



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	Active
Core Processor	SH-2
Core Size	32-Bit Single-Core
Speed	125MHz
Connectivity	EBI/EMI, Ethernet, FIFO, SCI, SIO
Peripherals	DMA, POR, WDT
Number of I/O	78
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	16K x 8
Voltage - Supply (Vcc/Vdd)	1.71V ~ 1.89V
Data Converters	-
Oscillator Type	Internal
Operating Temperature	-20°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	176-UFBGA
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/ds76191n125bgv

2.2.1 General Registers (Rn)

There are sixteen 32-bit general registers (Rn), designated R0 to R15. The general registers are used for data processing and address calculation. R0 is also used as an index register. With a number of instructions, R0 is the only register that can be used. R15 is used as a hardware stack pointer (SP). In exception handling, R15 is used for accessing the stack to save or restore the status register (SR) and program counter (PC) values.

2.2.2 Control Registers

There are three 32-bit control registers, designated status register (SR), global base register (GBR), and vector base register (VBR). SR indicates a processing state. GBR is used as a base address in GBR indirect addressing mode for data transfer of on-chip peripheral module registers. VBR is used as a base address of the exception handling (including interrupts) vector table.

- Status register (SR)

Bit	Bit name	Default	Read/Write	Description
31 to 10	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
9	M	Undefined	R/W	Used by the DIV0U, DIV0S, and DIV1 instructions.
8	Q	Undefined	R/W	Used by the DIV0U, DIV0S, and DIV1 instructions.
7	I3	1	R/W	Interrupt Mask
6	I2	1	R/W	
5	I1	1	R/W	
4	I0	1	R/W	
3, 2	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
1	S	Undefined	R/W	S Used by the multiply and accumulate instruction.

Table 2.7 Access with Displacement


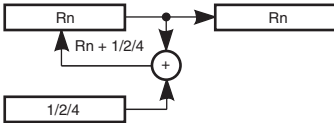
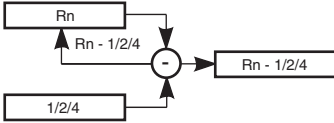
Type	CPU in this LSI	Example of Other CPUs
16-bit displacement	MOV.W @(disp,PC),R0	MOV.W
	MOV.W @(R0,R1),R2	@(H'1234,R1),R
	2
	.DATA.W H'1234	

Note: * Immediate data is referenced by @(disp,PC).

2.4.2 Addressing Modes

Table 2.8 lists addressing modes and effective address calculation methods.

Table 2.8 Addressing Modes and Effective Addresses

Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
Register direct	Rn	Effective address is register Rn. (Operand is register Rn contents.)	—
Register indirect	@Rn	Effective address is register Rn contents. 	Rn
Register indirect with post-increment	@Rn+	Effective address is register Rn contents. A constant is added to Rn after instruction execution: 1 for a byte operand, 2 for a word operand, and 4 for a longword operand. 	Rn After instruction execution Byte: $Rn + 1 \rightarrow Rn$ Word: $Rn + 2 \rightarrow Rn$ Longword: $Rn + 4 \rightarrow Rn$
Register indirect with pre-decrement	@-Rn	Effective address is register Rn contents, decremented by a constant beforehand: 1 for a byte operand, 2 for a word operand, 4 for a longword operand. 	Byte: $Rn - 1 \rightarrow Rn$ Word: $Rn - 2 \rightarrow Rn$ Longword: $Rn - 4 \rightarrow Rn$ (Instruction executed with Rn after calculation)

