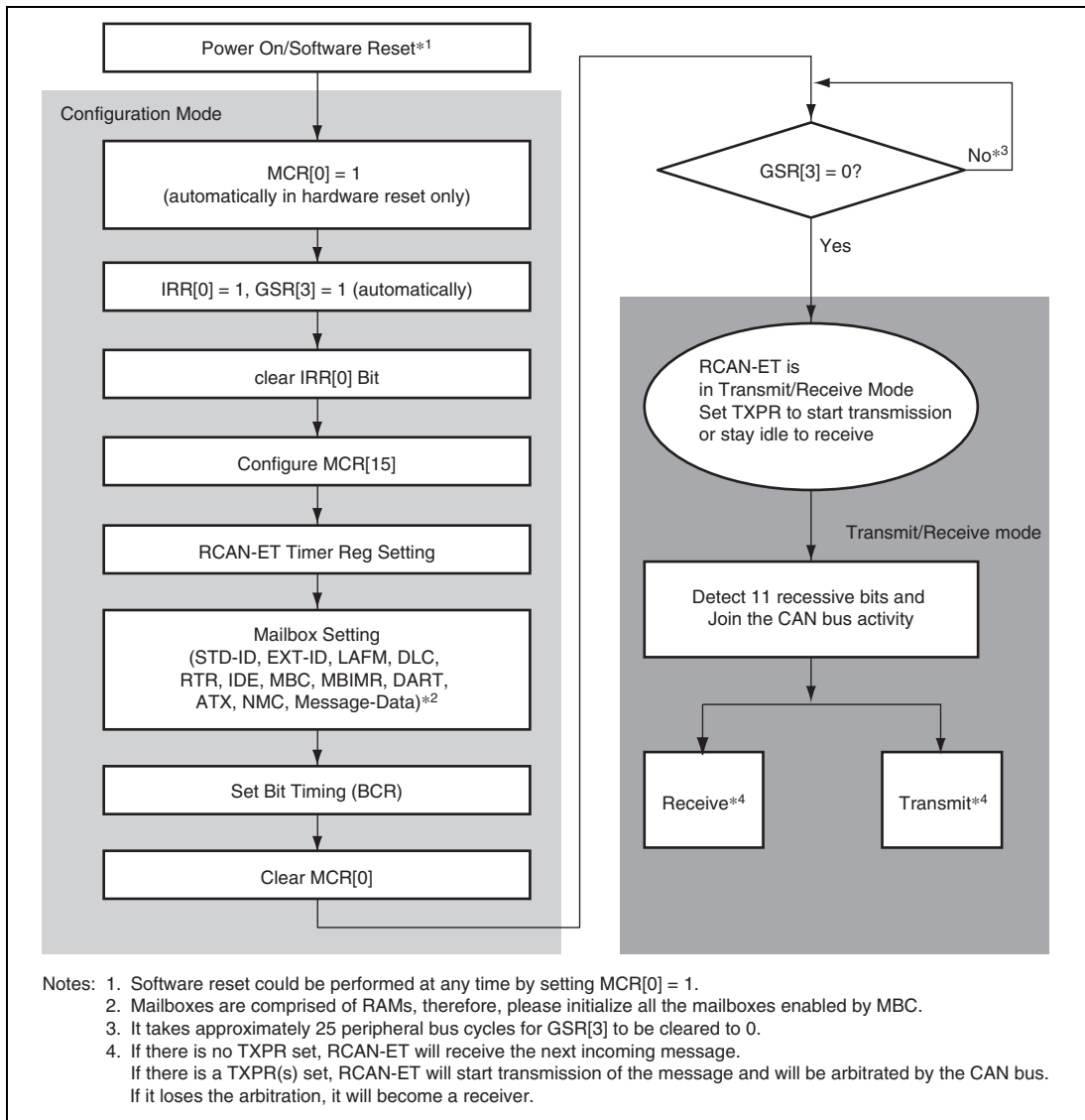
Figure 14.6   Reset Sequence

Notes: 1. Software reset could be performed at any time by setting MCR[0] = 1.
2. Mailboxes are comprised of RAMs, therefore, please initialize all the mailboxes enabled by MBC.
3. It takes approximately 25 peripheral bus cycles for GSR[3] to be cleared to 0.
4. If there is no TXPR set, RCAN-ET will receive the next incoming message.
   If there is a TXPR(s) set, RCAN-ET will start transmission of the message and will be arbitrated by the CAN bus.
   If it loses the arbitration, it will become a receiver.

(2)   Halt mode

When RCAN-ET is in Halt mode, it cannot take part to the CAN bus activity. Consequently the user can modify all the requested registers without influencing existing traffic on the CAN Bus. It is important for this that the user waits for the RCAN-ET to be in halt mode before to modify the requested registers—note that the transition to Halt Mode is not always immediate (transition will occurs when the CAN Bus is idle or in intermission). After RCAN-ET transit to Halt Mode, GSR4 is set.

Once the configuration is completed the Halt request needs to be released. RCAN-ET will join CAN Bus activity after the detection of 11 recessive bits on the CAN Bus.

(3)   Sleep mode

When RCAN-ET is in sleep mode the clock for the main blocks of the IP is stopped in order to reduce power consumption. Only the following user registers are clocked and can be accessed: MCR, GSR, IRR, and IMR. Interrupt related to transmission (TXACK and ABACK) and reception (RXPR and RFPR) cannot be cleared when in sleep mode (as TXACK, ABACK, RXPR and RFPR are not accessible) and must to be cleared beforehand.

The following diagram shows the flow to follow to move RCAN-ET into sleep mode.

RENESAS
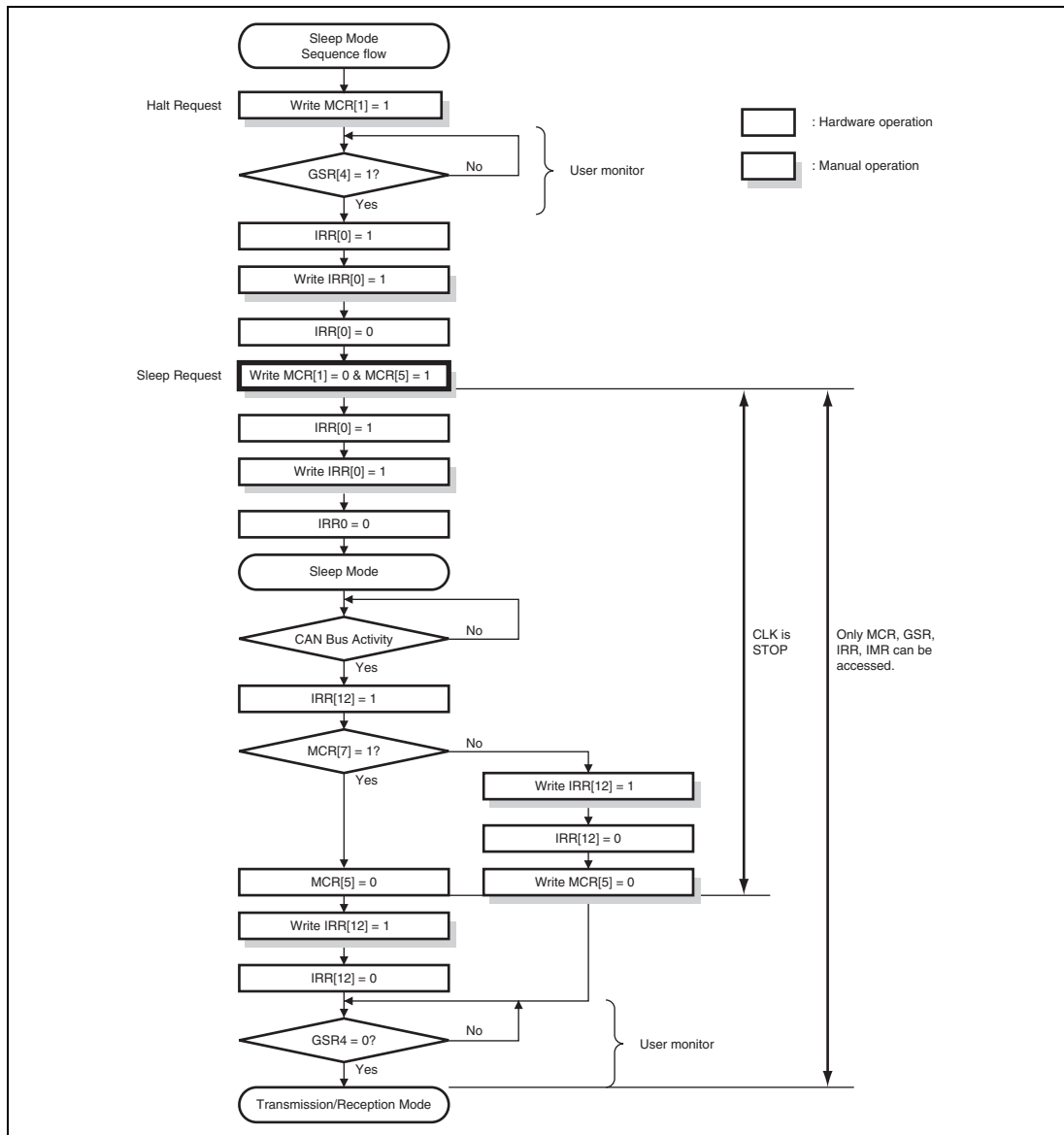
(4)   Can sleep mode

Figure 14.7   Halt Mode/Sleep Mode

Figure 14.8 shows allowed state transition.

- Do not set MCR5 (Sleep Mode) without entering Halt Mode.
- After setting MCR1, make sure that GSR4 is set and the RCAN-ET has entered Halt Mode before clearing MCR1.
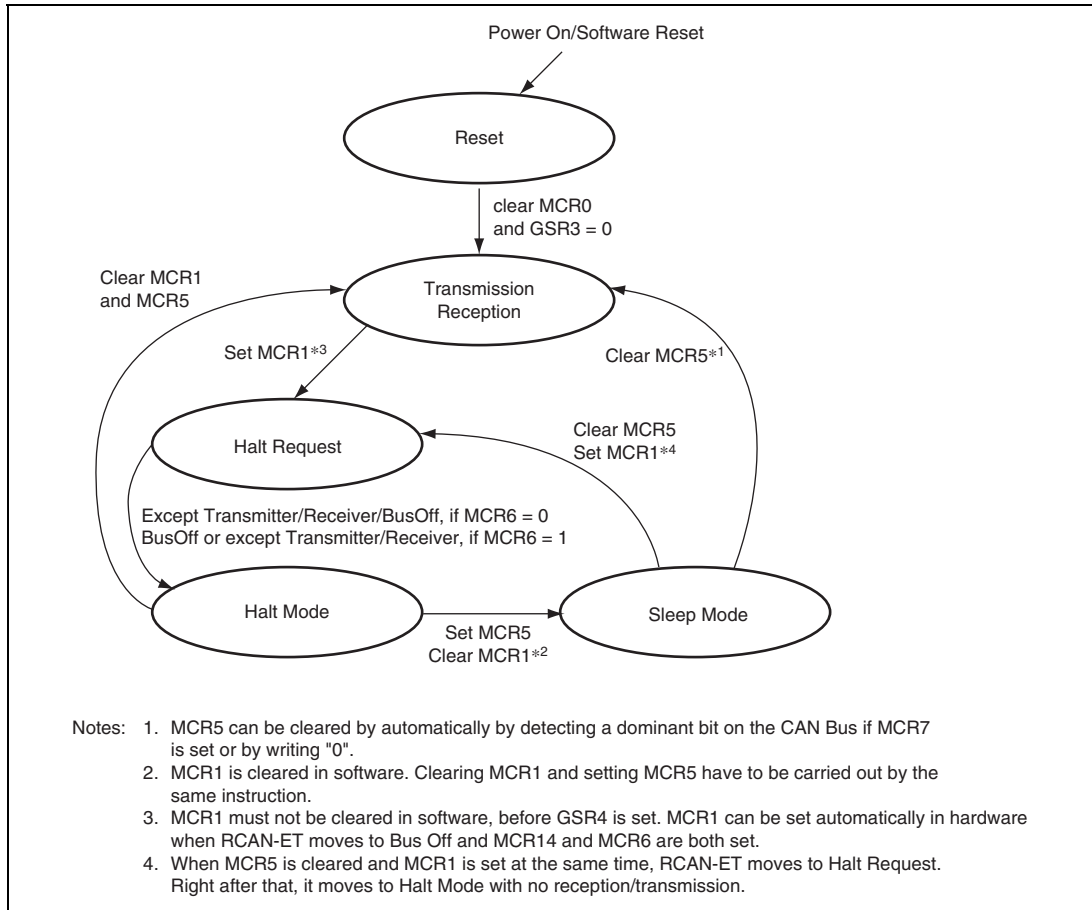
Figure 14.8   Halt Mode/Sleep Mode

Table 14.8 shows conditions to access registers.

Table 14.8   Conditions to Access Registers

						RCAN-ET Registers			
Status Mode	MCR GSR	IRR IMR	BCR	MBIMR	Flag_register	Mailbox (ctrl0, LAFM)		Mailbox (data)	Mailbox (ctrl1)
Reset	yes	yes	yes	yes	yes	yes		yes	yes
Transmission Reception Halt Request	yes	yes	no*[1]	yes	yes	no*[1]	yes*[2]	yes*[2]	no*[1] yes*[2]
Halt	yes	yes	no*[1]	yes	yes	yes		yes	yes
Sleep	yes	yes	no	no	no	no		no	no

Legend:

yes: Register access allowed

no: Register access prohibited

Notes: 1. No hardware protection

2. When TXPR is not set.

14.6.2   Test Mode Settings

The RCAN-ET has various test modes. The register TST[2:0] (MCR[10:8]) is used to select the RCAN-ET test mode. The default (initialised) settings allow RCAN-ET to operate in Normal mode. The following table is examples for test modes.

Test Mode can be selected only while in configuration mode. The user must then exit the configuration mode (ensuring BCR0/BCR1 is set) in order to run the selected test mode.

Table 14.9   Test Mode Settings

Bit 10: TST2	Bit 9: TST1	Bit 8: TST0	Description
0	0	0	Normal Mode (initial value)
0	0	1	Listen-Only Mode (Receive-Only Mode)
0	1	0	Self Test Mode 1 (External)
0	1	1	Self Test Mode 2 (Internal)
1	0	0	Write Error Counter
1	0	1	Error Passive Mode
1	1	0	Setting prohibited
1	1	1	Setting prohibited

- Normal Mode

  RCAN-ET operates in the normal mode.

- Listen-Only Mode:

  ISO-11898 requires this mode for baud rate detection. The Error Counters are cleared and disabled so that the TEC/REC does not increase the values, and the CTx Output is disabled so that RCAN-ET does not generate error frames or acknowledgment bits. IRR13 is set when a message error occurs.

- Self Test Mode 1

  RCAN-ET generates its own Acknowledge bit, and can store its own messages into a reception mailbox (if required). The CRx/CTx pins must be connected to the CAN bus.

- Self Test Mode 2

  RCAN-ET generates its own Acknowledge bit, and can store its own messages into a reception mailbox (if required). The CRx/CTx pins do not need to be connected to the CAN bus or any external devices, as the internal CTx is looped back to the internal CRx. CTx pin outputs only recessive bits and CRx pin is disabled.

- Write Error Counter

  TEC/REC can be written in this mode. RCAN-ET can be forced to become an Error Passive mode by writing a value greater than 127 into the Error Counters. The value written into TEC is used to write into REC, so only the same value can be set to these registers. Similarly, RCAN-ET can be forced to become an Error Warning by writing a value greater than 95 into them.

- Error Passive mode

  RCAN-ET needs to be in Halt Mode when writing into TEC/REC (MCR1 must be "1" when writing to the Error Counter). Furthermore this test mode needs to be exited prior to leaving Halt mode. Error Passive Mode: RCAN-ET can be forced to enter Error Passive mode.

Note: The REC will not be modified by implementing this Mode. However, once running in Error Passive Mode, the REC will increase normally should errors be received. In this Mode, RCAN-ET will enter BusOff if TEC reaches 256 (Dec). However when this mode is used RCAN-ET will not be able to become Error Active. Consequently, at the end of the Bus Off recovery sequence, RCAN-ET will move to Error Passive and not to Error Active

When message error occurs, IRR13 is set in all test modes.

RENESAS

14.6.3   Message Transmission Sequence

(1)   Message Transmission Request

Figure 14.9 shows the sequence of CAN frame transmission onto the bus. As described in the previous register section, please note that IRR8 is set when one of the TXACK or ABACK bits is set, meaning one of the Mailboxes has completed its transmission or transmission abortion and is now ready to be updated for the next transmission, whereas, the GSR2 means that there is currently no transmission request made (No TXPR flags set).
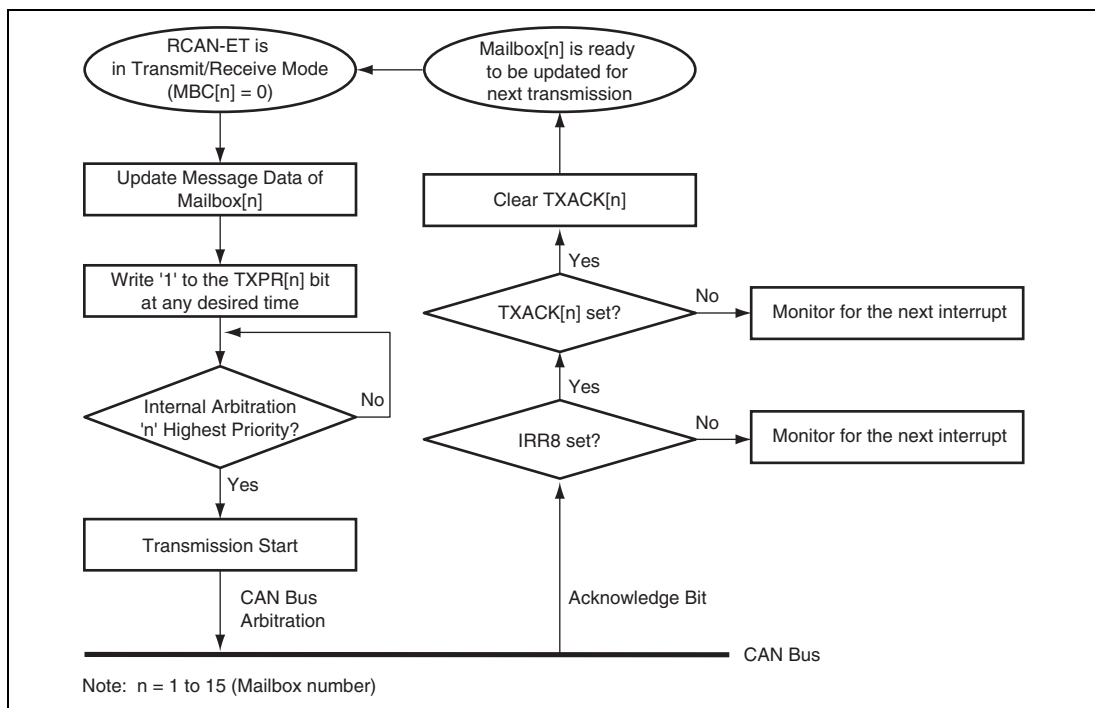
Figure 14.9   Transmission Request

(2)   Internal Arbitration for Transmission

The following diagram explains how RCAN-ET manages to schedule transmission-requested messages in the correct order based on the CAN identifier. 'Internal arbitration' picks up the highest priority message amongst transmit-requested messages.
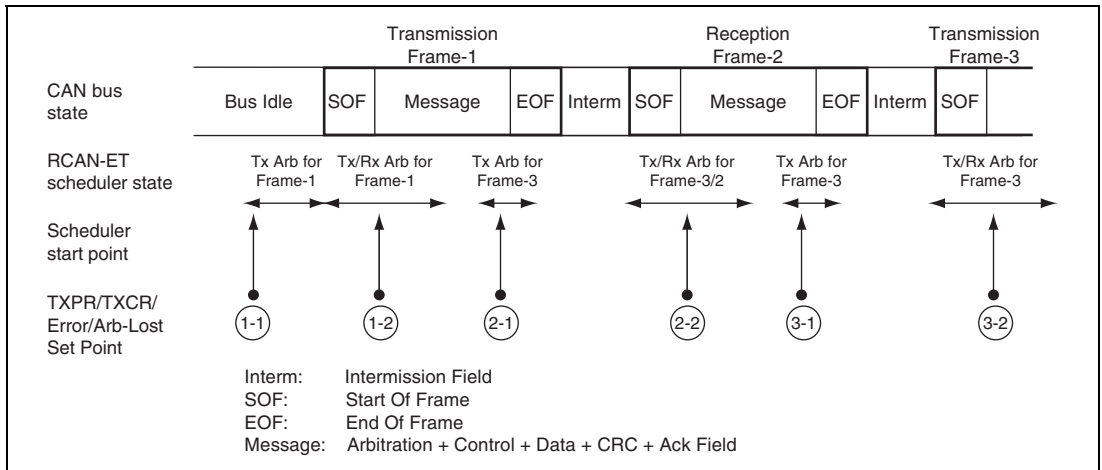
Figure 14.10   Internal Arbitration for Transmission

The RCAN-ET has two state machines. One is for transmission, and the other is for reception.

1-1: When a TXPR bit(s) is set while the CAN bus is idle, the internal arbitration starts running immediately and the transmission is started.

1-2: Operations for both transmission and reception starts at SOF. Since there is no reception frame, RCAN-ET becomes transmitter.

2-1: At crc delimiter, internal arbitration to search next message transmitted starts.

2-2: Operations for both transmission and reception starts at SOF. Because of a reception frame with higher priority, RCAN-ET becomes receiver. Therefore, Reception is carried out instead of transmitting Frame-3.

3-1: At crc delimiter, internal arbitration to search next message transmitted starts.

3-2: Operations for both transmission and reception starts at SOF. Since a transmission frame has higher priority than reception one, RCAN-ET becomes transmitter.

Internal arbitration for the next transmission is also performed at the beginning of each error delimiter in case of an error is detected on the CAN Bus. It is also performed at the beginning of error delimiters following overload frame.

As the arbitration for transmission is performed at CRC delimiter, in case a remote frame request is received into a Mailbox with ATX = 1 the answer can join the arbitration for transmission only at the following Bus Idle, CRC delimiter or Error Delimiter.

Depending on the status of the CAN bus, following the assertion of the TXCR, the corresponding Message abortion can be handled with a delay of maximum 1 CAN Frame.

14.6.4   Message Receive Sequence

The diagram below shows the message receive sequence.

CAN

End Of Arbitration Field

End Of Frame

RCAN-ET

IDLE

Valid CAN-ID Received

N = N - 1

Valid CAN Frame Received

Loop (N = 15; N ≥ 0; N = N - 1)

Compare ID with
Mailbox[N] + LAFM[N]
(if MBC is config to receive)

Yes

ID Matched?

No

N = 0?

No

Yes

Yes

Store Mailbox-Number[N]
and go back to idle state

RXPR[N]
(RFPR[N])
Already Set?

No

Yes

OverWrite

MSG
OverWrite or
OverRun?
(NMC)

OverRun

• Store Message by Overwriting
• Set UMSR
• Set IRR9 (if MBIMR[N] = 0)
• Generate Interrupt Signal
  (if IMR9 = 0)
• Set RXPR[N] (RFPR[N])
• Set IRR1 (IRR2)
  (if MBIMR[N] = 0)
• Generate Interrupt Signal
  (if IMR1 (IMR2) = 0)

• Reject Message
• Set UMSR
• Set IRR9 (if MBIMR[N] = 0)
• Generate Interrupt Signal
  (if IMR9 = 0)
• Set RXPR[N] (RFPR[N])*1

• Store Message
• Set RXPR[N] (RFPR[N])
• Set IRR1 (IRR2) (if MBIMR[N] = 0)
• Generate Interrupt Signal
  (if IMR1 (IMR2) = 0)

Intrrupt signal

Intrrupt signal

Intrrupt signal

CPU received interrupt due to CAN Message Reception

Exit Interrupt
Service Routine

Check and clear
UMSR[N]*2

Check and clear
UMSR[N]*2

Write 1 to RXPR[N]

Write 1 to RFPR[N]

Read Mailbox[N]

Read Mailbox[N]

Read RXPR[N] = 1

Read RFPR[N] = 1

Yes

IRR[1]
set?

No

Read IRR

Notes: N = 0 to 15 (Mailbox number)
1.   Only if CPU clears RXPR[N]/RFPR[N] at the same time that UMSR is set in overrun, RXPR[N]/RFPR[N]
     may be set again even though the message has not been updated.
2.   In case overwrite configuration (NMC = 1) is used for the Mailbox N the message must be discarded when
     UMSR[N] = 1, UMSR[N] cleared and the full Interrupt Service Routine started again. In case of overrun
     configuration (NMC = 0) is used clear again RXPR[N]/RFPR[N]/UMSR[N] when UMSR[N] = 1 and consider
     the message obsolete.

Figure 14.11   Message receive sequence

RENESAS

When RCAN-ET recognizes the end of the Arbitration field while receiving a message, it starts comparing the received identifier to the identifiers set in the Mailboxes, starting from Mailbox-15 down to Mailbox-0. It first checks the MBC if it is configured as a receive box, and reads LAFM, and reads the CAN-ID of Mailbox-15 (if configured as receive) to finally compare them to the received ID. If it does not match, the same check takes place at Mailbox-14 (if configured as receive). Once RCAN-ET finds a matching identifier, it stores the number of Mailbox-[N] into an internal buffer, stops the search, and goes back to idle state, waiting for the End Of Frame (EOF) to come. When the 6th bit of EOF is notified by the CAN Interface logic, the received message is written or abandoned, depending on the NMC bit. No modification of configuration during communication is allowed. Entering Halt Mode is one of ways to modify configuration. If it is written into the corresponding Mailbox, including the CAN-ID, i.e., there is a possibility that the CAN-ID is overwritten by a different CAN-ID of the received message due to the LAFM used. This also implies that, if the identifier of a received message matches to ID + LAFM of 2 or more Mailboxes, the higher numbered Mailbox will always store the relevant messages and the lower numbered Mailbox will never receive messages. Therefore, the settings of the identifiers and LAFMs need to be carefully selected.

With regards to the reception of data and remote frames described in the above flow diagram the clearing of the UMSR flag after the reading of IRR is to detect situations where a message is overwritten by a new incoming message stored in the same mailbox while the interrupt service routine is running. If during the final check of UMSR a overwrite condition is detected the message needs to be discarded and read again.

In case UMSR is set and the Mailbox is configured for overrun (NMC = 0), the message is still valid, however it is obsolete as it is not reflecting the latest message monitored on the CAN bus. Please access the full Mailbox content before clearing the related RXPR/RFPR flag.

Please note that in the case a received remote frame is overwritten by a data frame, both the remote frame request interrupt (IRR2) and data frame received interrupt (IRR1) and also the Receive Flags (RXPR and RFPR) are set. In an analogous way, the overwriting of a data frame by a remote frame, leads to setting both IRR2 and IRR1.

In the Overrun Mode (NMC = '0'), only the first Mailbox will cause the flags to be asserted. So, if a Data Frame is initially received, then RXPR and IRR1 are both asserted. If a Remote Frame is then received before the Data Frame has been read, then RFPR and IRR2 are NOT set. In this case UMSR of the corresponding Mailbox will still be set.

RENESAS

14.6.5   Reconfiguration of Mailbox

When re-configuration of Mailboxes is required, the following procedures should be taken.

(1)   Change Configuration of Transmit Box

Two cases are possible.

- Change of ID, RTR, IDE, LAFM, Data, DLC, NMC, ATX, DART

  This change is possible only when MBC=B'000. Confirm that the corresponding TXPR is not set. The configuration (except MBC bit) can be changed at any time.

- Change from transmit to receive configuration (MBC)

  Confirm that the corresponding TXPR is not set. The configuration can be changed only in Halt or reset state. Please note that it might take longer for RCAN-ET to transit to halt state if it is receiving or transmitting a message (as the transition to the halt state is delayed until the end of the reception/transmission), and also RCAN-ET will not be able to receive/transmit messages during the Halt state.

  In case RCAN-ET is in the Bus Off state the transition to halt state depends on the configuration of the bit 6 of MCR and also bit and 14 of MCR.

(2)   Change Configuration (ID, RTR, IDE, LAFM, Data, DLC, NMC, ATX, DART, MBC) of Receive Box or Change Receive Box to Transmit Box

The configuration can be changed only in Halt Mode.

RCAN-ET will not lose a message if the message is currently on the CAN bus and RCAN-ET is a receiver. RCAN-ET will be moving into Halt Mode after completing the current reception. Please note that it might take longer if RCAN-ET is receiving or transmitting a message (as the transition to the halt state is delayed until the end of the reception/transmission), and also RCAN-ET will not be able to receive/transmit messages during the Halt Mode.

In case RCAN-ET is in the Bus Off state the transition to halt mode depends on the configuration of the bit 6 and 14 of MCR.

Method by Halt Mode
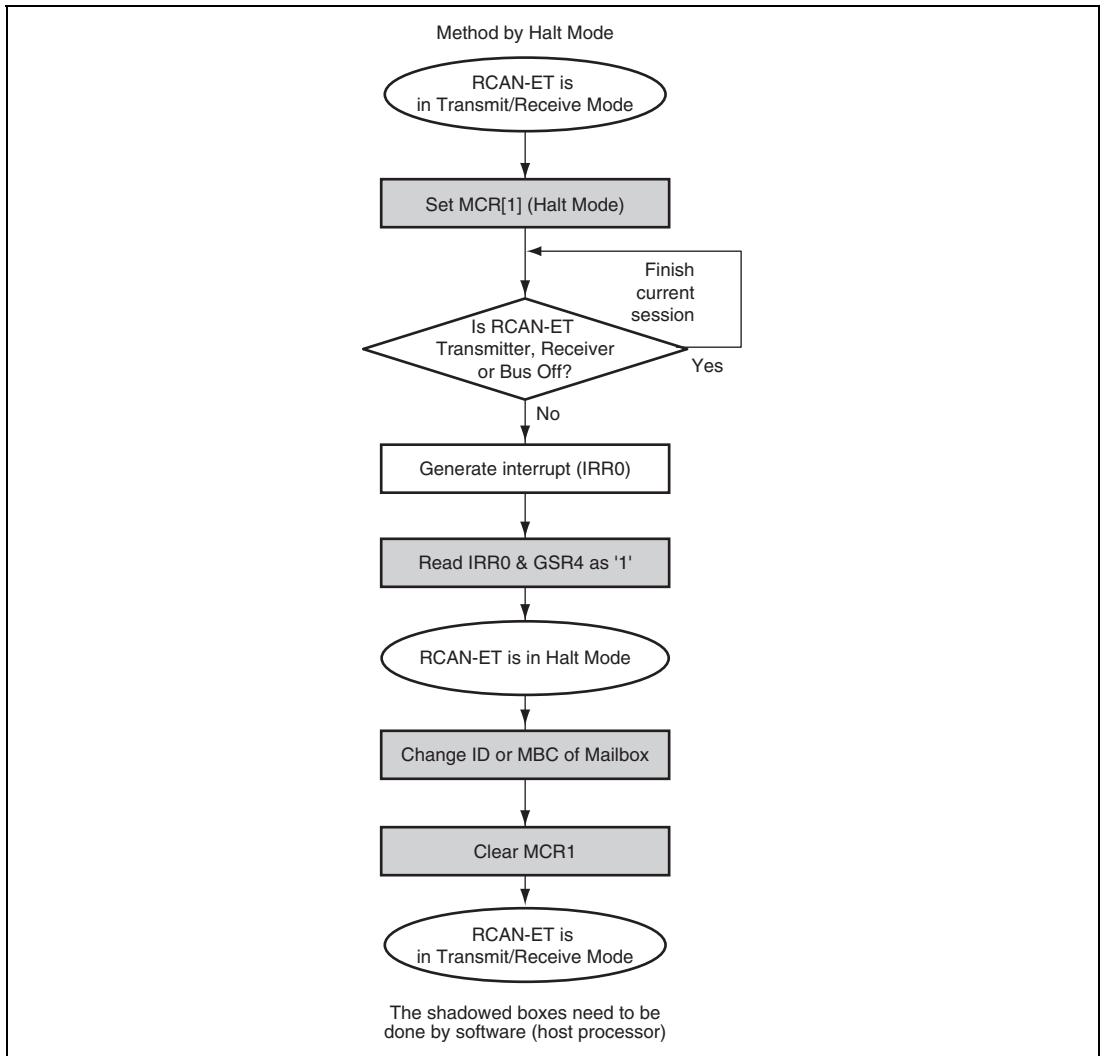
Figure 14.12   Change ID of Receive Box or Change Receive Box to Transmit Box

14.7   Interrupt Sources

Table 14.10 lists the RCAN-ET interrupt sources. With the exception of the reset processing interrupt (IRR0) by a power-on reset, these sources can be masked. Masking is implemented using the mailbox interrupt mask register 0 (MBIMR0) and interrupt mask register (IMR). For details on the interrupt vector of each interrupt source, see section 5, Interrupt Controller.

Table 14.10   RCAN-ET Interrupt Sources

Channel	Interrupt	Description	Interrupt Flag	DMAC Activation
0	ERS_0	Error Passive Mode (TEC ≥128 or REC ≥128)	IRR5	Not possible
		Bus Off (TEC ≥256)/Bus Off recovery	IRR6	
		Error warning (TEC ≥96)	IRR3	
		Error warning (REC ≥96)	IRR4	
	OVR_0	Message error detection	IRR13*[1]	
		Reset/halt/CAN sleep transition	IRR0	
		Overload frame transmission	IRR7	
		Unread message overwrite (overrun)	IRR9	
		Detection of CAN bus operation in CAN sleep mode	IRR12	
	SLE_0	Message transmission/transmission disabled (slot empty)	IRR8	
	RM1_0*[2]	Data frame reception/	IRR1*[3]	
	RM0_0*[2]	Remote frame reception	IRR2*[3]	Possible
1	ERS_1	Error Passive Mode (TEC ≥128 or REC ≥128)	IRR5	Not possible
		Bus Off (TEC ≥256)/Bus Off recovery	IRR6	
		Error warning (TEC ≥96)	IRR3	
		Error warning (REC ≥96)	IRR4	
	OVR_1	Message error detection	IRR13*[1]	
		Reset/halt/CAN sleep transition	IRR0	
		Overload frame transmission	IRR7	
		Unread message overwrite (overrun)	IRR9	
		Detection of CAN bus operation in CAN sleep mode	IRR12	
	SLE_1	Message transmission/transmission disabled (slot empty)	IRR8	
	RM1_1*[2]	Data frame reception/	IRR1*[3]	
	RM0_1*[2]	Remote frame reception	IRR2*[3]	Possible

Notes: 1. Available only in Test Mode.
2. RM0 is an interrupt generated by the remote request pending flag for mailbox 0 (RFPR0[0]) or the data frame receive flag for mailbox 0 (RXPR0[0]). RM1 is an interrupt generated by the remote request pending flag for mailbox n (RFPR0[n]) or the data frame receive flag for mailbox n (RXPR0[n]) (n = 1 to 15).
3. IRR1 is a data frame received interrupt flag for mailboxes 0 to 15, and IRR2 is a remote frame request interrupt flag for mailboxes 0 to 15.

RENESAS

14.8   PORT Interface

The RCAN-ET monitor register (RCANMON) controls RCAN-ET pins.

14.8.1   RCAN-ET Monitor Register (RCANMON)

RCANMON controls the transmit stop condition of transmit pins, enables/disables RCAN-ET transmit/receive pins, and monitors the status of RCAN-ET pins. This register is not affected by the module stop or CAN sleep mode, nor initialized by the software reset (MCR0).

Bit	7	6	5	4	3	2	1	0
Bit Name	—	CTxSTP	RCANE	—	—	—	CTxD	CRxD
Initial Value	0	0	0	0	0	0	Undefined	Undefined
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved
				The written value should always be '0'.
6	CTxSTP	0	R/W	RCAN Transmit Stop
				When this bit is set to 1, CTx pins are set to 1 regardless of the RCAN-ET transmit data.
5	RCANE	0	R/W	RCAN-ET Transmit/Receive Pin Enable
				When this bit is set to 1, CTx and CRx pins of RCAN-ET are enabled.
4 to 2	—	All 0	R/W	Reserved
				The written value should always be '0'.
1	CTxD	Undefined	R	RCAN Transmit Data Monitor
				The condition of CTx pins is acquired when this bit is read. Writing to this bit is invalid.
0	CRxD	Undefined	R	RCAN Receive Data Monitor
				The condition of CRx pins is acquired when this bit is read. Writing to this bit is invalid.

The overview of RCANMON bits in the PORT interface is shown in figure 14.13.
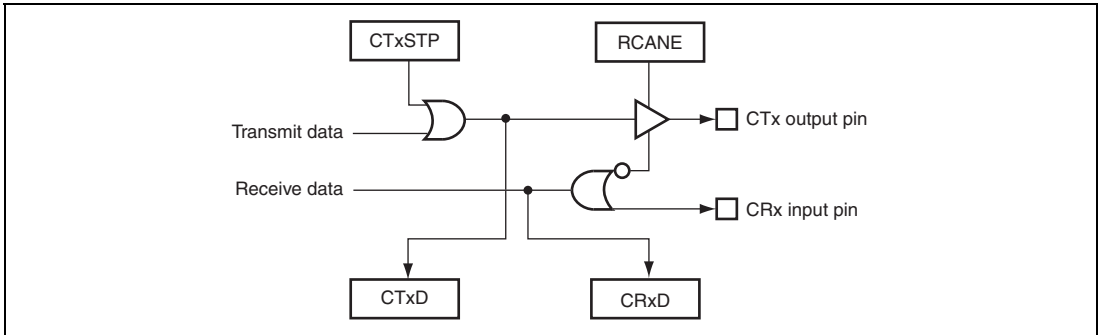

Figure 14.13   Overview of PORT Interface

14.9   CAN Bus Interface

A bus transceiver IC is necessary to connect this LSI to a CAN bus. A Renesas HA13721 transceiver IC and its compatible products are recommended. Figure 14.14 shows a sample connection diagram.
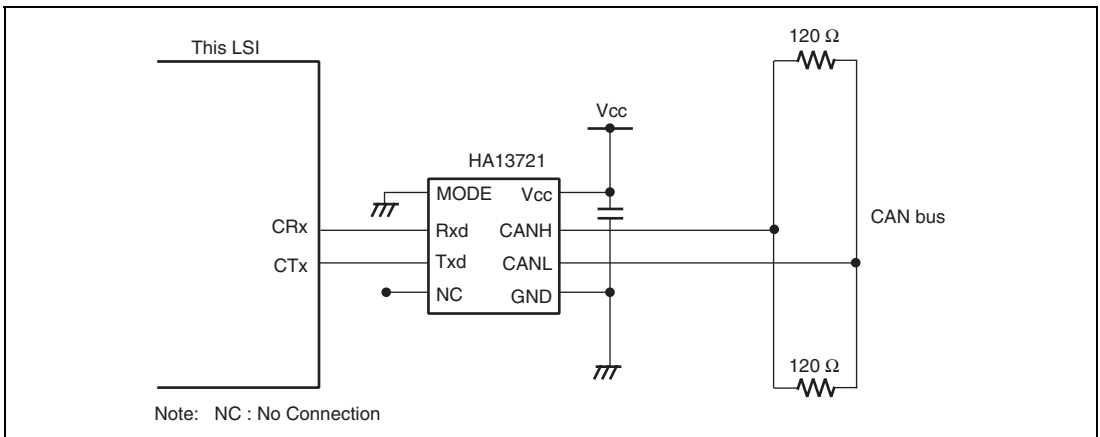
Figure 14.14   High-Speed Interface Using HA13721

14.10   Usage Notes

14.10.1   Module Stop Mode

The module stop control register (MSTPCRC) controls the supply of clocks to RCAN-ET. As an initial value, the clock to RCAN-ET is halted. Registers except for the RCAN-ET monitor register (RCANMON) should be accessed after the module stop mode is released.

14.10.2   Reset

Two types of resets are supported for RCAN-ET.

- Hardware reset
  RCAN-ET is initialized by a power-on reset, or hardware standby, module stop, or software standby mode.
- Software reset
  The MCR0 bit in the master control register (MCR) initializes registers other than MCR and CAN communication functions.

As the IRR0 bit in the interrupt request register (IRR) is initialized and set to 1 at a reset, it should be cleared to 0 in the configuration mode shown in the reset sequence diagram.

The area except for the message control field 1 (CONTROL1) of Mailbox is consisted of RAM, and not initialized at a reset. After a power-on reset, all the Mailboxes should be initialized in the configuration mode shown in the reset sequence diagram.

14.10.3   CAN Sleep Mode

The supply of main clocks in the modules is stopped in CAN sleep mode. Therefore, registers other than MCR, GSR, IRR, and IMR should not be accessed in CAN sleep mode.

14.10.4   Register Access

When the CAN bus receive frame is being stored in the Mailbox with the CAN communication functions of RCAN-ET, accessing the Mailbox area generates 0 to 5 peripheral bus cycles as a wait.

RENESAS

14.10.5   Interrupts

As shown in table 14.10, the Mailbox 0 receive interrupt enables the DMAC activation. When an interrupt is specified as to be activated by the Mailbox 0 receive interrupt and cleared by the interrupt source at the DMA transfer, up to the message control field 1 (CONTROL1) of Mailbox 0 should be read using the block transfer mode.

When clearing the interrupt source flags shown in table 14.10 by the CPU, be sure to read the flag after clearing it in the interrupt service routine. This must be done to prevent the CPU from executing the RTE instruction before the interrupt is cleared in the interrupt controller after the interrupt source flag has been cleared.

RENESAS

Section 15   Synchronous Serial Communication Unit (SSU)

This LSI has two independent synchronous serial communication unit (SSU*) channels. The SSU has master mode in which this LSI outputs clocks as a master device for synchronous serial communication and slave mode in which clocks are input from an external device for synchronous serial communication. Synchronous serial communication can be performed with devices having different clock polarity and clock phase. Figure 15.1 is a block diagram of the SSU.

Note: *   In this section, synchronous serial communication is referred to as SSU.

15.1   Features

- Choice of SSU mode and clock synchronous mode
- Choice of master mode and slave mode
- Choice of standard mode and bidirectional mode
- Synchronous serial communication with devices with different clock polarity and clock phase
- Choice of 8/16/24/32-bit width of transmit/receive data
- Full-duplex communication capability

  The shift register is incorporated, enabling transmission and reception to be executed simultaneously.
- Consecutive serial communication
- Choice of LSB-first or MSB-first transfer
- Choice of a clock source

  Pφ/4, Pφ/8, Pφ/16, Pφ/32, Pφ/64, Pφ/128, Pφ/256, or an external clock
- Five interrupt sources

  Transmit-end, transmit-data-register-empty, receive-data-full, overrun-error, and conflict error
- Module stop mode can be set

Figure 15.1 shows a block diagram of the SSU.
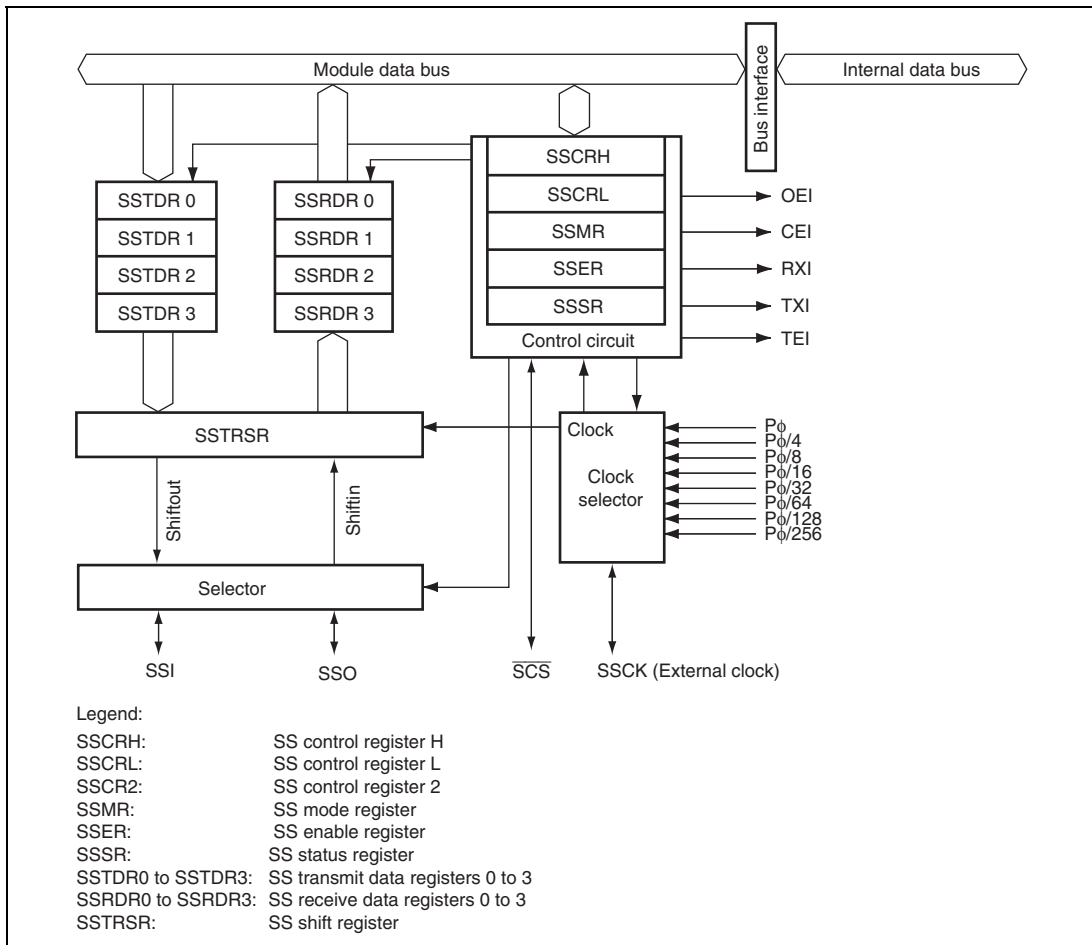
Figure 15.1 Block Diagram of SSU

15.2   Input/Output Pins

Table 15.1 shows the SSU pin configuration.

Table 15.1   Pin Configuration

Channel	Abbr.*	I/O	Function
0	SSCK0	I/O	Channel 0 SSU clock input/output
	SSI0	I/O	Channel 0 SSU data input/output
	SSO0	I/O	Channel 0 SSU data input/output
	$\overline{\text{SCS0}}$	I/O	Channel 0 SSU chip select input/output
1	SSCK1	I/O	Channel 1 SSU clock input/output
	SSI1	I/O	Channel 1 SSU data input/output
	SSO1	I/O	Channel 1 SSU data input/output
	$\overline{\text{SCS1}}$	I/O	Channel 1 SSU chip select input/output

Note: *   Because channel numbers are omitted in later descriptions, these are shown SSCK,
SSI, SSO, and $\overline{\text{SCS}}$.

RENESAS

15.3   Register Descriptions

The SSU has the following registers.

Channel 0:

- SS control register H_0 (SSCRH_0)
- SS control register L_0 (SSCRL_0)
- SS mode register_0 (SSMR_0)
- SS enable register_0 (SSER_0)
- SS status register_0 (SSSR_0)
- SS control register 2_0 (SSCR2_0)
- SS transmit data register 0_0 (SSTDR0_0)
- SS transmit data register 1_0 (SSTDR1_0)
- SS transmit data register 2_0 (SSTDR2_0)
- SS transmit data register 3_0 (SSTDR3_0)
- SS receive data register 0_0 (SSRDR0_0)
- SS receive data register 1_0 (SSRDR1_0)
- SS receive data register 2_0 (SSRDR2_0)
- SS receive data register 3_0 (SSRDR3_0)
- SS shift register_0 (SSTRSR_0)

Channel 1:

- SS control register H_1 (SSCRH_1)
- SS control register L_1 (SSCRL_1)
- SS mode register_1 (SSMR_1)
- SS enable register_1 (SSER_1)
- SS status register_1 (SSSR_1)
- SS control register 2_1 (SSCR2_1)
- SS transmit data register 0_1 (SSTDR0_1)
- SS transmit data register 1_1 (SSTDR1_1)
- SS transmit data register 2_1 (SSTDR2_1)
- SS transmit data register 3_1 (SSTDR3_1)
- SS receive data register 0_1 (SSRDR0_1)
- SS receive data register 1_1 (SSRDR1_1)
- SS receive data register 2_1 (SSRDR2_1)
- SS receive data register 3_1 (SSRDR3_1)
- SS shift register_1 (SSTRSR_1)

15.3.1 SS Control Register H (SSCRH)

SSCRH specifies master/slave device selection, bidirectional mode enable, SSO pin output value selection, SSCK pin selection, and $\overline{\text{SCS}}$ pin selection.

Bit	7	6	5	4	3	2	1	0
Bit Name	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0
Initial Value	0	0	0	0	1	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	MSS	0	R/W	Master/Slave Device Select
				Selects that this module is used in master mode or slave mode. When master mode is selected, transfer clocks are output from the SSCK pin. When the CE bit in SSSR is set, this bit is automatically cleared.
				0: Slave mode is selected
				1: Master mode is selected
6	BIDE	0	R/W	Bidirectional Mode Enable
				Selects that both serial data input pin and output pin are used or one of them is used. However, transmission and reception are not performed simultaneously when bidirectional mode is selected. For details, see section 15.4.3, Relationship between Data Input/Output Pins and Shift Register.
				0: Standard mode (two pins are used for data input and output)
				1: Bidirectional mode (one pin is used for data input and output)
5	—	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
4	SOL	0	R/W	Serial Data Output Value Select
				The serial data output retains its level of the last bit after completion of transmission. The output level before or after transmission can be specified by setting this bit. When specifying the output level, use the MOV instruction after clearing the SOLP bit to 0. Since writing to this bit during data transmission causes malfunctions, this bit should not be changed.
				0: Serial data output is changed to low
				1: Serial data output is changed to high
3	SOLP	1	R/W	SOL Bit Write Protect
				When changing the output level of serial data, set the SOL bit to 1 or clear the SOL bit to 0 after clearing the SOLP bit to 0 using the MOV instruction.
				0: Output level can be changed by the SOL bit
				1: Output level cannot be changed by the SOL bit. This bit is always read as 1
2	SCKS	0	R/W	SSCK Pin Select
				Selects that the SSCK pin functions as a port or a serial clock pin. When the SSCK pin is used as a serial clock pin, this bit must be set to 1.
				0: Functions as an I/O port
				1: Functions as a serial clock
1	CSS1	0	R/W	$\overline{SCS}$ Pin Select
0	CSS0	0	R/W	Select that the $\overline{SCS}$ pin functions as a port or $\overline{SCS}$ input or output. However, when MSS = 0, the $\overline{SCS}$ pin functions as an input pin regardless of the CSS1 and CSS0 settings.
				00: I/O port
				01: Function as $\overline{SCS}$ input
				10: Function as $\overline{SCS}$ automatic input/output (function as $\overline{SCS}$ input before and after transfer and output a low level during transfer)
				11: Function as $\overline{SCS}$ automatic output (outputs a high level before and after transfer and outputs a low level during transfer)

15.3.2   SS Control Register L (SSCRL)

SSCRL selects operating mode, software reset, and transmit/receive data length.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	SSUMS	SRES	—	—	—	DATS1	DATS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.
6	SSUMS	0	R/W	Selects transfer mode from SSU mode and clock synchronous mode.
				0: SSU mode
				1: Clock synchronous mode
5	SRES	0	R/W	Software Reset
				Setting this bit to 1 forcibly resets the SSU internal sequencer. After that, this bit is automatically cleared. The ORER, TEND, TDRE, RDRF, and CE bits in SSSR and the TE and RE bits in SSER are also initialized. Values of other bits for SSU registers are held.
				To stop transfer, set this bit to 1 to reset the SSU internal sequencer.
4 to 2	—	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
1	DATS1	0	R/W	Transmit/Receive Data Length Select
0	DATS0	0	R/W	Select serial data length.
				00: 8 bits
				01: 16 bits
				10: 32 bits
				11: 24 bits

RENESAS

15.3.3   SS Mode Register (SSMR)

SSMR selects the MSB first/LSB first, clock polarity, clock phase, and clock rate of synchronous serial communication.

Bit	7	6	5	4	3	2	1	0
Bit Name	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	MLS	0	R/W	MSB First/LSB First Select
				Selects that the serial data is transmitted in MSB first or LSB first.
				0: LSB first
				1: MSB first
6	CPOS	0	R/W	Clock Polarity Select
				Selects the SSCK clock polarity.
				0: High output in idle mode, and low output in active mode
				1: Low output in idle mode, and high output in active mode
5	CPHS	0	R/W	Clock Phase Select (Only for SSU Mode)
				Selects the SSCK clock phase.
				0: Data changes at the first edge
				1: Data is latched at the first edge
4, 3	—	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
2	CKS2	0	R/W	Transfer Clock Rate Select
1	CKS1	0	R/W	Select the transfer clock rate (prescaler division rate) when an internal clock is selected.
0	CKS0	0	R/W	
				000: Reserved 100: Pϕ/32
				001: Pϕ/4 101: Pϕ/64
				010: Pϕ/8 110: Pϕ/128
				011: Pϕ/16 111: Pϕ/256

RENESAS

15.3.4   SS Enable Register (SSER)

SSER performs transfer/receive control of synchronous serial communication and setting of interrupt enable.

Bit	7	6	5	4	3	2	1	0
Bit Name	TE	RE	—	—	TEIE	TIE	RIE	CEIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	TE	0	R/W	Transmit Enable
				When this bit is set to 1, transmission is enabled.
6	RE	0	R/W	Receive Enable
				When this bit is set to 1, reception is enabled.
5, 4	—	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.
3	TEIE	0	R/W	Transmit End Interrupt Enable
				When this bit is set to 1, a TEI interrupt request is enabled.
2	TIE	0	R/W	Transmit Interrupt Enable
				When this bit is set to 1, a TXI interrupt request is enabled.
1	RIE	0	R/W	Receive Interrupt Enable
				When this bit is set to 1, an RXI interrupt request and an OEI interrupt request are enabled.
0	CEIE	0	R/W	Conflict Error Interrupt Enable
				When this bit is set to 1, a CEI interrupt request is enabled.

RENESAS

15.3.5   SS Status Register (SSSR)

SSSR is a status flag register for interrupts.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	ORER	—	—	TEND	TDRE	RDRF	CE
Initial Value	0	0	0	0	0	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	—	Reserved
				This bit is always read as 0. The write value should always be 0.
6	ORER	0	R/W	Overrun Error
				If the next data is received while RDRF = 1, an overrun error occurs, indicating abnormal termination. SSRDR stores 1-frame receive data before an overrun error occurs and loses data to be received later. While ORER = 1, consecutive serial reception cannot be continued. Serial transmission cannot be continued, either.
				[Setting condition]
				When one byte of the next reception is completed with RDRF = 1
				[Clearing condition]
				When writing 0 after reading ORER = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
5, 4	—	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
3	TEND	1	R	Transmit End
				[Setting conditions]
				• When the last bit of transmit data is transmitted while the TENDSTS bit in SSCR2 is cleared to 0 and the TDRE bit is set to 1
				• After the last bit of transmit data is transmitted while the TENDSTS bit in SSCR2 is set to 1 and the TDRE bit is set to 1
				[Clearing conditions]
				• When writing 0 after reading TEND = 1
				• When writing data to SSTDR
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
2	TDRE	1	R/W	Transmit Data Empty
				Indicates whether or not SSTDR contains transmit data.
				[Setting conditions]
				• When the TE bit in SSER is 0
				• When data is transferred from SSTDR to SSTRSR and SSTDR is ready to be written to
				[Clearing conditions]
				• When writing 0 after reading TDRE = 1
				• When writing data to SSTDR with TE = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	RDRF	0	R/W	Receive Data Register Full
				Indicates whether or not SSRDR contains receive data.
				[Setting condition]
				• When receive data is transferred from SSTRSR to SSRDR after successful serial data reception
				[Clearing conditions]
				• When writing 0 after reading RDRF = 1
				• When reading receive data from SSRDR
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
0	CE	0	R/W	Conflict/Incomplete Error
				Indicates that a conflict error has occurred when 0 is externally input to the $\overline{\text{SCS}}$ pin with SSUMS = 0 (SSU mode) and MSS = 1 (master mode).
				If the $\overline{\text{SCS}}$ pin level changes to 1 with SSUMS = 0 (SSU mode) and MSS = 0 (slave mode), an incomplete error occurs because it is determined that a master device has terminated the transfer. Data reception does not continue while the CE bit is set to 1. Serial transmission also does not continue. Reset the SSU internal sequencer by setting the SRES bit in SSCRL to 1 before resuming transfer after incomplete error.
				[Setting conditions]
				• When a low level is input to the $\overline{\text{SCS}}$ pin in master mode (the MSS bit in SSCRH is set to 1)
				• When the $\overline{\text{SCS}}$ pin is changed to 1 during transfer in slave mode (the MSS bit in SSCRH is cleared to 0)
				[Clearing condition]
				• When writing 0 after reading CE = 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

RENESAS

15.3.6   SS Control Register 2 (SSCR2)

SSCR2 is a register that enables/disables the open-drain outputs of the SSO, SSI, SSCK, and $\overline{\text{SCS}}$ pins, selects the assert timing of the $\overline{\text{SCS}}$ pin, data output timing of the SSO pin, and set timing of the TEND bit.

Bit	7	6	5	4	3	2	1	0
Bit Name	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	SDOS	0	R/W	Serial Data Pin Open Drain Select
				Selects whether the serial data output pin is used as a CMOS or an NMOS open drain output. Pins to output serial data differ according to the register setting. For details, see section 15.4.3, Relationship between Data Input/Output Pins and Shift Register.
				0: CMOS output
				1: NMOS open drain output
6	SSCKOS	0	R/W	SSCK Pin Open Drain Select
				Selects whether the SSCK pin is used as a CMOS or an NMOS open drain output.
				0: CMOS output
				1: NMOS open drain output
5	SCSOS	0	R/W	$\overline{\text{SCS}}$ Pin Open Drain Select
				Selects whether the $\overline{\text{SCS}}$ pin is used as a CMOS or an NMOS open drain output.
				0: CMOS output
				1: NMOS open drain output
4	TENDSTS	0	R/W	Selects the timing of setting the TEND bit (valid in SSU and master mode).
				0: Sets the TEND bit when the last bit is being transmitted
				1: Sets the TEND bit after the last bit is transmitted

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	SCSATS	0	R/W	Selects the assertion timing of the $\overline{\text{SCS}}$ pin (valid in SSU and master mode).
				0: Min. values of t_{LEAD} and t_{LAG} are $1/2 \times t_{SUcyc}$
				1: Min. values of t_{LEAD} and t_{LAG} are $3/2 \times t_{SUcyc}$
2	SSODTS	0	R/W	Selects the data output timing of the SSO pin (valid in SSU and master mode)
				0: While BIDE = 0, MSS = 1, and TE = 1 or while BIDE = 1, TE = 1, and RE = 0, the SSO pin outputs data
				1: While BIDE = 0, MSS = 1, and TE = 1 or while BIDE = 1, TE = 1, and RE = 0, the SSO pin outputs data while the SCS pin is driven low
1, 0	—	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.

RENESAS

15.3.7   SS Transmit Data Registers 0 to 3 (SSTDR0 to SSTDR3)

SSTDR is an 8-bit register that stores transmit data. When 8-bit data length is selected by bits DATS1 and DATS0 in SSCRL, SSTDR0 is valid. When 16-bit data length is selected, SSTDR0 and SSTDR1 are valid. When 24-bit data length is selected, SSTDR0 to SSTDR2 are valid. When 32-bit data length is selected, SSTDR0 to SSTDR3 are valid. Be sure not to access to invalid SSTDR.

When the SSU detects that SSTRSR is empty, it transfers the transmit data written in SSTDR to SSTRSR and starts serial transmission. If the next transmit data has already been written to SSTDR during serial transmission, the SSU performs consecutive serial transmission.

Although SSTDR can always be read from or written to by the CPU and DMAC, to achieve reliable serial transmission, write transmit data to SSTDR after confirming that the TDRE bit in SSSR is set to 1.

- SSTDR0

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

- SSTDR1

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

- SSTDR2

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

- SSTDR3

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

15.3.8   SS Receive Data Registers 0 to 3 (SSRDR0 to SSRDR3)

SSRDR is an 8-bit register that stores receive data. When 8-bit data length is selected by bits DATS1 and DATS0 in SSCRL, SSRDR0 is valid. When 16-bit data length is selected, SSRDR0 and SSRDR1 are valid. When 24-bit data length is selected, SSRDR0 to SSRDR2 are valid. When 32-bit data length is selected, SSRDR0 to SSRDR3 are valid. Be sure not to access to invalid SSRDR.

When the SSU has received 1-byte data, it transfers the received serial data from SSTRSR to SSRDR where it is stored. After this, SSTRSR is ready for reception. Since SSTRSR and SSRDR function as a double buffer in this way, consecutive receive operations can be performed.

Read SSRDR after confirming that the RDRF bit in SSSR is set to 1.

SSRDR is a read-only register, therefore, cannot be written to by the CPU.

• SSRDR0

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

• SSRDR1

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

• SSRDR2

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

• SSRDR3

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

15.3.9   SS Shift Register (SSTRSR)

SSTRSR is a shift register that transmits and receives serial data.

When data is transferred from SSTDR to SSTRSR, bit 0 of transmit data is bit 0 in the SSTDR contents (MLS = 0: LSB first communication) and is bit 7 in the SSTDR contents (MLS = 1: MSB first communication). The SSU transfers data from the LSB (bit 0) in SSTRSR to the SSO pin to perform serial data transmission.

In reception, the SSU sets serial data that has been input via the SSI pin in SSTRSR from the LSB (bit 0). When 1-byte data has been received, the SSTRSR contents are automatically transferred to SSRDR. SSTRSR cannot be directly accessed by the CPU.

15.4   Operation

15.4.1   Transfer Clock

A transfer clock can be selected from eight internal clocks and an external clock. When using this module, set the SCKS bit in SSCRH to 1 to select the SSCK pin as a serial clock. When the MSS bit in SSCRH is 1, an internal clock is selected and the SSCK pin is used as an output pin. When transfer is started, the clock with the transfer rate set by bits CKS2 to CKS0 in SSMR is output from the SSCK pin. When MSS = 0, an external clock is selected and the SSCK pin is used as an input pin.

15.4.2   Relationship of Clock Phase, Polarity, and Data

The relationship of clock phase, polarity, and transfer data depends on the combination of the CPOS and CPHS bits in SSMR. Figure 15.2 shows the relationship. When SSUMS = 1, the CPHS setting is invalid although the CPOS setting is valid.

Setting the MLS bit in SSMR selects that MSB or LSB first communication. When MLS = 0, data is transferred from the LSB to the MSB. When MLS = 1, data is transferred from the MSB to the LSB.
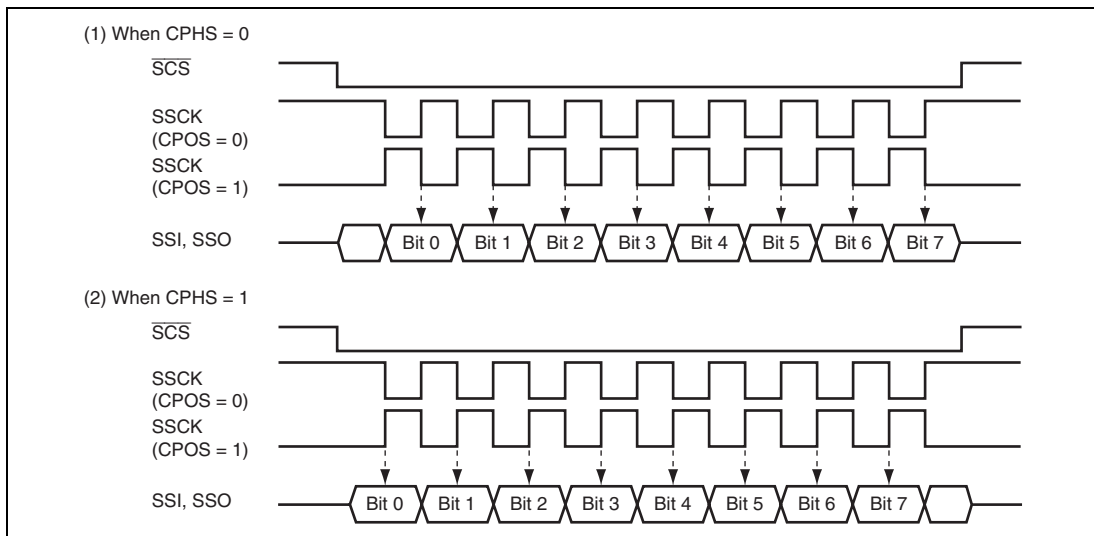
Figure 15.2   Relationship of Clock Phase, Polarity, and Data

15.4.3   Relationship between Data Input/Output Pins and Shift Register

The connection between data input/output pins and the SS shift register (SSTRSR) depends on the combination of the MSS and BIDE bits in SSCRH and the SSUMS bit in SSCRL. Figure 15.3 shows the relationship.

The SSU transmits serial data from the SSO pin and receives serial data from the SSI pin when operating with BIDE = 0 and MSS = 1 (standard, master mode) (see figure 15.3 (1)). The SSU transmits serial data from the SSI pin and receives serial data from the SSO pin when operating with BIDE = 0 and MSS = 0 (standard, slave mode) (see figure 15.3 (2)).

The SSU transmits and receives serial data from the SSO pin regardless of master or slave mode when operating with BIDE = 1 (bidirectional mode) (see figures 15.3 (3) and (4)).

However, even if both the TE and RE bits are set to 1, transmission and reception are not performed simultaneously. Either the TE or RE bit must be selected.

The SSU transmits serial data from the SSO pin and receives serial data from the SSI pin when operating with SSUMS = 1. The SSCK pin outputs the internal clock when MSS = 1 and function as an input pin when MSS = 0 (see figures 15.3 (5) and (6)).
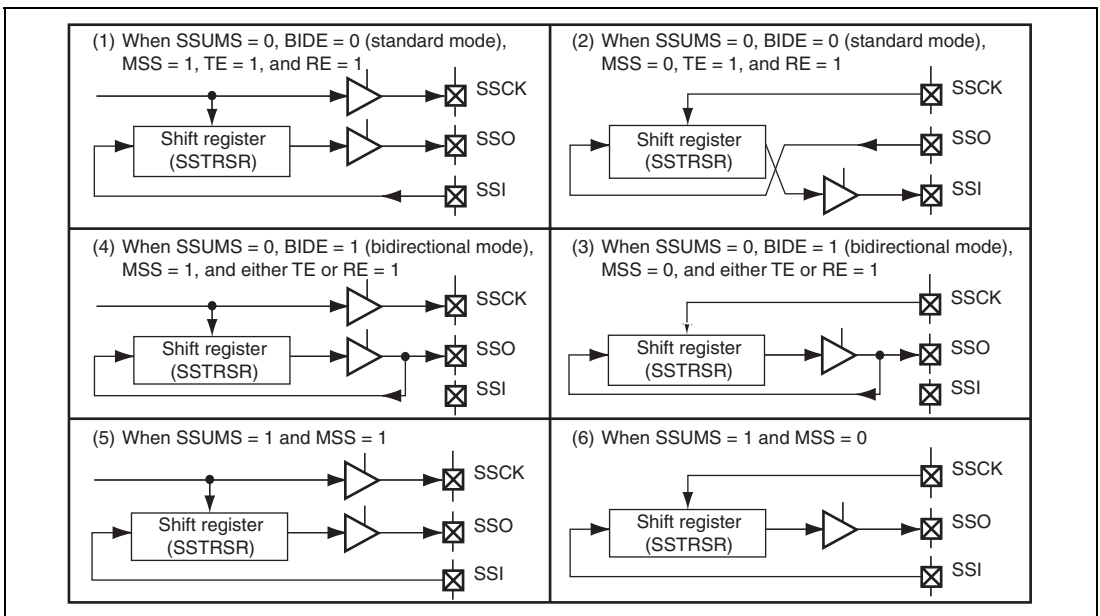
Figure 15.3   Relationship between Data Input/Output Pins and the Shift Register

RENESAS

15.4.4   Communication Modes and Pin Functions

The SSU switches the input/output pin (SSI, SSO, SSCK, and $\overline{\text{SCS}}$) functions according to the communication modes and register settings. When a pin is used as an input pin, set the corresponding bit in the input buffer control register (ICR) to 1. The relationship of communication modes and input/output pin functions are shown in tables 15.2 to 15.4.

Table 15.2   Communication Modes and Pin States of SSI and SSO Pins

Communication Mode	Register Setting					Pin State	
	SSUMS	BIDE	MSS	TE	RE	SSI	SSO
SSU communication mode	0	0	0	0	1	—	Input
				1	0	Output	—
				1	1	Output	Input
			1	0	1	Input	—
				1	0	—	Output
				1	1	Input	Output
SSU (bidirectional) communication mode	0	1	0	0	1	—	Input
				1	0	—	Output
			1	0	1	—	Input
				1	0	—	Output
Clock synchronous communication mode	1	0	0	0	1	Input	—
				1	0	—	Output
				1	1	Input	Output
			1	0	1	Input	—
				1	0	—	Output
				1	1	Input	Output

Legend:

—: Not used as SSU pin (can be used as I/O port)

Table 15.3   Communication Modes and Pin States of SSCK Pin

Communication Mode	Register Setting			Pin State
	SSUMS	MSS	SCKS	SSCK
SSU communication mode	0	0	0	—
			1	Input
		1	0	—
			1	Output
Clock synchronous communication mode	1	0	0	—
			1	Input
		1	0	—
			1	Output

Legend:

—: Not used as SSU pin (can be used as I/O port)

Table 15.4   Communication Modes and Pin States of $\overline{\text{SCS}}$ Pin

Communication Mode	Register Setting				Pin State
	SSUMS	MSS	CSS1	CSS0	$\overline{\text{SCS}}$
SSU communication mode	0	0	×	×	Input
		1	0	0	—
			0	1	Input
			1	0	Automatic input/output
			1	1	Output
Clock synchronous communication mode	1	×	×	×	—

Legend:

×: Don't care

—: Not used as SSU pin (can be used as I/O port)

RENESAS

15.4.5   SSU Mode

In SSU mode, data communications are performed via four lines: clock line (SSCK), data input line (SSI or SSO), data output line (SSI or SSO), and chip select line ($\overline{\text{SCS}}$).

In addition, the SSU supports bidirectional mode in which a single pin functions as data input and data output lines.

(1)   Initial Settings in SSU Mode

Figure 15.4 shows an example of the initial settings in SSU mode. Before data transfer, clear both the TE and RE bits in SSER to 0 to set the initial values.

Note: Before changing operating modes and communications formats, clear both the TE and RE bits to 0. Although clearing the TE bit to 0 sets the TDRE bit to 1, clearing the RE bit to 0 does not change the values of the RDRF and ORER bits and SSRDR. Those bits retain the previous values.

Start setting initial values

Clear TE and RE bits in SSER to 0

[1]   Set a bit in ICR to 1

[2]   Specify MSS, BIDE, SOL, SCKS, CSS1, and CSS0 bits in SSCRH

[3]   Clear SSUMS in SSCRH to 0 and specify bits DATS1 and DATS0

[4]   Specify MLS, CPOS, CPHS, CKS2, CKS1, and CKS0 bits in SSMR

[5]   Simultaneously specify TE, RE, TEIE, TIE, RIE, and CEIE bits in SSER

End

[1]   When the pin is used as an input.

[2]   Specify master/slave mode selection, bidirectional mode enable, SSO pin output value selection, SSCK pin selection, and $\overline{\text{SCS}}$ pin selection.

[3]   Selects SSU mode and specify transmit/receive data length.

[4]   Specify MSB first/LSB first selection, clock polarity selection, clock phase selection, and transfer clock rate selection.

[5]   Enables/disables interrupt request to the CPU.

Figure 15.4   Example of Initial Settings in SSU Mode

(2)   Data Transmission

Figure 15.5 shows an example of transmission operation, and figure 15.6 shows a flowchart example of data transmission.

When transmitting data, the SSU operates as shown below.

In master mode, the SSU outputs a transfer clock and data. In slave mode, when a low level signal is input to the $\overline{\text{SCS}}$ pin and a transfer clock is input to the SSCK pin, the SSU outputs data in synchronization with the transfer clock.

Writing transmit data to SSTDR after the completion of the initial setting clears the TDRE bit in SSSR to 0, and the SSTDR contents are transferred to SSTRSR. After that, the SSU sets the TDRE bit to 1 and starts transmission. At this time, if the TIE bit in SSER is set to 1, a TXI interrupt is generated.

When 1-frame data has been transferred with TDRE = 0, the SSTDR contents are transferred to SSTRSR to start the next frame transmission. When the 8th bit of transmit data has been transferred with TDRE = 1, the TEND bit in SSSR is set to 1 and the state is retained. At this time, if the TEIE bit is set to 1, a TEI interrupt is generated. After transmission, the output level of the SSCK pin is fixed high when CPOS = 0 and low when CPOS = 1.

While the ORER bit in SSSR is set to 1, transmission is not performed. Check that the ORER bit is cleared to 0.

Figure 15.5   Example of Transmission Operation (SSU Mode) (1)
When 8-bit data length is selected (SSTDR0 is valid) with CPOS = 0 and CPHS = 0

Figure 15.5   Example of Transmission Operation (SSU Mode) (2)
When 16-bit data length is selected (SSTDR0 and SSTDR1 are valid)
with CPOS = 0 and CPHS = 0

Figure 15.5   Example of Transmission Operation (SSU Mode) (3)
When 24-bit data length is selected (SSTDR0, SSTDR1, and SSTDR2 are valid)
with CPOS = 0 and CPHS = 0

Figure 15.5   Example of Transmission Operation (SSU Mode) (4)
When 32-bit data length is selected (SSTDR0, SSTDR1, SSTDR2, and SSTDR3 are valid)
with CPOS = 0 and CPHS = 0

[1] Initial setting:
Specify the transmit data format.

[2] Check that the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming that the TDRE bit in SSSR is 1. The TDRE bit is automatically cleared to 0 and transmission is started by writing data to SSTDR.

[3] Procedure for consecutive data transmission:
To continue data transmission, confirm that the TDRE bit is 1 meaning that SSTDR is ready to be written to. After that, data can be written to SSTDR. The TDRE bit is automatically cleared to 0 by writing data to SSTDR.

[4] Procedure for data transmission end:
To end data transmission, confirm that the TEND bit is cleared to 0. After completion of transmitting the last bit, clear the TE bit to 0.

Note: Hatching boxes represent SSU internal operations.
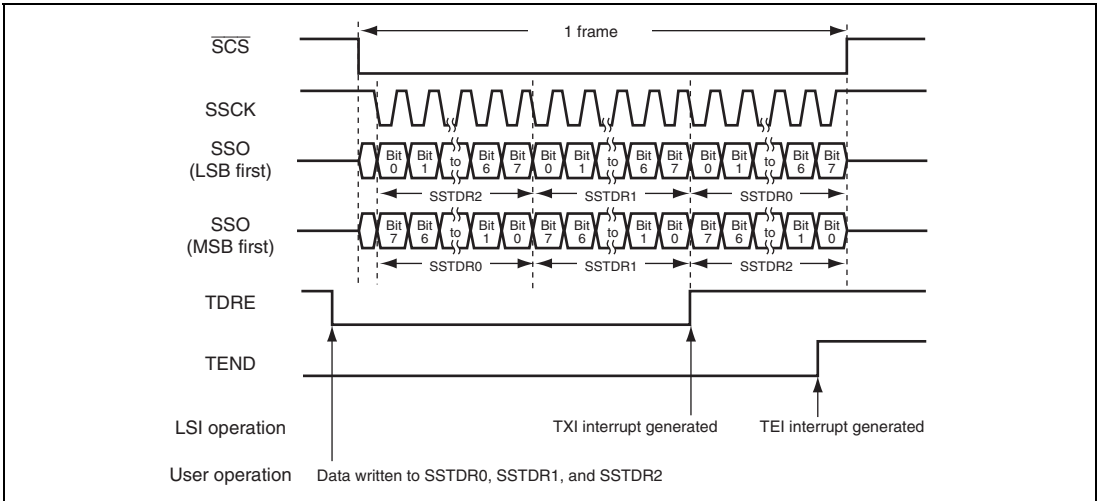
Figure 15.6   Flowchart Example of Data Transmission (SSU Mode)

(3)   Data Reception

Figure 15.7 shows an example of reception operation, and figure 15.8 shows a flowchart example of data reception. When receiving data, the SSU operates as shown below.

