SABLE IO SPECIFICATION

Revision/Update Information: Rev 1.3

4000 SYSTEMS AND SERVERS
DIGITAL EQUIPMENT CORPORATION
MLO5-5/E71

-

CONTENTS

FIGURES

TABLES

CHAPTER 1

REVISION HISTORY

This chapter contains the current revision history of the Motherboard specification.

1.1 REV 0.0

Initial Draft.
Contributors:

— Victoria Triolo
— Rachael Berman
— Tom Hunt
— Fidelma Hayes
— Lee Ridlon
— Paul Rotker
— Andy Russo
— Andy Stewart
— Nick Warchol
— Ralph Ware
— Sue Yuryan

1.2 REV 0.1

Revised from initial review. Contributors:

— Victoria Triolo
— Rachael Berman
— Tom Hunt
— Fidelma Hayes
— Jim Janetos
— Lee Ridlon
— Paul Rotker
— Andy Russo
— Andy Stewart
— Nick Warchol

— Ralph Ware

- Added a 3rd PCI slot
- Changed polarity in the CBUS Mask lines CAD[127:124] and CAD[95:92] in Table 39.
- IOCSR - added several bits.
- Added better PCI addressing map.
- Added Standard I/O block diagram.

1.3 REV 1.0

Revised from specification review. Contributors:

— Victoria Triolo
— Rachael Berman
— Carl Furbeck
— Tom Hunt
— Fidelma Hayes
— Jim Janetos
— Lee Ridlon
— Paul Rotker
— Andy Russo
— Andy Stewart
— Nick Warchol
— Ralph Ware

1.4 REV 1.1

Revised with Standard I/O changes. Contributors:

— Victoria Triolo
— Rachael Berman
— Fidelma Hayes
— Andy Russo

1.5 REV 1.2

Revise with review comments from Revision 1.1. Contributors:

— Victoria Triolo
— Rachael Berman

1.6 REV 1.3

Revise with review comments from Revision 1.2. Contributors:

— Victoria Triolo

— Andy Russo

— Rachael Berman

CHAPTER 2

INTRODUCTION

2.1 Scope

This document describes the Sable I/O internal subsystem modules. This includes the Motherboard module, which is also the backplane of the Sable system, the Standard I/O module, and the Remote I/O Module. All three modules are required for all Sable systems.

2.2 Requirements

The major components of the Sable I/O subsystem are:

1. CBUS-PCI bridge, (referred to as the T2 array)
2. 3 32-bit PCI slots
3. 8 EISA slots
4. 1 Ethernet port switchable between 10BaseT and AUI (twisted pair and thick-wire)
5. 1 fast SCSI port
6. PCI-EISA bridge
7. Interrupt controller
8. Keyboard and mouse controller
9. Parallel port
10. 2 serial ports
11. Floppy controller
12. 8 KB NVRAM
13. 512 KB Flash ROM
14. TOY clock - Real time clock
15. Speaker
16. I^2C controller and peripheral devices.

These features are implemented across three modules; the Motherboard, the Standard I/O module, and the Remote I/O module.

2.2.1 Motherboard Features

The Motherboard is the Sable I/O backplane. It contains System bus (called the CBUS), PCI and EISA slots. The CBUS clocking, PCI clocking, EISA clocking and EISA 14.1818 oscillator are also contained on the Motherboard. There is a 299 pin gate array, called the T2, that is the CBUS to PCI bridge. The Motherboard also has a slot for the standard I/O module. The following is on the Motherboard:

1. 7 CBUS slots
2. 3 32-bit PCI slots
3. 8 EISA slots
4. 1 Standard I/O Module slot
5. CBUS clock generator/distributor
6. PCI clock generator/distributor
7. EISA clock distributor
8. CBUS-PCI bridge, (referred to as the T2 array)

2.2.2 Standard I/O Module Features

The Standard I/O module resides on the internal 32 bit PCI bus. The PCI-EISA bridge resides on this card. The card is standard on all systems. The following is on the Standard I/O module:

1. Digital DC1003 PCI-Ethernet chip (TULIP)
2. NCR 53C810 PCI-SCSI I/O processor (CUTTHROAT)
3. Intel Mercury PCI-EISA bridge chip set (PCEB and the ESC)
4. TOY Clock - Dallas DS1287A device - Real time clock
5. Speaker Control
6. 8KB NVRAM
7. 512 KB Flash ROM
8. Keyboard and Mouse Controller - Intel/Phoenix 8242
9. Combo chip - National PC87312
   - Parallel port
   - Floppy Controller
   - 2 Serial ports
10. I^2C - Controller, Expander, and ROM
11. Interrupt controller for I/O subsystem (PCI and EISA) (8259s)
12. Ethernet ROM
13. Reset and sysevent logic
14. Fan Speed Monitor
15. Remote Fault Management Connector

16. External PCI sub-arbiter [1]
17. 3.3 Volt and -5 Volt generation.

2.2.3 Remote I/O Module Features

The Remote I/O module is mounted inside the back of the system enclosure. It is electrically connected to the system via a 60 pin cable to the Standard I/O Module. It contains the external physical mounting connections for the keyboard and mouse, parallel port, and serial ports. In addition, it contains a jumpered connection for the speaker, which is independently mounted at the back of the enclosure next to the Remote I/O.

2.2.4 Backplane Features

2.2.4.1 CBUS

The CBUS used in Sable is a derivative of the CBUS developed for Cobra. This is the main interconnect between memory, CPUs and the I/O subsystems. There are eight nodes on the CBUS in Sable. Of these, seven nodes are pluggable. The 7 pluggable slots use 240 pin connectors. T2 is the eighth CBUS node. The CBUS is running at 24 ns. (41.67 MHz).

2.2.4.2 PCI

The internal I/O subsystem bus is a 32 bit PCI bus running at 33 MHz. There are three 120 pin PCI connectors for plug-in options. The PCI interfaces to the Ethernet port, SCSI port and the PCI-EISA bridge, all of which reside on the Standard I/O module.

2.2.4.3 EISA

Sable contains 8 EISA slots capable of running at 33 MB/sec. All EISA slots are master capable.

2.2.5 Non-Goals

This specification does not describe optional PCI and EISA modules or the various storage media supported. It also does not describe the External I/O module. For further information, check the Sable System Specification.

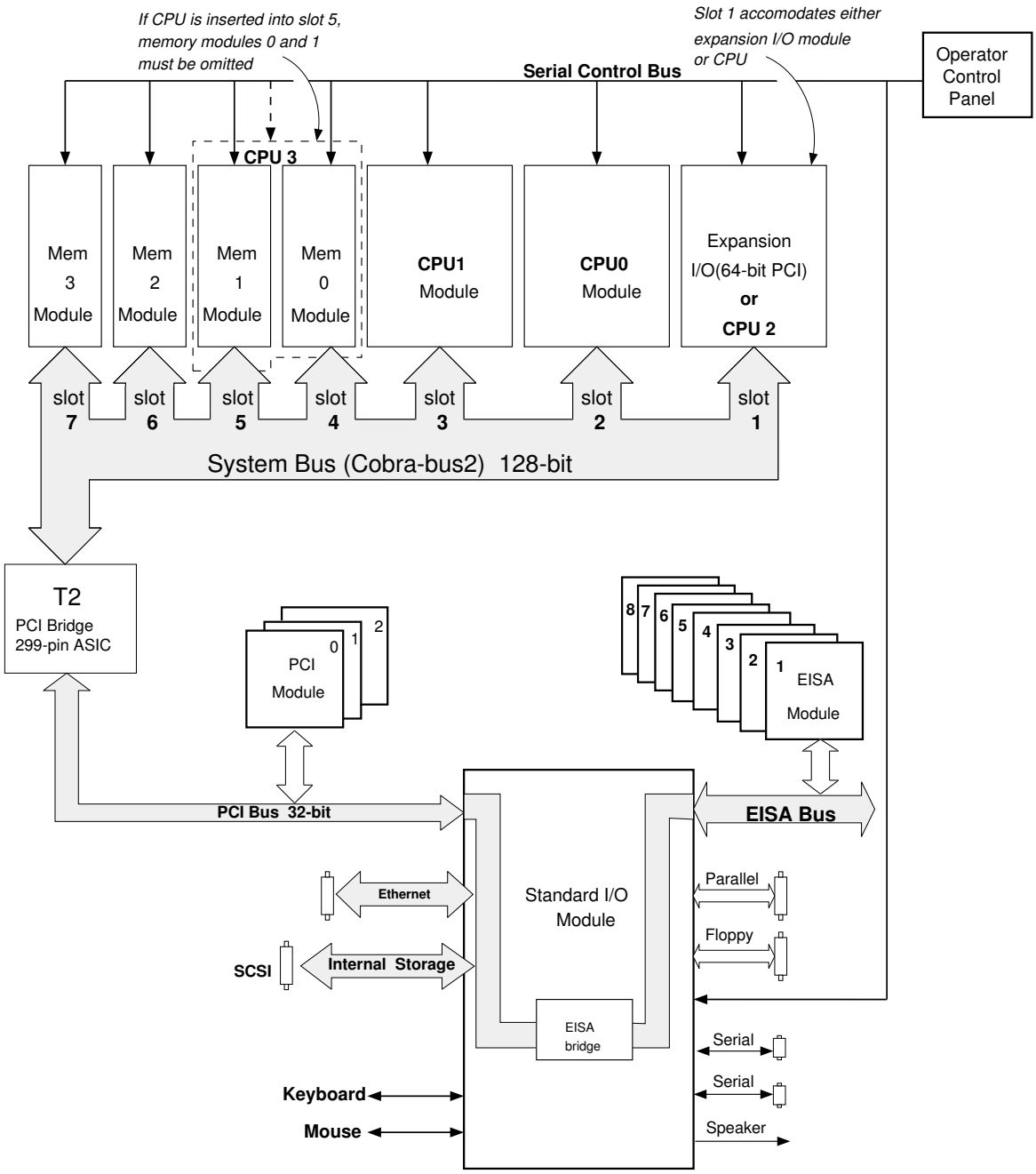
2.3 Block Diagrams

[1] The external PCI sub-arbiter arbitrates between the SCSI controller, and PCI Slot 2.

2.3.1 System Block Diagram

See Figure 1.

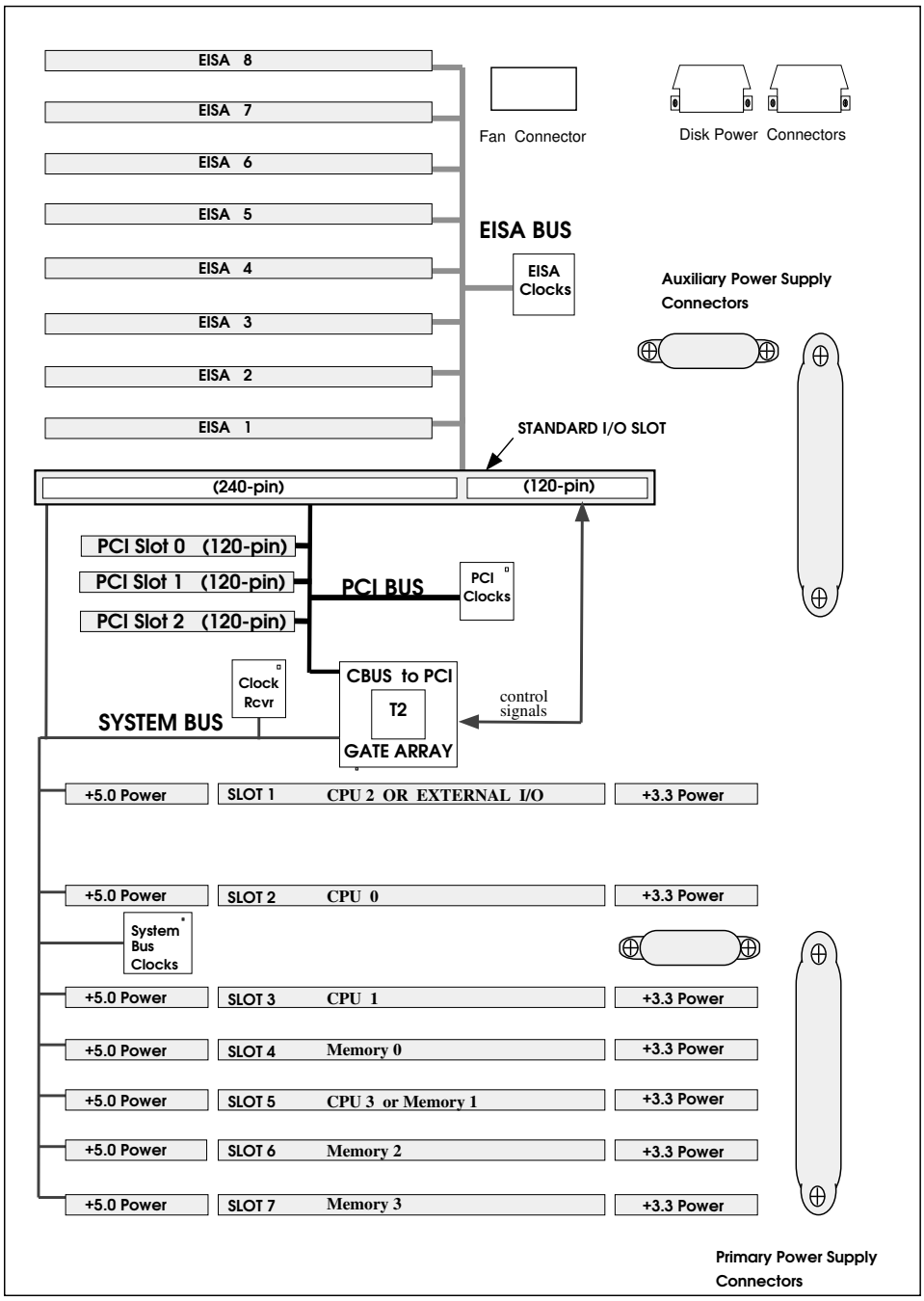
Figure 1: System Block Diagram



SAB_MOM3_BLOCK.DOC
PS @ 85%

2.3.2 Motherboard Physical Placement

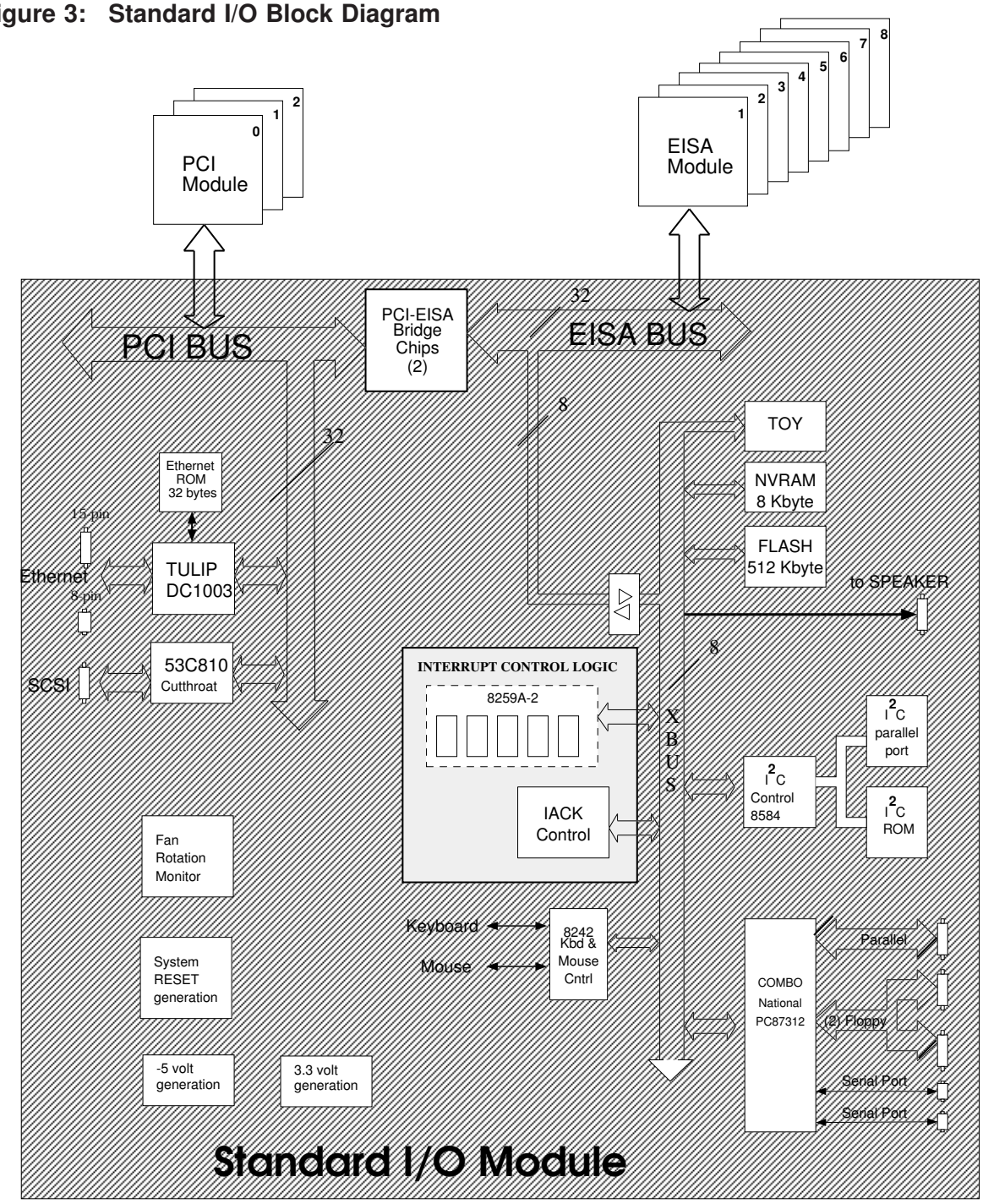
Figure 2: Motherboard Physical Placement



MOTHER_SIMPLE.DOC
PS @ 110%

2.3.3 Standard I/O Block Diagram

Figure 3: Standard I/O Block Diagram



STD_IO_BLOCK2.DOC
PS @ 75%

2.4 Pointers to Related Documentation

This document does not replicate information available elsewhere. The following document is useful:

- Sable System Specification

The rest of the Sable specifications are the SABLE_SPECS notesfile. Contact Subhash Dandage for membership, (PSDVAX::DANDAGE).

Contact Barry Maskas (BAMVAX::MASKAS) for a CBUS specification.

The PCI specification is available through the PCI notesfile. Contact Todd Comins (TECRUS::COMINS) for membership.

There are several chip specifications that are also useful:

- NCR 53C810, PCI-SCSI chip. Contact Bryan Cook, (508-858-0100).
- DC1003, TULIP, For the latest specification, contact Gady Daniely, JEREMY::GADY.
- Intel's 8259A-2, Interrupt controller. See Intel's Peripheral data book.
- Intel's Mercury chip set, PCI-EISA Bridge set. See Intel's 82420/82430 PCIset ISA and EISA b book.
- Dallas Semiconductor data book for RTC
- NVRAM - XICOR's X2864A EEPROM spec
- National's PC87312 spec
- Phillip's I^2C data books.
- Phoenix MultiKey/42 Technical Reference

CHAPTER 3

FUNCTIONALITY

3.1 CBUS

CBUS is the Sable system bus. It is a 128-bit bus that provides a snooping protocol for write back cache coherent 32-byte block read and write transactions to system memory address space. It supports up to 4 CPU modules, up to 4 memory modules, 1 external I/O module and a CBUS-PCI bridge. There are a total of eight nodes. Sable uses a new version of the CBUS which was originally developed for the Cobra project. The CPUs, memory and I/O subsystems all communicate with each other via the CBUS. The Sable implementation has added a number of features to the original CBUS. These include:

- Direct Mapped Access to I/O devices. Features added to support this are:
  - Retry mechanism added to bus
  - Longword write masks
- Support for PCI/EISA Locks via priority I/O arbitration mode.
- Expansion to CID filed to allow support for eight CBUS nodes.
- Arbitration to support up to 4 CPU nodes.
- Different CBUS slot definitions:
  1. Slot #1 - External I/O or CPU module 2
  2. Slot #2 - CPU 0
  3. Slot #3 - CPU 1
  4. Slot #4 - Memory 0
  5. Slot #5 - Memory 1 or CPU module 3
  6. Slot #6 - Memory 2
  7. Slot #7 - Memory 3
- Extended addressing to support 40-bit EV addressing.

A diagram of the Motherboard connector layout is shown in Figure 2.

3.1.1 CBUS Clock Generator/Distributor

The CBUS clock generator/distributor circuit is located on the Sable Motherboard. This clock is asynchronous to the CPU clock, PCI clock and EISA clocks. The clock period is 24 ns which translates to a clock frequency of 41.67 MHz. The outputs of the clock generator are PHI1, PHI1_l, PHI3, PHI3_L. The clock is made up of two phases where phase 3 is 270 degrees behind phase 1. The lock is distributed

as an ECL differential pair on tightly controlled/matched 50 ohm lines. Each CBUS module and the T2 gate array receive their own copy of the CBUS clock. If the oscillator stops, the CBUS reset signal will assert. Refer to the CBUS specification for more details.

NOTE

The T2 design requires the CBUS clock to always be faster than the PCI clock.

3.2 PCI

Sable supports a 32-bit internal PCI bus. Connected to the Sable's PCI bus are the following:

1. T2 array
2. PCI slot #0
3. PCI slot #1
4. PCI slot #2
5. Tulip chip, PCI-Ethernet bridge
6. NCR Cutthroat chip, PCI-SCSI bridge
7. Intel Mercury chip set, PCI-EISA bridge

3.2.1 PCI Clock Generator/Distributor

The PCI clock generator/distributor is located on the Sable Motherboard. It is a 33 MHz clock. The clock is distributed to the T2, PCI slots and four copies to the Standard I/O module.

3.2.2 PCI Arbitration

The PCEB component, of the Intel PCI to EISA bridge chipset on the Standard I/O module, provides an integrated PCI arbiter. The Sable system has one more PCI device than is supported by this arbiter, so one of the PCEB request and grant pairs is multiplexed between two devices by an external sub-arbiter PAL on the Standard I/O module. The ARB PAL sub-arbitrates between the Cutthroat SCSI controller, and PCI slot 2.

Arbitration is performed with rotating priority. See the databook for the PCEB for more information.

3.2.3 PCI/EISA Interrupts

All Sable PCI, EISA, ISA, and other I/O interrupts will be handled by five Intel 8259As arranged in an 8086 Fully Nested Mode cascade and in a level triggered mode. The hardware is located on the Standard I/O Module. Firmware support will be required for acknowledge and end of interrupt (EOI) handling. There is one master and 4 slave 8259s. See Figure 4 which illustrates the interrupt hierarchy, and Chapter 8, which describes the hardware in more detail.

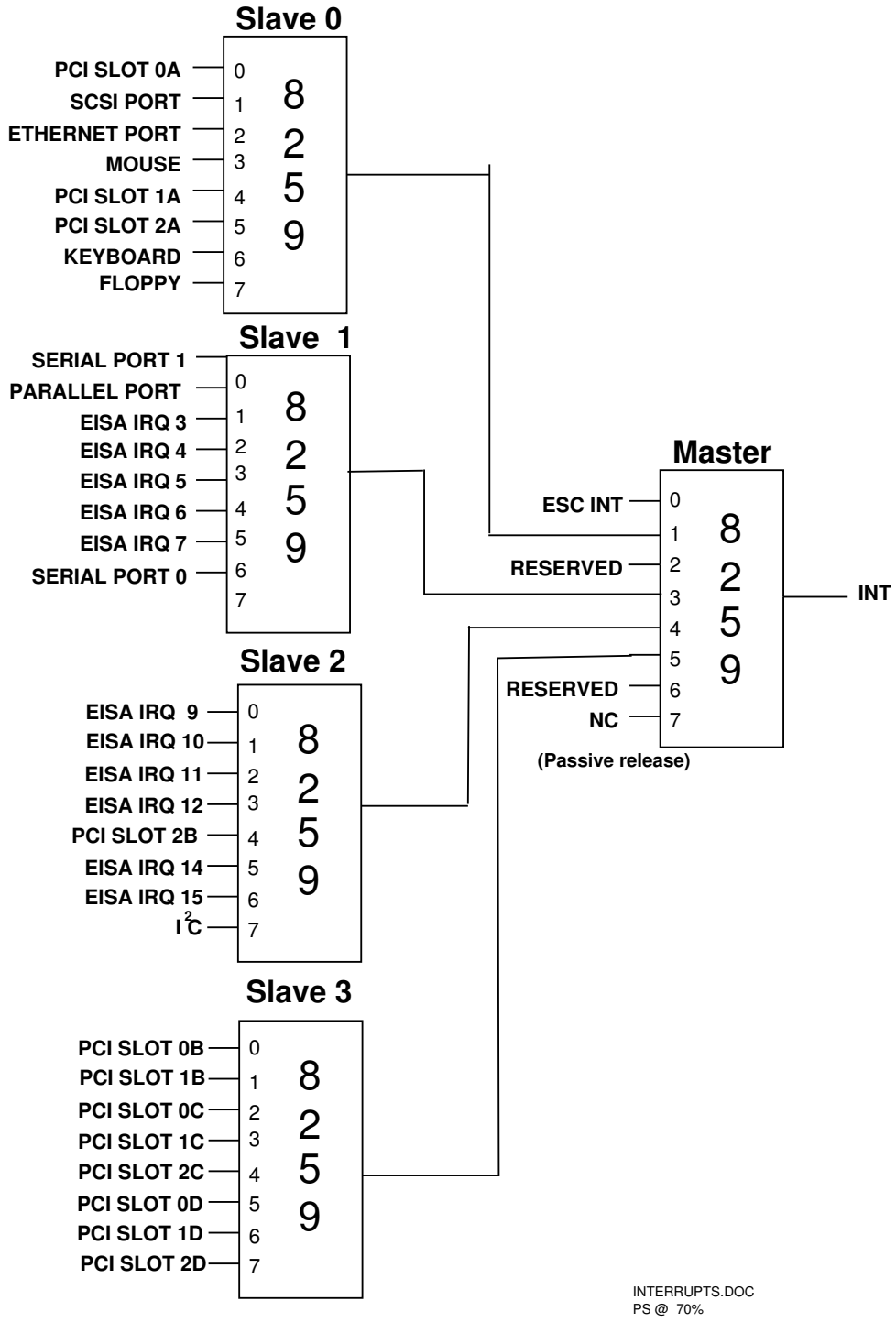
3.2.3.1 Interrupt sequence

1. An EISA, PCI or XBUS device posts an interrupt by asserting its interrupt line to the interrupt control logic.
2. The interrupt control logic asserts the INT line to the T2 gate array.
3. The T2 asserts the CIRQ<0> L to the CPUs.
4. A CPU (called CPUx) performs an I/O read to the IVR register (programmable address).
5. The T2 allocates an internal I/O buffer to the first CPU which did the read to the IVR register (CPUx)
6. The T2 passes the IVR read to the PCI bus where the interrupt control logic deals with it.
7. The T2 de-asserts the CIRQ<0> L and retries CPUx.
8. If a different CPU attempts to service the same interrupt as CPUx, it gets a passive release vector from T2.
9. The interrupt control logic sends the T2 the interrupt vector.
10. T2 passes the interrupt vector to CPUx the next time CPUx does the I/O read to the IVR.
11. CPUx handles the interrupt, and asserts SEOI when it has completed the task. See Section 3.2.3.3. The INT signal will de-assert during the SEOI.
12. T2 then allows the CIRQ<0> L line to follow INT. If another interrupt had become asserted during the interrupt handling process, INT will re-assert immediately following the SEOI.

3.2.3.2 Interrupt priority

The 8259As are programmable. See Section 8.3.2 for more information.

Figure 4: Cascaded 8259s Interrupt controller



3.2.3.3 SEOI - Specific End of Interrupt

In the mode that Sable uses the 8259s, an SEOI command must be issued by the firmware twice. The cascaded interrupt controller requires the two SEOI commands,

once for the master and once for the slaves. Since Sable is a multi-CPU system, the Specific End of Interrupt must be issued which includes as part of the command the IS level to be reset. See the 8259 specification for more information regarding programming.

3.2.3.4 NMI

An NMI is an interrupt requiring immediate attention. The ESC generates NMI interrupts based upon various hardware and software events:

1. PCI PERR#
2. PCI SERR#
3. Add-In Board Parity Errors on the EISA expansion bus.
4. Fail-Safe Timer Timeout
5. EISA Bus Timeout
6. Software Generated NMI

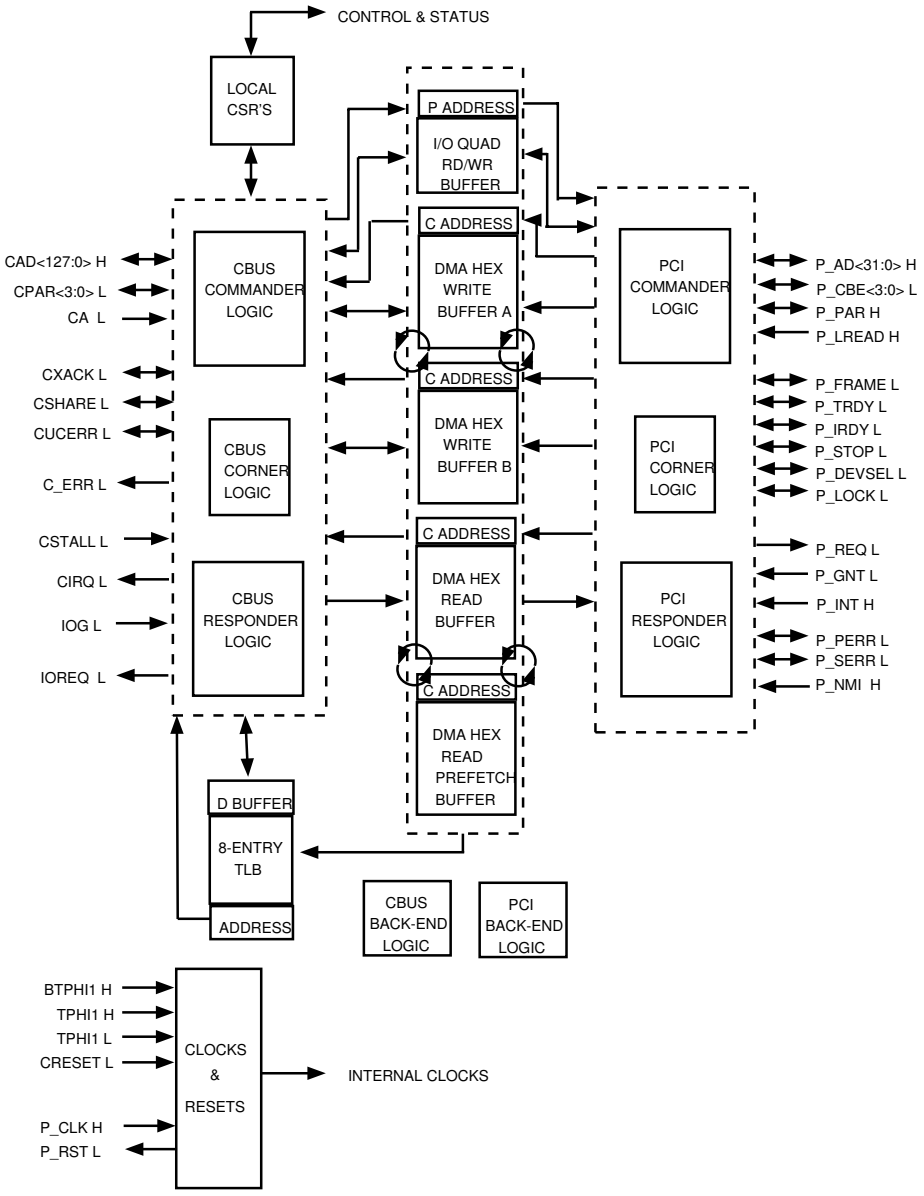
All NMI sources can be enabled or disabled by individual enable/disable bits or by one NMI enable/disable bit. The ESC generates NMI directly to the T2 gate array. There is also an enable/disable bit in the T2. The T2 will assert CERR L if NMIs are enabled in the IOCSR and the ESC asserts the NMI signal.

3.3 T2 gate array

The T2 gate array is a 60K 299 pin ceramic PGA. It is a CBUS node which provides a bridge to a 32-bit PCI bus running at 33 MHz. The T2 is a fully compliant PCI bridge. All I/O, DMA and interrupt transactions to the 32-bit PCI bus occur through the T2 array. The T2 provides logic for translating and extending the CBUS address space into PCI address space. The address decode in the T2 is described in Chapter 5. It also provides read and write channels with an 8 entry TLB and read/write I/O channel.

3.3.1 T2 Block diagram

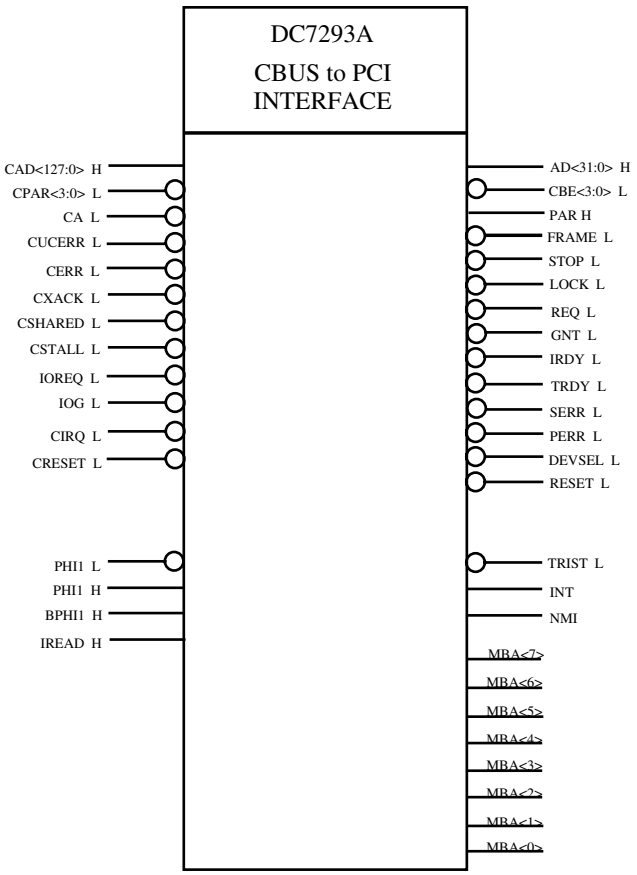
Figure 5: T2 Block Diagram



T2 GATE ARRAY BLOCK DIAGRAM
PS @95%

3.3.2 T2 Signals

Figure 6: T2 Signals



T2_PINOUT.DOC
100%

3.3.2.1 PCI Signals

3.3.2.1.1 P_AD[31:0] H - Bidirectional, TTL, BD8TRP, 8ma

Address and Data are multiplexed on the same PCI pins. A bus transaction consists of an address phase followed by one or more data phases. PCI supports both read and write bursts. The address phase is the clock cycle in which P_FRAME L is asserted. During the address phase P_AD[31:0] H contain a physical address (32 bits). For I/O, this is a byte address; for configuration and memory it is a DWORD address. During data phases P_AD[7:0] H contain the least significant byte (lsb) and P_AD[31:24] H contain the most significant byte (msb). Write data is stable and valid when P_IRDY L is asserted and read data is stable and valid when P_TRDY L is asserted. Data is transferred during those cycles where both P_IRDY L and P_TRDY L are asserted.

3.3.2.1.2 P_CBE[3:0] L - Bidirectional, TTL, BD8TRP, 8ma

Bus Command and Byte Enables are multiplexed on the same PCI pins. During the address phase of a transaction, P_CBE[3:0] L define the bus command (refer to the PCI specification, Section 3.1 for bus command definitions). During the data phase, P_CBE[3:0] L are used as byte enables. The byte enables are valid for the entire data phase and determine which byte lanes carry meaningful data. P_CBE[0] L applies to byte 0 (lsb) and P_CBE[3] L applies to byte 3 (msb).

3.3.2.1.3 P_PAR H - Bidirectional, TTL, BD8TRP, 8ma

Parity is even parity across P_AD[31:0] H and P_CBE[3:0] L. Parity generation is required by all PCI agents. P_PAR H is stable and valid one clock after the address phase. For data phases, P_PAR H is stable and valid one clock after either P_IRDY L is asserted on a write transaction or P_TRDY L is asserted on a read transaction. Once P_PAR H is valid, it remains valid until one clock after the completion of the current data phase. (P_PAR H has the same timing as P_AD[31:0] but delayed by one clock). The master drives P_PAR H for address and write data phases; the target drives P_PAR H for read data phases.

3.3.2.1.4 P_FRAME L - Bidirectional, TTL, BD8TRP, 8ma

Cycle Frame is driven by the current master to indicate the beginning and duration of an access. P_FRAME L is asserted to indicate a bus transaction is beginning. While P_FRAME L is asserted, data transfers continue. When P_FRAME L is de-asserted, the transaction is in the final data phase.

3.3.2.1.5 P_IRDY L - Bidirectional, TTL, BD8TRP, 8ma

Initiator Ready indicates the initiating agent's (bus master's) ability to complete the current data phase of the transaction. P_IRDY L is used in conjunction with P_ TRDY L. A data phase is completed on any clock both P_IRDY L and P_TRDY L are sampled asserted. During a write, P_IRDY L indicates that valid data is present on P_AD[31:0] H. During a read, it indicates the master is prepared to accept data. Wait cycles are inserted until both P_IRDY L and P_TRDY L are asserted together.

3.3.2.1.6 P_TRDY L - Bidirectional, TTL, BD8TRP, 8ma

Target Ready indicates the target agent's (selected device's) ability to complete the current data phase of the transaction. P_TRDY L is used in conjunction with P_IRDY L. A data phase is completed on any clock cycle both P_TRDY L and P_IRDY L are sampled asserted. During a read, P_TRDY L indicates that valid data is present on P_AD[31:0] H. During a write, it indicates the target is prepared to accept data. Wait cycles are inserted until both P_IRDY L and P_TRDY L are asserted together.

3.3.2.1.7 P_STOP L - Bidirectional, TTL, BD8TRP, 8ma

Stop indicates the current target is requesting the master to stop the current transaction.

3.3.2.1.8 P_LOCK L - Input, TTL

Lock indicates an atomic operation that may require multiple transactions to complete. When P_LOCK L is asserted, non-exclusive transactions may proceed to an address that is not currently locked. A grant to start a transaction on PCI does not guarantee control of P_LOCK L. Control of P_LOCK L is obtained under its own protocol in conjunction with P_GNT L. It is possible for different agents to use PCI while a single master retains ownership of P_LOCK L. If a device implements Executable Memory, it must also implement P_LOCK L and guarantee complete access exclusion in that memory. A target of an access that supports P_LOCK L must provide exclusion to a minimum of 16 bytes (aligned). Host bridges that have system memory behind them must also implement P_LOCK L.

3.3.2.1.9 P_DVSEL L - Bidirectional, TTL, BD8TRP, 8ma

Device Select, when actively driven, indicates the driving device has decoded its address as the target of the current access. As an input, P_DVSEL L indicates whether any device on the bus has been selected.

3.3.2.1.10 P_REQ L - Output, BD8TRP, 8ma

Request indicates to the arbiter that this agent desires use of the bus. This is a point to point signal. Every master has its own P_REQ L.

3.3.2.1.11 P_GNT L - Input, TTL

Grant indicates to the agent that access to the bus has been granted. This is a point to point signal. Every master has its own P_GNT L.