Bit	Bit Name	Initial Value	R/W	Description
1	IRQ1F	0	R/W	<p>Indicates the status of an IRQ1 interrupt request.</p> <ul style="list-style-type: none"> When level detection mode is selected <p>0: An IRQ1 interrupt has not been detected [Clearing condition] Driving pin IRQ1 high</p> <p>1: An IRQ1 interrupt has been detected [Setting condition] Driving pin IRQ1 low</p> <ul style="list-style-type: none"> When edge detection mode is selected <p>0: An IRQ1 interrupt has not been detected [Clearing conditions] — Writing 0 after reading IRQ1F = 1 — Accepting an IRQ1 interrupt</p> <p>1: An IRQ1 interrupt request has been detected [Setting condition] Detecting the specified edge of pin IRQ1</p>
0	IRQ0F	0	R/W	<p>Indicates the status of an IRQ0 interrupt request.</p> <ul style="list-style-type: none"> When level detection mode is selected <p>0: An IRQ0 interrupt has not been detected [Clearing condition] Driving pin IRQ0 high</p> <p>1: An IRQ0 interrupt has been detected [Setting condition] Driving pin IRQ0 low</p> <ul style="list-style-type: none"> When edge detection mode is selected <p>0: An IRQ0 interrupt has not been detected [Clearing conditions] — Writing 0 after reading IRQ0F = 1 — Accepting an IRQ0 interrupt</p> <p>1: An IRQ0 interrupt request has been detected [Setting condition] Detecting the specified edge of pin IRQ0</p>

Setting					Setting				
BSZ [1:0]	A3 ROW [1:0]	A3 COL [1:0]			BSZ [1:0]	A3 ROW [1:0]	A3 COL [1:0]		
11 (32 bits)	01 (12 bits)	01 (9 bits)			11 (32 bits)	01 (12 bits)	10 (10 bits)		
Output Pins of This LSI	Output Row Address	Output Column Address	Pins of SDRAM	Function	Output Pins of This LSI	Output Row Address	Output Column Address	Pins of SDRAM	Function
A13	A22	A13	A11	Address	A13	A23	A13	A11	Address
A12	A21	L/H* ¹	A10/AP	Specifies address/ precharge	A12	A22	L/H* ¹	A10/AP	Specifies address/ precharge
A11	A20	A11	A9	Address	A11	A21	A11	A9	Address
A10	A19	A10	A8		A10	A20	A10	A8	
A9	A18	A9	A7		A9	A19	A9	A7	
A8	A17	A8	A6		A8	A18	A8	A6	
A7	A16	A7	A5		A7	A17	A7	A5	
A6	A15	A6	A4		A6	A16	A6	A4	
A5	A14	A5	A3		A5	A15	A5	A3	
A4	A13	A4	A2		A4	A14	A4	A2	
A3	A12	A3	A1		A3	A13	A3	A1	
A2	A11	A2	A0		A2	A12	A2	A0	
A1	A10	A1		Unused	A1	A11	A1		Unused
A0	A9	A0			A0	A10	A0		
Example of memory connection					Example of memory connection				
One 256-Mbit product (2 Mwords x 32 bits x 4 banks, 9-bit column product)					One 512-Mbit product (4 Mwords x 32 bits x 4 banks, 10-bit column product)				
Two 128-Mbit products (2 Mwords x 16 bits x 4 banks, 9-bit column product)					Two 256-Mbit product (4 Mwords x 16 bits x 4 banks, 10-bit column product)				

Notes: 1. L/H is a bit used in the command specification; it is fixed low or high according to the access mode.

2. Bank address specification

Bit	Bit Name	Initial Value	R/W	Description
5	TE	0	R/W	<p>Transmission Enable</p> <p>If a frame is being transmitted when this bit is switched from transmit function enabled (TE = 1) to disabled (TE = 0), the transmit function will be enabled until transmission of the corresponding frame is completed.</p> <p>0: Transmit function is disabled</p> <p>1: Transmit function is enabled</p>
4	—	0	R	<p>Reserved</p> <p>This bit is always read as 0. The write value should always be 0.</p>
3	ILB	0	R/W	<p>Internal Loop Back Mode</p> <p>Specifies loopback mode in the EtherC.</p> <p>0: Normal data transmission/reception is performed.</p> <p>1: When DM = 1, data loopback is performed inside the MAC in the EtherC.</p>
2	ELB	0	R/W	<p>External Loop Back Mode</p> <p>This bit value is output directly to this LSI's general-purpose external output pin (EXOUT). This bit is used for loopback mode directives, etc., in the LSI, using the EXOUT pin. In order for LSI loopback to be implemented using this function, the LSI must have a pin corresponding to the EXOUT pin.</p> <p>0: Low-level output from the EXOUT pin</p> <p>1: High-level output from the EXOUT pin</p>
1	DM	0	R/W	<p>Duplex Mode</p> <p>Specifies the EtherC transfer method.</p> <p>0: Half-duplex transfer is specified</p> <p>1: Full-duplex transfer is specified</p>