After completing the initial setting and executing dummy-read of SSRDR, the SSU starts data reception.

In master mode, the SSU outputs a transfer clock and receives data. In slave mode, when a low level signal is input to the $\overline{SCS}$ pin and a transfer clock is input to the SSCK pin, the SSU receives data in synchronization with the transfer clock.

When 1-frame data has been received, the RDRF bit in SSSR is set to 1 and the receive data is stored in SSRDR. At this time, if the RIE bit in SSER is set to 1, an RXI interrupt is generated. The RDRF bit is automatically cleared to 0 by reading SSRDR.

When the RDRF bit has been set to 1 at the 8th rising edge of the transfer clock, the ORER bit in SSSR is set to 1. This indicates that an overrun error (OEI) has occurred. At this time, data reception is stopped. While the ORER bit in SSSR is set to 1, reception is not performed. To resume the reception, clear the ORER bit to 0.

Figure 15.7   Example of Reception Operation (SSU Mode) (1)
When 8-bit data length is selected (SSRDR0 is valid) with CPOS = 0 and CPHS = 0
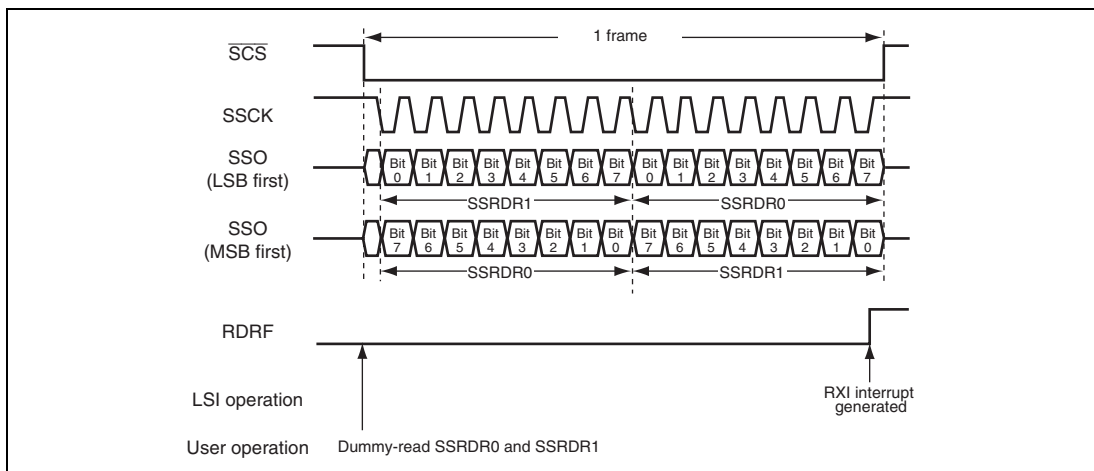
Figure 15.7   Example of Reception Operation (SSU Mode) (2)
When 16-bit data length is selected (SSRDR0 and SSRDR1 are valid)
with CPOS = 0 and CPHS = 0
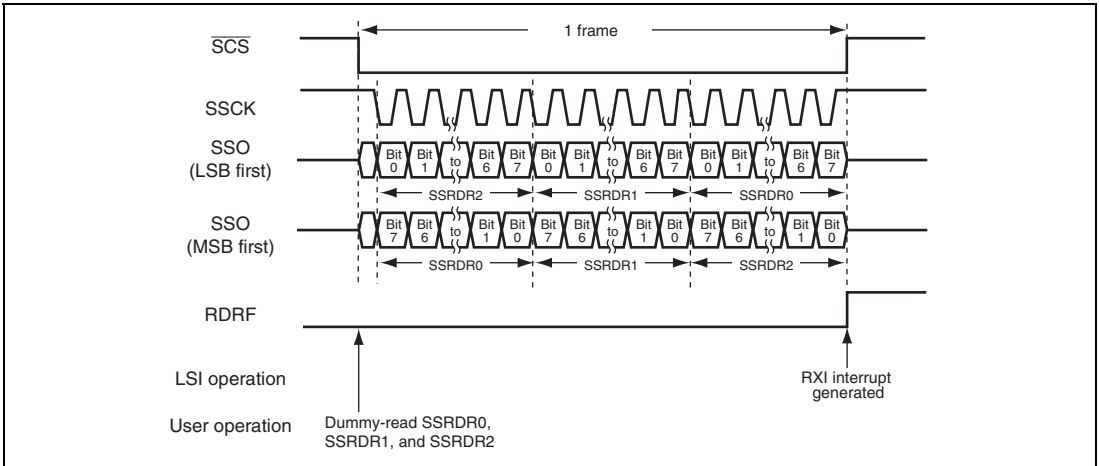
Figure 15.7   Example of Reception Operation (SSU Mode) (3)
When 24-bit data length is selected (SSRDR0, SSRDR1, and SSRDR2 are valid)
with CPOS = 0 and CPHS = 0
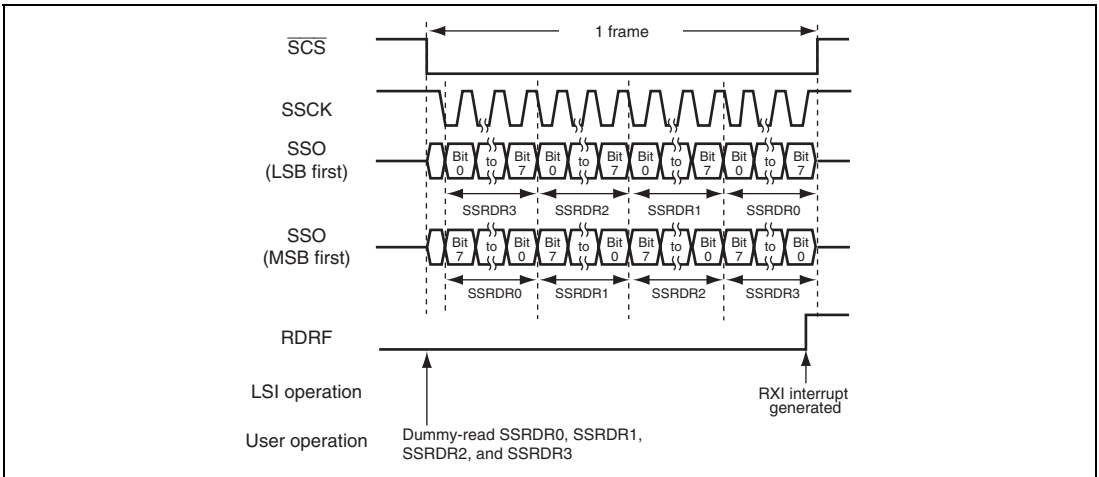
**Figure 15.7   Example of Reception Operation (SSU Mode) (4)**
When 32-bit data length is selected (SSRDR0, SSRDR1, SSRDR2 and SSRDR3 are valid)
with CPOS = 0 and CPHS = 0

Start

[1] Initial setting

[2] Dummy-read SSRDR

Read SSSR

RDRF = 1?　No

Yes

ORER = 1?　Yes [3]

No

[4] Consecutive data reception?　No

Yes

Read received data in SSRDR

RDRF automatically cleared

[5] RE = 0

Read receive data in SSRDR

End reception

[6] Overrun error processing

Clear ORER in SSSR

End reception

Note: Hatching boxes represent SSU internal operations.

[1] Initial setting:
     Specify the receive data format.

[2] Start reception:
     When SSRDR is dummy-read with RE = 1, reception is
     started.

[3], [6] Receive error processing:
     When a receive error occurs, execute the designated error
     processing after reading the ORER bit in SSSR. After that,
     clear the ORER bit to 0. While the ORER bit is set to 1,
     transmission or reception is not resumed.

[4] To continue single reception:
     When continuing single reception, wait for time of t_{SUcyc}
     while the RDRF flag is set to 1 and then read receive data
     in SSRDR.
     The next single reception starts after reading receive data
     in SSRDR.

[5] To complete reception:
     To complete reception, read receive data after clearing the
     RE bit to 0. When reading SSRDR without clearing the RE
     bit, reception is resumed.

Figure 15.8   Flowchart Example of Data Reception (SSU Mode)

(4)   Data Transmission/Reception

Figure 15.9 shows a flowchart example of simultaneous transmission/reception. The data transmission/reception is performed combining the data transmission and data reception as mentioned above. The data transmission/reception is started by writing transmit data to SSTDR with TE = RE = 1.

Before switching transmission mode (TE = 1) or reception mode (RE = 1) to transmission/reception mode (TE = RE = 1), clear the TE and RE bits to 0. When starting the transfer, confirm that the TEND, RDRF, and ORER bits are cleared to 0 before setting the TE or RE bit to 1.

RENESAS

Figure 15.9   Flowchart Example of Simultaneous Transmission/Reception (SSU Mode)

The flowchart contains the following steps and annotations:

Start

[1] Initial setting

[2] Read TDRE in SSSR.

TDRE = 1? — No (loops back)

Yes

Write transmit data to SSTDR

TDRE automatically cleared

Data transferred from SSTDR to SSTRSR

TDRE set to 1 to start transmission

Read SSSR

[3] RDRF = 1? — No (loops back)

Yes

ORER = 1? — Yes [4]

No

Read receive data in SSRDR

RDRF automatically cleared

Consecutive data transmission/reception? — Yes [5]

No

Read TEND in SSSR

TEND = 1? — No (loops back)

Yes

Clear TEND in SSSR to 0

Error processing

1-bit period elapsed? — No (loops back)

Yes

Clear TE and RE in SSER to 0

End transmission/reception

Note: Hatching boxes represent SSU internal operations.

[1] Initial setting:
Specify the transmit/receive data format.

[2] Check the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming that the TDRE bit in SSSR is 1. The TDRE bit is automatically cleared to 0 and transmission/reception is started by writing data to SSTDR.

[3] Check the SSU state:
Read SSSR confirming that the RDRF bit is 1. A change of the RDRF bit (from 0 to 1) can be notified by RXI interrupt.

[4] Receive error processing:
When a receive error occurs, execute the designated error processing after reading the ORER bit in SSSR. After that, clear the ORER bit to 0. While the ORER bit is set to 1, transmission or reception is not resumed.

[5] Procedure for consecutive data transmission/reception:
To continue serial data transmission/reception, confirm that the TDRE bit is 1 meaning that SSTDR is ready to be written to. After that, data can be written to SSTDR. The TDRE bit is automatically cleared to 0 by writing data to SSTDR.

15.4.6 $\overline{\text{SCS}}$ Pin Control and Conflict Error

When bits CSS1 and CSS0 in SSCRH are set to B'10 and the SSUMS bit in SSCRL is cleared to 0, the $\overline{\text{SCS}}$ pin functions as an input pin (Hi-Z) from the point the MSS bit in SSCRH is set to 1 until the serial transfer starts and after the transfer ends. During these periods, conflict error detection is performed. A conflict error occurs if a low level signal is input to the $\overline{\text{SCS}}$ pin while it is in a Hi-Z state. At this time, the CE bit in SSSR is set to 1 and the MSS bit is cleared to 0.

Note: While the CE bit is set to 1, transmission or reception cannot be resumed. Clear the CE bit to 0 before resuming the transmission or reception.
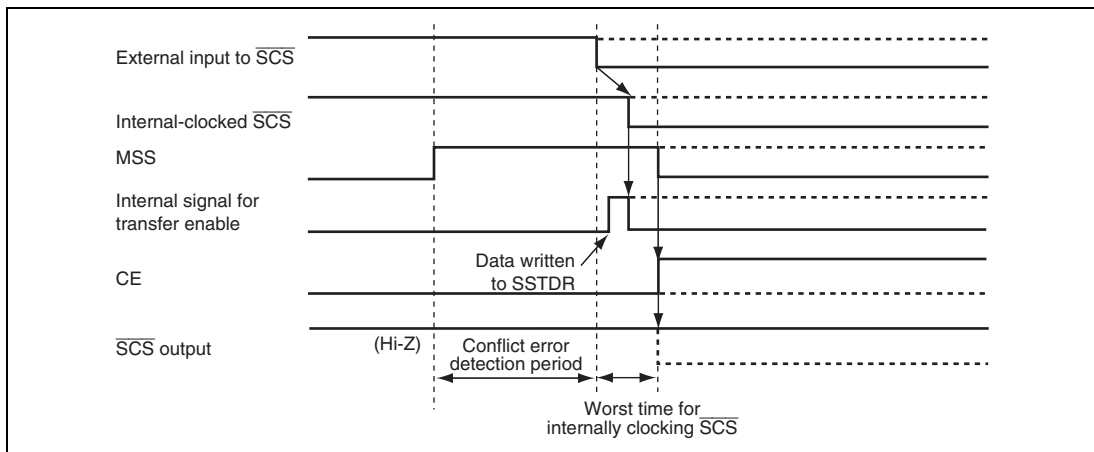
Figure 15.10 Conflict Error Detection Timing (Before Transfer)
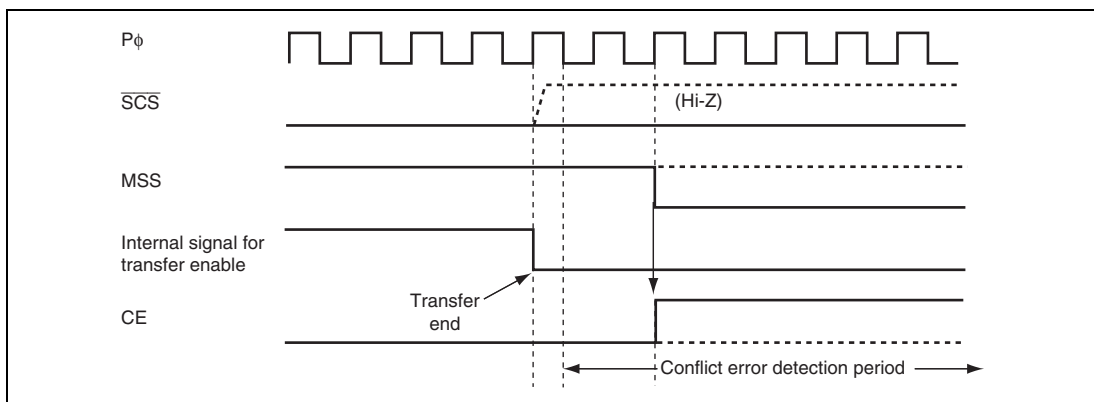
Figure 15.11 Conflict Error Detection Timing (After Transfer End)

15.4.7   Clock Synchronous Communication Mode

In clock synchronous communication mode, data communications are performed via three lines: clock line (SSCK), data input line (SSI), and data output line (SSO).

(1)   Initial Settings in Clock Synchronous Communication Mode

Figure 15.12 shows an example of the initial settings in clock synchronous communication mode. Before data transfer, clear both the TE and RE bits in SSER to 0 to set the initial values.

Note: Before changing operating modes and communications formats, clear both the TE and RE bits to 0. Although clearing the TE bit to 0 sets the TDRE bit to 1, clearing the RE bit to 0 does not change the values of the RDRF and ORER bits and SSRDR. Those bits retain the previous values.

Start setting initial values

Clear TE and RE bits in SSER to 0

[1] Set a bit in ICR to 1

[2] Specify MSS and SCKS in SSCRH

[3] Set SSUMS in SSCRL to 1 and specify bits DATS1 and DATS0

[4] Specify CPOS, CKS2, CKS1, and CKS0 bits in SSMR

Specify SDOS, SSCKOS, SCSOS, TENDSTS, SCSATS, and SSODTS bits in SSCR2

[5] Specify TE, RE, TEIE, TIE, RIE, and CEIE bits in SSER simultaneously

End

[1] When the pin is used as an input.

[2] Specify master/slave mode selection and SSCK pin selection.

[3] Selects clock synchronous communication mode and specify transmit/receive data length.

[4] Specify clock polarity selection and transfer clock rate selection.

[5] Enables/disables interrupt request to the CPU.
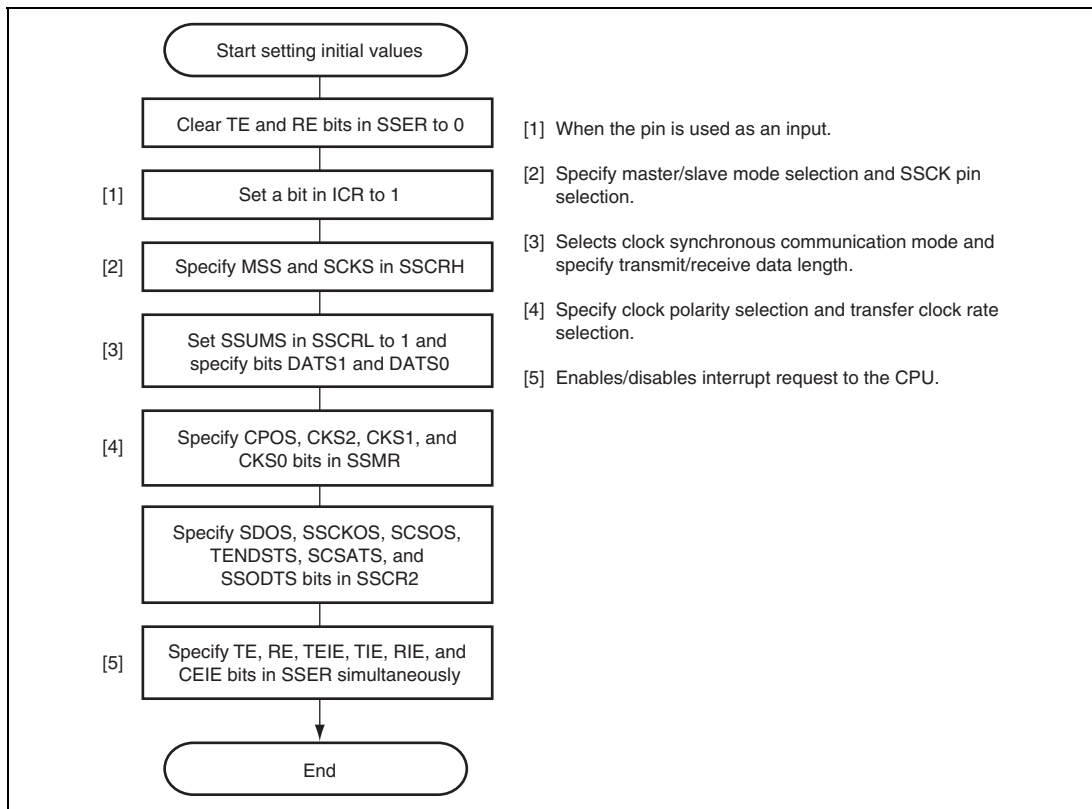
Figure 15.12   Example of Initial Settings in Clock Synchronous Communication Mode

(2)   Data Transmission

Figure 15.13 shows an example of transmission operation, and figure 15.14 shows a flowchart example of data transmission. When transmitting data in clock synchronous communication mode, the SSU operates as shown below.

In master mode, the SSU outputs a transfer clock and data. In slave mode, when a transfer clock is input to the SSCK pin, the SSU outputs data in synchronization with the transfer clock.

Writing transmit data to SSTDR after the initial setting completion clears the TDRE bit in SSSR to 0, and the SSTDR contents are transferred to SSTRSR. After that, the SSU sets the TDRE bit to 1 and starts transmission. At this time, if the TIE bit in SSER is set to 1, a TXI interrupt is generated.

When 1-frame data has been transferred with TDRE = 0, the SSTDR contents are transferred to SSTRSR to start the next frame transmission. When the 8th bit of transmit data has been transferred with TDRE = 1, the TEND bit in SSSR is set to 1 and the state is retained. At this time, if the TEIE bit is set to 1, a TEI interrupt is generated.

While the ORER bit in SSSR is set to 1, transmission is not performed. Check that the ORER bit is cleared to 0.
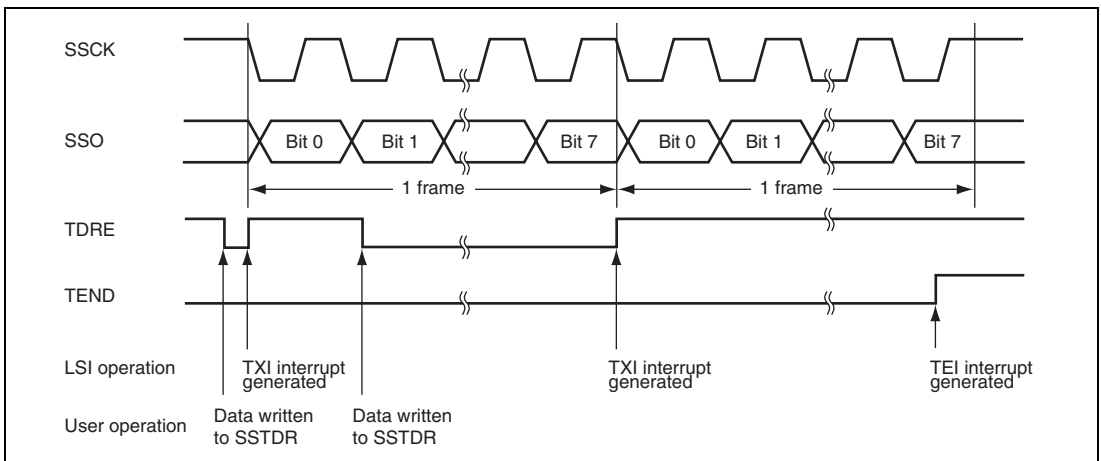
Figure 15.13   Example of Transmission Operation
(Clock Synchronous Communication Mode)

**Figure 15.14   Flowchart Example of Transmission Operation
(Clock Synchronous Communication Mode)**

[1] Initial setting:
Specify the transmit data format.

[2] Check that the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming
that the TDRE bit in SSSR is 1. The TDRE bit is automatically
cleared to 0 and transmission is started by writing data to SSTDR.

[3] Procedure for consecutive data transmission:
To continue data transmission, confirm that the TDRE bit is 1
meaning that SSTDR is ready to be written to. After that, data
can be written to SSTDR. The TDRE bit is automatically
cleared to 0 by writing data to SSTDR.

[4] Procedure for data transmission end:
To end data transmission, confirm that the TEND bit is cleared
to 0. After completion of transmitting the last bit, clear the TE
bit to 0.

(3)   Data Reception

Figure 15.15 shows an example of reception operation, and figure 15.16 shows a flowchart example of data reception. When receiving data, the SSU operates as shown below.

After setting the RE bit in SSER to 1, the SSU starts data reception.

In master mode, the SSU outputs a transfer clock and receives data. In slave mode, when a transfer clock is input to the SSCK pin, the SSU receives data in synchronization with the transfer clock.

When 1-frame data has been received, the RDRF bit in SSSR is set to 1 and the receive data is stored in SSRDR. At this time, if the RIE bit is set to 1, an RXI interrupt is generated. The RDRF bit is automatically cleared to 0 by reading SSRDR.

When the RDRF bit has been set to 1 at the 8th rising edge of the transfer clock, the ORER bit in SSSR is set to 1. This indicates that an overrun error (OEI) has occurred. At this time, data reception is stopped. While the ORER bit in SSSR is set to 1, reception is not performed. To resume the reception, clear the ORER bit to 0.
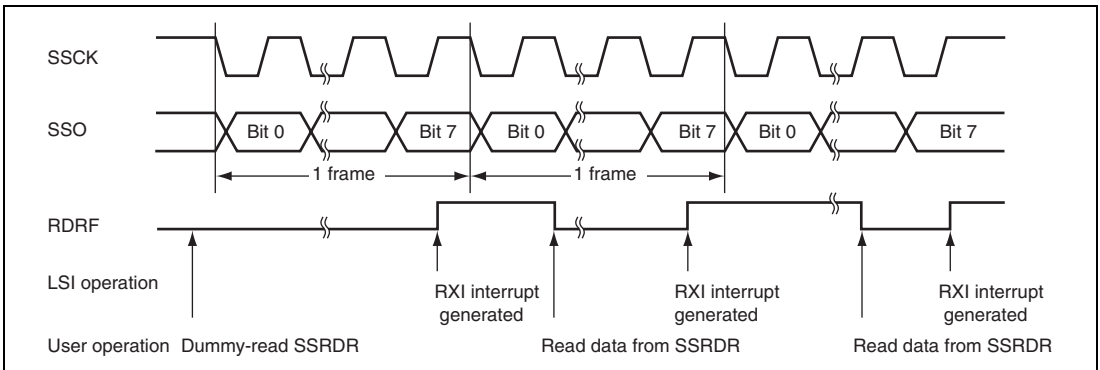
Figure 15.15   Example of Reception Operation
(Clock Synchronous Communication Mode)
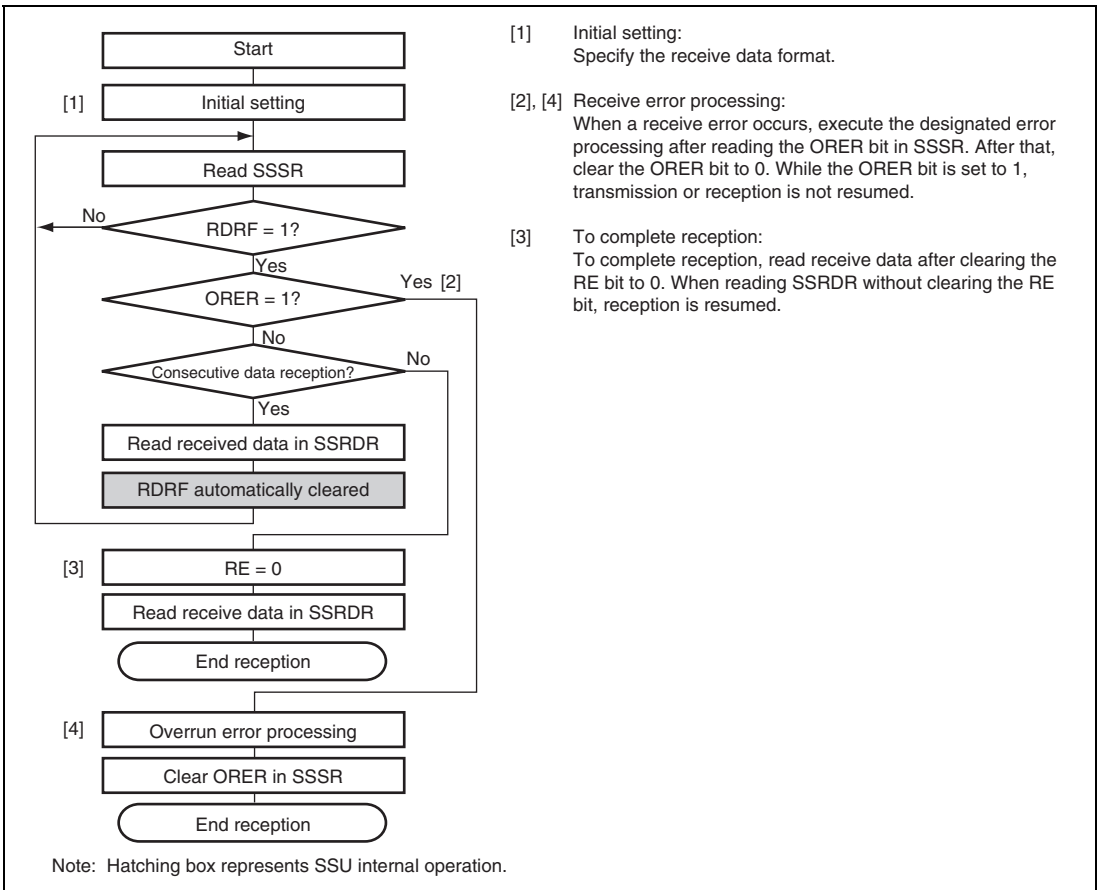
Figure 15.16   Flowchart Example of Data Reception
(Clock Synchronous Communication Mode)

(4)   Data Transmission/Reception

Figure 15.17 shows a flowchart example of simultaneous transmission/reception. The data transmission/reception is performed combining the data transmission and data reception as mentioned above. The data transmission/reception is started by writing transmit data to SSTDR with TE = RE = 1.

Before switching transmission mode (TE = 1) or reception mode (RE = 1) to transmission/reception mode (TE = RE = 1), clear the TE and RE bits to 0. When starting the transfer, confirm that the TEND, RDRF, and ORER bits are cleared to 0 before setting the TE or RE bits to 1.

RENESAS

```
                    Start

   [1]         Initial setting

   [2]       Read TDRE in SSSR.
                                           No
              TDRE = 1?
                  Yes
         Write transmit data to SSTDR

         TDRE automatically cleared

   Data transferred from SSTDR to SSTRSR

    TDRE set to 1 to start transmission

              Read SSSR
   [3]  No
              RDRF = 1?
                  Yes
                                        Yes [4]
              ORER = 1?
                  No
         Read receive data in SSRDR

         RDRF automatically cleared

         Consecutive data              Yes [5]
      transmission/reception?
                  No
            Read TEND in SSSR
                                           No
              TEND = 1?
                  Yes
        Clear TEND in SSSR to 0       Error processing
                                           No
            1-bit period elapsed?
                  Yes
        Clear TE and RE in SSER to 0

          End transmission/reception
```

[1] Initial setting:
    Specify the transmit/receive data format.

[2] Check the SSU state and write transmit data:
    Write transmit data to SSTDR after reading and
    confirming that the TDRE bit in SSSR is 1. The TDRE bit
    is automatically cleared to 0 and transmission is started
    by writing data to SSTDR.

[3] Check the SSU state:
    Read SSSR confirming that the RDRF bit is 1.
    A change of the RDRF bit (from 0 to 1) can be notified
    by RXI interrupt.

[4] Receive error processing:
    When a receive error occurs, execute the designated
    error processing after reading the ORER bit in SSSR.
    After that, clear the ORER bit to 0. While the ORER bit is
    set to 1, transmission or reception is not resumed.

[5] Procedure for consecutive data transmission/reception:
    To continue serial data transmission/reception, confirm
    that the TDRE bit is 1 meaning that SSTDR is ready to be
    written to. After that, data can be written to SSTDR. The
    TDRE bit is automatically cleared to 0 by writing data to
    SSTDR.

Note: Hatching boxes represent SSU internal operations.

**Figure 15.17   Flowchart Example of Simultaneous Transmission/Reception
(Clock Synchronous Communication Mode)**

15.5   Interrupt Requests

The SSU interrupt requests are an overrun error, a conflict error, a receive data register full, transmit data register empty, and a transmit end interrupts.

Since all of these interrupts are assigned to one vector address, the interrupt source must be determined from their flags. Table 15.5 lists the interrupt sources.

When an interrupt condition shown in table 15.5 is satisfied, an interrupt is requested. Clear the interrupt source by CPU data transfer.

Table 15.5   SSU Interrupt Sources

Channel	Abbreviation	Interrupt Source	Symbol	Interrupt Condition
0	SSERI0	Overrun error	OEI0	(RIE = 1) and (ORER = 1)
		Conflict error	CEI0	(CEIE = 1) and (CE = 1)
	SSRXI0	Receive data register full	RXI0	(RIE = 1) and (RDRF = 1)
	SSTXI0	Transmit data register empty	TXI0	(TIE = 1) and (TDRE = 1)
		Transmit end	TEI0	(TEIE = 1) and (TEND = 1)
1	SSERI1	Overrun error	OEI1	(RIE = 1) and (ORER = 1)
		Conflict error	CEI1	(CEIE = 1) and (CE = 1)
	SSRXI1	Receive data register full	RXI1	(RIE = 1) and (RDRF = 1)
	SSTXI1	Transmit data register empty	TXI1	(TIE = 1) and (TDRE = 1)
		Transmit end	TEI1	(TEIE = 1) and (TEND = 1)

Section 16   A/D Converter

This LSI includes two units (unit 0 and unit 1) of successive approximation type 10-bit A/D converters that allow up to 16 analog input channels to be selected.

Figures 16.1 and 16.2 are block diagrams for unit 0 and unit 1, respectively.

This section describes unit 0, which has the same functions as the other unit.

16.1   Features

- 10-bit resolution
- 16 input channels (eight channels for unit 0 and eight channels for unit 1)
- Conversion time: 13.3 μs per channel (at 20-MHz operation)
- Two kinds of operating modes
    — Single mode: Single-channel A/D conversion
    — Scan mode: Continuous A/D conversion on 1 to 4 channels, or 1 to 8 channels
- 16 data registers (eight registers for unit 0 and eight registers for unit 1)
    A/D conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three types of conversion start
    Conversion can be started by software, a conversion start trigger by the 16-bit timer pulse unit (TPU), or an external trigger signal.
- Interrupt source
    A/D conversion end interrupt (ADI) request can be generated.
- Module stop mode can be set
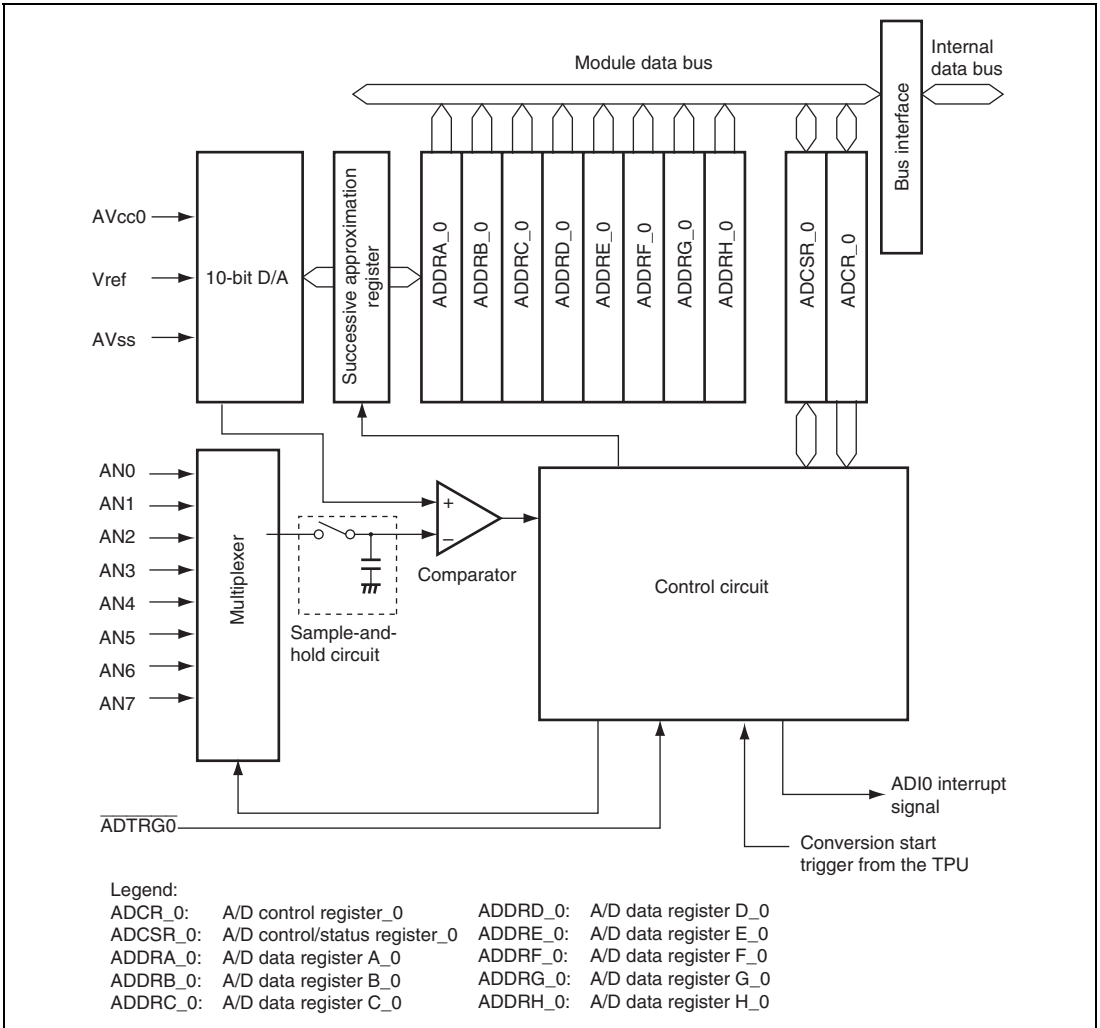
**Figure 16.1   Block Diagram of A/D Converter (Unit 0/AD_0)**
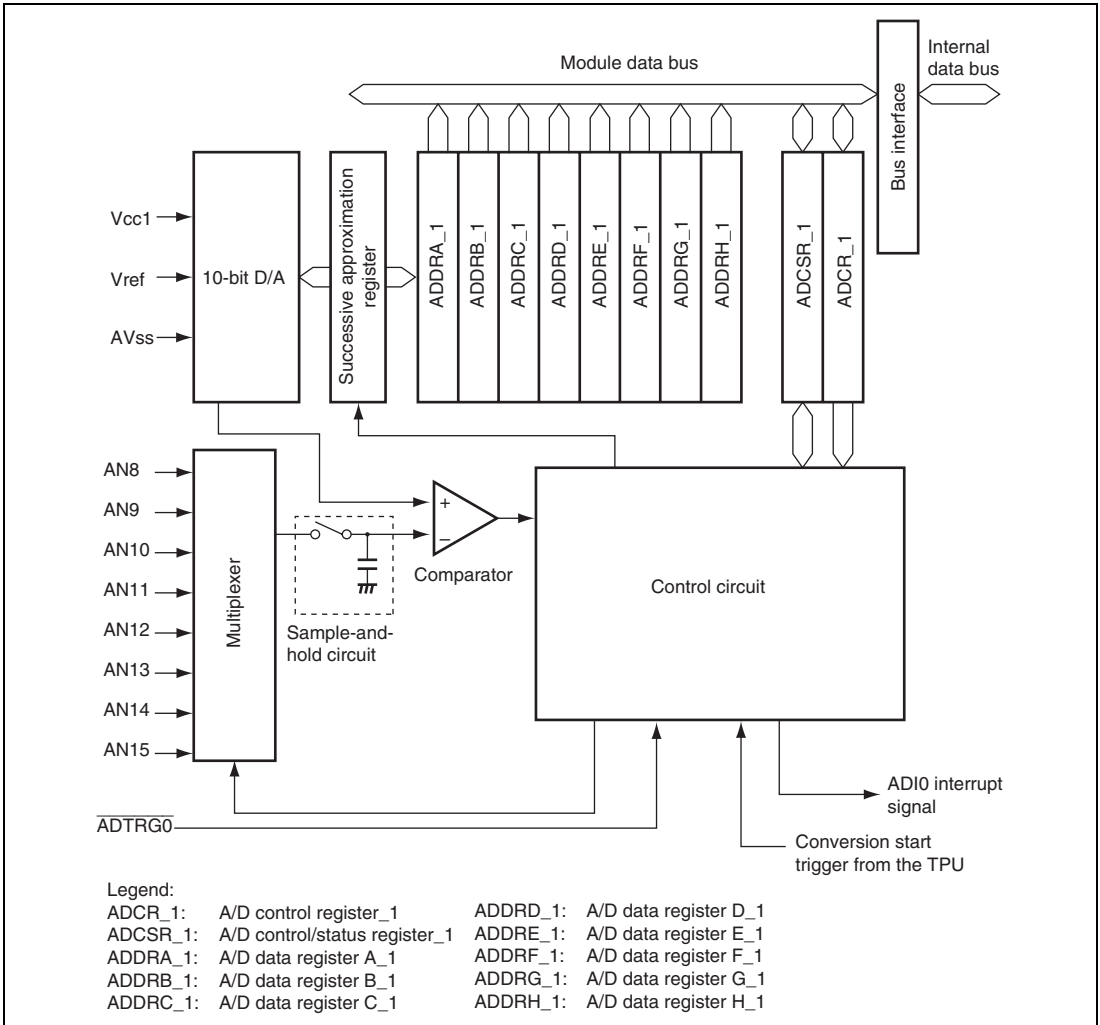
Figure 16.2   Block Diagram of A/D Converter (Unit 1/AD_1)

Legend:
ADCR_1:   A/D control register_1
ADCSR_1:   A/D control/status register_1
ADDRA_1:   A/D data register A_1
ADDRB_1:   A/D data register B_1
ADDRC_1:   A/D data register C_1
ADDRD_1:   A/D data register D_1
ADDRE_1:   A/D data register E_1
ADDRF_1:   A/D data register F_1
ADDRG_1:   A/D data register G_1
ADDRH_1:   A/D data register H_1

16.2   Input/Output Pins

Table 16.1 shows the pin configuration of the A/D converter.

Table 16.1   Pin Configuration

Unit	Abbr.	Pin Name	Symbol	I/O	Function
0	AD_0	Analog input pin 0	AN0	Input	Analog inputs
		Analog input pin 1	AN1	Input	
		Analog input pin 2	AN2	Input	
		Analog input pin 3	AN3	Input	
		Analog input pin 4	AN4	Input	
		Analog input pin 5	AN5	Input	
		Analog input pin 6	AN6	Input	
		Analog input pin 7	AN7	Input	
		A/D external trigger input pin 0	$\overline{\text{ADTRG0}}$	Input	External trigger input for starting A/D conversion
		Analog power supply pin 0	AV_{CC0}	Input	Analog block power supply
1	AD_1	Analog input pin 8	AN8	Input	Analog inputs
		Analog input pin 9	AN9	Input	
		Analog input pin 10	AN10	Input	
		Analog input pin 11	AN11	Input	
		Analog input pin 12	AN12	Input	
		Analog input pin 13	AN13	Input	
		Analog input pin 14	AN14	Input	
		Analog input pin 15	AN15	Input	
		A/D external trigger input pin 1	$\overline{\text{ADTRG1}}$	Input	External trigger input for starting A/D conversion
		Analog power supply pin 1	AV_{CC1}	Input	Analog block power supply
Common		Analog ground pin	AV_{SS}	Input	Analog block ground
		Reference voltage pin	Vref	Input	Reference voltage pin for the A/D converter

RENESAS

16.3   Register Descriptions

The A/D converter has the following registers.

The registers for unit 0 (A/D_0) and unit 1 (A/D_1) have the same functions. In this descriptions, AN8 to AN15 correspond to AN0 to AN7.

- Unit 0 (A/D_0)
  — A/D data register A_0 (ADDRA_0)
  — A/D data register B_0 (ADDRB_0)
  — A/D data register C_0 (ADDRC_0)
  — A/D data register D_0 (ADDRD_0)
  — A/D data register E_0 (ADDRE_0)
  — A/D data register F_0 (ADDRF_0)
  — A/D data register G_0 (ADDRG_0)
  — A/D data register H_0 (ADDRH_0)
  — A/D control/status register_0 (ADCSR_0)
  — A/D control register_0 (ADCR_0)

- Unit 1 (A/D_1)
  — A/D data register A_1 (ADDRA_1)
  — A/D data register B_1 (ADDRB_1)
  — A/D data register C_1 (ADDRC_1)
  — A/D data register D_1 (ADDRD_1)
  — A/D data register E_1 (ADDRE_1)
  — A/D data register F_1 (ADDRF_1)
  — A/D data register G_1 (ADDRG_1)
  — A/D data register H_1 (ADDRH_1)
  — A/D control/status register_1 (ADCSR_1)
  — A/D control register_1 (ADCR_1)

16.3.1   A/D Data Registers A to H (ADDRA to ADDRH)

There are eight 16-bit read-only ADDR registers, ADDRA to ADDRH, used to store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 16.2.

The converted 10-bit data is stored in bits 15 to 6. The lower 6-bit data is always read as 0.

The data bus between the CPU and the A/D converter has a 16-bit width. The data can be read directly from the CPU. ADDR must not be accessed in 8-bit units and must be accessed in 16-bit units.

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

Table 16.2   Analog Input Channels and Corresponding ADDR Registers

Analog Input Channel	A/D Data Register Which Stores Conversion Result
AN0	ADDRA
AN1	ADDRB
AN2	ADDRC
AN3	ADDRD
AN4	ADDRE
AN5	ADDRF
AN6	ADDRG
AN7	ADDRH

16.3.2   A/D Control/Status Register (ADCSR)

ADCSR controls A/D conversion operations.

Bit	7	6	5	4	3	2	1	0
Bit Name	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)∗	R/W	R/W	R	R/W	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W)∗	A/D End Flag
				A status flag that indicates the end of A/D conversion.
				[Setting conditions]
				• When A/D conversion ends in single mode
				• When A/D conversion ends on all specified channels in scan mode
				[Clearing conditions]
				• When 0 is written after reading ADF = 1
				• When the DMAC is activated by an ADI interrupt and ADDR is read
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
6	ADIE	0	R/W	A/D Interrupt Enable
				When this bit is set to 1, ADI interrupts by ADF are enabled.
5	ADST	0	R/W	A/D Start
				Clearing this bit to 0 stops A/D conversion, and the A/D converter enters wait state.
				Setting this bit to 1 starts A/D conversion. In single mode, this bit is cleared to 0 automatically when A/D conversion on the specified channel ends. In scan mode, A/D conversion continues sequentially on the specified channels until this bit is cleared to 0 by software, a reset, or a transition to software standby mode or module stop mode.
4	—	0	R	Reserved
				This is a read-only bit and cannot be modified.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	CH3	0	R/W	Channel Select 3 to 0
2	CH2	0	R/W	Selects analog input together with bits SCANE and SCANS in ADCR.
1	CH1	0	R/W	• When SCANE = 0 and SCANS = ×
0	CH0	0	R/W	

				• When SCANE = 0 and SCANS = ×
				0000: AN0
				0001: AN1
				0010: AN2
				0011: AN3
				0100: AN4
				0101: AN5
				0110: AN6
				0111: AN7
				1×××: Setting prohibited
				• When SCANE = 1 and SCANS = 0
				0000: AN0
				0001: AN0 and AN1
				0010: AN0 to AN2
				0011: AN0 to AN3
				0100: AN4
				0101: AN4 and AN5
				0110: AN4 to AN6
				0111: AN4 to AN7
				1×××: Setting prohibited
				• When SCANE = 1 and SCANS = 1
				0000: AN0
				0001: AN0 and AN1
				0010: AN0 to AN2
				0011: AN0 to AN3
				0100: AN0 to AN4
				0101: AN0 to AN5
				0110: AN0 to AN6
				0111: AN0 to AN7
				1×××: Setting prohibited

Legend:

×: Don't care

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

RENESAS

16.3.3   A/D Control Register (ADCR)

ADCR enables A/D conversion to be started by an external trigger input.

Bit	7	6	5	4	3	2	1	0
Bit Name	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	TRGS1	0	R/W	Timer Trigger Select 1 and 0
6	TRGS0	0	R/W	These bits select enabling or disabling of the start of A/D conversion by a trigger signal.
				00: A/D conversion start by external trigger is disabled
				01: A/D conversion start by external trigger from TPU is enabled
				10: Setting prohibited
				11: A/D conversion start by the $\overline{\text{ADTRG}}$ pin is enabled*
5	SCANE	0	R/W	Scan Mode
4	SCANS	0	R/W	These bits select the A/D conversion operating mode.
				0×: Single mode
				10: Scan mode. A/D conversion is performed continuously for channels 1 to 4.
				11: Scan mode. A/D conversion is performed continuously for channels 1 to 8.
3	CKS1	0	R/W	Clock Select 1 and 0
2	CKS0	0	R/W	These bits set the A/D conversion time. Set bits CKS1 and CKS0 only while A/D conversion is stopped (ADST = 0).
				00: A/D conversion time = 530 states (max)
				01: A/D conversion time = 266 states (max)
				10: A/D conversion time = 134 states (max)
				11: A/D conversion time = 68 states (max)
1, 0	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

Legend:

×: Don't care

16.4   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 in ADCSR to 0 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.

16.4.1   Single Mode

In single mode, A/D conversion is to be performed only once on the analog input of the specified single channel.

1. A/D conversion for the selected channel is started when the ADST bit in ADCSR is set to 1 by software or an external trigger input.
2. When A/D conversion is completed, the A/D conversion result is transferred to the corresponding A/D data register of the channel.
3. When A/D conversion is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when A/D conversion ends. The A/D converter enters wait state. If the ADST bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters wait state.
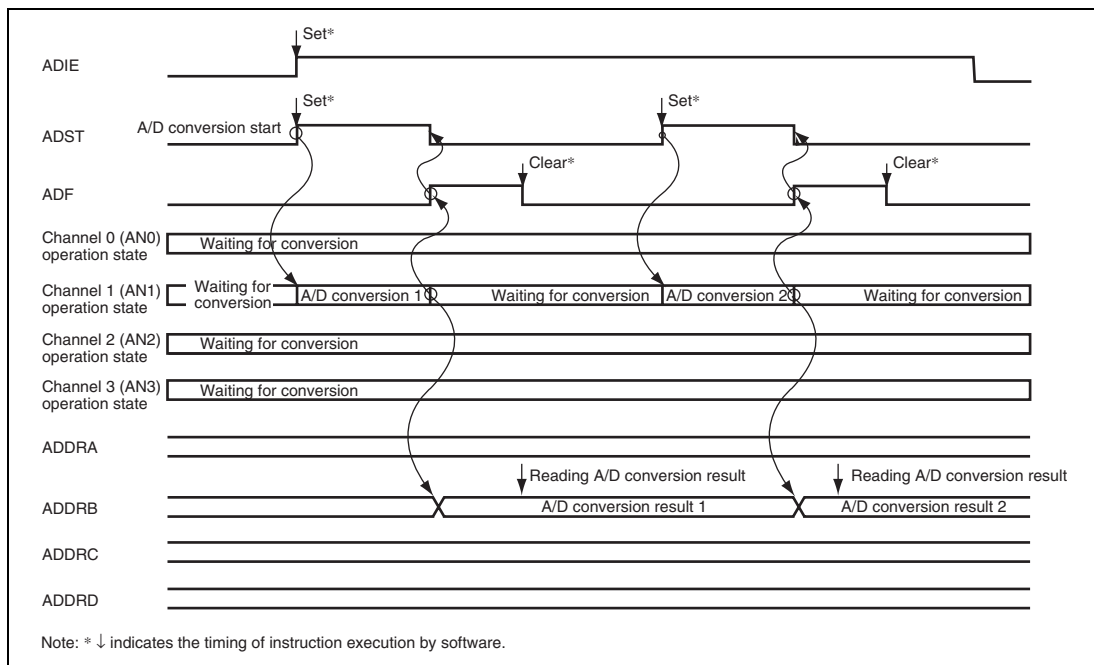
Figure 16.3   Example of A/D Converter Operation (Single Mode, Channel 1 Selected)

16.4.2   Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the analog inputs of the specified channels up to four or eight channels.

1. When the ADST bit in ADCSR is set to 1 by software, TPU, or an external trigger input, A/D conversion starts on the first channel in the group. Consecutive A/D conversion on a maximum of four channels (SCANE and SCANS = B'10) or on a maximum of eight channels (SCANE and SCANS = B'11) can be selected. When consecutive A/D conversion is performed on four channels, A/D conversion starts on AN0 when CH3 and CH2 = B'00 and on AN4 when CH3 and CH2 = B'01. When consecutive A/D conversion is performed on eight channels, A/D conversion starts on AN0 when CH3 = B'0.

2. When A/D conversion for each channel is completed, the A/D conversion result is sequentially transferred to the corresponding ADDR of each channel.

3. When A/D conversion of all selected channels is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. A/D conversion of the first channel in the group starts again.

RENESAS

4. The ADST bit is not cleared automatically, and steps [2] to [3] are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters wait state. If the ADST bit is later set to 1, A/D conversion starts again from the first channel in the group.
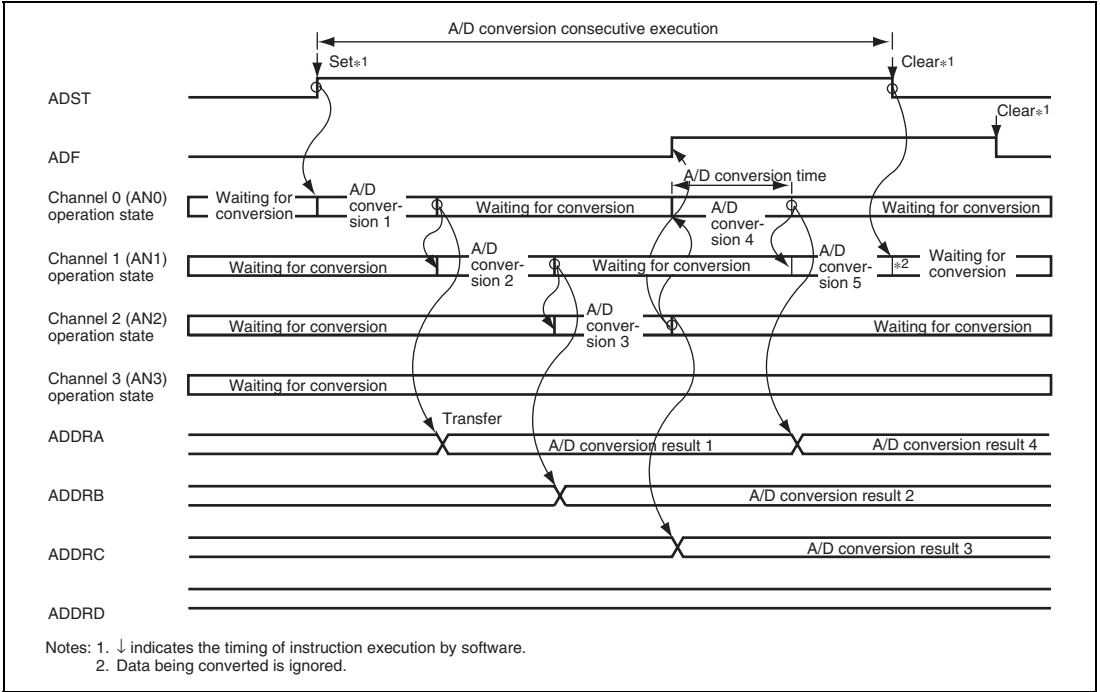
Figure 16.4   Example of A/D Conversion
(Scan Mode, Three Channels (AN0 to AN2) Selected)

16.4.3   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 16.5 shows the A/D conversion timing. Table 16.3 indicates the A/D conversion time.

As indicated in figure 16.5, the A/D conversion time (t_{CONV}) includes t_D and the input sampling time (t_{SPL}). The length of t_D varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 16.3.

In scan mode, the values given in table 16.3 apply to the first conversion time. The values given in table 16.4 apply to the second and subsequent conversions. In either case, bits CKS1 and CKS0 in ADCR should be set so that the conversion time is within the ranges indicated by the A/D conversion characteristics.

Legend:
(1):   ADCSR write cycle
(2):   ADCSR address
t_D: A/D conversion start delay time
t_{SPL}:   Input sampling time
t_{CONV}: A/D conversion time
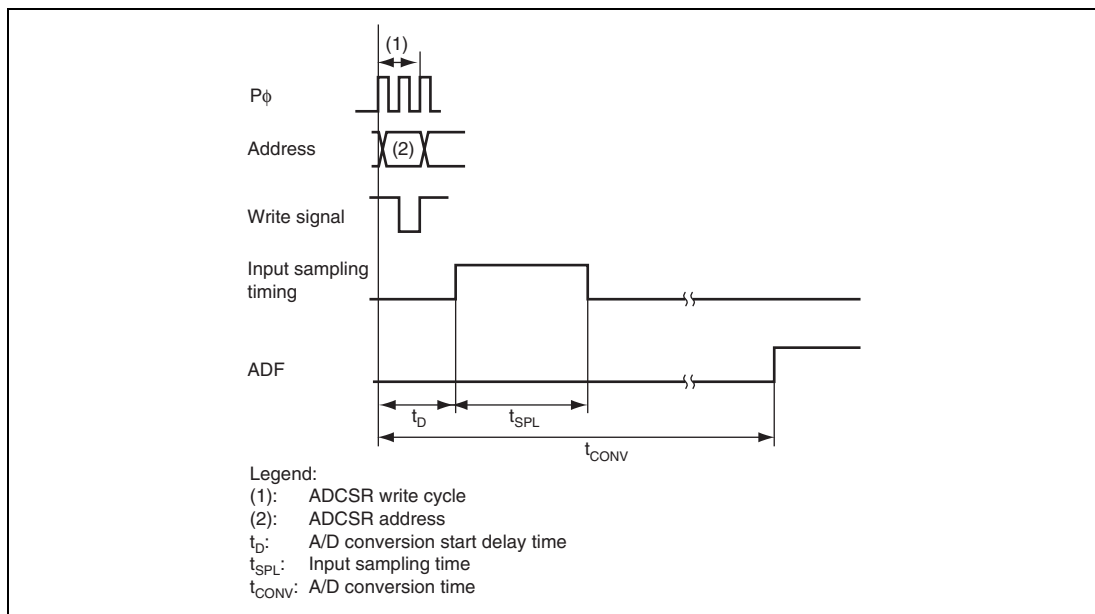
Figure 16.5   A/D Conversion Timing

Table 16.3   A/D Conversion Characteristics (Single Mode)

		CKS1 = 0						CKS1 = 1					
		CKS0 = 0			CKS0 = 1			CKS0 = 0			CKS0 = 1		
Item	Symbol	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.	Min.	Typ.	Max.
A/D conversion start delay time	t_D	18	—	33	10	—	17	6	—	9	4	—	5
Input sampling time	t_{SPL}	—	127	—	—	63	—	—	31	—	—	15	—
A/D conversion time	t_{CONV}	515	—	530	259	—	266	131	—	134	67	—	68

Note: Values in the table are the number of states.

Table 16.4   A/D Conversion Characteristics (Scan Mode)

CKS1	CKS0	Conversion Time (Number of States)
0	0	512 (Fixed)
	1	256 (Fixed)
1	0	128 (Fixed)
	1	64 (Fixed)

16.4.4   External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to B'11 in ADCR, an external trigger is input from the $\overline{\text{ADTRG}}$ pin. A/D conversion starts when the ADST bit in ADCSR is set to 1 on the falling edge of the $\overline{\text{ADTRG}}$ pin. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 16.6 shows the timing.
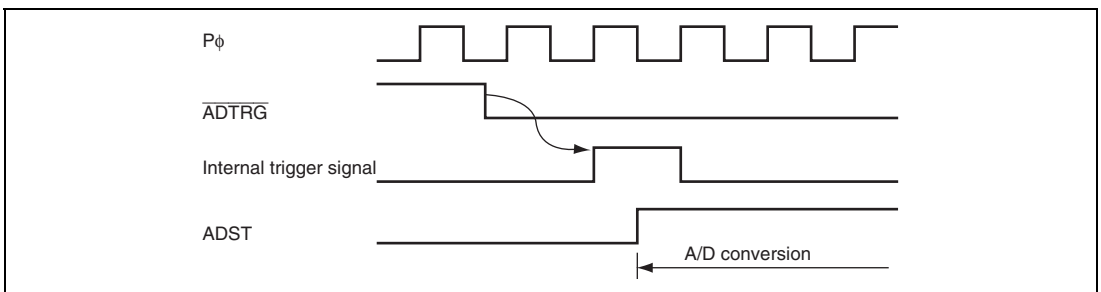
Figure 16.6   External Trigger Input Timing

16.5   Interrupt Sources

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 when the ADF bit in ADCSR is set to 1 after A/D conversion is completed enables ADI interrupt requests. The DMA controller (DMAC) can be activated by an ADI interrupt. Having the converted data read by the DMAC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

Table 16.5   A/D Converter Interrupt Sources

Unit	Abbr.	Interrupt Source	Interrupt Flag	DMAC Activation
0	ADI0	A/D_0 conversion end	ADF	Possible
1	ADI1	A/D_1 conversion end	ADF	Possible

16.6   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 16.7).

- Offset error

    The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'0000000000 (H'000) to B'0000000001 (H'001) (see figure 16.8).

- Full-scale error

    The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 (H'3FE) to B'1111111111 (H'3FF) (see figure 16.8).

- Nonlinearity error

    The error with respect to the ideal A/D conversion characteristic between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error (see figure 16.8).

- Absolute accuracy

    The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.
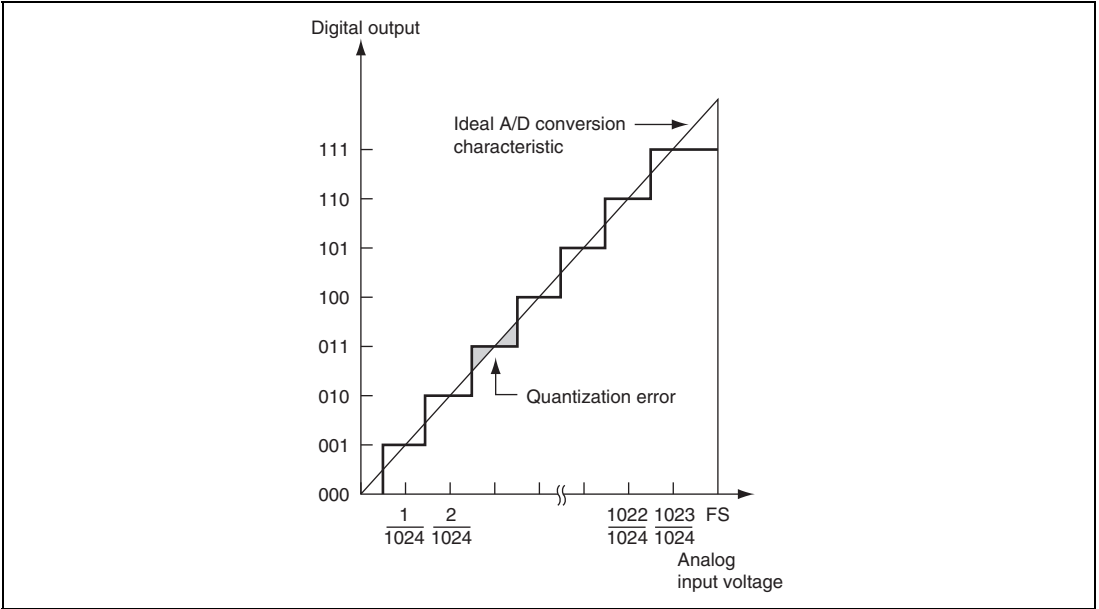
Figure 16.7   A/D Conversion Accuracy Definitions
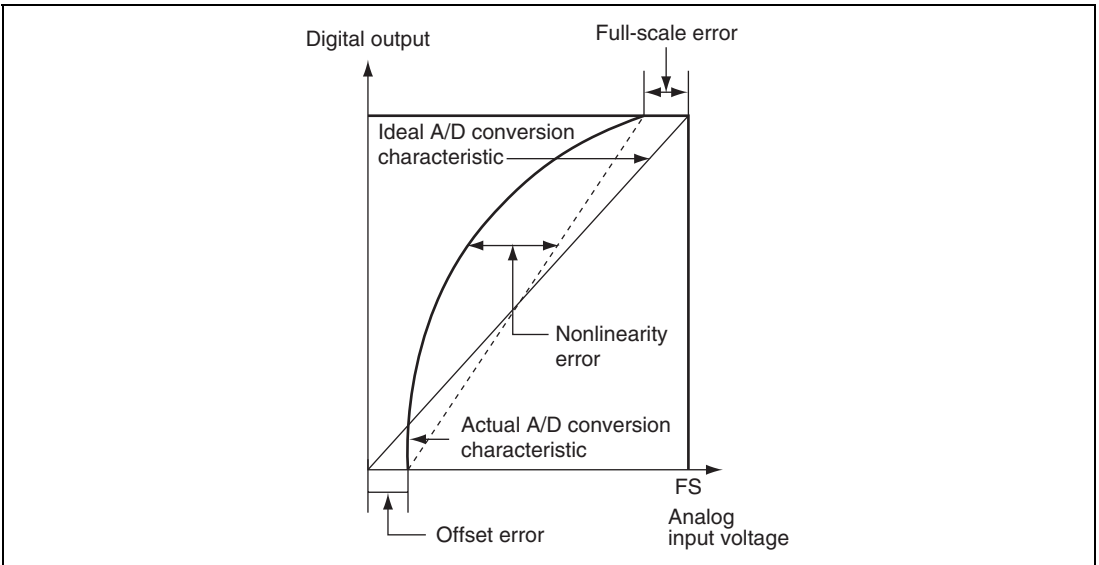
Figure 16.8   A/D Conversion Accuracy Definitions

16.7   Usage Notes

16.7.1   Module Stop Mode Setting

Operation of the A/D converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the A/D converter to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 24, Power-Down Modes.

16.7.2   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 for conversion 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 (e.g., 5 mV/μs or greater) (see figure 16.9). When converting a high-speed analog signal or conversion in scan mode, a low-impedance buffer should be inserted.
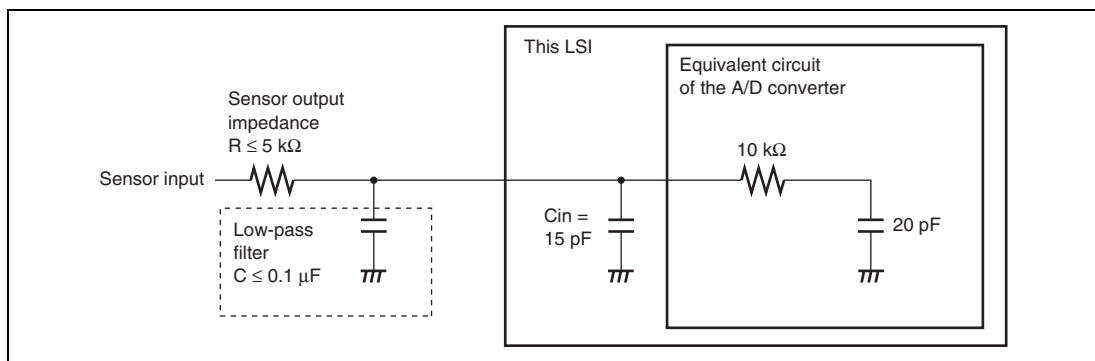

Figure 16.9   Example of Analog Input Circuit

16.7.3   Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute accuracy. Be sure to make the connection to an electrically stable GND such as AVss.

Care is also required to insure that digital signals on the board do not interfere with filter circuits and filter circuits do not act as antennas.

16.7.4   Setting Range of Analog Power Supply and Other Pins

If the conditions shown below are not met, the reliability of the LSI may be adversely affected.

- Analog input voltage range

  The voltage applied to analog input pin ANn during A/D conversion should be in the range AVss ≤ V_{AN} ≤ Vref.

- Relation between AVcc0, AVcc1, AVss and Vcc, Vss

  As the relationship between AVcc0, AVcc1, AVss and Vcc, Vss, set AVcc0 = Vcc ± 0.3 V, AVcc1 = Vcc ± 0.3 V, and AVss = Vss. If the A/D converter is not used, set AVcc0 = Vcc, AVcc1 = Vcc, and AVss = Vss.

- Vref range

  The reference voltage applied to the Vref pin should be in the range Vref ≤ AVcc0 and Vref ≤ AVcc1.

16.7.5   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.

Digital circuitry must be isolated from the analog input pins (AN0 to AN15), analog reference power supply (Vref), and analog power supply (AVcc0 and AVcc1) by the analog ground (AVss). Also, the analog ground (AVss) should be connected at one point to a stable ground (Vss) on the board.

RENESAS

16.7.6   Notes on Noise Countermeasures

A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN15) should be connected between AVcc0, AVcc1 and AVss as shown in figure 16.10. Also, the bypass capacitors connected to AVcc0 and AVcc1 and the filter capacitor connected to pins AN0 to AN15 must be connected to AVss.

If a filter capacitor is connected, the input currents at pins AN0 to AN15 are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance (R_{in}), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.
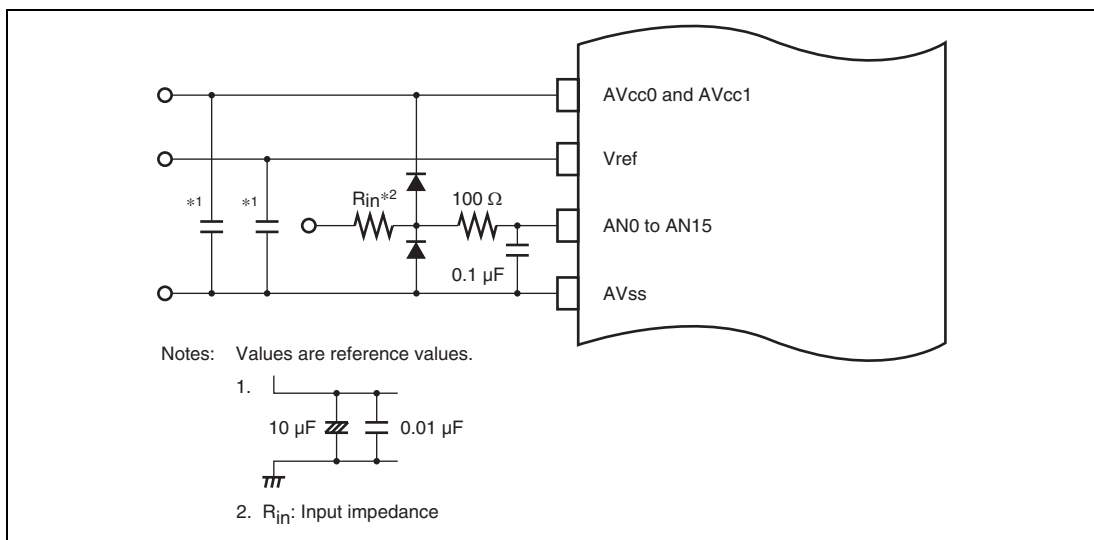
Figure 16.10   Example of Analog Input Protection Circuit

Table 16.6   Analog Pin Specifications

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Permissible signal source impedance	—	5	kΩ
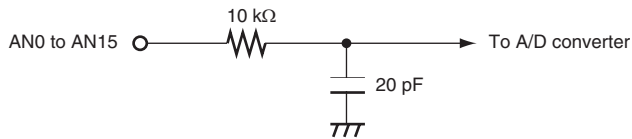
Note: Values are reference values.

Figure 16.11   Analog Input Pin Equivalent Circuit

16.7.7   A/D Input Hold Function in Software Standby Mode

When this LSI enters software standby mode with A/D conversion enabled, the A/D conversion
are retained, and the analog current is equal to as during A/D conversion. If the analog power
supply current needs to be reduced in software standby mode, clear the ADST, TRGS1, and
TRGS0 bits all to 0 to disable A/D conversion.

Section 17   D/A Converter

17.1   Features

- 8-bit resolution
- Two output channels
- Maximum conversion time of 10 μs (with 20-pF load)
- Output voltage of 0 V to V_{ref}
- D/A output hold function in software standby mode
- Module stop mode can be set
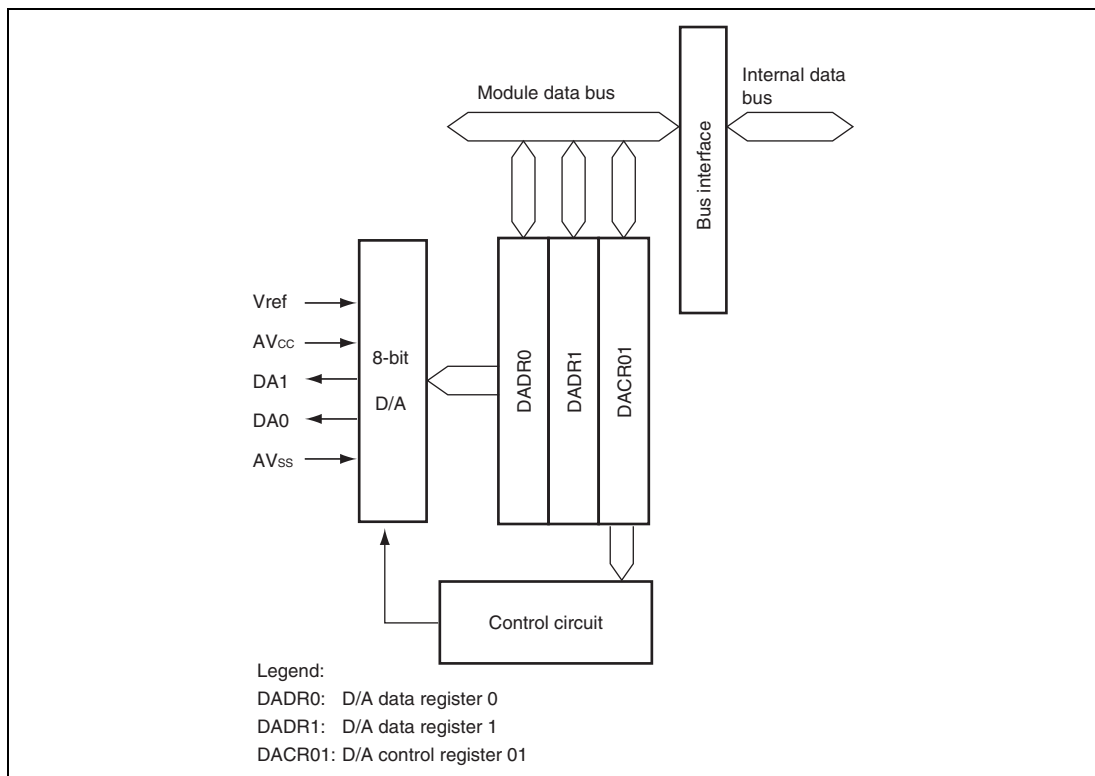
Figure 17.1   Block Diagram of D/A Converter

17.2   Input/Output Pins

Table 17.1 shows the pin configuration of the D/A converter.

Table 17.1   Pin Configuration

Pin Name	Symbol	I/O	Function
Analog power supply pin	AVcc	Input	Analog block power supply
Analog ground pin	AVss	Input	Analog block ground
Reference voltage pin	Vref	Input	D/A conversion reference voltage
Analog output pin 0	DA0	Output	Channel 0 analog output
Analog output pin 1	DA1	Output	Channel 1 analog output

17.3   Register Descriptions

The D/A converter has the following registers.

- D/A data register 0 (DADR0)
- D/A data register 1 (DADR1)
- D/A control register 01 (DACR01)

17.3.1   D/A Data Registers 0 and 1 (DADR0 and DADR1)

DADR0 and DADR1 are 8-bit readable/writable registers that store data to which D/A conversion is to be performed. Whenever an analog output is enabled, the values in DADR are converted and output to the analog output pins.

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

RENESAS

17.3.2   D/A Control Register 01 (DACR01)

DACR01 controls the operation of the D/A converter.

Bit	7	6	5	4	3	2	1	0
Bit Name	DAOE1	DAOE0	DAE	—	—	—	—	—
Initial Value	0	0	0	1	1	1	1	1
R/W	R/W	R/W	R/W	R	R	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	DAOE1	0	R/W	D/A Output Enable 1
				Controls D/A conversion and analog output.
				0: Analog output of channel 1 (DA1) is disabled
				1: D/A conversion of channel 1 is enabled. Analog output of channel 1 (DA1) is enabled.
6	DAOE0	0	R/W	D/A Output Enable 0
				Controls D/A conversion and analog output.
				0: Analog output of channel 0 (DA0) is disabled
				1: D/A conversion of channel 0 is enabled. Analog output of channel 0 (DA0) is enabled.
5	DAE	0	R/W	D/A Enable
				Used together with the DAOE0 and DAOE1 bits to control D/A conversion. When this bit is cleared to 0, D/A conversion is controlled independently for channels 0 and 1. When this bit is set to 1, D/A conversion for channels 0 and 1 is controlled together.
				Output of conversion results is always controlled by the DAOE0 and DAOE1 bits. For details, see table 17.2.
4 to 0	—	All 1	R	Reserved
				These are read-only bits and cannot be modified.

