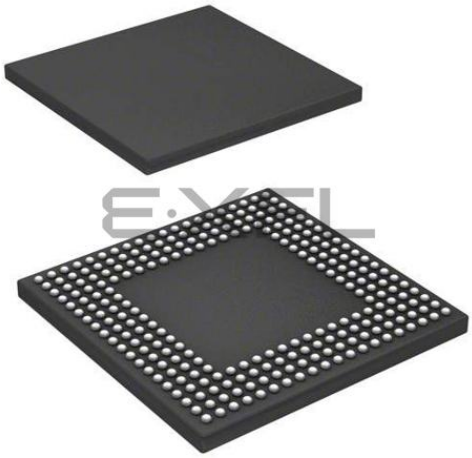

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Last Time Buy
Core Processor	SH-3 DSP
Core Size	32-Bit Single-Core
Speed	200MHz
Connectivity	EBI/EMI, Ethernet, FIFO, SCI, SIO
Peripherals	DMA, POR, WDT
Number of I/O	24
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	1.4V ~ 1.6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-20°C ~ 75°C (TA)
Mounting Type	Surface Mount
Package / Case	256-LFBGA
Supplier Device Package	256-LFBGA (17x17)
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/hd6417712bpv

2.6 Instruction Set

2.6.1 CPU Instruction Set Based on Functions

The CPU instruction set consists of 68 basic instruction types divided into six functional groups, as shown in table 2.5. Tables 2.6 to 2.11 show the instruction notation, machine code, execution time, and function.

Table 2.5 CPU Instruction Types

Type	Kinds of Instruction	Op Code	Function	Number of Instructions
Data transfer instructions	5	MOV	Data transfer	39
			Immediate data transfer	
			Peripheral module data transfer	
			Structure data transfer	
		MOVA	Effective address transfer	
		MOVT	T bit transfer	
		SWAP	Upper/lower swap	
		XTRCT	Extraction of middle of linked registers	
Arithmetic operation instructions	21	ADD	Binary addition	33
		ADDC	Binary addition with carry	
		ADDV	Binary addition with overflow check	
		CMP/cond	Comparison	
		DIV1	Division	
		DIV0S	Signed division initialization	
		DIV0U	Unsigned division initialization	
		DMULS	Signed double-precision multiplication	
		DMULU	Unsigned double-precision multiplication	
		DT	Decrement and test	
		EXTS	Sign extension	
		EXTU	Zero extension	

Table 2.8 Logic Operation Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
AND	Rm,Rn	0010nnnnmmmm1001	Rn & Rm → Rn	—	1	—
AND	#imm,R0	11001001iiiiiiii	R0 & imm → R0	—	1	—
AND.B	#imm,@(R0, GBR)	11001101iiiiiiii	(R0+GBR) & imm → (R0+GBR)	—	3	—
NOT	Rm,Rn	0110nnnnmmmm0111	~Rm → Rn	—	1	—
OR	Rm,Rn	0010nnnnmmmm1011	Rn | Rm → Rn	—	1	—
OR	#imm,R0	11001011iiiiiiii	R0 | imm → R0	—	1	—
OR.B	#imm,@(R0, GBR)	11001111iiiiiiii	(R0+GBR) | imm → (R0+GBR)	—	3	—
TAS.B	@Rn	0100nnnn00011011	If (Rn) is 0, 1 → T; 1 → MSB of (Rn)	—	4	Test result
TST	Rm,Rn	0010nnnnmmmm1000	Rn & Rm; if the result is 0, 1 → T	—	1	Test result
TST	#imm,R0	11001000iiiiiiii	R0 & imm; if the result is 0, 1 → T	—	1	Test result
TST.B	#imm,@(R0, GBR)	11001100iiiiiiii	(R0 + GBR) & imm; if the result is 0, 1 → T	—	3	Test result
XOR	Rm,Rn	0010nnnnmmmm1010	Rn ^ Rm → Rn	—	1	—
XOR	#imm,R0	11001010iiiiiiii	R0 ^ imm → R0	—	1	—
XOR.B	#imm,@(R0, GBR)	11001110iiiiiiii	(R0+GBR) ^ imm → (R0+GBR)	—	3	—

3.5 DSP Data Operation Instructions

3.5.1 DSP Registers

This LSI has eight data registers (A0, A1, X0, X1, Y0, Y1, M0 and M1) and one control register (DSR) as DSP registers (figure 3.3).