3.3.2.1.12 P_PERR L - Input, TTL

Parity Error is only for the reporting of data parity errors during all PCI transactions except a Special Cycle. The P_PERR L is a sustained tri-state and must be driven active by the agent receiving data two clocks following the data when a data parity error is detected. The minimum duration of P_PERR L is one clock for each data phase that a data parity error is detected. (If sequential data phases each have a data parity error, the P_PERR L signal will be asserted for more than a single clock). P_PERR L must be driven high for one clock before being tri-stated as with all sustained tri-state signals. There are no special conditions when a data parity error may be lost or when reporting of an error may be delayed. An agent cannot report a P_PERR L until it has claimed the access by asserting P_DVSEL L and completed a data phase.

3.3.2.1.13 P_SERR L - Input, TTL

System Error is for reporting address parity errors, data parity errors on the Special Cycle command, or any other system error where the result will be catastrophic. If an agent does not want a non-maskable interrupt (NMI) to be generated, a different reporting mechanism is required. P_SERR L is pure open drain and is actively driven for a single PCI clock by the agent reporting the error. The assertion of P_SERR L is synchronous to the clock and meets the setup and hold times of all bused signals. However, the restoring of P_SERR L to the de-asserted state is accomplished by a weak pullup which is provided by the system designer and not by the signaling agent or central resource. This pullup may take two to three clock periods to fully restore P_SERR L. The agent that reports P_SERR L to the operating system does so anytime P_SERR L is sampled asserted.

3.3.2.1.14 P_INT H - Input, TTL

P_INT H is used by PCI devices to inform the operating system that an interrupting condition has occurred which requires servicing.

3.3.2.1.15 P_RST L - Output, BD4T, 4ma

Reset is used to bring PCI-specific registers, sequencers, and signals to a consistent state. Anytime P_RST L is asserted, all PCI output signals must be driven to their benign state. In general, this means they must be tri-stated. P_SERR L (open drain) is floated. P_REQ L and P_GNT L must both be tri-stated (they can not be driven low or high during reset). To prevent P_AD, P_CBE L and P_PAR signals from floating during reset, the central device may drive these lines during reset (bus parking) but only to a logic low level. They may not be driven high. P_RST L may be asynchronous to P_CLK H when asserted or de-asserted. Although asynchronous, de-assertion is guaranteed to be a clean, bounce-free edge. Except for configuration accesses, only devices that are required to boot the system will respond after reset.

3.3.2.1.16 P_CLK H - Input, TTL, 50% duty

Clock provides timing for all transactions on PCI and is an input to every PCI device. All other PCI signals, except P_RST L are sampled on the rising edge of P_CLK H, and all other timing parameters are defined with respect to this edge. PCI operates up to 33MHz, and in general, the minimum frequency is DC (0Hz).

3.3.2.1.17 P_NMI H - Input, TTL

P_NMI H is asserted by the PCI/EISA Bridge and indicates that a fatal condition has occurred.

3.3.2.1.18 P_LREAD H - Input, TTL

LREAD H is an input into the T2 that provides an additional way for a PCI device to read more than one cache line (8 longwords). The PCI specification states that any PCI device wanting to read more than one cache line (16 longwords) should use either the Memory Read Line or Memory Read Multiple commands (Memory Read command is for one cache line). Some first pass PCI devices, however, were designed using only the Memory Read command, even though they possess the capabilities to read more than one cache line. Therefore, the LREAD H signal was implemented so that these devices could be allowed to read more than one cache line even though their command did not specify it. To operate correctly, LREAD H needs to be asserted during the address phase of the particular PCI devices read transaction. This signal is pulled up on the Motherboard.

3.3.2.2 CBUS Signals

3.3.2.2.1 CAD[127:0] H - Bidirectional, TTL, BD8TRP, 8ma

CAD[127:0] H are multiplexed between address and data information. Starting at cycle 0.5 to the end of cycle 1 CAD[127:0] H represent address, command, and commander identification information. For write cycles 2 and 3 and read cycle 5 and 6 they represent 128 bits of write or read data in each cycle, to complete a 256 bit data transfer. The command and address cycle is used by a commander to initiate a transaction. A commander shall drive the RESERVED fields high to ensure proper parity generation and to reduce switching noise. There are four types of transactions encoded by four command types: Write data, Read data, Exchanged read with write data, and the Null transaction. During write data cycles the commander drives data on CAD[127:0] H twice. The full 256 bits of data are written, since all memory and non-cachable address space write transactions are full hexaword. During Read Data Return and Read Data Error cycles the responder drives the data on CAD[127:0] H. A commander which does not receive acknowledgment to an initiated transaction aborts by completing the protocol in the specified number of transaction cycles. The commander is responsible for reporting this fault to the processor via a hard error interrupt.

3.3.2.2.2 CPAR[3:0] L - Bidirectional, TTL, BD8TRP, 8ma

CPAR[3:0] L is computed over each longword of the 128-bit CAD[127:0] H. Odd parity is used, where the "exclusive OR" of all bits including the parity bit is a "0". Good parity is defined as all bits, including the parity bit, have an odd number of true values. When the bus is idle, the backplane ensures that the bus defaults to a pulled up level. The 33 high levels cause incorrect parity to be guaranteed on an idle bus. If a device is granted the bus but chooses not to use it during a given cycle, it is responsible for driving the CAD[127:0] H to valid levels and CPAR[3:0] L with correct parity.

3.3.2.2.3 CXACK L - Bidirectional, TTL, BD8TRP, 8ma

In response to each address and command cycle, a responder is required to acknowledge it had received the command and address with good parity by asserting CXACK_ L two cycles after cycle 1, regardless of the CSTALL L state. This could be cycle 2 or cycle 3. A bystander must check parity for all address and command cycles, and if an error is detected, log the error and notify a CPU via the C_ERR_L interrupt. Bystanders should not check data parity. Responders must also acknowledge the reception of write data with good parity, by asserting CXACK L two cycles after the write data is received. That is in cycle 4 and cycle 5 in no CSTALL L assertion, or in the multiple cycle 4(s) if CSTALL L is asserted. A responder which accepts writes to non-cachable space registers should check octaword parity and if the first received octaword has a detected data parity error the write shall not be completed to the accessed register. CXACK L is NOT used for flow control. Commanders must check for the CXACK L response and complete the transaction cycle flow. If CXACK L assertion is not detected in the correct cycles, the commander completes the transaction, logs the failing event(s) and posts a C_ERR L interrupt. Address and command cycles to non-decoded address spaces are cycled through the transaction by the commander without acknowledgment.

3.3.2.2.4 BTPHI1 H, TPHI1 H - Input, TTL, 50% Duty

BTPHI1 provides timing for all transactions on CBUS and is an input to every CBUS interface. CBUS operates at 41.67 MHz with a 24ns cycle time.

3.3.2.2.5 TPHI1 L - Input, TTL, 50% Duty

TPHI1 L is an inverted version of TPHI1 H.

3.3.2.2.6 CSHARED L - Output, BD24TRP, 24ma

Based on the shared state of the cache block, the commander determines whether to do a write-through when the contents of the memory block are written. CSHARED L is asserted from cycle 3.5 to 4.5 by a CPU or I/O bystander that contains the referenced valid memory block during any transaction to be sampled at the end of cycle 4.5. CDIRTY L shall not assert during non-cachable address space transactions. CSHARED L could change state in cycles 2 and 5. Drivers must not be enabled in

cycle 0. In addition, a CPU node must respond with shared status for a CBUS read of a cache block which has a valid lock address and lock flag. Any CPU node may assert CSHARED L as the I/O node performs a transaction. During a write transaction, the bystander CPU snoop result may assert CSHARED L to signal that it accepted the write, and the commander must retain the shared state of the block. Alternatively, the bystander CPU must not assert the CSHARED L signal if it invalidates the cache block. Hence, CPU nodes can either accept and update writes or invalidate on writes. An I/O node monitors transaction addresses and returns CSHARED L status for read transactions to a block it has buffered for merging with write data. For write transactions from another node to an I/O node buffered block, the block is invalidated, and read again, to obtain the most recent copy. This enables the I/O node to merge write data to a block which a CPU node may be polling.

3.3.2.2.7 CUCERR L - Input, TTL

Per transaction unrecoverable data errors are reported via the CUCERR L signal. For read transactions CUCERR_L asserts in cycle 5 or 6 with the octaword of bad data. For write transactions CUCERR L asserts in cycle 2 or 3 with the octaword of bad data. Also, CUCERR L is used for flow control of store conditional writes to the non-cachable address space mailbox pointer register and shall assert in the first cycle 4 to be sampled by the commander at the end of the first cycle 4. The I/O responder may sustain the assertion of CUCERR L during multiple cycle 4s, but shall negate CUCERR L no later than the end of the last cycle 4. CUCERR L is sampled by the commander to signify data errors at the end of cycle 5 or 6 for reads, and to signify STxC non-cachable address space write failures at the end of the first cycle 4. The responder samples CUCERR L to check the validity of the written data at the end of cycles 2 or 3 for writes. CUCERR L shall assert with each read data return cycle or write data cycle which has uncorrectable data. The responder node which asserts CUCERR L in cycle 5 shall leave the signal asserted through cycle 6. The commander node which asserts CUCERR L in cycle 2 shall leave the signal asserted through cycle 3. The CPU nodes must know which octaword of data is bad as it is received to signal the ALPHA AXP chip correctly. The memory controller specification is to always assert CUCERR L for uncorrectable errors during read data return cycles 5 and 6.

3.3.2.2.8 CA L - Input, TTL

CA L is driven only by the CBUS arbiter located on the CPU0 node. Nodes receive CA L to identify cycle 0 as the start of a new transaction. The address and command must be driven by the granted commander in cycle 1, following cycle 0. There are cases where the next arbitration grant, which defines the next CA L assertion, overlap the last cycle of a transaction, however tri-state overlap is ensured to be avoided by the transaction timing.

3.3.2.2.9 IOG L - Input, TTL

IOG L asserts on a NOT TPHI1 edge. 0.5 cycle before cycle 0, 1 cycle after it was sampled, to grant the CBUS to the requesting I/O node. The arbiter asserts CA L in cycle 0, and the I/O node shall drive the address in cycle 1. IOG L may negate on the NOT TPHI1 clock after cycle 0, to be sampled 1 cycle later. IOG L is a unidirectional signal driven from CPU0 to the I/O nodes. An I/O node is guaranteed exclusive use of CBUS as long as the IOREQ L is held asserted, otherwise fair arbitration is assumed.

3.3.2.2.10 CRESET L - Input, TTL

This signal returns the system to an initial state when asserted by asynchronously forcing each node to disable its bus drivers. During a power-up, CRESET L remains asserted until backplane +5v and CPU +3.3v power are stable, CBUS oscillators are running at full harmonic content and voltage swing. These conditions are ANDed with the ASYNC_RESET L signal from the power-controller, synchronized to the TPHI1 clock to negate CRESET L synchronous to TPHI1. Nodes shall have 4nS minimum setup time of the negation of CRESET L to their TPHI1 clock edge. CRESET L may assert after the system has powered-up for a minimum of 10mS to force all CBUS nodes to an initial state. CRESET L shall assert 100nS maximum after the assertion of ASYNC_RESET L.

3.3.2.2.11 IOREQ L - Output, BD8T, 8ma

3.3.2.3 IOREQ_L, XIOREQ_L

Integral and expansion I/O nodes assert IOREQ L on any NOT TPHI1 clock edge to request the CBUS. IOREQ L is a unidirectional signal to the CPU0 module. These signals are sampled by a NOT TPHI1 edge 1 cycle before IOG L is asserted on a NOT TPHI1 edge. Once asserted, IOREQ L shall not negate before cycle 0.5 after IOG L has asserted. IOREQ L may be held asserted, and then IOG L could assert in the next arbitration cycle or the second arbitration cycle. An I/O node may retain commander status of the CBUS by leaving the request asserted. The I/O commander may signal the arbiter to search for another node to grant by negating the request for 1 cycle between cycles 0 and 4.

3.3.2.3.1 CIRQ L - Output, BD8T, 8ma

This level sensitive interrupt request signal asserts asynchronously from the I/O node to all CPU nodes. It is a general device level interrupt. System software can decide which CPU node is responsible for servicing this signal.

NOTE

The CIRQ L signal can be shared by CPU nodes, the interrupt controllers have a contention prevention mechanism for passive release.

3.3.2.3.2 CSTALL L - Input, TTL

CBUS transactions can be stalled an integral number of cycles by writing or exchanging commanders to memory address space in cycle 2 or by responders or bystanders in cycle 4 by asserting CSTALL L in cycle 1.5 or cycle 3.5. Commander driven stalls in cycle 2, with CSTALL L driven not later than cycle 1.5, allow a cached node to read data from the cache, merge it with processor write data, and then continue the write to memory. Stall shall not be asserted by a responder or bystander node later than cycle 3.5. A bystanding node shall not stall any non-cachable address space reference. This provides per transaction flow control. The negation in cycle 2.5 or 4.5 enables entry to cycle 3 or 5, 1.5 cycle later to continue a transaction. Arbitration request signals may assert during stalled cycles. There are two copies of this signal, each shall be driven by a single stub per node. Since the bus interface chips are QW sliced, one copy is for the even slice and the other copy is for the odd slice. It is possible for the even and odd CPU interface chips to become 1.5 cycles out of synchronization, if the snoop result differs due to faulty hardware. A recovery procedure that involves having the two slices communicate via an open drain bidirectional voted signal at the end of a transaction allows re-synchronization. Designers discretion is expected concerning permissible numbers of stalled cycles per transaction. However, due to the overall performance and I/O latency impact, stalling beyond 6 cycles, as a general design practice is not permitted. A timer shall not be implemented to enforce this policy.

3.3.2.3.3 C_ERR L - Output, BD24TRP, 24ma

This synchronous signal is asserted in cycle 0 by a CPU node or cycles 0 or 1 by an I/O or memory node of a transaction, or whenever the CBUS is idle, for a maximum of one cycle. The signal is sampled to cause an interrupt to all processor nodes. This interrupt is asynchronous to a CBUS transaction that may have caused an error. This signal indicates a hard or soft error or a latent error to a CPU from a CPU, memory, or an I/O node. A node shall provide a soft error interrupt enable method in its CSRs if it is a source of soft errors. Latent errors are defined as being reported to the CPU nodes after the associated CBUS transaction has completed. During CPU commander transactions to secondary I/O space, the PCI Bridge chip or the expansion I/O interface may assert C_ERR L on the cycle 3 TPHI1 edge for one and only one cycle, and negate C_ERR L on the cycle 4 TPHI1 edge. This signals a retry to the commanding CPU node. A commander implementing this simple retry signaling mechanism does not implement a limit counter. The responder node is obligated to abort the commander should an exception occur.

3.3.2.3.4 MBA<7:0> H - Bidirectional, TTL, BD2T, 2ma

These general purpose lines are used to multiplex out T2 internal state machine state bits for testing purposes. Additionally, these signals can be connected to external logic requiring storage of certain error or status conditions.

3.3.2.3.5 PARAMOUT - Output, B4, 4ma

A signal used for testing.

3.3.2.3.6 TRIST L - Input, TTL

A signal used to tri-state all T2 output or bidirectional signals simultaneously.

3.3.3 T2 Functionality

3.3.3.1 CBUS transactions

The T2 ASIC can operate as both a Master and Slave on the CBUS. It supports the following following CPU initiated transactions:

- **LDx_L to I/O Space** - This is reflected as a CBUS read operation to I/O space. The T2 retries this transaction until the data is read from the target device. Once the data is in the T2 array, all further DMA write traffic on the PCI is retried until the CPU has read the data successfully. This transaction has no effect on the lock flag.
- **STx_C to I/O Space** - Write to I/O space are treated as "dump and run" writes. There is a single quadword write buffer for I/O transactions. Subsequent writes are retried until the PCI write transaction is complete and the buffer is free.
- **Read/Write Clock to Local CSR space** - These cycles are complete in a single CBUS cycle.

3.3.3.2 PCI transactions

The T2 supports the following PCI transactions:

- Acting as a PCI Master:
  - **I/O Read** - This is the PCI transaction done in response to a LDx_L from the processor to Sparse I/O space. The T2 supports byte, word, tribyte, longword and quadword lengths.
  - **I/O Write** - This is the PCI transaction done in response to a STx_C from the processor to Sparse I/O space. The T2 supports byte, word, tribyte, longword and quadword lengths.
  - **Configuration Reads and Writes** - This is the PCI transaction done in response to a LDx_L or STx_C from the processor to Configuration Space. The T2 supports byte, word, tribyte, longword and quadword lengths. Both type 0 and type 1 are supported.
  - **Memory Read** - This is a PCI transaction done in response to LDx from the processor to either Sparse Memory space or Dense Memory space. In Sparse Memory space the T2 supports byte, word, tribyte, longword and quadword lengths. In Dense Memory Space the T2 supports quadword only.

— **Memory Write** - This is a PCI transaction done in response to STx from the processor to either Sparse Memory space or Dense Memory space. In Sparse Memory space the T2 supports byte, word, tribyte, longword and quadword lengths. In Dense Memory Space the T2 supports lengths ranging from longword to hexaword in longword increments.

— **Special Cycle** - Special cycles are initiated by a CPU write to a local T2 CSR. Refer to the Addressing Chapter, Chapter 5, and the PCI specification for more details on Special Cycles.

- Acting as a PCI Target:

— **Memory Read DMA** - The T2 contains a hexaword DMA read buffer with an enabled hexaword read prefetch buffer to support byte to burst length PCI transfers into system memory from some PCI device. When the read prefetch buffer is disabled, the maximum burst transfer size accepted by the T2 is a single hexaword. When the read prefetch buffer is enabled, there is no maximum burst transfer size restriction, although it is expected that each PCI Master use a fair time slice of the PCI as to not degrade overall system performance. There are two types of DMA transactions supported by the T2, direct mapped and scatter/gather mapped.

— **Memory Write DMA** - The T2 contains two hexaword DMA write buffers with two modes of operation to support byte to burst length PCI transfers into system memory from some PCI device. When the write buffers are in single DMA mode, each buffer functions individually and allows a maximum burst transfer size of a single hexaword. When the write buffers are in multiple DMA mode, both buffers can be used for a single DMA transaction. There is no maximum burst transfer size restriction in multiple DMA mode, although it is expected that each PCI Master use a fair time slice of the PCI as to not degrade overall system performance. There are two types of DMA transactions supported by the T2, direct mapped and scatter/gather mapped.

3.3.3.3 Translation Look-aside Buffer (TLB)

To improve performance during scatter/gather mapped DMA transactions, the T2 contains an eight entry TLB. The TLB allows the T2 to cache up to eight Page Frame Numbers (PFN's) from a page table stored in system memory. If the TLB were disabled, each scatter/gather DMA transaction would require the T2 to read the associated PFN directly from system memory. The TLB can be enabled from the T2's IOCSR. Replacement of PFN's in the TLB is performed in a round robin manner. Their are provisions built into the TLB logic that allow all resident entries to be invalidated, if necessary. However, the TLB also keeps track of any PFN modified in the System memory resident page table by a CPU and will invalidate entries on an individual basis if a page table PFN it contains is modified. The contents of the TLB is readable through T2 CSR registers.

3.4 Standard I/O Module

See Chapter 8 for a complete description of the Standard I/O module.

CHAPTER 4

CONTROL STATUS REGISTERS

Several types of CSRs exist in the Sable I/O subsystem. The first type are the CBUS CSRs. These are only accessible from the CBUS (System bus) and contain some required system registers. The only CBUS CSRs in the Sable I/O subsystem are found in the T2 gate array.

The second type of CSRs are found in PCI configuration space. They are accessed by the system through a certain address range and converted by the T2 gate array to Configuration cycles on the PCI. Configuration space is intended for configuration, initialization and catastrophic error handling functions. Its use should be restricted to initialization software and error handling software. All operational software must use I/O and/or memory space accesses to manipulate device registers.

The third type of CSRs are the device specific CSRs. These are found in PCI0 memory or I/O space. The address tables for the TULIP, NCR 53C810 and ESC/PCEB registers are found in this chapter. Descriptions of these CSRs can be found in their individual specifications.

4.1 CBUS CSRS

All the CBUS CSRs in the Sable I/O subsystem are in the T2 gate array. The CBUS CSRs in the T2 are all aligned on hexaword boundaries and have quadword length. The T2 contains the three standard CBUS registers, CSR0, CSR1 and CSR2. The contents of CSR1 and CSR2 are only valid when an error bit is set in CSR0.

4.1.1 T2 CBUS registers

Table 1: T2 Registers

Register Name	Mnemonic	EV Address	Length	Attributes
I/O Control/Status Register	IOCSR	3.8E00.0000	8 bytes	R/W
CBUS Error Register 1	CERR1	3.8E00.0020	8 bytes	R/W
CBUS Error Register 2	CERR2	3.8E00.0040	8 bytes	RO
CBUS Error Register 3	CERR3	3.8E00.0060	8 bytes	RO
PCI Error Register 1	PERR1	3.8E00.0080	8 bytes	R/W
PCI Error Register 2	PERR2	3.8E00.00A0	8 bytes	RO
PCI Special Cycle Register	PSCR	3.8E00.00C0	8 bytes	Write Only[1]
High Address Extension Register 1	HAE0_1	3.8E00.00E0	8 bytes	R/W
High Address Extension Register 2	HAE0_2	3.8E00.0100	8 bytes	R/W
PCI Hole Base Register	HBASE	3.8E00.0120	8 bytes	R/W
Window Base Register 1	WBASE1	3.8E00.0140	8 bytes	R/W
Window Mask Register 1	WMASK1	3.8E00.0160	8 bytes	R/W
Translated Base Register 1	TBASE1	3.8E00.0180	8 bytes	R/W
Window Base Register 2	WBASE2	3.8E00.01A0	8 bytes	R/W
Window Mask Register 2	WMASK2	3.8E00.01C0	8 bytes	R/W
Translated Base Register 2	TBASE2	3.8E00.01E0	8 bytes	R/W
TLB By-Pass Register	TLBBR	3.8E00.0200	8 bytes	R/W
IVR Passive Release Register	IVRPR	3.8E00.0220	4 lo bytes	R/W
IVR Interrupt Address Register	IVIAR	3.8E00.0220	4 hi bytes	R/W[2]
High Address Extension Register 3	HAE0_3	3.8E00.0240	8 bytes	R/W[3]
High Address Extension Register 4	HAE0_4	3.8E00.0260	8 bytes	R/W[3]
TLB Data Register 0	TDR0	3.8E00.0300	8 bytes	RO
TLB Data Register 1	TDR1	3.8E00.0320	8 bytes	RO
TLB Data Register 2	TDR2	3.8E00.0340	8 bytes	RO
TLB Data Register 3	TDR3	3.8E00.0360	8 bytes	RO
TLB Data Register 4	TDR4	3.8E00.0380	8 bytes	RO
TLB Data Register 5	TDR5	3.8E00.03A0	8 bytes	RO
TLB Data Register 6	TDR6	3.8E00.03C0	8 bytes	RO
TLB Data Register 7	TDR7	3.8E00.03E0	8 bytes	RO

[1] Read as 0.

[2] Available in Pass 2 T2's only. This register contains the I/O address of the interrupt vector register. Suggested EV address is 3.A000.A600 for a PCI address of 530.

[3] Available in Pass 2 T2's only.

Figure 7: I/O Control/Status Register Low (IOCSRL)

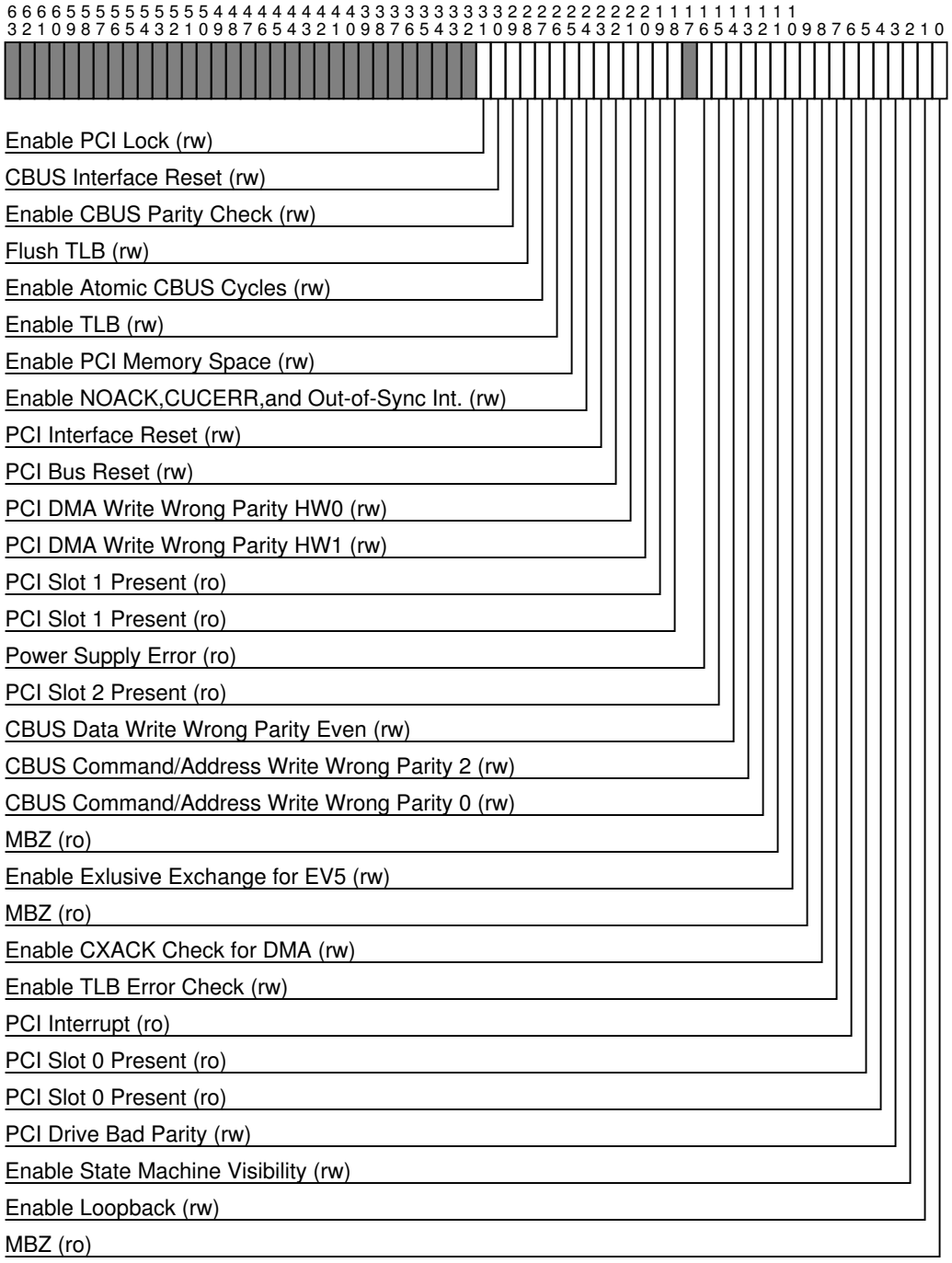


Table 2: I/O Control/Status Register Low Description

Field	Description
0	**MBZ** [iocsrl_m_mbz , read-only] This field must be zero.
1	**Enable Loopback** [iocsrl_m_el , read/write] When this bit is set the T2 is in internal loopback mode. This is a diagnostic mode which allows PCI I/O space Reads and Writes to be looped back from/to system memory. For a more detailed explanation of this diagnostic mode please refer to the diagnostic chapter, Chapter 6. This field is cleared on power-up or after a System bus reset.
2	**Enable State Machine Visibility** [iocsrl_m_esmv , read/write] When this bit is set the state bits of the T2 internal state machine selected by IOCSR<38:36> are driven onto the MBA<7:0> pins of the T2 gate array. This field is cleared on power-up or after a System bus reset.
3	**PCI Drive Bad Parity** [iocsrl_m_pdbp , read/write] Setting this bit forces the T2 to drive bad parity during any one of the following PCI bus cycle phases: Address phase, PCI Read or Write cycle, T2 as master Data phase, PCI Write cycle, T2 as master Data phase, PCI Read cycle, T2 as slave When the parity error occurs PERR1<3> will be set and, if IOCSR<60> is set, and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
4	**PCI Slot 0 Present** [iocsrl_m_pci_slot0_1 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<0>. The PCI slot 0 PRSNT1# is connected to MBA<0>. If either IOCSR<5:4> is set, a PCI option is plugged into PCI slot 0. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
5	**PCI Slot 0 Present** [iocsrl_m_pci_slot0_2 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<1>. The PCI slot 0 PRSNT2# is connected to MBA<1>. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
6	**PCI Interrupt** [iocsrl_m_pint , read-only] When this bit is set, it indicates that some device on the PCI is interrupting. Specifically, the INT line is asserted from the Standard I/O module.
7	**Enable TLB Error Check** [iocsrl_m_entlbec , read/write] When this bit is set, error reporting for the TLB is turned on. For example,an error will be reported. If this bit is not set, no error will be reported.
8	**Enable CXACK Check for DMA** [iocsrl_m_enccdma , read/write] When this bit is set, checking for CXACKS on the CBUS is enabled. If this bit is not set, no error will be reported if a CXACK does not occur properly.
9	**MBZ** [iocsrl_m_mbz , read-only] This field must be zero.
10	**Enable Exlusive Exchange for EV5** [iocsrl_m_enxxchg , read/write] When this bit is set, the exclusive exchange command is enabled for EV5 type CPU systems.
11	**MBZ** [iocsrl_m_mbz , read-only] This field must be zero.
12	**CBUS Command/Address Write Wrong Parity 0** [iocsrl_m_cawwp0 , read/write]

Table 2 (Cont.): I/O Control/Status Register Low Description

Field	Description
	When this bit is set, wrong parity is generated for longword 0 during the next CBUS command/address transfer phase when the T2 gate array is the commander. This bit is active for only one CBUS command/address transfer. This bit is cleared on power-up or after a System bus reset.
13	**CBUS Command/Address Write Wrong Parity 2** [iocsrl_m_cawwp2 , read/write] When this bit is set wrong parity is generated for longword 2 during the next CBUS command/address transfer phase when the T2 gate array is the commander. This bit is active for only one CBUS command/address transfer. This bit is cleared on power-up or after a System bus reset.
14	**CBUS Data Write Wrong Parity Even** [iocsrl_m_dwwpe , read/write] When this bit is set, wrong data parity is generated by the T2 gate array all even longwords during the next read of any T2 register. This bit is cleared on power-up or after a System bus reset.
15	**PCI Slot 2 Present** [iocsrl_m_pci_slot2_2 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<5>. The PCI slot 2 PRSNT2# is connected to MBA<5>. If either IOCSR<39> or IOCSR<15> is set, a PCI option is plugged into PCI slot 2. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
16	**Power Supply Error** [iocsrl_m_pserr , read-only] When this bit is clear this is inverted, a '0' is being asserted on the input pin of the MBA<6>. Power Supply Error is connected to MBA<6>. If there are two power supplies in redundant mode and one goes away, this bit will set. Reading the I^2C bus will indicate which power supply has gone away. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
17	**External MBA<7>** [iocsrl_m_mba7 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<7>.
18	**PCI Slot 1 Present** [iocsrl_m_pci_slot1_1 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<2>. The PCI slot 1 PRSNT1# is connected to MBA<2>. If either IOCSR<19:18> is set, a PCI option is plugged into PCI slot 1. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
19	**PCI Slot 1 Present** [iocsrl_m_pci_slot1_2 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<3>. The PCI slot 1 PRSNT2# is connected to MBA<3>. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
20	**PCI DMA Write Wrong Parity HW1** [iocsrl_m_pdwwp1 , read/write] When this bit is set, any subsequent DMA write operations will result in a CBUS parity error on the second hexaword of data. This bit is cleared on power-up or after a System bus reset.
21	**PCI DMA Write Wrong Parity HW0** [iocsrl_m_pdwwp0 , read/write] When this bit is set, any subsequent DMA write operations will result in a CBUS parity error on the first hexaword of data. This bit is cleared on power-up or after a System bus reset.
22	**PCI Bus Reset** [iocsrl_m_pbr , read/write] Setting this bit will assert the PCI Bus Reset signal. This bit needs to be set and cleared to toggle the PCI Bus Reset signal. This bit is cleared on power-up or after a System bus reset.
23	**PCI Interface Reset** [iocsrl_m_pir , read/write] Setting this bit will reset the PCI interface logic in the gate array. The PCI Bus Reset signal is *not asserted* when this bit is set. This bit is cleared on power-up or after a System bus reset.
24	**Enable NOACK,CUCERR,and Out-of-Sync Int.** [iocsrl_m_encoi , read/write]

Table 2 (Cont.): I/O Control/Status Register Low Description

Field	Description
	When this bit is set and any one of the following error conditions occurs:. Uncorrectable Read Error (CERR1<0>) No Acknowledge Error (CERR1<1>) Commander Write Data Parity Error (CERR1<16>) Bus Synchronization Error (CERR1<17>) the corresponding CERR1 status bit is set, the failing command/address is saved in CERR2 and CERR3 and the C_ERR hardware error interrupt will be asserted on the CBUS. This bit is cleared on power-up or after a System bus reset.
25	**Enable PCI Memory Space** [iocsrl_m_epms , read/write] Setting this bit will cause the T2 gate array to respond to CBUS addresses 2.0000.0000 (Hex) to 2.FFFF.FFFF (Hex). This bit is cleared on power-up or after a System bus reset to allow Memory Module CSR space to exist in this address range at system initalization time.
26	**Enable TLB** [iocsrl_m_etlb , read/write] Setting this bit will cause the Translation Look-aside Buffer to be enabled. This bit is cleared on power-up or after a System bus reset.
27	**Enable Atomic CBUS Cycles** [iocsrl_m_eacc , read/write] Setting this bit enables the T2 to perform two or more related CBUS transactions without intervening CBUS transactions from other CBUS devices. This bit is cleared on power-up or after a System bus reset. (Also known as hog-mode, exclusive access mode and haste mode.)
28	**Flush TLB** [iocsrl_m_ftlb , read/write] Setting this bit invalidates all eight entries within the TLB and returns the TLB update pointer to the first TLB entry. This bit is cleared on power-up or after a System bus reset.
29	**Enable CBUS Parity Check** [iocsrl_m_ecpc , read/write] Setting this bit will cause the CBUS parity generation and check logic to be enabled. This bit is cleared on power-up or after a System bus reset.
30	**CBUS Interface Reset** [iocsrl_m_cir , read/write] Setting this bit will reset the CBUS interface logic in the gate array. The CBUS Bus Reset signal is *not asserted* when this bit is set. This bit is cleared on power-up or after a System bus reset.
31	**Enable PCI Lock** [iocsrl_m_epl , read/write] Setting this bit will cause CBUS "hog mode" during PCI locks to be enabled. In this way the T2 can "lock" system memory during PCI lock cycles by gaining exclusive control of the CBUS. This bit is cleared on power-up or after a System bus reset.

Figure 8: I/O Control/Status Register High (IOCSRH)

Table 3: I/O Control/Status Register High Description

Field	Description
32	**CBUS Back-To-Back Cycle Enable** [iocsrh_m_cbbce , read/write] Setting this bit will cause T2 CBUS back-to-back cycles to be enabled. In this way the T2 can make most efficient use of CBUS bandwidth by performing back-to-back CBUS cycles where possible. This bit is cleared on power-up or after a System bus reset.
35:33	**T2 Revision Number** [iocsrh_m_trn , read-only] This field contains the T2 revision number. Writes to this field complete without error but have no effect on the data contents. Pass 1 = 000 Pass 2 = 001

Table 3 (Cont.): I/O Control/Status Register High Description

Field	Description
38:36	**State Machine Visibility Select** [iocsrh_m_smvl , read/write] This field controls which internal state machine's state bits will be muxed onto the MBA<7:0> pins of the gate array. The bit encoding is as follows: 000 - CBUS Cycle Counter 001 - CBUS Responder 010 - CBUS Commander 011 - PCI Commander 100 - PCI Responder 101 - TLB Invalidate 110 - PCI Corner 111 - CBUS Corner
39	**PCI Slot 2 Present** [iocsrh_m_pci_slot1_1 , read-only] When this bit is set, a '0' is being asserted on the input pin of the MBA<4>. The PCI slot 2 PRSNT2# is connected to MBA<4>. If IOCSR<2> (Enable State Machine Visibility) is set, this bit has no meaning.
40	**Enable Passive Release** [iocsrh_m_epr , read/write] When this bit is set and a PCI interrupt vector read from a CPU is pending, T2 will respond to subsequent PCI interrupt vector read attempts from other CPUs with a programmed passive release interrupt vector found in the IVRPR register. When this bit is clear, T2 will respond to subsequent PCI interrupt vector read attempts from other CPUs with a retry request. This bit is cleared on power-up or after a System bus reset.
43:41	**MBZ** [iocsrh_m_mbz , read-only] This field must be zero.
44	**CBUS Command/Address Write Wrong Parity 1** [iocsrh_m_cawwp1 , read/write] When this bit is set wrong parity is generated for longword 1 during the next CBUS command/address transfer phase when the T2 gate array is the commander. This bit is active for only one CBUS command/address transfer. This bit is cleared on power-up or after a System bus reset.
45	**CBUS Command/Address Write Wrong Parity 3** [iocsrh_m_cawwp3 , read/write] When this bit is set wrong parity is generated for longword 3 during the next CBUS command/address transfer phase when the T2 gate array is the commander. This bit is active for only one CBUS command/address transfer. This bit is cleared on power-up or after a System bus reset.
46	**CBUS Data Write Wrong Parity Odd** [iocsrh_m_dwwpo , read/write] When this bit is set, wrong data parity is generated by the T2 gate array for all odd longwords during the next read of any T2 register. This bit is cleared on power-up or after a System bus reset.
51:47	**MBZ** [iocsrh_m_mbz , read-only] This field must be zero.
52	**PCI Read Multiple** [iocsrh_m_prm , read/write] Setting this bit enables the T2 PCI interface to service PCI read cycles which span hexaword boundries by chaining hexaword read buffers. This function is transparent to the PCI master and can be sustained for an unlimited number of bytes of data. When this bit is clear and a PCI read cycle spans hexaword boundries, the T2 PCI interface must assert the STOP# signal on the PCI bus at hexaword boundries, forcing the PCI master to get off the bus and re-request with a new PCI address to complete the DMA transaction. This field is cleared on power-up or after a System bus reset.