Table 17.2   Control of D/A Conversion

Bit 5 DAE	Bit 7 DAOE1	Bit 6 DAOE0	Description
0	0	0	D/A conversion is disabled.
		1	D/A conversion of channel 0 is enabled and D/A conversion of channel 1 is disabled.
			Analog output of channel 0 (DA0) is enabled and analog output of channel 1 (DA1) is disabled.
	1	0	D/A conversion of channel 0 is disabled and D/A conversion of channel 1 is enabled.
			Analog output of channel 0 (DA0) is disabled and analog output of channel 1 (DA1) is enabled.
		1	D/A conversion of channels 0 and 1 is enabled.
			Analog output of channels 0 and 1 (DA0 and DA1) is enabled.
1	0	0	D/A conversion of channels 0 and 1 is enabled.
			Analog output of channels 0 and 1 (DA0 and DA1) is disabled.
		1	D/A conversion of channels 0 and 1 is enabled.
			Analog output of channel 0 (DA0) is enabled and analog output of channel 1 (DA1) is disabled.
	1	0	D/A conversion of channels 0 and 1 is enabled.
			Analog output of channel 0 (DA0) is disabled and analog output of channel 1 (DA1) is enabled.
		1	D/A conversion of channels 0 and 1 is enabled.
			Analog output of channels 0 and 1 (DA0 and DA1) is enabled.

RENESAS

17.4   Operation

The D/A converter includes D/A conversion circuits for two channels, each of which can operate independently. When the DAOE bit in DACR01 is set to 1, D/A conversion is enabled and the conversion result is output.

An operation example of D/A conversion on channel 0 is shown below. Figure 17.2 shows the timing of this operation.

1. Write the conversion data to DADR0.
2. Set the DAOE0 bit in DACR01 to 1 to start D/A conversion. The conversion result is output from the analog output pin DA0 after the conversion time t_{DCONV} has elapsed. The conversion result continues to be output until DADR0 is written to again or the DAOE0 bit is cleared to 0. The output value is expressed by the following formula:

    Contents of DADR/256 $\times$ V_{ref}

3. If DADR0 is written to again, the conversion is immediately started. The conversion result is output after the conversion time t_{DCONV} has elapsed.
4. If the DAOE0 bit is cleared to 0, analog output is disabled.
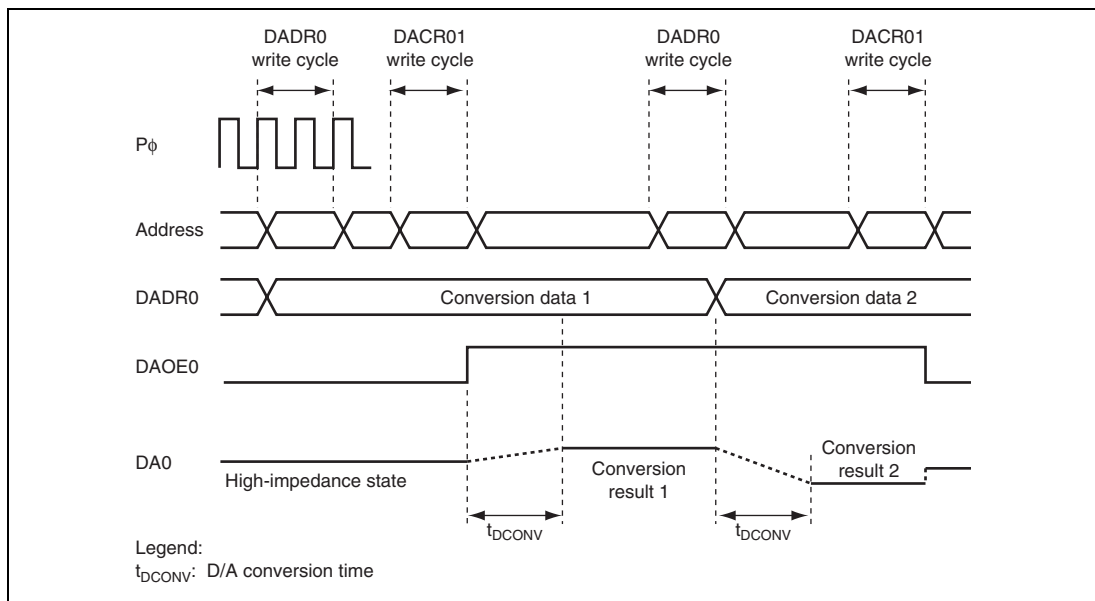
Figure 17.2   Example of D/A Converter Operation

17.5　　Usage Notes

17.5.1　　Module Stop Mode Setting

Operation of the D/A converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the D/A converter to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 24, Power-Down Modes.

17.5.2　　D/A Output Hold Function in Software Standby Mode

When this LSI enters software standby mode with D/A conversion enabled, the D/A outputs are retained, and the analog power supply current is equal to as during D/A conversion. If the analog power supply current needs to be reduced in software standby mode, clear the ADST, TRGS1, and TRGS0 bits all to 0 to disable D/A conversion.

RENESAS

Section 18   Motor Control PWM Timer (PWM)

This LSI has two channels of on-chip motor control PWM (pulse width modulator) with a maximum capability of 16 pulse outputs in total.

18.1   Features

- Maximum of 16 pulse outputs
  — Two 10-bit PWM channels, each with eight outputs.
  — 10-bit counter (PWCNT) and cycle register (PWCYR).
  — Duty and output polarity can be set for each output.
- Automatic data transfer in every cycle
  — Each of four duty registers (PWDTR) is provided with buffer registers (PWBFR), with data transferred automatically every cycle.
- Duty settings selectable
  — A duty cycle of 0% to 100% can be selected by means of a duty register setting.
- Counting clock selectable
  — There is a choice of five counting clocks (Pϕ, Pϕ/2, Pϕ/4, Pϕ/8, Pϕ/16).
- High-speed access via internal 16-bit bus
- Two interrupt sources
  — An interrupt can be requested independently for each channel by a cycle register compare match.
- Automatic transfer of register data
  — Block transfer and one-word data transfer are available by activating the DMAC.
- On-chip output driver
  — IOL/IOH: 15 mA typ. and 30 mA max.
  — Total IOL/IOH: 120 mA max. (target value)
- Module stop mode can be set

Figure 18.1 shows a block diagram of the PWM.
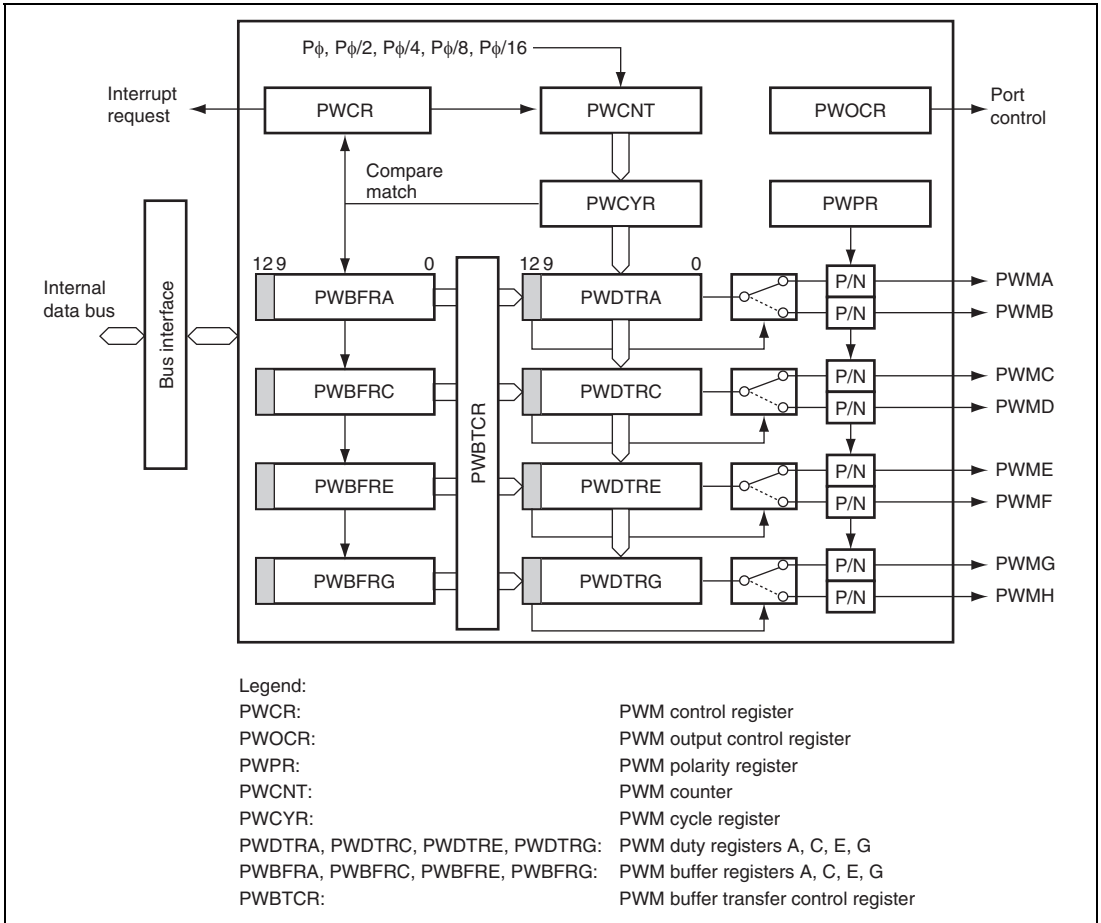
Figure 18.1   Block Diagram of PWM

18.2   Input/Output Pins

Table 18.1 shows the PWM pin configuration.

Table 18.1   Pin Configuration

Channel	Name	Abbrev.	I/O	Function
1	PWM output pin 1A	PWM1A	Output	Channel 1A PWM output
	PWM output pin 1B	PWM1B	Output	Channel 1B PWM output
	PWM output pin 1C	PWM1C	Output	Channel 1C PWM output
	PWM output pin 1D	PWM1D	Output	Channel 1D PWM output
	PWM output pin 1E	PWM1E	Output	Channel 1E PWM output
	PWM output pin 1F	PWM1F	Output	Channel 1F PWM output
	PWM output pin 1G	PWM1G	Output	Channel 1G PWM output
	PWM output pin 1H	PWM1H	Output	Channel 1H PWM output
2	PWM output pin 2A	PWM2A	Output	Channel 2A PWM output
	PWM output pin 2B	PWM2B	Output	Channel 2B PWM output
	PWM output pin 2C	PWM2C	Output	Channel 2C PWM output
	PWM output pin 2D	PWM2D	Output	Channel 2D PWM output
	PWM output pin 2E	PWM2E	Output	Channel 2E PWM output
	PWM output pin 2F	PWM2F	Output	Channel 2F PWM output
	PWM output pin 2G	PWM2G	Output	Channel 2G PWM output
	PWM output pin 2H	PWM2H	Output	Channel 2H PWM output

18.3   Register Descriptions

The PWM has the following registers for each channel.

- PWM control register (PWCR)
- PWM output control register (PWOCR)
- PWM polarity register (PWPR)
- PWM counter (PWCNT)
- PWM cycle register (PWCYR)
- PWM duty register A (PWDTRA)
- PWM duty register C (PWDTRC)
- PWM duty register E (PWDTRE)
- PWM duty register G (PWDTRG)
- PWM buffer register A (PWBFRA)
- PWM buffer register C (PWBFRC)
- PWM buffer register E (PWBFRE)
- PWM buffer register G (PWBFRG)
- PWM buffer transfer control register (PWBTCR)

RENESAS

18.3.1   PWM Control Register (PWCR)

PWCR performs interrupt control, starting/stopping of the counter, and counter clock selection. It also contains a flag that indicates a compare match with PWCYR.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	IE	CMF	CST	CKS2	CKS1	CKS0
Initial Value	1	1	0	0	0	0	0	0
R/W	—	—	R/W	R/(W)*	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	—	Reserved
				These bits are always read as 1 and cannot be modified.
5	IE	0	R/W	Interrupt Enable
				Enables or disables an interrupt request in the event of a compare match with PWCYR of the corresponding channel.
				0: Interrupt disabled
				1: Interrupt enabled
4	CMF	0	R/(W)*	Compare Match Flag
				Indicates the occurrence of a compare match with PWCYR of the corresponding channel.
				[Setting condition]
				When PWCNT = PWCYR
				[Clearing conditions]
				• When 0 is written to CMF after reading CMF = 1
				• When the DMAC is activated by a compare match interrupt, and the DTA bit in DMDR of the DMAC is 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	CST	0	R/W	Counter Start
				Selects starting or stopping of PWCNT of the corresponding channel.
				0: PWCNT is stopped
				1: PWCNT is started

Bit	Bit Name	Initial Value	R/W	Description
2	CKS2	0	R/W	Clock Select
1	CKS1	0	R/W	These bits select the operating clock for PWCNT of the corresponding channel.
0	CKS0	0	R/W	
				000: Counts on Pφ/1
				001: Counts on Pφ/2
				010: Counts on Pφ/4
				011: Counts on Pφ/8
				1XX: Counts on Pφ/16

Legend:

X: Don't care

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

18.3.2   PWM Output Control Register (PWOCR)

PWOCR enables or disables PWM output.

Bit	7	6	5	4	3	2	1	0
Bit Name	OEnH	OEnG	OEnF	OEnE	OEnD	OEnC	OEnB	OEnA
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	OEnH	0	R/W	Output Enable
6	OEnG	0	R/W	Each of these bits enables or disables the corresponding PWM output.
5	OEnF	0	R/W	
4	OEnE	0	R/W	0: PWM output disabled
3	OEnD	0	R/W	1: PWM output enabled
2	OEnC	0	R/W	
1	OEnB	0	R/W	
0	OEnA	0	R/W	

(n = 1, 2)

RENESAS

18.3.3   PWM Polarity Register (PWPR)

PWPR selects the PWM output polarity.

Bit	7	6	5	4	3	2	1	0
Bit Name	OPSnH	OPSnG	OPSnF	OPSnE	OPSnD	OPSnC	OPSnB	OPSnA
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	OPSnH	0	R/W	Output Polarity Select
6	OPSnG	0	R/W	Each of these bits selects the PWM output polarity.
5	OPSnF	0	R/W	0: PWM direct output
4	OPSnE	0	R/W	1: PWM inverse output
3	OPSnD	0	R/W	
2	OPSnC	0	R/W	
1	OPSnB	0	R/W	
0	OPSnA	0	R/W	

(n = 1, 2)

18.3.4   PWM Counter (PWCNT)

PWCNT is a 10-bit up-counter incremented by the input clock. The input clock is selected by clock select bits CKS2 to CKS0 in PWCR.

PWCNT can not be directly accessed by the CPU. PWCNT is initialized to H'FC00, when CST bit in PWCR is 0.

18.3.5   PWM Cycle Register (PWCYR)

PWCYR is a 16-bit readable/writable register that sets the PWM conversion cycle. When a PWCYR compare match occurs, PWCNT is cleared and data is transferred from the buffer register (PWBFR) to the duty register (PWDTR).

PWCYR should be written to only while PWCNT is stopped. A value of H'FC00 must not be set. PWCYR is initialized to H'FFFF.
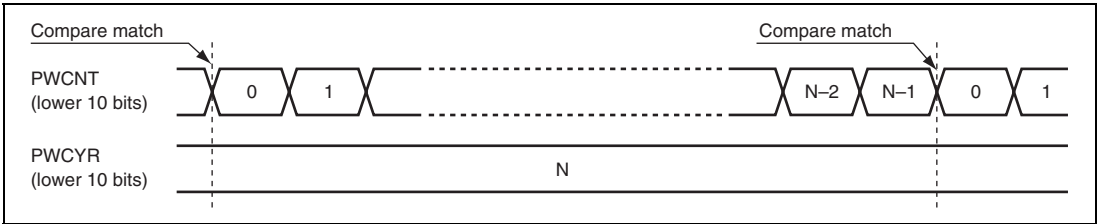
Figure 18.2   Cycle Register Compare Match

18.3.6   PWM Duty Registers A, C, E, G (PWDTRA, PWDTRC, PWDTRE, PWDTRG)

There are four PWDTR registers (PWDTRA, PWDTRC, PWDTRE, and PWDTRG). The PWDTRA is used for outputs PWMA and PWMB, PWDTRC for outputs PWMC and PWMD, PWDTRE for outputs PWME and PWMF, and PWDTRG for outputs PWMG and PWMH.

PWDTR can not be directly accessed by the CPU. When a PWCYR compare match occurs, data is transferred from the buffer register (PWBFR) to the duty register (PWDTR). PWDTR is initialized to H'00 when CST bit is 0.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	OTS	—	—	DT9	DT8
Initial Value	—	—	—	0	—	—	0	0
R/W	—	—	—	—	—	—	—	—

Bit	7	6	5	4	3	2	1	0
Bit Name	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Initial Value	0	0	0	0	0	0	0	0
R/W	—	—	—	—	—	—	—	—

Bit	Bit Name	Initial Value	R/W	Description
15 to 13	—	—	—	Reserved
12	OTS	0	—	Output Terminal Select
				Selects the pin used for PWM output. Unselected pins output a low level (or a high level when the corresponding bit in PWPR is set to 1). For details, see table 18.2.
11, 10	—	—	—	Reserved
9	DT9	0	—	Duty
8	DT8	0	—	These bits specify the PWM output duty. A high level (or a low level when the corresponding bit in PWPR is set to 1) is output from the time PWCNT is cleared by a PWCYR compare match until a PWDTR compare match occurs. When all of the bits are 0, there is no high-level (or low-level when the corresponding bit in PWPR is set to 1) output period.
7	DT7	0	—	
6	DT6	0	—	
5	DT5	0	—	
4	DT4	0	—	
3	DT3	0	—	
2	DT2	0	—	
1	DT1	0	—	
0	DT0	0	—	

Table 18.2   Output Selection by OTS Bit

Register	Bit 12 OTS	Description
PWDTR1A/	0	PWMA output selected
PWDTR2A	1	PWMB output selected
PWDTR1C/	0	PWMC output selected
PWDTR2C	1	PWMD output selected
PWDTR1E/	0	PWME output selected
PWDTR2E	1	PWMF output selected
PWDTR1G/	0	PWMG output selected
PWDTR2G	1	PWMH output selected
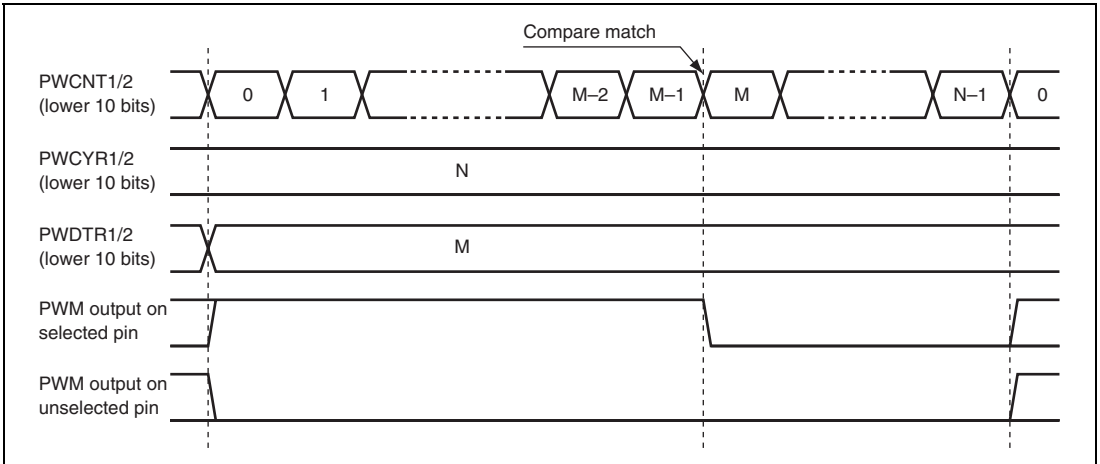
Figure 18.3   Duty Register Compare Match (OPS = 0 in PWPR)
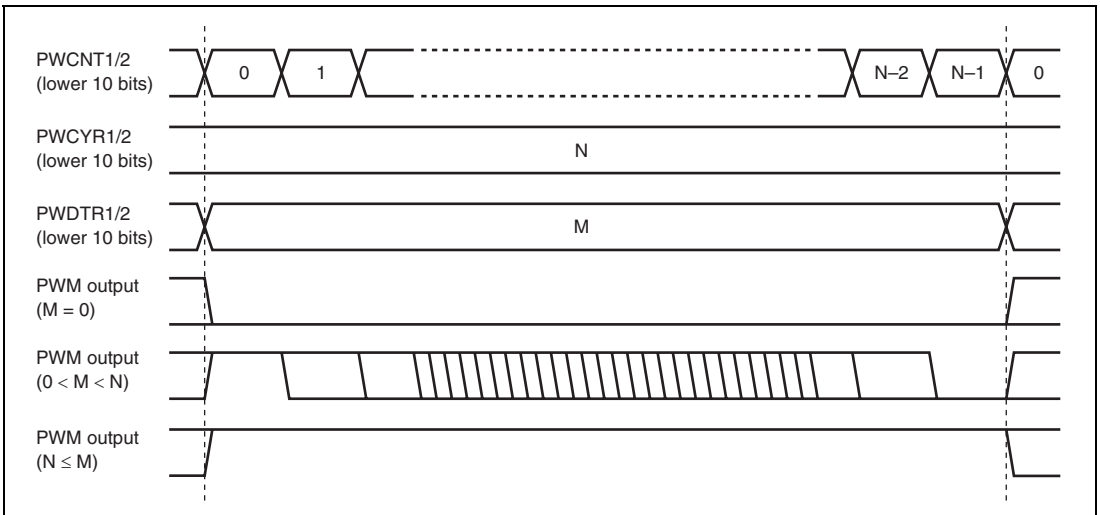
**Figure 18.4   Differences in PWM Output According to Duty Register Set Value
(OPS = 0 in PWPR)**

18.3.7   PWM Buffer Registers A, C, E, G (PWBFRA, PWBFRC, PWBFRE, PWBFRG)

There are four PWBFR registers (PWBFRA, PWBFRC, PWBFRE, and PWBFRG). When a PWCYR compare match occurs, data is transferred from the buffer register (PWBFR) to the duty register (PWDTR).

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	OTS	—	—	DT9	DT8
Initial Value	1	1	1	0	1	1	0	0
R/W	—	—	—	R/W	—	—	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15 to 13	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
12	OTS	0	R/W	Output Terminal Select Holds the data to be sent to bit 12 in PWDTR.
11, 10	—	All 1	—	Reserved These bits are always read as 1 and cannot be modified.
9	DT9	0	R/W	Duty
8	DT8	0	R/W	These bits hold the data to be sent to bits 9 to 0 in PWDTR.
7	DT7	0	R/W	
6	DT6	0	R/W	
5	DT5	0	R/W	
4	DT4	0	R/W	
3	DT3	0	R/W	
2	DT2	0	R/W	
1	DT1	0	R/W	
0	DT0	0	R/W	

18.3.8   PWM Buffer Transfer Control Register (PWBTCR)

PWBTCR enables or disables the data transfer from buffer register to duty register with the compare match of PWM counter and PWM cycle register.

Bit	7	6	5	4	3	2	1	0
Bit Name	BTC2G	BTC2E	BTC2C	BTC2A	BTC1G	BTC1E	BTC1C	BTC1A
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	BTC2G	0	R/W	0: Data transfer from PWBFR to PWDTR enabled with PWCNT and PWCYR compare match
6	BTC2E	0	R/W	
5	BTC2C	0	R/W	1: Data transfer from PWBFR to PWDTR disabled with PWCNT and PWCYR compare match
4	BTC2A	0	R/W	
3	BTC1G	0	R/W	
2	BTC1E	0	R/W	
1	BTC1C	0	R/W	
0	BTC1A	0	R/W	

18.4   Bus Master Interface

18.4.1   16-Bit Data Registers

PWCYR and PWBFR are 16-bit registers. These registers are linked to the bus master by a 16-bit data bus, and can be read or written in 16-bit units. They cannot be read or written by 8-bit access; 16-bit access must always be used.
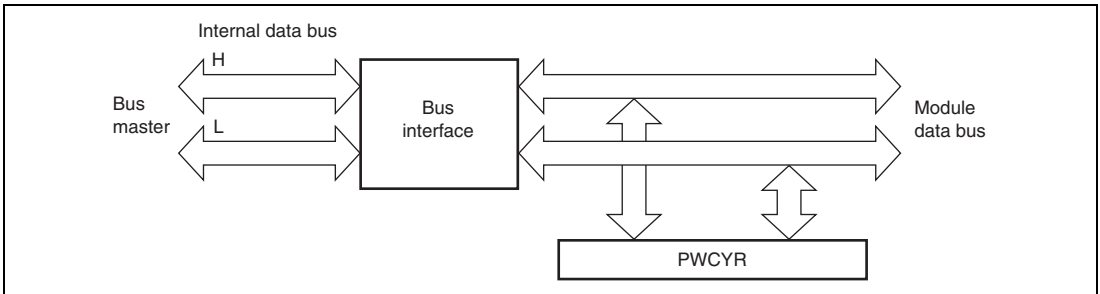
Figure 18.5   16-Bit Register Access Operation (Bus Master ↔ PWCYR (16 Bits))

18.4.2   8-Bit Data Registers

PWCR, PWOCR, PWPR, and PWBTCR are 8-bit registers that can be read and written to in 8-bit units. These registers are linked to the bus master by a 16-bit data bus, and can be read or written by 16-bit access; in this case, the lower eight bits are read as H'FF.
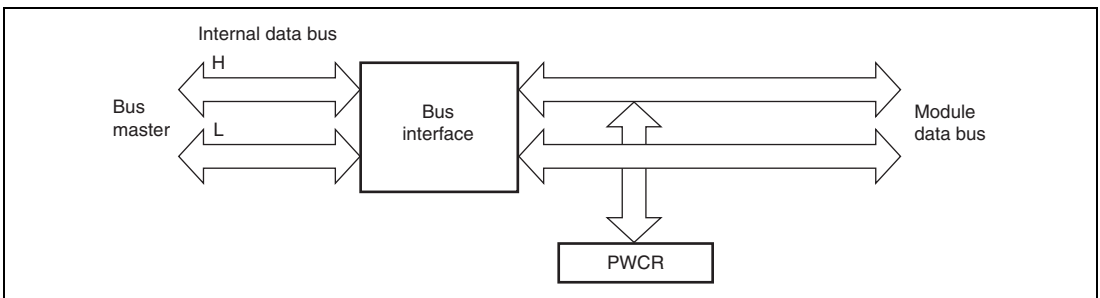
Figure 18.6   8-Bit Register Access Operation (Bus Master ↔ PWCR (Upper Eight Bits))

18.5   Operation

18.5.1   PWM Operation

PWM waveforms are output from pins PWM1A to PWM1H and PWM2A to PWM2H as shown in figure 18.7.

(1)   Initial Settings

Set the PWM output polarity in PWPR; set the OEn bit in PWOCR to 1 to enable PWM output from the corresponding pin; select the clock to be input to PWCNT with the CKS2 to CKS0 bits in PWCR; set the PWM conversion cycle in PWCYR; and set the first frame of data in PWBFRA, PWBFRC, PWBFRE, and PWBFRG.

(2)   Activation

When the CST bit in PWCR is set to 1, PWCNT starts counting up. On compare match between PWCNT and PWCYR, data is transferred from the buffer register to the duty register and the CMF bit in PWCR is set to 1. At the same time, if the IE bit in PWCR has been set to 1, an interrupt can be requested or the DMAC can be activated.

(3)   Waveform Output

The PWM outputs selected by the OTS bits in PWDTRA, PWDTRC, PWDTRE, and PWDTRG go high when a compare match occurs between PWCNT and PWCYR. The PWM outputs not selected by the OTS bit are low. When a compare match occurs between PWCNT and PWDTRA, PWDTRC, PWDTRE, or PWDTRG, the corresponding PWM output goes low. If the corresponding bit in PWPR is set to 1, the output is inverted.
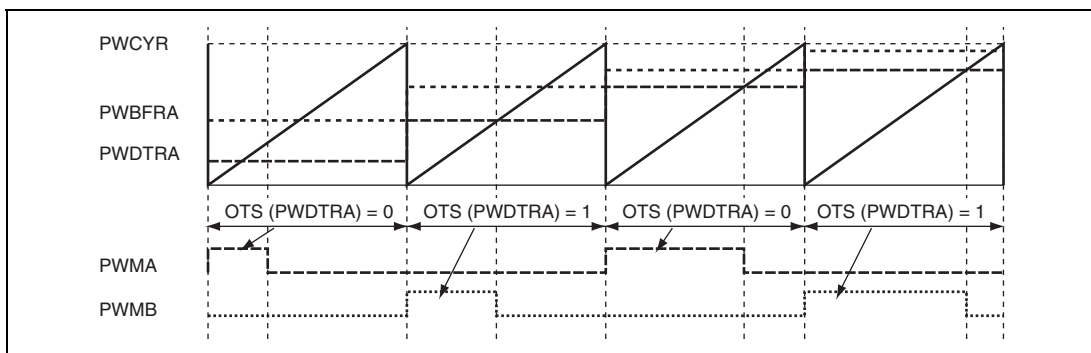
Figure 18.7   PWM Operation

(4)   Next Frame

When a compare match occurs between PWCNT and PWCYR, data is transferred from the buffer register to the duty register. PWCNT is reset and starts counting up from H'000. The CMF bit in PWCR is set, and if the IE bit in PWCR1 or PWCR2 has been set, an interrupt can be requested or the DMAC can be activated.

(5)   Stopping

When the CST bit in PWCR is cleared to 0, PWCNT is reset and stops. All PWM outputs go low (or high if the corresponding bit in PWPR is set to 1).

18.5.2   Buffer Transfer Control

Setting a corresponding bit in the PWM buffer transfer control register disables a buffer transfer on compare match. This prevents the output from changing when compare match occurs while the buffer register is being changed. A buffer transfer on compare match is resumed after cleaning the bit.
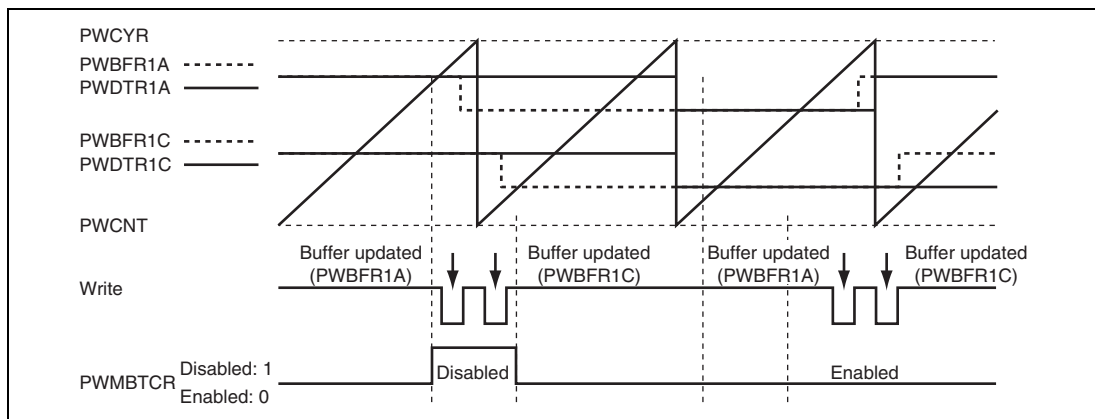
Figure 18.8   Disabling Buffer Transfer

18.6   Usage Note

18.6.1   Conflict between Buffer Register Write and Compare Match

If a PWBFR write is performed in the state immediately after a cycle register compare match, the buffer register and duty register are both modified. PWM output changed by the cycle register compare match is not changed by modification of the duty register due to conflict. This may result in unanticipated duty output.

Buffer register modification must be completed before automatic transfer by the DMAC, exception handling due to a compare match interrupt, or the occurrence of a cycle register compare match on detection of the rise of CMF (compare match flag) in PWCR.
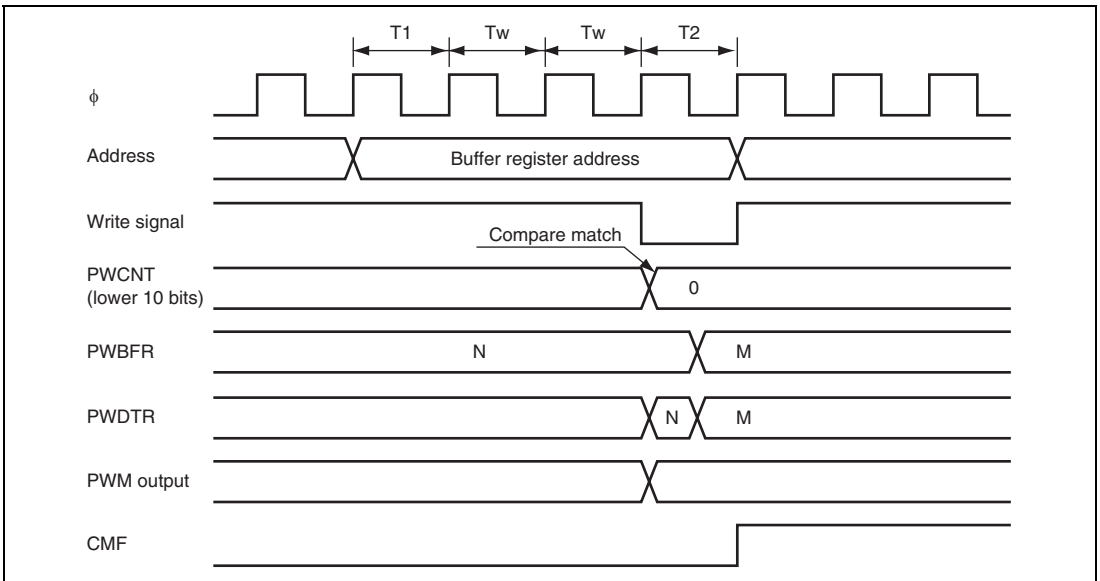
Figure 18.9   Conflict between Buffer Register Write and Compare Match

Section 19   16-Bit PWM

This LSI has an on-chip 16-bit PWM (pulse width modulator) that can output 12 pulses.

19.1   Features

- Capable of 12 pulse outputs

  The PWM includes three channels, each capable of 4 pulse outputs.

  Each channel is equipped with a 16-bit counter (PWCNT) and cycle register (PWCYR).

  The duty cycle can be set for each output.
- Switchable between 16-bit PWM mode and 10-bit stepping motor mode
- Automatic data transfers for each cycle

  Each duty register (PWDTR) is equipped with a buffer register (PWBFR) and can transfer data automatically for each cycle.
- Duty cycle settings selectable

  Duty register settings can be selected in the 0 % to 100 % range.
- Selection from five types of count clocks

  Count clocks can be selected from five types (Pϕ, Pϕ/2, Pϕ/4, Pϕ/8, and Pϕ/16).
- High-speed access via internal 16-bit bus

  16-bit bus interface allows high-speed access.
- Three interrupt sources

  Cycle register compare matches can be used to request interrupts independently for each channel.
- Automatic transfer of register data

  Activating the DMAC enables data transfer in units of blocks or single words.
- On-chip output driver (channels 0 and 1)

  IOH/IOL: typ. = 15 mA, max. = 30 mA

  Total IOH/IOL: max. = 120 mA (target value)
- Module stop mode can be selected

Figure 19.1 shows a block diagram of the PWM.
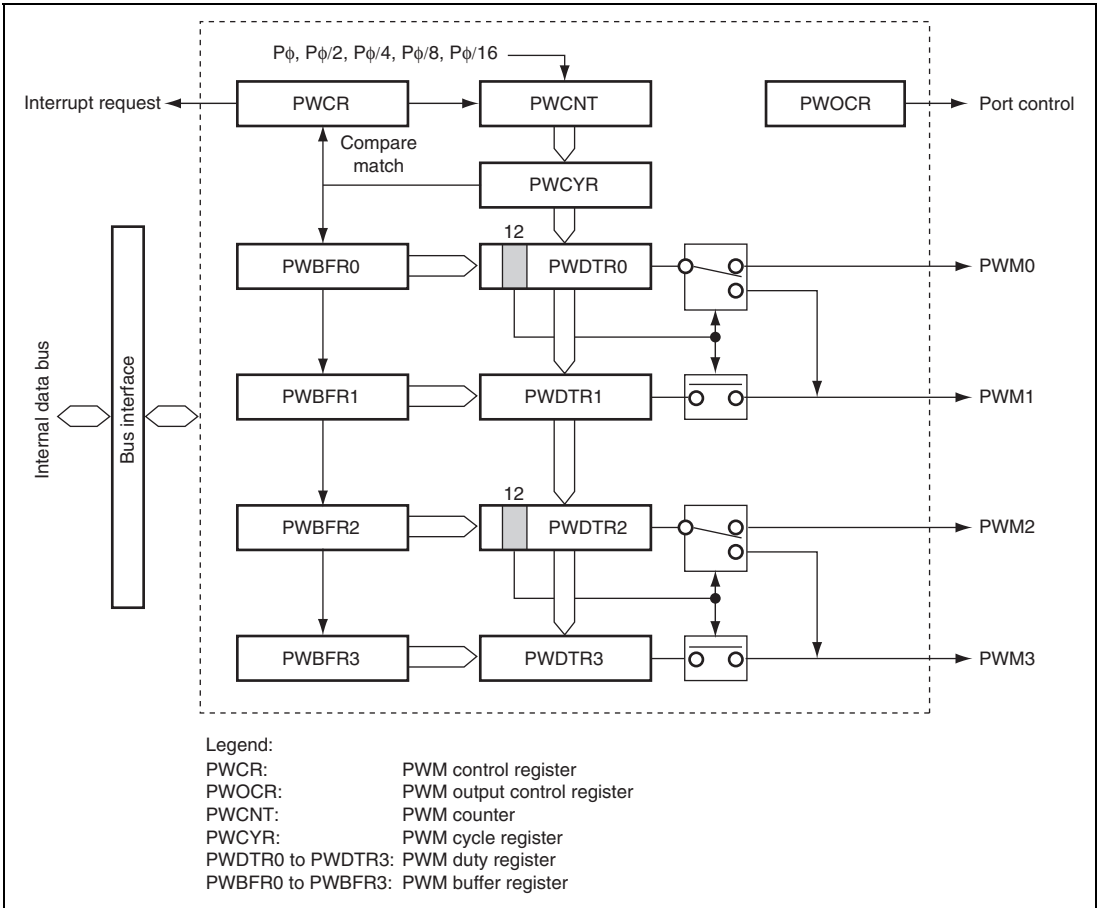
Figure 19.1   Block Diagram of PWM

19.2   Input/Output Pins

Table 19.1 shows the PWM pin configuration.

Table 19.1   Pin Configuration

Channel	Name	Abbr.	I/O	Function
0	PWM output pin 0	PWM0_0	Output	Channel 0 PWM0 output
	PWM output pin 1	PWM1_0	Output	Channel 0 PWM1 output
	PWM output pin 2	PWM2_0	Output	Channel 0 PWM2 output
	PWM output pin 3	PWM3_0	Output	Channel 0 PWM3 output
1	PWM output pin 0	PWM0_1	Output	Channel 1 PWM0 output
	PWM output pin 1	PWM1_1	Output	Channel 1 PWM1 output
	PWM output pin 2	PWM2_1	Output	Channel 1 PWM2 output
	PWM output pin 3	PWM3_1	Output	Channel 1 PWM3 output
2	PWM output pin 0	PWM0_2	Output	Channel 2 PWM0 output
	PWM output pin 1	PWM1_2	Output	Channel 2 PWM1 output
	PWM output pin 2	PWM2_2	Output	Channel 2 PWM2 output
	PWM output pin 3	PWM3_2	Output	Channel 2 PWM3 output

19.3   Register Descriptions

Channel 0:

- PWM control register_0 (PWCR_0)
- PWM output control register_0 (PWOCR_0)
- PWM counter_0 (PWCNT_0)
- PWM cycle register_0 (PWCYR_0)
- PWM duty register 0_0 (PWDTR0_0)
- PWM duty register 1_0 (PWDTR1_0)
- PWM duty register 2_0 (PWDTR2_0)
- PWM duty register 3_0 (PWDTR3_0)
- PWM buffer register 0_0 (PWBFR0_0)
- PWM buffer register 1_0 (PWBFR1_0)
- PWM buffer register 2_0 (PWBFR2_0)
- PWM buffer register 3_0 (PWBFR3_0)

Channel 1:

- PWM control register_1 (PWCR_1)
- PWM output control register_1 (PWOCR_1)
- PWM counter_1 (PWCNT_1)
- PWM cycle register_1 (PWCYR_1)
- PWM duty register 0_1 (PWDTR0_1)
- PWM duty register 1_1 (PWDTR1_1)
- PWM duty register 2_1 (PWDTR2_1)
- PWM duty register 3_1 (PWDTR3_1)
- PWM buffer register 0_1 (PWBFR0_1)
- PWM buffer register 1_1 (PWBFR1_1)
- PWM buffer register 2_1 (PWBFR2_1)
- PWM buffer register 3_1 (PWBFR3_1)

Channel 2:

- PWM control register_2 (PWCR_2)
- PWM output control register_2 (PWOCR_2)
- PWM counter_2 (PWCNT_2)
- PWM cycle register_2 (PWCYR_2)
- PWM duty register 0_2 (PWDTR0_2)
- PWM duty register 1_2 (PWDTR1_2)
- PWM duty register 2_2 (PWDTR2_2)
- PWM duty register 3_2 (PWDTR3_2)
- PWM buffer register 0_2 (PWBFR0_2)
- PWM buffer register 1_2 (PWBFR1_2)
- PWM buffer register 2_2 (PWBFR2_2)
- PWM buffer register 3_2 (PWBFR3_2)

19.3.1   PWM Control Register (PWCR)

PWCR performs operating mode selection, interrupt control, starting/stopping of the counter (PWCNT), and counter (PWCNT) clock selection. It also includes a flag used to indicate compare matches with the cycle register (PWCYR).

Make sure that the counter (PWCNT) has stopped before changing register settings.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	SMS	IE	CMF	CST	CKS2	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	R/W	R/W	R/(W)*	R/W	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	—	Reserved
				This bit is always read as 0 and cannot be modified.
6	SMS	0	R/W	Operating Mode Select
				Selects either 16-bit PWM mode or 10-bit stepping motor mode.
				0: 16-bit PWM mode
				1: 10-bit PWM mode

Bit	Bit Name	Initial Value	R/W	Description
5	IE	0	R/W	Interrupt Enable
				Enables or disables an interrupt request when a compare match occurs with PWCYR of the corresponding channel.
				0: Enables interrupts
				1: Disables interrupts
4	CMF	0	R/(W)*	Compare Match Flag
				Indicates the occurrence of a compare match with PWCYR of the corresponding channel.
				[Setting condition]
				• When PWCNT = PWCYR
				[Clearing conditions]
				• When 0 is written to CMF after CMF = 1 is read
				• When the DMAC is activated by a compare match interrupt, and the DTA bit in DMDR of the DMAC is 1
				(When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	CST	0	R/W	Counter Start
				Starts or stops PWCNT of the corresponding channel.
				0: Stops PWCNT
				1: Starts PWCNT
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	These bits select the count clock for PWCNT of the corresponding channel.
0	CKS0	0	R/W	
				000: Counts on Pϕ
				001: Counts on Pϕ/2
				010: Counts on Pϕ/4
				011: Counts on Pϕ/8
				1xx: Counts on Pϕ/16

Legend:

x: Don't care

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

RENESAS

19.3.2   PWM Output Control Register (PWOCR)

PWOCR enables or disables PWM outputs.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	OE3	OE2	OE1	OE0
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	—	Reserved These bits are always read as 0 and cannot be modified.
3	OE3	0	R/W	Enables or disables PWM3 output. 0: Disables PWM output 1: Enables PWM output
2	OE2	0	R/W	Enables or disables PWM2 output. 0: Disables PWM output 1: Enables PWM output
1	OE1	0	R/W	Enables or disables PWM1 output. 0: Disables PWM output 1: Enables PWM output
0	OE0	0	R/W	Enables or disables PWM0 output. 0: Disables PWM output 1: Enables PWM output

RENESAS

19.3.3   PWM Counter (PWCNT)

PWCNT is a 16-bit up-counter incremented on clock input. The input clock is selected by bits CKS2 to CKS0 in PWCR. PWCNT is not directly accessible by the CPU. PWCNT is initialized to H'0000 when the CST bit in PWCR is to 0.

19.3.4   PWM Cycle Register (PWCYR)

PWCYR is a 16-bit readable/writable register that sets the PWM conversion cycle. When a PWCYR compare match occurs, PWCNT is cleared and data is transferred from the buffer register (PWBFR) to the duty register (PWDTR).

In 16-bit PWM mode (the SMS bit in PWCR is 0), all of the PWCYR bits are available to set the conversion cycle, while in 10-bit stepping motor mode (the SMS bit in PWCR is 1), only the lower 10 bits of PWCYR are available to set the conversion cycle. Note, however, that the upper 6 bits can be written to or read from even in 10-bit stepping motor mode.

Make sure that PWCNT has stopped before writing to PWCYR. H'0000 should not be set to PWCYR. PWCYR is initialized to H'FFFF.

Bit:	15	14	13	12	11	10	9	8
Bit name:	CY15	CY14	CY13	CY12	CY11	CY10	CY9	CY8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	CY7	CY6	CY5	CY4	CY3	CY2	CY1	CY0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
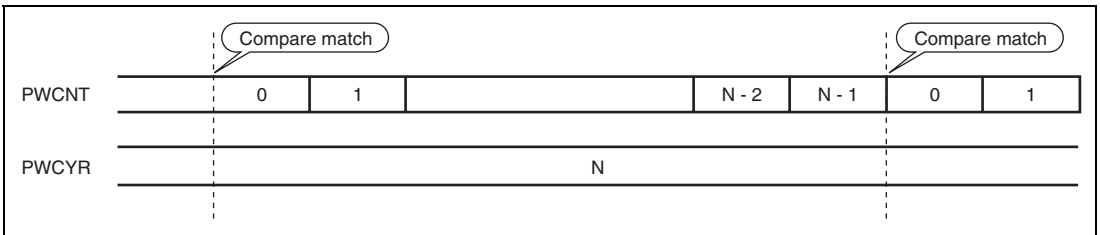
Figure 19.2   Cycle Register Compare Match

19.3.5   PWM Duty Registers 0 to 3 (PWDTR0 to PWDTR3)

PWDTR is comprised of four 16-bit registers (PWDTR0 to PWDTR3) and is not directly accessible by the CPU. When a PWCYR compare match occurs, data is transferred from the buffer register (PWBFR) to the duty register (PWDTR).

PWDTR is initialized to H'0000 when the CST bit is 0. Correspondences between the PWDTR setting and PWM output pins differ according to the operating mode. See tables 19.2 and 19.3 for the corresponding output pin functions.

- PWDTR0 and PWDTR2

Bit:	15	14	13	12	11	10	9	8
Bit name:	DT15	DT14	DT13	DT12/OTS	DT11	DT10	DT9	DT8
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	—	—	—	—

Bit:	7	6	5	4	3	2	1	0
Bit name:	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	—	—	—	—

- PWDTR1 and PWDTR3

Bit:	15	14	13	12	11	10	9	8
Bit name:	DT15	DT14	DT13	DT12/OTS	DT11	DT10	DT9	DT8
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	—	—	—	—

Bit:	7	6	5	4	3	2	1	0
Bit name:	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	—	—	—	—

- In 16-Bit PWM Mode:

Bit	Bit Name	Initial Value	R/W	Description
15 to 0	DT15 to DT0	All 0	—	These bits specify PWM output duty cycle.
				When a PWCYR compare match clears PWCNT, the pin outputs high until a PWDTR compare match occurs. When all bits are 0, there is no period of high level output.

- In 10-Bit Stepping Motor Mode:

Bit	Bit Name	Initial Value	R/W	Description
15 to 13	DT15 to DT13	All 0	—	Invalid
				These bit values will not affect operation.
12	OTS	0	—	Output Terminal Select
				Selects the pin for PWM waveform outputs.
				For details, see tables 19.2 and 19.3.
11	DT11	0	—	Invalid
10	DT10	0	—	These bit values will not affect operation.
9 to 0	DT9 to DT0	All 0	—	These bits specify PWM output duty cycle.
				When a PWCYR compare match clears PWCNT, the pin outputs high until a PWDTR compare match occurs. When all bits are 0, there is no period of high output.

Table 19.2   Selection for PWM0 and PWM1 Outputs

SMS in PWCR	DT12/OTS in PWDTR0	PWM0 Output	PWM1 Output
0	x	PWDTR0[15:0]	PWDTR1[15:0]
1	0	PWDTR0[9:0]	Outputs 0
1	1	Outputs 0	PWDTR0[9:0]

Legend:

x: Don't care

RENESAS

Table 19.3   Selection for PWM2 and PWM3 Outputs

SMS in PWCR	DT12/OTS in PWDTR0	PWM0 Output	PWM1 Output
0	x	PWDTR2[15:0]	PWDTR3[15:0]
1	0	PWDTR2[9:0]	Outputs 0
1	1	Outputs 0	PWDTR2[9:0]

Legend:

x: Don't care

19.3.6   PWM Buffer Registers 0 to 3 (PWBFR0 to PWBFR3)

PWDTR is comprised of four 16-bit readable/writable registers (PWBFR0 to PWBFR3). When a PWCYR compare match occurs, data is transferred from the buffer register (PWBFR) to the duty register (PWDTR).

Bit:	15	14	13	12	11	10	9	8
Bit name:	DT15	DT14	DT13	DT12/OTS	DT11	DT10	DT9	DT8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15 to 13	DT15 to DT13	All 0	R/W	Duty 15 to 13 These bits specify data for bits 15 to 13 in PWDTR.
12	DT12/OTS	0	R/W	Output Terminal Select Specifies data for bit 12 in PWDTR.
11 to 0	DT11 to DT0	All 0	R/W	Duty 11 to 0 These bits specify data for bits 11 to 0 in PWDTR.

19.4   Operation

19.4.1   Operation in 16-Bit PWM Mode

In 16-bit PWM mode, the PWM waveforms, as shown in figure 19.3, are output from pins PWM0 to PWM3.

(1)   Initial Settings

Clear the SMS bit in PWCR to 0 to select 16-bit PWM mode; set the OEn bit (n = 3 to 0) in PWOCR to 1 to enable PWM output from the corresponding pin; select the clock to be input to PWCNT with bits CKS2 to CKS0 in PWCR; set the PWM conversion cycle with PWCYR; and set the first frame of data to PWBFR0 to PWBFR3.

(2)   Activation

Setting the CST bit in PWCR to 1 starts PWCNT counting. A compare match between PWCNT and PWCYR will cause data transfer from PWBFR to PWDTR to set the CMF bit in PWCR to 1. At the same time, if the IE bit in PWCR is set to 1, an interrupt request or the DMAC can be activated.

(3)   Waveform Output

When a compare match occurs between PWCNT and PWCYR, the corresponding pin provides high level output.

When a compare match occurs between PWCNT and PWDTR0 to PWDTR3, the corresponding PWM pin provides low level output.

(4)   Next Frame

A compare match between PWCNT and PWCYR will cause data transfer from PWBFR to PWDTR. This also causes PWCNT to be reset to resume counting from H'000 and the CMF bit in PWCR to be set. At the same time, if the IE bit in PWCR is set, an interrupt request or the DMAC can be activated.

(5)   Stopping

When the CST bit in PWCR is cleared to 0, PWCNT is reset and stops. At the same time, the PWM output pins provide low level output.
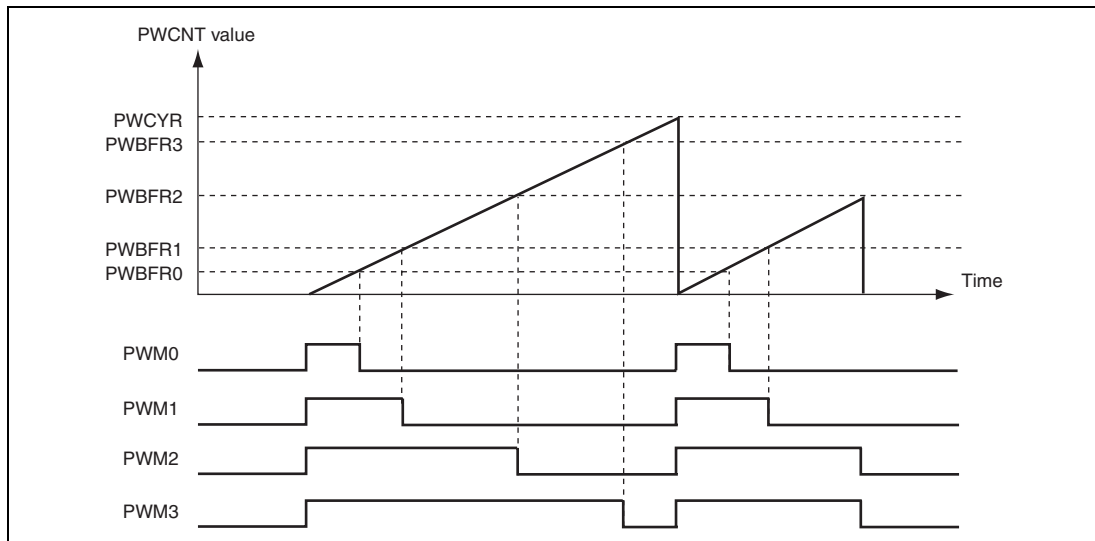
Figure 19.3   Operation in 16-Bit PWM Mode

19.4.2   Operation in 10-Bit Stepping Motor Mode

In 10-bit stepping motor mode, PWM waveforms, as shown in figure 19.4, are output from pins PWM0 to PWM3.

(1)   Initial Settings

Set the SMS bit in PWCR to 1 to select the 10-bit stepping motor mode; set the OEn bit (n = 3 to 0) in PWOCR to 1 to enable PWM output from the corresponding pin; select the clock to be input to PWCNT with bits CKS2 to CKS0 in PWCR; set the PWM conversion cycle with PWCYR; and set the first frame of data to PWBFR0 to PWBFR3.

(2)   Activation

Setting the CST bit in PWCR to 1 starts PWCNT counting. A compare match between the lower 10 bits of PWCNT and PWCYR will cause data transfer from PWBFR to PWDTR to set the CMF bit in PWCR to 1. At the same time, if the IE bit in PWCR is set, an interrupt request or the DMAC can be activated.

(3)   Waveform Output

The PWM outputs selected by the OTS bits in PWDTR0 and PWDTR2 provide high level output when a compare match occurs between PWCNT and PWCYR. The PWM outputs not selected by the OTS bits provide low level output. When a compare match occurs between PWCNT and PWDTR0 to PWDTR3, the corresponding PWM output goes low.

(4)   Next Frame

A compare match between PWCNT and PWCYR will cause data transfers from PWBFR to PWDTR. This also causes PWCNT to be reset to resume counting from H'000 and the CMF bit in PWCR to be set. At the same time, if the IE bit in PWCR is set, an interrupt request or the DMAC can be activated.

(5)   Stopping

When the CST bit in PWCR is cleared to 0, PWCNT is reset and stop. At the same time, the PWM output pins provide low level output.
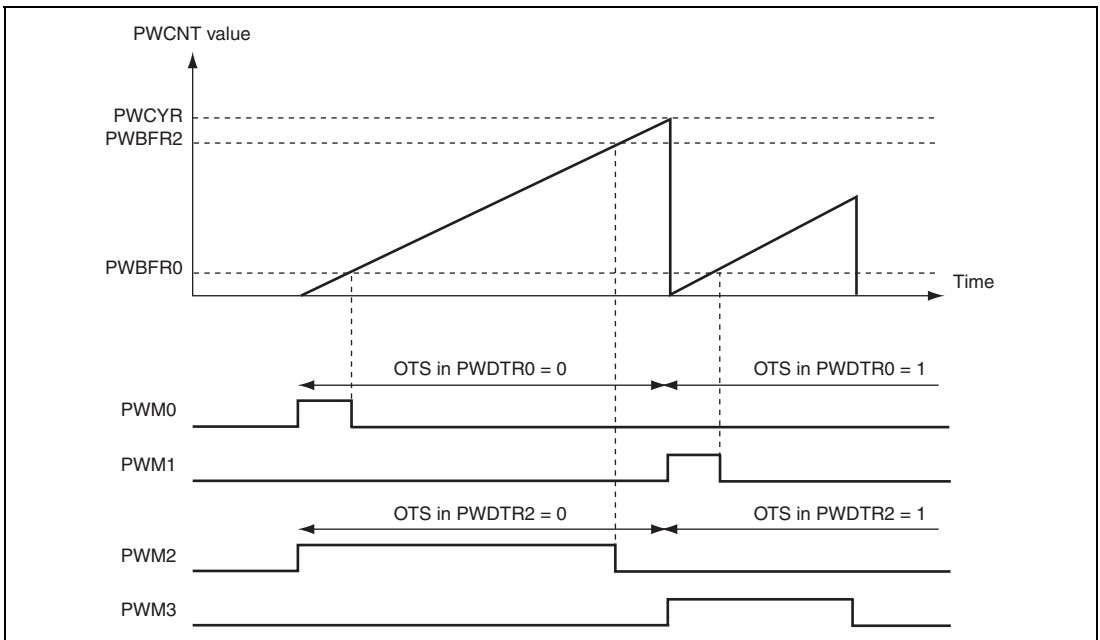
Figure 19.4   Operation in 10-Bit Stepping Motor Mode

19.5   Usage Notes

19.5.1   Precautions for Accessing Registers

(1)   16-Bit Data Registers

PWCYR and PWBFR0 to PWBFR3 are 16-bit registers. These registers can be read from or written to in 16-bit units via a 16-bit data bus between these registers and the bus master. These registers cannot be read from or written to in 8-bit units. Only 16-bit access is allowed.

(2)   8-Bit Data Registers

PWCR and PWOCR are 8-bit registers. These registers can be read from or written to in 8-bit units. Read or write in 16-bit unit is also possible via the 16-bit data bus between these registers and the bus master, however, the lower eight bits are always read as H'FF.

19.5.2   Conflict between Buffer Register Write and Compare Match

If PWBFR is written to immediately after a compare match between PWCYR and PWCNT, the buffer register and duty register are both modified. The PWM output changed by the cycle register compare match is not changed by modification of the duty register due to conflict. This may result in undesired output from the duty register.

Buffer register modification should be completed before automatic transfer by the DMAC, an exception handling triggered by a compare match interrupt or occurrence of a cycle register compare match on detection of the rising edge of the CMF bit in PWCR.
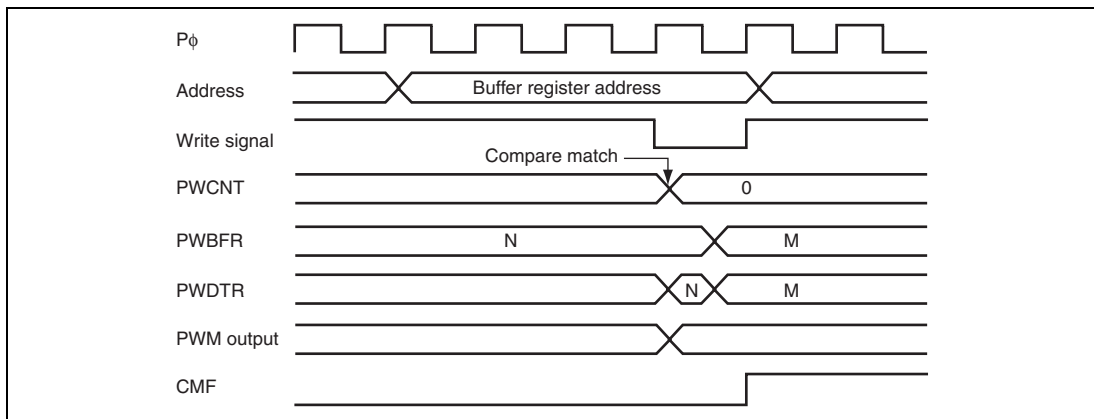
Figure 19.5   Conflict between Buffer Register Write and Compare Match

19.5.3   Rewriting of CKS2 to CKS0

Note that the counter may not be incremented correctly if bits CKS2 to CKS0 in PWCR are rewritten while PWCNT is in operation.

Make sure that PWCNT has stopped (CST bit is 0) before rewriting bits CKS2 to CKS0.

19.5.4   Switching between 16-Bit PWM Mode and 10-Bit Stepping Motor Mode

Note that correct operation is not guaranteed if the mode is switched between 16-bit PWM mode and 10-bit stepping motor mode while PWCNT is in operation. Make sure that PWCNT has stopped (the CST bit is 0) before switching the modes.

RENESAS

Section 20   Sound Generator (SDG)

This LSI has an on-chip sound generator (SDG) with four channels.

20.1   Features

- Capable of adjusting sound volume using 8-bit PWM output
- Selection of operating clocks
  Four types of operating clocks ($\phi/2$, $\phi/4$, $\phi/8$, and $\phi/16$) can be selected.
- Frequency settings in the 31-Hz to 20-kHz range with precision of 1% or less
- SDG output stop procedures can be selected
- Automatic attenuator function can be selected
- Interrupt source: one type
  Attenuation end interrupt can be requested
- Module stop mode can be set

Legend:
SGCR1:   Sound generator control register 1
SGCSR: Sound generator control status register
SGCR2:   Sound generator control register 2
SGLR: Sound generator loudness register
SGTFR: Sound generator tone frequency register
SGSFR: Sound generator reference frequency register
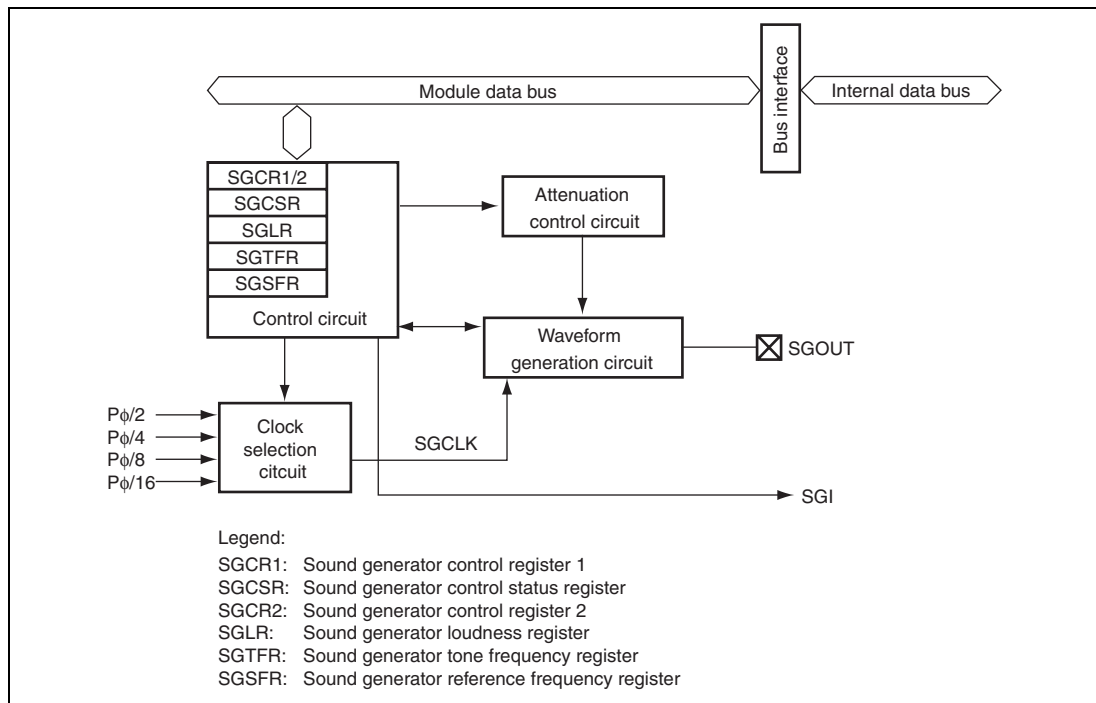
Figure 20.1   Block Diagram of SDG

20.2   Input/Output Pins

Table 20.1 shows the SDG pin configuration.

Table 20.1   Pin Configuration

Name	Abbr.	I/O	Function
Sound generator output pin 0	SGOUT_0	Output	Channel 0 sound generator output
Sound generator output pin 1	SGOUT_1	Output	Channel 1 sound generator output
Sound generator output pin 2	SGOUT_2	Output	Channel 2 sound generator output
Sound generator output pin 3	SGOUT_3	Output	Channel 3 sound generator output

20.3   Register Descriptions

- Sound generator control register 1 (SGCR1)
- Sound generator control status register (SGCSR)
- Sound generator control register 2 (SGCR2)
- Sound generator loudness register (SGLR)
- Sound generator tone frequency register (SGTFR)
- Sound generator reference frequency register (SGSFR)

20.3.1   Sound Generator Control Register 1 (SGCR1)

SGCR1 controls SDG operation.

Bit:	7	6	5	4	3	2	1	0
Bit name:	SGST	STPM	SGE	SGCK1	SGCK0	DPF2	DPF1	DPF0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	SGST	0	R/W	SDG Start
				Starts/stops SDG operation.
				0: Stops SDG operation
				1: Starts SDG operation. The stop procedures differ depending on the STPM bit setting even when SGST = 1
6	STPM	0	R/W	Stop Procedure Select
				Selects SDG operation stop procedure.
				0: Stops when SGST = 0
				1: When the attenuator function is on, operation stops if SGST = 0 and SGDEF = 1
				When the attenuator function is off, operation stops if SGST = 0 and SGEND = 1
5	SGE	0	R/W	SDG Output Enable
				0: Disables SGOUT output
				1: Enables SGOUT output
4	SGCK1	0	R/W	SDG Clock Select
3	SGCK0	0	R/W	Selects the SDG operating clock (SGCLK).
				00: Pϕ/2
				01: Pϕ/4
				10: Pϕ/8
				11: Pϕ/16
2	DPF2	0	R/W	Attenuator Function Select
1	DPF1	0	R/W	These bits turn the attenuator function on and off, and select the attenuation cycle.
0	DPF0	0	R/W	
				000: Attenuator function is off.
				001: Attenuates at the TONE frequency
				010: Attenuates at the TONE frequency/2
				011: Attenuates at the TONE frequency/4
				100: Attenuates at the TONE frequency/8
				101: Attenuates at the TONE frequency/16
				110: Attenuates at the TONE frequency/32
				111: Setting prohibited

RENESAS

20.3.2   Sound Generator Control Status Register (SGCSR)

SGCSR is a status register for the SDG.

Bit:	7	6	5	4	3	2	1	0
Bit name:	SGIE	SGDEF	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/(W)	R/W	R/W	R/W	R/W	R/W	R

Bit	Bit Name	Initial Value	R/W	Description
7	SGIE	0	R/W	SDG Interrupt Enable Enables/disables SDG attenuation end interrupt requests. 0: Disables interrupt requests 1: Enables interrupt requests
6	SGDEF	0	R/(W)	SDG Attenuation End Flag [Setting condition] • When attenuation operation ends [Clearing conditions] • When 0 is written after SGDEF = 1 is read • When SGLR is written • When the DMAC is activated by the SGI interrupt request and data is written to SGLR (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
5 to 1	—	All 0	R/W	Reserved The write value should always be 0.
0	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

RENESAS

20.3.3   Sound Generator Control Register 2 (SGCR2)

SGCR2 is used for SDG termination.

Bit:	7	6	5	4	3	2	1	0
Bit name:	SGEND	TCHG	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	SGEND	0	R/W	SDG Stop
				Controls SDG operation when the attenuator function is off and STPM = 1.
				0: Continues operation
				1: Stops operation
				When STPM = 0, the SGST bit controls operation regardless of this bit setting.
6	TCHG	0	R/W	TONE Change Protect
				Enables/disables writing to the TONE or SFS bit.
				Bits TONE and SFS can be modified when TCHG = 1.
				0: Disables writing to the TONE and SFS bit
				1: Enables writing to the TONE and SFS bit
5 to 0	—	All 0	R/W	Reserved
				The write value should always be 0.

20.3.4   Sound Generator Loudness Register (SGLR)

SGLR sets the SGOUT duty.

Bit:	7	6	5	4	3	2	1	0
Bit name:	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
7	LD7	0	R/W	Loudness Data
6	LD6	0	R/W	These bits store the duty data for pulse output.
5	LD5	0	R/W	
4	LD4	0	R/W	
3	LD3	0	R/W	
2	LD2	0	R/W	
1	LD1	0	R/W	
0	LD0	0	R/W	

20.3.5   Sound Generator Tone Frequency Register (SGTFR)

SGTFR sets the SDG tone frequency.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	TONE6	TONE5	TONE4	TONE3	TONE2	TONE1	TONE0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: * These bits can be written to only when TCHG = 1.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/(W)*	Reserved
				The write value should always be 0.
6	TONE6	0	R/(W)*	Tone Frequency Setting
5	TONE5	0	R/(W)*	These bits set the tone frequency based on the
4	TONE4	0	R/(W)*	reference frequency specified by the SFS bit. Setting H'00 is prohibited.
3	TONE3	0	R/(W)*	
2	TONE2	0	R/(W)*	
1	TONE1	0	R/(W)*	
0	TONE0	0	R/(W)*	

Note: * These bits can be written to when TCHG = 1.

RENESAS

20.3.6 Sound Generator Reference Frequency Register (SGSFR)

SGSFR sets the SDG reference frequency.

Bit:	7	6	5	4	3	2	1	0
Bit name:	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: *   These bits can be written to only when TCHG = 1.

Bit	Bit Name	Initial Value	R/W	Description
7	SFS7	0	R/(W)*	Reference Frequency Setting
6	SFS6	0	R/(W)*	These bits set the reference frequency based on the
5	SFS5	0	R/(W)*	operating clock (SGCLK) specified by the SGCK bit in SGCR1. Setting H'00 is prohibited.
4	SFS4	0	R/(W)*	
3	SFS3	0	R/(W)*	
2	SFS2	0	R/(W)*	
1	SFS1	0	R/(W)*	
0	SFS0	0	R/(W)*	

Note: *   These bits can be written to when TCHG = 1.

20.4   Operation

20.4.1   SDG Operation

(1)   Initial Settings

Make sure that the SDG has stopped before setting registers.

The operation stop procedure, operating clock and attenuator function on/off settings are specified by the SGCR1 setting. When the attenuator function is on, the attenuation cycle is selected. The interrupt request is set by the SGIE bit in SGCSR.

(2)   Activation

SDG operation is enabled by setting the SGST bit in SGCR1 to 1 and clearing the SGEND bit in SGCR2 to 0. Set the TCHG bit in SGCR2 to 1 to cancel the write protect for SGSFR and SGTFR. Use bits SFS7 to SFS0 in SGSFR to select the reference frequency and bits TONE6 to TONE0 in SGTFR to select the tone frequency. SGLR specifies the output pulse duty. The SDG starts operation after writing to SGCR2, SGLR, SGTFR, and SGSFR is completed.

(3)   Stopping

The SDG operation stop procedure is specified by the STPM bit in SGCR1.

When the attenuator function is off and STPM is 0, SDG operation is stopped by the SGST bit regardless of the SGEND bit setting. When the attenuator function is off and STPM is 1, SDG operation is stopped by clearing the SGST bit to 0 and setting the SGEND bit to 1.

When the attenuator function is on and STPM is 0, clearing the SGST bit to 0 stops operation even though the automatic attenuation does not end and the SGDEF bit is not set to 1. When the attenuator function is on and STPM is 1, clearing the SGST bit to 0 does not stop operation until the automatic attenuation ends and the SGDEF bit is set to 1.

RENESAS

The conditions for stopping the SDG are listed in table 20.2 and examples of SDG stopping operation are shown in figure 20.2.

Table 20.2   SDG Stop Conditions

Attenuator Function Off					Attenuator Function On				
STPM	SGST	SGEND	SGDEF	Operation	STPM	SGST	SGEND	SGDEF	Operation
0	0	x	x	Stopped	0	0	x	x	Stopped
0	1	x	x	Output	0	1	x	x	Output
1	0	0	x	Held*	1	0	x	0	Held*
1	0	1	x	Stopped	1	0	x	1	Stopped
1	1	0	x	Output	1	1	x	0	Output
1	1	1	x	Output	1	1	x	1	Output

Legend:

x: Don't care

Note: *   Previous state is held.

Attenuator function Off

• STPM = 0

SGST

SGEND

SGOUT

0 is written to SGST
Operation is stopped

• STPM = 1

SGST

SGEND

SGOUT

0 is written to SGST

Operation is stopped

Attenuator function On

• STPM = 0

SGST

SGDEF
(Attenuation end flag)

SGOUT

0 is written to SGST
Operation is stopped

• STPM = 1

SGST

SGDEF
(Attenuation end flag)

SGOUT

Flag set
(Attenuation end)

Flag clear

0 is written to SGST

Operation is stopped
Flag set
(Attenuation end)

Figure 20.2   Examples of SDG Stopping Operation

RENESAS
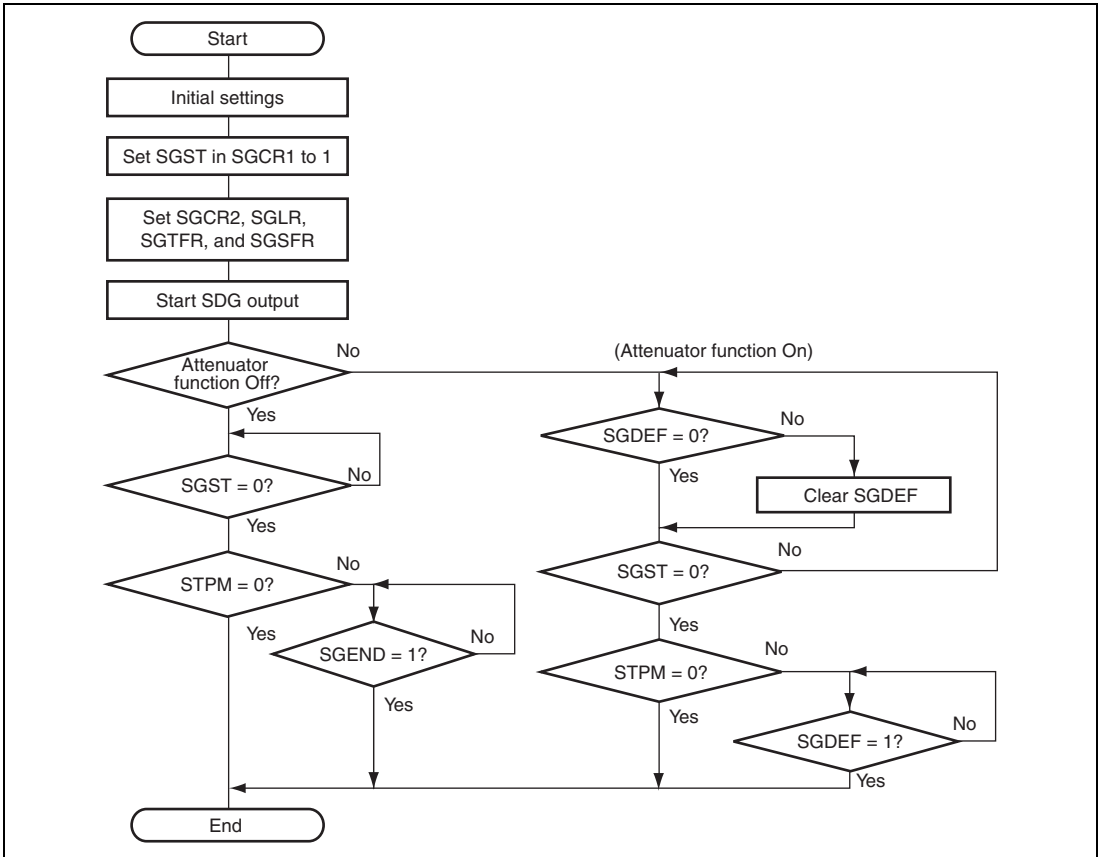
Figure 20.3   SDG Operation Flow

20.4.2   Tone Frequency Setting

The SDG provides tone frequency output in the range of 31 Hz to 20 kHz, with precision of 1% or less.