Four kinds of operation access the DSP data registers. The first is DSP data processing. When a DSP fixed-point data operation uses A0 or A1 as the source register, it uses the guard bits (bits 39 to 32). When it uses A0 or A1 as the destination register, guard bits 39 to 32 are valid. When a DSP fixed-point data operation uses a DSP register other than A0 or A1 as the source register, it sign-extends the source value to bits 39 to 32. When it uses one of these registers as the destination register, bits 39 to 32 of the result are discarded.

The second kind of operation is an X or Y data transfer operation, MOVX.W, MOVY.W. This operation accesses the X and Y memories through the 16-bit X and Y data buses (figure 3.4). The register to be loaded or stored by this operation always comprises the upper 16 bits (bits 31 to 16). X0 or X1 can be the destination of an X memory load and Y0 or Y1 can be the destination of a Y memory load, but no other register can be the destination register in this operation. When data is read into the upper 16 bits of a register (bits 31 to 16), the lower 16 bits of the register (bits 15 to 0) are automatically cleared. A0 and A1 can be stored in the X or Y memory by this operation, but no other registers can be stored.

The third kind of operation is a single-data transfer instruction, MOVS.W or MOVS.L. These instructions access any memory location through the LDB (figure 3.4). All DSP registers connect to the LDB and can be the source or destination register of the data transfer. These instructions have word and longword access modes. In word mode, registers to be loaded or stored by this instruction comprise the upper 16 bits (bits 31 to 16) for DSP registers except A0G and A1G. When data is loaded into a register other than A0G and A1G in word mode, the lower half of the register is cleared. When A0 or A1 is used, the data is sign-extended to bits 39 to 32 and the lower half is cleared. When A0G or A1G is the destination register in word mode, data is loaded into an 8-bit register, but A0 or A1 is not cleared. In longword mode, when the destination register is A0 or A1, it is sign-extended to bits 39 to 32.

The fourth kind of operation is system control instructions such as LDS, STS, LDS.L, or STS.L. The DSR, A0, X0, X1, Y0, and Y1 registers of the DSP register can be treated as system registers. For these registers, data transfer instructions between the CPU general registers and system registers or memory access instructions are supported.

SPC Saved by Exception in Repeat Control Period: If an exception is accepted in the repeat control period while the repeat counter (RC[11:0]) in the SR register is two or greater, the program counter to be saved may not indicate the value to be returned correctly. To execute the repeat control after returning from an exception processing, the return address must indicate an instruction prior to a repeat detection instruction. Accordingly, if an exception is accepted in repeat control period, an exception other than re-execution type exception by a repeat detection instruction cannot return to the repeat control correctly.

Table 4.3 SPC Value when Re-Execution Type Exception Occurs in Repeat Control (RC[11:0] ≥ 2)

Instruction where Exception Occurs	Number of Instructions in Repeat Loop			
	1	2	3	4 or Greater
RptDtct	RptDtct	RptDtct	RptDtct	RptDtct
RptDtct1	RptDtct1	RptDtct1	RptDtct1	RptDtct1
RptDtct2	—	RptDtct1	RptDtct1	RS-4
RptDtct3	—	—	RptDtct1	RS-2

Note: The following labels are used here.
RptDtct: Repeat detection instruction address
RptDtct1: An instruction address one instruction following the repeat detection instruction
RptDtct2: An instruction address two instruction following the repeat detection instruction
RptDtct3: An instruction address three instruction following the repeat detection instruction
RS: Repeat start instruction address
If a re-execution type exception is accepted at an instruction in the hatched areas above, a return address to be saved in the SPC is incorrect. If RC[11:0] is 1 or 0, a correct return address is saved in the SPC.