Table 3 (Cont.): I/O Control/Status Register High Description

Field	Description
53	**PCI Write Multiple** [iocsrh_m_pwm , read/write] Setting this bit enables the T2 PCI interface to service PCI write cycles which span hexaword boundries by chaining hexaword write buffers. This function is transparent to the PCI master and can be sustained for an unlimited number of bytes of data. When this bit is clear and a PCI write cycle spans hexaword boundries, the T2 PCI interface must assert the STOP# signal on the PCI bus at hexaword boundries, forcing the PCI master to get off the bus and re-request with a new PCI address to complete the DMA transaction. This field is cleared on power-up or after a System bus reset.
54	**Force PCI RDPE Detect** [iocsrh_m_fprdped , read/write] Setting this bit forces parity errors to be detected on read data supplied to T2 during PCI read cycles where T2 is acting as a PCI master. When the parity error occurs and IOCSR<61> is set, PERR1<3> will be set, and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
55	**Force PCI APE Detect** [iocsrh_m_fpadped , read/write] Setting this bit forces parity errors to be detected on PCI addresses during cycles where T2 is acting as a PCI slave. When the parity error occurs and IOCSR<62> is set, PERR1<1> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
56	**Force PCI WDPE Detect** [iocsrh_m_fpwdped , read/write] Setting this bit forces parity errors to be detected on write data during PCI write cycles where T2 is acting as a PCI slave. When the parity error occurs and IOCSR<63> is set, PERR1<0> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
57	**Enable PCI NMI** [iocsrh_m_epnmi , read/write] When this bit is set and the PCI signal NMI is asserted, PERR1<6> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
58	**Enable PCI DTI** [iocsrh_m_epdti , read/write] When this bit is set and T2 master PCI cycle ends in a PCI device timeout, PERR1<5> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
59	**Enable PCI SERR Int.** [iocsrh_m_epsei , read/write] When this bit is set and the PCI signal SERR# is asserted (i.e. address parity error or Special Cycle command parity error; T2 as PCI master), PERR1<4> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
60	**Enable PCI PERR Int.** [iocsrh_m_eppei , read/write] When this bit is set and the PCI signal PERR# is asserted during a cycle where T2 is a participant, PERR1<3> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
61	**Enable PCI RDP Int.** [iocsrh_m_erdpc , read/write] When this bit is set and a parity error is detected on read data during a PCI read cycle where T2 was acting as a PCI master, PERR1<2> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
62	**Enable PCI AP Int.** [iocsrh_m_eadpc , read/write]

Table 3 (Cont.): I/O Control/Status Register High Description

Field	Description
	When this bit is set and a parity error is detected on a PCI address during a PCI cycle where T2 was acting as a PCI slave, PERR1<1> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.
63	**Enable PCI WDP Int.** [iocsrh_m_ewdpc , read/write] When this bit is set and a parity error is detected on write data during a PCI write cycle where T2 was acting as a PCI slave, PERR1<0> will be set and the C_ERR hardware error interrupt will be asserted on the CBUS. This field is cleared on power-up or after a System bus reset.

Figure 9: CBUS Error Register 1 (CERR1)

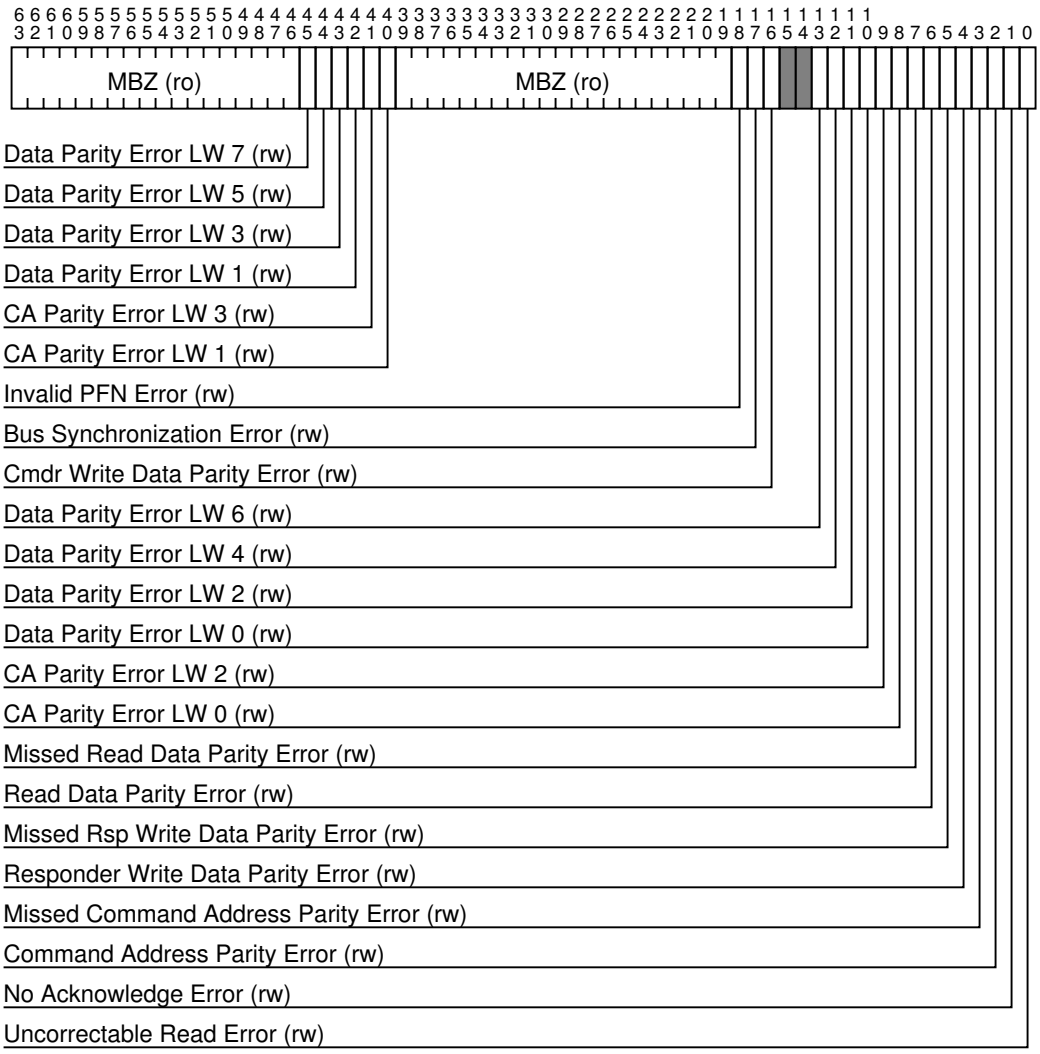


Table 4: CBUS Error Register 1 Description

Field	Description
0	**Uncorrectable Read Error** [cerr1_m_ure , read/write] This bit is set when, while acting as the CBUS master, the T2 gate array receives an uncorrectable read data error from the selected device. The failing command/address is saved in CERR2 and CERR3. Write one to clear. Cleared at power up or after a System bus reset.
1	**No Acknowledge Error** [cerr1_m_nae , read/write] This bit is set when, while acting as the CBUS master, the T2 gate array fails to receive an acknowledge for the command/address that it has placed on the bus. This error indicates that the selected device does not exist, the selected device is broken, the T2 gate array is broken, or a command/address parity error has occurred during the cycle. The failing command/address is saved in CERR2 and CERR3 and the C_ERR hardware error interrupt will be asserted on the CBUS. Write one to clear. Cleared at power up or after a System bus reset.
2	**Command Address Parity Error** [cerr1_m_cape , read/write] This bit is set when IOCSR<29> is set and a parity error is detected on any longword of a CBUS Command/Address transfer regardless of whether the transfer is directed to this module or not. When this bit is set the C_ERR hardware error interrupt will be asserted on the CBUS. The error status in the CERR2 and CERR3 registers and in CERR1 [9:8] is available to help locate the source of the error. Write one to clear. Cleared on power-up.
3	**Missed Command Address Parity Error** [cerr1_m_mcape , read/write] This bit is set when a longword parity error is detected during the transfer of Command/Address from a commander node to a selected responder node *after* some other error has been detected by the error logic. This indicates that the full error status (for example, the Failing Command/Address, and Data Parity Error bits) is not available for this error and, therefore, the error has been "missed". This bit is set by the T2 gate array when a parity error on any longword of the Command/Address transfer occurs, regardless of whether T2 is the intended responder. Write one to clear. Cleared on power-up.
4	**Responder Write Data Parity Error** [cerr1_m_rwdpe , read/write] This bit is set when IOCSR<29> is set and a longword parity error is detected during the transfer of write data from a commander node on the CBUS to a T2 register (i.e. T2 is the CBUS responder). If this indicator is set the C_ERR hardware error interrupt will be asserted on the CBUS. The full error status (including the Failing Command/Address, and Data Parity Error bits) is available to help identify the source of this error. Write one to clear. Cleared on power-up.
5	**Missed Rsp Write Data Parity Error** [cerr1_m_mrwdpe , read/write] This bit is set when a longword parity error is detected during the transfer of write data from a commander node to a T2 register (i.e. T2 is the CBUS responder) *after* some other error has been detected by the error logic. This indicates that the full error status (for example, the Failing Command/Address, and Data Parity Error bits) is not available for this error (and therefore, the error has been "missed"). Write one to clear. Cleared on power-up.
6	**Read Data Parity Error** [cerr1_m_rdpe , read/write] This bit is set when IOCSR<29> is set and a longword parity error is detected during the transfer of read data from a responder node when the T2 gate array is the commander. If this indicator is set the C_ERR hardware error interrupt will be asserted on the CBUS. The full error status (including the Failing Command/Address, and Data Parity Error bits) is available to help identify the source of this error. Write one to clear. Cleared on power-up.

Table 4 (Cont.): CBUS Error Register 1 Description

Field	Description
7	**Missed Read Data Parity Error** [cerr1_m_mrdpe , read/write] This bit is set when a longword parity error is detected during the transfer of read data from a responder node when the T2 gate array is the commander *after* some other error has been detected by the error logic. This indicates that the full error status (for example, the Failing Command/Address, and Data Parity Error bits) is not available for this error (and therefore, the error has been "missed"). Write one to clear. Cleared on power-up.
8	**CA Parity Error LW 0** [cerr1_m_cape0 , read/write] This bit is set when a parity error on longword 0 of a CBUS command/address transfer is detected. Please reference the C-bus specification for the mapping of the command/address field to the CBUS longwords Write one to clear. Undefined on power-up.
9	**CA Parity Error LW 2** [cerr1_m_cape2 , read/write] This bit is set when a parity error on longword 2 of a CBUS command/address transfer is detected. Please reference the Cobra Bus specification for the mapping of the commond/address field to the CBUS longwords Write one to clear. Undefined on power-up.
10	**Data Parity Error LW 0** [cerr1_m_dpe0 , read/write] This bit is set when a parity error is detected on longword 0 during the data portion of a CBUS transaction. This longword corresponds to bytes 3..0 of the hexaword transferred. Write one to clear. Undefined on power-up.
11	**Data Parity Error LW 2** [cerr1_m_dpe2 , read/write] This bit is set when a parity error is detected on longword 2 during the data portion of a CBUS transaction. This longword corresponds to bytes 11..8 of the hexaword transferred. Write one to clear. Undefined on power-up.
12	**Data Parity Error LW 4** [cerr1_m_dpe4 , read/write] This bit is set when a parity error is detected on longword 4 during the data portion of a CBUS transaction. This longword corresponds to bytes 19..16 of the hexaword transferred. Write one to clear. Undefined on power-up.
13	**Data Parity Error LW 6** [cerr1_m_dpe6 , read/write] This bit is set when a parity error is detected on longword 6 during the data portion of a CBUS transaction. This longword corresponds to bytes 27..24 of the hexaword transferred. Write one to clear. Undefined on power-up.
16	**Cmdr Write Data Parity Error** [cerr1_m_cwdp , read/write] This bit is set when a longword write data parity error is detected while the T2 gate array is acting as the CBUS commander. When this bit is set the failing command/address is saved in CERR2 and CERR3 and the C_ERR hardware error interrupt will be asserted on the CBUS. Write one to clear. Cleared on power up or after a CBUS reset.
17	**Bus Synchronization Error** [cerr1_m_bse , read/write]

Table 4 (Cont.): CBUS Error Register 1 Description

Field	Description
	This bit is set when the T2 gate array detects command fields on the upper and lower quadwords of the CBUS that do not match. This error is an indication that the gate array devices that make up the CBUS interface for the current bus master are out-of-sync. This error condition is not checked for during CBUS transactions where the T2 gate array is bus master since its CBUS interface is contained within a single gate array. When this bit is set the failing command/address is saved in CERR2 and CERR3 and the C_ERR hardware error interrupt will be asserted on the CBUS. Write one to clear. Undefined on power-up.
18	**Invalid PFN Error** [cerr1_m_ipfne , read/write] This bit is set when, during a scatter-gather map fetch, the T2 gate array determines that the PFN contained in the returned data is not valid (i.e. the valid bit is not set). When this bit is set the failing command/address is saved in CERR2 and CERR3. Write one to clear. Cleared on power up or after a System bus reset.
39:19	**MBZ** [cerr1_m_mbz , read-only] This field must be zero.
40	**CA Parity Error LW 1** [cerr1_m_cape1 , read/write] This bit is set when a parity error on longword 1 of a CBUS command/address transfer is detected. Please reference the CBUS specification for the mapping of the command/address field to the CBUS longwords Write one to clear. Undefined on power-up.
41	**CA Parity Error LW 3** [cerr1_m_cape3 , read/write] This bit is set when a parity error on longword 3 of a CBUS command/address transfer is detected. Please reference the Cobra Bus specification for the mapping of the commond/address field to the CBUS longwords Write one to clear. Undefined on power-up.
42	**Data Parity Error LW 1** [cerr1_m_dpe1 , read/write] This bit is set when a parity error is detected on longword 1 during the data portion of a CBUS transaction. This longword corresponds to bytes 7..4 of the hexaword transferred. Write one to clear. Undefined on power-up.
43	**Data Parity Error LW 3** [cerr1_m_dpe3 , read/write] This bit is set when a parity error is detected on longword 3 during the data portion of a CBUS transaction. This longword corresponds to bytes 15..12 of the hexaword transferred. Write one to clear. Undefined on power-up.
44	**Data Parity Error LW 5** [cerr1_m_dpe5 , read/write] This bit is set when a parity error is detected on longword 5 during the data portion of a CBUS transaction. This longword corresponds to bytes 23..20 of the hexaword transferred. Write one to clear. Undefined on power-up.
45	**Data Parity Error LW 7** [cerr1_m_dpe7 , read/write] This bit is set when a parity error is detected on longword 7 during the data portion of a CBUS transaction. This longword corresponds to bytes 32..28 of the hexaword transferred. Write one to clear. Undefined on power-up.
63:46	**MBZ** [cerr1_m_mbz , read-only] This field must be zero.

Figure 10: CBUS Error Register 2 (CERR2)

Table 5: CBUS Error Register 2 Description

Field	Description
63:0	**Command/Address[63:00]** [cerr2_m_cal , read-only] These bits correspond to CBUS CAD [63:00] during the command/address transfer of the failing cycle. Note: A write to this register will not error, but will not affect its contents.

Figure 11: CBUS Error Register 3 (CERR3)

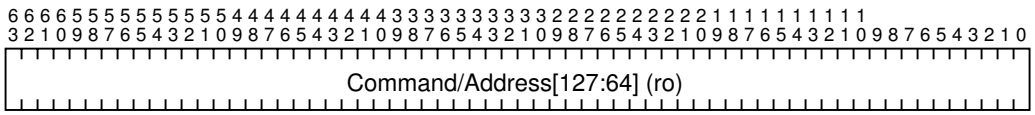


Table 6: CBUS Error Register 3 Description

Field	Description
63:0	**Command/Address[127:64]** [cerr3_m_cal , read-only] These bits correspond to CBUS CAD [127:64] during the command/address transfer of the failing cycle. Note: A write to this register will not error, but will not affect its contents.

Figure 12: PCI Error Register 1 (PERR1)

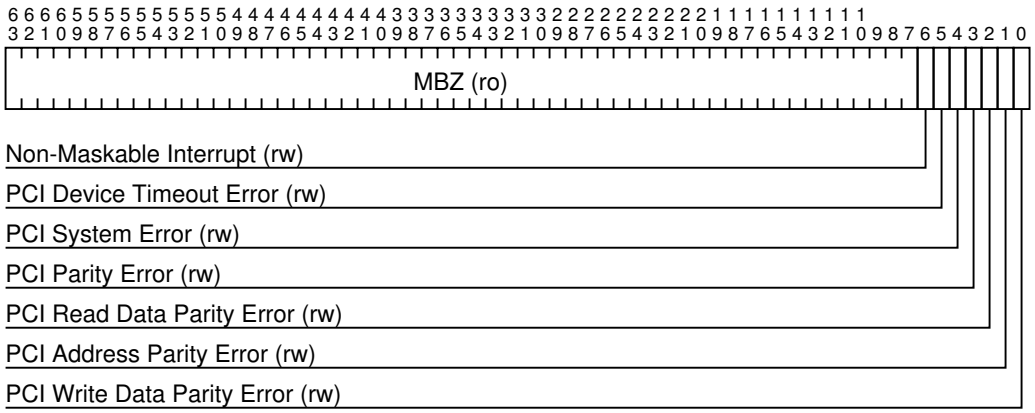


Table 7: PCI Error Register 1 Description

Field	Description
0	**PCI Write Data Parity Error** [perr1_m_pwdpe , read/write] When this bit is set it indicates that T2 detected a write data parity error during a PCI write cycle where T2 was acting as a PCI slave. This bit is enabled by IOCSR<63> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS. Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
1	**PCI Address Parity Error** [perr1_m_pape , read/write] When this bit is set it indicates that T2 detected an address parity error during a PCI cycle where T2 was acting as a PCI slave. This bit is enabled by IOCSR<62> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS. Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
2	**PCI Read Data Parity Error** [perr1_m_prdpe , read/write] When this bit is set it indicates that T2 detected a read data parity error during a PCI read cycle where T2 was acting as a PCI master. This bit is enabled by IOCSR<61> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS. Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
3	**PCI Parity Error** [perr1_m_ppe , read/write] When this bit is set it indicates that the PCI signal PERR# was asserted during a PCI cycle in which T2 was a participant. PERR# can be asserted by external logic to indicate a write data parity error (T2 as PCI master) or a read data parity error (T2 as a PCI slave). This bit is enabled by IOCSR<60> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS. Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
4	**PCI System Error** [perr1_m_pse , read/write] When this bit is set it indicates that the PCI signal SERR# was asserted during a PCI cycle where T2 was acting as a PCI master. SERR# is asserted by external logic to indicate an address parity error (T2 as master), a Special Cycle command parity error (T2 as master), or other fatal error. This bit is enabled by IOCSR<59> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS.Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
5	**PCI Device Timeout Error** [perr1_m_pdte , read/write]

Table 7 (Cont.): PCI Error Register 1 Description

Field	Description
	When this bit is set it indicates that a device timeout occurred during a PCI cycle where T2 was acting as a PCI master. This bit is enabled by IOCSR<58> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS. Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
6	**Non-Maskable Interrupt** [perr1_m_nmi , read/write] When this bit is set it indicates that the NMI signal was asserted. This bit is enabled by IOCSR<57> and , when set, causes the C_ERR hardware error interrupt to be asserted on the CBUS. Write one to clear, cleared on power up or by setting the PCI Reset bits in the IOCSR register.
63:7	**MBZ** [perr1_m_mbz , read-only] This field must be zero.

Figure 13: PCI Error Register 2 (PERR2)

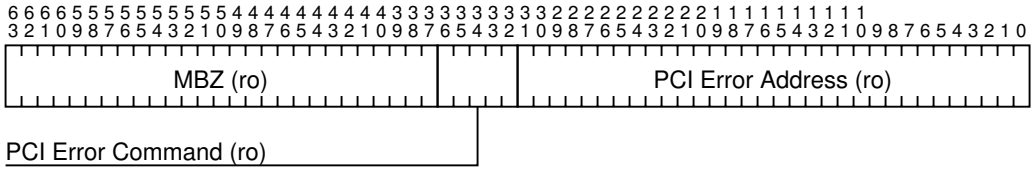


Table 8: PCI Error Register 2 Description

Field	Description
31:0	**PCI Error Address** [perr2_m_pea , read-only] If a parity error occurs during a PCI cycle in which the T2 is a participant this field will contain a latched version of the failing PCI address.
36:32	**PCI Error Command** [perr2_m_pec , read-only] If a parity error occurs during a PCI cycle in which the T2 is a participant this field will contain a latched version of the failing PCI command. Note: This register is not cleared by any reset or power up condition but is constantly loaded with a new address at the beginning of every PCI cycle. The contents of the register are frozen by a detected parity error. The loading of addresses resumes when the corresponding error bit in PERR1 is cleared. A write to this register will not error, but will not affect its contents.
63:37	**MBZ** [perr2_m_mbz , read-only] This field must be zero.

Figure 14: PCI Special Cycle Register (PSCR)

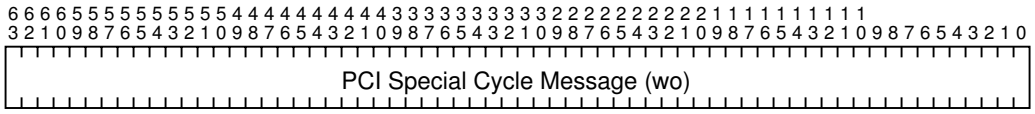


Table 9: PCI Special Cycle Register Description

Field	Description
63:0	**PCI Special Cycle Message** [pscr_m_pscm , write-only] Writing to this register will cause the T2 PCI interface to perform a single PCI Special Cycle using the data written to this register as the message data for that cycle. For a more complete description of PCI Special Cycles, please refer to the PCI Specification, section 3.6.2, Message Passing: Special Cycle Command. Note: A read of this register will not error, but will return zeros.

Figure 15: High Address Extension Register 1 (HAE0_1)

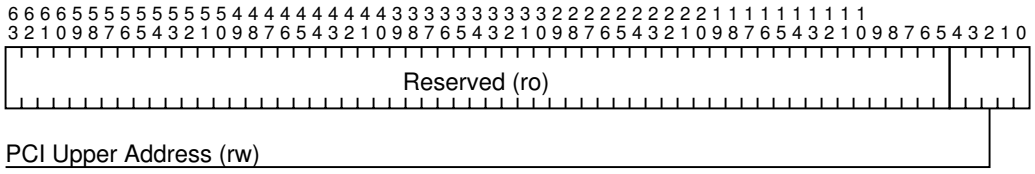


Table 10: High Address Extension Register 1 Description

Field	Description
4:0	**PCI Upper Address** [hae0_1_m_pua1 , read/write] These bits are used as PCI_AD<31:27> in accessing the 128MB of PCI/EISA Sparse Memory Space.
63:5	**Reserved** [hae0_1_m_res , read-only] Must Be Zero

Figure 16: High Address Extension Register 2 (HAE0_2)

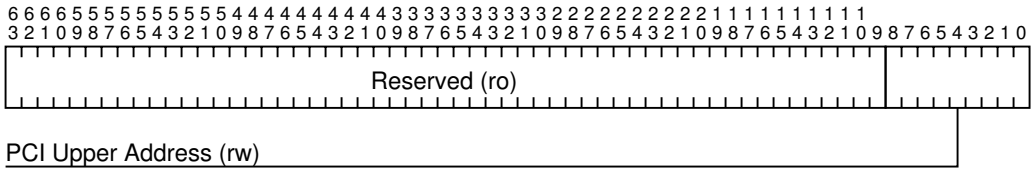


Table 11: High Address Extension Register 2 Description

Field	Description
8:0	**PCI Upper Address** [hae0_2_m_pua2 , read/write] These bits are used as PCI_AD<31:23> in accessing the 8MB of PCI I/O Space.
63:9	**Reserved** [hae0_2_m_res , read-only] Must Be Zero

Figure 17: High Address Extension Register 3 (HAE0_3)

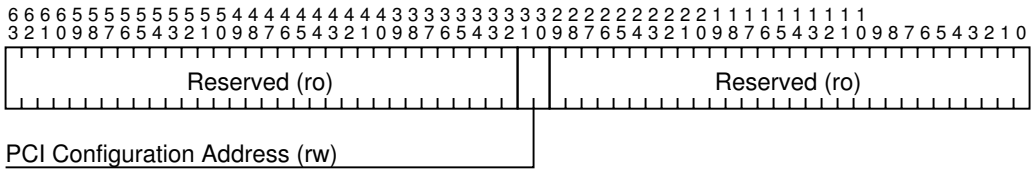


Table 12: High Address Extension Register 3 Description

Field	Description
29:0	**Reserved** [hae0_3_m_res , read-only] Must Be Zero
31:30	**PCI Configuration Address** [hae0_3_m_pca , read/write] These bits are used as PCI_AD<1:0> in accessing PCI Configuration Space and differentiate Type 0 and Type 1 cycles.
63:32	**Reserved** [hae0_3_m_res , read-only] Must Be Zero

Figure 18: High Address Extension Register 4 (HAE0_4)

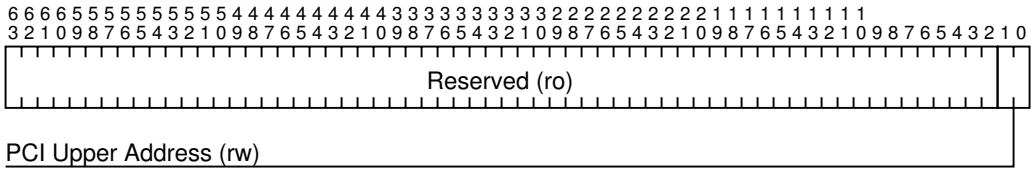


Table 13: High Address Extension Register 4 Description

Field	Description
1:0	**PCI Upper Address** [hae0_4_m_pua1 , read/write] These bits are used as PCI_AD<31:30> in accessing th 1GB of PCI/EISA Dense Memory Space.
63:2	**Reserved** [hae0_4_m_res , read-only] Must Be Zero

Figure 19: PC "Hole" Base Register (HBASE)

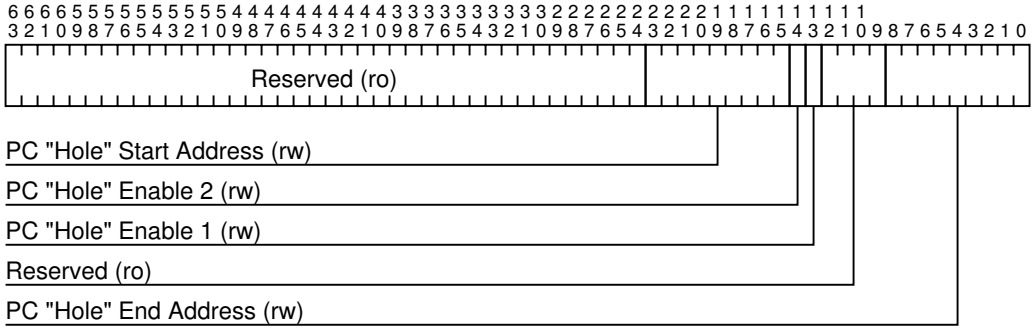


Table 14: PC "Hole" Base Register Description

Field	Description
8:0	**PC "Hole" End Address** [hbase_m_phea , read/write] When HBASE<14> is set, these bits are compared to PCI_AD<23:15> of incoming PCI cycles to determine if the address falls below the upper bound of the programmable PC "hole". The address contained in this register signifies the last valid upper PCI address in the hole. Bits 8:5 of this register must be non-zero when this hole is enabled.
12:9	**Reserved** [hbase_m_res , read-only] Must Be Zero
13	**PC "Hole" Enable 1** [hbase_m_phe1 , read/write] When this bit is set, the T2 gate array will not respond to PCI addresses which fall within the fixed PC "hole" between 512KB and 1MB.

Table 14 (Cont.): PC "Hole" Base Register Description

Field	Description
14	**PC "Hole" Enable 2** [hbase_m_phe1 , read/write] When this bit is set, the T2 gate array will not respond to PCI addresses which fall between the programmable PC "hole" start and end addresses contained in HBASE<23:26> and HBASE<7:0>, respectively.
23:15	**PC "Hole" Start Address** [hbase_m_phsa , read/write] When HBASE1<9> is set, these bits are compared to PCI_AD<23:15> of incoming PCI cycles to determine if the address falls above the lower bound of the programmable PC "hole". The address contained in this register signifies the last valid lower PCI address in the hole. Bits 23:20 of this register must be non-zero when this hole is enabled.
63:24	**Reserved** [hbase_m_res , read-only] Must Be Zero

Figure 20: Window Base Register 1 (WBASE1)

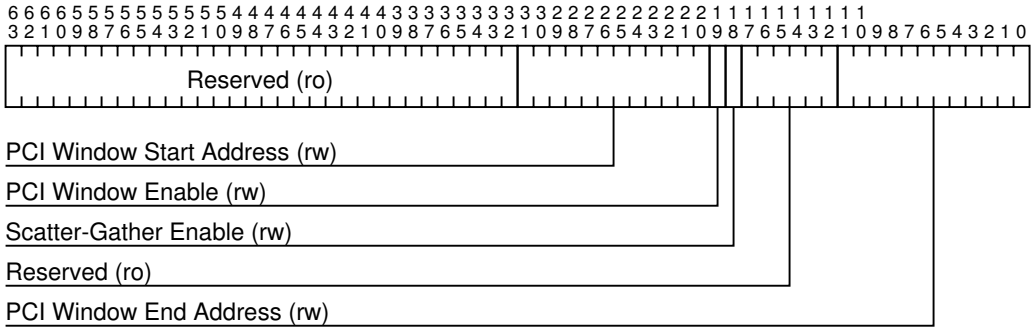


Table 15: Window Base Register 1 Description

Field	Description
11:0	**PCI Window End Address** [wbase1_m_pwea , read/write] When WBASE1<19> is set, these bits are compared to PCI_AD<31:20> of incoming PCI cycles to determine if the address falls at or below the upper bound of the first programmable PCI window.
17:12	**Reserved** [wbase1_m_res , read-only] Must Be Zero
18	**Scatter-Gather Enable** [wbase1_m_sge , read/write] When this bit is set, scatter-gather mapping of PCI addresses within the first PCI target window is enabled. When clear, PCI addresses are direct-mapped.
19	**PCI Window Enable** [wbase1_m_pwe , read/write] When this bit is set, the T2 gate array will respond to PCI addresses which fall at or between the first programmable PCI window start and end addresses contained in WBASE1<31:20> and WBASE1<11:0>, respectively.
31:20	**PCI Window Start Address** [wbase1_m_pwsa , read/write]

Table 15 (Cont.): Window Base Register 1 Description

Field	Description
	When WBASE1<19> is set, these bits are compared to PCI_AD<31:20> of incoming PCI cycles to determine if the address falls at or above the lower bound of the first programmable PCI window.
63:32	**Reserved** [wbase1_m_res , read-only] Must Be Zero

Figure 21: Window Mask Register 1 (WMASK1)

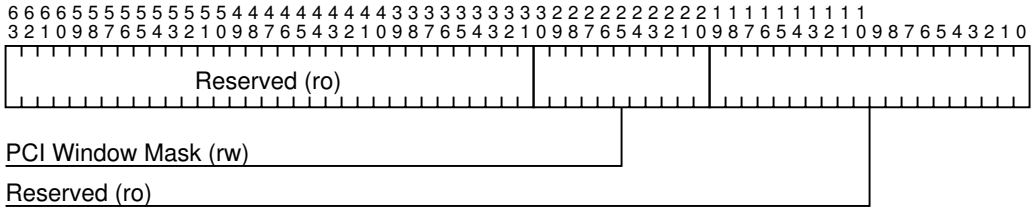


Table 16: Window Mask Register 1 Description

Field	Description
19:0	**Reserved** [wmask1_m_res , read-only] Must Be Zero
30:20	**PCI Window Mask** [wmask1_m_pwm , read/write] Specifies the size of the first PCI target window and is also used in the translation of the CPU address.
63:31	**Reserved** [wmask1_m_res , read-only] Must Be Zero

Figure 22: Translated Base Register 1 (TBASE1)

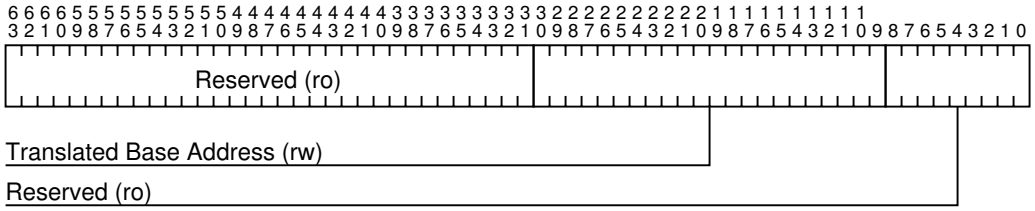


Table 17: Translated Base Register 1 Description

Field	Description
8:0	**Reserved** [tbase1_m_res , read-only] Must Be Zero
30:9	**Translated Base Address** [tbase1_m_tba , read/write] If WBASE1<18> is set, this specifies the base address for the scatter-gather map table for the first PCI target window. If WBASE1<18> is clear, this specifies the base CPU address of the translated PCI address for the first PCI target window.
63:31	**Reserved** [tbase1_m_res , read-only] Must Be Zero

Figure 23: Window Base Register 2 (WBASE2)

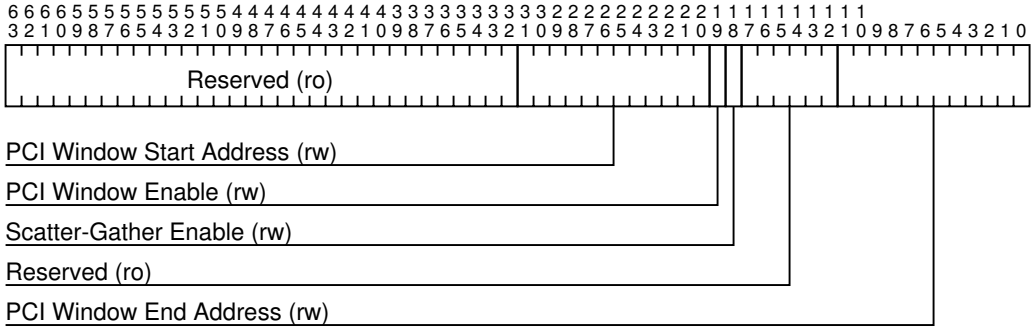


Table 18: Window Base Register 2 Description

Field	Description
11:0	**PCI Window End Address** [wbase2_m_pwea , read/write] When WBASE2<19> is set, these bits are compared to PCI_AD<31:20> of incoming PCI cycles to determine if the address falls at or below the upper bound of the second programmable PCI window.
17:12	**Reserved** [wbase2_m_res , read-only] Must Be Zero
18	**Scatter-Gather Enable** [wbase2_m_sge , read/write] When this bit is set, scatter-gather mapping of PCI addresses within the second PCI target window is enabled. When clear, PCI addresses are direct-mapped.
19	**PCI Window Enable** [wbase2_m_pwe , read/write] When this bit is set, the T2 gate array will respond to PCI addresses which fall at or between the second programmable PCI window start and end addresses contained in WBASE2<31:20> and WBASE2<11:0>, respectively.
31:20	**PCI Window Start Address** [wbase2_m_pwsa , read/write] When WBASE2<19> is set, these bits are compared to PCI_AD<31:20> of incoming PCI cycles to determine if the address falls at or above the lower bound of the second programmable PCI window.

Table 18 (Cont.): Window Base Register 2 Description

Field	Description
63:32	**Reserved** [wbase2_m_res , read-only] Must Be Zero

Figure 24: Window Mask Register 2 (WMASK2)

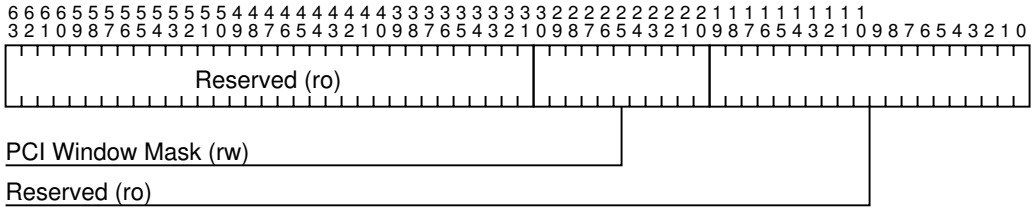


Table 19: Window Mask Register 2 Description

Field	Description
19:0	**Reserved** [wmask2_m_res , read-only] Must Be Zero
30:20	**PCI Window Mask** [wmask2_m_pwm , read/write] Specifies the size of the second PCI target window and is also used in the translation of the CPU address.
63:31	**Reserved** [wmask2_m_res , read-only] Must Be Zero

Figure 25: Translated Base Register 2 (TBASE2)

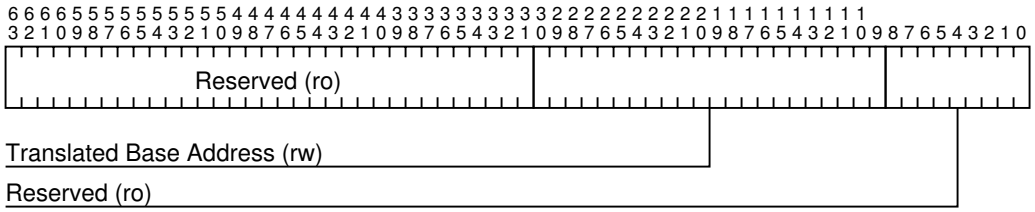


Table 20: Translated Base Register 2 Description

Field	Description
8:0	**Reserved** [tbase2_m_res , read-only] Must Be Zero

Table 20 (Cont.): Translated Base Register 2 Description

Field	Description
30:9	**Translated Base Address** [tbase2_m_tba , read/write] If WBASE2<18> is set, this specifies the base address for the scatter-gather map table for the second PCI target window. If WBASE2<18> is clear, this specifies the base CPU address of the translated PCI address for the second PCI target window.
63:31	**Reserved** [tbase2_m_res , read-only] Must Be Zero

Figure 26: TLB By-Pass Register (TLBBR)

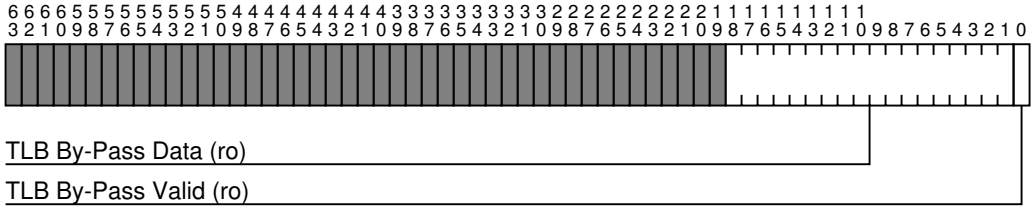


Table 21: TLB By-Pass Register Description

Field	Description
0	**TLB By-Pass Valid** [tlbbr_m_tlbbv , read-only]
18:1	**TLB By-Pass Data** [tlbbr_m_tlbbd , read-only] This register is used by the T2 CBUS Interface Logic as a temporary storage location for scatter-gather map entries read when the TLB has been disabled. Note: Writes to this register will not error but the contents of the register will not be affected. Reads or writes to this register while it is in use by the CBUS Interface Logic can result in unpredictable T2 behavior.

Figure 27: IVR Passive Release Register (IVRPR)

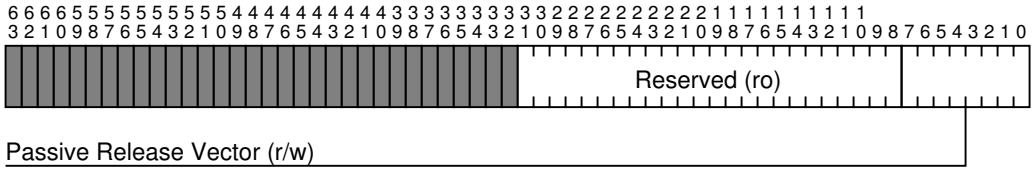


Table 22: IVR Passive Release Register Description

Field	Description
7:0	**Passive Release Vector** [ivrpr_m_prvect] This is the passive release vector. This should be the same passive release vector programmed into the 8259s.
31:8	**Reserved** [ivrpr_m_res , read-only] Must be Zero.