Bit	Bit Name	Initial value	R/W	Description
21	TC	0	R/W	<p>Frame Transmit Complete</p> <p>Indicates that all the data specified by the transmit descriptor has been transmitted to the EtherC. The transfer status is written back to the relevant descriptor. For 1-frame/1-buffer processing, when 1-frame transmission is completed and the transmission descriptor valid bit (TACT) in the next descriptor is not set, transmission is completed and this bit is set to 1. Likewise, for multiple-frame buffer processing, when the last data in the frame is transmitted and the transmission descriptor valid bit (TACT) in the next descriptor is not set, transmission is completed and this bit is set to 1. After frame transmission, the E-DMAC writes the transmission status back to the descriptor.</p> <p>0: Transfer not complete, or no transfer directive 1: Transfer complete</p>
20	TDE	0	R/W	<p>Transmit Descriptor Empty</p> <p>Indicates that the transmission descriptor valid bit (TACT) in the descriptor is not set when the E-DMAC reads the transmission descriptor when the previous descriptor is not the last one of the frame for multiple-buffer frame processing. As a result, an incomplete frame may be transmitted.</p> <p>0: Transmit descriptor active bit TACT = 1 detected 1: Transmit descriptor active bit TACT = 0 detected</p> <p>When transmission descriptor empty (TDE = 1) occurs, execute a software reset and initiate transmission. In this case, the address that is stored in the transmit descriptor list address register (TDLAR) is transmitted first.</p>
19	TFUF	0	R/W	<p>Transmit FIFO Underflow</p> <p>Indicates that underflow has occurred in the transmit FIFO during frame transmission. Incomplete data is sent onto the line.</p> <p>0: Underflow has not occurred 1: Underflow has occurred</p>

- Bit 19 (TFUF): Transmit FIFO underflow interrupt source bit in EESR may not be set. When this bit is used as an interrupt source but is not set when it should be, the software is not notified of the interrupt. However, the upper layer will recognize the error in the form of an underflow of the transmit FIFO.
- Bit 16 (RFOF): Receive FIFO overflow interrupt source bit in EESR may not be set. Since the state of the interrupt source is written back to the relevant descriptor, check the receive descriptor (RD0) to confirm the error status.
- Bit 11 (CND), bit 10 (DLC), bit 9 (CD), bit 8 (TRO): The interrupt source bits in EESR for the carrier not detected, loss of carrier detected, delayed collision detected, and transmit retry over interrupts may not be set. However, since the states of the interrupt sources are written back to the relevant descriptor, check the transmit descriptor (TD0) to confirm the error status.

(2) Example of a countermeasure when the software configuration is based on the frame transmit complete interrupt

The following descriptions are of sample countermeasures for cases when software processing is based on the frame transmit complete interrupt (bit 21 (TC) in EESR).

If the TC interrupt source bit (bit 21) in EESR is not set on completion of transmission, the system will continue to wait for the TC interrupt, leading to stoppage of transmission. This situation arises when the interrupt handler writes a 1 to clear the bit. The sample method given as case (a) below takes the above possibility into account and avoids the problem by monitoring the transmit descriptor in interrupt processing for interrupts other than the TC interrupt.

The sample method given as case (b) below avoids the above problem by setting a timeout limit for retry processing when multiple transmit descriptors are in use.

Note: The countermeasure should be the one that best suits the structure of your driver and other software.

8. If counter i reaches or exceeds n , the maximum specified time has elapsed and we can judge that the E-DMAC has stopped due to a transmit underflow. Initialize the EtherC and E-DMAC modules by setting the software-reset bit SWR in the E-DMAC mode register (EDMR). After re-making initial settings for the Ethernet module, initialize the transmit/receive descriptors and transmit/receive buffers.