Illegal Instruction Exception in Repeat Control Period: If one of the following instructions is executed at the address following RptDtct1, a general illegal instruction exception occurs. For details on an address to be saved in the SPC, refer to the description in section 4.4.3, Exception in Repeat Control Period.

- Branch instructions
  BRA, BSR, BT, BF, BT/S, BF/S, BSRF, RTS, BRAF, RTE, JSR, JMP, TRAPA
- Repeat control instructions
  SETRC, LDRS, LDRE
- Load instructions for SR, RS, and RE
  LDC Rn, SR, LDC @Rn+, SR, LDC Rn, RE, LDC @Rn+, RE, LDC Rn,RS, LDC @Rn+, Rs

Table 5.1 Access States Designated by D, C, and PR Bits

		Privileged Mode		User Mode	
		Reading	**Writing**	**Reading**	**Writing**
D bit	0	Permitted	Initial page write exception	Permitted	Initial page write exception
	1	Permitted	Permitted	Permitted	Permitted
C bit	0	Permitted (no caching)	Permitted (no caching)	Permitted (no caching)	Permitted (no caching)
	1	Permitted (with caching)	Permitted (with caching)	Permitted (with caching)	Permitted (with caching)
PR bit	00	Permitted	TLB protection violation exception	TLB protection violation exception	TLB protection violation exception
	01	Permitted	Permitted	TLB protection violation exception	TLB protection violation exception
	10	Permitted	TLB protection violation exception	Permitted	TLB protection violation exception
	11	Permitted	Permitted	Permitted	Permitted

8.4.7 Interrupt Request Register 3 (IRR3)

IRR3 is an 8-bit register that indicates whether interrupt requests from the RTC are generated. This register is initialized to H'00 by a power-on reset or manual reset, but is not initialized in standby mode.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
2	CUIR	0	R	CUI Interrupt Request Indicates whether the CUI (RTC) interrupt request is generated. 0: CUI interrupt request is not generated 1: CUI interrupt request is generated
1	PRIR	0	R	PRI Interrupt Request Indicates whether the PRI (RTC) interrupt request is generated. 0: PRI interrupt request is not generated 1: PRI interrupt request is generated
0	ATIR	0	R	ATI Interrupt Request Indicates whether the ATI (RTC) interrupt request is generated. 0: ATI interrupt request is not generated 1: ATI interrupt request is generated

Bit	Bit Name	Initial Value	R/W	Description
3	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.
2	EINT2R	0	R	EINT2 Interrupt Request Indicates whether the EINT2 (E-DMAC) interrupt request is generated. 0: EINT2 interrupt request is not generated 1: EINT2 interrupt request is generated Note: This bit is reserved in the SH7713. It is always read as 0 and the write value should always be 0.
1	EINT1R	0	R	EINT1 Interrupt Request Indicates whether the EINT1 (E-DMAC) interrupt request is generated. 0: EINT1 interrupt request is not generated 1: EINT1 interrupt request is generated Note: This bit is reserved in the SH7713. It is always read as 0 and the write value should always be 0.
0	EINT0R	0	R	EINT0 Interrupt Request Indicates whether the EINT0 (E-DMAC) interrupt request is generated. 0: EINT0 interrupt request is not generated 1: EINT0 interrupt request is generated

8.5.2 Multiple Interrupts

When handling multiple interrupts, an interrupt handler should include the following procedures:

1. Branch to a specific interrupt handler corresponding to a code set in INTEVT or INTEVT2. The code in INTEVT or INTEVT2 can be used as an offset for branching to the specific handler.
2. Clear the interrupt source in each specific handler.
3. Save SSR and SPC to memory.
4. Clear the BL bit in SR, and set the accepted interrupt level in the interrupt mask bits in SR.
5. Handle the interrupt.
6. Execute the RTE instruction.

When these procedures are followed in order, an interrupt of higher priority than the one being handled can be accepted after clearing the BL bit in step 4. See figure 8.3 on a sample interrupt operation flowchart.

Bit	Bit Name	Initial Value	R/W	Description
6 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.