The tone frequency is calculated by:

Reference frequency (Hz) = SGCLK (Hz)/SFS

Tone frequency (Hz) = Reference frequency (Hz)/(2 × TONE)

= SGCLK (Hz)/(2 × SFS × TONE)

Setting values of the TONE bit in SGTFR and SFS bit in SGFSR are calculated by:

SFS = SGCLK (Hz)/reference frequency (Hz) (0 < SFS ≤ 255)

TONE = Reference frequency (Hz)/(2 × TONE frequency (Hz)) (0 < TONE ≤ 127)

An example of relationship between the tone frequency and output error is shown in table 20.3.

Table 20.3   Relationship between Tone Frequency and Output Error

Tone Frequency	SFS[7:0]	TONE[6:0]	Error (%)
220.00	F7	2E	0.01
329.63	ED	20	0.003
440.00	F7	17	0.01
659.26	ED	10	0.003
880.00	8E	14	0.03
1318.50	ED	8	0.04
1760.00	8E	A	0.03
2637.00	ED	4	0.004
3520.00	8E	5	0.03
5274.00	ED	2	0.002
7040.00	47	5	0.03

Note: SGCLK = 5 MHz

Since changing Pϕ frequency affects the tone frequency, care should be taken when changing it.

RENESAS

20.4.3 Auto Attenuator Function

When the auto attenuator function is on, the loudness data (LD) determines the initial duty cycle for SGOUT.

The SGOUT duty cycle is attenuated by a factor of 1/32 using the attenuation cycle set by the DPF bit in SGCR1.

The SDG attenuation characteristics are calculated by the equation:

$$LD_n = \text{int } (LD_0 \times (1 - 1/32)^n)$$

LD: SGOUT duty cycle (The initial data is SGLR.)
n: Attenuation cycle number

Figure 20.4 is a graph of attenuation characteristics.
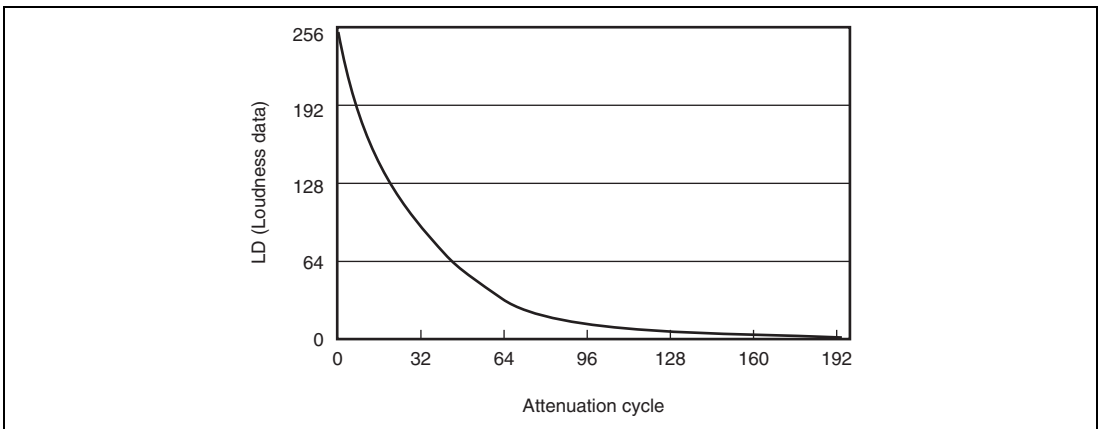
Figure 20.4 Attenuation Characteristics

20.4.4   Output Waveform

As shown in figure 20.5, the SDG output waveform is obtained by synthesis of the on-chip 8-bit PWM pulse output and the tone frequency output. The duty cycle for the on-chip 8-bit PWM pulse output is set by SGLR.
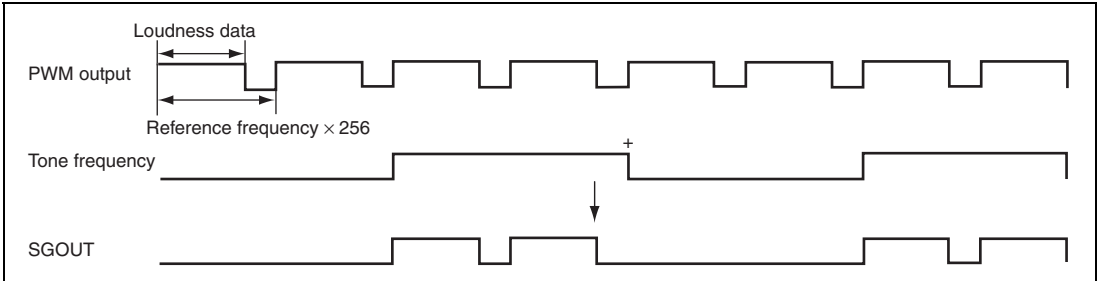
Figure 20.5   Output Waveforms

20.5   Interrupt Source

When the attenuator function is on and an automatic attenuation operation is completed, the SGDEF bit in SGCSR is set. An interrupt request is issued if the SGIE bit in SGCSR is set to 1. When STPM = 0, the SGDEF bit is set only when the first attenuation operation is completed. If the SGDEF bit is cleared during automatic attenuation, the SGDEF bit is set when the next attenuation operation is completed.

Table 20.4   SDG Interrupt Source

Name	Interrupt Source	Interrupt Flag
SGI	Attenuation end	SGDEF

20.6   Usage Note

20.6.1   Module Stop Mode Settings

The module stop control register enables or disables the SDG operation. With the initial setting, the SDG operation is stopped. Register access is enabled by canceling module stop mode. For details, see section 24, Power-Down Modes.

Section 21   RAM

This LSI has a 24-kbyte on-chip high-speed static RAM. The RAM is connected to the CPU by a 32-bit data bus, enabling one-state access for read and two-state access for write by the CPU to all byte data, word data, and longword data.

The on-chip RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on SYSCR, refer to section 3.2.2, System Control Register (SYSCR).

Product Classification		RAM Size	RAM Addresses
Flash memory version	H8SX/1544	24 kbytes	H'FF6000 to H'FFBFFF
	H8SX/1543	16 kbytes	H'FF8000 to H'FFBFFF

Section 22   Flash Memory

The flash memory has the following features. Figure 22.1 is a block diagram of the flash memory.

22.1   Features

- Size

Product Classification		ROM Size	ROM Address
H8SX/1544	R5F61544	512 kbytes	H'000000 to H'07FFFF (modes 2, 6, and 7)
H8SX/1543	R5F61543	384 kbytes	H'000000 to H'05FFFF (modes 2, 6, and 7)

- Programming/erasing interface by the download of on-chip program

  This LSI has a programming/erasing program. After downloading this program to the on-chip RAM, programming/erasing can be performed by setting the parameters. A user branch function is also supported.

- User branch function

  Programming processing is performed in 128-byte units and is made up of several steps, such as application of programming pulses and verify-and-read. Erasure is performed in block units and is also made up of several processing steps. Between these steps, jumping to a user-set branch is possible. This is called the user branch function.

- Programming/erasing time

  Programming time: 1 ms (typ) for 128-byte simultaneous programming

  Erasing time: 0.6 s (typ) per 1 block (64 kbytes)

- Number of programming

  The number of programming can be up to 100 times at the minimum. (1 to 100 times are guaranteed.)

- Two on-board programming modes

  Boot mode: Using the on-chip SCI_4, the user MAT can be programmed/erased. In boot mode, the bit rate between the host and this LSI can be adjusted automatically.

  User program mode: Using a desired interface, the user MAT can be programmed/erased.

- Off-board programming mode

  Programmer mode: Using a PROM programmer, the user MAT can be programmed/erased.

- Programming/erasing protection

  Protection against programming/erasing of the flash memory can be set by hardware protection, software protection, or error protection.

- Flash memory emulation function using the on-chip RAM

Realtime emulation of the flash memory programming can be performed by overlaying parts of the flash memory (user MAT) area and the on-chip RAM.

Internal address bus

Internal data bus (32 bits)

Module bus

FCCS
FPCS
FECS
FKEY
FTDAR
RAMER

Control unit

Memory MAT unit

User MAT
H8SX/1544: 512 kbytes
H8SX/1543: 384 kbytes

Flash memory

Mode pins ──→ Operating mode

Legend:
FCCS: Flash code control/status register
FPCS: Flash program code select register
FECS: Flash erase code select register
FKEY: Flash key code register
FTDAR: Flash transfer destination address register
RAMER: RAM emulation register

Figure 22.1   Block Diagram of Flash Memory

RENESAS

22.2 Mode Transition Diagram

When the mode pins are set in the reset state and reset start is performed, this LSI enters each operating mode as shown in figure 22.2. Although the flash memory can be read in user mode, it cannot be programmed or erased. The flash memory can be programmed or erased in boot mode, user program mode, and programmer mode. The differences between boot mode, user program mode, and programmer mode are shown in table 22.1.
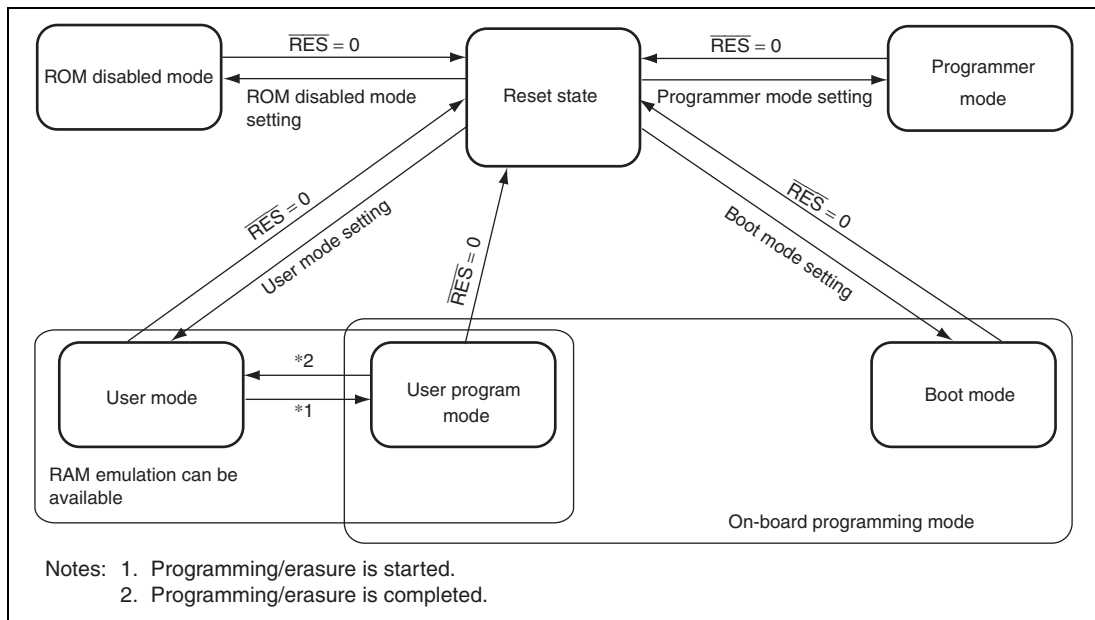
Figure 22.2   Mode Transition of Flash Memory

Table 22.1   Differences between Boot Mode, User Program Mode, and Programmer Mode

Item	Boot Mode	User Program Mode	Programmer Mode
Programming/ erasing environment	On-board programming	On-board programming	Off-board programming
Programming/ erasing enable MAT	• User MAT	• User MAT	• User MAT
Programming/ erasing control	Command	Programming/ erasing interface	Command
All erasure	O (Automatic)	O	O (Automatic)
Block division erasure	O*[1]	O	×
Program data transfer	From host via SCI	From desired device via RAM	Via programmer
User branch function	×	O	×
RAM emulation	×	O	×
Reset initiation MAT	Embedded program storage area	User MAT	—
Transition to user mode	Changing mode and reset	Completing Programming/ erasure*[3]	—

Notes: 1. All-erasure is performed. After that, the specified block can be erased.
2. In this LSI, the user programming mode is defined as the period from the timing when a program concerning programming and erasure is started to the timing when the program is completed. For details on a program concerning programming and erasure, see section 22.7.2, User Program Mode.

RENESAS

22.3   Block Structure

Figure 22.3 shows the block structure of the 512-kbyte user MAT. Figure 22.4 shows the block structure of the 384-kbyte user MAT. The heavy-line frames indicate the erase blocks. The thin-line frames indicate the programming units and the values inside the frames stand for the addresses. The 512-kbyte user MAT is divided into seven 64-kbyte blocks, one 32-kbyte block, and eight 4-kbyte blocks. The 384-kbyte user MAT is divided into five 64-kbyte blocks, one 32-kbyte block, and eight 4-kbyte blocks. The user MAT can be erased in these divided block units. Programming is done in 128-byte units starting from where the lower address is H'00 or H'80. RAM emulation can be performed in the eight 4-kbyte blocks.

Erase block	Address row				
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	H'020000	H'020001	H'020002	←Programming unit: 128 bytes→	H'02007F
	H'0AFF80	H'0AFF81	H'0AFF82	– – – – – – – – – – – – –	H'0AFFFF
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 22.3   Block Structure of 512-kbyte User MAT

RENESAS

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	H'020000	H'020001	H'020002	←Programming unit: 128 bytes→	H'02007F
	H'0AFF80	H'0AFF81	H'0AFF82	− − − − − − − − − − − − −	H'0AFFFF
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

Figure 22.4   Block Structure of 384-kbyte User MAT

RENESAS

22.4   Programming/Erasing Interface

Programming/erasing of the flash memory is done by downloading an on-chip programming/erasing program to the on-chip RAM and specifying the start address of the programming destination, the program data, and the erase block number using the programming/erasing interface registers and programming/erasing interface parameters.

The procedure program for user program mode is made by the user. Figure 22.5 shows the procedure for creating the procedure program. For details, see section 22.7.2, User Program Mode.
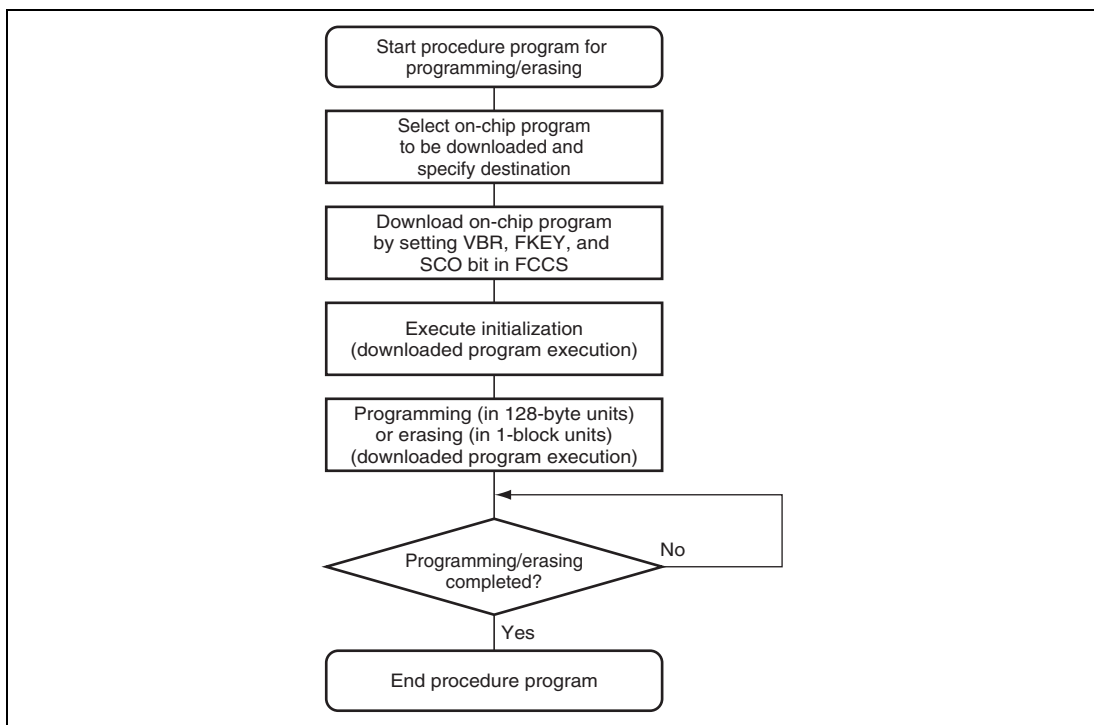

Figure 22.5   Procedure for Creating Procedure Program

(1)   Selection of On-Chip Program to Be Downloaded

This LSI has programming/erasing programs which can be downloaded to the on-chip RAM. The on-chip program to be downloaded is selected by the programming/erasing interface registers. The start address of the on-chip RAM where an on-chip program is downloaded is specified by the flash transfer destination address register (FTDAR).

(2)   Download of On-Chip Program

The on-chip program is automatically downloaded by setting the flash key code register (FKEY) and the SCO bit in the flash code control/status register (FCCS) after initializing the vector base register (VBR). The memory MAT is replaced with the embedded program storage area during download. Since the memory MAT cannot be read during programming/erasing, the procedure program must be executed in a space other than the flash memory (for example, on-chip RAM). Since the download result is returned to the programming/erasing interface parameter, whether download is normally executed or not can be confirmed. The VBR contents can be changed after completion of download.

(3)   Initialization of Programming/Erasing

Before executing programming or erasing, the operating frequency and addresses for user branching should be set. A user branch address should be neither in an on-chip flash memory area nor the area where the on-chip program has been downloaded. These settings are made through the programming/erasing interface parameters.

(4)   Execution of Programming/Erasing

The start address of the programming destination and the program data are specified in 128-byte units when programming. The block to be erased is specified with the erase block number in erase-block units when erasing. Specifications of the start address of the programming destination, program data, and erase block number are performed by the programming/erasing interface parameters, and the on-chip program is initiated. The on-chip program is executed by using the JSR or BSR instruction and executing the subroutine call of the specified address in the on-chip RAM. The execution result is returned to the programming/erasing interface parameter.

The area to be programmed must be erased in advance when programming flash memory. All interrupts are disabled during programming/erasing.

(5)   When Programming/Erasing Is Executed Consecutively

When processing does not end by 128-byte programming or 1-block erasure, consecutive programming/erasing can be realized by updating the start address of the programming destination and program data, or the erase block number. Since the downloaded on-chip program is left in the on-chip RAM even after programming/erasing completes, download and initialization are not required when the same processing is executed consecutively.

RENESAS

22.5   Input/Output Pins

The flash memory is controlled through the input/output pins shown in table 22.2.

Table 22.2   Pin Configuration

Abbreviation	I/O	Function
RES	Input	Reset
MD2 to MD0	Input	Set operating mode of this LSI
TxD4	Output	Serial transmit data output (used in boot mode)
RxD4	Input	Serial receive data input (used in boot mode)

22.6   Register Descriptions

The flash memory has the following registers.

Programming/Erasing Interface Registers:

- Flash code control/status register (FCCS)
- Flash program code select register (FPCS)
- Flash erase code select register (FECS)
- Flash key code register (FKEY)
- Flash transfer destination address register (FTDAR)

Programming/Erasing Interface Parameters:

- Download pass and fail result parameter (DPFR)
- Flash pass and fail result parameter (FPFR)
- Flash program/erase frequency parameter (FPEFEQ)
- Flash multipurpose address area parameter (FMPAR)
- Flash multipurpose data destination area parameter (FMPDR)
- Flash erase block select parameter (FEBS)
- Flash user branch address set parameter (FUBRA)
- RAM emulation register (RAMER)

There are several operating modes for accessing the flash memory. Respective operating modes, registers, and parameters are assigned to the user MAT. The correspondence between operating modes and registers/parameters for use is shown in table 22.3.

Table 22.3   Registers/Parameters and Target Modes

Register/Parameter		Download	Initialization	Programming	Erasure	Read	RAM Emulation
Programming/ erasing interface registers	FCCS	O	—	—	—	—	—
	FPCS	O	—	—	—	—	—
	FECS	O	—	—	—	—	—
	FKEY	O	—	O	O	—	—
	FTDAR	O	—	—	—	—	—
Programming/ erasing interface parameters	DPFR	O	—	—	—	—	—
	FPFR	—	O	O	O	—	—
	FPEFEQ	—	O	—	—	—	—
	FMPAR	—	—	O	—	—	—
	FMPDR	—	—	O	—	—	—
	FEBS	—	—	—	O	—	—
	FUBRA	—	O	—	—	—	—
RAM emulation	RAMER	—	—	—	—	—	O

22.6.1   Programming/Erasing Interface Registers

The programming/erasing interface registers are 8-bit registers that can be accessed only in bytes. These registers are initialized by a power-on reset.

(1)   Flash Code Control/Status Register (FCCS)

FCCS monitors errors during programming/erasing the flash memory and requests the on-chip program to be downloaded to the on-chip RAM.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	FLER	—	—	—	SCO
Initial Value	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	(R)/W

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R	Reserved
6	—	0	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	FLER	0	R	Flash Memory Error
				Indicates that an error has occurred during programming or erasing the flash memory. When this bit is set to 1, the flash memory enters the error protection state. When this bit is set to 1, high voltage is applied to the internal flash memory. To reduce the damage to the flash memory, the reset must be released after the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.
				0: Flash memory operates normally (Error protection is invalid)
				[Clearing condition]
				• At a power-on reset
				1: An error occurs during programming/erasing flash memory (Error protection is valid)
				[Setting conditions]
				• When an interrupt, such as NMI, occurs during programming/erasing
				• When the flash memory is read during programming/erasing (including a vector read and an instruction fetch)
				• When the SLEEP instruction is executed during programming/erasing (including software standby mode)
				• When a bus master other than the CPU, such as the DMAC, obtains bus mastership during programming/erasing
3 to 1	—	All 0	R	Reserved
				These are read-only bits and cannot be modified.

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 or FECS is automatically downloaded in the on-chip RAM area specified by FTDAR.
				In order to set this bit to 1, the RAM emulation mode must be canceled, H'A5 must be written to FKEY, and this operation must be executed in the on-chip RAM. Dummy read of FCCS must be executed twice immediately after setting this bit to 1. All interrupts must be disabled during download. This bit is cleared to 0 when download is completed.
				During program download initiated with this bit, particular processing which accompanies bank-switching of the program storage area is executed. Before a download request, initialize the VBR contents to H'00000000. After download is completed, the VBR contents can be changed.
				0: Download of the programming/erasing program is not requested
				[Clearing condition]
				• When download is completed
				1: Download of the programming/erasing program is requested
				[Setting conditions] (When all of the following conditions are satisfied)
				• Not in RAM emulation mode (the RAMS bit in RAMER is cleared to 0)
				• H'A5 is written to FKEY
				• Setting of this bit is executed in the on-chip RAM

Note: *   This is a write-only bit. This bit is always read as 0.

RENESAS

(2)   Flash Program Code Select Register (FPCS)

FPCS selects the programming program to be downloaded.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	PPVS	0	R/W	Program Pulse Verify Selects the programming program to be downloaded. 0: Programming program is not selected [Clearing condition] • When transfer is completed 1: Programming program is selected

(3)   Flash Erase Code Select Register (FECS)

FECS selects the erasing program to be downloaded.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	EPVB	0	R/W	Erase Pulse Verify Block Selects the erasing program to be downloaded. 0: Erasing program is not selected [Clearing condition] • When transfer is completed 1: Erasing program is selected

RENESAS

(4)   Flash Key Code Register (FKEY)

FKEY is a register for software protection that enables to download the on-chip program and perform programming/erasing of the flash memory.

Bit	7	6	5	4	3	2	1	0
Bit Name	K7	K6	K5	K4	K3	K2	K1	K0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	K7	0	R/W	Key Code
6	K6	0	R/W	When H'A5 is written to FKEY, writing to the SCO bit in FCCS is enabled. When a value other than H'A5 is written, the SCO bit cannot be set to 1. Therefore, the on-chip program cannot be downloaded to the on-chip RAM.
5	K5	0	R/W	
4	K4	0	R/W	
3	K3	0	R/W	
2	K2	0	R/W	Only when H'5A is written can programming/erasing of the flash memory be executed. When a value other than H'5A is written, even if the programming/erasing program is executed, programming/erasing cannot be performed.
1	K1	0	R/W	
0	K0	0	R/W	
				H'A5: Writing to the SCO bit is enabled (The SCO bit cannot be set to 1 when FKEY is a value other than H'A5)
				H'5A: Programming/erasing of the flash memory is enabled (When FKEY is a value other than H'A5, the software protection state is entered)
				H'00: Initial value

RENESAS

(5)   Flash Transfer Destination Address Register (FTDAR)

FTDAR specifies the start address of the on-chip RAM at which to download an on-chip program.
FTDAR must be set before setting the SCO bit in FCCS to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	TDER	0	R/W	Transfer Destination Address Setting Error
				This bit is set to 1 when an error has occurred in setting the start address specified by bits TDA6 to TDA0.
				A start address error is detected by judging whether the value set in bits TDA6 to TDA0 is within the range of H'00 to H'02 when downloading is executed by setting the SCO bit in FCCS to 1. Before setting the SCO bit to 1, make sure that TDER is cleared to 0 and set a value within the range of H'00 to H'02 in bits TDA6 to TDA0.
				0: The value specified by bits TDA6 to TDA0 is within the range
				1: The value specified by bits TDER and TDA6 to TDA0 is between H'03 and H'FF and download has stopped
6	TDA6	0	R/W	Transfer Destination Address
5	TDA5	0	R/W	Specifies the on-chip RAM start address of the download destination. A value between H'00 and H'02, and up to 4 kbytes can be specified as the start address of the on-chip RAM.
4	TDA4	0	R/W	
3	TDA3	0	R/W	
2	TDA2	0	R/W	H'00: H'FF9000 is specified as the start address
1	TDA1	0	R/W	H'01: H'FFA000 is specified as the start address
0	TDA0	0	R/W	H'02: H'FFB000 is specified as the start address
				H'03 to H'7F: Setting prohibited (Specifying a value from H'03 to H'7F sets the TDER bit to 1 and stops download of the on-chip program)

22.6.2   Programming/Erasing Interface Parameters

The programming/erasing interface parameters specify the operating frequency, user branch address, storage place for program data, start address of programming destination, and erase block number, and exchanges the execution result. These parameters use the general registers of the CPU (ER0 and ER1) or the on-chip RAM area. The initial values of programming/erasing interface parameters are undefined at a power-on reset or a transition to software standby mode.

Since the contents of the CPU registers except for ER0 and ER1 are saved in the stack area during downloading of the on-chip program, initialization, programming, or erasing, allocate the stack area before performing these operations (the maximum stack size is 128 bytes). The return value of the processing result is written in R0. The programming/erasing interface parameters are used in download control, initialization before programming or erasing, programming, and erasing. Table 22.4 shows the usable parameters and target modes. The meaning of the bits in the flash pass and fail result parameter (FPFR) varies in initialization, programming, and erasure.

Table 22.4   Parameters and Target Modes

Parameter	Download	Initialization	Programming	Erasure	R/W	Initial Value	Allocation
DPFR	O	—	—	—	R/W	Undefined	On-chip RAM*
FPFR	O	O	O	O	R/W	Undefined	R0L of CPU
FPEFEQ	—	O	—	—	R/W	Undefined	ER0 of CPU
FMPAR	—	—	O	—	R/W	Undefined	ER1 of CPU
FMPDR	—	—	O	—	R/W	Undefined	ER0 of CPU
FEBS	—	—	—	O	R/W	Undefined	ER0 of CPU
FUBRA	—	O	—	—	R/W	Undefined	ER0 of CPU

Note: *   A single byte of the start address of the on-chip RAM specified by FTDAR

(a)   Download Control

The on-chip program is automatically downloaded by setting the SCO bit in FCCS to 1. The on-chip RAM area to download the on-chip program is the 4-kbyte area starting from the start address specified by FTDAR. Download is set by the programming/erasing interface registers, and the download pass and fail result parameter (DPFR) indicates the return value.

RENESAS

(b)   Initialization before Programming/Erasing

The on-chip program includes the initialization program. A pulse with the specified period must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by CPU instructions. Accordingly, the operating frequency of the CPU must be set. Since the user branch function is supported, a user branch address should be set as necessary. The initial program is set as a parameter of the programming/erasing program which has been downloaded to perform these settings.

(c)   Programming

When the flash memory is programmed, the start address of the programming destination on the user MAT and the program data must be passed to the programming program.

The start address of the programming destination on the user MAT must be stored in general register ER1. This parameter is called the flash multipurpose address area parameter (FMPAR).

The program data is always in 128-byte units. When the program data does not satisfy 128 bytes, 128-byte program data is prepared by filling the dummy code (H'FF). The boundary of the start address of the programming destination on the user MAT is aligned at an address where the lower eight bits (A7 to A0) are H'00 or H'80.

The program data for the user MAT must be prepared in consecutive areas. The program data must be in a consecutive space which can be accessed using the MOV.B instruction of the CPU and is not in the flash memory space.

The start address of the area that stores the data to be written in the user MAT must be set in general register ER0. This parameter is called the flash multipurpose data destination area parameter (FMPDR).

For details on the programming procedure, see section 22.7.2, User Program Mode.

(d)   Erasure

When the flash memory is erased, the erase block number on the user MAT must be passed to the erasing program which is downloaded.

The erase block number on the user MAT must be set in general register ER0. This parameter is called the flash erase block select parameter (FEBS).

One block is selected from the block numbers of 0 to 15 as the erase block number.

For details on the erasing procedure, see section 22.7.2, User Program Mode.

(1)   Download Pass and Fail Result Parameter (DPFR: Single Byte of Start Address in On-Chip RAM Specified by FTDAR)

DPFR indicates the return value of the download result. The DPFR value is used to determine the download result.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused
				These bits return 0.
2	SS	—	R/W	Source Select Error Detect
				Only one type can be specified for the on-chip program which can be downloaded. When the program to be downloaded is not selected, more than two types of programs are selected, or a program which is not mapped is selected, an error occurs.
				0: Download program selection is normal
				1: Download program selection is abnormal
1	FK	—	R/W	Flash Key Register Error Detect
				Checks the FKEY value (H'A5) and returns the result.
				0: FKEY setting is normal (H'A5)
				1: FKEY setting is abnormal (value other than H'A5)
0	SF	—	R/W	Success/Fail
				Returns the download result. Reads back the program downloaded to the on-chip RAM and determines whether it has been transferred to the on-chip RAM.
				0: Download of the program has ended normally (no error)
				1: Download of the program has ended abnormally (error occurs)

RENESAS

(2)   Flash Pass/Fail Parameter (FPFR: General Register R0L of CPU)

FPFR indicates the return values of the initialization, programming, and erasure results. The meaning of the bits in FPFR varies depending on the processing.

(a)   Initialization before Programming/Erasing

FPFR indicates the return value of the initialization result.

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

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused These bits return 0.
2	BR	—	R/W	User Branch Error Detect Checks whether the specified user branch destination address is other than the area where the write/erasure-related programs to be downloaded are stored, and returns the result. 0: Setting of user branch address is normal 1: Setting of user branch address is abnormal
1	FQ	—	R/W	Frequency Error Detect Compares the specified CPU operating frequency with the operating frequencies supported by this LSI, and returns the result. 0: Setting of operating frequency is normal 1: Setting of operating frequency is abnormal
0	SF	—	R/W	Success/Fail Returns the initialization result. 0: Initialization has ended normally (no error) 1: Initialization has ended abnormally (error occurs)

(b)   Programming

FPFR indicates the return value of the programming result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	—	WD	WA	SF
Initial Value	—	—	—	—	—	—	—	—
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused
				Returns 0.
6	MD	—	R/W	Programming Mode Related Setting Error Detect
				Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 22.8.3, Error Protection.
				0: Normal operation (FLER = 0)
				1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Programming Execution Error Detect
				This bit is set to 1 when the specified data could not be written to the user MAT because the user MAT was not erased, or when any of the flash-related registers have been rewritten at the point the execution has returned from the user branch processing. If this bit is set to 1, there is a high possibility that the user MAT has been written to partially. In this case, after removing the error factor, erase the user MAT and program again.
				0: Programming has ended normally
				1: Programming has ended abnormally (programming result is not guaranteed)
4	FK	—	R/W	Flash Key Register Error Detect
				Checks the FKEY value (H'5A) before programming starts, and returns the result.
				0: FKEY setting is normal (H'5A)
				1: FKEY setting is abnormal (value other than H'5A)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	—	—	—	Unused
				Returns 0.
2	WD	—	R/W	Write Data Address Detect
				When an address not in the flash memory area is specified as the start address of the storage destination for the program data, an error occurs.
				0: Setting of the start address of the storage destination for the program data is normal
				1: Setting of the start address of the storage destination for the program data is abnormal
1	WA	—	R/W	Write Address Error Detect
				When the following items are specified as the start address of the programming destination, an error occurs.
				• An area other than flash memory
				• The specified address is not aligned with the 128-byte boundary (lower eight bits of the address are other than H'00 and H'80)
				0: Setting of the start address of the programming destination is normal
				1: Setting of the start address of the programming destination is abnormal
0	SF	—	R/W	Success/Fail
				Returns the programming result.
				0: Programming has ended normally (no error)
				1: Programming has ended abnormally (error occurs)

RENESAS

(c)   Erasure

FPFR indicates the return value of the erasure result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	EB	—	—	SF
Initial Value	—	—	—	—	—	—	—	—
R/W	—	R/W	R/W	R/W	R/W	—	—	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused
				Returns 0.
6	MD	—	R/W	Erasure Mode Related Setting Error Detect
				Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 22.8.3, Error Protection.
				0: Normal operation (FLER = 0)
				1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Erasure Execution Error Detect
				This bit is set to 1 when the user MAT could not be erased or when any of the flash-related registers have been rewritten at the point the execution has returned from the user branch processing. If this bit is set to 1, there is a high possibility that the user MAT has been erased partially. In this case, after removing the error factor, erase the user MAT.
				0: Erasure has ended normally
				1: Erasure has ended abnormally
4	FK	—	R/W	Flash Key Register Error Detect
				Checks the FKEY value (H'5A) before erasure starts, and returns the result.
				0: FKEY setting is normal (H'5A)
				1: FKEY setting is abnormal (value other than H'5A)

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
3	EB	—	R/W	Erase Block Select Error Detect
				Checks whether the specified erase block number is in the block range of the user MAT, and returns the result.
				0: Setting of erase block number is normal
				1: Setting of erase block number is abnormal
2, 1	—	—	—	Unused
				These bits return 0.
0	SF	—	R/W	Success/Fail
				Indicates the erasure result.
				0: Erasure has ended normally (no error)
				1: Erasure has ended abnormally (error occurs)

(3)   Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU)

FPEFEQ sets the operating frequency of the CPU and enables the user branch function. The operating frequency available in this LSI ranges from 8 MHz to 40 MHz.

Bit	31	30	29	28	27	26	25	24
Bit Name	FUBE15	FUBE14	FUBE13	FUBE12	FUBE11	FUBE10	FUBE9	FUBE8
Initial Value	—	—	—	—	—	—	—	—
R/W	—	—	—	—	—	—	—	—
Bit	23	22	21	20	19	18	17	16
Bit Name	FUBE7	FUBE6	FUBE5	FUBE4	FUBE3	FUBE2	FUBE1	FUBE0
Initial Value	—	—	—	—	—	—	—	—
R/W	—	—	—	—	—	—	—	—
Bit	15	14	13	12	11	10	9	8
Bit Name	F15	F14	F13	F12	F11	F10	F9	F8
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	F7	F6	F5	F4	F3	F2	F1	F0
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	FUBE15 to FUBE0	—	R/W	Flash User Branch Enable
				To enable the user branch function, set these bits to H'AA55. In other cases, set these bits to H'0000.
15 to 0	F15 to F0	—	R/W	Frequency Set
				These bits set the operating frequency of the CPU. When the PLL multiplication function is used, set the multiplied frequency. The setting value must be calculated as follows:
				1. Round off the operating frequency expressed in MHz unit at the third decimal place to make it into two decimal places.
				2. Multiply the rounded number by 100 and convert the result into binary and write it to FPEFEQ (general register ER0).
				For example, when the operating frequency of the CPU is 40.000 MHz, the setting value is as follows:
				1. Round 40.000 off at the third decimal place.
				2. Convert $40.00 \times 100 = 4000$ into a binary number and set B'0000 1111 1100 0000 (H'0FA0) in ER0.

RENESAS

(4)   Flash Multipurpose Address Area Parameter (FMPAR: General Register ER1 of CPU)

FMPAR stores the start address of the programming destination on the user MAT.

When an address in an area other than the flash memory is set, or the start address of the programming destination is not aligned with the 128-byte boundary, an error occurs. The error occurrence is indicated by the WA bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOA31	MOA30	MOA29	MOA28	MOA27	MOA26	MOA25	MOA24
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	MOA23	MOA22	MOA21	MOA20	MOA19	MOA18	MOA17	MOA16
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	MOA15	MOA14	MOA13	MOA12	MOA11	MOA10	MOA9	MOA8
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MOA7	MOA6	MOA5	MOA4	MOA3	MOA2	MOA1	MOA0
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOA31 to MOA0	—	R/W	These bits store the start address of the programming destination on the user MAT. Consecutive 128-byte programming is executed starting from the specified start address of the user MAT. Therefore, the specified start address of the programming destination becomes a 128-byte boundary, and MOA6 to MOA0 are always cleared to 0.

(5) Flash Multipurpose Data Destination Parameter (FMPDR: General Register ER0 of CPU)

FMPDR stores the start address in the area which stores the data to be programmed in the user MAT.

When the storage destination for the program data is in flash memory, an error occurs. The error occurrence is indicated by the WD bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOD31	MOD30	MOD29	MOD28	MOD27	MOD26	MOD25	MOD24
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	MOD23	MOD22	MOD21	MOD20	MOD19	MOD18	MOD17	MOD16
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	MOD15	MOD14	MOD13	MOD12	MOD11	MOD10	MOD9	MOD8
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MOD7	MOD6	MOD5	MOD4	MOD3	MOD2	MOD	MOD0
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MOD0	—	R/W	These bits store the start address of the area which stores the program data for the user MAT. Consecutive 128-byte data is programmed to the user MAT starting from the specified start address.

RENESAS

(6)   Flash Erase Block Select Parameter (FEBS: General Register ER0 of CPU)

FEBS specifies the erase block number. Settable values range from 0 to 15 (H'00000000 to H'0000000F). A value of 0 corresponds to block EB0 and a value of 15 corresponds to block EB15. An error occurs when a value over the range (from 0 to 15) is set.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	EBS7	EBS6	EBS5	EBS4	EBS3	EBS2	EBS1	EBS0
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	Undefined	R/W	Unused
				These bits should be set to 0.
7 to 0	EBS7 to EBS0	Undefined	R/W	These bits specify the erase block number from 0 to 15. A value of 0 corresponds to block EB0 and 15 corresponds to block EB15. An error occurs if a value outside the range from 0 to 15 (from H'00 to H'0F) is set.

(7)   Flash User Branch Address Set Parameter (FUBRA: General Register ER0 of CPU)

FUBRA specifies the start address of the routine at the user branch destination. A specified user program can be executed at points in between certain units of processing during execution of programming/erasing.

If user branching is not necessary, set address 0 (H'00000000) in this parameter.

Addresses settable as the user branch destination are those not in the on-chip flash memory or the RAM area to which the on-chip program is transferred.

Be careful not to jump to an address where there is no code for execution. Jumping to such areas can cause the LSI to go out of control and damage the contents in the on-chip program downloaded area or the stack area. The values in the flash memory cannot be guaranteed if the LSI goes out of control or the on-chip program downloaded area or stack area is damaged.

The user routine at the specified branch address must not include processing to initiate the programs for downloading the on-chip program, initialization, or programming/erasing. If these programs are initiated, the programming/erasing processing that is to be performed after returning from the user branch routine cannot be guaranteed. Also note that the data for programming that has already been set should not be rewritten.

General registers ER2 to ER7 should be saved, while ER0 and ER1 can be used without saving.

In addition, the user routine at the branch address must not change the values of the programming/erasing interface registers or must not cause the LSI to enter RAM emulation mode

After the user branch processing has ended, execute the RTS instruction to return to the programming/erasing program.

RENESAS

Bit	31	30	29	28	27	26	25	24
Bit Name	UA31	UA30	UA29	UA28	UA27	UA26	UA25	UA24
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	UA23	UA22	UA21	UA20	UA19	UA18	UA17	UA16
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	UA15	UA14	UA13	UA12	UA11	UA10	UA9	UA8
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	UA7	UA6	UA5	UA4	UA3	UA2	UA1	UA0
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	UA31 to UA0	—	R/W	These bits specify the start address of the routine at the user branch destination.

RENESAS

22.6.3   RAM Emulation Register (RAMER)

RAMER specifies the user MAT area overlaid with part of the on-chip RAM (H'FFA000 to H'FFAFFF) when performing emulation of programming the user MAT. RAMER should be set in user mode or user program mode. To ensure dependable emulation, the memory MAT to be emulated must not be accessed immediately after changing the RAMER contents. When accessed at such a timing, correct operation is not guaranteed.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	RAMS	RAM2	RAM1	RAM0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	0	R	Reserved
				These are read-only bits and cannot be modified.
3	RAMS	0	R/W	RAM Select
				Selects the function which emulates the flash memory using the on-chip RAM.
				0: Disables RAM emulation function
				1: Enables RAM emulation function (all blocks of the user MAT are protected against programming and erasing)
2	RAM2	0	R/W	Flash Memory Area Select
1	RAM1	0	R/W	These bits select the user MAT area overlaid with the
0	RAM0	0	R/W	on-chip RAM when RAMS = 1. The following areas correspond to the 4-kbyte erase blocks.
				000: H'000000 to H'000FFF (EB0)
				001: H'001000 to H'001FFF (EB1)
				010: H'002000 to H'002FFF (EB2)
				011: H'003000 to H'003FFF (EB3)
				100: H'004000 to H'004FFF (EB4)
				101: H'005000 to H'005FFF (EB5)
				110: H'006000 to H'006FFF (EB6)
				111: H'007000 to H'007FFF (EB7)

RENESAS

22.7   On-Board Programming Mode

When the mode pins (MD0, MD1, and MD2) are set to on-board programming mode and the reset start is executed, a transition is made to on-board programming mode in which the on-chip flash memory can be programmed/erased. On-board programming mode has three operating modes: boot mode, and user program mode.

Table 22.5 shows the pin setting for each operating mode. For details on the state transition of each operating mode for flash memory, see figure 22.2.

Table 22.5   On-Board Programming Mode Setting

Mode Setting	MD2	MD1	MD0
Boot mode	0	1	0
User program mode	1	1	—

22.7.1   Boot Mode

Boot mode executes programming/erasing of the user MAT by means of the control command and program data transmitted from the externally connected host via the on-chip SCI_4.

In boot mode, the tool for transmitting the control command and program data, and the program data must be prepared in the host. The serial communication mode is set to asynchronous mode. The system configuration in boot mode is shown in figure 22.6. Interrupts are ignored in boot mode. Configure the user system so that interrupts do not occur.
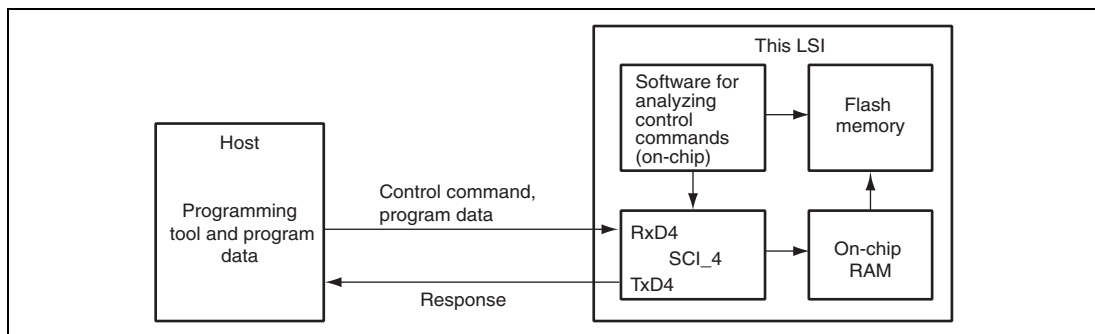
Figure 22.6   System Configuration in Boot Mode

(1)   Serial Interface Setting by Host

The SCI_4 is set to asynchronous mode, and the serial transmit/receive format is set to 8-bit data, one stop bit, and no parity.

When a transition to boot mode is made, the boot program embedded in this LSI is initiated.

When the boot program is initiated, this LSI measures the low period of asynchronous serial communication data (H'00) transmitted consecutively by the host, calculates the bit rate, and adjusts the bit rate of the SCI_4 to match that of the host.

When bit rate adjustment is completed, this LSI transmits 1 byte of H'00 to the host as the bit adjustment end sign. When the host receives this bit adjustment end sign normally, it transmits 1 byte of H'55 to this LSI. When reception is not executed normally, initiate boot mode again. The bit rate may not be adjusted within the allowable range depending on the combination of the bit rate of the host and the system clock frequency of this LSI. Therefore, the transfer bit rate of the host and the system clock frequency of this LSI must be as shown in table 22.6.
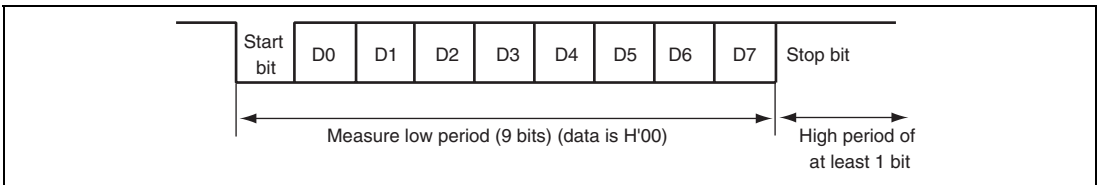
Figure 22.7   Automatic-Bit-Rate Adjustment Operation

Table 22.6   System Clock Frequency for Automatic-Bit-Rate Adjustment

Bit Rate of Host	System Clock Frequency of This LSI
9,600 bps	8 to 18 MHz
19,200 bps	8 to 18 MHz

RENESAS

(2)   State Transition Diagram

The state transition after boot mode is initiated is shown in figure 22.8.
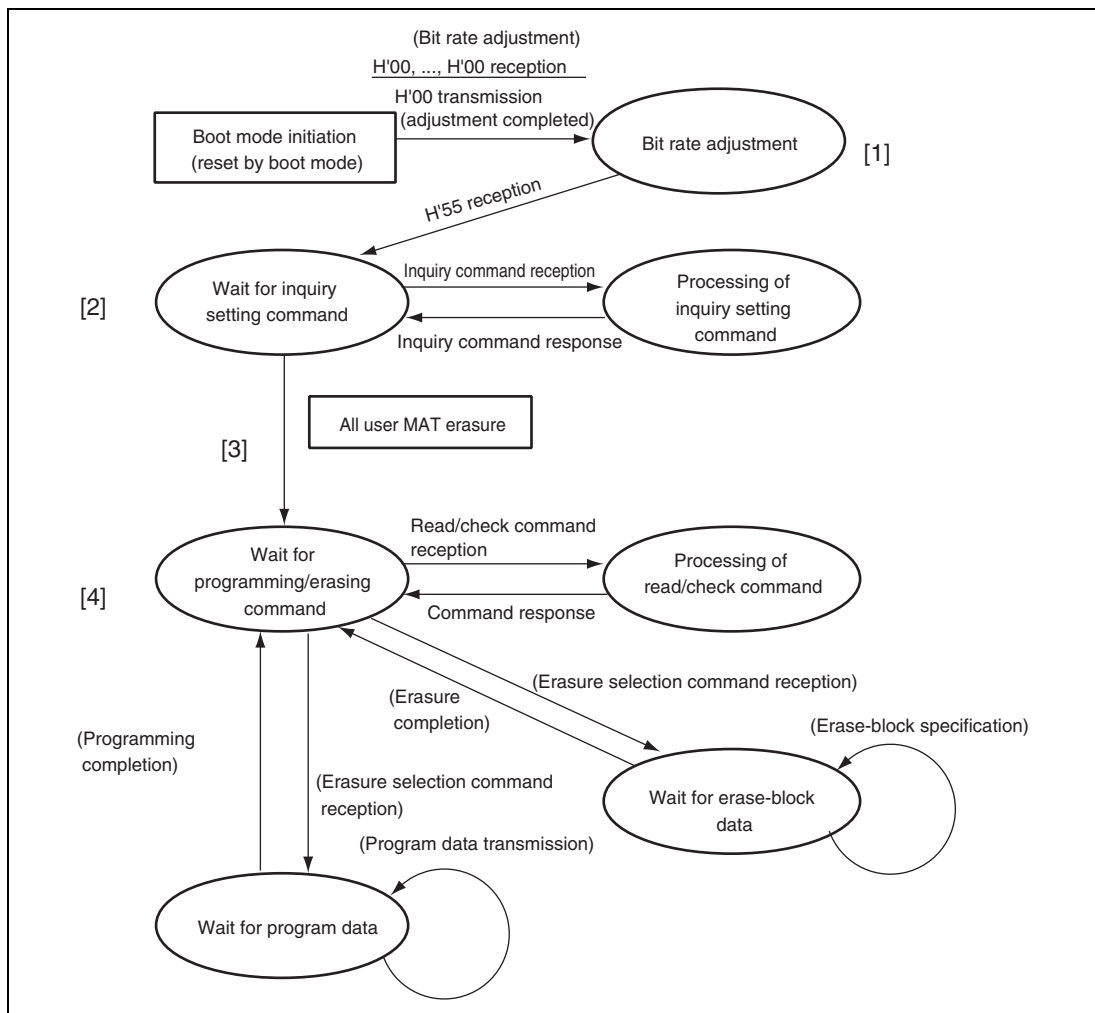
Figure 22.8   Boot Mode State Transition Diagram

[1] After boot mode is initiated, the bit rate of the SCI_4 is adjusted with that of the host.

[2] Inquiry information about the size, configuration, start address, and support status of the user MAT is transmitted to the host.

[3] After inquiry setting have finished, all user MAT are automatically erased.

[4] When a programming selection command is received, the state of waiting for program data is entered. The start address of the programming destination and program data must be transmitted after the programming command is transmitted. When programming is finished, the start address for programming must be set to H'FFFFFFFF and transmitted. Then the state of waiting for program data returns to the state of waiting for a programming/erasing command. When an erasing selection command is received, the state of waiting for erase block data is entered. The erase block number must be transmitted after the erasing command is transmitted. When the erasure is finished, the erase block number must be set to H'FF and transmitted. Then the state of waiting for erase block data is returned to the state of waiting for programming/erasing command. Erasure must be executed when the specified block is programmed without a reset start after programming is executed in boot mode. When programming can be executed by only one operation, all blocks are erased before entering the state of waiting for programming/erasing command or another command. Thus, in this case, the erasing operation is not required.

The commands other than the programming/erasing command are for sum check, blank check (erasure check), memory read of the user MAT, and acquisition of current status information.

Memory read of the user MAT can only read the data programmed after all user MAT has automatically been erased. No other data can be read.

RENESAS

22.7.2   User Program Mode

Programming/erasing of the user MAT is executed by downloading an on-chip program. The programming/erasing flow is shown in figure 22.9.

Since high voltage is applied to the internal flash memory during programming/erasing, a transition to the reset state or hardware standby mode must not be made during programming/erasing. A transition to the reset state or hardware standby mode during programming/erasing may damage the flash memory. If a reset is input, the reset must be released after the reset input period (period of $\overline{\text{RES}} = 0$) of at least 100 μs.

For the procedure of programming, see section 22.7.2 (2), Programming Procedure in User Program Mode. For the procedure of erasure, see section 22.7.2 (3), Erasing Procedure in User Program Mode.

```
┌─────────────────────────┐   [1]  Exit RAM emulation mode beforehand. Download is not allowed
│  Programming/erasing    │        in emulation mode.
│        start            │
└─────────────────────────┘   [2]  When the program data is adjusted in emulation mode, select
┌─────────────────────────┐        the download destination specified by FTDAR carefully. Make
│   When programming,     │        sure that the download area does not overlap the emulation
│  program data is        │        area.
│      prepared           │
└─────────────────────────┘   [3]  Programming/erasing is executed only in the on-chip RAM.
┌─────────────────────────┐
│ Programming/erasing     │   [4]  After programming/erasing is finished, protect the flash memory
│ procedure program is    │        by the hardware protection.
│ transferred to the on-  │
│ chip RAM and executed   │
└─────────────────────────┘
┌─────────────────────────┐
│  Programming/erasing    │
│        end              │
└─────────────────────────┘
```

Figure 22.9   Programming/Erasing Flow

(1)   On-Chip RAM Address Map when Programming/Erasing Is Executed

Parts of the procedure program that is made by the user, like download request, programming/erasing procedure, and decision of the result, must be executed in the on-chip RAM. Since the on-chip program to be downloaded is embedded in the on-chip RAM, make sure the on-chip program and procedure program do not overlap. Figure 22.10 shows the area of the on-chip program to be downloaded.
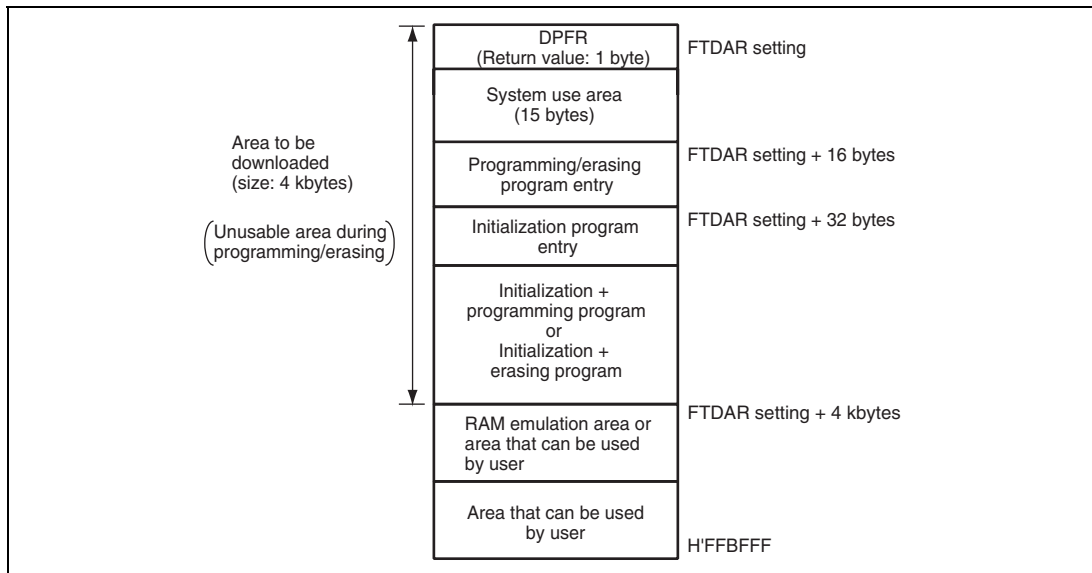
Figure 22.10   RAM Map when Programming/Erasing Is Executed

(2)   Programming Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and programming are shown in figure 22.11.
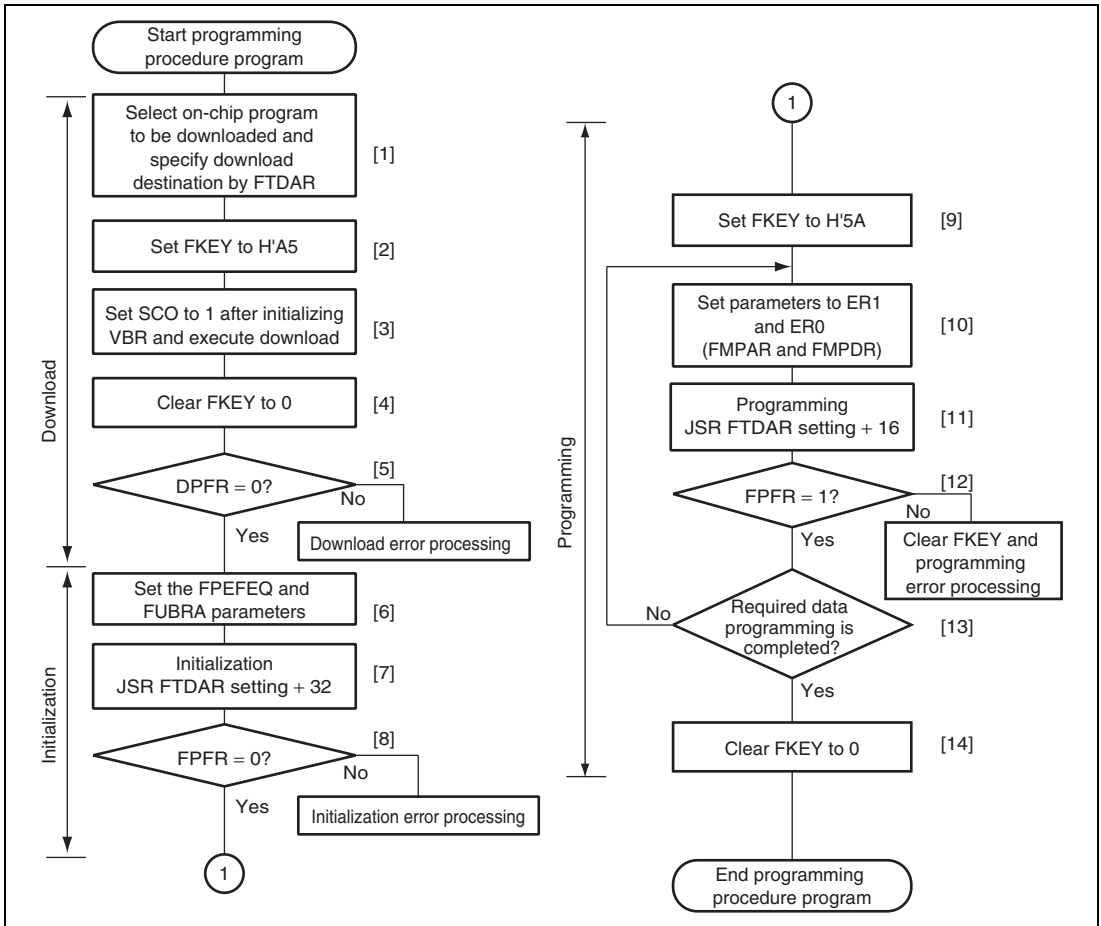
Figure 22.11   Programming Procedure in User Program Mode

The procedure program must be executed in an area other than the flash memory to be programmed. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 22.7.3, On-Chip Program and Storable Area for Program Data. The following description assumes that the area to be programmed on the user MAT is erased and that program data is prepared in the consecutive area.

The program data for one programming operation is always 128 bytes. When the program data exceeds 128 bytes, the start address of the programming destination and program data parameters are updated in 128-byte units and programming is repeated. When the program data is less than 128 bytes, invalid data is filled to prepare 128-byte program data. If the invalid data to be added is H'FF, the program processing time can be shortened.

[1] Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.

[2] Write H'A5 in FKEY. If H'A5 is not written to FKEY, the SCO bit in FCCS cannot be set to 1 to request download of the on-chip program.

[3] After initializing VBR to H'00000000, set the SCO bit to 1 to execute download. To set the SCO bit to 1, all of the following conditions must be satisfied.

— RAM emulation mode has been canceled.

— H'A5 is written to FKEY.

— Setting the SCO bit is executed in the on-chip RAM.

When the SCO bit is set to 1, download is started automatically. Since the SCO bit is cleared to 0 when the procedure program is resumed, the SCO bit cannot be confirmed to be 1 in the procedure program. The download result can be confirmed by the return value of the DPFR parameter. To prevent incorrect decision, before setting the SCO bit to 1, set one byte of the on-chip RAM start address specified by FTDAR, which becomes the DPFR parameter, to a value other than the return value (e.g. H'FF). Since particular processing that is accompanied by bank switching as described below is performed when download is executed, initialize the VBR contents to H'00000000. Dummy read of FCCS must be performed twice immediately after the SCO bit is set to 1.

RENESAS

- The user-MAT space is switched to the on-chip program storage area.
- After the program to be downloaded and the on-chip RAM start address specified by FTDAR are checked, they are transferred to the on-chip RAM.
- General registers ER0 and ER1 are used in downloading. If the data in ER0 and ER1 before the SCO bit s set should be retained, save their contents to the stack before setting the SCO bit and restore them after the downloading is finished.
- FPCS, FECS, and the SCO bit in FCCS are cleared to 0.
- The return value is set in the DPFR parameter.
- After the on-chip program storage area is returned to the user-MAT space, the execution returns to the user procedure program. After returned, VBR can be set again.

[Notes on Downloading]

— The values of general registers of the CPU except for ER0 and ER1 are retained.

— During download, no interrupts can be accepted. However, since the interrupt requests are held, when the procedure program is resumed, the interrupts are requested.

— To hold a level-detection interrupt request, the interrupt must continue to be input until the download is completed.

— Allocate a stack area of 128 bytes at the maximum in the on-chip RAM before setting the SCO bit to 1.

— If access to the flash memory is requested by the DMAC during download, the operation cannot be guaranteed. Make sure that an access request by the DMAC is not generated.

[4] Clear FKEY to H'00 for protection.

[5] Check the value of the DPFR parameter (value of the byte at the start address for downloading specified by FTDAR) for the result of the downloading. If the value of DPFR is H'00, downloading has been performed normally. If the value is not H'00, carry out the following procedure to find out the cause of downloading failure.

— If the value of the DPFR parameter is the same as that before downloading, the setting of the start address of the download destination in FTDAR may be abnormal. In this case, confirm the setting of the TDER bit in FTDAR.

— If the value of the DPFR parameter is different from that before downloading, check the SS bit or FK bit in the DPFR parameter to confirm the download program selection and FKEY setting, respectively.

[6] For initialization, set the operating frequency of the CPU in the FPEFEQ parameter and set the user branch destination in the FUBRA parameter. The settable operating frequency of the FPEFEQ parameter ranges from 8 to 40 MHz. When the frequency is set otherwise, an error is returned to the FPFR parameter of the initialization program and initialization is not performed. For details on setting the frequency, see section 22.6.2 (3), Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU).

RENESAS

— A user branch destination address should be set in the FUBRA parameter. If user branching is not necessary, set H'00000000 in FUBRA. When performing user branching, do not specify an address in flash memory that is to be programmed as the user branch address. Also do not specify an address in the area where the on-chip program is downloaded. Use the RTS instruction to return from the user branch processing to the programming processing. For how to set the start address of the user branch routine, see section 22.6.2 (7), Flash User Branch Address Set Parameter (FUBRA: General Register ER0 of CPU).

[7] Execute initialization. The initialization program is downloaded together with the programming program to the on-chip RAM. The entry point of the initialization program is at the address which is 32 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute initialization by using the following steps.

```
MOV.L #DLTOP+32,ER2     ; Set entry address to ER2
JSR   @ER2              ; Call initialization routine
NOP
```

— The general registers other than ER0 and ER1 are held in the initialization program.

— R0L is a return value of the FPFR parameter.

— Since the stack area is used in the initialization program, a stack area of 128 bytes at the maximum must be allocated in RAM.

— Interrupts can be accepted during execution of the initialization program. Make sure the program storage area and stack area in the on-chip RAM and register values are not overwritten.

[8] Check the FPFR parameter, in which the return value of the initialization program is set.

[9] Set H'5A in FKEY to enable user MAT programming.

[10] Set the parameters required for programming.

Set the start address of the programming destination on the user MAT (FMPAR parameter) in general register ER1, and set the start address of the program data storage area (FMPDR parameter) in general register ER0.

— Example of FMPAR parameter setting: When an address other than one in the user MAT area is specified for the start address of the programming destination, even if the programming program is executed, programming is not executed and an error is returned to the FPFR parameter. Since the program data for one programming operation is 128 bytes, the lower eight bits of the address must be H'00 or H'80 to be aligned with the 128-byte boundary.

— Example of FMPDR parameter setting: When the storage destination for the program data is flash memory, even if the programming routine is executed, programming is not executed and an error is returned to the FPFR parameter. In this case, the program data must be transferred to the on-chip RAM and then programming must be executed.

RENESAS

[11]Programming is executed. The entry point of the programming program is at the address which is 16 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute programming by using the following steps.

```
MOV.L    #DLTOP+16,ER2   ; Set entry address to ER2
JSR      @ER2            ; Call programming routine
NOP
```

—— The general registers other than ER0 and ER1 are held in the programming program.

—— R0L is a return value of the FPFR parameter.

—— Since the stack area is used in the programming program, a stack area of 128 bytes at the maximum must be allocated in RAM.

[12]Check the FPFR parameter, in which the return value of the programming program is set.

[13]Determine whether programming of the necessary data has finished. If more than 128 bytes of data are to be programmed, update the FMPAR and FMPDR parameters in 128-byte units, and repeat steps [11] to [14]. Increment the programming destination address by 128 bytes and update the programming data pointer correctly. If an address which has already been programmed is written to again, not only will a programming error occur, but also flash memory will be damaged.

[14]After programming has finished, clear FKEY to specify software protection. If this LSI is restarted by a reset immediately after programming has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.

RENESAS

(3)   Erasing Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and erasing are shown in figure 22.12.
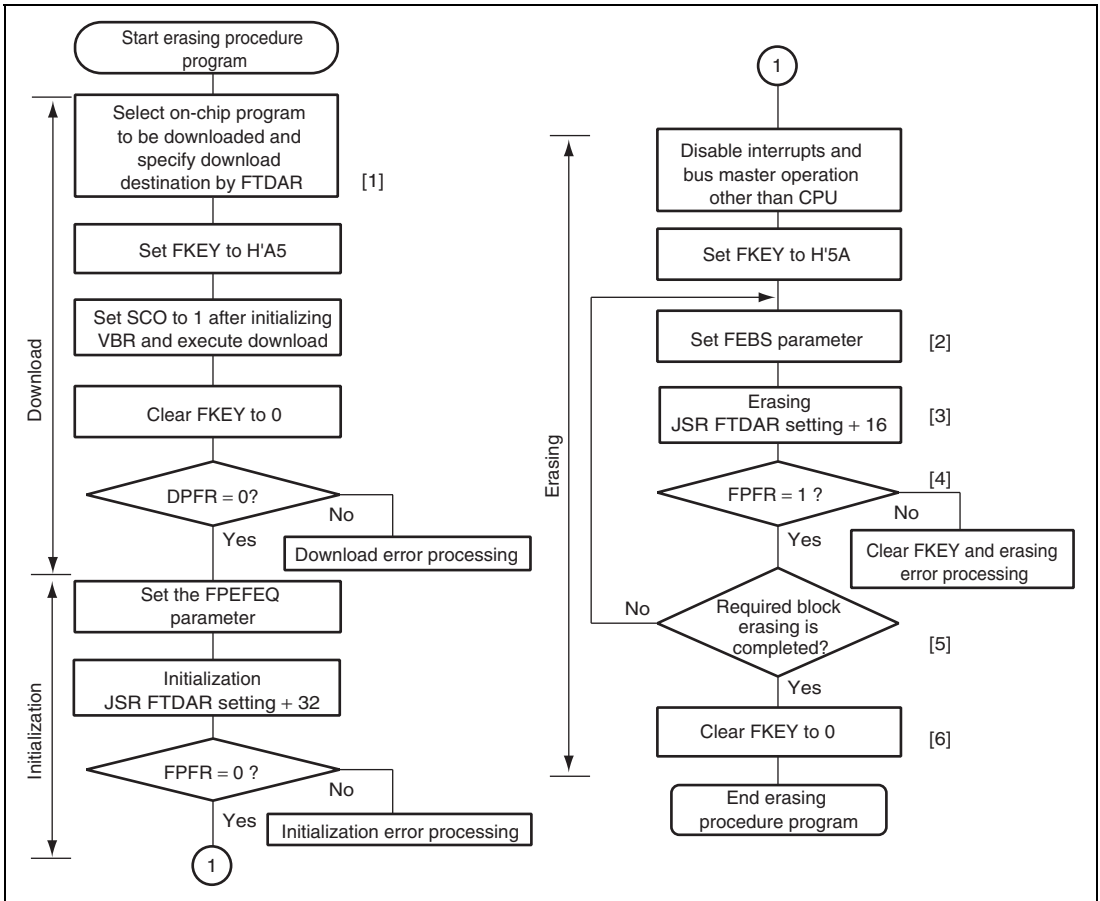
Figure 22.12   Erasing Procedure in User Program Mode

The procedure program must be executed in an area other than the user MAT to be erased. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM and user MAT) is shown in section 22.7.3, On-Chip Program and Storable Area for Program Data. For the downloaded on-chip program area, see figure 22.10.

One erasure processing erases one block. For details on block divisions, refer to figure 22.3 or figure 22.4. To erase two or more blocks, update the erase block number and repeat the erasing processing for each block.

[1] Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.

For the procedures to be carried out after setting FKEY, see section 22.7.2 (2), Programming Procedure in User Program Mode.

[2] Set the FEBS parameter necessary for erasure. Set the erase block number (FEBS parameter) of the user MAT in general register ER0. If a value other than an erase block number of the user MAT is set, no block is erased even though the erasing program is executed, and an error is returned to the FPFR parameter.

[3] Execute erasure. Similar to as in programming, the entry point of the erasing program is at the address which is 16 bytes after #DLTOP (start address of the download destination specified by FTDAR). Call the subroutine to execute erasure by using the following steps.

```
MOV.L #DLTOP+16, ER2     ; Set entry address to ER2
JSR   @ER2               ; Call erasing routine
NOP
```

- The general registers other than 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] Check the FPFR parameter, in which the return value of the erasing program is set.

[5] Determine whether erasure of the necessary blocks has finished. If more than one block is to be erased, update the FEBS parameter and repeat steps [2] to [5].

[6] After erasure completes, clear FKEY to specify software protection. If this LSI is restarted by a power-on reset immediately after erasure has finished, secure the reset input period (period of $\overline{\text{RES}}$ = 0) of at least 100 μs.

(4)   Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

By changing the on-chip RAM start address of the download destination in FTDAR, the erasing program and programming program can be downloaded to separate on-chip RAM areas.

Figure 22.13 shows a repeating procedure of erasing, programming, and RAM emulation.
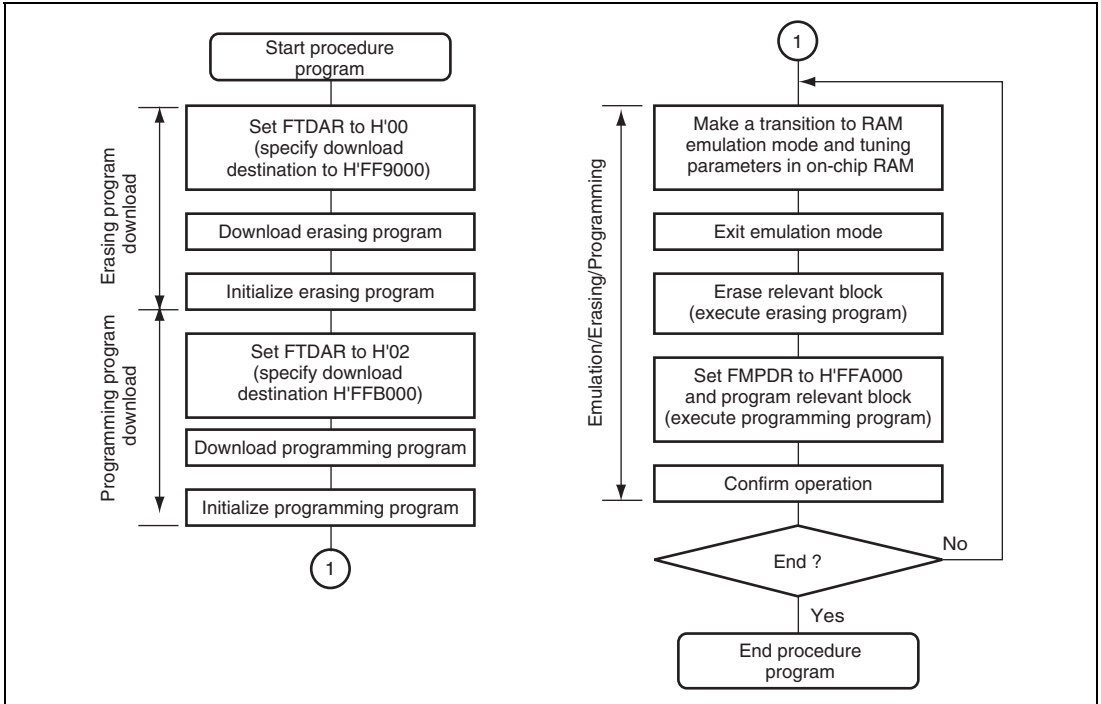
Figure 22.13   Repeating Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

In figure 22.13, since RAM emulation is performed, the erasing/programming program is downloaded to avoid the 4-kbyte on-chip RAM area (H'FFA000 to H'FFAFFF). Download and initialization are performed only once at the beginning. Note the following when executing the procedure program.

- Be careful not to overwrite data in the on-chip RAM with overlay settings. In addition to the programming program area, erasing program area, and RAM emulation area, areas for the procedure programs, work area, and stack area are reserved in the on-chip RAM. Do not make settings that will overwrite data in these areas.

RENESAS

- Be sure to initialize both the programming program and erasing program. When the FPEFEQ parameter is initialized, also initialize both the erasing program and programming program. Initialization must be executed for both entry addresses: #DLTOP (start address of download destination for erasing program) + 32 bytes, and #DLTOP (start address of download destination for programming program) + 32 bytes.

22.7.3   On-Chip Program and Storable Area for Program Data

In the descriptions in this manual, the on-chip programs and program data storage areas are assumed to be in the on-chip RAM. However, they can be executed from part of the flash memory which is not to be programmed or erased as long as the following conditions are satisfied.

- The on-chip program is downloaded to and executed in the on-chip RAM specified by FTDAR. Therefore, this on-chip RAM area is not available for use.
- Since the on-chip program uses a stack area, allocate 128 bytes at the maximum as a stack area.
- Download requested by setting the SCO bit in FCCS to 1 should be executed from the on-chip RAM because it will require switching of the memory MATs.
- In an operating mode in which the external address space is not accessible, such as single-chip mode, the required procedure programs, NMI handling vector table, and NMI handling routine should be transferred to the on-chip RAM before programming/erasing starts (download result is determined).
- The flash memory is not accessible during programming/erasing. Programming/erasing is executed by the program downloaded to the on-chip RAM. Therefore, the procedure program that initiates operation, the NMI handling vector table, and the NMI handling routine should be stored in the on-chip RAM other than the flash memory.
- After programming/erasing starts, access to the flash memory should be inhibited until FKEY is cleared. The reset input state (period of $\overline{\text{RES}} = 0$) must be set to at least 100 μs when the operating mode is changed and the reset start executed on completion of programming/erasing. Transitions to the reset state are inhibited during programming/erasing. When the reset signal is input, a reset input state (period of $\overline{\text{RES}} = 0$) of at least 100 μs is needed before the reset signal is released.
- When the program data storage area is within the flash memory area, an error will occur even when the data stored is normal program data. Therefore, the data should be transferred to the on-chip RAM to place the address that the FMPDR parameter indicates in an area other than the flash memory.

In consideration of these conditions, the areas in which the program data can be stored and executed are determined by the combination of the processing contents, operating mode, and bank structure of the memory MATs, as shown in tables 22.7 to 22.9.

Table 22.7   Executable Memory MAT

	Operating Mode	
Processing Contents	**User Program Mode**	
Programming	See table 22.8	
Erasing	See table 22.9	

Table 22.8   Usable Area for Programming in User Program Mode

	Storable/Executable Area		Selected MAT	
Item	**On-Chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage MAT**
Storage area for program data	O	×*	—	—
Operation for selecting on-chip program to be downloaded	O	O	O	
Operation for writing H'A5 to FKEY	O	O	O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×		O
Operation for clearing FKEY	O	O	O	
Decision of download result	O	O	O	
Operation for download error	O	O	O	
Operation for setting initialization parameter	O	O	O	
Execution of initialization	O	×	O	
Decision of initialization result	O	O	O	
Operation for initialization error	O	O	O	
NMI handling routine	O	×	O	
Operation for disabling interrupts	O	O	O	
Operation for writing H'5A to FKEY	O	O	O	
Operation for setting programming parameter	O	×	O	
Execution of programming	O	×	O	
Decision of programming result	O	×	O	
Operation for programming error	O	×	O	
Operation for clearing FKEY	O	×	O	

Note: *   Transferring the program data to the on-chip RAM beforehand enables this area to be used.