Figure 28: IVR Interrupt Address Register (IVIAR)

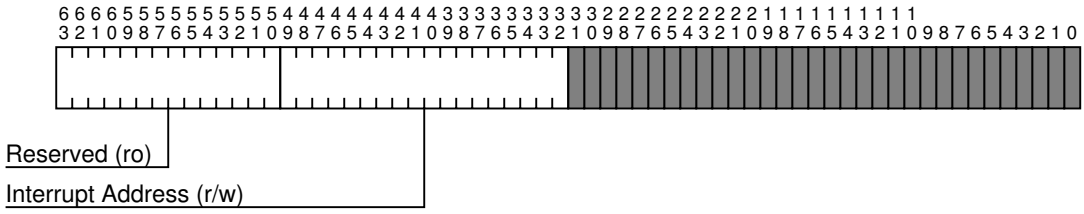


Table 23: IVR Interrupt Address Register Description

Field	Description
49:32	**Interrupt Address** [iviar_m_iv] This field contains the address of the interrupt vector. EV address bits 22:5 are loaded into this register.
63:50	**Reserved** [iviar_m_res , read-only] Must be Zero.

Figure 29: TLB Data Register 0 (TDR0)

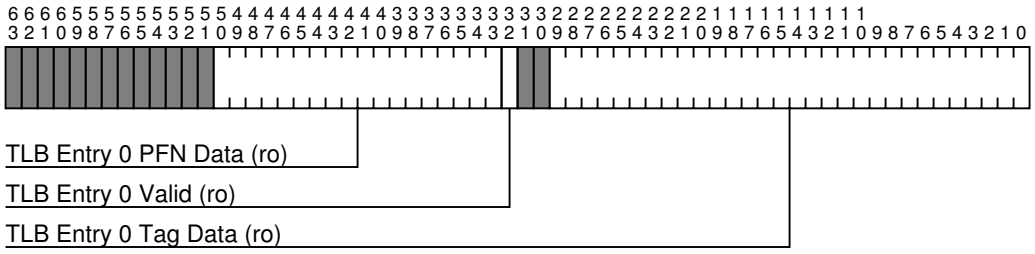


Table 24: TLB Data Register 0 Description

Field	Description
29:0	**TLB Entry 0 Tag Data** [tdr0_m_tlbtd0 , read-only] This field contains the value of the tag for TLB entry 0.

Table 24 (Cont.): TLB Data Register 0 Description

Field	Description
32	**TLB Entry 0 Valid** [tdr0_m_tlbv0 , read-only] This field contains the value of the valid bit for TLB entry 0.
50:33	**TLB Entry 0 PFN Data** [tdr0_m_tlbpfn0 , read-only] This field contains the value of the PFN for TLB entry 0. Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 30: TLB Data Register 1 (TDR1)

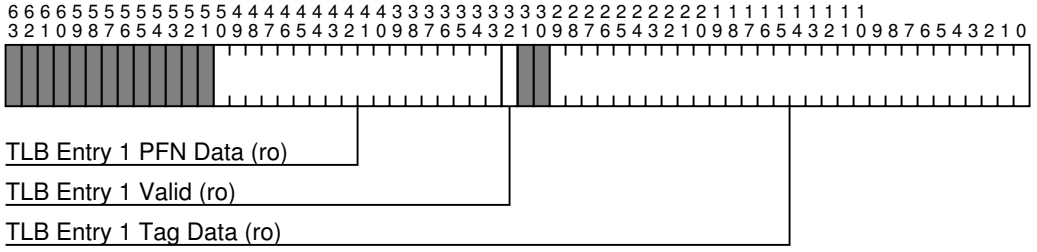


Table 25: TLB Data Register 1 Description

Field	Description
29:0	**TLB Entry 1 Tag Data** [tdr1_m_tlbtd1 , read-only] This field contains the value of the tag for TLB entry 1.
32	**TLB Entry 1 Valid** [tdr1_m_tlbv1 , read-only] This field contains the value of the valid bit for TLB entry 1.
50:33	**TLB Entry 1 PFN Data** [tdr1_m_tlbpfn1 , read-only] This field contains the value of the PFN for TLB entry 1. Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 31: TLB Data Register 2 (TDR2)

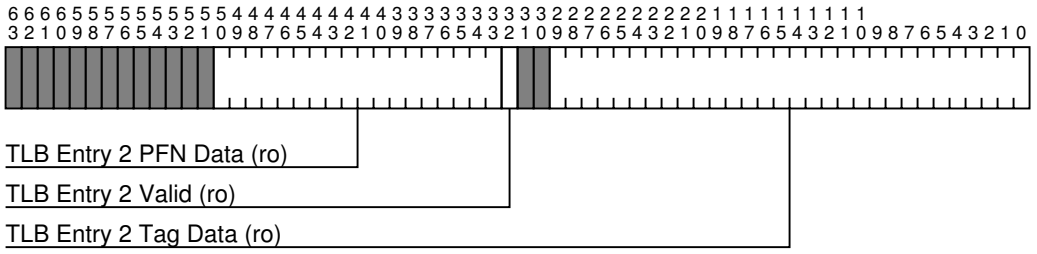


Table 26: TLB Data Register 2 Description

Field	Description
29:0	**TLB Entry 2 Tag Data** [tdr2_m_tlbtd2 , read-only]
	This field contains the value of the tag for TLB entry 2.
32	**TLB Entry 2 Valid** [tdr2_m_tlbv2 , read-only]
	This field contains the value of the valid bit for TLB entry 2.
50:33	**TLB Entry 2 PFN Data** [tdr2_m_tlbpfn2 , read-only]
	This field contains the value of the PFN for TLB entry 2.
	Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 32: TLB Data Register 3 (TDR3)

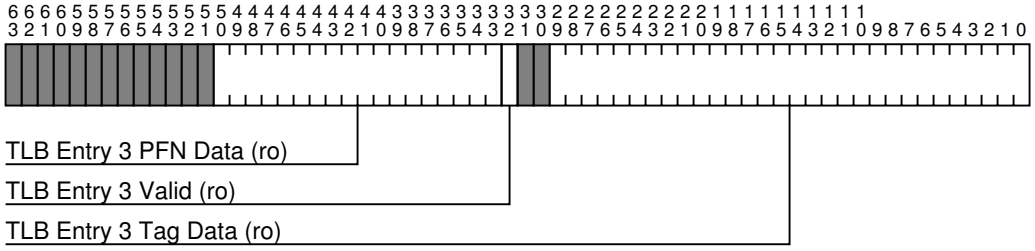


Table 27: TLB Data Register 3 Description

Field	Description
29:0	**TLB Entry 3 Tag Data** [tdr3_m_tlbtd3 , read-only]
	This field contains the value of the tag for TLB entry 3.
32	**TLB Entry 3 Valid** [tdr3_m_tlbv3 , read-only]
	This field contains the value of the valid bit for TLB entry 3.
50:33	**TLB Entry 3 PFN Data** [tdr3_m_tlbpfn3 , read-only]
	This field contains the value of the PFN for TLB entry 3.
	Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 33: TLB Data Register 4 (TDR4)

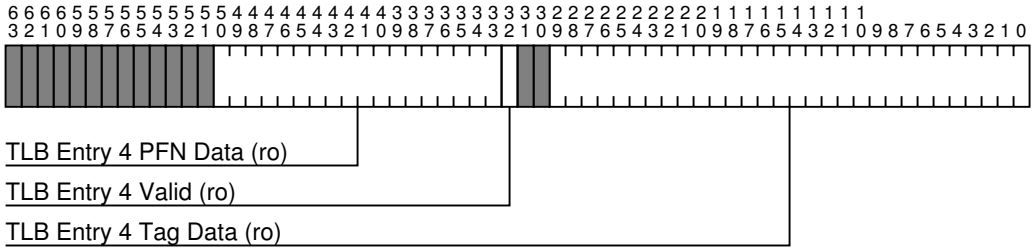


Table 28: TLB Data Register 4 Description

Field	Description
29:0	**TLB Entry 4 Tag Data** [tdr4_m_tlbtd4 , read-only] This field contains the value of the tag for TLB entry 4.
32	**TLB Entry 4 Valid** [tdr4_m_tlbv4 , read-only] This field contains the value of the valid bit for TLB entry 4.
50:33	**TLB Entry 4 PFN Data** [tdr4_m_tlbpfn4 , read-only] This field contains the value of the PFN for TLB entry 4. Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 34: TLB Data Register 5 (TDR5)

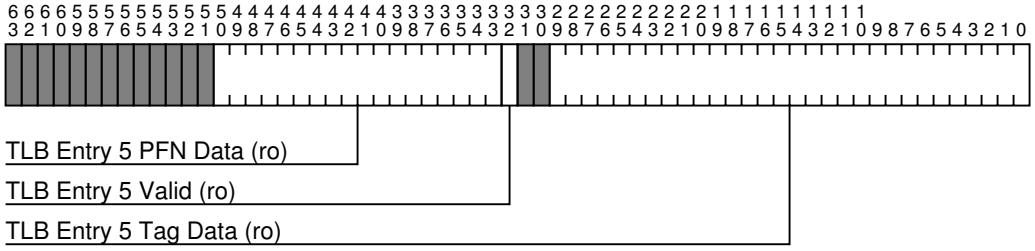


Table 29: TLB Data Register 5 Description

Field	Description
29:0	**TLB Entry 5 Tag Data** [tdr5_m_tlbtd5 , read-only] This field contains the value of the tag for TLB entry 5.
32	**TLB Entry 5 Valid** [tdr5_m_tlbv5 , read-only] This field contains the value of the valid bit for TLB entry 5.
50:33	**TLB Entry 5 PFN Data** [tdr5_m_tlbpfn5 , read-only] This field contains the value of the PFN for TLB entry 5. Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 35: TLB Data Register 6 (TDR6)

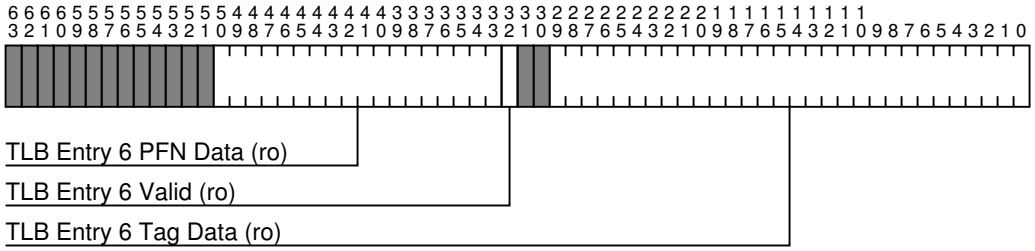


Table 30: TLB Data Register 6 Description

Field	Description
29:0	**TLB Entry 6 Tag Data** [tdr6_m_tlbtd6 , read-only]
	This field contains the value of the tag for TLB entry 6.
32	**TLB Entry 6 Valid** [tdr6_m_tlbv6 , read-only]
	This field contains the value of the valid bit for TLB entry 6.
50:33	**TLB Entry 6 PFN Data** [tdr6_m_tlbpfn6 , read-only]
	This field contains the value of the PFN for TLB entry 6.
	Note: Writes to this register will not error, but the contents of the TLB will not be affected.

Figure 36: TLB Data Register 7 (TDR7)

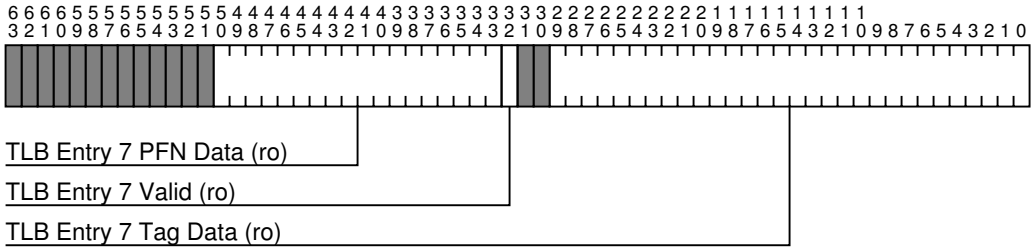


Table 31: TLB Data Register 7 Description

Field	Description
29:0	**TLB Entry 7 Tag Data** [tdr7_m_tlbtd7 , read-only]
	This field contains the value of the tag for TLB entry 7.
32	**TLB Entry 7 Valid** [tdr7_m_tlbv7 , read-only]
	This field contains the value of the valid bit for TLB entry 7.
50:33	**TLB Entry 7 PFN Data** [tdr7_m_tlbpfn7 , read-only]
	This field contains the value of the PFN for TLB entry 7.
	Note: Writes to this register will not error, but the contents of the TLB will not be affected.

4.2 PCI0 Configuration Registers

See Address chapter, Chapter 5 for a description of configuration space.

Table 32: PCI0 Configuration Space

EV Address	Device	PCI Address
3.9001.0000	Tulip (NI) Configuration Space	0000.0800
3.9002.0000	NCR 53C810 (SCSI) Configuration Space	0000.1000

Table 32 (Cont.): PCI0 Configuration Space

EV Address	Device	PCI Address
3.9004.0000	PCEB (EISA Bridge) Configuration Space	0000.2000
3.9008.0000	Reserved	0000.4000
3.9010.0000	Reserved	0000.8000
3.9020.0000	Reserved	0001.0000
3.9040.0000	PCI slot 0	0002.0000
3.9080.0000	PCI slot 1	0004.0000
3.9100.0000	PCI slot 2	0008.0000
3.9200.0000	Reserved	0010.0000
3.9400.0000	Reserved	0020.0000

4.2.1 T2

Since the T2 is the Cbus to PCI bridge, it does not contain any configuration registers. The T2 does not respond to configuration cycles on the PCI.

4.2.2 TULIP Configuration Registers

TULIP enables a full software driven initialization and configuration by providing a way for software to identify and query TULIP.

TULIP treats configuration space write operations to registers that are reserved as access that complete normally on the PCI bus but the data is discarded. Read accesses to reserved or unimplemented registers complete normally and a data value of "0" is returned.

Table 33: TULIP Configuration Registers

Register Name	Mnemonic	EV Address	PCI Address	Length	Attributes	Default Value
Configuration ID	CFID	3.9001.0018	0800	4 bytes	RO	00021011
Configuration Command/Status	CFCS	3.9001.0098	0804	4 bytes	R/W	02000000
Configuration Revision	CFRV	3.9001.0118	0808	4 bytes	RO	00000001
Configuration Latency Timer	CFLT	3.9001.0198	080C	4 bytes	R/W	00000000
Configuration Base IO Address	CBIO	3.9001.0218	0810	4 bytes	R/W	xxxxxxx1[1]

[1] Value after Reset is unpredictable except for last bit

4.2.3 NCR 53C810 Configuration Registers

The lower 128 bytes of the 53C810's 256-byte configuration space holds system parameters while the upper 128 bytes maps into the 53C810 operating registers. The operating (or device) registers should be accessed through PCI0 sparse I/O space. This just provides another way to access these registers. Registers not supported are not writeable and will return all zeroes when read.

Table 34: SCSI Configuration Registers

Register Name	Mnemonic	EV Address	PCI Address	Length	Attributes
Device ID/Vendor ID	DEVID	3.9002.0018	1000	4 bytes	RO
Status/Command	CSR	3.9002.0098	1004	4 bytes	R/W
Revision ID	REVID	3.9002.0118	1008	4 bytes?	RO
Master Latency Timer	MLTIM	3.9002.0198	100C	4 bytes	R/W
Base Address 0 (I/O)	BASE0IO	3.9002.0218	1010	4 bytes	R/W
Base Address 1 (Memory)	BASE1MEM	3.9002.0298	1014	4 bytes	R/W
Reserved or Not supported			1018-103C		
Not defined			1040-107F		
		SCSI Operating Registers[1]			
SCSI Control 0	SCNTL0	3.9002.1000	1080	1 byte	R/W
SCSI Control 1	SCNTL1	3.9002.1020	1081	1 byte	R/W
SCSI Control 2	SCNTL2	3.9002.1040	1082	1 byte	R/W
SCSI Control 3	SCNTL3	3.9002.1060	1083	1 byte	R/W
SCSI Chip ID	SCID	3.9002.1080	1084	1 byte	R/W
SCSI Transfer	SXFER	3.9002.10A0	1085	1 byte	R/W
SCSI Destination ID	SDID	3.9002.10C0	1086	1 byte	R/W
General Purpose Bits	GPREG	3.9002.10E0	1087	1 byte	R/W
SCSI First Byte Received	SFBR	3.9002.1100	1088	1 byte	R/W
SCSI Output Control Latch	SOCL	3.9002.1120	1089	1 byte	R/W
SCSI Selector ID	SSID	3.9002.1140	108A	1 byte	RO
SCSI Bus Control Lines	SBCL	3.9002.1160	108B	1 byte	R/W
DMA Status	DSTAT	3.9002.1180	108C	1 byte	R/W
SCSI Status 0	SSTAT0	3.9002.11A0	108D	1 byte	RO
SCSI Status 1	SSTAT1	3.9002.11C0	108E	1 byte	RO
SCSI Status 2	SSTAT2	3.9002.11E0	108F	1 byte	RO
Data Structure Address	DSA	3.9002.1218	1090	4 bytes	R/W
Interrupt Status	ISTAT	3.9002.1280	1094	1 byte	R/W
Reserved		3.9002.12A0	1095		
Chip Test 0	CTEST0	3.9002.1300	1098	1 byte	R/W
Chip Test 1	CTEST1	3.9002.1320	1099	1 byte	RO
Chip Test 2	CTEST2	3.9002.1340	109A	1 byte	RO
Chip Test 3	CTEST3	3.9002.1360	109B	1 byte	RO
Temporary Stack	TEMP	3.9002.1398	109C	4 bytes	R/W

[1]These registers can also be accessed through PCI0 Sparse I/O or Memory space. See table Table 37 for I/O addresses. This table just gives the addresses in configuration space.

Table 34 (Cont.): SCSI Configuration Registers

Register Name	Mnemonic	EV Address	PCI Address	Length	Attributes
DMA FIFO	DFIFO	3.9002.1400	10A0	1 byte	R/W
Chip Test 4	CTEST4	3.9002.1420	10A1	1 byte	R/W
Chip Test 5	CTEST5	3.9002.1440	10A2	1 byte	R/W
Chip Test 6	CTEST6	3.9002.1460	10A3	1 byte	R/W
DMA Byte Counter	DBC	3.9002.1498	10A4	4 bytes	R/W
DMA Command	DCMD	3.9002.14E0	10A7	1 byte	R/W
DMA Next Address for Data	DNAD	3.9002.1518	10A8	4 bytes	R/W
DMA SCRIPTS Pointer	DSP	3.9002.1598	10AC	4 bytes	R/W
DMA SCRIPTS Pointer Save	DSPS	3.9002.1618	10B0	4 bytes	R/W
General Purpose Scratch Pad A	SCRATCHA	3.9002.1698	10B4	4 bytes	R/W
DMA Mode	DMODE	3.9002.1700	10B8	1 byte	R/W
DMA Interrupt Enable	DIEN	3.9002.1720	10B9	1 byte	R/W
DMA Watchdog Timer	DWT	3.9002.1740	10BA	1 byte	R/W
DMA Control	DCNTL	3.9002.1760	10BB	1 byte	R/W
Sum Output of Internal Adder	ADDER	3.9002.1798	10BC	4 bytes	RO
SCSI Interrupt Enable 0	SIEN0	3.9002.1800	10C0	1 byte	R/W
SCSI Interrupt Enable 1	SIEN1	3.9002.1820	10C1	1 byte	R/W
SCSI Interrupt Status 0	SIST0	3.9002.1840	10C2	1 byte	RO
SCSI Interrupt Status 1	SIST1	3.9002.1860	10C3	1 byte	RO
SCSI Longitudinal Parity	SLPAR	3.9002.1880	10C4	1 byte	RO
Reserved		3.9002.18A0	10C5	1 byte	
Memory Access Control	MACNTL	3.9002.18C0	10C6	1 byte	R/W
General Purpose Control	GPCNTL	3.9002.18E0	10C7	1 byte	R/W
SCSI Timer 0	STIME0	3.9002.1900	10C8	1 byte	R/W
SCSI Timer 1	STIME1	3.9002.1920	10C9	1 byte	R/W
Response ID	RESPID	3.9002.1940	10CA	1 byte	R/W
Reserved		3.9002.1960	10CB	1 byte	
SCSI Test 0	STEST0	3.9002.1980	10CC	1 byte	RO
SCSI Test 1	STEST1	3.9002.19A0	10CD	1 byte	RO
SCSI Test 2	STEST2	3.9002.19C0	10CE	1 byte	R/W
SCSI Test 3	STEST3	3.9002.19E0	10CF	1 byte	R/W
SCSI Input Data Latch	SIDL	3.9002.1A00	10D0	1 byte	RO
Reserved		3.9002.1A20	10D1		
SCSI Output Data Latch	SODL	3.9002.1A80	10D4	1 byte	R/W
Reserved		3.9002.1AA0	10D5		
SCSI Bus Data Lines	SBDL	3.9002.1100	10D8	1 byte	RO
Reserved		3.9002.1B20	10D9		
General Purpose Scratch Pad B	SCRATCHB	3.9002.1B98	10DC	4 bytes	R/W

4.2.4 PCEB Configuration Registers

Table 35: PCEB Configuration Registers

Register Name	Mnemonic	EV Address	PCI Address	Length	Attributes	Default Value
Vendor ID	VENID	3.9004.0008	2000	2 bytes	RO	8086
Device ID	DEVID	3.9004.0048	2002	2 bytes	RO	0482
PCI Command Register	PCICMD	3.9004.0088	2004	2 bytes	R/W	0007
PCI Status Register	PCISTS	3.9004.00C8	2006	2 bytes	W1C[3]	0200
Revision ID	REVID	3.9004.0100	2008	1 byte	RO	03[1]
Reserved		2009-200C				
Master Latency Timer	MLTIM	3.9004.01A0	200D	1 byte	R/W	F8
Reserved			200E-203F			
PCI Control	PCICON	3.9004.0800	2040	1 byte	R/W	60
PCI Arbiter Control	ARBCON	3.9004.0820	2041	1 byte	R/W	80
PCI Arbiter Priority Control	ARBPRI	3.9004.0840	2042	1 byte	R/W	04
Reserved			2043			
MEMCS# Control	MCSCON	3.9004.0860	2044	1 byte	R/W	00
MEMCS# Bottom of Hole	MCSBOH	3.9004.0880	2045	1 byte	R/W	10
MEMCS# Top of Hole	MCSTOH	3.9004.08A0	2046	1 byte	R/W	0F
MEMCS# Top of Memory	MCSTOM	3.9004.08C0	2047	1 byte	R/W	00
EISA Address Decoder Control for 0-1MB Space	EADCIM	3.9004.0908	2048	2bytes	R/W	0001
Reserved			204A-204B			
ISA I/O Recovery Time Control	IORTC	3.9004.0980	204C	1 byte	R/W	56
Reserved			204D-2053			
MEMCS# Attribute Register #1	MAR1	3.9004.A80	2054	1 byte	R/W	00
MEMCS# Attribute Register #2	MAR2	3.9004.AA0	2055	1 byte	R/W	00
MEMCS# Attribute Register #3	MAR3	3.9004.AC0	2056	1 byte	R/W	00
Reserved			2057			
PCI Decode Control	PDCON	3.9004.0B00	2058	1 byte	R/W	00
Reserved			2059			
EISA Address Decoder Control - Extension	EADCX	3.9004.0B20	205A	1 byte	R/W	00
Reserved			205B			
EISA-PCI Memory Region Attributes	EPMRA	3.9004.0B80	205C	1 byte	R/W	00
Reserved			205D-205F			
EISA-PCI Memory Region 1 - Base Address	EPMB1	3.9004.0C08	2060	2 bytes	R/W	FFFF

[1]Revision ID changes with revision of part. This is the minimum revision as the date of the spec.

[3]Write 1 to Clear

Table 35 (Cont.): PCEB Configuration Registers

Register Name	Mnemonic	EV Address	PCI Address	Length	Attributes	Default Value
EISA-PCI Memory Region 1 - Limit Address	EPML1	3.9004.0C48	2062	2 bytes	R/W	0000
EISA-PCI Memory Region 2 - Base Address	EPMB2	3.9004.0C88	2064	2 bytes	R/W	FFFF
EISA-PCI Memory Region 2 - Limit Address	EPML2	3.9004.0CC8	2066	2 bytes	R/W	0000
EISA-PCI Memory Region 3 - Base Address	EPMB3	3.9004.0D08	2068	2 bytes	R/W	FFFF
EISA-PCI Memory Region 3 - Limit Address	EPML3	3.9004.0D48	206A	2 bytes	R/W	0000
EISA-PCI Memory Region 4 - Base Address	EPMB4	3.9004.0D88	206C	2 bytes	R/W	FFFF
EISA-PCI Memory Region 4 - Limit Address	EPML5	3.9004.0DC8	206E	2 bytes	R/W	0000
EISA-PCI I/O Region 1- Base Address	EPIOB1	3.9004.0E08	2070	2 bytes	R/W	FFFC
EISA-PCI I/O Region 1- Limit Address	EPIOL1	3.9004.0E48	2072	2 bytes	R/W	0000
EISA-PCI I/O Region 2- Base Address	EPIOB2	3.9004.0E88	2074	2 bytes	R/W	FFFC
EISA-PCI I/O Region 2- Limit Address	EPIOL2	3.9004.0EC8	2076	2 bytes	R/W	0000
EISA-PCI I/O Region 3- Base Address	EPIOB3	3.9004.0F08	2078	2 bytes	R/W	FFFC
EISA-PCI I/O Region 3- Limit Address	EPIOL3	3.9004.0F48	207A	2 bytes	R/W	0000
EISA-PCI I/O Region 4- Base Address	EPIOB4	3.9004.0F88	207C	2 bytes	R/W	FFFC
EISA-PCI I/O Region 4- Limit Address	EPIOL5	3.9004.0FC8	207E	2 bytes	R/W	0000
BIOS Timer Base Address	BTMR	3.9004.1008	2080	2 bytes	R/W	0078
Reserved			2082-2083			
EISA Latency Timer Control Register	ELTCR	3.9004.1080	2084	1 byte	R/W	0
Reserved			2085-2087			
PCEB Test Control Register[2]	PTCR	3.9004.1118	2088	4 bytes	R/W	00000000
Reserved			208C-20FF			

[2]**DO NOT WRITE**

4.3 PCI0 CSRs in PCI0 I/O Sparse Space

4.3.1 T2

T2's registers are only available from the system bus. They are not available from the PCI.

4.3.2 TULIP Device Registers

The description of the Tulip registers can be found in the TULIP specification. Table 36 gives the EV Address and the PCI address of the Tulip registers.

Table 36: TULIP Registers

Register Name	Mnemonic	EV Address[3]	PCI Address[4]	Length[1], [2]	Attributes
Bus Mode Register	CSR0	BB+0018	bb+00	4 bytes	R/W
Transmit Poll Demand	CSR1	BB+0118	bb+08	4 bytes	R/W
Receive Poll Demand	CSR2	BB+0218	bb+10	4 bytes	R/W
Rx Ring Base Address	CSR3	BB+0318	bb+18	4 bytes	R/W
Tx Ring Base Address	CSR4	BB+0418	Abb+0	4 bytes	R/W
Status Register	CSR5	BB+0518	bb+28	4 bytes	R/W
Serial Command Register	CSR6	BB+0618	bb+30	4 bytes	R/W
Interrupt Mask Register	CSR7	BB+0718	bb+38	4 bytes	R/W
Missed Frame Counter	CSR8	BB+0818	bb+40	4 bytes	RO
Address and Mode Diagnostic Register	CSR9	BB+0918	bb+48	4 bytes	R/W
Data Diagnostic Register	CSR10	BB+0A18	bb+50	4 bytes	R/W
Full Duplex Register	CSR11	BB+0B18	bb+58	4 bytes	RO
SIA Status Register	CSR12	BB+0C18	bb+60	4 bytes	R/W
SIA Connectivity Register	CSR13	BB+0D18	bb+68	4 bytes	R/W
SIA Tx Rx Register	CSR14	BB+0E18	bb+70	4 bytes	R/W
SIA General Register	CSR15	BB+0F18	bb+78	4 bytes	R/W

[1]Tulip Register access is only longword. Byte accesses are not supported. Accessing a non longword address register causes unpredicatable data results.

[2]Tulip CSRs are quadword aligned and can only be accessed using longword instructions.

[3]BB = Base EV address of Tulip registers.

[4]bb = Base PCI address of Tulip registers.

4.3.3 NCR 53C810 Device Registers

The description of the NCR 53C810 registers can be found in the NCR specification. Table 37 show the registers within the SCSI chip.
The SCSI registers can be accessed by byte addresses or longword addresses. This allows several byte size registers to be written at once if using a longword address. See the NCR specification and Table 37, to see how the registers line up.

Table 37: SCSI Registers

Register Name	Mnemonic	EV Address[2]	PCI Address[3]	Length	Attributes
SCSI Control 0	SCNTL0	BB+0000	bb+00	1 byte	R/W
SCSI Control 1	SCNTL1	BB+0020	bb+01	1 byte	R/W
SCSI Control 2	SCNTL2	BB+0040	bb+02	1 byte	R/W
SCSI Control 3	SCNTL3	BB+0060	bb+03	1 byte	R/W
SCSI Chip ID	SCID	BB+0080	bb+04	1 byte	R/W
SCSI Transfer	SXFER	BB+00A0	bb+05	1 byte	R/W
SCSI Destination ID	SDID	BB+00C0	bb+06	1 byte	R/W
General Purpose Bits	GPREG	BB+00E0	bb+07	1 byte	R/W
SCSI First Byte Received	SFBR	BB+0100	bb+08	1 byte	R/W
SCSI Output Control Latch	SOCL	BB+0120	bb+09	1 byte	R/W
SCSI Selector ID	SSID	BB+0140	bb+0A	1 byte	RO
SCSI Bus Control Lines	SBCL	BB+0160	bb+0B	1 byte	R/W
DMA Status	DSTAT	BB+0180	bb+0C	1 byte	R/W
SCSI Status 0	SSTAT0	BB+01A0	bb+0D	1 byte	RO
SCSI Status 1	SSTAT1	BB+01C0	bb+0E	1 byte	RO
SCSI Status 2	SSTAT2	BB+01E0	bb+0F	1 byte	RO
Data Structure Address	DSA	BB+0218	bb+10	4 bytes	R/W
Interrupt Status	ISTAT	BB+0280	bb+14	1 byte	R/W
Reserved		BB+02A0	bb+15		
Chip Test 0	CTEST0	BB+0300	bb+18	1 byte	R/W
Chip Test 1	CTEST1	BB+0320	bb+19	1 byte	RO
Chip Test 2	CTEST2	BB+0340	bb+1A	1 byte	RO
Chip Test 3	CTEST3	BB+0360	bb+1B	1 byte	RO
Temporary Stack	TEMP	BB+0398	bb+1C	4 bytes	R/W
DMA FIFO	DFIFO	BB+0400	bb+20	1 byte	R/W
Chip Test 4	CTEST4	BB+0420	bb+21	1 byte	R/W
Chip Test 5	CTEST5	BB+0440	bb+22	1 byte	R/W
Chip Test 6	CTEST6	BB+0460	bb+23	1 byte	R/W
DMA Byte Counter	DBC	BB+0498	bb+24	4 bytes	R/W
DMA Command	DCMD	BB+04E0	bb+27	1 byte	R/W
DMA Next Address for Data	DNAD	BB+0518	bb+28	4 bytes	R/W
DMA SCRIPTS Pointer	DSP	BB+0598	bb+2C	4 bytes	R/W
DMA SCRIPTS Pointer Save	DSPS	BB+0618	bb+30	4 bytes	R/W
General Purpose Scratch Pad A	SCRATCHA	BB+0698	bb+34	4 bytes	R/W
DMA Mode	DMODE	BB+0700	bb+38	1 byte	R/W
DMA Interrupt Enable	DIEN	BB+0720	bb+39	1 byte	R/W
DMA Watchdog Timer	DWT	BB+0740	bb+3A	1 byte	R/W
DMA Control	DCNTL	BB+0760	bb+3B	1 byte	R/W

[2]BB = Base EV address of Tulip registers.

[3]bb = Base PCI address of Tulip registers.

Table 37 (Cont.): SCSI Registers

Register Name	Mnemonic	EV Address[2]	PCI Address[3]	Length	Attributes
Sum Output of Internal Adder	ADDER	BB+0798	bb+3C	4 bytes	RO
SCSI Interrupt Enable 0	SIEN0	BB+0800	bb+40	1 byte	R/W
SCSI Interrupt Enable 1	SIEN1	BB+0820	bb+41	1 byte	R/W
SCSI Interrupt Status 0	SIST0	BB+0840	bb+42	1 byte	RO
SCSI Interrupt Status 1	SIST1	BB+0860	bb+43	1 byte	RO
SCSI Longitudinal Parity	SLPAR	BB+0880	bb+44	1 byte	RO
Reserved		BB+08A0	bb+45	1 byte	
Memory Access Control	MACNTL	BB+08C0	bb+46	1 byte	R/W
General Purpose Control	GPCNTL	BB+08E0	bb+47	1 byte	R/W
SCSI Timer 0	STIME0	BB+0900	bb+48	1 byte	R/W
SCSI Timer 1	STIME1	BB+0920	bb+49	1 byte	R/W
Response ID	RESPID	BB+0940	bb+4A	1 byte	R/W
Reserved		BB+0960	bb+4B	1 byte	
SCSI Test 0	STEST0	BB+0980	bb+4C	1 byte	RO
SCSI Test 1	STEST1	BB+09A0	bb+4D	1 byte	RO
SCSI Test 2	STEST2	BB+09C0	bb+4E	1 byte	R/W
SCSI Test 3	STEST3	BB+09E0	bb+4F	1 byte	R/W
SCSI Input Data Latch	SIDL	BB+0A00	bb+50	1 byte	RO
Reserved		BB+0A20	bb+51		
SCSI Output Data Latch	SODL	BB+0A80	bb+54	1 byte	R/W
Reserved		BB+0AA0	bb+55		
SCSI Bus Data Lines	SBDL	BB+0B00	bb+58	1 byte	RO
Reserved		BB+0B20	bb+59		
General Purpose Scratch Pad B	SCRATCHB	BB+0B98	bb+5C	4 bytes	R/W
Reserved		BB+0C00-BB+FFFF	bb+60-bb+FF		

[2]BB = Base EV address of Tulip registers.

[3]bb = Base PCI address of Tulip registers.

4.3.4 PCEB I/O Registers

The only PCEB internal resource mapped to the PCI I/O space is the BIOS Timer. It contains a single 32-bit register mapped in the I/O space on the location determined by the value written into the BTMR configuration register. The BIOS Timer is accessible only from the PCI bus and not from the EISA bus. Sable does not use this BIOS Timer.

4.3.5 ESC I/O Registers

See Table 55 in the Appendix A for the addresses of the ESC I/O registers.

CHAPTER 5

ADDRESSING

This chapter describes the mapping of the 34-bit processor physical address space into memory and I/O space address, the translation of the processor initiated address into a PCI address, and the translation of PCI initiated addresses into physical memory addresses.

5.1 CPU Address Map

This section describes the view that the processor has of the 34-bit address map. This address is divided to form a memory address space and an I/O address space. The address space from the perspective of the processor is shown in Figure 37 and in Figure 38.

Figure 37: System Address Map

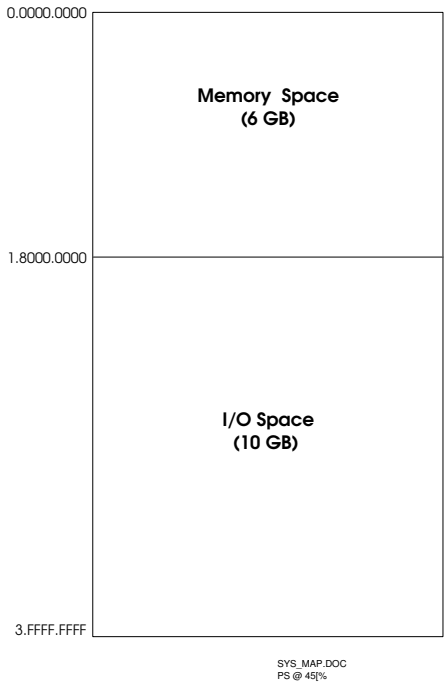
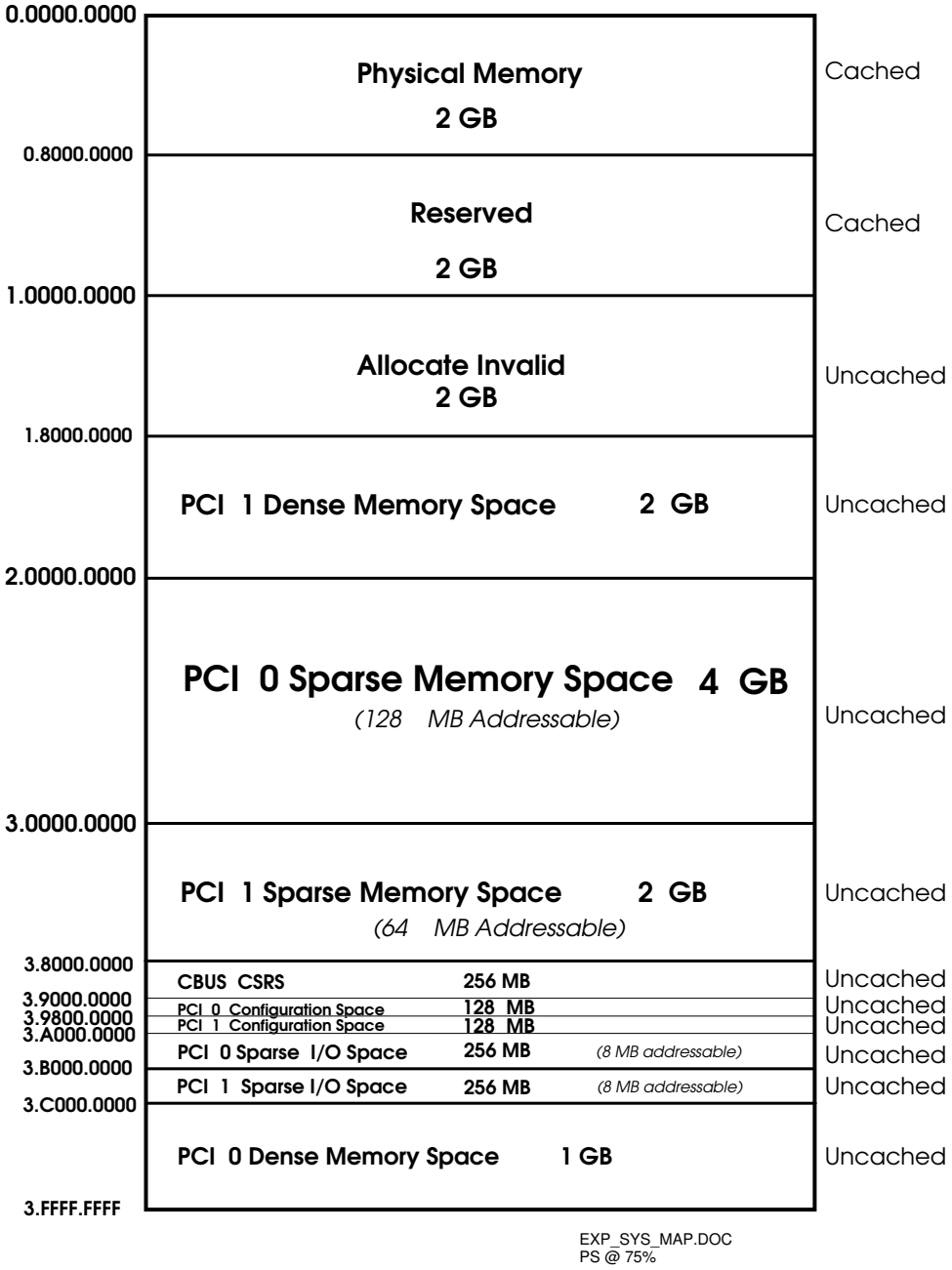


Figure 38: Expanded System Map



5.1.1 Memory Space

The 16GB address space is divided into four quadrants - Q0, Q1, Q2 and Q3. Quadrant 0 is cached . All the other quadrants are uncached. See System specification for more information on memory space.