- CS3WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 15	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
14	WTRP1	0	R/W	Number of Wait Cycles Waiting Completion of Precharge Specify the number of minimum wait cycles to be inserted to wait the completion of precharge. The setting for areas 2 and 3 is common. (1) From starting auto-charge to issuing the ACTV command for the same bank (2) From issuing the PRE/PALL command to issuing the ACTV command for the same bank (3) To transiting to power-down mode/deep power-down mode (4) From issuing the PALL command at auto-refresh to issuing the REF command (5) From issuing the PALL command at self-refresh to issuing the SELF command 00: 0 cycle 01: 1 cycles 10: 2 cycles 11: 3 cycles
13	WTRP0	0	R/W	
12	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
4	TRWL1	0	R/W	Number of Wait Cycles Waiting Start of Precharge
3	TRWL0	0	R/W	Specify the number of minimum wait cycles to be inserted to wait the start of precharge. The setting for areas 2 and 3 is common. (1) This LSI is in non-bank active mode from the issue of the WRITA command to the start of auto-precharge in SDRAM, and issues the ACTV command for the same bank after issuing the WRITA command. Confirm how many cycles are required from the reception of the WRITA command to the start of auto-precharge in each SDRAM data sheet. Set this bit so that the number of cycles is not above the cycles specified by this bit. (2) This LSI is in bank active mode from issuing the WRIT command to issuing the PRE command, and the access to different row address in the same bank is performed. 00: 0 cycle 01: 1 cycle 10: 2 cycles 11: 3 cycles
2	—	0	R	Reserved This bit is always read as 0. The write value should always be 0.

Table 12.12 Relationship between A2/3BSZ[1:0], A2/3ROW[1:0], A2/3COL[1:0], and Address Multiplex Output (1)-1

Setting				
A2/3 BSZ [1:0]	**A2/3 ROW [1:0]**	**A2/3 COL [1:0]**		
11 (32 bits)	**00 (11 bits)**	**00 (8 bits)**		
Output Pin of This LSI	**Row Address Output**	**Column Address Output**	**Synchronous DRAM Pin**	**Function**
A17	A25	A17		Unused
A16	A24	A16		
A15	A23	A15		
A14	A22*2	A22*2	A12 (BA1)*3	Specifies bank
A13	A21*2	A21*2	A11 (BA0)	
A12	A20	L/H*1	A10/AP	Specifies address/precharge
A11	A19	A11	A9	Address
A10	A18	A10	A8	
A9	A17	A9	A7	
A8	A16	A8	A6	
A7	A15	A7	A5	
A6	A14	A6	A4	
A5	A13	A5	A3	
A4	A12	A4	A2	
A3	A11	A3	A1	
A2	A10	A2	A0	

Bit	Bit Name	Initial Value	R/W	Descriptions
0	DE	0	R/W	DMA Enable Enables or disables the DMA transfer. In auto-request mode, DMA transfer starts by setting the DE bit and DME bit in DMAOR to 1. In this time, all of the bits TE, NMIF in DMAOR, and AE in DMAOR must be 0. In an external request or peripheral module request, DMA transfer starts if DMA transfer request is generated by the devices or peripheral modules after setting the bits DE and DME to 1. In this case, however, all of the bits TE, NMIF, and AE must be 0 as in the case of auto-request mode. Clearing the DE bit to 0 can terminate the DMA transfer. 0: DMA transfer disabled 1: DMA transfer enabled

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

13.3.5 DMA Operation Register (DMAOR)

DMAOR is a 16-bit readable/writable register that specifies the priority level of channels at the DMA transfer. This register indicates the DMA transfer status.

DMAOR is initialized to H′0000 at a reset and retains the current value in standby or module standby mode.