RENESAS

Table 22.9   Usable Area for Erasure in User Program Mode

	Storable/Executable Area		Selected MAT	
Item	On-Chip RAM	User MAT	User MAT	Embedded Program Storage MAT
Operation for selecting on-chip program to be downloaded	O	O	O	
Operation for writing H'A5 to FKEY	O	O	O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×		O
Operation for clearing FKEY	O	O	O	
Decision of download result	O	O	O	
Operation for download error	O	O	O	
Operation for setting initialization parameter	O	O	O	
Execution of initialization	O	×	O	
Decision of initialization result	O	O	O	
Operation for initialization error	O	O	O	
NMI handling routine	O	×	O	
Operation for disabling interrupts	O	O	O	
Operation for writing H'5A to FKEY	O	O	O	
Operation for setting erasure parameter	O	×	O	
Execution of erasure	O	×	O	
Decision of erasure result	O	×	O	
Operation for erasure error	O	×	O	
Operation for clearing FKEY	O	×	O	

22.8　Protection

There are three types of protection against the flash memory programming/erasing: hardware protection, software protection, and error protection.

22.8.1　Hardware Protection

Programming and erasure of the flash memory is forcibly disabled or suspended by hardware protection. In this state, download of an on-chip program and initialization are possible. However, programming or erasure of the user MAT cannot be performed even if the programming/erasing program is initiated, and the error in programming/erasing is indicated by the FPFR parameter.

Table 22.10 Hardware Protection

		Function to Be Protected	
Item	Description	Download	Programming/ Erasing
Reset protection	• The programming/erasing interface registers are initialized in the reset state (including a reset by the WDT) and the programming/erasing protection state is entered.	O	O
	• 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 settled after a power is initially supplied. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width given in the AC characteristics. If a reset is input during programming or erasure, data in the flash memory is not guaranteed. In this case, execute erasure and then execute programming again.		

22.8.2   Software Protection

The software protection protects the flash memory against programming/erasing by disabling download of the programming/erasing program, using the key code, and by the RAMER setting.

Table 22.11 Software Protection

Item	Description	Function to Be Protected	
		Download	Programming/ Erasing
Protection by SCO bit	The programming/erasing protection state is entered when the SCO bit in FCCS is cleared to 0 to disable download of the programming/erasing programs.	O	O
Protection by FKEY	The programming/erasing protection state is entered because download and programming/erasing are disabled unless the required key code is written in FKEY.	O	O
Emulation protection	The programming/erasing protection state is entered when the RAMS bit in the RAM emulation register (RAMER) is set to 1.	O	O

22.8.3   Error Protection

Error protection is a mechanism for aborting programming or erasure when a CPU runaway occurs or operations not according to the programming/erasing procedures are detected during programming/erasing of the flash memory. Aborting programming or erasure in such cases prevents damage to the flash memory due to excessive programming or erasing.

If an error occurs during programming/erasing of the flash memory, the FLER bit in FCCS is set to 1 and the error protection state is entered.

- When an interrupt request, such as NMI, occurs during programming/erasing.
- When the flash memory is read from during programming/erasing (including a vector read or an instruction fetch).
- When a SLEEP instruction is executed (including software-standby mode) during programming/erasing.
- When a bus master other than the CPU, such as the DMAC, obtains bus mastership during programming/erasing.

Error protection is canceled by a reset. Note that the reset should be released after the reset input period of at least 100μs has passed. 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 damaging the flash memory by extending the reset input period so that the charge is released.

The state-transition diagram in figure 22.14 shows transitions to and from the error protection state.
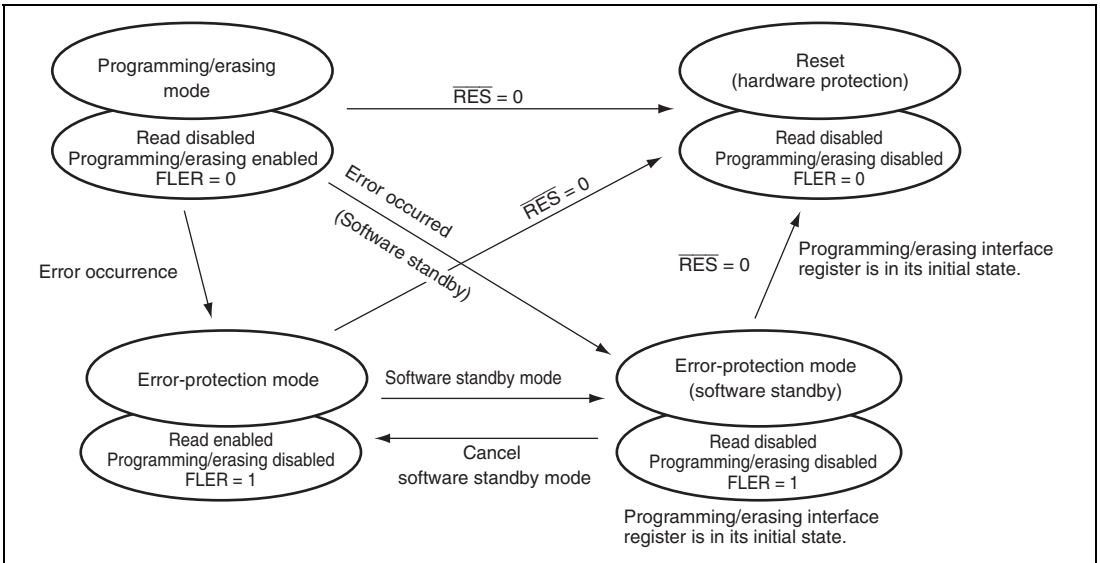
Figure 22.14   Transitions to Error Protection State

22.9   Flash Memory Emulation Using RAM

For realtime emulation of the data written to the flash memory using the on-chip RAM, the on-chip RAM area can be overlaid with several flash memory blocks (user MAT) using the RAM emulation register (RAMER).

The overlaid area can be accessed from both the user MAT area specified by RAMER and the overlaid RAM area. The emulation can be performed in user mode and user program mode.

Figure 22.15 shows an example of emulating realtime programming of the user MAT.

Note: To perform emulation of flash memory using RAM, set the RAME bit of the system
control register (SYSCR) to 1.
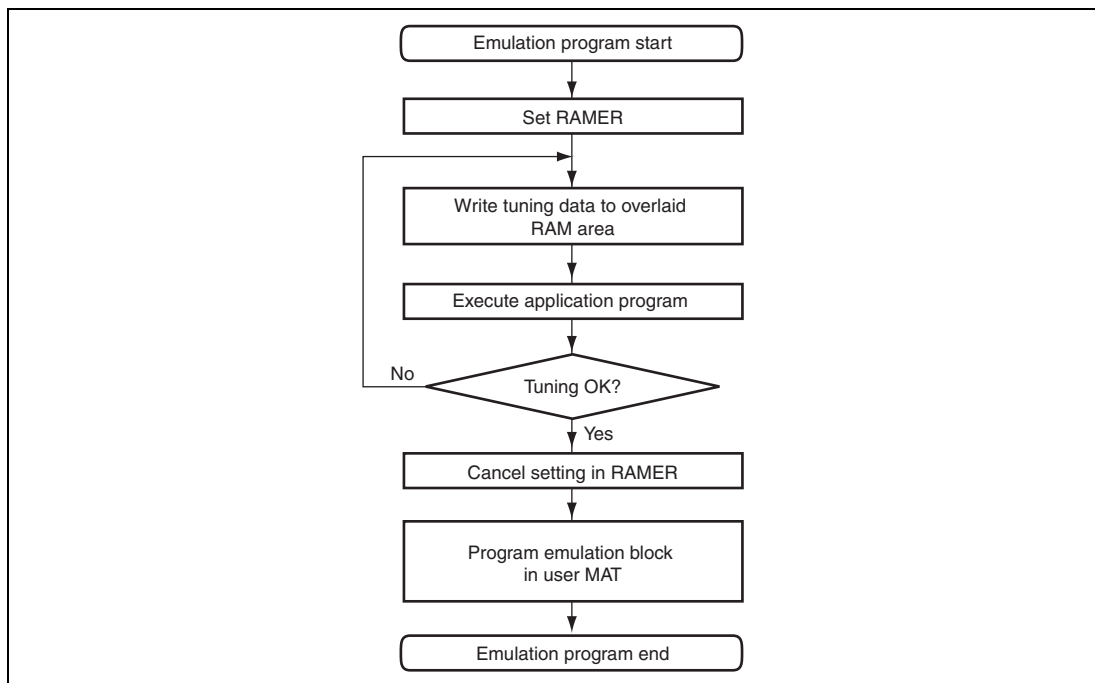
Figure 22.15   RAM Emulation Flow

Figure 22.16 shows an example of overlaying flash memory block area EB0.
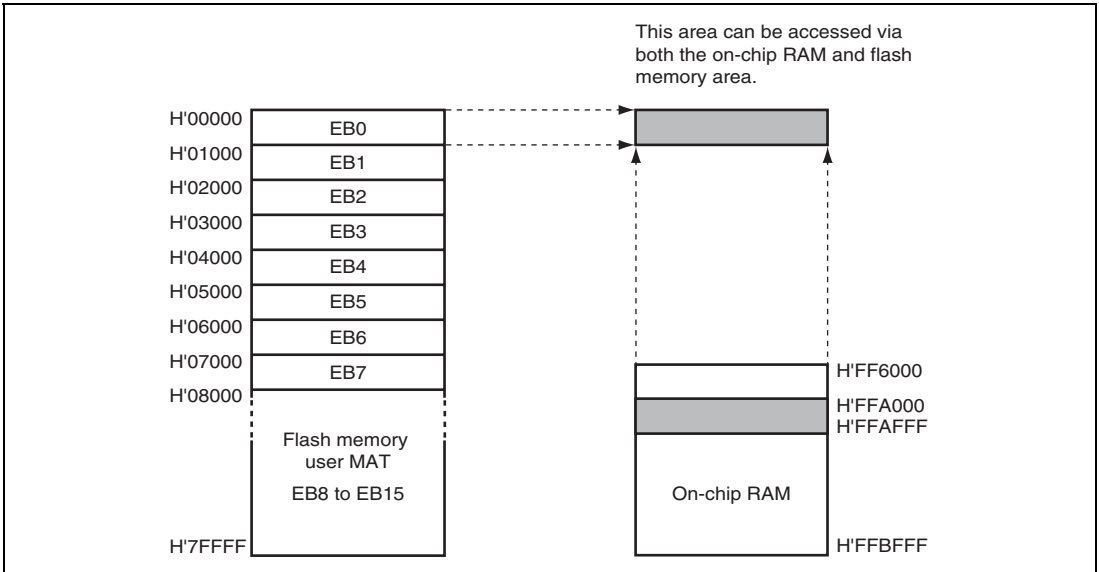
Figure 22.16   Address Map of Overlaid RAM Area

The flash memory area that can be emulated is the one area selected by bits RAM2 to RAM0 in RAMER from among the eight blocks, EB0 to EB7, of the user MAT.

To overlay a part of the on-chip RAM with block EB0 for realtime emulation, set the RAMS bit in RAMER to 1 and bits RAM2 to RAM0 to B'000.

For programming/erasing the user MAT, the procedure programs including a download program of the on-chip program must be executed. At this time, the download area should be specified so that the overlaid RAM area is not overwritten by downloading the on-chip program. Since the area in which the tuned data is stored is overlaid with the download area when FTDAR = H'01, the tuned data must be saved in an unused area beforehand.

RENESAS

Figure 22.17 shows an example of the procedure to program the tuned data in block EB0 of the user MAT.
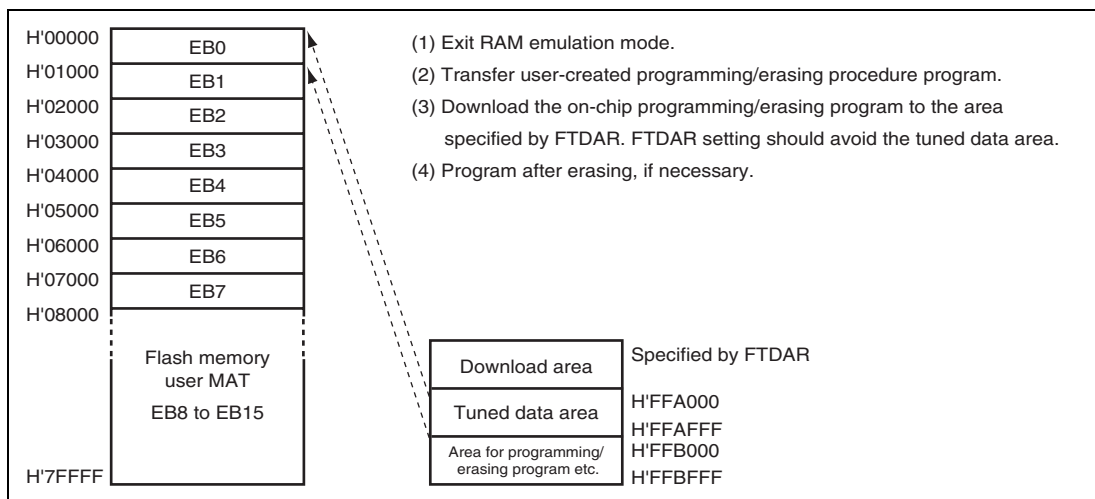
Figure 22.17   Programming Tuned Data

1. After tuning program data is completed, clear the RAMS bit in RAMER to 0 to cancel the overlaid RAM.

2. Transfer the user-created procedure program to the on-chip RAM.

3. Start the procedure program and download the on-chip program to the on-chip RAM. The start address of the download destination should be specified by FTDAR so that the tuned data area does not overlay the download area.

4. When block EB0 of the user MAT has not been erased, the programming program must be downloaded after block EB0 is erased. Specify the tuned data saved in the FMPAR and FMPDR parameters and then execute programming.

Note: Setting the RAMS bit to 1 makes all the blocks of the user MAT enter the programming/erasing protection state (emulation protection state) regardless of the setting of the RAM2 to RAM0 bits. Under this condition, the on-chip program cannot be downloaded. When data is to be actually programmed and erased, clear the RAMS bit to 0.

22.10   Programmer Mode

Along with its on-board programming mode, this LSI also has a programmer mode as a further mode for the writing and erasing of programs and data. In programmer mode, a general-purpose PROM programmer that supports the device types shown in table 22.12 can be used to write programs to the on-chip ROM without any limitation.

Table 22.12 Device Types Supported in Programmer Mode

Target Memory MAT	Size	Device Type
User MAT	512 kbytes	FZTAT512V5A
	384 kbytes	FZTAT512V5A

22.11   Standard Serial Communication Interface Specifications for Boot Mode

The boot program initiated in boot mode performs serial communication using the host and on-chip SCI_4. The serial communication interface specifications are shown below.

The boot program has three states.

1. Bit-rate-adjustment state

    In this state, the boot program adjusts the bit rate to achieve serial communication with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, the program enters the inquiry/selection state.

2. Inquiry/selection state

    In this state, the boot program responds to inquiry commands from the host. The device name, clock mode, and bit rate are selected. After selection of these settings, the program is made to enter the programming/erasing state by the command for a transition to the programming/erasing state. The program transfers the libraries required for erasure to the on-chip RAM and erases the user MATs before the transition.

3. Programming/erasing state

    Programming and erasure by the boot program take place in this state. The boot program is made to transfer the programming/erasing programs to the on-chip RAM by commands from the host. Sum checks and blank checks are executed by sending these commands from the host.

RENESAS

These boot program states are shown in figure 22.18.
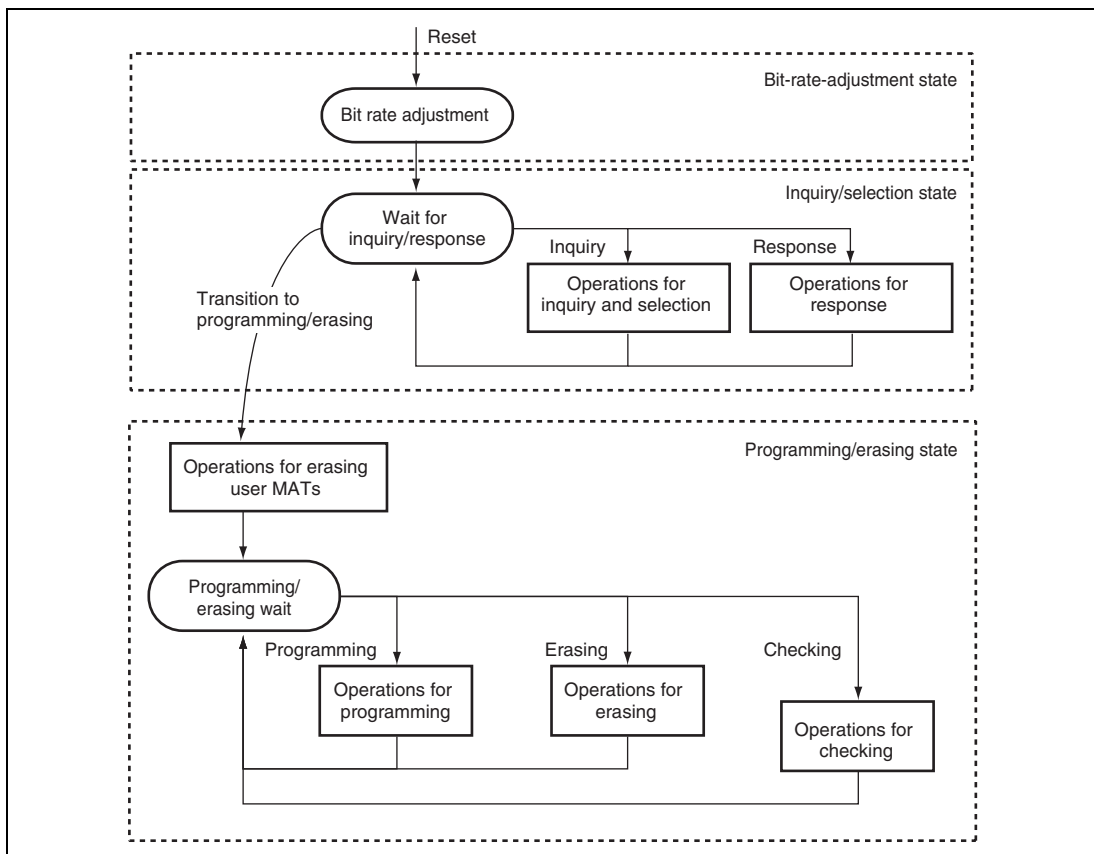
Figure 22.18   Boot Program States

(1)   Bit-Rate-Adjustment State

The bit rate is calculated by measuring the period of transfer of a low-level byte (H'00) from the host. The bit rate can be changed by the command for a new bit rate selection. After the bit rate has been adjusted, the boot program enters the inquiry and selection state. The bit-rate-adjustment sequence is shown in figure 22.19.
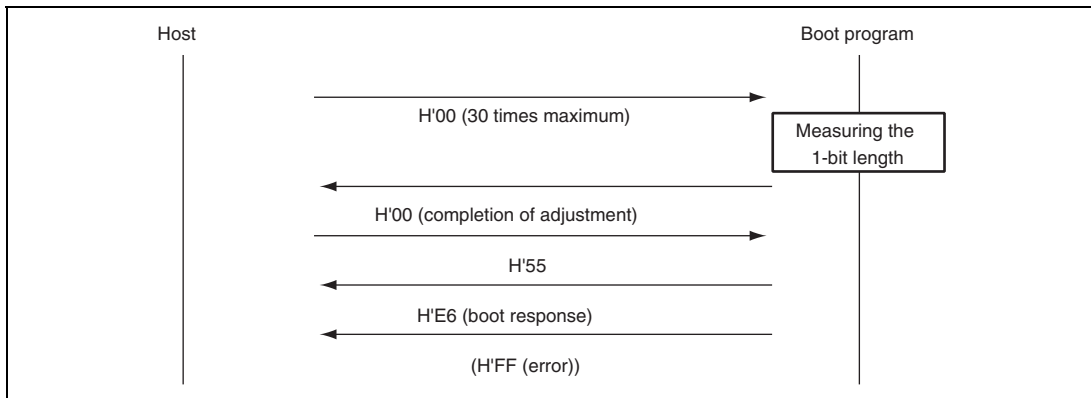
Figure 22.19   Bit-Rate-Adjustment Sequence

(2)   Communications Protocol

After adjustment of the bit rate, the protocol for serial communications between the host and the boot program is as shown below.

1. One-byte commands and one-byte responses

   These one-byte commands and one-byte responses consist of the inquiries and the ACK for successful completion.

2. n-byte commands or n-byte responses

   These commands and responses are comprised of n bytes of data. These are selections and responses to inquiries.

   The program data size is not included under this heading because it is determined in another command.

3. Error response

   The error response is a response to inquiries. It consists of an error response and an error code and comes two bytes.

RENESAS

4. Programming of 128 bytes

   The size is not specified in commands. The size of n is indicated in response to the programming unit inquiry.

5. Memory read response

   This response consists of four bytes of data.
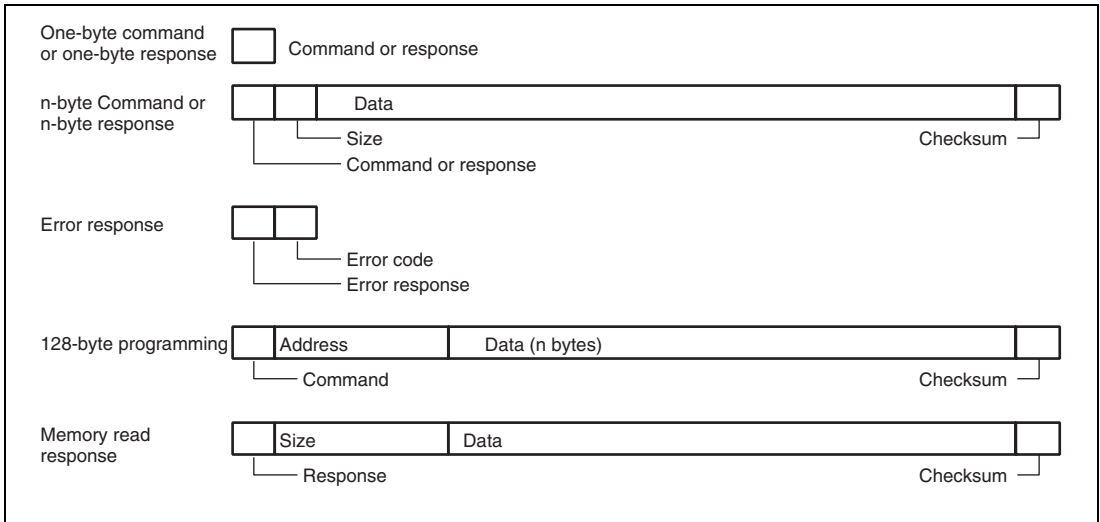
Figure 22.20   Communication Protocol Format

- Command (one byte): Commands including inquiries, selection, programming, erasing, and checking
- Response (one byte): Response to an inquiry
- Size (one byte): The amount of data for transmission excluding the command, amount of data, and checksum
- Checksum (one byte): The checksum is calculated so that the total of all values from the command byte to the SUM byte becomes H'00.
- Data (n bytes): Detailed data of a command or response
- Error response (one byte): Error response to a command
- Error code (one byte): Type of the error
- Address (four bytes): Address for programming
- Data (n bytes): Data to be programmed (the size is indicated in the response to the programming unit inquiry.)
- Size (four bytes): Four-byte response to a memory read

(3)   Inquiry and Selection States

The boot program returns information from the flash memory in response to the host's inquiry commands and sets the device code, clock mode, and bit rate in response to the host's selection command.

Table 22.13 lists the inquiry and selection commands.

Table 22.13 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'25	User MAT information inquiry	Inquiry regarding the a number of user MATs and the start and last addresses of each MAT
H'26	Block for erasing information Inquiry	Inquiry regarding the number of blocks and the start and last addresses of each block
H'27	Programming unit inquiry	Inquiry regarding the unit of program data
H'3F	New bit rate selection	Selection of new bit rate
H'40	Transition to programming/erasing state	Erasing of user MAT, and entry to programming/erasing state
H'4F	Boot program status inquiry	Inquiry into the operated status of the boot program

RENESAS

The selection commands, which are device selection (H'10), clock mode selection (H'11), and new bit rate selection (H'3F), should be sent from the host in that order. When two or more selection commands are sent at once, the last command will be valid.

All of these commands, except for the boot program status inquiry command (H'4F), will be valid until the boot program receives the programming/erasing transition (H'40). The host can choose the needed commands and make inquiries while the above commands are being transmitted. H'4F is valid even after the boot program has received H'40.

(a)   Supported Device Inquiry

The boot program will return the device codes of supported devices and the product code in response to the supported device inquiry.

Command | H'20 |

- Command, H'20, (one byte): Inquiry regarding supported devices

Response

H'30		Size	Number of devices	
Number of characters		Device code		Product name
...				
SUM				

- Response, H'30, (one byte): Response to the supported device inquiry
- Size (one byte): Number of bytes to be transmitted, excluding the command, size, and checksum, that is, the amount of data contributes by the number of devices, characters, device codes and product names
- Number of devices (one byte): The number of device types supported by the boot program
- Number of characters (one byte): The number of characters in the device codes and boot program's name
- Device code (four bytes): ASCII code of the supporting product
- Product name (n bytes): Type name of the boot program in ASCII-coded characters
- SUM (one byte): Checksum

  The checksum is calculated so that the total number of all values from the command byte to the SUM byte becomes H'00.

(b)   Device Selection

The boot program will set the supported device to the specified device code. The program will return the selected device code in response to the inquiry after this setting has been made.

Command | H'10 | Size | Device code | SUM |

- Command, H'10, (one byte): Device selection
- Size (one byte): Amount of device-code data
  This is fixed at 2
- Device code (four bytes): Device code (ASCII code) returned in response to the supported device inquiry
- SUM (one byte): Checksum

Response | H'06 |

- Response, H'06, (one byte): Response to the device selection command
  ACK will be returned when the device code matches.

Error response | H'90 | ERROR |

- Error response, H'90, (one byte): Error response to the device selection command
  ERROR   : (one byte): Error code
                 H'11: Sum check error
                 H'21: Device code error, that is, the device code does not match

(c)   Clock Mode Inquiry

The boot program will return the supported clock modes in response to the clock mode inquiry.

Command | H'21 |

- Command, H'21, (one byte): Inquiry regarding clock mode

Response | H'31 | Size | Number of modes | Mode | ··· | SUM |

- Response, H'31, (one byte): Response to the clock-mode inquiry
- Size (one byte): Amount of data that represents the number of modes and modes
- Number of clock modes (one byte): The number of supported clock modes
  H'00 indicates no clock mode or the device allows to read the clock mode.
- Mode (one byte): Values of the supported clock modes (i.e. H'01 means clock mode 1.)
- SUM (one byte): Checksum

RENESAS

(d)   Clock Mode Selection

The boot program will set the specified clock mode. The program will return the selected clock-mode information after this setting has been made.

The clock-mode selection command should be sent after the device-selection commands.

Command | H'11 | Size | Mode | SUM

- Command, H'11, (one byte): Selection of clock mode
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): A clock mode returned in reply to the supported clock mode inquiry.
- SUM (one byte): Checksum

Response | H'06

- Response, H'06, (one byte): Response to the clock mode selection command
  ACK will be returned when the clock mode matches.

Error Response | H'91 | ERROR

- Error response, H'91, (one byte): Error response to the clock mode selection command
- ERROR   : (one byte): Error code

  H'11: Checksum error

  H'22: Clock mode error, that is, the clock mode does not match.

Even if the clock mode numbers are H'00 and H'01 by a clock mode inquiry, the clock mode must be selected using these respective values.

(e)   Multiplication Ratio Inquiry

The boot program will return the supported multiplication and division ratios.

Command | H'22 |

- Command, H'22, (one byte): Inquiry regarding multiplication ratio

Response

H'32	Size	Number of types					
Number of multiplication ratios	Multiplica-tion ratio	...					
...							
SUM							

- Response, H'32, (one byte): Response to the multiplication ratio inquiry
- Size (one byte): The amount of data that represents the number of clock sources and multiplication ratios and the multiplication ratios
- Number of types (one 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 (one byte): The number of multiplication ratios for each type
  (e.g. the number of multiplication ratios to which the main clock can be set and the peripheral clock can be set.)
- Multiplication ratio (one byte)

  Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H'04.)

  Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

  The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are types.
- SUM (one byte): Checksum

RENESAS

(f)   Operating Clock Frequency Inquiry

The boot program will return the number of operating clock frequencies, and the maximum and minimum values.

Command | H'23 |

- Command, H'23, (one byte): Inquiry regarding operating clock frequencies

Response

H'33	Size	Number of operating clock frequencies
Minimum value of operating clock frequency	Maximum value of operating clock frequency	
...		
SUM		

- Response, H'33, (one byte): Response to operating clock frequency inquiry
- Size (one byte): The number of bytes that represents the minimum values, maximum values, and the number of frequencies.
- Number of operating clock frequencies (one byte): The number of supported operating clock frequency types
  (e.g. when there are two operating clock frequency types, which are the main and peripheral clocks, the number of types will be H'02.)
- Minimum value of operating clock frequency (two bytes): The minimum value of the multiplied or divided clock frequency.
  The minimum and maximum values of the operating clock frequency represent the values in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 17.00 MHz, it will be 2000, which is H'07D0.)
- Maximum value (two bytes): Maximum value among the multiplied or divided clock frequencies.
  There are as many pairs of minimum and maximum values as there are operating clock frequencies.
- SUM (one byte): Checksum

RENESAS

(g)   User MAT Information Inquiry

The boot program will return the number of user MATs and their addresses.

Command | H'25 |

- Command, H'25, (one byte): Inquiry regarding user MAT information

Response

H'35	Size	Number of areas	
Start address area		Last address area	
...			
SUM			

- Response, H'35, (one byte): Response to the user MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start address and area-last address
- Number of areas (one byte): The number of consecutive user MAT areas
  When the user MAT areas are consecutive, the number of areas is H'01.
- Area-start address (four bytes): Start address of the area
- Area-last address (four bytes): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(h)   Erased Block Information Inquiry

The boot program will return the number of erased blocks and their addresses.

Command | H'26 |

- Command, H'26, (two bytes): Inquiry regarding erased block information

Response

H'36	Size	Number of blocks	
Block start address		Block last address	
...			
SUM			

- Response, H'36, (one byte): Response to the number of erased blocks and addresses
- Size (two bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
- Number of blocks (one byte): The number of erased blocks
- Block start address (four bytes): Start address of a block

RENESAS

- Block last Address (four bytes): Last address of a block
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(i)   Programming Unit Inquiry

The boot program will return the programming unit used to program data.

Command | H'27 |

- Command, H'27, (one byte): Inquiry regarding programming unit

Response | H'37 | Size | Programming unit | SUM |

- Response, H'37, (one byte): Response to programming unit inquiry
- Size (one byte): The number of bytes that indicate the programming unit, which is fixed to 2
- Programming unit (two bytes): A unit for programming
  This is the unit for reception of programming.
- SUM (one byte): Checksum

(j)   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, (one byte): Selection of new bit rate
- Size (one byte): The number of bytes that represents the bit rate, input frequency, number of multiplication ratios, and multiplication ratio
- Bit rate (two bytes): New bit rate
  One hundredth of the value (e.g. when the value is 19200 bps, it will be 192, which is H'00C0.)
- Input frequency (two bytes): Frequency of the clock input to the boot program
  This is valid to the hundredths place and represents the value in MHz multiplied by 100. (E.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Number of multiplication ratios (one byte): The number of multiplication ratios to which the device can be set.

- Multiplication ratio 1 (one byte): The value of multiplication or division ratios for the main operating frequency

  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)

  Division ratio: The inverse of the division ratio, as a negative number (e.g. when the clock frequency is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

- Multiplication ratio 2 (one byte): The value of multiplication or division ratios for the peripheral frequency

  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)

  Division ratio: The inverse of the division ratio, as a negative number (E.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

- SUM (one byte): Checksum

Response | H'06 |

- Response, H'06, (one byte): Response to selection of a new bit rate
  When it is possible to set the bit rate, the response will be ACK.

Error Response | H'BF | ERROR |

- Error response, H'BF, (one byte): Error response to selection of new bit rate
- ERROR: (one byte): Error code

H'11:	Sum checking error
H'24:	Bit-rate selection error
	The rate is not available.
H'25:	Error in input frequency
	This input frequency is not within the specified range.
H'26:	Multiplication-ratio error
	The ratio does not match an available ratio.
H'27:	Operating frequency error
	The frequency is not within the specified range.

RENESAS

(4)   Receive Data Check

The methods for checking of receive data are listed below.

1. Input frequency

    The received value of the input frequency is checked to ensure that it is within the range of minimum to maximum frequencies which matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

2. Multiplication ratio

    The received value of the multiplication ratio or division ratio is checked to ensure that it matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

3. Operating frequency error

    Operating frequency is calculated from the received value of the input frequency and the multiplication or division ratio. The input frequency is input to the LSI and the LSI is operated at the operating frequency. The expression is given below.

    Operating frequency = Input frequency × Multiplication ratio, or
    Operating frequency = Input frequency ÷ Division ratio

    The calculated operating frequency should be checked to ensure that it is within the range of minimum to maximum frequencies which are available with the clock modes of the specified device. When it is out of this range, an operating frequency error is generated.

4. Bit rate

    To facilitate error checking, the value (n) of clock select (CKS) in the serial mode register (SMR), and the value (N) in the bit rate register (BRR), which are found from the peripheral operating clock frequency (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, (one byte): Confirmation of a new bit rate

Response | H'06 |

- Response, H'06, (one byte): Response to confirmation of a new bit rate

The sequence of new bit-rate selection is shown in figure 22.21.
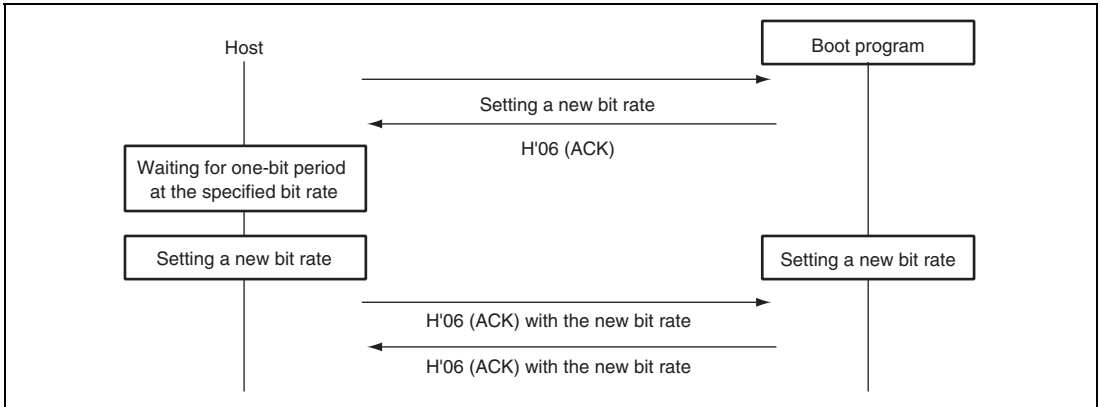
Figure 22.21   New Bit-Rate Selection Sequence

(5)   Transition to Programming/Erasing State

The boot program will transfer the erasing program and erase the user MATs. On completion of this erasure, ACK will be returned and the program will enter the programming/erasing state.

The host should select the device code, clock mode, and new bit rate with device selection, clock-mode selection, and new bit-rate selection commands, and then send the command for the transition to programming/erasing state. These procedures should be carried out before sending of the programming selection command or program data.

Command   H'40

- Command, H'40, (one byte): Transition to programming/erasing state

Response   H'06

- Response, H'06, (one byte): Response to transition to programming/erasing state
  The boot program will send ACK when the user MATs have been erased by the transferred erasing program.

Error Response   H'C0   H'51

- Error code, H'51, (one byte): Erasing error
  An error occurred and erasure was not completed.

RENESAS

(6)   Command Error

A command error will occur when a command is undefined, the order of commands is incorrect, or a command is unacceptable. Issuing a clock-mode selection command before a device selection or an inquiry command after the transition to programming/erasing state command, are examples.

Error Response | H'80 | H'xx |

- Error response, H'80, (one byte): Command error
- Command, H'xx, (one byte): Received command

(7)   Command Order

The order for commands in the inquiry selection state is shown below.

1. A supported device inquiry (H'20) should be made to inquire about the supported devices.
2. The device should be selected from among those described by the returned information and set with a device-selection (H'10) command.
3. A clock-mode inquiry (H'21) should be made to inquire about the supported clock modes.
4. The clock mode should be selected from among those described by the returned information and set with a clock mode-selection (H'11) command.
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 MAT should be made to inquire about the 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.

RENESAS

(8)   Programming/Erasing State

A programming selection command makes the boot program select the programming method, a 128-byte programming command makes it program the memory with data, and an erasing selection command and block erasing command make it erase the block. Table 22.14 lists the programming/erasing commands.

Table 22.14 Programming/Erasing Commands

Command	Command Name	Description
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'4B	User MAT sum check	Checks the checksum of the user MAT
H'4D	User MAT blank check	Checks the blank data of the user MAT
H'4F	Boot program status inquiry	Inquires into the boot program's status

- Programming

  Programming is executed by the programming selection and 128-byte programming commands.

  Firstly, the host should send the programming selection command

  After issuing the programming selection command, the host should send the 128-byte programming command. The 128-byte programming command that follows the selection command represents the data programmed according to the method specified by the selection command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H'FFFFFFFF as the address will stop the programming. On completion of programming, the boot program will wait for selection of programming or erasing.

  Where the sequence of programming operations that is executed includes programming with another method or of another MAT, the procedure must be repeated from the programming selection command.

  The sequence for the programming selection and 128-byte programming commands is shown in figure 22.22.
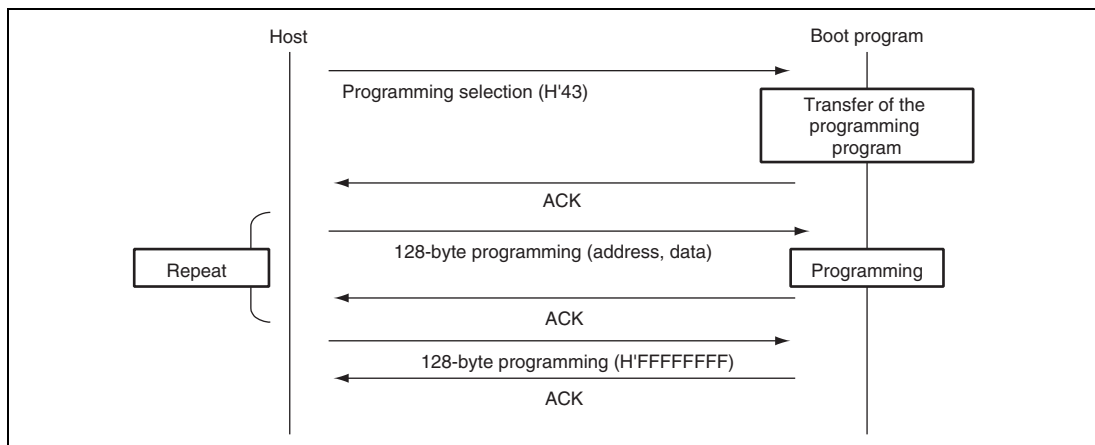
Figure 22.22   Programming Sequence

- Erasure

  Erasure is executed by the erasure selection and block erasure commands.

  Firstly, erasure is selected by the erasure selection command and the boot program then erases the specified block. The command should be repeatedly executed if two or more blocks are to be erased. Sending a block erasure command from the host with the block number H'FF will stop the erasure operating. On completion of erasing, the boot program will wait for selection of programming or erasing.

  The sequence for the erasure selection and block erasure commands is shown in figure 22.23.
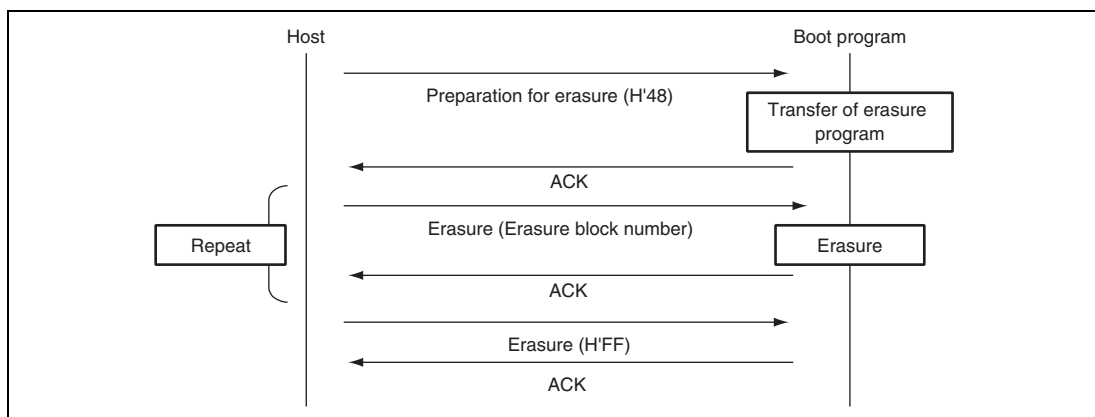
Figure 22.23   Erasure Sequence

(a)   User MAT Programming Selection

The boot program will transfer a program for user MAT programming selection. The data is
programmed to the user MATs by the transferred program for programming.

Command | H'43

- Command, H'43, (one byte): User-program programming selection

Response | H'06

- Response, H'06, (one byte): Response to user-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response | H'C3 | ERROR

- Error response : H'C3 (1 byte): Error response to user-program 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 MATs in response to 128-byte programming.

Command | H'50 | Address
Data	...					
...						
SUM						

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): Start address for programming
  Multiple of the size specified in response to the programming unit inquiry
  (i.e. H'00, H'01, H'00, H'00 : H'01000000)
- Program data (128 bytes): Data to be programmed
  The size is specified in the response to the programming unit inquiry.
- SUM (one byte): Checksum

Response | H'06

- Response, H'06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

RENESAS

Error Response | H'D0 | ERROR |

- Error response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

>   H'11: Checksum Error
>   H'2A: Address error
>   The address is not in the range of the specified MAT.
>   H'53: Programming error
>   A programming error has occurred and programming cannot be continued.

The specified address should match the unit for programming of data. For example, when the programming is in 128-byte units, the lower eight bits of the address should be H'00 or H'80. When there are less than 128 bytes of data to be programmed, the host should fill the rest with H'FF.

Sending the 128-byte programming command with the address of H'FFFFFFFF will stop the programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.

Command | H'50 | Address | SUM |

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): End code is H'FF, H'FF, H'FF, H'FF.
- SUM (one byte): Checksum

Response | H'06 |

- Response, H'06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response | H'D0 | ERROR |

- Error Response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

>   H'11: Checksum error
>   H'53: Programming error
>   An error has occurred in programming and programming cannot be continued.

(c)　Erasure Selection

The boot program will transfer the erasure program. User MAT data is erased by the transferred erasure program.

Command | H'48

• Command, H'48, (one byte): Erasure selection

Response | H'06

• Response, H'06, (one byte): Response for erasure selection
  After the erasure program has been transferred, the boot program will return ACK.

Error Response | H'C8 | ERROR

• ERROR: (one byte): Error code
  H'54:　Selection processing error (transfer error occurs and processing is not completed)

(d)　Block Erasure

The boot program will erase the contents of the specified block.

Command | H'58 | Size | Block number | SUM

• Command, H'58, (one byte): Erasure
• Size (one byte): The number of bytes that represents the erase block number
  This is fixed to 1.
• Block number (one byte): Number of the block to be erased
• SUM (one byte): Checksum

Response | H'06

• Response, H'06, (one byte): Response to Erasure
  After erasure has been completed, the boot program will return ACK.

Error Response | H'D8 | ERROR

• Error Response, H'D8, (one byte): Response to Erasure
• ERROR (one byte): Error code

　　　　　　　H'11:　　Sum check error
　　　　　　　H'29:　　Block number error
　　　　　　　　　　　Block number is incorrect.
　　　　　　　H'51:　　Erasure error
　　　　　　　　　　　An error has occurred during erasure.

RENESAS

On receiving block number H'FF, the boot program will stop erasure and wait for a selection command.

Command | H'58 | Size | Block number | SUM

- Command, H'58, (one byte): Erasure
- Size, (one byte): The number of bytes that represents the block number
  This is fixed to 1.
- Block number (one byte): H'FF
  Stop code for erasure
- SUM (one byte): Checksum

Response | H'06

- Response, H'06, (one byte): Response to end of erasure (ACK)
  When erasure is to be performed after the block number H'FF has been sent, the procedure should be executed from the erasure selection command.

(e)   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'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 that has been read
- Data (n bytes): Data for the read size from the read address
- SUM (1 byte): Checksum

RENESAS

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.

(f)   User-Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user program.

Command | H'4B

- Command, H'4B, (one byte): Sum check for user program

Response | H'5B | Size | Checksum of user program | SUM

- Response, H'5B, (one byte): Response to the sum check of the user program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(g)   User MAT Blank Check

The boot program will check whether or not all user MATs are blank and return the result.

Command | H'4D

- Command, H'4D, (one byte): Blank check for user MATs

Response | H'06

- Response, H'06, (one byte): Response to the blank check for user MATs
  If the contents of all user MATs are blank (H'FF), the boot program will return ACK.

Error Response | H'CD | H'52

- Error Response, H'CD, (one byte): Error response to the blank check of user MATs.
- Error code, H'52, (one byte): Erasure has not been completed.

(h)   Boot Program State Inquiry

The boot program will return indications of its present state and error condition. This inquiry can be made in the inquiry/selection state or the programming/erasing state.

Command | H'4F |

- Command, H'4F, (one byte):   Inquiry regarding boot program's state

Response | H'5F | Size | Status | ERROR | SUM |

- Response, H'5F, (one byte): Response to boot program state inquiry
- Size (one byte): The number of bytes. This is fixed to 2.
- Status (one byte): State of the boot program
- ERROR (one byte): Error status

> ERROR = 0 indicates normal operation.
> ERROR = 1 indicates error has occurred.

- SUM (one byte): Sum check

Table 22.15 Status Codes

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	Program data receive wait
H'5F	Erase block specification wait (erasure is completed)

Table 22.16 Error Codes

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

22.12   Usage Notes

1. The initial state of the product at its shipment is in the erased state. For the product whose revision of erasing is undefined, we recommend to execute automatic erasure for checking the initial state (erased state) and compensating.

2. For the PROM programmer suitable for programmer mode in this LSI and its program version, refer to the instruction manual of the socket adapter.

3. If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.

4. Use a PROM programmer that supports the device with 512-Kbyte on-chip flash memory and 5.0-V programming voltage. Use only the specified socket adapter.

5. Do not remove the chip from the PROM programmer nor input a reset signal during programming/erasing in which a high voltage is applied to the flash memory. Doing so may damage the flash memory permanently. If a reset is input accidentally, the reset must be released after the reset input period of at least 100μs.

6. The flash memory is not accessible until FKEY is cleared after programming/erasing starts. If the operating mode is changed and this LSI is restarted by a reset immediately after programming/erasing has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100μs. Transition to the reset state during programming/erasing is inhibited. If a reset is input accidentally, the reset must be released after the reset input period of at least 100μs.

7. At powering on or off the Vcc power supply, fix the $\overline{RES}$ pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.

8. In on-board programming mode or programmer mode, programming of the 128-byte programming-unit block must be performed only once. Perform programming in the state where the programming-unit block is fully erased.

9. When the chip is to be reprogrammed with the programmer after execution of programming or erasure in on-board programming mode, it is recommended that automatic programming is performed after execution of automatic erasure.

10. To program the flash memory, the program data and program must be allocated to addresses which are higher than those of the external interrupt vector table and H'FF must be written to all the system reserved areas in the exception handling vector table.

11. The programming program that includes the initialization routine and the erasing program that includes the initialization routine are each 4 kbytes or less. Accordingly, when the CPU clock frequency is 40 MHz, the download for each program takes approximately 100 μs at the maximum.

12. A programming/erasing program for the flash memory used in a conventional F-ZTAT H8 or H8S microcomputer which does not support download of the on-chip program by setting the SCO bit in FCCS to 1 cannot run in this LSI. Be sure to download the on-chip program to execute programming/erasing of the flash memory in this F-ZTAT H8SX microcomputer.

13. Unlike a conventional F-ZTAT H8 or H8S microcomputer, measures against a program crash are not taken by WDT while programming/erasing and downloading a programming/erasing program. When needed, measures should be taken by user. A periodic interrupt generated by the WDT can be used as the measures, as an example. In this case, the interrupt generation period should take into consideration time to program/erase the flash memory.

14. When downloading the programming/erasing program, do not clear the SCO bit in FCCS to 0 after immediately setting it to 1. Otherwise, download cannot be performed normally. Immediately after executing the instruction to set the SCO bit to 1, dummy read of the FCCS must be executed twice.

15. The contents of some registers are not saved in a programming/programming end/erasing program. When needed, save registers in the procedure program.

16. The intervals for executing the user branch processing differ in programming and erasing. They also differ depending on at which processing phase it is executed. Table 22.17 lists the maximum and minimum intervals for initiating the user branch processing when the CPU clock frequency is 40 MHz.

Table 22.17 Initiation Intervals of User Branch Processing

Processing Name	Maximum Interval	Minimum Interval
Programming	Approximately 1 ms	Approximately 19 μs
Erasing	Approximately 5 ms	Approximately 19 μs

17. While an instruction in on-chip RAM is being executed, the DMAC can write to the SCO bit in FCCS that is used for a download request. Make sure that this register is not accidentally written to, otherwise an on-chip program may be downloaded and the contents of RAM are damaged, which causes the CPU to go out of control.

18. States in Which NMI Interrupt Is Ignored

In the following modes or period, the NMI interrupt requests are ignored; they are not executed and the interrupt sources are not retained.

— Boot mode
— Programmer mode

(Approximately a period of A (s) if operation is done at an input clock frequency of B (Hz) after the reset signal is released)

$$A = \frac{1}{B \times 2} \times 1200$$

The following is an example when an input clock frequency is 5 MHz.

$$\frac{1}{5 \times 10^6 \times 2} \times 1200 = 120 \times 10^{-6} = 120 \ \mu s$$

19. Notes on NMI Interrupt during Programming/Erasing

Do not execute interrupt processing during programming/erasing. Doing so causes the error protection state to be entered, aborts programming/erasing, and prevents flash memory from being successfully programmed/erased.

When executing an NMI interrupt during programming/erasing for error processing, the following should be noted.

— The NMI interrupt is an only interrupt that can be used during programming/erasing.
— When flash memory is being programmed or erased, the user MAT cannot be accessed. Prepare the interrupt vector table and interrupt processing routine in on-chip RAM by setting VBR. Make sure the flash memory being programmed or erased is not accessed by the interrupt processing routine. If flash memory is read, the read values are not guaranteed.
— Since generation of NMI requests (or other interrupt requests) during programming/erasing causes the error protection state to be entered, programming or erasing is aborted. For details on the error protection, see section 22.8.3, Error Protection.

20. Notes on Returning from the Error Protection State

Allow a reset period of 100 μs or longer before releasing the reset signal to return from the error protection state. A transition to the error protection state aborts programming or erasing, and programming or erasing is not performed properly. Note the following when returning from the error protection state to each mode.

–– Boot mode

The boot program programmed in the LSI is initiated when the reset signal is released. All of the user MAT and user boot MAT are automatically erased before programming.

–– User program mode

The program programmed in the user MAT is initiated when the reset signal is released. However, the program may not be initiated properly from the user MAT after releasing the reset signal. Erase all of the user MAT in boot mode etc. before programming. Then, set to user program mode before releasing the reset signal again.

RENESAS

Section 23   Clock Pulse Generator

This LSI has an on-chip clock pulse generator (CPG) that generates the system clock (Iϕ), peripheral module clock (Pϕ), external bus clock (Bϕ), and sub clock (SUBϕ).

The clock pulse generator consists of an oscillator, PLL (Phase Locked Loop) circuit, main clock divider, sub-clock waveform shaping circuit, sub-clock divider, and selector circuit. Figure 23.1 shows a block diagram of the clock pulse generator.

Clock frequencies can be changed by the PLL circuit, main clock divider, and sub clock divider in the CPG. Changing the system clock control register (SCKCR) and sub-clock control register (SUBCKCR) settings by software can change the clock frequencies.

This LSI supports four types of clocks: a system clock provided to the CPU and bus masters, a peripheral module clock provided to the peripheral modules, an external bus clock provided to the external bus, and a sub clock provided to the WAT. These clocks can be set independently. Note, however, that the frequencies of the peripheral clock and external bus clock are lower than that of the system clock.
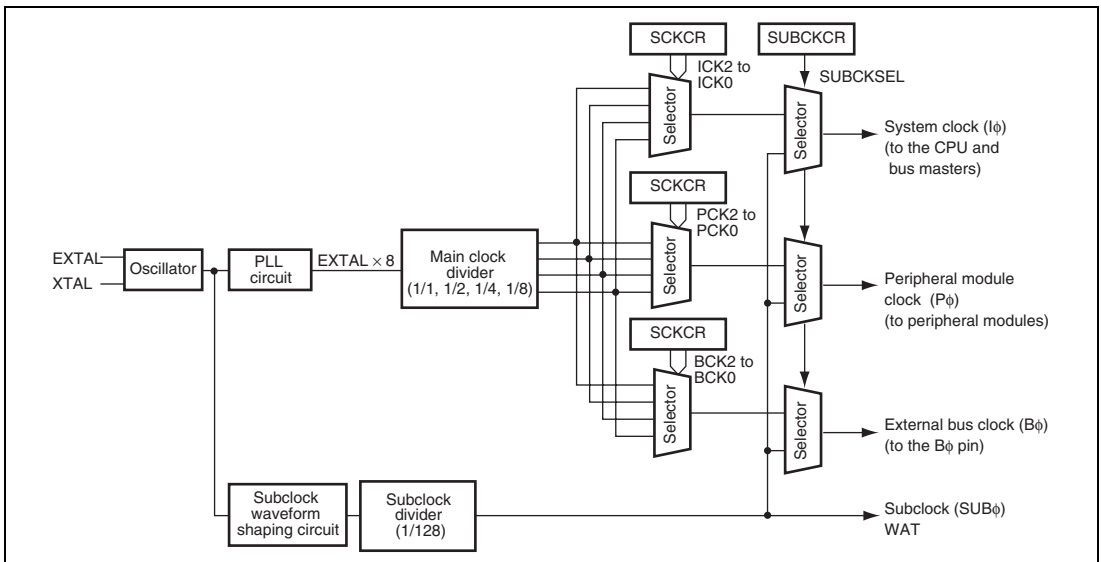
Figure 23.1   Block Diagram of Clock Pulse Generator

23.1   Register Description

The clock pulse generator has the following registers.

- System clock control register (SCKCR)
- Sub-clock control register (SUBCKCR)

23.1.1   System Clock Control Register (SCKCR)

SCKCR controls φ clock output and frequencies of the system, peripheral module, and external clocks, and selects the φ clock to be output.

Bit	15	14	13	12	11	10	9	8
Bit Name	PSTOP1	—	POSEL1	—	—	ICK2	ICK1	ICK0
Initial Value	0	0	0	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0
Initial Value	0	0	1	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	PSTOP1	0	R/W	φ Clock Output Enable Controls φ output on PA7. • Normal operation 0: φ output 1: Fixed high • Software standby mode X: Fixed high • Hardware standby mode X: Hi-Z
14	—	0	R/W	Reserved This bit can be read or written, but the write value should always be 0.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
13	POSEL1	0	R/W	ϕ Output Select 1
				Controls the ϕ output on PA7.
				0: External clock (Bϕ)
				1: Setting prohibited
12, 11	—	All 0	R/W	Reserved
				These bits can be read or written, but the write value should always be 0.
10	ICK2	0	R/W	System Clock (Iϕ) Select
9	ICK1	1	R/W	These bits select the frequency of the system clock
8	ICK0	0	R/W	provided to the CPU and DMAC. The ratio to the input clock is as follows:
				000: ×8
				001: ×4
				010: ×2
				011: ×1
				1XX: Setting prohibited
				The frequency of the peripheral module clock and the external bus clock changes to the same frequency as the system clock if the frequency of the system clock is lower than that of the peripheral module clock and the external bus clock.
7	—	0	R/W	Reserved
				This bit can be read or written, but the write value should always be 0.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
6	PCK2	0	R/W	Peripheral Module Clock (Pφ) Select
5	PCK1	1	R/W	These bits select the frequency of the peripheral module clock. The ratio to the input clock is as follows:
4	PCK0	0	R/W	
				000: ×8
				001: ×4
				010: ×2
				011: ×1
				1XX: Setting prohibited
				The frequency of the peripheral module clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the peripheral module clock higher than that of the system clock, the clocks will have the same frequency in reality.
3	—	0	R/W	Reserved
				This bit can be read or written, but the write value should always be 0.
2	BCK2	0	R/W	External Clock (Bφ) Select
1	BCK1	1	R/W	These bits select the frequency of the external clock. The ratio to the input clock is as follows:
0	BCK0	0	R/W	
				000: ×8
				001: ×4
				010: ×2
				011: ×1
				1XX: Setting prohibited
				The frequency of the external clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the external clock higher than that of the system clock, the clocks will have the same frequency in reality.

Note: X: Don't care

RENESAS

23.1.2   Sub-Clock Control Register (SUBCKCR)

SUBCKCR controls the sub clock.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	XTALSTP	PLLSTP	WKCKSEL	SUBCKSEL
Initial Value	0	0	0	0	1	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	R/W	Reserved
				These bits can be read or written, but the write value should always be 0.
3	XTALSTP	1	R/W	XTAL Stop
				Controls the oscillator operation in software standby mode.
				0: The oscillator operates in software standby mode
				1: The oscillator stops operating in software standby mode
2	PLLSTP	1	R/W	PLL Stop
				Controls the PLL circuit operation during sub-clock operation.
				0: The PLL circuit operates during sub-clock operation
				1: The PLL circuit stops operating during sub-clock operation

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
1	WKCKSEL	0	R/W	Wakeup Clock Select
				Selects the frequency for the system clock (Iϕ), peripheral module clock (Pϕ), and external bus clock (Bϕ) when the software standby mode is cancelled by an interrupt request. This bit setting is only valid when software standby mode entered from sub-clock operation is cancelled.
				0: Iϕ, Pϕ, and Bϕ are derived from the main clock selected by SCKCR
				1: Iϕ, Pϕ, and Bϕ are derived from the sub clock
0	SUBCKSEL	0	R/W	Sub clock Select
				Selects the frequency for the system clock (Iϕ), peripheral module clock (Pϕ), and external bus clock (Bϕ). When this bit is set to 1, Iϕ, Pϕ, and Bϕ are always derived from the sub clock regardless of the SCKCR setting.
				0: Iϕ, Pϕ, and Bϕ are derived from the main clock selected by SCKCR.
				1: Iϕ, Pϕ, and Bϕ are derived from the sub clock. [Clearing conditions]
				• When 0 is written
				• When a transition from sub-clock operation to software standby mode occurs while the WKCKSEL bit is cleared to 0

RENESAS

23.2   Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

23.2.1   Connecting Crystal Resonator

A crystal resonator can be connected as shown in the example in figure 23.2. Select the damping resistance R_d according to table 23.1. An AT-cut parallel-resonance type should be used.

When the clock is provided by connecting a crystal resonator, a crystal resonator having a frequency of 4 to 9 MHz should be connected.
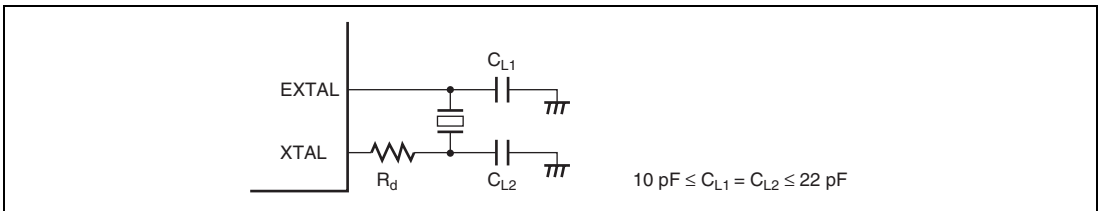
Figure 23.2   Connection of Crystal Resonator (Example)

Table 23.1   Damping Resistance Value

Frequency (MHz)	4	6	8	9
R_d (Ω)	500	300	200	100

Figure 23.3 shows an equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 23.2.
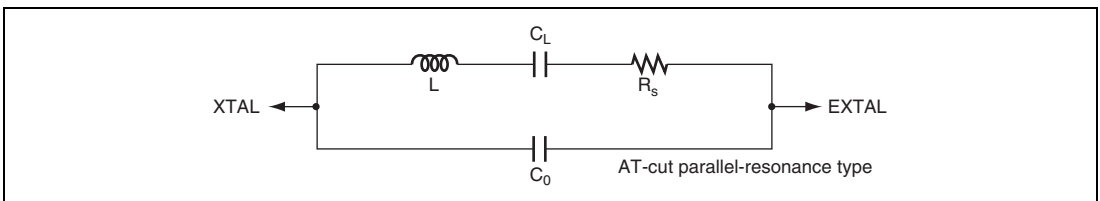
Figure 23.3   Crystal Resonator Equivalent Circuit

Table 23.2   Crystal Resonator Characteristics

Frequency (MHz)	4	6	8	9
R_s Max. (Ω)	120	100	80	80
C_0 Max. (pF)	7	7	7	7

23.2.2   External Clock Input

An external clock signal can be input as shown in the examples in figure 23.4. When the counter clock is input to the XTAL pin, make sure that the external clock is held high in standby mode.

EXTAL

XTAL

External clock input

Counter clock input on XTAL pin
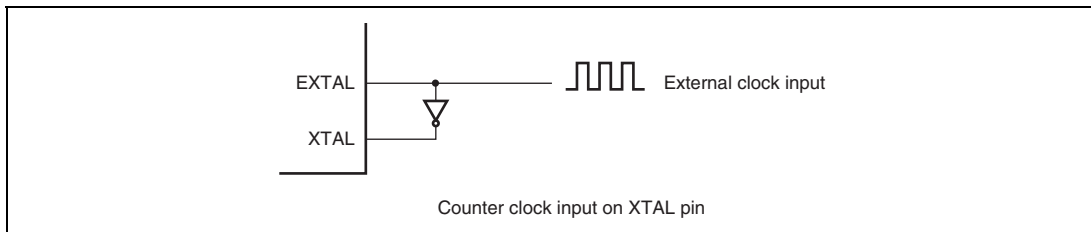
Figure 23.4   External Clock Input (Examples)

For the input conditions of the external clock, refer to table 26.4, Clock Timing, in section 26.3.1, Clock Timing. The input external clock should be from 4 to 9 MHz.

RENESAS

23.3   PLL Circuit

The PLL circuit has the function of multiplying the frequency of the clock from the oscillator by a factor of 8. The frequency multiplication factor is fixed. In this case, the phase of the rising edge of the internal clock is controlled to be matched with the phase of the EXTAL rising edge.

23.4   Main Clock Divider

The main clock divider divides the PLL clock to generate 1/2, 1/4, and 1/8 clocks. After bits ICK2 to ICK0, PCK 2 to PCK0, and BCK2 to BCK0 are modified, this LSI operates at the modified frequency.

23.5   Sub-Clock Waveform Shaping Circuit

To remove noises on the clock input from the EXTAL pin, the sub-clock waveform shaping circuit performs sampling at the rate of the peripheral module clock (Pϕ) divided by 32. The sub-clock waveform shaping circuit does not perform sampling during sub-clock operation (SUBCKSEL = 1) or software standby mode.

23.6   Sub-Clock Divider

The sub-clock divider generates a sub clock by dividing the clock provided from the sub-clock waveform shaping circuit by 128. When the SUBCKSEL bit is set to 1, operation of this LSI is based on the sub-clock.

23.7   Usage Notes

23.7.1   Notes on Clock Pulse Generator

1. The following points should be noted since the frequency of ϕ (Iϕ: system clock, Pϕ: peripheral module clock, and Bϕ: external bus clock) supplied to each module changes according to the setting of SCKCR.

   Select a clock division ratio that is within the operation guaranteed range of clock cycle time t_{cyc} shown in the AC timing of electrical characteristics.

   When the RCAN-ET is in use, Iϕmin = 8 MHz, Pϕmin = 8 MHz, Bϕmin = 8 MHz, Iϕmax = 40 MHz, Pϕmax = 20 MHz, and Bϕmax = 20 MHz; the following settings are not permitted:

   Iϕ < 8MHz, Iϕ > 40 MHz, Pϕ < 16 MHz, Pϕ > 20 MHz, Bϕ < 8 MHz, and Bϕ > 20 MHz.

   When the RCAN_ET is not in use, Iϕmin = 8 MHz, Pϕmin = 8 MHz, Bϕmin = 8 MHz, Iϕmax = 40 MHz, Pϕmax = 20 MHz, and Bϕmax = 20 MHz; the following settings are not permitted:

   Iϕ < 8MHz, Iϕ > 40 MHz, Pϕ < 8 MHz, Pϕ > 20 MHz, Bϕ < 8 MHz, and Bϕ > 20 MHz.

2. All the on-chip peripheral modules (except for the DMAC) operate on the Pϕ. Therefore, note that the time processing of modules such as a timer and SCI differs before and after changing the clock division ratio.

   In addition, wait time for clearing software standby mode differs by changing the clock division ratio. For details, see section 24.6.3, Setting Oscillation Settling Time after Clearing Software Standby Mode.

3. The relationship among the system clock, peripheral module clock, and external bus clock is Iϕ ≥ Pϕ and Iϕ ≥ Bϕ. In addition, the system clock setting has priority. Accordingly, Pϕ and Bϕ may have the frequency set by bits ICK2 to ICK0 regardless of the settings of bits PCK2 to PCK0 and BCK2 to BCK0.

4. Note that the frequency of ϕ will be changed during the external bus cycle if SCKCR is set by the write data buffer function during external bus cycle execution.

5. Figure 23.5 shows the clock modification timing. After a value is written to SCKCR, this LSI waits for the current bus cycle to complete. After the current bus cycle completes, each clock frequency will be modified within one cycle (worst case) of the external input clock ϕ.

RENESAS
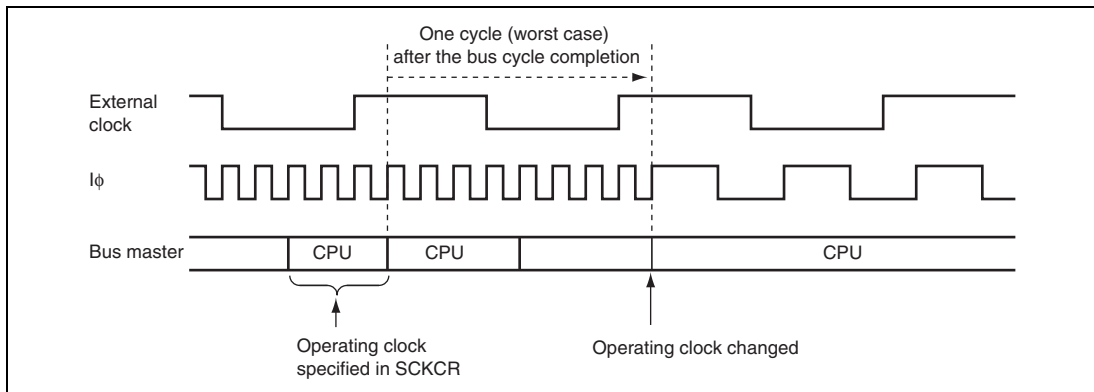
Figure 23.5   Clock Modification Timing

23.7.2   Notes on Resonator

Since various characteristics related to the resonator are closely linked to the user's board design, thorough evaluation is necessary on the user's part, using the resonator connection examples shown in this section as a reference. As the parameters for the resonator will depend on the floating capacitance of the resonator and the mounting circuit, the parameters should be determined in consultation with the resonator manufacturer. The design must ensure that a voltage exceeding the maximum rating is not applied to the resonator pin.

23.7.3   Notes on Board Design

When using the crystal resonator, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins. Other signal lines should be routed away from the oscillation circuit as shown in figure 23.6 to prevent induction from interfering with correct oscillation.
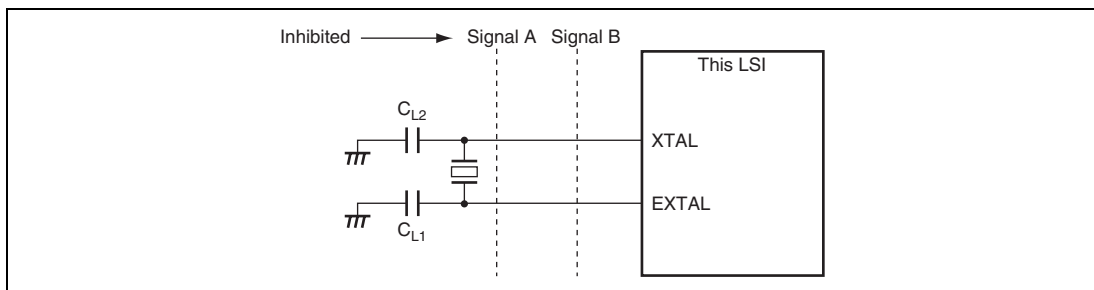
Figure 23.6   Note on Board Design for Oscillation Circuit

Figure 23.7 shows a connection example of bypass capacitor. Please be sure to insert bypass capacitor (CB) close to the Vcc and Vss pins and its capacitance meets the characteristics of the user system board.

Note: *   Use 0.1-μF laminated capacitors for C1 and CB (the value is a recommended value).
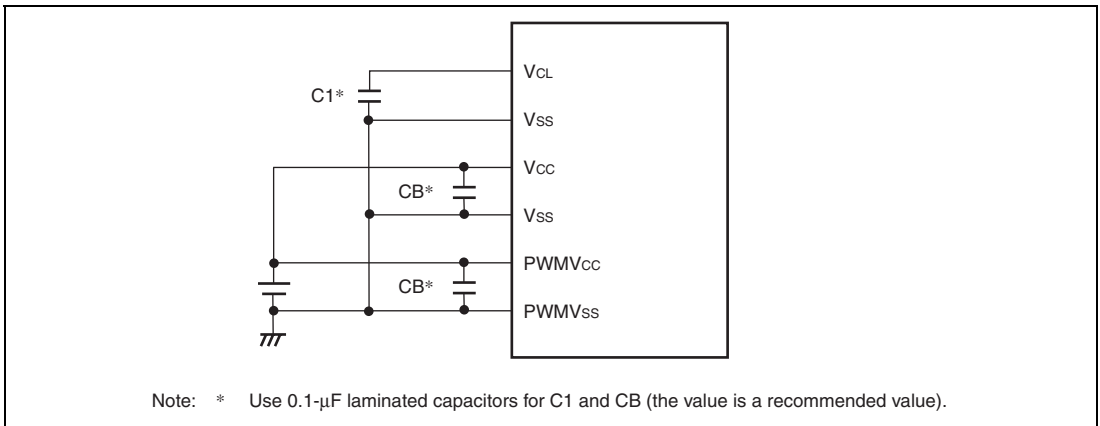
Figure 23.7   Connection Example of Bypass Capacitor

23.7.4   Notes on Input Clock Frequency

The frequency of the input clock is multiplied in the PLL circuit by a factor of 8. To reduce noises, a lower frequency ranging of 4 to 9 MHz is recommended.

RENESAS

Section 24   Power-Down Modes

This LSI has power consumption reduction functions, such as multi-clock function, module stop function, and transition function to power-down mode.

24.1   Features

- Multi-clock function of the main clock

  The frequency division ratio is settable independently for the system clock, peripheral module clock, and external bus clock.

- Sub-clock function

  The system clock, peripheral module clock, and external bus clock frequencies are settable to the frequency generated by dividing the main clock by 128.

- Module stop function

  The functions for each peripheral modules can be stopped to make a transition to a power-down mode.

- Transition function to power-down mode

  Transition to a power-down mode is possible to stop the CPU, peripheral modules, and oscillator.

- Four power-down modes

  Sleep mode

  All-module-clock-stop mode

  Sub-clock mode

  Software standby mode

  Hardware standby mode

Table 24.1 shows conditions for making a transition to a power-down mode, states of the CPU and peripheral modules, and clearing method for each mode. After reset, this LSI operates in normal program execution state and the modules other than the DMAC enter module stop mode.

Table 24.1   Operating States

Operating State	Sleep Mode	All-Module-Clock-Stop Mode	Sub-Clock Mode	Software Standby Mode	Hardware Standby Mode
Transition condition	Control register + instruction	Control register + instruction	Control register	Control register + instruction	Pin input
Cancellation method	Interrupt	Interrupt*[1]	Control register	Interrupt*[2]	Pin input
Oscillator	Operating	Operating	Operating	Halted/Operating*[3]	Halted
Sub-clock	Operating	Operating	Operating	Halted/Operating*[3]	Halted
CPU	Halted (retained)	Halted (retained)	Operating on sub-clock	Halted (retained)	Halted
Watchdog timer	Operating	Operating	Operating on sub-clock	Halted (retained)	Halted
WAT	Operating	Operating	Operating on sub-clock	Halted/Operating*[3]	Halted
Other peripheral modules	Operating	Halted*[4]	Halted*[6]	Halted*[4]	Halted*[5]
I/O port	Operating	Retained	Operating	Retained	Hi-Z

Notes: "Halted (retained)" in the table means that the internal register values are retained and internal operations are suspended.

1. External interrupts and some internal interrupts (watchdog timer)
2. External and WAT interrupts
3. When the XTALSTP bit in SUBSCCR is cleared to 0, the module can operate even in software standby mode.
4. Some SCI registers, the motor control PWM, 16-bit PWM, RCAN-ET, SSU*, and SDG are reset. Other peripheral modules retain their states.
5. All peripheral modules enter the reset state.
6. Stop a peripheral module with the corresponding module stop bit. Then, the corresponding peripheral module is stopped (reset).
* SSU: Synchronous Serial communication Unit
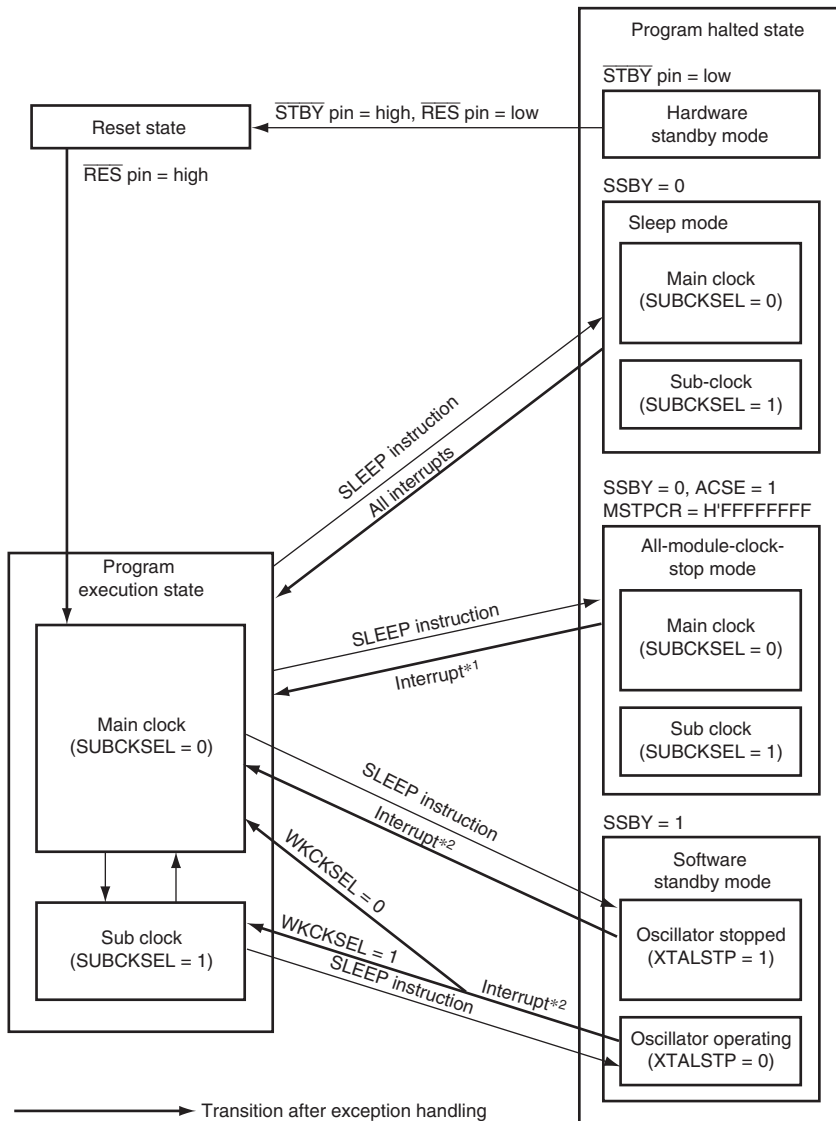
RENESAS

Figure 24.1   Mode Transitions

24.2   Register Descriptions

The registers related to the power-down modes are shown below. For details on the system clock control register (SCKCR), refer to section 23.1.1, System Clock Control Register (SCKCR).

