SuperH

SuperH™ (SH) 64-Bit RISC Series

SH-5 System Architecture, Volume 2: Peripherals

Last updated 18 March 2002

Issued by the SuperH Documentation Group on behalf of SuperH, Inc.

SuperH, Inc.
San Jose, U.S.A. - Bristol, United Kingdom - Tokyo, Japan

www.superh.com

Contents

Preface

This document is part of the SuperH SH-5 CPU system documentation suite detailed below. Comments on this or other books in the documentation suite should be made by contacting your local sales office or distributor.

SuperH SH-5 document identification and control

Each book in the documentation suite carries a unique identifier in the form:

05-SA-nnnnn Vx.x

Where, *n* is the document number and *x.x* is the revision.

Whenever making comments on a SuperH SH-5 document the complete identification 05-SA-1000n Vx.x should be quoted.

SuperH SH-5 system architecture documentation suite

The SuperH SH-5 system architecture documentation suite comprises the following volumes:

- SH-5 System Architecture, Volume 1: System (05-SA-10001)
- SH-5 System Architecture, Volume 2: Peripherals (05-SA-10002)
- SH-5 System Architecture, Volume 3: Debug (05-SA-10003)

SuperH

1 External memory interface

1.1 Introduction

The external memory interface (EMI) provides the interface between the ST50 EVAL the SuperHyway and the external main memory subsystem. The EMI module comprises a SuperHyway port and an SDRAM controller. The following descriptions highlight the key features of the main memory array, SDRAM controller and SuperHyway port.

Main memory organization

- The array is organized as rows.
- Each row consists of 1 or more discrete devices or DIMM (single or double sided) modules arranged in sockets on a PCB.

SDRAM controller features

SDRAM controller includes:

- programmable external bus width: 16-, 32- and 64-bit,
- dual or quad bank SDRAM, specifically the PC-SDRAM standard, or double data rate (DDR) SDRAM (types cannot be mixed in the same system),
- main memory size: from 2 Mbytes to 2 Gbytes,
- memory modules supported: 2 rows of discrete SDRAM, single and double density DIMMs,
- SDRAM technology: 16-, 64-, 128- and 256-Mbit,
- SDRAM speed: 66, 100 and 133 MHz.

SuperHyway port

The SuperHyway port includes:

- two 4-deep-in-order queue for requests and responses, respectively: supports pipelining of up to 8 outstanding transactions on the SuperHyway,
- one control block (16-Mbyte space), containing the EMI module's VCR and SDRAM control registers
- 127 data blocks (16 Mbytes each) with access routed to the external memory.

1.2 SuperHyway interface

1.2.1 SuperHyway port

The SuperHyway port is divided into 1 control block and [n] number of data blocks. Each block is 16 Mbytes in size. The address range of the EMI is defined by VCR.BOT_MB and VCR.TOP_MB. Data blocks populate from VCR.BOT_MB up to VCR.TOP_MB (exclusive). The control block is assigned to 16 Mbytes, starting from VCR.TOP_MB (inclusive). The control block contains the EMI module's VCR and SDRAM control registers.

	EMI control block	EMI data blocks
Port name	EMI_cb	EMI_db
Lowest address	0xFF000000	0x80000000
Highest address	0xFFFFFFFF	0xFEFFFFFF

Table 1: EMI address space

Note: In the ST50 EVAL implementation, VCR.BOT_MB *= 0x08 and* VCR.TOP_MB *= 0x0F. Therefore data blocks populate the address range from 0x80000000 to 0xFEFFFFFF. This is summarized in Table 1*

When the EMI is active, memory accesses from the SuperHyway to the data blocks cause accesses to be made on the external memory bus. Each external memory access consists of a number of phases, each one representing a specific action performed on the external memory bus. The behavior of the external pins of the EMI can be programmed to allow the EMI to drive the external memory bus in an appropriate way for different DDR SDRAM and SDRAM parts.

1.2.2 Control block

12 registers (64 bits each) are populated in the control block. Except VCR, each register contains no more than 32 defined bits. The following table summarizes all registers implemented in ST50 EVAL's EMI module.

Note: VCR.TOP_MB *is 8-bit. BaseAddress = 0x(*VCR.TOP_MB*)000000.*

Control register name	Description	Offset[a]	Behavior
VCR	Version control register	0x00000000	See *Section 1.4.1*
MIM	Memory interface mode	0x00000008	See *Section 1.4.2*
SCR	SDRAM control	0x00000010	See *Section 1.4.3*
STR	SDRAM timing	0x00000018	See *Section 1.4.4*
PBS	Pin buffer strength	0x00000020	See *Section 1.4.5*
COC	Clock offset control	0x00000028	See *Section 1.4.6*
SDRA0~1	SDRAM row attribute	0x00000030, 0x00000038	See *Section 1.4.7*
	Reserved	0x00000040-0x00000080	
SDMR0~1	SDRAM mode register	0x008xxxxx 0x009xxxxx	See *Section 1.4.8*
-	Undefined	Remaining	

Table 2: Control block registers

a. All addresses are offset from the EMI base address

Note: If the EMI is active, the transactions to the control block are processed only when there are no outstanding data block transactions. While the EMI is processing control block transactions, the SDRAMs are in idle state. After processing control block transactions, the EMI's DRAM controller then continues with its normal behavior which reflects the state of the control registers. The EMI ensures that the change from the original behavior to the subsequent behavior is achieved instantaneously at a boundary between SDRAM commands during the processing of that transaction.

1.2.3 Reaction to packets

The EMI does not initiate request packets to the SuperHyway. The EMI processes the following packets received from SuperHyway.

For accessing the control block

- **load** 1/2/4/8-byte,[1]
- **store** 1/2/4/8-byte.[1]

For accessing the data block:

- **load** 1/2/4/8-byte,[1]
- **load** 16-byte (burst),[2]
- **load** 32-byte (burst),
- **store** 1/2/4/8-byte,[1]
- **store** 16-byte (burst),[2]

1. EMI treats **load** 1/2/4-byte and **store** 1/2/4-byte as **load** 8-byte and **store** 8-byte. EMI performs read/write bytes according to 8 byte-enable (or byte-mask) signals from the SuperHyway interface, regardless of **load** 1/2/4/8-byte and **store** 1/2/4/8-byte. It is the responsibility of the SuperHyway initiator to assert correct byte-enable bits to ensure consistency with the intention of **load** 1/2/4/8-byte and **store** 1/2/4/8-byte.
2. EMI expects and performs wrap-around within a 32-byte range for **store** 32-byte and **load** 32-byte, respectively. In case of load and store 16-byte, EMI expects they are all 16-byte aligned. EMI raises the error flag in VCR when it detects a non-aligned **load/store**16 SuperHyway packet

- **store** 32-byte (burst),
- **SWAP** 4/8-byte,[1]
- **read-modify-write** 4/8-byte.[2]

When accessing the control block, the EMI observes the following rules:

When accessing the control block, the EMI observes the rules below.

- Reads from the reserved control registers return 0. Writes to the reserved control registers are ignored.
- Reads from an undefined control register return an undefined value. Writes to undefined control registers are ignored.
- Accesses to the EMI's undefined control registers result in an error bit being set to indicate an access to a bad address. The full behavior of the transactions serviced by the EMI's control block is shown in *Table 3*.

1. EMI treats both **SWAP** 4-byte and **SWAP** 8-byte as **SWAP** 8-byte. It is the responsibility of the SuperHyway initiator to assert correct byte-enable bits to ensure the consistency with the intention of **SWAP** 4-byte and **SWAP** 8-byte.
2. For supporting **read-modify-write**, EMI receives a LOCK signal from the SuperHyway. EMI treats **read-modify-write** as **Load** 8-byte and **Store** 8-byte when LOCK = 1 and LOCK = 0, respectively. EMI performs read or write bytes according to 8 byte-enable (or byte-mask) signals from the SuperHyway interface. EMI treats both **read-modify-write** 4-byte and **read-modify-write** 8-byte as **read-modify-write** 8-byte. It is the responsibility of the SuperHyway initiator to assert correct byte-enable bits to ensure the consistency with the intention of **SWAP** 4-byte and **SWAP** 8-byte.

Packet received	Condition	Effect
Load 1/2/4/8-byte	Request is a sub-word (less than 8-byte) or whole word load from an undefined control register	VCR.PERR.BAD_ADDR set VCR.PERR.ERR_SNT set Access ignored, error response packet sent
	Request is a sub-word or whole word load from a reserved control register	An ordinary response sent Return data all zeros
	Request is a sub-word or whole-word load from an defined control register	An ordinary response sent Return data determined by control register accessed
Store 1/2/4/8-byte	Request is a sub-word or whole-word store to an undefined control register	VCR.PERR.BAD_ADDR set VCR.PERR.ERR_SNT set Access ignored, error response packet sent
	Request is a sub-word or whole-word store to a reserved control register	An ordinary response sent Access ignored
	Request is a sub-word or whole-word store to a defined control register	An ordinary response sent. Written data determined by control register accessed
All other packets	Request is to an undefined control register	VCR.PERR.BAD_OPC set VCR.PERR.BAD_ADDR set. VCR.PERR.ERR_SNT set Access ignored, error response packet sent
	Request is to a reserved or defined control register	VCR.PERR.BAD_OPC set VCR.PERR.ERR_SNT set. Access ignored, error response packet sent

Table 3: Packets directed to EMI's control block

When the data block is addressed, the EMI observes the rules below:

- The EMI does not service any packet directed to the data block when the SDRAM controller is disabled (MIM.DCE = 0).
- An out-of-range address is defined as a location beyond the address defined in the SDRAM row attribute register, see *Section 1.4.7: SDRAM row attribute registers (EMI.SDRA0, 1) on page 51* for details. Reads from an out-of-range address return an undefined value. Writes to an out-of-range address are ignored. In either cases, the EMI responds with an error packet and set error flags in the VCR.The full behavior of the transactions serviced by the EMI's data block is shown in the following table.

Packet received	Condition	Effect
Load 1-, 2-, 4- and 8-byte, **Load** 32-byte, **Store** 1-, 2-, 4-, and 8-byte, **Store** 32-byte, **SWAP**, r**Read-modify-write**	DRAM controller disabled	VCR.MERR.ERR_SNT set VCR.MERR.DRAM_INACTIVE set Access ignored, error response packet sent
	DRAM controller enabled Address within the range	An external memory access made An ordinary response sent Effect of external memory accesses and any returned data depends on the external implementation
	DRAM controller enabled Address out of the range	VCR.MERR.ERR_SNT set VCR.MERR.BAD_ADDR set Access ignored, error response packet sent

Table 4: Behavior of the transactions serviced by the EMI's data block

Packet received	Condition	Effect
Load 16-byte, **Store** 16-byte	DRAM controller disabled	VCR.MERR.ERR_SNT set VCR.MERR.DRAM_INACTIVE set Access ignored, error response packet sent
	DRAM controller enabled Address within the range	External memory access made Ordinary response sent Effect of external memory accesses and any returned data depends on the external implementation
	DRAM controller enabled Address out of the range or not aligned at 16-byte boundary	VCR.MERR.ERR_SNT set VCR.MERR.BAD_ADDR set Access ignored, error response packet sent
All other packets	DRAM controller disabled	VCR.MERR.ERR_SNT set VCR.MERR.DRAM_INACTIVE set Access ignored, error response packet sent
	DRAM controller enabled	VCR.MERR.BAD_OPC set VCR.MERR.ERR_SNT set Access ignored, error response packet sent

Table 4: Behavior of the transactions serviced by the EMI's data block

1.2.4 Pipelining request queue

The SuperHyway port maintains two 4-deep-in-order queues, 1 for request and the other 1 for response. Each entry in these queues has a 32-byte data buffer and 32 byte-enable bits to accommodate the data to and from main memory. The request queue's entry is transferred to the response queue once it is serviced by the SDRAM controller. The response queue's entry is retired once it is output to the SuperHyway. Accesses to any given address are observed to occur in the order which they are received by the EMI. Any EMI requester will see responses in the same order as the requests. Byte-gathering for the subsequent write requests is not supported in the EMI. It is assumed that each bus initiator module on the SuperHyway performs byte-gathering itself.

1.2.5 Coherency

The memory of the EMI is coherent as viewed from the SuperHyway. All requests are processed sequentially on the External memory interface in the order of the receipt of those requests by EMI. However, the system can decouple the generation of response packets from the actual external memory bus accesses. For store transactions, the corresponding store response packets may not be returned to the initiator on the SuperHyway until the write operations are actually completed on the DRAM interface. Since the local system is the sole owner of the external main memory and all the requests to the same address are processed in order (as they are received from the SuperHyway interface) on the SDRAM interface, memory coherency is achieved.

A swap packet comes with store data. When processing a swap request, the EMI initiates a read transaction on the DRAM interface. Once the data is received by the SDRAM controller, a write command is issued.

1.2.6 Standby mode

Entering standby mode

The power management module asserts a STBY_REQ signal to the EMI module. From this point on, the EMI can not take new requests from the SuperHyway. The EMI continues to service all the transaction requests in the queue. Upon concluding the last request, the EMI issues a power-down command to the SDRAM, and asserts STBY_ACK to the power management module. In return, the power management module stops providing clock to the EMI. Consequently, the EMI's MCLKO pin is stopped at high.

Leaving standby mode

The power management module de-asserts STBY_REQ and the EMI responds with de-assertion of STBY_ACK. The power management module then restores the clock supply before it de-asserts STBY_REQ and MCLKOstarts toggling accordingly.

1.3 SDRAM interface

The EMI's SDRAM controller can be configured to support PC-SDRAM and DDR SDRAM's. The functionality of the SDRAM interface is described in the following subsections.

1.3.1 Main memory configuration

The main memory is organized in rows. The data bus width can be programmed to 16-bit, 32-bit or 64-bit by writing MIM's BW field, see *Section 1.4.2: Memory interface mode register (EMI.MIM) on page 41*. The population on each row ranges from 2 Mbytes to 2 Gbytes. The different rows may have different size or technology of SDRAM population, but must have the same data bus width, burst length and share the same timing parameters defined in the STR register. SDRAM devices on the same row must be the same kind (for example, 4Mx16, 2-bank). Either PC SDRAM or DDR SDRAM can be supported but they cannot be mixed in main memory.

Note: The term "row" is used in 2 places, that is, SDRAM device's internal "row" address and main memory subsystem's "row" array. In this chapter, "row" indicates subsystem's row, while "(internal) row" means SDRAM's row address.

The upper boundary address of each row is defined in the SDRA.UBA field (SDRAM row attribute register's upper bound address). The request address [31, 21] is compared to SDRA.UBA [31, 21] to determine which NOTCSA (chip selection) signal is to be asserted. A NOTCSA signal is applied to all SDRAM devices on the same row.

Memory locations in these 2 rows must be contiguous in physical address space. SDRA1.UBA must be larger or equal to SDRA0.UBA. If the system consists of only 1 row (or DIMM), then it needs to be placed in the area corresponding to CS0 and SDRA0.UBA = SDRA1.UBA must be programmed. CS0 is asserted if the SuperHyway request access to EMI's data block and the request address [31, 21] is less than SDRA0.UBA (exclusive).

SDRA0.UBA has the priority over SDRA1.UBA if they are equal. If the physical address is less than SDRA0.UBA, CS0is asserted. If it is not less than SDRA0.UBA but is less than SDRA1.UBA, CS1 is asserted. Other cases are errors and are recorded in the VCR's error flag.

The following figure depicts a 64-bit wide, 96 Mbyte main memory subsystem. It is assumed VCR.BOT_MB = 0x08.

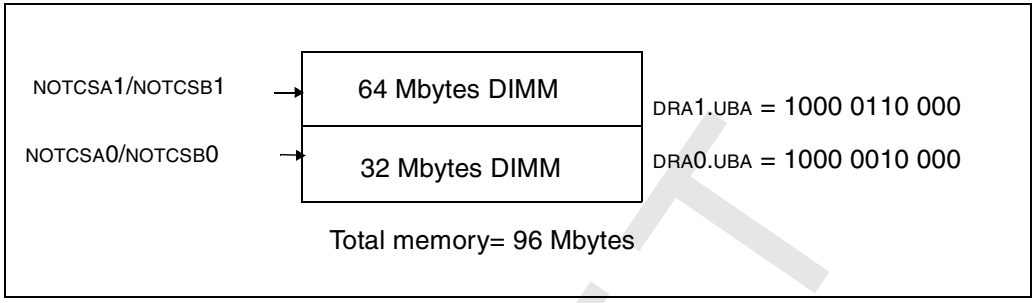

Figure 1: Main memory configuration example

Memory configuration can be little endian or big endian. The EMI is independent of endianess when the external bus width is 64 bit. When the external bus width is 32-bit, memory interface register MIM.ENDIAN bit (read only) indicates the endianess of the system.

1.3.2 SDRAM interface pins

The external pins are described in *Table 5*.

Name	In/out	Type*1	Size	Description
MCLKO	Out	LVTTL/ SSTL_2	± 3	SDRAM clock out: 66, 100 or 133 MHz clock output.
NOTMCLKO	Out	SSTL_2	± 3	MCLKO and NOTMCLKO are differential clock outputs to DDR SDRAM.
MCLKIN	In	LVTTL/ SSTL_2	1	Clock input for synchronization between internal clock and external clock
MCLKOUT	Out	LVTTL/ SSTL_2	1	Clock output for synchronization between internal clock and external clock

Table 5: SDRAM interface pins

Name	In/out	Type*1	Size	Description
CKE[1:0]	Out	LVTTL/ SSTL_2	2	Clock enable: Activates the clock signal when high and deactivates when low. By deactivating the clock, CKE low initiates the power down mode, self refresh mode or suspend mode.
NOTCS[1:0]	Out	LVTTL/ SSTL_2	2 2	Chip select: These pins perform the function of selecting the particular SDRAM components during the active state.
NOTWE	Out	LVTTL/ SSTL_2	1 1	Write enable signal: NOTWE is asserted during writes to SDRAM.
MA[14:0]	Out	LVTTL/ SSTL_2	15	Row and column address.
BA[1:0]	Out	LVTTL/ SSTL_2	2	Bank address.
MD[63:0]	In/out	LVTTL/ SSTL_2	64	Memory data.
DQS[7:0]	In/out	SSTL_2	8	Input/output data strobe: Used in DDR SDRAM only. These pins provide the read and write data strobe signal to/from the receiver circuit of DRAM controller. EMI drives DQS pins in write (store) cycles, while DDR SDRAM drives it in read (load) cycles.
DQM[7:0]	Out	LVTTL/ SSTL_2	8	For regular SDRAM: Input/output data mask: These pins act as synchronized output enables during read cycles and as byte enables during write cycles. For DDR SDRAM: These pins act as byte enables during write cycles
NOTRAS	Out	LVTTL/ SSTL_2	1 1	Row address strobe: NOTRAS signal used to generate encoded SDRAM command.
NOTCAS	Out	LVTTL/ SSTL_2	1 1	Column address strobe: NOTCAS signal used to generate encoded SDRAM command.

Table 5: SDRAM interface pins

Name	In/out	Type*1	Size	Description
VREF	In	SSTL_2	1	Input reference voltage

Table 5: SDRAM interface pins

To accommodate various loading conditions, the buffer strength of the pins is programmable. This feature can minimize unnecessary power consumption while still meeting the SDRAM device's timing requirements. See *Section 1.4.5* for details.

Note: CSB*[1:0],* WEB#*,* RASB#*,* CASB# *removed.* MCLKIN*,* MCLKOUT*, 2 pairs of* MCLKO *added.*

1.3.3 SDRAM devices

The EMI splits the physical memory address into banks, (internal) row and column addresses. The EMI contains 17 external address pins. BA[1, 0] specifies which bank, while MA[14, 0] indicates row and column addresses in each bank. The (internal) row address selects a page in an SDRAM. The column address selects a datum in a row. The EMI supports memories where row addresses are up to 15 bits.

The following 3 tables summarize various SDRAM devices which are used to construct the memory subsystem in 3 different data bus widths. They also illustrate MApins muxing versus SDRAM address split. EMI's MA[14, 0] pins are directly connected to SDRAM's A[14, 0]. The address split column in the table specifies the row and column address split within a given bank.

Note: SH-5 EMI will support 512 Mb single data rate and double data rate SDRAM. This table will be updated later to follow JEDEC specification of 512Mb SDRAM.

SDRAM EMI's MA*[14:13] pins are only relevant to 1 Gb generation. These tables will be revised once the pinout of 1Gb SDRAM is known.*

Using the second entry as an example, 2 of 16 Mbytes (2 M x 8 type, 2 banks) SDRAMs are used to construct a row of main memory. The SDRAM's internal row and column address bits are 11 and 9, respectively. The page size is 1 Kbyte. Total memory on this row is 4 Mbytes. The CPU's physical address PA [11] is output to BA[0] pins in both RAS and CAS phases. MA[10] is driven with PA [12] in RAS phase. AP (auto precharge) option is output to MA[10] in CAS phase, although the ST50 EVAL EMI does not issue either read-with auto-precharge or write-with auto-precharge commands.

16-bit data bus interface

SDRAM type	Address split	Page size	Row size	RAS CAS	BA1	BA0	MA12	MA11	MA10 /AP	MA9	MA8	MA [7, 0]
16 Mbit 2 bank												
1 Mbit x 16	11 x 8	512 bytes	2 Mbytes	RAS CAS		11 11			12 AP	10	9	[20, 13] [8, 1]
2 Mbit x 8	11 x 9	1 Kbytes	4 Mbytes	RAS CAS		11 11			12 AP	10	21 9	[20, 13] [8, 1]
4 Mbit x 4	11 x 10	2 Kbytes	8 Mbytes	RAS CAS		11 11			12 AP	22 10	21 9	[20, 13] [8, 1]
64 Mbit 2 bank												
4 Mbit x 16	13 x 8	512 bytes	8 Mbytes	RAS CAS		11 11	12	10	9 AP	22	21	[20, 13] [8, 1]
8 Mbit x 8	13 x 9	1 Kbytes	16 Mbytes	RAS CAS		11 11	12	10	23 AP	22	21 9	[20, 13] [8, 1]
16 Mbit x 4	13 x 10	2 Kbytes	32 Mbytes	RAS CAS		11 11	12	24	23 AP	22 10	21 9	[20, 13] [8, 1]
64 Mbit 4 bank												
4 Mbit x 16	12 x 8	512 bytes	8 Mbytes	RAS CAS	12 12	11 11		10	9 AP	22	21	[20, 13] [8, 1]
8 Mbit x 8	12 x 9	1 Kbyte	16 Mbytes	RAS CAS	12 12	11 11		10	23 AP	22	21 9	[20, 13] [8, 1]
16 Mbit x 4	12 x 10	2 Kbytes	32 Mbytes	RAS CAS	12 12	11 11		24	23 AP	22 10	21 9	[20, 13] [8, 1]
128 Mbit 4 bank												
8 Mbit x 16	12 x 9	1 Kbyte	16 Mbytes	RAS CAS	12 12	11 11		10	23 AP	22	21 9	[20, 13] [8, 1]

Table 6: Row and column addressing for memory size and number of banks (32-bit interface)

SDRAM type	Address split	Page size	Row size	RAS CAS	BA1	BA0	MA12	MA11	MA10 /AP	MA9	MA8	MA [7, 0]
16 Mbit x 8	12 x 10	2 Kbytes	32 Mbytes	RAS CAS	12 12	11 11		24	23 AP	22 10	21 9	[20, 13] [8, 1]
32 Mbit x 4	12 x 11	4 Kbytes	64 Mbytes	RAS CAS	12 12	25 25		24 11	23 AP	22 10	21 9	[20, 13] [8, 1]
256 Mbit 4 bank												
16 Mbit x 16	13 x 9	1 Kbyte	32 Mbytes	RAS CAS	12 12	11 11	10	24	23 AP	22	21 9	[20, 13] [8, 1]
32 Mbit x 8	13 x 10	2 Kbytes	64 Mbytes	RAS CAS	12 12	11 11	25	24	23 AP	22 10	21 9	[20, 13] [8, 1]
64 Mbit x 4	13 x 11	4 Kbytes	128 Mbytes	RAS CAS	12 12	26 26	25	24 11	23 AP	22 10	21 9	[20, 13] [8, 1]

Table 6: Row and column addressing for memory size and number of banks (32-bit interface)

32-bit data bus interface

SDRAM type	Address split	Page size	Row size	RAS CAS	BA1	BA0	MA12	MA11	MA10 /AP	MA9	MA8	MA [7, 0]
16 Mbit 2 bank												
1 Mbit x 16	11 x 8	1 Kbytes	4 Mbytes	RAS CAS		13 13			12 AP	11	10	[21, 14] [9, 2]
2 Mbits x 8	11 x 9	2 Kbytes	8 Mbytes	RAS CAS		13 13			12 AP	11	22 10	[21, 14] [9, 2]
4 Mbits x 4	11 x 10	4 Kbytes	16 Mbytes	RAS CAS		13 13			12 AP	23 11	22 10	[21, 14] [9, 2]
64 Mbit 2 bank												
4 Mbits x 16	13 x 8	1 Kbytes	16 Mbytes	RAS CAS		13 13	12	11	10 AP	23	22	[21, 14] [9, 2]
8 Mbits x 8	13 x 9	2 Kbytes	32 Mbytes	RAS CAS		13 13	12	11	24 AP	23	22 10	[21, 14] [9, 2]
16 Mbits x 4	13 x 10	4 Kbytes	64 Mbytes	RAS CAS		13 13	12	25	24 AP	23 11	22 10	[21, 14] [9, 2]
64 Mbit 4 bank												
2 Mbits x 32	11 x 8	1 Kbytes	8 Mbytes	RAS CAS	12 12	13 13			10 AP	11	22 AP*	[21, 14] [9, 2]
4 Mbits x 16	12 x 8	1 Kbytes	16 Mbytes	RAS CAS	12 12	13 13		11	10 AP	23	22	[21, 14] [9, 2]
8 Mbits x 8	12 x 9	2 Kbytes	32 Mbytes	RAS CAS	12 12	13 13		11	24 AP	23	22 10	[21, 14] [9, 2]
16 Mbits x 4	12 x 10	4 Kbytes	64 Mbytes	RAS CAS	12 12	13 13		25	24 AP	23 11	22 10	[21, 14] [9, 2]

Table 7: Row and column addressing for memory size and number of banks (32-bit interface)

SDRAM type	Address split	Page size	Row size	RAS CAS	BA1	BA0	MA12	MA11	MA10 /AP	MA9	MA8	MA [7, 0]
128 Mbit 4 bank												
8 Mbit x 16	12 x 9	2 Kbytes	32 Mbytes	RAS CAS	12 12	13 13		11	24 AP	23	22 10	[21, 14] [9, 2]
16 Mbit x 8	12 x 10	4 Kbytes	64 Mbytes	RAS CAS	12 12	13 13		25	24 AP	23 11	22 10	[21, 14] [9, 2]
32 Mbit x 4	12 x 11	8 Kbytes	128 Mbytes	RAS CAS	26 26	13 13		25 12	24 AP	23 11	22 10	[21, 14] [9, 2]
256 Mbit 4 bank												
16 Mbit x 16	13 x 9	2 Kbytes	64 Mbytes	RAS CAS	12 12	13 13	11	25	24 AP	23	22 10	[21, 14] [9, 2]
32 Mbit x 8	13 x 10	4 Kbytes	128 Mbytes	RAS CAS	12 12	13 13	26	25	24 AP	23 11	22 10	[21, 14] [9, 2]
64 Mbit x 4	13 x 11	8 Kbytes	256 Mbytes	RAS CAS	27 27	13 13	26	25 12	24 AP	23 11	22 10	[21, 14] [9, 2]

Table 7: Row and column addressing for memory size and number of banks (32-bit interface)

64-bit data bus interface

SDRAM type	Address split	Page size	Row size	RAS CAS	BA 1	BA0	MA12	MA11	MA10 /AP	MA9	MA8	MA [7, 0]
16 Mbit 2 bank												
1 Mbit x 16	11 x 8	2 Kbytes	8 Mbytes	RAS CAS		13 13			14 AP	12	11	[22, 15] [10, 3]
2 Mbits x 8	11 x 9	4 Kbytes	16 Mbytes	RAS CAS		13 13			14 AP	12	23 11	[22, 15] [10, 3]
4 Mbits x 4	11 x 10	8 Kbytes	32 Mbytes	RAS CAS		13 13			14 AP	24 12	23 11	[22, 15] [10, 3]
64 Mbit 2 bank												
4 Mbits x 16	13 x 8	2 Kbytes	32 Mbytes	RAS CAS		13 13	14	12	11 AP	24	23	[22, 15] [10, 3]
8 Mbits x 8	13 x 9	4 Kbytes	64 Mbytes	RAS CAS		13 13	14	12	25 AP	24	23 11	[22, 15] [10, 3]
16 Mbits x 4	13 x 10	8 Kbytes	126 Mbytes	RAS CAS		13 13	14	26	25 AP	24 12	23 11	[22, 15] [10, 3]
64 Mbit 4 bank												
2 Mbits x 32	11 x 8	2 Kbytes	16 Mbytes	RAS CAS	14 14	13 13			11 AP	12	23 AP*	[22, 15] [10, 3]
4 Mbits x 16	12 x 8	2 Kbytes	32 Mbytes	RAS CAS	14 14	13 13		12	11 AP	24	23	[22, 15] [10, 3]
8 Mbits x 8	12 x 9	4 Kbytes	64 Mbytes	RAS CAS	14 14	13 13		12	25 AP	24	23 11	[22, 15] [10, 3]
16 Mbits x 4	12x10	8K	128M	RAS CAS	14 14	13 13		26	25 AP	24 12	23 11	[22, 15] [10, 3]

Table 8: Row and column addressing for memory size and number of banks (64-bit interface)

SDRAM type	Address split	Page size	Row size	RAS CAS	BA 1	BA0	MA12	MA11	MA10 /AP	MA9	MA8	MA [7, 0]
128 Mbit 4 bank												
8 Mbit x 16	12 x 9	4 Kbytes	64 Mbytes	RAS CAS	14 14	13 13		12	25 AP	24	23 11	[22, 15] [10, 3]
16 Mbit x 8	12 x 10	8 Kbytes	128 Mbytes	RAS CAS	14 14	13 13		26	25 AP	24 12	23 11	[22, 15] [10, 3]
32 Mbit x 4	12 x 11	16 Kbytes	256 Mbytes	RAS CAS	14 14	27 27		26 13	25 AP	24 12	23 11	[22, 15] [10, 3]
256 Mbit 4 bank												
16 Mbit x 16	13 x 9	4 Kbytes	128 Mbytes	RAS CAS	14 14	13 13	12	26	25 AP	24	23 11	[22, 15] [10, 3]
32 Mbit x 8	13 x 10	8 Kbytes	256 Mbytes	RAS CAS	14 14	13 13	27	26	25 AP	24 12	23 11	[22, 15] [10, 3]
64 Mbit x 4	13 x 11	16 Kbytes	512 Mbytes	RAS CAS	14 14	28 28	27	26 13	25 AP	24 12	23 11	[22, 15] [10, 3]

Table 8: Row and column addressing for memory size and number of banks (64-bit interface)

Note: AP PIN: *EMI uses the* MIM.BY32AP *bit to determine if the* MA8 *pin is used to indicate the* PRE *and* PALL *commands.*

1.3.4 Initializing SDRAM devices

An initialization sequence to an SDRAM device must be done after power-on reset by the driver software. The operating system bootup code or driver software to initialize SDRAM should not read or write SDRAM before completion of the initialization.

Single data rate SDRAM

1 VDD and VDDQ are applied simultaneously on power-up by the system to meet the SDRAM specification.

2 CKE is initially low after a power-on reset. With the CKE enable command, write to the EMI.SCR register and all CKE becomes high.

3 A minimum pause of 200 µsec (some of SDRAM only require 100 µsec) needs to be provided after powers are stable.

4 A precharge all (PALL) is issued to SDRAM.

5 Eight autorefresh commands, that is CAS-before-RAS (CBR) cycles send to SDRAM.

6 A mode register set (MRS) command is issued to program the SDRAM parameters, for example burst length, and CAS_ Latency.

7 A legal command for normal operation can be started after an SDRAM-specific AC timing.

8 From reset to the completion of mode register set, the EMI should not drive DQ to avoid contention of drivers. At the same time DQM should keep high.

Mode register set and 8 CBR can be transposed. The minimum pause time might vary from the generation and the vendor of SDRAM. This EMI should be able to support both 100 µs or 200 µs providing the pause period by software allows this.

Double data rate SDRAM

1 The system provides four power in certain sequence. VDD first, VDDQ next then VREF and VTT. VTT is not provided to the EMI, it is externally connected to DQ, DQS and other pins through a series of termination registers. This is required to prevent latch-up in SDRAM devices. EMI should be able to support this power-up sequence. During and after power-on reset, CKE must be kept low.

2 After all power supply and reference voltages are stable, and the clock is stable a 200 µsec pause is necessary.

3 CKE should be brought high with the DESELECT command. After this point, unless EMI sends some command, EMI has to send the DESELECT command.

4 A PALL is issued to SDRAM.

5 A mode register set (MRS) command is issued to program the extended mode register to enable the DLL.

6 The MRS command is issued to program the mode register, reset the DLL in SDRAM and program the operating parameters, for example burst length and CAS_ Latency.

7 Wait ten cycles after the DLL reset and send two CBR commands to SDRAM.

8 A MRS command is issued to de-assert DLL initialization bit in the mode register. Other programing parameters should be the same as in previous programing. For some memory vendors, this step can be skipped because they support auto cleaning of the DLL initialization bit.

9 After 200 cycles from DLL reset, external memory becomes accessible.

The EMI's SDRAM controller provides two mechanisms for accomplishing the initialization sequence.

1 NOP, PALL, CKEH and CBR

The 's (SDRAM field written with appropriate values to prompt the to start issuing one of these commands. For instance, when = 3'b100, it results in a single CBR cycle on the SDRAM interface. When SMS = 3'b011, it results in the CKE signals going high. See *Section 1.4.3: SDRAM control register (EMI.SCR) on page 45* for details.

2 Setting the SDRAM device's

The SDRAM's needs to be initialized before actual operation. The software (boot code) initiates a write cycle to the register, and then a write to the SDMR[n] register in the control block. The SDRAM controller then issues an MRScommand to all SDRAM devices on row [n.

Example: issuing MRS command to row 0

Software does a dummy write to SDMR0, the physical address must be arranged in the following way:

- A[31, 20] = 1111 1111 1000r,
- A[16, 3] contains the value to be written to the SDRAM's mode register,
- DATA [63, 0] is ignored since SDMR0 is a write-only virtual register,
- A[12, 3] is copied to MA[9, 0], A[18, 15] to MA[13, 10] and A[14, 13] to BA[1, 0] when an MRS command is issued to the SDRAM devices.

Software needs to ensure that the SDRAM timing specification (between the MRS command and the first operational command) is met. 1 way to ensure this is to perform several SDRAM control register reads.

Subsequently, SCR.SMS is written with 3'b000, the normal SDRAM operation can then be started.