13.3 Register Descriptions

The DMAC has the following registers. See section 24, List of Registers, for the addresses of these registers and the state of them in each processing status. The SAR for channel 0 is expressed such as SAR_0.

Channel 0:

- DMA source address register_0 (SAR_0)
- DMA destination address register_0 (DAR_0)
- DMA transfer count register_0 (DMATCR_0)
- DMA channel control register_0 (CHCR_0)

Channel 1:

- DMA source address register_1 (SAR_1)
- DMA destination address register_1 (DAR_1)
- DMA transfer count register_1 (DMATCR_1)
- DMA channel control register_1 (CHCR_1)

Channel 2:

- DMA source address register_2 (SAR_2)
- DMA destination address register_2 (DAR_2)
- DMA transfer count register_2 (DMATCR_2)
- DMA channel control register_2 (CHCR_2)

Channel 3:

- DMA source address register_3 (SAR_3)
- DMA destination address register_3 (DAR_3)
- DMA transfer count register_3 (DMATCR_3)
- DMA channel control register_3 (CHCR_3)

Common:

- DMA operation register (DMAOR)
- DMA extended resource selector 0 (DMARS0)
- DMA extended resource selector 1 (DMARS1)

Address Modes:

- Dual Address Mode

In dual address mode, both the transfer source and destination are accessed by an address. The source and destination can be located externally or internally.

DMA transfer requires two bus cycles because data is read from the transfer source in a data read cycle and written to the transfer destination in a data write cycle. At this time, transfer data is temporarily stored in the DMAC. In the transfer between external memories as shown in figure 13.5, data is read to the DMAC from one external memory in a data read cycle, and then that data is written to the other external memory in a write cycle.

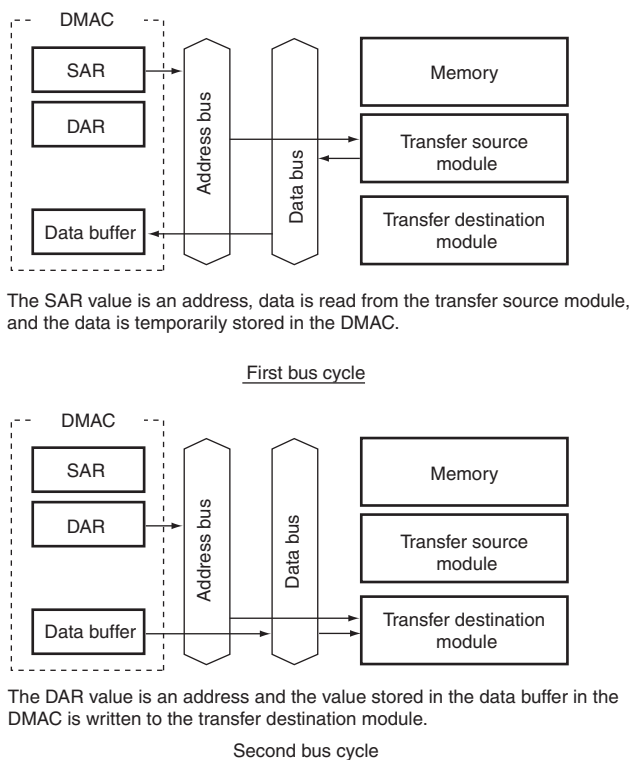


Figure 13.5 Data Flow of Dual Address Mode

Auto request, external request, and on-chip peripheral module request are available for the transfer request. DACK can be output in read cycle or write cycle in dual address mode. The channel control register (CHCR) can specify whether the DACK is output in read cycle or write cycle.

In serial transmission, the SCIF operates as described below.