- Standby control register (SBYCR)
- Module stop control register A (MSTPCRA)
- Module stop control register B (MSTPCRB)
- Module stop control register C (MSTPCRC)

24.2.1   Standby Control Register (SBYCR)

SBYCR controls software standby mode.

Bit	15	14	13	12	11	10	9	8
Bit Name	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0
Initial Value	0	1	0	0	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

Bit	Bit Name	Initial Value	R/W	Description
15	SSBY	0	R/W	Software Standby
				Specifies the transition mode after executing the SLEEP instruction
				0: Shifts to sleep mode after the SLEEP instruction is executed
				1: Shifts to software standby mode after the SLEEP instruction is executed
				This bit does not change when clearing the software standby mode by using external interrupts and shifting to normal operation. For clearing, write 0 to this bit. When the WDT is used as the watchdog timer, the setting of this bit is disabled. In this case, a transition is always made to sleep mode or all-module-clock-stop mode after the SLEEP instruction is executed.

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
14	OPE	1	R/W	Output Port Enable
				Specifies whether the output of the address bus and bus control signals ($\overline{\text{AS}}$, $\overline{\text{RD}}$, $\overline{\text{LHWR}}$, and $\overline{\text{LLWR}}$) is retained or set to the high-impedance state in software standby mode.
				0: In software standby mode, address bus and bus control signals are high-impedance
				1: In software standby mode, address bus and bus control signals retain output state
13	—	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
12	STS4	0	R/W	Standby Timer Select 4 to 0
11	STS3	1	R/W	These bits select the time the MCU waits for the clock to
10	STS2	1	R/W	settle when software standby mode is cleared by an
9	STS1	1	R/W	external interrupt. With a crystal resonator, refer to table
8	STS0	1	R/W	24.2 and make a selection according to the operating
				frequency so that the standby time is at least equal to the oscillation settling time. With an external clock, a PLL circuit settling time is necessary. Refer to table 24.2 to set the standby time.

The standby time is counted with the P ϕ frequency when shifted from the sub clock to the main clock in the program execution state or when shifted from software standby mode to main clock. This also applies when the multi-clock function is enabled. On the other hand, the standby time is counted with the sub clock when shifted from software standby mode (in sub-clock operation) to the sub clock in the program execution state. Careful consideration is required since the reference frequency depends on the transition state.

00000: Reserved

00001: Reserved

00010: Standby time = 8 states

00011: Standby time = 16 states

00100: Standby time = 32 states

00101: Standby time = 64 states

00110: Standby time = 512 states

00111: Standby time = 1024 states

01000: Standby time = 2048 states

01001: Standby time = 4096 states

01010: Standby time = 16384 states

01011: Standby time = 32768 states

01100: Standby time = 65536 states

01101: Standby time = 131072 states

01110: Standby time = 262144 states

01111: Standby time = 524288 states

1XXXX: Reserved

RENESAS

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R/W	Reserved
				These bits are always read as 0. The write value should always be 0.

24.2.2   Module Stop Control Registers A and B (MSTPCRA and MSTPCRB)

MSTPCRA and MSTPCRB control module stop mode. Setting a bit to 1 makes the corresponding module enter module stop mode, while clearing the bit to 0 clears module stop mode.

- MSTPCRA

Bit	15	14	13	12	11	10	9	8
Bit Name	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8
Initial Value	0	0	0	0	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MSTPCRB

Bit	15	14	13	12	11	10	9	8
Bit Name	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MSTPCRA

Bit	Bit Name	Initial Value	R/W	Module
15	ACSE	0	R/W	All-Module-Clock-Stop Mode Enable
				Enables/disables all-module-clock-stop mode for reducing current consumption by stopping the bus controller and I/O ports operations when the CPU executes the SLEEP instruction after module stop mode has been set for all the on-chip peripheral modules controlled by MSTPCR.
				0: All-module-clock-stop mode disabled
				1: All-module-clock-stop mode enabled
14	MSTPA14	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0 except when all-module-clock-stop mode is set.
13	MSTPA13	0	R/W	DMA controller (DMAC)
12	MSTPA12	0	R/W	Reserved
				This bit is always read as 0. The write value should always be 0 except when all-module-clock-stop mode is set.
11	MSTPA11	1	R/W	Reserved
10	MSTPA10	1	R/W	These bits are always read as 1. The write value should always be 1.
9	MSTPA9	1	R/W	
8	MSTPA8	1	R/W	
7	MSTPA7	1	R/W	
6	MSTPA6	1	R/W	
5	MSTPA5	1	R/W	
4	MSTPA4	1	R/W	A/D converter (unit 1)
3	MSTPA3	1	R/W	A/D converter (unit 0)
2	MSTPA2	1	R/W	Reserved
1	MSTPA1	1	R/W	These bits are always read as 1. The write value should always be 1.
0	MSTPA0	1	R/W	16-bit timer pulse unit (TPU channels 5 to 0)

RENESAS

- MSTPCRB

Bit	Bit Name	Initial Value	R/W	Module
15	MSTPB15	1	R/W	Reserved
14	MSTPB14	1	R/W	These bits are always read as 1. The write value should always be 1.
13	MSTPB13	1	R/W	Serial communication interface_5 (SCI_5)
12	MSTPB12	1	R/W	Serial communication interface_4 (SCI_4)
11	MSTPB11	1	R/W	Reserved
				This bit is always read as 1. The write value should always be 1.
10	MSTPB10	1	R/W	Serial communication interface_2 (SCI_2)
9	MSTPB9	1	R/W	Reserved
				This bit is always read as 1. The write value should always be 1.
8	MSTPB8	1	R/W	Serial communication interface_0 (SCI_0)
7	MSTPB7	1	R/W	I^{2}C bus interface_1 (IIC_1)
6	MSTPB6	1	R/W	I^{2}C bus interface_0 (IIC_0)
5	MSTPB5	1	R/W	Reserved
4	MSTPB4	1	R/W	These bits are always read as 1. The write value should always be 1.
3	MSTPB3	1	R/W	
2	MSTPB2	1	R/W	
1	MSTPB1	1	R/W	
0	MSTPB0	1	R/W	

24.2.3   Module Stop Control Register C (MSTPCRC)

The MSTPC15 to MSTPC8 bits control module stop mode. When these bits are set to 1, the corresponding modules enter module stop mode. When these bits are cleared to 0, module stop mode is cancelled. When bits MSTPC3 to MSTPC0 are set to 1, the corresponding on-chip RAM stops. Do not set the corresponding MSTPC3 to MSTPC0 bits to 1 while accessing on-chip RAM.

Bit	15	14	13	12	11	10	9	8
Bit Name	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Module
15	MSTPC15	1	R/W	Sound generator (SDG)
14	MSTPC14	1	R/W	16-bit PWM
13	MSTPC13	1	R/W	Motor control PWM
12	MSTPC12	1	R/W	D/A converter (channels 1 and 0)
11	MSTPC11	1	R/W	Controller area network_1, 0 (RCAN-ET_1, RCAN-ET_0)
10	MSTPC10	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
9	MSTPC9	1	R/W	Synchronous serial communication unit_1 (SSU_1)
8	MSTPC8	1	R/W	Synchronous serial communication unit_0 (SSU_0)
7	MSTPC7	0	R/W	Reserved
6	MSTPC6	0	R/W	These bits are always read as 0. The write value should always be 0.
5	MSTPC5	0	R/W	
4	MSTPC4	0	R/W	
3	MSTPC3	0	R/W	On-chip RAM_1 (H'FF6000 to H'FF7FFF)
2	MSTPC2	0	R/W	Write the same value to MSTPC3 and MSTPC2.
1	MSTPC1	0	R/W	On-chip RAM_0 (H'FF8000 to H'FFBFFF)
0	MSTPC0	0	R/W	Write the same value to MSTPC3 and MSTPC2.

RENESAS

24.3   Multi-Clock Function of Main Clock

When the ICK2 to ICK0 bits, PCK2 to PCK0 bits, and BCK2 to BCK0 bits in the SCKCR register are set, the multi-clock function is enabled at the end of the current bus cycle. With the multi-clock function, the CPU and bus masters are driven by the clock set by the ICK2 to ICK0 bits; while the peripheral modules are driven by the clock set by the PCK2 to PCK0 bits, and the external bus clock is driven by the clock set by the BCK2 to BCK0 bits.

Note, however, that if the clock frequencies selected by the PCK2 to PCK0 bits and BCK2 to BCK0 bits are higher than that set by the ICK2 to ICK0 bits, the setting values will not be reflected on the clock signals. The peripheral module clock and external bus clock frequencies are restricted to that of the operating clock set by the ICK2 to ICK0 bits.

The multi-clock function can be cancelled by clearing all of the ICK2 to ICK0 bits, PCK2 to PCK0 bits, and BCK2 to BCK0 bits to 0s. When the ICK2 to ICK0, PCK2 to PCK0, and BCK2 to BCK0 bits are cleared to 0, this LSI enters the normal state at the end of the bus cycle and the multi-clock function is cancelled.

If a SLEEP instruction is executed while the SSBY bit in SBYCR is cleared to 0, this LSI enters sleep mode. In sleep mode, the multi-clock function is still enabled. If a SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1, this LSI enters software standby mode. In software standby mode, the multi-clock function is stopped. After software standby mode is cancelled by an interrupt, the multi-clock function is enabled again.

When the $\overline{\text{RES}}$ pin is brought low, this LSI enters the reset state and the multi-clock function is cancelled. If this LSI is reset by an overflow of the watchdog timer, the multi-clock function is also cancelled.

When the $\overline{\text{STBY}}$ pin is brought low, this LSI enters hardware standby mode and the multi-clock function is cancelled.

24.4   Sub-Clock

When the SUBCKSEL bit in SUBCKCR is set to 1, the sub-clock function is enabled at the end of the current bus cycle regardless of the SCKCR setting. With the sub-clock function, the CPU, bus master, peripheral modules, and external bus clock all operate at the frequency of the main clock divided by 128.

Clearing the SUBCKSEL bit to 0 cancels the sub-clock function. It is cancelled when the oscillation stabilization time has elapsed after the end of the bus cycle.

If a SLEEP instruction is executed while the SSBY bit in SBYCR is cleared to 0, this LSI enters sleep mode. In sleep mode, the sub-clock function is still enabled.

If a SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1, this LSI enters software standby mode. In software standby mode, the sub-clock function is stopped but the sub-clock is provided to the WAT while the XTALSTP bit in SUBCKCR is cleared to 0. When software standby mode is cancelled by an interrupt, this LSI returns to the state in which the sub-clock function is enabled if WKCKSEL = 1, or returns to the state in which it is driven by the clock set by SCKCR if WKCKSEL = 0.

When the $\overline{\text{RES}}$ pin is brought low, this LSI enters the reset state and the sub-clock function is cancelled. When this LSI is reset by an overflow of the watchdog timer, the sub-clock function is also cancelled.

When the $\overline{\text{STBY}}$ pin is brought low, this LSI enters hardware standby mode and the sub-clock function is cancelled.

24.5   Sleep Mode

24.5.1   Transition to Sleep Mode

When the SLEEP instruction is executed when the SSBY bit in SBYCR is 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other peripheral functions do not stop.

24.5.2   Clearing Sleep Mode

Sleep mode is exited by any interrupt, signals on the $\overline{\text{RES}}$ or $\overline{\text{STBY}}$ pin, and a reset caused by a watchdog timer overflow.

1. Clearing by interrupt

   When an interrupt occurs, sleep mode is exited and interrupt exception processing starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.

2. Clearing by $\overline{\text{RES}}$ pin

   Setting the $\overline{\text{RES}}$ pin level low selects the reset state. After the stipulated reset input duration, driving the $\overline{\text{RES}}$ pin high makes the CPU start the reset exception processing.

3. Clearing by $\overline{\text{STBY}}$ pin

   When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.

4. Clearing by reset caused by watchdog timer overflow

   Sleep mode is exited by an internal reset caused by a watchdog timer overflow.

24.6   Software Standby Mode

24.6.1   Transition to Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip peripheral functions, and oscillator all stop. However, the contents of the CPU's internal registers, on-chip RAM data, and the states of on-chip peripheral functions other than the SCI, and the states of the I/O ports, are retained. Whether the address bus and bus control signals are placed in the high-impedance state or retain the output state can be specified by the OPE bit in SBYCR. In this mode the oscillator stops, allowing power consumption to be significantly reduced.

If the WDT is used as a watchdog timer, it is impossible to make a transition to software standby mode. The WDT should be stopped before the SLEEP instruction execution.

24.6.2   Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, or pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ15}}$*), an internal interrupt (WAT), or by means of the $\overline{\text{RES}}$ pin or $\overline{\text{STBY}}$ pin.

1. Clearing by interrupt

    When an NMI, IRQ0 to IRQ15* interrupt, or internal interrupt (WAT) request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS4 to STS0 in SBYCR, stable clocks are supplied to the entire LSI, software standby mode is cleared, and interrupt exception handling is started.

    When clearing software standby mode with an IRQ0 to IRQ15* interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ15* is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side.

    Note that the vector address for the WAT interrupt exception handling differs between the cases when the WAT interrupt cancels the software standby mode and when the WAT interrupt occurs during normal operation.

    Note: *   By setting the SSIn bit in SSIER to 1, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ15}}$ can be used as a software standby mode clearing source.

2. Clearing by $\overline{\text{RES}}$ pin

    When the $\overline{\text{RES}}$ pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire LSI. Note that the $\overline{\text{RES}}$ pin must be held low until clock oscillation settles. When the $\overline{\text{RES}}$ pin goes high, the CPU begins reset exception handling.

3. Clearing by $\overline{\text{STBY}}$ pin

    When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

RENESAS

24.6.3   Setting Oscillation Settling Time after Clearing Software Standby Mode

Bits STS4 to STS0 in SBYCR should be set as described below.

1. Using a crystal resonator

   Set bits STS4 to STS0 so that the standby time is at least equal to the oscillation settling time.

   Table 24.2 shows the standby times for operating frequencies and settings of bits STS4 to STS0.

2. Using an external clock

   A PLL circuit settling time is necessary. Refer to table 24.2 to set the standby time.

Table 24.2   Oscillation Settling Time Settings

STS4	STS3	STS2	STS1	STS0	Standby Time (states)	Pφ* [MHz] 20	13	10	8	Unit
0	0	0	0	0	Reserved	—	—	—	—	μs
				1	Reserved	—	—	—	—	
			1	0	8	0.4	0.6	0.8	1.0	
				1	16	0.8	1.2	1.6	2.0	
		1	0	0	32	1.6	2.46	3.2	4.0	
				1	64	3.2	4.9	6.4	8.0	
			1	0	512	25.6	39.4	51.2	64.0	
				1	1024	51.2	78.8	102.4	128.0	
	1	0	0	0	2048	102.4	157.5	204.8	256.0	
				1	4096	0.20	0.32	0.41	0.51	ms
		1	0	0	16384	0.82	1.26	1.64	2.05	
				1	32768	1.64	2.52	3.28	4.10	
			1	0	65536	3.28	5.04	6.55	8.19	
				1	131072	6.55	10.08	13.11	16.38	
			1	0	262144	13.11	20.16	26.21	32.77	
				1	524288	26.21	40.33	52.43	65.54	
1	0	0	0	0	Reserved	—	—	—	—	

▨ : Recommended time setting when using an external clock.

▢ : Recommended time setting when using a crystal resonator.

Note: *   Pφ is the output from the peripheral module frequency divider.

RENESAS

24.6.4   Software Standby Mode Application Example

Figure 24.2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in INTCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.
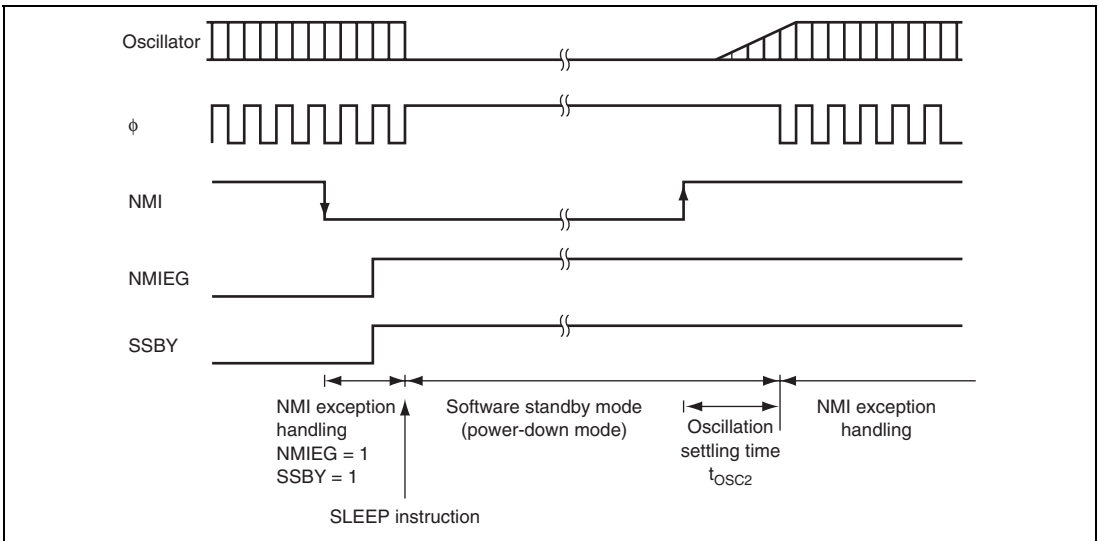
Figure 24.2   Software Standby Mode Application Example

24.7   Hardware Standby Mode

24.7.1   Transition to Hardware Standby Mode

When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power consumption. As long as the prescribed voltage is supplied, on-chip RAM data is retained. 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 (MD2 to MD0) while this LSI is in hardware standby mode.

24.7.2   Clearing Hardware Standby Mode

Hardware standby mode is cleared by means of the $\overline{\text{STBY}}$ pin and the $\overline{\text{RES}}$ pin. When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, the reset state is entered and clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until clock oscillation settles (for details on the oscillation settling time, refer to table 24.2). When the $\overline{\text{RES}}$ pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

24.7.3   Hardware Standby Mode Timing

Figure 24.3 shows an example of hardware standby mode timing.

When the $\overline{\text{STBY}}$ pin is driven low after the $\overline{\text{RES}}$ pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the $\overline{\text{STBY}}$ pin high, waiting for the oscillation settling time, then changing the $\overline{\text{RES}}$ pin from low to high.
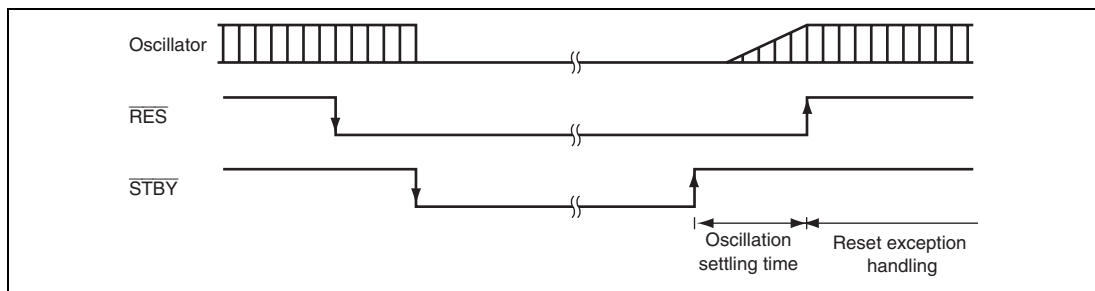
Figure 24.3   Hardware Standby Mode Timing

24.7.4   Timing Sequence at Power-On

Figure 24.4 shows the timing sequence at power-on.

At power-on, the $\overline{\text{RES}}$ pin must be driven low with the $\overline{\text{STBY}}$ pin driven high for a given time in order to clear the reset state.

To enter hardware standby mode immediately after power-on, drive the $\overline{\text{STBY}}$ pin low after exiting the reset state.

For details on clearing hardware standby mode, see section 24.7.3, Hardware Standby Mode Timing.
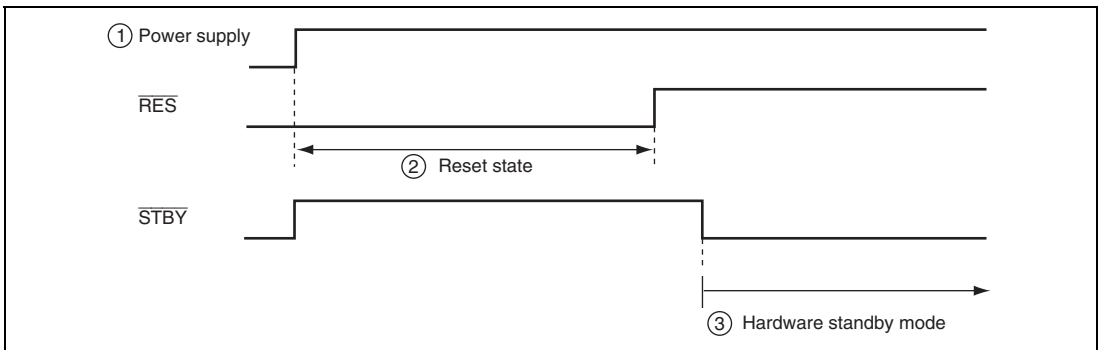
Figure 24.4   Timing Sequence at Power-On

24.8   Module Stop Function

24.8.1   Module Stop Function

The module stop function can be set for individual on-chip peripheral modules.

When the corresponding MSTP bit in MSTPCRA, MSTPCRB, or MSTPCRC is set to 1, module operation stops at the end of the bus cycle and the module enters a module stop state. The CPU continues operating independently.

When the corresponding MSTP bit is cleared to 0, the module stop state is exited and the module starts operating at the end of the bus cycle. In the module stop state, some SCI registers and the internal states of the motor control PWM, 16-bit PWM, RCAN, SSU*, and SDG are reset, while the internal states of other peripheral modules are retained.

After the reset state is cleared, all modules other than the DMAC and on-chip RAM are in the module stop state.

The registers of the modules placed in the module stop state cannot be read from or written to.

Note: *   SSU: Synchronous Serial communication Unit

24.8.2   All-Module-Clock-Stop Mode

When the ACSE bit is set to 1 and all modules controlled by MSTPCRA and MSTPCRB are stopped (MSTPCRA, MSTPCRB = H'FFFFFFFF), executing a SLEEP instruction with the SSBY bit in SBYCR cleared to 0 will cause all modules (except for the watchdog timer, WAT, and modules controlled by MSTPCRC), the bus controller, and the I/O ports to stop operating, and to make a transition to all-module-clock-stop mode at the end of the bus cycle.

If further reduction of power consumption is required in all-module-clock-stop mode, stop the modules that MSTPCRC controls (MSTPCRC[15:8] = H'FFFF).

All-module-clock-stop mode is cleared by an external interrupt (NMI or $\overline{IRQ0}$ to $\overline{IRQ15}$ pins), $\overline{RES}$ pin input, or an internal interrupt (watchdog timer or WAT), and the CPU returns to the normal program execution state via the exception handling state. All-module-clock-stop mode is not cleared if interrupts are disabled or interrupts other than NMI are masked on the CPU side.

When the $\overline{STBY}$ pin is driven low, a transition is made to hardware standby mode.

24.9   Bϕ Clock Output Control

Output of the Bϕ clock can be controlled by bits PSTOP1 and POSEL1 in SCKCR, and DDR for the corresponding PA7 pin.

Clearing both bits PSTOP1 and POSEL1 to 0 enables the Bϕ clock output on the PA7 pin. When bit PSTOP1 is set to 1, the Bϕ clock output stops at the end of the bus cycle, and the Bϕ clock output goes high. When DDR for the PA7 pin is cleared to 0, the Bϕ clock output is disabled and the pin becomes an input port.

Tables 24.3 shows the states of the Bϕ pin in each processing state.

Table 24.3   Bϕ Pin (PA7) State in Each Processing State

Register Setting Value			Normal Operating State	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode		Hardware Standby Mode
DDR	PSTOP1	POSEL1				OPE = 0	OPE = 1	
0	×	×	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
1	0	0	Bϕ output	Bϕ output	Bϕ output	High	High	Hi-Z
1	0	1	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
1	1	×	High	High	High	High	High	Hi-Z

24.10   Usage Notes

24.10.1   I/O Port Status

In software standby mode, the I/O port states are retained. Therefore, there is no reduction in current consumption for the output current when a high-level signal is output.

24.10.2   Current Consumption during Oscillation Settling Standby Period

Current consumption increases during the oscillation settling standby period.

24.10.3   Module Stop Mode of DMAC

Depending on the operating state of the DMAC, the MSTPA13 bit may not be set to 1. The module stop mode of the DMAC should be set only when the DMAC is not activated.

For details, refer to section 7, DMA Controller (DMAC).

24.10.4   On-Chip Peripheral Module Interrupts

Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC activation source. Interrupts should therefore be disabled before entering module stop mode.

24.10.5   Writing to MSTPCRA, MSTPCRB, and MSTPCRC

MSTPCRA, MSTPCRB, and MSTPCRC should only be written to by the CPU.

Section 25   List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register addresses (address order)
- Registers are listed from the lower allocation addresses.
- Registers are classified according to functional modules.
- Undefined and reserved addresses cannot be accessed. Do not access these addresses; otherwise, the operation when accessing these bits and subsequent operations cannot be guaranteed.

2. Register bits
- Bit configurations of the registers are listed in the same order as the register addresses.
- Reserved bits are indicated by — in the bit name column.
- Space in the bit name field indicates that the entire register is allocated to either the counter or data.
- For the registers of 16 or 32 bits, the MSB is listed first.
  Byte configuration description order is subject to big endian.

3. Register states in each operating mode
- Register states are listed in the same order as the register addresses.
- For the initialized state of each bit, refer to the register description in the corresponding section.
- The register states shown here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

RENESAS

25.1   Register Addresses (Address Order)

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Master control register_0	MCR_0	16	H'FEA00	RCAN-ET_0	16	4Pφ/4Pφ
General status register_0	GSR_0	16	H'FEA02	RCAN-ET_0	16	4Pφ/4Pφ
Bit configuration register 1_0	BCR1_0	16	H'FEA04	RCAN-ET_0	16	4Pφ/4Pφ
Bit configuration register 0_0	BCR0_0	16	H'FEA06	RCAN-ET_0	16	4Pφ/4Pφ
Interrupt register_0	IRR_0	16	H'FEA08	RCAN-ET_0	16	4Pφ/4Pφ
Interrupt mask register_0	IMR_0	16	H'FEA0A	RCAN-ET_0	16	4Pφ/4Pφ
Transmit error counter_0	TEC_0	16	H'FEA0C	RCAN-ET_0	16	4Pφ/4Pφ
Transmit error counter_0	REC_0					
Transmit wait register 1_0	TXPR1_0	16	H'FEA20	RCAN-ET_0	16	4Pφ/4Pφ
Transmit wait register 0_0	TXPR0_0	16	H'FEA22	RCAN-ET_0	16	4Pφ/4Pφ
Transmit wait cancel register 0_0	TXCR0_0	16	H'FEA2A	RCAN-ET_0	16	4Pφ/4Pφ
Transmit acknowledge register 0_0	TXACK0_0	16	H'FEA32	RCAN-ET_0	16	4Pφ/4Pφ
Abort acknowledge register 0_0	ABACK0_0	16	H'FEA3A	RCAN-ET_0	16	4Pφ/4Pφ
Data frame receive pending register 0_0	RXPR0_0	16	H'FEA42	RCAN-ET_0	16	4Pφ/4Pφ
Remote frame receive pending register 0_0	RFPR0_0	16	H'FEA4A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox interrupt mask register 0_0	MBIMR0_0	16	H'FEA52	RCAN-ET_0	16	4Pφ/4Pφ
Unread message status register 0_0	UMSR0_0	16	H'FEA5A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_control 0H	MB_0[0].CONTROL0H	16	H'FEB00	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_control 0L	MB_0[0].CONTROL0L	16	H'FEB02	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_LAFMH	MB_0[0].LAFMH	16	H'FEB04	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_LAFML	MB_0[0].LAFML	16	H'FEB06	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 0	MB_0[0].MSG_DATA[0]	8	H'FEB08	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 1	MB_0[0].MSG_DATA[1]	8	H'FEB09	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 2	MB_0[0].MSG_DATA[2]	8	H'FEB0A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 3	MB_0[0].MSG_DATA[3]	8	H'FEB0B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 4	MB_0[0].MSG_DATA[4]	8	H'FEB0C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 5	MB_0[0].MSG_DATA[5]	8	H'FEB0D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 6	MB_0[0].MSG_DATA[6]	8	H'FEB0E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_data 7	MB_0[0].MSG_DATA[7]	8	H'FEB0F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_control 1H	MB_0[0].CONTROL1H	8	H'FEB10	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 0_0_control 1L	MB_0[0].CONTROL1L	8	H'FEB11	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_control 0H	MB_0[1].CONTROL0H	16	H'FEB20	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_control 0L	MB_0[1].CONTROL0L	16	H'FEB22	RCAN-ET_0	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 1_0_LAFMH	MB_0[1].LAFMH	16	H'FEB24	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_LAFML	MB_0[1].LAFML	16	H'FEB26	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 0	MB_0[1].MSG_DATA[0]	8	H'FEB28	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 1	MB_0[1].MSG_DATA[1]	8	H'FEB29	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 2	MB_0[1].MSG_DATA[2]	8	H'FEB2A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 3	MB_0[1].MSG_DATA[3]	8	H'FEB2B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 4	MB_0[1].MSG_DATA[4]	8	H'FEB2C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 5	MB_0[1].MSG_DATA[5]	8	H'FEB2D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 6	MB_0[1].MSG_DATA[6]	8	H'FEB2E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_data 7	MB_0[1].MSG_DATA[7]	8	H'FEB2F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_control 1H	MB_0[1].CONTROL1H	8	H'FEB30	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 1_0_control 1L	MB_0[1].CONTROL1L	8	H'FEB31	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_control 0H	MB_0[2].CONTROL0H	16	H'FEB40	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_control 0L	MB_0[2].CONTROL0L	16	H'FEB42	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_LAFMH	MB_0[2].LAFMH	16	H'FEB44	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_LAFML	MB_0[2].LAFML	16	H'FEB46	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 0	MB_0[2].MSG_DATA[0]	8	H'FEB48	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 1	MB_0[2].MSG_DATA[1]	8	H'FEB49	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 2	MB_0[2].MSG_DATA[2]	8	H'FEB4A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 3	MB_0[2].MSG_DATA[3]	8	H'FEB4B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 4	MB_0[2].MSG_DATA[4]	8	H'FEB4C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 5	MB_0[2].MSG_DATA[5]	8	H'FEB4D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 6	MB_0[2].MSG_DATA[6]	8	H'FEB4E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_data 7	MB_0[2].MSG_DATA[7]	8	H'FEB4F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_control 1H	MB_0[2].CONTROL1H	8	H'FEB50	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 2_0_control 1L	MB_0[2].CONTROL1L	8	H'FEB51	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_control 0H	MB_0[3].CONTROL0H	16	H'FEB60	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_control 0L	MB_0[3].CONTROL0L	16	H'FEB62	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_LAFMH	MB_0[3].LAFMH	16	H'FEB64	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_LAFML	MB_0[3].LAFML	16	H'FEB66	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 0	MB_0[3].MSG_DATA[0]	8	H'FEB68	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 1	MB_0[3].MSG_DATA[1]	8	H'FEB69	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 2	MB_0[3].MSG_DATA[2]	8	H'FEB6A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 3	MB_0[3].MSG_DATA[3]	8	H'FEB6B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 4	MB_0[3].MSG_DATA[4]	8	H'FEB6C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 5	MB_0[3].MSG_DATA[5]	8	H'FEB6D	RCAN-ET_0	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 3_0_data 6	MB_0[3].MSG_DATA[6]	8	H'FEB6E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_data 7	MB_0[3].MSG_DATA[7]	8	H'FEB6F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_control 1H	MB_0[3].CONTROL1H	8	H'FEB70	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 3_0_control 1L	MB_0[3].CONTROL1L	8	H'FEB71	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_control 0H	MB_0[4].CONTROL0H	16	H'FEB80	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_control 0L	MB_0[4].CONTROL0L	16	H'FEB82	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_LAFMH	MB_0[4].LAFMH	16	H'FEB84	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_LAFML	MB_0[4].LAFML	16	H'FEB86	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 0	MB_0[4].MSG_DATA[0]	8	H'FEB88	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 1	MB_0[4].MSG_DATA[1]	8	H'FEB89	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 2	MB_0[4].MSG_DATA[2]	8	H'FEB8A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 3	MB_0[4].MSG_DATA[3]	8	H'FEB8B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 4	MB_0[4].MSG_DATA[4]	8	H'FEB8C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 5	MB_0[4].MSG_DATA[5]	8	H'FEB8D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 6	MB_0[4].MSG_DATA[6]	8	H'FEB8E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_data 7	MB_0[4].MSG_DATA[7]	8	H'FEB8F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_control 1H	MB_0[4].CONTROL1H	8	H'FEB90	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 4_0_control 1L	MB_0[4].CONTROL1L	8	H'FEB91	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_control 0H	MB_0[5].CONTROL0H	16	H'FEBA0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_control 0L	MB_0[5].CONTROL0L	16	H'FEBA2	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_LAFMH	MB_0[5].LAFMH	16	H'FEBA4	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_LAFML	MB_0[5].LAFML	16	H'FEBA6	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 0	MB_0[5].MSG_DATA[0]	8	H'FEBA8	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 1	MB_0[5].MSG_DATA[1]	8	H'FEBA9	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 2	MB_0[5].MSG_DATA[2]	8	H'FEBAA	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 3	MB_0[5].MSG_DATA[3]	8	H'FEBAB	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 4	MB_0[5].MSG_DATA[4]	8	H'FEBAC	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 5	MB_0[5].MSG_DATA[5]	8	H'FEBAD	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 6	MB_0[5].MSG_DATA[6]	8	H'FEBAE	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_data 7	MB_0[5].MSG_DATA[7]	8	H'FEBAF	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_control 1H	MB_0[5].CONTROL1H	8	H'FEBB0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 5_0_control 1L	MB_0[5].CONTROL1L	8	H'FEBB1	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_control 0H	MB_0[6].CONTROL0H	16	H'FEBC0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_control 0L	MB_0[6].CONTROL0L	16	H'FEBC2	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_LAFMH	MB_0[6].LAFMH	16	H'FEBC4	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_LAFML	MB_0[6].LAFML	16	H'FEBC6	RCAN-ET_0	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*1	Module	Data Width	Access Cycles*4 (Read/Write)
Mailbox 6_0_data 0	MB_0[6].MSG_DATA[0]	8	H'FEBC8	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 1	MB_0[6].MSG_DATA[1]	8	H'FEBC9	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 2	MB_0[6].MSG_DATA[2]	8	H'FEBCA	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 3	MB_0[6].MSG_DATA[3]	8	H'FEBCB	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 4	MB_0[6].MSG_DATA[4]	8	H'FEBCC	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 5	MB_0[6].MSG_DATA[5]	8	H'FEBCD	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 6	MB_0[6].MSG_DATA[6]	8	H'FEBCE	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_data 7	MB_0[6].MSG_DATA[7]	8	H'FEBCF	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_control 1H	MB_0[6].CONTROL1H	8	H'FEBD0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 6_0_control 1L	MB_0[6].CONTROL1L	8	H'FEBD1	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_control 0H	MB_0[7].CONTROL0H	16	H'FEBE0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_control 0L	MB_0[7].CONTROL0L	16	H'FEBE2	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_LAFMH	MB_0[7].LAFMH	16	H'FEBE4	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_LAFML	MB_0[7].LAFML	16	H'FEBE6	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 0	MB_0[7].MSG_DATA[0]	8	H'FEBE8	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 1	MB_0[7].MSG_DATA[1]	8	H'FEBE9	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 2	MB_0[7].MSG_DATA[2]	8	H'FEBEA	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 3	MB_0[7].MSG_DATA[3]	8	H'FEBEB	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 4	MB_0[7].MSG_DATA[4]	8	H'FEBEC	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 5	MB_0[7].MSG_DATA[5]	8	H'FEBED	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 6	MB_0[7].MSG_DATA[6]	8	H'FEBEE	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_data 7	MB_0[7].MSG_DATA[7]	8	H'FEBEF	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_control 1H	MB_0[7].CONTROL1H	8	H'FEBF0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 7_0_control 1L	MB_0[7].CONTROL1L	8	H'FEBF1	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_control 0H	MB_0[8].CONTROL0H	16	H'FEC00	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_control 0L	MB_0[8].CONTROL0L	16	H'FEC02	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_LAFMH	MB_0[8].LAFMH	16	H'FEC04	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_LAFML	MB_0[8].LAFML	16	H'FEC06	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 0	MB_0[8].MSG_DATA[0]	8	H'FEC08	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 1	MB_0[8].MSG_DATA[1]	8	H'FEC09	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 2	MB_0[8].MSG_DATA[2]	8	H'FEC0A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 3	MB_0[8].MSG_DATA[3]	8	H'FEC0B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 4	MB_0[8].MSG_DATA[4]	8	H'FEC0C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 5	MB_0[8].MSG_DATA[5]	8	H'FEC0D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 6	MB_0[8].MSG_DATA[6]	8	H'FEC0E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_data 7	MB_0[8].MSG_DATA[7]	8	H'FEC0F	RCAN-ET_0	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 8_0_control 1H	MB_0[8].CONTROL1H	8	H'FEC10	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 8_0_control 1L	MB_0[8].CONTROL1L	8	H'FEC11	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_control 0H	MB_0[9].CONTROL0H	16	H'FEC20	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_control 0L	MB_0[9].CONTROL0L	16	H'FEC22	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_LAFMH	MB_0[9].LAFMH	16	H'FEC24	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_LAFML	MB_0[9].LAFML	16	H'FEC26	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 0	MB_0[9].MSG_DATA[0]	8	H'FEC28	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 1	MB_0[9].MSG_DATA[1]	8	H'FEC29	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 2	MB_0[9].MSG_DATA[2]	8	H'FEC2A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 3	MB_0[9].MSG_DATA[3]	8	H'FEC2B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 4	MB_0[9].MSG_DATA[4]	8	H'FEC2C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 5	MB_0[9].MSG_DATA[5]	8	H'FEC2D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 6	MB_0[9].MSG_DATA[6]	8	H'FEC2E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_data 7	MB_0[9].MSG_DATA[7]	8	H'FEC2F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_control 1H	MB_0[9].CONTROL1H	8	H'FEC30	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 9_0_control 1L	MB_0[9].CONTROL1L	8	H'FEC31	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_control 0H	MB_0[10].CONTROL0H	16	H'FEC40	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_control 0L	MB_0[10].CONTROL0L	16	H'FEC42	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_LAFMH	MB_0[10].LAFMH	16	H'FEC44	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_LAFML	MB_0[10].LAFML	16	H'FEC46	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 0	MB_0[10].MSG_DATA[0]	8	H'FEC48	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 1	MB_0[10].MSG_DATA[1]	8	H'FEC49	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 2	MB_0[10].MSG_DATA[2]	8	H'FEC4A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 3	MB_0[10].MSG_DATA[3]	8	H'FEC4B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 4	MB_0[10].MSG_DATA[4]	8	H'FEC4C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 5	MB_0[10].MSG_DATA[5]	8	H'FEC4D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 6	MB_0[10].MSG_DATA[6]	8	H'FEC4E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_data 7	MB_0[10].MSG_DATA[7]	8	H'FEC4F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_control 1H	MB_0[10].CONTROL1H	8	H'FEC50	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 10_0_control 1L	MB_0[10].CONTROL1L	8	H'FEC51	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_control 0H	MB_0[11].CONTROL0H	16	H'FEC60	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_control 0L	MB_0[11].CONTROL0L	16	H'FEC62	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_LAFMH	MB_0[11].LAFMH	16	H'FEC64	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_LAFML	MB_0[11].LAFML	16	H'FEC66	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 0	MB_0[11].MSG_DATA[0]	8	H'FEC68	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 1	MB_0[11].MSG_DATA[1]	8	H'FEC69	RCAN-ET_0	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 11_0_data 2	MB_0[11].MSG_DATA[2]	8	H'FEC6A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 3	MB_0[11].MSG_DATA[3]	8	H'FEC6B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 4	MB_0[11].MSG_DATA[4]	8	H'FEC6C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 5	MB_0[11].MSG_DATA[5]	8	H'FEC6D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 6	MB_0[11].MSG_DATA[6]	8	H'FEC6E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_data 7	MB_0[11].MSG_DATA[7]	8	H'FEC6F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_control 1H	MB_0[11].CONTROL1H	8	H'FEC70	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 11_0_control 1L	MB_0[11].CONTROL1L	8	H'FEC71	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_control 0H	MB_0[12].CONTROL0H	16	H'FEC80	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_control 0L	MB_0[12].CONTROL0L	16	H'FEC82	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_LAFMH	MB_0[12].LAFMH	16	H'FEC84	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_LAFML	MB_0[12].LAFML	16	H'FEC86	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 0	MB_0[12].MSG_DATA[0]	8	H'FEC88	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 1	MB_0[12].MSG_DATA[1]	8	H'FEC89	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 2	MB_0[12].MSG_DATA[2]	8	H'FEC8A	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 3	MB_0[12].MSG_DATA[3]	8	H'FEC8B	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 4	MB_0[12].MSG_DATA[4]	8	H'FEC8C	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 5	MB_0[12].MSG_DATA[5]	8	H'FEC8D	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 6	MB_0[12].MSG_DATA[6]	8	H'FEC8E	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_data 7	MB_0[12].MSG_DATA[7]	8	H'FEC8F	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_control 1H	MB_0[12].CONTROL1H	8	H'FEC90	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 12_0_control 1L	MB_0[12].CONTROL1L	8	H'FEC91	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_control 0H	MB_0[13].CONTROL0H	16	H'FECA0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_control 0L	MB_0[13].CONTROL0L	16	H'FECA2	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_LAFMH	MB_0[13].LAFMH	16	H'FECA4	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_LAFML	MB_0[13].LAFML	16	H'FECA6	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 0	MB_0[13].MSG_DATA[0]	8	H'FECA8	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 1	MB_0[13].MSG_DATA[1]	8	H'FECA9	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 2	MB_0[13].MSG_DATA[2]	8	H'FECAA	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 3	MB_0[13].MSG_DATA[3]	8	H'FECAB	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 4	MB_0[13].MSG_DATA[4]	8	H'FECAC	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 5	MB_0[13].MSG_DATA[5]	8	H'FECAD	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 6	MB_0[13].MSG_DATA[6]	8	H'FECAE	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_data 7	MB_0[13].MSG_DATA[7]	8	H'FECAF	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_control 1H	MB_0[13].CONTROL1H	8	H'FECB0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 13_0_control 1L	MB_0[13].CONTROL1L	8	H'FECB1	RCAN-ET_0	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 14_0_control 0H	MB_0[14].CONTROL0H	16	H'FECC0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_control 0L	MB_0[14].CONTROL0L	16	H'FECC2	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_LAFMH	MB_0[14].LAFMH	16	H'FECC4	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_LAFML	MB_0[14].LAFML	16	H'FECC6	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 0	MB_0[14].MSG_DATA[0]	8	H'FECC8	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 1	MB_0[14].MSG_DATA[1]	8	H'FECC9	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 2	MB_0[14].MSG_DATA[2]	8	H'FECCA	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 3	MB_0[14].MSG_DATA[3]	8	H'FECCB	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 4	MB_0[14].MSG_DATA[4]	8	H'FECCC	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 5	MB_0[14].MSG_DATA[5]	8	H'FECCD	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 6	MB_0[14].MSG_DATA[6]	8	H'FECCE	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_data 7	MB_0[14].MSG_DATA[7]	8	H'FECCF	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_control 1H	MB_0[14].CONTROL1H	8	H'FECD0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 14_0_control 1L	MB_0[14].CONTROL1L	8	H'FECD1	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_control 0H	MB_0[15].CONTROL0H	16	H'FECE0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_control 0L	MB_0[15].CONTROL0L	16	H'FECE2	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_LAFMH	MB_0[15].LAFMH	16	H'FECE4	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_LAFML	MB_0[15].LAFML	16	H'FECE6	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 0	MB_0[15].MSG_DATA[0]	8	H'FECE8	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 1	MB_0[15].MSG_DATA[1]	8	H'FECE9	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 2	MB_0[15].MSG_DATA[2]	8	H'FECEA	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 3	MB_0[15].MSG_DATA[3]	8	H'FECEB	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 4	MB_0[15].MSG_DATA[4]	8	H'FECEC	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 5	MB_0[15].MSG_DATA[5]	8	H'FECED	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 6	MB_0[15].MSG_DATA[6]	8	H'FECEE	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_data 7	MB_0[15].MSG_DATA[7]	8	H'FECEF	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_control 1H	MB_0[15].CONTROL1H	8	H'FECF0	RCAN-ET_0	16	4Pφ/4Pφ
Mailbox 15_0_control 1L	MB_0[15].CONTROL1L	8	H'FECF1	RCAN-ET_0	16	4Pφ/4Pφ
Master control register_1	MCR_1	16	H'FF000	RCAN-ET_1	16	4Pφ/4Pφ
General status register_1	GSR_1	16	H'FF002	RCAN-ET_1	16	4Pφ/4Pφ
Bit configuration register 1_1	BCR1_1	16	H'FF004	RCAN-ET_1	16	4Pφ/4Pφ
Bit configuration register 0_1	BCR0_1	16	H'FF006	RCAN-ET_1	16	4Pφ/4Pφ
Interrupt register_1	IRR_1	16	H'FF008	RCAN-ET_1	16	4Pφ/4Pφ
Interrupt mask register_1	IMR_1	16	H'FF00A	RCAN-ET_1	16	4Pφ/4Pφ
Transmit error counter_1	TEC_1	16	H'FF00C	RCAN-ET_1	16	4Pφ/4Pφ
Receive error counter_1	REC_1					

RENESAS

Register Name	Abbreviation	Number of Bits	Address*1	Module	Data Width	Access Cycles*4 (Read/Write)
Transmit wait register 1_1	TXPR1_1	16	H'FF020	RCAN-ET_1	16	4Pφ/4Pφ
Transmit wait register 0_1	TXPR0_1	16	H'FF022	RCAN-ET_1	16	4Pφ/4Pφ
Transmit wait cancel register 0_1	TXCR0_1	16	H'FF02A	RCAN-ET_1	16	4Pφ/4Pφ
Transmit acknowledge register 0_1	TXACK0_1	16	H'FF032	RCAN-ET_1	16	4Pφ/4Pφ
Abort acknowledge register 0_1	ABACK0_1	16	H'FF03A	RCAN-ET_1	16	4Pφ/4Pφ
Receive complete register 0_1	RXPR0_1	16	H'FF042	RCAN-ET_1	16	4Pφ/4Pφ
Remote request register 0_1	RFPR0_1	16	H'FF04A	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox interrupt mask register 0_1	MBIMR0_1	16	H'FF052	RCAN-ET_1	16	4Pφ/4Pφ
Unread message status register 0_1	UMSR0_1	16	H'FF05A	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_control 0H	MB_1[0].CONTROL0H	16	H'FF100	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_control 0L	MB_1[0].CONTROL0L	16	H'FF102	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_LAFMH	MB_1[0].LAFMH	16	H'FF104	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_LAFML	MB_1[0].LAFML	16	H'FF106	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 0	MB_1[0].MSG_DATA[0]	8	H'FF108	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 1	MB_1[0].MSG_DATA[1]	8	H'FF109	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 2	MB_1[0].MSG_DATA[2]	8	H'FF10A	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 3	MB_1[0].MSG_DATA[3]	8	H'FF10B	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 4	MB_1[0].MSG_DATA[4]	8	H'FF10C	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 5	MB_1[0].MSG_DATA[5]	8	H'FF10D	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 6	MB_1[0].MSG_DATA[6]	8	H'FF10E	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_data 7	MB_1[0].MSG_DATA[7]	8	H'FF10F	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_control 1H	MB_1[0].CONTROL1H	8	H'FF110	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 0_1_control 1L	MB_1[0].CONTROL1L	8	H'FF111	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_control 0H	MB_1[1].CONTROL0H	16	H'FF120	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_control 0L	MB_1[1].CONTROL0L	16	H'FF122	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_LAFMH	MB_1[1].LAFMH	16	H'FF124	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_LAFML	MB_1[1].LAFML	16	H'FF126	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 0	MB_1[1].MSG_DATA[0]	8	H'FF128	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 1	MB_1[1].MSG_DATA[1]	8	H'FF129	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 2	MB_1[1].MSG_DATA[2]	8	H'FF12A	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 3	MB_1[1].MSG_DATA[3]	8	H'FF12B	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 4	MB_1[1].MSG_DATA[4]	8	H'FF12C	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 5	MB_1[1].MSG_DATA[5]	8	H'FF12D	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 6	MB_1[1].MSG_DATA[6]	8	H'FF12E	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 1_1_data 7	MB_1[1].MSG_DATA[7]	8	H'FF12F	RCAN-ET_1	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 1_1_control 1H	MB_1[1].CONTROL1H	8	H'FF130	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 1_1_control 1L	MB_1[1].CONTROL1L	8	H'FF131	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_control 0H	MB_1[2].CONTROL0H	16	H'FF140	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_control 0L	MB_1[2].CONTROL0L	16	H'FF142	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_LAFMH	MB_1[2].LAFMH	16	H'FF144	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_LAFML	MB_1[2].LAFML	16	H'FF146	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 0	MB_1[2].MSG_DATA[0]	8	H'FF148	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 1	MB_1[2].MSG_DATA[1]	8	H'FF149	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 2	MB_1[2].MSG_DATA[2]	8	H'FF14A	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 3	MB_1[2].MSG_DATA[3]	8	H'FF14B	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 4	MB_1[2].MSG_DATA[4]	8	H'FF14C	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 5	MB_1[2].MSG_DATA[5]	8	H'FF14D	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 6	MB_1[2].MSG_DATA[6]	8	H'FF14E	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_data 7	MB_1[2].MSG_DATA[7]	8	H'FF14F	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_control 1H	MB_1[2].CONTROL1H	8	H'FF150	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 2_1_control 1L	MB_1[2].CONTROL1L	8	H'FF151	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_control 0H	MB_1[3].CONTROL0H	16	H'FF160	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_control 0L	MB_1[3].CONTROL0L	16	H'FF162	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_LAFMH	MB_1[3].LAFMH	16	H'FF164	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_LAFML	MB_1[3].LAFML	16	H'FF166	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 0	MB_1[3].MSG_DATA[0]	8	H'FF168	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 1	MB_1[3].MSG_DATA[1]	8	H'FF169	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 2	MB_1[3].MSG_DATA[2]	8	H'FF16A	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 3	MB_1[3].MSG_DATA[3]	8	H'FF16B	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 4	MB_1[3].MSG_DATA[4]	8	H'FF16C	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 5	MB_1[3].MSG_DATA[5]	8	H'FF16D	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 6	MB_1[3].MSG_DATA[6]	8	H'FF16E	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_data 7	MB_1[3].MSG_DATA[7]	8	H'FF16F	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_control 1H	MB_1[3].CONTROL1H	8	H'FF170	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 3_1_control 1L	MB_1[3].CONTROL1L	8	H'FF171	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 4_1_control 0H	MB_1[4].CONTROL0H	16	H'FF180	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 4_1_control 0L	MB_1[4].CONTROL0L	16	H'FF182	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 4_1_LAFMH	MB_1[4].LAFMH	16	H'FF184	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 4_1_LAFML	MB_1[4].LAFML	16	H'FF186	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 4_1_data 0	MB_1[4].MSG_DATA[0]	8	H'FF188	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 4_1_data 1	MB_1[4].MSG_DATA[1]	8	H'FF189	RCAN-ET_1	16	4Pϕ/4Pϕ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*1	Module	Data Width	Access Cycles*4 (Read/Write)
Mailbox 4_1_data 2	MB_1[4].MSG_DATA[2]	8	H'FF18A	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_data 3	MB_1[4].MSG_DATA[3]	8	H'FF18B	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_data 4	MB_1[4].MSG_DATA[4]	8	H'FF18C	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_data 5	MB_1[4].MSG_DATA[5]	8	H'FF18D	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_data 6	MB_1[4].MSG_DATA[6]	8	H'FF18E	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_data 7	MB_1[4].MSG_DATA[7]	8	H'FF18F	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_control 1H	MB_1[4].CONTROL1H	8	H'FF190	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 4_1_control 1L	MB_1[4].CONTROL1L	8	H'FF191	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_control 0H	MB_1[5].CONTROL0H	16	H'FF1A0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_control 0L	MB_1[5].CONTROL0L	16	H'FF1A2	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_LAFMH	MB_1[5].LAFMH	16	H'FF1A4	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_LAFML	MB_1[5].LAFML	16	H'FF1A6	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 0	MB_1[5].MSG_DATA[0]	8	H'FF1A8	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 1	MB_1[5].MSG_DATA[1]	8	H'FF1A9	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 2	MB_1[5].MSG_DATA[2]	8	H'FF1AA	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 3	MB_1[5].MSG_DATA[3]	8	H'FF1AB	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 4	MB_1[5].MSG_DATA[4]	8	H'FF1AC	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 5	MB_1[5].MSG_DATA[5]	8	H'FF1AD	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 6	MB_1[5].MSG_DATA[6]	8	H'FF1AE	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_data 7	MB_1[5].MSG_DATA[7]	8	H'FF1AF	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_control 1H	MB_1[5].CONTROL1H	8	H'FF1B0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 5_1_control 1L	MB_1[5].CONTROL1L	8	H'FF1B1	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_control 0H	MB_1[6].CONTROL0H	16	H'FF1C0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_control 0L	MB_1[6].CONTROL0L	16	H'FF1C2	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_LAFMH	MB_1[6].LAFMH	16	H'FF1C4	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_LAFML	MB_1[6].LAFML	16	H'FF1C6	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 0	MB_1[6].MSG_DATA[0]	8	H'FF1C8	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 1	MB_1[6].MSG_DATA[1]	8	H'FF1C9	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 2	MB_1[6].MSG_DATA[2]	8	H'FF1CA	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 3	MB_1[6].MSG_DATA[3]	8	H'FF1CB	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 4	MB_1[6].MSG_DATA[4]	8	H'FF1CC	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 5	MB_1[6].MSG_DATA[5]	8	H'FF1CD	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 6	MB_1[6].MSG_DATA[6]	8	H'FF1CE	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_data 7	MB_1[6].MSG_DATA[7]	8	H'FF1CF	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_control 1H	MB_1[6].CONTROL1H	8	H'FF1D0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 6_1_control 1L	MB_1[6].CONTROL1L	8	H'FF1D1	RCAN-ET_1	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 7_1_control 0H	MB_1[7].CONTROL0H	16	H'FF1E0	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_control 0L	MB_1[7].CONTROL0L	16	H'FF1E2	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_LAFMH	MB_1[7].LAFMH	16	H'FF1E4	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_LAFML	MB_1[7].LAFML	16	H'FF1E6	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 0	MB_1[7].MSG_DATA[0]	8	H'FF1E8	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 1	MB_1[7].MSG_DATA[1]	8	H'FF1E9	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 2	MB_1[7].MSG_DATA[2]	8	H'FF1EA	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 3	MB_1[7].MSG_DATA[3]	8	H'FF1EB	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 4	MB_1[7].MSG_DATA[4]	8	H'FF1EC	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 5	MB_1[7].MSG_DATA[5]	8	H'FF1ED	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 6	MB_1[7].MSG_DATA[6]	8	H'FF1EE	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_data 7	MB_1[7].MSG_DATA[7]	8	H'FF1EF	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_control 1H	MB_1[7].CONTROL1H	8	H'FF1F0	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 7_1_control 1L	MB_1[7].CONTROL1L	8	H'FF1F1	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_control 0H	MB_1[8].CONTROL0H	16	H'FF200	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_control 0L	MB_1[8].CONTROL0L	16	H'FF202	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_LAFMH	MB_1[8].LAFMH	16	H'FF204	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_LAFML	MB_1[8].LAFML	16	H'FF206	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 0	MB_1[8].MSG_DATA[0]	8	H'FF208	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 1	MB_1[8].MSG_DATA[1]	8	H'FF209	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 2	MB_1[8].MSG_DATA[2]	8	H'FF20A	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 3	MB_1[8].MSG_DATA[3]	8	H'FF20B	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 4	MB_1[8].MSG_DATA[4]	8	H'FF20C	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 5	MB_1[8].MSG_DATA[5]	8	H'FF20D	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 6	MB_1[8].MSG_DATA[6]	8	H'FF20E	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_data 7	MB_1[8].MSG_DATA[7]	8	H'FF20F	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_control 1H	MB_1[8].CONTROL1H	8	H'FF210	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 8_1_control 1L	MB_1[8].CONTROL1L	8	H'FF211	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_control 0H	MB_1[9].CONTROL0H	16	H'FF220	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_control 0L	MB_1[9].CONTROL0L	16	H'FF222	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_LAFMH	MB_1[9].LAFMH	16	H'FF224	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_LAFML	MB_1[9].LAFML	16	H'FF226	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 0	MB_1[9].MSG_DATA[0]	8	H'FF228	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 1	MB_1[9].MSG_DATA[1]	8	H'FF229	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 2	MB_1[9].MSG_DATA[2]	8	H'FF22A	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 3	MB_1[9].MSG_DATA[3]	8	H'FF22B	RCAN-ET_1	16	4Pϕ/4Pϕ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 9_1_data 4	MB_1[9].MSG_DATA[4]	8	H'FF22C	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 5	MB_1[9].MSG_DATA[5]	8	H'FF22D	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 6	MB_1[9].MSG_DATA[6]	8	H'FF22E	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_data 7	MB_1[9].MSG_DATA[7]	8	H'FF22F	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_control 1H	MB_1[9].CONTROL1H	8	H'FF230	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 9_1_control 1L	MB_1[9].CONTROL1L	8	H'FF231	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_control 0H	MB_1[10].CONTROL0H	16	H'FF240	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_control 0L	MB_1[10].CONTROL0L	16	H'FF242	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_LAFMH	MB_1[10].LAFMH	16	H'FF244	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_LAFML	MB_1[10].LAFML	16	H'FF246	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 0	MB_1[10].MSG_DATA[0]	8	H'FF248	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 1	MB_1[10].MSG_DATA[1]	8	H'FF249	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 2	MB_1[10].MSG_DATA[2]	8	H'FF24A	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 3	MB_1[10].MSG_DATA[3]	8	H'FF24B	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 4	MB_1[10].MSG_DATA[4]	8	H'FF24C	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 5	MB_1[10].MSG_DATA[5]	8	H'FF24D	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 6	MB_1[10].MSG_DATA[6]	8	H'FF24E	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_data 7	MB_1[10].MSG_DATA[7]	8	H'FF24F	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_control 1H	MB_1[10].CONTROL1H	8	H'FF250	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 10_1_control 1L	MB_1[10].CONTROL1L	8	H'FF251	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_control 0H	MB_1[11].CONTROL0H	16	H'FF260	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_control 0L	MB_1[11].CONTROL0L	16	H'FF262	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_LAFMH	MB_1[11].LAFMH	16	H'FF264	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_LAFML	MB_1[11].LAFML	16	H'FF266	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 0	MB_1[11].MSG_DATA[0]	8	H'FF268	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 1	MB_1[11].MSG_DATA[1]	8	H'FF269	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 2	MB_1[11].MSG_DATA[2]	8	H'FF26A	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 3	MB_1[11].MSG_DATA[3]	8	H'FF26B	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 4	MB_1[11].MSG_DATA[4]	8	H'FF26C	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 5	MB_1[11].MSG_DATA[5]	8	H'FF26D	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 6	MB_1[11].MSG_DATA[6]	8	H'FF26E	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_data 7	MB_1[11].MSG_DATA[7]	8	H'FF26F	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_control 1H	MB_1[11].CONTROL1H	8	H'FF270	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 11_1_control 1L	MB_1[11].CONTROL1L	8	H'FF271	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 12_1_control 0H	MB_1[12].CONTROL0H	16	H'FF280	RCAN-ET_1	16	4Pϕ/4Pϕ
Mailbox 12_1_control 0L	MB_1[12].CONTROL0L	16	H'FF282	RCAN-ET_1	16	4Pϕ/4Pϕ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Mailbox 12_1_LAFMH	MB_1[12].LAFMH	16	H'FF284	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_LAFML	MB_1[12].LAFML	16	H'FF286	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 0	MB_1[12].MSG_DATA[0]	8	H'FF288	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 1	MB_1[12].MSG_DATA[1]	8	H'FF289	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 2	MB_1[12].MSG_DATA[2]	8	H'FF28A	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 3	MB_1[12].MSG_DATA[3]	8	H'FF28B	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 4	MB_1[12].MSG_DATA[4]	8	H'FF28C	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 5	MB_1[12].MSG_DATA[5]	8	H'FF28D	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 6	MB_1[12].MSG_DATA[6]	8	H'FF28E	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_data 7	MB_1[12].MSG_DATA[7]	8	H'FF28F	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_control 1H	MB_1[12].CONTROL1H	8	H'FF290	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 12_1_control 1L	MB_1[12].CONTROL1L	8	H'FF291	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_control 0H	MB_1[13].CONTROL0H	16	H'FF2A0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_control 0L	MB_1[13].CONTROL0L	16	H'FF2A2	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_LAFMH	MB_1[13].LAFMH	16	H'FF2A4	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_LAFML	MB_1[13].LAFML	16	H'FF2A6	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 0	MB_1[13].MSG_DATA[0]	8	H'FF2A8	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 1	MB_1[13].MSG_DATA[1]	8	H'FF2A9	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 2	MB_1[13].MSG_DATA[2]	8	H'FF2AA	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 3	MB_1[13].MSG_DATA[3]	8	H'FF2AB	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 4	MB_1[13].MSG_DATA[4]	8	H'FF2AC	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 5	MB_1[13].MSG_DATA[5]	8	H'FF2AD	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 6	MB_1[13].MSG_DATA[6]	8	H'FF2AE	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_data 7	MB_1[13].MSG_DATA[7]	8	H'FF2AF	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_control 1H	MB_1[13].CONTROL1H	8	H'FF2B0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 13_1_control 1L	MB_1[13].CONTROL1L	8	H'FF2B1	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_control 0H	MB_1[14].CONTROL0H	16	H'FF2C0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_control 0L	MB_1[14].CONTROL0L	16	H'FF2C2	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_LAFMH	MB_1[14].LAFMH	16	H'FF2C4	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_LAFML	MB_1[14].LAFML	16	H'FF2C6	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 0	MB_1[14].MSG_DATA[0]	8	H'FF2C8	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 1	MB_1[14].MSG_DATA[1]	8	H'FF2C9	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 2	MB_1[14].MSG_DATA[2]	8	H'FF2CA	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 3	MB_1[14].MSG_DATA[3]	8	H'FF2CB	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 4	MB_1[14].MSG_DATA[4]	8	H'FF2CC	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 5	MB_1[14].MSG_DATA[5]	8	H'FF2CD	RCAN-ET_1	16	4Pφ/4Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*1	Module	Data Width	Access Cycles*4 (Read/Write)
Mailbox 14_1_data 6	MB_1[14].MSG_DATA[6]	8	H'FF2CE	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_data 7	MB_1[14].MSG_DATA[7]	8	H'FF2CF	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_control 1H	MB_1[14].CONTROL1H	8	H'FF2D0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 14_1_control 1L	MB_1[14].CONTROL1L	8	H'FF2D1	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_control 0H	MB_1[15].CONTROL0H	16	H'FF2E0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_control 0L	MB_1[15].CONTROL0L	16	H'FF2E2	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_LAFMH	MB_1[15].LAFMH	16	H'FF2E4	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_LAFML	MB_1[15].LAFML	16	H'FF2E6	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 0	MB_1[15].MSG_DATA[0]	8	H'FF2E8	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 1	MB_1[15].MSG_DATA[1]	8	H'FF2E9	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 2	MB_1[15].MSG_DATA[2]	8	H'FF2EA	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 3	MB_1[15].MSG_DATA[3]	8	H'FF2EB	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 4	MB_1[15].MSG_DATA[4]	8	H'FF2EC	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 5	MB_1[15].MSG_DATA[5]	8	H'FF2ED	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 6	MB_1[15].MSG_DATA[6]	8	H'FF2EE	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_data 7	MB_1[15].MSG_DATA[7]	8	H'FF2EF	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_control 1H	MB_1[15].CONTROL1H	8	H'FF2F0	RCAN-ET_1	16	4Pφ/4Pφ
Mailbox 15_1_control 1L	MB_1[15].CONTROL1L	8	H'FF2F1	RCAN-ET_1	16	4Pφ/4Pφ
SS control register H_0	SSCRH_0	8	H'FF600	SSU*5_0	16	3Pφ/3Pφ
SS control register L_0	SSCRL_0	8	H'FF601	SSU*5_0	16	3Pφ/3Pφ
SS mode register_0	SSMR_0	8	H'FF602	SSU*5_0	16	3Pφ/3Pφ
SS enable register_0	SSER_0	8	H'FF603	SSU*5_0	16	3Pφ/3Pφ
SS status register_0	SSSR_0	8	H'FF604	SSU*5_0	16	3Pφ/3Pφ
SS control register 2_0	SSCR2_0	8	H'FF605	SSU*5_0	16	3Pφ/3Pφ
SS transmit data register 0_0	SSTDR0_0	8	H'FF606	SSU*5_0	16	3Pφ/3Pφ
SS transmit data register 1_0	SSTDR1_0	8	H'FF607	SSU*5_0	16	3Pφ/3Pφ
SS transmit data register 2_0	SSTDR2_0	8	H'FF608	SSU*5_0	16	3Pφ/3Pφ
SS transmit data register 3_0	SSTDR3_0	8	H'FF609	SSU*5_0	16	3Pφ/3Pφ
SS receive data register 0_0	SSRDR0_0	8	H'FF60A	SSU*5_0	16	3Pφ/3Pφ
SS receive data register 1_0	SSRDR1_0	8	H'FF60B	SSU*5_0	16	3Pφ/3Pφ
SS receive data register 2_0	SSRDR2_0	8	H'FF60C	SSU*5_0	16	3Pφ/3Pφ
SS receive data register 3_0	SSRDR3_0	8	H'FF60D	SSU*5_0	16	3Pφ/3Pφ
SS control register H_1	SSCRH_1	8	H'FF610	SSU*5_1	16	3Pφ/3Pφ
SS control register L_1	SSCRL_1	8	H'FF611	SSU*5_1	16	3Pφ/3Pφ
SS mode register_1	SSMR_1	8	H'FF612	SSU*5_1	16	3Pφ/3Pφ
SS enable register_1	SSER_1	8	H'FF613	SSU*5_1	16	3Pφ/3Pφ

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
SS status register_1	SSSR_1	8	H'FF614	SSU*[5]_1	16	3Pφ/3Pφ
SS control register 2_1	SSCR2_1	8	H'FF615	SSU*[5]_1	16	3Pφ/3Pφ
SS transmit data register 0_1	SSTDR0_1	8	H'FF616	SSU*[5]_1	16	3Pφ/3Pφ
SS transmit data register 1_1	SSTDR1_1	8	H'FF617	SSU*[5]_1	16	3Pφ/3Pφ
SS transmit data register 2_1	SSTDR2_1	8	H'FF618	SSU*[5]_1	16	3Pφ/3Pφ
SS transmit data register 3_1	SSTDR3_1	8	H'FF619	SSU*[5]_1	16	3Pφ/3Pφ
SS receive data register 0_1	SSRDR0_1	8	H'FF61A	SSU*[5]_1	16	3Pφ/3Pφ
SS receive data register 1_1	SSRDR1_1	8	H'FF61B	SSU*[5]_1	16	3Pφ/3Pφ
SS receive data register 2_1	SSRDR2_1	8	H'FF61C	SSU*[5]_1	16	3Pφ/3Pφ
SS receive data register 3_1	SSRDR3_1	8	H'FF61D	SSU*[5]_1	16	3Pφ/3Pφ
Sound generator control register 1_0	SGCR1_0	8	H'FF640	SDG_0	16	3Pφ/3Pφ
Sound generator control status register_0	SGCSR_0	8	H'FF641	SDG_0	16	3Pφ/3Pφ
Sound generator control register 2_0	SGCR2_0	8	H'FF642	SDG_0	16	3Pφ/3Pφ
Sound generator loudness register_0	SGLR_0	8	H'FF643	SDG_0	16	3Pφ/3Pφ
Sound generator tone frequency register_0	SGTFR_0	8	H'FF644	SDG_0	16	3Pφ/3Pφ
Sound generator reference frequency register_0	SGSFR_0	8	H'FF645	SDG_0	16	3Pφ/3Pφ
Sound generator control register 1_1	SGCR1_1	8	H'FF650	SDG_1	16	3Pφ/3Pφ
Sound generator control status register_1	SGCSR_1	8	H'FF651	SDG_1	16	3Pφ/3Pφ
Sound generator control register 2_1	SGCR2_1	8	H'FF652	SDG_1	16	3Pφ/3Pφ
Sound generator loudness register_1	SGLR_1	8	H'FF653	SDG_1	16	3Pφ/3Pφ
Sound generator tone frequency register_1	SGTFR_1	8	H'FF654	SDG_1	16	3Pφ/3Pφ
Sound generator reference frequency register_1	SGSFR_1	8	H'FF655	SDG_1	16	3Pφ/3Pφ
Sound generator control register 1_2	SGCR1_2	8	H'FF660	SDG_2	16	3Pφ/3Pφ
Sound generator control status register_2	SGCSR_2	8	H'FF661	SDG_2	16	3Pφ/3Pφ
Sound generator control register 2_2	SGCR2_2	8	H'FF662	SDG_2	16	3Pφ/3Pφ
Sound generator loudness register_2	SGLR_2	8	H'FF663	SDG_2	16	3Pφ/3Pφ
Sound generator tone frequency register_2	SGTFR_2	8	H'FF664	SDG_2	16	3Pφ/3Pφ
Sound generator reference frequency register_2	SGSFR_2	8	H'FF665	SDG_2	16	3Pφ/3Pφ
Sound generator control register 1_3	SGCR1_3	8	H'FF670	SDG_3	16	3Pφ/3Pφ
Sound generator control status register_3	SGCSR_3	8	H'FF671	SDG_3	16	3Pφ/3Pφ
Sound generator control register 2_3	SGCR2_3	8	H'FF672	SDG_3	16	3Pφ/3Pφ
Sound generator loudness register_3	SGLR_3	8	H'FF673	SDG_3	16	3Pφ/3Pφ
Sound generator tone frequency register_3	SGTFR_3	8	H'FF674	SDG_3	16	3Pφ/3Pφ