Bit	Bit Name	Initial Value	R/W	Description
15 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
9	PR1	0	R/W	Priority Mode
8	PR0	0	R/W	Select the priority level between channels when there are transfer requests for multiple channels simultaneously. 00: Fixed mode 1: CH0 > CH1 > CH2 > CH3 > CH4 > CH5 01: Fixed mode 2: CH0 > CH2 > CH3 > CH1 > CH4 > CH5 10: Reserved (setting prohibited) 11: All channel round-robin mode

16.4 Operation

16.4.1 Overview

The SCIF can carry out serial communication in asynchronous mode, in which synchronization is achieved character by character, and in clock synchronous mode, in which synchronization is achieved with clock pulses.

16-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed. Also, the $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ signals are included as modem control signals. Transfer format is selected by SCSMR. This is shown in table 16.3. The SCIF clock source is determined by the combination of the C/$\overline{\text{A}}$ bit in SCSMR and the CKE1 and CKE0 bits in SCSCR. This is shown in table 16.4.

Asynchronous Mode

- Data length: Choice of 7 or 8 bits
- Choice of parity addition and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing errors, parity errors, overrun errors, receive-FIFO-data-full state, receive-data-ready state, and breaks, during reception
- Indication of the number of data bytes stored in the transmit and receive FIFO registers
- The SCIF clock source: Choice of internal or external clock

  When internal clock is selected: The SCIF operates on the baud rate generator clock.

  When external clock is selected: Clock with frequency16 times the bit rate must be input. (The on-chip baud rate generator is not used.)

Clock synchronous Mode

- Transfer format: Fixed to 8-bit data
- Detection of overrun errors during reception
- The SCIF clock source: Choice of internal or external clock

  When internal clock is selected: The SCIF operates on the baud rate generator clock and outputs the synchronous clock

  When external clock is selected: The SCIF operates on the input synchronous clock. The on-chip baud rate generator is not used.

Bit	Bit Name	Initial Value	R/W	Description
24	TEN7	0	R/W	CAM Entry Table 7 (TSU_ADRH7 and TSU_ADRL7) Setting 0: Disabled 1: Enabled
23	TEN8	0	R/W	CAM Entry Table 8 (TSU_ADRH8 and TSU_ADRL8) Setting 0: Disabled 1: Enabled
22	TEN9	0	R/W	CAM Entry Table 9 (TSU_ADRH9 and TSU_ADRL9) Setting 0: Disabled 1: Enabled
21	TEN10	0	R/W	CAM Entry Table 10 (TSU_ADRH10 and TSU_ADRL10) Setting 0: Disabled 1: Enabled
20	TEN11	0	R/W	CAM Entry Table 11 (TSU_ADRH11 and TSU_ADRL11) Setting 0: Disabled 1: Enabled
19	TEN12	0	R/W	CAM Entry Table 12 (TSU_ADRH12 and TSU_ADRL12) Setting 0: Disabled 1: Enabled
18	TEN13	0	R/W	CAM Entry Table 13 (TSU_ADRH13 and TSU_ADRL13) Setting 0: Disabled 1: Enabled
17	TEN14	0	R/W	CAM Entry Table 14 (TSU_ADRH14and TSU_ADRL14) Setting 0: Disabled 1: Enabled

19.2.14 Receive Buffer Write Address Register (RBWAR)

RBWAR stores the address of data to be written in the receiving buffer when the E-DMAC writes data to the receiving buffer. Which addresses in the receiving buffer are processed by the E-DMAC can be recognized by monitoring addresses displayed in this register. The address that the E-DMAC is actually processing may be different from the value read from this register.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	RBWA31 to RBWA0	All 0	R	Receiving-Buffer Write Address These bits can only be read. Writing is prohibited.

19.2.15 Receive Descriptor Fetch Address Register (RDFAR)

RDFAR stores the descriptor start address that is required when the E-DMAC fetches descriptor information from the receiving descriptor. Which receiving descriptor information is used for processing by the E-DMAC can be recognized by monitoring addresses displayed in this register. The address from which the E-DMAC is actually fetching a descriptor may be different from the value read from this register.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	RDFA31 to RDFA0	All 0	R	Receiving-Descriptor Fetch Address These bits can only be read. Writing is prohibited.