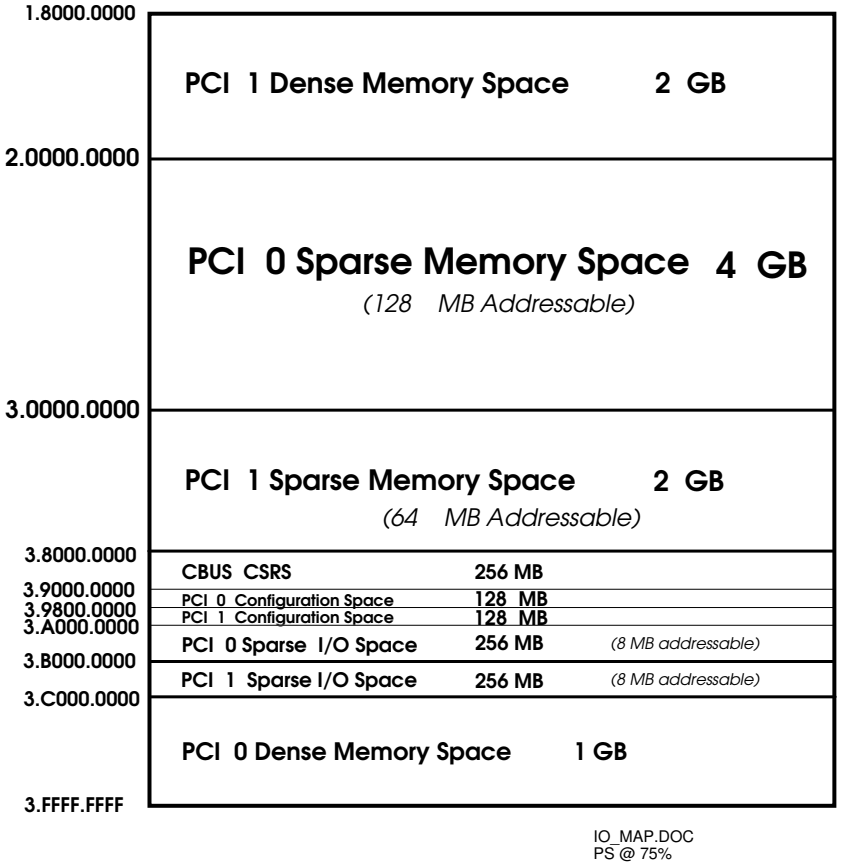
5.1.2 Memory Mapped I/O Space

All local CBUS CSRs and remote busses (PCI/EISA) reside in the I/O address space. The I/O address space is divided into a number of subspaces. Each subspace has different access characteristics. A reference to I/O space will cause a read/write to a CBUS CSR or a transaction on a remote bus.

10GB of the address map is used for memory mapped I/O. The devices which need to be memory mapped reside on the following busses in the Sable system.

- CBUS - Local CSR's for system operation.
- PCI0 - 32 bit PCI bus which is integral on all systems. Mapping is required for three slots and integral I/O.
- EISA - Provided by a PCI/EISA bridge on PCI0. Mapping is required for eight EISA slots and integral I/O on the Standard I/O module. The addressing for the EISA is found in the first 64KB of PCI0 sparse I/O space. See Figure 43.
- PCI1 - Optional 64 bit PCI bus. Mapping is required for two slots.

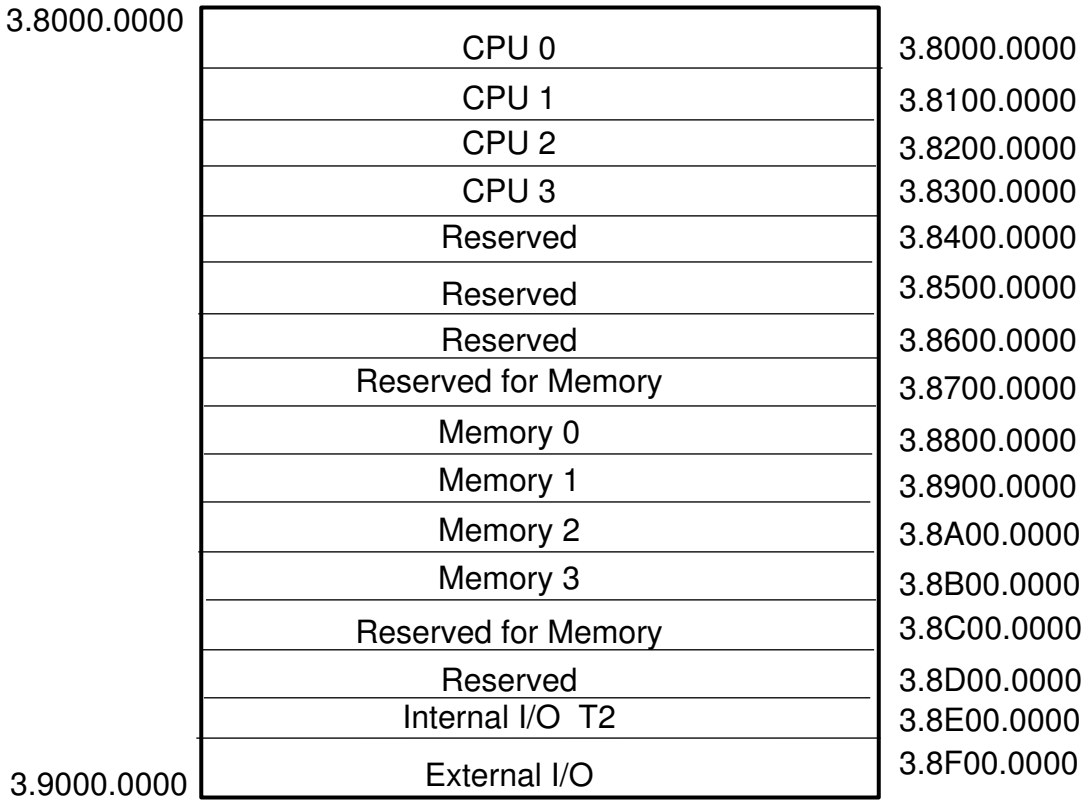
Figure 39: I/O Address Map (10GB)



5.1.2.1 CBUS CSRs

All accesses to this space follow the CBUS specification for access to local CSR's and must be implemented according to this specification. Figure 40 shows the CBUS CSR map.

Figure 40: CBUS CSRs



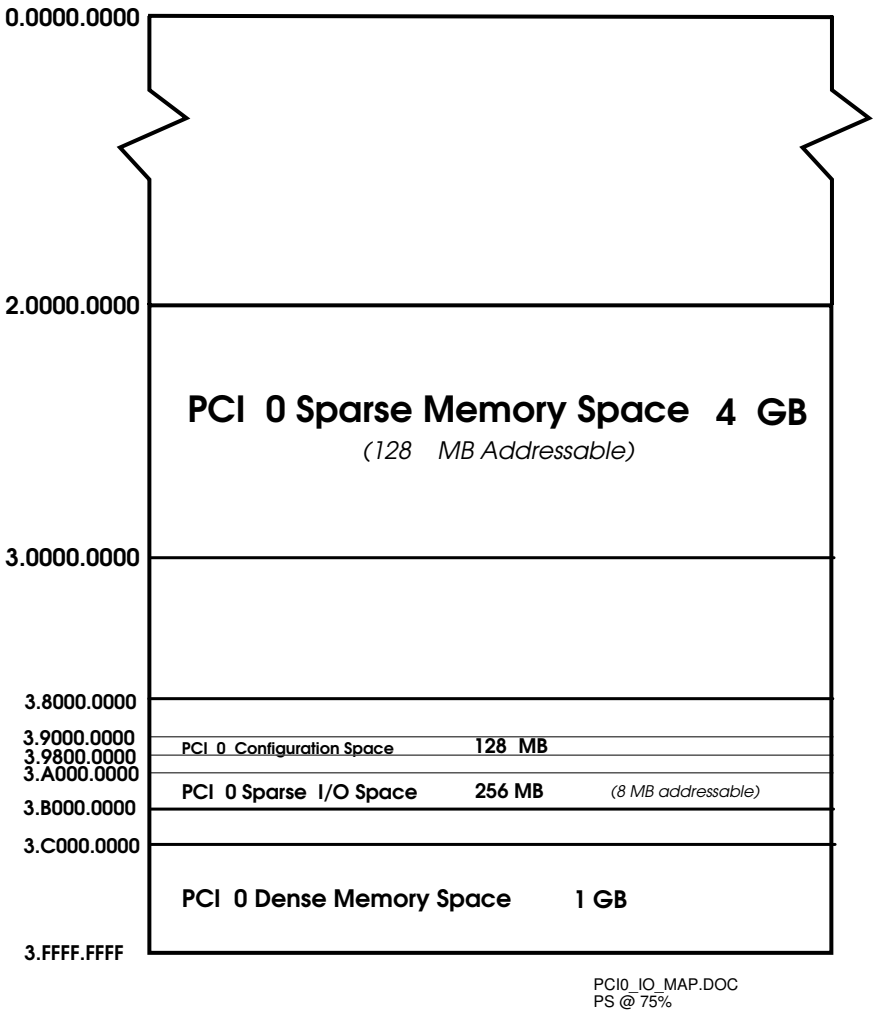
[YURYAN.NEW_SAB]CBUS_CSRS.DOC

5.1.2.2 PCI0

Each PCI bus in an Alpha system requires the following address spaces as defined by the PCI specification:

- PCI Memory - Used mostly for devices containing memory residing on PCI.
- PCI I/O Space
- PCI Configuration Space
- PCI Special Cycles

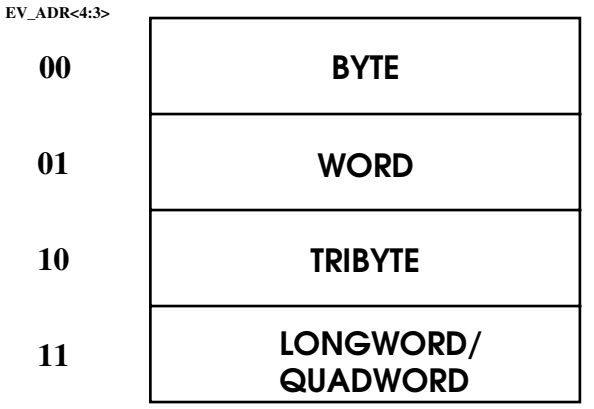
Figure 41: PCI0 Address Map from EV perspective



5.1.2.2.1 PCI0 Sparse Space

Sable supports regions of space referred to as PCI Sparse Space. Through these regions, the CPU can read or write devices residing on the PCI and EISA. A transaction through this region can be byte, word, longword, tribyte, or quadword in length. EISA and PCI are both longword wide buses with four byte enables to allow byte granularity reads and writes. The Alpha architecture allows only longword and quadword width reads and writes. A scheme was devised to generate masked PCI/EISA reads and writes using Alpha longword/quadword loads and stores. In this scheme, byte enables are generated using lower order address bits.

Figure 42: Sparse Space Attributes



SPARSE_SPACE.DOC
PS @50%

The rules for accessing sparse space are as follows:

- Sparse space supports all the byte encodings which may be generated in a Intel system to ensure compatibility with PCI devices/drivers. The results of some references are not explicitly defined. The hardware will complete the reference but the reference may produce unpredictable results. An error which can crash the system must not be generated.
- Software must use longword load or store instructions (LDL/STL) to perform a reference which is of longword length or less on the PCI bus. The bytes to be transferred must be positioned within the longword in the correct byte lanes as indicated by the PCI byte enables. The hardware will do no byte shifting within the longword. Quadword loads and stores must only be used to perform a quadword transfer. Use of STQ/LDQ instructions for any other references will produce unpredictable results which will vary from platform to platform.
- Read ahead is not allowed in sparse space as it may have detrimental side effects.
- Programmers must insert an MB (Memory Barrier) between sparse space accesses to prevent collapsing in the 21064 write buffer.
- Tribyte and longword encodings which are non-aligned will produce the same behavior as in Intel systems.
- The encoding of the EV address for sparse space read accesses to PCI space is shown in Table 38. An important point to note is that EV_Address[33:5] are directly available from the processor chip on the EV_ADDR lines. On read transactions the processor sends out address bits [4:3] on the write mask lines cWmask[1:0]. EV_Addr [2:0] are assumed to be zero. Accesses with EV_Addr[2:0] equal to a non-zero value will produce unpredictable results.

- The Encoding for sparse space write accesses is shown in Table 39. EV_ ADDR[4:3] have to be derived from the longword write masks cWmask[7:0]. The relation between cWmask[7:0] and EV_ADDR[4:3] is as follows:
  - — If cWmask[1:0] is non-zero, EV_Addr[4:3] is 00.
  - — If cWmask[3:2] is non-zero, EV_Addr[4:3] is 01.
  - — If cWmask[5:4] is non-zero, EV_Addr[4:3] is 10.
  - — If cWmask[7:6] is non-zero, EV_Addr[4:3] is 11.

  NOTE

  All other combinations of cWmask[7:0] are invalid and result in a longword write cycle on the PCI where the data is CAD_OW0:D31:D0 and a null byte mask field.

  NOTE

  EV_Addr [2:0] are assumed to be zero. Accesses with EV_Addr[2:0] equal to a non-zero value will produce unpredictable results.
- Sparse space accesses are allowed to both PCI I/O and PCI Memory space.
- Sparse space accesses to PCI memory/IO space can be byte, word, tribyte,longword or quadword accesses.
- PCI_AD[1:0] are zero on accesses to PCI memory space. PCI_AD[1:0] are encoded, as in the following tables, for accesses to PCI I/O space.

Table 38: Sable Sparse Space Read Encodings

	EV_ADDR[6:5] (Offset) CAD[4:3]	EV_ADDR[4:3] (Size) CAD[124,92]	Instruction	PCI C/BE	PCI_ AD[1:0][1]	Data Returned to Processor, EV_ DATA[127:00], CAD[127:00]
Byte	00	00	LDL[2]	1110	00	OW0:[D7:D0]
	01	00	LDL[2]	1101	01	OW0:[D15:D8]
	10	00	LDL[2]	1011	10	OW0:[D23:16]
	11	00	LDL[2]	0111	11	OW0:[D31:24]
-[3]	11	10	LDL[2]	0111	11	OW1:[D31:D24]
Word	00	01	LDL[2]	1100	00	OW0:[D79:D64]
	01	01	LDL[2]	1001	01	OW0:[D87:D72]
	10	01	LDL[2]	0011	10	OW0:[D95:D80]
-[3]	10	10	LDL[2]	0011	10	OW1:[D31:D16]
-[3]	10	10	LDL[2]	0011	10	OW1:[D95:D80]
Tribyte	00	10	LDL[2]	1000	00	OW1:[D23:D0]
	01	10	LDL[2]	0001	01	OW1:[D31:D8]
-[3]	01	11	LDL[2]	0001	01	OW1:[D95:d72]
-[3]	11	01	LDL[2]	0001	01	OW1:[D95:D72]
Longword	00	11	LDL[2]	0000	00	OW1:[D95:D64]
Quadword	11	11	LDQ	0000	00	OW1:[D95:64
				0000	00	OW1:[D127:D96]
Quadword[3]	11	11	LDL	0000	00	OW1:[D95:64]
				0000	00	OW1:[D127:D96]

[1]On I/O cycles only.

[2]All encodings which have LDL listed as the relevant EV instruction will produce the same effect if a LDQ is performed. Software should only use the encodings defined as **NOT** Sable implementation specific, as these will produce consistent results across Alpha platforms.

[3]Sable implementation specific.

Table 39: Sparse Space Write Encodings

	EV_ ADDR[6:5] (Offset) CAD[4:3]	EV_ ADDR[4:3] (Size)	cWmask[7:0][2]	CBUS Mask CAD[127:124] CAD[95:92]	Instruction	Valid EV Data EV_ DATA[127:0] CAD[127:0]	PCI C/BE	PCI_ AD[1:0][1]
Byte	00	00	0000 0001	0000 0001	STL	OW0:[D7:D0]	1110	00
	01	00	0000 0001	0000 0001	STL	OW0:[D15:D8]	1101	01
	10	00	0000 0001	0000 0001	STL	OW0:[D23:D16]	1011	10
	11	00	0000 0001	0000 0001	STL	OW0:[D31:D24]	0111	11
-[3]	11	10	0001 0000	0000 0100	STL	OW1:[D31:D24]	0111	11
-[3]	00	00	0000 0010	0001 0000	STL	OW0:[D39:D32]	1110	00
-[3]	01	00	0000 0010	0001 0000	STL	OW0:[D47:D40]	1101	01
-[3]	10	00	0000 0010	0001 0000	STL	OW0:[D55:D48]	1011	10
-[3]	11	00	0000 0010	0001 0000	STL	OW0:[D63:D56]	0111	11
-[3]	11	10	0010 0000	0100 0000	STL	OW1:[D63:D56]	0111	11
Word	00	01	0000 0100	0000 0010	STL	OW0:[D79:D64]	1100	00
	01	01	0000 0100	0000 0010	STL	OW0:[D87:D72]	1001	01
	10	01	0000 0100	0000 0010	STL	OW0:[D95:D80]	0011	10
-[3]	10	10	0001 0000	0000 0100	STL	OW1:[D31:D16]	0011	10
-[3]	10	11	0100 0000	0000 1000	STL	OW1:[D95:D80]	0011	10
-[3]	00	01	0000 1000	0010 0000	STL	OW0:[D111:D96]	1100	00
-[3]	01	01	0000 1000	0010 0000	STL	OW0:[D55:D40]	1001	01
-[3]	10	01	0000 1000	0010 0000	STL	OW0:[D63:D48]	0011	10
-[3]	10	10	0010 0000	0100 0000	STL	OW1:[D63:D48]	0011	10
-[3]	10	11	10900 0000	1000 0000	STL	OW1:[D63:D48]	0011	10
Tribyte	00	10	0001 0000	0000 0100	STL	OW1:[D23:D0]	1000	00
	01	10	0001 0000	0000 0100	STL	OW1:[D31:D8]	0001	01
-[3]	11	01	0000 0100	0000 0010	STL	OW0:[D95:D72]	0001	01
-[3]	01	11	0100 0000	0000 1000	STL	OW1:[D95:D72]	0001	01
-[3]	00	10	0010 0000	0100 0000	STL	OW1:[D55:D32]	1000	00
-[3]	01	10	0010 0000	0100 0000	STL	OW1:[D63:D40]	0001	01
-[3]	01	11	1000 0000	1000 0000	STL	OW1:[D127:D104]	0001	01
-[3]	11	01	0000 1000	0010 0000	STL	OW0:[D127:D104]	0001	01
Longword	00	11	0100 0000	0000 1000	STL	OW1:[D95:D64]	0000	00
-[3]	11	11	0100 0000	0000 1000	STL	OW1:[D95:D64]	0000	00
-[3]	00	11	1000 0000	1000 0000	STL	OW1:[D127:D96]	0000	00
-[3]	11	11	1000 0000	1000 0000	STL	OW1:[D127:D96]	0000	00
Quadword	11	11	1100 0000	1000 1000	STQ	OW1:[D95:D64]	0000	00

[1]On I/O cycles only.

[2]All other combinations of cWmask[7:0] are invalid and result in a longword write cycle on the PCI where the data is CAD_OW0:[D31:D0] and a null byte mask field.

[3]Sable Implementation specific

Table 39 (Cont.): Sparse Space Write Encodings

EV_ ADDR[6:5] (Offset) CAD[4:3]	EV_ ADDR[4:3] (Size)	cWmask[7:0][2]	CBUS Mask CAD[127:124] CAD[95:92]	Instruction	Valid EV Data EV_ DATA[127:0] CAD[127:0]	PCI C/BE	PCI_ AD[1:0][1]
					OW1:[D127:D96]	0000	00

[1]On I/O cycles only.

[2]All other combinations of cWmask[7:0] are invalid and result in a longword write cycle on the PCI where the data is CAD_OW0:[D31:D0] and a null byte mask field.

5.1.2.2.1.1 PCI0 Sparse Memory Space

PCI0 Sparse Memory Space allows the CPU to access devices residing on either the PCI or EISA buses. Accesses through this region will cause the T2 to generate a PCI Memory command cycle on the PCI Bus. There is 128MB of PCI Sparse memory address space available to the CPU. All accesses to this space follow the rules outlined above for sparse space accesses.

The mapping of this region onto the PCI Bus is software programmable. There is 4GB of available Memory space on the PCI Bus. The HAE0_1 register, bits 4:0, control which 128MB chunk of the 4GB PCI Memory space is used for PCI0 Sparse Memory space. The location of this region in PCI Memory may not overlap with any other spaces that are programmed there as well (ie. DMA windows), except for Dense Memory space. At initialization time, this space is programmed to start at PCI Memory address 8000.0000.

Optionally, the lower 16MB of Sparse Memory space can be rerouted from it's original PCI Memory mapped location by enabling one or both of the T2's PC holes. There are two PC holes. One is fixed at PCI Memory location 512K to 1M and the other is programmable in chunks of 32K from 1M to 16M. With holes disabled, the 128MB of CPU available Sparse Memory space maps directly to a contiguous 128MB of memory space on the PCI Bus. With holes enable, only the upper 112MB of CPU Sparse Memory space maps directly to memory space on the PCI. The other 16MB accesses will be rerouted to the PC holes.

5.1.2.2.1.1.1 PCI0 Sparse Memory Space - PC Holes Disabled

The table below shows how EV addresses are translated to PCI addresses when the PC holes are disabled.

Table 40: PCI0 Sparse Memory Space Address Translation for all 128MB - Holes disabled

Significance	EV_ADDR, Mask	CBUS Address	PCI Address
Decode of Sparse Space	EV_ADDR[33:32]=10	CAD[31:30]	
Sparse Memory Address			PCI_AD[31:27]=contents of HAE0_1

Table 40 (Cont.): PCI0 Sparse Memory Space Address Translation for all 128MB - Holes disabled

Significance	EV_ADDR, Mask	CBUS Address	PCI Address
Sparse Memory Address	EV_ADDR[31:7]=ADDR	CAD[29:5]=ADDR	PCI_AD[26:2]=ADDR
Offset	EV_ADDR[6:5]	CAD[4:3]	Use to generate PCI C/BE[1]
Size Field	EV_ADDR<4:3> on reads, cWmask on writes	CAD[124,92], CBUS_Mask[[7:0]	Used to generate PCI C/BE[1]. PCI_AD[1:0]=00

[1]See tables on Sparse Space encodings for these fields.

5.1.2.2.1.1.2 PCI0 Sparse Memory Space - PC Holes Enabled

The table below shows how EV addresses are translated to PCI addresses when the PC holes are enabled.

Table 41: PCI0 Sparse Memory Space Address Translation for lower 16MB - Holes enabled

Significance	EV_ADDR, Mask	CBUS Address	PCI Address
Decode of Sparse Space	EV_ADDR[33:32]=10	CAD[31:30]	
Sparse Memory Address			PCI_AD[31:27]=00000
Sparse Memory Address	EV_ADDR[31:7]=ADDR	CAD[29:5]=ADDR	PCI_AD[26:2]=ADDR
Offset	EV_ADDR[6:5]	CAD[4:3]	Use to generate PCI C/BE[1]
Size Field	EV_ADDR<4:3> on reads, cWmask on writes	CAD[124,92], CBUS_Mask[[7:0]	Used to generate PCI C/BE[1]. PCI_AD[1:0]=00

[1]See tables on Sparse Space encodings for these fields.

Table 42: PCI0 Sparse Memory Space Address Translation for 16MB to 128MB - Holes enabled

Significance	EV_ADDR, Mask	CBUS Address	PCI Address
Decode of Sparse Space	EV_ADDR[33:32]=10	CAD[31:30]	
Sparse Memory Address			PCI_AD[31:27]=contents of HAE0_1

Table 42 (Cont.): PCI0 Sparse Memory Space Address Translation for 16MB to 128MB - Holes enabled

Significance	EV_ADDR, Mask	CBUS Address	PCI Address
Sparse Memory Address	EV_ADDR[31:7]=ADDR	CAD[29:5]=ADDR	PCI_AD[26:2]=ADDR
Offset	EV_ADDR[6:5]	CAD[4:3]	Use to generate PCI C/BE[1]
Size Field	EV_ADDR<4:3> on reads, cWmask on writes	CAD[124,92], CBUS_Mask[[7:0]	Used to generate PCI C/BE[1]. PCI_AD[1:0]=00

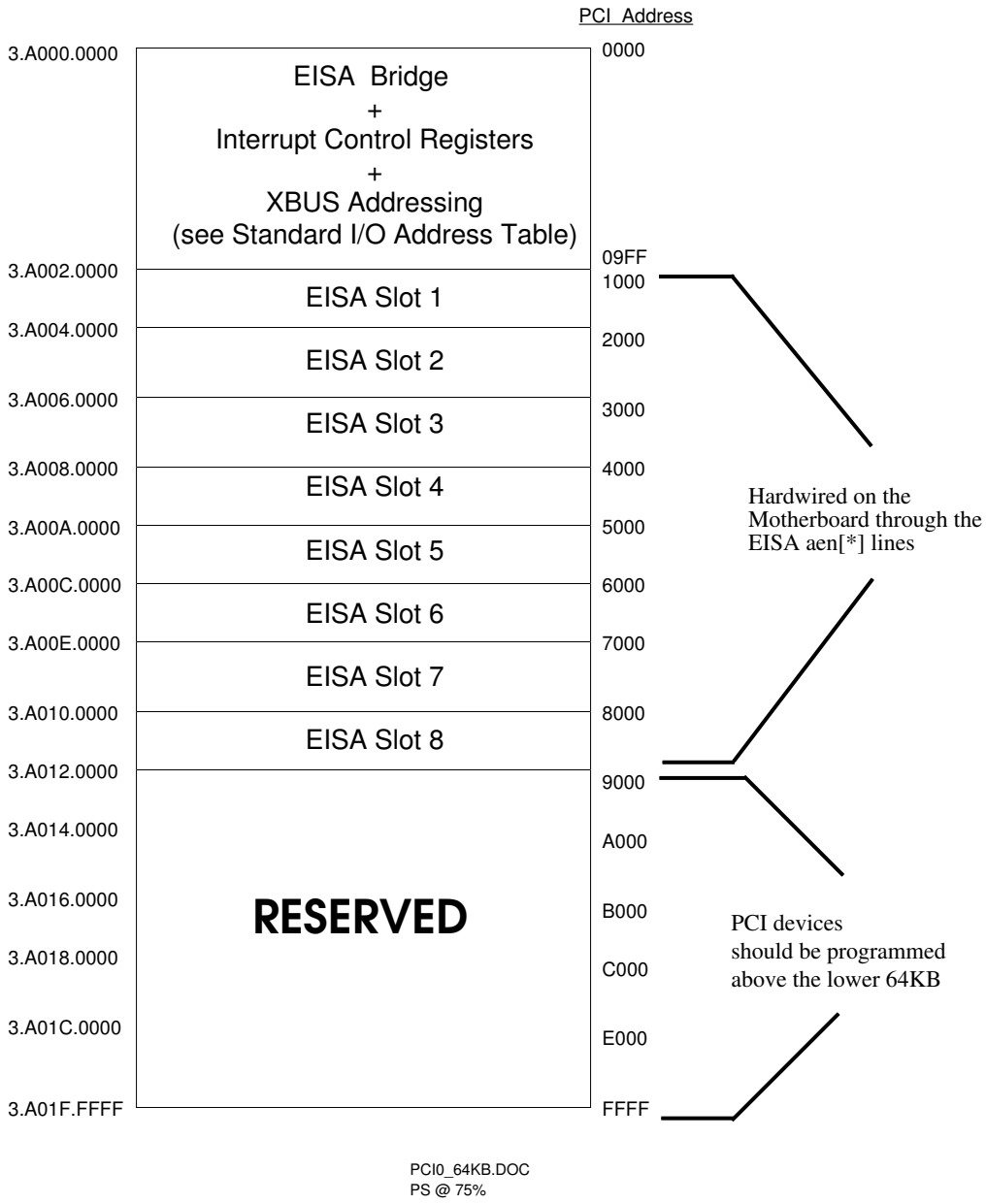
[1]See tables on Sparse Space encodings for these fields.

5.1.2.2.1.2 PCI0 Sparse I/O Space

PCI0 Sparse I/O Space allows the CPU to access devices residing on either the PCI or EISA buses. Accesses through this region will cause the T2 to generate a PCI I/O command cycle on the PCI Bus. There is 8MB of PCI Sparse I/O address space available to the CPU. All accesses to this space follow the rules outlined above for sparse space accesses.
The mapping of this region onto the PCI Bus is partly software programmable. The first 64KB of I/O space available to the CPU is fixed in PCI I/O space and maps directly to PCI address 0. The rest of the I/O space available to the CPU, 64KB to 8MB, can be moved by changing the value loaded into HAE0_2 [8:0].

The lower 64KB should be used to access local PCI devices and EISA/ISA expansion devices. This allows system designers to use device drivers written for other ALPHA systems using PCI,EISA, ISA busses in their systems. However, base addresses of PCI devices are programmable. Figure 43 suggests base addresses for the PCI devices. The PCI/EISA bridge expects its addresses to be start at 0. The EISA devices have been hardwired through the EISA AEN[*] lines on the Motherboard. Their base addresses are selected through the AEN lines.

Figure 43: Suggested Base Addresses for Lower 64KB PCI0 I/O Space



Byte enables are generated according to Table 43 and Table 44. The lower 64KB of this space must be mapped to the lower 64KB of PCI0 Sparse I/O space. The remaining 8MB-64KB can be mapped anywhere in system memory using contents of the HAE0_2 register.

Table 43: PCI0 Sparse I/O Space Address Translation for lower 64KB

Significance	EV_ADDR, MASK	CBUS Address	PCI Address
Decode of Sparse I/O Space	EV_ADDR[33:28]=111010	CAD[31:26]	PCI_AD[31:23]=000000000
Decode of Lower 64KB	EV_ADDR[27:21]=0000000	CAD[25:19]	PCI_AD[22:16]=0000000
	EV_ADDR[20:7]	CAD[18:5]	PCI_AD[15:2]
Offset	EV_ADDR[6:5]	CAD[4:3]	Used to generate PCI C/BE[1]
Size	EV_ADDR[4:3]on reads	CAD[124,92]	Used to generate PCI C/BE[1]
Size	cWmask on writes	CBUS_Mask[7:0]	Used to generate PCI C/BE[1]

[1]Refer to tables Table 38 through Table 39 for encodings of PCI_AD[1:0] and PCI C/BE.

Table 44: PCI0 Sparse I/O Space Address Translation for remaining 64KB-8MB

Significance	EV_ADDR, MASK	CBUS Address	PCI Address
Decode of Sparse I/O Space	EV_ADDR[33:28]=111010	CAD[31:26]	PCI_AD[31:23]=contents of HAE0_2
Decode of 64KB-8MB[1]	EV_ADDR[27:21]=non-zero	CAD[25:19]	PCI_AD[22:16]=CAD[25:19]
	EV_ADDR[20:7]	CAD[18:5]	PCI_AD[15:2]
Offset	EV_ADDR[6:5]	CAD[4:3]	Used to generate PCI C/BE[2]
Size	EV_ADDR[4:3]on reads	CAD[124,92]	Used to generate PCI C/BE[2]
Size	cWmask on writes	CBUS_Mask[7:0]	Used to generate PCI C/BE[2]

[1]Use HAE0_2.

[2]Refer to tables Table 38 through Table 39 for encodings of PCI_AD[1:0] and PCI C/BE.

5.1.2.2.1.3 Standard I/O Address Mapping

Table 55 documents the I/O addresses of the ESC internal registers and the rest of the Standard I/O addresses. Registers are accessed through the EISA interface as 8-bit I/O. The description of the PCEB and ESC registers are found in Intel's 82420/82430 PCIset ISA and EISA bridges data book. Table 55 can be found in Appendix A.

5.1.2.2.2 PCI0 Dense Memory Space

Sable provides 1GB of PCI dense memory space for PCI 0. PCI0 Dense Memory Space allows the CPU to access devices residing on either the PCI or EISA buses. Accesses through this region will cause the T2 to generate a PCI Memory command cycle on the PCI Bus.

The mapping of this region onto the PCI Bus is software programmable. There is 4GB of available Memory space on the PCI Bus. The HAE0_4 register, bits 1:0, control which 1GB chunk of the 4GB PCI Memory space is used for PCI0 Dense Memory space. The location of this region in PCI Memory may not overlap with any other spaces that are programmed there as well (ie. DMA windows), except for Sparse Memory space. At initialization time, this space is programmed to start at PCI Memory address C000.0000.

This space is typically used for data buffers on the PCI and has the following characteristics:

- There is a one-to-one mapping between CPU addresses and PCI addresses i.e. a quadword address from the CPU will map to a quadword on the PCI.
- Byte or word accesses are not allowed in this space. The minimum access granularity is a longword on writes and a quadword on reads. The maximum granularity supported on writes is a cache line, i.e. 32 bytes. Any combination of longwords may be valid on writes. Sable only supports a quadword read in this space.
- Reads will always be performed as a burst of two on the PCI. The processor can request a longword but Sable will always fetch a quadword i.e. prefetch a longword. Hence this space cannot be used for devices which have read side effects.
- Writes to this space are buffered in the 21064 chip. Sable supports a burst length of 8 on the PCI corresponding to a cache line of data. If the cache line written by the processor has holes i.e. some of the longwords have been masked out, the PCI transfer will be performed as a burst of 8 with the masked out longwords being transferred with null byte masks. The address generation in dense space is as in Table 45 and Table 46.

Table 45: Dense Space Address Translation for Reads

Significance	EV_ADDR	CBUS Address, CAD	PCI Address and cycle generated	Data Returned to EV, EV_DATA[127:0], CAD[127:0]
Decode of Dense Space	EV_ADDR[33:30]=1111	CAD[31:28]=1111	PCI_AD[31:30]=11	
	EV_ADDR[29:5]	CAD[27:3]	PCI_AD[29:5]=CAD[27:3]	
	EV_ADDR[4:3]	CAD[124,92]	PCI_AD[4:3]=CAD[124,92]	
			PCI_AD[2:0]=000	
			Burst of two on PCI with C/BE=0000	If CAD[124]=0, data is returned in OW0 on the CBUS.
				If CAD[124]=1, data is returned on OW1 on the CBUS.
				If CAD[92]=0, data is returned on OW0 on the CBUS.

Table 45 (Cont.): Dense Space Address Translation for Reads

Significance	EV_ADDR	CBUS Address, CAD	PCI Address and cycle generated	Data Returned to EV, EV_DATA[127:0], CAD[127:0]
				If CAD[92]=1, data is returned on OW1 on the CBUS.

Table 46: Dense Space Address Translation for Writes

Significance	EV_ADDR	CBUS Address, CAD	PCI Address and cycle generated	Data Driven to EV, EV_DATA[127:0], CAD[127:0]
Decode of Dense Space	EV_ADDR[33:30]=1111	CAD[31:28]=1111	PCI_AD[31:30]=11	
Hexaword Address	EV_ADDR[29:5]	CAD[27:3]	PCI_AD[29:5]=CAD[27:3]	
	EV_ADDR[4:3]		PCI_AD[4:2] generated by T2 sequentially ascending address starting at 000 and ending at 111	
			PCI_AD[1:0]=00	
			Burst of eight on PCI with null byte mask for non-valid longwords.	CAD[127:124,95:92] indicate valid longwords in hexaword. Any combination of write masks can be valid.

5.1.2.2.3 PCI0 Configuration Space

A read or write access to this space causes a PCI Configuration read or write cycle on the internal PCI, PCI0. The usage of the address during PCI configuration cycles varies depending on the intended target of the configuration cycle. There are two classes of targets; devices on the primary PCI bus (type 0) and peripherals on the secondary busses that are accessed via bridge chips like the EISA Bus (Type 1). Configuration space is intended for configuration, initialization and catastrophic error handling functions. Its use should be restricted to initialization software and error handling software. All operational software must continue to use I/O and/or memory space access to manipulated device registers.

The intent of PCI configuration space is to provide an appropriate set of configuration hooks which satisfy the needs of current and anticipated system configuration mechanisms. The criteria for these configuration hooks are:

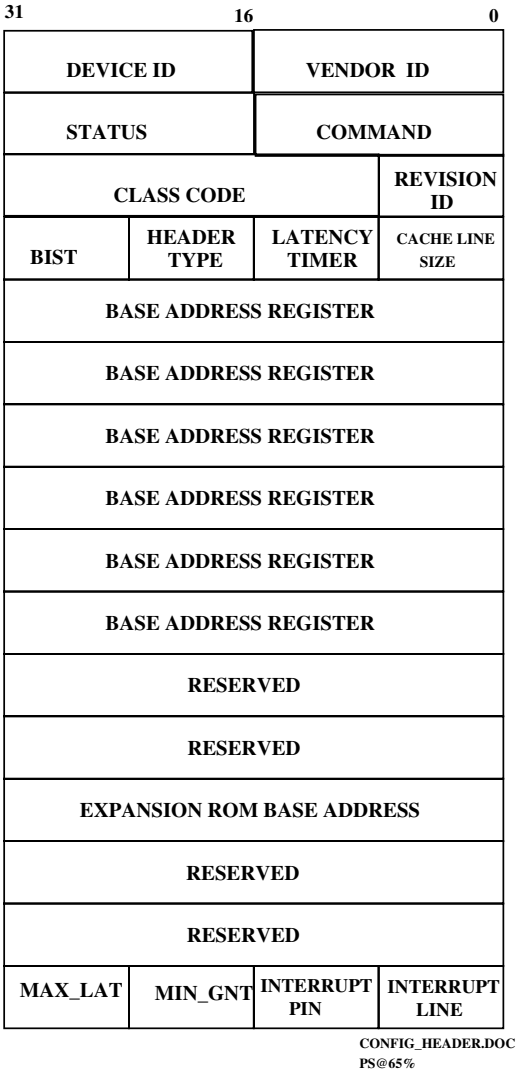
1. Sufficient support to allow future configuration mechanisms to provide:
   - full device relocation, including interrupt binding;
   - installation, configuration and booting without user intervention;

- system address map construction by device independent software.

2. Effective support of existing configuration mechanisms like the EISA Configuration Utility.
3. Leverage commonalty with a template approach to common functions, without precluding devices with unique requirements.

There is predefined organization of configuration space registers and an imposed specific record structure or template on the 256 registers. See the PCI specification for exact implementation. The predefined header region is 64 bytes and every device must support the register layout of this region.

Figure 44: PCI Configuration Header



PCI0 configuration space is further divided by each node on the PCI bus by wiring specific address lines to devices' IDSEL lines. The IDSEL lines are significant in Type 0 Configuration cycles. See Table 47 for the IDSEL lines.