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Sound generator reference frequency register_3	SGSFR_3	8	H'FF675	SDG_3	16	3Pφ/3Pφ
PWM control register_0	PWCR_0	8	H'FF6A0	PWM16*[2]_0	16	3Pφ/3Pφ
PWM output control register_0	PWOCR_0	8	H'FF6A2	PWM16*[2]_0	16	3Pφ/3Pφ
PWM cycle register_0	PWCYR_0	16	H'FF6A6	PWM16*[2]_0	16	3Pφ/3Pφ
PWM buffer register 0_0	PWBFR0_0	16	H'FF6A8	PWM16*[2]_0	16	3Pφ/3Pφ
PWM buffer register 2_0	PWBFR2_0	16	H'FF6AA	PWM16*[2]_0	16	3Pφ/3Pφ
PWM buffer register 1_0	PWBFR1_0	16	H'FF6AC	PWM16*[2]_0	16	3Pφ/3Pφ
PWM buffer register 3_0	PWBFR3_0	16	H'FF6AE	PWM16*[2]_0	16	3Pφ/3Pφ
PWM control register_1	PWCR_1	8	H'FF6B0	PWM16*[2]_1	16	3Pφ/3Pφ
PWM output control register _1	PWOCR_1	8	H'FF6B2	PWM16*[2]_1	16	3Pφ/3Pφ
PWM cycle register_1	PWCYR_1	16	H'FF6B6	PWM16*[2]_1	16	3Pφ/3Pφ
PWM buffer register 0_1	PWBFR0_1	16	H'FF6B8	PWM16*[2]_1	16	3Pφ/3Pφ
PWM buffer register 2_1	PWBFR2_1	16	H'FF6BA	PWM16*[2]_1	16	3Pφ/3Pφ
PWM buffer register 1_1	PWBFR1_1	16	H'FF6BC	PWM16*[2]_1	16	3Pφ/3Pφ
PWM buffer register 3_1	PWBFR3_1	16	H'FF6BE	PWM16*[2]_1	16	3Pφ/3Pφ
PWM control register_2	PWCR_2	8	H'FF6C0	PWM16*[2]_2	16	3Pφ/3Pφ
PWM output control register_2	PWOCR_2	8	H'FF6C2	PWM16*[2]_2	16	3Pφ/3Pφ
PWM cycle register_2	PWCYR_2	16	H'FF6C6	PWM16*[2]_2	16	3Pφ/3Pφ
PWM buffer register 0_2	PWBFR0_2	16	H'FF6C8	PWM16*[2]_2	16	3Pφ/3Pφ
PWM buffer register 2_2	PWBFR2_2	16	H'FF6CA	PWM16*[2]_2	16	3Pφ/3Pφ
PWM buffer register 1_2	PWBFR1_2	16	H'FF6CC	PWM16*[2]_2	16	3Pφ/3Pφ
PWM buffer register 3_2	PWBFR3_2	16	H'FF6CE	PWM16*[2]_2	16	3Pφ/3Pφ
Watch timer control register	WTCR	8	H'FF6D0	WAT	16	3Pφ/3Pφ
Watch timer status register	WTSR	8	H'FF6D2	WAT	16	3Pφ/3Pφ
Watch timer constant register	WTCOR	16	H'FF6D4	WAT	16	3Pφ/3Pφ
Watch timer counter	WTCNT	16	H'FF6D6	WAT	16	3Pφ/3Pφ
PWM control register 1	PWCR1	8	H'FF6E0	PWM10*[3]_1	16	3Pφ/3Pφ
PWM output control register 1	PWOCR1	8	H'FF6E2	PWM10*[3]_1	16	3Pφ/3Pφ
PWM polarity register 1	PWPR1	8	H'FF6E4	PWM10*[3]_1	16	3Pφ/3Pφ
PWM cycle register1	PWCYR1	16	H'FF6E6	PWM10*[3]_1	16	3Pφ/3Pφ
PWM buffer register 1A	PWBFR1A	16	H'FF6E8	PWM10*[3]_1	16	3Pφ/3Pφ
PWM buffer register 1C	PWBFR1C	16	H'FF6EA	PWM10*[3]_1	16	3Pφ/3Pφ
PWM buffer register 1E	PWBFR1E	16	H'FF6EC	PWM10*[3]_1	16	3Pφ/3Pφ
PWM buffer register 1G	PWBFR1G	16	H'FF6EE	PWM10*[3]_1	16	3Pφ/3Pφ
PWM control register 2	PWCR2	8	H'FF6F0	PWM10*[3]_2	16	3Pφ/3Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
PWM output control register 2	PWOCR2	8	H'FF6F2	PWM10*[3]_2	16	3Pφ/3Pφ
PWM polarity register 2	PWPR2	8	H'FF6F4	PWM10*[3]_2	16	3Pφ/3Pφ
PWM cycle register 2	PWCYR2	16	H'FF6F6	PWM10*[3]_2	16	3Pφ/3Pφ
PWM buffer register 2A	PWBFR2A	16	H'FF6F8	PWM10*[3]_2	16	3Pφ/3Pφ
PWM buffer register 2C	PWBFR2C	16	H'FF6FA	PWM10*[3]_2	16	3Pφ/3Pφ
PWM buffer register 2E	PWBFR2E	16	H'FF6FC	PWM10*[3]_2	16	3Pφ/3Pφ
PWM buffer register 2G	PWBFR2G	16	H'FF6FE	PWM10*[3]_2	16	3Pφ/3Pφ
PWM buffer transfer control register	PWBTCR	8	H'FF706	PWM10*[3]	16	3Pφ/3Pφ
D/A data register 0	DADR0	8	H'FF710	D/A	8	3Pφ/3Pφ
D/A data register 1	DADR1	8	H'FF711	D/A	8	3Pφ/3Pφ
D/A control register 01	DACR01	8	H'FF712	D/A	8	3Pφ/3Pφ
RCAN-ET monitor register_0	RCANMON_0	8	H'FF818	RCAN-ET_0	8	3Pφ/3Pφ
RCAN-ET monitor register_1	RCANMON_1	8	H'FF819	RCAN-ET_1	8	3Pφ/3Pφ
A/D data register A_1	ADDRA_1	16	H'FFA90	A/D_1	16	2Pφ/2Pφ
A/D data register B_1	ADDRB_1	16	H'FFA92	A/D_1	16	2Pφ/2Pφ
A/D data register C_1	ADDRC_1	16	H'FFA94	A/D_1	16	2Pφ/2Pφ
A/D data register D_1	ADDRD_1	16	H'FFA96	A/D_1	16	2Pφ/2Pφ
A/D data register E_1	ADDRE_1	16	H'FFA98	A/D_1	16	2Pφ/2Pφ
A/D data register F_1	ADDRF_1	16	H'FFA9A	A/D_1	16	2Pφ/2Pφ
A/D data register G_1	ADDRG_1	16	H'FFA9C	A/D_1	16	2Pφ/2Pφ
A/D data register H_1	ADDRH_1	16	H'FFA9E	A/D_1	16	2Pφ/2Pφ
A/D control/status register_1	ADCSR_1	8	H'FFAA0	A/D_1	16	2Pφ/2Pφ
A/D control register_1	ADCR_1	8	H'FFAA1	A/D_1	16	2Pφ/2Pφ
Port 1 data direction register	P1DDR	8	H'FFB80	I/O port	8	2Pφ/2Pφ
Port 2 data direction register	P2DDR	8	H'FFB81	I/O port	8	2Pφ/2Pφ
Port 3 data direction register	P3DDR	8	H'FFB82	I/O port	8	2Pφ/2Pφ
Port 6 data direction register	P6DDR	8	H'FFB85	I/O port	8	2Pφ/2Pφ
Port A data direction register	PADDR	8	H'FFB89	I/O port	8	2Pφ/2Pφ
Port D data direction register	PDDDR	8	H'FFB8C	I/O port	8	2Pφ/2Pφ
Port E data direction register	PEDDR	8	H'FFB8D	I/O port	8	2Pφ/2Pφ
Port F data direction register	PFDDR	8	H'FFB8E	I/O port	8	2Pφ/2Pφ
Port 1 input buffer control register	P1ICR	8	H'FFB90	I/O port	8	2Pφ/2Pφ
Port 2 input buffer control register	P2ICR	8	H'FFB91	I/O port	8	2Pφ/2Pφ
Port 3 input buffer control register	P3ICR	8	H'FFB92	I/O port	8	2Pφ/2Pφ
Port 4 input buffer control register	P4ICR	8	H'FFB93	I/O port	8	2Pφ/2Pφ
Port 5 input buffer control register	P5ICR	8	H'FFB94	I/O port	8	2Pφ/2Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*¹	Module	Data Width	Access Cycles*⁴ (Read/Write)
Port 6 input buffer control register	P6ICR	8	H'FFB95	I/O port	8	2Pφ/2Pφ
Port A input buffer control register	PAICR	8	H'FFB99	I/O port	8	2Pφ/2Pφ
Port D input buffer control register	PDICR	8	H'FFB9C	I/O port	8	2Pφ/2Pφ
Port E input buffer control register	PEICR	8	H'FFB9D	I/O port	8	2Pφ/2Pφ
Port F input buffer control register	PFICR	8	H'FFB9E	I/O port	8	2Pφ/2Pφ
Port H register	PORTH	8	H'FFBA0	I/O port	8	2Pφ/2Pφ
Port I register	PORTI	8	H'FFBA1	I/O port	8	2Pφ/2Pφ
Port J register	PORTJ	8	H'FFBA2	I/O port	8	2Pφ/2Pφ
Port K register	PORTK	8	H'FFBA3	I/O port	8	2Pφ/2Pφ
Port H data register	PHDR	8	H'FFBA4	I/O port	8	2Pφ/2Pφ
Port I data register	PIDR	8	H'FFBA5	I/O port	8	2Pφ/2Pφ
Port J data register	PJDR	8	H'FFBA6	I/O port	8	2Pφ/2Pφ
Port K data register	PKDR	8	H'FFBA7	I/O port	8	2Pφ/2Pφ
Port H data direction register	PHDDR	8	H'FFBA8	I/O port	8	2Pφ/2Pφ
Port I data direction register	PIDDR	8	H'FFBA9	I/O port	8	2Pφ/2Pφ
Port J data direction register	PJDDR	8	H'FFBAA	I/O port	8	2Pφ/2Pφ
Port K data direction register	PKDDR	8	H'FFBAB	I/O port	8	2Pφ/2Pφ
Port H input buffer control register	PHICR	8	H'FFBAC	I/O port	8	2Pφ/2Pφ
Port I input buffer control register	PIICR	8	H'FFBAD	I/O port	8	2Pφ/2Pφ
Port J input buffer control register	PJICR	8	H'FFBAE	I/O port	8	2Pφ/2Pφ
Port K input buffer control register	PKICR	8	H'FFBAF	I/O port	8	2Pφ/2Pφ
Port D pull-up MOS control register	PDPCR	8	H'FFBB4	I/O port	8	2Pφ/2Pφ
Port E pull-up MOS control register	PEPCR	8	H'FFBB5	I/O port	8	2Pφ/2Pφ
Port F pull-up MOS control register	PFPCR	8	H'FFBB6	I/O port	8	2Pφ/2Pφ
Port H pull-up MOS control register	PHPCR	8	H'FFBB8	I/O port	8	2Pφ/2Pφ
Port I pull-up MOS control register	PIPCR	8	H'FFBB9	I/O port	8	2Pφ/2Pφ
Port J pull-up MOS control register	PJPCR	8	H'FFBBA	I/O port	8	2Pφ/2Pφ
Port K pull-up MOS control register	PKPCR	8	H'FFBBB	I/O port	8	2Pφ/2Pφ
Port 2 open drain control register	P2ODR	8	H'FFBBC	I/O port	8	2Pφ/2Pφ
Port F open drain control register	PFODR	8	H'FFBBD	I/O port	8	2Pφ/2Pφ
Port function control register 2	PFCR2	8	H'FFBC2	I/O port	8	2Pφ/3Pφ
Port function control register 4	PFCR4	8	H'FFBC4	I/O port	8	2Pφ/3Pφ
Port function control register 9	PFCR9	8	H'FFBC9	I/O port	8	2Pφ/3Pφ
Software standby release IRQ enable register	SSIER	16	H'FFBCE	INTC	16	2Iφ/3Iφ
DMA source address register_0	DSAR_0	32	H'FFC00	DMAC_0	16	2Iφ/2Iφ
DMA destination address register_0	DDAR_0	32	H'FFC04	DMAC_0	16	2Iφ/2Iφ

Register Name	Abbreviation	Number of Bits	Address*1	Module	Data Width	Access Cycles*4 (Read/Write)
DMA offset register_0	DOFR_0	32	H'FFC08	DMAC_0	16	2Iφ/2Iφ
DMA transfer count register_0	DTCR_0	32	H'FFC0C	DMAC_0	16	2Iφ/2Iφ
DMA block size register_0	DBSR_0	32	H'FFC10	DMAC_0	16	2Iφ/2Iφ
DMA mode control register_0	DMDR_0	32	H'FFC14	DMAC_0	16	2Iφ/2Iφ
DMA address control register_0	DACR_0	32	H'FFC18	DMAC_0	16	2Iφ/2Iφ
DMA source address register_1	DSAR_1	32	H'FFC20	DMAC_1	16	2Iφ/2Iφ
DMA destination address register_1	DDAR_1	32	H'FFC24	DMAC_1	16	2Iφ/2Iφ
DMA offset register_1	DOFR_1	32	H'FFC28	DMAC_1	16	2Iφ/2Iφ
DMA transfer count register_1	DTCR_1	32	H'FFC2C	DMAC_1	16	2Iφ/2Iφ
DMA block size register_1	DBSR_1	32	H'FFC30	DMAC_1	16	2Iφ/2Iφ
DMA mode control register_1	DMDR_1	32	H'FFC34	DMAC_1	16	2Iφ/2Iφ
DMA address control register_1	DACR_1	32	H'FFC38	DMAC_1	16	2Iφ/2Iφ
DMA source address register_2	DSAR_2	32	H'FFC40	DMAC_2	16	2Iφ/2Iφ
DMA destination address register_2	DDAR_2	32	H'FFC44	DMAC_2	16	2Iφ/2Iφ
DMA offset register_2	DOFR_2	32	H'FFC48	DMAC_2	16	2Iφ/2Iφ
DMA transfer count register_2	DTCR_2	32	H'FFC4C	DMAC_2	16	2Iφ/2Iφ
DMA block size register_2	DBSR_2	32	H'FFC50	DMAC_2	16	2Iφ/2Iφ
DMA mode control register_2	DMDR_2	32	H'FFC54	DMAC_2	16	2Iφ/2Iφ
DMA address control register_2	DACR_2	32	H'FFC58	DMAC_2	16	2Iφ/2Iφ
DMA source address register_3	DSAR_3	32	H'FFC60	DMAC_3	16	2Iφ/2Iφ
DMA destination address register_3	DDAR_3	32	H'FFC64	DMAC_3	16	2Iφ/2Iφ
DMA offset register_3	DOFR_3	32	H'FFC68	DMAC_3	16	2Iφ/2Iφ
DMA transfer count register_3	DTCR_3	32	H'FFC6C	DMAC_3	16	2Iφ/2Iφ
DMA block size register_3	DBSR_3	32	H'FFC70	DMAC_3	16	2Iφ/2Iφ
DMA source address register_3	DMDR_3	32	H'FFC74	DMAC_3	16	2Iφ/2Iφ
DMA destination address register_3	DACR_3	32	H'FFC78	DMAC_3	16	2Iφ/2Iφ
DMA module request register_0	DMRSR_0	8	H'FFD20	DMAC_0	16	2Iφ/2Iφ
DMA module request register_1	DMRSR_1	8	H'FFD21	DMAC_1	16	2Iφ/2Iφ
DMA module request register_2	DMRSR_2	8	H'FFD22	DMAC_2	16	2Iφ/2Iφ
DMA module request register_3	DMRSR_3	8	H'FFD23	DMAC_3	16	2Iφ/2Iφ
Interrupt priority register A	IPRA	16	H'FFD40	INTC	16	2Iφ/3Iφ
Interrupt priority register B	IPRB	16	H'FFD42	INTC	16	2Iφ/3Iφ
Interrupt priority register C	IPRC	16	H'FFD44	INTC	16	2Iφ/3Iφ
Interrupt priority register D	IPRD	16	H'FFD46	INTC	16	2Iφ/3Iφ
Interrupt priority register E	IPRE	16	H'FFD48	INTC	16	2Iφ/3Iφ
Interrupt priority register F	IPRF	16	H'FFD4A	INTC	16	2Iφ/3Iφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Interrupt priority register G	IPRG	16	H'FFD4C	INTC	16	2Iφ/3Iφ
Interrupt priority register I	IPRI	16	H'FFD50	INTC	16	2Iφ/3Iφ
Interrupt priority register K	IPRK	16	H'FFD54	INTC	16	2Iφ/3Iφ
Interrupt priority register L	IPRL	16	H'FFD56	INTC	16	2Iφ/3Iφ
Interrupt priority register O	IPRO	16	H'FFD5C	INTC	16	2Iφ/3Iφ
Interrupt priority register Q	IPRQ	16	H'FFD60	INTC	16	2Iφ/3Iφ
Interrupt priority register R	IPRR	16	H'FFD62	INTC	16	2Iφ/3Iφ
IRQ sense control register H	ISCRH	16	H'FFD68	INTC	16	2Iφ/3Iφ
IRQ sense control register L	ISCRL	16	H'FFD6A	INTC	16	2Iφ/3Iφ
Bus width control register	ABWCR	16	H'FFD84	BSC	16	2Iφ/3Iφ
Access state control register	ASTCR	16	H'FFD86	BSC	16	2Iφ/3Iφ
Wait control register A	WTCRA	16	H'FFD88	BSC	16	2Iφ/3Iφ
Wait control register B	WTCRB	16	H'FFD8A	BSC	16	2Iφ/3Iφ
Read strove timing control register	RDNCR	16	H'FFD8C	BSC	16	2Iφ/3Iφ
Idle control register	IDLCR	16	H'FFD90	BSC	16	2Iφ/3Iφ
Bus control register 1	BCR1	16	H'FFD92	BSC	16	2Iφ/3Iφ
Bus control register 2	BCR2	8	H'FFD94	BSC	16	2Iφ/3Iφ
Endian control register	ENDIANCR	8	H'FFD95	BSC	16	2Iφ/3Iφ
RAM emulation register	RAMER	8	H'FFD9E	BSC	16	2Iφ/3Iφ
Mode control register	MDCR	16	H'FFDC0	SYSTEM	16	2Iφ/3Iφ
System control register	SYSCR	16	H'FFDC2	SYSTEM	16	2Iφ/3Iφ
System clock control register	SCKCR	16	H'FFDC4	SYSTEM	16	2Iφ/3Iφ
Standby control register	SBYCR	16	H'FFDC6	SYSTEM	16	2Iφ/3Iφ
Module stop control register A	MSTPCRA	16	H'FFDC8	SYSTEM	16	2Iφ/3Iφ
Module stop control register B	MSTPCRB	16	H'FFDCA	SYSTEM	16	2Iφ/3Iφ
Module stop control register C	MSTPCRC	16	H'FFDCC	SYSTEM	16	2Iφ/3Iφ
Subclock control register	SUBCKCR	8	H'FFDCF	SYSTEM	16	2Iφ/3Iφ
Flash code control status register	FCCS	8	H'FFDE8	FLASH	8	2Pφ/2Pφ
Flash program code select register	FPCS	8	H'FFDE9	FLASH	8	2Pφ/2Pφ
Flash erase code select register	FECS	8	H'FFDEA	FLASH	8	2Pφ/2Pφ
Flash key code register	FKEY	8	H'FFDEC	FLASH	8	2Pφ/2Pφ
Flash transfer destination address register	FTDAR	8	H'FFDEE	FLASH	8	2Pφ/2Pφ
Serial mode register_4	SMR_4	8	H'FFE90	SCI_4	8	2Pφ/2Pφ
Bit rate register_4	BRR_4	8	H'FFE91	SCI_4	8	2Pφ/2Pφ
Serial control register_4	SCR_4	8	H'FFE92	SCI_4	8	2Pφ/2Pφ
Transmit data register_4	TDR_4	8	H'FFE93	SCI_4	8	2Pφ/2Pφ

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Serial status register_4	SSR_4	8	H'FFE94	SCI_4	8	2Pφ/2Pφ
Receive data register_4	RDR_4	8	H'FFE95	SCI_4	8	2Pφ/2Pφ
Smart card mode register_4	SCMR_4	8	H'FFE96	SCI_4	8	2Pφ/2Pφ
Serial mode register_5	SMR_5	8	H'FFE98	SCI_5	8	2Pφ/2Pφ
Bit rate register_5	BRR_5	8	H'FFE99	SCI_5	8	2Pφ/2Pφ
Serial control register_5	SCR_5	8	H'FFE9A	SCI_5	8	2Pφ/2Pφ
Transmit data register_5	TDR_5	8	H'FFE9B	SCI_5	8	2Pφ/2Pφ
Serial status register_5	SSR_5	8	H'FFE9C	SCI_5	8	2Pφ/2Pφ
Receive data register_5	RDR_5	8	H'FFE9D	SCI_5	8	2Pφ/2Pφ
Smart card mode register_5	SCMR_5	8	H'FFE9E	SCI_5	8	2Pφ/2Pφ
I²C bus control register A_0	ICCRA_0	8	H'FFEB0	IIC2_0	8	2Pφ/2Pφ
I²C bus control register B_0	ICCRB_0	8	H'FFEB1	IIC2_0	8	2Pφ/2Pφ
I²C mode register_0	ICMR_0	8	H'FFEB2	IIC2_0	8	2Pφ/2Pφ
I²C bus interrupt enable register_0	ICIER_0	8	H'FFEB3	IIC2_0	8	2Pφ/2Pφ
I²C bus status register_0	ICSR_0	8	H'FFEB4	IIC2_0	8	2Pφ/2Pφ
Slave address register_0	SAR_0	8	H'FFEB5	IIC2_0	8	2Pφ/2Pφ
I²C bus transmit data register_0	ICDRT_0	8	H'FFEB6	IIC2_0	8	2Pφ/2Pφ
I²C bus receive data register_0	ICDRR_0	8	H'FFEB7	IIC2_0	8	2Pφ/2Pφ
I²C bus control register A_1	ICCRA_1	8	H'FFEB8	IIC2_1	8	2Pφ/2Pφ
I²C bus control register B_1	ICCRB_1	8	H'FFEB9	IIC2_1	8	2Pφ/2Pφ
I²C mode register_1	ICMR_1	8	H'FFEBA	IIC2_1	8	2Pφ/2Pφ
I²C bus interrupt enable register_1	ICIER_1	8	H'FFEBB	IIC2_1	8	2Pφ/2Pφ
I²C bus status register_1	ICSR_1	8	H'FFEBC	IIC2_1	8	2Pφ/2Pφ
Slave address register_1	SAR_1	8	H'FFEBD	IIC2_1	8	2Pφ/2Pφ
I²C bus transmit data register_1	ICDRT_1	8	H'FFEBE	IIC2_1	8	2Pφ/2Pφ
I²C bus receive data register_1	ICDRR_1	8	H'FFEBF	IIC2_1	8	2Pφ/2Pφ
Timer control register_4	TCR_4	8	H'FFEE0	TPU_4	16	2Pφ/2Pφ
Timer mode register_4	TMDR_4	8	H'FFEE1	TPU_4	16	2Pφ/2Pφ
Timer I/O control register_4	TIOR_4	8	H'FFEE2	TPU_4	16	2Pφ/2Pφ
Timer interrupt enable register_4	TIER_4	8	H'FFEE4	TPU_4	16	2Pφ/2Pφ
Timer status register_4	TSR_4	8	H'FFEE5	TPU_4	16	2Pφ/2Pφ
Timer counter_4	TCNT_4	16	H'FFEE6	TPU_4	16	2Pφ/2Pφ
Timer general register A_4	TGRA_4	16	H'FFEE8	TPU_4	16	2Pφ/2Pφ
Timer general register B_4	TGRB_4	16	H'FFEEA	TPU_4	16	2Pφ/2Pφ
Timer control register_5	TCR_5	8	H'FFEF0	TPU_5	16	2Pφ/2Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Timer mode register_5	TMDR_5	8	H'FFEF1	TPU_5	16	2Pφ/2Pφ
Timer I/O control register_5	TIOR_5	8	H'FFEF2	TPU_5	16	2Pφ/2Pφ
Timer interrupt enable register_5	TIER_5	8	H'FFEF4	TPU_5	16	2Pφ/2Pφ
Timer status register_5	TSR_5	8	H'FFEF5	TPU_5	16	2Pφ/2Pφ
Timer counter_5	TCNT_5	16	H'FFEF6	TPU_5	16	2Pφ/2Pφ
Timer general register A_5	TGRA_5	16	H'FFEF8	TPU_5	16	2Pφ/2Pφ
Timer general register B_5	TGRB_5	16	H'FFEFA	TPU_5	16	2Pφ/2Pφ
Interrupt control register	INTCR	8	H'FFF32	INTC	16	2Iφ/3Iφ
CPU priority control register	CPUPCR	8	H'FFF33	INTC	16	2Iφ/3Iφ
IRQ enable register	IER	16	H'FFF34	INTC	16	2Iφ/3Iφ
IRQ status register	ISR	16	H'FFF36	INTC	16	2Iφ/3Iφ
Port 1 register	PORT1	8	H'FFF40	I/O port	8	2Pφ/—
Port 2 register	PORT2	8	H'FFF41	I/O port	8	2Pφ/—
Port 3 register	PORT3	8	H'FFF42	I/O port	8	2Pφ/—
Port 4 register	PORT4	8	H'FFF43	I/O port	8	2Pφ/—
Port 5 register	PORT5	8	H'FFF44	I/O port	8	2Pφ/—
Port 6 register	PORT6	8	H'FFF45	I/O port	8	2Pφ/—
Port A register	PORTA	8	H'FFF49	I/O port	8	2Pφ/—
Port D register	PORTD	8	H'FFF4C	I/O port	8	2Pφ/—
Port E register	PORTE	8	H'FFF4D	I/O port	8	2Pφ/—
Port F register	PORTF	8	H'FFF4E	I/O port	8	2Pφ/—
Port 1 data register	P1DR	8	H'FFF50	I/O port	8	2Pφ/2Pφ
Port 2 data register	P2DR	8	H'FFF51	I/O port	8	2Pφ/2Pφ
Port 3 data register	P3DR	8	H'FFF52	I/O port	8	2Pφ/2Pφ
Port 6 data register	P6DR	8	H'FFF55	I/O port	8	2Pφ/2Pφ
Port A data register	PADR	8	H'FFF59	I/O port	8	2Pφ/2Pφ
Port D data register	PDDR	8	H'FFF5C	I/O port	8	2Pφ/2Pφ
Port E data register	PEDR	8	H'FFF5D	I/O port	8	2Pφ/2Pφ
Port F data register	PFDR	8	H'FFF5E	I/O port	8	2Pφ/2Pφ
Serial mode register_2	SMR_2	8	H'FFF60	SCI_2	8	2Pφ/2Pφ
Bit rate register_2	BRR_2	8	H'FFF61	SCI_2	8	2Pφ/2Pφ
Serial control register_2	SCR_2	8	H'FFF62	SCI_2	8	2Pφ/2Pφ
Transmit data register_2	TDR_2	8	H'FFF63	SCI_2	8	2Pφ/2Pφ
Serial status register_2	SSR_2	8	H'FFF64	SCI_2	8	2Pφ/2Pφ
Receive data register_2	RDR_2	8	H'FFF65	SCI_2	8	2Pφ/2Pφ
Smart card mode register_2	SCMR_2	8	H'FFF66	SCI_2	8	2Pφ/2Pφ

RENESAS

Register Name	Abbreviation	Number of Bits	Address*[1]	Module	Data Width	Access Cycles*[4] (Read/Write)
Serial mode register_0	SMR_0	8	H'FFF80	SCI_0	8	2Pφ/2Pφ
Bit rate register_0	BRR_0	8	H'FFF81	SCI_0	8	2Pφ/2Pφ
Serial control register_0	SCR_0	8	H'FFF82	SCI_0	8	2Pφ/2Pφ
Transmit data register_0	TDR_0	8	H'FFF83	SCI_0	8	2Pφ/2Pφ
Serial status register_0	SSR_0	8	H'FFF84	SCI_0	8	2Pφ/2Pφ
Receive data register_0	RDR_0	8	H'FFF85	SCI_0	8	2Pφ/2Pφ
Smart card mode register_0	SCMR_0	8	H'FFF86	SCI_0	8	2Pφ/2Pφ
A/D data register A_0	ADDRA_0	16	H'FFF90	A/D_0	16	2Pφ/2Pφ
A/D data register B_0	ADDRB_0	16	H'FFF92	A/D_0	16	2Pφ/2Pφ
A/D data register C_0	ADDRC_0	16	H'FFF94	A/D_0	16	2Pφ/2Pφ
A/D data register D_0	ADDRD_0	16	H'FFF96	A/D_0	16	2Pφ/2Pφ
A/D data register E_0	ADDRE_0	16	H'FFF98	A/D_0	16	2Pφ/2Pφ
A/D data register F_0	ADDRF_0	16	H'FFF9A	A/D_0	16	2Pφ/2Pφ
A/D data register G_0	ADDRG_0	16	H'FFF9C	A/D_0	16	2Pφ/2Pφ
A/D data register H_0	ADDRH_0	16	H'FFF9E	A/D_0	16	2Pφ/2Pφ
A/D control/status register_0	ADCSR_0	8	H'FFFA0	A/D_0	16	2Pφ/2Pφ
A/D control register_0	ADCR_0	8	H'FFFA1	A/D_0	16	2Pφ/2Pφ
Timer control/status register	TCSR	8	H'FFFA4	WDT	16	2Pφ/3Pφ
Timer counter	TCNT	8	H'FFFA5	WDT	16	2Pφ/3Pφ
Reset control/status register	RSTCSR	8	H'FFFA7	WDT	16	2Pφ/3Pφ
Timer start register	TSTR	8	H'FFFBC	TPU	16	2Pφ/2Pφ
Timer synchronous register	TSYR	8	H'FFFBD	TPU	16	2Pφ/2Pφ
Timer control register_0	TCR_0	8	H'FFFC0	TPU_0	16	2Pφ/2Pφ
Timer mode register_0	TMDR_0	8	H'FFFC1	TPU_0	16	2Pφ/2Pφ
Timer I/O control register H_0	TIORH_0	8	H'FFFC2	TPU_0	16	2Pφ/2Pφ
Timer I/O control register L_0	TIORL_0	8	H'FFFC3	TPU_0	16	2Pφ/2Pφ
Timer interrupt enable register_0	TIER_0	8	H'FFFC4	TPU_0	16	2Pφ/2Pφ
Timer status register_0	TSR_0	8	H'FFFC5	TPU_0	16	2Pφ/2Pφ
Timer counter_0	TCNT_0	16	H'FFFC6	TPU_0	16	2Pφ/2Pφ
Timer general register A_0	TGRA_0	16	H'FFFC8	TPU_0	16	2Pφ/2Pφ
Timer general register B_0	TGRB_0	16	H'FFFCA	TPU_0	16	2Pφ/2Pφ
Timer general register C_0	TGRC_0	16	H'FFFCC	TPU_0	16	2Pφ/2Pφ
Timer general register D_0	TGRD_0	16	H'FFFCE	TPU_0	16	2Pφ/2Pφ
Timer control register_1	TCR_1	8	H'FFFD0	TPU_1	16	2Pφ/2Pφ
Timer mode register_1	TMDR_1	8	H'FFFD1	TPU_1	16	2Pφ/2Pφ
Timer I/O control register_1	TIOR_1	8	H'FFFD2	TPU_1	16	2Pφ/2Pφ

Register Name	Abbreviation	Number of Bits	Address[1]	Module	Data Width	Access Cycles[4] (Read/Write)
Timer interrupt enable register_1	TIER_1	8	H'FFFD4	TPU_1	16	2Pφ/2Pφ
Timer status register_1	TSR_1	8	H'FFFD5	TPU_1	16	2Pφ/2Pφ
Timer counter_1	TCNT_1	16	H'FFFD6	TPU_1	16	2Pφ/2Pφ
Timer general register A_1	TGRA_1	16	H'FFFD8	TPU_1	16	2Pφ/2Pφ
Timer general register B_1	TGRB_1	16	H'FFFDA	TPU_1	16	2Pφ/2Pφ
Timer control register_2	TCR_2	8	H'FFFE0	TPU_2	16	2Pφ/2Pφ
Timer mode register_2	TMDR_2	8	H'FFFE1	TPU_2	16	2Pφ/2Pφ
Timer I/O control register_2	TIOR_2	8	H'FFFE2	TPU_2	16	2Pφ/2Pφ
Timer interrupt enable register_2	TIER_2	8	H'FFFE4	TPU_2	16	2Pφ/2Pφ
Timer status register_2	TSR_2	8	H'FFFE5	TPU_2	16	2Pφ/2Pφ
Timer counter_2	TCNT_2	16	H'FFFE6	TPU_2	16	2Pφ/2Pφ
Timer general register A_2	TGRA_2	16	H'FFFE8	TPU_2	16	2Pφ/2Pφ
Timer general register B_2	TGRB_2	16	H'FFFEA	TPU_2	16	2Pφ/2Pφ
Timer control register_3	TCR_3	8	H'FFFF0	TPU_3	16	2Pφ/2Pφ
Timer mode register_3	TMDR_3	8	H'FFFF1	TPU_3	16	2Pφ/2Pφ
Timer I/O control register H_3	TIORH_3	8	H'FFFF2	TPU_3	16	2Pφ/2Pφ
Timer I/O control register L_3	TIORL_3	8	H'FFFF3	TPU_3	16	2Pφ/2Pφ
Timer interrupt enable register_3	TIER_3	8	H'FFFF4	TPU_3	16	2Pφ/2Pφ
Timer status register_3	TSR_3	8	H'FFFF5	TPU_3	16	2Pφ/2Pφ
Timer counter_3	TCNT_3	16	H'FFFF6	TPU_3	16	2Pφ/2Pφ
Timer general register A_3	TGRA_3	16	H'FFFF8	TPU_3	16	2Pφ/2Pφ
Timer general register B_3	TGRB_3	16	H'FFFFA	TPU_3	16	2Pφ/2Pφ
Timer general register C_3	TGRC_3	16	H'FFFFC	TPU_3	16	2Pφ/2Pφ
Timer general register D_3	TGRD_3	16	H'FFFFE	TPU_3	16	2Pφ/2Pφ

Notes: 1. The lower 20 bits are indicated.

2. PWM16 is 16-bit PWM.

3. PWM10 is Motor Control PWM.

4. Access to the mailbox in RCAN-ET_0 or RCAN-ET_1 may generate waits of 0 to five Pf cycles.

5. SSU: Synchronous Serial communication Unit

RENESAS

25.2   Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, and 16-bit and 32-bit registers are shown as 2 or 4 lines, respectively.

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MCR_0	MCR15	MCR14	—	—	—	TST2	TST1	TST0	RCAN-ET_0
	MCR7	MCR6	MCR5	—	—	MCR2	MCR1	MCR0	
GSR_0	—	—	—	—	—	—	—	—	
	—	—	GSR5	GSR4	GSR3	GSR2	GSR1	GSR0	
BCR1_0	TSG13	TSG12	TSG11	TSG10	—	TSG22	TSG21	TSG20	
	—	—	SJW1	SJW0	—	—	—	BSP	
BCR0_0	—	—	—	—	—	—	—	—	
	BRP7	BRP6	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	
IRR_0	—	—	IRR13	IRR12	—	—	IRR9	IRR8	
	IRR7	IRR6	IRR5	IRR4	IRR3	IRR2	IRR1	IRR0	
IMR_0	IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	
	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0	
TEC_0	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	
REC_0	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	
TXPR1_0	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
TXPR0_0	TXPR15	TXPR14	TXPR13	TXPR12	TXPR11	TXPR10	TXPR9	TXPR8	
	TXPR7	TXPR6	TXPR5	TXPR4	TXPR3	TXPR2	TXPR1	—	
TXCR0_0	TXCR15	TXCR14	TXCR13	TXCR12	TXCR11	TXCR10	TXCR9	TXCR8	
	TXCR7	TXCR6	TXCR5	TXCR4	TXCR3	TXCR2	TXCR1	—	
TXACK0_0	TXACK15	TXACK14	TXACK13	TXACK12	TXACK11	TXACK10	TXACK9	TXACK8	
	TXACK7	TXACK6	TXACK5	TXACK4	TXACK3	TXACK2	TXACK1	—	
ABACK0_0	ABACK15	ABACK14	ABACK13	ABACK12	ABACK11	ABACK10	ABACK9	ABACK8	
	ABACK7	ABACK6	ABACK5	ABACK4	ABACK3	ABACK2	ABACK1	—	
RXPR0_0	RXPR15	RXPR14	RXPR13	RXPR12	RXPR11	RXPR10	RXPR9	RXPR8	
	RXPR7	RXPR6	RXPR5	RXPR4	RXPR3	RXPR2	RXPR1	RXPR0	
RFPR0_0	RFPR15	RFPR14	RFPR13	RFPR12	RFPR11	RFPR10	RFPR9	RFPR8	
	RFPR7	RFPR6	RFPR5	RFPR4	RFPR3	RFPR2	RFPR1	RFPR0	
MBIMR0_0	MBIMR15	MBIMR14	MBIMR13	MBIMR12	MBIMR11	MBIMR10	MBIMR9	MBIMR8	
	MBIMR7	MBIMR6	MBIMR5	MBIMR4	MBIMR3	MBIMR2	MBIMR1	MBIMR0	
UMSR0_0	UMSR15	UMSR14	UMSR13	UMSR12	UMSR11	UMSR10	UMSR9	UMSR8	
	UMSR7	UMSR6	UMSR5	UMSR4	UMSR3	UMSR2	UMSR1	UMSR0	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MB_0[0]. CONTROL0H	IDE	RTR	—	STDID10	STDID9	STDID8	STDID7	STDID6	RCAN-ET_0
	STDID5	STDID4	STDID3	STDID2	STDID1	STDID0	EXTID17	EXTID16	
MB_0[0]. CONTROL0L	EXTID15	EXTID14	EXTID13	EXTID12	EXTID11	EXTID10	EXTID9	EXTID8	
	EXTID7	EXTID6	EXTID5	EXTID4	EXTID3	EXTID2	EXTID1	EXTID0	
MB_0[0]. LAFMH	IED_LAFM	—	—	STDID_ LAFM10	STDID_ LAFM9	STDID_ LAFM8	STDID_ LAFM7	STDID_ LAFM6	
	STDID_ LAFM5	STDID_ LAFM4	STDID_ LAFM3	STDID_ LAFM2	STDID_ LAFM1	STDID_ LAFM0	EXTID_LAFM17	EXTID_LAFM16	
MB_0[0]. LAFML	EXTID_ LAFM15	EXTID_ LAFM14	EXTID_ LAFM13	EXTID_ LAFM12	EXTID_ LAFM11	EXTID_ LAFM10	EXTID_ LAFM9	EXTID_ LAFM8	
	EXTID_ LAFM7	EXTID_ LAFM6	EXTID_ LAFM5	EXTID_ LAFM4	EXTID_ LAFM3	EXTID_ LAFM2	EXTID_ LAFM1	EXTID_ LAFM0	
MB_0[0]. MSG_DATA[0]	MSG_DATA0								
MB_0[0]. MSG_DATA[1]	MSG_DATA1								
MB_0[0]. MSG_DATA[2]	MSG_DATA2								
MB_0[0]. MSG_DATA[3]	MSG_DATA3								
MB_0[0]. MSG_DATA[4]	MSG_DATA4								
MB_0[0]. MSG_DATA[5]	MSG_DATA5								
MB_0[0]. MSG_DATA[6]	MSG_DATA6								
MB_0[0]. MSG_DATA[7]	MSG_DATA7								
MB_0[0]. CONTROL1H	—	—	NMC	—	—	MBC2	MBC1	MBC0	
MB_0[0]. CONTROL1L	—	—	—	—	DLC3	DLC2	DLC1	DLC0	
MB_0[1]. CONTROL0H	IDE	RTR	—	STDID10	STDID9	STDID8	STDID7	STDID6	
	STDID5	STDID4	STDID3	STDID2	STDID1	STDID0	EXTID17	EXTID16	
MB_0[1]. CONTROL0L	EXTID15	EXTID14	EXTID13	EXTID12	EXTID11	EXTID10	EXTID9	EXTID8	
	EXTID7	EXTID6	EXTID5	EXTID4	EXTID3	EXTID2	EXTID1	EXTID0	
MB_0[1]. LAFMH	IED_LAFM	—	—	STDID_ LAFM10	STDID_ LAFM9	STDID_ LAFM8	STDID_ LAFM7	STDID_ LAFM6	
	STDID_ LAFM5	STDID_ LAFM4	STDID_ LAFM3	STDID_ LAFM2	STDID_ LAFM1	STDID_ LAFM0	EXTID_ LAFM17	EXTID_ LAFM16	
MB_0[1]. LAFML	EXTID_ LAFM15	EXTID_ LAFM14	EXTID_ LAFM13	EXTID_ LAFM12	EXTID_ LAFM11	EXTID_ LAFM10	EXTID_ LAFM9	EXTID_ LAFM8	
	EXTID_ LAFM7	EXTID_ LAFM6	EXTID_ LAFM5	EXTID_ LAFM4	EXTID_ LAFM3	EXTID_ LAFM2	EXTID_ LAFM1	EXTID_ LAFM0	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MB_0[1]. MSG_DATA[0]					MSG_DATA0				RCAN-ET_0
MB_0[1]. MSG_DATA[1]					MSG_DATA1				
MB_0[1]. MSG_DATA[2]					MSG_DATA2				
MB_0[1]. MSG_DATA[3]					MSG_DATA3				
MB_0[1]. MSG_DATA[4]					MSG_DATA4				
MB_0[1]. MSG_DATA[5]					MSG_DATA5				
MB_0[1]. MSG_DATA[6]					MSG_DATA6				
MB_0[1]. MSG_DATA[7]					MSG_DATA7				
MB_0[1]. CONTROL1H	—	—	NMC	ATX	DART	MBC2	MBC1	MBC0	
MB_0[1]. CONTROL1L	—	—	—	—	DLC3	DLC2	DLC1	DLC0	
MB_0[2]. CONTROL0H to MB_0[2]. CONTROL1L	Same as MB_0[1].CONTROL0H to MB_0[1].CONTROL1L bit configuration								
↓	(Repeated)								
MB_0[15]. CONTROL0H to MB_0[15]. CONTROL1L	Same as MB_0[1].CONTROL0H to MB_0[1].CONTROL1L] bit configuration								
MCR_1	RCAN-ET_1: Same as RCAN-ET_0 bit configuration								RCAN-ET_1
↓									
MB_1[15]. CONTROL0H to MB_1[15]. CONTROL1L									

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SSCRH_0	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0	SSU*[1]_0
SSCRL_0	—	SSUMS	SRES	—	—	—	DATS1	DATS0	
SSMR_0	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0	
SSER_0	TE	RE	—	—	TEIE	TIE	RIE	CEIE	
SSSR_0	—	ORER	—	—	TEND	TDRE	RDRF	CE	
SSCR2_0	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—	
SSTDR0_0									
SSTDR1_0									
SSTDR2_0									
SSTDR3_0									
SSRDR0_0									
SSRDR1_0									
SSRDR2_0									
SSRDR3_0									
SSCRH_1	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0	SSU*[1]_1
SSCRL_1	—	SSUMS	SRES	—	—	—	DATS1	DATS0	
SSMR_1	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0	
SSER_1	TE	RE	—	—	TEIE	TIE	RIE	CEIE	
SSSR_1	—	ORER	—	—	TEND	TDRE	RDRF	CE	
SSCR2_1	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—	
SSTDR0_1									
SSTDR1_1									
SSTDR2_1									
SSTDR3_1									
SSRDR0_1									
SSRDR1_1									
SSRDR2_1									
SSRDR3_1									
SGCR1_0	SGST	STPM	SGE	SGCK1	SGCK0	DPF2	DPF1	DPF0	SDG_0
SGCSR_0	SGIE	SGDEF	—	—	—	—	—	—	
SGCR2_0	SGEND	TCHG	—	—	—	—	—	—	
SGLR_0	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0	
SGTFR_0	—	TONE6	TONE5	TONE4	TONE3	TONE2	TONE1	TONE0	
SGSFR_0	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SGCR1_1	SGST	STPM	SGE	SGCK1	SGCK0	DPF2	DPF1	DPF0	SDG_1
SGCSR_1	SGIE	SGDEF	—	—	—	—	—	—	
SGCR2_1	SGEND	TCHG	—	—	—	—	—	—	
SGLR_1	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0	
SGTFR_1	—	TONE6	TONE5	TONE4	TONE3	TONE2	TONE1	TONE0	
SGSFR_1	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0	
SGCR1_2	SGST	STPM	SGE	SGCK1	SGCK0	DPF2	DPF1	DPF0	SDG_2
SGCSR_2	SGIE	SGDEF	—	—	—	—	—	—	
SGCR2_2	SGEND	TCHG	—	—	—	—	—	—	
SGLR_2	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0	
SGTFR_2	—	TONE6	TONE5	TONE4	TONE3	TONE2	TONE1	TONE0	
SGSFR_2	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0	
SGCR1_3	SGST	STPM	SGE	SGCK1	SGCK0	DPF2	DPF1	DPF0	SDG_3
SGCSR_3	SGIE	SGDEF	—	—	—	—	—	—	
SGCR2_3	SGEND	TCHG	—	—	—	—	—	—	
SGLR_3	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0	
SGTFR_3	—	TONE6	TONE5	TONE4	TONE3	TONE2	TONE1	TONE0	
SGSFR_3	SFS7	SFS6	SFS5	SFS4	SFS3	SFS2	SFS1	SFS0	
PWCR_0	—	SMS	IE	CMF	CST	CKS2	CKS1	CKS0	PWM16*[2]_0
PWOCR_0	—	—	—	—	OE3	OE2	OE1	OE0	
PWCYR_0									
PWBFR0_0	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR2_0	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR1_0	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR3_0	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWCR_1	—	SMS	IE	CMF	CST	CKS2	CKS1	CKS0	PWM16*[2]_1
PWOCR_1	—	—	—	—	OE3	OE2	OE1	OE0	
PWCYR_1									

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PWBFR0_1	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	PWM16*[2]_1
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR2_1	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR1_1	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR3_1	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWCR_2	—	SMS	IE	CMF	CST	CKS2	CKS1	CKS0	PWM16*[2]_2
PWOCR_2	—	—	—	—	OE3	OE2	OE1	OE0	
PWCYR_2									
PWBFR0_2	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR2_2	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR1_2	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR3_2	DT15	DT14	DT13	DT12	DT11	DT10	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
WTCR	—	CMT/$\overline{\text{IT}}$	TME	PSS	IE	CKS2	CKS1	CKS0	WAT
WTSR	CMF/OVF	—	—	—	—	—	WTCNT_WF	WTCR_WF	
WTCOR									
WTCNT									
PWCR1	—	—	IE	CMF	CST	CKS2	CKS1	CKS0	PWM10*[3]_1
PWOCR1	OE1H	OE1G	OE1F	OE1E	OE1D	OE1C	OE1B	OE1A	
PWPR1	OPS1H	OPS1G	OPS1F	OPS1E	OPS1D	OPS1C	OPS1B	OPS1A	
PWCYR1	—	—	—	—	—	—			
PWBFR1A	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR1C	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PWBFR1E	—	—	—	OTS	—	—	DT9	DT8	PWM10*[3]_1
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR1G	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWCR2	—	—	IE	CMF	CST	CKS2	CKS1	CKS0	PWM10*[3]_2
PWOCR2	OE2H	OE2G	OE2F	OE2E	OE2D	OE2C	OE2B	OE2A	
PWPR2	OPS2H	OPS2G	OPS2F	OPS2E	OPS2D	OPS2C	OPS2B	OPS2A	
PWCYR2	—	—	—	—	—	—			
PWBFR2A	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR2C	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR2E	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBFR2G	—	—	—	OTS	—	—	DT9	DT8	
	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
PWBTCR	BTC2G	BTC2E	BTC2C	BTC2A	BTC1G	BTC1E	BTC1C	BTC1A	PWM10*[3]
DADR0									D/A
DADR1									
DACR01	DAOE1	DAOE0	DAE	—	—	—	—	—	
RCANMON_0	—	CTxSTP	RCANE	—	—	—	CTxD	CRxD	RCAN-ET_0
RCANMON_1	—	CTxSTP	RCANE	—	—	—	CTxD	CRxD	RCAN-ET_1
ADDRA_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D_1
	AD1	AD0	—	—	—	—	—	—	
ADDRB_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRC_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRD_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRE_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRF_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
ADDRG_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D_1
	AD1	AD0	—	—	—	—	—	—	
ADDRH_1	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADCSR_1	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0	
ADCR_1	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—	
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	I/O port
P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P6DDR	P67DDR	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	
PADDR	PA7DDR	PA6DDR	PA5DDR	PA4DDR	PA3DDR	PA2DDR	PA1DDR	PA0DDR	
PDDDR	PD7DDR	PD6DDR	PD5DDR	PD4DDR	PD3DDR	PD2DDR	PD1DDR	PD0DDR	
PEDDR	PE7DDR	PE6DDR	PE5DDR	PE4DDR	PE3DDR	PE2DDR	PE1DDR	PE0DDR	
PFDDR	PF7DDR	PF6DDR	PF5DDR	PF4DDR	PF3DDR	PF2DDR	PF1DDR	PF0DDR	
P1ICR	P17ICR	P16ICR	P15ICR	P14ICR	P13ICR	P12ICR	P11ICR	P10ICR	
P2ICR	P27ICR	P26ICR	P25ICR	P24ICR	P23ICR	P22ICR	P21ICR	P20ICR	
P3ICR	P37ICR	P36ICR	P35ICR	P34ICR	P33ICR	P32ICR	P31ICR	P30ICR	
P4ICR	P47ICR	P46ICR	P45ICR	P44ICR	P43ICR	P42ICR	P41ICR	P40ICR	
P5ICR	P57ICR	P56ICR	P55ICR	P54ICR	P53ICR	P52ICR	P51ICR	P50ICR	
P6ICR	P67ICR	P66ICR	P65ICR	P64ICR	P63ICR	P62ICR	P61ICR	P60ICR	
PAICR	PA7ICR	PA6ICR	PA5ICR	PA4ICR	PA3ICR	PA2ICR	PA1ICR	PA0ICR	
PDICR	PD7ICR	PD6ICR	PD5ICR	PD4ICR	PD3ICR	PD2ICR	PD1ICR	PD0ICR	
PEICR	PE7ICR	PE6ICR	PE5ICR	PE4ICR	PE3ICR	PE2ICR	PE1ICR	PE0ICR	
PFICR	PF7ICR	PF6ICR	PF5ICR	PF4ICR	PF3ICR	PF2ICR	PF1ICR	PF0ICR	
PORTH	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0	
PORTI	PI7	PI6	PI5	PI4	PI3	PI2	PI1	PI0	
PORTJ	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0	
PORTK	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0	
PHDR	PH7DR	PH6DR	PH5DR	PH4DR	PH3DR	PH2DR	PH1DR	PH0DR	
PIDR	PI7DR	PI6DR	PI5DR	PI4DR	PI3DR	PI2DR	PI1DR	PI0DR	
PJDR	PJ7DR	PJ6DR	PJ5DR	PJ4DR	PJ3DR	PJ2DR	PJ1DR	PJ0DR	
PKDR	PK7DR	PK6DR	PK5DR	PK4DR	PK3DR	PK2DR	PK1DR	PK0DR	
PHDDR	PH7DDR	PH6DDR	PH5DDR	PH4DDR	PH3DDR	PH2DDR	PH1DDR	PH0DDR	
PIDDR	PIDDR	PI6DDR	PI5DDR	PI4DDR	PI3DDR	PI2DDR	PI1DDR	PI0DDR	
PJDDR	PJ7DDR	PJ6DDR	PJ5DDR	PJ4DDR	PJ3DDR	PJ2DDR	PJ1DDR	PJ0DDR	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PKDDR	PK7DDR	PK6DDR	PK5DDR	PK4DDR	PK3DDR	PK2DDR	PK1DDR	PK0DDR	I/O port
PHICR	PH7ICR	PH6ICR	PH5ICR	PH4ICR	PH3ICR	PH2ICR	PH1ICR	PH0ICR	
PIICR	PI7ICR	PI6ICR	PI5ICR	PI4ICR	PI3ICR	PI2ICR	PI1ICR	PI0ICR	
PJICR	PJ7ICR	PJ6ICR	PJ5ICR	PJ4ICR	PJ3ICR	PJ2ICR	PJ1ICR	PJ0ICR	
PKICR	PK7ICR	PK6ICR	PK5ICR	PK4ICR	PK3ICR	PK2ICR	PK1ICR	PK0ICR	
PDPCR	PD7PCR	PD6PCR	PD5PCR	PD4PCR	PD3PCR	PD2PCR	PD1PCR	PD0PCR	
PEPCR	PE7PCR	PE6PCR	PE5PCR	PE4PCR	PE3PCR	PE2PCR	PE1PCR	PE0PCR	
PFPCR	PF7PCR	PF6PCR	PF5PCR	PF4PCR	PF3PCR	PF2PCR	PF1PCR	PF0PCR	
PHPCR	PH7PCR	PH6PCR	PH5PCR	PH4PCR	PH3PCR	PH2PCR	PH1PCR	PH0PCR	
PIPCR	PI7PCR	PI6PCR	PI5PCR	PI4PCR	PI3PCR	PI2PCR	PI1PCR	PI0PCR	
PJPCR	PJ7PCR	PJ6PCR	PJ5PCR	PJ4PCR	PJ3PCR	PJ2PCR	PJ1PCR	PJ0PCR	
PKPCR	PK7PCR	PK6PCR	PK5PCR	PK4PCR	PK3PCR	PK2PCR	PK1PCR	PK0PCR	
P2ODR	P27ODR	P26ODR	P25ODR	P24ODR	P23ODR	P22ODR	P21ODR	P20ODR	
PFODR	PF7ODR	PF6ODR	PF5ODR	PF4ODR	PF3ODR	PF2ODR	PF1ODR	PF0ODR	
PFCR2	—	—	—	—	—	—	ASOE	—	
PFCR4	A23E	A22E	A21E	A20E	A19E	A18E	A17E	A16E	
PFCR9	TPUMS5	TPUMS4	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B	
SSIER	SSI15	SSI14	SSI13	SSI12	SSI11	SSI10	SSI9	SSI8	INTC
	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0	
DSAR_0									DMAC_0
DDAR_0									
DOFR_0									
DTCR_0									

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DBSR_0	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	DMAC_0
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_0	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_0	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_1									DMAC_1
DDAR_1									
DOFR_1									
DTCR_1									
DBSR_1	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_1	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DACR_1	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	DMAC_1
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_2									DMAC_2
DDAR_2									
DOFR_2									
DTCR_2									
DBSR_2	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_2	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_2	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_3									DMAC_3

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DDAR_3									DMAC_3
DOFR_3									
DTCR_3									
DBSR_3	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_3	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	
DACR_3	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DMRSR_0									DMAC_0
DMRSR_1									DMAC_1
DMRSR_2									DMAC_2
DMRSR_3									DMAC_3
IPRA	—	IPRA14	IPRA13	IPRA12	—	IPRA10	IPRA9	IPRA8	INTC
	—	IPRA6	IPRA5	IPRA4	—	IPRA2	IPRA1	IPRA0	
IPRB	—	IPRB14	IPRB13	IPRB12	—	IPRB10	IPRB9	IPRB8	
	—	IPRB6	IPRB5	IPRB4	—	IPRB2	IPRB1	IPRB0	
IPRC	—	IPRC14	IPRC13	IPRC12	—	IPRC10	IPRC9	IPRC8	
	—	IPRC6	IPRC5	IPRC4	—	IPRC2	IPRC1	IPRC0	
IPRD	—	IPRD14	IPRD13	IPRD12	—	IPRD10	IPRD9	IPRD8	
	—	IPRD6	IPRD5	IPRD4	—	IPRD2	IPRD1	IPRD0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
IPRE	—	IPRE14	IPRE13	IPRE12	—	IPRE10	IPRE9	IPRE8	INTC
	—	IPRE6	IPRE5	IPRE4	—	IPRE2	IPRE1	IPRE0	
IPRF	—	IPRF14	IPRF13	IPRF12	—	IPRF10	IPRF9	IPRF8	
	—	IPRF6	IPRF5	IPRF4	—	IPRF2	IPRF1	IPRF0	
IPRG	—	IPRG14	IPRG13	IPRG12	—	IPRG10	IPRG9	IPRG8	
	—	IPRG6	IPRG5	IPRG4	—	IPRG2	IPRG1	IPRG0	
IPRI	—	IPRI14	IPRI13	IPRI12	—	IPRI10	IPRI9	IPRI8	
	—	IPRI6	IPRI5	IPRI4	—	IPRI2	IPRI1	IPRI0	
IPRK	—	IPRK14	IPRK13	IPRK12	—	IPRK10	IPRK9	IPRK8	
	—	IPRK6	IPRK5	IPRK4	—	IPRK2	IPRK1	IPRK0	
IPRL	—	IPRL14	IPRL13	IPRL12	—	IPRL10	IPRL9	IPRL8	
	—	IPRL6	IPRL5	IPRL4	—	IPRL2	IPRL1	IPRL0	
IPRO	—	IPRO14	IPRO13	IPRO12	—	IPRO10	IPRO9	IPRO8	
	—	IPRO6	IPRO5	IPRO4	—	IPRO2	IPRO1	IPRO0	
IPRQ	—	IPRQ14	IPRQ13	IPRQ12	—	IPRQ10	IPRQ9	IPRQ8	
	—	IPRQ6	IPRQ5	IPRQ4	—	IPRQ2	IPRQ1	IPRQ0	
IPRR	—	IPRR14	IPRR13	IPRR12	—	IPRR10	IPRR9	IPRR8	
	—	IPRR6	IPRR5	IPRR4	—	IPRR2	IPRR1	IPRR0	
ISCRH	IRQ15SR	IRQ15SF	IRQ14SR	IRQ14SF	IRQ13SR	IRQ13SF	IRQ12SR	IRQ12SF	
	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF	
ISCRL	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF	
	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF	
ABWCR	ABWH7	ABWH6	ABWH5	ABWH4	ABWH3	ABWH2	ABWH1	ABWH0	BSC
	ABWL7	ABWL6	ABWL5	ABWL4	ABWL3	ABWL2	ABWL1	ABWL0	
ASTCR	AST7	AST6	AST5	AST4	AST3	AST2	AST1	AST0	
	—	—	—	—	—	—	—	—	
WTCRA	—	W72	W71	W70	—	W62	W61	W60	
	—	W52	W51	W50	—	W42	W41	W40	
WTCRB	—	W32	W31	W30	—	W22	W21	W20	
	—	W12	W11	W10	—	W02	W01	W00	
RDNCR	RDN7	RDN6	RDN5	RDN4	RDN3	RDN2	RDN1	RDN0	
	—	—	—	—	—	—	—	—	
IDLCR	IDLS3	IDLS2	IDLS1	IDLS0	IDLCB1	IDLCB0	IDLCA1	IDLCA0	
	IDLSEL7	IDLSEL6	IDLSEL5	IDLSEL4	IDLSEL3	IDLSEL2	IDLSEL1	IDLSEL0	
BCR1	—	—	—	—	—	—	WDBE	—	
	DKC	—	—	—	—	—	—	—	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
BCR2	—	—	—	IBCCS	—	—	—	PWDBE	BSC
ENDIANCR	LE7	LE6	LE5	LE4	LE3	LE2	—	—	
RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0	
MDCR	—	—	—	—	MDS3	MDS2	MDS1	MDS0	SYSTEM
	—	—	—	—	—	—	—	—	
SYSCR	—	—	MACS	—	FETCHMD	—	EXPE	RAME	
	—	—	—	—	—	—	—	—	
SCKCR	PSTOP1	—	POSEL1	—	—	ICK2	ICK1	ICK0	
	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0	
SBYCR	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0	
	—	—	—	—	—	—	—	—	
MSTPCRA	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8	
	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0	
MSTPCRB	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8	
	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0	
MSTPCRC	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8	
	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0	
SUBCKCR	—	—	—	—	XTALSTP	PLLSTP	WKCKSEL	SUBCKSEL	
FCCS	—	—	—	FLER	—	—	—	SCO	FLASH
FPCS	—	—	—	—	—	—	—	PPVS	
FECS	—	—	—	—	—	—	—	EPVB	
FKEY	K7	K6	K5	K4	K3	K2	K1	K0	
FTDAR	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0	
SMR_4*[4]	C/$\overline{\text{A}}$ (GM)	CHR (BLK)	PE	O/$\overline{\text{E}}$	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_4
BRR_4									
SCR_4*[4]	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_4									
SSR_4*[4]	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_4									
SCMR_4	—	—	—	—	SDIR	SINV	—	SMIF	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SMR_5*[4]	C/$\overline{\text{A}}$ (GM)	CHR (BLK)	PE	O/$\overline{\text{E}}$	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_5
BRR_5									
SCR_5*[4]	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_5									
SSR_5*[4]	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_5									
SCMR_5	—	—	—	—	SDIR	SINV	—	SMIF	
ICCRA_0	ICE	RCVD	MST	TRS	CKS3	CKS2	CKS1	CKS0	IIC2_0
ICCRB_0	BBSY	SCP	SDAO	—	SCLO	—	IICRST	—	
ICMR_0	—	WAIT	—	—	BCWP	BC2	BC1	BC0	
ICIER_0	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR	ACKBT	
ICSR_0	TDRE	TEND	RDRF	NACKF	STOP	AL	AAS	ADZ	
SAR_0	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	—	
ICDRT_0									
ICDRR_0									
ICCRA_1	ICE	RCVD	MST	TRS	CKS3	CKS2	CKS1	CKS0	IIC2_1
ICCRB_1	BBSY	SCP	SDAO	—	SCLO	—	IICRST	—	
ICMR_1	—	WAIT	—	—	BCWP	BC2	BC1	BC0	
ICIER_1	TIE	TEIE	RIE	NAKIE	STIE	ACKE	ACKBR	ACKBT	
ICSR_1	TDRE	TEND	RDRF	NACKF	STOP	AL	AAS	ADZ	
SAR_1	SVA6	SVA5	SVA4	SVA3	SVA2	SVA1	SVA0	—	
ICDRT_1									
ICDRR_1									
TCR_4	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_4
TMDR_4	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_4	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_4	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_4	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_4									
TGRA_4									
TGRB_4									

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_5	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_5
TMDR_5	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_5	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_5	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_5	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_5									
TGRA_5									
TGRB_5									
INTCR	—	—	INTM1	INTM0	NMIEG	—	—	—	INTC
CPUPCR	CPUPCE	—	—	—	IPSETE	CPUP2	CPUP1	CPUP0	
IER	IRQ15E	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E	
	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	
ISR	IRQ15F	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F	
	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
PORT1	P17	P16	P15	P14	P13	P12	P11	P10	I/O port
PORT2	P27	P26	P25	P24	P23	P22	P21	P20	
PORT3	P37	P36	P35	P34	P33	P32	P31	P30	
PORT4	P47	P46	P45	P44	P43	P42	P41	P40	
PORT5	P57	P56	P55	P54	P53	P52	P51	P50	
PORT6	P67	P66	P65	P64	P63	P62	P61	P60	
PORTA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0	
PORTD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	
PORTE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0	
PORTF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	P27DR	P26DR	P25DR	P14DR	P23DR	P22DR	P21DR	P20DR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P6DR	P67DR	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
PADR	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR	
PDDR	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR	
PEDR	PE7DR	PE6DR	PE5DR	PE4DR	PE3DR	PE2DR	PE1DR	PE0DR	
PFDR	PF7DR	PF6DR	PF5DR	PF4DR	PF3DR	PF2DR	PF1DR	PF0DR	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SMR_2*[4]	C/$\overline{\text{A}}$ (GM)	CHR (BLK)	PE	O/$\overline{\text{E}}$	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_2
BRR_2									
SCR_2*[4]	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_2									
SSR_2*[4]	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_2									
SCMR_2	—	—	—	—	SDIR	SINV	—	SMIF	
SMR_0*[4]	C/$\overline{\text{A}}$ (GM)	CHR (BLK)	PE	O/$\overline{\text{E}}$	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_0
BRR_0									
SCR_0*[4]	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_0									
SSR_0*[4]	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_0									
SCMR_0	—	—	—	—	SDIR	SINV	—	SMIF	
ADDRA_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D_0
	AD1	AD0	—	—	—	—	—	—	
ADDRB_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRC_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRD_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRE_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRF_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRG_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADDRH_0	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	
	AD1	AD0	—	—	—	—	—	—	
ADCSR_0	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0	
ADCR_0	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—	

RENESAS

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCSR	OVF	WT/$\overline{\text{IT}}$	TME	—	—	CKS2	CKS1	CKS0	WDT
TCNT									
RSTCSR	WOVF	RSTE	—	—	—	—	—	—	
TSTR	—	—	CST5	CST4	CST3	CST2	CST1	CST0	TPU
TSYR	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0	
TCR_0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_0
TMDR_0	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_0	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_0	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_0	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_0	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_0									
TGRA_0									
TGRB_0									
TGRC_0									
TGRD_0									
TCR_1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_1
TMDR_1	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_1	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_1	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_1	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_1									
TGRA_1									
TGRB_1									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_2
TMDR_2	—	—	—	—	MD3	MD2	MD1	MD0	
TIOR_2	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_2	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_2	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_2									
TGRA_2									
TGRB_2									
TCR_3	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_3
TMDR_3	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_3	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_3	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_3	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_3	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_3									
TGRA_3									
TGRB_3									
TGRC_3									
TGRD_3									

Notes: 1. SSU: Synchronous Serial communication Unit
2. PWM16 is 16-bit PWM.
3. PWM10 is Motor Control PWM.
4. Parts of the bit functions differ in normal mode and the smart card interface mode.

RENESAS

25.3   Register States in Each Operating Mode

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
MCR_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	RCAN-ET_0
GSR_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
BCR1_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
BCR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
IRR_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
IMR_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
TEC_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
REC_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
TXPR1_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
TXPR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
TXCR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
TXACK0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
ABACK0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
RXPR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
RFPR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
MBIMR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
UMSR0_0	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
MB_0[0]. CONTROL0H	—	—	—	—	—	—	—	
MB_0[0]. CONTROL0L	—	—	—	—	—	—	—	
MB_0[0]. LAFMH	—	—	—	—	—	—	—	
MB_0[0]. LAFML	—	—	—	—	—	—	—	
MB_0[0]. MSG_DATA[0]	—	—	—	—	—	—	—	
MB_0[0]. MSG_DATA[1]	—	—	—	—	—	—	—	
MB_0[0]. MSG_DATA[2]	—	—	—	—	—	—	—	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
MB_0[0]. MSG_DATA[3]	—	—	—	—	—	—	—	RCAN-ET_0
MB_0[0]. MSG_DATA[4]	—	—	—	—	—	—	—	
MB_0[0]. MSG_DATA[5]	—	—	—	—	—	—	—	
MB_0[0]. MSG_DATA[6]	—	—	—	—	—	—	—	
MB_0[0]. MSG_DATA[7]	—	—	—	—	—	—	—	
MB_0[0]. CONTROL1H	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
MB_0[0]. CONTROL1L	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
MB_0[1]. CONTROL0H	—	—	—	—	—	—	—	
MB_0[1]. CONTROL0L	—	—	—	—	—	—	—	
MB_0[1]. LAFMH	—	—	—	—	—	—	—	
MB_0[1]. LAFML	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[0]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[1]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[2]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[3]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[4]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[5]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[6]	—	—	—	—	—	—	—	
MB_0[1].MSG_DATA[7]	—	—	—	—	—	—	—	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
MB_0[1].CONTROL1H	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	RCAN-ET_0
MB_0[1].CONTROL1L	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
MB_0[2].CONTROL0H to MB_0[2].CONTROL1L	MB_0[1].CONTROL0H to MB_0 [1].CONTROL1L							
↓	(Repeat)							
MB_0[15].CONTROL0H to MB_0[15].CONTROL1L	MB_0 [1].CONTROL0H to MB_0 [1].CONTROL1L							
MCR_1	RCAN-ET_1 is RCAN-ET_0							RCAN-ET_1
↓								
MB_1[15].CONTROL0H to MB_1[15].CONTROL1L								
SSCRH_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SSU*[1]_0
SSCRL_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSMR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSER_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSSR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSCR2_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR0_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR1_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR2_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR3_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR0_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR1_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR2_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR3_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSCRH_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SSU*[1]_1
SSCRL_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSMR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSER_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
SSSR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SSU*[1]_1
SSCR2_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR0_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR1_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR2_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSTDR3_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR0_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR1_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR2_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSRDR3_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR1_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SDG_0
SGCSR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR2_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGLR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGTFR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGSFR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR1_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SDG_1
SGCSR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR2_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGLR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGTFR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGSFR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR1_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SDG_2
SGCSR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR2_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGLR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGTFR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGSFR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR1_3	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	SDG_3
SGCSR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGCR2_3	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGLR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGTFR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SGSFR_3	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
PWCR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM16*²_0
PWOCR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCYR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR0_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR1_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR3_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM16*²_1
PWOCR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCYR_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR0_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR1_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR3_1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM16*²_2
PWOCR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCYR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR0_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM16*²_2
PWBFR1_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR3_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
WTCR	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	WAT
WTSR	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
WTCOR	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
WTCNT	Initialized	—	—	Initialized	Initialized	Initialized	Initialized	
PWCR1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM10*³_1
PWOCR1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWPR1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCYR1	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR1A	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR1C	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR1E	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR1G	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
PWCR2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM10*³_2
PWOCR2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWPR2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWCYR2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2A	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2C	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2E	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBFR2G	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
PWBTCR	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	PWM10*³
DADR0	Initialized	—	—	—	—	—	Initialized	D/A
DADR1	Initialized	—	—	—	—	—	Initialized	
DACR01	Initialized	—	—	—	—	—	Initialized	
RCANMON_0	Initialized	—	—	—	—	—	Initialized	RCAN-ET_0
RCANMON_1	Initialized	—	—	—	—	—	Initialized	RCAN-ET_1
ADDRA_1	Initialized	—	—	—	—	—	Initialized	A/D_1
ADDRB_1	Initialized	—	—	—	—	—	Initialized	
ADDRC_1	Initialized	—	—	—	—	—	Initialized	
ADDRD_1	Initialized	—	—	—	—	—	Initialized	
ADDRE_1	Initialized	—	—	—	—	—	Initialized	
ADDRF_1	Initialized	—	—	—	—	—	Initialized	
ADDRG_1	Initialized	—	—	—	—	—	Initialized	
ADDRH_1	Initialized	—	—	—	—	—	Initialized	
ADCSR_1	Initialized	—	—	—	—	—	Initialized	
ADCR_1	Initialized	—	—	—	—	—	Initialized	
P1DDR	Initialized	—	—	—	—	—	Initialized	I/O port
P2DDR	Initialized	—	—	—	—	—	Initialized	
P3DDR	Initialized	—	—	—	—	—	Initialized	
P6DDR	Initialized	—	—	—	—	—	Initialized	
PADDR	Initialized	—	—	—	—	—	Initialized	
PDDDR	Initialized	—	—	—	—	—	Initialized	
PEDDR	Initialized	—	—	—	—	—	Initialized	
PFDDR	Initialized	—	—	—	—	—	Initialized	
P1ICR	Initialized	—	—	—	—	—	Initialized	
P2ICR	Initialized	—	—	—	—	—	Initialized	
P3ICR	Initialized	—	—	—	—	—	Initialized	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
P4ICR	Initialized	—	—	—	—	—	Initialized	I/O port
P5ICR	Initialized	—	—	—	—	—	Initialized	
P6ICR	Initialized	—	—	—	—	—	Initialized	
PAICR	Initialized	—	—	—	—	—	Initialized	
PDICR	Initialized	—	—	—	—	—	Initialized	
PEICR	Initialized	—	—	—	—	—	Initialized	
PFICR	Initialized	—	—	—	—	—	Initialized	
PORTH	—	—	—	—	—	—	—	
PORTI	—	—	—	—	—	—	—	
PORTJ	—	—	—	—	—	—	—	
PORTK	—	—	—	—	—	—	—	
PHDR	Initialized	—	—	—	—	—	Initialized	
PIDR	Initialized	—	—	—	—	—	Initialized	
PJDR	Initialized	—	—	—	—	—	Initialized	
PKDR	Initialized	—	—	—	—	—	Initialized	
PHDDR	Initialized	—	—	—	—	—	Initialized	
PIDDR	Initialized	—	—	—	—	—	Initialized	
PJDDR	Initialized	—	—	—	—	—	Initialized	
PKDDR	Initialized	—	—	—	—	—	Initialized	
PHICR	Initialized	—	—	—	—	—	Initialized	
PIICR	Initialized	—	—	—	—	—	Initialized	
PJICR	Initialized	—	—	—	—	—	Initialized	
PKICR	Initialized	—	—	—	—	—	Initialized	
PDPCR	Initialized	—	—	—	—	—	Initialized	
PEPCR	Initialized	—	—	—	—	—	Initialized	
PFPCR	Initialized	—	—	—	—	—	Initialized	
PHPCR	Initialized	—	—	—	—	—	Initialized	
PIPCR	Initialized	—	—	—	—	—	Initialized	
PJPCR	Initialized	—	—	—	—	—	Initialized	
PKPCR	Initialized	—	—	—	—	—	Initialized	
P2ODR	Initialized	—	—	—	—	—	Initialized	
PFODR	Initialized	—	—	—	—	—	Initialized	
PFCR2	Initialized	—	—	—	—	—	Initialized	
PFCR4	Initialized	—	—	—	—	—	Initialized	
PFCR9	Initialized	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
SSIER	Initialized	—	—	—	—	—	Initialized	INTC
DSAR_0	Initialized	—	—	—	—	—	Initialized	DMAC_0
DDAR_0	Initialized	—	—	—	—	—	Initialized	
DOFR_0	Initialized	—	—	—	—	—	Initialized	
DTCR_0	Initialized	—	—	—	—	—	Initialized	
DBSR_0	Initialized	—	—	—	—	—	Initialized	
DMDR_0	Initialized	—	—	—	—	—	Initialized	
DACR_0	Initialized	—	—	—	—	—	Initialized	
DSAR_1	Initialized	—	—	—	—	—	Initialized	DMAC_1
DDAR_1	Initialized	—	—	—	—	—	Initialized	
DOFR_1	Initialized	—	—	—	—	—	Initialized	
DTCR_1	Initialized	—	—	—	—	—	Initialized	
DBSR_1	Initialized	—	—	—	—	—	Initialized	
DMDR_1	Initialized	—	—	—	—	—	Initialized	
DACR_1	Initialized	—	—	—	—	—	Initialized	
DSAR_2	Initialized	—	—	—	—	—	Initialized	DMAC_2
DDAR_2	Initialized	—	—	—	—	—	Initialized	
DOFR_2	Initialized	—	—	—	—	—	Initialized	
DTCR_2	Initialized	—	—	—	—	—	Initialized	
DBSR_2	Initialized	—	—	—	—	—	Initialized	
DMDR_2	Initialized	—	—	—	—	—	Initialized	
DACR_2	Initialized	—	—	—	—	—	Initialized	
DSAR_3	Initialized	—	—	—	—	—	Initialized	DMAC_3
DDAR_3	Initialized	—	—	—	—	—	Initialized	
DOFR_3	Initialized	—	—	—	—	—	Initialized	
DTCR_3	Initialized	—	—	—	—	—	Initialized	
DBSR_3	Initialized	—	—	—	—	—	Initialized	
DMDR_3	Initialized	—	—	—	—	—	Initialized	
DACR_3	Initialized	—	—	—	—	—	Initialized	
DMRSR_0	Initialized	—	—	—	—	—	Initialized	DMAC_0
DMRSR_1	Initialized	—	—	—	—	—	Initialized	DMAC_1
DMRSR_2	Initialized	—	—	—	—	—	Initialized	DMAC_2
DMRSR_3	Initialized	—	—	—	—	—	Initialized	DMAC_3

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
IPRA	Initialized	—	—	—	—	—	Initialized	INTC
IPRB	Initialized	—	—	—	—	—	Initialized	
IPRC	Initialized	—	—	—	—	—	Initialized	
IPRD	Initialized	—	—	—	—	—	Initialized	
IPRE	Initialized	—	—	—	—	—	Initialized	
IPRF	Initialized	—	—	—	—	—	Initialized	
IPRG	Initialized	—	—	—	—	—	Initialized	
IPRI	Initialized	—	—	—	—	—	Initialized	
IPRK	Initialized	—	—	—	—	—	Initialized	
IPRL	Initialized	—	—	—	—	—	Initialized	
IPRO	Initialized	—	—	—	—	—	Initialized	
IPRQ	Initialized	—	—	—	—	—	Initialized	
IPRR	Initialized	—	—	—	—	—	Initialized	
ISCRH	Initialized	—	—	—	—	—	Initialized	
ISCRL	Initialized	—	—	—	—	—	Initialized	
ABWCR	Initialized	—	—	—	—	—	Initialized	BSC
ASTCR	Initialized	—	—	—	—	—	Initialized	
WTCRA	Initialized	—	—	—	—	—	Initialized	
WTCRB	Initialized	—	—	—	—	—	Initialized	
RDNCR	Initialized	—	—	—	—	—	Initialized	
IDLCR	Initialized	—	—	—	—	—	Initialized	
BCR1	Initialized	—	—	—	—	—	Initialized	
BCR2	Initialized	—	—	—	—	—	Initialized	
ENDIANCR	Initialized	—	—	—	—	—	Initialized	
RAMER	Initialized	—	—	—	—	—	Initialized	
MDCR	Initialized	—	—	—	—	—	Initialized	SYSTEM
SYSCR	Initialized	—	—	—	—	—	Initialized	
SCKCR	Initialized	—	—	—	—	—	Initialized	
SBYCR	Initialized	—	—	—	—	—	Initialized	
MSTPCRA	Initialized	—	—	—	—	—	Initialized	
MSTPCRB	Initialized	—	—	—	—	—	Initialized	
MSTPCRC	Initialized	—	—	—	—	—	Initialized	
SUBCKCR	Initialized	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
FCCS	Initialized	—	—	—	—	—	Initialized	FLASH
FPCS	Initialized	—	—	—	—	—	Initialized	
FECS	Initialized	—	—	—	—	—	Initialized	
FKEY	Initialized	—	—	—	—	—	Initialized	
FTDAR	Initialized	—	—	—	—	—	Initialized	
SMR_4	Initialized	—	—	—	—	Initialized	Initialized	SCI_4
BRR_4	Initialized	—	—	—	—	Initialized	Initialized	
SCR_4	Initialized	—	—	—	—	Initialized	Initialized	
TDR_4	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_4	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_4	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_4	Initialized	—	—	—	—	Initialized	Initialized	
SMR_5	Initialized	—	—	—	—	Initialized	Initialized	SCI_5
BRR_5	Initialized	—	—	—	—	Initialized	Initialized	
SCR_5	Initialized	—	—	—	—	Initialized	Initialized	
TDR_5	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_5	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_5	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_5	Initialized	—	—	—	—	Initialized	Initialized	
ICCRA_0	Initialized	—	—	—	—	—	Initialized	IIC2_0
ICCRB_0	Initialized	—	—	—	—	—	Initialized	
ICMR_0	Initialized	—	—	—	—	—	Initialized	
ICIER_0	Initialized	—	—	—	—	—	Initialized	
ICSR_0	Initialized	—	—	—	—	—	Initialized	
SAR_0	Initialized	—	—	—	—	—	Initialized	
ICDRT_0	Initialized	—	—	—	—	—	Initialized	
ICDRR_0	Initialized	—	—	—	—	—	Initialized	
ICCRA_1	Initialized	—	—	—	—	—	Initialized	IIC2_1
ICCRB_1	Initialized	—	—	—	—	—	Initialized	
ICMR_1	Initialized	—	—	—	—	—	Initialized	
ICIER_1	Initialized	—	—	—	—	—	Initialized	
ICSR_1	Initialized	—	—	—	—	—	Initialized	
SAR_1	Initialized	—	—	—	—	—	Initialized	
ICDRT_1	Initialized	—	—	—	—	—	Initialized	
ICDRR_1	Initialized	—	—	—	—	—	Initialized	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
TCR_4	Initialized	—	—	—	—	—	Initialized	TPU_4
TMDR_4	Initialized	—	—	—	—	—	Initialized	
TIOR_4	Initialized	—	—	—	—	—	Initialized	
TIER_4	Initialized	—	—	—	—	—	Initialized	
TSR_4	Initialized	—	—	—	—	—	Initialized	
TCNT_4	Initialized	—	—	—	—	—	Initialized	
TGRA_4	Initialized	—	—	—	—	—	Initialized	
TGRB_4	Initialized	—	—	—	—	—	Initialized	
TCR_5	Initialized	—	—	—	—	—	Initialized	TPU_5
TMDR_5	Initialized	—	—	—	—	—	Initialized	
TIOR_5	Initialized	—	—	—	—	—	Initialized	
TIER_5	Initialized	—	—	—	—	—	Initialized	
TSR_5	Initialized	—	—	—	—	—	Initialized	
TCNT_5	Initialized	—	—	—	—	—	Initialized	
TGRA_5	Initialized	—	—	—	—	—	Initialized	
TGRB_5	Initialized	—	—	—	—	—	Initialized	
INTCR	Initialized	—	—	—	—	—	Initialized	INTC
CPUPCR	Initialized	—	—	—	—	—	Initialized	
IER	Initialized	—	—	—	—	—	Initialized	
ISR	Initialized	—	—	—	—	—	Initialized	
PORT1	—	—	—	—	—	—	—	I/O port
PORT2	—	—	—	—	—	—	—	
PORT3	—	—	—	—	—	—	—	
PORT4	—	—	—	—	—	—	—	
PORT5	—	—	—	—	—	—	—	
PORT6	—	—	—	—	—	—	—	
PORTA	—	—	—	—	—	—	—	
PORTD	—	—	—	—	—	—	—	
PORTE	—	—	—	—	—	—	—	
PORTF	—	—	—	—	—	—	—	
P1DR	Initialized	—	—	—	—	—	Initialized	
P2DR	Initialized	—	—	—	—	—	Initialized	
P3DR	Initialized	—	—	—	—	—	Initialized	
P6DR	Initialized	—	—	—	—	—	Initialized	

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
PADR	Initialized	—	—	—	—	—	Initialized	I/O port
PDDR	Initialized	—	—	—	—	—	Initialized	
PEDR	Initialized	—	—	—	—	—	Initialized	
PFDR	Initialized	—	—	—	—	—	Initialized	
SMR_2	Initialized	—	—	—	—	Initialized	Initialized	SCI_2
BRR_2	Initialized	—	—	—	—	Initialized	Initialized	
SCR_2	Initialized	—	—	—	—	Initialized	Initialized	
TDR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_2	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_2	Initialized	—	—	—	—	Initialized	Initialized	
SMR_0	Initialized	—	—	—	—	Initialized	Initialized	SCI_0
BRR_0	Initialized	—	—	—	—	Initialized	Initialized	
SCR_0	Initialized	—	—	—	—	Initialized	Initialized	
TDR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SSR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
RDR_0	Initialized	—	Initialized	Initialized	Initialized	Initialized	Initialized	
SCMR_0	Initialized	—	—	—	—	Initialized	Initialized	
ADDRA_0	Initialized	—	—	—	—	—	Initialized	A/D_0
ADDRB_0	Initialized	—	—	—	—	—	Initialized	
ADDRC_0	Initialized	—	—	—	—	—	Initialized	
ADDRD_0	Initialized	—	—	—	—	—	Initialized	
ADDRE_0	Initialized	—	—	—	—	—	Initialized	
ADDRF_0	Initialized	—	—	—	—	—	Initialized	
ADDRG_0	Initialized	—	—	—	—	—	Initialized	
ADDRH_0	Initialized	—	—	—	—	—	Initialized	
ADCSR_0	Initialized	—	—	—	—	—	Initialized	
ADCR_0	Initialized	—	—	—	—	—	Initialized	
TCSR	Initialized	—	—	—	—	—	Initialized	WDT
TCNT	Initialized	—	—	—	—	—	Initialized	
RSTCSR	Initialized	—	—	—	—	—	Initialized	
TSTR	Initialized	—	—	—	—	—	Initialized	TPU
TSYR	Initialized	—	—	—	—	—	Initialized	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
TCR_0	Initialized	—	—	—	—	—	Initialized	TPU_0
TMDR_0	Initialized	—	—	—	—	—	Initialized	
TIORH_0	Initialized	—	—	—	—	—	Initialized	
TIORL_0	Initialized	—	—	—	—	—	Initialized	
TIER_0	Initialized	—	—	—	—	—	Initialized	
TSR_0	Initialized	—	—	—	—	—	Initialized	
TCNT_0	Initialized	—	—	—	—	—	Initialized	
TGRA_0	Initialized	—	—	—	—	—	Initialized	
TGRB_0	Initialized	—	—	—	—	—	Initialized	
TGRC_0	Initialized	—	—	—	—	—	Initialized	
TGRD_0	Initialized	—	—	—	—	—	Initialized	
TCR_1	Initialized	—	—	—	—	—	Initialized	TPU_1
TMDR_1	Initialized	—	—	—	—	—	Initialized	
TIOR_1	Initialized	—	—	—	—	—	Initialized	
TIER_1	Initialized	—	—	—	—	—	Initialized	
TSR_1	Initialized	—	—	—	—	—	Initialized	
TCNT_1	Initialized	—	—	—	—	—	Initialized	
TGRA_1	Initialized	—	—	—	—	—	Initialized	
TGRB_1	Initialized	—	—	—	—	—	Initialized	
TCR_2	Initialized	—	—	—	—	—	Initialized	TPU_2
TMDR_2	Initialized	—	—	—	—	—	Initialized	
TIOR_2	Initialized	—	—	—	—	—	Initialized	
TIER_2	Initialized	—	—	—	—	—	Initialized	
TSR_2	Initialized	—	—	—	—	—	Initialized	
TCNT_2	Initialized	—	—	—	—	—	Initialized	
TGRA_2	Initialized	—	—	—	—	—	Initialized	
TGRB_2	Initialized	—	—	—	—	—	Initialized	
TCR_3	Initialized	—	—	—	—	—	Initialized	TPU_3
TMDR_3	Initialized	—	—	—	—	—	Initialized	
TIORH_3	Initialized	—	—	—	—	—	Initialized	
TIORL_3	Initialized	—	—	—	—	—	Initialized	
TIER_3	Initialized	—	—	—	—	—	Initialized	
TSR_3	Initialized	—	—	—	—	—	Initialized	
TCNT_3	Initialized	—	—	—	—	—	Initialized	

RENESAS

Register Abbreviation	Reset	Sleep	Sub Clock	Module Stop	All-Module-Clock-Stop	Software Standby	Hardware Standby	Module
TGRA_3	Initialized	—	—	—	—	—	Initialized	TPU_3
TGRB_3	Initialized	—	—	—	—	—	Initialized	
TGRC_3	Initialized	—	—	—	—	—	Initialized	
TGRD_3	Initialized	—	—	—	—	—	Initialized	

Notes: 1. SSU: Synchronous Serial communication Unit
2. PWM16 is 16-bit PWM.
3. PWM10 is Motor Control PWM.