19.2.16 Transmit Buffer Read Address Register (TBRAR)

TBRAR stores the address of the transmission buffer when the E-DMAC reads data from the transmission buffer. Which addresses in the transmission buffer are processed by the E-DMAC can be recognized by monitoring addresses displayed in this register. The address from which the E-DMAC is actually reading in the buffer may be different from the value read from this register.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	TBRA31 to TBRA0	All 0	R	Transmission-Buffer Read Address These bits can only be read. Writing is prohibited.

(1) Countermeasure

This problem occurs under this condition:
size of transmit FIFO set in the FIFO depth register (FDR) ≤ maximum length of frame for transmission (1518 bytes).

To avoid this problem, the size of transmit FIFO set in the FIFO depth register (FDR) should be 2 kbytes (or 1792 bytes), and also the transmit FIFO threshold register (TFTR) should be set to the 'store and forward modes', in use.

(2) Countermeasure for the case where the software handles transmission without the aid of TC interrupts

The countermeasure described under (a), Processing transmission without handling of the frame transmission complete (TC) interrupt, below, is based on the method explained in the description of bit 21 in (1) Countermeasure of section 19.4.4, Usage Notes on SH-Ether EtherC/E-DMAC Status Register (EESR).

23.4.2 Reset Configuration

Table 23.4 Reset Configuration

$\overline{\text{ASEMD0}}$*1	$\overline{\text{RESETP}}$	$\overline{\text{TRST}}$	Chip State
H	L	L	Normal reset and H-UDI reset*4
		H	Normal reset*4
	H	L	H-UDI reset only
		H	Normal operation
L	L	L	Reset hold*2
		H	In ASE user mode*3: Normal reset In ASE break mode*3: $\overline{\text{RESETP}}$ assertion is masked
	H	L	H-UDI reset only
		H	Normal operation

Notes: 1. Performs normal mode and ASE mode settings
$\overline{\text{ASEMD0}}$ = H, normal mode
$\overline{\text{ASEMD0}}$ = L, ASE mode

2. In ASE mode, reset hold is enabled by driving the $\overline{\text{RESETP}}$ and $\overline{\text{TRST}}$ pins low for a constant cycle. In this state, the CPU does not activate, even if $\overline{\text{RESETP}}$ is driven high. When $\overline{\text{TRST}}$ is driven high, H-UDI operation is enabled, but the CPU does not activate. The reset hold state is canceled by the following conditions:
   - Another $\overline{\text{RESETP}}$ assertion (power-on reset)
   - $\overline{\text{TRST}}$ reassertion
3. ASE mode is classified into two modes; ASE break mode to execute the firmware program of an emulator and ASE user mode to execute the user program.
4. Make sure the $\overline{\text{TRST}}$ pin is low when the power is turned on.

23.4.3 TDO Output Timing

The timing of data output from the TDO is switched by the command type set in the SDIR. The timing changes at the TCK falling edge when JTAG commands (EXTEST, CLAMP, HIGHZ, SAMPLE/PRELOAD, IDCODE, and BYPASS) are set. This is a timing of the JTAG standard. When the H-UDI commands (H-UDI reset negate, H-UDI reset assert, and H-UDI interrupt) are set, TDO is output at the TCK rising edge earlier than the JTAG standard by a half cycle.