Table 47: PCI0 IDSEL

EV adr	PCI adr	Function	
EV_ADDR[16]	PCI_[11]	Ethernet, TULIP	
EV_ADDR[17]	PCI_[12]	SCSI, NCR 53C810	
EV_ADDR[18]	PCI_[13]	EISA bridge	
EV_ADDR[19]	PCI_[14]	reserved	
EV_ADDR[20]	PCI_[15]	reserved	
EV_ADDR[21]	PCI_[16]	reserved	
EV_ADDR[22]	PCI_[17]	PCI Slot	On Sable, physically connected to PCI Slot 0.
EV_ADDR[23]	PCI_[18]	PCI Slot	On Sable, physically connected to PCI Slot 1.
EV_ADDR[24]	PCI_[19]	PCI Slot	On Sable, physically connected to PCI Slot 2.
EV_ADDR[25]	PCI_[20]	reserved	
EV_ADDR[26]	PCI_[21]	reserved	

Sable supports Type 0 and Type 1 configuration cycles. Two bits are provided in the HAE0_3 register (HAE0_3[31:30]) to identify which type of configuration cycle to run. If HAE0_3[31:30]=00 then type 0 configuration cycles will be performed. If HAE0_3[31:30]=01, then type 1 configuration cycles will be performed. Software must first program this register before running a configuration cycle. If multiple processors are running configuration cycles then this register must be treated as a system resource and appropriate lock mechanisms must be used. See Table 48 for Type 0 Configuration Cycle Generation and Table 49 for Type 1 Cycle generation.

Table 48: Type 0 Configuration Cycle Generation

Significance	EV_ADDR	CAD Address	PCI_ADDR
PCI0 Configuration Space	EV_ADDR[33:27]=1110010	CAD[31:25]=1110010	PCI_ADDR[31:22]=0000000000
PCI0 IDSEL	EV_ADDR[26:16]	CAD[24:14]	PCI_ADDR[21:11]
PCI Device Function #	EV_ADDR[15:13]	CAD[13:11]	PCI_ADDR[10:8]
PCI Device Register	EV_ADDR[12:7]	CAD[10:5]	PCI_ADDR[7:2]
Offset	EV_ADDR[6:5]	CAD[4:3]	Use with size field for PCI C/BE generation according to Table 38 through Table 39.
Size	EV_ADDR[4:3] on reads	CAD[124,92]	
Size	EV_ADDR[4:3] on writes	CAD[127:124,95:92]	
Type 0 Configuraton Cycle			PCI_ADDR[1:0]=00

Table 49: Type 1 Configuration Cycle Generation

Significance	EV_ADDR	CAD Address	PCI_ADDR
PCI0 Configuration Space	EV_ADDR[33:27]=1110010	CAD[31:25]=1110010	PCI_ADDR[31:22]=0000000000
Indicates PCI Bus #[1]	EV_ADDR[26:21]	CAD[24:19]	PCI_ADDR[21:16]
PCI Device #	EV_ADDR[20:16]	CAD[18:14]	PCI_ADDR[15:11]
PCI Device Function #	EV_ADDR[15:13]	CAD[13:11]	PCI_ADDR[10:8]
PCI Device Register	EV_ADDR[12:7]	CAD[10:5]	PCI_ADDR[7:2]
Offset	EV_ADDR[6:5]	CAD[4:3]	Use with size field for PCI C/BE generation according to Table 38 through Table 39.
Size	EV_ADDR[4:3] on reads	CAD[124,92]	
Size	EV_ADDR[4:3] on writes	CAD[127:124,95:92]	
Type 1 Configuraton Cycle			PCI_ADDR[1:0]=01

[1] Up to 64 remote busses can be supported on Sable PCI 0.

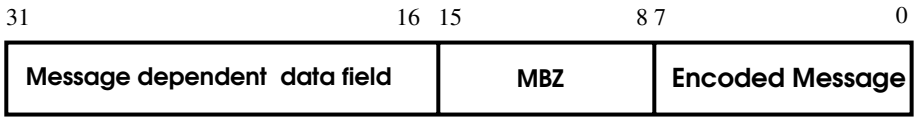
5.1.2.2.4 PCI0 Special Cycles

A write access to 3.8E00.0000 causes a Special Cycle on the PCI. This address is in the T2's CBUS CSR address space. A write to this address causes the T2 to drive 0001 on the PCI C/BE[3:0] indicating a Special Cycle. The data for the address field of the PCI will be driven in the data phase of the EV write cycle and will be passed unmodified to the PCI. Software must write the data in longword 0 of the hexaword. The longword field is used as follows:

- Byte 0 contains the encoded message.
- Byte 1 is reserved and must be written as zeros.
- Bytes 2 and 3 are message dependent data field which is optional. These bytes should be written as zero.

The PCI address field is ignored by PCI devices during Special cycles. The T2 drives all zeros.

Figure 45: PCI Special Cycle

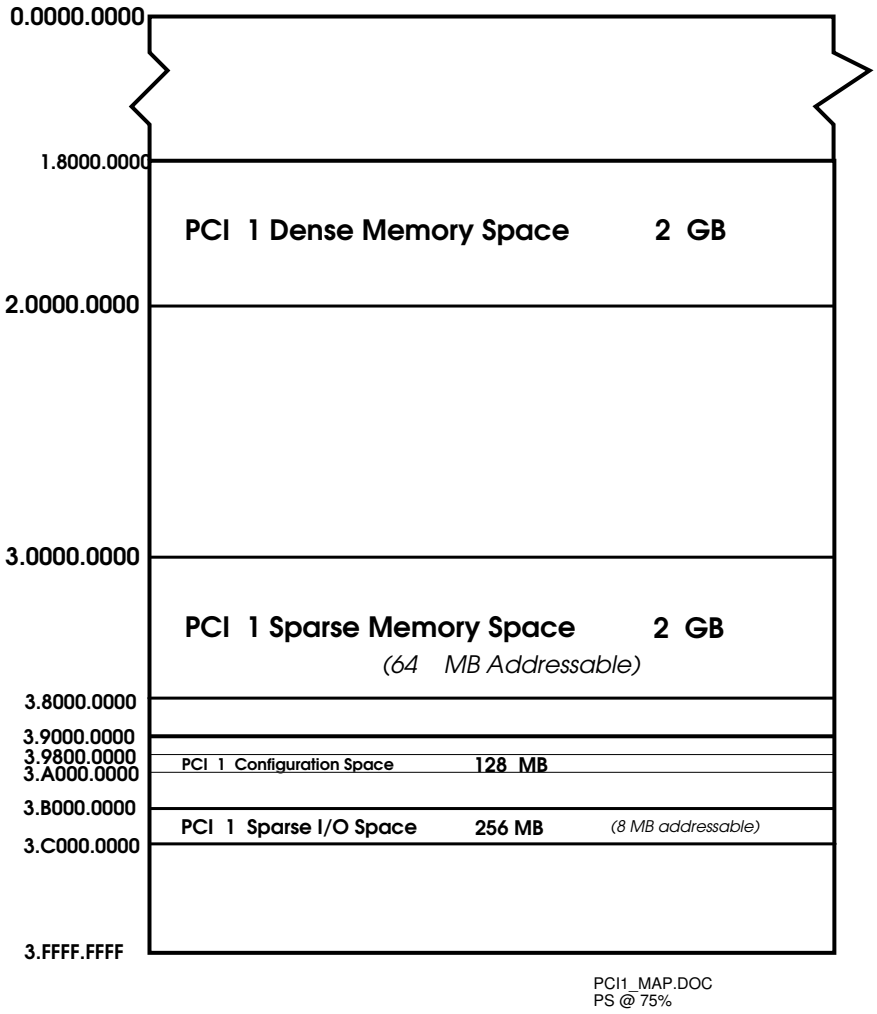


PCI_SPECIAL_CYCLE.DOC
PS@75%

5.1.2.3 PCI1

PCI1 is an optional 64-bit PCI bus that will reside on the XIO module which plugs into the CBUS. The major regions have been defined from the EV perspective and all PCI spaces will be similar to those defined in the PCI0 space. The rest is TBD.

Figure 46: PCI1 Address Map from EV perspective



5.1.2.3.1 PCI1 Configuration Space

PCI1 configuration space has all the same attributes as PCI0 configuration space except that it causes transactions on PCI1. Specific PCI1 device assignments are TBD.

5.2 PCI Address Map

This section describes transactions initiated from devices on the PCI Bus using PCI Memory space and the characteristics of these PCI devices.

5.2.1 PCI Memory Space and DMA

As mentioned previously, parts of PCI Memory space are allocated to accept both Sparse Memory and Dense Memory transactions. In addition, a third part of PCI Memory space can be allocated to map DMA transactions into System Memory so that a PCI device can have direct access to System memory. The programmer maps a section of PCI Memory space into System Memory space by defining a DMA window in the T2. Their are enough registers in the T2 to set up two DMA windows. Each window can additionally be programmed to accept a directed mapped DMA or a scatter/gather mapped DMA. Although Sparse and Dense Memory spaces can overlap in PCI Memory space, neither space can overlap DMA windows, nor can DMA windows overlap themselves. DMA windows can, however, overlap either or both of the enabled PC Holes, but the T2 will not respond to PCI generated traffic from a PCI device that accesses the enabled PC Holes. The section below describes in detail how DMA windows work.

5.2.1.1 DMA Windows

The T2 controls the flow of DMA data initiated from the PCI bus and directed to system memory through the use of DMA windows. In order to correctly enable the two DMA windows, the T2 provides 2 sets of 3 DMA specific registers as specified in the T2's CSR section of this specification. These registers are accessed by the system CPU and are set up during initialization time. Each window has a WBASE register, a WMASK register and a TBASE register. Each register provides a specific function that allows the T2 to complete a DMA transfer correctly.

The WBASE register is used to enable the DMA window, enable the DMA type (direct mapped or scatter/gather mapped), and to size the window. Acceptable PCI DMA addresses are those PCI addresses which fall between and include the address values loaded into the starting and ending address fields located in the WBASE register. The window size is programmable in powers of 2 with the smallest being 1 Mbyte and the largest being 2 Gbytes (For the 2 Gbyte case, Start[31:20]=%B 000000000000 and End[11:0]=%B 011111111111 and the PCI window would directly map into all of the 2 Gbytes of available system memory. Start and End each contain twelve bits and for a PCI address to match in the window the following formula must be true End[11:0] >= PCI Addr[31:20] >= Start[31:20]).

The WMASK register redundantly denotes the size of the enabled window and is provided to the T2 to facilitate certain internal window decode functions. The window size is programmable in powers of 2 with the smallest being 1 Mbyte and the largest being 2 Gbytes (For 1 Mbyte, WMASK[30:20]=%B 00000000000; for 2 Mbytes, WMASK[30:20]=%B 00000000001; for 4 Mbytes, WMASK[30:20]=%B 00000000011; up to 2 Gbytes, where WMASK[30:20]=%B 11111111111). The WMASK register must

contain a value which equals the window size specified in the WBASE register (results are unpredictable if WBASE and WMASK do not specify the same window size).

The TBASE register is a dual purpose register that, for direct mapped DMA's, provides a way to map smaller enabled windows into system memory at different offsets, and for scatter/gather mapped DMA's, provides a way to map the scatter/gather page table into system memory at different offsets. The function the TBASE register provides is determined by the DMA type enabled in the WBASE register for that register set. There are 2 register sets provided such that software has the option of enabling both types of DMA functions, direct mapped and scatter/gather mapped, simultaneously. Only TBASE bits 30:20 are relevant during direct mapped DMA transfers. As the size of the window specified in the WBASE and WMASK registers grows, less TBASE bits are available to provide an offset for the window in system memory. For instance, for a 1 Mbyte direct mapped DMA window, the actual 34-bit EV address produced equals:

— EV ADDR[33:31]=000 (fixed)

— EV ADDR[30:20]=TBASE[30:20]

— EV ADDR[19:5]=PCI ADDR[19:5]

— EV ADDR[4:0]=00000 (fixed)

And for a 2 Gbyte direct mapped DMA window, the actual 34-bit EV address produced equals:

— EV ADDR[33:31]=000 (fixed)

— EV ADDR[30:5]=PCI ADDR[30:5]

— EV ADDR[4:0]=00000 (fixed)

As shown in the above examples, as the window size grows, TBASE offset bits are replaced one at a time by the actual PCI ADDR bits coming from the PCI bus. Of course, to accommodate either of the above windows, the WBASE register would need to be setup with the correct window size values.

When scatter/gather mode is enabled in the WBASE register, the TBASE register provides an offset for the scatter/gather page table which resides in system memory. During scatter/gather DMA's, bits [29:9] of the TBASE register may be used for offset purposes. The number of bits of TBASE used, directly relates to the size of the window specified in the WBASE register, since the size of the page table directly relates to the number of valid PCI addresses accepted by the enabled window. With scatter/gather mode enabled, the T2 uses the incoming PCI ADDR and the TBASE register to access a page table in system memory, which contains Page Frame Numbers that, when concatenated with the original PCI ADDR index (bits PCI ADDR [12:0]), combine to become the actual PCI ADDR sent to system memory. Below is an example of what the EV address would like when accessing the page table in memory for an enabled 1 Mbyte window:

— EV ADDR[33:31]=000 (fixed)

— EV ADDR[30:21]=TBASE[29:20] (fixed)

— EV ADDR[20:10]=TBASE[19:9]

— EV ADDR[9:4]=PCI ADDR[19:14]

— EV ADDR[3:0]=0000 (fixed)

And for a 2 Gbyte scatter/gather DMA window, the EV address generated to access the page table would appear as follows:

— EV ADDR[33:31]=000 (fixed)

— EV ADDR[30:21]=TBASE[29:20] (fixed)

— EV ADDR[20:4]=PCI ADDR[30:14]

— EV ADDR[3:0]=0000 (fixed)

As the window grows, the page table in system memory must grow to accommodate more PFN's and therefore TBASE offset bits are replaced one at a time by the actual PCI ADDR bits.

Sable System Memory provides a maximum of 2 Gbytes of usable storage. The PCI provides the ability to address up to 4 Gbytes of memory. When mapping DMA addresses from PCI to System Memory, only 2 Gbytes of PCI memory space is allowed for DMA transfers. However, the entire 4 GB of PCI Memory space is available to setup the DMA windows. Care must be taken so that DMA windows do not overlap with other spaces (ie. Sparse or Dense) that may already exist in PCI Memory space. In addition, DMA windows can not overlap each other.

EXAMPLE 1: What are the values loaded into WBASE, WMASK and TBASE to enable a direct mapped DMA with an 8 Mbyte window that resides in PCI memory space at address %H 0060.0000 and is mapped into system memory at address %H 0.0C80.0000.

— WBASE[19]=1 (Enable the window)

— WBASE[18]=0 (Direct mapped)

— WBASE[31:20]=%B 0000.0000.0110 (Starting address of window)

— WBASE[11:00]=%B 0000.0000.1101 (Ending address of window)

— WMASK[30:20]=%B 000.0000.0111 (8 Mbyte window)

— TBASE[30:09]=%B 000.1100.1xxx.xxxx.xxxx.xxx (Window offset)

EXAMPLE 2: What are the values loaded into WBASE, WMASK and TBASE to enable a scatter/gather mapped DMA with an 2 Mbyte window that resides in PCI memory space at ad dress %H 0C80.0000 and whose page table is mapped into system memory at address %H 0.0A00.0000.

— WBASE[19]=1 (Enable the window)

— WBASE[18]=1 (Scatter/gather mapped)

— WBASE[31:20]=%B 0000.1100.1000 (Starting address of window)

— WBASE[11:00]=%B 0000.1100.1001 (Ending address of window)

— WMASK[30:20]=%B 000.0000.0001 (2 Mbyte window)

— TBASE[30:09]=%B 000.1010.0000.0000.0000.00x (Page table offset)

5.2.1.1.1 Direct Mapped PCI Address Translation

If the PCI address falls in a direct mapped PCI Target window, the PCI Mask register determines which bits of the Translated base register and the PCI addresses are used to generated the CAD address. Unused bits of the Translated Base Register must be cleared for proper operation.

Table 50: Direct Mapped PCI Address Translation

WMASK[1]	EV_ADDR	Value	Notes
	EV_ADDR[33:31]	=000	
111 1111 1111	EV_ADDR[30:20]	=PCI[30:20]	2GB window
011 1111 1111	EV_ADDR[30:20]	=TBASE[1][30]:PCI[29:20]	1GB window
001 1111 1111	EV_ADDR[30:20]	=TBASE[1][30:29]:PCI[28:20]	512MB window
000 1111 1111	EV_ADDR[30:20]	=TBASE[1][30:28]:PCI[27:20]	256MB window
000 0111 1111	EV_ADDR[30:20]	=TBASE[1][30:27]:PCI[26:20]	128MB window
000 0011 1111	EV_ADDR[30:20]	=TBASE[1][30:26]:PCI[25:20]	64MB window
000 0001 1111	EV_ADDR[30:20]	=TBASE[1][30:25]:PCI[24:20]	32MB window
000 0000 1111	EV_ADDR[30:20]	=TBASE[1][30:24]:PCI[23:20]	16MB window
000 0000 0111	EV_ADDR[30:20]	=TBASE[1][30:23]:PCI[22:20]	8MB window
000 0000 0011	EV_ADDR[30:20]	=TBASE[1][30:22]:PCI[21:20]	4MB window
000 0000 0001	EV_ADDR[30:20]	=TBASE[1][30:21]:PCI[20]	2MB window
000 0000 0000	EV_ADDR[30:20]	=TBASE[1][30:20]	1MB window
	EV_ADDR[19:5]	=PCI_ADDR[19:5]	Pass the address
	EV_ADDR[4:0]	=00000	hardcoded

[1]For each window, there is a WMASK and TBASE associated with it. For Window 1, WMASK1 and TBASE1. For Window 2, WMASK2 and TBASE2. They have the same format. TBASE gives the offset into system memory.

5.2.1.1.2 Scatter/Gather PCI Address Translation

If the SG (Scatter/Gather) bit is set, then the translated address is generated by a table lookup. The incoming PCI address is used to index a table stored in system memory. This is the Scatter/Gather table. The Translated Base Address register specifies the start of the Scatter/Gather Address map table. Bits of the incoming PCI address are used as an offset from the base of the table. The map entry provides the physical address of the page also known as the PTE.

Table 51: Scatter/Gather Map Address Translation for the PTE

WMASK[1]	EV_ADDR	Value	Notes
	EV_ADDR[33:31]	=000	
	EV_ADDR[30:21]	=TBASE[1][29:20]	
111 1111 1111	EV_ADDR[20:10]	=PCI[30:20]	2MB Scatter Gather Map Table Size
011 1111 1111	EV_ADDR[20:10]	=TBASE[1][19]:PCI[29:20]	1MB Scatter Gather Map Table Size
001 1111 1111	EV_ADDR[20:10]	=TBASE[1][19:18]:PCI[28:20]	512KB Scatter Gather Map Table Size
000 1111 1111	EV_ADDR[20:10]	=TBASE[1][19:17]:PCI[27:20]	256KB Scatter Gather Map Table Size
000 0111 1111	EV_ADDR[20:10]	=TBASE[1][19:16]:PCI[26:20]	128KB Scatter Gather Map Table Size
000 0011 1111	EV_ADDR[20:10]	=TBASE[1][19:15]:PCI[25:20]	64KB Scatter Gather Map Table Size
000 0001 1111	EV_ADDR[20:10]	=TBASE[1][19:14]:PCI[24:20]	32KB Scatter Gather Map Table Size
000 0000 1111	EV_ADDR[20:10]	=TBASE[1][19:13]:PCI[23:20]	16KB Scatter Gather Map Table Size
000 0000 0111	EV_ADDR[20:10]	=TBASE[1][19:12]:PCI[22:20]	8KB Scatter Gather Map Table Size
000 0000 0011	EV_ADDR[20:10]	=TBASE[1][19:11]:PCI[21:20]	4KB Scatter Gather Map Table Size
000 0000 0001	EV_ADDR[20:10]	=TBASE[1][19:10]:PCI[20]	2KB Scatter Gather Map Table Size
000 0000 0000	EV_ADDR[20:10]	=TBASE[1][19:9]	1KB Scatter Gather Map Table Size
	EV_ADDR[9:4]	=PCI_ADDR[19:14][2]	Pass the address
	EV_ADDR[3:0]	=0000	hardcoded

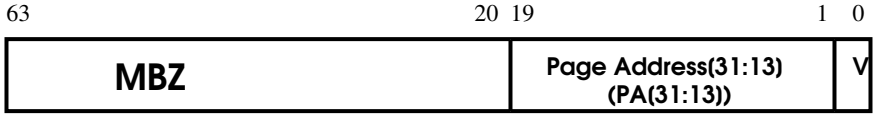
[1]For each window, there is a WMASK and TBASE associated with it. For Window 1, WMASK1 and TBASE1. For Window 2, WMASK2 and TBASE2. They have the same format. TBASE gives the offset into system memory.

[2]A full hexaword is returned for the PTE. The T2 only looks at the first octaword. The T2 saves PCI_ADDR[13] to determine which QW in the octaword is the PTE.

Each scatter/gather map entry maps an 8KB page of PCI address space into a 8KB page of the processor's space. Each scatter/gather map entry is a quadword. Since Sable only implements 2 GB of main memory, bits [63:20] of the scatter/gather map entry must be 0. The map entry allows for future expansion. The size of the scatter/gather map is determined by the size of the PCI Target Window as defined by the Mask Register.

The PTE returned has the following format:

Figure 47: PTE Format



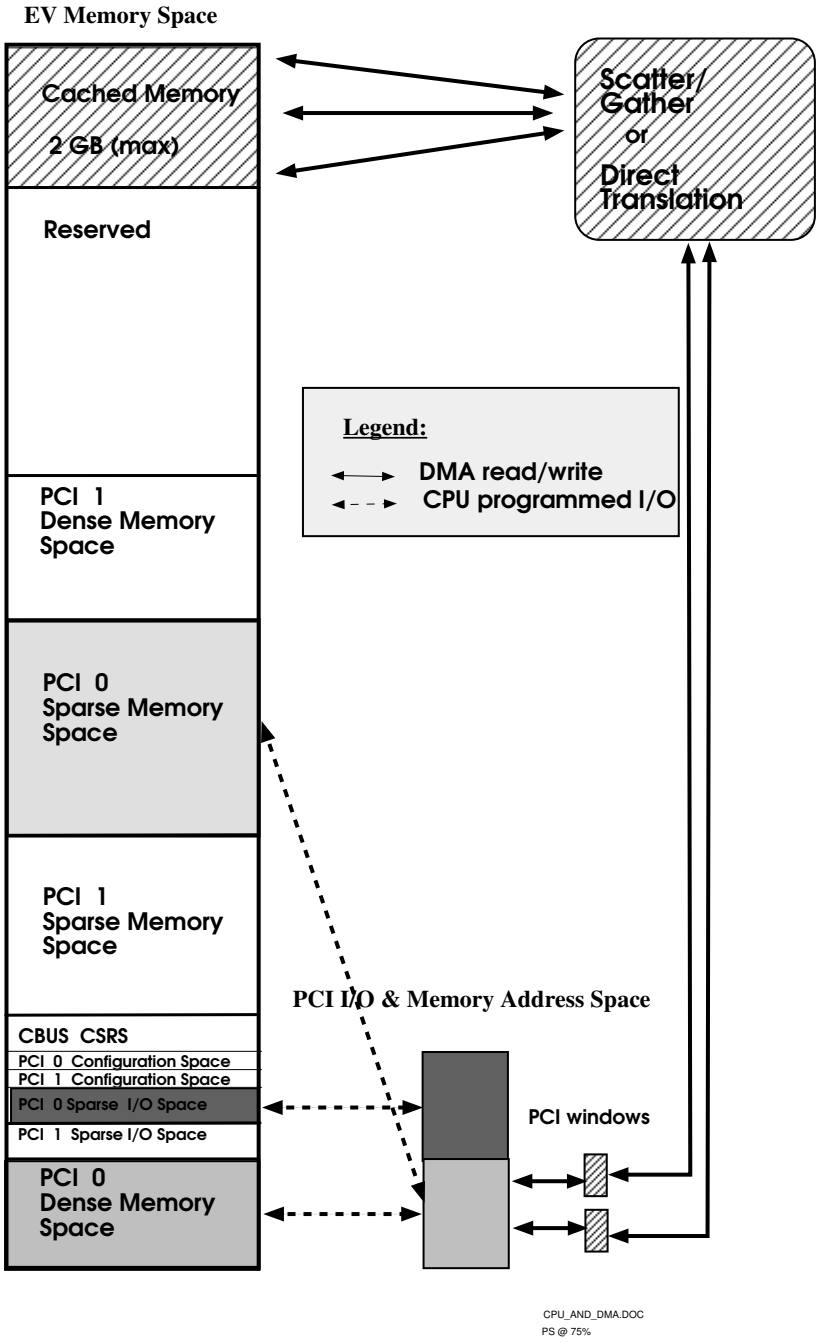
PTE.DOC
ps@75%

Once the PTE is returned, the new CBUS address is:

CAD[31:0]=PA[31:13]:PCI_ADDR[12:0]

EV_ADDR[33:0]=CAD[31:0]:00

Figure 48: CPU and DMA Reads and Writes



5.2.2 EISA Bridge in Sparse Space

PCI memory cycles that are decoded but not claimed (DEVSEL# not asserted), will also be forwarded to the EISA bus in a subtractive decode manner. Reference the PCI-EISA Bridge Component specification (PCEB) for more details.

5.2.2.1 PCI/EISA Bridge

The PCI/EISA bridge decodes PCI addresses in two ways; positive decoding and subtractive decoding. When a particular address is compared and a match is found, the bridge positively decodes the address and the cycle is either serviced internally by the bridge or forwarded to the EISA bus. Subtractive decoding is when no other PCI device claims the cycle by asserting DEVSEL#, the bridge forwards it to the EISA/ISA bus. This involves delay in propagating the cycle from PCI to the EISA bus because the bridge chip must wait for 3 PCI clocks on activation of DEVSEL# before it can claim the cycle and propagate it to the EISA bus. The bridge positively decodes the following addresses based on the contents of relevant configuration registers:

1. PCI configuration registers

The bridge subtractively decodes the rest. Only unclaimed PCI cycles with I/O address range 0 to 64KB will be forwarded to the EISA bus. Unclaimed PCI I/O cycles above 64KB will not be forwarded to the EISA bus and the bridge will not respond with DEVSEL#.

CHAPTER 6

DIAGNOSTIC FEATURES

6.1 T2 Loopback

This chapter describes the T2 loopback feature. A minimum of four registers need to be setup to do a loopback. WMASK1 determines the size of the first window. WBASE1 enables the window and gives the starting and ending register. TBASE1 specifies the base CPU address of the translated PCI address if direct mapped. The IOCSR needs to have the PCI memory space enabled and should have CBUS parity checking enabled also.

When IOCSR[1] is set, the T2 is in internal loopback mode. This is a diagnostic mode which allows PCI I/O space reads and writes to be looped back from/to system memory.

6.1.1 T2 Loopback Example

```
WMASK1:
 dep -si0 11 0 3ff00000  - 1: size of first window (1 gig window)
        111

WBASE1:
 dep -si0 10 0 000807ff  - 1: window address start at 0
                      1112 333    2: enable window
     3: end end address (2 gig window)

IOCSR:
 dep -si0 0 0 22000000   - 1: enable cbus parity checking
                     12           2: enable PCI memory space

TBASE1:
 comes up as zero

    To do a simple loopback test:

     dep -si0 0 0 22000002 - the last 2 enables T2 loopback
            mode

     dep -p [addr + mask] [data]

     example:

                   dep -p 078 a5a5a5a5

                   e 0
                     returns: a5a5a5a5
```

CHAPTER 7

ERROR

7.1 Errors

The T2 contains logic which detects and reports error conditions that occur on either the CBUS or the PCI Bus. The errors detected by the T2 are listed in Figure 49 and are classified as unrecoverable errors. Each error listed can be enabled or disabled. When an error is disabled, the T2 ignores the particular error completely (no logging, no interrupt). When an error is enabled, the T2 will log the error, store the address of the failing transaction, and generate an interrupt to the Host CPU system. All errors are disabled at power-up and therefore must be enabled by software.

7.2 MBA Bits

The MBA lines are general purpose input lines that allow an event to be logged and view through reading the IOCSR register in the T2. However, the T2 does not generate any interrupt on behalf of an event happening. There are eight MBA bits, as follows:

MBA Bit	IOCSR Bit	Description
0	4	PCI 0 Present
1	5	PCI 0 Present
2	18	PCI 1 Present
3	19	PCI 1 Present
4	39	PCI 2 Present
5	15	PCI 2 Present
6	16	Power Supply Error
7	17	Reserved

I am told that the power supply failure occuring will cause SYS_EVENT to be asserted by the STD I/O module. 1 The Power Supply failure line is connected to one of the 8 MBA lines that are inputs to the T2 (IOCSR_16, MBA<6>). So if you want, you could add in the Error area of the I/O Spec. a section that discribes what errors cause SYS_EVENT and list them. But I don't think that it should be added to the table that already exists because those errors are reported through CERR. I'll check back with you later, Ciao

Figure 49: T2 Error Detection

Bus Type	CSR BIT	ENA BIT	Error
CBUS	CERR1_0	IOCSR_24	UNCORRECTABLE READ
CBUS	CERR1_1	IOCSR_24	NO ACKNOWLEDGE
CBUS	CERR1_2	IOCSR_29	COMMAND/ADDRESS PARITY
CBUS	CERR1_3	IOCSR_29	MISSED COMMAND/ADDRESS PARITY
CBUS	CERR1_4	IOCSR_29	RESPONDER WRITE DATA PARITY
CBUS	CERR1_5	IOCSR_29	MISSED RSP WRITE DATA PARITY
CBUS	CERR1_6	IOCSR_29	READ DATA PARITY
CBUS	CERR1_7	IOCSR_29	MISSED READ DATA PARITY
CBUS	CERR1_8,40,9,41	IOCSR_29	CA PARITY ERROR LW 0, 1, 2, 3
CBUS	CERR1_10,42,11,43	IOCSR_29	DATA PARITY ERROR LW 0, 1, 2, 3
CBUS	CERR1_12,44,13,45	IOCSR_29	DATA PARITY ERROR LW 4, 5, 6, 7
CBUS	CERR1_16	IOCSR_24	CMDR WRITE DATA PARITY
CBUS	CERR1_17	IOCSR_24	BUS SYNCHRONIZATION
CBUS	CERR1_18	IOCSR_26	INVALID PFN
PCI	PERR1_0	IOCSR_63	WRITE DATA PARITY
PCI	PERR1_1	IOCSR_62	COMMAND/ADDRESS PARITY
PCI	PERR1_2	IOCSR_61	READ DATA PARITY
PCI	PERR1_3	IOCSR_60	PERR
PCI	PERR1_4	IOCSR_59	SERR
PCI	PERR1_5	IOCSR_58	DEVICE TIMEOUT
PCI	PERR1_6	IOCSR_57	NMI

ERROR_TABLE.DOC
ps @65%

1

CHAPTER 8

STANDARD AND REMOTE I/O MODULES

8.1 Introduction

The Sable Standard IO Module provides bridges between the PCI bus and the Ethernet, internal SCSI bus, and internal EISA bus. It also provides standard PC IO functions (keyboard, mouse, serial, parallel, and floppy port controllers), and miscellaneous system required functions. The Remote IO module provides physical mounting and electrical isolation for the keyboard and mouse, speaker, parallel, serial, and floppy ports.

The Standard IO module is comprised largely of third party components, with very little custom logic. The Remote IO is comprised entirely of passive components and connectors. This document will supplement the specifications provided by the vendors of each of the components on the Standard IO and Remote IO.

The Mechanical Drawings for the Standard and Remote IO are available on MDA.

8.2 Standard IO - PCI to other busses

8.2.1 PCI to Ethernet

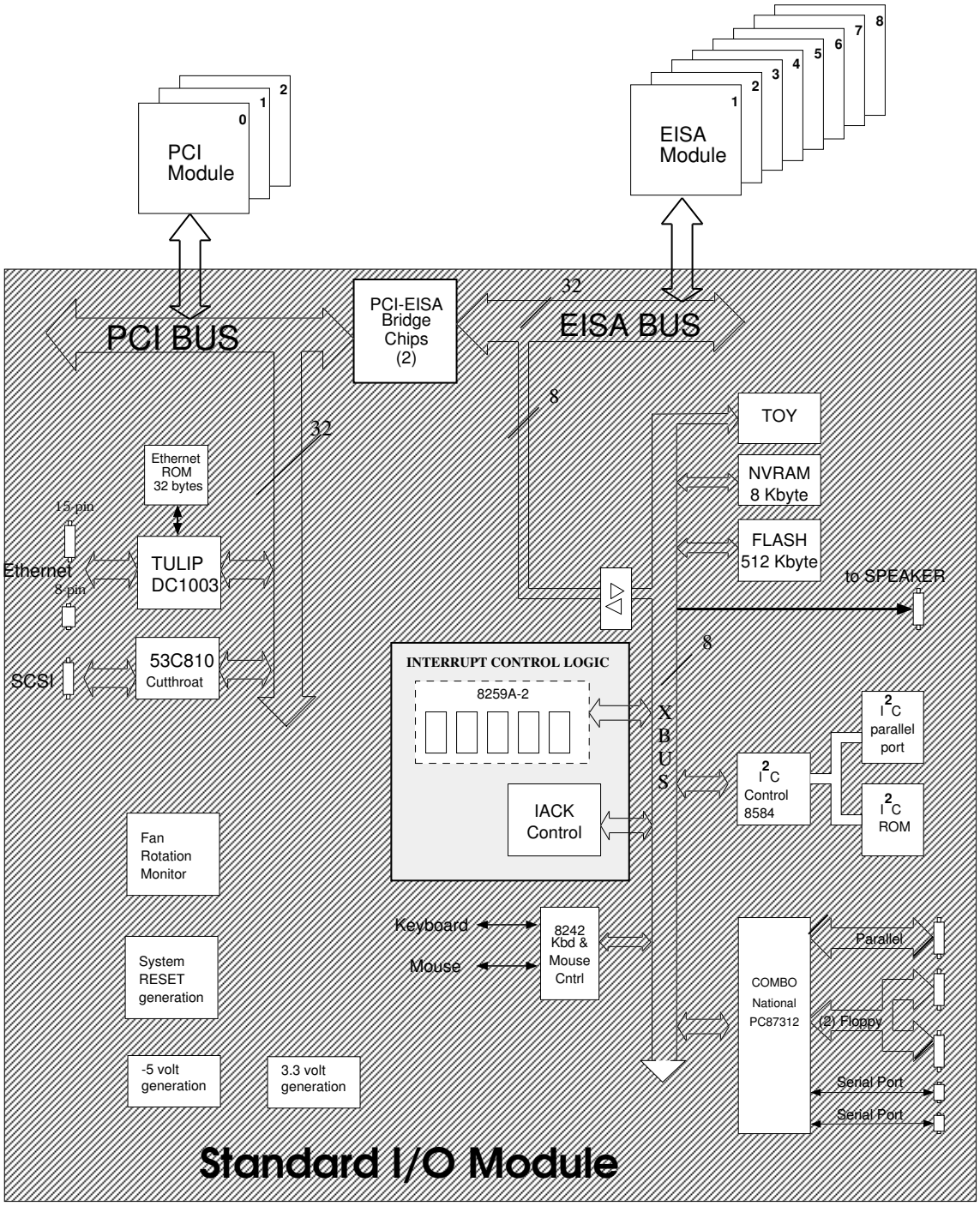
The bridge between the PCI and the Ethernet is provided by the Tulip (DC1003) chip, which is manufactured by Digital. For more information on this component, see the Tulip specification.[1]

There is a small amount of peripheral logic associated with the Ethernet interface. This includes:

- A 20 MHz crystal with associated passive components for tuning.
- Bypass capacitors as recommended by Tulip developers, located in close proximity to the Tulip IC.
- Passives for edge conditioning on the thickwire signals.
- Standard thinwire interface, with a buffer used to combine differential signals and a 10BaseT transformer with filters and balun.
- Two LEDs which are placed next to the thickwire and twisted pair connectors. One of the two will always be lit, indicating which interface is selected. The default selection is twisted pair.
- A thermistor fused source of +12 Volts sourced to the thickwire connector.

[1] A pointer to the current revision of the Tulip specification is available in the PCI notesfile. Contact Todd Commons at TECRUS::COMINS or Jim Pappas at TECRUS::JIM for membership.

Figure 50: Standard IO Block Diagram



STD_IO_BLOCK2.DOC
PS @ 75%

- Serial Ethernet ID ROM, supported by integrated Tulip logic.

8.2.2 PCI to SCSI

The PCI to SCSI bridge is provided by the NCR 53C810 (Cutthroat). The only external logic required is a 40 MHz oscillator input, and a thermistor protected source of SCSI termination power. The NCR 53C810 connects directly to a standard 50 pin SCSI header, which is located on the edge of the Standard IO module. Please see the NCR 53C810 PCI-SCSI IO Processor Data Manual for the IC component specification. [2]

8.2.3 PCI to EISA bridge

The PCI to EISA bridge is provided by Intel's ESC (82374EB) and PCEB (82375EB) components. Documentation for these components is in Intel's manual titled 82420/82430 PCIset ISA and EISA Bridges. [3] The ESC and PCEB also control an 8-bit buffered slave ISA bus, the XBUS, which is described in Section 8.3.

A small amount of additional logic is required to provide the complete function of the PCI to EISA bridge, as follows:

- The EISA address bus requires a external latching buffer. This is implemented on the Standard IO with F543 parts, which are controlled by the ESC.
- External logic is required to decode the EISA MACK signals (F138's) from the encoded version provided by the ESC, and to generate the EISA AEN signals. [4]
- The ARB PAL, which provides external sub-arbitration between the NCR 53C810 and PCI Slot 2 on the Motherboard. This is required because the internal PCI arbiter in the PCEB does not provide enough resolution.

For more information on EISA, consult the EISA specification [5] and the book ISA & EISA Theory and Operation by Edward Solari.

For more information on PCI, consult the PCI specification. [6]

8.3 Standard IO - The XBUS

The XBUS is an 8-bit buffered ISA slave bus, which is accessible from either the EISA or the PCI bus. The components on the XBUS provide slow memory, interrupt control, and common PC IO ports for the SABLE system.

XBUS data is buffered from the lower 8 bits of the ISA/EISA data bus by an ALS245 which is controlled by the ESC. The 17 lower address bits and IO and memory read and write signals are also buffered, to lessen the loading on the sources of these signals. This is done with ALS244 buffers.

---

[2] Note 13 of the PCI notesfile has a pointer to a Hudson, MA NCR sales contact for the NCR 53C810 documentation. Contact Todd Comins at TECRUS::COMINS or Jim Pappas at TECRUS::JIM for membership to the notesfile.

[3] Contact the Intel Literature Center, at phone number (800) 548-4725 if you are within the United States. The order number for the required document is 290483-001.

[4] The ESC generates an encoded AEN which we could simply decode, but due to bugs in the Intel chips, it was easier to generate the signals explicitly from the EISA bus in PAL EBHC.

[5] The Standard IO was built using EISA specification version 3.12, provided by BCPR Services, INC.

[6] A pointer to the current revision of the PCI specification is available in the PCI notesfile. Contact Todd Commons at TECRUS::COMINS or Jim Pappas at TECRUS::JIM for membership.

The ESC performs address decoding for the devices on the XBUS. The ESC provides decoded chip selects and read/write control for supported components, as necessary. Most of the chip selects are device specific and have a set address range associated with them. In addition, there are three general purpose chip selects, GPCS<2:0>. The user can specify an address range for each GPCS, using a base and mask register (one each per GPCS) in the ESC. There are more than 3 slave devices on the XBUS which the Standard IO requires additional chip selects for, so GPCS<0> addresses a range of addresses which include several devices. Further decoding is provided by PAL SIOA.[7]

For more information on XBUS operation, see the ESC data book.