Note: Software must program the EMI's MIM *register before writing to* SDMR[N].

1.3.5 Operations

The SDRAM controller supports most PC-SDRAM commands (with the exception of read/write with auto-precharge) and most DDR SDRAM commands. The following truth table lists up all commands supported.

Function	Symbol	CKE [n - 1]	CKE [n]	not CS	not RAS	not CAS	not WE	MA11	AP[a] (MA10 /MA8)	BA [1, 0]	MA [9, 0]
Device deselect	DSEL	H	X	H	X	X	X	X	X	X	X
No operation	NOP	H	X	L	H	H	H	X	X	X	X
Burst stop in read	BST	H	X	L	H	H	L	X	X	X	X
Read	READ	H	X	L	H	L	H	V	L	V	V
Write	WRITE	H	X	L	H	L	L	V	L	V	V
Bank activate	ACT	H	X	L	L	H	H	V	V	V	V
Precharge select bank	PRE	H	X	L	L	H	L	V	L	V	X
Precharge all banks	PALL	H	X	L	L	H	L	X	H	X	X
Auto refresh	CBR	H	H	L	L	L	H	X	X	X	X
Self refresh entry from idle	SLFRSH	H	L	L	L	L	H	X	X	X	X
Self refresh exit	SLFRSHX	L	H	H	X	X	X	X	X	X	X
Power-down entry from idle	PWRDN	H	L	X	X	X	X	X	X	X	X
Power-down exit	PWRDNX	L	H	H	X	X	X	X	X	X	X
Mode register set	MRS	H	X	L	L	L	L	V	V	V	V

Table 9: SDRAM command truth table

a. AP pin: EMI uses MIM.BY32AP bit to determine if MA8 pin is used to indicate PRE and PALL commands.

Note: The EMI does not support full-page burst operation. The EMI issues a BST *command to terminate the burst read-only in DDR SDRAM mode.*

The timing for issuing these commands is governed by the SDRAM timing register, see *Section 1.4.4: SDRAM timing register (EMI.STR) on page 47* for details. The EMI's SDRAM controller can open up to 4 pages for each SDRAM row and fully exploit the multi-bank architecture of modern SDRAM devices by tightly pipelining SDRAM commands. The EMI is capable of detecting multiple consecutive requests to the same SDRAM page. The SDRAM controller may combine same-page requests into a single same-page access, providing that the timing of the requests is suitable.

Multi-bank ping-pong transaction

2 bank ping-pong access is illustrated in the following diagram. The peak bandwidth is obtained in this scenario.

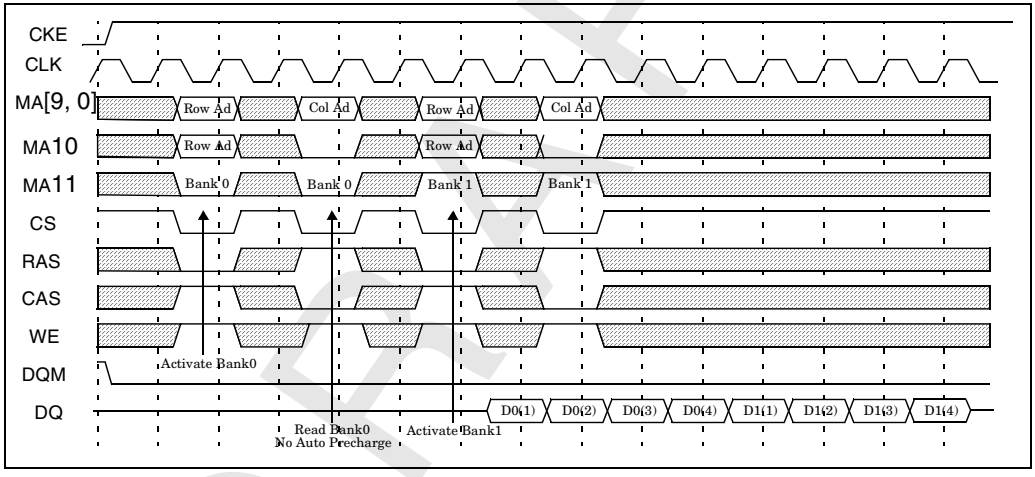

Figure 2: PC SDRAM 2 bank ping pong read

1.3.6 Refresh

When DRAM refresh enable is 1 (MIM.DRE = 1), The EMI can automatically generate refresh cycles. A 12-bit quantity (MIM.DRI, DRAM refresh interval) specifies the number of memory clock cycles between refreshes. Software should program MIM.DRI in the inclusive range [128, 4095]. The behavior of the DRAM controller is undefined if the EMI is enabled and if DRI is less than 128.

At the start of a refresh interval, the SDRAM controller loads DRI's 12-bit value into an internal counter. This counter is decremented by 1 in each memory clock cycle. When the counter reaches 0, DRI's value is reloaded into the counter and the next refresh interval is started.

All banks must be closed before refresh operation can be performed. The SDRAM controller issues a PRECHARGE ALL (PALL) command if there are any open pages. The SDRAM controller then issues an AUTO REFRESH command (CBR) after the TRP parameter is satisfied. The next row ACT command can be issued Trc clock (EMI.STR.SRC) later.

The SDRAM controller performs exactly 1 refresh operation for each refresh interval. It attempts to perform CBR as soon as possible within the refresh interval. When the counter ≤ 128 and CBR is not issued in the current refresh interval the SDRAM controller causes any current SDRAM access to complete in a timely manner by ensuring that the detection of same-page SDRAM access is prevented. Subsequently it performs PALL and CBR commands.

The maximum refresh rate that the EMI can support is 1 row every 128 clock cycles. At this rate, however, the detection of same-page SDRAM accesses will be permanently disabled.

As an example, the hard reset value of DRI is 1,562. For 100 MHz MCLKO, then this allows 1,024 refreshes in less than 16 ms.

Note: On average, the interval between 2 refreshes is determined by the DRI setting. However the interval between any 2 successive refreshes could be larger or smaller than DRI by (a page miss 32-byte transfer) clocks.

1.3.7 Power management

The EMI provides one power management mechanism.

When the EMI receives STBY_REQ from the power-down management unit (PMU), the EMI prepares the SDRAM rows to enter low power state. The sequence of events for both entering and leaving standby mode is described below. To make the correct sequence, cooperation with the software driver is important.

Entering standby

1 At first, no initiators should be issuing transaction requests to the EMI.

2 The standby management program should issue CBR command as the last command to EMI.

3 The standby management program asserts STBY_REQ to EMI.

4 All outstanding transaction requests are serviced.

5 The SDRAM controller issues a self refresh command and lowers CKE[1, 0] to both SDRAM rows. The SDRAM autonomously refreshes itself for the duration of the power-down mode.

6 EMI asserts STBY_ACK to PMU. The clock (MCLKO) can now be stopped.

Leaving standby by causes other than power-on reset

1 PMU resumes the EMI's SuperHyway clock and SDRAM clock and deasserts STBY_REQ.

2 The EMI de-asserts STBY_ACK, and starts to count down from (256 x SCR.CST) to zero every MClk cycle.

3 When count down reaches zero, the SDRAM controller asserts all CKE[1, 0] pins and sends DESELECT commands continuously. All SDRAM rows exit from self-refresh mode.

4 The first valid command can be issued ten cycles after CKE's rising edge.

5 In the case of DDR SDRAM, the EMI issues numbers of CBR commands defined by the CSR.BRFSH field.

1.3.8 Caution when programming SDRAM's mode register

To effectively support SuperHyway Load32 and Store32 packets, the EMI's SDRAM controller uses MIM.DT (SDRAM type) and MIM.BW (external data bus width) to determine the burst length.

MIM.DT device type	MIM.BW bus width	Burst length
0: PC SDRAM	00: 16-bit	8
	01: 32-bit	8
	10: 64-bit	4
1: DDR SDRAM	00: 16-bit	8
	01: 32-bit	8
	10: 64-bit	4

For a 16-bit external data bus width (for either PC-SDRAM or DDR SDRAM), the EMI splits a SuperHyway **Load32** or **Store32** packet into multiple SDRAM transactions, with a burst of 8 for each transaction. Therefore the BL field of the SDRAM device's mode register must be programmed to match the EMI's burst length behavior in the third column.

1.3.9 Using registered DIMM

When using registered DIMM, the MIM register's DIMM bit needs to be set to 1, so that EMI can:

- delay data output by 1 cycle to synchronize with the buffered (on DIMM card) command signals before they reach the SDRAM devices during a write operation,
- add 1 MCLK cycle to the setting of STR.SCL bit (CAS Latency). STR.SCL bits should be programmed with the same CL latency as the CL setting in the SDRAM device's mode register.

1.3.10 Others

Memory access to the address range of base address+(0x80000000 to 0xFEFFFFFF), is routed to the EMI. This address range may not be fully populated with the SDRAM's. Data access beyond the populated address, as defined in SDRA1's UBA, will not result in an external memory transaction. Software that dynamically sizes the amount of external memory must use an algorithm that is aware of this property. In the case of DIMMs, software can use I/O pins to implement a serial presence detect (SPD) mechanism for dynamic sizing of main memory.

1.4 Register description

1.4.1 Version control register (EMI.VCR)

The VCR is defined in the *SH-5 System Architecture, Volume 1: System, Section 2.7.3: Version control registers*. This register is universal to the control block of every SuperHyway module.

EMI.VCR				0xFF000000, i.e. 0x(VCR.TOP_MB)000000	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
PERR_FLAG	[7,0]	8	✓	Port error flags	Vary
	Operation		Indicates errors in the interface between EMI and the packet-router		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		
MERR_FLAG	[15,8]	8	✓	Memory error flags	Vary
	Operation		Indicates errors in the EMI		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		
MOD_VERS	[31,16]	16	-	Module version	RO
	Operation		Indicates module version number		
	Read		Returns 0x0000		
	Write		Ignored		
	Hard reset		0x0000		

Table 10: EMI.VCR

EMI.VCR				0xFF000000, i.e. 0x(VCR.TOP_MB)000000	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
MOD_ID	[47,32]	16	-	Module identity	RO
	Operation		Identifies module		
	Read		Returns 0x1000		
	Write		Ignored		
	Hard reset		0x1000		
BOT_MB	[55,48]	8	-	Bottom data memory block	RO
	Operation		Identifies top of data memory block		
	Read		0x80		
	Write		Ignored		
	Hard reset		0x80		
TOP_MB	[63, 56]	8	-	Top data memory block	RO
	Operation		Used to identify top of data memory block		
	Read				
	Write		Ignored		
	Hard reset		0xFF		

Table 10: EMI.VCR

The error status due to access to the EMI's control block is recorded in the PERR field. The set of supported PERR flags in EMI.VCR is given in the following table. The bit positions in this table are relative to the start of the EMI.VCR.PERR field; this field starts at bit 0 of EMI.VCR.

The error status due to access to the EMI's data block is recorded in MERR field. The set of supported MERR flags in EMI.VCR is given in the following table. The bit positions in this table are relative to the start of the EMI.VCR.MERR field. This field starts at bit 8 of EMI.VCR.

EMI.VCR.PERR			0xFF000000, i.e. 0x(VCR.TOP_MB)000000		
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
RESERVED	0	1	✓	Reserved	Res
	Operation		Reserved		
	Read		Returns 0		
	Write		Ignored		
	Hard reset		0		
ERR_SNT	1	1	✓	Error response sent	RW
	Operation		Indicates an error response has been sent This bit is set by the EMI hardware if an error response is sent by the EMI to SuperHyway. It indicates that an earlier request to the EMI was invalid		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		
BAD_ADDR	2	1	✓	Undefined control register	RW
	Operation		Indicates a request for an undefined control register has been received This bit is set by the EMI hardware if the EMI hardware receives a request for an undefined control register.		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		

Table 11: EMI.VCR.PERR

EMI.VCR.PERR			0xFF000000, i.e. 0x(VCR.TOP_MB)000000		
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
RESERVED	[4,3]	2	✓	Reserved	Res
	Operation				
	Read		Returns 0		
	Write		Ignored		
	Hard reset		0		
BAD_OPC	5	1	✓	Unsupported op code	RW
	Operation		Indicates a request with an unsupported opcode has been received This bit is set by the EMI hardware if a request with an unsupported opcode is received by EMI from SuperHyway.		
	Read		Returns current value		
	Write		Update current value		
	Hard reset		0		
RESERVED	[7,6]	2	✓	Reserved	Res
	Operation		Reserved		
	Read		Returns 0		
	Write		Ignored		
	Hard reset		0		

Table 11: EMI.VCR.PERR

EMI.VCR.MERR					
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
DRAM_INACTIVE	0	1	✓	Access to EMI data block (that is, external memory) when DRAM controller is disabled	RW
	Operation		This bit is set by the EMI hardware if a request is made to access external memory while DRAM controller is disabled.		
	Read		Returns current value		
	Write		Update current value		
	Hard reset		0		
ERR_SNT	1	1	✓	An error response has been sent	RW
	Operation		This bit is set by the EMI hardware if an error response is sent by the EMI to SuperHyway. It indicates that an earlier request to the EMI's data block was invalid		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		
BAD_ADDR	2	1	✓	A request to an out-of-range or unpopulated address has been received	RW
	Operation		This bit is set by the EMI hardware if the EMI hardware receives a request directed to an out-of-range address or an unpopulated address in data block		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		

Table 12: EMI.VCR.MERR

EMI.VCR.MERR					
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
RESERVED	[4,3]	2	✓	Reserved	Res
	Operation		Reserved		
	Read		Returns 0		
	Write		Ignored		
	Hard reset		0		
BAD_OPC	5	1	✓	A request with an unsupported opcode has been received	RW
	Operation		This bit is set by the EMI hardware if a request with an unsupported opcode is received by EMI from SuperHyway		
	Read		Returns current value		
	Write		Update current value		
	Hard reset		0		
RESERVED	[7,6]	2	✓	Reserved	Res
	Operation		Reserved		
	Read		Returns 0		
	Write		Ignored		
	Hard reset		0		

Table 12: EMI.VCR.MERR

1.4.2 Memory interface mode register (EMI.MIM)

EMI.MIM register specifies the configuration of the DRAM interface.

EMI.MIM				0x(VCR.TOP_MB)000000+0x00000008	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
DCE	0	1	-	DRAM controller enable	RW
	Operation		Indicates whether the SDRAM controller is enabled or disabled When the SDRAM controller is disabled, the EMI generates error responses to SuperHyway requests (for data block access) directed to the EMI.		
	Read		Returns current value		
	Write		0: SDRAM controller is disabled 1: SDRAM controller is enabled		
	Hard reset		0		
DT	1	1	-	DRAM type	RW
	Operation		Specifies the SDRAM type		
	Read		0: PC-SDRAM 1: DDR SDRAM		
	Write		Updates current value		
	Hard reset		Undefined		
RESERVED	[5,2]	4	-	Reserved	Res
	Operation		Reserved		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 13: EMI.MIM

EMI.MIM				0x(VCR.TOP_MB)000000+0x00000008	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
BW	[7,6]	2	-	Bus width	RW
	Operation		Indicates the data bus width of the EMI's SDRAM interface		
	Read		Returns current value		
	Write		00: 16 01: 32 10: 64 11: Reserved		
	Hard reset		10		
ENDIAN	8	1	-	Memory endianness	RO
	Operation		Indicates whether the memory configuration is little or big endian This bit only affects 16-bit and 32-bit bus width modes. For 64-bit external interface, the endianess is transparent to the EMI.		
	Read		0: Little endian memory configuration 1: Big endian memory configuration		
	Write		Ignored		
	Hard reset		0		
DRE	9	1	-	DRAM refresh enable	RW
	Operation		Enable refresh mechanism		
	Read		Returns current value		
	Write		0: Disables DRAM refresh 1: Enables DRAM refresh		
	Hard reset		0		

Table 13: EMI.MIM

EMI.MIM				0x(VCR.TOP_MB)000000+0x00000008	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
DIMM	10	1	-	Registered DIMM module	RW
	Operation		Constructs the external row		
	Read		Returns current value		
	Write		1: Data output delayed by 1 MCLK cycle and 1 MCLK cycle added to CAS latency		
	Hard reset		0		
BY32AP	11	1	-	Interfacing x32 SDRAM devices	RW
	Operation		Pre-charges all bank command's indicator for x32 SDRAM devices		
	Read		Returns current value		
	Write		0: BY32AP MA8 pin is not used when EMI issues PRE or PALL commands. 1: BY32AP MA8 pin is used when EMI issues PRE or PALL commands.		
	Hard reset		0		
RESERVED	[15,12]	4	-	Reserved	Res
	Operation		Reserved		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 13: EMI.MIM

EMI.MIM				0x(VCR.TOP_MB)000000+0x00000008	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
DRI	[27,16]	12	-	DRAM refresh interval	RW
	Operation		Determines the maximum number of memory clock cycles between row refreshes, when enabled		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0x61A		
RESERVED	[55,28]	27	-	Reserved	Res
	Operation		Software should always write 0 to these bits. Software should always ignore the value read from these bits.		
	Read				
	Write				
	Hard reset				
DLL1RST	See product datasheet for details				
DLL2RST	See product datasheet for details				
RESERVED	[63,58]	6	-	Reserved	Res
	Operation		Software should always write 0 to these bits. Software should always ignore the value read from these bits.		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 13: EMI.MIM

1.4.3 SDRAM control register (EMI.SCR)

EMI.SCR				0x(VCR.TOP_MB)000000 +0x00000010	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
SMS	[2,0]	3	-	SDRAM mode select	RW
	Operation		Enables the SDRAM controller to perform normal SDRAM operation and to issue NOP, PALL and CBR which are required in the SDRAM device initialization sequence for power up or reset		
	Read		Returns current value		
	Write		000: Normal SDRAM operation when MIM.DCE = 1 001: NOP command enable When SMS is written with this value and MIM.DCE = 1, the EMI issues 1 NOP command to the SDRAM interface. To have [n] number of NOP commands, SCR.SMS must be written with 001 [n] times. This command applies to all external SDRAM rows 010: Precharge all banks When SMS is written with this value and MIM.DCE = 1, the EMI issues 1 PRECHARGE ALL command to the SDRAM interface. To have [n] number of PALL commands, SCR.SMS must be written with 010 [n] times. This command applies to all external SDRAM rows. 011: Clock enable signals LCLKEN0 and LCLKEN1 active At reset the clocks are disabled. 100: CBR enable When SMS is written with this value and MIM.DCE = 1, the EMI issues 1 CBR command to the SDRAM interface. To have [n] number of CBR commands, SCR.SMS must be written with 011 [n] times. This command applies to all external SDRAM rows. 101, 110, 111: Reserved	
	Hard reset		3'b000		

Table 14: EMI.SCR

EMI.SCR				0x(VCR.TOP_MB)000000 +0x00000010	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
PDSE	3	1	-	Power-down SDRAM enable	RW
	Operation		Allows the SDRAM controller to issue a power-down command to an idle SDRAM row The SDRAM controller issues a power down exit command when there is a request to access this idle row.		
	Read		Returns current value		
	Write		0: Disable 1: Enable		
	Hard reset		0		
BRFSH	[6,4]	3	-	Burst refresh	RW
	Operation		This bit enables burst refresh after wake up from standby. 000: no. 001: 32 010: 512 011: 1024 100: 2048 101: 4096 110: reserved 111: reserved		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		0		
CST	See product datasheet for details				

Table 14: EMI.SCR

EMI.SCR				0x(VCR.TOP_MB)000000 +0x00000010	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
RESERVED	[15,7], [63,28]	45	-	Reserved	Res
	Operation		Software should always write 0 to these bits. Software should always ignore the value read from these bits.		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 14: EMI.SCR

1.4.4 SDRAM timing register (EMI.STR)

EMI.STR				0x(VCR.TOP_MB)000000 + 0x00000018	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
SRP	0	1	-	Trp, RAS precharge to ACT command	RW
	Operation		Controls the number of MCLKs for RAS precharge to ACT command (for the same bank)		
	Read		Returns current value		
	Write		0: 2 clocks of RAS precharge 1: 3 clocks of RAS precharge		
	Hard reset		Undefined		

Table 15: EMI.STR

EMI.STR				0x(VCR.TOP_MB)000000 + 0x00000018	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
SRCD	1	1	-	TRCD, RAS to CAS delay	RW
	Operation		Controls the number of MCLKs from a ROW ACTIVATE command to a column command (for the same bank)		
	Read		Returns current value		
	Write		0: 2 clocks of RAS to CAS delay. 1: 3 clocks of RAS to CAS delay		
	Hard reset		Undefined		
SCL	[4,2]	3	-	SDRAM CAS latency (CL)	RW
	Operation		Controls the number of MCLKs between when a read command is sampled by the SDRAMs and when the processor samples read data from SDRAMs		
	Read		Returns current value		
	Write		010: 2 clocks 011: 3 clocks 101: 1.5 clocks 110: 2.5 clocks All others: Reserved		
	Hard reset		Undefined		

Table 15: EMI.STR

EMI.STR				0x(VCR.TOP_MB)000000 + 0x00000018	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
SRC	[7,5]	3	-	Trc, RAS cycle time.	RW
	Operation		Minimum delay between ACT and Auto Refresh (to the same bank) ACT and ACT (to the same bank) Auto Refresh and ACT (to the same bank) Auto Refresh and Auto Refresh (to the same bank)		
	Read		Returns current value		
	Write		000: 6 clocks 001: 7 clocks 010: 8 clocks 011: 9 clocks All others: Reserved		
	Hard reset		Undefined		
SRAS	[10,8]	3	-	Tras, RAS active time	RW
	Operation		ACT to PRE command (for the same bank)		
	Read		Returns current value		
	Write		001: 5 clocks 010: 6 clocks 011: 7 clocks 100: 8 clocks 101: 9 clocks All others: Reserved		
	Hard reset		Undefined		

Table 15: EMI.STR

EMI.STR				0x(VCR.TOP_MB)000000 + 0x00000018	
Field	**Bits**	**Size**	**Volatile**	**Synopsis**	**Type**
SRRD	11	1	-	Trrd, RAS to RAS active delay	RW
	Operation		Specifies delay from ACT bank [n] to ACT bank [i] command (different bank)		
	Read		Returns current value		
	Write		0: 2 clocks 1: 3 clocks		
	Hard reset		Undefined		
SDPL	12	1	-	SDRAM Tdpl, as well as DDR SDRAM's Twr	RW
	Operation		SDRAM: last write-data to PRE or PALL command period DDR SDRAM: from the end of postamble to PRE or PALL command		
	Read		Returns current value		
	Write		0: 1 clock 1: 2 clocks		
	Hard reset		Undefined		
RESERVED	[63, 13]	51	-	Reserved	Res
	Operation		Reserved		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 15: EMI.STR

1.4.5 Pin buffer strength register (PBS)

Please see the product datasheet for the full register description.

1.4.6 Clock offset control (COC)

Please see the product datasheet for the full register description.

1.4.7 SDRAM row attribute registers (EMI.SDRA0, 1)

EMI.SDRA0-1				0x(VCR.TOP_MB)000000 + 0x00000030, 0x00000038	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[7,0]	8	-	Reserved	Res
	Operation		Reserved		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 16: EMI.DRA0, 1

EMI.SDRA0-1				0x(VCR.TOP_MB)000000 + 0x00000030, 0x00000038	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
SPLIT	[11,8]	4	-	SDRAM device address split for each bank	RW
	Operation		Defines the split of row and column address bits for a given bank within an SDRAM device		
	Read		Returns current value		
	Write		0000: 11x8 0001: 11x9 0010: 11x10 0011: Reserved 0100: 12x8 0101: 12x9 0110: 12x10 0111: Reserved 1000: 13x8 1001: 13x9 1010: 13x10 1011: 13x11 11nn: Reserved		
	Hard reset		Undefined		
BANK	12	1	-	SDRAM device bank number	RW
	Operation		Defines the SDRAM device bank number of the associated physical memory row		
	Read		Returns current value		
	Write		0: Dual-bank 1: Quad-bank		
	Hard reset		Undefined		

Table 16: EMI.DRA0, 1

EMI.SDRA0-1				0x(VCR.TOP_MB)000000 + 0x00000030, 0x00000038	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[20, 13]	8	-	Reserved	Res
	Operation		Reserved		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		
UBA	[31, 21]	11	-	Row upper boundary address	RW
	Operation		Defines the upper boundary address of the external SDRAM row in 2 Mbytes granularity UBA specifies the external row's upper boundary address [21, 31].		
	Read		Returns current value		
	Write		Updates current value		
	Hard reset		Undefined		
RESERVED	[63, 32]	32	-	Reserved	Res
	Operation		Software should always write 0 to these bits. Software should always ignore the value read from these bits.		
	Read		Ignored		
	Write		0		
	Hard reset		Undefined		

Table 16: EMI.DRA0, 1

1.4.8 SDRAM row mode registers (EMI.SDMR[0, 1])

These registers are write-only virtual registers, since physically they are not contained in the processor chip. A write to these virtual registers triggers an SDRAM mode register set command to be issued to a row of SDRAM devices.

The value on physical address A[12, 3] is copied to MA[9, 0] pins, A[14, 13] is output to BA[1, 0] and A[18, 15] is driven to MA[13, 10].

The values on data pins are undefined and are ignored by the SDRAM devices.

In response to the mode register set command, an SDRAM or DDR SDRAM device then latches MA[11, 0] and BA[1, 0] into its mode register. A read to these registers returns undefined value.

Please refer to the SDRAM and DDR SDRAM manufacture's data sheets for the definition of each bit in its mode register and extended mode register.

Note: The definition of a mode register's bit field varies with different SDRAM density.

1.5 References

JEDEC 64Meg double data rate (DDR) SDRAM

HM5251XXXB series 512M LVTTL SDRAM datasheet

HM5225XX5B series 256M LVTTL SDRAM datasheet

HM5212XX5F series 128M LVTTL SDRAM datasheet

HM54S64XXX series 64M LVTLL interface SDRAM datasheet

HM54S64XXD2 series double data rate SDRAM datasheet

NEC uPD4564xxx 64M synchronous DRAM datasheet

NEC uPD45D128xxx 128M Synchronous DRAM with double data rate datasheet

Micron MT48LCxxMxxA1/A2 64Meg SDRAM datasheet

Micron MT46V2M32 512k x 32 x 4 Banks DDR SDRAM datasheet

INTEL PC SDRAM specification version 1.7

2 Flash external memory interface (FEMI)

2.1 Overview

The flash external memory interface (FEMI) provides an interface between the SuperHyway and external memory subsystem. This interface may include non-DRAM/SDRAM components such as flash memory, SRAM, or other external devices using communications protocols such as MPX.

The FEMI module consists of a SuperHyway port and a memory controller which interfaces to external devices.

2.1.1 Main features

- The FEMI supports five data space areas (0 to 4). Area 0 is 64 Mbytes and areas 1 to 4 are 16 Mbytes each. Areas 1 to 4 can be merged to one 64 Mbyte area by a mode pin on a power-on reset.
- The bus width of each area can be specified by registers (8-, 16- or 32-bit). The bus width of the boot area (area 0) is specified by external pins.
- Wait insertion control by an external pin (RDY#).
- The number of wait states inserted is programmable.
- The memory type of each area is programmable.
- Programmable read data available time (from valid address to valid data out).
- Programmable data output hold cycle (one or two cycles).
- Conventional memory (SRAM, ROM) support.
- Byte control SRAM support.

- Flash memory interface (such as 28F800F3, 28F160F3 of Intel) with programmable features:
  - read mode (asynchronous page mode, synchronous burst mode),
  - 4, 8, 16, 32 burst length of synchronous burst mode (as SH-4),
  - write protection support for flash memory.
- Companion chip I/F support
  - MPX mode of the SH-4 with the additional feature that an external device can access SuperHyway space (for example, the EMI's local memory as a master).

The FEMI is assigned a 144 Mbyte region of the SuperHyway address space 0x00000000 to 0x08FFFFFF offset from the FEMI base address.

This is divided into a control space and a number of data spaces:

The region from 0x08000000 to 0x08FFFFFF (16 Mbyte) is allocated as a control block (CB)

There are five data spaces at the following addresses, each of which corresponds to an programmable external memory area:

Area 0	0x00000000 ~ 0x03FFFFFF	64 Mbyte
Area 1	0x04000000 ~ 0x04FFFFFF	16 Mbyte
Area 2	0x05000000 ~ 0x05FFFFFF	16 Mbyte
Area 3	0x06000000 ~ 0x06FFFFFF	16 Mbyte
Area 4	0x07000000 ~ 0x07FFFFFF	16 Mbyte

Memory types and protocols shown below can be specified for each area.

Memory type	Bus width (bit)	Access size (bit)
Normal memory	8, 16, 32	8, 16, 32, 64
Byte control SRAM	8, 16, 32	8, 16, 32, 64
Flash ROM	8, 16, 32	8, 16, 32, 64
MPX	32	8, 16, 32, 64

The bus width, type and protocol of each area can be specified.

- The bus width of area 0 is determined by external pins on reset.
- The bus width of other areas are specified by a control register located in the FEMI control block.

2.1.1 External pin

The FEMI has the following external pins:

Pin name[a]	I/O	Function
ADDR[25:0]	O	Address output
DATA[31:0]	I/O	Data input/output Multiplexed with address on MPX and MS mode
CKO	O	Clock out
CS[4:0]#	I/O	Chip select - when area 1~4 are merged into one 64 Mbyte area, CS1# is asserted for the merged area
BS#	I/O	Bus start notification
RD# /FRAME#	I/O	Read strobe/FRAME# on MPX mode
RD/WR#	I/O	0: Write 1: Read
RDY#	I/O	Wait state request on non-MS mode
WP#	O	Write protection for flash memory
WE[3:0]#/BE[3:0]#	I/O	Write strobe or byte enable for byte control SRAM WE3#: D31~D24 WE2#: D23~D16 WE1#: D15~D8 WE0#: D7~D0
CMD[5:3]	I	Bus command on MPX/MS mode
CMD[2:0]	I/O	Bus command on MPX/MS mode

Table 17: External pin

Pin name[a]	I/O	Function
BREQ#	I	External bus release request
BACK#	O	Bus release acknowledge
MODE8 MODE7	I	Specify bus width of area 0 00: 8-bit, 01: 16-bit, 10: 32-bit, 11: reserved
MODE6	I	Specify memory type of area 0 0: Normal memory 1: MPX
MODE5	I	Specify merge mode of area 1~4 into a signal area 0: Non merge 1: Merge
MODE4	I	Specify endianness 0: Big endian 1: Little endian

Table 17: External pin

a. The pin name may be prefixed with a character, generally 'F' to differentiate it from similarly named pins on other interfaces, refer to the product data sheet for details.

The FEMI samples a number of external pins during power-on reset. These pins control the following functionality:

- merge mode of area 1~4 into a signal area: MODE5,
- endianness of the FEMI: MODE4,
- area 0's bus type: MODE6,
- area 0's bus width: MODE8, MODE7.

Please refer to the product data sheet for more information.

2.2 Control registers

The FEMI functionality is controlled by a number of registers in the control block. Their functionality is summarized in *Table 18*.

Name	Address offset[a]	Functions specified/indicated
VCR	0x08000000	Version control register
MDCR	0x08000008	Mode control register (valid for all areas).
ANMCRN n=0~4	0x08000010 + 8 x n	Area [n] memory control register: memory type, bus width, operation mode of flash, burst parameters, number of idle and wait states, number of setup and data hold cycles
SNPCRN n=0,1	0x08000038 + 8 x n	MPX snoop control register
SNPAR[N] n=0, 1	0x08000048 + 8 x n	MPX snoop address register

Table 18: Control registers

a. This is offset from the FEMI base address

All control registers are defined to be 64 bits and may be accessed as a 64 bit quantity.

2.2.2 VCR (version control register)

The VCR register supplies general information about the module to the system. The following tables describe the register definition in the FEMI.

For more information, please refer to the *SH-5 System Architecture, Volume 1: System, Section 2.7.3: Version control registers.*

FEMI.VCR				0x08000000	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PERR_FLAGS	[7:0]	8	✓	P-port error flags	Varies
	Operation		Set “perr_flags” table below.		
	When read		Set “perr_flags” table below.		
	When written		Set “perr_flags” table below.		
	HARD reset		0x00		
MERR_FLAGS	[15:8]	8	✓	Module error flags (module specific)	RES
	Operation		RESERVED		
	When read		Returns 0x00		
	When written		Ignored		
	HARD reset		0x00		
MOD_VERS	[31:16]	16	-	Module version	RO
	Operation		Used to indicate module version number		
	When read		Returns 0x0000		
	When written		Ignored		
	HARD reset		0x0000		

Table 19: FEMI.VCR

FEMI.VCR				0x08000000	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
MOD_ID	[47:32]	16	-	Module identity	RO
	Operation		Used to identify module		
	When read		Returns 0x2185		
	When written		Ignored		
	HARD reset		0x2185		
BOT_MB	[55:48]	8	-	Bottom memory block	RO
	Operation		Used to identify bottom memory block		
	When read		0x00		
	When written		Ignored		
	HARD reset		0x00		
TOP_MB	[63:56]	8	-	Top memory block	RO
	Operation		Used to identify top memory block		
	When read		0x08		
	When written		Ignored		
	HARD reset		0x08		

Table 19: FEMI.VCR

VCR.perr_flags

Bit Name	Bit	Size	Volatile?	Synopsis	Type
ERR_RCV	[0]	1	✓	An error response has been received	RW
	Operation		This bit is set by the FEMI an error response is received by the FEMI from the packet-router. It indicates that an earlier request from the FEMI is invalid.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
ERR_SNT	[1]	1	✓	An error response has been sent	RW
	Operation		This bit is set by the FEMI if an error response is sent by the FEMI to the packet-router. It indicates that an earlier request to the FEMI was invalid.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
BAD_ADDR	[2]	1	✓	A request for an ‘undefined’ control register has been received	RW
	Operation		This bit is set by the FEMI hardware if the FEMI receives a LoadWord or StoreWord request for an ‘undefined’ control register in the FEMI memory map.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		

Table 20: VCR.perr.flags

Bit Name	Bit	Size	Volatile?	Synopsis	Type
UNSOL_RESP	[3]	1	✓	An unsolicited response has been received	RW
	Operation		This bit is set by the FEMI hardware if an unsolicited response is detected by the FEMI.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
BAD_DEST	[4]	1	✓	A request with illegal destination has been received	RW
	Operation		This bit is set by the FEMI hardware if a request with an illegal destination is received by the FEMI from the packet-router.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
BAD_OPC	[5]	1	✓	A request with an unsupported opcode has been received	RW
	Operation		This bit is set by the FEMI hardware if a request with an unsupported opcode is received by the FEMI from the packet-router.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
—	[7:6]	2	---	Reserved	RES
	Operation		Reserved		
	When read		Undefined		
	When written		Write 0		
	HARD reset		Undefined		

Table 20: VCR.perr.flags

2.2.3 MDCR (mode control register)

The mode control register specifies or indicates:

- control of High-Z functionality for ADDR[25:0], BS#, CS4~0#, RD/WR# pins during standby or when operating as a bus slave,
- enables support for external bus requests,
- selected endianess mode,
- status of areas 1 - 4, that is, if they are independently defined or merged into a single 64 Mbyte area with a single chip select signal CS1#.