Abbreviation	Module*1	Bus*2	Address	Size (bit)	Access Size (bit)*3
TSU_ADRH7	EtherC(TSU) (SH7710, SH7712)	I	H'A700 0938	32	32
TSU_ADRL7		I	H'A700 093C	32	32
TSU_ADRH8		I	H'A700 0940	32	32
TSU_ADRL8		I	H'A700 0944	32	32
TSU_ADRH9		I	H'A700 0948	32	32
TSU_ADRL9		I	H'A700 094C	32	32
TSU_ADRH10		I	H'A700 0950	32	32
TSU_ADRL10		I	H'A700 0954	32	32
TSU_ADRH11		I	H'A700 0958	32	32
TSU_ADRL11		I	H'A700 095C	32	32
TSU_ADRH12		I	H'A700 0960	32	32
TSU_ADRL12		I	H'A700 0964	32	32
TSU_ADRH13		I	H'A700 0968	32	32
TSU_ADRL13		I	H'A700 096C	32	32
TSU_ADRH14		I	H'A700 0970	32	32
TSU_ADRL14		I	H'A700 0974	32	32
TSU_ADRH15		I	H'A700 0978	32	32
TSU_ADRL15		I	H'A700 097C	32	32
TSU_ADRH16		I	H'A700 0980	32	32
TSU_ADRL16		I	H'A700 0984	32	32
TSU_ADRH17		I	H'A700 0988	32	32
TSU_ADRL17		I	H'A700 098C	32	32
TSU_ADRH18		I	H'A700 0990	32	32
TSU_ADRL18		I	H'A700 0994	32	32
TSU_ADRH19		I	H'A700 0998	32	32
TSU_ADRL19		I	H'A700 099C	32	32
TSU_ADRH20		I	H'A700 09A0	32	32
TSU_ADRL20		I	H'A700 09A4	32	32
TSU_ADRH21		I	H'A700 09A8	32	32
TSU_ADRL21		I	H'A700 09AC	32	32

Register Abbreviation	Power-on Reset*[1]	Manual Reset*[1]	Software standby	Module Standby	Sleep	Module
RMFCRn (n = 0, 1) (SH7710, SH7712) RMFCR (SH7713)	Initialized	Initialized	Retained	—	Retained	E-DMAC
TFTRn (n = 0, 1) (SH7710, SH7712) TFTR (SH7713)	Initialized	Initialized	Retained	—	Retained	
FDRn (n = 0, 1) (SH7710, SH7712) FDR (SH7713)	Initialized	Initialized	Retained	—	Retained	
RMCRn (n = 0, 1) (SH7710, SH7712) RMCR (SH7713)	Initialized	Initialized	Retained	—	Retained	
EDOCRn (n = 0, 1) (SH7710, SH7712) EDOCR (SH7713)	Initialized	Initialized	Retained	—	Retained	
RBWARn (n = 0, 1) (SH7710, SH7712) RBWAR (SH7713)	Initialized	Initialized	Retained	—	Retained	

Item	Page	Revision (See Manual for Details)
19.2.3 E-DMAC Receive Request Register (EDRRR)	755	Added Note: * If the receive function is disabled during frame reception, write-back is not performed successfully to the receive descriptor. Following pointers to read a receive descriptor become abnormal and the E-DMAC can not operate successfully. In this case, to make the E-DMAC reception enabled again, execute a software reset by the SWR bit in EDMR0 (EDMR1). To make the E-DMAC reception disabled without executing a software reset, specify the RE bit in ECMR0 (ECMR1). Next, after the E_DMAC has completed the reception and write-back to the receive descriptor has been confirmed, disable the receive function of this register. Note that EDMR is provided instead of EDMR0 (EDMR1) for the SH7713.
19.2.6 EtherC/E-DMAC Status Register (EESR)	756	Amended and added In the SH7710 and SH7712, the interrupts generated by this register are EINT0 for channel 0 and EINT1 for channel 1. In the SH7713, the interrupt generated by this register is EINT0. For interrupt priorities, see section 8, Interrupt Controller (INTC), and section 8.3.5, Interrupt Exception Handling and Priority. The EINT2, in the SH7710 or SH7712, is an interrupt generated by the TSU_FNSR in the EtherC.
19.3.1 Descriptors and Descriptor List	776	Amended and added In the SH7710 and SH7712, the E-DMAC consists of two systems: system 0 and system 1. The DMAC for transmission and the DMAC for reception operate independently of each other, and the DMAC for system 0 and the DMAC for system 1 operate independently of each other. For normal E-DMAC operation, place descriptors for transmission and reception and descriptors for system 0 and system 1 in those address spaces that do not overlap. In the SH7713, the E-DMAC consists of one system. The DMAC for transmission and the DMAC for reception operate independently of each other. For normal E-DMAC operation, place descriptors for transmission and reception in those address spaces that do not overlap.