1. When data is written into the transmit FIFO data register (SCFTDR), the SCIF transfers the data from SCFTDR to the transmit shift register (SCTSR) and starts transmitting. Confirm that the TDFE flag in the serial status register (SCFSR) is set to 1 before writing transmit data to SCFTDR. The number of data bytes that can be written is (16 – transmit trigger setting).
2. When data is transferred from SCFTDR to SCTSR and transmission is started, consecutive transmit operations are performed until there is no transmit data left in SCFTDR. When the number of transmit data bytes in SCFTDR falls below the transmit trigger number set in the FIFO control register (SCFCR), the TDFE flag is set. If the TIE bit in the serial control register (SCSR) is set to 1 at this time, a transmit-FIFO-data-empty interrupt (TXI) request is generated.

The serial transmit data is sent from the TxD pin in the following order.

- A. Start bit: One-bit 0 is output.
 - B. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
 - C. Parity bit: One parity bit (even or odd parity) is output. (A format in which a parity bit is not output can also be selected.)
 - D. Stop bit(s): One or two 1 bits (stop bits) are output.
 - E. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCIF checks the SCFTDR transmit data at the timing for sending the stop bit. If data is present, the data is transferred from SCFTDR to SCTSR, the stop bit is sent, and then serial transmission of the next frame is started. If there is no transmit data, the TEND flag in SCFSR is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output continuously.

18.1 Register Descriptions

The PFC has the following registers. For details on the addresses of these registers and the states of these registers in each processing state, see section 24, List of Registers.

- Port A IO register H (PAIORH)
- Port A control register H1 (PACRH1)
- Port A control register H2 (PACRH2)
- Port B IO register L (PBIORL)
- Port B control register L1 (PBCRL1)
- Port B control register L2 (PBCRL2)
- Port C IO register H (PCIORH)
- Port C IO register L (PCIORL)
- Port C control register H2 (PCCR2H2)
- Port C control register L1 (PCCRL1)
- Port C control register L2 (PCCRL2)
- Port D IO register L (PDIORL)
- Port D control register L2 (PDCRL2)
- Port E IO register H (PEIORH)
- Port E IO register L (PEIORL)
- Port E control register H1 (PECRH1)
- Port E control register H2 (PECRH2)
- Port E control register L1 (PECRL1)
- Port E control register L2 (PECRL2)

20.2.7 Break Data Mask Register B (BDMRB)

BDMRB is a 32-bit readable/writable register. BDMRB specifies bits masked in the break data specified by BDRB.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BDMB31 to BDMB 0	All 0	R/W	<p>Break Data Mask B</p> <p>Specify bits masked in the break data of channel B specified by BDRB (BDB31 to BDB0).</p> <p>0: Break data BDBn of channel B is included in the break condition</p> <p>1: Break data BDBn of channel B is masked and is not included in the break condition</p> <p>Note: n = 31 to 0</p>

- Notes:
1. Specify an operated size when including the value of the data bus in the break condition.
 2. When the byte size is selected as a break condition, the same data must be set in bits 15 to 8 and 7 to 0 in BDRB as the break mask data.

20.2.8 Break Bus Cycle Register B (BBRB)

Break bus cycle register B (BBRB) is a 16-bit readable/writable register, which specifies (1) L bus cycle or I bus cycle, (2) instruction fetch or data access, (3) read or write, and (4) operand size in the break conditions of channel B.

Bit	Bit Name	Initial Value	R/W	Description
15 to 8	—	All 0	R	<p>Reserved</p> <p>These bits are always read as 0. The write value should always be 0.</p>

Bit	Bit Name	Initial Value	R/W	Description
7	CDB1	0	R/W	L Bus Cycle/I Bus Cycle Select B
6	CDB0	0	R/W	Select the L bus cycle or I bus cycle as the bus cycle of the channel B break condition. 00: Condition comparison is not performed 01: The break condition is the L bus cycle 10: The break condition is the I bus cycle 11: The break condition is the L bus cycle
5	IDB1	0	R/W	Instruction Fetch/Data Access Select B
4	IDB0	0	R/W	Select the instruction fetch cycle or data access cycle as the bus cycle of the channel B break condition. 00: Condition comparison is not performed 01: The break condition is the instruction fetch cycle 10: The break condition is the data access cycle 11: The break condition is the instruction fetch cycle or data access cycle
3	RWB1	0	R/W	Read/Write Select B
2	RWB0	0	R/W	Select the read cycle or write cycle as the bus cycle of the channel B break condition. 00: Condition comparison is not performed 01: The break condition is the read cycle 10: The break condition is the write cycle 11: The break condition is the read cycle or write cycle
1	SZB1	0	R/W	Operand Size Select B
0	SZB0	0	R/W	Select the operand size of the bus cycle for the channel B break condition. 00: The break condition does not include operand size 01: The break condition is byte access 10: The break condition is word access 11: The break condition is longword access