FEMI.MDCR				0x08000008	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
HIZMEM	[0]	1	-	High-Z control	RW
	Operation		Specifies the state of signals (ADDR[25:0], BS#, CS3~0#, RD/WR#) in standby mode or when the bus is released.		
	When read		Returns the status of the signals 0: High-Z 1: The signals are driven		
	When written		Update value		
	HARD reset		0		
BREQEN	[1]	1	-	BREQ enable	RW
	Operation		Specifies if external requests can be accepted		
	When read		Returns if external requests can be accepted 0: External requests cannot be accepted 1: External requests can be accepted		
	When written		Update value		
	HARD reset		0		

Table 21: FEMI.MDCR

FEMI.MDCR				0x08000008	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
ENDIAN	[2]	1	-	FEMI endianess	RO
	Operation		Indicates endianess (big or little) of the FEMI The endianess is sampled during power-on reset and is static during normal operation		
	When read		Returns the endianess of all areas 0: Big endian 1: Little endian		
	When written		Ignored		
	HARD reset		Set as external pin specifies		
MERGE	[3]	1	-	Merge mode of area 1~area 4	RO
	Operation		Indicates if or not area 1 to area 4 are merged into one area. This is sampled during power-on reset and is static during normal operation		
	When read		Returns merge mode status 0: Area 1~4 are not merged 1: Area 1~4 are merged		
	When written		Ignored		
	HARD reset		Set as external pin specifies		
—	[63,4]	60	—	Reserved	RES
	Operation		Reserved		
	When read		Undefined		
	When written		Write 0		
	HARD reset		Undefined		

Table 21: FEMI.MDCR

2.2.4 AnMCR (area n memory control register, n=[4:0]

The AnMCR register defines or indicates the following information for each memory area:

- Memory type: Flash, SRAM or MPX,
  - area zero is defined by external pins during power-on reset,
  - areas 1 - 4 are defined by a write to this register.
- Memory bus width,
  - area zero is defined by external pins during power-on reset,
  - areas 1 - 4 are defined by a write to these register.
- Flash memory read operation mode,
  - asynchronous page mode,
  - synchronous burst mode.
- Flash memory WP# status,
- Burst parameters (number of accesses in a burst transfer),
- Number of idle cycles inserted between bus cycles when switching from one area to another, or switching from a read access to a write access in the same area,
- Number of wait state insertion cycles,
- Data access pitch when performing burst memory access,
- Cycles inserted in the setup time from the address until assertion of the read/ write strobe,
- The data hold time from negation of the strobe.

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
TYPE	[2:0]	3	-	Memory types of area [n]	RO/RW
	Operation		Specifies memory type of area [n]		
	When read		Returns memory type of area [n] 000: Normal memory (SRAM, ROM, EEPROM, Flash ROM) 001: MPX A0MCR is sampled during power-on reset and is static during normal operation Other values are reserved		
	When written		A0MCR: Ignored ANMCR: Update value (n=1~4)		
	HARD reset		A0MCR: Set as external pins specify ANMCR: 0 (n=1~4)		
SZ	[4:3]	2	—	Bus width of area [n]	RO/RW
	Operation		Specifies bus width of area [n]		
	When read		Returns bus width of area [n] 00: 8 bit 01: 16 bit 10: 32 bit 11: Reserved		
	When written		A0MCR: Ignored ANMCR: Update value (n=1~4)		
	HARD reset		A0MCR: Set as external pins specify ANMCR: 00 (n=1~4)		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
BST	[7:5]	3	-	Burst control of area [n]	RW
	Operation		Indicates number of accesses in a burst of area [n]. This field only significant when the area type is 0		
	When read		Returns if flush ROM is used. When flash ROM is used, number of accesses in a burst is returned. 000: Normal memory (non-flash ROM) 001: Flash ROM with 4 consecutive accesses 010: Flash ROM with 8 consecutive accesses 011: Flash ROM with 16 consecutive accesses 100: Flash ROM with 32 consecutive accesses 101~111: Reserved		
	When written		Updates value		
	HARD reset		000		
FLMD	[8]	1	-	Flash ROM read mode of area [n]	RW
	Operation		Specifies Flash ROM read mode of area [n] This field is only significant when the area corresponds to flash ROM (type=0 & bst =001, 010, 011, or 100) 0: Synchronous burst mode 1: Asynchronous page mode		
	When read		Returns flash ROM read mode		
	When written		Updates value		
	HARD reset		1		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
FLWP	[9]	1	-	WP# control of area [n]	RW
	Operation		Specifies WP# pin output state This field is defined only for areas defined as flash ROM area (type=0 & bst =001, 010, 011, or 100) 0: WP# is asserted high level 1: WP# is negated low level		
	When read		Returns value		
	When written		Updates value		
	HARD reset		0		
MBC	[10]	1	-	Byte control SRAM mode of area [n]	RW
	Operation		Specifies area [n] is normal memory or byte control SRAM. This field is defined only for areas with type=0 and bst=000. 0: Normal memory 1: Byte control SRAM		
	When read		Returns byte control SRAM mode		
	When written		Updates value		
	HARD reset		0		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
IW	[14:12]	3	-	Inter-cycle idle specification of area [n]	RW
	Operation		Specifies the number of idle cycles to be inserted when switching from area [n] to another, or from a read access to a write access in area [n]. The inter cycle idle time becomes valid following the end of any hold cycles.		
	When read		Returns number of idle cycles of area [n] 000: 0 001: 1 010: 2 011: 3 100: 6 101: 9 110: 12 111: 15		
	When written		Update value		
	HARD reset		111 (15 cycles are inserted)		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
BP	[18:16]	3	-	Burst pitch of area [n]	RW
	Operation		Specifies the data access pitch when performing burst memory access for area [n]. This indicates wait cycles inserted in each burst data transfer cycle. This field is valid areas with type=0 & bst= 001, 010, 011, or 100 & flmd=1 (asynchronous flash memory)		
	When read		Returns burst pitch per data transfer of area [n]. 000: 0 (external delay by RDY# is ignored) 001: 1 (external delay by RDY# is enabled) 010: 2 (external delay by RDY# is enabled) 011: 3 (external delay by RDY# is enabled) 100: 4 (external delay by RDY# is enabled) 101: 5 (external delay by RDY# is enabled) 110: 6 (external delay by RDY# is enabled) 111: 7 (external delay by RDY# is enabled)		
	When written		Update value		
	HARD reset		111		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
WS (NORMAL)	[22:20]	3	-	Wait states of area [n]	RW
	Operation		Specifies the number of wait states to be inserted for area [n] for type=0 & bst=000 (normal memory)		
	When read		Returns inserted wait states of area [n]. External delay by RDY# is disabled for ws=000, otherwise enabled. 000:0 001:1 010:2 011:3 100:6 101:9 110:12 111:15		
	When written		Update value		
	HARD reset		111		
WS (FLASH)	[22:20]	3	-	Wait states of area [n]	RW
	Operation		Specifies the number of wait states to be inserted for area [n] for type=0 & bst= 001, 010, 011, or 100 (flash memory)		
	When read		Returns inserted wait states from RD# assert to the 1st valid data output of area [n]. External delay by RDY# is disabled. 000:1 001:2 010:3 011:4 100:5 101:6 110:7 111:8		
	When written		Update value		
	HARD reset		111		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
WS (MPX)	[22:20]	3	-	Wait states of area [n]	RW
	Operation		Specifies the number of wait states to be inserted by a MPX master device (type=1) for area [n].		
	When read		Returns inserted wait states of area [n]. External delay by RDY# is enabled. 000: 1st read 1, 1st write 0, 2nd access or thereafter 0 001: 1st read 1, 1st write 1, 2nd access or thereafter 0 010: 1st read 2, 1st write 2, 2nd access or thereafter 0 011: 1st read 3, 1st write 3, 2nd access or thereafter 0 100: 1st read 1, 1st write 0, 2nd access or thereafter 1 101: 1st read 1, 1st write 1, 2nd access or thereafter 1 110: 1st read 2, 1st write 2, 2nd access or thereafter 1 111: 1st read 3, 1st write 3, 2nd access or thereafter 1		
	When written		Update value		
	HARD reset		111		
HLD	[25:24]	2	-	Data hold time of area [n]	RW
	Operation		Specifies the number of cycles inserted in the data hold time from negation of the read/write strobe of area [n]. AnMCR.hld is valid when AnMCR.type=0.		
	When read		Returns number of wait cycles inserted in hold time of area [n] 00: 0 01: 1 10: 2 11: 3		
	When written		Update value		
	HARD reset		11		

Table 22: FEMI.AnMCR

FEMI.AnMCR (n=0~4)				0x08000010 + 8 x n (n=0~4)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
STUP	[27]	1	-	Write strobe setup time of area [n]	RW
	Operation		Specifies the number of cycles inserted in the setup time from the address until assertion of the read/write strobe.		
	When read		Returns number of wait cycles inserted in setup time of area [n]		
	When written		Update value		
	HARD reset		1		
—	[2:1][10] [15][19] [23][26] [63:28]	--	—	Reserved	RES
	Operation		Reserved		
	When read		Undefined		
	When written		Write 0		
	HARD reset		0		

Table 22: FEMI.AnMCR

2.2.5 SNPCRn (snoop control register n, n=0,1)

External devices are able to access memory within the SH product via the FEMI which may have been cached by the CPU.

It is possible to define a window into memory for which the FEMI will ensure the access is coherent by issuing the appropriate commands via the SuperHyway.

Access from an external device to addresses mapped into the FEMI memory are forbidden and will lead to undefined behavior.

The SNPCR and SNPAR registers are used to define the areas of memory for which this functionality is required.

FEMI.SNPCRn, n=0,1				0x08000038 + 8 x n (n=0,1)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
SNPMD	[1:0]	2	-	Snoop mode for SNPAR[N]	RW
	Operation		Specific if and how SNPAR[N] is used when an external device accesses internal memory locations		
	When read		Returns the value contained within SNPAR[N]. 00: SNPAR[N] is not compared. 01: Reserved 10: SNPAR[n] is compared (inclusive) hit -> Make access within window coherent No hit -> Normal access 11: SNPAR[n] is compared (exclusive) Hit -> Normal access No hit -> Make access outside window coherent	
	When written		Update value		
	HARD reset		00		

Table 23: FEMI.SNPCRn

FEMI.SNPCRn, n=0,1				0x08000038 + 8 x n (n=0,1)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RANGE	[4:2]	3	-	Address range to be compared	RW
	Operation		Specifies address range of SNPAR[N] to be compared		
	When read		Returns which address bits are compared. 000: SNPAR[N].CADR[31:12] is compared (4 kbyte) 001: SNPAR[N].CADR[31:16] is compared (64 kbyte) 010: SNPAR[N].CADR[31:20] is compared (1 Mbyte) 011: SNPAR[N].CADR[31:24] is compared (16 Mbyte) 100~111: Reserved Valid only when SNPAR[N].SNPMD=10 or 11.	
	When written		Update value		
	HARD reset		000		
AREA	[7:4]	3	-	Area with which SNPAR[N] is compared	RW
	Operation		Specifies area whose connected external device's requested address is compared with SNPAR[N].		
	When read		Returns area whose connected external device's requested address is compared with SNPAR[N]. 000: area 0 001: area 1 010: area 2 011: area 3 100: area 4 101~111: reserved Valid only when SNPAR[N].SNPMD=10 or 11.		
	When written		Update value		
	HARD reset		000		

Table 23: FEMI.SNPCRn

FEMI.SNPCRn, n=0,1				0x08000038 + 8 x n (n=0,1)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
—	[63:8]	56	—	RESERVED	RES
	Operation		RESERVED		
	When read		Undefined		
	When written		write 0		
	HARD reset		Undefined		

Table 23: FEMI.SNPCRn

2.2.6 SNPAR[n] (snoop address register n, n=0,1)

The SNPAR0/1 specifies to the FEMI the address to be compared with an address requested by an external device.

FEMI.SNPAR[n], n=0,1				0x08000048 + 8 x n (n=0,1)	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
cadr	[31:0]	32	-	Address to be compared	RW
	Operation		Specifies to the FEMI the address to be compared with an address requested by an external device		
	When read		Returns value		
	When written		Update value		
	HARD reset		Undefined		
—	[63:32]	32	—	Reserved	RES
	Operation		Reserved		
	When read		Undefined		
	When written		Undefined		
	HARD reset		Undefined		

Table 24: FEMI.SNPARn

2.3 Operations

2.3.1 Endian/access size and data alignment

The FEMI supports both big-endian mode (in which the most significant byte (MSB) is at the 0 address end in a string of byte data) and little-endian mode (in which the least significant byte (LSB) is at the 0 address end). The mode is set during power-on reset.

A data bus width of 8, 16, or 32 bits can be selected for normal memory, 32 bits for MPX and MS interface. Data alignment is carried out according to the data bus width and endian mode of each device. For instance, four read operations are needed to read longword (32 bit) data from an 8-bit device. Data alignment and data length conversion between different interfaces is performed automatically.

The relationship between the endian mode, device data length, and access unit, is shown in following six tables.

32-bit external device, big-endian access and data alignment

In the table, “A” indicates “asserted”.

Operation	NO.	D31-D24	D23-D16	D15-D8	D7-D0	WE 3#	WE 2#	WE 1#	WE 0#
Byte addr=4n	1	Data 7- 0	---	---	---	A			
Byte addr=4n+1	1	---	Data 7- 0	---	---		A		
Byte addr=4n+2	1	---	---	Data 7 - 0	---			A	
Byte addr=4n+3	1	---	---	---	Data 7 - 0				A
Word addr=4n	1	Data 15 - 8	Data 7 - 0	---	---	A	A		
Word addr=4n+2	1	---	---	Data 15 - 8	Data 7 - 0			A	A
Longword addr=4n	1	Data 31 - 24	Data 23 - 16	Data 15 - 8	Data 7 - 0	A	A	A	A

Operation	NO.	D31-D24	D23-D16	D15-D8	D7-D0	WE 3#	WE 2#	WE 1#	WE 0#
Quadword	1	Data 63 - 56	Data 55 - 48	Data 47 - 40	Data 39 - 32	A	A	A	A
	2	Data 31 - 24	Data 23 - 16	Data 15 - 8	Data 7 - 0	A	A	A	A

(2)16-bit external device, big-endian access and data alignment

In the table, “A” indicates “asserted”.

Operation	NO.	D31-D24	D23-D16	D15-D8	D7-D0	WE 3#	WE 2#	WE 1#	WE 0#
Byte addr=2n	1	---	---	Data 7 - 0	---			A	
Byte addr=2n+1	1	---	---	---	Data 7 - 0				A
Word	1	---	---	Data 15 - 8	Data 7 - 0			A	A
Longword	1	---	---	Data 31 - 24	Data 23 - 16			A	A
	2			Data 15 - 8	Data 15 - 8			A	A
Quadword	1	---	---	Data 63 - 56	Data 55 - 48			A	A
	2	---	---	Data 47 - 40	Data 39 - 32			A	A
	3	---	---	Data 31 - 24	Data 23 - 16			A	A
	4	---	---	Data 15 - 8	Data 7 - 0			A	A

8-bit external device, big-endian access and data alignment

In the table, “A” indicates “asserted”.

Operation	NO.	D31-D24	D23-D16	D15-D8	D7-D0	WE 3#	WE 2#	WE 1#	WE 0#
Byte	1	---	---	---	Data 7 - 0				A
Word	1	---	---	---	Data 15 - 8				A
	2	---	---	---	Data 7 - 0				A
Longword	1	---	---	---	Data 31 - 24				A
	2	---	---	---	Data 23 - 16				A
	3	---	---	---	Data 15 - 8				A
	4	---	---	---	Data 7 - 0				A
Quadword	1	---	---	---	Data 63 - 56				A
	2	---	---	---	Data 55 - 48				A
	3	---	---	---	Data 47 - 40				A
	4	---	---	---	Data 39 - 32				A
	5	---	---	---	Data 31 - 24				A
	6	---	---	---	Data 23 - 16				A
	7	---	---	---	Data 15 - 8				A
	8	---	---	---	Data 7 - 0				A

32-bit external device, little-endian access and data alignment

In the table, “A” indicates “asserted”.

Operation	NO.	D31-D24	D23-D16	D15-D 8	D7-D0	WE 3#	WE 2#	WE 1#	WE 0#
Byte addr=4n	1	---	---	---	Data 7 - 0				A
Byte addr=4n+1	1	---	---	Data 7 - 0	---			A	
Byte addr=4n+2	1	---	Data 7 - 0	---	---		A		
Byte addr=4n+3	1	Data 7 - 0	---	---	---	A			
Word addr=4n	1	---	---	Data 15 - 8	Data 7 - 0			A	A
Word addr=4n+2	1	Data 15 - 8	Data 7 - 0	---	---	A	A		
Longword addr=4n	1	Data 31 - 24	Data 23 - 16	Data 15 - 8	Data 7 - 0	A	A	A	A
Quadword	1	Data 31 - 24	Data 23 - 16	Data 15 - 8	Data 7 - 0	A	A	A	A
	2	Data 63 - 56	Data 55 - 48	Data 47 - 40	Data 39 - 32	A	A	A	A

16-bit external device, little-endian access and data alignment

In the table, “A” indicates “asserted”.

Operation	NO.	D31-D24	D23-D16	D15-D 8	D7-D0	WE 3#	WE 2#	WE 1#	WE 0#
Byte addr=2n	1	---	---	---	Data 7 - 0				A
Byte addr=2n+1	1	---	---	Data 7 - 0	---			A	
Word	1	---	---	Data 15 - 8	Data 7 - 0			A	A
Longword	1	---	---	Data 15 - 8	Data 7 - 0			A	A
	2			Data 31 - 24	Data 23 - 16			A	A
Quadword	1	---	---	Data 15 - 8	Data 7 - 0			A	A
	2	---	---	Data 31 - 24	Data 23 - 16			A	
	3	---	---	Data 47 - 40	Data 39 - 32			A	A
	4	---	---	Data 63- 56	Data 55 - 48			A	A

8-bit external device, little-endian access and data alignment

In the table, “A” indicates “asserted”.

Operation	NO.	D31-D24	D23-D16	D15-D8	D7-D0	WE3#	WE2#	WE1#	WE0#
Byte	1	---	---	---	Data 7 - 0				A
Word	1	---	---	---	Data 7 - 0				A
	2	---	---	---	Data 15 - 8				A
Longword	1	---	---	---	Data 7 - 0				A
	2	---	---	---	Data 15 - 8				A
	3	---	---	---	Data 23 - 16				A
	4	---	---	---	Data 31 - 15				A
Quadword	1	---	---	---	Data 7 - 0				A
	2	---	---	---	Data 15 - 8				A
	3	---	---	---	Data 23 - 16				A
	4	---	---	---	Data 31 - 24				A
	5	---	---	---	Data 39 - 32				A
	6	---	---	---	Data 47 - 40				A
	7	---	---	---	Data 55 - 48				A
	8	---	---	---	Data 63 - 56				A

2.3.2 Interface

Normal memory (SRAM, ROM) - basic timing

The normal memory interface of the FEMI uses strobe signal output so that SRAM can be connected directly. A no-wait normal access is completed in two cycles. The BS# signal is asserted for one cycle to indicate the start of a bus cycle. The CSN# signal is asserted on the T1 rising edge, and negated on the next T2 clock rising edge. Therefore, there is no negation period in case of access at the minimum pitch.

There is no access size specification when reading. The correct access start address is output in the least significant bit of the address, but since there is no access size information, 32 bits are always read in the case of a 32-bit device, and 16 bits in the case of a 16-bit device. When writing, only the WE# signal for the write to be written is asserted.

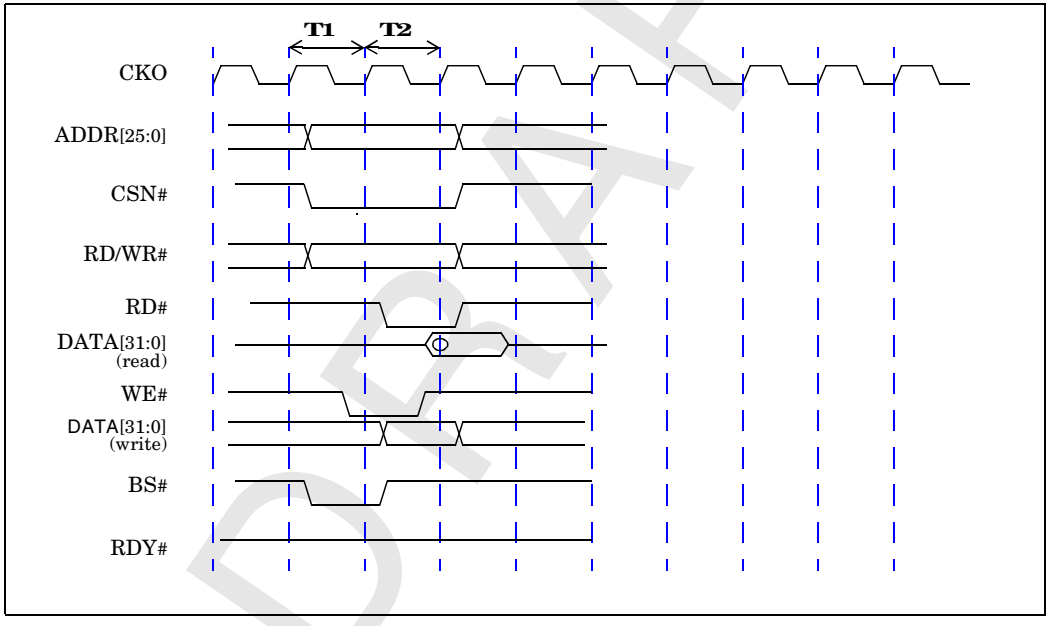

Figure 3: Normal memory interface (no wait)

Wait state control

The ANMCR.WS controls wait state insertion on the normal interface. If the ANMCR.WS is not 000, software wait state(s) are inserted.

When the software wait insertion is specified by the ANMCR, the external wait input RDY# signal is also sampled at the transition from the TW state to the T2 state on the rising edge of the clock.

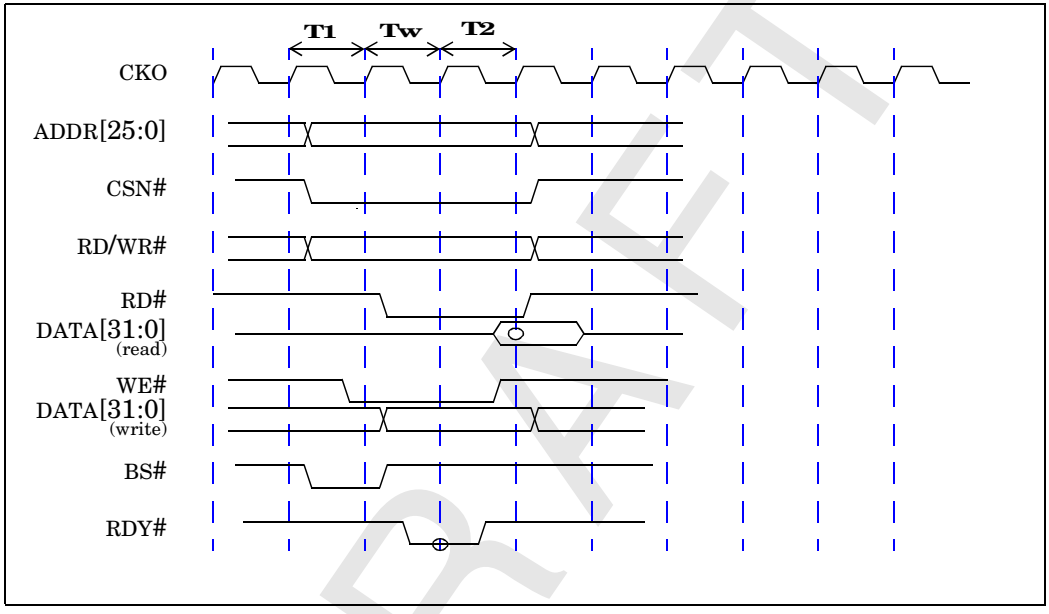

Figure 4: Normal memory interface (one wait by software)

Therefore, the RDY# signal has no effect if asserted in the T1 cycle or the first TW cycle.

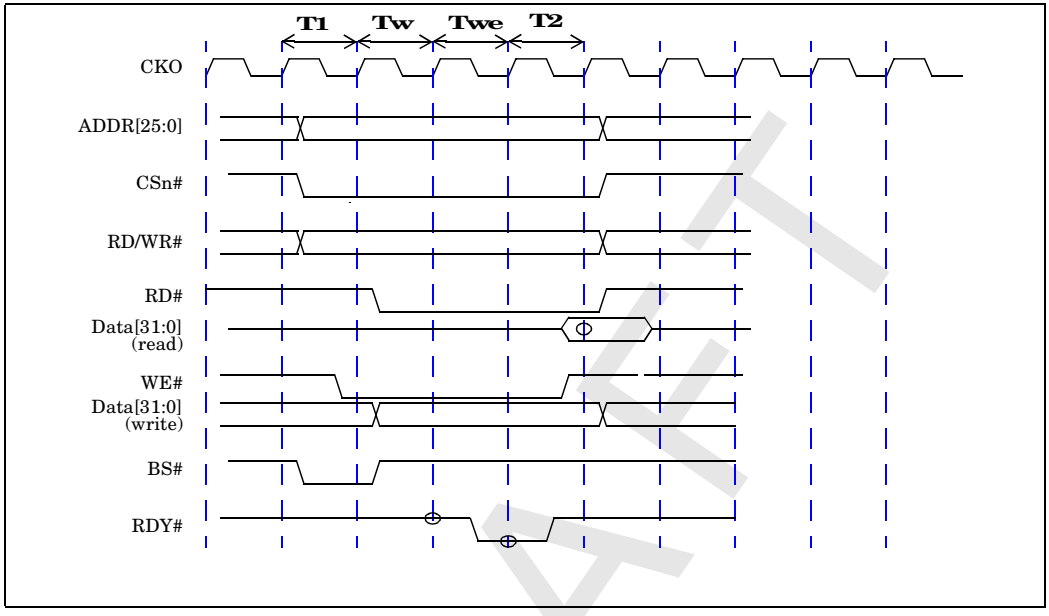

Figure 5: Normal memory interface (one wait by software and one wait by RDY#)

Normal memory interface (byte control SRAM)

The byte control SRAM interface outputs a byte select strobe (BEn#) for both read and write bus cycles. This interface can be specified by the following settings:

ANMCR.TYPE=0, ANMCR.BST=000 and ANMCR.MBC=1

Read

In a read access, only the BE# signal being read is asserted, synchronizing with the fall of CKO clock. The BE# is negated synchronizing with the rise of the clock.

Write

The byte control SRAM write timing is the same as for the normal SRAM interface.

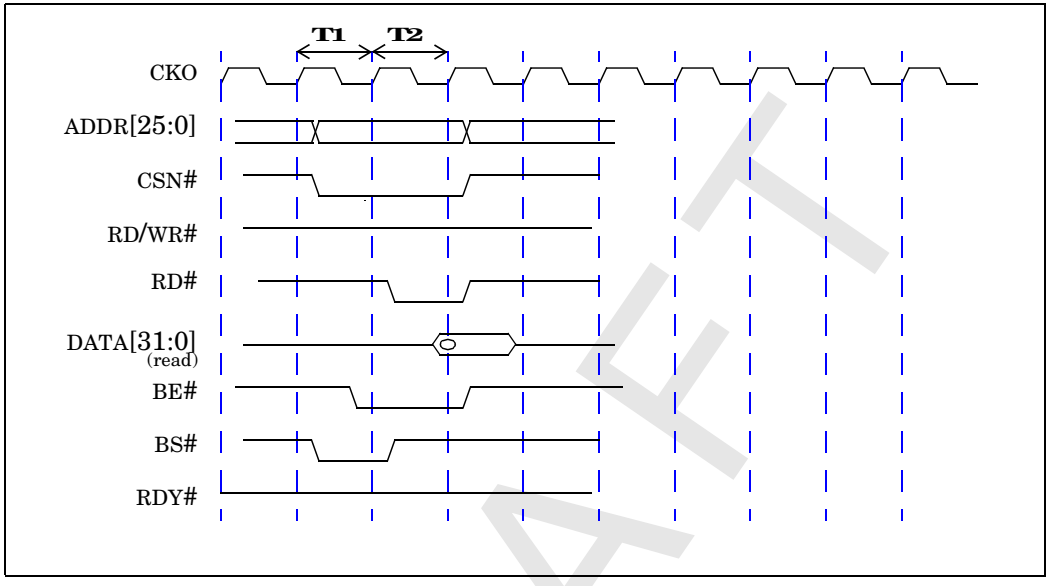

Figure 6: Byte control SRAM read cycle (no wait)

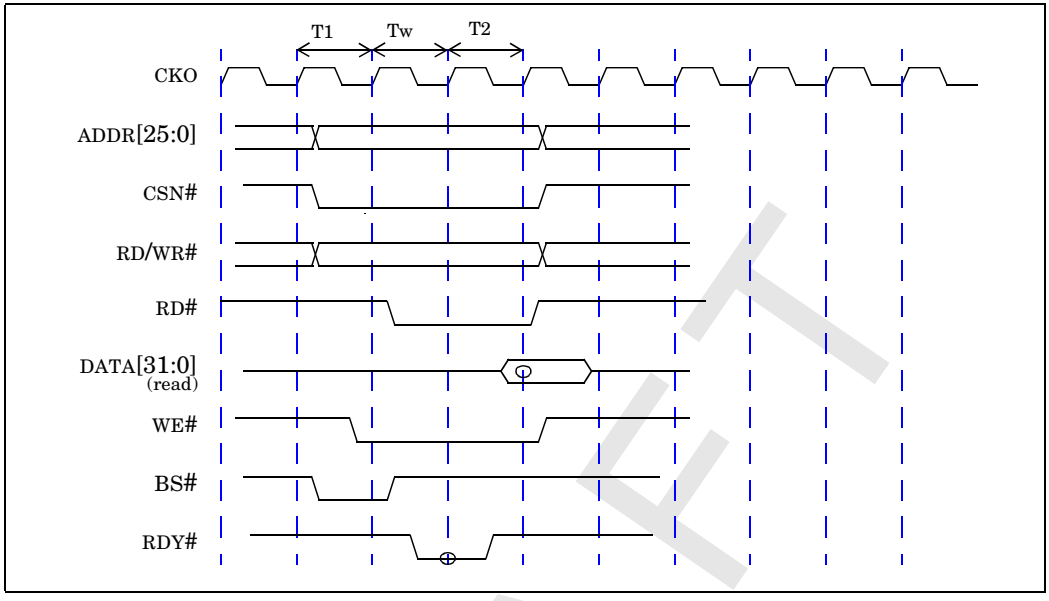

Figure 7: Byte control SRAM read cycle (one wait by software)

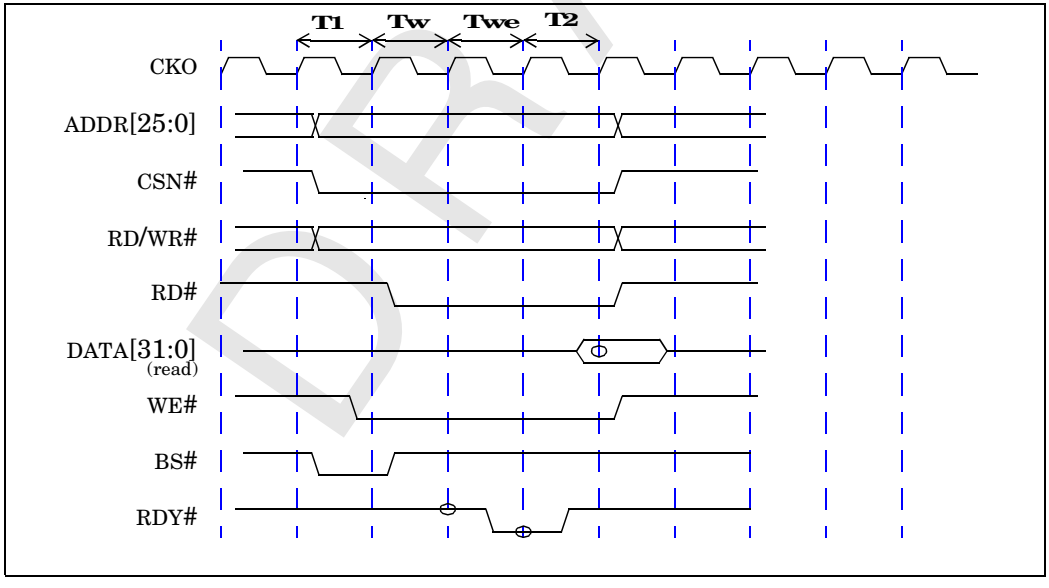

Figure 8: Byte control SRAM read cycle (one wait by software and one wait by RDY#)

Flash memory interface

Operation mode

The flash memory interface can be specified by setting ANMCR..TYPE to 0 and ANMCR..BST to 001, 010, 011 or 100. The flash memory interface can support asynchronous single read. The protocol is the same as that of the normal memory interface. When a single read operation is needed, the normal memory interface is specified by setting ANMCR.TYPE to 0 and ANMCR.BST to 000.

The flash memory interface can also support two fast read modes, asynchronous page mode and synchronous burst mode to read data from main data blocks. The ANMCR.FLMD bit can specify one of the two fast read modes (0: synchronous burst mode, 1: asynchronous page mode).

The flash memory interface supports only asynchronous single write. The protocol is the same as that of the normal memory interface. When a single write operation is needed, the normal memory interface is to be specified by setting ANMCR.TYPE to 0 and ANMCR.BST to 000.

Write protection

The FEMI has an external output pin (WP#). The status of WP# is specified by ANMCR..FLWP.