8.3.1 XBUS—Memory Devices

There are two types of slow memory devices controlled by the XBUS. An ALS245 buffers data to and from this memory, decoupling it from the XBUS data lines. The buffered data bus is called *bdata*. The buffer is used to avoid excess loading of the XBUS data bus. The enable for this buffer is generated by logically 'or'ing the memory chip selects in PAL SIOB. [7] The memory located on *bdata* includes the Configuration RAM, and the BIOS ROM. Some proto-type modules will also have a parallel Ethernet Station ID ROM on this bus. This was used before the Tulip chip directly supported a serial ROM.

8.3.1.1 Configuration RAM

Configuration RAM on Sable is implemented in 8KB of Non-Volatile Memory (NVRAM). The physical device used is an EEPROM. It is used primarily for storage of EISA/PCI Configuration Data, and Environment Variables. If there is any room is left, a user-defined powerup script such as the one used on Cobra will also be stored here.

Reading and writing of the Configuration RAM is directly controlled by the ESC. The address range is defined in the ESC specification, and is also noted in Appendix A.

The NVRAM will be installed in a socket, and may be transferred to a new module in the field, to preserve EISA configuration data during FRU replacement.

8.3.1.2 BIOS ROM

BIOS ROM is used on the Standard IO to store the system console. BIOS ROM is implemented with 1 MByte of FLASH ROM, but the user may only access one of eight 128 KB contiguous sections at a time. The FLASH is logically divided into two 512 KB "banks", with one bank containing the System Console, and the other (alternate) bank containing the Failsafe Console and the ARC Console.

There are three bits that select which of the eight sections of memory the user can access. The bits reside in the ELCR[1] register, at locations 5, 6 and 7. Bit 7 of ELCR[1] selects which "bank" of FLASH is accessed. Bit 7 may also be set by installing a jumper located on the edge of the module which is visible when the module is installed in a system. It will be obvious which jumper this is, as it is the only right angle jumper.

[7] All PAL code is available on MDA.

Bits 5 and 6 of ELCR[1] select among the 4 sections of memory contained in each bank, and act similar to upper order address bits for the 512KB space, except they must be set in the ELCR[1] register, rather than being driven onto the address bus during the cycle.

ELCR[1][7]	ELCR[1][6]	ELCR[1][5]	Selection
0	0	0	System Console - First Quarter
0	0	1	System Console - Second Quarter
0	1	0	System Console - Third Quarter
0	1	1	System Console - Fourth Quarter
1	0	0	Failsafe Console - First Quarter
1	0	1	Failsafe Console - Second Quarter
1	1	0	ARC Console - First Quarter
1	1	1	ARC Console - Second Quarter

It is purposefully awkward to overwrite the System Console and the ARC Console. In order to program these areas of the BIOS memory, the user must enable the write to FLASH in the appropriate ESC register. In addition, the user must set the voltage enable bit (vpon) on the Standard IO, which is in the ELCR[1] register, at location 4. The Failsafe Console also requires a jumper (J5) to be installed in order to connect programming voltage. The user may disable writes to both banks of the FLASH permanently at any time, as required by military applications, by removing the resistor, R143, which connects VPEN L to VPP L.

The memory address range for the 128 KB section is from 000E0000h to 000FFFFFh.

8.3.2 XBUS—Interrupt Control Logic

The XBUS Interrupt Control Logic provides interrupt handling for the Sable IO subsystem. It consists of a set of cascaded Intel 8259A-2 Interrupt Controllers, and two PALS which provide an interface between the XBUS and the 8259A-2's.

The 8259A-2's are arranged in a standard Master-Slave cascade, with one Master and four Slave devices. The details on an 8259A-2 are available in the Intel Peripheral Databook. The priority and assignments of the various interrupts are detailed in Section 3.2.3.

Name	IO Address	Bit(s)	Interrupt
Master	0534-0535	7-5,2	Not used
Master	0534-0535	5,4,3,1	Slave<3:0> Cascade In
Master	0534-0535	0	ESC interrupt
Slave3	053E-053F	7-5	PCI Slot<2:0> D
Slave3	053E-053F	4-2	PCI Slot<2:0> C
Slave3	053E-053F	1-0	PCI Slot<1:0> B
Slave2	053C-053D	7	I^2C Controller
Slave2	053C-053D	6,5	EISA <15:14>

Name	IO Address	Bit(s)	Interrupt
Slave2	053C-053D	4	PCI Slot 2 B
Slave2	053C-053D	3-0	EISA <12:9>
Slave1	053A-053B	7,0	Serial Port <1:2>
Slave1	053A-053B	6-2	EISA <7:3>
Slave1	053A-053B	1	Parallel Port
Slave0	0536-0537	7	Floppy Disk Controller
Slave0	0536-0537	6	Keyboard
Slave0	0536-0537	5,4,0	PCI Slots <2:0> A
Slave0	0536-0537	3	Mouse
Slave0	0536-0537	2	DC1003
Slave0	0536-0537	1	NCR 53C810

The 8259A-2's act as EISA slaves for CSR accesses. PAL SIOA[7] uses General Purpose Chip Select 0 from the ESC and the lower 4 bits of ISA addresses to generate individual device selects for each 8259A-2.

8.3.2.1 IACK Cycles

Obtaining the interrupt vector from the 8259A-2's requires a double pulse on a signal called INTA, upon the second of which the appropriate slave will drive the vector onto the bus. There is no explicit XBUS cycle which will do this. Instead, a read to XBUS address 0032h is interpreted as a request for a vector. PALS SIOD and SIOE [7] work together to produce the double pulse.

The time it takes to generate the double pulse, which is dictated by the timing parameters of the 8259A-2's, is so long that the ISA bus must be held off, else the data will be missed. The same state machine that generates the INTA signal also forces ISA bus signal IOCHRDY (IO Channel Ready) into a de-asserted state. The de-assertion of this signal, in accordance with the ISA specification, will signal to the ISA bus that the read data is not ready. This signal is held de-asserted until the proper time relative to INTA generation, when 8259A-2 timing parameters guarantee a vector will be present on the bus. The release of IOCHRDY allows the signal to assert, which signals the ISA bus that data is present for the read, and the vector is returned.

8.3.2.2 ELCR logic

The 8259A-2 controllers recognize either low-to-high edge or high-level triggered interrupts, and the choice must be consistent on all inputs to a controller. The controllers on the Standard IO are configured to handle edge-triggered interrupts. However, the Standard IO Interrupt logic supports both PCI and EISA low-level triggered interrupts, and EISA and ISA low-to-high level triggered interrupts.

PCI interrupts are all low-level triggered. However, EISA interrupts may either be low-level triggered or low-to-high edge triggered. The user must indicate which kind of interrupt is being used for installed options. This is done via two 8-bit registers, given the standard EISA name of ELCR's. Setting these bits indicates to

the Standard IO that the corresponding EISA interrupt is low-level triggered. The following table indicates which ELCR bit corresponds to each EISA interrupt.[8]

CSR	Bit	Signal
ELCR[0]	0	xbus.irq.e.l3
ELCR[0]	1	xbus.irq.e.l4
ELCR[0]	2	xbus.irq.e.l5
ELCR[0]	3	xbus.irq.e.l6
ELCR[0]	4	xbus.irq.e.l7
ELCR[0]	5	xbus.irq.e.l9
ELCR[0]	6	xbus.irq.e.l10
ELCR[0]	7	xbus.irq.e.l11
ELCR[1]	0	xbus.irq.e.l12
ELCR[1]	1	xbus.irq.e.l14
ELCR[1]	2	xbus.irq.e.l15
ELCR[1]	4	BIOS - vpon
ELCR[1]	5	BIOS - xbus.a[17]
ELCR[1]	6	BIOS - xbus.a[18]
ELCR[1]	7	BIOS - xbus.~sab

The ILHA and ILHC PALs[7] will use the ELCR and corresponding interrupt lines as input. They will condition any level triggered interrupt to look like an edge triggered interrupt, by reversing the polarity and causing a second edge if the interrupt is still asserted at the end of an interrupt handling cycle. Thus, shared interrupts will not be missed by the edge triggered controllers.

8.3.3 XBUS - Common PC IO Devices

8.3.3.1 Keyboard and Mouse Control

Keyboard and mouse control is provided by the Intel 8242. This component is an Intel processor with resident firmware written by Phoenix Technologies, LTD. Control signals for the processor are provided by the ESC. The controller is clocked by an inverted version of the 8.33 MHz EISA BCLK. See Intel's Peripheral Data book, the ESC specification, and the Phoenix MultiKey/42 Technical Reference for more details on the operation and control of the keyboard and mouse controller.

The keyboard interrupt is brought through PAL SIOD[7]. For the Sable platform, no conditioning is required, and the logic is provided for future flexibility. The TurboLaser platform is using the internal interrupt controller in the ESC, and the mouse is connected to an inverting output. This interrupt is inverted in SIOD. See Section 8.4.8 for more information on how the Standard IO will be used in TurboLaser.

[8] Bits 7-4 of ELCR[1], noted in the table, provide BIOS functions. Bits 7-5 are used to set the bank and a[18:17] for the FLASH ROM. Bit 4 is used to enable the programming voltage for the FLASH ROM.

8.3.3.2 Serial, Parallel, and Floppy Ports

Sable's two serial, one parallel, and two floppy ports are provided by the National Semiconductor PC87312 SuperI/O. The Floppy Disk Controller (FDC) has a 16 Byte FIFO, and the Serial Port has a configurable FIFO. The FIFO configuration in the SuperI/O is different then the one on Jensen. External logic required to service these ports includes:

- A 24 MHz oscillator.
- An electrically isolated analog power supply.
- RS232 TTL to +/- 12V level converters for the serial ports.
- Capacitors for edge shaping of the parallel port signals.
- External pullup on the CFG0 line, for hardware configuration, as per the specification.

See the National Semiconductor PC87311/PC87312 specification for more details on the operation of this component.

The signal lines for the floppy ports are wired to a standard 34 pin header for connection to a floppy bus. The other ports are connected to the 60-pin Remote IO header, which allows them to connect to the Remote IO module. The external connectors for the ports are mounted on the Remote IO module. See Appendix B for more details on the connectors and Section 8.5 for more details on the Remote IO Module.

8.3.3.3 Real Time Clock

The Real Time Clock is the Dallas Semiconductor DS1287, which includes an internal battery. The square wave output is buffered by an F125.

The Real Time Clock contains 50 Bytes of user RAM, 10 bytes of RAM for the RTC time, calendar, and alarm data, and 4 bytes which are used for control and status. Access to the DS1287 is controlled by the ESC via index and data ports 70h and 71h. The user must write the desired DS1287 address to the lower 6 bits of the ESC Port 70h. Bit 7 of this location is the NMI enable bit, and its integrity must be preserved in the write. The user may then read or write ESC Port 71h to transfer the desired data.

See the Dallas Semiconductor DS1287 and ESC specifications for more detail on Real Time Clock operation.

8.3.3.4 Speaker

The speaker logic consists of two signals and a standard PC speaker. One is a constant 5 volt level, for reference. This is implemented as a pair of pullup resistors, with the redundancy providing over-current protection. The other signal is generated as a pulse of varying frequency by the ESC, via internal Timer 1, Counter 2 and Port 61h.

In order to program the frequency desired and to turn on the speaker, the user must write the value B6 to register 0043h. The user must then write the low and then high bytes of a count to register 0042h. The frequency broadcast by the speaker will

be 1.193MHz divided by the count. To turn the speaker off, the user should disable it via port 0061h.

The speaker is mounted on the back of the enclosure, with a Second four pin header connecting the two signals from the Remote IO module. See Section 8.5 for more details on the Remote IO module, and the ESC specification for more details on the control of the speaker signal.

8.4 Standard IO - Miscellaneous

8.4.1 I²C System Control Bus Devices

The Standard IO module provides three I²C System Control Bus components, as follows:

8.4.1.1 Controller

The I²C Controller, the PCD8584, allows firmware access to the I²C System Control Bus via the XBUS. The chip select for the XBUS side of the interface is provided by PAL SIOA[7].

8.4.1.2 I²C E²PROM

The I²C E²PROM, the PCF8582, is used to store the Standard IO module serial number and system identification. It can also be used by firmware or software for the storage of error logging information.

8.4.1.3 Expander

The I²C Expander, the PCF8574, is an 8 bit parallel to serial interface for the I²C bus. It is used to communicate system status to the CPU via the I²C bus. All signals are asserted low, and are described in the following table:

Bit	Name	Description
7	fanfail l	Indicates the fan speed has dropped to an unacceptable level.
6	rMode l	Indicates the power supplies are in redundant mode.
5		Not used
4	ps.pok1 l	Indicates power supply 1 is good. Defaults to good if the supply is not installed.
3	ps.pok2 l	Indicates power supply 2 is good. Defaults to good if the supply is not installed.
2		Not used
1	ioprsnt l	Pulled high in TurboLaser systems
0	halt l	Shadows the halt button. *Note: This signal will not be detectable, since it is a one-shot. The latched version is available on the I²C bus on the OCP.*

8.4.2 OCP Interface

All OCP signals are brought to the Standard IO. Some are used as described in this specification, and others are passed through the fingers directly to the Motherboard. See Appendix B for a list of what signals comprise this interface.

8.4.3 Reset Circuitry

The Standard IO Module produces async reset from a wired 'or' of the ~ocpReset signal and the dcok signal, from the OCP and power supply, respectively. This signal is brought to the fingers and sourced to the Motherboard.

The Standard IO Module receives PCIreset as an input. This signal is buffered to generate individual resets for the PCI slots, the EISA bridge components, and the Ethernet and SCSI bridges.

8.4.4 Fan Speed Monitor

The Fan speed monitor consists of two complimentary functions. A dual timer is used to provide an extended hold-off signal upon power-up. A re-triggerable one-shot uses the fan tachometer pulse to signal a fan speed failure if the fan speed drops below 500 RPM. The fan failure signal is only latched when the power-up holdoff is de-asserted.

8.4.5 Voltage Generation

The Standard IO module generates the following voltages:

- -5 Volts, as required by EISA. This is fused with a thermistor.
- 3.3 Volts, as required by the PCI to Ethernet bridge.

8.4.6 Remote System Monitor Option

The Remote IO provides a connector to be used by the Remote System Monitor Option. All signals are used on a Read-Only basis. The option will be part of the Server management solution that AVS will be developing either alone, or with DELL. See Appendix B for more details on what signals are provided on the connector.

8.4.7 Jumpers

The Standard IO Module contains three jumpers, as follows:

RD	From	To	Function
J3	gnd	rMode l	Install jumper to indicate redundant mode power supply
J5	avp	vpp	Install jumper to enable writes to Failsafe Console section of FLASH memory
J6	xbus.sab	gnd	Install jumper to select alternate bank of FLASH[1] memory

[1]J6 for test-rack use only. Enclosed systems will use an automatic software switch.

8.4.8 TurboLaser

The Standard IO Module will be used in TurboLaser. TurboLaser will use the Standard IO Module differently than Sable, as follows:

- TurboLaser will use the internal interrupt controllers in the ESC for EISA interrupts, and will handle PCI interrupts on the Motherboard.
- TurboLaser was planning on having a PCI arbiter on the Motherboard, and not using the one resident in the PCEB, or the sub-arbiter in the ARB PAL. However, a late specification change for the PCEB eliminated support of the external arbiter. Thus, TurboLaser will be using the internal PCEB arbiter.

The following table details the connection of the EISA interrupts to the ESC's internal interrupt controllers:

ESC Controller Input	Signal
IRQ15	EISA IRQ15
IRQ14	EISA IRQ14
IRQ13	pulled up[1]
IRQ12	EISA IRQ12
IRQ11	EISA IRQ11
IRQ10	EISA IRQ10
IRQ9	EISA IRQ9
IRQ8#	inverted mouse interrupt from 8242
IRQ7	parallel port interrupt from PC87312
IRQ6	EISA IRQ6
IRQ5	EISA IRQ5
IRQ4	serial port 1 interrupt from PC87312
IRQ3	EISA IRQ3
IRQ1	keyboard interrupt from 8242

[1] Should actually be pulled down, but will work OK

The behavior of the CPUREQ# signal on the ESC during reset will determine if internal or external PCI arbitration will be enabled. This signal is connected to the T2 request at the fingers. The T2 must drive this line high on Sable during reset, to enable internal arbitration. The TurboLaser must drive this line low during reset, to disable internal arbitration. In addition, the TurboLaser must drive the REQ1# signal (connected to req.~slot1 at the fingers) low during reset, so that the PCEB doesn't drive the bus during reset.

8.5 Remote IO Module

The Remote IO module provides a means for the PC ports controlled on the Standard IO to have access to the bulkhead. The control signals are brought up to the Remote IO via the 60 pin connector (see Appendix B). The module contains bulkhead connectors for the following:

- Keyboard

- Mouse
- Two serial ports
- Parallel port

In addition, there is a set of pins for the speaker to plug into. The speaker is physically mounted directly to the enclosure in close proximity to the Remote IO module.

Figure 51: Remote IO Connectors, Viewed from the Rear of the Enclosure

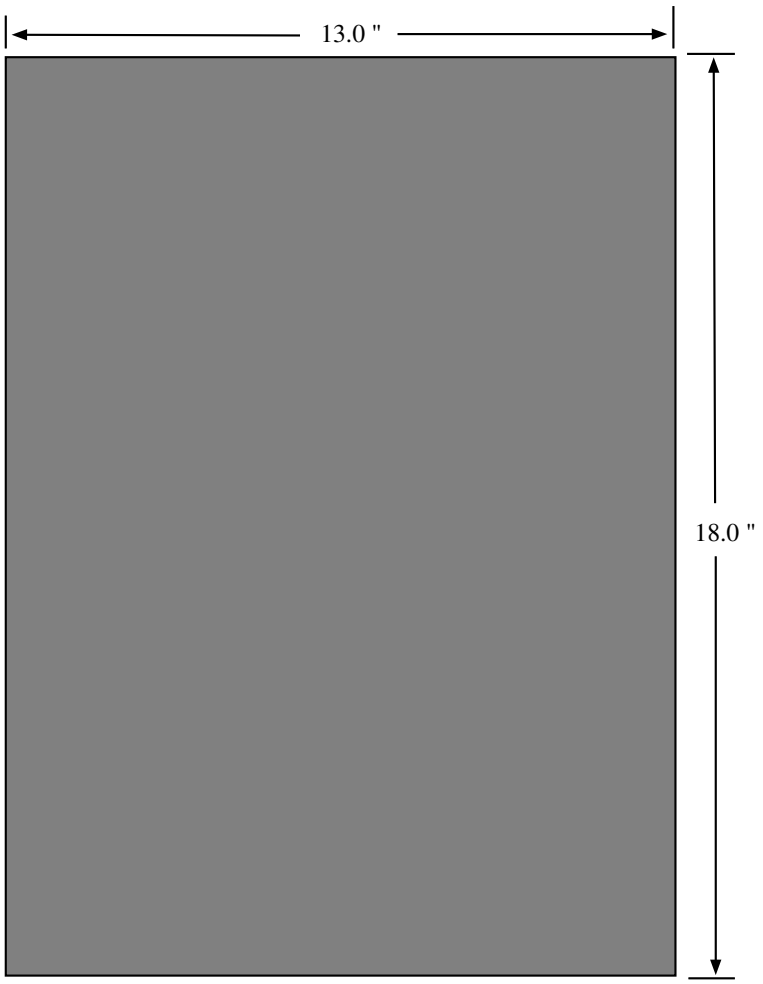
CHAPTER 9

PHYSICAL CHARACTERISTICS OF I/O MODULES

9.1 Motherboard

9.1.1 Motherboard Physical Dimensions

Figure 52: Motherboard Physical Dimensions



The Sable Motherboard/Backplane is a 18" x 13" module. It is .093" thick with 8 layers, 4 signals layers and 4 power/gnd layers.

mb_phy.doc

ps @ 80%

9.1.2 Motherboard Layer Construction

Table 52: Motherboard Layup

Layer	Type	Thickness	Weight
L1	Signal	.0014 min finished	.5 oz copper[1]
		.005 +.003/-.001	FR4
L2	Gnd	.0022 min	2 Oz copper
		.015 ±.003	FR4
L3	Signal[2]	.001 min	1 Oz copper
		.015 ±.003	FR4
L4	Pwr	.0022 min	2 Oz copper
		.005 +.003/-.001	FR4
L5	Gnd	.0022 min	2 Oz copper
		.015 ±.003	FR4
L6	Signal[2]	.001 min	1 Oz copper
		.015 ±.003	FR4
L7	Pwr	.0022 min	2 Oz copper
		.005 +.003/-.001	FR4
L8	Signal	.0014 min finished	.5 oz copper[1]
	Total Thickness	.090 ± .009	

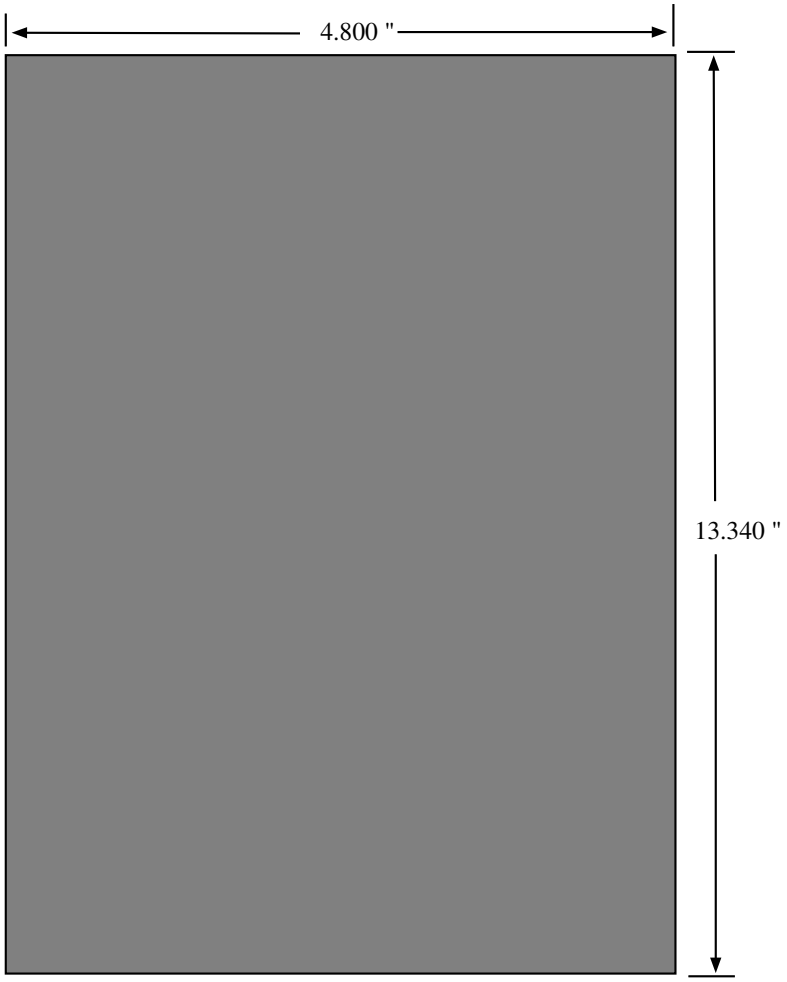
[1]Outer layers to be stared with .5 oz foil (.0007 ± .0003) and plated up to .0014 minimum.

[2]L3 and L6 use 6 mil etch for 68 ± 7 ohms. L3 and L6 use 14 mil etch for 48 +6/-5 ohms

9.2 Standard I/O

9.2.1 Standard I/O Physical Dimensions

Figure 53: Standard I/O Physical Dimensions



The Sable Standard IO is a 4.800" x 13.340" module. It is .062" thick with 8 layers, 6 signals layers and 2 power/gnd layers.

sio_phy.doc

ps @ 80%

9.2.2 Standard I/O Layer Construction

Table 53: Standard I/O Layup

Layer	Type	Thickness	Weight
L1	Signal[2]	.0014 min finished	.5 oz copper[1]
		.005 ±.0012	FR4
L2	Signal[2]	.001 min	1 Oz copper
		.008 ±.0015	FR4
L3	Power[2]	.001 min	1 Oz copper
		.009 ±.0020	FR4
L4	Signal[2]	.001 min	1 Oz copper
		.010 ±.0015	FR4
L5	Signal[2]	.001 min	1 Oz copper
		.009 ±.0020	FR4
L6	Ground[2]	.001 min	1 Oz copper
		.008 ±.0015	FR4
L7	Signal[2]	.001 min	1 Oz copper
		.005 ±.0012	FR4
L8	Signal[2]	.0014 min finished	.5 oz copper[1]
	Total Thickness	.062 ±.007	

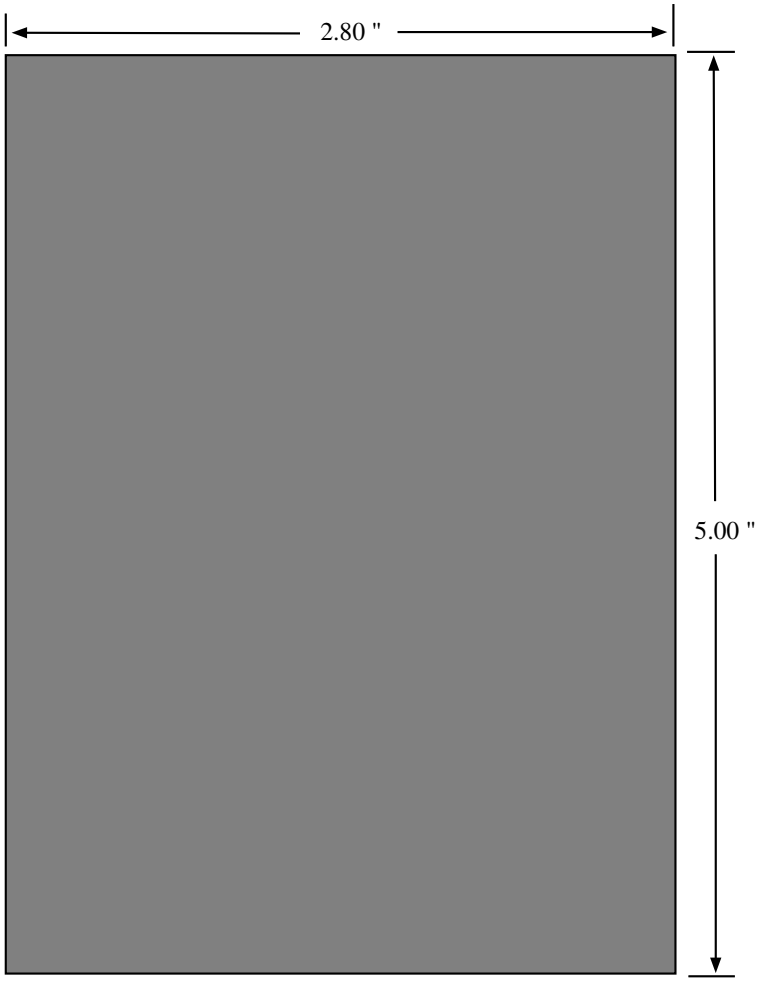
[1]Outer layers to be started with .5 oz copper plus plating to .0014 minimum.

[2]L1 and L8 are for slow routing only, no given specification.
L2 and L7 use 5 mil etch for 72 ± 7 ohms.
L4 and L5 use 5 mil etch for 68 ± 7 ohms.
L4 and L5 use 11 mil etch for 48 ± 5-6 ohms.

9.3 Remote I/O

9.3.1 Remote I/O Physical Dimensions

Figure 54: Remote I/O Physical Dimensions



The Sable Remote IO is a 2.80" x 5.00" module. It is .058" thick with 4 layers, 2 signals layers and 2 power/gnd layers.

rio_phy.doc
ps @ 80%

9.3.2 Remote I/O Layer Construction

Table 54: Remote I/O Layup

Layer	Type	Thickness	Weight
L1	Reference[1]	.0020 min	1 Oz copper
		.015 ±.002	FR4
L2	Signal	.0010 min	1 Oz copper
		.022 nominal	FR4
L3	Signal	.0010 min	1 Oz copper
		.015 ±.002	FR4
L4	Reference[1]	.0020 min	1 Oz copper
	Total Thickness	.058"	

[1]L1 and L4 are chassis ground planes.

CHAPTER 10

ISSUES

10.1 Pass 2 T2 changes

1. Register changes:
   - HAXR1 - added bits 4 and 3 to make total = [4:0].
   - HAXR2 - removed bits 31 and 30.
   - HAXR3 - added register with bits 31 and 30 from HAXR2.
   - HAXR4 - added register using bits 1 and 0 for DENSE space PCI address upper 2 bits (31 and 30).
   - WBASE1 - added bit 31 and 11 to starting and ending address fields.
   - WBASE2 - added bit 31 and 11 to starting and ending address fields.
   - IOCSR - added bit 10 : Enable Exclusive Exchange.
   - IOCSR - changed [35:33] Rev from 0 to 1.
2. Functional changes:
   - Expanded the DMA windows to cover the entire 4 Gbyte Memory Space.
   - Added programmability to the location of the Sparse and Dense Memory spaces.
   - Add programmability to the IVR address comparison register.
   - Added Exclusive Exchange Command for Gamma.
   - Fixed all pass one bugs.

10.1.1 T2 Pass 1 Bug List

1. Problem: Errant PCI stop pulse during DMA longword writes when the address offset is pointing to the last (8th) longword in a cache line. The errant pulse causes the Mercury chip set to malfunction.

   Solution: Pal fix on the Motherboard.
2. Problem: DMA writes while in Multiple Write Mode cause the system to hang. Caused by the same bug seen in problem 1.

   Solution: Multiple Write Mode can not be used in Pass 1 T2's.
3. Problem: PCI Lock protocol problem. Impelementation requires modification.

   Solution: Do not use PCI Lock transfer with Pass 1 T2's.

4. Problem: TLB invalidates do not invalidate on an entry per entry basis. Cache (Shared) protocol violation.

   Solution: Invalidate must be done by flushing the entire TLB for Pass 1 T2's.

5. Problem: C_ERR L timing violates CBUS spec.

   Solution: Only a problem in very severe worse case environments.

6. Problem: Dense Space writes cause system to hang and Dense Space reads can only access QW 0 of cache-line address.

   Solution: Do not do Dense Space transfers with Pass 1 T2's.

APPENDIX A

STANDARD I/O ADDRESSING

Table 55: STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
DMA1 CH0 Base and Current Address	3.A000.0000	0000	1 byte	R/W
DMA1 CH0 Base and Current Count	3.A000.0020	0001	1 byte	R/W
DMA1 CH1 Base and Current Address	3.A000.0040	0002	1 byte	R/W
DMA1 CH1 Base and Current Count	3.A000.0060	0003	1 byte	R/W
DMA1 CH2 Base and Current Address	3.A000.0080	0004	1 byte	R/W
DMA1 CH2 Base and Current Count	3.A000.00A0	0005	1 byte	R/W
DMA1 CH3 Base and Current Address	3.A000.00C0	0006	1 byte	R/W
DMA1 CH3 Base and Current Count	3.A000.00E0	0007	1 byte	R/W
DMA1 Status(r) Command(w) register	3.A000.0100	0008	1 byte	R/W
DMA1 Write Request register	3.A000.0120	0009	1 byte	WO
DMA1 Write Single Mask Bit	3.A000.0140	000A	1 byte	WO
DMA1 Write Mode register	3.A000.0160	000B	1 byte	WO
DMA1 Clear Byte Pointer	3.A000.0180	000C	1 byte	WO
DMA1 Master Clear	3.A000.01A0	000D	1 byte	WO
DMA1 Clear Mask register	3.A000.01C0	000E	1 byte	WO
DMA1 Read/Write All Mask Register Bits	3.A000.01E0	000F	1 byte	R/W
Reserved		0010-001F[1]		
Spare		0020-0021		
Configuration Address Index register	3.A000.0440	0022	1 byte	R/W
Configuration Data Index register	3.A000.0460	0023	1 byte	R/W
Spare		0024-25		
Edge Level Control Register 1	3.A000.04C0	0026	1 byte	R/W
Edge Level Control Register 2	3.A000.04E0	0027	1 byte	R/W
Aliased		0028-3F		
Timer 1 - Counter 0 System Clock	3.A000.0800	0040	1 byte	R/W

[1]Lost to don't cares in address decode in ESC chip

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
Timer 1 - Counter 1 Refresh Request	3.A000.0820	0041	1 byte	R/W
Timer 1 Counter 2 Speaker Tone	3.A000.0840	0042	1 byte	R/W
Timer 1 Command Mode register	3.A000.0860	0043	1 byte	WO
Spare		0044-0047		
Timer 2 - Counter 0 Fail-Safe Timer	3.A000.0880	0048	1 byte	R/W
Timer 2 - Counter 1 RESERVED	3.A000.08A0	0049	1 byte	R/W
Timer 2 - Counter 2 CPU Speed Control	3.A000.08C0	004A	1 byte	R/W
Timer 2 Command Mode register	3.A000.08E0	004B	1 byte	WO
Spare		004C-004F		
Reserved		0050-005B[1]		
Spare		005C-0060		
NMI Status and Control	3.A000.0C20	0061	1 byte	R/W
Spare		0062		
Reserved		0063[1]		
Spare		0064		
Reserved		0065[1]		
Spare		0066		
Reserved		0067[1]		
Spare		0068-006F		
Real Time Clock and NMI Enable Register[8]	3.A000.0E00	0070	1 byte	WO
Real Time Clock Control	3.A000.0E200	0071	1 byte	
Reserved		0072[1]		
Spare		0073		
Reserved		0074[1]		
Spare		0075		
Reserved		0076[1]		
Spare		0077-007F		
DMA Page register - reserved	3.A000.1000	0080	1 byte	R/W
DMA Channel 2 Page register	3.A000.1020	0081	1 byte	R/W
DMA Channel 3 Page register	3.A000.1040	0082	1 byte	R/W
DMA Channel 1 Page register	3.A000.1060	0083	1 byte	R/W
DMA Page register - reserved	3.A000.1080	0084	1 byte	R/W
DMA Page register - reserved	3.A000.10A0	0085	1 byte	R/W
DMA Page register - reserved	3.A000.10C0	0086	1 byte	R/W
DMA Channel 0 Page register	3.A000.10E0	0087	1 byte	R/W
DMA Page register - reserved	3.A000.1100	0088	1 byte	R/W

[1] Lost to don't cares in address decode in ESC chip

[8] Care should be taken when writing to this address. Integrity of bit 7 should be maintained when writing the Real Time Clock Address.

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
DMA Channel 6 Page register	3.A000.1120	0089	1 byte	R/W
DMA Channel 7 Page register	3.A000.1140	008A	1 byte	R/W
DMA Channel 5 Page register	3.A000.1160	008B	1 byte	R/W
DMA Page register - reserved	3.A000.1180	008C	1 byte	R/W
DMA Page register - reserved	3.A000.11A0	008D	1 byte	R/W
DMA Page register - reserved	3.A000.11C0	008E	1 byte	R/W
DMA DMA Refresh Page register	3.A000.11E0	008F	1 byte	R/W
Reserved		0090-0091[1]		
System Control Port	3.A000.1240	0092	1 byte	R/W
Reserved		0093-00A1[1]		
Reserved		00A2-00BF[1],[2]		
DMA2 CH0 Base and Current Address	3.A000.1800	00C0	1 byte	R/W
Reserved		00C1[1]		
DMA2 CH0 Base and Current Count	3.A000.1840	00C2	1 byte	R/W
Reserved		00C3[1]		
DMA2 CH1 Base and Current Address	3.A000.1880	00C4	1 byte	R/W
Reserved		00C5[1]		
DMA2 CH1 Base and Current Count	3.A000.18A0	00C6	1 byte	R/W
Reserved		00C7[1]		
DMA2 CH2 Base and Current Address	3.A000.18C0	00C8	1 byte	R/W
Reserved		00C9[1]		
DMA2 CH2 Base and Current Count	3.A000.1940	00CA	1 byte	R/W
Reserved		00CB[1]		
DMA2 CH3 Base and Current Address	3.A000.1980	00CC	1 byte	R/W
Reserved		00CD[1]		
DMA2 CH3 Base and Current Count	3.A000.19C0	00CE	1 byte	R/W
Reserved		00CF[1]		
DMA2 Status(r) Command(w) register	3.A000.1A00	00D0	1 byte	WO
Reserved		00D1[1]		
DMA2 Write Request register	3.A000.1A40	00D2	1 byte	WO
Reserved		00D3[1]		
DMA2 Write Single Mask Bit	3.A000.1A80	00D4	1 byte	WO
Reserved		00D5[1]		
DMA2 Write Mode register	3.A000.1AC0	00D6	1 byte	WO
Reserved		00D7[1]		

[1]Lost to don't cares in address decode in ESC chip

[2]Due to decoding, 2 are spare, 2 are reserved, etc. See ESC chip specification.

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
DMA2 Clear Byte Pointer	3.A000.1B00	00D8	1 byte	WO
Reserved		00D9[1]		
DMA2 Master Clear	3.A000.1B40	00DA	1 byte	WO
Reserved		00DB[1]		
DMA2 Clear Mask register	3.A000.1B80	00DC	1 byte	WO
Reserved		00DD[1]		
DMA2 Read/Write All Mask Register Bits	3.A000.1BC0	00DE	1 byte	R/W
Reserved		00DF[1]		
Spare		00E0-EF		
Reset IRQ13	3.A000.1E00	00F0	1 byte	WO
Spare		00F1-02F7		
Serial Port 1	3.A000.5F00-3.AA00.5FE0	2F8-2FF[4]		R/W
Spare		300-356		
Secondary Floppy Disk Controller	3.A000.6AE0-3.A000.6E00	0357-370[6]		WO
SPARE		371-397		
Combo Chip Configuration	3.A000.7300	0398	1 byte	R/W
Combo Chip Configuration	3.A000.7320	0399	1 byte	R/W
Spare		39A-3BB		
Parallel Port	3.A000.7780-3.A000.77C0	03BC-03BE[5]		R/W
Spare		03BF-03EF		
Primary Floppy Disk	3.A000.7E00-3.A000.7EE0	3F0-3F7[6]		R/W
Serial Port 0	3.A000.7F00-3.A000.7FE0	3F8-3FF[7]		R/W
Reserved		0400	1 byte	R/W
DMA1 CH0 Base /Current Count	3.A000.8020	0401	1 byte	R/W
Reserved	3.A000.8040	0402	1 byte	R/W
DMA1 CH1 Base/Current Count	3.A000.8060	0403	1 byte	R/W
Reserved	3.A000.8080	0404	1 byte	R/W
DMA1 CH2 Base/Current Count	3.A000.80A0	0405	1 byte	R/W
Reserved	3.A000.80C0	0406	1 byte	R/W
DMA1 CH3 Base/Current Count	3.A000.80E0	0407	1 byte	R/W
Reserved	3.A000.8100	0408	1 byte	R/W
Reserved	3.A000.8120	0409	1 byte	R/W

[1]Lost to don't cares in address decode in ESC chip

[4]Accesses to these addresses produce chip selects on the XBUS for Serial Port 1.

[5]Accesses to these addresses produce chip selects on the XBUS for the Parallel Port.

[6]Accesses to these addresses produce chip selects on the XBUS for the Floppy Disk Controllers.

[7]Accesses to these addresses produce chip selects on the XBUS for Serial Port 0.