21.4.2 Reset Configuration

Table 21.4 Reset Configuration

ASEMD*¹	RES	TRST	LSI State
High	Low	Low	Normal reset and H-UDI reset
		High	Normal reset
	High	Low	H-UDI reset only
		High	Normal operation
Low	Low	Low	Reset hold* ²
		High	Normal reset
	High	Low	H-UDI reset only
		High	Normal operation

Notes: 1. Selects to normal mode or ASE mode.

$\overline{\text{ASEMD0}}$ = high: normal mode

$\overline{\text{ASEMD0}}$ = low: ASE mode

2. In ASE mode, the reset hold state is entered by driving the $\overline{\text{RES}}$ and $\overline{\text{TRST}}$ pins low for the given time. In this state, the CPU does not start up, even if the $\overline{\text{RES}}$ pin is driven high. After that, when the $\overline{\text{TRST}}$ pin is driven high, H-UDI operation is enabled, but the CPU does not start up. The reset hold state is canceled by the following: another $\overline{\text{RES}}$ assert (power-on reset) or $\overline{\text{TRST}}$ reassert.

21.4.3 TDO Output Timing

The timing of data output from the TDO differs according to the command type set in 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, the TDO signal is output at the TCK rising edge earlier than the JTAG standard by a half cycle.

- Management signals

Signal Name	Type	Description
CO_MDI	I	Management Data Input: Serial management data input.
CO_MDO	O	Management Data Output: Serial management data output.
CO_MDC	I	Management Clock: Serial management clock.
CO_MDIO_DIR	O	Management Data Direction: May be used to control output enabled buffer for MDIO.

- General signals

Signal Name	Type	Description
CO_CLKIN	I	Clock Input - PHY clock. Can be 25MHz either from mck of CPG module or from CK_PHY pin.

22.12 Signals Relevant to PHY-IF

This PHY core has a part set up by the PHY-IF module.

(1) PHY address

The PHY address initialized by PHYIFADDR of PHY-IF, is same as the one that the ordinary external PHY LSI has. It gives each PHY a unique address. This address is latched into an internal register during Module reset and PHY power on reset. Originally, it enables a function to manage each PHY via the unique address in a multi-PHY application.

About this PHY module, you can not connect multiple PHYs to the MII interface within the LSI. But PHY address is also used to seed the scrambler, so that please accord the configuration of PHYIFADDR and the PHY address on the management interface.

(2) Operation mode

The co_st_mode of the PHYIFCR of PHY-IF controls the configuration of 10/100 digital block.

(3) Software Reset of the PHY

The software reset of the on-chip PHY of this LSI has a defect in characteristics, which can prevent correct resetting of the PHY in some cases. Because of this, the PHY should be reset by a module reset, which is generated by setting the PHYIFCR register (in the PHY-IF module).

- Note:
1. Software reset refers to the reset which is executed by bit 15 of register 0 (basic control) described in section 22.4.2, SMI Register Mapping.
 2. Module reset refers to the reset which is executed by bit 14 of PHYIFCR (PHY-IF control register) described in section 23.2, Register Descriptions, in section 23, PHY Interface (PHY-IF).

(4) Waveform Adjustment

The Ethernet PHY module of this LSI has test registers for adjustment of differential output waveforms. Using these test registers in their initial values produces no problem, but their specifications are shown below to facilitate printed circuit board design by the customer.

(a) Adjustment of Tx100 Waveform Output

The on-chip PHY module of this LSI has the following adjustment registers as SIM registers, which allow waveform adjustment in the Tx100 operation. These registers have been designed so that they are not accidentally written. To change their values, follow the example procedure shown in "How to Use" that is described later.