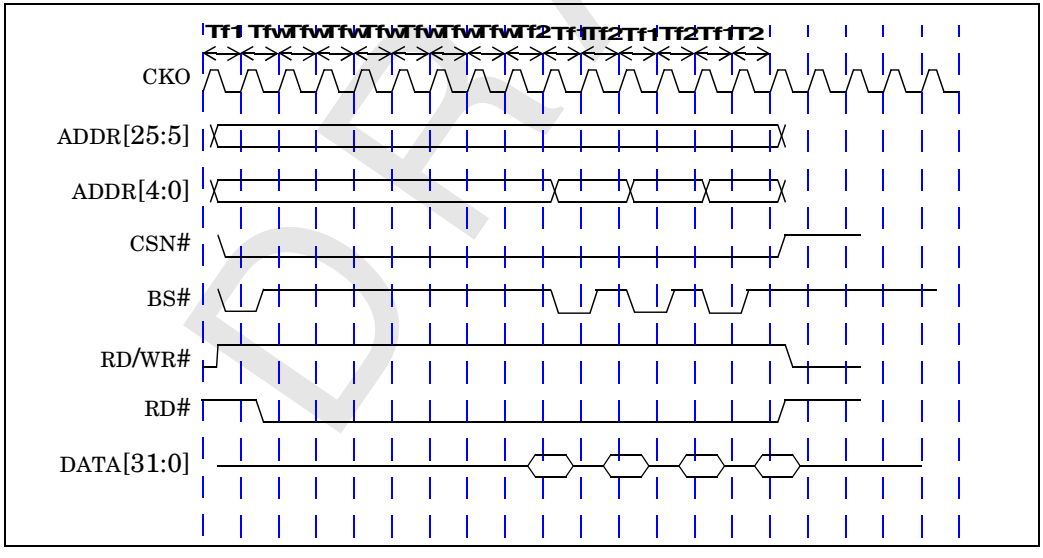

Figure 9: Flash memory access timing (asynchronous, 16B read)

Setting of AnMCR:

1 TYPE=0 (normal memory),

2 SZ=10 (32b),

3 BST=001 (flash memory with 4 consecutive),

4 FLMD=1 (asynchronous),

5 BP=010 (pitch 2),

6 WS=110 (wait 7),

7 HLD=0 (hold wait 0),

8 SETUP=0 (setup wait 0).

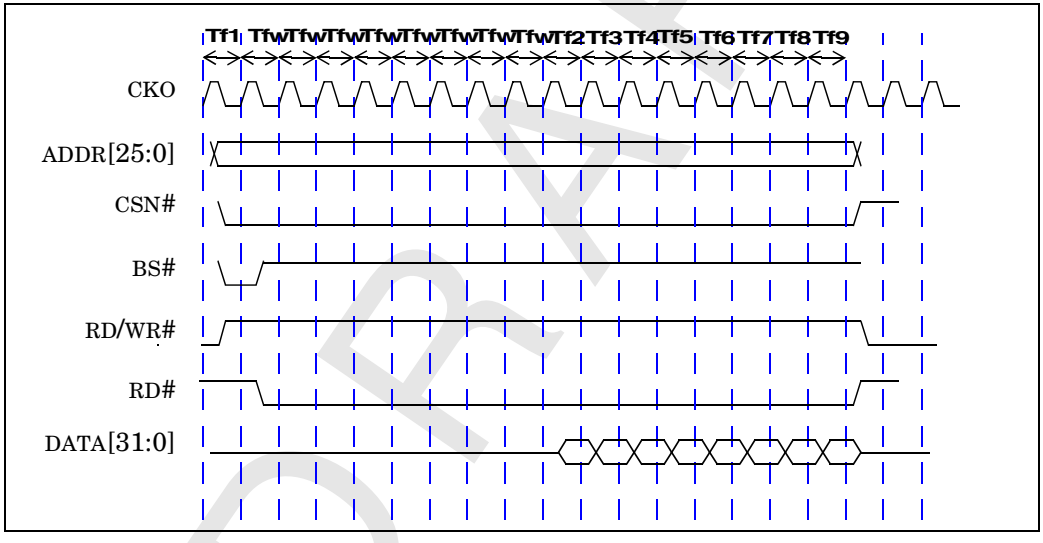

Figure 3 Flash memory access timing (synchronous, 32B read)

Setting of related fields of AnMCR:

1 TYPE=0 (normal memory),

2 SZ=10 (32b),

3 BST=010 (flash memory with 8 consecutive),

4 FLMD=0 (synchronous),

5 WS=111 (wait 8),

6 HLD=0 (hold wait 0),

7 SETUP=0 (setup wait 0)

MPX interface

The MPX interface offers a multiplexed address/data type protocol and enables easy connection to an external memory controller chip that uses a single 32-bit address/data bus.

This interface can be enabled by setting ANMCR..TYPE to 1. When the MPX interface is specified, the ANMCR..SZ must be 10 (32 bit). The address is output to DATA[25:0] and the access size to CMD[2:0] during the first part of the transfer, whilst the data is output/accepted in subsequent phases.

An external device connected with the FEMI via the MPX interface can access resources mapped on the SuperHyway bus such as the EMI's local memory. The address is output to DATA[25:0] and the access size to CMD[2:0]. The address outputs at ADDR[25:0] are undefined.

Definition of CMD[5:3]

CMD[5:3]	Area
000	0
001	1
010	2
011	3
100~111	4

Definition of CMD[2:0] during an MPX transfer

CMD[2:0]	Access size
000	8-bit
001	16-bit
010	32-bit
011	64-bit
100~111	32-bit burst

Note: The command information is output on D63-D61 in the SH-4's MPX interface. The CMD[2:0] of the FEMI is to be connected with I/O63-I/O61 of an MPX device developed for the SH-4.

If the FEMI is bus master:

The FEMI is accessed as a target on the SuperHyway bus.

The FEMI accesses memory and external devices as a bus master.

If the FEMI is a bus slave:

If an external device is a bus master, it may use the FEMI to access other system devices via the SuperHyway in addition the other components attached to the FEMI.

If it is the external device, it may access other external memory devices on the FEMI bus directory, or other system resources indirectly via the FEMI. Indirect access from the FEMI to locations mapped onto the FEMI (any area) are forbidden and will lead to undefined behavior.

The device can access other resources mapped on the SuperHyway space via the FEMI. The CMD[5:3] signal of the external device is driven to notify which device is accessing the FEMI. The device must negate CS[4:0]# to 5'b11111 and assert BS# to indicate its access to the FEMI.

When a memory space cached by the CPU is accessed by the device, the FEMI can ensure the accesses within a specific memory window are cache coherent. This functionality is controlled by the SNPCR0/1 and the SNPAR/1.

The SNPCR specifies how an address requested by the external device is compared with the address contained within the register SNPAR.

The parameters specified by the SNPCR are:

- Enable/disable address comparison. If disabled no coherency commands are issued.
- Address range to be compared (4 kbyte, 64 kbyte, 1 Mbyte, 16 Mbyte)
- Action taken on an address-hit and on an address-miss (snoop command issued on a hit or miss)
- Area including address space with which SNPAR is to be compared

Two examples of FEMI connections to an external MPX device are shown below:.

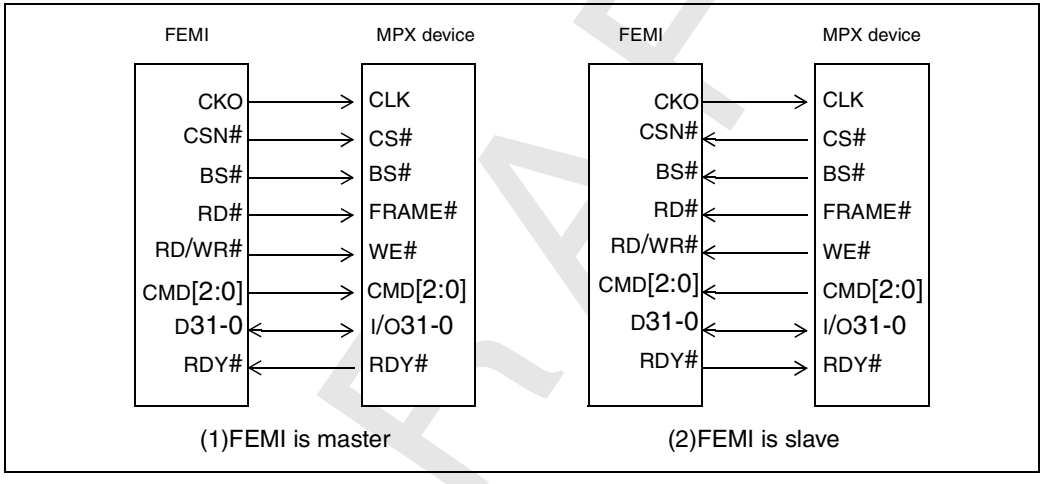

Figure 10: MPX interface connection example

Note: When the FEMI is master, CMD*[5:3] is undefined. When the FEMI is slave, a master device should drive* CMD*[5:3] to indicate the area with which it is connected.*

Wait control

Internal wait states can be inserted as the ANMCR.ws specifies and external wait states can be inserted by the RDY# pin. When the RDY# is asserted in a cycle n, it indicates that the cycle (n+1) following the cycle n is the valid data cycle

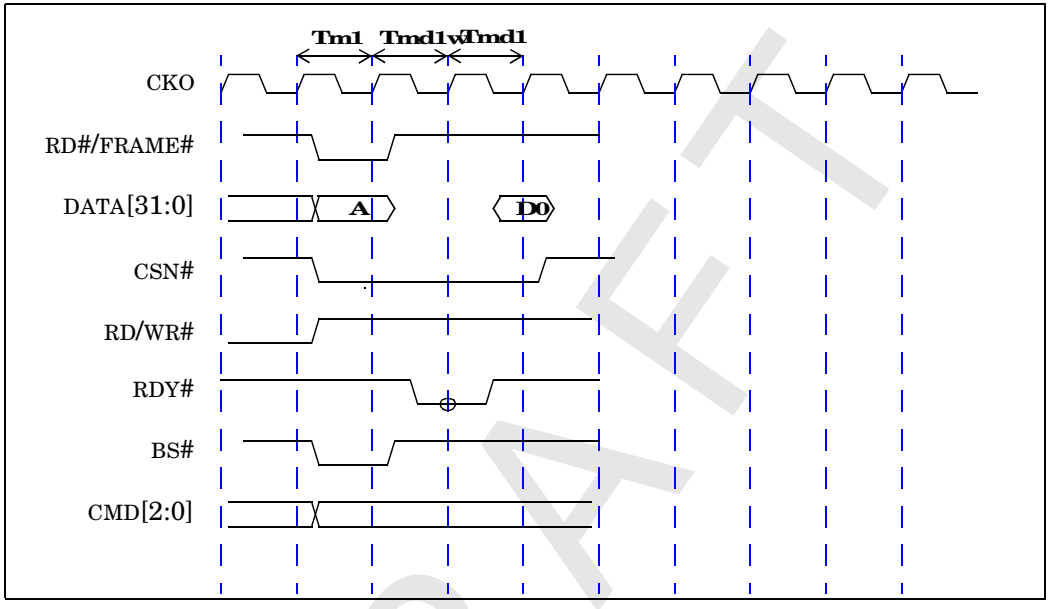

Figure 11: MPX interface timing 1 (single read Cycle, AnMCR.ws=000, 001,100, 101)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CND[5:3} is driven by an external device when it is a master. When the FEMI is master, CND[5:3] is undefined.

ANWCR.WS=000, 001, 100, or 101: one internal wait for the 1st read. The D0 is delayed by one cycles by internal wait.

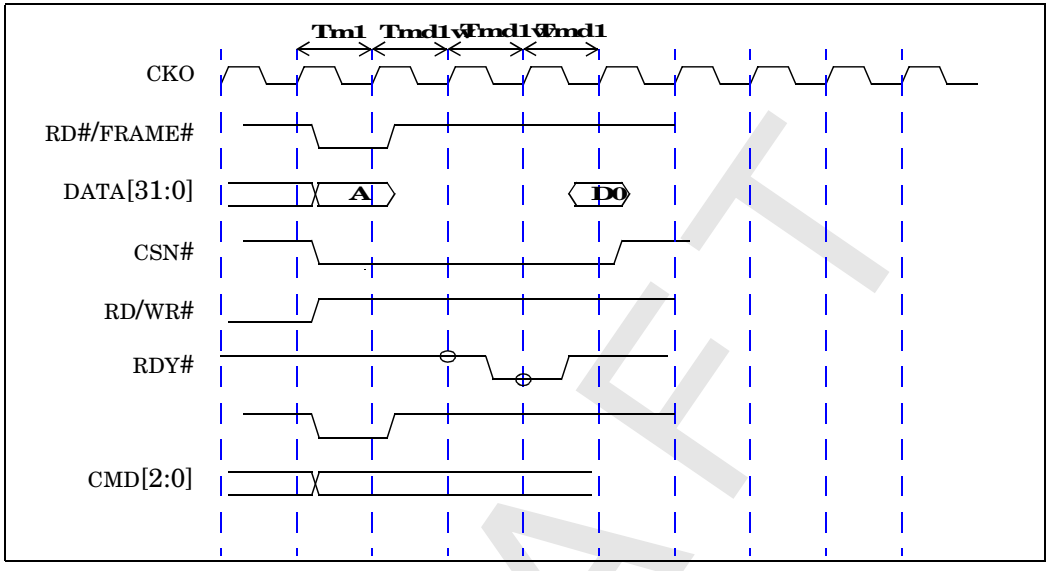

Figure 12: MPX interface timing 2 (single read cycle, ANMCR.WS=000, 001, 100, or 101)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is master, CMD[5:3] is undefined.

ANWCR.WS=000, 001, 100, or 101: one internal wait for the 1st read. The D0 is delayed by two cycles, one by internal and another by the RDY#.

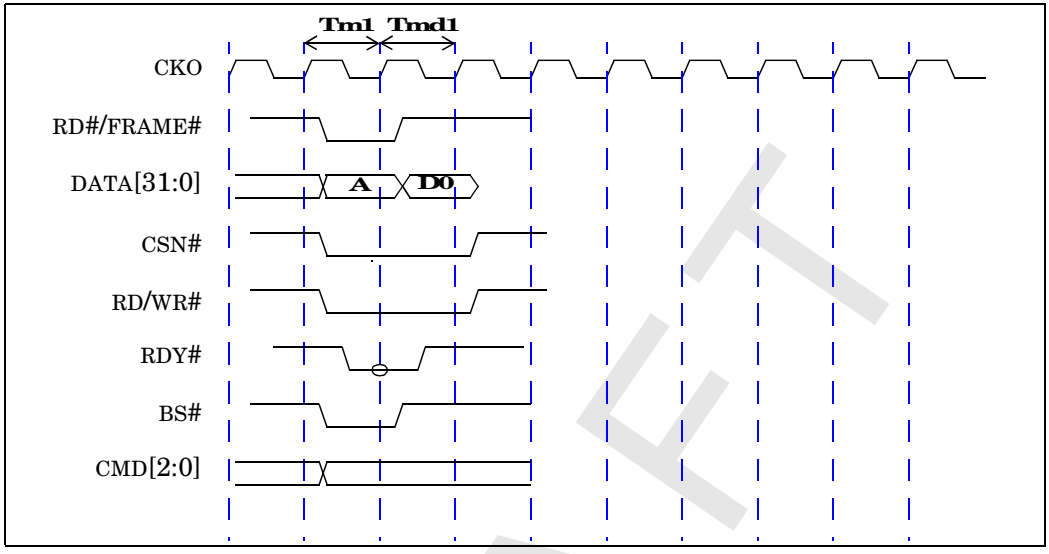

Figure 13: MPX interface timing 3 (single write cycle, no wait, AnMCR.ws=000 or 100)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is master, CMD[5:3] is undefined.

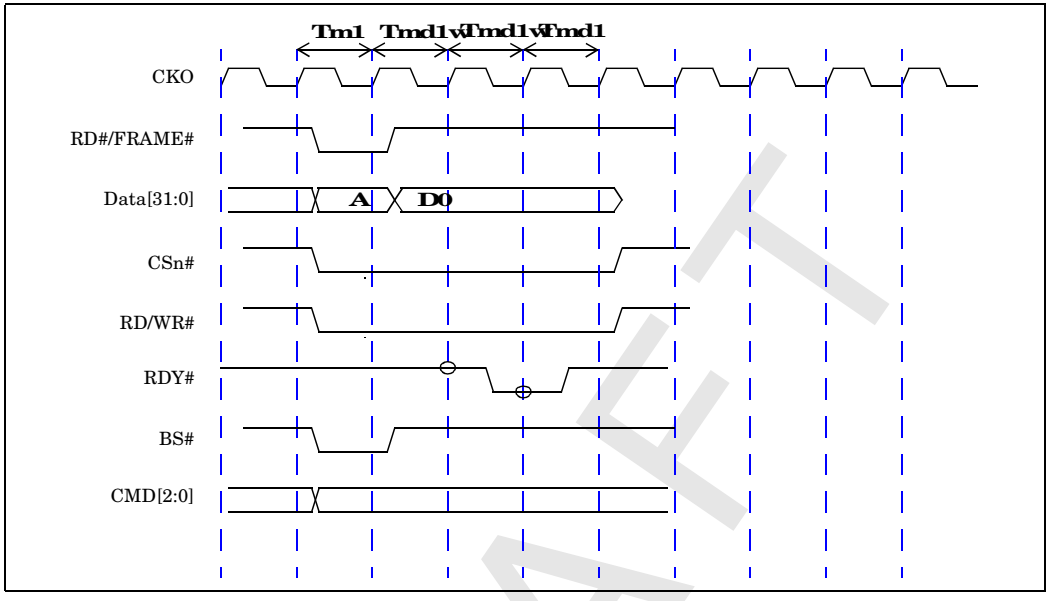

Figure 14: MPX interface timing 4 (single write cycle, AnMCR.ws=001 or 101)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is MASTER, CMD[5:3] IS UNDEFINED.

ANWCR..WS=001 or 101: one internal wait for the 1st write. The D0 is delayed by two cycles, one by internal and another by the RDY#.

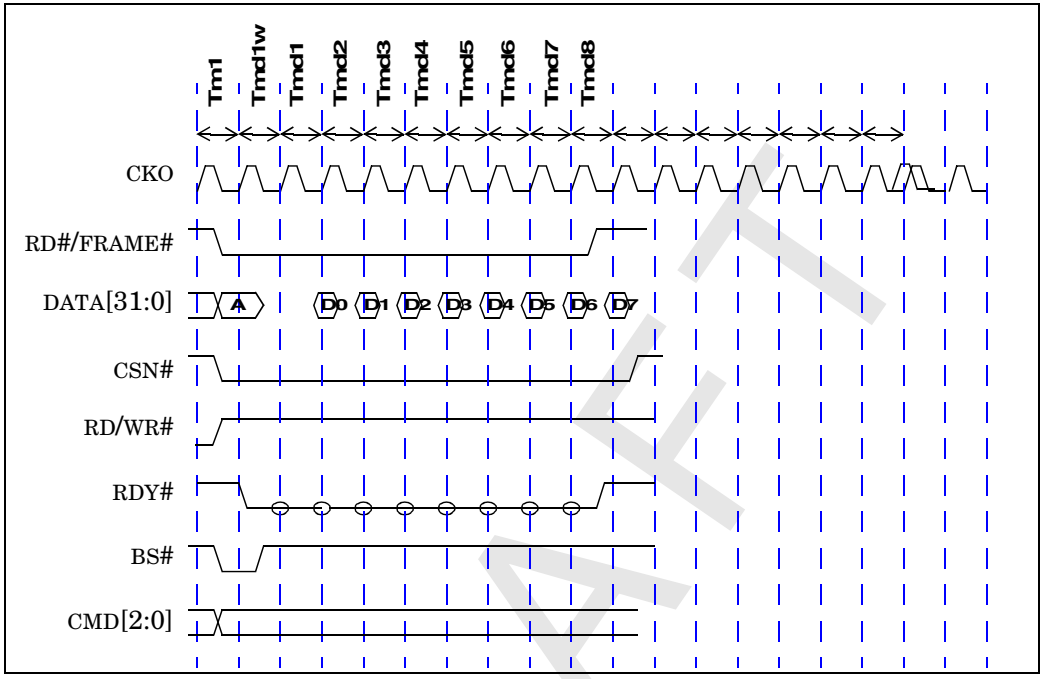

Figure 15: MPX interface timing 5 (burst read cycle, no wait, AnMCR.ws=000)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is master, CMD[5:3] is undefined.

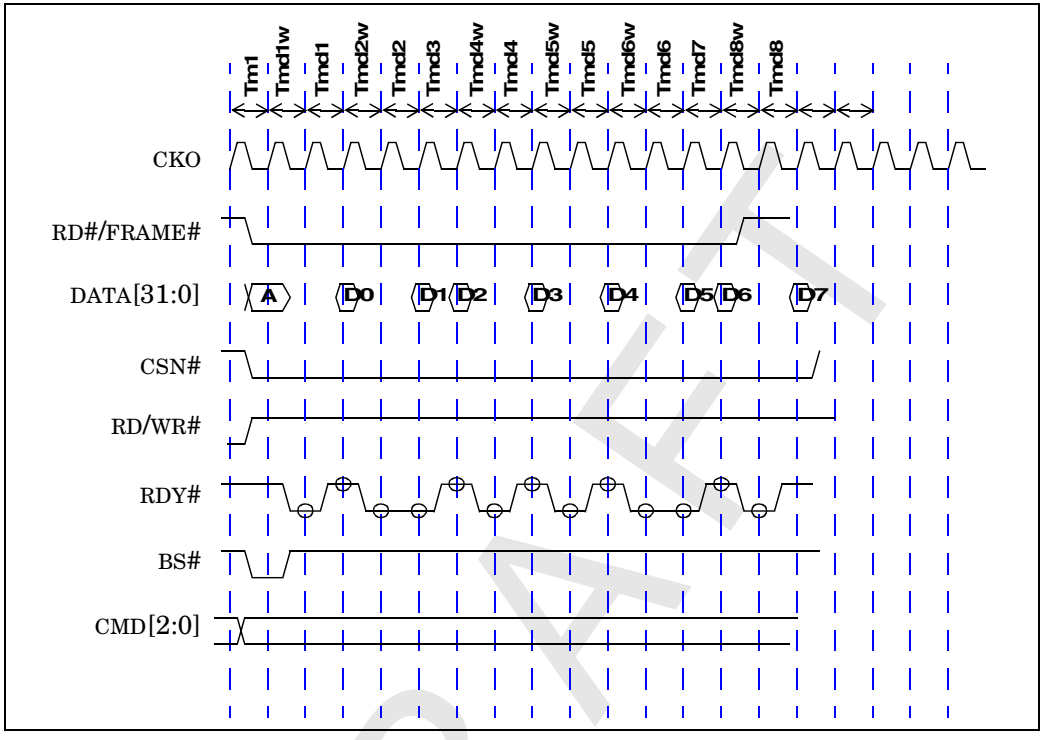

Figure 16: MPX interface timing 6 (burst read cycle, AnMCR.ws=000 or 001)

- CS# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSn#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is master, CMD[5:3] is undefined.

AnWCR.ws=000 or 001: one internal wait for the 1st read and no internal wait thereafter. The D1, D3, D4, D5 and D7 are delayed by one cycle by RDY#.

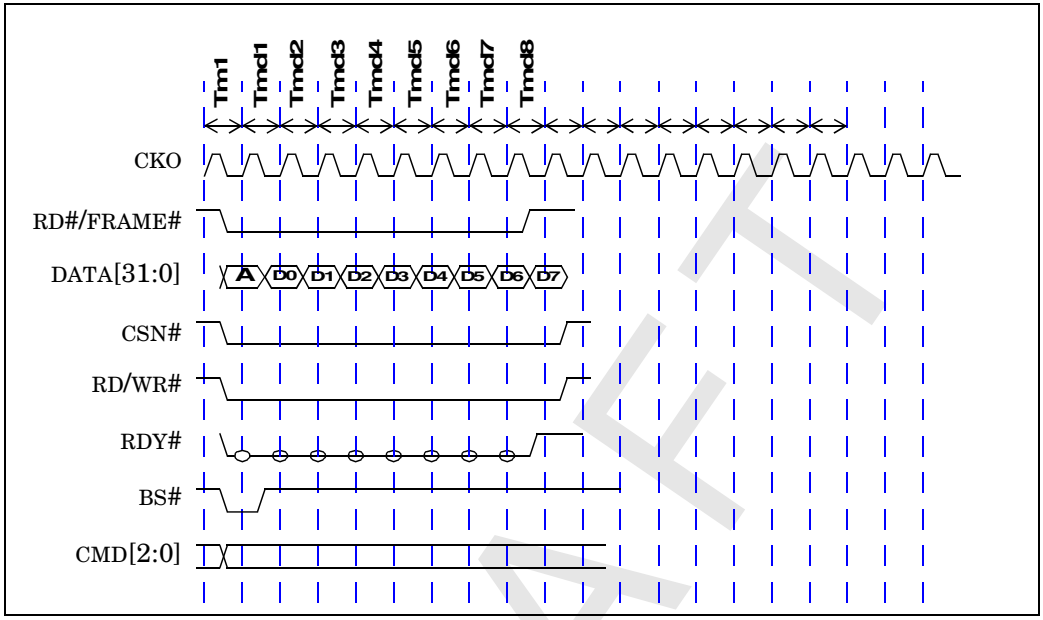

Figure 17: MPX interface timing 7 (burst write cycle, no wait, AnMCR.ws=000)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is master, CMD[5:3] is undefined.

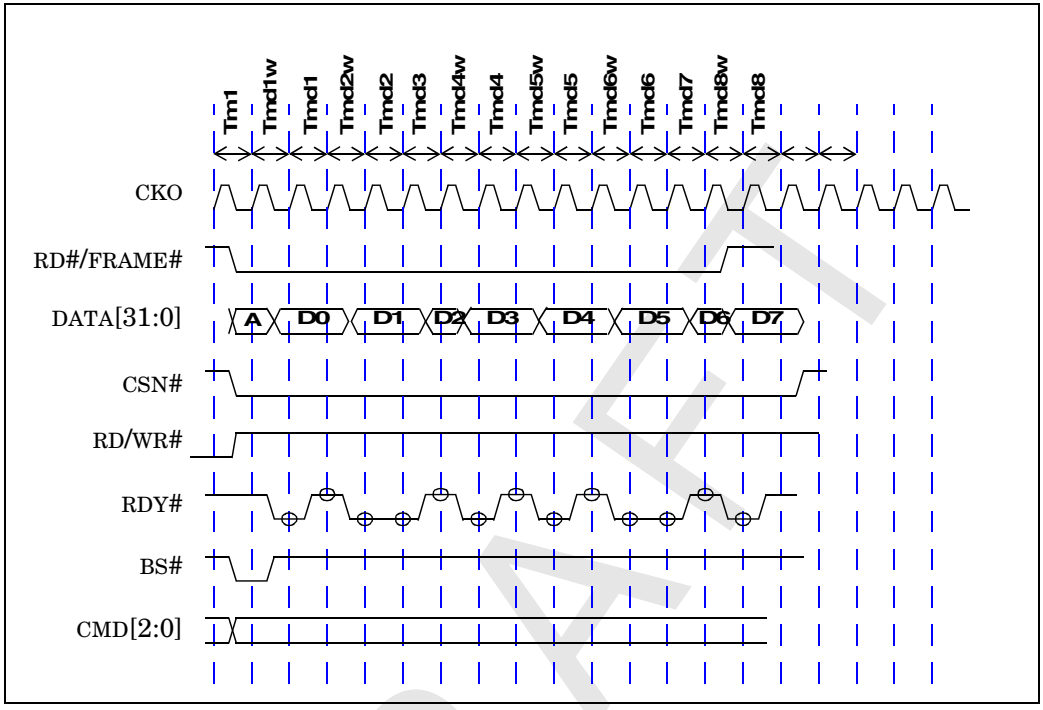

Figure 18: MPX interface timing 8 (burst write cycle, AnMCR.ws=001)

- CSN# is driven by the FEMI when it is a master. When an external device is a master, it doesn't drive CSN#.
- CMD[5:3} is driven by an external device when it is a master. When the FEMI is master, CMD[5:3] is undefined.

ANMCR.WS=001: one internal wait for the 1st write and no internal wait thereafter. The D1, D3, D4, D5 and D7 are delayed by one cycle by the RDY#.

Waits between access cycles

Both low and high speed devices may be connected to the external memory bus, leading to a potential issue with data collisions between accesses due to slow turn-off times associated with low speed device.

To avoid this problem, a feature to prevent data collision is provided by the FEMI. The FEMI keeps track of the area accessed and access type (read/write) so that it can detect the collision case and insert idle cycles. The number of idle cycles inserted is specified by the ANMCR..IW. In the following two cases, no idle cycle is inserted.

- When the FEMI performs consecutive write cycles, the data transfer direction is fixed (from the FEMI to other devices) and there is no collision problem.
- When the FEMI performs read cycles from one area, it is assumed that data is output from one data buffer, so there should be no collision problem.

When bus arbitration is performed, the bus is released after idle cycles are inserted.

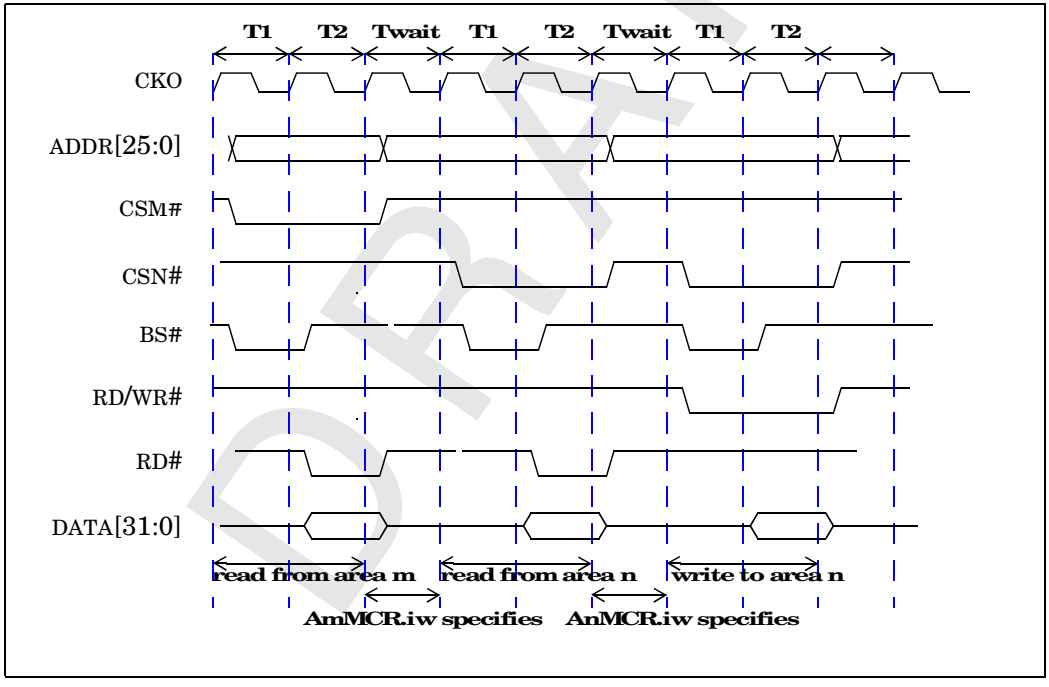

Figure 19: Waits between access cycles

Bus arbitration

The FEMI supports a bus arbitration function that permits an external device to make a bus request. The FEMI is bus master after a power-on reset.

Signal states in bus mastering exchange

To ensure safe operation when exchanging the bus master, the FEMI negates all control signals before it is transferred to the external device. It will then stop driving those signals after the exchange of bus master is complete.

This means a period exists when both devices are driving the external control signals, however, contention is avoided as both devices are driving the signals to the same inactive value.

This method ensures

(a) there is no high impedance period during the hand-over,

(b) pull-up registers on the control signals are not required.

This helps improve the hand-over performance, reduces the system component count and control signal loading.

Arbitration sequence

The transfer of the bus mastering is performed between bus cycles.

When the bus release request signal (BREQ#) is asserted, the FEMI releases the bus as soon as the currently executing bus cycle ends, and asserts the bus permission signal (BACK#).

The bus release is not performed during a burst transfer, or between multiple bus cycles generated where the bus width is smaller than the access size, that is, when an 8 byte access is requested to 16-bit memory.

When the BREQ# is negated, the BACK# is negated and the bus is resumed to be used.

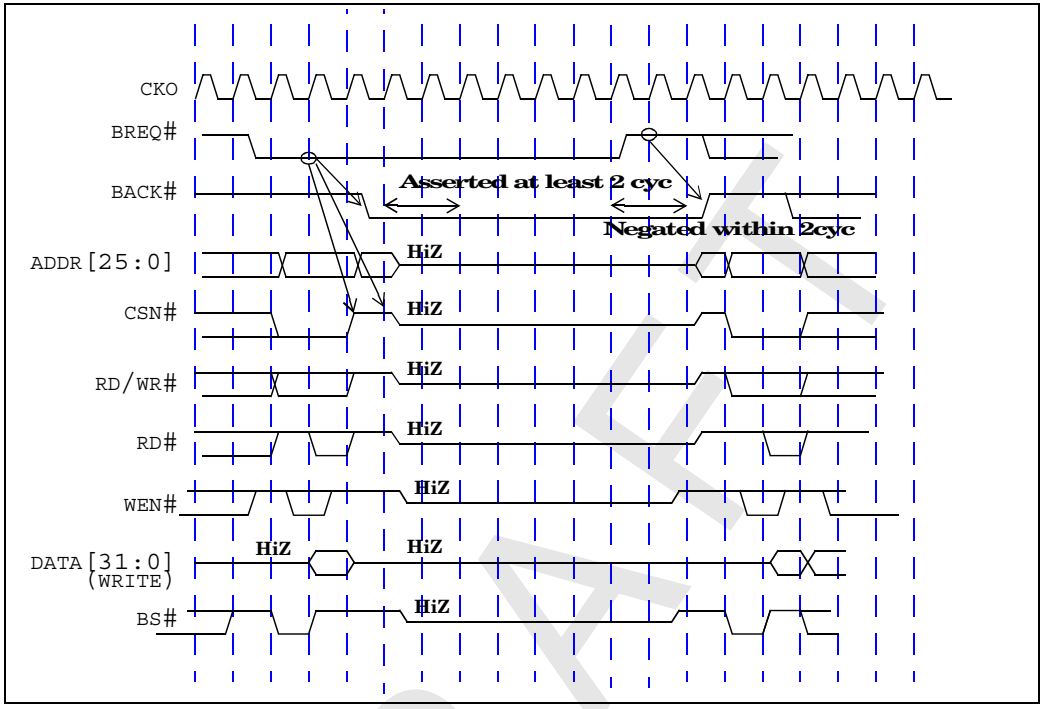

Figure 20: Arbitration sequence

3 PCI bus bridge

3.1 Overview

The SH-5 eval chip integrates a fully featured PCI 2.1 bus bridge. This chapter describes the operation of the PCI bridge and how it interfaces with the rest of the system.

The main features of the bridge are:

- Provision of a channel for a SuperHyway initiator to access PCI devices on the external PCI bus.
- Provision of a channel for PCI devices to access the address space of SH-5, particularly the system memory through the EMI.
- Support for high-speed DMA-controlled transfer of data between the SuperHyway address space and PCI address space.
- Support for host and normal modes of operation on the PCI bus.

Typically, the SH-5 PCI bridge will act as a host bridge, with the SH-5 CPU as the host processor. This is called *host bus bridge mode*. The remainder of this chapter will assume host bus bridge mode, except where noted. There are modes of operation in which the SH-5 PCI bridge will act as a *normal mode* PCI device in a system where another processor is the host (for example, a PC). In this case, the SH-5 is visible as a standard PCI master and a standard PCI target.

This chapter assumes some familiarity with PCI bus operation. For background reading on PCI, see *Section 3.10: References on page 221*.