RENESAS

Section 26   Electrical Characteristics

26.1   Absolute Maximum Ratings

Table 26.1   Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage	V_{CC}	−0.3 to +7.0	V
Input voltage (except ports 4 and 5)	V_{in}	−0.3 to V_{CC} + 0.3	V
Input voltage (port 4)	V_{in}	−0.3 to AV_{CC1} + 0.3	V
Input voltage (port 5)	V_{in}	−0.3 to AV_{CC0} + 0.3	V
Reference power supply voltage	Vref	−0.3 to AV_{CC0} + 0.3	V
Analog power supply voltage	AV_{CC0}	−0.3 to +7.0	V
	AV_{CC1}	−0.3 to +7.0	V
Analog input voltage (port 4)	V_{AN}	−0.3 to AV_{CC1} + 0.3	V
Analog input voltage (port 5)	V_{AN}	−0.3 to AV_{CC0} + 0.3	V
Operating temperature	T_{opr}	Regular specifications: −20 to +75*	°C
		Wide-range specifications: −40 to +85*	
Storage temperature	T_{stg}	−55 to +125	°C

Caution: Permanent damage to the LSI may result if absolute maximum ratings are exceeded.

Note: *   The operating temperature when programming/erasing the flash memory ranges from 0°C to +75°C for regular specification products and from 0°C to +85°C for wide-range specification products.

26.2   DC Characteristics

Table 26.2   DC Characteristics (1)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V*[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Schmitt trigger input pin voltage	$\overline{\text{IRQ}}$ input pin, TPU input pin	VT$^-$	$V_{CC} \times 0.2$	—	—	V	
		VT$^+$	—	—	$V_{CC} \times 0.7$		
		VT$^+$ – VT$^-$	$V_{CC} \times 0.05$	—	—		
Input high voltage (except Schmitt trigger input pin)	$\overline{\text{STBY}}$, EMLE, MD, $\overline{\text{RES}}$, NMI	V_{IH}	$V_{CC} - 0.7$	—	$V_{CC} + 0.3$	V	
	EXTAL		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
	Other input pins		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
	Port 4		$AV_{CC}1 \times 0.7$	—	$AV_{CC1} + 0.3$		
	Port 5		$AV_{CC}0 \times 0.7$	—	$AV_{CC0} + 0.3$		
Input low voltage (except Schmitt trigger input pin)	$\overline{\text{STBY}}$, EMLE, $\overline{\text{RES}}$, MD, NMI	V_{IL}	–0.3	—	$V_{CC} \times 0.1$	V	
	EXTAL		–0.3	—	$V_{CC} \times 0.2$		
	Other pins		–0.3	—	$V_{CC} \times 0.2$		
	Port 4		–0.3	—	$AV_{CC1} \times 0.2$		
	Port 5		–0.3	—	$AV_{CC0} \times 0.2$		
Output high voltage	All output pins	V_{OH}	$V_{CC} - 0.5$	—	—	V	$I_{OH} = -200$ μA
			$V_{CC} - 1.0$	—	—		$I_{OH} = -1$ mA
Output low voltage	All output pins	V_{OL}	—	—	0.4	V	$I_{OL} = 1.6$ mA
Input leakage current	$\overline{\text{STBY}}$, EMLE, $\overline{\text{RES}}$, NMI, MD	$|I_{in}|$	—	—	1.0	μA	V_{in} = 0.5 to $V_{CC} - 0.5$ V
	Port 4		—	—	1.0		V_{in} = 0.5 to $AV_{CC1} - 0.5$ V
	Port 5		—	—	1.0		V_{in} = 0.5 to $AV_{CC0} - 0.5$ V

RENESAS

Table 26.2   DC Characteristics (2)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{ss} = PWMV$_{ss}$ = AV_{ss} = 0 V[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Tri-state leakage current (off state)	Ports 1 to 3, 6, A, D, E, F, H, I, J, K	$\lvert I_{TSI} \rvert$	—	—	1.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input pull-up MOS current	Ports D, E, F, H, I, J, K	$-I_p$	50	—	300	μA	V_{in} = 0 V
Input capacitance	All input pins	C_{in}	—	—	15	pF	V_{in} = 0 V f = 1 MHz T_a = 25°C
Current consumption[2]	Normal operation	I_{CC}[4]	—	65	90	mA	f = 40 MHz
	Sleep mode		—	50	80		
	Standby mode[3] (watch timer not in use)		—	0.1	1.0	mA	$T_a \leq 50°C$
			—	—	2.5		$50°C < T_a$
	Watch mode		—	1.5	2.0		$T_a \leq 50°C$
			—	—	3.5		$50°C < T_a$
	Sub-clock mode		—	8	10		
	All-module-clock-stop mode[5]		—	30	—		Reference value

Table 26.2   DC Characteristics (3)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
$PWMV_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = $PWMV_{SS}$ = AV_{SS} = 0 V*[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Analog power supply current	During A/D or D/A conversion	$AI_{CC}0$	—	2.0	3.0	mA	AV_{CC0} = 5.0 V
	Standby for A/D or D/A conversion		—	10	100	μA	
	During A/D or D/A conversion	$AI_{CC}1$	—	2.0	3.0	mA	AV_{CC1} = 5.0 V
	Standby for A/D or D/A conversion		—	10	100	μA	
RAM standby voltage		V_{RAM}	3.0	—	—	V	

Notes: 1. When the A/D or D/A converter is not used, the AV_{CC0}, AV_{CC1}, Vref, and AV_{SS} pins should not be open. Connect the AV_{CC}, AV_{CC1}. and Vref pins to V_{CC}, and the AV_{SS} pin to V_{SS}.
2. Current consumption values are for V_{IH} = AV_{CC0} (port 5), AV_{CC1} (port 4), V_{CC} (others) and V_{IL} = 0 V with all output pins unloaded and all input pull-up MOSs in the off state.
3. The values are for $V_{RAM} \leq V_{CC} <$ 4.5 V, V_{IH}min. = V_{CC} – 0.1 V, and V_{IL}max. = 0.1 V.
4. I_{CC} depends on V_{CC} and f as follows:
I_{CC}max = 41 (mA) + 0.22 (mA/(MHz × V)) × V_{CC} × f (normal operation)
I_{CC}max = 25 (mA) + 0.25 (mA/(MHz × V)) × V_{CC} × f (sleep mode)
5. The values are for reference.

RENESAS

Table 26.3   Permissible Output Currents (1)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{ss} = PWMV$_{ss}$ = AV_{ss} = 0 V*[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Permissible output low current (per pin)	All output pins except for PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	I_{OL}	—	—	10	mA	
	PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	I_{OL}	—	—	25	mA	T_a = 75°C to 80°C
					30	mA	T_a = 25°C
					40	mA	T_a = –40°C
Permissible output low current (total)	Total of all output pins except for PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	ΣI_{OL}	—	—	120	mA	
	Total of PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	ΣI_{OL}	—	—	225	mA	T_a = 75°C to 80°C
					270	mA	T_a = 25°C
					330	mA	T_a = –40°C

Table 26.3   Permissible Output Currents (2)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V*[1],
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Permissible output high current (per pin)	All output pins except for PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	$-I_{OH}$	—	—	2.0	mA	
	PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	$-I_{OH}$	—	—	25	mA	T_a = 75°C to 80°C
					30	mA	T_a = 25°C
					40	mA	T_a = –40°C
Permissible output high current (total)	Total of all output pins except for PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	$-\Sigma I_{OH}$	—	—	60	mA	
	Total of PWM1A to PWM1H, PWM2A to PWM2H, PWM0_0 to PWM3_0, and PWM0_1 to PWM3_1	$-\Sigma I_{OH}$	—	—	225	mA	T_a = 75°C to 80°C
					270	mA	T_a = 25°C
					330	mA	T_a = –40°C

Caution: To ensure the LSI's reliability, do not exceed the output current values in table 26.3.

Note: *   When the A/D or D/A converter is not used, the AV_{CC0}, AV_{CC1}, Vref, and AV_{SS} pins should not be open. Connect the AV_{CC0}, AV_{CC1}, and Vref pins to V_{CC}, and the AV_{SS} pin to V_{SS}.

RENESAS

26.3   AC Characteristics
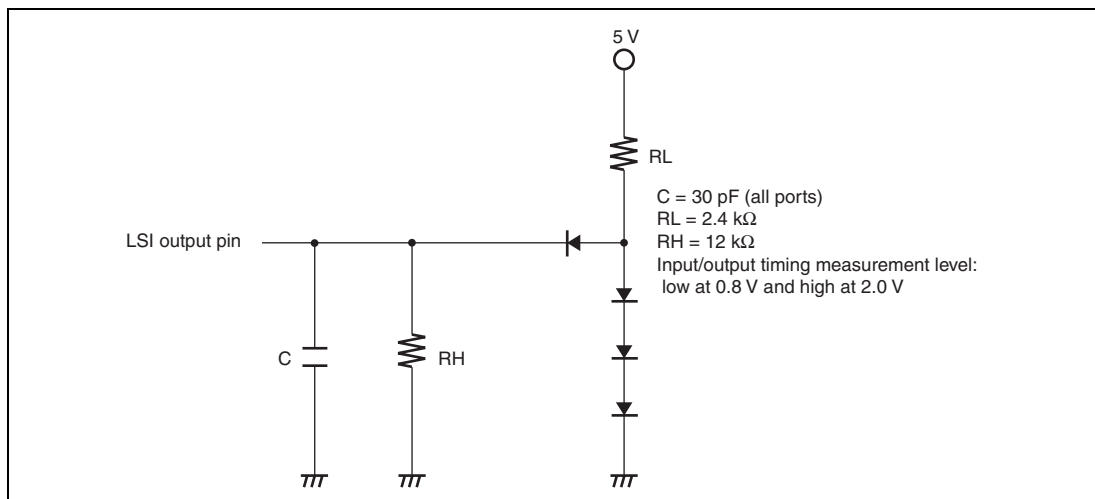
Figure 26.1   Output Load Circuit

26.3.1   Clock Timing

Table 26.4   Clock Timing

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V,
Iϕ = 8 MHz to 40 MHz, Pϕ, Bϕ= 8 MHz to 20 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit.	Test Conditions
Clock cycle time	t_{cyc}	50	125	ns	Figure 26.2
Clock high pulse width	t_{CH}	5	—	ns	
Clock low pulse width	t_{CL}	5	—	ns	
Clock rising time	t_{Cr}	—	5	ns	
Clock falling time	t_{Cf}	—	5	ns	
Oscillation settling time after reset (crystal)	t_{OSC1}	20	—	ms	Figure 26.4
Oscillation settling time after leaving software standby mode (crystal)	t_{OSC2}	10	—	ms	Figure 26.3
External clock output delay settling time	t_{DEXT}	2	—	ms	Figure 26.4
External clock input low pulse width	T_{EXL}	45	—	ns	Figure 26.5
External clock input high pulse width	T_{EXH}	45	—	ns	External clock input frequency = 4 to 9 MHz
External clock rising time	T_{EXr}	—	5	ns	
External clock falling time	T_{EXf}	—	5	ns	
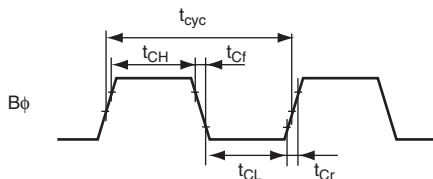
Figure 26.2   External Bus Clock Timing

RENESAS
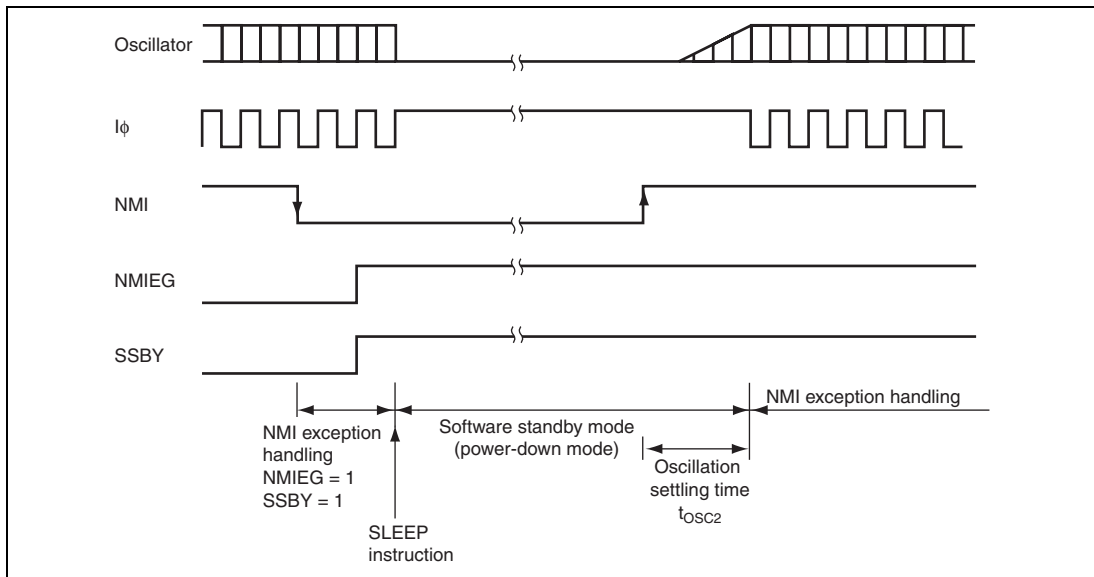
Figure 26.3   Oscillation Settling Timing after Software Standby Mode
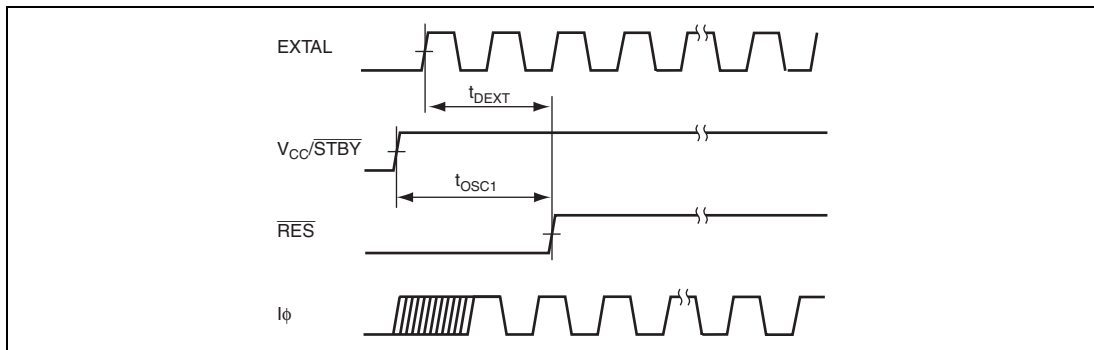
Figure 26.4   Oscillation Settling Timing
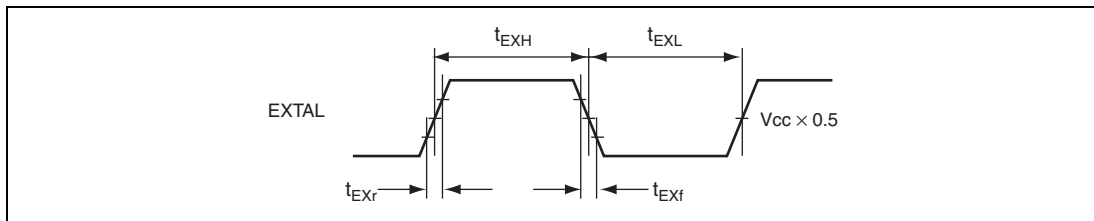
Figure 26.5   External Input Clock Timing

26.3.2   Control Signal Timing

Table 26.5   Control Signal Timing

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V,
Iϕ = 8 MHz to 40 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{RES}$ setup time	t_{RESS}	200	—	ns	Figure 26.6
$\overline{RES}$ pulse width	t_{RESW}	20	—	t_{cyc}	
NMI setup time	t_{NMIS}	150	—	ns	Figure 26.7
NMI hold time	t_{NMIH}	10	—	ns	
NMI pulse width (after leaving software standby mode)	t_{NMIW}	200	—	ns	
$\overline{IRQ}$ setup time	t_{IRQS}	150	—	ns	
$\overline{IRQ}$ hold time	t_{IRQH}	10	—	ns	
$\overline{IRQ}$ pulse width (after leaving software standby mode)	t_{IRQW}	200	—	ns	
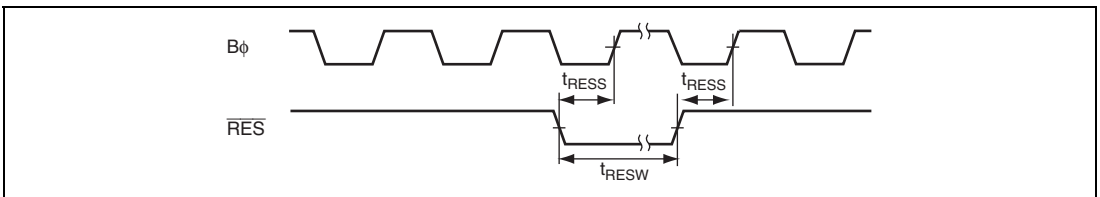
Figure 26.6   Reset Input Timing

Figure 26.7   Interrupt Input Timing

26.3.3   Bus Timing

Table 26.6   Bus Timing (1)

Conditions: V_{cc} = 4.5 V to 5.5 V, AV_{cc0} = 4.5 V to 5.5 V, AV_{cc1} = 4.5 V to 5.5 V,
PWMV$_{cc}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{cc}, V_{ss} = PWMV$_{ss}$ = AV_{ss} = 0 V,
Iϕ = 8 MHz to 40 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
Address delay time	t_{AD}	—	40	ns	Figures 26.8 and 26.9
Address setup time 1	t_{AS1}	$0.5 \times t_{cyc}$ – 16	—	ns	
Address setup time 2	t_{AS2}	$1.0 \times t_{cyc}$ – 16	—	ns	
Address setup time 3	t_{AS3}	$1.5 \times t_{cyc}$ – 16	—	ns	
Address setup time 4	t_{AS4}	$2.0 \times t_{cyc}$ – 16	—	ns	
Address hold time 1	t_{AH1}	$0.5 \times t_{cyc}$ – 16	—	ns	
Address hold time 2	t_{AH2}	$1.0 \times t_{cyc}$ – 16	—	ns	
Address hold time 3	t_{AH3}	$1.5 \times t_{cyc}$ – 16	—	ns	
$\overline{AS}$ delay time	t_{ASD}	—	40	ns	
$\overline{RD}$ delay time 1	t_{RSD1}	—	40	ns	
$\overline{RD}$ delay time 2	t_{RSD2}	—	40	ns	
Read data setup time 1	t_{RDS1}	15	—	ns	
Read data setup time 2	t_{RDS2}	15	—	ns	
Read data hold time 1	t_{RDH1}	0	—	ns	
Read data hold time 2	t_{RDH2}	0	—	ns	
Read data access time 2	t_{AC2}	—	$1.5 \times t_{cyc}$ – 45	ns	
Read data access time 4	t_{AC4}	—	$2.5 \times t_{cyc}$ – 45	ns	
Read data access time 5	t_{AC5}	—	$1.0 \times t_{cyc}$ – 45	ns	

RENESAS

Table 26.6   Bus Timing (2)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
              $PWMV_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = $PWMV_{SS}$ = AV_{SS} = 0 V,
              Iϕ = 8 MHz to 40 MHz,
              T_a = –20°C to +75°C (regular specifications),
              T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
Read data access time 6	t_{AC6}	—	$2.0 \times t_{cyc} - 45$	ns	Figures 26.8 and 26.9
Read data access time (from address) 2	t_{AA2}	—	$1.5 \times t_{cyc} - 55$	ns	
Read data access time (from address) 3	t_{AA3}	—	$2.0 \times t_{cyc} - 55$	ns	
Read data access time (from address) 4	t_{AA4}		$2.5 \times t_{cyc} - 55$	ns	
Read data access time (from address) 5	t_{AA5}	—	$3.0 \times t_{cyc} - 55$	ns	
$\overline{WR}$ delay time 1	t_{WRD1}	—	35	ns	
$\overline{WR}$ delay time 2	t_{WRD2}	—	40	ns	
$\overline{WR}$ pulse width 1	t_{WSW1}	$1.0 \times t_{cyc} - 20$	—	ns	
$\overline{WR}$ pulse width 2	t_{WSW2}	$1.5 \times t_{cyc} - 20$	—	ns	
Write data delay time	t_{WDD}	—	40	ns	
Write data setup time 1	t_{WDS1}	$0.5 \times t_{cyc} - 20$	—	ns	
Write data setup time 2	t_{WDS2}	$1.0 \times t_{cyc} - 20$	—	ns	
Write data setup time 3	t_{WDS3}	$1.5 \times t_{cyc} - 20$	—	ns	
Write data hold time 1	t_{WDH1}	$0.5 \times t_{cyc} - 16$	—	ns	
Write data hold time 3	t_{WDH3}	$1.5 \times t_{cyc} - 16$	—	ns	

RENESAS
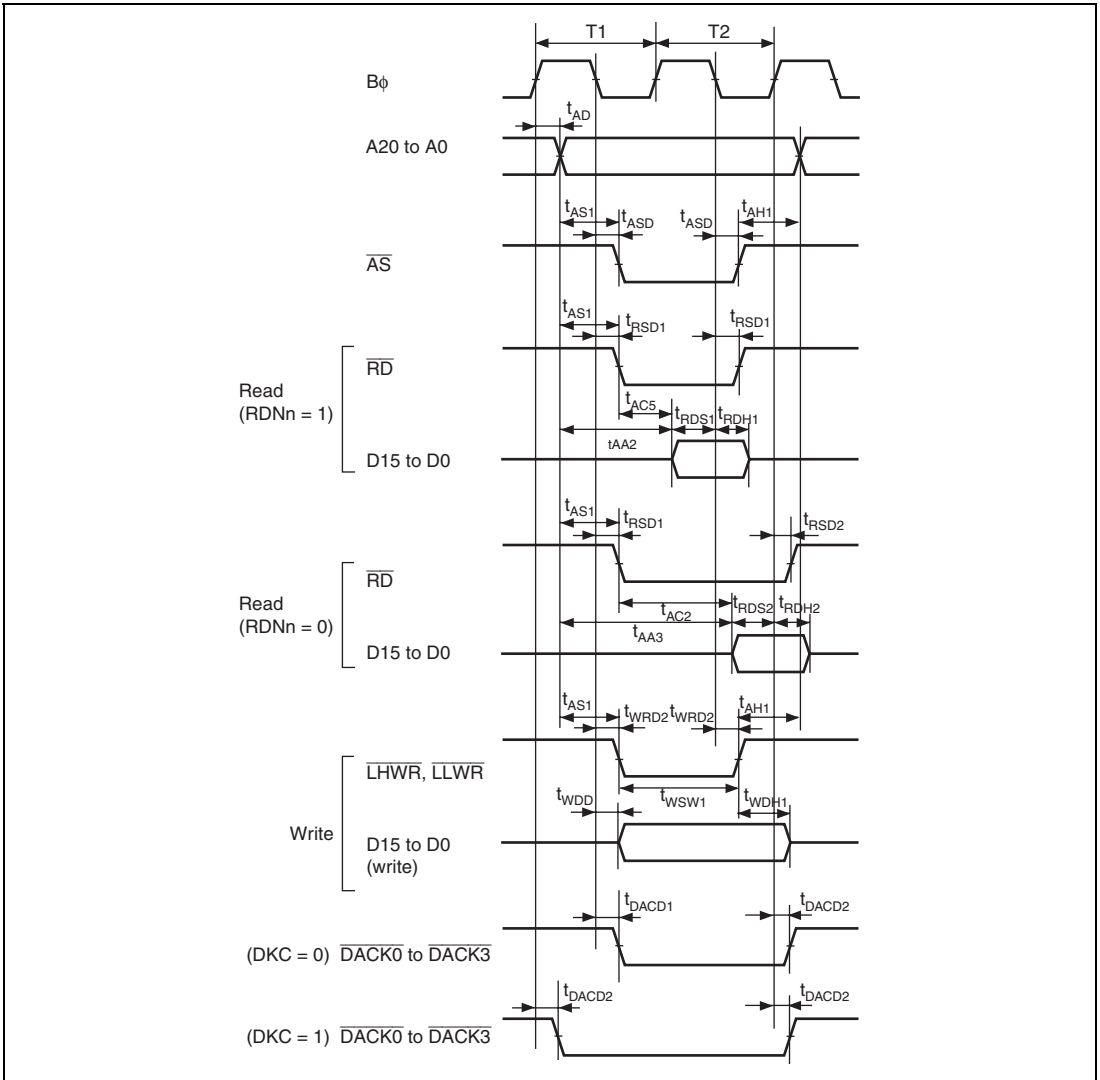
Figure 26.8   Basic Bus Timing: 2-State Access
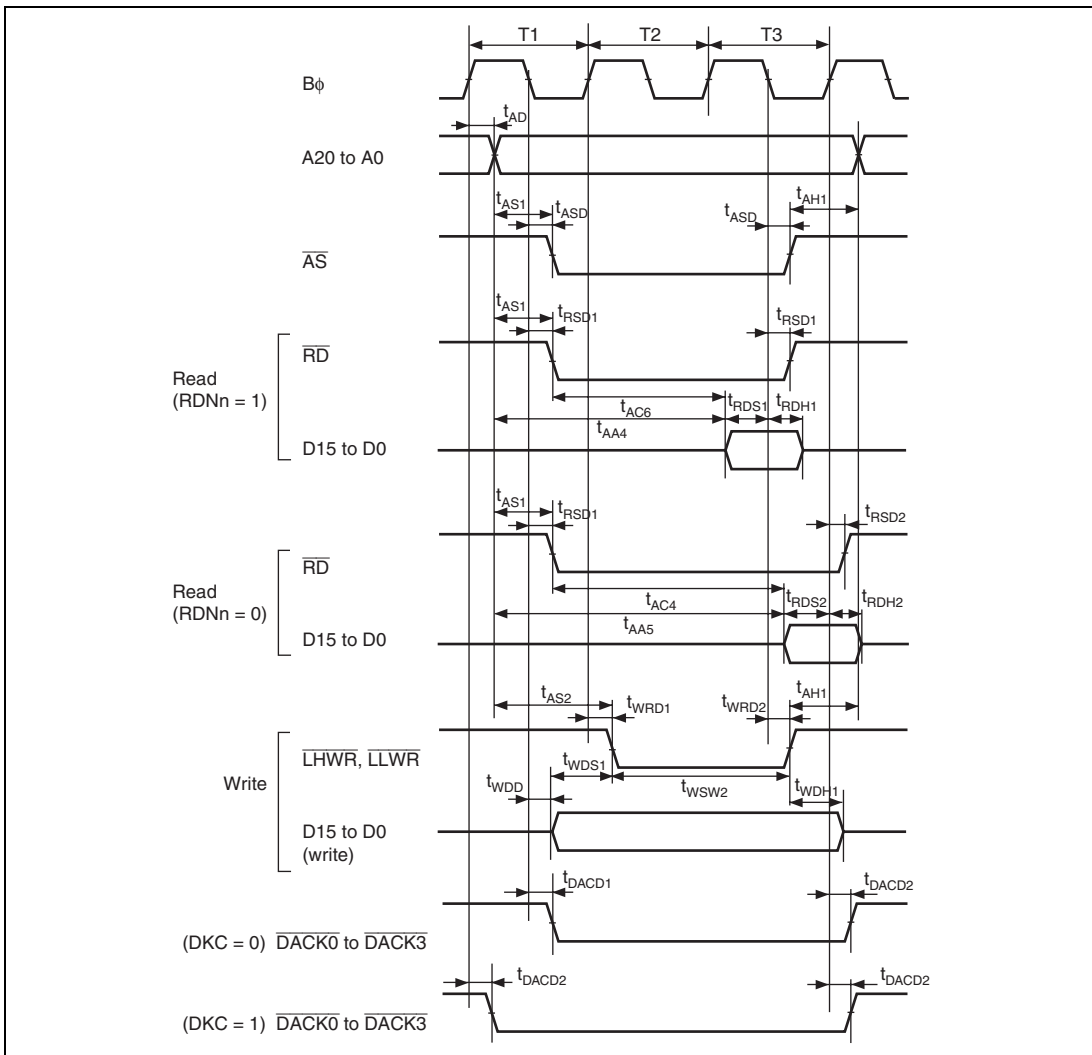
Figure 26.9   Basic Bus Timing: 3-State Access

RENESAS

26.3.4   DMAC Timing

Table 26.7   DMAC Timing

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC} = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC},
V_{SS} = AV_{SS} = 0 V, Bφ = 8 MHz to 20 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{DREQ}$ setup time	t_{DRQS}	20	—	ns	Figure 26.10
$\overline{DREQ}$ hold time	t_{DRQH}	5	—	ns	
$\overline{TEND}$ delay time	t_{TED}	—	15	ns	Figure 26.11
$\overline{DACK}$ delay time 1	t_{DACD1}	—	15	ns	Figures 26.12, 26.13
$\overline{DACK}$ delay time 2	t_{DACD2}	—	15	ns	
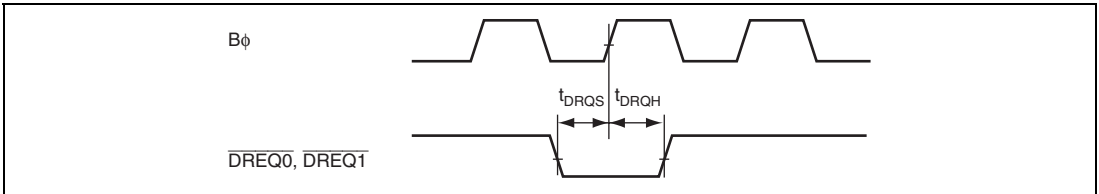
Figure 26.10   DMAC ($\overline{DREQ}$) Input Timing
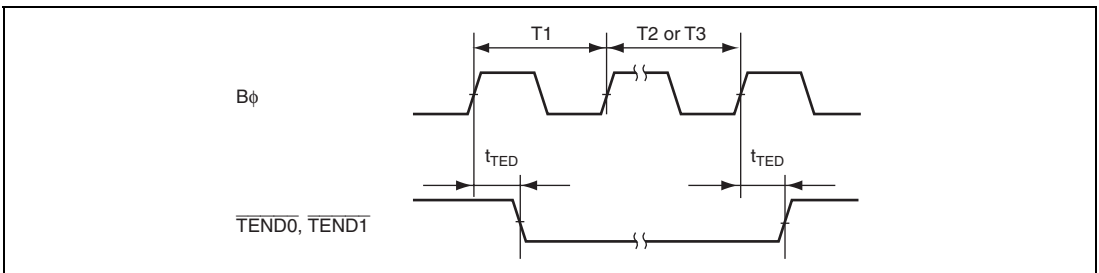
Figure 26.11   DMAC ($\overline{TEND}$) Output Timing
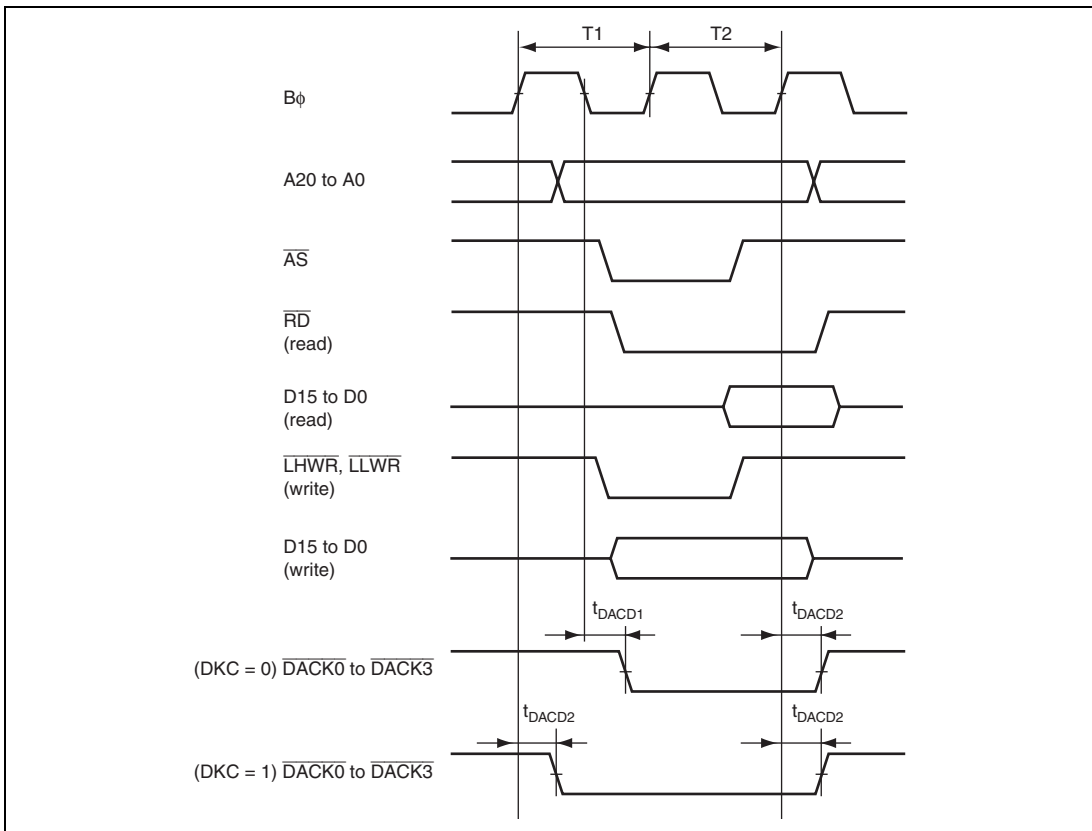
Figure 26.12   DMAC Single-Address Transfer Timing: 2-State Access
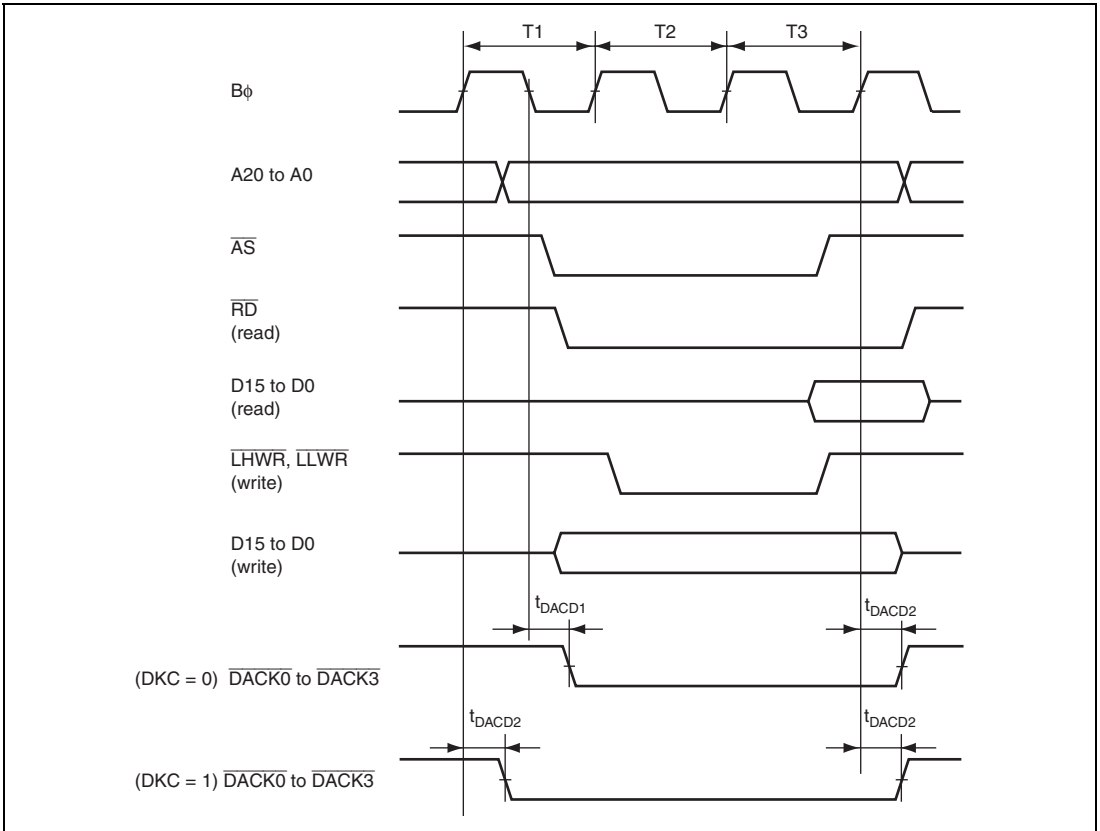
Figure 26.13   DMAC Single-Address Transfer Timing: 3-State Access

RENESAS

26.3.5   Timing of On-Chip Peripheral Modules

Table 26.8   Timing of On-Chip Peripheral Modules (1)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V,
Pϕ = 8 MHz to 20 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

	Item		Symbol	Min.	Max.	Unit	Test Conditions
I/O ports	Output data delay time		t_{PWD}	—	40	ns	Figure 26.14
	Input data setup time		t_{PRS}	25	—	ns	
	Input data hold time		t_{PRH}	25	—	ns	
TPU	Timer output delay time		t_{TOCD}	—	40	ns	Figure 26.15
	Timer input setup time		t_{TICS}	25	—	ns	
	Timer clock input setup time		t_{TCKS}	25	—	ns	Figure 26.16
	Timer clock pulse width	Single-edge setting	t_{TCKWH}	1.5	—	t_{cyc}	
		Both-edge setting	t_{TCKWL}	2.5	—	t_{cyc}	
PWM	Pulse output delay time		t_{MPWMOD}	—	50	ns	Figures 26.17, 26.18
SCI	Input clock cycle	Asynchronous	t_{Scyc}	4	—	t_{cyc}	Figure 26.19
		Clocked synchronous		6	—		
	Input clock pulse width		t_{SCKW}	0.4	0.6	t_{Scyc}	
	Input clock rise time		t_{SCKr}	—	20	ns	Figure 26.19 Reference voltage level $V_{cc} \times 0.3$ V to $V_{cc} \times 0.7$ V
	Input clock fall time		t_{SCKf}	—	20	ns	
	Output clock cycle	Asynchronous	t_{Scyc}	30	—	t_{cyc}	Figure 26.19
		Clocked synchronous		4	—		
	Output clock pulse width		t_{SCKW}	0.4	0.6	t_{Scyc}	

RENESAS

	Item	Symbol	Min.	Max.	Unit	Test Conditions
SCI	Output clock rise time	t_{SCKr}	—	20	ns	Figure 26.19
	Output clock fall time	t_{SCKf}	—	20	ns	Reference voltage level $V_{cc} \times 0.3$ V to $V_{cc} \times 0.7$ V
	Transmit data delay time	t_{TXD}	—	40	ns	Figure 26.20
	Receive data setup time (clocked synchronous)	t_{RXS}	40	—	ns	
	Receive data hold time (clocked synchronous)	t_{RXH}	40	—	ns	
IIC2	SCL input cycle time	t_{SCL}	12 t_{cyc} + 600	—	ns	Figure 26.21
	SCL input high pulse width	t_{SCLH}	3 t_{cyc} + 300	—	ns	
	SCL input low pulse width	t_{SCLL}	5 t_{cyc} + 300	—	ns	
	SCL, SDA input rise time	t_{Sr}	—	7.5 t_{cyc}	ns	
	SCL, SDA input fall time	t_{Sf}	—	300	ns	
	SCL, SDA output fall time	t_{Sf}	—	250	ns	
	SCL, SDA input spike pulse elimination time	t_{SP}	—	1 t_{cyc}	ns	
	SDA input bus free time	t_{BUF}	5 t_{cyc}	—	ns	
	Start condition input hold time	t_{STAH}	3 t_{cyc}	—	ns	
	Repeated start condition input setup time	t_{STAS}	3 t_{cyc}	—	ns	
	Stop condition input setup time	t_{STOS}	3 t_{cyc}	—	ns	
	Data input setup time	t_{SDAS}	1 t_{cyc} + 20	—	ns	
	Data input hold time	t_{SDAH}	0	—	ns	
	SCL, SDA capacitive load	Cb	0	400	pF	
A/D converter	Trigger input setup time	t_{TRGS}	30	—	ns	Figure 26.22

RENESAS

Table 26.8   Timing of On-Chip Peripheral Modules (2)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V,
Pϕ = 8 MHz to 20 MHz,
T_a = –20°C to +75°C (regular specifications),
T_a = –40°C to +85°C (wide-range specifications)

	Item		Symbol	Min.	Max.	Unit	Test Conditions
RCAN-ET[1]	Transmit data delay time		t_{CTXD}	—	100	ns	Figure 26.23
	Receive data setup time		t_{CRXS}	100	—	ns	
	Receive data hold time		t_{CRXH}	100	—	ns	
SSU[2]	Clock cycle time	Master	t_{SUcyc}	4	256	t_{cyc}	Figures 26.24 to 26.27
		Slave		4	256		
	Clock high pulse width	Master	t_{HI}	80	—	ns	
		Slave		80	—		
	Clock low pulse width	Master	t_{LO}	80	—	ns	
		Slave		80	—		
	Clock rising time		t_{RISE}	—	20	ns	
	Clock falling time		t_{FALL}	—	20	ns	
	Data input setup time	Master	t_{SU}	25	—	ns	
		Slave		30	—		
	Data input hold time	Master	t_{H}	10	—	ns	
		Slave		10	—		
	$\overline{SCS}$ setup time	Master	t_{LEAD}	2.5	—	t_{cyc}	
		Slave		2.5	—		
	$\overline{SCS}$ hold time	Master	t_{LAG}	2.5	—	t_{cyc}	
		Slave		2.5	—		
	Data output delay time	Master	t_{OD}	—	40	ns	
		Slave		—	40		

RENESAS

	Item		Symbol	Min.	Max.	Unit	Test Conditions
SSU*[2]	Data output hold time	Master	t_{OH}	0	—	ns	Figures 26.24 to 26.27
		Slave		0	—		
	Consecutive transmit delay time	Master	t_{TD}	2.5	—	t_{cyc}	
		Slave		2.5	—		
	Slave access time		t_{SA}	—	1	t_{cyc}	Figures 26.26, 26.27
	Slave out release time		t_{REL}	—	1	t_{cyc}	

Notes: 1. Although the RCAN-ET input/output signals are asynchronous, these are synchronized by the rising edge of the Pϕ as shown in figure 26.23.

2. SSU: Synchronous Serial communication Unit
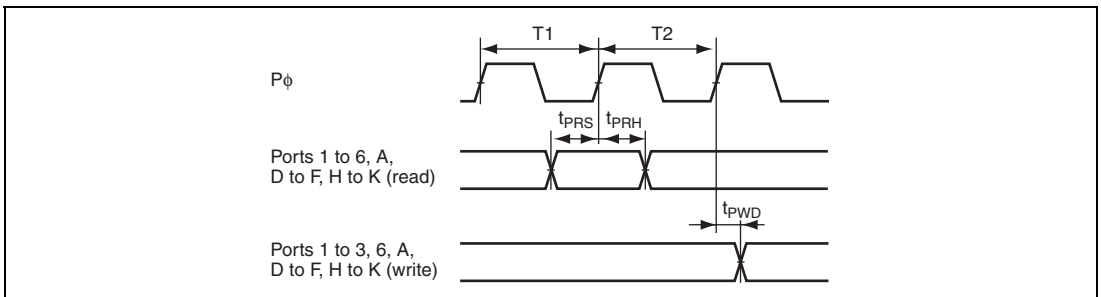
Figure 26.14   I/O Port Input/Output Timing
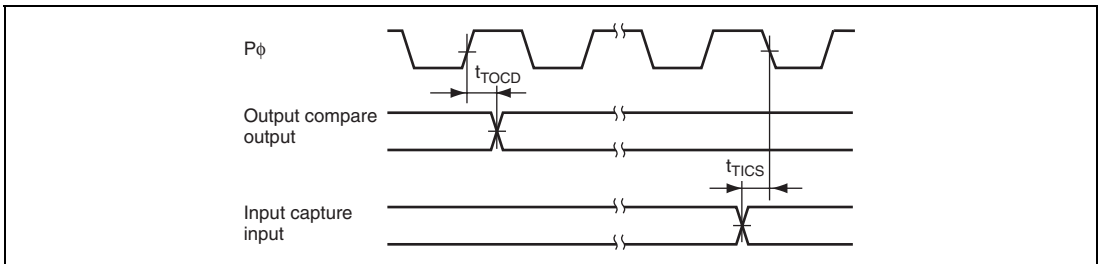
Figure 26.15   TPU Input/Output Timing
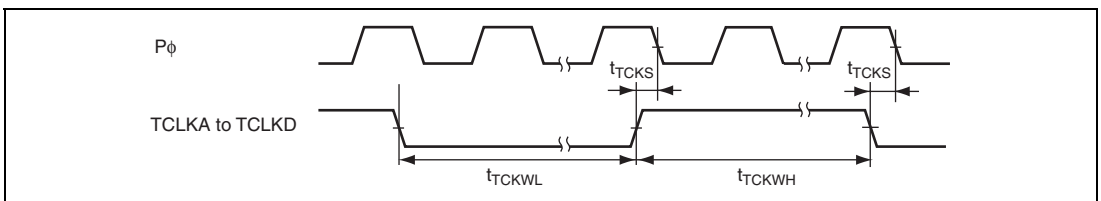
Figure 26.16   TPU Clock Input Timing

RENESAS
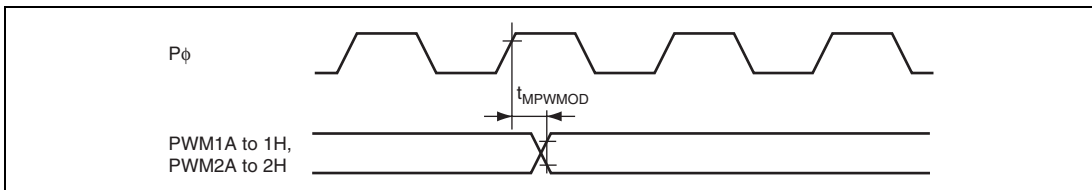
Figure 26.17   Motor Control PWM Output Timing
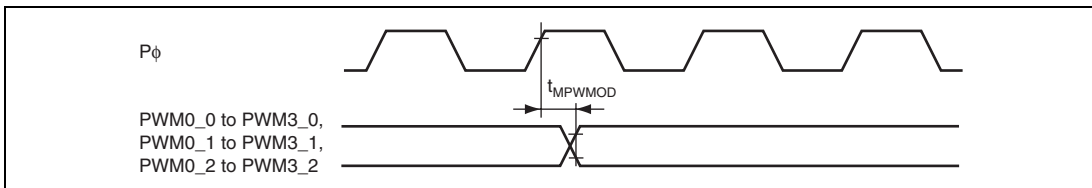
Figure 26.18   16-Bit PWM Output Timing
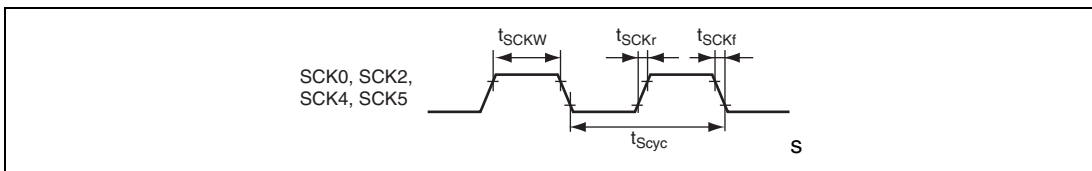
Figure 26.19   SCK Clock Input/Output Timing
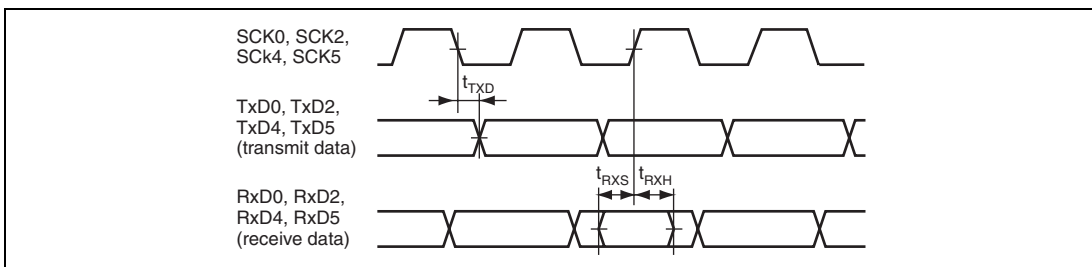
Figure 26.20   SCI Input/Output Timing: Clocked Synchronous Mode
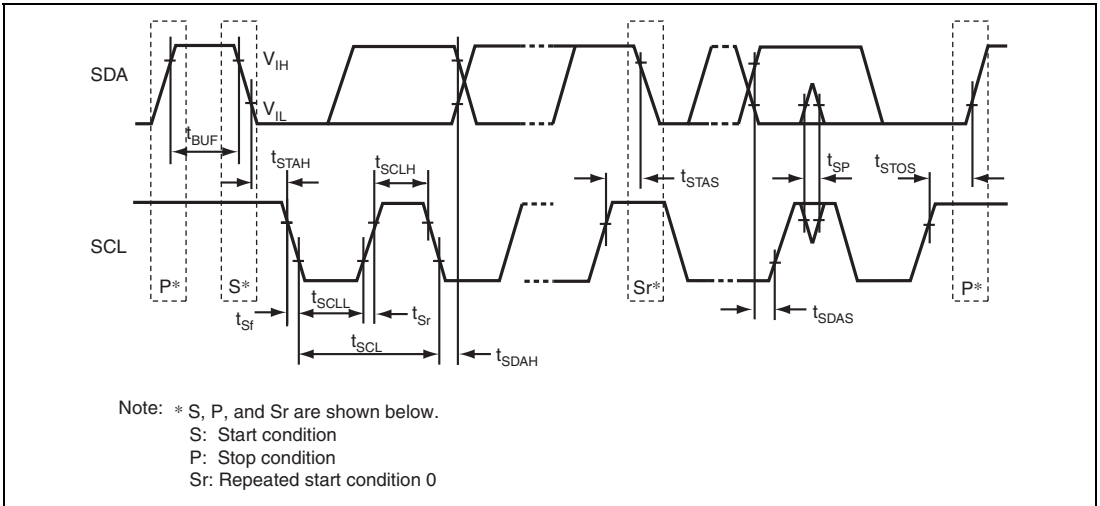
Figure 26.21   I²C Bus Interface Input/Output Timing
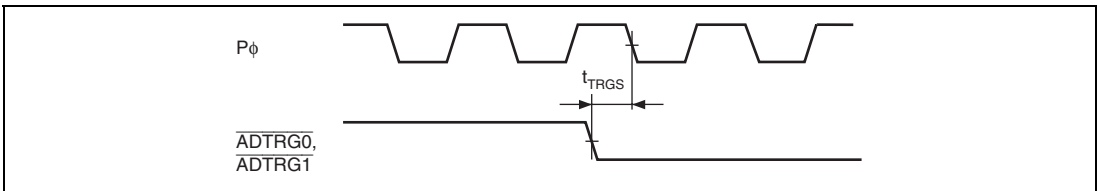
Figure 26.22   A/D Converter External Trigger Input Timing
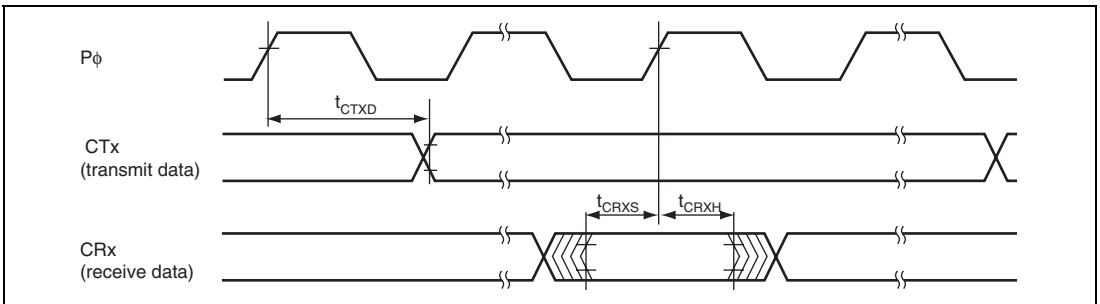
Figure 26.23   RCAN-ET Input/Output Timing
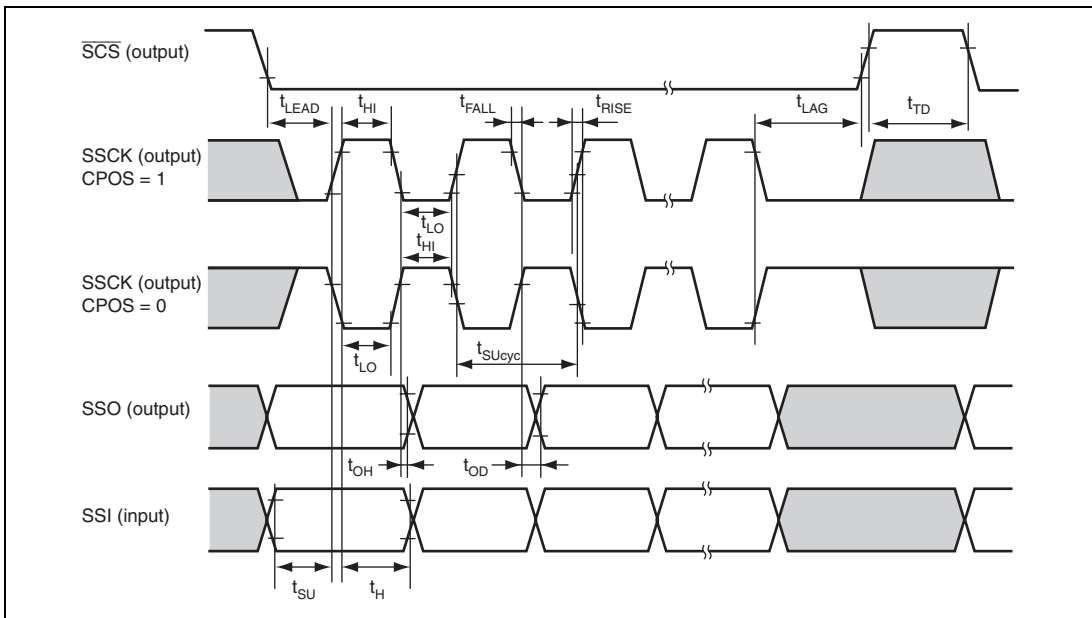
Figure 26.24   SSU Timing (Master, CPHS = 1)
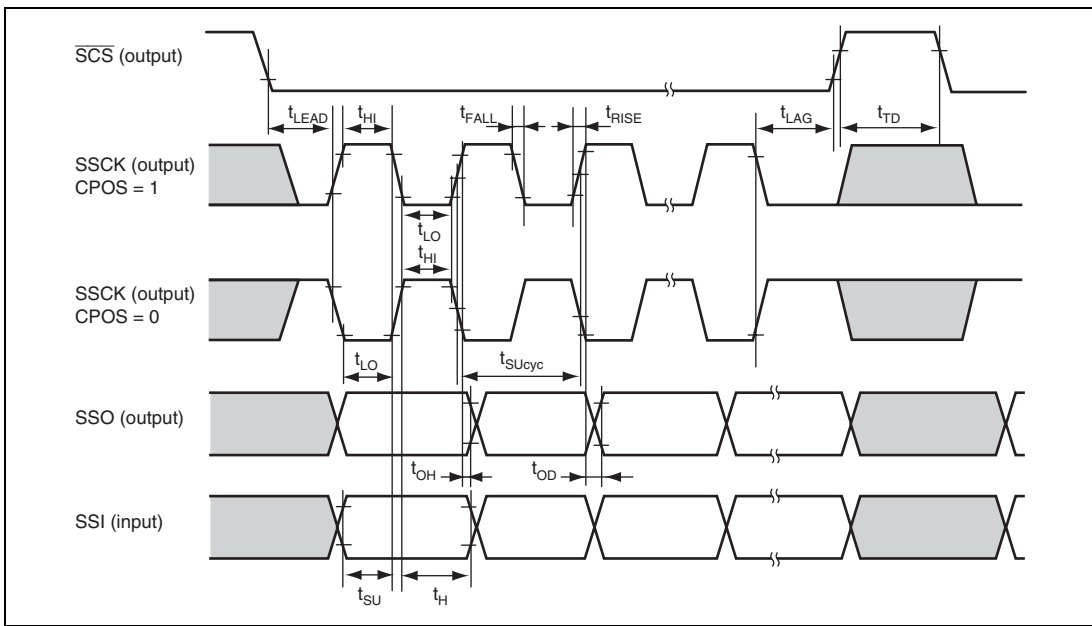
Figure 26.25   SSU Timing (Master, CPHS = 0)
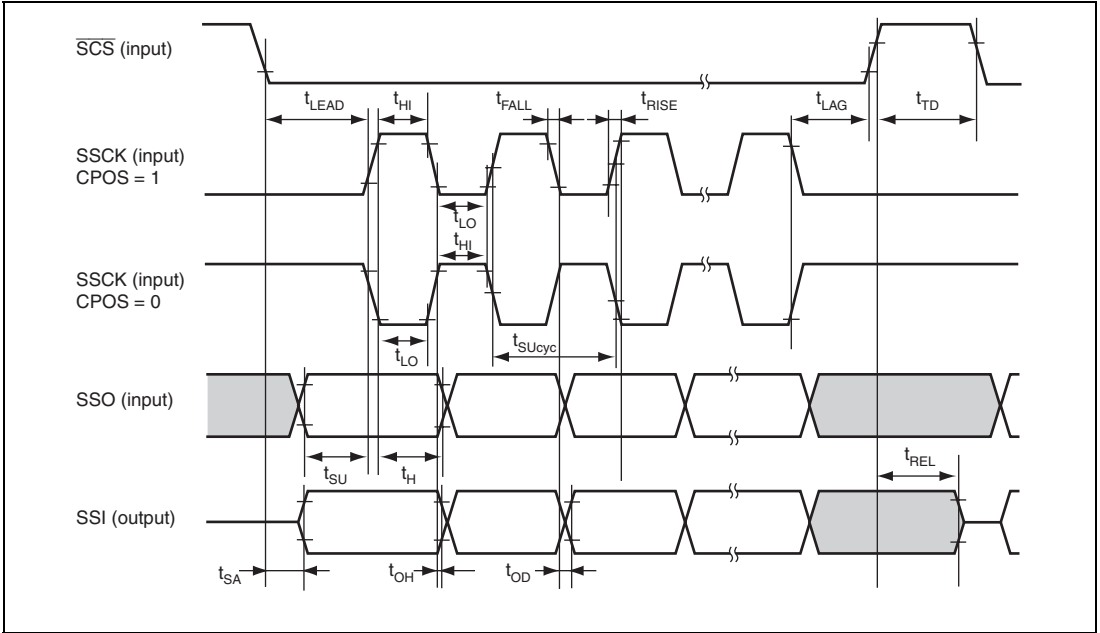
Figure 26.26   SSU Timing (Slave, CPHS = 1)
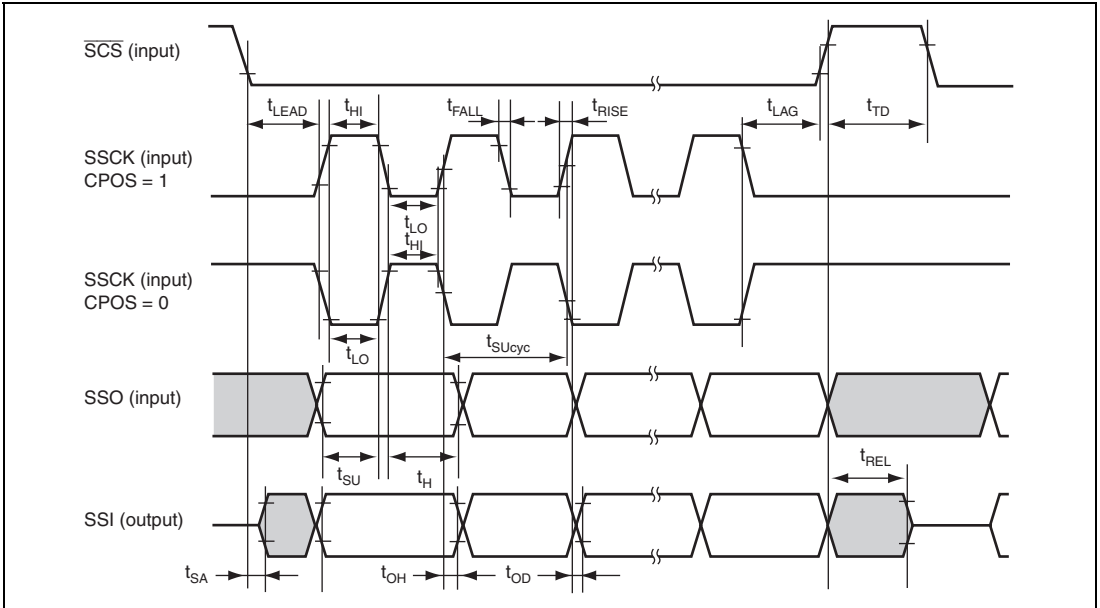
Figure 26.27   SSU Timing (Slave, CPHS = 0)

26.3.6   A/D Conversion Characteristics

Table 26.9   A/D Conversion Characteristics

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
 $PWMV_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{ss} = $PWMV_{ss}$ = AV_{ss} = 0 V,
 Pϕ = 8 MHz to 20 MHz,
 T_a = –20°C to +75°C (regular specifications),
 T_a = –40°C to +85°C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit
Resolution	10	10	10	Bit
Conversion time	7.4	—	200	μs
Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	5	kΩ
Nonlinearity error	—	—	±3.5	LSB
Offset error	—	—	±3.5	LSB
Full-scale error	—	—	±3.5	LSB
Quantization error	—	±0.5	—	LSB
Absolute accuracy	—	—	±4.0	LSB

26.3.7   D/A Conversion Characteristics

Table 26.10 D/A Conversion Characteristics

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC} = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC},
 V_{ss} = $PWMV_{ss}$ = AV_{ss} = 0 V, Pϕ = 8 MHz to 20 MHz,
 T_a = –20°C to +75°C (regular specifications),
 T_a = –40°C to +85°C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit	Test Conditions
Resolution	8	8	8	Bit	
Conversion time	—	—	10	μs	20-pF capacitive load
Absolute accuracy	—	±2.0	±3.0	LSB	2-MΩ resistive load
	—	—	±2.0	LSB	4-MΩ resistive load

26.3.8   Flash Memory Characteristics

Table 26.11 Flash Memory Characteristics

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
PWMV$_{CC}$ = 4.5 V to 5.5 V, Vref = 4.5 V to AV_{CC}, V_{SS} = PWMV$_{SS}$ = AV_{SS} = 0 V,
Iϕ = 8 MHz to 40 MHz, Pϕ = 8 MHz to 20 MHz,
T_a = 0°C to +75°C (regular specifications),
T_a = 0°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Condition
Programming time[1][2][4]	t_P	—	3	30	ms/128 bytes	
Erase time[1][2][4]	t_E	—	80	800	ms/4-kbyte block	
		—	500	5000	ms/32-kbyte block	
		—	1000	10000	ms/64-kbyte block	
Programming time (total) [1][2][4]	Σt_P	—	5	15	s/256 kbytes	T_a = 25°C, memory filled with 0.
Erase time (total) [1][2][4]	Σt_E	—	5	15	s/256 kbytes	T_a = 25°C
Programming/erase time (total) [1][2][4]	Σt_{PE}	—	10	30	s/256 kbytes	T_a = 25°C
Number of programming	N_{WEC}	100[3]	—	—	Times	
Data retention time[4]	t_{DRP}	10	—	—	Year	

Notes: 1. Programming time and erase time depend on data in the flash memory.
2. Programming time and erase time do not include time for data transfer.
3. All the characteristics after programming are guaranteed within this value (guaranteed value is from 1 to Min. value).
4. Characteristics when programming is performed within the Min. value.

RENESAS

Appendix

A. Port States in Each Pin State

Table A.1 Port States in Each Pin State

Port Name	Pin Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode	
					OPE = 1	OPE = 0
PORT1	P1[0:7]	All	Hi-Z	Hi-Z	Keep	
PORT2	P2[0:7]	All	Hi-Z	Hi-Z	Keep	
PORT3	P3[0:7]	All	Hi-Z	Hi-Z	Keep	
PORT4	P4[0:7]	All	Hi-Z	Hi-Z	Keep	
PORT5	P5[0:5]	All	Hi-Z	Hi-Z	Keep	
	P56	All	Hi-Z	Hi-Z	[DAOE0 = 1] Keep [DAOE0 = 0] Hi-Z	
	P57	All	Hi-Z	Hi-Z	[DAOE1 = 1] Keep [DAOE1 = 0] Hi-Z	
PORT6	P6[0:7]	All	Hi-Z	Hi-Z	Keep	
PORTA	PA[0:2]	All	Hi-Z	Hi-Z	Keep	
	PA[3:5]	Single chip mode (EXPE = 0)	Hi-Z	Hi-Z	Keep	
		External extended mode (EXPE = 1)	H	Hi-Z	H	Hi-Z
	PA6	Single chip mode (EXPE = 0)	Hi-Z	Hi-Z	[$\overline{\text{AS}}$ output] H [Other than above] Keep	[$\overline{\text{AS}}$ output] Hi-Z [Other than above] Keep
		External extended mode (EXPE = 1)	H	Hi-Z		
	PA7	Single chip mode (EXPE = 0)	Hi-Z	Hi-Z	[Clock output] H [Other than above] Keep	[Clock output] Clock output [Other than above] Keep
		External extended mode (EXPE = 1)	Clock output	Hi-Z		

RENESAS

Port Name	Pin Name	MCU Operating Mode		Reset	Hardware Standby Mode	Software Standby Mode OPE = 1	OPE = 0
PORTD	PD[0:7]	External extended mode (EXPE = 1)		L	Hi-Z	Keep	Hi-Z
		ROM enabled extended mode		Hi-Z	Hi-Z	Keep	[Address output] Hi-Z [Other than above] Keep
		Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
PORTE	PE[0:7]	External extended mode (EXPE = 1)		L	Hi-Z	Keep	Hi-Z
		ROM enabled extended mode		Hi-Z	Hi-Z	Keep	[Address output] Hi-Z [Other than above] Keep
		Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
PORTF	PF[0:7]	External extended mode (EXPE = 1)		L/Hi-Z	Hi-Z	Keep	[Address output] Hi-Z [Other than above] Keep
		Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
PORTH	PH[0:7]	Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
		External extended mode (EXPE = 1)		Hi-Z	Hi-Z	Keep	
PORTI	PI[0:7]	Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
		External extended mode (EXPE = 1)	8-bit bus mode	Hi-Z	Hi-Z	Keep	
			16-bit bus mode	Hi-Z	Hi-Z	Keep	
PORTJ	PJ[0:7]	Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
		External extended mode (EXPE = 1)		Hi-Z	Hi-Z	Keep	
PORTK	PK[0:7]	Single chip mode (EXPE = 0)		Hi-Z	Hi-Z	Keep	
		External extended mode (EXPE = 1)		Hi-Z	Hi-Z	Keep	

Legend:

Hi-Z: High impedance

H: High level

L: Low level

Keep: The input pin becomes high-impedance state and the output pin retains the state.

OPE: Output port enable

RENESAS

B. Product Lineup

Product Classification	Product Model	Marking	Package (Package Code)
H8SX/1544	R5F61544	R5F61544	LQFP-144 (FP-144L)
H8SX/1543	R5F61543	R5F61543	LQFP-144 (FP-144L)

C.   Package Dimensions

For the package dimensions, data in the Renesas IC Package General Catalog has priority.

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-LQFP144-20x20-0.50	PLQP0144KA-A	144P6Q-A / FP-144L / FP-144LV	1.2g

NOTE)
1. DIMENSIONS *1* AND *2* DO NOT INCLUDE MOLD FLASH.
2. DIMENSION *3* DOES NOT INCLUDE TRIM OFFSET.

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	19.9	20.0	20.1
E	19.9	20.0	20.1
A_2	—	1.4	—
H_D	21.8	22.0	22.2
H_E	21.8	22.0	22.2
A	—	—	1.7
A_1	0.05	0.1	0.15
b_p	0.17	0.22	0.27
b_1	—	0.20	—
c	0.09	0.145	0.20
c_1	—	0.125	—
θ	0°	—	8°
e	—	0.5	—
x	—	—	0.08
y	—	—	0.10
Z_D	—	1.25	—
Z_E	—	1.25	—
L	0.35	0.5	0.65
L_1	—	1.0	—

Terminal cross section

Detail F

Figure C.1   Package Dimensions (FP-144L)

RENESAS

Index

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

RENESAS

W

H8SX/1544 Group
Hardware Manual