- Register 20: Register for changing modes
- Register 23: Register for waveform adjustment
(The register numbers are decimal)
- Meanings of the value written to register 23

Bit	Bit Name	Initial Value	R/W	Description
15	—	1	RO	Reserved The write value should always be 1.
14 to 9	—	0	RO	Reserved The write value should always be 0.

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
DAR_3									DMAC
DMATCR_3									
CHCR_3	—	—	—	—	—	—	—	—	
	DO	TL	—	—	—	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	DL	DS	TB	TS1	TS0	IE	TE	DE	
DMAOR	—	—	CMS1	CMS0	—	—	PR1	PR0	
	—	—	—	—	—	AE	NMIF	DME	
PADRH	—	—	—	—	—	—	PA25DR	PA24DR	I/O
	PA23DR	PA22DR	PA21DR	PA20DR	PA19DR	PA18DR	PA17DR	PA16DR	
PAIORH	—	—	—	—	—	—	PA25IOR	PA24IOR	
	PA23IOR	PA22IOR	PA21IOR	PA20IOR	PA19IOR	PA18IOR	PA17IOR	PA16IOR	
PACRH1	—	—	—	—	—	—	—	—	
	—	—	—	—	PA25MD1	PA25MD0	PA24MD1	PA24MD0	
PACRH2	PA23MD1	PA23MD0	PA22MD1	PA22MD0	PA21MD1	PA21MD0	—	PA20MD0	
	—	PA19MD0	—	PA18MD0	—	PA17MD0	—	PA16MD0	
PBDRL	—	—	PB13DR	PB12DR	PB11DR	PB10DR	PB9DR	PB8DR	
	PB7DR	PB6DR	PB5DR	PB4DR	PB3DR	PB2DR	PB1DR	PB0DR	
PBIORL	—	—	PB13IOR	PB12IOR	PB11IOR	PB10IOR	PB9IOR	PB8IOR	
	PB7IOR	PB6IOR	PB5IOR	PB4IOR	PB3IOR	PB2IOR	PB1IOR	PB0IOR	
PBCRL1	—	—	—	—	—	PB13MD0	—	PB12MD0	
	—	PB11MD0	—	PB10MD0	—	PB9MD0	—	PB8MD0	
PBCRL2	—	PB7MD0	—	PB6MD0	—	PB5MD0	—	PB4MD0	
	—	PB3MD0	—	PB2MD0	—	PB1MD0	—	PB0MD0	

D

Data array	61
Data register.....	579
Data transfer instructions.....	43
Descrambling.....	652
Direct memory access controller (DMAC)	325
Divided areas and cache	55
Dual address mode.....	349

E

Endian/access size and data alignment ...	143
Equalizer, baseline wander correction and clock and data recovery	652
EtherC receiver	255
EtherC transmitter	253
Ethernet controller (EtherC)	233
Ethernet controller direct memory access controller (E-DMAC)	267
Exception handling	67
Exception handling operations	68
Exception handling vector table	69
Extension of chip select (\overline{CSn}) assertion period.....	156
External request mode	343

F

Features of instructions.....	29
Fixed mode	345
Flow control.....	262

G

General illegal instructions.....	77
General registers (Rn).....	25
General signals	667

H

Half-duplex and full-duplex.....	661
Host interface (HIF).....	511
H-UDI interrupt	628
H-UDI reset	628

I

I/O ports	579
Illegal slot instructions.....	77
Immediate data formats.....	29
Initial values of registers.....	27
Input clock to PHY module	668
Instruction formats.....	35
Instruction set.....	39
Intermittent mode.....	353
Interrupt controller (INTC)	83
Interrupt exception handling	75
Interrupt exception handling vector table.....	100
Interrupt priority	75
Interrupt response time	104
Interrupt sequence.....	102
Interrupt sources	74
IRQ interrupts	98
Isolate mode.....	662

J

Jabber detection	657
------------------------	-----

L

LED description.....	664
Link integrity test.....	662
Logic operation instructions	47
Loopback operation	664