3.1.1 Features

- 0-33 MHz or 66 MHz operation.
- 32-bit data path.
- PCI master and target functions.
- Power management rev 1.1 support.
- Host bus bridge mode and normal mode support.
- PCI bus arbiter (in host bus bridge mode),
  - supports 4 external masters,
  - pseudo-round-robin or fixed priority arbitration.
- Configuration mechanism #1 support (in host bus bridge mode) see PCI local bus specification revision 2.1.
- Burst transfer support.
- Parity check and error report.
- Exclusive access (both master and target) although exclusive access between internal initiator and external PCI master is not supported,
  - Master: memory or I/O transfer only,
  - Target: once locked, only accessible from owner of LOCK#,
    - no guarantee for SuperHyway resource lock,
    - (accessible by other SuperHyway modules during lock transfer).
- Hardware supported cache coherency between PCI bus and SH-5 CPU when it is in host mode.
- In host bus bridge mode support four external interrupt inputs (INT[A:D]#).
- In normal mode support one external interrupt output (INTA#). Control by PCI control register bit PCI_IOCS.
- Two clock sources: SuperHyway clock and PCI clock. Clocks can be asynchronous.

3.1.2 PCI features not supported

- cache support (no SBO# SDONE pins),
- address wrap-around mechanism,
- PCI JTAG (other module can support JTAG feature),
- dual address cycles,
- interrupt acknowledge,
- fast back-to-back transfer initiation (supported when performed as a target device).

3.1.1 Supported PCI commands

C/BE[3:0]#	Command type	PCI master	PCI target
0000	Interrupt acknowledge	8	Ignore
0001	Special cycle	✔a	Ignore
0010	I/O read	4	✔b
0011	I/O write	4	✔B
0100	Reserved	Ignore	Ignore
0101	Reserved	Ignore	Ignore
0110	Memory read	4	4
0111	Memory write	4	4
1000	Reserved	8	Ignore
1001	Reserved	8	Ignore
1010	Configuration read	✔A	✔B
1011	Configuration write	✔A	✔B
1100	Memory read multiple	8	✔c
1101	Dual address cycle	8	Ignore

Table 25: Supported PCI bus commands

C/BE[3:0]#	Command type	PCI master	PCI target
1110	Memory read line	8	4c
1111	Memory write and invalidate	8	✔d

Table 25: Supported PCI bus commands

a. In host bus bridge mode

b. Single transfer only

c. Aliased to memory read

d. Aliased to memory write

3.1.2 PCI signal description

Figure 3 illustrates the external signals used by the SH-5 interface. A complete functional description of the pin list is given in *Section 3.9 on page 219*.

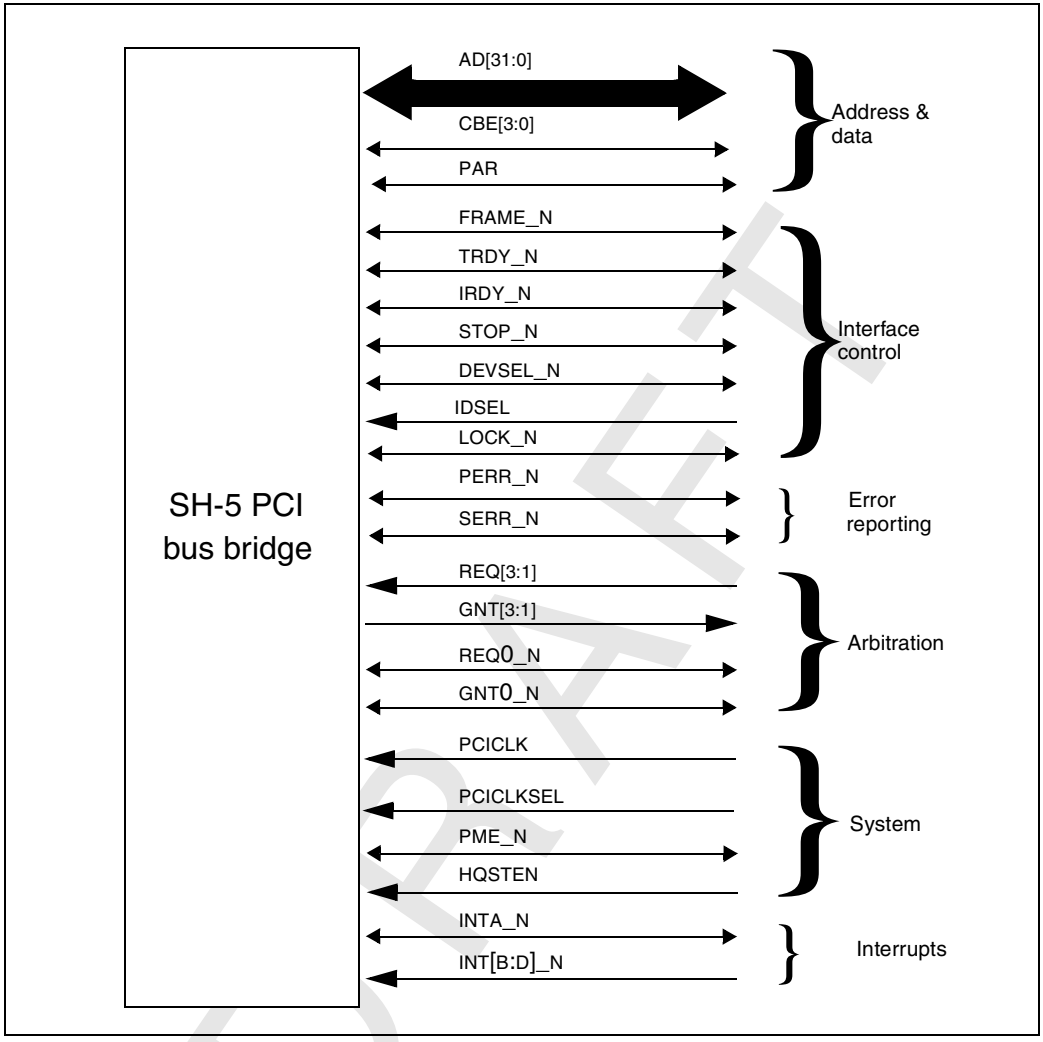

Figure 3 PCI bus pin list

3.1.3 PCI bus bridge module block architecture

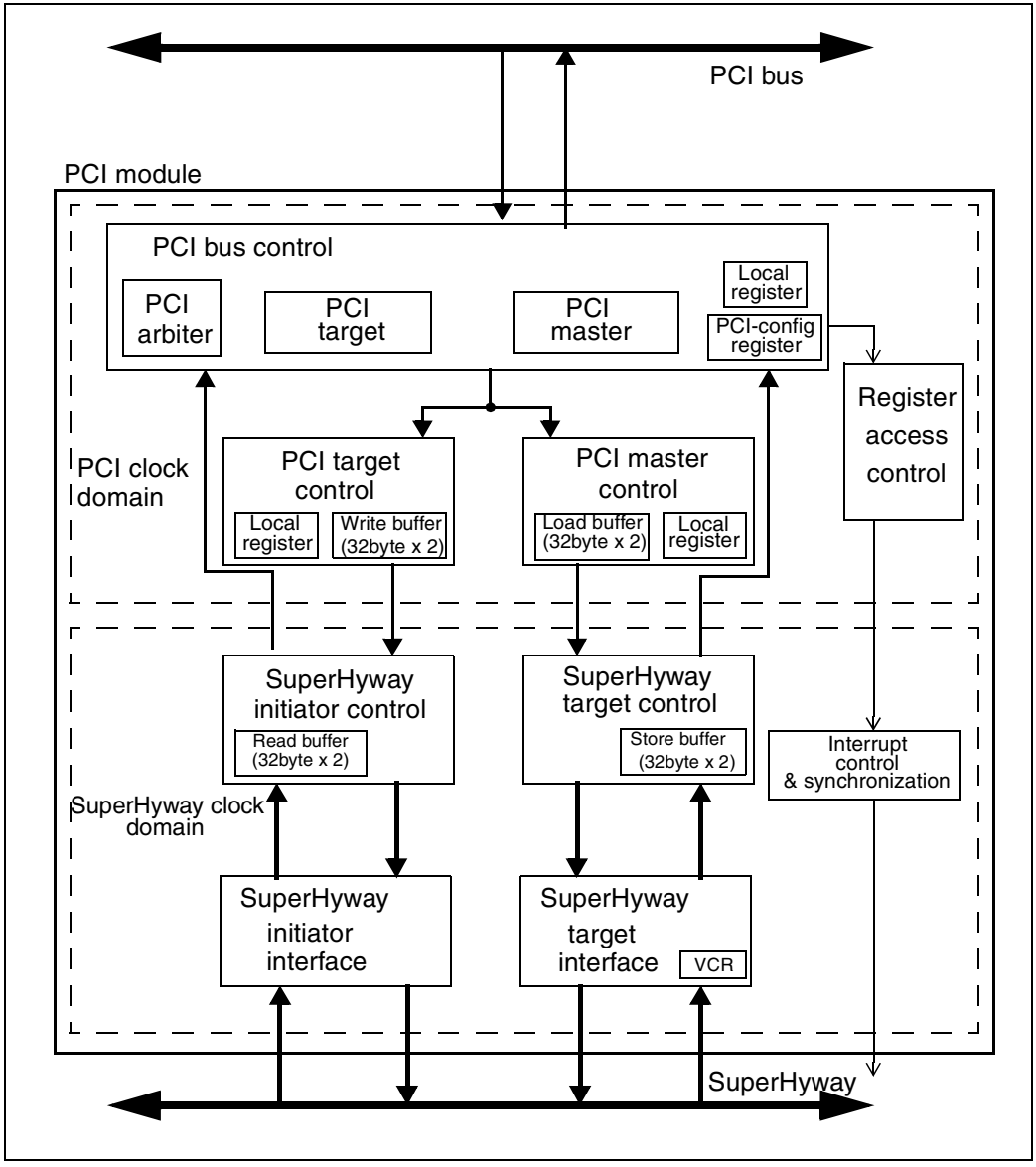

Figure 21: PCI module block diagram

The micro-architecture of the PCI bus bridge module is shown in *Figure 21 on page 110*. The function of the internal blocks are described as follows.

PCI-bus control Implements the PCI bus-level protocol. There are separate blocks which control transactions in which the SH-5 is the master and those in which the SH-5 is the target. Accesses to the PCI memory space are buffered to improve efficiency.

PCI bridge The PCI bridge has two clock domains: the PCI clock domain and SuperHyway clock domain. Both the PCI configuration register and the PCI local register can be accessed from the PCI bus when the SuperHyway clock is stopped (PCI clock is running).

SuperHyway initiator interface Receives request packets from the PCI bus bridge, and issues corresponding commands to the SuperHyway. The SuperHyway initiator interface also receives response packets from SuperHyway.

SuperHyway target interface Responsible for:

- the store buffers,
- VCR register,
- reception of request packet from the SuperHyway,
- for issuing the response packet to the SuperHyway,
- for issuing request packets to the PCI bus bridge.

Store buffers Enables use of burst transfer of PCI.

Register access control Responsible for the read and write access of the PCI configuration register and PCI local register.

Interrupt control Governs the assertion of interrupts to INTC of SH-5.

3.1.4 Address map

The PCI bridge memory map is shown below in *Table 26* .

Memory area	Base address	Address space size - bytes
PCI memory	0x4000 0000	512 M
Version control register (VCR)	0x6000 0000	8
Reserved	0x6000 0008	256 K - 8
PCI control registers	0x6004 0000	256 K
Reserved	0x6008 0000	7.5 M
PCI I/O	0x6080 0000	8 M

Table 26: PCI bridge memory map

The PCI control registers are implemented as two banks: SH-5 local register bank and the PCI configuration and status register (CSR) bank. The CSR bank format complies with the format prescribed in the PCI standard. The layout is illustrated in *Table 27*. The local register bank contains those registers which are specific to the PCI bridge and, in particular, control how the PCI is integrated into the SH-5 system. A complete register listing for this PCI bridge is given in *Section 3.8 on page 215*.

31:24	23:16	15:8	7:0	Offset[a]
DeviceID		Vendor ID		0x00
Status		Command		0x04
Base class code	Sub class code	Program interface	Revision	0x08
BIST	Header	Latency	Cacheline	0x0C
IO base address (IBAR)				0x10
Memory base address 0 (MBAR0)				0x14
Memory base address 1 (MBAR1)				0x18
Reserved				0x1C-0x28
Subsystem ID		Subsystem vendor ID		0x2C
Reserved				0x30
			cap_ptr	0x34
Reserved				0x38
max latency	min_gnt	Interrupt pin	Interrupt line	0x3C
Power management capability		Next item pointer	Capability ID	0x40
Power consumption/ dissipation data	Power management control/status	Power management control/status		0x44
Reserved				0x48 - 0xFF

	Read/write
	Read only
	Reserved

Table 27: PCI CSR bank layout

a. These are the config space offsets. For access from SuperHyway use base address 0x60040000

3.1.4 Interrupts

The PCI bridge is capable of signaling 10 different interrupts with unique codes to the interrupt controller module (INTC). These interrupts are shown below.

Interrupt name	Default priority	Cause	PCI maskable[a]
INT[A:D]	High	INT[A:D]# pin asserted	8
SERR#		SERR# asserted	4
ERR		PCI protocol error	4
PCI_PWRST_D[3:0]	Low	Power state transition	4

Table 28: PCI interrupts

a. Interrupts may also be masked by the interrupt controller (INTC) module.

The interrupt codes, associated with these interrupts may be found in the interrupt controller chapter.

3.1.5 Chapter organization

The PCI bridge's principal function is to convert SuperHyway transactions into PCI transactions and vice versa. These are described as follows:

- *Master transactions*: these are transactions in which the local SH-5 system is the initiator and a device on the PCI bus is the target. These are described in *Section 3.2 on page 115*. This is the view that the SuperHyway has of PCI address spaces and of the PCI bridge's register banks.
- *Target transactions*: these are transactions in which the local SH-5 system is the target and another device on the PCI bus initiates the transaction. These are described in *Section 3.3 on page 124*. This is the view that a device on the PCI has of the SH-5 address space.

Some functions, such as CSR access, arbitration and interrupts, of the PCI bridge are normally only used when the bridge is configured in *host mode*. Host mode functions are described in *Section 3.5 on page 133*.

The PCI bridge supports power management rev 1.1 of the PCI standard.

- Power management architecture is described in *Section 3.6 on page 136*.
- All registers are defined in *Section 3.7 on page 137*.
- A summary list of all registers is given in *Section 3.8 on page 215*.
- A complete pin list for the PCI bridge is given in *Section 3.9 on page 219*.

This chapter assumes familiarity with the PCI standard. A short recommended reading list is given in *Section 3.10 on page 221* for those requiring a primer on PCI.

3.2 SuperHyway view

This section describes how resources, which reside in one of the PCI address spaces, may be accessed by software running on the local SH-5 CPU or by a DMA engine.

PCI defines 3 address spaces:

- 4 Gbyte memory space
- 4 Gbyte I/O space
- 256 byte configuration space for each device on the PCI bus to support device configuration. This space is normally only configured by a host.

3.2.1 Address map

Memory area	Block type	Address range	Address space size	Access description
PCI memory	DB	0x4000 0000 0x5FFF FFFF	512 Mbyte	*Section 3.2.7 on page 117*
Version control register (VCR)	CB	0x6000 0000 0x6000 0007	8 bytes	*Section 3.2.6 on page 116*
Reserved	CB	0x6000 0008 0x6003 FFFF	256 Kbyte - 8 bytes	
PCI control registers	CB	0x6004 0000 0x6007 FFFF	256 Kbyte	*Section 3.2.6 on page 116*
Reserved	CB	0x6008 0000 0x607F FFFF	7.5 Mbyte	
PCI I/O	CB	0x6080 0000 0x60FF FFFF	8 Mbyte	*Section 3.2.9 on page 119*

Table 29: PCI bridge memory map

3.2.6 Accessing registers

All internal registers, that is, VCR, PCI configuration registers and PCI local registers, are accessible from the SuperHyway (and thus to software running on SH-5 core as well as the DMA).

Note: Only SuperHyway **load** *8-byte and* **store** *8-byte commands are supported in these registers.*

Accessing the VCR register

The VCR may be accessed using the SuperHyway address given in *Table 29 on page 116*. The VCR is defined in *Section 3.7.1 on page 138*.

Accessing local and configuration registers

The PCI local and configuration registers may be accessed using an offset from the PCI register base address. A full list of all these registers is given in*Section 3.8 on page 215*.

The configuration registers and local registers should only be accessed in their correct size. Accessing these registers in any other size is *not* defined.

3.2.7 PCI memory space (SH-5 master)

The 512 Mbyte SuperHyway address range 0x4000 0000 - 0x5FFF FFFF is referred to as the PCI memory aperture.

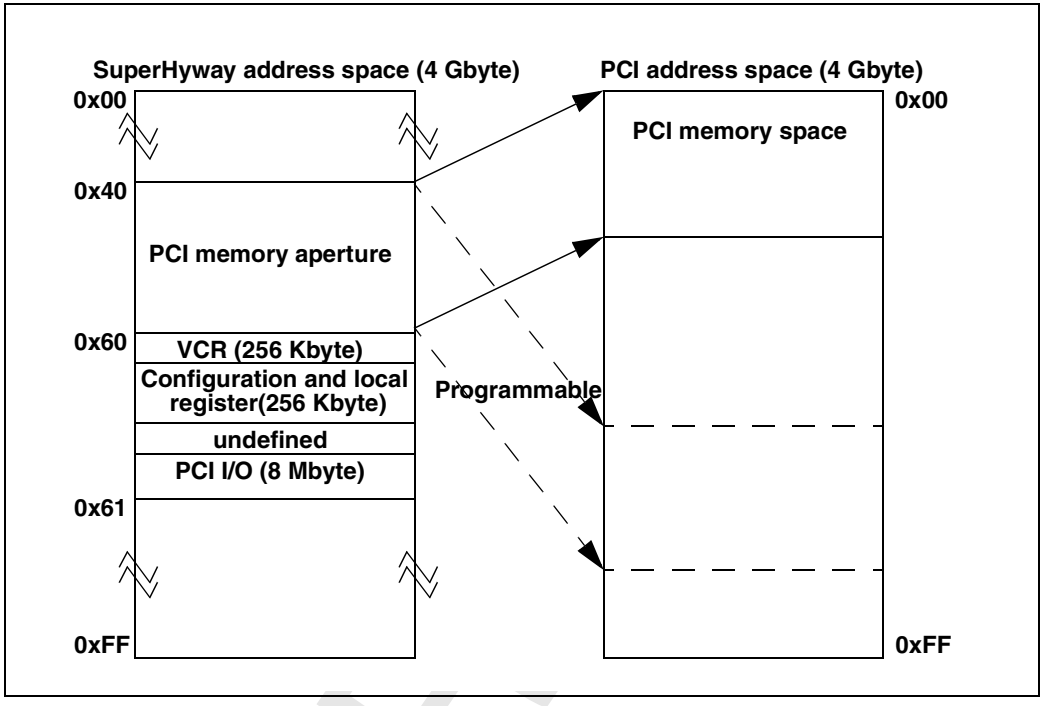

Figure 8 SuperHyway to PCI memory map

All SuperHyway memory accesses made to the PCI memory aperture are translated into accesses to the PCI memory space. The translation is controlled by two registers:

1 PCI.MBR which specifies the base address in PCI memory space to which local SuperHyway accesses are mapped.

2 PCI.MBMR is a mask which is used to specify the size of the PCI memory space.

The SuperHyway address is translated as follows:

- The bottom 2 bits of the PCI address[1:0] are always 00[1].
- The lower 16 bits ([17:2]) of the SuperHyway address[2] are copied to the PCI address [17:2] unchanged.
- The middle 11 bits ([28:18]) of the address are translated according to PCI memory bank mask register (PCI.MBMR). The mask register is a '1' in those bit positions in which the PCI address is a copy of the local address. In those bit positions in which the mask register is a '0', the PCI address is a copy of the PCI.MBR register. For example:

  PCI.MBMR[28:18] 11'b1 1111 1111 11 : PCI addr[28:18] = SuperHyway addr[28:18]

  PCI.MBMR[28:18] 11'b0 0000 0000 00 : PCI addr[28:18] = PCI.MBR[28:18]
- Upper 3 bits of the PCI address [31:29] are a copy of the upper 3 bits of the PCI memory bank register PCI.MBR[31:29].

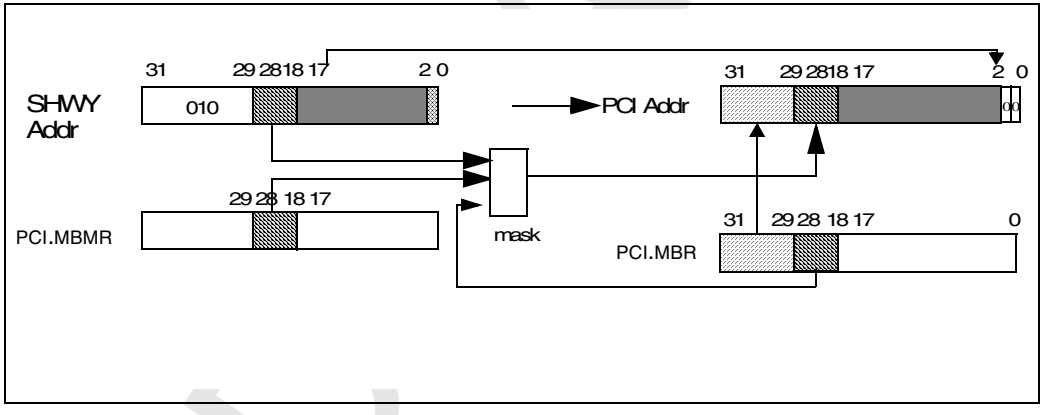

Figure 22: SuperHyway to PCI memory access

1. The PCI bridge only uses linear incrementing burst mode (see PCI standard). Although the PCI bridge manages this so that it transparent to the SuperHyway.
2. Note that the SuperHyway address may, in part, be specified by SuperHyway byte enables.

Burst transfer support for memory transfers.

The PCI bridge supports the generation of large bursts on the PCI bus by enabling continuous SuperHyway load 32-byte or store 32-byte accesses to be aggregated into a greater-than-32-byte burst transfer (that is,64-byte, 96 byte, and so on) on the PCI bus.

3.2.9 Accessing PCI I/O space (SH-5 master)

The 8 Mbyte SuperHyway address range 0x6080 0000 - 0x60FF FFFF is referred to as the PCI I/O aperture.

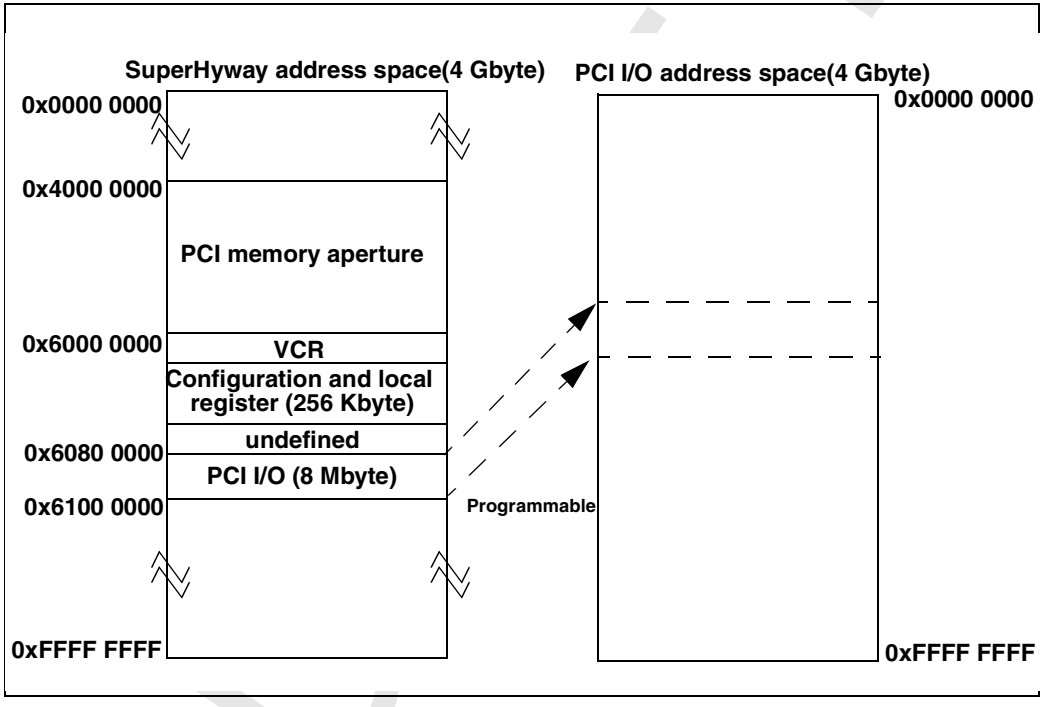

Figure 23: SuperHyway to PCI I/O map

SuperHyway memory accesses made to the PCI I/O aperture are translated into accesses to the PCI I/O space. The translation is controlled by two registers:

1 PCI.IOBR which specifies the base address in PCI memory space to which local SuperHyway accesses are mapped

2 PCI.IOBMR which is a mask used to specify the size of the PCI memory space.

The SuperHyway address is translated as follows:

- The SuperHyway address[1] lower 18 bits ([17:0]) are copied to PCI IO address [17:0] unchanged.
- The middle 5 bits ([22:18]) of the PCI I/O address are controlled by the PCI I/O bank mask register. The mask register is a '1' in those bit positions in which the PCI address is a copy of the local address. In those bit positions in which the mask register is a '0', the PCI address is a copy of the PCI.IOBR register. For example,

  PCI.IOBMR[22:18] 5'b11111 : PCI addr[22:18] = SuperHyway addr[22:18]

  PCI.IOBMR[22:18] 5'b00000 : PCI addr[22:18] = PCI.IOBR[22:18]

- The PCI I/O address upper 9 bits ([31:23]) are a copy of the upper 9 bits of the PCI I/O bank register PCI.IOBR[31:23].

This process is illustrated by below by *Figure 24*.

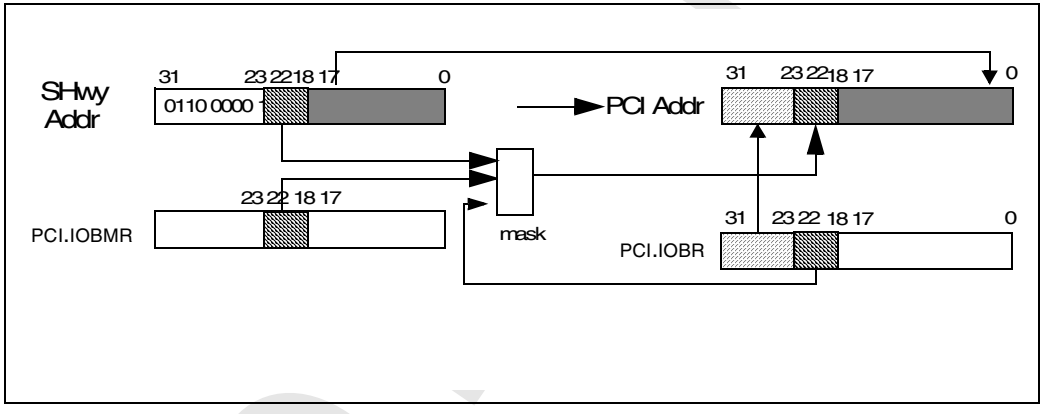

Figure 24: SuperHyway to PCI I/O access

Burst transfer support for I/O transfers.

Burst I/O transfer is not supported. That is, continuous **load8/store8** commands to PCI bridge's I/O area are not detected to be continuous in PCI bridge.

They are treated as separate transactions and do not generate more than 8-byte burst transfer on PCI bus.

1. Note that the SuperHyway address may, in part, be specified by SuperHyway byte enables.

3.2.1 Accessing PCI configuration space (SH-5 master)

Configuration space access is normally performed in *Host Mode* See *Section 3.5.1 on page 133* for details.

3.2.2 Exclusive acesss (host mode)

The SuperHyway SWAP 8-byte command (which software may cause by using the **SWAP.Q** instruction) generates lock access on PCI bus.[1]

3.2.3 Bi-endian support

The PCI bridge of SH-5 provides some support for big endian software and/or big-endian PCI address space. Since PCI is inherently little endian, both data switching and data non-switching are supported.

The configuration of the endian switching mode is decided by the PCI_TBS field of the PCI control register(PCI.CR defined in *PCI control register on page 176*).

1. Note that on the SH-5 eval implementation, only 4-byte data will be transferred to PCI devices. (Only single longword transfer is supported for lock access.). SuperHyway requestors should use the mask bits to specify which data is to be transferred. If 8 bytes are enabled, (as is the case by the **SWAP.Q** instruction) then only least significant 4 bytes (Address[2]=0)are transferred.

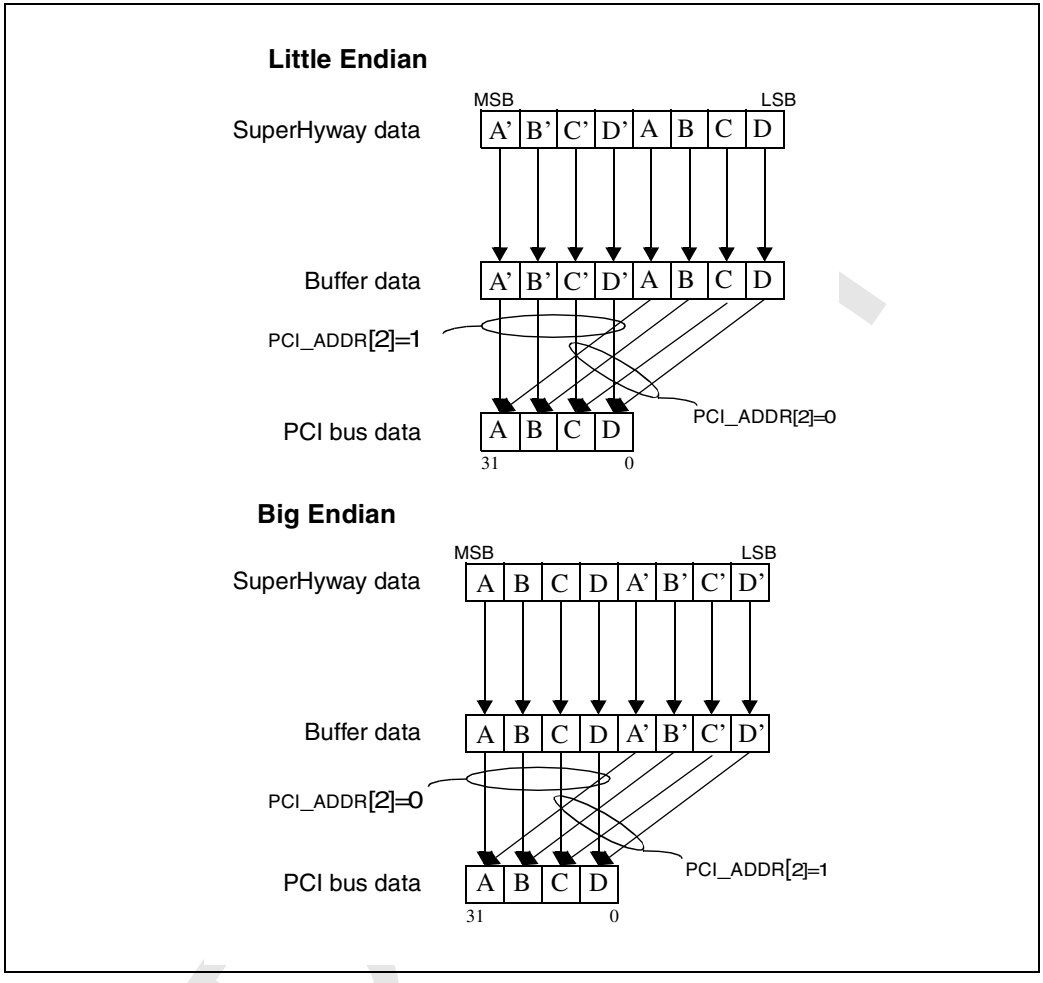

Figure 25: SuperHyway to PCI endian exchange (non-switching:PCI.TBS=0)

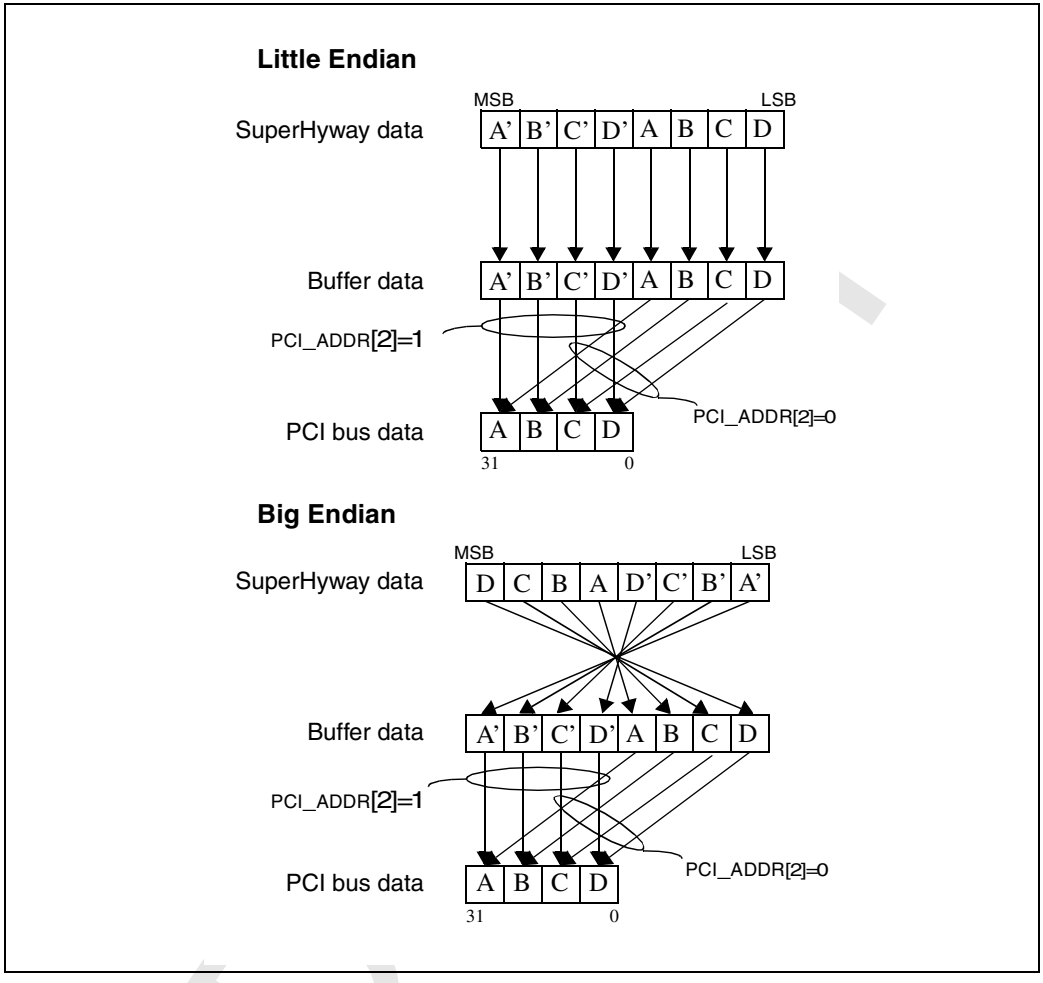

Figure 26: SuperHyway to PCI endian exchange(switching:pci.tbs=1)

3.2.4 Generating interrupts (non-host mode)

In normal mode, the INTA# pin is a software-controllable output. An interrupt may be asserted by setting the PCI_IOCS bit in the PCI.CR register. The interrupt is de-asserted by clearing the same bit. The PCI.CR register is defined in *Section on page 176*.