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
DMA Chaining Mode Status/Interrupt Pending	3.A000.8140	040A	1 byte	R/W
DMA1 Extended Mode register	3.A000.8160	040B	1 byte	WO
Chaining Buffer Control Register	3.A000.8180	040C	1 byte	WO
Reserved	3.A000.81A0	040D	1 byte	R/W
Reserved	3.A000.81C0	040E	1 byte	R/W
Reserved	3.A000.81E0	040F	1 byte	R/W
DMA CH0 S-G Command Register	3.A000.8200	0410[3]	1 byte	WO
DMA CH1 S-G Command Register	3.A000.8220	0411[3]	1 byte	WO
DMA CH2 S-G Command Register	3.A000.8240	0412[3]	1 byte	WO
DMA CH3 S-G Command Register	3.A000.8260	0413[3]	1 byte	WO
DMA CH5 S-G Command Register	3.A000.82A0	0415[3]	1 byte	WO
DMA CH6 S-G Command Register	3.A000.82C0	0416[3]	1 byte	WO
DMA CH7 S-G Command Register	3.A000.82E0	0417[3]	1 byte	WO
DMA CH0 S-G Status register	3.A000.8300	0418[3]	1 byte	WO
DMA CH1 S-G Status register	3.A000.8320	0419[3]	1 byte	WO
DMA CH2 S-G Status register	3.A000.8340	041A[3]	1 byte	WO
DMA CH3 S-G Status register	3.A000.8360	041B[3]	1 byte	WO
DMA CH5 S-G Status register	3.A000.83A0	041D[3]	1 byte	WO
DMA CH6 S-G Status register	3.A000.83C0	041E[3]	1 byte	WO
DMA CH7 S-G Status register	3.A000.83E0	041F[3]	1 byte	WO
DMA CH0 S-G Descriptor Pointer register	3.A000.8400	0420[3]	1 byte	RO
DMA CH0 S-G Descriptor Pointer register	3.A000.8420	0421[3]	1 byte	RO
DMA CH0 S-G Descriptor Pointer register	3.A000.8440	0422[3]	1 byte	RO
DMA CH0 S-G Descriptor Pointer register	3.A000.8460	0423[3]	1 byte	RO
DMA CH1 S-G Descriptor Pointer register	3.A000.8480	0424[3]	1 byte	RO
DMA CH1 S-G Descriptor Pointer register	3.A000.84A0	0425[3]	1 byte	RO
DMA CH1 S-G Descriptor Pointer register	3.A000.84C0	0426[3]	1 byte	RO
DMA CH1 S-G Descriptor Pointer register	3.A000.84E0	0427[3]	1 byte	RO
DMA CH2 S-G Descriptor Pointer register	3.A000.8500	0428[3]	1 byte	RO
DMA CH2 S-G Descriptor Pointer register	3.A000.8520	0429[3]	1 byte	RO

[3]The Scatter-Gather DMA registers are relocatable. The default I/O adress for the Scatter-Gather Registers on Power-up are 0410-043F. Thse registers can be mapped in the I/O range xx10-xx3F.

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
DMA CH2 S-G Descriptor Pointer register	3.A000.8540	042A[3]	1 byte	RO
DMA CH2 S-G Descriptor Pointer register	3.A000.8560	042B[3]	1 byte	RO
DMA CH3 S-G Descriptor Pointer register	3.A000.8580	042C[3]	1 byte	RO
DMA CH3 S-G Descriptor Pointer register	3.A000.85A0	042D[3]	1 byte	RO
DMA CH3 S-G Descriptor Pointer register	3.A000.85C0	042E[3]	1 byte	RO
DMA CH3 S-G Descriptor Pointer register	3.A000.85E0	042F[3]	1 byte	RO
Spare		0430-0433		
DMA CH5 S-G Descriptor Pointer register	3.A000.8680	0434[3]	1 byte	RO
DMA CH5 S-G Descriptor Pointer register	3.A000.86A0	0435[3]	1 byte	RO
DMA CH5 S-G Descriptor Pointer register	3.A000.86C0	0436[3]	1 byte	RO
DMA CH5 S-G Descriptor Pointer register	3.A000.86E0	0437[3]	1 byte	RO
DMA CH6 S-G Descriptor Pointer register	3.A000.8700	0438[3]	1 byte	RO
DMA CH6 S-G Descriptor Pointer register	3.A000.8720	0439[3]	1 byte	RO
DMA CH6 S-G Descriptor Pointer register	3.A000.8740	043A[3]	1 byte	RO
DMA CH6 S-G Descriptor Pointer register	3.A000.8760	043B[3]	1 byte	RO
DMA CH7 S-G Descriptor Pointer register	3.A000.8780	043C[3]	1 byte	RO
DMA CH7 S-G Descriptor Pointer register	3.A000.87A0	043D[3]	1 byte	RO
DMA CH7 S-G Descriptor Pointer register	3.A000.87C0	043E[3]	1 byte	RO
DMA CH7 S-G Descriptor Pointer register	3.A000.87E0	043F[3]	1 byte	RO
Spare		0440-0460		
Extended NMI and Reset Control	3.A000.8C20	0461	1 byte	R/W
NMI I/O Interrupt port	3.A000.8C40	0462	1 byte	R/W
Spare	3.A000.8C60	0463		
Last EISA Bus master granted (L)	3.A000.8C80	0464	1 byte	RO
Spare		0465-047F		

[3]The Scatter-Gather DMA registers are relocatable. The default I/O adress for the Scatter-Gather Registers on Power-up are 0410-043F. Thse registers can be mapped in the I/O range xx10-xx3F.

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
Reserved	3.A000.9000	0480	1 byte	R/W
DMA CH2 High Page register	3.A000.9020	0481	1 byte	R/W
DMA CH3 High Page register	3.A000.9040	0482	1 byte	R/W
DMA CH1 High Page register	3.A000.9060	0483	1 byte	R/W
Reserved	3.A000.9080	0484	1 byte	R/W
Reserved	3.A000.90A0	0485	1 byte	R/W
Reserved	3.A000.90C0	0486	1 byte	R/W
DMA CH0 High Page register	3.A000.90E0	0487	1 byte	R/W
Reserved	3.A000.9100	0488	1 byte	R/W
DMA CH6 High Page register	3.A000.9120	0489	1 byte	R/W
DMA CH7 High Page register	3.A000.9140	048A	1 byte	R/W
DMA CH5 High Page register	3.A000.9160	048B	1 byte	R/W
Reserved	3.A000.9180	048C	1 byte	R/W
Reserved	3.A000.91A0	048D	1 byte	R/W
Reserved	3.A000.91C0	048E	1 byte	R/W
DMA Refresh High Page register	3.A000.91D0	048F	1 byte	R/W
Spare		0490-4C1		
Reserved	3.A000.9840	04C2	1 byte	R/W
Spare		04C3-4C5		
DMA CH5 High Base & Current Count register	3.A000.98C0	04C6	1 byte	R/W
Spare		04C5-04C9		
DMA CH5 High Base & Current Count register	3.A000.9940	04CA	1 byte	R/W
Spare		04CB-04CD		
DMA CH5 High Base & Current Count register	3.A000.99C0	04CE	1 byte	R/W
Spare		04CF		
INT-1 edge/Level Control register	3.A000.9A00	04D0	1 byte	R/W
INT-2 edge/Level Control register	3.A000.9A20	04D1	1 byte	R/W
Reserved	3.A000.9A40	04D2	1 byte	R/W
Reserved	3.A000.9A60	04D3	1 byte	R/W
DMA2 Chaining Mode	3.A000.9A80	04D4	1 byte	R/W
Reserved	3.A000.9AA0	04D5	1 byte	R/W
DMA2 Extended Mode Register	3.A000.9AC0	04D6	1 byte	WO
Reserved	3.A000.9AE0	04D7	1 byte	R/W
Reserved	3.A000.9B00	04D8	1 byte	R/W
Reserved	3.A000.9B20	04D9	1 byte	R/W
Reserved	3.A000.9B40	04DA	1 byte	R/W
Reserved	3.A000.9B60	04DB	1 byte	R/W
Reserved	3.A000.9B80	04DC	1 byte	R/W

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
Reserved	3.A000.9BA0	04DD	1 byte	R/W
Reserved	3.A000.9BC0	04DE	1 byte	R/W
Reserved	3.A000.9BE0	04DF	1 byte	R/W
DMA CH0 Stop Register Bits [7:2]	3.A000.9C00	04E0	1 byte	R/W
DMA CH0 Stop Register Bits [15:8]	3.A000.9C20	04E1	1 byte	R/W
DMA CH0 Stop Register Bits [23:16]	3.A000.9C40	04E2	1 byte	R/W
Reserved	3.A000.9C60	04E3	1 byte	R/W
DMA CH1 Stop Register Bits [7:2]	3.A000.9C80	04E4	1 byte	R/W
DMA CH1 Stop Register Bits [15:8]	3.A000.9CA0	04E5	1 byte	R/W
DMA CH1 Stop Register Bits [23:16]	3.A000.9CC0	04E6	1 byte	R/W
Reserved	3.A000.9CE0	04E7	1 byte	R/W
DMA CH2 Stop Register Bits [7:2]	3.A000.9D00	04E8	1 byte	R/W
DMA CH2 Stop Register Bits [15:8]	3.A000.9D20	04E9	1 byte	R/W
DMA CH2 Stop Register Bits [23:16]	3.A000.9D40	04EA	1 byte	R/W
Reserved	3.A000.9D60	04EB	1 byte	R/W
DMA CH3 Stop Register Bits [7:2]	3.A000.9D80	04EC	1 byte	R/W
DMA CH3 Stop Register Bits [15:8]	3.A000.9DA0	04ED	1 byte	R/W
DMA CH3 Stop Register Bits [23:16]	3.A000.9DC0	04EE	1 byte	R/W
Reserved	3.A000.9DE0	04EF	1 byte	R/W
Reserved	3.A000.9E00	04F0	1 byte	R/W
Reserved	3.A000.9E20	04F1	1 byte	R/W
Reserved	3.A000.9E40	04F2	1 byte	R/W
Reserved	3.A000.9E60	04F3	1 byte	R/W
DMA CH5 Stop Register Bits [7:2]	3.A000.9E80	04F4	1 byte	R/W
DMA CH5 Stop Register Bits [15:8]	3.A000.9EA0	04F5	1 byte	R/W
DMA CH5 Stop Register Bits [23:16]	3.A000.9EC0	04F6	1 byte	R/W
Reserved	3.A000.9EE0	04F7	1 byte	R/W
DMA CH6 Stop Register Bits [7:2]	3.A000.9F00	04F8	1 byte	R/W
DMA CH6 Stop Register Bits [15:8]	3.A000.9F20	04F9	1 byte	R/W
DMA CH6 Stop Register Bits [23:16]	3.A000.9F40	04FA	1 byte	R/W
Reserved	3.A000.9F60	04FB	1 byte	R/W
DMA CH7 Stop Register Bits [7:2]	3.A000.9F80	04FC	1 byte	R/W
DMA CH7 Stop Register Bits [15:8]	3.A000.9FA0	04FD	1 byte	R/W
DMA CH7 Stop Register Bits [23:16]	3.A000.9FC0	04FE	1 byte	R/W
Reserved	3.A000.9FE0	04FF	1 byte	R/W
Reserved (Old Enet ROM)	3.A000.A000-3.A000.A3E0	0500-051F	32 bytes	R/W
Spare		0520-052F		
I^2C	3.A000.A600	0530	1 byte	R/W
I^2C	3.A000.A620	0531	1 byte	R/W
IACK	3.A000.A640	0532	1 byte	R/W

Table 55 (Cont.): STANDARD I/O ADDRESSING

Register Name or Address Space	EV Address	PCI Address	Length	Attributes
Spare		0533		
Master 8259 Interrupt Control Register	3.A000.A680	0534	1 byte	R/W
Master 8259 Interrupt Mask Register	3.A000.A6A0	0535	1 byte	R/W
Slave 0 8259 Interrupt Control register	3.A000.A6C0	0536	1 byte	R/W
Slave 0 8259 Interrupt Mask register	3.A000.A6E0	0537	1 byte	R/W
Spare		0538-39		
Slave 1 8259 Interrupt Control register	3.A000.A740	053A	1 byte	R/W
Slave 1 8259 Interrupt Mask register	3.A000.A760	053B	1 byte	R/W
Slave 2 8259 Interrupt Control register	3.A000.A780	053C	1 byte	R/W
Slave 2 8259 Interrupt Mask register	3.A000.A7A0	053D	1 byte	R/W
Slave 3 8259 Interrupt Control register	3.A000.A7C0	053E	1 byte	R/W
Slave 3 8259 Interrupt Mask register	3.A000.A7E0	053F	1 byte	R/W
Spare		0540-07FF		
NVRAM on Standard I/O	3.A001.0000-3.A001.1FE0	0800-08FF		R/W
Spare		0900-0BFF		
Configuration RAM Page register	3.A001.8000	0C00	1 byte	R/W
Spare		0C01-0C7F		
System Board ID Byte Lane 1 Bits [7:0]	3.A001.9000	0C80	1 byte	RO
System Board ID Byte Lane 2 Bits [15:8]	3.A001.9020	0C81	1 byte	RO
System Board ID Byte Lane 3 Bits [23:16]	3.A001.9040	OC82	1 byte	RO
System Board ID Byte Lane 4 Bits [31:24]	3.A001.9060	0C83	1 byte	RO
Spare	3.A001.9080-3.A001.FFFF	0C84-0FFF		

APPENDIX B

STANDARD I/O CONNECTORS AND HEADERS

The Standard IO has six connectors and 360 gold fingers.

B.1 120 Pin Section of the Gold Fingers

Pin	Signal Name
J11.1	EISASD12
J11.2	EISASD30
J11.3	EISASD13
J11.4	EISASD31
J11.5	EISASD14
J11.6	EISASD15
J11.7	GNDL
J11.8	NMI
J11.9	INT
J11.10	PCIIRQS2BL
J11.11	CSYSEVENTL
J11.12	GNDL
J11.13	EISAMACKL0
J11.14	EISAMREQL0
J11.15	EISAMACKL1
J11.16	EISAMREQL1
J11.17	EISAMACKL2
J11.18	EISAMREQL2
J11.19	EISAMACKL3
J11.20	EISAMREQL3
J11.21	EISAMACKL4
J11.22	EISAMREQL4
J11.23	EISAMACKL5
J11.24	EISAMREQL5
J11.25	EISAMACKL6
J11.26	EISAMREQL6
J11.27	EISAMACKL7
J11.28	EISAMREQL7
J11.29	P5000
J11.30	P5000
J11.31	P5000

Pin	Signal Name
J11.32	P5000
J11.33	P5000
J11.34	P5000
J11.35	P5000
J11.36	P5000
J11.37	P5000
J11.38	P5000
J11.39	GNDL
J11.40	GNDL
J11.41	GNDL
J11.42	GNDL
J11.43	GNDL
J11.44	GNDL
J11.45	GNDL
J11.46	GNDL
J11.47	GNDL
J11.48	GNDL
J11.49	GNDL
J11.52	GNDL
J11.53	MBP33
J11.54	PCIIRQS2DL
J11.55	IOPRSNTL
J11.56	PCIIRQS1DL
J11.57	PCIIRQS0DL
J11.58	FPOK
J11.59	TACH
J11.60	PGOODDCOK1
J11.61	PGOODPOK1
J11.62	DCONL
J11.63	EISADREQL7
J11.64	EISAMASTER16L
J11.65	GNDL
J11.66	EISABCLK
J11.67	GNDL
J11.68	BCLKOUT
J11.69	GNDL
J11.70	PCIPCLKMISC
J11.71	GNDL
J11.72	PCIREQT2L
J11.73	PCIGNTT2L
J11.74	GNDL
J11.75	EISAAEN0L
J11.76	GNDL

Pin	Signal Name
J11.77	EISAAEN1L
J11.78	GNDL
J11.79	EISAAEN2L
J11.80	GNDL
J11.81	EISAAEN3L
J11.82	GNDL
J11.83	EISAAEN4L
J11.84	GNDL
J11.85	EISAAEN5L
J11.86	GNDL
J11.87	EISAAEN6L
J11.88	GNDL
J11.89	EISAAEN7L
J11.90	GNDL
J11.91	P5000
J11.92	P5000
J11.93	P5000
J11.94	P5000
J11.95	P5000
J11.96	P5000
J11.97	P5000
J11.98	P5000
J11.99	P5000
J11.100	P5000
J11.101	GNDL
J11.102	GNDL
J11.103	GNDL
J11.104	GNDL
J11.105	GNDL
J11.106	GNDL
J11.107	GNDL
J11.108	GNDL
J11.109	GNDL
J11.110	GNDL
J11.111	GNDL
J11.114	GNDL
J11.115	GNDL
J11.116	ESCINT
J11.117	PCIIRQS2CL
J11.118	PCIIRQS1CL
J11.119	PCIIRQS0CL
J11.120	PCIIRQS1BL
J11.121	PCIIRQS0BL

Pin	Signal Name
J11.122	PGOODDCOK2
J11.123	PGOODPOK2
J11.124	PSERRL

B.2 240 Pin Section of the Gold Fingers

Pin	Signal Name
J12.1	CBUSXIOSYSEVENTL
J12.2	GNDL
J12.3	IBUSPCLK
J12.4	GNDL
J12.5	EISASD11
J12.6	EISASD10
J12.7	EISASD28
J12.8	EISASD9
J12.9	GNDL
J12.10	EISASD26
J12.11	EISASD8
J12.12	EISASD25
J12.13	EISAMWRL
J12.14	EISAMRDL
J12.15	EISASD22
J12.16	GNDL
J12.17	EISALA17
J12.18	EISASD20
J12.19	EISALA18
J12.20	EISASD19
J12.21	EISALA19
J12.22	EISASD17
J12.23	GNDL
J12.24	EISALA20
J12.25	EISALA21
J12.26	EISALA22
J12.27	EISALA3
J12.28	EISALA23
J12.29	EISALA4
J12.30	GNDL
J12.31	EISASBHEL
J12.32	EISALA7
J12.33	EISALA9
J12.34	EISASA0
J12.35	EISASA1

Pin	Signal Name
J12.36	EISALA11
J12.37	GNDL
J12.38	EISASA2
J12.39	EISALA12
J12.40	EISASA3
J12.41	EISALA13
J12.42	EISASA4
J12.43	EISALA15
J12.44	EISASA5
J12.45	GNDL
J12.46	EISASA6
J12.47	EISASA7
J12.48	EISALAL25
J12.49	PCIAD1
J12.50	GNDL
J12.51	PCIAD3
J12.52	PCIAD5
J12.53	PCIAD7
J12.54	PCIAD8
J12.55	GNDL
J12.56	EISASA8
J12.57	PCIAD10
J12.58	PCIAD12
J12.59	PCIAD14
J12.60	GNDL
J12.61	PCICBEL1
J12.62	EISALAL27
J12.63	EISASA9
J12.64	PCISERRL
J12.65	PCIPERRL
J12.66	PCILOCKL
J12.67	PCIDEVSELL
J12.68	PCIIRDYL
J12.69	GNDL
J12.70	PCICBEL2
J12.71	PCIAD17
J12.72	EISALAL28
J12.73	PCIAD19
J12.74	GNDL
J12.75	PCIAD21
J12.76	PCIAD23
J12.77	PCICBEL3
J12.78	PCIAD25

Pin	Signal Name
J12.79	GNDL
J12.80	EISASA11
J12.81	PCIAD27
J12.82	PCIAD29
J12.83	PCIAD31
J12.84	GNDL
J12.85	EISASA12
J12.86	EISALAL31
J12.87	EISASA13
J12.88	GNDL
J12.89	EISABEL1
J12.90	EISASA14
J12.91	EISASA15
J12.92	GNDL
J12.93	EISASA16
J12.94	EISASA17
J12.95	EISASA18
J12.96	EISASA19
J12.97	GNDL
J12.98	EISARDL
J12.99	CHRDY
J12.100	EISAMSBURSTL
J12.101	EISASD0
J12.102	EISASLBURSTL
J12.103	EISASD1
J12.104	EISAEX16L
J12.105	EISASD2
J12.106	EISADREQL2
J12.107	EISASD3
J12.108	M5000L
J12.109	EISASD4
J12.110	EISAEX32L
J12.111	EISAIRQL9
J12.112	EISASD5
J12.113	EISAEXRDY
J12.114	GNDL
J12.115	EISASD6
J12.116	EISASTARTL
J12.117	J12RESDRV
J12.118	EISASD7
J12.119	EISACMDL
J12.120	EISAIOCHKL
J12.121	PCIPCLKENET

Pin	Signal Name
J12.122	GNDL
J12.123	PCIPCLKSCSI
J12.124	GNDL
J12.125	EISASD29
J12.126	EISADACKL7L
J12.127	EISADREQL6
J12.128	EISASD27
J12.129	GNDL
J12.130	EISADACKL6L
J12.131	EISADREQL5
J12.132	EISASD24
J12.133	EISADACKL5L
J12.134	EISASD23
J12.135	EISADREQL0
J12.136	GNDL
J12.137	EISASD21
J12.138	EISADACKL0L
J12.139	EISAIRQL14
J12.140	EISASD18
J12.141	EISAIRQL15
J12.142	EISASD16
J12.143	GNDL
J12.144	EISAIRQL12
J12.145	EISAIRQL11
J12.146	EISALA2
J12.147	EISAIRQL10
J12.148	EISAIO16L
J12.149	EISALA5
J12.150	GNDL
J12.151	EISAM16L
J12.152	EISALA6
J12.153	EISALA8
J12.154	EISALA10
J12.155	CINTTIM
J12.156	ASYNCRESETL
J12.157	GNDL
J12.158	EISAOSC
J12.159	GNDL
J12.160	EISABALE
J12.161	EISALA14
J12.162	EISATC
J12.163	EISALA16
J12.164	EISADACKL2L

Pin	Signal Name
J12.165	GNDL
J12.166	EISAIRQL3
J12.167	EISALAL24
J12.168	EISAIRQL4
J12.169	PCIAD0
J12.170	GNDL
J12.171	PCIAD2
J12.172	PCIAD4
J12.173	PCIAD6
J12.174	PCICBEL0
J12.175	GNDL
J12.176	EISALAL26
J12.177	PCIAD9
J12.178	PCIAD11
J12.179	PCIAD13
J12.180	GNDL
J12.181	EISAIRQL5
J12.182	PCIAD15
J12.183	EISAIRQL6
J12.184	PCIPAR
J12.185	EISALAL29
J12.186	PCISTOPL
J12.187	PCITRDYL
J12.188	PCIFRAMEL
J12.189	GNDL
J12.190	EISAIRQL7
J12.191	PCIAD16
J12.192	PCIAD18
J12.193	PCIAD20
J12.194	GNDL
J12.195	PCIAD22
J12.196	EISASA10
J12.197	EISALAL30
J12.198	PCIAD24
J12.199	GNDL
J12.200	PCIAD26
J12.201	PCIAD28
J12.202	PCIAD30
J12.203	PCIRESETL
J12.204	GNDL
J12.205	EISAREFRESHL
J12.206	EISABEL0
J12.207	EISADREQL1

Pin	Signal Name
J12.208	GNDL
J12.209	EISABEL2
J12.210	EISADACKL1L
J12.211	EISADREQL3
J12.212	EISABEL3
J12.213	EISADACKL3L
J12.214	EISAIORDL
J12.215	EISAIOWRL
J12.216	EISASMRDL
J12.217	EISALOCKL
J12.218	EISASMWRL
J12.219	EISAIOL
J12.220	GNDL
J12.221	GNDL
J12.222	P12000
J12.223	EISANOWSL
J12.224	P12000
J12.225	M12000L
J12.226	PCIGNTS1L
J12.227	PCIREQS1L
J12.228	PCIGNTS0L
J12.229	PCIREQS0L
J12.230	PCIGNTS2L
J12.231	PCIREQS2L
J12.232	PCIGNTENETL
J12.233	PCIREQENETL
J12.234	GNDL
J12.235	PCIIRQENETL
J12.236	PCIIRQS2AL
J12.237	SDA
J12.238	PCIIRQS1AL
J12.239	SCL
J12.240	PCIIRQS0AL

B.3 Remote System Monitor Header

Please address any questions on this option to Erik Piip, at SPESHR::PIIP.

Pin	Signal Name
J15.1	RFMP33
J15.2	GND
J15.3	FANFAIL

Pin	Signal Name
J15.4	PS.DCOK1
J15.5	PS.DCOK2
J15.6	PS.POK1
J15.7	PS.POK2
J15.8	RMODE
J15.9	GND
J15.10	HALT
J15.11	OCPRESET
J15.12	DCON
J15.13	GND
J15.14	SCL
J15.15	GND
J15.16	SDA

B.4 50 Pin SCSI Header

This is a standard SCSI pinout, as detailed below. Please refer to the SCSI specification for more details on signal functions.

Pin	Signal Name
j10.1	gnd
j10.2	~Scsid0
j10.3	gnd
j10.4	~Scsid1
j10.5	gnd
j10.6	~Scsid2
j10.7	gnd
j10.8	~Scsid3
j10.9	gnd
j10.10	~Scsid4
j10.11	gnd
j10.12	~Scsid5
j10.13	gnd
j10.14	~Scsid6
j10.15	gnd
j10.16	~Scsid7
j10.17	gnd
j10.18	~Scsidp
j10.19	gnd
j10.20	gnd
j10.21	gnd
j10.22	gnd

Pin	Signal Name
j10.23	gnd
j10.24	gnd
j10.25	ncln
j10.26	termpwr
j10.27	gnd
j10.28	gnd
j10.29	gnd
j10.30	gnd
j10.31	gnd
j10.32	~Scsiatn
j10.33	gnd
j10.34	gnd
j10.35	gnd
j10.36	~Scsibsy
j10.37	gnd
j10.38	~Scsiack
j10.39	gnd
j10.40	~Scsisrst
j10.41	gnd
j10.42	~Scsimsg
j10.43	gnd
j10.44	~Scsisel
j10.45	gnd
j10.46	~Scsicd
j10.47	gnd
j10.48	~Scsireq
j10.49	gnd
j10.50	~Scsiio

B.5 60 Pin Header

Pin	Signal Name	Function
J8.1	gnd	ground
J8.2	XBus.msclk	Mouse
J8.3	XBus.msdata	Mouse
J8.4	gnd	ground
J8.5	XBus.kbclk	Keyboard
J8.6	XBus.kbdata	Keyboard
J8.7	gnd	ground
J8.8	p5000	+5 Volts
J8.9	p5000	+5 Volts
J8.10	p5000	+5 Volts

Pin	Signal Name	Function
J8.11	bs2.dsr	Serial Port 2
J8.12	bs2.dcd	Serial Port 2
J8.13	bs2.rts	Serial Port 2
J8.14	gnd	ground
J8.15	bs2.~rx	Serial Port 2
J8.16	bs2.~tx	Serial Port 2
J8.17	gnd	ground
J8.18	bs2.cts	Serial Port 2
J8.19	bs2.dtr	Serial Port 2
J8.20	bs2.ri	Serial Port 2
J8.21	gnd	ground
J8.22	bs1.dcd	Serial Port 1
J8.23	bs1.dsr	Serial Port 1
J8.24	gnd	ground
J8.25	bs1.~rx	Serial Port 1
J8.26	bs1.~tx	Serial Port 1
J8.27	gnd	ground
J8.28	bs1.rts	Serial Port 1
J8.29	bs1.dtr	Serial Port 1
J8.30	bs1.cts	Serial Port 1
J8.31	bs1.ri	Serial Port 1
J8.32	gnd	ground
J8.33	lp.~stb	Parallel Port
J8.34	gnd	ground
J8.35	lp.d[0]	Parallel Port
J8.36	lp.d[1]	Parallel Port
J8.37	lp.d[2]	Parallel Port
J8.38	lp.d[3]	Parallel Port
J8.39	gnd	ground
J8.40	lp.~afd	Parallel Port
J8.41	gnd	ground
J8.42	lp.~err	Parallel Port
J8.43	lp.~init	Parallel Port
J8.44	gnd	ground
J8.45	lp.~slin	Parallel Port
J8.46	gnd	ground
J8.47	lp.d[4]	Parallel Port
J8.48	lp.d[5]	Parallel Port
J8.49	lp.d[6]	Parallel Port
J8.50	lp.d[7]	Parallel Port
J8.51	gnd	ground
J8.52	lp.~ack	Parallel Port
J8.53	gnd	ground

Pin	Signal Name	Function
J8.54	lp.busy	Parallel Port
J8.55	gnd	ground
J8.56	lp.pe	Parallel Port
J8.57	lp.slct	Parallel Port
J8.58	gnd	ground
J8.59	spkr[1]	Speaker
J8.60	spkr[0]	Speaker

B.6 34 Pin Floppy Header

This is a standard 34 pin floppy pinout, as detailed below. Please refer to the FDC specification for more details on signal functions.

Pin	Signal Name
J9.1	gnd
J9.2	densel
J9.3	gnd
J9.4	ncIn
J9.5	gnd
J9.6	drate0
J9.7	gnd
J9.8	~index
J9.9	gnd
J9.10	~mtr0
J9.11	gnd
J9.12	~dr1
J9.13	gnd
J9.14	~dr0
J9.15	gnd
J9.16	~mtr1
J9.17	ncIn
J9.18	~dir
J9.19	gnd
J9.20	~step
J9.21	gnd
J9.22	~wdata
J9.23	gnd
J9.24	~wgate
J9.25	gnd
J9.26	~trko
J9.27	ncIn
J9.28	~wp

Pin	Signal Name
J9.29	gnd
J9.30	~rdata
J9.31	gnd
J9.32	~hdsel
J9.33	gnd
J9.34	~dskchg

B.7 10 Pin OCP Header

Pin	Signal Name	Function
J7.1	p12000	+12 Volts
J7.2	p5000	+5 Volts
J7.3	gnd	Ground
J7.4	p5000	+5 Volts
J7.5	gnd	Ground
J7.6	~halt	User directed Halt
J7.7	~ocpReset	User directed Reset
J7.8	sda	I²C data
J7.9	scl	I²C clock
J7.10	~dcON	Signal from OCP to Power supply logic??

B.8 8 Pin MJ Connector and 15 Pin Thickwire Connector

These are standard twisted pair and thickwire connectors. respectively. Please refer to the Ethernet specification for more details on signal functions.

Pin	Signal Name
J1.1	txo
J1.2	~txo
J1.3	rxo
J1.4	ncln
J1.5	ncln
J1.6	~rxo
J1.7	ncln
J1.8	ncln

Pin	Signal Name
J2.1	chassis
J2.2	Fthk.clsn
J2.3	Fthk.xmit

Pin	Signal Name
J2.4	chassis
J2.5	Fthk.rcv
J2.6	pthkret
J2.7	ncIn
J2.8	chassis
J2.9	Fthk.~clsn
J2.10	Fthk.~xmit
J2.11	chassis
J2.12	Fthk.~rcv
J2.13	pthick
J2.14	chassis
J2.15	ncIn

APPENDIX C

INTERRUPT CONTROL DETAILS

The following picture illustrates the functions implemented by and interaction of PALS SIOD and SIOE.

Figure 55: PALS SIOD and SIOE, Functional Diagram

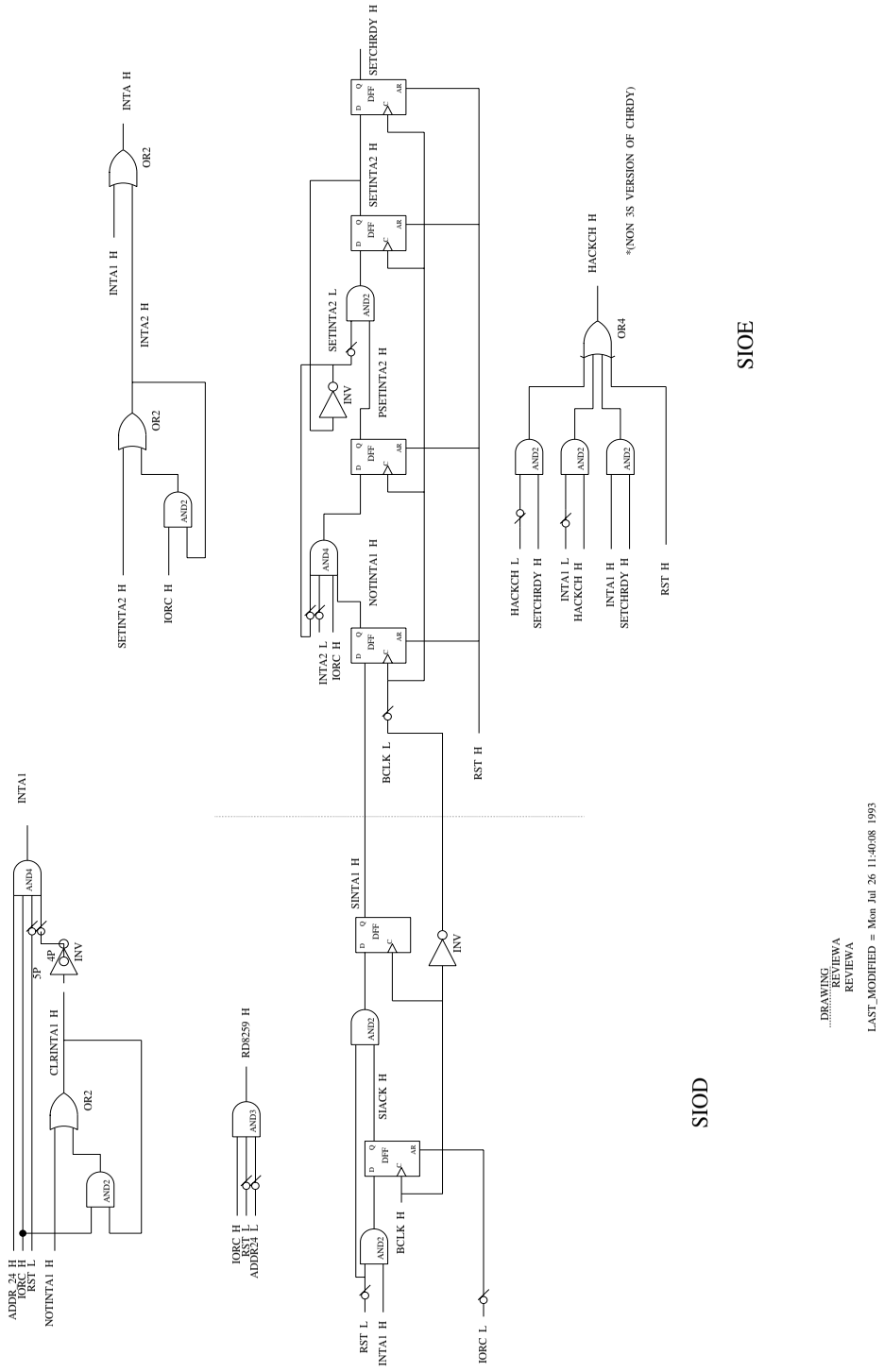
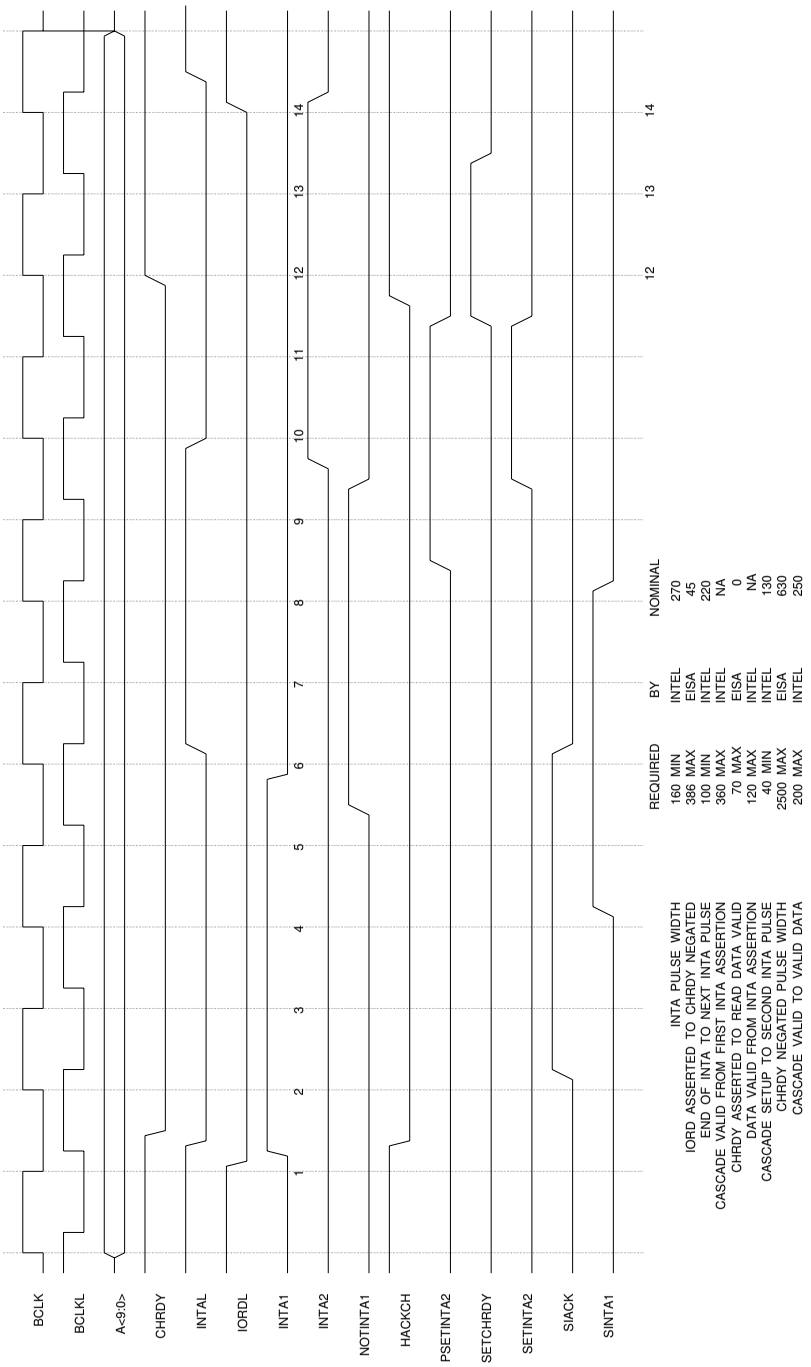


Figure 56: PALS SIOD and SIOE, Timing Diagram



APPENDIX D

PRELIMINARY POWER CONSUMPTION

```
        <<< MVBLAB::SYS$SYSDEVICE:[NOTES$LIBRARY]SABLE_QUAL.NOTE;1 >>>
                              -< SABLE_QUAL >-
================================================================================
Note 34.5                          CPU/IO DVT Plan                          5 of 5
JANVAX::NERL "John Nerl"                                42 lines  21-DEC-1993 10:21
                     -< Preliminary Power Consumption >-
--------------------------------------------------------------------------------

+---------------------------+ TM
|   |   |   |   |   |   |   |
| d | i | g | i | t | a | l |   I N T E R O F F I C E    M E M O R A N D U M
|   |   |   |   |   |   |   |
+---------------------------+


TO: Sable Team                              DATE:  December 21, 1993
                                                FROM:  John Nerl
                                                DEPT:  AVSPD - Design Assurance
                                                 EXT:  223-5062
                                              LOC/MS:  MLO5-5/U52
         ENET:  JANVAX::NERL

SUBJECT:  Preliminary Power Consumption Measurements for Sable modules

 I have completed preliminary power consumption measurements on a
small sample of Sable modules.  The numbers listed are maximum values
observed under worst-case operating conditions, with 5% added.  Measurements
were taken on a system running at 24nS Cobra-bus and 5.26nS processor cycle
times.

DEVICE  +5V +3.3V    +12V     -12V
-------------------------------------------------

CPU (A03) 9.36A 8.29A    -----    -----

SMM- 64MB       2.11A   -----      -----    -----

SMM- 128MB      3.21A   -----      -----    -----

SIO (CX05)
   &  6.21A -----    1.85A    0.06A
MB (CX01)
```