3.3 PCI bus view

This section describes how the PCI bridge is accessed by an external PCI master in both host mode and normal mode.

3.3.1 Address map (SH-5 target)

SH-5 presents 2 memory regions, an I/O region and the standard configuration space to the PCI bus. When the SH-5 is a target of a PCI transactions the address spaces are translated as illustrated in *Figure 27 on page 124*.

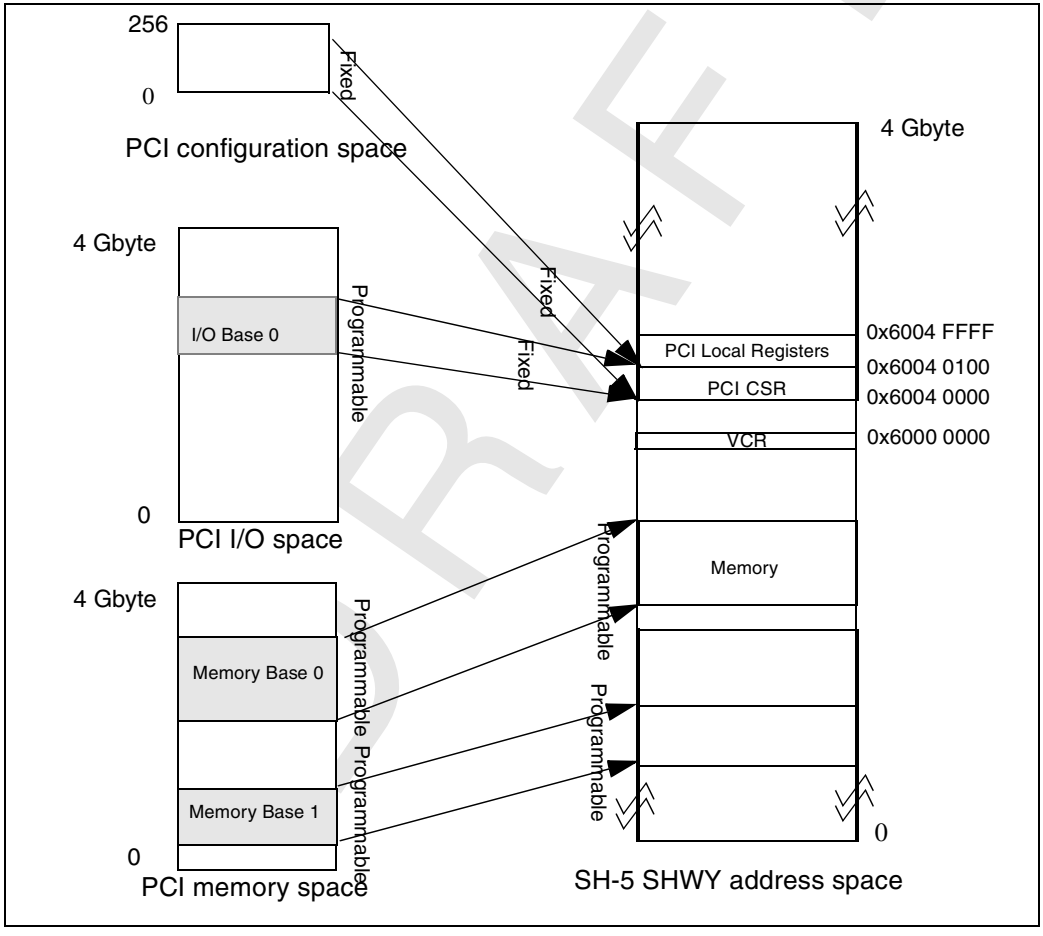

Figure 27: Mapping the 3 PCI target address spaces into the SH-5 address space

3.3.2 Accessing registers (SH-5 target)

Configuration registers

Configuration registers (like the CSR bank) can be accessed directly using configuration space reads and writes as specified in the PCI standard.

The SH-5 configuration registers may also be accessed using memory commands only if a memory region has been configured to map the CSR bank into SuperHyway address space.

Local registers

The SH-5 local registers may also be accessed using memory commands only if a memory region has been configured to map the local register bank into SuperHyway address space.

Local PCI bridge VCR

The VCR register may be accessed using PCI memory read or write commands. To do this one of the memory space regions should be programmed to include the VCR register.

A read can only be performed by pair of longword (LW) transfer with memory prefetching enabled (see the PFE field of the PCI.CR register *Section on page 176*). Attempting a burst will result in an error response on SuperHyway. The VCR should only be updated using a burst pair of two longword memory write commands. Burst more than two LW will result in an error response on SuperHyway.

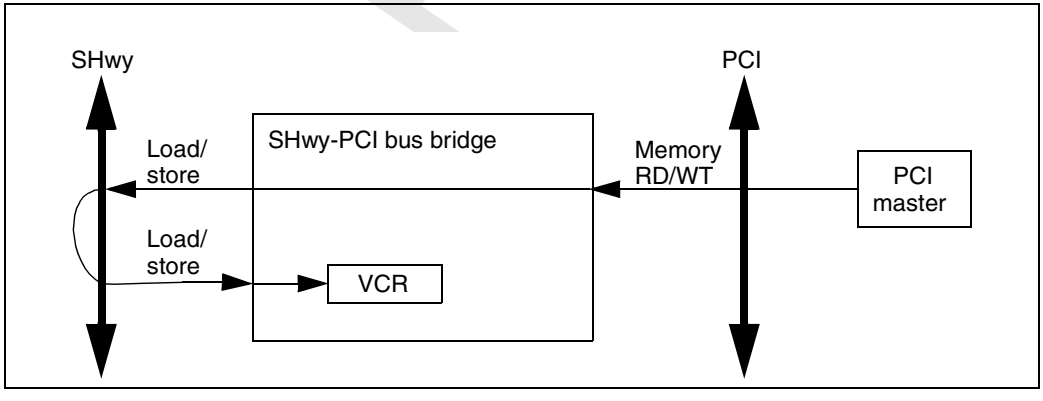

Figure 28: Local VCR access

Non PCI bridge register access

Accessing registers outside of the SH-5 PCI bridge from the PCI bus can be achieved using memory read/writes to a suitably mapped memory window and conforming to the access size guidelines shown in *Table 30 on page 126*.

Register size (bits)	PCI read access	PCI write access
64	Pair of non-burst 32-bit memory reads with pre-fetch enabled.	Pair of bursted 32-bit memory writes
32	Memory read with all byte enables set.	Memory write with all byte enables set.
16	Memory read with two byte enables set	Memory write with two byte enables set.
8	Memory read with one byte enable set.	Memory write with one byte enable set.

Table 30: Non-PCI bridge register access

3.3.3 Memory space (SH-5 target)

This section describes the access from external PCI master to PCI bridge of SH-5.

The translation between PCI memory space and SuperHyway space is governed by the following registers :

- PCI.MBAR[1:0] which determine the base addresses of two memory regions in PCI memory space (defined in *Section on page 158*). These conform to the PCI standard specification.
- PCI.LSR[1:0] which determine the size of the two memory regions and whether each is enabled or disabled (defined in *Section on page 181*).
- PCI.LAR[1:0] which determine the base address of the two memory regions in local SuperHyway address space (defined in *Section on page 183*).

The two memory regions are independent. Each translations occurs as illustrated by *Figure 29 on page 127*. The top bits of the address of an incoming PCI memory command are first compared to the top (at most 12) bits of *both* PCI.MBAR registers. The number of bits involved in the comparison is programmable (by PCI.LSR[1:0]) and depends on the size of the region. If one of the comparisons results in a match then the top bits of the PCI address are replaced by the top bits of the matching PCI.LAR[N] register. The remaining lower bits are copied from the PCI address to form the SuperHyway address.

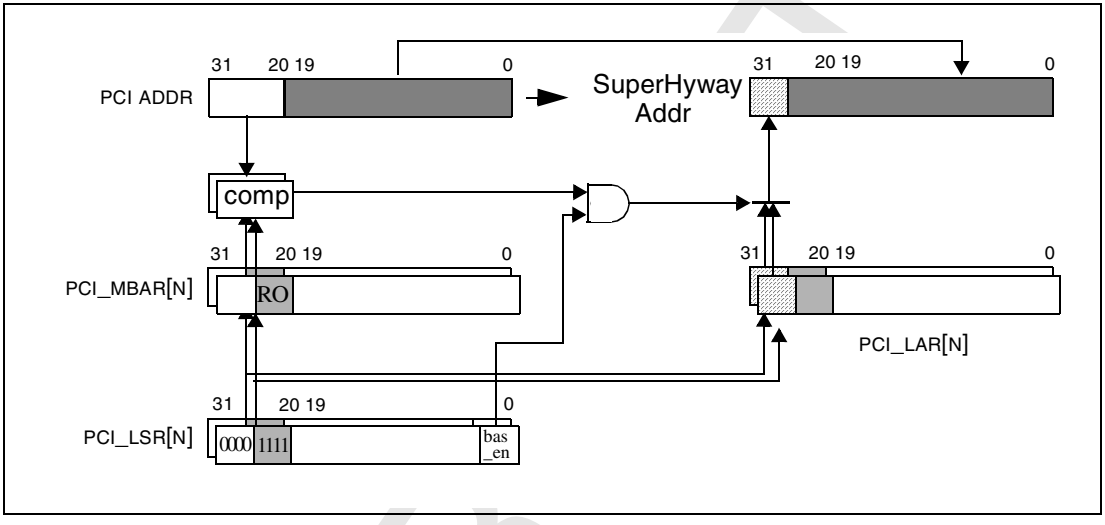

Figure 29: PCI to SuperHyway memory access

The translation allows for memory regions of size 1 M-512 M byte.

Single longword (LW) and burst transfers are supported to SuperHyway regions displaying simple “memory-like behaviour” (e.g. external memory areas) .

In general, access to registers is constrained because of possible side-effects and having to send a precise size of read or write to perform the access. Burst access to registers will, in general, give unexpected results unless it is performed in a

Data pre-fetching for memory read commands is supported.

When PCI single LW read occurs, 8 bytes are pre-fetched.

When PCI burst read occurs, 8 bytes, 32 bytes or blocks of 32 bytes are pre-fetched. This behavior is controlled by the PCI.CR.PFE and PCI.CR.PFCS fields of the PCI.CR register defined in *Section on page 176*

3.3.4 I/O space (SH-5 target)

PCI I/O address window is a fixed 256 bytes which is always mapped onto the SH-5's PCI CSR register bank. This is illustrated by *Figure 27 on page 124*.

The address of incoming PCI I/O commands is compared with the top bits in the PCI.IBAR register. If there is a match the address is translated into a local SuperHyway address as follows:

The lower 8 bits ([7:0]) are passed to SuperHyway unchanged.

The top 24 bits ([31:8] are replaced with 0x600400 when PCI address[31:8] and PCI I/O base address register[31:8] (PCI.IBAR) match.

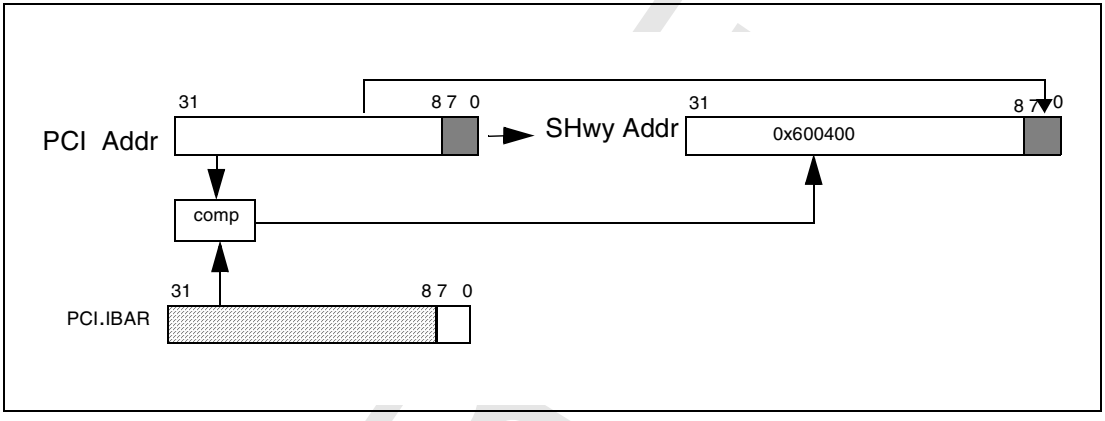

Figure 30: PCI to SuperHyway I/O access

3.3.5 Exclusive access (SH-5 target)

The PCI bus supports lock access. Once locked, the SH-5's PCI bridge is only accessible from the owner of lock. There is no exclusive access guarantee for any local SuperHyway resources which remain accessible by other SuperHyway modules during a PCI lock transfer.

3.3.6 Bi-endian support (SH-5 target)

The PCI bridge of SH-5 provides some support for big endian software and/or big-endian PCI address spaces. Since PCI is inherently little endian, both data switching and data non-switching are supported.

The configuration of the endian switching mode is decided by the PCI.TBS field of the PCI control register (PCI.CR defined in *Section on page 176*).

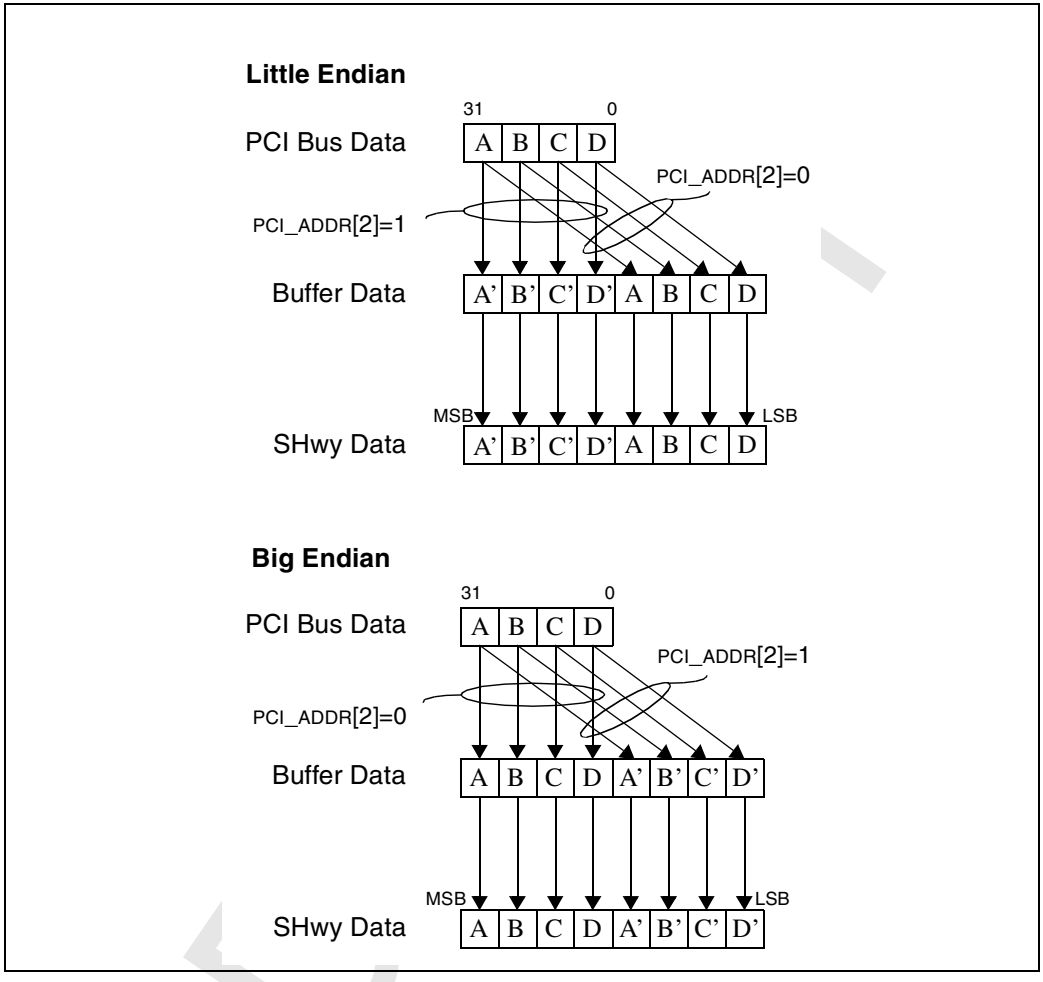

Figure 31: PCI to SuperHyway endian exchange (non-switching:PCI.TBS=0)

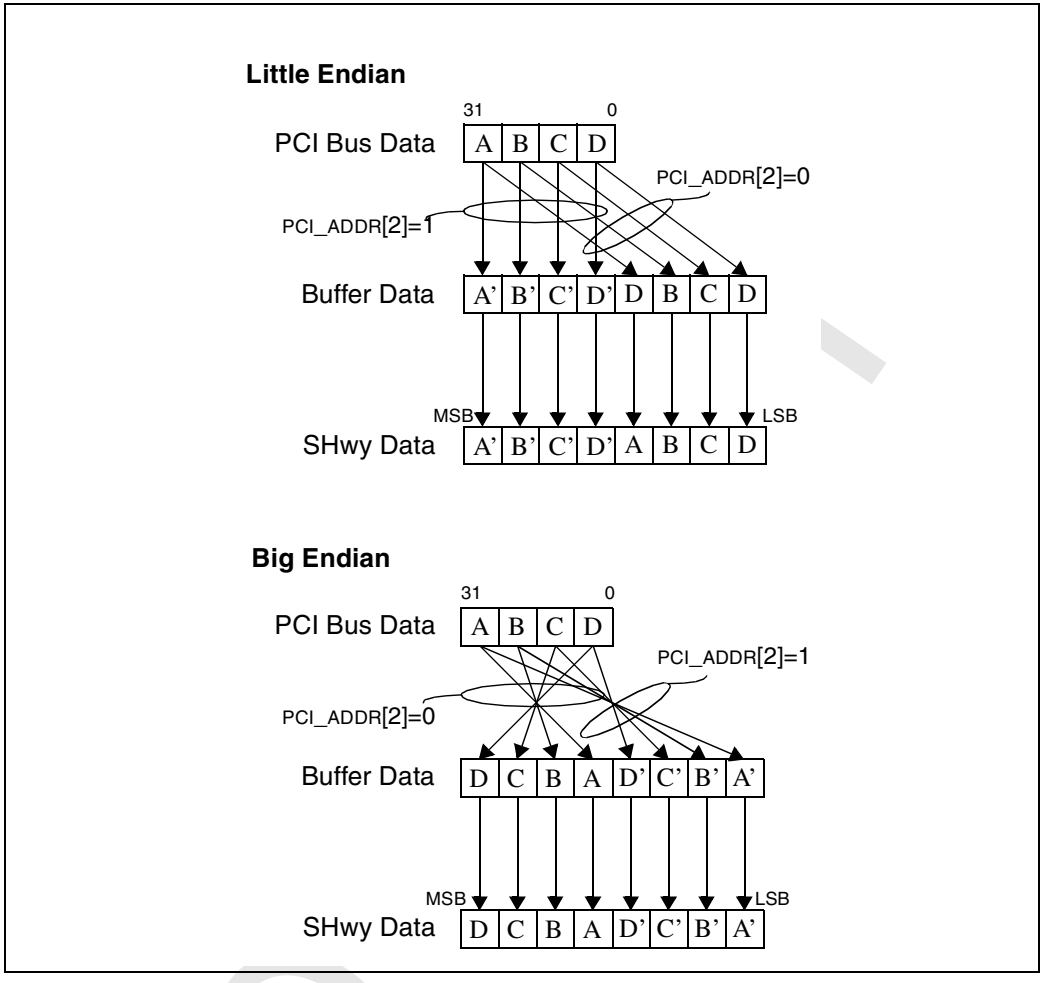

Figure 32: PCI to SuperHyway endian exchange (switching:PCI.TBS=1)

3.3.7 Cache coherency (SH-5 target)

SH-5's PCI bridge supports cache coherency. There is hardware support to enable incoming PCI commands to access the SuperHyway address space in a manner consistent with the SH-5 CPU caching the address space.

The PCI bridge warrants cache coherency to the access from the master device on PCI bus in the time that is actuated as the target device (host bus bridge mode and normal mode).

When accessing SuperHyway cacheable areas, set cache snoop registers, that is, PCI cache snoop control register (PCI.CSCR[N],n=0-1), PCI cache snoop address register (PCI.CSAR[n],n=0-1) to issue FLUSH or PURGE command on SuperHyway.

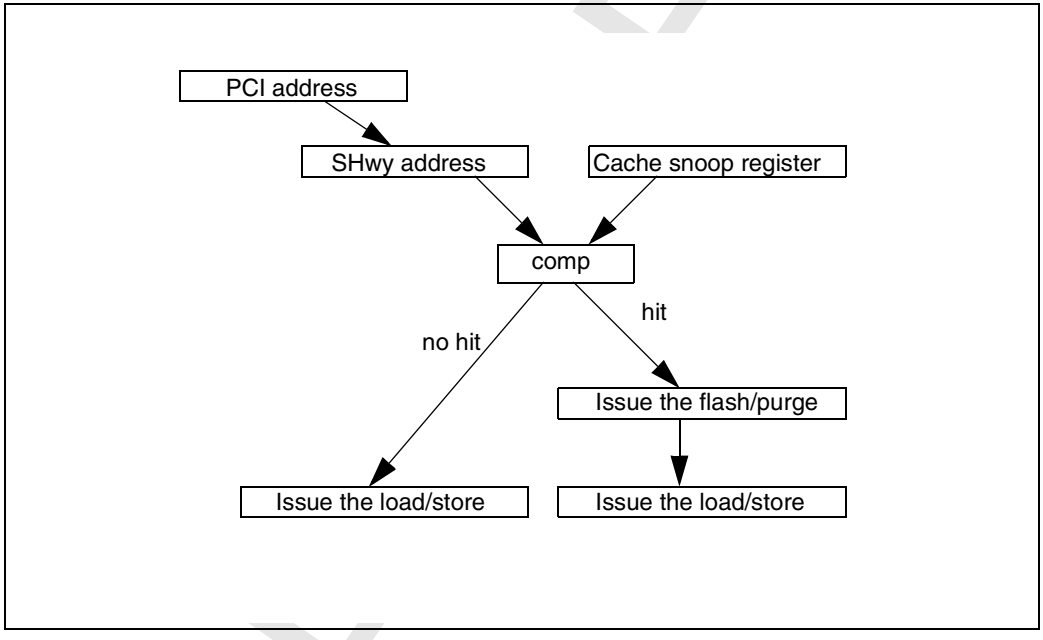

Figure 33: SuperHyway to PCI endian exchange

Note: This mechanism has been designed to support cache coherency on external memory areas (typically high speed external RAM or ROM). The use of this mechanism when asked to snoop other parts of the address space is not defined and should be treated as an error.

3.4 Accepting interrupts

The SH-5 PCI bridge can detect a large number of interrupts conditions which enable it to manage control of the PCI bus. A complete list of interrupts is given in *Table 31 on page 132*. Some of these are maskable locally within the PCI bridge all interrupts are maskable by the INTC (interrupt controller) module. For details see the PCI.INT register *Section on page 185* for general PCI interrupt conditions, the PCI.AINT register *Section on page 195* for interrupt conditions arising from PCI arbitration and the PCI.PINT register *Section on page 206* for those arising from power management.

Interrupt name	Default priority	Cause	PCI maskable[a]
INTA	High	INTA# asserted (host mode only)	-
INTB		INTB# asserted (host mode only)	-
INTC		INTC# asserted (host mode only)	-
INTD		INTD# asserted (host mode only)	-
SERR#		SERR# asserted (host mode only)	By PCI.INTM register
ERR		PCI error as recorded in PCI.INT and PCI.AINT register	By PCI.INTM, PCI.AINTM
PCI_PWRST_D3		Power state to D3	By PCI.PINTM register
PCI_PWRST_D2		Power state to D2	By PCI.PINTM register
PCI_PWRST_D1		Power state to D1	By PCI.PINTM register
PCI_PWRST_D0	Low	Power state to D0	By PCI.PINTM register

Table 31: PCI interrupts

a. Interrupts may also be masked by the interrupt controller (INTC) module.

3.5 Host mode functions

In host mode, the PCI bridge is responsible for configuring other devices on the PCI bus providing both arbitration and the fielding of certain types of interrupts. Configuration and arbitration is described as follows. Interrupts are described in *Section 3.4 on page 132*.

3.5.1 Configuration space access

The SH-5's PCI bridge supports configuration mechanism #1 as described in the PCI standard. The PCI PIO address register (PCI.PAR) and the PCI PIO data register (PCI.PDR), correspond to the configuration address register and the configuration data register respectively.

To perform a configuration space access, first write the address into the PCI.PAR register. A read from the PCI.PDR register will cause a configuration read cycle on the PCI bus and will return the result as the contents of the PCI.PDR register. A write to the PCI.PDR register will cause a configuration write cycle on the PCI bus.

There are 2 types of configuration transfers:

- type 1 is used to access PCI devices behind PCI-to-PCI bridge (bus number is not 0). For this transfer, PCI.PAR is AD[31:2]in the address phase of configuration access (bits [1:0]=2'b10).
- type 0 is used to access PCI devices connected to the same bus as host device (bus number is 0). For this transfer, bit [10:2]is passed on to PCI bus unchanged, but AD[31:11]is changed so that they are used as IDSEL signals.

Setting the device number

- Device number=0 makes AD16=1 and others=0
- Device number=1 makes AD17=1 and others=0
- Device number=16 makes AD31=1 and others=0

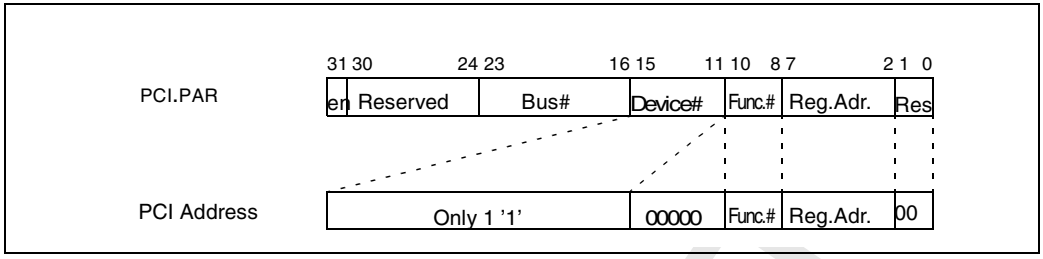

Figure 34: Address generate for type 0 configuration

In configuration access,

- PCI master abort (no device connected) will not cause interrupts.
- Configuration writes end normally.
- Configuration reads will return value of 0.
- Only SuperHyway **load** 8-byte and **store** 8-byte commands are supported.

3.5.2 PCI bus arbitration

In host bus bridge mode, the PCI bus arbiter inside PCI module is activated (PCI.CR:PCI_PBAM =0) this is the reset state.

The PCI bridge supports 4 external masters (i.e. 4 REQ# and GNT# pairs are provided). If the use of bus is requested simultaneously by more than one device, The bus is granted to a device with the highest priority.

The PCI bus arbiter supports 2 modes to determine the priority of devices,fixed priority and pseudo-round-robin. The mode is selected by the PCI.CR.BMABT register field.

Fixed priority (PCI.CR.BMABT = 0)

The priority of devices are fixed at following default value.

PCI module > device 0 > device 1 > device 2 > device 3

PCI module always gain use of the bus over other devices.

Pseudo-round-robin (PCI.CR.BMABT = 1)

The most recently granted device is assigned the lowest priority.

The initial priority is the same as fixed priority mode.

After device 1 has claimed and granted the bus and transferred data, the priority is as follows.

PCI module > device 0 > device 2 > device 3 > device 1

Then, after PCI module has claimed and granted the bus and transferred data, the priority is changed to :

device 0 > device 2 > device 3 > device 1 > PCI module

Then, after device 3 has claimed and granted the bus and transferred data, the priority is changed to :

device 0 > device 2 > device 1 > PCI module > device 3

In host bus bridge mode, bus parking is always done by PCI module.

In host mode (PCI_PBAM = 1'b1 in PCI.CR[12]), PCI bus arbiter is deactivated and PCI bus arbitration is done by an external PCI bus arbiter.

In normal mode, PCI bus arbiter is deactivated and PCI bus arbitration is done by an external PCI bus arbiter.

External arbitration (PCI.CR.PCI_PBAM =1)

In host mode arbitration may be delegated to an external agent by setting the PCI_PBAM field in the PCI.CR register. See *Section on page 176*.

3.6 Power management

The SH-5 PCI bridge supports PCI power management version 1.1. In particular supported features are:

- Power management control registers.
- Interrupts to INTC of SH-5.
- 4 power states : D0 - D3.
- PME# signal.

3.6.1 Host bus bridge mode

1 Power down

1.1 The sequence initiated by interrupt.

1.2 Before putting the PCI module to sleep, the PCI bus should be set to the appropriate power mode. Clearing the master enable bit in the command register in the configuration area will silence all masters.

2 Wake up

2.1 Sequence initiated by interrupt.

2.2 The PME# signals from PCI devices are connected to the INTC of SH-5 to detect wake up event.

3.6.1 Normal mode

1 Power down,

1.1 Sequence initiated by the PCI host device,

1.2 PCI clock stop is controlled by PCI host device,

1.3 Power state change (by register write) is detected and cause interrupts,

1.4 SH-5 put SuperHyway modules to appropriate power mode.

2 Wake up,

2.1 Sequence initiated by PCI host device,

2.2 Signal PME# and ask for wake up,

2.3 power state change is detected and cause interrupts,

2.4 SH-5 wake up all modules.

3.7 Register definitions

The PCI bridge registers are grouped into 3 banks:

1 The VCR (version control register)

2 PCI CSR (configuration space registers)

3 PCI local registers

PCI CSR registers conform to those specified in the PCI local bus specification 2.1. The VCR and PCI local registers are specific to the SH-5's implementation of the PCI Bridge.

The register definitions which follows use the base addresses and offsets as specified in *Table 32* below.

Register bank	Base address	Offset range	Directly accessible from:[a]
VCR	0x6000 0000 (PCI VCR BASE)	0x00 - 0x20	Local address space (sh)
CSR	0x6004 0000 (PCIBASE.SH)	0x000-0x0FF	Local address space (sh) PCI configuration space[b](pci)
Local	0x6004 0000 (PCIBASE.SH)	0x100-0x13B	Local address space (sh)

Table 32: PCI bridge register banks

a. Note that it is possible to map any or all registers into a PCI memory space window.

b. Not all registers in this bank are accessible from the PCI bus

A complete listing of all registers is given in *Section 3.8 on page 215*. The register address space is not fully populated as those addresses within the register address space for which no register is defined are reserved for future use and should not be accessed.

3.7.1 Version control register

This control register is specified in *Table 33*. The PERR_FLAGS field of this control register reports error conditions arising in both the PCI_CB (control block) and PCI_DB (data block) from SuperHyway transactions.

PCI.VCR[63:0]				PCI VCR BASE + 0x00	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
ERR_RCV	0	1	✓	An error response has been received	RW
	Operation		This bit is set by the PCI hardware if an error response is received by the PCI from the packet-router. It indicates that an earlier request from the PCI was invalid.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
ERR_SNT	1	1	✓	An error response has been sent	RW
	Operation		This bit is set by the PCI hardware if an error response is sent by the PCI bridge to the SuperHyway. It indicates that an earlier request to the PCI was invalid. This will occur if the PCI_CB or PCI_DB SuperHyway receive an unsupported request opcode.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
BAD_ADDR	2	1	✓	A request for an ‘undefined’ control register has been received	RW
	Operation		This bit is set by the PCI hardware if the PCI receives a **LoadWord** or **StoreWord** request for an ‘undefined’ control register in the PCI_CB memory map.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		

Table 33: PCI.VCR

PCI.VCR[63:0]				PCI VCR BASE + 0x00	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
UNSOL_RESP	3	1	✓	An unsolicited response has been received	RW
	Operation		This bit is set by the PCI hardware if an unsolicited response is received by the PCI from the packet-router.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
—	4	4	—	Reserved	RES
	Operation		Reserved		
	When read		Returns 0		
	When written		Ignored		
	HARD reset		0		
BAD_OPC	5	1	✓	A request with an unsupported opcode has been received	RW
	Operation		This bit is set by the PCI hardware if a request with an unsupported request opcode is received by the PCI from the packet-router.		
	When read		Returns current value		
	When written		Updates current value		
	HARD reset		0		
—	[7:6]	4	—	Reserved	RES
	Operation		Reserved		
	When read		Returns 0		
	When written		Ignored		
	HARD reset		0		

Table 33: PCI.VCR

PCI.VCR[63:0]				PCI VCR BASE + 0x00	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
MERR_FLAGS	[15:8]	8	—	module error flags (module specific)	RES
	Operation		Reserved		
	When read		Returns 0		
	When written		Ignored		
	HARD reset		0		
MOD_VERS	[31:16]	16	—	Module version	RO
	Operation		Used to indicate module version number		
	When read		Returns 0x0000		
	When written		Ignored		
	HARD reset		0x0000		
MOD_ID	[47:32]	16	—	Module identity	RO
	Operation		Used to identify module		
	When read		0x204F		
	When written		Ignored		
	HARD reset		0x204F		
BOT_MB	[55:48]	8	—	Bottom memory block	RO
	Operation		Used to identify bottom memory block		
	When read		Returns 0x20		
	When written		Ignored		
	HARD reset		0x20		

Table 33: PCI.VCR

PCI.VCR[63:0]				PCI VCR BASE + 0x00	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
TOP_MB	[63:56]	8	—	Top memory block	RO
	Operation		Used to identify top memory block		
	When read		0x01		
	When written		Ignored		
	HARD reset		0x01		

Table 33: PCI.VCR

PCI vendor ID

This field identifies the manufacturer of device. Valid vendor identifiers are allocated by PCI SIG to ensure uniqueness. Hitachi's vendor ID is 0x1054 (hardware fixed).

Pci.vid				PCIBASE.SH + 0x00, PCIBASE.PCI + 0x00	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_VID	[15:0]	16	-	PCI vendor ID	sh:RO pci:RO
	Operation		Hardware fixed		
	When read		0x1054		
	When written		-		
	HARD reset		0x1054		

PCI device ID

This field uniquely identifies this device amongst PCI devices manufactured by this vendor.

Pci.did				PCIBASE.SH + 0x02, PCIBASE.PCI + 0x02	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DID	[15:0]	16	-	PCI device ID	sh:RO pci:RO
	Operation		Hardware fixed		
	When read		0x350D		
	When written		-		
	HARD reset		0x350D		

PCI command

The PCI command register provides coarse control over a device's ability to generate and respond to PCI cycles. When 0 is written to this register, the device is logically disconnected from the PCI bus for all accesses except configuration accesses.

PCI.CMD				PCIBASE.SH + 0x04, PCIBASE.PCI + 0x04	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_IOS	0	1	-	PCI I/O space	sh:RW pci:RW
	Operation		Control a device's response to I/O space accesses.		
	When read		A value of 0 disables the device response. A value of 1 allows the device to respond to I/O space accesses.		
	When written		Update value		
	HARD reset		0		
PCI_MS	1	1	-	PCI memory space	sh:RW pci:RW
	Operation		Control a device's response to memory space accesses.		
	When read		A value of 0 disables the device response. A value of 1 allows the device to respond to memory space accesses.		
	When written		Update value		
	HARD reset		0		
PCI_BM	2	1	-	PCI bus master	sh:RW pci:RW
	Operation		Control a device's ability to act as a master on the PCI bus.		
	When read		A value of 0 disables the device from generating PCI accesses. A value of 1 allows the device to behave as a bus master.		
	When written		Update value		
	HARD reset		0		

PCI.CMD				PCIBASE.SH + 0x04, PCIBASE.PCI + 0x04	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_SC	3	1	-	PCI special cycles	sh:RO pci:RO
	Operation		Controls a device's action on special cycle operations.		
	When read		A value of 0 causes the device to ignore all special cycle operations. A value of 1 allows the device to monitor special cycle operation (not supported).		
	When written		-		
	HARD reset		0		
PCI_MWIE	4	1	-	PCI memory write and invalidate enable	sh:RO pci:RO
	Operation		This is an enable bit for using the memory write and invalidate command. PCI master doess not support this command.		
	When read		0		
	When written		-		
	HARD reset		0		
PCI_VGAPS	5	1	-	PCI VGA palette snoop	sh:RO pci:RO
	Operation		This bit controls how VGA compatible graphics device handle accesses to VGA palette register. This chip does not support VGA snoop.		
	When read		0		
	When written		-		
	HARD reset		0		

PCI.CMD				PCIBASE.SH + 0x04, PCIBASE.PCI + 0x04	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PER	6	1	-	PCI parity error response	sh:RW pci:RW
	Operation		This bit controls the device's response to parity errors. When the bit is set, the device must take its normal action when a parity error is detected. When the bit is 0, the device sets its detected parity error status bit (bit 15 in the status register) when an error is detected, but does not assert PCI_PERRN and continues normal operation.		
	When read		Read Value		
	When written		0: No response parity error 1: Response parity error		
	HARD reset		0		
PCI_STC	7	1	-	PCI stepping control	sh:RW pci:RW
	Operation		This bit used to control whether or not a device dose address/data stepping.		
	When read		Read value		
	When written		0: Address/data stepping is not permitted 1: Address/data stepping is permitted		
	HARD reset		1		
PCI_SERRE	8	1	-	PCI SERR enable	sh:RW pci:RW
	Operation		This bit is an enable bit for the pci_serrn driver. Address parity errors are reported only if this bit and bit 6 are 1.		
	When read		Read value		
	When written		0: PCI_SERRN output disable 1: PCI_SERRN output enable		
	HARD reset		0		

PCI.CMD				PCIBASE.SH + 0x04, PCIBASE.PCI + 0x04	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_FBBE	9	1	-	PCI fast back-to-back enable	sh:RO pci:RO
	Operation		This optional read/write bit controls whether or not master can do fast back-to-back transactions to different device. 0: Fast back-to-back transactions are only allowed to the same agent. 1: The master is allowed to generate fast back-to-back transactions to different agent (not supported).		
	When read		0		
	When written		-		
	HARD reset		0		
Reserved	[10,15]	6	-	Reserved	RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI status

This status register is used to record status information for PCI bus related events. The definition of each of the bits is given in the table below. A device may not need to implement all the bits, depending on device functionality. For instance, a device that acts as a target, but will never signal Target Abort, would not implement bit 11. Reserved bits should be read-only and return zero when read.

Reads to this register behave normally. Writes are slightly different in that bits can be reset, but not set. A one bit is reset whenever the register is written, and the write data in the corresponding bit location is a 1. For instance, to clear bit 14 and not affect any other bits, write the value 16'b0100_0000_0000_0000 to the register.

PCI.STATUS				PCIBASE.SH + 0x06, PCIBASE.PCI + 0x06	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
Reserved	[3:0],6	5	-	Reserved	sh:RO pci:RO
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		
PCI_CL	4	1	-	PCI capabilities list	sh:RO pci:RO
	Operation		This optional read-only bit indicates whether or not this device implements the pointer for a new capabilities linked list at offset 0x34.		
	When read		1		
	When written		-		
	HARD reset		1		

PCI.STATUS				PCIBASE.SH + 0x06, PCIBASE.PCI + 0x06	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_66C	5	1	-	PCI 66MHz capable	sh:RW pci:RO
	Operation		This bit indicates whether or not this device is capable of running at 66MHz.		
	When read		1: 66MHz capable 0: 33MHz capable		
	When written		Update value (sh)		
	HARD reset		0		
PCI_FBBC	7	1	-	PCI fast back-to-back capable	sh:RO pci:RO
	Operation		This optional read-only bit indicates whether or not the target is capable of accepting fast back-to-back transactions when the transactions are not to the same agent.		
	When read		1		
	When written		-		
	HARD reset		1		
PCI_MDPE	8	1	-	PCI master data parity error	sh:RWC pci:RWC
	Operation		This bit is used to bus masters only. It is set when three conditions are met: 1) The bus agent asserted PCI_PERR_N itself (on a read) or observed PCI_PERR_N asserted (on a write). 2) The agent setting the bit acted as the bus master for the operation in which the error occurred. 3) The parity error response bit (command register) is set.		
	When read		Value		
	When written		Clear this bit		
	HARD reset		0		

PCI.STATUS				PCIBASE.SH + 0x06, PCIBASE.PCI + 0x06	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DEVSEL	[10,9]	2	-	PCI DEVSEL timing	sh:RES pci:RES
	Operation		PCI_DEVSELN timing status 2b00: fast (not support) 2b01: medium 2b10: slow (not support) 2b11: reserved		
	When read		2b01		
	When written		-		
	HARD reset		2b01		
PCI_STA	11	1	-	PCI signaled target abort	sh:RWC pci:RWC
	Operation		This bit must be set by a target device whenever it terminates a transaction with target-abort.		
	When read		0: No target-abort termination 1: Target-abort used to terminate transaction.		
	When written		Clear this bit		
	HARD reset		0		
PCI_RTA	12	1	-	PCI received target abort	sh:RWC pci:RWC
	Operation		This bit must be set by a master device whenever its transaction is terminated with target-abort.		
	When read		0: No target-abort termination 1: Master device sets the target -abort to terminate a transaction		
	When written		Clear this bit		
	HARD reset		0		

PCI.STATUS				PCIBASE.SH + 0x06, PCIBASE.PCI + 0x06	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_RMA	13	1	-	PCI received master abort	sh:RWC pci:RWC
	Operation		This bit must be set by a master device whenever its transaction (expect for special cycle) is terminated with master-abort.		
	When read		0: No master-abort termination 1: Master-abort used to terminate transaction. This is not set in master-abort but in a special cycle.		
	When written		Clear this bit		
	HARD reset		0		
PCI_SSE	14	1	-	PCI signaled system error	sh:RWC pci:RWC
	Operation		This bit must be set whenever the device asserts PCI_SERRN.		
	When read		0: Device does not assert PCI_SERRN. 1: Device asserts PCI_SERRN.		
	When written		Clear this bit		
	HARD reset		0		
PCI_DPE	15	1	-	PCI detected parity error	sh:RWC pci:RWC
	Operation		This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled (as controlled by bit 6 in the command register).		
	When read		0: Device is not detecting parity error. 1: Device is detecting parity error.		
	When written		Clear this bit		
	HARD reset		0		

PCI revision ID

This register specifies a device specific revision identifier.

PCI.RID				PCIBASE.SH + 0x08, PCIBASE.PCI + 0x08	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_RID	[7:0]	8	-	PCI revision ID	sh:RO pci:RO
	Operation		Hardware fixed		
	When read		TBD		
	When written		-		
	HARD reset		TBD		

PCI program interface

This field is the programming interface for the IDE controller class code.

PCI.PIF				PCIBASE.SH + 0x09, PCIBASE.PCI + 0x09	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MIDED	7	1	-	PCI master IDE device	sh:RW pci:RO
	Operation		1: Master IDE Device 0: Slave IDE Device This register is not cleared by HARD reset.		
	When read		Value		
	When written		Update value (sh) if PCI.CR.PCI_CFINT == 0 else value not updated		
	HARD reset		Undefined		

PCI.PIF				PCIBASE.SH + 0x09, PCIBASE.PCI + 0x09	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PIS	3	1	-	PCI programmable indicator (secondary)	sh:RW pci:RO
	Operation		This register is not cleared by HARD reset.		
	When read		Value		
	When written		Update value (sh). if PCI.CR.PCI_CFINT = 0 else value not updated		
	HARD reset		Undefined		
PCI_OMS	2	1	-	PCI operating mode (secondary)	sh:RW pci:RO
	Operation		This register is not cleared by HARD reset.		
	When read		Value		
	When written		Update value (sh). if PCI.CR.PCI_CFINT = 0 else value not updated		
	HARD reset		Undefined		
PCI_PIP	1	1	-	PCI programmable indicator (primary)	sh:RW pci:RO
	Operation		This register is not cleared by HARD reset.		
	When read		Value		
	When written		Update value (sh). if PCI.CR.PCI_CFINT = 0 else value not updated		
	HARD reset		Undefined		
pci_omp	0	1	-	PCI operating mode (secondary)	sh:RW pci:RO
	Operation		This register is not cleared by HARD reset.		
	When read		Value		
	When written		Update value (sh). if PCI.CR.PCI_CFINT = 0 else value not updated		
	HARD reset		Undefined		

PCI.PIF				PCIBASE.SH + 0x09, PCIBASE.PCI + 0x09	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
Reserved	[6:4]	3	-	Reserved	sh:RES pci:RES
	Operation		Reserved		
	When read		3b000		
	When written		-		
	HARD reset		0		

PCI sub class code

PCI.SUB				PCIBASE.SH + 0x0a, PCIBASE.PCI + 0x0a	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_SUB	[7:0]	8	-	PCI sub class code	sh:RW pci:RO
	Operation		This register is not cleared by HARD reset.		
	When read		value		
	When written		Update value (sh). if PCI.CR.PCI_CFINT == 0 else value not updated		
	HARD reset		Undefined		

PCI base class code

PCI.BCC				PCIBASE.SH + 0x0b, PCIBASE.PCI + 0x0b	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_BCC	[7:0]	8	-	PCI base class code	sh:RW pci:RO
	Operation		This register is not cleared by HARD reset.		
	When read		Value		
	When written		Update value (sh). if PCI.CR.PCI_CFINT == 0 else value not updated		
	HARD reset		Undefined		

PCI cache line size

PCI.CLS				PCIBASE.SH + 0x0c, PCIBASE.PCI + 0x0c	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_CLS	[7:0]	8	-	PCI cache line size	sh:RO pci:RO
	Operation		HARD fixed		
	When read		0x00		
	When written		-		
	HARD reset		0x00		

PCI latency timer

This register specifies, in units of PCI bus clocks, the value of latency timer for this PCI bus master.

PCI.MLT				PCIBASE.SH + 0x0d, PCIBASE.PCI + 0x0d	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MLT	[7:0]	8	-	PCI latency timer	sh:RW pci:RW
	Operation		Binding in the biggest acquisition time of PCI bus		
	When read		value		
	When written		Update value		
	HARD reset		0x00		

PCI header type

This byte identifies the layout of the second part of the predefined header (beginning at byte 10h in configuration space) and also whether or not the device contains multiple functions.

PCI.HDR				PCIBASE.SH + 0x0e, PCIBASE.PCI + 0x0e	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MFE	7	1	-	PCI multiple function enable	sh:RO pci:RO
	Operation		Hard fixed (no multiple function support)		
	When read		0		
	When written		-		
	HARD reset		0		

PCI.HDR				PCIBASE.SH + 0x0e, PCIBASE.PCI + 0x0e	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_HDR	[6:0]	7	-	PCI header type	sh:RO pci:RO
	Operation		Hard fixed (no multiple function support)		
	When read		7b000 0000		
	When written		-		
	HARD reset		7b000 0000		

PCI BIST

BIST function is not supported.

PCI.BIST				PCIBASE.SH + 0x0f, PCIBASE.PCI + 0x0f	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_BISTC	7	1	-	PCI BIST capable	sh:RO pci:RO
	Operation		0: Incapable 1: Capable		
	When read		0		
	When written		-		
	HARD reset		0		
RESERVED	[6,0]	7	-	Reserved	sh:RES pci:RES
	Operation		Reserved		
	When read		2b00		
	When written		-		
	HARD reset		2b00		

PCI I/O base address register

This register packages the I/O space base address register of the PCI configuration register that is prescribed with PCI local specification.

Refer to *Section 3.2.9: Accessing PCI I/O space (SH-5 master) on page 119*.

PCI.IBAR				PCIBASE.SH + 0x10, PCIBASE.PCI 0x10	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_IOB1	[31:8]	24	-	PCI I/O base address (upper 24 bits)	sh:RW pci:RW
	Operation		PCI bridge internal register address upper 12-bit of PCI bus		
	When read		Returns current value		
	When written		Update value		
	HARD reset		0x000000		
PCI_IOB2	[7:4]	4	-	PCI I/O base address (lower 4 bits)	sh:RO pci:RO
	Operation		Hard fixed		
	When read		Returns 0x0		
	When written		-		
	HARD reset		0x0		
RESERVED	[3:1]	3	-	Reserved	sh:RES pci:RES
	Operation		Reserved		
	When read		3b000		
	When written		-		
	HARD reset		3b000		

PCI.IBAR				PCIBASE.SH + 0x10, PCIBASE.PCI 0x10	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_IOSI	0	1	-	PCI I/O space indicator	sh:RO pci:RO
	Operation		Hard fixed		
	When read		1		
	When written		-		
	HARD reset		1		

PCI memory base address register0

This register packages the memory space base address register of the PCI configuration register that is prescribed with PCI local specification.

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*

Pci.mbar0				PCIBASE.SH + 0x14, PCIBASE.PCI + 0x14	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MBA1	[31:20]	12	-	PCI memory base address (upper 12-bits)	sh:RW pci:RW
	Operation		Base address of access from PCI bus to SH-5's modules. This register is cleared by software reset or hard reset.		
	When read		Value		
	When written		Update value **PCI.LSE0[28:20]** **address space** **effective bit of PCI_MBA1** 9b0 0000 0000 1 Mbyte [31,20] 9b0 0000 0001 2 Mbyte [31,21] | | | 9b1 1111 1111 512 Mbyte [31,29]	
	HARD reset		12h000		

Pci.mbar0				PCIBASE.SH + 0x14, PCIBASE.PCI + 0x14	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MBA2	[19:4]	16	-	PCI memory base address (under 16 bits)	sh:RO pci:RO
	Operation		Hard fixed		
	When read		16b0000 0000 0000 0000		
	When written		-		
	HARD reset		18b0000 0000 0000 0000		
PCI_LAP	3	1	-	PCI local address prefetchable	sh:RO pci:RO
	Operation		Prefetchable support Controlled by cache snoop control registers		
	When read		1		
	When written		-		
	HARD reset		1		
PCI_LAT	[2,1]	2	-	PCI local address type	sh:RO pci:RO
	Operation		Only supported on a 32-bit address space		
	When read		2b00		
	When written		-		
	HARD reset		2b00		
PCI_MSI	0	1	-	PCI memory space indicator	sh:RO pci:RO
	Operation		Hard fixed		
	When read		0		
	When written		-		
	HARD reset		0		

PCI memory base address register1

This register packages the memory space base address register of the PCI configuration register that is prescribed with PCI local specification.

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

PCI.MBAR1				PCIBASE.SH + 0x18, PCIBASE.PCI + 0x18	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MBA1	[31:20]	12	-	PCI memory base address (upper 12 bits)	sh:RW pci:RW
	Operation		Base address of the time of access from PCI bus to SH-5's modules. This register is cleared by software reset or hard reset.		
	When read		Value		
	When written		Update value **PCI.LSE1[28:20]** **address space** **effective bit of PCI_MBA1** 9b0 0000 0000 1 Mbyte [31,20] 9b0 0000 0001 2 Mbyte [31,21] | | | 9b1 1111 1111 512 Mbyte [31,29]		
	HARD reset		12h000		
PCI_MBA2	[19:4]	16	-	PCI memory base address (lower 16 bits)	sh:RO pci:RO
	Operation		Hard fixed		
	When read		16b0000 0000 0000 0000		
	When written		-		
	HARD reset		18b0000 0000 0000 0000		

PCI.MBAR1				PCIBASE.SH + 0x18, PCIBASE.PCI + 0x18	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_LAP	3	1	-	PCI local address prefetchable	sh:RO pci:RO
	Operation		Prefetchable support Controlled by cache snoop control registers		
	When read		1		
	When written		-		
	HARD reset		1		
PCI_LAT	[2:1]	2	-	PCI local address type	sh:RO pci:RO
	Operation		Only support 32-bit address space		
	When read		2b00		
	When written		-		
	HARD reset		2b00		
PCI_MSI	0	1	-	PCI memory space indicator	sh:RO pci:RO
	Operation		Hard fixed		
	When read		0		
	When written		-		
	HARD reset		0		

PCI subsystem vendor ID

Refer to miscellaneous registers section of PCI local specification Rev 2.2.

PCI.SVID				PCIBASE.SH + 0x2c, PCIBASE.PCI + 0x2c	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_SVID	[15:0]	16	-	PCI subsystem vendor ID	sh:RW[a] pci:RO
	Operation		-		
	When read		Value		
	When written		Update value (sh only)		
	HARD reset		Undefined		

a. PCI.CR.PCI_CFINT=0: R/W PCI.CR.PCI_CFINT=1: RO

PCI subsystem ID

Refer to section about miscellaneous registers of PCI local specification Rev 2.2.

PCI.SID				PCIBASE.SH + 0x2e, PCIBASE.PCI + 0x2e	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_SVID	[15:0]	16	-	PCI subsystem ID	sh:RW[a] pci:RO
	Operation		-		
	When read		Value		
	When written		Update value (sh only)		
	HARD reset		Undefined		

a. PCI.CR.PCI_CFINT=0: R/W PCI.CR.PCI_CFINT=1: RO

PCI capabilities pointer

This register is the expansion function pointer register of the PCI configuration register that is prescribed in the PCI power management specification.

PCI.CP				PCIBASE.SH + 0x34, PCIBASE.PCI + 0x34	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_CP	[7:0]	8	-	PCI capabilities pointer	sh:RO pci:RO
	Operation		The offset address of the expansion function register		
	When read		8h40		
	When written		-		
	HARD reset		8h40		

PCI interrupt line

Set priority for INTn, select by PCI interrupt pin register.

Pci.intline				PCIBASE.SH + 0x3c, PCIBASE.PCI + 0x3c	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_INTLINE	[7:0]	8	-	PCI interrupt line	sh:RW pci:RW
	Operation		PCI interrupt connected to the external interrupt of SH-5 8h00: irl0 8h01: irl1 | | 8h0F: irl15 8h10 to 8hFE is reserved 8FF: no connection to the INTC		
	When read		Value		
	When written		update value		
	HARD reset		0x00		

PCI interrupt pin

This register identifies which interrupt pin the device uses.

PCI.INTPIN				PCIBASE.SH + 0x3d, PCIBASE.PCI + 0x3d	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_INTPIN	[7:0]	8	-	PCI interrupt pin	sh:RW pci:RO
	Operation		Selects the interrupt pin 8h00: dont use interrupt pin 8h01: pci_intan 8h02: pci_intbn 8h03: pci_intcn 8h04: pci_intdn 8h05 to 8'hFF: reserved		
	When read		Value		
	When written		Update value (sh). PCI.CR.PCI_CFINT == 0		
	HARD reset		8h01		

PCI minimum grant

This register is not programmable.

Pci.mingnt				PCIBASE.SH + 0x3e, PCIBASE.PCI + 0x3e	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MINGNT	[7:0]	8	-	PCI minimum grant	sh:RO pci:RO
	Operation		Hard fixed		
	When read		8h00		
	When written		-		
	HARD reset		8h00		

PCI maximum latency

This register is not programmable.

PCI.MAXLAT				PCIBASE.SH + 0x3f, PCIBASE.PCI + 0x3f	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MAXLAT	[7:0]	8	-	PCI maximum latency	sh:RO pci:RO
	Operation		Hard fixed		
	When read		0x00		
	When written		-		
	HARD reset		0x00		

PCI capability identifier

When 01H is read by system software it indicates that the data structure currently being pointed to is the PCI power management data structure. Each function of a PCI device may have only one item in its capability list with PCI_CID set to 01H.

PCI.CID				PCIBASE.SH + 0x40, PCIBASE.PCI + 0x40	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_CID	[7:0]	8	-	PCI capability identifier	sh:RO pci:RO
	Operation		Hard Fixed		
	When read		8h01		
	When written		-		
	HARD reset		8h01		

PCI next item pointer

The next item pointer register gives the location of the next item in the function's capability list. The value given is an offset into the function's PCI configuration space. If the function does not implement any other capabilities defined by the PCI SIG for inclusion in the capabilities list, or if power management is the last item in the list, then this register must be set to 00h.

PCI.NIP				PCIBASE.SH + 0x41, PCIBASE.PCI + 0x41	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_NIP	[7:0]	8	-	PCI next item pointer	sh:RO pci:RO
	Operation		Hard fixed		
	When read		8h00		
	When written		-		
	HARD reset		8h00		

PCI power management capability

This is a 16-bit read-only register which provides information on the capabilities of the power management related functions. The information in this register is generally static and known at design time. This register is not cleared by hard reset. This register must set when PCI.CR.PCI_CFINT is zero.

PCI.PMC				PCIBASE.SH + 0x42, PCIBASE.PCI + 0x42	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMES	[15:11]	5	-	PCI PME support	sh:RW pci:RO
	Operation		PME_SUPPORT - This 5-bit field indicates the power states in which the function may assert PME#. A value of 0b for any bit indicates that the function is not capable of asserting the PME# signal while in that power state. Bit(11) XXXX1b - PME# can be asserted from D0 Bit(12) XXX1Xb - PME# can be asserted from D1 Bit(13) XX1XXb - PME# can be asserted from D2 Bit(14) X1XXXb - PME# can be asserted from D3 hot Bit(15) 1XXXXb - PME# can be asserted from D3 cold		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		Undefined		
PCI_D2S	10	1	-	PCI D2 support	sh:RW pci:RO
	Operation		1: This function supports the D2 power management state 0: This functions does not support D2		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		Undefined		

PCI.PMC				PCIBASE.SH + 0x42, PCIBASE.PCI + 0x42	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_D1S	9	1	-	PCI D1 support	sh:RW pci:RO
	Operation		1: This function supports the D1 power management state 0: This functions does not support D1		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		Undefined		
PCI_DSI	5	1	-	PCI device specific initialization	sh:RO pci:RO
	Operation		Hard fixed		
	When read		1b0		
	When written		-		
	HARD reset		1b0		
PCI_PMEC	3	1	-	PCI PME clock	sh:RW pci:RO
	Operation		1: The function relies on the presence of the PCI clock for PME# operation. 0: No PCI clock is required for the function to generate PME#. Functions that do not support PME# generation in any state must return “0” for this field.		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		1b0		

PCI.PMC				PCIBASE.SH + 0x42, PCIBASE.PCI + 0x42	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMV	[2:0]	3	-	PCI power management version	sh:RW pci:RO
	Operation		A value of 3b010 indicates that this function complies with revision 1.1 of the PCI power management interface specification.		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		3b010		
RESERVED	[8:6],4	4	-	Reserved	sh:RES pci:RES
	Operation		Hard fixed		
	When read		4b0		
	When written		-		
	HARD reset		4b0		

PCI power management control/status

This 16-bit register is used to manage the PCI function's power management state as well as to enable/monitor PME.

The PME support bits, PME_STATUS and PME_EN, are defined to be sticky bits for functions that can generate PMEs from D3cold therefore their states are not affected by power on reset or transitions from D3cold to the D0 uninitialized state. Preservation of these bits is typically achieved by either powering them with an auxiliary power source or by using non-volatile storage cells for them. The only way to clear out these bits is to have system software write to them with the appropriate values.

As mentioned previously, the PME context is defined as the logic responsible for identifying PME, for generating the pci_pmen signal and the bits within this register that provide the standard system interface for this functionality. PME context also contains any device class specific status that must survive the transition to the D0 uninitialized state as well.

If a function supports PCI_PMEN generation from D3cold, its PME Context is not affected by either a PCI bus segment reset (hardware component reset) or the internal "soft" re-initialization that occurs when restoring a function from D3hot. This is because the function's PME functionality itself may have been responsible for the wake event which caused the transition back to D0. Therefore, the PME context must be preserved for the system software to process.

If pci_pmen generation is not supported from D3cold, then all PME context is initialized with the assertion of a bus segment reset.

Due to PCI bus PCI_RSTN assertion not necessarily clearing all PME context (functions that support pci_pmen from D3cold), the system software is required to explicitly initialize all PME context, including the PME support bits, for all functions during initial operating system load. In terms of the PMCSR, this means that during the initial operating system load each function's PME_EN bit must be written with a "0", and each function's PME_STATUS bit must be written with a "1" by system software as part of the process of initializing the system.

PCI.PMCSR				PCIBASE.SH + 0x44, PCIBASE.PCI +0x44	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMES	15	1	-	PCI PME status	sh:RWC pci:RWC
	Operation		This bit is set when the function asserts the PCI_PMEN signal independent of the state of the PCI.PMES.PCI_PMEEN bit. Writing a 1 to this bit will clear it and cause the function to stop asserting a PCI_PMEN (if enabled). Writing a 0 has no effect. This bit defaults to 0 if the function does not support PCI_PMEN generation from D3 cold. If the function supports PCI_PMEN from D3 cold, then this bit is sticky and must be explicitly cleared by the operating system each time the operating system is initially loaded.		
	When read		Value		
	When written		1b1: Clear this register 1b0: No effect this register		
	HARD reset		1bx		

PCI.PMCSR				PCIBASE.SH + 0x44, PCIBASE.PCI +0x44	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DSC	[14:13]	2	-	PCI data scale	sh:RW pci:RO
	Operation		2b00: unknown 2b01: 0.1 * PCI.PCDD (watts) 2b10: 0.01 * PCI.PCDD (watts) 2b11: 0.001 * PCI.PCDD (watts)		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		2bxx		
PCI_DSL	[12:9]	4	-	PCI data select	sh:RW pci:RW
	Operation		4b0000: D0 power consumed 4b0001: D1 power consumed 4b0010: D2 power consumed 4b0011: D3 power consumed 4b0100: D0 power dissipated 4b0101: D1 power dissipated 4b0110: D2 power dissipated 4b0111: D3 power dissipated 4b1000: common logic power consumption 4b1001 to 4b1111 is reserved		
	When read		Value		
	When written		Update value		
	HARD reset		4b0000		

PCI.PMCSR				PCIBASE.SH + 0x44, PCIBASE.PCI +0x44	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMEEN	8	1	-	PCI PME enable	sh:RW pci:RW
	Operation		A “1” enables the function to assert PCI_PMEN. When “0”, PCI_PMEN assertion is disabled. This bit defaults to “0” if the function does not support PME# generation from D3 cold. If the function supports PME# from D3 cold, then this bit is sticky and must be explicitly cleared by the operating system each time it is initially loaded. Functions that do not support PCI_PMEN generation from any D-state, may hardwire this bit to be read-only always returning a “0” when read by system software.		
	When read		Value		
	When written		Update value		
	HARD reset		1bx		
PCI_PS	[1,0]	2	-	PCI power state	sh:RW pci:RW
	Operation		This 2-bit field is used both to determine the current power state of a function and to set the function into a new power state. The definition of the field values is given below. 2b00: D0 2b01: D1 2b10: D2 2b11: D3 hot If software attempts to write an unsupported, optional state to this field, the write operation must complete normally on the bus; however, the data is discarded and no state change occurs.		
	When read		Value		
	When written		Update value		
	HARD reset		2b00		

PCI.PMCSR				PCIBASE.SH + 0x44, PCIBASE.PCI +0x44	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[7,2]	6	-	Reserved	sh:RES pci:RES
	Operation		Hard fixed		
	When read		6b00 0000		
	When written		-		
	HARD reset		6b00 0000		

PCI PMCSR bridge support extension

This register supports PCI bridge specific functionality and is required for all PCI-to-PCI bridges.

PCI.PMCSR_BSE				PCIBASE.SH + 0x46, PCIBASE.PCI + 0x46	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_BPCCEN	7	1	-	PCI bus power/clock control enable	sh:RW pci:RO
	Operation		When the bus power/clock control mechanism is disabled, the bridge's PMCSR power state field cannot be used by the system software to control the power or clock of the bridge's secondary bus.		
	When read		value		
	When written		Update value (sh)		
	HARD reset		1bx		

PCI.PMCSR_BSE				PCIBASE.SH + 0x46, PCIBASE.PCI + 0x46	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_B2B3N	6	1	-	PCI B2/B3 support for D3 hot	sh:RW pci:RO
	Operation		The state of this bit determines the action that is to occur as a direct result of programming the function to D3 hot. A “1” indicates that when the bridge function is programmed to D3 hot, its secondary bus’s PCI clock will be stopped (B2). A “0” indicates that when the bridge function is programmed to D3 hot, its secondary bus will have its power removed (B3). This bit is only meaningful if bit 7 (PCI_BPCCEN) is a “1”.		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		1bx		
RESERVED	[5:0]	6	-	Reserved	sh:RES pci:RES
	Operation		Hard fixed		
	When read		6b0 0000		
	When written		-		
	HARD reset		6b00 0000		

PCI power consumption/dissipation data

The data register is an optional, 8-bit read-only register that provides a mechanism to report state dependent operating data such as power consumed or heat dissipation. Typically the data returned through the data register is a static copy (look up table, for example) of the function’s worst case “DC characteristics” data sheet. This data, when made available to system software, could then be used to intelligently make decisions about power budgeting and cooling requirements for example. Any type of data could be reported through this register, but only power usage is defined by this version of the specification.

If the data register is implemented, then the PCI.PMCSR.PCI_DSL and PCI.PMCSR.PCI_DSC fields must also be implemented. If this register is not implemented, a value of 0 should always be returned by this register as well as for the PCI.PMCSR.PCI_DSL and PCI.PMCSR.PCI_DSC fields.

Software may check for the presence of the data register by writing different values into the PCI.PMCSR.PCI_DSL field, looking for non-zero return data in the data register and/or PCI.PMCSR.PCI_DSC field.

Any non-zero data register/PCI.PMCSR.PCI_DSL read data indicates that the Data register complex has been implemented. Since PCI.PMCSR.PCI_DSL is a 4-bit field, an exhaustive presence detection scan requires 16 write/read operations to the PCI.PMCSR.PCI_DSL field and data register/PCI.PMCSR.PCI_DSC field respectively.

PCI.PCDD				PCIBASE.SH + 0x47, PCIBASE.PCI + 0x47	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PCDD	[7:0]	8	-	PCI power consumption/dissipation data	sh:RW pci:RO
	Operation		This register is used to report the state dependent data requested by the PCI.PMCSR.PCI_DSL field. The value of this register is scaled by the value reported by the PCI.PMCSR.PCI_DSC field.		
	When read		Value		
	When written		Update value		
	HARD reset		0x00		

3.7.2 Local registers

PCI control register

PCI.CR is a 32-bit register which controls the major operation modes of the PCI bridge. The register is write protected; only writes in which the top 8 bits (that is, bits 32:24]) have the value 0xA5 are performed. All other writes are ignored.

PCI_CR[31:0]				PCIBASE.SH + 0x100	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DLLDLAY	[23:15]	9	-	PCI internal DLL clock delay	sh:RW
	Operation				
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		9b001100110		
PCI_DLLLOCK	14	1	-	PCI DLL Lock	sh:RO
	Operation		1b0: DLL is not Lock 1b1: DLL is Lock		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		

PCI_CR[31:0]				PCIBASE.SH + 0x100	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PBAM	12	1	-	PCI bus arbiter mode setting	sh:RW
	Operation		1b0: Internal bus arbiter is active REQ[3:0] is input from other bus master GNT[3:0] is output to other bus master 1b1: Internal bus arbiter is inactive (external bus arbiter mode) REQ[0] is output to external bus arbiter GNT[0] is input from external bus arbiter This bit is only effective in host mode		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
PCI_PFCS	11	1	-	PCI pre-fetch command setting	sh:RW
	Operation		1b0: Always 8 byte pre-fetching 1b1: Always 32 byte pre-fetching		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
PCI_FTO	10	1	-	PCI TRDY and IRDY control enable	sh:RW
	Operation		1b0: disable 1b1: enable		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		

PCI_CR[31:0]				PCIBASE.SH + 0x100	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PFE	9	1	-	PCI pre-fetch enable	sh:RW
	Operation		1b0: Disable 1b1: Enable		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
PCI_TBS	8	1	-	PCI target byte swap	sh:RW
	Operation		1b0: No swap 1b1: Byte is swapped		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
PCI_SPUE	7	1	-	PCI signal pull up disable	sh:RW
	Operation		1b0: Pull up 1b1: No pull up		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
PCI_BMAM	6	1	-	PCI bus master arbitration mode control	sh:RW
	Operation		1b0: Fixed mode (device0 > device1 > device2 > device3 > device4) 1b1: Round robin		
	When read		Value		
	When written		Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		

PCI_CR[31:0]				PCIBASE.SH + 0x100	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_HOSTNS	5	1	-	PCI HOSTEN Status	sh:RO
	Operation		Hardwired to HOSTEN		
	When read		HOSTEN status		
	When written		-		
	HARD reset		HOSTEN status		
PCI_CLKENS	4	1	-	PCI PCICKSEL status	sh:RO
	Operation		Hardwired to PCICKSEL		
	When read		PCICKSEL status		
	When written		-		
	HARD reset		PCICKSEL status		
PCI_SOCS	3	1	-	PCI PCI_SERR_N output control by software	sh:RW
	Operation		Only in the case of PCI.CMD.PCI_SERR == 1b1 effective		
	When read		1b0		
	When written		1b1: PCI_SERR_N is assert one PCI_CLK cycle (normal mode) This bit user normal mode only if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
PCI_IOCS	2	1	-	PCI PCI_INTA_A output control by software	sh:RW
	Operation		This bit use normal mode only		
	When read		Value		
	When written		1b1: PCI_INTA_N low level output 1b0: PCI_INTA_N high level output Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		

PCI_CR[31:0]				PCIBASE.SH + 0x100	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_CFINT	0	1	-	PCI internal register initialize control	sh:RW
	Operation		Case given this bit is 1b0: Host case: bus right is not given to the device other than the PCI bus Normal case: PCI bridge return RETRY receiving the access from the PCI bus		
	When read		Value		
	When written		1b1: PCI bridge is active 1b0: PCI bridge is initializing Update value if (PCI.CR[31:24] == 8hA5)		
	HARD reset		1b0		
RESERVED	[31:24], [13],[1]	8,1,1	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		Reserved		
	HARD reset		0		

PCI local space register 0

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

PCI.LSR0				PCIBASE.SH + 0x104	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_LSR	[28:20]	9	-	PCI local space register	sh:RW
	Operation		9b0 0000 0000: 1 Mbyte 9b0 0000 0001: 2 Mbyte 9b0 0000 0011: 4 Mbyte | | 9b1 1111 1111: 512 Mbyte		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		9b0 0000 0000		
PCI_MBARE0	0	1	-	PCI memory base address register 0 enable	sh:RW
	Operation		0: PCI.MBAR0 is disable 1: PCI.MBAR0 is enable		
	When read		Value		
	When written		Update value(sh)		
	HARD reset		0		
RESERVED	[31:29] [19:1]	22	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI local space register 1

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

PCI.LSR1				PCIBASE.SH + 0x108	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_LSR	[28:20]	9	-	PCI local space register	sh:RW
	Operation		9'b0 0000 0000: 1M byte 9'b0 0000 0001: 2M byte 9'b0 0000 0011: 4M byte | | 9'b1 1111 1111: 512M byte		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		9b0 0000 0000		
PCI_MBARE1	0	1	-	PCI memory base address register 1 enable	sh:RW
	Operation		0: PCI.MBAR1 is disable 1: PCI.MBAR1 is enable		
	When read		Value		
	When written		Update value(sh)		
	HARD reset		0		
RESERVED	[31:29] [19:1]	22	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI local address register 0

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

PCI.LAR0				PCIBASE.SH + 0x10c	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_LAR	[31:20]	12	-	PCI local address register	sh:RW
	Operation		PCI.LSR0.PCI_LSR == 9'b0 0000 0000: effective bits are [31:20] PCI.LSR0.PCI_LSR == 9'b0 0000 0001: effective bits are [31:21] PCI.LSR0.PCI_LSR == 9'b0 0000 0011: effective bits are [31:22] | | PCI.LSR0.PCI_LSR == 9'b1 1111 1111: effective bits are [31:29]		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		0		
RESERVED	[19:0]	20	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI local address register 1

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

Pci.lar1				PCIBASE.SH + 0x110	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_LAR	[31:20]	12	-	PCI local address register	sh:RW
	Operation		PCI.LSR1.PCI_LSR == 9'b0 0000 0000: effective bits are [31:20] PCI.LSR1.PCI_LSR == 9'b0 0000 0001: effective bits are [31:21] PCI.LSR1.PCI_LSR == 9'b0 0000 0011: effective bits are [31:22] | | PCI.LSR1.PCI_LSR == 9'b1 1111 1111: effective bits are [31:29]		
	When read		Value		
	When written		Update value (sh)		
	HARD reset		12h0 00		
RESERVED	[19:0]	20	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI interrupt register

The PCI interrupt register (PCI.INT) records the cause of an interrupt. Should multiple interrupts occur only the first cause is registered - so at most a single bit can be set in this register. Further information about the interrupt cause is recorded in the PCI.AIR(address) and PCI.CIR (command) registers described elsewhere.

When an interrupt is masked, (by the companion PCI.INTM register) but becomes active, the cause is registered in corresponding bit in this register even though no interrupt is passed to the interrupt controller (INTC). This register is a write '1' to clear register and is denoted as RWC.

PCI.INT				PCIBASE.SH + 0x114	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_TTADI	14	1	-	PCI target target-abort detection interrupt (master)	sh:RWC
	Operation		The PCI bridge has detected an illegal byte enable with memory or I/O read/write transfer (target).		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_TMTO	9	1	-	PCI target memory read/write timeout interrupt (target)	sh:RWC
	Operation		When the PCI bridge is the target, the master did not attempt a retry within the prescribed number of clocks (2^{15}) (detected only in the case of memory read operations).		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		

PCI.INT				PCIBASE.SH + 0x114	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MDEI	8	1	-	PCI master function disable error interrupt (master)	sh:RWC
	Operation		PCI bridge attempted a master transfer when such transfers are disabled (that is, when PCI.CMD.PCI_BM = 1b0).		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_APEDI	7	1	-	PCI address parity error detection interrupt (target)	sh:RWC
	Operation		Address parity error detected (target)		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_SDI	6	1	-	PCI PCI_SERR_N detection interrupt (master/target)	sh:RWC
	Operation		When PCI bridge is host Assertion of PCI_SERR_N detected (master and target)		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		

PCI.INT				PCIBASE.SH + 0x114	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DPEITW	5	1	-	PCI data parity error detection interrupt for target write (target)	sh:RWC
	Operation		When the PCI bridge is the target, a data parity error was detected while receiving a target write transfer (only detected when PCI.COM.PCI_PER = 1b1) (target)		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_PEDITR	4	1	-	PCI PCI_PERR_N detection interrupt for target read (target)	sh:RWC
	Operation		Assertion of PCI_PERR_N was detected when receiving a target read transfer (target)		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_TADIM	3	1	-	PCI target-abort detection interrupt for master (master)	sh:RWC
	Operation		When the PCI bridge is master PCI bridge detected target-abort (that is, PCI_DEVSEL_N was suddenly negated).		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		

PCI.INT				PCIBASE.SH + 0x114	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MADIM	2	1	-	PCI master-abort detection interrupt for master	sh:RWC
	Operation		When PCI bridge is master a master-abort was detected. (that is, PCI_DEVSEL_N undetected) (master)		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_MWPDI	1	1	-	PCI master write PCI_PERR_N detection interrupt (master)	sh:RWC
	Operation		When PCI bridge is the master the PCI bridge received PCI_PERR_N from the target when the data write to the target.		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		
PCI_MRDPEI	0	1	-	PCI master read data parity error detection interrupt (master)	sh:RWC
	Operation		When the PCI bridge is the master a data parity error was detected during a data read of the target.		
	When read		Value		
	When written		Write 1b1 to clear		
	HARD reset		1b0		

PCI.INT				PCIBASE.SH + 0x114	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[31:15], [13:10]	21	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI interrupt register mask

This register is the mask register for PCI.INT.

PCI.INTM				PCIBASE.SH + 0x118	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_TTADIM	14	1	-	PCI target target-abort detection interrupt mask	sh:RW
	Operation		1b1: Permits PCI.INT.PCI_TTADIM 1b0: Inhibits PCI.INT.PCI_TTADIM		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_TMTOM	9	1	-	PCI target memory read/write time out interrupt mask	sh:RW
	Operation		1b1: Permits PCI.INT.PCI_TMTO 1b0: Inhibits PCI.INT.PCI_TMTO		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.INTM				PCIBASE.SH + 0x118	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MDEIM	8	1	-	PCI master function disable error interrupt mask	sh:RW
	Operation		1b1: permits PCI.INT.PCI_MDEI 1b0: inhibits PCI.INT.PCI_MDEI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_APEDIM	7	1	-	PCI address parity error detection interrupt mask	sh:RW
	Operation		1b1: permits PCI.INT.PCI_APEDI 1b0: inhibits PCI.INT.PCI_APEDI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_SDIM	6	1	-	PCI PCI_SERR_N detection interrupt mask	sh:RW
	Operation		1b1: permits PCI.INT.PCI_SDI 1b0: inhibits PCI.INT.PCI_SDI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.INTM				PCIBASE.SH + 0x118	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DPEITWM	5	1	-	PCI data parity error detection interrupt for target write mask	sh:RW
	Operation		1b1: Permits PCI.INT.PCI_DPEITW 1b0: Inhibits PCI.INT.PCI_DPEITW		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_PEDITRM	4	1	-	PCI PCI_PERR_N detection interrupt for target read mask	sh:RW
	Operation		1'b1: Permits PCI.INT.PCI_PEDITR 1'b0: Inhibits PCI.INT.PCI_PEDITR		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_TADIMM	3	1	-	PCI target-abort detection interrupt for master mask	sh:RW
	Operation		1b1: Permits PCI.INT.PCI_TADIM 1b0: Inhibits PCI.INT.PCI_TADIM		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.INTM				PCIBASE.SH + 0x118	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MADIMM	2	1	-	PCI master-abort detection interrupt for master mask	sh:RW
	Operation		1b1: permits PCI.INT.PCI_MADIM 1b0: inhibits PCI.INT.PCI_MADIM		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_MWPDIM	1	1	-	PCI master write PCI_PERR_N detection interrupt mask	sh:RW
	Operation		1b1: permits PCI.INT.PCI_MWPDI 1b0: inhibits PCI.INT.PCI_MWPDI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_MRDPEIM	0	1	-	PCI master read data parity error detection interrupt mask	sh:RW
	Operation		1b1: permits PCI.INT.PCI_MRDPEI 1b0: inhibits PCI.INT.PCI_MRDPEI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.INTM				PCIBASE.SH + 0x118	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[31:15], [13:10]	21	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI error address information register

This register records the 32-bit PCI address at the time an error is detected.

PCI.AIR				PCIBASE.SH + 0x11c	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_AIR	[31:0]	32	-	PCI error address information register	sh:RO
	Operation		This register holds address information when PCI bridge finds an error.		
	When read		Value		
	When written		-		
	HARD reset		32hxxxxxxxx		

PCI error command information register

This register records the PCI command information at the time an error is detect detected.

PCI.CIR				PCIBASE.SH + 0x120	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PIOTEM	31	1	-	PCI memory/ IO transfer error for master	sh:RO
	Operation		Error occured during master transfer		
	When read		Value		
	When written		-		
	HARD reset		1bx		
PCI_RWTET	26	1	-	PCI read/write transfer error for target	sh:RO
	Operation		Error occurred in target read or target write transfer.		
	When read		Value		
	When written		-		
	HARD reset		1bx		
PCI_ECR	[3,0]	4	-	PCI error command register	sh:RO
	Operation		PCI command when error occurred.		
	When read		Value		
	When written		-		
	HARD reset		4bxxxx		
RESERVED	[30:27], [25:4]	26	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI arbiter interrupt register

In host bus bridge mode, this register records cause of interrupt.

- when multiple interrupts occur: only 1st cause is registered.
- when interrupt is disable: cause is registered in corresponding bit, no interrupt occurs

PCI.AINT				PCIBASE.SH + 0x130	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MBI	13	1	-	PCI master-broken interrupt	sh:RWC
	Operation		PCI_FRAME_N is not assert within 16 clock cycles, although the PCI bridge gave the bus.		
	When read		Value		
	When written		Write 1b1 is clear this register		
	HARD reset		1b0		
PCI_TBTOI	12	1	-	PCI target bus time-out interrupt	sh:RWC
	Operation		Case of first data transfer: PCI_TRDY_N or PCI_STOP_N have not asserted within 16 clock cycles Case of data transfer subsequent to the 2nd: PCI_TRDY_N or PCI_STOP_N have not asserted within 8 clock cycles		
	When read		Value		
	When written		Write 1b1 to clear this register		
	HARD reset		1b0		
PCI_MBTOI	11	1	-	PCI master bus time-out interrupt	sh:RWC
	Operation		PCI_IRDY_N has not returned within 8 clock cycles		
	When read		Value		
	When written		Write 1b1 to clear this register		
	HARD reset		1b0		

PCI.AINT				PCIBASE.SH + 0x130	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_TAI	3	1	-	PCI target-abort interrupt	sh:RWC
	Operation		PCI_DEVSEL_N is negated during the time taken to transfer the bus from a device to the PCI bridge.		
	When read		Value		
	When written		Write 1b1 is clear this register		
	HARD reset		1b0		
PCI_MAI	2	1	-	PCI master-abort interrupt	sh:RWC
	Operation		Master abort was detected when a device other than the PCI bridge has the bus.		
	When read		Value		
	When written		Write 1b1 is clear this register		
	HARD reset		1b0		
PCI_RDPEI	1	1	-	PCI read data parity error interrupt	sh:RWC
	Operation		Parity error was detected at the time of data read when a device other than the PCI bridge has the bus.		
	When read		Value		
	When written		Write 1b1 is clear this register		
	HARD reset		1b0		
PCI_WDPEI	0	1	-	PCI write data parity error interrupt	sh:RWC
	Operation		Parity error was detected at the time of data write when a device other than the PCI bridge has the bus.		
	When read		Value		
	When written		Write 1b1 is clear this register		
	HARD reset		1b0		

PCI.AINT				PCIBASE.SH + 0x130	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[31:14], [10:4]	25	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI arbiter interrupt mask register

This register is the mask register for PCI.AINT.

PCI.AINTM				PCIBASE.SH + 0x134	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MBIM	13	1	-	PCI master-broken interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_MBI 1b0: Inhibits PCI.AINT.PCI_MBI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_TBTOIM	12	1	-	PCI target bus time-out interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_TBTOI 1b0: Inhibits PCI.AINT.PCI_TBTOI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.AINTM				PCIBASE.SH + 0x134	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MBTOIM	11	1	-	PCI master bus time-out interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_MBTOI 1b0: Inhibits PCI.AINT.PCI_MBTOI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_TAIM	3	1	-	PCI target-abort interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_TAI 1b0: Inhibits PCI.AINT.PCI_TAI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_MAIM	2	1	-	PCI master-abort interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_MAI 1b0: Inhibits PCI.AINT.PCI_MAI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_RDPEIM	1	1	-	PCI read data parity error interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_RDPEI 1b0: Inhibits PCI.AINT.PCI_RDPEI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.AINTM				PCIBASE.SH + 0x134	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_WDPEIM	0	1	-	PCI write data parity error interrupt mask	sh:RW
	Operation		1b1: Permits PCI.AINT.PCI_WDPEI 1b0: Inhibits PCI.AINT.PCI_WDPEI		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
RESERVED	[31:14] [10:4]	25	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		1b0		

PCI arbiter bus master information register

In host bridge mode, this register records when the interrupt is invoked by PCI.AINT.

- when multiple interrupts occur: only 1st cause is registered.
- when interrupt is disable: cause is registered in corresponding bit, no interrupt occurs.

PCI.BMIR				PCIBASE.SH + 0x138	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_REQ4BME	4	1	-	PCI REQ4 bus master error	sh:RO
	Operation		-		
	When read		Value		
	When written		-		
	HARD reset		1bx		
PCI_REQ3BME	3	1	-	PCI REQ3 bus master error	sh:RO
	Operation		-		
	When read		Value		
	When written		-		
	HARD reset		1bx		
PCI_REQ2BME	2	1	-	PCI REQ2 bus master error	sh:RO
	Operation		-		
	When read		Value		
	When written		-		
	HARD reset		1bx		

PCI.BMIR				PCIBASE.SH + 0x138	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_REQ1BME	1	1	-	PCI REQ1 bus master error	sh:RO
	Operation		-		
	When read		Value		
	When written		-		
	HARD reset		1bx		
PCI_REQ0BME	0	1	-	PCI REQ0 bus master error	sh:RO
	Operation		SH-5 PCI bridge occurred error		
	When read		Value		
	When written		-		
	HARD reset		1bx		
RESERVED	[31:5]	27	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI PIO[1] address register

Refer to *Section 3.5.1: Configuration space access on page 133*.

PCI.PAR				PCIBASE.SH + 0x1c0	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_CCIE	31	1	-	PCI configuration cycle issue enable	sh:RO
	Operation		Hard fixed		
	When read		1b1		
	When written		-		
	HARD reset		1b1		
PCI_BN	[23:16]	8	-	PCI bus number	sh:RW
	Operation		8h00: bus number 0 8h01: bus number 1 | 8hFF: bus number 255		
	When read		Value		
	When written		Update value		
	HARD reset		Undefined		

1. This register is so named for historical reasons.

PCI.PAR				PCIBASE.SH + 0x1c0	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_DN	[15:11]	5	-	PCI device number	sh:RW
	Operation		5b00000: PCI_IDSEL= (PCI_AD[16] = 1) 5b00001: PCI_IDSEL= (PCI_AD[17] = 1) 5b00010: PCI_IDSEL= (PCI_AD[18] = 1) | 5b01111: PCI_IDSEL= (PCI_AD[31] = 1) 5b10000: PCI_AD[31,16] = 0 | 5b11111: PCI_AD[31,16] = 0		
	When read		Value		
	When written		Update value		
	HARD reset		Undefined		
PCI_FN	[10:8]	3	-	PCI function number	sh:RW -
	Operation		3b000: single function device or function 0 of multi-function device 3b001: function 1 of multi-function device | 3b111: function 7 of multi-function device		
	When read		Value		
	When written		Update value		
	HARD reset		Undefined		
PCI_CRA	[7:2]	6	-	PCI configuration register address	sh:RW
	Operation		Long word boundary		
	When read		Value		
	When written		Update value		
	HARD reset		Undefined		

PCI.PAR				PCIBASE.SH + 0x1c0	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RESERVED	[30:24] [1:0]	9	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI memory space bank register

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

PCI.MBR				PCIBASE.SH + 0x1c4	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMSBA	[31:18]	14	-	PCI memory space bank address register	sh:RW -
	Operation		Specifies the base address in PCI memory space to which SuperHyway accesses are mapped.		
	When read		Value		
	When written		Update value		
	HARD reset		14'bxxxx xxxx xxxx xx		
RESERVED	[17:0]	18	-	Reserved	sh:RES
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI I/O space bank register

Refer to *Section 3.2.9: Accessing PCI I/O space (SH-5 master) on page 119*.

PCI.IOBR				PCIBASE.SH + 0x1c8	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PIOSBA	[31:18]	14	-	PCI I/O space bank address register	sh:RW -
	Operation		Specifies the base address in PCI memory space to which the SuperHyway local accesses are mapped.		
	When read		Value		
	When written		Update value		
	HARD reset		14'bxxxx xxxx xxxx xx		
RESERVED	[17:0]	18	-	Reserved	sh:RES -
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI power management interrupt register

This register records the cause of the power management interrupt.

PCI.PINT				PCIBASE.SH + 0x1cc	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMD2	3	1	-	PCI power management D2 status transition interrupt	sh:RWC -
	Operation		Transition demand interrupt to the low power consumption mode to SuperHyway		
	When read		Value		
	When written		1b1: clear this bit		
	HARD reset		1b0		
PCI_PMD1	2	1	-	PCI power management D1 status transition interrupt	sh:RWC -
	Operation		Transition demand interrupt to the low power consumption mode to SuperHyway		
	When read		Value		
	When written		1b1: clear this bit		
	HARD reset		1b0		
PCI_PMD3H	1	1	-	PCI power management D3 hot status transition interrupt	sh:RWC -
	Operation		Transition demand interrupt to the low power consumption mode to SuperHyway		
	When read		Value		
	When written		1b1: clear this bit		
	HARD reset		1b0		

PCI.PINT				PCIBASE.SH + 0x1cc	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMD0	0	1	-	PCI power management D0 status transition interrupt	sh:RWC -
	Operation		Return demand interrupt from the low power consumption mode to SuperHyway		
	When read		Value		
	When written		1b1: clear this bit		
	HARD reset		1b0		
RESERVED	[31:4]	28	-	Reserved	sh:RES -
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI power management interrupt mask register

This is the mask register for PCI.PINT.

PCI.PINTM				PCIBASE.SH + 0x1d0	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMD2M	3	1	-	PCI power management D2 status transition interrupt mask	sh:RW -
	Operation		1b0: inhibits PCI.PINT.PCI_PMD2 1b1: permits PCI.PINT.PCI_PMD2		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_PMD1M	2	1	-	PCI power management D1 status transition interrupt mask	sh:RW -
	Operation		1b0: inhibits PCI.PINT.PCI_PMD1 1b1: permits PCI.PINT.PCI_PMD1		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		
PCI_PMD3HM	1	1	-	PCI power management D3 hot status transition interrupt mask	sh:RW -
	Operation		1b0: inhibits PCI.PINT.PCI_PMD3H 1b1: permits PCI.PINT.PCI_PMD3H		
	When read		Value		
	When written		Update value		
	HARD reset		1b0		

PCI.PINTM				PCIBASE.SH + 0x1d0	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PMD0M	0	1	-	PCI power management D0 status transition interrupt mask	sh:RW -
	Operation		1b0: inhibits PCI.PINT.PCI_PMD0 1b1: permits PCI.PINT.PCI_PMD0		
	When read		value		
	When written		update value		
	HARD reset		1b0		
RESERVED	[31:4]	28	-	Reserved	sh:RES -
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI memory space bank mask register

This is the mask register for PCI.MBR.

Refer to *Section 3.2.7: PCI memory space (SH-5 master) on page 117*.

Pci.mbmr				PCIBASE.SH + 0x1d8	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_MSBAMR	[28:18]	11	-	PCI memory space bank address mask register	sh:RW -
	Operation		0 0000 0000 00 : 256 Kbyte 0 0000 0000 01 : 512 Kbyte 0 0000 0000 11 : 1 Mbyte | | 1 1111 1111 11 : 512 Mbyte		
	When read		Value		
	When written		Update value		
	HARD reset		11'bx xxxx xxxx xx		
RESERVED	[31:29] [17:0]	21	-	Reserved	sh:RES -
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI I/O space bank address mask register

This register is the mask register for PCI.IOBR.

Refer to *Section 3.2.9: Accessing PCI I/O space (SH-5 master) on page 119*.

Pci.iobmr				PCIBASE.SH + 0x1dc	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_IOBAMR	[22:18]	5	-	PCI I/O space bank address mask register	sh:RW -
	Operation		0000 0 : 256 Kbyte 0000 1 : 512 Kbyte 0001 1 : 1 Mbyte 0011 1 : 2 Mbyte 0111 1 : 4 Mbyte 1111 1 : 8 Mbyte		
	When read		Value		
	When written		Update value		
	HARD reset		5bxxxx x		
RESERVED	[31:23] [17:0]	27	-	Reserved	sh:RES -
	Operation		Reserved		
	When read		0		
	When written		-		
	HARD reset		0		

PCI cache snoop control register n [n=0,1]

An external device can access SH-5 local memory via the PCI. When an external device accesses cacheable locations, the PCI can issue snoop commands to the on-chip caches via the SuperHyway. The PCI.CSCR[n] specifies this function that uses snoop address register 0/1.

PCI.CSCR[N], n=0,1				PCIBASE.SH + 0x210 + 4*n,	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
SNPMD	[1:0]	2	—	Snoop mode for PCI.CSAR[n]	sh:RW
	Operation		Specifies if PCI.CSAR[n] is compared with address requested by an external device. Also, specifies how snoop command is issued when PCI.CSAR[n] is compared.		
	When read		Returns if PCI.CSAR[n] is compared and how snoop command is issued when PCI.CSAR[n] is compared. 00: PCI.CSAR[n] is not compared. 01: Reserved 10: PCI.CSAR[n] is compared. If hit, snoop command isn't issued else issued. 11: PCI.CSAR[n] is compared. If hit. snoop command is issued, else not.	
	When written		Update value		
	HARD reset		Undefined		

PCI.CSCR[N], n=0,1				PCIBASE.SH + 0x210 + 4*n,	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
RANGE	[4:2]	3	—	Address range to be compared	sh:RW
	Operation		Specifies address range of pci.csar[n] to be compared		
	When read		Returns which address bits are compared. 000: PCI.CSAR[N].CADR[31:12] is compared (4 Kbyte) 001: PCI.CSAR[N].CADR[31:16] is compared (64 Kbyte) 010: PCI.CSAR[N].CADR[31:20] is compared (1 Mbyte) 011: PCI.CSAR[N].CADR[31:24] is compared (16 Mbyte) 100: PCI.CSAR[N].CADR[31:25] is compared (32 Mbyte) 101: PCI.CSAR[N].CADR[31:26] is compared (64 Mbyte) 110: PCI.CSAR[N].CADR[31:27] is compared (128 Mbyte) 111: PCI.CSAR[N].CADR[31:28] is compared (256 Mbyte) Valid only when PCI.CSCR[N].SNPMD=10 or 11.	
	When written		Update value		
	HARD reset		Undefined		
RESERVED	[31,5]	27	—	Reserved	RES
	Operation		Reserved		
	When read		Undefined		
	When written		Undefined		
	HARD reset		Undefined		

PCI cache snoop address register n [n=0,1]

The PCI.CSAR0/1 specifies the address to be compared with the PCI address requested by an external device.

PCI.CSAR[n], n=0,1				PCIBASE.SH + 0x218 + 4*n	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
CADR	[31:0]	32	—	Address to be compared	sh:RW
	Operation		Specifies address to be compared with the PCI address requested by external devices		
	When read		Returns value		
	When written		Update value		
	HARD reset		Undefined		

PCI PIO[1] data register

When accessed, this register will cause the generation of a configuration cycle on the PCI bus.

PCI.PDR				PCIBASE.SH + 0x220	
Field	**Bits**	**Size**	**Volatile?**	**Synopsis**	**Type**
PCI_PDR	[31:0]	32	—	PCI PIO data register	sh:RW-
	Operation		A read of this register will cause a PCI configuration read cycle on the PCI bus. A write to this register will cause a PCI configuration write cycle on the PCI bus.		
	When read		Returns value		
	When written		Update value		
	HARD reset		Undefined		

1. Note: this register is so named for historical reasons.

3.8 Register list

Name	Abbreviation	shwy R/W	PCI R/W	Size	Offset	Reset value	Definition
Version control register bank							
Version control register	PCI.VCR	R/W	R/W	64	0x000	TBD	page 138
PCI CSR bank							
PCI configuration register							
PCI vendor ID	PCI.VID	R	R	16	0x000	0x1054	page 141
PCI device ID	PCI.DID	R	R	16	0x002	TBD	page 141
PCI command	PCI.CMD	R/W	R/W	16	0x004	0x0006	page 143
PCI status	PCI.STATUS	R/W	R/W	16	0x006	0x0090	page 147
PCI revision ID	PCI.RID	R	R	8	0x008	0x00	page 151
PCI program interface	PCI.PIF	R/W	R	8	0x009	0x00	page 151
PCI sub class code	PCI.SUB	R/W	R	8	0x00A	0x00	page 151
PCI base class code	PCI.BCC	R/W	R	8	0x00B	0x06	page 154
PCI cache line size	PCI.CLS	R/W	R/W	8	0x00C	0x00	page 154
PCI latency timer	PCI.MLT	R/W	R/W	8	0x00D	0x00	page 155
PCI header type	PCI.HDR	R	R	8	0x00E	0x00	page 155
PCI BIST	PCI.BIST	R	R	8	0x00F	0x00	page 156
PCI I/O base addr	PCI.IBAR	R/W	R/W	32	0x010	0x0000 0000	page 157
PCI Mem0 base addr	PCI.MBAR0	R/W	R/W	32	0x014	0x0000 0008	page 158
PCI Mem1 base addr	PCI.MBAR1	R/W	R/W	32	0x018	0x0000 0000	page 160
PCI subsystem vendor ID	PCI.SVID	R/W	R	16	0x02C	TBD	page 162

Name	Abbreviation	shwy R/W	PCI R/W	Size	Offset	Reset value	Definition
PCI subsystem ID	PCI.SID	R/W	R	16	0x02E	TBD	page 162
PCI capabilities pointer	PCI.CP	R	R	8	0x034	0x40	page 163
PCI interrupt line	PCI.INTLINE	R/W	R/W	8	0x03C	0x00	page 163
PCI interrupt pin	PCI.INTPIN	R	R	8	0x03D	0x01	page 164
PCI minimum grant	PCI.MINGNT	R	R	8	0x03E	0x01	page 164
PCI maximum latency	PCI.MAXLAT	R	R	8	0x03F	0x01	page 165
PCI capability ID	PCI.CID	R	R	8	0x040	0x01	page 165
PCI next item pointer	PCI.NIP	R	R	8	0x041	0x00	page 166
PCI power management capability	PCI.PMC	R/W	R/W	16	0x042	16bxxxx xxx0 0000 0010	page 167
PCI power management control/ status	PCI.PMCSR	R/W	R/W	16	0x044	16bxxx0 000x 0000 0000	page 169
PCI PMCSR bridge support extension	PCI.PMCSR_BSE	R	R	8	0x046	8bxx00 0000	page 173
PCI power consumption/ dissipation data	PCI.PCDD	R	R	8	0x047	0x00	page 174
PCI local register							
PCI control register	PCI.CR	R/W		32	0x100	0x0000 00*0	page 176
PCI local space register 0	PCI.LSR0	R/W		32	0x104	0x0000 0000	page 181
PCI local space register 1	PCI.LSR1	R/W		32	0x108	0x0000 0000	page 182
PCI local address register0	PCI.LAR0	R/W		32	0x10C	0x0000 0000	page 183

Name	Abbreviation	shwy R/W	PCI R/W	Size	Offset	Reset value	Definition
PCI local address register1	PCI.LAR1	R/W		32	0x110	0x0000 0000	page 184
PCI interrupt register	PCI.INT	R/W		32	0x114	0x0000 0000	page 185
PCI interrupt mask register	PCI.INTM	R/W		32	0x118	0x0000 0000	page 189
PCI error address information register	PCI.AIR	R		32	0x11C	0x0000 0000	page 193
PCI error command information register	PCI.CIR	R		32	0x120	0x0000 0000	page 194
PCI arbiter interrupt register	PCI.AINT	R/W		32	0x130	0x0000 0000	page 195
PCI arbiter interrupt mask register	PCI.AINTM	R/W		32	0x134	0x0000 0000	page 197
PCI arbiter bus master information register	PCI.BMIR	R		32	0x138	0x0000 00xx	page 200
PCI PIO address register	PCI.PAR	R/W		32	0x1C0	0x80xx xxxx	page 202
PCI memory space bank register	PCI.MBR	R/W		32	0x1C4	0xxx00 0000	page 204
PCI I/O space bank register	PCI.IOBR	R/W		32	0x1C8	0xxxxx 0000	page 204
PCI power management interrupt register	PCI.PINT	R/W		32	0x1CC	0x0000 0000	page 206
PCI power management interrupt mask register	PCI.PINTM	R/W		32	0x1D0	0x0000 0000	page 208
PCI memory space bank mask register	PCI.MBMR	R/W		32	0x1D8	0x0000 0000	page 211

Name	Abbreviation	shwy R/W	PCI R/W	Size	Offset	Reset value	Definition
PCI I/O space bank mask register	PCI.IOBMR	R/W		32	0x1DC	0x00000000	page 211
PCI cache snoop control register0/1	PCI.CSCR0/1	R/W		32	0x210,0x214	0x00000000	page 212
PCI cache snoop address register0/1	PCI.CSAR0/1	R/W		32	0x218,0x21C	0x00000000	page 214
PCI PIO data register	PCI.PDR	R/W		32	0x220	0xxxxxxxxx	page 214

3.9 Pin list

Signal	Pins	Type	Pin	POR H/S	Description
AD[31:0]	32	TRI_PCI33		0/Z	PCI address/data bus - time-multiplexed address and data bus. Each bus transaction consists of an address phase followed by one or more data phases.
CBE0_N CBE1_N CBE2_N CBE3_N	4	TRI_PCI33		0/Z	PCI command/byte enable – time multiplexed bus command and byte enables. Selects the type of transaction during the address phase and the byte enables during the data phases.
PAR	1	TRI_PCI33		0/Z	PCI parity signal - generates/checks even parity across AD[31:0] and C/BE[3:0].
PCICLK	1	IN_PCI33		-/-	PCI clock - provides timing for all transactions on the PCI bus.
FRAME_N	1	STRI_PCI33		Z/Z	PCI frame - driven by the current initiator and indicates the start and duration of a transaction.
TRDY_N	1	STRI_PCI33		Z/Z	PCI target ready - driven by the selected target. Indicates the target's ability to complete the current data phase of the transaction.
IRDY_N	1	STRI_PCI33		Z/Z	PCI initiator ready – driven by the current bus master. During a write, indicates that valid data is present on the AD[31:0] lines. During a read indicates the master is ready to accept data.
STOP_N	1	STRI_PCI33		Z/Z	PCI stop - driven by the selected target to stop the current transaction.
LOCK_N	1	STRI_PCI33		Z/Z	PCI lock.
IDSEL	1	IN_PCI33		-	PCI Idsel - input to the PCI device to select for configuration cycles (only for normal mode).

Signal	Pins	Type	Pin	POR H/S	Description
DEVSEL_N	1	STRI_PCI33		Z/Z	PCI device select - indicates the driving device has decoded its address as the target. As an input indicates that a device has been selected.
INTB_N INTC_N INTD_N	3	IN_PCI33		-/-	Interrupt D,C,B - indicates a PCI device is requesting an interrupt. Only for host.
INTA_N	1	TRI_PCI33		-/Z	Interrupt A – indicates a PCI device is requesting an interrupt in host bus bridge mode. Output to request an interrupt in normal mode.
REQ1_N REQ2_N REQ3_N	3	IN_PCI33		-/-	PCI bus request (only in host bus bridge mode)
GNT1_N GNT2_N GNT3_N	3	OUT_PCI33		1/-	PCI bus grant (only in host bus bridge mode)
REQ0_N	1	INOUT_PCI33		-/1	PCI bus request (input for host bus bridge mode, output for normal mode)
GNT0_N	1	INOUT_PCI33		1/-	PCI bus grant (output for host bus bridge mode, input for normal mode)
SERR_N	1	INOUT_PCI33			PCI system error
PERR_N	1	INOUT_PCI33			PCI parity error
PME_N	1	INOUT_PCI33			PCI power management event
PCICKSEL	1	IN		-/-	PCI clock select 0: internal PCI bridge move by internal clock[a] 1: internal PCI bridge move by pci_clk

Signal	Pins	Type	Pin	POR H/S	Description
HOSTEN	1	IN_LVTTL		-/-	Strapping pin. Sets PCI bridge mode. 0 = normal mode, 1= host bus bridge mode, Sampled on the rising edge of RESET_n.

a. This mode is not supported by SH-5 eval chip implementation.

3.10 References

For more information on PCI, see:

- *PCI local bus specification revision 2.2* (available through the pci-sig, http://www.pcisig.com),
- PCI bus power management interface specification revision 1.1 (available through the pci-sig, http://www.pcisig.com).

A good companion to experienced and novice PCI users alike is:

- *PCI system architecture* (Tom Shanley, Don Anderson, MindShare, Inc., http://www.mindshare.com).

Index

R

S

T

U

V

W